Bayerisches Staatsministerium für
Ernährung, Landwirtschaft und Forsten

## Benutzerdefinierter Baustein

# Tomcat-Server

**für den IuK-Einsatz**

**im Geschäftsbereich des StMELF**

März 2014

## Dokumentation

| Version | Datum | Ersteller / Änderer | Beschreibung der Änderungen |
|---|---|---|---|
| V 0.1 | 18.10.13 | Hanel (secuvera) | Erster Entwurf |
| V 0.5 | 06.11.13 | Hanel (secuvera) | Ergänzung, Formatierung |
| V 0.9 | 18.02.14 | Hanel (secuvera) | Kommentare Ref. P5b |
| V 0.95 | 03.03.14 | Glemser (secuvera) | Interne QS |
| V 1.0 | 06.03.14 | Hanel (secuvera) | Finalisierung |

## Dokument Review

| Datum Review | Dokument Version | Reviewer |
|---|---|---|
| | | |
| | | |
| | | |
| | | |

## Mitarbeit

Folgende Mitarbeiter haben bei der Erstellung und Kommentierung tatkräftig unterstützt:
Herr Martin Duschl, StMELF
Frau Sabine Gramlich Huaroto, StMELF
Herr Sebastian Mey, StMELF

IMPRESSUM
Bayerisches Staatsministerium für Ernährung, Landwirtschaft und Forsten
Referat P5, Informations- und Kommunikationstechnik, IT-Sicherheit
Ludwigstraße 2, 80539 München
www.stmelf.bayern.de

Zuständigkeit:

Inhalt und Redaktion:
Sebastian Mey StMELF 089 2182-2560

## Inhaltsverzeichnis

# 1. Vorwort

Hintergrund für die Erstellung des Bausteins Tomcat Server ist die Umsetzung eines IT-Sicherheitskonzepts auf der auf Basis von IT-Grundschutz im Bayerischen Staatsministerium für Ernährung, Landwirtschaft und Forsten. Bedingt durch die Outsourcing-Strategie wurden Server des IT-Verbunds zum zentralen Landes-Rechenzentrum verlagert. Im Zuge dieser Verlagerung werden die Server-Hardware und das Server-Betriebssystem (Linux) durch das Landes-Rechenzentrum betreut. Die Betreuung der Fachverfahren und die Administration der Middleware (Tomcat) selbst, verblieben aber beim Staatsministerium. Das Landes-Rechenzentrum setzt seinerseits selbst IT-Grundschutz um, dennoch wurde durch das IT-Sicherheitsmanagement ein Weg gesucht, die verbleibenden Aufgaben der Systemadministration der Tomcat-Server im Rahmen des IT-Sicherheitsprozesses besser abbilden und dokumentieren zu können. Als Ergebnis dieser Überlegungen hat das IT-Sicherheitsmanagement beschlossen, einen benutzerdefinierten Baustein „**Tomcat-Server**“ zu erstellen.

Hierbei wurde insbesondere auf die **Lesbarkeit**, **Umsetzbarkeit** und **Praxisorientierung** des Dokuments Wert gelegt. Die Maßnahmen sollten kompakt gestaltet, möglichst frei von Redundanzen und konkret umsetzbar gestaltet werden. Der Baustein sollte deshalb nie für sich allein stehen, sondern fokussiert das Thema Tomcat und auch nur auf dieses. Zu tiefe Darstellungen werden zu Gunsten der Lesbarkeit vermieden. An diesen Stellen werden aber immer Links auf weiterführende Darstellungen oder ergänzende Informationen angeboten.

Dieser Baustein folgt daher bereits dem Ansatz von Grundschutz+ bzw. Grundschutz NEU. Das Bayerische Staatsministerium für Ernährung, Landwirtschaft und Forsten hofft, dass dieser Baustein für Sie hilfreich ist.

München, März 2014

Referat für Informations- und Kommunikationstechnik, IT-Sicherheit
im Bayerischen Staatsministerium für Ernährung, Landwirtschaft und Forsten

## 2. Anforderungen aus den BSI-Standards

Für die Erstellung dieses Bausteins wurden die Anforderungen aus den BSI-Standards berücksichtigt.
Zu nennen ist hier insbesondere das Kapitel 4 „Erstellung benutzerdefinierter Bausteine“ des Dokuments „Ergänzung zum BSI-Standard 100-3, Version 2.5“. In diesem Dokument wird ausgeführt: „Häufig soll auf Basis der Risikoanalyse ein benutzerdefinierter Baustein erstellt werden, für einen Themenbereich, der bisher in den IT-Grundschutz-Katalogen noch nicht ausreichend abgedeckt war, um den betrachteten Informationsverbund modellieren zu können.“

Das Kapitel 4 dieses Dokuments enthält eine Kurz-Beschreibung des Bausteins, die an die Beschreibung der Bausteine in den Grundschutz-Katalogen angelehnt ist. Das Kapitel 4.1 „Gefährdungslage“ enthält die Gefährdungsanalyse mit allen für den Baustein als relevant angesehenen Gefährdungen. Hierbei kommen benutzerdefinierte Gefährdungen und relevante Gefährdungen aus den Grundschutzkatalogen zur Anwendung. Die benutzerdefinierten Gefährdungen werden im Kapitel 5 detailliert erläutert. Benutzerdefinierte Gefährdungen werden im Rahmen dieses Bausteins als notwendig angesehen, da eine spezifische Behandlung der Sicherheitsprobleme in einem Tomcat-Server allein mit den im Standard 100-3 empfohlenen Elementaren Gefährdungen (sogenannte G 0.x-Gefährdungen) schlecht möglich sind. Weiterhin zeigen die Bausteine der IT-Grundschutzkataloge bis einschließlich zur 13. Ergänzungslieferung, dass neben allgemeinen Gefährdungen auch immer wieder bausteinspezifische Gefährdungen gewählt wurden. Dieser benutzerdefinierte Baustein lehnt sich daran an und beschreibt Tomcat-spezifische Gefährdung als benutzerdefinierte Gefährdungen. Zur Kontrolle, ob alle als relevant angesehenen Gefährdungen auch durch die Maßnahmen des Bausteins abgedeckt werden können, dient das Kapitel 7 „Kreuzreferenztabelle“. Diese aus den Hilfsmitteln der Grundschutzkataloge bekannte Tabelle, stellt die Abdeckung der Gefährdungen durch die Sicherheitsmaßnahmen in einer Matrix dar.

Zur Wahrung der Anwendbarkeit dieses benutzerdefinierten Bausteins werden Hinweise gegeben, wie die Modellierung in einem typischen IT-Verbund erfolgen sollte (siehe Kapitel 3). Die Maßnahmen im Kapitel 4.2 orientieren sich am Lebenszyklus-Modell der Grundschutz-Kataloge. Jede benutzerdefinierte Maßnahme wird in Kapitel 6 erläutert und enthält die Prüffragen, die mit der 13. Ergänzungslieferung durchgängig eingeführt wurden. Die Prüffragen sind so formuliert, dass sie als Checkliste benutzt werden können, um die Umsetzung der Maßnahmen kontrollieren zu können.

## 3. Hinweis zur Modellierung

Beim Tomcat-Server handelt es sich im Sinne des IT-Grundschutzes um einen Webserver. Gemäß Kapitel 2.1 „Modellierung nach IT-Grundschutz“ der Grundschutz-Kataloge des BSI ist daher immer der Baustein B 5.4 Webserver (parallel) zu modellieren! D.h., wird dieser benutzerdefinierte Baustein „Tomcat-Server“ verwendet, ist zusätzlich der Baustein B 5.4 Webserver zu verwenden, um eine komplette Maßnahmen-Abdeckung sicherzustellen. Die Maßnahmen des Baustein B 5.4 Webserver werden in diesem benutzerdefinierten Baustein deshalb, wie üblich, nicht nochmals aufgeführt.

