# OpenShift Container Platform 4.10

## ロギング

OpenShift Logging のインストール、使用法、およびリリースノート

Last Updated: 2024-01-12

# OpenShift Container Platform 4.10 ロギング

OpenShift Logging のインストール、使用法、およびリリースノート

## 法律上の通知

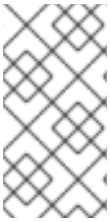


## 概要

本書では、OpenShift Logging のインストール、設定および使用方法について説明します。
OpenShift Logging は、各種の OpenShift Container Platform サービスについてのログを集計します。

# 目次

# 第1章 LOGGING のリリースノート



**注記**

Red Hat OpenShift のロギングサブシステムは、インストール可能なコンポーネントとして提供され、コアの OpenShift Container Platform とは異なるリリースサイクルを備えています。Red Hat OpenShift Container Platform ライフサイクルポリシー はリリースの互換性を概説しています。

**注記**

**stable** チャネルは、Logging の最新リリースを対象とする更新のみを提供します。以前のリリースの更新を引き続き受信するには、サブスクリプションチャネルを **stable-X** (**X** はインストールしたログのバージョン) に変更する必要があります。

## 1.1. LOGGING 5.6.11

このリリースには、OpenShift Logging バグ修正リリース 5.6.11 が含まれています。

### 1.1.1. バグ修正

- 今回の更新が行われる前は、LokiStack ゲートウェイは承認されたリクエストを非常に広範囲にキャッシュしていました。その結果、誤った認証結果が発生しました。今回の更新により、LokiStack ゲートウェイは詳細にキャッシュを行うようになり、この問題が解決されました。(LOG-4435)

### 1.1.2. CVE

- CVE-2023-3899
- CVE-2023-32360
- CVE-2023-34969

## 1.2. LOGGING 5.6.9

このリリースには、OpenShift Logging バグ修正リリース 5.6.9 が含まれています。

### 1.2.1. バグ修正

- この更新が行われる前は、AWS Cloudwatch 転送で STS を使用した認証に複数のロールが使用されていた場合、最近の更新により認証情報が一意でなくなりました。この更新により、STS ロールと静的認証情報の複数の組み合わせを再び AWS Cloudwatch での認証に使用できるようになりました。(LOG-4084)
- この更新が行われる前は、Vector コレクターに、ログに **thread 'vector-worker' panicked at 'all branches are disabled and there is no else branch', src/kubernetes/reflector.rs:26:9** エラーメッセージでパニックを発生させることがありました。今回の更新により、このエラーは解決されました。(LOG-4276)
- この更新が行われる前は、Loki はアクティブなストリームのラベル値をフィルタリングしていましたが、重複を削除しなかったため、Grafana の Label Browser が使用できなくなりました。今回の更新により、Loki はアクティブなストリームの重複するラベル値をフィルターで除

外し、問題を解決しました。(LOG-4390)

### 1.2.2. CVE

- CVE-2020-24736
- CVE-2022-48281
- CVE-2023-1667
- CVE-2023-2283
- CVE-2023-24329
- CVE-2023-26604
- CVE-2023-28466
- CVE-2023-32233

## 1.3. LOGGING 5.6.8

このリリースには、OpenShift Logging バグ修正リリース 5.6.8 が含まれています。

### 1.3.1. バグ修正

- この更新が行われる前は、入力一致ラベル値の **ClusterLogForwarder** 内に / 文字が含まれる場合は、vector コレクターが予期せず終了していました。今回の更新では、一致ラベルを引用符で囲み、コレクターがログを開始および収集できるようにすることで問題が解決されました。(LOG-4091)
- この更新が行われる前は、OpenShift Container Platform Web コンソール内でログを表示する際に **more data available** オプションをクリックすると、初回クリック時にのみ、より多くのログエントリーがロードされました。今回の更新では、クリックごとにさらに多くのエントリーが読み込まれるようになりました。(OU-187)
- この更新が行われる前は、OpenShift Container Platform Web コンソール内でログを表示する際に **streaming** オプションをクリックすると、実際のログは表示されず、**streaming logs** メッセージのみが表示されました。今回の更新により、メッセージとログストリームの両方が正しく表示されるようになりました。(OU-189)
- この更新が行われる前は、設定の問題が特定しにくい方法で Loki Operator がエラーをリセットしていました。今回の更新により、設定エラーが解決されるまでエラーが持続するようになりました。(LOG-4158)
- この更新が行われる前は、8,000 を超える namespace を持つクラスターの場合、namespace のリストが **http.max_header_size** 設定よりも大きくなるため Elasticsearch がクエリーを拒否していました。今回の更新では、ヘッダーサイズのデフォルト値が引き上げられ、問題が解決されました。(LOG-4278)

### 1.3.2. CVE

- CVE-2020-24736
- CVE-2022-48281

- CVE-2023-1667
- CVE-2023-2283
- CVE-2023-24329
- CVE-2023-26604
- CVE-2023-28466

## 1.4. LOGGING 5.6.7

このリリースには、OpenShift Logging Bug Fix Release 5.6.7 が含まれています。

### 1.4.1. バグ修正

- この更新が行われる前は、LokiStack ゲートウェイはユーザーのアクセス権を適用せずに namespace のラベル値を返していました。今回の更新により、LokiStack ゲートウェイはパーミッションをラベル値のリクエストに適用するようになり、問題は解決しました。(LOG-3728)
- この更新より前は、メッセージにタイムスタンプが含まれている場合、Fluentd ではログメッセージの **time** フィールドがデフォルトで **structured.time** として解析されませんでした。この更新により、出力先でサポートされている場合は、解析されたログメッセージに **structured.time** フィールドが含まれるようになります。(LOG-4090)
- この更新より前は、LokiStack ルート設定が原因で、クエリーの実行時間が 30 秒を超えるとタイムアウトが発生していました。今回の更新で、LokiStack global および per-tenant **queryTimeout** の設定によりルートタイムアウトの設定が影響を受け、問題は解決しました。(LOG-4130)
- この更新より前は、グローバル制限ではなくテナント制限に対して値が定義されている LokiStack CR により、Loki Operator がクラッシュしていました。今回の更新により、Operator はテナント制限のみが定義された LokiStack CR を処理できるようになり、問題が解決されました。(LOG-4199)
- この更新より前は、Web ブラウザーに保持されている以前のバージョンのキャッシュされたファイルが原因で、OpenShift Container Platform Web コンソールはアップグレード後にエラーを生成しました。今回の更新により、これらのファイルはキャッシュされなくなり、問題は解決されました。(LOG-4099)
- この更新が行われる前は、Vector はデフォルトの Loki インスタンスに転送するときに証明書エラーを生成していました。この更新により、Vector を使用してログをエラーなしで Loki に転送できるようになりました。(LOG-4184)
- この更新が行われる前は、**tls.insecureSkipVerify** オプションが **true** に設定されている場合に、Cluster Logging Operator API にはシークレットにより提供される証明書が必要でした。今回の更新により、そのような場合でも Cluster Logging Operator API はシークレットに証明書の提供を求めなくなりました。次の設定が Operator の CR に追加されました。

  ```
  tls.verify_certificate = false
  tls.verify_hostname = false
  ```

  (LOG-4146)

### 1.4.2. CVE

- CVE-2021-26341
- CVE-2021-33655
- CVE-2021-33656
- CVE-2022-1462
- CVE-2022-1679
- CVE-2022-1789
- CVE-2022-2196
- CVE-2022-2663
- CVE-2022-3028
- CVE-2022-3239
- CVE-2022-3522
- CVE-2022-3524
- CVE-2022-3564
- CVE-2022-3566
- CVE-2022-3567
- CVE-2022-3619
- CVE-2022-3623
- CVE-2022-3625
- CVE-2022-3627
- CVE-2022-3628
- CVE-2022-3707
- CVE-2022-3970
- CVE-2022-4129
- CVE-2022-20141
- CVE-2022-25147
- CVE-2022-25265
- CVE-2022-30594
- CVE-2022-36227
- CVE-2022-39188

- CVE-2022-39189
- CVE-2022-41218
- CVE-2022-41674
- CVE-2022-42703
- CVE-2022-42720
- CVE-2022-42721
- CVE-2022-42722
- CVE-2022-43750
- CVE-2022-47929
- CVE-2023-0394
- CVE-2023-0461
- CVE-2023-1195
- CVE-2023-1582
- CVE-2023-2491
- CVE-2023-22490
- CVE-2023-23454
- CVE-2023-23946
- CVE-2023-25652
- CVE-2023-25815
- CVE-2023-27535
- CVE-2023-29007

## 1.5. LOGGING 5.6.6

このリリースには、OpenShift Logging Bug Fix Release 5.6.6 が含まれています。

### 1.5.1. バグ修正

- この更新が行われるまでは、ペイロード内のキーに一致する Kafka 出力トピックに書き込むように **ClusterLogForwarder** カスタムリソースを設定すると、エラーによりメッセージのドロップが発生していました。今回の更新では、Fluentd のバッファー名の前にアンダースコアを付けることで問題が解決しました。(LOG-3458)
- この更新が行われるまでは、inode が再利用され、同じ inode を持つ複数のエントリーが存在する場合に、Fluentd でウォッチの早期終了が発生していました。この更新により、Fluentd 位置ファイル内のウォッチが早期に終了する問題が解決しました。(LOG-3629)

- この更新が行われるまでは、Fluentd による JavaScript クライアントの複数行例外の検出に失敗し、結果として例外が複数行として出力されていました。今回の更新により、例外が 1 行で出力されるようになり、問題が解決されました。(LOG-3761)

- この更新が行われるまでは、Red Hat Openshift Logging Operator バージョン 4.6 からバージョン 5.6 への直接アップグレードが許可されていたため、機能上の問題が発生していました。この更新により、アップグレードは 2 つのバージョン以内である必要があり、問題が解決されました。(LOG-3837)

- この更新が行われるまでは、Splunk または Google Logging 出力のメトリックは表示されませんでした。この更新では、HTTP エンドポイントのメトリックを送信することで問題が解決されました。(LOG-3932)

- この更新が行われるまでは、**ClusterLogForwarder** カスタムリソースが削除されても、コレクター Pod は実行されたままでした。この更新により、ログ転送が有効になっていない場合、コレクター Pod は実行されなくなります。(LOG-4030)

- この更新が行われるまでは、OpenShift Container Platform Web コンソールでログのヒストグラムをクリックしてドラッグしても時間範囲を選択できませんでした。今回の更新により、クリックとドラッグを使用して時間範囲を正常に選択できるようになりました。(LOG-4101)

- この更新が行われるまでは、監視ファイルの Fluentd ハッシュ値はログファイルへのパスを使用して生成されていたため、ログローテーション時に一意ではないハッシュが生成されました。今回の更新により、監視ファイルのハッシュ値が inode 番号で作成されるようになり、問題が解決されました。(LOG-3633)

- この更新が行われる前は、OpenShift Container Platform Web コンソールで **Show Resources** リンクをクリックしても何の効果もありませんでした。この更新では、ログエントリーごとにリソースの表示を切り替える **リソースの表示** リンクの機能を修正することで、この問題が解決されました。(LOG-4118)

### 1.5.2. CVE

- CVE-2023-21930
- CVE-2023-21937
- CVE-2023-21938
- CVE-2023-21939
- CVE-2023-21954
- CVE-2023-21967
- CVE-2023-21968
- CVE-2023-28617

## 1.6. LOGGING 5.6.5

このリリースには、OpenShift Logging Bug Fix Release 5.6.5 が含まれています。

### 1.6.1. バグ修正

- この更新前は、テンプレート定義により Elasticsearch が一部のラベルと namespace_label のイ

ンデックスを作成できず、データの取り込みで問題が発生していました。この更新では、ラベル内のドットとスラッシュが修正により置き換えられ、適切な取り込みが保証され、問題が効果的に解決されます。(LOG-3419)

- この更新より前は、OpenShift Web コンソールのログページが LokiStack への接続に失敗した場合、一般的なエラーメッセージが表示され、追加のコンテキストやトラブルシューティングの提案は提供されませんでした。この更新により、エラーメッセージが強化され、より具体的な詳細とトラブルシューティングの推奨事項が含まれるようになりました。(LOG-3750)

- この更新より前は、時間範囲形式が検証されていなかったため、カスタム日付範囲を選択するとエラーが発生していました。この更新により、時間形式が検証されるようになり、ユーザーが有効な範囲を選択できるようになりました。無効な時間範囲形式が選択された場合は、ユーザーにエラーメッセージが表示されます。(LOG-3583)

- この更新前は、Loki でログを検索すると、式の長さが 5120 文字を超えていなくても、多くの場合クエリーは失敗していました。この更新により、クエリー承認ラベルマッチャーが最適化され、問題が解決されました。(LOG-3480)

- この更新が行われる前は、Loki Operator は、プライベート IP のメンバーリストを使用する場合に、すべてのコンポーネントを見つけるのに十分なメンバーリスト設定を生成できませんでした。今回の更新により、生成された設定にアドバタイズされたポートが確実に含まれるようになり、すべてのコンポーネントを正常に検索できるようになりました。(LOG-4008)

### 1.6.2. CVE

- CVE-2022-4269
- CVE-2022-4378
- CVE-2023-0266
- CVE-2023-0361
- CVE-2023-0386
- CVE-2023-27539
- CVE-2023-28120

## 1.7. LOGGING 5.6.4

このリリースには、OpenShift Logging Bug Fix Release 5.6.4 が含まれています。

### 1.7.1. バグ修正

- この更新の前は、LokiStack がログストアとしてデプロイされたときに、Loki Pod によって生成されたログが収集され、LokiStack に送信されていました。今回の更新により、Loki によって生成されたログは収集から除外され、保存されなくなります。(LOG-3280)

- この更新の前は、OpenShift Web コンソールのログページのクエリーエディターが空の場合は、ドロップダウンメニューに値が入力されませんでした。今回の更新により、空のクエリーを実行しようとすると、エラーメッセージが表示され、ドロップダウンメニューが期待どおりに入力されるようになりました。(LOG-3454)

- この更新の前は、**tls.insecureSkipVerify** オプションが **true** に設定されている場合は、Cluster Logging Operator が誤った設定を生成していました。その結果、Operator は証明書の検証を

スキップしようとすると、データを Elasticsearch に送信できませんでした。今回の更新により、**tls.insecureSkipVerify が** 有効になっている場合でも、Cluster Logging Operator は正しい TLS 設定を生成します。その結果、証明書の検証をスキップしようとしても、データを Elasticsearch に正常に送信できます。(LOG-3475)

- この更新の前は、構造化された解析が有効になっていて、メッセージが複数の宛先に転送された場合に、それらはディープコピーされませんでした。これにより、構造化されたメッセージを含む一部の受信ログが生成されましたが、その他のログは生成されませんでした。今回の更新により、JSON 解析の前にメッセージをディープコピーするように設定生成が変更されました。その結果、複数の宛先に転送された場合でも、すべての受信メッセージに構造化メッセージが含まれるようになりました。(LOG-3640)

- この更新の前は、**collection** フィールドに **{}** が含まれていると、Operator がクラッシュする可能性がありました。今回の更新により、Operator はこの値を無視するようになり、オペレータは中断することなくスムーズに実行し続けることができます。(LOG-3733)

- この更新の前は、LokiStack のゲートウェイコンポーネントの **nodeSelector** 属性は効果がありませんでした。今回の更新により、**nodeSelector** 属性が期待どおりに機能するようになりました。(LOG-3783)

- この更新の前は、静的な LokiStack メンバーリストの設定は、プライベート IP ネットワークのみに依存していました。その結果、OpenShift Container Platform クラスター Pod ネットワークがパブリック IP 範囲で設定されている場合、LokiStack Pod がクラッシュループします。今回の更新により、LokiStack 管理者は、メンバーリストの設定に Pod ネットワークを使用するオプションを利用できるようになりました。これにより、問題が解決され、OpenShift Container Platform クラスター Pod ネットワークがパブリック IP 範囲で設定されている場合に、LokiStack Pod がクラッシュループ状態になるのを防ぐことができます。(LOG-3814)

- この更新の前は、**tls.insecureSkipVerify** フィールドが **true** に設定されている場合、Cluster Logging Operator は間違った設定を生成していました。その結果、証明書の検証をスキップしようとすると、Operator は Elasticsearch にデータを送信できませんでした。今回の更新により、**tls.insecureSkipVerify** が有効になっている場合でも、Operator は正しい TLS 設定を生成します。その結果、証明書の検証をスキップしようとしても、データを Elasticsearch に正常に送信できます。(LOG-3838)

- この更新の前に、Elasticsearch Operator を使用せずに Cluster Logging Operator (CLO) がインストールされた場合、CLO Pod は Elasticsearch の削除に関連するエラーメッセージを継続的に表示していました。今回の更新により、CLO はエラーメッセージを表示する前に追加のチェックを実行するようになりました。その結果、Elasticsearch Operator が存在しない場合は、Elasticsearch の削除に関連するエラーメッセージが表示されなくなりました。(LOG-3763)

### 1.7.2. CVE

- CVE-2022-4304
- CVE-2022-4450
- CVE-2023-0215
- CVE-2023-0286
- CVE-2023-0767
- CVE-2023-23916

## 1.8. LOGGING 5.6.3

このリリースには、OpenShift Logging Bug Fix Release 5.6.3 が含まれています。

### 1.8.1. バグ修正

- 今回の更新の前は、Operator はゲートウェイテナントのシークレット情報を config map に保存していました。今回の更新により、Operator はこの情報をシークレットに保存します。(LOG-3717)
- 今回の更新の前は、Fluentd コレクターは **/var/log/auth-server/audit.log** に保存されている OAuth ログインイベントをキャプチャーしませんでした。今回の更新により、Fluentd は、これらの OAuth ログインイベントをキャプチャーし、問題を解決しました。(LOG-3729)

### 1.8.2. CVE

- CVE-2020-10735
- CVE-2021-28861
- CVE-2022-2873
- CVE-2022-4415
- CVE-2022-40897
- CVE-2022-41222
- CVE-2022-43945
- CVE-2022-45061
- CVE-2022-48303

## 1.9. LOGGING 5.6.2

このリリースには、OpenShift Logging バグ修正リリース 5.6.2 が含まれます。

### 1.9.1. バグ修正

- この更新の前は、コレクターは systemd ログの優先度に基づいて **level** フィールドを正しく設定しませんでした。今回の更新により、**level** フィールドが正しく設定されるようになりました。(LOG-3429)
- 今回の更新以前は、Operator は OpenShift Container Platform 4.12 以降で非互換性の警告を誤って生成していました。今回の更新により、Operator の OpenShift Container Platform の最大バージョン値が修正され、問題が解決されました。(LOG-3584)
- 今回の更新以前は、**default** の出力値で **ClusterLogForwarder** カスタムリソース (CR) を作成し、エラーを生成しませんでした。今回の更新により、この値が適切に生成されるというエラーの警告が表示されるようになりました。(LOG-3437)
- この更新の前は、**ClusterLogForwarder** カスタムリソース (CR) に 1 つの出力が **default** として設定された複数のパイプラインがある場合、コレクター Pod は再起動していました。今回の更新で、出力検証のロジックが修正され、問題が解決されました。(LOG-3559)

- この更新の前は、コレクター Pod は作成後に再起動されていました。今回の更新により、デプロイされたコレクターが自動的に再起動しなくなりました。(LOG-3608)

- 今回の更新以前は、パッチリリースがカタログから以前のバージョンの Operator を削除していました。これにより、古いバージョンをインストールできませんでした。今回の更新により、バンドル設定が変更され、同じマイナーバージョンの以前のリリースがカタログに留まるようになりました。(LOG-3635)

### 1.9.2. CVE

- CVE-2022-23521
- CVE-2022-40303
- CVE-2022-40304
- CVE-2022-41903
- CVE-2022-47629
- CVE-2023-21835
- CVE-2023-21843

## 1.10. LOGGING 5.6.1

このリリースには、OpenShift Logging バグ修正リリース 5.6.1 が含まれます。

### 1.10.1. バグ修正

- この更新の前は、保持が有効な場合、コンパクターは、クエリーアとの通信による TLS 証明書エラーを報告していました。今回の更新により、コンパクターとクエリーアが HTTP 経由で誤って通信することがなくなりました。(LOG-3494)

- この更新の前は、Loki Operator は **LokiStack** CR のステータスの設定を再試行しないため、ステータス情報が古くなっていました。今回の更新により、Operator は競合時にステータス情報の更新を再試行するようになりました。(LOG-3496)

- この更新の前は、**kube-apiserver-operator** Operator が Webhook の有効性を確認したときに、Loki Operator Webhook サーバーが TLS エラーを引き起こしていました。今回の更新により、Loki Operator Webhook PKI は Operator Lifecycle Manager (OLM) によって管理されるようになり、問題が解決されました。(LOG-3510)

- この更新の前は、ブール式と組み合わせてラベルフィルターを使用した場合、LokiStack ゲートウェイラベルエンフォーサーが有効な LogQL クエリーの解析エラーを生成していました。今回の更新により、LokiStack LogQL の実装がブール式を使用したラベルフィルターをサポートするようになり、問題が解決されました。(LOG-3441)、(LOG-3397)

- この更新の前は、複数のラベルキーに同じ接頭辞があり、一部のキーにドットが含まれていると、Elasticsearch に書き込まれたレコードで障害が発生していました。今回の更新により、ラベルキーのドットがアンダースコアに置き換えられ、問題が解決されました。(LOG-3463)

- この更新の前は、OpenShift Container Platform コンソールと logging-view-plugin の間に互換性がないため、**Red Hat OpenShift Logging** Operator は OpenShift Container Platform 4.10 クラスターで使用できませんでした。今回の更新により、プラグインは OpenShift Container Platform 4.10 管理コンソールと適切に統合されるようになりました。(LOG-3447)

- この更新の前は、**ClusterLogForwarder** カスタムリソースの調整で、デフォルトログストアを参照するパイプラインの低下ステータスが誤って報告されていました。今回の更新により、パイプラインが適切に検証されるようになりました (LOG-3477)。

### 1.10.2. CVE

- CVE-2021-46848
- CVE-2022-3821
- CVE-2022-35737
- CVE-2022-42010
- CVE-2022-42011
- CVE-2022-42012
- CVE-2022-42898
- CVE-2022-43680
- CVE-2021-35065
- CVE-2022-46175

## 1.11. LOGGING 5.6

このリリースには、OpenShift Logging Release 5.6 が含まれています。

### 1.11.1. 非推奨の通知

Logging 5.6 では、Fluentd が非推奨となり、今後のリリースで削除される予定です。Red Hat は、この機能に対して現在のリリースライフサイクル中にバグ修正とサポートを提供しますが、拡張機能の提供はなく、この機能は今後削除される予定です。Fluentd の代わりに、Vector を使用できます。

### 1.11.2. 機能拡張

- 今回の更新により、Logging は OpenShift Container Platform の クラスター全体の暗号化ポリシーに 準拠します。(LOG-895)
- 今回の更新により、LokiStack カスタムリソースを使用して、テナントごと、ストリームごと、およびグローバルポリシーの保持ポリシーを優先度順に宣言できるようになります。(LOG-2695)
- 今回の更新により、Splunk がログ転送の出力オプションとして利用できるようになります。(LOG-2913)
- 今回の更新により、デフォルトコレクターが Fluentd から Vector になります。(LOG-2222)
- 今回の更新により、**Developer** ロールは、OpenShift Container Platform 4.11 以降を実行しているクラスターの Log Console Plugin 内で割り当てられているプロジェクトごとのワークロードログにアクセスできます。(LOG-3388)

- 今回の更新により、任意のソースからのログに、Operator がデプロイされているクラスターの一意識別子であるフィールド **openshift.cluster_id** が含まれるようになります。**clusterID** の値は、以下のコマンドで表示できます。(LOG-2715)

```
$ oc get clusterversion/version -o jsonpath='{.spec.clusterID}{"\n"}'
```

### 1.11.3. 既知の問題

- この更新の前は、複数のラベルキーに同じ接頭辞があり、一部のキーに **.** が含まれていると、Elasticsearch がログを拒否することがありました。これは、ラベルキーに含まれる **.** を **_** に置き換えることで、Elasticsearch の制限を修正します。この問題の回避策として、エラーの原因となっているラベルを削除するか、namespace をラベルに追加します。(LOG-3463)

### 1.11.4. バグ修正

- 今回の更新の前は、Kibana カスタムリソースを削除した場合、OpenShift Container Platform Web コンソールは引き続き Kibana へのリンクを表示していました。今回の更新で、Kibana カスタムリソースを削除すると、そのリンクも削除されます。(LOG-2993)

- この更新の前は、ユーザーはアクセス権を持つ namespace のアプリケーションログを表示できませんでした。今回の更新により、Loki Operator はクラスターロールとクラスターロールバインディングを自動的に作成し、ユーザーがアプリケーションログを読み取れるようにします。(LOG-3072)

- この更新の前に、LokiStack をデフォルトのログストレージとして使用する場合、Operator は **ClusterLogForwarder** カスタムリソースで定義されたカスタム出力を削除しました。今回の更新により、Operator は **ClusterLogForwarder** カスタムリソースの処理時にカスタム出力をデフォルト出力とマージします。(LOG-3090)

- この更新の前に、CA キーは CA を Loki にマウントするためのボリューム名として使用されていたため、CA キーに非準拠の文字 (ドットなど) が含まれているとエラー状態が発生していました。今回の更新により、ボリューム名が内部文字列に標準化され、問題が解決されました。(LOG-3331)

- この更新の前は、LokiStack カスタムリソース定義内で設定されたデフォルト値が原因で、**1** の **ReplicationFactor** なしで LokiStack インスタンスを作成できませんでした。今回の更新により、Operator は使用されるサイズの実際の値を設定するようになります。(LOG-3296)

- この更新の前は、Vector は、JSON 解析が有効になっている場合に、**structuredTypeKey** または **structuredTypeName** の値も定義せずにメッセージフィールドを解析していました。今回の更新により、構造化ログを Elasticsearch に書き込むときに、**structuredTypeKey** または **structuredTypeName** のいずれかに値が必要になりました。(LOG-3195)

- この更新の前は、Elasticsearch Operator のシークレット作成コンポーネントが内部シークレットを常に変更していました。今回の更新により、既存のシークレットが適切に処理されるようになりました。(LOG-3161)

- この更新の前は、Elasticsearch または Kibana デプロイメントのステータスが変更されている間に、Operator がコレクターデーモンセットの削除と再作成のループに入る可能性がありました。今回の更新では、Operator のステータス処理が修正され、問題が解決されました。(LOG-3157)

- この更新の前は、Kibana の OAuth cookie の有効期限は **24h** に固定されていたため、**accessTokenInactivityTimeout** フィールドが **24h** 未満の値に設定されていると、Kibana で 401 エラーが発生していました。今回の更新により、Kibana の OAuth cookie の有効期限が **accessTokenInactivityTimeout** に同期され、デフォルト値は **24h** になります。(LOG-3129)

- この更新の前は、リソースを調整するための Operator の一般的なパターンとして、取得または更新を試みる前に作成を試みていました。そのため、作成後に一定の HTTP 409 レスポンスが発生していました。今回の更新により、Operator は最初にオブジェクトの取得を試み、オブジェクトが欠落しているか指定されていない場合にのみ作成または更新するようになります。(LOG-2919)

- この更新の前は、Fluentd の **.level** フィールドと .structure.level フィールドに異なる値が含まれることがありました。今回の更新により、各フィールドの値が同じになります。(LOG-2819)

- この更新の前は、Operator は信頼された CA バンドルの作成を待たず、バンドルが更新された後に 2 回目のコレクターデプロイメントを実行していました。今回の更新により、Operator は、コレクターデプロイメントを続行する前に、バンドルが読み込まれたかどうかを確認するために少し待機するようになります。(LOG-2789)

- この更新の前は、メトリクスを確認するときに Telemetry ログ情報が 2 回表示されていました。今回の更新により、Telemetry ログ情報が想定どおりに表示されます。(LOG-2315)

- この更新の前は、JSON 解析の追加を有効にした後、Fluentd Pod ログに警告メッセージが含まれていました。今回の更新でにより、その警告メッセージは表示されなくなりました。(LOG-1806)

- この更新の前は、キャッシュをビルドするために書き込み権限を持つフォルダーを **oc** が必要とするため、**must-gather** スクリプトは完了しませんでした。今回の更新により、**oc** はフォルダーへの書き込み権限を持ち、**must-gather** スクリプトが正常に完了するようになります。(LOG-3446)

- この更新の前は、ログコレクター SCC がクラスター上の他の SCC に取って代わられ、コレクターが使用できなくなる可能性がありました。今回の更新により、ログコレクター SCC の優先度が設定され、他の SCC よりも優先されるようになりました。(LOG-3235)

- この更新の前は、Vector に **sequence** フィールドはなく、ナノ秒単位の精度の欠落に対処する方法として fluentd に追加されていました。今回の更新により、**openshift.sequence** フィールドがイベントログに追加されました。(LOG-3106)

### 1.11.5. CVE

- CVE-2020-36518
- CVE-2021-46848
- CVE-2022-2879
- CVE-2022-2880
- CVE-2022-27664
- CVE-2022-32190
- CVE-2022-35737
- CVE-2022-37601
- CVE-2022-41715
- CVE-2022-42003
- CVE-2022-42004

- CVE-2022-42010
- CVE-2022-42011
- CVE-2022-42012
- CVE-2022-42898
- CVE-2022-43680

## 1.12. LOGGING 5.5.16

このリリースには、OpenShift Logging バグ修正リリース 5.5.16 が含まれています。

### 1.12.1. バグ修正

- 今回の更新が行われる前は、LokiStack ゲートウェイは承認されたリクエストを非常に広範囲にキャッシュしていました。その結果、誤った認証結果が発生しました。今回の更新により、LokiStack ゲートウェイは詳細にキャッシュを行うようになり、この問題が解決されました。(LOG-4434)

### 1.12.2. CVE

- CVE-2023-3899
- CVE-2023-32360
- CVE-2023-34969

## 1.13. LOGGING 5.5.14

このリリースには、OpenShift Logging バグ修正リリース 5.5.14 が含まれています。

### 1.13.1. バグ修正

- この更新が行われる前は、Vector コレクターに、ログに **thread 'vector-worker' panicked at 'all branches are disabled and there is no else branch', src/kubernetes/reflector.rs:26:9** エラーメッセージでパニックを発生させることがありました。今回の更新により、このエラーは解決されました。(LOG-4279)

### 1.13.2. CVE

- CVE-2023-2828

## 1.14. LOGGING 5.5.13

本リリースには、OpenShift Logging バグ修正リリース 5.5.13 が含まれています。

### 1.14.1. バグ修正

なし。

### 1.14.2. CVE

- CVE-2023-1999
- CVE-2020-24736
- CVE-2022-48281
- CVE-2023-1667
- CVE-2023-2283
- CVE-2023-24329
- CVE-2023-26604
- CVE-2023-28466

## 1.15. LOGGING 5.5.11

このリリースには、OpenShift Logging Bug Fix Release 5.5.11 が含まれています。

### 1.15.1. バグ修正

- この更新が行われるまでは、OpenShift Container Platform Web コンソールでログのヒストグラムをクリックしてドラッグしても時間範囲を選択できませんでした。今回の更新により、クリックとドラッグを使用して時間範囲を正常に選択できるようになりました。(LOG-4102)
- この更新が行われる前は、OpenShift Container Platform Web コンソールで **Show Resources** リンクをクリックしても何の効果もありませんでした。この更新では、ログエントリーごとにリソースの表示を切り替える **リソースの表示** リンクの機能を修正することで、この問題が解決されました。(LOG-4117)

### 1.15.2. CVE

- CVE-2021-26341
- CVE-2021-33655
- CVE-2021-33656
- CVE-2022-1462
- CVE-2022-1679
- CVE-2022-1789
- CVE-2022-2196
- CVE-2022-2663
- CVE-2022-2795
- CVE-2022-3028
- CVE-2022-3239

- CVE-2022-3522
- CVE-2022-3524
- CVE-2022-3564
- CVE-2022-3566
- CVE-2022-3567
- CVE-2022-3619
- CVE-2022-3623
- CVE-2022-3625
- CVE-2022-3627
- CVE-2022-3628
- CVE-2022-3707
- CVE-2022-3970
- CVE-2022-4129
- CVE-2022-20141
- CVE-2022-24765
- CVE-2022-25265
- CVE-2022-29187
- CVE-2022-30594
- CVE-2022-36227
- CVE-2022-39188
- CVE-2022-39189
- CVE-2022-39253
- CVE-2022-39260
- CVE-2022-41218
- CVE-2022-41674
- CVE-2022-42703
- CVE-2022-42720
- CVE-2022-42721
- CVE-2022-42722

- CVE-2022-43750
- CVE-2022-47929
- CVE-2023-0394
- CVE-2023-0461
- CVE-2023-1195
- CVE-2023-1582
- CVE-2023-2491
- CVE-2023-23454
- CVE-2023-27535

## 1.16. LOGGING 5.5.10

このリリースには、OpenShift Logging Bug Fix Release 5.5.10 が含まれています。

### 1.16.1. バグ修正

- 今回の更新以前は、OpenShift Web コンソールのロギングビュープラグインは、LokiStack に到達できなかった場合にのみエラーテキストを表示していました。この更新により、プラグインでは適切なエラーメッセージと、到達できない LokiStack の詳しい修正方法が表示されるようになりました。(LOG-2874)

### 1.16.2. CVE

- CVE-2022-4304
- CVE-2022-4450
- CVE-2023-0215
- CVE-2023-0286
- CVE-2023-0361
- CVE-2023-23916

## 1.17. LOGGING 5.5.9

このリリースには、OpenShift Logging Bug Fix Release 5.5.9 が含まれています。

### 1.17.1. バグ修正

- この更新の前は、Fluentd コレクターの問題により、**/var/log/auth-server/audit.log** に保存されている OAuth ログインイベントがキャプチャーされませんでした。これにより、OAuth サービスからのログインイベントの収集が不完全になりました。今回の更新により、Fluentd コレクターは、予想どおり、**/var/log/auth-server/audit.log** に保存されているものを含め、OAuth サービスからすべてのログインイベントをキャプチャーすることで、この問題を解決するようになりました。(LOG-3730)

- この更新の前は、構造化された解析が有効になっていて、メッセージが複数の宛先に転送された場合に、それらはディープコピーされませんでした。これにより、構造化されたメッセージを含む一部の受信ログが生成されましたが、その他のログは生成されませんでした。今回の更新により、JSON 解析の前にメッセージをディープコピーするように設定生成が変更されました。その結果、複数の宛先に転送された場合でも、すべての受信ログに構造化メッセージが含まれるようになりました。(LOG-3767)

### 1.17.2. CVE

- CVE-2022-4304
- CVE-2022-4450
- CVE-2022-41717
- CVE-2023-0215
- CVE-2023-0286
- CVE-2023-0767
- CVE-2023-23916

## 1.18. LOGGING 5.5.8

このリリースには、OpenShift Logging Bug Fix Release 5.5.8 が含まれています。

### 1.18.1. バグ修正

- 今回の更新の前は、コレクターが **level** フィールドを設定する方法にエラーがあったため、**priority** フィールドが **systemd** ログから欠落していました。今回の更新により、これらのフィールドが正しく設定され、問題が解決されました。(LOG-3630)

### 1.18.2. CVE

- CVE-2020-10735
- CVE-2021-28861
- CVE-2022-2873
- CVE-2022-4415
- CVE-2022-24999
- CVE-2022-40897
- CVE-2022-41222
- CVE-2022-41717
- CVE-2022-43945
- CVE-2022-45061
- CVE-2022-48303

## 1.19. LOGGING 5.5.7

このリリースには、OpenShift Logging バグ修正リリース 5.5.7 が含まれます。

### 1.19.1. バグ修正

- この更新の前は、ブール式と組み合わせてラベルフィルターを使用した場合、LokiStack ゲートウェイラベルエンフォーサーが有効な LogQL クエリーの解析エラーを生成していました。今回の更新により、LokiStack LogQL の実装がブール式を使用したラベルフィルターをサポートするようになり、問題が解決されました。(LOG-3534)

- この更新の前は、**ClusterLogForwarder** カスタムリソース (CR) が syslog 出力の TLS 認証情報を Fluentd に渡さなかったため、転送中にエラーが発生していました。今回の更新により、認証情報が Fluentd に正しく渡されるようになり、問題が解決されました。(LOG-3533)

### 1.19.2. CVE

CVE-2021-46848CVE-2022-3821CVE-2022-35737CVE-2022-42010CVE-2022-42011CVE-2022-42012CVE-2022-42898CVE-2022-43680

## 1.20. LOGGING 5.5.6

このリリースには、OpenShift Logging バグ修正リリース 5.5.6 が含まれます。

### 1.20.1. バグ修正

- この更新の前は、Pod セキュリティーアドミッションコントローラーがラベル **podSecurityLabelSync = true** を **openshift-logging** namespace に追加していました。これにより、指定のセキュリティーラベルが上書きされ、その結果、コレクター Pod が起動しなくなりました。今回の更新により、ラベル **podSecurityLabelSync = false** がセキュリティーラベルを保持するようになります。コレクター Pod は期待どおりにデプロイされます。(LOG-3340)

- この更新の前は、コンソールビュープラグインがクラスターで有効になっていない場合でも、Operator はコンソールビュープラグインをインストールしていました。これにより、Operator がクラッシュしました。今回の更新により、クラスターのアカウントでコンソールビューが有効になっていない場合、Operator は正常に機能し、コンソールビューをインストールしなくなりました。(LOG-3407)

- この更新の前は、Elasticsearch デプロイメントのステータスが更新されないリグレッションに対応するための以前の修正が原因で、**Red Hat Elasticsearch Operator** がデプロイされていない限り、Operator がクラッシュしていました。今回の更新により、その修正が元に戻されたため、Operator は安定した状態になりました。ただし、報告されたステータスに関連する以前の問題が再び発生するようになりました。(LOG-3428)

- この更新の前は、Loki Operator は、選択されたスタックサイズに関係なく、LokiStack ゲートウェイのレプリカを 1 つだけデプロイしていました。今回の更新により、選択したサイズに応じてレプリカの数が正しく設定されるようになりました。(LOG-3478)

- この更新の前は、複数のラベルキーに同じ接頭辞があり、一部のキーにドットが含まれていると、Elasticsearch に書き込まれたレコードで障害が発生していました。今回の更新により、ラベルキーのドットがアンダースコアに置き換えられ、問題が解決されました。(LOG-3341)

- この更新の前は、ロギングビュープラグインに、OpenShift Container Platform の特定のバージョンと互換性のない機能が含まれていました。今回の更新で、プラグインの正しいリリースストリームにより、この問題は解決されます。(LOG-3467)

- この更新の前は、**ClusterLogForwarder** カスタムリソースの調整で、1 つ以上のパイプラインの低下ステータスが誤って報告され、コレクター Pod が 8 - 10 秒ごとに再起動していました。今回の更新により、**ClusterLogForwarder** カスタムリソースの調整が正しく処理されるようになり、問題が解決されました。(LOG-3469)

- この変更の前は、ClusterLogForwarder カスタムリソースの **outputDefaults** フィールドの仕様は、宣言されたすべての Elasticsearch 出力タイプに設定を適用していました。今回の変更により、拡張仕様に一致するように動作が修正され、デフォルトのマネージド Elasticsearch ストアにのみ設定が適用されるようになりました。(LOG-3342)

- この更新の前は、OpenShift CLI (oc) がそのキャッシュを構築するために書き込み権限を持つフォルダーを必要とするため、OpenShift CLI (oc) の **must-gather** スクリプトが完了しませんでした。今回の更新により、OpenShift CLI (oc) にフォルダーへの書き込み権限が付与され、**must-gather** スクリプトが正常に完了するようになりました。(LOG-3472)

- この更新の前は、Loki Operator Webhook サーバーが TLS エラーを引き起こしていました。今回の更新により、Loki Operator Webhook PKI は Operator Lifecycle Manager の動的 Webhook 管理によって管理されるようになり、問題が解決されました。(LOG-3511)

### 1.20.2. CVE

- CVE-2021-46848
- CVE-2022-2056
- CVE-2022-2057
- CVE-2022-2058
- CVE-2022-2519
- CVE-2022-2520
- CVE-2022-2521
- CVE-2022-2867
- CVE-2022-2868
- CVE-2022-2869
- CVE-2022-2953
- CVE-2022-2964
- CVE-2022-4139
- CVE-2022-35737
- CVE-2022-42010
- CVE-2022-42011
- CVE-2022-42012
- CVE-2022-42898

- CVE-2022-43680

## 1.21. LOGGING 5.5.5

このリリースには、OpenShift Logging バグ修正リリース 5.5.5 が含まれます。

### 1.21.1. バグ修正

- この更新の前は、Kibana の OAuth cookie の有効期限は **24h** に固定されていたため、**accessTokenInactivityTimeout** フィールドが **24h** 未満の値に設定されていると、Kibana で 401 エラーが発生していました。今回の更新により、Kibana の OAuth cookie の有効期限が **accessTokenInactivityTimeout** に同期され、デフォルト値は **24h** になります。(LOG-3305)

- この更新の前は、Vector は、JSON 解析が有効になっている場合に、**structuredTypeKey** または **structuredTypeName** の値も定義せずにメッセージフィールドを解析していました。今回の更新により、構造化ログを Elasticsearch に書き込むときに、**structuredTypeKey** または **structuredTypeName** のいずれかに値が必要になりました。(LOG-3284)

- この更新の前は、**FluentdQueueLengthIncreasing** アラート式から返された一連のラベルにカーディナリティの問題があった場合に、このアラートが発生しない可能性がありました。今回の更新により、アラートに必要なラベルのみが含まれるようにラベルが削減されました。(LOG-3226)

- この更新の前は、Loki は、切断されたクラスター内の外部ストレージへのアクセスをサポートしていませんでした。今回の更新では、これらの接続をサポートするために、プロキシー環境変数とプロキシーの信頼できる CA バンドルがコンテナーイメージに含まれています。(LOG-2860)

- 今回の更新前は、OpenShift Container Platform Web コンソールのユーザーは、Loki の CA 証明書を含む **ConfigMap** オブジェクトを選択できなかったため、Pod が CA なしで動作していました。今回の更新により、Web コンソールユーザーは設定マップを選択できるようになり、問題が解決されました。(LOG-3310)

- この更新の前に、CA キーは CA を Loki にマウントするためのボリューム名として使用されていたため、CA キーに非準拠の文字 (ドットなど) が含まれているとエラー状態が発生していました。今回の更新により、ボリューム名が内部文字列に標準化され、問題が解決されました。(LOG-3332)

### 1.21.2. CVE

- CVE-2016-3709
- CVE-2020-35525
- CVE-2020-35527
- CVE-2020-36516
- CVE-2020-36558
- CVE-2021-3640
- CVE-2021-30002
- CVE-2022-0168

- CVE-2022-0561
- CVE-2022-0562
- CVE-2022-0617
- CVE-2022-0854
- CVE-2022-0865
- CVE-2022-0891
- CVE-2022-0908
- CVE-2022-0909
- CVE-2022-0924
- CVE-2022-1016
- CVE-2022-1048
- CVE-2022-1055
- CVE-2022-1184
- CVE-2022-1292
- CVE-2022-1304
- CVE-2022-1355
- CVE-2022-1586
- CVE-2022-1785
- CVE-2022-1852
- CVE-2022-1897
- CVE-2022-1927
- CVE-2022-2068
- CVE-2022-2078
- CVE-2022-2097
- CVE-2022-2509
- CVE-2022-2586
- CVE-2022-2639
- CVE-2022-2938
- CVE-2022-3515

- CVE-2022-20368
- CVE-2022-21499
- CVE-2022-21618
- CVE-2022-21619
- CVE-2022-21624
- CVE-2022-21626
- CVE-2022-21628
- CVE-2022-22624
- CVE-2022-22628
- CVE-2022-22629
- CVE-2022-22662
- CVE-2022-22844
- CVE-2022-23960
- CVE-2022-24448
- CVE-2022-25255
- CVE-2022-26373
- CVE-2022-26700
- CVE-2022-26709
- CVE-2022-26710
- CVE-2022-26716
- CVE-2022-26717
- CVE-2022-26719
- CVE-2022-27404
- CVE-2022-27405
- CVE-2022-27406
- CVE-2022-27950
- CVE-2022-28390
- CVE-2022-28893
- CVE-2022-29581

- CVE-2022-30293
- CVE-2022-34903
- CVE-2022-36946
- CVE-2022-37434
- CVE-2022-39399

## 1.22. LOGGING 5.5.4

このリリースには、RHSA-2022:7434-OpenShift Logging バグ修正リリース 5.5.4 が含まれます。

### 1.22.1. バグ修正

- この更新の前は、ロギングビュープラグインのクエリーパーサーのエラーにより、クエリーに中かっこ **{}** が含まれていると、ログクエリーの一部が消えていました。これにより、クエリーが無効になり、有効なクエリーに対してエラーが返されました。今回の更新により、パーサーはこれらのクエリーを正しく処理するようになりました。(LOG-3042)
- この更新の前は、Elasticsearch または Kibana デプロイメントのステータスが変更されている間に、Operator がコレクターデーモンセットの削除と再作成のループに入る可能性がありました。今回の更新では、Operator のステータス処理が修正され、問題が解決されました。(LOG-3049)
- この更新の前は、Vector のコレクターの実装をサポートするためにアラートが実装されていませんでした。この変更により、Vector アラートが追加され、選択したコレクターの実装に応じて個別のアラートがデプロイメントされます。(LOG-3127)
- この更新の前は、Elasticsearch Operator のシークレット作成コンポーネントが内部シークレットを常に変更していました。今回の更新により、既存のシークレットが適切に処理されるようになりました。(LOG-3138)
- この更新の前に、ログ **must-gather** スクリプトの以前のリファクタリングにより、アーティファクトの予想される場所が削除されました。この更新により、アーティファクトを **/must-gather** フォルダーに書き込むという変更が元に戻ります。(LOG-3213)
- この更新の前は、特定のクラスターで、Prometheus エクスポーターが IPv6 ではなく IPv4 にバインドしていました。この更新後、Fluentd は IP バージョンを検出し、IPv4 の場合は **0.0.0.0**、IPv6 の場合は **[::]** にバインドします。(LOG-3162)

### 1.22.2. CVE

- CVE-2020-35525
- CVE-2020-35527
- CVE-2022-0494
- CVE-2022-1353
- CVE-2022-2509
- CVE-2022-2588

- CVE-2022-3515
- CVE-2022-21618
- CVE-2022-21619
- CVE-2022-21624
- CVE-2022-21626
- CVE-2022-21628
- CVE-2022-23816
- CVE-2022-23825
- CVE-2022-29900
- CVE-2022-29901
- CVE-2022-32149
- CVE-2022-37434
- CVE-2022-40674

## 1.23. LOGGING 5.5.3

このリリースには、OpenShift Logging バグ修正リリース 5.5.3 が含まれます。

### 1.23.1. バグ修正

- この更新前は、構造化されたメッセージを含むログエントリーに元のメッセージフィールドが含まれていたため、エントリーが大きくなりました。この更新により、構造化ログのメッセージフィールドが削除され、増加したサイズが縮小されます。(LOG-2759)
- この更新の前に、コレクター設定は、**Collector**、**default-log-store**、および **visualization** Pod からログを除外していましたが、**.gz** ファイルにアーカイブされたログを除外できませんでした。今回の更新により、**collector**、**default-log-store**、および **visualization** Pod の **.gz** ファイルとして保存されたアーカイブログも除外されます。(LOG-2844)
- この更新の前は、使用できない Pod へのリクエストがゲートウェイ経由で送信された場合、中断を警告するアラートはありませんでした。今回の更新により、ゲートウェイで書き込みまたは読み取り要求の完了に問題が発生した場合に、個別のアラートが生成されます。(LOG-2884)
- この更新の前は、値が参照によってパイプラインを通過したため、Pod メタデータは fluent プラグインによって変更される可能性がありました。この更新により、各ログメッセージが Pod メタデータのコピーを確実に受信するようになり、各メッセージが個別に処理されるようになりました。(LOG-3046)
- この更新の前に、OpenShift コンソールログビューで **不明な** 重大度を選択すると、**level=unknown** 値のログが除外されていました。今回の更新により、レベルのないログと **level=unknown** の値を持つログが、**不明な** 重大度でフィルタリングすると表示されるようになりました。(LOG-3062)

- この更新の前は、Elasticsearch に送信されたログレコードには、ログを送信する必要があるインデックスの名前を含む **write-index** という名前の追加フィールドがありました。このフィールドは、データモデルの一部ではありません。この更新後、このフィールドは送信されなくなりました。(LOG-3075)

- 新しい組み込み Pod Security Admission Controller の導入により、グローバルまたは namespace レベルで定義された強制セキュリティー規格に従って設定されていない Pod は実行できません。今回の更新により、Operator と Collector は特権実行を許可し、セキュリティー監査の警告やエラーなしで実行できるようになりました。(LOG-3077)

- この更新の前に、LokiStack をデフォルトのログストレージとして使用する場合、Operator は **ClusterLogForwarder** カスタムリソースで定義されたカスタム出力を削除しました。今回の更新により、Operator は **ClusterLogForwarder** カスタムリソースの処理時にカスタム出力をデフォルト出力とマージします。(LOG-3095)

### 1.23.2. CVE

- CVE-2015-20107
- CVE-2022-0391
- CVE-2022-2526
- CVE-2022-21123
- CVE-2022-21125
- CVE-2022-21166
- CVE-2022-29154
- CVE-2022-32206
- CVE-2022-32208
- CVE-2022-34903

## 1.24. LOGGING 5.5.2

このリリースには、OpenShift Logging バグ修正リリース 5.5.2 が含まれます。

### 1.24.1. バグ修正

- 今回の更新以前は、Fluentd コレクターのアラートルールは OpenShift Container Platform モニタリングスタイルのガイドラインに準拠していませんでした。今回の更新により、これらのアラートが namespace ラベルを含むように変更され、問題が解決されました。(LOG-1823)

- この更新の前は、インデックス名に複数のハイフン文字が含まれていると、インデックス管理ロールオーバースクリプトが新しいインデックス名を生成できませんでした。今回の更新により、インデックス名が正しく生成されるようになりました。(LOG-2644)

- この更新の前は、Kibana ルートは証明書が存在しない状態で **caCertificate** 値を設定していました。今回の更新により、**caCertificate** 値が設定されなくなりました。(LOG-2661)

- この更新の前は、コレクターの依存関係の変更により、未使用のパラメーターに対して警告メッセージが発行されていました。今回の更新で、未使用の設定パラメーターを削除すると、問題が解決されます。(LOG-2859)

- この更新の前に、Loki Operator が作成したデプロイメント用に作成された Pod は、Operator が実行されているクラスターでそのようなノードが利用可能な場合、Linux 以外のオペレーティングシステムのノードで誤ってスケジュールされていました。今回の更新により、Operator は追加のノードセレクターを Pod 定義に割り当て、Linux ベースのノードでのみ Pod をスケジュールできるようにします。(LOG-2895)

- この更新の前は、LokiStack ゲートウェイの LogQL パーサーの問題により、OpenShift コンソールのログビューはログを重大度でフィルタリングしませんでした。今回の更新では、パーサーの修正により問題が解決され、OpenShift コンソールのログビューは重大度でフィルタリングできるようになりました。(LOG-2908)

- この更新の前に、Fluentd コレクタープラグインのリファクタリングにより、イベントのタイムスタンプフィールドが削除されました。この更新により、イベントの受信時刻をソースとするタイムスタンプフィールドが復元されます。(LOG-2923)

- この更新の前は、監査ログに **level** フィールドがないため、ベクターログでエラーが発生していました。今回の更新で、監査ログレコードに **level** フィールドが追加され、問題が解決されました。(LOG-2961)

- 今回の更新の前は、Kibana カスタムリソースを削除した場合、OpenShift Container Platform Web コンソールは引き続き Kibana へのリンクを表示していました。今回の更新で、Kibana カスタムリソースを削除すると、そのリンクも削除されます。(LOG-3053)

- この更新の前は、**ClusterLogForwarder** カスタムリソースに JSON 解析が定義されている場合、各ロールオーバージョブで空のインデックスが作成されていました。今回の更新により、新しいインデックスは空ではなくなりました。(LOG-3063)

- この更新の前に、ユーザーが Loki Operator 5.5 への更新後に LokiStack を削除すると、もともと Loki Operator 5.4 によって作成されたリソースが残りました。今回の更新により、リソースの所有者参照は 5.5 LokiStack を指します。(LOG-2945)

- この更新の前は、ユーザーはアクセス権を持つ namespace のアプリケーションログを表示できませんでした。今回の更新により、Loki Operator はクラスターロールとクラスターロールバインディングを自動的に作成し、ユーザーがアプリケーションログを読み取れるようにします。(LOG-2918)

- この更新の前は、cluster-admin 権限を持つユーザーは、ログコンソールを使用してインフラストラクチャーと監査ログを適切に表示できませんでした。今回の更新により、認可チェックが拡張され、cluster-admin および dedicated-admin グループのユーザーも管理者として認識されるようになりました。(LOG-2970)

### 1.24.2. CVE

- CVE-2015-20107
- CVE-2022-0391
- CVE-2022-21123
- CVE-2022-21125
- CVE-2022-21166

- CVE-2022-29154
- CVE-2022-32206
- CVE-2022-32208
- CVE-2022-34903

## 1.25. LOGGING 5.5.1

このリリースには OpenShift Logging バグ修正リリース 5.5.1 が含まれます。

### 1.25.1. 機能強化

- 今回の機能拡張により、Logging Console プラグインが使用されている場合に、**Aggregated Logs** タブが OpenShift Container Platform Web コンソールの **Pod Details** ページに追加されました。この拡張機能は OpenShift Container Platform 4.10 以降でのみ利用できます。(LOG-2647)
- 今回の拡張機能により、ログ転送の出力オプションとして Google Cloud Logging が追加されました。(LOG-1482)

### 1.25.2. バグ修正

- 今回の更新以前は、Operator は Pod が準備状態にあることを確認しないことが原因で、クラスターの再起動時にクラスターが動作不能な状態になりました。今回の更新により、Operator は、再起動中に新しい Pod を準備完了としてマークしてから新しい Pod に移動を続けることで、問題を解決します。(LOG-2745)
- 今回の更新以前は、Fluentd は Kubernetes プラットフォームがログファイルをローテーションしたことを認識しない場合があり、ログメッセージを読み取らなくなっていました。今回の更新で、アップストリームの開発チームが提案する設定パラメーターを設定することにより修正されています。(LOG-2995)
- 今回の更新以前は、複数行のエラー検出機能が追加されたことが原因で、内部ルーティングが変更され、レコードが間違った宛先に転送されていました。今回の更新により、内部ルーティングが正しくなりました。(LOG-2801)
- 今回の更新前は、OpenShift Container Platform Web コンソールの更新間隔を変更すると、**Query** フィールドが空の場合にはエラーが発生していました。今回の更新で、**Query** フィールドが空の場合に、間隔の変更が選択不可能になりました。(LOG-2917)

### 1.25.3. CVE

- CVE-2022-1705
- CVE-2022-2526
- CVE-2022-29154
- CVE-2022-30631
- CVE-2022-32148
- CVE-2022-32206

- CVE-2022-32208

## 1.26. LOGGING 5.5

Logging 5.5 では、次のアドバイザリーを利用できます (リリース 5.5)。

### 1.26.1. 拡張機能

- 今回の更新では、構造化ログを同じ Pod 内の異なるコンテナーからさまざまなインデックスに転送できるようになりました。この機能を使用するには、複数コンテナーのサポートを使用してパイプラインを設定し、Pod にアノテーションを付ける必要があります。(LOG-1296)

**重要**

ログの JSON 形式は、アプリケーションによって異なります。作成するインデックスが多すぎるとパフォーマンスに影響するため、この機能の使用は、互換性のない JSON 形式のログのインデックスの作成に限定してください。クエリーを使用して、さまざまな namespace または互換性のある JSON 形式のアプリケーションからログを分離します。

- 今回の更新では、Kubernetes 共通ラベルである **app.kubernetes.io/component**、**app.kubernetes.io/managed-by**、**app.kubernetes.io/part-of**、および **app.kubernetes.io/version** を使用して、Elasticsearch 出力でログをフィルタリングできます。Elasticsearch 以外の出力タイプでは、**kubernetes.labels** に含まれるすべてのラベルを使用できます。(LOG-2388)

- 今回の更新では、AWS Security Token Service (STS) が有効になっているクラスターは、STS 認証を使用してログを Amazon CloudWatch に転送できます。(LOG-1976)

- 今回の更新では、'Loki Operator' Operator および Vector コレクターがテクニカルプレビュー機能から一般提供機能 (GA) に移行します。以前のリリースとの完全な機能パリティーに関しては作業中で、一部の API はテクニカルプレビュー機能のままです。詳細は、**LokiStack を使用したロギング** セクションを参照してください。

### 1.26.2. バグ修正

- この更新の前は、ログを Amazon CloudWatch に転送するように設定されたクラスターが、拒否されたログファイルを一時ストレージに書き込んでいたため、時間の経過とともにクラスターが不安定になりました。今回の更新により、すべてのストレージオプションの一括バックアップが無効になり、問題が解決されました。(LOG-2746)

- 今回の更新以前に、Operator は、非推奨で OpenShift Container Platform の今後のバージョンで削除予定の API のバージョンを一部使用していました。今回の更新により、依存関係がサポート対象の API バージョンに移動されます。(LOG-2656)

今回の更新以前に、Operator は、非推奨で OpenShift Container Platform の今後のバージョンで削除予定の API のバージョンを一部使用していました。今回の更新により、依存関係がサポート対象の API バージョンに移動されます。(LOG-2656)

- 今回の更新以前は、複数行エラー検出用に設定された複数の **ClusterLogForwarder** パイプラインにより、コレクターが **crashloopbackoff** エラー状態になりました。今回の更新により、複数の設定セクションに同じ一意の ID が使用される問題が修正されます。(LOG-2241)

- この更新の前は、コレクターは UTF-8 以外の記号を Elasticsearch ストレージログに保存できませんでした。今回の更新で、コレクターは UTF-8 以外の記号をエンコードし、問題を解決しました。(LOG-2203)
- 今回の更新以前は、ラテン文字以外の文字が Kibana で正しく表示されませんでした。今回の更新により、Kibana はすべての有効な UTF-8 シンボルを正しく表示します。(LOG-2784)

### 1.26.3. CVE

- CVE-2021-38561
- CVE-2022-1012
- CVE-2022-1292
- CVE-2022-1586
- CVE-2022-1785
- CVE-2022-1897
- CVE-2022-1927
- CVE-2022-2068
- CVE-2022-2097
- CVE-2022-21698
- CVE-2022-30631
- CVE-2022-32250

## 1.27. LOGGING 5.4.14

このリリースには、OpenShift Logging Bug Fix Release 5.4.14 が含まれています。

### 1.27.1. バグ修正

なし。

### 1.27.2. CVE

- CVE-2022-4304
- CVE-2022-4450
- CVE-2023-0215
- CVE-2023-0286
- CVE-2023-0361
- CVE-2023-23916

## 1.28. LOGGING 5.4.13

このリリースには、OpenShift Logging Bug Fix Release 5.4.13 が含まれています。

### 1.28.1. バグ修正

- この更新の前は、Fluentd コレクターの問題により、**/var/log/auth-server/audit.log** に保存されている OAuth ログインイベントがキャプチャーされませんでした。これにより、OAuth サービスからのログインイベントの収集が不完全になりました。今回の更新により、Fluentd コレクターは、予想どおり、**/var/log/auth-server/audit.log** に保存されているものを含め、OAuth サービスからすべてのログインイベントをキャプチャーすることで、この問題を解決するようになりました。(LOG-3731)

### 1.28.2. CVE

- CVE-2022-4304
- CVE-2022-4450
- CVE-2023-0215
- CVE-2023-0286
- CVE-2023-0767
- CVE-2023-23916

## 1.29. LOGGING 5.4.12

このリリースには、OpenShift Logging Bug Fix Release 5.4.12 が含まれています。

### 1.29.1. バグ修正

なし。

### 1.29.2. CVE

- CVE-2020-10735
- CVE-2021-28861
- CVE-2022-2873
- CVE-2022-4415
- CVE-2022-40897
- CVE-2022-41222
- CVE-2022-41717
- CVE-2022-43945
- CVE-2022-45061

- CVE-2022-48303

## 1.30. LOGGING 5.4.11

このリリースには、OpenShift Logging バグ修正リリース 5.4.11 が含まれます。

### 1.30.1. バグ修正

- BZ 2099524
- BZ 2161274

### 1.30.2. CVE

- CVE-2021-46848
- CVE-2022-3821
- CVE-2022-35737
- CVE-2022-42010
- CVE-2022-42011
- CVE-2022-42012
- CVE-2022-42898
- CVE-2022-43680

## 1.31. LOGGING 5.4.10

このリリースには、OpenShift Logging バグ修正リリース 5.4.10 が含まれます。

### 1.31.1. バグ修正

なし。

### 1.31.2. CVE

- CVE-2021-46848
- CVE-2022-2056
- CVE-2022-2057
- CVE-2022-2058
- CVE-2022-2519
- CVE-2022-2520
- CVE-2022-2521
- CVE-2022-2867

- CVE-2022-2868
- CVE-2022-2869
- CVE-2022-2953
- CVE-2022-2964
- CVE-2022-4139
- CVE-2022-35737
- CVE-2022-42010
- CVE-2022-42011
- CVE-2022-42012
- CVE-2022-42898
- CVE-2022-43680

## 1.32. LOGGING 5.4.9

このリリースには、OpenShift Logging バグ修正リリース 5.4.9 が含まれます。

### 1.32.1. バグ修正

- この更新の前は、Fluentd コレクターは未使用の設定パラメーターを警告していました。この更新により、これらの設定パラメーターとその警告メッセージが削除されます。(LOG-3074)
- この更新の前は、Kibana の OAuth cookie の有効期限は **24h** に固定されていたため、**accessTokenInactivityTimeout** フィールドが **24h** 未満の値に設定されていると、Kibana で 401 エラーが発生していました。今回の更新により、Kibana の OAuth cookie の有効期限が **accessTokenInactivityTimeout** に同期され、デフォルト値は **24h** になります。(LOG-3306)

### 1.32.2. CVE

- CVE-2016-3709
- CVE-2020-35525
- CVE-2020-35527
- CVE-2020-36516
- CVE-2020-36558
- CVE-2021-3640
- CVE-2021-30002
- CVE-2022-0168
- CVE-2022-0561

- CVE-2022-0562
- CVE-2022-0617
- CVE-2022-0854
- CVE-2022-0865
- CVE-2022-0891
- CVE-2022-0908
- CVE-2022-0909
- CVE-2022-0924
- CVE-2022-1016
- CVE-2022-1048
- CVE-2022-1055
- CVE-2022-1184
- CVE-2022-1292
- CVE-2022-1304
- CVE-2022-1355
- CVE-2022-1586
- CVE-2022-1785
- CVE-2022-1852
- CVE-2022-1897
- CVE-2022-1927
- CVE-2022-2068
- CVE-2022-2078
- CVE-2022-2097
- CVE-2022-2509
- CVE-2022-2586
- CVE-2022-2639
- CVE-2022-2938
- CVE-2022-3515
- CVE-2022-20368

- CVE-2022-21499
- CVE-2022-21618
- CVE-2022-21619
- CVE-2022-21624
- CVE-2022-21626
- CVE-2022-21628
- CVE-2022-22624
- CVE-2022-22628
- CVE-2022-22629
- CVE-2022-22662
- CVE-2022-22844
- CVE-2022-23960
- CVE-2022-24448
- CVE-2022-25255
- CVE-2022-26373
- CVE-2022-26700
- CVE-2022-26709
- CVE-2022-26710
- CVE-2022-26716
- CVE-2022-26717
- CVE-2022-26719
- CVE-2022-27404
- CVE-2022-27405
- CVE-2022-27406
- CVE-2022-27950
- CVE-2022-28390
- CVE-2022-28893
- CVE-2022-29581
- CVE-2022-30293

- CVE-2022-34903
- CVE-2022-36946
- CVE-2022-37434
- CVE-2022-39399

## 1.33. LOGGING 5.4.8

このリリースには、RHSA-2022:7435-OpenShift Logging バグ修正リリース 5.4.8 が含まれます。

### 1.33.1. バグ修正

なし。

### 1.33.2. CVE

- CVE-2016-3709
- CVE-2020-35525
- CVE-2020-35527
- CVE-2020-36518
- CVE-2022-1304
- CVE-2022-2509
- CVE-2022-3515
- CVE-2022-22624
- CVE-2022-22628
- CVE-2022-22629
- CVE-2022-22662
- CVE-2022-26700
- CVE-2022-26709
- CVE-2022-26710
- CVE-2022-26716
- CVE-2022-26717
- CVE-2022-26719
- CVE-2022-30293
- CVE-2022-32149

- CVE-2022-37434
- CVE-2022-40674
- CVE-2022-42003
- CVE-2022-42004

## 1.34. LOGGING 5.4.6

このリリースには、OpenShift Logging バグ修正リリース 5.4.6 が含まれます。

### 1.34.1. バグ修正

- 今回の更新以前は、Fluentd は Kubernetes プラットフォームがログファイルをローテーションしたことを認識しない場合があり、ログメッセージを読み取らなくなっていました。今回の更新で、アップストリームの開発チームが提案する設定パラメーターを設定することにより修正されています。(LOG-2792)
- この更新の前は、**ClusterLogForwarder** カスタムリソースに JSON 解析が定義されている場合、各ロールオーバージョブで空のインデックスが作成されていました。今回の更新により、新しいインデックスは空ではなくなりました。(LOG-2823)
- 今回の更新の前は、Kibana カスタムリソースを削除した場合、OpenShift Container Platform Web コンソールは引き続き Kibana へのリンクを表示していました。今回の更新で、Kibana カスタムリソースを削除すると、そのリンクも削除されます。(LOG-3054)

### 1.34.2. CVE

- CVE-2015-20107
- CVE-2022-0391
- CVE-2022-21123
- CVE-2022-21125
- CVE-2022-21166
- CVE-2022-29154
- CVE-2022-32206
- CVE-2022-32208
- CVE-2022-34903

## 1.35. LOGGING 5.4.5

このリリースには、RHSA-2022:6183-OpenShift Logging バグ修正リリース 5.4.5 が含まれます。

### 1.35.1. バグ修正

- 今回の更新以前は、Operator は Pod が準備状態にあることを確認しないことが原因で、クラスターの再起動時にクラスターが動作不能な状態になりました。今回の更新により、Operator

は、再起動中に新しい Pod を準備完了としてマークしてから新しい Pod に移動を続けることで、問題を解決します。(LOG-2881)

- 今回の更新以前は、複数行のエラー検出機能が追加されたことが原因で、内部ルーティングが変更され、レコードが間違った宛先に転送されていました。今回の更新により、内部ルーティングが正しくなりました。(LOG-2946)
- 今回の更新以前は、Operator は引用符で囲まれたブール値でインデックス設定 JSON 応答をデコードできず、エラーが発生していました。今回の更新により、Operator はこの JSON 応答を適切にデコードできるようになりました。(LOG-3009)
- この更新の前に、Elasticsearch インデックステンプレートは、ラベルのフィールドを間違ったタイプで定義していました。この変更により、これらのテンプレートが更新され、ログコレクターによって転送されると予想されるタイプと同じになりました。(LOG-2972)

### 1.35.2. CVE

- CVE-2022-1292
- CVE-2022-1586
- CVE-2022-1785
- CVE-2022-1897
- CVE-2022-1927
- CVE-2022-2068
- CVE-2022-2097
- CVE-2022-30631

## 1.36. LOGGING 5.4.4

このリリースには、RHBA-2022:5907-OpenShift Logging バグ修正リリース 5.4.4 が含まれます。

### 1.36.1. バグ修正

- 今回の更新以前は、ラテン文字以外の文字が Elasticsearch で正しく表示されませんでした。今回の更新により、Elasticsearch はすべての有効な UTF-8 シンボルを正しく表示します。(LOG-2794)
- 今回の更新以前は、ラテン文字以外の文字が Fluentd で正しく表示されませんでした。今回の更新により、Fluentd はすべての有効な UTF-8 シンボルを正しく表示します。(LOG-2657)
- この更新以前には、コレクターのメトリックサーバーは、環境値によって公開された値を使用してアドレスにバインドしようとしました。今回の変更により、使用可能なインターフェイスであればどれでもバインドするように設定が変更されます。(LOG-2821)
- 今回の更新以前は、**cluster-logging** Operator はクラスターに依存してシークレットを作成していました。このクラスターの動作は OpenShift Container Platform 4.11 で変更されたことが原因で、ロギングデプロイメントに失敗しました。今回の更新により、**cluster-logging** Operator は必要に応じてシークレットを作成することで問題を解決します。(LOG-2840)

### 1.36.2. CVE

- CVE-2022-21540
- CVE-2022-21541
- CVE-2022-34169

## 1.37. LOGGING 5.4.3

このリリースには、RHSA-2022:5556-OpenShift Logging Bug Fix Release 5.4.3 が含まれます。

### 1.37.1. Elasticsearch Operator の非推奨通知

logging サブシステム 5.4.3 では、Elasticsearch Operator は非推奨となり、今後のリリースで削除される予定です。Red Hat は、この機能に対して現在のリリースライフサイクル中にバグ修正とサポートを提供しますが、拡張機能の提供はなく、この機能は今後削除される予定です。Elasticsearch Operator を使用してデフォルトのログストレージを管理する代わりに、Loki Operator を使用できます。

### 1.37.2. バグ修正

- この更新の前は、OpenShift ロギングダッシュボードは、アクティブなすべてのシャードではなく、アクティブなプライマリーシャードの数を表示していました。今回の更新により、ダッシュボードにはすべてのアクティブなシャードが表示されます。(LOG-2781)
- この更新に、**elasticsearch-operator** が使用するライブラリーのバグには、DoS 攻撃の脆弱性が含まれていました。今回の更新により、ライブラリーがこの脆弱性を含まないバージョンに更新されました。(LOG-2816)
- 今回の更新以前は、ログを Loki に転送するように Vector を設定するときに、Loki で TLS が有効になっていると、カスタムベアラートークンを設定したり、デフォルトトークンを使用したりできませんでした。今回の更新により、Vector は TLS が有効なトークンを使用してログを Loki に転送できるようになりました。(LOG-2786
- 今回の更新以前に、ElasticSearch Operator は、**oauth-proxy** イメージを選択するときに、**ImageStream** カスタムリソースの **referencePolicy** プロパティーを省略していました。このようにプロパティーが省略されることが原因で特定の環境で Kibana のデプロイに失敗しました。今回の更新では、**referencePolicy** を使用することで問題が解決され、Operator は Kibana を正常にデプロイできるようになりました。(LOG-2791)
- 今回の更新以前は、**ClusterLogForwarder** カスタムリソースのアラートルールで、複数の転送出力が考慮されていませんでした。今回の更新で問題が解決されました。(LOG-2640)
- この更新の前は、ログを Amazon CloudWatch に転送するように設定されたクラスターが、拒否されたログファイルを一時ストレージに書き込んでいたため、時間の経過とともにクラスターが不安定になりました。今回の更新により、CloudWatch のチャンクバックアップが無効になり、問題が解決されました。(LOG-2768)

### 1.37.3. CVE

> **例1.1 クリックして CVE を展開**
>
> - CVE-2020-28915
> - CVE-2021-40528

- CVE-2022-1271
- CVE-2022-1621
- CVE-2022-1629
- CVE-2022-22576
- CVE-2022-25313
- CVE-2022-25314
- CVE-2022-26691
- CVE-2022-27666
- CVE-2022-27774
- CVE-2022-27776
- CVE-2022-27782
- CVE-2022-29824

## 1.38. LOGGING 5.4.2

このリリースには、RHBA-2022:4874-OpenShift Logging バグ修正リリース 5.4.2 が含まれます。

### 1.38.1. バグ修正

- 今回の更新以前は、空白の使用に一貫性がなかったため、**oc edit** を使用したコレクター設定の編集が困難でした。今回の変更により、Operator による更新の前に設定を正規化およびフォーマットするロジックが導入され、**oc edit** を使用して簡単に編集できるようになりました。(LOG-2319)
- 今回の更新以前は、**FluentdNodeDown** アラートは、メッセージセクションにインスタンスラベルを適切に提供できませんでした。今回の更新では、部分的なインスタンスエラーの場合にインスタンスラベルを提供するようにアラートルールを修正することで、問題を解決します。(LOG-2607)
- 今回の更新以前は、`critical` などの複数のログレベルで、ドキュメントにはサポート対象と記載されているにも拘らず、サポートされていませんでした。今回の更新により不一致が修正され、記載されているログレベルが製品でサポートされるようになりました。(LOG-2033)

### 1.38.2. CVE

**例1.2 クリックして CVE を展開**

- CVE-2018-25032
- CVE-2020-0404
- CVE-2020-4788
- CVE-2020-13974

- CVE-2020-19131
- CVE-2020-27820
- CVE-2021-0941
- CVE-2021-3612
- CVE-2021-3634
- CVE-2021-3669
- CVE-2021-3737
- CVE-2021-3743
- CVE-2021-3744
- CVE-2021-3752
- CVE-2021-3759
- CVE-2021-3764
- CVE-2021-3772
- CVE-2021-3773
- CVE-2021-4002
- CVE-2021-4037
- CVE-2021-4083
- CVE-2021-4157
- CVE-2021-4189
- CVE-2021-4197
- CVE-2021-4203
- CVE-2021-20322
- CVE-2021-21781
- CVE-2021-23222
- CVE-2021-26401
- CVE-2021-29154
- CVE-2021-37159
- CVE-2021-41617
- CVE-2021-41864

- CVE-2021-42739
- CVE-2021-43056
- CVE-2021-43389
- CVE-2021-43976
- CVE-2021-44733
- CVE-2021-45485
- CVE-2021-45486
- CVE-2022-0001
- CVE-2022-0002
- CVE-2022-0286
- CVE-2022-0322
- CVE-2022-1011
- CVE-2022-1271

## 1.39. LOGGING 5.4.1

このリリースには、RHSA-2022:2216-OpenShift Logging Bug Fix Release 5.4.1 が含まれます。

### 1.39.1. バグ修正

- この更新の前は、ログファイルメトリックエクスポーターは、エクスポーターの実行中に作成されたログのみを報告したため、ログの増加データが不正確になりました。この更新では、**/var/log/pods** をモニターすることでこの問題を解決しています。(LOG-2442)
- この更新の前は、コレクターは、ログを fluentd の転送レシーバーに転送するときに古い接続を継続的に使用しようとしたため、ブロックされていました。このリリースでは、**keepalive_timeout** 値が 30 秒 (**30s**) に設定されているため、コレクターは接続をリサイクルし、適切な時間内に失敗したメッセージの送信を再試行します。(LOG-2534)
- この更新の前は、ログを読み取るためのテナンシーを強制するゲートウェイコンポーネントのエラーにより、Kubernetes namespace を持つログへのアクセスが制限され、監査ログと一部のインフラストラクチャーログが読み取れなくなることがありました。この更新により、プロキシーは管理者アクセス権を持つユーザーを正しく検出し、namespace なしでログへのアクセスを許可します。(LOG-2448)
- 今回の更新の前は、**system:serviceaccount:openshift-monitoring:prometheus-k8s** サービスアカウントには、**clusterrole** および **clusterrolebinding** としてクラスターレベルの特権がありました。今回の更新により、ロールとロールバインディングを持つ **openshift-logging** namespace にサービスアカウントが制限されます。(LOG-2437)
- この更新の前は、Linux 監査ログの時間解析は、キーと値のペアの順序に依存していました。この更新により、時間エントリーを見つけるために正規表現を使用するように解析が変更されます。(LOG-2321)

### 1.39.2. CVE

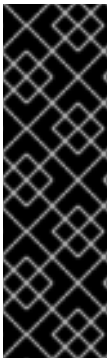
**例1.3 クリックして CVE を展開**

- CVE-2018-25032
- CVE-2021-4028
- CVE-2021-37136
- CVE-2021-37137
- CVE-2021-43797
- CVE-2022-0778
- CVE-2022-1154
- CVE-2022-1271
- CVE-2022-21426
- CVE-2022-21434
- CVE-2022-21443
- CVE-2022-21476
- CVE-2022-21496
- CVE-2022-21698
- CVE-2022-25636

## 1.40. LOGGING 5.4

以下のアドバイザリーは、Logging 5.4 に使用できます Logging subsystem for Red Hat OpenShift Release 5.4

### 1.40.1. テクノロジープレビュー



**重要**

Vector はテクノロジープレビュー機能です。テクノロジープレビュー機能は、Red Hat 製品のサービスレベルアグリーメント (SLA) の対象外であり、機能的に完全ではないことがあります。Red Hat は、実稼働環境でこれらを使用することを推奨していません。テクノロジープレビュー機能は、最新の製品機能をいち早く提供して、開発段階で機能のテストを行いフィードバックを提供していただくことを目的としています。

Red Hat のテクノロジープレビュー機能のサポート範囲に関する詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

### 1.40.2. Vector について

Vector は、ロギングサブシステムの現在のデフォルトコレクターに代わるテクノロジープレビューとして提供されるログコレクターです。

次の出力がサポートされています。

- **elasticsearch**。外部 Elasticsearch インスタンス。**elasticsearch** 出力では、TLS 接続を使用できます。
- **kafka**。Kafka ブローカー。**kafka** 出力は、セキュリティーで保護されていない接続または TLS 接続を使用できます。
- **loki**。Loki: 水平方向にスケーラブルで可用性の高いマルチテナントログ集計システム。

