

ERNW
providing security.

Apache Tomcat Hardening

Tomcat 7.x

Version:	1.00
Date:	1/11/2014
Classification:	Public
Author(s):	Matthias Luft, Florian Grunow, Hendrik Schmidt

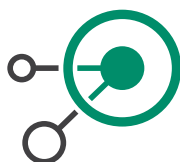


TABLE OF CONTENT

1	INTRODUCTION	4
2	OPERATIONS AND SYSTEM SECURITY	5
2.1	PATCH AND VULNERABILITY MANAGEMENT	5
2.2	LEAST PRIVILEGE FOR THE TOMCAT SERVICE	5
2.3	RESTRICT ACCESS TO TOMCAT FOLDER	6
3	ADMINISTRATIVE INTERFACES	6
3.1	NETWORK LEVEL RESTRICTIONS	6
3.2	'MINIMAL PRINCIPLE' AUTHORIZATION CONCEPT	7
4	AUTHENTICATION	7
4.1	SECURE AUTHENTICATION.....	7
4.2	DISABLE REALMS	7
5	SESSION HANDLING	8
5.1	SESSION TIMEOUT	8
5.2	HTTPOONLY FLAG	8
5.3	CSRF PROTECTION	8
6	NETWORK SECURITY	9
6.1	RESTRICT LISTENING INTERFACES.....	9
6.2	RESTRICT ALLOWED NETWORK CONNECTIONS.....	9
6.3	ENCRYPT NETWORK CONNECTIONS	9
7	JAVA RUNTIME	10
7.1	JAVA SECURITY MANAGER.....	10
7.2	PACKAGE ACCESS.....	10
8	GENERAL SETTINGS	10
8.1	SECURE DEFAULT SETTINGS.....	10
8.2	SECURE SHUTDOWN PORT	11
8.3	UNDEPLOY DEFAULT APPLICATIONS	11
8.4	CUSTOM ERROR PAGES	11
8.5	DISABLE AUTOMATIC DEPLOYMENT.....	12



ERNW
providing security.

9 APPENDIX: DEFAULT SETTINGS 13

1 INTRODUCTION

As no official hardening guide for Tomcat 7 is available yet, ERNW has compiled the most relevant settings into this checklist. While there is a significant amount of controls that can be applied, this document is supposed to provide a solid base of hardening measures. Settings which might have severe impact on the functionality of the operating system and need a lot of further testing are not part of this checklist or marked as optional.

We have marked each recommended setting in this checklist either with "mandatory" or "optional" to make a clear statement, which setting is a MUST (mandatory) or a SHOULD (optional) from our point of view. "Optional" also means that we recommend to apply this setting, but there may be required functionality on the system that will become unavailable once the setting is applied.

2 OPERATIONS AND SYSTEM SECURITY

2.1 Patch and Vulnerability Management

Security-relevant Tomcat updates must be installed in a timely manner:

- Critical or Important priority updates and patches must be applied within 10 days.
- Moderate priority updates and patches are to be deployed within maximum of 30 days after release.
- Low priority updates are to be applied after maximum of 90 days after release.

Information on the availability and severity of patches can be found at <http://tomcat.apache.org/lists.html#tomcat-announce>.



Updates may break functionality. In case of an update for a core/business Tomcat instance, check <http://tomcat.apache.org/lists.html#tomcat-announce> for potential side effects.

Mandatory

2.2 Least Privilege for the Tomcat Service

Run the Tomcat application server with low privileges on the system. Create a dedicated service user for Tomcat that only has a minimum set of privileges (e.g. remote login with the user must not be allowed). Installations using operating system repositories must be reviewed for the use of a low-privilege user in order to ensure that Tomcat doesn't run as root/administrative user. This must be ensured on the operating system level configuring the respective startup mechanisms [e.g. Windows services on Microsoft Windows¹ or the rc-scripts on Unix systems]:

