|  |  |
| --- | --- |
| **(Confidential)** | |
| Scope of disclosure | [TSDV] for ROMLINUX only |
| Period of confidentiality |  |
| Head of Information Owner | Head of Engineering Dept. |
| Handling restriction | N/A |

**User Manual**

**GERRIT SOUCE CODE MANAGEMENT**

**Toshiba Software Development (Vietnam) Co., Ltd**

|  |
| --- |
| Document ID: TSDV-ROMLINUX-GERRIT-UM |
| Total: 16 Page No. 1 |

**Revision History**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Rev. No.  (X.YY) | Date (YYYY-MM-DD) | Section No. Changed | Change Description | Author | Reviewed by | Approved by |
| 0.01 | 2015-11-17 | All | Initialize | LuyenTM | LamDN |  |
| 0.02 | 2015-11-17 | All | Update document | LamDN |  |  |
| 0.03 | 2015-11-18 | All | Update some pictures | LuyenTM | LamDN |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

**Contents**

[1. Introduction 6](#_Toc436388749)

[1.1. Overview 6](#_Toc436388750)

[1.2. Scope 6](#_Toc436388751)

[1.3. Definitions, Acronyms, Abbreviations 6](#_Toc436388752)

[1.4. References 7](#_Toc436388753)

[2. Installation & Configuration 8](#_Toc436388754)

[2.1. System Requirements 8](#_Toc436388755)

[2.2. Install Gerrit 8](#_Toc436388756)

[2.3. Install Gerrit Plugins 8](#_Toc436388757)

[2.4. Control Gerrit service 8](#_Toc436388758)

[2.5. Configure User Account with LDAP Server 8](#_Toc436388759)

[2.6. Configure Replication Plugin 9](#_Toc436388760)

[2.7. Import Existing Project to Gerrit 9](#_Toc436388761)

[2.8. Manage Permissions 9](#_Toc436388762)

[3. Working on Gerrit step-by-step 10](#_Toc436388763)

[3.1. Login 10](#_Toc436388764)

[3.2. Clone Project 11](#_Toc436388765)

[3.3. Pull Request 11](#_Toc436388766)

[3.4. Push Tag 12](#_Toc436388767)

[3.5. Rules 12](#_Toc436388768)

[4. Development Process for TZCS Team 13](#_Toc436388769)

[4.1. Overview 13](#_Toc436388770)

[4.2. Flow Chart 14](#_Toc436388771)

[4.3. Sequence Diagram 15](#_Toc436388772)

**LIST OF TABLES:**

Table 1.1 Definitions, Acronyms, Abbreviations 6

Table 1.2 References 7

**LIST OF FIGURES:**

No table of figures entries found.

# Introduction

## Overview

Gerrit is one of the best web-based Git Management Servers that helps to manage source code of ROMLINUX Project. It contains a lot of useful features such as code review (same as pull request of Github), user friendly UI, replication to another remote Git server, etc.

This document describes in detail how to install and use Gerrit. It also defines a development process for all members of TZCS Team.

## Scope

Gerrit is used internally by TZCS team for ROMLINUX project.

## Definitions, Acronyms, Abbreviations

Table 1.1 Definitions, Acronyms, Abbreviations

|  |  |  |
| --- | --- | --- |
| ID No | Acronyms | Definition |
| 1 | TSDV | Toshiba Software Development Vietnam |
| 2 | TZCS | Toshiba Zero Client System |
| 3 | PC | Personal Computer |

## References

Table 1.2 References

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID No | Reference | Version Number | Document Name/Source & Location reference | Source | Remarks |
|  |  |  |  |  | N/A |

# Installation & Configuration

## System Requirements

* Git 2.1.4
* Java Runtime Environment 1.7.0\_79
* Debian 3.16.7-ckt11-1+deb8u4 (2015-09-19) x86\_64 GNU/Linux
* IP address for Gerrit Server: 10.116.41.71

## Install Gerrit

Download Gerrit version what we need in this link: http://gerrit-releases.storage.googleapis.com/index.html

Run to install into server: java -jar gerrit.war init --batch -d ~/gerrit\_romlinux

For example: if we install in server 10.116.41.71, please check in Web browser: http://10.116.41.71:8080[http://10.116.41.71:8080](http://10.116.41.71:8080/). If we install successfully, we can see a Web UI of Gerrit.

