Red Hat OpenStack Platform 17.1

設定の参照

Red Hat OpenStack Platform 環境の設定

Last Updated: 2024-06-12

Red Hat OpenStack Platform 17.1 設定の参照

Red Hat OpenStack Platform 環境の設定

OpenStack Team
rhos-docs@redhat.com

法律上の通知

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

概要

このドキュメントは、設定オプションを調べるシステム管理者を対象としています。これには、OpenStack で利用可能な設定オプションのリストが含まれており、自動生成を使用して各プロジェクトのコードからのオプションおよび説明を生成します。

目次

前書き 3

多様性を受け入れるオープンソースの強化 4

RED HAT ドキュメントへのフィードバック (英語のみ) 5

第1章 BARBICAN 6
1.1. BARBICAN.CONF 6

第2章 CINDER 38
2.1. CINDER.CONF 38

第3章 GLANCE 150
3.1. GLANCE-API.CONF 150
3.2. GLANCE-SCRUBBER.CONF 300
3.3. GLANCE-CACHE.CONF 370

第4章 HEAT 439
4.1. HEAT.CONF 439

第5章 IRONIC 488
5.1. IRONIC.CONF 488

第6章 IRONIC-INSPECTOR 582
6.1. INSPECTOR.CONF 582

第7章 KEYSTONE 609
7.1. KEYSTONE.CONF 609

第8章 NEUTRON 672
8.1. DHCP_AGENT.INI 672
8.2. L3_AGENT.INI 680
8.3. LINUXBRIDGE_AGENT.INI 689
8.4. METADATA_AGENT.INI 697
8.5. METERING_AGENT.INI 706
8.6. ML2_CONF.INI 712
8.7. NEUTRON.CONF 721
8.8. OPENVSWITCH_AGENT.INI 763
8.9. SRIOV_AGENT.INI 772

第9章 NOVA 779
9.1. NOVA.CONF 779

前書き

本書では、Red Hat OpenStack Platform の主要なサービスごとに、設定ファイルで利用可能なオプションを説明します。コンテンツは設定ファイル自体の値に基づいて自動生成され、参照の目的でのみ提供されます。

警告

設定ファイルを手動で編集することはサポートされていません。すべての設定変更は、director を介して行う必要があります。Red Hat は、このガイドを技術リファレンスとしてのみ提供しています。

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージ をご覧ください。

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに対するご意見をお聞かせください。ドキュメントの改善点があればお知らせください。

Jira でドキュメントのフィードバックを提供する

ドキュメントに関するフィードバックを提供するには、Create Issue フォームを使用します。Red Hat OpenStack Platform Jira プロジェクトで Jira Issue が作成され、フィードバックの進行状況を追跡できます。

1. Jira にログインしていることを確認してください。Jira アカウントをお持ちでない場合は、アカウントを作成してフィードバックを送信してください。
2. Create Issue をクリックして、**Create Issue** ページを開きます。
3. **Summary** フィールドと **Description** フィールドに入力します。**Description** フィールドに、ドキュメントの URL、章またはセクション番号、および問題の詳しい説明を入力します。フォーム内の他のフィールドは変更しないでください。
4. **Create** をクリックします。

第1章 BARBICAN

以下の章では、**barbican** サービスの設定オプションについて説明します。

1.1. BARBICAN.CONF

このセクションには、**/etc/barbican/barbican.conf** ファイルのオプションが含まれます。

1.1.1. DEFAULT

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[DEFAULT]** グループで利用可能なオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
admin_role = admin	文字列値	認証されたユーザーを管理者として識別するために使用されるロール。
allow_anonymous_access = False	ブール値	認証されていないユーザーが、読み取り専用の権限で API にアクセスできるようにします。これは、ContextMiddleware を使用する場合にのみ適用されます。
api_paste_config = api-paste.ini	文字列値	api サービスの paste.deploy 設定のファイル名
backdoor_port = None	文字列値	eventlet バックドアを有効にします。設定可能な値は 0、<port>、および <start>:<end> です。ここで、0 はランダムの tcp ポート番号でリッスンします。<port> は指定されたポート番号でリッスンします (そのポートが使用中の場合にはバックドアを有効にしません)。また、<start>:<end> は、指定されたポート番号の範囲で最も小さい未使用のポート番号でリッスンします。選択したポートは、サービスのログファイルに表示されます。
backdoor_socket = None	文字列値	指定されたパスを接続を受信できる unix ソケットとして使用して、eventlet バックドアを有効にします。このオプションは、**backdoor_port** と相互に排他的です。したがって、どちらか 1 つだけを指定する必要があります。両方を指定すると、このオプションの存在によってそのオプションの使用が上書きされます。パス {pid} の中は、現在のプロセスの PID で置き換えられます。
client_socket_timeout = 900	整数値	クライアント接続のソケット操作のタイムアウト。受信接続がこの秒数の間アイドル状態であると、閉じられます。0 の値は、永久に待機することを意味します。

設定オプション = デフォルト値	型	説明
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
control_exchange = openstack	文字列値	トピックがスコープ設定されるデフォルトの変換。transport_url オプションで指定した変換名で上書きできます。
db_auto_create = False	ブール値	サービスの起動時に Barbican データベースを作成します。
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。
default_limit_paging = 10	整数値	**limit** ページング URL パラメーターのデフォルトページサイズ。

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
executor_thread_pool_size = 64	整数値	エグゼキューターがスレッディングまたはイベントレットの場合のエグゼキュータースレッドプールのサイズ。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
host_href = http://localhost:9311	文字列値	HATEOAS 形式の参照に使用するホスト名。注記: 通常これは、クライアントがこのサービスへの返信に使用する負荷分散エンドポイントになります。デプロイメントが wsgi 要求からホストを派生する場合は、これを空白のままにします。空白は、デフォルトの設定値 http://localhost:9311を上書きするために必要です。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。

設定オプション = デフォルト値	型	説明
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_allowed_request_size_in_bytes = 25000	整数値	barbican-api に対する最大許容 http 要求サイズ。
max_allowed_secret_in_bytes = 20000	整数値	シークレットの最大許容サイズ (バイト単位)。
max_header_line = 16384	整数値	許可されるメッセージヘッダー行の最大サイズ。max_header_line は、大きなトークンを使用する際に増やす必要がある場合があります (特に、keystone が大きなサービスカタログの PKI トークンを使用するように設定された場合に生成されるトークン)。
max_limit_paging = 100	整数値	**limit** ページング URL パラメーターの最大ページサイズ。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。

設定オプション = デフォルト値	型	説明
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
rpc_conn_pool_size = 30	整数値	RPC 接続プールのサイズ。
rpc_ping_enabled = False	ブール値	ping 呼び出しに応答するエンドポイントを追加します。エンドポイントの名前は oslo_rpc_server_ping です。
rpc_response_timeout = 60	整数値	呼び出しからの応答を待つ秒数。
run_external_periodic_tasks = True	ブール値	一部の定期的なタスクは個別のプロセスで実行できます。ここで実行すべきですか？
sql_connection = sqlite:///barbican.sqlite	文字列値	参照実装レジストリーサーバーの sqlalchemy 接続文字列。いずれかの有効な SQLAlchemy 接続文字列であれば問題ありません。http://www.sqlalchemy.org/docs/05/reference/sqlalchemy/connections.html#sqlalchemy.create_engine を参照してください。注記: 絶対アドレスの場合は、**sqlite:** の後に **////** スラッシュを使用します。
sql_idle_timeout = 3600	整数値	SQLAlchemy がデータベースへの接続を再確立するまでの時間 (秒単位)。MySQL は、デフォルトの 8 時間の **wait_timeout** を使用し、その後アイドル状態の接続を破棄します。これにより、**MySQL Gone Away** 例外が生じる可能性があります。その場合には、この値を下げて、MySQL が接続を切断する前に SQLAlchemy が再接続できるようにすることができます。
sql_max_retries = 60	整数値	起動時のデータベース接続の最大再試行数。再試行回数を無限に指定するには -1 に設定します。

設定オプション = デフォルト値	型	説明
sql_pool_class = QueuePool	文字列値	sqlalchemy.pool モジュールからインポートされたクラスを受け入れ、プール構築の詳細を処理します。コメントアウトされている場合、SQLAlchemy はデータベースダイアレクトに基づいて選択します。その他のオプションは QueuePool(SQLAlchemy が管理する接続の場合) および NullPool(接続の SQLAlchemy 管理が無効な場合) です。詳細は、http://docs.sqlalchemy.org/en/latest/core/pooling.html を参照してください。
sql_pool_logging = False	ブール値	指定した場合に SQLAlchemy プール関連のデバッグ出力をログ (DEBUG ログレベル出力を設定) に表示します。
sql_pool_max_overflow = 10	整数値	SQLAlchemy が使用するプールの最大オーバーフローサイズ。チェックアウトされた接続の数が sql_pool_size で設定されているサイズに達すると、この制限まで追加の接続が返されます。続いて、プールで許可される同時接続の合計数は sql_pool_size + sql_pool_max_overflow になります。オーバーフロー制限がないことを示すために -1 に設定すると、同時接続の合計数には制限が設けられません。SQLAlchemy がデフォルトを選択するのを許可するには、コメントアウトします。
sql_pool_size = 5	整数値	SQLAlchemy によって使用されるプールのサイズ。これは、プールに永続的に保存される接続の最大数です。0 に設定すると、サイズ制限なしを指定できます。プールを無効にするには、代わりに sql_pool_class を指定して NullPool を使用します。SQLAlchemy がデフォルトを選択するのを許可するには、コメントアウトします。
sql_retry_interval = 1	整数値	SQL 接続を開く再試行の間隔。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
tcp_keepidle = 600	整数値	各サーバーソケットの TCP_KEEPIDLE の値を設定します (秒単位)。OS X では対応していません。

設定オプション = デフォルト値	型	説明
transport_url = rabbit://	文字列値	メッセージングバックエンドに接続するためのネットワークアドレスおよびオプションのユーザー認証情報 (URL 形式)。想定される形式は次のとおりです。 driver://[user:pass@]host:port[, [userN:passN@]hostN:portN]/virtual_host?query 例:rabbit://rabbitmq:password@127.0.0.1:5672// URL のフィールドの詳細は、https://docs.openstack.org/oslo.messaging/latest/reference/transport.html で oslo_messaging.TransportURL のドキュメントを参照してください。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
wsgi_default_pool_size = 100	整数値	wsgi で使用される greenthread のプールのサイズ
wsgi_keep_alive = True	ブール値	False の場合は、クライアントのソケット接続を明示的に閉じます。
wsgi_log_format = %(client_ip)s "%(request_line)s" status: %(status_code)s len: %(body_length)s time: %(wall_seconds).7f	文字列値	ログ行を生成するためにテンプレートとして使用される Python 形式の文字列。client_ip、date_time、request_line、status_code、body_length、wall_seconds をこの形式にフォーマットできます。
wsgi_server_debug = False	ブール値	サーバーが 500 エラーで例外トレースバックをクライアントに送信する必要がある場合は true。False の場合、サーバーは空の本文で応答します。

1.1.2. certificate

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[certificate]** グループで利用可能なオプションの概要をまとめたものです。

表1.1 certificate

設定オプション = デフォルト値	型	説明
enabled_certificate_plugins = ['simple_certificate']	多値	読み込む証明書プラグインのリスト。
namespace = barbican.certificate.plugin	文字列値	プラグインを検索する拡張名前空間。

1.1.3. certificate_event

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[certificate_event]** グループで利用可能なオプションの概要をまとめたものです。

表1.2 certificate_event

設定オプション = デフォルト値	型	説明
enabled_certificate_event_plugins = ['simple_certificate_event']	多値	読み込む証明書プラグインのリスト。
namespace = barbican.certificate.event.plugin	文字列値	イベントプラグインを検索する拡張名前空間。

1.1.4. cors

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[cors]** グループで利用可能なオプションの概要をまとめたものです。

表1.3 cors

設定オプション = デフォルト値	型	説明
allow_credentials = True	ブール値	実際の要求にユーザーの認証情報を含めることができることを示します。
allow_headers = ['X-Auth-Token'、 'X-Openstack-Request-Id'、'X-Project-Id'、'X-Identity-Status'、'X-User-Id'、'X-Storage-Token'、'X-Domain-Id'、'X-User-Domain-Id'、'X-Project-Domain-Id'、'X-Roles']	リスト値	実際の要求時に使用されるヘッダーフィールド名を示します。
allow_methods = ['GET'、'PUT'、'POST'、'DELETE'、'PATCH']	リスト値	実際の要求時に使用できるメソッドを示します。
allowed_origin = None	リスト値	このリソースがリクエストの origin"ヘッダーで受信したドメインと共有されるかどうかを示します。形式:"<protocol>://<host>[:<port>]" (行末のスラッシュなし)例:https://horizon.example.com

設定オプション = デフォルト値	型	説明
expose_headers = ['X-Auth-Token'、'X-Openstack-Request-Id'、'X-Project-Id'、'X-Identity-Status'、'X-User-Id'、'X-Storage-Token'、'X-Domain-Id'、'X-User-Domain-Id'、'X-Project-Domain-Id'、'X-Roles']	リスト値	API に安全に公開できるヘッダーを示します。デフォルトは HTTP Simple ヘッダーです。
max_age = 3600	整数値	CORS プリフライトリクエストの最大キャッシュ期間。

1.1.5. crypto

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[crypto]** グループで利用可能なオプションの概要をまとめたものです。

表1.4 crypto

設定オプション = デフォルト値	型	説明
enabled_crypto_plugins = ['simple_crypto']	多値	読み込む crypto プラグインのリスト。
namespace = barbican.crypto.plugin	文字列値	プラグインを検索する拡張名前空間。

1.1.6. dogtag_plugin

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[dogtag_plugin]** グループで利用可能なオプションの概要をまとめたものです。

表1.5 dogtag_plugin

設定オプション = デフォルト値	型	説明
auto_approved_profiles = caServerCert	文字列値	自動承認された登録プロファイルのリスト
ca_expiration_time = 1	整数値	CA エントリーの有効期限が切れる日数
dogtag_host = localhost	文字列値	Dogtag インスタンスのホスト名
dogtag_port = 8443	ポート値	Dogtag インスタンスのポート

設定オプション = デフォルト値	型	説明
nss_db_path = /etc/barbican/alias	文字列値	NSS 証明書データベースへのパス
nss_password = None	文字列値	NSS 証明書データベースのパスワード
pem_path = /etc/barbican/kra_admin_cert.pem	文字列値	認証用 PEM ファイルへのパス
plugin_name = Dogtag KRA	文字列値	ユーザーフレンドリーなプラグイン名
plugin_working_dir = /etc/barbican/dogtag	文字列値	Dogtag プラグイン用の作業ディレクトリー
retries = 3	整数値	シークレットの保存または生成時の再試行回数
simple_cmc_profile = caOtherCert	文字列値	単純な CMC 要求のプロファイル

1.1.7. keystone_authtoken

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[keystone_authtoken]** グループで利用可能なオプションの概要をまとめたものです。

表1.6 keystone_authtoken

設定オプション = デフォルト値	型	説明
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ

設定オプション = デフォルト値	型	説明
auth_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。このオプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。**非推奨:**Queens 以降 *理由:* auth_uri オプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。
auth_version = None	文字列値	Identity API エンドポイントの API バージョン。
cache = None	文字列値	Swift キャッシュオブジェクトが保存される環境キーを要求します。auth_token ミドルウェアを Swift キャッシュと共にデプロイする場合は、このオプションを使用して、ミドルウェアが swift とキャッシングバックエンドを共有するようにします。それ以外の場合は、代わりに **memcached_servers** オプションを使用します。
cafile = None	文字列値	HTTPs 接続の検証時に使用する PEM でエンコードされた認証局。デフォルトはシステム CA です。
certfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
delay_auth_decision = False	ブール値	ミドルウェア内の認可要求を処理せず、承認の決定をダウンストリームの WSGI コンポーネントに委譲します。

設定オプション = デフォルト値	型	説明
enforce_token_bind = permissive	文字列値	トークンバインディングの使用および種別を制御するために使用されます。トークンバインディングのチェックを無効するには、"disabled"に設定します。バインドタイプがサーバーの認識する形式の場合にはバインディング情報を検証し、そうでない場合には無視するには、"permissive"(デフォルト) に設定します。"strict"は"permissive"と類似していますが、バインドタイプが不明な場合にはトークンが拒否されます。"required"の場合は、いずれかの形式のトークンバインディングが必要です。最後に、トークンに指定する必要のあるバインディングメソッドの名前。
http_connect_timeout = None	整数値	Identity API サーバーと通信する際の要求タイムアウト値。
http_request_max_retries = 3	整数値	Identity API サーバーと通信する際に再接続を試行する回数。
include_service_catalog = True	ブール値	(オプション)X-Service-Catalog ヘッダーを設定するかどうかを示します。False の場合、ミドルウェアはトークンの検証時にサービスカタログを要求せず、X-Service-Catalog ヘッダーを設定しません。
insecure = False	ブール値	HTTPS 接続を確認します。
interface = internal	文字列値	Identity API エンドポイントに使用するインターフェイス。有効な値は、"public"、"internal" (デフォルト)、または "admin" です。
keyfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
memcache_pool_conn_get_timeout = 10	整数値	(オプション) プールから memcached クライアント接続を取得するまで操作が待機する秒数。
memcache_pool_dead_retry = 300	整数値	(オプション)memcached サーバーが停止しているとみなされる秒数。この秒数が経過すると再試行されます。
memcache_pool_maxsize = 10	整数値	(オプション) すべての memcached サーバーへのオープン接続の最大合計数。
memcache_pool_socket_timeout = 3	整数値	(オプション)memcached サーバーと通信する際のソケットのタイムアウト (秒単位)。

設定オプション = デフォルト値	型	説明
memcache_pool_unused_timeout = 60	整数値	(オプション)memcached への接続がプール内で未使用の状態を維持する秒数。この秒数が経過すると終了されます。
memcache_secret_key = None	文字列値	(オプション、memcache_security_strategy が定義されている場合には必須) この文字列は鍵の導出に使用されます。
memcache_security_strategy = None	文字列値	(オプション) 定義されている場合は、トークンデータを認証、または認証して暗号化する必要があるかどうかを示します。MAC の場合、キャッシュでトークンデータが認証されます (HMAC を使用)。ENCRYPT の場合、キャッシュでトークンデータが暗号化され、認証されます。値がこれらのオプションのいずれでもない場合や空の場合には、auth_token は初期化時に例外を発生させます。
memcache_use_advanced_pool = False	ブール値	(オプション) 高度な (eventlet に対して安全な) memcached クライアントプールを使用します。高度なプールは python 2.x でのみ動作します。
memcached_servers = None	リスト値	オプションで、キャッシュに使用する memcached サーバーのリストを指定します。未定義のままの場合、トークンは代わりに処理中にキャッシュされます。
region_name = None	文字列値	アイデンティティーサーバーがあるリージョン。
service_token_roles = ['service']	リスト値	サービストークンに存在する必要があるロールの選択。サービストークンは、期限切れのトークンを使用できることを要求できるため、このチェックでは実際のサービスのみがこのトークンを送信するように厳密に制御する必要があります。ここでのロールは ANY チェックとして適用されるため、このリストのロールはすべて存在している必要があります。後方互換性の理由から、現在 allow_expired チェックにのみ影響します。
service_token_roles_required = False	ブール値	後方互換性の理由から、service_token_roles チェックを有効としてパスしない有効なサービストークンをパスさせる必要があります。これを true に設定することが今後のリリースでデフォルトとなり、可能な場合は有効にされる必要があります。
service_type = None	文字列値	サービスカタログに表示されるサービスの名前または種別。これは、制限されたアクセスルールがあるトークンを検証するために使用されます。

設定オプション = デフォルト値	型	説明
token_cache_time = 300	整数値	トークンの検証に過剰な時間を費やすのを防ぐために、ミドルウェアは、設定可能な期間 (秒単位) 中は以前に見たトークンをキャシュします。キャッシュを完全に無効にするには -1 に設定します。
www_authenticate_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。

1.1.8. keystone_notifications

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[keystone_notifications]** グループで利用可能なオプションの概要をまとめたものです。

表1.7 keystone_notifications

設定オプション = デフォルト値	型	説明
allow_requeue = False	ブール値	true は、通知処理エラーが発生した場合に再度キューに入れる機能を有効にします。この機能は、基礎となるトランスポートがこの機能をサポートしている場合にのみ有効にします。
control_exchange = keystone	文字列値	トピックがスコープ設定されるデフォルトの変換。transport_url オプションで指定した変換名で上書きできます。
enable = False	ブール値	true は keystone 通知リスナー機能を有効にします。

設定オプション = デフォルト値	型	説明
pool_name = None	文字列値	通知リスナーのプール名。これを固有の値に設定すると、バービカン通知リスナーは、同じトピックをリッスンしている他のサービスに干渉することなく、トピックからすべてのメッセージの独自のコピーを受信できます。この機能は、一部の oslo.messaging バックエンド (特に rabbitmq) でのみサポートされており、これらのバックエンドでは、barbican の個別の通知トピックの代わりに使用することを推奨します。
thread_pool_size = 10	整数値	通知サーバー処理機能に使用する最大スレッド数を定義します。
topic = notifications	文字列値	Keystone 通知キューのトピック名。この名前は、Keystone デプロイメントの **notification_topics** 設定 (例:notification_topics =notifications) に記載されている値のいずれかに一致させる必要があります。barbican_notificationsMultiple サーバーはトピックをリッスンし、メッセージがラウンドロビン方式でいずれかのサーバーにディスパッチされます。すべての Keystone 通知を受け取るように Barbican サービスが専用の通知キューを持つ必要があるのはこのためです。あるいは、選択した oslo.messaging バックエンドがリスナーのプール (rabbitmq など) をサポートしている場合は、デフォルト以外の **pool_name** オプションを設定することを推奨します。
version = 1.0	文字列値	通知を介して呼び出されるタスクのバージョン

1.1.9. kmip_plugin

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[kmip_plugin]** グループで利用可能なオプションの概要をまとめたものです。

表1.8 kmip_plugin

設定オプション = デフォルト値	型	説明
ca_certs = None	文字列値	連結された認証局証明書へのファイルパス
certfile = None	文字列値	ローカルクライアント証明書へのファイルパス
host = localhost	文字列値	KMIP サーバーのアドレス
keyfile = None	文字列値	ローカルクライアント証明書のキーファイルへのファイルパス

設定オプション = デフォルト値	型	説明
password = None	文字列値	KMIP サーバーとの認証のためのパスワード
pkcs1_only = False	ブール値	非対称鍵の PKCS#1 エンコーディングのみに対応
plugin_name = KMIP HSM	文字列値	ユーザーフレンドリーなプラグイン名
port = 5696	ポート値	KMIP サーバーのポート
ssl_version = PROTOCOL_TLSv1_2	文字列値	SSL バージョン。モジュール ssl の定数にマッピングします。
username = None	文字列値	KMIP サーバーとの認証のためのユーザー名

1.1.10. oslo_messaging_amqp

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[oslo_messaging_amqp]** グループで利用可能なオプションの概要をまとめたものです。

表1.9 oslo_messaging_amqp

設定オプション = デフォルト値	型	説明
addressing_mode = dynamic	文字列値	ドライバーが使用するアドレスモードを示します。許可される値:**legacy** - レガシーのルーティング不可能なアドレス指定を使用します。**routable** - ルーティング可能なアドレスを使用します。**dynamic** - メッセージバスがルーティングをサポートしていない場合はレガシーのアドレスを使用します。それ以外の場合は、ルーティング可能なアドレスを使用します。
anycast_address = anycast	文字列値	コンシューマーのグループに送信する際に、アドレス接頭辞に追加されます。コンシューマー間でラウンドロビン方式で配信される必要のあるメッセージを特定するために、メッセージバスで使用されます。
broadcast_prefix = broadcast	文字列値	すべてのサーバーにブロードキャストする場合に使用されるアドレス接頭辞
connection_retry_backoff = 2	整数値	フェイルオーバーの試みに失敗するたびに、connection_retry_interval をこの秒数だけ増やします。

設定オプション = デフォルト値	型	説明
connection_retry_interval = 1	整数値	再接続を試みる前に一時停止する秒数。
connection_retry_interval_max = 30	整数値	connection_retry_interval + connection_retry_backoff の上限
container_name = None	文字列値	AMQP コンテナーの名前。グローバルで一意でなければなりません。デフォルトは、生成された UUID です。
default_notification_exchange = None	文字列値	通知アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_notification_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **notify**
default_notify_timeout = 30	整数値	送信された通知メッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_reply_retry = 0	整数値	リカバリー可能なエラーが原因で失敗した返信メッセージを再送信する最大試行回数。
default_reply_timeout = 30	整数値	rpc 返信メッセージ配信の期限。
default_rpc_exchange = None	文字列値	RPC アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_rpc_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **rpc**
default_send_timeout = 30	整数値	rpc キャストまたは呼び出しメッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_sender_link_timeout = 600	整数値	アイドル状態の送信者リンクのパージをスケジュールする時間。期限切れ後にリンクの割り当てを解除します。
group_request_prefix = unicast	文字列値	グループの任意のサーバーに送信する際のアドレス接頭辞
idle_timeout = 0	整数値	非アクティブな接続のタイムアウト (秒単位)
link_retry_delay = 10	整数値	リカバリー可能なエラーが原因で失敗した AMQP 1.0 リンクの再接続間に一時停止する時間。

設定オプション = デフォルト値	型	説明
multicast_address = multicast	文字列値	ファンアウトメッセージを送信するときにアドレス接頭辞に追加されます。ファンアウトメッセージを識別するためにメッセージバスによって使用されます。
notify_address_prefix = openstack.org/om/notify	文字列値	すべての生成される通知アドレスのアドレス接頭辞
notify_server_credit = 100	整数値	受信通知メッセージのウィンドウサイズ
pre_settled = ['rpc-cast', 'rpc-reply']	多値	このタイプのメッセージを事前処置状態で送信します。事前処置されたメッセージは、ピアから確認応答を受信しません。注記: 事前処置されたメッセージは、配信に失敗した場合に警告なしに破棄される可能性があります。許可される値:**rpc-call** - RPC 呼び出しを事前処置状態で送信します。**rpc-reply**- RPC 返信を事前処置状態で送信します。**rpc-cast** - RPC キャストを事前処置状態で送信します。**notify** - 通知を事前処置状態で送信します。
pseudo_vhost = True	ブール値	仮想ホスト (qpidd など) をネイティブにサポートしていないメッセージバスの仮想ホストサポートを有効にします。true に設定すると、仮想ホスト名はすべてのメッセージバスアドレスに追加され、結果として仮想ホストごとにプライベート **サブネット** を作成します。メッセージバスが仮想ホストの名前として AMQP 1.0 オープン遂行動詞の **hostname** フィールドを使用して仮想ホストをサポートする場合は False に設定します。
reply_link_credit = 200	整数値	受信する RPC 返信メッセージのウィンドウサイズ。
rpc_address_prefix = openstack.org/om/rpc	文字列値	すべての生成される RPC アドレスのアドレス接頭辞
rpc_server_credit = 100	整数値	受信する RPC リクエストメッセージのウィンドウサイズ。
`sasl_config_dir = `	文字列値	SASL 設定が含まれるディレクトリーへのパス
`sasl_config_name = `	文字列値	設定ファイルの名前 (.conf 接尾辞なし)
`sasl_default_realm = `	文字列値	ユーザー名にレルムが存在しない場合に使用する SASL レルム
`sasl_mechanisms = `	文字列値	許可される SASL メカニズムのスペース区切りリスト

設定オプション = デフォルト値	型	説明
server_request_prefix = exclusive	文字列値	特定のサーバーに送信する時に使用するアドレス接頭辞
ssl = False	ブール値	SSL 経由で接続を試みます。その他の ssl 関連のパラメーターが指定されていない場合、システムの CA バンドルを使用してサーバーの証明書を検証します。
`ssl_ca_file = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_cert_file = `	文字列値	クライアント認証用の自己識別証明書 PEM ファイル
`ssl_key_file = `	文字列値	ssl_cert_file 証明書の署名に使用される秘密鍵 PEM ファイル (オプション)
ssl_key_password = None	文字列値	ssl_key_file を復号するためのパスワード (暗号化されている場合)
ssl_verify_vhost = False	ブール値	デフォルトでは、SSL は、サーバーの証明書の名前が transport_url のホスト名と一致することを確認します。設定によっては、代わりに仮想ホスト名を使用することが望ましい場合があります。たとえば、サーバーが Server Name Indication TLS 拡張 (rfc6066) を使用して、仮想ホストごとの証明書を提供する場合などです。サーバーの SSL 証明書が DNS 名ではなく仮想ホスト名を使用する場合は、ssl_verify_vhost を True に設定します。
trace = False	ブール値	デバッグ: AMQP フレームを標準出力 (stdout) にダンプします。
unicast_address = unicast	文字列値	特定の RPC/通知サーバーに送信する際に、アドレス接頭辞に追加されます。単一の送信先に送信されたメッセージを識別するためにメッセージバスによって使用されます。

1.1.11. oslo_messaging_kafka

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[oslo_messaging_kafka]** グループで利用可能なオプションの概要をまとめたものです。

表1.10 oslo_messaging_kafka

設定オプション = デフォルト値	型	説明
compression_codec = none	文字列値	プロデューサーによって生成されたすべてのデータの圧縮コーデック。設定されていない場合、圧縮は使用されません。この設定に許可される値は、kafka バージョンに依存することに注意してください。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
consumer_group = oslo_messaging_consumer	文字列値	Kafka コンシューマーのグループ ID。あるグループのコンシューマーは、メッセージ消費を調整します。
enable_auto_commit = False	ブール値	非同期コンシューマーコミットを有効にします。
kafka_consumer_timeout = 1.0	浮動小数点の値	Kafka コンシューマーのデフォルトタイムアウト
kafka_max_fetch_bytes = 1048576	整数値	Kafka コンシューマーの最大フェッチバイト
max_poll_records = 500	整数値	ポーリング呼び出しで返されるレコードの最大数
pool_size = 10	整数値	Kafka コンシューマーのプールサイズ
producer_batch_size = 16384	整数値	プロデューサーの非同期送信のバッチサイズ
producer_batch_timeout = 0.0	浮動小数点の値	KafkaProducer バッチ処理の遅延の上限 (秒単位)
sasl_mechanism = PLAIN	文字列値	セキュリティープロトコルが SASL である際のメカニズム
security_protocol = PLAINTEXT	文字列値	ブローカーとの通信に使用されるプロトコル
`ssl_cafile = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_client_cert_file = `	文字列値	認証に使用されるクライアント証明書の PEM ファイル。

設定オプション = デフォルト値	型	説明
`ssl_client_key_file = `	文字列値	認証に使用されるクライアントキーの PEM ファイル。
`ssl_client_key_password = `	文字列値	認証に使用されるクライアントキーパスワードファイル。

1.1.12. oslo_messaging_notifications

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[oslo_messaging_notifications]** グループで利用可能なオプションの概要をまとめたものです。

表1.11 oslo_messaging_notifications

設定オプション = デフォルト値	型	説明
driver = []	多値	通知の送信を処理するドライバー。使用できる値は messaging、messagingv2、routing、log、test、noop です。
retry = -1	整数値	リカバリー可能なエラーが原因で配信に失敗した通知メッセージを再送信する最大試行回数。0 - 再試行なし、-1 - 無限回
topics = ['notifications']	リスト値	OpenStack の通知に使用する AMQP トピック。
transport_url = None	文字列値	通知に使用するメッセージングドライバーを表す URL。設定されていない場合は、RPC に使用されるものと同じ設定にフォールバックします。

1.1.13. oslo_messaging_rabbit

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[oslo_messaging_rabbit]** グループで利用可能なオプションの概要をまとめたものです。

表1.12 oslo_messaging_rabbit

設定オプション = デフォルト値	型	説明
amqp_auto_delete = False	ブール値	AMQP の自動削除キュー。
amqp_durable_queues = False	ブール値	AMQP で永続キューを使用します。

設定オプション = デフォルト値	型	説明
direct_mandatory_flag = True	ブール値	(非推奨) ダイレクト送信のための RabbitMQ 必須フラグを有効/無効にします。直接送信は応答として使用されるため、クライアントキューが存在しない場合には MessageUndeliverable 例外が発生します。MessageUndeliverable 例外はタイムアウトをループして、送信者が回復する可能性が高くなります。このフラグは非推奨になり、この機能を非アクティブ化することはできません。
enable_cancel_on_failover = False	ブール値	x-cancel-on-ha-failover フラグを有効にして、rabbitmq サーバーがキューが停止しているときにコンシューマーをキャンセルし、通知できるようにします。
heartbeat_in_pthread = False	ブール値	デフォルトでは、ネイティブ Python スレッドを介してヘルスチェックハートビートスレッドを実行します。このオプションが False の場合、ヘルスチェックハートビートは親プロセスから実行モデルを継承します。たとえば、親プロセスが eventlet/greenlet を使用して stdlib にモンキーパッチを適用した場合、ハートビートはグリーンスレッドを介して実行されます。This option should be set to True only for the wsgi services.
heartbeat_rate = 2	整数値	heartbeat_timeout_threshold 中、ハートビートを確認する回数。
heartbeat_timeout_threshold = 60	整数値	ハートビートの keep-alive が失敗した場合に Rabbit ブローカーがダウンとみなされるまでの秒数 (0 はハートビートを無効にします)。
kombu_compression = None	文字列値	実験的用途: 許容値は gzip、bz2 です。設定されていない場合、圧縮は使用されません。このオプションは、今後のバージョンで利用できない可能性があります。
kombu_failover_strategy = round-robin	文字列値	現在接続しているノードが利用できなくなった場合に、次の RabbitMQ ノードを選択する方法を指定します。設定で複数の RabbitMQ ノードが指定される場合に限り有効になります。
kombu_missing_consumer_retry_timeout = 60	整数値	応答のないクライアントが応答を送信するのを待つ時間。この時間が経過すると無視します。この値は rpc_response_timeout より長くすることはできません。

設定オプション = デフォルト値	型	説明
kombu_reconnect_delay = 1.0	浮動小数点の値	AMQP コンシューマーの取り消し通知への応答で、再接続するまでの待機時間。
rabbit_ha_queues = False	ブール値	RabbitMQ (x-ha-policy: all) の HA キューの使用を試みます。このオプションを変更する場合は、RabbitMQ データベースを消去する必要があります。RabbitMQ 3.0 では、キューを宣言する際に x-ha-policy 引数によってキューのミラーリングが制御されなくなりました。すべてのキュー (自動生成された名前のキューを除く) がすべてのノードでミラーリングされるようにするには、"rabbitmqctl set_policy HA **^(?!amq\.).* {"ha-mode": "all"}**" を実行します。
rabbit_interval_max = 30	整数値	RabbitMQ 接続を再試行する最大間隔。デフォルトは 30 秒です。
rabbit_login_method = AMQPLAIN	文字列値	RabbitMQ ログイン方法
rabbit_qos_prefetch_count = 0	整数値	事前フェッチするメッセージの数を指定します。ゼロに設定すると、無制限のメッセージのフェッチが許可されます。
rabbit_retry_backoff = 2	整数値	RabbitMQ に接続する際に再試行間でバックオフする長さ
rabbit_retry_interval = 1	整数値	RabbitMQ との接続を再試行する頻度。
rabbit_transient_queues_ttl = 1800	整数値	キューの TTL (x-expires) の期間 (秒単位) を表す正の整数。TTL の期間使用されないキューは自動的に削除されます。このパラメーターは応答キューとファンアウトキューにのみ影響します。
ssl = False	ブール値	SSL 経由で接続します。
`ssl_ca_file = `	文字列値	SSL 認証局ファイル (SSL が有効な場合にのみ有効)。
`ssl_cert_file = `	文字列値	SSL 証明書ファイル (SSL が有効な場合にのみ有効)。
`ssl_key_file = `	文字列値	SSL キーファイル (SSL が有効な場合にのみ有効)。

設定オプション = デフォルト値	型	説明
`ssl_version = `	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

1.1.14. oslo_middleware

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[oslo_middleware]** グループで利用可能なオプションの概要をまとめたものです。

表1.13 oslo_middleware

設定オプション = デフォルト値	型	説明
enable_proxy_headers_parsing = False	ブール値	アプリケーションがプロキシーの背後であるかどうか。これにより、ミドルウェアがヘッダーを解析すべきかどうかを決定します。

1.1.15. oslo_policy

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[oslo_policy]** グループで利用可能なオプションの概要をまとめたものです。

表1.14 oslo_policy

設定オプション = デフォルト値	型	説明
enforce_new_defaults = False	ブール値	このオプションは、ポリシーを評価するときに古い非推奨のデフォルトを使用するかどうかを制御します。**True** の場合、古い非推奨のデフォルトは評価されません。これは、既存のトークンが古いデフォルトで許可されているが、新しいデフォルトで許可されていない場合、許可されないことを意味します。新しいデフォルトと **scope_type** の利点を一緒に得ることができるように、このフラグを **enforce_scope** フラグと一緒に有効にすることを推奨します
enforce_scope = False	ブール値	このオプションは、ポリシーを評価する際にスコープを適用するかどうかを制御します。**True** の場合、リクエストで使用されるトークンのスコープは、適用されるポリシーの **scope_types** と比較されます。スコープが一致しない場合、**InvalidScope** 例外が発生します。**False** の場合、ポリシーが一致しないスコープで呼び出されていることを運用者に通知するメッセージが記録されます。

設定オプション = デフォルト値	型	説明
policy_default_rule = default	文字列値	デフォルトルール。要求されたルールが見つからない場合に適用されます。
policy_dirs = ['policy.d']	多値	ポリシー設定ファイルが保存されるディレクトリー。これは、config_dir オプションで定義される検索パス内の任意のディレクトリーへの相対パスか、絶対パスにすることができます。このディレクトリーの検索には、policy_file で定義されたファイルが存在している必要があります。存在しないディレクトリーまたは空のディレクトリーは無視されます。
policy_file = policy.yaml	文字列値	ロールを指定のサービスのパーミッションにマッピングするファイルの相対パスまたは絶対パス。相対パスは、このオプションを設定する設定ファイルに対する相対パスとして指定する必要があります。
remote_content_type = application/x-www-form-urlencoded	文字列値	REST ベースのポリシーチェックのデータを送受信するコンテンツ種別
remote_ssl_ca_crt_file = None	文字列値	REST ベースのポリシーチェック用の認証局証明書ファイルの絶対パス
remote_ssl_client_crt_file = None	文字列値	REST ベースのポリシーチェック用のクライアント証明書の絶対パス
remote_ssl_client_key_file = None	文字列値	REST ベースのポリシーチェック用のクライアント鍵ファイルの絶対パス
remote_ssl_verify_server_crt = False	ブール値	REST ベースのポリシーチェックのサーバーアイデンティティー検証

1.1.16. p11_crypto_plugin

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[p11_crypto_plugin]** グループで利用可能なオプションの概要をまとめたものです。

表1.15 p11_crypto_plugin

設定オプション = デフォルト値	型	説明
aes_gcm_generate_iv = True	ブール値	CKM_AES_GCM メカニズムの IV を生成します。
always_set_cka_sensitive = True	ブール値	CKA_EXTRACTABLE=CK_TRUE キーを含め、CKA_SENSITIVE=CK_TRUE を常に設定します。

設定オプション = デフォルト値	型	説明
encryption_mechanism = CKM_AES_CBC	文字列値	シークレット暗号化メカニズム
hmac_key_type = CKK_AES	文字列値	HMAC 鍵の種別
hmac_keygen_mechanism = CKM_AES_KEY_GEN	文字列値	HMAC 鍵生成アルゴリズム
hmac_keywrap_mechanism = CKM_SHA256_HMAC	文字列値	HMAC 鍵ラッピングメカニズム
hmac_label = None	文字列値	マスター HMAC 鍵ラベル (HSM に保存されている)
library_path = None	文字列値	ベンダー PKCS11 ライブラリーへのパス
login = None	文字列値	PKCS11 セッションにログインするためのパスワード
mkek_label = None	文字列値	マスター KEK ラベル (HSM に保存される)
mkek_length = None	整数値	マスター KEK の長さ (バイト単位)。
os_locking_ok = False	ブール値	PKCS#11 クライアントライブラリーを初期化する際に、CKF_OS_LOCKING_OK フラグを有効にします。
pkek_cache_limit = 100	整数値	プロジェクト KEK キャッシュ項目の制限
pkek_cache_ttl = 900	整数値	Project KEK キャッシュの存続期間 (秒単位)
pkek_length = 32	整数値	プロジェクト KEK の長さ (バイト単位)。
plugin_name = PKCS11 HSM	文字列値	ユーザーフレンドリーなプラグイン名
rw_session = True	ブール値	読み取り/書き込み可能なセッションのフラグ
`seed_file = `	文字列値	RNG のシード用エントロピーをプルするためのファイル
seed_length = 32	整数値	シードのためにファイルから読み取るデータの量
slot_id = 1	整数値	(オプション) 使用するトークンデバイスが含まれる HSM スロット ID。

設定オプション = デフォルト値	型	説明
token_label = None	文字列値	非推奨: 代わりに token_labels を使用します。使用するトークンを識別するために使用されるトークンラベル。
token_labels = None	リスト値	使用する 1 つ以上のトークンのラベルのリスト。これは通常 1 つのラベルですが、一部の HSM デバイスには負荷分散または高可用性設定用に複数のラベルが必要な場合があります。
token_serial_number = None	文字列値	使用するトークンを識別するために使用されるトークンのシリアルナンバー。

1.1.17. queue

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[queue]** グループで利用可能なオプションの概要をまとめたものです。

表1.16 queue

設定オプション = デフォルト値	型	説明
asynchronous_workers = 1	整数値	非同期ワーカープロセスの数
enable = False	ブール値	true はキューを有効にし、False はワーカーを同期して呼び出します。
namespace = barbican	文字列値	キューの名前空間
server_name = barbican.queue	文字列値	RPC タスク処理サーバーのサーバー名
topic = barbican.workers	文字列値	キュートピック名
version = 1.1	文字列値	キューを介して呼び出されるタスクのバージョン

1.1.18. quotas

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[quotas]** グループで利用可能なオプションの概要をまとめたものです。

表1.17 quotas

設定オプション = デフォルト値	型	説明
quota_cas = -1	整数値	プロジェクトごとに許可される CA の数
quota_consumers = -1	整数値	プロジェクトごとに許可されるコンシューマーの数
quota_containers = -1	整数値	プロジェクトごとに許可されるコンテナー数
quota_orders = -1	整数値	プロジェクトごとに許可されるオーダーの数
quota_secrets = -1	整数値	プロジェクトごとに許可されるシークレットの数

1.1.19. retry_scheduler

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[retry_scheduler]** グループで利用可能なオプションの概要をまとめたものです。

表1.18 retry_scheduler

設定オプション = デフォルト値	型	説明
initial_delay_seconds = 10.0	浮動小数点の値	再試行スケジューラーを起動するまで待機する時間 (浮動小数点)
periodic_interval_max_seconds = 10.0	浮動小数点の値	定期的なスケジュールイベント間で待機する秒数 (浮動小数点)

1.1.20. secretstore

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[secretstore]** グループで利用可能なオプションの概要をまとめたものです。

表1.19 secretstore

設定オプション = デフォルト値	型	説明
enable_multiple_secret_stores = False	ブール値	複数のシークレットストアプラグインバックエンドのサポートを有効にするフラグ。デフォルトは False です。
enabled_secretstore_plugins = ['store_crypto']	多値	読み込むシークレットストアプラグインのリスト。
namespace = barbican.secretstore.plugin	文字列値	プラグインを検索する拡張名前空間。

設定オプション = デフォルト値	型	説明
stores_lookup_suffix = None	リスト値	複数のバックエンドサポートでサポートされるプラグインの検索に使用する接尾辞のリスト。

1.1.21. simple_crypto_plugin

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[simple_crypto_plugin]** グループで利用可能なオプションの概要をまとめたものです。

表1.20 simple_crypto_plugin

設定オプション = デフォルト値	型	説明
kek = dGhpcnR5X3R3b19ieXRlX2tleWJsYWhibGFoYmxhaGg=	文字列値	Simple Crypto プラグインによって使用されるキー暗号化キー
plugin_name = Software Only Crypto	文字列値	ユーザーフレンドリーなプラグイン名

1.1.22. snakeoil_ca_plugin

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[snakeoil_ca_plugin]** グループで利用可能なオプションの概要をまとめたものです。

表1.21 snakeoil_ca_plugin

設定オプション = デフォルト値	型	説明
ca_cert_chain_path = None	文字列値	CA 証明書チェーンファイルへのパス
ca_cert_key_path = None	文字列値	CA 証明書キーファイルへのパス
ca_cert_path = None	文字列値	CA 証明書ファイルへのパス
ca_cert_pkcs7_path = None	文字列値	CA チェーン pkcs7 ファイルへのパス
subca_cert_key_directory = /etc/barbican/snakeoil-cas	文字列値	サブキャスの証明書/キーを保存するディレクトリー

1.1.23. ssl

以下の表は、**/etc/barbican/barbican.conf** ファイルの **[ssl]** グループで利用可能なオプションの概要をまとめたものです。

表1.22 ssl

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	接続クライアントの検証に使用する CA 証明書ファイル。
cert_file = None	文字列値	サーバーをセキュアに起動する際に使用する証明書ファイル。
ciphers = None	文字列値	利用可能な暗号のリストを設定します。値は OpenSSL 暗号リスト形式の文字列である必要があります。
key_file = None	文字列値	サーバーをセキュアに起動する際に使用する秘密鍵ファイル。
version = None	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

第2章 CINDER

以下の章では、**cinder** サービスの設定オプションについて説明します。

2.1. CINDER.CONF

このセクションには、**/etc/cinder/cinder.conf** ファイルのオプションが含まれます。

2.1.1. DEFAULT

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[DEFAULT]** グループで利用可能なオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
acs5000_copy_interval = 5	整数値	ボリュームコピータスクが進行中の場合、ボリュームステータスの更新間隔
acs5000_volpool_name = ['pool01']	リスト値	ボリュームのストレージシステムストレージプールのコンマ区切りリスト。
allocated_capacity_weight_multiplier = -1.0	浮動小数点の値	割り当てられた容量を重み付けするのに使用する重みの乗数。正の数字は分散ではなくスタックを意味します。
allow_availability_zone_fallback = False	ブール値	要求された Cinder アベイラビリティーゾーンが利用できない場合は、失敗するのではなく、default_availability_zone、続いて storage_availability_zone の値にフォールバックします。
allow_compression_on_image_upload = False	ブール値	アップロード時にイメージ圧縮に使用するストラテジー。デフォルトは圧縮を無効にします。
allowed_direct_url_schemes = []	リスト値	direct_url で直接ダウンロードできる URL スキームのリスト。現在サポートされているスキーム:[file、cinder]
api_paste_config = api-paste.ini	文字列値	api サービスの paste.deploy 設定のファイル名
api_rate_limit = True	ブール値	API のレート制限を有効または無効にします。
as13000_ipsan_pools = ['Pool0']	リスト値	Cinder が使用すべきストレージプールのコンマ区切りリスト。

設定オプション = デフォルト値	型	説明
as13000_meta_pool = None	文字列値	ボリュームの作成時にメタプールとして使用されるプールで、現在そのプールはレプリケーションプールである必要があります。設定しないと、ドライバーは as13000_ipsan_pools の値からレプリケーションプールを選択します。
as13000_token_available_time = 3300	整数値	トークン検証の有効期間 (秒単位)。
auth_strategy = keystone	文字列値	認証に使用するストラテジー。noauth または keystone に対応します。
az_cache_duration = 3600	整数値	指定された期間ボリュームアベイラビリティーゾーンをメモリーにキャッシュします (秒単位)。
backdoor_port = None	文字列値	eventlet バックドアを有効にします。設定可能な値は 0、<port>、および <start>:<end> です。ここで、0 はランダムの tcp ポート番号でリッスンします。<port> は指定されたポート番号でリッスンします (そのポートが使用中の場合にはバックドアを有効にしません)。また、<start>:<end> は、指定されたポート番号の範囲で最も小さい未使用のポート番号でリッスンします。選択したポートは、サービスのログファイルに表示されます。
backdoor_socket = None	文字列値	指定されたパスを接続を受信できる unix ソケットとして使用して、eventlet バックドアを有効にします。このオプションは、**backdoor_port** と相互に排他的です。したがって、どちらか 1 つだけを指定する必要があります。両方を指定すると、このオプションの存在によってそのオプションの使用が上書きされます。パス {pid} の中は、現在のプロセスの PID で置き換えられます。
backend_availability_zone = None	文字列値	このボリュームバックエンドのアベイラビリティーゾーン。設定されていない場合、storage_availability_zone オプション値がすべてのバックエンドのデフォルトとして使用されます。
backend_stats_polling_interval = 60	整数値	バックエンドからの使用状況の統計を要求する間隔 (秒単位)。使用状況の統計の生成は一部のバックエンドには負荷がかかるため、この値を低く設定しすぎるとパフォーマンスに悪影響を与える可能性があることに注意してください。
backup_api_class = cinder.backup.api.API	文字列値	ボリュームバックアップ API クラスの完全なクラス名

設定オプション = デフォルト値	型	説明
backup_ceph_chunk_size = 134217728	整数値	Ceph オブジェクトストアに転送する前にバックアップが分割される際のチャンクサイズ (バイト単位)。
backup_ceph_conf = /etc/ceph/ceph.conf	文字列値	使用する Ceph 設定ファイル。
backup_ceph_image_journals = False	ブール値	True の場合には、JOURNALING および EXCLUSIVE_LOCK 機能ビットをバックアップ RBD オブジェクトに適用し、ミラーリングを許可します。
backup_ceph_pool = backups	文字列値	ボリュームのバックアップが保存される Ceph プール。
backup_ceph_stripe_count = 0	整数値	バックアップイメージの作成時に使用する RBD ストライプの数。
backup_ceph_stripe_unit = 0	整数値	バックアップイメージの作成時に使用する RBD ストライプユニット。
backup_ceph_user = cinder	文字列値	接続する Ceph ユーザー。デフォルトでは Cinder ボリュームと同じユーザーを使用します。cephx を使用していない場合は、これを None に設定する必要があります。
backup_compression_algorithm = zlib	文字列値	圧縮アルゴリズム (無効にする場合は none)
backup_container = None	文字列値	バックアップに使用するカスタムディレクトリー。
backup_driver = cinder.backup.drivers.swift.SwiftBackupDriver	文字列値	バックアップに使用するドライバー。
backup_driver_init_check_interval = 60	整数値	ドライバーの再起動時に、バックアップドライバーが正常に初期化されているかどうかを確認する間隔 (秒単位)。
backup_driver_stats_polling_interval = 60	整数値	バックアップドライバーのステータスをチェックする間隔 (秒単位)。機能していると報告されない場合、再起動されます。
backup_enable_progress_timer = True	ブール値	ボリュームをバックエンドストレージにバックアップする際に、Ceilometer に定期的な進捗通知を送信するタイマーを有効または無効にします。デフォルト値は True で、タイマーを有効にします。

設定オプション = デフォルト値	型	説明
backup_file_size = 1999994880	整数値	バックアップを保持するために使用されるファイルの最大サイズ (バイト単位)。バックアップされるボリュームがこのサイズを超える場合、これは複数のファイルにバックアップされます。backup_sha_file_size は、backup_sha_block_size_bytes の倍数でなければなりません。
backup_manager = cinder.backup.manager.BackupManager	文字列値	ボリュームバックアップ用の Manager の完全なクラス名
backup_max_operations = 15	整数値	メモリーを大量に消費する同時操作の最大数: バックアップと復元。値 0 は無制限を意味します。
backup_metadata_version = 2	整数値	ボリュームメタデータのバックアップ時に使用するバックアップメタデータバージョン。この数字を変更する場合は、復元を行うサービスが新しいバージョンをサポートすることを確認してください。
backup_mount_attempts = 3	整数値	エラーが発生するまで、NFS 共有のマウントを試行する回数。
backup_mount_options = None	文字列値	NFS クライアントに渡されるマウントオプション。詳細は、NFS の man ページを参照してください。
backup_mount_point_base = $state_path/backup_mount	文字列値	NFS 共有のマウントポイントが含まれるベースディレクトリー。
backup_name_template = backup-%s	文字列値	バックアップ名の生成に使用するテンプレート文字列
backup_native_threads_pool_size = 60	整数値	バックアップ用のネイティブスレッドプールのサイズ。ほとんどのバックアップドライバーは、これに大きく依存しています。依存しない特定のドライバーの場合には、値を減らすことができます。
backup_object_number_per_notification = 10	整数値	1 つの Ceilometer 通知が送信されるチャンクまたはオブジェクトの数。
backup_posix_path = $state_path/backup	文字列値	バックアップの保存先を指定するパス。

設定オプション = デフォルト値	型	説明
backup_s3_block_size = 32768	整数値	増分バックアップで追跡される変更のサイズ (バイト単位)。backup_s3_object_size は、backup_s3_block_size の倍数である必要があります。
backup_s3_ca_cert_file = None	文字列値	path/to/cert/bundle.pem - 使用する CA 証明書バンドルのファイル名。
backup_s3_enable_progress_timer = True	ブール値	ボリュームを S3 バックエンドストレージにバックアップするときに、定期的な進捗通知を Ceilometer に送信するタイマーを有効または無効にします。デフォルト値は True で、タイマーを有効にします。
backup_s3_endpoint_url = None	文字列値	S3 サーバーがリッスンしている URL。
`backup_s3_http_proxy = `	文字列値	http プロキシーサーバーのアドレスまたはホスト。
`backup_s3_https_proxy = `	文字列値	https プロキシーサーバーのアドレスまたはホスト。
backup_s3_max_pool_connections = 10	整数値	接続プールに保持する接続の最大数。
backup_s3_md5_validation = True	ブール値	s3 バックエンドで md5 検証を有効または無効にします。
backup_s3_object_size = 52428800	整数値	S3 バックアップオブジェクトのバイト単位のサイズ
backup_s3_retry_max_attempts = 4	整数値	1 回の要求で行われる再試行の最大回数を表す整数。
backup_s3_retry_mode = legacy	文字列値	再試行モードのタイプを表す文字列。例: レガシー、標準、適応型
backup_s3_sse_customer_algorithm = None	文字列値	SSECustomerAlgorithm。SSE を有効にするには、backup_s3_sse_customer_key を同時に設定する必要があります。
backup_s3_sse_customer_key = None	文字列値	SSECustomerKey。SSE を有効にするには、backup_s3_sse_customer_algorithm を同時に設定する必要があります。
backup_s3_store_access_key = None	文字列値	S3 クエリートークンアクセスキー。

設定オプション = デフォルト値	型	説明
backup_s3_store_bucket = volumebackups	文字列値	Cinder バックアップデータの保存に使用される S3 バケット。
backup_s3_store_secret_key = None	文字列値	S3 クエリートークンの秘密鍵。
backup_s3_timeout = 60	浮動小数点の値	タイムアウト例外が出力されるまでの秒数。
backup_s3_verify_ssl = True	ブール値	SSL 検証を有効または無効にします。
backup_service_inithost_offload = True	ブール値	バックアップサービスの起動時に、保留中であるバックアップの削除をオフロードします。false の場合、バックアップサービスは、保留中のバックアップがすべて削除されるまでダウンしたままになります。
backup_sha_block_size_bytes = 32768	整数値	増分バックアップで変更が追跡されるサイズ (バイト単位)。backup_file_size は、backup_sha_block_size_bytes の倍数でなければなりません。
backup_share = None	文字列値	hostname:path、ipv4addr:path、または "[ipv6addr]:path" 形式の NFS 共有。
backup_swift_auth = per_user	文字列値	Swift 認証メカニズム (per_user または single_user)
backup_swift_auth_insecure = False	ブール値	Swift への SSL 接続を確立する際のサーバー証明書の検証を回避します。
backup_swift_auth_url = None	URI 値	Keystone エンドポイントの URL
backup_swift_auth_version = 1	文字列値	Swift 認証バージョン。auth 1.0 には "1" を、auth 2.0 には "2"、または auth 3.0 の場合は "3" を指定します。
backup_swift_block_size = 32768	整数値	増分バックアップで変更が追跡されるサイズ (バイト単位)。backup_swift_object_size は、backup_swift_block_size の倍数でなければなりません。
backup_swift_ca_cert_file = None	文字列値	swift クライアント要求に使用する CA 証明書ファイルの場所。

設定オプション = デフォルト値	型	説明
backup_swift_container = volumebackups	文字列値	使用するデフォルトの Swift コンテナー
backup_swift_enable_progress_timer = True	ブール値	ボリュームを Swift バックエンドストレージにバックアップする際に、Ceilometer に定期的な進捗通知を送信するタイマーを有効または無効にします。デフォルト値は True で、タイマーを有効にします。
backup_swift_key = None	文字列値	認証用の Swift キー
backup_swift_object_size = 52428800	整数値	Swift バックアップオブジェクトのサイズ (バイト単位)。
backup_swift_project = None	文字列値	swift プロジェクト/アカウント名。auth 3.0 システムへの接続時に必須
backup_swift_project_domain = None	文字列値	Swift プロジェクトドメイン名。auth 3.0 システムへの接続時に必須
backup_swift_retry_attempts = 3	整数値	Swift 操作に対する再試行回数
backup_swift_retry_backoff = 2	整数値	Swift 再試行間のバックオフ時間 (秒単位)
backup_swift_tenant = None	文字列値	swift テナント/アカウント名。auth 2.0 システムへの接続時に必須
backup_swift_url = None	URI 値	Swift エンドポイントの URL
backup_swift_user = None	文字列値	Swift ユーザー名
backup_swift_user_domain = None	文字列値	Swift ユーザードメイン名auth 3.0 システムへの接続時に必須
backup_timer_interval = 120	整数値	バックアップステータスを報告する 2 つの進捗通知の間隔 (秒単位)
backup_use_same_host = False	ブール値	バックアップサービスは、同じバックエンドを使用します。
backup_use_temp_snapshot = False	ブール値	True に設定すると、非中断バックアップを実行するために一時的なスナップショットが作成されます。それ以外の場合は、バックアップを実行するために一時ボリュームのクローンを作成します。

設定オプション = デフォルト値	型	説明
backup_workers = 1	整数値	起動するバックアッププロセスの数。同時バックアップによりパフォーマンスを改善します。
capacity_weight_multiplier = 1.0	浮動小数点の値	空き容量を重み付けするのに使用する重みの乗数。負の数字は分散ではなくスタックを意味します。
`chap_password = `	文字列値	指定した CHAP アカウント名のパスワード。
chap_password_len = 12	整数値	CHAP パスワードのランダム文字列の長さ。
`chap_username = `	文字列値	CHAP ユーザー名
chiscsi_conf = /etc/chelsio-iscsi/chiscsi.conf	文字列値	Chiscsi (CXT) グローバルデフォルト設定ファイル
cinder_internal_tenant_project_id = None	文字列値	Cinder 内部テナントとして使用されるプロジェクトの ID。
cinder_internal_tenant_user_id = None	文字列値	Cinder 内部テナントとしてボリューム操作で使用されるユーザーの ID。
client_socket_timeout = 900	整数値	クライアント接続のソケット操作のタイムアウト。受信接続がこの秒数の間アイドル状態であると、閉じられます。0 の値は、永久に待機することを意味します。
clone_volume_timeout = 680	整数値	クローンボリュームタイムアウト作成の**非推奨化**: 14.0.0 以降 *理由:* FusionStorage cinder ドライバーは Restful メソッドでコードをリファクタリングし、古い CLI モードは廃止されています。そのため、これらの設定項目は使用されなくなりました。
cloned_volume_same_az = True	ブール値	新しいボリュームが、スナップショットまたはソースボリュームと同じ AZ となるようにします。
cluster = None	文字列値	このクラスターの名前。HA アクティブ/アクティブモードで機能するように、同じバックエンド設定を共有するボリュームホストをグループ化するために使用します。
compression_format = gzip	文字列値	イメージのアップロードでのイメージ圧縮形式

設定オプション = デフォルト値	型	説明
compute_api_class = cinder.compute.nova.API	文字列値	使用する Compute API クラスの完全なクラス名
config-dir = ['~/.project/project.conf.d/'、'~/project.conf.d/'、'/etc/project/project.conf.d/'、'/etc/project.conf.d/']	リスト値	***.conf** ファイルをプルする設定ディレクトリーへのパス。個々のオプションがオーバーライドされる場合、予測可能な解析順序を提供するように、このファイルセットはソートされます。セットは、以前の --config-file 引数を介して指定されたファイルの後に解析されます。従ってオーバーライドされたオプションが優先されます。このオプションはコマンドラインから設定する必要があります。
config-file = ['~/.project/project.conf'、'~/project.conf'、'/etc/project/project.conf'、'/etc/project.conf']	不明な値	使用する設定ファイルへのパス。複数の設定ファイルを指定することができます。この場合、後のファイルの値が優先されます。デフォルトは %(default)s です。このオプションはコマンドラインから設定する必要があります。
config_source = []	リスト値	ローカルファイル以外の場所から設定にアクセスするための詳細を提供する設定グループをリスト表示します。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
consistencygroup_api_class = cinder.consistencygroup.api.API	文字列値	consistencygroup API クラスの完全なクラス名
control_exchange = openstack	文字列値	トピックがスコープ設定されるデフォルトの変換。transport_url オプションで指定した変換名で上書きできます。
datera_503_interval = 5	整数値	503 の再試行の間隔
datera_503_timeout = 120	整数値	HTTP 503 再試行メッセージのタイムアウト
datera_api_port = 7717	文字列値	Datera API ポート
datera_api_version = 2.2	文字列値	Datera API バージョン。
datera_debug = False	ブール値	true: 関数引数を設定し、ロギングを返します。

設定オプション = デフォルト値	型	説明
datera_debug_replica_count_override = False	ブール値	デバッグ/テスト用途にのみ使用してください。replica_count を 1 に設定するには True に設定します。
datera_disable_extended_metadata = False	ブール値	Datera バックエンドへの追加のメタデータの送信を無効にするには True に設定します。
datera_disable_profiler = False	ブール値	Datera ドライバーでプロファイリングを無効にするには、True に設定します。
datera_disable_template_override = False	ブール値	テンプレートから作成する際に size 属性の自動テンプレートの上書きを無効にするには、True に設定します。
datera_enable_image_cache = False	ブール値	Datera バックエンドイメージのキャッシュを有効にするには、True に設定します。
datera_image_cache_volume_type_id = None	文字列値	キャッシュされたボリュームに使用する Cinder ボリュームタイプ ID
datera_ldap_server = None	文字列値	LDAP 認証サーバー
datera_tenant_id = None	文字列値	**Map** -→ OpenStack プロジェクト ID を暗黙的に Datera テナント ID にマッピングすると、ボリュームのプロビジョニング中には、Datera テナント ID に設定されず、Datera テナント ID は提供された値になります。-→ Datera テナント ID は指定された値になります。
datera_volume_type_defaults = {}	dict 値	volume-type 設定が指定されていない場合、ここでは volume-type のデフォルト値として使用されます。たとえば、ボリューム種別に何も指定されていない場合には、非常に低い total_iops_max 値を設定して、誤って過剰に消費されないようにすることができます。オプションは次の形式で指定します。以下は使用しません。**DF:** PREFIX: **datera_volume_type_defaults=iops_per_gb:100,bandwidth_per_gb:200...etc**
db_driver = cinder.db	文字列値	データベースアクセスに使用するドライバー
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。

設定オプション = デフォルト値	型	説明
default_availability_zone = None	文字列値	新規ボリュームのデフォルトのアベイラビリティーゾーン。設定されていない場合、storage_availability_zone オプションの値が新規ボリュームのデフォルトとして使用されます。
default_group_type = None	文字列値	使用するデフォルトのグループタイプ
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
default_sandstone_target_ips = []	リスト値	SandStone のデフォルトのターゲット IP。
default_volume_type = __DEFAULT__	文字列値	使用するデフォルトのボリュームタイプ
driver_client_cert = None	文字列値	ドライバーがサポートしている場合、検証用のクライアント証明書へのパス。

設定オプション = デフォルト値	型	説明
driver_client_cert_key = None	文字列値	ドライバーがサポートしている場合、検証用のクライアント証明書キーへのパス。
driver_data_namespace = None	文字列値	保存されるドライバープライベートデータ値の namespace。
driver_ssl_cert_path = None	文字列値	バックエンドの検証に使用される CA_BUNDLE ファイルまたは証明書を持つディレクトリーへのデフォルトパスの指定に使用できます。
driver_ssl_cert_verify = False	ブール値	True に設定すると、http クライアントはバックエンドエンドポイントの SSL 証明書を検証します。
driver_use_ssl = False	ブール値	ドライバーが SSL をサポートしている場合は、バックエンドストレージへの接続に SSL を使用するようドライバーに指示します。
dsware_isthin = False	ブール値	シンストレージ割り当てのフラグ。**非推奨となったバージョン:** 14.0.0 *理由:* FusionStorage cinder ドライバーは Restful メソッドでコードをリファクタリングし、古い CLI モードは廃止されています。そのため、これらの設定項目は使用されなくなりました。
`dsware_manager = `	文字列値	cider-volume 用の fusionstorage manager ip addr**非推奨となったバージョン:** 14.0.0 *理由:* FusionStorage cinder ドライバーは Restful メソッドでコードをリファクタリングし、古い CLI モードは廃止されています。そのため、これらの設定項目は使用されなくなりました。
`dsware_rest_url = `	文字列値	FusionStorage 配列のアドレス。例: "dsware_rest_url=xxx"
`dsware_storage_pools = `	文字列値	FusionStorage アレイのプールのリスト。セミコロン (;) は、ストレージプールの分割に "dsware_storage_pools = xxx1; xxx2; xxx3" でした。
enable_force_upload = False	ブール値	upload_to_image で Force オプションを有効にします。これにより、サポートするバックエンド用に、使用中のボリュームで upload_volume を実行できます。

設定オプション = デフォルト値	型	説明
enable_new_services = True	ブール値	作成時に利用可能なプールに追加するサービス
enable_unsupported_driver = False	ブール値	サポート対象外のドライバーを起動できるようにするには、このパラメーターを True に設定します。作業用の CI システムおよびテストを維持していないドライバーは、CI が再び動作するまで、サポート対象外とマークされます。これはドライバーを非推奨としてマークし、次のリリースで削除される可能性があります。
enable_v2_api = True	ブール値	非推奨: Cinder API の v2 をデプロイします。
enable_v3_api = True	ブール値	Cinder API の v3 をデプロイします。
enabled_backends = None	リスト値	使用するバックエンド名のリストこれらのバックエンド名は、一意の [CONFIG] グループとそのオプションでサポートされる必要があります。
enforce_multipath_for_image_xfer = False	ブール値	これが True に設定されている場合、multipathd が実行されていない場合、イメージ転送用のボリュームのアタッチが中断されます。それ以外の場合は、単一のパスにフォールバックします。このパラメーターは、すべてのバックエンドに共通の設定として、各バックエンドセクションまたは backend_defaults セクションで設定する必要があります。
executor_thread_pool_size = 64	整数値	エグゼキューターがスレッディングまたはイベントレットの場合のエグゼキュータースレッドプールのサイズ。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
filter_function = None	文字列値	ホストのフィルターに使用される式の文字列表現。Cinder スケジューラーがドライバーフィルターを使用するように設定されている場合にのみ使用されます。
`fusionstorageagent = `	文字列値	Fusionstorage agent ip addr range **非推奨となったバージョン:** 14.0.0 *理由:* FusionStorage cinder ドライバーは Restful メソッドでコードをリファクタリングし、古い CLI モードは廃止されています。そのため、これらの設定項目は使用されなくなりました。

設定オプション = デフォルト値	型	説明
glance_api_insecure = False	ブール値	視線に対する安全でない SSL (https) リクエストの実行を許可します (https は使用されますが、証明書の検証は実行されません)。
glance_api_servers = None	リスト値	cinder で利用可能な glance API サーバーの URL のリスト ([http[s]://][hostname|ip]:port)protocol が指定されていない場合、デフォルトは http に設定されます。
glance_api_ssl_compression = False	ブール値	SSL レイヤー圧縮のネゴシエーションを有効または無効にします。圧縮を無効にすると、ネットワーク帯域幅が高く、qcow2 などの圧縮されたイメージ形式を使用する場合など、データのスループットが向上します。
glance_ca_certificates_file = None	文字列値	glance クライアント要求に使用する ca 証明書ファイルの場所。
glance_catalog_info = image:glance:publicURL	文字列値	サービスカタログで glance を検索する際に一致する情報。フォーマットは <service_type>:<service_name>:<endpoint_type> の形式です。glance_api_servers が指定されていない場合のみ使用されます。
glance_certfile = None	文字列値	Glance クライアント要求に使用する証明書ファイルの場所。
glance_core_properties = ['checksum', 'container_format', 'disk_format', 'image_name', 'image_id', 'min_disk', 'min_ram', 'name', 'size']	リスト値	イメージのデフォルトのコアプロパティー
glance_keyfile = None	文字列値	Glance クライアント要求に使用する証明書キーファイルの場所。
glance_num_retries = 3	整数値	glance からイメージをダウンロードする際の再試行回数
glance_request_timeout = None	整数値	glance 操作の HTTP/https タイムアウト値。ここで値 (None) が指定されていない場合には、glanceclient のデフォルト値が使用されます。

設定オプション = デフォルト値	型	説明
glusterfs_backup_mount_point = $state_path/backup_mount	文字列値	gluster 共有のマウントポイントを含むベース dir。
glusterfs_backup_share = None	文字列値	GlusterFS share in <hostname|ipv4addr|ipv6addr>: <gluster_vol_name> format.Eg: 1.2.3.4:backup_vol
goodness_function = None	文字列値	ホストの有効性を決定するために使用される式の文字列表現。Cinder スケジューラーが使用する適切な重み付け関数を使用する場合にのみ使用されます。
graceful_shutdown_timeout = 60	整数値	サーバーを正常にシャットダウンするまでのタイムアウトを指定します。値をゼロにすると、endless の待機を意味します。
group_api_class = cinder.group.api.API	文字列値	グループ API クラスの完全クラス名
hitachi_mirror_auth_password = None	文字列値	iSCSI 認証パスワード
hitachi_mirror_auth_user = None	文字列値	iSCSI 認証のユーザー名
hitachi_mirror_compute_target_ports = []	リスト値	ホストグループまたは iSCSI ターゲットのコンピュートノードのターゲットポート名
hitachi_mirror_ldev_range = None	文字列値	セカンダリーストレージシステムの論理デバイス範囲
hitachi_mirror_pair_target_number = 0	整数値	ホストグループまたは iSCSI ターゲットのペアのターゲット名
hitachi_mirror_pool = None	文字列値	セカンダリーストレージシステムのプール
hitachi_mirror_rest_api_ip = None	文字列値	REST API サーバーの IP アドレス
hitachi_mirror_rest_api_port = 443	ポート値	REST API サーバーのポート番号
hitachi_mirror_rest_pair_target_ports = []	リスト値	ホストグループまたは iSCSI ターゲットのペアのターゲットポート名

設定オプション = デフォルト値	型	説明
hitachi_mirror_rest_password = None	文字列値	REST API のセカンダリーストレージシステムのパスワード
hitachi_mirror_rest_user = None	文字列値	REST API のセカンダリーストレージシステムのユーザー名
hitachi_mirror_snap_pool = None	文字列値	セカンダリーストレージシステムのシンプール
hitachi_mirror_ssl_cert_path = None	文字列値	バックエンドの検証に使用される CA_BUNDLE ファイルまたは証明書を持つディレクトリーへのデフォルトパスの指定に使用できます。
hitachi_mirror_ssl_cert_verify = False	ブール値	True に設定すると、http クライアントはバックエンドエンドポイントの SSL 証明書を検証します。
hitachi_mirror_storage_id = None	文字列値	セカンダリーストレージシステムの ID
hitachi_mirror_target_ports = []	リスト値	ホストグループまたは iSCSI ターゲットのターゲットポート名
hitachi_mirror_use_chap_auth = False	ブール値	iSCSI 認証を使用するかどうか
hitachi_path_group_id = 0	整数値	リモートレプリケーション用にリモート接続に割り当てられたパスグループ ID
hitachi_quorum_disk_id = None	整数値	グローバル/アクティブデバイスに使用されるクォーラムディスクの ID
hitachi_replication_copy_speed = 3	整数値	ストレージシステムのリモートコピー速度。1 または 2 は低速、3 は中速、4 - 15 は高速を示します。
hitachi_replication_number = 0	整数値	REST API のインスタンス番号
hitachi_replication_status_check_long_interval = 600	整数値	リモートレプリケーションペアのステータスをチェックする間隔。このパラメーターは、このパラメーターで指定された時間が経過してもステータスが、想定のステータスに変わらなかった場合に適用されます。
hitachi_replication_status_check_short_interval = 5	整数値	リモートレプリケーションペアのステータスがチェックされる初期間隔

設定オプション = デフォルト値	型	説明
hitachi_replication_status_check_timeout = 86400	整数値	リモートレプリケーションペアのステータスが想定のステータスに変更されるまでの最大待機時間
hitachi_set_mirror_reserve_attribute = True	ブール値	mirror 予約属性を設定するかどうか
host = <based on operating system>	文字列値	このノードの名前。これは不透明な ID にすることができます。必ずしもホスト名、FQDN、または IP アドレスではありません。
iet_conf = /etc/iet/ietd.conf	文字列値	非推奨: IET 設定ファイル
image_compress_on_upload = True	ブール値	可能な場合は、イメージサービスにアップロードしたイメージを圧縮します。
image_conversion_address_space_limit = 1	整数値	イメージを変換するギガバイト単位のアドレス領域制限
image_conversion_cpu_limit = 60	整数値	イメージを変換するための CPU の制限時間 (秒)
image_conversion_dir = $state_path/conversion	文字列値	イメージの変換中に一時ストレージに使用されるディレクトリー
image_upload_use_cinder_backend = False	ブール値	True に設定すると、raw 形式の upload-to-image は、ボリュームコンテンツをアップロードする代わりに、クローン作成されたボリュームを作成し、その場所をイメージサービスに登録します。Image サービスで cinder バックエンドおよび場所のサポートを有効にする必要があります。
image_upload_use_internal_tenant = False	ブール値	True に設定すると、upload-to-image で作成したイメージボリュームは内部テナントに配置されます。それ以外の場合は、イメージボリュームが現在のコンテキストのテナントに作成されます。
image_volume_cache_enabled = False	ブール値	このバックエンドのイメージボリュームキャッシュを有効にします。
image_volume_cache_max_count = 0	整数値	イメージボリュームキャッシュで許可されるエントリーの最大数。0 ⇒ 無制限
image_volume_cache_max_size_gb = 0	整数値	このバックエンドのイメージボリュームキャッシュの最大サイズ (単位: GB)。0 ⇒ 無制限

設定オプション = デフォルト値	型	説明
infortrend_cli_cache = False	ブール値	Infortrend CLI キャッシュ。True に設定すると、RAID ステータスレポートは CLI に保存されているキャッシュを使用します。RAID が Openstack によってのみ管理され、cinder-volume バックエンドによってのみ管理されていない限り、有効にしないでください。そうでない場合には、CLI が古い状態を cinder に報告する可能性があるため、すべてのバックエンド/CLI に競合状態が生じる場合があります。
infortrend_cli_max_retries = 5	整数値	コマンドが失敗した場合の最大再試行回数。
infortrend_cli_path = /opt/bin/Infortrend/raidcmd_ESDS10.jar	文字列値	Intrend CLI の絶対パス。
infortrend_cli_timeout = 60	整数値	CLI のタイムアウト (秒単位)。
infortrend_iqn_prefix = iqn.2002-10.com.infortrend	文字列値	iSCSI の Infortrend iqn 接頭辞。
`infortrend_pools_name = `	リスト値	Intrend 論理ボリューム名のリスト。これはコンマで区切ります。
`infortrend_slots_a_channels_id = `	リスト値	OpenStack の使用用に Slot A の Infortrend raid チャンネル ID リスト。これはコンマで区切ります。
`infortrend_slots_b_channels_id = `	リスト値	OpenStack の使用用に Slot B の Infortrend raid チャンネル ID リスト。これはコンマで区切ります。
init_host_max_objects_retrieval = 0	整数値	ボリュームマネージャーのホストの初期化中に、バッチごとに取得するボリュームおよびスナップショットの最大数。クエリー結果はデータベースからバッチで取得され、メモリーの使用量が極端になるのを防ぐために 1 つのショットでは実行されません。この機能を無効にするには 0 を設定します。
initiator_assign_sandstone_target_ip = {}	dict 値	assign ip を使用してイニシエータの割り当てターゲットをサポートします。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。

設定オプション = デフォルト値	型	説明
instorage_mcs_allow_tenant_qos = False	ブール値	テナントが作成時に QoS を指定できるようにする
instorage_mcs_iscsi_chap_enabled = True	ブール値	iSCSI 接続に CHAP 認証を設定する (デフォルト: Enabled)。
instorage_mcs_localcopy_rate = 50	整数値	完全なボリュームコピーの作成時に使用される InStorage LocalCopy のコピーレートを指定します。デフォルトのレートは 50 で、有効なレートは 1~100 です。
instorage_mcs_localcopy_timeout = 120	整数値	LocalCopy が準備されるまで待機する最大秒数。
instorage_mcs_vol_autoexpand = True	ブール値	ボリュームのストレージシステムの自動デプロイメントパラメーター (True/False)
instorage_mcs_vol_compression = False	ブール値	ボリュームのストレージシステム圧縮オプション
instorage_mcs_vol_grainsize = 256	整数値	ボリュームのストレージシステムの粒度パラメーター (32/64/128/256)
instorage_mcs_vol_intier = True	ブール値	ボリュームの InTier の有効化
instorage_mcs_vol_iogrp = 0	文字列値	ボリュームを割り当てる I/O グループ。これには、コンマ区切りのリストがあり、その場合は、io_group に関連付けられたボリュームの数が最も少ないボリュームに基づいて io_group を選択します。
instorage_mcs_vol_rsize = 2	整数値	ボリュームに対するストレージシステムの空間効率パラメーター (パーセンテージ)
instorage_mcs_vol_warning = 0	整数値	ボリューム容量の警告に対するストレージシステムのしきい値 (パーセンテージ)
instorage_mcs_volpool_name = ['volpool']	リスト値	ボリュームのストレージシステムストレージプールのコンマ区切りリスト。
instorage_san_secondary_ip = None	文字列値	san_ip が無効であるか、アクセスできなくなる場合に使用されるセカンダリー管理 IP またはホスト名を指定します。

設定オプション = デフォルト値	型	説明
iscsi_iotype = fileio	文字列値	iSCSI ターゲットの動作を、任意で blockio または fileio のいずれかを実行するように設定します。auto を設定すると、Cinder はバッキングデバイスのタイプを自動検出できます。
`iscsi_target_flags = `	文字列値	iSCSI ターゲットにターゲット固有のフラグを設定します。tgtadm で bsoflags オプションを使用してバッキングデバイスフラグを指定するためにのみ使用されます。指定された文字列は、基礎となるツールと同様に渡されます。
iscsi_write_cache = on	文字列値	iSCSI ターゲットの動作を、ライトバックオン (on) または write-through (off) のいずれかを実行するように設定します。このパラメーターは、target_helper が tgtadm に設定されている場合に有効です。
iser_helper = tgtadm	文字列値	使用する iSER ターゲットユーザーランドツールの名前
iser_ip_address = $my_ip	文字列値	iSER デーモンがリッスンする IP アドレス
iser_port = 3260	ポート値	iSER デーモンがリッスンするポート
iser_target_prefix = iqn.2010-10.org.openstack:	文字列値	iSER ボリュームの接頭辞
java_path = /usr/bin/java	文字列値	Java の絶対パス。
jovian_block_size = 64K	文字列値	ブロックサイズ: 32K、64K、128K、256K、512K、1M
jovian_ignore_tpath = []	リスト値	無視するマルチパス IP アドレスのリスト。
jovian_pool = Pool-0	文字列値	すべての cinder ボリュームを保持する JovianDSS プール
jovian_recovery_delay = 60	整数値	HA クラスター障害が発生するまでの時間。
keystone_catalog_info = identity:Identity Service:publicURL	文字列値	サービスカタログで keystone を検索する際に一致する情報形式: <service_type>:<service_name>: <endpoint_type> の形式区切りの値: backup_swift_auth_url が設定されていない場合のみ使用されます。

設定オプション = デフォルト値	型	説明
kioxia_block_size = 4096	整数値	ボリュームブロックサイズ (バイト単位) - 512 または 4096 (デフォルト)。
kioxia_cafile = None	文字列値	プロビジョナー REST API SSL の証明書
kioxia_desired_bw_per_gb = 0	整数値	必要な帯域幅 (B/s/GB)。
kioxia_desired_iops_per_gb = 0	整数値	必要な IOPS/GB。
kioxia_max_bw_per_gb = 0	整数値	GB あたりの B/s での帯域幅の上限。
kioxia_max_iops_per_gb = 0	整数値	IOPS/GB の上限。
kioxia_max_replica_down_time = 0	整数値	レプリカのレプリケートされたボリュームの最大ダウンタイム (分単位)。
kioxia_num_replicas = 1	整数値	ボリュームレプリカの数。
kioxia_provisioning_type = THICK	文字列値	シンまたはシックボリューム、デフォルトのシック。
kioxia_same_rack_allowed = False	ブール値	複数のレプリカを同じラックに割り当てることができます。
kioxia_snap_reserved_space_percentage = 0	整数値	ログに使用される親ボリュームの割合。
kioxia_snap_vol_reserved_space_percentage = 0	整数値	ログに使用される親ボリュームの書き込み可能なスナップショットの割合。
kioxia_snap_vol_span_allowed = True	ブール値	スナップショットボリュームのスパンを許可 - デフォルトは True。
kioxia_span_allowed = True	ブール値	スパンを許可 - デフォルトは True。
kioxia_token = None	文字列値	KumoScale プロビジョナー認証トークン。
kioxia_url = None	文字列値	KumoScale プロビジョナー REST API URL

設定オプション = デフォルト値	型	説明
kioxia_vol_reserved_space_percentage = 0	整数値	シンボリュームの予約容量の割り当ての割合。
kioxia_writable = False	ブール値	スナップショットからのボリュームが書き込み可能かどうか。
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_options = True	ブール値	サービスの起動時 (DEBUG レベル) に、登録されているすべてのオプションの値をログに記録することを有効または無効にします。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。

設定オプション = デフォルト値	型	説明
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
manager_ips = {}	dict 値	このオプションは、異なるノード全体でマウントする FSA をサポートします。パラメーターは、標準的な dict の設定形式を使います: manager_ips = host1:ip1, host2:ip2...
max_age = 0	整数値	次の使用状況の更新までの秒数
max_header_line = 16384	整数値	許可されるメッセージヘッダー行の最大サイズ。max_header_line は、大きなトークンを使用する際に増やす必要がある場合があります (特に、keystone が大きなサービスカタログの PKI トークンを使用するように設定された場合に生成されるトークン)。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。

設定オプション = デフォルト値	型	説明
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
max_over_subscription_ratio = 20.0	文字列値	シンプロビジョニングが有効な場合にサブスクリプションに対する比率を表す。デフォルトの比率は 20.0 です。つまり、プロビジョニングした容量は、物理容量の合計の 20 倍になります。比率が 10.5 の場合は、プロビジョニング容量が物理容量の合計の 10.5 倍になることを意味します。比率が 1.0 の場合は、プロビジョニングした容量が物理容量の合計を超えることができません。比率が **auto** の場合、Cinder はプロビジョニングされた容量と使用領域に基づいて比率を自動的に算出します。auto に設定されていない場合は、比率は 1.0 以上である必要があります。
message_reap_interval = 86400	整数値	期限切れのメッセージをクリーニングするための定期的なタスク実行の間隔を秒単位で指定します。
message_ttl = 2592000	整数値	メッセージの最小寿命を秒単位で表示します。
migration_create_volume_timeout_secs = 300	整数値	ボリュームマイグレーションを行う際に、マイグレーション先のボリュームを作成するためのタイムアウト (秒)
monkey_patch = False	ブール値	モンキーパッチの有効化
monkey_patch_modules = []	リスト値	モンキーパッチの対象となるモジュール/デコレーターのリスト
my_ip = <オペレーティングシステムに基づく>	ホストのアドレス値	このホストの IP アドレス
no_snapshot_gb_quota = False	ブール値	スナップショットがギガバイトクォータに対してカウントされるかどうか
num_iser_scan_tries = 3	整数値	ボリュームを見つけるために i SER ターゲットを再スキャンする最大回数
num_shell_tries = 3	整数値	flakey シェルコマンドの実行を試行する回数
num_volume_device_scan_tries = 3	整数値	ボリュームを検索するためにターゲットを再スキャンする最大回数

設定オプション = デフォルト値	型	説明
nvmeof_conn_info_version = 1	整数値	NVMe os-brick コネクターには 2 種類の接続情報形式があり、これにより、spdk や LVM-nvmet などの元の形式 (バージョン 1) を使用する一部の NVMe-oF ドライバーは、新しい形式を送信できるようになります。
nvmet_ns_id = 10	整数値	ターゲットを共有していない場合の LVM ボリュームのサブシステムのネームスペース ID。共有時の最小 ID 値。Linux でサポートされる最大値は 8192 です。
nvmet_port_id = 1	ポート値	ターゲットを共有していない場合の NVMe ターゲットポート定義の ID。共有時の開始ポート ID 値。セカンダリー IP アドレスごとに増加します。
osapi_max_limit = 1000	整数値	1 つの応答でコレクションリソースが返すアイテムの最大数。
osapi_volume_ext_list = []	リスト値	cinder.api.contrib.select_extensions で osapi_volume_extension オプションを使用する際にロードするエクステンションのリストを指定します。
osapi_volume_extension = ['cinder.api.contrib.standard_extensions']	多値	osapi ボリューム拡張によるロード
osapi_volume_listen = 0.0.0.0	文字列値	OpenStack Volume API がリッスンする IP アドレス
osapi_volume_listen_port = 8776	ポート値	OpenStack Volume API がリッスンするポート
osapi_volume_use_ssl = False	ブール値	True が設定されている場合は、SSL コンテキストでソケットをラップします。証明書ファイルとキーファイルを指定する必要があります。
osapi_volume_workers = None	整数値	OpenStack Volume API サービスのワーカー数。デフォルトは、利用可能な CPU の数と同じです。
per_volume_size_limit = -1	整数値	ボリュームごとに許可される最大サイズ (ギガバイト単位)
periodic_fuzzy_delay = 60	整数値	周期的タスクスケジューラーの起動時に、スタンピングを減らすためにランダムに遅延させる範囲 (秒単位)。(0 に設定すると無効になります)

設定オプション = デフォルト値	型	説明
periodic_interval = 60	整数値	周期的なタスクを実行する際のインターバル (秒単位)
pool_id_filter = []	リスト値	使用するプール ID 許可 **非推奨となったバージョン:** 14.0.0 *理由:* FusionStorage cinder ドライバーは Restful メソッドでコードをリファクタリングし、古い CLI モードは廃止されています。そのため、これらの設定項目は使用されなくなりました。
pool_type = default	文字列値	sata-2copy などのプールタイプ **非推奨となったバージョン:**14.0.0 *理由:* FusionStorage cinder ドライバーは Restful メソッドでコードをリファクタリングし、古い CLI モードは廃止されています。そのため、これらの設定項目は使用されなくなりました。
public_endpoint = None	文字列値	バージョンエンドポイントに使用するパブリック URL。デフォルトは None で、リクエストの host_url 属性を使用して URL ベースを設定します。Cinder がプロキシーの背後で動作している場合は、これをプロキシーの URL を表すように変更する必要があります。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
quota_backup_gigabytes = 1000	整数値	プロジェクトごとのバックアップに許容されるストレージの総量 (単位: ギガバイト)
quota_backups = 10	整数値	プロジェクトごとに許可されるボリュームバックアップの数
quota_consistencygroups = 10	整数値	プロジェクトごとに許可される整合性グループの数
quota_driver = cinder.quota.DbQuotaDriver	文字列値	クォータチェックに使用するデフォルトのドライバー
quota_gigabytes = 1000	整数値	プロジェクトごとのボリュームとスナップショットに許可されるストレージの総量 (ギガバイト)。

設定オプション = デフォルト値	型	説明
quota_groups = 10	整数値	プロジェクトごとに許可されるグループ数
quota_snapshots = 10	整数値	プロジェクトごとに許可されるボリュームスナップショットの数
quota_volumes = 10	整数値	プロジェクトごとに許可されるボリュームの数
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
reinit_driver_count = 3	整数値	ボリュームの初期化に失敗した場合にドライバーを再編成する最大時間。再試行の間隔は指数関数的にバックオフされ、1、2、4 などになります。
replication_device = None	dict 値	レプリケーションターゲットデバイスを表すディクショナリーのマルチオプト。このオプションは、1 つの config セクションで複数回指定して、複数のレプリケーションターゲットデバイスを指定できます。各エントリーは、以下の標準の dict 設定形式を使います: replication_device = target_device_id: <required>,key1:value1,key2:value2...
report_discard_supported = False	ブール値	Cinder のクライアントに、バックエンドが破棄 (別名: trim/unmap) をサポートしていることを報告します。これは実際にバックエンドやクライアントの動作を直接変更するものではなく、使用可能であることを通知するだけです。
report_interval = 10	整数値	ノードがデータストアに状態を報告する間隔 (秒)
reservation_clean_interval = $reservation_expire	整数値	期限切れの予約を削除するための定期的なタスク実行の間隔 (秒)。
reservation_expire = 86400	整数値	予約の有効期限が切れるまでの秒数
reserved_percentage = 0	整数値	バックエンド容量のパーセンテージが予約される

設定オプション = デフォルト値	型	説明
resource_query_filters_file = /etc/cinder/resource_filters.json	文字列値	リストクエリーのユーザーに表示されるフィルターパラメーターを示す JSON ファイル。
restore_discard_excess_bytes = True	ブール値	True の場合には、ゼロで pad などのボリュームを復元する際に、例外を超えるバイトを常に破棄します。
rootwrap_config = /etc/cinder/rootwrap.conf	文字列値	root としてコマンドを実行するために使用する rootwrap 設定ファイルへのパス
rpc_conn_pool_size = 30	整数値	RPC 接続プールのサイズ。
rpc_ping_enabled = False	ブール値	ping 呼び出しに応答するエンドポイントを追加します。エンドポイントの名前は oslo_rpc_server_ping です。
rpc_response_timeout = 60	整数値	呼び出しからの応答を待つ秒数。
run_external_periodic_tasks = True	ブール値	一部の定期的なタスクは個別のプロセスで実行できます。ここで実行すべきですか？
`san_hosts = `	リスト値	Open-E JovianDSS SA の IP アドレス
`sandstone_pool = `	文字列値	SandStone ストレージプールのリソース名。
scheduler_default_filters = ['AvailabilityZoneFilter', 'CapacityFilter', 'CapabilitiesFilter']	リスト値	要求で指定されていない場合に、ホストをフィルターするために使用するフィルタークラス名。
scheduler_default_weighers = ['CapacityWeigher']	リスト値	ホストの重み付けに使用するクラスを重み付けします。
scheduler_driver = cinder.scheduler.filter_scheduler.FilterScheduler	文字列値	使用するデフォルトのスケジューラードライバー
scheduler_driver_init_wait_time = 60	整数値	ドライバーがレディと報告するまでの最大待ち時間 (秒)

設定オプション = デフォルト値	型	説明
scheduler_host_manager = cinder.scheduler.host_manager.HostManager	文字列値	使用するスケジューラーホストマネージャークラス
`scheduler_json_config_location = `	文字列値	スケジューラー設定の JSON ファイルへの絶対パス。
scheduler_manager = cinder.scheduler.manager.SchedulerManager	文字列値	スケジューラー用の Manager のフルクラス名
scheduler_max_attempts = 3	整数値	ボリュームのスケジュールを試行する最大回数
scheduler_weight_handler = cinder.scheduler.weights.OrderedHostWeightHandler	文字列値	重み付け後にホスト/プールを選択するのに使用するハンドラー。
scst_target_driver = iscsi	文字列値	SCST ターゲット実装は、複数の SCST ターゲットドライバーから選択できます。
scst_target_iqn_name = None	文字列値	特定の ISCSI ターゲットには事前に定義されたターゲット名があり、SCST ターゲットドライバーはこの名前を使用します。
service_down_time = 60	整数値	サービスの最後のチェックインからの最大時間
snapshot_name_template = snapshot-%s	文字列値	スナップショット名の生成に使用するテンプレートの文字列
snapshot_same_host = True	ブール値	スナップショットがあるホストでのスナップショットからボリュームの作成
split_loggers = False	ブール値	複数のロガーへの要求をログに記録します。
ssh_hosts_key_file = $state_path/ssh_known_hosts	文字列値	Cinder が通信する必要のあるシステムの SSH ホストキーを含むファイル。オプション: Default=$state_path/ssh_known_hosts
state_path = /var/lib/cinder	文字列値	cinder の状態を維持する最上位のディレクトリー

設定オプション = デフォルト値	型	説明
storage_availability_zone = nova	文字列値	このノードのアベイラビリティーゾーン。ボリュームバックエンドごとに、"backend_availability_zone" オプションで上書きできます。
storage_protocol = iscsi	文字列値	ホストとストレージバックエンド間でデータを転送するためのプロトコル。
strict_ssh_host_key_policy = False	ブール値	厳密なホストキーチェックを有効にするオプション。True に設定すると、Cinder は設定された ssh_hosts_key_file に設定されたホスト鍵を持つシステムにのみ接続します。False に設定すると、ホストキーは最初の接続に保存され、その後の接続に使用されます。Default=False
swift_catalog_info = object-store:swift:publicURL	文字列値	サービスカタログで swift を検索する際に一致する情報。形式: <service_type>:<service_name>:<endpoint_type> の形式区切りの値: backup_swift_url が設定されていない場合にのみ使用されます。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
target_helper = tgtadm	文字列値	使用するターゲットユーザーランドツール。tgtadm はデフォルト で、LIO iSCSI サポートに lioadm、SCST ターゲットのサポートには scstadmin、または Chelsio iSCSI ターゲットの場合は ietadm、NVMEoF サポートの場合は nvmet、SPDK NVMe-oF の場合は spdk-nvmeof、テストには fake を使用します。注: IET ドライバーは推奨されておらず、V リリースで削除される予定です。
target_ip_address = $my_ip	文字列値	iSCSI/NVMEoF デーモンがリッスンする IP アドレス。
target_port = 3260	ポート値	iSCSI/NVMEoF デーモンがリッスンするポート。
target_prefix = iqn.2010-10.org.openstack:	文字列値	iSCSI/NVMEoF ボリュームの接頭辞。
target_protocol = iscsi	文字列値	tgtadm、lioadm、および nvmet ターゲットヘルパーで作成した新規ボリュームのターゲットプロトコルを決定します。RDMA を有効にするには、このパラメーターを "iser" の値で設定する必要があります。nvmet ターゲットが "nvmet_rdma" または "nvmet_tcp" に設定されている場合は、サポートされる iSCSI プロトコルの値は "iscsi" および "iser" です。

設定オプション = デフォルト値	型	説明
target_secondary_ip_addresses = []	リスト値	iSCSI/NVMEoF デーモンのセカンダリー IP アドレスのリスト。
tcp_keepalive = True	ブール値	各サーバーソケットの TCP_KEEPALIVE(True/False) の値を設定します。
tcp_keepalive_count = None	整数値	各サーバーソケットの TCP_KEEPCNT の値を設定します。OS X では対応していません。
tcp_keepalive_interval = None	整数値	各サーバーソケットの TCP_KEEPINTVL の値を秒単位で設定します。OS X では対応していません。
tcp_keepidle = 600	整数値	各サーバーソケットの TCP_KEEPIDLE の値を設定します (秒単位)。OS X では対応していません。
trace_flags = None	リスト値	開発者を支援する DEBUG ログレベルにどのトレース情報を書き込むかを制御するオプションのリスト。有効な値は method および api です。
transfer_api_class = cinder.transfer.api.API	文字列値	ボリューム転送 API クラスの完全なクラス名
transport_url = rabbit://	文字列値	メッセージングバックエンドに接続するためのネットワークアドレスおよびオプションのユーザー認証情報 (URL 形式)。想定される形式は次のとおりです。 driver://[user:pass@]host:port[,[userN:passN@]hostN:portN]/virtual_host?query 例:rabbit://rabbitmq:password@127.0.0.1:5672// URL のフィールドの詳細は、https://docs.openstack.org/oslo.messaging/latest/reference/transport.html で oslo_messaging.TransportURL のドキュメントを参照してください。
until_refresh = 0	整数値	使用率がリフレッシュされるまでの予約数
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_chap_auth = False	ブール値	ターゲットの CHAP 認証を有効/無効にするオプション。
use_default_quota_class = True	ブール値	デフォルトクォータでのデフォルトクォータクラスの使用を有効または無効にします。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_forwarded_for = False	ブール値	X-Forwarded-For を正規のリモートアドレスとして扱います。これは、プロキシーをサニタイズしている場合のみ有効にします。
use_multipath_for_image_xfer = False	ブール値	ボリュームとボリュームの譲渡用のマルチパスを使用して cinder のボリュームを接続/割り当て解除しますか。このパラメーターは、すべてのバックエンドに共通の設定として、各バックエンドセクションまたは backend_defaults セクションで設定する必要があります。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
verify_glance_signatures = enabled	文字列値	イメージ署名の検証を有効にします。 Cinder は Glance からのイメージ署名メタデータを使用して、そのイメージのダウンロード中に署名済みイメージの署名を検証します。ここには 2 つのオプションがあります。 1. **Enabled**: イメージに署名メタデータがある場合に検証します。 2. **disabled**: 検証はオフになっています。 イメージ署名を検証できないか、イメージ署名のメタデータが不完全である場合、Cinder はボリュームを作成してエラー状態に更新します。これにより、エンドユーザーはボリュームの作成に使用するイメージデータの整合性が強化されます。
vmdk_allowed_types = ['streamOptimized', 'monolithicSparse']	リスト値	許可されている VMDK createType サブフォーマットを説明する文字列のリスト。ボリュームへの書き込み時にイメージが RAW 形式に変換されるときに、名前付きエクステントを処理する際にホストファイルが公開される可能性を回避するために、スパースヘッダーを持つ単一ファイルのバリアントのみを含めることを推奨します。このリストが空の場合、VMDK イメージは許可されません。
volume_api_class = cinder.volume.api.API	文字列値	使用するボリューム API クラスの完全クラス名
volume_backend_name = None	文字列値	特定のドライバー実装のバックエンド名
volume_clear = zero	文字列値	古いボリュームの消去に使用するメソッド
volume_clear_ionice = None	文字列値	削除後のボリュームをゼロにするために使用されるプロセスの i/o 優先度を変更するために ionice に渡すフラグで、例えば-c3 はアイドルのみの優先度です。
volume_clear_size = 0	整数値	古いボリュームの開始時に消去する MiB 単位のサイズ。1024 MiB at max. 0 ⇒ all
volume_copy_blkio_cgroup_name = cinder-volume-copy	文字列値	ボリュームコピーの帯域幅を制限するために使用される blkio cgroup 名

設定オプション = デフォルト値	型	説明
volume_copy_bps_limit = 0	整数値	ボリュームコピーの帯域幅の上限。0 ⇒ 無制限
volume_dd_blocksize = 1M	文字列値	ボリュームのコピー/消去時に使用されるデフォルトのブロックサイズ
volume_manager = cinder.volume.manager.VolumeManager	文字列値	ボリュームの Manager のフルクラス名
volume_name_template = volume-%s	文字列値	ボリューム名の生成に使用するテンプレートの文字列
volume_number_multiplier = -1.0	浮動小数点の値	ボリューム番号の重み付けに使用する乗数。負の数値は、分散とスタックに対する意味です。
volume_service_inithost_offload = False	ブール値	ボリュームサービスの起動時に保留中のボリューム削除のオフロード
volume_transfer_key_length = 16	整数値	自動生成される認証キーの文字数。
volume_transfer_salt_length = 8	整数値	salt の文字数。
volume_usage_audit_period = month	文字列値	ボリュームの使用状況を生成する期間。オプションは、hour、day、month、または year です。
volumes_dir = $state_path/volumes	文字列値	ボリューム設定ファイルのストレージディレクトリー
vrts_lun_sparse = True	ブール値	スパース Lun を作成します。
vrts_target_config = /etc/cinder/vrts_target.xml	文字列値	VA 設定ファイル
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
wsgi_default_pool_size = 100	整数値	wsgi で使用される greenthread のプールのサイズ
wsgi_keep_alive = True	ブール値	False の場合は、クライアントのソケット接続を明示的に閉じます。
wsgi_log_format = %(client_ip)s "%(request_line)s" status: %(status_code)s len: %(body_length)s time: %(wall_seconds).7f	文字列値	ログ行を生成するためにテンプレートとして使用される Python 形式の文字列。client_ip、date_time、request_line、status_code、body_length、wall_seconds をこの形式にフォーマットできます。
wsgi_server_debug = False	ブール値	サーバーが 500 エラーで例外トレースバックをクライアントに送信する必要がある場合は true。False の場合、サーバーは空の本文で応答します。
zoning_mode = None	文字列値	FC Zoning モードが設定され、**ファブリック** のみに対応するようになりました。

2.1.2. backend

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[backend]** グループで使用できるオプションの概要をまとめたものです。

表2.1 backend

設定オプション = デフォルト値	型	説明
backend_host = None	文字列値	ホスト値のバックエンドオーバーライド。

2.1.3. backend_defaults

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[backend_defaults]** グループ下で使用できるオプションの概要をまとめたものです。

表2.2 backend_defaults

設定オプション = デフォルト値	型	説明
auto_calc_max_oversubscription_ratio = False	ブール値	K2 ドライバーは、このオプションを True に設定すると max_oversubscription_ratio が計算されます。

設定オプション = デフォルト値	型	説明
backend_availability_zone = None	文字列値	このボリュームバックエンドのアベイラビリティーゾーン。設定されていない場合、storage_availability_zone オプション値がすべてのバックエンドのデフォルトとして使用されます。
backend_native_threads_pool_size = 20	整数値	バックエンドのネイティブスレッドプールのサイズ。RBD ドライバーなど、これに大きく依存するバックエンドについては、増やします。
chap = disabled	文字列値	iscsi でのみ有効な CHAP 認証モード (disabled|enabled)
`chap_password = `	文字列値	指定した CHAP アカウント名のパスワード。
`chap_username = `	文字列値	CHAP ユーザー名
check_max_pool_luns_threshold = False	ブール値	非推奨: プール LUN の最大数の上限に達すると、free_capacity_gb を 0 として報告します。デフォルト値は false です。
chiscsi_conf = /etc/chelsio-iscsi/chiscsi.conf	文字列値	Chiscsi (CXT) グローバルデフォルト設定ファイル
cinder_eternus_config_file = /etc/cinder/cinder_fujitsu_eternus_dx.xml	文字列値	cinder eternus_dx ボリュームドライバーの設定ファイル。
cinder_huawei_conf_file = /etc/cinder/cinder_huawei_conf.xml	文字列値	Cinder Huawei ドライバーの設定ファイルです。
connection_type = iscsi	文字列値	IBM Storage アレイへの接続タイプ
cycle_period_seconds = 300	整数値	これは、シリクリングモードのマルチとのグローバルミラー関係に適用される任意のサイクル期間を定義します。複数の cycling_mode を使用するグローバルミラー関係は、各期間に最大 1 回の完全なサイクルを実行します。デフォルトは 300 秒で、有効な秒数は 60-86400 です。
datacore_api_timeout = 300	整数値	DataCore API 呼び出しからの応答を待機する秒数。

設定オプション = デフォルト値	型	説明
datacore_disk_failed_delay = 300	整数値	DataCore 仮想ディスクが失敗状態から回復するまで待機する秒数。
datacore_disk_pools = []	リスト値	ボリュームドライバーで使用できる DataCore ディスクプールのリスト。
datacore_disk_type = single	文字列値	DataCore 仮想ディスクタイプ (シングル/ミラーリング)。ミラーリングされた仮想ディスクには、サーバーグループ内に 2 台のストレージサーバーが必要です。
datacore_fc_unallowed_targets = []	リスト値	ボリュームの接続に使用できない FC ターゲットのリスト。DataCore FibreChannel ボリュームドライバーがボリュームアタッチメントで一部のフロントエンドターゲットを使用しないようにするには、このオプションを指定し、各ターゲットの iqn とターゲットマシンを値としてリストします (<wwpns:target name>, <wwpns:target name>, <wwpns:target name> など)。
datacore_iscsi_chap_storage = $state_path/.datacore_chap	文字列値	動的に生成された iSCSI CHAP シークレットが保存される完全修飾ファイル名。
datacore_iscsi_unallowed_targets = []	リスト値	ボリュームの接続に使用できない iSCSI ターゲットのリスト。DataCore iSCSI ボリュームドライバーがボリューム接続で一部のフロントエンドターゲットを使用しないようにするには、このオプションを指定し、各ターゲットの iqn とターゲットマシンを値としてリスト表示します (<iqn:target name>, <iqn:target name>, <iqn:target name> など)。
datacore_storage_profile = None	文字列値	DataCore 仮想ディスクストレージプロファイル。
default_timeout = 31536000	整数値	CLI 操作のデフォルトのタイムアウト (単位: 分)。たとえば、LUN 移行は通常の長時間実行される操作で、LUN のサイズとアレイの負荷により異なります。特定のデプロイメントの上限を設定して、不要な待機時間を避けることができます。デフォルトでは 365 日長くなります。
deferred_deletion_delay = 0	整数値	遅延削除のタグが付けられた後、ボリュームが永久削除の対象となるまでの遅延時間 (秒)。

設定オプション = デフォルト値	型	説明
deferred_deletion_purge_interval = 60	整数値	削除のタグが付けられたボリュームをパージするための定期的なタスクの実行間隔の秒数。
dell_api_async_rest_timeout = 15	整数値	Dell SC API 非同期呼び出しのデフォルトタイムアウトを秒単位で指定します。
dell_api_sync_rest_timeout = 30	整数値	Dell SC API 非同期呼び出しのデフォルトタイムアウトを秒単位で指定します。
dell_sc_api_port = 3033	ポート値	Dell API ポート
dell_sc_server_folder = openstack	文字列値	ストレージセンターで使用するサーバーフォルダーの名前
dell_sc_ssn = 64702	整数値	Storage Center システムのシリアル番号
dell_sc_verify_cert = False	ブール値	HTTPS SC 証明書の検証を有効にする
dell_sc_volume_folder = openstack	文字列値	ストレージセンターで使用するボリュームフォルダーの名前
dell_server_os = Red Hat Linux 6.x	文字列値	ストレージセンターで新しいサーバーを作成する際に使用するサーバー OS タイプ。
destroy_empty_storage_group = False	ブール値	最後の LUN が削除されたときにストレージグループを破棄します。デフォルト値は false です。
disable_discovery = False	ブール値	K2 ドライバーでマルチパス接続の iSCSI ディスカバリー (sendtargets) を無効にする。
`dpl_pool = `	文字列値	DPL プール uuid: DPL ボリュームを保存します。
dpl_port = 8357	ポート値	DPL ポート番号。
driver_client_cert = None	文字列値	ドライバーがサポートしている場合、検証用のクライアント証明書へのパス。
driver_client_cert_key = None	文字列値	ドライバーがサポートしている場合、検証用のクライアント証明書キーへのパス。
driver_data_namespace = None	文字列値	保存されるドライバープライベートデータ値の namespace。

設定オプション = デフォルト値	型	説明
driver_ssl_cert_path = None	文字列値	バックエンドの検証に使用される CA_BUNDLE ファイルまたは証明書を持つディレクトリーへのデフォルトパスの指定に使用できます。
driver_ssl_cert_verify = False	ブール値	True に設定すると、http クライアントはバックエンドエンドポイントの SSL 証明書を検証します。
driver_use_ssl = False	ブール値	ドライバーが SSL をサポートしている場合は、バックエンドストレージへの接続に SSL を使用するようドライバーに指示します。
`ds8k_devadd_unitadd_mapping = `	文字列値	IODevice アドレスとユニットアドレス間のマッピング。
ds8k_host_type = auto	文字列値	OpenStack のバージョンが取得前のバージョンで、zLinux システムに接続している場合は zLinux に設定します。それ以外の場合は、auto に設定されます。このパラメーターの有効な値は以下のとおりです。**auto、AMDLinuxRHEL、AMDLinuxSuse、AppleOSX、Fujitsu、Hp、HpTru64、HpVms、LinuxDT、LinuxRF、LinuxRHEL、LinuxSuse、Novell、SGI、SVC、SanFsAIX、SanFsLinux、Sun、VMWare、Win2000、Win2003、Win2008、Win2012、iLinux、nSeries、pLinux、pSeries、pSeriesPowerswap、zLinux、iSeries。**
ds8k_ssid_prefix = FF	文字列値	SSID の最初の 2 桁の数字を設定します。
enable_deferred_deletion = False	ブール値	遅延削除を有効にします。削除時にボリュームの削除についてのタグが付けられますが、ボリュームは後でのみ非同期的に削除されます。
enable_unsupported_driver = False	ブール値	サポート対象外のドライバーを起動できるようにするには、このパラメーターを True に設定します。作業用の CI システムおよびテストを維持していないドライバーは、CI が再び動作するまで、サポート対象外とマークされます。これはドライバーを非推奨としてマークし、次のリリースで削除される可能性があります。
enforce_multipath_for_image_xfer = False	ブール値	これが True に設定されている場合、multipathd が実行されていない場合、イメージ転送用のボリュームのアタッチが中断されます。それ以外の場合は、単一のパスにフォールバックします。このパラメーターは、すべてのバックエンドに共通の設定として、各バックエンドセクションまたは backend_defaults セクションで設定する必要があります。

設定オプション = デフォルト値	型	説明
excluded_domain_ip = None	IP アドレス値	非推奨: iSCSI から除外される障害ドメイン IP が返されます。**非推奨となったバージョン：** Stein *Reason:*Replaced by excluded_domain_ips option
excluded_domain_ips = []	リスト値	iSCSI から除外されるコンマ区切りのフォールトドメイン IP を返します。
expiry_thres_minutes = 720	整数値	このオプションは、NFS イメージキャッシュ内のイメージの最後のアクセス時間のしきい値を指定します。キャッシュクリーニングサイクルが開始されると、過去 M 分以内にアクセスされていないキャッシュ内のイメージが削除され、NFS 共有に空き領域が作られます (M はこのパラメーターの値)。
extra_capabilities = {}	文字列値	ユーザー定義のケイパビリティー。キーと値のペアを指定する JSON 形式の文字列です。キー/値のペアは、CapabilitiesFilter により使用され、リクエストがボリューム種別を指定するときにバックエンド間を選択できます。たとえば、バックエンドのサービスレベルまたは地理的な場所を指定してから、異なるプロパティーでユーザーが選択できるようにするボリューム種別を作成します。
filter_function = None	文字列値	ホストのフィルターに使用される式の文字列表現。Cinder スケジューラーがドライバーフィルターを使用するように設定されている場合にのみ使用されます。
flashsystem_connection_protocol = FC	文字列値	接続プロトコルは FC である必要があります。(デフォルトは FC です)
flashsystem_iscsi_portid = 0	整数値	デフォルトの iSCSI ポート ID は FlashSystem です。(デフォルトのポートは 0 です。)
flashsystem_multihostmap_enabled = True	ブール値	vdisk によるマルチホストマッピングを許可します。(デフォルトは True)
force_delete_lun_in_storagegroup = True	ブール値	ストレージグループに属する場合でも LUN を削除します。
goodness_function = None	文字列値	ホストの有効性を決定するために使用される式の文字列表現。Cinder スケジューラーが使用する適切な重み付け関数を使用する場合にのみ使用されます。

設定オプション = デフォルト値	型	説明
gpfs_hosts = []	リスト値	GPFS ノードの IP アドレスまたはホスト名のコンマ区切りリスト。
gpfs_hosts_key_file = $state_path/ssh_known_hosts	文字列値	gpfs ノードの SSH ホストキーを含むファイルで、どのドライバーが通信する必要があります。Default=$state_path/ssh_known_hosts
gpfs_images_dir = None	文字列値	GPFS の Image サービスリポジトリーのパスを指定します。イメージを GPFS に保存しない場合は、未定義のままにします。
gpfs_images_share_mode = None	文字列値	使用するイメージのコピーのタイプを指定します。Image サービスのリポジトリーで GPFS も使用している場合に、Image サービスから Block Storage サービスにイメージファイルを効率的に移行できるように設定します。有効な値は 2 つあります。"copy" は、イメージの完全なコピーを作成することを指定し、"copy_on_write" は、コピーオンライト最適化戦略が使用され、イメージファイルの変更されていないブロックが効率的に共有されることを指定します。
gpfs_max_clone_depth = 0	整数値	スナップショットやクローンによって特定のブロックに到達するために必要な間接回数の上限を指定します。コピーオンライトスナップショットまたはクローンの長いチェーンは、パフォーマンスに悪影響を与える可能性がありますが、領域の使用率が向上します。0 は、無制限のクローンの深度を示します。
gpfs_mount_point_base = None	文字列値	Block Storage ボリュームおよびスナップショットファイルが保存される GPFS ディレクトリーのパスを指定します。
`gpfs_private_key = `	文字列値	SSH 認証に使用する秘密鍵のファイル名。
gpfs_sparse_volumes = True	ブール値	最初に領域を消費するスパースファイルとしてボリュームが作成されるように指定します。False に設定すると、ボリュームが完全に割り当てられたファイルとして作成されます。その場合には、作成にはかなり時間がかかる可能性があります。
gpfs_ssh_port = 22	ポート値	使用する SSH ポート。
gpfs_storage_pool = system	文字列値	ボリュームが割り当てられるストレージプールを指定します。デフォルトでは、システムストレージプールが使用されます。

設定オプション = デフォルト値	型	説明
gpfs_strict_host_key_policy = False	ブール値	gpfs ノードへの接続中に厳密な gpfs ホストキーチェックを有効にするオプション。Default=False
gpfs_user_login = root	文字列値	GPFS ノードのユーザー名
`gpfs_user_password = `	文字列値	GPFS ノードユーザーのパスワード。
hitachi_async_copy_check_interval = 10	整数値	コピーペアの削除またはデータの復元中に非同期コピーの状態を確認する間隔 (秒単位)。
hitachi_compute_target_ports = []	リスト値	コンピュートノードにボリュームを接続するために使用されるストレージポートの ID。複数のポートを指定する場合は、コンマでつなぎます (例:CL1-A,CL2-A)。
hitachi_copy_check_interval = 3	整数値	ボリュームコピー中にコピー状態を確認する間隔 (秒単位)。
hitachi_copy_speed = 3	整数値	ストレージシステムのコピー速度。1 または 2 は低速、3 は中速、4 - 15 は高速を示します。
hitachi_discard_zero_page = True	ブール値	DP-VOL でゼロページリクラムを有効または無効にします。
hitachi_exec_retry_interval = 5	整数値	REST API 実行の再試行間隔 (秒単位)。
hitachi_extend_timeout = 600	整数値	ボリューム拡張が完了するまでの最大待機時間 (秒)。
hitachi_group_create = False	ブール値	True の場合、ドライバーは必要に応じてストレージポートにホストグループや iSCSI ターゲットを作成します。
hitachi_group_delete = False	ブール値	True の場合、ドライバーは必要に応じてストレージポートにホストグループや iSCSI ターゲットを削除します。
hitachi_group_name_format = None	文字列値	ホストグループ、iSCSI ターゲット、およびサーバーオブジェクトの形式。
hitachi_host_mode_options = []	リスト値	ホストグループまたは iSCSI ターゲットのホストモードオプション。

設定オプション = デフォルト値	型	説明
hitachi_ldev_range = None	文字列値	**xxxx-yyyy** 形式の LDEV 番号の範囲で、ドライバーが使用できるもの。値は、10 進法 (例: 1000) またはコロンで区切った 16 進法 (例: 00:03:E8) で入力できます。
hitachi_lock_timeout = 7200	整数値	ストレージへのログインまたはロック解除までの最大待機時間 (秒)。
hitachi_lun_retry_interval = 1	整数値	REST API がサーバーに LUN マッピングを追加する際の再試行間隔 (秒単位)。
hitachi_lun_timeout = 50	整数値	LUN マッピングをサーバーに追加するための最大待機時間 (秒)。
hitachi_pair_target_number = 0	整数値	ホストグループまたは iSCSI ターゲットのペアのターゲット名
hitachi_pools = []	リスト値	DP プールのプール番号またはプール名。
hitachi_port_scheduler = False	ブール値	設定されたポートへの WWN のポートスケジューリングを有効にして、WWN がラウンドロビン方式でポートに登録されるようにします。
hitachi_rest_another_ldev_mapped_retry_timeout = 600	整数値	新しい LUN 割り当て要求が失敗した場合の再試行時間 (秒)。
hitachi_rest_connect_timeout = 30	整数値	REST API セッションへの接続の最大待機時間 (秒)。
hitachi_rest_disable_io_wait = True	ブール値	このオプションを使用すると、ボリュームをすぐに切断できます。False に設定すると、ストレージが I/O 後にボリュームをデタッチするのに数分かかることがあります。
hitachi_rest_get_api_response_timeout = 1800	整数値	同期メソッド (GET など) に対する応答の最大待機時間 (秒)
hitachi_rest_job_api_response_timeout = 1800	整数値	PUT や DELETE などの REST API からの非同期メソッドに対する応答の最大待機時間 (秒)。
hitachi_rest_keep_session_loop_interval = 180	整数値	REST API セッションを維持するためのループ間隔 (秒単位)。
hitachi_rest_pair_target_ports = []	リスト値	ホストグループまたは iSCSI ターゲットのペアのターゲットポート名

設定オプション = デフォルト値	型	説明
hitachi_rest_server_busy_timeout = 7200	整数値	REST API がビジー状態を返す場合の最大待機時間 (秒)。
hitachi_rest_tcp_keepalive = True	ブール値	REST API tcp keepalive の使用を有効または無効にします。
hitachi_rest_tcp_keepcnt = 4	整数値	TCP キープアライブパケットの最大送信数。
hitachi_rest_tcp_keepidle = 60	整数値	最初の TCP キープアライブパケットを送信するまでの待機時間 (秒)。
hitachi_rest_tcp_keepintvl = 15	整数値	TCP キープアライブパケットの送信間隔 (秒単位)。
hitachi_rest_timeout = 30	整数値	各 REST API リクエストの最大待機時間 (秒単位)。
hitachi_restore_timeout = 86400	整数値	復元操作が完了するまでの最大待機時間 (秒)。
hitachi_snap_pool = None	文字列値	スナップショットプールのプール番号またはプール名。
hitachi_state_transition_timeout = 900	整数値	ボリューム遷移が完了するまでの最大待機時間 (秒)。
hitachi_storage_id = None	文字列値	ストレージシステムの製品数。
hitachi_target_ports = []	リスト値	ボリュームをコントローラーノードにアタッチするために使用されるストレージポートの ID。複数のポートを指定する場合は、コンマでつなぎます (例:CL1-A,CL2-A)。
hitachi_zoning_request = False	ブール値	True の場合、FC ゾーニングマネージャーが有効であれば、ドライバーはサーバーとストレージシステム間の FC ゾーニングを設定します。
`hpe3par_api_url = `	文字列値	WSAPI Server URLこの設定は、3PAR および Primera の両方に適用されます。例 1: 3PAR の場合、URL は、https://<3par ip>:8080/api/v1 です。例 2:Primera の場合、URL は、https://<primera ip>:443/api/v1 です。
hpe3par_cpg = ['OpenStack']	リスト値	ボリューム作成に使用する 3PAR / Primera CPG のリスト

設定オプション = デフォルト値	型	説明
`hpe3par_cpg_snap = `	文字列値	ボリュームのスナップショットに使用する 3PAR / Primera CPG。空の場合、userCPG が使用されます。
hpe3par_debug = False	ブール値	3PAR / Primera への HTTP デバッグの有効化
hpe3par_iscsi_chap_enabled = False	ブール値	iSCSI 接続の CHAP 認証を有効にします。
hpe3par_iscsi_ips = []	リスト値	使用するターゲット iSCSI アドレスのリスト。
`hpe3par_password = `	文字列値	hpe3par_username で指定したユーザーの 3PAR / Primera パスワード
`hpe3par_snapshot_expiration = `	文字列値	スナップショットの有効期限が切れて削除されるまでの時間 (時間単位)。これは、有効期限よりも大きくなければなりません。
`hpe3par_snapshot_retention = `	文字列値	スナップショットを保持する時間 (時間単位)。この期限が切れる前に削除することはできません。
`hpe3par_target_nsp = `	文字列値	次の場合に使用される 3PAR バックエンドの nsp: (1) cinder.conf でマルチパスが有効になっていません。(2) ファイバーチャネルゾーンマネージャーは使用されません。(3) 3PAR バックエンドは、この特定の nsp でのみ事前設定されます。たとえば、nsp が 2 1 2 の場合、オプションの値の形式は 2:1:2 になります。
`hpe3par_username = `	文字列値	**edit** ロールを持つ 3PAR / Primera ユーザー名
hpexp_async_copy_check_interval = 10	整数値	コピーを非同期でチェックする間隔 (秒)
hpexp_compute_target_ports = []	リスト値	コンピュートノードにボリュームを接続するために使用されるストレージポートの ID。複数のポートを指定する場合は、コンマでつなぎます (例:CL1-A,CL2-A)。
hpexp_copy_check_interval = 3	整数値	コピーをチェックする間隔 (秒)
hpexp_copy_speed = 3	整数値	ストレージシステムのコピー速度。1 または 2 は低速、3 は中速、4 - 15 は高速を示します。
hpexp_discard_zero_page = True	ブール値	THP V-VOL でのゼロページ再利用を有効または無効にします。

設定オプション = デフォルト値	型	説明
hpexp_exec_retry_interval = 5	整数値	REST API 実行の再試行間隔 (秒単位)。
hpexp_extend_timeout = 600	整数値	ボリューム拡張が完了するまでの最大待機時間 (秒)。
hpexp_group_create = False	ブール値	True の場合、ドライバーは必要に応じてストレージポートにホストグループや iSCSI ターゲットを作成します。
hpexp_group_delete = False	ブール値	True の場合、ドライバーは必要に応じてストレージポートにホストグループや iSCSI ターゲットを削除します。
hpexp_host_mode_options = []	リスト値	ホストグループまたは iSCSI ターゲットのホストモードオプション。
hpexp_ldev_range = None	文字列値	**xxxx-yyyy** 形式の LDEV 番号の範囲で、ドライバーが使用できるもの。値は、10 進法 (例: 1000) またはコロンで区切った 16 進法 (例: 00:03:E8) で入力できます。
hpexp_lock_timeout = 7200	整数値	ストレージのロックが解除されるまでの最大待機時間 (秒)。
hpexp_lun_retry_interval = 1	整数値	LUN を追加する REST API の再試行間隔 (秒単位)。
hpexp_lun_timeout = 50	整数値	LUN の追加が完了するまでの最大待機時間 (秒)。
hpexp_pools = []	リスト値	THP プールのプール番号またはプール名。
hpexp_rest_another_ldev_mapped_retry_timeout = 600	整数値	新しい LUN 割り当て要求が失敗した場合の再試行時間 (秒)。
hpexp_rest_connect_timeout = 30	整数値	REST API 接続が完了するまでの最大待機時間 (秒)。
hpexp_rest_disable_io_wait = True	ブール値	I/O 後にボリュームをデタッチするのに時間がかかる場合があります。このオプションを使用すると、ボリュームのデタッチがすぐに完了します。
hpexp_rest_get_api_response_timeout = 1800	整数値	REST API の GET メソッドに対する応答の最大待機時間 (秒)。

設定オプション = デフォルト値	型	説明
hpexp_rest_job_api_response_timeout = 1800	整数値	REST API からの応答の最大待機時間 (秒)。
hpexp_rest_keep_session_loop_interval = 180	整数値	REST API セッションを維持するためのループ間隔 (秒単位)。
hpexp_rest_server_busy_timeout = 7200	整数値	REST API がビジー状態を返す場合の最大待機時間 (秒)。
hpexp_rest_tcp_keepalive = True	ブール値	REST API tcp keepalive の使用を有効または無効にします。
hpexp_rest_tcp_keepcnt = 4	整数値	TCP キープアライブパケットの最大送信数。
hpexp_rest_tcp_keepidle = 60	整数値	最初の TCP キープアライブパケットを送信するまでの待機時間 (秒)。
hpexp_rest_tcp_keepintvl = 15	整数値	TCP キープアライブパケットの送信間隔 (秒単位)。
hpexp_rest_timeout = 30	整数値	REST API 実行が完了するまでの最大待機時間 (秒)。
hpexp_restore_timeout = 86400	整数値	復元操作が完了するまでの最大待機時間 (秒)。
hpexp_snap_pool = None	文字列値	スナップショットプールのプール番号またはプール名。
hpexp_state_transition_timeout = 900	整数値	ボリューム遷移が完了するまでの最大待機時間 (秒)。
hpexp_storage_id = None	文字列値	ストレージシステムの製品数。
hpexp_target_ports = []	リスト値	ボリュームをコントローラーノードにアタッチするために使用されるストレージポートの ID。複数のポートを指定する場合は、コンマでつなぎます (例:CL1-A,CL2-A)。
hpexp_zoning_request = False	ブール値	True の場合、FC ゾーニングマネージャーが有効であれば、ドライバーはサーバーとストレージシステム間の FC ゾーニングを設定します。
hpmsa_api_protocol = https	文字列値	HPMSA API インターフェイスプロトコル。

設定オプション = デフォルト値	型	説明
hpmsa_iscsi_ips = []	リスト値	コンマ区切りのターゲット iSCSI IP アドレスのリスト。
hpmsa_pool_name = A	文字列値	ボリュームの作成に使用するプールまたは Vdisk 名。
hpmsa_pool_type = virtual	文字列値	リニア (Vdisk) または仮想 (Pool)
hpmsa_verify_certificate = False	ブール値	HPMSA アレイ SSL 証明書を検証するかどうか。
hpmsa_verify_certificate_path = None	文字列値	HPMSA 配列 SSL 証明書パス。
hypermetro_devices = None	文字列値	リモートデバイスハイパーメトロが使用されます。
iet_conf = /etc/iet/ietd.conf	文字列値	非推奨: IET 設定ファイル
ignore_pool_full_threshold = False	ブール値	プールの完全なしきい値に達した場合でも LUN の作成を強制します。デフォルト値は false です。
image_upload_use_cinder_backend = False	ブール値	True に設定すると、raw 形式の upload-to-image は、ボリュームコンテンツをアップロードする代わりに、クローン作成されたボリュームを作成し、その場所をイメージサービスに登録します。Image サービスで cinder バックエンドおよび場所のサポートを有効にする必要があります。
image_upload_use_internal_tenant = False	ブール値	True に設定すると、upload-to-image で作成したイメージボリュームは内部テナントに配置されます。それ以外の場合は、イメージボリュームが現在のコンテキストのテナントに作成されます。
image_volume_cache_enabled = False	ブール値	このバックエンドのイメージボリュームキャッシュを有効にします。
image_volume_cache_max_count = 0	整数値	イメージボリュームキャッシュで許可されるエントリーの最大数。0 ⇒ 無制限
image_volume_cache_max_size_gb = 0	整数値	このバックエンドのイメージボリュームキャッシュの最大サイズ (単位: GB)。0 ⇒ 無制限
included_domain_ips = []	リスト値	iSCSI リターンに含まれるコンマ区切りのフォールトドメイン IP。

設定オプション = デフォルト値	型	説明
infinidat_iscsi_netspaces = []	リスト値	iSCSI 接続に使用するネットワークスペースの名前のリスト
infinidat_pool_name = None	文字列値	ボリュームの割り当て元となるプールの名前
infinidat_storage_protocol = fc	文字列値	ホストとストレージバックエンド間でデータを転送するためのプロトコル。
infinidat_use_compression = False	ブール値	新規に作成されたボリュームの圧縮を有効にするかどうかを指定します。
initiator_auto_deregistration = False	ブール値	関連するストレージグループが破棄された後に、イニシエーターを自動的に登録解除します。デフォルト値は false です。
initiator_auto_registration = False	ブール値	イニシエーターを自動的に登録します。デフォルト値は false です。
initiator_check = False	ブール値	この値を使用して initiator_check を有効にします。
interval = 3	整数値	この値を使用して間隔の長さを秒単位で指定します。
io_port_list = None	リスト値	Nova または Cinder で使用されるコンマ区切りの iSCSI ポートまたは FC ポート
iscsi_initiators = None	文字列値	ホスト名と iSCSI イニシエーター IP アドレス間のマッピング。
iscsi_iotype = fileio	文字列値	iSCSI ターゲットの動作を、任意で blockio または fileio のいずれかを実行するように設定します。auto を設定すると、Cinder はバッキングデバイスのタイプを自動検出できます。
`iscsi_target_flags = `	文字列値	iSCSI ターゲットにターゲット固有のフラグを設定します。tgtadm で bsoflags オプションを使用してバッキングデバイスフラグを指定するためにのみ使用されます。指定された文字列は、基礎となるツールと同様に渡されます。
iscsi_write_cache = on	文字列値	iSCSI ターゲットの動作を、ライトバックオン (on) または write-through (off) のいずれかを実行するように設定します。このパラメーターは、target_helper が tgtadm に設定されている場合に有効です。

設定オプション = デフォルト値	型	説明
iser_helper = tgtadm	文字列値	使用する iSER ターゲットユーザーランドツールの名前
iser_ip_address = $my_ip	文字列値	iSER デーモンがリッスンする IP アドレス
iser_port = 3260	ポート値	iSER デーモンがリッスンするポート
iser_target_prefix = iqn.2010-10.org.openstack:	文字列値	iSER ボリュームの接頭辞
lenovo_api_protocol = https	文字列値	Lenovo API インターフェイスプロトコル。
lenovo_iscsi_ips = []	リスト値	コンマ区切りのターゲット iSCSI IP アドレスのリスト。
lenovo_pool_name = A	文字列値	ボリュームの作成に使用するプールまたは Vdisk 名。
lenovo_pool_type = virtual	文字列値	リニア (VDisk) または仮想 (Pool)
lenovo_verify_certificate = False	ブール値	Lenovo アレイ SSL 証明書を検証するかどうか。
lenovo_verify_certificate_path = None	文字列値	Lenovo アレイの SSL 証明書のパスです。
linstor_autoplace_count = 0	整数値	ボリュームデプロイメントでレプリケーションカウントを自動配置します。0 = autoplace を使用しないフルクラスターレプリケーション、1= レプリケーションを使用しない単一ノードのデプロイメント、2 以上 = autoplace を使用して複製されたデプロイメント。
linstor_controller_diskless = True	ブール値	Cinder ノードがディスクレスの LINSTOR ノードであることを意味します。
linstor_default_blocksize = 4096	整数値	イメージの復元に使用するデフォルトのブロックサイズ。iSCSI トランスポートを使用する場合、このオプションはブロックサイズを指定します。
linstor_default_storage_pool_name = DfltStorPool	文字列値	LINSTOR のデフォルトのストレージプール名。

設定オプション = デフォルト値	型	説明
linstor_default_uri = linstor://localhost	文字列値	LINSTOR のデフォルトストレージ URI。
linstor_default_volume_group_name = drbd-vg	文字列値	LINSTOR のデフォルトのボリュームグループ名。Cinder ボリュームではありません。
linstor_volume_downsize_factor = 4096	浮動小数点の値	デフォルトのボリュームダウン (KiB)= 4 MiB。
load_balance = False	ブール値	PowerMax バックエンドのロードバランシングを有効または無効にします。
load_balance_real_time = False	ブール値	PowerMax バックエンドのポートレベルのロードバランシングのリアルタイムパフォーマンスメトリックを有効または無効にします。
load_data_format = Avg	文字列値	パフォーマンスデータ形式。リアルタイムメトリックには適用されません。利用可能なオプションは avg と max です。
load_look_back = 60	整数値	負荷計算で診断パフォーマンスメトリックをさかのぼる分数。最小値は 0 で、最大値は 1440 (24 時間) です。
load_look_back_real_time = 1	整数値	負荷計算でリアルタイムパフォーマンスメトリックをさかのぼる分数。最小 1 ~ 最大 10。
`lss_range_for_cg = `	文字列値	整合性グループの LSS を確保します。
lvm_conf_file = /etc/cinder/lvm.conf	文字列値	Cinder の LVM ドライバーに使用する LVM conf ファイル。指定されたファイルが存在しない場合、この設定は無視されます (conf ファイルが存在しても使用しない場合は **None** を指定することもできます)。
lvm_mirrors = 0	整数値	0 を超える場合、複数のミラーを持つ LV を作成します。これには、lvm_mirrors と利用可能な領域が 2 つの PV が必要であることに注意してください。
lvm_share_target = False	ブール値	すべての LUN で同じターゲットを共有するかどうか (現在は nvmet でのみサポートされています)。
lvm_suppress_fd_warnings = False	ブール値	LVM コマンドでリークしたファイル記述子の警告を抑制します。
lvm_type = auto	文字列値	配置する LVM ボリュームのタイプ (default、thin、auto) を指定します。thin がサポートされる場合、auto はデフォルトで thin になります。

設定オプション = デフォルト値	型	説明
macrosan_client = None	リスト値	Macrosan iscsi_clients list.複数のクライアントを設定できます。このような形式で設定することができます。(host; client_name; sp1_iscsi_port; sp2_iscsi_port)、(host; client_name; sp1_iscsi_port; sp2_iscsi_port) 重要な警告、client_name には以下の条件があります。[a-z A-Z0-9.-_:]、最大文字数は 31 文字です。(controller1; device1; eth-1:0; eth-2:0)、(controller2; device2; eth-1:0/eth-1:1; eth-2:0/eth-2:1)。
macrosan_client_default = None	文字列値	これは、iscsi のデフォルト接続ポート名です。このデフォルト設定は、ホスト関連の情報が取得されていない場合に使用されます。例: eth-1:0/eth-1:1; eth-2:0/eth-2:1
macrosan_fc_keep_mapped_ports = True	ブール値	FC 接続の場合は、ポートに関連する設定アイテムが維持されます。
macrosan_fc_use_sp_port_nr = 1	整数値	use_sp_port_nr パラメーターは、スイッチの非パスモードで FC 接続が確立されると、シングルエンドメモリーによって使用されるオンライン FC ポートの数です。最大 4
macrosan_force_unmap_itl = True	ブール値	ボリュームの削除時に強制的に切断される
macrosan_log_timing = True	ブール値	ログタイミングを有効にするかどうか
macrosan_pool = None	文字列値	ボリュームの作成に使用するプール
macrosan_replication_destination_ports = None	リスト値	セカンダリーデバイス
macrosan_replication_ipaddrs = None	リスト値	MacroSAN レプリケーションデバイスの IP アドレス
macrosan_replication_password = None	文字列値	MacroSAN レプリケーションデバイスのパスワード
macrosan_replication_username = None	文字列値	MacroSAN レプリケーションデバイスのユーザー名
macrosan_sdas_ipaddrs = None	リスト値	MacroSAN のデバイスの IP アドレス

設定オプション = デフォルト値	型	説明
macrosan_sdas_password = None	文字列値	MacroSAN のデバイスのパスワード
macrosan_sdas_username = None	文字列値	MacroSAN のデバイスのユーザー名
macrosan_snapshot_resource_ratio = 1.0	浮動小数点の値	スナップショットのリソース比率の設定
macrosan_thin_lun_extent_size = 8	整数値	シン lun のエクステントサイズを設定します。
macrosan_thin_lun_high_watermark = 20	整数値	シン lun の高基準値の設定
macrosan_thin_lun_low_watermark = 5	整数値	シン lun の低基準値の設定
`management_ips = `	文字列値	管理 IP アドレスのリスト (コンマで区切る)
max_luns_per_storage_group = 255	整数値	ストレージグループのデフォルトの LUN の最大数。デフォルトでは、値は 255 です。
max_over_subscription_ratio = 20.0	文字列値	シンプロビジョニングが有効な場合にサブスクリプションに対する比率を表す。デフォルトの比率は 20.0 です。つまり、プロビジョニングした容量は、物理容量の合計の 20 倍になります。比率が 10.5 の場合は、プロビジョニング容量が物理容量の合計の 10.5 倍になることを意味します。比率が 1.0 の場合は、プロビジョニングした容量が物理容量の合計を超えることができません。比率が **auto** の場合、Cinder はプロビジョニングされた容量と使用領域に基づいて比率を自動的に算出します。auto に設定されていない場合は、比率は 1.0 以上である必要があります。
metro_domain_name = None	文字列値	リモートクラスのデバイスドメイン名。
metro_san_address = None	文字列値	リモート Mero デバイス要求の URL。
metro_san_password = None	文字列値	リモートメトロデバイスの san パスワード。
metro_san_user = None	文字列値	リモート metro デバイス san ユーザー。

設定オプション = デフォルト値	型	説明
metro_storage_pools = None	文字列値	リモート Mero デバイスプール名。
`nas_host = `	文字列値	NAS システムの IP アドレスまたはホスト名。
nas_login = admin	文字列値	NAS システムに接続するためのユーザー名
nas_mount_options = None	文字列値	Cinder ボリュームが保存されるストレージバックエンドファイルシステムのマウントに使用されるオプション。
`nas_password = `	文字列値	NAS システムに接続するためのパスワード
`nas_private_key = `	文字列値	SSH 認証に使用する秘密鍵のファイル名。
nas_secure_file_operations = auto	文字列値	ネットワーク接続ストレージシステムは、ルートレベルのアクセスが許可されていない安全な環境で運用できるようにします。False に設定すると、アクセスは root ユーザーとして行われ、安全ではありません。True に設定すると、アクセスは root ではありません。auto に設定すると、これが新規インストールかどうかを判別するためにチェックが行われます。新規インストールの場合は True が使用され、それ以外の場合は False が使用されます。デフォルトは auto です。
nas_secure_file_permissions = auto	文字列値	ネットワーク接続されたストレージボリュームファイルによりセキュアなファイルパーミッションを設定し、幅広いその他/アクセスを制限します。False に設定すると、ボリュームは open パーミッションで作成されます。True に設定すると、cinder ユーザーおよびグループ (660) の権限でボリュームが作成されます。auto に設定すると、これが新規インストールかどうかを判別するためにチェックが行われます。新規インストールの場合は True が使用され、それ以外の場合は False が使用されます。デフォルトは auto です。
`nas_share_path = `	文字列値	Cinder ボリュームの格納に使用するファイル共有へのパス。たとえば、10.0.5.10:/srv/export1 で利用可能な NFS サーバーエクスポートの場合は "/srv/export1" になります。
nas_ssh_port = 22	ポート値	NAS システムへの接続に使用する SSH ポート。
nas_volume_prov_type = thin	文字列値	ボリュームの作成時に使用されるプロビジョニングタイプ。

設定オプション = デフォルト値	型	説明
naviseccli_path = None	文字列値	Naviseccli パス
nec_v_async_copy_check_interval = 10	整数値	コピーペアの削除またはデータの復元中に非同期コピーの状態を確認する間隔 (秒単位)。
nec_v_compute_target_ports = []	リスト値	コンピュートノードにボリュームを接続するために使用されるストレージポートの ID。複数のポートを指定する場合は、コンマでつなぎます (例:CL1-A,CL2-A)。
nec_v_copy_check_interval = 3	整数値	ボリュームコピー中にコピー状態を確認する間隔 (秒単位)。
nec_v_copy_speed = 3	整数値	ストレージシステムのコピー速度。1 または 2 は低速、3 は中速、4 - 15 は高速を示します。
nec_v_discard_zero_page = True	ブール値	DP-VOL でゼロページリクラムを有効または無効にします。
nec_v_exec_retry_interval = 5	整数値	REST API 実行の再試行間隔 (秒単位)。
nec_v_extend_timeout = 600	整数値	ボリューム拡張が完了するまでの最大待機時間 (秒)。
nec_v_group_create = False	ブール値	True の場合、ドライバーは必要に応じてストレージポートにホストグループや iSCSI ターゲットを作成します。
nec_v_group_delete = False	ブール値	True の場合、ドライバーは必要に応じてストレージポートにホストグループや iSCSI ターゲットを削除します。
nec_v_host_mode_options = []	リスト値	ホストグループまたは iSCSI ターゲットのホストモードオプション
nec_v_ldev_range = None	文字列値	**xxxx-yyyy** 形式の LDEV 番号の範囲で、ドライバーが使用できるもの。値は、10 進法 (例: 1000) またはコロンで区切った 16 進法 (例: 00:03:E8) で入力できます。
nec_v_lock_timeout = 7200	整数値	ストレージのロックが解除されるまでの最大待機時間 (秒)。
nec_v_lun_retry_interval = 1	整数値	LUN を追加する REST API の再試行間隔 (秒単位)。

設定オプション = デフォルト値	型	説明
nec_v_lun_timeout = 50	整数値	LUN の追加が完了するまでの最大待機時間 (秒)。
nec_v_pools = []	リスト値	DP プールのプール番号またはプール名。
nec_v_rest_another_ldev_mapped_retry_timeout = 600	整数値	新しい LUN 割り当て要求が失敗した場合の再試行時間 (秒)。
nec_v_rest_connect_timeout = 30	整数値	REST API 接続が完了するまでの最大待機時間 (秒)。
nec_v_rest_disable_io_wait = True	ブール値	I/O 後にボリュームをデタッチするのに時間がかかる場合があります。このオプションを使用すると、ボリュームのデタッチがすぐに完了します。
nec_v_rest_get_api_response_timeout = 1800	整数値	REST API の GET メソッドに対する応答の最大待機時間 (秒)。
nec_v_rest_job_api_response_timeout = 1800	整数値	REST API からの応答の最大待機時間 (秒)。
nec_v_rest_keep_session_loop_interval = 180	整数値	REST API セッションを維持するためのループ間隔 (秒単位)。
nec_v_rest_server_busy_timeout = 7200	整数値	REST API がビジー状態を返す場合の最大待機時間 (秒)。
nec_v_rest_tcp_keepalive = True	ブール値	REST API tcp keepalive の使用を有効または無効にします。
nec_v_rest_tcp_keepcnt = 4	整数値	TCP キープアライブパケットの最大送信数。
nec_v_rest_tcp_keepidle = 60	整数値	最初の TCP キープアライブパケットを送信するまでの待機時間 (秒)。
nec_v_rest_tcp_keepintvl = 15	整数値	TCP キープアライブパケットの送信間隔 (秒単位)。
nec_v_rest_timeout = 30	整数値	REST API 実行が完了するまでの最大待機時間 (秒)。
nec_v_restore_timeout = 86400	整数値	復元操作が完了するまでの最大待機時間 (秒)。
nec_v_snap_pool = None	文字列値	スナップショットプールのプール番号またはプール名。

設定オプション = デフォルト値	型	説明
nec_v_state_transition_timeout = 900	整数値	ボリューム遷移が完了するまでの最大待機時間 (秒)。
nec_v_storage_id = None	文字列値	ストレージシステムの製品数。
nec_v_target_ports = []	リスト値	ボリュームをコントローラーノードにアタッチするために使用されるストレージポートの ID。複数のポートを指定する場合は、コンマでつなぎます (例:CL1-A,CL2-A)。
nec_v_zoning_request = False	ブール値	True の場合、FC ゾーニングマネージャーが有効であれば、ドライバーはサーバーとストレージシステム間の FC ゾーニングを設定します。
netapp_api_trace_pattern = (.*)	文字列値	API トレースを制限する正規表現。このオプションは、**trace_flags** オプションで **api** トレースを有効にする場合にのみ許可されます。デフォルトでは、すべての API はトレースされます。
netapp_copyoffload_tool_path = None	文字列値	このオプションは、NetApp コピーオフロードツールバイナリーのパスを指定します。バイナリーに実行権限が設定され、cinder-volume プロセスの実効ユーザーがファイルを実行できるようにします。
netapp_host_type = None	文字列値	このオプションは、LUN にアクセスできるすべてのイニシエーターのオペレーティングシステムのタイプを定義します。この情報は、LUN を個々のホストまたはホストのグループにマッピングする際に使用されます。
netapp_login = None	文字列値	ストレージシステムまたはプロキシーサーバーへのアクセスに使用される管理ユーザーアカウント名。
netapp_lun_ostype = None	文字列値	このオプションは、データからエクスポートされた LUN にアクセスするオペレーティングシステムのタイプを定義します。これは、作成時に LUN に割り当てられます。
netapp_lun_space_reservation = enabled	文字列値	このオプションは、LUN の割り当て用にストレージ領域が予約されているかどうかを判断します。有効にすると、LUN はシックプロビジョニングされます。領域予約が無効になっていると、ストレージ領域はオンデマンドで割り当てられます。
netapp_nfs_image_cache_cleanup_interval = 600	整数値	NFS イメージキャッシュクリーンアップタスク間の時間を秒単位で設定します。

設定オプション = デフォルト値	型	説明
netapp_password = None	文字列値	netapp_login オプションで指定した管理ユーザーアカウントのパスワード。
netapp_pool_name_search_pattern = (.+)	文字列値	このオプションは、プロビジョニングを指定のプールに制限するために使用されます。このオプションの値を正規表現で指定します。これは、Cinder のプールを表すストレージバックエンドからオブジェクトの名前に適用されます。このオプションは、iSCSI または FC を使用するようにストレージプロトコルが設定されている場合にのみ使用されます。
netapp_replication_aggregate_map = None	dict 値	バックエンドレプリケーション全体を使用する場合に、ソースバックエンドと宛先バックエンド間の集約マッピングを表すディクショナリーのマルチオプト。cinder プール (NetApp FlexVol/FlexGroup) に関連付けられたソースアグリゲートごとに、レプリケーションターゲットデバイスで宛先アグリゲートを指定する必要があります。レプリケーションターゲットデバイスは、設定オプション replication_device を使用して設定されます。このオプションは、レプリケーションデバイスがある回数だけ指定します。各エントリーは標準の dict 設定形式を使います: netapp_replication_aggregate_map = backend_id: <name_of_replication_device_section>,src_aggr_name1:dest_aggr_name1,src_aggr_name2:dest_aggr_name2,...
netapp_replication_volume_online_timeout = 360	整数値	複製ボリュームの作成が完了してオンラインになるまで待機する時間を秒単位で設定します。
netapp_server_hostname = None	文字列値	ストレージシステムまたはプロキシーサーバーのホスト名 (または IP アドレス)。
netapp_server_port = None	整数値	ストレージシステムまたはプロキシーサーバーとの通信に使用する TCP ポート。指定されていない場合、DataListenerExternal ドライバーは HTTP に 80 を使用し、HTTPS に 443 を使用します。
netapp_size_multiplier = 1.2	浮動小数点の値	要求されたボリュームサイズに乗算する数量。仮想ストレージサーバー (Vserver) で十分な領域を確保して、ボリュームの作成要求に対応できるようにします。注記: このオプションは非推奨であり、Mitaka リリースでは reserved_percentage のために削除されます。

設定オプション = デフォルト値	型	説明
netapp_snapmirror_quiesce_timeout = 3600	整数値	フェイルオーバー中に中止する前に、既存の SnapMirror 転送が完了するまで待機する最大時間 (秒単位)。
netapp_storage_family = ontap_cluster	文字列値	ストレージシステムで使用されるストレージファミリータイプ。クラスター Data ONTAP を使用する場合の有効な値は ontap_cluster のみです。
netapp_storage_protocol = None	文字列値	ストレージシステムと共にデータパスで使用するストレージプロトコル。
netapp_transport_type = http	文字列値	ストレージシステムまたはプロキシーサーバーとの通信時に使用されるトランスポートプロトコル。
netapp_vserver = None	文字列値	このオプションは、ブロックストレージボリュームのプロビジョニングを行うストレージクラスター上の仮想ストレージサーバー (Vserver) 名を指定します。
nexenta_blocksize = 4096	整数値	データセットのブロックサイズ
nexenta_chunksize = 32768	整数値	NexentaEdge iSCSI LUN オブジェクトのチャンクサイズ
`nexenta_client_address = `	文字列値	非 VIP サービス用の NexentaEdge iSCSI ゲートウェイクライアントアドレス
nexenta_dataset_compression = on	文字列値	新しい ZFS フォルダーの圧縮値。
nexenta_dataset_dedup = off	文字列値	新しい ZFS フォルダーの重複排除値。
`nexenta_dataset_description = `	文字列値	フォルダーの人間が判読できる説明。
nexenta_encryption = False	ブール値	NexentaEdge iSCSI LUN オブジェクトで暗号化が有効になっているかどうかを定義します。
`nexenta_folder = `	文字列値	cinder がデータセットを作成するフォルダー。
nexenta_group_snapshot_template = group-snapshot-%s	文字列値	グループスナップショット名を生成するテンプレート文字列

設定オプション = デフォルト値	型	説明
`nexenta_host = `	文字列値	NexentaStor Appliance の IP アドレス
nexenta_host_group_prefix = cinder	文字列値	NexentaStor での iSCSI ホストグループの接頭辞
nexenta_iops_limit = 0	整数値	NexentaEdge iSCSI LUN オブジェクト IOPS の制限
`nexenta_iscsi_service = `	文字列値	NexentaEdge iSCSI サービス名
nexenta_iscsi_target_host_group = all	文字列値	ボリュームへのアクセスを許可するホストのグループ
`nexenta_iscsi_target_portal_groups = `	文字列値	NexentaStor ターゲットポータルグループ
nexenta_iscsi_target_portal_port = 3260	整数値	Nexenta アプライアンス iSCSI ターゲットポータルポート
`nexenta_iscsi_target_portals = `	文字列値	Nexenta Stor5 のポータルのコンマ区切りリスト (IP1:port1、IP2:port2 の形式)。ポートはオプションです。デフォルトは 3260 です。例: 10.10.10.1:3267,10.10.1.2
nexenta_lu_writebackcache_disabled = False	ブール値	バッキングストアへの延期書き込み
`nexenta_lun_container = `	文字列値	LUN のバケットの NexentaEdge 論理パス
nexenta_luns_per_target = 100	整数値	iSCSI ターゲットごとの LUN の量
nexenta_mount_point_base = $state_path/mnt	文字列値	NFS 共有マウントポイントを含むベースディレクトリー
nexenta_nbd_symlinks_dir = /dev/disk/by-path	文字列値	NBD へのシンボリックリンクを格納するディレクトリーの NexentaEdge 論理パス
nexenta_nms_cache_volroot = True	ブール値	True キャッシュの NexentaStor アプライアンスの volroot オプションの値が設定されている場合。
nexenta_ns5_blocksize = 32	整数値	データセットのブロックサイズ
nexenta_origin_snapshot_template = origin-snapshot-%s	文字列値	クローンの起点名を生成するテンプレート文字列

設定オプション = デフォルト値	型	説明
nexenta_password = nexenta	文字列値	NexentaStor 管理 REST API サーバーに接続するためのパスワード
nexenta_qcow2_volumes = False	ブール値	raw ファイルではなく QCOW2 ファイルとしてのボリュームの作成
nexenta_replication_count = 3	整数値	NexentaEdge iSCSI LUN オブジェクトレプリケーションの数。
`nexenta_rest_address = `	文字列値	NexentaStor 管理 REST API エンドポイントの IP アドレス
nexenta_rest_backoff_factor = 0.5	浮動小数点の値	接続試行を NexentaStor 管理 REST API サーバーに適用するバックオフ係数を指定します。
nexenta_rest_connect_timeout = 30	浮動小数点の値	Nexenta Stor 管理 RESTAPI サーバーへの接続を確立する必要がある制限時間 (秒単位) を指定します
nexenta_rest_password = nexenta	文字列値	NexentaEdge に接続するパスワード。
nexenta_rest_port = 0	整数値	NexentaStor 管理 REST API サーバーに接続するための HTTP(S) ポート。ゼロの場合、HTTPS の場合は 8443、HTTP の場合は 8080 が使用されます。
nexenta_rest_protocol = auto	文字列値	Nexenta Stor 管理 RESTAPI 接続には http または https を使用します (デフォルトは自動)
nexenta_rest_read_timeout = 300	浮動小数点の値	Nexenta Stor 管理 RESTAPI サーバーが応答を送信する必要がある制限時間 (秒単位) を指定します
nexenta_rest_retry_count = 3	整数値	接続エラーおよび NexentaStor アプライアンス EBUSY または ENOENT エラーが発生した場合に NexentaStor 管理 REST API 呼び出しを繰り返す回数を指定します。
nexenta_rest_user = admin	文字列値	NexentaEdge に接続するためのユーザー名。
nexenta_rrmgr_compression = 0	整数値	ストリーム圧縮、レベル 1..9 を有効にします。1 が最速となります。9 は圧縮が最大となります。
nexenta_rrmgr_connections = 2	整数値	TCP 接続の数。
nexenta_rrmgr_tcp_buf_size = 4096	整数値	KiloBytes の TCP バッファーサイズ。

設定オプション = デフォルト値	型	説明
nexenta_shares_config = /etc/cinder/nfs_shares	文字列値	利用可能な nfs 共有のリストが含まれるファイル
nexenta_sparse = False	ブール値	スパースデータセットの作成を有効または無効にします。
nexenta_sparsed_volumes = True	ブール値	領域を使用しないスパースファイルとしてボリュームの作成を有効または無効にします。無効 (False) の場合、ボリュームは通常のファイルとして作成されますが、これには時間がかかります。
nexenta_target_group_prefix = cinder	文字列値	NexentaStor での iSCSI ターゲットグループの接頭辞
nexenta_target_prefix = iqn.1986-03.com.sun:02:cinder	文字列値	NexentaStor iSCSI ターゲット用の IQN 接頭辞
nexenta_use_https = True	ブール値	NexentaStor 管理 REST API コネクションに HTTP セキュアプロトコルを使用
nexenta_user = admin	文字列値	NexentaStor 管理 REST API サーバーに接続するためのユーザー名
nexenta_volume = cinder	文字列値	すべてのボリュームを保持する NexentaStor プール名
nexenta_volume_group = iscsi	文字列値	NexentaStor5 iSCSI のボリュームグループ
nfs_mount_attempts = 3	整数値	エラーが発生するまで、NFS 共有のマウントを試行する回数。指定された値に関係なく、少なくとも 1 つの NFS 共有をマウントしようとします。
nfs_mount_options = None	文字列値	NFS クライアントに渡されるマウントオプション。詳細については、NFS(5) のマニュアルページを参照してください。
nfs_mount_point_base = $state_path/mnt	文字列値	NFS 共有のマウントポイントが含まれるベース dir。
nfs_qcow2_volumes = False	ブール値	raw ファイルではなく QCOW2 ファイルとしてボリュームを作成します。

設定オプション = デフォルト値	型	説明
nfs_shares_config = /etc/cinder/nfs_shares	文字列値	利用可能な NFS 共有のリストが含まれるファイル。
nfs_snapshot_support = False	ブール値	NFS ドライバーでスナップショットのサポートを有効にします。libvirt <1.2.7 を使用するプラットフォームでは、この機能に問題が発生します。
nfs_sparsed_volumes = True	ブール値	容量を取らないスパースファイルとしてボリュームを作成します。False ボリュームが設定されている場合は、通常のファイルとして作成されます。このような場合、ボリュームの作成には時間がかかります。
nimble_pool_name = default	文字列値	Nimble Controller プール名
nimble_subnet_label = *	文字列値	Nimble Subnet Label
nimble_verify_cert_path = None	文字列値	Nimble Array SSL 証明書へのパス
nimble_verify_certificate = False	ブール値	Nimble SSL 証明書を検証するかどうか。
num_iser_scan_tries = 3	整数値	ボリュームを見つけるために i SER ターゲットを再スキャンする最大回数
num_shell_tries = 3	整数値	flakey シェルコマンドの実行を試行する回数
num_volume_device_scan_tries = 3	整数値	ボリュームを検索するためにターゲットを再スキャンする最大回数
nvmeof_conn_info_version = 1	整数値	NVMe os-brick コネクターには 2 種類の接続情報形式があり、これにより、spdk や LVM-nvmet などの元の形式 (バージョン 1) を使用する一部の NVMe-oF ドライバーは、新しい形式を送信できるようになります。
nvmet_ns_id = 10	整数値	ターゲットを共有していない場合の LVM ボリュームのサブシステムのネームスペース ID。共有時の最小 ID 値。Linux でサポートされる最大値は 8192 です。

設定オプション = デフォルト値	型	説明
nvmet_port_id = 1	ポート値	ターゲットを共有していない場合の NVMe ターゲットポート定義の ID。共有時の開始ポート ID 値。セカンダリー IP アドレスごとに増加します。
port_group_load_metric = PercentBusy	文字列値	ポートグループの負荷計算に使用されるメトリック。
port_load_metric = PercentBusy	文字列値	ポート負荷の計算に使用されるメトリック。
powerflex_allow_migration_during_rebuild = False	ブール値	再ビルド時のボリューム移行を許可します。
powerflex_allow_non_padded_volumes = False	ブール値	ゼロパディングが無効になると、ストレージプールでボリュームを作成できるようにします。複数のテナントが共有ストレージプールのボリュームを使用する場合には、このオプションを有効にしないでください。
powerflex_max_over_subscription_ratio = 10.0	浮動小数点の値	ドライバーの max_over_subscription_ratio 設定。許可される最大値は 10.0.
powerflex_rest_server_port = 443	ポート値	ゲートウェイ REST サーバーポート。
powerflex_round_volume_capacity = True	ブール値	ボリュームサイズを最大 8GB の境界まで丸めます。PowerFlex/VxFlex OS では、8GB の倍数でボリュームのサイズを指定する必要があります。False に設定すると、ボリュームのサイズが正しくない場合にボリュームの作成に失敗します。
powerflex_server_api_version = None	文字列値	PowerFlex/ScaleIO API バージョン。この値は、テクニカルサポートで特に指示がない限り、デフォルト値のままにする必要があります。
powerflex_storage_pools = None	文字列値	ストレージプールボリュームを提供するために使用されるストレージプールのコンマ区切りリスト。各プールは protection_domain_name:storage_pool_name の値として指定する必要があります。
powerflex_unmap_volume_before_deletion = False	ブール値	削除する前にボリュームのマッピングを解除します。
powermax_array = None	文字列値	接続するアレイのシリアル番号。

