Installation and backup information for FS24 Gitlab and TeamCity

# Introduction

Document describes various installation folder location s and dependent systems for running, backup and restore of applications.

# GItLab

Version: 8.2.3

Web Url: <http://gitlab.fs24.local/>

Server: gitlab.fs24.local

Admin Credentials: root/gitlab15

### Installation:

1. curl https://packages.gitlab.com/install/repositories/gitlab/gitlab-ce/script.deb.sh | sudo bash
2. sudo apt-get install gitlab-ce

### Smtp Setup:

1. Edit **/etc/gitlab/gitlab.rb** and add following lines:

gitlab\_rails['smtp\_enable'] = true

gitlab\_rails['smtp\_address'] = "mx03.fs24.local"

gitlab\_rails['smtp\_port'] = 25

# gitlab\_rails['smtp\_user\_name'] = "smtp user"

# gitlab\_rails['smtp\_password'] = "smtp password"

gitlab\_rails['smtp\_domain'] = "financscout24.de"

# gitlab\_rails['smtp\_authentication'] = "login"

# gitlab\_rails['smtp\_enable\_starttls\_auto'] = true

# gitlab\_rails['smtp\_openssl\_verify\_mode'] = 'peer'

gitlab\_rails['gitlab\_email\_from'] = 'fs24gitlab@financescout24.de'

gitlab\_rails['gitlab\_email\_reply\_to'] = 'noreply@financescout24.de'

gitlab\_rails['smtp\_openssl\_verify\_mode'] = 'none'

gitlab\_rails['gitlab\_email\_enabled'] = true

## Backup

<https://gitlab.com/gitlab-org/gitlab-ce/blob/master/doc/raketasks/backup_restore.md>

/etc/gitlab/gitlab-secrets.json

{

"gitlab\_shell": {

"secret\_token": "37b18e84a3638aab434dc7659db8332e8c9970215b199f188b20ab1b9789e5977f31959444bcfd3fd8b295e005066eec1dd61207b23cbdff0de9ae0dbce6ea1e"

},

"gitlab\_rails": {

"secret\_token": "58f7c4a38fea1ecf7060449a52fdb3af8109f5436e32dfb31c75c0226bf4f6712a1f93161ba1a3db81f07ab0938ebf7a7ce0becb60e57941a8359807017e7d00"

},

"gitlab\_ci": {

"secret\_token": null,

"secret\_key\_base": "89c7113c556335486f0b55591170acbe1dad0e03c73b1352c7067da49e1aec7b2505f4766962ccb7da6166a4a22fcd0a26fe0a08d5953365fe077bd4c3b69a9c",

"db\_key\_base": "ee2dd59ecbbc82d47069e3ecfedac2162531cc6c2844b6307a5e2225f0412927d6a8ad12105d3e855eea92545238fe1af66a8b374965ea43cebf2351332c4b57"

},

"mattermost": {

"email\_invite\_salt": "ce44944de8f951baaf298547fc76149e",

"file\_public\_link\_salt": "622e8203346b57a4495fc72ecc8cd5b0",

"email\_password\_reset\_salt": "3265d52d5d40c0dbe8715b2da6a342a2",

"sql\_at\_rest\_encrypt\_key": "d29ce26ba501b3e7668d840806b4fdfa"

}

}

### Create a backup of the GitLab system

A backup creates an archive file that contains the database, all repositories and all attachments. This archive will be saved in backup\_path (see config/gitlab.yml). The filename will be [TIMESTAMP]\_gitlab\_backup.tar. This timestamp can be used to restore an specific backup. You can only restore a backup to exactly the same version of GitLab that you created it on, for example 7.2.1. The best way to migrate your repositories from one server to another is through backup restore.

You need to keep a separate copy of /etc/gitlab/gitlab-secrets.json (for omnibus packages) or /home/git/gitlab/.secret (for installations from source). This file contains the database encryption key used for two-factor authentication. If you restore a GitLab backup without restoring the database encryption key, users who have two-factor authentication enabled will lose access to your GitLab server.

If you are interested in GitLab CI backup please follow to the CI backup documentation\*

sudo gitlab-rake gitlab:backup:create

### Backup location

/var/opt/gitlab/backups

### Configure cron to make daily backups

For omnibus installations

To schedule a cron job that backs up your repositories and GitLab metadata, use the root user:

sudo su -

crontab -e

There, add the following line to schedule the backup for everyday at 2 AM:

0 2 \* \* \* /opt/gitlab/bin/gitlab-rake gitlab:backup:create CRON=1

You may also want to set a limited lifetime for backups to prevent regular backups using all your disk space. To do this add the following lines to /etc/gitlab/gitlab.rb and reconfigure:

# limit backup lifetime to 7 days - 604800 seconds

gitlab\_rails['backup\_keep\_time'] = 604800

## Restore a previously created backup

You can only restore a backup to exactly the same version of GitLab that you created it on, for example 7.2.1.

Prerequisites

You need to have a working GitLab installation before you can perform a restore. This is mainly because the system user performing the restore actions ('git') is usually not allowed to create or delete the SQL database it needs to import data into ('gitlabhq\_production'). All existing data will be either erased (SQL) or moved to a separate directory (repositories, uploads).

