



Red Hat OpenShift Pipelines 1.15

OpenShift Pipelines의 가시성

OpenShift Pipelines의 관찰 기능

Red Hat OpenShift Pipelines 1.15 OpenShift Pipelines의 가시성

OpenShift Pipelines의 관찰 기능

법적 공지

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

초록

이 문서에서는 OpenShift Pipelines의 가시성 기능에 대한 정보를 제공합니다.

차례

1장. OPENSIFT PIPELINES 관찰 기능에 TEKTON 결과 사용	3
1.1. TEKTON RESULTS 개념	3
1.2. TEKTON 결과 설치 준비	6
1.3. TEKTON 결과 설치	9
1.4. OPC 명령줄 유틸리티를 사용하여 TEKTON 결과 쿼리	10
1.5. 추가 리소스	17
2장. OPENSIFT LOGGING OPERATOR를 사용하여 파이프라인 로그 보기	18
2.1. 사전 요구 사항	18
2.2. KIBANA에서 파이프라인 로그 보기	18
2.3. 추가 리소스	20

1장. OPENSIFT PIPELINES 관찰 기능에 TEKTON 결과 사용

Tekton Results는 모든 파이프라인 실행 및 작업 실행에 대한 전체 정보를 보관하는 서비스입니다. 필요에 따라 **PipelineRun** 및 **TaskRun** 리소스를 정리하고 Tekton Results API 또는 **opc** 명령줄 유틸리티를 사용하여 YAML 매니페스트 및 로깅 정보에 액세스할 수 있습니다.



중요

Tekton 결과는 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 범위](#)를 참조하십시오.

1.1. TEKTON RESULTS 개념

Tekton Results 아카이브는 파이프라인 실행 및 작업 실행 결과 및 레코드 형태로 실행됩니다.

실행을 완료하는 모든 **PipelineRun** 및 **TaskRun** CR(사용자 정의 리소스)에 대해 Tekton Results는 *레코드*를 생성합니다.

*결과*는 하나 또는 여러 레코드를 포함할 수 있습니다. 레코드는 항상 정확히 하나의 결과의 일부입니다.

결과적으로 파이프라인 실행에 해당하며 **PipelineRun** CR 자체 및 파이프라인 실행의 일부로 시작된 모든 **TaskRun** CR에 대한 레코드가 포함됩니다.

파이프라인 실행을 사용하지 않고 작업 실행이 직접 시작된 경우 이 작업 실행에 대한 결과가 생성됩니다. 이 결과에는 동일한 작업 실행에 대한 레코드가 포함됩니다.

각 결과에는 **PipelineRun** 또는 **TaskRun** CR이 생성된 네임스페이스와 CR의 UUID가 포함된 이름이 있습니다. 결과 이름의 형식은 `<namespace_name>/results/<parent_run_uuid>`입니다. 이 형식에서 `<parent_run_uuid>`는 파이프라인 실행 또는 직접 시작된 작업 실행의 UUID 형식입니다.

결과 이름 예

```
results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed
```

각 레코드에는 레코드가 포함된 결과의 이름과 레코드가 해당하는 **PipelineRun** 또는 **TaskRun** CR의 UUID가 포함됩니다. 결과 이름의 형식은 `<namespace_name>/results/<parent_run_uuid>/results/<run_uuid>`입니다.

레코드 이름 예

```
results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed/records/e9c736db-5665-441f-922f-7c1d65c9d621
```

이 레코드에는 실행이 완료된 후 존재했던 **TaskRun** 또는 **PipelineRun** CR의 전체 YAML 매니페스트가 포함됩니다. 이 매니페스트에는 실행 사양, 실행에 지정된 주석, 실행 결과에 대한 특정 정보(예: 완료 시간 및 실행이 성공했는지 여부)가 포함되어 있습니다.

TaskRun 또는 **PipelineRun** CR이 있는 동안 다음 명령을 사용하여 YAML 매니페스트를 볼 수 있습니다.

```
$ oc get pipelinerun <cr_name> -o yaml
```

Tekton Results는 **TaskRun** 또는 **PipelineRun** CR이 삭제된 후 이 매니페스트를 유지하고 있으며 이를 보고 검색할 수 있도록 합니다.

