

Integration Strategy Using Gerrit and Migration Plan

Developer Guidelines

Created by Muniyappa, Rangappa, last modified by Mohan, Ravichandran on Apr 13, 2022

- 1 [Login to Gerrit](#)
- 2 [SSH keys setup - Setup gerrit1 server for sync access](#)
 - 2.1 [Git-for-windows](#)
 - 2.2 [Generate ssh key in development workstation](#)
 - 2.3 [Copy generated \(or existing\) ssh pub key to gerrit1.harman.com](#)
 - 2.4 [Preset of username for ssh gerrit access on workstation](#)
- 3 [Steps to clone gerrit project](#)
 - 3.1 [Login to Gerrit](#)
 - 3.2 [Finding project \(or repository\)](#)
 - 3.3 [Cloning the gerrit project](#)
 - 3.4 [Setting Commit template](#)
- 4 [Submitting, reviewing changes](#)
 - 4.1 [Checkout to branch](#)
 - 4.2 [Commit and push commands](#)
 - 4.3 [Pushing changes to existing review](#)
 - 4.4 [Reviewing the Change](#)
- 5 [Merging changes from one branch to other](#)
- 6 [Squashing git commits](#)
- 7 [Git pull --rebase vs. --merge](#)
 - 7.1 [rebasing](#)
 - 7.2 [merging](#)
 - 7.3 [best practice](#)
- 8 [Add Topic to review](#)
 - 8.1 [steps to add Topic to review from UI](#)
 - 8.2 [Set Topic on Push \(from Command line\)](#)
 - 8.3 [Topic Name format](#)

Login to Gerrit

Login to gerrit server <https://gerrit1.harman.com> with your LDAP authentication.

To get access to above URL , please reach out to IT.

SSH keys setup - Setup gerrit1 server for sync access

Git-for-windows

For **windows users**, please download git-bash from below link:

<https://github.com/git-for-windows/git/releases/download/v2.33.0.windows.2/Git-2.33.0.2-64-bit.exe>

Git-for-windows version

Note::

From version Git-for-windows 2.33.1 version and above have some ssh key algorithm changes due to which could not connect to gerrit1 via ssh. git clone wont be working.

So, Please use Git-for-windows version v2.33.0.windows.2 or below

Generate ssh key in development workstation

(Note: You can skip this step if you already have ssh keys generated at ~/.ssh/ with your harman mail id)

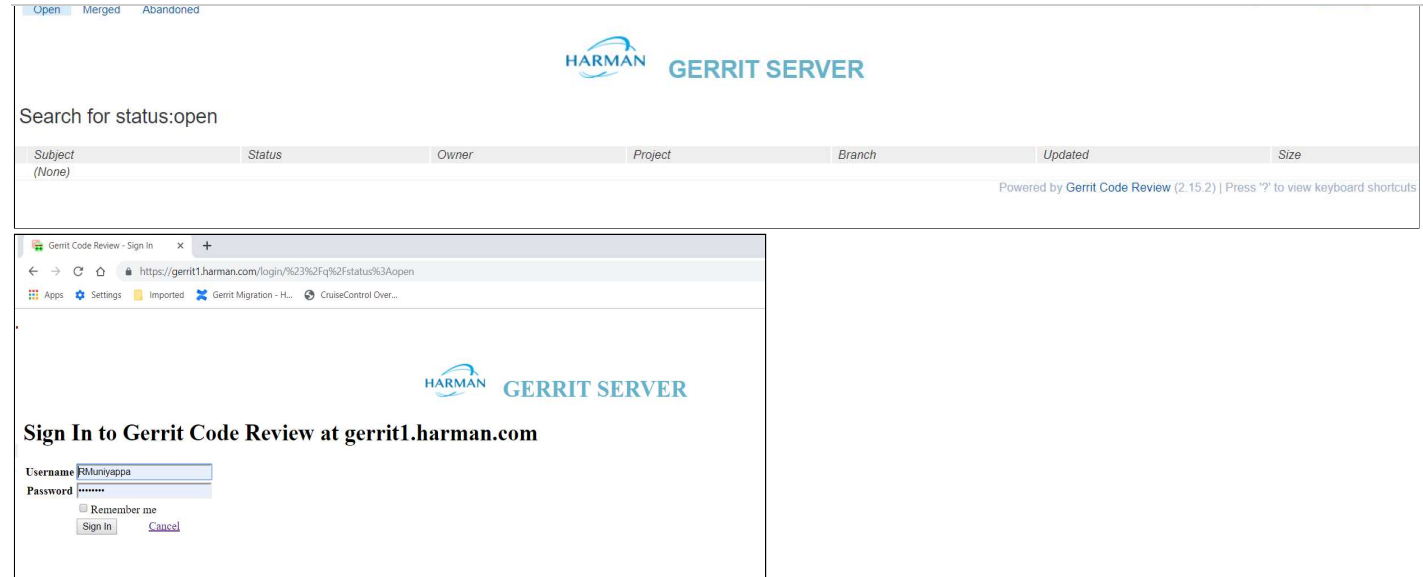
```
ssh-keygen -t rsa -C "email@harman.com"
```

