vRealize Code Stream Trigger for Gerrit

vRealize Code Stream 2.3

This document supports the version of each product listed and supports all subsequent versions until the document is replaced by a new edition. To check for more recent editions of this document, see http://www.vmware.com/support/pubs.

EN-002488-00

vmware®

You can find the most up-to-date technical documentation on the VMware Web site at:

http://www.vmware.com/support/

The VMware Web site also provides the latest product updates.

If you have comments about this documentation, submit your feedback to:

docfeedback@vmware.com

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Contents

Integrating vRealize Code Stream With Gerrit 1

The vRealize Code Stream Trigger for Gerrit integrates vRealize Code Stream with the Gerrit code review lifecycle. The Trigger for Gerrit enables reviews and merges of developer code to trigger a pipeline based on events.

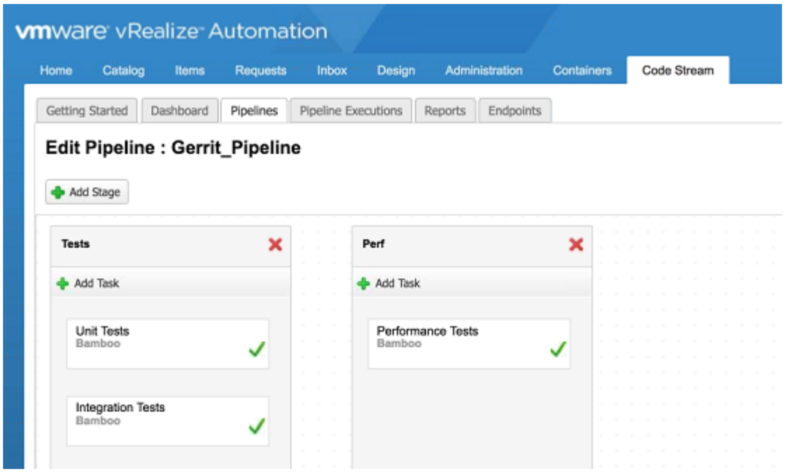
When developers create a patch set for a code change, or merge code changes, Gerrit sends the change events to the trigger for Gerrit. The events trigger the tasks in your pipeline.

For example, your pipeline might include the following stages and tasks.

Table 1-1. Tasks in a Pipeline

Task	Task type
Unit tests	Bamboo
Integration tests	Bamboo
Performance tests	Bamboo

These tasks might appear in your pipeline as shown here:



To connect the trigger for Gerrit to the Gerrit server, you must generate a configuration file. The configuration file establishes an SSH connection to the server. Gerrit sends the change events over the SSH connection, which trigger the pipeline.

Set Up and Run the Trigger for Gerrit 2

You can install and run the trigger for Gerrit on a 32-bit or 64-bit Linux machine, or on your vRealize Code Stream appliance. Then, you must create and run a configuration file for the tool to listen for events.

When a user changes code, that change modifies a file in your repository. Then, the trigger for Gerrit receives the JSON payload of events from the Gerrit server, and uses that payload to trigger the pipeline.

This procedure establishes the SSH connection to the Gerrit server.

Prerequisites

- Verify that Gerrit is installed on your development machine. See https://www.gerritcodereview.com/.
- Verify that you can access https://code.vmware.com to download the binary files.
- Familiarize yourself with the example configuration file and the sections that you must customize. See "Example Configuration File for Gerrit Events," on page 8.

Procedure

1. Download the trigger for Gerrit to your Linux machine or vRealize Code Stream appliance.

   a. Access https://code.vmware.com.

   b. To locate the trigger for Gerrit, search for vRealize Code Stream, and click the link to the latest version.

   c. Download `vrcs-trigger<version>.zip`, which contains the readme, binary, and sample configuration files for the trigger for Gerrit.

2. Install the trigger for Gerrit on your Linux machine or vRealize Code Stream appliance.

   a. Extract the ZIP file to a directory, such as `/opt/vrcs-trigger`.

      The top level directory includes a directory named `dist`, which includes a README file that summarizes the tool commands, samples, and deployment. It also includes the directories named `/bin` and `/bin32`, which contain the binary files. The directory named `samples` contains the sample YAML configuration file.

   b. Export the PATH variable so that it includes the directory named `/bin`.

      ```
      export PATH=$PATH:/opt/vrcs-trigger/dist/bin
      ```

3. Generate your configuration file.

   ```
   vrcsgt generate /your_path/your_YAML-configuration-file
   ```

4. In the configuration file, update the sections named `gerritServer`, `gerritProject`, and `vrcsServer` with your values.