**Beispiel für die Modellierung:**

Für die Modellierung eines „typischen“ Web-Servers so wie er für viele Verfahren im Einsatz ist, sind die folgenden Bausteine zu modellieren:

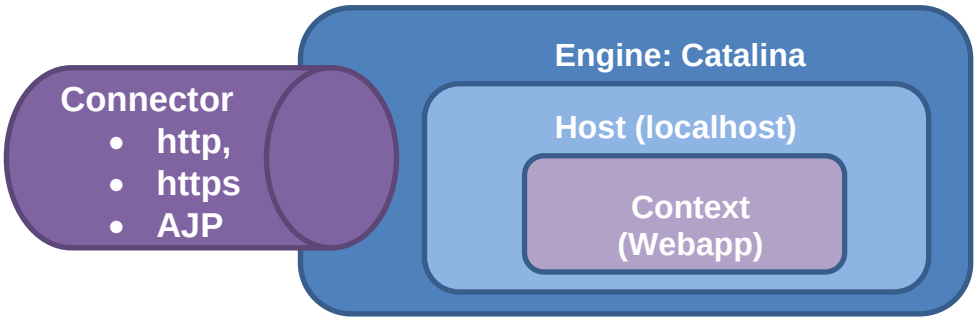
- B 3.101 Allgemeiner Server
- B 3.102 Server unter Unix bzw. der äquivalente Windows-Server Baustein
- bB Tomcat-Server
- B 5.4 Webserver
- B 5.21 Webanwendungen (Hinweis: Ist auf jede Webanwendung einmal anzuwenden)

# 4. Beschreibung des Bausteins

## 4.1 Erläuterungen zu Tomcat

Apache Tomcat ist ein Open Source Webserver und Webcontainer, der die Spezifikation für Java Servlets und JavaServer Pages implementiert und es erlaubt, in Java geschriebene Web-Anwendungen auf Servlet- bzw. JSP-Basis auszuführen[1].

Die Tomcat-Architektur ist in der folgenden Grafik dargestellt. Sie besteht aus dem Servlet-Container Catalina[1], der JSP-Engine[2] und dem Connector-Framework[3].
Über das Connector-Framework unterstützt Tomcat diverse Kommunikationsprotokolle und kann mit den HTTP-/HTTPs-Connectoren entweder als eigenständiger Webserver betrieben oder mittels des AJP-Connectors in andere Webserver z. B. Apache Web-Server integriert werden[4].




Die Engine ist der Top-Level Container und kann einen oder mehrere Hosts enthalten. Der Host definiert einen virtuellen Host unterhalb der Engine, welcher viele Contexts (Webapplikationen) enthalten kann.

## 4.2 Beschreibung des Bausteins

In diesem benutzerdefinierten Baustein wird die Tomcat-Server-Instanz betrachtet, die eine Schlüssel-Instanz für die Bereitstellung der Fachverfahren ist. Die Administration und der Betrieb der Tomcat-Server ist eine wichtige Aufgabe innerhalb der IT-Betreuung. Der benutzerdefinierte Baustein wird wie alle anderen Bausteine der Grundschutzkataloge strukturiert.

Zuerst wird die Gefährdungslage erläutert, danach werden Maßnahmenempfehlungen gemäß dem Lebenszyklus-Modell dokumentiert. Für jede individuelle Gefährdung und jede individuelle Maßnahme wird der entsprechende Inhalt dokumentiert.

---

[1] http://tomcat.apache.org/tomcat-8.0-doc/index.html
[2] http://tomcat.apache.org/tomcat-8.0-doc/jasper-howto.html
[3] http://tomcat.apache.org/tomcat-8.0-doc/connectors.html
[4] http://de.wikipedia.org/wiki/Apache_Tomcat

## 4.3 Gefährdungslage

Die Tomcat-Server sind wie alle IT-Systeme vielfältigen Gefahren ausgesetzt. In diesem benutzerdefinierten Baustein wird insbesondere auf die Gefährdungen eingegangen, die für die Tomcat-Server in der bestehenden Einsatzumgebung relevant sind. Benutzerdefinierte Gefährdungen sind in Klammern mit „(benutzerdefiniert)“ gekennzeichnet. Die Gefährdungslage wird gruppiert in:

- Höhere Gewalt
- Organisatorische Mängel
- Menschliche Fehlhandlungen
- Technisches Versagen und
- Vorsätzliche Handlungen

**Höhere Gewalt**
bG 1.2 Ausfall einer Tomcat-Instanz (benutzerdef.)
G 1.1 Personalausfall

**Organisatorische Mängel**
bG 2.1 Verwendung nicht abgesicherter Protokolle zur Administration (benutzerdef.)
G 2.7 Unerlaubte Ausübung von Rechten
G 2.9 Mangelhafte Anpassung an Veränderungen beim IT-Einsatz
G 2.111 Kompromittierung von Anmeldedaten bei Dienstleisterwechsel

**Menschliche Fehlhandlungen**
bG 3.1 Fehlerhafte Administration von Tomcat-Servern (benutzerdef.)
bG 3.2 Fehlerhafte Einbindung eines Tomcat-Servers in die Systemumgebung (benutzerdef.)
G 3.3 Nichtbeachtung von Sicherheitsmaßnahmen
G 3.38 Konfigurations- und Bedienungsfehler

**Technisches Versagen**
bG 4.1 Unautorisiertes Tomcat-Shutdown (benutzerdef.)
bG 4.2 Sicherheitsprobleme bei der Ausführung des Tomat-Servers als „root“ oder anderen Accounts mit administrativen Rechten (benutzerdef.)
bG 4.3 Sicherheitslücken im Tomcat-Server (benutzerdef.)
bG 4.4 Sicherheitslücken in der Java-Umgebung
G 4.20 Überlastung von Informationssystemen
G 4.22 Software-Schwachstellen oder –Fehler
G 4.39 Software-Konzeptionsfehler

**Vorsätzliche Handlungen**
G 5.18 Systematisches Ausprobieren von Passwörtern
G 5.28 Verhinderung von Diensten

G 5.48 IP-Spoofing
G 5.143 Man-in-the-Middle-Angriff
G 5.171 Cross-Site Request Forgery (CSRF, XSRF, Session Riding)

## 4.4 Maßnahmenempfehlungen

Zur Absicherung des Tomcat-Servers gegen die Gefährdungen des vorherigen Kapitels, wird in diesem Kapitel eine Reihe von Maßnahmen ausgewählt. Benutzerdefinierte Maßnahmen sind in Klammern mit „(benutzerdefiniert)“ gekennzeichnet. Die Maßnahmen sind im Lebenszyklus-Modell der Grundschutzkataloge gruppiert. Es werden die folgenden Lebenszyklen betrachtet:

- Planung und Konzeption
- Umsetzung
- Betrieb
- Aussonderung
- Notfallvorsorge

**Planung und Konzeption**
bM 2.1 Planung des Tomcat-Einsatzes
bM 2.2 Planung der Tomcat-Server Installation
bM 2.3 Berechtigungen externer Dienstleister
bM 2.4 Beschränkung der Rechte des Tomcat-Server-Prozesses
M 5.10 Restriktive Rechtevergabe

