Red Hat OpenShift Pipelines 1.15

사용자 정의 Tekton Hub 인스턴스

Tekton Hub의 사용자 정의 인스턴스 설치

Last Updated: 2024-09-02

Red Hat OpenShift Pipelines 1.15 사용자 정의 Tekton Hub 인스턴스

Tekton Hub의 사용자 정의 인스턴스 설치

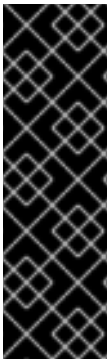
법적 공지

초록

이 문서에서는 Tekton Hub의 사용자 지정 인스턴스 설치 및 배포에 대한 정보를 제공합니다.

차례

1장. OPENSHIFT PIPELINES와 함께 TEKTON HUB 사용



중요

Tekton Hub는 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 Red Hat 프로덕션 서비스 수준 계약(SLA)에서 지원되지 않으며 기능적으로 완전하지 않을 수 있습니다. 따라서 프로덕션 환경에서 사용하는 것은 권장하지 않습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다.

Red Hat 기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위를 참조하십시오.

Tekton Hub를 사용하면 CI/CD 워크플로에 대해 재사용 가능한 작업과 파이프라인을 검색, 검색 및 공유할 수 있습니다. Tekton Hub의 공용 인스턴스는 hub.tekton.dev 에서 사용할 수 있습니다. 클러스터 관리자는 **TektonHub** CR(사용자 정의 리소스)의 구성을 수정하여 Tekton Hub의 사용자 정의 인스턴스를 설치하고 배포할 수도 있습니다.

1.1. OPENSHIFT CONTAINER PLATFORM 클러스터에 TEKTON HUB 설치 및 배포

Tekton Hub는 선택적 구성 요소입니다. 클러스터 관리자는 **TektonConfig** CR(사용자 정의 리소스)을 사용하여 설치할 수 없습니다. Tekton Hub를 설치하고 관리하려면 **TektonHub** CR을 사용합니다.

다음 두 가지 모드를 사용하여 클러스터에 Tekton Hub를 설치할 수 있습니다.

- *로그인* 인증 없이 Tekton Hub 아티팩트에 대한 등급
- Tekton Hub 아티팩트에 대한 로그인 권한 부여 및 등급 사용

참고

Github Enterprise 또는 Gitlab Enterprise를 사용하는 경우 엔터프라이즈 서버와 동일한 네트워크에 Tekton Hub를 설치 및 배포합니다. 예를 들어 엔터프라이즈 서버가 VPN 뒤에서 실행 중인 경우 VPN 뒤의 클러스터에 Tekton Hub를 배포합니다.

1.1.1. 로그인 및 평가 없이 Tekton Hub 설치

기본 구성으로 클러스터에 Tekton Hub를 자동으로 설치할 수 있습니다. 기본 구성을 사용하는 경우 Tekton Hub는 Tekton Hub 아티팩트에 대한 권한 부여 및 등급으로 로그인할 수 없습니다.

사전 요구 사항

- *Red Hat OpenShift Pipelines Operator가 클러스터의 기본 **openshift-pipelines** 네임스페이스에 설치되어 있는지 확인합니다.*

프로세스

1. *다음 예와 유사한 **TektonHub** CR을 생성합니다.*

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonHub
metadata:
```

```
  name: hub
spec:
  targetNamespace: openshift-pipelines (1)
  db:                 # Optional: If you want to use custom database
    secret: tekton-hub-db  # Name of db secret should be `tekton-hub-db`

  categories:             # Optional: If you want to use custom categories
    - Automation
    - Build Tools
    - CLI
    - Cloud
    - Code Quality
    - ...

  catalogs:             # Optional: If you want to use custom catalogs
    - name: tekton
      org: tektoncd
      type: community
      provider: github
      url: https://github.com/tektoncd/catalog
      revision: main

  scopes:               # Optional: If you want to add new users
    - name: agent:create
      users: [abc, qwe, pqr]
    - name: catalog:refresh
      users: [abc, qwe, pqr]
    - name: config:refresh
      users: [abc, qwe, pqr]

  default:              # Optional: If you want to add custom default scopes
    scopes:
      - rating:read
      - rating:write

  api:
    catalogRefreshInterval: 30m (2)
```

(1) Tekton Hub를 설치해야 하는 네임스페이스입니다. 기본값은 **openshift-pipelines** 입니다.

(2) 카탈로그를 자동으로 새로 고치는 시간 간격입니다. 지원되는 시간 단위는 초(**s**), 분(**m**), 시간(**h**), 일(**d**) 및 주(**w**)입니다. 기본 간격은 30분입니다.

참고

TektonHub CR에서 선택적 필드에 사용자 정의 값을 제공하지 않으면 Tekton Hub API 구성 맵에 구성된 기본값이 사용됩니다.

2. ***TektonHub*** CR을 적용합니다.