#### 1.40.2.1. Vector の有効化

Vector はデフォルトでは有効になっていません。以下のステップを使用して、OpenShift Container Platform クラスターで Vector を有効にします。

**重要**

Vector は、FIPS 対応クラスターをサポートしていません。

**前提条件**

- OpenShift Container Platform: 4.10
- Red Hat OpenShift のロギングサブシステム: 5.4
- FIPS が無効

**手順**

1. **openshift-logging** プロジェクトで **ClusterLogging** カスタムリソース (CR) を編集します。

```
$ oc -n openshift-logging edit ClusterLogging instance
```

2. **logging.openshift.io/preview-vector-collector: enabled** アノテーションを **ClusterLogging** カスタムリソース (CR) に追加します。
3. **ClusterLogging** カスタムリソース (CR) にコレクションタイプとして **vector** を追加します。

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
  namespace: "openshift-logging"
  annotations:
    logging.openshift.io/preview-vector-collector: enabled
spec:
  collection:
    logs:
      type: "vector"
      vector: {}
```

**関連情報**

- Vector ドキュメント

**重要**

Loki Operator はテクノロジープレビュー機能のみです。テクノロジープレビュー機能は、Red Hat 製品のサービスレベルアグリーメント (SLA) の対象外であり、機能的に完全ではないことがあります。Red Hat は、実稼働環境でこれらを使用することを推奨していません。テクノロジープレビュー機能は、最新の製品機能をいち早く提供して、開発段階で機能のテストを行いフィードバックを提供していただくことを目的としています。

Red Hat のテクノロジープレビュー機能のサポート範囲に関する詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

### 1.40.3. Loki について

Loki は、水平方向にスケーラブルで可用性の高いマルチテナントログ集約システムであり、現在、ロギングサブシステムのログストアとして Elasticsearch の代替として提供されています。

**関連情報**

- Loki ドキュメント

#### 1.40.3.1. Lokistack のデプロイ

OpenShift Container Platform コンソールを使用して Loki Operator をインストールできます。

**前提条件**

- OpenShift Container Platform: 4.10
- Red Hat OpenShift のロギングサブシステム: 5.4

OpenShift Container PlatformWeb コンソールを使用して Loki Operator をインストールするには:

1. Loki Operator をインストールします。
   a. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
   b. 使用可能な Operator のリストから **Loki Operator** を選択し、**Install** をクリックします。
   c. **Installation Mode** で、**All namespaces on the cluster**を選択します。
   d. **Installed Namespace** で、**openshift-operators-redhat** を選択します。**openshift-operators-redhat** namespace を指定する必要があります。**openshift-operators** namespace には信頼されていないコミュニティー Operator が含まれる可能性があり、OpenShift Container Platform メトリックと同じ名前でメトリックを公開する可能性があるため、これによって競合が生じる可能性があります。
   e. **Enable operator recommended cluster monitoring on this namespace**を選択します。このオプションは、namespace オブジェクトに **openshift.io/cluster-monitoring: "true"** ラベルを設定します。クラスターモニタリングが **openshift-operators-redhat** namespace を収集できるように、このオプションを選択する必要があります。

f. **Approval Strategy** を選択します。

- **Automatic** ストラテジーにより、Operator Lifecycle Manager (OLM) は新規バージョンが利用可能になると Operator を自動的に更新できます。
- **Manual** ストラテジーには、Operator の更新を承認するための適切な認証情報を持つユーザーが必要です。

g. **Install** をクリックします。

h. Loki Operator がインストールされていることを確認します。**Operators → Installed Operators** ページにアクセスして、"Loki Operator" を探します。

i. **Status** が **Succeeded** であるすべてのプロジェクトに **Loki Operator** がリストされていることを確認します。

### 1.40.4. バグ修正

- 今回の更新以前は、**cluster-logging-operator** は、クラスタースコープのロールとバインディングを利用して、Prometheus サービスアカウントがメトリックをスクレープするための権限を確立していました。これらの権限は、コンソールインターフェイスを使用して Operator をデプロイする時にだけ作成され、コマンドラインから この Operator をデプロイする場合に欠落していました。この更新では、ロールとバインディングを namespace スコープにすることで問題を修正しています。(LOG-2286)

- この更新の前に、ダッシュボードの調整を修正するための変更により、namespace 全体のリソースに **ownerReferences** フィールドが導入されました。その結果、設定マップとダッシュボードの両方が namespace に作成されませんでした。今回の更新では、**ownerReferences** フィールドを削除すると問題が解決し、コンソールで OpenShift Logging ダッシュボードを使用できるようになります。(LOG-2163)

- 今回の更新以前は、**cluster-logging-operator** でダッシュボードなど、既存の Config Map と目的の Config Map が正しく比較されなかったため、メトリックダッシュボードへの変更がデプロイされていませんでした。この更新では、オブジェクトラベルに一意のハッシュ値を追加することで問題が解決します。(LOG-2071)

- この更新の前は、OpenShift Logging ダッシュボードは、過去 24 時間に収集された上位のコンテナーを表示するテーブルに pod と namespace を正しく表示しませんでした。今回の更新では、pod と namespace が正しく表示されます。(LOG-2069)

- この更新の前は、**ClusterLogForwarder** が **Elasticsearch OutputDefault** で設定されていて、Elasticsearch 出力に構造化キーがないと、生成された設定に認証用の誤った値が含まれていました。この更新では、使用されているシークレットと証明書が修正されます。(LOG-2056)

- この更新以前は、無効なメトリックへの参照が原因で、OpenShift Logging ダッシュボードに空の CPU グラフが表示されていました。この更新により、正しいデータポイントが選択され、問題が解決されました。(LOG-2026)

- 今回の更新以前は、Fluentd コンテナーイメージには、実行時に不要なビルダーツールが含まれていました。この更新により、これらのツールがイメージから削除されます。.(LOG-1927)

- この更新の前は、5.3 リリースでデプロイされたコレクターの名前が変更されたため、ロギングコレクターは **FluentdNodeDown** アラートを生成していました。この更新では、Prometheus アラートのジョブ名を修正することで問題を解決しています。(LOG-1918)

- この更新の前は、コンポーネント名の変更のリファクタリングのために、ログコレクターは独自のログを収集していました。これにより、コレクターが自身のログを処理する潜在的な

フィードバックループが発生し、メモリーとログメッセージのサイズの問題が発生する可能性があります。この更新では、コレクターログをコレクションから除外することで問題を解決しています。(LOG-1774)

- この更新の前では、PVC がすでに存在すると、**Unable to create PersistentVolumeClaim due to forbidden: exceeded quota: infra-storage-quota** というエラーを生成していました。この更新により、Elasticsearch は既存の PVC をチェックし、問題を解決します。(LOG-2131)
- この更新の前は、**elasticsearch-signing** シークレットが削除されたときに Elasticsearch は準備完了状態に戻ることができませんでした。この更新により、Elasticsearch は、そのシークレットが削除された後、準備完了状態に戻ることができます。(LOG-2171)
- この更新の前は、コレクターがコンテナーログを読み取るパスが変更されたため、コレクターは一部のレコードを間違ったインデックスに転送していました。この更新により、コレクターは正しい設定を使用して問題を解決するようになりました。(LOG-2160)
- この更新の前は、namespace のリストがヘッダーサイズの最大制限に達したため、namespace が多数あるクラスターにより、Elasticsearch はリクエストの処理を停止していました。この更新により、ヘッダーには namespace 名のリストのみが含まれるようになり、問題が解決されました。(LOG-1899)
- この更新の前は、**OpenShift Container Platform Logging** ダッシュボードには、Elasticsearch に x ノードがある場合の実際の値の x 倍のシャードの数が表示されていました。この問題は、出力は常に Elasticsearch クラスター全体に対するものであったにも拘らず、Elasticsearch Pod ごとにすべてのプライマリーシャードを出力し、その合計を計算していたために発生しました。この更新により、シャードの数が正しく計算されるようになりました。(LOG-2156)
- この更新の前は、シークレット **kibana** と **kibana-proxy** が手動で削除されていると、再作成されませんでした。この更新により、**elasticsearch-operator** はリソースを監視し、削除された場合は自動的に再作成します。(LOG-2250)
- この更新の前に、バッファーチャンクサイズを調整すると、コレクターは、チャンクサイズがイベントストリームのバイト制限を超えているという警告を生成する可能性がありました。この更新では、読み取り行の制限を調整して、問題を解決することもできます。(LOG-2379)
- 今回の更新以前には、OpenShift WebConsole のロギングコンソールリンクが ClusterLogging CR で削除されていませんでした。この更新により、CR を削除するか、Cluster Logging Operator をアンインストールするとリンクが削除されます。(LOG-2373)
- 今回の更新以前は、コンテナーログパスを変更すると、元のパスで設定された古いリリースでは、このコレクションメトリックは常にゼロになりました。この更新では、収集されたログに関するメトリックを公開するプラグインが、問題を解決するためにいずれかのパスからの読み取りをサポートします。(LOG-2462)

### 1.40.5. CVE

- CVE-2022-0759
  - BZ-2058404
- CVE-2022-21698
  - BZ-2045880

## 1.41. ロギング 5.3.14

このリリースには、OpenShift Logging バグ修正リリース 5.3.14 が含まれます。

### 1.41.1. バグ修正

- この更新の前に、**log-file-metrics-exporter** コンポーネントによって生成されたログファイルサイズマップは、削除されたファイルのエントリーを削除しなかったため、ファイルサイズとプロセスメモリーが増加していました。今回の更新により、ログファイルサイズマップに、削除されたファイルのエントリーが含まれなくなりました。(LOG-3293)

### 1.41.2. CVE

- CVE-2016-3709
- CVE-2020-35525
- CVE-2020-35527
- CVE-2020-36516
- CVE-2020-36558
- CVE-2021-3640
- CVE-2021-30002
- CVE-2022-0168
- CVE-2022-0561
- CVE-2022-0562
- CVE-2022-0617
- CVE-2022-0854
- CVE-2022-0865
- CVE-2022-0891
- CVE-2022-0908
- CVE-2022-0909
- CVE-2022-0924
- CVE-2022-1016
- CVE-2022-1048
- CVE-2022-1055
- CVE-2022-1184
- CVE-2022-1292
- CVE-2022-1304

- CVE-2022-1355
- CVE-2022-1586
- CVE-2022-1785
- CVE-2022-1852
- CVE-2022-1897
- CVE-2022-1927
- CVE-2022-2068
- CVE-2022-2078
- CVE-2022-2097
- CVE-2022-2509
- CVE-2022-2586
- CVE-2022-2639
- CVE-2022-2938
- CVE-2022-3515
- CVE-2022-20368
- CVE-2022-21499
- CVE-2022-21618
- CVE-2022-21619
- CVE-2022-21624
- CVE-2022-21626
- CVE-2022-21628
- CVE-2022-22624
- CVE-2022-22628
- CVE-2022-22629
- CVE-2022-22662
- CVE-2022-22844
- CVE-2022-23960
- CVE-2022-24448
- CVE-2022-25255

- CVE-2022-26373
- CVE-2022-26700
- CVE-2022-26709
- CVE-2022-26710
- CVE-2022-26716
- CVE-2022-26717
- CVE-2022-26719
- CVE-2022-27404
- CVE-2022-27405
- CVE-2022-27406
- CVE-2022-27950
- CVE-2022-28390
- CVE-2022-28893
- CVE-2022-29581
- CVE-2022-30293
- CVE-2022-34903
- CVE-2022-36946
- CVE-2022-37434
- CVE-2022-39399
- CVE-2022-42898

## 1.42. ロギング 5.3.13

このリリースには、RHSA-2022:68828-OpenShift Logging バグ修正リリース 5.3.13 が含まれます。

### 1.42.1. バグ修正

なし。

### 1.42.2. CVE

> **例1.4 クリックして CVE を展開**
>
> - CVE-2020-35525
> - CVE-2020-35527

- CVE-2022-0494
- CVE-2022-1353
- CVE-2022-2509
- CVE-2022-2588
- CVE-2022-3515
- CVE-2022-21618
- CVE-2022-21619
- CVE-2022-21624
- CVE-2022-21626
- CVE-2022-21628
- CVE-2022-23816
- CVE-2022-23825
- CVE-2022-29900
- CVE-2022-29901
- CVE-2022-32149
- CVE-2022-37434
- CVE-2022-39399
- CVE-2022-40674

## 1.43. LOGGING 5.3.12

このリリースには、OpenShift Logging バグ修正リリース 5.3.12 が含まれます。

### 1.43.1. バグ修正

なし。

### 1.43.2. CVE

- CVE-2015-20107
- CVE-2022-0391
- CVE-2022-21123
- CVE-2022-21125
- CVE-2022-21166

- CVE-2022-29154
- CVE-2022-32206
- CVE-2022-32208
- CVE-2022-34903

## 1.44. ロギング 5.3.11

このリリースには、OpenShift Logging バグ修正リリース 5.3.11 が含まれます。

### 1.44.1. バグ修正

- 今回の更新以前は、Operator は Pod が準備状態にあることを確認しないことが原因で、クラスターの再起動時にクラスターが動作不能な状態になりました。今回の更新により、Operator は、再起動中に新しい Pod を準備完了としてマークしてから新しい Pod に移動を続けることで、問題を解決します。(LOG-2871)

### 1.44.2. CVE

- CVE-2022-1292
- CVE-2022-1586
- CVE-2022-1785
- CVE-2022-1897
- CVE-2022-1927
- CVE-2022-2068
- CVE-2022-2097
- CVE-2022-30631

## 1.45. LOGGING 5.3.10

このリリースには、RHSA-2022:5908-OpenShift Logging バグ修正リリース 5.3.10 が含まれます。

### 1.45.1. バグ修正

- BZ-2100495

### 1.45.2. CVE

**例1.5 クリックして CVE を展開**

- CVE-2021-38561
- CVE-2021-40528
- CVE-2022-1271

- CVE-2022-1621
- CVE-2022-1629
- CVE-2022-21540
- CVE-2022-21541
- CVE-2022-22576
- CVE-2022-25313
- CVE-2022-25314
- CVE-2022-27774
- CVE-2022-27776
- CVE-2022-27782
- CVE-2022-29824
- CVE-2022-34169

## 1.46. LOGGING 5.3.9

このリリースには、RHBA-2022:5557-OpenShift Logging バグ修正リリース 5.3.9 が含まれます。

### 1.46.1. バグ修正

- 今回の更新以前に、ロギングコレクターには、生成されたメトリックのラベルとしてパスが含まれていました。このパスは頻繁に変更され、Prometheus サーバーのストレージが大幅に変更されました。今回の更新で、問題を解決し、ストレージの消費を減らすために、ラベルが削除されました。(LOG-2682)

### 1.46.2. CVE

**例1.6 クリックして CVE を展開**

- CVE-2020-28915
- CVE-2021-40528
- CVE-2022-1271
- CVE-2022-1621
- CVE-2022-1629
- CVE-2022-22576
- CVE-2022-25313
- CVE-2022-25314

- CVE-2022-26691
- CVE-2022-27666
- CVE-2022-27774
- CVE-2022-27776
- CVE-2022-27782
- CVE-2022-29824

## 1.47. LOGGING 5.3.8

このリリースには、RHBA-2022:5010-OpenShift Logging バグ修正リリース 5.3.8 が含まれます。

### 1.47.1. バグ修正

(なし。)

### 1.47.2. CVE

**例1.7 クリックして CVE を展開**

- CVE-2018-25032
- CVE-2020-0404
- CVE-2020-4788
- CVE-2020-13974
- CVE-2020-19131
- CVE-2020-27820
- CVE-2021-0941
- CVE-2021-3612
- CVE-2021-3634
- CVE-2021-3669
- CVE-2021-3737
- CVE-2021-3743
- CVE-2021-3744
- CVE-2021-3752
- CVE-2021-3759
- CVE-2021-3764

- CVE-2021-3772
- CVE-2021-3773
- CVE-2021-4002
- CVE-2021-4037
- CVE-2021-4083
- CVE-2021-4157
- CVE-2021-4189
- CVE-2021-4197
- CVE-2021-4203
- CVE-2021-20322
- CVE-2021-21781
- CVE-2021-23222
- CVE-2021-26401
- CVE-2021-29154
- CVE-2021-37159
- CVE-2021-41617
- CVE-2021-41864
- CVE-2021-42739
- CVE-2021-43056
- CVE-2021-43389
- CVE-2021-43976
- CVE-2021-44733
- CVE-2021-45485
- CVE-2021-45486
- CVE-2022-0001
- CVE-2022-0002
- CVE-2022-0286
- CVE-2022-0322
- CVE-2022-1011

- CVE-2022-1271

## 1.48. OPENSHIFT LOGGING 5.3.7

このリリースには、RHSA-2022:2217 OpenShift Logging Bug Fix Release 5.3.7 が含まれます。

### 1.48.1. バグ修正

- この更新の前は、Linux 監査ログの時間解析はキー/値ペアの順序に依存していました。この更新により、時間エントリーを見つけるために正規表現を利用するように解析が変更されます。(LOG-2322)
- この更新以前には、一部のログフォワーダー出力は同じタイムスタンプでログを並べ替えることができました。この更新により、タイムスタンプが一致するエントリーを注文するためのシーケンス番号がログレコードに追加されました。(LOG-2334)
- この更新の前は、namespace のリストがヘッダーサイズの最大制限に達したため、namespace が多数あるクラスターにより、Elasticsearch はリクエストの処理を停止していました。この更新により、ヘッダーには namespace 名のリストのみが含まれるようになり、問題が解決されました。(LOG-2450)
- この更新の前は、**system:serviceaccount:openshift-monitoring:prometheus-k8s** には、**clusterrole** および **clusterrolebinding** としてクラスターレベルの特権がありました。この更新により、**serviceaccount** がロールとロールバインディングを持つ **openshift-logging** namespace に制限されます。(LOG-2481))

### 1.48.2. CVE

**例1.8 クリックして CVE を展開**

- CVE-2018-25032
- CVE-2021-4028
- CVE-2021-37136
- CVE-2021-37137
- CVE-2021-43797
- CVE-2022-0759
- CVE-2022-0778
- CVE-2022-1154
- CVE-2022-1271
- CVE-2022-21426
- CVE-2022-21434
- CVE-2022-21443

- CVE-2022-21476
- CVE-2022-21496
- CVE-2022-21698
- CVE-2022-25636

## 1.49. OPENSHIFT LOGGING 5.3.6

このリリースには、RHBA-2022:1377 OpenShift Logging Bug Fix Release 5.3.6 が含まれます。

### 1.49.1. バグ修正

- この更新の前は、キーなしで許容値を定義し、既存の Operator が原因で、Operator はアップグレードを完了できませんでした。今回の更新により、この許容範囲によってアップグレードの完了が妨げられることはなくなりました。(LOG-2126)
- この変更の前は、コレクターがチャンクバイト制限が発行されたイベントを超えている場合に警告を生成することが可能でした。この変更により、アップストリームのドキュメントのアドバイスに従って、リードラインの制限を調整して問題を解決できます。(LOG-2380)

## 1.50. OPENSHIFT LOGGING 5.3.5

このリリースには、RHSA-2022:0721 OpenShift Logging Bug Fix Release 5.3.5 が含まれます。

### 1.50.1. バグ修正

- この更新の前は、OpenShift Container Platform から OpenShift Logging を削除した場合、Web コンソールは引き続き **Logging** ページへのリンクを表示していました。この更新では、OpenShift Logging を削除またはアンインストールするとそのリンクも削除されます。(LOG-2182)

### 1.50.2. CVE

**例1.9 クリックして CVE を展開**

- CVE-2020-28491
- CVE-2021-3521
- CVE-2021-3872
- CVE-2021-3984
- CVE-2021-4019
- CVE-2021-4122
- CVE-2021-4192
- CVE-2021-4193

- CVE-2022-0552

## 1.51. OPENSHIFT LOGGING 5.3.4

このリリースには、RHBA-2022:0411 OpenShift Logging Bug Fix Release 5.3.4 が含まれます。

### 1.51.1. バグ修正

- この更新以前は、**cluster-logging-operator** でダッシュボードを含む既存の設定マップと目的の設定マップが正しく比較されなかったため、メトリックダッシュボードへの変更がまだデプロイされていませんでした。この更新では、オブジェクトラベルに一意のハッシュ値を追加することにより、ロジックを修正します。(LOG-2066)

- この更新の前は、FIPS を有効にして更新した後、Elasticsearch Pod を起動できませんでした。この更新により、Elasticsearch Pod は正常に起動します。(LOG-1974)

- この更新の前では、PVC がすでに存在する場合、Unable to create PersistentVolumeClaim due to forbidden: exceeded quota: infra-storage-quota. というエラーを生成していました。この更新により、elasticsearch は既存の PVC をチェックし、問題を解決します。(LOG-2127)

### 1.51.2. CVE

**例1.10 クリックして CVE を展開**

- CVE-2021-3521
- CVE-2021-3872
- CVE-2021-3984
- CVE-2021-4019
- CVE-2021-4122
- CVE-2021-4155
- CVE-2021-4192
- CVE-2021-4193
- CVE-2022-0185
- CVE-2022-21248
- CVE-2022-21277
- CVE-2022-21282
- CVE-2022-21283
- CVE-2022-21291
- CVE-2022-21293
- CVE-2022-21294

- CVE-2022-21296
- CVE-2022-21299
- CVE-2022-21305
- CVE-2022-21340
- CVE-2022-21341
- CVE-2022-21360
- CVE-2022-21365
- CVE-2022-21366

## 1.52. OPENSHIFT LOGGING 5.3.3

このリリースには、RHSA-2022:0227 OpenShift Logging のバグ修正リリース 5.3.3 が含まれます。

### 1.52.1. バグ修正

- 今回の更新以前は、cluster-logging Operator でダッシュボードを含む既存の設定マップと目的の設定マップが正しく比較されなかったため、メトリックダッシュボードへの変更がまだデプロイされていませんでした。この更新では、ダッシュボードの一意のハッシュ値をオブジェクトラベルに追加することで、ロジックを修正しています。(LOG-2066)
- 今回の更新により、log4j の依存関係が 2.17.1 に変更され、 CVE-2021-44832 が解決されました。(LOG-2102)

### 1.52.2. CVE

**例1.11 クリックして CVE を展開**

- CVE-2021-27292
  - BZ-1940613
- CVE-2021-44832
  - BZ-2035951

## 1.53. OPENSHIFT LOGGING 5.3.2

このリリースには、RHSA-2022:0044 OpenShift Logging のバグ修正リリース 5.3.2 が含まれます。

### 1.53.1. バグ修正

- 今回の更新以前は、Elasticsearch は解析エラーが原因でイベントルーターからのログを拒否していました。今回の更新では、解析エラーを解決するようにデータモデルが変更されています。ただし、その結果、以前のインデックスが原因で Kibana 内で警告またはエラーが発生する可能性があります。**kubernetes.event.metadata.resourceVersion** フィールドが原因で、既存

のインデックスが削除または再インデックスされるまでエラーが発生します。このフィールドが Kibana で使用されていない場合は、エラーメッセージを無視できます。古いインデックスを削除する保持ポリシーがある場合は、このポリシーにより、最終的に古いインデックスが削除されてエラーメッセージを停止します。それ以外の場合は、手動でインデックスを再作成してエラーメッセージを停止します。(LOG-2087)

- 今回の更新以前は、OpenShift Logging Dashboard は、過去 24 時間に作成および収集されたコンテナーの上位を表示するテーブルに、間違った Pod の namespace を表示していました。今回の更新により、OpenShift Logging Dashboard に正しい Pod の namespace が表示されます。(LOG-2051)

- この更新の前では、**ClusterLogForwarder** カスタムリソース (CR) インスタンスの **outputDefaults.elasticsearch.structuredTypeKey** に構造化キーがなかった場合、CR は出力シークレットをデフォルトのログストアとの通信に使用されるデフォルトのシークレットに置き換えていました。今回の更新では、定義された出力シークレットが正しく使用されます。(LOG-2046)

### 1.53.2. CVE

**例1.12 クリックして CVE を展開**

- CVE-2020-36327
  - BZ-1958999
- CVE-2021-45105
  - BZ-2034067
- CVE-2021-3712
- CVE-2021-20321
- CVE-2021-42574

## 1.54. OPENSHIFT LOGGING 5.3.1

このリリースには、RHSA-2021:5129 OpenShift Logging のバグ修正リリース 5.3.1 が含まれます。

### 1.54.1. バグ修正

- 今回の更新以前は、Fluentd コンテナーイメージには、実行時に不要なビルダーツールが含まれていました。今回の更新により、これらのツールがイメージから削除されます。(LOG-1998)

- 今回の更新以前は、無効なメトリックへの参照が原因で、ログダッシュボードに空の CPU グラフが表示されていました。今回の更新により、ロギングダッシュボードに CPU グラフが正しく表示されます。(LOG-1925)

- 今回の更新以前は、Elasticsearch Prometheus エクスポータープラグインは、Elasticsearch ノードのパフォーマンスに影響を与える高コストのクエリーを使用してインデックスレベルのメトリックをコンパイルしていました。この更新で、パフォーマンスを向上させる、低コストのクエリーが実装されています。(LOG-1897)

### 1.54.2. CVE

**例1.13 クリックして CVE を展開**

- CVE-2021-21409
  - BZ-1944888
- CVE-2021-37136
  - BZ-2004133
- CVE-2021-37137
  - BZ-2004135
- CVE-2021-44228
  - BZ-2030932
- CVE-2018-25009
- CVE-2018-25010
- CVE-2018-25012
- CVE-2018-25013
- CVE-2018-25014
- CVE-2019-5827
- CVE-2019-13750
- CVE-2019-13751
- CVE-2019-17594
- CVE-2019-17595
- CVE-2019-18218
- CVE-2019-19603
- CVE-2019-20838
- CVE-2020-12762
- CVE-2020-13435
- CVE-2020-14145
- CVE-2020-14155
- CVE-2020-16135
- CVE-2020-17541
- CVE-2020-24370
- CVE-2020-35521

- CVE-2020-35522
- CVE-2020-35523
- CVE-2020-35524
- CVE-2020-36330
- CVE-2020-36331
- CVE-2020-36332
- CVE-2021-3200
- CVE-2021-3426
- CVE-2021-3445
- CVE-2021-3481
- CVE-2021-3572
- CVE-2021-3580
- CVE-2021-3712
- CVE-2021-3800
- CVE-2021-20231
- CVE-2021-20232
- CVE-2021-20266
- CVE-2021-20317
- CVE-2021-22876
- CVE-2021-22898
- CVE-2021-22925
- CVE-2021-27645
- CVE-2021-28153
- CVE-2021-31535
- CVE-2021-33560
- CVE-2021-33574
- CVE-2021-35942
- CVE-2021-36084
- CVE-2021-36085

- CVE-2021-36086
- CVE-2021-36087
- CVE-2021-42574
- CVE-2021-43267
- CVE-2021-43527
- CVE-2021-45046

## 1.55. OPENSHIFT LOGGING 5.3.0

このリリースには、RHSA-2021:4627 OpenShift Logging のバグ修正リリース 5.3.0 が含まれます。

### 1.55.1. 新機能および拡張機能

- この更新により、ログ転送の承認オプションが拡張されました。出力は、SASL、ユーザー名/パスワード、または TLS で設定できるようになりました。

### 1.55.2. バグ修正

- 今回の更新以前は、syslog プロトコルを使用してログを転送した場合に、ルビーハッシュでエンコードされたキーと値のペアをシリアル化して⇒文字を含めてタブを #11 に置き換えました。今回の更新では、問題が修正され、有効な JSON としてログメッセージが正しくシリアル化されます。(LOG-1494)
- 今回の更新以前は、複数行のエラー検出が有効な場合に、正しい Cloudwatch ストリームに転送されるように、アプリケーションログが正しく設定されていませんでした。(LOG-1939)
- 今回の更新以前は、デプロイされたコレクターの名前が 5.3 リリースで変更されたため、アラート fluentnodedown が生成されていました。(LOG-1918)
- 今回の更新以前は、1 つ前のリリース設定で発生したリグレッションが原因で、コレクターはシャットダウン前にバッファーされたメッセージをフラッシュしてコレクター Pod の終了と再起動を遅らせていました。今回の更新で、fluentd はシャットダウン時にバッファーをフラッシュしなくなり、問題が解決しました。(LOG-1735)
- 今回の更新以前では、1 つ前のリリースで発生したリグレッションが原因で、JSON メッセージの解析が意図的に無効になっていました。今回の更新により、JSON 解析が再度有効になりました。また、解析された JSON メッセージの level フィールドをもとに、または正規表現を使用してメッセージフィールドから一致を抽出することで、ログエントリー level を設定します。(LOG-1199)
- 今回の更新以前では、 **Cluster Logging** カスタムリソース (CR) は、必要なバッファースペースが使用できない場合でも、**total Limit Size** フィールドの値を **Fluentdtotal_limit_size** フィールドに適用していました。今回の更新により、CR は 2 つの **total Limit Size** または 'default'値の小さい方を **Fluentdtotal_limit_size** フィールドに適用し、問題が解決されました。(LOG-1776)

### 1.55.3. 既知の問題

- ログを外部 Elasticsearch サーバーに転送してから、ユーザー名とパスワードなどのパイプライ

ンシークレットで設定された値を変更する場合に、Fluentd フォワーダーは新規シークレットを読み込むにもかかわらず、以前の値を使用して外部 Elasticsearch サーバーに接続します。この問題は、現在 Red Hat OpenShift Logging Operator がコンテンツの変更についてシークレットを監視しないために発生します。(LOG-1652)
回避策として、シークレットを変更した場合は、以下を入力して Fluentd Pod を強制的に再デプロイできます。

```
$ oc delete pod -l component=collector
```

### 1.55.4. 非推奨および削除された機能

以前のリリースで利用可能であった一部の機能が非推奨になるか、削除されました。

非推奨の機能は依然として OpenShift Container Logging に含まれており、引き続きサポートされますが、この製品の今後のリリースで削除されるため、新規デプロイメントでの使用は推奨されません。

#### 1.55.4.1. 従来の Fluentd および従来の syslog メソッドを使用したログの転送は削除されました

OpenShift Logging 5.3 では、ログを Syslog および Fluentd に転送する従来の方法が削除されています。バグ修正とサポートは、OpenShift Logging5.2 ライフサイクルの終了まで提供されます。その後は、新たな拡張機能は行われません。

代わりに、次にリリースされるレガシー以外のメソッドを使用してください。

- Fluentd 転送プロトコルを使用したログの転送
- syslog プロトコルを使用したログの転送

#### 1.55.4.2. 従来の転送方法の設定メカニズムは削除されました

OpenShift Logging 5.3 では、ログ転送のレガシー設定メカニズムが削除されました。レガシー Fluentd メソッドとレガシー Syslog メソッドを使用してログを転送できません。代わりに、標準のログ転送方法を使用してください。

### 1.55.5. CVE

**例1.14 クリックして CVE を展開**

- CVE-2018-20673
- CVE-2018-25009
- CVE-2018-25010
- CVE-2018-25012
- CVE-2018-25013
- CVE-2018-25014
- CVE-2019-5827
- CVE-2019-13750
- CVE-2019-13751

- CVE-2019-14615
- CVE-2019-17594
- CVE-2019-17595
- CVE-2019-18218
- CVE-2019-19603
- CVE-2019-20838
- CVE-2020-0427
- CVE-2020-10001
- CVE-2020-12762
- CVE-2020-13435
- CVE-2020-14145
- CVE-2020-14155
- CVE-2020-16135
- CVE-2020-17541
- CVE-2020-24370
- CVE-2020-24502
- CVE-2020-24503
- CVE-2020-24504
- CVE-2020-24586
- CVE-2020-24587
- CVE-2020-24588
- CVE-2020-26139
- CVE-2020-26140
- CVE-2020-26141
- CVE-2020-26143
- CVE-2020-26144
- CVE-2020-26145
- CVE-2020-26146
- CVE-2020-26147

- CVE-2020-27777
- CVE-2020-29368
- CVE-2020-29660
- CVE-2020-35448
- CVE-2020-35521
- CVE-2020-35522
- CVE-2020-35523
- CVE-2020-35524
- CVE-2020-36158
- CVE-2020-36312
- CVE-2020-36330
- CVE-2020-36331
- CVE-2020-36332
- CVE-2020-36386
- CVE-2021-0129
- CVE-2021-3200
- CVE-2021-3348
- CVE-2021-3426
- CVE-2021-3445
- CVE-2021-3481
- CVE-2021-3487
- CVE-2021-3489
- CVE-2021-3564
- CVE-2021-3572
- CVE-2021-3573
- CVE-2021-3580
- CVE-2021-3600
- CVE-2021-3635
- CVE-2021-3659

- CVE-2021-3679
- CVE-2021-3732
- CVE-2021-3778
- CVE-2021-3796
- CVE-2021-3800
- CVE-2021-20194
- CVE-2021-20197
- CVE-2021-20231
- CVE-2021-20232
- CVE-2021-20239
- CVE-2021-20266
- CVE-2021-20284
- CVE-2021-22876
- CVE-2021-22898
- CVE-2021-22925
- CVE-2021-23133
- CVE-2021-23840
- CVE-2021-23841
- CVE-2021-27645
- CVE-2021-28153
- CVE-2021-28950
- CVE-2021-28971
- CVE-2021-29155
- ICVE-2021-29646
- CVE-2021-29650
- CVE-2021-31440
- CVE-2021-31535
- CVE-2021-31829
- CVE-2021-31916

- CVE-2021-33033
- CVE-2021-33194
- CVE-2021-33200
- CVE-2021-33560
- CVE-2021-33574
- CVE-2021-35942
- CVE-2021-36084
- CVE-2021-36085
- CVE-2021-36086
- CVE-2021-36087
- CVE-2021-42574

## 1.56. LOGGING 5.2.13

このリリースには、RHSA-2022:5909-OpenShift Logging バグ修正リリース 5.2.13 が含まれます。

### 1.56.1. バグ修正

- BZ-2100495

### 1.56.2. CVE

**例1.15 クリックして CVE を展開**

- CVE-2021-38561
- CVE-2021-40528
- CVE-2022-1271
- CVE-2022-1621
- CVE-2022-1629
- CVE-2022-21540
- CVE-2022-21541
- CVE-2022-22576
- CVE-2022-25313
- CVE-2022-25314
- CVE-2022-27774

- CVE-2022-27776
- CVE-2022-27782
- CVE-2022-29824
- CVE-2022-34169

## 1.57. LOGGING 5.2.12

このリリースには、RHBA-2022:5558-OpenShift Logging バグ修正リリース 5.2.12 が含まれます。

### 1.57.1. バグ修正

なし。

### 1.57.2. CVE

**例1.16 クリックして CVE を展開**

- CVE-2020-28915
- CVE-2021-40528
- CVE-2022-1271
- CVE-2022-1621
- CVE-2022-1629
- CVE-2022-22576
- CVE-2022-25313
- CVE-2022-25314
- CVE-2022-26691
- CVE-2022-27666
- CVE-2022-27774
- CVE-2022-27776
- CVE-2022-27782
- CVE-2022-29824

## 1.58. LOGGING 5.2.11

このリリースには、RHBA-2022:5012-OpenShift Logging バグ修正リリース 5.2.11 が含まれます。

### 1.58.1. バグ修正

- この更新の前は、CloudWatch 転送を実行するように設定されたクラスターが、拒否されたログファイルを一時ストレージに書き込んでいたため、時間の経過とともにクラスターが不安定になりました。今回の更新により、CloudWatch のチャンクバックアップが無効になり、問題が解決されました。(LOG-2635)

### 1.58.2. CVE

**例1.17 クリックして CVE を展開**

- CVE-2018-25032
- CVE-2020-0404
- CVE-2020-4788
- CVE-2020-13974
- CVE-2020-19131
- CVE-2020-27820
- CVE-2021-0941
- CVE-2021-3612
- CVE-2021-3634
- CVE-2021-3669
- CVE-2021-3737
- CVE-2021-3743
- CVE-2021-3744
- CVE-2021-3752
- CVE-2021-3759
- CVE-2021-3764
- CVE-2021-3772
- CVE-2021-3773
- CVE-2021-4002
- CVE-2021-4037
- CVE-2021-4083
- CVE-2021-4157
- CVE-2021-4189

- CVE-2021-4197
- CVE-2021-4203
- CVE-2021-20322
- CVE-2021-21781
- CVE-2021-23222
- CVE-2021-26401
- CVE-2021-29154
- CVE-2021-37159
- CVE-2021-41617
- CVE-2021-41864
- CVE-2021-42739
- CVE-2021-43056
- CVE-2021-43389
- CVE-2021-43976
- CVE-2021-44733
- CVE-2021-45485
- CVE-2021-45486
- CVE-2022-0001
- CVE-2022-0002
- CVE-2022-0286
- CVE-2022-0322
- CVE-2022-1011
- CVE-2022-1271

## 1.59. OPENSHIFT LOGGING 5.2.10

このリリースには、OpenShift Logging バグ修正リリース 5.2.10 が含まれています

### 1.59.1. バグ修正

- この更新の前では、一部のログフォワーダー出力は同じタイムスタンプでログを並べ替えることができました。この更新により、タイムスタンプが一致するエントリーを注文するためのシーケンス番号がログレコードに追加されました。(LOG-2335)

- この更新の前は、namespace のリストがヘッダーサイズの最大制限に達したため、namespace が多数あるクラスターにより、Elasticsearch はリクエストの処理を停止していました。この更新により、ヘッダーには namespace 名のリストのみが含まれるようになり、問題が解決されました。(LOG-2475)

- この更新の前は、**system:serviceaccount:openshift-monitoring:prometheus-k8s** には、**clusterrole** および **clusterrolebinding** としてクラスターレベルの特権がありました。この更新により、**serviceaccount** がロールとロールバインディングを持つ **openshift-logging** namespace に制限されます。(LOG-2480)

- この更新の前は、**cluster-logging-operator** は、クラスタースコープのロールとバインディングを利用して、Prometheus サービスアカウントがメトリックをスクレープするための権限を確立していました。これらの権限は、コンソールインターフェイスを使用して Operator をデプロイする時にだけ作成され、コマンドラインから この Operator をデプロイする場合に欠落していました。これにより、このロールとバインディング namespace のスコープが設定され、問題が修正されます。(LOG-1972)

### 1.59.2. CVE

**例1.18 クリックして CVE を展開**

- CVE-2018-25032
- CVE-2021-4028
- CVE-2021-37136
- CVE-2021-37137
- CVE-2021-43797
- CVE-2022-0778
- CVE-2022-1154
- CVE-2022-1271
- CVE-2022-21426
- CVE-2022-21434
- CVE-2022-21443
- CVE-2022-21476
- CVE-2022-21496
- CVE-2022-21698
- CVE-2022-25636

## 1.60. OPENSHIFT LOGGING 5.2.9

このリリースには、RHBA-2022:1375 OpenShift Logging Bug Fix Release 5.2.9 が含まれます。

### 1.60.1. バグ修正

- この更新の前は、キーなしで許容値を定義し、既存の Operator が原因で、Operator はアップグレードを完了できませんでした。今回の更新により、この許容範囲によってアップグレードの完了が妨げられることはなくなりました。(LOG-2304)

## 1.61. OPENSHIFT LOGGING 5.2.8

このリリースには、RHSA-2022:0728 OpenShift Logging Bug Fix Release 5.2.8 が含まれます。

### 1.61.1. バグ修正

- この更新の前は、OpenShift Container Platform から OpenShift Logging を削除した場合、Web コンソールは引き続き **Logging** ページへのリンクを表示していました。この更新では、OpenShift Logging を削除またはアンインストールするとそのリンクも削除されます。(LOG-2180)

### 1.61.2. CVE

**例1.19 クリックして CVE を展開**

- CVE-2020-28491
  - BZ-1930423
- CVE-2022-0552
  - BG-2052539

## 1.62. OPENSHIFT LOGGING 5.2.7

このリリースには、RHBA-2022:0478 OpenShift Logging Bug Fix Release 5.2.7 が含まれます。

### 1.62.1. バグ修正

- 今回の更新以前には、FIPS を有効にして更新した後、Elasticsearch Pod を起動できませんでした。この更新により、Elasticsearch Pod は正常に起動します。(LOG-2000)
- この更新の前では、永続ボリュームクレーム (PVC) がすでに存在する場合、Elasticsearch は Unable to create PersistentVolumeClaim due to forbidden: exceeded quota: infra-storage-quota. というエラーを生成しました。 この更新により、Elasticsearch は既存の PVC をチェックし、問題を解決します。(LOG-2118)

### 1.62.2. CVE

**例1.20 クリックして CVE を展開**

- CVE-2021-3521
- CVE-2021-3872
- CVE-2021-3984

- CVE-2021-4019
- CVE-2021-4122
- CVE-2021-4155
- CVE-2021-4192
- CVE-2021-4193
- CVE-2022-0185

## 1.63. OPENSHIFT LOGGING 5.2.6

このリリースには、RHSA-2022:0230 OpenShift Logging のバグ修正リリース 5.2.6 が含まれます。

### 1.63.1. バグ修正

- 今回の更新以前のリリースにはフィルターの変更が含まれておらず、Fluentd をクラッシュさせる原因となっていました。今回の更新で、欠落していたフィルターが修正されました。(LOG-2104)
- 今回の更新により、log4j の依存関係が 2.17.1 に変更され、 CVE-2021-44832 が解決されました。(LOG-2101)

### 1.63.2. CVE

**例1.21 クリックして CVE を展開**

- CVE-2021-27292
  - BZ-1940613
- CVE-2021-44832
  - BZ-2035951

## 1.64. OPENSHIFT LOGGING 5.2.5

このリリースには、RHSA-2022:0043 OpenShift Logging のバグ修正リリース 5.2.5 が含まれます。

### 1.64.1. バグ修正

- 今回の更新以前は、Elasticsearch は解析エラーが原因でイベントルーターからのログを拒否していました。今回の更新では、解析エラーを解決するようにデータモデルが変更されています。ただし、その結果、以前のインデックスが原因で Kibana 内で警告またはエラーが発生する可能性があります。**kubernetes.event.metadata.resourceVersion** フィールドが原因で、既存のインデックスが削除または再インデックスされるまでエラーが発生します。このフィールドが Kibana で使用されていない場合は、エラーメッセージを無視できます。古いインデックスを削除する保持ポリシーがある場合は、このポリシーにより、最終的に古いインデックスが削除されてエラーメッセージを停止します。それ以外の場合は、手動でインデックスを再作成してエラーメッセージを停止します。LOG-2087)

### 1.64.2. CVE

**例1.22 クリックして CVE を展開**

- CVE-2021-3712
- CVE-2021-20321
- CVE-2021-42574
- CVE-2021-45105

## 1.65. OPENSHIFT LOGGING 5.2.4

このリリースには、RHSA-2021:5127 OpenShift Logging のバグ修正リリース 5.2.4 が含まれます。

### 1.65.1. バグ修正

- 今回の更新以前は、syslog 経由で送信されたレコードは、ルビーハッシュエンコーディングのキーと値のペアをシリアル化して⇒文字を含め、タブを #11 に置き換えていました。今回の更新では、メッセージが適切な JSON として正しくシリアル化されます。(LOG-1775)
- 今回の更新以前は、Elasticsearch Prometheus エクスポータープラグインは、Elasticsearch ノードのパフォーマンスに影響を与える高コストのクエリーを使用してインデックスレベルのメトリックをコンパイルしていました。この更新で、パフォーマンスを向上させる、低コストのクエリーが実装されています。(LOG-1970)
- 今回の更新以前は、ログ転送が複数の出力で設定されている場合に、Elasticsearch がメッセージを拒否することがありました。これは、出力の 1 つを設定すると、メッセージの内容が 1 つのメッセージに変更されたために発生しました。今回の更新で、ログ転送は出力ごとにメッセージを複製するようになり、出力固有の処理が他の出力に影響を与えることはありません。(LOG-1824)

### 1.65.2. CVE

**例1.23 クリックして CVE を展開**

- CVE-2018-25009
- CVE-2018-25010
- CVE-2018-25012
- CVE-2018-25013
- CVE-2018-25014
- CVE-2019-5827
- CVE-2019-13750
- CVE-2019-13751
- CVE-2019-17594

- CVE-2019-17595
- CVE-2019-18218
- CVE-2019-19603
- CVE-2019-20838
- CVE-2020-12762
- CVE-2020-13435
- CVE-2020-14145
- CVE-2020-14155
- CVE-2020-16135
- CVE-2020-17541
- CVE-2020-24370
- CVE-2020-35521
- CVE-2020-35522
- CVE-2020-35523
- CVE-2020-35524
- CVE-2020-36330
- CVE-2020-36331
- CVE-2020-36332
- CVE-2021-3200
- CVE-2021-3426
- CVE-2021-3445
- CVE-2021-3481
- CVE-2021-3572
- CVE-2021-3580
- CVE-2021-3712
- CVE-2021-3800
- CVE-2021-20231
- CVE-2021-20232
- CVE-2021-20266

- CVE-2021-20317
- CVE-2021-21409
- CVE-2021-22876
- CVE-2021-22898
- CVE-2021-22925
- CVE-2021-27645
- CVE-2021-28153
- CVE-2021-31535
- CVE-2021-33560
- CVE-2021-33574
- CVE-2021-35942
- CVE-2021-36084
- CVE-2021-36085
- CVE-2021-36086
- CVE-2021-36087
- CVE-2021-37136
- CVE-2021-37137
- CVE-2021-42574
- CVE-2021-43267
- CVE-2021-43527
- CVE-2021-44228
- CVE-2021-45046

## 1.66. OPENSHIFT LOGGING 5.2.3

このリリースには、RHSA-2021:4032 OpenShift Logging のバグ修正リリース 5.2.3 が含まれます。

### 1.66.1. バグ修正

- 今回の更新以前は、一部のアラートに namespace ラベルが含まれていませんでした。このようにラベルが省略されていると、OpenShift Container Platform でアラートルールを作成するための OpenShift Monitoring Team のガイドラインに準拠しません。今回の更新では、

Elasticsearch Operator のすべてのアラートに namespace ラベルが含まれ、OpenShift Container Platform でアラートルールを作成するための全ガイドラインに準拠します。(LOG-1857)

- 今回の更新以前では、1 つ前のリリースで発生したリグレッションが原因で、JSON メッセージの解析が意図的に無効になっていました。今回の更新により、JSON 解析が再度有効になりました。また、解析された JSON メッセージの **level** フィールドをもとに、または正規表現を使用してメッセージフィールドから一致を抽出することで、ログエントリー **level** を設定します。(LOG-1759)

### 1.66.2. CVE

**例1.24 クリックして CVE を展開**

- CVE-2021-23369
  - BZ-1948761
- CVE-2021-23383
  - BZ-1956688
- CVE-2018-20673
- CVE-2019-5827
- CVE-2019-13750
- CVE-2019-13751
- CVE-2019-17594
- CVE-2019-17595
- CVE-2019-18218
- CVE-2019-19603
- CVE-2019-20838
- CVE-2020-12762
- CVE-2020-13435
- CVE-2020-14155
- CVE-2020-16135
- CVE-2020-24370
- CVE-2021-3200
- CVE-2021-3426
- CVE-2021-3445
- CVE-2021-3572

- CVE-2021-3580
- CVE-2021-3778
- CVE-2021-3796
- CVE-2021-3800
- CVE-2021-20231
- CVE-2021-20232
- CVE-2021-20266
- CVE-2021-22876
- CVE-2021-22898
- CVE-2021-22925
- CVE-2021-23840
- CVE-2021-23841
- CVE-2021-27645
- CVE-2021-28153
- CVE-2021-33560
- CVE-2021-33574
- CVE-2021-35942
- CVE-2021-36084
- CVE-2021-36085
- CVE-2021-36086
- CVE-2021-36087

## 1.67. OPENSHIFT LOGGING 5.2.2

このリリースには、RHBA-2021:3747 OpenShift Logging のバグ修正リリース 5.2.2 が含まれます。

### 1.67.1. バグ修正

- 今回の更新以前では、 **Cluster Logging** カスタムリソース (CR) は、必要なバッファースペースが使用できない場合でも、**total Limit Size** フィールドの値を **Fluentdtotal_limit_size** フィールドに適用していました。今回の更新により、CR は 2 つの **totalLimitSize** または 'default'値の小さい方を **total_limit_size** フィールドに適用し、問題が解決されました (LOG-1738)。
- 今回の更新以前は、1 つ前のリリース設定で発生したリグレッションが原因で、コレクターはシャットダウン前にバッファーされたメッセージをフラッシュしてコレクター Pod の終了と再

起動を遅らせていました。今回の更新で、fluentd はシャットダウン時にバッファーをフラッシュしなくなり、問題が解決しました。(LOG-1739)

- 今回の更新以前は、バンドルマニフェストの問題が原因で、OpenShift Container Platform 4.9 の OLM を使用して Elasticsearch Operator をインストールできませんでした。今回の更新では、バンドルマニフェストが修正され、4.9 でインストールとアップグレードが再度可能になりました。(LOG-1780)

### 1.67.2. CVE

**例1.25 クリックして CVE を展開**

- CVE-2020-25648
- CVE-2021-22922
- CVE-2021-22923
- CVE-2021-22924
- CVE-2021-36222
- CVE-2021-37576
- CVE-2021-37750
- CVE-2021-38201

## 1.68. OPENSHIFT LOGGING 5.2.1

このリリースには、RHBA-2021:3550 OpenShift Logging のバグ修正リリース 5.2.1 が含まれます。

### 1.68.1. バグ修正

- 今回の更新以前は、リリースパイプラインスクリプトの問題が原因で、 **olm.skip Range** フィールドの値が、現在のリリース番号を反映するのではなく、 **5.2.0** のまま変更されていませんでした。今回の更新 Deha、リリース番号の変更時にこのフィールドの値を更新するようにパイプラインスクリプトが修正されてます。(LOG-1743)

### 1.68.2. CVE

(なし)

## 1.69. OPENSHIFT LOGGING 5.2.0

このリリースには、RHBA-2021:3393 OpenShift Logging のバグ修正リリース 5.2.0 が含まれます。

### 1.69.1. 新機能および拡張機能

- 今回の更新では、アプリケーションおよびインフラストラクチャーを監視する Amazon CloudWatch にログデータを転送できるようになりました。詳細は、ログの Amazon CloudWatch への転送 を参照してください。(LOG-1173)

- 今回の更新では、Loki (ログデータを水平方向にスケーラブルで可用性の高いマルチテナントログ集約システム) に転送できます。詳細は、ログの Loki への転送 を参照してください。(LOG-684)

- 今回の更新では、Fluentd 転送プロトコルを使用して TLS で暗号化された接続でログデータを転送した場合に、パスワードで暗号化された秘密鍵ファイルを使用して、クラスターログフォワーダー設定でパスフレーズを指定できるようになりました。詳細は、Fluentd 転送プロトコルを使用したログの転送 を参照してください。(LOG-1525)

- 今回の拡張機能により、ユーザー名とパスワードを使用して外部 Elasticsearch インスタンスへのログ転送接続を認証できるようになりました。たとえば、サードパーティーが Elasticsearch インスタンスを操作するため、相互 TLS (mTLS) を使用できない場合に、HTTP または HTTPS を使用してユーザー名とパスワードを含むシークレットを設定できます。詳細は、外部 Elasticsearch インスタンスへのログの転送 を参照してください。(LOG-1022)

- 今回の更新では、OVN ネットワークポリシー監査ログを収集し、ロギングサーバーに転送できるようになりました。(LOG-1526)

- デフォルトで、OpenShift Container Platform 4.5 で導入されたデータモデルは、複数の異なる namespace からのログを 1 つの共通のインデックスに送ります。今回の変更により、ログを最も多く生成した namespace を確認することが難しくなります。
  本リリースは namespace メトリックを OpenShift Container Platform コンソールの **Logging** ダッシュボードに追加します。これらのメトリクスを使用すると、ログを生成する namespace、および指定のタイムスタンプで各 namespace が生成するログ数を確認できます。

  これらのメトリックを表示するには、OpenShift Container Platform Web コンソールで **Administrator** パースペクティブを開き、**Observe → Dashboards → Logging/Elasticsearch** に移動します。(LOG-1680)

- 現在のリリース、OpenShift Logging 5.2 は 2 つの新規メトリクスを有効にします。指定のタイムスタンプまたは期間については、個別のコンテナーで生成またはログに記録された合計ログと、コレクターで収集される合計ログを表示できます。これらのメトリックには namespace、Pod、およびコンテナー名でラベルが付けられるため、各 namespace および Pod が収集および生成されるログの数を確認できます。(LOG-1213)

### 1.69.2. バグ修正

- 今回の更新以前は、OpenShift Elasticsearch Operator がインデックス管理 cron ジョブの作成時に、 **POLICY_MAPPING** 環境変数を 2 回追加したことが原因で apiserver が重複を報告していました。今回の更新では、問題が修正され、**POLICY_MAPPING** 環境変数が cronjob ごとに 1 回だけ設定され、apiserver で重複が報告されなくなりました。(LOG-1130)

- 今回の更新以前は、Elasticsearch クラスターを一時停止してノード 0 個にしても、インデックス管理 cron ジョブは一時停止されなかったため、cron ジョブのバックオフが最大になりました。さらに Elasticsearch クラスターの一時停止を解除した後に、バックオフが最大限に達したことが原因で、これらの cron ジョブは停止したままでした。今回の更新プログラムは、cron ジョブとクラスターを一時停止することで問題を解決します。(LOG-1268)

- 今回の更新以前は、OpenShift Container Platform コンソールの**ロギング**ダッシュボードで、上位 10 のログ生成コンテナーのリストに chart namespace のラベルがなく、誤ったメトリック名 **fluentd_input_status_total_bytes_logged** が提供されていました。今回の更新では、チャートには namespace ラベルと正しいメトリック名 **log_logged_bytes_total** が表示されます。(LOG-1271)

- 今回の更新以前は、インデックス管理 cronjob がエラーで終了した場合に、エラー終了コード

が報告されず、そのジョブのステータスが complete と報告されていました。 今回の更新では、エラーで終了するインデックス管理 cron ジョブのエラー終了コードを報告することで問題を解決します。(LOG-1273)

- **priorityclasses.v1beta1.scheduling.k8s.io** は 1.22 で削除され、**priorityclasses.v1.scheduling.k8s.io** に置き換えられました ( **v1beta1** は **v1** に置き換えられました)。今回の更新以前は、 **v1beta1** がまだ存在していたため、 **priorityclasses** に対して **APIRemoved In Next Release In Use** アラートが生成されていました。今回の更新では、**v1beta1** を **v1** に置き換えることで問題を解決し、アラートが生成されなくなりました。(LOG-1385)

- 以前は、OpenShift Elasticsearch Operator と Red Hat OpenShift Logging Operator には、オフライン環境で実行できる Operator の OpenShift Container Platform Web コンソールリストへの表示に必要なアノテーションがありませんでした。今回の更新では、**operators.openshift.io/Infrastructure-features: '["Disconnected"]'**アノテーションが上記 2 つの Operator に追加され、オフライン環境で実行される Operator リストに表示されるようになります。(LOG-1420)

- 今回の更新以前は、Red Hat OpenShift Logging Operator Pod は、パフォーマンスが最適化されたシングルノードクラスターで顧客のワークロード向けに予約された CPU コアで、スケジュールされていました。今回の更新では、クラスターロギング Operator Pod は、正しい CPU コアでスケジュールされます。(LOG-1440)

- 今回の更新以前は、一部のログエントリーで認識されない UTF-8 バイトが含まれていたため、Elasticsearch はメッセージを拒否し、バッファリングされたペイロード全体をブロックしていました。今回の更新では、拒否されたペイロードが無効なログエントリーを削除して、残りのエントリーを再送信して問題を解決します。(LOG-1499)

- 今回の更新以前には、**kibana-proxy** Pod が **CrashLoopBackoff** 状態になり、**Invalid configuration: cookie_secret must be 16, 24, or 32 bytes to create an AES cipher when pass_access_token == true or cookie_refresh != 0, but is 29 bytes.** というメッセージをログに記録することがありました。正確な実際のバイト数は異なる場合があります。今回の更新では、Kibana セッションシークレットが正しく生成されるようになり、このエラーが原因で kibana プロキシー Pod が **CrashLoopBackoff** 状態にならなくなりました。(LOG-1446)

- 今回の更新以前は、AWS Cloud Watch Fluentd プラグインはすべてのログレベルで AWS API 呼び出しを Fluentd ログに記録し、OpenShift Container Platform ノードリソースを追加で消費していました。今回の更新では、AWS Cloud Watch Fluentd プラグインはデバッグおよびトレースログレベルだけで AWS API 呼び出しをログに記録します。このように、デフォルトの警告ログレベルでは、Fluentd は余分なノードリソースを消費しません。(LOG-1071)

- 今回の更新以前は、Elasticsearch OpenDistro セキュリティープラグインが原因でユーザーインデックスの移行に失敗していました。今回の更新では、新しいバージョンのプラグインを提供して問題を解決しています。これで、インデックスの移行はエラーなしで進行します。(LOG-1276)

- 今回の更新以前は、OpenShift Container Platform コンソールの**ロギング**ダッシュボードで、上位 10 個のログ生成コンテナーのリストにデータポイントがありませんでした。今回の更新では、問題が解決され、ダッシュボードにすべてのデータポイントが表示されます。(LOG-1353)

- 今回の更新以前は、 **chunkLimitSize** と **totalLimitSize** の値を調整して Fluentd ログフォワーダーのパフォーマンスを調整していた場合に、**Setting queued_chunks_limit_size for each buffer to** のメッセージにあるように、値が低すぎることが報告されます。今回の更新ではこの問題が修正され、このメッセージが正しい値を報告するようになっています。(LOG-1411)

- 今回の更新以前は、Kibana OpenDistro セキュリティープラグインが原因でユーザーインデックスの移行に失敗していました。今回の更新では、新しいバージョンのプラグインを提供して問題を解決しています。これで、インデックスの移行はエラーなしで進行します。(LOG-1558)

- 今回の更新以前は、namespace 入力フィルターを使用すると、その namespace のログが他の入力に表示されませんでした。今回の更新では、ログを受け入れることができるすべての入力に送信されます。(LOG-1570)

- 今回の更新以前は、**viaq/logerr** 依存関係のライセンスファイルがないことが原因で、ライセンスのスキャナーが成功せずに中断していました。今回の更新では、 **viaq/logerr** 依存関係は Apache2.0 でライセンスが提供され、ライセンススキャナーは正常に実行されます。(LOG-1590)

- 今回の更新以前は、**elasticsearch-operator-bundle** ビルドパイプラインにある **curator5** の誤った brew タグが原因で、ダミーの SHA1 に固定されたイメージがプルされていました。今回の更新では、ビルドパイプラインは **curator5** の **logging-curator5-rhel8** 参照を使用し、インデックス管理 cron ジョブが **registry.redhat.io** から正しいイメージをプルできるようにします。(LOG-1624)

- 今回の更新以前は、**Service Account** 権限の問題により、**no permissions for [indices:admin/aliases/get]** などのエラーが発生していました。今回の更新では、権限が修正されて問題が解決されています。(LOG-1657)

- 今回の更新以前は、Red Hat OpenShift Logging Operator のカスタムリソース定義 (CRD) に Loki 出力タイプがないため、アドミッションコントローラーが **ClusterLogForwarder** カスタムリソースオブジェクトを拒否していました。今回の更新では、CRD に出力タイプとして Loki が含まれるため、管理者は、ログを Loki サーバーに送信するように **ClusterLogForwarder** を設定できます。(LOG-1683)

- 今回の更新以前は、OpenShift Elasticsearch Operator で **serviceaccounts** が調整され、シークレットを含むサードパーティーが所有するフィールドが上書きされていました。この問題が原因で、シークレットが頻繁に再作成されるため、メモリーと CPU の使用率が急増しました。今回の更新で問題が解決されました。現在のリリースでは、OpenShift Elasticsearch Operator は、サードパーティーが所有するフィールドを上書きしません。(LOG-1714)

- 今回の更新以前には、**Cluster Logging** カスタムリソース (CR) 定義で、 **flush_interval** 値を指定したにも拘らず、**flush_mode** を **interval** に設定しなかった場合に、Red Hat OpenShift Logging Operator は Fluentd 設定を生成しました。ただし、Fluentd コレクターは実行時にエラーを生成しました。今回の更新では、Red Hat OpenShift Logging Operator は **ClusterLoggingCR** 定義を検証して、両方のフィールドが指定されている場合にのみ Fluentd 設定を生成します。(LOG-1723)

### 1.69.3. 既知の問題

- ログを外部 Elasticsearch サーバーに転送してから、ユーザー名とパスワードなどのパイプラインシークレットで設定された値を変更する場合に、Fluentd フォワーダーは新規シークレットを読み込むにもかかわらず、以前の値を使用して外部 Elasticsearch サーバーに接続します。この問題は、現在 Red Hat OpenShift Logging Operator がコンテンツの変更についてシークレットを監視しないために発生します。(LOG-1652)
  回避策として、シークレットを変更した場合は、以下を入力して Fluentd Pod を強制的に再デプロイできます。

```
$ oc delete pod -l component=collector
```

### 1.69.4. 非推奨および削除された機能

以前のリリースで利用可能であった一部の機能が非推奨になるか、削除されました。

非推奨の機能は依然として OpenShift Container Logging に含まれており、引き続きサポートされますが、この製品の今後のリリースで削除されるため、新規デプロイメントでの使用は推奨されません。

### 1.69.5. 従来の Fluentd および従来の syslog メソッドを使用したログの転送が非推奨に

OpenShift Container Platform 4.6 から現バージョンまで、以下のレガシーメソッドを使用したログの転送は非推奨になり、今後のリリースで削除される予定です。

- レガシー Fluentd メソッドを使用したログの転送
- レガシー syslog メソッドを使用したログの転送

代わりに、次にリリースされるレガシー以外のメソッドを使用してください。

- Fluentdfohttps://www.redhat.com/security/data/cve/CVE-2021-22922.htmlrward プロトコルを使用したログの転送
- syslog プロトコルを使用したログの転送

### 1.69.6. CVE

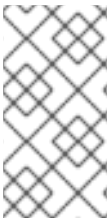
**例1.26 クリックして CVE を展開**

- CVE-2021-22922
- CVE-2021-22923
- CVE-2021-22924
- CVE-2021-32740
- CVE-2021-36222
- CVE-2021-37750

# 第2章 サポート

このドキュメントで説明されている設定オプションのみがログサブシステムでサポートされています。

他の設定オプションはサポートされていないため、使用しないでください。設定のパラダイムが OpenShift Container Platform リリース間で変更される可能性があり、このような変更は、設定のすべての可能性が制御されている場合のみ適切に対応できます。Operator は相違点を調整するように設計されているため、このドキュメントで説明されている以外の設定を使用すると、変更は上書きされます。



**注記**

OpenShift Container Platform ドキュメントで説明されていない設定を実行する必要がある場合は、Red Hat OpenShift Logging Operator を **Unmanaged** に設定する必要があります。マネージド外の OpenShift ロギング環境はサポートされておらず、OpenShift ロギングを **Managed** に戻すまで更新を受け取りません。

**注記**

Red Hat OpenShift のロギングサブシステムは、インストール可能なコンポーネントとして提供され、コアの OpenShift Container Platform とは異なるリリースサイクルを備えています。Red Hat OpenShift Container Platform ライフサイクルポリシー はリリースの互換性を概説しています。

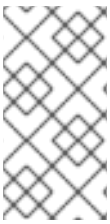
Red Hat OpenShift のロギングサブシステムは、アプリケーション、インフラストラクチャー、および監査ログの独自のコレクターおよびノーマライザーです。これは、サポートされているさまざまなシステムにログを転送するために使用することを目的としています。

Red Hat OpenShift のロギングサブシステムは次のとおりではありません。

- 大規模なログ収集システム
- セキュリティー情報およびイベント監視 (SIEM) に準拠
- 履歴または長期のログの保持または保管
- 保証されたログシンク
- 安全なストレージ - 監査ログはデフォルトでは保存されません

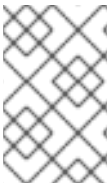
# 第3章 LOGGING 5.6

## 3.1. LOGGING 5.6 リリースノート



**注記**

Red Hat OpenShift のロギングサブシステムは、インストール可能なコンポーネントとして提供され、コアの OpenShift Container Platform とは異なるリリースサイクルを備えています。Red Hat OpenShift Container Platform ライフサイクルポリシー はリリースの互換性を概説しています。



**注記**

**stable** チャネルは、Logging の最新リリースを対象とする更新のみを提供します。以前のリリースの更新を引き続き受信するには、サブスクリプションチャネルを **stable-X** (**X** はインストールしたログのバージョン) に変更する必要があります。

### 3.1.1. Logging 5.6.11

このリリースには、OpenShift Logging バグ修正リリース 5.6.11 が含まれています。

#### 3.1.1.1. バグ修正

- 今回の更新が行われる前は、LokiStack ゲートウェイは承認されたリクエストを非常に広範囲にキャッシュしていました。その結果、誤った認証結果が発生しました。今回の更新により、LokiStack ゲートウェイは詳細にキャッシュを行うようになり、この問題が解決されました。(LOG-4435)

#### 3.1.1.2. CVE

- CVE-2023-3899
- CVE-2023-32360
- CVE-2023-34969

### 3.1.2. Logging 5.6.8

このリリースには、OpenShift Logging バグ修正リリース 5.6.8 が含まれています。

#### 3.1.2.1. バグ修正

- この更新が行われる前は、入力一致ラベル値の **ClusterLogForwarder** 内に / 文字が含まれる場合は、vector コレクターが予期せず終了していました。今回の更新では、一致ラベルを引用符で囲み、コレクターがログを開始および収集できるようにすることで問題が解決されました。(LOG-4091)
- この更新が行われる前は、OpenShift Container Platform Web コンソール内でログを表示する際に **more data available** オプションをクリックすると、初回クリック時にのみ、より多くのログエントリーがロードされました。今回の更新では、クリックごとにさらに多くのエントリーが読み込まれるようになりました。(OU-187)
- この更新が行われる前は、OpenShift Container Platform Web コンソール内でログを表示する

際に **streaming** オプションをクリックすると、実際のログは表示されず、**streaming logs** メッセージのみが表示されました。今回の更新により、メッセージとログストリームの両方が正しく表示されるようになりました。(OU-189)

- この更新が行われる前は、設定の問題が特定しにくい方法で Loki Operator がエラーをリセットしていました。今回の更新により、設定エラーが解決されるまでエラーが持続するようになりました。(LOG-4158)
- この更新が行われる前は、8,000 を超える namespace を持つクラスターの場合、namespace のリストが **http.max_header_size** 設定よりも大きくなるため Elasticsearch がクエリーを拒否していました。今回の更新では、ヘッダーサイズのデフォルト値が引き上げられ、問題が解決されました。(LOG-4278)

#### 3.1.2.2. CVE

- CVE-2020-24736
- CVE-2022-48281
- CVE-2023-1667
- CVE-2023-2283
- CVE-2023-24329
- CVE-2023-26604
- CVE-2023-28466

### 3.1.3. Logging 5.6.7

このリリースには、OpenShift Logging Bug Fix Release 5.6.7 が含まれています。

#### 3.1.3.1. バグ修正

- この更新が行われる前は、LokiStack ゲートウェイはユーザーのアクセス権を適用せずに namespace のラベル値を返していました。今回の更新により、LokiStack ゲートウェイはパーミッションをラベル値のリクエストに適用するようになり、問題は解決しました。(LOG-3728)
- この更新より前は、メッセージにタイムスタンプが含まれている場合、Fluentd ではログメッセージの **time** フィールドがデフォルトで **structured.time** として解析されませんでした。この更新により、出力先でサポートされている場合は、解析されたログメッセージに **structured.time** フィールドが含まれるようになります。(LOG-4090)
- この更新より前は、LokiStack ルート設定が原因で、クエリーの実行時間が 30 秒を超えるとタイムアウトが発生していました。今回の更新で、LokiStack global および per-tenant **queryTimeout** の設定によりルートタイムアウトの設定が影響を受け、問題は解決しました。(LOG-4130)
- この更新より前は、グローバル制限ではなくテナント制限に対して値が定義されている LokiStack CR により、Loki Operator がクラッシュしていました。今回の更新により、Operator はテナント制限のみが定義された LokiStack CR を処理できるようになり、問題が解決されました。(LOG-4199)
- この更新より前は、Web ブラウザーに保持されている以前のバージョンのキャッシュされたファイルが原因で、OpenShift Container Platform Web コンソールはアップグレード後にエ

ラーを生成しました。今回の更新により、これらのファイルはキャッシュされなくなり、問題は解決されました。(LOG-4099)

- この更新が行われる前は、Vector はデフォルトの Loki インスタンスに転送するときに証明書エラーを生成していました。この更新により、Vector を使用してログをエラーなしで Loki に転送できるようになりました。(LOG-4184)

- この更新が行われる前は、**tls.insecureSkipVerify** オプションが **true** に設定されている場合に、Cluster Logging Operator API にはシークレットにより提供される証明書が必要でした。今回の更新により、そのような場合でも Cluster Logging Operator API はシークレットに証明書の提供を求めなくなりました。次の設定が Operator の CR に追加されました。