완료 후 실행되는 파이프라인의 YAML 매니페스트 예

```
kind: PipelineRun
spec:
  params:
    - name: message
      value: five
  timeouts:
    pipeline: 1h0m0s
  pipelineRef:
    name: echo-pipeline
  taskRunTemplate:
    serviceAccountName: pipeline
status:
  startTime: "2023-08-07T11:41:40Z"
  conditions:
    - type: Succeeded
      reason: Succeeded
      status: "True"
      message: 'Tasks Completed: 1 (Failed: 0, Cancelled 0), Skipped: 0'
      lastTransitionTime: "2023-08-07T11:41:49Z"
  pipelineSpec:
    tasks:
      - name: echo-task
        params:
          - name: message
            value: five
        taskRef:
          kind: Task
          name: echo-task-pipeline
        params:
          - name: message
            type: string
    completionTime: "2023-08-07T11:41:49Z"
  childReferences:
    - kind: TaskRun
      name: echo-pipeline-run-gmzrx-echo-task
      apiVersion: tekton.dev/v1
      pipelineTaskName: echo-task
metadata:
  uid: 62c3b02e-f12b-416c-9771-c02af518f6d4
  name: echo-pipeline-run-gmzrx
  labels:
    tekton.dev/pipeline: echo-pipeline
  namespace: releasetest-js5tt
  finalizers:
    - chains.tekton.dev/pipelinerun
  generation: 2
  annotations:
    results.tekton.dev/log: releasetest-js5tt/results/62c3b02e-f12b-416c-9771-
```



```

c02af518f6d4/logs/c1e49dd8-d641-383e-b708-e3a02b6a4378
  chains.tekton.dev/signed: "true"
  results.tekton.dev/record: releasetest-js5tt/results/62c3b02e-f12b-416c-9771-
c02af518f6d4/records/62c3b02e-f12b-416c-9771-c02af518f6d4
  results.tekton.dev/result: releasetest-js5tt/results/62c3b02e-f12b-416c-9771-c02af518f6d4
generateName: echo-pipeline-run-
managedFields:
- time: "2023-08-07T11:41:39Z"
  manager: kubectl-create
  fieldsV1:
    f:spec:
      .: {}
      f:params: {}
      f:pipelineRef:
        .: {}
        f:name: {}
    f:metadata:
      f:generateName: {}
  operation: Update
  apiVersion: tekton.dev/v1
  fieldsType: FieldsV1
- time: "2023-08-07T11:41:40Z"
  manager: openshift-pipelines-controller
  fieldsV1:
    f:metadata:
      f:labels:
        .: {}
        f:tekton.dev/pipeline: {}
  operation: Update
  apiVersion: tekton.dev/v1
  fieldsType: FieldsV1
- time: "2023-08-07T11:41:49Z"
  manager: openshift-pipelines-chains-controller
  fieldsV1:
    f:metadata:
      f:finalizers:
        .: {}
        v:"chains.tekton.dev/pipelinerun": {}
      f:annotations:
        .: {}
        f:chains.tekton.dev/signed: {}
  operation: Update
  apiVersion: tekton.dev/v1
  fieldsType: FieldsV1
- time: "2023-08-07T11:41:49Z"
  manager: openshift-pipelines-controller
  fieldsV1:
    f:status:
      f:startTime: {}
      f:conditions: {}
      f:pipelineSpec:
        .: {}
        f:tasks: {}
        f:params: {}
      f:completionTime: {}
      f:childReferences: {}

```

```

operation: Update
apiVersion: tekton.dev/v1
fieldsType: FieldsV1
subresource: status
- time: "2023-08-07T11:42:15Z"
manager: openshift-pipelines-results-watcher
fieldsV1:
  f:metadata:
    f:annotations:
      f:results.tekton.dev/log: {}
      f:results.tekton.dev/record: {}
      f:results.tekton.dev/result: {}
operation: Update
apiVersion: tekton.dev/v1
fieldsType: FieldsV1
resourceVersion: "126429"
creationTimestamp: "2023-08-07T11:41:39Z"
deletionTimestamp: "2023-08-07T11:42:23Z"
deletionGracePeriodSeconds: 0
apiVersion: tekton.dev/v1

```

Tekton Results는 파이프라인 실행 또는 작업 실행의 일부로 실행된 모든 툴의 로깅 정보가 포함된 로그 레코드도 생성합니다.

모든 결과 및 레코드에 해당 이름으로 액세스할 수 있습니다. CEL(Common Expression Language) 쿼리를 사용하여 YAML 매니페스트를 포함하여 포함된 정보로 결과 및 레코드를 검색할 수도 있습니다.

1.2. TEKTON 결과 설치 준비

Tekton 결과를 설치하기 전에 여러 준비 단계를 완료해야 합니다.

1.2.1. SSL 인증서를 사용하여 보안 준비

Tekton Results는 SSL 인증서가 필요한 HTTPS 프로토콜을 사용하여 REST API를 제공합니다. 이 인증서와 함께 보안을 제공합니다. CA(인증 기관)에서 제공하는 기존 인증서가 있는 경우 이 인증서를 사용하여 자체 서명된 인증서를 생성합니다.

사전 요구 사항

- **openssl** 명령줄 유틸리티가 설치되어 있습니다.

프로세스

1. CA에서 제공한 인증서가 없는 경우 다음 명령을 입력하여 자체 서명 인증서를 생성합니다.

```

$ openssl req -x509 \
  -newkey rsa:4096 \
  -keyout key.pem \
  -out cert.pem \
  -days 365 \
  -nodes \
  -subj "/CN=tekton-results-api-service.openshift-pipelines.svc.cluster.local" \
  -addext "subjectAltName = DNS:tekton-results-api-service.openshift-
pipelines.svc.cluster.local"

```

`tekton-results-api-service.openshift-pipelines.svc.cluster.local` 을 Tekton Results API에 사용할 경로 끝점으로 교체합니다.