設定オプション = デフォルト値	型	説明
powermax_array_tag_list = None	リスト値	ストレージアレイのユーザー割り当て名のリスト。
powermax_port_group_name_template = portGroupName	文字列値	ポートグループ名に対してユーザー定義のオーバーライド。
powermax_port_groups = None	リスト値	サーバー接続前に設定されたフロントエンドポートが含まれるポートグループのリスト。
powermax_service_level = None	文字列値	プロビジョニングストレージに使用するサービスレベル。これを pool_name の追加スペックとして設定することが推奨されます。
powermax_short_host_name_template = shortHostName	文字列値	短縮ホスト名に対するユーザー定義オーバーライド。
powermax_srp = None	文字列値	プロビジョニングに使用するアレイ上のストレージリソースプール。
powerstore_appliances = []	リスト値	アプライアンスの名前。ボリュームのプロビジョニングに使用する PowerStore アプライアンス名のコンマ区切りリスト。**非推奨:** Wallaby *理由:* もう使用されていません。代わりに、PowerStore Load Balancer を使用してボリュームをプロビジョニングします。
powerstore_ports = []	リスト値	許可されるポート。PowerStore iSCSI IP または FC WWN (ex) のコンマ区切りのリスト。使用する 58:cc:f0:98:49:22:07:02)オプションが設定されていない場合は、すべてのポートが許可されます。
proxy = cinder.volume.drivers.ibm.ibm_storage.proxy.IBMStorageProxy	文字列値	IBM Storage アレイに接続するプロキシードライバー
pure_api_token = None	文字列値	REST API 認証トークン。
pure_automatic_max_oversubscription_ratio = True	ブール値	現在のデータ削減値に基づいて、オーバーサブスクリプション比率を自動的に決定します。この計算値を使用すると、max_over_subscription_ratio 設定オプションが上書きされます。

設定オプション = デフォルト値	型	説明
pure_eradicate_on_delete = False	ブール値	有効にすると、Cinder で削除時にすべての Pure ボリューム、スナップショット、および保護グループが消去されます。この設定を True に設定して削除しても、データは回復できません。無効にすると、ボリュームおよびスナップショットは保留中の削除 (eradication) 状態になり、復元できます。
pure_host_personality = None	文字列値	Purity システムがどのようにアレイとイニシエーター間で使用されるプロトコルを調整する方法を決定します。
pure_iscsi_cidr = 0.0.0.0/0	文字列値	FlashArray iSCSI ターゲットホストの CIDR は接続できます。デフォルトでは、任意の IPv4 アドレスへの接続が許可されます。このパラメーターは IPv6 サブネットをサポートするようになりました。pure_iscsi_cidr_list が設定されている場合は無視されます。
pure_iscsi_cidr_list = None	リスト値	ホストが接続を許可される FlashArray iSCSI ターゲットの CIDR のコンマ区切りリスト。IPv4 および IPv6 サブネットをサポートします。このパラメーターは、pure_iscsi_cidr よりも優先されます。
pure_replica_interval_default = 3600	整数値	秒単位のスナップショットレプリケーション間隔。
pure_replica_retention_long_term_default = 7	整数値	この時間 (日数) のターゲットで 1 日あたりのスナップショットを保持します。
pure_replica_retention_long_term_per_day_default = 3	整数値	各日のスナップショット数を保持します。
pure_replica_retention_short_term_default = 14400	整数値	この間 (秒単位)、すべてのスナップショットをターゲットに保持します。
pure_replication_pg_name = cinder-group	文字列値	非同期レプリケーションに使用する純粋な保護グループ名 (存在しない場合は作成されます)。
pure_replication_pod_name = cinder-pod	文字列値	同期レプリケーションに使用する純粋な Pod 名 (存在しない場合は作成されます)。
pvme_iscsi_ips = []	リスト値	コンマ区切りのターゲット iSCSI IP アドレスのリスト。
pvme_pool_name = A	文字列値	ボリュームの作成に使用するプールまたは Vdisk 名。

設定オプション = デフォルト値	型	説明
qnap_management_url = None	URI 値	QNAP ストレージを管理する URL。ドライバーは URL の IPv6 アドレスをサポートしません。
qnap_poolname = None	文字列値	QNAP ストレージのプール名
qnap_storage_protocol = iscsi	文字列値	QNAP ストレージにアクセスするための通信プロトコル
quobyte_client_cfg = None	文字列値	Quobyte クライアント設定ファイルへのパス。
quobyte_mount_point_base = $state_path/mnt	文字列値	Quobyte ボリュームのマウントポイントが含まれるベース dir。
quobyte_overlay_volumes = False	ブール値	フルコピーではなくオーバーレイファイルを作成し、volume_from_snapshot_cache から新しいボリュームを作成します。これにより、このキャッシュからボリュームの作成が速くなります。この機能では、quobyte_qcow2_volumes オプションおよび quobyte_volume_from_snapshot_cache オプションを True に設定する必要があります。上記のいずれかが False に設定されている場合、このオプションは無視されます。
quobyte_qcow2_volumes = True	ブール値	raw ファイルではなく QCOW2 ファイルとしてボリュームを作成します。
quobyte_sparsed_volumes = True	ブール値	領域を使用しないスパースファイルとしてボリュームを作成します。False に設定すると、ボリュームは通常のファイルとして作成されます。
quobyte_volume_from_snapshot_cache = False	ブール値	マージされたスナップショットからボリュームのキャッシュを作成し、1 つのスナップショットから複数のボリュームの作成を高速化します。
quobyte_volume_url = None	文字列値	DNS の SRV レコード (推奨)、または quobyte://<DIR host1>, <DIR host2>/<volume name> のようなホストリスト (代替) を使用して、Quobyte ボリュームの URL を指定します。
rados_connect_timeout = -1	整数値	ceph クラスターへの接続時に使用されるタイムアウト値 (秒単位)。値が 0 未満の場合、タイムアウトは設定されず、デフォルトの librados 値が使用されます。

設定オプション = デフォルト値	型	説明
rados_connection_interval = 5	整数値	Ceph クラスターへの接続再試行の間隔値 (秒単位)。
rados_connection_retries = 3	整数値	Ceph クラスターへの接続に失敗した場合の再試行回数。
`rbd_ceph_conf = `	文字列値	Ceph 設定ファイルへのパス
rbd_cluster_name = ceph	文字列値	Ceph クラスターの名前
rbd_exclusive_cinder_pool = True	ブール値	プールが他の用途と共有されている場合は、False に設定します。排他的な使用の場合には、イメージのプロビジョニングしたサイズを排他的にクエリーしません。これは、assigned_capacity_gb の Cinder コアコードで算出される値と一致します。これにより、Ceph クラスターやボリュームサービスの負荷が軽減されます。非排他的使用では、ドライバーはイメージごとに使用されるディスクを Ceph クラスターにクエリーします。これは、イメージごとに独立した要求を持つ集中的な操作です。
rbd_flatten_volume_from_snapshot = False	ブール値	スナップショットから作成されたボリュームをフラット化して、ボリュームからスナップショットへの依存関係を削除する
rbd_iscsi_api_debug = False	ブール値	クライアント要求のデバッグを有効にします。
`rbd_iscsi_api_password = `	文字列値	rbd_target_api サービスのユーザー名
`rbd_iscsi_api_url = `	文字列値	rbd_target_api サービスへの URL
`rbd_iscsi_api_user = `	文字列値	rbd_target_api サービスのユーザー名
rbd_iscsi_target_iqn = None	文字列値	iscsi ゲートウェイで事前設定された target_iqn。
rbd_max_clone_depth = 5	整数値	フラット化が発生する前に取られるネストされたボリュームクローンの最大数。クローン作成を無効にするには 0 に設定します。注記: この値は低くても、クローンの深度が新しい値を超える既存のボリュームには影響を与えません。
rbd_pool = rbd	文字列値	rbd ボリュームが保存される RADOS プール
rbd_secret_uuid = None	文字列値	rbd_user ボリュームのシークレットの libvirt uuid

設定オプション = デフォルト値	型	説明
rbd_store_chunk_size = 4	整数値	ボリュームは、このサイズ (メガバイト単位) のオブジェクトにチャンク化されます。
rbd_user = None	文字列値	rbd ボリュームにアクセスするための RADOS クライアント名。cephx 認証を使用する場合のみ設定されます。
remove_empty_host = False	ブール値	最後の LUN が切り離された時に、そのホストを Unity から削除します。デフォルトでは False です。
replication_connect_timeout = 5	整数値	ボリュームの降格/昇格を行うために ceph クラスターに接続するときに使用されるタイムアウト値 (秒単位)。値が 0 未満の場合、タイムアウトは設定されず、デフォルトの librados 値が使用されます。
replication_device = None	dict 値	レプリケーションターゲットデバイスを表すディクショナリーのマルチオプト。このオプションは、1 つの config セクションで複数回指定して、複数のレプリケーションターゲットデバイスを指定できます。各エントリーは、以下の標準の dict 設定形式を使います: replication_device = target_device_id: <required>,key1:value1,key2:value2...
report_discard_supported = False	ブール値	Cinder のクライアントに、バックエンドが破棄 (別名: trim/unmap) をサポートしていることを報告します。これは実際にバックエンドやクライアントの動作を直接変更するものではなく、使用可能であることを通知するだけです。
report_dynamic_total_capacity = True	ブール値	ドライバーが動的な値 (使用済み+現在の空き) として合計容量を報告する場合は True に設定し、静的な値 (定義されている場合はクォータ最大バイト数、定義されていない場合はクラスターのグローバルサイズ) を報告する場合は False に設定します。
reserved_percentage = 0	整数値	バックエンド容量のパーセンテージが予約される
retries = 200	整数値	この値を使用して、再試行回数を指定します。
san_api_port = None	ポート値	SAN API へのアクセスに使用するポート
`san_clustername = `	文字列値	ボリュームの作成に使用するクラスター名
`san_ip = `	文字列値	SAN コントローラーの IP アドレス

設定オプション = デフォルト値	型	説明
san_is_local = False	ブール値	SSH ではなくコマンドをローカルで実行します。ボリュームサービスが SAN デバイスで実行している場合は を使用します。
san_login = admin	文字列値	SAN コントローラーのユーザー名
`san_password = `	文字列値	SAN コントローラーのパスワード
`san_private_key = `	文字列値	SSH 認証に使用する秘密鍵のファイル名
san_ssh_port = 22	ポート値	SAN で使用する SSH ポート
san_thin_provision = True	ブール値	SAN ボリュームにはシンプロビジョニングを使用しますか？
scst_target_driver = iscsi	文字列値	SCST ターゲット実装は、複数の SCST ターゲットドライバーから選択できます。
scst_target_iqn_name = None	文字列値	特定の ISCSI ターゲットには事前に定義されたターゲット名があり、SCST ターゲットドライバーはこの名前を使用します。
seagate_iscsi_ips = []	リスト値	コンマ区切りのターゲット iSCSI IP アドレスのリスト。
seagate_pool_name = A	文字列値	ボリュームの作成に使用するプールまたは vdisk 名。
seagate_pool_type = virtual	文字列値	リニア (vdisk の場合) または仮想 (仮想プールの場合)。
`secondary_san_ip = `	文字列値	セカンダリー DSM コントローラーの IP アドレス
secondary_san_login = Admin	文字列値	セカンダリー DSM ユーザー名
`secondary_san_password = `	文字列値	セカンダリー DSM ユーザーのパスワード名
secondary_sc_api_port = 3033	ポート値	セカンダリー Dell API ポート
sf_account_prefix = None	文字列値	この接頭辞を使用して SolidFire アカウントを作成します。ここでは任意の文字列を使用できますが、文字列 hostname は特殊であり、cinder ノードのホスト名を使用して接頭辞を作成します (以前のデフォルトの動作)。デフォルトは接頭辞なしです。

設定オプション = デフォルト値	型	説明
sf_allow_tenant_qos = False	ブール値	テナントが作成時に QoS を指定できるようにする
sf_api_port = 443	ポート値	SolidFire API ポートデバイス API が別のポートのプロキシーの背後にある場合に役立ちます。
sf_api_request_timeout = 30	整数値	API 要求が完了するまで待機する時間を秒単位で設定します。
sf_cluster_pairing_timeout = 60	整数値	クラスターのペアリングが完了するまで待機する時間を秒単位で設定します。
sf_emulate_512 = True	ブール値	ボリュームの作成時に 512 バイトエミュレーションを設定します。
sf_enable_vag = False	ブール値	テナントごとにボリュームアクセスグループを利用します。
sf_provisioning_calc = maxProvisionedSpace	文字列値	SolidFire が使用領域とプロビジョニング計算をどのように報告するかを変更します。このパラメーターが **usedSpace** に設定されると、ドライバーは Cinder シンプロビジョニングで想定した正しい値をレポートします。
sf_svip = None	文字列値	デフォルトのクラスター SVIP を指定されたクラスター SVIP で上書きします。これは、クラウドで iSCSI ネットワークに VLAN を使用するための VLAN の使用を実装した必須またはデプロイメントです。
sf_volume_clone_timeout = 600	整数値	ボリュームまたはスナップショットのクローンが完了するまで待機する時間を秒単位で設定します。
sf_volume_create_timeout = 60	整数値	ボリューム作成操作が完了するまで待機する時間を秒単位で設定します。
sf_volume_pairing_timeout = 3600	整数値	移行中のボリュームがペアリングと同期を完了するまで待機する時間を秒単位で設定します。
sf_volume_prefix = UUID-	文字列値	この接頭辞を使用して、SolidFire ボリュームを作成します。ボリューム名は <sf_volume_prefix><cinder-volume-id> の形式になります。デフォルトでは、接頭辞として **UUID-** が使用されます。
smbfs_default_volume_format = vhd	文字列値	ボリューム形式が指定されていない場合に、ボリュームの作成時に使用されるデフォルトの形式。

設定オプション = デフォルト値	型	説明
smbfs_mount_point_base = C:\OpenStack_mnt	文字列値	smbfs 共有のマウントポイントを含むベース dir。
smbfs_pool_mappings = {}	dict 値	共有の場所とプール名間のマッピング。指定しない場合、共有名はプール名として使用されます。例: //addr/share:pool_name,//addr/share2:pool_name2
smbfs_shares_config = C:\OpenStack\smbfs_shares.txt	文字列値	使用可能な smbfs 共有のリストをファイルします。
spdk_max_queue_depth = 64	整数値	rdma トランスポートのキュー深度。
spdk_rpc_ip = None	文字列値	NVMe ターゲットリモート設定 IP アドレス。
spdk_rpc_password = None	文字列値	NVMe ターゲットリモート設定パスワード。
spdk_rpc_port = 8000	ポート値	NVMe ターゲットリモート設定ポート。
spdk_rpc_protocol = http	文字列値	SPDK RPC プロキシーで使用するプロトコル
spdk_rpc_username = None	文字列値	NVMe ターゲットリモート設定のユーザー名。
ssh_conn_timeout = 30	整数値	SSH 接続タイムアウト (秒単位)
ssh_max_pool_conn = 5	整数値	プールの ssh 接続の最大数
ssh_min_pool_conn = 1	整数値	プールの最小 ssh 接続
storage_protocol = iscsi	文字列値	ホストとストレージバックエンド間でデータを転送するためのプロトコル。
storage_vnx_authentication_type = global	文字列値	VNX 認証スコープタイプ。デフォルトでは、値は global です。
storage_vnx_pool_names = None	リスト値	使用されるストレージプール名のコンマ区切りリスト。
storage_vnx_security_file_dir = None	文字列値	VNX セキュリティーファイルが含まれるディレクトリーパス。セキュリティーファイルが最初に生成されることを確認します。

設定オプション = デフォルト値	型	説明
storpool_replication = 3	整数値	デフォルトの StorPool チェーンレプリケーション値。storpool_template が設定されていない場合は、指定したタイプでボリュームを作成する際に使用されます。統計で報告された明らかな空き領域の計算にも使われます。
storpool_template = None	文字列値	タイプのないボリュームの StorPool テンプレート。
storwize_peer_pool = None	文字列値	hyperswap ボリュームのピアプールの名前を指定します。ピアプールは他のサイトに存在する必要があります。
storwize_portset = None	文字列値	ホストの作成先となるポートセットの名前を指定します。
storwize_preferred_host_site = {}	dict 値	ホストのサイト情報を指定します。ホストで使用される WWPN または複数 WWPN のいずれかを指定できます。例: storwize_preferred_host_site=site1:wwpn1,site2:wwpn2&wwpn3 または storwize_preferred_host_site=site1:iqn1,site2:iqn2
storwize_san_secondary_ip = None	文字列値	san_ip が無効であるか、アクセスできなくなる場合に使用されるセカンダリー管理 IP またはホスト名を指定します。
storwize_svc_allow_tenant_qos = False	ブール値	テナントが作成時に QoS を指定できるようにする
storwize_svc_flashcopy_rate = 50	整数値	完全なボリュームコピーの作成時に使用される Storwize FlashCopy のコピーレートを指定します。デフォルトレート は 50 で、有効なレートは 1 - 150 です。
storwize_svc_flashcopy_timeout = 120	整数値	FlashCopy の作成を待機する最大秒数。
storwize_svc_iscsi_chap_enabled = True	ブール値	iSCSI 接続に CHAP 認証を設定する (デフォルト: Enabled)。
storwize_svc_mirror_pool = None	文字列値	ミラーリングされたコピーが保存されるプールの名前を指定します。例: "pool2"
storwize_svc_multihostmap_enabled = True	ブール値	このオプションは影響を受けなくなりました。これは非推奨で、次のリリースで削除されます。

設定オプション = デフォルト値	型	説明
storwize_svc_multipath_enabled = False	ブール値	マルチパスで接続 (FC のみ。iSCSI マルチパスは Nova によって制御されます)
storwize_svc_retain_aux_volume = False	ブール値	プライマリーストレージ上のボリュームの削除中、またはレプリケーションが有効な状態でプライマリーボリュームをミラーから非ミラーに移動する際に、セカンダリーストレージ上の補助ボリュームの保持を有効または無効にします。このオプションは、Spectrum Virtualize ファミリーで有効です。
storwize_svc_stretched_cluster_partner = None	文字列値	ストレッチクラスターモードで動作している場合は、ミラーリングされたコピーが保存されているプールの名前を指定します。例: pool2
storwize_svc_vol_autoexpand = True	ブール値	ボリュームのストレージシステムの自動デプロイメントパラメーター (True/False)
storwize_svc_vol_compression = False	ブール値	ボリュームのストレージシステム圧縮オプション
storwize_svc_vol_easytier = True	ブール値	ボリュームの簡単な階層の有効化
storwize_svc_vol_grainsize = 256	整数値	ボリュームのストレージシステムの粒度パラメーター (8/32/64/128/256)
storwize_svc_vol_iogrp = 0	文字列値	ボリュームを割り当てる I/O グループ。これには、コンマ区切りのリストがあり、その場合は、io_group に関連付けられたボリュームの数が最も少ないボリュームに基づいて io_group を選択します。
storwize_svc_vol_nofmtdisk = False	ブール値	ボリュームの作成時にフォーマットされないことを指定します。
storwize_svc_vol_rsize = 2	整数値	ボリュームに対するストレージシステムの空間効率パラメーター (パーセンテージ)
storwize_svc_vol_warning = 0	整数値	ボリューム容量の警告に対するストレージシステムのしきい値 (パーセンテージ)
storwize_svc_volpool_name = ['volpool']	リスト値	ボリュームのストレージシステムストレージプールのコンマ区切りリスト。
suppress_requests_ssl_warnings = False	ブール値	リクエストライブラリーの SSL 証明書の警告を表示しません。

設定オプション = デフォルト値	型	説明
synology_admin_port = 5000	ポート値	Synology ストレージの管理ポート。
synology_device_id = None	文字列値	OTP が有効な場合に Synology ストレージにログインするためのワンタイムパスワードチェックを行うデバイス ID。
synology_one_time_pass = None	文字列値	OTP が有効な場合、Synology ストレージにログインする管理者のワンタイムパスワード。
`synology_password = `	文字列値	Synology ストレージにログインする管理者のパスワード
`synology_pool_name = `	文字列値	lun の作成に使用する Synology ストレージ上のボリューム。
synology_ssl_verify = True	ブール値	証明書の検証を行うか、$driver_use_ssl が True の場合ではありません。
synology_username = admin	文字列値	Synology ストレージ管理者。
target_helper = tgtadm	文字列値	使用するターゲットユーザーランドツール。tgtadm はデフォルト で、LIO iSCSI サポートに lioadm、SCST ターゲットのサポートには scstadmin、または Chelsio iSCSI ターゲットの場合は ietadm、NVMEoF サポートの場合は nvmet、SPDK NVMe-oF の場合は spdk-nvmeof、テストには fake を使用します。注: IET ドライバーは推奨されておらず、V リリースで削除される予定です。
target_ip_address = $my_ip	文字列値	iSCSI/NVMEoF デーモンがリッスンする IP アドレス。
target_port = 3260	ポート値	iSCSI/NVMEoF デーモンがリッスンするポート。
target_prefix = iqn.2010-10.org.openstack:	文字列値	iSCSI/NVMEoF ボリュームの接頭辞。
target_protocol = iscsi	文字列値	tgtadm、lioadm、および nvmet ターゲットヘルパーで作成した新規ボリュームのターゲットプロトコルを決定します。RDMA を有効にするには、このパラメーターを "iser" の値で設定する必要があります。nvmet ターゲットが "nvmet_rdma" または "nvmet_tcp" に設定されている場合は、サポートされる iSCSI プロトコルの値は "iscsi" および "iser" です。

設定オプション = デフォルト値	型	説明
target_secondary_ip_addresses = []	リスト値	iSCSI/NVMEoF デーモンのセカンダリー IP アドレスのリスト。
thres_avl_size_perc_start = 20	整数値	NFS 共有の利用可能な領域の割合がこのオプションで指定された値以下にドロップすると、NFS イメージキャッシュは消去されます。
thres_avl_size_perc_stop = 60	整数値	NFS 共有上の使用可能な領域の割合が、このオプションで指定された割合に到達すると、ドライバーは、最後の M 分間アクセスされていない NFS イメージキャッシュからファイルを消去を停止します。M は expiry_thres_minutes 設定オプションの値になります。
trace_flags = None	リスト値	開発者を支援する DEBUG ログレベルにどのトレース情報を書き込むかを制御するオプションのリスト。有効な値は method および api です。
u4p_failover_autofailback = True	ブール値	ドライバーが正常な接続が再確立されると、Unisphere プライマリーインスタンスに自動的にフェイルバックするはずです。
u4p_failover_backoff_factor = 1	整数値	2 回目の試行後に試行されるまでのバックオフ係数 (ほとんどのエラーは遅延なく 2 回目の試行によって即座に解決されます)。再試行はスリープ状態になります。{backoff factor} * (2 ^ ({number of total retries} - 1)) 秒。
u4p_failover_retries = 3	整数値	各接続が試行する最大再試行回数。これは、失敗した DNS ルックアップ、ソケット接続、接続のタイムアウトにのみ適用され、サーバーにデータが発生した場所を要求しないことに注意してください。
u4p_failover_target = None	dict 値	Unisphere フェイルオーバーターゲット情報のディクショナリー。
u4p_failover_timeout = 20.0	整数値	サーバーがデータを送信するまで待機する時間。
unique_fqdn_network = True	ブール値	プライベートネットワークが各イニシエーターに固有の FQDN を持つかどうか。たとえば、QA システムを使用するネットワークには、通常、同じ FQDN を持つ複数のサーバー/VM があります。true の場合、FQDN を使用して 3PAR にホストエントリーを作成します。false の場合は、逆の IQN/WWNN を使用します。

設定オプション = デフォルト値	型	説明
unity_io_ports = []	リスト値	使用する iSCSI ポートまたは FC ポートのコンマ区切りのリスト。各ポートは Unix スタイルの glob 表現にすることができます。
unity_storage_pool_names = []	リスト値	使用されるストレージプール名のコンマ区切りリスト。
use_chap_auth = False	ブール値	ターゲットの CHAP 認証を有効/無効にするオプション。
use_multipath_for_image_xfer = False	ブール値	ボリュームとボリュームの譲渡用のマルチパスを使用して cinder のボリュームを接続/割り当て解除しますか。このパラメーターは、すべてのバックエンドに共通の設定として、各バックエンドセクションまたは backend_defaults セクションで設定する必要があります。
vmax_workload = None	文字列値	ワークロードでは、これを pool_name の追加スペックとして設定することが推奨されます。
vmware_adapter_type = lsiLogic	文字列値	ボリュームの割り当てに使用するデフォルトのアダプタータイプ。
vmware_api_retry_count = 10	整数値	接続関連の問題の発生時に VMware vCenter Server API を再試行する必要がある回数。
vmware_ca_file = None	文字列値	vCenter サーバー証明書の検証に使用する CA バンドルファイル。
vmware_cluster_name = None	多値	ボリュームが作成される vCenter コンピュートクラスターの名前。
vmware_connection_pool_size = 10	整数値	http 接続プールの接続の最大数。
vmware_datastore_regex = None	文字列値	バックエンドボリュームが作成されるデータストアの名前に一致する正規表現パターン。
vmware_enable_volume_stats = False	ブール値	true の場合、バックエンドからのボリューム統計の取得が有効になります。これには、大規模なパフォーマンスの問題が発生する可能性があります。False の場合、ドライバーはバックエンドに関する統計を収集しません。
vmware_host_ip = None	文字列値	VMware vCenter サーバーに接続するための IP アドレス。

設定オプション = デフォルト値	型	説明
vmware_host_password = None	文字列値	VMware vCenter サーバーに対して認証するためのパスワード。
vmware_host_port = 443	ポート値	VMware vCenter サーバーに接続するためのポート番号。
vmware_host_username = None	文字列値	VMware vCenter サーバーでの認証用のユーザー名。
vmware_host_version = None	文字列値	VMware vCenter サーバーバージョンを指定するオプションの文字列。ドライバーは、VMware vCenter サーバーからバージョンの取得を試みます。vCenter サーバーのバージョンを上書きする場合にのみ、この設定を設定します。
vmware_image_transfer_timeout_secs = 7200	整数値	Cinder と Glance 間の VMDK ボリューム転送の秒単位のタイムアウト。
vmware_insecure = False	ブール値	true の場合、vCenter サーバー証明書は検証されません。false の場合、デフォルトの CA トラストストアが検証に使用されます。このオプションは、vmware_ca_file が設定されている場合は無視されます。
vmware_lazy_create = True	ブール値	true の場合、ボリュームがソースなしで作成されると、vCenter サーバーのバックエンドボリュームは遅延的に作成されます。バックエンドボリュームは、ボリュームが割り当てられる際に作成され、イメージサービスにアップロードされるか、バックアップ時に作成されます。
vmware_max_objects_retrieval = 100	整数値	バッチごとに取得するオブジェクトの最大数。クエリー結果は、サーバーからバッチで取得され、1 つのショットでは取得されません。また、サーバーは、設定された値よりも小さい数に制限できます。
vmware_snapshot_format = template	文字列値	vCenter サーバーのボリュームスナップショットの形式。
vmware_storage_profile = None	多値	監視するストレージプロファイルの名前。vmware_enable_volume_stats が True の場合にのみ使用されます。
vmware_task_poll_interval = 2.0	浮動小数点の値	VMware v Center サーバーで呼び出されたリモートタスクをポーリングする間隔 (秒単位)。

設定オプション = デフォルト値	型	説明
vmware_tmp_dir = /tmp	文字列値	ボリュームのバックアップおよび復元時に仮想ディスクが保存されるディレクトリー。
vmware_volume_folder = Volumes	文字列値	Cinder ボリュームが含まれる vCenter インベントリーフォルダーの名前。このフォルダーは OpenStack/<project_folder>の下に作成されます。ここで、project_folder の形式は "Project (<volume_project_id>)" になります。
vmware_wsdl_location = None	文字列値	オプションの VIM サービスの WSDL Location (http://<server>/vimService.wsdl)任意設定: バグワークアラウンドの場合はデフォルトの場所にします。
vnx_async_migrate = True	ブール値	ボリュームのクローン作成およびスナップショットからの作成時には常に非同期移行を使用します。設定ドキュメントで説明されているように、非同期移行にはいくつかの制約があります。メタデータを使用する他に、このオプションを使用して非同期移行を無効にすることもできます。メタデータの **async_migrate** は、両方が設定されている場合にこのオプションを上書きすることに注意してください。デフォルトでは、値は True です。
volume_backend_name = None	文字列値	特定のドライバー実装のバックエンド名
volume_clear = zero	文字列値	古いボリュームの消去に使用するメソッド
volume_clear_ionice = None	文字列値	削除後のボリュームをゼロにするために使用されるプロセスの i/o 優先度を変更するために ionice に渡すフラグで、例えば-c3 はアイドルのみの優先度です。
volume_clear_size = 0	整数値	古いボリュームの開始時に消去する MiB 単位のサイズ。1024 MiB at max. 0 ⇒ all
volume_copy_blkio_cgroup_name = cinder-volume-copy	文字列値	ボリュームコピーの帯域幅を制限するために使用される blkio cgroup 名
volume_copy_bps_limit = 0	整数値	ボリュームコピーの帯域幅の上限。0 ⇒ 無制限
volume_dd_blocksize = 1M	文字列値	ボリュームのコピー/消去時に使用されるデフォルトのブロックサイズ

設定オプション = デフォルト値	型	説明
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver	文字列値	ボリュームの作成に使用するドライバー
volume_group = cinder-volumes	文字列値	エクスポートされたボリュームが含まれる VG の名前
volumes_dir = $state_path/volumes	文字列値	ボリューム設定ファイルのストレージディレクトリー
vxflexos_allow_migration_during_rebuild = False	ブール値	powerflex_allow_migration_during_rebuild に変更されました。
vxflexos_allow_non_padded_volumes = False	ブール値	powerflex_allow_non_padded_volumes に変更されました。
vxflexos_max_over_subscription_ratio = 10.0	浮動小数点の値	powerflex_max_over_subscription_ratio に変更されました。
vxflexos_rest_server_port = 443	ポート値	powerflex_rest_server_port に変更されました。
vxflexos_round_volume_capacity = True	ブール値	powerflex_round_volume_capacity に名前を変更します。
vxflexos_server_api_version = None	文字列値	powerflex_server_api_version に変更されました。
vxflexos_storage_pools = None	文字列値	powerflex_storage_pools に変更されました。
vxflexos_unmap_volume_before_deletion = False	ブール値	powerflex_round_volume_capacity に名前を変更します。
vzstorage_default_volume_format = raw	文字列値	ボリューム形式が指定されていない場合に、ボリュームの作成時に使用されるデフォルトの形式。
vzstorage_mount_options = None	リスト値	vzstorage クライアントに渡されるマウントオプション。詳細は、man ページの pstorage-mount セクションを参照してください。
vzstorage_mount_point_base = $state_path/mnt	文字列値	vzstorage 共有のマウントポイントが含まれるベース dir。

設定オプション = デフォルト値	型	説明
vzstorage_shares_config = /etc/cinder/vzstorage_shares	文字列値	利用可能な vzstorage 共有のリストが含まれるファイル。
vzstorage_sparsed_volumes = True	ブール値	raw 形式を使用する場合は、通常のファイルではなく、スペースを取り除くボリュームをスパースファイルとして作成します。この場合、ボリュームの作成には多くの時間がかかります。
vzstorage_used_ratio = 0.95	浮動小数点の値	ボリュームの宛先に新規ボリュームを割り当てないようにする前の、基礎となるボリュームの ACTUAL 使用量のパーセント。
windows_iscsi_lun_path = C:\iSCSIVirtualDisks	文字列値	VHD 対応ボリュームを保存するパス
xtremio_array_busy_retry_count = 5	整数値	アレイがビジー状態になる場合の再試行回数
xtremio_array_busy_retry_interval = 5	整数値	アレイがビジー状態になる場合の再試行の間隔
xtremio_clean_unused_ig = False	ブール値	最後の接続の終了後に、ドライバーがボリュームを持たないイニシエーターグループを削除する必要があります。これまでの動作は IG をそのままにしておくことであったため、デフォルトで False に設定されています (ボリュームが接続されていない IG は削除されません)。このパラメーターを True に設定すると、最後のボリュームへの接続が終了した後、IG が削除されます。
`xtremio_cluster_name = `	文字列値	マルチクラスター環境の XMS クラスター ID
xtremio_ports = []	リスト値	許可されるポート。XtremIO iSCSI IP または FC WWN (ex) のコンマ区切りのリスト。使用する 58:cc:f0:98:49:22:07:02)オプションが設定されていない場合は、すべてのポートが許可されます。
xtremio_volumes_per_glance_cache = 100	整数値	キャッシュされた各 glance イメージから作成されたボリュームの数
zadara_access_key = None	文字列値	VPSA アクセスキー

設定オプション = デフォルト値	型	説明
zadara_default_snap_policy = False	ブール値	VPSA - ボリュームのスナップショットポリシーを添付します。オプションが設定されておらず、メタデータとして提供されていない場合、VPSA はデフォルト値を継承します。
zadara_gen3_vol_compress = False	ブール値	VPSA - ボリュームの圧縮を有効にします。オプションが設定されておらず、メタデータとして提供されていない場合、VPSA はデフォルト値を継承します。
zadara_gen3_vol_dedupe = False	ブール値	VPSA - ボリュームの重複排除を有効にします。オプションが設定されておらず、メタデータとして提供されていない場合、VPSA はデフォルト値を継承します。
zadara_ssl_cert_verify = True	ブール値	True に設定すると、http クライアントは VPSA エンドポイントの SSL 証明書を検証します。
zadara_vol_encrypt = False	ブール値	VPSA - ボリュームのデフォルトの暗号化ポリシー。オプションが設定されておらず、メタデータとして提供されていない場合、VPSA はデフォルト値を継承します。
zadara_vpsa_host = None	ホストのアドレス値	VPSA - 管理ホスト名または IP アドレス
zadara_vpsa_poolname = None	文字列値	VPSA: ボリュームに割り当てられたストレージプール
zadara_vpsa_port = None	ポート値	VPSA: ポート番号
zadara_vpsa_use_ssl = False	ブール値	VPSA: SSL 接続を使用

2.1.4. barbican

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[barbican]** グループで使用できるオプションの概要をまとめたものです。

表2.3 barbican

設定オプション = デフォルト値	型	説明
auth_endpoint = http://localhost/identity/v3	文字列値	このエンドポイントを使用して Keystone に接続します。

設定オプション = デフォルト値	型	説明
barbican_api_version = None	文字列値	Barbican API のバージョン。例: "v1"
barbican_endpoint = None	文字列値	このエンドポイントを使用して、Barbican に接続します (例: http://localhost:9311/)。
barbican_endpoint_type = public	文字列値	エンドポイントのタイプを指定します。使用できる値は public、private、および admin です。
number_of_retries = 60	整数値	キー作成の完了のためにポーリングを再試行する回数
retry_delay = 1	整数値	キー作成の完了をポーリングを再試行するまで待機する秒数
verify_ssl = True	ブール値	セキュアでない TLS (https) 要求を指定します。False の場合、サーバーの証明書は検証されません。True の場合は、verify_ssl_path 設定の平均期間を設定できます。
verify_ssl_path = None	文字列値	チェックするバンドルまたは CA 証明書へのパス、または verify_ssh が True である証明書を見つけて使用しようとする要求の場合は None。verify_ssl が False の場合、これは無視されます。

2.1.5. brcd_fabric_example

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[brcd_fabric_example]** グループで使用できるオプションの概要をまとめたものです。

表2.4 brcd_fabric_example

設定オプション = デフォルト値	型	説明
`fc_fabric_address = `	文字列値	ファブリックの管理 IP。
`fc_fabric_password = `	文字列値	ユーザーのパスワード
fc_fabric_port = 22	ポート値	接続ポート
`fc_fabric_ssh_cert_path = `	文字列値	ローカルの SSH 証明書パス。
`fc_fabric_user = `	文字列値	ファブリックユーザー ID。

設定オプション = デフォルト値	型	説明
fc_southbound_protocol = REST_HTTP	文字列値	ファブリック用のサウスバウンドコネクター。
fc_virtual_fabric_id = None	文字列値	仮想ファブリック ID。
zone_activate = True	ブール値	アクティベーションの状態を上書きします。
zone_name_prefix = openstack	文字列値	ゾーン名の接頭辞を上書きします。
zoning_policy = initiator-target	文字列値	zoning ポリシーを上書きします。

2.1.6. cisco_fabric_example

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[cisco_fabric_example]** グループで使用できるオプションの概要をまとめたものです。

表2.5 cisco_fabric_example

設定オプション = デフォルト値	型	説明
`cisco_fc_fabric_address = `	文字列値	ファブリックの管理 IP
`cisco_fc_fabric_password = `	文字列値	ユーザーのパスワード
cisco_fc_fabric_port = 22	ポート値	接続ポート
`cisco_fc_fabric_user = `	文字列値	ファブリックユーザー ID
cisco_zone_activate = True	ブール値	アクティベーション状態の上書き
cisco_zone_name_prefix = None	文字列値	オーバーライドされるゾーン名の接頭辞
cisco_zoning_policy = initiator-target	文字列値	zoning ポリシーを上書きします。
cisco_zoning_vsan = None	文字列値	Fabric の VSAN

2.1.7. coordination

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[coordination]** グループで使用できるオプションの概要をまとめたものです。

表2.6 coordination

設定オプション = デフォルト値	型	説明
backend_url = file://$state_path	文字列値	分散調整に使用するバックエンド URL。

2.1.8. cors

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[cors]** グループで使用できるオプションの概要をまとめたものです。

表2.7 cors

設定オプション = デフォルト値	型	説明
allow_credentials = True	ブール値	実際の要求にユーザーの認証情報を含めることができることを示します。
allow_headers = ['X-Auth-Token', 'X-Identity-Status', 'X-Roles', 'X-Service-Catalog', 'X-User-Id', 'X-Tenant-Id', 'X-OpenStack-Request-ID', 'X-Trace-Info', 'X-Trace-HMAC', 'OpenStack-API-Version']	リスト値	実際の要求時に使用されるヘッダーフィールド名を示します。
allow_methods = ['GET', 'PUT', 'POST', 'DELETE', 'PATCH', 'HEAD']	リスト値	実際の要求時に使用できるメソッドを示します。
allowed_origin = None	リスト値	このリソースがリクエストの origin"ヘッダーで受信したドメインと共有されるかどうかを示します。形式:"<protocol>://<host>[:<port>]" (行末のスラッシュなし)例:https://horizon.example.com
expose_headers = ['X-Auth-Token', 'X-Subject-Token', 'X-Service-Token', 'X-OpenStack-Request-ID', 'OpenStack-API-Version']	リスト値	API に安全に公開できるヘッダーを示します。デフォルトは HTTP Simple ヘッダーです。

設定オプション = デフォルト値	型	説明
max_age = 3600	整数値	CORS プリフライトリクエストの最大キャッシュ期間。

2.1.9. database

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[database]** グループで利用可能なオプションの概要をまとめたものです。

表2.8 database

設定オプション = デフォルト値	型	説明
backend = sqlalchemy	文字列値	データベースに使用するバックエンド。
connection = None	文字列値	データベースへの接続に使用する SQLAlchemy 接続文字列。
connection_debug = 0	整数値	SQL デバッグ情報の冗長性: 0=なし、100=すべて。
`connection_parameters = `	文字列値	接続時に接続 URL に追加するオプションの URL パラメーター。param1=value1&param2=value2&... として指定します。
connection_recycle_time = 3600	整数値	この秒数より長く接続プールに存在していた接続は、次にプールからチェックアウトされたときに新しい接続に置き換えられます。
connection_trace = False	ブール値	Python スタックトレースをコメント文字列として SQL に追加します。
db_inc_retry_interval = True	ブール値	True の場合には、db_max_retry_interval までのデータベース操作の再試行回数を長くします。
db_max_retries = 20	整数値	エラーが発生する前に接続エラーまたはデッドロックが発生した場合の最大再試行。再試行回数を無限に指定するには -1 に設定します。
db_max_retry_interval = 10	整数値	db_inc_retry_interval が設定されている場合は、データベース操作を再試行するまでの最大秒数です。
db_retry_interval = 1	整数値	データベーストランザクションの再試行間の秒数。
max_overflow = 50	整数値	設定されている場合、この値を SQLAlchemy で max_overflow に使用します。

設定オプション = デフォルト値	型	説明
max_pool_size = 5	整数値	プールに開いたままにする SQL 接続の最大数。値を 0 に設定すると無制限を意味します。
max_retries = 10	整数値	起動時のデータベース接続の最大再試行数。再試行回数を無限に指定するには -1 に設定します。
mysql_enable_ndb = False	ブール値	True の場合には、MySQL Cluster (NDB) の処理のサポートを透過的に有効にします。
mysql_sql_mode = TRADITIONAL	文字列値	MySQL セッションに使用する SQL モード。このオプション (デフォルトを含む) は、サーバーセット SQL モードを上書きします。サーバー設定で設定された SQL モードを使用するには、これを no value に設定します。例: mysql_sql_mode=
pool_timeout = None	整数値	設定されている場合は、この値を SQLAlchemy で pool_timeout に使用します。
retry_interval = 10	整数値	SQL 接続を開く再試行の間隔。
slave_connection = None	文字列値	スレーブデータベースへの接続に使用する SQLAlchemy 接続文字列。
sqlite_synchronous = True	ブール値	True の場合、SQLite は同期モードを使用します。
use_db_reconnect = False	ブール値	失われた接続でデータベースの実験的な使用を有効にします。

2.1.10. fc-zone-manager

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[fc-zone-manager]** グループで利用可能なオプションの概要をまとめたものです。

表2.9 fc-zone-manager

設定オプション = デフォルト値	型	説明
brcd_sb_connector = HTTP	文字列値	zoning 操作のサウスバウンドコネクター

設定オプション = デフォルト値	型	説明
cisco_sb_connector = cinder.zonemanager.drivers.cisco.cisco_fc_zone_client_cli.CiscoFCZoneClientCLI	文字列値	zoning 操作のサウスバウンドコネクター
enable_unsupported_driver = False	ブール値	サポート対象外のゾーンマネージャードライバーの起動を許可する場合はこれを True に設定します。作業用の CI システムおよびテストを維持していないドライバーは、CI が再び動作するまで、サポート対象外とマークされます。これはドライバーを非推奨としてマークし、次のリリースで削除される可能性があります。
fc_fabric_names = None	文字列値	ファイバーチャネルファブリック名のコンマ区切りリスト。この名前のリストは、各 SAN ファブリックに接続するための他の SAN 認証情報の取得に使用されます。
fc_san_lookup_service = cinder.zonemanager.drivers.brocade.brcd_fc_san_lookup_service.BrcdFCSanLookupService	文字列値	FC SAN ルックアップサービス
zone_driver = cinder.zonemanager.drivers.brocade.brcd_fc_zone_driver.BrcdFCZoneDriver	文字列値	ゾーン管理を行う FC ゾーンドライバー
zoning_policy = initiator-target	文字列値	ユーザーが設定するポリシー。有効な値には "initiator-target" または "initiator" が含まれます。

2.1.11. Healthcheck

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[healthcheck]** グループで利用可能なオプションの概要をまとめたものです。

表2.10 Healthcheck

設定オプション = デフォルト値	型	説明
backends = []	リスト値	ヘルスチェックを実施し、リクエストの一部としてその情報を報告できる追加のバックエンド。

設定オプション = デフォルト値	型	説明
detailed = False	ブール値	応答の一部として詳細情報を表示します。セキュリティーメモ: このオプションを有効にすると、監視対象のサービスに関する機密情報が漏洩する可能性があります。セキュリティーポリシーに違反しないことを確認してください。
disable_by_file_path = None	文字列値	ファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを確認します。DisableByFileHealthcheck プラグインで使用されます。
disable_by_file_paths = []	リスト値	ポートに基づいてファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを判断します。文字列の port:path リストを想定します。DisableByFilesPortsHealthcheck プラグインで使用されます。
path = /healthcheck	文字列値	healtcheck 要求に応答するパス。

2.1.12. key_manager

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[key_manager]** グループで使用できるオプションの概要をまとめたものです。

表2.11 key_manager

設定オプション = デフォルト値	型	説明
auth_type = None	文字列値	作成する認証認証情報のタイプ。使用できる値は、**トークン**、**パスワード**、**keystone_token**、および **keystone_password** です。コンテキストが認証情報ファクトリーに渡されない場合に必要です。
auth_url = None	文字列値	このエンドポイントを使用して Keystone に接続します。
backend = barbican	文字列値	キーマネージャーの実装を指定します。オプションは barbican および vault です。デフォルトは barbican です。しばらくの間、[key_manager]/api_class を使用して以前に設定された値をサポートします。
domain_id = None	文字列値	ドメインスコーピングのドメイン ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。

設定オプション = デフォルト値	型	説明
domain_name = None	文字列値	ドメインスコーピングのドメイン名。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
fixed_key = None	文字列値	16 進法で指定されるキーマネージャーによって返されるキーを修正
password = None	文字列値	認証のパスワード。**パスワード** および **keystone_password** auth_type には必須です。
project_domain_id = None	文字列値	プロジェクトのドメイン ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
project_domain_name = None	文字列値	プロジェクトのドメイン名。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
project_id = None	文字列値	プロジェクトスコーピングのプロジェクト ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
project_name = None	文字列値	プロジェクトスコーピングのプロジェクト名。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
reauthenticate = True	ブール値	現在のトークンの有効期限が切れる場合の新規トークンの取得を許可します。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
token = None	文字列値	認証のトークン。認証情報ファクトリーにコンテキストが渡されない場合には、**トークン** および **keystone_token** auth_type には必須です。
trust_id = None	文字列値	信頼スコープの信頼 ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
user_domain_id = None	文字列値	認証用のユーザーのドメイン ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
user_domain_name = None	文字列値	認証用のユーザーのドメイン名。**keystone_token** および **keystone_password** auth_type の場合はオプションです。

設定オプション = デフォルト値	型	説明
user_id = None	文字列値	認証のユーザー ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
username = None	文字列値	認証用のユーザー名。**パスワード** auth_type には必須です。**keystone_password** auth_type の場合はオプションです。

2.1.13. keystone_authtoken

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[keystone_authtoken]** グループ下で使用できるオプションの概要をまとめたものです。

表2.12 keystone_authtoken

設定オプション = デフォルト値	型	説明
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ
auth_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。このオプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。**非推奨**:Queens 以降 *理由:* auth_uri オプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。
auth_version = None	文字列値	Identity API エンドポイントの API バージョン。

設定オプション = デフォルト値	型	説明
cache = None	文字列値	Swift キャッシュオブジェクトが保存される環境キーを要求します。auth_token ミドルウェアを Swift キャッシュと共にデプロイする場合は、このオプションを使用して、ミドルウェアが swift とキャッシングバックエンドを共有するようにします。それ以外の場合は、代わりに **memcached_servers** オプションを使用します。
cafile = None	文字列値	HTTPs 接続の検証時に使用する PEM でエンコードされた認証局。デフォルトはシステム CA です。
certfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
delay_auth_decision = False	ブール値	ミドルウェア内の認可要求を処理せず、承認の決定をダウンストリームの WSGI コンポーネントに委譲します。
enforce_token_bind = permissive	文字列値	トークンバインディングの使用および種別を制御するために使用されます。トークンバインディングのチェックを無効するには、"disabled"に設定します。バインドタイプがサーバーの認識する形式の場合にはバインディング情報を検証し、そうでない場合には無視するには、"permissive"(デフォルト) に設定します。"strict"は"permissive"と類似していますが、バインドタイプが不明な場合にはトークンが拒否されます。"required"の場合は、いずれかの形式のトークンバインディングが必要です。最後に、トークンに指定する必要のあるバインディングメソッドの名前。
http_connect_timeout = None	整数値	Identity API サーバーと通信する際の要求タイムアウト値。
http_request_max_retries = 3	整数値	Identity API サーバーと通信する際に再接続を試行する回数。
include_service_catalog = True	ブール値	(オプション)X-Service-Catalog ヘッダーを設定するかどうかを示します。False の場合、ミドルウェアはトークンの検証時にサービスカタログを要求せず、X-Service-Catalog ヘッダーを設定しません。
insecure = False	ブール値	HTTPS 接続を確認します。
interface = internal	文字列値	Identity API エンドポイントに使用するインターフェイス。有効な値は、"public"、"internal" (デフォルト)、または "admin" です。

設定オプション = デフォルト値	型	説明
keyfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
memcache_pool_conn_get_timeout = 10	整数値	(オプション) プールから memcached クライアント接続を取得するまで操作が待機する秒数。
memcache_pool_dead_retry = 300	整数値	(オプション)memcached サーバーが停止しているとみなされる秒数。この秒数が経過すると再試行されます。
memcache_pool_maxsize = 10	整数値	(オプション) すべての memcached サーバーへのオープン接続の最大合計数。
memcache_pool_socket_timeout = 3	整数値	(オプション)memcached サーバーと通信する際のソケットのタイムアウト (秒単位)。
memcache_pool_unused_timeout = 60	整数値	(オプション)memcached への接続がプール内で未使用の状態を維持する秒数。この秒数が経過すると終了されます。
memcache_secret_key = None	文字列値	(オプション、memcache_security_strategy が定義されている場合には必須) この文字列は鍵の導出に使用されます。
memcache_security_strategy = None	文字列値	(オプション) 定義されている場合は、トークンデータを認証、または認証して暗号化する必要があるかどうかを示します。MAC の場合、キャッシュでトークンデータが認証されます (HMAC を使用)。ENCRYPT の場合、キャッシュでトークンデータが暗号化され、認証されます。値がこれらのオプションのいずれでもない場合や空の場合には、auth_token は初期化時に例外を発生させます。
memcache_use_advanced_pool = False	ブール値	(オプション) 高度な (eventlet に対して安全な) memcached クライアントプールを使用します。高度なプールは python 2.x でのみ動作します。
memcached_servers = None	リスト値	オプションで、キャッシュに使用する memcached サーバーのリストを指定します。未定義のままの場合、トークンは代わりに処理中にキャッシュされます。
region_name = None	文字列値	アイデンティティーサーバーがあるリージョン。

設定オプション = デフォルト値	型	説明
service_token_roles = ['service']	リスト値	サービストークンに存在する必要があるロールの選択。サービストークンは、期限切れのトークンを使用できることを要求できるため、このチェックでは実際のサービスのみがこのトークンを送信するように厳密に制御する必要があります。ここでのロールは ANY チェックとして適用されるため、このリストのロールはすべて存在している必要があります。後方互換性の理由から、現在 allow_expired チェックにのみ影響します。
service_token_roles_required = False	ブール値	後方互換性の理由から、service_token_roles チェックを有効としてパスしない有効なサービストークンをパスさせる必要があります。これを true に設定することが今後のリリースでデフォルトとなり、可能な場合は有効にされる必要があります。
service_type = None	文字列値	サービスカタログに表示されるサービスの名前または種別。これは、制限されたアクセスルールがあるトークンを検証するために使用されます。
token_cache_time = 300	整数値	トークンの検証に過剰な時間を費やすのを防ぐために、ミドルウェアは、設定可能な期間 (秒単位) 中は以前に見たトークンをキャシュします。キャッシュを完全に無効にするには -1 に設定します。
www_authenticate_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。

2.1.14. nova

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[nova]** グループで使用できるオプションの概要をまとめたものです。

表2.13 nova

設定オプション = デフォルト値	型	説明
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
insecure = False	ブール値	HTTPS 接続を確認します。
interface = public	文字列値	使用する nova エンドポイントのタイプ。このエンドポイントは keystone カタログで検索され、パブリック、内部、または管理者のいずれかでなければなりません。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
region_name = None	文字列値	使用する nova リージョンの名前。keystone が複数のリージョンを管理する場合に便利です。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
timeout = None	整数値	http 要求のタイムアウト値
token_auth_url = None	文字列値	現在のユーザートークンを使用する場合の nova 接続の認証 URL

2.1.15. oslo_concurrency

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[oslo_concurrency]** グループで利用可能なオプションの概要をまとめたものです。

表2.14 oslo_concurrency

設定オプション = デフォルト値	型	説明
disable_process_locking = False	ブール値	プロセス間ロックを有効または無効にします。

設定オプション = デフォルト値	型	説明
lock_path = None	文字列値	ロックファイルに使用するディレクトリー。セキュリティー上、指定したディレクトリーは、ロックが必要なプロセスを実行しているユーザーのみが書き込み可能である必要があります。デフォルトは環境変数 OSLO_LOCK_PATH です。外部ロックを使用する場合は、ロックパスを設定する必要があります。

2.1.16. oslo_messaging_amqp

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[oslo_messaging_amqp]** グループで利用可能なオプションの概要をまとめたものです。

表2.15 oslo_messaging_amqp

設定オプション = デフォルト値	型	説明
addressing_mode = dynamic	文字列値	ドライバーが使用するアドレスモードを示します。許可される値:**legacy** - レガシーのルーティング不可能なアドレス指定を使用します。**routable** - ルーティング可能なアドレスを使用します。**dynamic** - メッセージバスがルーティングをサポートしていない場合はレガシーのアドレスを使用します。それ以外の場合は、ルーティング可能なアドレスを使用します。
anycast_address = anycast	文字列値	コンシューマーのグループに送信する際に、アドレス接頭辞に追加されます。コンシューマー間でラウンドロビン方式で配信される必要のあるメッセージを特定するために、メッセージバスで使用されます。
broadcast_prefix = broadcast	文字列値	すべてのサーバーにブロードキャストする場合に使用されるアドレス接頭辞
connection_retry_backoff = 2	整数値	フェイルオーバーの試みに失敗するたびに、connection_retry_interval をこの秒数だけ増やします。
connection_retry_interval = 1	整数値	再接続を試みる前に一時停止する秒数。
connection_retry_interval_max = 30	整数値	connection_retry_interval + connection_retry_backoff の上限
container_name = None	文字列値	AMQP コンテナーの名前。グローバルで一意でなければなりません。デフォルトは、生成された UUID です。

設定オプション = デフォルト値	型	説明
default_notification_exchange = None	文字列値	通知アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_notification_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **notify**
default_notify_timeout = 30	整数値	送信された通知メッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_reply_retry = 0	整数値	リカバリー可能なエラーが原因で失敗した返信メッセージを再送信する最大試行回数。
default_reply_timeout = 30	整数値	rpc 返信メッセージ配信の期限。
default_rpc_exchange = None	文字列値	RPC アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_rpc_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **rpc**
default_send_timeout = 30	整数値	rpc キャストまたは呼び出しメッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_sender_link_timeout = 600	整数値	アイドル状態の送信者リンクのパージをスケジュールする時間。期限切れ後にリンクの割り当てを解除します。
group_request_prefix = unicast	文字列値	グループの任意のサーバーに送信する際のアドレス接頭辞
idle_timeout = 0	整数値	非アクティブな接続のタイムアウト (秒単位)
link_retry_delay = 10	整数値	リカバリー可能なエラーが原因で失敗した AMQP 1.0 リンクの再接続間に一時停止する時間。
multicast_address = multicast	文字列値	ファンアウトメッセージを送信するときにアドレス接頭辞に追加されます。ファンアウトメッセージを識別するためにメッセージバスによって使用されます。
notify_address_prefix = openstack.org/om/notify	文字列値	すべての生成される通知アドレスのアドレス接頭辞
notify_server_credit = 100	整数値	受信通知メッセージのウィンドウサイズ

設定オプション = デフォルト値	型	説明
pre_settled = ['rpc-cast', 'rpc-reply']	多値	このタイプのメッセージを事前処置状態で送信します。事前処置されたメッセージは、ピアから確認応答を受信しません。注記: 事前処置されたメッセージは、配信に失敗した場合に警告なしに破棄される可能性があります。許可される値:**rpc-call** - RPC 呼び出しを事前処置状態で送信します。**rpc-reply**- RPC 返信を事前処置状態で送信します。**rpc-cast** - RPC キャストを事前処置状態で送信します。**notify** - 通知を事前処置状態で送信します。
pseudo_vhost = True	ブール値	仮想ホスト (qpidd など) をネイティブにサポートしていないメッセージバスの仮想ホストサポートを有効にします。true に設定すると、仮想ホスト名はすべてのメッセージバスアドレスに追加され、結果として仮想ホストごとにプライベート **サブネット** を作成します。メッセージバスが仮想ホストの名前として AMQP 1.0 オープン遂行動詞の **hostname** フィールドを使用して仮想ホストをサポートする場合は False に設定します。
reply_link_credit = 200	整数値	受信する RPC 返信メッセージのウィンドウサイズ。
rpc_address_prefix = openstack.org/om/rpc	文字列値	すべての生成される RPC アドレスのアドレス接頭辞
rpc_server_credit = 100	整数値	受信する RPC リクエストメッセージのウィンドウサイズ。
`sasl_config_dir = `	文字列値	SASL 設定が含まれるディレクトリーへのパス
`sasl_config_name = `	文字列値	設定ファイルの名前 (.conf 接尾辞なし)
`sasl_default_realm = `	文字列値	ユーザー名にレルムが存在しない場合に使用する SASL レルム
`sasl_mechanisms = `	文字列値	許可される SASL メカニズムのスペース区切りリスト
server_request_prefix = exclusive	文字列値	特定のサーバーに送信する時に使用するアドレス接頭辞
ssl = False	ブール値	SSL 経由で接続を試みます。その他の ssl 関連のパラメーターが指定されていない場合、システムの CA バンドルを使用してサーバーの証明書を検証します。

設定オプション = デフォルト値	型	説明
`ssl_ca_file = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_cert_file = `	文字列値	クライアント認証用の自己識別証明書 PEM ファイル
`ssl_key_file = `	文字列値	ssl_cert_file 証明書の署名に使用される秘密鍵 PEM ファイル (オプション)
ssl_key_password = None	文字列値	ssl_key_file を復号するためのパスワード (暗号化されている場合)
ssl_verify_vhost = False	ブール値	デフォルトでは、SSL は、サーバーの証明書の名前が transport_url のホスト名と一致することを確認します。設定によっては、代わりに仮想ホスト名を使用することが望ましい場合があります。たとえば、サーバーが Server Name Indication TLS 拡張 (rfc6066) を使用して、仮想ホストごとの証明書を提供する場合などです。サーバーの SSL 証明書が DNS 名ではなく仮想ホスト名を使用する場合は、ssl_verify_vhost を True に設定します。
trace = False	ブール値	デバッグ: AMQP フレームを標準出力 (stdout) にダンプします。
unicast_address = unicast	文字列値	特定の RPC/通知サーバーに送信する際に、アドレス接頭辞に追加されます。単一の送信先に送信されたメッセージを識別するためにメッセージバスによって使用されます。

2.1.17. oslo_messaging_kafka

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[oslo_messaging_kafka]** グループで使用できるオプションの概要をまとめたものです。

表2.16 oslo_messaging_kafka

設定オプション = デフォルト値	型	説明
compression_codec = none	文字列値	プロデューサーによって生成されたすべてのデータの圧縮コーデック。設定されていない場合、圧縮は使用されません。この設定に許可される値は、kafka バージョンに依存することに注意してください。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限

設定オプション = デフォルト値	型	説明
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
consumer_group = oslo_messaging_consumer	文字列値	Kafka コンシューマーのグループ ID。あるグループのコンシューマーは、メッセージ消費を調整します。
enable_auto_commit = False	ブール値	非同期コンシューマーコミットを有効にします。
kafka_consumer_timeout = 1.0	浮動小数点の値	Kafka コンシューマーのデフォルトタイムアウト
kafka_max_fetch_bytes = 1048576	整数値	Kafka コンシューマーの最大フェッチバイト
max_poll_records = 500	整数値	ポーリング呼び出しで返されるレコードの最大数
pool_size = 10	整数値	Kafka コンシューマーのプールサイズ
producer_batch_size = 16384	整数値	プロデューサーの非同期送信のバッチサイズ
producer_batch_timeout = 0.0	浮動小数点の値	KafkaProducer バッチ処理の遅延の上限 (秒単位)
sasl_mechanism = PLAIN	文字列値	セキュリティープロトコルが SASL である際のメカニズム
security_protocol = PLAINTEXT	文字列値	ブローカーとの通信に使用されるプロトコル
`ssl_cafile = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_client_cert_file = `	文字列値	認証に使用されるクライアント証明書の PEM ファイル。
`ssl_client_key_file = `	文字列値	認証に使用されるクライアントキーの PEM ファイル。
`ssl_client_key_password = `	文字列値	認証に使用されるクライアントキーパスワードファイル。

2.1.18. oslo_messaging_notifications

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[oslo_messaging_notifications]** グループで使用できるオプションの概要をまとめたものです。

表2.17 oslo_messaging_notifications

設定オプション = デフォルト値	型	説明
driver = []	多値	通知の送信を処理するドライバー。使用できる値は messaging、messagingv2、routing、log、test、noop です。
retry = -1	整数値	リカバリー可能なエラーが原因で配信に失敗した通知メッセージを再送信する最大試行回数。0 - 再試行なし、-1 - 無限回
topics = ['notifications']	リスト値	OpenStack の通知に使用する AMQP トピック。
transport_url = None	文字列値	通知に使用するメッセージングドライバーを表す URL。設定されていない場合は、RPC に使用されるものと同じ設定にフォールバックします。

2.1.19. oslo_messaging_rabbit

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[oslo_messaging_rabbit]** グループで使用できるオプションの概要をまとめたものです。

表2.18 oslo_messaging_rabbit

設定オプション = デフォルト値	型	説明
amqp_auto_delete = False	ブール値	AMQP の自動削除キュー。
amqp_durable_queues = False	ブール値	AMQP で永続キューを使用します。
direct_mandatory_flag = True	ブール値	(非推奨) ダイレクト送信のための RabbitMQ 必須フラグを有効/無効にします。直接送信は応答として使用されるため、クライアントキューが存在しない場合には MessageUndeliverable 例外が発生します。MessageUndeliverable 例外はタイムアウトをループして、送信者が回復する可能性が高くなります。このフラグは非推奨になり、この機能を非アクティブ化することはできません。
enable_cancel_on_failover = False	ブール値	x-cancel-on-ha-failover フラグを有効にして、rabbitmq サーバーがキューが停止しているときにコンシューマーをキャンセルし、通知できるようにします。

設定オプション = デフォルト値	型	説明
heartbeat_in_pthread = False	ブール値	デフォルトでは、ネイティブ Python スレッドを介してヘルスチェックハートビートスレッドを実行します。このオプションが False の場合、ヘルスチェックハートビートは親プロセスから実行モデルを継承します。たとえば、親プロセスが eventlet/greenlet を使用して stdlib にモンキーパッチを適用した場合、ハートビートはグリーンスレッドを介して実行されます。This option should be set to True only for the wsgi services.
heartbeat_rate = 2	整数値	heartbeat_timeout_threshold 中、ハートビートを確認する回数。
heartbeat_timeout_threshold = 60	整数値	ハートビートの keep-alive が失敗した場合に Rabbit ブローカーがダウンとみなされるまでの秒数 (0 はハートビートを無効にします)。
kombu_compression = None	文字列値	実験的用途: 許容値は gzip、bz2 です。設定されていない場合、圧縮は使用されません。このオプションは、今後のバージョンで利用できない可能性があります。
kombu_failover_strategy = round-robin	文字列値	現在接続しているノードが利用できなくなった場合に、次の RabbitMQ ノードを選択する方法を指定します。設定で複数の RabbitMQ ノードが指定される場合に限り有効になります。
kombu_missing_consumer_retry_timeout = 60	整数値	応答のないクライアントが応答を送信するのを待つ時間。この時間が経過すると無視します。この値は rpc_response_timeout より長くすることはできません。
kombu_reconnect_delay = 1.0	浮動小数点の値	AMQP コンシューマーの取り消し通知への応答で、再接続するまでの待機時間。
rabbit_ha_queues = False	ブール値	RabbitMQ (x-ha-policy: all) の HA キューの使用を試みます。このオプションを変更する場合は、RabbitMQ データベースを消去する必要があります。RabbitMQ 3.0 では、キューを宣言する際に x-ha-policy 引数によってキューのミラーリングが制御されなくなりました。すべてのキュー (自動生成された名前のキューを除く) がすべてのノードでミラーリングされるようにするには、"rabbitmqctl set_policy HA **^(?!amq\.).* {"ha-mode": "all"}**" を実行します。
rabbit_interval_max = 30	整数値	RabbitMQ 接続を再試行する最大間隔。デフォルトは 30 秒です。

設定オプション = デフォルト値	型	説明
rabbit_login_method = AMQPLAIN	文字列値	RabbitMQ ログイン方法
rabbit_qos_prefetch_count = 0	整数値	事前フェッチするメッセージの数を指定します。ゼロに設定すると、無制限のメッセージのフェッチが許可されます。
rabbit_retry_backoff = 2	整数値	RabbitMQ に接続する際に再試行間でバックオフする長さ
rabbit_retry_interval = 1	整数値	RabbitMQ との接続を再試行する頻度。
rabbit_transient_queues_ttl = 1800	整数値	キューの TTL (x-expires) の期間 (秒単位) を表す正の整数。TTL の期間使用されないキューは自動的に削除されます。このパラメーターは応答キューとファンアウトキューにのみ影響します。
ssl = False	ブール値	SSL 経由で接続します。
`ssl_ca_file = `	文字列値	SSL 認証局ファイル (SSL が有効な場合にのみ有効)。
`ssl_cert_file = `	文字列値	SSL 証明書ファイル (SSL が有効な場合にのみ有効)。
`ssl_key_file = `	文字列値	SSL キーファイル (SSL が有効な場合にのみ有効)。
`ssl_version = `	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

2.1.20. oslo_middleware

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[oslo_middleware]** グループで使用できるオプションの概要をまとめたものです。

表2.19 oslo_middleware

設定オプション = デフォルト値	型	説明
enable_proxy_headers_parsing = False	ブール値	アプリケーションがプロキシーの背後であるかどうか。これにより、ミドルウェアがヘッダーを解析すべきかどうかを決定します。

設定オプション = デフォルト値	型	説明
max_request_body_size = 114688	整数値	各リクエストの本文の最大サイズ (バイト単位)。
secure_proxy_ssl_header = X-Forwarded-Proto	文字列値	SSL 終端プロキシーによって非表示であった場合でも、元のリクエストプロトコルスキームを決定するために使用される HTTP ヘッダー。

2.1.21. oslo_policy

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[oslo_policy]** グループで使用できるオプションの概要をまとめたものです。

表2.20 oslo_policy

設定オプション = デフォルト値	型	説明
enforce_new_defaults = False	ブール値	このオプションは、ポリシーを評価するときに古い非推奨のデフォルトを使用するかどうかを制御します。**True** の場合、古い非推奨のデフォルトは評価されません。これは、既存のトークンが古いデフォルトで許可されているが、新しいデフォルトで許可されていない場合、許可されないことを意味します。新しいデフォルトと **scope_type** の利点を一緒に得ることができるように、このフラグを **enforce_scope** フラグと一緒に有効にすることを推奨します
enforce_scope = False	ブール値	このオプションは、ポリシーを評価する際にスコープを適用するかどうかを制御します。**True** の場合、リクエストで使用されるトークンのスコープは、適用されるポリシーの **scope_types** と比較されます。スコープが一致しない場合、**InvalidScope** 例外が発生します。**False** の場合、ポリシーが一致しないスコープで呼び出されていることを運用者に通知するメッセージが記録されます。
policy_default_rule = default	文字列値	デフォルトルール。要求されたルールが見つからない場合に適用されます。
policy_dirs = ['policy.d']	多値	ポリシー設定ファイルが保存されるディレクトリー。これは、config_dir オプションで定義される検索パス内の任意のディレクトリーへの相対パスか、絶対パスにすることができます。このディレクトリーの検索には、policy_file で定義されたファイルが存在している必要があります。存在しないディレクトリーまたは空のディレクトリーは無視されます。

設定オプション = デフォルト値	型	説明
policy_file = policy.yaml	文字列値	ロールを指定のサービスのパーミッションにマッピングするファイルの相対パスまたは絶対パス。相対パスは、このオプションを設定する設定ファイルに対する相対パスとして指定する必要があります。
remote_content_type = application/x-www-form-urlencoded	文字列値	REST ベースのポリシーチェックのデータを送受信するコンテンツ種別
remote_ssl_ca_crt_file = None	文字列値	REST ベースのポリシーチェック用の認証局証明書ファイルの絶対パス
remote_ssl_client_crt_file = None	文字列値	REST ベースのポリシーチェック用のクライアント証明書の絶対パス
remote_ssl_client_key_file = None	文字列値	REST ベースのポリシーチェック用のクライアント鍵ファイルの絶対パス
remote_ssl_verify_server_crt = False	ブール値	REST ベースのポリシーチェックのサーバーアイデンティティー検証

2.1.22. oslo_reports

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[oslo_reports]** グループで使用できるオプションの概要をまとめたものです。

表2.21 oslo_reports

設定オプション = デフォルト値	型	説明
file_event_handler = None	文字列値	シグナルではなくレポートをトリガーする変更を監視するファイルへのパス。このオプションを設定すると、レポートのシグナルトリガーが無効になります。アプリケーションが WSGI アプリケーションとして実行されている場合は、シグナルの代わりにこれを使用することが推奨されます。
file_event_handler_interval = 1	整数値	file_event_handler が設定されている場合にポーリングの間隔 (秒単位)
log_dir = None	文字列値	ファイルを作成するログディレクトリーへのパス

2.1.23. oslo_versionedobjects

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[oslo_versionedobjects]** グループで使用できるオプションの概要をまとめたものです。

表2.22 oslo_versionedobjects

設定オプション = デフォルト値	型	説明
fatal_exception_format_errors = False	ブール値	例外メッセージ形式のエラーを致命的にする

2.1.24. privsep

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[privsep]** グループ下で使用できるオプションの概要をまとめたものです。

表2.23 privsep

設定オプション = デフォルト値	型	説明
capabilities = []	リスト値	privsep デーモンによって保持される Linux 機能のリスト。
group = None	文字列値	privsep デーモンを実行する必要のあるグループです。
helper_command = None	文字列値	fork メソッドを使用していない場合に privsep デーモンを起動するコマンドを呼び出します。指定のない場合は、現在の設定を再作成するために設計された sudo privsep-helper および引数を使用してデフォルト値が生成されます。このコマンドは、適切な --privsep_context および --privsep_sock_path 引数を使用する必要があります。
logger_name = oslo_privsep.daemon	文字列値	Logger name to use for this privsep context.By default all contexts log with oslo_privsep.daemon.
thread_pool_size = <based on operating system>	整数値	privsep が同時にプロセスを実行できるスレッドの数。デフォルトは、システム内の CPU コア数に設定されます。
user = None	文字列値	privsep デーモンを実行する必要のあるユーザー。

2.1.25. profiler

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[profiler]** グループ下で使用できるオプションの概要をまとめたものです。

表2.24 profiler

設定オプション = デフォルト値	型	説明
connection_string = messaging://	文字列値	通知バックエンドの接続文字列。 デフォルト値は **messaging://** で、通知機能を oslo_messaging に設定します。 使用できる値の例: ● **messaging://**: スパンの送信に oslo_messaging ドライバーを使用します。 ● **redis://127.0.0.1:6379**: スパンの送信に redis ドライバーを使用します。 ● **mongodb://127.0.0.1:27017**: スパンの送信に mongodb ドライバーを使用します。 ● **elasticsearch://127.0.0.1:9200:** elasticsearch ドライバーを使用してスパンを送信します。 ● **jaeger://127.0.0.1:6831**: スパンの送信に jaeger トレースを使用します。
enabled = False	ブール値	このノード上の全サービスのプロファイリングを有効にします。 デフォルト値は False です (プロファイリング機能を無効にします)。 値: ● True: 機能を有効にします ● False: 機能を無効にします。このプロジェクトの操作ではプロファイリングを開始できません。プロファイリングが別のプロジェクトでトリガーされた場合には、このプロジェクトの部分は空になります。
es_doc_type = notification	文字列値	elasticsearch の通知インデックスについてのドキュメントタイプ。
es_scroll_size = 10000	整数値	elasticsearch は大規模な要求をバッチに分割します。このパラメーターは、各バッチの最大サイズを定義します (例: es_scroll_size=10000)。
es_scroll_time = 2m	文字列値	このパラメーターは時間値パラメーター (es_scroll_time=2m など) です。これは、検索に参加するノードが継続してサポートするために関連するリソースを維持する時間を示します。

設定オプション = デフォルト値	型	説明
filter_error_trace = False	ブール値	エラー/例外を含むフィルタートレースを分離された場所へ有効にします。 デフォルト値は False に設定されます。 値: ● True: エラー/例外を含むフィルタートレースを有効にします。 ● False: フィルターを無効にします。
hmac_keys = SECRET_KEY	文字列値	パフォーマンスプロファイリング用にコンテキストデータの暗号化に使用するシークレットキー。 この文字列値は <key1>[,<key2>,...<keyn>] の形式にする必要があります。各キーはランダムな文字列です。REST API 経由でプロファイリングをトリガーするユーザーは、REST API 呼び出しのヘッダーにこれらのキーの 1 つを設定し、この特定プロジェクトのノードのプロファイリング結果が含まれるようにする必要があります。 プロファイリングを有効にするには、enabled フラグと hmac_keys 設定オプションの両方を設定する必要があります。また、少なくとも 1 つのキーで全サービス間で正しいプロファイリング情報を生成するには、OpenStack プロジェクト間で一貫性を確保する必要があります。これにより、クライアント側からを使用して、すべての可能なリソースからの情報を含むトレースを生成できます。
sentinel_service_name = mymaster	文字列値	Redissentinel はサービス名を使用してマスターの redis サービスを識別します。このパラメーターは名前を定義します (例: **sentinal_service_name=mymaster**)。
socket_timeout = 0.1	浮動小数点の値	Redissentinel は、接続に timeout オプションを提供します。このパラメーターは、タイムアウトを定義します (例: socket_timeout=0.1)。

設定オプション = デフォルト値	型	説明
trace_sqlalchemy = False	ブール値	サービスで SQL 要求のプロファイリングを有効にします。 デフォルト値は False です (SQL 要求はトレースされません)。 値: ● True: SQL 要求のプロファイリングを有効にします。各 SQL クエリーはトレースの一部となり、それにどの程度の時間を費やしたかによって分析できます。 ● False: SQL 要求のプロファイリングを無効にします。費やした時間は、高いレベルの操作でのみ表示されます。単一の SQL クエリーをこのように分析できません。

2.1.26. sample_castellan_source

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[sample_castellan_source]** グループで使用できるオプションの概要をまとめたものです。

表2.25 sample_castellan_source

設定オプション = デフォルト値	型	説明
config_file = None	文字列値	castellan 設定ファイルへのパス。
driver = None	文字列値	この設定ソースを読み込むことができるドライバーの名前。
mapping_file = None	文字列値	configuration/castellan_id マッピングファイルへのパス。

2.1.27. sample_remote_file_source

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[sample_remote_file_source]** グループで使用できるオプションの概要をまとめたものです。

表2.26 sample_remote_file_source

設定オプション = デフォルト値	型	説明
ca_path = None	文字列値	CA_BUNDLE ファイルまたは信頼済み CA の証明書のあるディレクトリーへのパス。

設定オプション = デフォルト値	型	説明
client_cert = None	文字列値	クライアント側の証明書 (証明書のみまたは秘密鍵と証明書を含む単一のファイルパス)
client_key = None	文字列値	クライアント側の秘密鍵 (client_cert が指定されていても、秘密鍵は含まれません)。
driver = None	文字列値	この設定ソースを読み込むことができるドライバーの名前。
uri = None	URI 値	追加の設定ファイルの場所の URI で必須オプションです。

2.1.28. service_user

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[service_user]** グループで使用できるオプションの概要をまとめたものです。

表2.27 service_user

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID

設定オプション = デフォルト値	型	説明
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
send_service_user_token = False	ブール値	True の場合には、REST API にユーザートークンを送信すると、サービストークンも送信されます。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
system-scope = None	文字列値	システム操作のスコープ
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	User ID
username = None	文字列値	ユーザー名

2.1.29. ssl

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[ssl]** グループ下で使用できるオプションの概要をまとめたものです。

表2.28 ssl

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	接続クライアントの検証に使用する CA 証明書ファイル。
cert_file = None	文字列値	サーバーをセキュアに起動する際に使用する証明書ファイル。

設定オプション = デフォルト値	型	説明
ciphers = None	文字列値	利用可能な暗号のリストを設定します。値は OpenSSL 暗号リスト形式の文字列である必要があります。
key_file = None	文字列値	サーバーをセキュアに起動する際に使用する秘密鍵ファイル。
version = None	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

2.1.30. vault

以下の表は、**/etc/cinder/cinder.conf** ファイルの **[vault]** グループで利用可能なオプションの概要をまとめたものです。

表2.29 vault

設定オプション = デフォルト値	型	説明
approle_role_id = None	文字列値	Vault での認証の場合は Approle role_id
approle_secret_id = None	文字列値	Vault での認証の場合は Approle secret_id
kv_mountpoint = secret	文字列値	使用する Vault の KV ストアのマウントポイント (例: secret)。
kv_version = 2	整数値	使用する Vault 内の KV ストアのバージョン。例: 2
root_token_id = None	文字列値	Vault の root トークン
ssl_ca_crt_file = None	文字列値	証明書ファイルの絶対パス
use_ssl = False	ブール値	SSL の有効化/無効化
vault_url = http://127.0.0.1:8200	文字列値	このエンドポイントを使用して Vault に接続します (例: "http://127.0.0.1:8200")。

第3章 GLANCE

以下の章では、**glance** サービスの設定オプションについて説明します。

3.1. GLANCE-API.CONF

このセクションには、**/etc/glance/glance-api.conf** ファイルのオプションが含まれます。

3.1.1. DEFAULT

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[DEFAULT]** グループで利用可能なオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
allow_additional_image_properties = True	ブール値	ユーザーがイメージに追加/カスタムの属性を追加できるようにします。 Glance は、すべてのイメージに表示される属性の標準セット () スキーマ) を定義します。これらのプロパティーは、**base properties** とも呼ばれます。Glance では、これらの属性に加え、カスタムプロパティーをイメージに追加できます。これらは、**additional properties** と呼ばれています。 デフォルトでは、この設定オプションは **True** に設定され、ユーザーは追加のプロパティーを追加できます。イメージに追加できる追加プロパティーの数は、**image_property_quota** 設定オプションで制御できます。 値: ● True ● False 関連するオプション: ● image_property_quota **非推奨:** Ussuri **理由:** このオプションは冗長です。**image_property_quota** 設定オプションを使用して、カスタムイメージプロパティーの使用を制御します。このオプションは、Victoria の開発サイクル中に削除される予定です。

設定オプション = デフォルト値	型	説明
allow_anonymous_access = False	ブール値	認証されていないユーザーへの限定的なアクセスを許可します。 ブール値を割り当て、認証されていないユーザーの API アクセスを判別します。False に設定すると、認証されていないユーザーが API にアクセスできなくなります。True に設定すると、認証されていないユーザーは読み取り専用権限で API にアクセスできます。ただし、これはコンテキストミドルウェアを使用する場合にのみ適用されます。 値: ● True ● False 関連するオプション: ● なし
api_limit_max = 1000	整数値	要求によって返されることのできる最大結果数。 **limit_param_default** のヘルプテキストで説明されているように、リクエストによっては複数の結果が返される場合があります。返される結果の数は、要求の **limit** パラメーターまたは **limit_param_default** 設定オプションで管理されます。いずれの場合もこの値が、この設定オプションで定義される絶対最大値よりも大きくすることはできません。この値より大きいすべてのものは、ここで定義される最大値にトリミングされます。 **注記** これを非常に大きな値に設定すると、データベースのクエリーが遅くなり、応答時間が長くなる可能性があります。これを非常に低い値に設定すると、ユーザーエクスペリエンスが低下してしまう可能性があります。 値: ● 任意の正の整数 関連するオプション: ● limit_param_default

設定オプション = デフォルト値	型	説明
backlog = 4096	整数値	受信接続要求の数を設定します。 正の整数値を指定して、バックログキューの要求数を制限します。デフォルトのキューサイズは 4096 です。 TCP リスナーソケットへの受信接続は、サーバーとの接続が確立される前にキューイングされます。TCP ソケットのバックログを設定すると、受信トラフィックのキューサイズが限定されます。 値: ● 正の整数 関連するオプション: ● なし
bind_host = 0.0.0.0	ホストのアドレス値	glance サーバーをバインドするための IP アドレス。 glance サーバーをバインドするための IP アドレスを提供します。デフォルト値は **0.0.0.0** です。 このオプションを編集して、サーバーがネットワークカード上の特定の IP アドレスでリッスンできるようにします。これにより、サーバー用の特定のネットワークインターフェイスを選択しやすくなります。 値: ● 有効な IPv4 アドレス ● 有効な IPv6 アドレス 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
bind_port = None	ポート値	サーバーがリッスンするポート番号。 サーバーのソケットをバインドする有効なポート番号を指定します。次に、このポートはプロセスを特定し、サーバーに到達するネットワークメッセージを転送するように設定されます。API サーバーのデフォルトの bind_port 値は 9292 で、レジストリーサーバーの場合は 9191 です。 値: ● 有効なポート番号 (0 から 65535) 関連するオプション: ● なし
client_socket_timeout = 900	整数値	クライアント接続のソケット操作のタイムアウト。 入力された接続を閉じるまでの待ち時間を設定するために、時間を表す有効な整数値を秒単位で入力してください。デフォルト値は 900 秒です。 値 zero は永久に待機します。 値: ● ゼロ ● 正の整数 関連するオプション: ● なし
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
control_exchange = openstack	文字列値	トピックがスコープ設定されるデフォルトの変換。transport_url オプションで指定した変換名で上書きできます。
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
default_publisher_id = image.localhost	文字列値	送信 Glance 通知用のデフォルトの publisher_id。 これは、通知ドライバーが Glance サービスから発信されるイベントのメッセージを識別するために使用する値です。通常、これはメッセージを生成したインスタンスのホスト名です。 値: ● 任意の合理的なインスタンス識別子。例: image.host1 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
delayed_delete = False	ブール値	削除をオン/オフにします。 通常、イメージが削除されると、**glance-api** サービスはイメージを **delete** 状態にし、同時にそのデータを削除します。遅延削除とは、Glance の機能のひとつで、イメージデータの実際の削除を (設定オプション **scrub_time** で決定される) 後の時点まで遅らせるものです。遅延削除が有効になると、**glance-api** サービスは削除時にイメージを **pending_delete** の状態に入れ、イメージが scrubber のストレージバックエンドに残され、後で削除されるようにします。イメージスクラバは、イメージデータの削除に成功すると、イメージを **deleted** 状態に移行させます。 **注記** 遅延削除をオンにすると、バックエンドのストレージが不要な使用でいっぱいになるのを防ぐために、イメージスクラバを定期的なタスクとして実行しなければなりません。 値: ● True ● False 関連するオプション: ● **scrub_time** ● **wakeup_time** ● **scrub_pool_size**

設定オプション = デフォルト値	型	説明
digest_algorithm = sha256	文字列値	デジタル署名に使用するダイジェストアルゴリズム。 デジタル署名の生成に使用するダイジェストアルゴリズムを表す文字列値を指定します。デフォルトでは **sha256** が使用されます。 プラットフォームの OpenSSL バージョンでサポートされる利用可能なアルゴリズムのリストを取得するには、コマンド **openssl list-message-digest-algorithms** を実行します。たとえば、**sha1**、**sha256**、および **sha512** です。 **注記** **digest_algorithm** は Glance のイメージの署名および検証に関連しません。これは、証明書ファイルとキーファイル検証の一部として汎用一意識別子 (UUID) に署名するために使用されます。 値: - OpenSSL メッセージダイジェストアルゴリズム ID 関係オプション: - なし

設定オプション = デフォルト値	型	説明
disabled_notifications = []	リスト値	無効にする通知のリスト。 出力すべきでない通知のリストを指定します。通知タイプとして、単一のイベント通知を無効にする通知タイプを指定するか、グループ内のすべてのイベント通知を無効にする通知グループ接頭辞を指定できます。 可能な値: 無効にする個々の通知タイプまたは通知グループのコンマ区切りリスト。現在サポートされているグループ: ● image ● image.member ● task ● metadef_namespace ● metadef_object ● metadef_property ● metadef_resource_type ● metadef_tag > For a complete listing and description of each event refer to: http://docs.openstack.org/developer/glance/notifications.html > The values must be specified as: <group_name>.<event_name> For example: image.create,task.success,metadef_tag 関連するオプション: ● なし
enabled_backends = None	dict 値	ストア識別子とストアタイプの Key:Value ペア。複数のバックエンドを指定する場合は、コンマで区切る必要があります。

設定オプション = デフォルト値	型	説明
enabled_import_methods = ['glance-direct', 'web-download', 'copy-image']	リスト値	有効なイメージのインポートメソッドの一覧 'glance-direct', 'copy-image' and 'web-download' are enabled by default. Related options: ** [DEFAULT]/node_staging_uri
enforce_secure_rbac = False	ブール値	OpenStack 全体で使用される一般的なペルソナ定義に基づいて API アクセスを強制します。このオプションを有効にすると、**member** ロールの背後で、プライベートイメージの作成や共有イメージのステータスの更新など、プロジェクト固有の読み取り/書き込み操作が形式化されます。また、**reader** ロールの背後で、プロジェクト内のプライベートイメージをリスト表示するなど、プロジェクト固有の API 操作に役立つ読み取り専用バリアントを形式化します。 オペレーターはこの機会に、glance の新しいイメージポリシーを理解し、デプロイメントでの割り当てを監査し、keystone のデフォルトロール (**admin**、**member**、**reader** など) を使用して権限を更新する必要があります。 関連するオプション: ● [oslo_policy]/enforce_new_defaults **非推奨:** Wallaby **理由:** このオプションは、一般的な RBAC ペルソナに基づいて承認を実施することを選択することをオペレーターに要求するために導入されました。これは Wallaby リリースの時点で実験的です。この動作はデフォルトであり、将来のリリースでは STABLE になり、このオプションを削除できるようになります。
executor_thread_pool_size = 64	整数値	エグゼキューターがスレッディングまたはイベントレットの場合のエグゼキュータースレッドプールのサイズ。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。

設定オプション = デフォルト値	型	説明
hashing_algorithm = sha512	文字列値	**os_hash_value** プロパティーの計算に使用される安全なハッシュアルゴリズム。 このオプションは、**os_hash_algo** と **os_hash_value** の 2 つのイメージ属性で設定される Glance の multihash を設定します。**os_hash_algo** はこの設定オプションの値によって入力され、**os_hash_value** は、アルゴリズムがアップロードまたはインポートされたイメージデータに適用されると計算された hexdigest によって入力されます。 この値は、python **hashlib** ライブラリーが認識する有効なセキュアなハッシュアルゴリズム名である必要があります。Glance インストールで使用されているライブラリーのバージョンの **hashlib.algorithms_available** データメンバーを調べることで、これらを確認することができます。ただし、相互運用性のために、**hashlib.algorithms_guaranteed** データメンバーが提供する安全なハッシュ名のセットを使用することが推奨されます。これは、すべてのプラットフォームの **hashlib** ライブラリーでサポートされることが保証されているためです。したがって、**hashlib** を使用するイメージコンシューマーは、イメージの **os_hash_value** を検証できるはずです。 **sha512** のデフォルト値は、高性能なハッシュアルゴリズムです。 このオプションの設定が間違っていると、イメージデータの格納試行は失敗します。このため、デフォルト値を使用することが推奨されます。 値: ● Python **hashlib** ライブラリーが認識するセキュアなハッシュアルゴリズム名 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
http_keepalive = True	ブール値	TCP 上の HTTP に keep alive オプションを設定します。 keep alive パケットの送信を決定するブール値を指定します。**False** に設定すると、サーバーはヘッダー Connection: close を返します。**True** に設定すると、サーバーは応答に "Connection: Keep-Alive" を返します。これにより、新しいリクエストごとに新しいリクエストを開かなくても、HTTP 会話に同じ TCP 接続が保持されるようになります。 応答の受信およびクライアントによって正常に読み取られた後にクライアントソケット接続を明示的に閉じる必要がある場合は、このオプションを **False** に設定する必要があります。 値: ● True ● False 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
image_cache_dir = None	文字列値	イメージキャッシュのベースディレクトリー。 これは、イメージデータがキャッシュされ、提供される場所です。キャッシュされたイメージはすべて、このディレクトリーに直接保存されます。このディレクトリーには、**incomplete**、**invalid**、**queue** という 3 つのサブディレクトリーがあります。 **incomplete** サブディレクトリーは、イメージをダウンロードするためのステージングエリアです。イメージは最初にこのディレクトリーにダウンロードされます。イメージのダウンロードに成功すると、ベースディレクトリーに移動します。ただし、ダウンロードに失敗すると、部分的にダウンロードしたイメージファイルは **invalid** サブディレクトリーに移動します。 **queue`subdirectory は、ダウンロード用のイメージをキューに入れるために使用されます。これは主に cache-prefetcher によって使用されます。これは、使用前にイメージをキャッシュするために cache-pruner や cache-cleaner などの定期的なタスクとしてスケジュールできます。イメージのキャッシュを要求すると、Glance は `queue** ディレクトリー内のファイルに、イメージ ID をファイル名として関連付けます。cache-prefetcher の実行時に、実行時に **queue** ディレクトリーのファイルのポーリングを実行し、ファイルの作成順にダウンロードを開始します。ダウンロードに成功すると、ゼロサイズのファイルが **queue** ディレクトリーから削除されます。ダウンロードに失敗すると、ゼロサイズのファイルが残り、次に cache-prefetcher が実行されるときに再試行されます。 値: ● 有効なパス 関連するオプション: ● **image_cache_sqlite_db**

設定オプション = デフォルト値	型	説明
image_cache_driver = sqlite	文字列値	イメージキャッシュ管理に使用するドライバー。 このオプションを使用すると、利用可能な異なる image-cache ドライバーのいずれかを選択できます。image-cache ドライバーは、キャッシュからのイメージへの書き込み、キャッシュされたイメージの経過時間と使用状況などのイメージキャッシュの基本機能、キャッシュされたイメージのフェッチサイズ、キャッシュのサイズのフェッチ、キャッシュのキャッシュおよびクリーンアップ用のキューイメージの提供を行います。 ドライバーの基本機能は、基本クラス **glance.image_cache.drivers.base.Driver** で定義されています。すべての image-cache ドライバー (既存および見込み) はこのインターフェイスを実装する必要があります。現在利用可能なドライバーは **sqlite** および **xattr** です。これらのドライバーは、主にキャッシュされたイメージに関する情報を保存する方法によって異なります。 ● **sqlite** ドライバーは、キャッシュされたイメージの使用状況を追跡するために、sqlite データベース (すべての gloss ノードにローカルに置かれている) を使用します。 ● **xattr** ドライバーは、ファイルの拡張属性を使用してこの情報を保存します。また、アクセス時にファイルに **atime** を設定するファイルシステムも必要です。 値: ● sqlite ● xattr 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
image_cache_max_size = 10737418240	整数値	キャッシュサイズの上限 (バイト単位)。これを超えると、cache-pruner がイメージのキャッシュをクリーンアップします。 **注記** これは、実行する cache-pruner のしきい値です。これは、イメージキャッシュが拡張されることのないハード制限ではありません。実際、キャッシュプルーナーの実行頻度やキャッシュが満杯になる頻度に応じて、イメージキャッシュはここで非常に簡単に指定されるサイズにさらすことができます。そのため、cache-pruner を適切にスケジュールし、この制限を設定できるように注意して行ってください。 Glance はダウンロード時にイメージをキャッシュします。そのため、ダウンロード数が増えるにつれ、イメージキャッシュのサイズが長くなります。キャッシュサイズが管理できない状態にならないようにするには、cache-pruner を定期的なタスクとして実行することが推奨されます。キャッシュプルーナーが開始すると、現在のイメージキャッシュサイズを比較し、イメージキャッシュがここで指定されるサイズを超えて拡張した場合にクリーンアップがトリガーされます。クリーンアップ後、キャッシュのサイズはここで指定するサイズ以下になります。 値: ● 負の値以外の整数 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
image_cache_sqlite_db = cache.db	文字列値	イメージキャッシュ管理に使用される sqlite ファイルデータベースへの相対パス。 これは、イメージキャッシュの経過時間と使用状況の統計を追跡する sqlite ファイルデータベースの相対パスです。パスは、設定オプション **image_cache_dir** によって指定されるイメージキャッシュベースディレクトリーに相対的です。 これは、テーブルが 1 つしかない軽量データベースです。 値: ● sqlite ファイルデータベースへの有効な相対パス 関連するオプション: ● **image_cache_dir**
image_cache_stall_time = 86400	整数値	不完全なイメージがキャッシュに残っている時間 (秒) です。 不完全なイメージとは、ダウンロードが進行中であるイメージです。詳細は、設定オプション **image_cache_dir** の説明を参照してください。場合によっては、さまざまな理由でダウンロードがハングし、完全にダウンロードされたイメージは **incomplete** なディレクトリーに残ります。この設定オプションでは、不完全なイメージが **incomplete** ディレクトリーに残っている期間を、クリーンアップするまでの時間として設定します。不完全なイメージがここで指定するよりも時間がかかったら、次の実行時に cache-cleaner により削除されます。 不完全なイメージがディスク領域を占有しないように、Glance API ノードで定期的なタスクとして cache-cleaner を実行することが推奨されます。 値: ● 負の値以外の整数 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
image_location_quota = 10	整数値	イメージで許可される場所の最大数。 負の値は無制限として解釈されます。 関連するオプション: ● なし
image_member_quota = 128	整数値	1イメージあたりのイメージメンバーの最大数。 これにより、イメージが共有できるユーザーの最大数が制限されます。負の値は無制限として解釈されます。 関連するオプション: ● なし
image_property_quota = 128	整数値	イメージで許可されるプロパティーの最大数。 これにより、イメージに割り当てることができる追加のプロパティーの数の上限が適用されます。負の値は無制限として解釈されます。 **注記** 追加のプロパティーが無効の場合、影響はありません。**allow_additional_image_properties** を参照してください。 関連するオプション: ● **allow_additional_image_properties**

設定オプション = デフォルト値	型	説明
image_size_cap = 1099511627776	整数値	ユーザーがアップロードできるイメージの最大サイズ (バイト単位)。 上記のサイズより大きいイメージをアップロードすると、イメージの作成に失敗します。このオプションはデフォルトで 1099511627776 バイト (1 TiB) に設定されます。 注記: ● この値は、慎重に検討した後にのみ増やす必要があり、8 EiB (9223372036854775808) 以下を設定する必要があります。 ● この値は、バックエンドストレージ容量を慎重に検討して設定する必要があります。この値を非常に小さい値に設定すると、イメージが多数失敗する可能性があります。また、この値を非常に大きな値に設定すると、ストレージの使用が速くなる可能性があります。したがって、これは、利用可能なイメージおよび利用可能なストレージ容量の性質に応じて設定される必要があります。 値: ● 9223372036854775808 未満の正数
image_tag_quota = 128	整数値	イメージで許可されるタグの最大数。 負の値は無制限として解釈されます。 関連するオプション: ● なし
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。

設定オプション = デフォルト値	型	説明
limit_param_default = 25	整数値	リクエストに対して返すデフォルトの結果数。 リストイメージなどの特定の API リクエストへの応答により、複数のアイテムが返される場合があります。返される結果の数は、API 要求で **limit** パラメーターを指定して明示的に制御できます。ただし、**limit** パラメーターが指定されていない場合、この設定値は API 要求に対して返される結果のデフォルト数として使用されます。 注記: ● このオプションの値は、**api_limit_max** で指定された値よりも大きくすることはできません。 ● これを非常に大きな値に設定すると、データベースのクエリーが遅くなり、応答時間が長くなる可能性があります。これを非常に低い値に設定すると、ユーザーエクスペリエンスが低下してしまう可能性があります。 値: ● 任意の正の整数 関連するオプション: ● api_limit_max

設定オプション = デフォルト値	型	説明
location_strategy = location_order	文字列値	イメージの場所の優先順位を決定するストラテジー。 この設定オプションは、イメージのデータを提供するためにイメージの場所にアクセスする順序を決定するストラテジーを示します。Glance は次に、応答する最初のアクティブな場所からイメージデータを取得します。 このオプションには、**location_order** と **store_type** の 2 つの値を使用できます。デフォルト値は **location_order** です。これは、Glance に保存される順序でイメージデータを提供する場所により提供されることを示しています。**store_type** の値は、ストレージバックエンドが設定オプション **store_type_preference** のコンマ区切りリストとして記載される順序に基づいてイメージの場所の設定を設定します。 値: - location_order - store_type 関連するオプション: - store_type_preference
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
max_header_line = 16384	整数値	メッセージヘッダーの最大行サイズ。 メッセージヘッダーのサイズを制限するために長さを表す整数値を指定します。デフォルト値は 16384 です。 **注記** 大規模なトークンを使用する際に **max_header_line** を増やす必要がある場合があります (通常は大規模なサービスカタログを持つ Keystone v3 API によって生成されるトークン)。ただし、**max_header_line** の大きな値がログにあふれることに注意してください。 **max_header_line** を 0 に設定すると、メッセージヘッダーの行サイズの制限は設定されません。 値: ● 0 ● 正の整数 関連するオプション: ● なし
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。

設定オプション = デフォルト値	型	説明
max_request_id_length = 64	整数値	リクエスト ID の長さを制限します。 整数値を指定して、リクエスト ID の長さを指定された長さに制限します。デフォルト値は 64 です。これを 0 から 16384 までの任意の整数値に変更できますが、大きな値に設定した場合、ログが一杯になる可能性があることに注意してください。 値: ● 0 から 16384 までの整数値 関連するオプション: ● なし
metadata_encryption_key = None	文字列値	ストアの場所のメタデータを暗号化する際の AES 鍵。 Glance ストアメタデータの暗号化に使用する AES 暗号化を表す文字列値を指定します。 **注記** 使用する AES 鍵は、長さ 16、24、または 32 バイトのランダムな文字列に設定する必要があります。 値: ● 有効な AES 鍵を表す文字列値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
node_staging_uri = file:///tmp/staging/	文字列値	URL は一時データが保存される場所を指定します。 このオプションは、Glance の内部使用専用です。Glance は、イメージのインポートプロセス中、ユーザーがアップロードしたイメージデータを **ステージング** エンドポイントに保存します。 このオプションは、**ステージング** API エンドポイントを一切変更しません。 **注記** [task]/work_dir と同じパスを使用することは推奨されません。 **注記** file://<absolute-directory-path> is the only option api_image_import flow will support for now. **注記** ステージングパスは、すべての Glance API ノードからアクセス可能な共有ファイルシステム上になければなりません。 値: ● **file://** で始まり、ファイルシステムの絶対パスが続く文字列 関連するオプション: ● [task]/work_dir

設定オプション = デフォルト値	型	説明
property_protection_file = None	文字列値	属性保護ファイルの場所。 属性保護のルールおよびそれらに関連付けられたロール/ポリシーが含まれる属性保護ファイルへの有効なパスを指定します。 属性保護ファイルを設定すると、ロールまたはポリシーのいずれかによって識別される特定のユーザーセットによって作成、読み取り、更新、または削除される Glance イメージ属性が制限されます。この設定オプションが設定されていない場合、デフォルトでは属性保護は適用されません。値が指定されていてファイルが見つからない場合、glance-api サービスは起動に失敗します。属性保護の詳細は、https://docs.openstack.org/glance/latest/admin/property-protections.htmlを参照してください。 値: ● 空の文字列 ● 属性保護設定ファイルへの有効なパス 関連するオプション: ● property_protection_rule_format

設定オプション = デフォルト値	型	説明
property_protection_rule_format = roles	文字列値	プロパティー保護のルール形式。 Glance イメージ属性の属性保護を設定するのに必要な方法を指定します。許容される値は、**roles** と **policies** の 2 つです。デフォルト値は**roles** です。 値が **roles** の場合、属性保護ファイルには、保護されるそれぞれの属性に対する各 CRUD 操作のパーミッションを示す、ユーザーロールのコンマ区切りリストが含まれている必要があります。**policies** に設定すると、policy.yaml で定義されたポリシーを使用して、各 CRUD 操作のプロパティー保護が表現されます。**roles** または **policies** に基づいて属性保護を適用する方法の例は、https://docs.openstack.org/glance/latest/admin/property-protections.html#examplesを参照してください。 値: ● roles ● policies 関連するオプション: ● property_protection_file
public_endpoint = None	文字列値	Glance バージョンの応答に使用するパブリック URL エンドポイント。 Glance の versions 応答に表示されるパブリック URL エンドポイントです。値が指定されていない場合、バージョンの応答に表示されるエンドポイントは、API サービスを実行しているホストのエンドポイントです。API サービスがプロキシーの背後で実行されている場合は、プロキシー URL を表すようにエンドポイントを変更します。サービスがロードバランサーの背後で実行されている場合は、この値にロードバランサーの URL を追加します。 値: ● なし ● プロキシー URL ● ロードバランサー URL 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
pydev_worker_debug_host = None	ホストのアドレス値	pydev サーバーのホストアドレス。 デバッグに使用する pydev サーバーのホスト名または IP を表す文字列値を指定します。pydev サーバーはこのアドレスでデバッグ接続をリッスンし、Glance でのリモートデバッグを容易にします。 値: ● 有効なホスト名 ● 有効な IP アドレス 関連するオプション: ● なし
pydev_worker_debug_port = 5678	ポート値	pydev サーバーがリッスンするポート番号。 pydev サーバーをバインドするポート番号を指定します。pydev プロセスはこのポートでデバッグ接続を受け入れ、Glance でのリモートデバッグを容易にします。 値: ● 有効なポート番号 関連するオプション: ● なし
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
rpc_conn_pool_size = 30	整数値	RPC 接続プールのサイズ。

設定オプション = デフォルト値	型	説明
rpc_ping_enabled = False	ブール値	ping 呼び出しに応答するエンドポイントを追加します。エンドポイントの名前は oslo_rpc_server_ping です。
rpc_response_timeout = 60	整数値	呼び出しからの応答を待つ秒数。
scrub_pool_size = 1	整数値	イメージのスクラビングに使用されるスレッドプールのサイズ。 スクラブするイメージが多数ある場合には、スクラブキューが制御された状態に維持され、バックエンドストレージがタイムリーに要求されるように、イメージを並行してスクラブすると便利です。この設定オプションは、並行してスクラブされるイメージの最大数を示します。デフォルト値は 1 で、シリアルスクラビングを意味します。1 を超える値はすべて、並列スクラビングを意味します。 値: ● ゼロ以外の正の整数 関連するオプション: ● **delayed_delete**

設定オプション = デフォルト値	型	説明
scrub_time = 0	整数値	イメージのスクラビングを遅延させる時間 (秒単位)。 遅延削除がオンの場合、スクラバーがそのイメージデータを削除するまで、イメージは削除時に **pending_delete** 状態になります。通常、イメージが **pending_delete** 状態になると、すぐにスクラビングできます。ただし、この設定オプションを使用して、後の時点までスクラビングを遅らせることができます。このオプションは、スクラビングできるまで、イメージが **pending_delete** 状態である期間を示します。 これによりストレージに影響が出ることを認識することが重要です。**scrub_time** の値が大きいほど、削除されるイメージからバックエンドストレージを再要求する時間が長くなります。 値: ● 負の値以外の整数 関連するオプション: ● **delayed_delete**
secure_proxy_ssl_header = None	文字列値	SSL 終端プロキシーによって削除された場合でも、元のリクエストのスキームを決定するために使用される HTTP ヘッダー。通常の値は HTTP_X_FORWARDED_PROTO です。

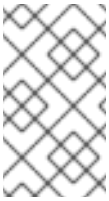
設定オプション = デフォルト値	型	説明
show_image_direct_url = False	ブール値	イメージを返す際に直接イメージの場所を示します。 この設定オプションは、イメージの詳細をユーザーに返すときに直接イメージの場所を表示するかどうかを指定します。直接イメージの場所は、バックエンドストレージ内でイメージデータが保存される場所です。このイメージの場所は、イメージ属性 **direct_url** セクションに表示されます。 イメージに対して複数のイメージの場所が存在する場合には、設定オプション **location_strategy** で示される場所ストラテジーに基づいて、最適な場所が表示されます。 注記: ● イメージの場所を表示すると、イメージの場所に認証情報が含まれることがあるため、重大なセキュリティーリスクが発生する可能性があります。したがって、これはデフォルトで **False** に設定されています。設定の意味するところを理解している場合に限り、細心の注意を払ってこのオプションを **True** に設定してください。 ● 運用者がユーザーにイメージの場所を表示しないことを希望する場合は、このオプションと **show_multiple_locations** の両方を **False** に設定する必要があります。 値: ● True ● False 関連するオプション: ● show_multiple_locations ● location_strategy

設定オプション = デフォルト値	型	説明
show_multiple_locations = False	ブール値	イメージを返す際にすべてのイメージの場所を示します。 この設定オプションは、イメージの詳細をユーザーに返すときにすべてのイメージの場所を表示するかどうかを指定します。イメージに対して複数のイメージの場所が存在する場合には、設定オプション **location_strategy** で示される場所ストラテジーに基づいて、場所が順番に表示されます。イメージの場所は、イメージ属性 **locations** セクションに表示されます。 注記: ● イメージの場所を表示すると、イメージの場所に認証情報が含まれることがあるため、重大なセキュリティーリスクが発生する可能性があります。したがって、これはデフォルトで **False** に設定されています。設定の意味するところを理解している場合に限り、細心の注意を払ってこのオプションを **True** に設定してください。 ● 詳細は、https://wiki.openstack.org/wiki/OSSN/OSSN-0065 を参照してください。 ● 運用者がユーザーにイメージの場所を表示しないことを希望する場合は、このオプションと **show_image_direct_url** の両方を **False** に設定する必要があります。 値: ● True ● False 関連するオプション: ● show_image_direct_url ● location_strategy **非推奨となったバージョン：** Newton *理由:* Newton 以降非推奨になったこのオプションの使用はセキュリティーリスクであり、現在このオプションを必要とするそれらのユースケースを満たす方法がわかれば削除されます。ポリシーを使用することで同じ機能をより細かい粒度で実現できるという以前の通知は、正しくありません。ポリシー設定でこのオプションを回避することが取るべき修正の方向と考えていますが、現在、この方法で回避することはできません。この問題への対処に関する最新の進捗状況を把握するために、常に Glance のリリースノートに注意を払ってください。

設定オプション = デフォルト値	型	説明
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
tcp_keepidle = 600	整数値	接続を再チェックするまでの待機時間を設定します。 TCP キープアライブパケットをホストに送信するまでのアイドル待機時間として設定される時間 (秒単位) を表す正の整数値を指定します。デフォルト値は 600 秒です。 **tcp_keepidle** を設定すると、接続が健全であることを定期的に確認するのに役立ち、TCP 接続を頻繁に再確立するのを防ぎます。 値: ● 時間を表す正の整数値 (秒単位) 関連するオプション: ● なし
transport_url = rabbit://	文字列値	メッセージングバックエンドに接続するためのネットワークアドレスおよびオプションのユーザー認証情報 (URL 形式)。想定される形式は次のとおりです。 driver://[user:pass@]host:port[, [userN:passN@]hostN:portN]/virtual_host?query 例:rabbit://rabbitmq:password@127.0.0.1:5672// URL のフィールドの詳細は、https://docs.openstack.org/oslo.messaging/latest/reference/transport.html で oslo_messaging.TransportURL のドキュメントを参照してください。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
user_storage_quota = 0	文字列値	テナントごとのイメージストレージの最大容量。 これにより、全ストアにわたってテナントのすべてのイメージによって消費される累積ストレージに上限が適用されます。これはテナントごとの制限です。 この設定オプションのデフォルトの単位はバイトです。ただし、ストレージの単位は、大文字/小文字を区別する文字**B**、**KB**、**MB**、**GB**、および **TB**を使用して指定することができ、それぞれバイト、キロバイト、メガバイト、ギガバイト、およびテラバイトを表します。値と単位の間にはスペースを入れないでください。値 **0** は、クォータが適用されないことを意味します。負の値は無効で、エラーが発生します。 値: ● 上記のように、ストレージの値を表す負ではない整数とストレージの単位を表すオプションの文字列の有効な組み合わせである文字列。 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。
worker_self_reference_url = None	文字列値	このワーカーへの URL。 これが設定されている場合、他のグランスワーカーは、必要に応じてこのワーカーに直接連絡する方法を知っています。イメージのインポートの場合、1 人のワーカーがイメージをステージングし、他のワーカーがインポートリクエストを適切なワーカーにプロキシーできる必要があります。 設定されていない場合、これは **public_endpoint** と見なされ、通常はすべてのワーカーで同じ値に設定され、プロキシー動作を効果的に無効にします。 値: ● このワーカーが他のワーカーから到達可能な URL 関連するオプション: ● public_endpoint

設定オプション = デフォルト値	型	説明
workers = None	整数値	起動する Glance ワーカープロセスの数。 要求を処理する子プロセスワーカーの数を設定する、負ではない整数値を指定します。デフォルトでは、使用可能な CPU の数は、8 に制限された**workers**の値として設定されます。たとえば、プロセッサー数が 6 の場合、6 つのワーカーが使用され、プロセッサー数が 24 の場合は、8 つのワーカーだけが使用されます。制限はデフォルト値にのみ適用され、24 のワーカーが設定されている場合、24 が使用されます。 各ワーカープロセスは、設定ファイルで設定されたポートでリッスンするように指定され、サイズ 1000 のグリーンスレッドプールが含まれています。  **注記** ワーカーの数をゼロに設定すると、サイズ 1000 のグリーンスレッドプールを持つ単一の API プロセスの作成がトリガーされます。 値: ● 0 ● 正の整数値 (通常は CPU の数と同じ) 関連するオプション: ● なし

3.1.2. cinder

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[cinder]** グループで利用可能なオプションの概要をまとめたものです。

表3.1 cinder

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
cinder_api_insecure = False	ブール値	cinder に対するセキュアではない SSL リクエストの実行を許可します。 このオプションが True に設定されている場合、HTTPS エンドポイント接続は **cinder_ca_certificates_file** オプションで指定された CA 証明書ファイルを使用して検証されます。 値: ● True ● False 関連するオプション: ● cinder_ca_certificates_file
cinder_ca_certificates_file = None	文字列値	cinder クライアントリクエストに使用する CA 証明書ファイルの場所。 CA 証明書ファイルが設定されている場合、指定されたファイルが、HTTPS エンドポイントを使用した cinder 接続の検証に使用されます。エンドポイントが HTTP の場合、この値は無視されます。検証を有効にするには、**cinder_api_insecure** を **True** に設定する必要があります。 値: ● CA 証明書ファイルへのパス 関連するオプション: ● cinder_api_insecure

設定オプション = デフォルト値	型	説明
cinder_catalog_info = volumev3::publicURL	文字列値	サービスカタログで cinder を検索する際に照合する情報。 **cinder_endpoint_template** が設定されておらず、**cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、**cinder_store_password** のいずれも設定されていない場合、cinder store はこの情報を使用して、現在のコンテキストでサービスカタログから cinder エンドポイントを検索します。**cinder_os_region_name** が設定されている場合は、適切なエンドポイントを取得するために考慮されます。 サービスカタログは、**openstack catalog list** コマンドでリスト表示できます。 値: ● 次の形式の文字列: **<service_type>:<service_name>:<interface>** 少なくとも **service_type** および **interface** を指定する必要があります。**service_name** は省略できます。 関連するオプション: ● cinder_os_region_name ● cinder_endpoint_template ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name ● cinder_store_password

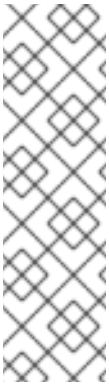
設定オプション = デフォルト値	型	説明
cinder_endpoint_template = None	文字列値	cinder エンドポイントのテンプレートでサービスカタログの検索を上書きします。 このオプションを設定すると、サービスカタログから検索するのではなく、この値を使用して cinder エンドポイントが生成されます。**cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、および **cinder_store_password** が指定されている場合には、この値は無視されます。 この設定オプションが設定されている場合、**cinder_catalog_info** は無視されます。 値: ● cinder エンドポイントの URL テンプレート文字列。ここで、**%%(tenant)s** は現在のテナント (プロジェクト) 名に置き換えます。例:**http://cinder.openstack.example.org/v2/%%(tenant)s** 関連するオプション: ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name ● cinder_store_password ● cinder_catalog_info
cinder_enforce_multipath = False	ブール値	これが True に設定されている場合、multipathd が実行されていない場合、イメージ転送用のボリュームのアタッチが中断されます。それ以外の場合は、単一のパスにフォールバックします。 値: ● True か False 関連するオプション: ● cinder_use_multipath

設定オプション = デフォルト値	型	説明
cinder_http_retries = 3	整数値	失敗した http 呼び出しでの cinderclient のリトライ回数。 何らかのエラーによって呼び出しに失敗すると、cinderclient は数秒スリープした後に呼び出しを指定の回数までリトライします。 値: ● 正の整数 関連するオプション: ● なし
cinder_mount_point_base = /var/lib/glance/mnt	文字列値	glance ノードで NFS ボリュームがマウントされるディレクトリー。 値: ● マウントポイントの絶対パスを表す文字列。
cinder_os_region_name = None	文字列値	サービスカタログから cinder サービスを検索する際のリージョン名。 これは、**cinder_catalog_info** を使用してエンドポイントを決定する場合にのみ使用されます。これが設定されている場合、このノードによる cinder エンドポイントの検索は、指定のリージョンにフィルターされます。これは、カタログに複数のリージョンがリスト表示される場合に役立ちます。これが設定されていない場合、エンドポイントはすべてのリージョンから検索されます。 値: ● 有効なリージョン名である文字列。 関連するオプション: ● cinder_catalog_info

設定オプション = デフォルト値	型	説明
cinder_state_transition_timeout = 300	整数値	cinder ボリュームの遷移が完了するまで待機する時間 (秒単位)。 ボリュームデータの読み取り/書き込みを行うために、cinder ボリュームを作成、削除、または glance ノードにアタッチすると、ボリュームの状態が変更されます。たとえば、新規作成されたボリュームのステータスは、作成プロセスの完了後に **creating** から **available** に変わります。このオプションは、ステータス変更を待機する最大時間を指定します。待機時間がタイムアウトするか、ステータスが予期しない値に変更される場合 (**error**など)、イメージの作成は失敗します。 値: ● 正の整数 関連するオプション: ● なし
cinder_store_auth_address = None	文字列値	cinder 認証サービスがリッスンしているアドレス。 **cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、および **cinder_store_password**オプションのすべてが指定された場合、指定した値は常に認証に使用されます。これは、イメージサービスに固有のプロジェクト/テナントにイメージボリュームを保存することにより、ユーザーからイメージボリュームを隠すのに役立ちます。また、ユーザーは glance の ACL の制御下で、他のプロジェクト間でイメージボリュームを共有できます。 これらのオプションのいずれかが設定されていない場合、cinder エンドポイントはサービスカタログから検索され、現在のコンテキストのユーザーとプロジェクトが使用されます。 値: ● 有効な認証サービスアドレス (例:**http://openstack.example.org/identity/v2.0**)。 関連するオプション: ● cinder_store_user_name ● cinder_store_password ● cinder_store_project_name

設定オプション = デフォルト値	型	説明
cinder_store_password = None	文字列値	cinder に対して認証を行うユーザーのパスワード。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのユーザーが使用されます。 値: ● **cinder_store_user_name**で指定したユーザーの有効なパスワード 関連するオプション: ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name
cinder_store_project_name = None	文字列値	cinder にイメージボリュームが保存されるプロジェクト名。 この設定オプションが設定されていない場合は、現在のコンテキストのプロジェクトが使用されます。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのプロジェクトが使用されます。 値: ● 有効なプロジェクト名 関連するオプション: ● **cinder_store_auth_address** ● **cinder_store_user_name** ● **cinder_store_password**

設定オプション = デフォルト値	型	説明
cinder_store_user_name = None	文字列値	cinder に対して認証を行うためのユーザー名。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのユーザーが使用されます。 値: ● 有効なユーザー名 関連するオプション: ● cinder_store_auth_address ● cinder_store_password ● cinder_store_project_name
cinder_use_multipath = False	ブール値	デプロイメントでマルチパスがサポートされるかどうかを識別するためのフラグ。 マルチパスがサポートされない場合は、False に設定します。 値: ● True か False 関連するオプション: ● cinder_enforce_multipath

設定オプション = デフォルト値	型	説明
cinder_volume_type = None	文字列値	cinder でのボリュームの作成に使用されるボリューム種別。 一部の cinder バックエンドには、ストレージの使用を最適化するための複数のボリューム種別が含まれる場合があります。このオプションを追加すると、運用者はイメージに対して最適化できる特定のボリューム種別を cinder で選択することができます。 これが設定されていない場合には、cinder 設定で指定したデフォルトのボリューム種別がボリュームの作成に使用されます。 値: ● cinder からの有効なボリューム種別 関連するオプション: ● なし  **注記** NFS バックエンドに関連付けられた暗号化された volume_type は使用できません。NFS バックエンドに保存されている暗号化されたボリュームは、glance_store がそのボリュームに保存されているイメージデータの書き込みまたはアクセスを試みるたびに、例外を発生させます。Cinder 管理者に相談して、適切な volume_type を決定してください。
rootwrap_config = /etc/glance/rootwrap.conf	文字列値	root としてコマンドを実行するのに使用する rootwrap 設定ファイルへのパス。 cinder ストアでは、イメージボリュームを操作するのに root 権限が必要です (iSCSI/FC ボリュームへの接続、ボリュームデータの読み取り/書き込みのためなど)。設定ファイルでは、cinder ストアおよび os-brick ライブラリーによる必要なコマンドを許可する必要があります。 値: ● rootwrap 設定ファイルへのパス 関連するオプション: ● なし

3.1.3. cors

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[cors]** グループで利用可能なオプションの概要をまとめたものです。

表3.2 cors

設定オプション = デフォルト値	型	説明
allow_credentials = True	ブール値	実際の要求にユーザーの認証情報を含めることができることを示します。
allow_headers = ['Content-MD5', 'X-Image-Meta-Checksum', 'X-Storage-Token', 'Accept-Encoding', 'X-Auth-Token', 'X-Identity-Status', 'X-Roles', 'X-Service-Catalog', 'X-User-Id', 'X-Tenant-Id', 'X-OpenStack-Request-ID']	リスト値	実際の要求時に使用されるヘッダーフィールド名を示します。
allow_methods = ['GET'、 'PUT'、 'POST'、 'DELETE'、 'PATCH']	リスト値	実際の要求時に使用できるメソッドを示します。
allowed_origin = None	リスト値	このリソースがリクエストの origin"ヘッダーで受信したドメインと共有されるかどうかを示します。形式:"<protocol>://<host>[:<port>]" (行末のスラッシュなし)例:https://horizon.example.com
expose_headers = ['X-Image-Meta-Checksum', 'X-Auth-Token', 'X-Subject-Token', 'X-Service-Token', 'X-OpenStack-Request-ID']	リスト値	API に安全に公開できるヘッダーを示します。デフォルトは HTTP Simple ヘッダーです。
max_age = 3600	整数値	CORS プリフライトリクエストの最大キャッシュ期間。

3.1.4. database

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[database]** グループで利用可能なオプションの概要をまとめたものです。

表3.3 database

設定オプション = デフォルト値	型	説明
backend = sqlalchemy	文字列値	データベースに使用するバックエンド。
connection = None	文字列値	データベースへの接続に使用する SQLAlchemy 接続文字列。
connection_debug = 0	整数値	SQL デバッグ情報の冗長性: 0=なし、100=すべて。
`connection_parameters = `	文字列値	接続時に接続 URL に追加するオプションの URL パラメーター。param1=value1&param2=value2&... として指定します。
connection_recycle_time = 3600	整数値	この秒数より長く接続プールに存在していた接続は、次にプールからチェックアウトされたときに新しい接続に置き換えられます。
connection_trace = False	ブール値	Python スタックトレースをコメント文字列として SQL に追加します。
db_inc_retry_interval = True	ブール値	True の場合には、db_max_retry_interval までのデータベース操作の再試行回数を長くします。
db_max_retries = 20	整数値	エラーが発生する前に接続エラーまたはデッドロックが発生した場合の最大再試行。再試行回数を無限に指定するには -1 に設定します。
db_max_retry_interval = 10	整数値	db_inc_retry_interval が設定されている場合は、データベース操作を再試行するまでの最大秒数です。
db_retry_interval = 1	整数値	データベーストランザクションの再試行間の秒数。
max_overflow = 50	整数値	設定されている場合、この値を SQLAlchemy で max_overflow に使用します。
max_pool_size = 5	整数値	プールに開いたままにする SQL 接続の最大数。値を 0 に設定すると無制限を意味します。
max_retries = 10	整数値	起動時のデータベース接続の最大再試行数。再試行回数を無限に指定するには -1 に設定します。
mysql_enable_ndb = False	ブール値	True の場合には、MySQL Cluster (NDB) の処理のサポートを透過的に有効にします。

設定オプション = デフォルト値	型	説明
mysql_sql_mode = TRADITIONAL	文字列値	MySQL セッションに使用する SQL モード。このオプション (デフォルトを含む) は、サーバーセット SQL モードを上書きします。サーバー設定で設定された SQL モードを使用するには、これを no value に設定します。例: mysql_sql_mode=
pool_timeout = None	整数値	設定されている場合は、この値を SQLAlchemy で pool_timeout に使用します。
retry_interval = 10	整数値	SQL 接続を開く再試行の間隔。
slave_connection = None	文字列値	スレーブデータベースへの接続に使用する SQLAlchemy 接続文字列。
sqlite_synchronous = True	ブール値	True の場合、SQLite は同期モードを使用します。
use_db_reconnect = False	ブール値	失われた接続でデータベースの実験的な使用を有効にします。
use_tpool = False	ブール値	すべての DB API 呼び出しのスレッドプールの実験的な使用を有効にします。

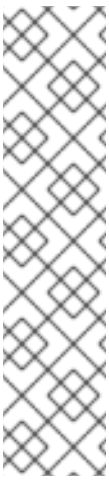
3.1.5. file

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[file]** グループで利用可能なオプションの概要をまとめたものです。

表3.4 file

設定オプション = デフォルト値	型	説明
filesystem_store_chunk_size = 65536	整数値	チャンクサイズ (バイト単位)。 イメージファイルの読み取り/書き込み時に使用されるチャンクサイズ。この値を大きくするとスループットが向上しますが、大量のリクエストを処理する際にメモリー使用量が若干増加する場合もあります。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
filesystem_store_datadir = /var/lib/glance/images	文字列値	ファイルシステムのバックエンドストアがイメージを書き込むディレクトリー。 起動時に、Glance はディレクトリーが存在しない場合は作成し、**glance-api** が実行されるユーザーへの書き込みアクセスを検証します。書き込みアクセスが使用できない場合は、**BadStoreConfiguration** 例外が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 **注記** このディレクトリーは、ファイルシステムストアがストレージバックエンドとして使用される場合にのみ使用されます。**filesystem_store_datadir** または **filesystem_store_datadirs** オプションのいずれかを **glance-api.conf** で指定する必要があります。両方のオプションを指定すると、**BadStoreConfiguration** が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 値: ● ディレクトリーへの有効なパス 関連するオプション: ● **filesystem_store_datadirs** ● **filesystem_store_file_perm**

設定オプション = デフォルト値	型	説明
filesystem_store_datadirs = None	多値	ファイルシステムのバックエンドストアがイメージを書き込むディレクトリーとその優先順位のリスト。 **filesystem_store_datadir** 設定オプションで指定した単一のディレクトリーを使用する場合とは対照的に、ファイルシステムストアは複数のディレクトリーにイメージを格納するように設定できます。複数のディレクトリーを使用する場合、各ディレクトリーには、使用される優先順位を指定するオプションの優先度を指定できます。優先度は、ディレクトリーパスとコロンで組み合わされる整数で、値が大きいほど優先度が高くなります。2 つのディレクトリーの優先度が同じ場合は、空き容量が最も多いディレクトリーが使用されます。優先度を指定しないと、デフォルトでゼロに設定されます。 複数のストアディレクトリーを持つファイルシステムストアの設定に関する詳細は、https://docs.openstack.org/glance/latest/configuration/configuring.html を参照してください。  **注記** このディレクトリーは、ファイルシステムストアがストレージバックエンドとして使用される場合にのみ使用されます。**filesystem_store_datadir** または **filesystem_store_datadirs** オプションのいずれかを **glance-api.conf** で指定する必要があります。両方のオプションを指定すると、**BadStoreConfiguration** が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 値: - 以下の形式の文字列のリスト: - **<a valid directory path>:<optional integer priority>** 関連するオプション: - **filesystem_store_datadir** - **filesystem_store_file_perm**

設定オプション = デフォルト値	型	説明
filesystem_store_file_perm = 0	整数値	イメージファイルへのファイルアクセスパーミッション。 イメージデータへの目的のファイルアクセスパーミッションを設定します。これにより、Nova などの他のサービスがファイルシステムストアから直接イメージを使用できるようにすることができます。アクセスが付与されるサービスを実行中のユーザーは、作成されるファイルを所有しているグループのメンバーにすることができます。この設定オプションにゼロ以下の値を割り当てることは、デフォルトのパーミッションに変更が加えられないことを意味します。この値は、8 進数の数字としてデコードされます。 詳細は、https://docs.openstack.org/glance/latest/configuration/configuring.htmlのドキュメントを参照してください。 値: ● 有効なファイルアクセスパーミッション ● ゼロ ● 任意の負の整数 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
filesystem_store_metadata_file = None	文字列値	ファイルシステムストアのメタデータファイル。 ファイルシステムストアに関連付けられた場所と共に返されるメタデータが含まれるファイルへのパス。このオプションが設定されると、後で作成される新しいイメージにのみ使用されます。以前の既存のイメージには影響しません。 このファイルには、有効な JSON オブジェクトが含まれている必要があります。オブジェクトには、キー **ID** と **マウントポイント** が含まれている必要があります。両方のキーの値は文字列でなければなりません。 値: ● ストアメタデータファイルへの有効なパス 関連するオプション: ● なし
filesystem_thin_provisioning = False	ブール値	このバックエンドでシンプロビジョニングを有効にするかどうか。 この設定オプションを使用すると、ファイルシステムに null バイトシーケンスを実際に書き込まないという機能が有効になります。表示されるホールは、ファイルシステムによって自動的に null バイトとして解釈され、ストレージを実際に消費しません。この機能を有効にすると、ネットワーク経由で null バイトシーケンスが送信されないため、バックエンドのスペースを節約できるだけでなく、イメージのアップロードが高速化され、ネットワークトラフィックが節約されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● なし

3.1.6. glance.store.http.store

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[glance.store.http.store]** グループで利用可能なオプションの概要をまとめたものです。

表3.5 glance.store.http.store

設定オプション = デフォルト値	型	説明
http_proxy_information = {}	dict 値	リモートサーバーへの接続に使用する http/https プロキシー情報。 この設定オプションは、リモートサーバーへの接続に使用する必要のある http/https プロキシー情報を指定します。プロキシー情報は、スキームとプロキシーのキー/値のペアにする必要があります (例:http:10.0.0.1:3128)。また、キーと値のペアをコンマで区切ることで、複数のスキームのプロキシーを指定することもできます (例: http:10.0.0.1:3128, https:10.0.0.1:1080)。 値: ● 上記のように スキーム: プロキシーペアのコンマ区切りリスト 関連するオプション: ● なし
https_ca_certificates_file = None	文字列値	CA バンドルファイルへのパス。 この設定オプションにより、運用者はカスタム認証局ファイルを使用してリモートサーバー証明書を検証できます。このオプションを設定すると、**https_insecure** オプションは無視され、指定された CA ファイルがサーバー証明書の認証に使用され、サーバーへのセキュアな接続が確立されます。 値: ● CA ファイルへの有効なパス 関連するオプション: ● https_insecure

設定オプション = デフォルト値	型	説明
https_insecure = True	ブール値	リモートサーバー証明書の検証を設定します。 この設定オプションはブール値を取り、リモートサーバーの証明書を検証するかどうかを決定します。True に設定すると、リモートサーバー証明書は検証されません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 このオプションは、**https_ca_certificates_file** が設定されている場合は無視されます。リモートサーバー証明書は、**https_ca_certificates_file** オプションを使用して指定されたファイルを使用して検証されます。 値: ● True ● False 関連するオプション: ● https_ca_certificates_file

3.1.7. glance.store.rbd.store

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[glance.store.rbd.store]** グループで利用可能なオプションの概要をまとめたものです。

表3.6 glance.store.rbd.store

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
rados_connect_timeout = 0	整数値	Ceph クラスターに接続する際のタイムアウト値。 この設定オプションは、Ceph クラスターに接続する際に使用されるタイムアウト値 (秒単位) を取ります。つまり、接続を閉じるまで glance-api が待機する時間を設定します。これにより、RBD への接続中に glance-api がハングアップするのを防ぐことができます。このオプションの値を 0 以下に設定すると、タイムアウトは設定されず、デフォルトの librados 値が使用されます。 以下の値を使用できます。 ● 任意の整数値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
`rbd_store_ceph_conf = `	文字列値	Ceph 設定ファイルへのパス。 この設定オプションは、使用する Ceph 設定ファイルへのパスを指定します。このオプションの値がユーザーにより設定されていない場合や、空の文字列に設定されている場合、librados は、デフォルトの Ceph 設定ファイルの場所を順に検索して標準の ceph.conf ファイルを読み取ります。詳細は、Ceph のドキュメントを参照してください。 **注記** Cephx 認証を使用する場合、このファイルの client.<USER>セクションに適切なキーリングへの参照を含める必要があります。 注 2: このオプションを空白のままにすると (デフォルト)、使用される実際の Ceph 設定ファイルは、使用されている librados のバージョンによって変わる可能性があります。有効になっている設定ファイルを正確に把握しておくことが重要な場合は、このオプションを使用してここにファイルを指定できます。 以下の値を使用できます。 - 設定ファイルへの有効なパス 関連するオプション: - rbd_store_user

設定オプション = デフォルト値	型	説明
rbd_store_chunk_size = 8	整数値	RADOS イメージをチャンクに分割する際のサイズ (メガバイト単位)。 Glance イメージをチャンクに分割する際のサイズ (メガバイト単位) を表す整数値を指定します。デフォルトのチャンクサイズは 8 メガバイトです。最適なパフォーマンスを得るには、値は 2 の累乗でなければなりません。 Ceph の RBD オブジェクトストレージシステムを Glance イメージを保存するためのストレージバックエンドとして使用する場合、イメージはこのオプションを使用して設定されるサイズのオブジェクトに分割されます。これらの分割されたオブジェクトは、分散ブロックデータストア全体に保存され、Glance に使用されます。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし
rbd_store_pool = images	文字列値	イメージが保存される RADOS プール。 RBD を Glance イメージを保存するためのストレージバックエンドとして使用する場合、イメージは **プール** へのオブジェクト (イメージのチャンク) の論理的なグループ化により保存されます。各プールは、含めることのできる配置グループの数で定義されます。使用されるデフォルトのプールは **イメージ** です。 RBD ストレージバックエンドの詳細は、http://ceph.com/planet/how-data-is-stored-in-ceph-cluster/を参照してください。 以下の値を使用できます。 ● 有効なプール名 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
rbd_store_user = None	文字列値	認証する RADOS ユーザー。 この設定オプションは、認証する RADOS ユーザーを設定します。このオプションは RADOS 認証が有効になっている場合にのみ必要で、ユーザーが Cephx 認証を使用している場合にのみ適用されます。このオプションの値がユーザーによって設定されていない場合や、None に設定されている場合には、デフォルト値が選択されます。これは、rbd_store_ceph_conf の client. セクションに基づきます。 以下の値を使用できます。 ● 有効な RADOS ユーザー 関連するオプション: ● rbd_store_ceph_conf
rbd_thin_provisioning = False	ブール値	このバックエンドでシンプロビジョニングを有効にするかどうか。 この設定オプションを使用すると、RBD バックエンドに null バイトシーケンスを実際に書き込まないという機能が有効になります。表示されるホールは、Ceph によって自動的に null バイトとして解釈され、ストレージを実際に消費しません。この機能を有効にすると、ネットワーク経由で null バイトシーケンスが送信されないため、バックエンドのスペースを節約できるだけでなく、イメージのアップロードが高速化され、ネットワークトラフィックが節約されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● なし

3.1.8. glance.store.s3.store

次の表は、**/etc/glance/glance-api.conf** ファイルの **[glance.store.s3.store]** グループで使用できるオプションの概要を示しています。

表3.7 glance.store.s3.store

設定オプション = デフォルト値	型	説明
s3_store_access_key = None	文字列値	S3 クエリートークンアクセスキー。 この設定オプションは、Amazon S3 または S3 互換ストレージサーバーで認証するためのアクセスキーを取得します。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。 値: ● 適切な権限を持つユーザーのアクセスキーである任意の文字列値 関連するオプション: ● s3_store_host ● s3_store_secret_key
s3_store_bucket = None	文字列値	Glance データの保存に使用される S3 バケット。 この設定オプションは、Glance イメージが S3 に保存される場所を指定します。**s3_store_create_bucket_on_put** を true に設定すると、バケットが存在しなくても自動的に作成されます。 値: ● 任意の文字列値 関連するオプション: ● s3_store_create_bucket_on_put ● s3_store_bucket_url_format

設定オプション = デフォルト値	型	説明
s3_store_bucket_url_format = auto	文字列値	オブジェクトを決定するために使用される S3 呼び出し形式。 この設定オプションは、S3 バケット内のオブジェクトのアドレスを指定するために使用されるアクセスモデルを取ります。 注記: **path** スタイルでは、オブジェクトのエンドポイントは https://s3.amazonaws.com/bucket/example.img のようになります。**virtual** スタイルでは、オブジェクトのエンドポイントは https://bucket.s3.amazonaws.com/example.img のようになります。バケット名で DNS 命名規則に従わない場合、パススタイルでオブジェクトを取得できますが、仮想スタイルでは取得できません。 値: ● **auto**、**virtual**、または **path** の任意の文字列値 関連するオプション: ● s3_store_bucket
s3_store_create_bucket_on_put = False	ブール値	S3 が新しいバケットを作成する必要があるかどうかを決定します。 この設定オプションはブール値を取り、Glance が S3 に新しいバケットを作成する必要があるかどうかを示します (存在しない場合)。 値: ● 任意のブール値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
s3_store_host = None	文字列値	S3 サーバーがリッスンしているホスト。 この設定オプションは、S3 または S3 互換ストレージサーバーのホストを設定します。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。ホストには、DNS 名 (s3.amazonaws.com、my-object-storage.com など) または IP アドレス (127.0.0.1) を含めることができます。 値: ● 有効な DNS 名 ● 有効な IPv4 アドレス 関連するオプション: ● s3_store_access_key ● s3_store_secret_key
s3_store_large_object_chunk_size = 10	整数値	パーツのアップロード時に S3 が使用するマルチパートアップロードパーツのサイズ (MB 単位)。 この設定オプションは、マルチパートアップロードのイメージ分割サイズを MB 単位で取得します。 注記: 分割できるイメージは 10,000 個までです。 値: ● 任意の正の整数値 (5M 以上である必要があります) 関連するオプション: ● s3_store_large_object_size ● s3_store_thread_pools

設定オプション = デフォルト値	型	説明
s3_store_large_object_size = 100	整数値	S3 がイメージファイルのチャンク化を開始し、S3 でマルチパートアップロードを実行する必要があるサイズ (MB 単位)。 この設定オプションは、イメージをそのまま S3 にアップロードするか、分割するか (マルチパートアップロード) を決定するために MB 単位でしきい値を取得します。 注記: 分割できるイメージは 10,000 個までです。 値: ● 任意の正の整数値 関連するオプション: ● s3_store_large_object_chunk_size ● s3_store_thread_pools
s3_store_secret_key = None	文字列値	S3 クエリートークンの秘密鍵。 この設定オプションは、Amazon S3 または S3 互換ストレージサーバーで認証するための秘密鍵を受け取ります。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。 値: ● **s3_store_host** オプションを使用して指定されたアクセスキーに対応する秘密鍵である任意の文字列値 関連するオプション: ● s3_store_host ● s3_store_access_key

設定オプション = デフォルト値	型	説明
s3_store_thread_pools = 10	整数値	S3 でマルチパートアップロードを実行するためのスレッドプールの数。 この設定オプションは、マルチパートアップロードの実行時にスレッドプールの数を取得します。 値: ● 任意の正の整数値 関連するオプション: ● s3_store_large_object_size ● s3_store_large_object_chunk_size

3.1.9. glance.store.swift.store

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[glance.store.swift.store]** グループで利用可能なオプションの概要をまとめたものです。

表3.8 glance.store.swift.store

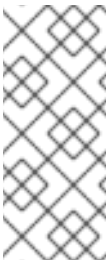
設定オプション = デフォルト値	型	説明
default_swift_reference = ref1	文字列値	デフォルトの Swift アカウント/バッキングストアパラメーターへの参照。 イメージストレージに swift アカウント/バッキングストアを使用するのに必要なデフォルトのパラメーターセットへの参照を表す文字列値を指定します。この設定オプションのデフォルト参照値は **ref1** です。この設定オプションはパラメーターを逆参照し、新しいイメージが追加されるたびに Swift ストレージバックエンドでのイメージ保管を容易にします。 値: ● 有効な文字列値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_buffer_on_upload = False	ブール値	Swift にアップロードする前にイメージのセグメントをバッファー処理します。 swift へのアップロード中に Glance がイメージデータをディスクにバッファーするかどうかを指定するブール値を指定します。これにより、Glance はエラー時にアップロードを再開できます。 注意: このオプションを有効にすると、API ノードのディスク使用量が増加するので、細心の注意を払う必要があります。ファイルシステムの設定状況によっては、バッファー処理に使用されるディスク領域により、glance イメージキャッシュに利用可能な実際のディスク容量が減る可能性があることに注意してください。ディスク使用率は、式**swift_store_large_object_chunk_size** * **workers** * 1000 に従って上限が設定されます。 値: ● True ● False 関連するオプション: ● swift_upload_buffer_dir
swift_store_admin_tenants = []	リスト値	管理者アクセスが付与されるテナントのリスト。 これは、マルチテナントモードの Glance によって作成されたすべての Swift コンテナーでの読み取り/書き込みアクセスが付与されるテナントのリストです。デフォルト値は空のリストです。 値: ● Keystone プロジェクト/テナントの UUID を表す文字列のコンマ区切りリスト 関連するオプション: ● なし
swift_store_auth_address = None	文字列値	Swift 認証サービスがリッスンしているアドレス。

設定オプション = デフォルト値	型	説明
swift_store_auth_insecure = False	ブール値	サーバー証明書の検証を設定します。 このブール値は、サーバー証明書を検証するかどうかを決定します。このオプションが True に設定されている場合、swiftclient は認証時に有効な SSL 証明書かどうかをチェックしません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 値: ● True ● False 関連するオプション: ● swift_store_cacert
swift_store_auth_version = 2	文字列値	使用する認証サービスのバージョン。有効なバージョンは、keystone の場合は 2 および 3 で、swauth および rackspace の場合は 1(非推奨) です。
swift_store_cacert = None	文字列値	CA バンドルファイルへのパス。 この設定オプションにより、運用者は Swift に接続する際の SSL 検証用のカスタム認証局ファイルへのパスを指定できます。 値: ● CA ファイルへの有効なパス 関連するオプション: ● swift_store_auth_insecure

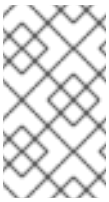
設定オプション = デフォルト値	型	説明
swift_store_config_file = None	文字列値	swift アカウント設定が含まれるファイルへの絶対パス。 設定済みの各 Swift アカウント/バッキングストアへの参照を持つ設定ファイルへのパスを表す文字列値を含めます。デフォルトでは、ファイルパスが指定されておらず、カスタマイズされた Swift 参照は無効になっています。データベースへの認証情報の保存を回避するため、このオプションを設定することは、イメージストレージに Swift ストレージバックエンドを使用する際に強く推奨されます。 **注記** **swift_store_multi_tenant** を **True** に設定した場合は、このオプションを設定しないでください。 値: - glance-api ノードの絶対パスを表す文字列値 関連するオプション: - swift_store_multi_tenant

設定オプション = デフォルト値	型	説明
swift_store_container = glance	文字列値	イメージを保存するための単一のコンテナーの名前/複数のコンテナーの名前の接頭辞 1 つのコンテナーを使用してイメージを保存する場合、この設定オプションは、すべてのイメージの保存に使用される Glance アカウント内のコンテナーを示します。複数のコンテナーがイメージの保存に使用される場合、これはすべてのコンテナーの名前の接頭辞になります。単一/複数コンテナーの使用は、設定オプション **swift_store_multiple_containers_seed** を使用して制御できます。 複数のコンテナーを使用する場合、コンテナーの名前は、この設定オプションに設定された値をベースに、イメージ UUID の最初の N 文字をアンダースコアで区切られた接尾辞として指定されます (N は **swift_store_multiple_containers_seed**によって指定されます)。 例: シードが 3 で swift_store_container = **glance** と設定されている場合には、UUID が**fdae39a1-bac5-4238-aba4-69bcc726e848** のイメージは **glance_fda** のコンテナーに配置されます。コンテナー名の作成時に、UUID のすべてのダッシュは含まれますが、文字の制限にはカウントされないため、N=10 の場合はコンテナー名は **glance_fdae39a1-ba** になります。 値: ● 単一のコンテナーを使用する場合、この設定オプションは Glance の Swift アカウントの有効な任意の swift コンテナーの名前の文字列にすることができます。 ● 複数のコンテナーを使用する場合、この設定オプションは、 Swift により適用されるコンテナーの命名ルールを満たす限り任意の文字列になります。**swift_store_multiple_containers_seed** の値も考慮する必要があります。 関連するオプション: ● **swift_store_multiple_containers_seed** ● **swift_store_multi_tenant** ● **swift_store_create_container_on_put**

設定オプション = デフォルト値	型	説明
swift_store_create_container_on_put = False	ブール値	イメージのアップロード時に、コンテナーを作成します (まだ存在しない場合)。 イメージのアップロード時に、対応するコンテナーが存在しない場合は、この設定オプションが True に設定されている場合に作成されます。デフォルトでは、コンテナーは作成されません。この動作は、単一および複数コンテナーモードの両方に適用されます。 値: ● True ● False 関連するオプション: ● なし
swift_store_endpoint = None	文字列値	Swift バックエンドストレージに使用する URL エンドポイント。 Glance イメージを Swift ストアに格納するために使用する URL エンドポイントを表す文字列値を指定します。デフォルトでは、エンドポイントは設定されず、**auth** によって返されるストレージ URL が 使用されます。**swift_store_endpoint** を使用してエンドポイントを設定すると、ストレージ URL を上書きされ、Glance イメージストレージに使用されます。  **注記** URL には、コンテナーまでのパスを含める必要がありますが、コンテナーそのものは除外します。オブジェクトの場所は、コンテナーおよびオブジェクトを設定した URL に追加すると得られます。 値: ● Swift コンテナーまでの有効な URL パスを表す文字列値 関連するオプション: ● なし

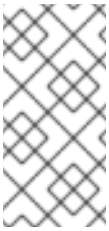
設定オプション = デフォルト値	型	説明
swift_store_endpoint_type = publicURL	文字列値	Swift サービスのエンドポイント種別。 この文字列値は、Swift エンドポイントの取得に使用するエンドポイント種別を示します。エンドポイント種別により、ユーザーが実行できるアクション (たとえば、ストアの読み取りおよび書き込み) が決定されます。この設定は、swift_store_auth_version が 1 よりも大きい場合にのみ使用されます。 値: ● publicURL ● adminURL ● internalURL 関連するオプション: ● swift_store_endpoint
swift_store_expire_soon_interval = 60	整数値	現在のトークンの有効期限が過ぎる前に新しいトークンが要求される可能性があるウィンドウのサイズを定義する時間 (秒単位)。 通常、Swift ストレージドライバーは、現在のトークンの有効期限が過ぎた時に新しいトークンを取得して、Swift へのアクセスを継続させます。ただし、処理中にトークンが期限切れになると、一部の Swift トランザクション (イメージのセグメントのアップロードなど) が適切に復元されないことがあります。 したがって、現在のトークンの有効期限前に新しいトークンを取得することにより、トランザクションを試行する前にトークンが失効したり期限切れに近づいたりしないようにします。デフォルトでは、Swift ストレージドライバーは、現在のトークンの有効期限の 60 秒前から新しいトークンを要求します。 値: ● ゼロ ● 正の整数値 関連するオプション: ● なし
swift_store_key = None	文字列値	Swift 認証サービスに対して認証するユーザーの認証鍵。

設定オプション = デフォルト値	型	説明
swift_store_large_object_chunk_size = 200	整数値	イメージデータがセグメント化される際のセグメントの最大サイズ (MB 単位)。 Swift クラスターによって適用される上限を上回るイメージをアップロードするためにイメージデータがセグメント化される場合、イメージデータはこの設定オプションで指定されるサイズ以下のセグメントに分割されます。詳細は、**swift_store_large_object_size** を参照してください。 たとえば、**swift_store_large_object_size** が 5 GB で、**swift_store_large_object_chunk_size** が 1 GB の場合、サイズが 6.2 GB のイメージは 7 セグメントに分割されます。この場合、最初の 6 つのセグメントのサイズは 1 GB で、7 番目のセグメントは 0.2 GB になります。 値: ● 該当する Swift クラスターによって適用される大型オブジェクト制限以下の正の整数。 関連するオプション: ● **swift_store_large_object_size**

設定オプション = デフォルト値	型	説明
swift_store_large_object_size = 5120	整数値	サイズしきい値 (MB 単位)。これを超えると、Glance はイメージデータのセグメント化を開始します。 Swift では、アップロードされる単一のオブジェクトのサイズに上限があります。デフォルトでは、これは 5 GB です。この制限を超えるオブジェクトをアップロードするには、オブジェクトをマニフェストファイルと関連付けられる複数の小さなオブジェクトにセグメント化します。詳細は、https://docs.openstack.org/swift/latest/overview_large_objects.htmlを参照してください。 この設定オプションは、サイズのしきい値を指定します。これを超えると、Swift ドライバーはイメージデータを複数の小さなファイルにセグメント化する処理を開始します。現在、Swift ドライバーは Dynamic Large Objects の作成のみをサポートしています。  **注記** これは、該当する Swift クラスターによって適用される大型オブジェクトの制限を考慮して設定する必要があります。 値: ● 該当する Swift クラスターによって適用される大型オブジェクト制限以下の正の整数。 関連するオプション: ● **swift_store_large_object_chunk_size**

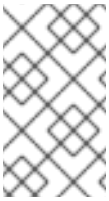
設定オプション = デフォルト値	型	説明
swift_store_multi_tenant = False	ブール値	イメージをテナントの Swift アカウントに保存します。 これにより、マルチテナントストレージモードが有効になり、Glance イメージがテナント固有の Swift アカウントに保存されます。これが無効になっていると、Glance はすべてのイメージを独自のアカウントに保存します。マルチテナントストアの詳細については、https://wiki.openstack.org/wiki/GlanceSwiftTenantSpecificStorageを参照してください。 **注記** マルチテナントの swift ストアを使用している場合は、**swift_store_config_file** オプションで swift 設定ファイルを設定しないようにしてください。 値: - True - False 関連するオプション: - swift_store_config_file

設定オプション = デフォルト値	型	説明
swift_store_multiple_containers_seed = 0	整数値	イメージの保存に使用するコンテナーの数を示すシード。 単一テナントストアを使用する場合は、イメージを 1 つ以上のコンテナーに保存できます。0 に設定すると、すべてのイメージが 1 つのコンテナーに保存されます。1 から 32 の整数値に設定すると、イメージの保存に複数のコンテナーが使用されます。この設定オプションにより、作成されるコンテナーの数が決まります。使用されるコンテナーの合計数は 16^N となるため、この設定オプションが 2 に設定されている場合は、イメージの保存に 16^2=256 のコンテナーが使用されます。 命名規則に関する詳細は、**swift_store_container** を参照してください。複数のコンテナーの使用についての詳細は、https://specs.openstack.org/openstack/glance-specs/specs/kilo/swift-store-multiple-containers.htmlを参照してください。 **注記** このオプションは、swift_store_multi_tenant が無効になっている場合にのみ使用されます。 値: - 32 以下の負の値ではない整数 関連するオプション: - **swift_store_container** - **swift_store_multi_tenant** - **swift_store_create_container_on_put**

設定オプション = デフォルト値	型	説明
swift_store_region = None	文字列値	Glance が使用する Swift エンドポイントのリージョン。 Glance がイメージの保存のために接続できる Swift リージョンを表す文字列値を指定します。デフォルトでは、リージョンは設定されません。 Glance がストレージバックエンドとして Swift を使用して複数のエンドポイントを持つ特定のテナントのイメージを保存する場合、**swift_store_region** で Swift リージョンを設定すると、Glance は単一リージョン接続ではなく、指定したリージョンの Swift に接続できるようになります。 このオプションは、単一テナントおよびマルチテナントストレージの両方に対して設定できます。  **注記** **swift_store_region** を使用したリージョンの設定はテナント固有のもので、テナントが異なるリージョンにまたがる複数のエンドポイントを持つ **場合にのみ** 必要です。 値: ● 有効な Swift リージョンを表す文字列値。 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_retry_get_count = 0	整数値	Swift のダウンロードがリトライされる回数。これを超えると、リクエストは失敗します。 エラーが発生する前にイメージのダウンロードをリトライする必要のある回数を表す整数値を指定します。デフォルト値はゼロです (イメージのダウンロードに失敗した場合、リトライされません)。正の整数値に設定すると、**swift_store_retry_get_count** により、ダウンロードの失敗時にこの回数ダウンロードが試みられます。これを超えると、エラーメッセージが送信されます。 値: ● ゼロ ● 正の整数値 関連するオプション: ● なし
swift_store_service_type = object-store	文字列値	使用する Swift サービスの種別。 Swift バックエンドストレージの使用中、イメージの保存に使用するサービス種別を表す文字列値を指定します。デフォルトのサービス種別は **object-store** に設定されます。 **注記** **swift_store_auth_version** が 2 に設定されている場合、この設定オプションの値は **object-store** である必要があります。より新しいバージョンの Keystone または別の認証スキームを使用している場合は、このオプションを変更できます。 値: ● Swift ストレージ用の有効なサービス種別を表す文字列。 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_ssl_compression = True	ブール値	HTTPS Swift リクエストの SSL レイヤーの圧縮。 SSL レイヤーでイメージの HTTPS Swift リクエストを圧縮するかどうかを決定するブール値を指定します。デフォルトでは、圧縮は有効になっています。 Swift を Glance イメージストレージのバックエンドストアとして使用する場合は、このオプションを使用して HTTPS Swift リクエストの SSL レイヤーの圧縮を設定できます。False に設定すると、HTTPS Swift リクエストの SSL レイヤーの圧縮は無効になります。このオプションを無効にすると、すでに圧縮された形式のイメージ (qcow2 など) に関するパフォーマンスが向上します。 値: ● True ● False 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_use_trusts = True	ブール値	マルチテナント Swift ストアの信頼を使用します。 マルチテナントストアが使用されている場合に、このオプションは、各追加/取得リクエストに対して信頼を作成するように Swift ストアに指示します。信頼を使用すると、Swift ストアは、データのアップロードまたはダウンロード中に、認証トークンの失効により発生する可能性がある問題を回避できます。 デフォルトでは、**swift_store_use_trusts** は **True**に設定されています (信頼の使用が有効)。**False** に設定すると、代わりに Swift 接続にユーザートークンが使用され、信頼の作成のオーバーヘッドがなくなります。  **注記** このオプションは、**swift_store_multi_tenant** が **True**に設定されている場合のみ考慮されます。 値: - True - False 関連するオプション: - swift_store_multi_tenant
swift_store_user = None	文字列値	Swift 認証サービスに対して認証するユーザー。

設定オプション = デフォルト値	型	説明
swift_upload_buffer_dir = None	文字列値	Swift にアップロードする前にイメージセグメントをバッファー処理するディレクトリー。 イメージセグメントが swift にアップロードされる前に一時的にバッファー処理される、glance ノード上のディレクトリーへの絶対パスを表す文字列値を指定します。 注記: ● これは、設定オプション **swift_buffer_on_upload** が True に設定されている場合にのみ必要です。 ● このディレクトリーは、**swift_store_large_object_chunk_size** と、特定の Glance ノードによって同時にアップロードできるイメージの最大数を念頭に置いてプロビジョニングする必要があります。 値: ● 絶対ディレクトリーパスを表す文字列値 関連するオプション: ● swift_buffer_on_upload ● swift_store_large_object_chunk_size

3.1.10. glance.store.vmware_datastore.store

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[glance.store.vmware_datastore.store]** グループで利用可能なオプションの概要をまとめたものです。

表3.9 glance.store.vmware_datastore.store

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
vmware_api_retry_count = 10	整数値	VMware API のリトライ回数。 この設定オプションは、接続関連の問題またはサーバー API 呼び出しのオーバーロード時に VMware ESX/VC サーバー API をリトライする回数を指定します。**retry forever** は指定できません。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし
vmware_ca_file = None	文字列値	CA バンドルファイルへの絶対パス。 この設定オプションにより、運用者はカスタム認証局ファイルを使用して ESX/vCenter 証明書を検証できます。 このオプションを設定すると、"vmware_insecure" オプションは無視され、指定された CA ファイルが ESX/vCenter サーバー証明書の認証に使用され、サーバーへのセキュアな接続が確立されます。 以下の値を使用できます。 ● CA ファイルへの有効な絶対パスである任意の文字列 関連するオプション: ● vmware_insecure

設定オプション = デフォルト値	型	説明
vmware_datastores = None	多値	イメージを保存できるデータストア。 この設定オプションは、VMWare ストアのバックエンドでイメージを保存できるデータストアを指定します。このオプションは、複数のデータストアを指定するために複数回指定できます。データストア名は、データセンターのパスの後に: で区切って指定する必要があります。オプションの重みをデータストア名の後に再度: で区切って指定して、優先順位を指定できます。したがって、必要な形式は <datacenter_path>:<datastore_name>: <optional_weight> になります。 イメージを追加する場合、イメージサイズが既知の場合に利用可能な空き領域が十分にない場合を除き、最も高い重みを持つデータストアが選択されます。重みが指定されない場合、これはゼロであると仮定され、ディレクトリーは最後に選択のために考慮されます。複数のデータストアの重みが同じ場合は、利用可能な空き領域が最も多いデータストアが選択されます。 以下の値を使用できます。 ● <datacenter_path>:<datastore_name>: <optional_weight>の形式の任意の文字列 関連するオプション:* なし

設定オプション = デフォルト値	型	説明
vmware_insecure = False	ブール値	ESX/vCenter サーバー証明書の検証を設定します。 この設定オプションはブール値を取り、ESX/vCenter サーバーの証明書を検証するかどうかを決定します。このオプションが True に設定されている場合、ESX/vCenter サーバー証明書は検証されません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 このオプションは、vmware_ca_file オプションが設定されている場合は無視されます。その場合、ESX/vCenter サーバー証明書は、"vmware_ca_file" オプションで指定したファイルを使用して検証されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● vmware_ca_file
vmware_server_host = None	ホストのアドレス値	ESX/ESXi または vCenter サーバーターゲットシステムのアドレス。 この設定オプションは、ESX/ESXi または vCenter サーバーターゲットシステムのアドレスを設定します。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。アドレスには、IP アドレス (127.0.0.1) または DNS 名 (www.my-domain.com) を含めることができます。 以下の値を使用できます。 ● 有効な IPv4 アドレスまたは IPv6 アドレス ● 有効な DNS 名 関連するオプション: ● vmware_server_username ● vmware_server_password

設定オプション = デフォルト値	型	説明
vmware_server_password = None	文字列値	サーバーのパスワード。 この設定オプションは、VMware ESX/ESXi または vCenter サーバーとの認証用のパスワードを取ります。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。 以下の値を使用できます。 ● "vmware_server_username" オプションを使用して指定されたユーザー名に対応するパスワードである任意の文字列 関連するオプション: ● vmware_server_host ● vmware_server_username
vmware_server_username = None	文字列値	サーバーのユーザー名。 この設定オプションは、VMware ESX/ESXi または vCenter サーバーとの認証用のユーザー名を取ります。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。 以下の値を使用できます。 ● 適切な権限を持つユーザーのユーザー名である任意の文字列 関連するオプション: ● vmware_server_host ● vmware_server_password

設定オプション = デフォルト値	型	説明
vmware_store_image_dir = /openstack_glance	文字列値	データストアで glance イメージが保存されるディレクトリー。 この設定オプションは、VMware データストアで glance イメージが保存されるディレクトリーへのパスを指定します。このオプションが設定されていない場合には、glance イメージが保存されるデフォルトのディレクトリーは openstack_glance になります。 以下の値を使用できます。 ● ディレクトリーへの有効なパスである任意の文字列 関連するオプション: ● なし
vmware_task_poll_interval = 5	整数値	VMware ESX/VC サーバーで呼び出されたリモートタスクをポーリングする間隔 (秒単位)。 この設定オプションは、VMWare ESX/VC サーバー API 呼び出しの一部として、処理中の非同期タスクをポーリングする際のスリープ時間 (秒単位) を取ります。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし

3.1.11. glance_store

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[glance_store]** グループで利用可能なオプションの概要をまとめたものです。

表3.10 glance_store

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
cinder_api_insecure = False	ブール値	cinder に対するセキュアではない SSL リクエストの実行を許可します。 このオプションが True に設定されている場合、HTTPS エンドポイント接続は **cinder_ca_certificates_file** オプションで指定された CA 証明書ファイルを使用して検証されます。 値: ● True ● False 関連するオプション: ● cinder_ca_certificates_file
cinder_ca_certificates_file = None	文字列値	cinder クライアントリクエストに使用する CA 証明書ファイルの場所。 CA 証明書ファイルが設定されている場合、指定されたファイルが、HTTPS エンドポイントを使用した cinder 接続の検証に使用されます。エンドポイントが HTTP の場合、この値は無視されます。検証を有効にするには、**cinder_api_insecure** を **True** に設定する必要があります。 値: ● CA 証明書ファイルへのパス 関連するオプション: ● cinder_api_insecure