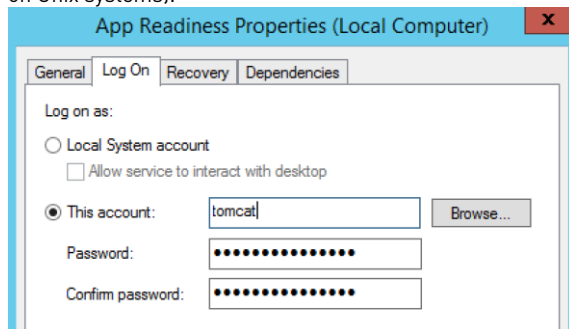


Figure 1: Sample Windows Service Configuration

```

1 #!/bin/sh
2 #
3
4 # PROVIDE: tomcat70
5 # REQUIRE: LOGIN
6 # KEYWORD: shutdown
7
8 tomcat70_enable="${tomcat70_enable:-"NO"}"
9 tomcat70_user="tomcat"

```

Figure 2: Sample FreeBSD rc-script²

Mandatory

¹ Please refer to the ERNW Windows Hardening Guides for secure service configuration.

² The particular configuration highly depends on both the Unix system and the Tomcat version. For example, the option to configure the user account in the rc.conf has been dropped for Tomcat 7 on FreeBSD, hence the username is hardcoded in the startup-script – which is not recommended. On FreeBSD, web and application servers are supposed to run as www user: <http://www.freebsd.org/doc/en/books/porters-handbook/using-php.html#WEB-APPS>.

Using a less privileged user to run Tomcat might impact the use of certain features Tomcat provides. For example, setting the listening port to a privileged port will fail. This can be mitigated by using a local HTTP proxy on the system (e.g. Apache with *mod_jk*) or packet forwarding rules that forward incoming traffic to the unprivileged port Tomcat is configured to listen on.

2.3 Restrict Access to Tomcat Folder

The Tomcat folders should only be accessible by the tomcat user itself. This is especially valid for the directories `${tomcat_home}/conf/` and `${tomcat_home}/webapps`.

When auto-deployment via the application server is not needed, the standard configuration is to have all Tomcat files owned by root with the group set to Tomcat. Then use `chmod 740` to only let root edit the files and make them available to read for the Tomcat user. Exceptions are, temporary and work directories that are owned by the Tomcat user rather than root.



This will break auto-deployment. (Refer also to Section 8.5.)

Optional

3 ADMINISTRATIVE INTERFACES

The main administrative interface of Tomcat is the so-called *Manager* application. This application is found to be very exposed (e.g. no network level restrictions combined with weak/default credentials) in many environments, even though its protection is essential for the secure operation of the Tomcat server. If possible, administrative tasks should be carried out on the operating system level (e.g. using RDP or SSH) and the Manager application should be undeployed (refer also to Section 8.3). If this is not possible, the controls described in the following subsection (maybe in combination with a jump server concept) should be applied in combination with secure authentication mechanisms (refer also to Section 4) and encrypted transport (refer to Section 6.3).

3.1 Network Level Restrictions

If the manager application is used, the administrative interface should only be accessible from authorized IP addresses.

This can be accomplished by setting the following configuration options in the file `CATALINA_HOME/webapps/manager/META-INF/context.xml` inside the Context option. The following configuration can be used to restrict access to only the localhost and a specific IP address or IP address range:



```
<Valve className="org.apache.catalina.valves.RemoteAddrValve"
  allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1|172\.16\.16\.\d{1,3}
"/>
```

Hostnames can also be used:



```
<Valve className="org.apache.catalina.valves.RemoteHostValve"
  allow=".*\.admins\.domain\.com" />
```

The allow and deny attributes interpret the given value as a regular expression (using `java.util.regex`).

Optional

3.2 'Minimal Principle' Authorization Concept

<p>Depending on the given tasks, only rights for respective roles must be assigned. The following roles are available and must be assigned with minimal privileges in mind:</p> <ul style="list-style-type: none"> - <code>manager-gui</code>: Access to the web interface - <code>manager-status</code>: Access to the "Server Status"-page only - <code>manager-script</code>: Access to the script-oriented plain-text interface and "Server Status" page - <code>manager-jmx</code>: Access to the JMX proxy interface and the "Server Status" page 	Mandatory
---	-----------

4 AUTHENTICATION

The following controls apply to Tomcat-based authentication in general. However, in most environments, this applies mainly to the Manager application, which is thus used as the example. If deployed applications use the Tomcat-based authentication, the controls apply to those as well.