```
$ oc apply -f <tekton-hub-cr>.yaml
```

3. 설치 상태를 확인합니다. ***TektonHub*** CR은 안정적인 상태를 유지하는 데 약간의 시간이 걸릴 수 있습니다.

```
$ oc get tektonhub.operator.tekton.dev
```

샘플 출력

```
NAME   VERSION   READY   REASON   APIURL                 UIURL
hub    v1.9.0    True             https://api.route.url/   https://ui.route.url/
```

1.1.2. 로그인 및 평가를 사용하여 Tekton Hub 설치

Tekton Hub 아티팩트에 대한 권한 부여 및 등급을 사용한 로그인을 지원하는 사용자 정의 구성으로 클러스터에 Tekton Hub를 설치할 수 있습니다.

사전 요구 사항

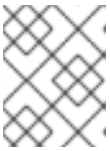
- Red Hat OpenShift Pipelines Operator가 클러스터의 기본 **openshift-pipelines** 네임스페이스에 설치되어 있는지 확인합니다.

프로세스

1. Git 리포지토리 호스팅 공급자를 사용하여 OAuth 애플리케이션을 생성하고 클라이언트 ID 및 클라이언트 시크릿을 기록해 둡니다. 지원되는 공급자는 GitHub, GitLab, BitBucket 입니다.
   - GitHub OAuth 애플리케이션 의 경우 Homepage URL 과 인증 콜백 URL 을 < **auth-route>로 설정합니다**.
   - GitLab OAuth 애플리케이션 의 경우 **REDIRECT_URI** 를 < **auth-route>/auth/gitlab/callback** 으로 설정합니다.
   - BitBucket OAuth 애플리케이션 의 경우 **콜백 URL** 을 < **auth-route>로 설정합니다**.
2. Tekton Hub API 시크릿을 포함하도록 < **tekton_hub_root>/config/02-api/20-api-secret.yaml** 파일을 편집합니다. 예를 들면 다음과 같습니다.

```
apiVersion: v1
kind: Secret
metadata:
  name: tekton-hub-api
  namespace: openshift-pipelines
type: Opaque
stringData:
  GH_CLIENT_ID: 1
  GH_CLIENT_SECRET: 2
  GL_CLIENT_ID: 3
  GL_CLIENT_SECRET: 4
  BB_CLIENT_ID: 5
  BB_CLIENT_SECRET: 6
  JWT_SIGNING_KEY: 7
  ACCESS_JWT_EXPIRES_IN: 8
  REFRESH_JWT_EXPIRES_IN: 9
  AUTH_BASE_URL: 10
  GHE_URL: 11
  GLE_URL: 12
```

1. *GitHub OAuth 애플리케이션의 클라이언트 ID 입니다.*
2. *GitHub OAuth 애플리케이션의 클라이언트 시크릿.*
3. *GitLab OAuth 애플리케이션의 클라이언트 ID 입니다.*
4. *GitLab OAuth 애플리케이션의 클라이언트 시크릿입니다.*
5. *BitBucket OAuth 애플리케이션의 클라이언트 ID 입니다.*
6. *BitBucket OAuth 애플리케이션의 클라이언트 시크릿입니다.*
7. *사용자를 위해 생성된 JSON 웹 토큰(JWT)에 서명하는 데 사용되는 긴 임의 문자열입니다.*
8. *액세스 토큰이 만료된 후 시간 제한을 추가합니다. 예를 들어 **1m** 에서는 m은 분을 나타냅니다. 지원되는 시간 단위는 초(**s**), 분(**m**), 시간(**h**), 일(**d**) 및 주(**w**) 입니다.*
9. *새로 고침 토큰이 만료된 후 시간 제한을 추가합니다. 예를 들어 **1m** 에서는 **m** 은 분을 나타냅니다. 지원되는 시간 단위는 초(**s**), 분(**m**), 시간(**h**), 일(**d**) 및 주(**w**) 입니다. 토큰 새로 고침에 설정된 만료 시간이 토큰 액세스에 설정된 만료 시간보다 큰지 확인합니다.*
10. *OAuth 애플리케이션의 경로 URL 입니다.*
11. *GitHub Enterprise URL(GitHub Enterprise을 사용하여 인증하는 경우). 이 필드의 값으로 카탈로그에 URL을 지정하지 마십시오.*
12. *GitLab Enterprise를 사용하여 인증하는 경우 GitLab Enterprise URL 입니다. 이 필드의 값으로 카탈로그에 URL을 지정하지 마십시오.*



참고

배포와 관련이 없는 Git 리포지토리 호스팅 서비스 공급자에 대해 사용되지 않는 필드를 삭제할 수 있습니다.

3. *다음 예와 유사한 **TektonHub** CR을 생성합니다.*

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonHub
metadata:
  name: hub
spec:
  targetNamespace: openshift-pipelines 1
  db: 2
    secret: tekton-hub-db 3

  categories: 4
    - Automation
    - Build Tools
    - CLI
    - Cloud
    - Code Quality
      ...

  catalogs: 5
```

```
  - name: tekton
    org: tektoncd
    type: community
    provider: github
    url: https://github.com/tektoncd/catalog
    revision: main

scopes: 6
  - name: agent:create
    users: [<username>]
  - name: catalog:refresh
    users: [<username>]
  - name: config:refresh
    users: [<username>]

default: 7
  scopes:
    - rating:read
    - rating:write

api:
  catalogRefreshInterval: 30m 8
```

