Installation on Linux using scripts

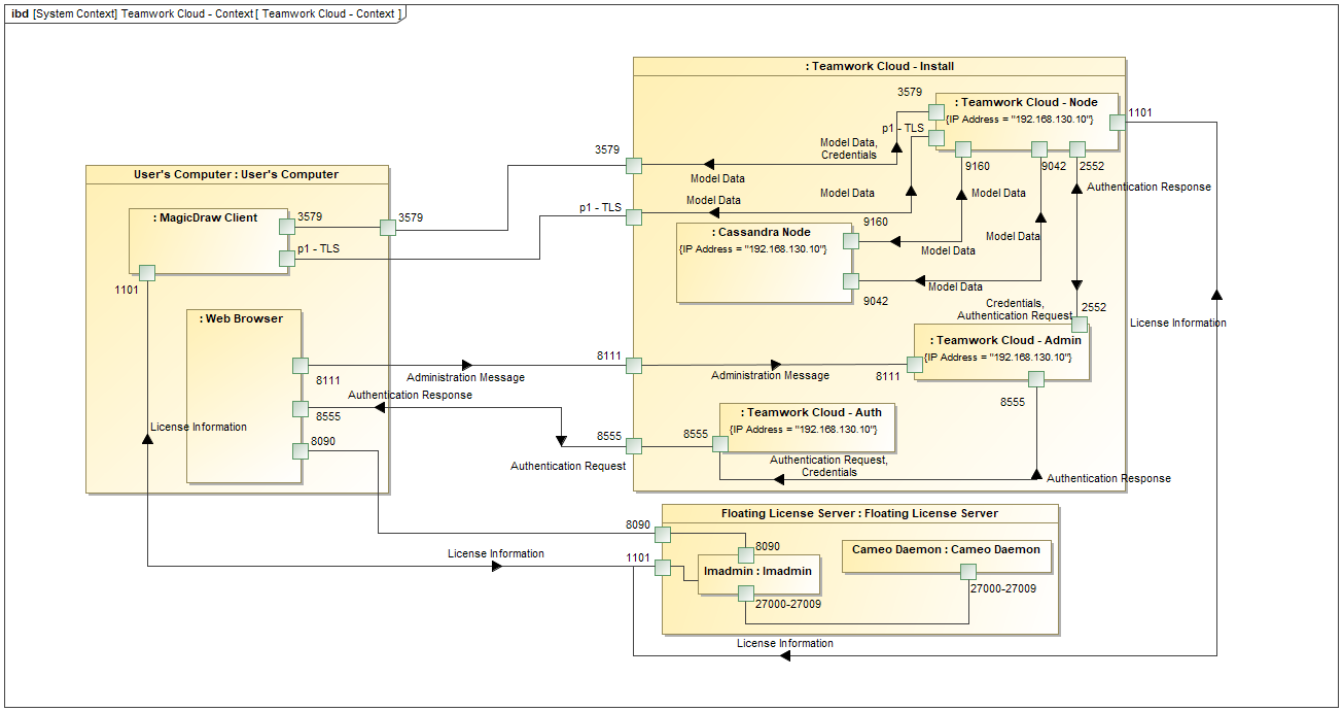
On this page:

Scripts

The following are the authserver and installation script files used in this example:

- backup.sh
- backup_all.sh
- install_flex_centos7.sh
- install_java_172.sh
- install_twc19_centos7.sh
- install_cassandra3_11_centos7.sh
- restore-single_node.sh
- fixcassandraservice.sh

This page shows how to install and configure Teamwork Cloud (TWCloud) Standard Edition on Centos 7.x, deployed on a single server. It also provides the configuration for installing the Teamwork Cloud node as well as the underlying Cassandra node on the same server.



TWCloud installation and configuration on Centos 7.x on a single server.

Minimum server system requirements

- 8 Processor Cores - i.e. Quad-Core Hyper-threaded CPU (such as Intel E3-1230 or faster).
- 32 GB RAM (Motherboard with an ECC RAM is always preferred on any critical database server).
- Linux (RedHat/CentOS 7), 64 bit. Scripts need to be modified slightly for Centos 6.6 - 6.9 due to different system daemon and firewall configurations.

Please read the article for additional server recommendations for capacity and performance in the following link:

https://docs.datastax.com/en/dse-planning/doc/planning/planningHardware.html

If you use SATA drives and not SSD's, we recommend using a caching controller with BBU, configured for write back. In this configuration (single node Cassandra), we recommend using RAID - the aforementioned link is referring to multi-node Cassandra deployments where native Cassandra replication is in place, which is not the case in this single node instance.

In order to install a full working environment, you would need:

- Oracle Java (Java Hotspot) 1.8.0_172
- A FlexNet License Server
- Cassandra 3.11.2
- Teamwork Cloud

Preparing the operating system

Partitioning the drives

Prior to installing Cassandra, it is important to understand how Cassandra utilizes disk space in order to properly configure the host server.

Disk space depends on usage, so it's important to understand the mechanism. The database writes data to disk when appending data to the commit log for durability and when flushing memtables to SSTable data files for persistent storage. The commit log has a different access pattern (read/writes ratio) than the pattern for accessing data from SSTables. This is more important for spinning disks than for SSDs.