5 Validate the configuration file, which reports any errors.

```
vrcsgt configure gerrittrigger.yaml
```

The trigger for Gerrit verifies the credentials for the vRealize Code Stream instance, and encrypts the password and private key. It logs the results and any error messages to `gerrittrigger/configuration.log`. The result resembles:

```
Logging to gerrittrigger/configuration.log
INFO: 2017/03/15 11:01:59 Validating vRCS Server Configuration.....
INFO: 2017/03/15 11:02:06 Validating Gerrit Server Configuration.....
INFO: 2017/03/15 11:02:06 Rest endpoint connection successful.
INFO: 2017/03/15 11:02:06 Attempting SSH connection....
INFO: 2017/03/15 11:02:07 Running command 'gerrit version'.
INFO: 2017/03/15 11:02:10 Response from server gerrit version 2.11.8
INFO: 2017/03/15 11:02:10 Configuration complete. Peform 'run' to start listening.
```

When the log files reach 1 MB, they rotate, and the old log is saved with the current time stamp.

6 Run the configuration file to have the trigger for Gerrit start to listen for Gerrit events.

```
vrcsgt run gerrittrigger.yaml
```

This command connects the trigger for Gerrit directly to the Gerrit server. It logs the results and any error messages to `gerrittrigger/execution.log`. The result resembles:

```
Logging to gerrittrigger/execution.log
INFO: 2017/01/31 16:15:39 Validating vRCS Server Configuration.....
INFO: 2017/01/31 16:15:40 Validating Gerrit Server Configuration.....
INFO: 2017/01/31 16:15:41 Rest endpoint connection successful.
INFO: 2017/01/31 16:15:41 Attempting SSH connection....
INFO: 2017/01/31 16:15:41 Running command 'gerrit version'.
INFO: 2017/01/31 16:15:44 Response from server gerrit version 2.11.8
INFO: 2017/01/31 16:15:44 Running command 'gerrit stream-events'.
INFO: 2017/01/31 16:16:14 patchset-created -> cs-preflight -> Verified
http://bellevue-ci.eng.vmware.com:8080/6237 refs/changes/37/6237/2 Event Created
```

The tool logs the events from Gerrit. It also logs the status of the pipeline as either started, completed, cancelled, or failed.

You established an SSH connection to the Gerrit server for the trigger for Gerrit to listen for events.

What to do next

Configure your pipeline to receive the input properties. All properties are optional. See Chapter 3, "Configure Your Pipeline to View Gerrit Events," on page 11.

Example Configuration File for Gerrit Events

You create a configuration file to listen for events from Gerrit. The configuration file links events for a specific Gerrit project to the pipeline in vRealize Code Stream.

After an event triggers the pipeline, and the pipeline completes, vRealize Code Stream returns the user configured label to the trigger for Gerrit to bind the tool with the Gerrit server, which defines the interaction.

Table 2-1. Sections in the YAML Configuration File

Section	Description
GerritServer	Identifies where the Gerrit server is hosted. The trigger for Gerrit connects to the Gerrit server through an SSH connection to listen for events. When a pipeline is mapped to an event, the event triggers the pipeline. When the pipeline completes or fails, an action is taken such as to add a comment or a verified label. To connect to the Gerrit server, a user in Gerrit must be in a non-interactive group, and must supply a privateKey to connect over SSH. Note that the trigger for Gerrit only supports a private key without a passphrase. You set the frequency to retry the connection to the server when it appears to be down.
GerritProject	Identifies a specific project in Gerrit. The trigger for Gerrit supports events that occur in the project when developers create patch sets, and when they merge changes after a code review. If the pipeline does not finish in the time indicated, the tool sends a failure message to Gerrit, which indicates that the event is no longer valid.
vrcsServer	Requires the vRealize Code Stream instance where the pipeline resides. When you enter the password the first time, it appears as clear text, but in following steps it is encrypted.

The configuration file includes the following code:

```
gerritServer:
  name: git-listener-1
  URL: https://review.example.com:8080
  username: non-interactive-user
  password: password (non-encrypted the first time)
  privateKeyPath: /path/to/privatekey
  retryWaitInMinutes: 1
gerritProject:
  name: project-name
  events:
  - eventName: patchset-created
    pipelineName: preflight
    verifiedLabel: Verified
  - eventName: change-merged
    pipelineName: postflight
  eventExpiryInDays: 2
vrcsServer:
  name: vrcs-server
  url: https://vrcs.example.com
  username: vrcs-user<@domain>
  password: vrcs-user-password
  tenant: tenant1
  pollIntervalInMinutes: 1
```

After the trigger for Gerrit verifies that the configuration is correct, it maps an event to a pipeline for the specific project. vRealize Code Stream then logs the Gerrit events received, and the status of the pipeline.