**Umsetzung**
bM 4.1 Shutdown-Schutz
bM 4.3 Absicherung der Administrations-Schnittstellen eines Tomcat-Servers
bM 4.4 Benutzung des Java Security Managers
bM 4.5 Cross-Site Request Forgery (CSRF) Protection in Tomcat
bM 4.6 Deaktivierung des Auto-Deployment-Features
bM 4.7 Absicherung der Tomcat-Manager Anwendung
bM 4.8 Absicherung der Datenbank-Accounts im Tomcat-Server
bM 4.9 Sichere Konfiguration eines Tomcat-Servers
bM 5.1 Sichere Konfiguration von SSL im Tomcat-Server

**Betrieb**
M 2.35 Informationsbeschaffung über Sicherheitslücken des Systems
bM 3.1 Bereithalten von qualifiziertem Administrationspersonal
bM 3.2 Vertretungsregelungen für die Tomcat-Administration
bM 4.2 Sicherer Betrieb eines Tomcat-Servers
bM 4.10 Aktualisierungen der Java-Umgebung

**Notfallvorsorge**
bM 6.1 Notfallvorsorge für einen Tomcat-Server
bM 6.2 Datensicherung einer Tomcat-Server-Instanz

## 5. Erläuterung der Gefährdungen

Im Folgenden werden die benutzerdefinierten Gefährdungen „bG“ definiert. Die anderen Gefährdungen, wie z. B. „G 1.1 Personalausfall“, sind Standard-Gefährdungen, die aus den Grundschutz-Katalogen der 13. Ergänzungslieferung entnommen wurden. Diese Maßnahmen werden entsprechend den Gepflogenheiten für benutzerdefinierte Bausteine hier nicht separat aufgeführt und können unter ihrer Gefährdungs-Nummer in den Grundschutzkatalogen aufgefunden werden.

## bG 1.2 Ausfall einer Tomcat-Instanz

Der Ausfall einer Tomcat-Instanz kann zum Ausfall wichtiger Fachverfahren führen. Ursache für den Ausfall einer Tomcat-Instanz kann unter anderem sein:

- Ausfall der IT-Systeme, die die Tomcat-Instanzen hosten (Verantwortungsbereich: Outsourcing-Dienstleister)
- Ausfall der Infrastruktur, welche die IT-Systeme beherbergt (Verantwortungsbereich: Outsourcing-Dienstleister)
- Logische Fehler bei der Konfiguration der Instanzen (z. B. durch menschliches Versagen oder fahrlässige Handlungen)
- Vorsätzliche Handlungen (z. B. Hacker-Angriff, Sabotage, Diebstahl)

Beispiele:

- Durch ein Update des Server-Betriebssystems werden Softwarekomponenten und Parameter geändert, so dass die Ausführung der Tomcat-Anwendung auf dem Server nicht mehr möglich ist.
- Durch die Wiederherstellung einer alten Datensicherung wird ein nicht mehr aktueller Patch-Level der Tomcat-Instanz restauriert. Infolge dessen fallen einzelne Funktionen der Fachanwendung aus und stehen nicht mehr zur Verfügung.
- Nach einem groß angelegten Hackerangriff wird beschlossen, die Tomcat-Server für eine ausführliche Untersuchung vom Netz zu nehmen. In dieser Zeit stehen keine Dienste zur Verfügung.

## bG 2.1 Verwendung nicht abgesicherter Protokolle zur Administration

Bei der Verwendung nicht verschlüsselter Protokolle (wie z. B. HTTP, Telnet, SQL, rlogin) ist es möglich, dass die Administrationssitzung abgehört und ggf. manipuliert wird. Hierdurch können administrative Passworte offengelegt oder Systeme übernommen werden. Dies gilt auch, wenn die Kommunikationsverbindung nicht Ende-zu-Ende verschlüsselt ist.

Diese Gefahr ist nicht nur in öffentlichen Kommunikationsnetzen relevant, auch im internen Netz sollten administrative Daten nur abgesichert übertragen werden. Hierbei ist zu berücksichtigen, dass ein Außentäter, dem das Eindringen in das interne Netz gelingt zum Innentäter wird. Weiterhin sollte auch die interne Rechtetrennung nicht einfach zu umgehen sein.

Besonders betroffen sind hierbei die folgenden Protokolle:

- das http-Protokoll, das bei der Kommunikation zwischen Webbrowsern und Webservern verwendet wird,
- das telnet-Protokoll, das für Remote Logins verwendet wird,
- das File Transfer Protocol ftp, welches zur Datenübertragung (Downloads) oder zum Login auf Servern benutzt wird,
- das smtp-Protokoll, das zur Übertragung von E-Mail Verwendung findet
- die Protokolle rsh (Remote Shell), rlogin (Remote Login),
- aber auch das SQL-Protokoll, welches für die Kommunikation mit der Datenbank verwendet wird, ist nicht gesichert.

## bG 3.1 Fehlerhafte Administration von Tomcat-Servern

Eine fehlerhafte Administration kann die Sicherheit eines Tomcat-Servers gefährden. Eine fehlerhafte Administration liegt z. B. vor, wenn

- Sicherheitsrichtlinien oder IT-Sicherheitsmaßnahmen missachtet werden,
- die Administration entgegen den Absprachen oder Regelungen erfolgt,
- Administrationseingriffe nicht ausreichend dokumentiert und kommuniziert werden.

Eine fehlerhafte Administration kann sich in einer fehlerhaften oder regelwidrigen Parametrierung des Tomcat-Servers äußern, die zu Sicherheitslücken führen.

Beispiele sind u.a.:

- „vergessene“ Nutzerkennzungen: Nutzerkennung werden nicht entfernt, sobald diese nicht mehr benötigt werden z. B. weil ein Mitarbeiter das Referat wechselt,
- unnötige Nutzerkennzungen: Kennungen die aus der Test- oder Entwicklungsphase stammen bleiben auch in der Produktiv-Installation aktiv,
- unnötige Privilegien für Nutzer oder Prozesse: der Tomcat-Server-Prozess läuft mit administrativen Rechten,
- Nichtauswertung von Protokollen: Protokolle werden nicht regelmäßig ausgewertet, sicherheitskritische Ereignisse werden unbemerkt überschrieben,
- fehlerhaftes, unautorisiertes oder nicht kommuniziertes Deployment von Änderungen oder Anwendungen.

## bG 3.2 Fehlerhafte Einbindung eines Tomcat-Servers in die Systemumgebung

Tomcat-Server können in unterschiedlichen Netzumgebungen eingesetzt werden. Die Absicherung der Server ist auch davon abhängig, ob dieser in einem gut gesicherten internen Netz oder in einer DMZ steht. Für den jeweiligen Aufstellungsort unangemessene Sicherheitsmaßnahmen (z. B. keine ausreichende Systemhärtung bei einer Aufstellung in einer DMZ) können zu Sicherheits-Risiken führen.

Ein weiteres Risiko kann durch eine falsche oder unvollständige Regel auf der Firewall entstehen. Durch diese können unerwünscht viele Ports des Tomcat-Servers von außen erreichbar sein.

Risiken können sich durch eine mangelnde Trennung zwischen internen und externen Tomcat-Servern ergeben. Risiken können auch entstehen, wenn weitere Systeme, wie z. B. Datenbankserver mit dem Tomcat-Server in die DMZ gestellt werden.

Weitere Risiken können durch nicht ausreichend dimensionierte Systemressourcen (z. B. Qualität der Netzanbindung) bestehen.

## bG 4.1 Unautorisierter Tomcat-Shutdown

Standardmäßig wird der Tomcat-Server auf dem TCP-Port 8005 localhost gebunden. Wird zu diesem das Kommando „SHUTDOWN“ gesendet, fährt der Tomcat-Server herunter. Für das Senden dieses Befehls bedarf es keiner administrativen Rechte und es ist keine Authentisierung erforderlich[5]. Es ist allerding ein lokaler Zugang auf den Tomcat-Server nötig.