SSTables are periodically compacted. Compaction improves performance by merging and rewriting data and discarding old data. However, depending on the type of compaction and size of the compactions, during compaction disk utilization and data directory volume temporarily increases. For this reason, be sure to leave an adequate amount of free disk space available on a node.

Cassandra's data and commit logs should not, under any circumstances, be placed on the drive where the operating system is installed. Ideally, a server would have 3-4 drives or partitions. The root partition, /, the OS partition, can be used as the target for the application. A /data partition should have adequate amounts of storage to accommodate your data. A /logs partition would hold your commit logs (and unless SSD, should be on a different physical disk than the data partition), and a /backup partition would be allocated for backups.

Please refer to http://cassandra.apache.org/doc/latest/operating/hardware.html for explanations on hardware selection.

In order to achieve adequate performance, separate partitions must be created, ideally on separate drives, to avoid i/o contention. We recommend 3 separate block devices (disks). The first block device will contain the operating system as well as a mount for the programs (*/opt/local*). The second block device (preferably SSD) will contain a mount point at /data - this is the device that must have high storage capacity for all of the data. The third block device will contain a mount point at /logs - this device should preferably be SSD but does not need to be of high capacity, since it will only store the commit logs, which are by default limited to 8GB (if using SSD, this can be a partition on the same block device as the data partition). All partitions should be formatted using the XFS file system, and there must not be a swap partition. The /backup partition can be a mount on a shared storage device, and should not be on the same physical drive as the /data partition.

The following is an example of the contents of */etc/fstab* after partitioning, where the partitions were created using LVM (without a mount for the /backup partition).

fstab

```
#
# /etc/fstab
# Created by anaconda on Tue May  2 16:31:05 2017
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/cl_twccentos7-root /                       xfs     defaults        0 0
/dev/mapper/cl_twccentos7-data /data                   xfs     defaults        0 0
/dev/mapper/cl_twccentos7-logs /logs                   xfs     defaults        0 0
/dev/mapper/cl_twccentos7-opt_local /opt/local         xfs     defaults        0 0
```

- Disk 1 will contain the following partitions: */opt/local* (40GB) and / (rest of the drive capacity).
- Disk 2 (the disk with the highest capacity) will contain the */data* partition - as a minimum, we recommend 250GB. Due to the way compactions are handled by Cassandra, in a worst-case scenario up to 50% of headroom may be needed.
- Disk 3 will contain the */logs* partition (40 GB).

The aforementioned partitioning scheme is an example. Internal security protocols in your organization may dictate that other directories not be located in the main partition. During the installation, all applications will be installed in */opt/local*. Cassandra will install by default in */var/lib*. Application logs will be written to */home/twcloud*.

Installing Oracle Java

From the Java version list, please check that the recommended Oracle JVM version is compatible with the TWCloud version you are using. It is not recommended to use OpenJDK. In order to consolidate all of the installed applications in a single location, we will be installing them under */opt/local/java*. T o facilitate deployment, you may deploy using the associated script (**install_java.sh)**. Oracle no longer allows direct download of their JDK, so it must be downloaded offline and placed in the same location as the **install** scripts. The installation script extracts it into the proper location, invokes the alternative command to point the system to this instance (you may need to select it when prompted), and creates entries in **/etc/environment**. Upon completing the installation, issue the following command:

```
java -version
```

You should receive output such as the following:

```
java version "1.8.0_172"
Java(TM) SE Runtime Environment (build 1.8.0_172-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.172-b11, mixed mode)
```

If properly installed, you will see Java identified as Java HotSpot(TM)

install_java_172.sh

```
#!/bin/bash
echo "========================================="
echo "Installing Oracle Java 1.8.0_172"
echo "========================================="
echo ""
echo "  Oracle Java can no longer be dowloaded directly due to new authentication requirements"
echo "  After manually downloading jdk-8u172-linux-x64.tar.gz, copy it to this directory"
echo ""
echo "  Archive downloads available from https://www.oracle.com/java/technologies/javase/javase8-archive-
downloads.html"
echo ""
read -p -"Press any key to continue, Ctl-C to exit ...: " -n1 -s
echo "========================================="
sudo mkdir -p /opt/local/java
sudo tar xzf jdk-8u172-linux-x64.tar.gz -C /opt/local/java
cd /opt/local/java/jdk1.8.0_172/
sudo alternatives --install /usr/bin/java java /opt/local/java/jdk1.8.0_172/bin/java 2
sudo alternatives --config java
sudo alternatives --install /usr/bin/jar jar /opt/local/java/jdk1.8.0_172/bin/jar 2
sudo alternatives --install /usr/bin/javac javac /opt/local/java/jdk1.8.0_172/bin/javac 2
sudo alternatives --set jar /opt/local/java/jdk1.8.0_172/bin/jar
sudo alternatives --set javac /opt/local/java/jdk1.8.0_172/bin/javac
sudo echo 'JAVA_HOME=/opt/local/java/jdk1.8.0_172' > /etc/environment
sudo echo 'JRE_HOME=/opt/local/java/jdk1.8.0_172/jre' >> /etc/environment
sudo chown -R root:root /opt/local/java/jdk1.8.0_172
```