Configure Your Pipeline to View Gerrit Events 3

You can configure your pipeline in vRealize Code Stream, and examine the Gerrit events that occurred.

When the trigger for Gerrit receives events from Gerrit through the SSH connection that you established, the tool sends a request to trigger the pipeline.

Prerequisites

- Generate and run the configuration file to establish the SSH connection to the Gerrit server. See Chapter 2, "Set Up and Run the Trigger for Gerrit," on page 7.
- You are familiar with the pipeline configuration input properties to integrate with Gerrit. See "Pipeline Input Properties for Integration with Gerrit," on page 12.

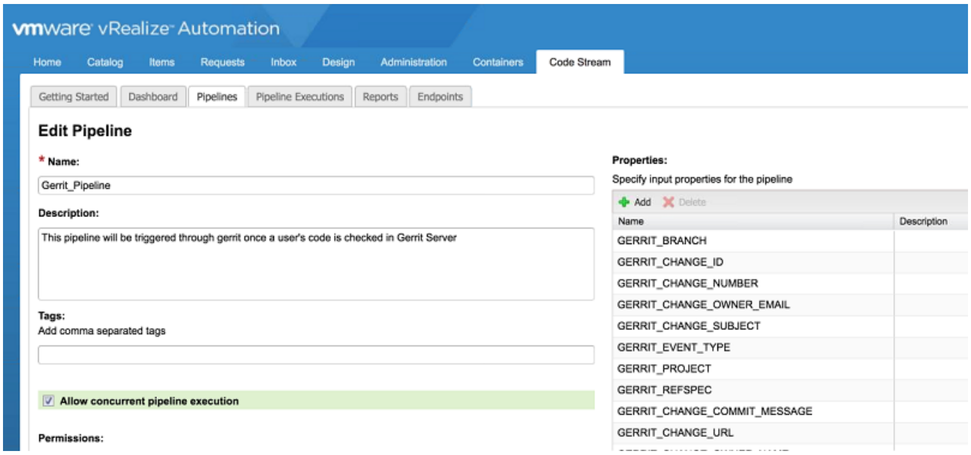
Procedure

1 In vRealize Code Stream, click **Pipelines**.

2 Configure the input properties to your pipeline so that it can receive events from Gerrit.

   a Select the pipeline that you included in the configuration file.

   b Configure the input properties for the pipeline. For example:



Edit Pipeline

* **Name:** Gerrit_Pipeline

Description: This pipeline will be triggered through gerrit once a user's code is checked in Gerrit Server

Tags: Add comma separated tags

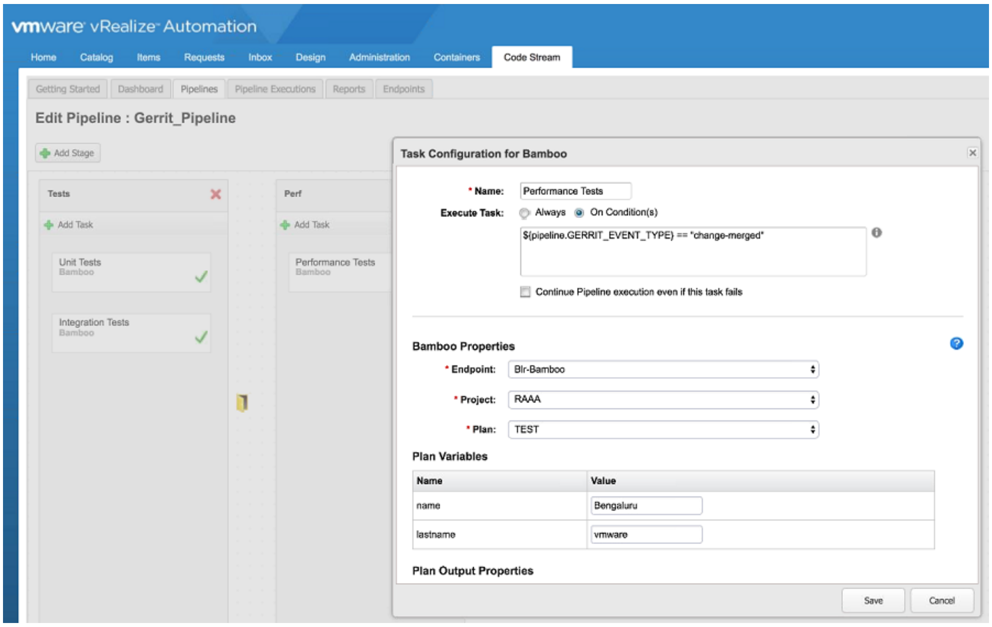
☑ **Allow concurrent pipeline execution**

