Red Hat Quay 3.12

配置 Red Hat Quay

使用配置选项自定义 Red Hat Quay

Last Updated: 2024-09-19

Red Hat Quay 3.12 配置 Red Hat Quay

使用配置选项自定义 Red Hat Quay

法律通告

摘要

配置 Red Hat Quay

目录

第 1 章 RED HAT QUAY 配置入门

Red Hat Quay 可以由一个独立的独立配置部署，也可以使用 OpenShift Container Platform 上的 Red Hat Quay Operator。

如何创建、检索、更新和验证 Red Hat Quay 配置，具体取决于您使用的部署类型。但是，对于任一部署类型，核心配置选项都相同。核心配置主要通过 **config.yaml** 文件设置，但也可以使用配置 API 来设置。

对于 Red Hat Quay 的独立部署，您必须提供最低所需的配置参数，然后才能启动 registry。启动 Red Hat Quay registry 的最低要求可在"检索当前配置"部分中找到。

如果使用 Red Hat Quay Operator 在 OpenShift Container Platform 上安装 Red Hat Quay，则不需要提供配置参数，因为 Red Hat Quay Operator 提供了用于部署 registry 的默认信息。

使用所需的配置部署了 Red Hat Quay 后，您应该从部署中检索并保存完整的配置。完整配置包含在重启或升级系统时可能需要的额外生成值。

第 2 章 RED HAT QUAY CONFIGURATION DISCLAIMER

使用独立和基于 Operator 的 Red Hat Quay 部署，某些功能和配置参数不会被活跃使用或实现。因此，功能标记（如启用或禁用某些功能的用户）以及不是由红帽支持文档明确记录或请求的配置参数，应谨慎修改。未使用的功能或参数可能无法被完全测试、支持或与 Red Hat Quay 兼容。修改未使用的功能参数可能会导致部署出现意外问题或中断。

有关在独立部署中配置 Red Hat Quay 的详情，请参考 高级 Red Hat Quay 配置

有关配置 Red Hat Quay Operator 部署的详情，请参考在 OpenShift Container Platform 中配置 Red Hat Quay

2.1. RED HAT QUAY 3.12 的配置更新

以下小节详细介绍了 Red Hat Quay 3.12 中的新配置字段。

2.1.1. registry auto-pruning 配置字段

在 Red Hat Quay auto-pruning 功能中添加了以下配置字段：

字段	类型	描述
NOTIFICATION_TASK_RUN_MINIMUM_INTERVAL_MINUTES	整数	间隔（以分钟为单位），定义过期镜像重新运行通知的频率。 **默认为 300**
DEFAULT_NAMESPACE_AUTOPRUNE_POLICY	对象	默认机构范围的自动修剪策略。
.method: number_of_tags	对象	选项指定要保留的标签数的选项。
.value: <integer>	整数	与 **方法一起使用时：number_of_tags**，表示要保留的标签数。 例如，若要保留两个标签，请指定 **2**。
.method: creation_date	对象	选项指定保留标签的持续时间。
.value: <integer>	整数	与 **creation_date** 一起使用时，表示保留标签的时间。 可以设置为秒(**s**)、天(**d**)、月(**m**)、周(**w**)或 years (**y**)。必须包含有效的整数。例如，若要保留一年的标签，请指定 **1y**。

AUTO_PRUNING_DEFAULT_POLICY_POLL_PERIOD	整数	自动修剪 worker 在 registry 级别运行的时间。默认情况下，它被设置为每天运行一次（每 24 小时一次）。值必须以秒为单位。

2.1.2. OAuth 访问令牌重新分配配置字段

添加了以下配置字段以重新分配 OAuth 访问令牌：

字段	类型	描述
FEATURE_ASSIGN_OAUTH_TOKEN	布尔值	允许机构管理员将 OAuth 令牌分配给其他用户。

OAuth 访问令牌重新分配 YAML 示例

```
# ...
FEATURE_ASSIGN_OAUTH_TOKEN: true
# ...
```

2.1.3. 漏洞检测通知配置字段

添加了以下配置字段来根据安全级别在检测到的漏洞中通知用户：

字段	类型	描述
NOTIFICATION_MIN_SEVERITY_ON_NEW_INDEX	字符串	为检测到的漏洞上的新通知设置最小安全级别。避免在首次索引后创建大量通知。如果没有定义，则默认为 **High**。可用的选项包括 **Critical**、**High**、**Medium**、**低**、**Negligible** 和 **Unknown**。

镜像漏洞策略通知 YAML 示例

```
NOTIFICATION_MIN_SEVERITY_ON_NEW_INDEX: High
```

2.1.4. OCI 引用 API 配置字段

以下配置字段允许用户使用 v2 API 列出存储库下清单的 OCI 引用器：

字段	类型	描述
FEATURE_REFERRERS_API	布尔值	启用 OCI 1.1 的引用 API。

OCI 引用启用 YAML 示例

```
# ...
FEATURE_REFERRERS_API: true
# ...
```

2.1.5. 禁用严格的日志记录配置字段

当 Splunk 或 ElasticSearch 等外部系统配置为审计日志目的地时，以下配置字段已被添加到地址，但不稳定。当设置为 **True** 时，日志记录事件会被记录到 stdout。

字段	类型	描述
ALLOW_WITHOUT_STRICT_LOGGING	布尔值	当设置为 **True** 时，允许您在无法写入审计日志时使用任何 registry 操作。

严格日志记录 **YAML** 示例

```
# ...
ALLOW_WITHOUT_STRICT_LOGGING: True
# ...
```

2.1.6. 通知间隔配置字段

添加了以下配置字段来增强 Red Hat Quay 通知：

字段	类型	描述
NOTIFICATION_TASK_RUN_MINIMUM_INTERVAL_MINUTES	整数	间隔（以分钟为单位），定义为过期镜像重新运行通知的频率。默认情况下，此字段设置为每 5 小时通知 Red Hat Quay 用户发生的事件。

通知重新运行 **YAML** 示例

```
# ...
NOTIFICATION_TASK_RUN_MINIMUM_INTERVAL_MINUTES: 10
# ...
```

2.1.7. Clair 索引层大小配置字段

为 Clair 安全扫描程序添加了以下配置字段，它允许 Red Hat Quay 管理员设置索引允许的最大层大小。

字段	类型	描述

SECURITY_SCANNER_V4_INDEX_MAX_LAYER_SIZE	字符串	索引允许的最大层大小。如果层大小超过配置的大小，Red Hat Quay UI 会返回 **以下消息：此标签的清单具有太大的层，供 Quay Security Scanner 进行索引**。默认值为 **8GB**，建议的最大值为 **10GB**。 **示例 ：8GB**

2.2. 编辑配置文件

要部署独立的 Red Hat Quay 实例，您必须提供最少的配置信息。最低配置的要求可在 "Red Hat Quay minimal configuration" 中找到。

提供必填字段后，您可以验证您的配置。如果有任何问题，则会突出显示它们。

要使更改生效，必须重启 registry。

2.3. 配置文件在独立部署中的位置

对于 Red Hat Quay 的独立部署，启动 Red Hat Quay registry 时必须指定 **config.yaml** 文件。此文件位于配置卷中。例如，当使用以下命令部署 Red Hat Quay 时，配置文件位于 **$QUAY/config/config.yaml** 中：

```
$ sudo podman run -d --rm -p 80:8080 -p 443:8443 \
  --name=quay \
  -v $QUAY/config:/conf/stack:Z \
  -v $QUAY/storage:/datastorage:Z \
  registry.redhat.io/quay/quay-rhel8:v3.12.2
```

2.4. 最小配置

独立部署 Red Hat Quay 需要以下配置选项：

- 服务器主机名
- HTTP 或 HTTPS
- 身份验证类型，如 Database 或轻量级目录访问协议(LDAP)
- 用于加密数据的 secret 密钥
- 镜像存储
- 元数据的数据库
- 用于构建日志和用户事件的 Redis
- 标签过期选项

2.4.1. 最小配置文件示例

以下示例显示了将本地存储用于镜像的最小配置文件示例：

```
AUTHENTICATION_TYPE: Database
BUILDLOGS_REDIS:
    host: quay-server.example.com
    password: strongpassword
    port: 6379
    ssl: false
DATABASE_SECRET_KEY: 0ce4f796-c295-415b-bf9d-b315114704b8
DB_URI: postgresql://quayuser:quaypass@quay-server.example.com:5432/quay
DEFAULT_TAG_EXPIRATION: 2w
DISTRIBUTED_STORAGE_CONFIG:
    default:
        - LocalStorage
        - storage_path: /datastorage/registry
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS: []
DISTRIBUTED_STORAGE_PREFERENCE:
    - default
PREFERRED_URL_SCHEME: http
SECRET_KEY: e8f9fe68-1f84-48a8-a05f-02d72e6eccba
SERVER_HOSTNAME: quay-server.example.com
SETUP_COMPLETE: true
TAG_EXPIRATION_OPTIONS:
    - 0s
    - 1d
    - 1w
    - 2w
    - 4w
USER_EVENTS_REDIS:
    host: quay-server.example.com
    port: 6379
    ssl: false
```

2.4.2. 本地存储

只有在部署 registry 进行 *概念验证* 时，才建议在镜像中使用本地存储。

在配置本地存储时，存储会在启动 registry 时在命令行中指定。

以下命令将本地目录 **$QUAY/storage** 映射到容器中的 **datastorage** 路径：

```
$ sudo podman run -d --rm -p 80:8080 -p 443:8443 \
  --name=quay \
  -v $QUAY/config:/conf/stack:Z \
  -v $QUAY/storage:/datastorage:Z \
  registry.redhat.io/quay/quay-rhel8:v3.12.2
```

2.4.3. 云存储

存储配置在镜像存储部分中详细介绍。对于某些用户，比较 Google Cloud Platform 和本地存储配置之间的区别可能很有用。例如，以下 YAML 提供了 Google Cloud Platform 存储配置：

$QUAY/config/config.yaml

```
DISTRIBUTED_STORAGE_CONFIG:
    default:
```

```
        - GoogleCloudStorage
        - access_key: GOOGQIMFB3ABCDEFGHIJKLMN
          bucket_name: quay_bucket
          secret_key: FhDAYe2HeuAKfvZCAGyOioNaaRABCDEFGHIJKLMN
          storage_path: /datastorage/registry
          boto_timeout: 120 (1)
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS: []
DISTRIBUTED_STORAGE_PREFERENCE:
    - default
```

(1) 可选。从连接时抛出超时异常的时间（以秒为单位）。默认值为 **60** 秒。另外，还包括时间（以秒为单位），直到尝试进行连接时抛出超时异常。默认值为 **60** 秒。

使用云存储启动 registry 时，在命令行中不需要配置。例如：

```
$ sudo podman run -d --rm -p 80:8080 -p 443:8443 \
  --name=quay \
  -v $QUAY/config:/conf/stack:Z \
  registry.redhat.io/quay/quay-rhel8:v3.12.2
```

第 3 章 配置字段

本节论述了部署 Red Hat Quay 时所需的配置字段和可选配置字段。

3.1. 所需的配置字段

配置 Red Hat Quay 所需的字段在以下部分中介绍：

- 常规必填字段
- 镜像存储
- 元数据的数据库
- 用于构建日志和用户事件的 Redis
- 标签过期选项

3.2. 自动化选项

以下小节描述了 Red Hat Quay 部署的可用自动化选项：

- 预配置 Red Hat Quay 以进行自动化
- 使用 API 创建第一个用户

3.3. 可选的配置字段

Red Hat Quay 的可选字段可在以下部分中找到：

- 基本配置
- SSL
- LDAP
- 仓库镜像
- 配额管理
- 安全扫描程序
- Helm
- 操作日志
- 构建日志
- Dockerfile 构建
- OAuth
- 配置嵌套软件仓库
- 在 Quay 中添加其他 OCI 介质类型

- Mail
- User
- reCAPTCHA
- ACI
- JWT
- 应用程序令牌
- 其它
- 用户界面 v2
- IPv6 配置字段
- 传统选项

3.4. 常规必填字段

下表描述了 Red Hat Quay 部署所需的配置字段：

表 3.1. 常规必填字段

字段	类型	描述
AUTHENTICATION_TYPE (Required)	字符串	用于凭证身份验证的身份验证引擎。 **，值：** **数据库,LDAP**, JWT **,JWT**,**Keystone**,**OIDC** **默认：数据库**
PREFERRED_URL_SCHEME (Required)	字符串	访问 Red Hat Quay 时使用的 URL 方案。 **值：** 以下之一 **http**, **https** Default: **http**
SERVER_HOSTNAME (Required)	字符串	可以访问 Red Hat Quay 的 URL，不包括网络协议。 **例如：** **quay-server.example.com**

字段	类型	描述
DATABASE_SECRET_KEY （必需）	字符串	用于加密数据库中敏感字段的密钥。设置后不应更改这个值，否则所有依赖字段（如存储库镜像用户名和密码配置）都无效。 这个值由 Red Hat Quay Operator 为基于 Operator 的部署自动设置。对于独立部署，管理员可以使用 Open SSL 或类似的工具提供自己的密钥。密钥长度不应超过 63 个字符。
SECRET_KEY (Required)	字符串	用于加密会话 Cookie 和正确解释用户会话所需的 CSRF 令牌的密钥。设置时不应更改该值。应该在所有 Red Hat Quay 实例中保留。如果没有在所有实例间具有持久性，则可能会出现登录失败以及与会话持久性相关的其他错误。
SETUP_COMPLETE (Required)	布尔值	这是一个来自早期版本的软件的工件，目前 **必须使用 true** 值指定。

3.5. 数据库配置

本节论述了 Red Hat Quay 部署可用的数据库配置字段。

3.5.1. 数据库 URI

使用 Red Hat Quay 时，使用所需的 **DB_URI** 字段配置与数据库的连接。

下表描述了 **DB_URI** 配置字段：

表 3.2. 数据库 URI

字段	类型	描述
DB_URI （必需）	字符串	用于访问数据库的 URI，包括任何凭据。 **DB_URI** 字段示例： **postgresql://quayuser:quaypass@quay-server.example.com:5432/quay**

3.5.2. 数据库连接参数

可选的连接参数由 **DB_CONNECTION_ARGS** 参数配置。**DB_CONNECTION_ARGS** 中定义的一些键值对是通用的，另一些则特定于数据库。

下表描述了数据库连接参数：

表 3.3. 数据库连接参数

字段	类型	描述
DB_CONNECTION_ARGS	对象	数据库的可选连接参数，如超时和 SSL/TLS。
.autorollback	布尔值	是否使用 thread-local 连接。应该始终为 **true**
.threadlocals	布尔值	是否使用自动滚动连接。应该始终为 **true**

3.5.2.1. PostgreSQL SSL/TLS 连接参数

使用 SSL/TLS 时，配置取决于您部署的数据库。以下示例显示了 PostgreSQL SSL/TLS 配置：

```
DB_CONNECTION_ARGS:
  sslmode: verify-ca
  sslrootcert: /path/to/cacert
```

sslmode 选项决定是否使用，安全 SSL/TLS TCP/IP 连接的优先级将与服务器协商。有六个模式：

表 3.4. SSL/TLS 选项

模式	描述
disable	您的配置只尝试非 SSL/TLS 连接。
allow	您的配置首先会尝试非 SSL/TLS 连接。失败时，尝试 SSL/TLS 连接。
首选（默认）	您的配置首先会尝试 SSL/TLS 连接。失败时，尝试非 SSL/TLS 连接。
require	您的配置只尝试 SSL/TLS 连接。如果存在 root CA 文件，它将像指定了 verify-ca 一样验证证书。
verify-ca	您的配置只尝试 SSL/TLS 连接，并验证服务器证书是否由可信证书颁发机构(CA)发布。
verify-full	仅尝试 SSL/TLS 连接，并验证服务器证书是否由可信 CA 发布，并且请求的服务器主机名与证书中的主机名匹配。

有关 PostgreSQL 有效参数的更多信息，请参阅 数据库连接控制功能。

3.5.2.2. MySQL SSL/TLS 连接参数

以下示例显示了 MySQL SSL/TLS 配置示例：

```
DB_CONNECTION_ARGS:
  ssl:
    ca: /path/to/cacert
```

有关 MySQL 的有效连接参数的信息，请访问 使用类似URI的字符串或键-值对连接到服务器。

3.6. 镜像存储

本节详细介绍了 Red Hat Quay 中提供的镜像存储功能和配置字段。

3.6.1. 镜像存储功能

下表描述了 Red Hat Quay 的镜像存储功能：

表 3.5. 存储配置特性

字段	类型	描述
FEATURE_REPO_MIRROR	布尔值	如果设置为 true，请启用存储库镜像。 **默认： false**
FEATURE_PROXY_STORAGE	布尔值	是否通过 NGINX 代理存储中的所有直接下载 URL。 **默认： false**
FEATURE_STORAGE_REPLICATION	布尔值	是否在存储引擎之间自动复制。 **默认： false**