設定オプション = デフォルト値	型	説明
cinder_catalog_info = volumev3::publicURL	文字列値	サービスカタログで cinder を検索する際に照合する情報。 **cinder_endpoint_template** が設定されておらず、**cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、**cinder_store_password** のいずれも設定されていない場合、cinder store はこの情報を使用して、現在のコンテキストでサービスカタログから cinder エンドポイントを検索します。**cinder_os_region_name** が設定されている場合は、適切なエンドポイントを取得するために考慮されます。 サービスカタログは、**openstack catalog list** コマンドでリスト表示できます。 値: ● 次の形式の文字列: **<service_type>:<service_name>:<interface>** 少なくとも **service_type** および **interface** を指定する必要があります。**service_name** は省略できます。 関連するオプション: ● cinder_os_region_name ● cinder_endpoint_template ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name ● cinder_store_password

設定オプション = デフォルト値	型	説明
cinder_endpoint_template = None	文字列値	cinder エンドポイントのテンプレートでサービスカタログの検索を上書きします。 このオプションを設定すると、サービスカタログから検索するのではなく、この値を使用して cinder エンドポイントが生成されます。**cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、および **cinder_store_password** が指定されている場合には、この値は無視されます。 この設定オプションが設定されている場合、**cinder_catalog_info** は無視されます。 値: ● cinder エンドポイントの URL テンプレート文字列。ここで、**%%(tenant)s** は現在のテナント (プロジェクト) 名に置き換えます。例:**http://cinder.openstack.example.org/v2/%%(tenant)s** 関連するオプション: ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name ● cinder_store_password ● cinder_catalog_info
cinder_enforce_multipath = False	ブール値	これが True に設定されている場合、multipathd が実行されていない場合、イメージ転送用のボリュームのアタッチが中断されます。それ以外の場合は、単一のパスにフォールバックします。 値: ● True か False 関連するオプション: ● cinder_use_multipath

設定オプション = デフォルト値	型	説明
cinder_http_retries = 3	整数値	失敗した http 呼び出しでの cinderclient のリトライ回数。 何らかのエラーによって呼び出しに失敗すると、cinderclient は数秒スリープした後に呼び出しを指定の回数までリトライします。 値: ● 正の整数 関連するオプション: ● なし
cinder_mount_point_base = /var/lib/glance/mnt	文字列値	glance ノードで NFS ボリュームがマウントされるディレクトリー。 値: ● マウントポイントの絶対パスを表す文字列。
cinder_os_region_name = None	文字列値	サービスカタログから cinder サービスを検索する際のリージョン名。 これは、**cinder_catalog_info** を使用してエンドポイントを決定する場合にのみ使用されます。これが設定されている場合、このノードによる cinder エンドポイントの検索は、指定のリージョンにフィルターされます。これは、カタログに複数のリージョンがリスト表示される場合に役立ちます。これが設定されていない場合、エンドポイントはすべてのリージョンから検索されます。 値: ● 有効なリージョン名である文字列。 関連するオプション: ● cinder_catalog_info

設定オプション = デフォルト値	型	説明
cinder_state_transition_timeout = 300	整数値	cinder ボリュームの遷移が完了するまで待機する時間 (秒単位)。 ボリュームデータの読み取り/書き込みを行うために、cinder ボリュームを作成、削除、または glance ノードにアタッチすると、ボリュームの状態が変更されます。たとえば、新規作成されたボリュームのステータスは、作成プロセスの完了後に **creating** から **available** に変わります。このオプションは、ステータス変更を待機する最大時間を指定します。待機時間がタイムアウトするか、ステータスが予期しない値に変更される場合 (**error**など)、イメージの作成は失敗します。 値: ● 正の整数 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
cinder_store_auth_address = None	文字列値	cinder 認証サービスがリッスンしているアドレス。 **cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、および **cinder_store_password**オプションのすべてが指定された場合、指定した値は常に認証に使用されます。これは、イメージサービスに固有のプロジェクト/テナントにイメージボリュームを保存することにより、ユーザーからイメージボリュームを隠すのに役立ちます。また、ユーザーは glance の ACL の制御下で、他のプロジェクト間でイメージボリュームを共有できます。 これらのオプションのいずれかが設定されていない場合、cinder エンドポイントはサービスカタログから検索され、現在のコンテキストのユーザーとプロジェクトが使用されます。 値: ● 有効な認証サービスアドレス (例:**http://openstack.example.org/identity/v2.0**)。 関連するオプション: ● cinder_store_user_name ● cinder_store_password ● cinder_store_project_name
cinder_store_password = None	文字列値	cinder に対して認証を行うユーザーのパスワード。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのユーザーが使用されます。 値: ● **cinder_store_user_name**で指定したユーザーの有効なパスワード 関連するオプション: ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name

設定オプション = デフォルト値	型	説明
cinder_store_project_name = None	文字列値	cinder にイメージボリュームが保存されるプロジェクト名。 この設定オプションが設定されていない場合は、現在のコンテキストのプロジェクトが使用されます。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのプロジェクトが使用されます。 値: ● 有効なプロジェクト名 関連するオプション: ● **cinder_store_auth_address** ● **cinder_store_user_name** ● **cinder_store_password**
cinder_store_user_name = None	文字列値	cinder に対して認証を行うためのユーザー名。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのユーザーが使用されます。 値: ● 有効なユーザー名 関連するオプション: ● cinder_store_auth_address ● cinder_store_password ● cinder_store_project_name

設定オプション = デフォルト値	型	説明
cinder_use_multipath = False	ブール値	デプロイメントでマルチパスがサポートされるかどうかを識別するためのフラグ。 マルチパスがサポートされない場合は、False に設定します。 値: ● True か False 関連するオプション: ● cinder_enforce_multipath
cinder_volume_type = None	文字列値	cinder でのボリュームの作成に使用されるボリューム種別。 一部の cinder バックエンドには、ストレージの使用を最適化するための複数のボリューム種別が含まれる場合があります。このオプションを追加すると、運用者はイメージに対して最適化できる特定のボリューム種別を cinder で選択することができます。 これが設定されていない場合には、cinder 設定で指定したデフォルトのボリューム種別がボリュームの作成に使用されます。 値: ● cinder からの有効なボリューム種別 関連するオプション: ● なし **注記** NFS バックエンドに関連付けられた暗号化された volume_type は使用できません。NFS バックエンドに保存されている暗号化されたボリュームは、glance_store がそのボリュームに保存されているイメージデータの書き込みまたはアクセスを試みるたびに、例外を発生させます。Cinder 管理者に相談して、適切な volume_type を決定してください。

設定オプション = デフォルト値	型	説明
default_backend = None	文字列値	データが保存されるデフォルトバックエンドのストア識別子。 値は、DEFAULT 設定グループの **enabled_backends** 設定オプションで定義される dict のキーのいずれかとして定義する必要があります。 このオプションに値が定義されていない場合は、以下のようになります。 ● 消費しているサービスは、起動を拒否することがあります。 ● 特定のバックエンドを指定しない store_add 呼び出しにより **glance_store.exceptions.UnknownScheme** 例外が発生します。 関連するオプション: ● enabled_backends

設定オプション = デフォルト値	型	説明
default_store = file	文字列値	イメージの保存に使用するデフォルトのスキーム。 イメージの保存に使用するデフォルトのスキームを表す文字列値を指定します。設定されていない場合、Glance は **file** をデフォルトのスキームとして使用して **file** ストアでイメージを保存します。 **注記** この設定オプションに指定する値は、**stores** 設定オプションで登録されたストアの有効なスキームである必要があります。 値: - file - Filesystem - http - https - swift - swift+http - swift+https - swift+config - rbd - cinder - vsphere - s3 関連するオプション: - stores **非推奨となったバージョン：** Rocky **理由:** このオプションは、''default_store'' 設定オプションと同様に機能する新しい設定オプション ''default_backend'' によって非推奨となりました。 このオプションは、U 開発サイクルで削除される予定です。

設定オプション = デフォルト値	型	説明
default_swift_reference = ref1	文字列値	デフォルトの Swift アカウント/バッキングストアパラメーターへの参照。 イメージストレージに swift アカウント/バッキングストアを使用するのに必要なデフォルトのパラメーターセットへの参照を表す文字列値を指定します。この設定オプションのデフォルト参照値は **ref1** です。この設定オプションはパラメーターを逆参照し、新しいイメージが追加されるたびに Swift ストレージバックエンドでのイメージ保管を容易にします。 値: ● 有効な文字列値 関連するオプション: ● なし
filesystem_store_chunk_size = 65536	整数値	チャンクサイズ (バイト単位)。 イメージファイルの読み取り/書き込み時に使用されるチャンクサイズ。この値を大きくするとスループットが向上しますが、大量のリクエストを処理する際にメモリー使用量が若干増加する場合もあります。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
filesystem_store_datadir = /var/lib/glance/images	文字列値	ファイルシステムのバックエンドストアがイメージを書き込むディレクトリー。 起動時に、Glance はディレクトリーが存在しない場合は作成し、**glance-api** が実行されるユーザーへの書き込みアクセスを検証します。書き込みアクセスが使用できない場合は、**BadStoreConfiguration** 例外が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 **注記** このディレクトリーは、ファイルシステムストアがストレージバックエンドとして使用される場合にのみ使用されます。**filesystem_store_datadir** または **filesystem_store_datadirs** オプションのいずれかを **glance-api.conf** で指定する必要があります。両方のオプションを指定すると、**BadStoreConfiguration** が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 値: ● ディレクトリーへの有効なパス 関連するオプション: ● **filesystem_store_datadirs** ● **filesystem_store_file_perm**

設定オプション = デフォルト値	型	説明
filesystem_store_datadirs = None	多値	ファイルシステムのバックエンドストアがイメージを書き込むディレクトリーとその優先順位のリスト。 **filesystem_store_datadir** 設定オプションで指定した単一のディレクトリーを使用する場合とは対照的に、ファイルシステムストアは複数のディレクトリーにイメージを格納するように設定できます。複数のディレクトリーを使用する場合、各ディレクトリーには、使用される優先順位を指定するオプションの優先度を指定できます。優先度は、ディレクトリーパスとコロンで組み合わされる整数で、値が大きいほど優先度が高くなります。2 つのディレクトリーの優先度が同じ場合は、空き容量が最も多いディレクトリーが使用されます。優先度を指定しないと、デフォルトでゼロに設定されます。 複数のストアディレクトリーを持つファイルシステムストアの設定に関する詳細は、https://docs.openstack.org/glance/latest/configuration/configuring.html を参照してください。 **注記** このディレクトリーは、ファイルシステムストアがストレージバックエンドとして使用される場合にのみ使用されます。**filesystem_store_datadir** または **filesystem_store_datadirs** オプションのいずれかを **glance-api.conf** で指定する必要があります。両方のオプションを指定すると、**BadStoreConfiguration** が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 値: - 以下の形式の文字列のリスト: - **<a valid directory path>:<optional integer priority>** 関連するオプション: - **filesystem_store_datadir** - **filesystem_store_file_perm**

設定オプション = デフォルト値	型	説明
filesystem_store_file_perm = 0	整数値	イメージファイルへのファイルアクセスパーミッション。 イメージデータへの目的のファイルアクセスパーミッションを設定します。これにより、Nova などの他のサービスがファイルシステムストアから直接イメージを使用できるようにすることができます。アクセスが付与されるサービスを実行中のユーザーは、作成されるファイルを所有しているグループのメンバーにすることができます。この設定オプションにゼロ以下の値を割り当てることは、デフォルトのパーミッションに変更が加えられないことを意味します。この値は、8 進数の数字としてデコードされます。 詳細は、https://docs.openstack.org/glance/latest/configuration/configuring.htmlのドキュメントを参照してください。 値: ● 有効なファイルアクセスパーミッション ● ゼロ ● 任意の負の整数 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
filesystem_store_metadata_file = None	文字列値	ファイルシステムストアのメタデータファイル。 ファイルシステムストアに関連付けられた場所と共に返されるメタデータが含まれるファイルへのパス。このオプションが設定されると、後で作成される新しいイメージにのみ使用されます。以前の既存のイメージには影響しません。 このファイルには、有効な JSON オブジェクトが含まれている必要があります。オブジェクトには、キー **ID** と **マウントポイント** が含まれている必要があります。両方のキーの値は文字列でなければなりません。 値: ● ストアメタデータファイルへの有効なパス 関連するオプション: ● なし
filesystem_thin_provisioning = False	ブール値	このバックエンドでシンプロビジョニングを有効にするかどうか。 この設定オプションを使用すると、ファイルシステムに null バイトシーケンスを実際に書き込まないという機能が有効になります。表示されるホールは、ファイルシステムによって自動的に null バイトとして解釈され、ストレージを実際に消費しません。この機能を有効にすると、ネットワーク経由で null バイトシーケンスが送信されないため、バックエンドのスペースを節約できるだけでなく、イメージのアップロードが高速化され、ネットワークトラフィックが節約されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
http_proxy_information = {}	dict 値	リモートサーバーへの接続に使用する http/https プロキシー情報。 この設定オプションは、リモートサーバーへの接続に使用する必要のある http/https プロキシー情報を指定します。プロキシー情報は、スキームとプロキシーのキー/値のペアにする必要があります (例:http:10.0.0.1:3128)。また、キーと値のペアをコンマで区切ることで、複数のスキームのプロキシーを指定することもできます (例: http:10.0.0.1:3128, https:10.0.0.1:1080)。 値: ● 上記のように スキーム: プロキシーペアのコンマ区切りリスト 関連するオプション: ● なし
https_ca_certificates_file = None	文字列値	CA バンドルファイルへのパス。 この設定オプションにより、運用者はカスタム認証局ファイルを使用してリモートサーバー証明書を検証できます。このオプションを設定すると、**https_insecure** オプションは無視され、指定された CA ファイルがサーバー証明書の認証に使用され、サーバーへのセキュアな接続が確立されます。 値: ● CA ファイルへの有効なパス 関連するオプション: ● https_insecure

設定オプション = デフォルト値	型	説明
https_insecure = True	ブール値	リモートサーバー証明書の検証を設定します。 この設定オプションはブール値を取り、リモートサーバーの証明書を検証するかどうかを決定します。True に設定すると、リモートサーバー証明書は検証されません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 このオプションは、**https_ca_certificates_file** が設定されている場合は無視されます。リモートサーバー証明書は、**https_ca_certificates_file** オプションを使用して指定されたファイルを使用して検証されます。 値: ● True ● False 関連するオプション: ● https_ca_certificates_file
rados_connect_timeout = 0	整数値	Ceph クラスターに接続する際のタイムアウト値。 この設定オプションは、Ceph クラスターに接続する際に使用されるタイムアウト値 (秒単位) を取ります。つまり、接続を閉じるまで glance-api が待機する時間を設定します。これにより、RBD への接続中に glance-api がハングアップするのを防ぐことができます。このオプションの値を 0 以下に設定すると、タイムアウトは設定されず、デフォルトの librados 値が使用されます。 以下の値を使用できます。 ● 任意の整数値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
`rbd_store_ceph_conf = `	文字列値	Ceph 設定ファイルへのパス。 この設定オプションは、使用する Ceph 設定ファイルへのパスを指定します。このオプションの値がユーザーにより設定されていない場合や、空の文字列に設定されている場合、librados は、デフォルトの Ceph 設定ファイルの場所を順に検索して標準の ceph.conf ファイルを読み取ります。詳細は、Ceph のドキュメントを参照してください。 **注記** Cephx 認証を使用する場合、このファイルの client.<USER>セクションに適切なキーリングへの参照を含める必要があります。 注 2: このオプションを空白のままにすると (デフォルト)、使用される実際の Ceph 設定ファイルは、使用されている librados のバージョンによって変わる可能性があります。有効になっている設定ファイルを正確に把握しておくことが重要な場合は、このオプションを使用してここにファイルを指定できます。 以下の値を使用できます。 ● 設定ファイルへの有効なパス 関連するオプション: ● rbd_store_user

設定オプション = デフォルト値	型	説明
rbd_store_chunk_size = 8	整数値	RADOS イメージをチャンクに分割する際のサイズ (メガバイト単位)。 Glance イメージをチャンクに分割する際のサイズ (メガバイト単位) を表す整数値を指定します。デフォルトのチャンクサイズは 8 メガバイトです。最適なパフォーマンスを得るには、値は 2 の累乗でなければなりません。 Ceph の RBD オブジェクトストレージシステムを Glance イメージを保存するためのストレージバックエンドとして使用する場合、イメージはこのオプションを使用して設定されるサイズのオブジェクトに分割されます。これらの分割されたオブジェクトは、分散ブロックデータストア全体に保存され、Glance に使用されます。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし
rbd_store_pool = images	文字列値	イメージが保存される RADOS プール。 RBD を Glance イメージを保存するためのストレージバックエンドとして使用する場合、イメージは **プール** へのオブジェクト (イメージのチャンク) の論理的なグループ化により保存されます。各プールは、含めることのできる配置グループの数で定義されます。使用されるデフォルトのプールは **イメージ** です。 RBD ストレージバックエンドの詳細は、http://ceph.com/planet/how-data-is-stored-in-ceph-cluster/を参照してください。 以下の値を使用できます。 ● 有効なプール名 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
rbd_store_user = None	文字列値	認証する RADOS ユーザー。 この設定オプションは、認証する RADOS ユーザーを設定します。このオプションは RADOS 認証が有効になっている場合にのみ必要で、ユーザーが Cephx 認証を使用している場合にのみ適用されます。このオプションの値がユーザーによって設定されていない場合や、None に設定されている場合には、デフォルト値が選択されます。これは、rbd_store_ceph_conf の client. セクションに基づきます。 以下の値を使用できます。 ● 有効な RADOS ユーザー 関連するオプション: ● rbd_store_ceph_conf
rbd_thin_provisioning = False	ブール値	このバックエンドでシンプロビジョニングを有効にするかどうか。 この設定オプションを使用すると、RBD バックエンドに null バイトシーケンスを実際に書き込まないという機能が有効になります。表示されるホールは、Ceph によって自動的に null バイトとして解釈され、ストレージを実際に消費しません。この機能を有効にすると、ネットワーク経由で null バイトシーケンスが送信されないため、バックエンドのスペースを節約できるだけでなく、イメージのアップロードが高速化され、ネットワークトラフィックが節約されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
rootwrap_config = /etc/glance/rootwrap.conf	文字列値	root としてコマンドを実行するのに使用する rootwrap 設定ファイルへのパス。 cinder ストアでは、イメージボリュームを操作するのに root 権限が必要です (iSCSI/FC ボリュームへの接続、ボリュームデータの読み取り/書き込みのためなど)。設定ファイルでは、cinder ストアおよび os-brick ライブラリーによる必要なコマンドを許可する必要があります。 値: ● rootwrap 設定ファイルへのパス 関連するオプション: ● なし
s3_store_access_key = None	文字列値	S3 クエリートークンアクセスキー。 この設定オプションは、Amazon S3 または S3 互換ストレージサーバーで認証するためのアクセスキーを取得します。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。 値: ● 適切な権限を持つユーザーのアクセスキーである任意の文字列値 関連するオプション: ● s3_store_host ● s3_store_secret_key

設定オプション = デフォルト値	型	説明
s3_store_bucket = None	文字列値	Glance データの保存に使用される S3 バケット。 この設定オプションは、Glance イメージが S3 に保存される場所を指定します。**s3_store_create_bucket_on_put** を true に設定すると、バケットが存在しなくても自動的に作成されます。 値: ● 任意の文字列値 関連するオプション: ● s3_store_create_bucket_on_put ● s3_store_bucket_url_format
s3_store_bucket_url_format = auto	文字列値	オブジェクトを決定するために使用される S3 呼び出し形式。 この設定オプションは、S3 バケット内のオブジェクトのアドレスを指定するために使用されるアクセスモデルを取ります。 注記: **path** スタイルでは、オブジェクトのエンドポイントは https://s3.amazonaws.com/bucket/example.img のようになります。**virtual** スタイルでは、オブジェクトのエンドポイントは https://bucket.s3.amazonaws.com/example.img のようになります。バケット名で DNS 命名規則に従わない場合、パススタイルでオブジェクトを取得できますが、仮想スタイルでは取得できません。 値: ● **auto**、**virtual**、または **path** の任意の文字列値 関連するオプション: ● s3_store_bucket

設定オプション = デフォルト値	型	説明
s3_store_create_bucket_on_put = False	ブール値	S3 が新しいバケットを作成する必要があるかどうかを決定します。 この設定オプションはブール値を取り、Glance が S3 に新しいバケットを作成する必要があるかどうかを示します (存在しない場合)。 値: ● 任意のブール値 関連するオプション: ● なし
s3_store_host = None	文字列値	S3 サーバーがリッスンしているホスト。 この設定オプションは、S3 または S3 互換ストレージサーバーのホストを設定します。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。ホストには、DNS 名 (s3.amazonaws.com、my-object-storage.com など) または IP アドレス (127.0.0.1) を含めることができます。 値: ● 有効な DNS 名 ● 有効な IPv4 アドレス 関連するオプション: ● s3_store_access_key ● s3_store_secret_key

設定オプション = デフォルト値	型	説明
s3_store_large_object_chunk_size = 10	整数値	パーツのアップロード時に S3 が使用するマルチパートアップロードパーツのサイズ (MB 単位)。 この設定オプションは、マルチパートアップロードのイメージ分割サイズを MB 単位で取得します。 注記: 分割できるイメージは 10,000 個までです。 値: ● 任意の正の整数値 (5M 以上である必要があります) 関連するオプション: ● s3_store_large_object_size ● s3_store_thread_pools
s3_store_large_object_size = 100	整数値	S3 がイメージファイルのチャンク化を開始し、S3 でマルチパートアップロードを実行する必要があるサイズ (MB 単位)。 この設定オプションは、イメージをそのまま S3 にアップロードするか、分割するか (マルチパートアップロード) を決定するために MB 単位でしきい値を取得します。 注記: 分割できるイメージは 10,000 個までです。 値: ● 任意の正の整数値 関連するオプション: ● s3_store_large_object_chunk_size ● s3_store_thread_pools

設定オプション = デフォルト値	型	説明
s3_store_secret_key = None	文字列値	S3 クエリートークンの秘密鍵。 この設定オプションは、Amazon S3 または S3 互換ストレージサーバーで認証するための秘密鍵を受け取ります。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。 値: ● **s3_store_host** オプションを使用して指定されたアクセスキーに対応する秘密鍵である任意の文字列値 関連するオプション: ● s3_store_host ● s3_store_access_key
s3_store_thread_pools = 10	整数値	S3 でマルチパートアップロードを実行するためのスレッドプールの数。 この設定オプションは、マルチパートアップロードの実行時にスレッドプールの数を取得します。 値: ● 任意の正の整数値 関連するオプション: ● s3_store_large_object_size ● s3_store_large_object_chunk_size

設定オプション = デフォルト値	型	説明
stores = ['file', 'http']	リスト値	有効な Glance ストアのリスト。 ディスクイメージを格納するために使用するストレージバックエンドを、コンマ区切りリストとして登録します。Glance でのディスクイメージの保存用に有効化されたデフォルトのストアは、**file** と **http** です。 値: ● 以下が含まれるコンマ区切りリスト: ○ file ○ http ○ swift ○ rbd ○ cinder ○ vmware ○ s3 関連するオプション: ● default_store **非推奨となったバージョン：** Rocky **理由:** このオプションは、異なるスキームの複数のバックエンドストアを設定するのに役立つ新たな設定オプション ''enabled_backends'' によって非推奨となりました。 このオプションは、U 開発サイクルで削除される予定です。

設定オプション = デフォルト値	型	説明
swift_buffer_on_upload = False	ブール値	Swift にアップロードする前にイメージのセグメントをバッファー処理します。 swift へのアップロード中に Glance がイメージデータをディスクにバッファーするかどうかを指定するブール値を指定します。これにより、Glance はエラー時にアップロードを再開できます。 注意: このオプションを有効にすると、API ノードのディスク使用量が増加するので、細心の注意を払う必要があります。ファイルシステムの設定状況によっては、バッファー処理に使用されるディスク領域により、glance イメージキャッシュに利用可能な実際のディスク容量が減る可能性があることに注意してください。ディスク使用率は、式**swift_store_large_object_chunk_size** * **workers** * 1000 に従って上限が設定されます。 値: ● True ● False 関連するオプション: ● swift_upload_buffer_dir
swift_store_admin_tenants = []	リスト値	管理者アクセスが付与されるテナントのリスト。 これは、マルチテナントモードの Glance によって作成されたすべての Swift コンテナーでの読み取り/書き込みアクセスが付与されるテナントのリストです。デフォルト値は空のリストです。 値: ● Keystone プロジェクト/テナントの UUID を表す文字列のコンマ区切りリスト 関連するオプション: ● なし
swift_store_auth_address = None	文字列値	Swift 認証サービスがリッスンしているアドレス。

設定オプション = デフォルト値	型	説明
swift_store_auth_insecure = False	ブール値	サーバー証明書の検証を設定します。 このブール値は、サーバー証明書を検証するかどうかを決定します。このオプションが True に設定されている場合、swiftclient は認証時に有効な SSL 証明書かどうかをチェックしません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 値: ● True ● False 関連するオプション: ● swift_store_cacert
swift_store_auth_version = 2	文字列値	使用する認証サービスのバージョン。有効なバージョンは、keystone の場合は 2 および 3 で、swauth および rackspace の場合は 1(非推奨) です。
swift_store_cacert = None	文字列値	CA バンドルファイルへのパス。 この設定オプションにより、運用者は Swift に接続する際の SSL 検証用のカスタム認証局ファイルへのパスを指定できます。 値: ● CA ファイルへの有効なパス 関連するオプション: ● swift_store_auth_insecure

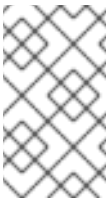
設定オプション = デフォルト値	型	説明
swift_store_config_file = None	文字列値	swift アカウント設定が含まれるファイルへの絶対パス。 設定済みの各 Swift アカウント/バッキングストアへの参照を持つ設定ファイルへのパスを表す文字列値を含めます。デフォルトでは、ファイルパスが指定されておらず、カスタマイズされた Swift 参照は無効になっています。データベースへの認証情報の保存を回避するため、このオプションを設定することは、イメージストレージに Swift ストレージバックエンドを使用する際に強く推奨されます。 **注記** **swift_store_multi_tenant** を **True** に設定した場合は、このオプションを設定しないでください。 値: ● glance-api ノードの絶対パスを表す文字列値 関連するオプション: ● swift_store_multi_tenant

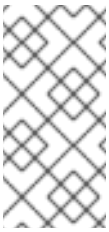
設定オプション = デフォルト値	型	説明
swift_store_container = glance	文字列値	イメージを保存するための単一のコンテナーの名前/複数のコンテナーの名前の接頭辞 1 つのコンテナーを使用してイメージを保存する場合、この設定オプションは、すべてのイメージの保存に使用される Glance アカウント内のコンテナーを示します。複数のコンテナーがイメージの保存に使用される場合、これはすべてのコンテナーの名前の接頭辞になります。単一/複数コンテナーの使用は、設定オプション **swift_store_multiple_containers_seed** を使用して制御できます。 複数のコンテナーを使用する場合、コンテナーの名前は、この設定オプションに設定された値をベースに、イメージ UUID の最初の N 文字をアンダースコアで区切られた接尾辞として指定されます (N は **swift_store_multiple_containers_seed**によって指定されます)。 例: シードが 3 で swift_store_container = **glance** と設定されている場合には、UUID が**fdae39a1-bac5-4238-aba4-69bcc726e848** のイメージは **glance_fda** のコンテナーに配置されます。コンテナー名の作成時に、UUID のすべてのダッシュは含まれますが、文字の制限にはカウントされないため、N=10 の場合はコンテナー名は **glance_fdae39a1-ba** になります。 値: ● 単一のコンテナーを使用する場合、この設定オプションは Glance の Swift アカウントの有効な任意の swift コンテナーの名前の文字列にすることができます。 ● 複数のコンテナーを使用する場合、この設定オプションは、 Swift により適用されるコンテナーの命名ルールを満たす限り任意の文字列になります。**swift_store_multiple_containers_seed** の値も考慮する必要があります。 関連するオプション: ● **swift_store_multiple_containers_seed** ● **swift_store_multi_tenant** ● **swift_store_create_container_on_put**

設定オプション = デフォルト値	型	説明
swift_store_create_container_on_put = False	ブール値	イメージのアップロード時に、コンテナーを作成します (まだ存在しない場合)。 イメージのアップロード時に、対応するコンテナーが存在しない場合は、この設定オプションが True に設定されている場合に作成されます。デフォルトでは、コンテナーは作成されません。この動作は、単一および複数コンテナーモードの両方に適用されます。 値: ● True ● False 関連するオプション: ● なし
swift_store_endpoint = None	文字列値	Swift バックエンドストレージに使用する URL エンドポイント。 Glance イメージを Swift ストアに格納するために使用する URL エンドポイントを表す文字列値を指定します。デフォルトでは、エンドポイントは設定されず、**auth** によって返されるストレージ URL が 使用されます。**swift_store_endpoint** を使用してエンドポイントを設定すると、ストレージ URL を上書きされ、Glance イメージストレージに使用されます。 **注記** URL には、コンテナーまでのパスを含める必要がありますが、コンテナーそのものは除外します。オブジェクトの場所は、コンテナーおよびオブジェクトを設定した URL に追加すると得られます。 値: ● Swift コンテナーまでの有効な URL パスを表す文字列値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_endpoint_type = publicURL	文字列値	Swift サービスのエンドポイント種別。 この文字列値は、Swift エンドポイントの取得に使用するエンドポイント種別を示します。エンドポイント種別により、ユーザーが実行できるアクション (たとえば、ストアの読み取りおよび書き込み) が決定されます。この設定は、swift_store_auth_version が 1 よりも大きい場合にのみ使用されます。 値: ● publicURL ● adminURL ● internalURL 関連するオプション: ● swift_store_endpoint
swift_store_expire_soon_interval = 60	整数値	現在のトークンの有効期限が過ぎる前に新しいトークンが要求される可能性があるウィンドウのサイズを定義する時間 (秒単位)。 通常、Swift ストレージドライバーは、現在のトークンの有効期限が過ぎた時に新しいトークンを取得して、Swift へのアクセスを継続させます。ただし、処理中にトークンが期限切れになると、一部の Swift トランザクション (イメージのセグメントのアップロードなど) が適切に復元されないことがあります。 したがって、現在のトークンの有効期限前に新しいトークンを取得することにより、トランザクションを試行する前にトークンが失効したり期限切れに近づいたりしないようにします。デフォルトでは、Swift ストレージドライバーは、現在のトークンの有効期限の 60 秒前から新しいトークンを要求します。 値: ● ゼロ ● 正の整数値 関連するオプション: ● なし
swift_store_key = None	文字列値	Swift 認証サービスに対して認証するユーザーの認証鍵。

設定オプション = デフォルト値	型	説明
swift_store_large_object_chunk_size = 200	整数値	イメージデータがセグメント化される際のセグメントの最大サイズ (MB 単位)。 Swift クラスターによって適用される上限を上回るイメージをアップロードするためにイメージデータがセグメント化される場合、イメージデータはこの設定オプションで指定されるサイズ以下のセグメントに分割されます。詳細は、**swift_store_large_object_size** を参照してください。 たとえば、**swift_store_large_object_size** が 5 GB で、**swift_store_large_object_chunk_size** が 1 GB の場合、サイズが 6.2 GB のイメージは 7 セグメントに分割されます。この場合、最初の 6 つのセグメントのサイズは 1 GB で、7 番目のセグメントは 0.2 GB になります。 値: ● 該当する Swift クラスターによって適用される大型オブジェクト制限以下の正の整数。 関連するオプション: ● **swift_store_large_object_size**

設定オプション = デフォルト値	型	説明
swift_store_large_object_size = 5120	整数値	サイズしきい値 (MB 単位)。これを超えると、Glance はイメージデータのセグメント化を開始します。 Swift では、アップロードされる単一のオブジェクトのサイズに上限があります。デフォルトでは、これは 5 GB です。この制限を超えるオブジェクトをアップロードするには、オブジェクトをマニフェストファイルと関連付けられる複数の小さなオブジェクトにセグメント化します。詳細は、https://docs.openstack.org/swift/latest/overview_large_objects.htmlを参照してください。 この設定オプションは、サイズのしきい値を指定します。これを超えると、Swift ドライバーはイメージデータを複数の小さなファイルにセグメント化する処理を開始します。現在、Swift ドライバーは Dynamic Large Objects の作成のみをサポートしています。  **注記** これは、該当する Swift クラスターによって適用される大型オブジェクトの制限を考慮して設定する必要があります。 値: ● 該当する Swift クラスターによって適用される大型オブジェクト制限以下の正の整数。 関連するオプション: ● **swift_store_large_object_chunk_size**

設定オプション = デフォルト値	型	説明
swift_store_multi_tenant = False	ブール値	イメージをテナントの Swift アカウントに保存します。 これにより、マルチテナントストレージモードが有効になり、Glance イメージがテナント固有の Swift アカウントに保存されます。これが無効になっていると、Glance はすべてのイメージを独自のアカウントに保存します。マルチテナントストアの詳細については、https://wiki.openstack.org/wiki/GlanceSwiftTenantSpecificStorageを参照してください。  **注記** マルチテナントの swift ストアを使用している場合は、**swift_store_config_file** オプションで swift 設定ファイルを設定しないようにしてください。 値: - True - False 関連するオプション: - swift_store_config_file

設定オプション = デフォルト値	型	説明
swift_store_multiple_containers_seed = 0	整数値	イメージの保存に使用するコンテナーの数を示すシード。 単一テナントストアを使用する場合は、イメージを 1 つ以上のコンテナーに保存できます。0 に設定すると、すべてのイメージが 1 つのコンテナーに保存されます。1 から 32 の整数値に設定すると、イメージの保存に複数のコンテナーが使用されます。この設定オプションにより、作成されるコンテナーの数が決まります。使用されるコンテナーの合計数は 16^N となるため、この設定オプションが 2 に設定されている場合は、イメージの保存に 16^2=256 のコンテナーが使用されます。 命名規則に関する詳細は、**swift_store_container** を参照してください。複数のコンテナーの使用についての詳細は、https://specs.openstack.org/openstack/glance-specs/specs/kilo/swift-store-multiple-containers.htmlを参照してください。 **注記** このオプションは、swift_store_multi_tenant が無効になっている場合にのみ使用されます。 値: ● 32 以下の負の値ではない整数 関連するオプション: ● **swift_store_container** ● **swift_store_multi_tenant** ● **swift_store_create_container_on_put**

設定オプション = デフォルト値	型	説明
swift_store_region = None	文字列値	Glance が使用する Swift エンドポイントのリージョン。 Glance がイメージの保存のために接続できる Swift リージョンを表す文字列値を指定します。デフォルトでは、リージョンは設定されません。 Glance がストレージバックエンドとして Swift を使用して複数のエンドポイントを持つ特定のテナントのイメージを保存する場合、**swift_store_region** で Swift リージョンを設定すると、Glance は単一リージョン接続ではなく、指定したリージョンの Swift に接続できるようになります。 このオプションは、単一テナントおよびマルチテナントストレージの両方に対して設定できます。 **注記** **swift_store_region** を使用したリージョンの設定はテナント固有のもので、テナントが異なるリージョンにまたがる複数のエンドポイントを持つ **場合にのみ** 必要です。 値: ● 有効な Swift リージョンを表す文字列値。 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_retry_get_count = 0	整数値	Swift のダウンロードがリトライされる回数。これを超えると、リクエストは失敗します。 エラーが発生する前にイメージのダウンロードをリトライする必要のある回数を表す整数値を指定します。デフォルト値はゼロです (イメージのダウンロードに失敗した場合、リトライされません)。正の整数値に設定すると、**swift_store_retry_get_count** により、ダウンロードの失敗時にこの回数ダウンロードが試みられます。これを超えると、エラーメッセージが送信されます。 値: ● ゼロ ● 正の整数値 関連するオプション: ● なし
swift_store_service_type = object-store	文字列値	使用する Swift サービスの種別。 Swift バックエンドストレージの使用中、イメージの保存に使用するサービス種別を表す文字列値を指定します。デフォルトのサービス種別は **object-store** に設定されます。 **注記** **swift_store_auth_version** が 2 に設定されている場合、この設定オプションの値は **object-store** である必要があります。より新しいバージョンの Keystone または別の認証スキームを使用している場合は、このオプションを変更できます。 値: ● Swift ストレージ用の有効なサービス種別を表す文字列。 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_ssl_compression = True	ブール値	HTTPS Swift リクエストの SSL レイヤーの圧縮。 SSL レイヤーでイメージの HTTPS Swift リクエストを圧縮するかどうかを決定するブール値を指定します。デフォルトでは、圧縮は有効になっています。 Swift を Glance イメージストレージのバックエンドストアとして使用する場合は、このオプションを使用して HTTPS Swift リクエストの SSL レイヤーの圧縮を設定できます。False に設定すると、HTTPS Swift リクエストの SSL レイヤーの圧縮は無効になります。このオプションを無効にすると、すでに圧縮された形式のイメージ (qcow2 など) に関するパフォーマンスが向上します。 値: ● True ● False 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_use_trusts = True	ブール値	マルチテナント Swift ストアの信頼を使用します。 マルチテナントストアが使用されている場合に、このオプションは、各追加/取得リクエストに対して信頼を作成するように Swift ストアに指示します。信頼を使用すると、Swift ストアは、データのアップロードまたはダウンロード中に、認証トークンの失効により発生する可能性がある問題を回避できます。 デフォルトでは、**swift_store_use_trusts** は **True**に設定されています (信頼の使用が有効)。**False** に設定すると、代わりに Swift 接続にユーザートークンが使用され、信頼の作成のオーバーヘッドがなくなります。 **注記** このオプションは、**swift_store_multi_tenant** が **True**に設定されている場合のみ考慮されます。 値: - True - False 関連するオプション: - swift_store_multi_tenant
swift_store_user = None	文字列値	Swift 認証サービスに対して認証するユーザー。

設定オプション = デフォルト値	型	説明
swift_upload_buffer_dir = None	文字列値	Swift にアップロードする前にイメージセグメントをバッファー処理するディレクトリー。 イメージセグメントが swift にアップロードされる前に一時的にバッファー処理される、glance ノード上のディレクトリーへの絶対パスを表す文字列値を指定します。 注記: ● これは、設定オプション **swift_buffer_on_upload** が True に設定されている場合にのみ必要です。 ● このディレクトリーは、**swift_store_large_object_chunk_size** と、特定の Glance ノードによって同時にアップロードできるイメージの最大数を念頭に置いてプロビジョニングする必要があります。 値: ● 絶対ディレクトリーパスを表す文字列値 関連するオプション: ● swift_buffer_on_upload ● swift_store_large_object_chunk_size
vmware_api_retry_count = 10	整数値	VMware API のリトライ回数。 この設定オプションは、接続関連の問題またはサーバー API 呼び出しのオーバーロード時に VMware ESX/VC サーバー API をリトライする回数を指定します。**retry forever** は指定できません。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
vmware_ca_file = None	文字列値	CA バンドルファイルへの絶対パス。 この設定オプションにより、運用者はカスタム認証局ファイルを使用して ESX/vCenter 証明書を検証できます。 このオプションを設定すると、"vmware_insecure" オプションは無視され、指定された CA ファイルが ESX/vCenter サーバー証明書の認証に使用され、サーバーへのセキュアな接続が確立されます。 以下の値を使用できます。 ● CA ファイルへの有効な絶対パスである任意の文字列 関連するオプション: ● vmware_insecure
vmware_datastores = None	多値	イメージを保存できるデータストア。 この設定オプションは、VMWare ストアのバックエンドでイメージを保存できるデータストアを指定します。このオプションは、複数のデータストアを指定するために複数回指定できます。データストア名は、データセンターのパスの後に: で区切って指定する必要があります。オプションの重みをデータストア名の後に再度: で区切って指定して、優先順位を指定できます。したがって、必要な形式は <datacenter_path>:<datastore_name>:<optional_weight> になります。 イメージを追加する場合、イメージサイズが既知の場合に利用可能な空き領域が十分にない場合を除き、最も高い重みを持つデータストアが選択されます。重みが指定されない場合、これはゼロであると仮定され、ディレクトリーは最後に選択のために考慮されます。複数のデータストアの重みが同じ場合は、利用可能な空き領域が最も多いデータストアが選択されます。 以下の値を使用できます。 ● <datacenter_path>:<datastore_name>:<optional_weight>の形式の任意の文字列 関連するオプション:* なし

設定オプション = デフォルト値	型	説明
vmware_insecure = False	ブール値	ESX/vCenter サーバー証明書の検証を設定します。 この設定オプションはブール値を取り、ESX/vCenter サーバーの証明書を検証するかどうかを決定します。このオプションが True に設定されている場合、ESX/vCenter サーバー証明書は検証されません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 このオプションは、vmware_ca_file オプションが設定されている場合は無視されます。その場合、ESX/vCenter サーバー証明書は、"vmware_ca_file" オプションで指定したファイルを使用して検証されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● vmware_ca_file
vmware_server_host = None	ホストのアドレス値	ESX/ESXi または vCenter サーバーターゲットシステムのアドレス。 この設定オプションは、ESX/ESXi または vCenter サーバーターゲットシステムのアドレスを設定します。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。アドレスには、IP アドレス (127.0.0.1) または DNS 名 (www.my-domain.com) を含めることができます。 以下の値を使用できます。 ● 有効な IPv4 アドレスまたは IPv6 アドレス ● 有効な DNS 名 関連するオプション: ● vmware_server_username ● vmware_server_password

設定オプション = デフォルト値	型	説明
vmware_server_password = None	文字列値	サーバーのパスワード。 この設定オプションは、VMware ESX/ESXi または vCenter サーバーとの認証用のパスワードを取ります。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。 以下の値を使用できます。 ● "vmware_server_username" オプションを使用して指定されたユーザー名に対応するパスワードである任意の文字列 関連するオプション: ● vmware_server_host ● vmware_server_username
vmware_server_username = None	文字列値	サーバーのユーザー名。 この設定オプションは、VMware ESX/ESXi または vCenter サーバーとの認証用のユーザー名を取ります。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。 以下の値を使用できます。 ● 適切な権限を持つユーザーのユーザー名である任意の文字列 関連するオプション: ● vmware_server_host ● vmware_server_password

設定オプション = デフォルト値	型	説明
vmware_store_image_dir = /openstack_glance	文字列値	データストアで glance イメージが保存されるディレクトリー。 この設定オプションは、VMware データストアで glance イメージが保存されるディレクトリーへのパスを指定します。このオプションが設定されていない場合には、glance イメージが保存されるデフォルトのディレクトリーは openstack_glance になります。 以下の値を使用できます。 ● ディレクトリーへの有効なパスである任意の文字列 関連するオプション: ● なし
vmware_task_poll_interval = 5	整数値	VMware ESX/VC サーバーで呼び出されたリモートタスクをポーリングする間隔 (秒単位)。 この設定オプションは、VMWare ESX/VC サーバー API 呼び出しの一部として、処理中の非同期タスクをポーリングする際のスリープ時間 (秒単位) を取ります。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし

3.1.12. image_format

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[image_format]** グループで利用可能なオプションの概要をまとめたものです。

表3.11 image_format

設定オプション = デフォルト値	型	説明
container_formats = ['ami', 'ari', 'aki', 'bare', 'ovf', 'ova', 'docker', 'compressed']	リスト値	container_format イメージ属性でサポートされる値

設定オプション = デフォルト値	型	説明
disk_formats = ['ami', 'ari', 'aki', 'vhd', 'vhdx', 'vmdk', 'raw', 'qcow2', 'vdi', 'iso', 'ploop']	リスト値	disk_format イメージ属性でサポートされる値
vmdk_allowed_types = ['streamOptimized', 'monolithicSparse']	リスト値	許可される VMDK create-type サブフォーマットを説明する文字列のリスト。名前付きエクステントの処理によるホストファイルの潜在的な露出を避けるために、スパースヘッダーを持つ単一ファイルのバリアントのみを含めることを推奨します。このリストが空の場合、VDMK イメージタイプは許可されません。これは現在、イメージの変換中にのみチェックされ (有効な場合)、変換元の VMDK イメージのタイプが制限されることに注意してください。

3.1.13. keystone_authtoken

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[keystone_authtoken]** グループ下で使用できるオプションの概要をまとめたものです。

表3.12 keystone_authtoken

設定オプション = デフォルト値	型	説明
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ

設定オプション = デフォルト値	型	説明
auth_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。このオプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。**非推奨:**Queens 以降 *理由:* auth_uri オプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。
auth_version = None	文字列値	Identity API エンドポイントの API バージョン。
cache = None	文字列値	Swift キャッシュオブジェクトが保存される環境キーを要求します。auth_token ミドルウェアを Swift キャッシュと共にデプロイする場合は、このオプションを使用して、ミドルウェアが swift とキャッシングバックエンドを共有するようにします。それ以外の場合は、代わりに **memcached_servers** オプションを使用します。
cafile = None	文字列値	HTTPs 接続の検証時に使用する PEM でエンコードされた認証局。デフォルトはシステム CA です。
certfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
delay_auth_decision = False	ブール値	ミドルウェア内の認可要求を処理せず、承認の決定をダウンストリームの WSGI コンポーネントに委譲します。

設定オプション = デフォルト値	型	説明
enforce_token_bind = permissive	文字列値	トークンバインディングの使用および種別を制御するために使用されます。トークンバインディングのチェックを無効するには、"disabled"に設定します。バインドタイプがサーバーの認識する形式の場合にはバインディング情報を検証し、そうでない場合には無視するには、"permissive"(デフォルト) に設定します。"strict"は"permissive"と類似していますが、バインドタイプが不明な場合にはトークンが拒否されます。"required"の場合は、いずれかの形式のトークンバインディングが必要です。最後に、トークンに指定する必要のあるバインディングメソッドの名前。
http_connect_timeout = None	整数値	Identity API サーバーと通信する際の要求タイムアウト値。
http_request_max_retries = 3	整数値	Identity API サーバーと通信する際に再接続を試行する回数。
include_service_catalog = True	ブール値	(オプション)X-Service-Catalog ヘッダーを設定するかどうかを示します。False の場合、ミドルウェアはトークンの検証時にサービスカタログを要求せず、X-Service-Catalog ヘッダーを設定しません。
insecure = False	ブール値	HTTPS 接続を確認します。
interface = internal	文字列値	Identity API エンドポイントに使用するインターフェイス。有効な値は、"public"、"internal" (デフォルト)、または "admin" です。
keyfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
memcache_pool_conn_get_timeout = 10	整数値	(オプション) プールから memcached クライアント接続を取得するまで操作が待機する秒数。
memcache_pool_dead_retry = 300	整数値	(オプション)memcached サーバーが停止しているとみなされる秒数。この秒数が経過すると再試行されます。
memcache_pool_maxsize = 10	整数値	(オプション) すべての memcached サーバーへのオープン接続の最大合計数。
memcache_pool_socket_timeout = 3	整数値	(オプション)memcached サーバーと通信する際のソケットのタイムアウト (秒単位)。

設定オプション = デフォルト値	型	説明
memcache_pool_unused_timeout = 60	整数値	(オプション)memcached への接続がプール内で未使用の状態を維持する秒数。この秒数が経過すると終了されます。
memcache_secret_key = None	文字列値	(オプション、memcache_security_strategy が定義されている場合には必須) この文字列は鍵の導出に使用されます。
memcache_security_strategy = None	文字列値	(オプション) 定義されている場合は、トークンデータを認証、または認証して暗号化する必要があるかどうかを示します。MAC の場合、キャッシュでトークンデータが認証されます (HMAC を使用)。ENCRYPT の場合、キャッシュでトークンデータが暗号化され、認証されます。値がこれらのオプションのいずれでもない場合や空の場合には、auth_token は初期化時に例外を発生させます。
memcache_use_advanced_pool = False	ブール値	(オプション) 高度な (eventlet に対して安全な) memcached クライアントプールを使用します。高度なプールは python 2.x でのみ動作します。
memcached_servers = None	リスト値	オプションで、キャッシュに使用する memcached サーバーのリストを指定します。未定義のままの場合、トークンは代わりに処理中にキャッシュされます。
region_name = None	文字列値	アイデンティティーサーバーがあるリージョン。
service_token_roles = ['service']	リスト値	サービストークンに存在する必要があるロールの選択。サービストークンは、期限切れのトークンを使用できることを要求できるため、このチェックでは実際のサービスのみがこのトークンを送信するように厳密に制御する必要があります。ここでのロールは ANY チェックとして適用されるため、このリストのロールはすべて存在している必要があります。後方互換性の理由から、現在 allow_expired チェックにのみ影響します。
service_token_roles_required = False	ブール値	後方互換性の理由から、service_token_roles チェックを有効としてパスしない有効なサービストークンをパスさせる必要があります。これを true に設定することが今後のリリースでデフォルトとなり、可能な場合は有効にされる必要があります。
service_type = None	文字列値	サービスカタログに表示されるサービスの名前または種別。これは、制限されたアクセスルールがあるトークンを検証するために使用されます。

設定オプション = デフォルト値	型	説明
token_cache_time = 300	整数値	トークンの検証に過剰な時間を費やすのを防ぐために、ミドルウェアは、設定可能な期間 (秒単位) 中は以前に見たトークンをキャシュします。キャッシュを完全に無効にするには -1 に設定します。
www_authenticate_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。

3.1.14. oslo_concurrency

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[oslo_concurrency]** グループで利用可能なオプションの概要をまとめたものです。

表3.13 oslo_concurrency

設定オプション = デフォルト値	型	説明
disable_process_locking = False	ブール値	プロセス間ロックを有効または無効にします。
lock_path = None	文字列値	ロックファイルに使用するディレクトリー。セキュリティー上、指定したディレクトリーは、ロックが必要なプロセスを実行しているユーザーのみが書き込み可能である必要があります。デフォルトは環境変数 OSLO_LOCK_PATH です。外部ロックを使用する場合は、ロックパスを設定する必要があります。

3.1.15. oslo_messaging_amqp

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[oslo_messaging_amqp]** グループで利用可能なオプションの概要をまとめたものです。

表3.14 oslo_messaging_amqp

設定オプション = デフォルト値	型	説明
addressing_mode = dynamic	文字列値	ドライバーが使用するアドレスモードを示します。許可される値:**legacy** - レガシーのルーティング不可能なアドレス指定を使用します。**routable** - ルーティング可能なアドレスを使用します。**dynamic** - メッセージバスがルーティングをサポートしていない場合はレガシーのアドレスを使用します。それ以外の場合は、ルーティング可能なアドレスを使用します。
anycast_address = anycast	文字列値	コンシューマーのグループに送信する際に、アドレス接頭辞に追加されます。コンシューマー間でラウンドロビン方式で配信される必要のあるメッセージを特定するために、メッセージバスで使用されます。
broadcast_prefix = broadcast	文字列値	すべてのサーバーにブロードキャストする場合に使用されるアドレス接頭辞
connection_retry_backoff = 2	整数値	フェイルオーバーの試みに失敗するたびに、connection_retry_interval をこの秒数だけ増やします。
connection_retry_interval = 1	整数値	再接続を試みる前に一時停止する秒数。
connection_retry_interval_max = 30	整数値	connection_retry_interval + connection_retry_backoff の上限
container_name = None	文字列値	AMQP コンテナーの名前。グローバルで一意でなければなりません。デフォルトは、生成された UUID です。
default_notification_exchange = None	文字列値	通知アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_notification_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **notify**
default_notify_timeout = 30	整数値	送信された通知メッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_reply_retry = 0	整数値	リカバリー可能なエラーが原因で失敗した返信メッセージを再送信する最大試行回数。
default_reply_timeout = 30	整数値	rpc 返信メッセージ配信の期限。

設定オプション = デフォルト値	型	説明
default_rpc_exchange = None	文字列値	RPC アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_rpc_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **rpc**
default_send_timeout = 30	整数値	rpc キャストまたは呼び出しメッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_sender_link_timeout = 600	整数値	アイドル状態の送信者リンクのパージをスケジュールする時間。期限切れ後にリンクの割り当てを解除します。
group_request_prefix = unicast	文字列値	グループの任意のサーバーに送信する際のアドレス接頭辞
idle_timeout = 0	整数値	非アクティブな接続のタイムアウト (秒単位)
link_retry_delay = 10	整数値	リカバリー可能なエラーが原因で失敗した AMQP 1.0 リンクの再接続間に一時停止する時間。
multicast_address = multicast	文字列値	ファンアウトメッセージを送信するときにアドレス接頭辞に追加されます。ファンアウトメッセージを識別するためにメッセージバスによって使用されます。
notify_address_prefix = openstack.org/om/notify	文字列値	すべての生成される通知アドレスのアドレス接頭辞
notify_server_credit = 100	整数値	受信通知メッセージのウィンドウサイズ
pre_settled = ['rpc-cast', 'rpc-reply']	多値	このタイプのメッセージを事前処置状態で送信します。事前処置されたメッセージは、ピアから確認応答を受信しません。注記: 事前処置されたメッセージは、配信に失敗した場合に警告なしに破棄される可能性があります。許可される値:**rpc-call** - RPC 呼び出しを事前処置状態で送信します。**rpc-reply**- RPC 返信を事前処置状態で送信します。**rpc-cast** - RPC キャストを事前処置状態で送信します。**notify** - 通知を事前処置状態で送信します。

設定オプション = デフォルト値	型	説明
pseudo_vhost = True	ブール値	仮想ホスト (qpidd など) をネイティブにサポートしていないメッセージバスの仮想ホストサポートを有効にします。true に設定すると、仮想ホスト名はすべてのメッセージバスアドレスに追加され、結果として仮想ホストごとにプライベート **サブネット** を作成します。メッセージバスが仮想ホストの名前として AMQP 1.0 オープン遂行動詞の **hostname** フィールドを使用して仮想ホストをサポートする場合は False に設定します。
reply_link_credit = 200	整数値	受信する RPC 返信メッセージのウィンドウサイズ。
rpc_address_prefix = openstack.org/om/rpc	文字列値	すべての生成される RPC アドレスのアドレス接頭辞
rpc_server_credit = 100	整数値	受信する RPC リクエストメッセージのウィンドウサイズ。
`sasl_config_dir = `	文字列値	SASL 設定が含まれるディレクトリーへのパス
`sasl_config_name = `	文字列値	設定ファイルの名前 (.conf 接尾辞なし)
`sasl_default_realm = `	文字列値	ユーザー名にレルムが存在しない場合に使用する SASL レルム
`sasl_mechanisms = `	文字列値	許可される SASL メカニズムのスペース区切りリスト
server_request_prefix = exclusive	文字列値	特定のサーバーに送信する時に使用するアドレス接頭辞
ssl = False	ブール値	SSL 経由で接続を試みます。その他の ssl 関連のパラメーターが指定されていない場合、システムの CA バンドルを使用してサーバーの証明書を検証します。
`ssl_ca_file = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_cert_file = `	文字列値	クライアント認証用の自己識別証明書 PEM ファイル
`ssl_key_file = `	文字列値	ssl_cert_file 証明書の署名に使用される秘密鍵 PEM ファイル (オプション)
ssl_key_password = None	文字列値	ssl_key_file を復号するためのパスワード (暗号化されている場合)

設定オプション = デフォルト値	型	説明
ssl_verify_vhost = False	ブール値	デフォルトでは、SSL は、サーバーの証明書の名前が transport_url のホスト名と一致することを確認します。設定によっては、代わりに仮想ホスト名を使用することが望ましい場合があります。たとえば、サーバーが Server Name Indication TLS 拡張 (rfc6066) を使用して、仮想ホストごとの証明書を提供する場合などです。サーバーの SSL 証明書が DNS 名ではなく仮想ホスト名を使用する場合は、ssl_verify_vhost を True に設定します。
trace = False	ブール値	デバッグ: AMQP フレームを標準出力 (stdout) にダンプします。
unicast_address = unicast	文字列値	特定の RPC/通知サーバーに送信する際に、アドレス接頭辞に追加されます。単一の送信先に送信されたメッセージを識別するためにメッセージバスによって使用されます。

3.1.16. oslo_messaging_kafka

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[oslo_messaging_kafka]** グループで利用可能なオプションの概要をまとめたものです。

表3.15 oslo_messaging_kafka

設定オプション = デフォルト値	型	説明
compression_codec = none	文字列値	プロデューサーによって生成されたすべてのデータの圧縮コーデック。設定されていない場合、圧縮は使用されません。この設定に許可される値は、kafka バージョンに依存することに注意してください。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
consumer_group = oslo_messaging_consumer	文字列値	Kafka コンシューマーのグループ ID。あるグループのコンシューマーは、メッセージ消費を調整します。
enable_auto_commit = False	ブール値	非同期コンシューマーコミットを有効にします。
kafka_consumer_timeout = 1.0	浮動小数点の値	Kafka コンシューマーのデフォルトタイムアウト

設定オプション = デフォルト値	型	説明
kafka_max_fetch_bytes = 1048576	整数値	Kafka コンシューマーの最大フェッチバイト
max_poll_records = 500	整数値	ポーリング呼び出しで返されるレコードの最大数
pool_size = 10	整数値	Kafka コンシューマーのプールサイズ
producer_batch_size = 16384	整数値	プロデューサーの非同期送信のバッチサイズ
producer_batch_timeout = 0.0	浮動小数点の値	KafkaProducer バッチ処理の遅延の上限 (秒単位)
sasl_mechanism = PLAIN	文字列値	セキュリティープロトコルが SASL である際のメカニズム
security_protocol = PLAINTEXT	文字列値	ブローカーとの通信に使用されるプロトコル
`ssl_cafile = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_client_cert_file = `	文字列値	認証に使用されるクライアント証明書の PEM ファイル。
`ssl_client_key_file = `	文字列値	認証に使用されるクライアントキーの PEM ファイル。
`ssl_client_key_password = `	文字列値	認証に使用されるクライアントキーパスワードファイル。

3.1.17. oslo_messaging_notifications

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[oslo_messaging_notifications]** グループで利用可能なオプションの概要をまとめたものです。

表3.16 oslo_messaging_notifications

設定オプション = デフォルト値	型	説明
driver = []	多値	通知の送信を処理するドライバー。使用できる値は messaging、messagingv2、routing、log、test、noop です。

設定オプション = デフォルト値	型	説明
retry = -1	整数値	リカバリー可能なエラーが原因で配信に失敗した通知メッセージを再送信する最大試行回数。0 - 再試行なし、-1 - 無限回
topics = ['notifications']	リスト値	OpenStack の通知に使用する AMQP トピック。
transport_url = None	文字列値	通知に使用するメッセージングドライバーを表す URL。設定されていない場合は、RPC に使用されるものと同じ設定にフォールバックします。

3.1.18. oslo_messaging_rabbit

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[oslo_messaging_rabbit]** グループで利用可能なオプションの概要をまとめたものです。

表3.17 oslo_messaging_rabbit

設定オプション = デフォルト値	型	説明
amqp_auto_delete = False	ブール値	AMQP の自動削除キュー。
amqp_durable_queues = False	ブール値	AMQP で永続キューを使用します。
direct_mandatory_flag = True	ブール値	(非推奨) ダイレクト送信のための RabbitMQ 必須フラグを有効/無効にします。直接送信は応答として使用されるため、クライアントキューが存在しない場合には MessageUndeliverable 例外が発生します。MessageUndeliverable 例外はタイムアウトをループして、送信者が回復する可能性が高くなります。このフラグは非推奨になり、この機能を非アクティブ化することはできません。
enable_cancel_on_failover = False	ブール値	x-cancel-on-ha-failover フラグを有効にして、rabbitmq サーバーがキューが停止しているときにコンシューマーをキャンセルし、通知できるようにします。

設定オプション = デフォルト値	型	説明
heartbeat_in_pthread = False	ブール値	デフォルトでは、ネイティブ Python スレッドを介してヘルスチェックハートビートスレッドを実行します。このオプションが False の場合、ヘルスチェックハートビートは親プロセスから実行モデルを継承します。たとえば、親プロセスが eventlet/greenlet を使用して stdlib にモンキーパッチを適用した場合、ハートビートはグリーンスレッドを介して実行されます。This option should be set to True only for the wsgi services.
heartbeat_rate = 2	整数値	heartbeat_timeout_threshold 中、ハートビートを確認する回数。
heartbeat_timeout_threshold = 60	整数値	ハートビートの keep-alive が失敗した場合に Rabbit ブローカーがダウンとみなされるまでの秒数 (0 はハートビートを無効にします)。
kombu_compression = None	文字列値	実験的用途: 許容値は gzip、bz2 です。設定されていない場合、圧縮は使用されません。このオプションは、今後のバージョンで利用できない可能性があります。
kombu_failover_strategy = round-robin	文字列値	現在接続しているノードが利用できなくなった場合に、次の RabbitMQ ノードを選択する方法を指定します。設定で複数の RabbitMQ ノードが指定される場合に限り有効になります。
kombu_missing_consumer_retry_timeout = 60	整数値	応答のないクライアントが応答を送信するのを待つ時間。この時間が経過すると無視します。この値は rpc_response_timeout より長くすることはできません。
kombu_reconnect_delay = 1.0	浮動小数点の値	AMQP コンシューマーの取り消し通知への応答で、再接続するまでの待機時間。
rabbit_ha_queues = False	ブール値	RabbitMQ (x-ha-policy: all) の HA キューの使用を試みます。このオプションを変更する場合は、RabbitMQ データベースを消去する必要があります。RabbitMQ 3.0 では、キューを宣言する際に x-ha-policy 引数によってキューのミラーリングが制御されなくなりました。すべてのキュー (自動生成された名前のキューを除く) がすべてのノードでミラーリングされるようにするには、"rabbitmqctl set_policy HA **^(?!amq\.).* {"ha-mode": "all"}**" を実行します。
rabbit_interval_max = 30	整数値	RabbitMQ 接続を再試行する最大間隔。デフォルトは 30 秒です。

設定オプション = デフォルト値	型	説明
rabbit_login_method = AMQPLAIN	文字列値	RabbitMQ ログイン方法
rabbit_qos_prefetch_count = 0	整数値	事前フェッチするメッセージの数を指定します。ゼロに設定すると、無制限のメッセージのフェッチが許可されます。
rabbit_retry_backoff = 2	整数値	RabbitMQ に接続する際に再試行間でバックオフする長さ
rabbit_retry_interval = 1	整数値	RabbitMQ との接続を再試行する頻度。
rabbit_transient_queues_ttl = 1800	整数値	キューの TTL (x-expires) の期間 (秒単位) を表す正の整数。TTL の期間使用されないキューは自動的に削除されます。このパラメーターは応答キューとファンアウトキューにのみ影響します。
ssl = False	ブール値	SSL 経由で接続します。
`ssl_ca_file = `	文字列値	SSL 認証局ファイル (SSL が有効な場合にのみ有効)。
`ssl_cert_file = `	文字列値	SSL 証明書ファイル (SSL が有効な場合にのみ有効)。
`ssl_key_file = `	文字列値	SSL キーファイル (SSL が有効な場合にのみ有効)。
`ssl_version = `	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

3.1.19. oslo_middleware

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[oslo_middleware]** グループで利用可能なオプションの概要をまとめたものです。

表3.18 oslo_middleware

設定オプション = デフォルト値	型	説明
enable_proxy_headers_parsing = False	ブール値	アプリケーションがプロキシーの背後であるかどうか。これにより、ミドルウェアがヘッダーを解析すべきかどうかを決定します。

3.1.20. oslo_policy

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[oslo_policy]** グループで利用可能なオプションの概要をまとめたものです。

表3.19 oslo_policy

設定オプション = デフォルト値	型	説明
enforce_new_defaults = False	ブール値	このオプションは、ポリシーを評価するときに古い非推奨のデフォルトを使用するかどうかを制御します。**True** の場合、古い非推奨のデフォルトは評価されません。これは、既存のトークンが古いデフォルトで許可されているが、新しいデフォルトで許可されていない場合、許可されないことを意味します。新しいデフォルトと **scope_type** の利点を一緒に得ることができるように、このフラグを **enforce_scope** フラグと一緒に有効にすることを推奨します
enforce_scope = False	ブール値	このオプションは、ポリシーを評価する際にスコープを適用するかどうかを制御します。**True** の場合、リクエストで使用されるトークンのスコープは、適用されるポリシーの **scope_types** と比較されます。スコープが一致しない場合、**InvalidScope** 例外が発生します。**False** の場合、ポリシーが一致しないスコープで呼び出されていることを運用者に通知するメッセージが記録されます。
policy_default_rule = default	文字列値	デフォルトルール。要求されたルールが見つからない場合に適用されます。
policy_dirs = ['policy.d']	多値	ポリシー設定ファイルが保存されるディレクトリー。これは、config_dir オプションで定義される検索パス内の任意のディレクトリーへの相対パスか、絶対パスにすることができます。このディレクトリーの検索には、policy_file で定義されたファイルが存在している必要があります。存在しないディレクトリーまたは空のディレクトリーは無視されます。
policy_file = policy.yaml	文字列値	ロールを指定のサービスのパーミッションにマッピングするファイルの相対パスまたは絶対パス。相対パスは、このオプションを設定する設定ファイルに対する相対パスとして指定する必要があります。
remote_content_type = application/x-www-form-urlencoded	文字列値	REST ベースのポリシーチェックのデータを送受信するコンテンツ種別
remote_ssl_ca_crt_file = None	文字列値	REST ベースのポリシーチェック用の認証局証明書ファイルの絶対パス

設定オプション = デフォルト値	型	説明
remote_ssl_client_crt_file = None	文字列値	REST ベースのポリシーチェック用のクライアント証明書の絶対パス
remote_ssl_client_key_file = None	文字列値	REST ベースのポリシーチェック用のクライアント鍵ファイルの絶対パス
remote_ssl_verify_server_crt = False	ブール値	REST ベースのポリシーチェックのサーバーアイデンティティー検証

3.1.21. paste_deploy

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[paste_deploy]** グループで利用可能なオプションの概要をまとめたものです。

表3.20 paste_deploy

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
config_file = None	文字列値	貼り付け設定ファイルの名前。 サーバーアプリケーションのデプロイメントのパイプラインを設定するために使用される貼り付け設定ファイルの名前を表す文字列値を指定します。 注記: ● 貼り付け設定ファイル用の glance ディレクトリーの名前、または絶対パスではなく相対パスを指定します。 ● Glance に同梱されるサンプルの貼り付け設定ファイルは、一般的なすべてのデプロイメントフレーバーの設定済みのパイプラインが含まれるため、ほとんどの場合編集する必要はありません。 このオプションに値が指定されていない場合、対応する Glance サービスの設定ファイル名を接頭辞に持つ **paste.ini** ファイルが既知の設定ディレクトリーで検索されます。(たとえば、**glance-api.conf** にこのオプションがない場合や、値が設定されていない場合には、サービスは **glance-api-paste.ini** という名前のファイルを探します)。貼り付け設定ファイルが見つからない場合、サービスは起動しません。 値: ● 貼り付け設定ファイルの名前を表す文字列値。 関連するオプション: ● flavor

設定オプション = デフォルト値	型	説明
flavor = None	文字列値	サーバーアプリケーションパイプラインで使用するデプロイメントフレーバー。 サーバーアプリケーションパイプラインで使用される適切なデプロイメントフレーバーを表す文字列値を指定します。通常、これはサービス名が削除された貼り付け設定ファイルのパイプラインの部分的な名前です。 たとえば、貼り付け設定ファイルの貼り付けセクション名が [pipeline:glance-api-keystone] の場合、**flavor** を **keystone** に設定します。 値: ● 部分的なパイプライン名を表す文字列値。 関連するオプション: ● config_file

3.1.22. profiler

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[profiler]** グループで利用可能なオプションの概要をまとめたものです。

表3.21 profiler

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
connection_string = messaging://	文字列値	通知バックエンドの接続文字列。 デフォルト値は **messaging://** で、通知機能を oslo_messaging に設定します。 使用できる値の例: ● **messaging://**: スパンの送信に oslo_messaging ドライバーを使用します。 ● **redis://127.0.0.1:6379**: スパンの送信に redis ドライバーを使用します。 ● **mongodb://127.0.0.1:27017**: スパンの送信に mongodb ドライバーを使用します。 ● **elasticsearch://127.0.0.1:9200:** elasticsearch ドライバーを使用してスパンを送信します。 ● **jaeger://127.0.0.1:6831**: スパンの送信に jaeger トレースを使用します。
enabled = False	ブール値	このノード上の全サービスのプロファイリングを有効にします。 デフォルト値は False です (プロファイリング機能を無効にします)。 値: ● True: 機能を有効にします ● False: 機能を無効にします。このプロジェクトの操作ではプロファイリングを開始できません。プロファイリングが別のプロジェクトでトリガーされた場合には、このプロジェクトの部分は空になります。
es_doc_type = notification	文字列値	elasticsearch の通知インデックスについてのドキュメントタイプ。
es_scroll_size = 10000	整数値	elasticsearch は大規模な要求をバッチに分割します。このパラメーターは、各バッチの最大サイズを定義します (例: es_scroll_size=10000)。
es_scroll_time = 2m	文字列値	このパラメーターは時間値パラメーター (es_scroll_time=2m など) です。これは、検索に参加するノードが継続してサポートするために関連するリソースを維持する時間を示します。

設定オプション = デフォルト値	型	説明
filter_error_trace = False	ブール値	エラー/例外を含むフィルタートレースを分離された場所へ有効にします。 デフォルト値は False に設定されます。 値: ● True: エラー/例外を含むフィルタートレースを有効にします。 ● False: フィルターを無効にします。
hmac_keys = SECRET_KEY	文字列値	パフォーマンスプロファイリング用にコンテキストデータの暗号化に使用するシークレットキー。 この文字列値は <key1>[,<key2>,...<keyn>] の形式にする必要があります。各キーはランダムな文字列です。REST API 経由でプロファイリングをトリガーするユーザーは、REST API 呼び出しのヘッダーにこれらのキーの 1 つを設定し、この特定プロジェクトのノードのプロファイリング結果が含まれるようにする必要があります。 プロファイリングを有効にするには、enabled フラグと hmac_keys 設定オプションの両方を設定する必要があります。また、少なくとも 1 つのキーで全サービス間で正しいプロファイリング情報を生成するには、OpenStack プロジェクト間で一貫性を確保する必要があります。これにより、クライアント側からを使用して、すべての可能なリソースからの情報を含むトレースを生成できます。
sentinel_service_name = mymaster	文字列値	Redissentinel はサービス名を使用してマスターの redis サービスを識別します。このパラメーターは名前を定義します (例: **sentinal_service_name=mymaster**)。
socket_timeout = 0.1	浮動小数点の値	Redissentinel は、接続に timeout オプションを提供します。このパラメーターは、タイムアウトを定義します (例: socket_timeout=0.1)。

設定オプション = デフォルト値	型	説明
trace_sqlalchemy = False	ブール値	サービスで SQL 要求のプロファイリングを有効にします。 デフォルト値は False です (SQL 要求はトレースされません)。 値: ● True: SQL 要求のプロファイリングを有効にします。各 SQL クエリーはトレースの一部となり、それにどの程度の時間を費やしたかによって分析できます。 ● False: SQL 要求のプロファイリングを無効にします。費やした時間は、高いレベルの操作でのみ表示されます。単一の SQL クエリーをこのように分析できません。

3.1.23. store_type_location_strategy

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[store_type_location_strategy]** グループで利用可能なオプションの概要をまとめたものです。

表3.22 store_type_location_strategy

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明

store_type_preference = []	リスト値	ストレージバックエンドの優先順位。
		
		ストレージバックエンドからイメージを取得する順序で、ストア名のコンマ区切りリストを指定します。これらのストア名は、**stores** 設定オプションで登録する必要があります。
		
		注記
		
		store_type が **location_strategy** 設定オプションの値として選択されている場合にのみ、**store_type_preference** 設定オプションは適用されます。空のリストは、場所の順序を変更しません。
		
		値:
		
		- 空のリスト
		- 登録ストア名のコンマ区切りリスト。有効な値は次のとおりです。
		  - file
		  - http
		  - rbd
		  - swift
		  - cinder
		  - vmware
		
		関連するオプション:
		
		- location_strategy
		- stores

3.1.24. task

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[task]** グループで利用可能なオプションの概要をまとめたものです。

表3.23 task

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
task_executor = taskflow	文字列値	タスクスクリプトの実行に使用するタスクエグゼキューター。 タスクの実行に使用するエグゼキューターを表す文字列値を指定します。デフォルトでは、**TaskFlow** エグゼキューターが使用されます。 **TaskFlow** は、タスクの実行を簡単にし、一貫性を待たせ、スケーラブルで信頼性を持たせるのに役立ちます。また、宣言型の方法でフローに統合される軽量のタスクオブジェクトや関数を作成することもできます。 値: ● taskflow 関連するオプション: ● なし
task_time_to_live = 48	整数値	成功するか、失敗するかのいずれかの後にタスクがライブである期間 (時間単位)

設定オプション = デフォルト値	型	説明
work_dir = None	文字列値	非同期タスク操作に使用する作業ディレクトリーへの絶対パス。 ここで設定したディレクトリーは、通常、インポート先ストアに送付する前にイメージを操作するために使用されます。 **注記** **work_dir** の値を指定する際には、同時タスクが領域を使い果たすことなく効率的に実行されるように、十分な領域を確保してください。 大まかな予測は、平均のイメージサイズ (例:500 MB) に **max_workers** の数を掛けることで計算できます。イメージサイズの推定は、実際のデプロイメントの平均サイズに基づいて行う必要があります。実行しているタスクの動作に応じて、この値に係数を乗算しないといけない点に注意してください。たとえば、イメージの変換が有効な場合は、利用可能なサイズを 2 倍にしなければならない場合があります。しかしながら、ここで紹介する計算方法は推測用であり、最悪ケースのシナリオに基づいて計算を行う必要があり、正しくなかった場合に対処できるように準備をしてください。 値: ● 作業ディレクトリーへの絶対パスを表す文字列値 関連するオプション: ● なし

3.1.25. taskflow_executor

以下の表は、**/etc/glance/glance-api.conf** ファイルの **[taskflow_executor]** グループで利用可能なオプションの概要をまとめたものです。

表3.24 taskflow_executor

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
conversion_format = None	文字列値	必要なイメージ変換形式を設定します。 Glance による消費のために保存する前にイメージを変換する有効なイメージ形式を指定します。ストレージインフラストラクチャーの帯域幅と使用状況を効率的に処理できるように、特定のストレージバックエンドでは適切なイメージ形式への変換が必要です。 デフォルトでは、**conversion_format** は設定されていないため、設定ファイルで明示的に設定する必要があります。 このオプションで使用できる値は **raw**、**qcow2**、および **vmdk** です。**raw** フォーマットは非構造化ディスクフォーマットであり、RBD または Ceph Storage バックエンドがイメージストレージに使用される場合に選択する必要があります。**qcow2** は、動的に拡張し、Copy on Write をサポートする QEMU エミュレーターによってサポートされています。**vmdk** は、VMWare Workstation などの多くの一般的な仮想マシンモニターでサポートされている別の一般的なディスク形式です。 値: ● qcow2 ● raw ● vmdk 関連するオプション: ● disk_formats

設定オプション = デフォルト値	型	説明
engine_mode = parallel	文字列値	タスクフローエンジンモードを設定します。 タスクフローエンジンがホスト上のワーカーにタスクをスケジュールするモードを設定する際の文字列種別の値を指定します。このモードに基づいて、エンジンは単一または複数のスレッドのいずれかでタスクを実行します。このオプションに使用できる値は **serial** と **parallel** です。**serial** に設定すると、エンジンは単一のスレッドですべてのタスクを実行し、タスクは順次実行されます。これを **parallel** に設定すると、エンジンは複数のスレッドでタスクを実行するようになります。これにより、タスクは並列で実行されます。 値: ● serial ● parallel 関連するオプション: ● max_workers
max_workers = 10	整数値	エンジンの実行可能タスクの数を設定します。 ホストでインスタンス化できるワーカーの数を制限する整数値を指定します。つまり、この数は、タスクフローエンジンで同時に実行できる並列タスクの数を定義します。エンジンモードが parallel に設定されている場合に、この値を 1 より大きい値に設定できます。 値: ● 1 以上の整数値 関連するオプション: ● engine_mode

3.1.26. wsgi

次の表は、**/etc/glance/glance-api.conf** ファイルの **[wsgi]** グループで使用できるオプションの概要を示しています。

表3.25 wsgi

設定オプション = デフォルト値	型	説明
python_interpreter = /usr/bin/python3	文字列値	外部プロセスを生成するときに使用する Python インタープリターへのパス。デフォルトでは、これは sys.executable であり、Glance 自体を実行しているのと同じインタープリターである必要があります。ただし、状況によっては (uwsgi など)、実際には Python インタープリター自体を指していない場合があります。
task_pool_threads = 16	整数値	非同期タスクを処理するためのプール内の (ワーカープロセスごとの) スレッドの数。これは、各ワーカーが一度に実行できる非同期タスク (イメージの相互運用可能なインポートなど) の数を制御します。これが大きすぎると、ワーカーあたりのメモリーフットプリントが増加したり、ディスクやアウトバウンドネットワーク帯域幅などの他のシステムリソースを圧倒したりする **可能性** があります。これが小さすぎる場合、イメージのインポート要求は、スレッドが処理を開始できるようになるまで待たなければなりません。

3.2. GLANCE-SCRUBBER.CONF

このセクションには、**/etc/glance/glance-scrubber.conf** ファイルのオプションが含まれます。

3.2.1. DEFAULT

以下の表は、**/etc/glance/glance-scrubber.conf** ファイルの **[DEFAULT]** グループで利用可能なオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
allow_additional_image_properties = True	ブール値	ユーザーがイメージに追加/カスタムの属性を追加できるようにします。 Glance は、すべてのイメージに表示される属性の標準セット () スキーマ) を定義します。これらのプロパティーは、**base properties** とも呼ばれます。Glance では、これらの属性に加え、カスタムプロパティーをイメージに追加できます。これらは、**additional properties** と呼ばれています。 デフォルトでは、この設定オプションは **True** に設定され、ユーザーは追加のプロパティーを追加できます。イメージに追加できる追加プロパティーの数は、**image_property_quota** 設定オプションで制御できます。 値: ● True ● False 関連するオプション: ● image_property_quota **非推奨:** Ussuri **理由:** このオプションは冗長です。**image_property_quota** 設定オプションを使用して、カスタムイメージプロパティーの使用を制御します。このオプションは、Victoria の開発サイクル中に削除される予定です。

設定オプション = デフォルト値	型	説明
api_limit_max = 1000	整数値	要求によって返されることのできる最大結果数。 **limit_param_default** のヘルプテキストで説明されているように、リクエストによっては複数の結果が返される場合があります。返される結果の数は、要求の **limit** パラメーターまたは **limit_param_default** 設定オプションで管理されます。いずれの場合もこの値が、この設定オプションで定義される絶対最大値よりも大きくすることはできません。この値より大きいすべてのものは、ここで定義される最大値にトリミングされます。 **注記** これを非常に大きな値に設定すると、データベースのクエリーが遅くなり、応答時間が長くなる可能性があります。これを非常に低い値に設定すると、ユーザーエクスペリエンスが低下してしまう可能性があります。 値: ● 任意の正の整数 関連するオプション: ● limit_param_default

設定オプション = デフォルト値	型	説明
daemon = False	ブール値	スクラバーをデーモンとして実行します。 このブール値設定オプションは、スクラバーが、イメージをスクラブするのに一定間隔で起動する長時間実行プロセスとして実行されるかどうかを示します。起動の間隔は、設定オプション **wakeup_time** を使用して指定できます。 この設定オプションがデフォルト値の **False** に設定されている場合、スクラバーはイメージのスクラブのために1回動作して終了します。この場合、運用者がイメージの継続的なスクラビングを実装する場合、スクラバーは cron ジョブとしてスケジュールする必要があります。 値: ● True ● False 関連するオプション: ● **wakeup_time**
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。

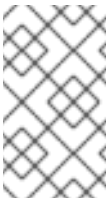
設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
delayed_delete = False	ブール値	削除をオン/オフにします。 通常、イメージが削除されると、**glance-api** サービスはイメージを **delete** 状態にし、同時にそのデータを削除します。遅延削除とは、Glance の機能のひとつで、イメージデータの実際の削除を (設定オプション **scrub_time** で決定される) 後の時点まで遅らせるものです。遅延削除が有効になると、**glance-api** サービスは削除時にイメージを **pending_delete** の状態に入れ、イメージが scrubber のストレージバックエンドに残され、後で削除されるようにします。イメージスクラバは、イメージデータの削除に成功すると、イメージを **deleted** 状態に移行させます。 **注記** 遅延削除をオンにすると、バックエンドのストレージが不要な使用でいっぱいになるのを防ぐために、イメージスクラバを定期的なタスクとして実行しなければなりません。 値: ● True ● False 関連するオプション: ● **scrub_time** ● **wakeup_time** ● **scrub_pool_size**

設定オプション = デフォルト値	型	説明
digest_algorithm = sha256	文字列値	デジタル署名に使用するダイジェストアルゴリズム。 デジタル署名の生成に使用するダイジェストアルゴリズムを表す文字列値を指定します。デフォルトでは **sha256** が使用されます。 プラットフォームの OpenSSL バージョンでサポートされる利用可能なアルゴリズムのリストを取得するには、コマンド **openssl list-message-digest-algorithms** を実行します。たとえば、sha1、sha256、および sha512 です。 **注記** **digest_algorithm** は Glance のイメージの署名および検証に関連しません。これは、証明書ファイルとキーファイル検証の一部として汎用一意識別子 (UUID) に署名するために使用されます。 値: ● OpenSSL メッセージダイジェストアルゴリズム ID 関係オプション: ● なし
enabled_import_methods = ['glance-direct', 'web-download', 'copy-image']	リスト値	有効なイメージのインポートメソッドの一覧 'glance-direct', 'copy-image' and 'web-download' are enabled by default. Related options: ** [DEFAULT]/node_staging_uri

設定オプション = デフォルト値	型	説明
enforce_secure_rbac = False	ブール値	OpenStack 全体で使用される一般的なペルソナ定義に基づいて API アクセスを強制します。このオプションを有効にすると、**member** ロールの背後で、プライベートイメージの作成や共有イメージのステータスの更新など、プロジェクト固有の読み取り/書き込み操作が形式化されます。また、**reader** ロールの背後で、プロジェクト内のプライベートイメージをリスト表示するなど、プロジェクト固有の API 操作に役立つ読み取り専用バリアントを形式化します。 オペレーターはこの機会に、glance の新しいイメージポリシーを理解し、デプロイメントでの割り当てを監査し、keystone のデフォルトロール (**admin**、**member**、**reader** など) を使用して権限を更新する必要があります。 関連するオプション: ● [oslo_policy]/enforce_new_defaults **非推奨:** Wallaby **理由:** このオプションは、一般的な RBAC ペルソナに基づいて承認を実施することを選択することをオペレーターに要求するために導入されました。これは Wallaby リリースの時点で実験的です。この動作はデフォルトであり、将来のリリースでは STABLE になり、このオプションを削除できるようになります。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。

設定オプション = デフォルト値	型	説明
hashing_algorithm = sha512	文字列値	os_hash_value プロパティーの計算に使用される安全なハッシュアルゴリズム。 このオプションは、**os_hash_algo** と **os_hash_value** の 2 つのイメージ属性で設定される Glance の multihash を設定します。**os_hash_algo** はこの設定オプションの値によって入力され、**os_hash_value** は、アルゴリズムがアップロードまたはインポートされたイメージデータに適用されると計算された hexdigest によって入力されます。 この値は、python **hashlib** ライブラリーが認識する有効なセキュアなハッシュアルゴリズム名である必要があります。Glance インストールで使用されているライブラリーのバージョンの **hashlib.algorithms_available** データメンバーを調べることで、これらを確認することができます。ただし、相互運用性のために、**hashlib.algorithms_guaranteed** データメンバーが提供する安全なハッシュ名のセットを使用することが推奨されます。これは、すべてのプラットフォームの **hashlib** ライブラリーでサポートされることが保証されているためです。したがって、**hashlib** を使用するイメージコンシューマーは、イメージの **os_hash_value** を検証できるはずです。 **sha512** のデフォルト値は、高性能なハッシュアルゴリズムです。 このオプションの設定が間違っていると、イメージデータの格納試行は失敗します。このため、デフォルト値を使用することが推奨されます。 値: ● Python **hashlib** ライブラリーが認識するセキュアなハッシュアルゴリズム名 関連するオプション: ● なし
image_location_quota = 10	整数値	イメージで許可される場所の最大数。 負の値は無制限として解釈されます。 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
image_member_quota = 128	整数値	1イメージあたりのイメージメンバーの最大数。 これにより、イメージが共有できるユーザーの最大数が制限されます。負の値は無制限として解釈されます。 関連するオプション: ● なし
image_property_quota = 128	整数値	イメージで許可されるプロパティーの最大数。 これにより、イメージに割り当てることができる追加のプロパティーの数の上限が適用されます。負の値は無制限として解釈されます。  **注記** 追加のプロパティーが無効の場合、影響はありません。**allow_additional_image_properties** を参照してください。 関連するオプション: ● **allow_additional_image_properties**

設定オプション = デフォルト値	型	説明
image_size_cap = 1099511627776	整数値	ユーザーがアップロードできるイメージの最大サイズ (バイト単位)。 上記のサイズより大きいイメージをアップロードすると、イメージの作成に失敗します。このオプションはデフォルトで 1099511627776 バイト (1 TiB) に設定されます。 注記: ● この値は、慎重に検討した後にのみ増やす必要があり、8 EiB (9223372036854775808) 以下を設定する必要があります。 ● この値は、バックエンドストレージ容量を慎重に検討して設定する必要があります。この値を非常に小さい値に設定すると、イメージが多数失敗する可能性があります。また、この値を非常に大きな値に設定すると、ストレージの使用が速くなる可能性があります。したがって、これは、利用可能なイメージおよび利用可能なストレージ容量の性質に応じて設定される必要があります。 値: ● 9223372036854775808 未満の正数
image_tag_quota = 128	整数値	イメージで許可されるタグの最大数。 負の値は無制限として解釈されます。 関連するオプション: ● なし
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。

設定オプション = デフォルト値	型	説明
limit_param_default = 25	整数値	リクエストに対して返すデフォルトの結果数。 リストイメージなどの特定の API リクエストへの応答により、複数のアイテムが返される場合があります。返される結果の数は、API 要求で **limit** パラメーターを指定して明示的に制御できます。ただし、**limit** パラメーターが指定されていない場合、この設定値は API 要求に対して返される結果のデフォルト数として使用されます。 注記: ● このオプションの値は、**api_limit_max** で指定された値よりも大きくすることはできません。 ● これを非常に大きな値に設定すると、データベースのクエリーが遅くなり、応答時間が長くなる可能性があります。これを非常に低い値に設定すると、ユーザーエクスペリエンスが低下してしまう可能性があります。 値: ● 任意の正の整数 関連するオプション: ● api_limit_max
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
metadata_encryption_key = None	文字列値	ストアの場所のメタデータを暗号化する際の AES 鍵。 Glance ストアメタデータの暗号化に使用する AES 暗号化を表す文字列値を指定します。 **注記** 使用する AES 鍵は、長さ 16、24、または 32 バイトのランダムな文字列に設定する必要があります。 値: ● 有効な AES 鍵を表す文字列値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
node_staging_uri = file:///tmp/staging/	文字列値	URL は一時データが保存される場所を指定します。 このオプションは、Glance の内部使用専用です。Glance は、イメージのインポートプロセス中、ユーザーがアップロードしたイメージデータを **ステージング** エンドポイントに保存します。 このオプションは、**ステージング** API エンドポイントを一切変更しません。 **注記** [task]/work_dir と同じパスを使用することは推奨されません。 **注記** file://<absolute-directory-path> is the only option api_image_import flow will support for now. **注記** ステージングパスは、すべての Glance API ノードからアクセス可能な共有ファイルシステム上になければなりません。 値: ● **file://** で始まり、ファイルシステムの絶対パスが続く文字列 関連するオプション: ● [task]/work_dir
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。

設定オプション = デフォルト値	型	説明
pydev_worker_debug_host = None	ホストのアドレス値	pydev サーバーのホストアドレス。 デバッグに使用する pydev サーバーのホスト名または IP を表す文字列値を指定します。pydev サーバーはこのアドレスでデバッグ接続をリッスンし、Glance でのリモートデバッグを容易にします。 値: ● 有効なホスト名 ● 有効な IP アドレス 関連するオプション: ● なし
pydev_worker_debug_port = 5678	ポート値	pydev サーバーがリッスンするポート番号。 pydev サーバーをバインドするポート番号を指定します。pydev プロセスはこのポートでデバッグ接続を受け入れ、Glance でのリモートデバッグを容易にします。 値: ● 有効なポート番号 関連するオプション: ● なし
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。

設定オプション = デフォルト値	型	説明
restore = None	文字列値	イメージのステータスを **pending_delete** から **active** に復元します。 このオプションは、イメージが誤って削除され、Glance で**削除の保留** 機能が有効な場合に、管理者がイメージのステータスを **pending_delete** から **active** にリセットする際に使用されます。イメージデータの不整合を避けるために、イメージを復元する前に glance-scrubber デーモンが停止していることを確認してください。 値: ● イメージの uuid
scrub_pool_size = 1	整数値	イメージのスクラビングに使用されるスレッドプールのサイズ。 スクラブするイメージが多数ある場合には、スクラブキューが制御された状態に維持され、バックエンドストレージがタイムリーに要求されるように、イメージを並行してスクラブすると便利です。この設定オプションは、並行してスクラブされるイメージの最大数を示します。デフォルト値は 1 で、シリアルスクラビングを意味します。1 を超える値はすべて、並列スクラビングを意味します。 値: ● ゼロ以外の正の整数 関連するオプション: ● **delayed_delete**

設定オプション = デフォルト値	型	説明
scrub_time = 0	整数値	イメージのスクラビングを遅延させる時間 (秒単位)。 遅延削除がオンの場合、スクラバーがそのイメージデータを削除するまで、イメージは削除時に **pending_delete** 状態になります。通常、イメージが **pending_delete** 状態になると、すぐにスクラビングできます。ただし、この設定オプションを使用して、後の時点までスクラビングを遅らせることができます。このオプションは、スクラビングできるまで、イメージが **pending_delete** 状態である期間を示します。 これによりストレージに影響が出ることを認識することが重要です。**scrub_time** の値が大きいほど、削除されるイメージからバックエンドストレージを再要求する時間が長くなります。 値: ● 負の値以外の整数 関連するオプション: ● **delayed_delete**

設定オプション = デフォルト値	型	説明
show_image_direct_url = False	ブール値	イメージを返す際に直接イメージの場所を示します。 この設定オプションは、イメージの詳細をユーザーに返すときに直接イメージの場所を表示するかどうかを指定します。直接イメージの場所は、バックエンドストレージ内でイメージデータが保存される場所です。このイメージの場所は、イメージ属性 **direct_url** セクションに表示されます。 イメージに対して複数のイメージの場所が存在する場合には、設定オプション **location_strategy** で示される場所ストラテジーに基づいて、最適な場所が表示されます。 注記: ● イメージの場所を表示すると、イメージの場所に認証情報が含まれることがあるため、重大なセキュリティーリスクが発生する可能性があります。したがって、これはデフォルトで **False** に設定されています。設定の意味するところを理解している場合に限り、細心の注意を払ってこのオプションを **True** に設定してください。 ● 運用者がユーザーにイメージの場所を表示しないことを希望する場合は、このオプションと **show_multiple_locations** の両方を **False** に設定する必要があります。 値: ● True ● False 関連するオプション: ● show_multiple_locations ● location_strategy

設定オプション = デフォルト値	型	説明
show_multiple_locations = False	ブール値	イメージを返す際にすべてのイメージの場所を示します。 この設定オプションは、イメージの詳細をユーザーに返すときにすべてのイメージの場所を表示するかどうかを指定します。イメージに対して複数のイメージの場所が存在する場合には、設定オプション **location_strategy** で示される場所ストラテジーに基づいて、場所が順番に表示されます。イメージの場所は、イメージ属性 **locations** セクションに表示されます。 注記: ● イメージの場所を表示すると、イメージの場所に認証情報が含まれることがあるため、重大なセキュリティーリスクが発生する可能性があります。したがって、これはデフォルトで **False** に設定されています。設定の意味するところを理解している場合に限り、細心の注意を払ってこのオプションを **True** に設定してください。 ● 詳細は、https://wiki.openstack.org/wiki/OSSN/OSSN-0065 を参照してください。 ● 運用者がユーザーにイメージの場所を表示しないことを希望する場合は、このオプションと **show_image_direct_url** の両方を **False** に設定する必要があります。 値: ● True ● False 関連するオプション: ● show_image_direct_url ● location_strategy **非推奨となったバージョン：** Newton *理由:* Newton 以降非推奨になったこのオプションの使用はセキュリティーリスクであり、現在このオプションを必要とするそれらのユースケースを満たす方法がわかれば削除されます。ポリシーを使用することで同じ機能をより細かい粒度で実現できるという以前の通知は、正しくありません。ポリシー設定でこのオプションを回避することが取るべき修正の方向と考えていますが、現在、この方法で回避することはできません。この問題への対処に関する最新の進捗状況を把握するために、常に Glance のリリースノートに注意を払ってください。

設定オプション = デフォルト値	型	説明
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
user_storage_quota = 0	文字列値	テナントごとのイメージストレージの最大容量。 これにより、全ストアにわたってテナントのすべてのイメージによって消費される累積ストレージに上限が適用されます。これはテナントごとの制限です。 この設定オプションのデフォルトの単位はバイトです。ただし、ストレージの単位は、大文字/小文字を区別する文字**B**、**KB**、**MB**、**GB**、および **TB**を使用して指定することができ、それぞれバイト、キロバイト、メガバイト、ギガバイト、およびテラバイトを表します。値と単位の間にはスペースを入れないでください。値 **0** は、クォータが適用されないことを意味します。負の値は無効で、エラーが発生します。 値: ● 上記のように、ストレージの値を表す負ではない整数とストレージの単位を表すオプションの文字列の有効な組み合わせである文字列。 関連するオプション: ● なし
wakeup_time = 300	整数値	スクラバーがデーモンモードで実行される間隔 (秒単位)。 スクラバーは、cron ジョブまたはデーモンのいずれかとして実行できます。デーモンとして実行する場合、この設定時間は 2 つの実行の間隔を指定します。スクラバーが起動すると、**scrub_time** を考慮した後に、スクラブできるすべての **pending_delete** イメージを取得してスクラブします。 起動間隔が大きな数値に設定されている場合には、実行ごとに多数のイメージがスクラブされる可能性があります。これにより、バックエンドストレージが再要求される速度にも影響が出ます。 値: ● 負の値以外の整数 関連するオプション: ● **daemon** ● **delayed_delete**

設定オプション = デフォルト値	型	説明
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
worker_self_reference_url = None	文字列値	このワーカーへの URL。 これが設定されている場合、他のグランスワーカーは、必要に応じてこのワーカーに直接連絡する方法を知っています。イメージのインポートの場合、1 人のワーカーがイメージをステージングし、他のワーカーがインポートリクエストを適切なワーカーにプロキシーできる必要があります。 設定されていない場合、これは **public_endpoint** と見なされ、通常はすべてのワーカーで同じ値に設定され、プロキシー動作を効果的に無効にします。 値: ● このワーカーが他のワーカーから到達可能な URL 関連するオプション: ● public_endpoint

3.2.2. database

以下の表は、**/etc/glance/glance-scrubber.conf** ファイルの **[database]** グループで利用可能なオプションの概要をまとめたものです。

表3.26 database

設定オプション = デフォルト値	型	説明
backend = sqlalchemy	文字列値	データベースに使用するバックエンド。
connection = None	文字列値	データベースへの接続に使用する SQLAlchemy 接続文字列。
connection_debug = 0	整数値	SQL デバッグ情報の冗長性: 0=なし、100=すべて。
`connection_parameters = `	文字列値	接続時に接続 URL に追加するオプションの URL パラメーター。param1=value1&param2=value2&... として指定します。
connection_recycle_time = 3600	整数値	この秒数より長く接続プールに存在していた接続は、次にプールからチェックアウトされたときに新しい接続に置き換えられます。
connection_trace = False	ブール値	Python スタックトレースをコメント文字列として SQL に追加します。

設定オプション = デフォルト値	型	説明
db_inc_retry_interval = True	ブール値	True の場合には、db_max_retry_interval までのデータベース操作の再試行回数を長くします。
db_max_retries = 20	整数値	エラーが発生する前に接続エラーまたはデッドロックが発生した場合の最大再試行。再試行回数を無限に指定するには -1 に設定します。
db_max_retry_interval = 10	整数値	db_inc_retry_interval が設定されている場合は、データベース操作を再試行するまでの最大秒数です。
db_retry_interval = 1	整数値	データベーストランザクションの再試行間の秒数。
max_overflow = 50	整数値	設定されている場合、この値を SQLAlchemy で max_overflow に使用します。
max_pool_size = 5	整数値	プールに開いたままにする SQL 接続の最大数。値を 0 に設定すると無制限を意味します。
max_retries = 10	整数値	起動時のデータベース接続の最大再試行数。再試行回数を無限に指定するには -1 に設定します。
mysql_enable_ndb = False	ブール値	True の場合には、MySQL Cluster (NDB) の処理のサポートを透過的に有効にします。
mysql_sql_mode = TRADITIONAL	文字列値	MySQL セッションに使用する SQL モード。このオプション (デフォルトを含む) は、サーバーセット SQL モードを上書きします。サーバー設定で設定された SQL モードを使用するには、これを no value に設定します。例: mysql_sql_mode=
pool_timeout = None	整数値	設定されている場合は、この値を SQLAlchemy で pool_timeout に使用します。
retry_interval = 10	整数値	SQL 接続を開く再試行の間隔。
slave_connection = None	文字列値	スレーブデータベースへの接続に使用する SQLAlchemy 接続文字列。
sqlite_synchronous = True	ブール値	True の場合、SQLite は同期モードを使用します。
use_db_reconnect = False	ブール値	失われた接続でデータベースの実験的な使用を有効にします。
use_tpool = False	ブール値	すべての DB API 呼び出しのスレッドプールの実験的な使用を有効にします。

3.2.3. glance_store

以下の表は、**/etc/glance/glance-scrubber.conf** ファイルの **[glance_store]** グループで利用可能なオプションの概要をまとめたものです。

表3.27 glance_store

設定オプション = デフォルト値	型	説明
cinder_api_insecure = False	ブール値	cinder に対するセキュアではない SSL リクエストの実行を許可します。 このオプションが True に設定されている場合、HTTPS エンドポイント接続は **cinder_ca_certificates_file** オプションで指定された CA 証明書ファイルを使用して検証されます。 値: ● True ● False 関連するオプション: ● cinder_ca_certificates_file
cinder_ca_certificates_file = None	文字列値	cinder クライアントリクエストに使用する CA 証明書ファイルの場所。 CA 証明書ファイルが設定されている場合、指定されたファイルが、HTTPS エンドポイントを使用した cinder 接続の検証に使用されます。エンドポイントが HTTP の場合、この値は無視されます。検証を有効にするには、**cinder_api_insecure** を **True** に設定する必要があります。 値: ● CA 証明書ファイルへのパス 関連するオプション: ● cinder_api_insecure

設定オプション = デフォルト値	型	説明
cinder_catalog_info = volumev3::publicURL	文字列値	サービスカタログで cinder を検索する際に照合する情報。 **cinder_endpoint_template** が設定されておらず、**cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、**cinder_store_password** のいずれも設定されていない場合、cinder store はこの情報を使用して、現在のコンテキストでサービスカタログから cinder エンドポイントを検索します。**cinder_os_region_name** が設定されている場合は、適切なエンドポイントを取得するために考慮されます。 サービスカタログは、**openstack catalog list** コマンドでリスト表示できます。 値: ● 次の形式の文字列: **<service_type>:<service_name>:<interface>** 少なくとも **service_type** および **interface** を指定する必要があります。**service_name** は省略できます。 関連するオプション: ● cinder_os_region_name ● cinder_endpoint_template ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name ● cinder_store_password

設定オプション = デフォルト値	型	説明
cinder_endpoint_template = None	文字列値	cinder エンドポイントのテンプレートでサービスカタログの検索を上書きします。 このオプションを設定すると、サービスカタログから検索するのではなく、この値を使用して cinder エンドポイントが生成されます。**cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、および **cinder_store_password** が指定されている場合には、この値は無視されます。 この設定オプションが設定されている場合、**cinder_catalog_info** は無視されます。 値: ● cinder エンドポイントの URL テンプレート文字列。ここで、**%%(tenant)s** は現在のテナント (プロジェクト) 名に置き換えます。例:**http://cinder.openstack.example.org/v2/%%(tenant)s** 関連するオプション: ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name ● cinder_store_password ● cinder_catalog_info
cinder_enforce_multipath = False	ブール値	これが True に設定されている場合、multipathd が実行されていない場合、イメージ転送用のボリュームのアタッチが中断されます。それ以外の場合は、単一のパスにフォールバックします。 値: ● True か False 関連するオプション: ● cinder_use_multipath

設定オプション = デフォルト値	型	説明
cinder_http_retries = 3	整数値	失敗した http 呼び出しでの cinderclient のリトライ回数。 何らかのエラーによって呼び出しに失敗すると、cinderclient は数秒スリープした後に呼び出しを指定の回数までリトライします。 値: ● 正の整数 関連するオプション: ● なし
cinder_mount_point_base = /var/lib/glance/mnt	文字列値	glance ノードで NFS ボリュームがマウントされるディレクトリー。 値: ● マウントポイントの絶対パスを表す文字列。
cinder_os_region_name = None	文字列値	サービスカタログから cinder サービスを検索する際のリージョン名。 これは、**cinder_catalog_info** を使用してエンドポイントを決定する場合にのみ使用されます。これが設定されている場合、このノードによる cinder エンドポイントの検索は、指定のリージョンにフィルターされます。これは、カタログに複数のリージョンがリスト表示される場合に役立ちます。これが設定されていない場合、エンドポイントはすべてのリージョンから検索されます。 値: ● 有効なリージョン名である文字列。 関連するオプション: ● cinder_catalog_info

設定オプション = デフォルト値	型	説明
cinder_state_transition_timeout = 300	整数値	cinder ボリュームの遷移が完了するまで待機する時間 (秒単位)。 ボリュームデータの読み取り/書き込みを行うために、cinder ボリュームを作成、削除、または glance ノードにアタッチすると、ボリュームの状態が変更されます。たとえば、新規作成されたボリュームのステータスは、作成プロセスの完了後に **creating** から **available** に変わります。このオプションは、ステータス変更を待機する最大時間を指定します。待機時間がタイムアウトするか、ステータスが予期しない値に変更される場合 (**error**など)、イメージの作成は失敗します。 値: ● 正の整数 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
cinder_store_auth_address = None	文字列値	cinder 認証サービスがリッスンしているアドレス。 **cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、および **cinder_store_password**オプションのすべてが指定された場合、指定した値は常に認証に使用されます。これは、イメージサービスに固有のプロジェクト/テナントにイメージボリュームを保存することにより、ユーザーからイメージボリュームを隠すのに役立ちます。また、ユーザーは glance の ACL の制御下で、他のプロジェクト間でイメージボリュームを共有できます。 これらのオプションのいずれかが設定されていない場合、cinder エンドポイントはサービスカタログから検索され、現在のコンテキストのユーザーとプロジェクトが使用されます。 値: ● 有効な認証サービスアドレス (例:**http://openstack.example.org/identity/v2.0**)。 関連するオプション: ● cinder_store_user_name ● cinder_store_password ● cinder_store_project_name
cinder_store_password = None	文字列値	cinder に対して認証を行うユーザーのパスワード。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのユーザーが使用されます。 値: ● **cinder_store_user_name**で指定したユーザーの有効なパスワード 関連するオプション: ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name

設定オプション = デフォルト値	型	説明
cinder_store_project_name = None	文字列値	cinder にイメージボリュームが保存されるプロジェクト名。 この設定オプションが設定されていない場合は、現在のコンテキストのプロジェクトが使用されます。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのプロジェクトが使用されます。 値: ● 有効なプロジェクト名 関連するオプション: ● **cinder_store_auth_address** ● **cinder_store_user_name** ● **cinder_store_password**
cinder_store_user_name = None	文字列値	cinder に対して認証を行うためのユーザー名。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのユーザーが使用されます。 値: ● 有効なユーザー名 関連するオプション: ● cinder_store_auth_address ● cinder_store_password ● cinder_store_project_name

設定オプション = デフォルト値	型	説明
cinder_use_multipath = False	ブール値	デプロイメントでマルチパスがサポートされるかどうかを識別するためのフラグ。 マルチパスがサポートされない場合は、False に設定します。 値: ● True か False 関連するオプション: ● cinder_enforce_multipath
cinder_volume_type = None	文字列値	cinder でのボリュームの作成に使用されるボリューム種別。 一部の cinder バックエンドには、ストレージの使用を最適化するための複数のボリューム種別が含まれる場合があります。このオプションを追加すると、運用者はイメージに対して最適化できる特定のボリューム種別を cinder で選択することができます。 これが設定されていない場合には、cinder 設定で指定したデフォルトのボリューム種別がボリュームの作成に使用されます。 値: ● cinder からの有効なボリューム種別 関連するオプション: ● なし **注記** NFS バックエンドに関連付けられた暗号化された volume_type は使用できません。NFS バックエンドに保存されている暗号化されたボリュームは、glance_store がそのボリュームに保存されているイメージデータの書き込みまたはアクセスを試みるたびに、例外を発生させます。Cinder 管理者に相談して、適切な volume_type を決定してください。

設定オプション = デフォルト値	型	説明
default_store = file	文字列値	イメージの保存に使用するデフォルトのスキーム。 イメージの保存に使用するデフォルトのスキームを表す文字列値を指定します。設定されていない場合、Glance は **file** をデフォルトのスキームとして使用して **file** ストアでイメージを保存します。 **注記** この設定オプションに指定する値は、**stores** 設定オプションで登録されたストアの有効なスキームである必要があります。 値: - file - Filesystem - http - https - swift - swift+http - swift+https - swift+config - rbd - cinder - vsphere - s3 関連するオプション: - stores **非推奨となったバージョン：** Rocky **理由:** このオプションは、''default_store'' 設定オプションと同様に機能する新しい設定オプション ''default_backend'' によって非推奨となりました。 このオプションは、U 開発サイクルで削除される予定です。

設定オプション = デフォルト値	型	説明
default_swift_reference = ref1	文字列値	デフォルトの Swift アカウント/バッキングストアパラメーターへの参照。 イメージストレージに swift アカウント/バッキングストアを使用するのに必要なデフォルトのパラメーターセットへの参照を表す文字列値を指定します。この設定オプションのデフォルト参照値は **ref1** です。この設定オプションはパラメーターを逆参照し、新しいイメージが追加されるたびに Swift ストレージバックエンドでのイメージ保管を容易にします。 値: ● 有効な文字列値 関連するオプション: ● なし
filesystem_store_chunk_size = 65536	整数値	チャンクサイズ (バイト単位)。 イメージファイルの読み取り/書き込み時に使用されるチャンクサイズ。この値を大きくするとスループットが向上しますが、大量のリクエストを処理する際にメモリー使用量が若干増加する場合もあります。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
filesystem_store_datadir = /var/lib/glance/images	文字列値	ファイルシステムのバックエンドストアがイメージを書き込むディレクトリー。 起動時に、Glance はディレクトリーが存在しない場合は作成し、**glance-api** が実行されるユーザーへの書き込みアクセスを検証します。書き込みアクセスが使用できない場合は、**BadStoreConfiguration** 例外が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 **注記** このディレクトリーは、ファイルシステムストアがストレージバックエンドとして使用される場合にのみ使用されます。**filesystem_store_datadir** または **filesystem_store_datadirs** オプションのいずれかを **glance-api.conf** で指定する必要があります。両方のオプションを指定すると、**BadStoreConfiguration** が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 値: ● ディレクトリーへの有効なパス 関連するオプション: ● **filesystem_store_datadirs** ● **filesystem_store_file_perm**

設定オプション = デフォルト値	型	説明
filesystem_store_datadirs = None	多値	ファイルシステムのバックエンドストアがイメージを書き込むディレクトリーとその優先順位のリスト。 **filesystem_store_datadir** 設定オプションで指定した単一のディレクトリーを使用する場合とは対照的に、ファイルシステムストアは複数のディレクトリーにイメージを格納するように設定できます。複数のディレクトリーを使用する場合、各ディレクトリーには、使用される優先順位を指定するオプションの優先度を指定できます。優先度は、ディレクトリーパスとコロンで組み合わされる整数で、値が大きいほど優先度が高くなります。2 つのディレクトリーの優先度が同じ場合は、空き容量が最も多いディレクトリーが使用されます。優先度を指定しないと、デフォルトでゼロに設定されます。 複数のストアディレクトリーを持つファイルシステムストアの設定に関する詳細は、https://docs.openstack.org/glance/latest/configuration/configuring.html を参照してください。 **注記** このディレクトリーは、ファイルシステムストアがストレージバックエンドとして使用される場合にのみ使用されます。**filesystem_store_datadir** または **filesystem_store_datadirs** オプションのいずれかを **glance-api.conf** で指定する必要があります。両方のオプションを指定すると、**BadStoreConfiguration** が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 値: - 以下の形式の文字列のリスト: - **<a valid directory path>:<optional integer priority>** 関連するオプション: - **filesystem_store_datadir** - **filesystem_store_file_perm**

設定オプション = デフォルト値	型	説明
filesystem_store_file_perm = 0	整数値	イメージファイルへのファイルアクセスパーミッション。 イメージデータへの目的のファイルアクセスパーミッションを設定します。これにより、Nova などの他のサービスがファイルシステムストアから直接イメージを使用できるようにすることができます。アクセスが付与されるサービスを実行中のユーザーは、作成されるファイルを所有しているグループのメンバーにすることができます。この設定オプションにゼロ以下の値を割り当てることは、デフォルトのパーミッションに変更が加えられないことを意味します。この値は、8 進数の数字としてデコードされます。 詳細は、https://docs.openstack.org/glance/latest/configuration/configuring.htmlのドキュメントを参照してください。 値: ● 有効なファイルアクセスパーミッション ● ゼロ ● 任意の負の整数 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
filesystem_store_metadata_file = None	文字列値	ファイルシステムストアのメタデータファイル。 ファイルシステムストアに関連付けられた場所と共に返されるメタデータが含まれるファイルへのパス。このオプションが設定されると、後で作成される新しいイメージにのみ使用されます。以前の既存のイメージには影響しません。 このファイルには、有効な JSON オブジェクトが含まれている必要があります。オブジェクトには、キー **ID** と **マウントポイント** が含まれている必要があります。両方のキーの値は文字列でなければなりません。 値: ● ストアメタデータファイルへの有効なパス 関連するオプション: ● なし
filesystem_thin_provisioning = False	ブール値	このバックエンドでシンプロビジョニングを有効にするかどうか。 この設定オプションを使用すると、ファイルシステムに null バイトシーケンスを実際に書き込まないという機能が有効になります。表示されるホールは、ファイルシステムによって自動的に null バイトとして解釈され、ストレージを実際に消費しません。この機能を有効にすると、ネットワーク経由で null バイトシーケンスが送信されないため、バックエンドのスペースを節約できるだけでなく、イメージのアップロードが高速化され、ネットワークトラフィックが節約されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
http_proxy_information = {}	dict 値	リモートサーバーへの接続に使用する http/https プロキシー情報。 この設定オプションは、リモートサーバーへの接続に使用する必要のある http/https プロキシー情報を指定します。プロキシー情報は、スキームとプロキシーのキー/値のペアにする必要があります (例:http:10.0.0.1:3128)。また、キーと値のペアをコンマで区切ることで、複数のスキームのプロキシーを指定することもできます (例: http:10.0.0.1:3128, https:10.0.0.1:1080)。 値: ● 上記のように スキーム: プロキシーペアのコンマ区切りリスト 関連するオプション: ● なし
https_ca_certificates_file = None	文字列値	CA バンドルファイルへのパス。 この設定オプションにより、運用者はカスタム認証局ファイルを使用してリモートサーバー証明書を検証できます。このオプションを設定すると、**https_insecure** オプションは無視され、指定された CA ファイルがサーバー証明書の認証に使用され、サーバーへのセキュアな接続が確立されます。 値: ● CA ファイルへの有効なパス 関連するオプション: ● https_insecure

設定オプション = デフォルト値	型	説明
https_insecure = True	ブール値	リモートサーバー証明書の検証を設定します。 この設定オプションはブール値を取り、リモートサーバーの証明書を検証するかどうかを決定します。True に設定すると、リモートサーバー証明書は検証されません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 このオプションは、**https_ca_certificates_file** が設定されている場合は無視されます。リモートサーバー証明書は、**https_ca_certificates_file** オプションを使用して指定されたファイルを使用して検証されます。 値: ● True ● False 関連するオプション: ● https_ca_certificates_file
rados_connect_timeout = 0	整数値	Ceph クラスターに接続する際のタイムアウト値。 この設定オプションは、Ceph クラスターに接続する際に使用されるタイムアウト値 (秒単位) を取ります。つまり、接続を閉じるまで glance-api が待機する時間を設定します。これにより、RBD への接続中に glance-api がハングアップするのを防ぐことができます。このオプションの値を 0 以下に設定すると、タイムアウトは設定されず、デフォルトの librados 値が使用されます。 以下の値を使用できます。 ● 任意の整数値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
`rbd_store_ceph_conf = `	文字列値	Ceph 設定ファイルへのパス。 この設定オプションは、使用する Ceph 設定ファイルへのパスを指定します。このオプションの値がユーザーにより設定されていない場合や、空の文字列に設定されている場合、librados は、デフォルトの Ceph 設定ファイルの場所を順に検索して標準の ceph.conf ファイルを読み取ります。詳細は、Ceph のドキュメントを参照してください。 **注記** Cephx 認証を使用する場合、このファイルの client.<USER>セクションに適切なキーリングへの参照を含める必要があります。 注 2: このオプションを空白のままにすると (デフォルト)、使用される実際の Ceph 設定ファイルは、使用されている librados のバージョンによって変わる可能性があります。有効になっている設定ファイルを正確に把握しておくことが重要な場合は、このオプションを使用してここにファイルを指定できます。 以下の値を使用できます。 ● 設定ファイルへの有効なパス 関連するオプション: ● rbd_store_user

設定オプション = デフォルト値	型	説明
rbd_store_chunk_size = 8	整数値	RADOS イメージをチャンクに分割する際のサイズ (メガバイト単位)。 Glance イメージをチャンクに分割する際のサイズ (メガバイト単位) を表す整数値を指定します。デフォルトのチャンクサイズは 8 メガバイトです。最適なパフォーマンスを得るには、値は 2 の累乗でなければなりません。 Ceph の RBD オブジェクトストレージシステムを Glance イメージを保存するためのストレージバックエンドとして使用する場合、イメージはこのオプションを使用して設定されるサイズのオブジェクトに分割されます。これらの分割されたオブジェクトは、分散ブロックデータストア全体に保存され、Glance に使用されます。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし
rbd_store_pool = images	文字列値	イメージが保存される RADOS プール。 RBD を Glance イメージを保存するためのストレージバックエンドとして使用する場合、イメージは **プール** へのオブジェクト (イメージのチャンク) の論理的なグループ化により保存されます。各プールは、含めることのできる配置グループの数で定義されます。使用されるデフォルトのプールは **イメージ** です。 RBD ストレージバックエンドの詳細は、http://ceph.com/planet/how-data-is-stored-in-ceph-cluster/を参照してください。 以下の値を使用できます。 ● 有効なプール名 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
rbd_store_user = None	文字列値	認証する RADOS ユーザー。 この設定オプションは、認証する RADOS ユーザーを設定します。このオプションは RADOS 認証が有効になっている場合にのみ必要で、ユーザーが Cephx 認証を使用している場合にのみ適用されます。このオプションの値がユーザーによって設定されていない場合や、None に設定されている場合には、デフォルト値が選択されます。これは、rbd_store_ceph_conf の client. セクションに基づきます。 以下の値を使用できます。 ● 有効な RADOS ユーザー 関連するオプション: ● rbd_store_ceph_conf
rbd_thin_provisioning = False	ブール値	このバックエンドでシンプロビジョニングを有効にするかどうか。 この設定オプションを使用すると、RBD バックエンドに null バイトシーケンスを実際に書き込まないという機能が有効になります。表示されるホールは、Ceph によって自動的に null バイトとして解釈され、ストレージを実際に消費しません。この機能を有効にすると、ネットワーク経由で null バイトシーケンスが送信されないため、バックエンドのスペースを節約できるだけでなく、イメージのアップロードが高速化され、ネットワークトラフィックが節約されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
rootwrap_config = /etc/glance/rootwrap.conf	文字列値	root としてコマンドを実行するのに使用する rootwrap 設定ファイルへのパス。 cinder ストアでは、イメージボリュームを操作するのに root 権限が必要です (iSCSI/FC ボリュームへの接続、ボリュームデータの読み取り/書き込みのためなど)。設定ファイルでは、cinder ストアおよび os-brick ライブラリーによる必要なコマンドを許可する必要があります。 値: ● rootwrap 設定ファイルへのパス 関連するオプション: ● なし
s3_store_access_key = None	文字列値	S3 クエリートークンアクセスキー。 この設定オプションは、Amazon S3 または S3 互換ストレージサーバーで認証するためのアクセスキーを取得します。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。 値: ● 適切な権限を持つユーザーのアクセスキーである任意の文字列値 関連するオプション: ● s3_store_host ● s3_store_secret_key

設定オプション = デフォルト値	型	説明
s3_store_bucket = None	文字列値	Glance データの保存に使用される S3 バケット。 この設定オプションは、Glance イメージが S3 に保存される場所を指定します。**s3_store_create_bucket_on_put** を true に設定すると、バケットが存在しなくても自動的に作成されます。 値: ● 任意の文字列値 関連するオプション: ● s3_store_create_bucket_on_put ● s3_store_bucket_url_format
s3_store_bucket_url_format = auto	文字列値	オブジェクトを決定するために使用される S3 呼び出し形式。 この設定オプションは、S3 バケット内のオブジェクトのアドレスを指定するために使用されるアクセスモデルを取ります。 注記: **path** スタイルでは、オブジェクトのエンドポイントは https://s3.amazonaws.com/bucket/example.img のようになります。**virtual** スタイルでは、オブジェクトのエンドポイントは https://bucket.s3.amazonaws.com/example.img のようになります。バケット名で DNS 命名規則に従わない場合、パススタイルでオブジェクトを取得できますが、仮想スタイルでは取得できません。 値: ● **auto**、**virtual**、または **path** の任意の文字列値 関連するオプション: ● s3_store_bucket

設定オプション = デフォルト値	型	説明
s3_store_create_bucket_on_put = False	ブール値	S3 が新しいバケットを作成する必要があるかどうかを決定します。 この設定オプションはブール値を取り、Glance が S3 に新しいバケットを作成する必要があるかどうかを示します (存在しない場合)。 値: ● 任意のブール値 関連するオプション: ● なし
s3_store_host = None	文字列値	S3 サーバーがリッスンしているホスト。 この設定オプションは、S3 または S3 互換ストレージサーバーのホストを設定します。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。ホストには、DNS 名 (s3.amazonaws.com、my-object-storage.com など) または IP アドレス (127.0.0.1) を含めることができます。 値: ● 有効な DNS 名 ● 有効な IPv4 アドレス 関連するオプション: ● s3_store_access_key ● s3_store_secret_key

設定オプション = デフォルト値	型	説明
s3_store_large_object_chunk_size = 10	整数値	パーツのアップロード時に S3 が使用するマルチパートアップロードパーツのサイズ (MB 単位)。 この設定オプションは、マルチパートアップロードのイメージ分割サイズを MB 単位で取得します。 注記: 分割できるイメージは 10,000 個までです。 値: ● 任意の正の整数値 (5M 以上である必要があります) 関連するオプション: ● s3_store_large_object_size ● s3_store_thread_pools
s3_store_large_object_size = 100	整数値	S3 がイメージファイルのチャンク化を開始し、S3 でマルチパートアップロードを実行する必要があるサイズ (MB 単位)。 この設定オプションは、イメージをそのまま S3 にアップロードするか、分割するか (マルチパートアップロード) を決定するために MB 単位でしきい値を取得します。 注記: 分割できるイメージは 10,000 個までです。 値: ● 任意の正の整数値 関連するオプション: ● s3_store_large_object_chunk_size ● s3_store_thread_pools

設定オプション = デフォルト値	型	説明
s3_store_secret_key = None	文字列値	S3 クエリートークンの秘密鍵。 この設定オプションは、Amazon S3 または S3 互換ストレージサーバーで認証するための秘密鍵を受け取ります。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。 値: ● **s3_store_host** オプションを使用して指定されたアクセスキーに対応する秘密鍵である任意の文字列値 関連するオプション: ● s3_store_host ● s3_store_access_key
s3_store_thread_pools = 10	整数値	S3 でマルチパートアップロードを実行するためのスレッドプールの数。 この設定オプションは、マルチパートアップロードの実行時にスレッドプールの数を取得します。 値: ● 任意の正の整数値 関連するオプション: ● s3_store_large_object_size ● s3_store_large_object_chunk_size

設定オプション = デフォルト値	型	説明
stores = ['file', 'http']	リスト値	有効な Glance ストアのリスト。 ディスクイメージを格納するために使用するストレージバックエンドを、コンマ区切りリストとして登録します。Glance でのディスクイメージの保存用に有効化されたデフォルトのストアは、**file** と **http** です。 値: ● 以下が含まれるコンマ区切りリスト: ○ file ○ http ○ swift ○ rbd ○ cinder ○ vmware ○ s3 関連するオプション: ● default_store **非推奨となったバージョン：** Rocky **理由:** このオプションは、異なるスキームの複数のバックエンドストアを設定するのに役立つ新たな設定オプション ''enabled_backends'' によって非推奨となりました。 このオプションは、U 開発サイクルで削除される予定です。

設定オプション = デフォルト値	型	説明
swift_buffer_on_upload = False	ブール値	Swift にアップロードする前にイメージのセグメントをバッファー処理します。 swift へのアップロード中に Glance がイメージデータをディスクにバッファーするかどうかを指定するブール値を指定します。これにより、Glance はエラー時にアップロードを再開できます。 注意: このオプションを有効にすると、API ノードのディスク使用量が増加するので、細心の注意を払う必要があります。ファイルシステムの設定状況によっては、バッファー処理に使用されるディスク領域により、glance イメージキャッシュに利用可能な実際のディスク容量が減る可能性があることに注意してください。ディスク使用率は、式**swift_store_large_object_chunk_size** * **workers** * 1000 に従って上限が設定されます。 値: ● True ● False 関連するオプション: ● swift_upload_buffer_dir
swift_store_admin_tenants = []	リスト値	管理者アクセスが付与されるテナントのリスト。 これは、マルチテナントモードの Glance によって作成されたすべての Swift コンテナーでの読み取り/書き込みアクセスが付与されるテナントのリストです。デフォルト値は空のリストです。 値: ● Keystone プロジェクト/テナントの UUID を表す文字列のコンマ区切りリスト 関連するオプション: ● なし
swift_store_auth_address = None	文字列値	Swift 認証サービスがリッスンしているアドレス。

設定オプション = デフォルト値	型	説明
swift_store_auth_insecure = False	ブール値	サーバー証明書の検証を設定します。 このブール値は、サーバー証明書を検証するかどうかを決定します。このオプションが True に設定されている場合、swiftclient は認証時に有効な SSL 証明書かどうかをチェックしません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 値: ● True ● False 関連するオプション: ● swift_store_cacert
swift_store_auth_version = 2	文字列値	使用する認証サービスのバージョン。有効なバージョンは、keystone の場合は 2 および 3 で、swauth および rackspace の場合は 1(非推奨) です。
swift_store_cacert = None	文字列値	CA バンドルファイルへのパス。 この設定オプションにより、運用者は Swift に接続する際の SSL 検証用のカスタム認証局ファイルへのパスを指定できます。 値: ● CA ファイルへの有効なパス 関連するオプション: ● swift_store_auth_insecure

設定オプション = デフォルト値	型	説明
swift_store_config_file = None	文字列値	swift アカウント設定が含まれるファイルへの絶対パス。 設定済みの各 Swift アカウント/バッキングストアへの参照を持つ設定ファイルへのパスを表す文字列値を含めます。デフォルトでは、ファイルパスが指定されておらず、カスタマイズされた Swift 参照は無効になっています。データベースへの認証情報の保存を回避するため、このオプションを設定することは、イメージストレージに Swift ストレージバックエンドを使用する際に強く推奨されます。 **注記** **swift_store_multi_tenant** を **True** に設定した場合は、このオプションを設定しないでください。 値: ● glance-api ノードの絶対パスを表す文字列値 関連するオプション: ● swift_store_multi_tenant

設定オプション = デフォルト値	型	説明
swift_store_container = glance	文字列値	イメージを保存するための単一のコンテナーの名前/複数のコンテナーの名前の接頭辞 1 つのコンテナーを使用してイメージを保存する場合、この設定オプションは、すべてのイメージの保存に使用される Glance アカウント内のコンテナーを示します。複数のコンテナーがイメージの保存に使用される場合、これはすべてのコンテナーの名前の接頭辞になります。単一/複数コンテナーの使用は、設定オプション **swift_store_multiple_containers_seed** を使用して制御できます。 複数のコンテナーを使用する場合、コンテナーの名前は、この設定オプションに設定された値をベースに、イメージ UUID の最初の N 文字をアンダースコアで区切られた接尾辞として指定されます (N は **swift_store_multiple_containers_seed**によって指定されます)。 例: シードが 3 で swift_store_container = **glance** と設定されている場合には、UUID が**fdae39a1-bac5-4238-aba4-69bcc726e848** のイメージは **glance_fda** のコンテナーに配置されます。コンテナー名の作成時に、UUID のすべてのダッシュは含まれますが、文字の制限にはカウントされないため、N=10 の場合はコンテナー名は **glance_fdae39a1-ba** になります。 値: ● 単一のコンテナーを使用する場合、この設定オプションは Glance の Swift アカウントの有効な任意の swift コンテナーの名前の文字列にすることができます。 ● 複数のコンテナーを使用する場合、この設定オプションは、 Swift により適用されるコンテナーの命名ルールを満たす限り任意の文字列になります。**swift_store_multiple_containers_seed** の値も考慮する必要があります。 関連するオプション: ● **swift_store_multiple_containers_seed** ● **swift_store_multi_tenant** ● **swift_store_create_container_on_put**

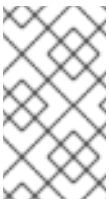
設定オプション = デフォルト値	型	説明
swift_store_create_container_on_put = False	ブール値	イメージのアップロード時に、コンテナーを作成します (まだ存在しない場合)。 イメージのアップロード時に、対応するコンテナーが存在しない場合は、この設定オプションが True に設定されている場合に作成されます。デフォルトでは、コンテナーは作成されません。この動作は、単一および複数コンテナーモードの両方に適用されます。 値: ● True ● False 関連するオプション: ● なし
swift_store_endpoint = None	文字列値	Swift バックエンドストレージに使用する URL エンドポイント。 Glance イメージを Swift ストアに格納するために使用する URL エンドポイントを表す文字列値を指定します。デフォルトでは、エンドポイントは設定されず、**auth** によって返されるストレージ URL が 使用されます。**swift_store_endpoint** を使用してエンドポイントを設定すると、ストレージ URL を上書きされ、Glance イメージストレージに使用されます。 **注記** URL には、コンテナーまでのパスを含める必要がありますが、コンテナーそのものは除外します。オブジェクトの場所は、コンテナーおよびオブジェクトを設定した URL に追加すると得られます。 値: ● Swift コンテナーまでの有効な URL パスを表す文字列値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_endpoint_type = publicURL	文字列値	Swift サービスのエンドポイント種別。 この文字列値は、Swift エンドポイントの取得に使用するエンドポイント種別を示します。エンドポイント種別により、ユーザーが実行できるアクション (たとえば、ストアの読み取りおよび書き込み) が決定されます。この設定は、swift_store_auth_version が 1 よりも大きい場合にのみ使用されます。 値: ● publicURL ● adminURL ● internalURL 関連するオプション: ● swift_store_endpoint
swift_store_expire_soon_interval = 60	整数値	現在のトークンの有効期限が過ぎる前に新しいトークンが要求される可能性があるウィンドウのサイズを定義する時間 (秒単位)。 通常、Swift ストレージドライバーは、現在のトークンの有効期限が過ぎた時に新しいトークンを取得して、Swift へのアクセスを継続させます。ただし、処理中にトークンが期限切れになると、一部の Swift トランザクション (イメージのセグメントのアップロードなど) が適切に復元されないことがあります。 したがって、現在のトークンの有効期限前に新しいトークンを取得することにより、トランザクションを試行する前にトークンが失効したり期限切れに近づいたりしないようにします。デフォルトでは、Swift ストレージドライバーは、現在のトークンの有効期限の 60 秒前から新しいトークンを要求します。 値: ● ゼロ ● 正の整数値 関連するオプション: ● なし
swift_store_key = None	文字列値	Swift 認証サービスに対して認証するユーザーの認証鍵。

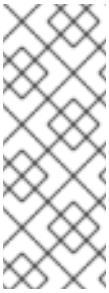
設定オプション = デフォルト値	型	説明
swift_store_large_object_chunk_size = 200	整数値	イメージデータがセグメント化される際のセグメントの最大サイズ (MB 単位)。 Swift クラスターによって適用される上限を上回るイメージをアップロードするためにイメージデータがセグメント化される場合、イメージデータはこの設定オプションで指定されるサイズ以下のセグメントに分割されます。詳細は、**swift_store_large_object_size** を参照してください。 たとえば、**swift_store_large_object_size** が 5 GB で、**swift_store_large_object_chunk_size** が 1 GB の場合、サイズが 6.2 GB のイメージは 7 セグメントに分割されます。この場合、最初の 6 つのセグメントのサイズは 1 GB で、7 番目のセグメントは 0.2 GB になります。 値: ● 該当する Swift クラスターによって適用される大型オブジェクト制限以下の正の整数。 関連するオプション: ● **swift_store_large_object_size**

設定オプション = デフォルト値	型	説明
swift_store_large_object_size = 5120	整数値	サイズしきい値 (MB 単位)。これを超えると、Glance はイメージデータのセグメント化を開始します。 Swift では、アップロードされる単一のオブジェクトのサイズに上限があります。デフォルトでは、これは 5 GB です。この制限を超えるオブジェクトをアップロードするには、オブジェクトをマニフェストファイルと関連付けられる複数の小さなオブジェクトにセグメント化します。詳細は、https://docs.openstack.org/swift/latest/overview_large_objects.htmlを参照してください。 この設定オプションは、サイズのしきい値を指定します。これを超えると、Swift ドライバーはイメージデータを複数の小さなファイルにセグメント化する処理を開始します。現在、Swift ドライバーは Dynamic Large Objects の作成のみをサポートしています。 **注記** これは、該当する Swift クラスターによって適用される大型オブジェクトの制限を考慮して設定する必要があります。 値: ● 該当する Swift クラスターによって適用される大型オブジェクト制限以下の正の整数。 関連するオプション: ● **swift_store_large_object_chunk_size**

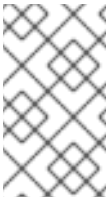
設定オプション = デフォルト値	型	説明
swift_store_multi_tenant = False	ブール値	イメージをテナントの Swift アカウントに保存します。 これにより、マルチテナントストレージモードが有効になり、Glance イメージがテナント固有の Swift アカウントに保存されます。これが無効になっていると、Glance はすべてのイメージを独自のアカウントに保存します。マルチテナントストアの詳細については、https://wiki.openstack.org/wiki/GlanceSwiftTenantSpecificStorageを参照してください。 **注記** マルチテナントの swift ストアを使用している場合は、**swift_store_config_file** オプションで swift 設定ファイルを設定しないようにしてください。 値: • True • False 関連するオプション: • swift_store_config_file

設定オプション = デフォルト値	型	説明
swift_store_multiple_containers_seed = 0	整数値	イメージの保存に使用するコンテナーの数を示すシード。 単一テナントストアを使用する場合は、イメージを 1 つ以上のコンテナーに保存できます。0 に設定すると、すべてのイメージが 1 つのコンテナーに保存されます。1 から 32 の整数値に設定すると、イメージの保存に複数のコンテナーが使用されます。この設定オプションにより、作成されるコンテナーの数が決まります。使用されるコンテナーの合計数は 16^N となるため、この設定オプションが 2 に設定されている場合は、イメージの保存に 16^2=256 のコンテナーが使用されます。 命名規則に関する詳細は、**swift_store_container** を参照してください。複数のコンテナーの使用についての詳細は、https://specs.openstack.org/openstack/glance-specs/specs/kilo/swift-store-multiple-containers.htmlを参照してください。  **注記** このオプションは、swift_store_multi_tenant が無効になっている場合にのみ使用されます。 値: ● 32 以下の負の値ではない整数 関連するオプション: ● **swift_store_container** ● **swift_store_multi_tenant** ● **swift_store_create_container_on_put**

設定オプション = デフォルト値	型	説明
swift_store_region = None	文字列値	Glance が使用する Swift エンドポイントのリージョン。 Glance がイメージの保存のために接続できる Swift リージョンを表す文字列値を指定します。デフォルトでは、リージョンは設定されません。 Glance がストレージバックエンドとして Swift を使用して複数のエンドポイントを持つ特定のテナントのイメージを保存する場合、**swift_store_region** で Swift リージョンを設定すると、Glance は単一リージョン接続ではなく、指定したリージョンの Swift に接続できるようになります。 このオプションは、単一テナントおよびマルチテナントストレージの両方に対して設定できます。 **注記** **swift_store_region** を使用したリージョンの設定はテナント固有のもので、テナントが異なるリージョンにまたがる複数のエンドポイントを持つ **場合にのみ** 必要です。 値: ● 有効な Swift リージョンを表す文字列値。 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_retry_get_count = 0	整数値	Swift のダウンロードがリトライされる回数。これを超えると、リクエストは失敗します。 エラーが発生する前にイメージのダウンロードをリトライする必要のある回数を表す整数値を指定します。デフォルト値はゼロです (イメージのダウンロードに失敗した場合、リトライされません)。正の整数値に設定すると、**swift_store_retry_get_count** により、ダウンロードの失敗時にこの回数ダウンロードが試みられます。これを超えると、エラーメッセージが送信されます。 値: ● ゼロ ● 正の整数値 関連するオプション: ● なし
swift_store_service_type = object-store	文字列値	使用する Swift サービスの種別。 Swift バックエンドストレージの使用中、イメージの保存に使用するサービス種別を表す文字列値を指定します。デフォルトのサービス種別は **object-store** に設定されます。  **注記** **swift_store_auth_version** が 2 に設定されている場合、この設定オプションの値は **object-store** である必要があります。より新しいバージョンの Keystone または別の認証スキームを使用している場合は、このオプションを変更できます。 値: ● Swift ストレージ用の有効なサービス種別を表す文字列。 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_ssl_compression = True	ブール値	HTTPS Swift リクエストの SSL レイヤーの圧縮。 SSL レイヤーでイメージの HTTPS Swift リクエストを圧縮するかどうかを決定するブール値を指定します。デフォルトでは、圧縮は有効になっています。 Swift を Glance イメージストレージのバックエンドストアとして使用する場合は、このオプションを使用して HTTPS Swift リクエストの SSL レイヤーの圧縮を設定できます。False に設定すると、HTTPS Swift リクエストの SSL レイヤーの圧縮は無効になります。このオプションを無効にすると、すでに圧縮された形式のイメージ (qcow2 など) に関するパフォーマンスが向上します。 値: ● True ● False 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_use_trusts = True	ブール値	マルチテナント Swift ストアの信頼を使用します。 マルチテナントストアが使用されている場合に、このオプションは、各追加/取得リクエストに対して信頼を作成するように Swift ストアに指示します。信頼を使用すると、Swift ストアは、データのアップロードまたはダウンロード中に、認証トークンの失効により発生する可能性がある問題を回避できます。 デフォルトでは、**swift_store_use_trusts** は **True**に設定されています (信頼の使用が有効)。**False** に設定すると、代わりに Swift 接続にユーザートークンが使用され、信頼の作成のオーバーヘッドがなくなります。  **注記** このオプションは、**swift_store_multi_tenant** が **True**に設定されている場合のみ考慮されます。 値: - True - False 関連するオプション: - swift_store_multi_tenant
swift_store_user = None	文字列値	Swift 認証サービスに対して認証するユーザー。

設定オプション = デフォルト値	型	説明
swift_upload_buffer_dir = None	文字列値	Swift にアップロードする前にイメージセグメントをバッファー処理するディレクトリー。 イメージセグメントが swift にアップロードされる前に一時的にバッファー処理される、glance ノード上のディレクトリーへの絶対パスを表す文字列値を指定します。 注記: ● これは、設定オプション **swift_buffer_on_upload** が True に設定されている場合にのみ必要です。 ● このディレクトリーは、**swift_store_large_object_chunk_size** と、特定の Glance ノードによって同時にアップロードできるイメージの最大数を念頭に置いてプロビジョニングする必要があります。 値: ● 絶対ディレクトリーパスを表す文字列値 関連するオプション: ● swift_buffer_on_upload ● swift_store_large_object_chunk_size
vmware_api_retry_count = 10	整数値	VMware API のリトライ回数。 この設定オプションは、接続関連の問題またはサーバー API 呼び出しのオーバーロード時に VMware ESX/VC サーバー API をリトライする回数を指定します。**retry forever** は指定できません。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
vmware_ca_file = None	文字列値	CA バンドルファイルへの絶対パス。 この設定オプションにより、運用者はカスタム認証局ファイルを使用して ESX/vCenter 証明書を検証できます。 このオプションを設定すると、"vmware_insecure" オプションは無視され、指定された CA ファイルが ESX/vCenter サーバー証明書の認証に使用され、サーバーへのセキュアな接続が確立されます。 以下の値を使用できます。 ● CA ファイルへの有効な絶対パスである任意の文字列 関連するオプション: ● vmware_insecure
vmware_datastores = None	多値	イメージを保存できるデータストア。 この設定オプションは、VMWare ストアのバックエンドでイメージを保存できるデータストアを指定します。このオプションは、複数のデータストアを指定するために複数回指定できます。データストア名は、データセンターのパスの後に: で区切って指定する必要があります。オプションの重みをデータストア名の後に再度: で区切って指定して、優先順位を指定できます。したがって、必要な形式は <datacenter_path>:<datastore_name>:<optional_weight> になります。 イメージを追加する場合、イメージサイズが既知の場合に利用可能な空き領域が十分にない場合を除き、最も高い重みを持つデータストアが選択されます。重みが指定されない場合、これはゼロであると仮定され、ディレクトリーは最後に選択のために考慮されます。複数のデータストアの重みが同じ場合は、利用可能な空き領域が最も多いデータストアが選択されます。 以下の値を使用できます。 ● <datacenter_path>:<datastore_name>:<optional_weight>の形式の任意の文字列 関連するオプション:* なし

設定オプション = デフォルト値	型	説明
vmware_insecure = False	ブール値	ESX/vCenter サーバー証明書の検証を設定します。 この設定オプションはブール値を取り、ESX/vCenter サーバーの証明書を検証するかどうかを決定します。このオプションが True に設定されている場合、ESX/vCenter サーバー証明書は検証されません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 このオプションは、vmware_ca_file オプションが設定されている場合は無視されます。その場合、ESX/vCenter サーバー証明書は、"vmware_ca_file" オプションで指定したファイルを使用して検証されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● vmware_ca_file
vmware_server_host = None	ホストのアドレス値	ESX/ESXi または vCenter サーバーターゲットシステムのアドレス。 この設定オプションは、ESX/ESXi または vCenter サーバーターゲットシステムのアドレスを設定します。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。アドレスには、IP アドレス (127.0.0.1) または DNS 名 (www.my-domain.com) を含めることができます。 以下の値を使用できます。 ● 有効な IPv4 アドレスまたは IPv6 アドレス ● 有効な DNS 名 関連するオプション: ● vmware_server_username ● vmware_server_password

設定オプション = デフォルト値	型	説明
vmware_server_password = None	文字列値	サーバーのパスワード。 この設定オプションは、VMware ESX/ESXi または vCenter サーバーとの認証用のパスワードを取ります。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。 以下の値を使用できます。 ● "vmware_server_username" オプションを使用して指定されたユーザー名に対応するパスワードである任意の文字列 関連するオプション: ● vmware_server_host ● vmware_server_username
vmware_server_username = None	文字列値	サーバーのユーザー名。 この設定オプションは、VMware ESX/ESXi または vCenter サーバーとの認証用のユーザー名を取ります。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。 以下の値を使用できます。 ● 適切な権限を持つユーザーのユーザー名である任意の文字列 関連するオプション: ● vmware_server_host ● vmware_server_password

設定オプション = デフォルト値	型	説明
vmware_store_image_dir = /openstack_glance	文字列値	データストアで glance イメージが保存されるディレクトリー。 この設定オプションは、VMware データストアで glance イメージが保存されるディレクトリーへのパスを指定します。このオプションが設定されていない場合には、glance イメージが保存されるデフォルトのディレクトリーは openstack_glance になります。 以下の値を使用できます。 ● ディレクトリーへの有効なパスである任意の文字列 関連するオプション: ● なし
vmware_task_poll_interval = 5	整数値	VMware ESX/VC サーバーで呼び出されたリモートタスクをポーリングする間隔 (秒単位)。 この設定オプションは、VMWare ESX/VC サーバー API 呼び出しの一部として、処理中の非同期タスクをポーリングする際のスリープ時間 (秒単位) を取ります。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし

3.2.4. oslo_concurrency

以下の表は、**/etc/glance/glance-scrubber.conf** ファイルの **[oslo_concurrency]** グループで利用可能なオプションの概要をまとめたものです。

表3.28 oslo_concurrency

設定オプション = デフォルト値	型	説明
disable_process_locking = False	ブール値	プロセス間ロックを有効または無効にします。

設定オプション = デフォルト値	型	説明
lock_path = None	文字列値	ロックファイルに使用するディレクトリー。セキュリティー上、指定したディレクトリーは、ロックが必要なプロセスを実行しているユーザーのみが書き込み可能である必要があります。デフォルトは環境変数 OSLO_LOCK_PATH です。外部ロックを使用する場合は、ロックパスを設定する必要があります。

3.2.5. oslo_policy

以下の表は、**/etc/glance/glance-scrubber.conf** ファイルの **[oslo_policy]** グループで利用可能なオプションの概要をまとめたものです。

表3.29 oslo_policy

設定オプション = デフォルト値	型	説明
enforce_new_defaults = False	ブール値	このオプションは、ポリシーを評価するときに古い非推奨のデフォルトを使用するかどうかを制御します。**True** の場合、古い非推奨のデフォルトは評価されません。これは、既存のトークンが古いデフォルトで許可されているが、新しいデフォルトで許可されていない場合、許可されないことを意味します。新しいデフォルトと **scope_type** の利点を一緒に得ることができるように、このフラグを **enforce_scope** フラグと一緒に有効にすることを推奨します
enforce_scope = False	ブール値	このオプションは、ポリシーを評価する際にスコープを適用するかどうかを制御します。**True** の場合、リクエストで使用されるトークンのスコープは、適用されるポリシーの **scope_types** と比較されます。スコープが一致しない場合、**InvalidScope** 例外が発生します。**False** の場合、ポリシーが一致しないスコープで呼び出されていることを運用者に通知するメッセージが記録されます。
policy_default_rule = default	文字列値	デフォルトルール。要求されたルールが見つからない場合に適用されます。
policy_dirs = ['policy.d']	多値	ポリシー設定ファイルが保存されるディレクトリー。これは、config_dir オプションで定義される検索パス内の任意のディレクトリーへの相対パスか、絶対パスにすることができます。このディレクトリーの検索には、policy_file で定義されたファイルが存在している必要があります。存在しないディレクトリーまたは空のディレクトリーは無視されます。

設定オプション = デフォルト値	型	説明
policy_file = policy.yaml	文字列値	ロールを指定のサービスのパーミッションにマッピングするファイルの相対パスまたは絶対パス。相対パスは、このオプションを設定する設定ファイルに対する相対パスとして指定する必要があります。
remote_content_type = application/x-www-form-urlencoded	文字列値	REST ベースのポリシーチェックのデータを送受信するコンテンツ種別
remote_ssl_ca_crt_file = None	文字列値	REST ベースのポリシーチェック用の認証局証明書ファイルの絶対パス
remote_ssl_client_crt_file = None	文字列値	REST ベースのポリシーチェック用のクライアント証明書の絶対パス
remote_ssl_client_key_file = None	文字列値	REST ベースのポリシーチェック用のクライアント鍵ファイルの絶対パス
remote_ssl_verify_server_crt = False	ブール値	REST ベースのポリシーチェックのサーバーアイデンティティー検証

3.3. GLANCE-CACHE.CONF

このセクションには、**/etc/glance/glance-cache.conf** ファイルのオプションが含まれます。

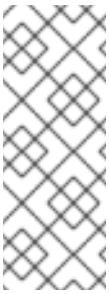
3.3.1. DEFAULT

以下の表は、**/etc/glance/glance-cache.conf** ファイルの **[DEFAULT]** グループで利用可能なオプションの概要をまとめたものです。

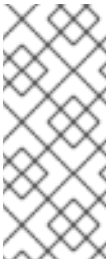
.

