How to build and secure Apache Tomcat on Linux.

At work recently, one of my colleagues asked for help setting up and securing Tomcat in order to test Jama Contour (a requirements capturing software that is deployed from Tomcat). This paper is my documented approach to how I modified and configured Tomcat in order to secure the application. I'll try and do another paper later on securing MySql, at this point I want to stay focused on Tomcat. This paper assumes the user is performing a clean install on a fresh system.

Target Platform:	Virtual Machine in VMware Workstation 6.5.	
CPUs:	2 CPUs allocated	
Memory:	2 GB allocated	
Hard Drive:	30 GB allocated for vmdk	
Software used:		
OS:	CentOS 5.5	http://www.microsoft.com
Applications:	Apache Tomcat 6	http://tomcat.apache.org
	Java 6	http://www.oracle.com

OS Install

Followed default install from vendor (literally, just hit next, next, next and then Finish).

Java 6 Install

Download Java from: http://www.oracle.com/technetwork/java/javase/downloads/jdk-6u26-download-400750.html

The directory /opt already existed on my system, if it doesn't exist for you, create it with these commands:

```
su - root
mkdir /opt
```

```
chmod 0755 /opt
```

Copy Java to /opt

```
su - root
cp <your_download_directory>/jdk-6u26-linux-i586.bin /opt
```

Change directory to /opt, extract the contents and create a sym link to the new directory:

```
cd /opt
umask 0022
chmod  750  jdk-6u26-linux-i586.bin
./jdk-6u26-linux-i586.bin
ln -s jdk1.6.0_26 java
rm  -f  jdk-6u26-linux-i586.bin
```

Base install of Apache Tomcat

Download Tomcat 6 from: http://tomcat.apache.org/download-60.cgi

I chose the tar.gz file and checked the file against the MD5 hash (optional).

```
su – root
cd /tmp/downloads/
md5sum apache-tomcat-6.0.32.tar.gz
  <output [your results will differ most likely]>
  928a960268adf610a7d6fe5c4fcd0b20  apache-tomcat-6.0.32.tar.gz
  </output>
```

Copy tomcat to /opt

```
su - root
cp <your_download_directory>/apache-tomcat-6.0.32.tar.gz /opt
```

Change directory to /opt, extract the contents and create a sym link to the new directory:

```
cd /opt
gzip  -dc  apache-tomcat-6.0.32.tar.gz  |  tar  -xvf   -
ln  -s  apache-tomcat-6.0.32  tomcat
rm  -f  apache-tomcat-6.0.32.tar.gz
```

Create a group and user for Tomcat:

```
# groupadd -g 555 tomcat
# useradd -d /home/tomcat -g 555 -n -s /bin/bash -u 555 tomcat
# passwd tomcat
  <set the password to something very complex, e.g.
   First name of girlfriend+4 digits+4 special characters
   Uppercase the 2nd and 4th position of her name, not the first or last
   Characters, e.g. -- jUaNita1975,,,,>
```

File System modifications

Change ownership of Tomcat to new tomcat user and group:

```
If not the root user, then:
su – root
Then:
cd /opt
chown  -R  tomcat:tomcat  apache-tomcat-6.0.32
```

Environment Variables

As the Tomcat User (assuming you are root, switch to Tomcat), add the following aliases to your shell:

```
su – tomcat
printf '\nexport JAVA_HOME=/opt/java\n' >> .bash_profile
printf '\nexport CATALINA_OPTS=
"-Xmx1024M –XX:MaxPermSize=256M -Djavax.net.debug=ssl"\n' >> .bashrc
```

Those are single quotes in the commands above, not backticks. I prefer printf because it is precise and I can do more with the command versus echo. Also, the second statement needs to be one line, not two.

Create a Startup Script

As the root user, cut and paste the following into the new file /etc/init.d/tomcat:

```
#!/bin/bash
#
# chkconfig: 35 95 05
# description:  Apache Tomcat 6
#
# Version: 1.0
#
# Author       Date       CR/DR#          Change Description
# SecHard      02JUL11    N/A             Initial Release
#

# parameters go here...

# start/stop functions go here...

start() {
    runuser - tomcat -c "cd /tmp && /opt/tomcat/bin/catalina.sh start"
    touch /var/lock/subsys/tomcat
}

stop () {
      runuser - tomcat -c "cd /tmp && /opt/tomcat/bin/catalina.sh stop"
    rm -f /var/lock/subsys/tomcat
}


case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    restart)
        stop
            sleep 3
        start
        ;;
    *)
        echo "Usage: $0 { start | stop | restart }"
        exit 1
        ;;
esac
exit 0
```
Run the following commands to add the new service:

```
chmod 755 /etc/init.d/tomcat
chkconfig --add tomcat
```

Testing Tomcat

From a command prompt execute the following commands:

```
service tomcat stop
service tomcat start
```
Open a web browser and type in the URL:

http://<your_ip_for_test_system>:8080/

Example:

http://192.168.15.22:8080/

You should see a page open with default tomcat page and links to status, manager and online resources. This page will be removed shortly to lock the system down.