3.6.2. 镜像存储配置字段

下表描述了 Red Hat Quay 的镜像存储配置字段：

表 3.6. 存储配置字段

字段	类型	描述
DISTRIBUTED_STORAGE_CONFIG (Required)	对象	在 Red Hat Quay 中使用的存储引擎的配置。每个键代表存储引擎的唯一标识符。该值由组成一个对象（键、值）的元组组成，用于描述存储引擎参数。 **默认： []**

字段	类型	描述
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS (Required)	字符串数组	默认情况下，存储引擎（由 **DISTRIBUTED_STORAGE_CONFIG**）中的存储引擎列表（默认为所有其他存储引擎）。
DISTRIBUTED_STORAGE_PREFERENCE (Required)	字符串数组	要使用的 **DISTRIBUTED_STORAGE_CONFIG**的首选存储引擎（按 ID）。首选引擎意味着首先检查拉取(pull)和镜像推送到其中的镜像。 **默认： false**
MAXIMUM_LAYER_SIZE	字符串	镜像层允许的最大大小。 Pattern:**^[0-9]+(G|M)$** **示例**:**100G** **默认 ：20G**

3.6.3. 本地存储

以下 YAML 显示了使用本地存储的示例配置：

```
DISTRIBUTED_STORAGE_CONFIG:
  default:
    - LocalStorage
    - storage_path: /datastorage/registry
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS: []
DISTRIBUTED_STORAGE_PREFERENCE:
    - default
```

3.6.4. OpenShift Container Storage/NooBaa

以下 YAML 显示了使用 OpenShift Container Storage/NooBaa 实例的配置示例：

```
DISTRIBUTED_STORAGE_CONFIG:
  rhocsStorage:
    - RHOCSStorage
    - access_key: access_key_here
      secret_key: secret_key_here
      bucket_name: quay-datastore-9b2108a3-29f5-43f2-a9d5-2872174f9a56
      hostname: s3.openshift-storage.svc.cluster.local
      is_secure: 'true'
      port: '443'
```

```
    storage_path: /datastorage/registry
    maximum_chunk_size_mb: 100 (1)
    server_side_assembly: true (2)
```

1. 定义最终副本的最大块大小（以 MB 为单位）。如果 **server_side_assembly** 设为 **false**，则无效。
2. 可选。Red Hat Quay 是否应该尝试使用服务器端装配和最终块的副本，而不是客户端装配。默认值为 **true**。

3.6.5. Ceph 对象网关/RadosGW 存储

以下 YAML 显示了使用 Ceph/RadosGW 的示例配置。

注意

radosgw 是一个与内部兼容 S3 的存储解决方案。请注意，这与常规 **AWS S3Storage** 不同，它专门用于 Amazon Web Services S3。这意味着 RadosGW 实施 S3 API，并且需要凭证，如 **access_key**、**secret_key** 和 **bucket_name**。有关 Ceph 对象网关和 S3 API 的更多信息，请参阅 Ceph 对象网关和 S3 API。

带有常规 **s3** 访问权限的 **radosgw**

```
DISTRIBUTED_STORAGE_CONFIG:
  radosGWStorage: (1)
    - RadosGWStorage
    - access_key: <access_key_here>
      bucket_name: <bucket_name_here>
      hostname: <hostname_here>
      is_secure: true
      port: '443'
      secret_key: <secret_key_here>
      storage_path: /datastorage/registry
      maximum_chunk_size_mb: 100 (2)
      server_side_assembly: true (3)
```

1. 用于常规 s3 访问。请注意，常规 s3 访问不严格限制为 Amazon Web Services (AWS) s3，可用于 RadosGW 或其他存储服务。有关使用 AWS S3 驱动程序的常规 s3 访问示例，请参阅"AWS S3 存储"。
2. 可选。定义最终副本的最大块大小（以 MB 为单位）。如果 **server_side_assembly** 设为 **false**，则无效。
3. 可选。Red Hat Quay 是否应该尝试使用服务器端装配和最终块的副本，而不是客户端装配。默认值为 **true**。

3.6.6. AWS S3 存储

以下 YAML 显示了使用 AWS S3 存储的示例配置。

```
# ...
DISTRIBUTED_STORAGE_CONFIG:
```

```
  default:
    - S3Storage (1)
    - host: s3.us-east-2.amazonaws.com
      s3_access_key: ABCDEFGHIJKLMN
      s3_secret_key: OL3ABCDEFGHIJKLMN
      s3_bucket: quay_bucket
      s3_region: <region> (2)
      storage_path: /datastorage/registry
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS: []
DISTRIBUTED_STORAGE_PREFERENCE:
    - default
# ...
```

1. **S3Storage** 存储驱动程序应该只用于 AWS S3 存储桶。请注意，这与常规 S3 访问不同，其中可以使用 RadosGW 驱动程序或其他存储服务。例如，请参阅"例如 B: 使用 RadosGW 带有常规 S3 访问"。
2. 可选。Amazon Web Services 区域。默认为 **us-east-1**。

3.6.6.1. AWS STS S3 存储

以下 YAML 显示了在 OpenShift Container Platform 配置中使用 Amazon Web Services (AWS)安全令牌服务(STS)与 Red Hat Quay 搭配使用的示例配置。

```
# ...
DISTRIBUTED_STORAGE_CONFIG:
  default:
   - STSS3Storage
   - sts_role_arn: <role_arn> (1)
     s3_bucket: <s3_bucket_name>
     storage_path: <storage_path>
     sts_user_access_key: <s3_user_access_key> (2)
     sts_user_secret_key: <s3_user_secret_key> (3)
     s3_region: <region> (4)
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS: []
DISTRIBUTED_STORAGE_PREFERENCE:
    - default
# ...
```

1. 唯一的 Amazon 资源名称(ARN)。
2. 生成的 AWS S3 用户访问密钥。
3. 生成的 AWS S3 用户 secret 密钥。
4. 可选。Amazon Web Services 区域。默认为 **us-east-1**。

3.6.7. Google Cloud Storage

以下 YAML 显示了使用 Google Cloud Storage 的示例配置：

```
DISTRIBUTED_STORAGE_CONFIG:
```

```
    googleCloudStorage:
        - GoogleCloudStorage
        - access_key: GOOGQIMFB3ABCDEFGHIJKLMN
          bucket_name: quay-bucket
          secret_key: FhDAYe2HeuAKfvZCAGyOioNaaRABCDEFGHIJKLMN
          storage_path: /datastorage/registry
          boto_timeout: 120 (1)
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS: []
DISTRIBUTED_STORAGE_PREFERENCE:
    - googleCloudStorage
```

(1) 可选。从连接时抛出超时异常的时间（以秒为单位）。默认值为 **60** 秒。另外，还包括时间（以秒为单位），直到尝试进行连接时抛出超时异常。默认值为 **60** 秒。

3.6.8. Azure Storage

以下 YAML 显示了使用 Azure Storage 的示例配置：

```
DISTRIBUTED_STORAGE_CONFIG:
  azureStorage:
    - AzureStorage
    - azure_account_name: azure_account_name_here
      azure_container: azure_container_here
      storage_path: /datastorage/registry
      azure_account_key: azure_account_key_here
      sas_token: some/path/
      endpoint_url: https://[account-name].blob.core.usgovcloudapi.net (1)
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS: []
DISTRIBUTED_STORAGE_PREFERENCE:
    - azureStorage
```

(1) Azure 存储的 **endpoint_url** 参数是可选的，可用于 Microsoft Azure Government (MAG)端点。如果留空，则 **endpoint_url** 将连接到普通的 Azure 区域。

从 Red Hat Quay 3.7 开始，您必须使用 MAG Blob 服务的主端点。使用 MAG Blob 服务的二级端点将导致以下错误： **AuthenticationErrorDetail:Cannot find the claimed account when trying the account whusc8-secondary**。

3.6.9. Swift 存储

以下 YAML 显示了使用 Swift 存储的配置示例：

```
DISTRIBUTED_STORAGE_CONFIG:
  swiftStorage:
    - SwiftStorage
    - swift_user: swift_user_here
      swift_password: swift_password_here
      swift_container: swift_container_here
      auth_url: https://example.org/swift/v1/quay
      auth_version: 1
      ca_cert_path: /conf/stack/swift.cert"
      storage_path: /datastorage/registry
```

```
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS: []
DISTRIBUTED_STORAGE_PREFERENCE:
  - swiftStorage
```

3.6.10. Nutanix 对象存储

以下 YAML 显示了使用 Nutanix 对象存储的示例配置。

```
DISTRIBUTED_STORAGE_CONFIG:
  nutanixStorage: #storage config name
    - RadosGWStorage #actual driver
    - access_key: access_key_here #parameters
      secret_key: secret_key_here
      bucket_name: bucket_name_here
      hostname: hostname_here
      is_secure: 'true'
      port: '443'
      storage_path: /datastorage/registry
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS: []
DISTRIBUTED_STORAGE_PREFERENCE: #must contain name of the storage config
    - nutanixStorage
```

3.6.11. IBM Cloud 对象存储

以下 YAML 显示了使用 IBM Cloud 对象存储的示例配置。

```
DISTRIBUTED_STORAGE_CONFIG:
  default:
  - IBMCloudStorage #actual driver
  - access_key: <access_key_here> #parameters
    secret_key: <secret_key_here>
    bucket_name: <bucket_name_here>
    hostname: <hostname_here>
    is_secure: 'true'
    port: '443'
    storage_path: /datastorage/registry
    maximum_chunk_size_mb: 100mb (1)
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS:
- default
DISTRIBUTED_STORAGE_PREFERENCE:
- default
```

(1) 可选。建议设置为 **100mb**。

3.6.12. NetApp ONTAP S3 对象存储

以下 YAML 显示了使用 NetApp ONTAP S3 的示例配置。

```
DISTRIBUTED_STORAGE_CONFIG:
  local_us:
  - RadosGWStorage
  - access_key: <access_key>
```

```
    bucket_name: <bucket_name>
    hostname: <host_url_address>
    is_secure: true
    port: <port>
    secret_key: <secret_key>
    storage_path: /datastorage/registry
    signature_version: v4
DISTRIBUTED_STORAGE_DEFAULT_LOCATIONS:
- local_us
DISTRIBUTED_STORAGE_PREFERENCE:
- local_us
```

3.7. REDIS 配置字段

本节详细介绍了 Redis 部署可用的配置字段。

3.7.1. 构建日志

以下构建日志配置字段可用于 Redis 部署：

表 3.7. 构建日志配置

字段	类型	描述
BUILDLOGS_REDIS (Required)	对象	构建日志缓存的 redis 连接详情。
.host (Required)	字符串	可以访问 Redis 的主机名。 **示例：** **quay-server.example.com**
.port （必需）	Number	可以访问 Redis 的端口。 **示例：** **6379**
.password	字符串	用于连接到 Redis 实例的密码。 **示例：** **strongpassword**
.SSL （可选）	布尔值	是否启用 Redis 和 Quay 之间的 TLS 通信。默认为false。

3.7.2. 用户事件

以下用户事件字段可用于 Redis 部署：

表 3.8. 用户事件配置

字段	类型	描述
USER_EVENTS_REDIS (Required)	对象	用于用户事件处理的 redis 连接详情。
.host (Required)	字符串	可以访问 Redis 的主机名。 **示例：** **quay-server.example.com**
.port （必需）	Number	可以访问 Redis 的端口。 **示例：** **6379**
.password	字符串	用于连接到 Redis 实例的密码。 **示例：** **strongpassword**
.ssl	布尔值	是否启用 Redis 和 Quay 之间的 TLS 通信。默认为false。
.ssl_keyfile （可选）	字符串	存储要使用的客户端证书的密钥数据库文件的名称。 **示例：** **ssl_keyfile:**/path/to/server/privatekey.pem
.ssl_certfile (Optional)	字符串	用于指定 SSL 证书的文件路径。 **示例：** **ssl_certfile: /path/to/server/certificate.pem**
.ssl_cert_reqs （可选）	字符串	用于指定在 SSL/TLS 握手过程中要执行的证书验证级别。 **示例：** **ssl_cert_reqs: CERT_REQUIRED**
.ssl_ca_certs (Optional)	字符串	用于指定包含可信证书颁发机构(CA)证书列表的文件的路径。 **示例：** **ssl_ca_certs: /path/to/ca_certs.pem**
.ssl_ca_data (Optional)	字符串	用于指定包含 PEM 格式的可信 CA 证书的字符串。 **示例：** **ssl_ca_data: <certificate>**

字段	类型	描述
.ssl_check_hostname (Optional)	布尔值	在设置到服务器的 SSL/TLS 连接时使用。它指定客户端是否应该检查服务器的 SSL/TLS 证书中的主机名是否与它所连接的服务器的主机名匹配。 **示例：** **ssl_check_hostname: true**

3.7.3. Redis 配置示例

以下 YAML 显示了使用带有可选 SSL/TLS 字段的 Redis 的示例配置：

```
BUILDLOGS_REDIS:
  host: quay-server.example.com
  password: strongpassword
  port: 6379
  ssl: true


USER_EVENTS_REDIS:
  host: quay-server.example.com
  password: strongpassword
  port: 6379
  ssl: true
  ssl_*: <path_location_or_certificate>
```

注意

如果您的部署对 Redis 使用 Azure Cache，并且 **ssl** 被设置为 **true**，则端口默认为 **6380**。

3.8. MODELCACHE 配置选项

Red Hat Quay 中提供了以下选项来配置 ModelCache。

3.8.1. memcache 配置选项

memcache 是默认的 ModelCache 配置选项。使用 Memcache 时，不需要额外的配置。

3.8.2. 单个 Redis 配置选项

以下配置适用于具有可选只读副本的单个 Redis 实例：

```
DATA_MODEL_CACHE_CONFIG:
  engine: redis
  redis_config:
    primary:
      host: <host>
      port: <port>
      password: <password if ssl is true>
```

```
      ssl: <true | false >
    replica:
      host: <host>
      port: <port>
      password: <password if ssl is true>
      ssl: <true | false >
```

3.8.3. 集群 Redis 配置选项

对集群的 Redis 实例使用以下配置：

```
DATA_MODEL_CACHE_CONFIG:
  engine: rediscluster
  redis_config:
    startup_nodes:
      - host: <cluster-host>
        port: <port>
    password: <password if ssl: true>
    read_from_replicas: <true|false>
    skip_full_coverage_check: <true | false>
    ssl: <true | false >
```

3.9. 标签过期配置字段

Red Hat Quay 提供了以下标签过期配置字段：

表 3.9. 标签过期配置字段

字段	类型	描述
FEATURE_GARBAGE_COLLECTION	布尔值	是否启用垃圾回收存储库。 **默认为** True
TAG_EXPIRATION_OPTIONS (Required)	字符串数组	如果启用，用户可以选择在其命名空间中过期标签的选项。 **Pattern:** **^[0-9]+(w|m|d|h|s)$**
DEFAULT_TAG_EXPIRATION (Required)	字符串	时间机器的默认可配置标签过期时间。 **Pattern:** **^[0-9]+(w|m|d|h|s)$** **Default: 2w**
FEATURE_CHANGE_TAG_EXPIRATION	布尔值	是否允许用户和机构更改其命名空间中标签过期时间。 **默认为** True

字段	类型	描述
FEATURE_AUTO_PRUNE	布尔值	当设置为 **True** 时，启用与 auto-pruning 标签相关的功能。 Default: **False**
NOTIFICATION_TASK_RUN_MINIMUM_INTERVAL_MINUTES	整数	间隔（以分钟为单位），定义过期镜像重新运行通知的频率。 **默认为 300**
DEFAULT_NAMESPACE_AUTOPRUNE_POLICY	对象	默认机构范围的自动修剪策略。
.method: number_of_tags	对象	选项指定要保留的标签数的选项。
.value: <integer>	整数	与 **方法一起使用时：number_of_tags**，表示要保留的标签数。 例如，若要保留两个标签，请指定 **2**。
.creation_date	对象	选项指定保留标签的持续时间。
.value: <integer>	整数	与 **creation_date** 一起使用时，表示保留标签的时间。 可以设置为秒(**s**)、天(**d**)、月(**m**)、周(**w**)或 years (**y**)。必须包含有效的整数。例如，若要保留一年的标签，请指定 **1y**。
AUTO_PRUNING_DEFAULT_POLICY_POLL_PERIOD	整数	自动修剪 worker 在 registry 级别运行的时间。默认情况下，它被设置为每天运行一次（每 24 小时一次）。值必须以秒为单位。

