



CGS 7.X Installation Instructions

Eran Shoucroun

Installation Guide

**Preliminary Comments**

1. Make sure that you are connected to the machine using the root user.
2. If you want to use the replication mechanism in the installation, take into consideration that:
   1. On the Master machine: Install Everything.
   2. On the Slave machine: Install Everything.
   3. On the arbiter machine: Install only Tomcat and MongoDB. However, copy and untar the CGS installation, without actually installing it.

# Pre-requisites

In this section a list of install prerequisites are mentioned that must be fulfilled before the install can begin.

All installations are located in the following location:   
[\\192.168.44.3\sqa$\GlobalAvailability\ProductionInstallations](file:///\\192.168.44.3\sqa$\GlobalAvailability\ProductionInstallations)

## Java and Tomcat

1. Copy **installEnv\_linux64.tar.gz** to /install
2. Extract tar file by executing:  
   # tar -xvf installEnv\_linux64.tar.gz
3. # cd installEnv\_linux64
4. Edit env.properties  
   make sure:  
   installDir=/opt/t2k  
     
   and set values for:

keystoreFileName=lmsdev.com.ks

keystorePassword=changeit

1. Install by executing:  
   #./installEnv.sh
2. Edit the startup.sh file:

On machines with 15 GB memory: JAVA\_OPTS="-server –Xmx6000m –Xms6000m …

On machines with 8 GB memory: JAVA\_OPTS="-server –Xmx3000m –Xms3000m …

## MongoDB – Single installation

1. Copy **mongo-install-2.4.3.tar.gz** to /install
2. Extract tar file by executing:  
   # tar -xvf mongo-install-2.4.3.tar.gz
3. # cd mongo-install
4. Install by executing:  
   # ./ install.sh

## MongoDB – replication installation

**On all machines:**

1. On all machines: Add the following lines to the hosts file (or just make sure that they are there):

<primary-ip> cgs-primary.timetoknow.cloud

<secondary-ip> cgs-secondary.timetoknow.cloud

<arbiter-ip> cgs-arbiter.timetoknow.cloud

1. Copy **mongo-install-2.4.3.tar.gz** to /install
2. Extract tar file by executing:  
   # tar -xvf mongo-install-2.4.3.tar.gz
3. # cd mongo-install
4. Install by executing:
   1. On primary machine  
       # ./ install.sh
   2. On secondary and arbiter

# ./ install2rep.sh

1. Stop the mongo on all machines.

# service mongod stop

1. Run

# cd replication

1. Run

# ./setup\_mongo\_replication.sh

1. You will be asked to reboot. Ignore any errors as long as you are asked to reboot. Reboot by running

# reboot

1. When the machine comes up make the necessary changes in the iptables. For now, this means running:

# service iptables stop

12 Change to **true** **mongoUseReplicaSet** property into t2k.properties when you install LMS

**On Primary machine only:**

1. Open the mongo shell by running:

# mongo -u admin -p rap2el admin

1. On the mongo shell run:

# config = {\_id: 't2kReplSet', members: [{\_id: 0, host: ' cgs-primary.timetoknow.cloud:27017', priority: 2},{\_id: 1, host: ' cgs-secondary.timetoknow.cloud:27017', priority: 1}]}

# rs.initiate(config)

Before continue keep on pressing ENTER to make sure that the current machine is PRIMARY

# rs.addArb("cgs-arbiter.timetoknow.cloud:27017")

1. Run

# rs.status()

Again and again until you see that cgs-primary.timetoknow.cloud is primary, cgs-secondary.timetoknow.cloud is secondary and cgs-arbiter.timetoknow.cloud is arbiter.

1. Exit the mongo shell by running:

# exit

# Installing the CGS

1. Copy **cgs-install-x.x.tar.gz** to /install  
   [File name is changed on every daily build.]
2. Extract tar file by executing:  
   # tar -xzf cgs-install-x.x.tar.gz
3. # cd cgs-install-x.x
4. Edit t2k.properties  
   installDir=/opt/t2k  
   packagedOutputLocation=/var/t2k/cgs/content   
   catalogueUrl=http://{GCR\_HOST}/gcr/rest/catalogue  
   appletsUrl=http://{GCR\_HOST}/gcr/rest/applets  
   appletsDownloadDir=/var/t2k/cgs/download

cmsHome=/var/t2k/cgs/cms

1. Install CGS by executing:  
   **./install.sh {action}**  
   where action is one of: **When installing a secondary use "skip.db"**

|  |  |
| --- | --- |
| create.all | (re)create the DB and CMS folder from scratch, usually for first time installation, or if you need to drop existing DB and delete existing CMS   * All existing data is LOST! |
| upgrade.db | upgrade existing DB   * upgrade DB and preserve existing data |
| skip.db | don't do any DB changes   * preserve existing data |