4.1 Secure Authentication

<p>If the Manager application is used, additional authentication mechanisms should be implemented. The authentication mechanisms are prioritized in the following order:</p> <ol style="list-style-type: none"> 1. LDAPS or client certificates 2. Local (digest based) <p>Additionally, the authentication procedure and the communication with the manager application must be secured with SSL (see below).</p>	Optional
--	----------

<p>For local and certificate based authentication, an account lockout must be enforced³. To prevent brute force attacks, the authentication realm in use must be placed within a lockout realm using the following configuration:</p> <p>Edit the file <code>CATALINA_HOME/conf/server.xml</code> and add the lockout realm with the following option:</p> <pre style="margin-top: 10px;"> <Realm className="org.apache.catalina.realm.LockOutRealm" failureCount="5" lockOutTime="30"> <!-- AUTHENTICATION REALM --> </Realm></pre>	Mandatory
---	-----------

4.2 Disable Realms

<p>There are multiple realms that are not intended for productive use. These realms must be disabled.</p> <p>Open the file <code>CATALINA_HOME/conf/server.xml</code>. Search for the <code>MemoryRealm</code> and disable it. The <code>JDBCRealm</code> must also be disabled. Use the <code>DataSourceRealm</code> instead. When using a large-scale installation, do not use the <code>UserDatabaseRealm</code> and disable it as well.</p>	Mandatory
---	-----------

³ In case of central authentication, the directory services must be configured accordingly.

5 SESSION HANDLING

5.1 Session Timeout

The session timeout for all web applications must be set to *20 minutes*.

This can be done by editing the file in the `CATALINA_HOME/conf/web.xml` and setting the following configuration option:

```

<session-config>
  <session-timeout>20</session-timeout>
</session-config>

```

Mandatory

5.2 HttpOnly Flag

Tomcat 7 sets the *HttpOnly cookie flag* automatically on session cookies. Review the configuration to ensure that this option is not disabled.

HttpOnly can be activated with the following configuration option that must be set in `CATALINA_BASE/conf/context.xml` to be enabled globally on all applications:

```

<Context useHttpOnly='true' .../>

```

Using *HttpOnly* may break application functionality if access to the session cookies via JavaScript is necessary. If an application needs access to *HttpOnly* cookies via JavaScript, an exception can be defined in the individual context inside the application files in `/META-INF/context.xml`.

Mandatory

5.3 CSRF Protection

Cross Site Request Forgery protection in Tomcat must be enabled in order to protect applications. Tomcat 7 provides a basic CSRF protection. A filter can be configured in `CATALINA_BASE/conf/web.xml` globally. The filter can be overwritten by each application using the file `WEB-INF/web.xml`.

The following configuration options must be set:

```

<filter-mapping>
  <filter-name>CSRFPreventionFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>

```

For a detailed explanation and additional options, refer to the Tomcat manual⁴.

Using CSRF prevention may break application functionality. This must be kept in mind especially in case of applications which make heavy use of asynchronous requests.

Mandatory

⁴ http://tomcat.apache.org/tomcat-7.0-doc/config/filter.html#CSRF_Prevention_Filter

6 NETWORK SECURITY

6.1 Restrict Listening Interfaces

Prevent the connectors from listening on all interfaces/IP addresses available on the server system. Instead, the IP address must be specified.

Edit the file `CATALINA_HOME/conf/server.xml`. Review every connector and specify the correct IP address:

```
<Connector port="TCP_PORT" address="LISTEN_IP_ADDRESS"...
```

This prevents applications from accidentally being served on an exposed interface.

Mandatory

6.2 Restrict Allowed Network Connections

Only open necessary Tomcat ports. These default to TCP port `8080` and `8443`. Make sure that these ports are correctly configured in Tomcat and on an existing packet filter that can be installed on the system.

Open the file `CATALINA_HOME/conf/server.xml` and review every Connector configuration for the correct/desired port assignment. Remove unused ports/connectors.