Copy generated (or existing) ssh pub key to gerrit1.harman.com

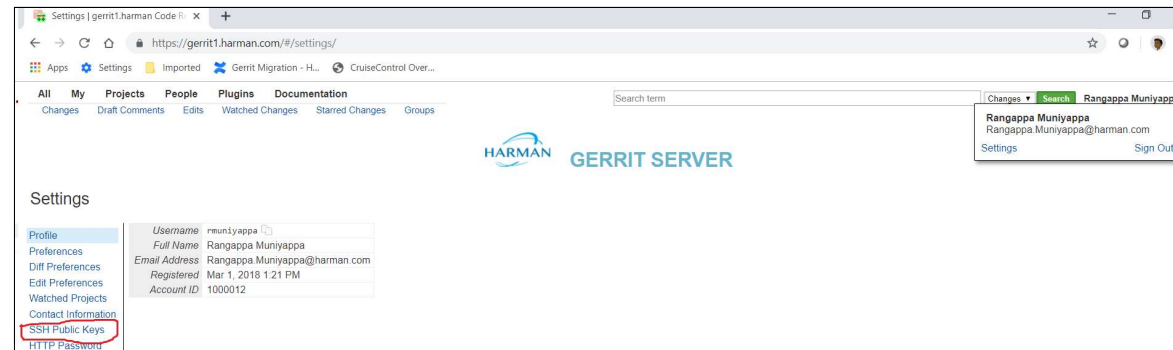
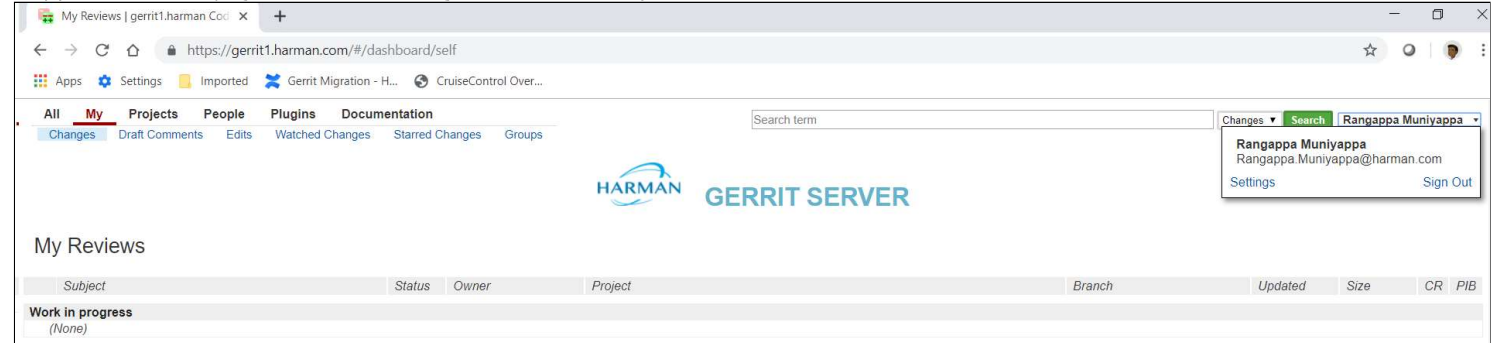
a. Login to gerrit server <https://gerrit1.harman.com/> with your LDAP authentication. (Click on "sign in" at top-right corner)



