5.0 Zero Footprint Tomcat Instances

Introduction

This is a method of installing and running Tomcat in a way that is portable and part of the BonsaiFramework 0FS approach. Tar (zip) up the directory and move your entire application server or duplicate it with a copy command.

This work comes from a corporate environment where isolation, control and ease of upgrade with a fallback are very important. Also, allows for fast horizontal or vertical scaling where multiple Tomcat instances are run on the same machine.

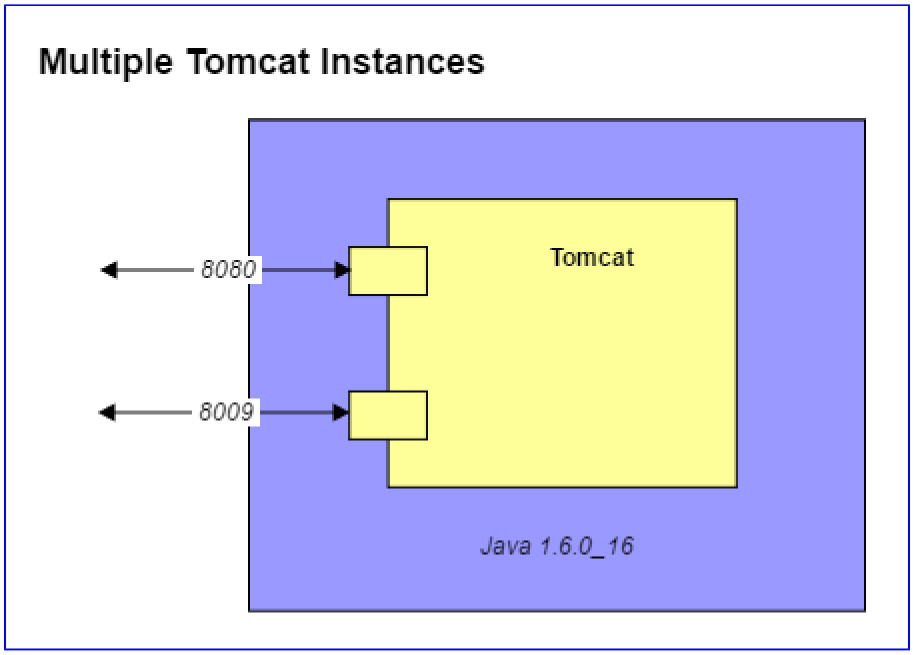
Also this also sets your application to moving towards cloud containor implementations like LXC or Docker.

The key concept here is that we setup Tomcat and Java as an isolated package. There is no install. As long as you have matching serveradmin GUID's across systems you may transport your package with **tar.**

Software Stack Selection,

- Java SE Runtime Environment (JRE) = 6 Update 16
- Tomcat = 6.0.20

Here is a diagram of what we will be building,



Shortcut

Everything is all packaged up at http://www.bonsaiframework.com/downloads/0fs-tomcat-linux/,

Package Version	Comments	Next Step Version
v0.1	Basic Tomcat with Java embedded. This should be run only as **serveradmin** per the Bonsai approach to server management.	Ongoing Maintenance with new versions of Java and documentation of who's using it and where.
v0.2	Harden Tomcat and maintain a log.	... will be done as part of the Bonsai group work ...
v0.3	Rolling Logs	Fix catalina.outlog rotation (determine best route) and petition to get this fixed. Kevin looking into this.

Install an unzip using a staff account. The use of **sudo** is necessary to retain permission,

```
wget
http://www.bonsaiframework.com/downloads/0fs-tomcat-linux/0fs-tomcat.tar.gz
# this symbolically points to the latest 64-bit version

# NOTE - to work properly ensure serveradmin is setup per Bonsai
instructions!
#

sudo tar -xvpf 0fs-tomcat.tar.gz # sudo will ensure the permissions and
users are kept
sudo mv /0fs-tomcat/ /opt/


# Optionally if you want to follow the Bonsaiframework convention and also
let users belonging to "staff" group to view files and restrict others,
cd /opt/
sudo chown -R serveradmin:staff ./0fs-tomcat/
sudo chmod -R o-rwx ./0fs-tomcat/
sudo chown -R serveradmin.staff ./0fs-tomcat/
```

You are now ready to go and start Tomcat as serveradmin,

```
su - serveradmin
cd /opt/0fs-tomcat/bin/
./startup.sh
```

Verify Tomcat is running,

```
ps -ef | grep java
```

To stop Tomcat,

```
su - serveradmin # if not already serveradmin
cd /opt/0fs-tomcat/bin/
./shutdown.sh
```

That's it.

Tomcat and serveradmin

I will re-iterated that you should run Tomcat or any public facing service for that matter as serveradmin. As noted in account creation, this is for security reasons. In the event that Tomcat is somehow attacked, the compromise would be limited to serveradmin which has reduced limited priviledges.

Further to this, using a central account ensures consistency.