Installing the FlexNet server (lmadmin)

A FlexNet license server is required for Teamwork Cloud to operate. It can be installed on the same system, or on a separate machine. The automated deployment script (**install_flex_centos7.sh**) downloads all required components, deploys the server, creates the systemctl service entry to control it, and creates the necessary firewall rules to allow the required traffic. The firewall rules are created for both the internal and public zones, and the script may require modification depending on which zone the interface is located in. Additionally, if the firewall is not running when the installation script is executed, the rules will not be created. The script creates a user, **lmadmin**, which runs the lmadmin service. The FlexNet server requires the Redhat LSB core files as well as the ld-linux library in order to execute. The script is configured for Centos 7, but can be modified for a different version. In order to identify which LSB Core library is required, the following command can be issued:

```
sudo yum provides /lib/ld-lsb.so.3
```

The application should be installed in */opt/local/FNPLicenseServerManager* (the installer's default location is */opt/FNPLicenseServerManager* - so make sure that you change the location when prompted). All other default values presented by the installer should be accepted.

- After the lmadmin server has been installed it can be started by issuing the command:

  ```
  sudo systemctl start lmadmin
  ```

- To check if the service is running, issue the following command:

  ```
  sudo systemctl status lmadmin
  ```

- If the service failed to start, it is often because the built-in web server cannot resolve the hostname. To check if this is the case, issue the following commands:

```
cd /opt/local/FNPLicenseServerManager/logs
tail web.log
```

You will see output similar to the following:

```
[Tue May 02 18:43:27 2017] [alert] (EAI 2)Name or service not known:
mod_unique_id: unable to find IPv4 address of "yourhostname"
Configuration Failed
```

Where *yourhostname* is the name of the host. If this is the case, you will need to edit the */etc/hosts* file and add an entry so the webserver can resolve the host. The line will be similar to the following:

```
 192.168.130.10  yourhostname
```

install_flex_centos7.sh

```
#!/bin/bash
echo "==============="
echo "Installing wget"
echo "==============="
sudo yum install -y wget
echo "=================="
echo "Installing lmadmin"
echo "=================="
sudo getent group lmadmin >/dev/null || groupadd -r lmadmin
sudo getent passwd lmadmin >/dev/null || useradd -d /home/lmadmin -g lmadmin -m -r lmadmin
sudo yum install -y ld-linux.so.2
LSB=$(yum provides /lib/ld-lsb.so.3 | grep lsb-core | tail -1 | cut -f 1 -d ' ')
sudo yum install -y $LSB
sudo echo "lmadmin ALL=(ALL) NOPASSWD:ALL " >> /etc/sudoers
# If Web GUI to Flex licensing is not a must - lmgrd can be used, can be placed in rc.local to startup on boot
# usage - ./lmgrd -c PATH_TO_KEY_FILE -l PATH_TO_LOG_FILE
# RW rights needed to both files
echo "============================================================"
echo "Getting Linux 32-bit IPv6 version 11.14 from AWS FrontCloud"
echo "============================================================"
wget http://d1g91r27pzl568.cloudfront.net/Cameo_daemon/FlexNet_11_14/ipv6/linux/lnx_32/cameo
chmod +x cameo
echo "======================================="
echo "Getting Linux 32-bit lmgrd version 11.14"
echo "======================================="
wget https://d1oqhepk9od1tu.cloudfront.net/Flex_License_Server_Utilities/v11.14/linux32/lmgrd
chmod +x lmgrd
echo "======================================"
echo "Making flex log file named FlexLog.log"
echo "======================================"
touch FlexLog.log
chmod 664 FlexLog.log
echo "=========================================="