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
allow_additional_image_properties = True	ブール値	ユーザーがイメージに追加/カスタムの属性を追加できるようにします。 Glance は、すべてのイメージに表示される属性の標準セット () スキーマ) を定義します。これらのプロパティーは、**base properties** とも呼ばれます。Glance では、これらの属性に加え、カスタムプロパティーをイメージに追加できます。これらは、**additional properties** と呼ばれています。 デフォルトでは、この設定オプションは **True** に設定され、ユーザーは追加のプロパティーを追加できます。イメージに追加できる追加プロパティーの数は、**image_property_quota** 設定オプションで制御できます。 値: ● True ● False 関連するオプション: ● image_property_quota **非推奨:** Ussuri **理由:** このオプションは冗長です。**image_property_quota** 設定オプションを使用して、カスタムイメージプロパティーの使用を制御します。このオプションは、Victoria の開発サイクル中に削除される予定です。

設定オプション = デフォルト値	型	説明
api_limit_max = 1000	整数値	要求によって返されることのできる最大結果数。 **limit_param_default** のヘルプテキストで説明されているように、リクエストによっては複数の結果が返される場合があります。返される結果の数は、要求の **limit** パラメーターまたは **limit_param_default** 設定オプションで管理されます。いずれの場合もこの値が、この設定オプションで定義される絶対最大値よりも大きくすることはできません。この値より大きいすべてのものは、ここで定義される最大値にトリミングされます。  **注記** これを非常に大きな値に設定すると、データベースのクエリーが遅くなり、応答時間が長くなる可能性があります。これを非常に低い値に設定すると、ユーザーエクスペリエンスが低下してしまう可能性があります。 値: ● 任意の正の整数 関連するオプション: ● limit_param_default
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
digest_algorithm = sha256	文字列値	デジタル署名に使用するダイジェストアルゴリズム。 デジタル署名の生成に使用するダイジェストアルゴリズムを表す文字列値を指定します。デフォルトでは **sha256** が使用されます。 プラットフォームの OpenSSL バージョンでサポートされる利用可能なアルゴリズムのリストを取得するには、コマンド **openssl list-message-digest-algorithms** を実行します。たとえば、**sha1**、**sha256**、および **sha512** です。  **注記** **digest_algorithm** は Glance のイメージの署名および検証に関連しません。これは、証明書ファイルとキーファイル検証の一部として汎用一意識別子 (UUID) に署名するために使用されます。 値: ● OpenSSL メッセージダイジェストアルゴリズム ID 関係オプション: ● なし
enabled_import_methods = ['glance-direct', 'web-download', 'copy-image']	リスト値	有効なイメージのインポートメソッドの一覧 'glance-direct', 'copy-image' and 'web-download' are enabled by default. Related options: ** [DEFAULT]/node_staging_uri

設定オプション = デフォルト値	型	説明
enforce_secure_rbac = False	ブール値	OpenStack 全体で使用される一般的なペルソナ定義に基づいて API アクセスを強制します。このオプションを有効にすると、**member** ロールの背後で、プライベートイメージの作成や共有イメージのステータスの更新など、プロジェクト固有の読み取り/書き込み操作が形式化されます。また、**reader** ロールの背後で、プロジェクト内のプライベートイメージをリスト表示するなど、プロジェクト固有の API 操作に役立つ読み取り専用バリアントを形式化します。 オペレーターはこの機会に、glance の新しいイメージポリシーを理解し、デプロイメントでの割り当てを監査し、keystone のデフォルトロール (**admin**、**member**、**reader** など) を使用して権限を更新する必要があります。 関連するオプション: ● [oslo_policy]/enforce_new_defaults **非推奨:** Wallaby **理由:** このオプションは、一般的な RBAC ペルソナに基づいて承認を実施することを選択することをオペレーターに要求するために導入されました。これは Wallaby リリースの時点で実験的です。この動作はデフォルトであり、将来のリリースでは STABLE になり、このオプションを削除できるようになります。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。

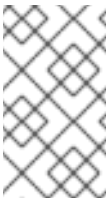
設定オプション = デフォルト値	型	説明
hashing_algorithm = sha512	文字列値	os_hash_value プロパティーの計算に使用される安全なハッシュアルゴリズム。 このオプションは、**os_hash_algo** と **os_hash_value** の 2 つのイメージ属性で設定される Glance の multihash を設定します。**os_hash_algo** はこの設定オプションの値によって入力され、**os_hash_value** は、アルゴリズムがアップロードまたはインポートされたイメージデータに適用されると計算された hexdigest によって入力されます。 この値は、python **hashlib** ライブラリーが認識する有効なセキュアなハッシュアルゴリズム名である必要があります。Glance インストールで使用されているライブラリーのバージョンの **hashlib.algorithms_available** データメンバーを調べることで、これらを確認することができます。ただし、相互運用性のために、**hashlib.algorithms_guaranteed** データメンバーが提供する安全なハッシュ名のセットを使用することが推奨されます。これは、すべてのプラットフォームの **hashlib** ライブラリーでサポートされることが保証されているためです。したがって、**hashlib** を使用するイメージコンシューマーは、イメージの **os_hash_value** を検証できるはずです。 **sha512** のデフォルト値は、高性能なハッシュアルゴリズムです。 このオプションの設定が間違っていると、イメージデータの格納試行は失敗します。このため、デフォルト値を使用することが推奨されます。 値: ● Python **hashlib** ライブラリーが認識するセキュアなハッシュアルゴリズム名 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
image_cache_dir = None	文字列値	イメージキャッシュのベースディレクトリー。 これは、イメージデータがキャッシュされ、提供される場所です。キャッシュされたイメージはすべて、このディレクトリーに直接保存されます。このディレクトリーには、**incomplete**、**invalid**、**queue** という 3 つのサブディレクトリーがあります。 **incomplete** サブディレクトリーは、イメージをダウンロードするためのステージングエリアです。イメージは最初にこのディレクトリーにダウンロードされます。イメージのダウンロードに成功すると、ベースディレクトリーに移動します。ただし、ダウンロードに失敗すると、部分的にダウンロードしたイメージファイルは **invalid** サブディレクトリーに移動します。 **queue`subdirectory は、ダウンロード用のイメージをキューに入れるために使用されます。これは主に cache-prefetcher によって使用されます。これは、使用前にイメージをキャッシュするために cache-pruner や cache-cleaner などの定期的なタスクとしてスケジュールできます。イメージのキャッシュを要求すると、Glance は `queue** ディレクトリー内のファイルに、イメージ ID をファイル名として関連付けます。cache-prefetcher の実行時に、実行時に **queue** ディレクトリーのファイルのポーリングを実行し、ファイルの作成順にダウンロードを開始します。ダウンロードに成功すると、ゼロサイズのファイルが **queue** ディレクトリーから削除されます。ダウンロードに失敗すると、ゼロサイズのファイルが残り、次に cache-prefetcher が実行されるときに再試行されます。 値: ● 有効なパス 関連するオプション: ● **image_cache_sqlite_db**

設定オプション = デフォルト値	型	説明
image_cache_driver = sqlite	文字列値	イメージキャッシュ管理に使用するドライバー。 このオプションを使用すると、利用可能な異なる image-cache ドライバーのいずれかを選択できます。image-cache ドライバーは、キャッシュからのイメージへの書き込み、キャッシュされたイメージの経過時間と使用状況などのイメージキャッシュの基本機能、キャッシュされたイメージのフェッチサイズ、キャッシュのサイズのフェッチ、キャッシュのキャッシュおよびクリーンアップ用のキューイメージの提供を行います。 ドライバーの基本機能は、基本クラス **glance.image_cache.drivers.base.Driver** で定義されています。すべての image-cache ドライバー (既存および見込み) はこのインターフェイスを実装する必要があります。現在利用可能なドライバーは **sqlite** および **xattr** です。これらのドライバーは、主にキャッシュされたイメージに関する情報を保存する方法によって異なります。 ● **sqlite** ドライバーは、キャッシュされたイメージの使用状況を追跡するために、sqlite データベース (すべての gloss ノードにローカルに置かれている) を使用します。 ● **xattr** ドライバーは、ファイルの拡張属性を使用してこの情報を保存します。また、アクセス時にファイルに **atime** を設定するファイルシステムも必要です。 値: ● sqlite ● xattr 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
image_cache_max_size = 10737418240	整数値	キャッシュサイズの上限 (バイト単位)。これを超えると、cache-pruner がイメージのキャッシュをクリーンアップします。 **注記** これは、実行する cache-pruner のしきい値です。これは、イメージキャッシュが拡張されることのないハード制限ではありません。実際、キャッシュプルーナーの実行頻度やキャッシュが満杯になる頻度に応じて、イメージキャッシュはここで非常に簡単に指定されるサイズにさらすことができます。そのため、cache-pruner を適切にスケジュールし、この制限を設定できるように注意して行ってください。 Glance はダウンロード時にイメージをキャッシュします。そのため、ダウンロード数が増えるにつれ、イメージキャッシュのサイズが長くなります。キャッシュサイズが管理できない状態にならないようにするには、cache-pruner を定期的なタスクとして実行することが推奨されます。キャッシュプルーナーが開始すると、現在のイメージキャッシュサイズを比較し、イメージキャッシュがここで指定されるサイズを超えて拡張した場合にクリーンアップがトリガーされます。クリーンアップ後、キャッシュのサイズはここで指定するサイズ以下になります。 値: ● 負の値以外の整数 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
image_cache_sqlite_db = cache.db	文字列値	イメージキャッシュ管理に使用される sqlite ファイルデータベースへの相対パス。 これは、イメージキャッシュの経過時間と使用状況の統計を追跡する sqlite ファイルデータベースの相対パスです。パスは、設定オプション **image_cache_dir** によって指定されるイメージキャッシュベースディレクトリーに相対的です。 これは、テーブルが 1 つしかない軽量データベースです。 値: ● sqlite ファイルデータベースへの有効な相対パス 関連するオプション: ● **image_cache_dir**
image_cache_stall_time = 86400	整数値	不完全なイメージがキャッシュに残っている時間 (秒) です。 不完全なイメージとは、ダウンロードが進行中であるイメージです。詳細は、設定オプション **image_cache_dir** の説明を参照してください。場合によっては、さまざまな理由でダウンロードがハングし、完全にダウンロードされたイメージは **incomplete** なディレクトリーに残ります。この設定オプションでは、不完全なイメージが **incomplete** ディレクトリーに残っている期間を、クリーンアップするまでの時間として設定します。不完全なイメージがここで指定するよりも時間がかかったら、次の実行時に cache-cleaner により削除されます。 不完全なイメージがディスク領域を占有しないように、Glance API ノードで定期的なタスクとして cache-cleaner を実行することが推奨されます。 値: ● 負の値以外の整数 関連するオプション: ● なし

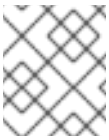
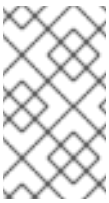
設定オプション = デフォルト値	型	説明
image_location_quota = 10	整数値	イメージで許可される場所の最大数。 負の値は無制限として解釈されます。 関連するオプション: ● なし
image_member_quota = 128	整数値	1イメージあたりのイメージメンバーの最大数。 これにより、イメージが共有できるユーザーの最大数が制限されます。負の値は無制限として解釈されます。 関連するオプション: ● なし
image_property_quota = 128	整数値	イメージで許可されるプロパティーの最大数。 これにより、イメージに割り当てることができる追加のプロパティーの数の上限が適用されます。負の値は無制限として解釈されます。  **注記** 追加のプロパティーが無効の場合、影響はありません。**allow_additional_image_properties** を参照してください。 関連するオプション: ● **allow_additional_image_properties**

設定オプション = デフォルト値	型	説明
image_size_cap = 1099511627776	整数値	ユーザーがアップロードできるイメージの最大サイズ (バイト単位)。 上記のサイズより大きいイメージをアップロードすると、イメージの作成に失敗します。このオプションはデフォルトで 1099511627776 バイト (1 TiB) に設定されます。 注記: ● この値は、慎重に検討した後にのみ増やす必要があり、8 EiB (9223372036854775808) 以下を設定する必要があります。 ● この値は、バックエンドストレージ容量を慎重に検討して設定する必要があります。この値を非常に小さい値に設定すると、イメージが多数失敗する可能性があります。また、この値を非常に大きな値に設定すると、ストレージの使用が速くなる可能性があります。したがって、これは、利用可能なイメージおよび利用可能なストレージ容量の性質に応じて設定される必要があります。 値: ● 9223372036854775808 未満の正数
image_tag_quota = 128	整数値	イメージで許可されるタグの最大数。 負の値は無制限として解釈されます。 関連するオプション: ● なし
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。

設定オプション = デフォルト値	型	説明
limit_param_default = 25	整数値	リクエストに対して返すデフォルトの結果数。 リストイメージなどの特定の API リクエストへの応答により、複数のアイテムが返される場合があります。返される結果の数は、API 要求で **limit** パラメーターを指定して明示的に制御できます。ただし、**limit** パラメーターが指定されていない場合、この設定値は API 要求に対して返される結果のデフォルト数として使用されます。 注記: ● このオプションの値は、**api_limit_max** で指定された値よりも大きくすることはできません。 ● これを非常に大きな値に設定すると、データベースのクエリーが遅くなり、応答時間が長くなる可能性があります。これを非常に低い値に設定すると、ユーザーエクスペリエンスが低下してしまう可能性があります。 値: ● 任意の正の整数 関連するオプション: ● api_limit_max
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
metadata_encryption_key = None	文字列値	ストアの場所のメタデータを暗号化する際の AES 鍵。 Glance ストアメタデータの暗号化に使用する AES 暗号化を表す文字列値を指定します。 **注記** 使用する AES 鍵は、長さ 16、24、または 32 バイトのランダムな文字列に設定する必要があります。 値: ● 有効な AES 鍵を表す文字列値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
node_staging_uri = file:///tmp/staging/	文字列値	URL は一時データが保存される場所を指定します。 このオプションは、Glance の内部使用専用です。Glance は、イメージのインポートプロセス中、ユーザーがアップロードしたイメージデータを **ステージング** エンドポイントに保存します。 このオプションは、**ステージング** API エンドポイントを一切変更しません。  **注記** [task]/work_dir と同じパスを使用することは推奨されません。 **注記** file://<absolute-directory-path> is the only option api_image_import flow will support for now.  **注記** ステージングパスは、すべての Glance API ノードからアクセス可能な共有ファイルシステム上になければなりません。 値: ● **file://** で始まり、ファイルシステムの絶対パスが続く文字列 関連するオプション: ● [task]/work_dir
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。

設定オプション = デフォルト値	型	説明
pydev_worker_debug_host = None	ホストのアドレス値	pydev サーバーのホストアドレス。 デバッグに使用する pydev サーバーのホスト名または IP を表す文字列値を指定します。pydev サーバーはこのアドレスでデバッグ接続をリッスンし、Glance でのリモートデバッグを容易にします。 値: ● 有効なホスト名 ● 有効な IP アドレス 関連するオプション: ● なし
pydev_worker_debug_port = 5678	ポート値	pydev サーバーがリッスンするポート番号。 pydev サーバーをバインドするポート番号を指定します。pydev プロセスはこのポートでデバッグ接続を受け入れ、Glance でのリモートデバッグを容易にします。 値: ● 有効なポート番号 関連するオプション: ● なし
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。

設定オプション = デフォルト値	型	説明
show_image_direct_url = False	ブール値	イメージを返す際に直接イメージの場所を示します。 この設定オプションは、イメージの詳細をユーザーに返すときに直接イメージの場所を表示するかどうかを指定します。直接イメージの場所は、バックエンドストレージ内でイメージデータが保存される場所です。このイメージの場所は、イメージ属性 **direct_url** セクションに表示されます。 イメージに対して複数のイメージの場所が存在する場合には、設定オプション **location_strategy** で示される場所ストラテジーに基づいて、最適な場所が表示されます。 注記: ● イメージの場所を表示すると、イメージの場所に認証情報が含まれることがあるため、重大なセキュリティーリスクが発生する可能性があります。したがって、これはデフォルトで **False** に設定されています。設定の意味するところを理解している場合に限り、細心の注意を払ってこのオプションを **True** に設定してください。 ● 運用者がユーザーにイメージの場所を表示しないことを希望する場合は、このオプションと **show_multiple_locations** の両方を **False** に設定する必要があります。 値: ● True ● False 関連するオプション: ● show_multiple_locations ● location_strategy

設定オプション = デフォルト値	型	説明
show_multiple_locations = False	ブール値	イメージを返す際にすべてのイメージの場所を示します。 この設定オプションは、イメージの詳細をユーザーに返すときにすべてのイメージの場所を表示するかどうかを指定します。イメージに対して複数のイメージの場所が存在する場合には、設定オプション **location_strategy** で示される場所ストラテジーに基づいて、場所が順番に表示されます。イメージの場所は、イメージ属性 **locations** セクションに表示されます。 注記: ● イメージの場所を表示すると、イメージの場所に認証情報が含まれることがあるため、重大なセキュリティーリスクが発生する可能性があります。したがって、これはデフォルトで **False** に設定されています。設定の意味するところを理解している場合に限り、細心の注意を払ってこのオプションを **True** に設定してください。 ● 詳細は、https://wiki.openstack.org/wiki/OSSN/OSSN-0065 を参照してください。 ● 運用者がユーザーにイメージの場所を表示しないことを希望する場合は、このオプションと **show_image_direct_url** の両方を **False** に設定する必要があります。 値: ● True ● False 関連するオプション: ● show_image_direct_url ● location_strategy **非推奨となったバージョン：** Newton *理由:* Newton 以降非推奨になったこのオプションの使用はセキュリティーリスクであり、現在このオプションを必要とするそれらのユースケースを満たす方法がわかれば削除されます。ポリシーを使用することで同じ機能をより細かい粒度で実現できるという以前の通知は、正しくありません。ポリシー設定でこのオプションを回避することが取るべき修正の方向と考えていますが、現在、この方法で回避することはできません。この問題への対処に関する最新の進捗状況を把握するために、常に Glance のリリースノートに注意を払ってください。

設定オプション = デフォルト値	型	説明
OG_USER	字列値	行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
user_storage_quota = 0	文字列値	テナントごとのイメージストレージの最大容量。 これにより、全ストアにわたってテナントのすべてのイメージによって消費される累積ストレージに上限が適用されます。これはテナントごとの制限です。 この設定オプションのデフォルトの単位はバイトです。ただし、ストレージの単位は、大文字/小文字を区別する文字**B**、**KB**、**MB**、**GB**、および **TB**を使用して指定することができ、それぞれバイト、キロバイト、メガバイト、ギガバイト、およびテラバイトを表します。値と単位の間にはスペースを入れないでください。値 **0** は、クォータが適用されないことを意味します。負の値は無効で、エラーが発生します。 値: ● 上記のように、ストレージの値を表す負ではない整数とストレージの単位を表すオプションの文字列の有効な組み合わせである文字列。 関連するオプション: ● なし
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
worker_self_reference_url = None	文字列値	このワーカーへの URL。 これが設定されている場合、他のグランスワーカーは、必要に応じてこのワーカーに直接連絡する方法を知っています。イメージのインポートの場合、1 人のワーカーがイメージをステージングし、他のワーカーがインポートリクエストを適切なワーカーにプロキシーできる必要があります。 設定されていない場合、これは **public_endpoint** と見なされ、通常はすべてのワーカーで同じ値に設定され、プロキシー動作を効果的に無効にします。 値: ● このワーカーが他のワーカーから到達可能な URL 関連するオプション: ● public_endpoint

3.3.2. glance_store

以下の表は、**/etc/glance/glance-cache.conf** ファイルの **[glance_store]** グループで利用可能なオプションの概要をまとめたものです。

表3.30 glance_store

設定オプション = デフォルト値	型	説明
cinder_api_insecure = False	ブール値	cinder に対するセキュアではない SSL リクエストの実行を許可します。 このオプションが True に設定されている場合、HTTPS エンドポイント接続は **cinder_ca_certificates_file** オプションで指定された CA 証明書ファイルを使用して検証されます。 値: ● True ● False 関連するオプション: ● cinder_ca_certificates_file

設定オプション = デフォルト値	型	説明
cinder_ca_certificates_file = None	文字列値	cinder クライアントリクエストに使用する CA 証明書ファイルの場所。 CA 証明書ファイルが設定されている場合、指定されたファイルが、HTTPS エンドポイントを使用した cinder 接続の検証に使用されます。エンドポイントが HTTP の場合、この値は無視されます。検証を有効にするには、**cinder_api_insecure** を **True** に設定する必要があります。 値: ● CA 証明書ファイルへのパス 関連するオプション: ● cinder_api_insecure

設定オプション = デフォルト値	型	説明
cinder_catalog_info = volumev3::publicURL	文字列値	サービスカタログで cinder を検索する際に照合する情報。 **cinder_endpoint_template** が設定されておらず、**cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、**cinder_store_password** のいずれも設定されていない場合、cinder store はこの情報を使用して、現在のコンテキストでサービスカタログから cinder エンドポイントを検索します。**cinder_os_region_name** が設定されている場合は、適切なエンドポイントを取得するために考慮されます。 サービスカタログは、**openstack catalog list** コマンドでリスト表示できます。 値: ● 次の形式の文字列: **<service_type>:<service_name>:<interface>** 少なくとも **service_type** および **interface** を指定する必要があります。**service_name** は省略できます。 関連するオプション: ● cinder_os_region_name ● cinder_endpoint_template ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name ● cinder_store_password

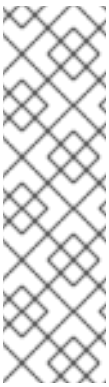
設定オプション = デフォルト値	型	説明
cinder_endpoint_template = None	文字列値	cinder エンドポイントのテンプレートでサービスカタログの検索を上書きします。 このオプションを設定すると、サービスカタログから検索するのではなく、この値を使用して cinder エンドポイントが生成されます。**cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、および **cinder_store_password** が指定されている場合には、この値は無視されます。 この設定オプションが設定されている場合、**cinder_catalog_info** は無視されます。 値: ● cinder エンドポイントの URL テンプレート文字列。ここで、**%%(tenant)s** は現在のテナント (プロジェクト) 名に置き換えます。例:**http://cinder.openstack.example.org/v2/%%(tenant)s** 関連するオプション: ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name ● cinder_store_password ● cinder_catalog_info
cinder_enforce_multipath = False	ブール値	これが True に設定されている場合、multipathd が実行されていない場合、イメージ転送用のボリュームのアタッチが中断されます。それ以外の場合は、単一のパスにフォールバックします。 値: ● True か False 関連するオプション: ● cinder_use_multipath

設定オプション = デフォルト値	型	説明
cinder_http_retries = 3	整数値	失敗した http 呼び出しでの cinderclient のリトライ回数。 何らかのエラーによって呼び出しに失敗すると、cinderclient は数秒スリープした後に呼び出しを指定の回数までリトライします。 値: ● 正の整数 関連するオプション: ● なし
cinder_mount_point_base = /var/lib/glance/mnt	文字列値	glance ノードで NFS ボリュームがマウントされるディレクトリー。 値: ● マウントポイントの絶対パスを表す文字列。
cinder_os_region_name = None	文字列値	サービスカタログから cinder サービスを検索する際のリージョン名。 これは、**cinder_catalog_info** を使用してエンドポイントを決定する場合にのみ使用されます。これが設定されている場合、このノードによる cinder エンドポイントの検索は、指定のリージョンにフィルターされます。これは、カタログに複数のリージョンがリスト表示される場合に役立ちます。これが設定されていない場合、エンドポイントはすべてのリージョンから検索されます。 値: ● 有効なリージョン名である文字列。 関連するオプション: ● cinder_catalog_info

設定オプション = デフォルト値	型	説明
cinder_state_transition_timeout = 300	整数値	cinder ボリュームの遷移が完了するまで待機する時間 (秒単位)。 ボリュームデータの読み取り/書き込みを行うために、cinder ボリュームを作成、削除、または glance ノードにアタッチすると、ボリュームの状態が変更されます。たとえば、新規作成されたボリュームのステータスは、作成プロセスの完了後に **creating** から **available** に変わります。このオプションは、ステータス変更を待機する最大時間を指定します。待機時間がタイムアウトするか、ステータスが予期しない値に変更される場合 (**error**など)、イメージの作成は失敗します。 値: ● 正の整数 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
cinder_store_auth_address = None	文字列値	cinder 認証サービスがリッスンしているアドレス。 **cinder_store_auth_address**、**cinder_store_user_name**、**cinder_store_project_name**、および **cinder_store_password**オプションのすべてが指定された場合、指定した値は常に認証に使用されます。これは、イメージサービスに固有のプロジェクト/テナントにイメージボリュームを保存することにより、ユーザーからイメージボリュームを隠すのに役立ちます。また、ユーザーは glance の ACL の制御下で、他のプロジェクト間でイメージボリュームを共有できます。 これらのオプションのいずれかが設定されていない場合、cinder エンドポイントはサービスカタログから検索され、現在のコンテキストのユーザーとプロジェクトが使用されます。 値: ● 有効な認証サービスアドレス (例:**http://openstack.example.org/identity/v2.0**)。 関連するオプション: ● cinder_store_user_name ● cinder_store_password ● cinder_store_project_name
cinder_store_password = None	文字列値	cinder に対して認証を行うユーザーのパスワード。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのユーザーが使用されます。 値: ● **cinder_store_user_name**で指定したユーザーの有効なパスワード 関連するオプション: ● cinder_store_auth_address ● cinder_store_user_name ● cinder_store_project_name

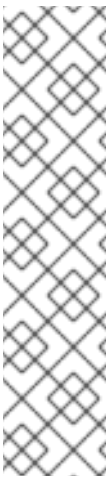
設定オプション = デフォルト値	型	説明
cinder_store_project_name = None	文字列値	cinder にイメージボリュームが保存されるプロジェクト名。 この設定オプションが設定されていない場合は、現在のコンテキストのプロジェクトが使用されます。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのプロジェクトが使用されます。 値: ● 有効なプロジェクト名 関連するオプション: ● **cinder_store_auth_address** ● **cinder_store_user_name** ● **cinder_store_password**
cinder_store_user_name = None	文字列値	cinder に対して認証を行うためのユーザー名。 このオプションは、以下の関連オプションすべてと共に使用する必要があります。これらのいずれかが指定されていない場合は、現在のコンテキストのユーザーが使用されます。 値: ● 有効なユーザー名 関連するオプション: ● cinder_store_auth_address ● cinder_store_password ● cinder_store_project_name

設定オプション = デフォルト値	型	説明
cinder_use_multipath = False	ブール値	デプロイメントでマルチパスがサポートされるかどうかを識別するためのフラグ。 マルチパスがサポートされない場合は、False に設定します。 値: ● True か False 関連するオプション: ● cinder_enforce_multipath
cinder_volume_type = None	文字列値	cinder でのボリュームの作成に使用されるボリューム種別。 一部の cinder バックエンドには、ストレージの使用を最適化するための複数のボリューム種別が含まれる場合があります。このオプションを追加すると、運用者はイメージに対して最適化できる特定のボリューム種別を cinder で選択することができます。 これが設定されていない場合には、cinder 設定で指定したデフォルトのボリューム種別がボリュームの作成に使用されます。 値: ● cinder からの有効なボリューム種別 関連するオプション: ● なし  **注記** NFS バックエンドに関連付けられた暗号化された volume_type は使用できません。NFS バックエンドに保存されている暗号化されたボリュームは、glance_store がそのボリュームに保存されているイメージデータの書き込みまたはアクセスを試みるたびに、例外を発生させます。Cinder 管理者に相談して、適切な volume_type を決定してください。

設定オプション = デフォルト値	型	説明
default_store = file	文字列値	イメージの保存に使用するデフォルトのスキーム。 イメージの保存に使用するデフォルトのスキームを表す文字列値を指定します。設定されていない場合、Glance は **file** をデフォルトのスキームとして使用して **file** ストアでイメージを保存します。 **注記** この設定オプションに指定する値は、**stores** 設定オプションで登録されたストアの有効なスキームである必要があります。 値: - file - Filesystem - http - https - swift - swift+http - swift+https - swift+config - rbd - cinder - vsphere - s3 関連するオプション: - stores **非推奨となったバージョン：** Rocky **理由:** このオプションは、''default_store'' 設定オプションと同様に機能する新しい設定オプション ''default_backend'' によって非推奨となりました。 このオプションは、U 開発サイクルで削除される予定です。

設定オプション = デフォルト値	型	説明
default_swift_reference = ref1	文字列値	デフォルトの Swift アカウント/バッキングストアパラメーターへの参照。 イメージストレージに swift アカウント/バッキングストアを使用するのに必要なデフォルトのパラメーターセットへの参照を表す文字列値を指定します。この設定オプションのデフォルト参照値は **ref1** です。この設定オプションはパラメーターを逆参照し、新しいイメージが追加されるたびに Swift ストレージバックエンドでのイメージ保管を容易にします。 値: ● 有効な文字列値 関連するオプション: ● なし
filesystem_store_chunk_size = 65536	整数値	チャンクサイズ (バイト単位)。 イメージファイルの読み取り/書き込み時に使用されるチャンクサイズ。この値を大きくするとスループットが向上しますが、大量のリクエストを処理する際にメモリー使用量が若干増加する場合もあります。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
filesystem_store_datadir = /var/lib/glance/images	文字列値	ファイルシステムのバックエンドストアがイメージを書き込むディレクトリー。 起動時に、Glance はディレクトリーが存在しない場合は作成し、**glance-api** が実行されるユーザーへの書き込みアクセスを検証します。書き込みアクセスが使用できない場合は、**BadStoreConfiguration** 例外が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 **注記** このディレクトリーは、ファイルシステムストアがストレージバックエンドとして使用される場合にのみ使用されます。**filesystem_store_datadir** または **filesystem_store_datadirs** オプションのいずれかを **glance-api.conf** で指定する必要があります。両方のオプションを指定すると、**BadStoreConfiguration** が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 値: ● ディレクトリーへの有効なパス 関連するオプション: ● **filesystem_store_datadirs** ● **filesystem_store_file_perm**

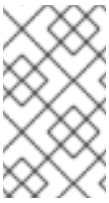
設定オプション = デフォルト値	型	説明
filesystem_store_datadirs = None	多値	ファイルシステムのバックエンドストアがイメージを書き込むディレクトリーとその優先順位のリスト。 **filesystem_store_datadir** 設定オプションで指定した単一のディレクトリーを使用する場合とは対照的に、ファイルシステムストアは複数のディレクトリーにイメージを格納するように設定できます。複数のディレクトリーを使用する場合、各ディレクトリーには、使用される優先順位を指定するオプションの優先度を指定できます。優先度は、ディレクトリーパスとコロンで組み合わされる整数で、値が大きいほど優先度が高くなります。2 つのディレクトリーの優先度が同じ場合は、空き容量が最も多いディレクトリーが使用されます。優先度を指定しないと、デフォルトでゼロに設定されます。 複数のストアディレクトリーを持つファイルシステムストアの設定に関する詳細は、https://docs.openstack.org/glance/latest/configuration/configuring.html を参照してください。  **注記** このディレクトリーは、ファイルシステムストアがストレージバックエンドとして使用される場合にのみ使用されます。**filesystem_store_datadir** または **filesystem_store_datadirs** オプションのいずれかを **glance-api.conf** で指定する必要があります。両方のオプションを指定すると、**BadStoreConfiguration** が発生し、ファイルシステムストアが新しいイメージの追加に利用できないことがあります。 値: - 以下の形式の文字列のリスト: - **<a valid directory path>:<optional integer priority>** 関連するオプション: - **filesystem_store_datadir** - **filesystem_store_file_perm**

設定オプション = デフォルト値	型	説明
filesystem_store_file_perm = 0	整数値	イメージファイルへのファイルアクセスパーミッション。 イメージデータへの目的のファイルアクセスパーミッションを設定します。これにより、Nova などの他のサービスがファイルシステムストアから直接イメージを使用できるようにすることができます。アクセスが付与されるサービスを実行中のユーザーは、作成されるファイルを所有しているグループのメンバーにすることができます。この設定オプションにゼロ以下の値を割り当てることは、デフォルトのパーミッションに変更が加えられないことを意味します。この値は、8 進数の数字としてデコードされます。 詳細は、https://docs.openstack.org/glance/latest/configuration/configuring.htmlのドキュメントを参照してください。 値: ● 有効なファイルアクセスパーミッション ● ゼロ ● 任意の負の整数 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
filesystem_store_metadata_file = None	文字列値	ファイルシステムストアのメタデータファイル。 ファイルシステムストアに関連付けられた場所と共に返されるメタデータが含まれるファイルへのパス。このオプションが設定されると、後で作成される新しいイメージにのみ使用されます。以前の既存のイメージには影響しません。 このファイルには、有効な JSON オブジェクトが含まれている必要があります。オブジェクトには、キー **ID** と **マウントポイント** が含まれている必要があります。両方のキーの値は文字列でなければなりません。 値: ● ストアメタデータファイルへの有効なパス 関連するオプション: ● なし
filesystem_thin_provisioning = False	ブール値	このバックエンドでシンプロビジョニングを有効にするかどうか。 この設定オプションを使用すると、ファイルシステムに null バイトシーケンスを実際に書き込まないという機能が有効になります。表示されるホールは、ファイルシステムによって自動的に null バイトとして解釈され、ストレージを実際に消費しません。この機能を有効にすると、ネットワーク経由で null バイトシーケンスが送信されないため、バックエンドのスペースを節約できるだけでなく、イメージのアップロードが高速化され、ネットワークトラフィックが節約されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
http_proxy_information = {}	dict 値	リモートサーバーへの接続に使用する http/https プロキシー情報。 この設定オプションは、リモートサーバーへの接続に使用する必要のある http/https プロキシー情報を指定します。プロキシー情報は、スキームとプロキシーのキー/値のペアにする必要があります (例:http:10.0.0.1:3128)。また、キーと値のペアをコンマで区切ることで、複数のスキームのプロキシーを指定することもできます (例: http:10.0.0.1:3128, https:10.0.0.1:1080)。 値: ● 上記のように スキーム: プロキシーペアのコンマ区切りリスト 関連するオプション: ● なし
https_ca_certificates_file = None	文字列値	CA バンドルファイルへのパス。 この設定オプションにより、運用者はカスタム認証局ファイルを使用してリモートサーバー証明書を検証できます。このオプションを設定すると、**https_insecure** オプションは無視され、指定された CA ファイルがサーバー証明書の認証に使用され、サーバーへのセキュアな接続が確立されます。 値: ● CA ファイルへの有効なパス 関連するオプション: ● https_insecure

設定オプション = デフォルト値	型	説明
https_insecure = True	ブール値	リモートサーバー証明書の検証を設定します。 この設定オプションはブール値を取り、リモートサーバーの証明書を検証するかどうかを決定します。True に設定すると、リモートサーバー証明書は検証されません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 このオプションは、**https_ca_certificates_file** が設定されている場合は無視されます。リモートサーバー証明書は、**https_ca_certificates_file** オプションを使用して指定されたファイルを使用して検証されます。 値: ● True ● False 関連するオプション: ● https_ca_certificates_file
rados_connect_timeout = 0	整数値	Ceph クラスターに接続する際のタイムアウト値。 この設定オプションは、Ceph クラスターに接続する際に使用されるタイムアウト値 (秒単位) を取ります。つまり、接続を閉じるまで glance-api が待機する時間を設定します。これにより、RBD への接続中に glance-api がハングアップするのを防ぐことができます。このオプションの値を 0 以下に設定すると、タイムアウトは設定されず、デフォルトの librados 値が使用されます。 以下の値を使用できます。 ● 任意の整数値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
`rbd_store_ceph_conf = `	文字列値	Ceph 設定ファイルへのパス。 この設定オプションは、使用する Ceph 設定ファイルへのパスを指定します。このオプションの値がユーザーにより設定されていない場合や、空の文字列に設定されている場合、librados は、デフォルトの Ceph 設定ファイルの場所を順に検索して標準の ceph.conf ファイルを読み取ります。詳細は、Ceph のドキュメントを参照してください。  **注記** Cephx 認証を使用する場合、このファイルの client.<USER>セクションに適切なキーリングへの参照を含める必要があります。 注 2: このオプションを空白のままにすると (デフォルト)、使用される実際の Ceph 設定ファイルは、使用されている librados のバージョンによって変わる可能性があります。有効になっている設定ファイルを正確に把握しておくことが重要な場合は、このオプションを使用してここにファイルを指定できます。 以下の値を使用できます。 ● 設定ファイルへの有効なパス 関連するオプション: ● rbd_store_user

設定オプション = デフォルト値	型	説明
rbd_store_chunk_size = 8	整数値	RADOS イメージをチャンクに分割する際のサイズ (メガバイト単位)。 Glance イメージをチャンクに分割する際のサイズ (メガバイト単位) を表す整数値を指定します。デフォルトのチャンクサイズは 8 メガバイトです。最適なパフォーマンスを得るには、値は 2 の累乗でなければなりません。 Ceph の RBD オブジェクトストレージシステムを Glance イメージを保存するためのストレージバックエンドとして使用する場合、イメージはこのオプションを使用して設定されるサイズのオブジェクトに分割されます。これらの分割されたオブジェクトは、分散ブロックデータストア全体に保存され、Glance に使用されます。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし
rbd_store_pool = images	文字列値	イメージが保存される RADOS プール。 RBD を Glance イメージを保存するためのストレージバックエンドとして使用する場合、イメージは **プール** へのオブジェクト (イメージのチャンク) の論理的なグループ化により保存されます。各プールは、含めることのできる配置グループの数で定義されます。使用されるデフォルトのプールは **イメージ** です。 RBD ストレージバックエンドの詳細は、http://ceph.com/planet/how-data-is-stored-in-ceph-cluster/を参照してください。 以下の値を使用できます。 ● 有効なプール名 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
rbd_store_user = None	文字列値	認証する RADOS ユーザー。 この設定オプションは、認証する RADOS ユーザーを設定します。このオプションは RADOS 認証が有効になっている場合にのみ必要で、ユーザーが Cephx 認証を使用している場合にのみ適用されます。このオプションの値がユーザーによって設定されていない場合や、None に設定されている場合には、デフォルト値が選択されます。これは、rbd_store_ceph_conf の client. セクションに基づきます。 以下の値を使用できます。 ● 有効な RADOS ユーザー 関連するオプション: ● rbd_store_ceph_conf
rbd_thin_provisioning = False	ブール値	このバックエンドでシンプロビジョニングを有効にするかどうか。 この設定オプションを使用すると、RBD バックエンドに null バイトシーケンスを実際に書き込まないという機能が有効になります。表示されるホールは、Ceph によって自動的に null バイトとして解釈され、ストレージを実際に消費しません。この機能を有効にすると、ネットワーク経由で null バイトシーケンスが送信されないため、バックエンドのスペースを節約できるだけでなく、イメージのアップロードが高速化され、ネットワークトラフィックが節約されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
rootwrap_config = /etc/glance/rootwrap.conf	文字列値	root としてコマンドを実行するのに使用する rootwrap 設定ファイルへのパス。 cinder ストアでは、イメージボリュームを操作するのに root 権限が必要です (iSCSI/FC ボリュームへの接続、ボリュームデータの読み取り/書き込みのためなど)。設定ファイルでは、cinder ストアおよび os-brick ライブラリーによる必要なコマンドを許可する必要があります。 値: ● rootwrap 設定ファイルへのパス 関連するオプション: ● なし
s3_store_access_key = None	文字列値	S3 クエリートークンアクセスキー。 この設定オプションは、Amazon S3 または S3 互換ストレージサーバーで認証するためのアクセスキーを取得します。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。 値: ● 適切な権限を持つユーザーのアクセスキーである任意の文字列値 関連するオプション: ● s3_store_host ● s3_store_secret_key

設定オプション = デフォルト値	型	説明
s3_store_bucket = None	文字列値	Glance データの保存に使用される S3 バケット。 この設定オプションは、Glance イメージが S3 に保存される場所を指定します。**s3_store_create_bucket_on_put** を true に設定すると、バケットが存在しなくても自動的に作成されます。 値: ● 任意の文字列値 関連するオプション: ● s3_store_create_bucket_on_put ● s3_store_bucket_url_format
s3_store_bucket_url_format = auto	文字列値	オブジェクトを決定するために使用される S3 呼び出し形式。 この設定オプションは、S3 バケット内のオブジェクトのアドレスを指定するために使用されるアクセスモデルを取ります。 注記: **path** スタイルでは、オブジェクトのエンドポイントは https://s3.amazonaws.com/bucket/example.img のようになります。**virtual** スタイルでは、オブジェクトのエンドポイントは https://bucket.s3.amazonaws.com/example.img のようになります。バケット名で DNS 命名規則に従わない場合、パススタイルでオブジェクトを取得できますが、仮想スタイルでは取得できません。 値: ● **auto**、**virtual**、または **path** の任意の文字列値 関連するオプション: ● s3_store_bucket

設定オプション = デフォルト値	型	説明
s3_store_create_bucket_on_put = False	ブール値	S3 が新しいバケットを作成する必要があるかどうかを決定します。 この設定オプションはブール値を取り、Glance が S3 に新しいバケットを作成する必要があるかどうかを示します (存在しない場合)。 値: ● 任意のブール値 関連するオプション: ● なし
s3_store_host = None	文字列値	S3 サーバーがリッスンしているホスト。 この設定オプションは、S3 または S3 互換ストレージサーバーのホストを設定します。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。ホストには、DNS 名 (s3.amazonaws.com、my-object-storage.com など) または IP アドレス (127.0.0.1) を含めることができます。 値: ● 有効な DNS 名 ● 有効な IPv4 アドレス 関連するオプション: ● s3_store_access_key ● s3_store_secret_key

設定オプション = デフォルト値	型	説明
s3_store_large_object_chunk_size = 10	整数値	パーツのアップロード時に S3 が使用するマルチパートアップロードパーツのサイズ (MB 単位)。 この設定オプションは、マルチパートアップロードのイメージ分割サイズを MB 単位で取得します。 注記: 分割できるイメージは 10,000 個までです。 値: ● 任意の正の整数値 (5M 以上である必要があります) 関連するオプション: ● s3_store_large_object_size ● s3_store_thread_pools
s3_store_large_object_size = 100	整数値	S3 がイメージファイルのチャンク化を開始し、S3 でマルチパートアップロードを実行する必要があるサイズ (MB 単位)。 この設定オプションは、イメージをそのまま S3 にアップロードするか、分割するか (マルチパートアップロード) を決定するために MB 単位でしきい値を取得します。 注記: 分割できるイメージは 10,000 個までです。 値: ● 任意の正の整数値 関連するオプション: ● s3_store_large_object_chunk_size ● s3_store_thread_pools

設定オプション = デフォルト値	型	説明
s3_store_secret_key = None	文字列値	S3 クエリートークンの秘密鍵。 この設定オプションは、Amazon S3 または S3 互換ストレージサーバーで認証するための秘密鍵を受け取ります。このオプションは、S3 ストレージバックエンドを使用する場合に必要です。 値: ● **s3_store_host** オプションを使用して指定されたアクセスキーに対応する秘密鍵である任意の文字列値 関連するオプション: ● s3_store_host ● s3_store_access_key
s3_store_thread_pools = 10	整数値	S3 でマルチパートアップロードを実行するためのスレッドプールの数。 この設定オプションは、マルチパートアップロードの実行時にスレッドプールの数を取得します。 値: ● 任意の正の整数値 関連するオプション: ● s3_store_large_object_size ● s3_store_large_object_chunk_size

設定オプション = デフォルト値	型	説明
stores = ['file', 'http']	リスト値	有効な Glance ストアのリスト。 ディスクイメージを格納するために使用するストレージバックエンドを、コンマ区切りリストとして登録します。Glance でのディスクイメージの保存用に有効化されたデフォルトのストアは、**file** と **http** です。 値: ● 以下が含まれるコンマ区切りリスト: ○ file ○ http ○ swift ○ rbd ○ cinder ○ vmware ○ s3 関連するオプション: ● default_store **非推奨となったバージョン：** Rocky **理由:** このオプションは、異なるスキームの複数のバックエンドストアを設定するのに役立つ新たな設定オプション ''enabled_backends'' によって非推奨となりました。 このオプションは、U 開発サイクルで削除される予定です。

設定オプション = デフォルト値	型	説明
swift_buffer_on_upload = False	ブール値	Swift にアップロードする前にイメージのセグメントをバッファー処理します。 swift へのアップロード中に Glance がイメージデータをディスクにバッファーするかどうかを指定するブール値を指定します。これにより、Glance はエラー時にアップロードを再開できます。 注意: このオプションを有効にすると、API ノードのディスク使用量が増加するので、細心の注意を払う必要があります。ファイルシステムの設定状況によっては、バッファー処理に使用されるディスク領域により、glance イメージキャッシュに利用可能な実際のディスク容量が減る可能性があることに注意してください。ディスク使用率は、式**swift_store_large_object_chunk_size** * **workers** * 1000 に従って上限が設定されます。 値: ● True ● False 関連するオプション: ● swift_upload_buffer_dir
swift_store_admin_tenants = []	リスト値	管理者アクセスが付与されるテナントのリスト。 これは、マルチテナントモードの Glance によって作成されたすべての Swift コンテナーでの読み取り/書き込みアクセスが付与されるテナントのリストです。デフォルト値は空のリストです。 値: ● Keystone プロジェクト/テナントの UUID を表す文字列のコンマ区切りリスト 関連するオプション: ● なし
swift_store_auth_address = None	文字列値	Swift 認証サービスがリッスンしているアドレス。

設定オプション = デフォルト値	型	説明
swift_store_auth_insecure = False	ブール値	サーバー証明書の検証を設定します。 このブール値は、サーバー証明書を検証するかどうかを決定します。このオプションが True に設定されている場合、swiftclient は認証時に有効な SSL 証明書かどうかをチェックしません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 値: ● True ● False 関連するオプション: ● swift_store_cacert
swift_store_auth_version = 2	文字列値	使用する認証サービスのバージョン。有効なバージョンは、keystone の場合は 2 および 3 で、swauth および rackspace の場合は 1(非推奨) です。
swift_store_cacert = None	文字列値	CA バンドルファイルへのパス。 この設定オプションにより、運用者は Swift に接続する際の SSL 検証用のカスタム認証局ファイルへのパスを指定できます。 値: ● CA ファイルへの有効なパス 関連するオプション: ● swift_store_auth_insecure

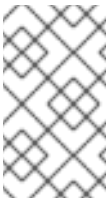
設定オプション = デフォルト値	型	説明
swift_store_config_file = None	文字列値	swift アカウント設定が含まれるファイルへの絶対パス。 設定済みの各 Swift アカウント/バッキングストアへの参照を持つ設定ファイルへのパスを表す文字列値を含めます。デフォルトでは、ファイルパスが指定されておらず、カスタマイズされた Swift 参照は無効になっています。データベースへの認証情報の保存を回避するため、このオプションを設定することは、イメージストレージに Swift ストレージバックエンドを使用する際に強く推奨されます。 **注記** **swift_store_multi_tenant** を **True** に設定した場合は、このオプションを設定しないでください。 値: ● glance-api ノードの絶対パスを表す文字列値 関連するオプション: ● swift_store_multi_tenant

設定オプション = デフォルト値	型	説明
swift_store_container = glance	文字列値	イメージを保存するための単一のコンテナーの名前/複数のコンテナーの名前の接頭辞 1 つのコンテナーを使用してイメージを保存する場合、この設定オプションは、すべてのイメージの保存に使用される Glance アカウント内のコンテナーを示します。複数のコンテナーがイメージの保存に使用される場合、これはすべてのコンテナーの名前の接頭辞になります。単一/複数コンテナーの使用は、設定オプション **swift_store_multiple_containers_seed** を使用して制御できます。 複数のコンテナーを使用する場合、コンテナーの名前は、この設定オプションに設定された値をベースに、イメージ UUID の最初の N 文字をアンダースコアで区切られた接尾辞として指定されます (N は **swift_store_multiple_containers_seed**によって指定されます)。 例: シードが 3 で swift_store_container = **glance** と設定されている場合には、UUID が**fdae39a1-bac5-4238-aba4-69bcc726e848** のイメージは **glance_fda** のコンテナーに配置されます。コンテナー名の作成時に、UUID のすべてのダッシュは含まれますが、文字の制限にはカウントされないため、N=10 の場合はコンテナー名は **glance_fdae39a1-ba** になります。 値: ● 単一のコンテナーを使用する場合、この設定オプションは Glance の Swift アカウントの有効な任意の swift コンテナーの名前の文字列にすることができます。 ● 複数のコンテナーを使用する場合、この設定オプションは、 Swift により適用されるコンテナーの命名ルールを満たす限り任意の文字列になります。**swift_store_multiple_containers_seed** の値も考慮する必要があります。 関連するオプション: ● **swift_store_multiple_containers_seed** ● **swift_store_multi_tenant** ● **swift_store_create_container_on_put**

設定オプション = デフォルト値	型	説明
swift_store_create_container_on_put = False	ブール値	イメージのアップロード時に、コンテナーを作成します (まだ存在しない場合)。 イメージのアップロード時に、対応するコンテナーが存在しない場合は、この設定オプションが True に設定されている場合に作成されます。デフォルトでは、コンテナーは作成されません。この動作は、単一および複数コンテナーモードの両方に適用されます。 値: ● True ● False 関連するオプション: ● なし
swift_store_endpoint = None	文字列値	Swift バックエンドストレージに使用する URL エンドポイント。 Glance イメージを Swift ストアに格納するために使用する URL エンドポイントを表す文字列値を指定します。デフォルトでは、エンドポイントは設定されず、**auth** によって返されるストレージ URL が 使用されます。**swift_store_endpoint** を使用してエンドポイントを設定すると、ストレージ URL を上書きされ、Glance イメージストレージに使用されます。 **注記** URL には、コンテナーまでのパスを含める必要がありますが、コンテナーそのものは除外します。オブジェクトの場所は、コンテナーおよびオブジェクトを設定した URL に追加すると得られます。 値: ● Swift コンテナーまでの有効な URL パスを表す文字列値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_endpoint_type = publicURL	文字列値	Swift サービスのエンドポイント種別。 この文字列値は、Swift エンドポイントの取得に使用するエンドポイント種別を示します。エンドポイント種別により、ユーザーが実行できるアクション (たとえば、ストアの読み取りおよび書き込み) が決定されます。この設定は、swift_store_auth_version が 1 よりも大きい場合にのみ使用されます。 値: ● publicURL ● adminURL ● internalURL 関連するオプション: ● swift_store_endpoint
swift_store_expire_soon_interval = 60	整数値	現在のトークンの有効期限が過ぎる前に新しいトークンが要求される可能性があるウィンドウのサイズを定義する時間 (秒単位)。 通常、Swift ストレージドライバーは、現在のトークンの有効期限が過ぎた時に新しいトークンを取得して、Swift へのアクセスを継続させます。ただし、処理中にトークンが期限切れになると、一部の Swift トランザクション (イメージのセグメントのアップロードなど) が適切に復元されないことがあります。 したがって、現在のトークンの有効期限前に新しいトークンを取得することにより、トランザクションを試行する前にトークンが失効したり期限切れに近づいたりしないようにします。デフォルトでは、Swift ストレージドライバーは、現在のトークンの有効期限の 60 秒前から新しいトークンを要求します。 値: ● ゼロ ● 正の整数値 関連するオプション: ● なし
swift_store_key = None	文字列値	Swift 認証サービスに対して認証するユーザーの認証鍵。

設定オプション = デフォルト値	型	説明
swift_store_large_object_chunk_size = 200	整数値	イメージデータがセグメント化される際のセグメントの最大サイズ (MB 単位)。 Swift クラスターによって適用される上限を上回るイメージをアップロードするためにイメージデータがセグメント化される場合、イメージデータはこの設定オプションで指定されるサイズ以下のセグメントに分割されます。詳細は、**swift_store_large_object_size** を参照してください。 たとえば、**swift_store_large_object_size** が 5 GB で、**swift_store_large_object_chunk_size** が 1 GB の場合、サイズが 6.2 GB のイメージは 7 セグメントに分割されます。この場合、最初の 6 つのセグメントのサイズは 1 GB で、7 番目のセグメントは 0.2 GB になります。 値: ● 該当する Swift クラスターによって適用される大型オブジェクト制限以下の正の整数。 関連するオプション: ● **swift_store_large_object_size**

設定オプション = デフォルト値	型	説明
swift_store_large_object_size = 5120	整数値	サイズしきい値 (MB 単位)。これを超えると、Glance はイメージデータのセグメント化を開始します。 Swift では、アップロードされる単一のオブジェクトのサイズに上限があります。デフォルトでは、これは 5 GB です。この制限を超えるオブジェクトをアップロードするには、オブジェクトをマニフェストファイルと関連付けられる複数の小さなオブジェクトにセグメント化します。詳細は、https://docs.openstack.org/swift/latest/overview_large_objects.htmlを参照してください。 この設定オプションは、サイズのしきい値を指定します。これを超えると、Swift ドライバーはイメージデータを複数の小さなファイルにセグメント化する処理を開始します。現在、Swift ドライバーは Dynamic Large Objects の作成のみをサポートしています。  **注記** これは、該当する Swift クラスターによって適用される大型オブジェクトの制限を考慮して設定する必要があります。 値: ● 該当する Swift クラスターによって適用される大型オブジェクト制限以下の正の整数。 関連するオプション: ● **swift_store_large_object_chunk_size**

設定オプション = デフォルト値	型	説明
swift_store_multi_tenant = False	ブール値	イメージをテナントの Swift アカウントに保存します。 これにより、マルチテナントストレージモードが有効になり、Glance イメージがテナント固有の Swift アカウントに保存されます。これが無効になっていると、Glance はすべてのイメージを独自のアカウントに保存します。マルチテナントストアの詳細については、https://wiki.openstack.org/wiki/GlanceSwiftTenantSpecificStorageを参照してください。 **注記** マルチテナントの swift ストアを使用している場合は、**swift_store_config_file** オプションで swift 設定ファイルを設定しないようにしてください。 値: - True - False 関連するオプション: - swift_store_config_file

設定オプション = デフォルト値	型	説明
swift_store_multiple_containers_seed = 0	整数値	イメージの保存に使用するコンテナーの数を示すシード。 単一テナントストアを使用する場合は、イメージを 1 つ以上のコンテナーに保存できます。0 に設定すると、すべてのイメージが 1 つのコンテナーに保存されます。1 から 32 の整数値に設定すると、イメージの保存に複数のコンテナーが使用されます。この設定オプションにより、作成されるコンテナーの数が決まります。使用されるコンテナーの合計数は 16^N となるため、この設定オプションが 2 に設定されている場合は、イメージの保存に 16^2=256 のコンテナーが使用されます。 命名規則に関する詳細は、**swift_store_container** を参照してください。複数のコンテナーの使用についての詳細は、https://specs.openstack.org/openstack/glance-specs/specs/kilo/swift-store-multiple-containers.htmlを参照してください。 **注記** このオプションは、swift_store_multi_tenant が無効になっている場合にのみ使用されます。 値: ● 32 以下の負の値ではない整数 関連するオプション: ● **swift_store_container** ● **swift_store_multi_tenant** ● **swift_store_create_container_on_put**

設定オプション = デフォルト値	型	説明
swift_store_region = None	文字列値	Glance が使用する Swift エンドポイントのリージョン。 Glance がイメージの保存のために接続できる Swift リージョンを表す文字列値を指定します。デフォルトでは、リージョンは設定されません。 Glance がストレージバックエンドとして Swift を使用して複数のエンドポイントを持つ特定のテナントのイメージを保存する場合、**swift_store_region** で Swift リージョンを設定すると、Glance は単一リージョン接続ではなく、指定したリージョンの Swift に接続できるようになります。 このオプションは、単一テナントおよびマルチテナントストレージの両方に対して設定できます。 **注記** **swift_store_region** を使用したリージョンの設定はテナント固有のもので、テナントが異なるリージョンにまたがる複数のエンドポイントを持つ **場合にのみ** 必要です。 値: ● 有効な Swift リージョンを表す文字列値。 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_retry_get_count = 0	整数値	Swift のダウンロードがリトライされる回数。これを超えると、リクエストは失敗します。 エラーが発生する前にイメージのダウンロードをリトライする必要のある回数を表す整数値を指定します。デフォルト値はゼロです (イメージのダウンロードに失敗した場合、リトライされません)。正の整数値に設定すると、**swift_store_retry_get_count** により、ダウンロードの失敗時にこの回数ダウンロードが試みられます。これを超えると、エラーメッセージが送信されます。 値: ● ゼロ ● 正の整数値 関連するオプション: ● なし
swift_store_service_type = object-store	文字列値	使用する Swift サービスの種別。 Swift バックエンドストレージの使用中、イメージの保存に使用するサービス種別を表す文字列値を指定します。デフォルトのサービス種別は **object-store** に設定されます。 **注記** **swift_store_auth_version** が 2 に設定されている場合、この設定オプションの値は **object-store** である必要があります。より新しいバージョンの Keystone または別の認証スキームを使用している場合は、このオプションを変更できます。 値: ● Swift ストレージ用の有効なサービス種別を表す文字列。 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_ssl_compression = True	ブール値	HTTPS Swift リクエストの SSL レイヤーの圧縮。 SSL レイヤーでイメージの HTTPS Swift リクエストを圧縮するかどうかを決定するブール値を指定します。デフォルトでは、圧縮は有効になっています。 Swift を Glance イメージストレージのバックエンドストアとして使用する場合は、このオプションを使用して HTTPS Swift リクエストの SSL レイヤーの圧縮を設定できます。False に設定すると、HTTPS Swift リクエストの SSL レイヤーの圧縮は無効になります。このオプションを無効にすると、すでに圧縮された形式のイメージ (qcow2 など) に関するパフォーマンスが向上します。 値: ● True ● False 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
swift_store_use_trusts = True	ブール値	マルチテナント Swift ストアの信頼を使用します。 マルチテナントストアが使用されている場合に、このオプションは、各追加/取得リクエストに対して信頼を作成するように Swift ストアに指示します。信頼を使用すると、Swift ストアは、データのアップロードまたはダウンロード中に、認証トークンの失効により発生する可能性がある問題を回避できます。 デフォルトでは、**swift_store_use_trusts** は **True**に設定されています (信頼の使用が有効)。**False** に設定すると、代わりに Swift 接続にユーザートークンが使用され、信頼の作成のオーバーヘッドがなくなります。 **注記** このオプションは、**swift_store_multi_tenant** が **True**に設定されている場合のみ考慮されます。 値: - True - False 関連するオプション: - swift_store_multi_tenant
swift_store_user = None	文字列値	Swift 認証サービスに対して認証するユーザー。

設定オプション = デフォルト値	型	説明
swift_upload_buffer_dir = None	文字列値	Swift にアップロードする前にイメージセグメントをバッファー処理するディレクトリー。 イメージセグメントが swift にアップロードされる前に一時的にバッファー処理される、glance ノード上のディレクトリーへの絶対パスを表す文字列値を指定します。 注記: ● これは、設定オプション **swift_buffer_on_upload** が True に設定されている場合にのみ必要です。 ● このディレクトリーは、**swift_store_large_object_chunk_size** と、特定の Glance ノードによって同時にアップロードできるイメージの最大数を念頭に置いてプロビジョニングする必要があります。 値: ● 絶対ディレクトリーパスを表す文字列値 関連するオプション: ● swift_buffer_on_upload ● swift_store_large_object_chunk_size
vmware_api_retry_count = 10	整数値	VMware API のリトライ回数。 この設定オプションは、接続関連の問題またはサーバー API 呼び出しのオーバーロード時に VMware ESX/VC サーバー API をリトライする回数を指定します。**retry forever** は指定できません。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし

設定オプション = デフォルト値	型	説明
vmware_ca_file = None	文字列値	CA バンドルファイルへの絶対パス。 この設定オプションにより、運用者はカスタム認証局ファイルを使用して ESX/vCenter 証明書を検証できます。 このオプションを設定すると、"vmware_insecure" オプションは無視され、指定された CA ファイルが ESX/vCenter サーバー証明書の認証に使用され、サーバーへのセキュアな接続が確立されます。 以下の値を使用できます。 ● CA ファイルへの有効な絶対パスである任意の文字列 関連するオプション: ● vmware_insecure
vmware_datastores = None	多値	イメージを保存できるデータストア。 この設定オプションは、VMWare ストアのバックエンドでイメージを保存できるデータストアを指定します。このオプションは、複数のデータストアを指定するために複数回指定できます。データストア名は、データセンターのパスの後に: で区切って指定する必要があります。オプションの重みをデータストア名の後に再度: で区切って指定して、優先順位を指定できます。したがって、必要な形式は <datacenter_path>:<datastore_name>:<optional_weight> になります。 イメージを追加する場合、イメージサイズが既知の場合に利用可能な空き領域が十分にない場合を除き、最も高い重みを持つデータストアが選択されます。重みが指定されない場合、これはゼロであると仮定され、ディレクトリーは最後に選択のために考慮されます。複数のデータストアの重みが同じ場合は、利用可能な空き領域が最も多いデータストアが選択されます。 以下の値を使用できます。 ● <datacenter_path>:<datastore_name>:<optional_weight>の形式の任意の文字列 関連するオプション:* なし

設定オプション = デフォルト値	型	説明
vmware_insecure = False	ブール値	ESX/vCenter サーバー証明書の検証を設定します。 この設定オプションはブール値を取り、ESX/vCenter サーバーの証明書を検証するかどうかを決定します。このオプションが True に設定されている場合、ESX/vCenter サーバー証明書は検証されません。このオプションが False に設定されている場合、デフォルトの CA トラストストアが検証に使用されます。 このオプションは、vmware_ca_file オプションが設定されている場合は無視されます。その場合、ESX/vCenter サーバー証明書は、"vmware_ca_file" オプションで指定したファイルを使用して検証されます。 以下の値を使用できます。 ● True ● False 関連するオプション: ● vmware_ca_file
vmware_server_host = None	ホストのアドレス値	ESX/ESXi または vCenter サーバーターゲットシステムのアドレス。 この設定オプションは、ESX/ESXi または vCenter サーバーターゲットシステムのアドレスを設定します。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。アドレスには、IP アドレス (127.0.0.1) または DNS 名 (www.my-domain.com) を含めることができます。 以下の値を使用できます。 ● 有効な IPv4 アドレスまたは IPv6 アドレス ● 有効な DNS 名 関連するオプション: ● vmware_server_username ● vmware_server_password

設定オプション = デフォルト値	型	説明
vmware_server_password = None	文字列値	サーバーのパスワード。 この設定オプションは、VMware ESX/ESXi または vCenter サーバーとの認証用のパスワードを取ります。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。 以下の値を使用できます。 ● "vmware_server_username" オプションを使用して指定されたユーザー名に対応するパスワードである任意の文字列 関連するオプション: ● vmware_server_host ● vmware_server_username
vmware_server_username = None	文字列値	サーバーのユーザー名。 この設定オプションは、VMware ESX/ESXi または vCenter サーバーとの認証用のユーザー名を取ります。このオプションは、VMware ストレージバックエンドを使用する場合に必要です。 以下の値を使用できます。 ● 適切な権限を持つユーザーのユーザー名である任意の文字列 関連するオプション: ● vmware_server_host ● vmware_server_password

設定オプション = デフォルト値	型	説明
vmware_store_image_dir = /openstack_glance	文字列値	データストアで glance イメージが保存されるディレクトリー。 この設定オプションは、VMware データストアで glance イメージが保存されるディレクトリーへのパスを指定します。このオプションが設定されていない場合には、glance イメージが保存されるデフォルトのディレクトリーは openstack_glance になります。 以下の値を使用できます。 ● ディレクトリーへの有効なパスである任意の文字列 関連するオプション: ● なし
vmware_task_poll_interval = 5	整数値	VMware ESX/VC サーバーで呼び出されたリモートタスクをポーリングする間隔 (秒単位)。 この設定オプションは、VMWare ESX/VC サーバー API 呼び出しの一部として、処理中の非同期タスクをポーリングする際のスリープ時間 (秒単位) を取ります。 以下の値を使用できます。 ● 任意の正の整数値 関連するオプション: ● なし

3.3.3. oslo_policy

以下の表は、**/etc/glance/glance-cache.conf** ファイルの **[oslo_policy]** グループで利用可能なオプションの概要をまとめたものです。

表3.31 oslo_policy

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
enforce_new_defaults = False	ブール値	このオプションは、ポリシーを評価するときに古い非推奨のデフォルトを使用するかどうかを制御します。**True** の場合、古い非推奨のデフォルトは評価されません。これは、既存のトークンが古いデフォルトで許可されているが、新しいデフォルトで許可されていない場合、許可されないことを意味します。新しいデフォルトと **scope_type** の利点を一緒に得ることができるように、このフラグを **enforce_scope** フラグと一緒に有効にすることを推奨します
enforce_scope = False	ブール値	このオプションは、ポリシーを評価する際にスコープを適用するかどうかを制御します。**True** の場合、リクエストで使用されるトークンのスコープは、適用されるポリシーの **scope_types** と比較されます。スコープが一致しない場合、**InvalidScope** 例外が発生します。**False** の場合、ポリシーが一致しないスコープで呼び出されていることを運用者に通知するメッセージが記録されます。
policy_default_rule = default	文字列値	デフォルトルール。要求されたルールが見つからない場合に適用されます。
policy_dirs = ['policy.d']	多値	ポリシー設定ファイルが保存されるディレクトリー。これは、config_dir オプションで定義される検索パス内の任意のディレクトリーへの相対パスか、絶対パスにすることができます。このディレクトリーの検索には、policy_file で定義されたファイルが存在している必要があります。存在しないディレクトリーまたは空のディレクトリーは無視されます。
policy_file = policy.yaml	文字列値	ロールを指定のサービスのパーミッションにマッピングするファイルの相対パスまたは絶対パス。相対パスは、このオプションを設定する設定ファイルに対する相対パスとして指定する必要があります。
remote_content_type = application/x-www-form-urlencoded	文字列値	REST ベースのポリシーチェックのデータを送受信するコンテンツ種別
remote_ssl_ca_crt_file = None	文字列値	REST ベースのポリシーチェック用の認証局証明書ファイルの絶対パス
remote_ssl_client_crt_file = None	文字列値	REST ベースのポリシーチェック用のクライアント証明書の絶対パス
remote_ssl_client_key_file = None	文字列値	REST ベースのポリシーチェック用のクライアント鍵ファイルの絶対パス

設定オプション = デフォルト値	型	説明
remote_ssl_verify_server_crt = False	ブール値	REST ベースのポリシーチェックのサーバーアイデンティティー検証

第4章 HEAT

以下の章では、**heat** サービスの設定オプションについて説明します。

4.1. HEAT.CONF

このセクションには、**/etc/heat/heat.conf** ファイルのオプションが含まれます。

4.1.1. DEFAULT

以下の表は、**/etc/heat/heat.conf** ファイルの **[DEFAULT]** グループで利用可能なオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
action_retry_limit = 5	整数値	リソースをエラー以外の状態にするためにリトライする回数。リトライを無効にするには 0 に設定します。
allow_trusts_redelegation = False	ブール値	再委譲を有効にして信頼を作成します。このオプションは、reauthentication_auth_method が trusts に設定されている場合にのみ使用されます。Heat が作成したすべての信頼は権限の借用と再委譲の両方を有効にして使用するため、このオプションを有効にすると、セキュリティーに影響が出る点に注意してください。アクセスするのに Heat が使用するトークンから信頼を作成しなければならない他のサービスがある場合に限り、有効にします。たとえば、信頼も使用するように設定された場合に、別のリージョンにある Aodh と Heat が例として挙げられます。
auth_encryption_key = notgood but just long enough i t	文字列値	データベースの認証情報を暗号化するのに使用される鍵。この鍵の長さは 32 文字である必要があります。
backdoor_port = None	文字列値	eventlet バックドアを有効にします。設定可能な値は 0、<port>、および <start>:<end> です。ここで、0 はランダムの tcp ポート番号でリッスンします。<port> は指定されたポート番号でリッスンします (そのポートが使用中の場合にはバックドアを有効にしません)。また、<start>:<end> は、指定されたポート番号の範囲で最も小さい未使用のポート番号でリッスンします。選択したポートは、サービスのログファイルに表示されます。

設定オプション = デフォルト値	型	説明
backdoor_socket = None	文字列値	指定されたパスを接続を受信できる unix ソケットとして使用して、eventlet バックドアを有効にします。このオプションは、**backdoor_port** と相互に排他的です。したがって、どちらか 1 つだけを指定する必要があります。両方を指定すると、このオプションの存在によってそのオプションの使用が上書きされます。パス {pid} の中は、現在のプロセスの PID で置き換えられます。
client_retry_limit = 2	整数値	クライアントが予想される断続的なエラーに遭遇した場合にリトライする回数。リトライを無効にするには 0 に設定します。
cloud_backend = heat.engine.clients.OpenStackClients	文字列値	クライアントバックエンドとして使用する完全修飾クラス名。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
control_exchange = openstack	文字列値	トピックがスコープ設定されるデフォルトの変換。transport_url オプションで指定した変換名で上書きできます。
convergence_engine = True	ブール値	コンバージェンスアーキテクチャーのエンジンを有効にします。このオプションのすべてのスタックは、コンバージェンスエンジンを使用して作成されます。
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。
default_deployment_signal_transport = CFN_SIGNAL	文字列値	デプロイメント出力値を使用して、サーバーが heat にシグナルを送信する方法に関するテンプレートのデフォルト。CFN_SIGNAL は、CFN キーペアの署名済み URL への HTTP POST を許可します (heat-api-cfn を有効にする必要があります)。TEMP_URL_SIGNAL は、HTTP PUT 経由でシグナル送信する Swift TempURL を作成します (TempURL をサポートする object-store エンドポイントが必要です)。HEAT_SIGNAL は、提供された keystone 認証情報を使用した Heat API resource-signal への呼び出しを許可します。ZAQAR_SIGNAL は、提供された keystone 認証情報を使用してシグラル送信する専用の zaqar キューを作成します。

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
default_notification_level = INFO	文字列値	送信通知のデフォルト通知レベル。
default_publisher_id = None	文字列値	送信通知のデフォルトの publisher_id。

設定オプション = デフォルト値	型	説明
default_software_config_transport = POLL_SERVER_CFN	文字列値	ソフトウェア設定に必要なメタデータをサーバーが受信する方法についてのテンプレートのデフォルト。POLL_SERVER_CFN は、提供されたキーペアで認証された cfn API アクション DescribeStackResource への呼び出しを許可します (heat-api-cfn を有効にする必要があります)。POLL_SERVER_HEAT は、提供された keystone 認証情報を使用した Heat API resource-show への呼び出しを許可します (keystone v3 API および stack_user_* 設定オプションの設定が必要です)。POLL_TEMP_URL は、ポーリングのメタデータを使用して Swift TempURL を作成し、設定します (TempURL をサポートする object-store エンドポイントが必要です)。ZAQAR_MESSAGE は専用の zaqar キューを作成し、ポーリングのメタデータを投稿します。
default_user_data_format = HEAT_CFNTOOLS	文字列値	user_data をサーバーにフォーマットする方法についてのテンプレートのデフォルト。HEAT_CFNTOOLS の場合、user_data は heat-cfntools cloud-init ブート設定データの一部としてバンドルされます。RAW の場合、user_data は変更されずに Nova に渡されます。SOFTWARE_CONFIG の場合、user_data はソフトウェア設定データの一部としてバンドルされ、メタデータは関連付けられた SoftwareDeployment リソースに由来します。
deferred_auth_method = trusts	文字列値	遅延認証方法、保存されたパスワード、または信頼を選択します。**非推奨となったバージョン:** 9.0.0 *理由:* keystone v3 と併用した場合、保存されたパスワードベースの遅延認証は機能しないため、サポートされません。
enable_cloud_watch_lite = False	ブール値	従来の OS::Heat::CWLiteAlarm リソースを有効にします。**非推奨となったバージョン:** 10.0.0 *理由:*Heat CloudWatch サービスは削除されました。
enable_stack_abandon = False	ブール値	Stack Abandon 機能のプレビューを有効にします。
enable_stack_adopt = False	ブール値	Stack Adopt 機能のプレビューを有効にします。
encrypt_parameters_and_properties = False	ブール値	データベースに保存する前に、非表示としてマークされたテンプレートパラメーターおよびすべてのリソース属性を暗号化します。

設定オプション = デフォルト値	型	説明
engine_life_check_timeout = 2	整数値	スタックのロックに使用される、エンジンの健全性チェックの RPC タイムアウト。
environment_dir = /etc/heat/environment.d	文字列値	環境ファイルを検索するディレクトリー。
error_wait_time = 240	整数値	エラーが発生してからタスクの実行がキャンセルされるまで、タスクが継続される時間 (秒単位)。
event_purge_batch_size = 200	整数値	スタックのイベントがパージされるたびに削除されるイベントの数を制御します。パージの頻度を犠牲にしてより多くのイベントを維持するには、このオプションを低く設定します。
executor_thread_pool_size = 64	整数値	エグゼキューターがスレッディングまたはイベントレットの場合のエグゼキュータースレッドプールのサイズ。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
graceful_shutdown_timeout = 60	整数値	サーバーを正常にシャットダウンするまでのタイムアウトを指定します。値をゼロにすると、endless の待機を意味します。
heat_metadata_server_url = None	文字列値	Heat メタデータサーバーの URL。注: この設定は、keystone カタログとは異なるエンドポイントを使用するインスタンスが必要な場合のみ必要です。
heat_stack_user_role = heat_stack_user	文字列値	heat テンプレートで定義されたユーザーの Keystone ロール。
heat_waitcondition_server_url = None	文字列値	Heat の待機条件サーバーの URL。
`heat_watch_server_url = `	文字列値	Heat CloudWatch サーバーの URL。**非推奨となったバージョン:** 10.0.0 *理由:*Heat CloudWatch サービスは削除されました。
hidden_stack_tags = ['data-processing-cluster']	リスト値	これらのタグ名を含むスタックは非表示になります。複数のタグをコンマ区切りのリストに指定する必要があります (例: hidden_stack_tags=hide_me,me_too)。

設定オプション = デフォルト値	型	説明
host = <based on operating system>	文字列値	エンジンノードの名前。これは不透明な ID にすることができます。必ずしもホスト名、FQDN、または IP アドレスではありません。
instance_connection_https_validate_certificates = 1	文字列値	SSL が使用される場合に、CFN/CW API へのインスタンス接続で証明書を検証します。
instance_connection_is_secure = 0	文字列値	https 経由の CFN/CW API へのインスタンス接続
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。
keystone_backend = heat.engine.clients.os.keystone.heat_keystoneclient.KsClientWrapper	文字列値	keystone バックエンドとして使用する完全修飾クラス名。
loadbalancer_template = None	文字列値	ビルトインロードバランサーのネスト化されたスタックのカスタムテンプレート。
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_options = True	ブール値	サービスの起動時 (DEBUG レベル) に、登録されているすべてのオプションの値をログに記録することを有効または無効にします。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_events_per_stack = 1000	整数値	スタックごとに利用可能な最大イベントの数。パージチェックは時間を無作為に 200/event_purge_batch_size パーセントで行われるため、実際のイベント数は少し大きい可能性があります。イベントがパージされると、古いイベントが削除されます。スタックごとの無制限のイベントの場合は 0 に設定します。
max_interface_check_attempts = 10	整数値	インターフェイスが接続されているか、または切断されているかどうかを確認する回数。
max_ironic_api_microversion = None	浮動小数点の値	クライアントプラグインの ironic API の最大バージョン。この制限により、max_ironic_api_microversion を超えるマイクロバージョン番号でサポートされている ironic 機能は使用できなくなります。
max_json_body_size = 1048576	整数値	JSON リクエストボディの最大 RAW バイトサイズ。max_template_size よりも大きくなければなりません。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
max_nested_stack_depth = 5	整数値	ネストされたスタックを使用する場合に許容される最大深度。
max_nova_api_microversion = None	浮動小数点の値	クライアントプラグインの nova API バージョンの最大数。この制約により、max_nova_api_microversion を超えるマイクロバージョン番号でサポートされる nova 機能はどれも利用できなくなります。
max_resources_per_stack = 1000	整数値	最上位のスタックごとに許容できる最大リソース。-1 は無制限に設定されます。
max_server_name_length = 53	整数値	nova で使用されるサーバー名の最大長。

設定オプション = デフォルト値	型	説明
max_stacks_per_tenant = 512	整数値	1 つのテナントが一度にアクティブにできるスタックの最大数。-1 は無制限を表します。
max_template_size = 524288	整数値	テンプレートの raw バイトの最大サイズ。
num_engine_workers = None	整数値	フォークおよび実行を行う heat-engine プロセスの数。デフォルトで、ホスト上の 4 個または CPU の数に設定されます。
observe_on_update = False	ブール値	更新すると、heat が、実際に収束されたテンプレートから既存のリソースプロパティーを収集し、更新されたテンプレートに収束します。
onready = None	文字列値	非推奨。
periodic_interval = 60	整数値	実行中の定期的なタスク間の秒数。
plugin_dirs = ['/usr/lib64/heat', '/usr/lib/heat', '/usr/local/lib/heat', '/usr/local/lib64/heat']	リスト値	プラグインを検索するディレクトリーのリスト。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
`reauthentication_auth_method = `	文字列値	長時間のタスクが完了するように、トークン期限切れ時の再認証を許可します。この設定は指定したあらゆるユーザートークンの有効期限に優先する点に注意してください。
region_name_for_services = None	文字列値	サービスエンドポイントの取得に使用されるデフォルトのリージョン名。

設定オプション = デフォルト値	型	説明
region_name_for_shared_services = None	文字列値	共有サービスエンドポイントのリージョン名。
rpc_conn_pool_size = 30	整数値	RPC 接続プールのサイズ。
rpc_ping_enabled = False	ブール値	ping 呼び出しに応答するエンドポイントを追加します。エンドポイントの名前は oslo_rpc_server_ping です。
rpc_response_timeout = 60	整数値	呼び出しからの応答を待つ秒数。
run_external_periodic_tasks = True	ブール値	一部の定期的なタスクは個別のプロセスで実行できます。ここで実行すべきですか？
`server_keystone_endpoint_type = `	文字列値	設定されている場合は、これが使用されて、ユーザーが制御したサーバーが Heat をコールバックするのに使用する認証エンドポイントを制御します。未設定の場合、www_authenticate_uri が使用されます。
shared_services_types = ['image', 'volume', 'volumev2']	リスト値	他のリージョンにある共有サービス。これを有効にするには、region_name_for_shared_services オプションを設定する必要があります。
stack_action_timeout = 3600	整数値	スタックアクションのタイムアウト (例: 作成または更新)。
stack_domain_admin = None	文字列値	Keystone ユーザー名。stack_user_domain のユーザーとプロジェクトを管理するのに十分なロールを持つユーザーです。
stack_domain_admin_password = None	文字列値	stack_domain_admin ユーザーの Keystone パスワード

設定オプション = デフォルト値	型	説明
stack_scheduler_hints = False	ブール値	この機能を有効にすると、サーバーまたはボリュームリソースの heat スタックコンテキストを特定するためのスケジューラーヒントを nova および cinder の設定済みスケジューラーに渡して、作成には heat リソースタイプ OS::Cinder::Volume、OS::Nova::Server、および AWS::EC2::Instance を使用します。heat_root_stack_id は、リソースのルートスタックの id に設定されます。heat_stack_id はリソースの親スタックの id に設定されます。heat_stack_name はリソースの親スタックの名前に設定されます。heat_path_in_stack は、stackresourcename のコンマ区切りの文字列のリストに設定され、list[0] は **rootstackname** で、heat_resource_name はリソースの名前に設定されます。heat_resource_uuid はリソースのオーケストレーション ID に設定されます。
stack_user_domain_id = None	文字列値	heat テンプレートで定義されたユーザーが含まれる Keystone ドメイン ID。このオプションを設定すると、stack_user_domain_name オプションは無視されます。
stack_user_domain_name = None	文字列値	heat テンプレートで定義されたユーザーが含まれる Keystone ドメイン名。**stack_user_domain_id** オプションが設定されている場合、このオプションは無視されます。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
template_dir = /etc/heat/templates	文字列値	テンプレートファイルを検索するディレクトリー。
transport_url = rabbit://	文字列値	メッセージングバックエンドに接続するためのネットワークアドレスおよびオプションのユーザー認証情報 (URL 形式)。想定される形式は次のとおりです。 driver://[user:pass@]host:port[,[userN:passN@]hostN:portN]/virtual_host?query 例:rabbit://rabbitmq:password@127.0.0.1:5672// URL のフィールドの詳細は、https://docs.openstack.org/oslo.messaging/latest/reference/transport.html で oslo_messaging.TransportURL のドキュメントを参照してください。

設定オプション = デフォルト値	型	説明
trusts_delegated_roles = []	リスト値	heat に委任される信頼ロールのサブセット。未設定のままにすると、スタックの作成時にユーザーのロールがすべて heat に委任されます。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

4.1.2. auth_password

以下の表は、**/etc/heat/heat.conf** ファイルの **[auth_password]** グループで使用できるオプションの概要をまとめたものです。

表4.1 auth_password

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
allowed_auth_uris = []	リスト値	multi_cloud が有効な場合に auth_uri に許可される keystone エンドポイント。少なくとも 1 つのエンドポイントを指定する必要があります。
multi_cloud = False	ブール値	複数のクラウドのオーケストレーションを許可します。

4.1.3. clients

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients]** グループで使用できるオプションの概要をまとめたものです。

表4.2 clients

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = publicURL	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = False	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.4. clients_aodh

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_aodh]** グループで使用できるオプションの概要をまとめたものです。

表4.3 clients_aodh

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。

設定オプション = デフォルト値	型	説明
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.5. clients_barbican

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_barbican]** グループで使用できるオプションの概要をまとめたものです。

表4.4 clients_barbican

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.6. clients_cinder

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_cinder]** グループで使用できるオプションの概要をまとめたものです。

表4.5 clients_cinder

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。

設定オプション = デフォルト値	型	説明
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
http_log_debug = False	ブール値	クライアントのデバッグログ出力を許可します。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.7. clients_designate

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_designate]** グループで使用できるオプションの概要をまとめたものです。

表4.6 clients_designate

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.8. clients_glance

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_glance]** グループで使用できるオプションの概要をまとめたものです。

表4.7 clients_glance

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.9. clients_heat

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_heat]** グループで使用できるオプションの概要をまとめたものです。

表4.8 clients_heat

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。
`url = `	文字列値	http://0.0.0.0:8004/v1/%(tenant_id)s などの形式のオプションの heat url。

4.1.10. clients_keystone

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_keystone]** グループで使用できるオプションの概要をまとめたものです。

表4.9 clients_keystone

設定オプション = デフォルト値	型	説明
`auth_uri = `	文字列値	バージョンを指定しない keystone URL (形式は http://0.0.0.0:5000)。
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.11. clients_magnum

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_magnum]** グループで使用できるオプションの概要をまとめたものです。

表4.10 clients_magnum

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.12. clients_manila

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_manila]** グループで使用できるオプションの概要をまとめたものです。

表4.11 clients_manila

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.13. clients_mistral

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_mistral]** グループで使用できるオプションの概要をまとめたものです。

表4.12 clients_mistral

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.14. clients_monasca

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_monasca]** グループで使用できるオプションの概要をまとめたものです。

表4.13 clients_monasca

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.15. clients_neutron

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_neutron]** グループで使用できるオプションの概要をまとめたものです。

表4.14 clients_neutron

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.16. clients_nova

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_nova]** グループで使用できるオプションの概要をまとめたものです。

表4.15 clients_nova

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
http_log_debug = False	ブール値	クライアントのデバッグログ出力を許可します。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.17. clients_octavia

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_octavia]** グループで使用できるオプションの概要をまとめたものです。

表4.16 clients_octavia

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.18. clients_sahara

以下の表では、**/etc/heat/heat.conf** ファイルの **[clients_sahara]** グループで利用可能なオプションの概要を説明します。

表4.17 clients_sahara

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.19. clients_senlin

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_senlin]** グループで使用できるオプションの概要をまとめたものです。

表4.18 clients_senlin

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.20. clients_swift

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_swift]** グループで使用できるオプションの概要をまとめたものです。

表4.19 clients_swift

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.21. clients_trove

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_trove]** グループで使用できるオプションの概要をまとめたものです。

表4.20 clients_trove

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.22. clients_vitrage

次の表は、**/etc/heat/heat.conf** ファイルの **[clients_vitrage]** グループで使用できるオプションの概要を示しています。

表4.21 clients_vitrage

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.23. clients_zaqar

以下の表は、**/etc/heat/heat.conf** ファイルの **[clients_zaqar]** グループで使用できるオプションの概要をまとめたものです。

表4.22 clients_zaqar

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
endpoint_type = None	文字列値	OpenStack サービスとの通信に使用する Identity サービスカタログのエンドポイントのタイプ。
insecure = None	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。

4.1.24. cors

以下の表は、**/etc/heat/heat.conf** ファイルの **[cors]** グループで使用できるオプションの概要をまとめたものです。

表4.23 cors

設定オプション = デフォルト値	型	説明
allow_credentials = True	ブール値	実際の要求にユーザーの認証情報を含めることができることを示します。
allow_headers = ['X-Auth-Token', 'X-Identity-Status', 'X-Roles', 'X-Service-Catalog', 'X-User-Id', 'X-Tenant-Id', 'X-OpenStack-Request-ID']	リスト値	実際の要求時に使用されるヘッダーフィールド名を示します。
allow_methods = ['GET'、 'PUT'、 'POST'、 'DELETE'、 'PATCH']	リスト値	実際の要求時に使用できるメソッドを示します。
allowed_origin = None	リスト値	このリソースがリクエストの origin"ヘッダーで受信したドメインと共有されるかどうかを示します。形式:"<protocol>://<host>[:<port>]" (行末のスラッシュなし)例:https://horizon.example.com
expose_headers = ['X-Auth-Token', 'X-Subject-Token', 'X-Service-Token', 'X-OpenStack-Request-ID']	リスト値	API に安全に公開できるヘッダーを示します。デフォルトは HTTP Simple ヘッダーです。
max_age = 3600	整数値	CORS プリフライトリクエストの最大キャッシュ期間。

4.1.25. database

以下の表は、**/etc/heat/heat.conf** ファイルの **[database]** グループで使用できるオプションの概要をまとめたものです。

表4.24 database

設定オプション = デフォルト値	型	説明
backend = sqlalchemy	文字列値	データベースに使用するバックエンド。
connection = None	文字列値	データベースへの接続に使用する SQLAlchemy 接続文字列。
connection_debug = 0	整数値	SQL デバッグ情報の冗長性: 0=なし、100=すべて。

設定オプション = デフォルト値	型	説明
`connection_parameters = `	文字列値	接続時に接続 URL に追加するオプションの URL パラメーター。param1=value1&param2=value2&... として指定します。
connection_recycle_time = 3600	整数値	この秒数より長く接続プールに存在していた接続は、次にプールからチェックアウトされたときに新しい接続に置き換えられます。
connection_trace = False	ブール値	Python スタックトレースをコメント文字列として SQL に追加します。
db_inc_retry_interval = True	ブール値	True の場合には、db_max_retry_interval までのデータベース操作の再試行回数を長くします。
db_max_retries = 20	整数値	エラーが発生する前に接続エラーまたはデッドロックが発生した場合の最大再試行。再試行回数を無限に指定するには -1 に設定します。
db_max_retry_interval = 10	整数値	db_inc_retry_interval が設定されている場合は、データベース操作を再試行するまでの最大秒数です。
db_retry_interval = 1	整数値	データベーストランザクションの再試行間の秒数。
max_overflow = 50	整数値	設定されている場合、この値を SQLAlchemy で max_overflow に使用します。
max_pool_size = 5	整数値	プールに開いたままにする SQL 接続の最大数。値を 0 に設定すると無制限を意味します。
max_retries = 10	整数値	起動時のデータベース接続の最大再試行数。再試行回数を無限に指定するには -1 に設定します。
mysql_enable_ndb = False	ブール値	True の場合には、MySQL Cluster (NDB) の処理のサポートを透過的に有効にします。
mysql_sql_mode = TRADITIONAL	文字列値	MySQL セッションに使用する SQL モード。このオプション (デフォルトを含む) は、サーバーセット SQL モードを上書きします。サーバー設定で設定された SQL モードを使用するには、これを no value に設定します。例: mysql_sql_mode=
pool_timeout = None	整数値	設定されている場合は、この値を SQLAlchemy で pool_timeout に使用します。
retry_interval = 10	整数値	SQL 接続を開く再試行の間隔。

設定オプション = デフォルト値	型	説明
slave_connection = None	文字列値	スレーブデータベースへの接続に使用する SQLAlchemy 接続文字列。
sqlite_synchronous = True	ブール値	True の場合、SQLite は同期モードを使用します。
use_db_reconnect = False	ブール値	失われた接続でデータベースの実験的な使用を有効にします。

4.1.26. ec2authtoken

以下の表は、**/etc/heat/heat.conf** ファイルの **[ec2authtoken]** グループで使用できるオプションの概要をまとめたものです。

表4.25 ec2authtoken

設定オプション = デフォルト値	型	説明
allowed_auth_uris = []	リスト値	multi_cloud が有効な場合に auth_uri に許可される keystone エンドポイント。少なくとも 1 つのエンドポイントを指定する必要があります。
auth_uri = None	文字列値	認証エンドポイント URI。
ca_file = None	文字列値	SSL 接続で使用するオプションの CA 証明書ファイル。
cert_file = None	文字列値	オプションの PEM 形式の証明書チェーンファイル。
insecure = False	ブール値	設定されている場合には、サーバーの証明書は検証されません。
key_file = None	文字列値	秘密鍵が含まれるオプションの PEM 形式のファイル。
multi_cloud = False	ブール値	複数のクラウドのオーケストレーションを許可します。

4.1.27. eventlet_opts

以下の表は、**/etc/heat/heat.conf** ファイルの **[eventlet_opts]** グループで使用できるオプションの概要をまとめたものです。

表4.26 eventlet_opts

設定オプション = デフォルト値	型	説明
client_socket_timeout = 900	整数値	クライアント接続のソケット操作のタイムアウト。受信接続がこの秒数の間アイドル状態であると、閉じられます。0 の値は、永久に待機することを意味します。
wsgi_keep_alive = True	ブール値	False の場合は、クライアントのソケット接続を明示的に閉じます。

4.1.28. Healthcheck

以下の表は、**/etc/heat/heat.conf** ファイルの **[healthcheck]** グループで使用できるオプションの概要をまとめたものです。

表4.27 Healthcheck

設定オプション = デフォルト値	型	説明
backends = []	リスト値	ヘルスチェックを実施し、リクエストの一部としてその情報を報告できる追加のバックエンド。
detailed = False	ブール値	応答の一部として詳細情報を表示します。セキュリティーメモ: このオプションを有効にすると、監視対象のサービスに関する機密情報が漏洩する可能性があります。セキュリティーポリシーに違反しないことを確認してください。
disable_by_file_path = None	文字列値	ファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを確認します。DisableByFileHealthcheck プラグインで使用されます。
disable_by_file_paths = []	リスト値	ポートに基づいてファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを判断します。文字列の port:path リストを想定します。DisableByFilesPortsHealthcheck プラグインで使用されます。
path = /healthcheck	文字列値	healtcheck 要求に応答するパス。

4.1.29. heat_api

以下の表は、**/etc/heat/heat.conf** ファイルの **[heat_api]** グループで使用できるオプションの概要をまとめたものです。

表4.28 heat_api

設定オプション = デフォルト値	型	説明
backlog = 4096	整数値	ソケットを設定するバックログ要求の数。
bind_host = 0.0.0.0	IP アドレス値	サーバーをバインドするアドレス。特定のネットワークインターフェイスを選択する際に便利です。
bind_port = 8004	ポート値	サーバーがリッスンするポートです。
cert_file = None	文字列値	SSL モードに使用する SSL 証明書ファイルの場所。
key_file = None	文字列値	SSL モードの有効化に使用する SSL キーファイルの場所。
max_header_line = 16384	整数値	許可されるメッセージヘッダーの最大サイズ。大規模なトークンを使用する場合に max_header_line を増やす必要がある場合があります (通常は、大規模なサービスカタログで Keystone v3 API で生成されるメッセージヘッダー)。
tcp_keepidle = 600	整数値	ソケットオプション TCP_KEEPIDLE の値。TCP が keepalive プローブの送信を開始する前に、接続がアイドル状態でなければならない期間 (秒単位) です。
workers = 0	整数値	Heat サービスのワーカー数。デフォルト値の 0 は、サービスが server 上のコア数と同じワーカー数を開始することを意味します。

4.1.30. heat_api_cfn

以下の表は、**/etc/heat/heat.conf** ファイルの **[heat_api_cfn]** グループで使用できるオプションの概要をまとめたものです。

表4.29 heat_api_cfn

設定オプション = デフォルト値	型	説明
backlog = 4096	整数値	ソケットを設定するバックログ要求の数。
bind_host = 0.0.0.0	IP アドレス値	サーバーをバインドするアドレス。特定のネットワークインターフェイスを選択する際に便利です。
bind_port = 8000	ポート値	サーバーがリッスンするポートです。
cert_file = None	文字列値	SSL モードに使用する SSL 証明書ファイルの場所。

設定オプション = デフォルト値	型	説明
key_file = None	文字列値	SSL モードの有効化に使用する SSL キーファイルの場所。
max_header_line = 16384	整数値	許可されるメッセージヘッダーの最大サイズ。大規模なトークンを使用する場合に max_header_line を増やす必要がある場合があります (通常は、大規模なサービスカタログで Keystone v3 API で生成されるメッセージヘッダー)。
tcp_keepidle = 600	整数値	ソケットオプション TCP_KEEPIDLE の値。TCP が keepalive プローブの送信を開始する前に、接続がアイドル状態でなければならない期間 (秒単位) です。
workers = 1	整数値	Heat サービスのワーカー数。

4.1.31. heat_api_cloudwatch

以下の表は、**/etc/heat/heat.conf** ファイルの **[heat_api_cloudwatch]** グループで利用可能なオプションの概要をまとめたものです。

表4.30 heat_api_cloudwatch

設定オプション = デフォルト値	型	説明
backlog = 4096	整数値	ソケットを設定するバックログ要求の数。**非推奨となったバージョン:** 10.0.0 *理由:*Heat CloudWatch API が削除されました。
bind_host = 0.0.0.0	IP アドレス値	サーバーをバインドするアドレス。特定のネットワークインターフェイスを選択する際に便利です。**非推奨となったバージョン:** 10.0.0 *理由:*Heat CloudWatch API が削除されました。
bind_port = 8003	ポート値	サーバーがリッスンするポートです。**非推奨となったバージョン:** 10.0.0 *理由:*Heat CloudWatch API が削除されました。
cert_file = None	文字列値	SSL モードに使用する SSL 証明書ファイルの場所。**非推奨となったバージョン:** 10.0.0 *理由: *Heat CloudWatch API が削除されました。

設定オプション = デフォルト値	型	説明
key_file = None	文字列値	SSL モードの有効化に使用する SSL キーファイルの場所。**非推奨となったバージョン:** 10.0.0 *理由: *Heat CloudWatch API が削除されました。
max_header_line = 16384	整数値	許可されるメッセージヘッダーの最大サイズ。大規模なトークンを使用する場合に max_header_line を増やす必要がある場合があります (通常は、大規模なサービスカタログで Keystone v3 API で生成されるメッセージヘッダー)。**非推奨となったバージョン:** 10.0.0 *理由: *Heat CloudWatch API が削除されました。
tcp_keepidle = 600	整数値	ソケットオプション TCP_KEEPIDLE の値。TCP が keepalive プローブの送信を開始する前に、接続がアイドル状態でなければならない期間 (秒単位) です。**非推奨となったバージョン:** 10.0.0 *理由: *Heat CloudWatch API が削除されました。
workers = 1	整数値	Heat サービスのワーカー数。**非推奨となったバージョン:** 10.0.0 *理由: *Heat CloudWatch API が削除されました。

4.1.32. keystone_authtoken

以下の表は、**/etc/heat/heat.conf** ファイルの **[keystone_authtoken]** グループで使用できるオプションの概要をまとめたものです。

表4.31 keystone_authtoken

設定オプション = デフォルト値	型	説明
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ

設定オプション = デフォルト値	型	説明
auth_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。このオプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。**非推奨:**Queens 以降 *理由:* auth_uri オプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。
auth_version = None	文字列値	Identity API エンドポイントの API バージョン。
cache = None	文字列値	Swift キャッシュオブジェクトが保存される環境キーを要求します。auth_token ミドルウェアを Swift キャッシュと共にデプロイする場合は、このオプションを使用して、ミドルウェアが swift とキャッシングバックエンドを共有するようにします。それ以外の場合は、代わりに **memcached_servers** オプションを使用します。
cafile = None	文字列値	HTTPs 接続の検証時に使用する PEM でエンコードされた認証局。デフォルトはシステム CA です。
certfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
delay_auth_decision = False	ブール値	ミドルウェア内の認可要求を処理せず、承認の決定をダウンストリームの WSGI コンポーネントに委譲します。

設定オプション = デフォルト値	型	説明
enforce_token_bind = permissive	文字列値	トークンバインディングの使用および種別を制御するために使用されます。トークンバインディングのチェックを無効するには、"disabled"に設定します。バインドタイプがサーバーの認識する形式の場合にはバインディング情報を検証し、そうでない場合には無視するには、"permissive"(デフォルト) に設定します。"strict"は"permissive"と類似していますが、バインドタイプが不明な場合にはトークンが拒否されます。"required"の場合は、いずれかの形式のトークンバインディングが必要です。最後に、トークンに指定する必要のあるバインディングメソッドの名前。
http_connect_timeout = None	整数値	Identity API サーバーと通信する際の要求タイムアウト値。
http_request_max_retries = 3	整数値	Identity API サーバーと通信する際に再接続を試行する回数。
include_service_catalog = True	ブール値	(オプション)X-Service-Catalog ヘッダーを設定するかどうかを示します。False の場合、ミドルウェアはトークンの検証時にサービスカタログを要求せず、X-Service-Catalog ヘッダーを設定しません。
insecure = False	ブール値	HTTPS 接続を確認します。
interface = internal	文字列値	Identity API エンドポイントに使用するインターフェイス。有効な値は、"public"、"internal" (デフォルト)、または "admin" です。
keyfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
memcache_pool_conn_get_timeout = 10	整数値	(オプション) プールから memcached クライアント接続を取得するまで操作が待機する秒数。
memcache_pool_dead_retry = 300	整数値	(オプション)memcached サーバーが停止しているとみなされる秒数。この秒数が経過すると再試行されます。
memcache_pool_maxsize = 10	整数値	(オプション) すべての memcached サーバーへのオープン接続の最大合計数。
memcache_pool_socket_timeout = 3	整数値	(オプション)memcached サーバーと通信する際のソケットのタイムアウト (秒単位)。

設定オプション = デフォルト値	型	説明
memcache_pool_unused_timeout = 60	整数値	(オプション)memcached への接続がプール内で未使用の状態を維持する秒数。この秒数が経過すると終了されます。
memcache_secret_key = None	文字列値	(オプション、memcache_security_strategy が定義されている場合には必須) この文字列は鍵の導出に使用されます。
memcache_security_strategy = None	文字列値	(オプション) 定義されている場合は、トークンデータを認証、または認証して暗号化する必要があるかどうかを示します。MAC の場合、キャッシュでトークンデータが認証されます (HMAC を使用)。ENCRYPT の場合、キャッシュでトークンデータが暗号化され、認証されます。値がこれらのオプションのいずれでもない場合や空の場合には、auth_token は初期化時に例外を発生させます。
memcache_use_advanced_pool = False	ブール値	(オプション) 高度な (eventlet に対して安全な) memcached クライアントプールを使用します。高度なプールは python 2.x でのみ動作します。
memcached_servers = None	リスト値	オプションで、キャッシュに使用する memcached サーバーのリストを指定します。未定義のままの場合、トークンは代わりに処理中にキャッシュされます。
region_name = None	文字列値	アイデンティティーサーバーがあるリージョン。
service_token_roles = ['service']	リスト値	サービストークンに存在する必要があるロールの選択。サービストークンは、期限切れのトークンを使用できることを要求できるため、このチェックでは実際のサービスのみがこのトークンを送信するように厳密に制御する必要があります。ここでのロールは ANY チェックとして適用されるため、このリストのロールはすべて存在している必要があります。後方互換性の理由から、現在 allow_expired チェックにのみ影響します。
service_token_roles_required = False	ブール値	後方互換性の理由から、service_token_roles チェックを有効としてパスしない有効なサービストークンをパスさせる必要があります。これを true に設定することが今後のリリースでデフォルトとなり、可能な場合は有効にされる必要があります。
service_type = None	文字列値	サービスカタログに表示されるサービスの名前または種別。これは、制限されたアクセスルールがあるトークンを検証するために使用されます。

設定オプション = デフォルト値	型	説明
token_cache_time = 300	整数値	トークンの検証に過剰な時間を費やすのを防ぐために、ミドルウェアは、設定可能な期間 (秒単位) 中は以前に見たトークンをキャシュします。キャッシュを完全に無効にするには -1 に設定します。
www_authenticate_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。

4.1.33. noauth

以下の表は、**/etc/heat/heat.conf** ファイルの **[noauth]** グループで使用できるオプションの概要をまとめたものです。

表4.32 noauth

設定オプション = デフォルト値	型	説明
`token_response = `	文字列値	noauth ミドルウェアによって返されるコンテンツが含まれる JSON ファイル。

4.1.34. oslo_messaging_amqp

以下の表では、**/etc/heat/heat.conf** ファイルの **[oslo_messaging_amqp]** グループで利用可能なオプションの概要を説明します。

表4.33 oslo_messaging_amqp

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
addressing_mode = dynamic	文字列値	ドライバーが使用するアドレスモードを示します。許可される値:**legacy** - レガシーのルーティング不可能なアドレス指定を使用します。**routable** - ルーティング可能なアドレスを使用します。**dynamic** - メッセージバスがルーティングをサポートしていない場合はレガシーのアドレスを使用します。それ以外の場合は、ルーティング可能なアドレスを使用します。
anycast_address = anycast	文字列値	コンシューマーのグループに送信する際に、アドレス接頭辞に追加されます。コンシューマー間でラウンドロビン方式で配信される必要のあるメッセージを特定するために、メッセージバスで使用されます。
broadcast_prefix = broadcast	文字列値	すべてのサーバーにブロードキャストする場合に使用されるアドレス接頭辞
connection_retry_backoff = 2	整数値	フェイルオーバーの試みに失敗するたびに、connection_retry_interval をこの秒数だけ増やします。
connection_retry_interval = 1	整数値	再接続を試みる前に一時停止する秒数。
connection_retry_interval_max = 30	整数値	connection_retry_interval + connection_retry_backoff の上限
container_name = None	文字列値	AMQP コンテナーの名前。グローバルで一意でなければなりません。デフォルトは、生成された UUID です。
default_notification_exchange = None	文字列値	通知アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_notification_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **notify**
default_notify_timeout = 30	整数値	送信された通知メッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_reply_retry = 0	整数値	リカバリー可能なエラーが原因で失敗した返信メッセージを再送信する最大試行回数。

設定オプション = デフォルト値	型	説明
default_reply_timeout = 30	整数値	rpc 返信メッセージ配信の期限。
default_rpc_exchange = None	文字列値	RPC アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_rpc_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **rpc**
default_send_timeout = 30	整数値	rpc キャストまたは呼び出しメッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_sender_link_timeout = 600	整数値	アイドル状態の送信者リンクのパージをスケジュールする時間。期限切れ後にリンクの割り当てを解除します。
group_request_prefix = unicast	文字列値	グループの任意のサーバーに送信する際のアドレス接頭辞
idle_timeout = 0	整数値	非アクティブな接続のタイムアウト (秒単位)
link_retry_delay = 10	整数値	リカバリー可能なエラーが原因で失敗した AMQP 1.0 リンクの再接続間に一時停止する時間。
multicast_address = multicast	文字列値	ファンアウトメッセージを送信するときにアドレス接頭辞に追加されます。ファンアウトメッセージを識別するためにメッセージバスによって使用されます。
notify_address_prefix = openstack.org/om/notify	文字列値	すべての生成される通知アドレスのアドレス接頭辞
notify_server_credit = 100	整数値	受信通知メッセージのウィンドウサイズ
pre_settled = ['rpc-cast', 'rpc-reply']	多値	このタイプのメッセージを事前処置状態で送信します。事前処置されたメッセージは、ピアから確認応答を受信しません。注記: 事前処置されたメッセージは、配信に失敗した場合に警告なしに破棄される可能性があります。許可される値:**rpc-call** - RPC 呼び出しを事前処置状態で送信します。**rpc-reply**- RPC 返信を事前処置状態で送信します。**rpc-cast** - RPC キャストを事前処置状態で送信します。**notify** - 通知を事前処置状態で送信します。

設定オプション = デフォルト値	型	説明
pseudo_vhost = True	ブール値	仮想ホスト (qpidd など) をネイティブにサポートしていないメッセージバスの仮想ホストサポートを有効にします。true に設定すると、仮想ホスト名はすべてのメッセージバスアドレスに追加され、結果として仮想ホストごとにプライベート **サブネット** を作成します。メッセージバスが仮想ホストの名前として AMQP 1.0 オープン遂行動詞の **hostname** フィールドを使用して仮想ホストをサポートする場合は False に設定します。
reply_link_credit = 200	整数値	受信する RPC 返信メッセージのウィンドウサイズ。
rpc_address_prefix = openstack.org/om/rpc	文字列値	すべての生成される RPC アドレスのアドレス接頭辞
rpc_server_credit = 100	整数値	受信する RPC リクエストメッセージのウィンドウサイズ。
`sasl_config_dir = `	文字列値	SASL 設定が含まれるディレクトリーへのパス
`sasl_config_name = `	文字列値	設定ファイルの名前 (.conf 接尾辞なし)
`sasl_default_realm = `	文字列値	ユーザー名にレルムが存在しない場合に使用する SASL レルム
`sasl_mechanisms = `	文字列値	許可される SASL メカニズムのスペース区切りリスト
server_request_prefix = exclusive	文字列値	特定のサーバーに送信する時に使用するアドレス接頭辞
ssl = False	ブール値	SSL 経由で接続を試みます。その他の ssl 関連のパラメーターが指定されていない場合、システムの CA バンドルを使用してサーバーの証明書を検証します。
`ssl_ca_file = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_cert_file = `	文字列値	クライアント認証用の自己識別証明書 PEM ファイル
`ssl_key_file = `	文字列値	ssl_cert_file 証明書の署名に使用される秘密鍵 PEM ファイル (オプション)
ssl_key_password = None	文字列値	ssl_key_file を復号するためのパスワード (暗号化されている場合)

設定オプション = デフォルト値	型	説明
ssl_verify_vhost = False	ブール値	デフォルトでは、SSL は、サーバーの証明書の名前が transport_url のホスト名と一致することを確認します。設定によっては、代わりに仮想ホスト名を使用することが望ましい場合があります。たとえば、サーバーが Server Name Indication TLS 拡張 (rfc6066) を使用して、仮想ホストごとの証明書を提供する場合などです。サーバーの SSL 証明書が DNS 名ではなく仮想ホスト名を使用する場合は、ssl_verify_vhost を True に設定します。
trace = False	ブール値	デバッグ: AMQP フレームを標準出力 (stdout) にダンプします。
unicast_address = unicast	文字列値	特定の RPC/通知サーバーに送信する際に、アドレス接頭辞に追加されます。単一の送信先に送信されたメッセージを識別するためにメッセージバスによって使用されます。

4.1.35. oslo_messaging_kafka

以下の表は、**/etc/heat/heat.conf** ファイルの **[oslo_messaging_kafka]** グループで使用できるオプションの概要をまとめたものです。

表4.34 oslo_messaging_kafka

設定オプション = デフォルト値	型	説明
compression_codec = none	文字列値	プロデューサーによって生成されたすべてのデータの圧縮コーデック。設定されていない場合、圧縮は使用されません。この設定に許可される値は、kafka バージョンに依存することに注意してください。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
consumer_group = oslo_messaging_consumer	文字列値	Kafka コンシューマーのグループ ID。あるグループのコンシューマーは、メッセージ消費を調整します。
enable_auto_commit = False	ブール値	非同期コンシューマーコミットを有効にします。
kafka_consumer_timeout = 1.0	浮動小数点の値	Kafka コンシューマーのデフォルトタイムアウト

設定オプション = デフォルト値	型	説明
kafka_max_fetch_bytes = 1048576	整数値	Kafka コンシューマーの最大フェッチバイト
max_poll_records = 500	整数値	ポーリング呼び出しで返されるレコードの最大数
pool_size = 10	整数値	Kafka コンシューマーのプールサイズ
producer_batch_size = 16384	整数値	プロデューサーの非同期送信のバッチサイズ
producer_batch_timeout = 0.0	浮動小数点の値	KafkaProducer バッチ処理の遅延の上限 (秒単位)
sasl_mechanism = PLAIN	文字列値	セキュリティープロトコルが SASL である際のメカニズム
security_protocol = PLAINTEXT	文字列値	ブローカーとの通信に使用されるプロトコル
`ssl_cafile = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_client_cert_file = `	文字列値	認証に使用されるクライアント証明書の PEM ファイル。
`ssl_client_key_file = `	文字列値	認証に使用されるクライアントキーの PEM ファイル。
`ssl_client_key_password = `	文字列値	認証に使用されるクライアントキーパスワードファイル。

4.1.36. oslo_messaging_notifications

以下の表は、**/etc/heat/heat.conf** ファイルの **[oslo_messaging_notifications]** グループで使用できるオプションの概要をまとめたものです。

表4.35 oslo_messaging_notifications

設定オプション = デフォルト値	型	説明
driver = []	多値	通知の送信を処理するドライバー。使用できる値は messaging、messagingv2、routing、log、test、noop です。

設定オプション = デフォルト値	型	説明
retry = -1	整数値	リカバリー可能なエラーが原因で配信に失敗した通知メッセージを再送信する最大試行回数。0 - 再試行なし、-1 - 無限回
topics = ['notifications']	リスト値	OpenStack の通知に使用する AMQP トピック。
transport_url = None	文字列値	通知に使用するメッセージングドライバーを表す URL。設定されていない場合は、RPC に使用されるものと同じ設定にフォールバックします。

4.1.37. oslo_messaging_rabbit

以下の表は、**/etc/heat/heat.conf** ファイルの **[oslo_messaging_rabbit]** グループで使用できるオプションの概要をまとめたものです。

表4.36 oslo_messaging_rabbit

設定オプション = デフォルト値	型	説明
amqp_auto_delete = False	ブール値	AMQP の自動削除キュー。
amqp_durable_queues = False	ブール値	AMQP で永続キューを使用します。
direct_mandatory_flag = True	ブール値	(非推奨) ダイレクト送信のための RabbitMQ 必須フラグを有効/無効にします。直接送信は応答として使用されるため、クライアントキューが存在しない場合には MessageUndeliverable 例外が発生します。MessageUndeliverable 例外はタイムアウトをループして、送信者が回復する可能性が高くなります。このフラグは非推奨になり、この機能を非アクティブ化することはできません。
enable_cancel_on_failover = False	ブール値	x-cancel-on-ha-failover フラグを有効にして、rabbitmq サーバーがキューが停止しているときにコンシューマーをキャンセルし、通知できるようにします。

設定オプション = デフォルト値	型	説明
heartbeat_in_pthread = False	ブール値	デフォルトでは、ネイティブ Python スレッドを介してヘルスチェックハートビートスレッドを実行します。このオプションが False の場合、ヘルスチェックハートビートは親プロセスから実行モデルを継承します。たとえば、親プロセスが eventlet/greenlet を使用して stdlib にモンキーパッチを適用した場合、ハートビートはグリーンスレッドを介して実行されます。This option should be set to True only for the wsgi services.
heartbeat_rate = 2	整数値	heartbeat_timeout_threshold 中、ハートビートを確認する回数。
heartbeat_timeout_threshold = 60	整数値	ハートビートの keep-alive が失敗した場合に Rabbit ブローカーがダウンとみなされるまでの秒数 (0 はハートビートを無効にします)。
kombu_compression = None	文字列値	実験的用途: 許容値は gzip、bz2 です。設定されていない場合、圧縮は使用されません。このオプションは、今後のバージョンで利用できない可能性があります。
kombu_failover_strategy = round-robin	文字列値	現在接続しているノードが利用できなくなった場合に、次の RabbitMQ ノードを選択する方法を指定します。設定で複数の RabbitMQ ノードが指定される場合に限り有効になります。
kombu_missing_consumer_retry_timeout = 60	整数値	応答のないクライアントが応答を送信するのを待つ時間。この時間が経過すると無視します。この値は rpc_response_timeout より長くすることはできません。
kombu_reconnect_delay = 1.0	浮動小数点の値	AMQP コンシューマーの取り消し通知への応答で、再接続するまでの待機時間。
rabbit_ha_queues = False	ブール値	RabbitMQ (x-ha-policy: all) の HA キューの使用を試みます。このオプションを変更する場合は、RabbitMQ データベースを消去する必要があります。RabbitMQ 3.0 では、キューを宣言する際に x-ha-policy 引数によってキューのミラーリングが制御されなくなりました。すべてのキュー (自動生成された名前のキューを除く) がすべてのノードでミラーリングされるようにするには、"rabbitmqctl set_policy HA **^(?!amq\.).* {"ha-mode": "all"}**" を実行します。
rabbit_interval_max = 30	整数値	RabbitMQ 接続を再試行する最大間隔。デフォルトは 30 秒です。

設定オプション = デフォルト値	型	説明
rabbit_login_method = AMQPLAIN	文字列値	RabbitMQ ログイン方法
rabbit_qos_prefetch_count = 0	整数値	事前フェッチするメッセージの数を指定します。ゼロに設定すると、無制限のメッセージのフェッチが許可されます。
rabbit_retry_backoff = 2	整数値	RabbitMQ に接続する際に再試行間でバックオフする長さ
rabbit_retry_interval = 1	整数値	RabbitMQ との接続を再試行する頻度。
rabbit_transient_queues_ttl = 1800	整数値	キューの TTL (x-expires) の期間 (秒単位) を表す正の整数。TTL の期間使用されないキューは自動的に削除されます。このパラメーターは応答キューとファンアウトキューにのみ影響します。
ssl = False	ブール値	SSL 経由で接続します。
`ssl_ca_file = `	文字列値	SSL 認証局ファイル (SSL が有効な場合にのみ有効)。
`ssl_cert_file = `	文字列値	SSL 証明書ファイル (SSL が有効な場合にのみ有効)。
`ssl_key_file = `	文字列値	SSL キーファイル (SSL が有効な場合にのみ有効)。
`ssl_version = `	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

4.1.38. oslo_middleware

以下の表は、**/etc/heat/heat.conf** ファイルの **[oslo_middleware]** グループで使用できるオプションの概要をまとめたものです。

表4.37 oslo_middleware

設定オプション = デフォルト値	型	説明
enable_proxy_headers_parsing = False	ブール値	アプリケーションがプロキシーの背後であるかどうか。これにより、ミドルウェアがヘッダーを解析すべきかどうかを決定します。

設定オプション = デフォルト値	型	説明
max_request_body_size = 114688	整数値	各リクエストの本文の最大サイズ (バイト単位)。
secure_proxy_ssl_header = X-Forwarded-Proto	文字列値	SSL 終端プロキシーによって非表示であった場合でも、元のリクエストプロトコルスキームを決定するために使用される HTTP ヘッダー。

4.1.39. oslo_policy

以下の表は、**/etc/heat/heat.conf** ファイルの **[oslo_policy]** グループで使用できるオプションの概要をまとめたものです。

表4.38 oslo_policy

設定オプション = デフォルト値	型	説明
enforce_new_defaults = False	ブール値	このオプションは、ポリシーを評価するときに古い非推奨のデフォルトを使用するかどうかを制御します。**True** の場合、古い非推奨のデフォルトは評価されません。これは、既存のトークンが古いデフォルトで許可されているが、新しいデフォルトで許可されていない場合、許可されないことを意味します。新しいデフォルトと **scope_type** の利点を一緒に得ることができるように、このフラグを **enforce_scope** フラグと一緒に有効にすることを推奨します
enforce_scope = False	ブール値	このオプションは、ポリシーを評価する際にスコープを適用するかどうかを制御します。**True** の場合、リクエストで使用されるトークンのスコープは、適用されるポリシーの **scope_types** と比較されます。スコープが一致しない場合、**InvalidScope** 例外が発生します。**False** の場合、ポリシーが一致しないスコープで呼び出されていることを運用者に通知するメッセージが記録されます。
policy_default_rule = default	文字列値	デフォルトルール。要求されたルールが見つからない場合に適用されます。
policy_dirs = ['policy.d']	多値	ポリシー設定ファイルが保存されるディレクトリー。これは、config_dir オプションで定義される検索パス内の任意のディレクトリーへの相対パスか、絶対パスにすることができます。このディレクトリーの検索には、policy_file で定義されたファイルが存在している必要があります。存在しないディレクトリーまたは空のディレクトリーは無視されます。

設定オプション = デフォルト値	型	説明
policy_file = policy.json	文字列値	ロールを指定のサービスのパーミッションにマッピングするファイルの相対パスまたは絶対パス。相対パスは、このオプションを設定する設定ファイルに対する相対パスとして指定する必要があります。
remote_content_type = application/x-www-form-urlencoded	文字列値	REST ベースのポリシーチェックのデータを送受信するコンテンツ種別
remote_ssl_ca_crt_file = None	文字列値	REST ベースのポリシーチェック用の認証局証明書ファイルの絶対パス
remote_ssl_client_crt_file = None	文字列値	REST ベースのポリシーチェック用のクライアント証明書の絶対パス
remote_ssl_client_key_file = None	文字列値	REST ベースのポリシーチェック用のクライアント鍵ファイルの絶対パス
remote_ssl_verify_server_crt = False	ブール値	REST ベースのポリシーチェックのサーバーアイデンティティー検証

4.1.40. paste_deploy

以下の表は、**/etc/heat/heat.conf** ファイルの **[paste_deploy]** グループ下で使用できるオプションの概要をまとめたものです。

表4.39 paste_deploy

設定オプション = デフォルト値	型	説明
api_paste_config = api-paste.ini	文字列値	使用する API 貼り付け設定ファイル。
flavor = None	文字列値	使用するフレーバー。

4.1.41. profiler

以下の表は、**/etc/heat/heat.conf** ファイルの **[profiler]** グループで使用できるオプションの概要をまとめたものです。

表4.40 profiler

設定オプション = デフォルト値	型	説明
connection_string = messaging://	文字列値	通知バックエンドの接続文字列。 デフォルト値は **messaging://** で、通知機能を oslo_messaging に設定します。 使用できる値の例: ● **messaging://**: スパンの送信に oslo_messaging ドライバーを使用します。 ● **redis://127.0.0.1:6379**: スパンの送信に redis ドライバーを使用します。 ● **mongodb://127.0.0.1:27017**: スパンの送信に mongodb ドライバーを使用します。 ● **elasticsearch://127.0.0.1:9200:** elasticsearch ドライバーを使用してスパンを送信します。 ● **jaeger://127.0.0.1:6831**: スパンの送信に jaeger トレースを使用します。
enabled = False	ブール値	このノード上の全サービスのプロファイリングを有効にします。 デフォルト値は False です (プロファイリング機能を無効にします)。 値: ● True: 機能を有効にします ● False: 機能を無効にします。このプロジェクトの操作ではプロファイリングを開始できません。プロファイリングが別のプロジェクトでトリガーされた場合には、このプロジェクトの部分は空になります。
es_doc_type = notification	文字列値	elasticsearch の通知インデックスについてのドキュメントタイプ。
es_scroll_size = 10000	整数値	elasticsearch は大規模な要求をバッチに分割します。このパラメーターは、各バッチの最大サイズを定義します (例: es_scroll_size=10000)。
es_scroll_time = 2m	文字列値	このパラメーターは時間値パラメーター (es_scroll_time=2m など) です。これは、検索に参加するノードが継続してサポートするために関連するリソースを維持する時間を示します。

設定オプション = デフォルト値	型	説明
filter_error_trace = False	ブール値	エラー/例外を含むフィルタートレースを分離された場所へ有効にします。 デフォルト値は False に設定されます。 値: ● True: エラー/例外を含むフィルタートレースを有効にします。 ● False: フィルターを無効にします。
hmac_keys = SECRET_KEY	文字列値	パフォーマンスプロファイリング用にコンテキストデータの暗号化に使用するシークレットキー。 この文字列値は <key1>[,<key2>,...<keyn>] の形式にする必要があります。各キーはランダムな文字列です。REST API 経由でプロファイリングをトリガーするユーザーは、REST API 呼び出しのヘッダーにこれらのキーの 1 つを設定し、この特定プロジェクトのノードのプロファイリング結果が含まれるようにする必要があります。 プロファイリングを有効にするには、enabled フラグと hmac_keys 設定オプションの両方を設定する必要があります。また、少なくとも 1 つのキーで全サービス間で正しいプロファイリング情報を生成するには、OpenStack プロジェクト間で一貫性を確保する必要があります。これにより、クライアント側からを使用して、すべての可能なリソースからの情報を含むトレースを生成できます。
sentinel_service_name = mymaster	文字列値	Redissentinel はサービス名を使用してマスターの redis サービスを識別します。このパラメーターは名前を定義します (例: **sentinal_service_name=mymaster**)。
socket_timeout = 0.1	浮動小数点の値	Redissentinel は、接続に timeout オプションを提供します。このパラメーターは、タイムアウトを定義します (例: socket_timeout=0.1)。

設定オプション = デフォルト値	型	説明
trace_sqlalchemy = False	ブール値	サービスで SQL 要求のプロファイリングを有効にします。 デフォルト値は False です (SQL 要求はトレースされません)。 値: ● True: SQL 要求のプロファイリングを有効にします。各 SQL クエリーはトレースの一部となり、それにどの程度の時間を費やしたかによって分析できます。 ● False: SQL 要求のプロファイリングを無効にします。費やした時間は、高いレベルの操作でのみ表示されます。単一の SQL クエリーをこのように分析できません。

4.1.42. revision

以下の表は、**/etc/heat/heat.conf** ファイルの **[revision]** グループで使用できるオプションの概要をまとめたものです。

表4.41 revision

設定オプション = デフォルト値	型	説明
heat_revision = unknown	文字列値	Heat ビルドのリビジョンビルドのリビジョンを個別に管理する場合には、このセクションを別のファイルに移動し、別の設定オプションとして追加することができます。

4.1.43. ssl

以下の表は、**/etc/heat/heat.conf** ファイルの **[ssl]** グループで使用できるオプションの概要をまとめたものです。

表4.42 ssl

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	接続クライアントの検証に使用する CA 証明書ファイル。
cert_file = None	文字列値	サーバーをセキュアに起動する際に使用する証明書ファイル。

設定オプション = デフォルト値	型	説明
ciphers = None	文字列値	利用可能な暗号のリストを設定します。値は OpenSSL 暗号リスト形式の文字列である必要があります。
key_file = None	文字列値	サーバーをセキュアに起動する際に使用する秘密鍵ファイル。
version = None	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

4.1.44. trustee

以下の表は、**/etc/heat/heat.conf** ファイルの **[trustee]** グループで使用できるオプションの概要をまとめたものです。

表4.43 trustee

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID

設定オプション = デフォルト値	型	説明
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
system-scope = None	文字列値	システム操作のスコープ
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名

4.1.45. volumes

以下の表は、**/etc/heat/heat.conf** ファイルの **[volumes]** グループで使用できるオプションの概要をまとめたものです。

表4.44 volumes

設定オプション = デフォルト値	型	説明
backups_enabled = True	ブール値	cinder-backup サービスが有効かどうかを指定します。これは、cinder-backup サービスが検出可能になるまで一時的な回避策となります。LP#1334856 を参照してください。

第5章 IRONIC

以下の章では、**ironic** サービスの設定オプションについて説明します。

5.1. IRONIC.CONF

本項では、**/etc/ironic/ironic.conf** ファイルのオプションについて説明します。

5.1.1. DEFAULT

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
auth_strategy = keystone	文字列値	すべての認証が無効になるため、ironic-api が使用する認証ストラテジー。noauth は本番環境では使用しないでください。
backdoor_port = None	文字列値	eventlet バックドアを有効にします。設定可能な値は 0、<port>、および <start>:<end> です。ここで、0 はランダムの tcp ポート番号でリッスンします。<port> は指定されたポート番号でリッスンします (そのポートが使用中の場合にはバックドアを有効にしません)。また、<start>:<end> は、指定されたポート番号の範囲で最も小さい未使用のポート番号でリッスンします。選択したポートは、サービスのログファイルに表示されます。
backdoor_socket = None	文字列値	指定されたパスを接続を受信できる unix ソケットとして使用して、eventlet バックドアを有効にします。このオプションは、**backdoor_port** と相互に排他的です。したがって、どちらか 1 つだけを指定する必要があります。両方を指定すると、このオプションの存在によってそのオプションの使用が上書きされます。パス {pid} の中は、現在のプロセスの PID で置き換えられます。
bindir = $pybasedir/bin	文字列値	ironic バイナリーがインストールされているディレクトリー。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
control_exchange = openstack	文字列値	トピックがスコープ設定されるデフォルトの変換。transport_url オプションで指定した変換名で上書きできます。

設定オプション = デフォルト値	型	説明
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。
debug_tracebacks_in_api = False	ブール値	API 応答のサーバーのトレースバックをエラー応答として返します。警告: これは安全ではないため、実稼働環境では使用しないでください。
default_bios_interface = None	文字列値	bios_interface フィールドが設定されていないノードに使用されるデフォルトの bios インターフェイス。システムに存在する bios インターフェイスの完全なリストは、ironic.hardware.interfaces.bios のエントリーポイントを列挙することで確認することができます。
default_boot_interface = None	文字列値	boot_interface フィールドが設定されていないノードに使用されるデフォルトのブートインターフェイス。システムに存在するブートインターフェイスの完全なリストは、ironic.hardware.interfaces.boot のエントリーポイントを列挙することで確認することができます。
default_console_interface = None	文字列値	console_interface フィールドが設定されていないノードに使用されるデフォルトのコンソールインターフェイス。システムに存在するコンソールインターフェイスの完全なリストは、ironic.hardware.interfaces.console のエントリーポイントを列挙することで確認することができます。
default_deploy_interface = None	文字列値	deploy_interface フィールドが設定されていないノードに使用されるデフォルトのデプロイインターフェイス。システムに存在するデプロイインターフェイスの完全なリストは、ironic.hardware.interfaces.deploy エントリーポイントを列挙することで確認することができます。
default_inspect_interface = None	文字列値	inspect_interface フィールドが設定されていないノードに使用されるデフォルトの検査インターフェイス。システムに存在する検査インターフェイスの完全なリストは、ironic.hardware.interfaces.inspect のエントリーポイントを列挙することで確認することができます。

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARNING', 'amqplib=WARNING', 'qpid.messaging=INFO', 'oslo.messaging=INFO', 'oslo_messaging=INFO', 'sqlalchemy=WARNING', 'stevedore=INFO', 'eventlet.wsgi.server=INFO', 'iso8601=WARNING', 'requests=WARNING', 'glanceclient=WARNING', 'urllib3.connectionpool=WARNING', 'keystonemiddleware.auth_token=INFO', 'keystoneauth.session=INFO', 'openstack=WARNING']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
default_management_interface = None	文字列値	management_interface フィールドが設定されていないノードに使用されるデフォルトの管理インターフェイス。システムに存在する管理インターフェイスの完全なリストは、ironic.hardware.interfaces.management のエントリーポイントを列挙することで確認することができます。
default_network_interface = None	文字列値	network_interface フィールドが設定されていないノードに使用されるデフォルトのネットワークインターフェイス。システムに存在するネットワークインターフェイスの完全なリストは、ironic.hardware.interfaces.network エントリーポイントを列挙することで確認することができます。
default_portgroup_mode = active-backup	文字列値	portgroups のデフォルトモード。許可される値は、ボンディングに関する Linux カーネルのドキュメント (https://www.kernel.org/doc/Documentation/networking/bonding.txt) を参照してください。
default_power_interface = None	文字列値	power_interface フィールドが設定されていないノードに使用されるデフォルトの電源インターフェイス。システムに存在する電源インターフェイスの完全なリストは、ironic.hardware.interfaces.power のエントリーポイントを列挙することで確認することができます。

設定オプション = デフォルト値	型	説明
default_raid_interface = None	文字列値	raid_interface フィールドが設定されていないノードに使用されるデフォルトの RAID インターフェイス。システムに存在する raid インターフェイスの完全なリストは、ironic.hardware.interfaces.raid のエントリーポイントを列挙することで確認することができます。
default_rescue_interface = None	文字列値	rescue_interface フィールドが設定されていないノードに使用されるデフォルトのレスキューインターフェイス。システムに存在するレスキューインターフェイスの完全なリストは、ironic.hardware.interfaces.rescue のエントリーポイントを列挙することで確認することができます。
default_resource_class = None	文字列値	作成要求にリソースクラスが指定されていない場合に、新規ノードに使用するリソースクラス。
default_storage_interface = noop	文字列値	storage_interface フィールドが設定されていないノードに使用されるデフォルトのストレージインターフェイス。システムに存在するストレージインターフェイスの完全なリストは、ironic.hardware.interfaces.storage のエントリーポイントを列挙することで確認することができます。
default_vendor_interface = None	文字列値	vendor_interface フィールドが設定されていないノードに使用されるデフォルトのベンダーインターフェイス。システムに存在するベンダーインターフェイスの完全なリストは、ironic.hardware.interfaces.vendor のエントリーポイントを列挙することで確認することができます。
enabled_bios_interfaces = ['no-bios']	リスト値	サービスの初期化中にロードする bios インターフェイスのリストを指定します。bios インターフェイス、または初期化に失敗した bios インターフェイスがないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされる少なくとも 1 つの bios インターフェイスを有効にしないと、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨される bios インターフェイスのセットです。システムに存在する bios インターフェイスの完全なリストは、ironic.hardware.interfaces.bios のエントリーポイントを列挙することで確認することができます。この値を設定する場合は、すべての ironic-conductor サービスで、有効なすべてのハードウェア種別が同じ bios インターフェイスのセットを持つようにしてください。

設定オプション = デフォルト値	型	説明
enabled_boot_interfaces = ['pxe']	リスト値	サービスの初期化中にロードする boot インターフェイスのリストを指定します。boot インターフェイス、または初期化に失敗した boot インターフェイスがないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされる少なくとも 1 つの boot インターフェイスを有効にしないと、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨される boot インターフェイスのセットです。システムに存在するブートインターフェイスの完全なリストは、ironic.hardware.interfaces.boot のエントリーポイントを列挙することで確認することができます。この値を設定する場合は、すべての ironic-conductor サービスで、有効なすべてのハードウェア種別が同じ boot インターフェイスのセットを持つようにしてください。
enabled_console_interfaces = ['no-console']	リスト値	サービスの初期化中にロードする console インターフェイスのリストを指定します。console インターフェイス、または初期化に失敗した console インターフェイスがないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされる少なくとも 1 つの console インターフェイスを有効にしないと、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨される console インターフェイスのセットです。システムに存在するコンソールインターフェイスの完全なリストは、ironic.hardware.interfaces.console のエントリーポイントを列挙することで確認することができます。この値を設定する場合は、すべての ironic-conductor サービスで、有効なすべてのハードウェア種別が同じ console インターフェイスのセットを持つようにしてください。

設定オプション = デフォルト値	型	説明
enabled_deploy_interfaces = ['direct']	リスト値	サービスの初期化中にロードする deploy インターフェイスのリストを指定します。deploy インターフェイス、または初期化に失敗した deploy インターフェイスがないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされる少なくとも 1 つの deploy インターフェイスを有効にしないと、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨される deploy インターフェイスのセットです。システムに存在するデプロイインターフェイスの完全なリストは、ironic.hardware.interfaces.deploy エントリーポイントを列挙することで確認することができます。この値を設定する場合は、すべての ironic-conductor サービスで、有効なすべてのハードウェア種別が同じ deploy インターフェイスのセットを持つようにしてください。
enabled_hardware_types = ['ipmi']	リスト値	サービスの初期化中にロードするハードウェアタイプのリストを指定します。ハードウェアタイプ、または初期化に失敗したハードウェアタイプがないと、コンダクターサービスが起動できなくなります。このオプションはデフォルトで、推奨の実稼働指向のハードウェアタイプのセットに設定されます。システムに存在するハードウェアタイプの完全なリストは、ironic.hardware.types エントリーポイントを列挙することで確認することができます。
enabled_inspect_interfaces = ['no-inspect']	リスト値	サービスの初期化中にロードする inspect インターフェイスのリストを指定します。inspect インターフェイス、または初期化に失敗した inspect インターフェイスがないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされる少なくとも 1 つの inspect インターフェイスを有効にしないと、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨される inspect インターフェイスのセットです。システムに存在する検査インターフェイスの完全なリストは、ironic.hardware.interfaces.inspect のエントリーポイントを列挙することで確認することができます。この値を設定する場合は、すべての ironic-conductor サービスで、有効なすべてのハードウェア種別が同じ inspect インターフェイスのセットを持つようにしてください。

設定オプション = デフォルト値	型	説明
enabled_management_interfaces = ['ipmitool']	リスト値	サービスの初期化中にロードする management インターフェイスのリストを指定します。management インターフェイス、または初期化に失敗した management インターフェイスがないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされる少なくとも 1 つの management インターフェイスを有効にしないと、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨される management インターフェイスのセットです。システムに存在する管理インターフェイスの完全なリストは、ironic.hardware.interfaces.management のエントリーポイントを列挙することで確認することができます。この値を設定する場合は、すべての ironic-conductor サービスで、有効なすべてのハードウェア種別が同じ management インターフェイスのセットを持つようにしてください。
enabled_network_interfaces = ['flat', 'noop']	リスト値	サービスの初期化中にロードする network インターフェイスのリストを指定します。ネットワークインターフェイスがない、または初期化に失敗したネットワークインターフェイスが見つからないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされるネットワークインターフェイスが少なくとも 1 つ有効化されている必要があります。有効にすると、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨されるネットワークインターフェイスのセットです。システムに存在するネットワークインターフェイスの完全なリストは、ironic.hardware.interfaces.network エントリーポイントを列挙することで確認することができます。この値を設定する際には、有効なすべてのハードウェア種別が ironic-conductor サービスで同じ有効なネットワークインターフェイスセットを持つようにしてください。

設定オプション = デフォルト値	型	説明
enabled_power_interfaces = ['ipmitool']	リスト値	サービスの初期化中にロードする power インターフェイスのリストを指定します。power インターフェイス、または初期化に失敗した power インターフェイスがないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされる少なくとも 1 つの power インターフェイスを有効にしないと、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨される power インターフェイスのセットです。システムに存在する電源インターフェイスの完全なリストは、ironic.hardware.interfaces.power のエントリーポイントを列挙することで確認することができます。この値を設定する場合は、すべての ironic-conductor サービスで、有効なすべてのハードウェア種別が同じ power インターフェイスのセットを持つようにしてください。
enabled_raid_interfaces = ['agent', 'no-raid']	リスト値	サービスの初期化中にロードする raid インターフェイスのリストを指定します。raid インターフェイス、または初期化に失敗した raid インターフェイスがないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされる少なくとも 1 つの raid インターフェイスを有効にしないと、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨される raid インターフェイスのセットです。システムに存在する raid インターフェイスの完全なリストは、ironic.hardware.interfaces.raid のエントリーポイントを列挙することで確認することができます。この値を設定する場合は、すべての ironic-conductor サービスで、有効なすべてのハードウェア種別が同じ raid インターフェイスのセットを持つようにしてください。

設定オプション = デフォルト値	型	説明
enabled_rescue_interfaces = ['no-rescue']	リスト値	サービスの初期化中に読み込むレスキューインターフェイスのリストを指定します。rescue インターフェイス、または初期化に失敗した rescue インターフェイスがないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされる少なくとも 1 つの rescue インターフェイスを有効にしないと、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨される rescue インターフェイスのセットです。システムに存在するレスキューインターフェイスの完全なリストは、ironic.hardware.interfaces.rescue のエントリーポイントを列挙することで確認することができます。この値を設定する場合は、すべての ironic-conductor サービスで、有効なすべてのハードウェア種別が同じ rescue インターフェイスのセットを持つようにしてください。
enabled_storage_interfaces = ['cinder', 'noop']	リスト値	サービスの初期化中に読み込む storage インターフェイスのリストを指定します。storage インターフェイス、または初期化に失敗した storage インターフェイスがないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされる少なくとも 1 つの storage インターフェイスを有効にしないと、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨される storage インターフェイスのセットです。システムに存在するストレージインターフェイスの完全なリストは、ironic.hardware.interfaces.storage のエントリーポイントを列挙することで確認することができます。この値を設定する場合は、すべての ironic-conductor サービスで、有効なすべてのハードウェア種別が同じ storage インターフェイスのセットを持つようにしてください。

設定オプション = デフォルト値	型	説明
enabled_vendor_interfaces = ['ipmitool', 'no-vendor']	リスト値	サービスの初期化中にロードする vendor インターフェイスのリストを指定します。vendor インターフェイス、または初期化に失敗した vendor インターフェイスがないと、ironic-conductor サービスが起動しなくなります。有効な各ハードウェアタイプでサポートされる少なくとも 1 つの vendor インターフェイスを有効にしないと、ironic-conductor サービスが起動しません。リストは空にしないでください。デフォルト値は、実稼働環境で推奨される vendor インターフェイスのセットです。システムに存在するベンダーインターフェイスの完全なリストは、ironic.hardware.interfaces.vendor のエントリーポイントを列挙することで確認することができます。この値を設定する場合は、すべての ironic-conductor サービスで、有効なすべてのハードウェア種別が同じ vendor インターフェイスのセットを持つようにしてください。
esp_image = None	文字列値	EFI システムパーティションイメージファイルへのパス。このファイルは、UEFI ブート可能な ISO イメージを効率的に作成するのに推奨されます。ESP イメージには、EFI ブートローダーを保持する FAT12/16/32 形式のファイルシステム (例: GRUB2) が各ハードウェアアーキテクチャーの ironic を起動する必要があります。このオプションは、デプロイされるノードに ESP や ISO のデプロイイメージが設定されていない場合にのみ使用され、その場合 ironic は設定された場所から ESP イメージを取得するか、UEFI ブート可能なデプロイ ISO イメージから ESP イメージを抽出しようと試みます。
executor_thread_pool_size = 64	整数値	エグゼキューターがスレッディングまたはイベントレットの場合のエグゼキュータースレッドプールのサイズ。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
force_raw_images = True	ブール値	True の場合には、バッキングイメージを raw ディスクイメージ形式に変換します。
graceful_shutdown_timeout = 60	整数値	サーバーを正常にシャットダウンするまでのタイムアウトを指定します。値をゼロにすると、endless の待機を意味します。

設定オプション = デフォルト値	型	説明
grub_config_path = /boot/grub/grub.cfg	文字列値	ironic が生成する UEFI ISO イメージ上の GRUB2 設定ファイルの場所。通常、デフォルト値は正しくないため、信頼すべきではありません。特定のディストリビューションの GRUB2 イメージを使用する場合は、ここでディストリビューション固有のパスを使用します (例: EFI/ubuntu/grub.cfg)。
grub_config_template = $pybasedir/common/grub_conf.template	文字列値	grub 設定ファイルのテンプレートファイル。
hash_partition_exponent = 5	整数値	コンダクター全体で負荷を分散する際に使用するハッシュパーティションの数を決定します。値が大きいと、リングのリバランス時に負荷が分散され、負荷が少なくなりますが、メモリー使用量が増えます。コンダクターごとのパーティション数は (2^hash_partition_exponent) です。これにより、リバランスの粒度が決定します。10 台のホストがあり、2 つの要求が急増し、リングに 40 個のパーティションがあります。多くのパーティションでは、ほとんどの場合でリバランスがスムーズに行われるはずです。デフォルトは、最大 100 個のコンダクターに適しています。パーティションが多すぎるように設定すると、CPU 使用率に悪影響があります。
hash_ring_algorithm = md5	文字列値	ハッシュリングを構築するときに使用するハッシュ関数。FIPS システムで実行している場合は、md5 を使用しないでください。警告: クラスター内のすべての ironic サービスは、常に同じアルゴリズムを使用する必要があります。アルゴリズムを変更するには、オフライン更新が必要です。
hash_ring_reset_interval = 15	整数値	ハッシュリングが古くなり、次のアクセスで更新されてから、ハッシュリングが古くなった時間 (秒単位)。
host = <based on operating system>	文字列値	このノードの名前。これは不透明な ID にすることができます。必ずしもホスト名、FQDN、または IP アドレスではありません。ただし、ノード名は AMQP キー内で有効で、ZeroMQ を使用している場合は (Stein リリースで削除される)、有効なホスト名、FQDN、または IP アドレス。
http_basic_auth_user_file = /etc/ironic/htpasswd	文字列値	auth_strategy=http_basic の場合に使用される Apache 形式のユーザー認証ファイルへのパス
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。

設定オプション = デフォルト値	型	説明
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。
isolinux_bin = /usr/lib/syslinux/isolinux.bin	文字列値	isolinux バイナリーファイルへのパス。
isolinux_config_template = $pybasedir/common/isolinux_config.template	文字列値	isolinux 設定ファイルのテンプレートファイル。
ldlinux_c32 = None	文字列値	ldlinux.c32 ファイルへのパス。このファイルは、syslinux 5.0 以降に必要です。指定されていない場合、ファイルは/usr/lib/syslinux/modules/bios/ldlinux.c32 と/usr/share/syslinux/ldlinux.c32 で検索されます。
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_in_db_max_size = 4096	整数値	last_error/maintenance_reason がデータベースにプッシュされたノードの最大文字数。

設定オプション = デフォルト値	型	説明
log_options = True	ブール値	サービスの起動時 (DEBUG レベル) に、登録されているすべてのオプションの値をログに記録することを有効または無効にします。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。

設定オプション = デフォルト値	型	説明
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
minimum_memory_wait_retries = 6	整数値	コンダクターが自動的に再試行できる場合、失敗するか、スレッドをプールに戻す前に、ワーカーに保持する再試行の回数。
minimum_memory_wait_time = 15	整数値	プロセスを起動する前に、メモリーの空きを再試行するまで待機する秒数。これを **memory_wait_retries** と組み合わせると、コンダクターは、直接再試行を試行する時間を決定できます。
minimum_memory_warning_only = False	ブール値	システムメモリーの枯渇を防ぐために、ironic が要求を保留しようとするのではなく、警告のみを行うかどうかを制御するための設定。
minimum_required_memory = 1024	整数値	コンダクターでメモリー集約型プロセスを開始する前に、システムが使用できる MiB 単位の最小メモリー。
my_ip = <オペレーティングシステムに基づく>	文字列値	このホストの IPv4 アドレス。設定しなければ、IP をプログラム的に判断します。これを行わない場合、は 127.0.0.1 を使用します。注記: このフィールドは、テンプレートと URL のオーバーライドとして IPv6 アドレスを受け入れますが、DEFAULTmy_ipv6 をデュアルスタック環境のサービス URL の DNS 名と共に使用することを推奨します。
my_ipv6 = None	文字列値	IPv6 を使用したこのホストの IP アドレス。この値は設定を介して指定する必要があり、IPv4 の DEFAULTmy_ip パラメーターのようにプログラムで適切に決定することはできません。
notification_level = None	文字列値	通知を送信する最小レベルを指定します。設定されていない場合は、通知は送信されません。デフォルトでは、このオプションは未設定になっています。
parallel_image_downloads = False	ブール値	イメージのダウンロードと raw 形式の変換を並行して実行します。
pecan_debug = False	ブール値	pecan デバッグモードを有効にします。警告: これは安全ではないため、実稼働環境では使用しないでください。

設定オプション = デフォルト値	型	説明
pin_release_version = None	文字列値	ローリングアップグレードに使用されます。このオプションを設定すると、Bare Metal API、内部の ironic RPC 通信、およびデータベースオブジェクトをそれぞれのバージョンにダウングレード (またはピニング) し、古いサービスと互換性を持たせることができます。バージョン N から N+1 へのローリングアップグレードを行う場合は、これを N に設定します。アンパン (デフォルト) には未設定のままにして、最新バージョンが使用されます。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
pybasedir = /usr/lib/python3.9/site-packages/ironic	文字列値	ironic python モジュールがインストールされているディレクトリー。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
raw_image_growth_factor = 2.0	浮動小数点の値	QCOW2 などのコンパクトなイメージフォーマットから変換された生のイメージのサイズを見積もるために使用される倍率。デフォルトは 2.0 で、1.0 より大きくする必要があります。
rootwrap_config = /etc/ironic/rootwrap.conf	文字列値	root としてコマンドを実行するのに使用する rootwrap 設定ファイルへのパス。
rpc_conn_pool_size = 30	整数値	RPC 接続プールのサイズ。
rpc_ping_enabled = False	ブール値	ping 呼び出しに応答するエンドポイントを追加します。エンドポイントの名前は oslo_rpc_server_ping です。
rpc_response_timeout = 60	整数値	呼び出しからの応答を待つ秒数。

設定オプション = デフォルト値	型	説明
rpc_transport = oslo	文字列値	コンダクターと API サービスの間に使用する RPC トランスポート実装
run_external_periodic_tasks = True	ブール値	一部の定期的なタスクは個別のプロセスで実行できます。ここで実行すべきですか？
state_path = $pybasedir	文字列値	cinder の状態を維持する最上位のディレクトリー
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
tempdir = /tmp	文字列値	一時作業ディレクトリー。デフォルトは Python temp dir です。
transport_url = rabbit://	文字列値	メッセージングバックエンドに接続するためのネットワークアドレスおよびオプションのユーザー認証情報 (URL 形式)。想定される形式は次のとおりです。 driver://[user:pass@]host:port[, [userN:passN@]hostN:portN]/virtual_host?query 例:rabbit://rabbitmq:password@127.0.0.1:5672// URL のフィールドの詳細は、https://docs.openstack.org/oslo.messaging/latest/reference/transport.html で oslo_messaging.TransportURL のドキュメントを参照してください。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。

設定オプション = デフォルト値	型	説明
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
versioned_notifications_topics = ['ironic_versioned_notifications']	リスト値	Ironic が発行するバージョン付き通知のトピックを指定します。 デフォルト値はほとんどのデプロイメントでは問題ありませんが、変更が必要になることはほとんどありません。ただし、バージョン付けされた通知を使用するサードパーティーのサービスがある場合は、そのサービスのトピックを取得することが望ましい場合があります。Ironic は、バージョン別の通知ペイロードを含むメッセージを、このリストの各トピックキューに送信します。 バージョン付けされた通知のリストは、https://docs.openstack.org/ironic/latest/admin/notifications.html に表示されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。
webserver_connection_timeout = 60	整数値	イメージを含むリモート Web サーバーにアクセスする際の接続タイムアウト。
webserver_verify_ca = True	文字列値	証明書の検証に使用される CA 証明書。これは、ブール値または CA_BUNDLE ファイルへのパスのいずれかです。True に設定されている場合、標準パスに存在する証明書を使用して、ホスト証明書が検証されます。False に設定されている場合、コンダクタは、によって提示された SSL 証明書の検証を無視します。ホスト。パスの場合、コンダクターは指定された証明書を SSL 検証に使用します。パスが存在しない場合、動作はこの値が True に設定されている場合と同じです。つまり、標準パスに存在する証明書が SSL 検証に使用されます。デフォルトは True です。

5.1.2. agent

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[agent]** グループで使用できるオプションの概要をまとめたものです。

表5.1 agent

設定オプション = デフォルト値	型	説明
agent_api_version = v1	文字列値	ramdisk エージェントとの通信に使用する API バージョン。
api_ca_file = None	文字列値	ベアメタル API を開始するために使用される TLS CA へのパス。一部の起動方法では、このファイルを ramdisk に渡すことができます。
certificates_path = /var/lib/ironic/certificates	文字列値	RAM ディスクへの接続を検証するために使用される、自動生成された TLS 証明書を格納するパス。
command_timeout = 60	整数値	IPA コマンドのタイムアウト (秒単位)。
command_wait_attempts = 100	整数値	タイムアウトする前に非同期コマンドの完了をチェックする試行回数。
command_wait_interval = 6	整数値	非同期コマンドの完了をチェックするまで待機する秒数。
deploy_logs_collect = on_failure	文字列値	Ironic がデプロイメントの失敗時にデプロイメントログを収集するかどうか (on_failure)、always または never。
deploy_logs_local_path = /var/log/ironic/deploy	文字列値	deploy_logs_storage_backend が "local" に設定されている場合に使用されるログが保存されるディレクトリーへのパス。
deploy_logs_storage_backend = local	文字列値	ログが保存されるストレージバックエンドの名前。
deploy_logs_swift_container = ironic_deploy_logs_container	文字列値	ログを保存する Swift コンテナーの名前。deploy_logs_storage_backend が "swift" に設定されている場合に使用されます。
deploy_logs_swift_days_to_expire = 30	整数値	ログオブジェクトが Swift で期限切れとマークされるまでの日数。None の場合、ログは永久に、または手動で削除されるまで保持されます。deploy_logs_storage_backend が "swift" に設定されている場合に使用されます。

設定オプション = デフォルト値	型	説明
image_download_source = http	文字列値	直接デプロイインターフェイスがイメージソースを直接使用しようとするか、ironic がコンダクター上でイメージをキャッシュして ironic の独自の http サーバーからそれを提供すべきかどうかを指定します。
manage_agent_boot = True	ブール値	Ironic がエージェント ramdisk のブートを管理するかどうか。False に設定した場合は、エージェント ramdisk を起動できるようにメカニズムを設定する必要があります。
max_command_attempts = 3	整数値	これは、ネットワークの問題が原因で失敗した IPA コマンドに対して行われる最大試行回数です。
memory_consumed_by_agent = 0	整数値	ベアメタルノードでブートする場合のエージェントによって使用されるメモリー容量 (MiB 単位)。これは、ブートエージェント ramdisk の後に、イメージをダウンロードしてベアメタルノードにデプロイすることができるかどうかを確認します。これは、エージェント ramdisk イメージが消費するメモリーに応じて設定できます。
neutron_agent_max_attempts = 100	整数値	デッドエージェントのネットワークエラーを発生させる前に、Neutron エージェントのステータスを検証する最大試行回数。
neutron_agent_poll_interval = 2	整数値	Neutron エージェントがデバイスの変更をポーリングするまで待つ秒数。この値は、Neutron 設定の CONF.AGENT.polling_interval と同じである必要があります。
neutron_agent_status_retry_interval = 10	整数値	Neutron エージェントのステータスの検証を試みる間隔 (秒単位)。
post_deploy_get_power_state_retries = 6	整数値	ソフトパワーオフ後にベアメタルノードの電源がオフになっているかどうかを確認するために電源状態の取得を再試行する回数。
post_deploy_get_power_state_retry_interval = 5	整数値	ソフト電源オフをトリガーした後に、電源状態をポーリングするまでの間待機する時間 (秒単位)。
require_tls = False	ブール値	True に設定すると、https:// のないコールバック URL はコンダクターによって拒否されます。

設定オプション = デフォルト値	型	説明
stream_raw_images = True	ブール値	エージェント ramdisk が raw イメージを直接ディスクにストリーミングするかどうか。raw イメージをディスクに直接ストリーミングすると、ディスクに書き込む前に、イメージが tmpfs パーティションにコピーされる (メモリーを少なくなる) に費やされません。イメージがコピーされるディスクが非常に遅い場合を除き、このオプションを True に設定する必要があります。デフォルトは True です。
verify_ca = True	文字列値	RAM ディスクへの接続を検証するための TLS CA へのパス。システムのデフォルト CA ストレージを使用するには、True に設定します。検証を無効にするには、False に設定します。自動 TLS セットアップが使用されている場合は無視されます。

5.1.3. anaconda

次の表は、**/etc/ironic/ironic.conf** ファイルの **[anaconda]** グループで使用できるオプションの概要を示しています。

表5.2 anaconda

設定オプション = デフォルト値	型	説明
default_ks_template = $pybasedir/drivers/modules/ks.cfg.template	文字列値	instance_info または glance OS イメージでキックスタートテンプレートが指定されていない場合に使用するキックスタートテンプレート。

5.1.4. ansible

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[ansible]** グループで使用できるオプションの概要をまとめたものです。

表5.3 ansible

設定オプション = デフォルト値	型	説明
ansible_extra_args = None	文字列値	Ansible のすべての呼び出しに渡す追加の引数。
ansible_playbook_script = ansible-playbook	文字列値	"ansible-playbook" スクリプトへのパスデフォルトでは、ironic-conductor プロセスを実行するユーザー用に設定された $PATH を検索します。ansible-playbook が $PATH にない場合や、デフォルトの場所でインストールされていない場合は完全パスを指定します。

設定オプション = デフォルト値	型	説明
config_file_path = $pybasedir/drivers/modules/ansible/playbooks/ansible.cfg	文字列値	Ansible 設定ファイルへのパス。空の場合、システムのデフォルトが使用されます。
default_clean_playbook = clean.yaml	文字列値	ノードのクリーニングに使用するデフォルトの Playbook へのパス ($playbooks_path または absolute との相対)。これは、ノードの **driver_info** フィールドのノードごとの **ansible_clean_playbook** オプションで上書きできます。
default_clean_steps_config = clean_steps.yaml	文字列値	ノードのクリーニング時に使用されるデフォルトの補助的なクリーニング手順ファイルへのパス ($playbooks_path または absolute との相対)。これは、ノードの **driver_info** フィールドのノードごとの **ansible_clean_steps_config** オプションで上書きできます。
default_deploy_playbook = deploy.yaml	文字列値	デプロイメントに使用するデフォルトの Playbook へのパス ($playbooks_path または絶対)。これは、ノードの **driver_info** フィールドのノードごとの **ansible_deploy_playbook** オプションで上書きできます。
default_key_file = None	文字列値	SSH 経由で ramdisk に接続する際に Ansible が使用する SSH 秘密鍵ファイルへの絶対パス。デフォルトでは、ironic-conductor サービスを実行するユーザーに設定されたデフォルトの SSH キーを使用します。パスワードのある秘密鍵は、**ssh-agent** に事前にロードする必要があります。これは、ノードの **driver_info** フィールドのノードごとの **ansible_key_file** オプションで上書きできます。
default_python_interpreter = None	文字列値	管理マシンの python インタープリターへの絶対パス。これは、ノードの **driver_info** フィールドで、ノードごとの **ansible_python_interpreter** オプションで上書きできます。デフォルトでは、ansible は /usr/bin/python を使用します。
default_shutdown_playbook = shutdown.yaml	文字列値	ノードの正常なシャットダウンに使用されるデフォルトの Playbook へのパス ($playbooks_path または absolute)。これは、ノードの **driver_info** フィールドのノードごとの **ansible_shutdown_playbook** オプションで上書きできます。
default_username = ansible	文字列値	SSH 経由で ramdisk に接続する際に Ansible に使用するユーザーの名前。これは、ノードの **driver_info** フィールドのノードごとの **ansible_username** オプションで上書きできます。

設定オプション = デフォルト値	型	説明
extra_memory = 10	整数値	ノード上の Ansible 関連のプロセスで、メモリーが MiB 単位で消費されることが想定されています。イメージが RAM に適合するかどうかの決定に影響します。
image_store_cafile = None	文字列値	イメージストアへの SSL 接続の検証に使用する特定の CA バンドル。指定しない場合は、ramdisk で利用可能な CA が使用されます。は、このドライバーに含まれるデフォルトの Playbook では使用されません。自己署名証明書を使用する環境に適しています。
image_store_certfile = None	文字列値	イメージストアへの SSL 接続に使用するクライアント証明書。は、このドライバーに含まれるデフォルトの Playbook では使用されません。
image_store_insecure = False	ブール値	イメージのダウンロード時にイメージストアへの SSL 接続の検証を省略します。自己署名証明書を使用する環境では、これを True に設定する場合にのみ推奨されます。
image_store_keyfile = None	文字列値	イメージストアへの SSL 接続に使用するクライアントキー。は、このドライバーに含まれるデフォルトの Playbook では使用されません。
playbooks_path = $pybasedir/drivers/modules/ansible/playbooks	文字列値	Playbook、ロール、およびローカルインベントリーを含むディレクトリーへのパス。
post_deploy_get_power_state_retries = 6	整数値	ソフトパワーオフ後にベアメタルノードの電源がオフになっているかどうかを確認するために電源状態の取得を再試行する回数。値が 0 の場合は、失敗時に再試行されないことを意味します。
post_deploy_get_power_state_retry_interval = 5	整数値	ソフト電源オフをトリガーした後に、電源状態をポーリングするまでの間待機する時間 (秒単位)。
verbosity = None	整数値	ansible-playbook コマンドを呼び出す際に要求される Ansible の詳細レベルを設定します。4 には、詳細な SSH セッションロギングが含まれます。グローバルデバッグが有効な場合は、デフォルトは 4 で、そうでない場合は 0 になります。

5.1.5. api

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[api]** グループで使用できるオプションの概要をまとめたものです。

表5.4 api

設定オプション = デフォルト値	型	説明
api_workers = None	整数値	OpenStack Ironic API サービスのワーカー数。デフォルトは、使用可能な CPU の数と同じですが、4 を超えてはなりません。CPU 番号が検出できない場合は、1 つのワーカーが使用されます。
enable_ssl_api = False	ブール値	HTTP ではなく HTTPS 経由で要求を処理するための統合されたスタンドアロン API を有効にします。サービスから HTTPS オフロードを実行するフロントエンドサービスがある場合は、このオプションを False にする必要があります。ただし、プロキシーヘッダーの解析を [oslo_middleware]enable_proxy_headers_parsing オプションで有効にするか、[api]public_endpoint オプションを設定して、SSL で終端した URL に設定します。
host_ip = 0.0.0.0	ホストのアドレス値	ironic-api がリッスンする IP アドレスまたはホスト名。
max_limit = 1000	整数値	コレクションリソースからの単一の応答で返されるアイテムの最大数。
network_data_schema = $pybasedir/api/controllers/v1/network-data-schema.json	文字列値	このデプロイメントで使用されるネットワークデータのスキーマ。
port = 6385	ポート値	ironic-api がリッスンする TCP ポート。
public_endpoint = None	文字列値	API リソースへのリンクを構築するときに使用するパブリック URL (例: "https://ironic.rocks:6384")。None の場合、リンクは要求のホスト URL を使用してビルドされます。API がプロキシーの背後で動作している場合、これをプロキシーの URL を表すように変更する必要があります。デフォルトは None です。プロキシーヘッダー解析が [oslo_middleware]enable_proxy_headers_parsing オプションで有効になっている場合は無視されます。
ramdisk_heartbeat_timeout = 300	整数値	エージェントハートビートの最大間隔 (秒単位)。
restrict_lookup = True	ブール値	lookup API を特定の状態のノードのみに制限するかどうか。

5.1.6. audit

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[audit]** グループで使用できるオプションの概要をまとめたものです。

表5.5 audit

設定オプション = デフォルト値	型	説明
audit_map_file = /etc/ironic/api_audit_map.conf	文字列値	ironic-api サービスの監査マップファイルへのパス。API 監査が有効になっている場合にのみ使用されます。
enabled = False	ブール値	API 要求の監査を有効にします (ironic-api サービス用)。
`ignore_req_list = `	文字列値	監査ロギング中に無視される Ironic REST API HTTP メソッドのコンマ区切りリスト。たとえば、監査は、GET,POST に設定されている場合には、GET または POST 要求では実行されません。これは、API 監査が有効になっている場合にのみ使用されます。

5.1.7. cinder

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[cinder]** グループで使用できるオプションの概要をまとめたものです。

表5.6 cinder

設定オプション = デフォルト値	型	説明
action_retries = 3	整数値	アクションの失敗時の再試行回数 (現在、ボリュームの割り当て解除時にのみ使用されます)。
action_retry_interval = 5	整数値	アクションの失敗時に再試行の間隔 (秒単位) です (特定のアクションのみが再試行されます)。
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。

設定オプション = デフォルト値	型	説明
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
max-version = None	文字列値	特定の API の最大メジャーバージョンで、min_version を持つ範囲の上限として使用されることが意図されています。version と相互に排他的です。
min-version = None	文字列値	指定の API の最小メジャーバージョンで、max_version を持つ範囲の下限として使用することが意図されています。version と相互に排他的です。max_version なしで min_version が指定されている場合は、max version が latest の場合になります。
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名

設定オプション = デフォルト値	型	説明
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
retries = 3	整数値	要求接続に失敗した場合にクライアントが再試行します。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = volumev3	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名

設定オプション = デフォルト値	型	説明
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。
version = None	文字列値	エンドポイント URL 検出用に指定の Major API バージョン内の最小の Major API バージョン。min_version および max_version と相互排他的になる

5.1.8. conductor

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[conductor]** グループで使用できるオプションの概要をまとめたものです。

表5.7 conductor

設定オプション = デフォルト値	型	説明
allow_deleting_available_nodes = True	ブール値	**利用可能**状態になっているノードの削除を許可します。デフォルトは True です。
allow_provisioning_in_maintenance = True	ブール値	メンテナンスモードの場合、ノードがデプロイまたはクリーニングに入ることを許可するかどうか。このオプションが False に設定され、ノードがデプロイまたはクリーニング中にメンテナンスに入ると、このプロセスは次のハートビートの後に中止されます。また、自動消去やノードの利用可能にも失敗します。True (デフォルト) の場合、プロセスは開始し、ノードがハートビートを開始した後に一時停止します。メンテナンスから移行すると、プロセスは継続されます。
automated_clean = True	ブール値	自動消去を有効または無効にします。自動クリーニングは、ディスクドライブの消去など、設定可能な手順のセットで、ベースラインの状態にあり、デプロイの準備が整うノードで実行されます。これは、インスタンスの削除後や管理可能から available 状態への移行中に行われます。有効にすると、ノードをクリーンアップするために実行する特定の手順は、ノードが管理するドライバーによって異なります。詳細は、個々のドライバーのドキュメントを参照してください。注: クリーニング操作の導入により、インスタンスの削除にかかる時間が大幅に長くなります。(テナントが 1 つしかないため) すべてのテナントが信頼されている環境では、このオプションを安全に無効にすることができます。

設定オプション = デフォルト値	型	説明
bootloader = None	文字列値	EFI ブートローダーを含む EFI システムパーティションイメージの Glance ID、http:// または file:// URL。このイメージは、カーネルおよび ramdisk から UEFI ブート可能な ISO を構築する際に ironic により使用されます。パーティションイメージからの UEFI ブートに必要です。
check_allocations_interval = 60	整数値	孤立した割り当てのチェックの間隔 (秒単位)。チェックを無効にするには 0 に設定します。
check_provision_state_interval = 60	整数値	プロビジョニングタイムアウトのチェックの間隔 (秒単位)。チェックを無効にするには 0 に設定します。
check_rescue_state_interval = 60	整数値	レスキュータイムアウトのチェックの間隔 (秒単位)。
clean_callback_timeout = 1800	整数値	クリーニングを行う ramdisk からのコールバックを待つタイムアウト (秒)。タイムアウトに達すると、ノードは clean failed のプロビジョニング状態になります。タイムアウトを無効にするには 0 に設定します。
clean_step_priority_override = {}	dict 値	インバンドとアウトオブバンドの両方のクリーンステップに対して自動クリーンステップを実行する優先度 (例: deploy.erase_devices_metadata:123)。このオプションを複数回指定して、複数のステップの優先度を定義できます。0 に設定すると、この特定のステップはクリーニング中に実行されません。インバンドクリーンステップ用に設定されていない場合は、RAM ディスクに設定されている優先順位が使用されます。
`conductor_group = `	文字列値	参加するコンダクターグループの名前。最大 255 文字を使用でき、大文字と小文字は区別されません。このコンダクターは、ノードで一致する "conductor_group" フィールドが設定されたノードのみを管理します。
configdrive_swift_container = ironic_configdrive_container	文字列値	コンフィグドライブデータを格納する Swift コンテナーの名前。configdrive_use_object_store が True の場合に使用されます。

設定オプション = デフォルト値	型	説明
configdrive_swift_temp_url_duration = None	整数値	configdrive の一時 URL が無効になるタイムアウト (秒単位)。deploy_callback_timeout が設定されている場合にはデフォルトで deploy_callback_timeout に設定されます。それ以外の場合は 1800 秒に設定されます。configdrive_use_object_store が True の場合に使用されます。
deploy_callback_timeout = 1800	整数値	デプロイ ramdisk からのコールバックを待つタイムアウト (秒)。タイムアウトを無効にするには 0 に設定します。
deploy_kernel = None	文字列値	デフォルトデプロイイメージのカーネルの Glance ID、http:// または file:// URL。
deploy_ramdisk = None	文字列値	デフォルトデプロイイメージの initramfs の Glance ID、http:// または file:// URL。
enable_mdns = False	ブール値	マルチキャスト DNS 経由でベアメタル API エンドポイントの公開を有効にするかどうか。
force_power_state_during_sync = True	ブール値	sync_power_state の実行中に、ハードウェアの電源状態がデータベース (True) で記録された状態に設定されているか、ハードウェアの状態 (False) に基づいてデータベースを更新するようにしてください。
heartbeat_interval = 10	整数値	コンダクターの中心者間の秒数。
heartbeat_timeout = 60	整数値	コンダクターの最後のチェックインからの最大時間 (秒単位)。この時間が経過すると、コンダクターは非アクティブとみなされます。
inspect_wait_timeout = 1800	整数値	ノードの検査を待機するタイムアウト (秒)。0 - 無制限。
node_locked_retry_attempts = 3	整数値	ノードのロックの取得を試行する回数。
node_locked_retry_interval = 1	整数値	ノードのロック試行間でスリープ状態になる秒数。
periodic_max_workers = 8	整数値	定期的なタスクで同時に起動できるワーカースレッドの最大数。RPC スレッドプールのサイズよりも小さくなければなりません。

設定オプション = デフォルト値	型	説明
power_failure_recovery_interval = 300	整数値	電源同期に失敗し、以前にメンテナンスモードになったノードの電源状態を確認する間隔 (秒)。ノードは、電源状態が正常に取得されると、自動的にメンテナンスモードから移動されます。このチェックを無効にするには 0 に設定します。
power_state_change_timeout = 60	整数値	電源操作が完了するのを待機する秒数。つまり、ベアメタルノードが目的の電源状態になるまで待機します。タイムアウトとなると、電源操作は障害と見なされます。
power_state_sync_max_retries = 3	整数値	sync_power_state の失敗時に、Ironic がハードウェアノードの電源状態を DB の電源状態と同期させる回数を制限します。
require_rescue_password_hashed = False	ブール値	コンダクターがレスキューパスワードのハッシュされていないバージョンにフォールバックしないようにするオプション。古い ironic-python-agent ramdisk でのレスキューを許可します。
rescue_callback_timeout = 1800	整数値	レスキュー ramdisk からのコールバックを待機するタイムアウト (秒)。タイムアウトに達すると、ノードは rescue failed のプロビジョニング状態になります。タイムアウトを無効にするには 0 に設定します。
rescue_kernel = None	文字列値	デフォルトのレスキューイメージのカーネルの Glance ID、http:// または file:// URL。
rescue_password_hash_algorithm = sha256	文字列値	レスキューパスワードに使用するパスワードハッシュアルゴリズム。
rescue_ramdisk = None	文字列値	デフォルトのレスキューイメージの initramfs の Glance ID、http:// または file:// URL。
send_sensor_data = False	ブール値	通知バスを介したセンサーデータメッセージの送信を有効にします。
send_sensor_data_for_undeployed_nodes = False	ブール値	センサーデータ収集のデフォルトは、デプロイされたマシンのデータのみを収集することですが、オペレーターは、現在使用されていないハードウェアに障害があるかどうかを確認することができません。true に設定すると、send_sensor_data 設定でセンサーデータ収集が有効な場合に、コンダクターはすべてのノードからセンサー情報を収集します。

設定オプション = デフォルト値	型	説明
send_sensor_data_interval = 600	整数値	通知バスを介して ceilometer にセンサーデータメッセージを送信するコンダクター間の秒数。
send_sensor_data_types = ['ALL']	リスト値	Ceilometer に送信する必要があるコンマ区切りのメータータイプのリスト。デフォルト値の ALL は特殊な値で、すべてのセンサーデータを送信します。
send_sensor_data_wait_timeout = 300	整数値	定期的な呼び出しが再度実行されるようにする前に、センサーデータ定期タスクが終了するまで待機する時間 (秒単位)。send_sensor_data_interval の値よりも小さくなければなりません。
send_sensor_data_workers = 4	整数値	センサーの定期的なタスクからデータを送信するために同時に開始できるワーカーの最大数。
soft_power_off_timeout = 600	整数値	ソフトリブートおよびソフト電源オフ操作のタイムアウト (秒単位)。この値は常に正である必要があります。
sync_local_state_interval = 180	整数値	コンダクターがクラスターに参加または離脱する場合、既存のコンダクターは、ノードがクラスター内に移動する際に永続的なローカル状態を更新する必要がある場合があります。このオプションは、各コンダクターが引き継ぎする必要のあるノードをチェックする頻度を秒単位で制御します。0 (または負の値) に設定してチェックを完全に無効にします。
sync_power_state_interval = 60	整数値	ノードの電源状態をデータベースと同期する間隔 (秒単位)。同期を無効にするには 0 に設定します。
sync_power_state_workers = 8	整数値	ノードの電源状態を周期タスクから同期するために同時に起動できるワーカースレッドの最大数。
workers_pool_size = 100	整数値	ワーカーの greenthread プールのサイズ。2 つのスレッドがコンダクター自体によって予約され、中心となるタスクや定期的なタスクを処理することに注意してください。その上で、**sync_power_state_workers** がデフォルト値の 8 のグリーンスレッドを最大 7 消費します。

5.1.9. console

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[console]** グループで使用できるオプションの概要をまとめたものです。