3.9.1. 标签过期配置示例

以下 YAML 示例显示了标签过期配置示例。

```
# ...
DEFAULT_TAG_EXPIRATION: 2w
TAG_EXPIRATION_OPTIONS:
    - 0s
    - 1d
    - 1w
    - 2w
    - 4w
# ...
```

3.9.2. registry 范围自动修剪策略示例

以下 YAML 示例通过标签和创建日期显示 registry 范围自动运行示例。

根据标签数自动修剪策略示例

```
# ...
DEFAULT_NAMESPACE_AUTOPRUNE_POLICY:
  method: number_of_tags
  value: 10 (1)
# ...
```

(1) 在这种情况下，保留十个标签。

创建日期的 registry 自动修剪策略示例

```
# ...
DEFAULT_NAMESPACE_AUTOPRUNE_POLICY:
  method: creation_date
  value: 1y
# ...
```

3.10. 配额管理配置字段

表 3.10. 配额管理配置

字段	类型	描述
FEATURE_QUOTA_MANAGEMENT	布尔值	为配额管理功能启用配置、缓存和验证。 **Default:** `False`
DEFAULT_SYSTEM_REJECT_QUOTA_BYTES	字符串	启用系统默认配额拒绝所有机构的字节允许。 默认情况下，不设置任何限制。
QUOTA_BACKFILL	布尔值	启用配额回填 worker 来计算预先存在的 Blob 的大小。 **默认 ：True**
QUOTA_TOTAL_DELAY_SECONDS	字符串	启动配额回填的时间延迟。滚动部署可能会导致总数不正确。此字段**必须设置为** 比滚动部署完成的时间更长的时间。 **默认 ：1800**

字段	类型	描述
PERMANENTLY_DELETE_TAGS	布尔值	启用与从时间窗中删除标签相关的功能。 **默认 ：False**
RESET_CHILD_MANIFEST_EXPIRATION	布尔值	重置以子清单为目标的临时标签过期。将此功能设置为 **True** 时，子清单会立即收集垃圾回收。 **默认 ：False**

3.10.1. 配额管理配置示例

以下 YAML 是启用配额管理时推荐的配置。

配额管理 YAML 配置

```
FEATURE_QUOTA_MANAGEMENT: true
FEATURE_GARBAGE_COLLECTION: true
PERMANENTLY_DELETE_TAGS: true
QUOTA_TOTAL_DELAY_SECONDS: 1800
RESET_CHILD_MANIFEST_EXPIRATION: true
```

3.11. 代理缓存配置字段

表 3.11. 代理配置

字段	类型	描述
FEATURE_PROXY_CACHE	布尔值	启用 Red Hat Quay 作为上游 registry 的拉取(pull)缓存。 **默认 ：false**

3.12. 机器人帐户配置字段

表 3.12. 机器人帐户配置字段

字段	类型	描述
ROBOTS_DISALLOW	布尔值	当设置为 **true** 时，机器人帐户会阻止所有交互，以及创建 **默认:False**

3.13. 预配置 RED HAT QUAY 以进行自动化

Red Hat Quay 支持几个启用自动化的配置选项。用户可以在部署前配置这些选项，以减少与用户界面交互的需求。

3.13.1. 允许 API 创建第一个用户

要创建第一个用户，用户需要将 **FEATURE_USER_INITIALIZE** 参数设置为 **true**，并调用 **/api/v1/user/initialize** API。与需要现有机构中 OAuth 应用生成的 OAuth 令牌的其他 registry API 调用不同，API 端点不需要身份验证。

在部署 Red Hat Quay 后，用户可以使用 API 创建用户，如 **quayadmin**，只要没有创建其他用户。如需更多信息，请参阅使用 API 创建第一个用户。

3.13.2. 启用常规 API 访问

用户应将 **BROWSER_API_CALLS_XHR_ONLY** 配置选项设置为 **false**，以允许常规访问 Red Hat Quay registry API。

3.13.3. 添加超级用户

部署 Red Hat Quay 后，用户可以创建用户，并授予第一个具有完整权限的管理员特权。用户可以使用 **SUPER_USER** 配置对象提前配置完全权限。例如：

```
# ...
SERVER_HOSTNAME: quay-server.example.com
SETUP_COMPLETE: true
SUPER_USERS:
  - quayadmin
# ...
```

3.13.4. 限制用户创建

配置超级用户后，您可以通过将 **FEATURE_USER_CREATION** 设置为 **false** 来限制新用户到超级用户组。例如：

```
# ...
FEATURE_USER_INITIALIZE: true
BROWSER_API_CALLS_XHR_ONLY: false
SUPER_USERS:
- quayadmin
FEATURE_USER_CREATION: false
# ...
```

3.13.5. 在 Red Hat Quay 3.12 中启用新功能

要使用新的 Red Hat Quay 3.12 功能，请启用一些或全部功能：

```
# ...
FEATURE_UI_V2: true
FEATURE_UI_V2_REPO_SETTINGS: true
FEATURE_AUTO_PRUNE: true
ROBOTS_DISALLOW: false
# ...
```

3.13.6. 推荐的自动化配置

建议对自动化使用以下 **config.yaml** 参数：

```
# ...
FEATURE_USER_INITIALIZE: true
BROWSER_API_CALLS_XHR_ONLY: false
SUPER_USERS:
- quayadmin
FEATURE_USER_CREATION: false
# ...
```

3.13.7. 使用初始配置部署 Red Hat Quay Operator

使用以下步骤使用初始配置在 OpenShift Container Platform 上部署 Red Hat Quay。

先决条件

- 已安装 **oc** CLI。

流程

1. 使用配置文件创建 secret：

```
$ oc create secret generic -n quay-enterprise --from-file config.yaml=./config.yaml init-config-bundle-secret
```

2. 创建 **quayregistry.yaml** 文件。识别非受管组件并引用创建的 secret，例如：

```
apiVersion: quay.redhat.com/v1
kind: QuayRegistry
metadata:
  name: example-registry
  namespace: quay-enterprise
spec:
  configBundleSecret: init-config-bundle-secret
```

3. 部署 Red Hat Quay registry：

```
$ oc create -n quay-enterprise -f quayregistry.yaml
```

后续步骤

- 使用 API 创建第一个用户

3.13.8. 使用 API 创建第一个用户

使用以下步骤在 Red Hat Quay 组织中创建第一个用户。

先决条件

- 配置选项 **FEATURE_USER_INITIALIZE** 必须设为 **true**。
- 数据库中不存在任何用户。

流程

此流程通过指定 **"access_token": true** 来请求 OAuth 令牌。

1. 打开 Red Hat Quay 配置文件并更新以下配置字段：

```
FEATURE_USER_INITIALIZE: true
SUPER_USERS:
    - quayadmin
```

2. 输入以下命令停止 Red Hat Quay 服务：

```
$ sudo podman stop quay
```

3. 输入以下命令启动 Red Hat Quay 服务：

```
$ sudo podman run -d -p 80:8080 -p 443:8443 --name=quay -v $QUAY/config:/conf/stack:Z
-v $QUAY/storage:/datastorage:Z {productrepo}/{quayimage}:{productminv}
```

4. 运行以下 **CURL** 命令，以使用用户名、密码、电子邮件和访问令牌生成新用户：

```
$ curl -X POST -k  http://quay-server.example.com/api/v1/user/initialize --header 'Content-
Type: application/json' --data '{ "username": "quayadmin", "password":"quaypass12345",
"email": "quayadmin@example.com", "access_token": true}'
```

如果成功，命令会返回带有用户名、电子邮件和加密密码的对象。例如：

```
{"access_token":"6B4QTRSTSD1HMIG915VPX7BMEZBVB9GPNY2FC2ED",
"email":"quayadmin@example.com","encrypted_password":"1nZMLH57RIE5UGdL/yYpDOHL
qiNCgimb6W9kfF8MjZ1xrfDpRyRs9NUnUuNuAitW","username":"quayadmin"} #
gitleaks:allow
```

如果用户存在于数据库中，则返回错误：

```
{"message":"Cannot initialize user in a non-empty database"}
```

如果您的密码至少没有 8 个字符或包含空格，则返回错误：

```
{"message":"Failed to initialize user: Invalid password, password must be at least 8
characters and contain no whitespace."}
```

5. 输入以下命令登录到 Red Hat Quay 部署：

```
$ sudo podman login -u quayadmin -p quaypass12345 http://quay-server.example.com --tls-
verify=false
```

输出示例

```
Login Succeeded!
```

3.13.8.1. 使用 OAuth 令牌

调用 API 后，您可以通过指定返回的 OAuth 代码来调用 Red Hat Quay API 的其余部分。

先决条件

- 您已调用 **/api/v1/user/initialize** API，并传递用户名、密码和电子邮件地址。

流程

- 输入以下命令来获取当前用户列表：

```
$ curl -X GET -k -H "Authorization: Bearer
6B4QTRSTSD1HMIG915VPX7BMEZBVB9GPNY2FC2ED" https://example-registry-quay-
quay-enterprise.apps.docs.quayteam.org/api/v1/superuser/users/
```

输出示例：

```
{
   "users": [
      {
         "kind": "user",
         "name": "quayadmin",
         "username": "quayadmin",
         "email": "quayadmin@example.com",
         "verified": true,
         "avatar": {
            "name": "quayadmin",
            "hash":
"3e82e9cbf62d25dec0ed1b4c66ca7c5d47ab9f1f271958298dea856fb26adc4c",
            "color": "#e7ba52",
            "kind": "user"
         },
         "super_user": true,
         "enabled": true
      }
   ]
}
```

在本例中，**quayadmin** 用户的详细信息将返回，因为它是到目前为止创建的唯一用户。

3.13.8.2. 使用 API 创建机构

以下过程详细介绍了如何使用 API 创建 Red Hat Quay 组织。

先决条件

- 您已调用 **/api/v1/user/initialize** API，并传递用户名、密码和电子邮件地址。
- 您已通过指定返回的 OAuth 代码来调用 Red Hat Quay API 的其余部分。

流程

1. 要创建机构，请使用对 **api/v1/organization/** 端点的 POST 调用：

```
$ curl -X POST -k --header 'Content-Type: application/json' -H "Authorization: Bearer
```

```
6B4QTRSTSD1HMIG915VPX7BMEZBVB9GPNY2FC2ED" https://example-registry-quay-quay-enterprise.apps.docs.quayteam.org/api/v1/organization/ --data '{"name": "testorg", "email": "testorg@example.com"}'
```

输出示例：

```
"Created"
```

2. 您可以输入以下命令来检索您创建的机构的详情：

```
$ curl -X GET -k --header 'Content-Type: application/json' -H "Authorization: Bearer 6B4QTRSTSD1HMIG915VPX7BMEZBVB9GPNY2FC2ED" https://min-registry-quay-quay-enterprise.apps.docs.quayteam.org/api/v1/organization/testorg
```

输出示例：

```
{
    "name": "testorg",
    "email": "testorg@example.com",
    "avatar": {
        "name": "testorg",
        "hash": "5f113632ad532fc78215c9258a4fb60606d1fa386c91b141116a1317bf9c53c8",
        "color": "#a55194",
        "kind": "user"
    },
    "is_admin": true,
    "is_member": true,
    "teams": {
        "owners": {
            "name": "owners",
            "description": "",
            "role": "admin",
            "avatar": {
                "name": "owners",
                "hash":
"6f0e3a8c0eb46e8834b43b03374ece43a030621d92a7437beb48f871e90f8d90",
                "color": "#c7c7c7",
                "kind": "team"
            },
            "can_view": true,
            "repo_count": 0,
            "member_count": 1,
            "is_synced": false
        }
    },
    "ordered_teams": [
        "owners"
    ],
    "invoice_email": false,
    "invoice_email_address": null,
    "tag_expiration_s": 1209600,
    "is_free_account": true
}
```

3.14. 基本配置字段

表 3.13. 基本配置

字段	类型	描述
REGISTRY_TITLE	字符串	如果指定，registry 的长格式标题。显示在 Red Hat Quay 部署的前端中，例如，位于您机构的登录页面中。不应超过 35 个字符。 **默认：** **Red Hat Quay**
REGISTRY_TITLE_SHORT	字符串	如果指定，registry 的简短格式标题。title 显示在您的机构的不同页面上，例如，作为您组织的 **教程** 标题。 **默认：** **Red Hat Quay**
CONTACT_INFO	字符串数组	如果指定，要在联系页面上显示联系信息。如果只指定了单个联系信息，请联系页脚将直接链接。
[0]	字符串	添加用于发送电子邮件的链接。 Pattern: **^mailto: (.)+$** Example: **mailto:support@quay.io**
[1]	字符串	添加访问 IRC 聊天房的链接。 Pattern: **^irc://(.)+$** Example: **irc://chat.freenode.net:6665/quay**
[2]	字符串	添加一个链接以调用电话号码。 Pattern: **^tel: (.)+$** Example: **tel:+1-888-930-3475**
[3]	字符串	添加指向定义的 URL 的链接。 Pattern: **^http (s)?://(.)+$** Example: **https://twitter.com/quayio**

3.15. SSL 配置字段

表 3.14. SSL 配置

字段	类型	描述
PREFERRED_URL_SCHEME	字符串	**http** 或 **https** 之一。请注意，当从客户端到 Quay 的通信路径中没有 TLS 加密时，用户只会将其 **PREFERRED_URL_SCHEME** 设置为 **http**。 在使用 TLS 终止负载均衡器、反向代理（如 Nginx）或直接使用自定义 SSL 证书的 Quay 时，用户必须将其 **PREFERRED_URL_SCHEME** ' 设置为 **'https**。在大多数情况下，**PREFERRED_URL_SCHEME** 应为 **https**。 **默认： http**
SERVER_HOSTNAME (Required)	字符串	可以访问 Red Hat Quay 的 URL，不包括网络协议。 **例如：** **quay-server.example.com**

字段	类型	描述
SSL_CIPHERS	字符串数组	如果指定，则要启用和禁用的 nginx 定义的 SSL 密码列表 ： [**ECDHE-RSA-AES128-GCM-SHA256**,**ECDHE-ECDSA-AES128-GCM-SHA256**,**ECDHE-RSA-AES256-GCM-SHA384**,**ECDHE-ECDSA-AES256-GCM-SHA384**,**DHE-RSA-AES128-GCM-SHA256**, DHE-**DSS-AES 128-GCM-SHA256, kEDH +AESGCM ,ECDHE-RSA-AES128-SHA256**,**ECDHE-ECDSA-AES128-SHA256**, ECDHE-RSA-**AES128-SHA**,**ECDHE-ECDSA-AES128-SHA ,ECDHE-RSA-AES 256-SHA384**, ECDHE-ECDSA-AES256-**AES256-SHA** 384,**ECDHE-ECDSA-AES128-SHA384, ECDHE-RSA-AES256-SHA**,**ECDHE-ECDSA-AES256-SHA**, DHE-RSA-AES128-**AES128-SHA256, DHE-RSA-**AES128-**SHA ,DHE-DSS-AES128-SHA256 , DHE-RSA-AES256-SHA256-SHA256**,**DHE-DSS-AES256-SHA, DHE-DSS - AES256-SHA ,AES128-GCM-SHA256**,**AES256-GCM-SHA384**, AES128-SHA256 ,**AES256-SHA256**, AES128-SHA , AES256-SHA , AES ,**!3DES"**, **!aNULL**, ! **eNULL**, ! **eNULL**,**!EXPORT, DES,!RC 4, MD 5 , !PSK ,!aECDH,!EDH-DSS-DES-CBC3-SHA,!EDH-RSA-DES-CBC3-SHA,!KRB5-DES-CBC3-SHA]**

字段	类型	描述
SSL_PROTOCOLS	字符串数组	如果指定了，nginx 被配置为启用列表中定义的 SSL 协议列表。从列表中删除 SSL 协议会禁用 Red Hat Quay 启动期间的协议。 ： **['TLSv1','TLSv1.1','TLSv1.2', 'TLSv1.3**]'
SESSION_COOKIE_SECURE	布尔值	对于使用 SSL 的所有安装，是否应在会话 Cookie 上设置 **secure** 属性 ： False **Recommendation:** Set to True

3.15.1. 配置 SSL

1. 将证书文件和主密钥文件复制到您的配置目录中，确保它们分别命名为 **ssl.cert** 和 **ssl.key** ：

```
$ cp ~/ssl.cert $QUAY/config
$ cp ~/ssl.key $QUAY/config
$ cd $QUAY/config
```

2. 编辑 **config.yaml** 文件并指定您希望 Quay 处理 TLS：

config.yaml

```
...
SERVER_HOSTNAME: quay-server.example.com
...
PREFERRED_URL_SCHEME: https
...
```