2. 다음 명령을 입력하여 인증서에서 TLS(전송 보안 계층) 보안을 생성합니다.

```
$ oc create secret tls -n openshift-pipelines tekton-results-tls --cert=cert.pem --key=key.pem
```

CA에서 제공하는 기존 인증서를 사용하려면 **cert.pem** 을 이 인증서가 포함된 파일 이름으로 교체합니다.

1.2.2. 데이터베이스 인증 정보를 사용하여 시크릿 준비

Tekton 결과는 PostgreSQL 데이터베이스를 사용하여 데이터를 저장합니다. Tekton Results와 함께 자동으로 설치된 PostgreSQL 서버 또는 배포에 이미 존재하는 외부 PostgreSQL 서버를 사용하도록 설치를 구성할 수 있습니다. 두 경우 모두 데이터베이스 인증 정보가 있는 시크릿을 제공합니다.

프로세스

다음 단계 중 하나를 완료합니다.

- 외부 PostgreSQL 서버를 사용할 필요가 없는 경우 다음 명령을 입력하여 **result** 라는 데이터베이스 사용자로 시크릿을 생성하고 **openshift-pipelines** 네임스페이스에 임의의 암호를 생성합니다.

```
$ oc create secret generic tekton-results-postgres \
  --namespace=openshift-pipelines \
  --from-literal=POSTGRES_USER=result \
  --from-literal=POSTGRES_PASSWORD=$(openssl rand -base64 20)
```



참고

이 명령과 후속 섹션에서 OpenShift Pipelines의 사용자 정의 대상 네임스페이스를 구성한 경우 **openshift-pipelines** 대신 이 네임스페이스의 이름을 사용합니다.

- 외부 PostgreSQL 데이터베이스 서버를 사용하여 Tekton Results 데이터를 저장하려면 다음 명령을 입력하여 이 서버의 인증 정보가 있는 시크릿을 생성합니다.

```
$ oc create secret generic tekton-results-postgres \
  --namespace=openshift-pipelines \
  --from-literal=POSTGRES_USER=<user> \ 1
  --from-literal=POSTGRES_PASSWORD=<password> 2
```

<user>를 Tekton 결과에서 사용해야 하는 PostgreSQL 사용자의 사용자 이름으로 바꿉니다. <password>를 동일한 계정의 암호로 바꿉니다.

1.2.3. 로깅 정보를 위한 스토리지 준비

Tekton Results는 파이프라인 실행 및 작업 실행과 관련된 정보를 로깅하기 위해 별도의 스토리지를 사용합니다. 다음 스토리지 유형 중 하나를 구성할 수 있습니다.

- Red Hat OpenShift Pipelines 클러스터의 PVC(영구 볼륨 클레임)
- Google Cloud Storage
- S3 버킷 스토리지

프로세스

다음 절차 중 하나를 완료합니다.

- PVC를 사용하려면 다음 단계를 완료합니다.
 - a. PVC에 대해 다음 정의를 사용하여 **pvc.yaml** 이라는 파일을 생성합니다.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: tekton-logs
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

- b. 다음 명령을 입력하여 정의를 적용합니다.

```
$ oc apply -n openshift-pipelines -f pvc.yaml
```

- Google Cloud Storage를 사용하려면 다음 단계를 완료합니다.
 - a. **gcloud** 명령을 사용하여 애플리케이션 자격 증명 파일을 생성합니다. 파일에 애플리케이션 자격 증명을 제공하는 방법에 대한 자세한 내용은 [Google Cloud 설명서](#)의 **gcloud CLI를 사용하여 제공되는 사용자 자격 증명**을 참조하십시오.
 - b. 다음 명령을 입력하여 애플리케이션 인증 정보 파일에서 시크릿을 생성합니다.

```
$ oc create secret generic gcs-credentials \
  --from-file=$HOME/.config/gcloud/application_default_credentials.json \
  -n openshift-pipelines
```

필요에 따라 애플리케이션 자격 증명 파일의 경로와 파일 이름을 조정합니다.

- S3 버킷 스토리지를 사용하려면 다음 단계를 완료합니다.
 - a. 다음 콘텐츠를 사용하여 **s3_secret.yaml** 이라는 파일을 생성합니다.

```
apiVersion: v1
kind: Secret
metadata:
  name: my_custom_secret
  namespace: tekton-pipelines
type: Opaque
stringData:
  S3_BUCKET_NAME: bucket1 ①
  S3_ENDPOINT: https://example.localhost.com ②
  S3_HOSTNAME_IMMUTABLE: "false"
  S3_REGION: region-1 ③
  S3_ACCESS_KEY_ID: "1234" ④
  S3_SECRET_ACCESS_KEY: secret_key ⑤
  S3_MULTI_PART_SIZE: "5242880"
```

- 1 1 S3 스토리지 버킷의 이름
- 2 2 S3 API 엔드 포인트 URL
- 3 S3 리전
- 4 S3 액세스 키 ID
- 5 S3 시크릿 액세스 키

b. 다음 명령을 입력하여 파일에서 시크릿을 생성합니다.

```
$ oc create secret generic s3-credentials \
--from-file=s3_secret.yaml -n openshift-pipelines
```

1.3. TEKTON 결과 설치

Tekton Results를 설치하려면 필요한 리소스를 제공한 다음 **TektonResult** CR(사용자 정의 리소스)을 생성하고 적용해야 합니다. OpenShift Pipelines Operator는 **TektonResult** 사용자 정의 리소스를 적용할 때 Results 서비스를 설치합니다.