表5.8 console

設定オプション = デフォルト値	型	説明
kill_timeout = 1	整数値	SIGKILL シグナルを送信する前に、コンソールサブプロセスが終了するまで待機する時間 (秒単位)。
port_range = None	文字列値	ironic コンダクターのホストで実行されているコンソールプロキシーサービスに使用できるポートの範囲 (<start>:<stop> の形式)。このオプションは、Shellinabox と Socat コンソールの両方で使用されます
socat_address = $my_ip	IP アドレス値	ironic コンダクターのホストで実行している Socat サービスの IP アドレス。Socat コンソールによってのみ使用されます。
subprocess_checking_interval = 1	整数値	コンソールサブプロセスのステータスを確認する時間間隔 (秒単位)。
subprocess_timeout = 10	整数値	コンソールのサブプロセスが起動するまで待機する時間 (秒単位)。
terminal = shellinaboxd	文字列値	シリアルコンソールターミナルプログラムへのパス。Shell In A Box コンソールによってのみ使用されます。
terminal_cert_dir = None	文字列値	シリアルコンソールアクセス用の端末 SSL 証明書 (PEM) を含むディレクトリー。Shell In A Box コンソールによってのみ使用されます。
terminal_pid_dir = None	文字列値	端末の pid ファイルを保持するためのディレクトリー。指定のない場合は、一時ディレクトリーが使用されます。
terminal_timeout = 600	整数値	非アクティブ時にターミナルセッションが閉じられるタイムアウト (秒単位)。タイムアウトを無効にするには 0 に設定します。Socat コンソールによってのみ使用されます。

5.1.10. cors

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[cors]** グループで使用できるオプションの概要をまとめたものです。

表5.9 cors

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
allow_credentials = True	ブール値	実際の要求にユーザーの認証情報を含めることができることを示します。
allow_headers = []	リスト値	実際の要求時に使用されるヘッダーフィールド名を示します。
allow_methods = ['OPTIONS', 'GET', 'HEAD', 'POST', 'PUT', 'DELETE', 'TRACE', 'PATCH']	リスト値	実際の要求時に使用できるメソッドを示します。
allowed_origin = None	リスト値	このリソースがリクエストの origin"ヘッダーで受信したドメインと共有されるかどうかを示します。形式:"<protocol>://<host>[:<port>]" (行末のスラッシュなし)例:https://horizon.example.com
expose_headers = []	リスト値	API に安全に公開できるヘッダーを示します。デフォルトは HTTP Simple ヘッダーです。
max_age = 3600	整数値	CORS プリフライトリクエストの最大キャッシュ期間。

5.1.11. database

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[database]** グループで使用できるオプションの概要をまとめたものです。

表5.10 database

設定オプション = デフォルト値	型	説明
backend = sqlalchemy	文字列値	データベースに使用するバックエンド。
connection = None	文字列値	データベースへの接続に使用する SQLAlchemy 接続文字列。
connection_debug = 0	整数値	SQL デバッグ情報の冗長性: 0=なし、100=すべて。
`connection_parameters = `	文字列値	接続時に接続 URL に追加するオプションの URL パラメーター。param1=value1&param2=value2&... として指定します。
connection_recycle_time = 3600	整数値	この秒数より長く接続プールに存在していた接続は、次にプールからチェックアウトされたときに新しい接続に置き換えられます。

設定オプション = デフォルト値	型	説明
connection_trace = False	ブール値	Python スタックトレースをコメント文字列として SQL に追加します。
db_inc_retry_interval = True	ブール値	True の場合には、db_max_retry_interval までのデータベース操作の再試行回数を長くします。
db_max_retries = 20	整数値	エラーが発生する前に接続エラーまたはデッドロックが発生した場合の最大再試行。再試行回数を無限に指定するには -1 に設定します。
db_max_retry_interval = 10	整数値	db_inc_retry_interval が設定されている場合は、データベース操作を再試行するまでの最大秒数です。
db_retry_interval = 1	整数値	データベーストランザクションの再試行間の秒数。
max_overflow = 50	整数値	設定されている場合、この値を SQLAlchemy で max_overflow に使用します。
max_pool_size = 5	整数値	プールに開いたままにする SQL 接続の最大数。値を 0 に設定すると無制限を意味します。
max_retries = 10	整数値	起動時のデータベース接続の最大再試行数。再試行回数を無限に指定するには -1 に設定します。
mysql_enable_ndb = False	ブール値	True の場合には、MySQL Cluster (NDB) の処理のサポートを透過的に有効にします。
mysql_engine = InnoDB	文字列値	使用する MySQL エンジン。
mysql_sql_mode = TRADITIONAL	文字列値	MySQL セッションに使用する SQL モード。このオプション (デフォルトを含む) は、サーバーセット SQL モードを上書きします。サーバー設定で設定された SQL モードを使用するには、これを no value に設定します。例: mysql_sql_mode=
pool_timeout = None	整数値	設定されている場合は、この値を SQLAlchemy で pool_timeout に使用します。
retry_interval = 10	整数値	SQL 接続を開く再試行の間隔。
slave_connection = None	文字列値	スレーブデータベースへの接続に使用する SQLAlchemy 接続文字列。
sqlite_synchronous = True	ブール値	True の場合、SQLite は同期モードを使用します。

設定オプション = デフォルト値	型	説明
use_db_reconnect = False	ブール値	失われた接続でデータベースの実験的な使用を有効にします。

5.1.12. deploy

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[deploy]** グループで使用できるオプションの概要をまとめたものです。

表5.11 deploy

設定オプション = デフォルト値	型	説明
configdrive_use_object_store = False	ブール値	コンフィグドライブをオブジェクトストアにアップロードするかどうか。swift エンドポイントでコンフィグドライブを保存するには、このオプションを True に設定します。
continue_if_disk_secure_erase_fails = False	ブール値	Ironic Python Agent でのクリーニング中に安全な消去操作 (NVMe または ATA) が失敗した場合の対処方法を定義します。False の場合、クリーニング操作が失敗し、ノードは **clean failed** 状態になります。True の場合には、shred が呼び出され、クリーニングが継続されます。
create_configuration_priority = None	整数値	Ironic Python Agent ramdisk を介して、デバイスから RAID 設定を作成するインバンドクリーンステップを実行する優先度。設定されていない場合は、RAM ディスクに設定されている優先順位が使用されます (GenericHardwareManager のデフォルトは 0 です)。0 に設定すると、クリーニング中は実行されません。
default_boot_mode = bios	文字列値	ノードの driver_info、ケイパビリティー、または **instance_info** 設定でブートモードが要求されない場合に使用するデフォルトのブートモード。現在、デフォルトのブートモードは bios ですが、今後は uefi に変更されます。このオプションに明示的な値を設定することが推奨されます。このオプションは、管理インターフェイスがブートモード管理に対応している場合に限り有効です。
default_boot_option = local	文字列値	ノードの driver_info で起動オプションが要求されない場合に使用するデフォルトの起動オプション。デフォルトはローカルです。Ussuri リリースより前は、デフォルトは "netboot" でした。

設定オプション = デフォルト値	型	説明
delete_configuration_priority = None	整数値	Ironic Python Agent ramdisk を介して、デバイスから RAID 設定を消去するインバンドクリーンステップを実行する優先度。設定されていない場合は、RAM ディスクに設定されている優先順位が使用されます (GenericHardwareManager のデフォルトは 0 です)。0 に設定すると、クリーニング中は実行されません。
disk_erasure_concurrency = 1	整数値	ディスクデバイスを消去するのに Ironic Python Agent ramdisk が使用するターゲットプールサイズを定義します。ディスクを消去するために作成されたスレッド数が、この値を超えたり、消去されるディスクの数を超えることはありません。
enable_ata_secure_erase = True	ブール値	クリーニングプロセスで ATA Secure Erase の使用をサポートするかどうか。デフォルトは True です。
enable_nvme_secure_erase = True	ブール値	クリーニングプロセス中に NVMe Secure Erase の使用をサポートするかどうか。現在、nvme-cli format コマンドは、デバイスの機能に応じて、ユーザーデータモードと暗号モードでサポートされています。デフォルトは True です。
erase_devices_metadata_priority = None	整数値	Ironic Python Agent ramdisk を使用してデバイスからメタデータを消去する、帯域インドリーステップを実行する優先度。未設定の場合は、ramdisk で設定した優先順位を使用します (GenericHardwareManager の場合は 99 に設定されます)。0 に設定すると、クリーニング中は実行されません。
erase_devices_priority = None	整数値	Ironic Python Agent ramdisk を通じてデバイスを帯域単位で実行する優先順位。未設定の場合は、ramdisk で設定した優先順位を使用します (GenericHardwareManager のデフォルトは 10)。0 に設定すると、クリーニング中は実行されません。
erase_skip_read_only = False	ブール値	ブロックデバイスが消去される erase_devices クリーンステップの実行時に、ironic-python-agent が読み取り専用デバイスをスキップする必要がある場合。これには、ironic-python-agent 6.0.0 以降が必要です。デフォルトでは、ベアメタルノードのデプロイメント間でデータが保持されるという操作上のセキュリティーリスクが発生する可能性があるため、読み取り専用デバイスは非メタデータベースのクリーニング操作を失敗させます。

設定オプション = デフォルト値	型	説明
external_callback_url = None	文字列値	プロビジョニングネットワークの外部でイメージを提供できる、仮想メディアなどの起動方法のベアメタル API のエージェントコールバック URL。デフォルトは service_catalog からの設定です。
external_http_url = None	文字列値	プロビジョニングネットワークの外部でイメージを提供できる、仮想メディアなどのブート方法用の ironic-conductor ノードの HTTP サーバーの URL。Swift を使用する場合は適用されません。デフォルトは http_url です。
fast_track = False	ブール値	デプロイメントエージェントがマシンライフサイクルの初期状態でハートビート操作を実行でき、ramdisk の通常の設定手順を渡すことができるかどうか。この機能により、fast_track_timeout 設定を使用して ramdisk がハートビート操作を実行すると、デプロイメントプロセスの一部となる電源操作はバイパスされます。
fast_track_timeout = 300	整数値	高速追跡シーケンスの目的で最後のハートビートイベントが有効とみなされる秒数。この設定は、通常 Power-On Self Test および通常の ramdisk 起動の秒数未満にする必要があります。この値は、[api]ramdisk_heartbeat_timeout 設定を超えてはいけません。
http_image_subdir = agent_images	文字列値	ironic-conductor ノードの HTTP ルートパス下のサブディレクトリーの名前。これは、swift tempurls ではなくインスタンスイメージを提供するためにローカルの HTTP サービスが組み込まれた場合に、直接デプロイインターフェイスのインスタンスイメージを配置するために使用されます。
http_root = /httpboot	文字列値	ironic-conductor ノードの HTTP ルートパス
http_url = None	文字列値	ironic-conductor ノードの HTTP サーバー URL例: http://192.1.2.3:8080
power_off_after_deploy_failure = True	ブール値	デプロイの失敗後にノードの電源をオフにするかどうか。デフォルトは True です。
ramdisk_image_download_source = local	文字列値	イメージソース URL を直接使用して独自の元の場所からブート ISO イメージを提供するか、または ironic がイメージをコンダクターにキャッシュして ironic 自身の http サーバーから提供するかを指定します。

設定オプション = デフォルト値	型	説明
shred_final_overwrite_with_zeros = True	ブール値	ランダムデータの書き込み後に、ノードのブロックデバイスにゼロを書き込むかどうか。これにより、deploy.shred_random_overwrite_iterations が 0 の場合でも、ゼロがデバイスに書き込まれます。このオプションは、デバイスが ATA セキュア Erased で使用できない場合にのみ使用されます。デフォルトは True です。
shred_random_overwrite_iterations = 1	整数値	縮小中に、すべてのブロックデバイスを、ランダムなデータで上書きします。これは、デバイスが ATA セキュア Erased で使用できない場合にのみ使用されます。デフォルトは 1 です。

5.1.13. dhcp

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[dhcp]** グループで使用できるオプションの概要をまとめたものです。

表5.12 dhcp

設定オプション = デフォルト値	型	説明
dhcp_provider = neutron	文字列値	使用する DHCP プロバイダー。"neutron" は Neutron を使用します。none は no-op プロバイダーを使用します。

5.1.14. disk_partitioner

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[disk_partitioner]** グループで使用できるオプションの概要をまとめたものです。

表5.13 disk_partitioner

設定オプション = デフォルト値	型	説明
check_device_interval = 1	整数値	Ironic がパーティションテーブルの作成を完了すると、この間隔で、ノードにイメージをコピーする前に、接続されている iSCSI デバイスのアクティビティーが引き続き確認されます (秒単位)。
check_device_max_retries = 20	整数値	別のプロセスでデバイスにアクセスできないことを確認する最大回数。この後もデバイスがビジー状態になると、ディスクのパーティショニングは失敗として扱われます。

5.1.15. disk_utils

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[disk_utils]** グループで使用できるオプションの概要をまとめたものです。

表5.14 disk_utils

設定オプション = デフォルト値	型	説明
bios_boot_partition_size = 1	整数値	BIOS でローカルブート用に GPT パーティションシステムを設定する場合の BIOS ブートパーティションのサイズ (MiB 単位)。
dd_block_size = 1M	文字列値	ノードディスクへの書き込み時に使用するブロックサイズ。
efi_system_partition_size = 200	整数値	ローカルブート用に UEFI システムを設定する場合の EFI システムパーティションのサイズ (MiB 単位)。
image_convert_attempts = 3	整数値	イメージの変換を試行する回数。
image_convert_memory_limit = 2048	整数値	MiB 単位の qemu-img convert のメモリー制限。アドレス領域のリソース制限を使用して実装されます。
partition_detection_attempts = 3	整数値	新規に作成されたパーティションの検出を試行する最大試行。
partprobe_attempts = 10	整数値	パーティションの読み取りを試行する最大試行回数。

5.1.16. drac

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[drac]** グループで使用できるオプションの概要をまとめたものです。

表5.15 drac

設定オプション = デフォルト値	型	説明
bios_factory_reset_timeout = 600	整数値	BIOS 設定の出荷時設定へのリセットが完了するまで待機する最大時間 (秒単位)。
boot_device_job_status_timeout = 30	整数値	ブートデバイスの設定ジョブが正しい状態に遷移し、再起動または電源投入が完了するのを待つ最大時間 (秒) です。
config_job_max_retries = 240	整数値	設定ジョブが正常に完了する最大再試行回数。

設定オプション = デフォルト値	型	説明
query_import_config_job_status_interval = 60	整数値	完了したインポート設定タスクを確認するまで待機する秒数
query_raid_config_job_status_interval = 120	整数値	非同期 RAID 設定が正常に完了したかどうかを判断するために、定期的な RAID ジョブステータスチェックの間隔 (秒単位)。

5.1.17. glance

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[glance]** グループで使用できるオプションの概要をまとめたものです。

表5.16 glance

設定オプション = デフォルト値	型	説明
allowed_direct_url_schemes = []	リスト値	direct_url から直接ダウンロードできる URL スキームのリスト。現在サポートされているスキーム [file]。
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。

設定オプション = デフォルト値	型	説明
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
max-version = None	文字列値	特定の API の最大メジャーバージョンで、min_version を持つ範囲の上限として使用されることが意図されています。version と相互に排他的です。
min-version = None	文字列値	指定の API の最小メジャーバージョンで、max_version を持つ範囲の下限として使用することが意図されています。version と相互に排他的です。max_version なしで min_version が指定されている場合は、max version が latest の場合になります。
num_retries = 0	整数値	glance からイメージをダウンロードする際の再試行回数。
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名

設定オプション = デフォルト値	型	説明
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = image	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
swift_account = None	文字列値	Glance が Swift との通信に使用するアカウント。形式は AUTH_uuid です。uuid は、glance-api.conf で設定したアカウントの UUID です。例: "AUTH_a422b2-91f3-2f46-74b7-d7c9e8958f5d30"設定されていない場合、デフォルト値は Swift へのアクセスに使用するプロジェクトの ID に基づいて計算されます ([swift] セクションで設定)。Swift テンポラリー URL 形式: "endpoint_url/api_version/account/container/object_id"
swift_api_version = v1	文字列値	一時 URL を作成する Swift API バージョン。デフォルトは v1 です。Swift テンポラリー URL 形式: "endpoint_url/api_version/account/container/object_id"
swift_container = glance	文字列値	Swift コンテナー Glance はイメージを保管するように設定されています。デフォルトは glance-api.conf のデフォルトである glance です。Swift テンポラリー URL 形式: "endpoint_url/api_version/account/container/object_id"

設定オプション = デフォルト値	型	説明
swift_endpoint_url = None	文字列値	endpoint_url/api_version/account/container/object_id" 形式の Swift URL の "endpoint" (スキーム、ホスト名、オプションのポート) です。末尾の/は含めないでください。たとえば、"https://swift.example.com" を使用します。RADOS Gateway を使用する場合、エンドポイントには /swift パスが含まれる場合もあります。存在しない場合は、そのパスが追加されます。一時 URL に使用され、指定されていない場合はサービスカタログから取得されます。
swift_store_multiple_containers_seed = 0	整数値	これは、Glance 設定ファイルの同じ名前で設定と一致する必要があります。0 に設定すると、単一テナントのストアはすべてのイメージを格納するために 1 つのコンテナーのみを使用します。1 から 32 までの整数値に設定すると、単一テナントのストアは複数のコンテナーを使用してイメージを保存するので、この値によって作成されるコンテナーの数が決まります。
swift_temp_url_cache_enabled = False	ブール値	生成された Swift の一時 URL をキャッシュするかどうか。これを true に設定すると、イメージキャッシュプロキシーが使用されている場合にのみ便利です。デフォルトは false です。
swift_temp_url_duration = 1200	整数値	一時 URL が有効である期間 (秒単位)。デフォルトは 20 分です。一時 URL からダウンロードしようとすると、デプロイに 401 応答コードが返される場合は、この期間を長くしてください。この値は swift_temp_url_expected_download_start_delay の値以上である必要があります。
swift_temp_url_expected_download_start_delay = 0	整数値	デプロイ要求時 (Swift 一時 URL 生成時) から IPA ラムディスクが起動し、イメージのダウンロードに URL が使用されるまでの遅延 (秒数) です。この値は、イメージのダウンロード開始を可能にするのに十分な大きさで Swift の一時 URL 期間があるかどうかを確認するために使用されます。また、一時的な URL キャッシュが有効になっていると、ダウンロードの開始時にキャッシュされたエントリーが有効であるかどうかを判別します。swift_temp_url_duration の値は、このオプションの値以下である必要があります。デフォルトは 0 です。
swift_temp_url_key = None	文字列値	一時 URL のダウンロードを可能にするために Swift に指定されるシークレットトークン。一時 URL に必要です。Swift バックエンドの場合、サービスプロジェクト上のキー ([swift] セクションに設定) がデフォルトで使用されます。

設定オプション = デフォルト値	型	説明
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。
version = None	文字列値	エンドポイント URL 検出用に指定の Major API バージョン内の最小の Major API バージョン。 min_version および max_version と相互排他的になる

5.1.18. Healthcheck

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[healthcheck]** グループで使用できるオプションの概要をまとめたものです。

表5.17 Healthcheck

設定オプション = デフォルト値	型	説明
backends = []	リスト値	ヘルスチェックを実施し、リクエストの一部としてその情報を報告できる追加のバックエンド。
detailed = False	ブール値	応答の一部として詳細情報を表示します。セキュリティーメモ: このオプションを有効にすると、監視対象のサービスに関する機密情報が漏洩する可能性があります。セキュリティーポリシーに違反しないことを確認してください。

設定オプション = デフォルト値	型	説明
disable_by_file_path = None	文字列値	ファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを確認します。DisableByFileHealthcheck プラグインで使用されます。
disable_by_file_paths = []	リスト値	ポートに基づいてファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを判断します。文字列の port:path リストを想定します。DisableByFilesPortsHealthcheck プラグインで使用されます。
enabled = False	ブール値	/healthcheck でヘルスチェックエンドポイントを有効にします。これは認証されていないことに注意してください。詳細は、https://docs.openstack.org/oslo.middleware/latest/reference/healthcheck_plugins.html を参照してください。
path = /healthcheck	文字列値	healtcheck 要求に応答するパス。

5.1.19. ilo

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[ilo]** グループで使用できるオプションの概要をまとめたものです。

表5.18 ilo

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	iLO を検証する CA 証明書ファイル。
clean_priority_clear_secure_boot_keys = 0	整数値	clear_secure_boot_keys clean ステップの優先度。この手順はデフォルトでは有効になっていません。有効にすると、iLO に登録されたセキュアなブートキーがすべてクリアされます。
clean_priority_reset_bios_to_default = 10	整数値	reset_bios_to_default clean ステップの優先度。
clean_priority_reset_ilo = 0	整数値	reset_ilo のクリーニング手順の優先度。
clean_priority_reset_ilo_credential = 30	整数値	reset_ilo_credential clean ステップの優先度。この手順では、ilo_change_password パラメーターをノードの driver_info で新しいパスワードで更新する必要があります。

設定オプション = デフォルト値	型	説明
clean_priority_reset_secure_boot_keys_to_default = 20	整数値	reset_secure_boot_keys clean ステップの優先度。この手順では、セキュアブートキーを manufacturing のデフォルトにリセットします。
client_port = 443	ポート値	iLO の操作に使用するポート
client_timeout = 60	整数値	iLO 操作のタイムアウト (秒単位)
default_boot_mode = auto	文字列値	ノードの properties/capabilities に boot_mode 機能が提供されていない場合、プロビジョニングで使用するデフォルトのブートモード。デフォルトは、後方互換性のための auto です。auto を指定すると、システムのブートモードの設定に基づいてデフォルトのブートモードが選択されます。
file_permission = 420	整数値	ファイルアクセス許可の 8 進数の許可表現を使用した、Swift レスイメージホスティングのファイル許可。この設定のデフォルトは **644**、または Python では 8 進数 **0o644** です。この設定は、**0o** で始まることを意味する 8 進数表現に設定する必要があります。
kernel_append_params = nofb nomodeset vga=normal	文字列値	インスタンスカーネルに渡す追加のカーネルパラメーター。これらのパラメーターは、/proc/cmdline を読み取ることにより、カーネルまたはアプリケーションで使用できます。厳しいコマンドラインのサイズ制限に注意してください。**instance_info/kernel_append_params** プロパティーでオーバーライドできます。
oob_erase_devices_job_status_interval = 300	整数値	定期的な Erase-devices ステータスチェックの間隔 (秒単位) で、非同期の out-of-band erase-devices が正常に終了したかどうかを判断します。平均して、デフォルトパターン上書きの 300 GB HDD は約 9 時間、デフォルトパターンブロックの 300 GB SSD は約 3 時間かかります。30 秒でサニタイズディスクの消去が完了します。
power_wait = 2	整数値	電源操作間の待機時間 (秒単位)
swift_ilo_container = ironic_ilo_container	文字列値	データを保存するための Swift iLO コンテナー。
swift_object_expiry_timeout = 900	整数値	Swift オブジェクトが自動的に期限切れになるまでの時間 (秒単位)。

設定オプション = デフォルト値	型	説明
use_web_server_for_images = False	ブール値	フロッピーイメージをホストし、ブート ISO を生成するために http Web サーバーを使用するには、このパラメーターを True に設定します。これには、http_root および http_url を設定ファイルの [deploy] セクションで設定する必要があります。これが False に設定されている場合、Ironic は Swift を使用してフロッピーイメージをホストし、boot_iso を生成します。
verify_ca = True	文字列値	iLO を検証するための CA 証明書。これは、ブール値、CA_BUNDLE ファイルへのパス、または信頼できる CA の証明書を含むディレクトリーのいずれかです。True に設定すると、ドライバーはホスト証明書を検証します。False の場合、ドライバーは SSL 証明書の検証を無視します。パスの場合、ドライバーは指定された証明書またはディレクトリー内の証明書の 1 つを使用します。デフォルトは True です。

5.1.20. inspector

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[inspector]** グループで使用できるオプションの概要をまとめたものです。

表5.19 inspector

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
callback_endpoint_override = None	文字列値	ブートが ironic で管理される場合にイントロスペクションデータをポストバックするためのコールバックとして使用するエンドポイント。標準の keystoneauth オプションはデフォルトで使用されます。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル

設定オプション = デフォルト値	型	説明
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
`extra_kernel_params = `	文字列値	(ironic-inspector ではなく) boot が ironic で管理される場合に、イントロスペクション ramdisk に渡す追加のカーネルパラメーターkey=value をスペースで区切って組み合わせます。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
max-version = None	文字列値	特定の API の最大メジャーバージョンで、min_version を持つ範囲の上限として使用されることが意図されています。version と相互に排他的です。

設定オプション = デフォルト値	型	説明
min-version = None	文字列値	指定の API の最小メジャーバージョンで、max_version を持つ範囲の下限として使用することが意図されています。version と相互に排他的です。max_version なしで min_version が指定されている場合は、max version が latest の場合になります。
password = None	文字列値	ユーザーのパスワード
power_off = True	ブール値	イントロスペクションが完了した後にノードの電源をオフにするかどうか
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
require_managed_boot = False	ブール値	インバンド検査ブートが ironic によって完全に管理されることを要求します。ironic-inspector のインストールに別の PXE ブート環境がない場合は、このパラメーターを True に設定します。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = baremetal-introspection	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。

設定オプション = デフォルト値	型	説明
status_check_period = 60	整数値	検査時にノードのステータスを確認する期間 (秒単位)
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。
version = None	文字列値	エンドポイント URL 検出用に指定の Major API バージョン内の最小の Major API バージョン。min_version および max_version と相互排他的になる

5.1.21. ipmi

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[ipmi]** グループで使用できるオプションの概要をまとめたものです。

表5.20 ipmi

設定オプション = デフォルト値	型	説明
additional_retryable_ipmi_errors = []	多値	実行中の環境に固有の追加のエラー ipmitool が発生する可能性があります。
cipher_suite_versions = []	リスト値	フィールド **cipher_suite** がノードに設定されていない場合に、ハードウェアでサポートできる可能な暗号スイートバージョンのリスト。

設定オプション = デフォルト値	型	説明
command_retry_timeout = 60	整数値	再試行可能な IPMI 操作を再試行する最大時間 (秒単位)。(例えば、BMC がビジー状態であるために要求された操作が失敗した場合、操作は再試行可能です)。これを高く設定すると、BMC が遅い場合や応答しない場合に、同期電源状態の定期的なタスクがハングする可能性があります。
debug = False	ブール値	すべての ipmi コマンドを追加のデバッグ出力で実行できるようにします。これは、ipmitool が、このモードにおいて非常に誤解を招くテキストを記録できるため、個別のオプションです。
disable_boot_timeout = True	ブール値	デフォルトのタイムアウト動作では、ironic が raw IPMI コマンドを送信してブート用に 60 秒のタイムアウトを無効にするかどうかを設定します。このオプションを False に設定すると、そのコマンドは送信されません。デフォルト値は True です。これは、ノードの **driver_info** フィールドのノードごとの **ipmi_disable_boot_timeout** オプションで上書きできます。
kill_on_timeout = True	ブール値	**command_retry_timeout** タイムアウトの期限が切れた後に **ipmitool** プロセスが終了しない場合には、ironic が起動している **ipmitool** プロセスを強制終了します。推奨される設定は True
min_command_interval = 5	整数値	サーバーに送信される IPMI 操作間の最小時間 (秒単位)。これを低く設定しすぎるハードウェアによっては、BMC がクラッシュするリスクがあります。推奨される設定は 5 秒です。
use_ipmitool_retries = False	ブール値	True に設定され、パラメーターが ipmitool によってサポートされる場合、再試行回数と再試行の間隔がパラメーターとして ipmitool に渡され、ipmitool は再試行を行います。False に設定すると、ironic は ipmitool コマンドを再試行します。推奨設定は False です。

5.1.22. irmc

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[irmc]** グループで使用できるオプションの概要をまとめたものです。

表5.21 irmc

設定オプション = デフォルト値	型	説明
auth_method = basic	文字列値	iRMC 操作に使用する認証方法
clean_priority_restore_irmc_bios_config = 0	整数値	restore_irmc_bios_config clean ステップの優先度。
client_timeout = 60	整数値	iRMC 操作のタイムアウト (秒単位)
fpga_ids = []	リスト値	検査する CPU FPGA のベンダー ID およびデバイス ID のリスト。リスト項目は vendorID/deviceID 形式であり、コンマで区切ります。CPU 検査はこの値を使用して、ノードに CPU FPGA の有無を確認します。このオプションが定義されていない場合は、ノードトレイトの CUSTOM_CPU_FPGA を除外します。サンプル fpga_ids 値: 0x1000/0x0079,0x2100/0x0080
gpu_ids = []	リスト値	検査する GPU デバイスのベンダー ID およびデバイス ID のリスト。リスト項目は vendorID/deviceID 形式であり、コンマで区切ります。GPU 検査は、この値を使用してノードの GPU デバイスの数をカウントします。このオプションが定義されていない場合には、pci_gpu_devices を capabilities プロパティーのままにします。サンプル gpu_ids 値: 0x1000/0x0079,0x2100/0x0080
port = 443	ポート値	iRMC 操作に使用するポート
query_raid_config_fgi_status_interval = 300	整数値	非同期 RAID 設定が正常に完了したかどうかを判断するために、定期的な RAID ステータスチェックの間隔 (秒単位)。フォアグラウンド初期化 (FGI) は、仮想ドライブの作成後 5 分後に起動します。
remote_image_server = None	文字列値	リモートイメージサーバーの IP
remote_image_share_name = share	文字列値	remote_image_server の共有名
remote_image_share_root = /remote_image_share_root	文字列値	ironic コンダクターノードの NFS または CIFS のルートパス
remote_image_share_type = CIFS	文字列値	仮想メディアの共有種別
`remote_image_user_domain = `	文字列値	remote_image_user_name のドメイン名

設定オプション = デフォルト値	型	説明
remote_image_user_name = None	文字列値	remote_image_server のユーザー名
remote_image_user_password = None	文字列値	remote_image_user_name のパスワード
sensor_method = ipmitool	文字列値	センサーデータの取得方法。
snmp_auth_proto = sha	文字列値	SNMPv3 メッセージ認証プロトコル ID。バージョン "**v3** で必須です。python-scciclient のバージョンが 0.10.1 より前の場合は無視されます。有効なオプションは **sha**、**sha256**、**sha384**、および **sha512** です。iRMC S4 および S5 でサポートされているプロトコルは **sha** のみで、iRMC S6 以降では、**sha256**、**sha384**、**sha512** がサポートされますが、**sha** はサポートされなくなりました。
snmp_community = public	文字列値	SNMP コミュニティー。バージョン "v1" および "v2c" で必須
snmp_polling_interval = 10	整数値	SNMP ポーリングの間隔 (秒単位)
snmp_port = 161	ポート値	SNMP ポート
snmp_priv_proto = aes	文字列値	SNMPv3 メッセージプライバシー (暗号化) プロトコル ID。バージョン "**v3** で必須です。python-scciclient のバージョンが 0.10.1 より前の場合は無視されます。**aes** はサポートされています。
snmp_security = None	文字列値	SNMP セキュリティー名。バージョン "**v3** で必須です。driver_info/irmc_snmp_user が設定されている場合は無視されます。
snmp_version = v2c	文字列値	SNMP プロトコルのバージョン

5.1.23. ironic_lib

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[ironic_lib]** グループで使用できるオプションの概要をまとめたものです。

表5.22 ironic_lib

設定オプション = デフォルト値	型	説明
fatal_exception_format_errors = False	ブール値	例外メッセージを生成する際にフォーマットエラーがある場合に使用されます (プログラミングエラー)。True の場合には、例外を発生させ、False の場合はフォーマットされていないメッセージを使用します。
root_helper = sudo ironic-rootwrap /etc/ironic/rootwrap.conf	文字列値	root として実行するコマンドに接頭辞が付けられたコマンド。指定しない場合、コマンドは root として実行されません。

5.1.24. iscsi

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[iscsi]** グループで使用できるオプションの概要をまとめたものです。

表5.23 iscsi

設定オプション = デフォルト値	型	説明
conv_flags = None	文字列値	ホストにコピーする際に元のファイルを変換するため、dd コマンドに送信する必要があるフラグ。コンマで区切られた複数のオプションを含めることができます。
portal_port = 3260	ポート値	iSCSI ポータルが受信接続をリッスンするポート番号。
verify_attempts = 3	整数値	iSCSI 接続が有効であることの確認を試行する最大試行。1 秒の間にスリープします。デフォルトは 3 です。

5.1.25. json_rpc

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[json_rpc]** グループで使用できるオプションの概要をまとめたものです。

表5.24 json_rpc

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_strategy = None	文字列値	JSON RPC が使用する認証ストラテジー。デフォルトは global auth_strategy 設定です。

設定オプション = デフォルト値	型	説明
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
host_ip = ::	ホストのアドレス値	JSON RPC がリッスンする IP アドレスまたはホスト名。
http_basic_auth_user_file = /etc/ironic/htpasswd-json-rpc	文字列値	auth_strategy=http_basic の場合に使用される Apache 形式のユーザー認証ファイルへのパス
http_basic_password = None	文字列値	HTTP 基本認証クライアント要求に使用するパスワード。
http_basic_username = None	文字列値	HTTP 基本認証クライアント要求に使用するユーザーの名前。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
password = None	文字列値	ユーザーのパスワード
port = 8089	ポート値	JSON RPC に使用するポート

設定オプション = デフォルト値	型	説明
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
use_ssl = False	ブール値	JSON RPC に TLS を使用するかどうか
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名

5.1.26. keystone_authtoken

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[keystone_authtoken]** グループで使用できるオプションの概要をまとめたものです。

表5.25 keystone_authtoken

設定オプション = デフォルト値	型	説明
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション

設定オプション = デフォルト値	型	説明
auth_type = None	文字列値	読み込む認証タイプ
auth_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。このオプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。**非推奨:**Queens 以降 *理由:* auth_uri オプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。
auth_version = None	文字列値	Identity API エンドポイントの API バージョン。
cache = None	文字列値	Swift キャッシュオブジェクトが保存される環境キーを要求します。auth_token ミドルウェアを Swift キャッシュと共にデプロイする場合は、このオプションを使用して、ミドルウェアが swift とキャッシングバックエンドを共有するようにします。それ以外の場合は、代わりに **memcached_servers** オプションを使用します。
cafile = None	文字列値	HTTPs 接続の検証時に使用する PEM でエンコードされた認証局。デフォルトはシステム CA です。
certfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
delay_auth_decision = False	ブール値	ミドルウェア内の認可要求を処理せず、承認の決定をダウンストリームの WSGI コンポーネントに委譲します。

設定オプション = デフォルト値	型	説明
enforce_token_bind = permissive	文字列値	トークンバインディングの使用および種別を制御するために使用されます。トークンバインディングのチェックを無効するには、"disabled"に設定します。バインドタイプがサーバーの認識する形式の場合にはバインディング情報を検証し、そうでない場合には無視するには、"permissive"(デフォルト) に設定します。"strict"は"permissive"と類似していますが、バインドタイプが不明な場合にはトークンが拒否されます。"required"の場合は、いずれかの形式のトークンバインディングが必要です。最後に、トークンに指定する必要のあるバインディングメソッドの名前。
http_connect_timeout = None	整数値	Identity API サーバーと通信する際の要求タイムアウト値。
http_request_max_retries = 3	整数値	Identity API サーバーと通信する際に再接続を試行する回数。
include_service_catalog = True	ブール値	(オプション)X-Service-Catalog ヘッダーを設定するかどうかを示します。False の場合、ミドルウェアはトークンの検証時にサービスカタログを要求せず、X-Service-Catalog ヘッダーを設定しません。
insecure = False	ブール値	HTTPS 接続を確認します。
interface = internal	文字列値	Identity API エンドポイントに使用するインターフェイス。有効な値は、"public"、"internal" (デフォルト)、または "admin" です。
keyfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
memcache_pool_conn_get_timeout = 10	整数値	(オプション) プールから memcached クライアント接続を取得するまで操作が待機する秒数。
memcache_pool_dead_retry = 300	整数値	(オプション)memcached サーバーが停止しているとみなされる秒数。この秒数が経過すると再試行されます。
memcache_pool_maxsize = 10	整数値	(オプション) すべての memcached サーバーへのオープン接続の最大合計数。
memcache_pool_socket_timeout = 3	整数値	(オプション)memcached サーバーと通信する際のソケットのタイムアウト (秒単位)。

設定オプション = デフォルト値	型	説明
memcache_pool_unused_timeout = 60	整数値	(オプション)memcached への接続がプール内で未使用の状態を維持する秒数。この秒数が経過すると終了されます。
memcache_secret_key = None	文字列値	(オプション、memcache_security_strategy が定義されている場合には必須) この文字列は鍵の導出に使用されます。
memcache_security_strategy = None	文字列値	(オプション) 定義されている場合は、トークンデータを認証、または認証して暗号化する必要があるかどうかを示します。MAC の場合、キャッシュでトークンデータが認証されます (HMAC を使用)。ENCRYPT の場合、キャッシュでトークンデータが暗号化され、認証されます。値がこれらのオプションのいずれでもない場合や空の場合には、auth_token は初期化時に例外を発生させます。
memcache_use_advanced_pool = False	ブール値	(オプション) 高度な (eventlet に対して安全な) memcached クライアントプールを使用します。高度なプールは python 2.x でのみ動作します。
memcached_servers = None	リスト値	オプションで、キャッシュに使用する memcached サーバーのリストを指定します。未定義のままの場合、トークンは代わりに処理中にキャッシュされます。
region_name = None	文字列値	アイデンティティーサーバーがあるリージョン。
service_token_roles = ['service']	リスト値	サービストークンに存在する必要があるロールの選択。サービストークンは、期限切れのトークンを使用できることを要求できるため、このチェックでは実際のサービスのみがこのトークンを送信するように厳密に制御する必要があります。ここでのロールは ANY チェックとして適用されるため、このリストのロールはすべて存在している必要があります。後方互換性の理由から、現在 allow_expired チェックにのみ影響します。
service_token_roles_required = False	ブール値	後方互換性の理由から、service_token_roles チェックを有効としてパスしない有効なサービストークンをパスさせる必要があります。これを true に設定することが今後のリリースでデフォルトとなり、可能な場合は有効にされる必要があります。
service_type = None	文字列値	サービスカタログに表示されるサービスの名前または種別。これは、制限されたアクセスルールがあるトークンを検証するために使用されます。

設定オプション = デフォルト値	型	説明
token_cache_time = 300	整数値	トークンの検証に過剰な時間を費やすのを防ぐために、ミドルウェアは、設定可能な期間 (秒単位) 中は以前に見たトークンをキャシュします。キャッシュを完全に無効にするには -1 に設定します。
www_authenticate_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。

5.1.27. mdns

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[mdns]** グループで使用できるオプションの概要をまとめたものです。

表5.26 mdns

設定オプション = デフォルト値	型	説明
interfaces = None	リスト値	mDNS に使用するインターフェイスの IP アドレスのリスト。デフォルトは、システムのすべてのインターフェイスに設定されます。
lookup_attempts = 3	整数値	サービスのルックアップを試行する回数。
params = {}	dict 値	登録したサービスに渡す追加のパラメーター。
registration_attempts = 5	整数値	サービス登録を試行する回数。現在、zeroconf ライブラリーの競合状態により、1 を超える必要があります。

5.1.28. metrics

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[metrics]** グループで使用できるオプションの概要をまとめたものです。

表5.27 metrics

設定オプション = デフォルト値	型	説明
agent_backend = noop	文字列値	メトリックに使用するエージェント ramdisk のバックエンド。デフォルトの可能なバックエンドは noop および statsd です。
agent_global_prefix = None	文字列値	エージェント ramdisk が送信したすべてのメトリック名の前に、この値を付けます。メトリック名の形式は [global_prefix.][uuid.] [host_name.]prefix.metric_name です。
agent_prepend_host = False	ブール値	エージェント ramdisk が送信するすべてのメトリック名にホスト名を付けます。メトリック名の形式は [global_prefix.][uuid.] [host_name.]prefix.metric_name です。
agent_prepend_host_reverse = True	ブール値	先頭に追加されたホスト値を.で分割し、エージェント ramdisk が送信するメトリックを逆方向にします (ドメイン名の逆階層形式により適切に一致します)。
agent_prepend_uuid = False	ブール値	ノードの Ironic uuid を、エージェント ramdisk が送信する全メトリック名に追加します。メトリック名の形式は [global_prefix.][uuid.] [host_name.]prefix.metric_name です。
backend = noop	文字列値	メトリックシステムに使用するバックエンド。
global_prefix = None	文字列値	すべてのメトリック名の前に、この値を付けます。デフォルトでは、グローバル接頭辞はありません。メトリック名の形式は [global_prefix.] [host_name.]prefix.metric_name です。
prepend_host = False	ブール値	ホスト名をすべてのメトリック名に追加します。メトリック名の形式は [global_prefix.] [host_name.]prefix.metric_name です。
prepend_host_reverse = True	ブール値	先頭に追加されたホスト値を.で分割し、逆にします (ドメイン名の逆引き階層形式により適切)。

5.1.29. metrics_statsd

以下の表では、**/etc/ironic/ironic.conf** ファイルの **[metrics_statsd]** グループで利用可能なオプションの概要を説明します。

表5.28 metrics_statsd

設定オプション = デフォルト値	型	説明
agent_statsd_host = localhost	文字列値	statsd バックエンドで使用するエージェント ramdisk のホスト。これは、エージェントが起動するネットワークからアクセス可能である必要があります。
agent_statsd_port = 8125	ポート値	statsd バックエンドで使用するエージェント ramdisk のポート。
statsd_host = localhost	文字列値	statsd バックエンドで使用するホスト。
statsd_port = 8125	ポート値	statsd バックエンドで使用するポート。

5.1.30. モールド

次の表は、**/etc/ironic/ironic.conf** ファイルの **[molds]** グループで使用できるオプションの概要を示しています。

表5.29 モールド

設定オプション = デフォルト値	型	説明
password = None	文字列値	http 基本認証のパスワード。デフォルトでは、空に設定されています。
retry_attempts = 3	整数値	設定モールドの保存または取得を再試行してください。
retry_interval = 3	整数値	設定モールドの保存または取得の再試行間隔。
storage = swift	文字列値	設定モールドの保存場所。swift と http に対応。デフォルトでは swift です。
user = None	文字列値	http 基本認証のユーザー。デフォルトでは、空に設定されています。

5.1.31. neutron

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[neutron]** グループで使用できるオプションの概要をまとめたものです。

表5.30 neutron

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
add_all_ports = False	ブール値	プロビジョニング、クリーニング、またはレスキュー用のポートの作成時に全ポートを neutron に送信できるようにするオプション。これは、ポートに割り当てられた IP アドレスを使用せずに行われ、ボンディングされたネットワーク設定で役に立つ場合があります。
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
cleaning_network = None	文字列値	ノードのクリーニング用に ramdisk が起動される Neutron ネットワーク UUID または名前。neutron ネットワークインターフェイスに必要です。また、flat ネットワークインターフェイスまたは neutronDHCP プロバイダーを使用する場合に、ノードをクリーニングする必要もあります。名前を指定すると、すべてのネットワーク間で一意である必要があり、クリーニングは失敗します。
cleaning_network_security_groups = []	リスト値	ノードのクリーニング中に適用される Neutron セキュリティーグループ UUID のリストneutron ネットワークインターフェイスの場合はオプションであり、flat または noop ネットワークインターフェイスに使用されません。指定されていない場合は、default セキュリティーグループが使用されます。
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。

設定オプション = デフォルト値	型	説明
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
dhcpv6_stateful_address_count = 4	整数値	DHCPv6 ステートフルネットワークでのプロビジョニング、クリーニング、レスキュー、または検査用に作成されるポートに割り当てる IPv6 アドレスの数。チェーンローディングプロセスのステージごとに、異なる CLID/IAID のアドレスが必要です。同一ではない識別子により、起動プロセスの各手順が正常にアドレスをリースできるように、ホスト用に複数のアドレスを確保する必要があります。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
inspection_network = None	文字列値	ノードのインバンド検査用に ramdisk が起動される Neutron ネットワーク UUID または名前。名前を指定すると、すべてのネットワークまたは検査で一意でなければなりません。
inspection_network_security_groups = []	リスト値	ノードの検査プロセス中に適用される Neutron セキュリティーグループ UUID のリスト。neutron ネットワークインターフェイスの場合はオプションであり、flat または noop ネットワークインターフェイスに使用されません。指定されていない場合は、default セキュリティーグループが使用されます。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
max-version = None	文字列値	特定の API の最大メジャーバージョンで、min_version を持つ範囲の上限として使用されることが意図されています。version と相互に排他的です。

設定オプション = デフォルト値	型	説明
min-version = None	文字列値	指定の API の最小メジャーバージョンで、max_version を持つ範囲の下限として使用することが意図されています。version と相互に排他的です。max_version なしで min_version が指定されている場合は、max version が latest の場合になります。
password = None	文字列値	ユーザーのパスワード
port_setup_delay = 0	整数値	Neutron エージェントがポート用に十分な DHCP 設定を設定することを待機する遅延値。
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
provisioning_network = None	文字列値	ノードのプロビジョニング用に ramdisk がブートする Neutron ネットワーク UUID または名前。neutron ネットワークインターフェイスに必要です。名前を指定すると、すべてのネットワーク間で一意である必要があり、デプロイは失敗します。
provisioning_network_security_groups = []	リスト値	ノードのプロビジョニング時に適用される Neutron セキュリティーグループ UUID のリスト。neutron ネットワークインターフェイスの場合はオプションであり、flat または noop ネットワークインターフェイスに使用されません。指定されていない場合は、default セキュリティーグループが使用されます。
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
request_timeout = 45	整数値	Neutron との対話時の要求処理のタイムアウト。neutron が API クライアントに戻る前に事前コミットの検証を行うため、neutron が通常のクライアント/サーバーの対話よりも長い時間がかかる可能性があるため、この値を大きくする必要があります。

設定オプション = デフォルト値	型	説明
rescuing_network = None	文字列値	レスキューモード用に ramdisk を起動する Neutron ネットワーク UUID または名前。これは、レスキュー ramdisk がデプロイ後のテナントネットワークを使用するネットワークではなく、そのネットワークに使用されます。neutron ネットワークインターフェイスに必要です。レスキューモードが使用されます。flat または noop ネットワークインターフェイスには使用されません。名前を指定すると、すべてのネットワーク間で一意である必要があり、すべてのネットワークで一意である必要があり、レスキューは失敗します。
rescuing_network_security_groups = []	リスト値	ノードのレスキュープロセス中に適用される Neutron セキュリティーグループ UUID のリスト neutron ネットワークインターフェイスの場合はオプションであり、flat または noop ネットワークインターフェイスに使用されません。指定されていない場合は、default セキュリティーグループが使用されます。
retries = 3	整数値	非推奨: リクエストが失敗した場合、クライアントは再試行します。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = network	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値

設定オプション = デフォルト値	型	説明
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。
version = None	文字列値	エンドポイント URL 検出用に指定の Major API バージョン内の最小の Major API バージョン。 min_version および max_version と相互排他的になる

5.1.32. nova

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[nova]** グループで使用できるオプションの概要をまとめたものです。

表5.31 nova

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。 設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。

設定オプション = デフォルト値	型	説明
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
max-version = None	文字列値	特定の API の最大メジャーバージョンで、min_version を持つ範囲の上限として使用されることが意図されています。version と相互に排他的です。
min-version = None	文字列値	指定の API の最小メジャーバージョンで、max_version を持つ範囲の下限として使用することが意図されています。version と相互に排他的です。max_version なしで min_version が指定されている場合は、max version が latest の場合になります。
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID

設定オプション = デフォルト値	型	説明
project-name = None	文字列値	スコープするプロジェクト名
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
send_power_notifications = True	ブール値	True に設定すると、電源状態変更コールバックのサポートが nova に有効化されます。このオプションは、openstack Compute サービスを持たないデプロイメントでは False に設定する必要があります。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = compute	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名

設定オプション = デフォルト値	型	説明
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。
version = None	文字列値	エンドポイント URL 検出用に指定の Major API バージョン内の最小の Major API バージョン。min_version および max_version と相互排他的になる

5.1.33. oslo_concurrency

以下の表では、**/etc/ironic/ironic.conf** ファイルの **[oslo_concurrency]** グループで利用可能なオプションについて説明しています。

表5.32 oslo_concurrency

設定オプション = デフォルト値	型	説明
disable_process_locking = False	ブール値	プロセス間ロックを有効または無効にします。
lock_path = None	文字列値	ロックファイルに使用するディレクトリー。セキュリティー上、指定したディレクトリーは、ロックが必要なプロセスを実行しているユーザーのみが書き込み可能である必要があります。デフォルトは環境変数 OSLO_LOCK_PATH です。外部ロックを使用する場合は、ロックパスを設定する必要があります。

5.1.34. oslo_messaging_amqp

以下の表では、**/etc/ironic/ironic.conf** ファイルの **[oslo_messaging_amqp]** グループで利用可能なオプションの概要を説明します。

表5.33 oslo_messaging_amqp

設定オプション = デフォルト値	型	説明
addressing_mode = dynamic	文字列値	ドライバーが使用するアドレスモードを示します。許可される値:**legacy** - レガシーのルーティング不可能なアドレス指定を使用します。**routable** - ルーティング可能なアドレスを使用します。**dynamic** - メッセージバスがルーティングをサポートしていない場合はレガシーのアドレスを使用します。それ以外の場合は、ルーティング可能なアドレスを使用します。

設定オプション = デフォルト値	型	説明
anycast_address = anycast	文字列値	コンシューマーのグループに送信する際に、アドレス接頭辞に追加されます。コンシューマー間でラウンドロビン方式で配信される必要のあるメッセージを特定するために、メッセージバスで使用されます。
broadcast_prefix = broadcast	文字列値	すべてのサーバーにブロードキャストする場合に使用されるアドレス接頭辞
connection_retry_backoff = 2	整数値	フェイルオーバーの試みに失敗するたびに、connection_retry_interval をこの秒数だけ増やします。
connection_retry_interval = 1	整数値	再接続を試みる前に一時停止する秒数。
connection_retry_interval_max = 30	整数値	connection_retry_interval + connection_retry_backoff の上限
container_name = None	文字列値	AMQP コンテナーの名前。グローバルで一意でなければなりません。デフォルトは、生成された UUID です。
default_notification_exchange = None	文字列値	通知アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_notification_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **notify**
default_notify_timeout = 30	整数値	送信された通知メッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_reply_retry = 0	整数値	リカバリー可能なエラーが原因で失敗した返信メッセージを再送信する最大試行回数。
default_reply_timeout = 30	整数値	rpc 返信メッセージ配信の期限。
default_rpc_exchange = None	文字列値	RPC アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_rpc_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **rpc**

設定オプション = デフォルト値	型	説明
default_send_timeout = 30	整数値	rpc キャストまたは呼び出しメッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_sender_link_timeout = 600	整数値	アイドル状態の送信者リンクのパージをスケジュールする時間。期限切れ後にリンクの割り当てを解除します。
group_request_prefix = unicast	文字列値	グループの任意のサーバーに送信する際のアドレス接頭辞
idle_timeout = 0	整数値	非アクティブな接続のタイムアウト (秒単位)
link_retry_delay = 10	整数値	リカバリー可能なエラーが原因で失敗した AMQP 1.0 リンクの再接続間に一時停止する時間。
multicast_address = multicast	文字列値	ファンアウトメッセージを送信するときにアドレス接頭辞に追加されます。ファンアウトメッセージを識別するためにメッセージバスによって使用されます。
notify_address_prefix = openstack.org/om/notify	文字列値	すべての生成される通知アドレスのアドレス接頭辞
notify_server_credit = 100	整数値	受信通知メッセージのウィンドウサイズ
pre_settled = ['rpc-cast', 'rpc-reply']	多値	このタイプのメッセージを事前処置状態で送信します。事前処置されたメッセージは、ピアから確認応答を受信しません。注記: 事前処置されたメッセージは、配信に失敗した場合に警告なしに破棄される可能性があります。許可される値:**rpc-call** - RPC 呼び出しを事前処置状態で送信します。**rpc-reply**- RPC 返信を事前処置状態で送信します。**rpc-cast** - RPC キャストを事前処置状態で送信します。**notify** - 通知を事前処置状態で送信します。
pseudo_vhost = True	ブール値	仮想ホスト (qpidd など) をネイティブにサポートしていないメッセージバスの仮想ホストサポートを有効にします。true に設定すると、仮想ホスト名はすべてのメッセージバスアドレスに追加され、結果として仮想ホストごとにプライベート **サブネット** を作成します。メッセージバスが仮想ホストの名前として AMQP 1.0 オープン遂行動詞の **hostname** フィールドを使用して仮想ホストをサポートする場合は False に設定します。

設定オプション = デフォルト値	型	説明
reply_link_credit = 200	整数値	受信する RPC 返信メッセージのウィンドウサイズ。
rpc_address_prefix = openstack.org/om/rpc	文字列値	すべての生成される RPC アドレスのアドレス接頭辞
rpc_server_credit = 100	整数値	受信する RPC リクエストメッセージのウィンドウサイズ。
`sasl_config_dir = `	文字列値	SASL 設定が含まれるディレクトリーへのパス
`sasl_config_name = `	文字列値	設定ファイルの名前 (.conf 接尾辞なし)
`sasl_default_realm = `	文字列値	ユーザー名にレルムが存在しない場合に使用する SASL レルム
`sasl_mechanisms = `	文字列値	許可される SASL メカニズムのスペース区切りリスト
server_request_prefix = exclusive	文字列値	特定のサーバーに送信する時に使用するアドレス接頭辞
ssl = False	ブール値	SSL 経由で接続を試みます。その他の ssl 関連のパラメーターが指定されていない場合、システムの CA バンドルを使用してサーバーの証明書を検証します。
`ssl_ca_file = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_cert_file = `	文字列値	クライアント認証用の自己識別証明書 PEM ファイル
`ssl_key_file = `	文字列値	ssl_cert_file 証明書の署名に使用される秘密鍵 PEM ファイル (オプション)
ssl_key_password = None	文字列値	ssl_key_file を復号するためのパスワード (暗号化されている場合)
ssl_verify_vhost = False	ブール値	デフォルトでは、SSL は、サーバーの証明書の名前が transport_url のホスト名と一致することを確認します。設定によっては、代わりに仮想ホスト名を使用することが望ましい場合があります。たとえば、サーバーが Server Name Indication TLS 拡張 (rfc6066) を使用して、仮想ホストごとの証明書を提供する場合などです。サーバーの SSL 証明書が DNS 名ではなく仮想ホスト名を使用する場合は、ssl_verify_vhost を True に設定します。

設定オプション = デフォルト値	型	説明
trace = False	ブール値	デバッグ: AMQP フレームを標準出力 (stdout) にダンプします。
unicast_address = unicast	文字列値	特定の RPC/通知サーバーに送信する際に、アドレス接頭辞に追加されます。単一の送信先に送信されたメッセージを識別するためにメッセージバスによって使用されます。

5.1.35. oslo_messaging_kafka

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[oslo_messaging_kafka]** グループで使用できるオプションの概要をまとめたものです。

表5.34 oslo_messaging_kafka

設定オプション = デフォルト値	型	説明
compression_codec = none	文字列値	プロデューサーによって生成されたすべてのデータの圧縮コーデック。設定されていない場合、圧縮は使用されません。この設定に許可される値は、kafka バージョンに依存することに注意してください。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
consumer_group = oslo_messaging_consumer	文字列値	Kafka コンシューマーのグループ ID。あるグループのコンシューマーは、メッセージ消費を調整します。
enable_auto_commit = False	ブール値	非同期コンシューマーコミットを有効にします。
kafka_consumer_timeout = 1.0	浮動小数点の値	Kafka コンシューマーのデフォルトタイムアウト
kafka_max_fetch_bytes = 1048576	整数値	Kafka コンシューマーの最大フェッチバイト
max_poll_records = 500	整数値	ポーリング呼び出しで返されるレコードの最大数
pool_size = 10	整数値	Kafka コンシューマーのプールサイズ
producer_batch_size = 16384	整数値	プロデューサーの非同期送信のバッチサイズ

設定オプション = デフォルト値	型	説明
producer_batch_timeout = 0.0	浮動小数点の値	KafkaProducer バッチ処理の遅延の上限 (秒単位)
sasl_mechanism = PLAIN	文字列値	セキュリティープロトコルが SASL である際のメカニズム
security_protocol = PLAINTEXT	文字列値	ブローカーとの通信に使用されるプロトコル
`ssl_cafile = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_client_cert_file = `	文字列値	認証に使用されるクライアント証明書の PEM ファイル。
`ssl_client_key_file = `	文字列値	認証に使用されるクライアントキーの PEM ファイル。
`ssl_client_key_password = `	文字列値	認証に使用されるクライアントキーパスワードファイル。

5.1.36. oslo_messaging_notifications

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[oslo_messaging_notifications]** グループで使用できるオプションの概要をまとめたものです。

表5.35 oslo_messaging_notifications

設定オプション = デフォルト値	型	説明
driver = []	多値	通知の送信を処理するドライバー。使用できる値は messaging、messagingv2、routing、log、test、noop です。
retry = -1	整数値	リカバリー可能なエラーが原因で配信に失敗した通知メッセージを再送信する最大試行回数。0 - 再試行なし、-1 - 無限回
topics = ['notifications']	リスト値	OpenStack の通知に使用する AMQP トピック。
transport_url = None	文字列値	通知に使用するメッセージングドライバーを表す URL。設定されていない場合は、RPC に使用されるものと同じ設定にフォールバックします。

5.1.37. oslo_messaging_rabbit

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[oslo_messaging_rabbit]** グループで使用できるオプションの概要をまとめたものです。

表5.36 oslo_messaging_rabbit

設定オプション = デフォルト値	型	説明
amqp_auto_delete = False	ブール値	AMQP の自動削除キュー。
amqp_durable_queues = False	ブール値	AMQP で永続キューを使用します。
direct_mandatory_flag = True	ブール値	(非推奨) ダイレクト送信のための RabbitMQ 必須フラグを有効/無効にします。直接送信は応答として使用されるため、クライアントキューが存在しない場合には MessageUndeliverable 例外が発生します。MessageUndeliverable 例外はタイムアウトをループして、送信者が回復する可能性が高くなります。このフラグは非推奨になり、この機能を非アクティブ化することはできません。
enable_cancel_on_failover = False	ブール値	x-cancel-on-ha-failover フラグを有効にして、rabbitmq サーバーがキューが停止しているときにコンシューマーをキャンセルし、通知できるようにします。
heartbeat_in_pthread = False	ブール値	デフォルトでは、ネイティブ Python スレッドを介してヘルスチェックハートビートスレッドを実行します。このオプションが False の場合、ヘルスチェックハートビートは親プロセスから実行モデルを継承します。たとえば、親プロセスが eventlet/greenlet を使用して stdlib にモンキーパッチを適用した場合、ハートビートはグリーンスレッドを介して実行されます。This option should be set to True only for the wsgi services.
heartbeat_rate = 2	整数値	heartbeat_timeout_threshold 中、ハートビートを確認する回数。
heartbeat_timeout_threshold = 60	整数値	ハートビートの keep-alive が失敗した場合に Rabbit ブローカーがダウンとみなされるまでの秒数 (0 はハートビートを無効にします)。
kombu_compression = None	文字列値	実験的用途: 許容値は gzip、bz2 です。設定されていない場合、圧縮は使用されません。このオプションは、今後のバージョンで利用できない可能性があります。

設定オプション = デフォルト値	型	説明
kombu_failover_strategy = round-robin	文字列値	現在接続しているノードが利用できなくなった場合に、次の RabbitMQ ノードを選択する方法を指定します。設定で複数の RabbitMQ ノードが指定される場合に限り有効になります。
kombu_missing_consumer_retry_timeout = 60	整数値	応答のないクライアントが応答を送信するのを待つ時間。この時間が経過すると無視します。この値は rpc_response_timeout より長くすることはできません。
kombu_reconnect_delay = 1.0	浮動小数点の値	AMQP コンシューマーの取り消し通知への応答で、再接続するまでの待機時間。
rabbit_ha_queues = False	ブール値	RabbitMQ (x-ha-policy: all) の HA キューの使用を試みます。このオプションを変更する場合は、RabbitMQ データベースを消去する必要があります。RabbitMQ 3.0 では、キューを宣言する際に x-ha-policy 引数によってキューのミラーリングが制御されなくなりました。すべてのキュー (自動生成された名前のキューを除く) がすべてのノードでミラーリングされるようにするには、"rabbitmqctl set_policy HA **^(?!amq\.).* {"ha-mode": "all"}**" を実行します。
rabbit_interval_max = 30	整数値	RabbitMQ 接続を再試行する最大間隔。デフォルトは 30 秒です。
rabbit_login_method = AMQPLAIN	文字列値	RabbitMQ ログイン方法
rabbit_qos_prefetch_count = 0	整数値	事前フェッチするメッセージの数を指定します。ゼロに設定すると、無制限のメッセージのフェッチが許可されます。
rabbit_retry_backoff = 2	整数値	RabbitMQ に接続する際に再試行間でバックオフする長さ
rabbit_retry_interval = 1	整数値	RabbitMQ との接続を再試行する頻度。
rabbit_transient_queues_ttl = 1800	整数値	キューの TTL (x-expires) の期間 (秒単位) を表す正の整数。TTL の期間使用されないキューは自動的に削除されます。このパラメーターは応答キューとファンアウトキューにのみ影響します。
ssl = False	ブール値	SSL 経由で接続します。
`ssl_ca_file = `	文字列値	SSL 認証局ファイル (SSL が有効な場合にのみ有効)。

設定オプション = デフォルト値	型	説明
`ssl_cert_file = `	文字列値	SSL 証明書ファイル (SSL が有効な場合にのみ有効)。
`ssl_key_file = `	文字列値	SSL キーファイル (SSL が有効な場合にのみ有効)。
`ssl_version = `	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

5.1.38. oslo_middleware

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[oslo_middleware]** グループで使用できるオプションの概要をまとめたものです。

表5.37 oslo_middleware

設定オプション = デフォルト値	型	説明
enable_proxy_headers_parsing = False	ブール値	アプリケーションがプロキシーの背後であるかどうか。これにより、ミドルウェアがヘッダーを解析すべきかどうかを決定します。

5.1.39. oslo_policy

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[oslo_policy]** グループで使用できるオプションの概要をまとめたものです。

表5.38 oslo_policy

設定オプション = デフォルト値	型	説明
enforce_new_defaults = False	ブール値	このオプションは、ポリシーを評価するときに古い非推奨のデフォルトを使用するかどうかを制御します。**True** の場合、古い非推奨のデフォルトは評価されません。これは、既存のトークンが古いデフォルトで許可されているが、新しいデフォルトで許可されていない場合、許可されないことを意味します。新しいデフォルトと **scope_type** の利点を一緒に得ることができるように、このフラグを **enforce_scope** フラグと一緒に有効にすることを推奨します

設定オプション = デフォルト値	型	説明
enforce_scope = False	ブール値	このオプションは、ポリシーを評価する際にスコープを適用するかどうかを制御します。**True** の場合、リクエストで使用されるトークンのスコープは、適用されるポリシーの **scope_types** と比較されます。スコープが一致しない場合、**InvalidScope** 例外が発生します。**False** の場合、ポリシーが一致しないスコープで呼び出されていることを運用者に通知するメッセージが記録されます。
policy_default_rule = default	文字列値	デフォルトルール。要求されたルールが見つからない場合に適用されます。
policy_dirs = ['policy.d']	多値	ポリシー設定ファイルが保存されるディレクトリー。これは、config_dir オプションで定義される検索パス内の任意のディレクトリーへの相対パスか、絶対パスにすることができます。このディレクトリーの検索には、policy_file で定義されたファイルが存在している必要があります。存在しないディレクトリーまたは空のディレクトリーは無視されます。
policy_file = policy.json	文字列値	ロールを指定のサービスのパーミッションにマッピングするファイルの相対パスまたは絶対パス。相対パスは、このオプションを設定する設定ファイルに対する相対パスとして指定する必要があります。
remote_content_type = application/x-www-form-urlencoded	文字列値	REST ベースのポリシーチェックのデータを送受信するコンテンツ種別
remote_ssl_ca_crt_file = None	文字列値	REST ベースのポリシーチェック用の認証局証明書ファイルの絶対パス
remote_ssl_client_crt_file = None	文字列値	REST ベースのポリシーチェック用のクライアント証明書の絶対パス
remote_ssl_client_key_file = None	文字列値	REST ベースのポリシーチェック用のクライアント鍵ファイルの絶対パス
remote_ssl_verify_server_crt = False	ブール値	REST ベースのポリシーチェックのサーバーアイデンティティー検証

5.1.40. oslo_reports

次の表は、**/etc/ironic/ironic.conf** ファイルの **[oslo_reports]** グループで使用できるオプションの概要を示しています。

表5.39 oslo_reports

設定オプション = デフォルト値	型	説明
file_event_handler = None	文字列値	シグナルではなくレポートをトリガーする変更を監視するファイルへのパス。このオプションを設定すると、レポートのシグナルトリガーが無効になります。アプリケーションが WSGI アプリケーションとして実行されている場合は、シグナルの代わりにこれを使用することが推奨されます。
file_event_handler_interval = 1	整数値	file_event_handler が設定されている場合にポーリングの間隔 (秒単位)
log_dir = None	文字列値	ファイルを作成するログディレクトリーへのパス

5.1.41. profiler

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[profiler]** グループで使用できるオプションの概要をまとめたものです。

表5.40 profiler

設定オプション = デフォルト値	型	説明
connection_string = messaging://	文字列値	通知バックエンドの接続文字列。 デフォルト値は **messaging://** で、通知機能を oslo_messaging に設定します。 使用できる値の例: ● **messaging://**: スパンの送信に oslo_messaging ドライバーを使用します。 ● **redis://127.0.0.1:6379**: スパンの送信に redis ドライバーを使用します。 ● **mongodb://127.0.0.1:27017**: スパンの送信に mongodb ドライバーを使用します。 ● **elasticsearch://127.0.0.1:9200:** elasticsearch ドライバーを使用してスパンを送信します。 ● **jaeger://127.0.0.1:6831**: スパンの送信に jaeger トレースを使用します。

設定オプション = デフォルト値	型	説明
enabled = False	ブール値	このノード上の全サービスのプロファイリングを有効にします。 デフォルト値は False です (プロファイリング機能を無効にします)。 値: ● True: 機能を有効にします ● False: 機能を無効にします。このプロジェクトの操作ではプロファイリングを開始できません。プロファイリングが別のプロジェクトでトリガーされた場合には、このプロジェクトの部分は空になります。
es_doc_type = notification	文字列値	elasticsearch の通知インデックスについてのドキュメントタイプ。
es_scroll_size = 10000	整数値	elasticsearch は大規模な要求をバッチに分割します。このパラメーターは、各バッチの最大サイズを定義します (例: es_scroll_size=10000)。
es_scroll_time = 2m	文字列値	このパラメーターは時間値パラメーター (es_scroll_time=2m など) です。これは、検索に参加するノードが継続してサポートするために関連するリソースを維持する時間を示します。
filter_error_trace = False	ブール値	エラー/例外を含むフィルタートレースを分離された場所へ有効にします。 デフォルト値は False に設定されます。 値: ● True: エラー/例外を含むフィルタートレースを有効にします。 ● False: フィルターを無効にします。

設定オプション = デフォルト値	型	説明
hmac_keys = SECRET_KEY	文字列値	パフォーマンスプロファイリング用にコンテキストデータの暗号化に使用するシークレットキー。 この文字列値は <key1>[,<key2>,...<keyn>] の形式にする必要があります。各キーはランダムな文字列です。REST API 経由でプロファイリングをトリガーするユーザーは、REST API 呼び出しのヘッダーにこれらのキーの 1 つを設定し、この特定プロジェクトのノードのプロファイリング結果が含まれるようにする必要があります。 プロファイリングを有効にするには、enabled フラグと hmac_keys 設定オプションの両方を設定する必要があります。また、少なくとも 1 つのキーで全サービス間で正しいプロファイリング情報を生成するには、OpenStack プロジェクト間で一貫性を確保する必要があります。これにより、クライアント側からを使用して、すべての可能なリソースからの情報を含むトレースを生成できます。
sentinel_service_name = mymaster	文字列値	Redissentinel はサービス名を使用してマスターの redis サービスを識別します。このパラメーターは名前を定義します (例: **sentinal_service_name=mymaster**)。
socket_timeout = 0.1	浮動小数点の値	Redissentinel は、接続に timeout オプションを提供します。このパラメーターは、タイムアウトを定義します (例: socket_timeout=0.1)。
trace_sqlalchemy = False	ブール値	サービスで SQL 要求のプロファイリングを有効にします。 デフォルト値は False です (SQL 要求はトレースされません)。 値: ● True: SQL 要求のプロファイリングを有効にします。各 SQL クエリーはトレースの一部となり、それにどの程度の時間を費やしたかによって分析できます。 ● False: SQL 要求のプロファイリングを無効にします。費やした時間は、高いレベルの操作でのみ表示されます。単一の SQL クエリーをこのように分析できません。

5.1.42. pxe

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[pxe]** グループで使用できるオプションの概要をまとめたものです。

表5.41 pxe

設定オプション = デフォルト値	型	説明
boot_retry_check_interval = 90	整数値	PXE ブート再試行の定期的なチェックの間隔 (秒単位)。boot_retry_timeout が設定されていない場合は効果はありません。
boot_retry_timeout = None	整数値	PXE ブートを再試行するタイムアウト (秒単位)。[conductor]deploy_callback_timeout 未満にする必要があります。デフォルトでは無効になっています。
default_ephemeral_format = ext4	文字列値	一時パーティションのデフォルトファイルシステム形式 (作成されている場合)。
dir_permission = None	整数値	作成時に TFTP フォルダーに適用されるパーミッション。これは、tftpserver が、設定した TFTP フォルダーのコンテンツを読み取るパーミッションに設定する必要があります。この設定は、オペレーティングシステムの umask が制限的であるため、ironic-conductor が TFTP サーバーで読み取ることができないファイルを作成する場合にのみ必要です。<None> に設定すると、オペレーティングシステムの umask が新しい tftp フォルダーの作成に使用されます。8 進数表現を指定することが推奨されます。例：0o755
enable_netboot_fallback = False	ブール値	True の場合、ローカルブートを使用するノードに対しても PXE 環境を生成します。これは、ドライバーがノードをローカルブートに切り替えることができない場合に役立ちます。たとえば、永続的なブートを実行できないマシンで SNMP や Redfish を使用する場合などです。Neutron は誤った PXE ブートを防止するため、スタンドアロンの ironic に最も役立ちます。
image_cache_size = 20480	整数値	マスターイメージの最大キャッシュサイズ (MiB 単位)。
image_cache_ttl = 10080	整数値	古いマスターイメージのキャッシュの最大 TTL (分単位)。
images_path = /var/lib/ironic/images/	文字列値	ironic-conductor ノードで、イメージがディスクに保存されるディレクトリーです。

設定オプション = デフォルト値	型	説明
instance_master_path = /var/lib/ironic/master_images	文字列値	ironic-conductor ノードで、マスターインスタンスのイメージがディスクに保存されるディレクトリーです。空の文字列に設定すると、イメージのキャッシュが無効になります。
ip_version = 4	文字列値	PXE ブートに使用する IP バージョン。デフォルトは 4 です。実験的
ipxe_boot_script = $pybasedir/drivers/modules/boot.ipxe	文字列値	ironic-conductor ノードで、メイン iPXE スクリプトファイルへのパス
ipxe_bootfile_name = undionly.kpxe	文字列値	bootFile DHCP パラメーター。
ipxe_bootfile_name_by_arch = {}	dict 値	ノードのアーキテクチャーごとに bootFile DHCP パラメーター。例: aarch64:ipxe_aa64.efi
ipxe_config_template = $pybasedir/drivers/modules/ipxe_config.template	文字列値	ironic-conductor ノードで、iPXE 操作用のテンプレートファイル。
ipxe_timeout = 0	整数値	iPXE 経由でイメージをダウンロードするタイムアウト値 (秒単位)。デフォルトは 0 (タイムアウトなし) です。
ipxe_use_swift = False	ブール値	一時 URL を使用して swift から deploy および rescue イメージを直接ダウンロードします。false (デフォルト) に設定すると、イメージは ironic-conductor ノードにダウンロードされ、ローカルの HTTP サーバー上で提供されます。**ipxe** 互換のブートインターフェイスが使用されている場合にのみ適用されます。
pxe_append_params = nofb nomodeset vga=normal	文字列値	ベアメタル PXE ブートの追加の追加パラメーター。
pxe_bootfile_name = pxelinux.0	文字列値	bootFile DHCP パラメーター。
pxe_bootfile_name_by_arch = {}	dict 値	ノードのアーキテクチャーごとに bootFile DHCP パラメーター。例: aarch64:grubaa64.efi
pxe_config_subdir = pxelinux.cfg	文字列値	ノード上のポートの MAC または IP アドレスを表すシンボリックリンクを作成し、ブートローダーがノードの PXE ファイルをロードできるようにするディレクトリー。このディレクトリー名は、PXE または iPXE フォルダーに対して相対的になります。

設定オプション = デフォルト値	型	説明
pxe_config_template = $pybasedir/drivers/modules/pxe_config.template	文字列値	ironic-conductor ノードで、PXE ローダー設定用のテンプレートファイル。
pxe_config_template_by_arch = {}	dict 値	ironic-conductor ノードでは、ノードアーキテクチャーごとの PXE 設定用のテンプレートファイル。例: aarch64:/opt/share/grubaa64_pxe_config.template
tftp_master_path = /tftpboot/master_images	文字列値	ironic-conductor ノードでは、マスター TFTP イメージがディスクに保存されるディレクトリーです。空の文字列に設定すると、イメージのキャッシュが無効になります。
tftp_root = /tftpboot	文字列値	ironic-conductor ノードの TFTP ルートパスironic-conductor には、このパスに読み取り/書き込みアクセスが必要です。
tftp_server = $my_ip	文字列値	ironic-conductor ノードの TFTP サーバーの IP アドレス
uefi_ipxe_bootfile_name = ipxe.efi	文字列値	UEFI ブートモードの bootFile DHCP パラメーター。それを使用して起動する際に問題が発生した場合は、snponly.efi を試してください。
uefi_pxe_bootfile_name = bootx64.efi	文字列値	UEFI ブートモードの bootFile DHCP パラメーター。
uefi_pxe_config_template = $pybasedir/drivers/modules/pxe_grub_config.template	文字列値	ironic-conductor ノードでは、UEFI ブートローダーの PXE 設定用のテンプレートファイル。通常、これは GRUB 固有のテンプレートに使用されます。

5.1.43. Redfish

次の表は、**/etc/ironic/ironic.conf** ファイルの **[redfish]** グループで使用できるオプションの概要を示しています。

表5.42 Redfish

設定オプション = デフォルト値	型	説明
auth_type = auto	文字列値	Redfish HTTP クライアント認証方法。
connection_attempts = 5	整数値	Redfish への接続試行の最大試行回数

設定オプション = デフォルト値	型	説明
connection_cache_size = 1000	整数値	最大 Redfish クライアント接続キャッシュサイズ。Redfish ドライバーは、認証された BMC 接続 (Redfish セッションサービスを介して取得) を再利用しようとします。このオプションは、維持する接続の最大数を制限します。値 **0** は、クライアント接続キャッシュを完全に無効にします。
connection_retry_interval = 4	整数値	Redfish への接続試行間の待機秒数
file_permission = 420	整数値	ファイルアクセス許可の 8 進数の許可表現を使用した、Swift レスイメージホスティングのファイル許可。この設定のデフォルトは **644**、または Python では 8 進数 **0o644** です。この設定は、**0o** で始まることを意味する 8 進数表現に設定する必要があります。
firmware_update_fail_interval = 60	整数値	失敗したファームウェア更新タスクのチェックの間に待機する秒数
firmware_update_status_interval = 60	整数値	完了したファームウェア更新タスクを確認するまで待機する秒数
kernel_append_params = nofb nomodeset vga=normal	文字列値	インスタンスカーネルに渡す追加のカーネルパラメーター。これらのパラメーターは、/proc/cmdline を読み取ることにより、カーネルまたはアプリケーションで使用できます。厳しいコマンドラインのサイズ制限に注意してください。**instance_info/kernel_append_params** プロパティーでオーバーライドできます。
raid_config_fail_interval = 60	整数値	失敗した RAID 設定タスクのチェックの間に待機する秒数
raid_config_status_interval = 60	整数値	完了した RAID 設定タスクを確認するまで待機する秒数
swift_container = ironic_redfish_container	文字列値	Redfish ドライバーデータを格納する Swift コンテナー。**use_swift** が有効な場合にのみ適用されます。
swift_object_expiry_timeout = 900	整数値	Swift オブジェクトが自動的に期限切れになるまでの時間 (秒単位)。**use_swift** が有効な場合にのみ適用されます。

設定オプション = デフォルト値	型	説明
use_swift = True	ブール値	仮想メディアブート用に生成された ISO イメージを Swift にアップロードし、ノードをブートするために一時 URL を BMC に渡します。false に設定すると、イメージは ironic-conductor ノードに配置され、そのローカル HTTP サーバーを介して提供されます。

5.1.44. service_catalog

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[service_catalog]** グループで使用できるオプションの概要をまとめたものです。

表5.43 service_catalog

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID

設定オプション = デフォルト値	型	説明
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
max-version = None	文字列値	特定の API の最大メジャーバージョンで、min_version を持つ範囲の上限として使用されることが意図されています。version と相互に排他的です。
min-version = None	文字列値	指定の API の最小メジャーバージョンで、max_version を持つ範囲の下限として使用することが意図されています。version と相互に排他的です。max_version なしで min_version が指定されている場合は、max version が latest の場合になります。
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = baremetal	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。

設定オプション = デフォルト値	型	説明
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。
version = None	文字列値	エンドポイント URL 検出用に指定の Major API バージョン内の最小の Major API バージョン。min_version および max_version と相互排他的になる

5.1.45. snmp

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[snmp]** グループで使用できるオプションの概要をまとめたものです。

表5.44 snmp

設定オプション = デフォルト値	型	説明
power_timeout = 10	整数値	電源アクションが完了するまで待機する秒数
reboot_delay = 0	整数値	再起動時 (電源オフと再電源) の間にスリープ状態になる時間 (秒単位)
udp_transport_retries = 5	整数値	UDP リクエスト再試行の最大数。0 は再試行なしを意味します。
udp_transport_timeout = 1.0	浮動小数点の値	UDP トランスポートに使用されるレスポンスのタイムアウト (秒単位)。タイムアウトは 0.5 秒の倍数で、再試行ごとに適用する必要があります。

5.1.46. ssl

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[ssl]** グループで使用できるオプションの概要をまとめたものです。

表5.45 ssl

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	接続クライアントの検証に使用する CA 証明書ファイル。
cert_file = None	文字列値	サーバーをセキュアに起動する際に使用する証明書ファイル。
ciphers = None	文字列値	利用可能な暗号のリストを設定します。値は OpenSSL 暗号リスト形式の文字列である必要があります。
key_file = None	文字列値	サーバーをセキュアに起動する際に使用する秘密鍵ファイル。
version = None	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

5.1.47. swift

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[swift]** グループで使用できるオプションの概要をまとめたものです。

表5.46 swift

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル

設定オプション = デフォルト値	型	説明
max-version = None	文字列値	特定の API の最大メジャーバージョンで、min_version を持つ範囲の上限として使用されることが意図されています。version と相互に排他的です。
min-version = None	文字列値	指定の API の最小メジャーバージョンで、max_version を持つ範囲の下限として使用することが意図されています。version と相互に排他的です。max_version なしで min_version が指定されている場合は、max version が latest の場合になります。
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = object-store	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
swift_max_retries = 2	整数値	失敗する前に Swift 要求を再試行する最大回数。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID

設定オプション = デフォルト値	型	説明
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。
version = None	文字列値	エンドポイント URL 検出用に指定の Major API バージョン内の最小の Major API バージョン。 min_version および max_version と相互排他的になる

5.1.48. xclarity

以下の表は、**/etc/ironic/ironic.conf** ファイルの **[xclarity]** グループで使用できるオプションの概要をまとめたものです。

表5.47 xclarity

設定オプション = デフォルト値	型	説明
manager_ip = None	文字列値	XClarity コントローラーの IP アドレスここでの設定は非推奨となり、Stein リリースで削除されます。driver_info フィールドを、代わりに "xclarity_manager_ip" を使用するように更新してください。
password = None	文字列値	XClarity Controller ユーザー名のパスワードここでの設定は非推奨となり、Stein リリースで削除されます。driver_info フィールドを、代わりに "xclarity_password" を使用するように更新してください。
port = 443	ポート値	XClarity コントローラー接続に使用するポート。

設定オプション = デフォルト値	型	説明
username = None	文字列値	XClarity Controller のユーザー名。ここでの設定は非推奨となり、Stein リリースで削除されます。driver_info フィールドを、代わりに "xclarity_username" を使用するように更新してください。

第6章 IRONIC-INSPECTOR

以下の章では、**ironic-inspector** サービスの設定オプションについて説明します。

6.1. INSPECTOR.CONF

本項では、**/etc/ironic-inspector/inspector.conf** ファイルのオプションについて説明します。

6.1.1. DEFAULT

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
api_max_limit = 1000	整数値	API list-call が返す要素の数を制限します。
auth_strategy = keystone	文字列値	ironic-inspector API で使用される認証方法。noauth、keystone、または http_basic が有効なオプションです。noauth はすべての認証を無効にします。
can_manage_boot = True	ブール値	ironic-inspector の現在のインストールがノードの PXE ブートを管理できるかどうか。False に設定すると、API は manage_boot のない または True に設定されたイントロスペクション要求を拒否します。
clean_up_period = 60	整数値	タイムアウトしたノードと古いノードのステータス情報を繰り返すまでの時間 (秒単位)。警告: 値を 0 に設定すると、定期的なタスクが無効になり、インスペクターは ironic と同期して内部クリーンアッププロセスを完了しません。デプロイで PXE フィルターが使用されている場合は推奨できません。その結果、ironic-inspector が定期的なクリーンアップアクティビティーを停止することになります。
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。

設定オプション = デフォルト値	型	説明
default_log_levels = ['sqlalchemy=WARNING', 'iso8601=WARNING', 'requests=WARNING', 'urllib3.connectionpool=WARNING', 'keystonemiddleware=WARNING', 'keystoneauth=WARNING', 'ironicclient=WARNING', 'amqp=WARNING', 'amqplib=WARNING', 'oslo.messaging=WARNING', 'oslo_messaging=WARNING']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
enable_mdns = False	ブール値	マルチキャスト DNS 経由で ironic-inspector API エンドポイントの公開を有効にするかどうかを定義します。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
host = <based on operating system>	文字列値	このノードの名前。これは不透明な ID にすることができます。必ずしもホスト名、FQDN、または IP アドレスではありません。ただし、ノード名は AMQP キー内で有効で、ZeroMQ を使用している場合は、有効なホスト名、FQDN、または IP アドレスを使用する必要があります。
http_basic_auth_user_file = /etc/ironic-inspector/htpasswd	文字列値	auth_strategy=http_basic の場合に使用される Apache 形式のユーザー認証ファイルへのパス
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。
introspection_delay = 5	整数値	2 つのイントロスペクション間の遅延 (秒単位)。これは、ブートが ironic-inspector (つまり manage_boot==True) で管理される場合にのみ適用されます。

設定オプション = デフォルト値	型	説明
ipmi_address_fields = ['redfish_address', 'ilo_address', 'drac_host', 'drac_address', 'ibmc_address']	リスト値	ipmi_address と同等な ironic driver_info フィールド。
leader_election_interval = 10	整数値	リーダー選択の間隔 (秒単位)。
listen_address = ::	文字列値	リッスンする IP。
listen_port = 5050	ポート値	リッスンするポート。
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。

設定オプション = デフォルト値	型	説明
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_concurrency = 1000	整数値	グリーンスレッドプールのサイズ。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。

設定オプション = デフォルト値	型	説明
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
rootwrap_config = /etc/ironic-inspector/rootwrap.conf	文字列値	root としてコマンドを実行するために使用する rootwrap 設定ファイルへのパス
standalone = True	ブール値	ironic-inspector をスタンドアロンのサービスとして実行するかどうか。False に設定されているのは EXPERIMENTAL です。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
timeout = 3600	整数値	イントロスペクションが失敗したと見なされるタイムアウトで、無効にするには 0 に設定します。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_ssl = False	ブール値	SSL の有効化/無効化

設定オプション = デフォルト値	型	説明
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

6.1.2. capabilities

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[capabilities]** グループで使用できるオプションの概要をまとめたものです。

表6.1 capabilities

設定オプション = デフォルト値	型	説明
boot_mode = False	ブール値	ブートモード (BIOS または UEFI) を保存するかどうか。
cpu_flags = {'aes': 'cpu_aes', 'pdpe1gb': 'cpu_hugepages_1g', 'pse': 'cpu_hugepages', 'smx': 'cpu_txt', 'svm': 'cpu_vt', 'vmx': 'cpu_vt'}	dict 値	このフラグが存在する場合に設定する CPU フラグと機能間のマッピング。

6.1.3. coordination

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[coordination]** グループで使用できるオプションの概要をまとめたものです。

表6.2 coordination

設定オプション = デフォルト値	型	説明
backend_url = memcached://localhost:11211	文字列値	分散調整に使用するバックエンド URL。実験的

6.1.4. cors

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[cors]** グループで使用できるオプションの概要をまとめたものです。

表6.3 cors

設定オプション = デフォルト値	型	説明
allow_credentials = True	ブール値	実際の要求にユーザーの認証情報を含めることができることを示します。
allow_headers = ['X-Auth-Token', 'X-OpenStack-Ironic-Inspector-API-Minimum-Version', 'X-OpenStack-Ironic-Inspector-API-Maximum-Version', 'X-OpenStack-Ironic-Inspector-API-Version']	リスト値	実際の要求時に使用されるヘッダーフィールド名を示します。
allow_methods = ['GET', 'POST', 'PUT', 'HEAD', 'PATCH', 'DELETE', 'OPTIONS']	リスト値	実際の要求時に使用できるメソッドを示します。
allowed_origin = None	リスト値	このリソースがリクエストの origin"ヘッダーで受信したドメインと共有されるかどうかを示します。形式:"<protocol>://<host>[:<port>]" (行末のスラッシュなし)例:https://horizon.example.com
expose_headers = []	リスト値	API に安全に公開できるヘッダーを示します。デフォルトは HTTP Simple ヘッダーです。
max_age = 3600	整数値	CORS プリフライトリクエストの最大キャッシュ期間。

6.1.5. database

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[database]** グループで使用できるオプションの概要をまとめたものです。

表6.4 database

設定オプション = デフォルト値	型	説明
backend = sqlalchemy	文字列値	データベースに使用するバックエンド。

設定オプション = デフォルト値	型	説明
connection = None	文字列値	データベースへの接続に使用する SQLAlchemy 接続文字列。
connection_debug = 0	整数値	SQL デバッグ情報の冗長性: 0=なし、100=すべて。
`connection_parameters = `	文字列値	接続時に接続 URL に追加するオプションの URL パラメーター。param1=value1&param2=value2&... として指定します。
connection_recycle_time = 3600	整数値	この秒数より長く接続プールに存在していた接続は、次にプールからチェックアウトされたときに新しい接続に置き換えられます。
connection_trace = False	ブール値	Python スタックトレースをコメント文字列として SQL に追加します。
db_inc_retry_interval = True	ブール値	True の場合には、db_max_retry_interval までのデータベース操作の再試行回数を長くします。
db_max_retries = 20	整数値	エラーが発生する前に接続エラーまたはデッドロックが発生した場合の最大再試行。再試行回数を無限に指定するには -1 に設定します。
db_max_retry_interval = 10	整数値	db_inc_retry_interval が設定されている場合は、データベース操作を再試行するまでの最大秒数です。
db_retry_interval = 1	整数値	データベーストランザクションの再試行間の秒数。
max_overflow = 50	整数値	設定されている場合、この値を SQLAlchemy で max_overflow に使用します。
max_pool_size = 5	整数値	プールに開いたままにする SQL 接続の最大数。値を 0 に設定すると無制限を意味します。
max_retries = 10	整数値	起動時のデータベース接続の最大再試行数。再試行回数を無限に指定するには -1 に設定します。
mysql_enable_ndb = False	ブール値	True の場合には、MySQL Cluster (NDB) の処理のサポートを透過的に有効にします。
mysql_sql_mode = TRADITIONAL	文字列値	MySQL セッションに使用する SQL モード。このオプション (デフォルトを含む) は、サーバーセット SQL モードを上書きします。サーバー設定で設定された SQL モードを使用するには、これを no value に設定します。例: mysql_sql_mode=

設定オプション = デフォルト値	型	説明
pool_timeout = None	整数値	設定されている場合は、この値を SQLAlchemy で pool_timeout に使用します。
retry_interval = 10	整数値	SQL 接続を開く再試行の間隔。
slave_connection = None	文字列値	スレーブデータベースへの接続に使用する SQLAlchemy 接続文字列。
sqlite_synchronous = True	ブール値	True の場合、SQLite は同期モードを使用します。
use_db_reconnect = False	ブール値	失われた接続でデータベースの実験的な使用を有効にします。

6.1.6. discovery

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[discovery]** グループで使用できるオプションの概要をまとめたものです。

表6.5 discovery

設定オプション = デフォルト値	型	説明
enabled_bmc_address_version = ['4', '6']	リスト値	新規ノードを Ironic に登録する際に使用される BMC アドレスの IP バージョン。デフォルトは 4,6 です。4(v4 アドレスのみ使用)、4,6(v4 アドレスがより優先され、両方のアドレスが見つかった場合 v6 バージョンは無視される)、6,4(v6 が望ましいが v4 アドレスを持つ BMC のために v4 アドレスにフォールバック、4,6 の反対)、6(v6 アドレスのみ使用し v4 バージョンを無視) が可能です。
enroll_node_driver = fake-hardware	文字列値	Ironic に新規ノードを作成する際に enroll フックで使用される Ironic ドライバーの名前。
enroll_node_fields = {}	dict 値	新しく検出されたノードに設定する追加フィールド。

6.1.7. dnsmasq_pxe_filter

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[dnsmasq_pxe_filter]** グループで使用できるオプションの概要をまとめたものです。

表6.6 dnsmasq_pxe_filter

設定オプション = デフォルト値	型	説明
dhcp_hostsdir = /var/lib/ironic-inspector/dhcp-hostsdir	文字列値	dnsmasq に公開されている MAC アドレスキャッシュディレクトリー。このディレクトリーはドライバーを排他的に制御する必要があります。
`dnsmasq_start_command = `	文字列値	フィルターの初期化時に dnsmasq サービスを開始する (シェル) コマンドライン。デフォルト: don't start
`dnsmasq_stop_command = `	文字列値	インスペクター (エラー) 終了時に dnsmasq サービスを停止する (シェル) コマンドライン。デフォルト: don't stop
purge_dhcp_hostsdir = True	ブール値	ドライバーの初期化時に hostsdir をパージします。false に設定すると、インスペクターのデプロイメントが、同じホストおよび名前空間内で複数のプロセスが実行される可能性がある場合にのみ実行する必要があります。この場合、Operator はカスタムクリーニング機能を設定します。

6.1.8. extra_hardware

次の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[extra_hardware]** グループで使用できるオプションの概要を示しています。

表6.7 extra_hardware

設定オプション = デフォルト値	型	説明
strict = False	ブール値	True の場合、少なくとも 1 つのレコードが短すぎる場合、余分なデータの解析を拒否します。さらに、解析に失敗した場合でも、受信したデータを削除します。

6.1.9. iptables

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[iptables]** グループで使用できるオプションの概要をまとめたものです。

表6.8 iptables

設定オプション = デフォルト値	型	説明
dnsmasq_interface = br-ctlplane	文字列値	dnsmasq がリッスンするインターフェイスです。デフォルトは仮想マシンの です。

設定オプション = デフォルト値	型	説明
ethoib_interfaces = []	リスト値	DHCP ネットワークへの物理アクセスに使用される Inspector ホスト上の Ethernet Over InfiniBand インターフェイスのリスト。複数のインターフェイスが dnsmasq_interface に指定されたボンディングまたはブリッジにアタッチされます。望ましい状態にない InfiniBand ノードの MAC は、これらのインターフェイスの隣接する MAC のリストに基づいてブラックリスト指定されます。
firewall_chain = ironic-inspector	文字列値	使用する iptables チェーン名。
ip_version = 4	文字列値	iptables フィルターに使用される IP バージョン。デフォルトは 4 です。

6.1.10. ironic

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[ironic]** グループで使用できるオプションの概要をまとめたものです。

表6.9 ironic

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。

設定オプション = デフォルト値	型	説明
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
max-version = None	文字列値	特定の API の最大メジャーバージョンで、min_version を持つ範囲の上限として使用されることが意図されています。version と相互に排他的です。
max_retries = 30	整数値	競合エラー時の最大再試行回数 (HTTP 409)。
min-version = None	文字列値	指定の API の最小メジャーバージョンで、max_version を持つ範囲の下限として使用することが意図されています。version と相互に排他的です。max_version なしで min_version が指定されている場合は、max version が latest の場合になります。
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID

設定オプション = デフォルト値	型	説明
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
retry_interval = 2	整数値	競合エラーが発生した場合の再試行の間隔 (HTTP 409)。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = baremetal	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID

設定オプション = デフォルト値	型	説明
username = None	文字列値	ユーザー名
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。
version = None	文字列値	エンドポイント URL 検出用に指定の Major API バージョン内の最小の Major API バージョン。 min_version および max_version と相互排他的になる

6.1.11. keystone_authtoken

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[keystone_authtoken]** グループで使用できるオプションの概要をまとめたものです。

表6.10 keystone_authtoken

設定オプション = デフォルト値	型	説明
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ
auth_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。このオプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。**非推奨:**Queens 以降 *理由:* auth_uri オプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。
auth_version = None	文字列値	Identity API エンドポイントの API バージョン。

設定オプション = デフォルト値	型	説明
cache = None	文字列値	Swift キャッシュオブジェクトが保存される環境キーを要求します。auth_token ミドルウェアを Swift キャッシュと共にデプロイする場合は、このオプションを使用して、ミドルウェアが swift とキャッシングバックエンドを共有するようにします。それ以外の場合は、代わりに **memcached_servers** オプションを使用します。
cafile = None	文字列値	HTTPs 接続の検証時に使用する PEM でエンコードされた認証局。デフォルトはシステム CA です。
certfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
delay_auth_decision = False	ブール値	ミドルウェア内の認可要求を処理せず、承認の決定をダウンストリームの WSGI コンポーネントに委譲します。
enforce_token_bind = permissive	文字列値	トークンバインディングの使用および種別を制御するために使用されます。トークンバインディングのチェックを無効するには、"disabled"に設定します。バインドタイプがサーバーの認識する形式の場合にはバインディング情報を検証し、そうでない場合には無視するには、"permissive"(デフォルト) に設定します。"strict"は"permissive"と類似していますが、バインドタイプが不明な場合にはトークンが拒否されます。"required"の場合は、いずれかの形式のトークンバインディングが必要です。最後に、トークンに指定する必要のあるバインディングメソッドの名前。
http_connect_timeout = None	整数値	Identity API サーバーと通信する際の要求タイムアウト値。
http_request_max_retries = 3	整数値	Identity API サーバーと通信する際に再接続を試行する回数。
include_service_catalog = True	ブール値	(オプション)X-Service-Catalog ヘッダーを設定するかどうかを示します。False の場合、ミドルウェアはトークンの検証時にサービスカタログを要求せず、X-Service-Catalog ヘッダーを設定しません。
insecure = False	ブール値	HTTPS 接続を確認します。

設定オプション = デフォルト値	型	説明
interface = internal	文字列値	Identity API エンドポイントに使用するインターフェイス。有効な値は、"public"、"internal" (デフォルト)、または "admin" です。
keyfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
memcache_pool_conn_get_timeout = 10	整数値	(オプション) プールから memcached クライアント接続を取得するまで操作が待機する秒数。
memcache_pool_dead_retry = 300	整数値	(オプション)memcached サーバーが停止しているとみなされる秒数。この秒数が経過すると再試行されます。
memcache_pool_maxsize = 10	整数値	(オプション) すべての memcached サーバーへのオープン接続の最大合計数。
memcache_pool_socket_timeout = 3	整数値	(オプション)memcached サーバーと通信する際のソケットのタイムアウト (秒単位)。
memcache_pool_unused_timeout = 60	整数値	(オプション)memcached への接続がプール内で未使用の状態を維持する秒数。この秒数が経過すると終了されます。
memcache_secret_key = None	文字列値	(オプション、memcache_security_strategy が定義されている場合には必須) この文字列は鍵の導出に使用されます。
memcache_security_strategy = None	文字列値	(オプション) 定義されている場合は、トークンデータを認証、または認証して暗号化する必要があるかどうかを示します。MAC の場合、キャッシュでトークンデータが認証されます (HMAC を使用)。ENCRYPT の場合、キャッシュでトークンデータが暗号化され、認証されます。値がこれらのオプションのいずれでもない場合や空の場合には、auth_token は初期化時に例外を発生させます。
memcache_use_advanced_pool = False	ブール値	(オプション) 高度な (eventlet に対して安全な) memcached クライアントプールを使用します。高度なプールは python 2.x でのみ動作します。
memcached_servers = None	リスト値	オプションで、キャッシュに使用する memcached サーバーのリストを指定します。未定義のままの場合、トークンは代わりに処理中にキャッシュされます。
region_name = None	文字列値	アイデンティティーサーバーがあるリージョン。

設定オプション = デフォルト値	型	説明
service_token_roles = ['service']	リスト値	サービストークンに存在する必要があるロールの選択。サービストークンは、期限切れのトークンを使用できることを要求できるため、このチェックでは実際のサービスのみがこのトークンを送信するように厳密に制御する必要があります。ここでのロールは ANY チェックとして適用されるため、このリストのロールはすべて存在している必要があります。後方互換性の理由から、現在 allow_expired チェックにのみ影響します。
service_token_roles_required = False	ブール値	後方互換性の理由から、service_token_roles チェックを有効としてパスしない有効なサービストークンをパスさせる必要があります。これを true に設定することが今後のリリースでデフォルトとなり、可能な場合は有効にされる必要があります。
service_type = None	文字列値	サービスカタログに表示されるサービスの名前または種別。これは、制限されたアクセスルールがあるトークンを検証するために使用されます。
token_cache_time = 300	整数値	トークンの検証に過剰な時間を費やすのを防ぐために、ミドルウェアは、設定可能な期間 (秒単位) 中は以前に見たトークンをキャシュします。キャッシュを完全に無効にするには -1 に設定します。
www_authenticate_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。

6.1.12. oslo_policy

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[oslo_policy]** グループで使用できるオプションの概要をまとめたものです。

表6.11 oslo_policy

設定オプション = デフォルト値	型	説明
enforce_new_defaults = False	ブール値	このオプションは、ポリシーを評価するときに古い非推奨のデフォルトを使用するかどうかを制御します。**True** の場合、古い非推奨のデフォルトは評価されません。これは、既存のトークンが古いデフォルトで許可されているが、新しいデフォルトで許可されていない場合、許可されないことを意味します。新しいデフォルトと **scope_type** の利点を一緒に得ることができるように、このフラグを **enforce_scope** フラグと一緒に有効にすることを推奨します
enforce_scope = False	ブール値	このオプションは、ポリシーを評価する際にスコープを適用するかどうかを制御します。**True** の場合、リクエストで使用されるトークンのスコープは、適用されるポリシーの **scope_types** と比較されます。スコープが一致しない場合、**InvalidScope** 例外が発生します。**False** の場合、ポリシーが一致しないスコープで呼び出されていることを運用者に通知するメッセージが記録されます。
policy_default_rule = default	文字列値	デフォルトルール。要求されたルールが見つからない場合に適用されます。
policy_dirs = ['policy.d']	多値	ポリシー設定ファイルが保存されるディレクトリー。これは、config_dir オプションで定義される検索パス内の任意のディレクトリーへの相対パスか、絶対パスにすることができます。このディレクトリーの検索には、policy_file で定義されたファイルが存在している必要があります。存在しないディレクトリーまたは空のディレクトリーは無視されます。
policy_file = policy.json	文字列値	ロールを指定のサービスのパーミッションにマッピングするファイルの相対パスまたは絶対パス。相対パスは、このオプションを設定する設定ファイルに対する相対パスとして指定する必要があります。
remote_content_type = application/x-www-form-urlencoded	文字列値	REST ベースのポリシーチェックのデータを送受信するコンテンツ種別
remote_ssl_ca_crt_file = None	文字列値	REST ベースのポリシーチェック用の認証局証明書ファイルの絶対パス
remote_ssl_client_crt_file = None	文字列値	REST ベースのポリシーチェック用のクライアント証明書の絶対パス
remote_ssl_client_key_file = None	文字列値	REST ベースのポリシーチェック用のクライアント鍵ファイルの絶対パス

設定オプション = デフォルト値	型	説明
remote_ssl_verify_server_crt = False	ブール値	REST ベースのポリシーチェックのサーバーアイデンティティー検証

6.1.13. pci_devices

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[pci_devices]** グループで使用できるオプションの概要をまとめたものです。

表6.12 pci_devices

設定オプション = デフォルト値	型	説明
alias = []	多値	**vendor_id** フィールドおよび **product_id** フィールドで識別される PCI デバイスのエイリアス。形式: {"vendor_id": "1234", "product_id": "5678", "name": "pci_dev1"}

6.1.14. port_physnet

次の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[port_physnet]** グループで使用できるオプションの概要を示しています。

表6.13 port_physnet

設定オプション = デフォルト値	型	説明
cidr_map = []	リスト値	IP サブネット CIDR から物理ネットワークへのマッピング。physnet_cidr_map 処理フックが有効になっている場合、ベアメタルポートの physical_network プロパティーは、このマッピングに基づいて入力されます。

6.1.15. 処理

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[processing]** グループで使用できるオプションの概要をまとめたものです。

表6.14 処理

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
add_ports = pxe	文字列値	イントロスペクション中にポートとして追加する MAC アドレス。使用できる値: all (すべての MAC アドレス)、active (IP アドレスを使用する NIC の MAC アドレス)、pxe (PXE ブート元である NIC ノードの MAC アドレスのみ)、ramdisk で PXE MAC が指定されていない場合のアクティブにフォールバックします。
always_store_ramdisk_logs = False	ブール値	エラーメッセージが返されない場合でも ramdisk ログを保存するかどうか (ramdisk_logs_dir オプションが設定されているかどうか)。
default_processing_hooks = ramdisk_error,root_disk_selection,scheduler,validate_interfaces,capabilities,pci_devices	文字列値	パイプラインを処理するデフォルトフックのコンマ区切りリスト。フック **スケジューラー** により、Nova スケジューラーが必要とする最低限のプロパティーでノードが更新されます。フック **validate_interfaces** は、ramdisk が有効な NIC データを提供します。実際に何を行うのかがわかっていない限り、この 2 つを除外しないでください。
disk_partitioning_spacing = True	ブール値	パーティショニングのために 1 GiB のディスクサイズを残すかどうか。IPA で ramdisk として使用する場合に限り、以前の ramdisk local_gb が ramdisk 側で算出されます。
keep_ports = all	文字列値	イントロスペクションの後に保持するポート (ノード上に準備済み) します。使用できる値は、all (何も削除しない)、ind (イントロスペクションデータに存在する MAC ポート)、added (イントロスペクション中に追加した MAC のみ) です。
node_not_found_hook = None	文字列値	インスペクターがノードから検査情報を受け取る際に実行されるフックの名前。このフックはデフォルトで無視されます。
overwrite_existing = True	ブール値	ノードデータベースの既存の値を上書きするかどうか。このオプションを無効にして、非破壊的な操作を行います。
permit_active_introspection = False	ブール値	稼働状態のノードを処理するかどうか。
power_off = True	ブール値	アクティブで introspection.Nodes の後にノードの電源をオフにするか、イントロスペクションデータを送信すると、その機能が **permit_active_introspection** 設定オプションで有効になっている場合は、その電源をオフにします。

設定オプション = デフォルト値	型	説明
processing_hooks = $default_processing_hooks	文字列値	パイプラインを処理するための有効なフックのコンマ区切りリスト。このデフォルトは $default_processing_hooks で、"prehook,$default_processing_hooks,posthook" のように、デフォルトの前または後にフックを追加することができます。
ramdisk_logs_dir = None	文字列値	これを設定すると、ramdisk からのログはこのディレクトリーに保存されます。
ramdisk_logs_filename_format = {uuid}_{dt:%Y%m%d-%H%M%S.%f}.tar.gz	文字列値	ramdisk ログを保存するファイル名のテンプレート。{uuid} - node UUID または "unknown", {bmc} - node BMC address or "unknown", {dt} - current UTC date and time, {mac} - PXE boot MAC or "unknown" の代わりとして使用できます。
store_data = none	文字列値	イントロスペクションデータを保存するためのストレージバックエンド。使用できる値は **none**、**database**、**swift** です。**none** に設定すると、イントロスペクションデータは保存されません。
update_pxe_enabled = True	ブール値	イントロスペクションデータに従って pxe_enabled 値を更新するかどうか。[processing]overwrite_existing が False に設定されている場合、このオプションは効果がありません。

6.1.16. pxe_filter

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[pxe_filter]** グループで使用できるオプションの概要をまとめたものです。

表6.15 pxe_filter

設定オプション = デフォルト値	型	説明
deny_unknown_macs = False	ブール値	デフォルトでは、イントロスペクションがアクティブな場合、インスペクターは任意のノードの DHCP サーバーを開きます。イントロスペクションがアクティブなときに未知の MAC アドレスに対して DHCP を開くと、ユーザーはポートのないノードを ironic に追加し、ノードのイントロスペクションの結果に基づいて ironic-inspector にポートを登録させることができます。注記: このオプションが True の場合、ノードにはイントロスペクションの前に少なくとも 1 つの登録済みポートが必要です。

設定オプション = デフォルト値	型	説明
driver = iptables	文字列値	使用する PXE ブートフィルタードライバー、使用可能なフィルターは iptables、dnsmasq、および noop です。noop を設定して、ファイアウォールのフィルタリングを無効にします。
sync_period = 15	整数値	フィルターの定期的な更新を繰り返すまでの時間 (秒単位)。

6.1.17. service_catalog

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[service_catalog]** グループで使用できるオプションの概要をまとめたものです。

表6.16 service_catalog

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。

設定オプション = デフォルト値	型	説明
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
max-version = None	文字列値	特定の API の最大メジャーバージョンで、min_version を持つ範囲の上限として使用されることが意図されています。version と相互に排他的です。
min-version = None	文字列値	指定の API の最小メジャーバージョンで、max_version を持つ範囲の下限として使用することが意図されています。version と相互に排他的です。max_version なしで min_version が指定されている場合は、max version が latest の場合になります。
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。

設定オプション = デフォルト値	型	説明
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = baremetal-introspection	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。
version = None	文字列値	エンドポイント URL 検出用に指定の Major API バージョン内の最小の Major API バージョン。min_version および max_version と相互排他的になる

6.1.18. swift

以下の表は、**/etc/ironic-inspector/inspector.conf** ファイルの **[swift]** グループで使用できるオプションの概要をまとめたものです。

表6.17 swift

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
container = ironic-inspector	文字列値	オブジェクトの作成時に使用するデフォルトの Swift コンテナー。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
delete_after = 0	整数値	Swift オブジェクトが削除されるまでの秒数 (オブジェクトを削除しない場合は 0 に設定します)。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名

設定オプション = デフォルト値	型	説明
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
max-version = None	文字列値	特定の API の最大メジャーバージョンで、min_version を持つ範囲の上限として使用されることが意図されています。version と相互に排他的です。
min-version = None	文字列値	指定の API の最小メジャーバージョンで、max_version を持つ範囲の下限として使用することが意図されています。version と相互に排他的です。max_version なしで min_version が指定されている場合は、max version が latest の場合になります。
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = object-store	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。

設定オプション = デフォルト値	型	説明
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。
version = None	文字列値	エンドポイント URL 検出用に指定の Major API バージョン内の最小の Major API バージョン。min_version および max_version と相互排他的になる

第7章 KEYSTONE

以下の章では、**keystone** サービスの設定オプションについて説明します。

7.1. KEYSTONE.CONF

本項では、**/etc/keystone/keystone.conf** ファイルのオプションについて説明します。

7.1.1. DEFAULT

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
admin_token = None	文字列値	この機能の使用は推奨され**ていません**。代わりに **keystone-manage bootstrap** コマンドを使用します。このオプションの値は、共有シークレットとして扱われ、API 経由で Keystone をブートストラップするために使用できます。このトークンはユーザーを表すものではありません (ID を持たない)。また明示的な認可を伴わないません (ほとんどの認可チェックを事実上回避する)。**None** に設定すると、値は無視され、**admin_token** ミドルウェアは事実上無効になります。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
control_exchange = keystone	文字列値	トピックがスコープ設定されるデフォルトの変換。transport_url オプションで指定した変換名で上書きできます。
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
default_publisher_id = None	文字列値	送信通知のデフォルト **publisher_id**。定義されていない場合は、Keystone はデフォルトでサーバーのホスト名を使用します。
executor_thread_pool_size = 64	整数値	エグゼキューターがスレッディングまたはイベントレットの場合のエグゼキュータースレッドプールのサイズ。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
insecure_debug = False	ブール値	true に設定すると、サーバーは、認証されていない、または認証されたユーザーが通常の情報 (認証が失敗した理由に関する追加情報など) を取得できるように HTTP 応答の情報を返します。これはデバッグに役立ちますが、安全ではありません。

設定オプション = デフォルト値	型	説明
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。
list_limit = None	整数値	コレクションで返されるエンティティーの最大数。次に、適切なセクションに list_limit を指定して、特定のドライバーに対してこのグローバル制限を上書きできます (例: **[assignment]**)。デフォルトでは制限は設定されません。大規模なデプロイメントでは、全ユーザーおよびプロジェクトをリスト表示するような操作がシステムに不必要な負荷が配置されないようにするために、これを妥当な数値に設定することが推奨されます。
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。

設定オプション = デフォルト値	型	説明
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
max_param_size = 64	整数値	ユーザー & プロジェクト ID/名前のサイズを制限します。

設定オプション = デフォルト値	型	説明
max_project_tree_depth = 5	整数値	プロジェクト階層の最大深さ (階層の上部にあるドメインとして機能するプロジェクトを除く)。警告: この値を大きな値に設定すると、パフォーマンスに悪影響を与える可能性があります。
max_token_size = 255	整数値	**[DEFAULT] max_param_size** と同様ですが、トークン値の例外を提供します。Fernet トークンでは、この値を 255 未満に設定できます。
notification_format = cadf	文字列値	Identity サービスイベントの通知形式を定義します。**basic** 通知には、操作するリソースに関する情報のみがあります。**cadf** 通知には、イベントのイニシエーターに関する情報と、同じ情報があります。**cadf** オプションは **basic** オプションと完全に後方互換性がありますが、CADF は完全に準拠しており、監査のユースケースで推奨されます。
notification_opt_out = ['identity.authenticate.success', 'identity.authenticate.pending', 'identity.authenticate.failed']	多値	明示的にオプトアウトすることで、keystone が出力する通知数を減らすことができます。Keystone は、このリストで表現されるパターンに一致する通知を出力しません。値は **identity.<resource_type>.<operation>** の形式でなることが予想されます。デフォルトでは、認証に関連するすべての通知が自動的に抑制されます。複数の通知トピックをオプトアウトするには、このフィールドを複数回設定できます。たとえば、以下は、ユーザー作成または正常な認証イベントを記述する通知を表示しません。 notification_opt_out=identity.user.create notification_opt_out=identity.authenticate.success
public_endpoint = None	URI 値	クライアントに通知される Keystone のベースパブリックエンドポイント URL (Keystone が接続をリッスンする方法には影響しません)。デフォルトは、要求のベースホスト URL です。たとえば、keystone が **http://server:5000/v3/users** へのリクエストを受け取ると、このオプションは自動的に **http://server:5000** として処理されます。オプションを設定する必要があります。これは、ベース URL の値に keystone が自動的に推測されないパス (**/prefix/v3**) が含まれている場合に限ります。または、エンドポイントが別のホストにある必要がある場合に限ります。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。

設定オプション = デフォルト値	型	説明
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
rpc_conn_pool_size = 30	整数値	RPC 接続プールのサイズ。
rpc_ping_enabled = False	ブール値	ping 呼び出しに応答するエンドポイントを追加します。エンドポイントの名前は oslo_rpc_server_ping です。
rpc_response_timeout = 60	整数値	呼び出しからの応答を待つ秒数。
strict_password_check = False	ブール値	true に設定すると、パスワード操作に対して、パスワードの長さの厳密なチェックが実行されます。パスワードの最大長を超えると、操作は HTTP 403 Forbidden エラーを出して失敗します。false に設定すると、パスワードは自動的に最大長に切り捨てられます。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
transport_url = rabbit://	文字列値	メッセージングバックエンドに接続するためのネットワークアドレスおよびオプションのユーザー認証情報 (URL 形式)。想定される形式は次のとおりです。 driver://[user:pass@]host:port[, [userN:passN@]hostN:portN]/virtual_host?query 例:rabbit://rabbitmq:password@127.0.0.1:5672// URL のフィールドの詳細は、https://docs.openstack.org/oslo.messaging/latest/reference/transport.html で oslo_messaging.TransportURL のドキュメントを参照してください。

設定オプション = デフォルト値	型	説明
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

7.1.2. application_credential

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[application_credential]** グループで使用できるオプションの概要をまとめたものです。

表7.1 application_credential

設定オプション = デフォルト値	型	説明
cache_time = None	整数値	アプリケーション認証情報データを秒単位でキャッシュする時間。これは、グローバルキャッシングが有効でない限り効果はありません。

設定オプション = デフォルト値	型	説明
caching = True	ブール値	アプリケーション認証情報のキャッシュを切り替えます。これは、グローバルキャッシングが有効でない限り効果はありません。
driver = sql	文字列値	**keystone.application_credential** 名前空間のアプリケーション認証情報のバックエンドドライバーのエントリーポイント。Keystone は **sql** ドライバーのみを提供するため、カスタムエントリーポイントを提供しない限り、これを変更する必要はありません。
user_limit = -1	整数値	ユーザーが作成できるアプリケーション認証情報の最大数。-1 の値は無制限を意味します。制限が設定されていない場合には、ユーザーはアプリケーション認証情報を作成でき、これにより keystone データベースでエラーが発生したり、keystone が DoS 攻撃を行う可能性がありました。

7.1.3. assignment

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[assignment]** グループで使用できるオプションの概要をまとめたものです。

表7.2 assignment

設定オプション = デフォルト値	型	説明
driver = sql	文字列値	**keystone.assignment** 名前空間に、割り当てバックエンドドライバーのエントリーポイント (ロール割り当てが保存されている)keystone が提供しているのは SQL ドライバーのみです。keystone 用のプロプライエタリードライバーを記述していない限り、このオプションを設定する必要はありません。
prohibited_implied_role = ['admin']	リスト値	暗黙のロールになることを禁止されているロール名のリスト。

7.1.4. auth

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[auth]** グループで使用できるオプションの概要をまとめたものです。

表7.3 auth

設定オプション = デフォルト値	型	説明
application_credential = None	文字列値	**keystone.auth.application_credential** 名前空間の application_credential auth プラグインモジュールのエントリーポイント。keystone の独自の **application_credential** 認証プラグインを上書きする場合以外は、この設定を行う必要はありません。
external = None	文字列値	**keystone.auth.external** 名前空間の外部 (**REMOTE_USER**) 認証プラグインモジュールのエントリーポイント。提供されるドライバーは **DefaultDomain** および **Domain** です。デフォルトのドライバーは **DefaultDomain** です。ここでは、**REMOTE_USER** 変数の keystone に指定したユーザー名で識別されているすべてのユーザーが、デフォルトドメインのコンテキスト内に存在することを前提とします。**Domain** オプションは、**REMOTE_USER** のドメイン名を含む追加の環境変数 **REMOTE_DOMAIN** が keystone に提示されることを期待します (**REMOTE_DOMAIN** が設定されていない場合、代わりにデフォルトのドメインが使用されることになります)。外部認証を利用する場合、アプリケーションサーバー (Apache など) が keystone ではなく認証を処理している場合を除き、この設定を行う必要はありません。
mapped = None	文字列値	**keystone.auth.mapped** 名前空間のマップされた auth プラグインモジュールのエントリーポイント。keystone の独自の **mapped** 認証プラグインを上書きする場合以外は、この設定を行う必要はありません。
methods = ['external', 'password', 'token', 'oauth1', 'mapped', 'application_credential']	リスト値	許可される認証方法。注意: 現在フェデレーションを使用している場合は、**external** 認証メソッドを無効にする必要があります。外部認証とフェデレーションはいずれも REMOTE_USER 変数を使用します。マッピングされたプラグインと外部プラグインの両方がリクエスト環境で属性を検証するため、競合が発生する可能性があります。
oauth1 = None	文字列値	**keystone.auth.oauth1** 名前空間の OAuth 1.0a auth プラグインモジュールのエントリーポイント。keystone 自身の **oauth1** 認証プラグインを上書きする場合以外は、この設定を行う必要はありません。
password = None	文字列値	**keystone.auth.password** 名前空間のパスワード認証プラグインモジュールのエントリーポイント。keystone 自体のパスワード認証プラグインを上書きする場合を除き、この設定は必要ありません。

設定オプション = デフォルト値	型	説明
token = None	文字列値	**keystone.auth.token** 名前空間のトークン認証プラグインモジュールのエントリーポイント。keystone 自体のトークン認証プラグインを上書きする場合を除き、この設定は必要ありません。

7.1.5. cache

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[cache]** グループで使用できるオプションの概要をまとめたものです。

表7.4 cache

設定オプション = デフォルト値	型	説明
backend = dogpile.cache.null	文字列値	キャッシュバックエンドモジュール。数百のスレッドサーバーを持つイベントレットベースまたは環境の場合は、プーリングを含む Memcache (oslo_cache.memcache_pool) が推奨されます。100 台未満のスレッドサーバーがある環境では、Memcached(dogpile.cache.memcached) または Redis(dogpile.cache.redis) が推奨されます。サーバーの単一のインスタンスを含むテスト環境は、dogpile.cache.memory バックエンドを使用できます。
backend_argument = []	多値	バックエンドモジュールに指定された引数。このオプションは、dogpile.cache バックエンドに渡される引数ごとに 1 回指定します。例: "<argname>:<value>"
config_prefix = cache.oslo	文字列値	キャッシュリージョンの設定ディクショナリーを構築するための接頭辞。同じ設定名を持つ別の dogpile.cache リージョンがない場合は、これを変更する必要はありません。
dead_timeout = 60	浮動小数点の値	HashClient の内部メカニズムでプールにノードを再度追加しようとするまでの時間 (秒)。
debug_cache_backend = False	ブール値	キャッシュバックエンドからの追加デバッグ (キャッシュキー、get/set/delete/etc コール)。これは、キー/値を含む特定の cache-backend get/set/delete 呼び出しを確認する必要がある場合にのみ非常に便利です。通常、これは false に設定されている必要があります。

設定オプション = デフォルト値	型	説明
enable_retry_client = False	ブール値	失敗を処理する再試行クライアントメカニズムを有効にします。これらのメカニズムは、あらゆる種類の pymemcache クライアントをラップするために使用できます。ラッパーを使用すると、試行回数と attemot 間の待機時間を定義できます。
enable_socket_keepalive = False	ブール値	dogpile の pymemcache バックエンドのソケットキープアライブのグローバルトグル
enabled = True	ブール値	キャッシュのグローバル切り替え。
expiration_time = 600	整数値	dogpile.cache リージョンのキャッシュされた項目に対するデフォルトの TTL (秒単位)。これは、キャッシュの有効期限が明示的に定義されていないキャッシュされたメソッドに適用されます。
hashclient_retry_attempts = 2	整数値	HashClient の内部メカニズムでクライアントが無効としてマークされ、プールから削除されるまで、クライアントを試行する必要がある回数。
hashclient_retry_delay = 1	浮動小数点の値	HashClient の内部メカニズムでの再試行の間に経過する時間 (秒単位)。
memcache_dead_retry = 300	整数値	再試行するまで memcached サーバーが停止されているとみなされる秒数 (dogpile.cache.memcache および oslo_cache.memcache_pool バックエンドのみ)。
memcache_pool_connection_get_timeout = 10	整数値	操作が memcache クライアント接続を取得するのを待つ秒数。
memcache_pool_flush_on_reconnect = False	ブール値	再接続時に memcache がフラッシュされるかどうかをグローバルに切り替えます (oslo_cache.memcache_pool バックエンドのみ)。
memcache_pool_maxsize = 10	整数値	すべての memcached サーバーへのオープン接続の最大数 (oslo_cache.memcache_pool バックエンドのみ)。
memcache_pool_unused_timeout = 60	整数値	memcached への接続が閉じられる前にプールで未使用のままになる秒数 (oslo_cache.memcache_pool バックエンドのみ)。

設定オプション = デフォルト値	型	説明
memcache_servers = ['localhost:11211']	リスト値	ホスト: ポートの形式の Memcache サーバー (dogpile.cache.memcached および oslo_cache.memcache_pool バックエンドのみ)。特定のホストが IPv6 を参照する場合、または特定のドメインが IPv6 を参照する場合、特定のアドレスの前にアドレスファミリー (**inet6**) を付ける必要があります (例: **inet6[::1]:11211**、**inet6:[fd12:3456:789a:1::1]:11211**、**inet6:[controller-0.internalapi]:11211**)。アドレスファミリーが指定されていない場合、使用されるデフォルトのアドレスファミリーは、IPv4 に対応する **inet** になります。
memcache_socket_timeout = 1.0	浮動小数点の値	サーバーへの呼び出しごとにタイムアウト (秒単位)。(dogpile.cache.memcache および oslo_cache.memcache_pool バックエンドのみ)。
proxies = []	リスト値	dogpile.cache バックエンドの機能に影響を与えるプロキシークラス。changing-backend-behavior についての dogpile.cache ドキュメントを参照してください。
retry_attempts = 2	整数値	失敗するまでアクションを試行する回数。
retry_delay = 0	浮動小数点の値	各試行の間にスリープする秒数。
socket_keepalive_count = 1	整数値	接続を切断する前に TCP が送信するキープアライブプローブの最大数。ゼロより大きい正の整数である必要があります。
socket_keepalive_idle = 1	整数値	TCP がキープアライブプローブの送信を開始する前に、接続がアイドル状態を維持する必要がある時間 (秒単位)。ゼロより大きい正の整数である必要があります。
socket_keepalive_interval = 1	整数値	個々のキープアライブプローブ間の時間 (秒単位)。ゼロより大きい正の整数である必要があります。
tls_allowed_ciphers = None	文字列値	TLS コンテキストで作成されたソケットで利用可能な暗号を設定します。OpenSSL 暗号リスト形式の文字列である必要があります。指定されていない場合は、OpenSSL 対応の暗号がすべて利用可能になります。

設定オプション = デフォルト値	型	説明
tls_cafile = None	文字列値	キャッシュサーバーの信頼性を確立するのに必要な PEM 形式の連結された CA 証明書のファイルへのパス。tls_enabled が False の場合、このオプションは無視されます。
tls_certfile = None	文字列値	クライアントの証明書と証明書の信頼性を確立するために必要な任意の数の CA 証明書を含む PEM 形式の単一ファイルへのパス。このファイルは、クライアント側の認証が必要な場合にのみ必要です。tls_enabled が False の場合、このオプションは無視されます。
tls_enabled = False	ブール値	キャッシングサーバーと連携する際に、TLS の使用をグローバルに切り替えます。
tls_keyfile = None	文字列値	クライアントの秘密鍵を含む単一ファイルへのパス。それ以外の場合、秘密鍵は tls_certfile で指定されたファイルから取得されます。tls_enabled が False の場合、このオプションは無視されます。

7.1.6. catalog

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[catalog]** グループで使用できるオプションの概要をまとめたものです。

表7.5 catalog

設定オプション = デフォルト値	型	説明
cache_time = None	整数値	カタログデータのキャッシュ時間 (秒単位)。グローバルキャッシュとカタログキャッシングの両方が有効でない限り、効果はありません。カタログデータ (サービス、エンドポイントなど) は通常頻繁に変更されないため、グローバルデフォルトよりも長い期間が望ましい場合があります。
caching = True	ブール値	カタログキャッシュを切り替えます。これは、グローバルキャッシングが有効でない限り効果はありません。一般的なデプロイメントでは、これを無効にする理由はありません。

設定オプション = デフォルト値	型	説明
driver = sql	文字列値	**keystone.catalog** 名前空間内のカタログドライバーのエントリーポイント。Keystone は **sql** オプション (SQL 経由の基本的な CRUD 操作)、**templated** オプション (ディスク上のテンプレート化されたカタログからカタログを読み込む)、および **endpoint_filter.sql** オプション (プロジェクトごとに任意のサービスカタログをサポートする) を提供します。
list_limit = None	整数値	カタログコレクションで返されるエンティティーの最大数。デプロイメントに十分なサービスやエンドポイントがあり、妥当な制限を超えるのに十分なサービスやエンドポイントがあるため、通常はこれを設定する必要はありません。
template_file = default_catalog.templates	文字列値	テンプレート化されたカタログバックエンドに使用するファイルへの絶対パス。このオプションは、**[catalog] driver** が **templated** に設定されている場合にのみ使用さ れ ます。

7.1.7. cors

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[cors]** グループで使用できるオプションの概要をまとめたものです。

表7.6 cors

設定オプション = デフォルト値	型	説明
allow_credentials = True	ブール値	実際の要求にユーザーの認証情報を含めることができることを示します。
allow_headers = ['X-Auth-Token', 'X-Openstack-Request-Id', 'X-Subject-Token', 'X-Project-Id', 'X-Project-Name', 'X-Project-Domain-Id', 'X-Project-Domain-Name', 'X-Domain-Id', 'X-Domain-Name', 'Openstack-Auth-Receipt']	リスト値	実際の要求時に使用されるヘッダーフィールド名を示します。

設定オプション = デフォルト値	型	説明
allow_methods = ['GET'、'PUT'、'POST'、'DELETE'、'PATCH']	リスト値	実際の要求時に使用できるメソッドを示します。
allowed_origin = None	リスト値	このリソースがリクエストの origin"ヘッダーで受信したドメインと共有されるかどうかを示します。形式:"<protocol>://<host>[:<port>]" (行末のスラッシュなし)例:https://horizon.example.com
expose_headers = ['X-Auth-Token', 'X-Openstack-Request-Id', 'X-Subject-Token', 'Openstack-Auth-Receipt']	リスト値	API に安全に公開できるヘッダーを示します。デフォルトは HTTP Simple ヘッダーです。
max_age = 3600	整数値	CORS プリフライトリクエストの最大キャッシュ期間。

7.1.8. credential

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[credential]** グループで使用できるオプションの概要をまとめたものです。

表7.7 credential

設定オプション = デフォルト値	型	説明
auth_ttl = 15	整数値	署名付き EC2 または S3 トークン要求がトークン要求に含まれるタイムスタンプから有効になる期間 (分単位)。
cache_time = None	整数値	認証情報データを秒単位でキャッシュする時間 (秒単位)。これは、グローバルキャッシングが有効でない限り効果はありません。
caching = True	ブール値	ユーザー認証情報の取得時にのみキャッシュを有効にします。これは、グローバルキャッシングが有効でない限り効果はありません。
driver = sql	文字列値	**keystone.credential** 名前空間内のクレデンシャルバックエンドドライバーのエントリーポイント。Keystone は **sql** ドライバーのみを提供するため、カスタムエントリーポイントを提供しない限り、この変更の理由はありません。

設定オプション = デフォルト値	型	説明
key_repository = /etc/keystone/credential-keys/	文字列値	認証情報のバックエンドに保存されている認証情報を暗号化および復号化するために使用される Fernet キーが含まれるディレクトリー。認証情報の暗号化に使用される Fernet キーは、Fernet トークンの暗号化に使用される Fernet キーとは関係ありません。両方の鍵は別々に管理する必要があり、異なるローテーションポリシーが必要です。Fernet トークンのキーの管理に使用するリポジトリーとこのリポジトリーを共有しないでください。
provider = fernet	文字列値	**keystone.credential.provider** 名前空間での認証情報の暗号化および復号操作のエントリーポイント。Keystone は **fernet** ドライバーのみを提供するため、認証情報を暗号化および復号化するカスタムエントリーポイントを提供しない限り、この変更の理由はありません。
user_limit = -1	整数値	ユーザーが作成できる認証情報の最大数。-1 の値は無制限を意味します。制限が設定されていない場合、ユーザーは認証情報を自由に作成できます。これにより、keystone データベースが肥大化したり、keystone が DoS 攻撃を受ける可能性があります。

7.1.9. database

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[database]** グループで使用できるオプションの概要をまとめたものです。

表7.8 database

設定オプション = デフォルト値	型	説明
backend = sqlalchemy	文字列値	データベースに使用するバックエンド。
connection = None	文字列値	データベースへの接続に使用する SQLAlchemy 接続文字列。
connection_debug = 0	整数値	SQL デバッグ情報の冗長性: 0=なし、100=すべて。
`connection_parameters = `	文字列値	接続時に接続 URL に追加するオプションの URL パラメーター。param1=value1&param2=value2&... として指定します。

設定オプション = デフォルト値	型	説明
connection_recycle_time = 3600	整数値	この秒数より長く接続プールに存在していた接続は、次にプールからチェックアウトされたときに新しい接続に置き換えられます。
connection_trace = False	ブール値	Python スタックトレースをコメント文字列として SQL に追加します。
db_inc_retry_interval = True	ブール値	True の場合には、db_max_retry_interval までのデータベース操作の再試行回数を長くします。
db_max_retries = 20	整数値	エラーが発生する前に接続エラーまたはデッドロックが発生した場合の最大再試行。再試行回数を無限に指定するには -1 に設定します。
db_max_retry_interval = 10	整数値	db_inc_retry_interval が設定されている場合は、データベース操作を再試行するまでの最大秒数です。
db_retry_interval = 1	整数値	データベーストランザクションの再試行間の秒数。
max_overflow = 50	整数値	設定されている場合、この値を SQLAlchemy で max_overflow に使用します。
max_pool_size = 5	整数値	プールに開いたままにする SQL 接続の最大数。値を 0 に設定すると無制限を意味します。
max_retries = 10	整数値	起動時のデータベース接続の最大再試行数。再試行回数を無限に指定するには -1 に設定します。
mysql_enable_ndb = False	ブール値	True の場合には、MySQL Cluster (NDB) の処理のサポートを透過的に有効にします。
mysql_sql_mode = TRADITIONAL	文字列値	MySQL セッションに使用する SQL モード。このオプション (デフォルトを含む) は、サーバーセット SQL モードを上書きします。サーバー設定で設定された SQL モードを使用するには、これを no value に設定します。例: mysql_sql_mode=
pool_timeout = None	整数値	設定されている場合は、この値を SQLAlchemy で pool_timeout に使用します。
retry_interval = 10	整数値	SQL 接続を開く再試行の間隔。
slave_connection = None	文字列値	スレーブデータベースへの接続に使用する SQLAlchemy 接続文字列。
sqlite_synchronous = True	ブール値	True の場合、SQLite は同期モードを使用します。

設定オプション = デフォルト値	型	説明
use_db_reconnect = False	ブール値	失われた接続でデータベースの実験的な使用を有効にします。

7.1.10. domain_config

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[domain_config]** グループで使用できるオプションの概要をまとめたものです。

表7.9 domain_config

設定オプション = デフォルト値	型	説明
cache_time = 300	整数値	ドメイン固有の設定データをキャッシュするための Time-to-live (TTL)。**[domain_config] caching** が有効でない限り、これは効果がありません。
caching = True	ブール値	ドメイン固有の設定バックエンドのキャッシュを切り替えます。これは、グローバルキャッシングが有効でない限り効果はありません。通常、これを無効にする理由はありません。
driver = sql	文字列値	**keystone.resource.domain_config** 名前空間のドメイン固有の設定ドライバーのエントリーポイント。keystone では **sql** オプションのみが提供されないので、カスタムエントリーポイントを指定しない限り、これを設定する必要はありません。

7.1.11. endpoint_filter

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[endpoint_filter]** グループで使用できるオプションの概要をまとめたものです。

表7.10 endpoint_filter

設定オプション = デフォルト値	型	説明
driver = sql	文字列値	**keystone.endpoint_filter** 名前空間内のエンドポイントフィルタードライバーのエントリーポイント。keystone では **sql** オプションのみが提供されないので、カスタムエントリーポイントを指定しない限り、これを設定する必要はありません。

設定オプション = デフォルト値	型	説明
return_all_endpoints_if_no_filter = True	ブール値	これにより、設定されたエンドポイントフィルターでユーザー + プロジェクトペアのエンドポイントが生じない場合 (したがって、空のサービスカタログ) が keystone の動作を制御します。true に設定すると、keystone はサービスカタログ全体を返します。false に設定すると、keystone は空のサービスカタログを返します。

7.1.12. endpoint_policy

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[endpoint_policy]** グループで使用できるオプションの概要をまとめたものです。

表7.11 endpoint_policy

設定オプション = デフォルト値	型	説明
driver = sql	文字列値	**keystone.endpoint_policy** 名前空間内のエンドポイントポリシードライバーのエントリーポイント。keystone では **sql** ドライバーのみが提供されないので、カスタムエントリーポイントを指定しない限り、これを設定する必要はありません。

7.1.13. eventlet_server

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[eventlet_server]** グループで使用できるオプションの概要をまとめたものです。

表7.12 eventlet_server

設定オプション = デフォルト値	型	説明
admin_bind_host = 0.0.0.0	ホストのアドレス値	admin サービスがリッスンするネットワークインターフェイスの IP アドレス。**非推奨となったバージョン:** K *reason:*eventlet の下に keystone を実行するサポートは、Newton リリースで削除されました。これらのオプションは URL 置換に使用されるため、後方互換性を維持します。

設定オプション = デフォルト値	型	説明
admin_port = 35357	ポート値	リッスンする admin サービスのポート番号。**非推奨となったバージョン:** K *reason:*eventlet の下に keystone を実行するサポートは、Newton リリースで削除されました。これらのオプションは URL 置換に使用されるため、後方互換性を維持します。
public_bind_host = 0.0.0.0	ホストのアドレス値	パブリックサービスがリッスンするネットワークインターフェイスの IP アドレス。**非推奨となったバージョン:** K *reason:*eventlet の下に keystone を実行するサポートは、Newton リリースで削除されました。これらのオプションは URL 置換に使用されるため、後方互換性を維持します。
public_port = 5000	ポート値	リッスンするパブリックサービスのポート番号。**非推奨となったバージョン:** K *reason:*eventlet の下に keystone を実行するサポートは、Newton リリースで削除されました。これらのオプションは URL 置換に使用されるため、後方互換性を維持します。

7.1.14. federation

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[federation]** グループで使用できるオプションの概要をまとめたものです。

表7.13 federation

設定オプション = デフォルト値	型	説明
`assertion_prefix = `	文字列値	フェデレーションされたアサーションの環境変数名をフィルターする際に使用する接頭辞。一致する変数は、フェデレーションされたマッピングエンジンに渡されます。
caching = True	ブール値	フェデレーションキャッシングを切り替えます。これは、グローバルキャッシングが有効でない限り効果はありません。通常、これを無効にする理由はありません。

設定オプション = デフォルト値	型	説明
default_authorization_ttl = 0	整数値	マッピングから引き継がれたグループメンバーシップの有効期間のデフォルト時間 (分単位)。デフォルトは 0 で、無効を意味します。
driver = sql	文字列値	**keystone.federation** 名前空間内のフェデレーションバックエンドドライバーのエントリーポイント。Keystone は **sql** ドライバーのみを提供するため、カスタムエントリーポイントを提供しない限り、このオプションを設定する必要はありません。
federated_domain_name = Federated	文字列値	フェデレーションされた一時ユーザーがドメインの概念を設定できるように予約される任意のドメイン名。管理者は、この名前でドメインを作成したり、既存のドメインをこの名前に更新したりすることはできません。本当に必要でない限り、この値を変更しないようにしてください。**非推奨となったバージョン:** T *理由:* このオプションは、アイデンティティープロバイダーのドメインに存在する一時ユーザーに置き換えられました。
remote_id_attribute = None	文字列値	環境からアイデンティティープロバイダーのエンティティー ID を取得するために使用されるすべてのプロトコルのデフォルト値。**mod_shib** の場合、これは **Shib-Identity-Provider** になります。**mod_auth_openidc** の場合、これは **HTTP_OIDC_ISS** にすることができます。**mod_auth_mellon** の場合、これは **MELLON_IDP** になります。これは、API を使用して、**remote_id_attribute** をフェデレーションプロトコルに提供することで、プロトコルごとに上書きできます。
sso_callback_template = /etc/keystone/sso_callback_template.html	文字列値	シングルサインオンコールバックハンドラーとして使用される HTML ファイルへの絶対パス。このページには、POST 要求でトークンをエンコードすることで、keystone からのユーザーを信頼済みのダッシュボードホストにリダイレクトすることが想定されています。ほとんどのデプロイメントでは、Keystone のデフォルト値で十分です。

設定オプション = デフォルト値	型	説明
trusted_dashboard = []	多値	信頼済みダッシュボードホストのリスト。トークンを返すためのシングルサインオン要求を受け入れる前に、元のホストはこのリストのメンバーである必要があります。この設定オプションは、複数の値に対して繰り返すことができます。これは、web ベースの SSO フローを使用するために設定する必要があります。たとえば、trusted_dashboard=https://acme.example.com/auth/websso trusted_dashboard=https://beta.example.com/auth/websso のようになります。

7.1.15. fernet_receipts

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[fernet_receipts]** グループで使用できるオプションの概要をまとめたものです。

表7.14 fernet_receipts

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
key_repository = /etc/keystone/fernet-keys/	文字列値	Fernet receipt キーを含むディレクトリー。このディレクトリーは、**keystone-manage fernet_setup** を初めて使用する前に存在し、**keystone-manage fernet_setup** または **keystone-manage fernet_rotate** を実行しているユーザーが書き込み可能で、もちろん keystone のサーバープロセスも読み取り可能でなければなりません。リポジトリーには、受信検証に使用される単一のステージキー (常にインデックス 0)、受信に使用される 1 つのプライマリーキー (常に最も高いインデックス)、受信および検証に使用される 1 つのセカンダリーキー (他のすべてのインデックス値)、および受信検証に使用される 1 つのセカンダリーキー (他のすべてのインデックス値) が含まれます。複数の keystone ノードの場合、各ノードはステージドキー (インデックス 0) を除き、同じキーリポジトリーコンテンツを共有する必要があります。1 つのノードで **keystone-manage fernet_rotate** を 1 回実行し、ステージングキー (インデックス 0) を新しいプライマリーに昇格させ (前の最高インデックスから増量)、新しいステージングキー (インデックス 0 の新しいキー) を生成することは安全である。出来上がったリポジトリーは、レースコンディションのリスクなしに他のノードにアトミックに複製できます。(例えば、ホスト A で**keystone-manage fernet_rotate** を実行し、任意の時間を待ち、ホスト A でディレクトリーの tarball を作成し、ホスト B でそれを一時的に解凍し、ホスト B の場所にディレクトリーをアトミック (**mv**) 移動しても問題ないです)。他のノードを同期せずにキーリポジトリーで **keystone-manage fernet_rotate** を **2 回** 実行すると、受信が全ノードで検証できなくなります。
max_active_keys = 3	整数値	これは、**keystone-manage fernet_rotate** が破棄される前に、ローテーションで保持される鍵の数を制御します。デフォルト値の 3 は、keystone がステージドキー (常にインデックス 0)、1 つのプライマリーキー (最大数インデックス)、および 1 つのセカンダリーキー (他のすべてのインデックス) を維持することを意味します。この値を増やすと、セカンダリーキーがローテーションに保持されます。

7.1.16. fernet_tokens

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[fernet_tokens]** グループで使用できるオプションの概要をまとめたものです。

表7.15 fernet_tokens

設定オプション = デフォルト値	型	説明
key_repository = /etc/keystone/fernet-keys/	文字列値	Fernet トークンキーを含むディレクトリー。このディレクトリーは、**keystone-manage fernet_setup** を初めて使用する前に存在し、**keystone-manage fernet_setup** または **keystone-manage fernet_rotate** を実行しているユーザーが書き込み可能で、もちろん keystone のサーバープロセスも読み取り可能でなければなりません。リポジトリーには、トークン検証に使用される単一のステージキー (常にインデックス 0)、トークンの作成と検証に使用される単一のプライマリーキー (最も高いインデックス)、およびトークン検証に使用される 1 つのセカンダリーキー (他のすべてのインデックス値) の 3 つの状態のキーが含まれる可能性があります。複数の keystone ノードの場合、各ノードはステージドキー (インデックス 0) を除き、同じキーリポジトリーコンテンツを共有する必要があります。1 つのノードで **keystone-manage fernet_rotate** を 1 回実行し、ステージングキー (インデックス 0) を新しいプライマリーに昇格させ (前の最高インデックスから増量)、新しいステージングキー (インデックス 0 の新しいキー) を生成することは安全である。出来上がったリポジトリーは、レースコンディションのリスクなしに他のノードにアトミックに複製できます。(例えば、ホスト A で**keystone-manage fernet_rotate** を実行し、任意の時間を待ち、ホスト A でディレクトリーの tarball を作成し、ホスト B でそれを一時的に解凍し、ホスト B の場所にディレクトリーをアトミック (**mv**) 移動しても問題ないです)。他のノードを同期せずに、キーリポジトリーで **keystone-manage fernet_rotate** を 2 **回** 実行すると、全ノードで検証できないトークンが生成されます。
max_active_keys = 3	整数値	これは、**keystone-manage fernet_rotate** が破棄される前に、ローテーションで保持される鍵の数を制御します。デフォルト値の 3 は、keystone がステージドキー (常にインデックス 0)、1 つのプライマリーキー (最大数インデックス)、および 1 つのセカンダリーキー (他のすべてのインデックス) を維持することを意味します。この値を増やすと、セカンダリーキーがローテーションに保持されます。

7.1.17. Healthcheck

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[healthcheck]** グループで使用できるオプションの概要をまとめたものです。

表7.16 Healthcheck

設定オプション = デフォルト値	型	説明
backends = []	リスト値	ヘルスチェックを実施し、リクエストの一部としてその情報を報告できる追加のバックエンド。
detailed = False	ブール値	応答の一部として詳細情報を表示します。セキュリティーメモ: このオプションを有効にすると、監視対象のサービスに関する機密情報が漏洩する可能性があります。セキュリティーポリシーに違反しないことを確認してください。
disable_by_file_path = None	文字列値	ファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを確認します。DisableByFileHealthcheck プラグインで使用されます。
disable_by_file_paths = []	リスト値	ポートに基づいてファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを判断します。文字列の port:path リストを想定します。DisableByFilesPortsHealthcheck プラグインで使用されます。
path = /healthcheck	文字列値	healtcheck 要求に応答するパス。

7.1.18. identity

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[identity]** グループで使用できるオプションの概要をまとめたものです。

表7.17 identity

設定オプション = デフォルト値	型	説明
cache_time = 600	整数値	アイデンティティーデータのキャッシュ時間 (秒単位)。グローバルおよびアイデンティティーキャッシングが有効でない限り、効果はありません。
caching = True	ブール値	アイデンティティーキャッシングを切り替えます。これは、グローバルキャッシングが有効でない限り効果はありません。通常、これを無効にする理由はありません。

設定オプション = デフォルト値	型	説明
default_domain_id = default	文字列値	これは、すべての Identity API v2 要求に使用するドメインを参照します (これはドメインを認識しません)。この ID を持つドメインは、オプションで **keystone-manage bootstrap** により作成することができます。v2 API が誤って破損しないように、この ID によって参照されるドメインは v3 API で削除できません。このドメインには特別なことは何もありませんが、v2 クライアントのサポートを維持するために存在する必要があるという事実とは関係ありません。通常、この値を変更する理由はありません。
domain_config_dir = /etc/keystone/domains	文字列値	keystone がドメイン固有の **[identity]** 設定ファイルを見つける必要のある絶対パス。**[identity] domain_specific_drivers_enabled** が true に設定されていない限り、このオプションは機能しません。通常、この値を変更する理由はありません。
domain_configurations_from_database = False	ブール値	デフォルトでは、ドメイン固有の設定データは、**[identity] domain_config_dir** で識別されるディレクトリー内のファイルから読み込まれます。このオプションを有効にすると、代わりに API 経由でドメイン固有の設定を管理することができます。これは、ディスク上の設定ファイルを使用するのではなく、バックエンド (通常は SQL データベース) で永続化されます。
domain_specific_drivers_enabled = False	ブール値	ドメインのサブセット (またはすべて) には独自のアイデンティティードライバーがあり、それぞれに独自の部分的な設定オプションがあり、リソースバックエンドまたはドメイン設定ディレクトリーのファイルのいずれかに保存されます (**[identity] domain_configurations_from_database** の設定により異なります)。このようにドメインに固有の値のみを指定する必要があります。この機能はデフォルトでは無効になっていますが、今後のリリースではデフォルトで有効化される可能性があります。有効にするには true に設定します。

設定オプション = デフォルト値	型	説明
driver = sql	文字列値	**keystone.identity** 名前空間の identity バックエンドドライバーのエントリーポイント。Keystone は **sql** ドライバーと **ldap** ドライバーを提供します。このオプションは、[identity] domain_specific_drivers_enabled が有効になった場合に、**[identity] domain_specific_drivers_enabled** が有効になった場合に、デフォルトのドライバー選択 (およびこのセクションの他の設定変数とともに) として使用されますが、該当するドメイン固有の設定は定義されません。デプロイメントが主に **ldap** と を使用しておらず、ドメイン固有の設定を使用していない限り、通常はこれを **sql** に設定したままにする必要があります。
list_limit = None	整数値	アイデンティティーコレクションで返されるエンティティーの最大数。
max_password_length = 4096	整数値	ユーザーパスワードの最大許容長。パフォーマンスを向上させるために、この値を小さくします。この値を変更しても、既存のパスワードは影響を受けません。この値は、設定された値よりも優先される特定のハッシュアルゴリズムの最大許容長によってオーバーライドされることもあります。bcrypt max_password_length は 72 バイトです。
password_hash_algorithm = bcrypt	文字列値	keystone に保存されているパスワードに使用するパスワードハッシュアルゴリズム。
password_hash_rounds = None	整数値	このオプションは、セキュリティーとパフォーマンス間のトレードオフを表します。値が大きいほどパフォーマンスは低下しますが、セキュリティーが強化されます。このオプションを変更すると、既存のパスワードハッシュにすでに固定数のラウンドが適用されているため、新しく作成されたパスワードにのみ影響するため、実行中のクラスターでこのオプションを調整しても安全です。bcrypt のデフォルト値は 12 で、4 から 31 までの値である必要があります。scrypt のデフォルト値は 16 で、**range(1,32)** 内に指定する必要があります。pbkdf_sha512 のデフォルトは 60000 で、**range (1,1<<32)** 内である必要があります。警告: scrypt を使用する場合は、この値を大きくして、パスワードのハッシュに BOTH 時間およびメモリー要件を増やします。

設定オプション = デフォルト値	型	説明
salt_bytesize = None	整数値	scrypt および pbkfd2_sha512 ハッシュの salt で使用するバイト数。scrypt のデフォルト値は 16 バイトです。デフォルトは、pbkfd2_sha512 は 16 バイトです。パスワードハッシュを保存するために使用される列のサイズにより、96 バイトの最大値に制限されます。
scrypt_block_size = None	整数値	scrypt ハッシュ関数に渡す任意のブロックサイズ (**r** パラメーター)。scrypt を CPU アーキテクチャーのパフォーマンスを最適に調整するのに便利です。このオプションは、**password_hash_algorithm** オプションが **scrypt** に設定されている場合にのみ使用されます。デフォルトは 8 です。
scrypt_parallelism = None	整数値	scrypt ハッシュ関数に渡すオプションの並列処理 (**p** パラメーター)。このオプションは、**password_hash_algorithm** オプションが **scrypt** に設定されている場合にのみ使用されます。デフォルトは 1 です。

7.1.19. identity_mapping

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[identity_mapping]** グループで使用できるオプションの概要をまとめたものです。

表7.18 identity_mapping

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
backward_compatible_ids = True	ブール値	UUID を生成しないバックエンド (例: LDAP) で、Jupno でユーザーおよびグループ ID の形式が変更され、keystone は LDAP の基礎となる属性にハッシュのマッピングを提供します。デフォルトでは、このマッピングが無効になり、既存の ID が変更されなくなります。ドメイン固有のドライバー (**[identity] domain_specific_drivers_enabled**) を使用してマッピングが有効になっている場合でも、LDAP によって処理されるデフォルトドメインのユーザーおよびグループは、ID が後方互換性を維持するようにマッピングされません。この値を false に設定すると、デフォルトの LDAP ドライバーを含め、すべてのバックエンドの新しいマッピングが有効になります。デフォルトの LDAP ドメインからユーザーおよびグループが割り当てられておらず、Keystone で以前に行った異なる ID を提供していない場合 (API の既存 ID は突然変更される)、このオプションを安全に保つことが保証されます。通常、この値を false に設定できるのは、新しいインストールを設定するときだけであることを意味しますが、これは推奨値です。
driver = sql	文字列値	**keystone.identity.id_mapping** 名前空間の identity マッピングバックエンドドライバーのエントリーポイント。Keystone は **sql** ドライバーのみを提供するため、カスタムエントリーポイントを提供しない限り、これを変更する必要はありません。
generator = sha256	文字列値	**keystone.identity.id_generator** namespace 内のユーザーおよびグループエンティティーのパブリック ID ジェネレーターのエントリーポイント。Keystone アイデンティティーマッパーは、64 バイト以下を生成するジェネレーターのみをサポートします。Keystone は **sha256** のエントリーポイントのみを提供するため、カスタムエントリーポイントを指定しない限り、この値を変更する必要はありません。

7.1.20. jwt_tokens

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[jwt_tokens]** グループで使用できるオプションの概要をまとめたものです。

表7.19 jwt_tokens

設定オプション = デフォルト値	型	説明
jws_private_key_repository = /etc/keystone/jws-keys/private	文字列値	JWS トークンを署名するための秘密鍵が含まれるディレクトリー。keystone のサーバープロセスを開始するには、このディレクトリーが存在している必要があります。また、keystone の server プロセスでも読み取り可能でなければなりません。**keystone.conf [jwt_tokens] jws_public_key_repository** の公開鍵に対応する秘密鍵が少なくとも 1 つ含まれる必要があります。このディレクトリーに複数の秘密鍵がある場合、keystone は **private.pem** という名前の鍵を使用してトークンに署名します。今後、keystone は複数の秘密鍵でトークンに署名する機能をサポートする場合があります。現在、JWS トークンを発行するには、このディレクトリーに **private.pem** という名前のキーのみが必要になります。このオプションは、JWS トークンを発行し、**keystone.conf [token] provider = jws** を設定するデプロイメントにのみ適用されます。
jws_public_key_repository = /etc/keystone/jws-keys/public	文字列値	JWS トークン署名を検証するための公開鍵を含むディレクトリー。keystone のサーバープロセスを開始するには、このディレクトリーが存在している必要があります。また、keystone の server プロセスでも読み取り可能でなければなりません。**keystone.conf [jwt_tokens] jws_private_key_repository** の秘密鍵に対応するパブリックキーを少なくとも 1 つ含む必要があります。このオプションは、JWS トークンを発行し、**keystone.conf [token] provider = jws** を設定するデプロイメントにのみ適用されます。

7.1.21. ldap

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[ldap]** グループで使用できるオプションの概要をまとめたものです。

表7.20 ldap

設定オプション = デフォルト値	型	説明
alias_dereferencing = default	文字列値	エイリアスを含むクエリーに使用する LDAP 逆参照オプション。**default** の値は、**ldap.conf** によって設定されたデフォルトの逆参照動作を使用するようにフォールバックされます。値が **never** の場合は、エイリアスが逆参照されなくなります。**searching** の値は、名前解決の後にのみエイリアスの参照を解除します。**finding** の値は、名前解決時にのみエイリアスを参照します。**always** の値は、すべての場合において、エイリアスを非参照とする。

設定オプション = デフォルト値	型	説明
auth_pool_connection_lifetime = 60	整数値	LDAP サーバーへのエンドユーザー認証接続の最大有効期間 (秒単位)。このライフタイムを超えると、接続はバインドされず、接続プールから削除されます。このオプションは、**[ldap] use_auth_pool** も有効でない限り効果はありません。
auth_pool_size = 100	整数値	エンドユーザー認証に使用する接続プールのサイズ。このオプションは、**[ldap] use_auth_pool** も有効でない限り効果はありません。
chase_referrals = None	ブール値	ディレクトリーパーティション全体で、keystone の参照処理の動作を設定します。未設定のままにすると、システムのデフォルトの動作が使用されます。
connection_timeout = -1	整数値	LDAP サーバーで使用する接続のタイムアウト。**-1** を値として指定すると、接続はタイムアウトしません。
debug_level = None	整数値	LDAP 呼び出しの LDAP デバッグレベルを設定します。値が 0 の場合は、デバッグが有効になっていないことを意味します。この値はビットマスクです。可能な値については LDAP ドキュメントを参照してください。
group_ad_nesting = False	ブール値	有効にすると、グループクエリーはネスト化されたグループに Active Directory 固有のフィルターを使用します。
group_additional_attribute_mapping = []	リスト値	追加の属性を keystone のグループにマッピングするために使用される keystone グループ属性のペアへの LDAP 属性のリスト。予期される形式は **<ldap_attr>:<group_attr>** です。**ldap_attr** は LDAP オブジェクトの属性で、**group_attr** は Identity API に表示される属性です。
group_attribute_ignore = []	リスト値	作成および更新時に無視するグループ属性のリスト。または、特定のグループ属性をリストまたは表示するグループ属性にフィルターする必要があるかどうか。
group_desc_attribute = description	文字列値	keystone のグループの説明にマッピングされた LDAP 属性。
group_filter = None	文字列値	グループに使用する LDAP 検索フィルター。

設定オプション = デフォルト値	型	説明
group_id_attribute = cn	文字列値	keystone のグループ ID にマッピングされる LDAP 属性。これは多値属性にしないでください。グループ ID は、keystone ドメイン全体でグローバルに一意であり、URL セーフになることが予想されます。
group_member_attribute = member	文字列値	ユーザーがグループのメンバーであることを示すために使用される LDAP 属性です。
group_members_are_ids = False	ブール値	グループオブジェクトクラスのメンバーが LDAP DN ではなく keystone ユーザー ID である場合は、このオプションを有効にします。Open Directory のグループオブジェクトクラスとして **posixGroup** を使用した場合です。
group_name_attribute = ou	文字列値	keystone のグループ名にマッピングされる LDAP 属性。グループ名は keystone ドメイン内でのみ一意となることが想定されており、URL セーフであることが予想されます。
group_objectclass = groupOfNames	文字列値	グループに使用する LDAP オブジェクトクラス。このオプションを **posixGroup** に設定 する 場合は、**[ldap] group_members_are_ids** オプションを有効にすることもできます。
group_tree_dn = None	文字列値	グループに使用する検索ベース。デフォルトは、**[ldap] suffix** が追加された **ou=UserGroups** です。
page_size = 0	整数値	オブジェクトをリスト表示する際に keystone が LDAP サーバーから要求する必要のある、ページごとの最大結果数を定義します。**0** を値として指定すると、ページングが無効になります。
password = None	文字列値	LDAP サーバーにクエリーを実行するときに使用する管理者バインド DN のパスワードは、LDAP サーバーが必要な場合です。
pool_connection_lifetime = 600	整数値	LDAP サーバーへの最大接続寿命を秒単位で指定します。このライフタイムを超えると、接続はバインドされず、接続プールから削除されます。このオプションは、**[ldap] use_pool** が有効でない限り効果はありません。
pool_connection_timeout = -1	整数値	LDAP 接続のプーリング時に使用する接続のタイムアウト。**-1** を値として指定すると、接続はタイムアウトしません。このオプションは、**[ldap] use_pool** が有効でない限り効果はありません。

設定オプション = デフォルト値	型	説明
pool_retry_delay = 0.1	浮動小数点の値	LDAP サーバーへの再接続を試みるまで待機する秒数。このオプションは、**[ldap] use_pool** が有効でない限り効果はありません。
pool_retry_max = 3	整数値	LDAP サーバーへの再接続を試みる最大回数 (中止するまでの回数)。値をゼロにすると再試行が回避されます。このオプションは、**[ldap] use_pool** が有効でない限り効果はありません。
pool_size = 10	整数値	LDAP 接続プールのサイズ。このオプションは、**[ldap] use_pool** が有効でない限り効果はありません。
query_scope = one	文字列値	検索ベース内での詳細な検索方法を定義する検索範囲。値 **one** (**oneLevel** または **singleLevel** を表す) は、ベースオブジェクトのすぐ下のオブジェクトを検索することを示すが、ベースオブジェクト自体は含まれない。**sub** の値 (**subtree** または **wholeSubtree**) は、ベースオブジェクト自体と、その下のサブツリー全体の検索を示します。
randomize_urls = False	ブール値	各 keystone プロセスの URL の順序をランダム化します。これにより、最初のサーバーがダウンした場合、プロセス/スレッドはリストの下位のサーバーへの接続を試行する前に、指定のタイムアウトに達するまで待機するため、障害の動作がより緩やかになります。下位互換性のために、デフォルトは False です。
suffix = cn=example,cn=com	文字列値	**[ldap] user_tree_dn** または **[ldap] group_tree_dn** のいずれかを介して DN が定義されていない場合に使用するデフォルトの LDAP サーバー接尾辞。
tls_cacertdir = None	文字列値	LDAP サーバーと通信する際に使用する CA 証明書ディレクトリーへの絶対パス。**[ldap] tls_cacertfile** も設定している場合は、このオプションを設定する理由はありません。
tls_cacertfile = None	文字列値	LDAP サーバーと通信する際に使用する CA 証明書ファイルへの絶対パス。このオプションは **[ldap] tls_cacertdir** よりも優先されます。したがって、両方を設定する必要はありません。