3. 停止 **Quay** 容器并重启 registry

3.16. 在 RED HAT QUAY CONTAINER 中添加 TLS 证书

要将自定义 TLS 证书添加到 Red Hat Quay 中，请在 Red Hat Quay 配置目录下创建一个名为 **extra_ca_certs/** 的新目录。将任何所需的特定于站点的 TLS 证书复制到这个新目录中。

3.16.1. 在 Red Hat Quay 中添加 TLS 证书

1. 查看要添加到容器中的证书

```
$ cat storage.crt
```

```
-----BEGIN CERTIFICATE-----
MIIDTTCCAjWgAwIBAgIJAMVr9ngjJhzbMA0GCSqGSIb3DQEBCwUAMD0xCzAJBgNV
[...]
-----END CERTIFICATE-----
```

2. 创建 certs 目录并在其中复制证书

```
$ mkdir -p quay/config/extra_ca_certs
$ cp storage.crt quay/config/extra_ca_certs/
$ tree quay/config/
├── config.yaml
├── extra_ca_certs
│   ├── storage.crt
```

3. 使用 **podman ps** 获取 **Quay** 容器的 **CONTAINER ID** ：

```
$ sudo podman ps
CONTAINER ID  IMAGE                           COMMAND               CREATED
STATUS             PORTS
5a3e82c4a75f  <registry>/<repo>/quay:v3.12.2 "/sbin/my_init"  24 hours ago  Up
18 hours  0.0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp, 443/tcp  grave_keller
```

4. 使用该 ID 重启容器：

```
$ sudo podman restart 5a3e82c4a75f
```

5. 检查复制到容器命名空间中的证书：

```
$ sudo podman exec -it 5a3e82c4a75f cat /etc/ssl/certs/storage.pem
-----BEGIN CERTIFICATE-----
MIIDTTCCAjWgAwIBAgIJAMVr9ngjJhzbMA0GCSqGSIb3DQEBCwUAMD0xCzAJBgNV
```

3.17. LDAP 配置字段

表 3.15. LDAP 配置

字段	类型	描述
AUTHENTICATION_TYPE (Required)	字符串	必须设置为 **LDAP**。
FEATURE_TEAM_SYNCING	布尔值	是否允许团队成员资格与身份验证引擎中的后备组同步(OIDC、LDAP 或 Keystone)。 **默认： true**
FEATURE_NONSUPERUSER_TEAM_SYNCING_SETUP	布尔值	如果启用，非超级用户可设置团队同步。 **默认： false**

字段	类型	描述
LDAP_ADMIN_DN	字符串	用于 LDAP 验证的管理 DN。
LDAP_ADMIN_PASSWD	字符串	LDAP 验证的 admin 密码。
LDAP_ALLOW_INSECURE_FALLBACK	布尔值	是否允许 LDAP 身份验证的 SSL 不安全回退。
LDAP_BASE_DN	字符串数组	用于 LDAP 身份验证的基本 DN。
LDAP_EMAIL_ATTR	字符串	LDAP 身份验证的电子邮件属性。
LDAP_UID_ATTR	字符串	LDAP 身份验证的 uid 属性。
LDAP_URI	字符串	LDAP URI。
LDAP_USER_FILTER	字符串	LDAP 身份验证的用户过滤器。
LDAP_USER_RDN	字符串数组	用于 LDAP 身份验证的用户 RDN。
LDAP_SECONDARY_USER_RDNS	字符串数组	如果用户对象所在的多个机构单元，则提供二级用户相对 DN。
TEAM_RESYNC_STALE_TIME	字符串	如果为团队启用了团队同步，则检查其成员资格和重新同步的频率。 Pattern: **^[0-9]+(w|m|d|h|s)$** Example: **2h** Default: **30m**
LDAP_SUPERUSER_FILTER	字符串	**LDAP_USER_FILTER** 配置字段的子集。配置后，当 Red Hat Quay 使用 LDAP 作为其身份验证提供程序时，允许 Red Hat Quay 管理员将轻量级目录访问协议 (LDAP)用户配置为超级用户。 使用此字段，管理员可以添加或删除超级用户，而无需更新 Red Hat Quay 配置文件并重启其部署。 此字段要求将 **AUTHENTICATION_TYPE** 设置为 **LDAP**。

字段	类型	描述
GLOBAL_READONLY_SUPER_USERS	字符串	设置后，可授予用户对所有存储库的读取访问权限，无论它们是公共存储库。仅适用于通过 **LDAP_SUPERUSER_FILTER** 配置字段定义的超级用户。
LDAP_RESTRICTED_USER_FILTER	字符串	**LDAP_USER_FILTER** 配置字段的子集。配置后，当 Red Hat Quay 使用 LDAP 作为其身份验证提供程序时，允许 Red Hat Quay 管理员将轻量级目录访问协议 (LDAP)用户配置为受限用户。 此字段要求将 **AUTHENTICATION_TYPE** 设置为 **LDAP**。
FEATURE_RESTRICTED_USERS	布尔值	当设为 **True** 时，**LDAP_RESTRICTED_USER_FILTER** 处于活动状态时，只有定义的 LDAP 组中的用户才会受到限制。 Default: **False**
LDAP_TIMEOUT	整数	指定 LDAP 操作的时间限制（以秒为单位）。这限制了 LDAP 搜索、绑定或其他操作可以花费的时间。与 **ldapsearch** 中的 **-l** 选项类似，它会设置客户端操作超时。 **默认 ：10**
LDAP_NETWORK_TIMEOUT	整数	指定与 LDAP 服务器建立连接的时间限制（以秒为单位）。这是 Red Hat Quay 在网络操作期间等待响应的最长时间，类似于 **ldapsearch** 中的 **-o nettimeout** 选项。 **默认 ：10**

3.17.1. LDAP 配置参考

使用以下引用，使用所需的 LDAP 设置更新 **config.yaml** 文件。

3.17.1.1. 基本 LDAP 配置

对基本 LDAP 配置使用以下参考。

```
---
```

```
AUTHENTICATION_TYPE: LDAP (1)
---
LDAP_ADMIN_DN: uid=<name>,ou=Users,o=<organization_id>,dc=
<example_domain_component>,dc=com (2)
LDAP_ADMIN_PASSWD: ABC123 (3)
LDAP_ALLOW_INSECURE_FALLBACK: false (4)
LDAP_BASE_DN: (5)
  - dc=example
  - dc=com
LDAP_EMAIL_ATTR: mail (6)
LDAP_UID_ATTR: uid (7)
LDAP_URI: ldap://<example_url>.com (8)
LDAP_USER_FILTER: (memberof=cn=developers,ou=Users,dc=<domain_name>,dc=com) (9)
LDAP_USER_RDN: (10)
  - ou=people
LDAP_SECONDARY_USER_RDNS: (11)
    - ou=<example_organization_unit_one>
    - ou=<example_organization_unit_two>
    - ou=<example_organization_unit_three>
    - ou=<example_organization_unit_four>
```

(1) 必需。必须设置为 **LDAP**。

(2) 必需。用于 LDAP 验证的管理 DN。

(3) 必需。LDAP 验证的 admin 密码。

(4) 必需。是否为 LDAP 身份验证允许 SSL/TLS 不安全回退。

(5) 必需。用于 LDAP 身份验证的基本 DN。

(6) 必需。LDAP 身份验证的电子邮件属性。

(7) 必需。LDAP 身份验证的 UID 属性。

(8) 必需。LDAP URI。

(9) 必需。LDAP 身份验证的用户过滤器。

(10) 必需。用于 LDAP 身份验证的用户 RDN。

(11) 可选。如果有多个机构单元，则二级用户相对 DN。

3.17.1.2. LDAP 受限用户配置

对 LDAP 受限用户配置使用以下引用。

```
# ...
AUTHENTICATION_TYPE: LDAP
# ...
FEATURE_RESTRICTED_USERS: true (1)
# ...
LDAP_ADMIN_DN: uid=<name>,ou=Users,o=<organization_id>,dc=
<example_domain_component>,dc=com
```

```
LDAP_ADMIN_PASSWD: ABC123
LDAP_ALLOW_INSECURE_FALLBACK: false
LDAP_BASE_DN:
    - o=<organization_id>
    - dc=<example_domain_component>
    - dc=com
LDAP_EMAIL_ATTR: mail
LDAP_UID_ATTR: uid
LDAP_URI: ldap://<example_url>.com
LDAP_USER_FILTER: (memberof=cn=developers,ou=Users,o=<example_organization_unit>,dc=
<example_domain_component>,dc=com)
LDAP_RESTRICTED_USER_FILTER: (<filterField>=<value>) 2
LDAP_USER_RDN:
    - ou=<example_organization_unit>
    - o=<organization_id>
    - dc=<example_domain_component>
    - dc=com
# ...
```

1 在配置 LDAP 受限用户时，必须设置为 **true**。

2 将指定用户配置为受限用户。

3.17.1.3. LDAP 超级用户配置参考

对 LDAP 超级用户配置使用以下引用：

```
# ...
AUTHENTICATION_TYPE: LDAP
# ...
LDAP_ADMIN_DN: uid=<name>,ou=Users,o=<organization_id>,dc=
<example_domain_component>,dc=com
LDAP_ADMIN_PASSWD: ABC123
LDAP_ALLOW_INSECURE_FALLBACK: false
LDAP_BASE_DN:
    - o=<organization_id>
    - dc=<example_domain_component>
    - dc=com
LDAP_EMAIL_ATTR: mail
LDAP_UID_ATTR: uid
LDAP_URI: ldap://<example_url>.com
LDAP_USER_FILTER: (memberof=cn=developers,ou=Users,o=<example_organization_unit>,dc=
<example_domain_component>,dc=com)
LDAP_SUPERUSER_FILTER: (<filterField>=<value>) 1
LDAP_USER_RDN:
    - ou=<example_organization_unit>
    - o=<organization_id>
    - dc=<example_domain_component>
    - dc=com
# ...
```

1 将指定用户配置为超级用户。

3.18. 镜像配置字段

表 3.16. 镜像配置

字段	类型	描述
FEATURE_REPO_MIRROR	布尔值	启用或禁用存储库镜像 **默认值： false**
REPO_MIRROR_INTERVAL	Number	检查存储库镜像候选间隔的秒数 **默认为** 30
REPO_MIRROR_SERVER_HOSTNAME	字符串	替换 **SERVER_HOSTNAME** 作为镜像目的地。 **默认：** None **示例**: **openshift-quay-service**
REPO_MIRROR_TLS_VERIFY	布尔值	在镜像过程中需要 HTTPS 并验证 Quay registry 的证书。 **默认： false**
REPO_MIRROR_ROLLBACK	布尔值	当设置为 **true** 时，存储库会在镜像失败后回滚。 **默认 ：false**

3.19. 安全扫描程序配置字段

表 3.17. 安全扫描程序配置

字段	类型	描述
FEATURE_SECURITY_SCANNER	布尔值	启用或禁用安全扫描程序 **默认值： false**
FEATURE_SECURITY_NOTIFICATIONS	布尔值	如果启用了安全扫描程序，请打开或关闭安全通知 **默认： false**

字段	类型	描述
SECURITY_SCANNER_V4_REINDEX_THRESHOLD	字符串	此参数用于确定在重新索引后重新索引清单之前要等待的最短时间（以秒为单位）。数据从 manifestsecuritystatus 表中的 **last_indexed datetime** 计算。这个参数用于避免在每次索引运行时尝试重新索引每个失败的清单。re-index 的默认时间为 300 秒。
SECURITY_SCANNER_V4_ENDPOINT	字符串	V4 安全扫描程序的端点 模式： **^http (s)?://(.)+$** **示例：** **http://192.168.99.101:6060**
SECURITY_SCANNER_V4_PSK	字符串	为 Clair 生成的预共享密钥(PSK)
SECURITY_SCANNER_ENDPOINT	字符串	V2 安全扫描程序的端点 模式： **^http (s)?://(.)+$** **示例：** **http://192.168.99.100:6060**
SECURITY_SCANNER_INDEXING_INTERVAL	整数	此参数用于确定安全扫描程序索引间隔之间的秒数。触发索引时，Red Hat Quay 将查询其数据库以获取需要由 Clair 索引的清单。这包括还没有索引的清单，以及之前失败的索引的清单。 **默认：30**
FEATURE_SECURITY_SCANNER_NOTIFY_ON_NEW_INDEX	布尔值	是否允许向新推送发送有关漏洞的通知。 **默认 ：True**
SECURITY_SCANNER_V4_MANIFEST_CLEANUP	布尔值	Red Hat Quay 垃圾回收程序是否移除不由其他标签或清单引用的清单。 **默认 ：True**

字段	类型	描述
NOTIFICATION_MIN_SEVERITY_ON_NEW_INDEX	字符串	为检测到的漏洞上的新通知设置最小安全级别。避免在首次索引后创建大量通知。如果没有定义，则默认为 **High**。可用的选项包括 **Critical**、**High**、**Medium**、**低**、**Negligible** 和 **Unknown**。
SECURITY_SCANNER_V4_INDEX_MAX_LAYER_SIZE	字符串	索引允许的最大层大小。如果层大小超过配置的大小，Red Hat Quay UI 会返回 **以下消息：此标签的清单具有太大的层，供 Quay Security Scanner 进行索引**。默认值为 **8GB**，建议的最大值为 **10GB**。 **默认 ：8GB**

3.19.1. 使用 Clair v4 重新索引

当 Clair v4 索引清单时，结果应该是确定的。例如，同一清单应生成相同的索引报告。在更改扫描程序前，这是 true，因为使用不同的扫描程序会在报告中生成与特定清单相关的不同信息。因此，Clair v4 会公开索引引擎(**/indexer/api/v1/index_state**)的状态表示，以确定扫描程序配置是否已更改。

Red Hat Quay 通过在解析到 Quay 数据库时将其保存到索引报告来利用此索引状态。如果此状态自之前扫描后更改了，Red Hat Quay 会在定期索引过程中尝试重新索引该清单。

默认情况下，此参数被设置为 30 秒。如果他们希望索引过程更频繁地运行，用户可能会缩短时间，例如，如果他们不希望等待 30 秒才能在推送新标签后看到安全扫描结果。如果用户希望将请求模式更多地控制到 Clair，以及在 Red Hat Quay 数据库上执行的数据库操作模式，用户也可以更改该参数。

3.19.2. 安全扫描程序配置示例

以下 YAML 是启用安全扫描程序功能时推荐的配置。

安全扫描程序 **YAML** 配置

```
FEATURE_SECURITY_NOTIFICATIONS: true
FEATURE_SECURITY_SCANNER: true
FEATURE_SECURITY_SCANNER_NOTIFY_ON_NEW_INDEX: true
...
SECURITY_SCANNER_INDEXING_INTERVAL: 30
SECURITY_SCANNER_V4_MANIFEST_CLEANUP: true
SECURITY_SCANNER_V4_ENDPOINT: http://quay-server.example.com:8081
SECURITY_SCANNER_V4_PSK: MTU5YzA4Y2ZkNzJoMQ==
SERVER_HOSTNAME: quay-server.example.com
SECURITY_SCANNER_V4_INDEX_MAX_LAYER_SIZE: 8GB (1)
...
```

(1) 建议的最大值为 **10GB**。

3.20. HELM 配置字段

表 3.18. Helm 配置字段

字段	类型	描述
FEATURE_GENERAL_OCI_SUPPORT	布尔值	启用对 OCI 工件的支持。 **默认值：** True

以下 Open Container Initiative (OCI)工件类型默认内置在 Red Hat Quay 中，并通过 FEATURE_GENERAL_OCI_SUPPORT 配置字段启用：

字段	介质类型	支持的内容类型
Helm	**application/vnd.cncf.helm.config.v1+json**	**application/tar+gzip**, **application/vnd.cncf.helm.chart.content.v1.tar+gzip**
Cosign	**application/vnd.oci.image.config.v1+json**	**application/vnd.dev.cosign.simplesigning.v1+json**,**application/vnd.dsse.envelope.v1+json**
SPDX	**application/vnd.oci.image.config.v1+json**	**text/spdx**, **text/spdx+xml**, **text/spdx+json**
Syft	**application/vnd.oci.image.config.v1+json**	**application/vnd.syft+json**
CycloneDX	**application/vnd.oci.image.config.v1+json**	**application/vnd.cyclonedx**,**application/vnd.cyclonedx+xml**,**application/vnd.cyclonedx+json**
in-toto	**application/vnd.oci.image.config.v1+json**	**application/vnd.in-toto+json**
Unknown	**application/vnd.cncf.openpolicyagent.policy.layer.v1+rego**	**application/vnd.cncf.openpolicyagent.policy.layer.v1+rego**,**application/vnd.cncf.openpolicyagent.data.layer.v1+json**