```
tls.verify_certificate = false
tls.verify_hostname = false
```

(LOG-4146)

#### 3.1.3.2. CVE

- CVE-2021-26341
- CVE-2021-33655
- CVE-2021-33656
- CVE-2022-1462
- CVE-2022-1679
- CVE-2022-1789
- CVE-2022-2196
- CVE-2022-2663
- CVE-2022-3028
- CVE-2022-3239
- CVE-2022-3522
- CVE-2022-3524
- CVE-2022-3564
- CVE-2022-3566
- CVE-2022-3567
- CVE-2022-3619
- CVE-2022-3623
- CVE-2022-3625
- CVE-2022-3627

- CVE-2022-3628
- CVE-2022-3707
- CVE-2022-3970
- CVE-2022-4129
- CVE-2022-20141
- CVE-2022-25147
- CVE-2022-25265
- CVE-2022-30594
- CVE-2022-36227
- CVE-2022-39188
- CVE-2022-39189
- CVE-2022-41218
- CVE-2022-41674
- CVE-2022-42703
- CVE-2022-42720
- CVE-2022-42721
- CVE-2022-42722
- CVE-2022-43750
- CVE-2022-47929
- CVE-2023-0394
- CVE-2023-0461
- CVE-2023-1195
- CVE-2023-1582
- CVE-2023-2491
- CVE-2023-22490
- CVE-2023-23454
- CVE-2023-23946
- CVE-2023-25652
- CVE-2023-25815

- CVE-2023-27535
- CVE-2023-29007

### 3.1.4. Logging 5.6.6

このリリースには、OpenShift Logging Bug Fix Release 5.6.6 が含まれています。

#### 3.1.4.1. バグ修正

- この更新が行われるまでは、ペイロード内のキーに一致する Kafka 出力トピックに書き込むように **ClusterLogForwarder** カスタムリソースを設定すると、エラーによりメッセージのドロップが発生していました。今回の更新では、Fluentd のバッファー名の前にアンダースコアを付けることで問題が解決しました。(LOG-3458)
- この更新が行われるまでは、inode が再利用され、同じ inode を持つ複数のエントリーが存在する場合に、Fluentd でウォッチの早期終了が発生していました。この更新により、Fluentd 位置ファイル内のウォッチが早期に終了する問題が解決しました。(LOG-3629)
- この更新が行われるまでは、Fluentd による JavaScript クライアントの複数行例外の検出に失敗し、結果として例外が複数行として出力されていました。今回の更新により、例外が 1 行で出力されるようになり、問題が解決されました。(LOG-3761)
- この更新が行われるまでは、Red Hat Openshift Logging Operator バージョン 4.6 からバージョン 5.6 への直接アップグレードが許可されていたため、機能上の問題が発生していました。この更新により、アップグレードは 2 つのバージョン以内である必要があり、問題が解決されました。(LOG-3837)
- この更新が行われるまでは、Splunk または Google Logging 出力のメトリックは表示されませんでした。この更新では、HTTP エンドポイントのメトリックを送信することで問題が解決されました。(LOG-3932)
- この更新が行われるまでは、**ClusterLogForwarder** カスタムリソースが削除されても、コレクター Pod は実行されたままでした。この更新により、ログ転送が有効になっていない場合、コレクター Pod は実行されなくなります。(LOG-4030)
- この更新が行われるまでは、OpenShift Container Platform Web コンソールでログのヒストグラムをクリックしてドラッグしても時間範囲を選択できませんでした。今回の更新により、クリックとドラッグを使用して時間範囲を正常に選択できるようになりました。(LOG-4101)
- この更新が行われるまでは、監視ファイルの Fluentd ハッシュ値はログファイルへのパスを使用して生成されていたため、ログローテーション時に一意ではないハッシュが生成されました。今回の更新により、監視ファイルのハッシュ値が inode 番号で作成されるようになり、問題が解決されました。(LOG-3633)
- この更新が行われる前は、OpenShift Container Platform Web コンソールで **Show Resources** リンクをクリックしても何の効果もありませんでした。この更新では、ログエントリーごとにリソースの表示を切り替える **リソースの表示** リンクの機能を修正することで、この問題が解決されました。(LOG-4118)

#### 3.1.4.2. CVE

- CVE-2023-21930
- CVE-2023-21937

- CVE-2023-21938
- CVE-2023-21939
- CVE-2023-21954
- CVE-2023-21967
- CVE-2023-21968
- CVE-2023-28617

### 3.1.5. Logging 5.6.5

このリリースには、OpenShift Logging Bug Fix Release 5.6.5 が含まれています。

#### 3.1.5.1. バグ修正

- この更新前は、テンプレート定義により Elasticsearch が一部のラベルと namespace_label のインデックスを作成できず、データの取り込みで問題が発生していました。この更新では、ラベル内のドットとスラッシュが修正により置き換えられ、適切な取り込みが保証され、問題が効果的に解決されます。(LOG-3419)
- この更新より前は、OpenShift Web コンソールのログページが LokiStack への接続に失敗した場合、一般的なエラーメッセージが表示され、追加のコンテキストやトラブルシューティングの提案は提供されませんでした。この更新により、エラーメッセージが強化され、より具体的な詳細とトラブルシューティングの推奨事項が含まれるようになりました。(LOG-3750)
- この更新より前は、時間範囲形式が検証されていなかったため、カスタム日付範囲を選択するとエラーが発生していました。この更新により、時間形式が検証されるようになり、ユーザーが有効な範囲を選択できるようになりました。無効な時間範囲形式が選択された場合は、ユーザーにエラーメッセージが表示されます。(LOG-3583)
- この更新前は、Loki でログを検索すると、式の長さが 5120 文字を超えていなくても、多くの場合クエリーは失敗していました。この更新により、クエリー承認ラベルマッチャーが最適化され、問題が解決されました。(LOG-3480)
- この更新が行われる前は、Loki Operator は、プライベート IP のメンバーリストを使用する場合に、すべてのコンポーネントを見つけるのに十分なメンバーリスト設定を生成できませんでした。今回の更新により、生成された設定にアドバタイズされたポートが確実に含まれるようになり、すべてのコンポーネントを正常に検索できるようになりました。(LOG-4008)

#### 3.1.5.2. CVE

- CVE-2022-4269
- CVE-2022-4378
- CVE-2023-0266
- CVE-2023-0361
- CVE-2023-0386
- CVE-2023-27539

- CVE-2023-28120

### 3.1.6. Logging 5.6.4

このリリースには、OpenShift Logging Bug Fix Release 5.6.4 が含まれています。

#### 3.1.6.1. バグ修正

- この更新の前は、LokiStack がログストアとしてデプロイされたときに、Loki Pod によって生成されたログが収集され、LokiStack に送信されていました。今回の更新により、Loki によって生成されたログは収集から除外され、保存されなくなります。(LOG-3280)

- この更新の前は、OpenShift Web コンソールのログページのクエリーエディターが空の場合は、ドロップダウンメニューに値が入力されませんでした。今回の更新により、空のクエリーを実行しようとすると、エラーメッセージが表示され、ドロップダウンメニューが期待どおりに入力されるようになりました。(LOG-3454)

- この更新の前は、**tls.insecureSkipVerify** オプションが **true** に設定されている場合は、Cluster Logging Operator が誤った設定を生成していました。その結果、Operator は証明書の検証をスキップしようとすると、データを Elasticsearch に送信できませんでした。今回の更新により、**tls.insecureSkipVerify** が有効になっている場合でも、Cluster Logging Operator は正しい TLS 設定を生成します。その結果、証明書の検証をスキップしようとしても、データを Elasticsearch に正常に送信できます。(LOG-3475)

- この更新の前は、構造化された解析が有効になっていて、メッセージが複数の宛先に転送された場合に、それらはディープコピーされませんでした。これにより、構造化されたメッセージを含む一部の受信ログが生成されましたが、その他のログは生成されませんでした。今回の更新により、JSON 解析の前にメッセージをディープコピーするように設定生成が変更されました。その結果、複数の宛先に転送された場合でも、すべての受信メッセージに構造化メッセージが含まれるようになりました。(LOG-3640)

- この更新の前は、**collection** フィールドに **{}** が含まれていると、Operator がクラッシュする可能性がありました。今回の更新により、Operator はこの値を無視するようになり、オペレータは中断することなくスムーズに実行し続けることができます。(LOG-3733)

- この更新の前は、LokiStack のゲートウェイコンポーネントの **nodeSelector** 属性は効果がありませんでした。今回の更新により、**nodeSelector** 属性が期待どおりに機能するようになりました。(LOG-3783)

- この更新の前は、静的な LokiStack メンバーリストの設定は、プライベート IP ネットワークのみに依存していました。その結果、OpenShift Container Platform クラスター Pod ネットワークがパブリック IP 範囲で設定されている場合、LokiStack Pod がクラッシュループします。今回の更新により、LokiStack 管理者は、メンバーリストの設定に Pod ネットワークを使用するオプションを利用できるようになりました。これにより、問題が解決され、OpenShift Container Platform クラスター Pod ネットワークがパブリック IP 範囲で設定されている場合に、LokiStack Pod がクラッシュループ状態になるのを防ぐことができます。(LOG-3814)

- この更新の前は、**tls.insecureSkipVerify** フィールドが **true** に設定されている場合、Cluster Logging Operator は間違った設定を生成していました。その結果、証明書の検証をスキップしようとすると、Operator は Elasticsearch にデータを送信できませんでした。今回の更新により、**tls.insecureSkipVerify** が有効になっている場合でも、Operator は正しい TLS 設定を生成します。その結果、証明書の検証をスキップしようとしても、データを Elasticsearch に正常に送信できます。(LOG-3838)

- この更新の前に、Elasticsearch Operator を使用せずに Cluster Logging Operator (CLO) がインストールされた場合、CLO Pod は Elasticsearch の削除に関連するエラーメッセージを継続

的に表示していました。今回の更新により、CLO はエラーメッセージを表示する前に追加のチェックを実行するようになりました。その結果、Elasticsearch Operator が存在しない場合は、Elasticsearch の削除に関連するエラーメッセージが表示されなくなりました。(LOG-3763)

#### 3.1.6.2. CVE

- CVE-2022-4304
- CVE-2022-4450
- CVE-2023-0215
- CVE-2023-0286
- CVE-2023-0767
- CVE-2023-23916

### 3.1.7. Logging 5.6.3

このリリースには、OpenShift Logging Bug Fix Release 5.6.3 が含まれています。

#### 3.1.7.1. バグ修正

- 今回の更新の前は、Operator はゲートウェイテナントのシークレット情報を config map に保存していました。今回の更新により、Operator はこの情報をシークレットに保存します。(LOG-3717)
- 今回の更新の前は、Fluentd コレクターは **/var/log/auth-server/audit.log** に保存されている OAuth ログインイベントをキャプチャーしませんでした。今回の更新により、Fluentd は、これらの OAuth ログインイベントをキャプチャーし、問題を解決しました。(LOG-3729)

#### 3.1.7.2. CVE

- CVE-2020-10735
- CVE-2021-28861
- CVE-2022-2873
- CVE-2022-4415
- CVE-2022-40897
- CVE-2022-41222
- CVE-2022-43945
- CVE-2022-45061
- CVE-2022-48303

### 3.1.8. Logging 5.6.2

このリリースには、OpenShift Logging バグ修正リリース 5.6.2 が含まれます。

#### 3.1.8.1. バグ修正

- この更新の前は、コレクターは systemd ログの優先度に基づいて **level** フィールドを正しく設定しませんでした。今回の更新により、**level** フィールドが正しく設定されるようになりました。(LOG-3429)
- 今回の更新以前は、Operator は OpenShift Container Platform 4.12 以降で非互換性の警告を誤って生成していました。今回の更新により、Operator の OpenShift Container Platform の最大バージョン値が修正され、問題が解決されました。(LOG-3584)
- 今回の更新以前は、**default** の出力値で **ClusterLogForwarder** カスタムリソース (CR) を作成し、エラーを生成しませんでした。今回の更新により、この値が適切に生成されるというエラーの警告が表示されるようになりました。(LOG-3437)
- この更新の前は、**ClusterLogForwarder** カスタムリソース (CR) に 1 つの出力が **default** として設定された複数のパイプラインがある場合、コレクター Pod は再起動していました。今回の更新で、出力検証のロジックが修正され、問題が解決されました。(LOG-3559)
- この更新の前は、コレクター Pod は作成後に再起動されていました。今回の更新により、デプロイされたコレクターが自動的に再起動しなくなりました。(LOG-3608)
- 今回の更新以前は、パッチリリースがカタログから以前のバージョンの Operator を削除していました。これにより、古いバージョンをインストールできませんでした。今回の更新により、バンドル設定が変更され、同じマイナーバージョンの以前のリリースがカタログに留まるようになりました。(LOG-3635)

#### 3.1.8.2. CVE

- CVE-2022-23521
- CVE-2022-40303
- CVE-2022-40304
- CVE-2022-41903
- CVE-2022-47629
- CVE-2023-21835
- CVE-2023-21843

### 3.1.9. Logging 5.6.1

このリリースには、OpenShift Logging バグ修正リリース 5.6.1 が含まれます。

#### 3.1.9.1. バグ修正

- この更新の前は、保持が有効な場合、コンパクターは、クエリーアとの通信による TLS 証明書エラーを報告していました。今回の更新により、コンパクターとクエリーアが HTTP 経由で誤って通信することがなくなりました。(LOG-3494)
- この更新の前は、Loki Operator は **LokiStack** CR のステータスの設定を再試行しないため、ステータス情報が古くなっていました。今回の更新により、Operator は競合時にステータス情報の更新を再試行するようになりました。(LOG-3496)

- この更新の前は、**kube-apiserver-operator** Operator が Webhook の有効性を確認したときに、Loki Operator Webhook サーバーが TLS エラーを引き起こしていました。今回の更新により、Loki Operator Webhook PKI は Operator Lifecycle Manager (OLM) によって管理されるようになり、問題が解決されました。(LOG-3510)

- この更新の前は、ブール式と組み合わせてラベルフィルターを使用した場合、LokiStack ゲートウェイラベルエンフォーサーが有効な LogQL クエリーの解析エラーを生成していました。今回の更新により、LokiStack LogQL の実装がブール式を使用したラベルフィルターをサポートするようになり、問題が解決されました。(LOG-3441)、(LOG-3397)

- この更新の前は、複数のラベルキーに同じ接頭辞があり、一部のキーにドットが含まれていると、Elasticsearch に書き込まれたレコードで障害が発生していました。今回の更新により、ラベルキーのドットがアンダースコアに置き換えられ、問題が解決されました。(LOG-3463)

- この更新の前は、OpenShift Container Platform コンソールと logging-view-plugin の間に互換性がないため、**Red Hat OpenShift Logging** Operator は OpenShift Container Platform 4.10 クラスターで使用できませんでした。今回の更新により、プラグインは OpenShift Container Platform 4.10 管理コンソールと適切に統合されるようになりました。(LOG-3447)

- この更新の前は、**ClusterLogForwarder** カスタムリソースの調整で、デフォルトログストアを参照するパイプラインの低下ステータスが誤って報告されていました。今回の更新により、パイプラインが適切に検証されるようになりました (LOG-3477)。

#### 3.1.9.2. CVE

- CVE-2021-46848
- CVE-2022-3821
- CVE-2022-35737
- CVE-2022-42010
- CVE-2022-42011
- CVE-2022-42012
- CVE-2022-42898
- CVE-2022-43680
- CVE-2021-35065
- CVE-2022-46175

### 3.1.10. Logging 5.6.0

このリリースには、OpenShift Logging Release 5.6 が含まれています。

#### 3.1.10.1. 非推奨の通知

Logging バージョン 5.6 では、Fluentd は非推奨であり、今後のリリースで削除される予定です。Red Hat は、この機能に対して現在のリリースライフサイクル中にバグ修正とサポートを提供しますが、拡張機能の提供はなく、この機能は今後削除される予定です。Fluentd の代わりに、Vector を使用できます。

#### 3.1.10.2. 機能拡張

- 今回の更新により、Logging は OpenShift Container Platform のクラスター全体の暗号化ポリシーに準拠します。(LOG-895)
- 今回の更新により、LokiStack カスタムリソースを使用して、テナントごと、ストリームごと、およびグローバルポリシーの保持ポリシーを優先度順に宣言できるようになります。(LOG-2695)
- 今回の更新により、Splunk がログ転送の出力オプションとして利用できるようになります。(LOG-2913)
- 今回の更新により、デフォルトコレクターが Fluentd から Vector になります。(LOG-2222)
- 今回の更新により、**Developer** ロールは、OpenShift Container Platform 4.11 以降を実行しているクラスターの Log Console Plugin 内で割り当てられているプロジェクトごとのワークロードログにアクセスできます。(LOG-3388)
- 今回の更新により、任意のソースからのログに、Operator がデプロイされているクラスターの一意識別子であるフィールド **openshift.cluster_id** が含まれるようになります。**clusterID** の値は、以下のコマンドで表示できます。(LOG-2715)

```
$ oc get clusterversion/version -o jsonpath='{.spec.clusterID}{"\n"}'
```

#### 3.1.10.3. 既知の問題

- この更新の前は、複数のラベルキーに同じ接頭辞があり、一部のキーに **.** が含まれていると、Elasticsearch がログを拒否することがありました。これは、ラベルキーに含まれる **.** を **_** に置き換えることで、Elasticsearch の制限を修正します。この問題の回避策として、エラーの原因となっているラベルを削除するか、namespace をラベルに追加します。(LOG-3463)

#### 3.1.10.4. バグ修正

- 今回の更新の前は、Kibana カスタムリソースを削除した場合、OpenShift Container Platform Web コンソールは引き続き Kibana へのリンクを表示していました。今回の更新で、Kibana カスタムリソースを削除すると、そのリンクも削除されます。(LOG-2993)
- この更新の前は、ユーザーはアクセス権を持つ namespace のアプリケーションログを表示できませんでした。今回の更新により、Loki Operator はクラスターロールとクラスターロールバインディングを自動的に作成し、ユーザーがアプリケーションログを読み取れるようにします。(LOG-3072)
- この更新の前に、LokiStack をデフォルトのログストレージとして使用する場合、Operator は **ClusterLogForwarder** カスタムリソースで定義されたカスタム出力を削除しました。今回の更新により、Operator は **ClusterLogForwarder** カスタムリソースの処理時にカスタム出力をデフォルト出力とマージします。(LOG-3090)
- この更新の前に、CA キーは CA を Loki にマウントするためのボリューム名として使用されていたため、CA キーに非準拠の文字 (ドットなど) が含まれているとエラー状態が発生していました。今回の更新により、ボリューム名が内部文字列に標準化され、問題が解決されました。(LOG-3331)
- この更新の前は、LokiStack カスタムリソース定義内で設定されたデフォルト値が原因で、**1** の **ReplicationFactor** なしで LokiStack インスタンスを作成できませんでした。今回の更新により、Operator は使用されるサイズの実際の値を設定するようになります。(LOG-3296)

- この更新の前は、Vector は、JSON 解析が有効になっている場合に、**structuredTypeKey** または **structuredTypeName** の値も定義せずにメッセージフィールドを解析していました。今回の更新により、構造化ログを Elasticsearch に書き込むときに、**structuredTypeKey** または **structuredTypeName** のいずれかに値が必要になりました。(LOG-3195)

- この更新の前は、Elasticsearch Operator のシークレット作成コンポーネントが内部シークレットを常に変更していました。今回の更新により、既存のシークレットが適切に処理されるようになりました。(LOG-3161)

- この更新の前は、Elasticsearch または Kibana デプロイメントのステータスが変更されている間に、Operator がコレクターデーモンセットの削除と再作成のループに入る可能性がありました。今回の更新では、Operator のステータス処理が修正され、問題が解決されました。(LOG-3157)

- この更新の前は、Kibana の OAuth cookie の有効期限は **24h** に固定されていたため、**accessTokenInactivityTimeout** フィールドが **24h** 未満の値に設定されていると、Kibana で 401 エラーが発生していました。今回の更新により、Kibana の OAuth cookie の有効期限が **accessTokenInactivityTimeout** に同期され、デフォルト値は **24h** になります。(LOG-3129)

- この更新の前は、リソースを調整するための Operator の一般的なパターンとして、取得または更新を試みる前に作成を試みていました。そのため、作成後に一定の HTTP 409 レスポンスが発生していました。今回の更新により、Operator は最初にオブジェクトの取得を試み、オブジェクトが欠落しているか指定されていない場合にのみ作成または更新するようになります。(LOG-2919)

- この更新の前は、Fluentd の **.level** フィールドと .structure.level フィールドに異なる値が含まれることがありました。今回の更新により、各フィールドの値が同じになります。(LOG-2819)

- この更新の前は、Operator は信頼された CA バンドルの作成を待たず、バンドルが更新された後に 2 回目のコレクターデプロイメントを実行していました。今回の更新により、Operator は、コレクターデプロイメントを続行する前に、バンドルが読み込まれたかどうかを確認するために少し待機するようになります。(LOG-2789)

- この更新の前は、メトリクスを確認するときに Telemetry ログ情報が 2 回表示されていました。今回の更新により、Telemetry ログ情報が想定どおりに表示されます。(LOG-2315)

- この更新の前は、JSON 解析の追加を有効にした後、Fluentd Pod ログに警告メッセージが含まれていました。今回の更新でにより、その警告メッセージは表示されなくなりました。(LOG-1806)

- この更新の前は、キャッシュをビルドするために書き込み権限を持つフォルダーを **oc** が必要とするため、**must-gather** スクリプトは完了しませんでした。今回の更新により、**oc** はフォルダーへの書き込み権限を持ち、**must-gather** スクリプトが正常に完了するようになります。(LOG-3446)

- この更新の前は、ログコレクター SCC がクラスター上の他の SCC に取って代わられ、コレクターが使用できなくなる可能性がありました。今回の更新により、ログコレクター SCC の優先度が設定され、他の SCC よりも優先されるようになりました。(LOG-3235)

- この更新の前は、Vector に **sequence** フィールドはなく、ナノ秒単位の精度の欠落に対処する方法として fluentd に追加されていました。今回の更新により、**openshift.sequence** フィールドがイベントログに追加されました。(LOG-3106)

#### 3.1.10.5. CVE

- CVE-2020-36518

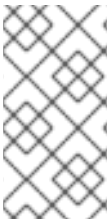
- CVE-2021-46848
- CVE-2022-2879
- CVE-2022-2880
- CVE-2022-27664
- CVE-2022-32190
- CVE-2022-35737
- CVE-2022-37601
- CVE-2022-41715
- CVE-2022-42003
- CVE-2022-42004
- CVE-2022-42010
- CVE-2022-42011
- CVE-2022-42012
- CVE-2022-42898
- CVE-2022-43680

## 3.2. LOGGING 5.6 を使い始める

ロギングデプロイメントプロセスの概要は、参照しやすいように提供されています。完全なドキュメントに代わるものではありません。新規インストールの場合は、**Vector** と **LokiStack** を推奨します。

**注記**

Logging バージョン 5.5 の時点で、**Fluentd** または **Vector** コレクター実装から選択するオプション、およびログストアとして **Elasticsearch** または **LokiStack** を選択するオプションがあります。ログのドキュメントは、これらの基本的なコンポーネントの変更を反映するために更新中です。



**注記**

Red Hat OpenShift のロギングサブシステムは、インストール可能なコンポーネントとして提供され、コアの OpenShift Container Platform とは異なるリリースサイクルを備えています。Red Hat OpenShift Container Platform ライフサイクルポリシー はリリースの互換性を概説しています。

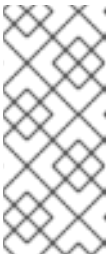
**前提条件**

- LogStore 設定: **Elasticsearch** または **LokiStack**
- コレクターの実装設定: **Fluentd** または **Vector**
- ログ転送出力の認証情報

**注記**

Logging バージョン 5.4.3 の時点で、Elasticsearch Operator は非推奨であり、今後のリリースで削除される予定です。Red Hat は、この機能に対して現在のリリースライフサイクル中にバグ修正とサポートを提供しますが、拡張機能の提供はなく、この機能は今後削除される予定です。Elasticsearch Operator を使用してデフォルトのログストレージを管理する代わりに、Loki Operator を使用できます。

1. 使用するログストアの Operator をインストールします。
   - **Elasticsearch** の場合は、**OpenShift Elasticsearch Operator** をインストールします。
   - **LokiStack** の場合は、**Loki Operator** をインストールします。
     - **LokiStack** カスタムリソース (CR) インスタンスを作成します。
2. **Red Hat OpenShift Logging Operator** をインストールします。
3. **ClusterLogging** カスタムリソース (CR) インスタンスを作成します。
   a. コレクターの実装を選択します。



**注記**

Logging バージョン 5.6 の時点で、Fluentd は非推奨であり、今後のリリースで削除される予定です。Red Hat は、この機能に対して現在のリリースライフサイクル中にバグ修正とサポートを提供しますが、拡張機能の提供はなく、この機能は今後削除される予定です。Fluentd の代わりに、Vector を使用できます。

4. **ClusterLogForwarder** カスタムリソース (CR) インスタンスを作成します。
5. 選択した出力パイプラインのシークレットを作成します。

## 3.3. ロギングについて理解する

ロギングサブシステムは、次の論理コンポーネントで設定されています。

- **Collector** - 各ノードからコンテナーログデータを読み取り、ログデータを設定済みの出力に転送します。
- **Store** - 分析用のログデータを保存します。フォワーダーのデフォルト出力。
- **Visualization** - 保存されたログを検索、クエリー、および表示するためのグラフィカルインターフェイス。

これらのコンポーネントは、Operator とカスタムリソース (CR) YAML ファイルによって管理されます。

Red Hat OpenShift のロギングサブシステムは、コンテナーログとノードログを収集します。これらは次のタイプに分類されます。

- **application** - 非インフラストラクチャーコンテナーによって生成されたコンテナーログ。
- **infrastructure** - namespace **kube-*** および **openshift-*** からのコンテナーログ、および **journald** からのノードログ。

- **audit** - 有効な場合は、**auditd**、**kube-apiserver**、**openshift-apiserver**、および **ovn** からのログ。

ロギングコレクターは、Pod を各 OpenShift Container Platform ノードにデプロイするデーモンセットです。システムおよびインフラストラクチャーのログは、オペレーティングシステム、コンテナーランタイム、および OpenShift Container Platform からの journald ログメッセージによって生成されます。

コンテナーログは、クラスターで実行されている Pod で実行されているコンテナーによって生成されます。各コンテナーは個別のログストリームを生成します。コレクターは、これらのソースからログを収集し、**ClusterLogForwarder** カスタムリソースで設定されているように、それらを内部または外部に転送します。

## 3.4. ロギングデプロイメントの管理

### 3.4.1. Web コンソールを使用した Red Hat OpenShift Logging Operator のデプロイ

OpenShift Container Platform Web コンソールを使用して、Red Hat OpenShift Logging Operator をデプロイできます。

**前提条件**

Red Hat OpenShift のロギングサブシステムは、インストール可能なコンポーネントとして提供され、コアの OpenShift Container Platform とは異なるリリースサイクルを備えています。Red Hat OpenShift Container Platform ライフサイクルポリシー はリリースの互換性を概説しています。

**手順**

OpenShift Container Platform Web コンソールを使用して、Red Hat OpenShift Logging Operator をデプロイするには、以下を実行します。

1. Red Hat OpenShift Logging Operator をインストールします。
   a. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
   b. **Filter by keyword** フィールドに **Logging** と入力します。
   c. 利用可能な Operator のリストから **Red Hat OpenShift Logging** を選択し、**Install** をクリックします。
   d. **Update Channel** として **stable** または **stable-5.y** を選択します。

**注記**

**stable** チャネルは、Logging の最新リリースを対象とする更新のみを提供します。以前のリリースの更新を引き続き受信するには、サブスクリプションチャネルを **stable-X** (**X** はインストールしたログのバージョン) に変更する必要があります。

   e. **Installation Mode** で **A specific namespace on the cluster**が選択されていることを確認します。

f. **Operator recommended namespace** が **Installed Namespace** で **openshift-logging** になっていることを確認します。

g. **Enable Operator recommended cluster monitoring on this Namespace**を選択します。

h. **Update approval**のオプションを選択します。

- **Automatic** オプションを使用すると、Operator Lifecycle Manager (OLM) は、新しいバージョンが利用可能になった際、Operator を自動的に更新できます。
- **Manual** オプションでは、適切な認証情報を持つユーザーが Operator の更新を承認する必要があります。

i. Console プラグインの **Enable** または **Disable** を選択します。

j. **Install** をクリックします。

2. **Operators → Installed Operators** ページに切り替えて、**Red Hat OpenShift Logging Operator** がインストールされていることを確認します。

a. **Red Hat OpenShift Logging** が **Status** が **Succeeded** の状態で **openshift-logging** プロジェクトにリスト表示されていることを確認します。

3. **ClusterLogging** インスタンスを作成します。

**注記**

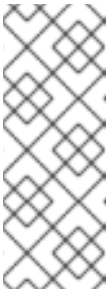
Web コンソールのフォームビューには、使用可能なすべてのオプションが含まれているわけではありません。セットアップを完了するには、**YAML ビュー** を推奨します。

a. **collection** セクションで、コレクターの実装を選択します。

**注記**

Logging バージョン 5.6 の時点で、Fluentd は非推奨であり、今後のリリースで削除される予定です。Red Hat は、この機能に対して現在のリリースライフサイクル中にバグ修正とサポートを提供しますが、拡張機能の提供はなく、この機能は今後削除される予定です。Fluentd の代わりに、Vector を使用できます。

b. **logStore** セクションで、タイプを選択します。



**注記**

Logging バージョン 5.4.3 の時点で、Elasticsearch Operator は非推奨であり、今後のリリースで削除される予定です。Red Hat は、この機能に対して現在のリリースライフサイクル中にバグ修正とサポートを提供しますが、拡張機能の提供はなく、この機能は今後削除される予定です。Elasticsearch Operator を使用してデフォルトのログストレージを管理する代わりに、Loki Operator を使用できます。

c. **Create** をクリックします。

### 3.4.2. Web コンソールを使用した Loki Operator のデプロイ

OpenShift Container Platform コンソールを使用して Loki Operator をインストールできます。

**前提条件**

- 対応ログストア (AWS S3、Google Cloud Storage、Azure、Swift、Minio、OpenShift Data Foundation)

**手順**

OpenShift Container PlatformWeb コンソールを使用して Loki Operator をインストールするには:

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** フィールドに **Loki** と入力します。
   a. 使用可能な Operator のリストから **Loki Operator** を選択し、**Install** をクリックします。
3. **Update Channel** として **stable** または **stable-5.y** を選択します。

**注記**

**stable** チャネルは、Logging の最新リリースを対象とする更新のみを提供します。以前のリリースの更新を引き続き受信するには、サブスクリプションチャネルを **stable-X** (**X** はインストールしたログのバージョン) に変更する必要があります。

4. **Installation Mode** で **All namespaces on the cluster**が選択されていることを確認します。
5. **openshift-operators-redhat** が **Installed Namespace** で選択されていることを確認します。
6. **Enable Operator recommended cluster monitoring on this Namespace**を選択します。このオプションは、namespace オブジェクトに **openshift.io/cluster-monitoring: "true"** ラベルを設定します。クラスターモニタリングが **openshift-operators-redhat** namespace を収集できるように、このオプションを選択する必要があります。
7. **Update approval** のオプションを選択します。
   - **Automatic** オプションを使用すると、Operator Lifecycle Manager (OLM) は、新しいバージョンが利用可能になった際、Operator を自動的に更新できます。
   - **Manual** オプションでは、適切な認証情報を持つユーザーが Operator の更新を承認する必要があります。
8. **Install** をクリックします。
9. **Operators → Installed Operators** ページに切り替えて、**LokiOperator** がインストールされていることを確認します。
   a. **LokiOperator** の全プロジェクトの **Status** が **Succeeded** として表示されているようにします。
10. **access_key_id** および **access_key_secret** フィールドを使用して、認証情報を指定し、**bucketnames**、**endpoint**、および **region** を指定して、オブジェクトストレージの場所を定義する **Secret** YAML ファイルを作成します。次の例では、AWS が使用されています。

```
apiVersion: v1
kind: Secret
metadata:
  name: logging-loki-s3
  namespace: openshift-logging
stringData:
  access_key_id: AKIAIOSFODNN7EXAMPLE
  access_key_secret: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
  bucketnames: s3-bucket-name
  endpoint: https://s3.eu-central-1.amazonaws.com
  region: eu-central-1
```

11. **Details** タブの LokiStack の下にある **Create instance** を選択します。次に、**YAML view** を選択します。次のテンプレートに貼り付け、必要に応じて値を置き換えます。

```
apiVersion: loki.grafana.com/v1
kind: LokiStack
metadata:
  name: logging-loki (1)
  namespace: openshift-logging
spec:
  size: 1x.small (2)
  storage:
    schemas:
    - version: v12
      effectiveDate: '2022-06-01'
    secret:
      name: logging-loki-s3 (3)
      type: s3 (4)
  storageClassName: <storage_class_name> (5)
  tenants:
    mode: openshift-logging
```

(1) 名前は、**logging-loki** にする必要があります。

(2) Loki のデプロイメントサイズを選択します。

(3) ログストレージに使用するシークレットを定義します。

(4) 対応するストレージタイプを定義します。

(5) 一時ストレージ用の既存のストレージクラスの名前を入力します。最適なパフォーマンスを得るには、ブロックストレージを割り当てるストレージクラスを指定します。クラスターで使用可能なストレージクラスは、**oc get storageclasses** で一覧表示できます。

    a. 設定を適用します。

```
oc apply -f logging-loki.yaml
```

12. **ClusterLogging** CR を作成または編集します。

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogging
```

```
  metadata:
    name: instance
    namespace: openshift-logging
  spec:
    managementState: Managed
    logStore:
      type: lokistack
      lokistack:
        name: logging-loki
      collection:
        type: vector
```

a. 設定を適用します。

```
oc apply -f cr-lokistack.yaml
```

### 3.4.3. CLI を使用した OperatorHub からのインストール

OpenShift Container Platform Web コンソールを使用する代わりに、CLI を使用して OperatorHub から Operator をインストールできます。**oc** コマンドを使用して、**Subscription** オブジェクトを作成または更新します。

**前提条件**

- **cluster-admin** パーミッションを持つアカウントを使用して OpenShift Container Platform クラスターにアクセスできる。
- **oc** コマンドをローカルシステムにインストールする。

**手順**

1. OperatorHub からクラスターで利用できる Operator のリストを表示します。

```
$ oc get packagemanifests -n openshift-marketplace
```

**出力例**

```
NAME                              CATALOG               AGE
3scale-operator                   Red Hat Operators     91m
advanced-cluster-management       Red Hat Operators     91m
amq7-cert-manager                 Red Hat Operators     91m
...
couchbase-enterprise-certified    Certified Operators   91m
crunchy-postgres-operator         Certified Operators   91m
mongodb-enterprise                Certified Operators   91m
...
etcd                              Community Operators   91m
jaeger                            Community Operators   91m
kubefed                           Community Operators   91m
...
```

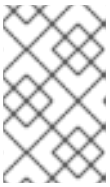
必要な Operator のカタログをメモします。

2. 必要な Operator を検査して、サポートされるインストールモードおよび利用可能なチャネルを確認します。

```
$ oc describe packagemanifests <operator_name> -n openshift-marketplace
```

3. **OperatorGroup** で定義される Operator グループは、Operator グループと同じ namespace 内のすべての Operator に必要な RBAC アクセスを生成するターゲット namespace を選択します。
Operator をサブスクライブする namespace には、Operator のインストールモードに一致する Operator グループが必要になります (**AllNamespaces** または **SingleNamespace** モードのいずれか)。インストールする Operator が **AllNamespaces** を使用する場合、 **openshift-operators** namespace には適切な Operator グループがすでに配置されます。

   ただし、Operator が **SingleNamespace** モードを使用し、適切な Operator グループがない場合、それらを作成する必要があります。



**注記**

この手順の Web コンソールバージョンでは、**SingleNamespace** モードを選択する際に、**OperatorGroup** および **Subscription** オブジェクトの作成を背後で自動的に処理します。

   a. **OperatorGroup** オブジェクト YAML ファイルを作成します (例: **operatorgroup.yaml**)。

**OperatorGroup オブジェクトのサンプル**

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: <operatorgroup_name>
  namespace: <namespace>
spec:
  targetNamespaces:
  - <namespace>
```

   b. **OperatorGroup** オブジェクトを作成します。

```
$ oc apply -f operatorgroup.yaml
```

4. **Subscription** オブジェクトの YAML ファイルを作成し、namespace を Operator にサブスクライブします (例: **sub.yaml**)。

**Subscription オブジェクトの例**

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: <subscription_name>
  namespace: openshift-operators 1
spec:
  channel: <channel_name> 2
  name: <operator_name> 3
  source: redhat-operators 4
```

```
sourceNamespace: openshift-marketplace 5
config:
 env: 6
 - name: ARGS
   value: "-v=10"
 envFrom: 7
 - secretRef:
     name: license-secret
 volumes: 8
 - name: <volume_name>
   configMap:
     name: <configmap_name>
 volumeMounts: 9
 - mountPath: <directory_name>
   name: <volume_name>
 tolerations: 10
 - operator: "Exists"
 resources: 11
   requests:
     memory: "64Mi"
     cpu: "250m"
   limits:
     memory: "128Mi"
     cpu: "500m"
 nodeSelector: 12
   foo: bar
```

1. **AllNamespaces** インストールモードの使用については、**openshift-operators** namespace を指定します。それ以外の場合は、**SingleNamespace** インストールモードの使用について関連する単一の namespace を指定します。
2. サブスクライブするチャネルの名前。
3. サブスクライブする Operator の名前。
4. Operator を提供するカタログソースの名前。
5. カタログソースの namespace。デフォルトの OperatorHub カタログソースには **openshift-marketplace** を使用します。
6. **env** パラメーターは、OLM によって作成される Pod のすべてのコンテナーに存在する必要がある環境変数の一覧を定義します。
7. **envFrom** パラメーターは、コンテナーの環境変数に反映するためのソースの一覧を定義します。
8. **volumes** パラメーターは、OLM によって作成される Pod に存在する必要があるボリュームの一覧を定義します。
9. **volumeMounts** パラメーターは、OLM によって作成される Pod のすべてのコンテナーに存在する必要があるボリュームマウントの一覧を定義します。**volumeMount** が存在しない **ボリューム** を参照する場合、OLM は Operator のデプロイに失敗します。
10. **tolerations** パラメーターは、OLM によって作成される Pod の容認の一覧を定義します。

11 **resources** パラメーターは、OLM によって作成される Pod のすべてのコンテナーのリソース制約を定義します。

12 **nodeSelector** パラメーターは、OLM によって作成される Pod の **ノードセレクター** を定義します。

5. **Subscription** オブジェクトを作成します。

```
$ oc apply -f sub.yaml
```

この時点で、OLM は選択した Operator を認識します。Operator のクラスターサービスバージョン (CSV) はターゲット namespace に表示され、Operator で指定される API は作成用に利用可能になります。

### 3.4.4. Web コンソールの使用によるクラスターからの Operator の削除

クラスター管理者は Web コンソールを使用して、選択した namespace からインストールされた Operator を削除できます。

**前提条件**

- **cluster-admin** パーミッションを持つアカウントを使用して OpenShift Container Platform クラスター Web コンソールにアクセスできること。

**手順**

1. **Operators → Installed Operators** ページに移動します。
2. スクロールするか、キーワードを **Filter by name** フィールドに入力して、削除する Operator を見つけます。次に、それをクリックします。
3. **Operator Details** ページの右側で、**Actions** 一覧から **Uninstall Operator** を選択します。**Uninstall Operator?** ダイアログボックスが表示されます。
4. **Uninstall** を選択し、Operator、Operator デプロイメント、および Pod を削除します。このアクションの後には、Operator は実行を停止し、更新を受信しなくなります。

**注記**

このアクションは、カスタムリソース定義 (CRD) およびカスタムリソース (CR) など、Operator が管理するリソースは削除されません。Web コンソールおよび継続して実行されるクラスター外のリソースによって有効にされるダッシュボードおよびナビゲーションアイテムには、手動でのクリーンアップが必要になる場合があります。Operator のアンインストール後にこれらを削除するには、Operator CRD を手動で削除する必要があります。

### 3.4.5. CLI の使用によるクラスターからの Operator の削除

クラスター管理者は CLI を使用して、選択した namespace からインストールされた Operator を削除できます。

**前提条件**

- **cluster-admin** パーミッションを持つアカウントを使用して OpenShift Container Platform クラスターにアクセスできる。
- **oc** コマンドがワークステーションにインストールされていること。

**手順**

1. サブスクライブされた Operator (例: **jaeger**) の現行バージョンを **currentCSV** フィールドで確認します。

```
$ oc get subscription jaeger -n openshift-operators -o yaml | grep currentCSV
```

**出力例**

```
  currentCSV: jaeger-operator.v1.8.2
```

2. サブスクリプション (例: **jaeger**) を削除します。

```
$ oc delete subscription jaeger -n openshift-operators
```

**出力例**

```
subscription.operators.coreos.com "jaeger" deleted
```

3. 直前の手順で **currentCSV** 値を使用し、ターゲット namespace の Operator の CSV を削除します。

```
$ oc delete clusterserviceversion jaeger-operator.v1.8.2 -n openshift-operators
```

**出力例**

```
clusterserviceversion.operators.coreos.com "jaeger-operator.v1.8.2" deleted
```

## 3.5. ロギング参照

### 3.5.1. コレクター機能

| 出力 | プロトコル | テストで使用 | Fluentd | Vector |
|---|---|---|---|---|
| Cloudwatch | REST over HTTP(S) | | ✓ | ✓ |
| Elasticsearch v6 | | v6.8.1 | ✓ | ✓ |
| Elasticsearch v7 | | v7.12.2、7.17.7 | ✓ | ✓ |
| Elasticsearch v8 | | v8.4.3 | | ✓ |

| 出力 | プロトコル | テストで使用 | Fluentd | Vector |
|---|---|---|---|---|
| Fluent Forward | Fluentd forward v1 | Fluentd 1.14.6、Logstash 7.10.1 | ✓ | |
| Google Cloud Logging | | | | ✓ |
| HTTP | HTTP 1.1 | Fluentd 1.14.6、Vector 0.21 | | |
| Kafka | Kafka 0.11 | Kafka 2.4.1、2.7.0、3.3.1 | ✓ | ✓ |
| Loki | REST over HTTP(S) | Loki 2.3.0、2.7 | ✓ | ✓ |
| Splunk | HEC | v8.2.9、9.0.0 | | ✓ |
| Syslog | RFC3164、RFC5424 | Rsyslog 8.37.0-9.el7 | ✓ | |

**表3.1 ログソース**

| 機能 | Fluentd | Vector |
|---|---|---|
| アプリコンテナーのログ | ✓ | ✓ |
| アプリ固有のルーティング | ✓ | ✓ |
| namespace 別のアプリ固有のルーティング | ✓ | ✓ |
| インフラコンテナーログ | ✓ | ✓ |
| インフラジャーナルログ | ✓ | ✓ |
| Kube API 監査ログ | ✓ | ✓ |
| OpenShift API 監査ログ | ✓ | ✓ |
| Open Virtual Network (OVN) 監査ログ | ✓ | ✓ |

**表3.2 認証および認可**

| 機能 | Fluentd | Vector |
| --- | --- | --- |
| Elasticsearch 証明書 | ✓ | ✓ |
| Elasticsearch ユーザー名/パスワード | ✓ | ✓ |
| Cloudwatch キー | ✓ | ✓ |
| クラウドウォッチ STS | ✓ | ✓ |
| Kafka 証明書 | ✓ | ✓ |
| Kafka のユーザー名/パスワード | ✓ | ✓ |
| Kafka SASL | ✓ | ✓ |
| Loki ベアラートークン | ✓ | ✓ |

**表3.3 正規化と変換**

| 機能 | Fluentd | Vector |
| --- | --- | --- |
| Viaq データモデル - アプリ | ✓ | ✓ |
| Viaq データモデル - インフラ | ✓ | ✓ |
| Viaq データモデル - インフラ (ジャーナル) | ✓ | ✓ |
| Viaq データモデル - Linux 監査 | ✓ | ✓ |
| Viaq データモデル - kube-apiserver 監査 | ✓ | ✓ |
| Viaq データモデル - OpenShift API 監査 | ✓ | ✓ |
| Viaq データモデル - OVN | ✓ | ✓ |
| ログレベルの正規化 | ✓ | ✓ |
| JSON 解析 | ✓ | ✓ |
| 構造化インデックス | ✓ | ✓ |
| 複数行エラー検出 | ✓ | |

| 機能 | Fluentd | Vector |
| --- | --- | --- |
| マルチコンテナー/分割インデックス | ✓ | ✓ |
| ラベルのフラット化 | ✓ | ✓ |
| CLF 静的ラベル | ✓ | ✓ |

**表3.4 チューニング**

| 機能 | Fluentd | Vector |
| --- | --- | --- |
| Fluentd readlinelimit | ✓ | |
| Fluentd バッファー | ✓ | |
| -chunklimitsize | ✓ | |
| - totallimitsize | ✓ | |
| - overflowaction | ✓ | |
| -flushThreadCount | ✓ | |
| - flushmode | ✓ | |
| - flushinterval | ✓ | |
| - retrywait | ✓ | |
| - retrytype | ✓ | |
| - retrymaxinterval | ✓ | |
| - retrytimeout | ✓ | |

**表3.5 制約**

| 機能 | Fluentd | Vector |
| --- | --- | --- |
| メトリクス | ✓ | ✓ |
| ダッシュボード | ✓ | ✓ |
| アラート | ✓ | |

表3.6 その他

| 機能 | Fluentd | Vector |
|---|---|---|
| グローバルプロキシーサポート | ✓ | ✓ |
| x86 サポート | ✓ | ✓ |
| ARM サポート | ✓ | ✓ |
| IBM Power のサポート | ✓ | ✓ |
| IBM Z サポート | ✓ | ✓ |
| IPv6 サポート | ✓ | ✓ |
| ログイベントのバッファリング | ✓ | |
| 非接続クラスター | ✓ | ✓ |

**関連情報**

- Vector ドキュメント

## 3.5.2. Logging 5.6 API リファレンス

### 3.5.2.1. ClusterLogForwarder

ClusterLogForwarder は、転送ログを設定するための API です。

名前付き入力のセットから名前付き出力のセットに転送する **pipelines** のリストを指定して、転送を設定します。

一般的なログカテゴリーには組み込みの入力名があり、カスタム入力を定義して、追加のフィルタリングを行うことができます。

デフォルトの OpenShift ログストアには組み込みの出力名がありますが、URL やその他の接続情報を使用して、独自の出力を定義し、クラスターの内部または外部の他のストアまたはプロセッサーにログを転送できます。

詳細については、API フィールドに関するドキュメントを参照してください。

| プロパティー | 型 | 説明 |
|---|---|---|
| spec | object | ClusterLogForwarder の期待される動作の仕様 |
| status | object | ClusterLogForwarder のステータス |

#### 3.5.2.1.1. .spec

##### 3.5.2.1.1.1. 説明

ClusterLogForwarderSpec は、ログをリモートターゲットに転送する方法を定義します。

##### 3.5.2.1.1.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| inputs | array | **(オプション)** 入力は、転送されるログメッセージの名前付きフィルターです。 |
| outputDefaults | object | **(オプション)** DEPRECATED OutputDefaults は、デフォルトストアのフォワーダー設定を明示的に指定します。 |
| outputs | array | **(オプション)** 出力は、ログメッセージの名前付きの宛先です。 |
| pipelines | array | pipelines は、一連の入力によって選択されたメッセージを一連の出力に転送します。 |

#### 3.5.2.1.2. .spec.inputs[]

##### 3.5.2.1.2.1. 説明

InputSpec は、ログメッセージのセレクターを定義します。

##### 3.5.2.1.2.1.1. 型

- array

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| application | object | **(オプション)** アプリケーション (存在する場合) は、**application** ログの名前付きセットを有効にします。 |
| name | string | **pipeline** の入力を参照するために使用される名前。 |

#### 3.5.2.1.3. .spec.inputs[].application

#### 3.5.2.1.3.1. 説明

アプリケーションログセレクター。ログを選択するには、セレクターのすべての条件が満たされる (論理 AND) 必要があります。

#### 3.5.2.1.3.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| namespace | array | **(オプション)** アプリケーションログを収集する namespace。 |
| selector | object | **(オプション)** ラベルが一致する Pod からのログのセレクター。 |

### 3.5.2.1.4. .spec.inputs[].application.namespaces[]

#### 3.5.2.1.4.1. 説明

#### 3.5.2.1.4.1.1. 型

- array

### 3.5.2.1.5. .spec.inputs[].application.selector

#### 3.5.2.1.5.1. 説明

ラベルセレクターとは、一連のリソースに対するラベルクエリー機能です。

#### 3.5.2.1.5.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| matchLabels | object | **(オプション)** matchLabels は {key,value} ペアのマップです。matchLabels の単一の {key,value} |

### 3.5.2.1.6. .spec.inputs[].application.selector.matchLabels

#### 3.5.2.1.6.1. 説明

#### 3.5.2.1.6.1.1. 型

- object

#### 3.5.2.1.7. .spec.outputDefaults

##### 3.5.2.1.7.1. 説明

##### 3.5.2.1.7.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| elasticsearch | object | **(オプション)** Elasticsearch OutputSpec のデフォルト値 |

#### 3.5.2.1.8. .spec.outputDefaults.elasticsearch

##### 3.5.2.1.8.1. 説明

ElasticsearchStructuredSpec は、elasticsearch インデックスを決定するための構造化ログの変更に関連する仕様です。

##### 3.5.2.1.8.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| enableStructuredContainerLogs | bool | **(オプション)** EnableStructuredContainerLogs は、複数コンテナーの構造化ログを許可します。 |
| structuredTypeKey | string | **(オプション)** StructuredTypeKey は、elasticsearch インデックスの名前として使用されるメタデータキーを指定します。 |
| structuredTypeName | string | **(オプション)** StructuredTypeName は、elasticsearch スキーマの名前を指定します。 |

#### 3.5.2.1.9. .spec.outputs[]

##### 3.5.2.1.9.1. 説明

出力は、ログメッセージの宛先を定義します。

##### 3.5.2.1.9.1.1. 型

- array

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| syslog | object | **(オプション)** |
| fluentdForward | object | **(オプション)** |
| elasticsearch | object | **(オプション)** |
| kafka | object | **(オプション)** |
| cloudwatch | object | **(オプション)** |
| loki | object | **(オプション)** |
| googleCloudLogging | object | **(オプション)** |
| splunk | object | **(オプション)** |
| name | string | **pipeline** からの出力を参照するために使用される名前。 |
| secret | object | **(オプション)** 認証のシークレット。 |
| tls | object | TLS には、TLS クライアント接続のオプションを制御するための設定が含まれています。 |
| type | string | 出力プラグインのタイプ。 |
| url | string | **(オプション)** ログレコードの送信先 URL。 |

#### 3.5.2.1.10. .spec.outputs[].secret

##### 3.5.2.1.10.1. 説明

OutputSecretSpec は、名前のみを含み、namespace を含まないシークレット参照です。

##### 3.5.2.1.10.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| name | string | ログフォワーダーシークレット用に設定された namespace 内のシークレットの名前。 |

#### 3.5.2.1.11. .spec.outputs[].tls

##### 3.5.2.1.11.1. 説明

OutputTLSSpec には、出力タイプに依存しない TLS 接続のオプションが含まれています。

###### 3.5.2.1.11.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| insecureSkipVerify | bool | InsecureSkipVerify が true の場合、TLS クライアントは証明書のエラーを無視するように設定されます。 |

#### 3.5.2.1.12. .spec.pipelines[]

##### 3.5.2.1.12.1. 説明

PipelinesSpec は、一連の入力を一連の出力にリンクします。

###### 3.5.2.1.12.1.1. 型

- array

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| detectMultilineErrors | bool | **(オプション)** DetectMultilineErrors は、コンテナーログの複数行エラー検出を有効にします。 |
| inputRefs | array | InputRefs は、このパイプラインへの入力の名前 (**input.name**) をリストします。 |
| labels | object | **(オプション)** このパイプラインを通過するログレコードに適用されるラベル。 |
| name | string | **(オプション)** 名前は省略可能ですが、指定する場合は、**pipelines** リスト内で一意である必要があります。 |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| outputRefs | array | OutputRefs は、このパイプラインからの出力の名前 (**output.name**) を一覧表示します。 |
| parse | string | **(オプション)** 解析により、ログエントリーを構造化ログに解析できます。 |

#### 3.5.2.1.13. .spec.pipelines[].inputRefs[]

##### 3.5.2.1.13.1. 説明

##### 3.5.2.1.13.1.1. 型

- array

#### 3.5.2.1.14. .spec.pipelines[].labels

##### 3.5.2.1.14.1. 説明

##### 3.5.2.1.14.1.1. 型

- object

#### 3.5.2.1.15. .spec.pipelines[].outputRefs[]

##### 3.5.2.1.15.1. 説明

##### 3.5.2.1.15.1.1. 型

- array

#### 3.5.2.1.16. .status

##### 3.5.2.1.16.1. 説明

ClusterLogForwarderStatus は、ClusterLogForwarder の監視状態を定義します。

##### 3.5.2.1.16.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| conditions | object | ログフォワーダーの条件。 |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| inputs | Conditions | 入力は、入力名を入力の条件にマッピングします。 |
| outputs | Conditions | 出力は、出力名を出力の条件にマッピングします。 |
| pipelines | Conditions | pipelines は、パイプライン名をパイプラインの条件にマッピングします。 |

#### 3.5.2.1.17. .status.conditions

#### 3.5.2.1.17.1. 説明

#### 3.5.2.1.17.1.1. 型

- object

#### 3.5.2.1.18. .status.inputs

#### 3.5.2.1.18.1. 説明

#### 3.5.2.1.18.1.1. 型

- Conditions

#### 3.5.2.1.19. .status.outputs

#### 3.5.2.1.19.1. 説明

#### 3.5.2.1.19.1.1. 型

- Conditions

#### 3.5.2.1.20. .status.pipelines

#### 3.5.2.1.20.1. 説明

#### 3.5.2.1.20.1.1. 型

- Conditions== ClusterLogging A Red Hat OpenShift Logging インスタンス。ClusterLogging は、clusterloggings API のスキーマです。

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| spec | object | ClusterLogging の期待される動作の仕様 |
| status | object | Status は、ClusterLogging の監視状態を定義します。 |

#### 3.5.2.1.21. .spec

##### 3.5.2.1.21.1. 説明

ClusterLoggingSpec は ClusterLogging の期待される状態を定義します。

##### 3.5.2.1.21.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| コレクション | object | クラスターの Collection コンポーネントの仕様 |
| キュレーション | object | **(非推奨) (オプション)** 非推奨。クラスターの Curation コンポーネントの仕様 |
| フォワーダー | object | **(非推奨) (オプション)** 非推奨。クラスターの Forwarder コンポーネントの仕様 |
| logStore | object | **(オプション)** クラスターの Log Storage コンポーネントの仕様 |
| managementState | string | **(オプション)** リソースが Operator によって管理されているか管理されていないかを示す指標 |
| 可視化 | object | **(オプション)** クラスターの Visualization コンポーネントの仕様 |

#### 3.5.2.1.22. .spec.collection

##### 3.5.2.1.22.1. 説明

これは、ログおよびイベントコレクションに関連する情報を含む構造体です。

#### 3.5.2.1.22.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| resources | object | **(オプション)** コレクターのリソース要件 |
| nodeSelector | object | **(オプション)** Pod がスケジュールされるノードを定義します。 |
| Toleration | array | **(オプション)** Pod が受け入れる Toleration を定義します。 |
| fluentd | object | **(オプション)** Fluentd は、fluentd タイプのフォワーダーの設定を表します。 |
| logs | object | **(非推奨) (オプション)** 非推奨。クラスターのログ収集の仕様 |
| type | string | **(オプション)** 設定するログ収集のタイプ |

### 3.5.2.1.23. .spec.collection.fluentd

#### 3.5.2.1.23.1. 説明

FluentdForwarderSpec は、fluentd タイプのフォワーダーの設定を表します。

#### 3.5.2.1.23.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| buffer | object | |
| inFile | object | |

### 3.5.2.1.24. .spec.collection.fluentd.buffer

#### 3.5.2.1.24.1. 説明

FluentdBufferSpec は、すべての fluentd 出力のバッファー設定をチューニングするための fluentd バッファーパラメーターのサブセットを表します。パラメーターのサブセットをサポートして、バッファーとキューのサイズ設定、フラッシュ操作、フラッシュの再試行を設定します。

一般的なパラメーターについては、https://docs.fluentd.org/configuration/buffer-section#buffering-parameters を参照してください。

フラッシュパラメーターについては、https://docs.fluentd.org/configuration/buffer-section#flushing-parameters を参照してください。

再試行パラメーターについては、https://docs.fluentd.org/configuration/buffer-section#retries-parameters を参照してください。

#### 3.5.2.1.24.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| chunkLimitSize | string | **(オプション)** ChunkLimitSize は、各チャンクの最大サイズを表します。イベントは以下のようになります。 |
| flushInterval | string | **(オプション)** FlushInterval は、2 つの連続するフラッシュの間の待機時間を表します。 |
| flushMode | string | **(オプション)** FlushMode は、チャンクを書き込むフラッシュスレッドのモードを表します。モード |
| flushThreadCount | int | **(オプション)** FlushThreadCount は、fluentd バッファーによって使用されるスレッドの数を表します。 |
| overflowAction | string | **(オプション)** OverflowAction は、fluentd バッファープラグインが実行するアクションを表します。 |
| retryMaxInterval | string | **(オプション)** RetryMaxInterval は、指数バックオフの最大時間間隔を表します。 |
| retryTimeout | string | **(オプション)** RetryTimeout は、あきらめる前に再試行を試みる最大時間間隔を表します。 |
| retryType | string | **(オプション)** RetryType は、再試行するフラッシュ操作のタイプを表します。フラッシュ操作は以下を実行できます。 |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| retryWait | string | **(オプション)** RetryWait は、2 回連続して再試行してフラッシュするまでの時間を表します。 |
| totalLimitSize | string | **(オプション)** TotalLimitSize は、fluentd ごとに許可されるノード領域のしきい値を表します。 |

#### 3.5.2.1.25. .spec.collection.fluentd.inFile

#### 3.5.2.1.25.1. 説明

FluentdInFileSpec は、すべての fluentd in-tail 入力の設定をチューニングするための fluentd in-tail プラグインパラメーターのサブセットを表します。

一般的なパラメーターについては、https://docs.fluentd.org/input/tail#parameters を参照してください。

#### 3.5.2.1.25.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| readLinesLimit | int | **(オプション)** ReadLinesLimit は、各 I/O 操作で読み取る行数を表します。 |

#### 3.5.2.1.26. .spec.collection.logs

#### 3.5.2.1.26.1. 説明

#### 3.5.2.1.26.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| fluentd | object | Fluentd Log Collection コンポーネントの仕様 |
| type | string | 設定するログ収集のタイプ |

#### 3.5.2.1.27. .spec.collection.logs.fluentd

#### 3.5.2.1.27.1. 説明

CollectorSpec は、コレクターのスケジュールとリソースを定義するための仕様です。

#### 3.5.2.1.27.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| nodeSelector | object | **(オプション)** Pod がスケジュールされるノードを定義します。 |
| resources | object | **(オプション)** コレクターのリソース要件 |
| Toleration | array | **(オプション)** Pod が受け入れる Toleration を定義します。 |

### 3.5.2.1.28. .spec.collection.logs.fluentd.nodeSelector

#### 3.5.2.1.28.1. 説明

#### 3.5.2.1.28.1.1. 型

- object

### 3.5.2.1.29. .spec.collection.logs.fluentd.resources

#### 3.5.2.1.29.1. 説明

#### 3.5.2.1.29.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| limits | object | **(オプション)** Limits は、許可されるコンピューティングリソースの最大量を示します。 |
| requests | object | **(オプション)** Requests は、必要なコンピューティングリソースの最小量を示します。 |

### 3.5.2.1.30. .spec.collection.logs.fluentd.resources.limits

#### 3.5.2.1.30.1. 説明

#### 3.5.2.1.30.1.1. 型

- object

#### 3.5.2.1.31. .spec.collection.logs.fluentd.resources.requests

##### 3.5.2.1.31.1. 説明

##### 3.5.2.1.31.1.1. 型

- object

#### 3.5.2.1.32. .spec.collection.logs.fluentd.tolerations[]

##### 3.5.2.1.32.1. 説明

##### 3.5.2.1.32.1.1. 型

- array

| プロパティー | 型 | 説明 |
|---|---|---|
| effect | string | **(オプション)** Effect は、一致する Taint 効果を示します。空の場合は、すべてのテイント効果に一致します。 |
| 鍵 (key) | string | **(オプション)** Key は、Toleration が適用される Taint キーです。空の場合は、すべてのテイントキーに一致します。 |
| operator | string | **(オプション)** Operator は、キーと値の関係を表します。 |
| tolerationSeconds | int | **(オプション)** TolerationSeconds は、Toleration の期間を表します。 |
| value | string | **(オプション)** Value は、Toleration が一致する Taint 値です。 |

#### 3.5.2.1.33. .spec.collection.logs.fluentd.tolerations[].tolerationSeconds

##### 3.5.2.1.33.1. 説明

##### 3.5.2.1.33.1.1. 型

- int

#### 3.5.2.1.34. .spec.curation

##### 3.5.2.1.34.1. 説明

これは、ログのキュレーション (Curator) に関連する情報を含む構造体です。

##### 3.5.2.1.34.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| curator | object | 設定するキュレーションの仕様 |
| type | string | 設定するキュレーションの種類 |

#### 3.5.2.1.35. .spec.curation.curator

##### 3.5.2.1.35.1. 説明

##### 3.5.2.1.35.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| nodeSelector | object | Pod がスケジュールされているノードを定義します。 |
| resources | object | **(オプション)** Curator のリソース要件 |
| schedule | string | Curator ジョブが実行される cron スケジュール。デフォルトは「30 3 * * *」です。 |
| Toleration | array | |

#### 3.5.2.1.36. .spec.curation.curator.nodeSelector

##### 3.5.2.1.36.1. 説明

##### 3.5.2.1.36.1.1. 型

- object

#### 3.5.2.1.37. .spec.curation.curator.resources

#### 3.5.2.1.37.1. 説明

#### 3.5.2.1.37.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| limits | object | **(オプション)** Limits は、許可されるコンピューティングリソースの最大量を示します。 |
| requests | object | **(オプション)** Requests は、必要なコンピューティングリソースの最小量を示します。 |

### 3.5.2.1.38. .spec.curation.curator.resources.limits

#### 3.5.2.1.38.1. 説明

#### 3.5.2.1.38.1.1. 型

- object

### 3.5.2.1.39. .spec.curation.curator.resources.requests

#### 3.5.2.1.39.1. 説明

#### 3.5.2.1.39.1.1. 型

- object

### 3.5.2.1.40. .spec.curation.curator.tolerations[]

#### 3.5.2.1.40.1. 説明

#### 3.5.2.1.40.1.1. 型

- array

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| effect | string | **(オプション)** Effect は、一致する Taint 効果を示します。空の場合は、すべてのテイント効果に一致します。 |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| 鍵 (key) | string | **(オプション)** Key は、Toleration が適用される Taint キーです。空の場合は、すべてのテイントキーに一致します。 |
| operator | string | **(オプション)** Operator は、キーと値の関係を表します。 |
| tolerationSeconds | int | **(オプション)** TolerationSeconds は、Toleration の期間を表します。 |
| value | string | **(オプション)** Value は、Toleration が一致する Taint 値です。 |

#### 3.5.2.1.41. .spec.curation.curator.tolerations[].tolerationSeconds

##### 3.5.2.1.41.1. 説明

##### 3.5.2.1.41.1.1. 型

- int

#### 3.5.2.1.42. .spec.forwarder

##### 3.5.2.1.42.1. 説明

ForwarderSpec には、特定のフォワーダー実装のグローバルチューニングパラメーターが含まれています。このフィールドは、一般的な使用には必要ありません。基礎となるフォワーダーテクノロジーに精通しているユーザーがパフォーマンスをチューニングできるようにします。現在サポートされているもの: **fluentd**。

##### 3.5.2.1.42.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| fluentd | object | |

#### 3.5.2.1.43. .spec.forwarder.fluentd

##### 3.5.2.1.43.1. 説明

FluentdForwarderSpec は、fluentd タイプのフォワーダーの設定を表します。

#### 3.5.2.1.43.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| buffer | object | |
| inFile | object | |

### 3.5.2.1.44. .spec.forwarder.fluentd.buffer

#### 3.5.2.1.44.1. 説明

FluentdBufferSpec は、すべての fluentd 出力のバッファー設定をチューニングするための fluentd バッファーパラメーターのサブセットを表します。パラメーターのサブセットをサポートして、バッファーとキューのサイズ設定、フラッシュ操作、フラッシュの再試行を設定します。

一般的なパラメーターについては、https://docs.fluentd.org/configuration/buffer-section#buffering-parameters を参照してください。

フラッシュパラメーターについては、https://docs.fluentd.org/configuration/buffer-section#flushing-parameters を参照してください。

再試行パラメーターについては、https://docs.fluentd.org/configuration/buffer-section#retries-parameters を参照してください。

#### 3.5.2.1.44.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| chunkLimitSize | string | **(オプション)** ChunkLimitSize は、各チャンクの最大サイズを表します。イベントは以下のようになります。 |
| flushInterval | string | **(オプション)** FlushInterval は、2 つの連続するフラッシュの間の待機時間を表します。 |
| flushMode | string | **(オプション)** FlushMode は、チャンクを書き込むフラッシュスレッドのモードを表します。モード |
| flushThreadCount | int | **(オプション)** FlushThreadCount は、fluentd バッファーによって使用されるスレッドの数を表します。 |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| overflowAction | string | **(オプション)** OverflowAction は、fluentd バッファープラグインが実行するアクションを表します。 |
| retryMaxInterval | string | **(オプション)** RetryMaxInterval は、指数バックオフの最大時間間隔を表します。 |
| retryTimeout | string | **(オプション)** RetryTimeout は、あきらめる前に再試行を試みる最大時間間隔を表します。 |
| retryType | string | **(オプション)** RetryType は、再試行するフラッシュ操作のタイプを表します。フラッシュ操作は以下を実行できます。 |
| retryWait | string | **(オプション)** RetryWait は、2 回連続して再試行してフラッシュするまでの時間を表します。 |
| totalLimitSize | string | **(オプション)** TotalLimitSize は、fluentd ごとに許可されるノード領域のしきい値を表します。 |

#### 3.5.2.1.45. .spec.forwarder.fluentd.inFile

##### 3.5.2.1.45.1. 説明

FluentdInFileSpec は、すべての fluentd in-tail 入力の設定をチューニングするための fluentd in-tail プラグインパラメーターのサブセットを表します。

一般的なパラメーターについては、https://docs.fluentd.org/input/tail#parameters を参照してください。

###### 3.5.2.1.45.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| readLinesLimit | int | **(オプション)** ReadLinesLimit は、各 I/O 操作で読み取る行数を表します。 |

#### 3.5.2.1.46. .spec.logStore

#### 3.5.2.1.46.1. 説明

LogStoreSpec には、ログの保存方法に関する情報が含まれています。

#### 3.5.2.1.46.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| elasticsearch | object | Elasticsearch Log Store コンポーネントの仕様 |
| lokistack | object | LokiStack には、Type が LogStoreTypeLokiStack に設定されている場合、ログストレージに使用する LokiStack に関する情報が含まれています。 |
| retentionPolicy | object | **(オプション)** 保持ポリシーは、インデックスが削除されるまでの最大期間を定義します。 |
| type | string | 設定するログストレージのタイプ。現在、Operator は、ElasticSearch を使用して、いずれかをサポートしています。 |

### 3.5.2.1.47. .spec.logStore.elasticsearch

#### 3.5.2.1.47.1. 説明

#### 3.5.2.1.47.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| nodeCount | int | Elasticsearch 用にデプロイするノードの数 |
| nodeSelector | object | Pod がスケジュールされているノードを定義します。 |
| proxy | object | Elasticsearch Proxy コンポーネントの仕様 |
| redundancyPolicy | string | **(オプション)** |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| resources | object | **(オプション)** Elasticsearch のリソース要件 |
| storage | object | **(オプション)** Elasticsearch データノードのストレージ仕様 |
| Toleration | array | |

#### 3.5.2.1.48. .spec.logStore.elasticsearch.nodeSelector

##### 3.5.2.1.48.1. 説明

##### 3.5.2.1.48.1.1. 型

- object

#### 3.5.2.1.49. .spec.logStore.elasticsearch.proxy

##### 3.5.2.1.49.1. 説明

##### 3.5.2.1.49.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| resources | object | |

#### 3.5.2.1.50. .spec.logStore.elasticsearch.proxy.resources

##### 3.5.2.1.50.1. 説明

##### 3.5.2.1.50.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| limits | object | **(オプション)** Limits は、許可されるコンピューティングリソースの最大量を示します。 |
| requests | object | **(オプション)** Requests は、必要なコンピューティングリソースの最小量を示します。 |

#### 3.5.2.1.51. .spec.logStore.elasticsearch.proxy.resources.limits

##### 3.5.2.1.51.1. 説明

##### 3.5.2.1.51.1.1. 型

- object

#### 3.5.2.1.52. .spec.logStore.elasticsearch.proxy.resources.requests

##### 3.5.2.1.52.1. 説明

##### 3.5.2.1.52.1.1. 型

- object

#### 3.5.2.1.53. .spec.logStore.elasticsearch.resources

##### 3.5.2.1.53.1. 説明

##### 3.5.2.1.53.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| limits | object | **(オプション)** Limits は、許可されるコンピューティングリソースの最大量を示します。 |
| requests | object | **(オプション)** Requests は、必要なコンピューティングリソースの最小量を示します。 |

#### 3.5.2.1.54. .spec.logStore.elasticsearch.resources.limits

##### 3.5.2.1.54.1. 説明

##### 3.5.2.1.54.1.1. 型

- object

#### 3.5.2.1.55. .spec.logStore.elasticsearch.resources.requests

##### 3.5.2.1.55.1. 説明

##### 3.5.2.1.55.1.1. 型

- object

#### 3.5.2.1.56. .spec.logStore.elasticsearch.storage

##### 3.5.2.1.56.1. 説明

##### 3.5.2.1.56.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| size | object | ノードがプロビジョニングする最大ストレージ容量。 |
| storageClassName | string | **(オプション)** ノードの PVC の作成に使用するストレージクラスの名前。 |

#### 3.5.2.1.57. .spec.logStore.elasticsearch.storage.size

##### 3.5.2.1.57.1. 説明

##### 3.5.2.1.57.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| フォーマット | string | 形式を自由に変更します。Canonicalize のコメントを参照してください。 |
| d | object | d.Dec != nil の場合、d は inf.Dec 形式の数量です。 |
| i | int | d.Dec == nil の場合、i は int64 でスケーリングされた形式の数量です。 |
| s | string | s は、再計算を避けるために生成されたこの量の値です。 |

#### 3.5.2.1.58. .spec.logStore.elasticsearch.storage.size.d

##### 3.5.2.1.58.1. 説明

##### 3.5.2.1.58.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| Dec | object | |

#### 3.5.2.1.59. .spec.logStore.elasticsearch.storage.size.d.Dec

#### 3.5.2.1.59.1. 説明

#### 3.5.2.1.59.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| scale | int | |
| unscaled | object | |

#### 3.5.2.1.60. .spec.logStore.elasticsearch.storage.size.d.Dec.unscaled

#### 3.5.2.1.60.1. 説明

#### 3.5.2.1.60.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| abs | Word | sign |
| neg | bool | |

#### 3.5.2.1.61. .spec.logStore.elasticsearch.storage.size.d.Dec.unscaled.abs

#### 3.5.2.1.61.1. 説明

#### 3.5.2.1.61.1.1. 型

- Word

#### 3.5.2.1.62. .spec.logStore.elasticsearch.storage.size.i

#### 3.5.2.1.62.1. 説明

#### 3.5.2.1.62.1.1. 型

- int

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| scale | int | |
| value | int | |

#### 3.5.2.1.63. .spec.logStore.elasticsearch.tolerations[]

##### 3.5.2.1.63.1. 説明

###### 3.5.2.1.63.1.1. 型

- array

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| effect | string | **(オプション)** Effect は、一致する Taint 効果を示します。空の場合は、すべてのテイント効果に一致します。 |
| 鍵 (key) | string | **(オプション)** Key は、Toleration が適用される Taint キーです。空の場合は、すべてのテイントキーに一致します。 |
| operator | string | **(オプション)** Operator は、キーと値の関係を表します。 |
| tolerationSeconds | int | **(オプション)** TolerationSeconds は、Toleration の期間を表します。 |
| value | string | **(オプション)** Value は、Toleration が一致する Taint 値です。 |

#### 3.5.2.1.64. .spec.logStore.elasticsearch.tolerations[].tolerationSeconds

##### 3.5.2.1.64.1. 説明

###### 3.5.2.1.64.1.1. 型

- int

#### 3.5.2.1.65. .spec.logStore.lokistack

#### 3.5.2.1.65.1. 説明

LokiStackStoreSpec は、LokiStack をログストレージとして使用するように、cluster-logging を設定するために使用されます。これは、同じ namespace 内の既存の LokiStack を指しています。

#### 3.5.2.1.65.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| name | string | LokiStack リソースの名前。 |

### 3.5.2.1.66. .spec.logStore.retentionPolicy

#### 3.5.2.1.66.1. 説明

#### 3.5.2.1.66.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| application | object | |
| audit | object | |
| infra | object | |

### 3.5.2.1.67. .spec.logStore.retentionPolicy.application

#### 3.5.2.1.67.1. 説明

#### 3.5.2.1.67.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| diskThresholdPercent | int | **(オプション)** ES ディスク使用率のしきい値に達した場合、古いインデックスを削除する必要があります (例: 75)。 |
| maxAge | string | **(オプション)** |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| namespaceSpec | array | **(オプション)** 指定された最小期間よりも古いドキュメントを削除する namespace ごとの仕様 |
| pruneNamespacesInterval | string | **(オプション)** 新しい prune-namespaces ジョブを実行する頻度 |

#### 3.5.2.1.68. .spec.logStore.retentionPolicy.application.namespaceSpec[]

##### 3.5.2.1.68.1. 説明

##### 3.5.2.1.68.1.1. 型

- array

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| minAge | string | **(オプション)** この MinAge よりも古い namespace に一致するレコードを削除します (例: 1d)。 |
| namespace | string | MinAge より古いログを削除するターゲット namespace (デフォルトは 7d) |

#### 3.5.2.1.69. .spec.logStore.retentionPolicy.audit

##### 3.5.2.1.69.1. 説明

##### 3.5.2.1.69.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| diskThresholdPercent | int | **(オプション)** ES ディスク使用率のしきい値に達した場合、古いインデックスを削除する必要があります (例: 75)。 |
| maxAge | string | **(オプション)** |
| namespaceSpec | array | **(オプション)** 指定された最小期間よりも古いドキュメントを削除する namespace ごとの仕様 |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| pruneNamespacesInterval | string | **(オプション)** 新しい prune-namespaces ジョブを実行する頻度 |

#### 3.5.2.1.70. .spec.logStore.retentionPolicy.audit.namespaceSpec[]

#### 3.5.2.1.70.1. 説明

#### 3.5.2.1.70.1.1. 型

- array

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| minAge | string | **(オプション)** この MinAge よりも古い namespace に一致するレコードを削除します (例: 1d)。 |
| namespace | string | MinAge より古いログを削除するターゲット namespace (デフォルトは 7d) |

#### 3.5.2.1.71. .spec.logStore.retentionPolicy.infra

#### 3.5.2.1.71.1. 説明

#### 3.5.2.1.71.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| diskThresholdPercent | int | **(オプション)** ES ディスク使用率のしきい値に達した場合、古いインデックスを削除する必要があります (例: 75)。 |
| maxAge | string | **(オプション)** |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| namespaceSpec | array | **(オプション)** 指定された最小期間よりも古いドキュメントを削除する namespace ごとの仕様 |
| pruneNamespacesInterval | string | **(オプション)** 新しい prune-namespaces ジョブを実行する頻度 |

#### 3.5.2.1.72. .spec.logStore.retentionPolicy.infra.namespaceSpec[]

##### 3.5.2.1.72.1. 説明

##### 3.5.2.1.72.1.1. 型

- array

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| minAge | string | **(オプション)** この MinAge よりも古い namespace に一致するレコードを削除します (例: 1d)。 |
| namespace | string | MinAge より古いログを削除するターゲット namespace (デフォルトは 7d) |

#### 3.5.2.1.73. .spec.visualization

##### 3.5.2.1.73.1. 説明

これは、ログの視覚化 (Kibana) に関連する情報を含む構造体です。

##### 3.5.2.1.73.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| kibana | object | Kibana Visualization コンポーネントの仕様 |
| type | string | 設定する可視化のタイプ |

#### 3.5.2.1.74. .spec.visualization.kibana

#### 3.5.2.1.74.1. 説明

#### 3.5.2.1.74.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| nodeSelector | object | Pod がスケジュールされているノードを定義します。 |
| proxy | object | Kibana Proxy コンポーネントの仕様 |
| replicas | int | Kibana デプロイメント用にデプロイするインスタンスの数 |
| resources | object | **(オプション)** Kibana のリソース要件 |
| Toleration | array | |

### 3.5.2.1.75. .spec.visualization.kibana.nodeSelector

#### 3.5.2.1.75.1. 説明

#### 3.5.2.1.75.1.1. 型

- object

### 3.5.2.1.76. .spec.visualization.kibana.proxy

#### 3.5.2.1.76.1. 説明

#### 3.5.2.1.76.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| resources | object | |

### 3.5.2.1.77. .spec.visualization.kibana.proxy.resources

#### 3.5.2.1.77.1. 説明

#### 3.5.2.1.77.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| limits | object | **(オプション)** Limits は、許可されるコンピューティングリソースの最大量を示します。 |
| requests | object | **(オプション)** Requests は、必要なコンピューティングリソースの最小量を示します。 |

#### 3.5.2.1.78. .spec.visualization.kibana.proxy.resources.limits

#### 3.5.2.1.78.1. 説明

#### 3.5.2.1.78.1.1. 型

- object

#### 3.5.2.1.79. .spec.visualization.kibana.proxy.resources.requests

#### 3.5.2.1.79.1. 説明

#### 3.5.2.1.79.1.1. 型

- object

#### 3.5.2.1.80. .spec.visualization.kibana.replicas

#### 3.5.2.1.80.1. 説明

#### 3.5.2.1.80.1.1. 型

- int

#### 3.5.2.1.81. .spec.visualization.kibana.resources

#### 3.5.2.1.81.1. 説明

#### 3.5.2.1.81.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| limits | object | **(オプション)** Limits は、許可されるコンピューティングリソースの最大量を示します。 |
| requests | object | **(オプション)** Requests は、必要なコンピューティングリソースの最小量を示します。 |

#### 3.5.2.1.82. .spec.visualization.kibana.resources.limits

##### 3.5.2.1.82.1. 説明

###### 3.5.2.1.82.1.1. 型

- object

#### 3.5.2.1.83. .spec.visualization.kibana.resources.requests

##### 3.5.2.1.83.1. 説明

###### 3.5.2.1.83.1.1. 型

- object

#### 3.5.2.1.84. .spec.visualization.kibana.tolerations[]

##### 3.5.2.1.84.1. 説明

###### 3.5.2.1.84.1.1. 型

- array

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| effect | string | **(オプション)** Effect は、一致する Taint 効果を示します。空の場合は、すべてのテイント効果に一致します。 |
| 鍵 (key) | string | **(オプション)** Key は、Toleration が適用される Taint キーです。空の場合は、すべてのテイントキーに一致します。 |
| operator | string | **(オプション)** Operator は、キーと値の関係を表します。 |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| tolerationSeconds | int | **(オプション)** TolerationSeconds は、Toleration の期間を表します。 |
| value | string | **(オプション)** Value は、Toleration が一致する Taint 値です。 |

#### 3.5.2.1.85. .spec.visualization.kibana.tolerations[].tolerationSeconds

##### 3.5.2.1.85.1. 説明

##### 3.5.2.1.85.1.1. 型

- int

#### 3.5.2.1.86. .status

##### 3.5.2.1.86.1. 説明

ClusterLoggingStatus は、ClusterLogging の監視状態を定義します。

##### 3.5.2.1.86.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| コレクション | object | **(オプション)** |
| conditions | object | **(オプション)** |
| キュレーション | object | **(オプション)** |
| logStore | object | **(オプション)** |
| 可視化 | object | **(オプション)** |

#### 3.5.2.1.87. .status.collection

##### 3.5.2.1.87.1. 説明

##### 3.5.2.1.87.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| logs | object | **(オプション)** |

#### 3.5.2.1.88. .status.collection.logs

##### 3.5.2.1.88.1. 説明

##### 3.5.2.1.88.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| fluentdStatus | object | **(オプション)** |

#### 3.5.2.1.89. .status.collection.logs.fluentdStatus

##### 3.5.2.1.89.1. 説明

##### 3.5.2.1.89.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| clusterCondition | object | **(オプション)** |
| daemonSet | string | **(オプション)** |
| nodes | object | **(オプション)** |
| pods | string | **(オプション)** |

#### 3.5.2.1.90. .status.collection.logs.fluentdStatus.clusterCondition

##### 3.5.2.1.90.1. 説明

**operator-sdk generate crds** は、map-of-slice を許可していません。名前付きタイプを使用する必要があります。

##### 3.5.2.1.90.1.1. 型

- object

#### 3.5.2.1.91. .status.collection.logs.fluentdStatus.nodes

#### 3.5.2.1.91.1. 説明

#### 3.5.2.1.91.1.1. 型

- object

### 3.5.2.1.92. .status.conditions

#### 3.5.2.1.92.1. 説明

#### 3.5.2.1.92.1.1. 型

- object

### 3.5.2.1.93. .status.curation

#### 3.5.2.1.93.1. 説明

#### 3.5.2.1.93.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| curatorStatus | array | **(オプション)** |

### 3.5.2.1.94. .status.curation.curatorStatus[]

#### 3.5.2.1.94.1. 説明

#### 3.5.2.1.94.1.1. 型

- array

| プロパティー | 型 | 説明 |
|---|---|---|
| clusterCondition | object | **(オプション)** |
| cronJobs | string | **(オプション)** |
| スケジュール | string | **(オプション)** |
| suspended | bool | **(オプション)** |

### 3.5.2.1.95. .status.curation.curatorStatus[].clusterCondition

#### 3.5.2.1.95.1. 説明

**operator-sdk generate crds** は、map-of-slice を許可していません。名前付きタイプを使用する必要があります。

#### 3.5.2.1.95.1.1. 型

- object

### 3.5.2.1.96. .status.logStore

#### 3.5.2.1.96.1. 説明

#### 3.5.2.1.96.1.1. 型

- object

| プロパティー | 型 | 説明 |
|---|---|---|
| elasticsearchStatus | array | **(オプション)** |

### 3.5.2.1.97. .status.logStore.elasticsearchStatus[]

#### 3.5.2.1.97.1. 説明

#### 3.5.2.1.97.1.1. 型

- array

| プロパティー | 型 | 説明 |
|---|---|---|
| cluster | object | **(オプション)** |
| clusterConditions | object | **(オプション)** |
| clusterHealth | string | **(オプション)** |
| clusterName | string | **(オプション)** |
| デプロイメント | array | **(オプション)** |
| nodeConditions | object | **(オプション)** |
| nodeCount | int | **(オプション)** |
| pods | object | **(オプション)** |
| replicaSets | array | **(オプション)** |

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| shardAllocationEnabled | string | **(オプション)** |
| statefulSets | array | **(オプション)** |

#### 3.5.2.1.98. .status.logStore.elasticsearchStatus[].cluster

##### 3.5.2.1.98.1. 説明

###### 3.5.2.1.98.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| activePrimaryShards | int | Elasticsearch クラスターのアクティブなプライマリーシャードの数 |
| activeShards | int | Elasticsearch クラスターのアクティブなシャードの数 |
| initializingShards | int | Elasticsearch クラスターの初期化中のシャードの数 |
| numDataNodes | int | Elasticsearch クラスターのデータノードの数 |
| numNodes | int | Elasticsearch クラスターのノードの数 |
| pendingTasks | int | |
| relocatingShards | int | Elasticsearch クラスターの再配置シャードの数 |
| status | string | Elasticsearch クラスターの現在のステータス |
| unassignedShards | int | Elasticsearch クラスターの未割り当てシャードの数 |

#### 3.5.2.1.99. .status.logStore.elasticsearchStatus[].clusterConditions

##### 3.5.2.1.99.1. 説明

#### 3.5.2.1.99.1.1. 型

- object

### 3.5.2.1.100. .status.logStore.elasticsearchStatus[].deployments[]

#### 3.5.2.1.100.1. 説明

#### 3.5.2.1.100.1.1. 型

- array

### 3.5.2.1.101. .status.logStore.elasticsearchStatus[].nodeConditions

#### 3.5.2.1.101.1. 説明

#### 3.5.2.1.101.1.1. 型

- object

### 3.5.2.1.102. .status.logStore.elasticsearchStatus[].pods

#### 3.5.2.1.102.1. 説明

#### 3.5.2.1.102.1.1. 型

- object

### 3.5.2.1.103. .status.logStore.elasticsearchStatus[].replicaSets[]

#### 3.5.2.1.103.1. 説明

#### 3.5.2.1.103.1.1. 型

- array

### 3.5.2.1.104. .status.logStore.elasticsearchStatus[].statefulSets[]

#### 3.5.2.1.104.1. 説明

#### 3.5.2.1.104.1.1. 型

- array

### 3.5.2.1.105. .status.visualization

#### 3.5.2.1.105.1. 説明

#### 3.5.2.1.105.1.1. 型

- object

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| kibanaStatus | array | **(オプション)** |