1. Tekton Hub를 설치해야 하는 네임스페이스입니다. 기본값은 **openshift-pipelines** 입니다.
2. 선택 사항: Crunchy Postgres 데이터베이스와 같은 사용자 지정 데이터베이스.
3. 데이터베이스 시크릿의 이름은 **tekton-hub-db** 여야 합니다.
4. 선택사항: Tekton Hub의 작업 및 파이프라인에 대한 사용자 지정 카테고리입니다.
5. 선택사항: Tekton Hub에 대해 사용자 지정된 카탈로그입니다.
6. 선택 사항: 추가 사용자. **[<username_1>, <username_2>, <username_3>]** 과 같은 여러 사용자를 사용할 수 있습니다.
7. 선택사항: 사용자 지정된 기본 범위입니다.
8. 카탈로그를 자동으로 새로 고치는 시간 간격입니다. 지원되는 시간 단위는 초(**s**), 분(**m**), 시간(**h**), 일(**d**) 및 주(**w**)입니다. 기본 간격은 30분입니다.

참고

TektonHub CR에서 선택적 필드에 사용자 정의 값을 제공하지 않으면 Tekton Hub API 구성 맵에 구성된 기본값이 사용됩니다.

4. ***TektonHub*** CR을 적용합니다.

```
$ oc apply -f <tekton-hub-cr>.yaml
```

5. 설치 상태를 확인합니다. ***TektonHub*** CR은 안정적인 상태를 유지하는 데 약간의 시간이 걸릴 수 있습니다.

```
$ oc get tektonhub.operator.tekton.dev
```

샘플 출력

```
NAME   VERSION   READY   REASON   APIURL                 UIURL
hub    v1.9.0    True             https://api.route.url/   https://ui.route.url/
```

1.2. 선택 사항: TEKTON HUB 에서 사용자 정의 데이터베이스 사용

*클러스터 관리자는 Operator가 설치한 기본 PostgreSQL 데이터베이스 대신 Tekton Hub와 함께 사용자 지정 데이터베이스를 사용할 수 있습니다. 설치 시 사용자 지정 데이터베이스를 연결하여 Tekton Hub에서 제공하는 **db-migration**,**api** 및 **ui** 인터페이스와 함께 사용할 수 있습니다. 또는 설치가 default 데이터베이스와 완료된 후에도 사용자 지정 데이터베이스를 Tekton Hub와 연결할 수 있습니다.*

프로세스

1. *다음 키를 사용하여 대상 네임스페이스에 **tekton-hub-db** 라는 시크릿을 생성합니다.*
   - ***POSTGRES_HOST***
   - ***POSTGRES_DB***
   - ***POSTGRES_USER***
   - ***POSTGRES_PASSWORD***
   - ***POSTGRES_PORT***