## Install Gerrit Plugins

For TZCS we need 03 plugins:

* Download-command : support download commit (even that commit never merge in our source)
* Replication: support update all change in source code to other server source
* Gitblit : show graph UI about project

We can install these plugins by SSH or directly on Gerrit server

* For SSH : ssh -p 29418 10.116.41.71 gerrit plugin install - -n plugin.jar < plugin.jar
* For Server : copy file plugin.jar into ~/gerrit\_romlinux/plugin on Gerrit server.

## Control Gerrit service

Restart: gerrit\_romlinux/bin/gerrit.sh restart

Start: gerrit\_romlinux/bin/gerrit.sh start

Stop: gerrit\_romlinux/bin/gerrit.sh stop

Linkweb: git config -f ~/gerrit\_romlinux/etc/gerrit.config gerrit.canonicalWebUrl

## Configure User Account with LDAP Server

With this work, all of members on LDAP mail server can become member on Gerrit server. We only control access permission of all members. It means that any member can login in Gerrit system, but no project visible if they have not any permission.

Edit file gerrit/etc/gerrit.config as follow:

[gerrit]

basePath = git

canonicalWebUrl = http://10.116.41.71:8080/

[database]

type = h2

database = db/ReviewDB

[index]

type = LUCENE

[auth]

type = LDAP

[ldap]

server = ldap://ldap.tsdv.com.vn:389

accountBase = ou=Users,dc=ldap,dc=tsdv,dc=com,dc=vn

accountPattern = (&(objectClass=person)(uid=${username}))

accountFullName = displayName

accountEmailAddress = mail

[sendemail]

from = git@toshiba-tsdv.com

includeDiff = true

smtpServer = mail.toshiba-tsdv.com

[container]

user = gerrit

javaHome = /usr/lib/jvm/java-7-openjdk-amd64/jre

[sshd]

listenAddress = \*:29418

[httpd]

listenUrl = http://\*:8080/

[cache]

directory = cache

[plugins]

allowRemoteAdmin = true

## Configure Replication Plugin

Create file gerrit\_romlinux/etc/replication.config

[remote "meta-tzcs"]

url = gitosis@10.116.41.86:project/DSPc/${name}.git

push = +refs/heads/\*:refs/heads/\*

## Import Existing Project to Gerrit

Create a new project

ssh -p 29418 user@localhost gerrit create-project --name meta-tzcs

Change directory to old project which we need to import to Gerrit. Using below script to import all the thing into server Gerrit

# This script auto import all data

project="meta-tzcs"

branches=`git branch -a | cut -d/ -f3 `

for branch in $branches; do

git checkout $branch

git push ssh://luyentm@10.116.41.71:29418/$project \*:\*

done

## Manage Permissions

Admin must assign permission for all users.

For the most comfortable, please separate user into small group

Example:

* admin\_group : all permission
* test\_group : read, review code, comment, and so on
* dev\_group : read, push (pull request), comment, and so on

In which:

* dev only push into reference : refs/for/\*
* admin can do all thing : refs/\*
* if admin want someone only have permission to push into one branch, please using:

refs/for/name\_branch

# Working on Gerrit step-by-step

## Login

Go to website : http://10.116.41.71:8080[http://10.116.41.71:8080](http://10.116.41.71:8080/)