3.20.1. 配置 Helm

以下 YAML 是启用 Helm 时的示例配置。

Helm YAML 配置

```
FEATURE_GENERAL_OCI_SUPPORT: true
```

3.21. 开放容器计划配置字段

表 3.19. 其他 OCI 工件配置字段

字段	类型	描述
FEATURE_REFERRERS_API	布尔值	启用 OCI 1.1 的引用 API。

OCI 引用启用 YAML 示例

```
# ...
FEATURE_REFERRERS_API: True
# ...
```

3.22. 未知介质类型

表 3.20. 未知介质类型配置字段

字段	类型	描述
IGNORE_UNKNOWN_MEDIATYPES	布尔值	启用后，允许容器 registry 平台忽略对支持的工件类型的特定限制，并接受任何未识别或未知介质类型。 **默认： false**

3.22.1. 配置未知介质类型

以下 YAML 是启用未知或未识别介质类型时的示例配置。

未知介质类型 YAML 配置

```
IGNORE_UNKNOWN_MEDIATYPES: true
```

3.23. 操作日志配置字段

3.23.1. 操作日志存储配置

表 3.21. 操作日志存储配置

字段	类型	描述
FEATURE_LOG_EXPORT	布尔值	是否允许导出操作日志。 **默认为 True**

字段	类型	描述
LOGS_MODEL	字符串	指定处理日志数据的首选方法。 值： **database**,**transition_reads_both_writes_es**,**elasticsearch**, mvapich Default: **database**
LOGS_MODEL_CONFIG	对象	操作日志的日志模型配置。
ALLOW_WITHOUT_STRICT_LOGGING	布尔值	当设置为 **True** 时，如果 Splunk 或 ElasticSearch 等外部日志系统间歇性不可用，则允许用户正常推送镜像。事件会被记录到 stdout。 **Default:** False

3.23.1.1. Elasticsearch 配置字段

为 Red Hat Quay 配置 Elasticsearch 时，可以使用以下字段。

- **LOGS_MODEL_CONFIG** [object]: Logs model config for action logs.
  - **elasticsearch_config** [object]: Elasticsearch 集群配置。
    - **access_key** [字符串]：Elasticsearch 用户（或 AWS ES 的 IAM 密钥）。
      - **示例 ：some_string**
    - **Host** [string]: Elasticsearch 集群端点。
      - **示例 ：host.elasticsearch.example**
    - **index_prefix** [string]: Elasticsearch 的索引前缀。
      - **示例 ：logentry_**
    - **index_settings** [object]: Elasticsearch 的索引设置
    - **use_ssl** [boolean]: 为 Elasticsearch 使用 ssl。默认值为 **True**。
      - **示例:True**
    - **SECRET_KEY** [字符串]：Elasticsearch 密码（或 AWS ES 的 IAM secret）。
      - **示例 ：some_secret_string**
    - **AWS_REGION** [字符串]：Amazon Web 服务区域。

- **示例**:**us-east-1**

- **port** [number]: Elasticsearch 集群端点端口。
  - **示例 ：1234**

- **kinesis_stream_config** [object]: AWS Kinesis Stream 配置。
  - **aws_secret_key** [字符串]: AWS secret key。
    - **示例 ：some_secret_key**
  - **stream_name** [string]: Kinesis stream to send action logs to.
    - **示例 ：logentry-kinesis-stream**
  - **aws_access_key** [字符串]：AWS access key。
    - **示例 ：some_access_key**
  - **retries** [number]: 在单个请求中尝试次数上限。
    - **示例 ：5**
  - **read_timeout** [数字]：从连接时超时前的秒数。
    - **示例 ：5**
  - **max_pool_connections** [number] ：连接池中要保留的最大连接数。
    - **示例 ：10**
  - **AWS_REGION** [字符串]：AWS region。
    - **示例**:**us-east-1**
  - **connect_timeout** [数字]: 在尝试建立连接时超时前的秒数。
    - **示例 ：5**

- **制作者** [字符串]：如果日志记录到 Elasticsearch，日志生成者。
  - **enum**: kafka, elasticsearch, kinesis_stream
  - **示例 ：kafka**

- **kafka_config** [object]: Kafka 集群配置。
  - **topic** [string]: Kafka topic 用来发布日志条目。
    - **示例 ：logentry**
  - **bootstrap_servers** [array]： Kafka 代理列表来引导客户端。
  - **max_block_seconds** [number]: 在 **send （）** 期间阻止的最大秒数（以秒为单位），因为缓冲区已满或元数据不可用。
    - **示例 ：10**

3.23.1.2. Splunk 配置字段

为 Red Hat Quay 配置 Splunk 时，可以使用以下字段。

- **producer** [string]: **mvapich .**在配置 Splunk 时使用。
- **splunk_config** [object]: Splunk 操作日志或 Splunk 集群配置的日志模型配置。
  - **Host** [string]: Splunk 集群端点。
  - **port** [integer]: Splunk 管理集群端点端口。
  - **bearer_token** [字符串] ：Splunk 的 bearer 令牌。
  - **verify_ssl** [boolean]: Enable (**True**)或禁用(**False**) TLS/SSL 验证 HTTPS 连接。
  - **index_prefix** [string]: Splunk 的索引前缀。
  - **ssl_ca_path** [字符串] ：指向单个 **.pem** 文件的相对路径，其中包含用于 SSL 验证的证书颁发机构(CA)。

Splunk 配置示例

```
# ...
LOGS_MODEL: splunk
LOGS_MODEL_CONFIG:
    producer: splunk
    splunk_config:
        host: http://<user_name>.remote.csb
        port: 8089
        bearer_token: <bearer_token>
        url_scheme: <http/https>
        verify_ssl: False
        index_prefix: <splunk_log_index_name>
        ssl_ca_path: <location_to_ssl-ca-cert.pem>
# ...
```

3.23.1.3. Splunk HEC 配置字段

在为 Red Hat Quay 配置 Splunk HTTP 事件收集器(HEC)时，可以使用以下字段。

- **producer** [string]: **mvapich_hec**.在配置 Splunk HEC 时使用。
- **splunk_hec_config** [object]: Splunk HTTP 事件收集器操作日志配置的日志模型配置。
  - **Host** [string]: Splunk 集群端点。
  - **port** [integer]: Splunk 管理集群端点端口。
  - **hec_token** [字符串]: 用于 Splunk 的 HEC 令牌。
  - **url_scheme** [string] ：用于访问 Splunk 服务的 URL 方案。如果 Splunk 位于 SSL/TLS 之后，则必须是 **https**。
  - **verify_ssl** [boolean]: Enable (**true**)或为 HTTPS 连接禁用(**false**) SSL/TLS 验证。
  - **index** [string] ：要使用的 Splunk 索引。
  - **mvapich_host** [字符串] ：记录此事件的主机名。

- **mvapich_sourcetype** [string] ：要使用的 Splunk **sourcetype** 的名称。

```
# ...
LOGS_MODEL: splunk
LOGS_MODEL_CONFIG:
  producer: splunk_hec
  splunk_hec_config: (1)
    host: prd-p-aaaaaq.splunkcloud.com (2)
    port: 8088 (3)
    hec_token: 12345678-1234-1234-1234-1234567890ab (4)
    url_scheme: https (5)
    verify_ssl: False (6)
    index: quay (7)
    splunk_host: quay-dev (8)
    splunk_sourcetype: quay_logs (9)
# ...
```

3.23.2. 操作日志轮转和归档配置

表 3.22. 操作日志轮转和归档配置

字段	类型	描述
FEATURE_ACTION_LOG_ROTATION	布尔值	启用日志轮转和归档将将所有超过 30 天的日志移到存储中。 **默认： false**
ACTION_LOG_ARCHIVE_LOCATION	字符串	如果启用了操作日志归档，则要放置归档数据的存储引擎。 **示例： s3_us_east**
ACTION_LOG_ARCHIVE_PATH	字符串	如果启用了操作日志归档，则要放置 **归档数据的路径。** **sample: archive /actionlogs**
ACTION_LOG_ROTATION_THRESHOLD	字符串	轮转日志的时间间隔。 **示例： 30d**

3.23.3. 操作日志审计日志配置

表 3.23. 审计日志配置字段

字段	类型	描述

字段	类型	描述
ACTION_LOG_AUDIT_LOGINS	布尔值	当设置为 **True** 时，跟踪高级事件，如登录和注销，UI 以及将 Docker 用于常规用户、机器人帐户，以及用于特定于应用的令牌帐户。 **默认为 True**

3.24. 构建日志配置字段

表 3.24. 构建日志配置字段

字段	类型	描述
FEATURE_READER_BUILD_LOGS	布尔值	如果设置为 true，则构建日志可以被有权访问存储库的 **用户读取**，而不只是 **写入访问权限** 或 **admin** 访问权限。 **默认： False**
LOG_ARCHIVE_LOCATION	字符串	存储位置，在 **DISTRIBUTED_STORAGE_CONFIG** 中定义的，用于放置归档的构建日志。 **示例： s3_us_east**
LOG_ARCHIVE_PATH	字符串	配置的存储引擎下的路径，其中将归档的构建日志放在 **.JSON** 格式。 **示例：** archive/**buildlogs**

3.25. DOCKERFILE 构建触发器字段

表 3.25. Dockerfile 构建支持

字段	类型	描述
FEATURE_BUILD_SUPPORT	布尔值	是否支持 Dockerfile 构建。 Default: **False**

字段	类型	描述
SUCCESSIVE_TRIGGER_FAILURE_DISABLE_THRESHOLD	Number	如果没有设置 **None**，则自动禁用构建触发器前可能会出现的连续失败数量。 **默认 ：100**
SUCCESSIVE_TRIGGER_INTERNAL_ERROR_DISABLE_THRESHOLD	Number	如果没有设置 **None**，则会自动禁用构建触发器前出现的连续内部错误数量 **默认： 5**

3.25.1. GitHub 构建触发器

表 3.26. GitHub 构建触发器

字段	类型	描述
FEATURE_GITHUB_BUILD	布尔值	是否支持 GitHub 构建触发器。 **默认： False**
GITHUB_TRIGGER_CONFIG	对象	使用 GitHub Enterprise 进行构建触发器的配置。
.GITHUB_ENDPOINT (Required)	字符串	GitHub Enterprise 的端点。 **示例： https://github.com/**
.API_ENDPOINT	字符串	要使用的 GitHub Enterprise API 的端点。对于 **github.com** 必须被覆盖。 ，例 如：**https://api.github.com/**
.CLIENT_ID (Required)	字符串	此 Red Hat Quay 实例的注册客户端 ID；这不能与 **GITHUB_LOGIN_CONFIG** 共享。
.CLIENT_SECRET (Required)	字符串	此 Red Hat Quay 实例的注册客户端 secret。

3.25.2. Bitbucket 构建触发器

表 3.27. Bitbucket 构建触发器

字段	类型	描述
FEATURE_BITBUCKET_BUILD	布尔值	是否支持 Bitbucket 构建触发器。 **默认：** False
BITBUCKET_TRIGGER_CONFIG	对象	将 BitBucket 用于构建触发器的配置。
.CONSUMER_KEY (Required)	字符串	此 Red Hat Quay 实例注册的消费者密钥（客户端 ID）。
.CONSUMER_SECRET (Required)	字符串	此 Red Hat Quay 实例注册的消费者 secret （客户端 secret）。

3.25.3. GitLab 构建触发器

表 3.28. GitLab 构建触发器

字段	类型	描述
FEATURE_GITLAB_BUILD	布尔值	是否支持 GitLab 构建触发器。 **默认：** False
GITLAB_TRIGGER_CONFIG	对象	使用 Gitlab 进行构建触发器的配置。
.GITLAB_ENDPOINT (Required)	字符串	运行 Gitlab Enterprise 的端点。
.CLIENT_ID (Required)	字符串	此 Red Hat Quay 实例的注册客户端 ID。
.CLIENT_SECRET (Required)	字符串	此 Red Hat Quay 实例的注册客户端 secret。

3.26. 构建管理器配置字段

表 3.29. 构建管理器配置字段

字段	类型	描述
ALLOWED_WORKER_COUNT	字符串	定义每个 Red Hat Quay pod 实例化多少个 Build Workers。通常设置为 **1**。
ORCHESTRATOR_PREFIX	字符串	定义要添加到所有 Redis 密钥的唯一前缀。这可用于将 Orchestrator 值与其他 Redis 键隔离。
REDIS_HOST	对象	Redis 服务的主机名。
REDIS_PASSWORD	字符串	要在 Redis 服务中进行身份验证的密码。
REDIS_SSL	布尔值	定义 Redis 连接是否使用 SSL/TLS。
REDIS_SKIP_KEYSPACE_EVENT_SETUP	布尔值	默认情况下，Red Hat Quay 不会在运行时设置密钥事件所需的 keyspace 事件。为此，请将 **REDIS_SKIP_KEYSPACE_EVENT_SETUP** 设置为 **false**。
EXECUTOR	字符串	启动此类型的可执行文件的定义。有效值为 **kubernetes** 和 **ec2**。
BUILDER_NAMESPACE	字符串	将在其中进行 Red Hat Quay 构建的 Kubernetes 命名空间。
K8S_API_SERVER	对象	构建将在其中进行的 OpenShift Container Platform 集群的 API 服务器的主机名。
K8S_API_TLS_CA	对象	构建集群 CA 证书的 **Quay** 容器中文件路径，供 **Quay** 应用在发出 API 调用时信任。
KUBERNETES_DISTRIBUTION	字符串	指明正在使用的 Kubernetes 类型。有效值为 **openshift** 和 **k8s**。
CONTAINER_*	对象	定义每个构建 Pod 的资源请求和限值。

字段	类型	描述
NODE_SELECTOR_*	对象	定义应调度 **构建** Pod 的节点选择器标签 name-value 对。
CONTAINER_RUNTIME	对象	指定 Builder 是否应该运行 **docker** 或 **podman**。使用红帽 **quay-builder** 镜像的客户应将其设置为 **podman**。
SERVICE_ACCOUNT_NAME/SERVICE_ACCOUNT_TOKEN	对象	定义 **构建** Pod 将使用的服务帐户名称或令牌。
QUAY_USERNAME/QUAY_PASSWORD	对象	定义拉取在 **WORKER_IMAGE** 字段中指定的 Red Hat Quay 构建 worker 镜像所需的 registry 凭证。客户应提供一个 Red Hat Service Account 凭证，如 https://access.redhat.com/RegistryAuthentication 文章中的针对 registry.redhat.io 的"创建 Registry 服务账户"部分。
WORKER_IMAGE	对象	Red Hat Quay Builder 镜像的镜像引用。 registry.redhat.io/quay/quay-builder
WORKER_TAG	对象	Builder 镜像标签。最新版本为 3.12。
BUILDER_VM_CONTAINER_IMAGE	对象	对包含运行每个 Red Hat Quay Build. (**registry.redhat.io/quay/quay-builder-qemu-rhcos:3.12**) 所需的内部虚拟机的完整引用。
SETUP_TIME	字符串	指定构建尚未通过 Build Manager 注册自己时超时的秒数。默认值为 **500** 秒。尝试重启三次的构建。如果构建在三次尝试失败后没有注册自己，则被视为失败。

字段	类型	描述
MINIMUM_RETRY_THRESHOLD	字符串	此设置用于多个可执行文件。它指示在选择其他可执行文件前尝试启动构建的次数。设置为 **0** 表示构建作业需要有多少个尝试。这个值应该被有意保持小（三个或更少），以确保在基础架构故障期间快速发生故障切换。您必须为此设置指定一个值。例如，**Kubernetes** 设置为第一个 executor，**EC2** 设为第二个 executor。如果您希望最后一次尝试运行作业在 EC2 上，而不是 Kubernetes，您可以将 Kubernetes executor 的 **MINIMUM_RETRY_THRESHOLD** 设置为 **1**，EC2 的 **MINIMUM_RETRY_THRESHOLD** 设置为 0（如果没有设置，则默认为 **0**）。在这种情况下，Kubernetes 的 **MINIMUM_RETRY_THRESHOLD retries_remaining (1)** 将评估为 **False**，因此回退到配置的第二个 executor。
SSH_AUTHORIZED_KEYS	对象	**ignition** 配置中 bootstrap 的 SSH 密钥列表。这允许使用其他密钥 SSH 到 EC2 实例或 QEMU 虚拟机(VM)。

3.27. OAUTH 配置字段

表 3.30. OAuth 字段

字段	类型	描述
DIRECT_OAUTH_CLIENTID_WHITELIST	字符串数组	允许在没有用户批准的情况下执行直接 OAuth 批准的 **Quay 管理** 的应用程序的客户端 ID 列表。
FEATURE_ASSIGN_OAUTH_TOKEN	布尔值	允许机构管理员将 OAuth 令牌分配给其他用户。