   예: 사용자 정의 데이터베이스 시크릿

   ```
   apiVersion: v1
   kind: Secret
   metadata:
     name: tekton-hub-db
     labels:
       app: tekton-hub-db
   type: Opaque
   stringData:
     POSTGRES_HOST: <The name of the host of the database>
     POSTGRES_DB: <Name of the database>
     POSTGRES_USER: <username>
     POSTGRES_PASSWORD: <password>
     POSTGRES_PORT: <The port that the database is listening on>
   ...
   ```

   참고

   *기본 대상 네임스페이스는 **openshift-pipelines** 입니다.*

2. ***TektonHub** CR에서 데이터베이스 시크릿 속성 값을 **tekton-hub-db** 로 설정합니다.*

   예: 사용자 정의 데이터베이스 시크릿 추가

   ```
   apiVersion: operator.tekton.dev/v1alpha1
   kind: TektonHub
   ```

```
metadata:
  name: hub
spec:
  targetNamespace: openshift-pipelines
  db:
    secret: tekton-hub-db
  api:
    hubConfigUrl: https://raw.githubusercontent.com/tektoncd/hub/main/config.yaml
    catalogRefreshInterval: 30m
...
```

3. 업데이트된 **TektonHub** CR을 사용하여 사용자 지정 데이터베이스를 Tekton Hub와 연결합니다.

   a. 클러스터에 Tekton Hub를 설치할 때 사용자 지정 데이터베이스를 연결하는 경우 업데이트된 **TektonHub** CR을 적용합니다.

   ```
   $ oc apply -f <tekton-hub-cr>.yaml
   ```

   b. 또는 Tekton Hub 설치가 완료된 후 사용자 정의 데이터베이스를 연결하는 경우 기존 TektonHub CR을 업데이트된 **TektonHub** CR로 교체합니다.

   ```
   $ oc replace -f <tekton-hub-cr>.yaml
   ```

4. 설치 상태를 확인합니다. **TektonHub** CR은 안정적인 상태를 유지하는 데 약간의 시간이 걸릴 수 있습니다.

   ```
   $ oc get tektonhub.operator.tekton.dev
   ```

   샘플 출력

   ```
   NAME   VERSION   READY   REASON   APIURL                 UIURL
   hub    v1.9.0    True             https://api.route.url/   https://ui.route.url/
   ```

1.2.1. 선택 사항: Crunchy Postgres 데이터베이스 및 Tekton Hub 설치

클러스터 관리자는 Crunchy Postgres 데이터베이스를 설치하고 기본 데이터베이스 대신 사용할 Tekton Hub를 구성할 수 있습니다.

사전 요구 사항

- Operator Hub에서 Crunchy Postgres Operator를 설치합니다.
- Crunchy Postgres 데이터베이스를 시작하는 Postgres 인스턴스를 생성합니다.

프로세스

1. Crunchy Postgres Pod로 이동합니다.

   예: test-instance1-m7hh-0 Pod로 가져오기

   ```
   $ oc exec -it -n openshift-operators test-instance1-m7hh-0 -- /bin/sh

   Defaulting container name to database.
   ```

```
Use 'oc describe pod/test-instance1-m7hh-0 -n openshift-operators' to see all of the
containers in this pod.
sh-4.4$ psql -U postgres
psql (14.4)
Type "help" for help.
```

2. ***pg_hba.conf*** *파일을 찾습니다.*

```
postgres=# SHOW hba_file;
         hba_file
--------------------------
 /pgdata/pg14/pg_hba.conf
(1 row)

postgres=#
```

3. *데이터베이스를 종료합니다.*

4. ***pg_hba.conf*** *파일에 들어오는 모든 연결에 액세스하는 데 필요한* ***모든 0.0.0.0/0 md5*** *항목이 있는지 확인합니다. 또한* ***pg_hba.conf*** *파일의 끝에 항목을 추가합니다.*

   예: pg_hba.conf 파일

```
sh-4.4$ cat /pgdata/pg14/pg_hba.conf

# Do not edit this file manually!
# It will be overwritten by Patroni!
local all "postgres" peer
hostssl replication "_crunchyrepl" all cert
hostssl "postgres" "_crunchyrepl" all cert
host all "_crunchyrepl" all reject
hostssl all all all md5
host  all  all 0.0.0.0/0 md5
```