Ebenfalls ist es möglich dass eine Webanwendung die auf dem Tomcatserver gestartet ist durch den Befehl

*System.exit(0);*

den Tomcat-Server unautorisiert herunterfährt.

[5] Siehe: http://wiki.apache.org/tomcat/FAQ/Security#Q2

## bG 4.2 Sicherheitsprobleme bei Ausführung des Tomat-Servers als „root“ oder anderen Accounts mit administrativen Rechten

Wie bei allen exponierten Servern (insbesondere Webservern), sollte der Tomcat-Server-Prozess nicht mit root-Privilegien laufen. Vergebene root-Privilegien erlauben Angreifern, denen es gelingt den Tomcat-Prozess zu kapern, den ganzen Server zu übernehmen, da Superuser-Rechte erlangt wurden.

Beispiel:
Ein Exploit (Schadcode der für die Ausnutzung einer bestimmten Sicherheitslücke der Tomcat-Server-Software entwickelt wurde) ermöglicht das Einschleusen einer Backdoor auf die Tomcat-Server-Instanz. Läuft diese mit root-Privilegien, besitzt die Backdoor ebenfalls root-Rechte. Diese ermöglichen ihr sämtliche Systemfunktionen auszuführen, in die Konten anderer Nutzer einzubrechen und alle Informationen des Servers auszuwerten. Weiterhin ist es denkbar, dass sich die Backdoor tarnt, indem sie Protokolldateien manipuliert und ihre Erkennung verhindert, indem sie die Antivirensoftware beendet oder manipuliert.

## bG 4.3 Sicherheitslücken im Tomcat-Server

Im Tomcat-Webserver sind in den letzten Jahren eine signifikante Anzahl von Schwachstellen entdeckt worden[6], die durch CERTs und einschlägige Medien publiziert wurden. Obwohl es sich bei Apache Tomcat um eine Open-Source-Software handelt, darf nicht davon ausgegangen werden, dass diese per se sicher ist.

Für nicht gepatchte Software-Schwachstellen werden erfahrungsgemäß früher oder später sogenannte Exploits programmiert. Ein Exploit ist eine Schadsoftware, die eine ganz bestimmte Sicherheitslücke im Tomcat-Server ausnutzt.

[6] http://www.cvedetails.com/product/887/Apache-Tomcat.html?vendor_id=45

## bG 4.4 Sicherheitslücken in der Java-Umgebung

Die Java-Umgebung stellt den Übergang vom Tomcat-Server zum Betriebssystem her. Mit den verschiedenen Start-Mechanismen der Betriebssysteme wir eine Java Virtual Machine gestartet, in welcher der Tomcat-Server zusammen mit den Anwendungen läuft. Der Tomcat-Server selbst ist komplett in Java geschrieben und daher für Sicherheitslücken in der Java-Umgebung anfällig.

In der Java-Umgebung wurden in den letzten Jahren häufig Schwachstellen entdeckt. Die Schwachstellen in der Java-Umgebung waren z. T. signifikanter als die Schwachstellen des Tomcat-Servers selbst.

# 6. Erläuterung der Maßnahmenempfehlungen

In diesem Kapitel werden Maßnahmen mit ihrem Maßnahmentext erläutert, die im Kapitel „Maßnahmenempfehlungen“ aufgeführt wurden. Es werden nur die benutzerdefinierten Maßnahmen erläutert. Maßnahmen, welche nicht mit „(benutzerdefiniert)“ gekennzeichnet sind, sind Maßnahmen, die unverändert den Grundschutzkatalogen entnommen wurden. Sie sind unter ihrer Maßnahmennummer und ihrem Titel in den Grundschutzkatalogen einsehbar.

## bM 2.1 Planung des Tomcat-Einsatzes

Tomcat-Server werden im Rahmen von Fachverfahren (Webanwendungen) benutzt. Daher ist das Fachkonzept für das jeweilige Fachverfahren der Ausgangspunkt der technischen Planung. Bei der Planung des Tomcat-Einsatzes sind die folgenden Punkte zu berücksichtigen:

- Authentisierung und Benutzerverwaltung: Wie findet die Authentisierung statt, z. B. über das ZAD-Verfahren, zentraler Anmeldedienst oder erfolgt eine Authentisierung über das AD? Wie werden neue Nutzer eingepflegt und wie werden diese wieder gelöscht (jeweils technisch und organisatorisch)?
- Rollenkonzept: Existieren im Fachverfahren bestimmte organisatorische oder technische Rollen und haben diese Einfluss auf Benutzergruppen auf dem Tomcat-Server?
- Administration: Wer ist für die Administration der Tomcat-Server verantwortlich? Wie sind Vertretungen geregelt? Welche Tools sind für die Administration zugelassen? Wie werden Änderungen dokumentiert?
- Kommunikationsanalyse (Netzeinbindung): Welche Kommunikationsbeziehungen resultieren aus den Anforderungen? Ist der Einsatz lokaler Paketfilter notwendig?
- Protokollierung: Welche Ereignisse sind zu protokollieren? Wo werden die Protokolle abgelegt? Wie häufig werden die Protokolle ausgewertet?
- Monitoring: Wie erfolgt das Monitoring? Welche Schwellwerte für Almarmierungen sind gesetzt? Wer wird benachrichtigt?
- Load-Balancing: Ist ein Load-balancing geplant? Wie ist dieses zu realisieren?

Prüffragen:

- Existiert eine Planungsdokumentation für den sicheren Einsatz des Tomcat-Servers?
- Existieren die in der Maßnahme angesprochenen Teilkonzepte?

## bM 2.2 Planung der Tomcat-Server Installation

Ausgehend vom Fachkonzept für das jeweilige Fachverfahren und der Planung des Tomcat-Einsatzes (bM 2.1) ist die Installation des Tomcat-Servers zu planen. Hierbei sind insbesondere folgende Aspekte zu berücksichtigen:

- Partition und Dateisystemlayout: Welche Partitionen werden benötigt? Ist eine eigene Partition für die Tomcat-Anwendungen empfehlenswert? Welche Ressourcen werden benötigt? Wie wird der Füllstand der Partitionen überwacht?
- Funktionen und benötigte Module: Welche Funktionen und Schnittstellen werden im Tomcat-Server benötigt (z. B. für Anbindung an externe Datenquellen und Dienste)? Nicht benötigte Dienste sind zu deaktivieren.
- Betrieb als Standalone-Webserver: Soll Tomcat als separater Webserver eingesetzt werden? Wenn ja, ist die Konfiguration des HTTP-Connector zu planen. Sind ein oder mehrere Connectoren nötig? Welche Ports sind zu freizuschalten? Wird Proxy- oder SSL-Support benötigt? Welche Parameter für den Connector sind zu setzen? (Siehe auch: http://tomcat.apache.org/tomcat-8.0-doc/config/http.html)
- Betrieb mit separatem Webserver: Soll der Tomcat-Server lediglich als Servlet-Container dienen und ein separater Webserver (z. B. Apache) benutzt werden? Wenn ja, ist die Konfiguration des AJP-Connectors (oder eines anderen Connectors) zu planen. Sollen Load-Balancing eingesetzt werden? Welche Parameter für den Connector sind zu setzen?
  (Siehe auch: http://tomcat.apache.org/tomcat-8.0-doc/config/ajp.html)
- Netzeinbindung: Wie erfolgt die Einbindung in das Netz (dedizierte DMZ, geschütztes Intranet). Wie erfolgt die Kommunikation mit anderen relevanten Servern (weitere Webserver, Datenbankserver, etc.). Befinden sich diese Server in einem anderen Netzsegment? Wenn ja existieren Regelungen für diesen Netzübergang?