echo "Getting Linux 32-bit lmadmin version 11.14"
echo "=========================================="
wget https://d1oqhepk9od1tu.cloudfront.net/Flex_License_Server_Utilities/v11.14/linux32/lmadmin-i86_lsb-
11_14_0_0.bin
chmod +x lmadmin-i86_lsb-11_14_0_0.bin
echo "========================================="
echo "Executing lmadmin version 11.14 installer"
echo "IMPORTANT: Install into directory /opt/local/FNPLicenseServerManager"
echo ""
echo " Note:  Accept all defaults for script to work properly!!!"
read -p -"Press any key to continue ...: " -n1 -s
echo "========================================="
sudo ./lmadmin-i86_lsb-11_14_0_0.bin
sudo mkdir -p /opt/local/FNPLicenseServerManager/licenses/cameo/
sudo mv cameo /opt/local/FNPLicenseServerManager/licenses/cameo/cameo
```

```
sudo mv lmgrd /opt/local/FNPLicenseServerManager/lmgrd
sudo mv cameo /opt/local/FNPLicenseServerManager/cameo
sudo mv FlexLog.log /opt/local/FNPLicenseServerManager/FlexLog.log
sudo chown -R lmadmin:lmadmin /opt/local/FNPLicenseServerManager/
sudo chmod +x /opt/local/FNPLicenseServerManager/lib*
sudo cp /opt/local/FNPLicenseServerManager/lib* /usr/lib/
echo "======================"
echo "Opening firewall ports"
echo "======================"
FWZONE=$(sudo firewall-cmd --list-all | grep "(active)" | tail -1 | cut -f 1 -d " ")
cat <<EOF | sudo tee /etc/firewalld/services/lmadmin.xml
<?xml version="1.0" encoding="utf-8"?>
<service version="1.0">
    <short>lmadmin</short>
    <description>lmadmin</description>
    <port port="8090" protocol="tcp"/>
    <port port="1101" protocol="tcp"/>
</service>
EOF
sleep 5
sudo firewall-cmd --zone=public --remove-port=8090/tcp --permanent
sudo firewall-cmd --zone=public --remove-port=1101/tcp --permanent
sudo firewall-cmd --zone=public --remove-port=27000-27009/tcp --permanent
sudo firewall-cmd --zone=internal --remove-port=8090/tcp --permanent
sudo firewall-cmd --zone=internal --remove-port=1101/tcp --permanent
sudo firewall-cmd --zone=internal --remove-port=27000-27009/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --add-service=lmadmin --permanent
sudo firewall-cmd --reload
IP_ADDRESS=$(ip route get 1 | awk '{print $NF;exit}')
HOSTNAME=$(hostname)
echo "$IP_ADDRESS     $HOSTNAME" >> /etc/hosts
echo "========================================="
echo "Creating systemd service - lmadmin"
echo "========================================="
sudo echo "[Unit]" > /etc/systemd/system/lmadmin.service
sudo echo "Description=Flexnet License Daemon" >> /etc/systemd/system/lmadmin.service
sudo echo "After=network.target network.service" >> /etc/systemd/system/lmadmin.service
sudo echo "" >> /etc/systemd/system/lmadmin.service
sudo echo "[Service]" >> /etc/systemd/system/lmadmin.service
sudo echo "User=lmadmin" >> /etc/systemd/system/lmadmin.service
sudo echo "WorkingDirectory=/opt/local/FNPLicenseServerManager/" >> /etc/systemd/system/lmadmin.service
sudo echo "ExecStart=/opt/local/FNPLicenseServerManager/lmadmin -allowStopServer yes" >> /etc/systemd/system
/lmadmin.service
sudo echo "Restart=always" >> /etc/systemd/system/lmadmin.service
sudo echo "RestartSec=30" >> /etc/systemd/system/lmadmin.service
sudo echo "Type=forking" >> /etc/systemd/system/lmadmin.service
sudo echo "" >> /etc/systemd/system/lmadmin.service
sudo echo "[Install]" >> /etc/systemd/system/lmadmin.service
sudo echo "WantedBy=multi-user.target" >> /etc/systemd/system/lmadmin.service
sudo echo "" >> /etc/systemd/system/lmadmin.service
sudo chown root:root /etc/systemd/system/lmadmin.service
sudo chmod 755 /etc/systemd/system/lmadmin.service
sudo systemctl daemon-reload
sudo systemctl enable lmadmin.service
echo "========================================="
echo "lmadmin service installation complete"
echo "  usage: systemctl start|stop lmadmin"
echo "========================================="
```