5. ***pg_hba.conf*** *파일을 저장하고 데이터베이스를 다시 로드합니다.*

```
sh-4.4$ psql -U postgres
psql (14.4)
Type "help" for help.

postgres=# SHOW hba_file;
         hba_file
--------------------------
 /pgdata/pg14/pg_hba.conf
(1 row)

postgres=# SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)
```

6. *데이터베이스를 종료합니다.*

7. *Crunchy Postgres 호스트의 시크릿 값을 디코딩합니다.*

예: Crunchy Postgres 호스트의 시크릿 값 삭제

```
$ echo 'aGlwcG8tcHJpbWFyeS5vcGVuc2hpZnQtb3BlcmF0b3JzLnN2YyA=' | base64 --
decode
test-primary.openshift-operators.svc
```

8. 다음 키를 사용하여 대상 네임스페이스에 **tekton-hub-db** 라는 시크릿을 생성합니다.

- ***POSTGRES_HOST***
- ***POSTGRES_DB***
- ***POSTGRES_USER***
- ***POSTGRES_PASSWORD***
- ***POSTGRES_PORT***

예: 사용자 정의 데이터베이스 시크릿

```
apiVersion: v1
kind: Secret
metadata:
  name: tekton-hub-db
  labels:
    app: tekton-hub-db
type: Opaque
stringData:
  POSTGRES_HOST: test-primary.openshift-operators.svc
  POSTGRES_DB: test
  POSTGRES_USER: <username>
  POSTGRES_PASSWORD: <password>
  POSTGRES_PORT: '5432'
...
```

참고

*기본 대상 네임스페이스는 **openshift-pipelines** 입니다.*

9. ***TektonHub*** *CR에서 데이터베이스 시크릿 속성 값을* ***tekton-hub-db*** *로 설정합니다.*

예: 사용자 정의 데이터베이스 시크릿 추가

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonHub
metadata:
  name: hub
spec:
  targetNamespace: openshift-pipelines
  db:
    secret: tekton-hub-db
...
```

10. 업데이트된 ***TektonHub*** CR을 사용하여 사용자 지정 데이터베이스를 Tekton Hub와 연결합니다.

```
$ oc apply -f <tekton-hub-cr>.yaml
```

11. 설치 상태를 확인합니다. **TektonHub** CR은 안정적인 상태를 유지하는 데 약간의 시간이 걸릴 수 있습니다.

```
$ oc get tektonhub.operator.tekton.dev
```

샘플 출력

```
NAME   VERSION   READY   REASON   APIURL                 UIURL
hub    v1.9.0    True             https://api.route.url/   https://ui.route.url/
```

1.2.2. 선택 사항: Tekton Hub 데이터를 기존 Crunchy Postgres 데이터베이스로 마이그레이션

Tekton Hub는 Crunchy Postgres를 사용자 지정 데이터베이스로 사용할 수 있습니다. 기본 데이터베이스가 있는 사전 설치된 Tekton Hub의 경우 클러스터 관리자는 Tekton Hub 데이터를 내부 또는 기본 데이터베이스에서 외부 Crunchy Postgres 데이터베이스로 마이그레이션한 후 사용자 지정 데이터베이스로 Crunchy Postgres를 사용할 수 있습니다.

프로세스

1. 내부 또는 기본 데이터베이스의 기존 데이터를 포드의 파일로 덤프합니다.

예: 데이터 덤프

```
$ pg_dump -Ft -h localhost -U postgres hub -f /tmp/hub.dump
```

2. 데이터 덤프가 포함된 파일을 로컬 시스템에 복사합니다.

명령 형식

```
$ oc cp -n <namespace> <podName>:<path-to-hub.dump> <path-to-local-system>
```

예

```
$ oc cp -n openshift-pipelines tekton-hub-db-7d6d888c67-p7mdr:/tmp/hub.dump
/home/test_user/Downloads/hub.dump
```

3. 로컬 시스템의 데이터 덤프가 포함된 파일을 외부 Crunchy Postgres 데이터베이스를 실행하는 포드로 복사합니다.

명령 형식

```
$ oc cp -n <namespace> <path-to-local-system> <podName>:<path-to-hub.dump>
```

예