사전 요구 사항

- Operator를 사용하여 OpenShift Pipelines를 설치했습니다.
- SSL 인증서를 사용하여 보안을 준비합니다.
- 로깅 정보를 위한 스토리지를 준비합니다.
- 데이터베이스 인증 정보를 사용하여 시크릿을 준비합니다.

프로세스

1. 다음 예제를 기반으로 **result.yaml**이라는 리소스 정의 파일을 생성합니다. 필요에 따라 설정을 조정할 수 있습니다.

```
apiVersion: operator.tekton.dev/v1alpha1
kind: TektonResult
metadata:
  name: result
spec:
  targetNamespace: openshift-pipelines
  logs_api: true
  log_level: debug
  db_port: 5432
  db_host: tekton-results-postgres-service.openshift-pipelines.svc.cluster.local
  logs_path: /logs
  logs_type: File
  logs_buffer_size: 32768
  auth_disable: true
  tls_hostname_override: tekton-results-api-service.openshift-pipelines.svc.cluster.local
  db_enable_auto_migration: true
  server_port: 8080
  prometheus_port: 9090
```

2. 이 파일에 로깅 정보를 위한 스토리지 구성을 추가합니다.

- PVC(영구 볼륨 클레임)를 구성한 경우 다음 행을 추가하여 PVC 이름을 제공합니다.

```
logging_pvc_name: tekton-logs
```

- Google Cloud Storage를 구성한 경우 다음 행을 추가하여 시크릿 이름, 인증 정보 파일 이름, Google Cloud Storage 버킷의 이름을 제공합니다.

```
gcs_creds_secret_name: gcs-credentials
gcs_creds_secret_key: application_default_credentials.json ①
gcs_bucket_name: bucket-name ②
```

- ① 시크릿을 생성할 때 사용한 애플리케이션 인증 정보 파일의 경로 없이 이름을 제공합니다.
- ② Google Cloud Storage에 버킷 이름을 입력합니다. Tekton 체인은 이 버킷을 사용하여 파이프라인 실행 및 작업 실행에 대한 로깅 정보를 저장합니다.

- S3 버킷 스토리지를 구성한 경우 다음 행을 추가하여 S3 시크릿 이름을 제공합니다.

```
secret_name: s3-credentials
```

3. 선택 사항: 외부 PostgreSQL 데이터베이스 서버를 사용하여 Tekton Results 정보를 저장하려면 다음 행을 파일에 추가합니다.

```
db_host: postgres.internal.example.com ①
db_port: 5432 ②
is_external_db: true
```

- ① PostgreSQL 서버의 호스트 이름입니다.
- ② PostgreSQL 서버의 포트입니다.

4. 다음 명령을 입력하여 리소스 정의를 적용합니다.

```
$ oc apply -n openshift-pipelines -f result.yaml
```

5. 다음 명령을 입력하여 Tekton Results 서비스 API의 경로를 노출합니다.

```
$ oc create route -n openshift-pipelines \
  passthrough tekton-results-api-service \
  --service=tekton-results-api-service --port=8080
```

1.4. OPC 명령줄 유틸리티를 사용하여 TEKTON 결과 쿼리

opc 명령줄 유틸리티를 사용하여 결과 및 레코드에 대한 Tekton 결과를 쿼리할 수 있습니다. **opc** 명령줄 유틸리티를 설치하려면 **tkn** 명령줄 유틸리티에 대한 패키지를 설치합니다. 이 패키지 설치에 대한 자세한 내용은 [tkn 설치](#)를 참조하십시오.

레코드 이름과 결과를 사용하여 해당 레코드에서 데이터를 검색할 수 있습니다.

CEL(Common Expression Language) 쿼리를 사용하여 결과 및 레코드를 검색할 수 있습니다. 이러한 검색에는 결과 또는 레코드의 UUID가 표시됩니다. 제공된 예제를 사용하여 공통 검색 유형에 대한 쿼리를 생성할 수 있습니다. 참조 정보를 사용하여 다른 쿼리를 생성할 수도 있습니다.

1.4.1. Tekton 결과 쿼리를 위한 **opc** 유틸리티 환경 준비

Tekton Results를 쿼리하려면 먼저 **opc** 유틸리티에 대한 환경을 준비해야 합니다.

사전 요구 사항

- Tekton 결과가 설치되어 있어야 합니다.
- **opc** 유틸리티를 설치했습니다.

프로세스

1. 다음 명령을 입력하여 **RESULTS_API** 환경 변수를 Tekton Results API의 경로로 설정합니다.

```
$ export RESULTS_API=$(oc get route tekton-results-api-service -n openshift-pipelines --no-headers -o custom-columns=":spec.host"):443
```

2. 다음 명령을 입력하여 Tekton Results API에 대한 인증 토큰을 생성합니다.

```
$ oc create token <service_account>
```

이 명령이 출력하는 문자열을 저장합니다.

