

# Securing Apache Tomcat for your environment

Mark Thomas

April 2009

# Who am I?

---

Apache Tomcat committer

Resolved 1,500+ Tomcat bugs

Apache Tomcat PMC member

Member of the Apache Software Foundation

Member of the ASF security committee

Created the Tomcat security pages

Senior Software Engineer and Consultant at  
SpringSource

# Background

---

Based on 6.0.x

Much applies to 5.5.x and 4.1.x

Security:

Confidentiality

Integrity

Availability

Usability

Need to balance for your environment

# Threats seen in the WILD

---

Very few Tomcat specific

Malicious webapp seen from July 2008

fex\*.war

Some variants download and execute code

Some variants provide (effectively) a shell prompt

Infection via insecure manager app

# Recent vulnerabilities

---

CVE-2008-5519

mod\_jk response mix-ups

CVE-2008-2370

RequestDispatcher

CVE-2008-2938

UTF-8

CVE-2007-5461

Write enabled WebDAV

CVE-2007-6286

APR / SSL

# Other areas

---

Tomcat configuration should not be your only line of defence

## OS

- Limit Tomcat privileges

- Limit access to Tomcat's files

- Minimise impact of a successful attack

## Firewall

- In and out

# Installation

---

If you don't need it, remove it

Web applications

docs

examples

host-manager

manager (possibly)

ROOT

# Security Manager



---

Provided by the JVM

Web applications run in a 'sandbox'

Some Tomcat testing is performed with this enabled

Enabling the security manager breaks expression language

- fails parts of the TCK

- fixes in hand (6.0.20?)

Configured in catalina.policy

# Security Manager

---

Do you trust your web applications?

Restricts actions of malicious web applications

Default policy very likely to break your application

Don't use it without testing

Develop with it enabled from the start

# server.xml

---

Configuration is reasonably secure by default

Discuss options to make it more/less secure

Values shown are defaults unless otherwise noted

If you are upgrading

- Don't copy server.xml from 5.5.x to 6.0.x

- Start with the server.xml from 6.0.x

- Add what you need

- Remove what you don't need

# server.xml

---

## Comments

Delete components you aren't using

Minimise other comments

`<Server ... />`

port="-1" (non-default) disables it  
shutdown should use strong password

`<Listener .../>`

Native (APR) on Solaris is not stable

# server.xml

---

<GlobalNamingResources .../>

Should not be using UserDatabase as primary realm

Only used for shared resources

Changes will require a Tomcat restart

<Service .../>

Nothing to see here

<Engine .../>

Nothing to see here

# server.xml

---

<Connector .../>

Do you need HTTP and AJP enabled?

address="..." (defaults to all)

allowTrace="false"

maxPostSize="2097152" only parameters

maxSavePostSize="4096"

xpoweredBy="false"

server="Server: Apache-Coyote/1.1"

server = "Microsoft-IIS/5.0"

# server.xml

---

## SSL

BIO & NIO use JSSE

Native (APR) uses OpenSSL

SSLEnabled="true"

scheme="https"

secure="true"

## Faking SSL

secure="true"

# server.xml

---

## AJP specific

request.secret=""..."" should be strong if used

request.shutdownEnabled="false" (default)

request.useSecret="false" (default)

tomcatAuthentication="true" (default)

# server.xml

---

`<Host .../>`

`autoDeploy="false"`

default is true

`deployOnStartup="true"`

If both false, only contexts defined in server.xml would be deployed

`deployXML="true"`

Set to false to ignore META-INF/context.xml

# server.xml / context.xml

---

<Context .../>

crossContext="false"

privileged="false"

allowLinking="false"

Don't change this on a case-insensitive OS

This includes Windows

# Nested components

---

<Valve .../>

Always configure an access log valve

Remove/archive old log files

Typically, one per host

e.g.:

```
<Valve className="org.apache.catalina.valves.AccessLogValve"  
  directory="logs" prefix="localhost_access_log."  
  suffix=".txt" pattern="common" resolveHosts="false" />
```

# Nested components

---

<Valve .../>

Use a Remote Address Filter where possible

The allow/deny attributes are regular expressions

allow is better than deny

e.g.:

```
<Valve className="org.apache.catalina.valves.RemoteAddrValve"  
  allow="127\.\0\.\0\.\1" />
```

# Nested components

---

<Realm .../>

Don't use these in production:

- Memory Realm

- JDBC Realm

Try to avoid:

- UserDatabase Realm

That leaves:

- DataSource Realm

- JNDI Realm

- JAAS Realm (not widely used)

# Nested components

---

<Realm .../>

There is no account lock-out implemented

Brute-force attacks will work (eventually)