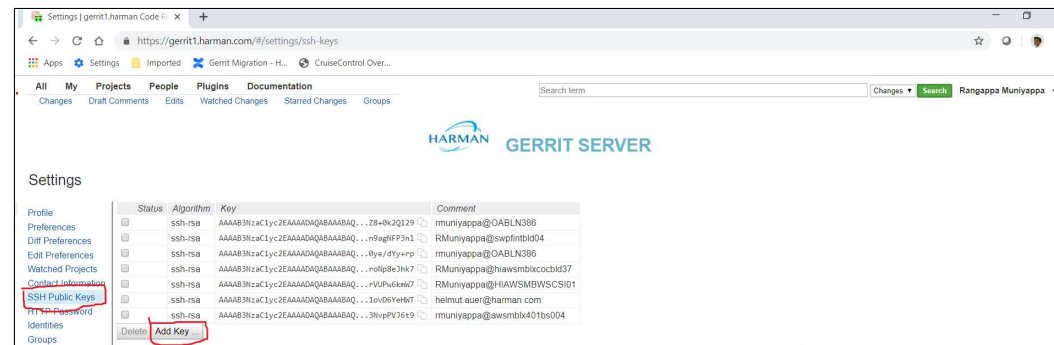
Integration Strategy Using Gerrit and Migration Plan



b. By your name in the top-right corner, select "Settings", then SSH Public Keys



c. Press add key, then paste in the contents of "~/.ssh/id_rsa.pub" that you generated in step "Generate ssh key in development workstation". Then, click on add.



Add SSH Public Key

► How to Generate an SSH Key

Integration Strategy Using Gerrit and Migration Plan

```
dVw4N/Ppt31AR6ZTgLo0NFIAHLkFXbByQOE7CMHbd+S0d12+1xjaCrwatTxxPnHSg8E1bKIk+a6S1PIk
HgvrhV75u4Qx2J5Z4/qxyR9tzmqFP1FadvfnthyVM/T6mxBL6XIdOt+J48j8PoPoHqYxY2aZDIgtNGC9
dfX7Yo/WSkbHbn1PaF09KL2kwuv2BHET7+ITJv1TFFiYLSMIRFEZ
ravichandran.mohan@harman.com
```

Clear Add Close

Preset of username for ssh gerrit access on workstation

if you are not familiar with vim editor, please run this command in git bash to modify the default editor:

```
git config core.editor notepad
```

a. Edit/Create file "~/.ssh/config" (In your development workstation) with below content

##Note: Replace "gerrit1_username" with your gerrit1 username which can be seen from gerrit1.harman.com → your username (on top right) → Settings → profile → username

```
Host gerrit1.harman.com
Port 29418
User gerrit1_username
```

b. Set a Git username

```
git config --global user.name "Firstname Familyname"
```

c. Set an email address in Git

```
git config --global user.email "email@harman.com"
```

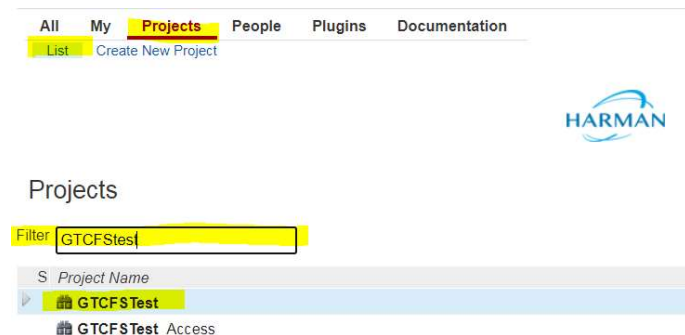
Steps to clone gerrit project

Login to Gerrit

→ Login to gerrit server <https://gerrit1.harman.com> with your LDAP authentication.