3. 선택 사항: Tekton Results API를 사용하여 자동 인증을 위해 **~/config/tnk/results.yaml** 파일을 생성합니다. 파일에는 다음 내용이 포함되어야 합니다.

```
address: <tekton_results_route> 1
token: <authentication_token> 2
ssl:
  roots_file_path: /home/example/cert.pem 3
  server_name_override: tekton-results-api-service.openshift-pipelines.svc.cluster.local 4
service_account:
  namespace: service_acc_1 5
  name: service_acc_1 6
```

- 1 Tekton Results API의 경로입니다. **RESULTS_API**에 설정한 것과 동일한 값을 사용합니다.
- 2 **oc create token** 명령으로 생성된 인증 토큰입니다. 이 토큰을 제공하는 경우 **service_account** 설정을 재정의하고 **opc**는 이 토큰을 사용하여 인증합니다.
- 3 API 엔드포인트에 대해 구성된 SSL 인증서가 있는 파일의 위치입니다.
- 4 OpenShift Pipelines에 대한 사용자 정의 대상 네임스페이스를 구성한 경우 **openshift-pipelines**를 이 네임스페이스의 이름으로 교체합니다.
- 5 6 Tekton Results API를 사용하여 인증하는 서비스 계정의 이름입니다. 인증 토큰을 제공한 경우 **service_account** 매개변수를 제공할 필요가 없습니다.

또는 `~/.config/tnk/results.yaml` 파일을 생성하지 않으면 `--authtoken` 옵션을 사용하여 각 `opc` 명령에 토큰을 전달할 수 있습니다.

1.4.2. 이름으로 결과 및 레코드 쿼리

해당 이름을 사용하여 결과 및 레코드를 나열하고 쿼리할 수 있습니다.

사전 요구 사항

- Tekton 결과가 설치되어 있어야 합니다.
- `opc` 유틸리티를 설치하고 Tekton 결과를 쿼리할 수 있는 환경을 준비합니다.
- `jq` CLI를 설치하셨습니다.

프로세스

1. 네임스페이스에서 생성된 파이프라인 실행 및 작업 실행에 해당하는 모든 결과의 이름을 나열합니다. 다음 명령을 실행합니다.

```
$ opc results list --addr ${RESULTS_API} <namespace_name>
```

명령 예

```
$ opc results list --addr ${RESULTS_API} results-testing
```

출력 예

Name	Start	Update
results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed	2023-06-29 02:49:53 +0530 IST	2023-06-29 02:50:05 +0530 IST
results-testing/results/ad7eb937-90cc-4510-8380-defe51ad793f	2023-06-29 02:49:38 +0530 IST	2023-06-29 02:50:06 +0530 IST
results-testing/results/d064ce6e-d851-4b4e-8db4-7605a23671e4	2023-06-29 02:49:45 +0530 IST	2023-06-29 02:49:56 +0530 IST

2. 다음 명령을 입력하여 결과에서 모든 레코드 이름을 나열합니다.

```
$ opc results records list --addr ${RESULTS_API} <result_name>
```

명령 예

```
$ opc results records list --addr ${RESULTS_API} results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed
```

출력 예

Name	Start	Update	Type
results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed/records/e9c736db-5665-441f-922f-7c1d65c9d621	2023-06-29 02:49:57 +0530 IST	2023-06-29 02:49:53 +0530 IST	tekton.dev/v1.TaskRun
results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed/records/5de23a76-a12b-			


```
3a72-8a6a-4f15a3110a3e results.tekton.dev/v1alpha2.Log      2023-06-29 02:49:57 +0530
IST      2023-06-29 02:49:57 +0530 IST
results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed/records/57ce92f9-9bf8-3a0a-
aefb-dc20c3e2862d results.tekton.dev/v1alpha2.Log      2023-06-29 02:50:05 +0530 IST
2023-06-29 02:50:05 +0530 IST
results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed/records/e9a0c21a-f826-42ab-
a9d7-a03bcefed4fd tekton.dev/v1.TaskRun      2023-06-29 02:49:57 +0530 IST
2023-06-29 02:50:05 +0530 IST
results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed/records/04e2fbf2-8653-405f-
bc42-a262bcf02bed tekton.dev/v1.PipelineRun      2023-06-29 02:49:53 +0530 IST
2023-06-29 02:50:05 +0530 IST
results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed/records/e6eea2f9-ec80-388c-
9982-74a018a548e4 results.tekton.dev/v1alpha2.Log      2023-06-29 02:50:05 +0530 IST
2023-06-29 02:50:05 +0530 IST
```

- 다음 명령을 입력하여 레코드에서 파이프라인 실행 또는 작업 실행에 대한 YAML 매니페스트를 검색합니다.

```
$ opc results records get --addr ${RESULTS_API} <record_name> \
  | jq -r .data.value | base64 -d | \
  xargs -0 python3 -c 'import sys, yaml, json; j=json.loads(sys.argv[1]);
print(yaml.safe_dump(j))'
```

명령 예

```
$ opc results records get --addr ${RESULTS_API} \
  results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed/records/e9c736db-5665-
441f-922f-7c1d65c9d621 | \
  jq -r .data.value | base64 -d | \
  xargs -0 python3 -c 'import sys, yaml, json; j=json.loads(sys.argv[1]);
print(yaml.safe_dump(j))'
```