3.27.1. GitHub OAuth 配置字段

表 3.31. GitHub OAuth 字段

字段	类型	描述
FEATURE_GITHUB_LOGIN	布尔值	是否支持 GitHub 登录 **默认：**False**
GITHUB_LOGIN_CONFIG	对象	使用 GitHub (Enterprise)作为外部登录提供程序的配置。
.ALLOWED_ORGANIZATIONS	字符串数组	GitHub （企业）组织的名称，使用 ORG_RESTRICT 选项。
.API_ENDPOINT	字符串	要使用的 GitHub （企业）API 的端点。对于 github.com ，必须**被覆盖：** **https://api.github.com/**
.CLIENT_ID (Required)	字符串	此 Red Hat Quay 实例的注册客户端 ID; 无法与 **GITHUB_TRIGGER_CONFIG**. **示例：** **0e8dbe15c4c7630b6780**
.CLIENT_SECRET (Required)	字符串	此 Red Hat Quay 实例的注册客户端 secret。 **：** **e4a58ddd3d7408b7aec109e85564a0d153d3e846**
.GITHUB_ENDPOINT (Required)	字符串	GitHub 的端点(Enterprise)。 **示例 ：https://github.com/**
.ORG_RESTRICT	布尔值	如果为 true，只有机构白名单中的用户才能使用这个供应商登录。

3.27.2. Google OAuth 配置字段

表 3.32. Google OAuth 字段

字段	类型	描述
FEATURE_GOOGLE_LOGIN	布尔值	是否支持 Google 登录。 （默认： **False**）

字段	类型	描述
GOOGLE_LOGIN_CONFIG	对象	使用 Google 进行外部身份验证的配置。
.CLIENT_ID (Required)	字符串	此 Red Hat Quay 实例的注册客户端 ID。 **：0e8dbe15c4c7630b6780**
.CLIENT_SECRET (Required)	字符串	此 Red Hat Quay 实例的注册客户端 secret。 **：e4a58ddd3d7408b7aec109e85564a0d153d3e846**

3.28. OIDC 配置字段

表 3.33. OIDC 字段

字段	类型	描述
<string>_LOGIN_CONFIG (Required)	字符串	包含 OIDC 配置设置的父密钥。通常，OIDC 供应商的名称，如 **AZURE_LOGIN_CONFIG**，但接受任何任意字符串。
.CLIENT_ID (Required)	字符串	此 Red Hat Quay 实例的注册客户端 ID。 **：0e8dbe15c4c7630b6780**
.CLIENT_SECRET (Required)	字符串	此 Red Hat Quay 实例的注册客户端 secret。 **：e4a58ddd3d7408b7aec109e85564a0d153d3e846**
.DEBUGLOG	布尔值	是否启用调试。
.LOGIN_BINDING_FIELD	字符串	当内部授权设置为 LDAP 时使用。Red Hat Quay 读取此参数，并尝试使用此用户名为用户搜索 LDAP 树。如果存在，它会自动创建到该 LDAP 帐户的链接。
.LOGIN_SCOPES	对象	添加 Red Hat Quay 用来与 OIDC 供应商通信的其他范围。

.OIDC_ENDPOINT_CUSTOM_PARAMS	字符串	支持 OIDC 端点上的自定义查询参数。支持以下端点：**authorization_endpoint**、**token_endpoint** 和 **user_endpoint**。
.OIDC_ISSUER	字符串	允许用户定义签发者进行验证。例如，JWT 令牌容器(**称为**)参数定义谁签发令牌。默认情况下，这从 **.well-know/openid/configuration** 端点读取，该端点由每个 OIDC 供应商公开。如果这个验证失败，则没有登录。
.OIDC_SERVER (Required)	字符串	用于身份验证的 OIDC 服务器的地址。 **https://sts.windows.net/6c878... /**
.PREFERRED_USERNAME_CLAIM_NAME	字符串	将首选 username 设置为来自令牌的参数。
.SERVICE_ICON	字符串	更改登录屏幕上的图标。
.SERVICE_NAME (Required)	字符串	要验证的服务名称。 **：** **Microsoft**Entra ID
.VERIFIED_EMAIL_CLAIM_NAME	字符串	用于验证用户电子邮件地址的声明名称。
.PREFERRED_GROUP_CLAIM_NAME	字符串	OIDC 令牌有效负载中的密钥名称，其中包含用户组成员资格的信息。
.OIDC_DISABLE_USER_ENDPOINT	布尔值	是否允许或拒绝 **/userinfo** 端点。如果使用 Azure Entra ID，此字段必须设置为 **true**，因为 Azure 从令牌获取用户信息，而不是调用 **/userinfo** 端点。 **默认： false**

3.28.1. OIDC 配置

以下示例显示了 OIDC 配置示例。

OIDC 配置示例

```
AUTHENTICATION_TYPE: OIDC
# ...
AZURE_LOGIN_CONFIG:
    CLIENT_ID: <client_id>
    CLIENT_SECRET: <client_secret>
    OIDC_SERVER: <oidc_server_address_>
    DEBUGGING: true
    SERVICE_NAME: Microsoft Entra ID
    VERIFIED_EMAIL_CLAIM_NAME: <verified_email>
    OIDC_DISABLE_USER_ENDPOINT: true
    OIDC_ENDPOINT_CUSTOM_PARAMS":
            "authorization_endpoint":
               "some": "param",
# ...
```

3.29. 嵌套存储库配置字段

在 **FEATURE_EXTENDED_REPOSITORY_NAMES** 属性下添加了对嵌套存储库路径名称的支持。此可选配置默认添加到 config.yaml 中。启用允许在仓库名称中使用 /。

表 3.34. OCI 和嵌套的存储库配置字段

字段	类型	描述
FEATURE_EXTENDED_REPOSITORY_NAMES	布尔值	启用对嵌套存储库的支持 **默认值：** True

OCI 和嵌套存储库配置示例

```
FEATURE_EXTENDED_REPOSITORY_NAMES: true
```

3.30. QUAYINTEGRATION 配置字段

QuayIntegration 自定义资源提供了以下配置字段：

Name	描述	模式
allowlistNamespaces （可选）	要包含的命名空间列表。	Array
clusterid （必需）	与此集群关联的 ID。	字符串
credentialsSecret.key (Required)	包含与 Quay registry 通信的凭证的 secret。	对象

Name	描述	模式
denylistNamespaces （可选）	要排除的命名空间列表。	Array
insecureRegistry (Optional)	是否将 TLS 验证跳过到 Quay registry	布尔值
quayHostname (Required)	Quay registry 的主机名。	字符串
scheduledImageStreamImport (Optional)	是否启用镜像流导入。	布尔值

3.31. 邮件配置字段

表 3.35. 邮件配置字段

字段	类型	描述
FEATURE_MAILING	布尔值	是否启用电子邮件 **默认值： False**
MAIL_DEFAULT_SENDER	字符串	如果指定了，当 Red Hat Quay 发送电子邮件时，电子邮件地址用作 **from**。如果没有，则默认为 **support@quay.io** **示例： support@example.com**
MAIL_PASSWORD	字符串	发送电子邮件时要使用的 SMTP 密码
MAIL_PORT	Number	要使用的 SMTP 端口。如果没有指定，则默认为 587。
MAIL_SERVER	字符串	用于发送电子邮件的 SMTP 服务器。仅在将 FEATURE_MAILING 设置为 true 时才需要。 Example: **smtp.example.com**
MAIL_USERNAME	字符串	发送电子邮件时要使用的 SMTP 用户名

字段	类型	描述
MAIL_USE_TLS	布尔值	如果指定，是否使用 TLS 发送电子邮件 **默认： True**

3.32. 用户配置字段

表 3.36. 用户配置字段

字段	类型	描述
FEATURE_SUPER_USERS	布尔值	是否支持超级用户 **默认为 true**
FEATURE_USER_CREATION	布尔值	是否可以创建用户（非超级用户） **默认值： true**
FEATURE_USER_LAST_ACCESSED	布尔值	是否记录用户被访问最后一次访问的时间 **默认为 true**
FEATURE_USER_LOG_ACCESS	布尔值	如果设置为 true，用户有权访问其命名空间的审计日志 **默认： false**
FEATURE_USER_METADATA	布尔值	是否收集和支持用户元数据 **默认： false**
FEATURE_USERNAME_CONFIRMATION	布尔值	如果设置为 true，用户可以在通过 OpenID Connect (OIDC)或非数据库内部身份验证供应商（如 LDAP）登录时确认并修改其初始用户名。 **默认： true**
FEATURE_USER_RENAME	布尔值	如果设置为 true，用户可以重命名自己的命名空间 **默认： false**

字段	类型	描述
FEATURE_INVITE_ONLY_USER_CREATION	布尔值	要创建的用户是否必须被其他用户邀请 **默认为 false**
FRESH_LOGIN_TIMEOUT	字符串	新登录的时间需要用户重新输入其密码 **示例 ：5m**
USERFILES_LOCATION	字符串	放置用户上传文件的存储引擎 ID **示例:s3_us_east**
USERFILES_PATH	字符串	放置用户上传文件的存储的路径 **示例 ：userfiles**
USER_RECOVERY_TOKEN_LIFETIME	字符串	恢复用户帐户的令牌时间长度是有效的 Pattern:**^[0-9]+(w|m|d|h|s)$** Default:**30m**
FEATURE_SUPERUSERS_FULL_ACCESS	布尔值	授予超级用户从命名空间中的其他存储库读取、写入和删除它们没有拥有或明确权限的那些存储库的权限。 Default: **False**
FEATURE_SUPERUSERS_ORG_CREATION_ONLY	布尔值	是否只允许超级用户创建机构。 Default: **False**

字段	类型	描述
FEATURE_RESTRICTED_USERS	布尔值	当使用 **RESTRICTED_USERS_WHITELIST** 设置为 **True** 时： • 所有普通用户和超级用户都仅限于在其自己的命名空间中创建机构或内容，除非它们通过 **RESTRICTED_USERS_WHITELIST** 进行允许。 • 受限用户根据团队成员资格在机构中保留其正常权限。 Default: **False**
RESTRICTED_USERS_WHITELIST	字符串	当使用 **FEATURE_RESTRICTED_USERS** 设置设置时，特定用户将不包括在 **FEATURE_RESTRICTED_USERS** 设置中。
GLOBAL_READONLY_SUPER_USERS	字符串	设置后，可授予用户对所有存储库的读取访问权限，无论它们是公共存储库。仅适用于通过 **SUPER_USERS** 配置字段定义的超级用户。

3.32.1. 用户配置字段参考

使用以下引用，使用所需的配置字段更新 **config.yaml** 文件。

3.32.1.1. FEATURE_SUPERUSERS_FULL_ACCESS 配置参考

```
---
SUPER_USERS:
- quayadmin
FEATURE_SUPERUSERS_FULL_ACCESS: True
---
```

3.32.1.2. GLOBAL_READONLY_SUPER_USERS 配置参考

```
---
GLOBAL_READONLY_SUPER_USERS:
    - user1
---
```

3.32.1.3. FEATURE_RESTRICTED_USERS 配置参考

```
---
AUTHENTICATION_TYPE: Database
---
---
FEATURE_RESTRICTED_USERS: true
---
```

3.32.1.4. RESTRICTED_USERS_WHITELIST 配置参考

先决条件

- 在 **config.yaml** 文件中，**FEATURE_RESTRICTED_USERS** 设置为 **true**。

```
---
AUTHENTICATION_TYPE: Database
---
---
FEATURE_RESTRICTED_USERS: true
RESTRICTED_USERS_WHITELIST:
    - user1
---
```

注意

设置此字段后，可以列入白名单用户，也可以从存储库创建或写入内容，即使 **FEATURE_RESTRICTED_USERS** 设置为 **true**。其他用户（如 **user2**、 **user3** 和 **user4**）仅限于创建机构、读取或写入内容

3.33. RECAPTCHA 配置字段

表 3.37. reCAPTCHA 配置字段

字段	类型	描述
FEATURE_RECAPTCHA	布尔值	用户登录和恢复需要 Recaptcha **Default:** False
RECAPTCHA_SECRET_KEY	字符串	如果启用了 recaptcha，则 Recaptcha 服务的 secret 键
RECAPTCHA_SITE_KEY	字符串	如果启用了 recaptcha，则 Recaptcha 服务的站点键

3.34. ACI 配置字段

表 3.38. ACI 配置字段

字段	类型	描述
FEATURE_ACI_CONVERSION	布尔值	是否启用到 ACI **默认值：** False
GPG2_PRIVATE_KEY_FILENAME	字符串	用于解密 ACI 的私钥的文件名
GPG2_PRIVATE_KEY_NAME	字符串	用于为 ACI 签名的私钥名称
GPG2_PUBLIC_KEY_FILENAME	字符串	用于加密 ACI 的公钥的文件名

3.35. JWT 配置字段

表 3.39. JWT 配置字段

字段	类型	描述
JWT_AUTH_ISSUER	字符串	JWT 用户的端点 Pattern:**^http (s)?://(.)+$** Example:**http://192.168.99.101:6060**
JWT_GETUSER_ENDPOINT	字符串	JWT 用户的端点 Pattern:**^http (s)?://(.)+$** **示例:http://192.168.99.101:6060**
JWT_QUERY_ENDPOINT	字符串	JWT 查询的端点 Pattern:**^http (s)?://(.)+$** Example:**http://192.168.99.101:6060**
JWT_VERIFY_ENDPOINT	字符串	JWT 验证的端点 Pattern:**^http (s)?://(.)+$** Example:**http://192.168.99.101:6060**

3.36. 应用令牌配置字段

表 3.40. 应用令牌配置字段

字段	类型	描述

字段	类型	描述
FEATURE_APP_SPECIFIC_TOKENS	布尔值	如果启用，用户可以创建令牌以供 Docker CLI **默认值：** True
APP_SPECIFIC_TOKEN_EXPIRATION	字符串	外部应用令牌的过期时间。 **默认** None **Pattern: ^[0-9]+(w|m|d|h|s)$**
EXPIRED_APP_SPECIFIC_TOKEN_GC	字符串	过期的外部应用程序令牌的持续时间将在垃圾回收前保留 **默认：** 1d

3.37. 其它配置字段

表 3.41. 其它配置字段

字段	类型	描述
ALLOW_PULLS_WITHOUT_STRICT_LOGGING	字符串	如果为 true，则拉取仍然会成功，即使无法写入 pull audit 日志条目。如果数据库处于只读状态，并且需要在此时间段内拉取以继续，这非常有用。 **Default:** False
AVATAR_KIND	字符串	要显示的 avatars 的类型，可以生成内联（本地）或 Gravatar (gravatar) **值：** local, gravatar
BROWSER_API_CALLS_XHR_ONLY	布尔值	如果启用，则只允许被标记为来自浏览器的、由 XHR 发出的 API 调用 **默认：** True
DEFAULT_NAMESPACE_MAXIMUM_BUILD_COUNT	Number	可在命名空间中排队的默认构建的最大数量。 **默认：** None

字段	类型	描述
ENABLE_HEALTH_DEBUG_SECRET	字符串	如果指定，可以授予健康端点的 secret，以便在未作为超级用户验证时查看完整的调试信息
EXTERNAL_TLS_TERMINATION	布尔值	如果支持 TLS，则设为 **true**，但在 Quay 之前的一个层终止。当 Quay 使用自己的 SSL 证书运行时，并直接接收 TLS 流量时，设置为 **false**。
FRESH_LOGIN_TIMEOUT	字符串	新登录的时间需要用户重新输入其密码 **示例： 5m**
HEALTH_CHECKER	字符串	配置的健康检查 **示例： ('RDSAwareHealthCheck', {'access_key': 'foo', 'secret_key': 'bar'})**
PROMETHEUS_NAMESPACE	字符串	适用于所有公开的 Prometheus metrics 的前缀 **默认： quay**
PUBLIC_NAMESPACES	字符串数组	如果命名空间定义在公共命名空间列表中，则它将在 **所有** 用户的存储库列表页上显示，无论用户是否是命名空间的成员。通常，这由企业客户用来配置一组"well-known"命名空间。
REGISTRY_STATE	字符串	registry 的状态 **值： normal** 或 **read-only**
SEARCH_MAX_RESULT_PAGE_COUNT	Number	用户在搜索限制前可以分页的最大页面数 **默认：10**
SEARCH_RESULTS_PER_PAGE	Number	通过搜索页面返回的结果数 **默认值：10**