#### 3.5.2.1.106. .status.visualization.kibanaStatus[]

##### 3.5.2.1.106.1. 説明

##### 3.5.2.1.106.1.1. 型

- array

| プロパティー | 型 | 説明 |
| --- | --- | --- |
| clusterCondition | object | **(オプション)** |
| deployment | string | **(オプション)** |
| pods | string | **(オプション)** 可視化コンポーネントの各 Kibana Pod のステータス |
| replicaSets | array | **(オプション)** |
| replicas | int | **(オプション)** |

#### 3.5.2.1.107. .status.visualization.kibanaStatus[].clusterCondition

##### 3.5.2.1.107.1. 説明

##### 3.5.2.1.107.1.1. 型

- object

#### 3.5.2.1.108. .status.visualization.kibanaStatus[].replicaSets[]

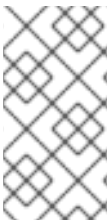
##### 3.5.2.1.108.1. 説明

##### 3.5.2.1.108.1.1. 型

- array

# 第4章 LOGGING 5.5

## 4.1. LOGGING 5.5 リリースノート



**注記**

Red Hat OpenShift のロギングサブシステムは、インストール可能なコンポーネントとして提供され、コアの OpenShift Container Platform とは異なるリリースサイクルを備えています。Red Hat OpenShift Container Platform ライフサイクルポリシー はリリースの互換性を概説しています。

### 4.1.1. Logging 5.5.16

このリリースには、OpenShift Logging バグ修正リリース 5.5.16 が含まれています。

#### 4.1.1.1. バグ修正

- 今回の更新が行われる前は、LokiStack ゲートウェイは承認されたリクエストを非常に広範囲にキャッシュしていました。その結果、誤った認証結果が発生しました。今回の更新により、LokiStack ゲートウェイは詳細にキャッシュを行うようになり、この問題が解決されました。(LOG-4434)

#### 4.1.1.2. CVE

- CVE-2023-3899
- CVE-2023-32360
- CVE-2023-34969

### 4.1.2. Logging 5.5.14

このリリースには、OpenShift Logging バグ修正リリース 5.5.14 が含まれています。

#### 4.1.2.1. バグ修正

- この更新が行われる前は、Vector コレクターに、ログに **thread 'vector-worker' panicked at 'all branches are disabled and there is no else branch', src/kubernetes/reflector.rs:26:9** エラーメッセージでパニックを発生させることがありました。今回の更新により、このエラーは解決されました。(LOG-4279)

#### 4.1.2.2. CVE

- CVE-2023-2828

### 4.1.3. Logging 5.5.13

本リリースには、OpenShift Logging バグ修正リリース 5.5.13 が含まれています。

#### 4.1.3.1. バグ修正

なし。

#### 4.1.3.2. CVE

- CVE-2023-1999
- CVE-2020-24736
- CVE-2022-48281
- CVE-2023-1667
- CVE-2023-2283
- CVE-2023-24329
- CVE-2023-26604
- CVE-2023-28466

### 4.1.4. Logging 5.5.11

このリリースには、OpenShift Logging Bug Fix Release 5.5.11 が含まれています。

#### 4.1.4.1. バグ修正

- この更新が行われるまでは、OpenShift Container Platform Web コンソールでログのヒストグラムをクリックしてドラッグしても時間範囲を選択できませんでした。今回の更新により、クリックとドラッグを使用して時間範囲を正常に選択できるようになりました。(LOG-4102)
- この更新が行われる前は、OpenShift Container Platform Web コンソールで **Show Resources** リンクをクリックしても何の効果もありませんでした。この更新では、ログエントリーごとにリソースの表示を切り替える **リソースの表示** リンクの機能を修正することで、この問題が解決されました。(LOG-4117)

#### 4.1.4.2. CVE

- CVE-2021-26341
- CVE-2021-33655
- CVE-2021-33656
- CVE-2022-1462
- CVE-2022-1679
- CVE-2022-1789
- CVE-2022-2196
- CVE-2022-2663
- CVE-2022-2795
- CVE-2022-3028
- CVE-2022-3239

- CVE-2022-3522
- CVE-2022-3524
- CVE-2022-3564
- CVE-2022-3566
- CVE-2022-3567
- CVE-2022-3619
- CVE-2022-3623
- CVE-2022-3625
- CVE-2022-3627
- CVE-2022-3628
- CVE-2022-3707
- CVE-2022-3970
- CVE-2022-4129
- CVE-2022-20141
- CVE-2022-24765
- CVE-2022-25265
- CVE-2022-29187
- CVE-2022-30594
- CVE-2022-36227
- CVE-2022-39188
- CVE-2022-39189
- CVE-2022-39253
- CVE-2022-39260
- CVE-2022-41218
- CVE-2022-41674
- CVE-2022-42703
- CVE-2022-42720
- CVE-2022-42721
- CVE-2022-42722

- CVE-2022-43750
- CVE-2022-47929
- CVE-2023-0394
- CVE-2023-0461
- CVE-2023-1195
- CVE-2023-1582
- CVE-2023-2491
- CVE-2023-23454
- CVE-2023-27535

### 4.1.5. Logging 5.5.10

このリリースには、OpenShift Logging Bug Fix Release 5.5.10 が含まれています。

#### 4.1.5.1. バグ修正

- 今回の更新以前は、OpenShift Web コンソールのロギングビュープラグインは、LokiStack に到達できなかった場合にのみエラーテキストを表示していました。この更新により、プラグインでは適切なエラーメッセージと、到達できない LokiStack の詳しい修正方法が表示されるようになりました。(LOG-2874)

#### 4.1.5.2. CVE

- CVE-2022-4304
- CVE-2022-4450
- CVE-2023-0215
- CVE-2023-0286
- CVE-2023-0361
- CVE-2023-23916

### 4.1.6. Logging 5.5.9

このリリースには、OpenShift Logging Bug Fix Release 5.5.9 が含まれています。

#### 4.1.6.1. バグ修正

- この更新の前は、Fluentd コレクターの問題により、**/var/log/auth-server/audit.log** に保存されている OAuth ログインイベントがキャプチャーされませんでした。これにより、OAuth サービスからのログインイベントの収集が不完全になりました。今回の更新により、Fluentd コレクターは、予想どおり、**/var/log/auth-server/audit.log** に保存されているものを含め、OAuth サービスからすべてのログインイベントをキャプチャーすることで、この問題を解決するようになりました。(LOG-3730)

- この更新の前は、構造化された解析が有効になっていて、メッセージが複数の宛先に転送された場合に、それらはディープコピーされませんでした。これにより、構造化されたメッセージを含む一部の受信ログが生成されましたが、その他のログは生成されませんでした。今回の更新により、JSON 解析の前にメッセージをディープコピーするように設定生成が変更されました。その結果、複数の宛先に転送された場合でも、すべての受信ログに構造化メッセージが含まれるようになりました。(LOG-3767)

#### 4.1.6.2. CVE

- CVE-2022-4304
- CVE-2022-4450
- CVE-2022-41717
- CVE-2023-0215
- CVE-2023-0286
- CVE-2023-0767
- CVE-2023-23916

### 4.1.7. Logging 5.5.8

このリリースには、OpenShift Logging Bug Fix Release 5.5.8 が含まれています。

#### 4.1.7.1. バグ修正

- 今回の更新の前は、コレクターが **level** フィールドを設定する方法にエラーがあったため、**priority** フィールドが **systemd** ログから欠落していました。今回の更新により、これらのフィールドが正しく設定され、問題が解決されました。(LOG-3630)

#### 4.1.7.2. CVE

- CVE-2020-10735
- CVE-2021-28861
- CVE-2022-2873
- CVE-2022-4415
- CVE-2022-24999
- CVE-2022-40897
- CVE-2022-41222
- CVE-2022-41717
- CVE-2022-43945
- CVE-2022-45061
- CVE-2022-48303

### 4.1.8. Logging 5.5.7

このリリースには、OpenShift Logging バグ修正リリース 5.5.7 が含まれます。

#### 4.1.8.1. バグ修正

- この更新の前は、ブール式と組み合わせてラベルフィルターを使用した場合、LokiStack ゲートウェイラベルエンフォーサーが有効な LogQL クエリーの解析エラーを生成していました。今回の更新により、LokiStack LogQL の実装がブール式を使用したラベルフィルターをサポートするようになり、問題が解決されました。(LOG-3534)

- この更新の前は、**ClusterLogForwarder** カスタムリソース (CR) が syslog 出力の TLS 認証情報を Fluentd に渡さなかったため、転送中にエラーが発生していました。今回の更新により、認証情報が Fluentd に正しく渡されるようになり、問題が解決されました。(LOG-3533)

#### 4.1.8.2. CVE

CVE-2021-46848CVE-2022-3821CVE-2022-35737CVE-2022-42010CVE-2022-42011CVE-2022-42012CVE-2022-42898CVE-2022-43680

### 4.1.9. Logging 5.5.6

このリリースには、OpenShift Logging バグ修正リリース 5.5.6 が含まれます。

#### 4.1.9.1. バグ修正

- この更新の前は、Pod セキュリティーアドミッションコントローラーがラベル **podSecurityLabelSync = true** を **openshift-logging** namespace に追加していました。これにより、指定のセキュリティーラベルが上書きされ、その結果、コレクター Pod が起動しなくなりました。今回の更新により、ラベル **podSecurityLabelSync = false** がセキュリティーラベルを保持するようになります。コレクター Pod は期待どおりにデプロイされます。(LOG-3340)

- この更新の前は、コンソールビュープラグインがクラスターで有効になっていない場合でも、Operator はコンソールビュープラグインをインストールしていました。これにより、Operator がクラッシュしました。今回の更新により、クラスターのアカウントでコンソールビューが有効になっていない場合、Operator は正常に機能し、コンソールビューをインストールしなくなりました。(LOG-3407)

- この更新の前は、Elasticsearch デプロイメントのステータスが更新されないリグレッションに対応するための以前の修正が原因で、**Red Hat Elasticsearch Operator** がデプロイされていない限り、Operator がクラッシュしていました。今回の更新により、その修正が元に戻されたため、Operator は安定した状態になりました。ただし、報告されたステータスに関連する以前の問題が再び発生するようになりました。(LOG-3428)

- この更新の前は、Loki Operator は、選択されたスタックサイズに関係なく、LokiStack ゲートウェイのレプリカを 1 つだけデプロイしていました。今回の更新により、選択したサイズに応じてレプリカの数が正しく設定されるようになりました。(LOG-3478)

- この更新の前は、複数のラベルキーに同じ接頭辞があり、一部のキーにドットが含まれていると、Elasticsearch に書き込まれたレコードで障害が発生していました。今回の更新により、ラベルキーのドットがアンダースコアに置き換えられ、問題が解決されました。(LOG-3341)

- この更新の前は、ロギングビュープラグインに、OpenShift Container Platform の特定のバージョンと互換性のない機能が含まれていました。今回の更新で、プラグインの正しいリリースストリームにより、この問題は解決されます。(LOG-3467)

- この更新の前は、**ClusterLogForwarder** カスタムリソースの調整で、1 つ以上のパイプラインの低下ステータスが誤って報告され、コレクター Pod が 8 - 10 秒ごとに再起動していました。今回の更新により、**ClusterLogForwarder** カスタムリソースの調整が正しく処理されるようになり、問題が解決されました。(LOG-3469)

- この変更の前は、ClusterLogForwarder カスタムリソースの **outputDefaults** フィールドの仕様は、宣言されたすべての Elasticsearch 出力タイプに設定を適用していました。今回の変更により、拡張仕様に一致するように動作が修正され、デフォルトのマネージド Elasticsearch ストアにのみ設定が適用されるようになりました。(LOG-3342)

- この更新の前は、OpenShift CLI (oc) がそのキャッシュを構築するために書き込み権限を持つフォルダーを必要とするため、OpenShift CLI (oc) の **must-gather** スクリプトが完了しませんでした。今回の更新により、OpenShift CLI (oc) にフォルダーへの書き込み権限が付与され、**must-gather** スクリプトが正常に完了するようになりました。(LOG-3472)

- この更新の前は、Loki Operator Webhook サーバーが TLS エラーを引き起こしていました。今回の更新により、Loki Operator Webhook PKI は Operator Lifecycle Manager の動的 Webhook 管理によって管理されるようになり、問題が解決されました。(LOG-3511)

#### 4.1.9.2. CVE

- CVE-2021-46848
- CVE-2022-2056
- CVE-2022-2057
- CVE-2022-2058
- CVE-2022-2519
- CVE-2022-2520
- CVE-2022-2521
- CVE-2022-2867
- CVE-2022-2868
- CVE-2022-2869
- CVE-2022-2953
- CVE-2022-2964
- CVE-2022-4139
- CVE-2022-35737
- CVE-2022-42010
- CVE-2022-42011
- CVE-2022-42012
- CVE-2022-42898

- CVE-2022-43680

### 4.1.10. Logging 5.5.5

このリリースには、OpenShift Logging バグ修正リリース 5.5.5 が含まれます。

#### 4.1.10.1. バグ修正

- この更新の前は、Kibana の OAuth cookie の有効期限は **24h** に固定されていたため、**accessTokenInactivityTimeout** フィールドが **24h** 未満の値に設定されていると、Kibana で 401 エラーが発生していました。今回の更新により、Kibana の OAuth cookie の有効期限が **accessTokenInactivityTimeout** に同期され、デフォルト値は **24h** になります。(LOG-3305)

- この更新の前は、Vector は、JSON 解析が有効になっている場合に、**structuredTypeKey** または **structuredTypeName** の値も定義せずにメッセージフィールドを解析していました。今回の更新により、構造化ログを Elasticsearch に書き込むときに、**structuredTypeKey** または **structuredTypeName** のいずれかに値が必要になりました。(LOG-3284)

- この更新の前は、**FluentdQueueLengthIncreasing** アラート式から返された一連のラベルにカーディナリティの問題があった場合に、このアラートが発生しない可能性がありました。今回の更新により、アラートに必要なラベルのみが含まれるようにラベルが削減されました。(LOG-3226)

- この更新の前は、Loki は、切断されたクラスター内の外部ストレージへのアクセスをサポートしていませんでした。今回の更新では、これらの接続をサポートするために、プロキシー環境変数とプロキシーの信頼できる CA バンドルがコンテナーイメージに含まれています。(LOG-2860)

- 今回の更新前は、OpenShift Container Platform Web コンソールのユーザーは、Loki の CA 証明書を含む **ConfigMap** オブジェクトを選択できなかったため、Pod が CA なしで動作していました。今回の更新により、Web コンソールユーザーは設定マップを選択できるようになり、問題が解決されました。(LOG-3310)

- この更新の前に、CA キーは CA を Loki にマウントするためのボリューム名として使用されていたため、CA キーに非準拠の文字 (ドットなど) が含まれているとエラー状態が発生していました。今回の更新により、ボリューム名が内部文字列に標準化され、問題が解決されました。(LOG-3332)

#### 4.1.10.2. CVE

- CVE-2016-3709
- CVE-2020-35525
- CVE-2020-35527
- CVE-2020-36516
- CVE-2020-36558
- CVE-2021-3640
- CVE-2021-30002
- CVE-2022-0168

- CVE-2022-0561
- CVE-2022-0562
- CVE-2022-0617
- CVE-2022-0854
- CVE-2022-0865
- CVE-2022-0891
- CVE-2022-0908
- CVE-2022-0909
- CVE-2022-0924
- CVE-2022-1016
- CVE-2022-1048
- CVE-2022-1055
- CVE-2022-1184
- CVE-2022-1292
- CVE-2022-1304
- CVE-2022-1355
- CVE-2022-1586
- CVE-2022-1785
- CVE-2022-1852
- CVE-2022-1897
- CVE-2022-1927
- CVE-2022-2068
- CVE-2022-2078
- CVE-2022-2097
- CVE-2022-2509
- CVE-2022-2586
- CVE-2022-2639
- CVE-2022-2938
- CVE-2022-3515

- CVE-2022-20368
- CVE-2022-21499
- CVE-2022-21618
- CVE-2022-21619
- CVE-2022-21624
- CVE-2022-21626
- CVE-2022-21628
- CVE-2022-22624
- CVE-2022-22628
- CVE-2022-22629
- CVE-2022-22662
- CVE-2022-22844
- CVE-2022-23960
- CVE-2022-24448
- CVE-2022-25255
- CVE-2022-26373
- CVE-2022-26700
- CVE-2022-26709
- CVE-2022-26710
- CVE-2022-26716
- CVE-2022-26717
- CVE-2022-26719
- CVE-2022-27404
- CVE-2022-27405
- CVE-2022-27406
- CVE-2022-27950
- CVE-2022-28390
- CVE-2022-28893
- CVE-2022-29581

- CVE-2022-30293
- CVE-2022-34903
- CVE-2022-36946
- CVE-2022-37434
- CVE-2022-39399

### 4.1.11. Logging 5.5.4

このリリースには、OpenShift Logging Bug Fix Release 5.5.4 が含まれています。

#### 4.1.11.1. バグ修正

- この更新の前は、ロギングビュープラグインのクエリーパーサーのエラーにより、クエリーに中かっこ **{}** が含まれていると、ログクエリーの一部が消えていました。これにより、クエリーが無効になり、有効なクエリーに対してエラーが返されました。今回の更新により、パーサーはこれらのクエリーを正しく処理するようになりました。(LOG-3042)
- この更新の前は、Elasticsearch または Kibana デプロイメントのステータスが変更されている間に、Operator がコレクターデーモンセットの削除と再作成のループに入る可能性がありました。今回の更新では、Operator のステータス処理が修正され、問題が解決されました。(LOG-3049)
- この更新の前は、Vector のコレクターの実装をサポートするためにアラートが実装されていませんでした。この変更により、Vector アラートが追加され、選択したコレクターの実装に応じて個別のアラートがデプロイメントされます。(LOG-3127)
- この更新の前は、Elasticsearch Operator のシークレット作成コンポーネントが内部シークレットを常に変更していました。今回の更新により、既存のシークレットが適切に処理されるようになりました。(LOG-3138)
- この更新の前に、ログ **must-gather** スクリプトの以前のリファクタリングにより、アーティファクトの予想される場所が削除されました。この更新により、アーティファクトを **/must-gather** フォルダーに書き込むという変更が元に戻ります。(LOG-3213)
- この更新の前は、特定のクラスターで、Prometheus エクスポーターが IPv6 ではなく IPv4 にバインドしていました。この更新後、Fluentd は IP バージョンを検出し、IPv4 の場合は **0.0.0.0**、IPv6 の場合は **[::]** にバインドします。(LOG-3162)

#### 4.1.11.2. CVE

- CVE-2020-35525
- CVE-2020-35527
- CVE-2022-0494
- CVE-2022-1353
- CVE-2022-2509
- CVE-2022-2588

- CVE-2022-3515
- CVE-2022-21618
- CVE-2022-21619
- CVE-2022-21624
- CVE-2022-21626
- CVE-2022-21628
- CVE-2022-23816
- CVE-2022-23825
- CVE-2022-29900
- CVE-2022-29901
- CVE-2022-32149
- CVE-2022-37434
- CVE-2022-40674

### 4.1.12. Logging 5.5.3

このリリースには、OpenShift Logging バグ修正リリース 5.5.3 が含まれます。

#### 4.1.12.1. バグ修正

- この更新前は、構造化されたメッセージを含むログエントリーに元のメッセージフィールドが含まれていたため、エントリーが大きくなりました。この更新により、構造化ログのメッセージフィールドが削除され、増加したサイズが縮小されます。(LOG-2759)
- この更新の前に、コレクター設定は、**Collector**、**default-log-store**、および **visualization** Pod からログを除外していましたが、**.gz** ファイルにアーカイブされたログを除外できませんでした。今回の更新により、**collector**、**default-log-store**、および **visualization** Pod の **.gz** ファイルとして保存されたアーカイブログも除外されます。(LOG-2844)
- この更新の前は、使用できない Pod へのリクエストがゲートウェイ経由で送信された場合、中断を警告するアラートはありませんでした。今回の更新により、ゲートウェイで書き込みまたは読み取り要求の完了に問題が発生した場合に、個別のアラートが生成されます。(LOG-2884)
- この更新の前は、値が参照によってパイプラインを通過したため、Pod メタデータは fluent プラグインによって変更される可能性がありました。この更新により、各ログメッセージが Pod メタデータのコピーを確実に受信するようになり、各メッセージが個別に処理されるようになりました。(LOG-3046)
- この更新の前に、OpenShift コンソールログビューで **不明な** 重大度を選択すると、**level=unknown** 値のログが除外されていました。今回の更新により、レベルのないログと **level=unknown** の値を持つログが、**不明な** 重大度でフィルタリングすると表示されるようになりました。(LOG-3062)
- この更新の前は、Elasticsearch に送信されたログレコードには、ログを送信する必要があるイ

ンデックスの名前を含む **write-index** という名前の追加フィールドがありました。このフィールドは、データモデルの一部ではありません。この更新後、このフィールドは送信されなくなりました。(LOG-3075)

- 新しい組み込み Pod Security Admission Controller の導入により、グローバルまたは namespace レベルで定義された強制セキュリティー規格に従って設定されていない Pod は実行できません。今回の更新により、Operator と Collector は特権実行を許可し、セキュリティー監査の警告やエラーなしで実行できるようになりました。(LOG-3077)
- この更新の前に、LokiStack をデフォルトのログストレージとして使用する場合、Operator は **ClusterLogForwarder** カスタムリソースで定義されたカスタム出力を削除しました。今回の更新により、Operator は **ClusterLogForwarder** カスタムリソースの処理時にカスタム出力をデフォルト出力とマージします。(LOG-3095)

#### 4.1.12.2. CVE

- CVE-2015-20107
- CVE-2022-0391
- CVE-2022-2526
- CVE-2022-21123
- CVE-2022-21125
- CVE-2022-21166
- CVE-2022-29154
- CVE-2022-32206
- CVE-2022-32208
- CVE-2022-34903

### 4.1.13. Logging 5.5.2

このリリースには、OpenShift Logging バグ修正リリース 5.5.2 が含まれます。

#### 4.1.13.1. バグ修正

- 今回の更新以前は、Fluentd コレクターのアラートルールは OpenShift Container Platform モニタリングスタイルのガイドラインに準拠していませんでした。今回の更新により、これらのアラートが namespace ラベルを含むように変更され、問題が解決されました。(LOG-1823)
- この更新の前は、インデックス名に複数のハイフン文字が含まれていると、インデックス管理ロールオーバースクリプトが新しいインデックス名を生成できませんでした。今回の更新により、インデックス名が正しく生成されるようになりました。(LOG-2644)
- この更新の前は、Kibana ルートは証明書が存在しない状態で **caCertificate** 値を設定していました。今回の更新により、**caCertificate** 値が設定されなくなりました。(LOG-2661)
- この更新の前は、コレクターの依存関係の変更により、未使用のパラメーターに対して警告メッセージが発行されていました。今回の更新で、未使用の設定パラメーターを削除すると、問題が解決されます。(LOG-2859)

- この更新の前に、Loki Operator が作成したデプロイメント用に作成された Pod は、Operator が実行されているクラスターでそのようなノードが利用可能な場合、Linux 以外のオペレーティングシステムのノードで誤ってスケジュールされていました。今回の更新により、Operator は追加のノードセレクターを Pod 定義に割り当て、Linux ベースのノードでのみ Pod をスケジュールできるようにします。(LOG-2895)

- この更新の前は、LokiStack ゲートウェイの LogQL パーサーの問題により、OpenShift コンソールのログビューはログを重大度でフィルタリングしませんでした。今回の更新では、パーサーの修正により問題が解決され、OpenShift コンソールのログビューは重大度でフィルタリングできるようになりました。(LOG-2908)

- この更新の前に、Fluentd コレクタープラグインのリファクタリングにより、イベントのタイムスタンプフィールドが削除されました。この更新により、イベントの受信時刻をソースとするタイムスタンプフィールドが復元されます。(LOG-2923)

- この更新の前は、監査ログに **level** フィールドがないため、ベクターログでエラーが発生していました。今回の更新で、監査ログレコードに **level** フィールドが追加され、問題が解決されました。(LOG-2961)

- 今回の更新の前は、Kibana カスタムリソースを削除した場合、OpenShift Container Platform Web コンソールは引き続き Kibana へのリンクを表示していました。今回の更新で、Kibana カスタムリソースを削除すると、そのリンクも削除されます。(LOG-3053)

- この更新の前は、**ClusterLogForwarder** カスタムリソースに JSON 解析が定義されている場合、各ロールオーバージョブで空のインデックスが作成されていました。今回の更新により、新しいインデックスは空ではなくなりました。(LOG-3063)

- この更新の前に、ユーザーが Loki Operator 5.5 への更新後に LokiStack を削除すると、もともと Loki Operator 5.4 によって作成されたリソースが残りました。今回の更新により、リソースの所有者参照は 5.5 LokiStack を指します。(LOG-2945)

- この更新の前は、ユーザーはアクセス権を持つ namespace のアプリケーションログを表示できませんでした。今回の更新により、Loki Operator はクラスターロールとクラスターロールバインディングを自動的に作成し、ユーザーがアプリケーションログを読み取れるようにします。(LOG-2918)

- この更新の前は、cluster-admin 権限を持つユーザーは、ログコンソールを使用してインフラストラクチャーと監査ログを適切に表示できませんでした。今回の更新により、認可チェックが拡張され、cluster-admin および dedicated-admin グループのユーザーも管理者として認識されるようになりました。(LOG-2970)

#### 4.1.13.2. CVE

- CVE-2015-20107
- CVE-2022-0391
- CVE-2022-21123
- CVE-2022-21125
- CVE-2022-21166
- CVE-2022-29154
- CVE-2022-32206

- CVE-2022-32208
- CVE-2022-34903

### 4.1.14. Logging 5.5.1

このリリースには OpenShift Logging バグ修正リリース 5.5.1 が含まれます。

#### 4.1.14.1. 機能拡張

- 今回の機能拡張により、Logging Console プラグインが使用されている場合に、**Aggregated Logs** タブが OpenShift Container Platform Web コンソールの **Pod Details** ページに追加されました。この拡張機能は OpenShift Container Platform 4.10 以降でのみ利用できます。(LOG-2647)
- 今回の拡張機能により、ログ転送の出力オプションとして Google Cloud Logging が追加されました。(LOG-1482)

#### 4.1.14.2. バグ修正

- 今回の更新以前は、Operator は Pod が準備状態にあることを確認しないことが原因で、クラスターの再起動時にクラスターが動作不能な状態になりました。今回の更新により、Operator は、再起動中に新しい Pod を準備完了としてマークしてから新しい Pod に移動を続けることで、問題を解決します。(LOG-2745)
- 今回の更新以前は、Fluentd は Kubernetes プラットフォームがログファイルをローテーションしたことを認識しない場合があり、ログメッセージを読み取らなくなっていました。今回の更新で、アップストリームの開発チームが提案する設定パラメーターを設定することにより修正されています。(LOG-2995)
- 今回の更新以前は、複数行のエラー検出機能が追加されたことが原因で、内部ルーティングが変更され、レコードが間違った宛先に転送されていました。今回の更新により、内部ルーティングが正しくなりました。(LOG-2801)
- 今回の更新前は、OpenShift Container Platform Web コンソールの更新間隔を変更すると、**Query** フィールドが空の場合にはエラーが発生していました。今回の更新で、**Query** フィールドが空の場合に、間隔の変更が選択不可能になりました。(LOG-2917)

#### 4.1.14.3. CVE

- CVE-2022-1705
- CVE-2022-2526
- CVE-2022-29154
- CVE-2022-30631
- CVE-2022-32148
- CVE-2022-32206
- CVE-2022-32208

### 4.1.15. Logging 5.5.0

このリリースには、OpenShift Logging Bug Fix Release 5.5.0 が含まれています。

#### 4.1.15.1. 機能拡張

- 今回の更新では、構造化ログを同じ Pod 内の異なるコンテナーからさまざまなインデックスに転送できるようになりました。この機能を使用するには、複数コンテナーのサポートを使用してパイプラインを設定し、Pod にアノテーションを付ける必要があります。(LOG-1296)

**重要**

ログの JSON 形式は、アプリケーションによって異なります。作成するインデックスが多すぎるとパフォーマンスに影響するため、この機能の使用は、互換性のない JSON 形式のログのインデックスの作成に限定してください。クエリーを使用して、さまざまな namespace または互換性のある JSON 形式のアプリケーションからログを分離します。

- 今回の更新では、Kubernetes 共通ラベルである **app.kubernetes.io/component**、**app.kubernetes.io/managed-by**、**app.kubernetes.io/part-of**、および **app.kubernetes.io/version** を使用して、Elasticsearch 出力でログをフィルタリングできます。Elasticsearch 以外の出力タイプでは、**kubernetes.labels** に含まれるすべてのラベルを使用できます。(LOG-2388)

- 今回の更新では、AWS Security Token Service (STS) が有効になっているクラスターは、STS 認証を使用してログを Amazon CloudWatch に転送できます。(LOG-1976)

- 今回の更新では、'LokiOperator' Operator および Vector コレクターがテクニカルプレビュー機能から一般提供機能 (GA) に移行します。以前のリリースとの完全な機能パリティーに関しては作業中で、一部の API はテクニカルプレビュー機能のままです。詳細は、**LokiStack を使用したロギング** セクションを参照してください。

#### 4.1.15.2. バグ修正

- この更新の前は、ログを Amazon CloudWatch に転送するように設定されたクラスターが、拒否されたログファイルを一時ストレージに書き込んでいたため、時間の経過とともにクラスターが不安定になりました。今回の更新により、すべてのストレージオプションの一括バックアップが無効になり、問題が解決されました。(LOG-2746)

- 今回の更新以前に、Operator は、非推奨で OpenShift Container Platform の今後のバージョンで削除予定の API のバージョンを一部使用していました。今回の更新により、依存関係がサポート対象の API バージョンに移動されます。(LOG-2656)

- 今回の更新以前は、複数行エラー検出用に設定された複数の **ClusterLogForwarder** パイプラインにより、コレクターが **crashloopbackoff** エラー状態になりました。今回の更新により、複数の設定セクションに同じ一意の ID が使用される問題が修正されます。(LOG-2241)

- この更新の前は、コレクターは UTF-8 以外の記号を Elasticsearch ストレージログに保存できませんでした。今回の更新で、コレクターは UTF-8 以外の記号をエンコードし、問題を解決しました。(LOG-2203)

- 今回の更新以前は、ラテン文字以外の文字が Kibana で正しく表示されませんでした。今回の更新により、Kibana はすべての有効な UTF-8 シンボルを正しく表示します。(LOG-2784)

#### 4.1.15.3. CVE

- CVE-2021-38561

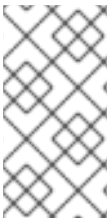
- CVE-2022-1012
- CVE-2022-1292
- CVE-2022-1586
- CVE-2022-1785
- CVE-2022-1897
- CVE-2022-1927
- CVE-2022-2068
- CVE-2022-2097
- CVE-2022-21698
- CVE-2022-30631
- CVE-2022-32250

## 4.2. LOGGING 5.5 を使い始める

ロギングデプロイメントプロセスの概要は、参照しやすいように提供されています。完全なドキュメントに代わるものではありません。新規インストールの場合は、**Vector** と **LokiStack** を推奨します。

**注記**

Logging バージョン 5.5 の時点で、**Fluentd** または **Vector** コレクター実装から選択するオプション、およびログストアとして **Elasticsearch** または **LokiStack** を選択するオプションがあります。ログのドキュメントは、これらの基本的なコンポーネントの変更を反映するために更新中です。

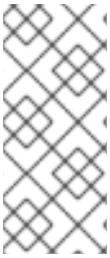


**注記**

Red Hat OpenShift のロギングサブシステムは、インストール可能なコンポーネントとして提供され、コアの OpenShift Container Platform とは異なるリリースサイクルを備えています。Red Hat OpenShift Container Platform ライフサイクルポリシー はリリースの互換性を概説しています。

**前提条件**

- LogStore 設定: **Elasticsearch** または **LokiStack**
- コレクターの実装設定: **Fluentd** または **Vector**
- ログ転送出力の認証情報



**注記**

Logging バージョン 5.4.3 の時点で、Elasticsearch Operator は非推奨であり、今後のリリースで削除される予定です。Red Hat は、この機能に対して現在のリリースライフサイクル中にバグ修正とサポートを提供しますが、拡張機能の提供はなく、この機能は今後削除される予定です。Elasticsearch Operator を使用してデフォルトのログストレージを管理する代わりに、Loki Operator を使用できます。

1. 使用するログストアの Operator をインストールします。
   - **Elasticsearch** の場合は、**OpenShift Elasticsearch Operator** をインストールします。
   - **LokiStack** の場合は、**Loki Operator** をインストールします。
     - **LokiStack** カスタムリソース (CR) インスタンスを作成します。
2. **Red Hat OpenShift Logging Operator** をインストールします。
3. **ClusterLogging** カスタムリソース (CR) インスタンスを作成します。
   a. コレクターの実装を選択します。

**注記**

Logging バージョン 5.6 の時点で、Fluentd は非推奨であり、今後のリリースで削除される予定です。Red Hat は、この機能に対して現在のリリースライフサイクル中にバグ修正とサポートを提供しますが、拡張機能の提供はなく、この機能は今後削除される予定です。Fluentd の代わりに、Vector を使用できます。

4. **ClusterLogForwarder** カスタムリソース (CR) インスタンスを作成します。
5. 選択した出力パイプラインのシークレットを作成します。

## 4.3. ロギングアーキテクチャーについて

ロギングサブシステムは、次の論理コンポーネントで設定されています。

- **Collector** - 各ノードからコンテナーログデータを読み取り、ログデータを設定済みの出力に転送します。
- **Store** - 分析用のログデータを保存します。フォワーダーのデフォルト出力。
- **Visualization** - 保存されたログを検索、クエリー、および表示するためのグラフィカルインターフェイス。

これらのコンポーネントは、Operator とカスタムリソース (CR) YAML ファイルによって管理されます。

Red Hat OpenShift のロギングサブシステムは、コンテナーログとノードログを収集します。これらは次のタイプに分類されます。

- **application** - 非インフラストラクチャーコンテナーによって生成されたコンテナーログ。
- **infrastructure** - namespace **kube-*** および **openshift-*** からのコンテナーログ、および **journald** からのノードログ。

- **audit** - 有効な場合は、**auditd**、**kube-apiserver**、**openshift-apiserver**、および **ovn** からのログ。

ロギングコレクターは、Pod を各 OpenShift Container Platform ノードにデプロイするデーモンセットです。システムおよびインフラストラクチャーのログは、オペレーティングシステム、コンテナーランタイム、および OpenShift Container Platform からの journald ログメッセージによって生成されます。

コンテナーログは、クラスターで実行されている Pod で実行されているコンテナーによって生成されます。各コンテナーは個別のログストリームを生成します。コレクターは、これらのソースからログを収集し、**ClusterLogForwarder** カスタムリソースで設定されているように、それらを内部または外部に転送します。

## 4.4. ロギングデプロイメントの管理

### 4.4.1. Web コンソールを使用した Red Hat OpenShift Logging Operator のデプロイ

OpenShift Container Platform Web コンソールを使用して、Red Hat OpenShift Logging Operator をデプロイできます。

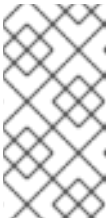
**前提条件**

Red Hat OpenShift のロギングサブシステムは、インストール可能なコンポーネントとして提供され、コアの OpenShift Container Platform とは異なるリリースサイクルを備えています。Red Hat OpenShift Container Platform ライフサイクルポリシー はリリースの互換性を概説しています。

**手順**

OpenShift Container Platform Web コンソールを使用して、Red Hat OpenShift Logging Operator をデプロイするには、以下を実行します。

1. Red Hat OpenShift Logging Operator をインストールします。
   a. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
   b. **Filter by keyword** フィールドに **Logging** と入力します。
   c. 利用可能な Operator のリストから **Red Hat OpenShift Logging** を選択し、**Install** をクリックします。
   d. **Update Channel** として **stable** または **stable-5.y** を選択します。



**注記**

**stable** チャネルは、Logging の最新リリースを対象とする更新のみを提供します。以前のリリースの更新を引き続き受信するには、サブスクリプションチャネルを **stable-X** (**X** はインストールしたログのバージョン) に変更する必要があります。

   e. **Installation Mode** で **A specific namespace on the cluster**が選択されていることを確認します。

f. **Operator recommended namespace** が **Installed Namespace** で **openshift-logging** になっていることを確認します。

g. **Enable Operator recommended cluster monitoring on this Namespace**を選択します。

h. **Update approval** のオプションを選択します。

- **Automatic** オプションを使用すると、Operator Lifecycle Manager (OLM) は、新しいバージョンが利用可能になった際、Operator を自動的に更新できます。
- **Manual** オプションでは、適切な認証情報を持つユーザーが Operator の更新を承認する必要があります。

i. Console プラグインの **Enable** または **Disable** を選択します。

j. **Install** をクリックします。

2. **Operators → Installed Operators** ページに切り替えて、**Red Hat OpenShift Logging Operator** がインストールされていることを確認します。

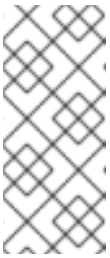
a. **Red Hat OpenShift Logging** が **Status** が **Succeeded** の状態で **openshift-logging** プロジェクトにリスト表示されていることを確認します。

3. **ClusterLogging** インスタンスを作成します。

**注記**

Web コンソールのフォームビューには、使用可能なすべてのオプションが含まれているわけではありません。セットアップを完了するには、**YAML ビュー** を推奨します。

a. **collection** セクションで、コレクターの実装を選択します。



**注記**

Logging バージョン 5.6 の時点で、Fluentd は非推奨であり、今後のリリースで削除される予定です。Red Hat は、この機能に対して現在のリリースライフサイクル中にバグ修正とサポートを提供しますが、拡張機能の提供はなく、この機能は今後削除される予定です。Fluentd の代わりに、Vector を使用できます。

b. **logStore** セクションで、タイプを選択します。

**注記**

Logging バージョン 5.4.3 の時点で、Elasticsearch Operator は非推奨であり、今後のリリースで削除される予定です。Red Hat は、この機能に対して現在のリリースライフサイクル中にバグ修正とサポートを提供しますが、拡張機能の提供はなく、この機能は今後削除される予定です。Elasticsearch Operator を使用してデフォルトのログストレージを管理する代わりに、Loki Operator を使用できます。

c. **Create** をクリックします。

### 4.4.2. Web コンソールを使用した Loki Operator のデプロイ

OpenShift Container Platform コンソールを使用して Loki Operator をインストールできます。

**前提条件**

- 対応ログストア (AWS S3、Google Cloud Storage、Azure、Swift、Minio、OpenShift Data Foundation)

**手順**

OpenShift Container PlatformWeb コンソールを使用して Loki Operator をインストールするには:

1. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
2. **Filter by keyword** フィールドに **Loki** と入力します。
   a. 使用可能な Operator のリストから **Loki Operator** を選択し、**Install** をクリックします。
3. **Update Channel** として **stable** または **stable-5.y** を選択します。

**注記**

**stable** チャネルは、Logging の最新リリースを対象とする更新のみを提供します。以前のリリースの更新を引き続き受信するには、サブスクリプションチャネルを **stable-X** (**X** はインストールしたログのバージョン) に変更する必要があります。

4. **Installation Mode** で **All namespaces on the cluster**が選択されていることを確認します。
5. **openshift-operators-redhat** が **Installed Namespace** で選択されていることを確認します。
6. **Enable Operator recommended cluster monitoring on this Namespace**を選択します。このオプションは、namespace オブジェクトに **openshift.io/cluster-monitoring: "true"** ラベルを設定します。クラスターモニタリングが **openshift-operators-redhat** namespace を収集できるように、このオプションを選択する必要があります。
7. **Update approval** のオプションを選択します。
   - **Automatic** オプションを使用すると、Operator Lifecycle Manager (OLM) は、新しいバージョンが利用可能になった際、Operator を自動的に更新できます。
   - **Manual** オプションでは、適切な認証情報を持つユーザーが Operator の更新を承認する必要があります。
8. **Install** をクリックします。
9. **Operators → Installed Operators** ページに切り替えて、**LokiOperator** がインストールされていることを確認します。
   a. **LokiOperator** の全プロジェクトの **Status** が **Succeeded** として表示されているようにします。
10. **access_key_id** および **access_key_secret** フィールドを使用して、認証情報を指定し、**bucketnames**、**endpoint**、および **region** を指定して、オブジェクトストレージの場所を定義する **Secret** YAML ファイルを作成します。次の例では、AWS が使用されています。

```
apiVersion: v1
kind: Secret
metadata:
  name: logging-loki-s3
  namespace: openshift-logging
stringData:
  access_key_id: AKIAIOSFODNN7EXAMPLE
  access_key_secret: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
  bucketnames: s3-bucket-name
  endpoint: https://s3.eu-central-1.amazonaws.com
  region: eu-central-1
```

11. **Details** タブの LokiStack の下にある **Create instance** を選択します。次に、**YAML view** を選択します。次のテンプレートに貼り付け、必要に応じて値を置き換えます。

```
apiVersion: loki.grafana.com/v1
kind: LokiStack
metadata:
  name: logging-loki (1)
  namespace: openshift-logging
spec:
  size: 1x.small (2)
  storage:
    schemas:
    - version: v12
      effectiveDate: '2022-06-01'
    secret:
      name: logging-loki-s3 (3)
      type: s3 (4)
  storageClassName: <storage_class_name> (5)
  tenants:
    mode: openshift-logging
```

(1) 名前は、**logging-loki** にする必要があります。

(2) Loki のデプロイメントサイズを選択します。

(3) ログストレージに使用するシークレットを定義します。

(4) 対応するストレージタイプを定義します。

(5) 一時ストレージ用の既存のストレージクラスの名前を入力します。最適なパフォーマンスを得るには、ブロックストレージを割り当てるストレージクラスを指定します。クラスターで使用可能なストレージクラスは、**oc get storageclasses** で一覧表示できます。

    a. 設定を適用します。

```
oc apply -f logging-loki.yaml
```

12. **ClusterLogging** CR を作成または編集します。

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogging
```

```
  metadata:
    name: instance
    namespace: openshift-logging
  spec:
    managementState: Managed
    logStore:
      type: lokistack
      lokistack:
        name: logging-loki
      collection:
        type: vector
```

a. 設定を適用します。

```
oc apply -f cr-lokistack.yaml
```

### 4.4.3. CLI を使用した OperatorHub からのインストール

OpenShift Container Platform Web コンソールを使用する代わりに、CLI を使用して OperatorHub から Operator をインストールできます。**oc** コマンドを使用して、**Subscription** オブジェクトを作成または更新します。

**前提条件**

- **cluster-admin** パーミッションを持つアカウントを使用して OpenShift Container Platform クラスターにアクセスできる。
- **oc** コマンドをローカルシステムにインストールする。

**手順**

1. OperatorHub からクラスターで利用できる Operator のリストを表示します。

```
$ oc get packagemanifests -n openshift-marketplace
```

**出力例**

```
NAME                              CATALOG               AGE
3scale-operator                   Red Hat Operators     91m
advanced-cluster-management       Red Hat Operators     91m
amq7-cert-manager                 Red Hat Operators     91m
...
couchbase-enterprise-certified    Certified Operators   91m
crunchy-postgres-operator         Certified Operators   91m
mongodb-enterprise                Certified Operators   91m
...
etcd                              Community Operators   91m
jaeger                            Community Operators   91m
kubefed                           Community Operators   91m
...
```

必要な Operator のカタログをメモします。

2. 必要な Operator を検査して、サポートされるインストールモードおよび利用可能なチャネルを確認します。

```
$ oc describe packagemanifests <operator_name> -n openshift-marketplace
```

3. **OperatorGroup** で定義される Operator グループは、Operator グループと同じ namespace 内のすべての Operator に必要な RBAC アクセスを生成するターゲット namespace を選択します。
Operator をサブスクライブする namespace には、Operator のインストールモードに一致する Operator グループが必要になります (**AllNamespaces** または **SingleNamespace** モードのいずれか)。インストールする Operator が **AllNamespaces** を使用する場合、 **openshift-operators** namespace には適切な Operator グループがすでに配置されます。

   ただし、Operator が **SingleNamespace** モードを使用し、適切な Operator グループがない場合、それらを作成する必要があります。

**注記**

この手順の Web コンソールバージョンでは、**SingleNamespace** モードを選択する際に、**OperatorGroup** および **Subscription** オブジェクトの作成を背後で自動的に処理します。

   a. **OperatorGroup** オブジェクト YAML ファイルを作成します (例: **operatorgroup.yaml**)。

      **OperatorGroup オブジェクトのサンプル**

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: <operatorgroup_name>
  namespace: <namespace>
spec:
  targetNamespaces:
  - <namespace>
```

   b. **OperatorGroup** オブジェクトを作成します。

```
$ oc apply -f operatorgroup.yaml
```

4. **Subscription** オブジェクトの YAML ファイルを作成し、namespace を Operator にサブスクライブします (例: **sub.yaml**)。

   **Subscription オブジェクトの例**

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: <subscription_name>
  namespace: openshift-operators (1)
spec:
  channel: <channel_name> (2)
  name: <operator_name> (3)
  source: redhat-operators (4)
```

```
  sourceNamespace: openshift-marketplace (5)
  config:
    env: (6)
    - name: ARGS
      value: "-v=10"
    envFrom: (7)
    - secretRef:
        name: license-secret
    volumes: (8)
    - name: <volume_name>
      configMap:
        name: <configmap_name>
    volumeMounts: (9)
    - mountPath: <directory_name>
      name: <volume_name>
    tolerations: (10)
    - operator: "Exists"
    resources: (11)
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
    nodeSelector: (12)
      foo: bar
```

(1) **AllNamespaces** インストールモードの使用については、**openshift-operators** namespace を指定します。それ以外の場合は、**SingleNamespace** インストールモードの使用について関連する単一の namespace を指定します。

(2) サブスクライブするチャネルの名前。

(3) サブスクライブする Operator の名前。

(4) Operator を提供するカタログソースの名前。

(5) カタログソースの namespace。デフォルトの OperatorHub カタログソースには **openshift-marketplace** を使用します。

(6) **env** パラメーターは、OLM によって作成される Pod のすべてのコンテナーに存在する必要がある環境変数の一覧を定義します。

(7) **envFrom** パラメーターは、コンテナーの環境変数に反映するためのソースの一覧を定義します。

(8) **volumes** パラメーターは、OLM によって作成される Pod に存在する必要があるボリュームの一覧を定義します。

(9) **volumeMounts** パラメーターは、OLM によって作成される Pod のすべてのコンテナーに存在する必要があるボリュームマウントの一覧を定義します。**volumeMount** が存在しない **ボリューム** を参照する場合、OLM は Operator のデプロイに失敗します。

(10) **tolerations** パラメーターは、OLM によって作成される Pod の容認の一覧を定義します。

11 **resources** パラメーターは、OLM によって作成される Pod のすべてのコンテナーのリソース制約を定義します。

12 **nodeSelector** パラメーターは、OLM によって作成される Pod の **ノードセレクター** を定義します。

5. **Subscription** オブジェクトを作成します。

```
$ oc apply -f sub.yaml
```

この時点で、OLM は選択した Operator を認識します。Operator のクラスターサービスバージョン (CSV) はターゲット namespace に表示され、Operator で指定される API は作成用に利用可能になります。

### 4.4.4. Web コンソールの使用によるクラスターからの Operator の削除

クラスター管理者は Web コンソールを使用して、選択した namespace からインストールされた Operator を削除できます。

**前提条件**

- **cluster-admin** パーミッションを持つアカウントを使用して OpenShift Container Platform クラスター Web コンソールにアクセスできること。

**手順**

1. **Operators → Installed Operators** ページに移動します。
2. スクロールするか、キーワードを **Filter by name** フィールドに入力して、削除する Operator を見つけます。次に、それをクリックします。
3. **Operator Details** ページの右側で、**Actions** 一覧から **Uninstall Operator** を選択します。**Uninstall Operator?** ダイアログボックスが表示されます。
4. **Uninstall** を選択し、Operator、Operator デプロイメント、および Pod を削除します。このアクションの後には、Operator は実行を停止し、更新を受信しなくなります。

**注記**

このアクションは、カスタムリソース定義 (CRD) およびカスタムリソース (CR) など、Operator が管理するリソースは削除されません。Web コンソールおよび継続して実行されるクラスター外のリソースによって有効にされるダッシュボードおよびナビゲーションアイテムには、手動でのクリーンアップが必要になる場合があります。Operator のアンインストール後にこれらを削除するには、Operator CRD を手動で削除する必要があります。

### 4.4.5. CLI の使用によるクラスターからの Operator の削除

クラスター管理者は CLI を使用して、選択した namespace からインストールされた Operator を削除できます。

**前提条件**

- **cluster-admin** パーミッションを持つアカウントを使用して OpenShift Container Platform クラスターにアクセスできる。
- **oc** コマンドがワークステーションにインストールされていること。

**手順**

1. サブスクライブされた Operator (例: **jaeger**) の現行バージョンを **currentCSV** フィールドで確認します。

```
$ oc get subscription jaeger -n openshift-operators -o yaml | grep currentCSV
```

**出力例**

```
  currentCSV: jaeger-operator.v1.8.2
```

2. サブスクリプション (例: **jaeger**) を削除します。

```
$ oc delete subscription jaeger -n openshift-operators
```

**出力例**

```
subscription.operators.coreos.com "jaeger" deleted
```

3. 直前の手順で **currentCSV** 値を使用し、ターゲット namespace の Operator の CSV を削除します。

```
$ oc delete clusterserviceversion jaeger-operator.v1.8.2 -n openshift-operators
```

**出力例**

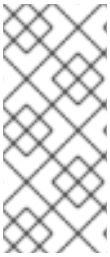
```
clusterserviceversion.operators.coreos.com "jaeger-operator.v1.8.2" deleted
```

# 第5章 RED HAT のロギングサブシステムを理解する

クラスター管理者は、ロギングシステムをデプロイし、ノードシステムの監査ログ、アプリケーションコンテナーログ、およびインフラストラクチャーログなどの OpenShift Container Platform クラスターからのすべてのログを集計できます。ロギングサブシステムは、クラスター全体からこれらのログを集約し、デフォルトのログストアに保存します。Kibana Web コンソールを使用して、ログデータを可視化 できます。

ロギングサブシステムは、次のタイプのログを集約します。

- **application**: クラスターで実行される、インフラストラクチャーコンテナーアプリケーションを除くユーザーアプリケーションによって生成されるコンテナーログ。
- **infrastructure**: ジャーナルログなどの、クラスターで実行されるインフラストラクチャーコンポーネントおよび OpenShift Container Platform ノードで生成されるログ。インフラストラクチャーコンポーネントは、**openshift***、 **kube***、または **default** プロジェクトで実行される Pod です。
- **audit**: ノード監査システム (auditd) で生成されるログ (**/var/log/audit/audit.log** ファイルに保存される)、および Kubernetes apiserver および OpenShift apiserver の監査ログ。



**注記**

内部 OpenShift Container Platform Elasticsearch ログストアは監査ログのセキュアなストレージを提供しないため、デフォルトで監査ログは内部 Elasticsearch インスタンスに保存されません。監査ログをデフォルトの内部 Elasticsearch ログストアに送信する必要がある場合 (Kibana で監査ログを表示するなど)、監査ログのログストアへの転送 で説明されているようにログ転送 API を使用する必要があります。

## 5.1. OPENSHIFT CONTAINER PLATFORM ロギングの共通用語集

この用語集は、OpenShift Container Platform Logging コンテンツで使用される一般的な用語を定義します。

**アノテーション**

アノテーションを使用して、メタデータをオブジェクトに添付できます。

**Cluster Logging Operator (CLO)**

Cluster Logging Operator は、アプリケーション、インフラストラクチャー、および監査ログの収集と転送を制御するための一連の API を提供します。

**カスタムリソース (CR)**

CR は Kubernetes API のエクステンションです。OpenShift Container Platform Logging およびログ転送を設定するために、**ClusterLogging** および **ClusterLogForwarder** カスタムリソースをカスタマイズできます。

**イベントルーター**

イベントルーターは、OpenShift Container Platform イベントを監視する Pod です。OpenShift Container Platform Logging を使用してログを収集します。

**Fluentd**

Fluentd は、各 OpenShift Container Platform ノードに常駐するログコレクターです。アプリケーション、インフラストラクチャー、および監査ログを収集し、それらをさまざまな出力に転送します。

**ガベージコレクション**

ガベージコレクションは、終了したコンテナーや実行中の Pod によって参照されていないイメージなどのクラスターリソースをクリーンアップするプロセスです。

**Elasticsearch**

Elasticsearch は、分散検索および分析エンジンです。OpenShift Container Platform は、OpenShift Container Platform Logging のデフォルトのログストアとして ELasticsearch を使用します。

**Elasticsearch Operator**

Elasticsearch Operator は、OpenShift Container Platform 上で Elasticsearch クラスターを実行するために使用されます。Elasticsearch Operator は、Elasticsearch クラスター操作のセルフサービスを提供し、OpenShift Container Platform Logging により使用されます。

**インデックス化**

インデックス作成は、データをすばやく見つけてアクセスするために使用されるデータ構造手法です。インデックスを作成すると、クエリーの処理時に必要なディスクアクセスの量が最小限に抑えられるため、パフォーマンスが最適化されます。

**JSON ロギング**

OpenShift Container Platform Logging Log Forwarding API を使用すると、JSON ログを解析して構造化されたオブジェクトにし、それらを OpenShift Container Platform Logging が管理する Elasticsearch またはログ転送 API でサポートされるその他のサードパーティーシステムに転送できます。

**Kibana**

Kibana は、ヒストグラム、折れ線グラフ、円グラフを使用して Elasticsearch データを照会、検出、視覚化するためのブラウザーベースのコンソールインターフェイスです。

**Kubernetes API サーバー**

Kubernetes API サーバーは、API オブジェクトのデータを検証して設定します。

**Labels**

ラベルは、Pod などのオブジェクトのサブセットを整理および選択するために使用できるキーと値のペアです。

**ロギング**

OpenShift Container Platform Logging を使用すると、クラスター全体でアプリケーション、インフラストラクチャー、および監査ログを集約できます。また、ログをデフォルトのログストアに保存したり、サードパーティーのシステムに転送したり、デフォルトのログストアに保存されているログを照会して視覚化したりすることもできます。

**ロギングコレクター**

ロギングコレクターは、クラスターからログを収集してフォーマットし、ログストアまたはサードパーティーシステムに転送します。

**ログストア**

ログストアは、集約されたログを格納するために使用されます。デフォルトの Elasticsearch ログストアを使用、またはログを外部ログストアに転送できます。デフォルトのログストアは、短期の保存について最適化され、テストされています。

**ログビジュアライザー**

ログビジュアライザーは、ログ、グラフ、チャート、その他のメトリックなどの情報を表示するために使用できるユーザーインターフェイス (UI) コンポーネントです。現在の実装は Kibana です。

**node**

ノードは、OpenShift Container Platform クラスター内のワーカーマシンです。ノードは、仮想マシン (VM) または物理マシンのいずれかです。

**Operator**

Operator は、OpenShift Container Platform クラスターで Kubernetes アプリケーションをパッケージ化、デプロイ、および管理するための推奨される方法。Operator は、人間による操作に関する知識を取り入れて、簡単にパッケージ化してお客様と共有できるソフトウェアにエンコードします。

**pod**

Pod は、Kubernetes における最小の論理単位です。Pod は 1 つ以上のコンテナーで設定され、ワーカーノードで実行されます。

**ロールベースアクセス制御 (RBAC)**

RBAC は、クラスターユーザーとワークロードが、ロールを実行するために必要なリソースにのみアクセスできるようにするための重要なセキュリティーコントロールです。

**shards**

Elasticsearch は、Fluentd からのログデータをデータストアまたはインデックスに編成し、各インデックスをシャードと呼ばれる複数の部分に分割します。

**Taint**

テイントは、Pod が適切なノードに確実にスケジュールされるようにします。ノードに 1 つ以上のテイントを適用できます。

**容認**

Pod に容認を適用できます。Tolerations を使用すると、スケジューラーは、テイントが一致する Pod をスケジュールできます。

**Web コンソール**

OpenShift Container Platform を管理するためのユーザーインターフェイス (UI)。

## 5.2. RED HAT OPENSHIFT の LOGGING サブシステムのデプロイについて

OpenShift Container Platform クラスター管理者は、OpenShift Container Platform Web コンソールまたは CLI コマンドを使用してロギングシステムをデプロイし、OpenShift Elasticsearch Operator および Red Hat OpenShift Logging Operator をインストールできます。Operator がインストールされたら、**ClusterLogging** カスタムリソース (CR) を作成して、ロギングサブシステム pod およびロギングサブシステムをサポートするために必要なその他のリソースをスケジュールします。Operator は、ロギングサブシステムのデプロイ、アップグレード、および保守を担当します。

**ClusterLogging** CR は、ログを収集し、保存し、視覚化するために必要なロギングスタックのすべてのコンポーネントを含む完全なロギングシステム環境を定義します。Red Hat OpenShift Logging Operator はロギングシステム CR を監視し、ロギングデプロイメントを適宜調整します。

管理者およびアプリケーション開発者は、表示アクセスのあるプロジェクトのログを表示できます。

詳細は、Configuring the log collector を参照してください。

### 5.2.1. JSON OpenShift コンテナープラットフォームロギング

JSON ロギングを使用して、JSON 文字列を構造化オブジェクトに解析するようにログ転送 API を設定できます。以下のタスクを実行します。

- JSON ログの解析
- Elasticsearch の JSON ログデータの設定
- JSON ログの Elasticsearch ログストアへの転送

### 5.2.2. Kubernetes イベントの収集および保存

OpenShift Container Platform イベントルーターは、Kubernetes イベントを監視し、それらを OpenShift Container Platform Logging によって収集できるようにログに記録する Pod です。イベントルーターは手動でデプロイする必要があります。

詳細は、Kubernetes イベントの収集および保存 を参照してください。

### 5.2.3. OpenShift Container Platform ロギングの更新

OpenShift Container Platform を使用すると、OpenShift Container Platform のロギングを更新できます。OpenShift Container Platform Logging の更新時には、以下の Operator を更新する必要があります。

- Elasticsearch Operator
- Cluster Logging Operator

詳細は、About updating OpenShift Container Platform Logging を参照してください。

### 5.2.4. クラスターダッシュボードの表示

OpenShift Container Platform Logging ダッシュボードには、クラスターレベルで Elasticsearch インスタンスに関する詳細を示すチャートが含まれています。これらのチャートは、問題の診断と予測に役立ちます。

詳細は、クラスターダッシュボードの表示 を参照してください。

### 5.2.5. OpenShift Container Platform ロギングのトラブルシューティング

次のタスクを実行してログの問題をトラブルシューティングできます。

- ロギングステータスの表示
- ログストアのステータスの表示
- ロギングアラートの理解
- Red Hat サポート用のロギングデータの収集
- Critical Alerts のトラブルシューティング

### 5.2.6. OpenShift Container Platform ロギングのアンインストール

ClusterLogging カスタムリソース (CR) を削除して、ログ集計を停止できます。CR の削除後に残る他のクラスターロギングコンポーネントがあり、これらはオプションで削除できます。

詳細は、About uninstalling OpenShift Container Platform Logging を参照してください。

### 5.2.7. フィールドのエクスポート

ロギングシステムはフィールドをエクスポートします。エクスポートされたフィールドはログレコードに存在し、Elasticsearch および Kibana から検索できます。

詳細は、フィールドのエクスポート を参照してください。

### 5.2.8. サブシステムコンポーネントのロギングについて

ロギングシステムコンポーネントには、すべてのノードおよびコンテナーログを収集し、それらをログストアに書き込む OpenShift Container Platform クラスターの各ノードにデプロイされるコレクターが含まれます。一元化された Web UI を使用し、集計されたデータを使用して高度な可視化 (visualization) およびダッシュボードを作成できます。

ロギングサブシステムの主なコンポーネントは次のとおりです。

- collection: これは、クラスターからログを収集し、それらをフォーマットし、ログストアに転送するコンポーネントです。現在の実装は Fluentd です。
- log store: これはログが保存される場所です。デフォルトの実装は Elasticsearch です。デフォルトの Elasticsearch ログストアを使用、またはログを外部ログストアに転送できます。デフォルトのログストアは、短期の保存について最適化され、テストされています。
- visualization: これは、ログ、グラフ、グラフなどを表示するために使用される UI コンポーネントです。現在の実装は Kibana です。

本書では、特筆されない限り、log store と Elasticsearch、visualization と Kibana、collection と Fluentd を区別せずに使用する場合があります。

### 5.2.9. ロギングコレクターについて

Red Hat OpenShift のロギングサブシステムは、コンテナーとノードのログを収集します。

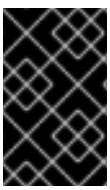
デフォルトでは、ログコレクターは以下のソースを使用します。

- すべてのシステムログ用の journald
- すべてのコンテナーログ用の **/var/log/containers/*.log**

監査ログを収集するようにログコレクターを設定すると、**/var/log/audit/audit.log** から取得されます。

ロギングコレクターは、Pod を各 OpenShift Container Platform ノードにデプロイするデーモンセットです。システムおよびインフラストラクチャーのログは、オペレーティングシステム、コンテナーランタイム、および OpenShift Container Platform からの journald ログメッセージによって生成されます。アプリケーションログは CRI-O コンテナーエンジンによって生成されます。Fluentd はこれらのソースからログを収集し、OpenShift Container Platform で設定したように内部または外部に転送します。

コンテナーランタイムは、プロジェクト、Pod 名、およびコンテナー ID などのログメッセージのソースを特定するための最小限の情報を提供します。この情報だけでは、ログのソースを一意に特定することはできません。ログコレクターがログを処理する前に、指定された名前およびプロジェクトを持つ Pod が削除される場合は、ラベルやアノテーションなどの API サーバーからの情報は利用できない可能性があります。そのため、似たような名前の Pod やプロジェクトからログメッセージを区別したり、ログのソースを追跡できない場合があります。この制限により、ログの収集および正規化は **ベストエフォート** ベースであると見なされます。



**重要**

利用可能なコンテナーランタイムは、ログメッセージのソースを特定するための最小限の情報を提供し、個別のログメッセージが一意となる確証はなく、これらのメッセージにより、そのソースを追跡できる訳ではありません。

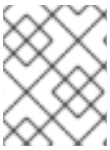
詳細は、Configuring the log collector を参照してください。

### 5.2.10. ログストアについて

デフォルトで、OpenShift Container Platform は Elasticsearch (ES) を使用してログデータを保存します。オプションで、Log Forwarder API を使用して、ログを外部ストアに転送できます。fluentd、rsyslog、kafka など、いくつかのタイプのストアがサポートされています。

ロギングサブシステム Elasticsearch インスタンスは、約 7 日間の短期ストレージ用に最適化およびテストされています。長期間ログを保持する必要がある場合は、データをサードパーティーのストレージシステムに移動することが推奨されます。

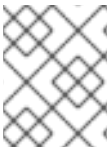
Elasticsearch は Fluentd からのログデータをデータストアまたは **インデックス** に編成し、それぞれのインデックスを **シャード** と呼ばれる複数の部分に分割します。これは、Elasticsearch クラスターの Elasticsearch ノードセット全体に分散されます。Elasticsearch を、**レプリカ** と呼ばれるシャードのコピーを作成するように設定できます。Elasticsearch はこれを Elasticsearch ノード全体に分散します。**ClusterLogging** カスタムリソース (CR) により、データの冗長性および耐障害性を確保するためにシャードを複製する方法を指定できます。また、**ClusterLogging** CR の保持ポリシーを使用して各種のログが保持される期間を指定することもできます。



**注記**

インデックステンプレートのプライマリーシャードの数は Elasticsearch データノードの数と等しくなります。

Red Hat OpenShift Logging Operator および OpenShift Elasticsearch Operator は、各 Elasticsearch ノードが独自のストレージボリュームを含む一意のデプロイメントを使用してデプロイされるようにします。**ClusterLogging** カスタムリソース (CR) を使用して Elasticsearch ノードの数を適宜増やすことができます。ストレージの設定に関する考慮事項は、Elasticsearch ドキュメント を参照してください。



**注記**

可用性の高い Elasticsearch 環境には 3 つ以上の Elasticsearch ノードが必要で、それぞれが別のホストに置かれる必要があります。

Elasticsearch インデックスに適用されているロールベースアクセス制御 (RBAC) は、開発者のログの制御アクセスを可能にします。管理者はすべてのログに、開発者は各自のプロジェクトのログにのみアクセスできます。

詳細は、ログストアの設定 を参照してください。

### 5.2.11. ロギングの可視化について

OpenShift Container Platform は Kibana を使用して、Fluentd によって収集され、Elasticsearch によってインデックス化されるログデータを表示します。

Kibana は、ヒストグラム、線グラフ、円グラフその他の可視化機能を使用して Elasticsearch データをクエリーし、検出し、可視化するためのブラウザーベースのコンソールインターフェイスです。

詳細は、ログビジュアライザーの設定 を参照してください。

### 5.2.12. イベントのルーティングについて

イベントルーターは、OpenShift Container Platform イベントを監視する pod であるため、Red Hat のロギングサブシステムによってイベントを収集できます。イベントルーターはすべてのプロジェクトからイベントを収集し、それらを **STDOUT** に書き込みます。Fluentd はそれらのイベントを収集し、そ

れらを OpenShift Container Platform Elasticsearch インスタンスに転送します。Elasticsearch はイベントを **infra** インデックスにインデックス化します。

イベントルーターは手動でデプロイする必要があります。

詳細は、Kubernetes イベントの収集および保存 を参照してください。

### 5.2.13. ログ転送

デフォルトでは、Red Hat OpenShift のロギングサブシステムは、**ClusterLogging** カスタムリソース (CR) で定義されているデフォルトの内部 Elasticsearch ログストアにログを送信します。ログを他のログアグリゲーターに転送する必要がある場合は、ログ転送機能を使用してログをクラスター内外の特定のエンドポイントに送信できます。

詳細は、ログのサードパーティーシステムへの転送 を参照してください。

## 5.3. VECTOR について

Vector は、ロギングサブシステムの Fluentd の代替として提供されるログコレクターです。

次の出力がサポートされています。

- **elasticsearch**。外部 Elasticsearch インスタンス。**elasticsearch** 出力では、TLS 接続を使用できます。
- **kafka**。Kafka ブローカー。**kafka** 出力は、セキュリティーで保護されていない接続または TLS 接続を使用できます。
- **loki**。水平的にスケーラビリティーが高く、マルチテナントログ集約システムです i。

### 5.3.1. Vector の有効化

Vector はデフォルトでは有効になっていません。以下のステップを使用して、OpenShift Container Platform クラスターで Vector を有効にします。

**重要**

Vector は、FIPS 対応クラスターをサポートしていません。

**前提条件**

- OpenShift Container Platform: 4.11
- Red Hat OpenShift のロギングサブシステム: 5.4
- FIPS が無効

**手順**

1. **openshift-logging** プロジェクトで **ClusterLogging** カスタムリソース (CR) を編集します。

```
$ oc -n openshift-logging edit ClusterLogging instance
```

2. **logging.openshift.io/preview-vector-collector: enabled** アノテーションを **ClusterLogging** カスタムリソース (CR) に追加します。

3. **ClusterLogging** カスタムリソース (CR) にコレクションタイプとして **vector** を追加します。

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
  namespace: "openshift-logging"
  annotations:
    logging.openshift.io/preview-vector-collector: enabled
spec:
  collection:
  logs:
    type: "vector"
    vector: {}
```

**関連情報**

- Vector ドキュメント

### 5.3.2. コレクター機能

**表5.1 ログソース**

| 機能 | Fluentd | Vector |
|---|---|---|
| アプリコンテナーのログ | ✓ | ✓ |
| アプリ固有のルーティング | ✓ | ✓ |
| namespace 別のアプリ固有のルーティング | ✓ | ✓ |
| インフラコンテナーログ | ✓ | ✓ |
| インフラジャーナルログ | ✓ | ✓ |
| Kube API 監査ログ | ✓ | ✓ |
| OpenShift API 監査ログ | ✓ | ✓ |
| Open Virtual Network (OVN) 監査ログ | ✓ | ✓ |

**表5.2 出力**

| 機能 | Fluentd | Vector |
|---|---|---|
| Elasticsearch v5-v7 | ✓ | ✓ |
| Fluent 転送 | ✓ | |
| Syslog RFC3164 | ✓ | |
| Syslog RFC5424 | ✓ | |
| Kafka | ✓ | ✓ |
| Cloudwatch | ✓ | ✓ |
| Loki | ✓ | ✓ |

**表5.3 認証および認可**

| 機能 | Fluentd | Vector |
|---|---|---|
| Elasticsearch 証明書 | ✓ | ✓ |
| Elasticsearch ユーザー名/パスワード | ✓ | ✓ |
| Cloudwatch キー | ✓ | ✓ |
| クラウドウォッチ STS | ✓ | |
| Kafka 証明書 | ✓ | ✓ |
| Kafka のユーザー名/パスワード | ✓ | ✓ |
| Kafka SASL | ✓ | ✓ |
| Loki ベアラートークン | ✓ | ✓ |

**表5.4 正規化と変換**

| 機能 | Fluentd | Vector |
|---|---|---|
| Viaq データモデル - アプリ | ✓ | ✓ |
| Viaq データモデル - インフラ | ✓ | ✓ |

| 機能 | Fluentd | Vector |
| --- | --- | --- |
| Viaq データモデル - インフラ (ジャーナル) | ✓ | ✓ |
| Viaq データモデル - Linux 監査 | ✓ | ✓ |
| Viaq データモデル - kube-apiserver 監査 | ✓ | ✓ |
| Viaq データモデル - OpenShift API 監査 | ✓ | ✓ |
| Viaq データモデル - OVN | ✓ | ✓ |
| ログレベルの正規化 | ✓ | ✓ |
| JSON 解析 | ✓ | ✓ |
| 構造化インデックス | ✓ | ✓ |
| 複数行エラー検出 | ✓ | |
| マルチコンテナー/分割インデックス | ✓ | ✓ |
| ラベルのフラット化 | ✓ | ✓ |
| CLF 静的ラベル | ✓ | ✓ |

**表5.5 チューニング**

| 機能 | Fluentd | Vector |
| --- | --- | --- |
| Fluentd readlinelimit | ✓ | |
| Fluentd バッファー | ✓ | |
| -chunklimitsize | ✓ | |
| - totallimitsize | ✓ | |
| - overflowaction | ✓ | |
| -flushThreadCount | ✓ | |

| 機能 | Fluentd | Vector |
| --- | --- | --- |
| - flushmode | ✓ | |
| - flushinterval | ✓ | |
| - retrywait | ✓ | |
| - retrytype | ✓ | |
| - retrymaxinterval | ✓ | |
| - retrytimeout | ✓ | |

**表5.6 制約**

| 機能 | Fluentd | Vector |
| --- | --- | --- |
| メトリクス | ✓ | ✓ |
| ダッシュボード | ✓ | ✓ |
| アラート | ✓ | |

**表5.7 その他**

| 機能 | Fluentd | Vector |
| --- | --- | --- |
| グローバルプロキシーサポート | ✓ | ✓ |
| x86 サポート | ✓ | ✓ |
| ARM サポート | ✓ | ✓ |
| PowerPC サポート | ✓ | ✓ |
| IBM Z サポート | ✓ | ✓ |
| IPv6 サポート | ✓ | ✓ |
| ログイベントのバッファリング | ✓ | |
| 非接続クラスター | ✓ | ✓ |

# 第6章 RED HAT のロギングサブシステムのインストール

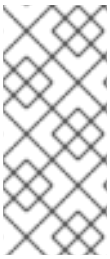
OpenShift Elasticsearch と Red Hat Logging Operators をデプロイすることにより、Red Hat のロギングサブシステムをインストールできます。OpenShift Elasticsearch Operator は、OpenShift Logging によって使用される Elasticsearch クラスターを作成し、管理します。ロギングサブシステム Operator は、ロギングスタックのコンポーネントを作成および管理します。

ロギングサブシステムを OpenShift Container Platform にデプロイするためのプロセスには以下が含まれます。

- Logging サブシステムのストレージに関する考慮事項 を確認します。
- OpenShift Container Platform Web コンソール、または CLI を使用した OpenShift Elasticsearch Operator および Red Hat OpenShift Logging Operator のインストール

## 6.1. WEB コンソールを使用した RED HAT のロギングサブシステムのインストール

OpenShift Container Platform Web コンソールを使用して OpenShift Elasticsearch および Red Hat OpenShift Logging Operator をインストールすることができます。

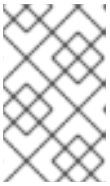


**注記**

デフォルトの Elasticsearch ログストアを使用しない場合、内部 Elasticsearch **logStore**、Kibana **visualization** コンポーネントを **ClusterLogging** カスタムリソース (CR) から削除することができます。これらのコンポーネントの削除はオプションですが、これによりリソースを節約できます。詳細は、Removing unused components if you do not use the default Elasticsearch log store を参照してください。

**前提条件**

- Elasticsearch の必要な永続ストレージがあることを確認します。各 Elasticsearch ノードには独自のストレージボリュームが必要であることに注意してください。



**注記**

永続ストレージにローカルボリュームを使用する場合は、**LocalVolume** オブジェクトの **volumeMode: block** で記述される raw ブロックボリュームを使用しないでください。Elasticsearch は raw ブロックボリュームを使用できません。

Elasticsearch はメモリー集約型アプリケーションです。デフォルトで、OpenShift Container Platform はメモリー要求および 16 GB の制限を持つ 3 つの Elasticsearch ノードをインストールします。OpenShift Container Platform ノードの最初の 3 つのセットには、Elasticsearch をクラスター内で実行するのに十分なメモリーがない可能性があります。Elasticsearch に関連するメモリーの問題が発生した場合、既存ノードのメモリーを増やすのではなく、Elasticsearch ノードをクラスターにさらに追加します。

**手順**

OpenShift Container Platform Web コンソールを使用して OpenShift Elasticsearch Operator および Red Hat OpenShift Logging Operator をインストールするには、以下を実行します。

1. OpenShift Elasticsearch Operator をインストールします。
   a. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリック

します。

b. 利用可能な Operator のリストから **OpenShift Elasticsearch Operator** を選択し、**Install** をクリックします。

c. **All namespaces on the cluster**が **Installation Mode** で選択されていることを確認します。

d. **openshift-operators-redhat** が **Installed Namespace** で選択されていることを確認します。
**openshift-operators-redhat** namespace を指定する必要があります。**openshift-operators** namespace には信頼されていないコミュニティー Operator が含まれる可能性があり、OpenShift Container Platform メトリックと同じ名前でメトリックを公開する可能性があるため、これによって競合が生じる可能性があります。

e. **Enable operator recommended cluster monitoring on this namespace**を選択します。
このオプションは、namespace オブジェクトに **openshift.io/cluster-monitoring: "true"** ラベルを設定します。クラスターモニタリングが **openshift-operators-redhat** namespace を収集できるように、このオプションを選択する必要があります。

f. **Update Channel** として **stable-5.x** を選択します。

g. **Approval Strategy** を選択します。

- **Automatic** ストラテジーにより、Operator Lifecycle Manager (OLM) は新規バージョンが利用可能になると Operator を自動的に更新できます。
- **Manual** ストラテジーには、Operator の更新を承認するための適切な認証情報を持つユーザーが必要です。

h. **Install** をクリックします。

i. **Operators → Installed Operators** ページに切り替えて、OpenShift Elasticsearch Operator がインストールされていることを確認します。

j. **Status** が **Succeeded** の状態で、**OpenShift Elasticsearch Operator** が すべてのプロジェクトにリスト表示されていることを確認します。

2. Red Hat OpenShift Logging Operator をインストールします。

a. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。

b. 利用可能な Operator のリストから **Red Hat OpenShift Logging** を選択し、**Install** をクリックします。

c. **A specific namespace on the cluster**が **Installation Mode** で選択されていることを確認します。

d. **Operator recommended namespace** が **Installed Namespace** で **openshift-logging** になっていることを確認します。

e. **Enable operator recommended cluster monitoring on this namespace**を選択します。
このオプションは、namespace オブジェクトに **openshift.io/cluster-monitoring: "true"** ラベルを設定します。クラスターモニタリングが **openshift-logging** namespace を収集できるように、このオプションを選択する必要があります。

f. **Update Channel** として **stable-5.x** を選択します。

g. **Approval Strategy** を選択します。

- **Automatic** ストラテジーにより、Operator Lifecycle Manager (OLM) は新規バージョンが利用可能になると Operator を自動的に更新できます。
- **Manual** ストラテジーには、Operator の更新を承認するための適切な認証情報を持つユーザーが必要です。

h. **Install** をクリックします。

i. **Operators → Installed Operators** ページに切り替えて、Red Hat OpenShift Logging Operator がインストールされていることを確認します。

j. **Red Hat OpenShift Logging** が **Status** が **Succeeded** の状態で **openshift-logging** プロジェクトにリスト表示されていることを確認します。
Operator がインストール済みとして表示されない場合に、さらにトラブルシューティングを実行します。