- 선택 사항: 로그 레코드 이름을 사용하여 레코드에서 실행되는 작업의 로깅 정보를 검색합니다. 로그 레코드 이름을 가져오려면 레코드를 레코드 이름에 로그 로 바꿉니다. 다음 명령을 실행합니다.

```
$ opc results logs get --addr ${RESULTS_API} <log_record_name> | jq -r .data | base64 -d
```

명령 예

```
$ opc results logs get --addr ${RESULTS_API} \
  results-testing/results/04e2fbf2-8653-405f-bc42-a262bcf02bed/logs/e9c736db-5665-441f-
922f-7c1d65c9d621 | \
  jq -r .data | base64 -d
```

1.4.3. 결과 검색

CEL(Common Expression Language) 쿼리를 사용하여 결과를 검색할 수 있습니다. 예를 들어 성공하지 못한 파이프라인 실행에 대한 결과를 찾을 수 있습니다. 그러나 관련 정보의 대부분은 결과 개체에 포함되지 않습니다; 이름, 완료 시간 및 기타 데이터로 검색하고 레코드를 검색합니다.

사전 요구 사항

- Tekton 결과가 설치되어 있어야 합니다.

- **opc** 유틸리티를 설치하고 Tekton 결과를 쿼리할 수 있는 환경을 준비합니다.

프로세스

- 다음 명령을 입력하여 CEL 쿼리를 사용하여 결과를 검색합니다.

```
$ opc results list --addr ${RESULTS_API} --filter="<cel_query>" <namespace-name>
```

<namespace_name >을 파이프라인 실행 또는 작업 실행이 생성된 네임스페이스로 바꿉니다.

표 1.1. 결과에 대한 CEL 쿼리 예

목적	CEL 쿼리
실패한 모든 실행의 결과	!(summary.status == SUCCESS)
모든 파이프라인이 주석 ann1 및 ann2 가 포함된 결과	summary.annotations.contains('ann1') && summary.annotations.contains('ann2') && summary.type=='PIPELINE_RUN'

1.4.4. 레코드 검색

CEL(Common Expression Language) 쿼리를 사용하여 레코드를 검색할 수 있습니다. 각 레코드에는 파이프라인 실행 또는 작업 실행에 대한 전체 YAML 정보가 포함되어 있으므로 다양한 기준으로 레코드를 찾을 수 있습니다.

사전 요구 사항

- Tekton 결과가 설치되어 있어야 합니다.
- **opc** 유틸리티를 설치하고 Tekton 결과를 쿼리할 수 있는 환경을 준비합니다.

프로세스

- 다음 명령을 입력하여 CEL 쿼리를 사용하여 레코드를 검색합니다.

```
$ opc results records list --addr ${RESULTS_API} --filter="<cel_query>" <namespace_name>/result/-
```

<namespace_name >을 파이프라인 실행 또는 작업 실행이 생성된 네임스페이스로 바꿉니다. 또는 다음 명령을 입력하여 단일 결과 내에서 레코드를 검색합니다.

```
$ opc results records list --addr ${RESULTS_API} --filter="<cel_query>" <result_name>
```

<result_name >을 결과의 전체 이름으로 바꿉니다.

표 1.2. 레코드에 대한 CEL 쿼리의 예

목적	CEL 쿼리
실패한 모든 작업 실행 또는 파이프라인 실행 기록	!(data.status.conditions[0].status == 'True')

목적	CEL 쿼리
TaskRun 또는 PipelineRun CR(사용자 정의 리소스)의 이름이 run1 인 레코드	<code>data.metadata.name == 'run1'</code>
run1 이라는 PipelineRun CR에 의해 시작된 모든 작업 실행에 대한 레코드	<code>data_type == 'TASK_RUN' && data.metadata.labels['tekton.dev/pipelineRun'] == 'run1'</code>
pipeline1 이라는 Pipeline CR에서 생성된 모든 파이프라인 실행 및 작업 실행 기록	<code>data.metadata.labels['tekton.dev/pipeline'] == 'pipeline1'</code>
pipeline1 이라는 Pipeline CR에서 생성된 모든 파이프라인 실행 레코드	<code>data.metadata.labels['tekton.dev/pipeline'] == 'pipeline1' && data_type == 'PIPELINE_RUN'</code>
TaskRun CR 이름이 hello 로 표시된 모든 작업 실행 레코드	<code>data.metadata.name.startsWith('hello') && data_type == 'TASK_RUN'</code>
완료하는 데 5분 이상 걸리는 모든 파이프라인 실행 레코드	<code>data.status.completionTime - data.status.startTime > duration('5m') && data_type == 'PIPELINE_RUN'</code>
2023년 10월 7일에 완료된 모든 파이프라인 실행 및 작업 실행 기록	<code>data.status.completionTime.getDate() == 7 && data.status.completionTime.getMonth() == 10 && data.status.completionTime.getFullYear() == 2023</code>
세 개 이상의 작업을 포함하는 모든 파이프라인 실행 레코드	<code>size(data.status.pipelineSpec.tasks) >= 3 && data_type == 'PIPELINE_RUN'</code>
ann1 이 포함된 주석이 있는 모든 파이프라인 실행 레코드	<code>data.metadata.annotations.contains('ann1') && data_type == 'PIPELINE_RUN'</code>
ann1 이 포함된 주석과 PipelineRun CR의 이름이 hello 로 시작된 모든 파이프라인 실행 레코드	<code>data.metadata.annotations.contains('ann1') && data.metadata.name.startsWith('hello') && data_type == 'PIPELINE_RUN'</code>

1.4.5. 결과 검색에 대한 참조 정보

결과를 위해 CEL(Common Expression Language) 쿼리에서 다음 필드를 사용할 수 있습니다.