Installing Apache Cassandra 3.11.2

The deployment script for Cassandra removes Datastax Community Edition 2.2.x as well as OpsCenter and the Datastax Agent (which are not compatible with Cassandra 3.x), downloads and installs Cassandra the Cassandra tools from the Apache Software Foundation repository, and creates the necessary firewall rules to allow proper operation both for a single node or a cluster installation. To install, execute the installation script (**install_cassandra_3_11_centos7.sh**).

install_cassandra3_11_centos7.sh

```
#!/bin/bash
echo "========================================="
echo "Installing Apache Cassandra 3.11.x"
echo "========================================="
echo "Removing Datastax Community Edition"
sudo yum remove -y datastax-agent
sudo yum remove -y opscenter
sudo yum remove -y cassandra22-tools
sudo yum remove -y cassandra22
sudo yum remove -y dsc22
sudo rm -f /etc/yum.repos.d/datastax.repo
echo "Creating Apache Cassandra Repository File"
sudo echo "[cassandra]" > /etc/yum.repos.d/cassandra.repo
sudo echo "name=Apache Cassandra" >> /etc/yum.repos.d/cassandra.repo
sudo echo "baseurl=http://www.apache.org/dist/cassandra/redhat/311x/" >> /etc/yum.repos.d/cassandra.repo
sudo echo "gpgcheck=1" >> /etc/yum.repos.d/cassandra.repo
sudo echo "repo_gpgcheck=1" >> /etc/yum.repos.d/cassandra.repo
sudo echo "gpgkey=https://www.apache.org/dist/cassandra/KEYS" >> /etc/yum.repos.d/cassandra.repo
sudo yum install -y epel-release
sudo yum install -y cassandra
sudo yum install -y cassandra-tools
sudo yum install -y jemalloc
sudo chkconfig --add cassandra
sudo chkconfig cassandra on
echo "========================"
echo "Configuring firewall"
echo "========================"
FWZONE=$(sudo firewall-cmd --list-all | grep "(active)" | tail -1 | cut -f 1 -d " ")
echo "Discovered firewall zone $FWZONE"
cat <<EOF | sudo tee /etc/firewalld/services/cassandra.xml
<?xml version="1.0" encoding="utf-8"?>
<service version="1.0">
    <short>cassandra</short>
    <description>cassandra</description>
    <port port="7000" protocol="tcp"/>
    <port port="7001" protocol="tcp"/>
        <port port="9042" protocol="tcp"/>
        <port port="9160" protocol="tcp"/>
        <port port="9142" protocol="tcp"/>
</service>
EOF
sleep 5
sudo firewall-cmd --zone=$FWZONE --remove-port=7000/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --remove-port=7001/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --remove-port=7199/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --remove-port=9042/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --remove-port=9160/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --remove-port=9142/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --add-service=cassandra --permanent

sudo firewall-cmd --reload
echo "====================================================="
echo "Changing ownership of data and commit log directories"
echo "====================================================="
sudo mkdir /data
sudo mkdir /logs
sudo chown cassandra:cassandra /data
sudo chown cassandra:cassandra /logs
echo "====================================================="
echo "Making configuration file changes"
echo "====================================================="
IP_ADDRESS=$(ip route get 1 | awk '{print $NF;exit}')
HOSTNAME=$(hostname)
sudo cp /etc/cassandra/default.conf/cassandra.yaml /etc/cassandra/default.conf/cassandra.yaml.backup
sudo cp /etc/cassandra/default.conf/cassandra.yaml ./cassandra.yaml.template
sudo sed -i "s/ - seeds: \"127.0.0.1\"/ - seeds: \"$IP_ADDRESS\"/g" cassandra.yaml.template
sudo sed -i "s/listen_address:.*/listen_address: $IP_ADDRESS/g" cassandra.yaml.template
sudo sed -i "s/# broadcast_rpc_address:.*/broadcast_rpc_address: $IP_ADDRESS/g" cassandra.yaml.template
sudo sed -i "s/broadcast_rpc_address:.*/broadcast_rpc_address: $IP_ADDRESS/g" cassandra.yaml.template
sudo sed -i "s/# commitlog_total_space_in_mb:.*/commitlog_total_space_in_mb: 8192/g" cassandra.yaml.template
sudo sed -i "s/commitlog_total_space_in_mb:.*/commitlog_total_space_in_mb: 8192/g" cassandra.yaml.template
```

```
sudo sed -i "s/^rpc_address:.*/rpc_address: 0.0.0.0/g" cassandra.yaml.template
sudo sed -i "s/start_rpc:.*/start_rpc: true/g" cassandra.yaml.template
sudo sed -i "s/thrift_framed_transport_size_in_mb:.*/thrift_framed_transport_size_in_mb: 100/g" cassandra.yaml.
template
sudo sed -i "s/commitlog_segment_size_in_mb:.*/commitlog_segment_size_in_mb: 192/g" cassandra.yaml.template
sudo sed -i "s/read_request_timeout_in_ms:.*/read_request_timeout_in_ms: 1800000/g" cassandra.yaml.template
sudo sed -i "s/range_request_timeout_in_ms:.*/range_request_timeout_in_ms: 1800000/g" cassandra.yaml.template
sudo sed -i "s/write_request_timeout_in_ms:.*/write_request_timeout_in_ms: 1800000/g" cassandra.yaml.template
sudo sed -i "s/cas_contention_timeout_in_ms:.*/cas_contention_timeout_in_ms: 1000/g" cassandra.yaml.template
sudo sed -i "s/truncate_request_timeout_in_ms:.*/truncate_request_timeout_in_ms: 1800000/g" cassandra.yaml.
template
sudo sed -i "s/request_timeout_in_ms:.*/request_timeout_in_ms: 1800000/g" cassandra.yaml.template
sudo sed -i "s/batch_size_warn_threshold_in_kb:.*/batch_size_warn_threshold_in_kb: 3000/g" cassandra.yaml.
template
sudo sed -i "s/batch_size_fail_threshold_in_kb:.*/batch_size_fail_threshold_in_kb: 5000/g" cassandra.yaml.
template
sudo sed -i '/data_file_directories:.*/!b;n;c\ \ \ \ - \/data\/data' cassandra.yaml.template
sudo sed -i "s/hints_directory:.*/hints_directory: \/data\/hints/g" cassandra.yaml.template
sudo sed -i "s/commitlog_directory:.*/commitlog_directory: \/logs\/commitlog/g" cassandra.yaml.template
sudo sed -i "s/saved_caches_directory:.*/saved_caches_directory: \/data\/saved_caches/g" cassandra.yaml.
template
sudo \cp -fR ./cassandra.yaml.template /etc/cassandra/default.conf/cassandra.yaml
```

Upon completion of the installation, issue the following command to ensure that Cassandra starts automatically upon system reboot.

```
chkconfig --add cassandra
```

Now, proceed to edit */etc/cassandra/default.conf/cassandra.yaml:*

```
sudo nano /etc/cassandra/default.conf/cassandra.yaml
```

⚠ The script above makes all of the changes to the configuration files stated below. However, please verify that all of them have been made.

The first items we will be editing relate to the IP address of the Cassandra node and communication settings. In our diagram above, this IP address is **192.168.130.10**. You will need to search for 3 keys in the configuration file and modify them accordingly. The seeds parameter is a comma-delimited list containing all of the seeds in the Cassandra cluster. Since our cluster consists of only a single node, it contains only one entry - our IP address. The other 2 parameters contain the IP address on which Cassandra listens for connections and the IP address to broadcast to other Cassandra nodes in the cluster. The **broadcast_rpc_address** may be commented out using a # character. If so, remove the "#" and make sure there are no leading spaces.