字段	类型	描述
V2_PAGINATION_SIZE	Number	V2 registry API 中每个页面返回的结果数 **默认值：** 50
WEBHOOK_HOSTNAME_BLACKLIST	字符串数组	验证超过 localhost 时禁止 Webhook 中的主机名集合
CREATE_PRIVATE_REPO_ON_PUSH	布尔值	通过推送创建的新存储库是否设置为私有可见性 **默认为** True
CREATE_NAMESPACE_ON_PUSH	布尔值	新推送到不存在的机构时是否创建它 **默认：** False
NON_RATE_LIMITED_NAMESPACES	字符串数组	如果使用 **FEATURE_RATE_LIMITS** 启用速率限制，您可以为需要无限访问权限的特定命名空间覆盖它。
FEATURE_UI_V2	布尔值	设置后，允许用户尝试 beta UI 环境。 **Default: True**
FEATURE_REQUIRE_TEAM_INVITE	布尔值	将用户添加到团队时是否需要邀请 **默认值：** True
FEATURE_REQUIRE_ENCRYPTED_BASIC_AUTH	布尔值	非加密密码（与加密令牌相反）可用于基本身份验证 **默认值：** False
FEATURE_RATE_LIMITS	布尔值	是否在 API 和 registry 端点上启用速率限值。将 FEATURE_RATE_LIMITS 设置为 **true** 会导致 **nginx** 将某些 API 调用限制为每秒 30 个。如果没有设置该功能，API 调用将限制为每秒 300 个（具有无限限制）。 **Default:** False

字段	类型	描述
FEATURE_FIPS	布尔值	如果设置为 true，Red Hat Quay 将使用 FIPS 兼容哈希功能运行 **Default:** False
FEATURE_AGGREGATED_LOG_COUNT_RETRIEVAL	布尔值	是否允许检索聚合日志计数 **默认：** True
FEATURE_ANONYMOUS_ACCESS	布尔值	是否允许匿名用户浏览和拉取公共存储库 **默认：** True
FEATURE_DIRECT_LOGIN	布尔值	用户是否可以直接登录到 UI **默认值：** True
FEATURE_LIBRARY_SUPPORT	布尔值	从 Docker 拉取和推送时是否允许"无命名空间"软件仓库 **默认值：** True
FEATURE_PARTIAL_USER_AUTOCOMPLETE	布尔值	如果设置为 true，自动完成功能将应用到部分用户名+ **默认：** True
FEATURE_PERMANENT_SESSIONS	布尔值	会话是永久的 **默认值：** True
FEATURE_PUBLIC_CATALOG	布尔值	如果设置为 true，则 **_catalog** 端点会返回公共存储库。否则，只能返回私有存储库。 **Default:** False

3.38. 旧配置字段

以下字段已弃用或过时。

表 3.42. 旧配置字段

字段	类型	描述
FEATURE_BLACKLISTED_EMAILS	布尔值	如果设置为 true，则在其电子邮件域黑名单时无法创建新的用户帐户

字段	类型	描述
BLACKLISTED_EMAIL_DOMAINS	字符串数组	如果 FEATURE_BLACKLISTED_EMAILS 设置为 true ，则使用 email-address 域列表：**"example.com", "example.org"**
BLACKLIST_V2_SPEC	字符串	Red Hat Quay 将响应的 Docker CLI 版本将响应 V2 是 **不支持的** **示例** ：<**1.8.0** **Default:** &lt **;1.6.0**
DOCUMENTATION_ROOT	字符串	文档链接的根 URL。当将 Red Hat Quay 配置为断开连接的环境来设置或允许列表的文档链接时，此字段很有用。
SECURITY_SCANNER_V4_NAMESPACE_WHITELIST	字符串	应该启用安全扫描程序的命名空间
FEATURE_RESTRICTED_V1_PUSH	布尔值	如果设置为 true，则只有 V1_PUSH_WHITELIST 中列出的命名空间支持 V1 push **Default: True**
V1_PUSH_WHITELIST	字符串数组	如果 FEATURE_RESTRICTED_V1_PUSH 设置为 true，支持 V1 push 的命名空间名称的数组
FEATURE_HELM_OCI_SUPPORT	布尔值	启用对 Helm 工件的支持。 **默认： False**
ALLOWED_OCI_ARTIFACT_TYPES	对象	允许 OCI 工件 MIME 类型和关联的层类型的集合。

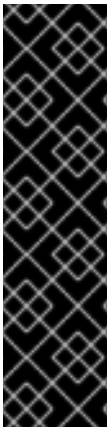
3.39. 用户界面 V2 配置字段

表 3.43. 用户界面 v2 配置字段

字段	类型	描述
FEATURE_UI_V2	布尔值	设置后，允许用户尝试 beta UI 环境。 + **默认：** **False**
FEATURE_UI_V2_REPO_SETTINGS	布尔值	当设置为 **True** 时，在 Red Hat Quay v2 UI 中启用存储库设置。 + **默认：** **False**

3.39.1. v2 用户界面配置

启用 **FEATURE_UI_V2** 后，您可以在用户界面的当前版本和用户界面的新版本间切换。



重要

- 这个 UI 目前处于 beta 阶段，可能会有变化。在当前状态中，用户只能创建、查看和删除机构、存储库和镜像标签。
- 在旧的 UI 中运行 Red Hat Quay 时，超时会话要求用户在弹出窗口中再次输入密码。使用新的 UI 时，用户会返回主页，需要输入其用户名和密码凭证。这是一个已知问题，将在新 UI 的未来版本中修复。
- 在传统 UI 和新 UI 之间如何报告镜像清单大小的方式存在差异。在传统的 UI 中，镜像清单以兆字节为单位报告。在新 UI 中，Red Hat Quay 使用标准定义 megabyte (MB)来报告镜像清单大小。

流程

1. 在部署的 **config.yaml** 文件中，添加 **FEATURE_UI_V2** 参数并将其设置为 **true**，例如：

```
---
FEATURE_TEAM_SYNCING: false
FEATURE_UI_V2: true
FEATURE_USER_CREATION: true
---
```

2. 登录到您的 Red Hat Quay 部署。
3. 在 Red Hat Quay 部署的导航窗格中，会给一个可以在 **Current UI** 和 **New UI** 之间切换的选项。点击切换按钮将其设置为新 UI，然后点 **Use Beta Environment**，例如：

3.40. IPV6 配置字段

表 3.44. IPv6 配置字段

字段	类型	描述
FEATURE_LISTEN_IP_VERSION	字符串	启用 IPv4、IPv6 或双栈协议系列。必须正确设置此配置字段，否则 Red Hat Quay 无法启动。 **默认： IPv4** **其他配置： IPv6、双栈**

3.41. 品牌配置字段

表 3.45. 品牌配置字段

字段	类型	Description
品牌	对象	在 Red Hat Quay UI 中自定义徽标和 URL 的品牌。
.logo (Required)	字符串	主徽标图像 URL. 标头徽标默认为 205x30 PX。Web UI 的屏幕中的 Red Hat Quay 符号上的表单徽标默认为 356.5x39.7 PX。 **示例：** **/static/img/quay-horizontal-color.svg**
.footer_img	字符串	UI 页徽标.默认为 144x34 PX. **示例：** **/static/img/RedHat.svg**
.footer_url	字符串	footer 镜像的链接。 **示例：** **https://redhat.com**

3.41.1. Red Hat Quay 品牌配置示例

品牌 **config.yaml** 示例

```
BRANDING:
    logo: https://www.mend.io/wp-content/media/2020/03/5-tips_small.jpg
    footer_img: https://www.mend.io/wp-content/media/2020/03/5-tips_small.jpg
    footer_url: https://opensourceworld.org/
```

3.42. 会话超时配置字段

以下配置字段依赖于相同名称的 Flask API 配置字段。

表 3.46. 会话注销配置字段

字段	类型	描述
PERMANENT_SESSION_LIFETIME	整数	一个 **timedelta**，用于设置永久会话的过期日期。默认值为 31 天，这会在大约一个月内保留永久会话。 **默认：** **2678400**

3.42.1. 会话超时配置示例

以下 YAML 是启用会话生命周期的建议配置。

重要

不建议更改会话生命周期。管理员应注意设置会话超时时的分配时间。如果设置时间过早，它可能会中断您的工作流。

会话超时 **YAML** 配置

```
PERMANENT_SESSION_LIFETIME: 3000
```

第 4 章 环境变量

Red Hat Quay 支持有限的环境变量用于动态配置。

4.1. GEO-REPLICATION

所有区域都应使用相同的配置，但存储后端除外，这可以使用 **QUAY_DISTRIBUTED_STORAGE_PREFERENCE** 环境变量明确进行配置。

表 4.1. 异地复制配置

变量	类型	描述
QUAY_DISTRIBUTED_STORAGE_PREFERENCE	字符串	首选存储引擎（通过 ID (DISTRIBUTED_STORAGE_CONFIG)使用。

4.2. 数据库连接池

Red Hat Quay 由很多不同的进程组成，它们都在同一个容器中运行。其中许多进程与数据库交互。

默认启用数据库连接池，与数据库交互的每个进程都包含连接池。这些每个进程连接池配置为最多维护 20 个连接。在负载过重时，可以填写 Red Hat Quay 容器中每个进程的连接池。在某些部署和负载下，这可能需要分析以确保 Red Hat Quay 不会超过配置的数据库的最大连接数。

overtime，连接池释放闲置连接。要立即释放所有连接，Red Hat Quay 需要重启。

对于独立 Red Hat Quay 部署，在启动部署时可以关闭数据库连接池。例如：

```
$ sudo podman run -d --rm -p 80:8080 -p 443:8443  \
  --name=quay \
  -v $QUAY/config:/conf/stack:Z \
  -v $QUAY/storage:/datastorage:Z \
  -e DB_CONNECTION_POOLING=false
  registry.redhat.io/quay/quay-rhel8:v3.12.1
```

对于 OpenShift Container Platform 上的 Red Hat Quay，可以通过修改 **QuayRegistry** 自定义资源定义(CRD)来配置数据库连接池。例如：

QuayRegistry CRD 示例

```
spec:
  components:
  - kind: quay
    managed: true
    overrides:
      env:
      - name: DB_CONNECTION_POOLING
        value: "false"
```

表 4.2. 数据库连接池配置

变量	类型	描述
DB_CONNECTION_POOLING	字符串	是否要启用或禁用数据库连接池。默认值为 true。接受的值是 **"true"** 或 **"false"**

如果启用了数据库连接池，可以更改连接池的最大大小。这可以通过以下 **config.yaml** 选项完成：

config.yaml

```
...
DB_CONNECTION_ARGS:
  max_connections: 10
...
```

4.3. HTTP 连接数

可以使用环境变量指定同时 HTTP 连接的数量。它们可以作为整个组件或特定组件指定。每个进程的默认值为 **50** 个并行连接。

表 4.3. HTTP 连接计数配置

变量	类型	描述
WORKER_CONNECTION_COUNT	Number	同步 HTTP 连接 **默认：** 50
WORKER_CONNECTION_COUNT_REGISTRY	Number	registry 的同步 HTTP 连接 **默认：** WORKER_CONNECTION_COUNT
WORKER_CONNECTION_COUNT_WEB	Number	Web UI 的同步 HTTP 连接 **默认值：** WORKER_CONNECTION_COUNT
WORKER_CONNECTION_COUNT_SECSCAN	Number	Clair 的同步 HTTP 连接 **默认值：** WORKER_CONNECTION_COUNT

4.4. WORKER 数量变量

表 4.4. worker 数量变量

变量	类型	描述
WORKER_COUNT	Number	进程数量的通用覆盖
WORKER_COUNT_REGISTRY	Number	指定在 **Quay** 容器中处理 Registry 请求的进程数量 **值：** **8** 到 **64**之间的整数
WORKER_COUNT_WEB	Number	指定在容器内处理 UI/Web 请求的进程数 **值：** **2** 到 **32**之间的整数
WORKER_COUNT_SECSCAN	Number	指定在容器内处理安全扫描（如 Clair）集成的进程数 **值：** 整数。因为 Operator 为资源请求和限值指定 2 个 vCPU，所以在 **2** 和 **4** 之间设置这个值是安全的。但是，如果保证，用户可以运行更多，例如 **16**。

4.5. 调试变量

Red Hat Quay 上提供了以下调试变量：

表 4.5. 调试配置变量

变量	类型	描述
DEBUGLOG	布尔值	是否启用或禁用调试日志。

变量	类型	描述
USERS_DEBUG	整数.**0** 或 **1**。	用于以明文中调试 LDAP 操作，包括密码。必须与 **DEBUGLOG=TRUE** 一起使用。  重要 设置 **USERS_DEBUG=1** 以明文形式公开凭据。该变量应该在调试后从 Red Hat Quay 部署中删除。使用此环境变量生成的日志文件应该被清理，在发送到其他用户前应删除密码。请谨慎使用。

第 5 章 CLAIR 安全扫描程序

5.1. CLAIR 配置概述

Clair 由一个结构化的 YAML 文件配置。每个 Clair 节点都需要指定在其中运行的模式，以及通过 CLI 标记或环境变量配置文件的路径。例如：

```
$ clair -conf ./path/to/config.yaml -mode indexer
```

或者

```
$ clair -conf ./path/to/config.yaml -mode matcher
```

上述命令各自使用相同的配置文件启动两个 Clair 节点。一个运行索引设施，另一个则运行匹配的设施。

如果您以 **combo** 模式运行 Clair，则必须在配置中提供索引器、匹配器和通知程序配置块。

5.1.1. 有关在代理环境中使用 Clair 的信息

如果需要，可以指定 Go 标准库所遵守的环境变量，例如：

- **HTTP_PROXY**

  ```
  $ export HTTP_PROXY=http://<user_name>:<password>@<proxy_host>:<proxy_port>
  ```

- **HTTPS_PROXY**。

  ```
  $ export HTTPS_PROXY=https://<user_name>:<password>@<proxy_host>:<proxy_port>
  ```

- **SSL_CERT_DIR**

  ```
  $ export SSL_CERT_DIR=/<path>/<to>/<ssl>/<certificates>
  ```

- **NO_PROXY**

  ```
  $ export NO_PROXY=<comma_separated_list_of_hosts_and_domains>
  ```

如果您在带有 Clair 的更新器 URL 环境中使用代理服务器，您必须识别哪些 URL 需要添加到代理 allowlist 中，以确保 Clair 可以访问它们。例如，**osv** 更新器需要访问 **https://osv-vulnerabilities.storage.googleapis.com** 来获取生态系统数据转储。在这种情况下，URL 必须添加到代理允许列表中。有关更新器 URL 的完整列表，请参阅 "Clair updater URL"。

您还必须确保将标准 Clair URL 添加到代理 allowlist 中：

- **https://search.maven.org/solrsearch/select**
- **https://catalog.redhat.com/api/containers/**
- **https://access.redhat.com/security/data/metrics/repository-to-cpe.json**
- **https://access.redhat.com/security/data/metrics/container-name-repos-map.json**

在配置代理服务器时，请考虑在 Clair 和这些 URL 之间启用无缝通信所需的任何身份验证要求或特定的代理设置。通过全面记录和解决这些注意事项，您可以在通过代理路由其更新器流量时，确保 Clair 功能有效地路由。

5.1.2. Clair 配置参考

以下 YAML 显示了一个 Clair 配置示例：

```
http_listen_addr: ""
introspection_addr: ""
log_level: ""
tls: {}
indexer:
    connstring: ""
    scanlock_retry: 0
    layer_scan_concurrency: 5
    migrations: false
    scanner: {}
    airgap: false
matcher:
    connstring: ""
    indexer_addr: ""
    migrations: false
    period: ""
    disable_updaters: false
    update_retention: 2
matchers:
    names: nil
    config: nil
updaters:
    sets: nil
    config: nil
notifier:
    connstring: ""
    migrations: false
    indexer_addr: ""
    matcher_addr: ""
    poll_interval: ""
    delivery_interval: ""
    disable_summary: false
    webhook: null
    amqp: null
    stomp: null
auth:
  psk: nil
trace:
    name: ""
    probability: null
    jaeger:
        agent:
            endpoint: ""
        collector:
            endpoint: ""
            username: null
            password: null
        service_name: ""
```

```
        tags: nil
        buffer_max: 0
metrics:
    name: ""
    prometheus:
        endpoint: null
    dogstatsd:
        url: ""
```

注意

为了完整，以上 YAML 文件列出了每个键。如原样使用此配置文件将导致某些选项未正常设置它们的默认值。

5.1.3. Clair 常规字段

下表描述了 Clair 部署可用的通用配置字段。

字段	Typhttp_listen_ae	描述
http_listen_addr	字符串	配置公开 HTTP API 的位置。 默认： **:6060**
introspection_addr	字符串	配置 Clair 的指标和健康端点在哪里。
log_level	字符串	设置日志记录级别。需要以下字符串之一： **debug-color,debug,info,warn,error,fatal,panic**
tls	字符串	包含提供 TLS/SSL 和 HTTP/2 的 HTTP API 配置的映射。
.cert	字符串	要使用的 TLS 证书。必须是 full-chain 证书。