Finding project (or repository)

→ Click on **projects** from top menu → and then click on **list** → type project_name in **Filter text box**. Matching projects will be listed down below filter box. click on project



Cloning the gerrit project

→ Once you clicked on project from above, it will take you to "General" section.

→ Click on **"Clone with commit-msg hook"** and **"ssh"**, and then click on copy icon as shown below.

→ Paste it on linux terminal/git-bash window

All	My	Projects	People	Plugins	Documentation
List	General	Branches	Tags	Access	Dashboards
				Create New Project	Reviewers

Integration Strategy Using Gerrit and Migration Plan

Project GTCFSTest

Clone | Clone with commit-msg hook | anonymous http | http | ssh |

git clone ssh://rmohan@gerrit1.harman.com:29418/GTCFSTest && scp -p -P ...

Description

```
ravi@sup710000002:~$ git clone ssh://rmohan@gerrit1.harman.com:29418/GTCFSTest &
& scp -p -P 29418 rmohan@gerrit1.harman.com:hooks/commit-msg GTCFSTest/.git/hook
s/
Cloning into 'GTCFSTest'...
remote: Total 856 (delta 0), reused 856 (delta 0)
Receiving objects: 100% (856/856), 129.47 MiB | 25.21 MiB/s, done.
Resolving deltas: 100% (259/259), done.
Updating files: 100% (536/536), done.
commit-msg
100% 4780 4.7KB/s 00:00
```

Setting Commit template

For getting commit-msg template please refer this page [Gerrit : Commit-Message-Format](#)

Submitting, reviewing changes

Checkout to branch

```
cd <cloned-directory>      #change directory to cloned directory
git branch -a              #To see all local and remote branches
git checkout <branch-name>  #branch-name where you want to work, then modify files through your favorite editor
```

Commit and push commands

Once done with the changes in project, follow below commands to push your changes,

```
git add <file-name>        #file-name is which you modified
git commit                 #format will be open, modify it with proper data, otherwise push will be denied
                             #confluence for detailed info: https://confluence.harman.com/confluence/display/PFINTG/Gerrit+%3A+Commit-Mess
git log --decorate         #check how may commits you are pushing, because gerrit will create one review for each commit in local
                             #one review will be created for each commit above the remote branch Example_remote_branch: "origin/<branch>"
git push origin HEAD:refs/for/<branch-name> #branch-name is where your changes you want to push (target branch)
                             #Ex: git push origin HEAD:refs/for/master --> to push your changes to master.
```

Pushing changes to existing review

User already pushed review. From the same workspace, if user want to modify on the same change or review.

```
git add <file-name>        #file-name - modified file name
git commit --amend         #to amend the changes on the same change or review.
                             #check once change-id displaying in commit message, matching with Change-Id in gerrit review UI.
                             #If change-Id are matching, changes will go to same review.
git push origin HEAD:refs/for/<branch-name> #to push your changes to remote.
```

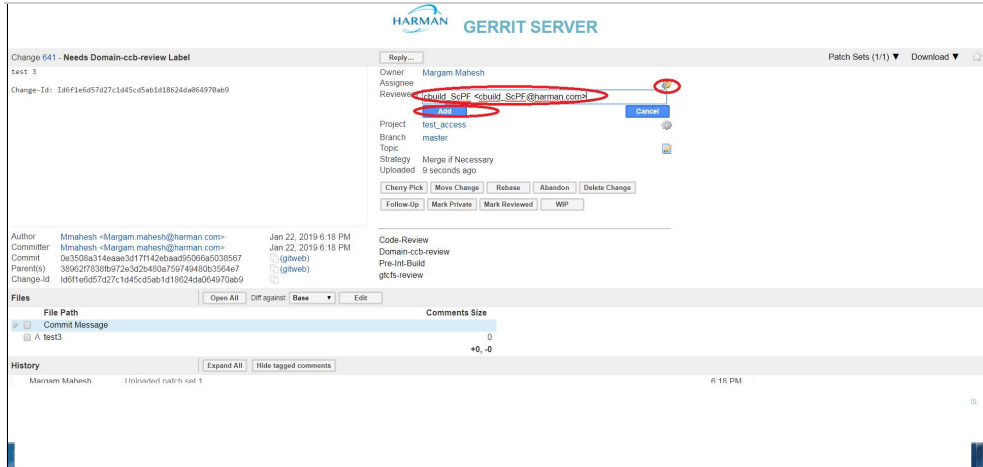
Reviewing the Change

User chose to manually add as a reviewer, gerrit offers other ways for reviewers to find changes, including:

- Selecting Open from the Changes menu
- Setting up email notification to stay informed of changes even if you are not added as a reviewer.