Additionally, we need to set **rpc_address** to **0.0.0.0** (meaning, it will listen to rpc requests on all interfaces), and **start_rpc** to **true** (so it will process rpc requests).

```
seeds: "192.168.130.10"
listen_address: 192.168.130.10
broadcast_rpc_address: 192.168.130.10
rpc_address: 0.0.0.0
start_rpc: true
```

The next set of parameters controls thresholds to ensure that the data being sent is processed properly.

```
thrift_framed_transport_size_in_mb: 100
commitlog_segment_size_in_mb: 192
read_request_timeout_in_ms: 1800000
range_request_timeout_in_ms: 1800000
write_request_timeout_in_ms: 1800000
cas_contention_timeout_in_ms: 1000
truncate_request_timeout_in_ms: 1800000
request_timeout_in_ms: 1800000
batch_size_warn_threshold_in_kb: 3000
batch_size_fail_threshold_in_kb: 5000
```

If you have installed your commit log in its own partition, the default commit log size will be the lesser of ¼ of the partition size of 8GB. In order to ensure that the recommended 8GB is used, you must uncomment the **commitlog_total_space_in_mb**, such that it will show as below. However, if you are uncommenting this value, please ensure that the partition has enough space to accommodate an 8GB commit log.

```
commitlog_total_space_in_mb: 8192
```

The next step is to point the data to the new locations. There are 4 entries that will be modified: **data_file_directories**, **commitlog_directory**, **hints_directory**, and **saved_caches_directory**. Search for these keys and edit them as follows:

```
data_file_directories:
- /data/data
commitlog_directory: /logs/commitlog
hints_directory: /data/hints
saved_caches_directory: /data/saved_caches
```

After you have made these changes, save the **cassandra.yaml** file. Now, start the related services, as follows:

```
sudo service cassandra start
```

Now, proceed to check if Cassandra is running. To do this, issue the following command:

```
nodetool status
```

If the service is running, you will receive output such as below:

```
Datacenter: datacenter1
=======================
Status=Up/Down
|/ State=Normal/Leaving/Joining/Moving
--  Address    Load       Tokens   Owns (effective)  Host ID                               Rack
UN  127.0.0.1  128.4 KB   256      100.0%            ea3f99eb-c4ad-4d13-95a1-80aec71b750f  rack1
```

If the service is fully operational, the first 2 characters on the last line will state "**UN**", indicating the node's status is **U**p, and its state is **N**ormal.

Tuning Linux for Cassandra Performance

There are multiple tunings that can be performed on Linux to improve the performance of Cassandra. The first step is to configure the TCP settings by adding the following tuning parameters to */etc/sysctl.conf* file:

```
net.ipv4.tcp_keepalive_time=60
net.ipv4.tcp_keepalive_probes=3
net.ipv4.tcp_keepalive_intvl=10
net.core.rmem_max=16777216
net.core.wmem_max=16777216
net.core.rmem_default=16777216
net.core.wmem_default=16777216
net.core.optmem_max=40960
net.ipv4.tcp_rmem=4096 87380 16777216
net.ipv4.tcp_wmem=4096 65536 16777216
vm.max_map_count=1048575
```

To apply the setting without requiring a reboot issue the command:

```
# sysctl -p
```

For a full list of steps to take to tune Linux, go to:

https://docs.datastax.com/en/dse/5.1/dse-admin/datastax_enterprise/config/configRecommendedSettings.html

Installing Teamwork Cloud

Initial Installation

The deployment script for TWCloud (**install_twc190_centos7.sh**) creates a TWCloud user, under which the service will run, downloads all of the necessary files, and executes the installer.

install_twc190_centos7.sh

```
#!/bin/bash
echo "========================================="
echo "Installing Teamwork Cloud 19.0"
echo "========================================="
echo "Installing unzip"
sudo yum install -y unzip
echo "Creating twcloud group and user"
sudo getent group twcloud >/dev/null || groupadd -r twcloud
sudo getent passwd twcloud >/dev/null || useradd -d /home/twcloud -g twcloud -m -r twcloud
echo ""
echo "IMPORTANT: Install into directory /opt/local/TeamworkCloud"
echo "           When prompted for user to run service, use twcloud"
read -p -"Press any key to continue ...: " -n1 -s
sudo wget http://download1.nomagic.com/twcloud190/twcloud_190_installer_linux64.bin
sudo chmod +x twcloud_190_installer_linux64.bin
sudo ./twcloud_190_installer_linux64.bin
sudo chown -R twcloud:twcloud /opt/local/TeamworkCloud/
IP_ADDRESS=$(ip route get 1 | awk '{print $NF;exit}')
sudo sed -i "s/\"localhost\"/\"$IP_ADDRESS\"/" /opt/local/TeamworkCloud/configuration/application.conf
sudo sed -i "s/localhost/$IP_ADDRESS/" /opt/local/TeamworkCloud/AuthServer/config/authserver.properties
echo "========================"
echo "Configuring firewall"
echo "========================"
FWZONE=$(sudo firewall-cmd --list-all | grep "(active)" | tail -1 | cut -f 1 -d " ")
echo "Discovered firewall zone $FWZONE"
cat <<EOF | sudo tee /etc/firewalld/services/twcloud.xml
<?xml version="1.0" encoding="utf-8"?>
<service version="1.0">
    <short>twcloud</short>
    <description>twcloud</description>
    <port port="8111" protocol="tcp"/>
    <port port="3579" protocol="tcp"/>
        <port port="8555" protocol="tcp"/>
        <port port="2552" protocol="tcp"/>
        <port port="2468" protocol="tcp"/>
</service>
EOF
sleep 5
sudo firewall-cmd --zone=$FWZONE --remove-port=8111/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --remove-port=3579/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --remove-port=8555/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --remove-port=2552/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --remove-port=2468/tcp --permanent
sudo firewall-cmd --zone=$FWZONE --add-service=twcloud --permanent
sudo firewall-cmd --reload
```