표 1.3. 결과에 대해 CEL 쿼리에서 사용할 수 있는 필드

EL 필드	설명
상위	PipelineRun 또는 TaskRun CR(사용자 정의 리소스)이 생성된 네임스페이스입니다.

EL 필드	설명
uid	결과에 대한 고유 식별자입니다.
annotations	PipelineRun 또는 TaskRun CR에 추가된 주석입니다.
summary	결과에 대한 요약입니다.
create_time	결과 생성 시간입니다.
update_time	결과 마지막 업데이트 시간입니다.

summary.status 필드를 사용하여 파이프라인 실행이 성공했는지 확인할 수 있습니다. 이 필드에는 다음과 같은 값이 있을 수 있습니다.

- 알 수 없음
- **SUCCESS**
- 실패
- **TIMEOUT**
- 취소됨



참고

이 필드의 값을 제공하기 위해 " 또는 ' 와 같은 인용 문자를 사용하지 마십시오.

1.4.6. 레코드 검색에 대한 참조 정보

CEL(Common Expression Language) 쿼리에서 다음 필드를 사용할 수 있습니다.

표 1.4. CEL 쿼리에서 레코드에 사용할 수 있는 필드

EL 필드	설명	값
name	레코드 이름	
data_type	레코드 유형 식별자	Tekton .dev/v1.TaskRun 또는 TASK_RUN tekton.dev/v1.PipelineRun 또는 PIPELINE_RUN results.tekton.dev/v1alpha2.Log
data	작업 실행 또는 파이프라인 실행에 대한 YAML 데이터입니다. 로그 레코드에서 이 필드에는 로깅 출력이 포함됩니다.	

data 필드에 작업 실행 또는 파이프라인 실행에 대한 전체 YAML 데이터가 포함되어 있으므로 CEL 쿼리에서 이 데이터의 모든 요소를 사용할 수 있습니다. 예를 들어 **data.status.completionTime**에는 작업 실행 또는 파이프라인 실행의 완료 시간이 포함됩니다.

1.5. 추가 리소스

- [공통 표현식 언어 사양](#)

2장. OPENSIFT LOGGING OPERATOR를 사용하여 파이프라인 로그 보기

파이프라인 실행, 작업 실행 및 이벤트 리스너로 생성된 로그는 해당 Pod에 저장됩니다. 문제 해결 및 감사의 로그를 검토하고 분석하는 것이 유용합니다.

그러나 Pod를 무기한 유지하면 불필요한 리소스 소비와 누락된 네임스페이스가 발생합니다.

파이프라인 로그를 보기 위해 Pod에 대한 종속성을 제거하기 위해 OpenShift Elasticsearch Operator 및 OpenShift Logging Operator를 사용할 수 있습니다. 이러한 Operator를 사용하면 로그가 포함된 Pod를 삭제한 후에도 [Elasticsearch Kibana](#) 스택을 사용하여 파이프라인 로그를 볼 수 있습니다.

2.1. 사전 요구 사항

Kibana 대시보드에서 파이프라인 로그를 보기 전에 다음을 확인하십시오.

- 단계는 클러스터 관리자가 수행합니다.
- 파이프라인 실행 및 작업 실행에 대한 로그를 사용할 수 있습니다.
- OpenShift Elasticsearch Operator 및 OpenShift Logging Operator가 설치되어 있습니다.

2.2. KIBANA에서 파이프라인 로그 보기

Kibana 웹 콘솔에서 파이프라인 로그를 보려면 다음을 수행합니다.

프로세스

1. OpenShift Container Platform 웹 콘솔에 클러스터 관리자로 로그인합니다.
2. 메뉴 모음 오른쪽 상단에서 **그리드** 아이콘 → **Observability** → **로깅** 을 클릭합니다. Kibana 웹 콘솔이 표시됩니다.
3. 인덱스 패턴을 생성합니다.
 - a. Kibana 웹 콘솔의 왼쪽 탐색 패널에서 **Management** 를 클릭합니다.
 - b. 인덱스 패턴 생성 을 클릭합니다.
 - c. 2 단계 중 2 단계 아래에서 인덱스 패턴 →인덱스 패턴을 정의하고 * 패턴을 입력하고 다음 단계를 클릭합니다.
 - d. 2단계: 설정 구성 → 시간 필터 필드 이름에서 드롭다운 메뉴에서 @timestamp 를 선택한 다음 인덱스 패턴 생성 을 클릭합니다.
4. 필터를 추가합니다.
 - a. Kibana 웹 콘솔의 왼쪽 탐색 패널에서 검색을 클릭합니다.
 - b. 필터 추가 + → 쿼리 DSL 편집을 클릭합니다.