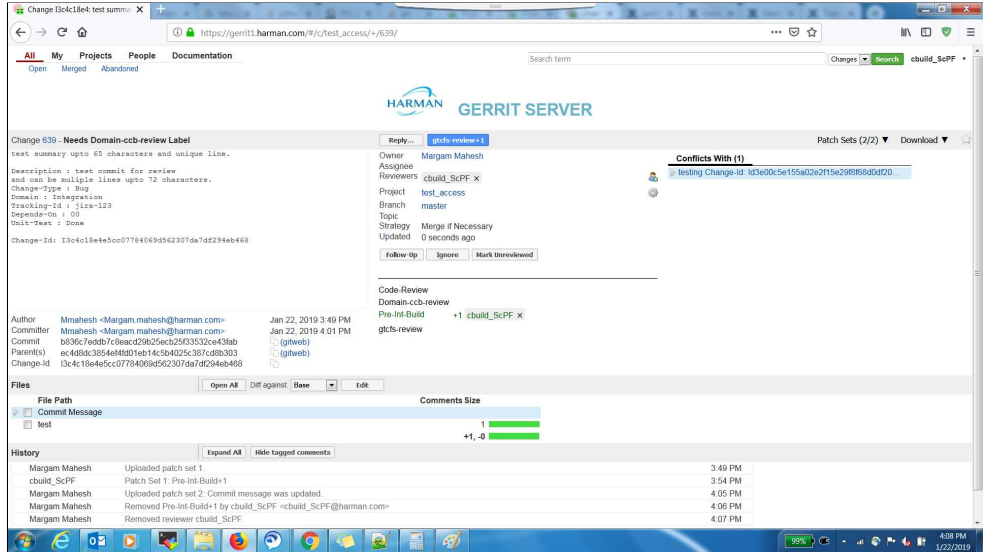
Please find below image to add as a reviewer.

tegration Strategy Using Gerrit and Migration Plan

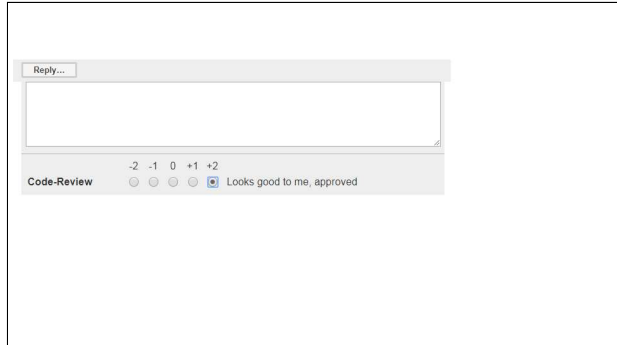


Once Change is pushed, CI build will be triggered, and it will post the review.

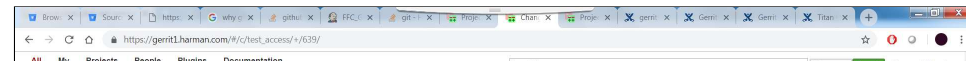
if the Pre-Int-Build is fine, it will post the review as below.



if there is no other dependency owners, it can be reviewed using code-review.



if it depends on domain-ccb-review and gtcfs-review, it has be reviewed first and then code-review will be enabled.



Integration Strategy Using Gerrit and Migration Plan

Click on submit to merge the change into the respective branch.