When you are installing TWCloud

---

1. Press **ENTER** until the licensing agreement is accepted.
2. Configure the machine IP - enter the local IP address of the machine (i.e. 192.168.130.10).
3. Configure the cluster seed node IP - enter the local IP address of the machine (i.e. 19.168.130.10).
4. Configure the TWCloud service owner - enter **twcloud**.
5. Configure **JAVA_HOME** - it should display */opt/local/java/jdk1.8.0_172* - accept this default.
6. Choose **Install Folder** - */opt/local/TeamworkCloud*.

Next, TWCloud's Pre-Installation Summary will appear. It should look as follows:

```
===============================================================================
Pre-Installation Summary
------------------------
Please Review the Following Before Continuing:

Product Name:
    Teamwork Cloud

Install Folder:
    /opt/local/TeamworkCloud

Machine ip:
    "192.168.130.10"

Seed node ip:
    "192.168.130.10"

JAVA_HOME:
    "/opt/local/java/jdk1.8.0_172"

Disk Space Information (for Installation Target):
    Required:  395,614,661 Bytes
    Available: 31,608,475,648 Bytes
```

Anywhere where 192.168.130.10 is displayed, you must replace it with the IP address of your machine.

Post-Install Configuration

The installer has created the preliminary configuration of TWCloud. There are a few more changes which we need to make to the various configurations files in order for TWCloud to be fully functional.

1. */opt/local/TeamworkCloud/configuration/application.conf* **-** the configuration file for the TWCloud service**.**

   If TWCloud is installed behind a proxy or firewall with NAT, upon the initial connection the MagicDraw client must know the external IP address to which it must connect. Search for **server-broadcast-host**, and enter the public IP address instead of the local IP address.

   We now need to point TWCloud to the Cassandra database. Search for **seeds =**, which is located in the connection section. Edit the value inside the quotes to point to the **listen_address** you set in **cassandra.yaml** (i.e. seeds = ["192.168.130.10"]).

   A default password has been entered in the configuration file for its communication with the authorization server. It is recommended that it be changed. To do so, search for **CHANGE_ME,** which is associated with the field **pswd**, and replace it with a password of your choosing.

2. */opt/local/TeamworkCloud/AuthServer/config/authserver.properties* **-** the configuration file for Authorization service**.**

- **server.public.host**=public IP address (same as server-broadcast-host in **application.conf**). If you are accessing the server via an FQDN, use it instead of the IP address.
- **twc.server.host**=local IP address.
- If you changed the **pswd** field in */opt/local/TeamworkCloud/configuration/application.conf* from the default, you must modify this file accordingly. Search for **authentication.client.secret**. Remove the leading **#** (to uncomment the directive), and replace the **CHANGE_ME** value with the same value as that in **application.conf**.
- If you are accessing the server by FQDN, you must edit the property **authentication.redirect.uri.whitelist** by adding an entry to whitelist the FQDN. For example: **authentication.redirect.uri.whitelist**=*https://192.168.130.10:8111/twcloud_admin/,https://FQDN:8111/twcloud_admin/, https://md_redirect*

To start the authserver service, execute the following command:

```
sudo service authserver start
```

To start the Teamwork Cloud service, execute the command:

```
sudo service twcloud-svc start
```

To ensure the services start on reboot, execute the following commands:

```
sudo chkconfig twcloud-svc on
sudo chkconfig authserver on
```

Additional information which may affect installations in restricted environments

Log Files

TWCloud executes under the TWCloud user, and by default will store log files under this user's profile (*/home/twcloud*). There are 2 configuration files that control the location of these log files:

- /opt/local/TeamworkCloud/configuration/logback.xml controls the location of the Teamwork Cloud log files, whereas
- /opt/local/TeamworkCloud/Authserver/config/logback-spring.xml controls the location of the Authserver log files.