設定オプション = デフォルト値	型	説明
tls_req_cert = demand	文字列値	受信 TLS セッションでクライアント証明書に対して実行するチェックを指定します。**demand** に設定すると、証明書は常に要求され、LDAP サーバーから必要になります。**allow** に設定すると、証明書は常に要求されますが、LDAP サーバーからは必要ありません。**never** に設定すると、証明書は要求されません。
url = ldap://localhost	文字列値	LDAP サーバーに接続するための URL。複数の LDAP URL をコンマ区切りの文字列として指定することができます。接続には、正常にバインドするための最初の URL が使用されます。
use_auth_pool = True	ブール値	エンドユーザー認証の LDAP 接続プールを有効にします。通常、これを無効にする理由はありません。
use_pool = True	ブール値	LDAP サーバーへのクエリーの LDAP 接続プールを有効にします。通常、これを無効にする理由はありません。
use_tls = False	ブール値	LDAP サーバーと通信する際に TLS を有効にします。また、このオプションを使用する際に **[ldap] tls_cacertfile** および **[ldap] tls_cacertdir** オプションも設定する必要があります。TLS ではなく LDAP over SSL (LDAPS) を使用している場合は、このオプションを設定しないでください。
user = None	文字列値	LDAP サーバーに必要な場合に、LDAP サーバーにクエリーする際に使用する管理者バインド DN のユーザー名。
user_additional_attribute_mapping = []	リスト値	追加属性を keystone のユーザー属性にマッピングするために使用される keystone ユーザー属性のペアへの LDAP 属性リスト予期される形式は **<ldap_attr>:<user_attr>** です。**ldap_attr** は LDAP オブジェクトの属性で、**group_attr** は Identity API に表示される属性です。
user_attribute_ignore = ['default_project_id']	リスト値	作成および更新時に無視するユーザー属性のリスト、または特定のユーザー属性をリストまたは表示するユーザー属性にフィルターする必要があるかどうか。
user_default_project_id_attribute = None	文字列値	keystone のユーザーの default_project_id にマッピングされる LDAP 属性。これは、keystone が LDAP への書き込みアクセスを持つ場合に最も一般的に使用されます。

設定オプション = デフォルト値	型	説明
user_description_attribute = description	文字列値	keystone のユーザーの説明にマッピングされた LDAP 属性。
user_enabled_attribute = enabled	文字列値	keystone の user enabled 属性にマッピングされた LDAP 属性。このオプションを **userAccountControl** に設定する場合は、**[ldap] user_enabled_mask** と **[ldap] user_enabled_default** を設定することを推奨します。
user_enabled_default = True	文字列値	ユーザーを有効にするためのデフォルト値。LDAP サーバーが、非ブール値 (ビットマスク) 値を使用してユーザーが有効または無効になっているかを示す場合は、適切な整数値と一致させる必要があります。これが **True** に設定されていない場合には、一般的な値は **512** になります。通常、これは **[ldap] user_enabled_attribute = userAccountControl** 時に使用されます。
user_enabled_emulation = False	ブール値	有効な場合には、keystone は別の方法を使用して、ユーザーが **[ldap] user_enabled_emulation_dn** オプションで定義したグループのメンバーかどうかをチェックして、有効かどうかを判別します。このオプションを有効にすると、keystone は **[ldap] user_enabled_invert** の値を無視します。
user_enabled_emulation_dn = None	文字列値	有効なエミュレーションを使用する場合に有効なユーザーを保持するグループエントリーの DN。**[ldap] user_enabled_emulation** も有効でない限り、このオプションを設定しても効果はありません。
user_enabled_emulation_use_group_config = False	ブール値	**[ldap] group_member_attribute** および **[ldap] group_objectclass** の設定を使用して、エミュレーションされた有効なグループのメンバーシップを決定します。**[ldap] user_enabled_emulation** も有効でない限り、このオプションを有効にしても効果はありません。
user_enabled_invert = False	ブール値	LDAP サーバーから取得した enabled 属性のブール値を論理的に無視します。一部の LDAP サーバーはブール値ロック属性を使用するものもあります。"true" はアカウントが無効を意味します。**[ldap] user_enabled_invert = true** を設定すると、これらのロック属性を使用できます。**[ldap] user_enabled_mask** オプションまたは **[ldap] user_enabled_emulation** オプションを使用する場合は、このオプションは機能しません。

設定オプション = デフォルト値	型	説明
user_enabled_mask = 0	整数値	LDAP サーバーが個別のブール値ではなく、整数の整数を整数で表した場合、有効な値を示すビットマスク整数。**0** を値として指定すると、マスクは使用されません。これを **0** に設定されていない場合、一般的な値は **2** になります。通常、これは **[ldap] user_enabled_attribute = userAccountControl** 時に使用されます。このオプションを設定すると、keystone は **[ldap] user_enabled_invert** の値を無視します。
user_filter = None	文字列値	ユーザーに使用する LDAP 検索フィルター。
user_id_attribute = cn	文字列値	keystone のユーザー ID にマッピングされた LDAP 属性これは多値属性にしないでください。ユーザー ID は、keystone ドメイン全体でグローバルに一意であり、URL セーフになることが予想されます。
user_mail_attribute = mail	文字列値	keystone のユーザーのメールにマップされた LDAP 属性。
user_name_attribute = sn	文字列値	keystone のユーザー名にマッピングされる LDAP 属性。ユーザー名は keystone ドメイン内でのみ一意となることが想定されており、URL セーフであることが予想されます。
user_objectclass = inetOrgPerson	文字列値	ユーザーに使用する LDAP オブジェクトクラス。
user_pass_attribute = userPassword	文字列値	keystone のユーザーパスワードにマッピングされる LDAP 属性。
user_tree_dn = None	文字列値	ユーザーに使用する検索ベース。デフォルトは、**[ldap] suffix** が追加された **ou=Users** です。

7.1.22. memcache

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[memcache]** グループで使用できるオプションの概要をまとめたものです。

表7.21 memcache

設定オプション = デフォルト値	型	説明
dead_retry = 300	整数値	memcached サーバーが再び試行されるまで停止されているとみなされる秒数。これは、キー値ストアシステムによって使用されます。

設定オプション = デフォルト値	型	説明
pool_connection_get_timeout = 10	整数値	操作が memcache クライアント接続を取得するのを待つ秒数。これは、キー値ストアシステムによって使用されます。
pool_maxsize = 10	整数値	すべての memcached サーバーへのオープン接続の最大数。これは、キー値ストアシステムによって使用されます。
pool_unused_timeout = 60	整数値	memcached への接続が、閉じられる前にプールに未使用のまま保持される秒数。これは、キー値ストアシステムによって使用されます。
socket_timeout = 3	整数値	サーバーへの呼び出しごとにタイムアウト (秒単位)。これは、キー値ストアシステムによって使用されます。**非推奨となったバージョン:** T *reason:* このオプションは oslo.cache で複製されます。代わりに memcached の socket_timeout を設定するように ''keystone.conf [cache] memcache_socket_timeout'' オプションを設定します。

7.1.23. oauth1

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[oauth1]** グループで使用できるオプションの概要をまとめたものです。

表7.22 oauth1

設定オプション = デフォルト値	型	説明
access_token_duration = 86400	整数値	OAuth アクセストークンの作成後も有効な状態を維持するための秒数。これは、コンシューマーがサービスプロバイダー (通常は keystone) と対話する必要のある時間です。このオプションをゼロに設定すると、アクセストークンが最後に実行されることを意味します。
driver = sql	文字列値	**keystone.oauth1** namespace の OAuth バックエンドドライバーのエントリーポイント。通常、カスタムエントリーポイントを指定しない限り、このオプションを設定する必要はありません。

設定オプション = デフォルト値	型	説明
request_token_duration = 28800	整数値	OAuth 要求トークンの作成後も有効な状態を維持するための秒数。これは、ユーザーがトークンを承認する必要がある時間です。このオプションをゼロに設定すると、リクエストトークンが永久に継続されます。

7.1.24. oslo_messaging_amqp

以下の表では、**/etc/keystone/keystone.conf** ファイルの **[oslo_messaging_amqp]** グループで利用可能なオプションの概要を説明します。

表7.23 oslo_messaging_amqp

設定オプション = デフォルト値	型	説明
addressing_mode = dynamic	文字列値	ドライバーが使用するアドレスモードを示します。許可される値:**legacy** - レガシーのルーティング不可能なアドレス指定を使用します。**routable** - ルーティング可能なアドレスを使用します。**dynamic** - メッセージバスがルーティングをサポートしていない場合はレガシーのアドレスを使用します。それ以外の場合は、ルーティング可能なアドレスを使用します。
anycast_address = anycast	文字列値	コンシューマーのグループに送信する際に、アドレス接頭辞に追加されます。コンシューマー間でラウンドロビン方式で配信される必要のあるメッセージを特定するために、メッセージバスで使用されます。
broadcast_prefix = broadcast	文字列値	すべてのサーバーにブロードキャストする場合に使用されるアドレス接頭辞
connection_retry_backoff = 2	整数値	フェイルオーバーの試みに失敗するたびに、connection_retry_interval をこの秒数だけ増やします。
connection_retry_interval = 1	整数値	再接続を試みる前に一時停止する秒数。
connection_retry_interval_max = 30	整数値	connection_retry_interval + connection_retry_backoff の上限
container_name = None	文字列値	AMQP コンテナーの名前。グローバルで一意でなければなりません。デフォルトは、生成された UUID です。

設定オプション = デフォルト値	型	説明
default_notification_exchange = None	文字列値	通知アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_notification_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **notify**
default_notify_timeout = 30	整数値	送信された通知メッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_reply_retry = 0	整数値	リカバリー可能なエラーが原因で失敗した返信メッセージを再送信する最大試行回数。
default_reply_timeout = 30	整数値	rpc 返信メッセージ配信の期限。
default_rpc_exchange = None	文字列値	RPC アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_rpc_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **rpc**
default_send_timeout = 30	整数値	rpc キャストまたは呼び出しメッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_sender_link_timeout = 600	整数値	アイドル状態の送信者リンクのパージをスケジュールする時間。期限切れ後にリンクの割り当てを解除します。
group_request_prefix = unicast	文字列値	グループの任意のサーバーに送信する際のアドレス接頭辞
idle_timeout = 0	整数値	非アクティブな接続のタイムアウト (秒単位)
link_retry_delay = 10	整数値	リカバリー可能なエラーが原因で失敗した AMQP 1.0 リンクの再接続間に一時停止する時間。
multicast_address = multicast	文字列値	ファンアウトメッセージを送信するときにアドレス接頭辞に追加されます。ファンアウトメッセージを識別するためにメッセージバスによって使用されます。
notify_address_prefix = openstack.org/om/notify	文字列値	すべての生成される通知アドレスのアドレス接頭辞

設定オプション = デフォルト値	型	説明
notify_server_credit = 100	整数値	受信通知メッセージのウィンドウサイズ
pre_settled = ['rpc-cast', 'rpc-reply']	多値	このタイプのメッセージを事前処置状態で送信します。事前処置されたメッセージは、ピアから確認応答を受信しません。注記: 事前処置されたメッセージは、配信に失敗した場合に警告なしに破棄される可能性があります。許可される値:**rpc-call** - RPC 呼び出しを事前処置状態で送信します。**rpc-reply**- RPC 返信を事前処置状態で送信します。**rpc-cast** - RPC キャストを事前処置状態で送信します。**notify** - 通知を事前処置状態で送信します。
pseudo_vhost = True	ブール値	仮想ホスト (qpidd など) をネイティブにサポートしていないメッセージバスの仮想ホストサポートを有効にします。true に設定すると、仮想ホスト名はすべてのメッセージバスアドレスに追加され、結果として仮想ホストごとにプライベート **サブネット** を作成します。メッセージバスが仮想ホストの名前として AMQP 1.0 オープン遂行動詞の **hostname** フィールドを使用して仮想ホストをサポートする場合は False に設定します。
reply_link_credit = 200	整数値	受信する RPC 返信メッセージのウィンドウサイズ。
rpc_address_prefix = openstack.org/om/rpc	文字列値	すべての生成される RPC アドレスのアドレス接頭辞
rpc_server_credit = 100	整数値	受信する RPC リクエストメッセージのウィンドウサイズ。
`sasl_config_dir = `	文字列値	SASL 設定が含まれるディレクトリーへのパス
`sasl_config_name = `	文字列値	設定ファイルの名前 (.conf 接尾辞なし)
`sasl_default_realm = `	文字列値	ユーザー名にレルムが存在しない場合に使用する SASL レルム
`sasl_mechanisms = `	文字列値	許可される SASL メカニズムのスペース区切りリスト
server_request_prefix = exclusive	文字列値	特定のサーバーに送信する時に使用するアドレス接頭辞
ssl = False	ブール値	SSL 経由で接続を試みます。その他の ssl 関連のパラメーターが指定されていない場合、システムの CA バンドルを使用してサーバーの証明書を検証します。

設定オプション = デフォルト値	型	説明
`ssl_ca_file = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_cert_file = `	文字列値	クライアント認証用の自己識別証明書 PEM ファイル
`ssl_key_file = `	文字列値	ssl_cert_file 証明書の署名に使用される秘密鍵 PEM ファイル (オプション)
ssl_key_password = None	文字列値	ssl_key_file を復号するためのパスワード (暗号化されている場合)
ssl_verify_vhost = False	ブール値	デフォルトでは、SSL は、サーバーの証明書の名前が transport_url のホスト名と一致することを確認します。設定によっては、代わりに仮想ホスト名を使用することが望ましい場合があります。たとえば、サーバーが Server Name Indication TLS 拡張 (rfc6066) を使用して、仮想ホストごとの証明書を提供する場合などです。サーバーの SSL 証明書が DNS 名ではなく仮想ホスト名を使用する場合は、ssl_verify_vhost を True に設定します。
trace = False	ブール値	デバッグ: AMQP フレームを標準出力 (stdout) にダンプします。
unicast_address = unicast	文字列値	特定の RPC/通知サーバーに送信する際に、アドレス接頭辞に追加されます。単一の送信先に送信されたメッセージを識別するためにメッセージバスによって使用されます。

7.1.25. oslo_messaging_kafka

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[oslo_messaging_kafka]** グループで使用できるオプションの概要をまとめたものです。

表7.24 oslo_messaging_kafka

設定オプション = デフォルト値	型	説明
compression_codec = none	文字列値	プロデューサーによって生成されたすべてのデータの圧縮コーデック。設定されていない場合、圧縮は使用されません。この設定に許可される値は、kafka バージョンに依存することに注意してください。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)

設定オプション = デフォルト値	型	説明
consumer_group = oslo_messaging_consumer	文字列値	Kafka コンシューマーのグループ ID。あるグループのコンシューマーは、メッセージ消費を調整します。
enable_auto_commit = False	ブール値	非同期コンシューマーコミットを有効にします。
kafka_consumer_timeout = 1.0	浮動小数点の値	Kafka コンシューマーのデフォルトタイムアウト
kafka_max_fetch_bytes = 1048576	整数値	Kafka コンシューマーの最大フェッチバイト
max_poll_records = 500	整数値	ポーリング呼び出しで返されるレコードの最大数
pool_size = 10	整数値	Kafka コンシューマーのプールサイズ
producer_batch_size = 16384	整数値	プロデューサーの非同期送信のバッチサイズ
producer_batch_timeout = 0.0	浮動小数点の値	KafkaProducer バッチ処理の遅延の上限 (秒単位)
sasl_mechanism = PLAIN	文字列値	セキュリティープロトコルが SASL である際のメカニズム
security_protocol = PLAINTEXT	文字列値	ブローカーとの通信に使用されるプロトコル
`ssl_cafile = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_client_cert_file = `	文字列値	認証に使用されるクライアント証明書の PEM ファイル。
`ssl_client_key_file = `	文字列値	認証に使用されるクライアントキーの PEM ファイル。
`ssl_client_key_password = `	文字列値	認証に使用されるクライアントキーパスワードファイル。

7.1.26. oslo_messaging_notifications

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[oslo_messaging_notifications]** グループで使用できるオプションの概要をまとめたものです。

表7.25 oslo_messaging_notifications

設定オプション = デフォルト値	型	説明
driver = []	多値	通知の送信を処理するドライバー。使用できる値は messaging、messagingv2、routing、log、test、noop です。
retry = -1	整数値	リカバリー可能なエラーが原因で配信に失敗した通知メッセージを再送信する最大試行回数。0 - 再試行なし、-1 - 無限回
topics = ['notifications']	リスト値	OpenStack の通知に使用する AMQP トピック。
transport_url = None	文字列値	通知に使用するメッセージングドライバーを表す URL。設定されていない場合は、RPC に使用されるものと同じ設定にフォールバックします。

7.1.27. oslo_messaging_rabbit

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[oslo_messaging_rabbit]** グループで使用できるオプションの概要をまとめたものです。

表7.26 oslo_messaging_rabbit

設定オプション = デフォルト値	型	説明
amqp_auto_delete = False	ブール値	AMQP の自動削除キュー。
amqp_durable_queues = False	ブール値	AMQP で永続キューを使用します。
direct_mandatory_flag = True	ブール値	(非推奨) ダイレクト送信のための RabbitMQ 必須フラグを有効/無効にします。直接送信は応答として使用されるため、クライアントキューが存在しない場合には MessageUndeliverable 例外が発生します。MessageUndeliverable 例外はタイムアウトをループして、送信者が回復する可能性が高くなります。このフラグは非推奨になり、この機能を非アクティブ化することはできません。
enable_cancel_on_failover = False	ブール値	x-cancel-on-ha-failover フラグを有効にして、rabbitmq サーバーがキューが停止しているときにコンシューマーをキャンセルし、通知できるようにします。

設定オプション = デフォルト値	型	説明
heartbeat_in_pthread = False	ブール値	デフォルトでは、ネイティブ Python スレッドを介してヘルスチェックハートビートスレッドを実行します。このオプションが False の場合、ヘルスチェックハートビートは親プロセスから実行モデルを継承します。たとえば、親プロセスが eventlet/greenlet を使用して stdlib にモンキーパッチを適用した場合、ハートビートはグリーンスレッドを介して実行されます。This option should be set to True only for the wsgi services.
heartbeat_rate = 2	整数値	heartbeat_timeout_threshold 中、ハートビートを確認する回数。
heartbeat_timeout_threshold = 60	整数値	ハートビートの keep-alive が失敗した場合に Rabbit ブローカーがダウンとみなされるまでの秒数 (0 はハートビートを無効にします)。
kombu_compression = None	文字列値	実験的用途: 許容値は gzip、bz2 です。設定されていない場合、圧縮は使用されません。このオプションは、今後のバージョンで利用できない可能性があります。
kombu_failover_strategy = round-robin	文字列値	現在接続しているノードが利用できなくなった場合に、次の RabbitMQ ノードを選択する方法を指定します。設定で複数の RabbitMQ ノードが指定される場合に限り有効になります。
kombu_missing_consumer_retry_timeout = 60	整数値	応答のないクライアントが応答を送信するのを待つ時間。この時間が経過すると無視します。この値は rpc_response_timeout より長くすることはできません。
kombu_reconnect_delay = 1.0	浮動小数点の値	AMQP コンシューマーの取り消し通知への応答で、再接続するまでの待機時間。
rabbit_ha_queues = False	ブール値	RabbitMQ (x-ha-policy: all) の HA キューの使用を試みます。このオプションを変更する場合は、RabbitMQ データベースを消去する必要があります。RabbitMQ 3.0 では、キューを宣言する際に x-ha-policy 引数によってキューのミラーリングが制御されなくなりました。すべてのキュー (自動生成された名前のキューを除く) がすべてのノードでミラーリングされるようにするには、"rabbitmqctl set_policy HA **^(?!amq\.).* {"ha-mode": "all"}**" を実行します。
rabbit_interval_max = 30	整数値	RabbitMQ 接続を再試行する最大間隔。デフォルトは 30 秒です。

設定オプション = デフォルト値	型	説明
rabbit_login_method = AMQPLAIN	文字列値	RabbitMQ ログイン方法
rabbit_qos_prefetch_count = 0	整数値	事前フェッチするメッセージの数を指定します。ゼロに設定すると、無制限のメッセージのフェッチが許可されます。
rabbit_retry_backoff = 2	整数値	RabbitMQ に接続する際に再試行間でバックオフする長さ
rabbit_retry_interval = 1	整数値	RabbitMQ との接続を再試行する頻度。
rabbit_transient_queues_ttl = 1800	整数値	キューの TTL (x-expires) の期間 (秒単位) を表す正の整数。TTL の期間使用されないキューは自動的に削除されます。このパラメーターは応答キューとファンアウトキューにのみ影響します。
ssl = False	ブール値	SSL 経由で接続します。
`ssl_ca_file = `	文字列値	SSL 認証局ファイル (SSL が有効な場合にのみ有効)。
`ssl_cert_file = `	文字列値	SSL 証明書ファイル (SSL が有効な場合にのみ有効)。
`ssl_key_file = `	文字列値	SSL キーファイル (SSL が有効な場合にのみ有効)。
`ssl_version = `	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

7.1.28. oslo_middleware

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[oslo_middleware]** グループで使用できるオプションの概要をまとめたものです。

表7.27 oslo_middleware

設定オプション = デフォルト値	型	説明
enable_proxy_headers_parsing = False	ブール値	アプリケーションがプロキシーの背後であるかどうか。これにより、ミドルウェアがヘッダーを解析すべきかどうかを決定します。

設定オプション = デフォルト値	型	説明
max_request_body_size = 114688	整数値	各リクエストの本文の最大サイズ (バイト単位)。
secure_proxy_ssl_header = X-Forwarded-Proto	文字列値	SSL 終端プロキシーによって非表示であった場合でも、元のリクエストプロトコルスキームを決定するために使用される HTTP ヘッダー。

7.1.29. oslo_policy

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[oslo_policy]** グループで使用できるオプションの概要をまとめたものです。

表7.28 oslo_policy

設定オプション = デフォルト値	型	説明
enforce_new_defaults = False	ブール値	このオプションは、ポリシーを評価するときに古い非推奨のデフォルトを使用するかどうかを制御します。**True** の場合、古い非推奨のデフォルトは評価されません。これは、既存のトークンが古いデフォルトで許可されているが、新しいデフォルトで許可されていない場合、許可されないことを意味します。新しいデフォルトと **scope_type** の利点を一緒に得ることができるように、このフラグを **enforce_scope** フラグと一緒に有効にすることを推奨します
enforce_scope = False	ブール値	このオプションは、ポリシーを評価する際にスコープを適用するかどうかを制御します。**True** の場合、リクエストで使用されるトークンのスコープは、適用されるポリシーの **scope_types** と比較されます。スコープが一致しない場合、**InvalidScope** 例外が発生します。**False** の場合、ポリシーが一致しないスコープで呼び出されていることを運用者に通知するメッセージが記録されます。
policy_default_rule = default	文字列値	デフォルトルール。要求されたルールが見つからない場合に適用されます。
policy_dirs = ['policy.d']	多値	ポリシー設定ファイルが保存されるディレクトリー。これは、config_dir オプションで定義される検索パス内の任意のディレクトリーへの相対パスか、絶対パスにすることができます。このディレクトリーの検索には、policy_file で定義されたファイルが存在している必要があります。存在しないディレクトリーまたは空のディレクトリーは無視されます。

設定オプション = デフォルト値	型	説明
policy_file = policy.yaml	文字列値	ロールを指定のサービスのパーミッションにマッピングするファイルの相対パスまたは絶対パス。相対パスは、このオプションを設定する設定ファイルに対する相対パスとして指定する必要があります。
remote_content_type = application/x-www-form-urlencoded	文字列値	REST ベースのポリシーチェックのデータを送受信するコンテンツ種別
remote_ssl_ca_crt_file = None	文字列値	REST ベースのポリシーチェック用の認証局証明書ファイルの絶対パス
remote_ssl_client_crt_file = None	文字列値	REST ベースのポリシーチェック用のクライアント証明書の絶対パス
remote_ssl_client_key_file = None	文字列値	REST ベースのポリシーチェック用のクライアント鍵ファイルの絶対パス
remote_ssl_verify_server_crt = False	ブール値	REST ベースのポリシーチェックのサーバーアイデンティティー検証

7.1.30. policy

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[policy]** グループで使用できるオプションの概要をまとめたものです。

表7.29 policy

設定オプション = デフォルト値	型	説明
driver = sql	文字列値	**keystone.policy** 名前空間のポリシーバックエンドドライバーのエントリーポイント。提供されるドライバーは **rules** (v3 ポリシー API の CRUD 操作をサポートしません) および **sql** です。通常、カスタムエントリーポイントを指定しない限り、このオプションを設定する必要はありません。
list_limit = None	整数値	ポリシーコレクションで返されるエンティティーの最大数。

7.1.31. profiler

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[profiler]** グループで使用できるオプションの概要をまとめたものです。

表7.30 profiler

設定オプション = デフォルト値	型	説明
connection_string = messaging://	文字列値	通知バックエンドの接続文字列。 デフォルト値は **messaging://** で、通知機能を oslo_messaging に設定します。 使用できる値の例: ● **messaging://**: スパンの送信に oslo_messaging ドライバーを使用します。 ● **redis://127.0.0.1:6379**: スパンの送信に redis ドライバーを使用します。 ● **mongodb://127.0.0.1:27017**: スパンの送信に mongodb ドライバーを使用します。 ● **elasticsearch://127.0.0.1:9200:** elasticsearch ドライバーを使用してスパンを送信します。 ● **jaeger://127.0.0.1:6831**: スパンの送信に jaeger トレースを使用します。
enabled = False	ブール値	このノード上の全サービスのプロファイリングを有効にします。 デフォルト値は False です (プロファイリング機能を無効にします)。 値: ● True: 機能を有効にします ● False: 機能を無効にします。このプロジェクトの操作ではプロファイリングを開始できません。プロファイリングが別のプロジェクトでトリガーされた場合には、このプロジェクトの部分は空になります。
es_doc_type = notification	文字列値	elasticsearch の通知インデックスについてのドキュメントタイプ。
es_scroll_size = 10000	整数値	elasticsearch は大規模な要求をバッチに分割します。このパラメーターは、各バッチの最大サイズを定義します (例: es_scroll_size=10000)。
es_scroll_time = 2m	文字列値	このパラメーターは時間値パラメーター (es_scroll_time=2m など) です。これは、検索に参加するノードが継続してサポートするために関連するリソースを維持する時間を示します。

設定オプション = デフォルト値	型	説明
filter_error_trace = False	ブール値	エラー/例外を含むフィルタートレースを分離された場所へ有効にします。 デフォルト値は False に設定されます。 値: ● True: エラー/例外を含むフィルタートレースを有効にします。 ● False: フィルターを無効にします。
hmac_keys = SECRET_KEY	文字列値	パフォーマンスプロファイリング用にコンテキストデータの暗号化に使用するシークレットキー。 この文字列値は <key1>[,<key2>,...<keyn>] の形式にする必要があります。各キーはランダムな文字列です。REST API 経由でプロファイリングをトリガーするユーザーは、REST API 呼び出しのヘッダーにこれらのキーの 1 つを設定し、この特定プロジェクトのノードのプロファイリング結果が含まれるようにする必要があります。 プロファイリングを有効にするには、enabled フラグと hmac_keys 設定オプションの両方を設定する必要があります。また、少なくとも 1 つのキーで全サービス間で正しいプロファイリング情報を生成するには、OpenStack プロジェクト間で一貫性を確保する必要があります。これにより、クライアント側からを使用して、すべての可能なリソースからの情報を含むトレースを生成できます。
sentinel_service_name = mymaster	文字列値	Redissentinel はサービス名を使用してマスターの redis サービスを識別します。このパラメーターは名前を定義します (例: **sentinal_service_name=mymaster**)。
socket_timeout = 0.1	浮動小数点の値	Redissentinel は、接続に timeout オプションを提供します。このパラメーターは、タイムアウトを定義します (例: socket_timeout=0.1)。

設定オプション = デフォルト値	型	説明
trace_sqlalchemy = False	ブール値	サービスで SQL 要求のプロファイリングを有効にします。 デフォルト値は False です (SQL 要求はトレースされません)。 値: ● True: SQL 要求のプロファイリングを有効にします。各 SQL クエリーはトレースの一部となり、それにどの程度の時間を費やしたかによって分析できます。 ● False: SQL 要求のプロファイリングを無効にします。費やした時間は、高いレベルの操作でのみ表示されます。単一の SQL クエリーをこのように分析できません。

7.1.32. receipt

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[receipt]** グループで使用できるオプションの概要をまとめたものです。

表7.31 receipt

設定オプション = デフォルト値	型	説明
cache_on_issue = True	ブール値	発行した受信データを取得して検証キャッシュを取得し、最初の受信検証で完全な検証サイクルが発生しないようにします。グローバルキャッシングおよび受信キャッシングが有効でない限り、このオプションは機能しません。
cache_time = 300	整数値	受信データおよび検証データをキャッシュする秒数。グローバルと **[receipt] caching** が有効でない限り、効果はありません。
caching = True	ブール値	キャッシュ受信データおよび検証データのキャッシュを切り替えます。これは、グローバルキャッシングが有効でない場合や、問題のキャッシュ受信のみのために cache_on_issue が無効でない限り効果はありません。
expiration = 300	整数値	受信が有効な時間 (秒単位)。この値は、認証メソッドが欠落している場合にユーザーが認証を再試行する必要がある時間を表すため、常に非常に短くする必要があります。

設定オプション = デフォルト値	型	説明
provider = fernet	文字列値	**keystone.receipt.provider** 名前空間の Receipt プロバイダーに対するエントリーポイント。受信プロバイダーは、受信および検証操作を制御します。Keystone には、現時点では **fernet** レシートプロバイダーのみが含まれています。**fernet** レシートを永続化する必要はまったくありませんが、**keystone-manage fernet_setup** を実行する必要があります (**keystone-manage fernet_rotate** コマンドも参照してください)。

7.1.33. resource

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[resource]** グループで使用できるオプションの概要をまとめたものです。

表7.32 resource

設定オプション = デフォルト値	型	説明
admin_project_domain_name = None	文字列値	**admin_project_name** を所有するドメインの名前。未設定の場合、管理プロジェクトはありません。**[resource] admin_project_name** も、このオプションを使用するように設定する必要があります。
admin_project_name = None	文字列値	これは、サービス全体のクラウドレベルの管理者権限を表す特別なプロジェクトです。このプロジェクトにスコープ指定されたトークンには true の **is_admin_project** 属性が含まれ、その特定のプロジェクトのロール割り当てがすべてのプロジェクトで同様に適用する必要があるポリシーシステムを示します。設定されていない場合、管理プロジェクトは存在しないため、プロジェクト間のロール割り当ての明示的な手段はありません。**[resource] admin_project_domain_name** もこのオプションを使用するように設定する必要があります。
cache_time = None	整数値	リソースデータを秒単位でキャッシュする時間 (秒単位)。これは、グローバルキャッシングが有効でない限り効果はありません。
caching = True	ブール値	リソースキャッシュを切り替えます。これは、グローバルキャッシングが有効でない限り効果はありません。

設定オプション = デフォルト値	型	説明
domain_name_url_safe = off	文字列値	これは、ドメインの名前が URL-reserved 文字を含むように制限されるかどうかを制御します。**new** に設定されている場合、ドメインの作成または更新を URL-unsafe 名で試行すると失敗します。**strict** に設定すると、URL-unsafe ドメイン名を持つトークンのスコープ設定に失敗するため、すべてのドメイン名が URL セーフになるよう強制されます。
driver = sql	文字列値	**keystone.resource** 名前空間のリソースドライバーのエントリーポイント。keystone が提供するのは **sql** ドライバーのみです。keystone 用のプロプライエタリードライバーを記述していない限り、このオプションを設定する必要はありません。
list_limit = None	整数値	リソースコレクションで返されるエンティティーの最大数。
project_name_url_safe = off	文字列値	これは、プロジェクトの名前が URL-reserved 文字を含むように制限されるかどうかを制御します。**new** に設定されている場合、URL-unsafe 名でプロジェクトの作成または更新に失敗します。**strict** に設定されている場合、URL 安全でないプロジェクト名を持つトークンのスコープ試行は失敗します。これにより、すべてのプロジェクト名が URL セーフになるよう強制されます。

7.1.34. revoke

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[revoke]** グループで使用できるオプションの概要をまとめたものです。

表7.33 revoke

設定オプション = デフォルト値	型	説明
cache_time = 3600	整数値	失効リストと失効イベントをキャッシュする時間 (秒単位)。これは、グローバルキャッシュと **[revoke] caching** の両方が有効でない限り効果はありません。
caching = True	ブール値	失効イベントキャッシュを切り替えます。これは、グローバルキャッシングが有効でない限り効果はありません。

設定オプション = デフォルト値	型	説明
driver = sql	文字列値	**keystone.revoke** 名前空間のトークン失効バックエンドドライバーのエントリーポイント。Keystone は **sql** ドライバーのみを提供するため、カスタムエントリーポイントを提供しない限り、このオプションを設定する必要はありません。
expiration_buffer = 1800	整数値	対応する失効イベントがバックエンドからパージされるまでのトークンの期限切れ後の秒数。

7.1.35. role

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[role]** グループで使用できるオプションの概要をまとめたものです。

表7.34 role

設定オプション = デフォルト値	型	説明
cache_time = None	整数値	ロールデータのキャッシュ時間 (秒単位)。グローバルキャッシングと **[role]** キャッシュの両方が有効でない限り、これは効果はありません。
caching = True	ブール値	ロールのキャッシュを切り替えます。これは、グローバルキャッシングが有効でない限り効果はありません。一般的なデプロイメントでは、これを無効にする理由はありません。
driver = None	文字列値	**keystone.role** 名前空間内のロールバックエンドドライバーのエントリーポイント。Keystone は **sql** ドライバーのみを提供するため、カスタムエントリーポイントを提供しない限り、この変更の理由はありません。
list_limit = None	整数値	ロールコレクションで返されるエンティティーの最大数。これは、デプロイメントに多数の個別ロールがある場合に調整するのに便利です。

7.1.36. saml

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[saml]** グループで使用できるオプションの概要をまとめたものです。

表7.35 saml

設定オプション = デフォルト値	型	説明
assertion_expiration_time = 3600	整数値	**NotOnOrAfter** 属性を使用して、keystone によって生成される SAML アサーションの有効期間を決定します。
certfile = /etc/keystone/ssl/certs/signing_cert.pem	文字列値	SAML 署名に使用するパブリック証明書ファイルへの絶対パス。値にはコンマ (,) を含めることはできません。
idp_contact_company = Example, Inc.	文字列値	これは、アイデンティティープロバイダーの連絡先個人の会社名です。
idp_contact_email = support@example.com	文字列値	これは、アイデンティティープロバイダーの連絡先ユーザーのメールアドレスです。
idp_contact_name = SAML Identity Provider Support	文字列値	これは、アイデンティティープロバイダーの連絡先人の名前になります。
idp_contact_surname = Support	文字列値	これは、アイデンティティープロバイダーの連絡先ユーザーの名称です。
idp_contact_telephone = +1 800 555 0100	文字列値	これは、アイデンティティープロバイダーの連絡先ユーザーの電話番号です。
idp_contact_type = other	文字列値	これは、アイデンティティープロバイダーの連絡先担当者に最適な連絡先のタイプです。
idp_entity_id = None	URI 値	これは、SAML アサーションを生成するときに使用するアイデンティティープロバイダー (keystone) の一意のエンティティー識別子です。この値はアイデンティティープロバイダーのメタデータを生成するために必要です。この値は URI でなければなりません (URL の使用が推奨されます)。例: **https://keystone.example.com/v3/OS-FEDERATION/saml2/idp**
idp_lang = en	文字列値	これは、アイデンティティープロバイダーの組織によって使用される言語です。
idp_metadata_path = /etc/keystone/saml2_idp_metadata.xml	文字列値	アイデンティティープロバイダーメタデータファイルへの絶対パス。このファイルは **keystone-manage saml_idp_metadata** コマンドで生成する必要があります。通常、この値を変更する理由はありません。
idp_organization_display_name = OpenStack SAML Identity Provider	文字列値	これは、表示するアイデンティティープロバイダーの組織の名前です。

設定オプション = デフォルト値	型	説明
idp_organization_name = SAML Identity Provider	文字列値	これは、アイデンティティープロバイダーの組織の名前です。
idp_organization_url = https://example.com/	URI 値	これは、アイデンティティープロバイダーの組織の URL です。ここで参照される URL は人間にとって役に立ちます。
idp_sso_endpoint = None	URI 値	これは、HTTP POST 要求を受け入れるアイデンティティープロバイダーのシングルサインオン (SSO) サービスの場所です。アイデンティティープロバイダーのメタデータを生成するには、値が必要です。例: **https://keystone.example.com/v3/OS-FEDERATION/saml2/sso**
keyfile = /etc/keystone/ssl/private/signing_key.pem	文字列値	SAML 署名に使用するプライベートキーファイルへの絶対パス。値にはコンマ (**,**) を含めることはできません。
relay_state_prefix = ss:mem:	文字列値	強化されたクライアントおよびプロキシー (ECP) アサーションを生成するときに使用する RelayState SAML 属性の接頭辞。一般的なデプロイメントでは、この値を変更する必要はありません。
xmlsec1_binary = xmlsec1	文字列値	XML 署名に使用するバイナリーの名前、または絶対パス。XML Security Library (**xmlsec1**) のみがサポートされますが、システムに非標準名またはパスが存在する可能性があります。keystone がバイナリー自体を見つけられない場合は、適切なパッケージをインストールするか、このオプションを使用して絶対パスを指定するか、keystone の PATH 環境変数を調整する必要があります。

7.1.37. security_compliance

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[security_compliance]** グループで使用できるオプションの概要をまとめたものです。

表7.36 security_compliance

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
change_password_upon_first_use = False	ブール値	このオプションを有効にすると、ユーザーの作成時や、管理者がパスワードをリセットした場合に、ユーザーによるパスワードの変更が必要となります。サービスにアクセスする前に、影響を受けるユーザーはパスワードを変更する必要があります。サービスユーザーなどの特定ユーザーのこの要件を無視するには、更新ユーザー API で必要なユーザーの **options** 属性 **ignore_change_password_upon_first_use** を **True** に設定します。デフォルトでは無効になっています。この機能は、**[identity] driver** の **sql** バックエンドにのみ適用されます。
disable_user_account_days_inactive = None	整数値	ユーザーが認証なしでアカウントを使用し続けることのできる最大日数。この期間が過ぎるとアカウントは非アクティブと見なされて自動的に無効 (ロック状態) になります。この機能はデフォルトでは無効になっています。有効にするには、任意の値を設定します。この機能は、**[identity] driver** の **sql** バックエンドによって異なります。ユーザーがこのしきい値を超え、inactive とみなされると、HTTP API のユーザーの **enabled** 属性は、ユーザーテーブルのユーザーの **enabled** コラムの値と一致しない場合があります。
lockout_duration = 1800	整数値	**security_compliance] lockout_failure_attempts** で指定されているように、認証失敗回数の最大値を超えた場合に、ユーザーアカウントがロックされる秒数を指定します。**[security_compliance] lockout_failure_attempts** をゼロ以外の値に設定しても、このオプションを設定しても効果はありません。この機能は、**[identity] driver** の **sql** バックエンドによって異なります。
lockout_failure_attempts = None	整数値	**[security_compliance] lockout_duration** で指定された秒数の間、ユーザーアカウントがロックされるまで、ユーザーが認証に失敗できる最大回数。デフォルトでは無効になっています。この機能が有効になり、**[security_compliance] lockout_duration** が設定されていない場合、ユーザーが API 経由で明示的に有効にされるまで、ユーザーは無期限にロックアウトされる可能性があります。この機能は、**[identity] driver** の **sql** バックエンドによって異なります。

設定オプション = デフォルト値	型	説明
minimum_password_age = 0	整数値	ユーザーがパスワードを変更できるようになるまで、そのパスワードを使用する必要のある日数。これは、パスワードの履歴を消去して古いパスワードを再利用するためにユーザーがパスワードを直ちに変更するのを防ぎます。この機能は、管理者がパスワードを手動でリセットするのを防ぐものではありません。これはデフォルトで無効になっており、即時にパスワードの変更を可能にします。この機能は、**[identity] driver** の **sql** バックエンドによって異なります。注意: **security_compliance password_expires_days** が設定されている場合、このオプションの値は **password_expires_days** より小さくする必要があります。
password_expires_days = None	整数値	パスワードが有効と見なされる日数。この期間が過ぎるとパスワードの変更が要求されます。デフォルトでは無効になっています。有効にすると、新しいパスワードの変更には有効期限が設定されますが、既存のパスワードは影響を受けません。この機能は、**[identity] driver** の **sql** バックエンドによって異なります。
password_regex = None	文字列値	パスワードの強度要件の検証に使用される正規表現デフォルトでは、正規表現は任意のパスワードと一致します。次は、少なくとも 1 つの文字と 1 つの数字を必要とし、最小長が 7 文字であるパターンの例です。^(?=.\\d)(?=.[a-zA-Z]).{7,}$。この機能は、**[identity] driver** の **sql** バックエンドに依存します。
password_regex_description = None	文字列値	人が判読できる言語でパスワードの正規表現を記述してください。パスワードが正規表現と一致しない場合、この設定変数の内容はユーザーに返され、要求されたパスワードが不十分な理由が説明されています。
unique_last_password_count = 0	整数値	これにより、新たに作成されたパスワードが一意であることを強制するために、履歴に保管する以前のユーザーパスワードのイテレーション数が制御されます。新しいパスワードを含む合計数はこの値以上にしないでください。値をゼロ (デフォルト) に設定すると、この機能が無効になります。したがって、この機能を有効にするには、値は 0 より大きい必要があります。この機能は、**[identity] driver** の **sql** バックエンドによって異なります。

7.1.38. shadow_users

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[shadow_users]** グループで使用できるオプションの概要をまとめたものです。

表7.37 shadow_users

設定オプション = デフォルト値	型	説明
driver = sql	文字列値	**keystone.identity.shadow_users** 名前空間内のシャドウユーザーバックエンドドライバーのエントリーポイント。このドライバーは、(フェデレーション、LDAP などを介して) 外部で管理されるアイデンティティーへのローカルユーザー参照を保持するために使用されます。Keystone は **sql** ドライバーのみを提供するため、カスタムエントリーポイントを提供しない限り、このオプションを変更する理由はありません。

7.1.39. token

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[token]** グループで使用できるオプションの概要をまとめたものです。

表7.38 token

設定オプション = デフォルト値	型	説明
allow_expired_window = 172800	整数値	これは、組み込みの有効期限を超えたトークンを取得できる秒数を制御します。これにより、長時間実行される操作が正常に実行されることができます。デフォルトは 2 日です。
allow_rescope_scoped_token = True	ブール値	これにより、スコープ付きトークンが新規プロジェクトまたはドメインに再スコープされるかどうかを切り替えて、他のトークンに対してスコープ付きトークン (デフォルトプロジェクトスコープを持つトークンを含む) を変更できないようにします。これにより、ユーザーに非スコープトークンの認証 (さらに特定のスコープを持つトークンに対してその非スコープのトークンを交換) または、スコープ付きトークンにすべての要求で認証情報を指定して再スコープ設定を回避することができます。

設定オプション = デフォルト値	型	説明
cache_on_issue = True	ブール値	発行されたトークンデータをトークン検証キャッシュに保存し、最初のトークン検証によって検証サイクルが完全に発生しないようにします。グローバルキャッシングが有効でない限り、このオプションは効果がなく、**[token] caching = False** であってもトークンをキャッシュします。**非推奨となったバージョン:** S *理由:*Keystone は、トークンのキャッシュオプションをすでに公開しています。発行時にトークンをキャッシュする個別の設定オプションは冗長で、不必要に複雑で、トークンが発行時にデフォルトで事前キャッシュされるため、トークンが無効になっていると誤解を招く可能性があります。トークンが発行時に事前キャッシュされる機能は、今後 ''keystone.conf [token] caching'' オプションにのみ依存します。
cache_time = None	整数値	トークンの作成および検証データをキャッシュする秒数。グローバルと **[token] caching** が有効でない限り、効果はありません。
caching = True	ブール値	トークンの作成および検証データのキャッシュを切り替えます。これは、グローバルキャッシングが有効でない限り効果はありません。
expiration = 3600	整数値	トークンが有効でなければならない期間 (秒単位)。この値を大きく減らすと、複数のサービスを組み合わせて調整する長時間実行される操作が中断される可能性があり、ユーザーは keystone でより頻繁に認証を強制します。この値を大きくすると、同時に有効なトークンの数が増えます。Keystone トークンもベアラートークンであるため、期間は短く、危険にさらされたトークンの潜在的なセキュリティーの影響を軽減します。

設定オプション = デフォルト値	型	説明
provider = fernet	文字列値	**keystone.token.provider** 名前空間内のトークンプロバイダーのエントリーポイント。トークンプロバイダーは、トークンの構築、検証、および失効操作を制御します。サポートされるアップストリームプロバイダーは **fernet** および **jws** です。**fernet** トークンまたは **jws** トークンはいずれも永続性を必要とせず、追加の設定が必要です。**fernet** を使用する場合には、**keystone-manage fernet_setup** を実行する必要があります。これにより、トークンの暗号化に使用する対称鍵が作成されます。**jws** を使用する場合には、トークンに署名および検証するために SHA-256 ハッシュアルゴリズムを使用して ECDSA キーペアを生成する必要があります。これは **keystone-manage create_jws_keypair** で行うことができます。**fernet** トークンは暗号化され、**jws** トークンのみが署名されることに注意してください。お使いのデプロイメントでトークン ID の生成に使用されるペイロードの内容に関するセキュリティー要件がある場合は、必ず検討してください。
revoke_by_id = True	ブール値	これにより、トークン識別子で個別のトークンを取り消すサポートと、さまざまなトークンの列挙操作 (特定のユーザーに発行した全トークンのリスト表示など) がサポートされます。これらの操作は、取り消されるトークンのリストを決定するために使用されます。**kvs [revoke] driver** を使用している場合は、このオプションを無効にしないでください。

7.1.40. tokenless_auth

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[tokenless_auth]** グループで使用できるオプションの概要をまとめたものです。

表7.39 tokenless_auth

設定オプション = デフォルト値	型	説明
issuer_attribute = SSL_CLIENT_I_DN	文字列値	クライアント証明書の発行者を keystone に渡すために使用される WSGI 環境変数の名前。この属性は、X.509 トークンレス承認のアイデンティティープロバイダー ID として使用され、対応するマッピングを検索します。一般的なデプロイメントでは、この値を変更する必要はありません。

設定オプション = デフォルト値	型	説明
protocol = x509	文字列値	X.509 トークンレス承認を表すために使用されるフェデレーションされたプロトコル ID。これは **[tokenless_auth] issuer_attribute** の値と組み合わせて、対応するフェデレーションマッピングを見つけます。一般的なデプロイメントでは、この値を変更する必要はありません。
trusted_issuer = []	多値	X.509 トークンレス認証の使用が許可されるクライアント証明書の信頼できる発行者を識別する識別名のリスト。オプションが存在しない場合、証明書は許可されません。識別名 (DN) の値の形式はコンマで区切り、スペースは使用しないでください。さらに、個別の DN にはコンマが含まれる可能性があるため、このオプションを複数回繰り返して、複数の値を表すことができます。たとえば、keystone.conf には、**trusted_issuer = CN=john,OU=keystone,O=openstack** および **trusted_issuer = CN=mary,OU=eng,O=abc** の 2 つの異なる DN を信頼するために 2 つの連続した行が含まれます。

7.1.41. totp

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[totp]** グループで使用できるオプションの概要をまとめたものです。

表7.40 totp

設定オプション = デフォルト値	型	説明
included_previous_windows = 1	整数値	TOTP の乗車の乗車パスを処理するときに確認する前のウィンドウの数。

7.1.42. 信頼 (trust)

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[trust]** グループで使用できるオプションの概要をまとめたものです。

表7.41 信頼 (trust)

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
allow_redelegation = False	ブール値	あるユーザーから別のユーザーに承認を交換し、信頼を効果的にチェーンできるようにします。無効にすると、信頼の **remaining_uses** 属性はゼロに制限されます。
driver = sql	文字列値	**keystone.trust** 名前空間の信頼バックエンドドライバーのエントリーポイント。Keystone は **sql** ドライバーのみを提供するため、カスタムエントリーポイントを提供しない限り、これを変更する必要はありません。
max_redelegation_count = 3	整数値	あるユーザーから別のユーザーに承認をトラストチェーンで交換できる最大回数。この数は、特定の信頼に対してさらに削減される可能性があります。

7.1.43. unified_limit

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[unified_limit]** グループで使用できるオプションの概要をまとめたものです。

表7.42 unified_limit

設定オプション = デフォルト値	型	説明
cache_time = None	整数値	統一された制限データをキャッシュする時間 (秒単位)。グローバルキャッシングと **[unified_limit]** キャッシュが有効でない限り、効果はありません。
caching = True	ブール値	統一された制限キャッシュに切り替えます。これは、グローバルキャッシングが有効でない限り効果はありません。一般的なデプロイメントでは、これを無効にする理由はありません。
driver = sql	文字列値	**keystone.unified_limit** 名前空間内の統一された制限バックエンドドライバーのエントリーポイント。Keystone は **sql** ドライバーのみを提供するため、カスタムエントリーポイントを提供しない限り、この変更の理由はありません。
enforcement_model = flat	文字列値	プロジェクトに関連する制限を検証するときに使用する実施モデル。適用モデルは、既存の制限に応じて動作が異なります。これにより、実行中のデプロイメントでモデルを切り替えると、後方互換性のない変更が生じる可能性があります。

設定オプション = デフォルト値	型	説明
list_limit = None	整数値	統合制限コレクションで返されるエンティティーの最大数。これは、デプロイメントに統合されている制限が多数ある場合にチューニングに役立つことがあります。

7.1.44. wsgi

以下の表は、**/etc/keystone/keystone.conf** ファイルの **[wsgi]** グループで使用できるオプションの概要をまとめたものです。

表7.43 wsgi

設定オプション = デフォルト値	型	説明
debug_middleware = False	ブール値	true に設定すると、Keystone の oslo デバッグミドルウェアが有効になります。このミドルウェアは、リクエストとレスポンスに関する多くの情報を出力します。ワイヤ上のデータに関する情報 (コーディング済み) を取得し、WSGI アプリケーションパイプラインに渡す場合に便利です。このミドルウェアは、設定ファイルの [DEFAULT] セクションのデバッグ設定や、Keystone のログレベルを DEBUG に設定することや、Keystone が Keystone (特定の要求関連のデータ) のままにしているので、WSGI データのデバッグに固有の設定には影響を与えません。このオプションは、Web サーバー (apache、nginx など) と Keystone 間の要求および応答データのイントロスペクションに使用されます。このミドルウェアは、ミドルウェアチェーンの最初の要素として挿入され、ネットワークに最も近いデータを表示します。警告: 実稼働環境での使用を目的としていません。このミドルウェアは、機密/特権データを生成することができます。

第8章 NEUTRON

以下の章では、**neutron** サービスの設定オプションについて説明します。

8.1. DHCP_AGENT.INI

本項では、**/etc/neutron/dhcp_agent.ini** ファイルのオプションについて説明します。

8.1.1. DEFAULT

以下の表は、**/etc/neutron/dhcp_agent.ini** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
bulk_reload_interval = 0	整数値	DHCP 割り当てのリロード間でスリープ状態になる時間。これは、値が 0 でない場合にのみ呼び出されます。ネットワークに X 秒の N 更新がある場合は、N 回ではなく X 秒でポートが変更された状態で 1 回リロードします。
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
dhcp_broadcast_reply = False	ブール値	DHCP 応答で broadcast を使用します。
dhcp_confs = $state_path/dhcp	文字列値	DHCP サーバー設定ファイルを保存する場所。
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq	文字列値	DHCP サーバーの管理に使用するドライバー。
dhcp_rebinding_time = 0	整数値	DHCP の再バインド時間 T2 (秒単位)。0 に設定すると、デフォルトでリースタイムの 7/8 に設定されます。
dhcp_renewal_time = 0	整数値	DHCP 更新時間 T1 (秒単位)。0 に設定すると、デフォルトでリース時間の半分に設定されます。

設定オプション = デフォルト値	型	説明
dnsmasq_base_log_dir = None	文字列値	dnsmasq ロギングのベースログディレクトリー。ログには DHCP および DNS ログ情報が含まれており、DHCP または DNS のいずれかの問題のデバッグに役立ちます。このセクションが null の場合は、dnsmasq ログを無効にします。
`dnsmasq_config_file = `	文字列値	このファイルでデフォルトの dnsmasq 設定を上書きします。
dnsmasq_dns_servers = []	リスト値	フォワーダーとして使用される DNS サーバーのコンマ区切りのリスト。
dnsmasq_enable_addr6_list = False	ブール値	ポートに同じサブネットに複数の IPv6 アドレスがある場合に、dhcp-host エントリーをアドレスのリストで有効にします。
dnsmasq_lease_max = 16777216	整数値	リースの数を制限して、サービス拒否 (DoS) を防ぎます。
dnsmasq_local_resolv = False	ブール値	dnsmasq サービスが、DHCP エージェントを実行しているホストの DNS リゾルバーを介してインスタンスの名前解決を提供できるようにします。dnsmasq プロセス引数から **--no-resolv** オプションを効果的に削除します。カスタム DNS リゾルバーを **dnsmasq_dns_servers** オプションに追加すると、この機能が無効になります。
enable_isolated_metadata = False	ブール値	DHCP サーバーは、分離されたネットワークでメタデータのサポートを提供するのに役立ちます。この値を True に設定すると、DHCP サーバーは特定のホストルートを DHCP 要求に追加します。メタデータサービスは、サブネットにルーターポートが含まれていない場合にのみアクティベートされます。ゲストインスタンスは、DHCP 経由でホストルートを要求するように設定する必要があります (オプション 121)。force_metadata が True に設定されている場合には、このオプションは有効ではありません。
enable_metadata_network = False	ブール値	CIDR が 169.254.169.254/16 (またはそれ以上の接頭辞) である専用のメタデータアクセスネットワークからのメタデータ要求に対応し、仮想マシンが metadata:1 要求を送信する Neutron ルーターに接続されます。この場合、ルーターを介して 169.254.169.254 に到達できるため、DHCP オプション 121 は仮想マシンにインジェクトされません。このオプションには enable_isolated_metadata = True が必要です。

設定オプション = デフォルト値	型	説明
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
force_metadata = False	ブール値	Neutron ルーターはメタデータ IP を提供するために存在しませんが、この情報を提供するために DHCP サーバーを使用することができます。この値を設定すると、DHCP サーバーが特定のホストルートを DHCP 要求に追加するように強制されます。このオプションを設定すると、すべてのネットワークに対してメタデータサービスがアクティベートされます。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。
interface_driver = None	文字列値	仮想インターフェイスの管理に使用するドライバー。
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。

設定オプション = デフォルト値	型	説明
num_sync_threads = 4	整数値	同期プロセス時に使用するスレッドの数。サーバーに設定された接続プールサイズを超えることはできません。
ovs_integration_bridge = br-int	文字列値	使用する Open vSwitch ブリッジの名前
ovs_use_veth = False	ブール値	OVS インターフェイス用に veth を使用します。ovs_use_veth が True に設定されている場合に、namespace のサポートが制限されたカーネル (例: RHEL 6.5) およびルーターのゲートウェイポートでのレート制限をサポートします。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
resync_interval = 5	整数値	DHCP エージェントは、一時的な通知または RPC エラーから回復するために、その状態を Neutron に再同期します。間隔は、試行するまでの最大秒数です。再同期は、トリガーされたイベントに基づいてより頻繁に実行できます。
resync_throttle = 1	整数値	ローカルの DHCP 状態と Neutron との間の状態イベントの再同期回数を、**resync_throttle** 秒ごとに 1 回のみスロットルします。スロットルの値により、再同期の状態イベントの間隔が最小限に抑えられます。そうでない場合は、再同期がビジーループになる可能性があります。値は resync_interval 未満である必要があります。
rpc_response_max_timeout = 600	整数値	RPC 呼び出しからの応答を待つ最大時間 (秒単位)。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

8.1.2. agent

以下の表は、**/etc/neutron/dhcp_agent.ini** ファイルの **[agent]** グループで使用できるオプションの概要をまとめたものです。

表8.1 agent

設定オプション = デフォルト値	型	説明
availability_zone = nova	文字列値	このノードのアベイラビリティーゾーン
log_agent_heartbeats = False	ブール値	ログエージェントのハートビート

設定オプション = デフォルト値	型	説明
report_interval = 30	浮動小数点の値	状態を server に報告するノード間の秒数。agent_down_time 未満の場合は agent_down_time 未満である必要があります。これは agent_down_time 未満の場合に推奨されます。

8.1.3. ovs

以下の表は、**/etc/neutron/dhcp_agent.ini** ファイルの **[ovs]** グループで使用できるオプションの概要をまとめたものです。

表8.2 ovs

設定オプション = デフォルト値	型	説明
bridge_mac_table_size = 50000	整数値	Neutron OVS エージェントが管理するブリッジについて学習する MAC アドレスの最大数。妥当な範囲外の値 (10 から 1,000,000) は、ドキュメントに従って Open vSwitch で上書きされる可能性があります。
igmp_snooping_enable = False	ブール値	統合ブリッジに IGMP スヌーピングを有効にします。このオプションを True に設定すると、統合ブリッジで Internet Group Management Protocol (IGMP) のサポートが有効になります。このオプションを True に設定すると、Open vSwitch mcast-snooping-disable-flood-unregistered フラグも有効になります。このオプションを使用すると、登録解除されたマルチキャストパケットがすべてのポートにあふれなくなります。スイッチは、未登録のマルチキャストパケットを、マルチキャストルーターに接続されたポートのみに送信します。
ovsdb_connection = tcp:127.0.0.1:6640	文字列値	OVSDB バックエンドの接続文字列。すべての ovsdb コマンドに使用され、監視時に ovsdb-client によって使用されます。
ovsdb_debug = False	ブール値	OVSDB デバッグログの有効化
ovsdb_timeout = 10	整数値	ovsdb コマンドのタイムアウト (秒単位)。タイムアウトの期限が切れると、ovsdb コマンドは ALARMCLOCK エラーで失敗します。
ssl_ca_cert_file = None	文字列値	OVSDB との対話時に使用する認証局 (CA) 証明書。接頭辞 ovsdb_connection を使用する場合は必須
ssl_cert_file = None	文字列値	OVSDB との対話時に使用する SSL 証明書ファイル。接頭辞 ovsdb_connection を使用する場合は必須

設定オプション = デフォルト値	型	説明
ssl_key_file = None	文字列値	OVSDB との対話時に使用する SSL 秘密鍵ファイル。接頭辞 ovsdb_connection を使用する場合は必須

8.2. L3_AGENT.INI

このセクションには、**/etc/neutron/l3_agent.ini** ファイルのオプションが含まれます。

8.2.1. DEFAULT

以下の表は、**/etc/neutron/l3_agent.ini** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
agent_mode = legacy	文字列値	エージェントの作業モード。許可されるモード **legacy**: これにより、L3 エージェントが集中ネットワークノードにデプロイされた既存の動作を保持し、DNAT や SNAT などの L3 サービスが提供されます。DVR を採用したくない場合は、このモードを使用してください。**dvr** - このモードは DVR 機能を有効にし、計算ホストで実行される L3 エージェントに使用する必要があります。**dvr_snat** - これにより、DVR と組み合わせて集中型 SNAT サポートが有効になります。このモードは、中央ノード (または devstack などの単一ホスト展開) で実行されている L3 エージェントに使用する必要があります。**dvr_no_external**: このモードは、コンピュートホストで実行される L3 エージェントの East/West DVR ルーティング機能のみを有効にします。例: DNAT および SNAT などの North/South 機能は、**dvr_snat** モードで実行されている中央ネットワークノードによって提供されます。このモードは、コンピュートホストに外部ネットワーク接続がない場合に使用する必要があります。
api_workers = None	整数値	サービスの個別の API ワーカープロセスの数。指定のない場合は、デフォルトの CPU 数で最高のパフォーマンスを得るために利用可能な CPU の数と同じになり、RAM 使用率は上限になります。

設定オプション = デフォルト値	型	説明
cleanup_on_shutdown = False	ブール値	L3 エージェントのシャットダウン時にすべてのルーターを削除します。L3 HA ルーターの場合、keepalived のシャットダウンと状態変更モニターが含まれます。注記: True に設定すると、L3 エージェントを停止または再起動するときにデータプレーンに影響を与える可能性があります。
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
enable_metadata_proxy = True	ブール値	メタデータプロキシーの実行を許可します。
external_ingress_mark = 0x2	文字列値	外部ネットワークからの ingress マークを示すために使用される iptables の mangle マーク。このマークは 0xffff でマスクされるため、16 ビット未満のビットのみが使用されます。

設定オプション = デフォルト値	型	説明
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
ha_confs_path = $state_path/ha_confs	文字列値	keepalived 設定ファイルを保存する場所
ha_keepalived_state_change_server_threads = <based on operating system>	整数値	keepalived サーバー接続要求の同時スレッドの数。スレッドを増やすと、エージェントノードに CPU 負荷が高くなります。
ha_vrrp_advert_int = 2	整数値	広告の間隔 (秒単位)
ha_vrrp_auth_password = None	文字列値	VRRP 認証パスワード
ha_vrrp_auth_type = PASS	文字列値	VRRP 認証タイプ
ha_vrrp_garp_master_delay = 5	整数値	MASTER の場合に、優先度の低い広告後の Gratuitous ARP の 2 番目のセットの遅延。注記: この設定オプションは OSP13 および OSP16 でのみ利用できます。今後のリリースでは、keepalived 設定を提供するためにテンプレート形式が実装されます。
ha_vrrp_garp_master_repeat = 5	整数値	MASTER に移行後に一度に送信する余計な ARP メッセージの数。注記: この設定オプションは OSP13 および OSP16 でのみ利用できます。今後のリリースでは、keepalived 設定を提供するためにテンプレート形式が実装されます。
ha_vrrp_health_check_interval = 0	整数値	VRRP ヘルスチェックの間隔 (秒単位)。値 > 0 は VRRP ヘルスチェックを有効にします。これを 0 に設定すると、VRRP ヘルスチェックが無効になります。推奨される値は 5 です。これにより、ping がゲートウェイの IP アドレスに送信されます。ゲートウェイで ICMP_ECHO_REQUEST を有効にする必要があります。ゲートウェイに障害が発生した場合、すべてのルーターがプライマリーとして報告され、ルーターの 1 つがゲートウェイ接続を復元するまで、ラウンドロビン方式でプライマリー選択が繰り返されます。

設定オプション = デフォルト値	型	説明
handle_internal_only_routers = True	ブール値	この L3 エージェントは、外部ネットワークゲートウェイが設定されていないルーターも処理する必要があることを示します。このオプションは、Neutron デプロイメントにある単一のエージェントに対してのみ True にする必要があります。すべてのルーターが外部ネットワークゲートウェイを持つ必要がある場合には、すべてのエージェントに対して False になる可能性があります。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。
interface_driver = None	文字列値	仮想インターフェイスの管理に使用するドライバー。
`ipv6_gateway = `	文字列値	IPv6 では、自動的に割り当てられたリンクローカルアドレス (LLA) を使用できるため、外部ゲートウェイに使用されるネットワークにはサブネットが関連付けられる必要はありません。ただし、IPv6 ゲートウェイアドレスは、デフォルトルートの次ホップとして使用するために必要です。ここで IPv6 ゲートウェイアドレスが設定されていない場合には、neutron ルーターは、アップストリームルーターからルーター広告 (RA) からデフォルトルートを取得するように設定されます。この場合、アップストリームルーターはこれらの RA を送信するようにも設定する必要があります。ipv6_gateway が設定されている場合は、アップストリームルーターのインターフェイスの LLA である必要があります。グローバル一意のアドレス (GUA) を使用するネクストホップが必要な場合は、このパラメーターではなく、ネットワークに割り当てられたサブネット経由で実行する必要があります。
keepalived_use_no_track = True	ブール値	no_track オプションのサポートのない keepalived が使用される場合は、このパラメーターを False に設定する必要があります。このオプションのサポートは keepalived 2.x で導入されました。

設定オプション = デフォルト値	型	説明
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
max_rtr_adv_interval = 100	整数値	radvd.conf の MaxRtrAdvInterval 設定
metadata_access_mark = 0x1	文字列値	メタデータの有効な要求をマークするために使用される iptables の mangle マーク。このマークは 0xffff でマスクされるため、16 ビット未満のビットのみが使用されます。
metadata_port = 9697	ポート値	Neutron メタデータ名前空間プロキシーが使用する TCP ポート。
min_rtr_adv_interval = 30	整数値	MinRtrAdvInterval setting for radvd.conf
ovs_integration_bridge = br-int	文字列値	使用する Open vSwitch ブリッジの名前
ovs_use_veth = False	ブール値	OVS インターフェイス用に veth を使用します。ovs_use_veth が True に設定されている場合に、namespace のサポートが制限されたカーネル (例: RHEL 6.5) およびルーターのゲートウェイポートでのレート制限をサポートします。

設定オプション = デフォルト値	型	説明
pd_confs = $state_path/pd	文字列値	IPv6 PD ファイルを保存する場所。
periodic_fuzzy_delay = 5	整数値	タイムスタンプを減らすために定期的なタスクスケジューラーを開始する際の無作為に遅延する秒数。(0 に設定すると無効になります)
periodic_interval = 40	整数値	実行中の定期的なタスク間の秒数。
prefix_delegation_driver = dibbler	文字列値	ipv6 接頭辞委任に使用されるドライバー。これは、neutron.agent.linux.pd_drivers 名前空間で定義されているエントリーポイントでなければなりません。neutron ソースに含まれるエントリーポイントは、setup.cfg を参照してください。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
ra_confs = $state_path/ra	文字列値	IPv6 RA 設定ファイルを保存する場所
`radvd_user = `	文字列値	radvd に渡されるユーザー名。root 権限をドロップし、ユーザー ID を username および group ID を、ユーザー名のプライマリーグループに変更します。ユーザーの指定がない場合 (デフォルト) では、L3 エージェントを実行しているユーザーは渡されます。root を指定すると、radvd が root として生成されるため、username パラメーターは渡されません。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
rpc_response_max_timeout = 600	整数値	RPC 呼び出しからの応答を待つ最大時間 (秒単位)。
rpc_state_report_workers = 1	整数値	状態レポートキュー専用の RPC ワーカープロセスの数。

設定オプション = デフォルト値	型	説明
rpc_workers = None	整数値	サービスの RPC ワーカープロセスの数。指定されていない場合、デフォルトは API ワーカー数の半分に等しくなります。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
vendor_pen = 8888	文字列値	RFC3315 DUID-EN で必要な Vendor's Registered Private Enterprise Number として 10 進数の値。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

8.2.2. agent

以下の表は、**/etc/neutron/l3_agent.ini** ファイルの **[agent]** グループで使用できるオプションの概要をまとめたものです。

表8.3 agent

設定オプション = デフォルト値	型	説明
availability_zone = nova	文字列値	このノードのアベイラビリティーゾーン
extensions = []	リスト値	使用するエクステンションリスト
log_agent_heartbeats = False	ブール値	ログエージェントのハートビート
report_interval = 30	浮動小数点の値	状態を server に報告するノード間の秒数。agent_down_time 未満の場合は agent_down_time 未満である必要があります。これは agent_down_time 未満の場合に推奨されます。

8.2.3. network_log

以下の表は、**/etc/neutron/l3_agent.ini** ファイルの **[network_log]** グループで使用できるオプションの概要をまとめたものです。

表8.4 network_log

設定オプション = デフォルト値	型	説明
burst_limit = 25	整数値	rate_limit あたりの最大パケット数。
local_output_log_base = None	文字列値	エージェント側の出力ログファイルパス、デフォルトの syslog ファイル。
rate_limit = 100	整数値	1 秒あたりの最大パケットロギング。

8.2.4. ovs

以下の表は、**/etc/neutron/l3_agent.ini** ファイルの **[ovs]** グループで使用できるオプションの概要をまとめたものです。

表8.5 ovs

設定オプション = デフォルト値	型	説明
bridge_mac_table_size = 50000	整数値	Neutron OVS エージェントが管理するブリッジについて学習する MAC アドレスの最大数。妥当な範囲外の値 (10 から 1,000,000) は、ドキュメントに従って Open vSwitch で上書きされる可能性があります。

設定オプション = デフォルト値	型	説明
igmp_snooping_enable = False	ブール値	統合ブリッジに IGMP スヌーピングを有効にします。このオプションを True に設定すると、統合ブリッジで Internet Group Management Protocol (IGMP) のサポートが有効になります。このオプションを True に設定すると、Open vSwitch mcast-snooping-disable-flood-unregistered フラグも有効になります。このオプションを使用すると、登録解除されたマルチキャストパケットがすべてのポートにあふれなくなります。スイッチは、未登録のマルチキャストパケットを、マルチキャストルーターに接続されたポートのみに送信します。
ovsdb_connection = tcp:127.0.0.1:6640	文字列値	OVSDB バックエンドの接続文字列。すべての ovsdb コマンドに使用され、監視時に ovsdb-client によって使用されます。
ovsdb_debug = False	ブール値	OVSDB デバッグログの有効化
ovsdb_timeout = 10	整数値	ovsdb コマンドのタイムアウト (秒単位)。タイムアウトの期限が切れると、ovsdb コマンドは ALARMCLOCK エラーで失敗します。
ssl_ca_cert_file = None	文字列値	OVSDB との対話時に使用する認証局 (CA) 証明書。接頭辞 ovsdb_connection を使用する場合は必須
ssl_cert_file = None	文字列値	OVSDB との対話時に使用する SSL 証明書ファイル。接頭辞 ovsdb_connection を使用する場合は必須
ssl_key_file = None	文字列値	OVSDB との対話時に使用する SSL 秘密鍵ファイル。接頭辞 ovsdb_connection を使用する場合は必須

8.3. LINUXBRIDGE_AGENT.INI

本項では、**/etc/neutron/plugins/ml2/linuxbridge_agent.ini** ファイルのオプションについて説明します。

8.3.1. DEFAULT

以下の表は、**/etc/neutron/plugins/ml2/linuxbridge_agent.ini** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。

設定オプション = デフォルト値	型	説明
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
rpc_response_max_timeout = 600	整数値	RPC 呼び出しからの応答を待つ最大時間 (秒単位)。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

8.3.2. agent

以下の表は、**/etc/neutron/plugins/ml2/linuxbridge_agent.ini** ファイルの **[agent]** グループで使用できるオプションの概要をまとめたものです。

表8.6 agent

設定オプション = デフォルト値	型	説明
dscp = None	整数値	トンネルカプセル化時に外部ヘッダーに使用する DSCP 値。

設定オプション = デフォルト値	型	説明
dscp_inherit = False	ブール値	True に設定すると、トンネルインターフェイスの DSCP 値が上書きされ、継承 に設定されます。内部ヘッダーの DSCP 値は、外側のヘッダーにコピーされます。
extensions = []	リスト値	使用するエクステンションリスト
polling_interval = 2	整数値	ローカルデバイスの変更をポーリングするまでエージェントが待機する秒数。
quitting_rpc_timeout = 10	整数値	エージェントが SIGTERM を受信した後に、新しい rpc 呼び出しに新しいタイムアウトを秒単位で設定します。値を 0 に設定すると、rpc-timeout は変更されない

8.3.3. linux_bridge

以下の表は、**/etc/neutron/plugins/ml2/linuxbridge_agent.ini** ファイルの **[linux_bridge]** グループで使用できるオプションの概要をまとめたものです。

表8.7 linux_bridge

設定オプション = デフォルト値	型	説明
bridge_mappings = []	リスト値	<physical_network>:<physical_bridge> のリスト
physical_interface_mappings = []	リスト値	物理ネットワーク名を、フラットネットワークおよび VLAN ネットワークに使用するエージェントのノード固有の物理ネットワークインターフェイスにマッピングする <physical_network>:<physical_interface> タプルのコンマ区切りリスト。サーバーの network_vlan_ranges にリスト表示されるすべての物理ネットワークには、各エージェントの適切なインターフェイスへのマッピングが必要です。

8.3.4. network_log

以下の表は、**/etc/neutron/plugins/ml2/linuxbridge_agent.ini** ファイルの **[network_log]** グループで使用できるオプションの概要をまとめたものです。

表8.8 network_log

設定オプション = デフォルト値	型	説明
burst_limit = 25	整数値	rate_limit あたりの最大パケット数。
local_output_log_base = None	文字列値	エージェント側の出力ログファイルパス、デフォルトの syslog ファイル。
rate_limit = 100	整数値	1 秒あたりの最大パケットロギング。

8.3.5. securitygroup

以下の表は、**/etc/neutron/plugins/ml2/linuxbridge_agent.ini** ファイルの **[securitygroup]** グループで使用できるオプションの概要をまとめたものです。

表8.9 securitygroup

設定オプション = デフォルト値	型	説明
enable_ipset = True	ブール値	ipset を使用して、iptables ベースのセキュリティーグループをスピードアップします。ipset サポートを有効にするには、ipset が L2 エージェントノードにインストールする必要があります。
enable_security_group = True	ブール値	サーバーで neutron セキュリティーグループ API を有効にするかどうかを制御します。セキュリティーグループを使用しない場合や、nova セキュリティーグループ API を使用する場合は false にする必要があります。
firewall_driver = None	文字列値	L2 エージェントのセキュリティーグループファイアウォール用のドライバー
permitted_ethertypes = []	リスト値	16 進数 (0x で始まる) で、許可されるイーサネットタイプのコンマ区切りリスト。たとえば、"0x4008" は InfiniBand を許可します。

8.3.6. vxlan

以下の表は、**/etc/neutron/plugins/ml2/linuxbridge_agent.ini** ファイルの **[vxlan]** グループで使用できるオプションの概要をまとめたものです。

表8.10 vxlan

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
arp_responder = False	ブール値	オーバーレイへの ARP ブロードキャストを実行する代わりに、ローカル応答を提供するローカル ARP レスポンダーを有効にします。ローカル ARP レスポンダーの有効化は、allow-address-pairs 拡張と完全に互換性がありません。
enable_vxlan = True	ブール値	エージェントで VXLAN を有効にします。linuxbridge メカニズムドライバーを使用する ml2 プラグインでエージェントを管理する場合に有効にできます。
l2_population = False	ブール値	ml2 プラグインの l2population メカニズムドライバーと共に使用する拡張機能。これにより、プラグインが VXLAN 転送テーブルに設定できるようになります。
local_ip = None	IP アドレス値	ローカルオーバーレイ (tunnel) ネットワークエンドポイントの IP アドレス。ホストネットワークインターフェイスのいずれかにある IPv4 アドレスまたは IPv6 アドレスを使用します。この値の IP バージョンは、neutron サーバーノード上の ML2 プラグイン設定ファイルの **overlay_ip_version** オプションの値と一致する必要があります。
multicast_ranges = []	リスト値	VNI ID に従ってマルチキャストアドレスを VXLAN に割り当てる方法を説明する <multicast address>:<vni_min>:<vni_max> トリプルのコンマ区切りリスト (任意)。
tos = None	整数値	vxlan インターフェイスプロトコルパケット用の TOS。このオプションは、AGENT セクションの dscp オプションが優先されるため非推奨になり、今後のリリースで削除される予定です。TOS の値を DSCP に変換するには、4 で割ります。
ttl = None	整数値	vxlan インターフェイスプロトコルパケットの TTL。
udp_dstport = None	ポート値	VXLAN 通信に使用される UDP ポート。デフォルトでは、Linux カーネルは IANA が割り当てられた標準値を使用しないため、これを使用する場合は、このオプションを 4789 に設定する必要があります。後方互換のために、これはデフォルトでは設定されません。
udp_srcport_max = 0	ポート値	VXLAN 通信に使用される UDP ソースポート範囲の最大値。

設定オプション = デフォルト値	型	説明
udp_srcport_min = 0	ポート値	VXLAN 通信に使用される UDP ソースポート範囲の最小数。
vxlan_group = 224.0.0.1	文字列値	vxlan インターフェイスのマルチキャストグループ。グループアドレスの範囲は、CIDR 表記を使用して指定できます。範囲を指定すると、異なる VNI で異なるグループアドレスを使用でき、トンネルエンドポイントへの誤ったブロードキャストトラフィックを増減できます。可能な限り (24 ビット) VNI に一意のグループを確保するには、239.0.0.0/8 などの /8 を使用します。この設定は、すべてのエージェントで同じである必要があります。

8.4. METADATA_AGENT.INI

本項では、**/etc/neutron/metadata_agent.ini** ファイルのオプションについて説明します。

8.4.1. DEFAULT

以下の表は、**/etc/neutron/metadata_agent.ini** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
auth_ca_cert = None	文字列値	ssl の認証局公開鍵 (CA 証明書) ファイル
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。

設定オプション = デフォルト値	型	説明
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
metadata_backlog = 4096	整数値	メタデータサーバーソケットを設定するためのバックログ要求の数
`metadata_proxy_group = `	文字列値	グループ (gid または名前) は、初期化後にメタデータプロキシーを実行します (空の場合: agent effective group)。
`metadata_proxy_shared_secret = `	文字列値	メタデータ要求をプロキシー処理する場合、Neutron は、スプーフィングを防ぐために、共有シークレットで Instance-ID ヘッダーに署名します。シークレットの文字列を選択することはできますが、ここでは同じ文字列と、Nova Metadata Server が使用する設定で一致させる必要があります。注意: Nova は同じ config キーを使用しますが、[neutron] セクションで行います。
metadata_proxy_socket = $state_path/metadata_proxy	文字列値	メタデータプロキシー UNIX ドメインソケットの場所。

設定オプション = デフォルト値	型	説明
metadata_proxy_socket_mode = deduce	文字列値	メタデータプロクシ UNIX ドメインソケットモード、4 つの値が使用可能:**deduce**: metadata_proxy_user/group の値からモードを推測,**user**: メタデータプロキシーソケットモードを 0o644 に設定,metadata_proxy_user がエージェントの実効ユーザーまたはルート時に使用、**group**: メタデータプロキシーソケットモードを 0o664 に設定、metadata_proxy_group がエージェントの実効グループまたはルートの時に使用、**all**: メタデータプロキシーソケットモードを 0o666 に 設定、その他の場合に使用。
`metadata_proxy_user = `	文字列値	ユーザー (uid または名前) の初期化後にメタデータプロキシーを実行する (空の場合: agent effective user)。
metadata_workers = <オペレーティングシステムに基づく>	整数値	メタデータサーバーの個別のワーカープロセスの数 (ML2/OVN で使用する場合はデフォルトで 2、他のバックエンドドライバーでは CPU 数の半分)
`nova_client_cert = `	文字列値	nova metadata api サーバーのクライアント証明書。
`nova_client_priv_key = `	文字列値	クライアント証明書の秘密鍵。
nova_metadata_host = 127.0.0.1	ホストのアドレス値	Nova メタデータサーバーの IP アドレスまたは DNS 名。
nova_metadata_insecure = False	ブール値	nova メタデータに対してセキュアではない SSL (https) 要求を実行できるようにする
nova_metadata_port = 8775	ポート値	Nova メタデータサーバーが使用する TCP ポート。
nova_metadata_protocol = http	文字列値	nova メタデータ、http、または https にアクセスするプロトコル
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。

設定オプション = デフォルト値	型	説明
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
rpc_response_max_timeout = 600	整数値	RPC 呼び出しからの応答を待つ最大時間 (秒単位)。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

8.4.2. agent

以下の表は、**/etc/neutron/metadata_agent.ini** ファイルの **[agent]** グループで使用できるオプションの概要をまとめたものです。

表8.11 agent

設定オプション = デフォルト値	型	説明
log_agent_heartbeats = False	ブール値	ログエージェントのハートビート
report_interval = 30	浮動小数点の値	状態を server に報告するノード間の秒数。agent_down_time 未満の場合は agent_down_time 未満である必要があります。これは agent_down_time 未満の場合に推奨されます。

8.4.3. cache

以下の表は、**/etc/neutron/metadata_agent.ini** ファイルの **[cache]** グループで使用できるオプションの概要をまとめたものです。

表8.12 cache

設定オプション = デフォルト値	型	説明
backend = dogpile.cache.null	文字列値	キャッシュバックエンドモジュール。数百のスレッドサーバーを持つイベントレットベースまたは環境の場合は、プーリングを含む Memcache (oslo_cache.memcache_pool) が推奨されます。100 台未満のスレッドサーバーがある環境では、Memcached(dogpile.cache.memcached) または Redis(dogpile.cache.redis) が推奨されます。サーバーの単一のインスタンスを含むテスト環境は、dogpile.cache.memory バックエンドを使用できます。
backend_argument = []	多値	バックエンドモジュールに指定された引数。このオプションは、dogpile.cache バックエンドに渡される引数ごとに 1 回指定します。例: "<argname>:<value>"
config_prefix = cache.oslo	文字列値	キャッシュリージョンの設定ディクショナリーを構築するための接頭辞。同じ設定名を持つ別の dogpile.cache リージョンがない場合は、これを変更する必要はありません。
dead_timeout = 60	浮動小数点の値	HashClient の内部メカニズムでプールにノードを再度追加しようとするまでの時間 (秒)。
debug_cache_backend = False	ブール値	キャッシュバックエンドからの追加デバッグ (キャッシュキー、get/set/delete/etc コール)。これは、キー/値を含む特定の cache-backend get/set/delete 呼び出しを確認する必要がある場合にのみ非常に便利です。通常、これは false に設定されている必要があります。

設定オプション = デフォルト値	型	説明
enable_retry_client = False	ブール値	失敗を処理する再試行クライアントメカニズムを有効にします。これらのメカニズムは、あらゆる種類の pymemcache クライアントをラップするために使用できます。ラッパーを使用すると、試行回数と attemot 間の待機時間を定義できます。
enable_socket_keepalive = False	ブール値	dogpile の pymemcache バックエンドのソケットキープアライブのグローバルトグル
enabled = False	ブール値	キャッシュのグローバル切り替え。
expiration_time = 600	整数値	dogpile.cache リージョンのキャッシュされた項目に対するデフォルトの TTL (秒単位)。これは、キャッシュの有効期限が明示的に定義されていないキャッシュされたメソッドに適用されます。
hashclient_retry_attempts = 2	整数値	HashClient の内部メカニズムでクライアントが無効としてマークされ、プールから削除されるまで、クライアントを試行する必要がある回数。
hashclient_retry_delay = 1	浮動小数点の値	HashClient の内部メカニズムでの再試行の間に経過する時間 (秒単位)。
memcache_dead_retry = 300	整数値	再試行するまで memcached サーバーが停止されているとみなされる秒数 (dogpile.cache.memcache および oslo_cache.memcache_pool バックエンドのみ)。
memcache_pool_connection_get_timeout = 10	整数値	操作が memcache クライアント接続を取得するのを待つ秒数。
memcache_pool_flush_on_reconnect = False	ブール値	再接続時に memcache がフラッシュされるかどうかをグローバルに切り替えます (oslo_cache.memcache_pool バックエンドのみ)。
memcache_pool_maxsize = 10	整数値	すべての memcached サーバーへのオープン接続の最大数 (oslo_cache.memcache_pool バックエンドのみ)。
memcache_pool_unused_timeout = 60	整数値	memcached への接続が閉じられる前にプールで未使用のままになる秒数 (oslo_cache.memcache_pool バックエンドのみ)。

設定オプション = デフォルト値	型	説明
memcache_servers = ['localhost:11211']	リスト値	ホスト: ポートの形式の Memcache サーバー (dogpile.cache.memcached および oslo_cache.memcache_pool バックエンドのみ)。特定のホストが IPv6 を参照する場合、または特定のドメインが IPv6 を参照する場合、特定のアドレスの前にアドレスファミリー (**inet6**) を付ける必要があります (例: **inet6[::1]:11211**、**inet6:[fd12:3456:789a:1::1]:11211**、**inet6:[controller-0.internalapi]:11211**)。アドレスファミリーが指定されていない場合、使用されるデフォルトのアドレスファミリーは、IPv4 に対応する **inet** になります。
memcache_socket_timeout = 1.0	浮動小数点の値	サーバーへの呼び出しごとにタイムアウト (秒単位)。(dogpile.cache.memcache および oslo_cache.memcache_pool バックエンドのみ)。
proxies = []	リスト値	dogpile.cache バックエンドの機能に影響を与えるプロキシークラス。changing-backend-behavior についての dogpile.cache ドキュメントを参照してください。
retry_attempts = 2	整数値	失敗するまでアクションを試行する回数。
retry_delay = 0	浮動小数点の値	各試行の間にスリープする秒数。
socket_keepalive_count = 1	整数値	接続を切断する前に TCP が送信するキープアライブプローブの最大数。ゼロより大きい正の整数である必要があります。
socket_keepalive_idle = 1	整数値	TCP がキープアライブプローブの送信を開始する前に、接続がアイドル状態を維持する必要がある時間 (秒単位)。ゼロより大きい正の整数である必要があります。
socket_keepalive_interval = 1	整数値	個々のキープアライブプローブ間の時間 (秒単位)。ゼロより大きい正の整数である必要があります。
tls_allowed_ciphers = None	文字列値	TLS コンテキストで作成されたソケットで利用可能な暗号を設定します。OpenSSL 暗号リスト形式の文字列である必要があります。指定されていない場合は、OpenSSL 対応の暗号がすべて利用可能になります。

設定オプション = デフォルト値	型	説明
tls_cafile = None	文字列値	キャッシュサーバーの信頼性を確立するのに必要な PEM 形式の連結された CA 証明書のファイルへのパス。tls_enabled が False の場合、このオプションは無視されます。
tls_certfile = None	文字列値	クライアントの証明書と証明書の信頼性を確立するために必要な任意の数の CA 証明書を含む PEM 形式の単一ファイルへのパス。このファイルは、クライアント側の認証が必要な場合にのみ必要です。tls_enabled が False の場合、このオプションは無視されます。
tls_enabled = False	ブール値	キャッシングサーバーと連携する際に、TLS の使用をグローバルに切り替えます。
tls_keyfile = None	文字列値	クライアントの秘密鍵を含む単一ファイルへのパス。それ以外の場合、秘密鍵は tls_certfile で指定されたファイルから取得されます。tls_enabled が False の場合、このオプションは無視されます。

8.5. METERING_AGENT.INI

このセクションには、**/etc/neutron/metering_agent.ini** ファイルのオプションが含まれます。

8.5.1. DEFAULT

以下の表では、**/etc/neutron/metering_agent.ini** ファイルの **[DEFAULT]** グループで利用可能なオプションについて説明しています。

.

設定オプション = デフォルト値	型	説明
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
driver = neutron.services.metering.drivers.noop.noop_driver.NoopMeteringDriver	文字列値	メータリングドライバー
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
granular_traffic_data = False	ブール値	ラベルが割り当てられたすべてのプロジェクトとルーターのすべてのトラフィックデータをグループ化するのではなく、メータリングエージェントドライバーがトラフィックデータを詳細な方法で表示するかどうかを定義します。デフォルト値は下位互換性のために **False** です。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。

設定オプション = デフォルト値	型	説明
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。
interface_driver = None	文字列値	仮想インターフェイスの管理に使用するドライバー。
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。

設定オプション = デフォルト値	型	説明
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
measure_interval = 30	整数値	2 つのメータリング計測値の間隔
ovs_integration_bridge = br-int	文字列値	使用する Open vSwitch ブリッジの名前

設定オプション = デフォルト値	型	説明
ovs_use_veth = False	ブール値	OVS インターフェイス用に veth を使用します。ovs_use_veth が True に設定されている場合に、namespace のサポートが制限されたカーネル (例: RHEL 6.5) およびルーターのゲートウェイポートでのレート制限をサポートします。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
report_interval = 300	整数値	2 つのメータリングレポートの間隔
rpc_response_max_timeout = 600	整数値	RPC 呼び出しからの応答を待つ最大時間 (秒単位)。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。

設定オプション = デフォルト値	型	説明
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

8.5.2. agent

以下の表では、**/etc/neutron/metering_agent.ini** ファイルの **[agent]** グループで利用可能なオプションについて説明しています。

表8.13 agent

設定オプション = デフォルト値	型	説明
log_agent_heartbeats = False	ブール値	ログエージェントのハートビート
report_interval = 30	浮動小数点の値	状態を server に報告するノード間の秒数。agent_down_time 未満の場合は agent_down_time 未満である必要があります。これは agent_down_time 未満の場合に推奨されます。

8.5.3. ovs

以下の表では、**/etc/neutron/metering_agent.ini** ファイルの **[ovs]** グループで利用可能なオプションについて説明しています。

表8.14 ovs

設定オプション = デフォルト値	型	説明
bridge_mac_table_size = 50000	整数値	Neutron OVS エージェントが管理するブリッジについて学習する MAC アドレスの最大数。妥当な範囲外の値 (10 から 1,000,000) は、ドキュメントに従って Open vSwitch で上書きされる可能性があります。

設定オプション = デフォルト値	型	説明
igmp_snooping_enable = False	ブール値	統合ブリッジに IGMP スヌーピングを有効にします。このオプションを True に設定すると、統合ブリッジで Internet Group Management Protocol (IGMP) のサポートが有効になります。このオプションを True に設定すると、Open vSwitch mcast-snooping-disable-flood-unregistered フラグも有効になります。このオプションを使用すると、登録解除されたマルチキャストパケットがすべてのポートにあふれなくなります。スイッチは、未登録のマルチキャストパケットを、マルチキャストルーターに接続されたポートのみに送信します。
ovsdb_connection = tcp:127.0.0.1:6640	文字列値	OVSDB バックエンドの接続文字列。すべての ovsdb コマンドに使用され、監視時に ovsdb-client によって使用されます。
ovsdb_debug = False	ブール値	OVSDB デバッグログの有効化
ovsdb_timeout = 10	整数値	ovsdb コマンドのタイムアウト (秒単位)。タイムアウトの期限が切れると、ovsdb コマンドは ALARMCLOCK エラーで失敗します。
ssl_ca_cert_file = None	文字列値	OVSDB との対話時に使用する認証局 (CA) 証明書。接頭辞 ovsdb_connection を使用する場合は必須
ssl_cert_file = None	文字列値	OVSDB との対話時に使用する SSL 証明書ファイル。接頭辞 ovsdb_connection を使用する場合は必須
ssl_key_file = None	文字列値	OVSDB との対話時に使用する SSL 秘密鍵ファイル。接頭辞 ovsdb_connection を使用する場合は必須

8.6. ML2_CONF.INI

本項では、**/etc/neutron/plugins/ml2/ml2_conf.ini** ファイルのオプションについて説明します。

8.6.1. DEFAULT

以下の表は、**/etc/neutron/plugins/ml2/ml2_conf.ini** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。

設定オプション = デフォルト値	型	説明
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

8.6.2. ml2

以下の表は、**/etc/neutron/plugins/ ml2/ml2_conf.ini** ファイルの **[ml2]** グループで使用できるオプションの概要をまとめたものです。

表8.15 ml2

設定オプション = デフォルト値	型	説明
extension_drivers = []	リスト値	neutron.ml2.extension_drivers 名前空間から読み込まれる拡張ドライバーのエントリーポイントの順序付きリスト。例: extension_drivers = port_security,qos

設定オプション = デフォルト値	型	説明
external_network_type = None	文字列値	プロバイダー属性が指定されていない場合の外部ネットワークのデフォルトネットワークタイプ。デフォルトでは None です。これは、外部ネットワークの作成時にプロバイダー属性が指定されていない場合には、テナントネットワークと同じタイプを持ちます。external_network_type 設定オプションに許可される値は、type_drivers 設定オプションで設定されたネットワーク種別の値によって異なります。
mechanism_drivers = []	リスト値	neutron.ml2.mechanism_drivers 名前空間から読み込まれる認証メカニズムドライバーのエントリーポイントの順序付きリスト。
overlay_ip_version = 4	整数値	すべてのオーバーレイ (tunnel) ネットワークエンドポイントの IP バージョン。IPv4 には 4、IPv6 には 6 の値を使用します。
path_mtu = 0	整数値	オーバーレイ/トンネルプロトコルを使用する場合に断片化せずに基礎となる物理ネットワークインフラストラクチャーを通過できる IP パケット (MTU) の最大サイズ。このオプションを使用すると、デフォルトの global_physnet_mtu 値とは異なる物理ネットワーク MTU 値を指定できます。
physical_network_mtus = []	リスト値	物理ネットワークと MTU 値のマッピングリスト。マッピングのフォーマットは <physnet>:<mtu val> です。このマッピングにより、デフォルトの global_physnet_mtu 値とは異なる物理ネットワーク MTU 値を指定できます。
tenant_network_types = ['local']	リスト値	テナントネットワークとして割り当てる network_types の順序付きリスト。デフォルト値の **local** は単一ボックスのテストに役立ちますが、ホスト間の接続を提供しません。
type_drivers = ['local', 'flat', 'vlan', 'gre', 'vxlan', 'geneve']	リスト値	neutron.ml2.type_drivers 名前空間から読み込まれるネットワーク種別ドライバーのエントリーポイントのリスト。

8.6.3. ml2_type_flat

以下の表は、**/etc/neutron/plugins/ml2/ml2_conf.ini** ファイルの **[ml2_type_flat]** グループで使用できるオプションの概要をまとめたものです。

表8.16 ml2_type_flat

設定オプション = デフォルト値	型	説明
flat_networks = *	リスト値	フラットネットワークを作成できる physical_network 名のリストデフォルトの * を使用して、任意の physical_network 名を持つフラットネットワークを許可します。空のリストを使用してフラットネットワークを無効にします。

8.6.4. ml2_type_geneve

以下の表は、**/etc/neutron/plugins/ ml2/ml2_conf.ini** ファイルの **[ml2_type_geneve]** グループで使用できるオプションの概要をまとめたものです。

表8.17 ml2_type_geneve

設定オプション = デフォルト値	型	説明
max_header_size = 30	整数値	Geneve カプセル化ヘッダーサイズは動的であり、この値はドライバーの最大 MTU を計算するために使用されます。このフィールドのデフォルトサイズは 30 です。これは、追加のオプションヘッダーを含まない Geneve ヘッダーのサイズです。
vni_ranges = []	リスト値	テナントネットワークの割り当てに使用できる Geneve VNI ID の範囲を列挙した <vni_min>:<vni_max> タプルのコンマ区切りリスト

8.6.5. ml2_type_gre

以下の表は、**/etc/neutron/plugins/ml2/ml2_conf.ini** ファイルの **[ml2_type_gre]** グループで使用できるオプションの概要をまとめたものです。

表8.18 ml2_type_gre

設定オプション = デフォルト値	型	説明
tunnel_id_ranges = []	リスト値	テナントネットワークの割り当てに使用できる GRE トンネル ID の範囲を列挙した <tun_min>:<tun_max> タプルのコンマ区切りリスト

8.6.6. ml2_type_vlan

以下の表は、**/etc/neutron/plugins/ml2/ml2_conf.ini** ファイルの **[ml2_type_vlan]** グループで使用できるオプションの概要をまとめたものです。

表8.19 ml2_type_vlan

設定オプション = デフォルト値	型	説明
network_vlan_ranges = []	リスト値	<physical_network>:<vlan_min>:<vlan_max> または <physical_network> のリストで、VLAN プロバイダーおよびテナントネットワークで利用可能な physical_network の名前を指定し、テナントネットワークに割り当てるのに利用できる各 VLAN タグの範囲。

8.6.7. ml2_type_vxlan

以下の表は、**/etc/neutron/plugins/ml2/ml2_conf.ini** ファイルの **[ml2_type_vxlan]** グループで使用できるオプションの概要をまとめたものです。

表8.20 ml2_type_vxlan

設定オプション = デフォルト値	型	説明
vni_ranges = []	リスト値	テナントネットワークの割り当てに使用できる VXLAN VNI ID の範囲を列挙した <vni_min>:<vni_max> タプルのコンマ区切りリスト
vxlan_group = None	文字列値	VXLAN のマルチキャストグループ。設定されると、はすべてのブロードキャストトラフィックをこのマルチキャストグループへ送信できるようにします。未設定のままにすると、マルチキャスト VXLAN モードが無効になります。

8.6.8. ovs_driver

以下の表は、**/etc/neutron/plugins/ml2/ml2_conf.ini** ファイルの **[ovs_driver]** グループで使用できるオプションの概要をまとめたものです。

表8.21 ovs_driver

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
vnic_type_prohibit_list = []	リスト値	メカニズムドライバーにより管理が禁止されている VNIC タイプのコンマ区切りのリスト。サポートされる vnic_types は、お使いのネットワークインターフェイスカード、オペレーティングシステムのカーネルバージョン、および OVS バージョンなどの他の要素に依存することに注意してください。ovs メカニズムドライバーの場合には、有効な vnic タイプは normal と direct です。direct は、カーネル 4.8 および ovs 2.8.0 からのみサポートされます。Bind DIRECT (SR-IOV) ポートを使用すると、tc を使用して OVS フローを SR-IOV NIC にオフロードすることができます。これにより、tc によるハードウェアオフロードをサポートし、レプリゼンター net-device を使用して OpenFlow コントロールプレーンにより VF を管理することができます。

8.6.9. securitygroup

以下の表は、**/etc/neutron/plugins/ml2/ml2_conf.ini** ファイルの **[securitygroup]** グループで使用できるオプションの概要をまとめたものです。

表8.22 securitygroup

設定オプション = デフォルト値	型	説明
enable_ipset = True	ブール値	ipset を使用して、iptables ベースのセキュリティーグループをスピードアップします。ipset サポートを有効にするには、ipset が L2 エージェントノードにインストールする必要があります。
enable_security_group = True	ブール値	サーバーで neutron セキュリティーグループ API を有効にするかどうかを制御します。セキュリティーグループを使用しない場合や、nova セキュリティーグループ API を使用する場合は false にする必要があります。
firewall_driver = None	文字列値	L2 エージェントのセキュリティーグループファイアウォール用のドライバー
permitted_ethertypes = []	リスト値	16 進数 (0x で始まる) で、許可されるイーサネットタイプのコンマ区切りリスト。たとえば、"0x4008" は InfiniBand を許可します。

8.6.10. sriov_driver

以下の表は、**/etc/neutron/plugins/ml2/ml2_conf.ini** ファイルの **[sriov_driver]** グループで使用できるオプションの概要をまとめたものです。

表8.23 sriov_driver

設定オプション = デフォルト値	型	説明
vnic_type_prohibit_list = []	リスト値	メカニズムドライバーにより管理が禁止されている VNIC タイプのコンマ区切りのリスト。サポートされる vnic_types は、ネットワークインターフェイスカード、オペレーティングシステムのカーネルバージョン、およびその他の要因によって異なることに注意してください。sriov メカニズムドライバーの場合には、有効な VNIC タイプは direct、macvtap および direct-physical になります。

8.7. NEUTRON.CONF

本項では、**/etc/neutron/neutron.conf** ファイルのオプションについて説明します。

8.7.1. DEFAULT

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
agent_down_time = 75	整数値	エージェントがダウンしていると見なされる秒数です。適切なためにエージェントが停止していることを確認するために、少なくとも 2 回の report_interval にする必要があります。
allow_automatic_dhcp_failover = True	ブール値	オフラインの DHCP エージェントからネットワークを自動的に削除します。
allow_automatic_l3agent_failover = False	ブール値	オフラインの L3 エージェントからオンライン L3 エージェントへのルーターの自動再スケジュール。
allow_bulk = True	ブール値	一括 API の使用を許可する
allow_overlapping_ips = False	ブール値	Neutron での IP アドレスのオーバーラップを許可します。注記: Neutron が Nova セキュリティーグループと併用されている場合は、以下のパラメーターを False に設定する必要があります。

設定オプション = デフォルト値	型	説明
allowed_conntrack_helpers = [{'amanda': 'tcp'}, {'ftp': 'tcp'}, {'h323': 'udp'}, {'h323': 'tcp'}, {'irc': 'tcp'}, {'netbios-ns': 'udp'}, {'pptp': 'tcp'}, {'sane': 'tcp'}, {'sip': 'udp'}, {'sip': 'tcp'}, {'snmp': 'udp'}, {'tftp': 'udp'}]	リスト値	許可される conntrack ヘルパーおよび conntack ヘルパーモジュールのプロトコル制約を定義します。
`api_extensions_path = `	文字列値	API エクステンションのパス。これには、コロンで区切られたパスのリストを使用できます。例: api_extensions_path = extensions:/path/to/more/exts:/even/more/exts.neutron.extensions の **パス** がこれに追加されるので、拡張がある場合はここで指定する必要がありません。
api_paste_config = api-paste.ini	文字列値	api サービスの paste.deploy 設定のファイル名
api_workers = None	整数値	サービスの個別の API ワーカープロセスの数。指定のない場合は、デフォルトの CPU 数で最高のパフォーマンスを得るために利用可能な CPU の数と同じになり、RAM 使用率は上限になります。
auth_strategy = keystone	文字列値	使用する認証のタイプ
backdoor_port = None	文字列値	eventlet バックドアを有効にします。設定可能な値は 0、<port>、および <start>:<end> です。ここで、0 はランダムの tcp ポート番号でリッスンします。<port> は指定されたポート番号でリッスンします (そのポートが使用中の場合にはバックドアを有効にしません)。また、<start>:<end> は、指定されたポート番号の範囲で最も小さい未使用のポート番号でリッスンします。選択したポートは、サービスのログファイルに表示されます。
backdoor_socket = None	文字列値	指定されたパスを接続を受信できる unix ソケットとして使用して、eventlet バックドアを有効にします。このオプションは、**backdoor_port** と相互に排他的です。したがって、どちらか 1 つだけを指定する必要があります。両方を指定すると、このオプションの存在によってそのオプションの使用が上書きされます。パス {pid} の中は、現在のプロセスの PID で置き換えられます。
backlog = 4096	整数値	ソケットを設定するバックログリクエストの数

設定オプション = デフォルト値	型	説明
base_mac = fa:16:3e:00:00:00	文字列値	Neutron が VIF に使用するベース MAC アドレス最初の 3 オクテットは変更しません。4 番目のオクテットが 00 でない場合は、これも使用されます。他は無作為に生成されます。
bind_host = 0.0.0.0	ホストのアドレス値	バインド先のホスト IP。
bind_port = 9696	ポート値	バインドするポート
client_socket_timeout = 900	整数値	クライアント接続のソケット操作のタイムアウト。受信接続がこの秒数の間アイドル状態であると、閉じられます。0 の値は、永久に待機することを意味します。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
control_exchange = neutron	文字列値	トピックがスコープ設定されるデフォルトの変換。transport_url オプションで指定した変換名で上書きできます。
core_plugin = None	文字列値	Neutron が使用するコアプラグイン
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。
default_availability_zones = []	リスト値	アベイラビリティーゾーンのデフォルト値はヒントです。アベイラビリティーゾーンを認識するスケジューラーは、リソース availability_zone_hints が空の場合にこれを使用します。複数のアベイラビリティーゾーンをコンマ区切りの文字列で指定できます。この値は空白にすることができます。この場合、リソースの availability_zone_hints が空であっても、リソースのスケジューリング中にアベイラビリティーゾーンが高可用性のために考慮されます。

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
dhcp_agent_notification = True	ブール値	DHCP エージェントへのリソース操作通知送信の許可
dhcp_agents_per_network = 1	整数値	テナントネットワークをホストするためにスケジュールされている DHCP エージェントの数。この数が 1 を超える場合、スケジューラーは特定のテナントネットワークに複数の DHCP エージェントを自動的に割り当て、DHCP サービスに高可用性を提供します。ただし、これでは、分離されたネットワークにおける IPv6 メタデータサービスの高可用性は提供されません。
dhcp_lease_duration = 86400	整数値	DHCP リース期間 (秒単位)。-1 を使用して、無限リース時間を使用するように dnsmasq に指示します。

設定オプション = デフォルト値	型	説明
dhcp_load_type = networks	文字列値	エージェントによって負荷がレポートされるリソースタイプを表します。これには、networks、subnets、または ports を使用できます。指定すると (デフォルトがネットワークの場合)、サーバーはエージェントレポートの状態からエージェント設定オブジェクトの一部として送信される特定の負荷を抽出します。これは消費されるリソースの数で、すべての report_interval.dhcp_load_type は network_scheduler_driver = neutron.scheduler.dhcp_agent_scheduler.WeightScheduler と組み合わせて使用できます。network_scheduler_driver が WeightScheduler の場合、dhcp_load_type は分散されるリソースの選択を表すように設定できます。例: dhcp_load_type=networks
dns_domain = openstacklocal	文字列値	ホスト名の構築に使用するドメイン
dvr_base_mac = fa:16:3f:00:00:00	文字列値	Neutron が一意の DVR インスタンスに使用するベース mac アドレス最初の 3 オクテットは変更しません。4 番目のオクテットが 00 でない場合は、これも使用されます。他は無作為に生成されます。テナントポート用に割り当てられた MAC と混同しないようにするため、**dvr_base_mac** は **base_mac** とは異なる **必要** があります。4 つのオクテットの例は、dvr_base_mac = fa:16:3f:4f:00:00 のようになります。デフォルトは 3 オクテットです。
enable_dvr = True	ブール値	設定が DVR に設定されているかどうかを判断します。False の場合、DVR API 拡張は無効になります。
enable_new_agents = True	ブール値	enable_new_agents=False の場合、エージェントは admin_state_up=False で始まります。この場合、admin が admin_state_up を True に変更するまで、ユーザーのリソースはエージェントに自動的にスケジュールされません。
enable_services_on_agents_with_admin_state_down = False	ブール値	admin_state_up False を使用してエージェントでサービスを有効にします。このオプションが False の場合、エージェントの admin_state_up が False になっていると、そのエージェント上のサービスが無効になります。このオプションに関係なく、自動スケジューリングでは、admin_state_up False のエージェントが選択されていません。ただし、このオプションが True の場合は、そのようなエージェントに手動でスケジュールすることができます。

設定オプション = デフォルト値	型	説明
enable_snat_by_default = True	ブール値	external_gateway_info に指定されていない場合は、enable_snat のデフォルト値を定義します。
enable_traditional_dhcp = True	ブール値	False の場合、neutron-server は次の DHCP エージェント関連の機能を無効にします。1.DHCP プロビジョニングブロック。2.DHCP スケジューラー API 拡張。3.ネットワークスケジューリングメカニズム。4.DHCP RPC/通知
executor_thread_pool_size = 64	整数値	エグゼキューターがスレッディングまたはイベントレットの場合のエグゼキュータースレッドプールのサイズ。
external_dns_driver = None	文字列値	外部 DNS インテグレーションのドライバー。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
filter_validation = True	ブール値	True の場合には、プラグインがフィルターパラメーターの検証を実行するかどうかを決定します。この設定が有効で、すべてのプラグインでサポートされる場合、フィルター検証が有効になります。
global_physnet_mtu = 1500	整数値	下層の物理ネットワークの MTU。Neutron はこの値を使用して、すべての仮想ネットワークコンポーネントの MTU を算出します。フラットネットワークおよび VLAN ネットワークの場合は、neutron はこの値を変更せずに使用します。VXLAN などのオーバーレイネットワークの場合には、neutron は自動的にこの値からオーバーレイプロトコルのオーバーヘッドを除算します。デフォルトは、Ethernet の標準値である 1500 です。
graceful_shutdown_timeout = 60	整数値	サーバーを正常にシャットダウンするまでのタイムアウトを指定します。値をゼロにすると、endless の待機を意味します。
host = <based on operating system>	ホストのアドレス値	このマシンで実行している Neutron サーバー、エージェント、サービスが使用するホスト名。このマシンで実行しているすべてのエージェントとサービスは、同じホスト値を使用する必要があります。
host_dvr_for_dhcp = True	ブール値	DVR ローカルルーターを DHCP エージェントにホストする必要があるかどうかを判別するためのフラグ。False の場合には、DHCP エージェントインスタンス: DNS がサポートする L3 機能は、たとえば、以下のようになります。

設定オプション = デフォルト値	型	説明
http_retries = 3	整数値	失敗した HTTP 呼び出しでクライアント接続 (nova、ironic) を再試行する回数。0 (ゼロ) は、接続が 1 回だけ試行される (再試行されない) ことを意味します。任意の正の整数に設定すると、接続の失敗時に何度も再試行されます。たとえば、を 3 に設定すると、接続の試行の合計は 4 になります。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。
interface_driver = None	文字列値	仮想インターフェイスの管理に使用するドライバー。
ipam_driver = internal	文字列値	使用する Neutron IPAM (IP アドレス管理) ドライバーデフォルトでは、Neutron IPAM ドライバーの参照実装が使用されます。
ipv6_pd_enabled = False	ブール値	自動サブネット CIDR 割り当ての IPv6 接頭辞委任を有効にします。PD 対応環境でサブネットの割り当て用に IPv6 接頭辞委任を有効にするには、True に設定します。CIDR または subnetpool ID を指定せずに IPv6 サブネット用にサブネット作成要求を行うユーザーには、Prefix Delegation メカニズムを介して CIDR が付与されます。PD を有効にすると、デフォルトの IPv6 サブネットプールの動作を上書きすることに注意してください。
l3_ha = False	ブール値	仮想ルーターの HA モードを有効にします。
l3_ha_net_cidr = 169.254.192.0/18	文字列値	l3 HA 管理ネットワークに使用するサブネット。
`l3_ha_network_physical_name = `	文字列値	HA ネットワークを作成できる物理ネットワーク名。
`l3_ha_network_type = `	文字列値	HA ルーターの HA ネットワークの作成時に使用するネットワーク種別。デフォルトでは、または空の場合は最初の **tenant_network_types** が使用されます。これは、VRRP トラフィックがデフォルトではない特定のネットワークを使用する場合に役立ちます。

設定オプション = デフォルト値	型	説明
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_options = True	ブール値	サービスの起動時 (DEBUG レベル) に、登録されているすべてのオプションの値をログに記録することを有効または無効にします。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_allowed_address_pair = 10	整数値	許可するアドレスペアの最大数
max_dns_nameservers = 5	整数値	サブネットごとの DNS ネームサーバーの最大数
max_header_line = 16384	整数値	許可されるメッセージヘッダー行の最大サイズ。max_header_line は、大きなトークンを使用する際に増やす必要がある場合があります (特に、keystone が大きなサービスカタログの PKI トークンを使用するように設定された場合に生成されるトークン)。
max_l3_agents_per_router = 3	整数値	HA ルーターがスケジュールされる L3 エージェントの最大数。これを 0 に設定すると、ルーターはすべてのエージェントにスケジュールされます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。

設定オプション = デフォルト値	型	説明
max_routes = 30	整数値	ルーターごとのルートの最大数
max_subnet_host_routes = 20	整数値	サブネットごとのホストルートの最大数
`metadata_proxy_group = `	文字列値	グループ (gid または名前) は、初期化後にメタデータプロキシーを実行します (空の場合: agent effective group)。
metadata_proxy_socket = $state_path/metadata_proxy	文字列値	メタデータプロキシー UNIX ドメインソケットの場所。
`metadata_proxy_user = `	文字列値	ユーザー (uid または名前) の初期化後にメタデータプロキシーを実行する (空の場合: agent effective user)。
migration_mode = False	ブール値	このオプションは、環境が OVS から OVN へのメカニズムドライバーの移行プロセス中であることを示します。
network_auto_schedule = True	ブール値	DHCP エージェントへの自動スケジューリングネットワークを許可します。
network_link_prefix = None	文字列値	この文字列は、OpenStack Network API へのリンクで返される通常の URL の前に付けられます。空の場合 (デフォルト)、URL は変更せずに返されます。
network_scheduler_driver = neutron.scheduler.dhcp_agent_scheduler.WeightScheduler	文字列値	DHCP エージェントへのネットワークのスケジューリングに使用するドライバー
notify_nova_on_port_data_changes = True	ブール値	ポートデータ (fixed_ips/floatingip) が変更される際に nova に通知を送信し、nova がキャッシュを更新できるようにします。
notify_nova_on_port_status_changes = True	ブール値	ポートのステータスが変更になったときに nova に通知を送信する
pagination_max_limit = -1	文字列値	単一の応答で返される項目の最大数。値は **無限** または負の整数で制限なしを意味します。
periodic_fuzzy_delay = 5	整数値	タイムスタンプを減らすために定期的なタスクスケジューラーを開始する際の無作為に遅延する秒数。(0 に設定すると無効になります)

設定オプション = デフォルト値	型	説明
periodic_interval = 40	整数値	実行中の定期的なタスク間の秒数。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
retry_until_window = 30	整数値	リッスンする再試行を続ける秒数
router_auto_schedule = True	ブール値	L3 エージェントへのルーターの自動スケジューリングを許可します。
router_distributed = False	ブール値	システム全体のフラグ。テナントが作成できるルーターのタイプを判断します。admin のみが上書き可能です。
router_scheduler_driver = neutron.scheduler.l3_agent_scheduler.LeastRoutersScheduler	文字列値	デフォルトの L3 エージェントへのルータースケジューリングに使用するドライバー
rpc_conn_pool_size = 30	整数値	RPC 接続プールのサイズ。
rpc_ping_enabled = False	ブール値	ping 呼び出しに応答するエンドポイントを追加します。エンドポイントの名前は oslo_rpc_server_ping です。
rpc_response_max_timeout = 600	整数値	RPC 呼び出しからの応答を待つ最大時間 (秒単位)。
rpc_response_timeout = 60	整数値	呼び出しからの応答を待つ秒数。
rpc_state_report_workers = 1	整数値	状態レポートキュー専用の RPC ワーカープロセスの数。
rpc_workers = None	整数値	サービスの RPC ワーカープロセスの数。指定されていない場合、デフォルトは API ワーカー数の半分に等しくなります。

設定オプション = デフォルト値	型	説明
run_external_periodic_tasks = True	ブール値	一部の定期的なタスクは個別のプロセスで実行できます。ここで実行すべきですか？
send_events_interval = 2	整数値	送信するイベントがある場合は nova にイベントを送信するまでの秒数。
service_plugins = []	リスト値	Neutron が使用するサービスプラグイン
setproctitle = on	文字列値	子ワーカーロールに一致するようにプロセス名を設定します。利用可能なオプションは、**off** (以前の動作を保持)**on** (プロセスの名前を **neutron-server: role (元の文字列)** に変更します。**brief** (rename は **on** と同じですが、**neutron-server: role** などの元の文字列はありません。)
state_path = /var/lib/neutron	文字列値	Neutron の状態ファイルを保存する場所。このディレクトリーは、エージェントによって書き込み可能である必要があります。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
tcp_keepidle = 600	整数値	各サーバーソケットの TCP_KEEPIDLE の値を設定します (秒単位)。OS X では対応していません。
transport_url = rabbit://	文字列値	メッセージングバックエンドに接続するためのネットワークアドレスおよびオプションのユーザー認証情報 (URL 形式)。想定される形式は次のとおりです。 driver://[user:pass@]host:port[, [userN:passN@]hostN:portN]/virtual_host?query 例:rabbit://rabbitmq:password@127.0.0.1:5672// URL のフィールドの詳細は、https://docs.openstack.org/oslo.messaging/latest/reference/transport.html で oslo_messaging.TransportURL のドキュメントを参照してください。

設定オプション = デフォルト値	型	説明
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_ssl = False	ブール値	API サーバーでの SSL の有効化
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
vlan_transparent = False	ブール値	True の場合には、プラグインが VLAN トランスペアレントネットワークの作成に対応していれば、そのタイプのネットワークの作成を許可します。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。
wsgi_default_pool_size = 100	整数値	wsgi で使用される greenthread のプールのサイズ
wsgi_keep_alive = True	ブール値	False の場合は、クライアントのソケット接続を明示的に閉じます。

設定オプション = デフォルト値	型	説明
wsgi_log_format = %(client_ip)s "%(request_line)s" status: %(status_code)s len: %(body_length)s time: %(wall_seconds).7f	文字列値	ログ行を生成するためにテンプレートとして使用される Python 形式の文字列。client_ip、date_time、request_line、status_code、body_length、wall_seconds をこの形式にフォーマットできます。
wsgi_server_debug = False	ブール値	サーバーが 500 エラーで例外トレースバックをクライアントに送信する必要がある場合は true。False の場合、サーバーは空の本文で応答します。

8.7.2. agent

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[agent]** グループで使用できるオプションの概要をまとめたものです。

表8.24 agent

設定オプション = デフォルト値	型	説明
availability_zone = nova	文字列値	このノードのアベイラビリティーゾーン
check_child_processes_action = respawn	文字列値	子プロセスが停止したときに実行されるアクション
check_child_processes_interval = 60	整数値	子プロセス liveness (秒) のチェックの間隔。無効にする場合は 0 を使用します。
comment_iptables_rules = True	ブール値	iptables ルールにコメントを追加します。各ルールの目的を記述する、生成された iptables ルールへのコメントの追加を無効にするには、false に設定します。システムは、コメントを追加するために iptables コメントモジュールをサポートする必要があります。
debug_iptables_rules = False	ブール値	フォーマットが生成されることを確認するすべての iptables の差異計算を iptables-save のフォーマットと一致させます。このオプションは、パフォーマンスのペナルティーを課すため、実稼働システムでは有効にしないでください。

設定オプション = デフォルト値	型	説明
kill_scripts_path = /etc/neutron/kill_scripts/	文字列値	外部プロセスを強制終了するために使用されるスクリプトの場所。ここでのスクリプト名は、パターン (<process-name>-kill) に従う必要があります。ここで、<process-name> は、このスクリプトを使用して強制終了するプロセスの名前になります。たとえば、dnsmasq プロセスの kill スクリプトは "dnsmasq-kill" という名前にする必要があります。path を None に設定すると、デフォルトの kill コマンドを使用してプロセスを停止します。
log_agent_heartbeats = False	ブール値	ログエージェントのハートビート
report_interval = 30	浮動小数点の値	状態を server に報告するノード間の秒数。agent_down_time 未満の場合は agent_down_time 未満である必要があります。これは agent_down_time 未満の場合に推奨されます。
root_helper = sudo	文字列値	ルートヘルパーアプリケーション。実際のルートフィルター機能を使用するには、**sudo neutron-rootwrap /etc/neutron/rootwrap.conf** を使用します。**sudo** に変更してフィルタリングをスキップし、コマンドを直接実行します。
root_helper_daemon = None	文字列値	可能な場合は、使用するルートヘルパーデーモンアプリケーション。 **sudo neutron-rootwrap-daemon /etc/neutron/rootwrap.conf** を使用して、daemon mode で rootwrap を実行します。これは、大規模なパフォーマンスを向上させるために報告されています。rootwrap をデーモンモードで実行する方法は、以下を参照してください。 https://docs.openstack.org/oslo.rootwrap/latest/user/usage.html#daemon-mode
use_helper_for_ns_read = True	ブール値	システムの名前空間をリスト表示するときは、root ヘルパーを使用します。セキュリティー設定によっては、これは必要ありません。ルートヘルパーが必要ない場合は、パフォーマンスを向上させるためにこれを False に設定します。
use_random_fully = True	ブール値	SNAT マスカレードルールで完全にランダムに使用します。

8.7.3. cors

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[cors]** グループで使用できるオプションの概要をまとめたものです。

表8.25 cors

設定オプション = デフォルト値	型	説明
allow_credentials = True	ブール値	実際の要求にユーザーの認証情報を含めることができることを示します。
allow_headers = ['X-Auth-Token', 'X-Identity-Status', 'X-Roles', 'X-Service-Catalog', 'X-User-Id', 'X-Tenant-Id', 'X-OpenStack-Request-ID']	リスト値	実際の要求時に使用されるヘッダーフィールド名を示します。
allow_methods = ['GET'、 'PUT'、 'POST'、 'DELETE'、 'PATCH']	リスト値	実際の要求時に使用できるメソッドを示します。
allowed_origin = None	リスト値	このリソースがリクエストの origin"ヘッダーで受信したドメインと共有されるかどうかを示します。形式:"<protocol>://<host>[:<port>]" (行末のスラッシュなし)例:https://horizon.example.com
expose_headers = ['X-Auth-Token', 'X-Subject-Token', 'X-Service-Token', 'X-OpenStack-Request-ID', 'OpenStack-Volume-microversion']	リスト値	API に安全に公開できるヘッダーを示します。デフォルトは HTTP Simple ヘッダーです。
max_age = 3600	整数値	CORS プリフライトリクエストの最大キャッシュ期間。

8.7.4. database

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[database]** グループで使用できるオプションの概要をまとめたものです。

表8.26 database

設定オプション = デフォルト値	型	説明
backend = sqlalchemy	文字列値	データベースに使用するバックエンド。
connection = None	文字列値	データベースへの接続に使用する SQLAlchemy 接続文字列。

設定オプション = デフォルト値	型	説明
connection_debug = 0	整数値	SQL デバッグ情報の冗長性: 0=なし、100=すべて。
`connection_parameters = `	文字列値	接続時に接続 URL に追加するオプションの URL パラメーター。param1=value1&param2=value2&... として指定します。
connection_recycle_time = 3600	整数値	この秒数より長く接続プールに存在していた接続は、次にプールからチェックアウトされたときに新しい接続に置き換えられます。
connection_trace = False	ブール値	Python スタックトレースをコメント文字列として SQL に追加します。
db_inc_retry_interval = True	ブール値	True の場合には、db_max_retry_interval までのデータベース操作の再試行回数を長くします。
db_max_retries = 20	整数値	エラーが発生する前に接続エラーまたはデッドロックが発生した場合の最大再試行。再試行回数を無限に指定するには -1 に設定します。
db_max_retry_interval = 10	整数値	db_inc_retry_interval が設定されている場合は、データベース操作を再試行するまでの最大秒数です。
db_retry_interval = 1	整数値	データベーストランザクションの再試行間の秒数。
`engine = `	文字列値	オフラインマイグレーションの使用時にスクリプトが生成されるデータベースエンジン。
max_overflow = 50	整数値	設定されている場合、この値を SQLAlchemy で max_overflow に使用します。
max_pool_size = 5	整数値	プールに開いたままにする SQL 接続の最大数。値を 0 に設定すると無制限を意味します。
max_retries = 10	整数値	起動時のデータベース接続の最大再試行数。再試行回数を無限に指定するには -1 に設定します。
mysql_enable_ndb = False	ブール値	True の場合には、MySQL Cluster (NDB) の処理のサポートを透過的に有効にします。
mysql_sql_mode = TRADITIONAL	文字列値	MySQL セッションに使用する SQL モード。このオプション (デフォルトを含む) は、サーバーセット SQL モードを上書きします。サーバー設定で設定された SQL モードを使用するには、これを no value に設定します。例: mysql_sql_mode=

設定オプション = デフォルト値	型	説明
pool_timeout = None	整数値	設定されている場合は、この値を SQLAlchemy で pool_timeout に使用します。
retry_interval = 10	整数値	SQL 接続を開く再試行の間隔。
slave_connection = None	文字列値	スレーブデータベースへの接続に使用する SQLAlchemy 接続文字列。
sqlite_synchronous = True	ブール値	True の場合、SQLite は同期モードを使用します。
use_db_reconnect = False	ブール値	失われた接続でデータベースの実験的な使用を有効にします。

8.7.5. Healthcheck

次の表は、**/etc/neutron/neutron.conf** ファイルの **[healthcheck]** グループで使用できるオプションの概要を示しています。

表8.27 Healthcheck

設定オプション = デフォルト値	型	説明
backends = []	リスト値	ヘルスチェックを実施し、リクエストの一部としてその情報を報告できる追加のバックエンド。
detailed = False	ブール値	応答の一部として詳細情報を表示します。セキュリティーメモ: このオプションを有効にすると、監視対象のサービスに関する機密情報が漏洩する可能性があります。セキュリティーポリシーに違反しないことを確認してください。
disable_by_file_path = None	文字列値	ファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを確認します。DisableByFileHealthcheck プラグインで使用されます。
disable_by_file_paths = []	リスト値	ポートに基づいてファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを判断します。文字列の port:path リストを想定します。DisableByFilesPortsHealthcheck プラグインで使用されます。
path = /healthcheck	文字列値	healtcheck 要求に応答するパス。

8.7.6. ironic

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[ironic]** グループで使用できるオプションの概要をまとめたものです。

表8.28 ironic

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
enable_notifications = False	ブール値	通知イベントを ironic に送信します。(たとえば、関連するポートステータスが変わります。)
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名

設定オプション = デフォルト値	型	説明
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名

8.7.7. keystone_authtoken

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[keystone_authtoken]** グループ下で使用できるオプションの概要をまとめたものです。

表8.29 keystone_authtoken

設定オプション = デフォルト値	型	説明
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ

設定オプション = デフォルト値	型	説明
auth_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。このオプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。**非推奨:**Queens 以降 *理由:* auth_uri オプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。
auth_version = None	文字列値	Identity API エンドポイントの API バージョン。
cache = None	文字列値	Swift キャッシュオブジェクトが保存される環境キーを要求します。auth_token ミドルウェアを Swift キャッシュと共にデプロイする場合は、このオプションを使用して、ミドルウェアが swift とキャッシングバックエンドを共有するようにします。それ以外の場合は、代わりに **memcached_servers** オプションを使用します。
cafile = None	文字列値	HTTPs 接続の検証時に使用する PEM でエンコードされた認証局。デフォルトはシステム CA です。
certfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
delay_auth_decision = False	ブール値	ミドルウェア内の認可要求を処理せず、承認の決定をダウンストリームの WSGI コンポーネントに委譲します。

設定オプション = デフォルト値	型	説明
enforce_token_bind = permissive	文字列値	トークンバインディングの使用および種別を制御するために使用されます。トークンバインディングのチェックを無効するには、"disabled"に設定します。バインドタイプがサーバーの認識する形式の場合にはバインディング情報を検証し、そうでない場合には無視するには、"permissive"(デフォルト) に設定します。"strict"は"permissive"と類似していますが、バインドタイプが不明な場合にはトークンが拒否されます。"required"の場合は、いずれかの形式のトークンバインディングが必要です。最後に、トークンに指定する必要のあるバインディングメソッドの名前。
http_connect_timeout = None	整数値	Identity API サーバーと通信する際の要求タイムアウト値。
http_request_max_retries = 3	整数値	Identity API サーバーと通信する際に再接続を試行する回数。
include_service_catalog = True	ブール値	(オプション)X-Service-Catalog ヘッダーを設定するかどうかを示します。False の場合、ミドルウェアはトークンの検証時にサービスカタログを要求せず、X-Service-Catalog ヘッダーを設定しません。
insecure = False	ブール値	HTTPS 接続を確認します。
interface = internal	文字列値	Identity API エンドポイントに使用するインターフェイス。有効な値は、"public"、"internal" (デフォルト)、または "admin" です。
keyfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
memcache_pool_conn_get_timeout = 10	整数値	(オプション) プールから memcached クライアント接続を取得するまで操作が待機する秒数。
memcache_pool_dead_retry = 300	整数値	(オプション)memcached サーバーが停止しているとみなされる秒数。この秒数が経過すると再試行されます。
memcache_pool_maxsize = 10	整数値	(オプション) すべての memcached サーバーへのオープン接続の最大合計数。
memcache_pool_socket_timeout = 3	整数値	(オプション)memcached サーバーと通信する際のソケットのタイムアウト (秒単位)。

設定オプション = デフォルト値	型	説明
memcache_pool_unused_timeout = 60	整数値	(オプション)memcached への接続がプール内で未使用の状態を維持する秒数。この秒数が経過すると終了されます。
memcache_secret_key = None	文字列値	(オプション、memcache_security_strategy が定義されている場合には必須) この文字列は鍵の導出に使用されます。
memcache_security_strategy = None	文字列値	(オプション) 定義されている場合は、トークンデータを認証、または認証して暗号化する必要があるかどうかを示します。MAC の場合、キャッシュでトークンデータが認証されます (HMAC を使用)。ENCRYPT の場合、キャッシュでトークンデータが暗号化され、認証されます。値がこれらのオプションのいずれでもない場合や空の場合には、auth_token は初期化時に例外を発生させます。
memcache_use_advanced_pool = False	ブール値	(オプション) 高度な (eventlet に対して安全な) memcached クライアントプールを使用します。高度なプールは python 2.x でのみ動作します。
memcached_servers = None	リスト値	オプションで、キャッシュに使用する memcached サーバーのリストを指定します。未定義のままの場合、トークンは代わりに処理中にキャッシュされます。
region_name = None	文字列値	アイデンティティーサーバーがあるリージョン。
service_token_roles = ['service']	リスト値	サービストークンに存在する必要があるロールの選択。サービストークンは、期限切れのトークンを使用できることを要求できるため、このチェックでは実際のサービスのみがこのトークンを送信するように厳密に制御する必要があります。ここでのロールは ANY チェックとして適用されるため、このリストのロールはすべて存在している必要があります。後方互換性の理由から、現在 allow_expired チェックにのみ影響します。
service_token_roles_required = False	ブール値	後方互換性の理由から、service_token_roles チェックを有効としてパスしない有効なサービストークンをパスさせる必要があります。これを true に設定することが今後のリリースでデフォルトとなり、可能な場合は有効にされる必要があります。
service_type = None	文字列値	サービスカタログに表示されるサービスの名前または種別。これは、制限されたアクセスルールがあるトークンを検証するために使用されます。

設定オプション = デフォルト値	型	説明
token_cache_time = 300	整数値	トークンの検証に過剰な時間を費やすのを防ぐために、ミドルウェアは、設定可能な期間 (秒単位) 中は以前に見たトークンをキャシュします。キャッシュを完全に無効にするには -1 に設定します。
www_authenticate_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。

8.7.8. nova

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[nova]** グループで使用できるオプションの概要をまとめたものです。

表8.30 nova

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。

設定オプション = デフォルト値	型	説明
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint_type = public	文字列値	使用する nova エンドポイントのタイプ。このエンドポイントは keystone カタログで検索され、パブリック、内部、または管理者のいずれかでなければなりません。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
live_migration_events = False	ブール値	このオプションを有効にすると、ライブマイグレーション中に OVS エージェントは移行先ホストインターフェイスがバインドされている場合にのみ vif-plugged-event を送信します。このオプションは、ポートがプロビジョニングされているときに、他のエージェント (DHCP など) がこのイベントを Nova に送信することも無効にします。このオプションは、Nova パッチhttps://review.opendev.org/c/openstack/nova/+/767368 の場合に有効にできます。このオプションは一時的なものであり、Y で削除され、動作は True になります。
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名

設定オプション = デフォルト値	型	説明
region_name = None	文字列値	使用する nova リージョンの名前。keystone が複数のリージョンを管理する場合に便利です。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名

8.7.9. oslo_concurrency

以下の表では、**/etc/neutron/neutron.conf** ファイルの **[oslo_concurrency]** グループで利用可能なオプションについて説明しています。

表8.31 oslo_concurrency

設定オプション = デフォルト値	型	説明
disable_process_locking = False	ブール値	プロセス間ロックを有効または無効にします。
lock_path = None	文字列値	ロックファイルに使用するディレクトリー。セキュリティー上、指定したディレクトリーは、ロックが必要なプロセスを実行しているユーザーのみが書き込み可能である必要があります。デフォルトは環境変数 OSLO_LOCK_PATH です。外部ロックを使用する場合は、ロックパスを設定する必要があります。

8.7.10. oslo_messaging_amqp

以下の表では、**/etc/neutron/neutron.conf** ファイルの **[oslo_messaging_amqp]** グループで利用可能なオプションの概要を説明します。

表8.32 oslo_messaging_amqp

設定オプション = デフォルト値	型	説明
addressing_mode = dynamic	文字列値	ドライバーが使用するアドレスモードを示します。許可される値:**legacy** - レガシーのルーティング不可能なアドレス指定を使用します。**routable** - ルーティング可能なアドレスを使用します。**dynamic** - メッセージバスがルーティングをサポートしていない場合はレガシーのアドレスを使用します。それ以外の場合は、ルーティング可能なアドレスを使用します。
anycast_address = anycast	文字列値	コンシューマーのグループに送信する際に、アドレス接頭辞に追加されます。コンシューマー間でラウンドロビン方式で配信される必要のあるメッセージを特定するために、メッセージバスで使用されます。
broadcast_prefix = broadcast	文字列値	すべてのサーバーにブロードキャストする場合に使用されるアドレス接頭辞
connection_retry_backoff = 2	整数値	フェイルオーバーの試みに失敗するたびに、connection_retry_interval をこの秒数だけ増やします。
connection_retry_interval = 1	整数値	再接続を試みる前に一時停止する秒数。
connection_retry_interval_max = 30	整数値	connection_retry_interval + connection_retry_backoff の上限
container_name = None	文字列値	AMQP コンテナーの名前。グローバルで一意でなければなりません。デフォルトは、生成された UUID です。
default_notification_exchange = None	文字列値	通知アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_notification_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **notify**
default_notify_timeout = 30	整数値	送信された通知メッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_reply_retry = 0	整数値	リカバリー可能なエラーが原因で失敗した返信メッセージを再送信する最大試行回数。

設定オプション = デフォルト値	型	説明
default_reply_timeout = 30	整数値	rpc 返信メッセージ配信の期限。
default_rpc_exchange = None	文字列値	RPC アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_rpc_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **rpc**
default_send_timeout = 30	整数値	rpc キャストまたは呼び出しメッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_sender_link_timeout = 600	整数値	アイドル状態の送信者リンクのパージをスケジュールする時間。期限切れ後にリンクの割り当てを解除します。
group_request_prefix = unicast	文字列値	グループの任意のサーバーに送信する際のアドレス接頭辞
idle_timeout = 0	整数値	非アクティブな接続のタイムアウト (秒単位)
link_retry_delay = 10	整数値	リカバリー可能なエラーが原因で失敗した AMQP 1.0 リンクの再接続間に一時停止する時間。
multicast_address = multicast	文字列値	ファンアウトメッセージを送信するときにアドレス接頭辞に追加されます。ファンアウトメッセージを識別するためにメッセージバスによって使用されます。
notify_address_prefix = openstack.org/om/notify	文字列値	すべての生成される通知アドレスのアドレス接頭辞
notify_server_credit = 100	整数値	受信通知メッセージのウィンドウサイズ
pre_settled = ['rpc-cast', 'rpc-reply']	多値	このタイプのメッセージを事前処置状態で送信します。事前処置されたメッセージは、ピアから確認応答を受信しません。注記: 事前処置されたメッセージは、配信に失敗した場合に警告なしに破棄される可能性があります。許可される値:**rpc-call** - RPC 呼び出しを事前処置状態で送信します。**rpc-reply**- RPC 返信を事前処置状態で送信します。**rpc-cast** - RPC キャストを事前処置状態で送信します。**notify** - 通知を事前処置状態で送信します。

設定オプション = デフォルト値	型	説明
pseudo_vhost = True	ブール値	仮想ホスト (qpidd など) をネイティブにサポートしていないメッセージバスの仮想ホストサポートを有効にします。true に設定すると、仮想ホスト名はすべてのメッセージバスアドレスに追加され、結果として仮想ホストごとにプライベート **サブネット** を作成します。メッセージバスが仮想ホストの名前として AMQP 1.0 オープン遂行動詞の **hostname** フィールドを使用して仮想ホストをサポートする場合は False に設定します。
reply_link_credit = 200	整数値	受信する RPC 返信メッセージのウィンドウサイズ。
rpc_address_prefix = openstack.org/om/rpc	文字列値	すべての生成される RPC アドレスのアドレス接頭辞
rpc_server_credit = 100	整数値	受信する RPC リクエストメッセージのウィンドウサイズ。
`sasl_config_dir = `	文字列値	SASL 設定が含まれるディレクトリーへのパス
`sasl_config_name = `	文字列値	設定ファイルの名前 (.conf 接尾辞なし)
`sasl_default_realm = `	文字列値	ユーザー名にレルムが存在しない場合に使用する SASL レルム
`sasl_mechanisms = `	文字列値	許可される SASL メカニズムのスペース区切りリスト
server_request_prefix = exclusive	文字列値	特定のサーバーに送信する時に使用するアドレス接頭辞
ssl = False	ブール値	SSL 経由で接続を試みます。その他の ssl 関連のパラメーターが指定されていない場合、システムの CA バンドルを使用してサーバーの証明書を検証します。
`ssl_ca_file = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_cert_file = `	文字列値	クライアント認証用の自己識別証明書 PEM ファイル
`ssl_key_file = `	文字列値	ssl_cert_file 証明書の署名に使用される秘密鍵 PEM ファイル (オプション)
ssl_key_password = None	文字列値	ssl_key_file を復号するためのパスワード (暗号化されている場合)

設定オプション = デフォルト値	型	説明
ssl_verify_vhost = False	ブール値	デフォルトでは、SSL は、サーバーの証明書の名前が transport_url のホスト名と一致することを確認します。設定によっては、代わりに仮想ホスト名を使用することが望ましい場合があります。たとえば、サーバーが Server Name Indication TLS 拡張 (rfc6066) を使用して、仮想ホストごとの証明書を提供する場合などです。サーバーの SSL 証明書が DNS 名ではなく仮想ホスト名を使用する場合は、ssl_verify_vhost を True に設定します。
trace = False	ブール値	デバッグ: AMQP フレームを標準出力 (stdout) にダンプします。
unicast_address = unicast	文字列値	特定の RPC/通知サーバーに送信する際に、アドレス接頭辞に追加されます。単一の送信先に送信されたメッセージを識別するためにメッセージバスによって使用されます。

8.7.11. oslo_messaging_kafka

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[oslo_messaging_kafka]** グループで使用できるオプションの概要をまとめたものです。

表8.33 oslo_messaging_kafka

設定オプション = デフォルト値	型	説明
compression_codec = none	文字列値	プロデューサーによって生成されたすべてのデータの圧縮コーデック。設定されていない場合、圧縮は使用されません。この設定に許可される値は、kafka バージョンに依存することに注意してください。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
consumer_group = oslo_messaging_consumer	文字列値	Kafka コンシューマーのグループ ID。あるグループのコンシューマーは、メッセージ消費を調整します。
enable_auto_commit = False	ブール値	非同期コンシューマーコミットを有効にします。

設定オプション = デフォルト値	型	説明
kafka_consumer_timeout = 1.0	浮動小数点の値	Kafka コンシューマーのデフォルトタイムアウト
kafka_max_fetch_bytes = 1048576	整数値	Kafka コンシューマーの最大フェッチバイト
max_poll_records = 500	整数値	ポーリング呼び出しで返されるレコードの最大数
pool_size = 10	整数値	Kafka コンシューマーのプールサイズ
producer_batch_size = 16384	整数値	プロデューサーの非同期送信のバッチサイズ
producer_batch_timeout = 0.0	浮動小数点の値	KafkaProducer バッチ処理の遅延の上限 (秒単位)
sasl_mechanism = PLAIN	文字列値	セキュリティープロトコルが SASL である際のメカニズム
security_protocol = PLAINTEXT	文字列値	ブローカーとの通信に使用されるプロトコル
`ssl_cafile = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_client_cert_file = `	文字列値	認証に使用されるクライアント証明書の PEM ファイル。
`ssl_client_key_file = `	文字列値	認証に使用されるクライアントキーの PEM ファイル。
`ssl_client_key_password = `	文字列値	認証に使用されるクライアントキーパスワードファイル。

8.7.12. oslo_messaging_notifications

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[oslo_messaging_notifications]** グループで使用できるオプションの概要をまとめたものです。

表8.34 oslo_messaging_notifications

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
driver = []	多値	通知の送信を処理するドライバー。使用できる値は messaging、messagingv2、routing、log、test、noop です。
retry = -1	整数値	リカバリー可能なエラーが原因で配信に失敗した通知メッセージを再送信する最大試行回数。0 - 再試行なし、-1 - 無限回
topics = ['notifications']	リスト値	OpenStack の通知に使用する AMQP トピック。
transport_url = None	文字列値	通知に使用するメッセージングドライバーを表す URL。設定されていない場合は、RPC に使用されるものと同じ設定にフォールバックします。

8.7.13. oslo_messaging_rabbit

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[oslo_messaging_rabbit]** グループで使用できるオプションの概要をまとめたものです。

表8.35 oslo_messaging_rabbit

設定オプション = デフォルト値	型	説明
amqp_auto_delete = False	ブール値	AMQP の自動削除キュー。
amqp_durable_queues = False	ブール値	AMQP で永続キューを使用します。
direct_mandatory_flag = True	ブール値	(非推奨) ダイレクト送信のための RabbitMQ 必須フラグを有効/無効にします。直接送信は応答として使用されるため、クライアントキューが存在しない場合には MessageUndeliverable 例外が発生します。MessageUndeliverable 例外はタイムアウトをループして、送信者が回復する可能性が高くなります。このフラグは非推奨になり、この機能を非アクティブ化することはできません。
enable_cancel_on_failover = False	ブール値	x-cancel-on-ha-failover フラグを有効にして、rabbitmq サーバーがキューが停止しているときにコンシューマーをキャンセルし、通知できるようにします。

設定オプション = デフォルト値	型	説明
heartbeat_in_pthread = False	ブール値	デフォルトでは、ネイティブ Python スレッドを介してヘルスチェックハートビートスレッドを実行します。このオプションが False の場合、ヘルスチェックハートビートは親プロセスから実行モデルを継承します。たとえば、親プロセスが eventlet/greenlet を使用して stdlib にモンキーパッチを適用した場合、ハートビートはグリーンスレッドを介して実行されます。This option should be set to True only for the wsgi services.
heartbeat_rate = 2	整数値	heartbeat_timeout_threshold 中、ハートビートを確認する回数。
heartbeat_timeout_threshold = 60	整数値	ハートビートの keep-alive が失敗した場合に Rabbit ブローカーがダウンとみなされるまでの秒数 (0 はハートビートを無効にします)。
kombu_compression = None	文字列値	実験的用途: 許容値は gzip、bz2 です。設定されていない場合、圧縮は使用されません。このオプションは、今後のバージョンで利用できない可能性があります。
kombu_failover_strategy = round-robin	文字列値	現在接続しているノードが利用できなくなった場合に、次の RabbitMQ ノードを選択する方法を指定します。設定で複数の RabbitMQ ノードが指定される場合に限り有効になります。
kombu_missing_consumer_retry_timeout = 60	整数値	応答のないクライアントが応答を送信するのを待つ時間。この時間が経過すると無視します。この値は rpc_response_timeout より長くすることはできません。
kombu_reconnect_delay = 1.0	浮動小数点の値	AMQP コンシューマーの取り消し通知への応答で、再接続するまでの待機時間。
rabbit_ha_queues = False	ブール値	RabbitMQ (x-ha-policy: all) の HA キューの使用を試みます。このオプションを変更する場合は、RabbitMQ データベースを消去する必要があります。RabbitMQ 3.0 では、キューを宣言する際に x-ha-policy 引数によってキューのミラーリングが制御されなくなりました。すべてのキュー (自動生成された名前のキューを除く) がすべてのノードでミラーリングされるようにするには、"rabbitmqctl set_policy HA **^(?!amq\.).* {"ha-mode": "all"}**" を実行します。
rabbit_interval_max = 30	整数値	RabbitMQ 接続を再試行する最大間隔。デフォルトは 30 秒です。

設定オプション = デフォルト値	型	説明
rabbit_login_method = AMQPLAIN	文字列値	RabbitMQ ログイン方法
rabbit_qos_prefetch_count = 0	整数値	事前フェッチするメッセージの数を指定します。ゼロに設定すると、無制限のメッセージのフェッチが許可されます。
rabbit_retry_backoff = 2	整数値	RabbitMQ に接続する際に再試行間でバックオフする長さ
rabbit_retry_interval = 1	整数値	RabbitMQ との接続を再試行する頻度。
rabbit_transient_queues_ttl = 1800	整数値	キューの TTL (x-expires) の期間 (秒単位) を表す正の整数。TTL の期間使用されないキューは自動的に削除されます。このパラメーターは応答キューとファンアウトキューにのみ影響します。
ssl = False	ブール値	SSL 経由で接続します。
`ssl_ca_file = `	文字列値	SSL 認証局ファイル (SSL が有効な場合にのみ有効)。
`ssl_cert_file = `	文字列値	SSL 証明書ファイル (SSL が有効な場合にのみ有効)。
`ssl_key_file = `	文字列値	SSL キーファイル (SSL が有効な場合にのみ有効)。
`ssl_version = `	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

8.7.14. oslo_middleware

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[oslo_middleware]** グループで使用できるオプションの概要をまとめたものです。

表8.36 oslo_middleware

設定オプション = デフォルト値	型	説明
enable_proxy_headers_parsing = False	ブール値	アプリケーションがプロキシーの背後であるかどうか。これにより、ミドルウェアがヘッダーを解析すべきかどうかを決定します。

8.7.15. oslo_policy

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[oslo_policy]** グループ下で使用できるオプションの概要をまとめたものです。

表8.37 oslo_policy

設定オプション = デフォルト値	型	説明
enforce_new_defaults = False	ブール値	このオプションは、ポリシーを評価するときに古い非推奨のデフォルトを使用するかどうかを制御します。**True** の場合、古い非推奨のデフォルトは評価されません。これは、既存のトークンが古いデフォルトで許可されているが、新しいデフォルトで許可されていない場合、許可されないことを意味します。新しいデフォルトと **scope_type** の利点を一緒に得ることができるように、このフラグを **enforce_scope** フラグと一緒に有効にすることを推奨します
enforce_scope = False	ブール値	このオプションは、ポリシーを評価する際にスコープを適用するかどうかを制御します。**True** の場合、リクエストで使用されるトークンのスコープは、適用されるポリシーの **scope_types** と比較されます。スコープが一致しない場合、**InvalidScope** 例外が発生します。**False** の場合、ポリシーが一致しないスコープで呼び出されていることを運用者に通知するメッセージが記録されます。
policy_default_rule = default	文字列値	デフォルトルール。要求されたルールが見つからない場合に適用されます。
policy_dirs = ['policy.d']	多値	ポリシー設定ファイルが保存されるディレクトリー。これは、config_dir オプションで定義される検索パス内の任意のディレクトリーへの相対パスか、絶対パスにすることができます。このディレクトリーの検索には、policy_file で定義されたファイルが存在している必要があります。存在しないディレクトリーまたは空のディレクトリーは無視されます。
policy_file = policy.yaml	文字列値	ロールを指定のサービスのパーミッションにマッピングするファイルの相対パスまたは絶対パス。相対パスは、このオプションを設定する設定ファイルに対する相対パスとして指定する必要があります。
remote_content_type = application/x-www-form-urlencoded	文字列値	REST ベースのポリシーチェックのデータを送受信するコンテンツ種別
remote_ssl_ca_crt_file = None	文字列値	REST ベースのポリシーチェック用の認証局証明書ファイルの絶対パス

設定オプション = デフォルト値	型	説明
remote_ssl_client_crt_file = None	文字列値	REST ベースのポリシーチェック用のクライアント証明書の絶対パス
remote_ssl_client_key_file = None	文字列値	REST ベースのポリシーチェック用のクライアント鍵ファイルの絶対パス
remote_ssl_verify_server_crt = False	ブール値	REST ベースのポリシーチェックのサーバーアイデンティティー検証

8.7.16. oslo_reports

次の表は、**/etc/neutron/neutron.conf** ファイルの **[oslo_reports]** グループで使用できるオプションの概要を示しています。

表8.38 oslo_reports

設定オプション = デフォルト値	型	説明
file_event_handler = None	文字列値	シグナルではなくレポートをトリガーする変更を監視するファイルへのパス。このオプションを設定すると、レポートのシグナルトリガーが無効になります。アプリケーションが WSGI アプリケーションとして実行されている場合は、シグナルの代わりにこれを使用することが推奨されます。
file_event_handler_interval = 1	整数値	file_event_handler が設定されている場合にポーリングの間隔 (秒単位)
log_dir = None	文字列値	ファイルを作成するログディレクトリーへのパス

8.7.17. placement

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[placement]** グループで使用できるオプションの概要をまとめたものです。

表8.39 placement

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_type = None	文字列値	読み込む認証タイプ

設定オプション = デフォルト値	型	説明
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint_type = public	文字列値	使用する配置エンドポイントのタイプ。このエンドポイントは keystone カタログで検索され、パブリック、内部、または管理者のいずれかでなければなりません。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名

設定オプション = デフォルト値	型	説明
region_name = None	文字列値	使用する配置リージョンの名前。keystone が複数のリージョンを管理する場合に便利です。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	ユーザー ID
username = None	文字列値	ユーザー名

8.7.18. privsep

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[privsep]** グループ下で使用できるオプションの概要をまとめたものです。

表8.40 privsep

設定オプション = デフォルト値	型	説明
capabilities = []	リスト値	privsep デーモンによって保持される Linux 機能のリスト。
group = None	文字列値	privsep デーモンを実行する必要のあるグループです。

設定オプション = デフォルト値	型	説明
helper_command = None	文字列値	fork メソッドを使用していない場合に privsep デーモンを起動するコマンドを呼び出します。指定のない場合は、現在の設定を再作成するために設計された sudo privsep-helper および引数を使用してデフォルト値が生成されます。このコマンドは、適切な --privsep_context および --privsep_sock_path 引数を使用する必要があります。
logger_name = oslo_privsep.daemon	文字列値	Logger name to use for this privsep context.By default all contexts log with oslo_privsep.daemon.
thread_pool_size = <based on operating system>	整数値	privsep が同時にプロセスを実行できるスレッドの数。デフォルトは、システム内の CPU コア数に設定されます。
user = None	文字列値	privsep デーモンを実行する必要のあるユーザー。

8.7.19. profiler

次の表は、**/etc/neutron/neutron.conf** ファイルの **[profiler]** グループ下で使用できるオプションの概要をまとめたものです。

表8.41 profiler

設定オプション = デフォルト値	型	説明
connection_string = messaging://	文字列値	通知バックエンドの接続文字列。 デフォルト値は **messaging://** で、通知機能を oslo_messaging に設定します。 使用できる値の例: ● **messaging://**: スパンの送信に oslo_messaging ドライバーを使用します。 ● **redis://127.0.0.1:6379**: スパンの送信に redis ドライバーを使用します。 ● **mongodb://127.0.0.1:27017**: スパンの送信に mongodb ドライバーを使用します。 ● **elasticsearch://127.0.0.1:9200:** elasticsearch ドライバーを使用してスパンを送信します。 ● **jaeger://127.0.0.1:6831**: スパンの送信に jaeger トレースを使用します。

設定オプション = デフォルト値	型	説明
enabled = False	ブール値	このノード上の全サービスのプロファイリングを有効にします。 デフォルト値は False です (プロファイリング機能を無効にします)。 値: ● True: 機能を有効にします ● False: 機能を無効にします。このプロジェクトの操作ではプロファイリングを開始できません。プロファイリングが別のプロジェクトでトリガーされた場合には、このプロジェクトの部分は空になります。
es_doc_type = notification	文字列値	elasticsearch の通知インデックスについてのドキュメントタイプ。
es_scroll_size = 10000	整数値	elasticsearch は大規模な要求をバッチに分割します。このパラメーターは、各バッチの最大サイズを定義します (例: es_scroll_size=10000)。
es_scroll_time = 2m	文字列値	このパラメーターは時間値パラメーター (es_scroll_time=2m など) です。これは、検索に参加するノードが継続してサポートするために関連するリソースを維持する時間を示します。
filter_error_trace = False	ブール値	エラー/例外を含むフィルタートレースを分離された場所へ有効にします。 デフォルト値は False に設定されます。 値: ● True: エラー/例外を含むフィルタートレースを有効にします。 ● False: フィルターを無効にします。

設定オプション = デフォルト値	型	説明
hmac_keys = SECRET_KEY	文字列値	パフォーマンスプロファイリング用にコンテキストデータの暗号化に使用するシークレットキー。 この文字列値は <key1>[,<key2>,...<keyn>] の形式にする必要があります。各キーはランダムな文字列です。REST API 経由でプロファイリングをトリガーするユーザーは、REST API 呼び出しのヘッダーにこれらのキーの 1 つを設定し、この特定プロジェクトのノードのプロファイリング結果が含まれるようにする必要があります。 プロファイリングを有効にするには、enabled フラグと hmac_keys 設定オプションの両方を設定する必要があります。また、少なくとも 1 つのキーで全サービス間で正しいプロファイリング情報を生成するには、OpenStack プロジェクト間で一貫性を確保する必要があります。これにより、クライアント側からを使用して、すべての可能なリソースからの情報を含むトレースを生成できます。
sentinel_service_name = mymaster	文字列値	Redissentinel はサービス名を使用してマスターの redis サービスを識別します。このパラメーターは名前を定義します (例: **sentinal_service_name=mymaster**)。
socket_timeout = 0.1	浮動小数点の値	Redissentinel は、接続に timeout オプションを提供します。このパラメーターは、タイムアウトを定義します (例: socket_timeout=0.1)。
trace_sqlalchemy = False	ブール値	サービスで SQL 要求のプロファイリングを有効にします。 デフォルト値は False です (SQL 要求はトレースされません)。 値: ● True: SQL 要求のプロファイリングを有効にします。各 SQL クエリーはトレースの一部となり、それにどの程度の時間を費やしたかによって分析できます。 ● False: SQL 要求のプロファイリングを無効にします。費やした時間は、高いレベルの操作でのみ表示されます。単一の SQL クエリーをこのように分析できません。

8.7.20. quotas

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[quotas]** グループで使用できるオプションの概要をまとめたものです。

表8.42 quotas

設定オプション = デフォルト値	型	説明
default_quota = -1	整数値	テナントごとに許可されるデフォルトのリソース数。負の値は無制限を意味します。
quota_driver = neutron.db.quota.driver_nolock.DbQuotaNoLockDriver	文字列値	クォータチェックに使用するデフォルトのドライバー。
quota_floatingip = 50	整数値	テナントごとに許可される Floating IP の数。負の値は無制限を意味します。
quota_network = 100	整数値	テナントごとに許可されるネットワークの数。負の値は無制限を意味します。
quota_port = 500	整数値	テナントごとに許可されるポート数。負の値は無制限を意味します。
quota_router = 10	整数値	テナントごとに許可されるルーターの数。負の値は無制限を意味します。
quota_security_group = 10	整数値	テナントごとに許可されるセキュリティーグループの数。負の値は無制限を意味します。
quota_security_group_rule = 100	整数値	テナントごとに許可されるセキュリティールールの数。負の値は無制限を意味します。
quota_subnet = 100	整数値	テナントごとに許容できるサブネットの数。負の値は無制限を意味します。
track_quota_usage = True	ブール値	現在のリソースクォータの使用状況のデータベースを追跡します。neutron データベースを使用しないプラグインは、このフラグを False に設定する必要があります。

8.7.21. ssl

以下の表は、**/etc/neutron/neutron.conf** ファイルの **[ssl]** グループで使用できるオプションの概要をまとめたものです。

表8.43 ssl

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	接続クライアントの検証に使用する CA 証明書ファイル。
cert_file = None	文字列値	サーバーをセキュアに起動する際に使用する証明書ファイル。
ciphers = None	文字列値	利用可能な暗号のリストを設定します。値は OpenSSL 暗号リスト形式の文字列である必要があります。
key_file = None	文字列値	サーバーをセキュアに起動する際に使用する秘密鍵ファイル。
version = None	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

8.8. OPENVSWITCH_AGENT.INI

本項では、**/etc/neutron/plugins/ml2/openvswitch_agent.ini** ファイルのオプションについて説明します。

8.8.1. DEFAULT

以下の表は、**/etc/neutron/plugins/ml2/openvswitch_agent.ini** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。

設定オプション = デフォルト値	型	説明
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
rpc_response_max_timeout = 600	整数値	RPC 呼び出しからの応答を待つ最大時間 (秒単位)。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

8.8.2. agent

以下の表は、**/etc/neutron/plugins/ml2/openvswitch_agent.ini** ファイルの **[agent]** グループで使用できるオプションの概要をまとめたものです。

表8.44 agent

設定オプション = デフォルト値	型	説明
arp_responder = False	ブール値	ローカル ARP レスポンダーがサポートされる場合は有効にします。OVS 2.1 および ML2 l2opulation ドライバーが必要です。スイッチが (オーバーレイをサポートしている場合)、オーバーレイへのコストのかかる ARP ブロードキャストを実行せずに、ローカルで ARP 要求に応答できるようにします。enable_distributed_routing が True に設定されている場合は、設定ファイルの設定に関係なく、エージェントでは arp_responder が自動的に True に設定されます。
baremetal_smartnic = False	ブール値	エージェントを有効にして、スマート NIC ポートを処理します。

設定オプション = デフォルト値	型	説明
dont_fragment = True	ブール値	GRE/VXLAN トンネルを伝送する発信 IP パケットに don't fragment (DF) ビットを設定または設定します。
drop_flows_on_start = False	ブール値	開始時にフローテーブルをリセットします。これを True に設定すると、短いトラフィックが中断します。
enable_distributed_routing = False	ブール値	l2 エージェントを DVR モードで実行させます。
explicitly_egress_direct = False	ブール値	True に設定すると、許可される egress ユニキャストトラフィックはアクション NORMAL を使用しません。許可される egress パケットは、ユニキャストトラフィックの最終的な egress テーブルの出力フローに対応します。
extensions = []	リスト値	使用するエクステンションリスト
l2_population = False	ブール値	ML2 l2population メカニズムドライバーを使用して、リモート MAC および IP を確認し、トンネルのスケーラビリティーを向上させます。
minimize_polling = True	ブール値	インターフェイスの変更について ovsdb を監視することにより、ポーリングを最小限に抑えます。
ovsdb_monitor_respawn_interval = 30	整数値	ovsdb モニターの通信が失われた後に ovsdb モニターを再試行するまで待機する秒数。
tunnel_csum = False	ブール値	発信 IP パケットを転送する GRE/VXLAN トンネルのトンネルヘッダーチェックサムを設定または設定します。
tunnel_types = []	リスト値	エージェントがサポートするネットワークタイプ (gre、vxlan、および/ または geneve)。
veth_mtu = 9000	整数値	veth インターフェイスの MTU サイズ
vxlan_udp_port = 4789	ポート値	VXLAN トンネルに使用する UDP ポート。

8.8.3. network_log

以下の表は、**/etc/neutron/plugins/ml2/openvswitch_agent.ini** ファイルの **[network_log]** グループで使用できるオプションの概要をまとめたものです。

表8.45 network_log

設定オプション = デフォルト値	型	説明
burst_limit = 25	整数値	rate_limit あたりの最大パケット数。
local_output_log_base = None	文字列値	エージェント側の出力ログファイルパス、デフォルトの syslog ファイル。
rate_limit = 100	整数値	1 秒あたりの最大パケットロギング。

8.8.4. ovs

以下の表は、**/etc/neutron/plugins/ml2/openvswitch_agent.ini** ファイルの **[ovs]** グループで使用できるオプションの概要をまとめたものです。

表8.46 ovs

設定オプション = デフォルト値	型	説明
bridge_mappings = []	リスト値	フラットネットワークおよび VLAN ネットワークに使用する、物理ネットワーク名をエージェントのノード固有の Open vSwitch ブリッジ名にマッピングする <physical_network>:<bridge> タプルのコンマ区切りリスト。ブリッジ名の長さは 11 を超えないようにする必要があります。各ブリッジは存在している必要があり、ポートとして設定された物理ネットワークインターフェイスが必要です。サーバーに設定したすべての物理ネットワークには、各エージェント上の適切なブリッジへのマッピングが必要です。注意: このマッピングからブリッジを削除する場合は、エージェントによって管理されないため、必ずブリッジからブリッジを切断してください。
datapath_type = system	文字列値	使用する OVS データパス。**system** はデフォルト値で、カーネルデータパスに対応します。ユーザー空間データパスを有効にするには、この値を **netdev** に設定します。
disable_packet_marking = False	ブール値	QoS 拡張機能が有効な場合、パケットマーキングを無効にします。**skb_priority**、**sbk_mark**、および出力キューフィールドがサポートされ、オフロードできるようになるまで、ハードウェアオフロードで OVS を使用する場合は、このオプションを有効にする必要があります。このオプションが有効になっている場合、レート QoS ルール (帯域幅制限または最小帯域幅) は VirtIO ポートに対して機能しません。
int_peer_patch_port = patch-tun	文字列値	トンネルブリッジ用の統合ブリッジのピアパッチポート。

設定オプション = デフォルト値	型	説明
integration_bridge = br-int	文字列値	使用する統合ブリッジ。適切な理由がない限り、このパラメーターを変更しないでください。これは OVS 統合ブリッジの名前です。ハイパーバイザーごとに 1 つがあります。統合ブリッジは、仮想 **パッチベイ** として機能します。すべての VM VIF は、このブリッジに接続されてから、ネットワーク接続に従って **パッチが当てられます**。
local_ip = None	IP アドレス値	ローカルオーバーレイ (tunnel) ネットワークエンドポイントの IP アドレス。ホストネットワークインターフェイスのいずれかにある IPv4 アドレスまたは IPv6 アドレスを使用します。この値の IP バージョンは、neutron サーバーノード上の ML2 プラグイン設定ファイルの **overlay_ip_version** オプションの値と一致する必要があります。
of_connect_timeout = 300	整数値	コントローラーを接続しているローカルスイッチを待つタイムアウト (秒単位)。
of_inactivity_probe = 10	整数値	コントローラーへのローカルスイッチ接続の間隔 (秒単位)。0 を値として指定すると、非アクティブプローブが無効になります。
of_listen_address = 127.0.0.1	IP アドレス値	OpenFlow 接続をリッスンするアドレス。
of_listen_port = 6633	ポート値	OpenFlow 接続をリッスンするポート。
of_request_timeout = 300	整数値	単一の OpenFlow 要求を待つタイムアウト (秒単位)。
ovsdb_connection = tcp:127.0.0.1:6640	文字列値	OVSDB バックエンドの接続文字列。すべての ovsdb コマンドに使用され、監視時に ovsdb-client によって使用されます。
ovsdb_debug = False	ブール値	OVSDB デバッグログの有効化

設定オプション = デフォルト値	型	説明
resource_provider_bandwidths = []	リスト値	<bridge>:<egress_bw>:<ingress_bw> タプルのコンマ区切りリスト。指定のブリッジで利用可能な帯域幅が指定の方向に表示されます。方向は、VM の視点から行われます。帯域幅は、1秒あたりのキロバイト単位 (kbps) で測定されます。ブリッジは bridge_mappings に値として表示されるはずです。ただし、bridge_mappings のすべてのブリッジをここに記載する必要があるわけではありません。ここで記載されていないブリッジでは、配置にリソースプロバイダーを作成したり、インベントリーを報告したりすることはありません。省略の方向は、対応するクラスのインベントリーを報告しないことを意味します。
resource_provider_default_hypervisor = None	文字列値	リソースプロバイダーの親を見つけるために使用されるデフォルトのハイパーバイザー名。このオプションが設定されていない場合、標準的な名前が使用されます
resource_provider_hypervisors = {}	dict 値	ブリッジのハイパーバイザーへのマッピング: <bridge>:<hypervisor>,... ハイパーバイザー名は、リソースプロバイダーツリーの親を見つけるために使用されます。ハイパーバイザー名が、そのハイパーバイザーを管理する nova-compute によって認識される resource_provider_default_hypervisor 設定オプションの値と異なる場合にのみ、設定する必要があります。
resource_provider_inventory_defaults = {'allocation_ratio': 1.0, 'min_unit': 1, 'reserved': 0, 'step_size': 1}	dict 値	Key:Value ペアは、リソースプロバイダーのインベントリーのレポート中に使用されるデフォルトを指定します。それぞれのタイプを持つ、考えられるキー: allocation_ratio:float、max_unit:int、min_unit:int、reserved:int、step_size:int。https://docs.openstack.org/api-ref/placement/#update-resource-provider-inventories も参照してください。
ssl_ca_cert_file = None	文字列値	OVSDB との対話時に使用する認証局 (CA) 証明書。接頭辞 ovsdb_connection を使用する場合は必須
ssl_cert_file = None	文字列値	OVSDB との対話時に使用する SSL 証明書ファイル。接頭辞 ovsdb_connection を使用する場合は必須
ssl_key_file = None	文字列値	OVSDB との対話時に使用する SSL 秘密鍵ファイル。接頭辞 ovsdb_connection を使用する場合は必須
tun_peer_patch_port = patch-int	文字列値	統合ブリッジ用のトンネルブリッジのピアパッチポート。

設定オプション = デフォルト値	型	説明
tunnel_bridge = br-tun	文字列値	使用するトンネルブリッジ。
vhostuser_socket_dir = /var/run/openvswitch	文字列値	OVS vhost-user ソケットディレクトリー。

8.8.5. securitygroup

以下の表は、**/etc/neutron/plugins/ml2/openvswitch_agent.ini** ファイルの **[securitygroup]** グループで使用できるオプションの概要をまとめたものです。

表8.47 securitygroup

設定オプション = デフォルト値	型	説明
enable_ipset = True	ブール値	ipset を使用して、iptables ベースのセキュリティーグループをスピードアップします。ipset サポートを有効にするには、ipset が L2 エージェントノードにインストールする必要があります。
enable_security_group = True	ブール値	サーバーで neutron セキュリティーグループ API を有効にするかどうかを制御します。セキュリティーグループを使用しない場合や、nova セキュリティーグループ API を使用する場合は false にする必要があります。
firewall_driver = None	文字列値	L2 エージェントのセキュリティーグループファイアウォール用のドライバー
permitted_ethertypes = []	リスト値	16 進数 (0x で始まる) で、許可されるイーサネットタイプのコンマ区切りリスト。たとえば、"0x4008" は InfiniBand を許可します。

8.9. SRIOV_AGENT.INI

本項では、**/etc/neutron/plugins/ml2/sriov_agent.ini** ファイルのオプションについて説明します。

8.9.1. DEFAULT

以下の表は、**/etc/neutron/plugins/ml2/sriov_agent.ini** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。
default_log_levels = ['amqp=WARN'、 'amqplib=WARN'、 'boto=WARN'、 'qpid=WARN'、 'sqlalchemy=WARN'、 'suds=INFO'、 'oslo.messaging=INFO'、 'oslo_messaging=INFO'、 'iso8601=WARN'、 'requests.packages.urllib3.connectionpool=WARN'、 'urllib3.connectionpool=WARN'、 'websocket=WARN'、 'requests.packages.urllib3.util.retry=WARN'、 'urllib3.util.retry=WARN'、 'keystonemiddleware=WARN'、 'routes.middleware=WARN'、 'stevedore=WARN'、 'taskflow=WARN'、 'keystoneauth=WARN'、 'oslo.cache=INFO'、 'oslo_policy=INFO'、 'dogpile.core.dogpile=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。

設定オプション = デフォルト値	型	説明
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
rpc_response_max_timeout = 600	整数値	RPC 呼び出しからの応答を待つ最大時間 (秒単位)。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。

8.9.2. agent

以下の表は、**/etc/neutron/plugins/ml2/sriov_agent.ini** ファイルの **[agent]** グループで使用できるオプションの概要をまとめたものです。

表8.48 agent

設定オプション = デフォルト値	型	説明
extensions = []	リスト値	使用するエクステンションリスト

8.9.3. sriov_nic

以下の表は、**/etc/neutron/plugins/ml2/sriov_agent.ini** ファイルの **[sriov_nic]** グループで使用できるオプションの概要をまとめたものです。

表8.49 sriov_nic

設定オプション = デフォルト値	型	説明
exclude_devices = []	リスト値	<network_device>:<vfs_to_exclude> タプルのコンマ区切りリスト。network_device を、仮想ネットワークに使用しないでください。vfs_to_exclude は、network_device から除外する仮想機能のセミコロン区切りのリストです。マッピングの network_device が physical_device_mappings リストに表示されます。
physical_device_mappings = []	リスト値	物理ネットワーク名を VLAN ネットワークに使用する SR-IOV Physical Function のノード固有の物理ネットワークデバイスインターフェイスにマッピングする <physical_network>:<network_device> タプルのコンマ区切りリスト。サーバーの network_vlan_ranges にリスト表示されるすべての物理ネットワークには、各エージェントの適切なインターフェイスへのマッピングが必要です。
resource_provider_bandwidths = []	リスト値	<network_device>:<egress_bw>:<ingress_bw>: <ingress_bw> タプルのコンマ区切りリスト。指定のデバイスの利用可能な帯域幅が指定の方向に表示されます。方向は、VM の視点から行われます。帯域幅は、1 秒あたりのキロバイト単位 (kbps) で測定されます。デバイスは、physical_device_mappings に値として表示されるはずです。ただし、physical_device_mappings 内の全デバイスをここに記載する必要があるわけではありません。ここに記載されていないデバイスの場合、配置にリソースプロバイダーを作成したり、インベントリーを報告したりすることはありません。省略の方向は、対応するクラスのインベントリーを報告しないことを意味します。
resource_provider_default_hypervisor = None	文字列値	リソースプロバイダーの親を見つけるために使用されるデフォルトのハイパーバイザー名。このオプションが設定されていない場合、標準的な名前が使用されます
resource_provider_hypervisors = {}	dict 値	ネットワークデバイスのハイパーバイザーへのマッピング:<network_device>:<hypervisor>,... ハイパーバイザー名は、リソースプロバイダーツリーの親を見つけるために使用されます。ハイパーバイザー名が、そのハイパーバイザーを管理する nova-compute によって認識される resource_provider_default_hypervisor 設定オプションの値と異なる場合にのみ、設定する必要があります。

設定オプション = デフォルト値	型	説明
resource_provider_inventory_defaults = {'allocation_ratio': 1.0, 'min_unit': 1, 'reserved': 0, 'step_size': 1}	dict 値	Key:Value ペアは、リソースプロバイダーのインベントリーのレポート中に使用されるデフォルトを指定します。それぞれのタイプを持つ、考えられるキー: allocation_ratio:float、max_unit:int、min_unit:int、reserved:int、step_size:int。https://docs.openstack.org/api-ref/placement/#update-resource-provider-inventories も参照してください。

第9章 NOVA

以下の章では、**nova** サービスの設定オプションについて説明します。

9.1. NOVA.CONF

本項では、**/etc/nova/nova.conf** ファイルのオプションについて説明します。

9.1.1. DEFAULT

以下の表は、**/etc/nova/nova.conf** ファイルの **[DEFAULT]** グループで使用できるオプションの概要をまとめたものです。

.