Prüffragen:

- Existiert ein Installationskonzept, welches alle Vorgaben und Planungen berücksichtigt?
- Ist geregelt, ob Tomcat als Standalone-Webserver oder mit einem separaten Webserver betrieben werden soll?
- Existiert eine Planung der Netzeinbindung des Tomcat-Servers?

## bM 2.3 Berechtigungen externer Dienstleister

Bei der Berechtigungsplanungsplanung sind auch die externen Dienstleister zu berücksichtigen. Externe Dienstleister nehmen z.T. fachliche Aufgaben in Entwicklung und Implementierung war, wie dies auch interne Mitarbeiter tun. Hierdurch bekleiden sie wichtige Positionen und sind komplett in Prozesse und Arbeitsabläufe einbezogen. Für die Aufgabenerfüllung müssen sie angemessene Berechtigungen erhalten. Hinzu kommt, dass externe Mitarbeiter z.T. nicht immer als Extern wahrgenommen werden. Dennoch sollte berücksichtigt werden, dass externe Dienstleistungsverhältnisse zeitlich beschränkt sind.

Die Berechtigungen externer Mitarbeiter sind bei der Planung des Tomcat-Servers zu berücksichtigen. Auch für externe Mitarbeiter sollte der Grundsatz gelten, dass nur die für die Aufgabenerfüllung notwendigen Rechte vergeben werden.

Beim Ausscheiden externer Mitarbeiter sollte bedacht werden, dass alle Passworte, die dem betreffenden Mitarbeiter bekannt geworden sind, zu ändern sind. Aus diesem Grund sollte grundsätzlich mit personenbezogenen Accounts gearbeitet werden. Root-Accounts sollten gesichert hinterlegt und bei Benutzung geändert werden.

Prüffragen:

- Werden bei der Berechtigungsplanung externe Mitarbeiter berücksichtigt?
- Werden grundsätzlich personenbezogene Accounts verwendet?
- Werden beim Ausscheiden externer Mitarbeiter die betreffenden Accounts deaktiviert?
- Werden Root-Accounts beim Ausscheiden von Mitarbeitern gewechselt?

## bM 2.4 Beschränkung der Rechte des Tomcat-Server-Prozesses

Der Tomcat-Server-Prozess sollte nur mit minimalen Rechten ausgestattet sein, um Auswirkungen von Angriffen zu minimieren. Tomcat sollte auf keinen Fall als root-User laufen. Es empfiehlt sich einen eigenen Nutzer, z. B. „tomcat“ und eine eigene Gruppe anzulegen und nur mit eingeschränkten Rechten auszustatten. Es sollte z. B. nicht möglich sein, sich mit dem Tomcat-User remote einzuloggen.

Die Rechteeinschränkung kann zu Problemen bei der Abfrage privilegierter Ports führen. Als Gegenmaßnahme kann hier der Einsatz eines Reverse-Proxies, wie z. B. HTTPD erwogen werden.

Die Dateiberechtigungen sollten ebenfalls restriktiv gesetzt werden. Der Eigentümer (owner) sollte Read/Write-Rechte, die Gruppe (group) nur Leserechte und andere (world) sollte keine Rechte an den Dateien haben. Für den Tomcat-Prozess empfiehlt sich ein umask von 007.

Prüffragen:

- Wurden für die Ausführung des Tomcat-Servers ein eigner Nutzer mit eingeschränkten Rechten benutzt?
- Wurden die Dateiberechtigungen eingeschränkt?

## bM 3.1 Bereithalten von qualifiziertem Administrationspersonal

Um den sicheren Betrieb der Tomcat-Server garantieren zu können muss ausreichend qualifiziertes Personal in ausreichender Personalstärke bereitgestellt werden.

Ausreichend qualifiziertes Personal bedeutet hierbei, dass die Administratoren die Tomcat-Umgebung sicher beherrschen können müssen und über ausreichend Praxiserfahrung verfügen, um einen sicheren Betrieb bewerkstelligen zu können. Weiterhin ist eine Einweisung nötig, wenn neue Mitarbeiter diese Aufgabe zum ersten Mal übernehmen. Bei der Einweisung ist insbesondere die speziellen betrieblichen und sicherheitstechnischen Rahmenbedingungen, die Art der Dokumentation, das Change-Management und die Abstimmung im täglichen Betrieb, sowie die Notfallvorsorge zu thematisieren.

Ausreichend Personalstärke bedeutet, dass auch im Fall von Krankheit und Urlaub noch eine ausreichende Administration sichergestellt werden kann.

Prüffragen:

- Wurden neue Mitarbeiter in die speziellen betrieblichen und sicherheitstechnischen Rahmenbedingungen eingewiesen?
- Kann auch im Fall von Krankheit und Urlaub noch eine ausreichende Administration sichergestellt werden?

## bM 3.2 Vertretungsregelungen für die Tomcat-Administration

Neben dem Bereithalten von ausreichend qualifiziertem Personal, muss auch eine gegenseitige Vertretung möglich und geregelt sein. Bei der Regelung der Vertretung ist insbesondere auf folgende Punkte zu achten:

- Vertretungsregelungen müssen formal (z. B. schriftlich) festgelegt sein.
- Beim vertretenden Administrator müssen ausreichende Kenntnisse über die jeweilige Tomcat-Instanz bestehen, für die eine Vertretung erfolgen soll. Hierbei sind insbesondere eine möglichst aktuelle und transparente Dokumentationen des Systems und Einweisungen durch den betreuenden Kollegen hilfreich.
- Administrative Eingriffe an Tomcat-Servern müssen dokumentiert werden, um auch nach Beendigung der Vertretung eine konsistente Administration sicherstellen zu können.
- Die Administration sollte mit personenbezogenen Kennungen durchgeführt werden, um Aktionen konkreten Personen zuordnen zu können.
- Für den Notfall sollte die Hinterlegung der Root-Kennung geregelt werden.

Prüffragen:

- Sind Vertretungsregelungen schriftlich fixiert?
- Werden administrative Eingriffe an den Tomcat-Servern dokumentiert?
- Ist die Hinterlegung der Root-Kennung geregelt?

## bM 4.1 Shutdown-Schutz

Zum Schutz des Tomcat-Server vor unautorisiertem Herunterfahren sind (alternativ) mehrere Maßnahmen möglich:

**Möglichkeit 1**: Ändern des Port-Parameters der Tomcat-Konfiguration:
*<Server port="VALUE" .../>*
Wird der Wert auf „-1“ gesetzt, ist ein Shutdown nur noch per Terminal durch einen entsprechend privilegierten Benutzer möglich.

**Möglichkeit 2**: Modifikation der await() Methode
Die Methode kann mit einer Authentisierung erweitert oder aber deaktiviert werden.

**Möglichkeit 3**: Nutzung eines Paketfilters auf dem jeweiligen Server

Prüffragen:

- Sind die Tomcat-Server vor einem unautorisiertem Shutdown geschützt?