Merging changes from one branch to other

Below are the steps to merge your changes from one branch to other:

```
git branch -a
git checkout <dev-branch>
git checkout <main-branch>
git merge --no-ff <dev-branch>

## opens commit message window. update it.

git push origin HEAD:refs/for/<main_branch>
```

Example:

```
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ git branch
master
* merge_test
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ ls
abc helloworld.c merge Readme.txt test.txt
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ vim merge_test
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ git add .
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ git commit
```

```
[merge_test f440501] merge 1
1 file changed, 1 insertion(+)
```

tegration Strategy Using Gerrit and Migration Plan

```
Counting objects: 100% (4/4), done.
Delta compression using up to 20 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 445 bytes | 445.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1)
remote: Processing changes: new: 1, done
remote:
remote: New Changes:
remote:   https://gerrit1.harman.com/#/c/GTC_Foundation_Software/Integration/sandbox/+/9029 merge 1
remote:
To ssh://gerrit1.harman.com:29418/GTC_Foundation_Software/Integration/sandbox
* [new branch]      HEAD -> refs/for/merge_test
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ vim merge_test
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ git add .
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ git commit
[merge_test bc4ac82] 2nd merge
1 file changed, 1 insertion(+)
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ git push origin HEAD:refs/for/merge_test
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 20 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (6/6), 704 bytes | 704.00 KiB/s, done.
Total 6 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3)
remote: Processing changes: new: 1, done
remote:
remote: New Changes:
remote:   https://gerrit1.harman.com/#/c/GTC_Foundation_Software/Integration/sandbox/+/9030 2nd merge
remote:
To ssh://gerrit1.harman.com:29418/GTC_Foundation_Software/Integration/sandbox
* [new branch]      HEAD -> refs/for/merge_test
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ ls
abc  helloworld.c  merge  merge_test  Readme.txt  test.txt
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ git branch
  master
* merge_test
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ git merge --no-ff merge_test
Merge made by the 'recursive' strategy.
 merge_test | 2 ++
1 file changed, 2 insertions(+)
 create mode 100644 merge_test
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$ git push origin HEAD:refs/for/master
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 372 bytes | 372.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0)
remote: Processing changes: new: 1, done
remote:
remote: New Changes:
remote:   https://gerrit1.harman.com/#/c/GTC_Foundation_Software/Integration/sandbox/+/9031 Merge branch 'merge_test'
remote:
To ssh://gerrit1.harman.com:29418/GTC_Foundation_Software/Integration/sandbox
* [new branch]      HEAD -> refs/for/master
Mmahesh@swpfintbld02:/data/work/Mmahesh/work_new/gerrit/sandbox$
```

Squashing git commits

Refer page [Squashing git commit for Gerrit code review](#)

Git pull --rebase vs. --merge

rebasing

If you pull remote changes with the flag --rebase, then your local changes are reapplied on top of the remote changes.

```
git pull --rebase
```

merging

If you pull remote changes with the flag `--merge`, which is also the default, then your local changes are merged with the remote changes. This results in a merge commit that points to [tegration Strategy Using Gerrit and Migration Plan](#)

```
git pull --merge
```

best practice

It is best practice to **always rebase your local commits** when you pull before pushing them. As nobody knows your commits yet, nobody will be confused when they are rebased but the additional commit of a merge would be unnecessarily confusing. Published commits are, however, usually merged, for example when branches are merged.


Add Topic to review

steps to add Topic to review from UI

Go to review page from browser. On left pane, you can find field "Topic". Click on "ADD TOPIC". Add a topic and save

NOTE: Please DO NOT add space, comma and other special characters in the topic name. "_" and "-" are allowed

Branch master

Parent e4144b1 

Topic ADD TOPIC

Strategy Merge if Necessary

Set Topic on Push (from Command line)

It is also possible to set a topic on push, by appending `%topic=...` to the ref name like below

NOTE: Please DO NOT add space, comma and other special characters in the topic name. "_" and "-" are allowed

```
#Example:

git push origin HEAD:refs/for/master%topic=feature_ALM_12345


git push origin HEAD:refs/for/master%topic=bugfix_ELVIS_12345
```

Topic Name format

<feature>_ALM_<number>
<bugfix>_ELVIS_<number>

No labels

1 Comment

 **Raji, Arjun**

Does gerrit1 has some limitation with ed25519 keys ?