```
$ oc cp -n openshift-operators /home/test_user/Downloads/hub.dump test-instance1-spnz-
0:/tmp/hub.dump
```

4. *Crunchy Postgres 데이터베이스에서 데이터를 복원합니다.*

   명령 형식

   ```
   $ pg_restore -d <database-name> -h localhost -U postgres <path-where-file-is-copied>
   ```

   예

   ```
   $ pg_restore -d test -h localhost -U postgres /tmp/hub.dump
   ```

5. *Crunchy Postgres Pod로 이동합니다. 예:* ***test-instance1-m7h-0*** *Pod로 이동합니다.*

   ```
   $ oc exec -it -n openshift-operators test-instance1-m7hh-0 -- /bin/sh

   Defaulting container name to database.
   Use 'oc describe pod/test-instance1-m7hh-0 -n openshift-operators' to see all of the
   containers in this pod.
   sh-4.4$ psql -U postgres
   psql (14.4)
   Type "help" for help.
   ```

6. ***pg_hba.conf*** *파일을 찾습니다.*

   ```
   postgres=# SHOW hba_file;
           hba_file
   --------------------------
    /pgdata/pg14/pg_hba.conf
   (1 row)

   postgres=#
   ```

7. *데이터베이스를 종료합니다.*

8. ***pg_hba.conf*** *파일에 모든 들어오는 연결에 액세스하는 데 필요한* ***모든 0.0.0.0/0 md5가 항목 호스트가*** *있는지 확인합니다. 필요한 경우* ***pg_hba.conf*** *파일의 끝에 항목을 추가합니다.*

   예: pg_hba.conf 파일

   ```
   sh-4.4$ cat /pgdata/pg14/pg_hba.conf

   # Do not edit this file manually!
   # It will be overwritten by Patroni!
   local all "postgres" peer
   hostssl replication "_crunchyrepl" all cert
   hostssl "postgres" "_crunchyrepl" all cert
   host all "_crunchyrepl" all reject
   hostssl all all all md5
   host  all  all 0.0.0.0/0 md5
   ```

9. ***pg_hba.conf*** *파일을 저장하고 데이터베이스를 다시 로드합니다.*

   ```
   sh-4.4$ psql -U postgres
   psql (14.4)
   ```

```
Type "help" for help.

postgres=# SHOW hba_file;
         hba_file
--------------------------
 /pgdata/pg14/pg_hba.conf
(1 row)

postgres=# SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)
```

10. *데이터베이스를 종료합니다.*

11. *대상 네임스페이스의* ***tekton-hub-db*** *라는 보안에 다음 키가 있는지 확인합니다.*

    - ***POSTGRES_HOST***
    - ***POSTGRES_DB***
    - ***POSTGRES_USER***
    - ***POSTGRES_PASSWORD***
    - ***POSTGRES_PORT***

    예: 사용자 정의 데이터베이스 시크릿

    ```
    apiVersion: v1
    kind: Secret
    metadata:
      name: tekton-hub-db
      labels:
        app: tekton-hub-db
    type: Opaque
    stringData:
      POSTGRES_HOST: test-primary.openshift-operators.svc
      POSTGRES_DB: test
      POSTGRES_USER: test
      POSTGRES_PASSWORD: woXOisU5>ocJiTF7y{{;1[Q(
      POSTGRES_PORT: '5432'
    ...
    ```

참고

POSTGRES_HOST *필드의 값은 시크릿으로 인코딩됩니다. 다음 예제를 사용하여 Crunchy Postgres 호스트의 값을 디코딩할 수 있습니다.*

예: Crunchy Postgres 호스트의 시크릿 값 삭제

```
$ echo
'aGlwcG8tcHJpbWFyeS5vcGVuc2hpZnQtb3BlcmF0b3JzLnN2YyA=' |
base64 --decode
test-primary.openshift-operators.svc
```

12. ***TektonHub*** *CR에서 데이터베이스 시크릿 속성 값이* ***tekton-hub-db*** *인지 확인합니다.*

예: 데이터베이스 시크릿 이름이 있는 TektonHub CR

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonHub
metadata:
  name: hub
spec:
  targetNamespace: openshift-pipelines
  db:
    secret: tekton-hub-db
...
```

13. *외부 Crunchy Postgres 데이터베이스를 Tekton Hub와 연결하려면 기존* ***TektonHub*** *CR을 업데이트된* ***TektonHub*** *CR로 교체합니다.*

```
$ oc replace -f <updated-tekton-hub-cr>.yaml
```