For audit purposes, make sure to log in with your own staff id first and then sudo into serveradmin for managing Tomcat. In true Cloud world where everything works as a recipe, use your recipes to make Tomcat adjustments.

Directory Structure

The directory structure will be as follows,

/opt/ - root directory for all our applications and services (anything custom if possible should be here)

/opt/0fs-tomcat - directory for Tomcat running on port 8180
/opt/0fs-tomcat/java - directory we will place java for Tomcat

First step, create the apache directory under opt,

```
su - bhitch # We need a staff user who can sudo
cd /opt
sudo mkdir apache
sudo chown -R serveradmin:staff ./apache # Make sure serveradmin can use
the folder.
```

Manually Setup JRE

Go to the Java website, choose the 64-bit JRE (Java Runtime Engine). Download the **.tar.gz** file and upload the file to the serveradmin home folder. For older versions of Java go to the Java Archives site.

Use the account that will be launching the Java process. In this example it will be serveradmin,

```
su - serveradmin # If you are not already serveradmin
cd ~
wget --no-check-certificate --no-cookies --header "Cookie:
oraclelicense=accept-securebackup-cookie"
http://download.oracle.com/otn/java/jdk/7u7-b10/jre-7u7-linux-x64.tar.gz
```

The steps for JRE and JDK are both the same. Here is an example of a **JRE** setup,

```
su - serveradmin # If you are not already serveradmin
cd ~ # Switch to the serveradmin home directory
tar -xvpf jre-7u7-linux-x64.tar.gz
```

The result will be an uncompressed jre directory using the same name as the package. In this example the folder name would be, **jre-7u7-linux-x64**.

If you plan to use multiple versions of Java, we recommend keeping the folder name with the version number information and using symbolic links. If you are only using one version of Java, then simply rename the folder.

For the server example, we will rename the folder,

```
mv jre-7u7-linux-x64 java
```

You may be interested in how to Zero Footprint Java on Windows.

Leave the setup Java folder alone for now. It will be moved into the Tomcat folder as part of the Tomcat setup.

Manually Setup Tomcat and Package Java In

By manually setting up Tomcat there is much more control and you can run multiple tomcat instances. Download tomcat. The tar.gz file is used because permissions are already setup such as execute for **startup.sh**. A zip file will lose the permissions.

Go to the Apache Tomcat download site.

```
su - serveradmin
cd ~
tar -xvpf apache-tomcat-6.0.20.tar.gz # All the permissions will be kept
mv ./apache-tomcat-6.0.20/ ./0fs-tomcat/
exit # Switch back to your staff account
```

Next move the extracted Java folder into your tomcat folder,

```
mv ./java/ ./0fs-tomcat/
```

Log in as your staff account which has sudo access to perform the actual move to /opt/

```
cd /home/serveradmin
sudo mv ./0fs-tomcat/ /opt/apache/
```

Only Allow serveradmin to Run Tomcat

Setting up and running Tomcat with **serveradmin** has the advantage that you can manage the Application server without having to go into root. It's also makes things much safer if somebody breaks into Tomcat.

We want to ensure that only serveradmin starts Tomcat to prevent any issues with permissions. For example, once you start Tomcat as root you may find log files spawned from during startup can no longer be managed by serveradmin.

First login as serveradmin. All modification to Tomcat and running of tomcat will happen as serveradmin.

```
su - serveradmin
```

Modify Tomcat's **/opt/0fs-tomcat/bin/startup.sh** and **opt/apache/0fs-tomcat/bin/shutdown.sh** to only allow serveradmin to start and stop Tomcat.

Do this by adding the block of lines marked with # Bonsaiframework as shown below,

```
#!/bin/sh

# Bonsaiframework - Modification Start
# ---------------------------------------
if [ "$LOGNAME" != "serveradmin" ]; then
echo "This service should only managed with the user serveradmin"
exit 1
fi
# ---------------------------------------
# Bonsaiframework - Modification End

# Licensed to the Apache Software Foundation (ASF) under one or more
```

Challenge to a reader after learning this. Make the above edit automated with copy and paste with sed and wget like my other tutorials.

Bind Tomcat to Java Using setenv.sh

Tomcat can be run with a separate version of JRE or JDK that is not the default system version. To do so, you will have to explicitly set the JRE_HOME variable. The JAVA_HOME variable is also configured as some applications will want to make use of this variable instead.

Tomcat has a facility for this via a file called **setenv.sh** which actually does not exist by default. As soon as you create the file, Tomcat will run **set env.sh** as part of its startup.

First, let's run the Tomcat diagnostics to see what happens. Because we have no default java and Tomcat does not know about it we should expect a message that we need Java.

```
su - serveradmin # If you are not already serveradmin.
cd /opt/0fs-tomcat/bin
./version.sh
Neither the JAVA_HOME nor the JRE_HOME environment variable is defined
At least one of these environment variable is needed to run this program
```

So now let's create the setenv.sh file. As **serveradmin** create **/opt/0fs-tomcat/bin/setenv.sh** using your favourite editor. Your file contents will look like this,

```
#!/bin/sh

# Bonsaiframework - Modification Start
# ---------------------------------------
JRE_HOME="$CATALINA_HOME"/java
JAVA_HOME="$CATALINA_HOME"/java
# ---------------------------------------
# Bonsaiframework - Modification End
```

The $CATALINA_HOME is a script variable that is established by Tomcat to set the directory it is running from.