常规 Clair 字段配置示例
以下示例显示了 Clair 配置。

常规 Clair 字段配置示例

```
# ...
http_listen_addr: 0.0.0.0:6060
introspection_addr: 0.0.0.0:8089
log_level: info
# ...
```

5.1.4. Clair 索引器配置字段

下表描述了 Clair 的 **indexer** 组件的配置字段。

字段	类型	描述
indexer	对象	提供 Clair 索引器节点配置。
.airgap	布尔值	为索引和获取者禁用对互联网的 HTTP 访问。允许私有 IPv4 和 IPv6 地址。数据库连接不受影响。
.connstring	字符串	Postgres 连接字符串。接受格式为 URL 或 libpq 连接字符串。
.index_report_request_concurrency	整数	速率限制索引报告创建请求的数量。把它设置为 **0** 时，以自动调整这个值的大小。设置负值表示无限。自动大小是可用内核数的倍数。 如果超过并发，API 会返回 **429** 状态代码。
.scanlock_retry	整数	一个代表 秒的正整数。在清单扫描时并发索引器锁定，以避免冲突。这个值调整等待索引器轮询锁定的频率。
.layer_scan_concurrency	整数	正整数限制并发层扫描的数量。Indexers 将同时匹配清单的层。这个值调整索引程序并行扫描的层数。
.migrations	布尔值	索引节点处理迁移到其数据库。
.scanner	字符串	索引器配置。 扫描程序允许将配置选项传递给层扫描程序。如果设计于这样做，扫描程序会将此配置传递给它。
.scanner.dist	字符串	具有特定扫描程序的名称和任意 YAML 作为值的映射。
.scanner.package	字符串	具有特定扫描程序的名称和任意 YAML 作为值的映射。
.scanner.repo	字符串	具有特定扫描程序的名称和任意 YAML 作为值的映射。

索引器配置示例

以下示例显示了 Clair 的假设索引器配置。

索引器配置示例

```
# ...
indexer:
  connstring: host=quay-server.example.com port=5433 dbname=clair user=clairuser
password=clairpass sslmode=disable
  scanlock_retry: 10
  layer_scan_concurrency: 5
  migrations: true
# ...
```

5.1.5. Clair 匹配器配置字段

下表描述了 Clair 的 **matcher** 组件的配置字段。

注意

与 **matchers** 配置字段不同。

字段	类型	描述
matcher	对象	提供 Clair 匹配器节点配置。
.cache_age	字符串	控制要提示用户缓存响应的时长。
.connstring	字符串	Postgres 连接字符串。接受格式为 URL 或 libpq 连接字符串。
.max_conn_pool	整数	限制数据库连接池大小。 Clair 允许自定义连接池大小。这个数字直接设定同时允许的活跃数据库连接的数量。 以后的版本将忽略此参数。用户应该通过连接字符串进行配置。
.indexer_addr	字符串	匹配者联系一个索引程序来创建漏洞报告。此索引器的位置是必需的。 默认值为 **30m**。
.migrations	布尔值	匹配者节点处理迁移到其数据库。
.period	字符串	决定新安全公告的更新频率。 默认值为 **30m**。

字段	类型	描述
.disable_updaters	布尔值	是否运行后台更新。 Default: **False**
.update_retention	整数	设置在垃圾回收周期之间保留的更新操作数量。这应该设置为基于数据库大小限制的安全 MAX 值。 默认值为 **10m**。 如果提供的值小于 **0,** 则禁用垃圾回收。**2** 是确保将更新与通知进行比较的最小值。

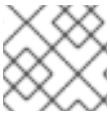
matcher 配置示例

matcher 配置示例

```
# ...
matcher:
  connstring: >-
    host=<DB_HOST> port=5432 dbname=<matcher> user=<DB_USER> password=D<B_PASS>
    sslmode=verify-ca sslcert=/etc/clair/ssl/cert.pem sslkey=/etc/clair/ssl/key.pem
    sslrootcert=/etc/clair/ssl/ca.pem
  indexer_addr: http://clair-v4/
  disable_updaters: false
  migrations: true
  period: 6h
  update_retention: 2
# ...
```

5.1.6. Clair 匹配器配置字段

下表描述了 Clair 的 **matchers** 组件的配置字段。



注意

与 **matcher** 配置字段不同。

表 5.1. matchers 配置字段

字段	类型	描述
matchers	字符串数组	为树内 **matchers** 提供配置。

字段	类型	描述
.names	字符串	一个字符串值列表，用于告知匹配者工厂关于启用的匹配者信息。如果值设为 **null**，则运行匹配者的默认列表。以下字符串被接受：**alpine-matcher**,**aws-matcher**,**debian-matcher**,**gobin**,**java-maven**,**oracle**,**photon**,**python**,**rhel**,**rhel-container-matcher**,**ruby**,**suse**,**ubuntu-matcher**
.config	字符串	提供特定匹配器的配置。 由包含子对象的 matcher 名称键的映射，它将提供给 matchers 工厂构造器。例如：

matchers 配置示例
以下示例显示了一个假设 Clair 部署，它只需要 **alpine**,**aws**,**debian**,**oracle** matchers。

matchers 配置示例

```
# ...
matchers:
  names:
  - "alpine-matcher"
  - "aws"
  - "debian"
  - "oracle"
# ...
```

5.1.7. Clair 更新器配置字段

下表描述了 Clair **更新器**组**件的配置字段**。

表 5.2. 更新器配置字段

字段	类型	描述
Updaters	对象	为匹配者的更新管理器提供配置。

字段	类型	描述
.sets	字符串	一个值列表，告知更新管理器要运行的更新程序。 如果值设为 **null**，则默认的更新程序集合运行如下：**alpine,aws,clair.cvss,debian,oracle,photon,osv,rhel,rhcc suse,ubuntu** 如果留空，则运行零更新器。
.config	字符串	提供特定更新器集的配置。 由 updater 集合的名称键，包含将提供给更新器集的子对象。有关每个更新器的子对象列表，请参阅"高级更新器配置"。

更新器配置示例
在以下配置中，仅配置 **rhel** set。也会定义特定于 **rhel** updater 的 **ignore_unpatched** 变量。

更新器配置示例

```
# ...
updaters:
  sets:
    - rhel
  config:
    rhel:
      ignore_unpatched: false
# ...
```

5.1.8. Clair 通知程序配置字段

Clair 的一般通知配置字段如下所示。

字段	类型	描述
通知程序	对象	提供 Clair 通知程序节点配置。
.connstring	字符串	Postgres 连接字符串。接受格式为 URL 或 libpq 连接字符串。
.migrations	布尔值	通知程序节点处理迁移到其数据库。

字段	类型	描述
.indexer_addr	字符串	通知程序会联系一个索引程序来创建或获取受漏洞影响的清单。此索引器的位置是必需的。
.matcher_addr	字符串	通知程序联系一个匹配器来列出更新操作并获取 diffs。此匹配器的位置是必需的。
.poll_interval	字符串	通知程序将查询与更新操作的匹配者的频率。
.delivery_interval	字符串	通知程序尝试发送创建或之前失败通知的频率。
.disable_summary	布尔值	控制每个清单是否应当向其中一个通知进行汇总。

通知程序配置示例
以下 **通知程序** 片断用于最小配置。

通知程序配置示例

```
# ...
notifier:
  connstring: >-
    host=DB_HOST port=5432 dbname=notifier user=DB_USER password=DB_PASS
    sslmode=verify-ca sslcert=/etc/clair/ssl/cert.pem sslkey=/etc/clair/ssl/key.pem
    sslrootcert=/etc/clair/ssl/ca.pem
  indexer_addr: http://clair-v4/
  matcher_addr: http://clair-v4/
  delivery_interval: 5s
  migrations: true
  poll_interval: 15s
  webhook:
    target: "http://webhook/"
    callback: "http://clair-notifier/notifier/api/v1/notifications"
    headers: ""
  amqp: null
  stomp: null
# ...
```

5.1.8.1. Clair Webhook 配置字段

以下 webhook 字段可用于 Clair notifier 环境。

表 5.3. Clair Webhook 字段

.webhook	对象	为 webhook 发送配置通知程序。

.webhook.target	字符串	提供 webhook 的 URL。
.webhook.callback	字符串	可以检索通知的回调 URL。通知 ID 将附加到此 URL。 这通常是托管 Clair notifier 的位置。
.webhook.headers	字符串	将标头名称与值列表关联的映射。

Webhook 配置示例

Webhook 配置示例

```
# ...
notifier:
# ...
  webhook:
    target: "http://webhook/"
    callback: "http://clair-notifier/notifier/api/v1/notifications"
# ...
```

5.1.8.2. Clair amqp 配置字段

以下高级消息队列协议(AMQP)字段可用于 Clair 通知程序环境。

.amqp	对象	为 AMQP 交付配置通知程序。 [注意] ==== Clair 不自行声明任何 AMQP 组件。所有尝试使用交换或队列的尝试都只是被动，并且会失败。代理管理员应提前设置交换和队列。====
.amqp.direct	布尔值	如果为 **true**，则通知程序会向配置的 AMQP 代理提供单独的通知（而不是回调）。
.amqp.rollup	整数	当 **amqp.direct** 设为 **true** 时，这个值会通知通知程序在直接发送中发送了多少通知。例如，如果 **direct** 设为 **true**，并且 **amqp.rollup** 设为 **5**，则通知程序在单个 JSON 有效负载中不再提供 5 通知到代理。将值设为 **0** 实际的效果是将其设置为 **1**。
.amqp.exchange	对象	要连接的 AMQP 交换。
.amqp.exchange.name	字符串	要连接的交换的名称。

.amqp.exchange.type	字符串	交换的类型。通常，以下之一：**direct**,**fanout**,**topic**,**标头**.
.amqp.exchange.durability	布尔值	配置的队列是否持久。
.amqp.exchange.auto_delete	布尔值	配置的队列是否使用 **auto_delete_policy**。
.amqp.routing_key	字符串	每个通知发送的路由密钥的名称。
.amqp.callback	字符串	如果 **amqp.direct** 设为 **false**，则会在发送到代理的通知回调中提供此 URL。此 URL 应指向 Clair 的通知 API 端点。
.amqp.uris	字符串	按优先级顺序连接的一个或多个 AMQP 代理的列表。
.amqp.tls	对象	配置到 AMQP 代理的 TLS/SSL 连接。
.amqp.tls.root_ca	字符串	可以读取 root CA 的文件系统路径。
.amqp.tls.cert	字符串	可以读取 TLS/SSL 证书的文件系统路径。 [注意] ==== Clair 还允许 **SSL_CERT_DIR**，如 Go **crypto/x509** 软件包所记录。 ====
.amqp.tls.key	字符串	可以读取 TLS/SSL 私钥的文件系统路径。

AMQP 配置示例
以下示例显示了 Clair 的假设 AMQP 配置。

AMQP 配置示例

```
# ...
notifier:
# ...
  amqp:
    exchange:
        name: ""
        type: "direct"
        durable: true
        auto_delete: false
    uris: ["amqp://user:pass@host:10000/vhost"]
    direct: false
```

```
    routing_key: "notifications"
    callback: "http://clair-notifier/notifier/api/v1/notifications"
    tls:
     root_ca: "optional/path/to/rootca"
     cert: "madatory/path/to/cert"
     key: "madatory/path/to/key"
# ...
```

5.1.8.3. Clair STOMP 配置字段

以下简单文本介绍的消息协议(STOMP)字段可用于 Clair 通知程序环境。

.stomp	对象	为 STOMP 交付配置通知程序。
.stomp.direct	布尔值	如果为 **true**，则通知程序会向配置的 STOMP 代理提供单独的通知（而非回调）。
.stomp.rollup	整数	如果 **stomp.direct** 设为 **true**，则这个值会限制在单个直接发送中发送的通知数量。例如，如果 **direct** 设置为 **true**，并且 **rollup** 设为 **5**，则通知程序在单个 JSON 有效负载中不会提供 5 通知到代理。将值设为 **0** 实际的效果是将其设置为 **1**。
.stomp.callback	字符串	如果 **stomp.callback** 设为 **false**，则通知回调中提供的 URL 将发送到代理。此 URL 应指向 Clair 的通知 API 端点。
.stomp.destination	字符串	向其发送通知的 STOMP 目的地。
.stomp.uris	字符串	以优先级顺序连接到的一个或多个 STOMP 代理的列表。
.stomp.tls	对象	配置了到 STOMP 代理的 TLS/SSL 连接。
.stomp.tls.root_ca	字符串	可以读取 root CA 的文件系统路径。 [注意] ==== Clair 还尊重 **SSL_CERT_DIR**，如 Go **crypto/x509** 软件包所述。====
.stomp.tls.cert	字符串	可以读取 TLS/SSL 证书的文件系统路径。

.stomp	对象	为 STOMP 交付配置通知程序。
.stomp.tls.key	字符串	可以读取 TLS/SSL 私钥的文件系统路径。
.stomp.user	字符串	配置 STOMP 代理的登录详情。
.stomp.user.login	字符串	要连接的 STOMP 登录。
.stomp.user.passcode	字符串	要连接的 STOMP 传递码。

STOMP 配置示例
以下示例显示了 Clair 的假设 STOMP 配置。

STOMP 配置示例

```
# ...
notifier:
# ...
  stomp:
    desitnation: "notifications"
    direct: false
    callback: "http://clair-notifier/notifier/api/v1/notifications"
    login:
      login: "username"
      passcode: "passcode"
    tls:
     root_ca: "optional/path/to/rootca"
     cert: "madatory/path/to/cert"
     key: "madatory/path/to/key"
# ...
```

5.1.9. Clair 授权配置字段

以下授权配置字段可用于 Clair。

字段	类型	描述
auth	对象	定义 Clair 的外部和基于服务 JWT 的身份验证。如果定义了多个 **auth** 机制，Clair 会选择一个。目前，不支持多个机制。
.psk	字符串	定义预共享密钥身份验证。
.psk.key	字符串	在所有方签名和验证 JWT 之间分发的 base64 编码密钥。

字段	类型	描述
.psk.iss	字符串	要进行验证的 JWT 签发者列表。空列表接受 JWT 声明中的任何签发者。

授权配置示例
以下 **授权** 片段用于最小配置。

授权配置示例

```
# ...
auth:
  psk:
    key: MTU5YzA4Y2ZkNzJoMQ== 1
    iss: ["quay"]
# ...
```

5.1.10. Clair trace 配置字段

以下 trace 配置字段可用于 Clair。

字段	类型	描述
trace	对象	根据 OpenTelemetry 定义分布式追踪配置。
.name	字符串	应用程序 trace 的名称将属于。
.probability	整数	将发生 trace 的可能性。
.jaeger	对象	定义 Jaeger tracing 的值。
.jaeger.agent	对象	定义用于配置发送到 Jaeger 代理的值。
.jaeger.agent.endpoint	字符串	可以在 < **host>:<post** > 语法中提交 trace 的地址。
.jaeger.collector	对象	为配置发送到 Jaeger 收集器定义值。
.jaeger.collector.endpoint	字符串	可以在 < **host>:<post** > 语法中提交 trace 的地址。
.jaeger.collector.username	字符串	Jaeger 用户名。
.jaeger.collector.password	字符串	Jaeger 密码。

字段	类型	描述
.jaeger.service_name	字符串	在 Jaeger 中注册的服务名称。
.jaeger.tags	字符串	用于提供额外的元数据的键值对。
.jaeger.buffer_max	整数	在将内存发送到 Jaeger 后端以进行存储和分析前，可以缓冲的最大 span 数量。

trace 配置示例
以下示例显示了 Clair 的假设跟踪配置。

trace 配置示例

```
# ...
trace:
  name: "jaeger"
  probability: 1
  jaeger:
    agent:
      endpoint: "localhost:6831"
    service_name: "clair"
# ...
```

5.1.11. Clair 指标配置字段

以下指标配置字段可用于 Clair。

字段	类型	描述
metrics	对象	根据 OpenTelemetry 定义分布式追踪配置。
.name	字符串	使用的指标的名称。
.prometheus	字符串	配置 Prometheus 指标导出器。
.prometheus.endpoint	字符串	定义提供指标的路径。

指标配置示例
以下示例显示了 Clair 的假设指标配置。

指标配置示例

```
# ...
metrics:
  name: "prometheus"
```

```
  prometheus:
    endpoint: "/metricsz"
# ...
```