14. *Tekton Hub의 상태를 확인합니다. 업데이트된* ***TektonHub*** *CR은 안정적인 상태를 유지하는 데 약간의 시간이 걸릴 수 있습니다.*

```
$ oc get tektonhub.operator.tekton.dev
```

샘플 출력

```
NAME   VERSION   READY   REASON   APIURL                 UIURL
hub    v1.9.0    True             https://api.route.url/   https://ui.route.url/
```

1.3. 사용자 정의 카테고리 및 카탈로그를 사용하여 TEKTON HUB 업데이트

클러스터 관리자는 조직의 컨텍스트를 반영하는 사용자 정의 카테고리, 카탈로그, 범위 및 기본 범위로 Tekton Hub를 업데이트할 수 있습니다.

프로세스

1. *선택 사항: Tekton Hub CR의* ***카테고리*** *,* ***카탈로그****, 범위,* ***default:scopes*** *필드를 편집합니다.*

참고

*카테고리, 카탈로그, 범위 및 기본 범위에 대한 기본 정보는 Tekton Hub API 구성 맵에서 가져옵니다. **TektonHub** CR에서 사용자 지정 값을 제공하는 경우 기본값을 덮어씁니다.*

2. *Tekton Hub CR을 적용합니다.*

```
$ oc apply -f <tekton-hub-cr>.yaml
```

3. *Tekton Hub 상태를 확인합니다.*

```
$ oc get tektonhub.operator.tekton.dev
```

샘플 출력

```
NAME   VERSION   READY   REASON   APIURL              UIURL
hub    v1.9.0    True             https://api.route.url   https://ui.route.url
```

1.4. TEKTON HUB의 카탈로그 새로 고침 간격 수정

*Tekton Hub의 기본 카탈로그 새로 고침 간격은 30분입니다. 클러스터 관리자는 **TektonHub** CR의 **catalogRefreshInterval** 필드 값을 수정하여 자동 카탈로그 새로 고침 간격을 수정할 수 있습니다.*

프로세스

1. ***TektonHub** CR의 **catalogRefreshInterval** 필드 값을 수정합니다.*

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonHub
metadata:
  name: hub
spec:
  targetNamespace: openshift-pipelines (1)
  api:
    catalogRefreshInterval: 30m (2)
```

(1) *Tekton Hub가 설치된 네임스페이스입니다. 기본값은 **openshift-pipelines** 입니다.*

(2) *카탈로그를 자동으로 새로 고치는 시간 간격입니다. 지원되는 시간 단위는 초(**s**), 분(**m**), 시간(**h**), 일(**d**) 및 주(**w**)입니다. 기본 간격은 30분입니다.*

2. ***TektonHub** CR을 적용합니다.*

```
$ oc apply -f <tekton-hub-cr>.yaml
```

3. *설치 상태를 확인합니다. **TektonHub** CR은 안정적인 상태를 유지하는 데 약간의 시간이 걸릴 수 있습니다.*

```
$ oc get tektonhub.operator.tekton.dev
```

샘플 출력

```
NAME   VERSION   READY   REASON   APIURL                 UIURL
hub    v1.9.0    True             https://api.route.url/   https://ui.route.url/
```

1.5. TEKTON HUB 구성에 새 사용자 추가

클러스터 관리자는 범위가 다른 Tekton Hub에 새 사용자를 추가할 수 있습니다.

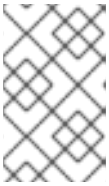
프로세스

1. ***TektonHub*** *CR을 수정하여 다양한 범위가 있는 새 사용자를 추가합니다.*

```
...
scopes:
  - name: agent:create
    users: [<username_1>, <username_2>] (1)
  - name: catalog:refresh
    users: [<username_3>, <username_4>]
  - name: config:refresh
    users: [<username_5>, <username_6>]

default:
  scopes:
    - rating:read
    - rating:write
...
```

(1) *Git 리포지토리 호스팅 서비스 공급자에 등록된 사용자 이름입니다.*



참고

Tekton Hub에 처음 로그인하는 새 사용자는 기본 범위만 갖습니다. 추가 범위를 활성화하려면 ***TektonHub*** *CR의* ***scopes*** *필드에 사용자의 사용자 이름이 추가되었는지 확인합니다.*

2. *업데이트된* ***TektonHub*** *CR을 적용합니다.*