참고

- 다음 예제 필터 각각에 대해 쿼리를 편집하고 저장을 클릭합니다.
- 필터가 차례로 적용됩니다.

- 파이프라인과 관련된 컨테이너를 필터링합니다.

파이프라인 컨테이너를 필터링하는 쿼리의 예

```
{
  "query": {
    "match": {
      "kubernetes.flat_labels": {
        "query": "app_kubernetes_io/managed-by=tekton-pipelines",
        "type": "phrase"
      }
    }
  }
}
```

- place-tools** 컨테이너가 아닌 모든 컨테이너를 필터링합니다. 쿼리 DSL을 편집하는 대신 그래픽 드롭다운 메뉴를 사용하는 방법을 보여 줍니다. As an illustration of using the graphical drop-down menus instead of editing the query DSL, consider the following approach:

그림 2.1. 드롭다운 필드를 사용하여 필터링의 예

- 강조 표시를 위해 라벨에서 **pipelinerun** 을 필터링합니다.

강조 표시를 위해 라벨에서 **pipelinerun** 을 필터링하는 쿼리의 예

```
{
  "query": {
    "match": {
      "kubernetes.flat_labels": {
```

```

"query": "tekton_dev/pipelineRun=",
"type": "phrase"
}
}
}
}

```

iv. 강조 표시를 위해 라벨에서 파이프라인 을 필터링합니다.

강조 표시를 위해 라벨에서 파이프라인 을 필터링하는 쿼리의 예

```

{
  "query": {
    "match": {
      "kubernetes.flat_labels": {
        "query": "tekton_dev/pipeline=",
        "type": "phrase"
      }
    }
  }
}
}
}

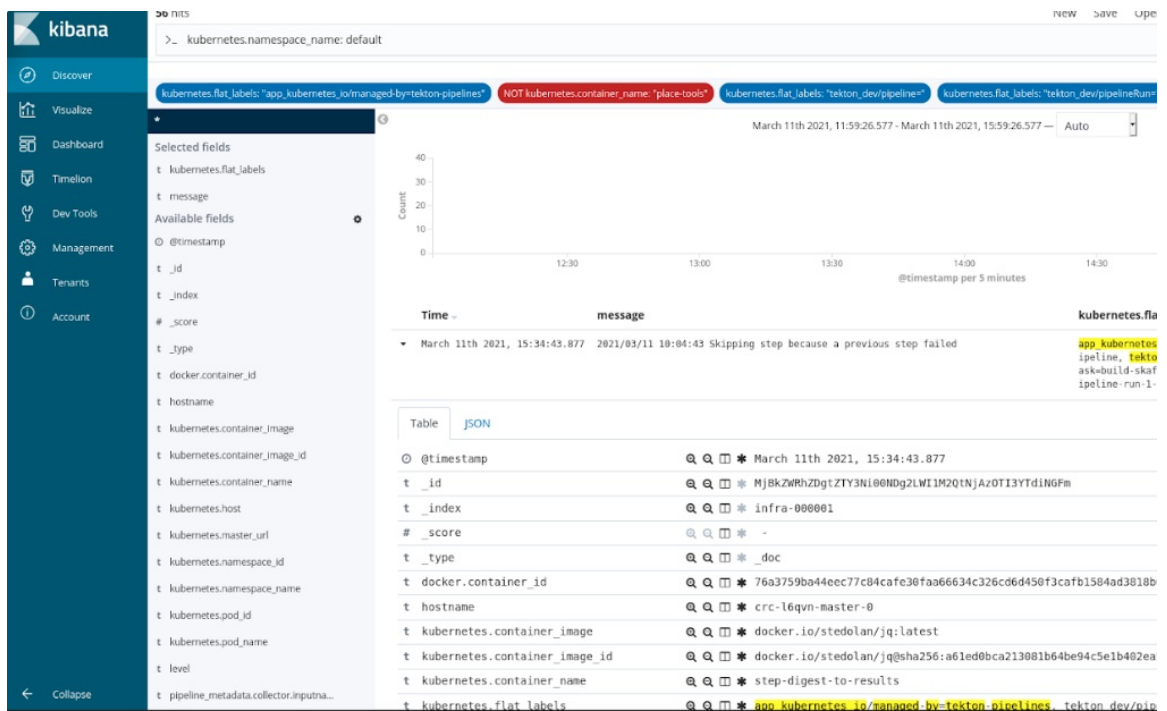
```

c. 사용 가능한 필드 목록에서 다음 필드를 선택합니다.

- **kubernetes.flat_labels**
- **message**
선택한 필드 목록에 선택한 필드가 표시되는지 확인합니다.

d. 로그는 message 필드에 표시됩니다.

그림 2.2. 필터링된 메시지



2.3. 추가 리소스

- [OpenShift Logging 설치](#)
- [리소스의 로그 보기](#)
- [Kibana를 사용하여 클러스터 로그 보기](#)