Now exeucting version.sh works,

```
./version.sh
Using CATALINA_BASE:   /opt/0fs-tomcat
Using CATALINA_HOME:   /opt/0fs-tomcat
Using CATALINA_TMPDIR: /opt/0fs-tomcat/temp
Using JRE_HOME:       /opt/0fs-tomcat/java
Server version: Apache Tomcat/6.0.20
Server built:   May 14 2009 01:13:50
Server number:  6.0.20.0
OS Name:        Linux
OS Version:     2.6.31-302-rs
Architecture:   amd64
JVM Version:    1.6.0_16-b01
JVM Vendor:     Sun Microsystems Inc.
```

Using this method, you can have different Tomcat instances running different versions of Java and control when you want to move between Java versions.

Secure Directory

Finally, because this is a multi-user machine, we secure tomcat from other users and processes. The only users should be serveradmin for read and write and staff for read to debug. All others should not even be able to go into the directory.

Change the permissions,

```
cd /opt/
sudo chown -R serveradmin:staff ./0fs-tomcat/ # Only serveradmin and staff
can manage files.
sudo chmod o-rwx ./0fs-tomcat/ # Remove "other" from getting any access.
sudo chown -R serveradmin.staff # Ensure new files created follow the
Directory's setgid.
```

However, this is not enough. Any new files created in those directories will change to what the particular user has set in terms of that user's groups. This also includes the process user serveradmin. The log files created when the process starts will belong to serveradmin user and serveradmin **group** - which we don't want. So to fix this we tell the directory to set the **setgid** bit,

```
cd /opt/
sudo chown -R serveradmin.staff # Ensure new files created follow the
Directory's setgid.
```

Verify Process is Running

Finally startup your Tomcat instances and verify that they are listening,

```
su - serveradmin

cd /opt/0fs-tomcat/bin/
./startup.sh

netstat -an | grep LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp6       0      0 127.0.0.1:8105          :::*                    LISTEN
tcp6       0      0 :::8009                 :::*                    LISTEN
tcp6       0      0 :::8080                 :::*                    LISTEN
tcp6       0      0 :::80                   :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
unix  2      [ ACC ]     STREAM     LISTENING     7376
@/com/ubuntu/upstart
unix  2      [ ACC ]     STREAM     LISTENING     11434
/var/run/fail2ban/fail2ban.sock
unix  2      [ ACC ]     STREAM     LISTENING     21228
/var/run/apache2/cgisock.4973
```

Notice that here we have started Tomcat and it is listening on port 8009 and 8080.

Tomcat by default will have a sample application installed and running.

> If you have a firewall setup do not forget to open port 8080 for testing and then close them afterwards if you plan to front with the Apache Web Server.

If your server has a web browser you can load the examples page using http://localhost:8080/examples/. From another computer you can see the examples application by browsing to, http://www.krypton.com:8080/examples/ where if www.krypton.com is not a real dns, use the server's IP address or add a host file entry to your client system.

Additional Layers

In my experience all my real world systems do not need any more layers to Tomcat. However, there are some odd scenarios which are covered here.

Automatic Startup and Shutdown of Tomcat

Not recommend until you proper monitoring in place. If you system reboots you want to know about it.

Even then, most Enterprise simply do not have automatic startup setup. The closest I have personally seen is a delayed startup. The reason being that the appliction server itself may hang the entire machine.

In a Cloud world this is more self-contained so this section will be written out in the future.

> This section is still to be written.

Setup SSL on Tomcat

For testing purposes or if the only thing you want to do is encrypt the channel of communication you can Setup a Self-Signed Certificate for Tomcat.

Otherwise, Setup of a Real SSL Certificate for Tomcat is very similar but with a few extra steps.

The more enterprise solution is to front Tomcat with Apache and setup SSL on Apache.

Make Your Own 0FS TAR Package

Once you are happy with your setup you can make your own package. Using your staff account,

```
cd /opt/
sudo tar -czvf 0fs-tomcat.tar.gz ./0fs-tomcat/ # Don't change the parameter
orders.
```

Notice the use of **sudo** to run the **tar** command. This ensures that proper ownership and permissions will carry over to the new system.

Before unpacking to the target system, ensure the users, groups and GUID match following the Bonsai Standards.

References

http://wiki.apache.org/tomcat/FAQ/CharacterEncoding#Q9 - still to finish reading

http://confluence.atlassian.com/display/DOC/Configuring+Tomcat%27s+URI+encoding?focusedCommentId=231343170&#comment-231343170 - Confluence related notes