If some or all of your GitLab users are using two-factor authentication (2FA) then you must also make sure to restore **/etc/gitlab/gitlab-secrets.json** (Omnibus) or /home/git/gitlab/.secret (installations from source). Note that you need to run gitlab-ctl

reconfigure after changing gitlab-secrets.json.

### Steps:

**First** make sure your backup tar file is in /var/opt/gitlab/backups

sudo cp 1393513186\_gitlab\_backup.tar /var/opt/gitlab/backups/

**Next**, restore the backup by running the restore command. You need to specify the timestamp of the backup you are restoring.

# Stop processes that are connected to the database

sudo gitlab-ctl stop unicorn

sudo gitlab-ctl stop sidekiq

# This command will overwrite the contents of your GitLab database!

sudo gitlab-rake gitlab:backup:restore BACKUP=1393513186

# Start GitLab

sudo gitlab-ctl start

# Check GitLab

sudo gitlab-rake gitlab:check SANITIZE=true

# Teamcity

Team city is setup to use mysql running as local instance on teamcity.fs24.local server.

Version: 9.1.4

Web Url: <http://teamcity.fs24.local:8111>

Server: teamcity.fs24.local

Installation folder: /opt/jetbrains/TeamCity/

Admin Credentials: admin/teamcity

## INSTALLATION PROCESS

<http://blog.fire-development.com/2014/09/23/teamcity-8-setup-on-linux/>

## Team City Server resources

Data Directory:

/opt/jetbrains/TeamCity/.BuildServer

## my sql Database resources

Database type: MySQL

root/so2dBcioGhWfNk7Z3LQ0

Connection URL: jdbc:mysql:///teamcity?user=teamcityuser

Schema: teamcity  
Password: teamcityuser /teamcity

## Backup

Location:

/opt/jetbrains/TeamCity/backups

Configuration:

/opt/jetbrains/TeamCity/.BuildServer/config/backup-config.xml

<backup-settings>  
 <general>  
 <backup-dir path="/opt/jetbrains/TeamCity/backups"/>  
 </general>  
</backup-settings>

### Backup Process:

Need to create a Cron job script with following steps:

1. Make a **post** request to
2. <http://teamcity.fs24.local:8111/httpAuth/app/rest/server/backup?includeConfigs=true&includeDatabase=true&includeBuildLogs=true&fileName=fs24_>
3. check status at <http://teamcity.fs24.local:8111/httpAuth/app/rest/server/backup>
   1. return Idle when complete
4. Copy/move the **fs24\_\*.zip** file from backup location **/opt/jetbrains/TeamCity/backups**

## Restoring from backup:

### Following link describes:

<https://confluence.jetbrains.com/display/TCD9/Restoring+TeamCity+Data+from+Backup#RestoringTeamCityDatafromBackup-Performingfullrestore>

### Performing full restore

To perform full restore a TeamCity server from a backup file:

1. Install the TeamCity server from a tar.gz or .exe installation package. Do not start the TeamCity server.
2. Create a new empty TeamCity Data Directory.
3. Select one of the options:
   1. To restore the backup into a new external database, create and configure the database, placing the database.properties file into any directory other than the TeamCity Data Directory.
   2. To restore the backup into the internal database, save the code below to the database.properties file and place the file into any directory other than TeamCity Data Directory:
      1. # Database: HSQLDB (HyperSonic) version 2.x
      2. connectionUrl=jdbc:hsqldb:file:$TEAMCITY\_SYSTEM\_PATH/buildserver
4. Place the required database drivers into the lib/jdbc sub directory.
5. Use the maintainDB utility located in the <TeamCity Home>/bin directory.
6. Use the restore command:
   1. maintainDB.[cmd|sh] restore -A <absolute path to the newly created TeamCity Data Directory> -F <path to the TeamCity backup file> -T <absolute path to the database.properties file of the target database>
   2. The -A argument can be omitted if you have the TEAMCITY\_DATA\_PATH environment variable set.
   3. The -F argument can be an absolute path or a path relative to the <TeamCity Data Directory>/backup directory.

By default, if no other option except -F is specified, all of the backed up scopes will be restored from the backup file. To restore only specific scopes from the backup file, use the corresponding options of the maintainDB utility: -D, -C, -U, -L, and -P.

To get the reference for the available options of maintainDB, run the utility without any command or option.

You can also copy the files that were not included into the backup into the data directory (most importantly, build artifacts, located in <TeamCity Data Directory>/system/artifacts by default), see details on the directories in the TeamCity Data Directory description.

### Restoring database only

Before restoring a TeamCity database to an existing server, make sure the Teamcity server is not running.

To restore a TeamCity database only from a backup file to an existing server:

1. Create and configure the database, placing the database.properties file into any directory other than the TeamCity Data Directory.
2. Ensure that the required database drivers are present in the TeamCity Data Directory/lib/jdbc sub directory.
3. Use the maintainDB utility located in the <TeamCity Home>/bin directory (only available in TeamCity .tar.gz and .exe distributions).
4. Use the restore command:
   1. maintainDB.[cmd|sh] restore -A <absolute path to the newly created TeamCity Data Directory> -F <path to the TeamCity backup file> -T <absolute path to the database.properties file of the target database> -D
5. See the maintainDB utility console output. You may have to copy the database.properties file manually if requested.