Using LDAP mail account to login into gerrit website

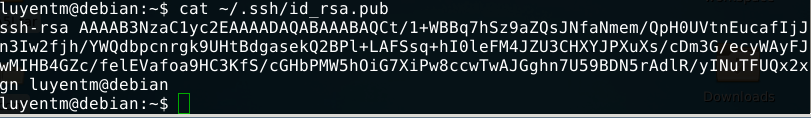
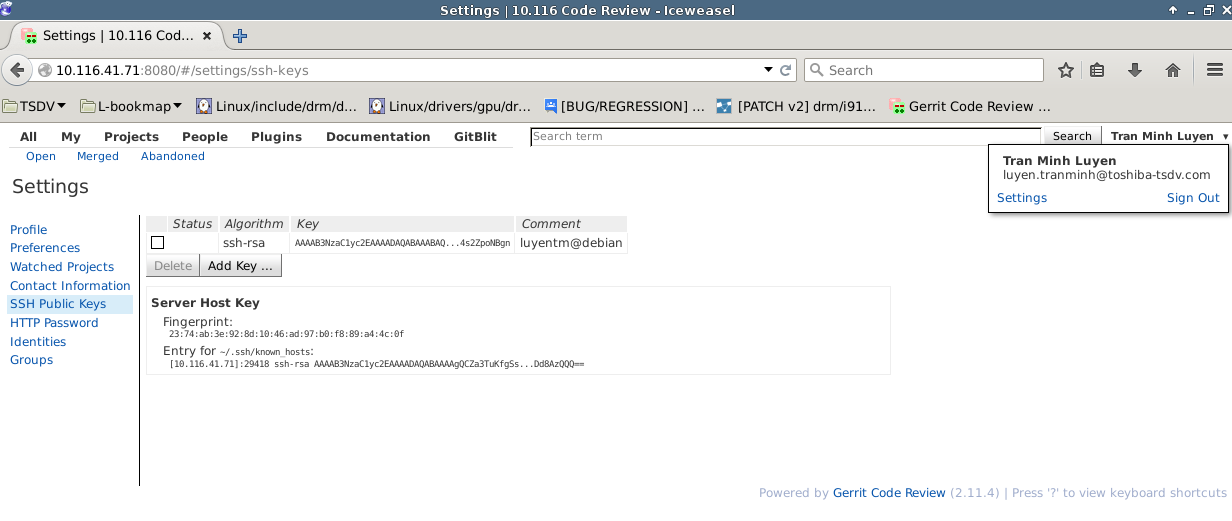
by pass authentication using SSH public key

Create ssh key

ssh-keygen -t rsa

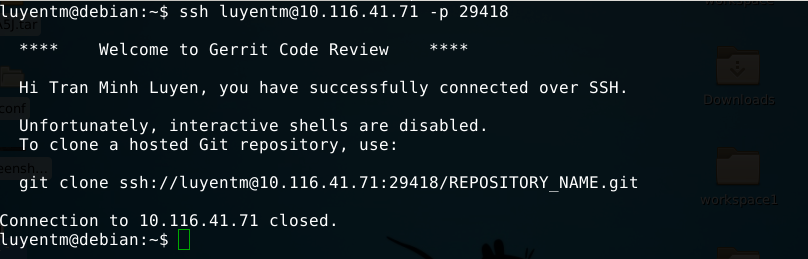
Copy your public key and paste to your web (Settings/SSH Public Keys)