/opt/local/TeamworkCloud/configuration/logback.xml

In this file, there are settings for 2 log files that must be edited.

```
<appender name="SERVER-FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
                <file>${user.home}/.twcloud/19.0/server.log</file>
                <rollingPolicy class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy">
                        <fileNamePattern>${user.home}/.twcloud/19.0/server.%i.log.zip</fileNamePattern>
                        <minIndex>1</minIndex>
                        <maxIndex>1000</maxIndex>
                </rollingPolicy>

                <triggeringPolicy class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">
                        <maxFileSize>20MB</maxFileSize>
                </triggeringPolicy>
                <encoder>
                  <pattern>%-5.5level %date{YYYY-MM-dd HH:mm:ss.SSS} %message [%logger{200}, %thread{10}]%n<
/pattern>
                </encoder>
        </appender>

        <appender name="SECURITY-FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
                <file>${user.home}/.twcloud/19.0/security.log</file>
                <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
                <!-- daily rollover -->
                <fileNamePattern>${user.home}/.twcloud/19.0/security.%d{yyyy-MM-dd}.log</fileNamePattern>

                <!-- keep 365 days' worth of history -->
                <!-- maxHistory>365</maxHistory -->
        </rollingPolicy>
                <encoder>
                  <pattern>%-5.5level %date{YYYY-MM-dd HH:mm:ss.SSS} %message [%logger{200}, %thread{10}]%n<
/pattern>
                </encoder>
        </appender>
```

In each section, there are 2 settings that must be modified: **file** and **fileNamePattern**. The first setting (file) controls the absolute path to the latest log file. The second setting (**fileNamePattern**) controls the naming convention for the archiving of the log files. In most cases, it will suffice to replace the **${user.home}** token with a different location, but you must ensure that the TWCloud user has ownership of the target directories.

/opt/local/TeamworkCloud/Authserver/config/logback-spring.xml

This file contains one section which must be modified.

```
  <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
               <file>${user.home}/.authserver/19.0./authserver.log</file>
               <rollingPolicy class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy">
                       <fileNamePattern>${user.home}/.authserver/19.0./authserver.%i.log.zip</fileNamePattern>
                       <minIndex>1</minIndex>
                       <maxIndex>10</maxIndex>
               </rollingPolicy>

               <triggeringPolicy class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">
                       <maxFileSize>30MB</maxFileSize>
               </triggeringPolicy>
               <encoder>
                 <pattern>%-5.5level %date{YYYY-MM-dd HH:mm:ss.SSS} %message [%logger{0}, %thread{10}]%n<
/pattern>
               </encoder>
       </appender>
```

The same changes and permissions apply to the changes to this file as to those for /opt/local/TeamworkCloud/configuration/logback.xml.

Files installed on system locations

Daemon control files	Environment files	Cassandra installation
• /etc/init.d/authserver • /etc/init.d/cassandra • /etc/init.d/twcloud-svc • /etc/system/system/lmadmin.service	• /etc/cassandra/* • /etc/twcloud/*	The script will place the data files in /data/data and commit logs in /logs/commitlog

Frequently Asked Questions

I am receiving an error when trying to access the Teamwork Cloud Admin Console, before being prompted for user credentials.

This is usually caused by the authentication server not running, and depending on the browser may include a page beginning with:

```
{ "issystemerror": true, "data": "Class: org.springframework.web.client.ResourceAccessException
```

- To check if the authentication server is running, issue the command **sudo service authserver status**.
- If it states it is stopped, start it via the command **sudo service authserver start**.
- If it is running and you are receiving a browser window requesting you contact the system administrator, the cause for this may lie with the **authentication.redirect.uri.whitelist** field in **authserver.properties**.
- If you are accessing via the FQDN, and the UEL is showing the FQDN of the machine, please add it in the form of https://FQDN:8111/twcloud_admin/ and restart the authserver service via the command **sudo service authserver restart**.

I am unable to access the Teamwork Cloud Admin Console.

First, let's make sure that the service is running. This is done via the command **sudo service twcloud-svc status**. Also, make sure that the authserver service is running, via the command **sudo service authserver status**. If the services are running, the result of the command will be:

```
Running [PID]
```

Where PID is a number representing the process ID of the service. If the services are running, let's ensure that they are listening on the expected ports. This is done by issuing the following commands:

```
netstat -anp | grep tcp | grep 8111 | grep LISTEN


and

netstat -anp | grep tcp | grep 8555 | grep LISTEN



The result should be something along the lines of

 [root@twccentos7 ~]# netstat -anp | grep tcp | grep 8111 | grep LISTEN
 tcp6       0      0 :::8111                 :::*                    LISTEN      28294/java
 [root@twccentos7 ~]# netstat -anp | grep tcp | grep 8555 | grep LISTEN
 tcp6       0      0 :::8555                 :::*                    LISTEN      28466/java
```

If you get a **command not found** message when executing netstat, this means it is not installed on your computer. To install it, execute the command:

```
sudo yum install net-tools
```

and then retry once the package has been installed.

If the ports are listening, then the issue relates to traffic not being allowed into the computer on these ports. This may be caused either by the Linux firewall or by an external firewall. To temporarily turn off the Linux firewall, issue the following command:

```
sudo systemctl stop firewalld
```

If you can connect when the firewall is stopped, then you must check the firewall rules to ensure you are allowing traffic on both ports **8111** and **8555**. To restart the Linux firewall, issue the following command:

```
sudo systemctl start firewalld
```

If you were unable to connect, please contact your IT system administrators to ensure that they are not blocking traffic on these ports to the computer.