Configuring VCS Post-Commit Hooks for TeamCity

- Overview
- Post-commit generic script
- Setting up post-receive hook on Git server
- Setting up hook on Mercurial server
- Setting up post-commit hook on Subversion server
- Setting up post-commit trigger on Perforce server
- Setting up service hook on Team Foundation Server for TFVC and Git
  - TFVC Repository
  - Git Repository
- Troubleshooting

Overview

By default TeamCity uses a polling approach to detect changes in a VCS repository, i.e. for each VCS Root, it periodically sends requests to the version control repository server to find out whether there are new revisions. For large installations with hundreds of VCS roots, this may create a noticeable load on the VCS server and on TeamCity itself.

To avoid background polling, it is possible to set up a post-commit hook on the VCS server, which will notify TeamCity to start checking for changes procedure. This way TeamCity will make background requests for changes detection only when such changes are available.

Even with commit hooks configured and working properly TeamCity still makes requests for changes on the server start and on each build queuing (or starting) to ensure the latest changes are used even if commit hooks stopped to function.

When a commit hook call comes in, TeamCity automatically increases the VCS polling interval (the minimum after the increase is 15 minutes, maximum is 4 hours, increased by 2 times on each successful check). If the commit hook stops working (e.g. TeamCity finds a change in a VCS root which it did not receive a commit hook call for), the VCS polling interval value is reset to default.

Commit hooks are received via TeamCity REST API requests which should typically be configured to in the post-commit repository triggers:

```
POST .../app/rest/vcs-root-instances/commitHookNotification?locator=<vcsRootInstancesLocator>
```

The request returns textual details as to the performed operation or an error message.

It is important to find the "<vcsRootInstancesLocator>" for the request to match only the affected VCS roots from those configured in the TeamCity instance. If too many VCS roots are matched by the request configured in the commit hook, it will lead to more requests and more overload on the VCS repository and TeamCity than using default polling approach. Some examples of the "locator" are provided below.
The request should be performed by a user who has ""View project and all parent projects" permission for all the projects where VCS root is defined.
⚠ Note that by default only the first 100 matched "VCS root instances" will be matched by the request. To match more, "count:9999" can be added as below.

The most common form of the "<vcsRootInstancesLocator>" is "`vcsRoot:(type:<TYPE>,count:99999),property:(name:<URL_PROPERTY_NAME>,value:<VCS_REPOSITORY_URL_PART>,matchType:contains,ignoreCase:true),count:99999`".
However, for VCS roots without parameter %-references, a more performant variance can be used: "`vcsRoot:(type:<TYPE>,property:(name:<URL_PROPERTY_NAME>,value:<VCS_REPOSITORY_URL_PART>,matchType:contains,ignoreCase:true),count:99999),count:99999`".

Commit hooks examples for UNIX-based VCS servers are described below.

Post-commit generic script

Save the script below on a VCS server as `teamcity-trigger.sh`:

```
SERVER=https://buildserver-url
USER=buildserver-user
PASS="<password>"

LOCATOR=$1

# The following is one-line:
(sleep 10;  curl --user $USER:$PASS -X POST
"$SERVER/app/rest/vcs-root-instances/commitHookNotification?locator=$LOCATOR" -o /dev/null)
>/dev/null 2>&1 <&1 &

exit 0
```

Set the variables according to your TeamCity server. The user must have View build configuration settings permission for projects where VCS root is defined. This permission is included in the Project developer role by default.

If your TeamCity server uses a custom SSL certificate, you'll need to pass `-k` or `--cacert /path/to/correct/internal/CACertificate` parameter to the `curl` command above.

Setting up post-receive hook on Git server

1. Locate the Git repository root on the target VCS server. It should contain the **`.git/hooks`** directory with some templates.
2. Create the .git/hooks/post-receive file with a line:

   ```
   /path/to/teamcity-trigger.sh
   'vcsRoot:(type:jetbrains.git,count:99999),property:(name:url,value:<VCS root repository
   URL>,matchType:contains,ignoreCase:true),count:99999'
   ```

   where `<VCS root repository URL>` must be replaced with the repository URL specified in the corresponding TeamCity VCS root and the value should be URL-escaped. Note that locator has `matchType:contains` in it, which means you can specify some part of URL too.file
3. Make sure that both `teamcity-trigger.sh` and `hooks/post-receive` scripts can be read and executed by Git user(s). You may need to execute the following command:

   ```
   chmod 755 /path/to/teamcity-trigger.sh /path/to/git_root/.git/hooks/post-receive
   ```

Setting up hook on Mercurial server

1. Locate the Mercurial repository root on the target VCS server.
2. Create or edit the `.hg/hgrc config` and add the following snippet:

   ```
   [hooks]
   changegroup = /path/to/teamcity-trigger.sh
   'vcsRoot:(type:mercurial,count:99999),property:(name:repositoryPath,value:<VCS root
   repository url>,matchType:contains,ignoreCase:true),count:99999'
   ```

   where `<VCS root repository URL>` must be replaced with the repository URL specified in the corresponding TeamCity VCS root and the value should be URL-escaped. Note that the locator has `matchType:contains` in it, which means you can specify some part of the URL too.