設定オプション = デフォルト値	型	説明
allow_resize_to_same_host = False	ブール値	サイズ変更のために、宛先マシンがソースに一致することを許可します。単一ホスト環境でテストする場合に便利です。デフォルトでは、同じホストへのリサイズは許可されません。このオプションを true に設定すると、同じホストが宛先オプションに追加されます。ServerGroupAffinityFilter を許可し、サイズ変更が必要な場合は true に設定します。
arq_binding_timeout = 300	整数値	アクセラレーター要求 (ARQ) バインドイベントメッセージの受信のタイムアウト。 ARQ バインド解決イベントの受信を待機する秒数。このイベントは、インスタンスのすべての ARQ が正常にバインドされたか、バインドに失敗したことを示します。受信しない場合、インスタンスの起動は例外で中止されます。
backdoor_port = None	文字列値	eventlet バックドアを有効にします。設定可能な値は 0、<port>、および <start>:<end> です。ここで、0 はランダムの tcp ポート番号でリッスンします。<port> は指定されたポート番号でリッスンします (そのポートが使用中の場合にはバックドアを有効にしません)。また、<start>:<end> は、指定されたポート番号の範囲で最も小さい未使用のポート番号でリッスンします。選択したポートは、サービスのログファイルに表示されます。

設定オプション = デフォルト値	型	説明
backdoor_socket = None	文字列値	指定されたパスを接続を受信できる unix ソケットとして使用して、eventlet バックドアを有効にします。このオプションは、**backdoor_port** と相互に排他的です。したがって、どちらか 1 つだけを指定する必要があります。両方を指定すると、このオプションの存在によってそのオプションの使用が上書きされます。パス {pid} の中は、現在のプロセスの PID で置き換えられます。
block_device_allocate_retries = 60	整数値	サーバー作成時に、ボリュームを available に割り当てるまでのチェック回数。 ブロックデバイスマッピングでサーバーを作成する場合、**source_type** は **空** の、**image** または **snapshot** のいずれか、および **destination_type** が **ボリューム** の場合、**nova-compute** サービスはボリュームを作成し、それをサーバーに割り当てます。ボリュームにアタッチする前に、ステータスが available である必要があります。このオプションは、作成されたボリュームが接続される前に、使用可能であることを確認する回数を制御します。 操作がタイムアウトした場合、ブロックデバイスマッピング **delete_on_termination** 値が True の場合、ボリュームが削除されます。 この操作を高速化するには、ブロックストレージサービスにイメージキャッシュを設定することを推奨します。詳細は、https://docs.openstack.org/cinder/latest/admin/blockstorage-image-volume-cache.html を参照してください。 値: ● 60 (デフォルト) ● 値が 0 の場合は、1 回試行されます。 ● 任意の値 > 0 の場合、合計試行回数は (値 + 1) です。 関連するオプション: ● **block_device_allocate_retries_interval**: チェックの間隔を制御します。

設定オプション = デフォルト値	型	説明
block_device_allocate_retries_interval = 3	整数値	ブロックデバイスの割り当て失敗時の再試行の間隔(秒単位)。 このオプションを使用すると、ユーザーは連続する再試行の間隔を指定できます。**block_device_allocate_retries** オプションは、再試行の最大数を指定します。 値: - 0: オプションを無効にします。 - 任意の正の整数 (秒単位) でオプションを有効にします。 関連するオプション: - **block_device_allocate_retries**: 再試行回数を制御します。
cert = self.pem	文字列値	SSL 証明書ファイルへのパス。 関連するオプション: - 鍵 (key) - ssl_only - [console] ssl_ciphers - [console] ssl_minimum_version
compute_driver = None	文字列値	仮想化を制御するために使用するドライバーを定義します。 値: - **libvirt.LibvirtDriver** - **fake.FakeDriver** - **ironic.IronicDriver** - **vmwareapi.VMwareVCDriver** - **hyperv.HyperVDriver** - **powervm.PowerVMDriver** - **zvm.ZVMDriver**

設定オプション = デフォルト値	型	説明
compute_monitors = []	リスト値	コンピュートメトリックの取得に使用できるモニターのコンマ区切りリスト。nova.compute.monitors.* namespace の setuptools エントリーポイントからエイリアス/名前を使用できます。namespace が指定されていない場合、cpu.名前空間が後方互換性のために想定されます。 **注記** namespace ごとに 1 つのモニター (例: cpu) のみをロードできます。 値: ● 空のリストは機能を無効にします (デフォルト)。 ● CPU を有効にする値の例 **virt ドライバーのバリアントを使用する帯域幅モニター** compute_monitors = cpu.virt_driver

設定オプション = デフォルト値	型	説明
config_drive_format = iso9660	文字列値	コンフィグドライブの形式。 インスタンスのブート時にアタッチされたメタデータが含まれるコンフィグドライブ形式。 関連するオプション: ● このオプションは、以下のいずれかのオプションが発生した場合に有効となります。 1. **force_config_drive** オプションが **true** に設定されている 2. インスタンスを作成する REST API 呼び出しには、コンフィグドライブオプションの enable フラグが含まれます。 3. インスタンスの作成に使用するイメージにはコンフィグドライブが必要です。これは、そのイメージの **img_config_drive** プロパティーで定義されます。 ● Hyper-V ハイパーバイザーを実行するコンピュートノードは、コンフィグドライブを CD ドライブとして接続するように設定できます。コンフィグドライブを CD ドライブとして割り当てるには、**[hyperv] config_drive_cdrom** オプションを true に設定します。 **非推奨となったバージョン:** 19.0.0 **理由:** このオプションは、libvirt v1.2.17 で解決された libvirt、#1246201 にバグの回避策として追加されました。その結果、このオプションは不要になったり、便利になったりします。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
console_host = <オペレーティングシステムに基づく>	文字列値	このホストのインスタンスへの接続に使用するコンソールプロキシーホスト。これは、コンソールホストの公開されている名前です。 値: ● 現在のホスト名 (デフォルト) またはホスト名を表す文字列。

設定オプション = デフォルト値	型	説明
control_exchange = nova	文字列値	トピックがスコープ設定されるデフォルトの変換。transport_url オプションで指定した変換名で上書きできます。
cpu_allocation_ratio = None	浮動小数点の値	物理 CPU に対する仮想 CPU の割当て比率。 このオプションは、**VCPU** インベントリーの割り当て比率を設定して、Placement API が選択したホストに影響を与えるために使用されます。 a. 注記: > This option does not affect `PCPU` inventory, which cannot be overcommitted. b. 注記: > If this option is set to something *other than* `None` or `0.0`, the allocation ratio will be overwritten by the value of this option, otherwise, the allocation ratio will not change. Once set to a non-default value, it is not possible to "unset" the config to get back to the default behavior. If you want to reset back to the initial value, explicitly specify it to the value of `initial_cpu_allocation_ratio`. 値: ● 有効な正の整数または浮動小数点値 関連するオプション: ● **initial_cpu_allocation_ratio**
daemon = False	ブール値	バックグラウンドプロセスとして実行します。
debug = False	ブール値	true に設定すると、ログレベルはデフォルトの INFO レベルではなく DEBUG に設定されます。

設定オプション = デフォルト値	型	説明
default_access_ip_network_name = None	文字列値	インスタンスのアクセス IP を設定するために使用されるネットワークの名前。選択する IP が複数ある場合には、任意の IP が選択されます。 値: ● none (デフォルト) ● ネットワーク名を表す文字列。
default_availability_zone = nova	文字列値	Compute サービスのデフォルトのアベイラビリティーゾーン。 このオプションは、**nova-compute** サービスのデフォルトアベイラビリティーゾーンを決定します。 値: ● 既存のアベイラビリティーゾーン名を表す文字列。
default_ephemeral_format = None	文字列値	ephemeral_volume のデフォルトの形式は、作成時にでフォーマットされます。 値: ● **ext2** ● **ext3** ● **ext4** ● **xfs** ● **ntfs** (Windows ゲストのみ)

設定オプション = デフォルト値	型	説明
default_log_levels = ['amqp=WARN', 'amqplib=WARN', 'boto=WARN', 'qpid=WARN', 'sqlalchemy=WARN', 'suds=INFO', 'oslo.messaging=INFO', 'oslo_messaging=INFO', 'iso8601=WARN', 'requests.packages.urllib3.connectionpool=WARN', 'urllib3.connectionpool=WARN', 'websocket=WARN', 'requests.packages.urllib3.util.retry=WARN', 'urllib3.util.retry=WARN', 'keystonemiddleware=WARN', 'routes.middleware=WARN', 'stevedore=WARN', 'taskflow=WARN', 'keystoneauth=WARN', 'oslo.cache=INFO', 'oslo_policy=INFO', 'dogpile.core.dogpile=INFO', 'glanceclient=WARN', 'oslo.privsep.daemon=INFO']	リスト値	logger=LEVEL ペアのパッケージロギングレベルのリスト。このオプションは、log_config_append が設定されている場合は無視されます。

設定オプション = デフォルト値	型	説明
default_schedule_zone = None	文字列値	インスタンスのデフォルトのアベイラビリティーゾーン。 このオプションは、インスタンスの作成時にユーザーが指定しなかった場合に使用される、インスタンスのデフォルトのアベイラビリティーゾーンを決定します。インスタンスは、ライフタイム間、このアベイラビリティーゾーンにバインドされます。 値: ● 既存のアベイラビリティーゾーン名を表す文字列。 ● none: つまり、あるコンピュートノードから別のアベイラビリティーゾーンに移動すると、インスタンスがそのアベイラビリティーゾーンから別のアベイラビリティーゾーンに移動できます。 関連するオプション: ● **[cinder]/cross_az_attach**

設定オプション = デフォルト値	型	説明
disk_allocation_ratio = None	浮動小数点の値	物理ディスクに対する仮想ディスクの割当て比率。 このオプションは、**DISK_GB** インベントリーの割り当て比率を設定して、Placement API で選択したホストに影響を与えるのに使用します。 設定されると、1.0 を超える比率により、利用可能な物理ディスクが過剰にサブスクライブされます。これは、スパースイメージや圧縮イメージなどの仮想ディスク全体を使用しないイメージで作成されたインスタンスの効率的なパックに役立ちます。インスタンス以外の使用のためにディスクのパーセンテージを保持するために、0.0 から 1.0 までの値に設定できます。 a. 注記: > If the value is set to `>1`, we recommend keeping track of the free disk space, as the value approaching `0` may result in the incorrect functioning of instances using it at the moment. b. 注記: > If this option is set to something *other than* `None` or `0.0`, the allocation ratio will be overwritten by the value of this option, otherwise, the allocation ratio will not change. Once set to a non-default value, it is not possible to "unset" the config to get back to the default behavior. If you want to reset back to the initial value, explicitly specify it to the value of `initial_disk_allocation_ratio`. 値: ● 有効な正の整数または浮動小数点値 関連するオプション: ● **initial_disk_allocation_ratio**

設定オプション = デフォルト値	型	説明
enable_new_services = True	ブール値	このホストで新しい nova-compute サービスを自動的に有効にします。 新しい nova-compute サービスが起動すると、有効なサービスとしてデータベースに登録されます。新しい Compute サービスを無効な状態で登録し、後で有効にすると便利な場合があります。このオプションは、nova-compute サービスに対してのみこの動作を設定します。nova-conductor、nova-scheduler、nova-osapi_compute などの他のサービスは自動的に無効にしません。 値: ● **True**: 新しい Compute サービスはそれぞれ、登録するとすぐに有効になります。 ● **False**: コンピュートサービスは、os-services REST API コールで有効にするか、**nova service-enable <hostname> <binary>** で CLI で有効にする必要があります。有効にしないと、使用する準備ができていません。
enabled_apis = ['osapi_compute', 'metadata']	リスト値	デフォルトで有効にする API のリスト。
enabled_ssl_apis = []	リスト値	SSL が有効になっている API のリスト。 Nova は API サーバーの SSL サポートを提供します。enabled_ssl_apis オプションでは、SSL サポートを設定できます。
executor_thread_pool_size = 64	整数値	エグゼキューターがスレッディングまたはイベントレットの場合のエグゼキュータースレッドプールのサイズ。
fatal_deprecations = False	ブール値	非推奨の致命的なステータスを有効または無効にします。
flat_injected = False	ブール値	このオプションは、起動する前に、ネットワーク設定情報が仮想マシンに挿入されるかどうかを決定します。元々は nova-network のみが使用するように設計されていましたが、ネットワーク情報が VM に挿入されるかどうかを制御するために vmware virt ドライバーによっても使用されます。libvirt 仮想ドライバーは、config_drive を使用してネットワークを設定し、ネットワーク情報が仮想マシンに挿入されるかどうかを制御する際にもそれを使用します。

設定オプション = デフォルト値	型	説明
force_config_drive = False	ブール値	コンフィグドライブで注入を強制します。 このオプションが true に設定されている場合には、デフォルトで、強制的に有効になります。有効にしないと、ユーザーは REST API またはイメージメタデータ属性を介してコンフィグドライブを有効にできます。起動されたインスタンスはこのオプションの影響を受けません。 値: ● True: REST API 呼び出しのユーザー入力に関係なく、コンフィグドライブを強制的に使用させます。 ● False: コンフィグドライブを強制的に使用しないでください。コンフィグドライブは、REST API またはイメージメタデータ属性を使用して有効にできます。 関連するオプション: ● **mkisofs_cmd** フラグを使用して、genisoimage プログラムをインストールするパスを設定します。genisoimage が nova-compute サービスと同じパスにある場合は、このフラグを設定する必要はありません。 ● Hyper-V でコンフィグドライブを使用するには、**mkisofs_cmd** の値を mkisofs.exe インストールへの完全パスに設定する必要があります。さらに、hyperv 設定セクションの qemu_img_cmd の値を、qemu-img コマンドインストールへのフルパスに設定する必要があります。

設定オプション = デフォルト値	型	説明
force_raw_images = True	ブール値	バッキングイメージの raw 形式への強制変換。 値: ● True: バッキングイメージファイルは raw イメージ形式に変換されます。 ● False: バッキングイメージファイルが変換されない 関連するオプション: ● **compute_driver**: libvirt ドライバーのみがこのオプションを使用します。 ● **[libvirt]/images_type**: images_type が rbd の場合には、このオプションを False に設定することはできません。詳細は、バグ https://bugs.launchpad.net/nova/+bug/1816686 を参照してください。
graceful_shutdown_timeout = 60	整数値	サーバーを正常にシャットダウンするまでのタイムアウトを指定します。値をゼロにすると、endless の待機を意味します。
heal_instance_info_cache_interval = 60	整数値	インスタンスネットワーク情報キャッシュの更新の間隔。 各コンピュートノードが、そのすべてのインスタンスのネットワーク情報に対して Neutron をクエリーするタスクを実行してから、その情報で Nova db を更新する秒数。nova は、このオプションが 0 に設定されている場合、キャッシュを更新しません。 キャッシュを更新しない場合、メタデータサービスと nova-api エンドポイントがインスタンスに関する間違ったネットワークデータをプロキシー送信します。そのため、このオプションを 0 に設定することは推奨されません。 値: ● 正の整数 (秒単位)。 ● 任意の値 ⇐0 は同期を無効にします。これは、推奨されません。

設定オプション = デフォルト値	型	説明
host = <based on operating system>	文字列値	このホストのホスト名、FQDN、または IP アドレス。 以下のように使用します。 ● nova-compute ワーカーの oslo.messaging キュー名 ● この値は、neutron に送信される binding_host に使用します。つまり、neutron エージェントを使用する場合は、ホストに同じ値を指定する必要があります。 ● Cinder ホストのアタッチ情報 AMQP キー内で有効である必要があります。 値: ● ホスト名、FQDN、または IP アドレスの文字列デフォルトはこのホストのホスト名です。
initial_cpu_allocation_ratio = 16.0	浮動小数点の値	物理 CPU に対する最初の仮想 CPU の比率。 これは、特定の nova-compute サービス用に **computes_nodes** テーブルレコードを最初に作成した場合にのみ使用されます。 詳細と使用シナリオは、https://docs.openstack.org/nova/latest/admin/configuration/schedulers.html を参照してください。 関連するオプション: ● **cpu_allocation_ratio**
initial_disk_allocation_ratio = 1.0	浮動小数点の値	物理ディスクに対する最初の仮想ディスクの比率。 これは、特定の nova-compute サービス用に **computes_nodes** テーブルレコードを最初に作成した場合にのみ使用されます。 詳細と使用シナリオは、https://docs.openstack.org/nova/latest/admin/configuration/schedulers.html を参照してください。 関連するオプション: ● **disk_allocation_ratio**

設定オプション = デフォルト値	型	説明
initial_ram_allocation_ratio = 1.5	浮動小数点の値	物理メモリーに対する初期仮想 RAM の比率。 これは、特定の nova-compute サービス用に **computes_nodes** テーブルレコードを最初に作成した場合にのみ使用されます。 詳細と使用シナリオは、https://docs.openstack.org/nova/latest/admin/configuration/schedulers.html を参照してください。 関連するオプション: ● **ram_allocation_ratio**
injected_network_template = $pybasedir/nova/virt/interfaces.template	文字列値	**/etc/network/interfaces** テンプレートへのパスです。 **/etc/network/interfaces**-style ファイルのテンプレートファイルへのパス。これにより、DHCP サーバーがない環境でネットワーク接続を設定する方法が提供されます。 テンプレートは Jinja2 テンプレートエンジンを使用してレンダリングされ、**interfaces** と呼ばれる最上位のキーを受け取ります。このキーには、インターフェイスごとに 1 つのディクショナリーのリストが含まれます。 詳細は、cloudinit ドキュメントを参照してください。 > https://cloudinit.readthedocs.io/en/latest/topics/datasources.html 値: ● Debian **/etc/network/interfaces** ファイルの Jinja2 形式のテンプレートへのパス。これは、Debian-derived 以外のゲストを使用している場合でも該当します。 関連するオプション: ● **flat_inject**: nova がコンフィグドライブを通じて提供されるメタデータにネットワーク設定情報を埋め込むには、これを **True** に設定する必要があります。

設定オプション = デフォルト値	型	説明
instance_build_timeout = 0	整数値	インスタンスがビルド可能な最大時間 (秒単位)。 このタイマーの期限が切れると、インスタンスのステータスは ERROR に変わります。このオプションを有効にすると、インスタンスが長期間 BUILD 状態のままにならないようにします。 値: ● 0: オプションを無効にします (デフォルト) ● 任意の正の整数 (秒単位) でオプションを有効にします。
instance_delete_interval = 300	整数値	失敗したインスタンスファイルの削除を再試行する間隔。 このオプションは **maximum_instance_delete_attempts** によって異なります。このオプションは、再試行する頻度を指定します。一方、**maximum_instance_delete_attempts** は、作成可能な再試行回数の最大値を指定します。 値: ● 0: デフォルトの間隔で実行されます。 ● 任意の値 < 0: オプションを無効にします。 ● 正の整数 (秒単位)。 関連するオプション: ● instance_cleaning_opts グループからの **maximum_instance_delete_attempts**。
instance_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンスの形式。

設定オプション = デフォルト値	型	説明
instance_name_template = instance-%08x	文字列値	インスタンス名の生成に使用するテンプレートの文字列。 このテンプレートは、インスタンスのデータベース名の作成を制御します。これは (Horizon または CLI を介して) インスタンスの作成時に入力する表示名では **ありません**。新しいデプロイメントでは、デフォルト値 (データベースの自動増分を使用する) を、**instance-%(uuid)s** などのインスタンスの属性を使用する別の値に変更することが推奨 されます。これを変更すると、すでにデプロイメントにインスタンスがある場合、デプロイメントは中断します。 値: ● インスタンスのデータベース ID (デフォルトなど) を使用する文字列 ● 名前付きのデータベース列のリストが含まれる文字列 (例: **%(id)d** または **%(uuid)s** または **%(hostname)s**。
instance_usage_audit = False	ブール値	このオプションは、定期的な compute.instance.exists 通知を有効にします。各コンピュートノードは、システムの使用状況データを生成するように設定する必要があります。これらの通知は、OpenStack Telemetry サービスが消費されます。
instance_usage_audit_period = month	文字列値	インスタンスの使用状況を生成する期間。指定期間にオプションのオフセットを定義するには、@ 文字の後にオフセットを定義する数字を追加します。 値: ● 期間 (例: **hour**、**day**、**month**、または **year**) ● オフセットの期間 (例: **month@15** を使用すると、月次監査はその月の15 日から始まります)。
instance_uuid_format = [instance: %(uuid)s]	文字列値	ログメッセージで渡されるインスタンス UUID の形式。

設定オプション = デフォルト値	型	説明
instances_path = $state_path/instances	文字列値	インスタンスがハイパーバイザーのディスクに保存されるかどうかを指定します。ローカルに割り当てられたストレージまたは NFS のディレクトリーを参照できます。 値: ● $state_path/instances。state_path は、nova の状態を維持する最上位ディレクトリーを指定する設定オプション (デフォルト) またはディレクトリーパスを表す任意の文字列です。 関連するオプション: ● **[workarounds]/ensure_libvirt_rbd_instance_dir_cleanup**
internal_service_availability_zone = internal	文字列値	内部サービスのアベイラビリティーゾーン。 **nova-scheduler**、**nova-conductor** など、各種の内部 nova サービスのアベイラビリティーゾーンを決定します。 値: ● 既存のアベイラビリティーゾーン名を表す文字列。
key = None	文字列値	SSL キーファイル (証明書とは別の場合)。 関連するオプション: ● cert
live_migration_retry_count = 30	整数値	live_migration における最大 1 秒の再試行回数。これは、エラーが出たときの iptables への再試行回数を指定します。これは、ユーザーが同じホストにライブマイグレーションの要求を継続的に送信し、iptables に同時要求する場合に発生します。 値: ● 再試行数を表す正の整数。

設定オプション = デフォルト値	型	説明
log-config-append = None	文字列値	ロギング設定ファイルの名前。このファイルは、既存のロギング設定ファイルに追加されます。ロギング設定ファイルの詳細は、Python のロギングモジュールのドキュメントを参照してください。ロギング設定ファイルを使用すると、すべてのロギング設定が設定ファイルで設定され、その他のロギング設定オプションは無視されます (例:log-date-format)。
log-date-format = %Y-%m-%d %H:%M:%S	文字列値	ログレコードの %%(asctime)s のフォーマット文字列を定義します。デフォルト:%(default)sこのオプションは、log_config_append が設定されている場合は無視されます。
log-dir = None	文字列値	(オプション)log_file の相対パスに使用されるベースディレクトリー。このオプションは、log_config_append が設定されている場合は無視されます。
log-file = None	文字列値	(オプション) ロギング出力を送信するログファイルの名前。デフォルトが設定されていない場合、ロギングは use_stderr で定義されているように stderr に送信されます。このオプションは、log_config_append が設定されている場合は無視されます。
log_options = True	ブール値	サービスの起動時 (DEBUG レベル) に、登録されているすべてのオプションの値をログに記録することを有効または無効にします。
log_rotate_interval = 1	整数値	ログファイルがローテーションされるまでの時間。このオプションは、log_rotation_type が interval に設定されていない限り無視されます。
log_rotate_interval_type = days	文字列値	ローテーション間隔の種別。次のローテーションをスケジューリングする際に、最後のファイル変更の時刻 (またはサービスの起動時刻) が使用されます。
log_rotation_type = none	文字列値	ログローテーションの種別。
logging_context_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s	文字列値	コンテキスト付きログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。

設定オプション = デフォルト値	型	説明
logging_debug_format_suffix = %(funcName)s %(pathname)s:%(lineno)d	文字列値	メッセージのロギングレベルが DEBUG の場合にログメッセージに追加する追加のデータ。oslo_log.formatters.ContextFormatter により使用されます。
logging_default_format_string = %(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s	文字列値	コンテキストが定義されていない場合に、ログメッセージに使用するフォーマット文字列。oslo_log.formatters.ContextFormatter により使用されます。
logging_exception_prefix = %(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s	文字列値	この形式で、例外出力の各行の前に接頭辞が付けられます。oslo_log.formatters.ContextFormatter により使用されます。
logging_user_identity_format = %(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s	文字列値	logging_context_format_string で使用される %(user_identity)s のフォーマット文字列を定義します。oslo_log.formatters.ContextFormatter により使用されます。
long_rpc_timeout = 1800	整数値	このオプションを使用すると、RPC 呼び出しに長い時間がかかる可能性がある別のタイムアウト値を設定できます。これを設定すると、他のサービスへの RPC 呼び出しでは、グローバルな rpc_response_timeout 値ではなく、タイムアウト (秒単位) にこの値が使用されます。 この値を使用する RPC 呼び出しを使用した操作: ● ライブマイグレーション ● scheduling ● コンピュートサービスの有効化/終了 ● イメージの事前キャッシュ ● スナップショットベース/セル間のサイズ変更 ● サイズ変更/コールドマイグレーション ● ボリュームの接続 関連するオプション: ● rpc_response_timeout

設定オプション = デフォルト値	型	説明
max_concurrent_builds = 10	整数値	nova-compute で同時に実行するインスタンスビルドの最大数を制限します。Compute サービスは、インスタンスの構築を無限に試行できます (確認を求められた場合)。この制限は、コンピュートノードに無制限のインスタンスを同時に構築しないように強制されます。この値は、コンピュートノードごとに設定できます。 以下の値を使用できます。 ● 0: 無制限として処理されます。 ● 同時ビルドの最大数を表す正の整数。
max_concurrent_live_migrations = 1	整数値	同時に実行するライブマイグレーションの最大数。この制限は、送信ライブマイグレーションがホスト/ネットワークに圧倒され、障害が発生するのを防ぐために実施されます。お使いの環境で安全に、安定していることが不明な場合を除き、これを変更することは推奨されません。 値: ● 0: 無制限として処理されます。 ● 同時に実行するライブマイグレーションの最大数を表す正の整数。
max_concurrent_snapshots = 5	整数値	同時に実行するインスタンススナップショット操作の最大数。この制限は、スナップショットがホスト/ネットワーク/ストレージを圧迫して障害を引き起こすのを防ぐために適用されます。この値は、コンピュートノードごとに設定できます。 以下の値を使用できます。 ● 0: 無制限として処理されます。 ● 最大同時スナップショットを表す任意の正の整数。

設定オプション = デフォルト値	型	説明
max_local_block_devices = 3	整数値	ハイパーバイザーノードでローカルイメージが作成されるデバイスの最大数。 負の値は無制限を意味します。**max_local_block_devices** を 0 に設定すると、ローカルディスクの作成を試行する要求は失敗します。このオプションは、ローカルディスクの数を制限することを目的としています (サーバーの作成時に**imageRef**が使用された結果であるルートローカルディスク、およびその他のエフェメラルディスクとスワップディスク)。0 は、イメージが自動的にボリュームに変換され、ボリュームからインスタンスを起動することを意味するのではなく、ローカルディスクを作成しようとするすべての要求が失敗することを意味します。 値: ● 0: ローカルディスクの作成はできません。 ● 負の数: ローカルディスクの数に制限はありません。 ● 正の数: これらの数のローカルディスクのみを許可します。
max_logfile_count = 30	整数値	ローテーションされたログファイルの最大数。
max_logfile_size_mb = 200	整数値	ログファイルの最大サイズ (MB 単位)。log_rotation_type が size に設定されていない場合は、このオプションは無視されます。
maximum_instance_delete_attempts = 5	整数値	インスタンスのファイルの取得を試行する回数。 このオプションは、実行できる再試行の最大回数を指定します。 値: ● 正の整数は、試行回数を定義します。 関連するオプション: ● **[DEFAULT] instance_delete_interval** を使用して、このオプションを無効にすることができます。
metadata_listen = 0.0.0.0	文字列値	メタデータ API がリッスンする IP アドレス。 メタデータ API サービスは、この IP アドレスで受信要求をリッスンします。

設定オプション = デフォルト値	型	説明
metadata_listen_port = 8775	ポート値	メタデータ API がリッスンするポート。 メタデータ API サービスは、受信要求についてこのポート番号をリッスンします。
metadata_workers = <オペレーティングシステムに基づく>	整数値	メタデータサービスのワーカー数。指定のない場合は、利用可能な CPU の数が使用されます。 メタデータサービスは、マルチプロセス (ワーカー) として実行するように設定することができます。これにより、API リクエストコンカレンシーが増加するとスループットが低下するという問題が解消されます。メタデータサービスは、指定されたプロセス数で実行されます。 以下の値を使用できます。 ● 任意の正の整数 ● none (デフォルト値)
migrate_max_retries = -1	整数値	ライブマイグレーションが失敗する前のライブマイグレーションを再試行する回数。 値: ● == -1 の場合、ホストから実行するまで試行します (デフォルト)。 ● == 0 の場合、再試行は 1 回のみ試行しません。 ● 0 より大きい整数

設定オプション = デフォルト値	型	説明
mkisofs_cmd = genisoimage	文字列値	ISO イメージの作成に使用するツールの名前またはパス。 **mkisofs_cmd** フラグを使用して、**genisoimage** プログラムをインストールするパスを設定 し ます。**genisoimage** がシステムパスにある場合は、デフォルト値を変更する必要はありません。 Hyper-V でコンフィグドライブを使用するには、**mkisofs_cmd** の値を **mkisofs.exe** インストールへの完全パスに設定する必要があります。さらに、hyperv 設定セクションの **qemu_img_cmd** の値を、**qemu-img** コマンドインストールへのフルパスに設定する必要があります。 値: ● ISO イメージ作成者の名前 (nova-compute サービスと同じディレクトリー) ● ISO イメージ作成者プログラムへのパス 関連するオプション: ● このオプションは、コンフィグドライブが有効な場合に有効となります。 ● Hyper-V でコンフィグドライブを使用するには、Hyperv 設定セクションの **qemu_img_cmd** の値を、**qemu-img** コマンドインストールへの完全パスに設定する必要があります。
my_block_storage_ip = $my_ip	文字列値	ブロックストレージネットワークへの接続に使用される IP アドレス。 値: ● 有効な IP アドレスを含む文字列。デフォルトは、このホストの IP アドレスです。 関連するオプション: ● my_ip - my_block_storage_ip が設定されていない場合は、my_ip の値が使用されます。

設定オプション = デフォルト値	型	説明
my_ip = <オペレーティングシステムに基づく>	文字列値	管理ネットワークへの接続にホストが使用する IP アドレス。 値: ● 有効な IP アドレスを含む文字列。デフォルトは、このホストの IPv4 アドレスです。 関連するオプション: ● my_block_storage_ip
network_allocate_retries = 0	整数値	ネットワーク割り当てを再試行する回数。仮想インターフェイスプラグインが失敗した場合は、ネットワーク割り当ての再試行を試行する必要があります。 値: ● 再試行数を表す正の整数。

設定オプション = デフォルト値	型	説明
non_inheritable_image_properties = ['cache_in_nova', 'bittorrent']	リスト値	スナップショットの作成時にインスタンスから継承すべきではないイメージ属性。 このオプションを使用すると、新たに作成したスナップショットにより継承されない image-properties を選択できます。 a. 注記: > The following image properties are *never* inherited regardless of whether they are listed in this configuration option or not: - cinder_encryption_key_id - cinder_encryption_key_deletion_policy - img_signature - img_signature_hash_method - img_signature_key_type - img_signature_certificate_uuid 値: - 項目がイメージ属性のコンマ区切りリスト。ベースイメージから作成されたスナップショットはそれらを必要としないため、通常、ベースイメージでのみ必要なイメージプロパティーのみをここに含めることができます。 - デフォルトのリスト: cache_in_nova、bittorrent
osapi_compute_listen = 0.0.0.0	文字列値	OpenStack API がリッスンする IP アドレス。 OpenStack API サービスは、この IP アドレスを受信要求をリッスンします。
osapi_compute_listen_port = 8774	ポート値	OpenStack API がリッスンするポートです。 OpenStack API サービスは、受信要求についてこのポート番号をリッスンします。

設定オプション = デフォルト値	型	説明
`osapi_compute_unique_server_name_scope = `	文字列値	一意のインスタンス名のチェックの範囲を設定します。 デフォルトでは、一意の名前がチェックされません。名前チェックのスコープが設定されると、新規インスタンスを起動するか、重複する名前で既存のインスタンスを更新すると 'InstanceExists' エラーが出されます。一意性は大文字と小文字を区別しません。このオプションを設定すると、同じ名前のインスタンス間を ID で区別する必要がないため、エンドユーザーのユーザービリティーが向上します。
osapi_compute_workers = None	整数値	OpenStack API サービスのワーカー数。デフォルトは、利用可能な CPU の数になります。 OpenStack API サービスは、マルチプロセス (ワーカー) として実行するように設定できます。これにより、API リクエストコンカレンシーが増加するとスループットが低下するという問題が解消されます。OpenStack API サービスは、指定されたプロセス数で実行されます。 以下の値を使用できます。 ● 任意の正の整数 ● none (デフォルト値)
password_length = 12	整数値	生成されたインスタンス管理者パスワードの長さ。
periodic_enable = True	ブール値	定期的なタスクを有効にします。 このオプションを true に設定すると、サービスはマネージャーでタスクを定期的に実行することができます。 複数のスケジューラーまたはコンダクターを実行している場合は、1 つのホストでのみ定期的なタスクを実行することができます。この場合、1 台のホストでこのオプションを無効にします。

設定オプション = デフォルト値	型	説明
periodic_fuzzy_delay = 60	整数値	スタンピングを減らすために定期的なタスクスケジューラーを開始するときにランダムに遅延する秒数。 コンピュートワーカーがクラスター全体でプロビジョニング解除された状態で再起動すると、すべて定期的なタスクが同時に実行され、外部サービスで問題が発生します。この動作を軽減するために、periodic_fuzzy_delay オプションを使用すると、定期的なタスクスケジューラーの開始時にランダムな初期遅延が発生する可能性があります。 以下の値を使用できます。 ● 正の整数 (秒単位): ● 0: ランダムな遅延を無効にします。
pointer_model = usbtablet	文字列値	ポインターの種類を指定する一般的なプロパティー。 入力デバイスを使用すると、グラフィカルフレームバッファーとの対話が可能になります。たとえば、カーソル移動のグラフィックタブレットを指定します。 設定されている場合、**hw_input_bus** または **hw_pointer_model** イメージメタデータプロパティーのいずれかがこの設定オプションよりも優先されます。 関連するオプション: ● usbtablet は、VNC が有効または SPICE を有効にし、SPICE エージェントを無効にして設定する必要があります。libvirt で使用する場合は、インスタンスモードを HVM として設定する必要があります。
preallocate_images = none	文字列値	使用するイメージ事前割り当てモード。 イメージの事前割り当てにより、インスタンスの初回プロビジョニング時に、インスタンスイメージのストレージを事前に割り当てることができます。これにより、十分なスペースが利用できない場合に、即時のフィードバックが提供されます。さらに、新しいブロックへの書き込みのパフォーマンスが大幅に向上するはずです。また、断片化が減少するため、事前に書き込まれたブロックへの I/O パフォーマンスも向上する可能性があります。

設定オプション = デフォルト値	型	説明
publish_errors = False	ブール値	エラーイベントの公開を有効または無効にします。
pybasedir = /usr/lib/python3.9/site-packages	文字列値	Nova python モジュールがインストールされているディレクトリー。 このディレクトリーは、ネットワークおよびリモートコンソールアクセス用のテンプレートファイルを保存するために使用されます。また、Nova 内部データの永続化が必要な他の設定オプションのデフォルトパスでもあります。このオプションをデフォルト値から変更する必要がある可能性はあまりありません。 値: ● ディレクトリーへの完全パス。 関連するオプション: ● **state_path**
ram_allocation_ratio = None	浮動小数点の値	物理メモリーに対する仮想メモリーの割当て比率。 このオプションは、**MEMORY_MB** インベントリーの割り当て比率を設定して、Placement API で選択したホストに影響を与えるのに使用します。 a. 注記: > If this option is set to something *other than* `None` or `0.0`, the allocation ratio will be overwritten by the value of this option, otherwise, the allocation ratio will not change. Once set to a non-default value, it is not possible to "unset" the config to get back to the default behavior. If you want to reset back to the initial value, explicitly specify it to the value of `initial_ram_allocation_ratio`. 値: ● 有効な正の整数または浮動小数点値 関連するオプション: ● **initial_ram_allocation_ratio**
rate_limit_burst = 0	整数値	rate_limit_interval ごとのログ記録されたメッセージの最大数。

設定オプション = デフォルト値	型	説明
rate_limit_except_level = CRITICAL	文字列値	レート制限で使用されるログレベル名:CRITICAL、ERROR、INFO、WARNING、DEBUG または空の文字列。rate_limit_except_level 以上のレベルのログはフィルターされません。空の文字列は、すべてのレベルがフィルターされることを意味します。
rate_limit_interval = 0	整数値	ログのレート制限の間隔 (秒数)。
reboot_timeout = 0	整数値	インスタンスが自動的にハードリブートされる間隔。 ソフトリブートを行うと、ソフトリブートタスクが終了しないように、ゲストカーネルが完全にハングしている可能性があります。このオプションを秒単位の時間に設定すると、インスタンスが N 秒より長く再起動状態でスタックしている場合、インスタンスは自動的にハードリブートされます。 値: ● 0: オプションを無効にします (デフォルト)。 ● 任意の正の整数 (秒単位) でオプションを有効にします。

設定オプション = デフォルト値	型	説明
reclaim_instance_interval = 0	整数値	削除されたインスタンスを回収する間隔。 0 より大きい値を指定すると、インスタンスの SOFT_DELETE が有効になります。このオプションは、削除するサーバーが SOFT_DELETED 状態に配置されるかどうかを決定します。この値が 0 より大きい場合、削除されたサーバーはすぐには削除されず、代わりに、古くなるまで (削除された時間が reclaim_instance_interval の値よりも大きい) キューに入れられます。サーバーは、復元アクションを使用して削除キューから回復できます。削除されたサーバーが reclaim_instance_interval の値よりも長い場合は、Compute サービスの定期的なタスクによって自動的に削除されます。 このオプションは API と compute ノードの両方から読み取られ、グローバルに設定する必要があります。設定しないと、サーバーは API でソフト削除された状態になり、compute ノードで実際に回収 (削除) されることがありません。 a. 注記: このオプションを使用する場合は、**cinder** 認証オプション (**auth_type**、**auth_url**、**username** など) も設定する必要があります。再利用は定期的なタスクで発生するため、SOFT_DELETED サーバーに接続されたボリュームをクリーンアップするためのユーザートークンがないため、nova は cinder でこれらのリソースをクリーンアップするための管理者ロールアクセスで設定する必要があります。 値: ● 0 より大きい正の整数 (秒単位) は、このオプションを有効にします。 ● 任意の値 ⇐0 は、オプションを無効にします。 関連するオプション: ● cinder 再利用プロセス中にサーバーに接続されたボリュームをクリーンアップするための認証オプション
record = None	文字列値	このホストで実行されているプロキシーサービス (VNC、スパイス、シリアルなど) によって送受信される WebSocket フレームを保存するために使用されるファイル名。これが設定されていない場合、録画は行われません。

設定オプション = デフォルト値	型	説明
report_interval = 10	整数値	特定のハイパーバイザーのサービスの状態が報告される頻度を示す秒数。nova は、デプロイメントの全体的な健全性を判断するためにこれを把握しておく必要があります。 関連するオプション: ● service_down_time report_interval は service_down_time 未満である必要があります。service_down_time が report_interval 未満の場合、サービスはほとんど報告されないため、通常はダウンしていると見なされます。
rescue_timeout = 0	整数値	インスタンスが RESCUE のままになるまで待機する間隔。 値: ● 0: オプションを無効にします (デフォルト) ● 任意の正の整数 (秒単位) でオプションを有効にします。
reserved_host_cpus = 0	整数値	ホストのプロセス用に確保するホスト CPU の数。 ホストリソースの使用量は、compute ノードで実行されている nova-compute から継続的にスケジューラーに報告されます。この値は、配置に報告される **reserved** 値を決定するために使用されます。 **[compute] cpu_shared_set** または **[compute] cpu_dedicated_set** 設定オプションが定義されている場合は、このオプションは設定できません。これらのオプションが定義されると、これらの値に含まれていないホスト CPU はホスト用に予約済みとみなされます。 値: ● ホスト用に確保する物理 CPU の数を表す正の整数。 関連するオプション: ● **[compute] cpu_shared_set** ● **[compute] cpu_dedicated_set**

設定オプション = デフォルト値	型	説明
reserved_host_disk_mb = 0	整数値	ホストで常に利用できるようにするディスクリソースの容量 (MB 単位)。ディスク使用量は、compute ノードで実行されている nova-compute からスケジューラーに再び報告されます。ディスクリソースが available と見なされないようにするには、このオプションを使用して、そのホストのディスク領域を確保できます。 値: ● ホスト用に予約するディスクの量を MB 単位で表す正の整数。
reserved_host_memory_mb = 512	整数値	ホストプロセスが常に使用できるようにホスト用に予約するメモリーの量 (MB 単位)。ホストリソースの使用量は、compute ノードで実行されている nova-compute から継続的にスケジューラーに報告されます。ホストメモリーが使用可能であると見なされないようにするために、このオプションはホスト用にメモリーを予約するために使用されます。 値: ● ホスト用に予約するメモリーの量を MB 単位で表す正の整数。
reserved_huge_pages = None	dict 値	NUMA ホストセルごとに予約するヒュージページ数/大きいメモリーページ数。 値: ● NUMA ノード ID、ページサイズを反映する有効な key=value のリスト **(デフォルトの単位は KiB) と予約するページ数以下に例を示します。** reserved_huge_pages = node:0,size:2048,count:64 reserved_huge_pages = node:1,size:1GB,count:1 > In this example we are reserving on NUMA node 0 64 pages of 2MiB and on NUMA node 1 1 page of 1GiB.

設定オプション = デフォルト値	型	説明
resize_confirm_window = 0	整数値	N 秒後にサイズを自動的に確認します。 機能のサイズを変更すると、サイズ変更前に既存のサーバーが追加されます。サイズ変更が完了すると、ユーザーがサイズ変更を確認するよう要求されます。ユーザーは、すべての変更を確認するか、元に戻すことができます。サイズ変更によって元のサーバーが削除され、サーバーのステータスがサイズ変更からアクティブに変更されていることを確認します。このオプションを期間 (秒単位) に設定すると、サーバーがその時間より長くサイズ変更された状態にある場合は、サイズ変更が自動的に確認されます。 値: ● 0: オプションを無効にします (デフォルト) ● 任意の正の整数 (秒単位) でオプションを有効にします。
resize_fs_using_block_device = False	ブール値	ブロックデバイスでファイルシステムのサイズ変更を有効にします。 有効にすると、ブロックデバイスでイメージにアクセスして、ファイルシステムのサイズ変更を試行します。これはホストによって行われ、イメージに cloud-init のバージョンが含まれる場合は不要です。考えられるメカニズムには、nbd ドライバー (qcow および raw の場合) またはループ (raw の場合) が必要です。
resume_guests_state_on_host_boot = False	ブール値	このオプションは、ホストの再起動前に実行されていたゲストを起動するかどうかを指定します。これにより、Nova コンピュートノードのすべてのインスタンスが、compute ノードがブートまたは再起動されるたびに状態を再開します。
rootwrap_config = /etc/nova/rootwrap.conf	文字列値	rootwrap 設定ファイルへのパス。 root ラッパーの目的は、サービス固有の非特権ユーザーが可能な限り多くのアクションを root ユーザーとして実行することを許可することです。ここで使用される設定ファイルは、sudoers エントリーで定義される設定ファイルと一致する必要があります。
rpc_conn_pool_size = 30	整数値	RPC 接続プールのサイズ。

設定オプション = デフォルト値	型	説明
rpc_ping_enabled = False	ブール値	ping 呼び出しに応答するエンドポイントを追加します。エンドポイントの名前は oslo_rpc_server_ping です。
rpc_response_timeout = 60	整数値	呼び出しからの応答を待つ秒数。
run_external_periodic_tasks = True	ブール値	一部の定期的なタスクは個別のプロセスで実行できます。ここで実行すべきですか？
running_deleted_instance_action = reap	文字列値	Compute サービスは、データベースで削除されたインスタンスを定期的にチェックし、compute ノードで実行を続けます。上記のオプションを使用すると、このようなインスタンスが特定されたときにアクションを実行できます。 関連するオプション: ● **running_deleted_instance_poll_interval** ● **running_deleted_instance_timeout**
running_deleted_instance_poll_interval = 1800	整数値	クリーンアップ操作の実行間隔 (秒単位)。0 に設定すると、上記のチェックは無効になります。"running_deleted_instance _action" が "log" または "reap" に設定されている場合、0 を超える値を設定する必要があります。 値: ● 任意の正の整数 (秒単位) でオプションを有効にします。 ● 0: オプションを無効にします。 ● 1800: デフォルト値。 関連するオプション: ● running_deleted_instance_action

設定オプション = デフォルト値	型	説明
running_deleted_instance_timeout = 0	整数値	データベースで削除済みとしてマークされたインスタンスがクリーンアップの対象になるまで待機する時間間隔 (秒単位)。 値: ● 秒単位の任意の正の整数 (デフォルトは 0)。 関連するオプション: ● "running_deleted_instance_action"
scheduler_instance_sync_interval = 120	整数値	インスタンスのビューが nova と同期していることを確認するために、スケジューラーに現在のインスタンス UUID のリストを送信する間隔。 CONF オプション **scheduler_tracks_instance_changes** が False の場合は、同期呼び出しは行われません。そのため、このオプションを変更しても効果はありません。 同期が不足する状況が非常に一般的でない場合は、この間隔を増やして送信する RPC メッセージの数を減らします。同様に、同期の問題が問題になる場合は、間隔を低くしてより頻繁にチェックできます。 値: ● 0: デフォルトの間隔で実行されます。 ● 任意の値 < 0: オプションを無効にします。 ● 正の整数 (秒単位)。 関連するオプション: ● このオプションは、**scheduler_tracks_instance_changes** が False に設定されている場合には影響はありません。

設定オプション = デフォルト値	型	説明
service_down_time = 60	整数値	起動しているサービスの前回のチェックインからの最大時間 (秒単位) 各コンピュートノードは、指定されたレポート間隔に基づいてデータベースのステータスを定期的に更新します。コンピュートノードが service_down_time を超えるステータスを更新しない場合、コンピュートノードは down と見なされます。 関連するオプション: ● report_interval (service_down_time は report_interval より小さくすることはできません)
servicegroup_driver = db	文字列値	このオプションは、servicegroup サービスに使用するドライバーを指定します。 nova の ServiceGroup API により、コンピュートノードのステータスの確認が可能になります。nova-compute デーモンを実行するコンピュートワーカーが起動すると、このデーモンは join API を呼び出してコンピュートグループに参加します。nova スケジューラーなどのサービスは、ServiceGroup API に対してクエリーを実行して、ノードが有効かどうかをチェックすることができます。ServiceGroup クライアントドライバーは、内部的にコンピュートワーカーのステータスを自動的に更新します。このサービスには複数のバックエンド実装があります: Database ServiceGroup ドライバーと Memcache ServiceGroup ドライバー。 関連するオプション: ● **service_down_time**(アップサービスの最後のチェックインからの最大時間)

設定オプション = デフォルト値	型	説明
shelved_offload_time = 0	整数値	退避するインスタンスがホストから削除されるまでの時間。 デフォルトでは、このオプションは 0 に設定され、退避操作直後にハイパーバイザーから退避したインスタンスが削除されます。そうでない場合には、インスタンスは shelved_offload_time (秒単位) の値のために保持されるため、unshelve アクションが高速になると、定期的なタスクは、sheed_offload_time に合格した後にハイパーバイザーからインスタンスを削除します。 値: ● 0: インスタンスは、退避直後にオフロードされます。 ● 任意の値 < 0: インスタンスはオフロードされません。 ● 秒単位の整数: インスタンスは、オフロードされるまでの指定の秒数に存在します。
shelved_poll_interval = 3600	整数値	退避したインスタンスをオフロードする間隔。 定期的なタスクは、selloed_poll_interval 毎の秒数で実行され、退避されたインスタンスがないかどうかを確認します。**shelved_offload_time** の設定値に基づいて、退避したインスタンスが見つかると、退避したインスタンスをオフロードします。詳細については、**shelved_offload_time** 設定オプションの説明を確認してください。 値: ● 任意の値 ⇐ 0: オプションを無効にします。 ● 正の整数 (秒単位)。 関連するオプション: ● **shelved_offload_time**

設定オプション = デフォルト値	型	説明
shutdown_timeout = 60	整数値	インスタンスがクリーンシャットダウンを実行するまで待機する合計時間 (秒単位)。 これは、VM がクリーンシャットダウンを実行できる全体の期間 (秒単位) を決定します。停止、レスキュー、シェルフ、再構築の操作を実行しているときに、このオプションを設定すると、インスタンスの電源がオフになる前に、VM が制御されたシャットダウンを実行できるようになります。デフォルトのタイムアウトは 60 秒です。値 0 (ゼロ) は、ゲスト OS のクリーンアップの機会がなく、ゲストの電源がすぐにオフになることを意味します。 このタイムアウト値は、os_shutdown_timeout でイメージごとに上書きすることが可能です。これは、異なるタイプのオペレーティングシステムでクリーンにシャットダウンするために必要な時間を指定するイメージのメタデータ設定です。 値: ● 正の整数または 0 (デフォルト値は 60)。
source_is_ipv6 = False	ブール値	ソースホストが IPv6 でアドレス指定されている場合は、True に設定します。
ssl_only = False	ブール値	暗号化されていない接続を許可しない。 関連するオプション: ● cert ● 鍵 (key)

設定オプション = デフォルト値	型	説明
state_path = $pybasedir	文字列値	Nova の状態を維持する最上位のディレクトリー。 このディレクトリーは、Nova の内部状態を保存するために使用されます。これは、これから派生するさまざまな設定オプションによって使用されます。一部のシナリオでは (移行など)、複数のコンピュートホスト間で共有されるストレージの場所を使用する方が適切です (例: NFS を使用)。**instances_path** オプションが上書きされない限り、このディレクトリーのサイズが非常に大きくなる可能性があります。 値: ● ディレクトリーへの完全パス。デフォルトは、**pybasedir** で提供される値に設定されます。
sync_power_state_interval = 600	整数値	データベースとハイパーバイザー間で電源の状態を同期する間隔。 Nova が実際の仮想マシンの電源状態を確認し、Nova がデータベースにある電源状態をチェックする間隔。ユーザーが仮想マシンの電源を切ると、Nova は API を更新して、仮想マシンの電源が切れています。仮想マシンを予期せずに有効にすると、Nova は仮想マシンをオフにして、システムを想定した状態に維持します。 値: ● 0: デフォルトの間隔で実行されます。 ● 任意の値 < 0: オプションを無効にします。 ● 正の整数 (秒単位)。 関連するオプション: ● **workaround** グループの **handle_virt_lifecycle_events** が false で、このオプションが負の場合、ハイパーバイザーと Nova データベースの間で同期がとれていないインスタンスを手動で同期する必要があります。

設定オプション = デフォルト値	型	説明
sync_power_state_pool_size = 1000	整数値	電源の状態の同期に使用できる greenthreads の数。 このオプションを使用すると、たとえば Ironic を使用するなど、パフォーマンス上の理由から、ハイパーバイザーや実際のインスタンスの電源が入っている、同時リクエストの数を減らすことができます。 値: ● greenthreads 数を表す正の整数。
syslog-log-facility = LOG_USER	文字列値	ログ行を受け取る syslog ファシリティー。このオプションは、log_config_append が設定されている場合は無視されます。
tempdir = None	文字列値	一時作業ディレクトリーを明示的に指定します。
timeout_nbd = 10	整数値	NBD デバイスの起動を待機する時間 (秒単位)。
transport_url = rabbit://	文字列値	メッセージングバックエンドに接続するためのネットワークアドレスおよびオプションのユーザー認証情報 (URL 形式)。想定される形式は次のとおりです。 driver://[user:pass@]host:port[, [userN:passN@]hostN:portN]/virtual_host?query 例:rabbit://rabbitmq:password@127.0.0.1:5672// URL のフィールドの詳細は、https://docs.openstack.org/oslo.messaging/latest/reference/transport.html で oslo_messaging.TransportURL のドキュメントを参照してください。

設定オプション = デフォルト値	型	説明
update_resources_interval = 0	整数値	コンピュートリソースを更新する間隔。 このオプションは、update_available_resource 定期タスクを実行する頻度を指定します。0 未満の数字は、タスクを完全に無効にすることを意味します。これをデフォルトの 0 のままにすると、これはデフォルトの周期間隔で実行されます。正の値に設定すると、約秒数で実行されます。 値: ● 0: デフォルトの間隔で実行されます。 ● 任意の値 < 0: オプションを無効にします。 ● 正の整数 (秒単位)。
use-journal = False	ブール値	ロギング用の journald を有効にします。systemd 環境で実行している場合は、ジャーナルサポートを有効にしたい場合があります。その場合、ログメッセージに加えて構造化されたメタデータが含まれる journal ネイティブプロトコルが使用されます。このオプションは、log_config_append が設定されている場合は無視されます。
use-json = False	ブール値	ロギングに JSON 形式を使用します。このオプションは、log_config_append が設定されている場合は無視されます。
use-syslog = False	ブール値	ロギングに syslog を使用します。既存の syslog 形式は非推奨であり、後に RFC5424 に従うように変更されます。このオプションは、log_config_append が設定されている場合は無視されます。
use_cow_images = True	ブール値	コピーオンライト (cow) イメージの使用を有効にします。 QEMU/KVM では、qcow2 をバッキングファイルとして使用できます。これを無効にすると、バッキングファイルは使用されません。
use_eventlog = False	ブール値	出力を Windows イベントログに記録します。
use_rootwrap_daemon = False	ブール値	root 権限で実行する必要があるコマンドを実行できるデーモンを開始して使用します。このオプションは、通常 nova Compute プロセスを実行するノードで有効化されます。

設定オプション = デフォルト値	型	説明
use_stderr = False	ブール値	出力を標準エラーに記録します。このオプションは、log_config_append が設定されている場合は無視されます。
vcpu_pin_set = None	文字列値	**VCPU** リソースに使用可能なホスト CPU のマスク。 このオプションの動作は、**[compute] cpu_dedicated_set** オプションの定義に依存し、**[compute]cpu_shared_set** オプションの動作に影響します。 ● **[compute] cpu_dedicated_set** が定義されている場合、このオプションを定義するとエラーが発生します。 ● **[compute] cpu_dedicated_set** が定義されていない場合には、このオプションを使用して、**VCPU** リソースのインベントリーを決定し、ピニングされたインスタンスとピニングされていないインスタンスの両方をスケジューリングできるホスト CPU を制限し、**[compute] cpu_shared_set** オプションを上書きします。 値: ● 仮想 CPU の割り当て元となる物理 CPU 番号のコンマ区切りリスト。各要素は、単一の CPU 番号、CPU 番号の範囲、またはその順で、次に除外される CPU 番号のいずれかでなければなりません。 **以前の範囲。以下に例を示します。** vcpu_pin_set = "4-12,^8,15" 関連するオプション: ● **[compute] cpu_dedicated_set** ● **[compute] cpu_shared_set** **非推奨となったバージョン:** 20.0.0 **理由:**このオプションは、[compute]cpu_dedicated_set および [compute]cpu_shared_set オプションに置き換えられました。これらのオプションを使用すると、同じホスト上で固定されたインスタンスと固定されていないインスタンスを共存させることができます (libvirt ドライバーの場合)。

設定オプション = デフォルト値	型	説明
vif_plugging_is_fatal = True	ブール値	VIF プラグインのタイムアウトでインスタンスをブートまたは失敗させるかどうかを決定します。 インスタンスがスケジュールされると、nova はポート更新を Neutron に送信し、ポートの設定を完了するのに必要な情報を提供します。完了すると、Neutron は Nova にポートの設定が完了したことを通知します。この場合、Nova はネットワーク接続が存在するはずですので、インスタンスのブートを再開します。応答が指定の間隔の後に受信されない場合、タイムアウトが発生します。 このオプションは、VIF プラグインのタイムアウトイベントの発生時に Nova が何を行うかを決定します。有効にすると、インスタンスがエラーになります。無効にすると、インスタンスはポートが準備状態にあると仮定してブートを継続します。 値: ● True: VIF プラグインのタイムアウト後にインスタンスが失敗する ● False: VIF プラグインのタイムアウト後もインスタンスの起動を継続する
vif_plugging_timeout = 300	整数値	Neutron VIF プラグインイベントメッセージのタイムアウト。 Neutron vif プラグイベントが到着するのを待ってから続行または失敗するまでの秒数 (**vif_plugging_is_fatal** を参照)。 大規模なタイムアウトエラーが発生している場合は、**[agent]/root_helper_daemonneutron** 設定オプションを使用して neutron エージェントでデーモンモードで rootwrap を実行することを検討してください。 関連するオプション: ● **vif_plugging_is_fatal**: **vif_plugging_timeout** がゼロに設定され、vif_plugging_is_fatal が False に設定されている場合、イベントは全く到達できないことが予想されます。
virt_mkfs = []	多値	一時デバイス用の mkfs コマンドの名前。 形式は <os_type>=<mkfs command> です。

設定オプション = デフォルト値	型	説明
volume_usage_poll_interval = 0	整数値	ボリュームの使用状況を収集する間隔。 このオプションは、volume_usage_poll_interval ごとのボリューム使用状況のキャッシュを秒単位で更新します。 値: ● 0 より大きい正の整数 (秒単位) は、このオプションを有効にします。 ● 任意の値 ⇐0 は、オプションを無効にします。
watch-log-file = False	ブール値	ファイルシステムを監視するように設計されたログハンドラーを使用します。ログファイルが移動または削除されると、このハンドラーは、指定されたパスで新しいログファイルを即時に開きます。これは、log_file オプションを指定し、Linux プラットフォームが使用される場合にのみ有効です。このオプションは、log_config_append が設定されている場合は無視されます。
web = /usr/share/spice-html5	文字列値	Web サーバーによって提供されるコンテンツを含むディレクトリーへのパス。

9.1.2. api

以下の表は、**/etc/nova/nova.conf** ファイルの **[api]** グループで使用できるオプションの概要をまとめたものです。

表9.1 api

設定オプション = デフォルト値	型	説明
auth_strategy = keystone	文字列値	認証に使用するストラテジーを決定します。 **非推奨:** 21.0.0 **理由:** 唯一のデフォルト以外の選択肢である ``noauth2`` は、内部開発およびテストのみを目的としており、デプロイメントでは使用しないでください。このオプションとそのミドルウェアである NoAuthMiddleware[V2_18] は、将来のリリースで削除される予定です。

設定オプション = デフォルト値	型	説明
compute_link_prefix = None	文字列値	この文字列は、OpenStack Compute API へのリンクで返される通常の URL の前に付けられます。空の場合 (デフォルト)、URL は変更せずに返されます。 値: ● 空の文字列 (デフォルト) を含む文字列。
config_drive_skip_versions = 1.0 2007-01-19 2007-03-01 2007-08-29 2007-10-10 2007-12-15 2008-02-01 2008-09-01	文字列値	コンフィグドライブの既存のメタデータを収集する際に、このオプションに表示されないすべてのバージョンで EC2 形式のメタデータが返されます。2.4.0 リリース時点では、利用可能なバージョンは以下のとおりです。 ● 1.0 ● 2007-01-19 ● 2007-03-01 ● 2007-08-29 ● 2007-10-10 ● 2007-12-15 ● 2008-02-01 ● 2008-09-01 ● 2009-04-04 オプションは 1 つの文字列の形式であり、各バージョンはスペースで区切られます。 値: ● 空白で区切られた 0 個以上のバージョンを表す文字列。
dhcp_domain = novalocal	文字列値	インスタンスの FQDN の設定に使用するドメイン名。 インスタンスのホスト名の完全修飾ドメイン名を設定します。設定されていない場合、ドメインのないホスト名のみが設定されます。 値: ● 有効なドメイン名である文字列。

設定オプション = デフォルト値	型	説明
enable_instance_password = True	ブール値	作成、再構築、退避、またはレスキューなどの関連するサーバー API 呼び出しによるインスタンスのパスワードの返送を有効にします。ハイパーバイザーがパスワードの注入に対応していない場合、返されるパスワードは正しくありません。そのため、ハイパーバイザーがパスワードの注入をサポートしていない場合は、これを False に設定します。
glance_link_prefix = None	文字列値	この文字列は、Glance リソースへのリンクで返される通常の URL の前に付けられます。空の場合 (デフォルト)、URL は変更せずに返されます。 値: ● 空の文字列 (デフォルト) を含む文字列。
instance_list_cells_batch_fixed_size = 100	整数値	これは、**instance_list_cells_batch_strategy'** が **fixed** に設定されている場合、各セルデータベースから要求されるインスタンスのバッチサイズを制御します。この必須値は、システムのセルの数や他の要素に関係なく、インスタンスのバッチが要求されるたびに各セルに発行される制限を定義します。**instance_list_cells_batch_strategy** のドキュメントで呼び出される一般的なロジックによると、この最小値はバッチごとに 100 レコードになります。 関連するオプション: ● instance_list_cells_batch_strategy ● max_limit

設定オプション = デフォルト値	型	説明
instance_list_cells_batch_strategy = distributed	文字列値	これは、大規模なインスタンスリスト操作中の、小規模なバッチで API がセルのデータベースをクエリーする方法を制御します。バッチ処理が実行される場合、大規模なインスタンスリスト操作は、最初に各セルデータベースから全体的な API 制限の一部を要求し、必要に応じて各セルからレコードが消費 (返される) されるのと同じバッチサイズを再要求します。バッチが大きいほど、API とデータベース間のチャットが少なくなりますが、データベースからの結果を処理するための無駄な労力が増える可能性があり、ユーザーには返されません。リクエストで多くの小さなデータベースクエリーをユーザーが行わないように、ストラテジーはすべて 100 以上のレコードのバッチサイズを生成します。 関連するオプション: ● instance_list_cells_batch_fixed_size ● max_limit
instance_list_per_project_cells = False	ブール値	有効にすると、API はテナントがインスタンスをマッピングしたセルデータベースのみをクエリーします。これには、各リストの前に API データベースで追加の (高速) クエリーが必要ですが、結果を提供するためにクエリーする必要のあるセルデータベースの数も (潜在的に) 制限されます。セルの数が少ない場合、またはテナントがすべてのセルにインスタンスを持っている可能性が高い場合、これは False である必要があります。多くのセルがある場合、特にテナントをそれらのセルの小さなサブセットに限定する場合、これは True である必要があります。

設定オプション = デフォルト値	型	説明
list_records_by_skipping_down_cells = True	ブール値	False に設定すると、応答しないセルなどのインフラストラクチャーが失敗した場合に、API は 500 エラーを返します。API がダウンセルを省略し、up のセルから結果を返す場合は、このオプションを True に設定します。 API マイクロバージョン 2.69 では、特定のレコードが利用できず、それらのレコードを含む特定のリクエストの結果の一部となる可能性がある、デプロイメントの一時的な状況が生じる可能性があることに注意してください。この場合は、このオプションは無視されます。詳細は、Compute API ガイド (https://docs.openstack.org/api-guide/compute/down_cells.html) の HandlingDownCells セクションを参照してください。
local_metadata_per_cell = False	ブール値	nova-metadata API サービスがセルごとにデプロイされていることを示します。この場合、マルチセルデプロイメントでのパフォーマンスとデータの分離性が向上します。neutron の設定方法に応じて、この設定の使用を検討する必要があります。ネットワークが複数セルにまたがる場合は、nova-metadata API サービスをグローバルに実行しなければならない場合があります。ネットワークがセル境界でセグメント化されている場合は、セルごとに nova-metadata API サービスを実行できます。セルごとに nova-metadata API サービスを実行する際には、対応する nova-metadata API サービスを参照するように各 neutron metadata-agent を設定する必要もあります。
max_limit = 1000	整数値	クエリーは多数のアイテムを返す可能性があるため、このオプションを設定すると、1 つの応答でアイテムの最大数を制限できます。
metadata_cache_expiration = 15	整数値	このオプションは、メタデータをキャッシュする時間 (秒単位) です。0 に設定すると、メタデータキャッシュは完全に無効になります。これは、パフォーマンス上の理由から、通常は推奨しません。この設定を増やすと、負荷が大きい場合にメタデータ API の応答時間が向上します。値が大きいと、メモリーの使用量が長くなる可能性があり、ホストメタデータの変更が有効になるまでの時間が長くなります。

設定オプション = デフォルト値	型	説明
neutron_default_tenant_id = default	文字列値	Neutron API からデフォルトネットワークを取得するためのテナント ID (一部の場所では**プロジェクト ID** とも呼ばれます) を使用します。 関連するオプション: ● use_neutron_default_nets
use_forwarded_for = False	ブール値	True の場合、**X-Forwarded-For** ヘッダーは正規のリモートアドレスとして処理されます。False (デフォルト) の場合、**remote_address** ヘッダーが使用されます。 これは、HTML プロキシーをサニタイズしている場合のみ有効にしてください。
use_neutron_default_nets = False	ブール値	True の場合、TenantNetworkController は Neutron API に対してクエリーを実行して、使用するデフォルトのネットワークを取得します。 関連するオプション: ● neutron_default_tenant_id
vendordata_dynamic_connect_timeout = 5	整数値	外部 REST サービスが接続する最大時間。 値: ● 値が 3 より大きい任意の整数 (TCP パケット再送信タイムアウト)。この待機時間中にインスタンスの開始がブロックされる可能性があるため、この値は小さく保つ必要があることに注意してください。 関連するオプション: ● vendordata_providers ● vendordata_dynamic_targets ● vendordata_dynamic_ssl_certfile ● vendordata_dynamic_read_timeout ● vendordata_dynamic_failure_fatal

設定オプション = デフォルト値	型	説明
vendordata_dynamic_failure_fatal = False	ブール値	動的なベンダーデータを取得する失敗はインスタンスのブートに致命的であるか？ 関連するオプション: ● vendordata_providers ● vendordata_dynamic_targets ● vendordata_dynamic_ssl_certfile ● vendordata_dynamic_connect_timeout ● vendordata_dynamic_read_timeout
vendordata_dynamic_read_timeout = 5	整数値	外部 REST サービスが接続後にデータを返す最大時間。 値: ● 任意の整数。この待機時間でインスタンスの起動がブロックされるため、この値を小さく維持する必要があります。 関連するオプション: ● vendordata_providers ● vendordata_dynamic_targets ● vendordata_dynamic_ssl_certfile ● vendordata_dynamic_connect_timeout ● vendordata_dynamic_failure_fatal

設定オプション = デフォルト値	型	説明
`vendordata_dynamic_ssl_certfile = `	文字列値	動的ベンダーデータ REST サービス SSL 証明書を検証するオプション証明書ファイルまたは CA バンドルへのパス。 値: ● 空の文字列または有効な証明書ファイルへのパス 関連するオプション: ● vendordata_providers ● vendordata_dynamic_targets ● vendordata_dynamic_connect_timeout ● vendordata_dynamic_read_timeout ● vendordata_dynamic_failure_fatal
vendordata_dynamic_targets = []	リスト値	動的ベンダーデータプロバイダーのターゲットリストこれらのターゲットの形式は **<name>@<url>** です。 動的ベンダーデータプロバイダーは、外部の REST サービスと通信し、インスタンスに関する情報のクエリーを行い、メタデータを収集します。この動作は、nova 開発者リファレンスの vendordata.rst ファイルに記載されています。
vendordata_jsonfile_path = None	文字列値	クラウドプロバイダーは、カスタムデータをベンダーデータファイルに保存できます。このファイルは、メタデータサービスを介してインスタンスで利用でき、config-drive のレンダリングで利用できます。このデフォルトのクラス JsonFileVendorData は、このオプションで設定したパスを持つ JSON ファイルからこの情報を読み込みます。このオプションでパスが設定されていない場合、クラスは空のディクショナリーを返します。 このパラメーターを使用して、コンフィグドライブに静的ベンダーデータを提供する場合には、nova-compute サービスをこのオプションと共に設定して、ファイルは nova-compute ホストからアクセスできる必要があることに注意してください。 値: ● データファイルへのパスを表す任意の文字列、または空の文字列 (デフォルト)。

設定オプション = デフォルト値	型	説明
vendordata_providers = ['StaticJSON']	リスト値	ベンダーデータプロバイダーのリスト。 ベンダーデータプロバイダーは、デプロイヤーがデプロイメント固有の configdrive およびメタデータを使用してメタデータを提供する方法です。 vendordata 動的エンドポイントの実装要件についての詳細は、nova developer reference の vendordata.rst ファイルを参照してください。 関連するオプション: ● **vendordata_dynamic_targets** ● **vendordata_dynamic_ssl_certfile** ● **vendordata_dynamic_connect_timeout** ● **vendordata_dynamic_read_timeout** ● **vendordata_dynamic_failure_fatal**

9.1.3. api_database

以下の表は、**/etc/nova/nova.conf** ファイルの **[api_database]** グループで使用できるオプションの概要をまとめたものです。

表9.2 api_database

設定オプション = デフォルト値	型	説明
connection = None	文字列値	データベースへの接続に使用する SQLAlchemy 接続文字列。**nova-compute** サービスにはこれを設定しないでください。
connection_debug = 0	整数値	SQL デバッグ情報の冗長性: 0=なし、100=すべて。
`connection_parameters = `	文字列値	接続時に接続 URL に追加するオプションの URL パラメーター。param1=value1&param2=value2&... として指定します。
connection_recycle_time = 3600	整数値	この秒数より長く接続プールに存在していた接続は、次にプールからチェックアウトされたときに新しい接続に置き換えられます。
connection_trace = False	ブール値	Python スタックトレースをコメント文字列として SQL に追加します。

設定オプション = デフォルト値	型	説明
max_overflow = None	整数値	設定されている場合、この値を SQLAlchemy で max_overflow に使用します。
max_pool_size = None	整数値	プールに開いたままにする SQL 接続の最大数。値を 0 に設定すると無制限を意味します。
max_retries = 10	整数値	起動時のデータベース接続の最大再試行数。再試行回数を無限に指定するには -1 に設定します。
mysql_sql_mode = TRADITIONAL	文字列値	MySQL セッションに使用する SQL モード。このオプション (デフォルトを含む) は、サーバーセット SQL モードを上書きします。サーバー設定で設定された SQL モードを使用するには、これを no value に設定します。例: mysql_sql_mode=
pool_timeout = None	整数値	設定されている場合は、この値を SQLAlchemy で pool_timeout に使用します。
retry_interval = 10	整数値	SQL 接続を開く再試行の間隔。
slave_connection = None	文字列値	スレーブデータベースへの接続に使用する SQLAlchemy 接続文字列。
sqlite_synchronous = True	ブール値	True の場合、SQLite は同期モードを使用します。

9.1.4. barbican

以下の表は、**/etc/nova/nova.conf** ファイルの **[barbican]** グループで使用できるオプションの概要をまとめたものです。

表9.3 barbican

設定オプション = デフォルト値	型	説明
auth_endpoint = http://localhost/identity/v3	文字列値	このエンドポイントを使用して Keystone に接続します。
barbican_api_version = None	文字列値	Barbican API のバージョン。例: "v1"
barbican_endpoint = None	文字列値	このエンドポイントを使用して、Barbican に接続します (例: http://localhost:9311/)。

設定オプション = デフォルト値	型	説明
barbican_endpoint_type = public	文字列値	エンドポイントのタイプを指定します。使用できる値は public、private、および admin です。
number_of_retries = 60	整数値	キー作成の完了のためにポーリングを再試行する回数
retry_delay = 1	整数値	キー作成の完了をポーリングを再試行するまで待機する秒数
verify_ssl = True	ブール値	セキュアでない TLS (https) 要求を指定します。False の場合、サーバーの証明書は検証されません。True の場合は、verify_ssl_path 設定の平均期間を設定できます。
verify_ssl_path = None	文字列値	チェックするバンドルまたは CA 証明書へのパス、または verify_ssh が True である証明書を見つけて使用しようとする要求の場合は None。verify_ssl が False の場合、これは無視されます。

9.1.5. cache

以下の表は、**/etc/nova/nova.conf** ファイルの **[cache]** グループで使用できるオプションの概要をまとめたものです。

表9.4 cache

設定オプション = デフォルト値	型	説明
backend = dogpile.cache.null	文字列値	キャッシュバックエンドモジュール。数百のスレッドサーバーを持つイベントレットベースまたは環境の場合は、プーリングを含む Memcache (oslo_cache.memcache_pool) が推奨されます。100 台未満のスレッドサーバーがある環境では、Memcached(dogpile.cache.memcached) または Redis(dogpile.cache.redis) が推奨されます。サーバーの単一のインスタンスを含むテスト環境は、dogpile.cache.memory バックエンドを使用できます。
backend_argument = []	多値	バックエンドモジュールに指定された引数。このオプションは、dogpile.cache バックエンドに渡される引数ごとに 1 回指定します。例: "<argname>:<value>"
config_prefix = cache.oslo	文字列値	キャッシュリージョンの設定ディクショナリーを構築するための接頭辞。同じ設定名を持つ別の dogpile.cache リージョンがない場合は、これを変更する必要はありません。

設定オプション = デフォルト値	型	説明
dead_timeout = 60	浮動小数点の値	HashClient の内部メカニズムでプールにノードを再度追加しようとするまでの時間 (秒)。
debug_cache_backend = False	ブール値	キャッシュバックエンドからの追加デバッグ (キャッシュキー、get/set/delete/etc コール)。これは、キー/値を含む特定の cache-backend get/set/delete 呼び出しを確認する必要がある場合にのみ非常に便利です。通常、これは false に設定されている必要があります。
enable_retry_client = False	ブール値	失敗を処理する再試行クライアントメカニズムを有効にします。これらのメカニズムは、あらゆる種類の pymemcache クライアントをラップするために使用できます。ラッパーを使用すると、試行回数と attemot 間の待機時間を定義できます。
enable_socket_keepalive = False	ブール値	dogpile の pymemcache バックエンドのソケットキープアライブのグローバルトグル
enabled = False	ブール値	キャッシュのグローバル切り替え。
expiration_time = 600	整数値	dogpile.cache リージョンのキャッシュされた項目に対するデフォルトの TTL (秒単位)。これは、キャッシュの有効期限が明示的に定義されていないキャッシュされたメソッドに適用されます。
hashclient_retry_attempts = 2	整数値	HashClient の内部メカニズムでクライアントが無効としてマークされ、プールから削除されるまで、クライアントを試行する必要がある回数。
hashclient_retry_delay = 1	浮動小数点の値	HashClient の内部メカニズムでの再試行の間に経過する時間 (秒単位)。
memcache_dead_retry = 300	整数値	再試行するまで memcached サーバーが停止されているとみなされる秒数 (dogpile.cache.memcache および oslo_cache.memcache_pool バックエンドのみ)。
memcache_pool_connection_get_timeout = 10	整数値	操作が memcache クライアント接続を取得するのを待つ秒数。
memcache_pool_flush_on_reconnect = False	ブール値	再接続時に memcache がフラッシュされるかどうかをグローバルに切り替えます (oslo_cache.memcache_pool バックエンドのみ)。
memcache_pool_maxsize = 10	整数値	すべての memcached サーバーへのオープン接続の最大数 (oslo_cache.memcache_pool バックエンドのみ)。

設定オプション = デフォルト値	型	説明
memcache_pool_unused_timeout = 60	整数値	memcached への接続が閉じられる前にプールで未使用のままになる秒数 (oslo_cache.memcache_pool バックエンドのみ)。
memcache_servers = ['localhost:11211']	リスト値	ホスト: ポートの形式の Memcache サーバー (dogpile.cache.memcached および oslo_cache.memcache_pool バックエンドのみ)。特定のホストが IPv6 を参照する場合、または特定のドメインが IPv6 を参照する場合、特定のアドレスの前にアドレスファミリー (**inet6**) を付ける必要があります (例: **inet6[::1]:11211**、**inet6:[fd12:3456:789a:1::1]:11211**、**inet6:[controller-0.internalapi]:11211**)。アドレスファミリーが指定されていない場合、使用されるデフォルトのアドレスファミリーは、IPv4 に対応する **inet** になります。
memcache_socket_timeout = 1.0	浮動小数点の値	サーバーへの呼び出しごとにタイムアウト (秒単位)。(dogpile.cache.memcache および oslo_cache.memcache_pool バックエンドのみ)。
proxies = []	リスト値	dogpile.cache バックエンドの機能に影響を与えるプロキシークラス。changing-backend-behavior についての dogpile.cache ドキュメントを参照してください。
retry_attempts = 2	整数値	失敗するまでアクションを試行する回数。
retry_delay = 0	浮動小数点の値	各試行の間にスリープする秒数。
socket_keepalive_count = 1	整数値	接続を切断する前に TCP が送信するキープアライブプローブの最大数。ゼロより大きい正の整数である必要があります。
socket_keepalive_idle = 1	整数値	TCP がキープアライブプローブの送信を開始する前に、接続がアイドル状態を維持する必要がある時間 (秒単位)。ゼロより大きい正の整数である必要があります。
socket_keepalive_interval = 1	整数値	個々のキープアライブプローブ間の時間 (秒単位)。ゼロより大きい正の整数である必要があります。
tls_allowed_ciphers = None	文字列値	TLS コンテキストで作成されたソケットで利用可能な暗号を設定します。OpenSSL 暗号リスト形式の文字列である必要があります。指定されていない場合は、OpenSSL 対応の暗号がすべて利用可能になります。

設定オプション = デフォルト値	型	説明
tls_cafile = None	文字列値	キャッシュサーバーの信頼性を確立するのに必要な PEM 形式の連結された CA 証明書のファイルへのパス。tls_enabled が False の場合、このオプションは無視されます。
tls_certfile = None	文字列値	クライアントの証明書と証明書の信頼性を確立するために必要な任意の数の CA 証明書を含む PEM 形式の単一ファイルへのパス。このファイルは、クライアント側の認証が必要な場合にのみ必要です。tls_enabled が False の場合、このオプションは無視されます。
tls_enabled = False	ブール値	キャッシングサーバーと連携する際に、TLS の使用をグローバルに切り替えます。
tls_keyfile = None	文字列値	クライアントの秘密鍵を含む単一ファイルへのパス。それ以外の場合、秘密鍵は tls_certfile で指定されたファイルから取得されます。tls_enabled が False の場合、このオプションは無視されます。

9.1.6. cinder

以下の表は、**/etc/nova/nova.conf** ファイルの **[cinder]** グループで使用できるオプションの概要をまとめたものです。

表9.5 cinder

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。

設定オプション = デフォルト値	型	説明
catalog_info = volumev3::publicURL	文字列値	サービスカタログで cinder を検索する際に一致する情報 **<service_name>** はオプションで、ほとんどのデプロイメントでは不要であるため、デフォルトで省略されます。 値: ● 形式は <service_type>:<service_name>:<endpoint_type> の形式で区切られた値です。 注意: Nova 17.0.0 Queens リリース以降、nova は Cinder v2 API をサポートしません。 関連するオプション: ● endpoint_template - このオプションを設定すると、catalog_info が上書きされます。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
cross_az_attach = True	ブール値	異なるアベイラビリティーゾーンのインスタンスとボリューム間の接続を許可します。 False の場合、インスタンスに接続されているボリュームは、Cinder 内のインスタンスアベイラビリティーゾーンと同じアベイラビリティーゾーンにある必要があります。 また、これは、ソースがボリュームではないボリュームからインスタンスを起動する際には、注意すべきです。Nova は、インスタンスに割り当てられたものと同じアベイラビリティーゾーンを使用してボリュームの作成を試みるためです。 その AZ が Cinder にない (または cinder.conf で **allow_availability_zone_fallback=False**) 場合、ボリューム作成リクエストは失敗し、インスタンスはビルドリクエストに失敗します。 デフォルトでは、ボリュームの割り当てにはアベイラビリティーゾーンの制限はありません。 関連するオプション: ● **[DEFAULT]/default_schedule_zone**

設定オプション = デフォルト値	型	説明
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint_template = None	文字列値	このオプションを設定すると、cinder エンドポイントについてのこのテンプレートでサービスカタログの検索が上書きされます。 値: ● cinder エンドポイント API の URL (例: http://localhost:8776/v3/%(project_id)s) 注意: Nova 17.0.0 Queens リリース以降、nova は Cinder v2 API をサポートしません。 関連するオプション: ● catalog_info: endpoint_template が設定されていない場合、catalog_info が使用されます。
http_retries = 3	整数値	失敗した http 呼び出しで cinderclient が再試行される回数。0 は、接続が 1 回のみ試行されることを意味します。正の整数に設定すると、接続に失敗すると何度も再試行されます。たとえば、これを 3 に設定すると、接続合計の試みは 4 になります。 値: ● 整数値。0 は、接続が 1 回のみ試行されることを意味します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル

設定オプション = デフォルト値	型	説明
os_region_name = None	文字列値	このノードのリージョン名。これは、サービスカタログで URL を選択する場合に使用されます。 値: ● リージョン名を表す文字列
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	User ID
username = None	文字列値	ユーザー名

9.1.7. compute

以下の表は、**/etc/nova/nova.conf** ファイルの **[compute]** グループで使用できるオプションの概要をまとめたものです。

表9.6 compute

設定オプション = デフォルト値	型	説明
consecutive_build_service_disable_threshold = 10	整数値	スケジューラーへのビルド失敗の報告を有効にします。 ゼロ以外の値を使用すると、BuildFailureWeigher が使用するスケジューラーにビルド失敗統計を送信できるようになります。 値: ● 任意の正の整数により、ビルドの失敗を報告できます。 ● ゼロの場合、ビルドの失敗の報告を無効にします。 関連するオプション: ● [filter_scheduler]/build_failure_weight_multiplier

設定オプション = デフォルト値	型	説明
cpu_dedicated_set = None	文字列値	**PCPU** リソースに使用できるホスト CPU のマスク。 このオプションの動作は、非推奨の **vcpu_pin_set** オプションの動作に影響します。 ● このオプションが定義されている場合、**vcpu_pin_set** を定義するとエラーが発生します。 ● このオプションが定義されていない場合には、**vcpu_pin_set** を使用して **VCPU** リソースのインベントリーを決定し、ピニングされたインスタンスとピニングされていないインスタンスの両方をスケジューリングできるホスト CPU を制限します。 この動作は、**vcpu_pin_set** が削除されると、今後のリリースで単純化されます。 値: ● インスタンスの仮想 CPU の割り当て元となる物理 CPU 番号のコンマ区切りリスト。各要素は、単一の CPU 番号、CPU 番号の範囲、またはその順で、次に除外される CPU 番号のいずれかでなければなりません。 **以前の範囲。以下に例を示します。** cpu_dedicated_set = "4-12,^8,15" 関連するオプション: ● **[compute] cpu_shared_set**: これは、**VCPU** リソースの割り当て元を定義するための対応するオプションです。 ● **vcpu_pin_set**: このオプションが部分的に置き換える従来のオプション。

設定オプション = デフォルト値	型	説明
cpu_shared_set = None	文字列値	**VCPU** リソースに使用可能なホスト CPU をマスクし、エミュレータースレッドをオフロードします。 このオプションの動作は、非推奨の **vcpu_pin_set** オプションの定義によって異なります。 ● **vcpu_pin_set** が定義されていない場合、**[compute] cpu_shared_set** を使用して **VCPU** インベントリーを提供し、ピニングされていないインスタンスをスケジューリングできるホスト CPU を決定します。また、**share**エミュレータースレッドポリシー (**hw:emulator_threads_policy = share**) で設定されたインスタンスのインスタンスエミュレータースレッドをオフロードする必要があるホスト CPU を決定するためにも使用されます。 ● **vcpu_pin_set** が定義されている場合、**[compute] cpu_shared_set** は、**share** エミュレータースレッドポリシー (**hw:emulator_threads_policy=share**) で設定されたインスタンスのインスタンスエミュレータースレッドがオフロードされるホスト CPU を決定するためにのみ使用されます。**vcpu_pin_set** は、**VCPU** インベントリーを提供し、固定されたインスタンスと固定されていないインスタンスの両方をスケジュールできるホスト CPU を決定するために使用されます。 この動作は、**vcpu_pin_set** が削除されると、今後のリリースで単純化されます。 値: ● インスタンスの仮想 CPU の割り当て元となる物理 CPU 番号のコンマ区切りリスト。各要素は、単一の CPU 番号、CPU 番号の範囲、またはその順で、次に除外される CPU 番号のいずれかでなければなりません。 **以前の範囲。以下に例を示します。** cpu_shared_set = "4-12,^8,15" 関連するオプション: ● **[compute] cpu_dedicated_set**: これは、**PCPU** リソースの割り当て元を定義するための対応するオプションです。 ● **vcpu_pin_set**: このオプションの動作を変更する定義があるレガシーオプション。

設定オプション = デフォルト値	型	説明
image_type_exclude_list = []	リスト値	このコンピュートノードでは対応していないイメージ形式のリスト。 状況によっては、コンピュートノードが高価なイメージフォーマットまたは複雑なイメージ形式のサポートを拒否することが望ましい場合があります。この要因は、特定のイメージを使用して起動時に選択するコンピュートノードをスケジューラーが決定します。 値: ● glance イメージの **disk_format** 名 (**raw**、**qcow2** など) 関連するオプション: ● **[scheduler]query_placement_for_image_type_support**: サポートされているイメージタイプに基づいてコンピュートをフィルタリングできるようにします。
live_migration_wait_for_vif_plug = True	ブール値	移行元コンピュートホストが (neutron) ネットワークサービスからの **network-vif-plugged** イベントを待つ必要があるかどうかを判別してから、ゲストから移行先コンピュートホストへの実際の転送を開始します。 このオプションは、ライブマイグレーションの移行先ホストで読み取られることに注意してください。このオプションをすべてのコンピュートホストで同じ設定した場合、同じネットワークバックエンドをユニバーサルに使用する場合には、これについて心配する必要はありません。 ゲストの移動を開始する前に、仮想インターフェイスのプラグインなど、移行先のコンピュートホストで一部の設定が行われます。**宛先ホスト** のネットワークバックエンドによっては、**network-vif-plugged** イベントがトリガーされ、移行元コンピュートホストで受信される可能性があり、移行元コンピュートは、ハイパーバイザーでゲストの移行を開始する前に、そのイベントでネットワークが移行先ホストで設定されるのを待つことができます。 a. 注記: > The compute service cannot reliably determine which types of virtual interfaces (`port.binding:vif_type`) will send `network-vif-plugged` events without an accompanying port

設定オプション = デフォルト値	型	説明
		`binding:host_id` change. Open vSwitch and linuxbridge should be OK, but OpenDaylight is at least one known backend that will not currently work in this case, see bug https://launchpad.net/bugs/1755890 for more details. 値: • true: **network-vif-plugged** イベントを待機してからゲストの転送を開始 • false: ゲスト転送を開始する前に、**network-vif-plugged** イベントを待機しません (これはレガシー動作です)。 関連するオプション: • [DEFAULT]/vif_plugging_is_fatal: **live_migration_wait_for_vif_plug** が True で、**vif_plugging_timeout**が 0 より大きい場合、タイムアウトに達すると、ライブ移行プロセスはエラーで失敗しますが、ゲスト転送は宛先ホストに開始されません • [DEFAULT]/vif_plugging_timeout: **live_migration_wait_for_vif_plug**が True の場合、これはタイムアウトするまで待機する時間を制御し、**vif_plugging_is_fatal** が True の場合は失敗するか、単にライブ移行を続行します
max_concurrent_disk_ops = 0	整数値	並行して実行する同時ディスク IO 集約操作の数 (glance イメージのダウンロード、イメージフォーマットの変換など)。これが高すぎると、応答時間に影響を与えます。デフォルト値の 0 は無制限を意味します。
max_disk_devices_to_attach = -1	整数値	1 つのサーバーに接続できる最大ディスクデバイス数。サーバーがサポートするディスク数は、使用するバスによって異なることに注意してください。たとえば、**ide** ディスクバスは、アタッチされている 4 つのデバイスに制限されます。設定された最大値は、サーバーの作成、再構築、退避、取り消し、ライブマイグレーション、および接続時に実施されます。 通常、ディスクバスは、デバイスタイプまたはディスクデバイス、および仮想化タイプから自動的に決定されます。ただし、ディスクバスは、ブロックデバイスのマッピングまたはイメージ属性で指定することもできます。ブロックデバイスマッピングでのディスクバスの指定の詳細については、:doc: **/user/block-device-mapping** の **disk_bus** フィールドを参照してください。ま

設定オプション = デフォルト値	型	説明
		た、https://docs.openstack.org/glance/latest/admin/useful-image を参照してください。-properties.html で、**hw_disk_bus** イメージプロパティーの詳細を確認してください。 オペレーターは、サーバーをホストする Compute サービスの **[compute]/max_disk_devices_to_attach** を変更すると、サーバーにすでにアタッチされているデバイスの数より小さいと、リビルドが失敗する可能性があることを認識する必要があります。たとえば、サーバー A に 26 のデバイスがアタッチされていて、オペレーターが **[compute]/max_disk_devices_to_attach** を 20 に変更すると、サーバー A の再構築要求が失敗し、26 デバイスがすでに割り当てられ、新たに設定された最大 20 を超えているため、ERROR 状態になります。 **[compute]/max_disk_devices_to_attach** を設定する Operator は、コールドマイグレーション時にも、設定された最大値がインプレースのみ適用され、移動前に宛先がチェックされないことに注意してください。これは、Operator がコンピュートホスト A に最大 26 個、コンピュートホスト B に最大 20 個を設定した場合、26 個のデバイスがアタッチされたサーバーのコンピュートホスト A からコンピュートホスト B へのコールドマイグレーションが成功することを意味します。次に、サーバーがコンピュートホスト B に置かれると、サーバーを再構築するリクエストが失敗し、26 デバイスがすでにアタッチされ、コンピュートホスト B の設定された最大 20 を超えているため、サーバーを再構築して ERROR 状態になります。 設定された最大値は、コンピュートホストがないため、退避オフロードサーバーには強制されません。 a. 警告:: このオプションを 0 に設定すると、**nova-compute** サービスは起動に失敗してしまいます。これは、0 ディスクデバイスは無効な設定で、インスタンスの起動を防ぎます。 値: ● -1 は無制限を意味します。 ● 整数 >= 1 は最大許容値を表します。0 を値として指定すると、**nova-compute** サービスが起動に失敗します。これは、0 ディスクデバイスが無効な設定で、インスタンスが起動できなくなるためです。

設定オプション = デフォルト値	型	説明
provider_config_location = /etc/nova/provider_config/	文字列値	リソースプロバイダーの設定データを含む YAML ファイルの場所。 これらのファイルを使用すると、オペレーターは追加のカスタムインベントリーと特性を指定して、1 つ以上のリソースプロバイダーに割り当てることができます。 追加のドキュメントは次の場所から入手できます。 > https://docs.openstack.org/nova/latest/admin/managing-resource-providers.html
resource_provider_association_refresh = 300	整数値	compute ノードリソースプロバイダーのインベントリー、集約、および特性の nova-compute 側のキャッシュを更新する間隔。 このオプションは、コンピュートノードのローカルキャッシュでプロバイダーのインベントリー、集約、および特性の更新を試みる間隔の秒数を指定します。 値をゼロにすると、キャッシュの更新を完全に無効にします。 SIGHUP をコンピュートプロセスに送信して次回データにアクセスしたときに再度入力することで、いつでも手動でキャッシュを消去できます。 値: - 正の整数 (秒単位)、または更新を無効にするにはゼロ。
shutdown_retry_interval = 10	整数値	ACPI シャットダウンシグナルをインスタンスに再送信するまでの待機時間 (秒単位)。 全体の待機時間は **shutdown_timeout** によって設定されます。 値: - 0 より大きい整数 (秒単位) 関連するオプション: - **shutdown_timeout**

設定オプション = デフォルト値	型	説明
vmdk_allowed_types = ['streamOptimized', 'monolithicSparse']	リスト値	許可される VMDK の create-type サブフォーマットを説明する文字列のリスト。名前付きエクステントの処理によるホストファイルの潜在的な露出を避けるために、スパースヘッダーを持つ単一ファイルのバリアントのみを含めることを推奨します。このリストが空の場合、VMDK イメージの形式は許可されません。

9.1.8. conductor

以下の表は、**/etc/nova/nova.conf** ファイルの **[conductor]** グループで使用できるオプションの概要をまとめたものです。

表9.7 conductor

設定オプション = デフォルト値	型	説明
workers = None	整数値	OpenStack Conductor サービスのワーカー数。デフォルトは、利用可能な CPU の数になります。

9.1.9. console

以下の表は、**/etc/nova/nova.conf** ファイルの **[console]** グループで使用できるオプションの概要をまとめたものです。

表9.8 console

設定オプション = デフォルト値	型	説明
allowed_origins = []	リスト値	コンソールの Websocket プロキシーに許可されるオリジンのリストを追加して、他の元のホスト名からの接続を許可します。WebSocket プロキシーは、サイト間の要求を防ぐために、ホストヘッダーとオリジンヘッダーと一致します。このリストは、オリジンのヘッダーで host 以外の値が許可されるかどうかを指定します。 値: ● 各要素が許可される元のホスト名であるリスト。それ以外の場合は空のリスト

設定オプション = デフォルト値	型	説明
ssl_ciphers = None	文字列値	クライアントからの TLS 接続に許可する暗号を指定する OpenSSL 暗号設定文字列。以下に例を示します。 ssl_ciphers = "kEECDH+aECDSA+AES:kEECDH+AES+aRSA:kEDH+aRSA+AES" 暗号設定の文字列形式および許可される値の詳細は、OpenSSL **ciphers** コマンドの man ページを参照してください。 https://www.openssl.org/docs/man1.1.0/man1/ciphers.html 関連するオプション: ● [DEFAULT] cert ● [DEFAULT] key
ssl_minimum_version = default	文字列値	許可される SSL/TLS プロトコルの最低バージョン。 関連するオプション: ● [DEFAULT] cert ● [DEFAULT] key

9.1.10. consoleauth

以下の表は、**/etc/nova/nova.conf** ファイルの **[consoleauth]** グループで使用できるオプションの概要をまとめたものです。

表9.9 consoleauth

設定オプション = デフォルト値	型	説明
token_ttl = 600	整数値	コンソール認証トークンの有効期間 (秒単位)。 コンソールの認証トークンは、ユーザーのコンソールアクセスの承認に使用されます。認証トークンの生存期間が経過すると、トークンは期限切れとみなされます。その後、期限切れのトークンが削除されます。

9.1.11. cors

以下の表は、**/etc/nova/nova.conf** ファイルの **[cors]** グループで使用できるオプションの概要をまとめたものです。

表9.10 cors

設定オプション = デフォルト値	型	説明
allow_credentials = True	ブール値	実際の要求にユーザーの認証情報を含めることができることを示します。
allow_headers = ['X-Auth-Token', 'X-Openstack-Request-Id', 'X-Identity-Status', 'X-Roles', 'X-Service-Catalog', 'X-User-Id', 'X-Tenant-Id', 'X-OpenStack-Nova-API-Version', 'OpenStack-API-Version']	リスト値	実際の要求時に使用されるヘッダーフィールド名を示します。
allow_methods = ['GET'、 'PUT'、 'POST'、 'DELETE'、 'PATCH']	リスト値	実際の要求時に使用できるメソッドを示します。
allowed_origin = None	リスト値	このリソースがリクエストの origin"ヘッダーで受信したドメインと共有されるかどうかを示します。形式:"<protocol>://<host>[:<port>]" (行末のスラッシュなし)例:https://horizon.example.com
expose_headers = ['X-Auth-Token', 'X-Openstack-Request-Id', 'X-Subject-Token', 'X-Service-Token', 'X-OpenStack-Nova-API-Version', 'OpenStack-API-Version']	リスト値	API に安全に公開できるヘッダーを示します。デフォルトは HTTP Simple ヘッダーです。
max_age = 3600	整数値	CORS プリフライトリクエストの最大キャッシュ期間。

9.1.12. cyborg

次の表は、**/etc/nova/nova.conf** ファイルの **[cyborg]** グループで使用できるオプションの概要を示しています。

表9.11 cyborg

設定オプション = デフォルト値	型	説明
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = accelerator	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
timeout = None	整数値	http 要求のタイムアウト値

設定オプション = デフォルト値	型	説明
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。

9.1.13. database

以下の表は、**/etc/nova/nova.conf** ファイルの **[database]** グループで使用できるオプションの概要をまとめたものです。

表9.12 database

設定オプション = デフォルト値	型	説明
backend = sqlalchemy	文字列値	データベースに使用するバックエンド。
connection = None	文字列値	データベースへの接続に使用する SQLAlchemy 接続文字列。
connection_debug = 0	整数値	SQL デバッグ情報の冗長性: 0=なし、100=すべて。
`connection_parameters = `	文字列値	接続時に接続 URL に追加するオプションの URL パラメーター。param1=value1&param2=value2&... として指定します。
connection_recycle_time = 3600	整数値	この秒数より長く接続プールに存在していた接続は、次にプールからチェックアウトされたときに新しい接続に置き換えられます。
connection_trace = False	ブール値	Python スタックトレースをコメント文字列として SQL に追加します。
db_inc_retry_interval = True	ブール値	True の場合には、db_max_retry_interval までのデータベース操作の再試行回数を長くします。
db_max_retries = 20	整数値	エラーが発生する前に接続エラーまたはデッドロックが発生した場合の最大再試行。再試行回数を無限に指定するには -1 に設定します。
db_max_retry_interval = 10	整数値	db_inc_retry_interval が設定されている場合は、データベース操作を再試行するまでの最大秒数です。
db_retry_interval = 1	整数値	データベーストランザクションの再試行間の秒数。

設定オプション = デフォルト値	型	説明
max_overflow = 50	整数値	設定されている場合、この値を SQLAlchemy で max_overflow に使用します。
max_pool_size = 5	整数値	プールに開いたままにする SQL 接続の最大数。値を 0 に設定すると無制限を意味します。
max_retries = 10	整数値	起動時のデータベース接続の最大再試行数。再試行回数を無限に指定するには -1 に設定します。
mysql_enable_ndb = False	ブール値	True の場合には、MySQL Cluster (NDB) の処理のサポートを透過的に有効にします。
mysql_sql_mode = TRADITIONAL	文字列値	MySQL セッションに使用する SQL モード。このオプション (デフォルトを含む) は、サーバーセット SQL モードを上書きします。サーバー設定で設定された SQL モードを使用するには、これを no value に設定します。例: mysql_sql_mode=
pool_timeout = None	整数値	設定されている場合は、この値を SQLAlchemy で pool_timeout に使用します。
retry_interval = 10	整数値	SQL 接続を開く再試行の間隔。
slave_connection = None	文字列値	スレーブデータベースへの接続に使用する SQLAlchemy 接続文字列。
sqlite_synchronous = True	ブール値	True の場合、SQLite は同期モードを使用します。
use_db_reconnect = False	ブール値	失われた接続でデータベースの実験的な使用を有効にします。
use_tpool = False	ブール値	すべての DB API 呼び出しのスレッドプールの実験的な使用を有効にします。

9.1.14. devices

以下の表は、**/etc/nova/nova.conf** ファイルの **[devices]** グループで使用できるオプションの概要をまとめたものです。

表9.13 devices

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
enabled_vgpu_types = []	リスト値	コンピュートノードで有効な仮想 GPU の種別。 一部の pGPU (例: NVIDIA GRID K1) は、さまざまな vGPU タイプをサポートしています。このオプションを使用して、ゲストインスタンスに割り当てることのできる有効な仮想 GPU 種別のリストを指定できます。 複数の単一の vGPU タイプが指定されている場合、**vGPU タイプ** ごとに追加のセクション **[vgpu_$(VGPU_TYPE)]** を設定ファイルに追加する必要があります。各セクションは、単一の設定オプション **device_addresses** で設定する **必要** があります。これは、このタイプに割り当てる物理 GPU に対応する PCI アドレスのリストである必要があります。 1 つ以上のセクションが欠落している場合 (少なくとも 1 つの物理 GPU に特定のタイプを使用したくない場合)、またはデバイスアドレスが提供されていない場合、Nova は **[devices]/enabled_vgpu_types** によって提供された最初のタイプのみを使用します。 2 つの異なるタイプに同じ PCI アドレスが指定されている場合、nova-compute は再起動時に InvalidLibvirtGPUConfig 例外を返します。 **例は次のとおりです。** [devices] enabled_vgpu_types = nvidia-35, nvidia-36 [vgpu_nvidia-35] device_addresses = 0000:84:00.0,0000:85:00.0 [vgpu_nvidia-36] device_addresses = 0000:86:00.0

9.1.15. ephemeral_storage_encryption

以下の表は、**/etc/nova/nova.conf** ファイルの **[ephemeral_storage_encryption]** グループで使用できるオプションの概要をまとめたものです。

表9.14 ephemeral_storage_encryption

設定オプション = デフォルト値	型	説明
cipher = aes-xts-plain64	文字列値	使用される cipher-mode 文字列。 一時ストレージの暗号化に使用する暗号およびモード。利用可能な暗号モードの組み合わせは、カーネルのサポートによって異なります。dm-crypt のドキュメントによると、暗号は "<cipher>-<chainmode>-<ivmode>" 形式になることが予想されます。 値: ● **/proc/crypto** に記載されている crypto オプション。
enabled = False	ブール値	LVM 一時ストレージの暗号化を有効/無効にします。
key_size = 512	整数値	暗号化キーの長さ (ビット単位)。 一時ストレージの暗号化に使用する暗号化キーのビット長。XTS モードでは、ビットの半分のみが暗号化キーに使用されます。

9.1.16. filter_scheduler

以下の表は、**/etc/nova/nova.conf** ファイルの **[filter_scheduler]** グループで使用できるオプションの概要をまとめたものです。

表9.15 filter_scheduler

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
aggregate_image_properties_isolation_namespace = None	文字列値	ホストアグリゲートで使用するイメージ属性の名前空間。 イメージおよびホストは、特定のアグリゲートのホストのみに特定のイメージをスケジュールできるように設定できます。これは、このオプションの最初に特定されるホストアグリゲートで設定されるメタデータの値で実行されます。ホストがこのようなメタデータキーを持つアグリゲートの一部である場合、スケジューラーがホストを許容可能なとみなすために、要求仕様のイメージにそのメタデータの値をプロパティーに指定する必要があります。 この設定は、**AggregateImagePropertiesIsolation** フィルターが有効になっている場合にのみスケジュールに影響することに注意してください。 値: ● 文字列。文字列が image プロパティーの名前空間に対応します。 関連するオプション: ● **[filter_scheduler] aggregate_image_properties_isolation_separator**

設定オプション = デフォルト値	型	説明
aggregate_image_properties_isolation_separator = .	文字列値	イメージプロパティーの名前空間および名前に対する区切り文字。 aggregate_image_properties_isolation フィルターを使用する場合には、関連するメタデータキーの前に aggregate_image_properties_isolation_namespace 設定オプションで定義した名前空間にセパレーターが追加されます。このオプションは、使用する区切り文字を定義します。 この設定は、**AggregateImagePropertiesIsolation** フィルターが有効になっている場合にのみスケジュールに影響することに注意してください。 値: ● 文字列。文字列がイメージプロパティーの名前空間セパレーターに対応する 関連するオプション: ● **[filter_scheduler] aggregate_image_properties_isolation_namespace**
available_filters = ['nova.scheduler.filters.all_filters']	多値	スケジューラーが使用できるフィルター。 nova スケジューラーが適用するフィルタークラスの順不同リスト。**[filter_scheduler] enabled_filters** オプションで指定されたフィルターのみが使用されますが、そのオプションに表示されるすべてのフィルターもこのリストに含まれている必要があります。 デフォルトでは、nova に含まれる全フィルターに設定されます。 値: ● ゼロ以上の文字列のリスト。各文字列はホストの選択に使用できるフィルターの名前に対応します。 関連するオプション: ● **[filter_scheduler] enabled_filters**

設定オプション = デフォルト値	型	説明
build_failure_weight_multiplier = 1000000.0	浮動小数点の値	直近のビルドに失敗したホストを重み付けするのに使用する乗数。 このオプションは、最新のビルドに失敗して、コンピュートノードに配置される重みを決定します。ビルドの失敗は、失敗、間違った設定、またはコンピュートノードの実行などのことを示し、スケジューリング中にこれを回避するのが有益です。重みは、コンピュートノードが経験した直近のビルド失敗数とは逆に比例します。この値は、利用可能なリソースが原因で、他の有効化された重み付け関数で指定されるオフセットの重みに高い値に設定する必要があります。直近の失敗回数でコンピュートホストを重み付けするには、これをゼロに設定します。 **BuildFailureWeigher** 重み付け関数が有効になっている場合、この設定はスケジューリングにのみ影響することに注意してください。 値: ● 整数または浮動小数点の値。ここで、値はこの重み付け関数の乗数比率に対応します。 関連するオプション: ● **[compute] consecutive_build_service_disable_threshold** - この重み付けによって考慮されるデータを報告するための計算では、ゼロ以外である必要があります。 ● **[filter_scheduler] weight_classes**

設定オプション = デフォルト値	型	説明
cpu_weight_multiplier = 1.0	浮動小数点の値	CPU 重みの乗数比率。 空き仮想 CPU の重み付けに使用する乗数。負の数字は、分散ではなくスタックを示します。 この設定は、**CPUWeigher** 重み付けが有効になっている場合にのみスケジューリングに影響することに注意してください。 値: ● 整数または浮動小数点の値。ここで、値はこの重み付け関数の乗数比率に対応します。 関連するオプション: ● **[filter_scheduler] weight_classes**

設定オプション = デフォルト値	型	説明
cross_cell_move_weight_multiplier = 1000000.0	浮動小数点の値	セル間の移動中にホストの重み付けに使用される乗数。 このオプションは、セル間のサイズ変更などでサーバーを移動するときに、同じソースセル内にあるホストに適用される重みを決定します。デフォルトでは、インスタンスを移動する場合、スケジューラーは同じセル内のホストを優先します。これは、クロスセル移行の複雑な性質により、クロスセル移動操作が遅くなり、リスクが高くなる可能性があるためです。 この設定は、**CrossCellWeigher** 重み付け関数が有効になっている場合にのみスケジューリングに影響することに注意してください。クロスセル移行をサポートするようにクラウドが設定されていない場合、このオプションは効果がありません。 この設定オプションの値は、同じ名前 (**cross_cell_move_weight_multiplier**) を持つ集約メタデータキーを設定することにより、ホスト集約ごとにオーバーライドできます。 値: ● 整数または浮動小数点の値。ここで、値はこの重み付け関数の乗数比率に対応します。正の値は、インスタンスが現在実行されているのと同じセル内のホストを重み付けが優先することを意味します。負の値は、インスタンスが現在実行されている **他** のセル内のホストを重み付けが優先することを意味します。 関連するオプション: ● **[filter_scheduler] weight_classes**

設定オプション = デフォルト値	型	説明
disk_weight_multiplier = 1.0	浮動小数点の値	ディスク重量倍数比。 空きディスク容量を重み付けする際に使用する乗数。負の数字は分散ではなくスタックを意味します。 この設定は、**DiskWeigher** 重み付け関数が有効になっている場合にのみスケジューリングに影響することに注意してください。 値: ● 整数または浮動小数点の値。ここで、値はこの重み付け関数の乗数比率に対応します。
enabled_filters = ['AvailabilityZoneFilter', 'ComputeFilter', 'ComputeCapabilitiesFilter', 'ImagePropertiesFilter', 'ServerGroupAntiAffinityFilter', 'ServerGroupAffinityFilter']	リスト値	スケジューラーが使用するフィルター。 ホストのフィルタリングに使用されるフィルタークラス名の順序付きリスト。これらのフィルターはリスト順に適用されるため、最も制限の厳しいフィルターを最初に配置して、フィルタリングプロセスを効率化します。 このオプションのすべてのフィルターは、**[scheduler_filter] available_filter** オプションに存在する **必要** があります。存在しない場合、**SchedulerHostFilterNotFound** 例外が発生します。 値: ● ゼロ以上の文字列のリスト。各文字列はホストの選択に使用するフィルターの名前に対応します。 関連するオプション: ● **[filter_scheduler] available_filters**

設定オプション = デフォルト値	型	説明
host_subset_size = 1	整数値	スケジューラーが選択する最適なホストのサブセットのサイズ。 新規インスタンスは、N best hosts のサブセットからランダムに選択されたホストでスケジュールされます。N はこのオプションで設定された値です。 これを 1 より大きい値に設定すると、同様の要求を処理する複数のスケジューラープロセスで同じホストが選択され、潜在的な競合状態が発生する可能性があります。要求に最も適した N 台のホストからホストを無作為に選択することで、競合の可能性が低減されます。ただし、この値を高く設定すると、選択されるホストが特定の要求に対して最適ではない可能性が高くなります。 値: ● 整数。整数はホストサブセットのサイズに対応します。

設定オプション = デフォルト値	型	説明
hypervisor_version_weight_multiplier = 1.0	浮動小数点の値	ハイパーバイザーのバージョンの重み乗数の比。 乗数は、報告されたハイパーバイザーのバージョンに基づきホストを重み付けするために使用されます。負の数値は古いホストから優先されることを示します。デフォルトでは、アップグレードを支援するために新しいホストが優先されます。 値: ● 整数または浮動小数点の値。ここで、値はこの重み付け関数の乗数比率に対応します。 以下に例を示します。 ● 古いホストを強く優先する a. code-block:: ini [filter_scheduler] hypervisor_version_weight_multiplier=-1000 ● 新しいホストを適度に優先する a. code-block:: ini [filter_scheduler] hypervisor_version_weight_multiplier=2.5 ● 重み付けの影響を無効にする a. code-block:: ini [filter_scheduler] hypervisor_version_weight_multiplier=0 関連するオプション: ● **[filter_scheduler] weight_classes**

設定オプション = デフォルト値	型	説明
image_properties_default_architecture = None	文字列値	イメージプロパティーフィルターの使用時に使用されるデフォルトのアーキテクチャー。 **ImagePropertiesFilter** を使用する場合、デフォルトのアーキテクチャーを定義して、ユーザーエクスペリエンスをより簡単にし、ユーザーが Glance で **hw_architecture** プロパティーを指定しなかったために x86_64 イメージのようなものが AARCH64 コンピュートノードにランディングするのを避けることができます。 値: ● x86_64、aarch64、s390x などの CPU アーキテクチャー。
io_ops_weight_multiplier = -1.0	浮動小数点の値	IO 操作の重みの倍率。 このオプションは、ワークロードが異なるホストが重み付けされる方法を決定します。デフォルトなど、負の値を使用すると、スケジューラーはワークロードが若干のホストを優先することになりますが、正の値を指定すると、負荷が重いホストが優先されます。また、このオプションの正の値では、すでにビジー状態のホストにインスタンスをスケジュールする傾向があり、負の値はより多くのホストに分散する傾向があります。正または負の絶対値で、他の重み付け関数に対して io_ops の重み付け関数をどれだけ優先するかを指定します。 この設定は、**IoOpsWeigher** 重み付け関数が有効になっている場合にのみスケジューリングに影響することに注意してください。 値: ● 整数または浮動小数点の値。ここで、値はこの重み付け関数の乗数比率に対応します。 関連するオプション: ● **[filter_scheduler] weight_classes**

設定オプション = デフォルト値	型	説明
isolated_hosts = []	リスト値	特定のイメージのみを実行できるホストのリスト。 特定のホストでのみイメージを実行するように制限する必要がある場合は、これらのホスト名をリスト表示します。 この設定は、**IsolatedHostsFilter** フィルターが有効になっている場合にのみスケジュールに影響することに注意してください。 値: ● 文字列のリスト。各文字列はホストの名前に対応します。 関連するオプション: ● **[filter_scheduler] isolated_images** ● **[filter_scheduler] restrict_isolated_hosts_to_isolated_images**
isolated_images = []	リスト値	特定のホストでしか実行できないイメージの UUID のリスト。 特定のホストでのみイメージを実行するように制限する必要がある場合は、イメージの UUID をリスト表示します。 この設定は、**IsolatedHostsFilter** フィルターが有効になっている場合にのみスケジュールに影響することに注意してください。 値: ● UUID 文字列のリスト。各文字列はイメージの UUID に対応します。 関連するオプション: ● **[filter_scheduler] isolated_hosts** ● **[filter_scheduler] restrict_isolated_hosts_to_isolated_images**

設定オプション = デフォルト値	型	説明
max_instances_per_host = 50	整数値	ホストに存在できる最大インスタンス数 指定ホスト上のインスタンスの数を制限する必要がある場合は、このオプションで許可する必要のある最大インスタンス数に設定します。NumInstancesFilter および AggregateNumInstancesFilter は、このオプションの値と同じ以上のインスタンスを持つホストをすべて拒否します。 この設定は、**NumInstancesFilter** または **AggregateNumInstancesFilter** フィルターが有効になっている場合にのみスケジュールに影響することに注意してください。 値: ● 整数。この整数は、ホストにスケジュール可能な最大インスタンスに対応します。 関連するオプション: ● **[filter_scheduler] enabled_filters**
max_io_ops_per_host = 8	整数値	ホストで IO をアクティブに実行できるインスタンスの数。 IO を実行するインスタンスには、build、resize、snapshot、migrate、rescue、unshve などの状態が含まれます。 この設定は、**IoOpsFilter** フィルターが有効になっている場合にのみスケジューリングに影響することに注意してください。 値: ● 整数。この整数は、任意のホストで IO をアクティブに実行できるインスタンスの最大数に対応します。 関連するオプション: ● **[filter_scheduler] enabled_filters**

設定オプション = デフォルト値	型	説明
pci_weight_multiplier = 1.0	浮動小数点の値	PCI デバイスアフィニティーの重み乗数。 PCI デバイスアフィニティーの重み付け関数上の PCI デバイスの数と、インスタンスが要求する PCI デバイスの数に基づいて重み付けを計算します。 この設定は、**PCIWeigher** 重み付けと **NUMATopologyFilter** フィルターが有効になっている場合にのみスケジューリングに影響することに注意してください。 値: ● 正の整数または浮動小数点値。ここで、値はこの重み付け関数の乗数比率に対応します。 関連するオプション: ● **[filter_scheduler] weight_classes**
ram_weight_multiplier = 1.0	浮動小数点の値	RAM 重量倍数比。 このオプションは、利用可能な RAM が多いか、少ないホストを重み付けする方法を決定します。正の値を指定すると、スケジューラーは RAM がより多くなるホストを優先し、スケジューラーが利用可能な RAM が少ないホストを優先します。別の見方をすれば、このオプションの正の値はインスタンスを多くのホストに分散させる傾向があり、負の値は使用頻度の低いホストにスケジュールする前にホストを可能な限りいっぱいにする (スタックする) 傾向があります。正または負の絶対値で、他の重み付け関数と比べて{b> <b}RAM の重み付け関数をどれだけ優先するかを指定します。 この設定は、**RAMWeigher** 重み付け関数が有効になっている場合にのみスケジューリングに影響することに注意してください。 値: ● 整数または浮動小数点の値。ここで、値はこの重み付け関数の乗数比率に対応します。 関連するオプション: ● **[filter_scheduler] weight_classes**

設定オプション = デフォルト値	型	説明
restrict_isolated_hosts_to_isolated_images = True	ブール値	分離していないイメージが分離したホストにビルドされないようにします。 この設定は、**IsolatedHostsFilter** フィルターが有効になっている場合にのみスケジュールに影響することに注意してください。それでも、このオプションは分離されたイメージのリクエストの動作には影響しません。分離されたイメージは**常に**分離されたホストに制限されます。 関連するオプション: ● **[filter_scheduler] isolated_images** ● **[filter_scheduler] isolated_hosts**
shuffle_best_same_weighed_hosts = False	ブール値	同じ最善の重みを持つホスト間でインスタンスを分散できるようにします。 これを有効にすると、**[filter_scheduler] host_subset_size** が 1 (デフォルト) であるが、同じ最大重みを持つ多数のホストが存在する場合に役立ちます。このシナリオは、通常、スケジューラーに同一の重みが返されるベアメタルノードが多数ある Ironic デプロイメントでよく見られます。このような場合、このオプションを有効にすると競合が減少し、イベントの再スケジュールの可能性が低くなります。同時に、インスタンスのパッキング (重み付けされていない場合でも) の密度が低くなります。
soft_affinity_weight_multiplier = 1.0	浮動小数点の値	グループのソフトアフィニティーのホストを重み付けするために使用される乗数。 この設定は、**ServerGroupSoftAffinityWeigher** 重み付けが有効になっている場合にのみスケジューリングに影響することに注意してください。 値: ● 負の整数または浮動小数点値。この値は、グループのソフトアフィニティーを持つホストの重みの乗数に対応します。 関連するオプション: ● **[filter_scheduler] weight_classes**

設定オプション = デフォルト値	型	説明
soft_anti_affinity_weight_multiplier = 1.0	浮動小数点の値	グループのソフト非アフィニティーのホストを重み付けするために使用される乗数。 この設定は、**ServerGroupSoftAntiAffinityWeigher** 重み付けが有効になっている場合にのみスケジューリングに影響することに注意してください。 値: ● 負の整数または浮動小数点値。この値は、グループのソフト非アフィニティーを持つホストの重みの乗数に対応します。 関連するオプション: ● **[filter_scheduler] weight_classes**

設定オプション = デフォルト値	型	説明
track_instance_changes = True	ブール値	インスタンス情報に対する個別ホストのクエリーを有効にします。 スケジューラーは、フィルターと重み付け関数を評価するために、ホスト上のインスタンスに関する情報が必要になる場合があります。この情報の最も一般的な必要性は、(アンチ) アフィニティーフィルターであり、ホスト上ですでに実行されているインスタンスに基づいてホストを選択する必要があります。 設定したフィルターと重み付け関数がこの情報を必要としない場合、このオプションを無効にするとパフォーマンスが向上します。また、追跡オーバーヘッドが大きすぎると無効にすることもできます。ただし、これにより、ホストの使用データを必要とするクラスが各リクエストでデータベースを照会することになります。 a. 注記: > In a multi-cell (v2) setup where the cell MQ is separated from the top-level, computes cannot directly communicate with the scheduler. Thus, this option cannot be enabled in that scenario. See also the `[workarounds] disable_group_policy_check_upcall` option. 関連するオプション: ● **[filter_scheduler] enabled_filters** ● **[workarounds] disable_group_policy_check_upcall**

設定オプション = デフォルト値	型	説明
weight_classes = ['nova.scheduler.weights.all_weighers']	リスト値	スケジューラーが使用する重み付け関数。 フィルターをパスするホストのみが重み付けされます。ホストの重みは 0 で始まり、以前の重み付け関数により割り当てられた重みから追加または減算することで、これらのホストの順序が重み付けされます。重みが負の値になる可能性があります。インスタンスは、最も重み付けされた N 個のホストの 1 つにスケジュールされます。ここで、N は **[filter_scheduler] host_subset_size** です。 デフォルトでは、Nova に含まれる全重み付け関数に設定されます。 値: ● 0 個以上の文字列のリスト。各文字列は、ホストの選択に使用する重み付け関数の名前に対応します。

9.1.17. glance

以下の表は、**/etc/nova/nova.conf** ファイルの **[glance]** グループで使用できるオプションの概要をまとめたものです。

表9.16 glance

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
api_servers = None	リスト値	nova で利用可能な glance api サーバーエンドポイントのリスト HTTPS は、ssl ベースの glance api サーバーに使用されます。 **注記** エンドポイント検出で優先されるメカニズムは、keystoneauth1 の読み込みオプションを使用することです。複数のエンドポイントが必要で、何らかの理由でロードバランサーを使用できない場合にのみ api_servers を使用します。 値: - scheme://hostname:port[/path](つまり、http://10.0.1.0:9292 または https://my.glance.server/image) の形式の完全修飾 URL のリスト)。 **非推奨:** 21.0.0 **理由:** 標準の keystoneauth1 アダプターオプションによるイメージサービス設定のサポートは、17.0.0 Queens リリースで追加されました。api_servers オプションは、コンシューマーが実際の負荷分散ソリューションに切り替わる時間を確保するために一時的に保持されました。
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
debug = False	ブール値	glanceclient を使用したデバッグロギングを有効または無効にします。

設定オプション = デフォルト値	型	説明
default_trusted_certificate_ids = []	リスト値	信頼される必要のある証明書の証明書 ID のリスト。 証明書検証の信頼済み証明書 ID のデフォルトリストとして使用できます。このオプションの値は、ユーザーがインスタンス API 要求と共に信頼される証明書 ID のリストを提供した場合は無視されます。署名の検証および証明書の検証が有効で、ユーザーが代替リストを提供しなかった場合には、このオプションの値はインスタンスデータで永続化されます。証明書の検証が有効な場合に空のままにすると、ユーザーは信頼できる証明書 ID のリストを提供しないと、証明書の検証に失敗します。 関連するオプション: ● verify_glance_signatures と enable_certificate_validation の両方が有効な場合には、このオプションの値を使用することができます。

設定オプション = デフォルト値	型	説明
enable_certificate_validation = False	ブール値	イメージ署名の検証の証明書の検証を有効にします。 イメージの署名の検証中に、nova はまず、インスタンスに関連付けられた信頼された証明書のセットを使用して、イメージの署名証明書の有効性を検証します。証明書の検証が失敗した場合、署名の検証は実行されず、インスタンスはエラー状態になります。これにより、エンドユーザーは、イメージデータが変更されず、信頼できるものであることが保証されます。まだ無効にされている場合、イメージ署名の検証は発生する可能性がありますが、エンドユーザーには、イメージ署名の生成に使用される署名証明書がまだ信頼されているという保証はありません。 関連するオプション: ● このオプションは、verify_glance_signatures が有効な場合にのみ有効です。 ● このオプションが有効な場合には、default_trusted_certificate_ids の値を使用することができます。 **非推奨となったバージョン:** 16.0.0 **理由:** このオプションは、イメージ署名の検証を利用してデプロイメントの移行を容易にすることを目的としています。目的の状態の long-term は、署名の検証と証明書の検証が常に一緒に行われることです。

設定オプション = デフォルト値	型	説明
enable_rbd_download = False	ブール値	RBD 経由で Glance イメージを直接ダウンロードできるようにします。 計算ホストが、Glance API からの遅いダウンロードではなく、Ceph からローカルに直接イメージをすばやくダウンロードしてキャッシュできるようにします。これにより、イメージのダウンロード時間が 10 分から数十秒まで短縮されます。ただし、Ceph ベースのデプロイメントとコンピュートノードから Ceph へのアクセスが必要です。 関連するオプション: ● **[glance] rbd_user** ● **[glance] rbd_connect_timeout** ● **[glance] rbd_pool** ● **[glance] rbd_ceph_conf**
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
num_retries = 3	整数値	glance 操作の再試行を有効にします。 glance から / にイメージをアップロード/ダウンロードする際の再試行回数を指定します。0 は再試行なしを意味します。
`rbd_ceph_conf = `	文字列値	使用する Ceph 設定ファイルへのパス。 関連するオプション: ● このオプションは、**[glance] enable_rbd_download** が True に設定されている場合にのみ使用されます。

設定オプション = デフォルト値	型	説明
rbd_connect_timeout = 5	整数値	最初にクラスターに接続するときの RADOS クライアントのタイムアウト (秒単位)。 関連するオプション: ● このオプションは、**[glance] enable_rbd_download** が True に設定されている場合にのみ使用されます。
`rbd_pool = `	文字列値	Glance イメージが rbd ボリュームとして保存される RADOS プール。 関連するオプション: ● このオプションは、**[glance] enable_rbd_download** が True に設定されている場合にのみ使用されます。
`rbd_user = `	文字列値	rbd ボリュームとして保存されている Glance イメージにアクセスするための RADOS クライアント名。 関連するオプション: ● このオプションは、**[glance] enable_rbd_download** が True に設定されている場合にのみ使用されます。
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = image	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
timeout = None	整数値	http 要求のタイムアウト値

設定オプション = デフォルト値	型	説明
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。
verify_glance_signatures = False	ブール値	イメージ署名の検証を有効にします。 Nova は glance からのイメージ署名メタデータを使用して、そのイメージのダウンロード中に署名済みイメージの署名を検証します。イメージの署名を検証できない、またはイメージ署名のメタデータが不完全なか、利用できない場合には、nova はそのイメージをブートせず、代わりにインスタンスをエラー状態にします。これにより、エンドユーザーはサーバーを作成するために使用するイメージデータの整合性が強化されます。 関連するオプション: ● **key_manager** は署名の検証に使用されるため、key_manager グループのオプション。 ● 以下の enable_certificate_validation と default_trusted_certificate_ids はいずれも、有効になっているこのオプションによって異なります。

9.1.18. guestfs

以下の表は、**/etc/nova/nova.conf** ファイルの **[guestfs]** グループで使用できるオプションの概要をまとめたものです。

表9.17 guestfs

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
debug = False	ブール値	guestfs ロギングを有効/無効にします。 これにより、guestfs がデバッグメッセージを実行し、それらを OpenStack ロギングシステムにプッシュするように設定されます。True に設定すると、libguestfs API 呼び出しを追跡し、詳細なデバッグメッセージを有効にします。上記の機能を使用するには、libguestfs パッケージをインストールする必要があります。 関連するオプション: libguestfs は libvirt が管理する仮想マシンのアクセスと変更を行うため、以下のオプションを設定して、その仮想マシンのアクセスを付与する必要があります。 • **libvirt.inject_key** • **libvirt.inject_partition** • **libvirt.inject_password**

9.1.19. Healthcheck

以下の表は、**/etc/nova/nova.conf** ファイルの **[healthcheck]** グループで使用できるオプションの概要をまとめたものです。

表9.18 Healthcheck

設定オプション = デフォルト値	型	説明
backends = []	リスト値	ヘルスチェックを実施し、リクエストの一部としてその情報を報告できる追加のバックエンド。
detailed = False	ブール値	応答の一部として詳細情報を表示します。セキュリティーメモ: このオプションを有効にすると、監視対象のサービスに関する機密情報が漏洩する可能性があります。セキュリティーポリシーに違反しないことを確認してください。
disable_by_file_path = None	文字列値	ファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを確認します。DisableByFileHealthcheck プラグインで使用されます。

設定オプション = デフォルト値	型	説明
disable_by_file_paths = []	リスト値	ポートに基づいてファイルが存在することを確認し、アプリケーションがポート上で実行されているかどうかを判断します。文字列の port:path リストを想定します。DisableByFilesPortsHealthcheck プラグインで使用されます。
path = /healthcheck	文字列値	healtcheck 要求に応答するパス。

9.1.20. hyperv

以下の表は、**/etc/nova/nova.conf** ファイルの **[hyperv]** グループで使用できるオプションの概要をまとめたものです。

表9.19 hyperv

設定オプション = デフォルト値	型	説明
config_drive_cdrom = False	ブール値	コンフィグドライブを CD ドライブとしてマウントします。 OpenStack は、インスタンスメタデータをコンフィグドライブに書き込むように設定できます。これは、インスタンスのブート前にアタッチされます。設定ドライブは、ディスクドライブ (デフォルト) または CD ドライブとして接続できます。 関連するオプション: ● このオプションは、**force_config_drive** オプションを **True** に設定したり、インスタンスを作成する REST API 呼び出しに **--config-drive=True** フラグを付ける場合に意味を持ちます。 ● コンフィグドライブイメージとして CD ドライブを使用するには、**config_drive_format** オプションを **iso9660** に設定する必要があります。 ● Hyper-V でコンフィグドライブを使用するには、**mkisofs_cmd** の値を **mkisofs.exe** インストールへの完全パスに設定する必要があります。さらに、**qemu_img_cmd** の値を **qemu-img** コマンドインストールへのフルパスに設定する必要があります。 ● **force_config_drive** オプションを **True** に設定すると、Compute サービスが常にコンフィグドライブを作成するように設定できます。

設定オプション = デフォルト値	型	説明
config_drive_inject_password = False	ブール値	設定ドライブにパスワードを挿入します。 有効な場合、admin パスワードはコンフィグドライブイメージから利用可能になります。 関連するオプション: ● このオプションは、**force_config_drive** など、Hyper-V での設定ドライブの使用を可能にする他のオプションと一緒に使用すると意味があります。
dynamic_memory_ratio = 1.0	浮動小数点の値	動的なメモリー比率 1 より大きい値に設定すると、動的メモリー割り当て (バルーニング) が有効になります。この値は、インスタンスに割り当てられた RAM の合計と、その起動 RAM 容量の比率を表します。たとえば、RAM が 1024MB のインスタンスで 2.0 の比率は、起動時に 512 MB の RAM に割り当てられることを意味します。 値: ● 1.0: 動的メモリー割り当てを無効にします (デフォルト)。 ● 1.0 より大きい浮動小数点値: 起動時に、この値で割った暗黙の RAM の合計の割り当てを有効にします。
enable_instance_metrics_collection = False	ブール値	インスタンスメトリック収集の有効化 Hyper-V のメトリック API を使用してインスタンスのメトリックコレクションを有効にします。収集したデータは、以下のような他のアプリケーションやサービスで取得できます。例: Ceilometer

設定オプション = デフォルト値	型	説明
enable_remotefx = False	ブール値	RemoteFX 機能の有効化 これには、Windows / Hyper-V Server 2012 R2 以降および RDS-Virtualization 機能に少なくとも 1 つの DirectX 11 対応グラフィックスアダプターが必要です。 RemoteFX を使用するインスタンスは、以下のフレーバーの追加スペックを使用して要求できます。 **os:resolution.ゲスト仮想マシン画面の解像度のサイズ。許可される値** 1024x768, 1280x1024, 1600x1200, 1920x1200, 2560x1600, 3840x2160 **3840x2160** は Windows / Hyper-V Server 2016 でのみ利用できます。 **os:monitorsゲスト VM 番号のモニター数。許可される値** [1, 4] - Windows / Hyper-V Server 2012 R2 [1, 8] - Windows / Hyper-V Server 2016 **os:vram**ゲスト仮想マシン VRAM の量。Windows / Hyper-V Server 2016 でのみ利用可能です。 Acceptable values:: 64, 128, 256, 512, 1024
`instances_path_share = `	文字列値	インスタンスパス共有 instances_path ディレクトリーにマッピングされ、変更機能でファイルをターゲットホストにコピーするために使用される Windows 共有の名前。空白のままにすると、管理共有 (非表示のネットワーク共有) が使用され、ローカルで使用されているのと同じ instances_path が検索されます。 値: - "": 管理共有が使用されます (デフォルト)。 - Windows 共有の名前。 関連するオプション: - "instances_path": このオプションが空白のままの場合に使用されるディレクトリー。

設定オプション = デフォルト値	型	説明
iscsi_initiator_list = []	リスト値	iSCSI セッションを安定させるために使用される iSCSI イニシエーターのリスト。 何も指定しないと、Microsoft iSCSI イニシエーターサービスがイニシエーターを選択します。
limit_cpu_features = False	ブール値	CPU 機能の制限 このフラグは、さまざまな CPU 機能を備えたホストへのライブマイグレーションをサポートするために必要であり、インスタンスによって使用される CPU 機能を制限するためにインスタンスの作成中にチェックされます。
mounted_disk_query_retry_count = 10	整数値	マウントされたディスククエリーの再試行回数 マウントされたディスクの確認を再試行する回数。クエリーは、デバイスが見つかるか、再試行回数に達するまで実行されます。 値: ● 正の整数値。1 より大きい値を推奨します (デフォルト: 10)。 関連するオプション: ● ディスクマウントの再試行間の時間間隔は、mounted_disk_query_retry_interval オプションで宣言されます。
mounted_disk_query_retry_interval = 5	整数値	マウントされたディスククエリーの再試行間隔 マウントされたディスクのチェック間隔 (秒単位)。 値: ● 秒単位の時間 (デフォルト: 5)。 関連するオプション: ● このオプションは、mount_disk_query_retry_count が 1 よりも大きい場合に意味を持ちます。 ● 再試行ループは、mount_disk_query_retry_count および mounted_disk_query_retry_interval 設定オプションで実行されます。

設定オプション = デフォルト値	型	説明
power_state_check_timeframe = 60	整数値	電源状態チェックのタイムフレーム インスタンスの電源状態の変更のためにチェックする時間枠。このオプションは、指定の時間内に WMI インターフェイスを介して Hyper-V からインスタンスの状態を取得するのに使用します。 値: ● 秒単位の時間枠 (デフォルト: 60)。
power_state_event_polling_interval = 2	整数値	電源状態イベントのポーリング間隔 インスタンスの電源状態の変更イベントのポーリング頻度。電源状態イベントのリスナー間隔を指定の値に設定します。このオプションは、それ自体を再起動するインスタンスのライフサイクルに関する通知を強化します。Operator はこの値を変更する必要がある訳ではありません。 値: ● 秒単位の時間 (デフォルト: 2)。

設定オプション = デフォルト値	型	説明
qemu_img_cmd = qemu-img.exe	文字列値	qemu-img コマンド QEMU-img は、異なるイメージタイプ間の変換など、一部のイメージ関連の操作に必要です。ここから入手できます: (http://qemu.weilnetz.de/) または自動的に設定される Cloudbase OpenStack Hyper-V Compute Driver (https://cloudbase.it/openstack-hyperv-driver/) をインストールできますこの設定オプションの適切なパス。qemu-img.exe の完全パスを指定するか、PATH 環境変数でそのパスを設定し、このオプションをデフォルト値のままにします。 値: ● qemu-img 実行可能ファイルの名前。nova-compute サービスと同じディレクトリーにある場合、またはそのパスが PATH 環境変数にある場合 (デフォルト)。 ● qemu-img コマンドのパス (DRIVELETTER:\PATH \ TO \ QEMU-IMG \ COMMAND)。 関連するオプション: ● config_drive_cdrom オプションが False の場合、qemu-img を使用して ISO を VHD に変換し、それ以外の場合はコンフィグドライブは ISO のままになります。Hyper-V でコンフィグドライブを使用するには、**mkisofs_cmd** の値を **mkisofs.exe** インストールへの完全パスに設定する必要があります。
use_multipath_io = False	ブール値	iSCSI ディスクまたは FC ディスクを接続する際にマルチパス接続を使用します。 これには、Multipath IO Windows 機能を有効にする必要があります。MPIO は、このようなデバイスを要求するように設定する必要があります。

設定オプション = デフォルト値	型	説明
volume_attach_retry_count = 10	整数値	ボリュームアタッチの再試行回数 ボリュームの割り当てを再試行する回数。ボリュームのアタッチは、成功までか、指定された再試行数に達するまで再試行されます。 値: ● 正の整数値 (デフォルト: 10)。 関連するオプション: ● 接続試行の間隔は、volume_attach_retry_interval オプションで宣言されます。
volume_attach_retry_interval = 5	整数値	ボリュームアタッチの再試行の間隔 ボリューム接続の試行間隔 (秒単位)。 値: ● 秒単位の時間 (デフォルト: 5)。 関連するオプション: ● このオプションは、volume_attach_retry_count が 1 よりも大きい場合に意味を持ちます。 ● 再試行ループは、volume_attach_retry_count および volume_attach_retry_interval 設定オプションで実行されます。

設定オプション = デフォルト値	型	説明
vswitch_name = None	文字列値	外部仮想スイッチ名 Hyper-V Virtual Switch は、Hyper-V サーバーロールのインストールで利用可能なソフトウェアベースの layer-2 Ethernet ネットワークスイッチです。スイッチには、プログラムで管理され、拡張可能な機能が追加され、仮想マシンを仮想ネットワークと物理ネットワークの両方に接続することができます。また、Hyper-V Virtual Switch は、セキュリティー、分離、およびサービスレベルのポリシー適用を提供します。この設定オプションで表される vSwitch は、外部のものである必要があります (内部またはプライベートではありません)。 値: ● 指定しないと、利用可能な vswitches リストの最初に使用されます。このリストは WQL を使用してクエリーされます。 ● 仮想スイッチ名。
wait_soft_reboot_seconds = 60	整数値	ソフトリブート秒を待ちます。 ソフトリブート要求後にインスタンスがシャットダウンするのを待機する秒数。インスタンスがこのウィンドウ内でシャットダウンしない場合は、ハードリブートにフォールバックします。 値: ● 秒単位の時間 (デフォルト: 60)。

9.1.21. image_cache

次の表に、**/etc/nova/nova.conf** ファイルの **[image_cache]** グループで使用できるオプションの概要を示します。

表9.20 image_cache

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
manager_interval = 2400	整数値	イメージキャッシュマネージャーの実行間で待機する秒数。 複数の nova-compute サービスにわたって **[DEFAULT]/instances_path** 設定オプションに共有ストレージを使用する場合、この周期は多数のインスタンスを処理する可能性があることに注意してください。同様に、クラスターを管理するコンピューティングドライバー (vmwareapi.VMwareVCDriver など) を使用すると、多数のインスタンスが処理される可能性があります。したがって、予想される負荷に合わせて時間間隔を調整するか、共有ストレージアグリゲート内の 1 つの nova-compute サービスでのみ実行する必要がある場合があります。 値: ● 0: デフォルトの 60 秒間隔で実行 (非推奨) ● -1: 無効 ● その他の値 関連するオプション: ● **[DEFAULT]/compute_driver** ● **[DEFAULT]/instances_path**
precache_concurrency = 1	整数値	イメージの事前キャッシュを並行してトリガーする計算ホストの最大数。 イメージの事前キャッシュ要求が行われると、コンピュートノードに接続してダウンロードを開始します。この数は、並行して発生するそれらの数を制限します。数値が大きいと、compute が並行して機能し、操作完了までの時間が短縮される可能性がありますが、Image サービスにも DDoS が実行される可能性があります。数値が小さいと、順次操作が実行され、イメージサービスの負荷は低くなりますが、完了までの実行時間が長くなる可能性が高くなります。
remove_unused_base_images = True	ブール値	未使用のベースイメージを削除するべきか？
remove_unused_original_minimum_age_seconds = 86400	整数値	未使用のサイズ変更されたベースイメージは、この期間削除されません。

設定オプション = デフォルト値	型	説明
remove_unused_resized_minimum_age_seconds = 3600	整数値	未使用のサイズ変更されたベースイメージは、この期間削除されません。
subdirectory_name = _base	文字列値	キャッシュされたイメージの場所。 これは完全パスではありません。**$instances_path** に相対するフォルダー名のみになります。compute-host キャッシュされたイメージの場合は、**base$my_ip**に設定します。

9.1.22. ironic

以下の表は、**/etc/nova/nova.conf** ファイルの **[ironic]** グループで使用できるオプションの概要をまとめたものです。

表9.21 ironic

設定オプション = デフォルト値	型	説明
api_max_retries = 60	整数値	要求の競合時に再試行する回数。0 に設定される場合、再試行は 1 回のみ試行します。 関連するオプション: ● api_retry_interval
api_retry_interval = 2	整数値	要求を再試行するまで待機する秒数。 関連するオプション: ● api_max_retries
auth-url = None	文字列値	認証 URL
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル

設定オプション = デフォルト値	型	説明
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
partition_key = None	文字列値	大文字小文字の区別のないキーで、このサービスが管理することのできるノードのセットを、一致する conductor_group プロパティーを持つ Ironic 内のノードセットに制限します。未設定の場合は、このサービスで利用可能なすべてのノードが管理できるようになります。これを空の文字列 ("") に設定すると、デフォルトのコンダクタグループと一致し、オプションを未設定のままにするのとは異なることに注意してください。
password = None	文字列値	ユーザーのパスワード
peer_list = []	リスト値	この partition_key 設定値を持つすべての nova-compute サービス (このホストを含む) のホスト名のリスト。partition_key 値に一致するノードは、ここで指定したすべてのサービス間で分散されます。partition_key が設定されていない場合は、このオプションは無視されます。
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID

設定オプション = デフォルト値	型	説明
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
serial_console_state_timeout = 10	整数値	ノードのシリアルコンソールの状態が変更されるのを待つタイムアウト (秒)。タイムアウトを無効にするには 0 に設定します。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = baremetal	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
system-scope = None	文字列値	システム操作のスコープ
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	User ID
username = None	文字列値	ユーザー名

設定オプション = デフォルト値	型	説明
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。

9.1.23. key_manager

以下の表は、**/etc/nova/nova.conf** ファイルの **[key_manager]** グループで使用できるオプションの概要をまとめたものです。

表9.22 key_manager

設定オプション = デフォルト値	型	説明
auth_type = None	文字列値	作成する認証認証情報のタイプ。使用できる値は、**トークン**、**パスワード**、keystone_token、および keystone_password です。コンテキストが認証情報ファクトリーに渡されない場合に必要です。
auth_url = None	文字列値	このエンドポイントを使用して Keystone に接続します。
backend = barbican	文字列値	キーマネージャーの実装を指定します。オプションは barbican および vault です。デフォルトは barbican です。しばらくの間、[key_manager]/api_class を使用して以前に設定された値をサポートします。
domain_id = None	文字列値	ドメインスコーピングのドメイン ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
domain_name = None	文字列値	ドメインスコーピングのドメイン名。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
fixed_key = None	文字列値	16 進法で指定されるキーマネージャーによって返されるキーが修正されました。 値: ● 空の文字列または 16 進値のキー
password = None	文字列値	認証のパスワード。**パスワード** および **keystone_password** auth_type には必須です。

設定オプション = デフォルト値	型	説明
project_domain_id = None	文字列値	プロジェクトのドメイン ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
project_domain_name = None	文字列値	プロジェクトのドメイン名。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
project_id = None	文字列値	プロジェクトスコーピングのプロジェクト ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
project_name = None	文字列値	プロジェクトスコーピングのプロジェクト名。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
reauthenticate = True	ブール値	現在のトークンの有効期限が切れる場合の新規トークンの取得を許可します。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
token = None	文字列値	認証のトークン。認証情報ファクトリーにコンテキストが渡されない場合には、**トークン** および **keystone_token** auth_type には必須です。
trust_id = None	文字列値	信頼スコープの信頼 ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
user_domain_id = None	文字列値	認証用のユーザーのドメイン ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
user_domain_name = None	文字列値	認証用のユーザーのドメイン名。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
user_id = None	文字列値	認証のユーザー ID。**keystone_token** および **keystone_password** auth_type の場合はオプションです。
username = None	文字列値	認証用のユーザー名。**パスワード** auth_type には必須です。**keystone_password** auth_type の場合はオプションです。

9.1.24. keystone

以下の表は、**/etc/nova/nova.conf** ファイルの **[keystone]** グループ下で使用できるオプションの概要をまとめたものです。

表9.23 keystone

設定オプション = デフォルト値	型	説明
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = identity	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。

設定オプション = デフォルト値	型	説明
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
timeout = None	整数値	http 要求のタイムアウト値
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。

9.1.25. keystone_authtoken

以下の表は、**/etc/nova/nova.conf** ファイルの **[keystone_authtoken]** グループで使用できるオプションの概要をまとめたものです。

表9.24 keystone_authtoken

設定オプション = デフォルト値	型	説明
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ
auth_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。このオプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。**非推奨**:Queens 以降 *理由:* auth_uri オプションは www_authenticate_uri が優先されるため非推奨となり、S リリースで削除される予定です。

設定オプション = デフォルト値	型	説明
auth_version = None	文字列値	Identity API エンドポイントの API バージョン。
cache = None	文字列値	Swift キャッシュオブジェクトが保存される環境キーを要求します。auth_token ミドルウェアを Swift キャッシュと共にデプロイする場合は、このオプションを使用して、ミドルウェアが swift とキャッシングバックエンドを共有するようにします。それ以外の場合は、代わりに **memcached_servers** オプションを使用します。
cafile = None	文字列値	HTTPs 接続の検証時に使用する PEM でエンコードされた認証局。デフォルトはシステム CA です。
certfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
delay_auth_decision = False	ブール値	ミドルウェア内の認可要求を処理せず、承認の決定をダウンストリームの WSGI コンポーネントに委譲します。
enforce_token_bind = permissive	文字列値	トークンバインディングの使用および種別を制御するために使用されます。トークンバインディングのチェックを無効するには、"disabled"に設定します。バインドタイプがサーバーの認識する形式の場合にはバインディング情報を検証し、そうでない場合には無視するには、"permissive"(デフォルト) に設定します。"strict"は"permissive"と類似していますが、バインドタイプが不明な場合にはトークンが拒否されます。"required"の場合は、いずれかの形式のトークンバインディングが必要です。最後に、トークンに指定する必要のあるバインディングメソッドの名前。
http_connect_timeout = None	整数値	Identity API サーバーと通信する際の要求タイムアウト値。
http_request_max_retries = 3	整数値	Identity API サーバーと通信する際に再接続を試行する回数。
include_service_catalog = True	ブール値	(オプション)X-Service-Catalog ヘッダーを設定するかどうかを示します。False の場合、ミドルウェアはトークンの検証時にサービスカタログを要求せず、X-Service-Catalog ヘッダーを設定しません。
insecure = False	ブール値	HTTPS 接続を確認します。

設定オプション = デフォルト値	型	説明
interface = internal	文字列値	Identity API エンドポイントに使用するインターフェイス。有効な値は、"public"、"internal" (デフォルト)、または "admin" です。
keyfile = None	文字列値	ID サーバーでクライアント証明書が必要な場合に必要です。
memcache_pool_conn_get_timeout = 10	整数値	(オプション) プールから memcached クライアント接続を取得するまで操作が待機する秒数。
memcache_pool_dead_retry = 300	整数値	(オプション)memcached サーバーが停止しているとみなされる秒数。この秒数が経過すると再試行されます。
memcache_pool_maxsize = 10	整数値	(オプション) すべての memcached サーバーへのオープン接続の最大合計数。
memcache_pool_socket_timeout = 3	整数値	(オプション)memcached サーバーと通信する際のソケットのタイムアウト (秒単位)。
memcache_pool_unused_timeout = 60	整数値	(オプション)memcached への接続がプール内で未使用の状態を維持する秒数。この秒数が経過すると終了されます。
memcache_secret_key = None	文字列値	(オプション、memcache_security_strategy が定義されている場合には必須) この文字列は鍵の導出に使用されます。
memcache_security_strategy = None	文字列値	(オプション) 定義されている場合は、トークンデータを認証、または認証して暗号化する必要があるかどうかを示します。MAC の場合、キャッシュでトークンデータが認証されます (HMAC を使用)。ENCRYPT の場合、キャッシュでトークンデータが暗号化され、認証されます。値がこれらのオプションのいずれでもない場合や空の場合には、auth_token は初期化時に例外を発生させます。
memcache_use_advanced_pool = False	ブール値	(オプション) 高度な (eventlet に対して安全な) memcached クライアントプールを使用します。高度なプールは python 2.x でのみ動作します。
memcached_servers = None	リスト値	オプションで、キャッシュに使用する memcached サーバーのリストを指定します。未定義のままの場合、トークンは代わりに処理中にキャッシュされます。
region_name = None	文字列値	アイデンティティーサーバーがあるリージョン。

設定オプション = デフォルト値	型	説明
service_token_roles = ['service']	リスト値	サービストークンに存在する必要があるロールの選択。サービストークンは、期限切れのトークンを使用できることを要求できるため、このチェックでは実際のサービスのみがこのトークンを送信するように厳密に制御する必要があります。ここでのロールは ANY チェックとして適用されるため、このリストのロールはすべて存在している必要があります。後方互換性の理由から、現在 allow_expired チェックにのみ影響します。
service_token_roles_required = False	ブール値	後方互換性の理由から、service_token_roles チェックを有効としてパスしない有効なサービストークンをパスさせる必要があります。これを true に設定することが今後のリリースでデフォルトとなり、可能な場合は有効にされる必要があります。
service_type = None	文字列値	サービスカタログに表示されるサービスの名前または種別。これは、制限されたアクセスルールがあるトークンを検証するために使用されます。
token_cache_time = 300	整数値	トークンの検証に過剰な時間を費やすのを防ぐために、ミドルウェアは、設定可能な期間 (秒単位) 中は以前に見たトークンをキャシュします。キャッシュを完全に無効にするには -1 に設定します。
www_authenticate_uri = None	文字列値	パブリックの Identity API エンドポイントを完了します。このエンドポイントは、すべてのエンドユーザーがアクセスできる必要があるため、"admin" エンドポイントにすることはできません。認証されていないクライアントは、認証のためにこのエンドポイントにリダイレクトされます。このエンドポイントにはバージョンを指定しないことが理想的ですが、ワイルドカードでのクライアントのサポートは異なります。バージョン指定された v2 エンドポイントを使用している場合、通常エンドユーザーがそのエンドポイントに到達できない可能性があるため、これはサービスユーザーがトークンを検証するために使用するエンドポイントにすることは**できません**。

9.1.26. libvirt

以下の表は、**/etc/nova/nova.conf** ファイルの **[libvirt]** グループで使用できるオプションの概要をまとめたものです。

表9.25 libvirt

設定オプション = デフォルト値	型	説明
`connection_uri = `	文字列値	選択した仮想化タイプのデフォルトの libvirt URI を上書きします。 これが設定されている場合、Nova はこの URI を使用して libvirt に接続します。 値: ● **qemu:///system** のような URI。 > This is only necessary if the URI differs to the commonly known URIs for the chosen virtualization type. 関連するオプション: ● **virt_type**: ここでデフォルト値として使用されるものに影響を与えます。
cpu_mode = None	文字列値	インスタンスが持つ CPU モードの設定に使用されます。 **virt_type="kvm|qemu"** の場合、これはデフォルトで **host-model** になります。それ以外の場合は、デフォルトは **none** になります。 関連するオプション: ● **cpu_models**: これは、**cpu_mode** が **custom** に設定されている場合にのみ設定する必要があります。そうでない場合にはエラーが生じ、インスタンスの起動は失敗します。
cpu_model_extra_flags = []	リスト値	ゲスト CPU フラグを有効または無効にします。 CPU フラグを明示的に有効または無効にするには、**+flag** または **-flag** 表記を使用します。**+** 記号を使用すると、ゲストの CPU フラグが有効になります。一方、**-** 記号はこれを無効にします。**+** および **-** を指定しない場合、フラグが有効になります。これがデフォルトの動作になります。たとえば、次のように指定した場合 (前述の CPU モデルと機能がホストハードウェアとソフトウェアでサポートされていると仮定):: > [libvirt] cpu_mode = custom cpu_models = Cascadelake-Server cpu_model_extra_flags = -hle, -rtm, +ssbd, mtrr

設定オプション = デフォルト値	型	説明
		Nova は、ゲストの **hle** フラグと **rtm** フラグを無効にします。**ssbd** と **mttr** が有効になります (**+** 接頭辞も **-** 接頭辞も指定されていないため)。 CPU フラグは大文字と小文字を区別しません。以下の例では、ゲストに対して **pdpe1gb** フラグが無効になります。**vmx** フラグおよび **pcid** フラグが有効になります。 [libvirt] cpu_mode = custom cpu_models = Haswell-noTSX-IBRS cpu_model_extra_flags = -PDPE1GB, +VMX, pcid 追加の CPU フラグを指定することは、**cpu_mode** config 属性の 3 つの可能な値すべてと組み合わせて有効です: **custom** (これには、**cpu_models** config 属性を介して明示的な CPU モデルを指定する必要もあります)、**host-model**、または **host-passthrough**。 QEMU が特定の CPU 機能を無効にする可能性があるため、**host-passthrough** CPU モードであっても追加の CPU フラグを設定しなければならない場合があります。この例として、Intel の invtsc (不変タイムスタンプカウンター)CPU フラグがあります。このフラグを Nova インスタンスに公開する必要がある場合は、明示的に有効にする必要があります。 **cpu_model_extra_flags** で使用できる値は、使用中の CPU モデルによって異なります。特定の CPU モデルで使用可能な CPU 機能フラグについては、**/usr/share/libvirt/cpu_map/*.xml** を参照してください。 特定の CPU フラグに関する特記事項: **pcid**(**Meltdown** CVE 修正を適用した結果としてのゲストパフォーマンスの低下を軽減する Intel プロセッサー機能)。**custom** CPU モードでこのフラグを設定する場合、すべての CPU モデル (QEMU および libvirt で定義されている) が必要なわけではありません。 ● **pcid** 機能を含む唯一の仮想 CPU モデルは、Intel の Haswell、Broadwell、および Skylake バリアントです。 ● libvirt/QEMU CPU モデル Nehalem、Westmere、SandyBridge、および IvyBridge は、同じ名前のホスト CPU に含まれている場合でも、デフォルトで **pcid** 機能を公開し**ません**。つまり上記の仮想 CPU モデルを使用する場合は、**PCID** を明示的に指定する必要があります。 libvirt ドライバーのデフォルトの CPU モードである **host-model** は、ゲストの **PCID** CPU フラグの処理

設定オプション = デフォルト値	型	説明
		に関して正しいことを行います–更新されたプロセッサマイクロコード、ホストとゲストのカーネル、libvirt、および QEMU を実行していると **仮定し**ます。もう1つのモードである **host-passthrough** は、**PCID** がハードウェアで使用可能かどうかを確認し、使用可能な場合は直接 Nova ゲストに渡します。したがって、**PCID** の コンテキストで は、CPU モード (**host-model** または **host-passthrough**) のいずれかについては、**cpu_model_extra_flags** を使用する必要はありません。 関連するオプション: ● cpu_mode ● cpu_models
cpu_models = []	リスト値	ホストが対応する CPU モデルの順序付きリスト。 より一般的で詳細度の低い CPU モデルが先にリストされるように、リストが順序付けられることが予想されます。例: **SandyBridge,IvyBridge,Haswell,Broadwell**、後者の CPU モデルの機能は、以前の CPU モデルに多大なものです。 値: ● 名前付きの CPU モデルは、**virsh cpu-models ARCH** から確認でき、ARCH はホストのアーキテクチャーになります。 関連するオプション: ● **cpu_mode**: これは、特定の名前付き CPU モデルを (**cpu_models** を介して) 設定する場合にのみ **custom** に設定する必要があります。そうでない場合にはエラーが生じ、インスタンスの起動は失敗します。 ● **virt_type**: 仮想化タイプ **kvm** および **qemu** のみがこれを使用します。 a. 注意:: ハードウェアで完全にサポートできるモデルのみを指定するように注意してください。

設定オプション = デフォルト値	型	説明
device_detach_attempts = 8	整数値	ドライバーが libvirt でデバイスの接続解除を試行する最大試行回数。 関連するオプション: ● :oslo.config:option:**libvirt.device_detach_timeout**
device_detach_timeout = 20	整数値	ドライバーが、デバイスの接続解除を再トリガーする前に、libvirt からの成功イベントまたは失敗イベントを待機する最大秒数。 関連するオプション: ● :oslo.config:option:**libvirt.device_detach_attempts**
disk_cachemodes = []	リスト値	異なるディスクタイプに使用する特定のキャッシュモード。 例: file=directsync,block=none,network=writeback ローカルストレージまたは直接接続ストレージの場合は、ライトスルー (デフォルト) モードを使用することを推奨します。これにより、データの整合性が確保され、ゲストで実行されるアプリケーション、特に読み取り操作で許容できる I/O パフォーマンスが得られます。ただし、(O_SYNC を使用する) 同期 I/O 操作よりも、ダイレクト I/O 操作 (O_DIRECT) よりもパフォーマンスが高いため、リモート NFS ストレージにはキャッシュモード none が推奨されます。キャッシュモードは、すべてのゲスト I/O 操作を効果的にホスト上のダイレクト I/O 操作 (この環境の NFS クライアント) に変換します。 可能なキャッシュモード: ● デフォルト: "It Depends" – Nova マネージドディスクの場合、ホストファイルシステムが Linux の **O_DIRECT** セマンティクスに対応している場合は **none**。それ以外の場合は **writeback**。ボリュームドライバーの場合、デフォルトはドライバーに依存します。SMBFS と Virtuzzo (**writeback** を使用) を除くすべてのドライバーは、**none** となります。 ● なし: キャッシングモードを none に設定すると、ホストのページキャッシュは無効になりますが、ゲストのディスクライトキャッシュは有効になります。このモードでは、書き込み操作がホストページキャッシュをバイパスし、ディスク書き込みキャッシュに直接移動するため、ゲストの

設定オプション = デフォルト値	型	説明
		書き込みパフォーマンスが最適です。ディスク書き込みキャッシュがバッテリバックアップされている場合、またはゲスト内のアプリケーションまたはストレージスタックがデータを適切に転送する場合 (fsync 操作またはファイルシステムバリアを介して)、データの整合性を確保できます。ただし、ホストページキャッシュは無効になっているため、writethrough モードなど、ホストページキャッシュが有効になっているモードの場合、ゲストの読み取りパフォーマンスは適切ではありません。マルチ接続可能なブロックストレージボリュームと同様に、共有可能なディスクデバイスは、設定に関係なくキャッシュモードが **none** に設定されます。 ● writethrough: caching を writethrough モードに設定したと、ホストページキャッシュが有効になりますが、ゲストのディスク書き込みキャッシュは無効になります。したがって、このキャッシュモードは、ゲスト内のアプリケーションとストレージスタックがデータを永続ストレージに適切に転送しない場合でも (fsync 操作またはファイルシステムバリアを介して) データの整合性を保証します。ホストページキャッシュはこのモードで有効になっているため、ゲストで実行されているアプリケーションの読み取りパフォーマンスが一般的です。ただし、ディスク書き込みキャッシュが無効になるため、書き込みパフォーマンスが低下する可能性があります。 ● writeback: キャッシングを writeback モードに設定すると、ゲストに対してホストページキャッシュとディスク書き込みキャッシュの両方が有効になります。このため、ゲストで実行しているアプリケーションの I/O パフォーマンスは優れていますが、データは電源障害で保護されません。そのため、このキャッシュモードは、潜在的なデータ損失が懸念しない一時的なデータにのみ推奨されます。注意: 定のバックエンドディスクメカニズムは、安全なライトバックキャッシュセマンティクスを提供する場合があります。具体的には、QEMU の統合 RBD ドライバーなどのホストページキャッシュをバイパスします。Ceph ドキュメントは、データの安全性を維持しながら、パフォーマンスを最大化するためにこれをライトバックに設定することを推奨します。 ● directsync: writethrough と同様に、ホストページキャッシュはバイパスします。 ● unsafe: 安全でないキャッシュモードは、キャッシュ転送操作を完全に無視します。名前が示すように、このキャッシュモードは、データ損失が懸念しない一時的なデータにのみ使用する必要があります。このモードは、ゲストのインストールを迅速化するのに便利ですが、実稼働環境では別のキャッシュモードに切り替える必要があります。

設定オプション = デフォルト値sk_prefix = None	型 文字列値	説明 インスタンスに接続されているデバイスのデフォルトディスク接頭辞を上書きします。 これが設定されている場合は、バスの空きディスクデバイス名を特定するために使用されます。 値: ● たとえば、**sda** や **hda** などの有効なディスクデバイス名になります。これは、デバイス名が、sd、xvd、uvd、vd などの仮想化タイプの一般的に既知のデバイス名の接頭辞と異なる場合にのみ必要です。 関連するオプション: ● **virt_type**: 使用するデバイスの種類に影響します。これにより、デフォルトのディスク接頭辞が決まります。

enabled_perf_events = []	リスト値	統計を監視および収集するパフォーマンスイベント。
		
		これにより、イベントのリストを指定してゲストの低レベルのパフォーマンスを監視し、Linux カーネルの **perf** インフラストラクチャーを使用する libvirt ドライバーを介して関連する統計を収集できます。この設定属性が設定されると、Nova は libvirt ゲスト XML を生成して指定されたイベントを監視します。
		
		たとえば、CPU サイクルの数 (合計/ 経過) とキャッシュミスの数を監視するには、次のように有効にします。
		
		`[libvirt]`
		`enabled_perf_events = cpu_clock, cache_misses`
		
		可能な値: 文字列リスト。サポートされているイベントのリストをここで見つけることができます。**Intel CMT イベント - `cmt**、**mbmbt** および **mbml** - 最近の Linux カーネルバージョン (4.14+) ではサポートされておらず、nova によって無視されます。
		
		— https://libvirt.org/formatdomain.html#elementsPerf.