Removing superfluous objects

Run the following commands as root user:

```
su - root
cd /opt/tomcat
```
Secure the Configuration files from writes:

```
find /opt/tomcat/conf/ -type f -exec chmod 0400 {} \;
```
Secure the Logs and their directory (only the root user will be able to read the log files):

```
find /opt/tomcat/logs -type f -exec chmod 0300 {} \;
chmod 0750 /opt/tomcat/logs
```
Remove everything from the webapps directory (docs, examples, host-manager, manager, ROOT):

```
cd /opt/tomcat/webapps
rm -rf docs examples host-manager manager ROOT
```
Change the server version string from HTTP headers in server responses by changing the server keyword in conf/server.xml (add the line for server):

```
    <Connector port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />
    <!-- A "Connector" using the shared thread pool-->
```
Protect the shutdown port 8005. Modify conf/server.xml and add a really complex password.

Change from:

```
<Server port="8005" shutdown="SHUTDOWN">
```

To something really complex:

```
<Server port="8005" shutdown="2e8a48cd1adf84059fc43c33edb9e119b9136fc90b8c">
```

In the above line, I used md5sum on a random file under /var and plugged the hashed value here.

Configure Secure Socket Layer (SSL)

Run the following commands as tomcat user:

```
su – tomcat
cd /opt/tomcat/conf/
/opt/java/bin/keytool -genkeypair -v -alias "tomcat" \
-keyalg RSA -validity 2190 -keystore /opt/tomcat/conf/tomcat.keystore \
-dname "CN=foo.sechard.lan, OU=Security, O=Hard, L=MyCity, ST=TX, C=US" \
-storepass "changeme" -keypass "changeme"
```

This creates a new file in the /opt/tomcat/conf/ directory called tomcat.keystore.jks. The default password tomcat tries is “changeme”.

Modify server.xml and change the following lines from:

```
<!-- Define a SSL HTTP/1.1 Connector on port 8443
         This connector uses the JSSE configuration, when using APR, the
         connector should be using the OpenSSL style configuration
         described in the APR documentation -->
    <!--
    <Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
               maxThreads="150" scheme="https" secure="true"
               clientAuth="false" sslProtocol="TLS" />
    -->
```

To:

```
    <!-- Define a SSL HTTP/1.1 Connector on port 8443
         This connector uses the JSSE configuration, when using APR, the
         connector should be using the OpenSSL style configuration
         described in the APR documentation -->

<Connector port="8443"
         maxHttpHeaderSize="8192"
         server="Apache"
         protocol="HTTP/1.1"
         SSLEnabled="true"
         maxThreads="150"
         minSpareThreads="25"
         maxSpareThreads="75"
         disableUploadTimeout="true"
         acceptCount="100"
         scheme="https"
         secure="true"
         URIEncoding="UTF-8"
         clientAuth="false"
         sslProtocol="TLSv1"
         keyAlias="tomcat"
         keystoreFile="/opt/tomcat/conf/tomcat.keystore"
         keystorePass="changeme"
         />
```

Now test your new SSL configuration.

From a command prompt execute the following commands:

```
service tomcat stop
service tomcat start
```

Open a web browser and type in the URL:

https://<your_ip_for_test_system>:8443/

At this point you are done with securing the system. You could comment out port 8080 and only use the SSL port on 8443 or keep both if you choose.

Now you can deploy your applications from this secure platform. Deposit them into the /opt/tomcat/webapps/ directory and go to their respective root context, e.g. if I was deploying Jama Contour, I would place the web application into /opt/tomcat/webapps/ and then go to the URL:

https://<your_ip_for_test_system>:8443/contour

Appendix:

/opt/tomcat/conf/server.xml:

```
<?xml version='1.0' encoding='utf-8'?>
<Server port="8005" shutdown="2e8a48cd1adf84059fc43c33edb9e119b9136fc90b8c ">
  <Listener className="org.apache.catalina.core.AprLifecycleListener"
SSLEngine="on" />
  <Listener className="org.apache.catalina.core.JasperListener" />
  <Listener
className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />
  <Listener
className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />
  <Listener
className="org.apache.catalina.core.ThreadLocalLeakPreventionListener" />

  <GlobalNamingResources>
    <Resource name="UserDatabase" auth="Container"
              type="org.apache.catalina.UserDatabase"
              description="User database that can be updated and saved"
              factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
              pathname="conf/tomcat-users.xml" />
  </GlobalNamingResources>

  <Service name="Catalina">

    <Connector port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />

    <Connector port="8443"
         maxHttpHeaderSize="8192"
         server="Apache"
         protocol="HTTP/1.1"
         SSLEnabled="true"
         maxThreads="150"
         minSpareThreads="25"
         maxSpareThreads="75"
         disableUploadTimeout="true"
         acceptCount="100"
         scheme="https"
         secure="true"
         URIEncoding="UTF-8"
         clientAuth="false"
         sslProtocol="TLSv1"
         keyAlias="tomcat"
         keystoreFile="/opt/tomcat/conf/tomcat.keystore"
         keystorePass="changeme"
         />

    <!-- Define an AJP 1.3 Connector on port 8009 -->
    <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />

    <Engine name="Catalina" defaultHost="localhost">

      <Realm className="org.apache.catalina.realm.LockOutRealm">
        <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
```

```
               resourceName="UserDatabase"/>
      </Realm>

      <Host name="localhost" appBase="webapps"
            unpackWARs="true" autoDeploy="true"
            xmlValidation="false" xmlNamespaceAware="false">

        <Valve className="org.apache.catalina.valves.AccessLogValve"
directory="logs"
               prefix="localhost_access_log." suffix=".txt"
               pattern="%h %l %u %t &quot;%r&quot; %s %b" />

      </Host>
    </Engine>
  </Service>
</Server>
```