May be protected by JNDI service if userPassword is not set

Suspected infection method for fex\* malicious web application

# Nested components

---

<Realm .../>

New for 6.0.19

Combined Realm

- Realm nesting

- Use multiple Realms for authentication

LockOut Realm

- Wraps standard realms

- Locks accounts after multiple failed logins

# Nested components

---

<Realm .../>

LockOut Realm example:

```
<Realm className="org.apache.catalina.realm.LockOutRealm"
  failureCount="5" lockOutTime="300" cacheSize="1000"
  cacheRemovalWarningTime="3600">
  <Realm className="org.apache.catalina.realm.DataSourceRealm"
    dataSourceName=... />
</Realm>
```

# Nested components

---

<Loader .../>

Nothing to see here

<Manager .../>

entropy="this.toString()"

This has been shown to be predictable

APR provides real random value

randomClass="java.security.SecureRandom"

sessionIdLength="16"

# System properties

---

org.apache.catalina.connector.  
RECYCLE\_FACADES="false"

org.apache.catalina.connector.  
CoyoteAdapter.ALLOW\_BACKSLASH="false"

org.apache.tomcat.util.buf.  
Udecoder.ALLOW\_ENCODED\_SLASH="false"

org.apache.coyote.  
USE\_CUSTOM\_STATUS\_MSG\_IN\_HEADER="false"

STRICT\_SERVLET\_COMPLIANCE="true"

XSS UTF7 browser issues

# JNDI resources

---

server.xml or context.xml

Why is the password in plain text?

Tomcat needs the plain text password to connect to the external resource

Encrypting the password means Tomcat would need a decryption key – back to the original problem

# JNDI resources

---

What are the risks of a plain text password?

Remote information disclosure

Is the resource remotely accessible?

Local information disclosure

If an attacker has console access you probably have bigger issues

If the attacker can replace Tomcat code they may well be able to read the plain text password from memory

# JNDI resources

---

## Solutions to the plain text password issue:

- Enter the password at Tomcat start

  - Needs custom code

  - Password still in memory

  - Tomcat restart requires manual input

- Encode the password

  - Needs custom code

  - Encoding is not encryption

  - May prevent some accidental disclosures

# web.xml

---

## Default servlet

`readonly="true"`

`listings="false"`

## Invoker servlet

bypass security constraints

will be removed from 7.0.x onwards

# Monitoring

---

Most monitoring tools also provide management functionality

Is the benefit of monitoring worth the risk of exposing the management tools?

## Options

manager

LambdaProbe

JMX

AMS

Numerous others

# Monitoring

---

Always configure an access log  
Always a Remote Address Filter  
Configure strong passwords

# Policy / process: Review logs

---

Which logs?

- Access logs

- Application logs

- Tomcat logs

What do you do if you find an attempted attack?

What do you do if you find a successful attack?

# Policy / process

---

Do you monitor Tomcat vulnerability announcements?

What do you do if one affects you?

Impact will be:

- configuration change

- patch

- upgrade

Likely to require a Tomcat restart

# Avoiding downtime

---

Restart = downtime

Using a cluster minimises downtime

- one httpd instance

- two Tomcat instances

- mod\_proxy\_http

- mod\_proxy\_balancer

- sticky sessions (optional)

- session replication (optional)

# Easing upgrades

---

## Standard installation

```
/tomcat/apache-tomcat-6.0.18  
cd /tomcat/apache-tomcat-6.0.18/bin  
./catalina.sh start
```

## CATALINA\_HOME / CATALINA\_BASE

```
/tomcat/apache-tomcat-6.0.18  
/tomcat/instance01  
CATALINA_BASE=/tomcat/instance01  
export CATALINA_BASE  
cd /tomcat/apache-tomcat-6.0.18/bin  
./catalina.sh start
```

# Easing upgrades

---

## Upgrading

```
/tomcat/apache-tomcat-6.0.18
/tomcat/apache-tomcat-6.0.19
/tomcat/instance01

CATALINA_BASE=/tomcat/instance01
export CATALINA_BASE
cd /tomcat/apache-tomcat-6.0.18/bin
./catalina.sh stop
cd /tomcat/apache-tomcat-6.0.19/bin
./catalina.sh start
```

# Easing upgrades

---

## Downgrading

```
/tomcat/apache-tomcat-6.0.18  
/tomcat/apache-tomcat-6.0.19  
/tomcat/instance01  
  
CATALINA_BASE=/tomcat/instance01  
export CATALINA_BASE  
cd /tomcat/apache-tomcat-6.0.19/bin  
./catalina.sh stop  
cd /tomcat/apache-tomcat-6.0.18/bin  
./catalina.sh start
```

# Questions?

---

[mark.thomas@springsource.com](mailto:mark.thomas@springsource.com)

<http://www.springsource.com/webinars>

[markt@apache.org](mailto:markt@apache.org)

[users@tomcat.apache.org](mailto:users@tomcat.apache.org)