設定オプション = デフォルト値	型	説明
file_backed_memory = 0	整数値	ファイルベースのメモリーに利用可能な容量 (MiB 単位)。 ファイルベースのメモリーを無効にするには 0 に設定します。 有効にすると、インスタンスは **/etc/libvirt/qemu.conf** の **memory_backing_dir** オプションで指定したディレクトリーにメモリーファイルを作成します。デフォルトの場所は **/var/lib/libvirt/qemu/ram** です。 有効にすると、このオプションに定義された値はノードのメモリー容量として報告されます。コンピュートノードシステムのメモリーは、カーネルのページキャッシュメカニズムを介してファイルベースのメモリーのキャッシュとして使用されます。 a. 注意: この機能は hugepages と互換性がありません。 b. note:: この機能はメモリーのオーバーコミットと互換性がありません。 関連するオプション: ● **virt_type** は **kvm** または **qemu** に設定する必要があります。 ● **ram_allocation_ratio** は 1.0 に設定する必要があります。
gid_maps = []	リスト値	guid ターゲットと範囲のリスト。Syntax は guest-gid:host-gid:count です。最大 5 つまで許容されます。
hw_disk_discard = None	文字列値	nova 管理ディスク用の discard オプション。 以下が必要です。 ● Libvirt >= 1.0.6 ● Qemu> = 1.5 (raw 形式) ● Qemu>= 1.6 (qcow2 形式)

設定オプション = デフォルト値	型	説明
hw_machine_type = None	リスト値	qemu ゲストまたは KVM ゲストの場合は、ホストアーキテクチャーごとにデフォルトのマシン種別を指定するには、このオプションを設定します。お使いの環境でサポートされているマシンタイプのリストは、**virsh capabilities** コマンドの出力を確認することで見つけることができます。この設定オプションの値の形式は **host-arch=machine-type** です。例: **x86_64=machinetype1,armv7l=machinetype2**
`images_rbd_ceph_conf = `	文字列値	使用する Ceph 設定ファイルへのパス
images_rbd_glance_copy_poll_interval = 15	整数値	イメージをローカル rbd ストアにコピーするように要求した後、Glance をポーリングする間隔 (秒単位)。これは、Glance にコピー完了の報告を依頼する頻度に影響するため、すぐにわかるように十分に短くする必要がありますが、Glance サーバーに過度の負荷が発生するほど積極的ではありません。 関連するオプション: ● images_type - **rbd** に設定する必要があります。 ● images_rbd_glance_store_name - must be set to a store name
images_rbd_glance_copy_timeout = 600	整数値	Glance がローカル rbd ストアへのイメージコピーを完了するまで待機する全体の最大時間。これは、ローカルストアとイメージが通常存在するストアとの間のネットワークリンクを介して大きなイメージをコピーできるようにするのに十分な長さである必要があります。これを長く設定することの欠点は、イメージコピーが停止したり、処理が遅すぎて役に立たない場合を捉えることです。実際のエラーは Glance によって報告され、ポーリング間隔に従って通知されます。 関連オプション: * images_type - **rbd** に設定する必要があります。* images_rbd_glance_store_name - ストア名に設定する必要があります。* images_rbd_glance_copy_poll_interval - 障害通知までの時間を制御します。

設定オプション = デフォルト値	型	説明
`images_rbd_glance_store_name = `	文字列値	このノードで使用されている rbd クラスターを表す Glance ストアの名前。設定すると、Nova は既存の非ローカルストアからこのオプションで指定されたストアにイメージをコピーするように、起動前に Glance に要求できるため、適切なコピーオンライト動作が維持されます。 関連するオプション: ● images_type - **rbd** に設定する必要があります。 ● images_rbd_glance_copy_poll_interval - ステータスのポーリング頻度を制御します。 ● images_rbd_glance_copy_timeout - 全体的なコピータイムアウトを制御します。
images_rbd_pool = rbd	文字列値	rbd ボリュームが保存される RADOS プール
images_type = default	文字列値	VM イメージの形式。 default が指定されている場合には、このフラグの代わりに use_cow_images フラグが使用されます。 関連するオプション: ● compute.use_cow_images ● images_volume_group ● [workarounds]/ensure_libvirt_rbd_instance_dir_cleanup ● compute.force_raw_images
images_volume_group = None	文字列値	images_type=lvm を指定すると、仮想マシンイメージに使用される LVM ボリュームグループ 関連するオプション: ● images_type

設定オプション = デフォルト値	型	説明
inject_key = False	ブール値	起動時に SSH キーの挿入を許可します。 これを実行するために、イメージ内にエージェントは必要ありません。ホストで **libguestfs** が利用可能な場合は、それが使用されます。それ以外の場合は **nbd** が使用されます。イメージのファイルシステムがマウントされ、REST API 呼び出しで提供される SSH キーは root ユーザーの SSH キーとして挿入され、そのユーザーの **authorized_keys** に追加されます。必要に応じて SELinux コンテキストが設定されます。インスタンスがボリュームから起動すると、インジェクションができ **ない** ことに注意してください。 この設定オプションは、インスタンスディスクを直接変更できるようにし、config_drive オプションまたはメタデータサービスからのデータを使用して cloud-init が実行できる内容には影響しません。 **Linux** ディストリビューションゲストのみ。 関連するオプション: ● **inject_partition**: このオプションは、ファイルシステムの検出および使用方法について決定します。また、インジェクションを無効にすることもできます。

設定オプション = デフォルト値	型	説明
inject_partition = -2	整数値	データを挿入するためにファイルシステムを選択する方法を決定します。 **libguestfs** は、データの挿入に使用されます。libguestfs がルートパーティションを特定できない場合 (ルートパーティションが複数または少ないため)、またはファイルシステムをマウントできない場合、エラーが発生し、インスタンスは起動しません。 値: ● -2 ⇒ データの挿入を無効にします。 ● -1 ⇒ libguestfs でマウントするファイルシステムでルートパーティションを見つけます ● 0 ⇒ イメージが分割されていません ● >0 ⇒ 注入に使用するパーティションの数 **Linux** ディストリビューションゲストのみ。 関連するオプション: ● **inject_key**: このオプションが SSH キーの挿入を許可する場合、**inject_partition** の -1 以上の値に依存します。 ● **inject_password**: このオプションが管理者パスワードの挿入を許可する場合は、**inject_partition** の -1 以上の値に依存します。 ● **[guestfs]/debug** この設定オプションで libguestfs のデバッグログレベルを有効にできます。より詳細な出力は、問題のデバッグに役立ちます。 ● **virt_type**: **lxc** を virt_type として使用する場合、単一のパーティションイメージとして処理されます。

設定オプション = デフォルト値	型	説明
inject_password = False	ブール値	**create** および **rebuild** プロセス時のみなどに、管理者パスワードの挿入を許可します。 これを実行するために、イメージ内にエージェントは必要ありません。ホストで **libguestfs** が利用可能な場合は、それが使用されます。それ以外の場合は **nbd** が使用されます。イメージのファイルシステムがマウントされ、管理者パスワードが REST API 呼び出しに提供されたものが root ユーザーのパスワードとしてインジェクトされます。利用可能な root ユーザーがない場合、インスタンスは起動せず、エラーが発生します。インスタンスがボリュームから起動すると、インジェクションができ **ない** ことに注意してください。 Linux ディストリビューションゲストのみ。 値: ● インジェクションを許可します。 ● インジェクションを無効にします。REST API が提供する管理者パスワード経由はすべて警告なしで無視されます。 関連するオプション: ● **inject_partition**: このオプションは、ファイルシステムの検出および使用方法について決定します。また、インジェクションを無効にすることもできます。
iscsi_iface = None	文字列値	オフロードサポートが必要な場合に、ターゲットへの接続に使用する iSCSI トランスポートが必要です。 デフォルトの形式は **<transport_name>.<hwaddress>** です。**<transport_name>** は (**be2iscsi**、**bnx2i**、**cxgb3i**、**cxgb4i**、**qla4xxx**、**ocs**、**tcp**) のいずれかであり、**<hwaddress>** はインターフェイスの MAC アドレスであり、**iscsiadm -m iface** コマンドで生成されます。ここで指定する **iscsi_iface** パラメーターと実際のトランスポート名を混同しないでください。
iser_use_multipath = False	ブール値	iSER ボリュームのマルチパス接続を使用します。 iSER ボリュームはマルチパスデバイスとして接続できます。これにより、高可用性とフォルトトレランスが提供されます。

設定オプション = デフォルト値	型	説明
live_migration_bandwidth = 0	整数値	移行時に使用する最大帯域幅 (MiB/秒)。 0 に設定すると、ハイパーバイザーは適切なデフォルトを選択します。一部のハイパーバイザーはこの機能をサポートしておらず、帯域幅が 0 ではない場合はエラーを返します。詳細は libvirt のドキュメントを参照してください。
live_migration_completion_timeout = 800	整数値	操作を中止する前に、移行がデータの転送を正常に完了するまで待機する時間 (秒単位)。 値は、転送されるゲスト RAM あたり 2 GiB あたりで、最小 2 GiB の境界が低くなっています。通常、ダウンタイムの遅延 * ダウンタイムステップよりも大きくする必要があります。タイムアウトを無効にするには 0 に設定します。 関連するオプション: ● live_migration_downtime ● live_migration_downtime_steps ● live_migration_downtime_delay
live_migration_downtime = 500	整数値	ライブマイグレーションスイッチオーバーの最大許容ダウンタイム (ミリ秒単位)。 最小 100 ミリ秒まで丸められます。ライブマイグレーションをより迅速に完了させたい場合や、ライブマイグレーション間の切り替え中にゲストを一時停止できるようにすることで、ライブマイグレーションのタイムアウトエラーを回避したい場合は、この値を大きくすることができます。 関連するオプション: ● live_migration_completion_timeout
live_migration_downtime_delay = 75	整数値	移行のダウンタイムが各ステップで増加するまでの待機時間 (秒単位)。 最小遅延は 3 秒です。値は、転送されるゲスト RAM あたり 2 GiB あたりで、デバイスあたり最低 2 GiB のバインドが低くなっています。
live_migration_downtime_steps = 10	整数値	最大ダウンタイム値に達する増分ステップの数。 少なくとも 3 つのステップに丸められます。

設定オプション = デフォルト値	型	説明
live_migration_inbound_addr = None	ホストのアドレス値	このホストのライブマイグレーションアドレスとして使用される IP アドレス。 このオプションは、このハイパーバイザーに移行するときにライブマイグレーショントラフィックのターゲットとして使用する IP アドレスを示します。このメタデータは、ライブマイグレーショントラフィックのソースによって使用され、マイグレーション URI が構築されます。 このオプションが None に設定されている場合、移行先のコンピュートノードのホスト名が使用されます。 このオプションは、ライブマイグレーションのトラフィックがネットワークプレーンに大きく影響する環境で役に立ちます。ライブマイグレーショントラフィックに別のネットワークを使用すると、この設定オプションを使用して管理ネットワークに影響が及ばないようにすることができます。
live_migration_permit_auto_converge = False	ブール値	このオプションにより、nova は自動収束を使用してライブマイグレーションを開始できます。 ライブマイグレーションの進行が遅い場合に、自動収束が CPU ダウンします。自動収束は、このフラグが True に設定されている場合にのみ使用されます。また、使用中の libvirt および QEMU のバージョンにより、ポストコピーが許可またはポストコピーが利用できない場合にのみ使用されます。 関連するオプション: ● live_migration_permit_post_copy

設定オプション = デフォルト値	型	説明
live_migration_permit_post_copy = False	ブール値	このオプションにより、nova は、移行が完了する必要のあるメモリーの上限が上限になるように、ライブマイグレーションをポストコピーモードに切り替えることができます。Post-copy には libvirt>=1.3.3 および QEMU>=2.5.0 が必要です。 **live_migration_completion_timeout** で定義したタイムアウトに到達し、**live_migration_timeout_action** が **force_complete** に設定されていると、コピー後のモードは自動的にアクティベートされます。タイムアウトなし、または**中止を**選択した場合 (**live_migration_completion_timeout = 0**)、ポストコピーへの自動切り替えが行われないことに注意してください。 ライブマイグレーションの強制完全な API は、許可された場合にコピー後にも使用します。post-copy モードが利用できない場合は、仮想マシンが一時停止するように強制し、ライブマイグレーションの操作が完了するようにします。 post-copy モードを使用する場合、移行元ホストと移行先ホストがネットワーク接続を失った場合は、ライブマイグレーションされている仮想マシンを再起動する必要があります。詳細は、管理ガイドを参照してください。 関連するオプション: ● live_migration_permit_auto_converge ● live_migration_timeout_action

設定オプション = デフォルト値	型	説明
live_migration_scheme = None	文字列値	ライブマイグレーショントラフィックのソースによって使用されるライブマイグレーションの URI スキーム。 デフォルトの libvirt ライブマイグレーションスキームをオーバーライドします (virt_type に依存します)。このオプションが None に設定されている場合、nova はハイパーバイザーに基づいて適切なデフォルトを自動的に選択します。ハイパーバイザーが特定のスキームをサポートしていることが分からない場合は、この変更は推奨されません。 関連するオプション: ● **virt_type**: このオプションは、**virt_type** が **kvm** または **qemu** に設定されている場合にのみ意味があります。 ● **live_migration_uri**: **live_migration_uri** 値が None でない場合、ライブマイグレーションに使用されるスキームは代わりに **live_migration_uri** から取得されます。
live_migration_timeout_action = abort	文字列値	このオプションは、**live_migration_completion_timeout** の期限が切れた後に仮想マシンに対して実行するアクションを決定するのに使用します。デフォルトでは、ライブマイグレーション操作は完了タイムアウト後に中止されます。**force_complete** に設定された場合、ポストコピーが有効で利用可能かどうかによって、コンピュートサービスは VM を一時停止するか、ポストコピーを起動します (**live_migration_permit_post_copy** が True に設定されている場合)。 関連するオプション: ● live_migration_completion_timeout ● live_migration_permit_post_copy

設定オプション = デフォルト値	型	説明
live_migration_tunnelled = False	ブール値	トンネル化された移行を有効にします。 このオプションにより、移行データが libvirtd 接続で転送されるトンネル化された移行機能が有効になります。有効にすると、VIR_MIGRATE_TUNNELLED 移行フラグを使用し、ハイパーバイザーからハイパーバイザーへの直接通信を許可するようにネットワークを設定する必要がなくなります。False の場合は、ネイティブトランスポートを使用します。設定されていない場合、Nova はハイパーバイザーでのネイティブ暗号化サポートの可用性など、適切なデフォルトをベースとして選択します。このオプションを有効にすると、パフォーマンスに大きく影響します。 このオプションは、ブロックマイグレーションの使用と互換性がありません。 **非推奨:** 23.0.0 **理由:** トンネル化されたライブマイグレーションには固有の制限が 2 つあります。そして、それには莫大なパフォーマンスコストがかかります。これらの問題は両方とも ``live_migration_with_native_tls`` (事前に設定された TLS 環境が必要です) によって解決されます。これは、すべてのライブマイグレーションストリームを保護するための推奨されるアプローチです。

設定オプション = デフォルト値	型	説明
live_migration_uri = None	文字列値	ライブマイグレーショントラフィックのソースによって使用されるライブマイグレーションターゲット URI。 デフォルトの libvirt ライブマイグレーションターゲット URL をオーバーライドします (virt_type に依存します)。含まれる %s は、移行ターゲットのホスト名、または設定されている場合は **live_migration_inbound_addr** に置き換えられます。 このオプションを None に設定すると (これがデフォルトです)、Nova は以下のリストにある 4 つのサポートされる **virt_type** のみに基づいて **live_migration_uri** 値を自動的に生成します。 ● **kvm**: qemu+tcp://%s/system ● **qemu**: qemu+tcp://%s/system ● **parallels**: parallels+tcp://%s/system 関連するオプション: ● **live_migration_inbound_addr**: **live_migration_inbound_addr の** 値が None ではなく、**live_migration_tunnelled** が False の場合、ライブマイグレーションの uri として **live_migration_uri** の代わりにターゲットコンピュートノードの IP/ホスト名アドレスが使用されます。 ● **live_migration_scheme**: **live_migration_uri** が設定されていない場合、ライブマイグレーションに使用されるスキームは代わりに **live_migration_scheme** から取得されます。 **非推奨となったバージョン:** 15.0.0 **理由:** live_migration_uri は非推奨で、ライブマイグレーションスキームとターゲット URI を変更できる他の 2 つのオプション、``live_migration_scheme`` と ``live_migration_inbound_addr`` に変更されました。

設定オプション = デフォルト値	型	説明
live_migration_with_native_tls = False	ブール値	ライブマイグレーション時に QEMU ネイティブ TLS 暗号化を使用します。 このオプションにより、マイグレーションストリーム (ゲスト RAM とデバイスの状態) と ディスクストリームの両方がネイティブ TLS (QEMU に組み込まれた TLS サポート) で転送されるようになります。 前提条件: 関連するすべてのコンピュートノードで TLS 環境が正しく設定されている。つまり、認証局 (CA)、サーバー、クライアント証明書、それらに対応する鍵、およびそれらのファイルパーミッションが適切に配置され、検証されていることを意味する。 注記: ● マイグレーションストリームとディスクストリームを暗号化するには、**live_migration_tunnelled** の代わりに **live_migration_with_native_tls** が望ましい設定属性です。 ● **live_migration_tunnelled** は、主要な 2 つの理由により長期で非推奨になりました。(a) は、パフォーマンスが大幅に低下し、ブロックマイグレーションと互換性がない (b) です。したがって、コンピュートノードに libvirt 4.4.0 および QEMU 2.11.0 以上がある場合は、**live_migration_with_native_tls** を使用することを強く推奨します。 ● **live_migration_tunnelled** および **live_migration_with_native_tls** は同時に使用しないでください。 ● **live_migration_tunnelled** とは異なり、**live_migration_with_native_tls** はブロック移行と互換性があります。つまり、このオプションを使用すると、ディスクがターゲットホストに移行される NBD ストリームが暗号化されます。 関連するオプション: **live_migration_tunnelled**: libvirtd 経由で移行ストリーム (ディスクストリームではない) を転送します。

設定オプション = デフォルト値	型	説明
max_queues = None	整数値	マルチキューゲストの作成時に有効にできる virtio キューペアの最大数。割り当てられる virtio キューの数は、ゲストによって要求される CPU と定義される最大値のどちらか小さい方です。デフォルトでは、この値は none に設定されています。つまり、報告されたカーネルのメジャーバージョンに基づくレガシー制限が使用されます。
mem_stats_period_seconds = 10	整数値	メモリー使用量の統計期間の秒数。ゼロまたは負の値を指定すると、メモリー使用率の統計が無効になります。
nfs_mount_options = None	文字列値	NFS クライアントに渡されるマウントオプション。詳細は、nfs の man ページの セクションを参照してください。 マウントオプションは、ファイルシステムのマウント方法と、このマウントポイント上のファイルへのアクセス時に NFS クライアントがどのように動作するかを制御します。 値: ● コンマで区切られたマウントオプションを表す文字列。 ● 例の文字列: vers=3,lookupcache=pos
nfs_mount_point_base = $state_path/mnt	文字列値	NFS ボリュームがコンピュートノードにマウントされているディレクトリー。デフォルトは、nova の Python モジュールがインストールされている場所の **mnt** ディレクトリーです。 NFS は、OpenStack Block Storage サービスの共有ストレージを提供します。 値: ● マウントポイントの絶対パスを表す文字列。
num_aoe_discover_tries = 3	整数値	AoE ターゲットを再検出してボリュームを見つける回数。 Nova は、AOE (ATA over Ethernet) 経由でホストに接続するブロックストレージをサポートしています。このオプションを使用すると、ユーザーは AoE デバイスを検出できる再試行の最大試行回数を指定できます。

設定オプション = デフォルト値	型	説明
num_iser_scan_tries = 5	整数値	ボリュームを見つけるために iSER ターゲットをスキャンする回数。 iSER は、RDMA (Remote Direct Memory Access) を使用するように iSCSI プロトコルを拡張するサーバーネットワークプロトコルです。このオプションを使用すると、ユーザーは iSER ボリュームを検索することができるスキャン試行の最大数を指定できます。

設定オプション = デフォルト値	型	説明
num_memory_encrypted_guests = None	整数値	このコンピュートホストで同時に実行できる、暗号化されたメモリーを持つゲストの最大数。 現在、これは SEV (Secure Encrypted Virtualization) をサポートする AMD マシンにのみ関連します。このようなマシンには、暗号鍵を格納するためのメモリーコントローラー内のスロット数が限定されています。暗号化したメモリーを持つ実行中のゲストは、これらのスロットの 1 つを消費します。 このオプションは、今後、他の同等の技術で再利用できます。マシンがメモリーの暗号化に対応していない場合、オプションは無視され、インベントリーが 0 に設定されます。 マシンがメモリーの暗号化に対応している場合、**現在** は **None** という値は、実質的に無制限のインベントリーを意味します。つまり、基盤のハードウェアが独自の制限を実施する場合でも、Nova が起動できる SEV ゲストの数に制限はありません。ただし、今後は、ハードウェアからのインベントリーの自動検出が可能となることが予想され、その時点で **None** の場合、自動検出が自動的に正しい制限を課すことになります。 a. 注記: > It is recommended to read :ref:`the deployment documentation's section on this option <num_memory_encrypted_guests>` before deciding whether to configure this setting or leave it at the default. 関連するオプション: - :oslo.config:option:**libvirt.virt_type** を **kvm** に設定する必要があります。 - :oslo.config:option:**libvirt.hw_machine_type** に **x86_64=q35** を含めることが推奨されます。詳細は、:ref:**deploying-sev-capable-infrastructure** を参照してください。

設定オプション = デフォルト値	型	説明
num_nvme_discover_tries = 5	整数値	ボリュームを見つけるために NVMe ターゲットを再検出する回数 Nova は、NVMe (Non-Volatile Memory Express) 経由でホストにアタッチするブロックストレージをサポートします。このオプションを使用すると、ユーザーは NVMe デバイスを検出できる再試行の最大数を指定できます。
num_pcie_ports = 0	整数値	インスタンスが取得する PCIe ポートの数。 libvirt では、対象のインスタンスが取得する PCIe ポートの数をカスタマイズすることができます (pcie-root-port コントローラー)。一部はデフォルトで使用されますが、残りはホットプラグの使用に利用されます。 デフォルトでは、ホットプラグを制限する空きポートが 1 - 2 だけあります。 詳細: https://github.com/qemu/qemu/blob/master/docs/pcie.txt aarch64/virt の最大値の QEMU の制限により、28 に設定されます。 デフォルト値 0 は、libvirt へのポート数の計算を行います。
num_volume_scan_tries = 5	整数値	ボリュームを検索するために特定のストレージプロトコルをスキャンする回数。

設定オプション = デフォルト値	型	説明
pmem_namespaces = []	リスト値	永続メモリー (pmem) 名前空間を設定します。 これらの名前空間は、すでにホスト上に作成されている必要があります。この設定オプションは、以下の形式になります。 "$LABEL:$NSNAME[|$NSNAME][,$LABEL:$NSNAME[|$NSNAME]]" - **$NSNAME** は pmem 名前空間の名前です。 - **$LABEL** は 1 つのリソースクラスを表し、これはリソースクラス名を **CUSTOM_PMEM_NAMESPACE_$LABEL** として生成するために使用されます。 **以下に例を示します。** [libvirt] pmem_namespaces=128G:ns0|ns1|ns2|ns3,262144MB:ns4|ns5,MEDIUM:ns6|ns7
quobyte_client_cfg = None	文字列値	Quobyte クライアント設定ファイルへのパス。
quobyte_mount_point_base = $state_path/mnt	文字列値	Quobyte ボリュームがコンピュートノードにマウントされているディレクトリー。 Nova は Quobyte ストレージバックエンドに Block Storage サービスボリュームの格納を可能にする Quobyte ボリュームドライバーをサポートします。このオプションは、Quobyte ボリュームをマウントするディレクトリーのパスを指定します。 値: - マウントポイントの絶対パスを表す文字列。
rbd_connect_timeout = 5	整数値	最初にクラスターに接続するときの RADOS クライアントのタイムアウト (秒単位)。
rbd_destroy_volume_retries = 12	整数値	RBD ボリュームを破棄するための再試行回数。 関連するオプション: - [libvirt]/images_type = **rbd**

設定オプション = デフォルト値	型	説明
rbd_destroy_volume_retry_interval = 5	整数値	RBD ボリュームを破棄するための連続した各再試行の間に待機する秒数。 関連するオプション: ● [libvirt]/images_type = **rbd**
rbd_secret_uuid = None	文字列値	rbd_user ボリュームのシークレットの libvirt UUID。
rbd_user = None	文字列値	rbd(RADOS Block Devices) ボリュームにアクセスするための RADOS クライアント名。 libvirt は、Ceph RBD サーバーへの接続および認証時にこのユーザーを参照します。
realtime_scheduler_priority = 1	整数値	リアルタイムホストコンテキストの仮想 CPU は、そのスケジューリングの優先度で実行されます。優先順位は、ホストカーネル (通常は 1-99) に依存します。
remote_filesystem_transport = ssh	文字列値	libvirt のリモートファイル操作のためのトランスポートメソッド。 libvirt は、ネットワークを介して他のコンピュートノードにファイルをコピーするために RPC を使用できないため、以下の場合に他の方法を使用する必要があります。 ● リモートホストでのディレクトリーの作成 ● リモートホストでのファイルの作成 ● リモートホストからのファイルの削除 ● リモートホストへのファイルのコピー

設定オプション = デフォルト値	型	説明
rescue_image_id = None	文字列値	破損したインスタンスからレスキューデータにブートするイメージの ID。 レスキュー REST API オペレーションが使用するイメージの ID を提供しない場合、この ID によって参照されるイメージが使用されます。このオプションが設定されていない場合には、インスタンスのイメージが使用されます。 値: ● イメージの ID、または何もしない。**Amazon Machine Image** (AMI) を参照する場合は、設定オプション **rescue_kernel_id** と **rescue_ramdisk_id** も設定することを検討してください。何も設定しないと、インスタンスのイメージが使用されます。 関連するオプション: ● **rescue_kernel_id**: 選択したレスキューイメージでカーネルディスクを個別に定義できる場合、このオプションの値が指定されていれば、それが使用されます。これは、**Amazon** の AMI/AKI/ARI イメージフォーマットがレスキューイメージに使用される場合です。 ● **rescue_ramdisk_id**: 選択したレスキューイメージで RAM ディスクを個別に定義できる場合、このオプションの値が指定されている場合に使用されます。これは、**Amazon** の AMI/AKI/ARI イメージフォーマットがレスキューイメージに使用される場合です。

設定オプション = デフォルト値	型	説明
rescue_kernel_id = None	文字列値	レスキューイメージで使用するカーネル (AKI) イメージの ID。 選択したレスキューイメージがカーネルディスクの個別の定義を許可する場合は、指定した場合はこのオプションの値が使用されます。これは、**Amazon** の AMI/AKI/ARI イメージフォーマットがレスキューイメージに使用される場合です。 値: ● カーネルイメージの ID または何もしない。何も指定しないと、インスタンスのカーネルディスクが使用されます (いずれか 1 つで起動した場合)。 関連するオプション: ● **rescue_image_id**: そのオプションが **Amazon** の AMI/AKI/ARI イメージ形式のイメージを指している場合は、**rescue_kernel_id** も使用すると便利です。
rescue_ramdisk_id = None	文字列値	レスキューイメージで使用する RAM ディスク (ARI) イメージの ID。 選択したレスキューイメージが RAM ディスクの個別の定義を許可する場合は、指定した場合はこのオプションの値が使用されます。これは、**Amazon** の AMI/AKI/ARI イメージフォーマットがレスキューイメージに使用される場合です。 値: ● RAM ディスクイメージの ID。何も指定しないと、インスタンスの RAM ディスクは、インスタンスの RAM ディスクを使用して起動されます。 関連するオプション: ● **rescue_image_id**: そのオプションが **Amazon** の AMI/AKI/ARI イメージ形式のイメージを指している場合は、**rescue_ramdisk_id** も使用すると便利です。

設定オプション = デフォルト値	型	説明
rng_dev_path = /dev/urandom	文字列値	ホストのエントロピーのソースとして使用される RNG (乱数ジェネレーター) デバイスへのパス。libvirt 1.3.4 以降、任意のパス (読み取りの場合は乱数を返す) が受け入れられます。推奨されるエントロピーのソースは **/dev/urandom** MYBACKUPDIR-ipmilanit はノンブロッキングであるため、比較的高速で、**/dev/random** (レガシーインターフェイス) の制限を回避しています。詳細 (および異なる RNG ソース間の比較) については、Linux カーネル API ドキュメントの **[u]random**: http://man7.org/linux/man-pages/man4/urandom.4.html および http://man7.org/linux/man-pages/man7/random.7.html の項を参照してください。
rx_queue_size = None	整数値	virtio rx キューサイズを設定します。 このオプションは、vhost および vhost-user バックエンドを使用する virtio-net デバイスでのみ利用できます。QEMU/KVM でのみ利用できます。libvirt v2.3 QEMU v2.7 が必要です。
`smbfs_mount_options = `	文字列値	SMBFS クライアントに渡されるマウントオプション。 SMBFS オプションを、すべてのパラメーターを含む単一の文字列として提供します。詳細は、man ページの mount.cifs を参照してください。libvirt-qemu **uid** および **gid** を指定する必要があることに注意してください。
smbfs_mount_point_base = $state_path/mnt	文字列値	SMBFS 共有が、コンピュートノードにマウントされているディレクトリー。
snapshot_compression = False	ブール値	**qcow2** イメージのスナップショット圧縮を有効にします。 注記: **snapshot_image_format** を **qcow2** に設定すると、元のイメージタイプとは別に、すべてのスナップショットが **qcow2** 形式で強制的に設定できます。 関連するオプション: • snapshot_image_format

設定オプション = デフォルト値	型	説明
snapshot_image_format = None	文字列値	イメージサービスに送信する際に、スナップショットイメージの形式を決定します。 これが設定されている場合は、スナップショットをイメージサービスに送信する際に使用される形式を決定します。設定されていない場合、デフォルトはソースイメージと同じタイプになります。
snapshots_directory = $instances_path/snapshots	文字列値	libvirt ドライバーがイメージサービスにアップロードする前にスナップショットを保存する場所
sparse_logical_volumes = False	ブール値	このフラグが True に設定されている場合、スパース論理ボリューム (virtualsize を使用) を作成します。 **非推奨となったバージョン:** 18.0.0 **理由:** スパース論理ボリュームは、テストされていないのでサポートされていない機能です。LVM 論理ボリュームは、デフォルトで事前に割り当てられます。シンプロビジョニングが必要な場合は、Cinder のシンプロビジョニングボリュームを使用します。
swtpm_enabled = False	ブール値	ゲストでエミュレーションされた TPM (Trusted Platform Module) を有効にします。
swtpm_group = tss	文字列値	swtpm バイナリーが実行されるグループ。 エミュレーションされた TPM を使用すると、**swtpm** バイナリーが実行されて TPM デバイスがエミュレーションされます。このバイナリーが実行されるユーザーは libvirt の設定に依存し、デフォルトは **tss** です。 コールドマイグレーションとサイズ変更をサポートするために、nova は、swtpm バイナリーが実行されているグループを認識して、ノード間で移動された後にファイルが適切な所有権を取得できるようにする必要があります。 関連するオプション: ● **swtpm_user** も設定する必要があります。

設定オプション = デフォルト値	型	説明
swtpm_user = tss	文字列値	swtpm バイナリーを実行するユーザー。 エミュレーションされた TPM を使用すると、**swtpm** バイナリーが実行されて TPM デバイスがエミュレーションされます。このバイナリーが実行されるユーザーは libvirt の設定に依存し、デフォルトは **tss** です。 コールドマイグレーションとサイズ変更をサポートするために、nova は、swtpm バイナリーが実行されているユーザーを認識して、ノード間で移動された後にファイルが適切な所有権を取得できるようにする必要があります。 関連するオプション: ● **swtpm_group** も設定する必要があります。
sysinfo_serial = unique	文字列値	仮想 BIOS でゲストに公開されるホスト serial UUID の設定に使用するデータソース。**unique** 以外の選択肢はすべて、インスタンスを別のホストに移行する際にシリアルを変更します。このオプションの選択を変更すると、停止して再起動されると、このホストの既存のインスタンスにも影響します。インスタンスの移行時に変更されないため、デフォルトの選択 (**unique**) を使用することが推奨されます。ただし、インスタンスごとのシリアル番号に加えてホストごとのシリアル番号が必要な場合は、ホストアグリゲートを使用してフレーバーを制限することを検討してください。
tx_queue_size = None	整数値	virtio tx キューサイズを設定します。 このオプションは、vhost-user バックエンドを使用する virtio-net デバイスでのみ利用できます。QEMU/KVM でのみ利用できます。libvirt v3.7 QEMU v2.10 が必要です。
uid_maps = []	リスト値	uid ターゲットと範囲のリスト。Syntax は guest-uid:host-uid:count です。最大 5 つまで許容されます。
use_virtio_for_bridges = True	ブール値	KVM/QEMU のブリッジインターフェイスに virtio を使用

設定オプション = デフォルト値	型	説明
virt_type = kvm	文字列値	libvirt が使用する仮想化タイプ (またはドメインタイプと呼ばれるもの) を記述します。 このタイプの選択は、このホストに選択した基礎となる仮想化ストラテジーと一致する必要があります。 関連するオプション: ● **connection_uri**: これに依存します。 ● **disk_prefix**: これにより異なります。 ● **cpu_mode**: これによって異なります。 ● **cpu_models**: これによって異なります。
volume_clear = zero	文字列値	一時ディスクの削除時に、一時ディスクを消去する方法。LVM がバッキングストレージとして設定されている場合に限り有効です。 関連するオプション: ● images_type - **lvm**に設定する必要があります。 ● volume_clear_size
volume_clear_size = 0	整数値	**volume_clear** オプションで設定した方法でクリアされる、割り当てられたボリュームの開始からカウントされるエリアのサイズ (MiB 単位)。 値: ● 0 - ボリューム全体を消去 ● >0 - 明確に指定した MiB の量です。 関連するオプション: ● images_type - **lvm**に設定する必要があります。 ● volume_clear - を設定する必要があります。この値は、このオプションに影響を与えるために **none** とは異なる必要があります。

設定オプション = デフォルト値	型	説明
volume_use_multipath = False	ブール値	iSCSI または FC ボリュームのマルチパス接続を使用します。 ボリュームは、マルチパスデバイスとして LibVirt で接続できます。これにより、高可用性とフォルトトレランスが提供されます。
vzstorage_cache_path = None	文字列値	SSD キャッシュファイルへのパス。 SSD ドライブをクライアントに割り当て、ドライブを設定して頻繁にアクセスされるデータのローカルキャッシュを保存できます。クライアントの SSD ドライブにローカルキャッシュがあると、クラスター全体のパフォーマンスを最大 10 以上に増やすことができます。WARNING!サーバーグレードにはない SSD モデルが多数あり、電源損失時に任意のデータセットが遅れる可能性があります。このような SSD は、Vstorage で使用しないでください。データの破損や不整合が発生する可能性があるため、危険です。安全であることが分かっている SSD モデルを手動で参照するか、vstorage-hwflush-check(1) ユーティリティーを使用して検証してください。 このオプションは、複数の共有からキャッシュを分離するために "%(cluster_name)s" テンプレートを含むパスを定義します。 関連するオプション: ● vzstorage_mount_opts には、より詳細なキャッシュオプションが含まれる場合があります。
vzstorage_log_path = /var/log/vstorage/%(cluster_name)s/nova.log.gz	文字列値	vzstorage クライアントログへのパス。 このオプションは、クラスター操作のログを定義します。これには、複数の共有からのログを分離するために "%(cluster_name)s" テンプレートを含める必要があります。 関連するオプション: ● vzstorage_mount_opts には、より詳細なロギングオプションが含まれる場合があります。

設定オプション = デフォルト値	型	説明
vzstorage_mount_group = qemu	文字列値	所有者グループ名をマウントします。 このオプションは、Vzstorage クラスターのマウントポイントの所有者グループを定義します。 関連するオプション: ● vzstorage_mount_* group of parameters
vzstorage_mount_opts = []	リスト値	pstorage-mount の追加マウントオプション これらの詳細な説明は、https://static.openvz.org/vz-man/man1/pstorage-mount.1.gz.html を参照してください。形式は、"[**-v**, **-R**, **500**]" などの引数リストを示す python 文字列です。-c、-l、-C、-u、-g、-m などの引数リストの Python 文字列表現です。これらのオプションは、明示的な vzstorage_* オプションであるためです。 関連するオプション: ● 他のすべての vzstorage_* オプション
vzstorage_mount_perms = 0770	文字列値	マウントアクセスモード。 このオプションは、Vzstorage クラスターマウントポイントのアクセスビットを、chmod(1) ユーティリティーの 1 つと同様の形式 (0770 など) で定義します。0 から 7 までの 1 桁から 4 桁で構成され、先頭の桁が欠落している場合は 0 と見なされます。 関連するオプション: ● vzstorage_mount_* group of parameters
vzstorage_mount_point_base = $state_path/mnt	文字列値	Virtuozo Storage クラスターがコンピュートノードにマウントされているディレクトリー。 このオプションは、Vzstorage クラスターの標準以外のマウントポイントを定義します。 関連するオプション: ● vzstorage_mount_* group of parameters

設定オプション = デフォルト値	型	説明
vzstorage_mount_user = stack	文字列値	所有者のユーザー名をマウントします。 このオプションは、Vzstorage クラスターのマウントポイントの所有者ユーザーを定義します。 関連するオプション: ● vzstorage_mount_* group of parameters
wait_soft_reboot_seconds = 120	整数値	ソフトリブート要求後にインスタンスがシャットダウンするのを待機する秒数。インスタンスがこのウィンドウ内でシャットダウンしない場合は、ハードリブートにフォールバックします。

9.1.27. metrics

以下の表は、**/etc/nova/nova.conf** ファイルの **[metrics]** グループで使用できるオプションの概要をまとめたものです。

表9.26 metrics

設定オプション = デフォルト値	型	説明
required = True	ブール値	指標が必要かどうか。 この設定は、使用できないメトリックの処理方法を決定します。このオプションを True に設定すると、メトリックが利用できない場合に例外が発生するため、メトリックフィルターを使用してホストを重み付けする前に使用することが推奨されます。 値: ● ブール値。False の場合、ホストで使用できないメトリックはホストの重みを **metrics weight_of_unavailable** に設定します。 関連するオプション: ● **[metrics] weight_of_unavailable**

設定オプション = デフォルト値	型	説明
weight_multiplier = 1.0	浮動小数点の値	報告された指標に基づいてホストを重み付けするために使用される乗数。 メトリックを使用してホストの適合性を重み付けするには、このオプションを使用して、以下のように、計算された重みがホストに割り当てられた重みにどのように影響するかを変更することができます。 ● **>1.0**: 全体の重みに対するメトリックの影響を増加します。 ● **1.0**: 計算された重みに変更なし。 ● **>0.0,<1.0**: 全体の重みに対するメトリックの影響を減少します。 ● **0.0**: メトリック値は無視され、代わりに **[metrics] weight_of_unavailable** オプションの値が返されます。 ● **>-1.0,<0.0**: 効果が減少し、逆になります。 ● **-1.0**: 効果が反転します。 ● **<-1.0**: 効果が比例して増加し、逆になります。 値: ● 整数または浮動小数点の値。ここで、値はこの重み付け関数の乗数比率に対応します。 関連するオプション: ● **[filter_scheduler] weight_classes** ● **[metrics] weight_of_unavailable**

設定オプション = デフォルト値	型	説明
weight_of_unavailable = -10000.0	浮動小数点の値	使用できないメトリックのデフォルトの重み。 以下の条件が満たされると、実際のメトリック値の代わりにこの値が使用されます。 ● **[metrics] weight_setting** で指定されたメトリックの 1 つがホストで使用できず、**required** の値が **False** です。 ● **[metrics] weight_setting** でメトリックに指定された比率が 0 です。 ● **[metrics] weight_multiplier** オプションは 0 に設定されています。 値: ● 整数または浮動小数点の値。ここで、値はこの重み付け関数の乗数比率に対応します。 関連するオプション: ● **[metrics] weight_setting** ● **[metrics] required** ● **[metrics] weight_multiplier**

設定オプション = デフォルト値	型	説明

weight_setting = []	リスト値	重量修飾子へのメトリックのマッピング。
		
		この設定は、重み付けするメトリックと各メトリックの相対比率を指定します。これは，コンマで区切られた一連の 1 つ以上の**name=ratio** の組からなる単一の文字列値であるべきで、ここで **name** は計量されるメートル法の名前、**ratio** はそのメートル法に対する相対的な重量です。
		
		比率が 0 に設定されている場合、メトリック値は無視され、代わりに重みが **[metrics] weight_of_unavailable** オプションの値に設定されることに注意してください。
		
		たとえば、このオプションが設定されているケースを見てみましょう。
		
		`name1=1.0, name2=-1.3`
		
		最終的な重みは以下のようになります。
		
		`(name1.value * 1.0) + (name2.value * -1.3)`
		
		値:
		
		● コンマで区切られた 0 個以上のキー/値のペアのリストです。キーは、メトリックの名前を表す文字列で、値はそのメトリックの数値の重みになります。いずれかの値が 0 に設定されている場合、その値は無視され、重みは **[metrics] weight_of_unavailable** オプションの値に設定されます。
		
		関連するオプション:
		
		● **[metrics] weight_of_unavailable**

9.1.28. mks

以下の表は、**/etc/nova/nova.conf** ファイルの **[mks]** グループで使用できるオプションの概要をまとめたものです。

表9.27 mks

設定オプション = デフォルト値	型	説明
enabled = False	ブール値	仮想マシンのグラフィカルコンソールアクセスを有効にします。

設定オプション = デフォルト値	型	説明
mksproxy_base_url = http://127.0.0.1:6090/	URI 値	MKS Web コンソールプロキシーの場所 応答の URL は、WebMKS プロキシーをポイントします。これは、クライアントと、インスタンスが実行される対応する vCenter サーバーとの間のプロキシーを開始します。Web ベースのコンソールアクセスを使用するには、WebMKS プロキシーをインストールし、設定する必要があります。 値: ● **http://host:port/** または **https://host:port/**形式の有効な URL でなければなりません。

9.1.29. neutron

以下の表は、**/etc/nova/nova.conf** ファイルの **[neutron]** グループで使用できるオプションの概要をまとめたものです。

表9.28 neutron

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。

設定オプション = デフォルト値	型	説明
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default_floating_pool = nova	文字列値	Floating IP プールのデフォルト名。 Floating IP アドレスの割り当てに使用する Floating IP プールの名前を指定します。このオプションは、ポートバインディング応答で Neutron が Floating IP プール名を指定しない場合にのみ使用されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
extension_sync_interval = 600	整数値	Neutron に拡張機能を問い合わせるまで待機する秒数を表す整数値。この秒数の後、次に Nova が Neutron でリソースを作成する必要がある場合、Neutron にロードした拡張機能を要求します。値を 0 に設定すると、待機なしで拡張機能が更新されます。

設定オプション = デフォルト値	型	説明
http_retries = 3	整数値	失敗した http 呼び出しで neutronclient が再試行される回数。 0 は、接続が 1 回のみ試行されることを意味します。正の整数に設定すると、接続に失敗すると何度も再試行されます。たとえば、これを 3 に設定すると、接続合計の試みは 4 になります。 値: ● 整数値。0 は、接続が 1 回のみ試行されることを意味します。
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
`metadata_proxy_shared_secret = `	文字列値	このオプションは、Neutron メタデータ要求に対するプロキシー要求の検証に使用される共有シークレット文字列を保持します。使用するには、**X-Metadata-Provider-Signature** ヘッダーをリクエストに指定する必要があります。 関連するオプション: ● service_metadata_proxy
ovs_bridge = br-int	文字列値	Open vSwitch 統合ブリッジのデフォルト名。 OpenvSwitch が使用する統合ブリッジインターフェイスの名前を指定します。このオプションは、Neutron がポートバインディング応答で OVS ブリッジ名を指定しない場合にのみ使用されます。
password = None	文字列値	ユーザーのパスワード

設定オプション = デフォルト値	型	説明
physnets = []	リスト値	このホストに存在する物理ネットワークのリスト。 リストされた各 physnet 似対して、**[neutron_physnet_$PHYSNET]** という追加のセクションが設定ファイルに追加されます。各セクションには、単一の設定オプション **numa_nodes** を指定する必要があります。これは、この物理ネットワークに関連付けられたすべての NUMA ノードに対するノード ID のリストでなければなりません。以下に例を示します。 [neutron] physnets = foo, bar [neutron_physnet_foo] numa_nodes = 0 [neutron_physnet_bar] numa_nodes = 0,1 このオプションを使用して記載されていない **物理** ネットワークは、特定の NUMA ノードアフィニティーを持つものとして処理されます。 トンネル型ネットワーク (VXLAN, GRE, ...) はこの方法では説明できず、代わりに **[neutron_tunnel]** グループを使用して設定されます。以下に例を示します。 [neutron_tunnel] numa_nodes = 1 関連するオプション: ● **[neutron_tunnel] numa_nodes** を使用して、すべてのトンネリングネットワークの NUMA アフィニティーを設定できます。 ● **[neutron_physnet_$PHYSNET] numa_nodes** は、このオプションで指定した **$PHYSNET** の各値に対して設定する必要があります。
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名

設定オプション = デフォルト値	型	説明
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = network	文字列値	エンドポイント URL 検出のデフォルト service_type。
service_metadata_proxy = False	ブール値	True に設定すると、このオプションは Neutron を使用してメタデータ要求をプロキシー送信し、インスタンス ID を解決することを示しています。それ以外の場合は、インスタンス ID を **X-Instance-ID** ヘッダーのメタデータ要求に渡す必要があります。 関連するオプション: ● metadata_proxy_shared_secret
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID

設定オプション = デフォルト値	型	説明
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	User ID
username = None	文字列値	ユーザー名
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。

9.1.30. 通知

以下の表は、**/etc/nova/nova.conf** ファイルの **[notifications]** グループで使用できるオプションの概要をまとめたものです。

表9.29 通知

設定オプション = デフォルト値	型	説明
bdms_in_notifications = False	ブール値	有効な場合は、バージョン付けされた通知ペイロードにブロックデバイス情報を含めます。情報をデータベースから読み込む必要があるため、デフォルトでブロックデバイス情報の送信は、システムでオーバーヘッドが発生する可能性があるため、デフォルトで無効になっています。
default_level = INFO	文字列値	送信通知のデフォルト通知レベル。
notification_format = unversioned	文字列値	nova が出力すべき通知形式を指定します。 バージョン付けされた通知インターフェイスはレガシーインターフェイスと同等で、バージョン化されたインターフェイスはアクティブに開発され、新しいコンシューマーはバージョン付けされたインターフェイスを使用する必要があります。 ただし、従来のインターフェイスは ceilometer と他の成熟した OpenStack コンポーネントによって多用化されているため、デフォルトのままです。 **[oslo_messaging_notifications]** グループに **driver=noop** を設定して、通知を完全に無効にできることに注意してください。 バージョン付けされた通知のリストは、https://docs.openstack.org/nova/latest/reference/notifications.html に表示されます。

設定オプション = デフォルト値	型	説明
notify_on_state_change = None	文字列値	設定されている場合は、インスタンスの状態の変更時に compute.instance.update 通知を送信します。 通知の詳細は、https://docs.openstack.org/nova/latest/reference/notifications.html を参照してください。
versioned_notifications_topics = ['versioned_notifications']	リスト値	nova が発行するバージョン化された通知のトピックを指定します。 デフォルト値はほとんどのデプロイメントでは問題ありませんが、変更が必要になることはほとんどありません。ただし、バージョン付けされた通知を使用するサードパーティーのサービスがある場合は、そのサービスのトピックを取得することが望ましい場合があります。Nova は、バージョン別の通知ペイロードを含むメッセージを、このリストの各トピックキューに送信します。 バージョン付けされた通知のリストは、https://docs.openstack.org/nova/latest/reference/notifications.html に表示されます。

9.1.31. oslo_concurrency

以下の表では、**/etc/nova/nova.conf** ファイルの **[oslo_concurrency]** グループで利用可能なオプションについて説明しています。

表9.30 oslo_concurrency

設定オプション = デフォルト値	型	説明
disable_process_locking = False	ブール値	プロセス間ロックを有効または無効にします。
lock_path = None	文字列値	ロックファイルに使用するディレクトリー。セキュリティー上、指定したディレクトリーは、ロックが必要なプロセスを実行しているユーザーのみが書き込み可能である必要があります。デフォルトは環境変数 OSLO_LOCK_PATH です。外部ロックを使用する場合は、ロックパスを設定する必要があります。

9.1.32. oslo_messaging_amqp

以下の表では、**/etc/nova/nova.conf** ファイルの **[oslo_messaging_amqp]** グループで利用可能なオプションの概要を説明します。

表9.31 oslo_messaging_amqp

設定オプション = デフォルト値	型	説明
addressing_mode = dynamic	文字列値	ドライバーが使用するアドレスモードを示します。許可される値:**legacy** - レガシーのルーティング不可能なアドレス指定を使用します。**routable** - ルーティング可能なアドレスを使用します。**dynamic** - メッセージバスがルーティングをサポートしていない場合はレガシーのアドレスを使用します。それ以外の場合は、ルーティング可能なアドレスを使用します。
anycast_address = anycast	文字列値	コンシューマーのグループに送信する際に、アドレス接頭辞に追加されます。コンシューマー間でラウンドロビン方式で配信される必要のあるメッセージを特定するために、メッセージバスで使用されます。
broadcast_prefix = broadcast	文字列値	すべてのサーバーにブロードキャストする場合に使用されるアドレス接頭辞
connection_retry_backoff = 2	整数値	フェイルオーバーの試みに失敗するたびに、connection_retry_interval をこの秒数だけ増やします。
connection_retry_interval = 1	整数値	再接続を試みる前に一時停止する秒数。
connection_retry_interval_max = 30	整数値	connection_retry_interval + connection_retry_backoff の上限
container_name = None	文字列値	AMQP コンテナーの名前。グローバルで一意でなければなりません。デフォルトは、生成された UUID です。
default_notification_exchange = None	文字列値	通知アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_notification_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **notify**
default_notify_timeout = 30	整数値	送信された通知メッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_reply_retry = 0	整数値	リカバリー可能なエラーが原因で失敗した返信メッセージを再送信する最大試行回数。
default_reply_timeout = 30	整数値	rpc 返信メッセージ配信の期限。

設定オプション = デフォルト値	型	説明
default_rpc_exchange = None	文字列値	RPC アドレスで使用される変換名。エクスチェンジ名の解決の優先順位:Target.exchange、他に設定されている場合は default_rpc_exchange、他に設定されている場合は control_exchange、他に設定されている場合は **rpc**
default_send_timeout = 30	整数値	rpc キャストまたは呼び出しメッセージ配信の期限。呼び出し元がタイムアウトを明示的に指定しない場合にのみ使用されます。
default_sender_link_timeout = 600	整数値	アイドル状態の送信者リンクのパージをスケジュールする時間。期限切れ後にリンクの割り当てを解除します。
group_request_prefix = unicast	文字列値	グループの任意のサーバーに送信する際のアドレス接頭辞
idle_timeout = 0	整数値	非アクティブな接続のタイムアウト (秒単位)
link_retry_delay = 10	整数値	リカバリー可能なエラーが原因で失敗した AMQP 1.0 リンクの再接続間に一時停止する時間。
multicast_address = multicast	文字列値	ファンアウトメッセージを送信するときにアドレス接頭辞に追加されます。ファンアウトメッセージを識別するためにメッセージバスによって使用されます。
notify_address_prefix = openstack.org/om/notify	文字列値	すべての生成される通知アドレスのアドレス接頭辞
notify_server_credit = 100	整数値	受信通知メッセージのウィンドウサイズ
pre_settled = ['rpc-cast', 'rpc-reply']	多値	このタイプのメッセージを事前処置状態で送信します。事前処置されたメッセージは、ピアから確認応答を受信しません。注記: 事前処置されたメッセージは、配信に失敗した場合に警告なしに破棄される可能性があります。許可される値:**rpc-call** - RPC 呼び出しを事前処置状態で送信します。**rpc-reply**- RPC 返信を事前処置状態で送信します。**rpc-cast** - RPC キャストを事前処置状態で送信します。**notify** - 通知を事前処置状態で送信します。

設定オプション = デフォルト値	型	説明
pseudo_vhost = True	ブール値	仮想ホスト (qpidd など) をネイティブにサポートしていないメッセージバスの仮想ホストサポートを有効にします。true に設定すると、仮想ホスト名はすべてのメッセージバスアドレスに追加され、結果として仮想ホストごとにプライベート **サブネット** を作成します。メッセージバスが仮想ホストの名前として AMQP 1.0 オープン遂行動詞の **hostname** フィールドを使用して仮想ホストをサポートする場合は False に設定します。
reply_link_credit = 200	整数値	受信する RPC 返信メッセージのウィンドウサイズ。
rpc_address_prefix = openstack.org/om/rpc	文字列値	すべての生成される RPC アドレスのアドレス接頭辞
rpc_server_credit = 100	整数値	受信する RPC リクエストメッセージのウィンドウサイズ。
`sasl_config_dir = `	文字列値	SASL 設定が含まれるディレクトリーへのパス
`sasl_config_name = `	文字列値	設定ファイルの名前 (.conf 接尾辞なし)
`sasl_default_realm = `	文字列値	ユーザー名にレルムが存在しない場合に使用する SASL レルム
`sasl_mechanisms = `	文字列値	許可される SASL メカニズムのスペース区切りリスト
server_request_prefix = exclusive	文字列値	特定のサーバーに送信する時に使用するアドレス接頭辞
ssl = False	ブール値	SSL 経由で接続を試みます。その他の ssl 関連のパラメーターが指定されていない場合、システムの CA バンドルを使用してサーバーの証明書を検証します。
`ssl_ca_file = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_cert_file = `	文字列値	クライアント認証用の自己識別証明書 PEM ファイル
`ssl_key_file = `	文字列値	ssl_cert_file 証明書の署名に使用される秘密鍵 PEM ファイル (オプション)
ssl_key_password = None	文字列値	ssl_key_file を復号するためのパスワード (暗号化されている場合)

設定オプション = デフォルト値	型	説明
ssl_verify_vhost = False	ブール値	デフォルトでは、SSL は、サーバーの証明書の名前が transport_url のホスト名と一致することを確認します。設定によっては、代わりに仮想ホスト名を使用することが望ましい場合があります。たとえば、サーバーが Server Name Indication TLS 拡張 (rfc6066) を使用して、仮想ホストごとの証明書を提供する場合などです。サーバーの SSL 証明書が DNS 名ではなく仮想ホスト名を使用する場合は、ssl_verify_vhost を True に設定します。
trace = False	ブール値	デバッグ: AMQP フレームを標準出力 (stdout) にダンプします。
unicast_address = unicast	文字列値	特定の RPC/通知サーバーに送信する際に、アドレス接頭辞に追加されます。単一の送信先に送信されたメッセージを識別するためにメッセージバスによって使用されます。

9.1.33. oslo_messaging_kafka

以下の表は、**/etc/nova/nova.conf** ファイルの **[oslo_messaging_kafka]** グループで使用できるオプションの概要をまとめたものです。

表9.32 oslo_messaging_kafka

設定オプション = デフォルト値	型	説明
compression_codec = none	文字列値	プロデューサーによって生成されたすべてのデータの圧縮コーデック。設定されていない場合、圧縮は使用されません。この設定に許可される値は、kafka バージョンに依存することに注意してください。
conn_pool_min_size = 2	整数値	接続有効期限ポリシーのプールサイズ制限
conn_pool_ttl = 1200	整数値	プールのアイドル状態の接続の存続期間 (秒単位)
consumer_group = oslo_messaging_consumer	文字列値	Kafka コンシューマーのグループ ID。あるグループのコンシューマーは、メッセージ消費を調整します。
enable_auto_commit = False	ブール値	非同期コンシューマーコミットを有効にします。

設定オプション = デフォルト値	型	説明
kafka_consumer_timeout = 1.0	浮動小数点の値	Kafka コンシューマーのデフォルトタイムアウト
kafka_max_fetch_bytes = 1048576	整数値	Kafka コンシューマーの最大フェッチバイト
max_poll_records = 500	整数値	ポーリング呼び出しで返されるレコードの最大数
pool_size = 10	整数値	Kafka コンシューマーのプールサイズ
producer_batch_size = 16384	整数値	プロデューサーの非同期送信のバッチサイズ
producer_batch_timeout = 0.0	浮動小数点の値	KafkaProducer バッチ処理の遅延の上限 (秒単位)
sasl_mechanism = PLAIN	文字列値	セキュリティープロトコルが SASL である際のメカニズム
security_protocol = PLAINTEXT	文字列値	ブローカーとの通信に使用されるプロトコル
`ssl_cafile = `	文字列値	サーバーの証明書の検証に使用する CA 証明書 PEM ファイル
`ssl_client_cert_file = `	文字列値	認証に使用されるクライアント証明書の PEM ファイル。
`ssl_client_key_file = `	文字列値	認証に使用されるクライアントキーの PEM ファイル。
`ssl_client_key_password = `	文字列値	認証に使用されるクライアントキーパスワードファイル。

9.1.34. oslo_messaging_notifications

以下の表は、**/etc/nova/nova.conf** ファイルの **[oslo_messaging_notifications]** グループで使用できるオプションの概要をまとめたものです。

表9.33 oslo_messaging_notifications

設定オプション = デフォルト値	型	説明
driver = []	多値	通知の送信を処理するドライバー。使用できる値は messaging、messagingv2、routing、log、test、noop です。
retry = -1	整数値	リカバリー可能なエラーが原因で配信に失敗した通知メッセージを再送信する最大試行回数。0 - 再試行なし、-1 - 無限回
topics = ['notifications']	リスト値	OpenStack の通知に使用する AMQP トピック。
transport_url = None	文字列値	通知に使用するメッセージングドライバーを表す URL。設定されていない場合は、RPC に使用されるものと同じ設定にフォールバックします。

9.1.35. oslo_messaging_rabbit

以下の表は、**/etc/nova/nova.conf** ファイルの **[oslo_messaging_rabbit]** グループで使用できるオプションの概要をまとめたものです。

表9.34 oslo_messaging_rabbit

設定オプション = デフォルト値	型	説明
amqp_auto_delete = False	ブール値	AMQP の自動削除キュー。
amqp_durable_queues = False	ブール値	AMQP で永続キューを使用します。
direct_mandatory_flag = True	ブール値	(非推奨) ダイレクト送信のための RabbitMQ 必須フラグを有効/無効にします。直接送信は応答として使用されるため、クライアントキューが存在しない場合には MessageUndeliverable 例外が発生します。MessageUndeliverable 例外はタイムアウトをループして、送信者が回復する可能性が高くなります。このフラグは非推奨になり、この機能を非アクティブ化することはできません。
enable_cancel_on_failover = False	ブール値	x-cancel-on-ha-failover フラグを有効にして、rabbitmq サーバーがキューが停止しているときにコンシューマーをキャンセルし、通知できるようにします。

設定オプション = デフォルト値	型	説明
heartbeat_in_pthread = False	ブール値	デフォルトでは、ネイティブ Python スレッドを介してヘルスチェックハートビートスレッドを実行します。このオプションが False の場合、ヘルスチェックハートビートは親プロセスから実行モデルを継承します。たとえば、親プロセスが eventlet/greenlet を使用して stdlib にモンキーパッチを適用した場合、ハートビートはグリーンスレッドを介して実行されます。This option should be set to True only for the wsgi services.
heartbeat_rate = 2	整数値	heartbeat_timeout_threshold 中、ハートビートを確認する回数。
heartbeat_timeout_threshold = 60	整数値	ハートビートの keep-alive が失敗した場合に Rabbit ブローカーがダウンとみなされるまでの秒数 (0 はハートビートを無効にします)。
kombu_compression = None	文字列値	実験的用途: 許容値は gzip、bz2 です。設定されていない場合、圧縮は使用されません。このオプションは、今後のバージョンで利用できない可能性があります。
kombu_failover_strategy = round-robin	文字列値	現在接続しているノードが利用できなくなった場合に、次の RabbitMQ ノードを選択する方法を指定します。設定で複数の RabbitMQ ノードが指定される場合に限り有効になります。
kombu_missing_consumer_retry_timeout = 60	整数値	応答のないクライアントが応答を送信するのを待つ時間。この時間が経過すると無視します。この値は rpc_response_timeout より長くすることはできません。
kombu_reconnect_delay = 1.0	浮動小数点の値	AMQP コンシューマーの取り消し通知への応答で、再接続するまでの待機時間。
rabbit_ha_queues = False	ブール値	RabbitMQ (x-ha-policy: all) の HA キューの使用を試みます。このオプションを変更する場合は、RabbitMQ データベースを消去する必要があります。RabbitMQ 3.0 では、キューを宣言する際に x-ha-policy 引数によってキューのミラーリングが制御されなくなりました。すべてのキュー (自動生成された名前のキューを除く) がすべてのノードでミラーリングされるようにするには、"rabbitmqctl set_policy HA **^(?!amq\.).* {"ha-mode": "all"}**" を実行します。
rabbit_interval_max = 30	整数値	RabbitMQ 接続を再試行する最大間隔。デフォルトは 30 秒です。

設定オプション = デフォルト値	型	説明
rabbit_login_method = AMQPLAIN	文字列値	RabbitMQ ログイン方法
rabbit_qos_prefetch_count = 0	整数値	事前フェッチするメッセージの数を指定します。ゼロに設定すると、無制限のメッセージのフェッチが許可されます。
rabbit_retry_backoff = 2	整数値	RabbitMQ に接続する際に再試行間でバックオフする長さ
rabbit_retry_interval = 1	整数値	RabbitMQ との接続を再試行する頻度。
rabbit_transient_queues_ttl = 1800	整数値	キューの TTL (x-expires) の期間 (秒単位) を表す正の整数。TTL の期間使用されないキューは自動的に削除されます。このパラメーターは応答キューとファンアウトキューにのみ影響します。
ssl = False	ブール値	SSL 経由で接続します。
`ssl_ca_file = `	文字列値	SSL 認証局ファイル (SSL が有効な場合にのみ有効)。
`ssl_cert_file = `	文字列値	SSL 証明書ファイル (SSL が有効な場合にのみ有効)。
`ssl_key_file = `	文字列値	SSL キーファイル (SSL が有効な場合にのみ有効)。
`ssl_version = `	文字列値	使用する SSL バージョン (SSL が有効な場合にのみ有効)。有効な値は TLSv1 および SSLv23 です。SSLv2、SSLv3、TLSv1_1、および TLSv1_2 は、一部のディストリビューションで利用できます。

9.1.36. oslo_middleware

以下の表は、**/etc/nova/nova.conf** ファイルの **[oslo_middleware]** グループで使用できるオプションの概要をまとめたものです。

表9.35 oslo_middleware

設定オプション = デフォルト値	型	説明
enable_proxy_headers_parsing = False	ブール値	アプリケーションがプロキシーの背後であるかどうか。これにより、ミドルウェアがヘッダーを解析すべきかどうかを決定します。

設定オプション = デフォルト値	型	説明
max_request_body_size = 114688	整数値	各リクエストの本文の最大サイズ (バイト単位)。
secure_proxy_ssl_header = X-Forwarded-Proto	文字列値	SSL 終端プロキシーによって非表示であった場合でも、元のリクエストプロトコルスキームを決定するために使用される HTTP ヘッダー。

9.1.37. oslo_policy

以下の表は、**/etc/nova/nova.conf** ファイルの **[oslo_policy]** グループで使用できるオプションの概要をまとめたものです。

表9.36 oslo_policy

設定オプション = デフォルト値	型	説明
enforce_new_defaults = False	ブール値	このオプションは、ポリシーを評価するときに古い非推奨のデフォルトを使用するかどうかを制御します。**True** の場合、古い非推奨のデフォルトは評価されません。これは、既存のトークンが古いデフォルトで許可されているが、新しいデフォルトで許可されていない場合、許可されないことを意味します。新しいデフォルトと **scope_type** の利点を一緒に得ることができるように、このフラグを **enforce_scope** フラグと一緒に有効にすることを推奨します
enforce_scope = False	ブール値	このオプションは、ポリシーを評価する際にスコープを適用するかどうかを制御します。**True** の場合、リクエストで使用されるトークンのスコープは、適用されるポリシーの **scope_types** と比較されます。スコープが一致しない場合、**InvalidScope** 例外が発生します。**False** の場合、ポリシーが一致しないスコープで呼び出されていることを運用者に通知するメッセージが記録されます。
policy_default_rule = default	文字列値	デフォルトルール。要求されたルールが見つからない場合に適用されます。
policy_dirs = ['policy.d']	多値	ポリシー設定ファイルが保存されるディレクトリー。これは、config_dir オプションで定義される検索パス内の任意のディレクトリーへの相対パスか、絶対パスにすることができます。このディレクトリーの検索には、policy_file で定義されたファイルが存在している必要があります。存在しないディレクトリーまたは空のディレクトリーは無視されます。

設定オプション = デフォルト値	型	説明
policy_file = policy.yaml	文字列値	ロールを指定のサービスのパーミッションにマッピングするファイルの相対パスまたは絶対パス。相対パスは、このオプションを設定する設定ファイルに対する相対パスとして指定する必要があります。
remote_content_type = application/x-www-form-urlencoded	文字列値	REST ベースのポリシーチェックのデータを送受信するコンテンツ種別
remote_ssl_ca_crt_file = None	文字列値	REST ベースのポリシーチェック用の認証局証明書ファイルの絶対パス
remote_ssl_client_crt_file = None	文字列値	REST ベースのポリシーチェック用のクライアント証明書の絶対パス
remote_ssl_client_key_file = None	文字列値	REST ベースのポリシーチェック用のクライアント鍵ファイルの絶対パス
remote_ssl_verify_server_crt = False	ブール値	REST ベースのポリシーチェックのサーバーアイデンティティー検証

9.1.38. pci

以下の表は、**/etc/nova/nova.conf** ファイルの **[pci]** グループで使用できるオプションの概要をまとめたものです。

表9.37 pci

設定オプション = デフォルト値	型	説明
alias = []	多値	PCI パススルーデバイス要件のエイリアス。 これにより、すべての PCI プロパティー要件を繰り返す必要なく、フレーバーの追加スペックでエイリアスを指定できます。 これは **nova-api** サービス用に設定する必要があり、**nova-compute** サービスごとに move 操作を使用することを前提とします。 以下の値を使用できます。 ● エイリアスを記述する JSON 値のディクショナリー。以下に例を示します。 alias = { "name": "QuickAssist", "product_id": "0443",

設定オプション = デフォルト値	型	説明
		"vendor_id": "8086", "device_type": "type-PCI", "numa_policy": "required" } This defines an alias for the Intel QuickAssist card. (multi valued). Valid key values are : `name` Name of the PCI alias. `product_id` Product ID of the device in hexadecimal. `vendor_id` Vendor ID of the device in hexadecimal. `device_type` Type of PCI device. Valid values are: `type-PCI`, `type-PF` and `type-VF`. Note that `"device_type": "type-PF"` **must** be specified if you wish to passthrough a device that supports SR-IOV in its entirety. `numa_policy` Required NUMA affinity of device. Valid values are: `legacy`, `preferred` and `required`. ● オプションを繰り返すことにより、複数のエイリアスをサポートします (リスト値は **指定しません)。** alias = { "name": "QuickAssist-1", "product_id": "0443", "vendor_id": "8086", "device_type": "type-PCI", "numa_policy": "required" } alias = { "name": "QuickAssist-2", "product_id": "0444", "vendor_id": "8086", "device_type": "type-PCI", "numa_policy": "required" }
passthrough_whitelist = []	多値	仮想マシンで利用可能な PCI デバイスのホワイトリスト。 値: ● ホワイトリスト化された PCI デバイスを記述する JSON ディクショナリー。以下の形式を

設定オプション = デフォルト値	型	説明

		使用します。
		["vendor_id": "<id>",] ["product_id": "<id>",] ["address": "[[[[<domain>]:]<bus>].][<slot>][.[<function>]]" | "devname": "<name>",] {"<tag>": "<tag_value>",}
		Where `[` indicates zero or one occurrences, `{` indicates zero or multiple occurrences, and `|` mutually exclusive options. Note that any missing fields are automatically wildcarded.
		Valid key values are :
		`vendor_id`
		Vendor ID of the device in hexadecimal.
		`product_id`
		Product ID of the device in hexadecimal.
		`address`
		PCI address of the device. Both traditional glob style and regular expression syntax is supported. Please note that the address fields are restricted to the following maximum values:
		• domain - 0xFFFF
		• bus - 0xFF
		• slot - 0x1F
		• function - 0x7
		`devname`
		Device name of the device (for e.g. interface name). Not all PCI devices have a name.
		`<tag>`
		Additional `<tag>` and `<tag_value>` used for matching PCI devices.
		Supported `<tag>` values are :
		◦ **physical_network**

設定オプション = デフォルト値	型	説明
		○ **trusted** **有効な例は、** passthrough_whitelist = {"devname":"eth0", "physical_network":"physnet"} passthrough_whitelist = {"address":"**:0a:00.**"} passthrough_whitelist = {"address":":0a:00.", "physical_network":"physnet1"} passthrough_whitelist = {"vendor_id":"1137", "product_id":"0071"} passthrough_whitelist = {"vendor_id":"1137", "product_id":"0071", "address": "0000:0a:00.1", "physical_network":"physnet1"} passthrough_whitelist = {"address": {"domain": ".**", "bus": "02", "slot": "01", "function": "[2-7]"}, "physical_network":"physnet1"}** **passthrough_whitelist = {"address":{"domain": "."**, "bus": "02", "slot": "0[1-2]", "function": ".*"}, "physical_network":"physnet1"} passthrough_whitelist = {"devname": "eth0", "physical_network":"physnet1", "trusted": "true"} **相互排他的オプションを指定するため、以下は無効です。** passthrough_whitelist = {"devname":"eth0", "physical_network":"physnet", "address":"**:0a:00.**"} ● 上記の形式に対応する JSON ディクショナリーの JSON リスト。以下に **例を示します。** passthrough_whitelist = [{"product_id":"0001", "vendor_id":"8086"}, {"product_id":"0002", "vendor_id":"8086"}]

9.1.39. placement

以下の表は、**/etc/nova/nova.conf** ファイルの **[placement]** グループで使用できるオプションの概要をまとめたものです。

表9.38 placement

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
connect-retries = None	整数値	接続エラーに対して試行される再試行の最大数。
connect-retry-delay = None	浮動小数点の値	接続エラーに対する 2 つの再試行の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
endpoint-override = None	文字列値	このクライアントのリクエストには常にこのエンドポイント URL を使用します。注意: バージョンを指定しないエンドポイントはここで指定する必要があります。特定の API バージョンを要求するには、**version**、**min-version** オプション、または **max-version** オプション、もしくはそれを組み合わせてを使用します。
insecure = False	ブール値	HTTPS 接続を確認します。

設定オプション = デフォルト値	型	説明
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名
region-name = None	文字列値	エンドポイント URL 検出のデフォルト region_name。
service-name = None	文字列値	エンドポイント URL 検出のデフォルト service_name。
service-type = placement	文字列値	エンドポイント URL 検出のデフォルト service_type。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
status-code-retries = None	整数値	再試行可能な HTTP ステータスコードに対して試行される再試行の最大数。
status-code-retry-delay = None	浮動小数点の値	再試行可能なステータスコードに対する 2 つの再試行間の遅延 (秒単位)。設定されていない場合は、最大 60 秒まで 0.5 秒から開始する指数関数の再試行が使用されます。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID

設定オプション = デフォルト値	型	説明
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	User ID
username = None	文字列値	ユーザー名
valid-interfaces = ['internal', 'public']	リスト値	エンドポイント URL のインターフェイスのリスト (優先順)。

9.1.40. powervm

以下の表は、**/etc/nova/nova.conf** ファイルの **[powervm]** グループで使用できるオプションの概要をまとめたものです。

表9.39 powervm

設定オプション = デフォルト値	型	説明
disk_driver = localdisk	文字列値	PowerVM ディスクに使用するディスクドライバー。PowerVM は、localdisk および PowerVM 共有ストレージプールのディスクドライバーに対応します。 関連するオプション: ● volume_group_name - localdisk を使用する場合に必要です。
proc_units_factor = 0.1	浮動小数点の値	各仮想 CPU に提供される物理プロセッサーコンピュート電力の算出に使用される係数。つまり、値が 1.0 の場合は物理プロセッサー全体を意味しますが、0.05 は物理プロセッサーの 1/20 分の 1/20 になります。
`volume_group_name = `	文字列値	ブロックデバイスの操作に使用するボリュームグループdisk_driver が localdisk の場合は、この属性を指定する必要があります。rootvg を使用しないことを強く推奨します。これは、管理パーティションで使用し、いっぱいになるとエラーが発生してしまうからです。

9.1.41. privsep

以下の表は、**/etc/nova/nova.conf** ファイルの **[privsep]** グループで使用できるオプションの概要をまとめたものです。

表9.40 privsep

設定オプション = デフォルト値	型	説明
capabilities = []	リスト値	privsep デーモンによって保持される Linux 機能のリスト。
group = None	文字列値	privsep デーモンを実行する必要のあるグループです。
helper_command = None	文字列値	fork メソッドを使用していない場合に privsep デーモンを起動するコマンドを呼び出します。指定のない場合は、現在の設定を再作成するために設計された sudo privsep-helper および引数を使用してデフォルト値が生成されます。このコマンドは、適切な --privsep_context および --privsep_sock_path 引数を使用する必要があります。
logger_name = oslo_privsep.daemon	文字列値	Logger name to use for this privsep context.By default all contexts log with oslo_privsep.daemon.
thread_pool_size = <based on operating system>	整数値	privsep が同時にプロセスを実行できるスレッドの数。デフォルトは、システム内の CPU コア数に設定されます。
user = None	文字列値	privsep デーモンを実行する必要のあるユーザー。

9.1.42. profiler

以下の表は、**/etc/nova/nova.conf** ファイルの **[profiler]** グループで使用できるオプションの概要をまとめたものです。

表9.41 profiler

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
connection_string = messaging://	文字列値	通知バックエンドの接続文字列。 デフォルト値は **messaging://** で、通知機能を oslo_messaging に設定します。 使用できる値の例: ● **messaging://**: スパンの送信に oslo_messaging ドライバーを使用します。 ● **redis://127.0.0.1:6379**: スパンの送信に redis ドライバーを使用します。 ● **mongodb://127.0.0.1:27017**: スパンの送信に mongodb ドライバーを使用します。 ● **elasticsearch://127.0.0.1:9200:** elasticsearch ドライバーを使用してスパンを送信します。 ● **jaeger://127.0.0.1:6831**: スパンの送信に jaeger トレースを使用します。
enabled = False	ブール値	このノード上の全サービスのプロファイリングを有効にします。 デフォルト値は False です (プロファイリング機能を無効にします)。 値: ● True: 機能を有効にします ● False: 機能を無効にします。このプロジェクトの操作ではプロファイリングを開始できません。プロファイリングが別のプロジェクトでトリガーされた場合には、このプロジェクトの部分は空になります。
es_doc_type = notification	文字列値	elasticsearch の通知インデックスについてのドキュメントタイプ。
es_scroll_size = 10000	整数値	elasticsearch は大規模な要求をバッチに分割します。このパラメーターは、各バッチの最大サイズを定義します (例: es_scroll_size=10000)。
es_scroll_time = 2m	文字列値	このパラメーターは時間値パラメーター (es_scroll_time=2m など) です。これは、検索に参加するノードが継続してサポートするために関連するリソースを維持する時間を示します。

設定オプション = デフォルト値	型	説明
filter_error_trace = False	ブール値	エラー/例外を含むフィルタートレースを分離された場所へ有効にします。 デフォルト値は False に設定されます。 値: ● True: エラー/例外を含むフィルタートレースを有効にします。 ● False: フィルターを無効にします。
hmac_keys = SECRET_KEY	文字列値	パフォーマンスプロファイリング用にコンテキストデータの暗号化に使用するシークレットキー。 この文字列値は <key1>[,<key2>,...<keyn>] の形式にする必要があります。各キーはランダムな文字列です。REST API 経由でプロファイリングをトリガーするユーザーは、REST API 呼び出しのヘッダーにこれらのキーの 1 つを設定し、この特定プロジェクトのノードのプロファイリング結果が含まれるようにする必要があります。 プロファイリングを有効にするには、enabled フラグと hmac_keys 設定オプションの両方を設定する必要があります。また、少なくとも 1 つのキーで全サービス間で正しいプロファイリング情報を生成するには、OpenStack プロジェクト間で一貫性を確保する必要があります。これにより、クライアント側からを使用して、すべての可能なリソースからの情報を含むトレースを生成できます。
sentinel_service_name = mymaster	文字列値	Redissentinel はサービス名を使用してマスターの redis サービスを識別します。このパラメーターは名前を定義します (例: **sentinal_service_name=mymaster**)。
socket_timeout = 0.1	浮動小数点の値	Redissentinel は、接続に timeout オプションを提供します。このパラメーターは、タイムアウトを定義します (例: socket_timeout=0.1)。

設定オプション = デフォルト値	型	説明
trace_sqlalchemy = False	ブール値	サービスで SQL 要求のプロファイリングを有効にします。 デフォルト値は False です (SQL 要求はトレースされません)。 値: ● True: SQL 要求のプロファイリングを有効にします。各 SQL クエリーはトレースの一部となり、それにどの程度の時間を費やしたかによって分析できます。 ● False: SQL 要求のプロファイリングを無効にします。費やした時間は、高いレベルの操作でのみ表示されます。単一の SQL クエリーをこのように分析できません。

9.1.43. quota

以下の表は、**/etc/nova/nova.conf** ファイルの **[quota]** グループで使用できるオプションの概要をまとめたものです。

表9.42 quota

設定オプション = デフォルト値	型	説明
cores = 20	整数値	プロジェクトごとに許可されるインスタンスコアまたは vCPU の数。 値: ● 正の整数または 0。 ● -1: クォータを無効にします。
count_usage_from_placement = False	ブール値	Placement サービスからクォータ使用状況の数を有効にします。 Train からは、セルデータベースからカウントするのではなく、プレイスメントサービスからコアとラム、API データベースからインスタンスのクォータ使用量をカウントすることが可能になりました。 これは、配置デプロイメントごとに 1 つの Nova デプロイメントが実行されている場合に適切に機能します。ただし、オペレーターが配置デプロイメントを共有する複数の Nova デプロイメントを実行している場合は、このオプションを True に設定しないでください。現在配置サービスでは Nova デプロイメン

設定オプション = デフォルト値	型	説明
		トごとにリソースプロバイダーをパーティション分割する方法がないためです。このオプションがデフォルトのまま、または False に設定されている場合には、Nova は従来の counting メソッドを使用して、セルのデータベースからインスタンス、コア、および ram のクォータ使用量をカウントします。 このオプションが True に設定されている場合、サイズ変更に関連するクォータの使用動作は影響を受けることに注意してください。配置リソースの割り当ては、サイズ変更が確認されるか、元に戻されるまで、サイズ変更時にソースの割り当てを保持している間に宛先で要求されます。この間、サーバーが VERIFY_RESIZE 状態にあると、クォータの使用量は送信元と宛先の両方のリソース消費を反映します。これは、ダウンしたサイズを元に戻すために領域を確保するため便利ですが、サイズ変更の確認または元に戻すまでクォータの使用量が平文化されることを意味します。 動作は、ERROR 状態の未スケジュールのサーバーでも異なります。コンピュートホストにスケジュールされていない ERROR 状態のサーバーは配置割り当てを持たないため、コアと ram のクォータ使用量は消費されません。 動作は SHELVED_OFFLOADED 状態のサーバーでは異なります。SHELVED_OFFLOADED 状態のサーバーには配置割り当てがないため、コアと RAM のクォータ使用量は消費しません。このため、ユーザーに、サーバーの状態を外すためにサーバーに必要とされるコアと RAM をサポートできるクォータが十分にない場合、要求が拒否される可能性があります。 使用状況を配置からカウントするには、オンラインデータ移行の **populate_queued_for_delete** および **populate_user_id** を完了する必要があります。この設定オプションが True に設定されている場合、データ移行が完了するまで、各クォータチェック中の EXISTS データベースクエリーの結果に応じて、システムはセルデータベースからのレガシークォータ使用量カウントにフォールバックします。EXISTS クエリーによるパフォーマンスの低下を回避したい場合は、**nova-manage db online_data_migrations** でオンラインデータ移行を完了するまでこの設定オプションを True に設定するのを待つべきです。
driver = nova.quota.DbQuotaDriver	文字列値	クォータチェックの抽象化を提供します。クォータチェックに使用する特定のドライバーを設定できます。

設定オプション = デフォルト値	型	説明
injected_file_content_bytes = 10240	整数値	挿入されたファイルごとに許可されるバイト数。 値: - 正の整数または 0。 - -1: クォータを無効にします。
injected_file_path_length = 255	整数値	許可される最大ファイルパスの長さ。 値: - 正の整数または 0。 - -1: クォータを無効にします。
injected_files = 5	整数値	許可されるインジェクションファイルの数。 ファイルインジェクションにより、ユーザーは起動時にデータをインジェクトしてインスタンスのパーソナリティーをカスタマイズできます。テキストファイルの挿入のみが許可されます: バイナリーまたは ZIP ファイルは許可されません。ファイルインジェクション中に、指定したファイルに一致する既存のファイルの名前が変更され、タイムスタンプで追加された **.bak** 拡張子が含まれます。 値: - 正の整数または 0。 - -1: クォータを無効にします。
instances = 10	整数値	プロジェクトごとに許可されるインスタンス数。 設定可能な値 - 正の整数または 0。 - -1: クォータを無効にします。

設定オプション = デフォルト値	型	説明
key_pairs = 100	整数値	ユーザー 1 つにつき許容されるキーペアの最大数。 ユーザーは、プロジェクトごとに少なくとも 1 つのキーペアを作成し、そのプロジェクトに属する複数のインスタンスにキーペアを使用することができます。 値: ● 正の整数または 0。 ● -1: クォータを無効にします。
metadata_items = 128	整数値	インスタンスごとに許可されるメタデータ項目の数。 インスタンスの作成時にメタデータをインスタンスに関連付けることができます。このメタデータはキーと値のペアの形式です。 値: ● 正の整数または 0。 ● -1: クォータを無効にします。
ram = 51200	整数値	プロジェクトごとに許可されるインスタンス RAM のメガバイト数。 値: ● 正の整数または 0。 ● -1: クォータを無効にします。

設定オプション = デフォルト値	型	説明
recheck_quota = True	ブール値	リソースの作成後にクォータを再チェックして、クォータを超過できないようにします。 デフォルトは True (リソース作成後にクォータを再確認) ですが、レース要求のためにクォータを超過することを許容すると見なされる場合、追加の負荷を避けるために False に設定することが可能です。たとえば、False に設定すると、ユーザーがサーバーを作成するために非常に並列な REST API 要求を行う場合、競合時に許可されたクォータよりも多くのサーバーを作成することができます。クォータが 10 サーバーの場合、それらはバースト中に 50 を作成できる可能性があります。バーストの後には、より多くのサーバーを作成できなくなりますが、サーバーを削除するまで 50 台のサーバーを保持することができます。 最初のクォータチェックはリソースが作成される前に実行されるので、複数の並列要求が同時に到達すると、すべてがクォータチェックとリソースの作成をパスして、クォータを超過する可能性があります。recheck_quota が True の場合、リソースが作成された後に 2 回目のクォータチェックが行われ、リソースが quota を超えている場合は削除されて OverQuota が発生し、通常 REST API ユーザーに対して 403 応答が返されます。これにより、ユーザーがクォータを超過する可能性は不可能になります。ただし、ユーザーが要求を行ったときに利用可能なクォータが十分であっても、競合が近い場合に、REST API ユーザーが 403 応答で拒否される可能性があります。
server_group_members = 10	整数値	サーバーグループごとのサーバーの最大数。 値: - 正の整数または 0。 - -1: クォータを無効にします。

設定オプション = デフォルト値	型	説明
server_groups = 10	整数値	プロジェクトごとのサーバーグループの最大数。 サーバーグループは、サーバーまたはインスタンスのグループに対するアフィニティーおよびアンチアフィニティーのスケジューリングポリシーを制御するために使用されます。クォータを縮小しても、既存のグループには影響はありませんが、クォータを超えるグループには新しいサーバーは許可されません。 値: ● 正の整数または 0。 ● -1: クォータを無効にします。

9.1.44. rdp

以下の表は、**/etc/nova/nova.conf** ファイルの **[rdp]** グループで使用できるオプションの概要をまとめたものです。

表9.43 rdp

設定オプション = デフォルト値	型	説明
enabled = False	ブール値	Remote Desktop Protocol (RDP) の関連機能を有効にします。 Hyper-V は、Nova コンピュートノードで採用されているハイパーバイザーの大半とは異なり、デスクトップ共有プロトコルとして VNC および SPICE の代わりに RDP を使用して、インスタンスのコンソールアクセスを提供します。このオプションを使用すると、Hyper-V により作成される仮想マシンのグラフィカルコンソールアクセスに RDP が有効になります。 **注意:** RDP は、Hyper-V 仮想化プラットフォームをサポートするコンピュートノードでのみ有効にする必要があります。 関連するオプション: ● **compute_driver**: hyperv である必要があります。

設定オプション = デフォルト値	型	説明
html5_proxy_base_url = http://127.0.0.1:6083/	URI 値	エンドユーザーが RDP HTML5 コンソールプロキシーへの接続に使用する URL。コンソールプロキシーサービスは、この token-embedded URL で呼び出され、適切なインスタンスへの接続を確立します。 RDP HTML5 コンソールプロキシーサービスは、ここで設定されたアドレスをリッスンするように設定する必要があります。通常、コンソールプロキシーサービスはコントローラーノードで実行されます。デフォルトで使用されている localhost アドレスは、単一ノード環境 (devstack など) でのみ動作するものです。 RDP HTML5 プロキシーを使用すると、ユーザーは RDP を使用して Windows サーバーまたはワークステーションのテキストまたはグラフィカルコンソールを介してアクセスできます。RDP HTML5 コンソールプロキシーサービスには、FreeRDP、wsgate があります。https://github.com/FreeRDP/FreeRDP-WebConnect を参照してください。 値: ● <scheme>://<ip-address>:<port-number>/ > The scheme must be identical to the scheme configured for the RDP HTML5 > console proxy service. It is `http` or `https`. > The IP address must be identical to the address on which the RDP HTML5 > console proxy service is listening. > The port must be identical to the port on which the RDP HTML5 console proxy > service is listening. 関連するオプション: ● **rdp.enabled**: **html5_proxy_base_url** を有効にするには、**True** に設定する必要があります。

9.1.45. remote_debug

以下の表は、**/etc/nova/nova.conf** ファイルの **[remote_debug]** グループで使用できるオプションの概要をまとめたものです。

表9.44 remote_debug

設定オプション = デフォルト値	型	説明
host = None	ホストのアドレス値	接続するホスト (IP または名前) のデバッグ。 このコマンドラインパラメーターは、別のホストで実行しているデバッガーを使用して nova サービスに接続する場合に使用されます。 リモートデバッグオプションを使用すると、nova がイベントレットライブラリーを使用して非同期 IO をサポートする方法が変わることに注意してください。これにより、通常の操作でエラーが発生しない可能性がありました。自己責任でお使いください。 以下の値を使用できます。 ● nova サービスに対するコマンドラインパラメーターとしてリモートホストの IP アドレス **以下に例を示します。** nova-compute --config-file /etc/nova/nova.conf --remote_debug-host <IP address of the debugger>
port = None	ポート値	接続するポートをデバッグします。 このコマンドラインパラメーターを使用すると、異なるホストで実行しているデバッガーで nova サービスへの接続に使用するポートを指定できます。 リモートデバッグオプションを使用すると、nova がイベントレットライブラリーを使用して非同期 IO をサポートする方法が変わることに注意してください。これにより、通常の操作でエラーが発生しない可能性がありました。自己責任でお使いください。 以下の値を使用できます。 ● nova サービスのコマンドラインパラメーターとして使用するポート番号。 **以下に例を示します。** nova-compute --config-file /etc/nova/nova.conf --remote_debug-host <IP address of the debugger> --remote_debug-port <port debugger is listening on>.

9.1.46. scheduler

以下の表は、**/etc/nova/nova.conf** ファイルの **[scheduler]** グループで使用できるオプションの概要をまとめたものです。

表9.45 scheduler

設定オプション = デフォルト値	型	説明
discover_hosts_in_cells_interval = -1	整数値	定期的なタスク間隔。 この値は、セルに追加された新規ホストの検出をスケジューラーが試みる頻度 (秒単位) です。負の値 (デフォルト) の場合、自動検出は行われません。 コンピュートノードが頻繁に出入りするような環境では、この機能を有効にした方が良いでしょうし、ホストが追加されたときに手動で検出する方が、常にチェックすることによるオーバーヘッドを避けられます。有効にすると、この実行のたびに、各セルデータベースからマッピングされていないホストを選択するようになります。 値: ● 整数。整数は秒単位の定期的なタスク間隔に対応します。0 はデフォルトの間隔 (60 秒) を使用します。負の値を指定すると、定期的なタスクが無効になります。
enable_isolated_aggregate_filtering = False	ブール値	集計の使用を一致するメタデータを持つインスタンスに制限します。 この設定により、スケジューラーは、アグリゲートメタデータおよびインスタンスのフレーバー/イメージの要求される特性のマッチングに基づき、アグリゲートのホストを制限することができます。アグリゲートにキー **trait:$TRAIT_NAME** と値が **required** 属性が設定されている場合、そのアグリゲートのホストにスケジューリングされるためには、インスタンスのフレーバー extra_specs またはイメージのメタデータにも **trait:$TRAIT_NAME=required** が含まれている必要があります。その他の技術詳細: https://docs.openstack.org/nova/latest/reference/isolate-aggregates.html 値: ● ブール値。

設定オプション = デフォルト値	型	説明
image_metadata_prefilter = False	ブール値	配置を使用して、イメージメタデータに基づいてホストをフィルタリングします。 この設定により、スケジューラーは、イメージメタデータに基づいてホストをフィルタリングするために、既知のイメージメタデータプロパティーを配置に必要な特性に変換します。この機能にはホストサポートが必要であり、現在、次のコンピューティングドライバーでサポートされています。 ● **libvirt.LibvirtDriver** (Ussuri (21.0.0) 以降) 値: ● ブール値。 関連するオプション: ● **[compute] compute_driver**
limit_tenants_to_placement_aggregate = False	ブール値	テナントを特定の配置集約に制限します。 この設定により、スケジューラーは、受信リクエストのプロジェクトに設定された **filter_tenant_id** のメタデータキーを持つホストアグリゲートを検索し、配置からのリクエスト結果をそのアグリゲートに限定するようになります。**filter_tenant_id:123** などのシリアル番号をキーに追加して、複数のテナントを単一のアグリゲートに追加できます。 一致するアグリゲートの UUID は、適切な操作のために配置にミラーリングする必要があります。テナント ID を持つホストアグリゲートが見つからない場合や、そのアグリゲートが配置のものと一致しない場合、リクエストに適したホストが見つからない場合と同じ結果になります。 値: ● ブール値。 関連するオプション: ● **[scheduler] placement_aggregate_required_for_tenants**

設定オプション = デフォルト値	型	説明
max_attempts = 3	整数値	スケジュール試行の最大数。 これは、指定のインスタンスのビルド/移動操作に対して行われる最大試行回数です。スケジューラーによって返される代替ホストの数を制限します。ホストのリストが使い果たされると、**MaxRetriesExceeded** 例外が発生し、インスタンスはエラー状態に設定されます。 値: ● 正の整数。この整数は、インスタンスを構築または移動する際に行うことができる最大試行回数に対応する。
max_placement_results = 1000	整数値	リクエストする配置結果の最大数。 この設定は、スケジューリング中に配置サービスから受け取る結果の上限を決定します。これは、多数の候補に一致するスケジュール要求について考慮される可能性のあるホストの数を制限します。 1(最小値) を指定すると、適合するかどうかという理由だけで、事実上、配置サービスにスケジュールを委ねることになります。高い値を設定すると、スケジューラーがフィルタリングと重み付けの過程で考慮する結果の数に上限が設定されます。大規模な環境では、スケジューラーのメモリー消費量やネットワークトラフィックなどを制限するために、この値を利用可能なホストの総数よりも少なく設定する必要があるかもしれません。 値: ● 返される配置結果の数に対応する整数。

設定オプション = デフォルト値	型	説明
placement_aggregate_required_for_tenants = False	ブール値	すべてのテナントに配置集約の関連付けを要求します。 limit_tenants_to_placement_aggregate=True に設定すると、アグリゲートアフィニティーが設定されていないテナントを任意のノードにスケジュールできるかどうかを制御します。アグリゲートを使用して一部のテナントだけを制限する場合は、このパラメーターを False に設定する必要があります。すべてのテナントをアグリゲートで制限する必要がある場合には、利用可能なノードに対して無制限のスケジューリングを受け取れないように True にする必要があります。 値: ● ブール値。 関連するオプション: ● **[scheduler] placement_aggregate_required_for_tenants**
query_placement_for_availability_zone = False	ブール値	配置を使用して可用性ゾーンを決定します。 この設定により、スケジューラーは **availability_zone** のメタデータキーが受信要求により提供された値に設定されたホストアグリゲートを検索し、配置結果をそのアグリゲートに制限するように要求します。 一致するアグリゲートの UUID は、適切な操作のために配置にミラーリングする必要があります。**availability_zone** キーを持つホストアグリゲートが見つからない場合や、そのアグリゲートが配置にあるものと一致しない場合、結果は適切なホストを見つけられないのと同じになります。 このフラグを有効にすると、スケジューラーで (効率の低い) AvailabilityZoneFilter を無効にできることに注意してください。 値: ● ブール値。 関連するオプション: ● **[filter_scheduler] enabled_filters**

設定オプション = デフォルト値	型	説明
query_placement_for_image_type_support = False	ブール値	配置を使用して、インスタンスのイメージタイプに対するホストサポートを決定します。 この設定により、スケジューラーは、要求で使用されるイメージの **disk_format** をサポートするコンピュートホストの配置だけを要求するようになります。 値: ● ブール値。
query_placement_for_routed_network_aggregates = False	ブール値	スケジューラーを有効にして、ルーティングされたネットワークセグメントアグリゲートに関連付けられたコンピューティングホストをフィルタリングします。 詳細については、https://docs.openstack.org/neutron/latest/admin/config-routed-networks.html を参照してください。
workers = None	整数値	nova-scheduler サービスのワーカー数。 デフォルトは、使用可能な CPU の数です。 値: ● 整数。整数はワーカープロセスの数に対応します。

9.1.47. serial_console

以下の表は、**/etc/nova/nova.conf** ファイルの **[serial_console]** グループで使用できるオプションの概要をまとめたものです。

表9.46 serial_console

設定オプション = デフォルト値	型	説明

設定オプション = デフォルト値	型	説明
base_url = ws://127.0.0.1:6083/	URI 値	エンドユーザーが **nova-serialproxy** サービスへの接続に使用する URL。 **nova-serialproxy** サービスは、このトークンの強化 URL を使用して呼び出され、適切なインスタンスへの接続を確立します。 関連するオプション: ● IP アドレスは、**nova-serialproxy** サービスがリッスンしているアドレスと同じである必要があります (このセクションのオプション **serialproxy_host** を参照)。 ● ポートは、本セクションの **serialproxy_port** オプションと同じでなければなりません。 ● セキュリティーで保護された WebSocket 接続を使用する場合は、セキュリティーで保護されていない **ws://** ではなく **wss://** でこのオプションを開始します。そのためには、**[DEFAULT]** セクションのオプション **cert** と **key** を設定する必要があります。
enabled = False	ブール値	シリアルコンソール機能を有効にします。 この機能を使用するには、サービス **nova-serialproxy** を実行する必要があります。このサービスは通常、コントローラーノードで実行されます。
port_range = 10000:20000	文字列値	ゲストがバックエンドに使用できる TCP ポートの範囲。 作成される各インスタンスは、この範囲から 1 つのポートを使用します。新しいインスタンスに別のポートを提供する上で範囲が十分に大きくない場合は、このインスタンスは起動されません。 値: ● 正規表現 **^\d+:\d+$** を渡す各文字列 (例: **10000:20000**)。最初のポート番号が 2 番目のポート番号より小さく、両方が 0 から 65535 の範囲内にあることを確認してください。

設定オプション = デフォルト値	型	説明
proxyclient_address = 127.0.0.1	文字列値	インスタンスのシリアルコンソールを取得するために、プロキシークライアント (**nova-serialproxy** など) が接続する IP アドレス。 これは通常、**nova-compute** サービスのホストの IP アドレスになります。
serialproxy_host = 0.0.0.0	文字列値	**nova-serialproxy** サービスが着信要求をリッスンするために使用する IP アドレス。 **nova-serialproxy** サービスは、シリアルコンソールを公開するインスタンスへの着信接続要求をこの IP アドレスでリッスンします。 関連するオプション: ● これが、このセクションのオプション **base_url** で定義されているものと同じ IP アドレスであることを確認するか、**0.0.0.0** を使用してすべてのアドレスをリッスンします。
serialproxy_port = 6083	ポート値	**nova-serialproxy** サービスが受信リクエストをリッスンするために使用するポート番号。 **nova-serialproxy** サービスは、シリアルコンソールを公開するインスタンスへの着信接続要求をこのポート番号でリッスンします。 関連するオプション: ● これが、このセクションのオプション **base_url** で定義されているポート番号と同じであることを確認してください。

9.1.48. service_user

以下の表は、**/etc/nova/nova.conf** ファイルの **[service_user]** グループで使用できるオプションの概要をまとめたものです。

表9.47 service_user

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション

設定オプション = デフォルト値	型	説明
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名

設定オプション = デフォルト値	型	説明
send_service_user_token = False	ブール値	True の場合には、REST API にユーザートークンを送信すると、サービストークンも送信されます。 Nova は、Cinder、Glance、Neutron などの他の REST API と通信するために、nova-api に提供されたユーザートークンを再利用することがよくあります。要求が Nova に対して行われたときにユーザートークンが有効であったとしても、他のサービスに到達する前にトークンが期限切れになる可能性があります。障害を回避し、Nova がユーザーに代わってサービスを呼び出していることを明確にするために、ユーザートークンと共にサービストークンを含めます。ユーザーのトークンの有効期限が切れた場合でも、有効なサービストークンにより、REST API リクエストが引き続き keystone ミドルウェアによって受け入れられることが保証されます。
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	User ID
username = None	文字列値	ユーザー名

9.1.49. spice

以下の表は、**/etc/nova/nova.conf** ファイルの **[spice]** グループで使用できるオプションの概要をまとめたものです。

表9.48 spice

設定オプション = デフォルト値	型	説明
agent_enabled = True	ブール値	インスタンスで SPICE ゲストエージェントのサポートを有効にします。 Spice エージェントは Spice プロトコルを使用して、より優れたゲストコンソールエクスペリエンスを提供します。ただし、Spice コンソールは、Spice エージェントがなくても使用できます。Spice エージェントをインストールすると、次の機能が有効になります。 ● ゲストとクライアントマシン間でのテキストとイメージのコピーアンドペースト ● クライアント画面が変更されたときの解像度の自動調整: たとえば、Spice コンソールを全画面表示にすると、ゲストの解像度はレターボックスではなく、全画面表示に合わせて調整されます。 ● マウス統合の改善: コンソール内をクリックしたり、キーを押してリリースしたりすることなく、マウスをキャプチャーしてリリースすることができます。マウス移動のパフォーマンスも向上しています。
enabled = False	ブール値	SPICE 関連の機能を有効にします。 関連するオプション: ● SPICE コンソールにアクセスするには、VNC を明示的に無効にする必要があります。[vnc] セクションで有効化されているオプションを False に設定して、VNC コンソールを無効にします。

設定オプション = デフォルト値	型	説明
html5proxy_base_url = http://127.0.0.1:6082/spice_auto.html	URI 値	SPICE HTML5 コンソールプロキシーの場所。 エンドユーザーは、この URL を使用して **nova-spicehtml5proxy** サービスに接続します。このサービスは、リクエストをインスタンスのコンソールに転送します。 SPICE コンソールを使用するには、サービス **nova-spicehtml5proxy** が実行されている必要があります。このサービスは通常、コントローラーノードで起動されます。 値: ● **http://host:port/spice_auto.html** の形式の有効な URL である必要があります。ここで、ホストは **nova-spicehtml5proxy** を実行しているノードで、ポートは通常 6082 になります。デフォルト値は、実際のデプロイメントでは適切に定義されていないため、使用しないことを検討してください。 関連するオプション: ● このオプションは、**html5proxy_host** および **html5proxy_port** オプションに依存します。コンピュートノードによって返されるアクセス URL には、**nova-spicehtml5proxy** サービスがリッスンしているホストとポートが含まれている必要があります。
html5proxy_host = 0.0.0.0	ホストのアドレス値	**nova-spicehtml5proxy** サービスが着信要求をリッスンする IP アドレスまたはホスト名。 関連するオプション: ● このオプションは、**html5proxy_base_url** オプションに依存します。**nova-spicehtml5proxy** サービスは、HTML5 クライアントからアクセスできるホストでリッスンしている必要があります。

設定オプション = デフォルト値	型	説明
html5proxy_port = 6082	ポート値	**nova-spicehtml5proxy** サービスが受信リクエストをリッスンするポート。 関連するオプション: ● このオプションは、**html5proxy_base_url** オプションに依存します。**nova-spicehtml5proxy** サービスは、HTML5 クライアントからアクセスできるポートでリッスンしている必要があります。
server_listen = 127.0.0.1	文字列値	インスタンスで実行されている SPICE サーバーがリッスンする必要のあるアドレス。 通常、**nova-spicehtml5proxy** プロキシークライアントはコントローラーノードで実行され、プライベートネットワークを介してコンピュートノード上のこのアドレスに接続します。 値: ● リッスンする IP アドレス。
server_proxyclient_address = 127.0.0.1	文字列値	**nova-spicehtml5proxy** クライアントがインスタンスコンソールに接続するために使用するアドレス。 通常、**nova-spicehtml5proxy** プロキシークライアントはコントローラーノードで実行され、プライベートネットワークを介してコンピュートノード上のこのアドレスに接続します。 値: ● コンピュートノード上の任意の有効な IP アドレス。 関連するオプション: ● このオプションは、**server_listen** オプションに依存します。プロキシークライアントは、このオプションの値を使用して、**server_listen** で指定されたアドレスにアクセスできる必要があります。

9.1.50. upgrade_levels

以下の表は、**/etc/nova/nova.conf** ファイルの **[upgrade_levels]** グループで使用できるオプションの概要をまとめたものです。

表9.49 upgrade_levels

設定オプション = デフォルト値	型	説明
baseapi = None	文字列値	Base API RPC API バージョンキャップ。 値: ● デフォルトでは、クライアントが認識している最新バージョンを送信します。 ● **NN** 形式のバージョン番号を表す文字列。たとえば、可能な値は **1.12** または **2.0** などです。 ● **mitaka** や **liberty** など、小文字の OpenStack リリース名。
cert = None	文字列値	Cert RPC API バージョンキャップ。 値: ● デフォルトでは、クライアントが認識している最新バージョンを送信します。 ● **NN** 形式のバージョン番号を表す文字列。たとえば、可能な値は **1.12** または **2.0** などです。 ● **mitaka** や **liberty** など、小文字の OpenStack リリース名。 **非推奨となったバージョン:** 18.0.0 **理由:** nova-cert サービスは 16.0.0 (Pike) で削除されたため、このオプションは使用されなくなりました。

設定オプション = デフォルト値	型	説明
compute = None	文字列値	Compute RPC API バージョンキャップ。 デフォルトでは、クライアントが認識している最新のバージョンを使用して常にメッセージを送信します。 古いコンピュートサービスと新しいコンピュートサービスを実行している場合は、これを最も低いデプロイされたバージョンに設定する必要があります。これは、すべてのサービスが、いずれかのコンピュートノードが理解できないメッセージを送信することがないことを保証するためです。リリース N からリリース N+1 へのアップグレードのみがサポートされていることに注意してください。 デプロイメント内のサービスバージョンに基づいて、使用するバージョンをコンピュート RPC モジュールに自動的に決定させる場合は、このオプションを "auto" に設定します。 値: ● デフォルトでは、クライアントが認識している最新バージョンを送信します。 ● **auto**: デプロイメント内のサービスバージョンに基づいて、使用するバージョンを自動的に決定します。 ● **NN** 形式のバージョン番号を表す文字列。たとえば、可能な値は **1.12** または **2.0** などです。 ● **mitaka** や **liberty** など、小文字の OpenStack リリース名。
conductor = None	文字列値	Conductor RPC API バージョンキャップ。 値: ● デフォルトでは、クライアントが認識している最新バージョンを送信します。 ● **NN** 形式のバージョン番号を表す文字列。たとえば、可能な値は **1.12** または **2.0** などです。 ● **mitaka** や **liberty** など、小文字の OpenStack リリース名。

設定オプション = デフォルト値	型	説明
scheduler = None	文字列値	Scheduler RPC API バージョンキャップ。 値: ● デフォルトでは、クライアントが認識している最新バージョンを送信します。 ● NN 形式のバージョン番号を表す文字列。たとえば、可能な値は 1.12 または 2.0 などです。 ● **mitaka** や **liberty** など、小文字の OpenStack リリース名。

9.1.51. vault

以下の表は、**/etc/nova/nova.conf** ファイルの **[vault]** グループで使用できるオプションの概要をまとめたものです。

表9.50 vault

設定オプション = デフォルト値	型	説明
approle_role_id = None	文字列値	Vault での認証の場合は Approle role_id
approle_secret_id = None	文字列値	Vault での認証の場合は Approle secret_id
kv_mountpoint = secret	文字列値	使用する Vault の KV ストアのマウントポイント (例: secret)。
kv_version = 2	整数値	使用する Vault 内の KV ストアのバージョン。例: 2
root_token_id = None	文字列値	Vault の root トークン
ssl_ca_crt_file = None	文字列値	証明書ファイルの絶対パス
use_ssl = False	ブール値	SSL の有効化/無効化
vault_url = http://127.0.0.1:8200	文字列値	このエンドポイントを使用して Vault に接続します (例: "http://127.0.0.1:8200")。

9.1.52. vendordata_dynamic_auth

以下の表は、**/etc/nova/nova.conf** ファイルの **[vendordata_dynamic_auth]** グループで使用できるオプションの概要をまとめたものです。

表9.51 vendordata_dynamic_auth

設定オプション = デフォルト値	型	説明
auth-url = None	文字列値	認証 URL
auth_section = None	文字列値	プラグイン固有のオプションを読み込む config セクション
auth_type = None	文字列値	読み込む認証タイプ
cafile = None	文字列値	HTTP 接続の検証時に使用する PEM でエンコードされた認証局。
certfile = None	文字列値	PEM でエンコードされたクライアント証明書の証明書ファイル
collect-timing = False	ブール値	API ごとの呼び出しタイミング情報を収集します。
default-domain-id = None	文字列値	v3 および v2 パラメーターで使用するオプションのドメイン ID。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
default-domain-name = None	文字列値	v3 API および v2 パラメーターで使用するオプションのドメイン名です。これは v3 のユーザーとプロジェクトドメインの両方に使用され、v2 認証では無視されます。
domain-id = None	文字列値	スコープするドメイン ID
domain-name = None	文字列値	スコープするドメイン名
insecure = False	ブール値	HTTPS 接続を確認します。
keyfile = None	文字列値	PEM でエンコードされたクライアント証明書キーファイル
password = None	文字列値	ユーザーのパスワード
project-domain-id = None	文字列値	プロジェクトを含むドメイン ID
project-domain-name = None	文字列値	プロジェクトを含むドメイン名
project-id = None	文字列値	スコープするプロジェクト ID
project-name = None	文字列値	スコープするプロジェクト名

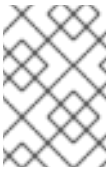
設定オプション = デフォルト値	型	説明
split-loggers = False	ブール値	複数のロガーへの要求をログに記録します。
system-scope = None	文字列値	システム操作のスコープ
tenant-id = None	文字列値	テナント ID
tenant-name = None	文字列値	テナント名
timeout = None	整数値	http 要求のタイムアウト値
trust-id = None	文字列値	Trust ID
user-domain-id = None	文字列値	ユーザーのドメイン ID
user-domain-name = None	文字列値	ユーザーのドメイン名
user-id = None	文字列値	User ID
username = None	文字列値	ユーザー名

9.1.53. vmware

以下の表は、**/etc/nova/nova.conf** ファイルの **[vmware]** グループで使用できるオプションの概要をまとめたものです。

表9.52 vmware

設定オプション = デフォルト値	型	説明
api_retry_count = 10	整数値	ソケットエラーなどの接続障害が発生した場合に、VMware vCenter Server API を再試行しなければならない回数。
ca_file = None	文字列値	vCenter サーバー証明書の検証に使用する CA バンドルファイルを指定します。

設定オプション = デフォルト値	型	説明
cache_prefix = None	文字列値	このオプションは、キャッシュされたイメージが保存されるフォルダーに接頭辞を追加します。 これは完全なパスではなく、単なるフォルダーの接頭辞です。これは、データストアキャッシュがコンピュートノード間で共有されている場合にのみ使用してください。 注意: これは、コンピュートノードが同じホストで実行されている場合、または共有ファイルシステムがある場合にのみ使用してください。 値: ● フォルダーのキャッシュ 接頭辞を表す任意の文字列
cluster_name = None	文字列値	VMware Cluster ComputeResource の名前。
connection_pool_size = 10	整数値	このオプションは、http 接続プールのサイズを設定します 接続プールサイズは、nova から vSphere への接続の最大数です。これは、接続プールがいっぱいであることを示す警告がある場合にのみ増やす必要があります。それ以外の場合は、デフォルトで十分です。
console_delay_seconds = None	整数値	リモートコンソールに入力するときに文字が繰り返されるネットワーク遅延の増加の影響を受ける場合は、この値を設定します。
datastore_regex = None	文字列値	データストアの名前に一致する正規表現パターン。 datastore_regex 設定は、Compute で使用するデータストアを指定します。たとえば、datastore_regex="nas.*" は、名前が "nas" で始まるすべてのデータストアを選択します。  **注記** 正規表現が指定されていない場合は、空き容量が最も多いデータストアが選択されます。 値: ● データストアに一致する正規表現を指定する必要があります。

設定オプション = デフォルト値	型	説明
host_ip = None	ホストのアドレス値	VMware vCenter ホストに接続するためのホスト名または IP アドレス。
host_password = None	文字列値	VMware vCenter ホストに接続するためのパスワード。
host_port = 443	ポート値	VMware vCenter ホストに接続するためのポート。
host_username = None	文字列値	VMware vCenter ホストに接続するためのユーザー名。
insecure = False	ブール値	true の場合、vCenter サーバー証明書は検証されません。false の場合、デフォルトの CA トラストストアが検証に使用されます。 関連オプション: * ca_file: ca_file が設定されている場合、このオプションは無視されます。
integration_bridge = None	文字列値	このオプションは、NSX-MH Neutron プラグインを使用する場合にのみ設定する必要があります。これは、ESXi サーバーまたはホスト上の統合ブリッジの名前です。これは、他の Neutron プラグインには設定しないでください。したがって、デフォルト値は設定されていません。 値: ● 統合ブリッジの名前を表す任意の有効な文字列。
maximum_objects = 100	整数値	このオプションは、1 つの結果で返されるオブジェクトの最大数の制限を指定します。 正の値を指定すると、オブジェクトの数が指定された制限に達したときに操作が取得を一時停止します。サーバーは引き続き、設定された値よりも小さい値に制限する可能性があります。残りのオブジェクトは、追加のリクエストで取得できます。

設定オプション = デフォルト値	型	説明
pbm_default_policy = None	文字列値	このオプションは、使用するデフォルトポリシーを指定します。 pbm_enabled が設定されていて、特定の要求に対してストレージポリシーが定義されていない場合は、このポリシーが使用されます。 値: ● VSAN のデフォルトストレージポリシーなどの有効なストレージポリシー 関連するオプション: ● pbm_enabled
pbm_enabled = False	ブール値	このオプションは、ストレージポリシーに基づくインスタンスの配置を有効または無効にします。 関連するオプション: ● pbm_default_policy
pbm_wsdl_location = None	文字列値	このオプションは、PBM サービスの WSDL ファイルの場所を示す URL を指定します。 これを設定すると、ストレージポリシーに基づくインスタンスの配置が無効になります。 値: ● file:///opt/SDK/spbm/wsdl/pbmService.wsdl などの任意の有効なファイルパス
serial_log_dir = /opt/vmware/vspc	文字列値	仮想シリアルポートコンセントレーターがコンソールログファイルを保存するディレクトリーを指定します。VSPC の **serial_log_dir** 設定値と一致する必要があります。
serial_port_proxy_uri = None	URI 値	serial_port_service_uri へのネットワークアクセスを提供するプロキシーサービスを識別します。 値: ● 任意の有効な URI (スキームは **telnet** または **telnets** です。) 関連オプション: serial_port_service_uri が指定されていない場合、このオプションは無視されます。* serial_port_service_uri

設定オプション = デフォルト値	型	説明
serial_port_service_uri = None	文字列値	シリアルポートトラフィックが送信されるリモートシステムを識別します。 このオプションは、コンソール出力を設定可能なサービス URI に送信する仮想シリアルポートを追加します。サービス URI アドレスには、コンソールログを収集する仮想シリアルポートコンセントレーターがあります。これが設定されていない場合、作成された VM にシリアルポートは追加されません。 値: ● 任意の有効な URI
task_poll_interval = 0.5	浮動小数点の値	VMware VC サーバーで呼び出されたリモートタスクをポーリングする時間間隔 (秒単位)。
use_linked_clone = True	ブール値	このオプションは、リンクされたクローンの使用を有効または無効にします。 ESX ハイパーバイザーは、仮想マシンを起動するために VMDK ファイルのコピーを必要とします。コンピュートドライバーは、HTTP 経由で OpenStack Image サービスからハイパーバイザーから見えるデータストアに VMDK をダウンロードし、それをキャッシュする必要があります。VMDK を必要とする後続の仮想マシンは、キャッシュされたバージョンを使用するため、OpenStack Image Service からファイルを再度コピーする必要はありません。 false に設定すると、VMDK がキャッシュされている場合でも、キャッシュの場所から共有データストア内のハイパーバイザーファイルディレクトリーへのコピー操作が実行されます。true に設定すると、仮想ディスクを親 VM と共有する仮想マシンのコピーが作成されるため、上記のコピー操作は回避されます。
vnc_keymap = en-us	文字列値	VNC のキーマップ。 キーボードマッピング (キーマップ) は、VNC セッションがデフォルトで使用するキーボードレイアウトを決定します。 値: ● このノードの基になるハイパーバイザーによってサポートされるキーボードレイアウト。これは通常、**IETF 言語タグ** (**en-us** など) になります。

設定オプション = デフォルト値	型	説明
vnc_port = 5900	ポート値	このオプションは、VNC 開始ポートを指定します。 ESX ホストによって作成されたすべての VM には、リモート接続用に VNC クライアントを有効にするオプションがあります。上記のオプション **vnc_port** は、VNC クライアントのデフォルトの開始ポートを設定する際に役立ちます。 値: ● 5900 -(5900 + vnc_port_total) 内の任意の有効なポート番号 関連オプション: VNC クライアントを有効にするには、以下のオプションを設定する必要があります。* vnc.enabled = True * vnc_port_total
vnc_port_total = 10000	整数値	VNC ポートの総数。

9.1.54. vnc

以下の表は、**/etc/nova/nova.conf** ファイルの **[vnc]** グループで使用できるオプションの概要をまとめたものです。

表9.53 vnc

設定オプション = デフォルト値	型	説明
auth_schemes = ['none']	リスト値	コンピュートノードで使用する認証スキーム。 プロキシーとコンピュートホスト間の接続で許可される RFB 認証スキームを制御します。複数のスキームが有効になっている場合、最初に一致するスキームが使用されるため、最も強力なスキームを最初にリストする必要があります。 関連するオプション: ● **[vnc]vencrypt_client_key**、**[vnc]vencrypt_client_cert**: これらも設定する必要があります。
enabled = True	ブール値	VNC 関連の機能を有効にします。 ゲストは、これをサポートするためにグラフィカルなデバイスを作成することになります。その後、クライアント (Horizon など) は、ゲストへの VNC 接続を確立できます。

設定オプション = デフォルト値	型	説明
novncproxy_base_url = http://127.0.0.1:6080/vnc_auto.html	URI 値	noVNC VNC コンソールプロキシーのパブリックアドレス。 VNC プロキシーは、コンピュートサービスのユーザーが、VNC クライアントを介してインスタンスにアクセスできるようにする OpenStack コンポーネントです。noVNC は、websocket ベースのクライアントを介して VNC サポートを提供します。 このオプションは、クライアントシステムが接続するパブリックベース URL を設定します。noVNC クライアントは、このアドレスを使用して noVNC インスタンスに接続し、エクステンションにより VNC セッションにも接続することができます。 noVNC >= 1.0.0 を使用する場合、**vnc_auto.html** の代わりに **vnc_lite.html** を使用する必要があります。 関連するオプション: ● novncproxy_host ● novncproxy_port
novncproxy_host = 0.0.0.0	文字列値	noVNC コンソールプロキシーがバインドする必要のある IP アドレス。 VNC プロキシーは、コンピュートサービスのユーザーが、VNC クライアントを介してインスタンスにアクセスできるようにする OpenStack コンポーネントです。noVNC は、websocket ベースのクライアントを介して VNC サポートを提供します。 このオプションは、noVNC コンソールプロキシーサービスがバインドする必要のあるプライベートアドレスを設定します。 関連するオプション: ● novncproxy_port ● novncproxy_base_url

設定オプション = デフォルト値	型	説明
novncproxy_port = 6080	ポート値	noVNC コンソールプロキシーがバインドする必要のあるポート。 VNC プロキシーは、コンピュートサービスのユーザーが、VNC クライアントを介してインスタンスにアクセスできるようにする OpenStack コンポーネントです。noVNC は、websocket ベースのクライアントを介して VNC サポートを提供します。 このオプションは、noVNC コンソールプロキシーサービスがバインドする必要のあるプライベートポートを設定します。 関連するオプション: ● novncproxy_host ● novncproxy_base_url
server_listen = 127.0.0.1	ホストのアドレス値	このノードで着信 VNC 接続要求をインスタンスがリッスンする必要がある IP アドレスまたはホスト名。
server_proxyclient_address = 127.0.0.1	ホストのアドレス値	VNC コンソールプロキシーのプライベート、内部 IP アドレスまたはホスト名。 VNC プロキシーは、コンピュートサービスのユーザーが、VNC クライアントを介してインスタンスにアクセスできるようにする OpenStack コンポーネントです。 このオプションは、**nova-novncproxy** などのプロキシークライアントが接続する必要のあるプライベートアドレスを設定します。
vencrypt_ca_certs = None	文字列値	CA 証明書の PEM ファイルへのパス コンピュートノード VNC サーバーによって使用される認証局の 1 つ以上の x509 証明書を含む PEM ファイルへの完全修飾パス。 関連するオプション: ● **vnc.auth_schemes** : **vencrypt** を含める必要があります

設定オプション = デフォルト値	型	説明
vencrypt_client_cert = None	文字列値	クライアントキーファイルへのパス (x509 の場合) VNC 認証中に VNC プロキシーサーバーがコンピュートノードに提示する x509 証明書を含む PEM ファイルへの完全修飾パス。 関連するオプション: ● **vnc.auth_schemes**:**vencrypt** を含める必要があります ● **vnc.vencrypt_client_key**: これも設定する必要があります
vencrypt_client_key = None	文字列値	クライアント証明書の PEM ファイルへのパス (x509 の場合) VNC 認証中に VNC プロキシーサーバーがコンピュートノードに提示する秘密鍵を含む PEM ファイルへの完全修飾パス。 関連するオプション: ● **vnc.auth_schemes**:**vencrypt** を含める必要があります ● **vnc.vencrypt_client_cert**: これも設定する必要があります

9.1.55. workarounds

以下の表は、**/etc/nova/nova.conf** ファイルの **[workarounds]** グループで使用できるオプションの概要をまとめたものです。

表9.54 workarounds

設定オプション = デフォルト値	型	説明
disable_compute_service_check_for_ffu = False	ブール値	これが設定されている場合、古いコンピューティングサービスの通常の安全性チェックは、エラーではなく警告として扱われます。これは、コンピュートノードがサービスレコードを更新できるようになる前に、新しい制御サービスが開始される Fast-Forward アップグレードを容易にするためにのみ有効にします。FFU では、データベース内のサービスレコードは、コンピュートノードが起動するまで複数のバージョンが古いものになりますが、制御サービスは最初にオンラインにする必要があります。

設定オプション = デフォルト値	型	説明
disable_fallback_pcpu_query = False	ブール値	固定されたインスタンスを使用する場合、VCPU 割り当てのフォールバックリクエストを無効にします。 Train 以降、libvirt virt ドライバーを使用するコンピュートノードは **PCPU** インベントリーを報告でき、これを固定されたインスタンスに使用します。スケジューラーは、従来の CPU ピニング関連のフレーバーエクストラ仕様 **hw:cpu_policy** と **hw:cpu_thread_policy**、それらに相当するイメージメタデータプロパティー、およびエミュレータースレッドピニングフレーバーエクストラ仕様 **hw:emulator_threads_policy** を使用して、リクエストを新しい配置リクエストに自動的に変換します。ただし、コンピュートノードでは、**PCPU** インベントリーを報告するために追加の設定が必要であり、この設定はアップグレード直後には存在しない場合があります。この追加設定なしで固定されたインスタンスを確実に作成できるようにするために、スケジューラーは古いスタイルの **VCPU** ベースの割り当ての配置に対して 2 番目の要求を行い、必要に応じてこれらの割り当て候補にフォールバックします。これはパフォーマンスにわずかな影響を与えますが、新しい設定がすべてのホストに設定されている新規またはアップグレードされたデプロイメントでは必要ありません。このオプションを設定すると、2 番目のルックアップが無効になり、スケジューラーは **PCPU** ベースの割り当てのみを要求します。 **非推奨となったバージョン:** 20.0.0 *理由:*None

設定オプション = デフォルト値	型	説明
disable_group_policy_check_upcall = False	ブール値	コンピュートで、サーバーグループポリシーチェックアップコールを無効にします。 サーバーグループアフィニティーポリシーとの競合を検出するために、コンピュートサービスは、ポリシーがスケジューラーによって違反されていないことを検証しようとします。これは、API データベースへのアップコールを行って、起動中のインスタンスのサーバーグループ内のインスタンスをリスト表示することによって行われます。これは、API/セル分離の目標に違反しています。最終的には、スケジューラーと配置サービスでの適切なアフィニティー保証によってこれが解決されますが、それまでは、適切なアフィニティーポリシーを確保するためにこの遅延チェックが必要です。 このチェックで API/セルの分離を望む Operator は、このフラグを有効にする必要があります。これにより、コンピュートからのアップコールが回避されます。 関連するオプション: ● [filter_scheduler]/track_instance_changes は、コンピュートサービスからスケジューラーサービスへのアップコールにも依存しています。

設定オプション = デフォルト値	型	説明
disable_libvirt_livesnapshot = False	ブール値	libvirt ドライバーを使用する場合は、ライブスナップショットを無効にします。 ライブスナップショットを使用すると、ゲストエージェントと連携してファイルシステムを静止させることで、ゲストを中断することなくディスクのスナップショットを作成できます。 libvirt 1.2.2 を使用すると、負荷がかかるとライブスナップショットが断続的に失敗します (libvirt/qemu の同時操作に関連している可能性があります)。この設定オプションは、この問題が解決されるまでの間、ライブスナップショットを無効にし、コールドスナップショットを優先させるメカニズムを提供します。コールドスナップショットは、ゲストがスナップショットプロセスを実行している間、インスタンスの停止を引き起こします。 詳細については、バグレポートを参照してください。 > https://bugs.launchpad.net/nova/+bug/1334398 値: - True: libvirt の使用時にライブスナップショットが無効化されます。 - False: ライブスナップショットは、スナップショットを作成するときに常に使用されます (十分な数の新しい libvirt があり、バックエンドストレージがそれをサポートしている限り)。 **非推奨となったバージョン:** 19.0.0 **理由:** このオプションは、libvirt 1.2.2 の問題を回避するために追加されました。このバージョンの libvirt はサポートされなくなりました。つまり、この回避策は不要になりました。これは今後のリリースで削除されます。

設定オプション = デフォルト値	型	説明
disable_native_luksv1 = False	ブール値	暗号化された LUKSv1 Cinder ボリュームをインスタンスにアタッチすると、Libvirt ドライバーは、暗号化されたディスクが QEMU によってネイティブに復号化されるように設定します。 QEMU が使用する libgcrypt ライブラリーで、このシナリオで I/O パフォーマンスをサーバー的に制限するパフォーマンスの問題が発見されました。 詳細については、次のバグレポートを参照してください。 RFE: ハードウェアアクセラレーション AES-XTS モード https://bugzilla.redhat.com/show_bug.cgi?id=1762765 この回避策オプションを有効にすると、Nova は従来の dm-crypt ベースの os-brick エンクリプターを使用して LUKSv1 ボリュームを復号化します。 Ceph などのホストブロックデバイスを提供しないボリュームを使用しているときにこのオプションを有効にすると、インスタンスからの起動またはインスタンスへのボリュームのアタッチに失敗することに注意してください。RBD でこれを回避する方法については、**[workarounds]/rbd_block_device** オプションを参照してください。 関連するオプション: ● **compute_driver** (libvirt) ● **rbd_block_device** (回避策) **非推奨:** 23.0.0 **理由:** この回避策の原因となった libgcrypt 内の根本的なパフォーマンスの低下は、1.8.5 で解決されました。

設定オプション = デフォルト値	型	説明
disable_rootwrap = False	ブール値	rootwrap の代わりに sudo を使用します。 パフォーマンス上の理由から、sudo へのフォールバックを許可します。 詳細については、バグレポートを参照してください。 > https://bugs.launchpad.net/nova/+bug/1415106 値: - True: rootwrap の代わりに sudo を使用する - False: rootwrap を通常どおり使用します。 他のオプションとの相互依存性: - **rootwrap** に影響するオプションはすべて無視されます。

設定オプション = デフォルト値	型	説明
enable_numa_live_migration = False	ブール値	NUMA トポロジーを使用したインスタンスのライブマイグレーションを有効にします。 libvirt ドライバーを使用する場合の NUMA トポロジーを使用したインスタンスのライブマイグレーションは、Train に完全にアップグレードされたデプロイメントでのみサポートされます。以前のバージョン、またはローリングアップグレードが進行中の Stein/Train 混合デプロイメントでは、libvirt ドライバーを使用する場合、NUMA トポロジーを使用するインスタンスのライブマイグレーションはデフォルトで無効になっています。これには、CPU ピニングまたは hugepage を使用したインスタンスのライブマイグレーションが含まれます。`bug #1289064`_ に記載されているように、そのようなインスタンスの CPU ピニングと hugepage の情報は、現時点では再計算されません。これは、インスタンスが宛先ホストにすでに存在する場合、移行されたインスタンスがこれらのインスタンスと同じ専用コアに配置されるか、別のインスタンスに割り当てられた hugepage を使用できることを意味します。また、ホストプラットフォームが同一でない場合、インスタンスが存在しないコアに割り当てられたり、ホスト NUMA ノード間で誤って分割されたりする可能性があります。 これらの既知の問題にもかかわらず、ライブマイグレーションが必要になる場合があります。このオプションを有効にすることで、問題を認識し、手動で回避する意思のあるオペレーターは、これらのインスタンスのライブマイグレーションサポートを有効にすることができます。 関連するオプション: ● **compute_driver**: libvirt ドライバーのみが影響を受けます。 a. _bug #1289064: https://bugs.launchpad.net/nova/+bug/1289064 **非推奨となったバージョン:** 20.0.0 *理由:* このオプションは、libvirt ドライバーを使用して NUMA トポロジーでインスタンスをライブマイグレーションする際の既知の問題を軽減するために追加されました。これらの問題は、Train で解決されます。libvirt ドライバーを使用し、Train に完全にアップグレードされたクラウドは、NUMA 対応のライブマイグレーションをサポートします。このオプションは、今後のリリースで削除される予定です。

設定オプション = デフォルト値	型	説明
enable_qemu_monitor_announce_self = False	ブール値	これが True に設定されている場合、libvirt ドライバーは、宛先のライブマイグレーション後のフェーズでネットワークスイッチを更新するための RARP フレームを生成するように、QEMU モニターに Announce-self コマンドを送信するための最善の努力を試みます。 これにより、ドメインが libvirt によって taint されていると見なされることに注意してください。 関連するオプション: ● :oslo.config:option:**DEFAULT.compute_driver** (libvirt)
ensure_libvirt_rbd_instance_dir_cleanup = False	ブール値	rbd を使用する場合、クリーンアップ中にインスタンスディレクトリーが削除されていることを確認します。 この回避策を有効にすると、**[libvirt]/images_type=rbd** を使用して、ホストでのクリーンアップ中にインスタンスディレクトリーが常に削除されるようになります。これにより、インスタンスディレクトリーがホストに残る原因となる、退避とサイズ変更のクリーンアップの取り消しに関する次のバグが回避されます。 https://bugs.launchpad.net/nova/+bug/1414895 https://bugs.launchpad.net/nova/+bug/1761062 これらのバグは両方とも、インスタンスがホストに戻ろうとすると、**DestinationDiskExists** エラーが発生する可能性があります。 a. 警告:: Operator は、この回避策を有効にする前に、**[DEFAULT]/instances_path** で指定されたインスタンスディレクトリー自体がコンピュート間で共有されていないことを確認する必要があります。そうしないと、実行中のインスタンスによって使用されている console.log、カーネル、ramdisk、および追加のファイルが失われます。 関連するオプション: ● **compute_driver** (libvirt) ● **[libvirt]/images_type** (rbd) ● **instances_path**

設定オプション = デフォルト値	型	説明
handle_virt_lifecycle_events = True	ブール値	コンピュートドライバーから発行されたイベントの処理を有効にします。 多くのコンピュートドライバーは、ライフサイクルイベントを発行します。ライフサイクルイベントは、たとえばインスタンスの開始時や停止時などに発生するイベントです。サイズ変更などの API 操作によってインスタンスのタスク状態が変化している場合、イベントは無視されます。 これは、インスタンスで予期しない状態変化が発生したこと、およびインスタンスを自動的にシャットダウンできることを、ハイパーバイザーがコンピュートサービスに通知できるようにする高度な機能です。残念ながら、これは、たとえば再起動操作中、コンピュートサービスの再起動時、またはホストの再起動時 (計画的または停電による) など、一部の条件で競合する可能性があります。このような競合が一般的である場合は、この機能を無効にすることを推奨します。 この機能が無効化され、**sync_power_state_interval** が負の値に設定されている場合は、注意が必要です。この場合、ハイパーバイザーと Nova データベースの間で同期が取れなくなったインスタンスは、手動で同期する必要があります。 詳細については、バグレポートを参照してください: https://bugs.launchpad.net/bugs/1444630 他のオプションとの相互依存性: ● **sync_power_state_interval** が負の値で、この機能が無効になっている場合、ハイパーバイザーと Nova データベースの間で非同期になったインスタンスは、手動で同期する必要があります。
libvirt_disable_apic = False	ブール値	一部のカーネルでゲスト API を初期化すると、ゲストが使用できなくなるカーネルハングが発生する可能性があります。これは、カーネルのバグの結果として発生します。ほとんどの場合、ゲストイメージカーネルをパッチが適用されたカーネルに更新することで修正されますが、これが不可能な場合もあります。この回避策により、API のエミュレーションをホストごとに無効にすることができますが、CI または開発者クラウドの外部で使用することは推奨しません。

設定オプション = デフォルト値	型	説明
never_download_image_if_on_rbd = False	ブール値	ceph でバックアップされたコンピュートノードのイメージから起動する場合、イメージが ceph クラスターにまだ存在しない場合 (glance も同じクラスターを使用している場合など)、nova は glance からイメージをダウンロードし、ceph 自体にアップロードします。複数の ceph クラスターを使用している場合、これにより nova がローカルの ceph デプロイメントで COW を使用できない方法で意図せずにイメージを複製し、スペースを浪費する可能性があります。 詳細については、バグレポートを参照してください。 https://bugs.launchpad.net/nova/+bug/1858877 このオプションを有効にすると、glance からイメージをダウンロードして ceph 自体にアップロードする必要がある場合、nova はインスタンスの起動を **拒否** します。 関連するオプション: ● **compute_driver** (libvirt) ● **[libvirt]/images_type** (rbd)

設定オプション = デフォルト値	型	説明
rbd_volume_local_attach = False	ブール値	RBD Cinder ボリュームをホストブロックデバイスとしてコンピュートにアタッチします。 このオプションを有効にすると、os-brick は RBD ボリュームを QEMU を介してネイティブに接続するのではなく、コンピュートホスト上でローカルにブロックデバイスとして接続するように指示されます。 この回避策は現在、アタッチされたボリュームの拡張をサポートしていません。 これを disable_native_luksv1 回避策設定オプションと共に使用して、libgcrypt ライブラリー内の最近発見されたパフォーマンスの問題を回避することができます。 この回避策は一時的なものであり、影響を受けるすべてのディストリビューションが libgcrypt ライブラリーのバージョンを更新できるようになると、W リリース中に削除される予定です。 関連するオプション: ● **compute_driver** (libvirt) ● **disable_qemu_native_luksv1** (回避策) **非推奨:** 23.0.0 **理由:** この回避策の原因となった libgcrypt 内の根本的なパフォーマンスの低下は、1.8.5 で解決されました。

設定オプション = デフォルト値	型	説明
reserve_disk_resource_for_image_cache = False	ブール値	True に設定されている場合、libvirt ドライバーはイメージキャッシュに保存されているイメージ用に DISK_GB リソースを予約します。:oslo.config:option:**DEFAULT.instances_path** がイメージキャッシュディレクトリーとは異なるディスクパーティションにある場合、ドライバーはキャッシュ用のリソースを予約しません。 このようなディスクの予約は、:oslo.config:option:**update_resources_interval** 秒ごとに実行されるリソーストラッカーの定期的なタスクによって行われます。そのため、イメージがキャッシュされても予約はすぐには更新されません。 関連するオプション: ● :oslo.config:option:**DEFAULT.instances_path** ● :oslo.config:option:**image_cache.subdirectory_name** ● :oslo.config:option:**update_resources_interval**
skip_cpu_compare_at_startup = False	ブール値	これにより、Compute サービスの起動時に CPU 比較の呼び出しがスキップされ、libvirt がそれを処理できるようになります。
skip_cpu_compare_on_dest = False	ブール値	これを有効にすると、宛先ホストでの CPU 比較がスキップされます。QEMU >= 2.9 および libvirt >= 4.4.0 を使用する場合、libvirt は、ライブマイグレーション中に宛先ホストの CPU 互換性をチェックすることについて正しい動作を実行します。
skip_hypervisor_version_check_on_lm = False	ブール値	これを有効にすると、ライブマイグレーション中にハイパーバイザーのバージョンチェックがスキップされます。
wait_for_vif_plugged_event_during_hard_reboot = []	リスト値	libvirt virt ドライバーは、再起動されるインスタンスのすべての vif を破棄してから再度接続することにより、電源オンとハード再起動を実装します。デフォルトでは、nova はインスタンスを実行する前に neutron からの network-vif-plugged イベントを待機しません。これにより、neutron バックエンドが vif プラグの後でネットワークバックエンドをセットアップする前に、インスタンスが DHCP 経由で IP を要求する可能性があります。 このフラグは、ハードリブート時に nova がどの vif

設定オプション = デフォルト値	型	説明
		から network-vif-plugged イベントを期待するかを定義します。可能な値は、neutron ポートの vnic タイプです。 • normal • direct • macvtap • baremetal • direct-physical • virtio-forwarder • smart-nic • vdpa • accelerator-direct • accelerator-direct-physical この設定に **vnic_type** を追加すると、Nova はインスタンスの一時停止を解除する前に、特定の **vnic_type** を持つインスタンスの各 vif の network-vif-plugged イベントを待機するようになります。 すべての neutron ネットワークバックエンドがプラグタイムイベントを送信するわけではないことに注意してください。特定の **vnic_type** については、この設定はデフォルトで空です。 ml2/ovs および networking-odl バックエンドは、**normal vnic_type** を持つポートのプラグ時間イベントを送信することが知られているため、計算ホストでこれらのバックエンドのみを使用している場合は、この設定に **normal** を追加しても安全です。 neutron in-tree SRIOV バックエンドは、**direct** vnic_type のポートのプラグ時間中に network-vif-plugged イベントを確実に送信せず、プラグ時間中に **direct-physical** vnic_type のポートに対してそのイベントを送信しません。他の **vnic_type** とバックエンドのペアについては、バックエンドの開発者に相談してください。 関連するオプション: • :oslo.config:option:**DEFAULT.vif_plugging_timeout**

9.1.56. wsgi

以下の表は、**/etc/nova/nova.conf** ファイルの **[wsgi]** グループで使用できるオプションの概要をまとめたものです。

表9.55 wsgi

設定オプション = デフォルト値	型	説明
api_paste_config = api-paste.ini	文字列値	このオプションは、nova-api の paste.deploy 設定のファイル名を表します。 値: ● paste.deploy 設定のファイル名を表す文字列。
client_socket_timeout = 900	整数値	このオプションは、クライアント接続のソケット操作のタイムアウトを指定します。受信接続がこの秒数の間アイドル状態であると、閉じられます。これは、ソケット接続での個々の読み取り/書き込みのタイムアウトを示します。永久に待機するには、0 に設定します。
default_pool_size = 1000	整数値	このオプションは、wsgi が使用する greenthreads のプールのサイズを指定します。このオプションを使用して、同時接続数を制限することができます。
keep_alive = True	ブール値	このオプションを使用すると、単一の要求/応答ペアごとに新しい接続を開くのではなく、同じ TCP 接続を使用して複数の HTTP 要求/応答を送受信できます。HTTP キープアライブは、HTTP 接続の再利用を示します。 値: ● True : HTTP 接続を再利用します。 ● False: クライアントのソケット接続を明示的に閉じます。 関連するオプション: ● tcp_keepidle
max_header_line = 16384	整数値	このオプションは、許可されるメッセージヘッダーの行の最大サイズを指定します。大規模なトークン (通常は、大規模なサービスカタログで Keystone v3 API で生成されたもの) を使用する場合に max_header_line を増やす必要がある場合があります。 TCP はストリームベースのプロトコルであるため、接続を再利用するために、HTTP は前の応答の終わりと次の応答の始まりを示す方法を持たなければなりません。したがって、keep_alive の場合、すべてのメッセージは自己定義のメッセージ長を持つ必要があります。

設定オプション = デフォルト値	型	説明
secure_proxy_ssl_header = None	文字列値	このオプションは、SSL 終了プロキシーによって削除された場合でも、元の要求のプロトコルスキームを決定するために使用される HTTP ヘッダーを指定します。 値: ● なし (デフォルト): リクエストスキームは HTTP ヘッダーの影響を受けません。 ● **HTTP_X_FORWARDED_PROTO** などの有効な HTTP ヘッダー **警告** よくわからない場合は、設定しないでください。 これを設定する前に、次のすべてが当てはまることを確認してください (上記の例の値を想定)。 ● API はプロキシーの背後にあります。 ● プロキシーは、すべての受信リクエストから X-Forwarded-Proto ヘッダーを削除します。つまり、エンドユーザーがそのヘッダーをリクエストに含めると、プロキシーはそれを破棄します。 ● プロキシーは X-Forwarded-Proto ヘッダーを設定し、それを API に送信しますが、元々 HTTPS 経由で受信したリクエストに対してのみ行います。 これらのいずれかが当てはまらない場合は、この設定を なしのままにしておく必要があります。
ssl_ca_file = None	文字列値	このオプションを使用すると、接続しているクライアントを検証するために使用する必要のある CA 証明書ファイルへのパスを設定できます。 値: ● CA 証明書ファイルへのパスを表す文字列。 関連するオプション: ● enabled_ssl_apis

設定オプション = デフォルト値	型	説明
ssl_cert_file = None	文字列値	このオプションを使用すると、API サーバーの SSL 証明書へのパスを設定できます。 値: ● SSL 証明書へのパスを表す文字列。 関連するオプション: ● enabled_ssl_apis
ssl_key_file = None	文字列値	このオプションは、SSL が有効な場合に、API サーバーの SSL 秘密鍵が保管されているファイルへのパスを指定します。 値: ● SSL 秘密鍵へのパスを表す文字列。 関連するオプション: ● enabled_ssl_apis
tcp_keepidle = 600	整数値	このオプションは、各サーバーソケットの TCP_KEEPIDLE の値を秒単位で設定します。接続をアクティブにしておく時間の長さを指定します。TCP は、接続をアクティブに保つことを要求するアプリケーションに対して KEEPALIVE 送信を生成します。OS X では対応していません。 関連するオプション: ● keep_alive

設定オプション = デフォルト値	型	説明
wsgi_log_format = %(client_ip)s "%(request_line)s" status: %(status_code)s len: %(body_length)s time: %(wall_seconds).7f	文字列値	これは、ログ行を生成するためのテンプレートとして使用される Python 形式の文字列を表します。client_ip、date_time、request_line、status_code、body_length、wall_seconds などの値をこれにフォーマットすることができます。 このオプションは、eventlet の下で nova-api を実行するときに、カスタムリクエストログラインを作成するために使用されます。uwsgi または apache で使用した場合、このオプションは効果がありません。 値: ● **%(client_ip)s "%(request_line)s" status: %(status_code)s ' 'len: %(body_length)s time: %(wall_seconds).7f** (デフォルト) ● 特定の値で形成された任意のフォーマットされた文字列。 **非推奨となったバージョン:** 16.0.0 **理由:** このオプションは、eventlet の下で nova-api を実行している場合にのみ機能し、非常に eventlet 固有の情報をエンコードします。Pike 以降では、nova-api を実行するための推奨モデルは uwsgi または apache mod_wsgi の下にあります。

9.1.57. zvm

以下の表は、**/etc/nova/nova.conf** ファイルの **[zvm]** グループで使用できるオプションの概要をまとめたものです。

表9.56 zvm

設定オプション = デフォルト値	型	説明
ca_file = None	文字列値	TLS が有効になっている httpd サーバーで検証される CA 証明書ファイル 文字列。使用する CA バンドルへのパスである必要があります。
cloud_connector_url = None	URI 値	z/VM Cloud Connector との通信に使用される URL。

設定オプション = デフォルト値	型	説明
image_tmp_path = $state_path/images	文字列値	イメージが保存されるパス (スナップショット、デプロイなど)。 デプロイに使用されるイメージと、スナップショットによってキャプチャーされたイメージは、コンピュートホストのローカルディスクに保存する必要があります。この設定は、ディレクトリーの場所を識別します。 可能な値: コンピューティングサービスを実行しているホスト上のファイルシステムパス。
reachable_timeout = 300	整数値	インスタンスの起動を待機するタイムアウト (秒)。 z/VM ドライバーは、インスタンスとクラウドコネクター間の通信に依存しています。インスタンスが作成された後、すべてのネットワーク情報がユーザーディレクトリーに書き込まれるまで待機する十分な時間が必要です。ドライバーは、タイムアウト値を使用してインスタンスのネットワークステータスを再確認し続けます。ネットワークの設定に失敗した場合、ドライバーはインスタンスの起動に失敗したことをユーザーに通知し、インスタンスを ERROR 状態にします。その後、基盤となる z/VM ゲストが削除されます。 可能な値: 任意の正の整数。300 秒 (5 分) 以上にすることを推奨しますが、インスタンスとシステムの負荷によって異なります。デバッグには値 0 が使用されます。この場合、インスタンスが ERROR 状態でマークされても、基礎となる z/VM ゲストは削除されません。