- **Operators → Installed Operators** ページに切り替え、**Status** 列でエラーまたは失敗の有無を確認します。
- **Workloads → Pods** ページに切り替え、 **openshift-logging** プロジェクトの Pod で問題を報告しているログの有無を確認します。

3. OpenShift Logging インスタンスを作成します。

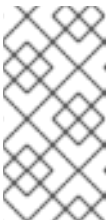
a. **Administration → Custom Resource Definitions**ページに切り替えます。

b. **Custom Resource Definitions**ページで、**ClusterLogging** をクリックします。

c. **Custom Resource Definition details**ページで、**Actions** メニューから **View Instances** を選択します。

d. **ClusterLoggings** ページで、 **Create ClusterLogging** をクリックします。
データを読み込むためにページの更新が必要になる場合があります。

e. YAML フィールドで、コードを以下に置き換えます。



**注記**

このデフォルトの OpenShift Logging 設定は各種の環境をサポートすることが予想されます。OpenShift Logging クラスターに加えることのできる変更の詳細は、ロギングシステムコンポーネントのチューニングおよび設定に関するトピックを確認してください。

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance" 1
  namespace: "openshift-logging"
spec:
  managementState: "Managed"  2
  logStore:
    type: "elasticsearch"  3
    retentionPolicy: 4
```

```
    application:
      maxAge: 1d
    infra:
      maxAge: 7d
    audit:
      maxAge: 7d
  elasticsearch:
    nodeCount: 3 5
    storage:
      storageClassName: "<storage_class_name>" 6
      size: 200G
    resources: 7
        limits:
          memory: "16Gi"
        requests:
          memory: "16Gi"
    proxy: 8
      resources:
        limits:
          memory: 256Mi
        requests:
          memory: 256Mi
    redundancyPolicy: "SingleRedundancy"
visualization:
  type: "kibana" 9
  kibana:
    replicas: 1
collection:
  logs:
    type: "fluentd" 10
    fluentd: {}
```

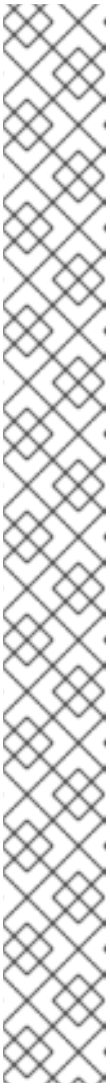
1. 名前は **instance** である必要があります。
2. OpenShift Logging の管理状態。OpenShift Logging のデフォルト値を変更する場合は、これを **Unmanaged** (管理外) に設定することが求められる場合があります。ただし、管理外のデプロイメントは OpenShift Logging がマネージドの状態に戻されるまで更新を受信しません。
3. Elasticsearch の設定に必要な設定。CR を使用してシャードのレプリケーションポリシーおよび永続ストレージを設定できます。
4. Elasticsearch が各ログソースを保持する期間を指定します。整数および時間の指定 (weeks(w)、hour(h/H)、minutes(m)、および seconds(s)) を入力します。たとえば、7 日の場合は **7d** となります。**maxAge** よりも古いログは削除されます。各ログソースの保持ポリシーを指定する必要があります。指定しないと、Elasticsearch インデックスはそのソースに対して作成されません。
5. Elasticsearch ノードの数を指定します。このリストに続く注記を確認してください。
6. Elasticsearch ストレージの既存のストレージクラスの名前を入力します。最適なパフォーマンスを得るには、ブロックストレージを割り当てるストレージクラスを指定します。ストレージクラスを指定しないと、OpenShift Logging は一時ストレージを使用します。

7 必要に応じて CPU およびメモリー要求を指定します。これらの値を空のままにすると、OpenShift Elasticsearch Operator はデフォルト値を設定します。これらのデフォ

8 必要に応じて Elasticsearch プロキシーの CPU およびメモリーの制限および要求を指定します。これらの値を空のままにすると、OpenShift Elasticsearch Operator はデフォルト値を設定します。これらのデフォルト値はほとんどのデプロイメントでは問題なく使用できるはずです。デフォルト値は、メモリー要求の場合は **256Mi**、CPU 要求の場合は **100m** です。

9 Kibana の設定に必要な設定。CR を使用して、冗長性を確保するために Kibana をスケーリングし、Kibana ノードの CPU およびメモリーを設定できます。詳細は、**ログビジュアライザーの設定** を参照してください。

10 Fluentd の設定に必要な設定。CR を使用して Fluentd の CPU およびメモリー制限を設定できます。詳細は**Fluentd の設定**を参照してください。



**注記**

Elasticsearch コントロールプレーンノードの最大数は 3 です。**3** を超える **nodeCount** を指定する場合、OpenShift Container Platform は、マスター、クライアントおよびデータロールを使用して、3 つのマスターとしての適格性のあるノードである Elasticsearch ノードを作成します。追加の Elasticsearch ノードは、クライアントおよびデータロールを使用してデータのみのノードとして作成されます。コントロールプレーンノードは、インデックスの作成および削除、シャードの割り当て、およびノードの追跡などのクラスター全体でのアクションを実行します。データノードはシャードを保持し、CRUD、検索、および集計などのデータ関連の操作を実行します。データ関連の操作は、I/O、メモリーおよび CPU 集約型の操作です。これらのリソースを監視し、現行ノードがオーバーロードする場合にデータノード追加することが重要です。

たとえば、**nodeCount=4** の場合に、以下のノードが作成されます。

```
$ oc get deployment
```

**出力例**

```
cluster-logging-operator       1/1     1          1         18h
elasticsearch-cd-x6kdekli-1    0/1     1          0         6m54s
elasticsearch-cdm-x6kdekli-1   1/1     1          1         18h
elasticsearch-cdm-x6kdekli-2   0/1     1          0         6m49s
elasticsearch-cdm-x6kdekli-3   0/1     1          0         6m44s
```

インデックステンプレートのプライマリーシャードの数は Elasticsearch データノードの数と等しくなります。

f. **Create** をクリックします。これにより、ロギングサブシステムコンポーネント、**Elasticsearch** カスタムリソースとコンポーネント、および Kibana インターフェイスが作成されます。

4. インストールを確認します。

a. **Workloads → Pods** ページに切り替えます。

b. **openshift-logging** プロジェクトを選択します。
以下のリストのような OpenShift Logging、Elasticsearch、Fluentd、および Kibana のいくつかの Pod が表示されるはずです。

- cluster-logging-operator-cb795f8dc-xkckc
- elasticsearch-cdm-b3nqzchd-1-5c6797-67kfz
- elasticsearch-cdm-b3nqzchd-2-6657f4-wtprv
- elasticsearch-cdm-b3nqzchd-3-588c65-clg7g
- fluentd-2c7dg
- fluentd-9z7kk
- fluentd-br7r2
- fluentd-fn2sb
- fluentd-pb2f8
- fluentd-zqgqx
- kibana-7fb4fd4cc9-bvt4p

**関連情報**

- OperatorHub からの Operator のインストール

## 6.2. インストール後のタスク

Kibana を使用する場合、Kibana のデータを確認し、び可視化するために、Kibana インデックスパターンおよびビジュアライゼーションを手動で作成する 必要があります。

クラスターネットワークプロバイダーがネットワークの分離を実施している場合、ロギングシステム Operator が含まれるプロジェクト間のネットワークトラフィックを許可します。

## 6.3. CLI を使用した RED HAT OPENSHIFT のロギングサブシステムのインストール

OpenShift Container Platform CLI を使用して OpenShift Elasticsearch および Red Hat OpenShift Logging Operator をインストールすることができます。

**前提条件**

- Elasticsearch の必要な永続ストレージがあることを確認します。各 Elasticsearch ノードには独自のストレージボリュームが必要であることに注意してください。

**注記**

永続ストレージにローカルボリュームを使用する場合は、**LocalVolume** オブジェクトの **volumeMode: block** で記述される raw ブロックボリュームを使用しないでください。Elasticsearch は raw ブロックボリュームを使用できません。

Elasticsearch はメモリー集約型アプリケーションです。デフォルトで、OpenShift Container Platform はメモリー要求および 16 GB の制限を持つ 3 つの Elasticsearch ノードをインストールします。OpenShift Container Platform ノードの最初の 3 つのセットには、Elasticsearch をクラスター内で実行するのに十分なメモリーがない可能性があります。Elasticsearch に関連するメモリーの問題が発生した場合、既存ノードのメモリーを増やすのではなく、Elasticsearch ノードをクラスターにさらに追加します。

**手順**

CLI を使用して OpenShift Elasticsearch Operator および Red Hat OpenShift Logging Operator をインストールするには、以下を実行します。

1. OpenShift Elasticsearch Operator の namespace を作成します。

   a. OpenShift Elasticsearch Operator の namespace オブジェクト YAML ファイル (**eo-namespace.yaml** など) を作成します。

```
apiVersion: v1
kind: Namespace
metadata:
  name: openshift-operators-redhat (1)
  annotations:
    openshift.io/node-selector: ""
  labels:
    openshift.io/cluster-monitoring: "true" (2)
```

   (1) **openshift-operators-redhat** namespace を指定する必要があります。メトリクスとの競合が発生する可能性を防ぐには、Prometheus のクラスターモニタリングスタックを、**openshift-operators** namespace からではなく、**openshift-operators-redhat** namespace からメトリクスを収集するように設定する必要があります。**openshift-operators** namespace には信頼されていないコミュニティー Operator が含まれる可能性があり、OpenShift Container Platform メトリックと同じ名前でメトリックを公開する可能性があるため、これによって競合が生じる可能性があります。

   (2) 文字列。クラスターモニタリングが **openshift-operators-redhat** namespace を収集できるように、このラベルを上記のように指定する必要があります。

   b. namespace を作成します。

```
$ oc create -f <file-name>.yaml
```

   以下に例を示します。

```
$ oc create -f eo-namespace.yaml
```

2. Red Hat OpenShift Logging Operator の namespace を作成します。

   a. Red Hat OpenShift Logging Operator の namespace オブジェクト YAML ファイル (**olo-namespace.yaml** など) を作成します。

```
apiVersion: v1
kind: Namespace
metadata:
  name: openshift-logging
  annotations:
```

```
    openshift.io/node-selector: ""
  labels:
    openshift.io/cluster-monitoring: "true"
```

b. namespace を作成します。

```
$ oc create -f <file-name>.yaml
```

以下に例を示します。

```
$ oc create -f olo-namespace.yaml
```

3. 以下のオブジェクトを作成して OpenShift Elasticsearch Operator をインストールします。

a. OpenShift Elasticsearch Operator の Operator グループオブジェクトの YAML ファイル (**eo-og.yaml** など) を作成します。

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: openshift-operators-redhat
  namespace: openshift-operators-redhat (1)
spec: {}
```

(1) **openshift-operators-redhat** namespace を指定する必要があります。

b. Operator グループオブジェクトを作成します。

```
$ oc create -f <file-name>.yaml
```

以下に例を示します。

```
$ oc create -f eo-og.yaml
```

c. Subscription オブジェクト YAML ファイル (**eo-sub.yaml** など) を作成し、namespace を OpenShift Elasticsearch Operator にサブスクライブします。

**Subscription の例**

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: "elasticsearch-operator"
  namespace: "openshift-operators-redhat" (1)
spec:
  channel: "stable-5.1" (2)
  installPlanApproval: "Automatic" (3)
  source: "redhat-operators" (4)
  sourceNamespace: "openshift-marketplace"
  name: "elasticsearch-operator"
```

(1) **openshift-operators-redhat** namespace を指定する必要があります。

2 チャネルとして **stable** または **stable-5.<x>** を指定します。以下の注意点を参照してください。

3 **Automatic** により、Operator Lifecycle Manager (OLM) は新規バージョンが利用可能になると Operator を自動的に更新できます。**Manual** には、Operator の更新を承認するための適切な認証情報を持つユーザーが必要です。

4 **redhat-operators** を指定します。OpenShift Container Platform クラスターが、非接続クラスターとも呼ばれるネットワークが制限された環境でインストールされている場合、Operator Lifecycle Manager (OLM) の設定時に作成される CatalogSource オブジェクトの名前を指定します。

**注記**

**stable** を指定すると、最新の安定したリリースの現行バージョンがインストールされます。**installPlanApproval: "Automatic"** で **stable** 使用すると、Operatar が自動的に最新の安定したメジャーおよびマイナーリリースにアップグレードします。

**stable-5.<x>** を指定すると、特定のメジャーリリースの現在のマイナーバージョンがインストールされます。**installPlanApproval: "Automatic"** で **stable-5.<x>** を使用すると、**x** で指定したメジャーリリース内で最新の安定マイナーリリースに Operator が自動的にアップグレードされます。

d. Subscription オブジェクトを作成します。

```
$ oc create -f <file-name>.yaml
```

以下に例を示します。

```
$ oc create -f eo-sub.yaml
```

OpenShift Elasticsearch Operator は **openshift-operators-redhat** namespace にインストールされ、クラスター内の各プロジェクトにコピーされます。

e. Operator のインストールを確認します。

```
$ oc get csv --all-namespaces
```

**出力例**

```
NAMESPACE                                 NAME                                         DISPLAY
VERSION            REPLACES   PHASE
default                                   elasticsearch-operator.5.1.0-202007012112.p0
OpenShift Elasticsearch Operator   5.1.0-202007012112.p0            Succeeded
kube-node-lease                           elasticsearch-operator.5.1.0-202007012112.p0
OpenShift Elasticsearch Operator   5.1.0-202007012112.p0            Succeeded
kube-public                               elasticsearch-operator.5.1.0-202007012112.p0
OpenShift Elasticsearch Operator   5.1.0-202007012112.p0            Succeeded
kube-system                               elasticsearch-operator.5.1.0-202007012112.p0
OpenShift Elasticsearch Operator   5.1.0-202007012112.p0            Succeeded
openshift-apiserver-operator              elasticsearch-operator.5.1.0-
202007012112.p0    OpenShift Elasticsearch Operator   5.1.0-202007012112.p0
```

```
Succeeded
openshift-apiserver                                elasticsearch-operator.5.1.0-202007012112.p0
OpenShift Elasticsearch Operator   5.1.0-202007012112.p0              Succeeded
openshift-authentication-operator                  elasticsearch-operator.5.1.0-
202007012112.p0    OpenShift Elasticsearch Operator   5.1.0-202007012112.p0
Succeeded
openshift-authentication                           elasticsearch-operator.5.1.0-
202007012112.p0    OpenShift Elasticsearch Operator   5.1.0-202007012112.p0
Succeeded
...
```

それぞれの namespace には OpenShift Elasticsearch Operator がなければなりません。バージョン番号が表示されるものと異なる場合があります。

4. 以下のオブジェクトを作成して Red Hat OpenShift Logging Operator をインストールします。

   a. Red Hat OpenShift Logging Operator の Operator グループオブジェクトの YAML ファイル (**olo-og.yaml** など) を作成します。

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: cluster-logging
  namespace: openshift-logging (1)
spec:
  targetNamespaces:
  - openshift-logging (2)
```

   (1) (2) **openshift-logging** namespace を指定する必要があります。

   b. OperatorGroup オブジェクトを作成します。

```
$ oc create -f <file-name>.yaml
```

   以下に例を示します。

```
$ oc create -f olo-og.yaml
```

   c. Subscription オブジェクト YAML ファイル (**olo-sub.yaml** など) を作成し、namespace を Red Hat OpenShift Logging Operator にサブスクライブします。

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: cluster-logging
  namespace: openshift-logging (1)
spec:
  channel: "stable" (2)
  name: cluster-logging
  source: redhat-operators (3)
  sourceNamespace: openshift-marketplace
```

   (1) **openshift-logging** namespace を指定する必要があります。

2 チャネルとして **stable** または **stable-5.<x>** を指定します。

3 **redhat-operators** を指定します。OpenShift Container Platform クラスターが、非接続クラスターとも呼ばれる制限されたネットワークにインストールされている場合、Operator Lifecycle Manager (OLM) の設定時に作成した CatalogSource オブジェクトの名前を指定します。

```
$ oc create -f <file-name>.yaml
```

以下に例を示します。

```
$ oc create -f olo-sub.yaml
```

Red Hat OpenShift Logging Operator は **openshift-logging** namespace にインストールされます。

d. Operator のインストールを確認します。
**openshift-logging** namespace には Red Hat OpenShift Logging Operator がなければなりません。バージョン番号が表示されるものと異なる場合があります。

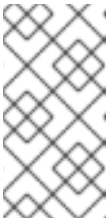
```
$ oc get csv -n openshift-logging
```

**出力例**

```
NAMESPACE                                       NAME                                         DISPLAY
VERSION            REPLACES   PHASE
...
openshift-logging                               clusterlogging.5.1.0-202007012112.p0
OpenShift Logging         5.1.0-202007012112.p0              Succeeded
...
```

5. OpenShift Logging インスタンスを作成します。

a. Red Hat OpenShift Logging Operator のインスタンスオブジェクト YAML ファイル (**olo-instance.yaml** など) を作成します。



**注記**

このデフォルトの OpenShift Logging 設定は各種の環境をサポートすることが予想されます。OpenShift Logging クラスターに加えることのできる変更の詳細は、ロギングシステムコンポーネントのチューニングおよび設定に関するトピックを確認してください。

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance" 1
  namespace: "openshift-logging"
spec:
  managementState: "Managed"  2
  logStore:
    type: "elasticsearch"  3
```

```
  retentionPolicy: 4
    application:
      maxAge: 1d
    infra:
      maxAge: 7d
    audit:
      maxAge: 7d
  elasticsearch:
    nodeCount: 3 5
    storage:
      storageClassName: "<storage-class-name>" 6
      size: 200G
    resources: 7
      limits:
        memory: "16Gi"
      requests:
        memory: "16Gi"
    proxy: 8
      resources:
        limits:
          memory: 256Mi
        requests:
          memory: 256Mi
    redundancyPolicy: "SingleRedundancy"
visualization:
  type: "kibana" 9
  kibana:
    replicas: 1
collection:
  logs:
    type: "fluentd" 10
    fluentd: {}
```

1. 名前は **instance** である必要があります。
2. OpenShift Logging の管理状態。OpenShift Logging のデフォルト値を変更する場合は、これを **Unmanaged** (管理外) に設定することが求められる場合があります。ただし、管理外のデプロイメントは OpenShift Logging がマネージドの状態に戻されるまで更新を受信しません。デプロイメントをマネージドの状態に戻すと、加えた変更が元に戻される可能性があります。
3. Elasticsearch の設定に必要な設定。カスタムリソース (CR) を使用してシャードのレプリケーションポリシーおよび永続ストレージを設定できます。
4. Elasticsearch が各ログソースを保持する期間を指定します。整数および時間の指定 (weeks(w)、hour(h/H)、minutes(m)、および seconds(s)) を入力します。たとえば、7 日の場合は **7d** となります。**maxAge** よりも古いログは削除されます。各ログソースの保持ポリシーを指定する必要があります。指定しないと、Elasticsearch インデックスはそのソースに対して作成されません。
5. Elasticsearch ノードの数を指定します。このリストに続く注記を確認してください。
6. Elasticsearch ストレージの既存のストレージクラスの名前を入力します。最適なパフォーマンスを得るには、ブロックストレージを割り当てるストレージクラスを指定します。ストレージクラスを指定しない場合、OpenShift Container Platform は一時

ストレージのみの OpenShift Logging をデプロイします。

7 必要に応じて CPU およびメモリー要求を指定します。これらの値を空のままにすると、OpenShift Elasticsearch Operator はデフォルト値を設定します。これらのデフォルト値はほとんどのデプロイメントでは問題なく使用できるます。デフォルト値は、メモリー要求の場合は **16Gi** であり、CPU 要求の場合は **1** です。

8 必要に応じて Elasticsearch プロキシーの CPU およびメモリーの制限および要求を指定します。これらの値を空のままにすると、OpenShift Elasticsearch Operator はデフォルト値を設定します。これらのデフォルト値はほとんどのデプロイメントでは問題なく使用できるはずです。デフォルト値は、メモリー要求の場合は **256Mi**、CPU 要求の場合は **100m** です。

9 Kibana の設定に必要な設定。CR を使用して、冗長性を確保するために Kibana をスケーリングし、Kibana Pod の CPU およびメモリーを設定できます。詳細は、**ログビジュアライザーの設定** を参照してください。

10 Fluentd の設定に必要な設定。CR を使用して Fluentd の CPU およびメモリー制限を設定できます。詳細は**Fluentd の設定**を参照してください。

**注記**

Elasticsearch コントロールプレーンノードの最大数は 3 です。**3** を超える **nodeCount** を指定する場合、OpenShift Container Platform は、マスター、クライアントおよびデータロールを使用して、3 つのマスターとしての適格性のあるノードである Elasticsearch ノードを作成します。追加の Elasticsearch ノードは、クライアントおよびデータロールを使用してデータのみのノードとして作成されます。コントロールプレーンノードは、インデックスの作成および削除、シャードの割り当て、およびノードの追跡などのクラスター全体でのアクションを実行します。データノードはシャードを保持し、CRUD、検索、および集計などのデータ関連の操作を実行します。データ関連の操作は、I/O、メモリーおよび CPU 集約型の操作です。これらのリソースを監視し、現行ノードがオーバーロードする場合にデータノード追加することが重要です。

たとえば、**nodeCount=4** の場合に、以下のノードが作成されます。

```
$ oc get deployment
```

**出力例**

```
cluster-logging-operator       1/1     1          1          18h
elasticsearch-cd-x6kdekli-1    1/1     1          0          6m54s
elasticsearch-cdm-x6kdekli-1   1/1     1          1          18h
elasticsearch-cdm-x6kdekli-2   1/1     1          0          6m49s
elasticsearch-cdm-x6kdekli-3   1/1     1          0          6m44s
```

インデックステンプレートのプライマリーシャードの数は Elasticsearch データノードの数と等しくなります。

b. インスタンスを作成します。

```
$ oc create -f <file-name>.yaml
```

以下に例を示します。

```
$ oc create -f olo-instance.yaml
```

これにより、ロギングサブシステムコンポーネント、**Elasticsearch** カスタムリソースとコンポーネント、および Kibana インターフェイスが作成されます。

6. **openshift-logging** プロジェクトに Pod を一覧表示して、インストールを検証します。
次のリストのように、Logging サブシステムのコンポーネントの Pod がいくつか表示されます。

```
$ oc get pods -n openshift-logging
```

**出力例**

```
NAME                                        READY   STATUS    RESTARTS   AGE
cluster-logging-operator-66f77ffccb-ppzbg   1/1     Running   0          7m
elasticsearch-cdm-ftuhduuw-1-ffc4b9566-q6bhp   2/2     Running   0          2m40s
elasticsearch-cdm-ftuhduuw-2-7b4994dbfc-rd2gc   2/2     Running   0          2m36s
elasticsearch-cdm-ftuhduuw-3-84b5ff7ff8-gqnm2   2/2     Running   0          2m4s
collector-587vb                             1/1     Running   0          2m26s
collector-7mpb9                             1/1     Running   0          2m30s
collector-flm6j                             1/1     Running   0          2m33s
collector-gn4rn                             1/1     Running   0          2m26s
collector-nlgb6                             1/1     Running   0          2m30s
collector-snpkt                             1/1     Running   0          2m28s
kibana-d6d5668c5-rppqm                      2/2     Running   0          2m39s
```

## 6.4. インストール後のタスク

Kibana を使用する場合、Kibana のデータを確認し、び可視化するために、Kibana インデックスパターンおよびビジュアライゼーションを手動で作成する 必要があります。

クラスターネットワークプロバイダーがネットワークの分離を実施している場合、ロギングシステム Operator が含まれるプロジェクト間のネットワークトラフィックを許可します。

### 6.4.1. Kibana インデックスパターンの定義

インデックスパターンは、可視化する必要のある Elasticsearch インデックスを定義します。Kibana でデータを確認し、可視化するには、インデックスパターンを作成する必要があります。

**前提条件**

- Kibana で **infra** および **audit** インデックスを表示するには、ユーザーには **cluster-admin** ロール、 **cluster-reader** ロール、または両方のロールが必要です。デフォルトの **kubeadmin** ユーザーには、これらのインデックスを表示するための適切なパーミッションがあります。
**default**、**kube-** および **openshift-** プロジェクトで Pod およびログを表示できる場合に、これらのインデックスにアクセスできるはずです。以下のコマンドを使用して、現在のユーザーが適切なパーミッションを持っているかどうかを確認できます。

```
$ oc auth can-i get pods/log -n <project>
```

**出力例**

```
yes
```

**注記**

監査ログは、デフォルトでは内部 OpenShift Container Platform Elasticsearch インスタンスに保存されません。Kibana で監査ログを表示するには、ログ転送 API を使用して監査ログの **default** 出力を使用するパイプラインを設定する必要があります。

- Elasticsearch ドキュメントは、インデックスパターンを作成する前にインデックス化する必要があります。これは自動的に実行されますが、新規または更新されたクラスターでは数分の時間がかかる可能性があります。

**手順**

Kibana でインデックスパターンを定義し、ビジュアライゼーションを作成するには、以下を実行します。

1. OpenShift Container Platform コンソールで、Application Launcher をクリックし、**Logging** を選択します。
2. **Management → Index Patterns → Create index pattern** をクリックして Kibana インデックスパターンを作成します。
   - 各ユーザーは、プロジェクトのログを確認するために、Kibana に初めてログインする際にインデックスパターンを手動で作成する必要があります。ユーザーは **app** という名前のインデックスパターンを作成し、**@timestamp** 時間フィールドを使用してコンテナーログを表示する必要があります。
   - 管理ユーザーはそれぞれ、最初に Kibana にログインする際に、**@timestamp** 時間フィールドを使用して **app**、 **infra** および **audit** インデックスのインデックスパターンを作成する必要があります。
3. 新規インデックスパターンから Kibana のビジュアライゼーション (Visualization) を作成します。

### 6.4.2. ネットワークの分離が有効にされている場合のプロジェクト間のトラフィックの許可

クラスターネットワークプロバイダーはネットワークの分離を有効にする可能性があります。その場合、OpenShift Logging によってデプロイされる Operator が含まれるプロジェクト間のネットワークトラフィックを許可する必要があります。

ネットワークの分離は、異なるプロジェクトにある Pod およびサービス間のネットワークトラフィックをブロックします。ロギングシステムは、**OpenShift Elasticsearch Operator** を **openshift-operators-redhat** プロジェクトにインストールし、**Red Hat OpenShift Logging Operator** を **openshift-logging** プロジェクトにインストールします。したがって、これら 2 つのプロジェクト間のトラフィックを許可する必要があります。

OpenShift Container Platform は、2 つのサポート対象のオプションをデフォルトの Container Network Interface (CNI) ネットワークプロバイダー、OpenShift SDN および OVN-Kubernetes 用に提供します。これら 2 つのプロバイダーはさまざまなネットワーク分離ポリシーを実装します。

OpenShift SDN には 3 つのモードがあります。

**network policy (ネットワークポリシー)**

これはデフォルトモードになります。ポリシーが定義されていない場合は、すべてのトラフィックを許可します。ただし、ユーザーがポリシーを定義する場合、通常はすべてのトラフィックを拒否し、例外を追加して開始します。このプロセスでは、異なるプロジェクトで実行されているアプリケーションが破損する可能性があります。そのため、ポリシーを明示的に設定し、1 つのロギング関連のプロジェクトから他のプロジェクトへの egress のトラフィックを許可します。

**multitenant (マルチテナント)**

このモードは、ネットワークの分離を実行します。2 つのロギング関連のプロジェクトを結合して、それらのプロジェクト間のトラフィックを許可します。

**subnet (サブネット)**

このモードでは、すべてのトラフィックを許可します。ネットワーク分離は実行しません。アクションは不要です。

OVN-Kubernetes は常に **ネットワークポリシー** を使用します。そのため、OpenShift SDN の場合と同様に、ポリシーを明示的に設定し、1 つのロギング関連のプロジェクトから他のプロジェクトへの egress のトラフィックを許可する必要があります。

**手順**

- **multitenant** モードで OpenShift SDN を使用している場合は、2 つのプロジェクトに参加します。以下に例を示します。

```
$ oc adm pod-network join-projects --to=openshift-operators-redhat openshift-logging
```

- または、**network policy** の OpenShift SDN および OVN-Kubernetes の場合は、以下の操作を実行します。

  a. **openshift-operators-redhat** namespace にラベルを設定します。以下に例を示します。

```
$ oc label namespace openshift-operators-redhat project=openshift-operators-redhat
```

  b. **openshift-operators-redhat**、**openshift-monitoring**、および**openshift-ingress**プロジェクトから openshift-logging プロジェクトへの入力を許可する、**openshift-logging** namespace にネットワークポリシーオブジェクトを作成します。以下に例を示します。

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-from-openshift-monitoring-ingress-operators-redhat
spec:
  ingress:
  - from:
    - podSelector: {}
  - from:
    - namespaceSelector:
        matchLabels:
          project: "openshift-operators-redhat"
  - from:
    - namespaceSelector:
        matchLabels:
          name: "openshift-monitoring"
  - from:
    - namespaceSelector:
```

```
      matchLabels:
        network.openshift.io/policy-group: ingress
  podSelector: {}
  policyTypes:
  - Ingress
```

**関連情報**

- ネットワークポリシーについて
- OpenShift SDN デフォルト CNI ネットワークプロバイダーについて
- OVN-Kubernetes デフォルト Container Network Interface (CNI) ネットワークプロバイダーについて

# 第7章 ロギングデプロイメントの設定

## 7.1. クラスターロギングカスタムリソースについて

Red Hat OpenShift のロギングサブシステムを設定するには、**ClusterLogging** カスタムリソース (CR) をカスタマイズします。

### 7.1.1. ClusterLogging カスタムリソースについて

ロギングサブシステム環境に変更を加えるには、**ClusterLogging** カスタムリソース (CR) を作成および変更します。

CR の作成または変更方法については、このドキュメントで適宜説明されます。

次の例は、ロギングサブシステムの一般的なカスタムリソースを示しています。

**ClusterLogging カスタムリソース (CRD) のサンプル**

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance" 1
  namespace: "openshift-logging" 2
spec:
  managementState: "Managed" 3
  logStore:
    type: "elasticsearch" 4
    retentionPolicy:
      application:
        maxAge: 1d
      infra:
        maxAge: 7d
      audit:
        maxAge: 7d
    elasticsearch:
      nodeCount: 3
      resources:
        limits:
          memory: 16Gi
        requests:
          cpu: 500m
          memory: 16Gi
      storage:
        storageClassName: "gp2"
        size: "200G"
      redundancyPolicy: "SingleRedundancy"
  visualization: 5
    type: "kibana"
    kibana:
      resources:
        limits:
          memory: 736Mi
        requests:
          cpu: 100m
```

```
        memory: 736Mi
    replicas: 1
  collection: 6
    logs:
      type: "fluentd"
      fluentd:
        resources:
          limits:
            memory: 736Mi
          requests:
            cpu: 100m
            memory: 736Mi
```

1. CR の名前は **instance** である必要があります。
2. CR は **openshift-logging** namespace にインストールされる必要があります。
3. Red Hat OpenShift Logging Operator の管理状態。**Unmanaged** に設定すると、Operator はサポート対象外となり、更新を取得しません。
4. 保持ポリシー、ノード数、リソース要求および制限およびストレージクラスなどのログストアの設定。
5. リソース要求および制限、Pod レプリカ数などのビジュアライザーの設定。
6. リソース要求および制限を含むログコレクターの設定。

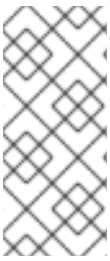
## 7.2. ロギングコレクターの設定

Red Hat OpenShift のロギングサブシステムは、クラスターからオペレーションとアプリケーションログを収集し、Kubernetes Pod とプロジェクトメタデータでデータを強化します。

ログコレクターの CPU およびメモリー制限を設定し、ログコレクター Pod を特定のノードに移動 できます。ログコレクターに対するサポートされるすべての変更は、**ClusterLogging** カスタムリソース (CR) の **spec.collection.log.fluentd** スタンザを使用して実行できます。

### 7.2.1. サポートされる設定

Red Hat OpenShift のロギングサブシステムを設定するためにサポートされている方法は、このドキュメントで説明されているオプションを使用して設定することです。サポートされていない他の設定は使用しないでください。設定のパラダイムが OpenShift Container Platform リリース間で変更される可能性があり、このような変更は、設定のすべての可能性が制御されている場合のみ適切に対応できます。本書で説明されている設定以外の設定を使用する場合、OpenShift Elasticsearch Operator および Red Hat OpenShift Logging Operator が差分を調整するため、変更内容は失われます。Operator はデフォルトで定義された状態にすべて戻します。



**注記**

OpenShift Container Platform ドキュメントで説明されていない設定を実行する **必要がある** 場合、Red Hat OpenShift Logging Operator または OpenShift Elasticsearch Operator を **Unmanaged** (管理外) に設定する **必要があります**。管理外の OpenShift Logging 環境は **サポート外** であり、OpenShift Logging を **Managed** に戻すまで変更を受信しません。

### 7.2.2. ロギングコレクター Pod の表示

Fluentd ロギングコレクター Pod およびそれらが実行されている対応するノードを表示できます。Fluentd ロギングコレクター Pod は **openshift-logging** プロジェクトでのみ実行されます。

**手順**

- **openshift-logging** プロジェクトで以下のコマンドを実行し、Fluentd ロギングコレクター Pod とそれらの詳細を表示します。

```
$ oc get pods --selector component=collector -o wide -n openshift-logging
```

**出力例**

```
NAME          READY  STATUS    RESTARTS   AGE     IP           NODE              NOMINATED
NODE   READINESS GATES
fluentd-8d69v  1/1    Running   0          134m    10.130.2.30   master1.example.com   <none>
<none>
fluentd-bd225  1/1    Running   0          134m    10.131.1.11   master2.example.com   <none>
<none>
fluentd-cvrzs  1/1    Running   0          134m    10.130.0.21   master3.example.com   <none>
<none>
fluentd-gpqg2  1/1    Running   0          134m    10.128.2.27   worker1.example.com   <none>
<none>
fluentd-l9j7j  1/1    Running   0          134m    10.129.2.31   worker2.example.com   <none>
<none>
```

### 7.2.3. ログコレクター CPU およびメモリー制限の設定

ログコレクターは、CPU とメモリー制限の両方への調整を許可します。

**手順**

1. **openshift-logging** プロジェクトで **ClusterLogging** カスタムリソース (CR) を編集します。

```
$ oc -n openshift-logging edit ClusterLogging instance
```

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
  namespace: openshift-logging

...

spec:
  collection:
    logs:
      fluentd:
        resources:
          limits: 1
            memory: 736Mi
```

```
      requests:
        cpu: 100m
        memory: 736Mi
```

1. 必要に応じて CPU、メモリー制限および要求を指定します。表示される値はデフォルト値です。

### 7.2.4. ログフォワーダーの高度な設定

Red Hat OpenShift のロギングサブシステムには、Fluentd ログ転送のパフォーマンスを調整するために使用できる複数の Fluentd パラメーターが含まれています。これらのパラメーターを使用すると、以下の Fluentd の動作を変更できます。

- チャンクおよびチャンクのバッファーサイズ
- チャンクのフラッシュ動作
- チャンク転送の再試行動作

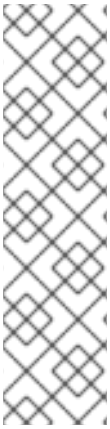
Fluentd は、**チャンク** という単一の Blob でログデータを収集します。Fluentd がチャンクを作成する際に、チャンクは **ステージ** にあると見なされます。ここでチャンクはデータで一杯になります。チャンクが一杯になると、Fluentd はチャンクを **キュー** に移動します。ここでチャンクはフラッシュされる前か、送信先に書き込まれるまで保持されます。Fluentd は、ネットワークの問題や送信先での容量の問題などのさまざまな理由でチャンクをフラッシュできない場合があります。チャンクをフラッシュできない場合、Fluentd は設定通りにフラッシュを再試行します。

OpenShift Container Platform のデフォルトで、Fluentd は **指数関数的バックオフ** 方法を使用してフラッシュを再試行します。この場合、Fluentd はフラッシュを再試行するまで待機する時間を 2 倍にします。これは、送信先への接続要求を減らすのに役立ちます。指数バックオフを無効にし、代わりに **定期的な** 再試行方法を使用できます。これは、指定の間隔でチャンクのフラッシュを再試行します。

これらのパラメーターは、待ち時間とスループット間のトレードオフを判断するのに役立ちます。

- Fluentd のスループットを最適化するには、これらのパラメーターを使用して、より大きなバッファーおよびキューを設定し、フラッシュを遅延し、再試行の間隔の長く設定することで、ネットワークパケット数を減らすことができます。より大きなバッファーにはノードのファイルシステムでより多くの領域が必要になることに注意してください。
- 待機時間が低い場合に最適化するには、パラメーターを使用してすぐにデータを送信し、バッチの蓄積を回避し、キューとバッファーが短くして、より頻繁にフラッシュおよび再試行を使用できます。

**ClusterLogging** カスタムリソース (CR) で以下のパラメーターを使用して、チャンクおよびフラッシュ動作を設定できます。次に、パラメーターは Fluentd で使用するために Fluentd 設定マップに自動的に追加されます。



### 注記

これらのパラメーターの特徴は以下の通りです。

- ほとんどのユーザーには関連性がありません。デフォルト設定で、全般的に良いパフォーマンスが得られるはずです。
- Fluentd 設定およびパフォーマンスに関する詳しい知識を持つ上級ユーザーのみが対象です。
- パフォーマンスチューニングのみを目的とします。ロギングの機能面に影響を与えることはありません。

**表7.1 高度な Fluentd 設定パラメーター**

| パラメーター | 説明 | デフォルト |
|---|---|---|
| **chunkLimitSize** | 各チャンクの最大サイズ。Fluentd はこのサイズに達するとデータのチャンクへの書き込みを停止します。次に、Fluentd はチャンクをキューに送信し、新規のチャンクを開きます。 | **8m** |
| **totalLimitSize** | ステージおよびキューの合計サイズであるバッファーの最大サイズ。バッファーサイズがこの値を超えると、Fluentd はデータのチャンクへの追加を停止し、エラーを出して失敗します。チャンクにないデータはすべて失われます。 | **8G** |
| **flushInterval** | チャンクのフラッシュの間隔。**s** (秒)、**m** (分)、 **h** (時間)、または **d** (日) を使用できます。 | **1s** |
| **flushMode** | フラッシュを実行する方法:<br>● **lazy**: **timekey** パラメーターに基づいてチャンクをフラッシュします。**timekey** パラメーターを変更することはできません。<br>● **interval**: **flushInterval** パラメーターに基づいてチャンクをフラッシュします。<br>● **immediate**: データをチャンクに追加後すぐにチャンクをフラッシュします。 | **interval** |

| パラメーター | 説明 | デフォルト |
| --- | --- | --- |
| **flushThreadCount** | チャンクのフラッシュを実行するスレッドの数。スレッドの数を増やすと、フラッシュのスループットが改善し、ネットワークの待機時間が非表示になります。 | **2** |
| **overflowAction** | キューが一杯になると、チャンク動作は以下のようになります。<br>● **throw_exception**: ログに表示される例外を発生させます。<br>● **block**: 詳細のバッファーの問題が解決されるまでデータのチャンクを停止します。<br>● **drop_oldest_chunk**: 新たな受信チャンクを受け入れるために最も古いチャンクをドロップします。古いチャンクの値は新しいチャンクよりも小さくなります。 | **block** |
| **retryMaxInterval** | **exponential_backoff** 再試行方法の最大時間 (秒単位)。 | **300s** |
| **retryType** | フラッシュに失敗する場合の再試行方法:<br>● **exponential_backoff**: フラッシュの再試行の間隔を増やします。Fluentd は、**retry_max_interval** パラメーターに達するまで、次の試行までに待機する時間を 2 倍にします。<br>● **periodic**: **retryWait** パラメーターに基づいてフラッシュを定期的に再試行します。 | **exponential_backoff** |
| **retryTimeOut** | レコードが破棄される前に再試行を試みる最大時間間隔。 | **60m** |
| **retryWait** | 次のチャンクのフラッシュまでの時間 (秒単位)。 | **1s** |

Fluentd チャンクのライフサイクルの詳細は、Fluentd ドキュメントの Buffer Plugins を参照してください。

**手順**

1. **openshift-logging** プロジェクトで **ClusterLogging** カスタムリソース (CR) を編集します。

```
$ oc edit ClusterLogging instance
```

2. 以下のパラメーターを追加または変更します。

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogging
metadata:
  name: instance
  namespace: openshift-logging
spec:
  forwarder:
    fluentd:
      buffer:
        chunkLimitSize: 8m 1
        flushInterval: 5s 2
        flushMode: interval 3
        flushThreadCount: 3 4
        overflowAction: throw_exception 5
        retryMaxInterval: "300s" 6
        retryType: periodic 7
        retryWait: 1s 8
        totalLimitSize: 32m 9
...
```

1. 各チャンクの最大サイズを指定してから、フラッシュ用にキューに入れます。
2. チャンクのフラッシュの間隔を指定します。
3. チャンクのフラッシュを実行する方法を指定します ( **lazy**、**interval**、または **immediate**)。
4. チャンクのフラッシュに使用するスレッドの数を指定します。
5. キューが一杯になる場合のチャンクの動作を指定します (**throw_exception**、**block**、または **drop_oldest_chunk**)。
6. **exponential_backoff** チャンクのフラッシュ方法について最大の間隔 (秒単位) を指定します。
7. チャンクのフラッシュが失敗する場合の再試行タイプ (**exponential_backoff** または **periodic**) を指定します。
8. 次のチャンクのフラッシュまでの時間 (秒単位) を指定します。
9. チャンクバッファーの最大サイズを指定します。

3. Flunentd Pod が再デプロイされていることを確認します。

```
$ oc get pods -l component=collector -n openshift-logging
```

4. 新規の値が **fluentd** 設定マップにあることを確認します。

```
$ oc extract configmap/fluentd --confirm
```

**fluentd.conf の例**

```
<buffer>
 @type file
 path '/var/lib/fluentd/default'
 flush_mode interval
 flush_interval 5s
 flush_thread_count 3
 retry_type periodic
 retry_wait 1s
 retry_max_interval 300s
 retry_timeout 60m
 queued_chunks_limit_size "#{ENV['BUFFER_QUEUE_LIMIT'] || '32'}"
 total_limit_size 32m
 chunk_limit_size 8m
 overflow_action throw_exception
</buffer>
```

### 7.2.5. デフォルトの Elasticsearch ログストアを使用しない場合の未使用のコンポーネントの削除

管理者がログをサードパーティーのログストアに転送し、デフォルトの Elasticsearch ログストアを使用しない場合は、ロギングクラスターからいくつかの未使用のコンポーネントを削除できます。

つまり、デフォルトの Elasticsearch ログストアを使用しない場合は、内部 Elasticsearch **logStore**、Kibana **visualization** コンポーネントを **ClusterLogging** カスタムリソース (CR) から削除できます。これらのコンポーネントの削除はオプションですが、これによりリソースを節約できます。

**前提条件**

- ログフォワーダーがログデータをデフォルトの内部 Elasticsearch クラスターに送信しないことを確認します。ログ転送の設定に使用した **ClusterLogForwarder** CR YAML ファイルを検査します。これには **default** を指定する **outputRefs** 要素が **ない** ことを確認します。以下に例を示します。

```
outputRefs:
- default
```

**警告**

**ClusterLogForwarder** CR がログデータを内部 Elasticsearch クラスターに転送し、**ClusterLogging** CR から **logStore** コンポーネントを削除するとします。この場合、内部 Elasticsearch クラスターはログデータを保存するために表示されません。これがないと、データが失われる可能性があります。

**手順**

1. **openshift-logging** プロジェクトで **ClusterLogging** カスタムリソース (CR) を編集します。

```
$ oc edit ClusterLogging instance
```

2. これらが存在する場合は、**logStore**、**visualization** スタンザを **ClusterLogging** CR から削除します。

3. **ClusterLogging** CR の **collection** スタンザを保持します。結果は以下の例のようになります。

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
  namespace: "openshift-logging"
spec:
  managementState: "Managed"
  collection:
    logs:
      type: "fluentd"
      fluentd: {}
```

4. コレクター Pod が再デプロイされたことを確認します。

```
$ oc get pods -l component=collector -n openshift-logging
```

**関連情報**

- ログのサードパーティーシステムへの転送

## 7.3. ログストアの設定

Red Hat OpenShift のロギングサブシステムは、Elasticsearch 6 (ES) を使用してログデータを保存および整理します。

ログストアに加えることのできる変更には、以下が含まれます。

- Elasticsearch クラスターのストレージ。
- シャードをクラスター内の複数のデータノードにレプリケートする方法 (完全なレプリケーションからレプリケーションなしまで)。

- Elasticsearch データへの外部アクセス

Elasticsearch はメモリー集約型アプリケーションです。それぞれの Elasticsearch ノードには、**ClusterLogging** カスタムリソースで指定しない限り、メモリー要求および制限の両方に 16G 以上のメモリーが必要です。初期設定の OpenShift Container Platform ノードのセットは、Elasticsearch クラスターをサポートするのに十分な大きさではない場合があります。その場合、推奨されるサイズ以上のメモリー (各 Elasticsearch ノードに最大 64G) を使用して実行できるようにノードを OpenShift Container Platform クラスターに追加する必要があります。

各 Elasticsearch ノードはこれより低い値のメモリー設定でも動作しますが、これは実稼働環境には推奨されません。

### 7.3.1. 監査ログのログストアへの転送

デフォルトで、OpenShift Logging では監査ログを内部の OpenShift Container Platform Elasticsearch ログストアに保存しません。Kibana で表示するなど、監査ログをこのログストアに送信できます。

監査ログをデフォルトの内部 Elasticsearch ログストアに送信するには、Kibana で監査ログを表示するなど、ログ転送 API を使用する必要があります。

**重要**

内部 OpenShift Container Platform Elasticsearch ログストアは、監査ログのセキュアなストレージを提供しません。監査ログを転送するシステムが組織および政府の規制に適合し、適切にセキュリティーが保護されていることを確認します。Red Hat OpenShift のロギングサブシステムは、これらの規制に準拠していません。

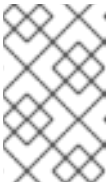
**手順**

ログ転送 API を使用して監査ログを内部 Elasticsearch インスタンスに転送するには、以下を実行します。

1. **ClusterLogForwarder** CR オブジェクトを定義する YAML ファイルを作成または編集します。
   - すべてのログタイプを内部 Elasticsearch インスタンスに送信するために CR を作成します。変更せずに以下の例を使用できます。

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogForwarder
metadata:
  name: instance
  namespace: openshift-logging
spec:
  pipelines: 1
  - name: all-to-default
    inputRefs:
    - infrastructure
    - application
    - audit
    outputRefs:
    - default
```

1 パイプラインは、指定された出力を使用して転送するログのタイプを定義します。デフォルトの出力は、ログを内部 Elasticsearch インスタンスに転送します。



**注記**

パイプラインの 3 つのすべてのタイプのログをパイプラインに指定する必要があります (アプリケーション、インフラストラクチャー、および監査)。ログの種類を指定しない場合、それらのログは保存されず、失われます。

- 既存の **ClusterLogForwarder** CR がある場合は、パイプラインを監査ログのデフォルト出力に追加します。デフォルトの出力を定義する必要はありません。以下に例を示します。

```
apiVersion: "logging.openshift.io/v1"
kind: ClusterLogForwarder
metadata:
  name: instance
  namespace: openshift-logging
spec:
  outputs:
   - name: elasticsearch-insecure
     type: "elasticsearch"
     url: http://elasticsearch-insecure.messaging.svc.cluster.local
     insecure: true
   - name: elasticsearch-secure
     type: "elasticsearch"
     url: https://elasticsearch-secure.messaging.svc.cluster.local
     secret:
       name: es-audit
   - name: secureforward-offcluster
     type: "fluentdForward"
     url: https://secureforward.offcluster.com:24224
     secret:
       name: secureforward
  pipelines:
   - name: container-logs
     inputRefs:
     - application
     outputRefs:
     - secureforward-offcluster
   - name: infra-logs
     inputRefs:
     - infrastructure
     outputRefs:
     - elasticsearch-insecure
   - name: audit-logs
     inputRefs:
     - audit
     outputRefs:
     - elasticsearch-secure
     - default (1)
```

(1) このパイプラインは、外部インスタンスに加えて監査ログを内部 Elasticsearch インスタンスに送信します。

**関連情報**

- ログ転送 API の詳細は、ログ転送 API を使用したログの転送 を参照してください。

### 7.3.2. ログ保持時間の設定

デフォルトの Elasticsearch ログストアがインフラストラクチャーログ、アプリケーションログ、監査ログなどの 3 つのログソースのインデックスを保持する期間を指定する **保持ポリシー** を設定できます。

保持ポリシーを設定するには、**ClusterLogging** カスタムリソース (CR) に各ログソースの **maxAge** パラメーターを設定します。CR はこれらの値を Elasticsearch ロールオーバースケジュールに適用し、Elasticsearch がロールオーバーインデックスを削除するタイミングを決定します。

Elasticsearch はインデックスをロールオーバーし、インデックスが以下の条件のいずれかに一致する場合に現在のインデックスを移動し、新規インデックスを作成します。

- インデックスは **Elasticsearch** CR の **rollover.maxAge** の値よりも古い値になります。
- インデックスサイズは、40 GB x プライマリーシャードの数よりも大きくなります。
- インデックスの doc 数は、40960 KB × プライマリーシャードの数よりも大きくなります。

Elasticsearch は、設定する保持ポリシーに基づいてロールオーバーインデックスを削除します。ログソースの保持ポリシーを作成しない場合、ログはデフォルトで 7 日後に削除されます。

**前提条件**

- Red HatOpenShift のロギングサブシステムと OpenShift Elasticsearch Operator がインストールされている必要があります。

**手順**

ログの保持時間を設定するには、以下を実行します。

1. **ClusterLogging** CR を編集して、**retentionPolicy** パラメーターを追加するか、変更します。

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
...
spec:
  managementState: "Managed"
  logStore:
    type: "elasticsearch"
    retentionPolicy: 1
      application:
        maxAge: 1d
      infra:
        maxAge: 7d
      audit:
        maxAge: 7d
    elasticsearch:
      nodeCount: 3
...
```

1 Elasticsearch が各ログソースを保持する時間を指定します。整数および時間の指定 (weeks(w)、hour(h/H)、minutes(m)、および seconds(s)) を入力します。たとえば、1 日の場合は **1d** になります。**maxAge** よりも古いログは削除されます。デフォルトで、ログは 7 日間保持されます。

2. **Elasticsearch** カスタムリソース (CR) で設定を確認できます。
たとえば、Red Hat OpenShift Logging Operator は以下の **Elasticsearch** CR を更新し、8 時間ごとにインフラストラクチャーログのアクティブなインデックスをロールオーバーし、ロールオーバーされたインデックスはロールオーバーの 7 日後に削除される設定を含む保持ポリシーを設定するとします。OpenShift Container Platform は 15 分ごとにチェックし、インデックスをロールオーバーする必要があるかどうかを判別します。

```
apiVersion: "logging.openshift.io/v1"
kind: "Elasticsearch"
metadata:
  name: "elasticsearch"
spec:
...
  indexManagement:
    policies: 1
      - name: infra-policy
        phases:
          delete:
            minAge: 7d 2
          hot:
            actions:
              rollover:
                maxAge: 8h 3
        pollInterval: 15m 4
...
```

1 各ログソースについて、保持ポリシーは、そのソースのログを削除/ロールオーバーするタイミングを示します。

2 OpenShift Container Platform がロールオーバーされたインデックスを削除する場合。この設定は、**ClusterLogging** CR に設定する **maxAge** になります。

3 インデックスをロールオーバーする際に考慮する OpenShift Container Platform のインデックス期間。この値は、**ClusterLogging** CR に設定する **maxAge** に基づいて決定されます。

4 OpenShift Container Platform がインデックスをロールオーバーする必要があるかどうかをチェックする場合。この設定はデフォルトであるため、変更できません。

**注記**

**Elasticsearch** CR の変更はサポートされていません。保持ポリシーに対するすべての変更は **ClusterLogging** CR で行う必要があります。

OpenShift Elasticsearch Operator は cron ジョブをデプロイし、**pollInterval** を使用してスケジュールされる定義されたポリシーを使用して各マッピングのインデックスをロールオーバーします。

```
$ oc get cronjob
```

**出力例**

```
NAME              SCHEDULE      SUSPEND   ACTIVE   LAST SCHEDULE   AGE
```

```
elasticsearch-im-app    */15 * * * *   False     0       <none>         4s
elasticsearch-im-audit  */15 * * * *   False     0       <none>         4s
elasticsearch-im-infra  */15 * * * *   False     0       <none>         4s
```

### 7.3.3. ログストアの CPU およびメモリー要求の設定

それぞれのコンポーネント仕様は、CPU とメモリー要求の両方への調整を許可します。OpenShift Elasticsearch Operator は環境に適した値を設定するため、これらの値を手動で調整する必要はありません。

**注記**

大規模なクラスターでは、Elasticsearch プロキシーコンテナーのデフォルトのメモリー制限が不十分な場合があり、これにより、プロキシーコンテナーが OOM による強制終了 (OOMKilled) が生じます。この問題が発生した場合は、Elasticsearch プロキシーのメモリー要求および制限を引き上げます。

各 Elasticsearch ノードはこれより低い値のメモリー設定でも動作しますが、これは実稼働環境でのデプロイメントには推奨 **されていません**。実稼働環境で使用する場合は、デフォルトの 16Gi よりも小さい値を各 Pod に割り当てることはできません。Pod ごとに割り当て可能な最大値は 64Gi であり、この範囲の中で、できるだけ多くのメモリーを割り当てることを推奨します。

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

**手順**

1. **openshift-logging** プロジェクトで **ClusterLogging** カスタムリソース (CR) を編集します。

```
$ oc edit ClusterLogging instance
```

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
....
spec:
    logStore:
      type: "elasticsearch"
      elasticsearch: 1
        resources:
          limits: 2
            memory: "32Gi"
          requests: 3
            cpu: "1"
            memory: "16Gi"
        proxy: 4
          resources:
            limits:
```

```
        memory: 100Mi
      requests:
        memory: 100Mi
```

1. 必要に応じて CPU およびメモリー要求を指定します。これらの値を空のままにすると、OpenShift Elasticsearch Operator はデフォルト値を設定します。これらのデフォルト値はほとんどのデプロイメントでは問題なく使用できるはずです。デフォルト値は、メモリー要求の場合は **16Gi** であり、CPU 要求の場合は **1** です。
2. Pod が使用できるリソースの最大量。
3. Pod のスケジュールに必要最小限のリソース。
4. 必要に応じて Elasticsearch プロキシーの CPU およびメモリーの制限および要求を指定します。これらの値を空のままにすると、OpenShift Elasticsearch Operator はデフォルト値を設定します。これらのデフォルト値はほとんどのデプロイメントでは問題なく使用できるます。デフォルト値は、メモリー要求の場合は **256Mi**、CPU 要求の場合は **100m** です。

Elasticsearch メモリーの量を調整するときは、**要求** と **制限** の両方に同じ値を使用する必要があります。

以下に例を示します。

```
resources:
  limits: 1
    memory: "32Gi"
  requests: 2
    cpu: "8"
    memory: "32Gi"
```

1. リソースの最大量。
2. 必要最小限の量。

Kubernetes は一般的にはノードの設定に従い、Elasticsearch が指定された制限を使用することを許可しません。**requests** と **limits** に同じ値を設定することにより、Elasticseach が必要なメモリーを確実に使用できるようにします (利用可能なメモリーがノードにあることを前提とします)。

### 7.3.4. ログストアのレプリケーションポリシーの設定

Elasticsearch シャードをクラスター内の複数のデータノードにレプリケートする方法を定義できます。

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

**手順**

1. **openshift-logging** プロジェクトで **ClusterLogging** カスタムリソース (CR) を編集します。

```
$ oc edit clusterlogging instance
```

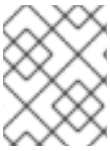
```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"

....

spec:
  logStore:
    type: "elasticsearch"
    elasticsearch:
      redundancyPolicy: "SingleRedundancy" 1
```

1 シャードの冗長性ポリシーを指定します。変更の保存時に変更が適用されます。

- **FullRedundancy**:Elasticsearch は、各インデックスのプライマリーシャードをすべてのデータノードに完全にレプリケートします。これは最高レベルの安全性を提供しますが、最大量のディスクが必要となり、パフォーマンスは最低レベルになります。
- **MultipleRedundancy**:Elasticsearch は、各インデックスのプライマリーシャードをデータノードの半分に完全にレプリケートします。これは、安全性とパフォーマンス間の適切なトレードオフを提供します。
- **SingleRedundancy**:Elasticsearch は、各インデックスのプライマリーシャードのコピーを 1 つ作成します。2 つ以上のデータノードが存在する限り、ログは常に利用可能かつ回復可能です。5 以上のノードを使用する場合は、MultipleRedundancy よりもパフォーマンスが良くなります。このポリシーは、単一 Elasticsearch ノードのデプロイメントには適用できません。
- **ZeroRedundancy**:Elasticsearch は、プライマリーシャードのコピーを作成しません。ノードが停止または失敗した場合は、ログは利用不可となるか、失われる可能性があります。安全性よりもパフォーマンスを重視する場合や、独自のディスク/PVC バックアップ/復元ストラテジーを実装している場合は、このモードを使用できます。



**注記**

インデックステンプレートのプライマリーシャードの数は Elasticsearch データノードの数と等しくなります。

### 7.3.5. Elasticsearch Pod のスケールダウン

クラスター内の Elasticsearch Pod 数を減らすと、データ損失や Elasticsearch のパフォーマンスが低下する可能性があります。

スケールダウンする場合は、一度に 1 つの Pod 分スケールダウンし、クラスターがシャードおよびレプリカのリバランスを実行できるようにする必要があります。Elasticsearch のヘルスステータスが **green** に戻された後に、別の Pod でスケールダウンできます。

**注記**

Elasticsearch クラスターが **ZeroRedundancy** に設定される場合は、Elasticsearch Pod をスケールダウンしないでください。

### 7.3.6. ログストアの永続ストレージの設定

Elasticsearch には永続ストレージが必要です。ストレージが高速になると、Elasticsearch のパフォーマンスも高速になります。

**警告**

NFS ストレージをボリュームまたは永続ボリュームを使用 (または Gluster などの NAS を使用する) ことは Elasticsearch ストレージではサポートされません。Lucene は NFS が指定しないファイルシステムの動作に依存するためです。データの破損およびその他の問題が発生する可能性があります。

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

**手順**

1. **ClusterLogging** CR を編集してクラスターの各データノードが永続ボリューム要求 (PVC) にバインドされるように指定します。

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
# ...
spec:
  logStore:
    type: "elasticsearch"
    elasticsearch:
      nodeCount: 3
      storage:
        storageClassName: "gp2"
        size: "200G"
```

この例では、クラスターの各データノードが、200G の AWS General Purpose SSD (gp2) ストレージを要求する永続ボリューム要求 (PVC) にバインドされるように指定します。

**注記**

永続ストレージにローカルボリュームを使用する場合は、**LocalVolume** オブジェクトの **volumeMode: block** で記述される raw ブロックボリュームを使用しないでください。Elasticsearch は raw ブロックボリュームを使用できません。

### 7.3.7. emptyDir ストレージのログストアの設定

ログストアで emptyDir を使用できます。これは、Pod のデータすべてが再起動時に失われる一時デプロイメントを作成します。

**注記**

emptyDir を使用する場合は、ログストアが再起動するか、再デプロイされる場合にデータが失われます。

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

**手順**

1. **ClusterLogging** CR を編集して emptyDir を指定します。

```
spec:
  logStore:
    type: "elasticsearch"
    elasticsearch:
      nodeCount: 3
      storage: {}
```

### 7.3.8. Elasticsearch クラスターのローリング再起動の実行

**elasticsearch** 設定マップまたは **elasticsearch-*** デプロイメント設定のいずれかを変更する際にローリング再起動を実行します。

さらにローリング再起動は、Elasticsearch Pod が実行されるノードで再起動が必要な場合に推奨されます。

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

**手順**

クラスターのローリング再起動を実行するには、以下を実行します。

1. **openshift-logging** プロジェクトに切り替えます。

```
$ oc project openshift-logging
```

2. Elasticsearch Pod の名前を取得します。

```
$ oc get pods -l component=elasticsearch-
```

3. コレクター Pod をスケールダウンして、Elasticsearch への新しいログの送信を停止します。

```
$ oc -n openshift-logging patch daemonset/collector -p '{"spec":{"template":{"spec":
{"nodeSelector":{"logging-infra-collector": "false"}}}}}'
```

4. OpenShift Container Platform es_util ツールを使用してシャードの同期フラッシュを実行して、シャットダウンの前にディスクへの書き込みを待機している保留中の操作がないようにします。

```
$ oc exec <any_es_pod_in_the_cluster> -c elasticsearch -- es_util --query="_flush/synced" -
XPOST
```

以下に例を示します。

```
$ oc exec -c elasticsearch-cdm-5ceex6ts-1-dcd6c4c7c-jpw6  -c elasticsearch -- es_util --
query="_flush/synced" -XPOST
```

**出力例**

```
{"_shards":{"total":4,"successful":4,"failed":0},".security":
{"total":2,"successful":2,"failed":0},".kibana_1":{"total":2,"successful":2,"failed":0}}
```

5. OpenShift Container Platform es_util ツールを使用して、ノードを意図的に停止する際のシャードのバランシングを防ぎます。

```
$ oc exec <any_es_pod_in_the_cluster> -c elasticsearch -- es_util --
query="_cluster/settings" -XPUT -d '{ "persistent": { "cluster.routing.allocation.enable" :
"primaries" } }'
```

以下に例を示します。

```
$ oc exec elasticsearch-cdm-5ceex6ts-1-dcd6c4c7c-jpw6 -c elasticsearch -- es_util --
query="_cluster/settings" -XPUT -d '{ "persistent": { "cluster.routing.allocation.enable" :
"primaries" } }'
```

**出力例**

```
{"acknowledged":true,"persistent":{"cluster":{"routing":{"allocation":
{"enable":"primaries"}}}},"transient":
```

6. コマンドが完了したら、ES クラスターのそれぞれのデプロイメントについて、以下を実行します。

   a. デフォルトで、OpenShift Container Platform Elasticsearch クラスターはノードのロールアウトをブロックします。以下のコマンドを使用してロールアウトを許可し、Pod が変更を取得できるようにします。

   ```
   $ oc rollout resume deployment/<deployment-name>
   ```

   以下に例を示します。

   ```
   $ oc rollout resume deployment/elasticsearch-cdm-0-1
   ```

   **出力例**

   ```
   deployment.extensions/elasticsearch-cdm-0-1 resumed
   ```

   新規 Pod がデプロイされます。Pod に準備状態のコンテナーがある場合は、次のデプロイメントに進むことができます。

```
$ oc get pods -l component=elasticsearch-
```

**出力例**

```
NAME                                            READY   STATUS    RESTARTS   AGE
elasticsearch-cdm-5ceex6ts-1-dcd6c4c7c-jpw6k    2/2     Running   0          22h
elasticsearch-cdm-5ceex6ts-2-f799564cb-l9mj7    2/2     Running   0          22h
elasticsearch-cdm-5ceex6ts-3-585968dc68-k7kjr   2/2     Running   0          22h
```

b. デプロイメントが完了したら、ロールアウトを許可しないように Pod をリセットします。

```
$ oc rollout pause deployment/<deployment-name>
```

以下に例を示します。

```
$ oc rollout pause deployment/elasticsearch-cdm-0-1
```

**出力例**

```
deployment.extensions/elasticsearch-cdm-0-1 paused
```

c. Elasticsearch クラスターが **green** または **yellow** 状態にあることを確認します。

```
$ oc exec <any_es_pod_in_the_cluster> -c elasticsearch -- es_util --
query=_cluster/health?pretty=true
```

**注記**

直前のコマンドで使用した Elasticsearch Pod でロールアウトを実行した場合、Pod は存在しなくなっているため、ここで新規 Pod 名が必要になります。

以下に例を示します。

```
$ oc exec elasticsearch-cdm-5ceex6ts-1-dcd6c4c7c-jpw6 -c elasticsearch -- es_util --
query=_cluster/health?pretty=true
```

```
{
  "cluster_name" : "elasticsearch",
  "status" : "yellow", (1)
  "timed_out" : false,
  "number_of_nodes" : 3,
  "number_of_data_nodes" : 3,
  "active_primary_shards" : 8,
  "active_shards" : 16,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 1,
  "delayed_unassigned_shards" : 0,
  "number_of_pending_tasks" : 0,
  "number_of_in_flight_fetch" : 0,
```

```
  "task_max_waiting_in_queue_millis" : 0,
  "active_shards_percent_as_number" : 100.0
}
```

1 次に進む前に、このパラメーターが **green** または **yellow** であることを確認します。

7. Elasticsearch 設定マップを変更した場合は、それぞれの Elasticsearch Pod に対してこれらの手順を繰り返します。

8. クラスターのすべてのデプロイメントがロールアウトされたら、シャードのバランシングを再度有効にします。

```
$ oc exec <any_es_pod_in_the_cluster> -c elasticsearch -- es_util --
query="_cluster/settings" -XPUT -d '{ "persistent": { "cluster.routing.allocation.enable" : "all" }
}'
```

以下に例を示します。

```
$ oc exec elasticsearch-cdm-5ceex6ts-1-dcd6c4c7c-jpw6 -c elasticsearch -- es_util --
query="_cluster/settings" -XPUT -d '{ "persistent": { "cluster.routing.allocation.enable" : "all" }
}'
```

**出力例**

```
{
  "acknowledged" : true,
  "persistent" : { },
  "transient" : {
    "cluster" : {
      "routing" : {
        "allocation" : {
          "enable" : "all"
        }
      }
    }
  }
}
```

9. 新しいログが Elasticsearch に送信されるように、コレクター Pod をスケールアップします。

```
$ oc -n openshift-logging patch daemonset/collector -p '{"spec":{"template":{"spec":
{"nodeSelector":{"logging-infra-collector": "true"}}}}}'
```

### 7.3.9. ログストアサービスのルートとしての公開

デフォルトでは、Red Hat OpenShift のロギングサブシステムとともにデプロイされているログストアには、ロギングクラスターの外部からアクセスできません。データにアクセスするツールについては、ログストアへの外部アクセスのために re-encryption termination でルートを有効にすることができます。

re-encrypt ルート、OpenShift Container Platform トークンおよびインストールされたログストア CA 証明書を作成して、ログストアに外部からアクセスすることができます。次に、以下を含む cURL 要求でログストアサービスをホストするノードにアクセスします。

- **Authorization: Bearer ${token}**
- Elasticsearch reencrypt ルートおよび Elasticsearch API 要求

内部からは、ログストアクラスター IP を使用してログストアサービスにアクセスできます。これは、以下のコマンドのいずれかを使用して取得できます。

```
$ oc get service elasticsearch -o jsonpath={.spec.clusterIP} -n openshift-logging
```

**出力例**

```
172.30.183.229
```

```
$ oc get service elasticsearch -n openshift-logging
```

**出力例**

```
NAME            TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)    AGE
elasticsearch   ClusterIP   172.30.183.229   <none>        9200/TCP   22h
```

以下のようなコマンドを使用して、クラスター IP アドレスを確認できます。

```
$ oc exec elasticsearch-cdm-oplnhinv-1-5746475887-fj2f8 -n openshift-logging -- curl -tlsv1.2 --
insecure -H "Authorization: Bearer ${token}" "https://172.30.183.229:9200/_cat/health"
```

**出力例**

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    29  100    29    0     0    108      0 --:--:-- --:--:-- --:--:--   108
```

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。
- ログにアクセスできるようになるには、プロジェクトへのアクセスが必要です。

**手順**

ログストアを外部に公開するには、以下を実行します。

1. **openshift-logging** プロジェクトに切り替えます。

   ```
   $ oc project openshift-logging
   ```

2. ログストアから CA 証明書を抽出し、**admin-ca** ファイルに書き込みます。

   ```
   $ oc extract secret/elasticsearch --to=. --keys=admin-ca
   ```

   **出力例**

```
admin-ca
```

3. ログストアサービスのルートを YAML ファイルとして作成します。

   a. 以下のように YAML ファイルを作成します。

```
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: elasticsearch
  namespace: openshift-logging
spec:
  host:
  to:
    kind: Service
    name: elasticsearch
  tls:
    termination: reencrypt
    destinationCACertificate: | 1
```

   1 次の手順でログストア CA 証明書を追加するか、コマンドを使用します。一部の re-encrypt ルートで必要とされる **spec.tls.key**、**spec.tls.certificate**、および **spec.tls.caCertificate** パラメーターを設定する必要はありません。

   b. 以下のコマンドを実行して、前のステップで作成したルート YAML にログストア CA 証明書を追加します。

```
$ cat ./admin-ca | sed -e "s/^/      /" >> <file-name>.yaml
```

   c. ルートを作成します。

```
$ oc create -f <file-name>.yaml
```

   **出力例**

```
route.route.openshift.io/elasticsearch created
```

4. Elasticsearch サービスが公開されていることを確認します。

   a. 要求に使用されるこのサービスアカウントのトークンを取得します。

```
$ token=$(oc whoami -t)
```

   b. 作成した **elasticsearch** ルートを環境変数として設定します。

```
$ routeES=`oc get route elasticsearch -o jsonpath={.spec.host}`
```

   c. ルートが正常に作成されていることを確認するには、公開されたルート経由で Elasticsearch にアクセスする以下のコマンドを実行します。

```
curl -tlsv1.2 --insecure -H "Authorization: Bearer ${token}" "https://${routeES}"
```

以下のような出力が表示されます。

**出力例**

```
{
  "name" : "elasticsearch-cdm-i40ktba0-1",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "0eY-tJzcR3KOdpgeMJo-MQ",
  "version" : {
  "number" : "6.8.1",
  "build_flavor" : "oss",
  "build_type" : "zip",
  "build_hash" : "Unknown",
  "build_date" : "Unknown",
  "build_snapshot" : true,
  "lucene_version" : "7.7.0",
  "minimum_wire_compatibility_version" : "5.6.0",
  "minimum_index_compatibility_version" : "5.0.0"
},
  "<tagline>" : "<for search>"
}
```

## 7.4. ログビジュアライザーの設定

OpenShift Container Platform は、Kibana を使用して、ロギングサブシステムによって収集されたログデータを表示します。

冗長性を確保するために Kibana をスケーリングし、Kibana ノードの CPU およびメモリーを設定することができます。

### 7.4.1. CPU およびメモリー制限の設定

ロギングサブシステムコンポーネントを使用すると、CPU とメモリーの両方の制限を調整できます。

**手順**

1. **openshift-logging** プロジェクトで **ClusterLogging** カスタムリソース (CR) を編集します。

```
$ oc -n openshift-logging edit ClusterLogging instance
```

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
  namespace: openshift-logging

...

spec:
  managementState: "Managed"
  logStore:
    type: "elasticsearch"
    elasticsearch:
      nodeCount: 3
```

```
    resources: (1)
      limits:
        memory: 16Gi
      requests:
        cpu: 200m
        memory: 16Gi
    storage:
      storageClassName: "gp2"
      size: "200G"
    redundancyPolicy: "SingleRedundancy"
  visualization:
    type: "kibana"
    kibana:
      resources: (2)
        limits:
          memory: 1Gi
        requests:
          cpu: 500m
          memory: 1Gi
      proxy:
        resources: (3)
          limits:
            memory: 100Mi
          requests:
            cpu: 100m
            memory: 100Mi
      replicas: 2
  collection:
    logs:
      type: "fluentd"
      fluentd:
        resources: (4)
          limits:
            memory: 736Mi
          requests:
            cpu: 200m
            memory: 736Mi
```

(1) 必要に応じてログの CPU およびメモリーの制限および要求を指定します。Elasticsearch の場合は、要求値と制限値の両方を調整する必要があります。

(2) (3) 必要に応じて、ログビジュアライザーの CPU およびメモリーの制限および要求を指定します。

(4) 必要に応じて、ログコレクターの CPU およびメモリーの制限および要求を指定します。

### 7.4.2. ログビジュアライザーノードの冗長性のスケーリング

冗長性を確保するために、ログビジュアライザーをホストする Pod をスケーリングできます。

**手順**

1. **openshift-logging** プロジェクトで **ClusterLogging** カスタムリソース (CR) を編集します。

```
$ oc edit ClusterLogging instance
```

```
$ oc edit ClusterLogging instance

apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"

....

spec:
    visualization:
      type: "kibana"
      kibana:
        replicas: 1 (1)
```

(1) Kibana ノードの数を指定します。

## 7.5. ロギングサブシステムストレージの設定

Elasticsearch はメモリー集約型アプリケーションです。デフォルトのロギングサブシステムのインストールでは、メモリー要求とメモリー制限の両方に 16G のメモリーがデプロイされます。初期設定の OpenShift Container Platform ノードのセットは、Elasticsearch クラスターをサポートするのに十分な大きさではない場合があります。その場合、推奨されるサイズ以上のメモリーを使用して実行できるようにノードを OpenShift Container Platform クラスターに追加する必要があります。各 Elasticsearch ノードはこれより低い値のメモリー設定でも動作しますが、これは実稼働環境には推奨されません。

### 7.5.1. Red Hat OpenShift のロギングサブシステムのストレージに関する考慮事項

永続ボリュームがそれぞれの Elasticsearch デプロイメント設定に必要です。OpenShift Container Platform では、これは永続ボリューム要求 (PVC) を使用して実行されます。

**注記**

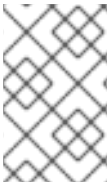
永続ストレージにローカルボリュームを使用する場合は、**LocalVolume** オブジェクトの **volumeMode: block** で記述される raw ブロックボリュームを使用しないでください。Elasticsearch は raw ブロックボリュームを使用できません。

OpenShift Elasticsearch Operator は Elasticsearch リソース名を使用して PVC に名前を付けます。

Fluentd は **systemd ジャーナル** および **/var/log/containers/** から Elasticsearch にログを送信します。

Elasticsearch では、大規模なマージ操作を実行するのに十分なメモリーが必要です。十分なメモリーがない場合は、応答しなくなります。この問題を回避するには、必要なアプリケーションのログデータの量を評価し、空き容量の約 2 倍を割り当てます。

デフォルトで、ストレージ容量が 85% に達すると、Elasticsearch は新規データのノードへの割り当てを停止します。90% になると、Elasticsearch は可能な場合に既存のシャードをそのノードから他のノードに移動しようとします。ただし、空き容量のレベルが 85% 未満のノードがない場合、Elasticsearch は新規インデックスの作成を拒否し、ステータスは RED になります。



**注記**

これらの基準値 (高い値および低い値を含む) は現行リリースにおける Elasticsearch のデフォルト値です。これらのデフォルト値は変更できます。アラートは同じデフォルト値を使用しますが、これらの値をアラートで変更することはできません。

### 7.5.2. 関連情報

- ログストアの永続ストレージの設定

## 7.6. ロギングサブシステムコンポーネントの CPU およびメモリー制限の設定

必要に応じて、ロギングサブシステムコンポーネントごとに CPU とメモリーの両方の制限を設定できます。

### 7.6.1. CPU およびメモリー制限の設定

ロギングサブシステムコンポーネントを使用すると、CPU とメモリーの両方の制限を調整できます。

**手順**

1. **openshift-logging** プロジェクトで **ClusterLogging** カスタムリソース (CR) を編集します。

```
$ oc -n openshift-logging edit ClusterLogging instance
```

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
  namespace: openshift-logging

...

spec:
  managementState: "Managed"
  logStore:
    type: "elasticsearch"
    elasticsearch:
      nodeCount: 3
      resources: 1
        limits:
          memory: 16Gi
        requests:
          cpu: 200m
          memory: 16Gi
      storage:
        storageClassName: "gp2"
        size: "200G"
      redundancyPolicy: "SingleRedundancy"
  visualization:
    type: "kibana"
    kibana:
      resources: 2
```

```
      limits:
        memory: 1Gi
      requests:
        cpu: 500m
        memory: 1Gi
    proxy:
      resources: 3
        limits:
          memory: 100Mi
        requests:
          cpu: 100m
          memory: 100Mi
    replicas: 2
collection:
  logs:
    type: "fluentd"
    fluentd:
      resources: 4
        limits:
          memory: 736Mi
        requests:
          cpu: 200m
          memory: 736Mi
```

1 必要に応じてログの CPU およびメモリーの制限および要求を指定します。Elasticsearch の場合は、要求値と制限値の両方を調整する必要があります。

2 3 必要に応じて、ログビジュアライザーの CPU およびメモリーの制限および要求を指定します。

4 必要に応じて、ログコレクターの CPU およびメモリーの制限および要求を指定します。

## 7.7. 容認を使用した OPENSHIFT LOGGING POD 配置の制御

taint と toleration を使用することで、ロギングシステム pod が特定のノードで実行され、その他のワークロードがそれらのノードで実行されないようにします。

テイントおよび容認は、単純な **key:value** のペアです。ノードのテイントはノードに対し、テイントを容認しないすべての Pod を拒否するよう指示します。