1. In QA only: Edit /opt/t2k/tomcat/webapps/cgs/WEB-INF/classes/config/mongo.properties:

mongo.replicaSet = cgs-qa-primary.timetoknow.cloud:27017,cgs-qa-secondary.timetoknow.cloud:27017,cgs-qa-arbiter.timetoknow.cloud:27017

## CMS replication.

1. Copy **cms-replication.tar.gz** and **unison-precompiled-2.40.102.tar.gz** to /install directory on master and slave machines.
2. On both machines unzip tar files by running **tar xvf \*.tar.gz**
3. Run **cd unison-2.40.102** on both machines.
4. Run **cp -Rp unison /usr/bin** on both machines.
5. Run on master and slave:

**mkdir ~/.ssh**

1. On primary machine check if crond service exist by running: **service crond status**
2. If not:
   1. Run **yum install vixie-cron crontabs**
   2. Run **/sbin/chkconfig crond on**
   3. Run **/sbin/service crond start**
3. Go accordingly installation instructions into README file in cm-replication directory.

Now you can put the tomcat back up: (Both machines)

# service Tomcat7T2K start

## Troubleshooting

### File System doesn't replicate

In case the file system isn't replicated (recall that the replication should occur after approximately one minute), repeat the bidirectional ssh stage ("Basic Installations and Configurations" above) on both machines.

### Running Unison manually

In case you want to run unisom manually (e.g. to see its output or to save the one-minute wait), go to the master and run:

# cd ~

# ./replication.sh cgs-secondary.timetoknow.cloud

# Disaster Recovery actions

## On Master Crash

1. Create a new master machine.
2. On all machines, update the new ip of the master (for the name "cgs-primary.timetoknow.cloud") in the hosts file.
3. On the new master, do all pre-requisite section and cgs-installation section.
4. Stop the tomcat on the new master (it was probably started during the installation)

**Mongo Recovery**

1. On the master, go to the replication directory inside the installation directory and run

# ./setup\_mongo\_replication.sh

You will be asked to reboot. Reboot.

1. On the master, enter the mongo shell
2. Run

# rs.status()

Again and again until you see that cgs-primary.timetoknow.cloud is primary.

**CMS Recovery**

1. Go to the new master.
2. Repeat the cms section in the disaster recovery installation section **on both machines.**
3. Run

# mkdir –p <cms-dir>

# scp cgs-secondary.timetoknow.cloud:/<cms-dir> <cmd-sir>

1. Only when you see that the cms is completely replicated, bring the tomcat up.

**Coming back up**

1. On the new master, start mongo and then tomcat:

# service mongod start

# service Tomcat7T2K start

1. On the slave, restart mongo:

# service mongod start

## On Slave Crash

1. Create a new slave machine.
2. On both machines, update the new ip of the slave (for the name "cgs-secondary.timetoknow.cloud") in the hosts file.
3. do all pre-requisite section and cgs-installation section.
4. Stop the tomcat and mongo on the slave.

**MONGO**

1. On the salve, go to the replication directory inside the installation directory and run

# ./setup\_mongo\_replication.sh

You will be asked to reboot. Reboot.

1. On the master, enter the mongo shell
2. Run

# rs.status()

Again and again until you see that cgs-primary.timetoknow.cloud is primary.

**CMS**

1. Go to the new slave.
2. Repeat the entire disaster recovery installation section.
3. Run

# mkdir –p <cms-dir>

# scp cgs-secondary.timetoknow.cloud:/<cms-dir> <cmd-sir

## On arbiter crash

1. Get a new arbiter machine.
2. Update all hosts files in all machines with its new ip.
3. go to the replication directory inside the installation directory and run

# ./setup\_mongo\_replication.sh

You will be asked to reboot. Reboot.

1. Go to the master machine.
2. Run the mongo shell:

# mongo

1. In the mongo shell, run

# rs.status() until you see that the arbiter is set to be an arbiter, and is up and running.