cat .ssh/id\_rsa.pub

Test again

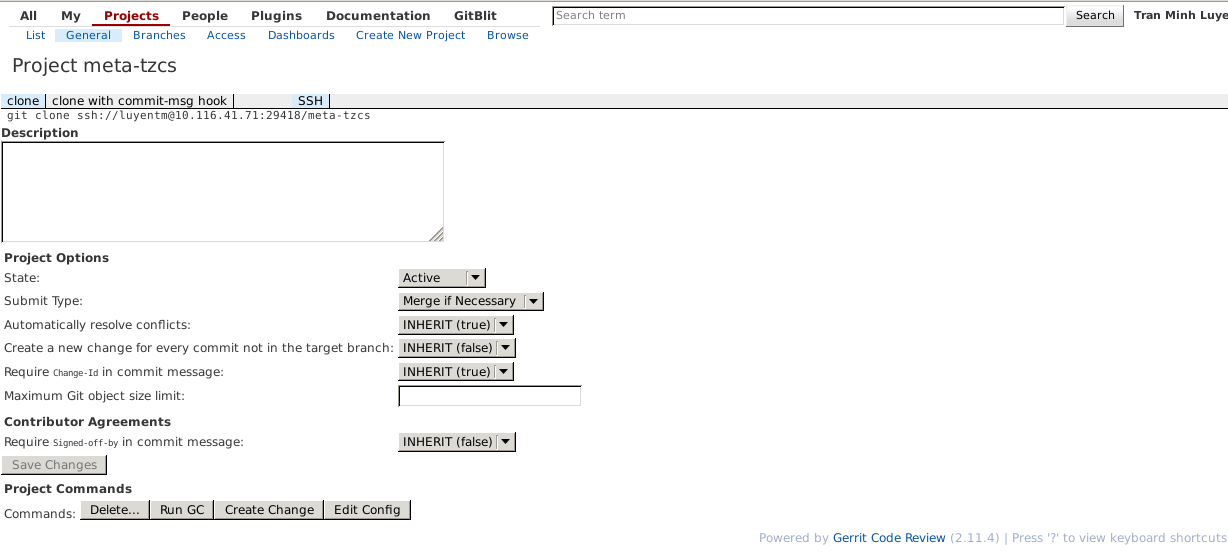
ssh luyentm@10.116.41.71 -p 29418

=> if server not require authentication => OK



## Clone Project

Login into server git, link to project which you need, find out link about SSH:



Example:

git clone ssh://username@10.116.41.71[//luyentm@10.116.41.71](mailto://luyentm@10.116.41.71):29418/meta-tzcs

## Pull Request

Pull request (request to merge source code into server)

git push origin HEAD:refs/for/branch

Example:

git push origin HEAD:refs/for/daisy-dspc-tsdv-vdi-citrix-temp-v1.2-tamtv

When we meet error about missing changedID

gitdir=$(git rev-parse --git-dir); scp -p -P 29418 luyentm@10.116.41.71:hooks/commit-msg $ {gitdir}/hooks/

Read carefully hint in error message and do as guide

## Push Tag

Create tag: $git tag –a {TagName} –m “{TagMessage}” {CommitID}

Push tag: $git push –tags

Note: Consider add “Force Push” right of “Push”

## Rules

You must pull all source code before push anything

Never using SSH private key from other

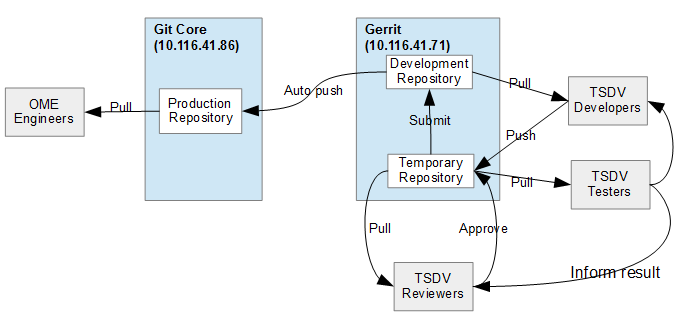
Test carefully before push into server

Commit the most clearly as you can

# Development Process for TZCS Team

## Overview

This section describes the development process relating to managing source code on Gerrit. Basically, it is illustrated in this figure below:



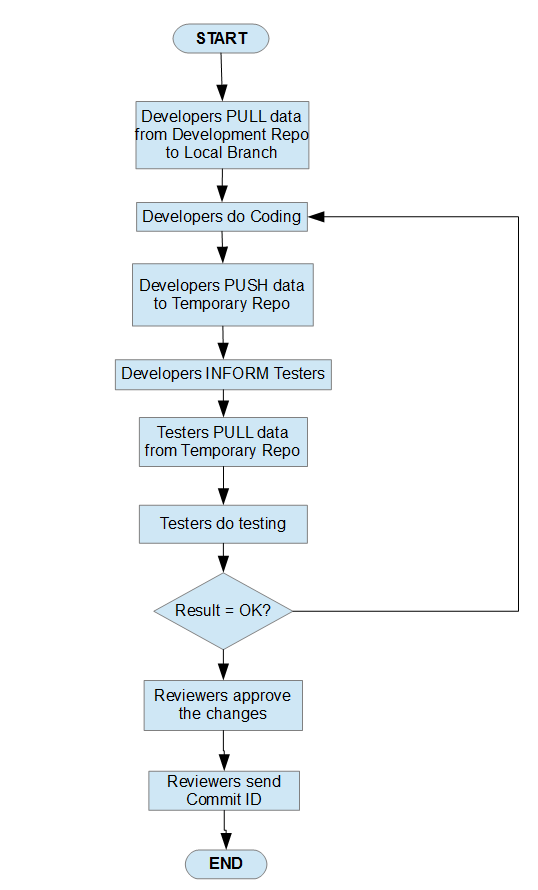
Basically, we have 03 roles including: Reviewer, Developer and Tester. In which:

* Developer: Who are in charge of developing software. Before pushing data to Temporary Repository, Developer should pull data from Development Repository first.
* Tester: Who are in charge of testing data being pushed by Developer. He/she informs back to Reviewer and Developer if testing is done.
* Reviewer: Who are in charge of reviewing source code being pushed by Developer on Temporary Repository. Reviewer also be able to pull data to his/her local machine to check again. After that, he/she approves the changes to submit data to Development Repository. Data is automatically synchronized with Production Repository located at Git Core Server.

We define the development process as follows:

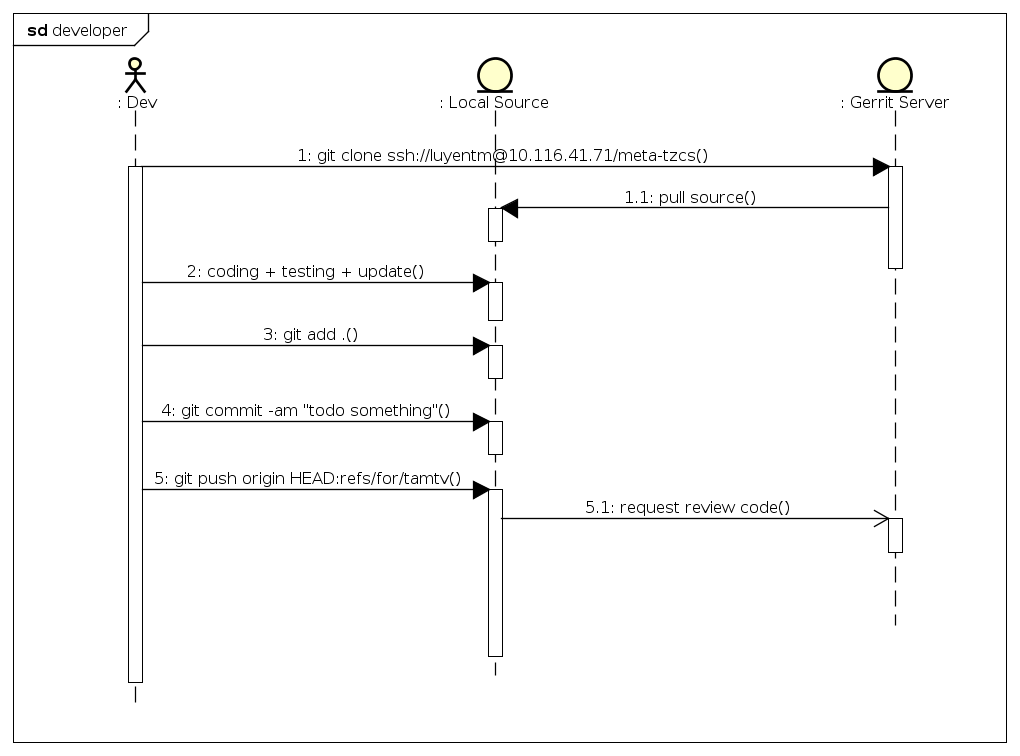
* (1) TSDV developers PULL source code from Development Repository (master branch) to local branch on his/her PC/Laptop
* (2) TSDV developers coding on the local branch.
* (3) TSDV developers PUSH his/her changes to Temporary Repository.
* (4) TSDV developers INFORM to testers to do testing.
  + Developers update Design document
* (5) TSDV testers pull new data on Temporary Repository to his/her local branch.
* (6) TSDV testers test and inform to TSDV reviewers and TSDV developers
  + Testers update Test Spec/Test Result document
  + If testing failed, TSDV developers come back to Step (2)
  + If testing succeeded, go to next step (7).
* (7) TSDV reviewers approve the changes to submit to Development Repository.
  + TSDV reviewers should PULL data to test again for sure if possible.
* (8) GERRIT automatically push data to Production Repository
* (9) TSDV reviewers notify to OME the changes with specific commit ID(s)