**key** は最大 253 文字までの文字列で、 **value** は最大 63 文字までの文字列になります。文字列は文字または数字で開始する必要があり、文字、数字、ハイフン、ドットおよびアンダースコアを含めることができます。

### toleration のあるサンプルロギングサブシステム CR

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
  namespace: openshift-logging

...
```

```
spec:
  managementState: "Managed"
  logStore:
    type: "elasticsearch"
    elasticsearch:
      nodeCount: 3
      tolerations: 1
      - key: "logging"
        operator: "Exists"
        effect: "NoExecute"
        tolerationSeconds: 6000
      resources:
        limits:
          memory: 16Gi
        requests:
          cpu: 200m
          memory: 16Gi
      storage: {}
      redundancyPolicy: "ZeroRedundancy"
  visualization:
    type: "kibana"
    kibana:
      tolerations: 2
      - key: "logging"
        operator: "Exists"
        effect: "NoExecute"
        tolerationSeconds: 6000
      resources:
        limits:
          memory: 2Gi
        requests:
          cpu: 100m
          memory: 1Gi
      replicas: 1
  collection:
    logs:
      type: "fluentd"
      fluentd:
        tolerations: 3
        - key: "logging"
          operator: "Exists"
          effect: "NoExecute"
          tolerationSeconds: 6000
        resources:
          limits:
            memory: 2Gi
          requests:
            cpu: 100m
            memory: 1Gi
```

1 この容認は Elasticsearch Pod に追加されます。

2 この容認は Kibana Pod に追加されます。

3 この容認はロギングコレクター Pod に追加されます。

### 7.7.1. 容認を使用したログストア Pod の配置の制御

ログストア Pod が実行するノードを制御し、Pod の容認を使用して他のワークロードがそれらのノードを使用しないようにできます。

**ClusterLogging** カスタムリソース (CR) を使用して容認をログストア Pod に適用し、テイントをノード仕様でノードに適用します。ノードのテイントは、テイントを容認しないすべての Pod を拒否するようノードに指示する **key:value pair** です。他の Pod にはない特定の **key:value** ペアを使用することで、ログストア Pod のみがそのノード上で実行されるようにできます。

デフォルトで、ログストア Pod には以下の容認があります。

```
tolerations:
- effect: "NoExecute"
  key: "node.kubernetes.io/disk-pressure"
  operator: "Exists"
```

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

**手順**

1. 以下のコマンドを使用して、OpenShift Logging Pod をスケジュールするノードにテイントを追加します。

```
$ oc adm taint nodes <node-name> <key>=<value>:<effect>
```

以下に例を示します。

```
$ oc adm taint nodes node1 elasticsearch=node:NoExecute
```

この例では、テイントをキー **elasticsearch**、値 **node**、およびテイントの効果 **NoExecute** のある **node1** に配置します。**NoExecute** effect のノードは、テイントに一致する Pod のみをスケジュールし、一致しない既存の Pod を削除します。

2. **ClusterLogging** CR の **logstore** セクションを編集し、Elasticsearch Pod の容認を設定します。

```
logStore:
  type: "elasticsearch"
  elasticsearch:
    nodeCount: 1
    tolerations:
    - key: "elasticsearch" (1)
      operator: "Exists" (2)
      effect: "NoExecute" (3)
      tolerationSeconds: 6000 (4)
```

(1) ノードに追加したキーを指定します。

(2) **Exists** Operator を指定し、キー **elasticsearch** のあるテイントがノードに存在する必要があるようにします。

3 **NoExecute** effect を指定します。

4 オプションで、**tolerationSeconds** パラメーターを指定して、エビクトされる前に Pod がノードにバインドされる期間を設定します。

この容認は、**oc adm taint** コマンドで作成されたテイントと一致します。この容認のある Pod は **node1** にスケジュールできます。

### 7.7.2. 容認を使用したログビジュアライザー Pod の配置の制御

ログビジュアライザー Pod が実行されるノードを制御し、Pod の容認を使用して他のワークロードがそれらのノードを使用しないようにできます。

**ClusterLogging** カスタムリソース (CR) を使用して容認をログビジュアライザー Pod に適用し、テイントをノード仕様でノードに適用します。ノードのテイントは、テイントを容認しないすべての Pod を拒否するようノードに指示する **key:value pair** です。他の Pod にはない特定の **key:value** ペアを使用することで、Kibana Pod のみをそのノード上で実行できます。

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

**手順**

1. 以下のコマンドを使用して、ログビジュアライザー Pod をスケジュールする必要のあるノードにテイントを追加します。

```
$ oc adm taint nodes <node-name> <key>=<value>:<effect>
```

以下に例を示します。

```
$ oc adm taint nodes node1 kibana=node:NoExecute
```

この例では、テイントをキー **kibana**、値 **node**、およびテイントの効果 **NoExecute** のある **node1** に配置します。**NoExecute** テイント effect を使用する必要があります。**NoExecute** は、テイントに一致する Pod のみをスケジュールし、一致しない既存の Pod を削除します。

2. **ClusterLogging** CR の **visualization** セクションを編集し、Kibana Pod の容認を設定します。

```
visualization:
  type: "kibana"
  kibana:
    tolerations:
    - key: "kibana" 1
      operator: "Exists" 2
      effect: "NoExecute" 3
      tolerationSeconds: 6000 4
```

1 ノードに追加したキーを指定します。

2 **Exists** Operator を指定して、**key/value/effect** パラメーターが一致するようにします。

3 **NoExecute** effect を指定します。

4 オプションで、**tolerationSeconds** パラメーターを指定して、エビクトされる前に Pod がノードにバインドされる期間を設定します。

この容認は、**oc adm taint** コマンドで作成されたテイントと一致します。この容認のある Pod は、**node1** にスケジュールできます。

### 7.7.3. 容認を使用したログコレクター Pod 配置の制御

ロギングコレクター Pod が実行するノードを確認し、Pod の容認を使用して他のワークロードがそれらのノードを使用しないようにできます。

容認を **ClusterLogging** カスタムリソース (CR) でロギングコレクター Pod に適用し、テイントをノード仕様でノードに適用します。テイントおよび容認を使用すると、Pod がメモリーや CPU などの問題によってエビクトされないようにできます。

デフォルトで、ロギングコレクター Pod には以下の容認があります。

```
tolerations:
- key: "node-role.kubernetes.io/master"
  operator: "Exists"
  effect: "NoExecute"
```

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

**手順**

1. 以下のコマンドを使用して、ロギングコレクター Pod がロギングコレクター Pod をスケジュールする必要のあるノードにテイントを追加します。

```
$ oc adm taint nodes <node-name> <key>=<value>:<effect>
```

以下に例を示します。

```
$ oc adm taint nodes node1 collector=node:NoExecute
```

この例では、テイントをキー **collector**、値 **node**、およびテイント effect **NoExecute** のある **node1** に配置します。**NoExecute** テイント effect を使用する必要があります。**NoExecute** は、テイントに一致する Pod のみをスケジュールし、一致しない既存の Pod を削除します。

2. **ClusterLogging** カスタムリソース (CR) の **collection** スタンザを編集して、ロギングコレクター Pod の容認を設定します。

```
collection:
  logs:
    type: "fluentd"
    fluentd:
      tolerations:
      - key: "collector" 1
```

```
      operator: "Exists" (2)
      effect: "NoExecute" (3)
      tolerationSeconds: 6000 (4)
```

1. ノードに追加したキーを指定します。
2. **Exists** Operator を指定して、**key/value/effect** パラメーターが一致するようにします。
3. **NoExecute** effect を指定します。
4. オプションで、**tolerationSeconds** パラメーターを指定して、エビクトされる前に Pod がノードにバインドされる期間を設定します。

この容認は、**oc adm taint** コマンドで作成されたテイントと一致します。この容認のある Pod は、**node1** にスケジュールできます。

### 7.7.4. 関連情報

- ノードテイントを使用した Pod 配置の制御

## 7.8. ノードセレクターを使用したロギングサブシステムリソースの移動

ノードセレクターを使用して Elasticsearch、Kibana Pod を異なるノードにデプロイできます。

### 7.8.1. OpenShift Logging リソースの移動

Elasticsearch および Kibana などのロギングシステムコンポーネントの pod を異なるノードにデプロイするように Cluster Logging Operator を設定できます。Cluster Logging Operator Pod をインストールした場所から移動することはできません。

たとえば、Elasticsearch Pod の CPU、メモリーおよびディスクの要件が高いために、この Pod を別のノードに移動できます。

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。これらの機能はデフォルトでインストールされません。

**手順**

1. **openshift-logging** プロジェクトで **ClusterLogging** カスタムリソース (CR) を編集します。

```
$ oc edit ClusterLogging instance
```

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogging

...

spec:
  collection:
    logs:
      fluentd:
```

```
      resources: null
    type: fluentd
  logStore:
    elasticsearch:
      nodeCount: 3
      nodeSelector: 1
        node-role.kubernetes.io/infra: ''
      tolerations:
      - effect: NoSchedule
        key: node-role.kubernetes.io/infra
        value: reserved
      - effect: NoExecute
        key: node-role.kubernetes.io/infra
        value: reserved
      redundancyPolicy: SingleRedundancy
      resources:
        limits:
          cpu: 500m
          memory: 16Gi
        requests:
          cpu: 500m
          memory: 16Gi
      storage: {}
    type: elasticsearch
  managementState: Managed
  visualization:
    kibana:
      nodeSelector: 2
        node-role.kubernetes.io/infra: ''
      tolerations:
      - effect: NoSchedule
        key: node-role.kubernetes.io/infra
        value: reserved
      - effect: NoExecute
        key: node-role.kubernetes.io/infra
        value: reserved
      proxy:
        resources: null
      replicas: 1
      resources: null
    type: kibana

...
```

1 2 適切な値が設定された **nodeSelector** パラメーターを、移動する必要のあるコンポーネントに追加します。表示されている形式の **nodeSelector** を使用することも、ノードに指定された値に基づいて **<key>: <value>** ペアを使用することもできます。インフラストラクチャーノードにテイントを追加した場合は、一致する容認も追加します。

**検証**

コンポーネントが移動したことを確認するには、**oc get pod -o wide** コマンドを使用できます。

以下に例を示します。

- Kibana Pod を **ip-10-0-147-79.us-east-2.compute.internal** ノードから移動する必要がある場合は、以下を実行します。

```
$ oc get pod kibana-5b8bdf44f9-ccpq9 -o wide
```

**出力例**

```
NAME                     READY   STATUS    RESTARTS   AGE   IP           NODE
NOMINATED NODE   READINESS GATES
kibana-5b8bdf44f9-ccpq9   2/2     Running   0          27s   10.129.2.18   ip-10-0-147-79.us-
east-2.compute.internal   <none>           <none>
```

- Kibana Pod を、専用インフラストラクチャーノードである **ip-10-0-139-48.us-east-2.compute.internal** ノードに移動する必要がある場合は、以下を実行します。

```
$ oc get nodes
```

**出力例**

```
NAME                                         STATUS   ROLES          AGE    VERSION
ip-10-0-133-216.us-east-2.compute.internal   Ready    master         60m    v1.23.0
ip-10-0-139-146.us-east-2.compute.internal   Ready    master         60m    v1.23.0
ip-10-0-139-192.us-east-2.compute.internal   Ready    worker         51m    v1.23.0
ip-10-0-139-241.us-east-2.compute.internal   Ready    worker         51m    v1.23.0
ip-10-0-147-79.us-east-2.compute.internal    Ready    worker         51m    v1.23.0
ip-10-0-152-241.us-east-2.compute.internal   Ready    master         60m    v1.23.0
ip-10-0-139-48.us-east-2.compute.internal    Ready    infra          51m    v1.23.0
```

ノードには **node-role.kubernetes.io/infra: ''** ラベルがあることに注意してください。

```
$ oc get node ip-10-0-139-48.us-east-2.compute.internal -o yaml
```

**出力例**

```
kind: Node
apiVersion: v1
metadata:
  name: ip-10-0-139-48.us-east-2.compute.internal
  selfLink: /api/v1/nodes/ip-10-0-139-48.us-east-2.compute.internal
  uid: 62038aa9-661f-41d7-ba93-b5f1b6ef8751
  resourceVersion: '39083'
  creationTimestamp: '2020-04-13T19:07:55Z'
  labels:
    node-role.kubernetes.io/infra: ''
...
```

- Kibana Pod を移動するには、**ClusterLogging** CR を編集してノードセレクターを追加します。

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogging

...
```

```
spec:

...

  visualization:
    kibana:
      nodeSelector: 1
        node-role.kubernetes.io/infra: ''
      proxy:
        resources: null
      replicas: 1
      resources: null
    type: kibana
```

1 ノード仕様のラベルに一致するノードセレクターを追加します。

- CR を保存した後に、現在の Kibana Pod は終了し、新規 Pod がデプロイされます。

```
$ oc get pods
```

**出力例**

```
NAME                                            READY   STATUS        RESTARTS   AGE
cluster-logging-operator-84d98649c4-zb9g7       1/1     Running       0          29m
elasticsearch-cdm-hwv01pf7-1-56588f554f-kpmlg   2/2     Running       0          28m
elasticsearch-cdm-hwv01pf7-2-84c877d75d-75wqj   2/2     Running       0          28m
elasticsearch-cdm-hwv01pf7-3-f5d95b87b-4nx78    2/2     Running       0          28m
fluentd-42dzz                                   1/1     Running       0          28m
fluentd-d74rq                                   1/1     Running       0          28m
fluentd-m5vr9                                   1/1     Running       0          28m
fluentd-nkxl7                                   1/1     Running       0          28m
fluentd-pdvqb                                   1/1     Running       0          28m
fluentd-tflh6                                   1/1     Running       0          28m
kibana-5b8bdf44f9-ccpq9                         2/2     Terminating   0          4m11s
kibana-7d85dcffc8-bfpfp                         2/2     Running       0          33s
```

- 新規 Pod が **ip-10-0-139-48.us-east-2.compute.internal** ノードに置かれます。

```
$ oc get pod kibana-7d85dcffc8-bfpfp -o wide
```

**出力例**

```
NAME                     READY   STATUS       RESTARTS   AGE   IP          NODE
NOMINATED NODE   READINESS GATES
kibana-7d85dcffc8-bfpfp  2/2     Running      0          43s   10.131.0.22   ip-10-0-139-48.us-
east-2.compute.internal   <none>          <none>
```

- しばらくすると、元の Kibana Pod が削除されます。

```
$ oc get pods
```

**出力例**

```
NAME                                            READY   STATUS    RESTARTS   AGE
cluster-logging-operator-84d98649c4-zb9g7       1/1     Running   0          30m
elasticsearch-cdm-hwv01pf7-1-56588f554f-kpmlg   2/2     Running   0          29m
elasticsearch-cdm-hwv01pf7-2-84c877d75d-75wqj   2/2     Running   0          29m
elasticsearch-cdm-hwv01pf7-3-f5d95b87b-4nx78    2/2     Running   0          29m
fluentd-42dzz                                   1/1     Running   0          29m
fluentd-d74rq                                   1/1     Running   0          29m
fluentd-m5vr9                                   1/1     Running   0          29m
fluentd-nkxl7                                   1/1     Running   0          29m
fluentd-pdvqb                                   1/1     Running   0          29m
fluentd-tflh6                                   1/1     Running   0          29m
kibana-7d85dcffc8-bfpfp                         2/2     Running   0          62s
```

## 7.9. SYSTEMD-JOURNALD および FLUENTD の設定

Fluentd のジャーナルからの読み取りや、ジャーナルのデフォルト設定値は非常に低く、ジャーナルがシステムサービスからのロギング速度に付いていくことができないためにジャーナルエントリーが失われる可能性があります。

ジャーナルでエントリーが失われるのを防ぐことができるように **RateLimitIntervalSec=30s** および **RateLimitBurst=10000** (必要な場合はさらに高い値) を設定することが推奨されます。

### 7.9.1. OpenShift Logging 用の systemd-journald の設定

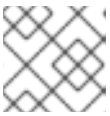
プロジェクトのスケールアップ時に、デフォルトのロギング環境にはいくらかの調整が必要になる場合があります。

たとえば、ログが見つからない場合は、journald の速度制限を引き上げる必要がある場合があります。一定期間保持するメッセージ数を調整して、OpenShift Logging がログをドロップせずに過剰なリソースを使用しないようにすることができます。

また、ログを圧縮する必要があるかどうか、ログを保持する期間、ログを保存する方法、ログを保存するかどうかやその他の設定を決定することもできます。

**手順**

1. 必要な設定で **/etc/systemd/journald.conf** ファイルが含まれる Butane 設定ファイル **40-worker-custom -journald.bu** を作成します。



**注記**

Butane の詳細は、Butane を使用したマシン設定の作成を参照してください。

```
variant: openshift
version: 4.10.0
metadata:
  name: 40-worker-custom-journald
  labels:
    machineconfiguration.openshift.io/role: "worker"
storage:
  files:
  - path: /etc/systemd/journald.conf
    mode: 0644 (1)
```

```
    overwrite: true
    contents:
      inline: |
        Compress=yes (2)
        ForwardToConsole=no (3)
        ForwardToSyslog=no
        MaxRetentionSec=1month (4)
        RateLimitBurst=10000 (5)
        RateLimitIntervalSec=30s
        Storage=persistent (6)
        SyncIntervalSec=1s (7)
        SystemMaxUse=8G (8)
        SystemKeepFree=20% (9)
        SystemMaxFileSize=10M (10)
```

(1) **journald.conf** ファイルのパーミッションを設定します。**0644** パーミッションを設定することが推奨されます。

(2) ログがファイルシステムに書き込まれる前にそれらのログを圧縮するかどうかを指定します。**yes** を指定してメッセージを圧縮するか、**no** を指定して圧縮しないようにします。デフォルトは **yes** です。

(3) ログメッセージを転送するかどうかを設定します。それぞれについて、デフォルトで **no** に設定されます。以下を指定します。

- **ForwardToConsole**: ログをシステムコンソールに転送します。
- **ForwardToKsmg**: ログをカーネルログバッファーに転送します。
- **ForwardToSyslog**: syslog デーモンに転送します。
- **ForwardToWall**: メッセージを wall メッセージとしてすべてのログインしているユーザーに転送します。

(4) ジャーナルエントリーを保存する最大時間を指定します。数字を入力して秒数を指定します。または、year、month、week、day、h または m などの単位を含めます。無効にするには **0** を入力します。デフォルトは **1month** です。

(5) レート制限を設定します。**RateLimitIntervalSec** で定義される期間に、**RateLimitBurst** で指定される以上のログが受信される場合、この期間内の追加のメッセージはすべてこの期間が終了するまでにドロップされます。デフォルト値である **RateLimitIntervalSec=30s** および **RateLimitBurst=10000** を設定することが推奨されます。

(6) ログの保存方法を指定します。デフォルトは **persistent** です。

- **volatile**: ログを **/var/log/journal/** のメモリーに保存します。
- **persistent**: ログを **/var/log/journal/** のディスクに保存します。systemd は存在しない場合はディレクトリーを作成します。
- **auto**: ディレクトリーが存在する場合に、ログを **/var/log/journal/** に保存します。存在しない場合は、systemd はログを **/run/systemd/journal** に一時的に保存します。
- **none**: ログを保存しません。systemd はすべてのログをドロップします。

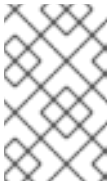
(7) ERR、WARNING、NOTICE、INFO、および DEBUG ログについてジャーナルファイルを

ディスクに同期させるまでのタイムアウトを指定します。 systemd は、**CRIT**、**ALERT**、または **EMERG** ログの受信後すぐに同期を開始します。デフォルトは **1s** です。

8 ジャーナルが使用できる最大サイズを指定します。デフォルトは **8G** です。

9 systemd が残す必要のあるディスク領域のサイズを指定します。デフォルトは **20%** です。

10 **/var/log/journal** に永続的に保存される個別のジャーナルファイルの最大サイズを指定します。デフォルトは **10M** です。



**注記**

レート制限を削除する場合、システムロギングデーモンの CPU 使用率が高くなることがあります。 以前はスロットリングされていた可能性のあるメッセージが処理されるためです。

systemd 設定の詳細については、https://www.freedesktop.org/software/systemd/man/journald.conf.html を参照してください。このページに一覧表示されるデフォルト設定は OpenShift Container Platform には適用されない可能性があります。

2. Butane を使用して、ノードに配信される設定を含む **MachineConfig** オブジェクトファイル (**40-worker-custom-journald.yaml**) を生成します。

```
$ butane 40-worker-custom-journald.bu -o 40-worker-custom-journald.yaml
```

3. マシン設定を適用します。以下に例を示します。

```
$ oc apply -f 40-worker-custom-journald.yaml
```

コントローラーは新規の **MachineConfig** オブジェクトを検出し、新規の **rendered-worker-<hash>** バージョンを生成します。

4. 新規のレンダリングされた設定の各ノードへのロールアウトのステータスをモニターします。

```
$ oc describe machineconfigpool/worker
```

**出力例**

```
Name:         worker
Namespace:
Labels:       machineconfiguration.openshift.io/mco-built-in=
Annotations:  <none>
API Version:  machineconfiguration.openshift.io/v1
Kind:         MachineConfigPool

...

Conditions:
  Message:
  Reason:               All nodes are updating to rendered-worker-
913514517bcea7c93bd446f4830bc64e
```

# 7.10. メンテナンスとサポート

## 7.10.1. サポートされる設定

Red Hat OpenShift のロギングサブシステムを設定するためにサポートされている方法は、このドキュメントで説明されているオプションを使用して設定することです。サポートされていない他の設定は使用しないでください。設定のパラダイムが OpenShift Container Platform リリース間で変更される可能性があり、このような変更は、設定のすべての可能性が制御されている場合のみ適切に対応できます。本書で説明されている設定以外の設定を使用する場合、OpenShift Elasticsearch Operator および Red Hat OpenShift Logging Operator が差分を調整するため、変更内容は失われます。Operator はデフォルトで定義された状態にすべて戻します。

**注記**

OpenShift Container Platform ドキュメントで説明されていない設定を実行する **必要がある** 場合、Red Hat OpenShift Logging Operator または OpenShift Elasticsearch Operator を **Unmanaged** (管理外) に設定する **必要があります**。管理外の OpenShift Logging 環境は **サポート外** であり、OpenShift Logging を **Managed** に戻すまで変更を受信しません。

## 7.10.2. サポートされない設定

以下のコンポーネントを変更するには、Red Hat OpenShift Logging Operator を Unmanaged (管理外) の状態に設定する必要があります。

- **Elasticsearch** CR
- Kibana デプロイメント
- **fluent.conf** ファイル
- Fluentd デーモンセット

以下のコンポーネントを変更するには、OpenShift Elasticsearch Operator を Unmanaged(管理外) の状態に設定する必要があります。

- Elasticsearch デプロイメントファイル。

明示的にサポート対象外とされているケースには、以下が含まれます。

- **デフォルトのログローテーションの設定**。デフォルトのログローテーション設定は変更できません。
- **収集したログの場所の設定**。ログコレクターの出力ファイルの場所を変更することはできません。デフォルトは **/var/log/fluentd/fluentd.log** です。
- **ログコレクションのスロットリング**。ログコレクターによってログが読み取られる速度を調整して減速することはできません。
- **環境変数を使用したロギングコレクターの設定**。環境変数を使用してログコレクターを変更することはできません。
- **ログコレクターによってログを正規化する方法の設定**。デフォルトのログの正規化を変更することはできません。

### 7.10.3. 管理外の Operator のサポートポリシー

Operator の **管理状態** は、Operator が設計通りにクラスター内の関連するコンポーネントのリソースをアクティブに管理しているかどうかを定めます。Operator が **unmanaged** 状態に設定されていると、これは設定の変更に応答せず、更新を受信しません。

これは非実稼働クラスターやデバッグ時に便利ですが、管理外の状態の Operator はサポートされず、クラスター管理者は個々のコンポーネント設定およびアップグレードを完全に制御していることを前提としています。

Operator は以下の方法を使用して管理外の状態に設定できます。

- **個別の Operator 設定**
  個別の Operator には、それらの設定に **managementState** パラメーターがあります。これは Operator に応じてさまざまな方法でアクセスできます。たとえば、Red Hat OpenShift Logging Operator は管理するカスタムリソース (CR) を変更することによってこれを実行しますが、Cluster Samples Operator はクラスター全体の設定リソースを使用します。

  **managementState** パラメーターを **Unmanaged** に変更する場合、Operator はそのリソースをアクティブに管理しておらず、コンポーネントに関連するアクションを取らないことを意味します。Operator によっては、クラスターが破損し、手動リカバリーが必要になる可能性があるため、この管理状態に対応しない可能性があります。

  **警告**

  個別の Operator を **Unmanaged** 状態に変更すると、特定のコンポーネントおよび機能がサポート対象外になります。サポートを継続するには、報告された問題を **Managed** 状態で再現する必要があります。

- **Cluster Version Operator (CVO) のオーバーライド**
  **spec.overrides** パラメーターを CVO の設定に追加すると、管理者はコンポーネントの CVO の動作に対してオーバーライドの一覧を追加できます。コンポーネントの **spec.overrides[].unmanaged** パラメーターを **true** に設定すると、クラスターのアップグレードがブロックされ、CVO のオーバーライドが設定された後に管理者にアラートが送信されます。

  > Disabling ownership via cluster version overrides prevents upgrades. Please remove overrides before continuing.

  **警告**

  CVO のオーバーライドを設定すると、クラスター全体がサポートされない状態になります。サポートを継続するには、オーバーライドを削除した後に、報告された問題を再現する必要があります。

# 第8章 LOKISTACK を使用したロギング

ロギングサブシステムのドキュメントでは、**LokiStack** はロギングサブシステムでサポートされている Loki と Web プロキシーと OpenShift Container Platform 認証統合の組み合わせを指します。LokiStack のプロキシーは、OpenShift Container Platform 認証を使用してマルチテナンシーを適用します。**Loki** では、ログストアを個別のコンポーネントまたは外部ストアと呼んでいます。

Loki は、水平方向にスケーラブルで可用性の高いマルチテナントログ集約システムであり、現在、ロギングサブシステムのログストアとして Elasticsearch の代替として提供されています。Elasticsearch は、取り込み中に受信ログレコードを完全にインデックス化します。Loki は、取り込み中にいくつかの固定ラベルのみをインデックスに登録し、ログが保存されるまで、より複雑な解析が延期されるので Loki がより迅速にログを収集できるようになります。LogQL ログクエリー言語を使用して Loki にクエリー を実行できます。

## 8.1. デプロイメントのサイズ

Loki のサイズは **N<x>.<size>** の形式に従います。**<N>** はインスタンスの数を、**<size>** はパフォーマンスの機能を指定します。

**注記**

1x.extra-small はデモ用であり、サポートされていません。

**表8.1 Loki のサイズ**

| | 1x.extra-small | 1x.small | 1x.medium |
|---|---|---|---|
| **データ転送** | デモ使用のみ。 | 500GB/day | 2TB/日 |
| **1秒あたりのクエリー数 (QPS)** | デモ使用のみ。 | 200 ミリ秒で 25 - 50 QPS | 200 ミリ秒で 25 ~ 75 QPS |
| **レプリケーション係数** | なし | 2 | 3 |
| **合計 CPU 要求** | 仮想 CPU 5 個 | 仮想 CPU 36 個 | 仮想 CPU 54 個 |
| **合計メモリー要求** | 7.5Gi | 63Gi | 139Gi |
| **ディスク要求の合計** | 150Gi | 300Gi | 450Gi |

### 8.1.1. サポート対象の API カスタムリソース定義

LokiStack は開発中で、現在すべての API がサポートされているわけではありません。

| CustomResourceDefinition (CRD) | ApiVersion | サポートの状態 |
|---|---|---|
| LokiStack | lokistack.loki.grafana.com/v1 | 5.5 でサポート |

| CustomResourceDefinition (CRD) | ApiVersion | サポートの状態 |
|---|---|---|
| RulerConfig | rulerconfig.loki.grafana/v1beta1 | テクノロジープレビュー |
| AlertingRule | alertingrule.loki.grafana/v1beta1 | テクノロジープレビュー |
| RecordingRule | recordingrule.loki.grafana/v1beta1 | テクノロジープレビュー |

**重要**

**RulerConfig**、**AlertingRule**、**RecordingRule** カスタムリソース定義 (CRD) の使用は、テクノロジープレビュー機能のみとなっています。テクノロジープレビュー機能は、Red Hat 製品サポートのサービスレベルアグリーメント (SLA) の対象外であり、機能的に完全ではない場合があります。Red Hat は、実稼働環境でこれらを使用することを推奨していません。テクノロジープレビュー機能は、最新の製品機能をいち早く提供して、開発段階で機能のテストを行いフィードバックを提供していただくことを目的としています。

Red Hat のテクノロジープレビュー機能のサポート範囲に関する詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

## 8.2. LOKISTACK のデプロイ

OpenShift Container Platform Web コンソールを使用して LokiStack をデプロイできます。

**前提条件**

- Red Hat OpenShift Operator 5.5 以降のロギングサブシステム
- 対応ログストア (AWS S3、Google Cloud Storage、Azure、Swift、Minio、OpenShift Data Foundation)

**手順**

1. **Loki Operator** Operator をインストールします。
   a. OpenShift Container Platform Web コンソールで、**Operators → OperatorHub** をクリックします。
   b. 使用可能な Operator のリストから **Loki Operator** を選択し、**Install** をクリックします。
   c. **Installation Mode** で、**All namespaces on the cluster**を選択します。
   d. **Installed Namespace** で、**openshift-operators-redhat** を選択します。**openshift-operators-redhat** namespace を指定する必要があります。**openshift-operators** namespace には信頼されていないコミュニティー Operator が含まれる可能性があり、OpenShift Container Platform メトリックと同じ名前でメトリックを公開する可能性があるため、これによって競合が生じる可能性があります。
   e. **Enable operator recommended cluster monitoring on this namespace**を選択します。

このオプションは、namespace オブジェクトに **openshift.io/cluster-monitoring: "true"** ラベルを設定します。クラスターモニタリングが **openshift-operators-redhat** namespace を収集できるように、このオプションを選択する必要があります。

f. **Approval Strategy** を選択します。

- **Automatic** ストラテジーにより、Operator Lifecycle Manager (OLM) は新規バージョンが利用可能になると Operator を自動的に更新できます。
- **Manual** ストラテジーには、Operator の更新を承認するための適切な認証情報を持つユーザーが必要です。

g. **Install** をクリックします。

h. Loki Operator がインストールされていることを確認します。**Operators→ Installed Operators** ページにアクセスして、**Loki Operator** を探します。

i. **Loki Operator** がすべてのプロジェクトで **Succeeded** の **Status** でリストされていることを確認します。

2. **access_key_id** フィールドと **access_key_secret** フィールドを使用して AWS 認証情報と **bucketnames**、**endpoint**、および **region** を指定し、オブジェクトストレージの場所を定義する **Secret** YAML ファイルを作成します。以下に例を示します。

```
apiVersion: v1
kind: Secret
metadata:
  name: logging-loki-s3
  namespace: openshift-logging
stringData:
  access_key_id: AKIAIOSFODNN7EXAMPLE
  access_key_secret: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
  bucketnames: s3-bucket-name
  endpoint: https://s3.eu-central-1.amazonaws.com
  region: eu-central-1
```

3. **LokiStack** カスタムリソース (CR) を作成します。

```
apiVersion: loki.grafana.com/v1
kind: LokiStack
metadata:
  name: logging-loki
  namespace: openshift-logging
spec:
  size: 1x.small
  storage:
    schemas:
    - version: v12
      effectiveDate: "2022-06-01"
    secret:
      name: logging-loki-s3
      type: s3
  storageClassName: gp2
  tenants:
    mode: openshift-logging
```

4. **LokiStack** CR を適用します。

```
$ oc apply -f logging-loki.yaml
```

5. **ClusterLogging** カスタムリソース (CR) を作成します。

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogging
metadata:
  name: instance
  namespace: openshift-logging
spec:
  managementState: Managed
  logStore:
    type: lokistack
    lokistack:
      name: logging-loki
  collection:
    type: vector
```

6. **ClusterLogging** CR を適用します。

```
$ oc apply -f cr-lokistack.yaml
```

7. RedHat OpenShift ログコンソールプラグインを有効にします。

   a. OpenShift Container Platform Web コンソールで、**Operators → Installed Operators** をクリックします。

   b. **RedHat OpenShift Logging** Operator を選択します。

   c. コンソールプラグインで **無効** をクリックします。

   d. **有効**、**保存** の順に選択します。この変更により、**openshift-console** Pod が再起動されます。

   e. Pod の再起動後、Web コンソールの更新が利用可能であるという通知を受け取り、更新するよう求められます。

   f. Web コンソールを更新したら、左側のメインメニューから **監視** をクリックします。**Logs** の新しいオプションが利用可能です。

## 8.3. ログの LOKISTACK への転送

LokiStack ゲートウェイへのログ転送を設定するには、**ClusterLogging** カスタムリソース (CR) を作成する必要があります。

**前提条件**

- Red Hat OpenShift バージョン 5.5 以降のログサブシステムがクラスターにインストールされている。
- Loki Operator がクラスターにインストールされている。

**手順**

- **ClusterLogging** カスタムリソース (CR) を作成します。

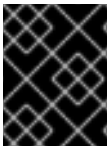
```
apiVersion: logging.openshift.io/v1
kind: ClusterLogging
metadata:
  name: instance
  namespace: openshift-logging
spec:
  managementState: Managed
  logStore:
    type: lokistack
    lokistack:
      name: logging-loki
  collection:
    type: vector
```

### 8.3.1. Loki レート制限エラーのトラブルシューティング

Log Forwarder API がレート制限を超える大きなメッセージブロックを Loki に転送すると、Loki により、レート制限 (**429**) エラーが生成されます。

これらのエラーは、通常の動作中に発生する可能性があります。たとえば、すでにいくつかのログがあるクラスターにログサブシステムを追加する場合、ログサブシステムが既存のログエントリーをすべて取り込もうとするとレート制限エラーが発生する可能性があります。この場合、新しいログの追加速度が合計レート制限よりも低い場合、履歴データは最終的に取り込まれ、ユーザーの介入を必要とせずにレート制限エラーが解決されます。

レート制限エラーが引き続き発生する場合は、**LokiStack** カスタムリソース (CR) を変更することで問題を解決できます。



**重要**

**LokiStack** CR は、Grafana がホストする Loki では利用できません。このトピックは、Grafana がホストする Loki サーバーには適用されません。

**条件**

- Log Forwarder API は、ログを Loki に転送するように設定されている。
- システムは、次のような 2MB を超えるメッセージのブロックを Loki に送信する。以下に例を示します。

```
"values":[["1630410392689800468","{\"kind\":\"Event\",\"apiVersion\":\
.......
......
......
......
\"received_at\":\"2021-08-31T11:46:32.800278+00:00\",\"version\":\"1.7.4
1.6.0\"}},\"@timestamp\":\"2021-08-
31T11:46:32.799692+00:00\",\"viaq_index_name\":\"audit-
write\",\"viaq_msg_id\":\"MzFjYjJkZjItNjY0MC00YWU4LWIwMTEtNGNmM2E5ZmViMGU4\",\"log_type\":\"audit\"}"]]}]}
```

- **oc logs -n openshift-logging -l component=collector** と入力すると、クラスター内のコレクターログに、次のいずれかのエラーメッセージを含む行が表示されます。

```
429 Too Many Requests Ingestion rate limit exceeded
```

**Vector エラーメッセージの例**

```
2023-08-25T16:08:49.301780Z  WARN sink{component_kind="sink"
component_id=default_loki_infra component_type=loki component_name=default_loki_infra}:
vector::sinks::util::retries: Retrying after error. error=Server responded with an error: 429 Too
Many Requests internal_log_rate_limit=true
```

**Fluentd エラーメッセージの例**

```
2023-08-30 14:52:15 +0000 [warn]: [default_loki_infra] failed to flush the buffer. retry_times=2
next_retry_time=2023-08-30 14:52:19 +0000
chunk="604251225bf5378ed1567231a1c03b8b"
error_class=Fluent::Plugin::LokiOutput::LogPostError error="429 Too Many Requests
Ingestion rate limit exceeded for user infrastructure (limit: 4194304 bytes/sec) while
attempting to ingest '4082' lines totaling '7820025' bytes, reduce log volume or contact your
Loki administrator to see if the limit can be increased\n"
```

このエラーは受信側にも表示されます。たとえば、LokiStack 取り込み Pod で以下を行います。

**Loki 取り込みエラーメッセージの例**

```
level=warn ts=2023-08-30T14:57:34.155592243Z caller=grpc_logging.go:43
duration=1.434942ms method=/logproto.Pusher/Push err="rpc error: code = Code(429) desc
= entry with timestamp 2023-08-30 14:57:32.012778399 +0000 UTC ignored, reason: 'Per
stream rate limit exceeded (limit: 3MB/sec) while attempting to ingest for stream
```

**手順**

- **LokiStack** CR の **ingestionBurstSize** および **ingestionRate** フィールドを更新します。

```
apiVersion: loki.grafana.com/v1
kind: LokiStack
metadata:
  name: logging-loki
  namespace: openshift-logging
spec:
  limits:
    global:
      ingestion:
        ingestionBurstSize: 16 (1)
        ingestionRate: 8 (2)
# ...
```

(1) **ingestionBurstSize** フィールドは、ディストリビューターレプリカごとに最大ローカルレート制限サンプルサイズを MB 単位で定義します。この値はハードリミットです。この値を、少なくとも 1 つのプッシュリクエストで想定される最大ログサイズに設定します。**ingestionBurstSize** 値より大きい単一リクエストは使用できません。

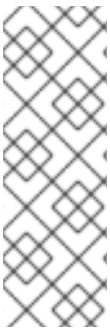
2 **ingestionRate** フィールドは、1秒あたりに取り込まれるサンプルの最大量 (MB 単位) に対するソフト制限です。ログのレートが制限を超えているにもかかわらず、コレクターがログの送信を再試行すると、レート制限エラーが発生します。合計平均が制限よりも少ない場合に限り、システムは回復し、ユーザーの介入なしでエラーが解決されます。

## 8.4. 関連情報

- Loki クエリー言語 (LogQL) ドキュメント
- Grafana ダッシュボードのドキュメント
- Loki オブジェクトストレージのドキュメント
- Loki Operator **IngestionLimitSpec** ドキュメント
- Loki ストレージスキーマドキュメント

# 第9章 リソースのログの表示

OpenShift CLI (oc) および Web コンソールを使用して、ビルド、デプロイメント、および Pod などの各種リソースのログを表示できます。



**注記**

リソースログは、制限されたログ表示機能を提供するデフォルトの機能です。ログの取得および表示のエクスペリエンスを強化するには、OpenShift Logging をインストールすることが推奨されます。ロギングシステムは、ノードシステムの監査ログ、アプリケーションコンテナーログ、およびインフラストラクチャーログなどの OpenShift Container Platform クラスターからのすべてのログを専用のログストアに集計します。次に、Kibana インターフェイス を使用してログデータをクエリーし、検出し、可視化できます。リソースログは、ロギングサブシステムのログストアにアクセスしません。

## 9.1. リソースログの表示

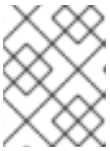
OpenShift CLI (oc) および Web コンソールで、各種リソースのログを表示できます。ログの末尾から読み取られるログ。

### 前提条件

- OpenShift CLI (oc) へのアクセス。

### 手順 (UI)

1. OpenShift Container Platform コンソールで **Workloads → Pods** に移動するか、調査するリソースから Pod に移動します。



**注記**

ビルドなどの一部のリソースには、直接クエリーする Pod がありません。このような場合は、リソースの **Details** ページで **Logs** リンクを特定できます。

2. ドロップダウンメニューからプロジェクトを選択します。
3. 調査する Pod の名前をクリックします。
4. **Logs** をクリックします。

### 手順 (CLI)

- 特定の Pod のログを表示します。

```
$ oc logs -f <pod_name> -c <container_name>
```

ここでは、以下のようになります。

**-f**
オプション: ログに書き込まれている内容に沿って出力することを指定します。

**<pod_name>**
Pod の名前を指定します。

**<container_name>**

オプション: コンテナーの名前を指定します。Pod に複数のコンテナーがある場合は、コンテナー名を指定する必要があります。

以下に例を示します。

```
$ oc logs ruby-58cd97df55-mww7r
```

```
$ oc logs -f ruby-57f7f4855b-znl92 -c ruby
```

ログファイルの内容が出力されます。

- 特定のリソースのログを表示します。

```
$ oc logs <object_type>/<resource_name> 1
```

1 リソースタイプおよび名前を指定します。

以下に例を示します。

```
$ oc logs deployment/ruby
```

ログファイルの内容が出力されます。

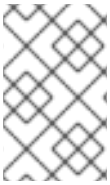
# 第10章 KIBANA を使用したクラスターログの表示

ロギングサブシステムには、収集されたログデータを視覚化するための Web コンソールが含まれています。現時点で、OpenShift Container Platform では、可視化できるように Kibana コンソールをデプロイします。

ログビジュアライザーを使用して、データで以下を実行できます。

- **Discover** タブを使用してデータを検索し、参照します。
- **Visualize** タブを使用してデータのグラフを表示し、データをマップします。
- **Dashboard** タブを使用してカスタムダッシュボードを作成し、表示します。

Kibana インターフェイスの使用および設定は、このドキュメントでは扱いません。詳細については、Kibana ドキュメント を参照してください。



**注記**

監査ログは、デフォルトでは内部 OpenShift Container Platform Elasticsearch インスタンスに保存されません。Kibana で監査ログを表示するには、ログ転送 API を使用して、監査ログの **default** 出力を使用するパイプラインを設定する必要があります。

## 10.1. KIBANA インデックスパターンの定義

インデックスパターンは、可視化する必要のある Elasticsearch インデックスを定義します。Kibana でデータを確認し、可視化するには、インデックスパターンを作成する必要があります。

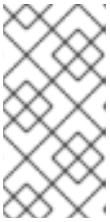
**前提条件**

- Kibana で **infra** および **audit** インデックスを表示するには、ユーザーには **cluster-admin** ロール、 **cluster-reader** ロール、または両方のロールが必要です。デフォルトの **kubeadmin** ユーザーには、これらのインデックスを表示するための適切なパーミッションがあります。**default**、**kube-** および **openshift-** プロジェクトで Pod およびログを表示できる場合に、これらのインデックスにアクセスできるはずです。以下のコマンドを使用して、現在のユーザーが適切なパーミッションを持っているかどうかを確認できます。

```
$ oc auth can-i get pods/log -n <project>
```

**出力例**

```
yes
```



**注記**

監査ログは、デフォルトでは内部 OpenShift Container Platform Elasticsearch インスタンスに保存されません。Kibana で監査ログを表示するには、ログ転送 API を使用して監査ログの **default** 出力を使用するパイプラインを設定する必要があります。

- Elasticsearch ドキュメントは、インデックスパターンを作成する前にインデックス化する必要があります。これは自動的に実行されますが、新規または更新されたクラスターでは数分の時間がかかる可能性があります。

### 手順

Kibana でインデックスパターンを定義し、ビジュアライゼーションを作成するには、以下を実行します。

1. OpenShift Container Platform コンソールで、Application Launcher をクリックし、**Logging** を選択します。
2. **Management → Index Patterns → Create index pattern** をクリックして Kibana インデックスパターンを作成します。
   - 各ユーザーは、プロジェクトのログを確認するために、Kibana に初めてログインする際にインデックスパターンを手動で作成する必要があります。ユーザーは **app** という名前のインデックスパターンを作成し、**@timestamp** 時間フィールドを使用してコンテナーログを表示する必要があります。
   - 管理ユーザーはそれぞれ、最初に Kibana にログインする際に、**@timestamp** 時間フィールドを使用して **app**、 **infra** および **audit** インデックスのインデックスパターンを作成する必要があります。
3. 新規インデックスパターンから Kibana のビジュアライゼーション (Visualization) を作成します。

## 10.2. KIBANA でのクラスターログの表示

Kibana Web コンソールでクラスターのログを表示します。Kibana でデータを表示し、可視化する方法は、このドキュメントでは扱いません。詳細は、Kibana ドキュメント を参照してください。

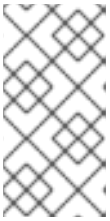
### 前提条件

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。
- Kibana インデックスパターンが存在する。
- Kibana で **infra** および **audit** インデックスを表示するには、ユーザーには **cluster-admin** ロール、 **cluster-reader** ロール、または両方のロールが必要です。デフォルトの **kubeadmin** ユーザーには、これらのインデックスを表示するための適切なパーミッションがあります。
  **default**、**kube-** および **openshift-** プロジェクトで Pod およびログを表示できる場合に、これらのインデックスにアクセスできるはずです。以下のコマンドを使用して、現在のユーザーが適切なパーミッションを持っているかどうかを確認できます。

```
$ oc auth can-i get pods/log -n <project>
```

**出力例**

```
yes
```



**注記**

監査ログは、デフォルトでは内部 OpenShift Container Platform Elasticsearch インスタンスに保存されません。Kibana で監査ログを表示するには、ログ転送 API を使用して監査ログの **default** 出力を使用するパイプラインを設定する必要があります。

### 手順

Kibana でログを表示するには、以下を実行します。

1. OpenShift Container Platform コンソールで、Application Launcher  をクリックし、**Logging** を選択します。

2. OpenShift Container Platform コンソールにログインするために使用するものと同じ認証情報を使用してログインします。
Kibana インターフェイスが起動します。

3. Kibana で **Discover** をクリックします。

4. 左上隅のドロップダウンメニューから作成したインデックスパターン (**app**、**audit**、または **infra**) を選択します。
ログデータは、タイムスタンプ付きのドキュメントとして表示されます。

5. タイムスタンプ付きのドキュメントの 1 つをデプロイメントします。

6. **JSON** タブをクリックし、ドキュメントのログエントリーを表示します。

**例10.1 Kibana のインフラストラクチャーログエントリーのサンプル**


```
{
  "_index": "infra-000001",
  "_type": "_doc",
  "_id": "YmJmYTBlNDkZTRmLTliMGQtMjE3NmFiOGUyOWM3",
  "_version": 1,
  "_score": null,
  "_source": {
    "docker": {
      "container_id": "f85fa55bbef7bb783f041066be1e7c267a6b88c4603dfce213e32c1"
    },
    "kubernetes": {
      "container_name": "registry-server",
      "namespace_name": "openshift-marketplace",
      "pod_name": "redhat-marketplace-n64gc",
      "container_image": "registry.redhat.io/redhat/redhat-marketplace-index:v4.7",
      "container_image_id": "registry.redhat.io/redhat/redhat-marketplace-
index@sha256:65fc0c45aabb95809e376feb065771ecda9e5e59cc8b3024c4545c168f",
      "pod_id": "8f594ea2-c866-4b5c-a1c8-a50756704b2a",
      "host": "ip-10-0-182-28.us-east-2.compute.internal",
      "master_url": "https://kubernetes.default.svc",
      "namespace_id": "3abab127-7669-4eb3-b9ef-44c04ad68d38",
      "namespace_labels": {
        "openshift_io/cluster-monitoring": "true"
      },
      "flat_labels": [
        "catalogsource_operators_coreos_com/update=redhat-marketplace"
      ]
    },
    "message": "time=\"2020-09-23T20:47:03Z\" level=info msg=\"serving registry\"
database=/database/index.db port=50051",
    "level": "unknown",
    "hostname": "ip-10-0-182-28.internal",
    "pipeline_metadata": {
      "collector": {
```

```
          "ipaddr4": "10.0.182.28",
          "inputname": "fluent-plugin-systemd",
          "name": "fluentd",
          "received_at": "2020-09-23T20:47:15.007583+00:00",
          "version": "1.7.4 1.6.0"
        }
      },
      "@timestamp": "2020-09-23T20:47:03.422465+00:00",
      "viaq_msg_id": "YmJmYTBlNDktMDMGQtMjE3NmFiOGUyOWM3",
      "openshift": {
        "labels": {
          "logging": "infra"
        }
      }
    },
    "fields": {
      "@timestamp": [
        "2020-09-23T20:47:03.422Z"
      ],
      "pipeline_metadata.collector.received_at": [
        "2020-09-23T20:47:15.007Z"
      ]
    },
    "sort": [
      1600894023422
    ]
  }
```

# 第11章 ログの外部のサードパーティーロギングシステムへの転送

デフォルトで、ロギングシステムは **ClusterLogging** カスタムリソースに定義されるデフォルトの内部ログストアにコンテナーおよびインフラストラクチャーログを送信します。ただし、監査ログは内部ストアに送信しません。セキュアなストレージを提供しないためです。このデフォルト設定が要件を満たす場合には、クラスターログフォワーダーを設定する必要はありません。

ログを他のログアグリゲーターに送信するには、OpenShift Container Platform クラスターログフォワーダーを使用します。この API を使用すると、コンテナー、インフラストラクチャーおよび監査ログをクラスター内外の特定のエンドポイントに送信できます。さらに、異なるタイプのログをさまざまなシステムに送信できるため、さまざまなユーザーがそれぞれのタイプにアクセスできます。また、Transport Layer Security (TLS) のサポートを有効にして、組織の要件に合わせてログを安全に送信することもできます。

**注記**

監査ログをデフォルトの内部 Elasticsearch ログストアに送信するには、監査ログのログストアへの転送 で説明されているようにクラスターログフォワーダーを使用します。

ログを外部に転送する場合、ログサブシステムは Fluentd 設定マップを作成または変更して、目的のプロトコルを使用してログを送信します。外部ログアグリゲーターでプロトコルを設定する必要があります。

## 11.1. ログのサードパーティーシステムへの転送

OpenShift Container Platform クラスターの内外の特定のエンドポイントにログを送信するには、**ClusterLogForwarder** カスタムリソース (CR) で **出力**と**パイプライン**の組み合わせを指定します。**入力**を使用して、特定のプロジェクトに関連付けられたアプリケーションログをエンドポイントに転送することもできます。認証は Kubernetes**シークレット**オブジェクトによって提供されます。

**output**

定義するログデータの宛先、またはログの送信先です。出力は以下のいずれかのタイプになります。

- **elasticsearch**。外部 Elasticsearch インスタンス。**elasticsearch** 出力では、TLS 接続を使用できます。
- **fluentdForward**。Fluentd をサポートする外部ログ集計ソリューション。このオプションは Fluentd **forward** プロトコルを使用します。**fluentForward** 出力は TCP または TLS 接続を使用でき、シークレットに **shared_key** フィールドを指定して共有キーの認証をサポートします。共有キーの認証は、TLS の有無に関係なく使用できます。
- **syslog**。syslog RFC3164 または RFC5424 プロトコルをサポートする外部ログ集計ソリューション。**syslog** 出力は、UDP、TCP、または TLS 接続を使用できます。
- **cloudwatch**.Amazon Web Services (AWS) がホストするモニタリングおよびログストレージサービスである Amazon CloudWatch。
- **loki**。Loki: 水平方向にスケーラブルで可用性の高いマルチテナントログ集計システム。
- **kafka**。Kafka ブローカー。**kafka** 出力は TCP または TLS 接続を使用できます。
- **default**。内部 OpenShift Container Platform Elasticsearch インスタンス。デフォルトの出力を設定する必要はありません。**default** 出力を設定する場合、**default** 出力は Red Hat OpenShift Logging Operator 用に予約されるため、エラーメッセージが送信されます。

パイプライン

1つのログタイプから1つまたは複数の出力への単純なルーティング、または送信するログを定義します。ログタイプは以下のいずれかになります。

- **application**。クラスターで実行される、インフラストラクチャーコンテナーアプリケーションを除くユーザーアプリケーションによって生成されるコンテナーログ。
- **infrastructure**。**openshift***、**kube***、または **default** プロジェクトで実行される Pod のコンテナーログおよびノードファイルシステムから取得されるジャーナルログ。
- **audit**ノード監査システム、**auditd**、Kubernetes API サーバー、OpenShift API サーバー、および OVN ネットワークで生成される監査ログ。

パイプラインで **key:value** ペアを使用すると、アウトバウンドログメッセージにラベルを追加できます。たとえば、他のデータセンターに転送されるメッセージにラベルを追加したり、タイプ別にログにラベルを付けたりできます。オブジェクトに追加されるラベルもログメッセージと共に転送されます。

input

特定のプロジェクトに関連付けられるアプリケーションログをパイプラインに転送します。
パイプラインでは、**inputRef** パラメーターを使用して転送するログタイプと、**outputRef** パラメーターを使用してログを転送する場所を定義します。

Secret

ユーザー認証情報などの機密データを含む **Key:Value マップ**。

次の点に注意してください。

- **ClusterLogForwarder** CR オブジェクトが存在する場合に、**default** 出力のあるパイプラインがない限り、ログはデフォルト Elasticsearch インスタンスに転送されません。
- デフォルトで、ロギングシステムは **ClusterLogging** カスタムリソースに定義されるデフォルトの内部 Elasticsearch ログストアにコンテナーおよびインフラストラクチャーログを送信します。ただし、監査ログは内部ストアに送信しません。セキュアなストレージを提供しないためです。このデフォルト設定が要件を満たす場合は、ログ転送 API を設定する必要はありません。
- ログタイプのパイプラインを定義しない場合、未定義タイプのログはドロップされます。たとえば、**application** および **audit** タイプのパイプラインを指定するものの、**infrastructure** タイプのパイプラインを指定しないと、**infrastructure** ログはドロップされます。
- **ClusterLogForwarder** カスタムリソース (CR) で出力の複数のタイプを使用し、ログを複数の異なるプロトコルをサポートするサーバーに送信できます。
- 内部 OpenShift Container Platform Elasticsearch インスタンスは、監査ログのセキュアなストレージを提供しません。監査ログを転送するシステムが組織および政府の規制に準拠しており、適切にセキュリティーが保護されていることを確認することが推奨されています。ロギングサブシステムはこれらの規制に準拠していません。

以下の例では、監査ログをセキュアな外部 Elasticsearch インスタンスに転送し、インフラストラクチャーログをセキュアでない外部 Elasticsearch インスタンスに、アプリケーションログを Kafka ブローカーに転送し、アプリケーションログを **my-apps-logs** プロジェクトから内部 Elasticsearch インスタンスに転送します。

**ログ転送の出力とパイプラインのサンプル**

```
apiVersion: "logging.openshift.io/v1"
kind: ClusterLogForwarder
metadata:
  name: instance (1)
  namespace: openshift-logging (2)
spec:
  outputs:
   - name: elasticsearch-secure (3)
     type: "elasticsearch"
     url: https://elasticsearch.secure.com:9200
     secret:
       name: elasticsearch
   - name: elasticsearch-insecure (4)
     type: "elasticsearch"
     url: http://elasticsearch.insecure.com:9200
   - name: kafka-app (5)
     type: "kafka"
     url: tls://kafka.secure.com:9093/app-topic
  inputs: (6)
   - name: my-app-logs
     application:
       namespaces:
       - my-project
  pipelines:
   - name: audit-logs (7)
     inputRefs:
      - audit
     outputRefs:
      - elasticsearch-secure
      - default
     parse: json (8)
     labels:
       secure: "true" (9)
       datacenter: "east"
   - name: infrastructure-logs (10)
     inputRefs:
      - infrastructure
     outputRefs:
      - elasticsearch-insecure
     labels:
       datacenter: "west"
   - name: my-app (11)
     inputRefs:
      - my-app-logs
     outputRefs:
      - default
   - inputRefs: (12)
      - application
     outputRefs:
      - kafka-app
     labels:
       datacenter: "south"
```

(1) **ClusterLogForwarder** CR の名前は **instance** である必要があります。

2 **ClusterLogForwarder** CR の namespace は **openshift-logging** である必要があります。

3 シークレットとセキュアな URL を使用したセキュアな Elasticsearch 出力の設定。

- 出力を記述する名前。
- 出力のタイプ: **elasticsearch**。
- 接頭辞を含む、有効な絶対 URL としての Elasticsearch インスタンスのセキュアな URL およびポート。
- TLS 通信のエンドポイントで必要なシークレット。シークレットは **openshift-logging** プロジェクトに存在する必要があります。

4 非セキュアな Elasticsearch 出力の設定:

- 出力を記述する名前。
- 出力のタイプ: **elasticsearch**。
- 接頭辞を含む、有効な絶対 URL として Elasticsearch インスタンスのセキュアではない URL およびポート。

5 セキュアな URL を介したクライアント認証 TLS 通信を使用した Kafka 出力の設定

- 出力を記述する名前。
- 出力のタイプ: **kafka**。
- Kafka ブローカーの URL およびポートを、接頭辞を含む有効な絶対 URL として指定します。

6 **my-project** namespace からアプリケーションログをフィルターするための入力の設定。

7 監査ログをセキュアな外部 Elasticsearch インスタンスに送信するためのパイプラインの設定。

- パイプラインを説明する名前。
- **inputRefs** はログタイプです (例: **audit**)。
- **outputRefs** は使用する出力の名前です。この例では、**elasticsearch-secure** はセキュアな Elasticsearch インスタンスに転送され、**default** は内部 Elasticsearch インスタンスに転送されます。
- オプション: ログに追加する複数のラベル。

8 オプション: 構造化された JSON ログエントリーを **structured** フィールドの JSON オブジェクトとして転送するかどうかを指定します。ログエントリーに有効な構造化された JSON が含まれる必要があります。そうでない場合は、OpenShift Logging は **構造化** フィールドを削除し、代わりにログエントリーをデフォルトのインデックス **app-00000x** に送信します。

9 オプション: 文字列。ログに追加する 1 つまたは複数のラベル。"true" などの引用値は、ブール値としてではなく、文字列値として認識されるようにします。

10 インフラストラクチャーログをセキュアでない外部 Elasticsearch インスタンスに送信するためのパイプラインの設定。

11 **my-project** プロジェクトから内部 Elasticsearch インスタンスにログを送信するためのパイプライ

ンの設定。

- パイプラインを説明する名前。
- **inputRefs** は特定の入力 **my-app-logs** です。
- **outputRefs** は **default** です。
- オプション: 文字列。ログに追加する 1 つまたは複数のラベル。

12 パイプライン名がない場合にログを Kafka ブローカーに送信するためのパイプラインの設定。

- **inputRefs** はログタイプです (例: **application**)。
- **outputRefs** は使用する出力の名前です。
- オプション: 文字列。ログに追加する 1 つまたは複数のラベル。

**外部ログアグリゲーターが利用できない場合の Fluentd のログの処理**

外部ロギングアグリゲーターが利用できず、ログを受信できない場合、Fluentd は継続してログを収集し、それらをバッファーに保存します。ログアグリゲーターが利用可能になると、バッファーされたログを含む、ログの転送が再開されます。バッファーが完全に一杯になると、Fluentd はログの収集を停止します。OpenShift Container Platform はログをローテーションし、それらを削除します。バッファーサイズを調整したり、永続ボリューム要求 (PVC) を Fluentd デーモンセットまたは Pod に追加したりすることはできません。

**サポート対象の認証キー**

ここでは、一般的なキータイプを示します。出力タイプは追加の特殊キーをサポートするものもあります。出力固有の設定フィールにまとめられています。すべての秘密鍵はオプションです。関連するキーを設定して、必要なセキュリティー機能を有効にします。キーやシークレット、サービスアカウント、ポートのオープン、またはグローバルプロキシー設定など、外部の宛先で必要となる可能性のある追加設定を作成し、維持する必要があります。OpenShift Logging は、認証の組み合わせ間の不一致を検証しません。

Transport Layer Security (TLS)

シークレットなしで TLSURL( 'http://...'または 'ssl://...') を使用すると、基本的な TLS サーバー側認証が有効になります。シークレットを含め、次のオプションフィールドを設定して、追加の TLS 機能を有効にします。

- **tls.crt**:(文字列) クライアント証明書を含むファイル名。相互認証を有効にします。**tls.key** が必要です。
- **tls.key**:(文字列) クライアント証明書のロックを解除するための秘密鍵を含むファイル名。**tls.crt** が必要です。
- **passphrase**:(文字列) エンコードされた TLS 秘密鍵をデコードするためのパスフレーズ。**tls.key** が必要です。
- **ca-bundle.crt**:(文字列) サーバー認証用のカスタマー CA のファイル名。

**ユーザー名およびパスワード**

- **username**:(文字列) 認証ユーザー名。**パスワード** が必要です。
- **password**:(文字列) 認証パスワード。**ユーザー名** が必要です。

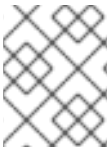
Simple Authentication Security Layer (SASL)

- **sasl.enable**(boolean)SASL を明示的に有効または無効にします。ない場合は、SASL は、他の **sasl.** キーが設定されている場合に自動的に有効になります。
- **sasl.mechanisms**:(配列) 許可された SASL メカニズム名のリスト。欠落しているか空の場合は、システムのデフォルトが使用されます。
- **sasl.allow-insecure**:(ブール値) クリアテキストのパスワードを送信するメカニズムを許可します。デフォルトは false です。

### 11.1.1. シークレットの作成

次のコマンドを使用して、証明書とキーファイルを含むディレクトリーにシークレットを作成できます。

```
$ oc create secret generic -n openshift-logging <my-secret> \
 --from-file=tls.key=<your_key_file>
 --from-file=tls.crt=<your_crt_file>
 --from-file=ca-bundle.crt=<your_bundle_file>
 --from-literal=username=<your_username>
 --from-literal=password=<your_password>
```

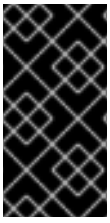


**注記**

最適な結果を得るには一般的または不透明なシークレットを使用することを推奨します。

## 11.2. 同じ POD 内のコンテナーから別のインデックスへの JSON ログの転送

構造化ログを、同じ Pod 内の異なるコンテナーから別のインデックスに転送できます。この機能を使用するには、複数コンテナーのサポートを使用してパイプラインを設定し、Pod にアノテーションを付ける必要があります。ログは接頭辞が **app-** のインデックスに書き込まれます。これに対応するために、エイリアスを使用して Elasticsearch を設定することを推奨します。



**重要**

ログの JSON 形式は、アプリケーションによって異なります。作成するインデックスが多すぎるとパフォーマンスに影響するため、この機能の使用は、互換性のない JSON 形式のログのインデックスの作成に限定してください。クエリーを使用して、さまざまな namespace または互換性のある JSON 形式のアプリケーションからログを分離します。

**前提条件**

- Red Hat OpenShift のロギングサブシステム: 5.5

**手順**

1. **ClusterLogForwarder** CR オブジェクトを定義する YAML ファイルを作成または編集します。

```
apiVersion: "logging.openshift.io/v1"
kind: ClusterLogForwarder
metadata:
```

```
  name: instance
  namespace: openshift-logging
spec:
  outputDefaults:
    elasticsearch:
      enableStructuredContainerLogs: true (1)
  pipelines:
  - inputRefs:
    - application
    name: application-logs
    outputRefs:
    - default
    parse: json
```

(1) マルチコンテナー出力を有効にします。

2. **Pod** CR オブジェクトを定義する YAML ファイルを作成または編集します。

```
  apiVersion: v1
  kind: Pod
  metadata:
    annotations:
      containerType.logging.openshift.io/heavy: heavy (1)
      containerType.logging.openshift.io/low: low
  spec:
    containers:
    - name: heavy (2)
      image: heavyimage
    - name: low
      image: lowimage
```

(1) 形式: **containerType.logging.openshift.io/<container-name>: <index>**

(2) アノテーション名はコンテナー名と同じでなければなりません。

**警告**

この設定により、クラスター上のシャードの数が大幅に増加する可能性があります。

**関連情報**

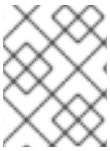
Kubernetes Annotations

## 11.3. OPENSHIFT LOGGING 5.1 でサポートされるログデータ出力タイプ

Red Hat OpenShift Logging 5.1 は、ログデータをターゲットログコレクターに送信するために以下の出力タイプおよびプロトコルを提供します。

Red Hat は、以下の表に記載されているそれぞれの組み合わせをテストします。ただし、これらのプロトコルを取り込むより広範囲のターゲットログコレクターにログデータを送信できるはずです。

| 出力のタイプ | プロトコル | テストで使用 |
| --- | --- | --- |
| elasticsearch | elasticsearch | Elasticsearch 6.8.1<br>Elasticsearch 6.8.4<br>Elasticsearch 7.12.2 |
| fluentdForward | fluentd forward v1 | fluentd 1.7.4<br>logstash 7.10.1 |
| kafka | kafka 0.11 | kafka 2.4.1<br>kafka 2.7.0 |
| syslog | RFC-3164、RFC-5424 | rsyslog-8.39.0 |



**注記**

以前のバージョンでは、syslog 出力は RFC-3164 でのみサポートされました。現在の syslog 出力では RFC-5424 のサポートを追加します。

## 11.4. OPENSHIFT LOGGING 5.2 でサポートされるログデータ出力タイプ

Red Hat OpenShift Logging バージョン 5.2 は、ログデータをターゲットログコレクターに送信するために以下の出力タイプおよびプロトコルを提供します。

Red Hat は、以下の表に記載されているそれぞれの組み合わせをテストします。ただし、これらのプロトコルを取り込むより広範囲のターゲットログコレクターにログデータを送信できるはずです。

| 出力のタイプ | プロトコル | テストで使用 |
| --- | --- | --- |
| Amazon CloudWatch | REST over HTTPS | Amazon CloudWatch の現行バージョン |
| elasticsearch | elasticsearch | Elasticsearch 6.8.1<br>Elasticsearch 6.8.4<br>Elasticsearch 7.12.2 |
| fluentdForward | fluentd forward v1 | fluentd 1.7.4<br>logstash 7.10.1 |
| Loki | REST over HTTP and HTTPS | OCP および Grafana ラボにデプロイされた loki 2.3.0 |

| 出力のタイプ | プロトコル | テストで使用 |
| --- | --- | --- |
| kafka | kafka 0.11 | kafka 2.4.1<br>kafka 2.7.0 |
| syslog | RFC-3164、RFC-5424 | rsyslog-8.39.0 |

## 11.5. OPENSHIFT LOGGING 5.3 でサポートされるログデータ出力タイプ

Red Hat OpenShift Logging 5.3 は、ログデータをターゲットログコレクターに送信するために以下の出力タイプおよびプロトコルを提供します。

Red Hat は、以下の表に記載されているそれぞれの組み合わせをテストします。ただし、これらのプロトコルを取り込むより広範囲のターゲットログコレクターにログデータを送信できるはずです。

| 出力のタイプ | プロトコル | テストで使用 |
| --- | --- | --- |
| Amazon CloudWatch | REST over HTTPS | Amazon CloudWatch の現行バージョン |
| elasticsearch | elasticsearch | Elasticsearch 7.10.1 |
| fluentdForward | fluentd forward v1 | fluentd 1.7.4<br>logstash 7.10.1 |
| Loki | REST over HTTP and HTTPS | OCP にデプロイされた Loki 2.2.1 |
| kafka | kafka 0.11 | kafka 2.7.0 |
| syslog | RFC-3164、RFC-5424 | rsyslog-8.39.0 |

## 11.6. OPENSHIFT LOGGING 5.4 でサポートされるログデータ出力タイプ

Red Hat OpenShift Logging 5.4 は、ログデータをターゲットログコレクターに送信するために以下の出力タイプおよびプロトコルを提供します。

Red Hat は、以下の表に記載されているそれぞれの組み合わせをテストします。ただし、これらのプロトコルを取り込むより広範囲のターゲットログコレクターにログデータを送信できるはずです。

| 出力のタイプ | プロトコル | テストで使用 |
| --- | --- | --- |
| Amazon CloudWatch | REST over HTTPS | Amazon CloudWatch の現行バージョン |
| elasticsearch | elasticsearch | Elasticsearch 7.10.1 |

| 出力のタイプ | プロトコル | テストで使用 |
| --- | --- | --- |
| fluentdForward | fluentd forward v1 | fluentd 1.14.5<br>logstash 7.10.1 |
| Loki | REST over HTTP and HTTPS | OCP にデプロイされた Loki 2.2.1 |
| kafka | kafka 0.11 | kafka 2.7.0 |
| syslog | RFC-3164、RFC-5424 | rsyslog-8.39.0 |

## 11.7. OPENSHIFT LOGGING 5.5 でサポートされるログデータ出力タイプ

Red Hat OpenShift Logging 5.5 は、ログデータをターゲットログコレクターに送信するために以下の出力タイプおよびプロトコルを提供します。

Red Hat は、以下の表に記載されているそれぞれの組み合わせをテストします。ただし、これらのプロトコルを取り込むより広範囲のターゲットログコレクターにログデータを送信できるはずです。

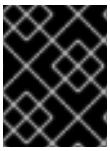
| 出力のタイプ | プロトコル | テストで使用 |
| --- | --- | --- |
| Amazon CloudWatch | REST over HTTPS | Amazon CloudWatch の現行バージョン |
| elasticsearch | elasticsearch | Elasticsearch 7.10.1 |
| fluentdForward | fluentd forward v1 | fluentd 1.14.6<br>logstash 7.10.1 |
| Loki | REST over HTTP and HTTPS | OCP にデプロイされた Loki 2.5.0 |
| kafka | kafka 0.11 | kafka 2.7.0 |
| syslog | RFC-3164、RFC-5424 | rsyslog-8.39.0 |

## 11.8. OPENSHIFT LOGGING 5.6 でサポートされるログデータ出力タイプ

Red Hat OpenShift Logging 5.6 は、ログデータをターゲットログコレクターに送信するために以下の出力タイプおよびプロトコルを提供します。

Red Hat は、以下の表に記載されているそれぞれの組み合わせをテストします。ただし、これらのプロトコルを取り込むより広範囲のターゲットログコレクターにログデータを送信できるはずです。

| 出力のタイプ | プロトコル | テストで使用 |
| --- | --- | --- |
| Amazon CloudWatch | REST over HTTPS | Amazon CloudWatch の現行バージョン |
| elasticsearch | elasticsearch | Elasticsearch 6.8.23<br>Elasticsearch 7.10.1<br>Elasticsearch 8.6.1 |
| fluentdForward | fluentd forward v1 | fluentd 1.14.6<br>logstash 7.10.1 |
| Loki | REST over HTTP and HTTPS | OCP にデプロイされた Loki 2.5.0 |
| kafka | kafka 0.11 | kafka 2.7.0 |
| syslog | RFC-3164、RFC-5424 | rsyslog-8.39.0 |



**重要**

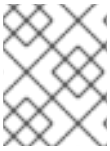
Fluentd は、5.6.2 の時点で Elasticsearch 8 をサポートしていません。Vector は、5.7.0 より前の fluentd/logstash/rsyslog をサポートしていません。

## 11.9. 外部 ELASTICSEARCH インスタンスへのログの送信

オプションで、内部 OpenShift Container Platform Elasticsearch インスタンスに加えて、またはその代わりに外部 Elasticsearch インスタンスにログを転送できます。外部ログアグリゲーターを OpenShift Container Platform からログデータを受信するように設定する必要があります。

外部 Elasticsearch インスタンスへのログ転送を設定するには、そのインスタンスへの出力および出力を使用するパイプラインで **ClusterLogForwarder** カスタムリソース (CR) を作成する必要があります。外部 Elasticsearch 出力では、HTTP(セキュアでない) または HTTPS(セキュアな HTTP) 接続を使用できます。

外部 Elasticsearch インスタンスと内部 Elasticsearch インスタンスの両方にログを転送するには、出力および外部インスタンスへのパイプライン、および **default** 出力を使用してログを内部インスタンスに転送するパイプラインを作成します。**default** 出力を作成する必要はありません。**default** 出力を設定する場合、**default** 出力は Red Hat OpenShift Logging Operator 用に予約されるため、エラーメッセージが送信されます。



**注記**

ログを内部 OpenShift Container Platform Elasticsearch インスタンス ***のみ*** に転送する必要がある場合は、**ClusterLogForwarder** CR を作成する必要はありません。

**前提条件**

- 指定されたプロトコルまたは形式を使用してロギングデータを受信するように設定されたロギングサーバーが必要です。

**手順**

1. **ClusterLogForwarder** CR オブジェクトを定義する YAML ファイルを作成または編集します。

```
apiVersion: "logging.openshift.io/v1"
kind: ClusterLogForwarder
metadata:
  name: instance (1)
  namespace: openshift-logging (2)
spec:
  outputs:
   - name: elasticsearch-insecure (3)
     type: "elasticsearch" (4)
     url: http://elasticsearch.insecure.com:9200 (5)
   - name: elasticsearch-secure
     type: "elasticsearch"
     url: https://elasticsearch.secure.com:9200 (6)
     secret:
       name: es-secret (7)
  pipelines:
   - name: application-logs (8)
     inputRefs: (9)
     - application
     - audit
     outputRefs:
     - elasticsearch-secure (10)
     - default (11)
     parse: json (12)
     labels:
       myLabel: "myValue" (13)
   - name: infrastructure-audit-logs (14)
     inputRefs:
     - infrastructure
     outputRefs:
     - elasticsearch-insecure
     labels:
       logs: "audit-infra"
```

(1) **ClusterLogForwarder** CR の名前は **instance** である必要があります。

(2) **ClusterLogForwarder** CR の namespace は **openshift-logging** である必要があります。

(3) 出力の名前を指定します。

(4) **elasticsearch** タイプを指定します。

(5) 外部 Elasticsearch インスタンスの URL およびポートを有効な絶対 URL として指定します。**http** (セキュアでない) プロトコルまたは **https** (セキュアな HTTP) プロトコルを使用できます。CIDR アノテーションを使用するクラスター全体のプロキシーが有効になっている場合、出力は IP アドレスではなくサーバー名または FQDN である必要があります。

(6) セキュアな接続では、**シークレット** を指定して、認証する **https** または **http** URL を指定できます。

7 **https** 接頭辞の場合は、TLS 通信のエンドポイントに必要なシークレットの名前を指定します。シークレットは **openshift-logging** プロジェクトに存在し、**tls.crt**、**tls.key** および

8 オプション: パイプラインの名前を指定します。

9 パイプラインを使用して転送するログタイプ (**application**、**infrastructure** または **audit**) を指定します。

10 このパイプラインでログを転送する時に使用する出力の名前を指定します。

11 オプション: ログを内部 Elasticsearch インスタンスに送信するために **default** 出力を指定します。

12 オプション: 構造化された JSON ログエントリーを **structured** フィールドの JSON オブジェクトとして転送するかどうかを指定します。ログエントリーに有効な構造化された JSON が含まれる必要があります。そうでない場合は、OpenShift Logging は **構造化** フィールドを削除し、代わりにログエントリーをデフォルトのインデックス **app-00000x** に送信します。

13 オプション: 文字列。ログに追加する 1 つまたは複数のラベル。

14 オプション: サポートされるタイプの他の外部ログアグリゲーターにログを転送するように複数の出力を設定します。

- パイプラインを説明する名前。
- **inputRefs** は、そのパイプラインを使用して転送するログタイプです (**application**、**infrastructure**、または **audit**)。
- **outputRefs** は使用する出力の名前です。
- オプション: 文字列。ログに追加する 1 つまたは複数のラベル。

2. CR オブジェクトを作成します。

```
$ oc create -f <file-name>.yaml
```

### 例: ユーザー名とパスワードを含むシークレットの設定

ユーザー名とパスワードを含むシークレットを使用して、外部 Elasticsearch インスタンスへのセキュアな接続を認証できます。

たとえば、サードパーティーが Elasticsearch インスタンスを操作するため、相互 TLS (mTLS) キーを使用できない場合に、HTTP または HTTPS を使用してユーザー名とパスワードを含むシークレットを設定できます。

1. 以下の例のような **Secret** YAML ファイルを作成します。**username** および **password** フィールドに base64 でエンコードされた値を使用します。シークレットタイプはデフォルトで不透明です。

```
apiVersion: v1
kind: Secret
metadata:
  name: openshift-test-secret
```

```
data:
  username: <username>
  password: <password>
```

2. シークレットを作成します。

```
$ oc create secret -n openshift-logging openshift-test-secret.yaml
```

3. **ClusterLogForwarder** CR にシークレットの名前を指定します。

```
kind: ClusterLogForwarder
metadata:
  name: instance
  namespace: openshift-logging
spec:
  outputs:
   - name: elasticsearch
     type: "elasticsearch"
     url: https://elasticsearch.secure.com:9200
     secret:
        name: openshift-test-secret
```

**注記**

**url** フィールドの値では、接頭辞は **http** または **https** になります。

4. CR オブジェクトを作成します。

```
$ oc create -f <file-name>.yaml
```

## 11.10. FLUENTD 転送プロトコルを使用したログの転送

Fluentd **forward** プロトコルを使用して、デフォルトの Elasticsearch ログストアの代わり、またはこれに加えてプロトコルを受け入れるように設定された外部ログアグリゲーターにログのコピーを送信できます。外部ログアグリゲーターを OpenShift Container Platform からログを受信するように設定する必要があります。

**forward** プロトコルを使用してログ転送を設定するには、Fluentd サーバーに対する 1 つ以上の出力およびそれらの出力を使用するパイプラインと共に **ClusterLogForwarder** カスタムリース (CR) を作成します。Fluentd の出力は TCP(セキュアでない) または TLS(セキュアな TCP) 接続を使用できます。