Mandatory

6.3 Encrypt Network Connections

To secure sensitive applications (such as the Manager application), the usage of SSL must be configured (and it is also mandatory for hosted applications that process sensitive data/provide log-in functionality). The first step is the creation of a trustworthy certificate to avoid certificate warning messages, and to offer the end user a way to verify a trustworthy connection.

The second step is the creation of a certificate keystore, containing `CA` and server certificate and the corresponding private key. The password for the keystore should be created according to the recommendations from the "Ensure password security" subsection above.

To enable SSL support, the following line can be used (the exact configuration options depend on the given platform and requirements) and must be placed in the `CATALINA_HOME/conf/server.xml`:

```
<Connector protocol="org.apache.coyote.http11.Http11Protocol"
port="8443" scheme="https" secure="true" SSLEnabled="true"
sslProtocol="TLS" keystoreFile="path to keystore file"
keystorePass="keystore password"/>
```

Restriction of available SSL Ciphers by adding the ciphers attribute to the SSL Connector with the following cipher suites:

```
<Connector ciphers="SSL_RSA_WITH_RC4_128_SHA,
TLS_RSA_WITH_AES_128_CBC_SHA, TLS_DHE_RSA_WITH_AES_128_CBC_SHA,
TLS_DHE_DSS_WITH_AES_128_CBC_SHA, SSL_RSA_WITH_3DES_EDE_CBC_SHA,
SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA,
SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA" ...
```

The enforcement of SSL only communication must be done in one of the following 2 ways: To force the usage of HTTPS for all Web Applications hosted on the Tomcat, each security-constraint tag within each `CATALINA_HOME/webapps/$WEBAPP/WEB-INF/web.xml` must

Mandatory



include the following lines right before the closing `</security-constraint>` tag:



```
<user-data-constraint>
  <transport-guarantee>CONFIDENTIAL</transport-guarantee>
</user-data-constraint>
```

7 JAVA RUNTIME

7.1 Java Security Manager

In order to restrict the capabilities of single applications, the Java Security Manager can be used. The security policies implemented by the Java SecurityManager are configured in the file `$CATALINA_HOME/conf/catalina.policy`. Once the `catalina.policy` file is configured, Tomcat can be started with a SecurityManager in place by using the `--security` option. For a comprehensive description of the configuration settings, see the official Tomcat Security Manager How-To⁵.



As basically all kinds of permissions (e.g. access to single files and directories or Java packages) should be configured per application, this control significantly increases operational effort. In addition, policy files which are too restrictive will break application functionality.

Optional

7.2 Package Access

Tomcat provides the possibility to restrict access to certain Java packages. If access to restricted packages is detected, a security exception will be thrown.



To restrict packages from being accessed, open the file `$CATALINA_BASE/conf/catalina.properties` and add disallowed packages to the `package.access` list.

In order to assess which packages are needed by which applications, the Java imports can be analyzed. On a Unix system, this can for example be achieved using the following command:

```
grep -R import ${tomcat home}/webapps/WEBAPP
```

Optional

8 GENERAL SETTINGS

8.1 Secure Default Settings

To prevent possible vulnerabilities, several default values must be checked. Refer to Section 3 for a list of default configuration settings which must not be changed.



Mandatory

⁵ <http://tomcat.apache.org/tomcat-7.0-doc/security-manager-howto.html>

8.2 Secure Shutdown Port

<p>If the functionality to shut down Tomcat using a network port on the local Tomcat system down is needed, the password must be set to a strong and hard to guess passphrase.</p> <p>Edit the file <code>CATALINA_HOME/conf/server.xml</code> and set the shutdown passphrase:</p> <pre><Server port="8005" shutdown="NonDeterministicWordSoShutdownPWisNotEasyToGuess"></pre> <p>If this functionality is not needed, it must be deactivated with the following option</p> <pre><Server port="-1" shutdown="SHUTDOWN"></pre> <p>The local management scripts allow a shutdown of the server even if the shutdown port is disabled.</p>	<p>Mandatory</p>
--	-------------------------