## bM 4.2 Sicherer Betrieb eines Tomcat-Servers

Der sichere Betrieb eines Tomcat-Servers sollte in einer Betriebsdokumentation (z. B. Betriebshandbuch) definiert werden. Hierbei sind die folgenden Aspekte zu berücksichtigen:

- Verantwortliche Administratoren für den Tomcat-Server
- Zugelassene Administrationswerkzeuge
- Dokumentation von Änderungen am Tomcat-Server
- Einspielen von Sicherheitsupdates und Patches
- Dokumentation der auf dem Tomcat-Server aktivierten Dienste
- Dokumentation verwendeter Ports
- Dokumentation der Installationsanleitung (oder Verweis)
- Protokollierung von Systemereignissen und regelmäßige Auswertung der Protokolle
- Art der Datensicherung, Vorgehensweise bei einem Restore
- Vorgehensweise bei einem Notfall

Es ist wichtig, dass alle in Betrieb befindlichen Tomcat-Instanzen dokumentiert werden. Dies ist insbesondere dann wichtig, wenn unterschiedliche Fachverfahren auf eine Vielzahl von Instanzen zugreifen und diese z. T. auch virtualisiert sind. Für die Erfassung der Instanzen sollten zumindest die folgenden Informationen erhoben werden:

- Name der Tomcat-Instanz
- Verwendungszweck (Fachverfahren, Dienst, etc.)
- Kontext/Domäne (falls zutreffend)
- Ggf. URL
- Ansprechpartner (fachlich, betrieblich)
- Letzte Änderung
- Server auf dem die Instanz liegt (physikalisch oder virtuell)

Die folgenden Einstellungen sollten für einen sicheren Betrieb berücksichtigt werden:

- Es sollte ein AccessLogValve definiert werden, um Zugriffe auf den Server zu dokumentieren.
- Sollen verlinkte Dateien deployt werden, soll auf dem Context-Tag das Attribut aliases statt allowLinking gesetzt werden, da ansonsten im Falle eines Undeployments den Links gefolgt wird und die verlinkten Dateien gelöscht werden.
- Um Serverinformationen, insbesondere die Versionsnummer zu verbergen sollte eine Datei $CATALINA_BASE/lib/org/apache/catalina/util/ServerInfo.properties mit folgendem angelegt werden:

  *server.info=Apache Tomcat/8.0.x*

Hiermit wird verhindert, dass im standardmäßig verwendeten ErrorReportValve die genaue Tomcat-Version angezeigt wird.
Zudem sollen Webanwendungen in ihrer *web.xml* anwendungsspezifische Fehlerseiten konfigurieren, da ansonsten der Default ErrorReportValve den Stacktrace anzeigt.
Will man sogar vermeiden, dass das Serverprodukt angezeigt wird, so muss man zudem im Connector das Attribut *server* setzen, damit nicht Apache-Coyote/1.1 als Standard Server-Header im http-Response gesetzt wird. (Zitat aus der Tomcat-Dokumentation: “Unless you are paranoid, you won't need this feature.”)

- Nutzen Webanwendungen Container-Managed Security für ihre Benutzer- und Rollenverwaltung sollte man ggf. über den Einsatz des sog. LockOutRealms nachdenken, welcher Benutzer nach einer gewissen Anzahl (Standard: 5) an fehlerhaften Login-Versuchen für eine bestimmte Zeit (Standard 5 Minuten) sperrt.

Prüffragen:

- Existiert eine Betriebsdokumentation für den sicheren Betrieb der Tomcat-Server?
- Ist sichergestellt, dass alle in Betrieb befindlichen Tomcat-Instanzen dokumentiert wurden?
- Ist diese Dokumentation aktuell?

## bM 4.3 Absicherung der Administrations-Schnittstellen eines Tomcat-Servers

Die administrativen Schnittstellen zur Administration des Tomcat-Servers müssen abgesichert werden. Die Tomcat-Instanzen werden derzeit hauptsächlich mit SSH administriert. Durch die SSH-Administration ist die verschlüsselte Übertragung der Administrations-Kommandos sichergestellt.
Als weitere Tools wird der Tomcat-Manager und VisualVM eingesetzt. Auch die Kommunikationsverbindungen dieser Administrations- und Überwachungs-Tools sollten verschlüsselt werden.
Ebenso sind alle weiteren verwendeten Administrations-Werkzeuge abzusichern. Hier muss der Grundsatz gelten, dass Authentisierungsdaten nicht im Klartext übertragen werden, auch nicht im internen LAN.

**Benutzen vorhandener sicherer Protokolle**
Soweit in der Administrationssoftware vorhanden, sollten sichere Protokolle (wie z. B. https anstelle von http) eingesetzt werden. Auch die gezielte Auswahl entsprechender Administrations-Tools mit sicheren Kommunikationsschnittstellen (wie z. B. ssh) kann hier hilfreich sein.

**Kapselung der Kommunikation**
Sollten die verwendeten Administrationstools keine starke Verschlüsselung beherrschen, sollte überlegt werden, die Administrationssitzung mit in einem sicheren Protokoll (z. B. IPsec) zu kapseln. Alternativ wäre hier die Verwendung dedizierter VLANs möglich.

Prüffragen:

- Werden die Tomcat-Instanzen mit Hilfe verschlüsselter Protokolle administriert?
- Ist die Nutzung schwacher oder unverschlüsselter Protokolle weiterhin möglich?
- Sind alle administrativen Schnittstellen aller Tomcat-Server abgesichert?

## bM 4.4 Benutzung des Java Security Managers

Der Java Security Manager erlaubt es, virtuelle Maschinen voneinander abzuschotten, den Zugriff auf Ressourcen des Rechners für jede Maschine zu steuern und Webanwendungen sinnvoll einzuschränken. Für die Benutzung des Security Managers ist die Definition einer Policy für die Java Virtual Machine (JVM) Voraussetzung.

**Tomcat Security-Policy**
Tomcat bringt bereits seine Security-Policy mit, welche sich im Verzeichnis *conf* in der Datei *catalina.policy* befindet. Durch den Security Manager lassen sich die Rechte (Lesen/Schreiben von Properties, Dateisystemzugriffe, Socket-Zugriffe) für jede Klasse steuern.

Zur Aktivierung des Security Managers dient der Befehl:
*java -cp ... -Djava.security.Manager -Djava.security.policy=Pfad_der_Policy-Datei bin/startup.sh –security* für den Start des Tomcat-Servers mit aktivierten Security Manager

**Policy-Regeln definieren**
Achtung: Nach Aktivierung Security des Managers darf eine Klasse gar nichts mehr. Es ist daher nötig entsprechende Regeln zu definieren. Eine Dokumentation für die Definition von Regeln findet sich hier:
http://docs.oracle.com/javase/1.4.2/docs/guide/security/PolicyFiles.html

Sollen z. B. für das Verzeichnis lib der Standard-Java-Installation und alle rekursiven Dateien alle Rechte freigegeben werden, lautet die Regel folgendermaßen:

```
grant codeBase “file:${java.home}/lib/-“
        {permission java.security.AllPermission;};
```

**Best-Practices**
Bei der Konfiguration eines Policy-Sets für den Security Managers werden häufig die folgenden Best-Practices genannt:

- Webanwendungen dürfen nur Dateien in ihrem eigenen ${docBase} lesen.
- Webanwendungen sind ausschließlich im ${docBase} und nicht verstreut über das Dateisystem zu halten.
- Protokolldateien (AccessLog und SystemLog) werden für jeden virtuellen Host in einem speziellen Verzeichnis logs geschrieben, das außerhalb des ${docBase} liegt.
- Es dürfen nur Daten der eigenen Anwendung verändert werden.

- Temporäre Dateien werden in einen dafür vorgesehen Ordner geschrieben und gelesen.
- Systemeinstellung sollen nicht grundlos und undokumentiert verändert werden.

Diese sollten bei der Erstellung eigener Policies berücksichtigt werden.

**Vor Inbetriebnahme testen**
Zu beachten ist, dass Security Policies vor der produktiven Verwendung auf Funktionalität und Performance zu testen sind.

Prüffragen:

- Wird der Java Security Manager eingesetzt?
- Wurde Policy-Regeln für den Java Security Manager definiert?
- Wurde die Tomcat Security-Policy „*catalina.policy“* entsprechend den eigenen Erfordernissen und Sicherheitsanforderungen angepasst?
- Wurde die Policy vor der der produktiven Verwendung auf Interoperabilität und Performance getestet?
- Werden Änderungen an den Policy-Regeln abgestimmt und dokumentiert?