**前提条件**

- 指定されたプロトコルまたは形式を使用してロギングデータを受信するように設定されたロギングサーバーが必要です。

**手順**

1. **ClusterLogForwarder** CR オブジェクトを定義する YAML ファイルを作成または編集します。

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogForwarder
metadata:
```

```
  name: instance 1
  namespace: openshift-logging 2
spec:
  outputs:
   - name: fluentd-server-secure 3
     type: fluentdForward 4
     url: 'tls://fluentdserver.security.example.com:24224' 5
     secret: 6
       name: fluentd-secret
   - name: fluentd-server-insecure
     type: fluentdForward
     url: 'tcp://fluentdserver.home.example.com:24224'
  pipelines:
   - name: forward-to-fluentd-secure 7
     inputRefs: 8
     - application
     - audit
     outputRefs:
     - fluentd-server-secure 9
     - default 10
     parse: json 11
     labels:
       clusterId: "C1234" 12
   - name: forward-to-fluentd-insecure 13
     inputRefs:
     - infrastructure
     outputRefs:
     - fluentd-server-insecure
     labels:
       clusterId: "C1234"
```

1. **ClusterLogForwarder** CR の名前は **instance** である必要があります。
2. **ClusterLogForwarder** CR の namespace は **openshift-logging** である必要があります。
3. 出力の名前を指定します。
4. **fluentdForward** タイプを指定します。
5. 外部 Fluentd インスタンスの URL およびポートを有効な絶対 URL として指定します。**tcp** (セキュアでない) プロトコルまたは **tls** (セキュアな TCP) プロトコルを使用できます。CIDR アノテーションを使用するクラスター全体のプロキシーが有効になっている場合、出力は IP アドレスではなくサーバー名または FQDN である必要があります。
6. **tls** 接頭辞を使用する場合は、TLS 通信のエンドポイントに必要なシークレットの名前を指定する必要があります。シークレットは **openshift-logging** プロジェクトに存在し、**tls.crt**、**tls.key** および **ca-bundle.crt** のキーが含まれる必要があります。これらは、それぞれが表す証明書を参照します。それ以外の場合は、http および https 接頭辞の場合は、ユーザー名とパスワードを含むシークレットを指定できます。詳細は、Example: Setting secret that contains a username and password.を参照してください。
7. オプション: パイプラインの名前を指定します。
8. パイプラインを使用して転送するログタイプ (**application**、**infrastructure** または **audit**) を指定します。

9 このパイプラインでログを転送する時に使用する出力の名前を指定します。

10 オプション: ログを内部 Elasticsearch インスタンスに転送するために **default** 出力を指定します。

11 オプション: 構造化された JSON ログエントリーを **structured** フィールドの JSON オブジェクトとして転送するかどうかを指定します。ログエントリーに有効な構造化された JSON が含まれる必要があります。そうでない場合は、OpenShift Logging は **構造化** フィールドを削除し、代わりにログエントリーをデフォルトのインデックス **app-00000x** に送信します。

12 オプション: 文字列。ログに追加する 1 つまたは複数のラベル。

13 オプション: サポートされるタイプの他の外部ログアグリゲーターにログを転送するように複数の出力を設定します。

- パイプラインを説明する名前。
- **inputRefs** は、そのパイプラインを使用して転送するログタイプです (**application**、**infrastructure**、または **audit**)。
- **outputRefs** は使用する出力の名前です。
- オプション: 文字列。ログに追加する 1 つまたは複数のラベル。

2. CR オブジェクトを作成します。

```
$ oc create -f <file-name>.yaml
```

### 11.10.1. Logstash が fluentd からデータを取り込むためのナノ秒精度の有効化

Logstash が fluentd からログデータを取り込むには、Logstash 設定ファイルでナノ秒精度を有効にする必要があります。

**手順**

- Logstash 設定ファイルで、**nanosecond_precision** を **true** に設定します。

**Logstash 設定ファイルの例**

```
input { tcp { codec => fluent { nanosecond_precision => true } port => 24114 } }
filter { }
output { stdout { codec => rubydebug } }
```

## 11.11. SYSLOG プロトコルを使用したログの転送

**syslog** RFC3164 または RFC5424 プロトコルを使用して、デフォルトの Elasticsearch ログストアの代わり、またはこれに加えてプロトコルを受け入れるように設定された外部ログアグリゲーターにログのコピーを送信できます。syslog サーバーなど、外部ログアグリゲーターを OpenShift Container Platform からログを受信するように設定する必要があります。

**syslog** プロトコルを使用してログ転送を設定するには、syslog サーバーに対する 1 つ以上の出力およびそれらの出力を使用するパイプラインと共に **ClusterLogForwarder** カスタムリース (CR) を作成します。syslog 出力では、UDP、TCP、または TLS 接続を使用できます。

**前提条件**

- 指定されたプロトコルまたは形式を使用してロギングデータを受信するように設定されたロギングサーバーが必要です。

**手順**

1. **ClusterLogForwarder** CR オブジェクトを定義する YAML ファイルを作成または編集します。

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogForwarder
metadata:
  name: instance 1
  namespace: openshift-logging 2
spec:
  outputs:
   - name: rsyslog-east 3
     type: syslog 4
     syslog: 5
       facility: local0
       rfc: RFC3164
       payloadKey: message
       severity: informational
     url: 'tls://rsyslogserver.east.example.com:514' 6
     secret: 7
        name: syslog-secret
   - name: rsyslog-west
     type: syslog
     syslog:
      appName: myapp
      facility: user
      msgID: mymsg
      procID: myproc
      rfc: RFC5424
      severity: debug
     url: 'udp://rsyslogserver.west.example.com:514'
  pipelines:
   - name: syslog-east 8
     inputRefs: 9
     - audit
     - application
     outputRefs: 10
     - rsyslog-east
     - default 11
     parse: json 12
     labels:
       secure: "true" 13
       syslog: "east"
   - name: syslog-west 14
     inputRefs:
     - infrastructure
     outputRefs:
     - rsyslog-west
```

```
    - default
    labels:
      syslog: "west"
```

1. **ClusterLogForwarder** CR の名前は **instance** である必要があります。
2. **ClusterLogForwarder** CR の namespace は **openshift-logging** である必要があります。
3. 出力の名前を指定します。
4. **syslog** タイプを指定します。
5. オプション: 以下にリスト表示されている syslog パラメーターを指定します。
6. 外部 syslog インスタンスの URL およびポートを指定します。**udp** (セキュアでない)、**tcp** (セキュアでない) プロトコル、または **tls** (セキュアな TCP) プロトコルを使用できます。CIDR アノテーションを使用するクラスター全体のプロキシーが有効になっている場合、出力は IP アドレスではなくサーバー名または FQDN である必要があります。
7. **tls** 接頭辞を使用する場合は、TLS 通信のエンドポイントに必要なシークレットの名前を指定する必要があります。シークレットは **openshift-logging** プロジェクトに存在し、**tls.crt**、**tls.key** および **ca-bundle.crt** のキーが含まれる必要があります。これらは、それぞれが表す証明書を参照します。
8. オプション: パイプラインの名前を指定します。
9. パイプラインを使用して転送するログタイプ (**application**、**infrastructure** または **audit**) を指定します。
10. このパイプラインでログを転送する時に使用する出力の名前を指定します。
11. オプション: ログを内部 Elasticsearch インスタンスに転送するために **default** 出力を指定します。
12. オプション: 構造化された JSON ログエントリーを **structured** フィールドの JSON オブジェクトとして転送するかどうかを指定します。ログエントリーに有効な構造化された JSON が含まれる必要があります。そうでない場合は、OpenShift Logging は **構造化** フィールドを削除し、代わりにログエントリーをデフォルトのインデックス **app-00000x** に送信します。
13. オプション: 文字列。ログに追加する 1 つまたは複数のラベル。"true" などの引用値は、ブール値としてではなく、文字列値として認識されるようにします。
14. オプション: サポートされるタイプの他の外部ログアグリゲーターにログを転送するように複数の出力を設定します。
    - パイプラインを説明する名前。
    - **inputRefs** は、そのパイプラインを使用して転送するログタイプです (**application**、**infrastructure**、または **audit**)。
    - **outputRefs** は使用する出力の名前です。
    - オプション: 文字列。ログに追加する 1 つまたは複数のラベル。

2. CR オブジェクトを作成します。

```
$ oc create -f <file-name>.yaml
```

### 11.11.1. メッセージ出力へのログソース情報の追加

**AddLogSource** フィールドを **ClusterLogForwarder** カスタムリソース (CR) に追加することで、**namespace_name**、**pod_name**、および **container_name** 要素をレコードの **メッセージ** フィールドに追加できます。

```
spec:
  outputs:
  - name: syslogout
    syslog:
      addLogSource: true
      facility: user
      payloadKey: message
      rfc: RFC3164
      severity: debug
      tag: mytag
    type: syslog
    url: tls://syslog-receiver.openshift-logging.svc:24224
  pipelines:
  - inputRefs:
    - application
    name: test-app
    outputRefs:
    - syslogout
```

**注記**

この設定は、RFC3164 と RFC5424 の両方と互換性があります。

**AddLogSource を使用しない場合の syslog メッセージ出力の例**

```
<15>1 2020-11-15T17:06:14+00:00 fluentd-9hkb4 mytag - - -  {"msgcontent"=>"Message Contents", "timestamp"=>"2020-11-15 17:06:09", "tag_key"=>"rec_tag", "index"=>56}
```

**AddLogSource を使用した syslog メッセージ出力の例**

```
<15>1 2020-11-16T10:49:37+00:00 crc-j55b9-master-0 mytag - - -  namespace_name=clo-test-6327,pod_name=log-generator-ff9746c49-qxm7l,container_name=log-generator,message={"msgcontent":"My life is my message", "timestamp":"2020-11-16 10:49:36", "tag_key":"rec_tag", "index":76}
```

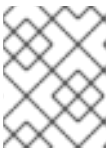
### 11.11.2. syslog パラメーター

**syslog** 出力には、以下を設定できます。詳細は、syslog の RFC3164 または RFC5424 RFC を参照してください。

- facility: syslog ファシリティー。値には 10 進数の整数または大文字と小文字を区別しないキーワードを使用できます。
  - カーネルメッセージの場合は、**0** または **kern**

- ユーザーレベルのメッセージの場合は、**1** または **user**。デフォルトです。
- メールシステムの場合は、**2** または **mail**
- システムデーモンの場合は、**3** または **daemon**
- セキュリティー/認証メッセージの場合は、**4** または **auth**
- syslogd によって内部に生成されるメッセージの場合は、**5** または **syslog**
- ラインプリンターサブシステムの場合は、**6** または **lpr**
- ネットワーク news サブシステムの場合は、**7** または **news**
- UUCP サブシステムの場合は、**8** または **uucp**
- クロックデーモンの場合は、**9** または **cron**
- セキュリティー認証メッセージの場合は、**10** または **authpriv**
- FTP デーモンの場合は、**11** または **ftp**
- NTP サブシステムの場合は、**12** または **ntp**
- syslog 監査ログの場合は、**13** または **security**
- syslog アラートログの場合は、**14** または **console**
- スケジューリングデーモンの場合は、**15** または **solaris-cron**
- ローカルに使用される facility の場合は、**16–23** または **local0** – **local7**

- オプション: **payloadKey**: syslog メッセージのペイロードとして使用するレコードフィールド。



**注記**

**payloadKey** パラメーターを設定すると、他のパラメーターが syslog に転送されなくなります。

- rfc: syslog を使用してログを送信するために使用される RFC。デフォルトは RFC5424 です。
- severity: 送信 syslog レコードに設定される syslog の重大度。値には 10 進数の整数または大文字と小文字を区別しないキーワードを使用できます。
  - システムが使用不可であることを示すメッセージの場合は、**0** または **Emergency**
  - 即時にアクションを実行する必要があることを示すメッセージの場合は、**1** または **Alert**
  - 重大な状態を示すメッセージの場合は、**2** または **Critical**
  - エラーの状態を示すメッセージの場合は、**3** または **Error**
  - 警告状態を示すメッセージの場合は、**4** または **Warning**
  - 正常であるが重要な状態を示すメッセージの場合は、**5** または **Notice**
  - 情報を提供するメッセージの場合は、**6** または **Informational**

- デバッグレベルのメッセージを示唆するメッセージの場合は、**7** または **Debug**。デフォルトです。

- tag: タグは、syslog メッセージでタグとして使用するレコードフィールドを指定します。
- trimPrefix: 指定された接頭辞をタグから削除します。

### 11.11.3. 追加の RFC5424 syslog パラメーター

以下のパラメーターは RFC5424 に適用されます。

- appName: APP-NAME は、ログを送信したアプリケーションを識別するフリーテキストの文字列です。**RFC5424** に対して指定する必要があります。
- msgID: MSGID は、メッセージのタイプを識別するフリーテキスト文字列です。**RFC5424** に対して指定する必要があります。
- procID: PROCID はフリーテキスト文字列です。値が変更される場合は、syslog レポートが中断していることを示します。**RFC5424** に対して指定する必要があります。

## 11.12. ログの AMAZON CLOUDWATCH への転送

Amazon Web Services (AWS) がホストするモニタリングおよびログストレージサービスである Amazon CloudWatch にログを転送できます。デフォルトのログストアに加えて、またはログストアの代わりに、CloudWatch にログを転送できます。

CloudWatch へのログ転送を設定するには、CloudWatch の出力および出力を使用するパイプラインで **ClusterLogForwarder** カスタムリソース (CR) を作成する必要があります。

**手順**

1. **aws_access_key_id** および **aws_secret_access_key** フィールドを使用する **Secret** YAML ファイルを作成し、base64 でエンコードされた AWS 認証情報を指定します。以下に例を示します。

```
apiVersion: v1
kind: Secret
metadata:
  name: cw-secret
  namespace: openshift-logging
data:
  aws_access_key_id: QUtJQUlPU0ZPRE5ON0VYQU1QTEUK
  aws_secret_access_key:
d0phbHJYVXRuRkVNSS9LN01ERU5HL2JQeFJmaUNZRVhBTVBMRUtFWQo=
```

2. シークレットを作成します。以下に例を示します。

```
$ oc apply -f cw-secret.yaml
```

3. **ClusterLogForwarder** CR オブジェクトを定義する YAML ファイルを作成または編集します。このファイルに、シークレットの名前を指定します。以下に例を示します。

```
apiVersion: "logging.openshift.io/v1"
kind: ClusterLogForwarder
```

```
metadata:
  name: instance 1
  namespace: openshift-logging 2
spec:
  outputs:
   - name: cw 3
     type: cloudwatch 4
     cloudwatch:
       groupBy: logType 5
       groupPrefix: <group prefix> 6
       region: us-east-2 7
     secret:
        name: cw-secret 8
  pipelines:
    - name: infra-logs 9
      inputRefs: 10
        - infrastructure
        - audit
        - application
      outputRefs:
        - cw 11
```

1. **ClusterLogForwarder** CR の名前は **instance** である必要があります。
2. **ClusterLogForwarder** CR の namespace は **openshift-logging** である必要があります。
3. 出力の名前を指定します。
4. **cloudwatch** タイプを指定します。
5. オプション: ログをグループ化する方法を指定します。
   - **logType** は、ログタイプごとにロググループを作成します。
   - **namespaceName** は、アプリケーションの namespace ごとにロググループを作成します。また、インフラストラクチャーおよび監査ログ用の個別のロググループも作成します。
   - **namespaceUUID** は、アプリケーション namespace UUID ごとに新しいロググループを作成します。また、インフラストラクチャーおよび監査ログ用の個別のロググループも作成します。
6. オプション: ロググループの名前に含まれるデフォルトの **infrastructureName** 接頭辞を置き換える文字列を指定します。
7. AWS リージョンを指定します。
8. AWS 認証情報が含まれるシークレットの名前を指定します。
9. オプション: パイプラインの名前を指定します。
10. パイプラインを使用して転送するログタイプ (**application**、**infrastructure** または **audit**) を指定します。
11. このパイプラインでログを転送する時に使用する出力の名前を指定します。

4. CR オブジェクトを作成します。

```
$ oc create -f <file-name>.yaml
```

### 例: Amazon CloudWatch での ClusterLogForwarder の使用

ここでは、**ClusterLogForwarder** カスタムリソース (CR) のサンプルと、Amazon CloudWatch に出力するログデータが表示されます。

**mycluster** という名前の OpenShift Container Platform クラスターを実行しているとします。以下のコマンドは、クラスターの **infrastructureName** を返します。これは、後で **aws** コマンドの作成に使用します。

```
$ oc get Infrastructure/cluster -ojson | jq .status.infrastructureName
"mycluster-7977k"
```

この例のログデータを生成するには、**app** という名前の namespace で **busybox** pod を実行します。**busybox** pod は、3 秒ごとに stdout にメッセージを書き込みます。

```
$ oc run busybox --image=busybox -- sh -c 'while true; do echo "My life is my message"; sleep 3;
done'
$ oc logs -f busybox
My life is my message
My life is my message
My life is my message
...
```

**busybox** pod が実行される **app** namespace の UUID を検索できます。

```
$ oc get ns/app -ojson | jq .metadata.uid
"794e1e1a-b9f5-4958-a190-e76a9b53d7bf"
```

**ClusterLogForwarder** カスタムリソース (CR) で、**インフラストラクチャー**、**監査**、および **アプリケーションログ** タイプを **all-logs** パイプラインへの入力として設定します。また、このパイプラインを **cw** 出力に接続し、**us-east-2** リージョンの CloudWatch インスタンスに転送します。

```
apiVersion: "logging.openshift.io/v1"
kind: ClusterLogForwarder
metadata:
  name: instance
  namespace: openshift-logging
spec:
  outputs:
   - name: cw
     type: cloudwatch
     cloudwatch:
       groupBy: logType
       region: us-east-2
     secret:
        name: cw-secret
  pipelines:
    - name: all-logs
      inputRefs:
        - infrastructure
        - audit
```

```
    - application
  outputRefs:
    - cw
```

CloudWatch の各リージョンには、3 つのレベルのオブジェクトが含まれます。

- ロググループ
  - ログストリーム
    - ログイベント

**ClusterLogForwarding** CR の **groupBy: logType** の場合に、**inputRefs** にある 3 つのログタイプで Amazon Cloudwatch に 3 つのロググループを生成します。

```
$ aws --output json logs describe-log-groups | jq .logGroups[].logGroupName
"mycluster-7977k.application"
"mycluster-7977k.audit"
"mycluster-7977k.infrastructure"
```

各ロググループにはログストリームが含まれます。

```
$ aws --output json logs describe-log-streams --log-group-name mycluster-7977k.application | jq
.logStreams[].logStreamName
"kubernetes.var.log.containers.busybox_app_busybox-
da085893053e20beddd6747acdbaf98e77c37718f85a7f6a4facf09ca195ad76.log"
```

```
$ aws --output json logs describe-log-streams --log-group-name mycluster-7977k.audit | jq
.logStreams[].logStreamName
"ip-10-0-131-228.us-east-2.compute.internal.k8s-audit.log"
"ip-10-0-131-228.us-east-2.compute.internal.linux-audit.log"
"ip-10-0-131-228.us-east-2.compute.internal.openshift-audit.log"
...
```

```
$ aws --output json logs describe-log-streams --log-group-name mycluster-7977k.infrastructure | jq
.logStreams[].logStreamName
"ip-10-0-131-228.us-east-2.compute.internal.kubernetes.var.log.containers.apiserver-69f9fd9b58-
zqzw5_openshift-oauth-apiserver_oauth-apiserver-
453c5c4ee026fe20a6139ba6b1cdd1bed25989c905bf5ac5ca211b7cbb5c3d7b.log"
"ip-10-0-131-228.us-east-2.compute.internal.kubernetes.var.log.containers.apiserver-797774f7c5-
lftrx_openshift-apiserver_openshift-apiserver-
ce51532df7d4e4d5f21c4f4be05f6575b93196336be0027067fd7d93d70f66a4.log"
"ip-10-0-131-228.us-east-2.compute.internal.kubernetes.var.log.containers.apiserver-797774f7c5-
lftrx_openshift-apiserver_openshift-apiserver-check-endpoints-
82a9096b5931b5c3b1d6dc4b66113252da4a6472c9fff48623baee761911a9ef.log"
...
```

各ログストリームにはログイベントが含まれます。**busybox** Pod からログイベントを表示するには、**application** ロググループからログストリームを指定します。

```
$ aws logs get-log-events --log-group-name mycluster-7977k.application --log-stream-name
kubernetes.var.log.containers.busybox_app_busybox-
da085893053e20beddd6747acdbaf98e77c37718f85a7f6a4facf09ca195ad76.log
{
```

```
    "events": [
        {
            "timestamp": 1629422704178,
            "message": "{\"docker\":
{\"container_id\":\"da085893053e20beddd6747acdbaf98e77c37718f85a7f6a4facf09ca195ad76\"},\"kub
ernetes\":
{\"container_name\":\"busybox\",\"namespace_name\":\"app\",\"pod_name\":\"busybox\",\"container_ima
ge\":\"docker.io/library/busybox:latest\",\"container_image_id\":\"docker.io/library/busybox@sha256:0f35
4ec1728d9ff32edcd7d1b8bbdfc798277ad36120dc3dc683be44524c8b60\",\"pod_id\":\"870be234-
90a3-4258-b73f-4f4d6e2777c7\",\"host\":\"ip-10-0-216-3.us-east-2.compute.internal\",\"labels\":
{\"run\":\"busybox\"},\"master_url\":\"https://kubernetes.default.svc\",\"namespace_id\":\"794e1e1a-
b9f5-4958-a190-e76a9b53d7bf\",\"namespace_labels\":
{\"kubernetes_io/metadata_name\":\"app\"}},\"message\":\"My life is my
message\",\"level\":\"unknown\",\"hostname\":\"ip-10-0-216-3.us-east-
2.compute.internal\",\"pipeline_metadata\":{\"collector\":
{\"ipaddr4\":\"10.0.216.3\",\"inputname\":\"fluent-plugin-
systemd\",\"name\":\"fluentd\",\"received_at\":\"2021-08-
20T01:25:08.085760+00:00\",\"version\":\"1.7.4 1.6.0\"}},\"@timestamp\":\"2021-08-
20T01:25:04.178986+00:00\",\"viaq_index_name\":\"app-
write\",\"viaq_msg_id\":\"NWRjZmUyMWQtZjgzNC00MjI4LTk3MjMtNTk3NmY3ZjU4NDk1\",\"log_type\":
\"application\",\"time\":\"2021-08-20T01:25:04+00:00\"}",
            "ingestionTime": 1629422744016
        },
...
```

### 例: ロググループ名の接頭辞のカスタマイズ

ロググループ名では、デフォルトの **infrastructureName** 接頭辞 **mycluster-7977k** は **demo-group-prefix** のように任意の文字列に置き換えることができます。この変更を加えるには、**ClusterLogForwarding** CR の **groupPrefix** フィールドを更新します。

```
cloudwatch:
    groupBy: logType
    groupPrefix: demo-group-prefix
    region: us-east-2
```

**groupPrefix** の値は、デフォルトの **infrastructureName** 接頭辞を置き換えます。

```
$ aws --output json logs describe-log-groups | jq .logGroups[].logGroupName
"demo-group-prefix.application"
"demo-group-prefix.audit"
"demo-group-prefix.infrastructure"
```

### 例: アプリケーションの namespace 名をもとにロググループの命名

クラスター内のアプリケーション namespace ごとに、名前がアプリケーション namespace 名をもとにする CloudWatch にロググループを作成できます。

アプリケーションの namespace オブジェクトを削除して、同じ名前の新しいオブジェクトを作成する場合は、CloudWatch は以前と同じロググループを使用し続けます。

相互に名前が同じアプリケーション namespace オブジェクトを引き継ぐ予定の場合は、この例で説明されている方法を使用します。それ以外で、生成されるログメッセージを相互に区別する必要がある場合は、代わりに Naming log groups for application namespace UUIDs のセクションを参照してください。

アプリケーション namespace 名を基にした名前を指定してアプリケーションロググループを作成するには、**ClusterLogForwarder** CR で **groupBy** フィールドの値を **namespaceName** に設定します。

```
cloudwatch:
  groupBy: namespaceName
  region: us-east-2
```

**groupBy** を **namespaceName** に設定すると、アプリケーションロググループのみが影響を受けます。これは、**audit** および **infrastructure** のロググループには影響しません。

Amazon Cloudwatch では、namespace 名が各ロググループ名の最後に表示されます。アプリケーション namespace (app) が 1 つであるため、以下の出力は **mycluster-7977k.application** ではなく、新しい **mycluster-7977k.app** ロググループを示しています。

```
$ aws --output json logs describe-log-groups | jq .logGroups[].logGroupName
"mycluster-7977k.app"
"mycluster-7977k.audit"
"mycluster-7977k.infrastructure"
```

この例のクラスターに複数のアプリケーション namespace が含まれる場合は、出力には namespace ごとに複数のロググループが表示されます。

**groupBy** フィールドは、アプリケーションロググループだけに影響します。これは、**audit** および **infrastructure** のロググループには影響しません。

### 例: アプリケーション namespace UUID をもとにロググループの命名

クラスター内のアプリケーション namespace ごとに、名前がアプリケーション namespace の UUID をもとにする CloudWatch にロググループを作成できます。

アプリケーションの namespace オブジェクトを削除して新規のロググループを作成する場合は、CloudWatch で新しいロググループを作成します。

相互に名前が異なるアプリケーション namespace オブジェクトを引き継ぐ予定の場合は、この例で説明されている方法を使用します。それ以外の場合は、前述の例: Naming log groups for application namespace name のセクションを参照してください。

アプリケーション namespace UUID をもとにログエントリーに名前を付けるには、**ClusterLogForwarder** CR で **groupBy** フィールドの値を **namespaceUUID** に設定します。

```
cloudwatch:
  groupBy: namespaceUUID
  region: us-east-2
```

Amazon Cloudwatch では、namespace UUID が各ロググループ名の最後に表示されます。アプリケーション namespace (app) が 1 つであるため、以下の出力は **mycluster-7977k.application** ではなく、新しい **mycluster-7977k.794e1e1a-b9f5-4958-a190-e76a9b53d7bf** ロググループを示しています。

```
$ aws --output json logs describe-log-groups | jq .logGroups[].logGroupName
"mycluster-7977k.794e1e1a-b9f5-4958-a190-e76a9b53d7bf" // uid of the "app" namespace
"mycluster-7977k.audit"
"mycluster-7977k.infrastructure"
```

**groupBy** フィールドは、アプリケーションロググループだけに影響します。これは、**audit** および **infrastructure** のロググループには影響しません。

### 11.12.1. STS 対応クラスターから Amazon CloudWatch へのログ転送

AWS Security Token Service (STS) が有効になっているクラスターの場合に、AWS サービスアカウントを手動で作成するか、Cloud Credential Operator (CCO) ユーティリティー **ccoctl** を使用してクレデンシャルのリクエストを作成できます。

**注記**

この機能は、vector コレクターではサポートされていません。

**AWS 認証情報リクエストの作成**

1. 以下のテンプレートを使用して、**CredentialsRequest** カスタムリソース YAML を作成します。

   **CloudWatch クレデンシャルリクエストのテンプレート**

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <your_role_name>-credrequest
  namespace: openshift-cloud-credential-operator
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AWSProviderSpec
    statementEntries:
      - action:
          - logs:PutLogEvents
          - logs:CreateLogGroup
          - logs:PutRetentionPolicy
          - logs:CreateLogStream
          - logs:DescribeLogGroups
          - logs:DescribeLogStreams
        effect: Allow
        resource: arn:aws:logs:*:*:*
  secretRef:
    name: <your_role_name>
    namespace: openshift-logging
  serviceAccountNames:
    - logcollector
```

2. **ccoctl** コマンドを使用して、**CredentialsRequest** CR を使用して AWS のロールを作成します。**CredentialsRequest**オブジェクトでは、この **ccoctl** コマンドを使用すると、特定の OIDC アイデンティティープロバイダーに紐付けされたトラストポリシーと、CloudWatch リソースでの操作実行パーミッションを付与するパーミッションポリシーを指定して IAM ロールを作成します。このコマンドは、**/<path_to_ccoctl_output_dir>/manifests/openshift-logging-<your_role_name>-credentials.yaml** に YAML 設定ファイルも作成します。このシークレットファイルには、AWS IAM ID プロバイダーでの認証中に使用される **role_arn** キー/値が含まれています。

```
ccoctl aws create-iam-roles \
--name=<name> \
--region=<aws_region> \
--credentials-requests-dir=
```

```
<path_to_directory_with_list_of_credentials_requests>/credrequests \
--identity-provider-arn=arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.
<aws_region>.amazonaws.com (1)
```

(1) <name> は、クラウドリソースのタグ付けに使用される名前であり、STS クラスターのインストール中に使用される名前と一致する必要があります。

3. 作成したシークレットを適用します。

```
oc apply -f output/manifests/openshift-logging-<your_role_name>-credentials.yaml
```

4. **ClusterLogForwarder** カスタムリソースを作成または編集します。

```
apiVersion: "logging.openshift.io/v1"
kind: ClusterLogForwarder
metadata:
  name: instance (1)
  namespace: openshift-logging (2)
spec:
  outputs:
   - name: cw (3)
     type: cloudwatch (4)
     cloudwatch:
       groupBy: logType (5)
       groupPrefix: <group prefix> (6)
       region: us-east-2 (7)
     secret:
        name: <your_role_name> (8)
  pipelines:
   - name: to-cloudwatch (9)
     inputRefs: (10)
       - infrastructure
       - audit
       - application
     outputRefs:
       - cw (11)
```

(1) **ClusterLogForwarder** CR の名前は **instance** である必要があります。

(2) **ClusterLogForwarder** CR の namespace は **openshift-logging** である必要があります。

(3) 出力の名前を指定します。

(4) **cloudwatch** タイプを指定します。

(5) オプション: ログをグループ化する方法を指定します。

- **logType** は、ログタイプごとにロググループを作成します。
- **namespaceName** は、アプリケーションの namespace ごとにロググループを作成します。インフラストラクチャーおよび監査ログは影響を受けず、**logType** によってグループ化されたままになります。

- **namespaceUUID** は、アプリケーション namespace UUID ごとに新しいロググループを作成します。また、インフラストラクチャーおよび監査ログ用の個別のロググループも作成します。

6 オプション: ロググループの名前に含まれるデフォルトの **infrastructureName** 接頭辞を置き換える文字列を指定します。

7 AWS リージョンを指定します。

8 AWS 認証情報が含まれるシークレットの名前を指定します。

9 オプション: パイプラインの名前を指定します。

10 パイプラインを使用して転送するログタイプ (**application**、**infrastructure** または **audit**) を指定します。

11 このパイプラインでログを転送する時に使用する出力の名前を指定します。

**関連情報**

- AWS STS API リファレンス

#### 11.12.1.1. 既存の AWS ロールを使用した AWS CloudWatch のシークレット作成

AWS の既存のロールがある場合は、**oc create secret --from-literal** コマンドを使用して、STS で AWS のシークレットを作成できます。

```
oc create secret generic cw-sts-secret -n openshift-logging --from-
literal=role_arn=arn:aws:iam::123456789012:role/my-role_with-permissions
```

**シークレットの例**

```
apiVersion: v1
kind: Secret
metadata:
  namespace: openshift-logging
  name: my-secret-name
stringData:
  role_arn: arn:aws:iam::123456789012:role/my-role_with-permissions
```

## 11.13. ログの LOKI への転送

内部のデフォルト OpenShift Container Platform Elasticsearch インスタンスに加えて、またはその代わりに外部の Loki ロギングシステムにログを転送できます。

Loki へのログ転送を設定するには、Loki の出力と、出力を使用するパイプラインで **ClusterLogForwarder** カスタムリソース (CR) を作成する必要があります。Loki への出力は HTTP (セキュアでない) または HTTPS (セキュアな HTTP) 接続を使用できます。

**前提条件**

- CR の **url** フィールドで指定する URL で Loki ロギングシステムが実行されている必要がある。

**手順**

1. **ClusterLogForwarder** CR オブジェクトを定義する YAML ファイルを作成または編集します。

```
apiVersion: "logging.openshift.io/v1"
kind: ClusterLogForwarder
metadata:
  name: instance 1
  namespace: openshift-logging 2
spec:
  outputs:
   - name: loki-insecure 3
     type: "loki" 4
     url: http://loki.insecure.com:3100 5
     loki:
       tenantKey: kubernetes.namespace_name
       labelKeys: kubernetes.labels.foo
   - name: loki-secure 6
     type: "loki"
     url: https://loki.secure.com:3100
     secret:
       name: loki-secret 7
     loki:
       tenantKey: kubernetes.namespace_name 8
       labelKeys: kubernetes.labels.foo 9
  pipelines:
   - name: application-logs 10
     inputRefs: 11
     - application
     - audit
     outputRefs: 12
     - loki-secure
```

1. **ClusterLogForwarder** CR の名前は **instance** である必要があります。
2. **ClusterLogForwarder** CR の namespace は **openshift-logging** である必要があります。
3. 出力の名前を指定します。
4. タイプを **loki** として指定します。
5. Loki システムの URL およびポートを有効な絶対 URL として指定します。**http** (セキュアでない) プロトコルまたは **https** (セキュアな HTTP) プロトコルを使用できます。CIDR アノテーションを使用するクラスター全体のプロキシーが有効になっている場合、出力は IP アドレスではなくサーバー名または FQDN である必要があります。HTTP(S) 通信用の Loki のデフォルトポートは 3100 です。
6. セキュアな接続では、**シークレット** を指定して、認証する **https** または **http** URL を指定できます。
7. **https** 接頭辞の場合は、TLS 通信のエンドポイントに必要なシークレットの名前を指定します。シークレットは **openshift-logging** プロジェクトに存在し、**tls.crt**、**tls.key** および **ca-bundle.crt** のキーが含まれる必要があります。これらは、それぞれが表す証明書を参照します。それ以外の場合、**http** および **https** 接頭辞の場合は、ユーザー名とパスワードを含むシークレットを指定できます。詳細は、Example: Setting secret that contains a

username and password.を参照してください。

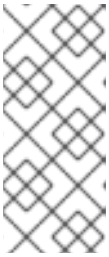
8 オプション: メタデータキーフィールドを指定して、Loki の **TenantID** フィールドの値を生成します。たとえば、**tenantKey: kubernetes.namespace_name** を設定すると、Kubernetes namespace の名前を Loki のテナント ID の値として使用します。他にどのログレコードフィールドを指定できるかを確認するには、以下の Additional resources セクションの Log Record Fields リンクを参照してください。

9 オプション: デフォルトの Loki ラベルを置き換えるメタデータフィールドキーのリストを指定します。loki ラベル名は、正規表現 **[a-zA-Z_:][a-zA-Z0-9_:]*** と一致する必要があります。ラベル名を形成するため、メタデータキーの無効な文字は _ に置き換えられます。たとえば、**kubernetes.labels.foo** meta-data キーは Loki ラベル **kubernetes_labels_foo** になります。**labelKeys** を設定しないと、デフォルト値は **[log_type, kubernetes.namespace_name, kubernetes.pod_name, kubernetes_host]** です。Loki で指定可能なラベルのサイズと数に制限があるため、ラベルのセットを小さくします。Configuring Loki, limits_config を参照してください。クエリーフィルターを使用して、ログレコードフィールドに基づいてクエリーを実行できます。

10 オプション: パイプラインの名前を指定します。

11 パイプラインを使用して転送するログタイプ (**application**、**infrastructure** または **audit**) を指定します。

12 このパイプラインでログを転送する時に使用する出力の名前を指定します。



**注記**

Loki ではログストリームを正しくタイムスタンプで順序付ける必要があるため、**labelKeys** には指定しなくても **kubernetes_host** ラベルセットが常に含まれます。このラベルセットが含まれることで、各ストリームが 1 つのホストから発信されるので、ホストのクロック間の誤差が原因でタイムスタンプの順番が乱れないようになります。

2. CR オブジェクトを作成します。

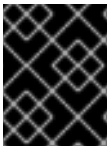
```
$ oc create -f <file-name>.yaml
```

### 11.13.1. Loki レート制限エラーのトラブルシューティング

Log Forwarder API がレート制限を超える大きなメッセージブロックを Loki に転送すると、Loki により、レート制限 (**429**) エラーが生成されます。

これらのエラーは、通常の動作中に発生する可能性があります。たとえば、すでにいくつかのログがあるクラスターにログサブシステムを追加する場合、ログサブシステムが既存のログエントリーをすべて取り込もうとするとレート制限エラーが発生する可能性があります。この場合、新しいログの追加速度が合計レート制限よりも低い場合、履歴データは最終的に取り込まれ、ユーザーの介入を必要とせずにレート制限エラーが解決されます。

レート制限エラーが引き続き発生する場合は、**LokiStack** カスタムリソース (CR) を変更することで問題を解決できます。



**重要**

**LokiStack** CR は、Grafana がホストする Loki では利用できません。このトピックは、Grafana がホストする Loki サーバーには適用されません。

**条件**

- Log Forwarder API は、ログを Loki に転送するように設定されている。
- システムは、次のような 2MB を超えるメッセージのブロックを Loki に送信する。以下に例を示します。

```
"values":[["1630410392689800468","{\"kind\":\"Event\",\"apiVersion\":\
.......
......
......
......
\"received_at\":\"2021-08-31T11:46:32.800278+00:00\",\"version\":\"1.7.4
1.6.0\"}},\"@timestamp\":\"2021-08-
31T11:46:32.799692+00:00\",\"viaq_index_name\":\"audit-
write\",\"viaq_msg_id\":\"MzFjYjJkZjItNjY0MC00YWU4LWIwMTEtNGNmM2E5ZmViMGU4\",\"log_type\":\"audit\"}"]]}]}
```

- **oc logs -n openshift-logging -l component=collector** と入力すると、クラスター内のコレクターログに、次のいずれかのエラーメッセージを含む行が表示されます。

```
429 Too Many Requests Ingestion rate limit exceeded
```

**Vector エラーメッセージの例**

```
2023-08-25T16:08:49.301780Z  WARN sink{component_kind="sink"
component_id=default_loki_infra component_type=loki component_name=default_loki_infra}:
vector::sinks::util::retries: Retrying after error. error=Server responded with an error: 429 Too
Many Requests internal_log_rate_limit=true
```

**Fluentd エラーメッセージの例**

```
2023-08-30 14:52:15 +0000 [warn]: [default_loki_infra] failed to flush the buffer. retry_times=2
next_retry_time=2023-08-30 14:52:19 +0000
chunk="604251225bf5378ed1567231a1c03b8b"
error_class=Fluent::Plugin::LokiOutput::LogPostError error="429 Too Many Requests
Ingestion rate limit exceeded for user infrastructure (limit: 4194304 bytes/sec) while
attempting to ingest '4082' lines totaling '7820025' bytes, reduce log volume or contact your
Loki administrator to see if the limit can be increased\n"
```

このエラーは受信側にも表示されます。たとえば、LokiStack 取り込み Pod で以下を行います。

**Loki 取り込みエラーメッセージの例**

```
level=warn ts=2023-08-30T14:57:34.155592243Z caller=grpc_logging.go:43
duration=1.434942ms method=/logproto.Pusher/Push err="rpc error: code = Code(429) desc
= entry with timestamp 2023-08-30 14:57:32.012778399 +0000 UTC ignored, reason: 'Per
stream rate limit exceeded (limit: 3MB/sec) while attempting to ingest for stream
```

-

**手順**

- **LokiStack** CR の **ingestionBurstSize** および **ingestionRate** フィールドを更新します。

```
apiVersion: loki.grafana.com/v1
kind: LokiStack
metadata:
  name: logging-loki
  namespace: openshift-logging
spec:
  limits:
    global:
      ingestion:
        ingestionBurstSize: 16 (1)
        ingestionRate: 8 (2)
# ...
```

(1) **ingestionBurstSize** フィールドは、ディストリビューターレプリカごとに最大ローカルレート制限サンプルサイズを MB 単位で定義します。この値はハードリミットです。この値を、少なくとも 1 つのプッシュリクエストで想定される最大ログサイズに設定します。**ingestionBurstSize** 値より大きい単一リクエストは使用できません。

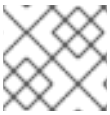
(2) **ingestionRate** フィールドは、1 秒あたりに取り込まれるサンプルの最大量 (MB 単位) に対するソフト制限です。ログのレートが制限を超えているにもかかわらず、コレクターがログの送信を再試行すると、レート制限エラーが発生します。合計平均が制限よりも少ない場合に限り、システムは回復し、ユーザーの介入なしでエラーが解決されます。

**関連情報**

- ログレコードのフィールド
- Loki サーバーの設定

## 11.14. ログの GOOGLE CLOUD PLATFORM (GCP) への転送

内部のデフォルトの OpenShift Container Platform ログストアに加えて、またはその代わりに、ログを Google Cloud Logging に転送できます。



**注記**

この機能を Fluentd で使用することはサポートされていません。

**前提条件**

- Red Hat OpenShift Operator 5.5.1 以降のロギングサブシステム

**手順**

1. Google サービスアカウントキー を使用してシークレットを作成します。

```
$ oc -n openshift-logging create secret generic gcp-secret --from-file google-application-credentials.json=<your_service_account_key_file.json>
```

2. 以下のテンプレートを使用して、**ClusterLogForwarder** カスタムリソース YAML を作成します。

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogForwarder"
metadata:
  name: "instance"
  namespace: "openshift-logging"
spec:
  outputs:
    - name: gcp-1
      type: googleCloudLogging
      secret:
        name: gcp-secret
      googleCloudLogging:
        projectId : "openshift-gce-devel" 1
        logId : "app-gcp" 2
  pipelines:
    - name: test-app
      inputRefs: 3
        - application
      outputRefs:
        - gcp-1
```

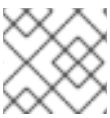
1. ログを保存する GCP リソース階層 の場所に応じて、**projectId**、**folderId**、**organizationId**、または **billingAccountId** フィールドとそれに対応する値を設定します。
2. Log Entry の **logName** フィールドに追加する値を設定します。
3. パイプラインを使用して転送するログタイプ (**application**、**infrastructure**、または **audit**) を指定します。

**関連情報**

- Google Cloud Billing に関するドキュメント
- Google Cloud Logging クエリー言語のドキュメント

## 11.15. ログの SPLUNK への転送

内部のデフォルトの OpenShift Container Platform ログストアに加えて、またはその代わりに、Splunk HTTP Event Collector (HEC) にログを転送できます。



**注記**

この機能を Fluentd で使用することはサポートされていません。

**前提条件**

- Red Hat OpenShift Logging Operator 5.6 以降

- コレクターとして vector が指定された ClusterLogging インスタンス
- Base64 でエンコードされた Splunk HEC トークン

**手順**

1. Base64 でエンコードされた Splunk HEC トークンを使用してシークレットを作成します。

```
$ oc -n openshift-logging create secret generic vector-splunk-secret --from-literal hecToken=
<HEC_Token>
```

2. 以下のテンプレートを使用して、**ClusterLogForwarder** カスタムリソース (CR) を作成または編集します。

```
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogForwarder"
metadata:
  name: "instance" 1
  namespace: "openshift-logging" 2
spec:
  outputs:
    - name: splunk-receiver 3
      secret:
        name: vector-splunk-secret 4
      type: splunk 5
      url: <http://your.splunk.hec.url:8088> 6
  pipelines: 7
    - inputRefs:
        - application
        - infrastructure
      name: 8
      outputRefs:
        - splunk-receiver 9
```

1. ClusterLogForwarder CR の名前は **instance** である必要があります。
2. ClusterLogForwarder CR の namespace は **openshift-logging** である必要があります。
3. 出力の名前を指定します。
4. HEC トークンが含まれるシークレットの名前を指定します。
5. 出力タイプを **splunk** として指定します。
6. Splunk HEC の URL (ポートを含む) を指定します。
7. パイプラインを使用して転送するログタイプ (**application**、**infrastructure**、または **audit**) を指定します。
8. オプション: パイプラインの名前を指定します。
9. このパイプラインでログを転送する時に使用する出力の名前を指定します。

## 11.16. 特定のプロジェクトからのアプリケーションログの転送

クラスターログフォワーダーを使用して、外部ログアグリゲーターに、特定のプロジェクトからアプリケーションログのコピーを送信できます。これは、デフォルトの Elasticsearch ログストアの代わりに、またはこれに加えてデフォルトの Elasticsearch ログストアを使用して実行できます。また、外部ログアグリゲーターを OpenShift Container Platform からログデータを受信できるように設定する必要もあります。

アプリケーションログのプロジェクトからの転送を設定するには、プロジェクトから少なくとも 1 つの入力で **ClusterLogForwarder** カスタムリソース (CR) を作成し、他のログアグリゲーターのオプション出力、およびそれらの入出力を使用するパイプラインを作成する必要があります。

### 前提条件

- 指定されたプロトコルまたは形式を使用してロギングデータを受信するように設定されたロギングサーバーが必要です。

### 手順

1. **ClusterLogForwarder** CR オブジェクトを定義する YAML ファイルを作成または編集します。

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogForwarder
metadata:
  name: instance 1
  namespace: openshift-logging 2
spec:
  outputs:
   - name: fluentd-server-secure 3
     type: fluentdForward 4
     url: 'tls://fluentdserver.security.example.com:24224' 5
     secret: 6
        name: fluentd-secret
   - name: fluentd-server-insecure
     type: fluentdForward
     url: 'tcp://fluentdserver.home.example.com:24224'
  inputs: 7
   - name: my-app-logs
     application:
        namespaces:
        - my-project
  pipelines:
   - name: forward-to-fluentd-insecure 8
     inputRefs: 9
     - my-app-logs
     outputRefs: 10
     - fluentd-server-insecure
     parse: json 11
     labels:
       project: "my-project" 12
   - name: forward-to-fluentd-secure 13
     inputRefs:
     - application
     - audit
```

```
  - infrastructure
  outputRefs:
  - fluentd-server-secure
  - default
  labels:
    clusterId: "C1234"
```

1. **ClusterLogForwarder** CR の名前は **instance** である必要があります。
2. **ClusterLogForwarder** CR の namespace は **openshift-logging** である必要があります。
3. 出力の名前を指定します。
4. 出力タイプ **elasticsearch**、**fluentdForward**、**syslog**、または **kafka** を指定します。
5. 外部ログアグリゲーターの URL およびポートを有効な絶対 URL として指定します。CIDR アノテーションを使用するクラスター全体のプロキシーが有効になっている場合、出力は IP アドレスではなくサーバー名または FQDN である必要があります。
6. **tls** 接頭辞を使用する場合は、TLS 通信のエンドポイントに必要なシークレットの名前を指定する必要があります。シークレットは **openshift-logging** プロジェクトに存在し、**tls.crt**、 **tls.key**、 および **ca-bundle.crt** キーが含まれる必要があります。これらは、それぞれが表す証明書を参照します。
7. 指定されたプロジェクトからアプリケーションログをフィルターするための入力の設定。
8. 入力を使用してプロジェクトアプリケーションログを外部 Fluentd インスタンスに送信するためのパイプラインの設定。
9. **my-app-logs** 入力。
10. 使用する出力の名前。
11. オプション: 構造化された JSON ログエントリーを **structured** フィールドの JSON オブジェクトとして転送するかどうかを指定します。ログエントリーに有効な構造化された JSON が含まれる必要があります。そうでない場合は、OpenShift Logging は **構造化** フィールドを削除し、代わりにログエントリーをデフォルトのインデックス **app-00000x** に送信します。
12. オプション: 文字列。ログに追加する 1 つまたは複数のラベル。
13. ログを他のログアグリゲーターに送信するためのパイプラインの設定。
    - オプション: パイプラインの名前を指定します。
    - パイプラインを使用して転送するログタイプ (**application**、**infrastructure** または **audit**) を指定します。
    - このパイプラインでログを転送する時に使用する出力の名前を指定します。
    - オプション: ログを内部 Elasticsearch インスタンスに転送するために **default** 出力を指定します。
    - オプション: 文字列。ログに追加する 1 つまたは複数のラベル。

2. CR オブジェクトを作成します。

-

```
$ oc create -f <file-name>.yaml
```

## 11.17. 特定の POD からのアプリケーションログの転送

クラスター管理者は、Kubernetes Pod ラベルを使用して特定の Pod からログデータを収集し、これをログコレクターに転送できます。

アプリケーションがさまざまな namespace の他の Pod と共に実行される Pod で設定されるとします。これらの Pod にアプリケーションを識別するラベルがある場合は、それらのログデータを収集し、特定のログコレクターに出力できます。

Pod ラベルを指定するには、1 つ以上の **matchLabels** のキー/値のペアを使用します。複数のキー/値のペアを指定する場合、Pod は選択されるそれらすべてに一致する必要があります。

**手順**

1. **ClusterLogForwarder** CR オブジェクトを定義する YAML ファイルを作成または編集します。ファイルで、以下の例が示すように **inputs[].name.application.selector.matchLabels** の下で単純な等価ベース (Equality-based) のセレクターを使用して Pod ラベルを指定します。

   **ClusterLogForwarder CR YAML ファイルのサンプル**

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogForwarder
metadata:
  name: instance 1
  namespace: openshift-logging 2
spec:
  pipelines:
    - inputRefs: [ myAppLogData ] 3
      outputRefs: [ default ] 4
      parse: json 5
  inputs: 6
    - name: myAppLogData
      application:
        selector:
          matchLabels: 7
            environment: production
            app: nginx
        namespaces: 8
        - app1
        - app2
  outputs: 9
    - default
    ...
```

1 **ClusterLogForwarder** CR の名前は **instance** である必要があります。

2 **ClusterLogForwarder** CR の namespace は **openshift-logging** である必要があります。

3 **inputs[].name** から 1 つ以上のコンマ区切りの値を指定します。

4 **outputs[]** から 1 つ以上のコンマ区切りの値を指定します。

5 オプション: 構造化された JSON ログエントリーを **structured** フィールドの JSON オブジェクトとして転送するかどうかを指定します。ログエントリーに有効な構造化された

6 Pod ラベルの一意のセットを持つ各アプリケーションの一意の **inputs[].name** を定義します。

7 収集するログデータを持つ Pod ラベルのキー/値のペアを指定します。キーだけではなく、キーと値の両方を指定する必要があります。Pod を選択するには、Pod はすべてのキーと値のペアと一致する必要があります。

8 オプション: namespace を 1 つ以上指定します。

9 ログデータを転送する 1 つ以上の出力を指定します。ここで表示されるオプションの **default** 出力はログデータを内部 Elasticsearch インスタンスに送信します。

2. オプション: ログデータの収集を特定の namespace に制限するには、前述の例のように **inputs[].name.application.namespaces** を使用します。

3. オプション: 異なる Pod ラベルを持つ追加のアプリケーションから同じパイプラインにログデータを送信できます。

   a. Pod ラベルの一意の組み合わせごとに、表示されるものと同様の追加の **inputs[].name** セクションを作成します。

   b. このアプリケーションの Pod ラベルに一致するように、**selectors** を更新します。

   c. 新規の **inputs[].name** 値を **inputRefs** に追加します。以下に例を示します。

   ```
   - inputRefs: [ myAppLogData, myOtherAppLogData ]
   ```

4. CR オブジェクトを作成します。

   ```
   $ oc create -f <file-name>.yaml
   ```

**関連情報**

- Kubernetes の **matchLabels** の詳細は、セットベースの要件をサポートするリソース を参照してください。

**関連情報**

- ネットワークポリシー監査ロギング

## 11.18. ログ転送のトラブルシューティング

**ClusterLogForwarder** カスタムリソース (CR) の作成時に、Red Hat OpenShift Logging Operator により Fluentd Pod が自動的に再デプロイされない場合は、Fluentd Pod を削除して、強制的に再デプロイできます。

**前提条件**

- **ClusterLogForwarder** カスタムリソース (CR) オブジェクトを作成している。

**手順**

- Fluentd Pod を削除して強制的に再デプロイします。

```
$ oc delete pod --selector logging-infra=collector
```

# 第12章 JSON ロギングの有効化

ログ転送 API を設定して、構造化されたオブジェクトに対して JSON 文字列を解析できます。

## 12.1. JSON ログの解析

JSON ログなどのログは、通常 **message** フィールド内の文字列として表されます。これにより、JSON ドキュメント内の特定のフィールドをクエリーすることが困難になります。OpenShift Logging のログ転送 API を使用すると、JSON ログを構造化オブジェクトに解析し、それらを OpenShift Logging が管理する Elasticsearch またはログ転送 API でサポートされる他のサードパーティーシステムに転送できます。

以下の構造化された JSON ログエントリーがあると想定して、これがどのように機能するか説明します。

**構造化された JSON ログエントリーの例**

```
{"level":"info","name":"fred","home":"bedrock"}
```

通常、**ClusterLogForwarder** カスタムリソース (CR) は、そのログエントリーを **message** フィールドに転送します。**message** フィールドには、以下の例のように JSON ログエントリーと同等の JSON 引用符で囲まれた文字列が含まれます。

**message フィールドの例**

```
{"message":"{\"level\":\"info\",\"name\":\"fred\",\"home\":\"bedrock\"",
 "more fields..."}
```

JSON ログの解析を有効にするには、以下の例のように、**parse: json** を **ClusterLogForwarder** CR のパイプラインに追加します。

**parse: json を示すスニペット例**

```
pipelines:
- inputRefs: [ application ]
  outputRefs: myFluentd
  parse: json
```

**parse: json** を使用して JSON ログの解析を有効にすると、以下の例のように CR は **構造化** フィールドに JSON-structured ログエントリーをコピーします。元の **message** フィールドは変更されません。

**構造化された JSON ログエントリーを含む 構造化された 出力例**

```
{"structured": { "level": "info", "name": "fred", "home": "bedrock" },
 "more fields..."}
```

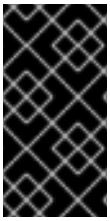
**重要**

ログエントリーに有効な構造化された JSON が含まれていない場合に、**構造化された** フィールドはなくなります。

特定のロギングプラットフォームの JSON ログの解析を有効にするには、ログのサードパーティーシステムへの転送 を参照してください。

## 12.2. ELASTICSEARCH の JSON ログデータの設定

JSON ログが複数のスキーマに従う場合は、それらを 1 つのインデックスに保存すると、タイプの競合やカーディナリティーの問題が発生する可能性があります。これを回避するには、1 つの出力定義に、各スキーマをグループ化するように **ClusterLogForwarder** カスタムリソース (CR) を設定する必要があります。これにより、各スキーマが別のインデックスに転送されます。



**重要**

JSON ログを OpenShift Logging によって管理されるデフォルトの Elasticsearch インスタンスに転送する場合に、設定に基づいて新規インデックスが生成されます。インデックスが多すぎることが原因のパフォーマンスの問題を回避するには、共通のスキーマに標準化して使用できるスキーマの数を保持することを検討してください。

### 構造化タイプ

**ClusterLogForwarder** CR で以下の構造タイプを使用し、Elasticsearch ログストアのインデックス名を作成できます。

- **structuredTypeKey** (string, optional) は、メッセージフィールドの名前です。このフィールドの値 (ある場合) はインデックス名の作成に使用されます。
  - **kubernetes.labels.<key>** は、インデックス名の作成に使用される Kubernetes pod ラベルの値です。
  - **openshift.labels.<key>** は、インデックス名の作成に使用される **ClusterLogForwarder** CR の **pipeline.label.<key>**; 要素です。
  - **kubernetes.container_name** はコンテナー名を使用してインデックス名を作成します。
- **structuredTypeName**:(文字列、オプション) **structuredTypeKey** が設定されておらず、そのキーが存在しない場合、OpenShift Logging は **structuredTypeName** の値を構造化型として使用します。**structuredTypeKey** and **structuredTypeName** の両方を使用する場合に、**structuredTypeName** は、構造化された **TypeKey** のキーが JSON ログデータにない場合にフォールバックインデックス名を指定します。

**注記**

**structuredTypeKey** の値を Log Record Fields トピックに記載されている任意のフィールドに設定できますが、構造タイプの前に来るリストに最も便利なフィールドが表示されます。

### structuredTypeKey: kubernetes.labels.<key> の例

以下と仮定します。

- クラスターが、apache および google という 2 つの異なる形式で JSON ログを生成するアプリケーション Pod を実行している。
- ユーザーはこれらのアプリケーション Pod に **logFormat=apache** と **logFormat=google** のラベルを付ける。
- 以下のスニペットを **ClusterLogForwarder** CR YAML ファイルで使用する。

```
outputDefaults:
 elasticsearch:
    structuredTypeKey: kubernetes.labels.logFormat 1
    structuredTypeName: nologformat
pipelines:
- inputRefs: <application>
  outputRefs: default
  parse: json 2
```

1 Kubernetes **logFormat** ラベルで形成される key-value ペアの値を使用します。

2 JSON ログの解析を有効にします。

この場合は、以下の構造化ログレコードが **app-apache-write** インデックスに送信されます。

```
{
  "structured":{"name":"fred","home":"bedrock"},
  "kubernetes":{"labels":{"logFormat": "apache", ...}}
}
```

また、以下の構造化ログレコードは **app-google-write** インデックスに送信されます。

```
{
  "structured":{"name":"wilma","home":"bedrock"},
  "kubernetes":{"labels":{"logFormat": "google", ...}}
}
```

### A structuredTypeKey: openshift.labels.<key> の例

以下のスニペットを **ClusterLogForwarder** CR YAML ファイルで使用すると仮定します。

```
outputDefaults:
 elasticsearch:
    structuredTypeKey: openshift.labels.myLabel 1
    structuredTypeName: nologformat
pipelines:
 - name: application-logs
   inputRefs:
   - application
   - audit
   outputRefs:
   - elasticsearch-secure
   - default
   parse: json
   labels:
     myLabel: myValue 2
```

1 OpenShift **myLabel** ラベルによって形成されるキーと値のペアの値を使用します。

2 **myLabel** 要素は、文字列の値 **myValue** を構造化ログレコードに提供します。

この場合は、以下の構造化ログレコードが **app-myValue-write** インデックスに送信されます。

```
{
  "structured":{"name":"fred","home":"bedrock"},
  "openshift":{"labels":{"myLabel": "myValue", ...}}
}
```

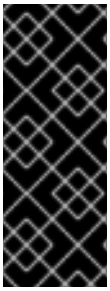
**その他の考慮事項**

- 構造化されたレコードの Elasticsearch **インデックス** は、構造化型の前に app-を、後ろに-write を追加して設定されます。
- 非構造化レコードは、構造化されたインデックスに送信されません。これらは、通常アプリケーション、インフラストラクチャー、または監査インデックスでインデックス化されます。
- 空でない構造化タイプがない場合は、**unstructured** レコードを **structured** フィールドなしで転送します。

過剰なインデックスで Elasticsearch を読み込まないようにすることが重要です。各アプリケーションや namespace ごとに**ではなく**、個別のログ**形式** のみに特定の構造化タイプを使用します。たとえば、ほとんどの Apache アプリケーションは、**LogApache** などの同じ JSON ログ形式と構造化タイプを使用します。

## 12.3. JSON ログの ELASTICSEARCH ログストアへの転送

Elasticsearch ログストアの場合は、JSON ログエントリーが**異なるスキーマに従う場合**、各 JSON スキーマを 1 つの出力定義にグループ化するように **ClusterLogForwarder** カスタムリソース (CR) を設定します。これにより、Elasticsearch はスキーマごとに個別のインデックスを使用します。



**重要**

異なるスキーマを同じインデックスに転送するとタイプの競合やカーディナリティーの問題を引き起こす可能性があるため、データを Elasticsearch ストアに転送する前にこの設定を実行する必要があります。

インデックスが多すぎることが原因のパフォーマンスの問題を回避するには、共通のスキーマに標準化して使用できるスキーマの数を保持することを検討してください。

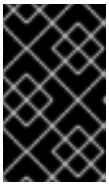
**手順**

1. 以下のスニペットを **ClusterLogForwarder** CR YAML ファイルに追加します。

```
outputDefaults:
 elasticsearch:
   structuredTypeKey: <log record field>
   structuredTypeName: <name>
pipelines:
- inputRefs:
  - application
  outputRefs: default
  parse: json
```

2. オプション: Elasticsearch の JSON ログデータの設定 で前述しているように、**structuredTypeKey** を使用してログレコードフィールドのいずれかを指定します。それ以外の場合は、この行を削除します。

3. オプション: Elasticsearch の JSON ログデータの設定 で前述しているように **structuredTypeName** を使用して **<name>** を指定します。それ以外の場合は、この行を削除します。



**重要**

JSON ログを解析するには、**structuredTypeKey** または **structuredTypeName** か、**structuredTypeKey** と **structuredTypeName** の両方を設定する必要があります。

4. **inputRefs** の場合は、**application**、**infrastructure** または **audit** などのパイプラインを使用して転送するログタイプを指定します。

5. **parse: json** 要素をパイプラインに追加します。

6. CR オブジェクトを作成します。

```
$ oc create -f <file-name>.yaml
```

Red Hat OpenShift Logging Operator は Fluentd Pod を再デプロイします。ただし、再デプロイが完了しない場合は、Fluentd Pod を削除して、強制的に再デプロイされるようにします。

```
$ oc delete pod --selector logging-infra=collector
```

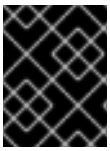
**関連情報**

- ログのサードパーティーシステムへの転送

# 第13章 KUBERNETES イベントの収集および保存

OpenShift Container Platform イベントルーターは、Kubernetes イベントを監視し、それらをロギングシステムによって収集できるようにログに記録する pod です。イベントルーターは手動でデプロイする必要があります。

イベントルーターはすべてのプロジェクトからイベントを収集し、それらを **STDOUT** に書き込みます。次に、コレクターはそれらのイベントを **ClusterLogForwarder** カスタムリソース (CR) で定義されたストアに転送します。



**重要**

イベントルーターは追加の負荷を Fluentd に追加し、処理できる他のログメッセージの数に影響を与える可能性があります。

## 13.1. イベントルーターのデプロイおよび設定

以下の手順を使用してイベントルーターをクラスターにデプロイします。イベントルーターを **openshift-logging** プロジェクトに常にデプロイし、クラスター全体でイベントが収集されるようにする必要があります。

以下のテンプレートオブジェクトは、イベントルーターに必要なサービスアカウント、クラスターロールおよびクラスターロールバインディングを作成します。テンプレートはイベントルーター Pod も設定し、デプロイします。このテンプレートは変更せずに使用するか、デプロイメントオブジェクトの CPU およびメモリー要求を変更できます。

### 前提条件

- サービスアカウントを作成し、クラスターロールバインディングを更新するには、適切なパーミッションが必要です。たとえば、以下のテンプレートを、**cluster-admin** ロールを持つユーザーで実行できます。
- Red Hat OpenShift のロギングサブシステムをインストールする必要があります。

### 手順

1. イベントルーターのテンプレートを作成します。

```
kind: Template
apiVersion: template.openshift.io/v1
metadata:
  name: eventrouter-template
  annotations:
    description: "A pod forwarding kubernetes events to OpenShift Logging stack."
    tags: "events,EFK,logging,cluster-logging"
objects:
  - kind: ServiceAccount (1)
    apiVersion: v1
    metadata:
      name: eventrouter
      namespace: ${NAMESPACE}
  - kind: ClusterRole (2)
    apiVersion: rbac.authorization.k8s.io/v1
    metadata:
```

```
    name: event-reader
  rules:
  - apiGroups: [""]
    resources: ["events"]
    verbs: ["get", "watch", "list"]
- kind: ClusterRoleBinding (3)
  apiVersion: rbac.authorization.k8s.io/v1
  metadata:
    name: event-reader-binding
  subjects:
  - kind: ServiceAccount
    name: eventrouter
    namespace: ${NAMESPACE}
  roleRef:
    kind: ClusterRole
    name: event-reader
- kind: ConfigMap (4)
  apiVersion: v1
  metadata:
    name: eventrouter
    namespace: ${NAMESPACE}
  data:
    config.json: |-
      {
        "sink": "stdout"
      }
- kind: Deployment (5)
  apiVersion: apps/v1
  metadata:
    name: eventrouter
    namespace: ${NAMESPACE}
    labels:
      component: "eventrouter"
      logging-infra: "eventrouter"
      provider: "openshift"
  spec:
    selector:
      matchLabels:
        component: "eventrouter"
        logging-infra: "eventrouter"
        provider: "openshift"
    replicas: 1
    template:
      metadata:
        labels:
          component: "eventrouter"
          logging-infra: "eventrouter"
          provider: "openshift"
        name: eventrouter
      spec:
        serviceAccount: eventrouter
        containers:
          - name: kube-eventrouter
            image: ${IMAGE}
            imagePullPolicy: IfNotPresent
            resources:
```

```
            requests:
              cpu: ${CPU}
              memory: ${MEMORY}
            volumeMounts:
            - name: config-volume
              mountPath: /etc/eventrouter
          volumes:
            - name: config-volume
              configMap:
                name: eventrouter
parameters:
  - name: IMAGE 6
    displayName: Image
    value: "registry.redhat.io/openshift-logging/eventrouter-rhel8:v0.4"
  - name: CPU 7
    displayName: CPU
    value: "100m"
  - name: MEMORY 8
    displayName: Memory
    value: "128Mi"
  - name: NAMESPACE
    displayName: Namespace
    value: "openshift-logging" 9
```

1. イベントルーターの **openshift-logging** プロジェクトでサービスアカウントを作成します。
2. ClusterRole を作成し、クラスター内のイベントを監視します。
3. ClusterRoleBinding を作成し、ClusterRole をサービスアカウントにバインドします。
4. **openshift-logging** プロジェクトで設定マップを作成し、必要な **config.json** ファイルを生成します。
5. **openshift-logging** プロジェクトでデプロイメントを作成し、イベントルーター Pod を生成し、設定します。
6. **v0.4** などのタグで識別されるイメージを指定します。
7. イベントルーター Pod に割り当てる CPU の最小量を指定します。デフォルトは **100m** に設定されます。
8. イベントルーター Pod に割り当てるメモリーの最小量を指定します。デフォルトは **128Mi** に設定されます。
9. オブジェクトをインストールする **openshift-logging** プロジェクトを指定します。

2. 以下のコマンドを使用してテンプレートを処理し、これを適用します。

```
$ oc process -f <templatefile> | oc apply -n openshift-logging -f -
```

以下に例を示します。

```
$ oc process -f eventrouter.yaml | oc apply -n openshift-logging -f -
```

**出力例**

```
serviceaccount/eventrouter created
clusterrole.authorization.openshift.io/event-reader created
clusterrolebinding.authorization.openshift.io/event-reader-binding created
configmap/eventrouter created
deployment.apps/eventrouter created
```

3. イベントルーターが **openshift-logging** プロジェクトにインストールされていることを確認します。

   a. 新規イベントルーター Pod を表示します。

   ```
   $ oc get pods --selector  component=eventrouter -o name -n openshift-logging
   ```

   **出力例**

   ```
   pod/cluster-logging-eventrouter-d649f97c8-qvv8r
   ```

   b. イベントルーターによって収集されるイベントを表示します。

   ```
   $ oc logs <cluster_logging_eventrouter_pod> -n openshift-logging
   ```

   以下に例を示します。

   ```
   $ oc logs cluster-logging-eventrouter-d649f97c8-qvv8r -n openshift-logging
   ```

   **出力例**

   ```
   {"verb":"ADDED","event":{"metadata":{"name":"openshift-service-catalog-controller-
   manager-remover.1632d931e88fcd8f","namespace":"openshift-service-catalog-
   removed","selfLink":"/api/v1/namespaces/openshift-service-catalog-
   removed/events/openshift-service-catalog-controller-manager-
   remover.1632d931e88fcd8f","uid":"787d7b26-3d2f-4017-b0b0-
   420db4ae62c0","resourceVersion":"21399","creationTimestamp":"2020-09-
   08T15:40:26Z"},"involvedObject":{"kind":"Job","namespace":"openshift-service-catalog-
   removed","name":"openshift-service-catalog-controller-manager-
   remover","uid":"fac9f479-4ad5-4a57-8adc-
   cb25d3d9cf8f","apiVersion":"batch/v1","resourceVersion":"21280"},"reason":"Completed","
   message":"Job completed","source":{"component":"job-
   controller"},"firstTimestamp":"2020-09-08T15:40:26Z","lastTimestamp":"2020-09-
   08T15:40:26Z","count":1,"type":"Normal"}}
   ```

   また、Elasticsearch **infra** インデックスを使用してインデックスパターンを作成し、Kibana を使用してイベントを表示することもできます。

# 第14章 OPENSHIFT LOGGING の更新

## 14.1. サポート対象バージョン

バージョンの互換性とサポート情報については、Red Hat OpenShift Container Platform Life Cycle Policy を参照してください。

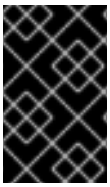
OpenShift Container Platform バージョン 4.6 以前でクラスターロギングから OpenShift Logging 5.x にアップグレードするには、OpenShift Container Platform クラスターをバージョン 4.7 または 4.8 に更新します。次に、以下の Operator を更新します。

- Elasticsearch Operator 4.x から OpenShift Elasticsearch Operator 5.x へ
- Cluster Logging Operator 4.x から Red Hat OpenShift Logging Operator 5.x へ

以前のバージョンの OpenShift Logging から現行バージョンにアップグレードするには、OpenShift Elasticsearch Operator および Red Hat OpenShift Logging Operator を現行バージョンに更新します。

## 14.2. LOGGING を現在のバージョンに更新する

Logging を現在のバージョンに更新するには、OpenShift Elasticsearch Operator および Red Hat OpenShift Logging Operator のサブスクリプションを変更します。



**重要**

Red Hat OpenShift Logging Operator を更新する**前に** OpenShift Elasticsearch Operator を更新する必要があります。また、**両方の** Operator を同じバージョンに更新する必要があります。

Operator を間違った順序で更新すると、Kibana は更新されず、Kibana カスタムリソース (CR) は作成されません。この問題を回避するには、Red Hat OpenShift Logging Operator Pod を削除します。Red Hat OpenShift Logging Operator Pod が再デプロイされると、Kibana CR が作成され、Kibana が再度利用可能になります。

**前提条件**

- OpenShift Container Platform バージョンが 4.7 以降である。
- ロギングステータスは正常です。
  - すべての Pod が **Ready** 状態にある。
  - Elasticsearch クラスターが正常である。
- Elasticsearch および Kibana データのバックアップが作成されている。

**手順**

1. OpenShift Elasticsearch Operator を更新します。
   a. Web コンソールで **Operators → Installed Operators** をクリックします。
   b. **openshift-operators-redhat** プロジェクトを選択します。

c. **OpenShift Elasticsearch Operator** をクリックします。

d. **Subscription → Channel** をクリックします。

e. **Change Subscription Update Channel**ウィンドウで **stable-5.x** を選択し、**Save** をクリックします。

f. 数秒待ってから **Operators → Installed Operators** をクリックします。

g. OpenShift Elasticsearch Operator のバージョンが 5.x.x であることを確認します。

h. **Status** フィールドで **Succeeded** を報告するのを待機します。

2. Red Hat OpenShift Logging Operator を更新します。

a. Web コンソールで **Operators → Installed Operators** をクリックします。

b. **openshift-logging** プロジェクトを選択します。

c. **Red Hat OpenShift Logging Operator** をクリックします。

d. **Subscription → Channel** をクリックします。

e. **Change Subscription Update Channel**ウィンドウで **stable-5.x** を選択し、**Save** をクリックします。

f. 数秒待ってから **Operators → Installed Operators** をクリックします。

g. Red Hat OpenShift Logging Operator のバージョンが 5.y.z であることを確認します。

h. **Status** フィールドで **Succeeded** を報告するのを待機します。

3. ロギングコンポーネントを確認します。

a. すべての Elasticsearch Pod が **Ready** ステータスであることを確認します。

```
$ oc get pod -n openshift-logging --selector component=elasticsearch
```

**出力例**

```
NAME                                            READY   STATUS    RESTARTS   AGE
elasticsearch-cdm-1pbrl44l-1-55b7546f4c-mshhk   2/2     Running   0          31m
elasticsearch-cdm-1pbrl44l-2-5c6d87589f-gx5hk   2/2     Running   0          30m
elasticsearch-cdm-1pbrl44l-3-88df5d47-m45jc     2/2     Running   0          29m
```

b. Elasticsearch クラスターが正常であることを確認します。

```
$ oc exec -n openshift-logging -c elasticsearch elasticsearch-cdm-1pbrl44l-1-
55b7546f4c-mshhk -- health
```

```
{
  "cluster_name" : "elasticsearch",
  "status" : "green",
}
```

c. Elasticsearch cron ジョブが作成されていることを確認します。

```
$ oc project openshift-logging
```

```
$ oc get cronjob
```

```
NAME                     SCHEDULE      SUSPEND   ACTIVE   LAST SCHEDULE   AGE
elasticsearch-im-app     */15 * * * *   False     0        <none>          56s
elasticsearch-im-audit   */15 * * * *   False     0        <none>          56s
elasticsearch-im-infra   */15 * * * *   False     0        <none>          56s
```

d. ログストアが 5.x に更新され、インデックスが **green** であることを確認します。

```
$ oc exec -c elasticsearch <any_es_pod_in_the_cluster> -- indices
```

e. 出力に **app-00000x**、 **infra-00000x**、**audit-00000x**、**.security** インデックス が含まれることを確認します。

**例14.1 緑色のステータスのインデックスを含む出力例**

```
Tue Jun 30 14:30:54 UTC 2020
health status index                                                    uuid                   pri rep
docs.count docs.deleted store.size pri.store.size
green  open   infra-000008                                            bnBvUFEXTWi92z3zWAzieQ   3 1      222195            0       289          144
green  open   infra-000004                                            rtDSzoqsSl6saisSK7Au1Q   3 1      226717            0       297          148
green  open   infra-000012                                            RSf_kUwDSR2xEuKRZMPqZQ   3 1      227623            0       295          147
green  open   .kibana_7                                               1SJdCqlZTPWlIAaOUd78yg   1 1          4            0         0            0
green  open   infra-000010                                            iXwL3bnqTuGEABbUDa6OVw   3 1      248368            0       317          158
green  open   infra-000009                                            YN9EsULWSNaxWeeNvOs0RA   3 1      258799            0       337          168
green  open   infra-000014                                            YP0U6R7FQ_GVQVQZ6Yh9Ig   3 1      223788            0       292          146
green  open   infra-000015                                            JRBbAbEmSMqK5X40df9HbQ   3 1      224371            0       291          145
green  open   .orphaned.2020.06.30                                    n_xQC2dWQzConkvQqei3YA   3 1          9            0         0            0
green  open   infra-000007                                            llkkAVSzSOmosWTSAJM_hg   3 1      228584            0       296          148
green  open   infra-000005                                            d9BoGQdiQASsS3BBFm2iRA   3 1      227987            0       297          148
green  open   infra-000003                                            1-
goREK1QUKlQPAIVkWVaQ   3 1      226719            0       295          147
green  open   .security                                                zeT65uOuRTKZMjg_bbUc1g   1 1          5            0         0            0
green  open   .kibana-377444158_kubeadmin                                                     wvMhDwJkR-
mRZQO84K0gUQ   3 1          1            0         0            0
green  open   infra-000006                                                                   5H-
KBSXGQKiO7hdapDE23g   3 1      226676            0       295          147
green  open   infra-000001                                                                   eH53BQ-
bSxSWR5xYZB6lVg   3 1      341800            0       443          220
green  open   .kibana-6                                               RVp7TemSSemGJcsSUmuf3A   1 1          4            0         0            0
```

```
green  open   infra-000011
J7XWBauWSTe0jnzX02fU6A   3 1      226100          0       293          146
green  open   app-000001
axSAFfONQDmKwatkjPXdtw   3 1      103186          0       126          57
green  open   infra-000016
m9c1iRLtStWSF1GopaRyCg   3 1       13685          0       19           9
green  open   infra-000002                                    Hz6WvINtTvKcQzw-
ewmbYg   3 1       228994          0       296          148
green  open   infra-000013                                    KR9mMFUpQl-
jraYtanyIGw   3 1       228166          0       298          148
green  open   audit-000001
eERqLdLmQOiQDFES1LBATQ   3 1          0          0       0            0
```

f. ログコレクターが以下に更新されていることを確認します。

```
$ oc get ds collector -o json | grep collector
```

g. 出力に **collectort** コンテナーが含まれていることを確認します。

```
"containerName": "collector"
```

h. Kibana CRD を使用してログビジュアライザーが 5.x に更新されていることを確認します。

```
$ oc get kibana kibana -o json
```

i. 出力に **ready** ステータスの Kibana Pod が含まれることを確認します。

**例14.2 準備状態にある Kibana Pod の出力例**

```
[
{
"clusterCondition": {
"kibana-5fdd766ffd-nb2jj": [
{
"lastTransitionTime": "2020-06-30T14:11:07Z",
"reason": "ContainerCreating",
"status": "True",
"type": ""
},
{
"lastTransitionTime": "2020-06-30T14:11:07Z",
"reason": "ContainerCreating",
"status": "True",
"type": ""
}
]
},
"deployment": "kibana",
"pods": {
"failed": [],
"notReady": []
"ready": []
},
"replicaSets": [
```

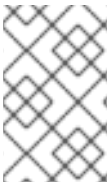
```
    "kibana-5fdd766ffd"
    ],
    "replicas": 1
    }
    ]
```

# 第15章 クラスターダッシュボードの表示

OpenShift Container Platform Web コンソールの **Logging/Elasticsearch Nodes** および **Openshift Logging** ダッシュボードは、 Elasticsearch インスタンスや、問題の発生防止および診断に使用できる個別の Elasticsearch ノードについての詳細情報を表示します。