8.3 Undeploy Default Applications

<p>Tomcat might ship with a number of default web applications. These must be removed if not absolutely needed.</p> <p> Remove all default web applications from <code>\${tomcat_home}/webapps</code>. Standard applications which must be removed are <i>ROOT</i>, <i>docs</i>, <i>examples</i>, <i>host-manager</i>, and <i>manager</i>.</p> <p> The manager application provides administrative functionality such as deploying apps and retrieving log information. This should and can be performed using the local server command line interface. However, if this application is absolutely needed, it must be secured using SSL.</p>	<p>Mandatory</p>
---	-------------------------


8.4 Custom Error Pages

<p>As the standard error pages disclose internal information (such as version number and stack traces), they should be replaced by custom error pages using a generic error message like <i>"An error occurred while processing your request"</i>.</p> <p>The following lines must be included in the <code>web.xml</code> of each web application, located in <code>CATALINA_HOME/webapps/\$WEB_APP/WEB-INF</code>:</p> <pre><error-page> <error-code>500</error-code> <location>/errorpages/error.html</location> </error-page> <error-page> <exception-type>java.lang.Throwable</exception-type> <location>/errorpages/error.html</location> </error-page></pre> <p>In addition, if the manager application has not been removed, the "Tomcat 7" version information must be manually removed from the error pages located in the <code>CATALINA_HOME/webapps/manager/WEB-INF/jsp/</code>.</p>	<p>Optional</p>
---	------------------------

8.5 Disable Automatic Deployment

Tomcat allows automatic deployment of applications while Tomcat is running. This functionality must be disabled as it may allow malicious or untested applications to be deployed.

Automatic deployment is controlled by the *autoDeploy* and *deployOnStartup* attributes. If both are *false*, only Contexts defined in `server.xml` will be deployed and any changes will require a Tomcat restart. To disable automatic deployment change the following lines in the `$CATALINA_HOME/conf/server.xml` file:

```
 autoDeploy="false"  
deployOnStartup="false"
```

In a hosted environment where web applications may not be trusted, also set the *deployXML* attribute to false to ignore any `context.xml` packaged with the web application that may try to assign increased privileges to the web application.

Mandatory

9 APPENDIX: DEFAULT SETTINGS

The following items list default settings which must not be changed as these are considered secure by default:

- The value of *allowTrace* for each Connector within the *server.xml* is either not present or set to *false*.
- In all *context.xml* files set the *privileged* attribute to false unless it is required like for the manager application:
 - `<Context ... privileged="false" />`
- Ensure that the *crossContext* value is either not present or set to false.
Allowing *crossContext* creates the possibility for a malicious application to make requests to a restricted application.
`<Context ... crossContext="false" />`
- Ensure that the *allowLinking* value is either not present or set to false.
Allowing symbolic links creates the possibility for directory traversal and source code disclosure vulnerabilities.
`<Context ... allowLinking="false" />`
- Do not allow write access for default servlet.
 - The *DefaultServlet* is configured with *read-only* set to true in the *web.xml* file. Changing this to false allow clients to delete or modify static resources on the server and to upload new resources. This should not normally be changed without requiring authentication.
- Do not enable listing
 - The *DefaultServlet* is configured with *listings* set to false. This is not only because allowing directory listings is considered unsafe but because generating listings of directories with thousands of files can consume significant CPU resources, leading to a DoOS attack.
- When the *RECYCLE_FACADES* option is set to true, Tomcat will recycle the session facade between requests. This will lead to information leakage between requests. By default, this parameter is not set. Ensure that the startup script in use does not contain the following line:
 - `-Dorg.apache.catalina.connector.RECYCLE_FACADES=false`
- Being able to specify different path delimiters on Tomcat gives an attacker the possibility to access applications that were previously blocked by a proxy like *mod_proxy*. Per default, this parameter is not set.
 - Ensure that the startup script in use does not contain the following line:
 - `-Dorg.apache.catalina.connector.CoyoteAdapter.ALLOW_BACKSLASH=false`
 - `-Dorg.apache.tomcat.util.buf.UDecoder.ALLOW_ENCODED_SLASH=false`
- Being able to specify custom header status messages enables an attacker to inject headers. This allows a potential attack vector for Cross-Site Scripting attacks. Per default, this parameter is not set.
 - Ensure that the startup script in use does not contain the following line:
 - `-Dorg.apache.coyote.USE_CUSTOM_STATUS_MSG_IN_HEADER=false`