3. Make sure that `teamcity-trigger.sh` is executable. You may need to execute the following command:

```
chmod 755 /path/to/teamcity-trigger.sh
```

Setting up post-commit hook on Subversion server

1. Locate the Subversion repository root containing the `db, hooks, locks,` and other directories. We need the **hooks** dire ctory.
2. Create the **`hooks/post-commit`** file with a line:

```
/path/to/teamcity-trigger.sh 'vcsRoot:(type:svn,count:99999),property:(name:url,value:<VCS root repository url>,matchType:contains,ignoreCase:true),count:99999'
```

   where `<VCS root repository URL>` must be replaced with the repository URL specified in the corresponding TeamCity VCS root and the value should be URL-escaped. Note that the locator has `matchType:contains` in it, which means you can specify some part of URL too.

3. Make sure that both `teamcity-trigger.sh` and `hooks/post-commit` script can be read and executed by the process of the Subversion server. You may need to execute the following command:

```
chmod 755 /path/to/teamcity-trigger.sh /path/to/svn_repository_root/hooks/post-commit
```

Setting up post-commit trigger on Perforce server

Set up a `change-commit` trigger by adding one or several lines when editing `p4 triggers` specification (the text below must be placed in one line, one line per a VCS Root):

```
check-for-changes-teamcity change-commit //depot/project1/... "/path/teamcity-trigger.sh '<VCS Root locator>'"
```

Where <VCS Root locator> can be one of the following:

- for Stream-based VCS roots: `vcsRoot:(type:perforce,count:99999),property:(name:stream,value://streamdepot/streamname,matchType:contains,ignoreCase:true),count:99999`
- for Client-based VCS roots: `vcsRoot:(type:perforce,count:99999),property:(name:client,value:<client name>,matchType:contains,ignoreCase:true),count:99999`
- for Client-mapping VCS roots: `vcsRoot:(type:perforce,count:99999),property:(name:client-mapping,value:<some unique part of client mapping>,matchType:contains,ignoreCase:true),count:99999`

Setting up service hook on Team Foundation Server for TFVC and Git

The latest TFS 2015 and Visual Studio Team Services provides service hooks for code commit events. To create a hook, perform the following steps:

1. Open the admin page for a team project in web access.
2. Create a subscription by running the wizard.
3. Select the "Web Hooks" service to integrate with.
4. Select the "Code checked in" event and specify a filter.
5. Fill in the TeamCity username, password, and server URL in the format:

```
"$SERVER/app/rest/vcs-root-instances/commitHookNotification?locator=$LOCATOR",
```

Where the `$LOCATOR` value depends on the TFS repository type as described in the sections below.

TFVC Repository

```
vcsRoot:(type:tfs,count:99999),property:(name:tfs-url,value:<TFS server
url>,matchType:contains,ignoreCase:true),property:(name:tfs-root,value:<TFS
project>,matchType:contains,ignoreCase:true),count:99999
```

where `<TFS server url>` must be replaced with the value specified in the TFS VCS root URL and path properties. Example:

```
http://teamcity/app/rest/vcs-root-instances/commitHookNotification?locator=vcsRoot:(type:tfs,
count:99999),property:(name:tfs-url,value:http%3A%2F%2Ftfs%3Aport%2Ftfs%2Fcollection,
matchType:contains,ignoreCase:true),property:(name:tfs-root,value:Project,matchType:contain
s,ignoreCase:true),count:99999
```

Git Repository

```
vcsRoot:(type:jetbrains.git,count:99999),property:(name:url,value:<VCS root repository
URL>,matchType:contains,ignoreCase:true),count:99999
```

where `<VCS root repository URL>` must be replaced with the repository URL specified in the corresponding TeamCity VCS root and the value should be URL-escaped. Example:

```
http://teamcity/app/rest/vcs-root-instances/commitHookNotification?locator=vcsRoot:(type:jet
brains.git,count:99999),property:(name:url,value:https%3A%2F%2Faccount.visualstudio.com
%2FDefaultCollection%2FProject%2F_git%2FRepository,matchType:contains,ignoreCase:true),c
ount:99999
```

6. Press Finish to create the service hook.

Troubleshooting

It is recommended to try executing the following command from the command line before configuring the actual hook:

```
curl --user $USER:$PASS -X POST
"$SERVER/app/rest/vcs-root-instances/commitHookNotification?locator=$LOCATOR"
```

If the commit hook matches the VCS root on the server correctly, you should see the output similar to this:

```
Scheduled checking for changes for 1 VCS roots. (Server time: 20160719T192540.787+0300)
```

If the commit hook has not found any VCS roots, it will report an error:

```
No VCS roots are found ...
```

Possible reasons for this output:

- the specified locator is incorrect, it does not match any VCS root on the server
- the specified user does not have enough permission for at least one of the matched VCS roots.

To check what roots are actually matched, use the request (see also details):

```
curl --user $USER:$PASS -X POST "$SERVER/app/rest/vcs-root-instances?locator=$LOCATOR"
```

A commit hook supports matching more than one VCS root, but it is highly recommended to limit the matched VCS roots only to those affected by the change generating the event.

It is recommended to set up a commit hook per a VCS repository. In this case on a check-in to some repository, TeamCity will not spend resources trying to find commits in other non-related VCS roots which were also matched by the commit hook.

By default, the number of VCS roots matched by a commit hook in TeamCity is limited by 100. If you want to match more than 100 VCS roots, add the count parameter: `<locator>,count:1000` Check corresponding request description.