**OpenShift Logging** ダッシュボードには、クラスターリソース、ガベージコレクション、クラスターのシャード、Fluentd 統計など、クラスターレベルでの Elasticsearch インスタンスについての詳細を表示するチャートが含まれます。

**Logging/Elasticsearch Nodes** ダッシュボードには、Elasticsearch インスタンスの詳細を表示するチャートが含まれます。これらのチャートの多くはノードレベルのものであり、これには、インデックス、シャード、リソースなどの詳細が含まれます。



**注記**

より詳細なデータについては、ダッシュボードの **Grafana UI** リンクをクリックして Grafana ダッシュボードを起動します。Grafana には OpenShift cluster monitoring が同梱されています。

## 15.1. ELASTISEARCH および OPENSHIFT LOGGING ダッシュボードへのアクセス

OpenShift Container Platform Web コンソールで **Logging/Elasticsearch Nodes** および **Openshift Logging** ダッシュボードを表示できます。

### 手順

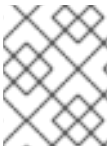
ダッシュボードを起動するには、以下を実行します。

1. OpenShift Container Platform Web コンソールで、 **Observe → Dashboards** をクリックします。

2. **Dashboards** ページで、 **Dashboard** メニューから **Logging/Elasticsearch Nodes** または **Openshift Logging** を選択します。
   **Logging/Elasticsearch Nodes** ダッシュボードの場合は、表示する必要のある Elasticsearch ノードを選択し、データの解像度を設定できます。

   適切なダッシュボードが表示され、データの複数のチャートが表示されます。

3. 必要に応じて、 **Time Range** メニューおよび **Refresh Interval** メニューから、データを表示するさまざまな時間の範囲またはデータのリフレッシュレートを選択します。



**注記**

より詳細なデータについては、 **Grafana UI** リンクをクリックして Grafana ダッシュボードを起動します。

ダッシュボードチャートについての詳細は、About the OpenShift Logging dashboard および About the Logging/Elastisearch Nodes dashboard を参照してください。

## 15.2. OPENSHIFT LOGGING ダッシュボードについて

**OpenShift Logging** ダッシュボードには、クラスターレベルで Elasticsearch インスタンスの詳細を表示するチャートが含まれており、これを使用して問題を診断し、予測できます。

**表15.1 OpenShift Logging チャート**

| メトリック | 説明 |
| --- | --- |
| Elastic Cluster Status (Elastic Cluster のステータス) | Elasticsearch の現行ステータス:<br>● ONLINE: Elasticsearch インスタンスがオンラインであることを示します。<br>● OFFLINE: Elasticsearch インスタンスがオフラインであることを示します。 |
| Elastic Nodes (Elastic ノード) | Elasticsearch インスタンス内の Elasticsearch ノードの合計数。 |
| Elastic Shards (Elastic シャード) | Elasticsearch インスタンス内の Elasticsearch シャードの合計数。 |
| Elastic Documents (Elastic ドキュメント) | Elasticsearch インスタンス内の Elasticsearch ドキュメントの合計数。 |
| Total Index Size on Disk (ディスク上の合計インデックスサイズ) | Elasticsearch インデックスに使用されるディスク容量の合計。 |
| Elastic Pending Tasks (Elastic の保留中のタスク) | インデックスの作成、インデックスのマッピング、シャードの割り当て、シャードの失敗など、完了していない Elasticsearch 変更の合計数。 |
| Elastic JVM GC time (Elastic JVM GC 時間) | JVM がクラスターでの Elasticsearch ガベージコレクション操作の実行に費した時間。 |
| Elastic JVM GC Rate (Elastic JVM GC レート) | JVM が 1 秒ごとにガベージアクティビティーを実行する合計回数。 |
| Elastic Query/Fetch Latency Sum (Elastic クエリー/フェッチのレイテンシーの合計) | ● クエリーレイテンシー: 各 Elasticsearch 検索クエリーの実行に必要な平均時間。<br>● フェッチレイテンシー: 各 Elasticsearch 検索クエリーがデータのフェッチに費す平均時間。<br>通常、フェッチレイテンシーの時間はクエリーレイテンシーよりも短くなります。フェッチレイテンシーが一貫して増加する場合、これはディスクの速度の低下、データの増加、または結果が多すぎる大規模な要求があることを示している可能性があります。 |

| メトリック | 説明 |
| --- | --- |
| Elastic Query Rate (Elastic クエリーレート) | 各 Elasticsearch ノードの 1 秒あたりに Elasticsearch インスタンスに対して実行されたクエリーの合計。 |
| CPU | コンポーネントごとに表示される Elasticsearch、Fluentd、および Kibana によって使用される CPU の量。 |
| Elastic JVM Heap Used (Elastic JVM ヒープの使用) | 使用される JVM メモリーの量。正常なクラスターでは、JVM ガベージコレクションによってメモリーが解放されると、グラフは定期的な低下を示します。 |
| Elasticsearch Disk Usage (Elasticsearch ディスクの使用) | 各 Elasticsearch ノードの Elasticsearch インスタンスによって使用されるディスク容量の合計。 |
| File Descriptors In Use (使用中のファイル記述子) | Elasticsearch、Fluentd、および Kibana によって使用されるファイル記述子の合計数。 |
| FluentD emit count (Fluentd の生成数) | Fluentd デフォルト出力の 1 秒あたりの Fluentd メッセージの合計数およびデフォルト出力の再試行数。 |
| FluentD Buffer Availability (Fluentd バッファーの可用性) | チャンクに使用できる Fluentd バッファーのパーセント。バッファーが一杯になると、Fluentd が受信するログ数を処理できないことを示す可能性があります。 |
| Elastic rx bytes (Elastic rx バイト) | Elasticsearch が FluentD、Elasticsearch ノード、およびその他のソースから受信した合計バイト数。 |
| Elastic Index Failure Rate (Elastic インデックス失敗率) | Elasticsearch インデックスで失敗した 1 秒あたりの合計回数。レートが高い場合は、インデックスに問題があることを示す可能性があります。 |
| FluentD Output Error Rate (Fluentd 出力エラー率) | FluentD がログの出力に失敗する 1 秒あたりの合計回数。 |

## 15.3. LOGGING/ELASTICSEARCH ノードダッシュボードのチャート

**Logging/Elasticsearch Nodes** ダッシュボードには、追加の診断に使用できる Elasticsearch インスタンスの詳細を表示するチャートが含まれます。これらのチャートの多くはノードレベルのものです。

### Elasticsearch ステータス

**Logging/Elasticsearch Nodes** ダッシュボードには、Elasticsearch インスタンスのステータスに関する以下のチャートが含まれます。

**表15.2 Elasticsearch ステータスフィールド**

| メトリック | 説明 |
| --- | --- |
| Cluster status (クラスターステータス) | Elasticsearch の green、yellow、および red ステータスを使用する、選択された期間におけるクラスターの正常性ステータス。<br>● 0: Elasticsearch インスタンスが green ステータスであることを示します。これは、すべてのシャードが割り当てられることを意味します。<br>● 1: Elasticsearch インスタンスが yellow ステータスであることを示します。これは、1 つ以上のシャードのレプリカシャードが割り当てられないことを意味します。<br>● 2: Elasticsearch インスタンスが red ステータスであることを示します。これは、1 つ以上のプライマリーシャードとそのレプリカが割り当てられないことを意味します。 |
| Cluster nodes (クラスターノード) | クラスター内の Elasticsearch ノードの合計数。 |
| Cluster data nodes (クラスターデータノード) | クラスター内の Elasticsearch データノードの数。 |
| Cluster pending tasks (クラスターの保留中のタスク) | 終了しておらず、クラスターキューで待機中のクラスター状態変更の数。たとえば、インデックスの作成、インデックスの削除、シャードの割り当てなどがあります。増加傾向は、クラスターが変更に対応できないことを示します。 |

**Elasticsearch クラスターインデックスシャードのステータス**

各 Elasticsearch インデックスは、永続化されたデータの基本単位である 1 つ以上のシャードの論理グループです。インデックスシャードには、プライマリーシャードとレプリカシャードの 2 つのタイプがあります。ドキュメントがインデックスにインデックス化されると、これはプライマリーシャードのいずれかに保存され、そのシャードのすべてのレプリカにコピーされます。プライマリーシャードの数はインデックスの作成時に指定され、この数はインデックスの有効期間に変更することはできません。レプリカシャードの数はいつでも変更できます。

インデックスシャードは、ライフサイクルフェーズまたはクラスターで発生するイベントに応じて複数の状態に切り替わります。シャードが検索およびインデックス要求を実行できる場合、シャードはアクティブになります。シャードがこれらの要求を実行できない場合、シャードは非アクティブになります。シャードが初期化、再割り当て、未割り当てなどの状態にある場合は、シャードが非アクティブになる可能性があります。

インデックスシャードは、データの物理表現であるインデックスセグメントと呼ばれる数多くの小さな内部ブロックで設定されます。インデックスセグメントは、Lucene が新たにインデックス化されたデータをコミットしたときに作成される比較的小さく、イミュータブルな Lucene インデックスです。Lucene (Elasticsearch によって使用される検索ライブラリー) は、バックグラウンドでインデックスセグメントをより大きなセグメントにマージし、セグメントの合計数を低い状態に維持します。セグメントをマージするプロセスが新規セグメントが作成される速度よりも遅くなる場合は、問題があることを示す可能性があります。

Lucene が検索操作などのデータ操作を実行する場合、Lucene は関連するインデックスのインデックス

セグメントに対して操作を実行します。そのため、各セグメントには、メモリーにロードされ、マップされる特定のデータ構造が含まれます。インデックスマッピングは、セグメントデータ構造で使用されるメモリーに大きく影響を与える可能性があります。

**Logging/Elasticsearch Nodes** ダッシュボードには、Elasticsearch インデックスシャードに関する以下のチャートが含まれます。

**表15.3 Elasticsearch クラスターのシャードステータスのチャート**

| メトリック | 説明 |
|---|---|
| Cluster active shards (クラスターのアクティブシャード) | クラスターにおけるアクティブなプライマリーシャードの数と、レプリカを含むシャードの合計数。シャードの数が大きくなると、クラスターのパフォーマンスが低下し始める可能性があります。 |
| Cluster initializing shards (クラスターの初期化シャード) | クラスターのアクティブではないシャードの数。アクティブではないシャードは、初期化され、別のノードに再配置されるているシャードや、割り当てられていないシャードを指します。通常、クラスターには短期間アクティブではないシャードがあります。長期間にわたってアクティブではないシャードの数が増える場合は、問題があることを示す可能性があります。 |
| Cluster relocating shards (クラスターの再配置シャード) | Elasticsearch が新規ノードに再配置されているシャードの数。Elasticsearch は、ノードでのメモリー使用率が高い場合や新規ノードがクラスターに追加された後などの複数の理由によりノードを再配置します。 |
| Cluster unassigned shards (クラスター未割り当てシャード) | 未割り当てのシャードの数。Elasticsearch シャードは、新規インデックスの追加やノードの障害などの理由で割り当てられない可能性があります。 |

### Elasticsearch ノードメトリック

各 Elasticsearch ノードには、タスクの処理に使用できるリソースの量に制限があります。すべてのリソースが使用中で、Elasticsearch が新規タスクの実行を試行する場合、Elasticsearch は一部のリソースが利用可能になるまでタスクをキューに入れます。

**Logging/Elasticsearch Nodes** ダッシュボードには、選択されたノードのリソース使用状況に関する以下のチャートと Elasticsearch キューで待機中のタスクの数が含まれます。

**表15.4 Elasticsearch ノードのメトリックチャート**

| メトリック | 説明 |
|---|---|
| ThreadPool tasks (ThreadPool タスク) | 個別のキューの待機中のタスクの数 (タスクタイプ別に表示されます)。キュー内のタスクの長期間累積した状態は、ノードリソースの不足やその他の問題があることを示す可能性があります。 |

| メトリック | 説明 |
| --- | --- |
| CPU usage (CPU の使用率) | ホストコンテナーに割り当てられる CPU の合計の割合として、選択した Elasticsearch ノードによって使用される CPU の量。 |
| メモリー使用量 | 選択した Elasticsearch ノードによって使用されるメモリー量。 |
| Disk usage (ディスク使用量) | 選択された Elasticsearch ノードのインデックスデータおよびメタデータに使用されるディスク容量の合計。 |
| Documents indexing rate (ドキュメントインデックス化レート) | ドキュメントが選択された Elasticsearch ノードでインデックス化されるレート。 |
| Indexing latency (インデックス化レイテンシー) | 選択された Elasticsearch ノードでドキュメントをインデックス化するのに必要となる時間。インデックス化レイテンシーは、JVM ヒープメモリーや全体の負荷などの多くの要素による影響を受ける可能性があります。レイテンシーが増加する場合は、インスタンス内のリソース容量が不足していることを示します。 |
| Search rate (検索レート) | 選択された Elasticsearch ノードで実行される検索要求の数。 |
| Search latency (検索レイテンシー) | 選択された Elasticsearch ノードで検索要求を完了するのに必要となる時間。検索レイテンシーは、数多くの要因の影響を受ける可能性があります。レイテンシーが増加する場合は、インスタンス内のリソース容量が不足していることを示します。 |
| Documents count (with replicas)(ドキュメント数 (レプリカ使用)) | 選択された Elasticsearch ノードに保管される Elasticsearch ドキュメントの数。これには、ノードで割り当てられるプライマリーシャードとレプリカシャードの両方に保存されるドキュメントが含まれます。 |
| Documents deleting rate (ドキュメントの削除レート) | 選択された Elasticsearch ノードに割り当てられるいずれかのインデックスシャードから削除される Elasticsearch ドキュメントの数。 |
| Documents merging rate (ドキュメントのマージレート) | 選択された Elasticsearch ノードに割り当てられるインデックスシャードのいずれかでマージされる Elasticsearch ドキュメントの数。 |

**Elasticsearch ノードフィールドデータ**

Fielddata はインデックスの用語のリストを保持する Elasticsearch データ構造であり、JVM ヒープに保持されます。fielddata のビルドはコストのかかる操作であるため、Elasticsearch は fielddata

構造をキャッシュします。Elasticsearch は、基礎となるインデックスセグメントが削除されたり、マージされる場合や、すべての fielddata キャッシュに JVM HEAP メモリーが十分にない場合に、fielddata キャッシュをエビクトできます。

**Logging/Elasticsearch Nodes** ダッシュボードには、Elasticsearch fielddata に関する以下のチャートが含まれます。

**表15.5 Elasticsearch ノードフィールドデータチャート**

| メトリック | 説明 |
|---|---|
| Fielddata memory size (Fielddata メモリーサイズ) | 選択された Elasticsearch ノードの fielddata キャッシュに使用される JVM ヒープの量。 |
| Fielddata evictions (Fielddata エビクション) | 選択された Elasticsearch ノードから削除された fielddata 構造の数。 |

**Elasticsearch ノードのクエリーキャッシュ**

インデックスに保存されているデータが変更されない場合、検索クエリーの結果は Elasticsearch で再利用できるようにノードレベルのクエリーキャッシュにキャッシュされます。

**Logging/Elasticsearch Nodes** ダッシュボードには、Elasticsearch ノードのクエリーキャッシュに関する以下のチャートが含まれます。

**表15.6 Elasticsearch ノードのクエリーチャート**

| メトリック | 説明 |
|---|---|
| Query cache size (クエリーキャッシュサイズ) | 選択された Elasticsearch ノードに割り当てられるすべてのシャードのクエリーキャッシュに使用されるメモリーの合計量。 |
| Query cache evictions (クエリーキャッシュエビクション) | 選択された Elasticsearch ノードでのクエリーキャッシュのエビクション数。 |
| Query cache hits (クエリーキャッシュヒット) | 選択された Elasticsearch ノードでのクエリーキャッシュのヒット数。 |
| Query cache misses (クエリーキャッシュミス) | 選択された Elasticsearch ノードでのクエリーキャッシュのミス数。 |

**Elasticsearch インデックスのスロットリング**

ドキュメントのインデックスを作成する場合、Elasticsearch はデータの物理表現であるインデックスセグメントにドキュメントを保存します。同時に、Elasticsearch はリソースの使用を最適化する方法として、より小さなセグメントをより大きなセグメントに定期的にマージします。インデックス処理がセグメントをマージする機能よりも高速になる場合は、マージプロセスが十分前もって終了せずに、検索やパフォーマンスに関連した問題が生じる可能性があります。この状況を防ぐために、Elasticsearch はインデックスをスロットリングします。通常、インデックスに割り当てられるスレッド数を 1 つのスレッドに減らすことで制限できます。

**Logging/Elasticsearch Nodes** ダッシュボードには、Elasticsearch インデックスのスロットリングに関する以下のチャートが含まれます。

**表15.7 インデックススロットリングチャート**

| メトリック | 説明 |
| --- | --- |
| Indexing throttling (インデックスのスロットリング) | Elasticsearch が選択された Elasticsearch ノードでインデックス操作をスロットリングしている時間。 |
| Merging throttling (マージのスロットリング) | Elasticsearch が選択された Elasticsearch ノードでセグメントのマージ操作をスロットリングしている時間。 |

### ノード JVM ヒープの統計

**Logging/Elasticsearch Nodes** ダッシュボードには、JVM ヒープ操作に関する以下のチャートが含まれます。

**表15.8 JVM ヒープ統計チャート**

| メトリック | 説明 |
| --- | --- |
| Heap used (ヒープの使用) | 選択された Elasticsearch ノードで使用される割り当て済みの JVM ヒープ領域の合計。 |
| GC count (GC 数) | 新旧のガベージコレクションによって、選択された Elasticsearch ノードで実行されてきたガベージコレクション操作の数。 |
| GC time (GC 時間) | JVM が、新旧のガベージコレクションによって選択された Elasticsearch ノードでガベージコレクションを実行してきた時間。 |

# 第16章 ロギングのトラブルシューティング

## 16.1. OPENSHIFT LOGGING ステータスの表示

Red Hat OpenShift Logging Operator のステータスおよびいくつかのロギングサブシステムコンポーネントを表示できます。

### 16.1.1. Red Hat OpenShift Logging Operator のステータス表示

Red Hat OpenShift Logging Operator のステータスを表示できます。

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

**手順**

1. **openshift-logging** プロジェクトに切り替えます。

```
$ oc project openshift-logging
```

2. OpenShift Logging のステータスを表示するには、以下を実行します。
   a. OpenShift Logging のステータスを取得します。

```
$ oc get clusterlogging instance -o yaml
```

**出力例**

```
apiVersion: logging.openshift.io/v1
kind: ClusterLogging

....

status: 1
  collection:
    logs:
      fluentdStatus:
        daemonSet: fluentd 2
        nodes:
          fluentd-2rhqp: ip-10-0-169-13.ec2.internal
          fluentd-6fgjh: ip-10-0-165-244.ec2.internal
          fluentd-6l2ff: ip-10-0-128-218.ec2.internal
          fluentd-54nx5: ip-10-0-139-30.ec2.internal
          fluentd-flpnn: ip-10-0-147-228.ec2.internal
          fluentd-n2frh: ip-10-0-157-45.ec2.internal
        pods:
          failed: []
          notReady: []
          ready:
          - fluentd-2rhqp
          - fluentd-54nx5
```

```
        - fluentd-6fgjh
        - fluentd-6l2ff
        - fluentd-flpnn
        - fluentd-n2frh
    logstore: 3
      elasticsearchStatus:
      - ShardAllocationEnabled:  all
        cluster:
          activePrimaryShards:    5
          activeShards:           5
          initializingShards:     0
          numDataNodes:           1
          numNodes:               1
          pendingTasks:           0
          relocatingShards:       0
          status:                 green
          unassignedShards:       0
        clusterName:              elasticsearch
        nodeConditions:
          elasticsearch-cdm-mkkdys93-1:
        nodeCount:  1
        pods:
          client:
            failed:
            notReady:
            ready:
            - elasticsearch-cdm-mkkdys93-1-7f7c6-mjm7c
          data:
            failed:
            notReady:
            ready:
            - elasticsearch-cdm-mkkdys93-1-7f7c6-mjm7c
          master:
            failed:
            notReady:
            ready:
            - elasticsearch-cdm-mkkdys93-1-7f7c6-mjm7c
    visualization:  4
      kibanaStatus:
      - deployment: kibana
        pods:
          failed: []
          notReady: []
          ready:
          - kibana-7fb4fd4cc9-f2nls
        replicaSets:
        - kibana-7fb4fd4cc9
        replicas: 1
```

1. 出力の **status** スタンザに、クラスターステータスのフィールドが表示されます。
2. Fluentd Pod に関する情報
3. Elasticsearch クラスターの健全性 (**green**、**yellow**、または **red**) などの Elasticsearch Pod に関する情報

4 Kibana Pod に関する情報

#### 16.1.1.1. 状態メッセージ (condition message) のサンプル

以下は、OpenShift Logging インスタンスの **Status.Nodes** セクションからの一部の状態メッセージの例です。

以下のようなステータスメッセージは、ノードが設定された低基準値を超えており、シャードがこのノードに割り当てられないことを示します。

**出力例**

```
nodes:
- conditions:
  - lastTransitionTime: 2019-03-15T15:57:22Z
    message: Disk storage usage for node is 27.5gb (36.74%). Shards will be not
      be allocated on this node.
    reason: Disk Watermark Low
    status: "True"
    type: NodeStorage
  deploymentName: example-elasticsearch-clientdatamaster-0-1
  upgradeStatus: {}
```

以下のようなステータスメッセージは、ノードが設定された高基準値を超えており、シャードが他のノードに移動させられることを示します。

**出力例**

```
nodes:
- conditions:
  - lastTransitionTime: 2019-03-15T16:04:45Z
    message: Disk storage usage for node is 27.5gb (36.74%). Shards will be relocated
      from this node.
    reason: Disk Watermark High
    status: "True"
    type: NodeStorage
  deploymentName: cluster-logging-operator
  upgradeStatus: {}
```

以下のようなステータスメッセージは、CR の Elasticsearch ノードセレクターがクラスターのいずれのノードにも一致しないことを示します。

**出力例**

```
 Elasticsearch Status:
   Shard Allocation Enabled:  shard allocation unknown
   Cluster:
     Active Primary Shards:  0
     Active Shards:          0
     Initializing Shards:    0
     Num Data Nodes:         0
     Num Nodes:              0
     Pending Tasks:          0
     Relocating Shards:      0
```

```
    Status:                 cluster health unknown
    Unassigned Shards:      0
  Cluster Name:             elasticsearch
  Node Conditions:
    elasticsearch-cdm-mkkdys93-1:
      Last Transition Time:  2019-06-26T03:37:32Z
      Message:               0/5 nodes are available: 5 node(s) didn't match node selector.
      Reason:                Unschedulable
      Status:                True
      Type:                  Unschedulable
    elasticsearch-cdm-mkkdys93-2:
  Node Count:  2
  Pods:
    Client:
      Failed:
      Not Ready:
        elasticsearch-cdm-mkkdys93-1-75dd69dccd-f7f49
        elasticsearch-cdm-mkkdys93-2-67c64f5f4c-n58vl
      Ready:
    Data:
      Failed:
      Not Ready:
        elasticsearch-cdm-mkkdys93-1-75dd69dccd-f7f49
        elasticsearch-cdm-mkkdys93-2-67c64f5f4c-n58vl
      Ready:
    Master:
      Failed:
      Not Ready:
        elasticsearch-cdm-mkkdys93-1-75dd69dccd-f7f49
        elasticsearch-cdm-mkkdys93-2-67c64f5f4c-n58vl
      Ready:
```

以下のようなステータスメッセージは、要求された PVC が PV にバインドされないことを示します。

**出力例**

```
  Node Conditions:
    elasticsearch-cdm-mkkdys93-1:
      Last Transition Time:  2019-06-26T03:37:32Z
      Message:               pod has unbound immediate PersistentVolumeClaims (repeated 5 times)
      Reason:                Unschedulable
      Status:                True
      Type:                  Unschedulable
```

以下のようなステータスメッセージは、ノードセレクターがいずれのノードにも一致しないため、Fluentd Pod をスケジュールできないことを示します。

**出力例**

```
Status:
  Collection:
    Logs:
      Fluentd Status:
        Daemon Set:  fluentd
        Nodes:
```

```
Pods:
  Failed:
  Not Ready:
  Ready:
```

### 16.1.2. ロギングサブシステムコンポーネントのステータスの表示

いくつかのロギングサブシステムコンポーネントのステータスを表示できます。

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

**手順**

1. **openshift-logging** プロジェクトに切り替えます。

```
$ oc project openshift-logging
```

2. Red Hat OpenShift 環境のロギングサブシステムのステータスを表示します。

```
$ oc describe deployment cluster-logging-operator
```

**出力例**

```
Name:                   cluster-logging-operator

....

Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable

....

Events:
  Type    Reason             Age   From                    Message
  ----    ------             ----  ----                    -------
  Normal  ScalingReplicaSet  62m   deployment-controller  Scaled up replica set cluster-
logging-operator-574b8987df to 1----
```

3. ロギングサブシステムレプリカセットのステータスを表示します。

   a. レプリカセットの名前を取得します。

**出力例**

```
$ oc get replicaset
```

**出力例**

```
NAME                                DESIRED   CURRENT   READY   AGE
cluster-logging-operator-574b8987df       1         1         1       159m
elasticsearch-cdm-uhr537yu-1-6869694fb    1         1         1       157m
elasticsearch-cdm-uhr537yu-2-857b6d676f   1         1         1       156m
elasticsearch-cdm-uhr537yu-3-5b6fdd8cfd   1         1         1       155m
kibana-5bd5544f87                         1         1         1       157m
```

b. レプリカセットのステータスを取得します。

```
$ oc describe replicaset cluster-logging-operator-574b8987df
```

**出力例**

```
Name:           cluster-logging-operator-574b8987df

....

Replicas:       1 current / 1 desired
Pods Status:    1 Running / 0 Waiting / 0 Succeeded / 0 Failed

....

Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  66m   replicaset-controller  Created pod: cluster-logging-
operator-574b8987df-qjhqv----
```

## 16.2. ELASTICSEARCH ログストアのステータスの表示

OpenShift Elasticsearch Operator のステータスや、数多くの Elasticsearch コンポーネントを表示できます。

### 16.2.1. ログストアのステータスの表示

ログストアのステータスを表示できます。

**前提条件**

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

**手順**

1. **openshift-logging** プロジェクトに切り替えます。

```
$ oc project openshift-logging
```

2. ステータスを表示するには、以下を実行します。

a. ログストアインスタンスの名前を取得します。

```
$ oc get Elasticsearch
```

**出力例**

```
NAME            AGE
elasticsearch   5h9m
```

b. ログストアのステータスを取得します。

```
$ oc get Elasticsearch <Elasticsearch-instance> -o yaml
```

以下に例を示します。

```
$ oc get Elasticsearch elasticsearch -n openshift-logging -o yaml
```

出力には、以下のような情報が含まれます。

**出力例**

```
status: 1
  cluster: 2
    activePrimaryShards: 30
    activeShards: 60
    initializingShards: 0
    numDataNodes: 3
    numNodes: 3
    pendingTasks: 0
    relocatingShards: 0
    status: green
    unassignedShards: 0
  clusterHealth: ""
  conditions: [] 3
  nodes: 4
  - deploymentName: elasticsearch-cdm-zjf34ved-1
    upgradeStatus: {}
  - deploymentName: elasticsearch-cdm-zjf34ved-2
    upgradeStatus: {}
  - deploymentName: elasticsearch-cdm-zjf34ved-3
    upgradeStatus: {}
  pods: 5
    client:
      failed: []
      notReady: []
      ready:
      - elasticsearch-cdm-zjf34ved-1-6d7fbf844f-sn422
      - elasticsearch-cdm-zjf34ved-2-dfbd988bc-qkzjz
      - elasticsearch-cdm-zjf34ved-3-c8f566f7c-t7zkt
    data:
      failed: []
      notReady: []
      ready:
      - elasticsearch-cdm-zjf34ved-1-6d7fbf844f-sn422
      - elasticsearch-cdm-zjf34ved-2-dfbd988bc-qkzjz
      - elasticsearch-cdm-zjf34ved-3-c8f566f7c-t7zkt
    master:
```

```
    failed: []
    notReady: []
    ready:
    - elasticsearch-cdm-zjf34ved-1-6d7fbf844f-sn422
    - elasticsearch-cdm-zjf34ved-2-dfbd988bc-qkzjz
    - elasticsearch-cdm-zjf34ved-3-c8f566f7c-t7zkt
shardAllocationEnabled: all
```

1. 出力の **status** スタンザに、クラスターステータスのフィールドが表示されます。
2. ログストアのステータス:
   - アクティブなプライマリーシャードの数
   - アクティブなシャードの数
   - 初期化されるシャードの数
   - ログストアデータノードの数。
   - ログストアノードの合計数。
   - 保留中のタスクの数。
   - ログストアのステータス: **green**、**red**、**yellow**。
   - 未割り当てのシャードの数。
3. ステータス状態 (ある場合)。ログストアのステータスは、Pod が配置されていない場合にスケジューラーからの理由を示します。以下の状況に関連したイベントが表示されます。
   - コンテナーログストアとプロキシーコンテナーの両方を待機中。
   - ログストアコンテナーとプロキシーコンテナーの両方でコンテナーが終了している。
   - Pod がスケジュール対象外である。また、いくつかの問題に関する条件も示されています。詳細は、**状態メッセージのサンプル** を参照してください。
4. **upgradeStatus** のあるクラスター内のログストアノード。
5. 'failed`、**notReady** または **ready** 状態の下にリスト表示された、クラスター内のログストアクライアント、データ、およびマスター Pod。

#### 16.2.1.1. 状態メッセージ (condition message) のサンプル

以下は、Elasticsearch インスタンスの **Status** セクションからの一部の状態メッセージの例になります。

以下のステータスメッセージは、ノードが設定された低基準値を超えており、シャードがこのノードに割り当てられないことを示します。

```
status:
  nodes:
  - conditions:
```

```
    - lastTransitionTime: 2019-03-15T15:57:22Z
      message: Disk storage usage for node is 27.5gb (36.74%). Shards will be not
        be allocated on this node.
      reason: Disk Watermark Low
      status: "True"
      type: NodeStorage
    deploymentName: example-elasticsearch-cdm-0-1
    upgradeStatus: {}
```

以下のステータスメッセージは、ノードが設定された高基準値を超えており、シャードが他のノードに移動させられることを示します。

```
status:
  nodes:
  - conditions:
    - lastTransitionTime: 2019-03-15T16:04:45Z
      message: Disk storage usage for node is 27.5gb (36.74%). Shards will be relocated
        from this node.
      reason: Disk Watermark High
      status: "True"
      type: NodeStorage
    deploymentName: example-elasticsearch-cdm-0-1
    upgradeStatus: {}
```

以下のステータスメッセージは、CR のログストアノードセレクターがクラスターのいずれのノードにも一致しないことを示します。

```
status:
    nodes:
    - conditions:
      - lastTransitionTime: 2019-04-10T02:26:24Z
        message: '0/8 nodes are available: 8 node(s) didn''t match node selector.'
        reason: Unschedulable
        status: "True"
        type: Unschedulable
```

以下のステータスメッセージは、ログストア CR が存在しない 永続ボリューム要求 (PVC) を使用することを示します。

```
status:
    nodes:
    - conditions:
      - last Transition Time:  2019-04-10T05:55:51Z
        message:               pod has unbound immediate PersistentVolumeClaims (repeated 5 times)
        reason:              Unschedulable
        status:              True
        type:                Unschedulable
```

以下のステータスメッセージは、ログストアクラスターには冗長性ポリシーをサポートするための十分なノードがないことを示します。

```
status:
  clusterHealth: ""
  conditions:
```

```
  - lastTransitionTime: 2019-04-17T20:01:31Z
    message: Wrong RedundancyPolicy selected. Choose different RedundancyPolicy or
      add more nodes with data roles
    reason: Invalid Settings
    status: "True"
    type: InvalidRedundancy
```

このステータスメッセージは、クラスターにコントロールプレーンノードが多すぎることを示しています。

```
status:
  clusterHealth: green
  conditions:
    - lastTransitionTime: '2019-04-17T20:12:34Z'
      message: >-
        Invalid master nodes count. Please ensure there are no more than 3 total
        nodes with master roles
      reason: Invalid Settings
      status: 'True'
      type: InvalidMasters
```

以下のステータスメッセージは、加えようとした変更が Elasticsearch ストレージでサポートされないことを示します。

以下に例を示します。

```
status:
  clusterHealth: green
  conditions:
    - lastTransitionTime: "2021-05-07T01:05:13Z"
      message: Changing the storage structure for a custom resource is not supported
      reason: StorageStructureChangeIgnored
      status: 'True'
      type: StorageStructureChangeIgnored
```

**reason** および **type** フィールドは、サポート対象外の変更のタイプを指定します。

**StorageClassNameChangeIgnored**

ストレージクラス名の変更がサポートされていません。

**StorageSizeChangeIgnored**

ストレージサイズの変更がサポートされていません。

**StorageStructureChangeIgnored**

一時ストレージと永続ストレージ構造間での変更がサポートされていません。

**重要**

**ClusterLogging** カスタムリソース (CR) を一時ストレージから永続ストレージに切り替えるように設定する場合に、OpenShift Elasticsearch Operator は永続ボリューム要求 (PVC) を作成しますが、永続ボリューム (PV) は作成されません。**StorageStructureChangeIgnored** ステータスを削除するには、**ClusterLogging** CR への変更を元に戻し、PVC を削除する必要があります。

### 16.2.2. ログストアコンポーネントのステータスの表示

数多くのログストアコンポーネントのステータスを表示できます。

**Elasticsearch インデックス**

Elasticsearch インデックスのステータスを表示できます。

1. Elasticsearch Pod の名前を取得します。

```
$ oc get pods --selector component=elasticsearch -o name
```

**出力例**

```
pod/elasticsearch-cdm-1godmszn-1-6f8495-vp4lw
pod/elasticsearch-cdm-1godmszn-2-5769cf-9ms2n
pod/elasticsearch-cdm-1godmszn-3-f66f7d-zqkz7
```

2. インデックスのステータスを取得します。

```
$ oc exec elasticsearch-cdm-4vjor49p-2-6d4d7db474-q2w7z -- indices
```

**出力例**

```
Defaulting container name to elasticsearch.
Use 'oc describe pod/elasticsearch-cdm-4vjor49p-2-6d4d7db474-q2w7z -n openshift-
logging' to see all of the containers in this pod.

green  open   infra-000002                                    S4QANnf1QP6NgCegfnrnbQ
3   1    119926            0       157           78
green  open   audit-000001                                    8_EQx77iQCSTzFOXtxRqFw
3   1          0            0         0            0
green  open   .security                                       iDjscH7aSUGhIdq0LheLBQ   1
1          5            0         0            0
green  open   .kibana_-377444158_kubeadmin
yBywZ9GfSrKebz5gWBZbjw   3   1          1            0         0            0
green  open   infra-000001                                    z6Dpe__ORgiopEpW6Yl44A
3   1    871000            0       874           436
green  open   app-000001                                      hIrazQCeSISewG3c2VIvsQ
3   1      2453            0         3             1
green  open   .kibana_1                                       JCitcBMSQxKOvIq6iQW6wg
1   1          0            0         0            0
green  open   .kibana_-1595131456_user1                       gIYFIEGRRe-
ka0W3okS-mQ   3   1          1            0         0            0
```

**ログストア Pod**

ログストアをホストする Pod のステータスを表示できます。

1. Pod の名前を取得します。

```
$ oc get pods --selector component=elasticsearch -o name
```

**出力例**

```
pod/elasticsearch-cdm-1godmszn-1-6f8495-vp4lw
pod/elasticsearch-cdm-1godmszn-2-5769cf-9ms2n
pod/elasticsearch-cdm-1godmszn-3-f66f7d-zqkz7
```

2. Pod のステータスを取得します。

```
$ oc describe pod elasticsearch-cdm-1godmszn-1-6f8495-vp4lw
```

出力には、以下のようなステータス情報が含まれます。

**出力例**

```
....
Status:             Running

....

Containers:
  elasticsearch:
    Container ID:   cri-o://b7d44e0a9ea486e27f47763f5bb4c39dfd2
    State:          Running
      Started:      Mon, 08 Jun 2020 10:17:56 -0400
    Ready:          True
    Restart Count:  0
    Readiness:  exec [/usr/share/elasticsearch/probe/readiness.sh] delay=10s timeout=30s
period=5s #success=1 #failure=3

....

  proxy:
    Container ID:  cri-
o://3f77032abaddbb1652c116278652908dc01860320b8a4e741d06894b2f8f9aa1
    State:          Running
      Started:      Mon, 08 Jun 2020 10:18:38 -0400
    Ready:          True
    Restart Count:  0

....

Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True

....

Events:         <none>
```

**ログストレージ Pod デプロイメント設定**

ログストアのデプロイメント設定のステータスを表示できます。

1. デプロイメント設定の名前を取得します。

```
$ oc get deployment --selector component=elasticsearch -o name
```

**出力例**

```
deployment.extensions/elasticsearch-cdm-1gon-1
deployment.extensions/elasticsearch-cdm-1gon-2
deployment.extensions/elasticsearch-cdm-1gon-3
```

2. デプロイメント設定のステータスを取得します。

```
$ oc describe deployment elasticsearch-cdm-1gon-1
```

出力には、以下のようなステータス情報が含まれます。

**出力例**

```
....
  Containers:
   elasticsearch:
    Image:      registry.redhat.io/openshift-logging/elasticsearch6-rhel8
    Readiness:  exec [/usr/share/elasticsearch/probe/readiness.sh] delay=10s timeout=30s
period=5s #success=1 #failure=3

....

Conditions:
  Type           Status   Reason
  ----           ------   ------
  Progressing    Unknown  DeploymentPaused
  Available      True     MinimumReplicasAvailable

....

Events:          <none>
```

**ログストアのレプリカセット**

ログストアのレプリカセットのステータスを表示できます。

1. レプリカセットの名前を取得します。

```
$ oc get replicaSet --selector component=elasticsearch -o name

replicaset.extensions/elasticsearch-cdm-1gon-1-6f8495
replicaset.extensions/elasticsearch-cdm-1gon-2-5769cf
replicaset.extensions/elasticsearch-cdm-1gon-3-f66f7d
```

2. レプリカセットのステータスを取得します。

```
$ oc describe replicaSet elasticsearch-cdm-1gon-1-6f8495
```

出力には、以下のようなステータス情報が含まれます。

**出力例**

```
....
  Containers:
   elasticsearch:
    Image:      registry.redhat.io/openshift-logging/elasticsearch6-
rhel8@sha256:4265742c7cdd85359140e2d7d703e4311b6497eec7676957f455d6908e7b1
c25
    Readiness:  exec [/usr/share/elasticsearch/probe/readiness.sh] delay=10s timeout=30s
period=5s #success=1 #failure=3

....

Events:         <none>
```

### 16.2.3. Elasticsearch クラスターのステータス

OpenShift Container Platform Web コンソールの**Observe**セクションにある Grafana ダッシュボードには、Elasticsearch クラスターのステータスが表示されます。

OpenShift Elasticsearch クラスターのステータスを取得するには、OpenShift Container Platform Web コンソールの**Observe**セクションにある Grafana ダッシュボード **<cluster_url>/monitoring/dashboards/grafana-dashboard-cluster-logging** にアクセスします。

#### Elasticsearch ステータスフィールド

**eo_elasticsearch_cr_cluster_management_state**

Elasticsearch クラスターがマネージドか、マネージド外かをを示します。以下に例を示します。

```
eo_elasticsearch_cr_cluster_management_state{state="managed"} 1
eo_elasticsearch_cr_cluster_management_state{state="unmanaged"} 0
```

**eo_elasticsearch_cr_restart_total**

Elasticsearch ノードが証明書の再起動、ローリング再起動、またはスケジュールされた再起動など、再起動した回数を示します。以下に例を示します。

```
eo_elasticsearch_cr_restart_total{reason="cert_restart"} 1
eo_elasticsearch_cr_restart_total{reason="rolling_restart"} 1
eo_elasticsearch_cr_restart_total{reason="scheduled_restart"} 3
```

**es_index_namespaces_total**

Elasticsearch インデックス namespace の総数を表示します。以下に例を示します。

```
Total number of Namespaces.
es_index_namespaces_total 5
```

**es_index_document_count**

各 namespace のレコード数を表示します。以下に例を示します。

```
es_index_document_count{namespace="namespace_1"} 25
es_index_document_count{namespace="namespace_2"} 10
es_index_document_count{namespace="namespace_3"} 5
```

### Secret Elasticsearch フィールドが見つからないか、空というメッセージ

Elasticsearch に **admin-cert**、**admin-key**、**logging-es.crt**、または **logging-es.key** ファイルがない場合、ダッシュボードには次の例のようなステータスメッセージが表示されます。

```
message": "Secret \"elasticsearch\" fields are either missing or empty: [admin-cert, admin-key,
logging-es.crt, logging-es.key]",
"reason": "Missing Required Secrets",
```

## 16.3. ロギングサブシステムアラートについて

ロギングコレクターのアラートはすべて、OpenShift Container Platform Web コンソールの Alerting UI に一覧表示されます。

### 16.3.1. ロギングコレクターアラートの表示

アラートは、OpenShift Container Platform Web コンソールの、Alerting UI の **Alerts** タブに表示されます。アラートは以下の状態のいずれかになります。

- **Firing**アラートの状態はタイムアウトの期間は true になります。Firing アラートの末尾の **Option** メニューをクリックし、詳細情報を表示するか、アラートを非通知 (silence) にします。
- **Pending**: このアラート状態は現時点で true ですが、タイムアウトに達していません。
- **Not Firing**アラートは現時点でトリガーされていません。

#### 手順

ロギングサブシステムおよびその他の OpenShift Container Platform アラートを表示するには:

1. OpenShift Container Platform コンソールで **Observe → Alerting** の順にクリックします。
2. **Alerts** タブをクリックします。選択したフィルターに基づいてアラートがリスト表示されます。

#### 関連情報

- Alerting UI の詳細は、アラートの管理 を参照してください。

### 16.3.2. ロギングコレクターのアラートについて

以下のアラートはロギングコレクターによって生成されます。これらのアラートは、OpenShift Container Platform Web コンソールの Alerting UI の **Alerts** ページで表示できます。

**表16.1 Fluentd Prometheus アラート**

| アラート | メッセージ | 説明 | 重大度 |
|---|---|---|---|
| **FluentDHighErrorRate** | **<value> of records have resulted in an error by fluentd <instance>.** | FluentD 出力エラーの数は、デフォルトでは直前の 15 分間で 10 分を超えます。 | Warning |

| アラート | メッセージ | 説明 | 重大度 |
| --- | --- | --- | --- |
| **FluentdNodeDown** | **Prometheus could not scrape fluentd <instance> for more than 10m.** | Fluentd は Prometheus が特定の Fluentd インスタンスを収集できなかったことを報告します。 | Critical |
| **FluentdQueueLengthIncreasing** | **In the last 12h, fluentd <instance> buffer queue length constantly increased more than 1.Current value is <value>.** | Fluentd はキューサイズが増加していることを報告しています。 | Critical |
| **FluentDVeryHighErrorRate** | **<value> of records have resulted in an error by fluentd <instance>.** | FluentD 出力エラーの数は非常に高くなります。デフォルトでは、直前の 15 分間で 25 を超えます。 | Critical |

### 16.3.3. Elasticsearch アラートルール

これらのアラートルールを Prometheus に表示できます。

**表16.2 アラートルール**

| アラート | 説明 | 重大度 |
| --- | --- | --- |
| **ElasticsearchClusterNotHealthy** | クラスターのヘルスステータスは少なくとも 2m の間 RED になります。クラスターは書き込みを受け入れず、シャードが見つからない可能性があるか、マスターノードがまだ選択されていません。 | Critical |
| **ElasticsearchClusterNotHealthy** | クラスターのヘルスステータスは少なくとも 20m の間 YELLOW になります。一部のシャードレプリカは割り当てられません。 | Warning |
| **ElasticsearchDiskSpaceRunningLow** | クラスターでは、次の 6 時間以内にディスク領域が不足することが予想されます。 | Critical |
| **ElasticsearchHighFileDescriptorUsage** | クラスターでは、次の 1 時間以内にファイル記述子が不足することが予想されます。 | Warning |
| **ElasticsearchJVMHeapUseHigh** | 指定されたノードでの JVM ヒープの使用率が高くなっています。 | アラート |

| アラート | 説明 | 重大度 |
| --- | --- | --- |
| **ElasticsearchNodeDiskWatermarkReached** | 指定されたノードは、ディスクの空き容量が少ないために低基準値に達しています。シャードをこのノードに割り当てることはできません。ノードにディスク領域を追加することを検討する必要があります。 | Info |
| **ElasticsearchNodeDiskWatermarkReached** | 指定されたノードは、ディスクの空き容量が少ないために高基準値に達しています。一部のシャードは可能な場合に別のノードに再度割り当てられる可能性があります。ノードにディスク領域が追加されるか、このノードに割り当てられる古いインデックスをドロップします。 | Warning |
| **ElasticsearchNodeDiskWatermarkReached** | 指定されたノードは、ディスクの空き容量が少ないために高基準値に達しています。このノードにシャードが割り当てられるすべてのインデックスは、読み取り専用ブロックになります。インデックスブロックは、ディスクの使用状況が高基準値を下回る場合に手動で解放される必要があります。 | Critical |
| **ElasticsearchJVMHeapUseHigh** | 指定されたノードの JVM ヒープの使用率が高すぎます。 | アラート |
| **ElasticsearchWriteRequestsRejectionJumps** | Elasticsearch では、指定されたノードで書き込み拒否が増加しています。このノードはインデックスの速度に追い付いていない可能性があります。 | Warning |
| **AggregatedLoggingSystemCPUHigh** | 指定されたノードのシステムで使用される CPU が高すぎます。 | アラート |
| **ElasticsearchProcessCPUHigh** | 指定されたノードで Elasticsearch によって使用される CPU が高すぎます。 | アラート |

## 16.4. RED HAT サポート用のロギングデータの収集

サポートケースを作成する際、ご使用のクラスターのデバッグ情報を Red Hat サポートに提供していただくと Red Hat のサポートに役立ちます。

**must-gather ツール** を使用すると、プロジェクトレベルのリソース、クラスターレベルのリソース、および各ロギングシステムコンポーネントについての診断情報を収集できます。

迅速なサポートを得るには、OpenShift Container Platform と OpenShift Logging の両方の診断情報を提供してください。

**注記**

**hack/logging-dump.sh** スクリプトは使用しないでください。このスクリプトはサポートされなくなり、データを収集しません。

### 16.4.1. must-gather ツールについて

**oc adm must-gather** CLI コマンドは、問題のデバッグに必要となる可能性のあるクラスターからの情報を収集します。

ロギングサブシステムの場合、**must-gather** は次の情報を収集します。

- プロジェクトレベルの Pod、設定マップ、サービスアカウント、ロール、ロールバインディングおよびイベントを含むプロジェクトレベルのリソース
- クラスターレベルでのノード、ロール、およびロールバインディングを含むクラスターレベルのリソース
- ログコレクター、ログストア、およびログビジュアライザーなどの **openshift-logging** および **openshift-operators-redhat** namespace の OpenShift Logging リソース

**oc adm must-gather** を実行すると、新しい Pod がクラスターに作成されます。データは Pod で収集され、**must-gather.local** で始まる新規ディレクトリーに保存されます。このディレクトリーは、現行の作業ディレクトリーに作成されます。

### 16.4.2. 前提条件

- ロギングサブシステムと Elasticsearch をインストールする必要があります。

### 16.4.3. OpenShift Logging データの収集

**oc adm must-gather** CLI コマンドを使用して、ロギングサブシステムに関する情報を収集できます。

**手順**

**must-gather** でロギングサブシステム情報を収集するには、以下を実行します。

1. **must-gather** 情報を保存する必要のあるディレクトリーに移動します。
2. OpenShift Logging イメージに対して **oc adm must-gather** コマンドを実行します。

```
$ oc adm must-gather --image=$(oc -n openshift-logging get deployment.apps/cluster-logging-operator -o jsonpath='{.spec.template.spec.containers[?(@.name == "cluster-logging-operator")].image}')
```

   **must-gather** ツールは、現行ディレクトリー内の **must-gather.local** で始まる新規ディレクトリーを作成します。例: **must-gather.local.4157245944708210408**

3. 作成された **must-gather** ディレクトリーから圧縮ファイルを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar -cvaf must-gather.tar.gz must-gather.local.4157245944708210408
```

4. 圧縮ファイルを Red Hat カスタマーポータル で作成したサポートケースに添付します。

## 16.5. CRITICAL ALERTS のトラブルシューティング

### 16.5.1. Elasticsearch クラスターの正常性が赤である

1 つ以上のプライマリーシャードとそのレプリカがノードに割り当てられません。

**トラブルシューティング**

1. Elasticsearch クラスターの正常性を確認し、クラスターの **ステータス** が赤であることを確認します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- health
```

2. クラスターにに参加したノードをリスト表示します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=_cat/nodes?v
```

3. Elasticsearch Pod をリスト表示し、この Pod を直前の手順のコマンド出力にあるノードと比較します。

```
oc -n openshift-logging get pods -l component=elasticsearch
```

4. 一部の Elasticsearch ノードがクラスターに参加していない場合は、以下の手順を実行します。

   a. Elasticsearch に選ばれたコントロールプレーンノードがあることを確認します。

   ```
   oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
   query=_cat/master?v
   ```

   b. 選ばれたコントロールプレーンノードの Pod ログで問題を確認します。

   ```
   oc logs <elasticsearch_master_pod_name> -c elasticsearch -n openshift-logging
   ```

   c. 問題がないか、クラスターに参加していないノードのログを確認します。

   ```
   oc logs <elasticsearch_node_name> -c elasticsearch -n openshift-logging
   ```

5. 全ノードがクラスターに参加している場合は、以下の手順を実行し、クラスターがリカバリープロセスにあるかどうかを確認します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=_cat/recovery?active_only=true
```

   コマンドの出力がない場合は、リカバリープロセスが保留中のタスクによって遅延しているか、停止している可能性があります。

6. 保留中のタスクがあるかどうかを確認します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- health |grep
number_of_pending_tasks
```

7. 保留中のタスクがある場合は、そのステータスを監視します。
   そのステータスが変化し、クラスターがリカバリー中の場合は、そのまま待機します。リカバリー時間は、クラスターのサイズや他の要素により異なります。

   保留中のタスクのステータスが変更されない場合は、リカバリーが停止していることがわかります。

8. リカバリーが停止しているようであれば、**cluster.routing.allocation.enable** が **none** に設定されているかどうかを確認します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=_cluster/settings?pretty
```

9. **cluster.routing.allocation.enable** が **none** に設定されている場合は、これを **all** に設定します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=_cluster/settings?pretty -X PUT -d '{"persistent":
{"cluster.routing.allocation.enable":"all"}}'
```

10. どのインデックスが赤のままかを確認します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=_cat/indices?v
```

11. インデックスがまだ赤い場合は、以下の手順を実行して赤のインデックスをなくします。

    a. キャッシュをクリアします。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=<elasticsearch_index_name>/_cache/clear?pretty
```

    b. 最大割り当ての再試行回数を増やします。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=<elasticsearch_index_name>/_settings?pretty -X PUT -d
'{"index.allocation.max_retries":10}'
```

    c. スクロールアイテムをすべて削除します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=_search/scroll/_all -X DELETE
```

    d. タイムアウトを増やします。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=<elasticsearch_index_name>/_settings?pretty -X PUT -d
'{"index.unassigned.node_left.delayed_timeout":"10m"}'
```

12. 前述の手順で赤色のインデックスがなくならない場合は、インデックスを個別に削除します。

    a. 赤色のインデックスの名前を特定します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=_cat/indices?v
```

    b. 赤色のインデックスを削除します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=<elasticsearch_red_index_name> -X DELETE
```

13. 赤色のインデックスがなく、クラスターのステータスが赤の場合は、データノードで継続的に過剰な処理負荷がかかっていないかを確認します。

    a. Elasticsearch JVM ヒープの使用量が多いかどうかを確認します。

    ```
    oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --query=_nodes/stats?pretty
    ```

    コマンド出力で **node_name.jvm.mem.heap_used_percent** フィールドを確認し、JVM ヒープ使用量を判別します。

    b. 使用量が多い CPU がないかを確認します。

**関連情報**

- Elasticsearch で "Free up or increase disk space" と検索し、クラスターのステータスが赤または黄色の問題を修正します。

### 16.5.2. Elasticsearch クラスターの正常性が黄色である

1 つ以上のプライマリーシャードのレプリカシャードがノードに割り当てられません。

**トラブルシューティング**

1. **ClusterLogging** CR で **nodeCount** を調整してノード数を増やします。

**関連情報**

- クラスターロギングカスタムリソースについて
- ログストアの永続ストレージの設定
- Elasticsearch で "Free up or increase disk space" と検索し、クラスターのステータスが赤または黄色の問題を修正します。

### 16.5.3. Elasticsearch Node Disk Low Watermark Reached (Elasticsearch ノードのディスクで低い基準値に達する)

Elasticsearch で、低基準値に到達した ノードにシャードが割り当てられません。

**トラブルシューティング**

1. Elasticsearch のデプロイ先のノードを特定します。

    ```
    oc -n openshift-logging get po -o wide
    ```

2. **未割り当てのシャード** があるかどうかを確認します。

    ```
    oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --query=_cluster/health?pretty | grep unassigned_shards
    ```

3. 未割り当てのシャードがある場合は、各ノードのディスク領域を確認します。

```
for pod in `oc -n openshift-logging get po -l component=elasticsearch -o
jsonpath='{.items[*].metadata.name}'`; do echo $pod; oc -n openshift-logging exec -c
elasticsearch $pod -- df -h /elasticsearch/persistent; done
```

4. **nodes.node_name.fs** フィールドで、対象のノードの空きディスク領域を確認します。使用済みディスクの割合が 85% を超える場合は、ノードは低基準値を超えており、シャードがこのノードに割り当てられなくなります。

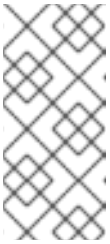
5. すべてのノードでディスク領域を増やしてみてください。

6. ディスク領域を増やせない場合は、新しいデータノードをクラスターに追加してみてください。

7. 新規データノードの追加に問題がある場合は、クラスターの冗長性ポリシー総数を減らします。

   a. 現在の **redundancyPolicy** を確認します。

   ```
   oc -n openshift-logging get es elasticsearch -o jsonpath='{.spec.redundancyPolicy}'
   ```

   **注記**

   **ClusterLogging** CR を使用している場合は、以下を入力します。

   ```
   oc -n openshift-logging get cl -o
   jsonpath='{.items[*].spec.logStore.elasticsearch.redundancyPolicy}'
   ```

   b. クラスター **redundancyPolicy** が **SingleRedundancy** よりも大きい場合は、**SingleRedundancy** に設定し、この変更を保存します。

8. 前述の手順で問題が解決しない場合は、古いインデックスを削除します。

   a. Elasticsearch の全インデックスのステータスを確認します。

   ```
   oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- indices
   ```

   b. 古いインデックスで削除できるものを特定します。

   c. インデックスを削除します。

   ```
   oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
   query=<elasticsearch_index_name> -X DELETE
   ```

**関連情報**

- クラスターロギングカスタムリソースについて の「**ClusterLogging** カスタムリソース (CR) のサンプル」で「redundancyPolicy」を参照します。

### 16.5.4. Elasticsearch Node Disk High Watermark Reached (Elasticsearch ノードのディスクで高い基準値に達する)

Elasticsearch が 高基準値に達した ノードからシャードを移動しようとします。

**トラブルシューティング**

1. Elasticsearch のデプロイ先のノードを特定します。

```
oc -n openshift-logging get po -o wide
```

2. 各ノードのディスク容量を確認します。

```
for pod in `oc -n openshift-logging get po -l component=elasticsearch -o
jsonpath='{.items[*].metadata.name}'`; do echo $pod; oc -n openshift-logging exec -c
elasticsearch $pod -- df -h /elasticsearch/persistent; done
```

3. クラスターがリバランスされているかどうかを確認します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=_cluster/health?pretty | grep relocating_shards
```

コマンドの出力でシャードの再配置が表示される場合は、高い基準値を超過しています。高い基準値のデフォルト値は 90% です。

基準値のしきい値上限を超えておらず、ディスクの使用量が少ないノードに、シャードを移動します。

4. シャードを特定ノードに割り当てるには、領域の一部を解放します。

5. すべてのノードでディスク領域を増やしてみてください。

6. ディスク領域を増やせない場合は、新しいデータノードをクラスターに追加してみてください。

7. 新規データノードの追加に問題がある場合は、クラスターの冗長性ポリシー総数を減らします。

   a. 現在の **redundancyPolicy** を確認します。

```
oc -n openshift-logging get es elasticsearch -o jsonpath='{.spec.redundancyPolicy}'
```

**注記**

**ClusterLogging** CR を使用している場合は、以下を入力します。

```
oc -n openshift-logging get cl -o
jsonpath='{.items[*].spec.logStore.elasticsearch.redundancyPolicy}'
```

   b. クラスター **redundancyPolicy** が **SingleRedundancy** よりも大きい場合は、**SingleRedundancy** に設定し、この変更を保存します。

8. 前述の手順で問題が解決しない場合は、古いインデックスを削除します。

   a. Elasticsearch の全インデックスのステータスを確認します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- indices
```

   b. 古いインデックスで削除できるものを特定します。

c. インデックスを削除します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=<elasticsearch_index_name> -X DELETE
```

**関連情報**

- クラスターロギングカスタムリソースについて の「**ClusterLogging** カスタムリソース (CR) のサンプル」で「redundancyPolicy」を参照します。

### 16.5.5. Elasticsearch Node Disk Flood Watermark Reached (Elasticsearch ノードのディスクがいっぱいの基準値に達する)

Elasticsearch は、両条件が含まれるすべてのインデックスに対して読み取り専用のインデックスブロックを強制的に適用します。

- 1 つ以上のシャードがノードに割り当てられます。
- 1 つ以上のディスクが いっぱいの段階 を超えています。

**トラブルシューティング**

1. Elasticsearch ノードのディスク領域を確認します。

```
for pod in `oc -n openshift-logging get po -l component=elasticsearch -o
jsonpath='{.items[*].metadata.name}'`; do echo $pod; oc -n openshift-logging exec -c
elasticsearch $pod -- df -h /elasticsearch/persistent; done
```

   **nodes.node_name.fs** フィールドで、対象のノードの空きディスク領域を確認します。

2. 使用済みディスクの割合が 95% を超える場合は、ノードがいっぱいの基準値が越えたことを意味します。この特定のノードに割り当てられたシャードへの書き込みは、ブロックされます。

3. すべてのノードでディスク領域を増やしてみてください。

4. ディスク領域を増やせない場合は、新しいデータノードをクラスターに追加してみてください。

5. 新規データノードの追加に問題がある場合は、クラスターの冗長性ポリシー総数を減らします。

   a. 現在の **redundancyPolicy** を確認します。

```
oc -n openshift-logging get es elasticsearch -o jsonpath='{.spec.redundancyPolicy}'
```

**注記**

**ClusterLogging** CR を使用している場合は、以下を入力します。

```
oc -n openshift-logging get cl -o
jsonpath='{.items[*].spec.logStore.elasticsearch.redundancyPolicy}'
```

b. クラスター **redundancyPolicy** が **SingleRedundancy** よりも大きい場合は、**SingleRedundancy** に設定し、この変更を保存します。

6. 前述の手順で問題が解決しない場合は、古いインデックスを削除します。

a. Elasticsearch の全インデックスのステータスを確認します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- indices
```

b. 古いインデックスで削除できるものを特定します。

c. インデックスを削除します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=<elasticsearch_index_name> -X DELETE
```

7. ディスク使用領域が 90% 未満になるまで、このままディスク領域を解放して監視します。次に、この特定のノードへの書き込み禁止を解除します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=_all/_settings?pretty -X PUT -d '{"index.blocks.read_only_allow_delete": null}'
```

**関連情報**

- クラスターロギングカスタムリソースについて の「**ClusterLogging** カスタムリソース (CR) のサンプル」で「redundancyPolicy」を参照します。

### 16.5.6. Elasticsearch JVM ヒープの使用量が高い

Elasticsearch ノードで使用済みの JVM ヒープメモリーが 75% を超えます。

#### トラブルシューティング

ヒープサイズを増やす ことを検討してください。

### 16.5.7. 集計ロギングシステムの CPU が高い

ノード上のシステムの CPU 使用量が高くなります。

#### トラブルシューティング

クラスターノードの CPU を確認します。ノードへ割り当てる CPU リソースを増やすことを検討してください。

### 16.5.8. Elasticsearch プロセスの CPU が高い

ノードでの Elasticsearch プロセスの CPU 使用量が高くなります。

#### トラブルシューティング

クラスターノードの CPU を確認します。ノードへ割り当てる CPU リソースを増やすことを検討してください。

### 16.5.9. Elasticsearch ディスク領域が不足している

Elasticsearch クラスターは、現在のディスク使用量に基づいて次の 6 時間以内にディスク領域が不足することが予想します。

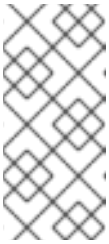
**トラブルシューティング**

1. Elasticsearch ノードのディスク領域を取得します。

```
for pod in `oc -n openshift-logging get po -l component=elasticsearch -o
jsonpath='{.items[*].metadata.name}'`; do echo $pod; oc -n openshift-logging exec -c
elasticsearch $pod -- df -h /elasticsearch/persistent; done
```

2. コマンド出力の **nodes.node_name.fs** フィールドで、対象ノードの空きディスク領域を確認します。
3. すべてのノードでディスク領域を増やしてみてください。
4. ディスク領域を増やせない場合は、新しいデータノードをクラスターに追加してみてください。
5. 新規データノードの追加に問題がある場合は、クラスターの冗長性ポリシー総数を減らします。
   a. 現在の **redundancyPolicy** を確認します。

```
oc -n openshift-logging get es elasticsearch -o jsonpath='{.spec.redundancyPolicy}'
```



**注記**

**ClusterLogging** CR を使用している場合は、以下を入力します。

```
oc -n openshift-logging get cl -o
jsonpath='{.items[*].spec.logStore.elasticsearch.redundancyPolicy}'
```

   b. クラスター **redundancyPolicy** が **SingleRedundancy** よりも大きい場合は、**SingleRedundancy** に設定し、この変更を保存します。

6. 前述の手順で問題が解決しない場合は、古いインデックスを削除します。
   a. Elasticsearch の全インデックスのステータスを確認します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- indices
```

   b. 古いインデックスで削除できるものを特定します。
   c. インデックスを削除します。

```
oc exec -n openshift-logging -c elasticsearch <elasticsearch_pod_name> -- es_util --
query=<elasticsearch_index_name> -X DELETE
```

**関連情報**

- クラスターロギングカスタムリソースについて の「**ClusterLogging** カスタムリソース (CR) のサンプル」で「redundancyPolicy」を参照します。

- Elasticsearch アラートルール で "ElasticsearchDiskSpaceRunningLow" を参照します。
- Elasticsearch で "Free up or increase disk space" と検索し、クラスターのステータスが赤または黄色の問題を修正します。

### 16.5.10. Elasticsearch FileDescriptor の使用量が高い

現在の使用傾向に基づいて、ノードで予測されるファイル記述子の数は十分ではありません。

#### トラブルシューティング

必要に応じて、Elasticsearch ファイル記述子 のトピックで説明されているように、各ノードの **max_file_descriptors** の値を確認して設定します。

#### 関連情報

- Elasticsearch アラートルール で "ElasticsearchHighFileDescriptorUsage" を参照します。
- OpenShift Logging ダッシュボード で "File Descriptors In Use" を参照します。

# 第17章 OPENSHIFT LOGGING のアンインストール

OpenShift Container Platform クラスターからロギングサブシステムを削除できます。

## 17.1. RED HAT のロギングサブシステムのアンインストール

**ClusterLogging** カスタムリソース (CR) を削除して、ログ集計を停止できます。CR を削除した後、残っている他のロギングサブシステムコンポーネントがあり、オプションで削除できます。

**ClusterLogging** CR を削除しても、永続ボリューム要求 (PVC) は削除されません。残りの PVC、永続ボリューム (PV)、および関連データを保持するか、削除するには、さらにアクションを実行する必要があります。

### 前提条件

- Red Hat OpenShift Logging および Elasticsearch Operators がインストールされている必要があります。

### 手順

OpenShift Logging を削除するには、以下を実行します。

1. OpenShift Container Platform Web コンソールを使用して **ClusterLogging** CR を削除できます。
   a. **Administration → Custom Resource Definitions**ページに切り替えます。
   b. **Custom Resource Definitions**ページで、**ClusterLogging** をクリックします。
   c. **Custom Resource Definition Details**ページで、 **Instances** をクリックします。
   d. インスタンスの横にある Options メニュー をクリックし、**Delete ClusterLogging** を選択します。
2. オプション: カスタムリソース定義 (CRD) を削除します。
   a. **Administration → Custom Resource Definitions**ページに切り替えます。
   b. **ClusterLogForwarder** の横にある Options メニュー をクリックし、**Delete Custom Resource Definition** を選択します。
   c. **ClusterLogging** の横にある Options メニュー をクリックし、**Delete Custom Resource Definition** を選択します。
   d. **Elasticsearch** の横にある Options メニュー をクリックし、**Delete Custom Resource Definition** を選択します。
3. オプション: Red Hat OpenShift Logging Operator および OpenShift Elasticsearch Operator を削除します。
   a. **Operators → Installed Operators** ページに切り替えます

a. **Operators → Installed Operators** ページに切り替えます。

b. Red Hat OpenShift Logging Operator の横にある Options メニュー をクリックし、**Uninstall Operator** を選択します。

c. OpenShift Elasticsearch Operator の横にある Options メニュー をクリックし、**Uninstall Operator** を選択します。

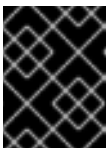
4. オプション: Cluster Logging および Elasticsearch プロジェクトを削除します。

a. **Home → Projects** ページに切り替えます。

b. **openshift-logging** プロジェクトの横にある Options メニュー をクリックし、**Delete Project** を選択します。

c. ダイアログボックスで **openshift-logging** を入力して、**Delete** をクリックし、削除を確認します。

d. **openshift-operators-redhat** プロジェクトの横にある Options メニュー をクリックし、**Delete Project** を選択します。



**重要**

他のグローバル Operator がこの namespace にインストールされている場合は、**openshift-operators-redhat** プロジェクトを削除しないでください。

e. ダイアログボックスで **openshift-operators-redhat** を入力し、**Delete** をクリックして削除を確認します。

5. 他の Pod で再利用するために PVC を保持するには、PVC の回収に必要なラベルまたは PVC 名を保持します。

6. オプション: PVC を保持する必要がない場合は、それを削除できます。

**警告**

PVC の解放または削除により PV が削除され、データの損失が生じる可能性があります。

a. **Storage → Persistent Volume Claims** ページに切り替えます。

b. 各 PVC の横にある Options メニュー をクリックし、**Delete Persistent Volume Claim** を選択します。

c. ストレージ領域を回復する必要がある場合は、PV を削除できます。

**関連情報**

- 永続ボリュームの手動回収

# 第18章 ログレコードのフィールド

次のフィールドは、ロギングサブシステムによってエクスポートされたログレコードに存在する可能性があります。ログレコードは通常 JSON オブジェクトとしてフォーマットされますが、同じデータモデルは他のエンコーディングに適用できます。

Elasticsearch および Kibana からこれらのフィールドを検索するには、検索時に点線の全フィールド名を使用します。たとえば、Elasticsearch **/_search URL** の場合、Kubernetes Pod 名を検索するには、**/_search/q=kubernetes.pod_name:name-of-my-pod** を使用します。

最上位フィールドはすべてのレコードに存在する可能性があります。

# 第19章 MESSAGE

元のログエントリーテキスト (UTF-8 エンコード)。このフィールドが存在しないか、空でない **構造化** フィールドが存在する可能性があります。詳細は、**structured** の説明を参照してください。

| | |
|---|---|
| データのタイプ | text |
| 値の例 | **HAPPY** |

# 第20章 STRUCTURED

構造化されたオブジェクトとしての元のログエントリー。このフィールドは、フォワーダーが構造化された JSON ログを解析するように設定されている場合に存在する可能性があります。元のログエントリーの構造化ログが有効である場合に、このフィールドには同等の JSON 構造が含まれます。それ以外の場合は、このフィールドは空または存在しないため、**message** フィールドに元のログメッセージが含まれます。**構造化された** フィールドには、ログメッセージに含まれるサブフィールドがあるので、ここでは制約が定義されていません。

| | |
|---|---|
| データのタイプ | group |
| 値の例 | map[message:starting fluentd worker pid=21631 ppid=21618 worker=0 pid:21631 ppid:21618 worker:0] |

# 第21章 @TIMESTAMP

ログペイロードが作成された時点か、作成時間が不明な場合は、ログペイロードが最初に収集された時点の UTC 値のマーキング。@接頭辞は、特定の用途で使用できるように予約されているフィールドを表します。Elasticsearch の場合、ほとんどのツールはデフォルトで “@timestamp" を検索します。

| | |
|---|---|
| データのタイプ | 日付 |
| 値の例 | **2015-01-24 14:06:05.071000000 Z** |

# 第22章 HOSTNAME

このログメッセージの発信元のホスト名。Kubernetes クラスターでは、これは **kubernetes.host** と同じです。

| | |
|---|---|
| データのタイプ | キーワード |

# 第23章 IPADDR4

ソースサーバーの IPv4 アドレス。配列を指定できます。

| | |
|---|---|
| データのタイプ | ip |

# 第24章 IPADDR6

ソースサーバーの IPv6 アドレス (ある場合)。配列を指定できます。

| | |
|---|---|
| データのタイプ | ip |

# 第25章 LEVEL

**rsyslog (severitytext プロパティー)**、python のロギングモジュールなどのさまざまなソースのロギングレベル。

以下の値は **syslog.h** から取得されます。値の前には 同等の数値 が追加されます。

- **0** = **emerg**、システムが使用できない。
- **1** = **alert**。アクションをすぐに実行する必要がある。
- **2** = **crit**、致命的な状況。
- **3** = **err**、エラーのある状況。
- **4** = **warn**、警告のある状況。
- **5** = **notice**、通常ではあるが、影響が大きい状況。
- **6** = **info**、情報提供。
- **7** = **debug**、デバッグレベルのメッセージ。

以下の 2 つの値は **syslog.h** の一部ではありませんが、広く使用されています。

- **8** = **trace**、トレースレベルメッセージ。これは、**debug** メッセージよりも詳細にわたります。
- **9** = **unknown**、ロギングシステムで認識できない値を取得した場合。

他のロギングシステムのログレベルまたは優先度を前述のリストで最も近い一致にマップします。たとえば python logging では、**CRITICAL** と **crit**、**ERROR** と **err** が同じです。

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **info** |

# 第26章 PID

ロギングエンティティーのプロセス ID です (ある場合)。

| | |
|---|---|
| データのタイプ | キーワード |

# 第27章 サービス

ロギングエンティティーに関連付けられたサービスの名前です (ある場合)。たとえば、syslog の **APP-NAME** および rsyslog の **programname** プロパティーはサービスフィールドにマップされます。

| | |
|---|---|
| データのタイプ | キーワード |

# 第28章 TAGS

オプション:コレクターまたはノーマライザーによって各ログに配置される、Operator 定義のタグのリストです。ペイロードには、ホワイトスペースで区切られた文字列トークンまたは文字列トークンのJSON 一覧を使用した文字列を指定できます。

| | |
|---|---|
| データのタイプ | text |

# 第29章 FILE

コレクターがこのログエントリーを読み取るログファイルへのパス。通常、これはクラスターノードの **/var/log** ファイルシステム内のパスです。

| | |
|---|---|
| データのタイプ | text |

# 第30章 OFFSET

オフセット値。値が単一ログファイルで単調に増加する場合に、バイトの値をファイルのログ行 (ゼロまたは1ベース) またはログ行の番号 (ゼロまたは1ベース) の開始地点に表示できます。この値はラップでき、ログファイルの新規バージョンを表示できます (ローテーション)。

| | |
|---|---|
| データのタイプ | Long |

# 第31章 KUBERNETES

Kubernetes 固有メタデータの namespace です。

| | |
|---|---|
| データのタイプ | group |

## 31.1. KUBERNETES.POD_NAME

Pod の名前。

| | |
|---|---|
| データのタイプ | キーワード |

## 31.2. KUBERNETES.POD_ID

Pod の Kubernetes ID。

| | |
|---|---|
| データのタイプ | キーワード |

## 31.3. KUBERNETES.NAMESPACE_NAME

Kubernetes の namespace の名前。

| | |
|---|---|
| データのタイプ | キーワード |

## 31.4. KUBERNETES.NAMESPACE_ID

Kubernetes の namespace ID。

| | |
|---|---|
| データのタイプ | キーワード |

## 31.5. KUBERNETES.HOST

Kubernetes ノード名。

| | |
|---|---|
| データのタイプ | キーワード |

## 31.6. KUBERNETES.CONTAINER_NAME

Kubernetes のコンテナーの名前。

| | |
|---|---|
| データのタイプ | キーワード |

## 31.7. KUBERNETES.ANNOTATIONS

Kubernetes オブジェクトに関連付けられるアノテーション。

| | |
|---|---|
| データのタイプ | group |

## 31.8. KUBERNETES.LABELS

元の Kubernetes Pod にあるラベル

| | |
|---|---|
| データのタイプ | group |

## 31.9. KUBERNETES.EVENT

Kubernetes マスター API から取得した Kubernetes イベント。このイベントの説明は基本的に、Event v1 core の **type Event** に準拠します。

| | |
|---|---|
| データのタイプ | group |

### 31.9.1. kubernetes.event.verb

イベントのタイプ: **ADDED**、**MODIFIED** または **DELETED**

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **追加済み** |

### 31.9.2. kubernetes.event.metadata

イベント作成の場所および時間に関する情報

| | |
|---|---|
| データのタイプ | group |

#### 31.9.2.1. kubernetes.event.metadata.name

イベント作成をトリガーしたオブジェクトの名前

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **java-mainclass-1.14d888a4cfc24890** |

#### 31.9.2.2. kubernetes.event.metadata.namespace

イベントが最初に発生した namespace の名前。これは、**eventrouter** アプリケーションのデプロイ先の namespace である **kubernetes.namespace_name** とは異なることに注意してください。

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **default** |

#### 31.9.2.3. kubernetes.event.metadata.selfLink

イベントへのリンク

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **/api/v1/namespaces/javaj/events/java-mainclass-1.14d888a4cfc24890** |

#### 31.9.2.4. kubernetes.event.metadata.uid

イベントの一意の ID

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **d828ac69-7b58-11e7-9cf5-5254002f560c** |

#### 31.9.2.5. kubernetes.event.metadata.resourceVersion

イベントが発生したサーバーの内部バージョンを識別する文字列。クライアントはこの文字列を使用して、オブジェクトが変更されたタイミングを判断できます。

| | |
|---|---|
| データのタイプ | integer |
| 値の例 | **311987** |

### 31.9.3. kubernetes.event.involvedObject

イベントに関するオブジェクト。

| | |
|---|---|
| データのタイプ | group |

#### 31.9.3.1. kubernetes.event.involvedObject.kind

オブジェクトのタイプ

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **ReplicationController** |

#### 31.9.3.2. kubernetes.event.involvedObject.namespace

関係するオブジェクトの namespace 名。これは、**eventrouter** アプリケーションのデプロイ先の namespace である **kubernetes.namespace_name** とは異なる可能性があることに注意してください。

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **default** |

#### 31.9.3.3. kubernetes.event.involvedObject.name

イベントをトリガーしたオブジェクトの名前

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **java-mainclass-1** |

#### 31.9.3.4. kubernetes.event.involvedObject.uid

オブジェクトの一意の ID

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **e6bff941-76a8-11e7-8193-5254002f560c** |

#### 31.9.3.5. kubernetes.event.involvedObject.apiVersion

kubernetes マスター API のバージョン

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **v1** |

#### 31.9.3.6. kubernetes.event.involvedObject.resourceVersion

イベントをトリガーしたサーバーの内部バージョンの Pod を識別する文字列。クライアントはこの文字列を使用して、オブジェクトが変更されたタイミングを判断できます。

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **308882** |

### 31.9.4. kubernetes.event.reason

このイベントを生成する理由を示す、マシンが理解可能な短い文字列

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **SuccessfulCreate** |

### 31.9.5. kubernetes.event.source_component

このイベントを報告したコンポーネント

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **replication-controller** |

### 31.9.6. kubernetes.event.firstTimestamp

イベントが最初に記録された時間

| | |
|---|---|
| データのタイプ | 日付 |
| 値の例 | **2017-08-07 10:11:57.000000000 Z** |

### 31.9.7. kubernetes.event.count

このイベントが発生した回数

| | |
|---|---|
| データのタイプ | integer |
| 値の例 | **1** |

### 31.9.8. kubernetes.event.type

イベントのタイプ、**Normal** または **Warning**。今後、新しいタイプが追加される可能性があります。

| | |
|---|---|
| データのタイプ | キーワード |
| 値の例 | **Normal** |

# 第32章 OPENSHIFT

openshift-logging 固有のメタデータの namespace

| | |
|---|---|
| データのタイプ | group |

## 32.1. OPENSHIFT.LABELS

クラスターログフォワーダー設定によって追加されるラベル

| | |
|---|---|
| データのタイプ | group |