## bM 4.5 Cross-Site Request Forgery (CSRF) Protection in Tomcat

Tomcat bietet standardmäßig ab der Version 7 einen Token-basierten CSRF-Schutz für den Tomcat Manager und die Tomcat Host-Anwendungen. Hierfür wird ein von Tomcat ein CSRF-Filter angeboten, der von der Webanwendung genutzt werden kann, um entsprechende Tokens an den Web-Client zu senden. Durch die Token-Prüfung werden CSRF-Attacken verhindert.

Der CSRF-Schutz muss allerdings aktiv in der Webanwendung implementiert werden. Die Verwendung des CSRF-Schutzes empfiehlt sich insbesondere für diejenigen Webanwendungen, welche eine Authentisierung durchführen.

Weiterhin sollte die Gültigkeit der Sitzungsdauer in einer Webanwendung begrenzt werden, da hierdurch die Angriffsfläche verkleinert werden kann. Hierfür eignet sich die Funktion *session-timeout* für Tomcat.

Beispiel für 10 Minuten Timeout:
*<session-config>*
*<session-timeout>10</session-timeout>*
*</session-config>*

**Alternative WAF**
Eine Alternative zur Integration eines CSRF-Schutzes in die jeweilige Webanwendung ist der Einsatz einer Web Application Firewall (WAF). Eine Web Application Firewall ist in der Lage den http-Protokoll-Strom nach bekannten Angriffssignaturen zu durchsuchen und entsprechende Angriffe zu filtern.

Eine Web Application Firewall ist somit in der Lage mehrere Webanwendungen zu schützen. Auch ist der Schutz von Webanwendungen möglich, die nicht mehr aktualisiert werden (können). Ein Nachteil besteht darin, dass nur bekannte Angriffsmuster erkannt werden können. Somit müssen auch die Angriffssignaturen der WAF immer wieder aktualisiert werden.

Es ist aber zu beachten, dass der Einsatz einer WAF immer anwendungsspezifisch zu planen und zu implementieren ist.

Prüffragen:

- Wird CSRF-Schutz in alle Webanwendungen implementiert, welche eine Authentisierung durchführen?
- Wurde alternativ der Einsatz einer Web Application Firewall (WAF) geprüft und ggf. anwendungsspezifisch geplant?

## bM 4.6 Deaktivierung des Auto-Deployment-Features

Jeder Tomcat-Server unterstützt 3 Auto-Deployment-Szenarien:

- autoDeploy,
- DeployOnStartup,
- deployXML.

Für die Entwicklungsumgebung sind die Auto-Deployment-Features wichtige Arbeitsmittel. In der Produktiv-Umgebung können sie zusätzliche Sicherheitsprobleme bereiten. Es ist daher überlegenswert, das Auto-Deployment zu deaktivieren[7].

**Deployment-Prozess regeln**
Davon unabhängig sollte der Deployment-Prozess geregelt ablaufen. Bei einem Deployment handelt es sich um die Verteilung von selbstentwickelter Verfahrenssoftware (Individualsoftware) auf die entsprechenden IT-Systeme. Bei der Regelung dieses Prozesses können auch einzelne Aspekte aus dem Baustein „Standardsoftware“ entlehnt werden.

Es ist festzulegen, wie die folgenden Aspekte im Deployment erfolgen:

- Beantragung von Änderung
- Freigabe einer Änderung
- Realisierung einer Änderung
- Testen der Realisierung
- Freigabe der Realisierung
- Dokumentation der Änderung auf den betreffenden Systemen

Prüffragen:

- Ist festgelegt und dokumentiert, wie das Auto-Deployment-Feature in Tomcat zu konfigurieren ist?
- Ist der Deployment-Prozess geregelt?

[7] http://www.mulesoft.com/improving-apache-tomcat-security-step-step-guide

## bM 4.7 Absicherung der Tomcat-Manager Anwendung

Die Tomcat-Manager Anwendung ist eine web-basierte administrative Anwendung, mit der die Tomcat-Instanzen, das Deployment von Anwendungen und andere Einstellungen auf dem Tomcat-Server gesteuert werden können.

Standardmäßig ist der Tomcat-Manager deaktiviert. Sollte er für die Administration nicht benötigt werden, sollte diese Einstellung aus Sicherheitsgründen beibehalten werden. Wird der Tomcat-Manager eingesetzt sind die folgenden Sicherheitseinstellungen erforderlich[8]:

- Zugriff beschränken: Der Zugriff auf das Tool sollte auf die erforderlichen Personen beschränkte werden.
- Zur Zugriffsbeschränkungen können autorisierte IP-Adressen mittels
  *RemoteAddrValve* vergeben werden
- Eine Beschränkung von Passwort-Eingaben ist mittels LockOut Realm[9] möglich.
- Zur Authentisierung ist die Verwendung starker und komplexer Passworte zu erzwingen.
- Ab Tomcat 7 können in der Tomcat-Manager Anwendung granularere Rechte vergeben werden.

Beispiel für Zugriffsbeschränkungen auf definierte IP-Adressen:
*$CATALINA_HOME/conf/Catalina/localhost/manager.xml*
*<Valve className="org.apache.catalina.valves.RemoteAddrValve" allow="127.0.0.1,192.168.*.*"*
*/>*

Prüffragen:

- Ist die Tomcat-Manager Anwendung abgesichert?
- Sind Zugriffsbeschränkungen auf die Tomcat-Manager Anwendung eingerichtet?

[8] http://www.mulesoft.com/improving-apache-tomcat-security-step-step-guide
[9] http://tomcat.apache.org/tomcat-8.0-doc/config/realm.html#LockOut_Realm_-_org.apache.catalina.realm.LockOutRealm

## bM 4.8 Absicherung der Datenbank-Accounts im Tomcat-Server

In der Maßnahme M 2.129 Zugriffskontrolle einer Datenbank des Baustein B 5.7 Datenbank wird gefordert, dass der Zugriff auf Objekte der Datenbank je Anwendung mit eigens hierfür einzurichtenden Kennungen erfolgt. Diese Kennungen müssen dann auf den jeweiligen Fachanwendungen oder Tomcat-Instanzen hinterlegt werden. Allerdings muss diese Hinterlegung gesichert erfolgen.

Die Absicherung der hinterlegten Passworte muss mit großer Sorgfalt erfolgen, da durch ein unautorisiertes Bekanntwerden Zugriff auf die Fachdatenbank möglich werden und hierdurch große Schäden entstehen können. Es darf nicht möglich sein, dass Unbefugte auf die hinterlegten Passwörter Zugriff nehmen.

Die Datei, welche die hinterlegten Passworte enthält muss mit Zugriffsrechten so abgesichert werden, dass nur der autorisierte technische Prozess auf diese Datei Zugriff besitzt. Andere Nutzer oder Gruppen dürfen keinen Zugriff besitzen. Es ist zu beachten, dass sich diese Restriktion sowohl auf die Schreib-, als auch auf die Lese-Rechte erstrecken muss.

Es sollte überlegt werden (optional), Passworte auf dem Tomcat-Server verschlüsselt zu hinterlegen.

Die Übertragung der Authentisierungsdaten vom Tomcat-Server zum Datenbank-Server muss in jedem Fall verschlüsselt erfolgen, um unautorisiertes Mittschneiden der Account-Daten zu verhindern. Hierfür bietet sich die Benutzung entsprechender JDBC-Funktionalitäten an.