```
$ oc apply -f <tekton-hub-cr>.yaml
```

3. *Tekton Hub의 상태를 확인합니다. 업데이트된* ***TektonHub*** *CR은 안정적인 상태를 유지하는 데 약간의 시간이 걸릴 수 있습니다.*

```
$ oc get tektonhub.operator.tekton.dev
```

샘플 출력

```
NAME   VERSION   READY   REASON   APIURL                 UIURL
hub    v1.9.0    True             https://api.route.url/   https://ui.route.url/
```

4. *구성을 새로 고칩니다.*

```
$ curl -X POST -H "Authorization: <access-token>" \ 1
  --header "Content-Type: application/json" \
  --data '{"force": true} \
  <api-route>/system/config/refresh
```

1 JWT 토큰입니다.

1.6. RED HAT OPENSHIFT PIPELINES OPERATOR를 1.7에서 1.8로 업그레이드한 후 TEKTON HUB 권한 부여 비활성화

Red Hat OpenShift Pipelines Operator 1.8을 사용하여 Tekton Hub를 설치하면 기본 설치에 대해 Tekton Hub 아티팩트의 로그인 권한 부여 및 등급이 비활성화됩니다. 그러나 Operator를 1.7에서 1.8로 업그레이드하면 클러스터의 Tekton Hub 인스턴스가 로그인 권한 부여 및 평가를 자동으로 비활성화하지 않습니다.

Operator를 1.7에서 1.8로 업그레이드한 후 Tekton Hub의 로그인 권한 부여 및 평가를 비활성화하려면 다음 절차의 단계를 수행합니다.

사전 요구 사항

- Red Hat OpenShift Pipelines Operator가 클러스터의 기본 **openshift-pipelines** 네임스페이스에 설치되어 있는지 확인합니다.

프로세스

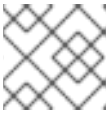
1. Operator 1.7에 Tekton Hub를 수동으로 설치하는 동안 생성한 기존 Tekton Hub API 시크릿을 삭제합니다.

```
$ oc delete secret tekton-hub-api -n <targetNamespace> 1
```

1 Tekton Hub API 시크릿 및 Tekton Hub CR의 공통 네임스페이스입니다. 기본적으로 대상 네임스페이스는 **openshift-pipelines** 입니다.

2. Tekton Hub API의 **TektonInstallerSet** 오브젝트를 삭제합니다.

```
$ oc get tektoninstallerset -o name | grep tekton-hub-api | xargs oc delete
```



참고

삭제 후 Operator는 새 Tekton Hub API 설치 프로그램을 자동으로 생성합니다.

기다린 후 Tekton Hub의 상태를 확인합니다. **READY** 열에 **True** 가 표시되면 다음 단계로 이동합니다.

```
$ oc get tektonhub hub
```

샘플 출력

```
NAME   VERSION      READY   REASON   APIURL
UIURL
```

```
hub    1.8.0        True        https://tekton-hub-api-openshift-pipelines.apps.example.com
https://tekton-hub-ui-openshift-pipelines.apps.example.com
```

3. Tekton Hub UI의 **ConfigMap** 오브젝트를 삭제합니다.

```
$ oc delete configmap tekton-hub-ui -n <targetNamespace>
```
(1)

(1) Tekton Hub UI 및 Tekton Hub CR의 공통 네임스페이스입니다. 기본적으로 대상 네임스페이스는 **openshift-pipelines** 입니다.

4. Tekton Hub UI의 **TektonInstallerSet** 오브젝트를 삭제합니다.

```
$ oc get tektoninstallerset -o name | grep tekton-hub-ui | xargs oc delete
```

참고

삭제 후 Operator는 새 Tekton Hub UI 설치 프로그램을 자동으로 생성합니다.

기다린 후 Tekton Hub의 상태를 확인합니다. **READY** 열에 **True** 가 표시되면 다음 단계로 이동합니다.

```
$ oc get tektonhub hub
```

샘플 출력

```
NAME   VERSION      READY   REASON   APIURL
UIURL
hub    1.8.0        True        https://tekton-hub-api-openshift-pipelines.apps.example.com
https://tekton-hub-ui-openshift-pipelines.apps.example.com
```

1.7. 추가 리소스

- Tekton Hub의 GitHub 리포지토리
- OpenShift Pipelines 설치
- Red Hat OpenShift Pipelines 릴리스 정보