Permissions:

Properties: Specify input properties for the pipeline

Name	Description
GERRIT_BRANCH	
GERRIT_CHANGE_ID	
GERRIT_CHANGE_NUMBER	
GERRIT_CHANGE_OWNER_EMAIL	
GERRIT_CHANGE_SUBJECT	
GERRIT_EVENT_TYPE	
GERRIT_PROJECT	
GERRIT_REFSPEC	
GERRIT_CHANGE_COMMIT_MESSAGE	
GERRIT_CHANGE_URL	

c. To have a pipeline task trigger only when a developer merges changes in Gerrit, enter the condition string in the Execute Task check box. For example:



Edit Pipeline : Gerrit_Pipeline

Task Configuration for Bamboo

* Name: Performance Tests

Execute Task: ○ Always ● On Condition(s)

${pipeline.GERRIT_EVENT_TYPE} == "change-merged"

□ Continue Pipeline execution even if this task fails

Bamboo Properties

* Endpoint: Blr-Bamboo

* Project: RAAA

* Plan: TEST

Plan Variables

Name	Value
name	Bengaluru
lastname	vmware

Plan Output Properties

When the input property named GERRIT_CHANGE_ID matches the change number of the merged change, the pipeline triggers.

d. Save the task and the pipeline.

3. Click **Pipeline Executions** and view the status of the pipeline that the events triggered.

You configured your pipeline and examined the Gerrit events that triggered the pipeline.

Pipeline Input Properties for Integration with Gerrit

To integrate vRealize Code Stream with Gerrit, you enter input properties in your pipeline configuration.

Gerrit Events

The events that the pipeline receives are based on the input properties that you enter in the pipeline configuration.

The following Gerrit events are allowed.

Table 3-1. Gerrit Input Properties for the Pipeline Configuration

Event	Description
GERRIT_BRANCH	Branch on which the change was sent.
GERRIT_CHANGE_COMMIT_MESSAGE	Commit message.
GERRIT_CHANGE_ID	Identifies a single review or patch. Created when you commit a change.
GERRIT_CHANGE_NUMBER	Number that identifies the change.
GERRIT_CHANGE_OWNER	User name and email of the user who owns the change.
GERRIT_CHANGE_OWNER_EMAIL	Email address for the user who owns the change.
GERRIT_CHANGE_OWNER_NAME	Name of the user who owns the change.
GERRIT_CHANGE_SUBJECT	Subject line of the change sent.

Table 3-1. Gerrit Input Properties for the Pipeline Configuration (Continued)

Event	Description
GERRIT_CHANGE_URL	URL of the change requested.
GERRIT_EVENT_ACCOUNT	User responsible for the event.
GERRIT_EVENT_ACCOUNT_EMAIL	Email address for the user.
GERRIT_EVENT_ACCOUNT_NAME	Name of the user who owns the change.
GERRIT_EVENT_TYPE	Name of the Gerrit event.
GERRIT_HOST	Hostname and IP address of the Gerrit server.
GERRIT_NAME	Name of the Gerrit server.
GERRIT_PATCHSET_NUMBER	Revision number of the patch submitted.
GERRIT_PATCHSET_REVISION	Revision ID of the patch submitted.
GERRIT_PATCHSET_UPLOADER	User who uploaded the patch set.
GERRIT_PATCHSET_UPLOADER_EMAIL	Email address for the user.
GERRIT_PATCHSET_UPLOADER_NAME	Name of the user.
GERRIT_PORT	Port on which the Gerrit server runs SSH.
GERRIT_PROJECT	Name of the Gerrit project.
GERRIT_REFSPEC	Used with `git fetch` to retrieve the actual change.
GERRIT_SCHEME	SSH protocol used to connect to Gerrit.
GERRIT_TOPIC	Name of the local branch created.
GERRIT_URL	URL of the Gerrit server.
GERRIT_VERSION	Version number of the Gerrit server.

Index