## Flow Chart



## Sequence Diagram

**For Developer:**

**Step by step:**

1. Dev clone source code from server:

git clone ssh://username@10.116.41.71/meta-tzcs

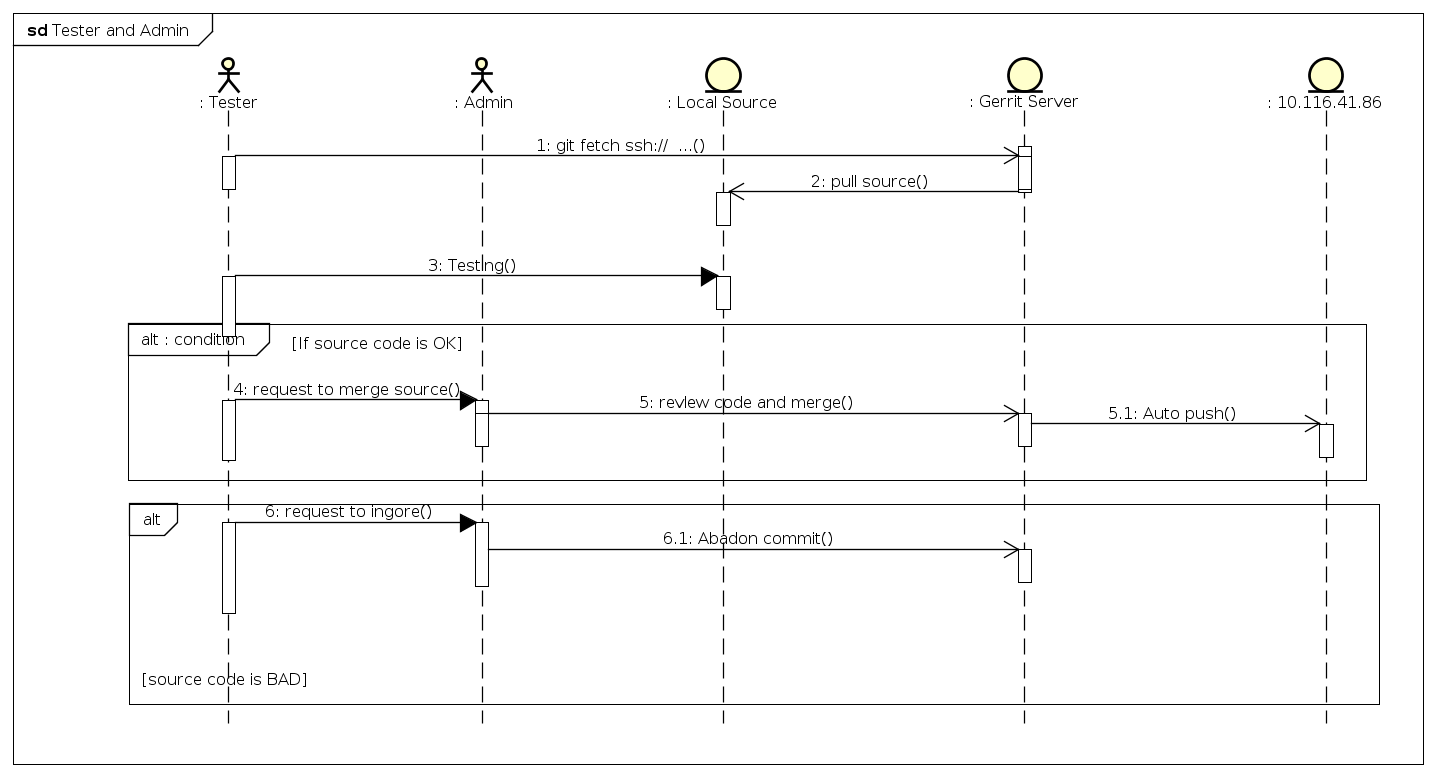
1.1. In case of Dev already cloned, Dev pull the newest source code to local branch

2. Dev implement task

3. Dev push request into server

git push origin HEAD:refs/for/branch

**For Tester and Admin**



**Step by step:**

1. Tester clone source with exactly commit want to test

git fetch ssh:// … find our command in web site

2. Tester builds and testing features

3. Tester goes to website and set point review, submit or abandon commit

4. Admin improves or abandon commit