Prüffragen:

- Ist die Datei mit den hinterlegten Datenbank-Accounts ausreichend gegen unbefugtes Auslesen oder Schreiben geschützt?
- Werden die Datenbank-Accounts vom Tomcat-Server zum Datenbankserver verschlüsselt übertragen?

## bM 4.9 Sichere Konfiguration eines Tomcat-Servers

Die folgenden Schritte sollten nach einer erfolgten Tomcat Installation erfolgen:

- Zur Überprüfung von Zertifikaten verwendet der in JAVA geschriebene Tomca-Server den JAVA Zertifikatspeicher, der in der Datei *${JAVA_HOME}/jre/lib/security/cacerts* abgelegt ist. Das Standardpasswort *changeit* des Zertifikatspeichers sollte neu gesetzt werden.
- Alternativ können die Zertifikate der Certification Authorities (CAs), im Tomcat-Zertifikatspeicher abgelegt werden, der mit den zusätzlichen Optionen *truststoreFile* und ggf. *truststorePass* in der <Connector>-Deklaration der Konfigurationsdatei
  *$CATALINA_HOME/conf/server.xml* spezifiziert wird.
  Mittels *javax.net.ssl.trustStorePassword* sollte dieser Zertifikatsspeicher ebenfalls passwortgesichert werden.

**Protokollierung**
Für den Produktiveinsatz des Tomcat-Servers ist zu prüfen, welche Log-Informationen für den Betrieb und zur Fehlersuche tatsächlich benötigt werden, um übermäßigen Ressourcen-Verbrauch zu vermeiden. Das Logging kann unterschieden werden nach:

- dem Logging auf der Ebene der Betriebszustände von Tomcat auf der Stufe der Tomcat-Engine, der virtuellen Hosts, etc. (z. B. Hoch- oder Herunterfahren von Tomcat oder Webanwendungen, Änderung der Konfiguration, Protokollierung von Fehlern zur Laufzeit)
- dem Logging der Zugriffe auf die bereitgestellten Webanwendungen

Im Produktivsystem sollten die *debug* Attribute auf 0 gesetzt und das
*AccessLogValve* nur bei Bedarf eingesetzt werden[10]. Datenschutzrechtliche Aspekte bei der Protokollierung sind zu beachten.

Prüffragen:

- Ist der Java Zertifikatspeicher mittels Passwort abgesichert?
- Wurden für den Produktiveinsatz die Protokollierungseinstellungen angepasst?

[10] Weitere Information sind in „BSI: Sicherheitsuntersuchung des Apache Jakarta Tomcat Servlet Containers“, Kapitel 6.3.3 Logging verfügbar.

## bM 4.10 Aktualisierungen der Java-Umgebung

Die Java-Laufzeitumgebung (Java Runtime Environment) stellt die Laufzeitumgebung für den Tomcat-Server und dessen Anwendungen bereit. Schwachstellen in der Java-Umgebung könnten sich auf die Sicherheit der gesamten Tomcat-Umgebung auswirken. Von daher ist eine regelmäßige Aktualisierung der Java-Laufzeitumgebung empfehlenswert. Bei der Aktualisierung sind allerdings einige Punkte zu beachten:

- Erfolgt das Update innerhalb einer Hauptversion oder soll eine neue Hauptversion verwendet werden?
- Unterstützen die eingesetzten Web-Anwendungen die neue Version der Java-Umgebung?
- Welche Java-Erweiterungen sollen genutzt werden? Werden durch die neue Version Erweiterungen obsolet?

Prüffragen:

- Werden die eingesetzten Web-Anwendungen vor einer Aktualisierung der Java-Umgebung getestet?
- Werden nicht genutzte oder obsolete Java-Erweiterungen deaktiviert?

## bM 5.1 Sichere Konfiguration von SSL im Tomcat-Server

Tomcat unterstützt 2 unterschiedliche SSL-Implementierungen:

- die JSSE Implementierung, die Teil der Java Runtime (seit Version 1.4) ist und
- die APR[11] Implementierung, welche standardmäßig OpenSSL nutzt.

Die Implementierung kann gewählt werden. Wahl des Java (JSSE) Connectors:

Beispiel JSSE:
*<Connector protocol="org.apache.coyote.http11.Http11Protocol"*
*port="8443" .../>*

Wahl des ARP Connectors:

Beispiel ARP:
*<Connector protocol="org.apache.coyote.http11.Http11AprProtocol"*
*port="8443" .../>*

Das *SSLEngine* Attribut darf **nicht** „off“ sein!

Zum Schluss ist der Connector zu konfigurieren:
Datei: *$CATALINA_BASE/conf/server.xml*

Beispiel JSSE:
*<Connector*
*protocol="HTTP/1.1"*
*port="8443" maxThreads="200"*
*scheme="https" secure="true" SSLEnabled="true"*
*keystoreFile="${user.home}/.keystore" keystorePass="<Passwort>"*
*clientAuth="false" sslProtocol="TLS"/>*

Beispiel APR:
*<Connector*
*protocol="HTTP/1.1"*
*port="8443" maxThreads="200"*
*scheme="https" secure="true" SSLEnabled="true"*
*SSLCertificateFile="/usr/local/ssl/server.crt"*
*SSLCertificateKeyFile="/usr/local/ssl/server.pem"*
*SSLVerifyClient="optional" SSLProtocol="TLSv1"/>*

[11] Apache Portable Runtime: siehe http://tomcat.apache.org/tomcat-8.0-doc/apr.html

Die Installation des, ggf. von einer Certification Authority signierten, Server-Zertifikats in das angegebene keystoreFile erfolgt mit dem Java-Werkzeug keytool[12].

Weiterhin ist zu berücksichtigen, dass ohne weitere Maßnahmen die Authentisierungsinformationen bei der Nutzung von JDBC Realm- bzw. JNDI Realm-basierter Authentisierung im Klartext übermittelt werden können[13]. Dies kommt zum Tragen:

- bei Verwendung des HTTP-Connectors zwischen Tomcat und der Datenbank bzw. dem LDAP-Server,
- bei Verwendung des AJP-Connectors, darüber hinaus auch zwischen Apache / mod_jk und Tomcat,

Zur Absicherung empfiehlt sich ein Port-Forwarding per Secure Shell (SSH) oder alternativ eine IPsec-Verschlüsselung.

Im SSL-Connector sollten mit dem Attribut *ciphers* nur starke Krypto-Algorithmen eingetragen werden. Dadurch werden andere schwächere Ciphers deaktiviert. Dies sollte vor allem im DMZ-Umfeld beachtet werden, wenn Tomcat als Standalone-Server verwendet wird. Allerdings weist das BSI in seiner Studie[14] darauf hin, dass bei Tippfehlern oder fehlerhaften Angaben die Eintragungen ohne Fehlermeldung ignoriert werden. Eine Liste aller validen Chiffren ist hier[15] erhältlich. Eine Prüfung der eingesetzten Ciphers ist z. B. mit dem freien Tool OWASP O-Saft[16] möglich.

Prüffragen:

- Wurde die SSL-Konfiguration im Rahmen eines Kryptokonzepts geplant?

[12] Siehe: http://tomcat.apache.org/tomcat-8.0-doc/ssl-howto.html#Installing_a_Certificate_from_a_Certificate_Authority
[13] Siehe „BSI: Sicherheitsuntersuchung des Apache Jakarta Tomcat Servlet Containers“, Kapitel 6.3.7 Verschlüsselte Verbindung (SSL).
[14] Siehe „BSI: Sicherheitsuntersuchung des Apache Jakarta Tomcat Servlet Containers“, Kapitel 5.1.2.6.3
[15] Siehe „BSI: Sicherheitsuntersuchung des Apache Jakarta Tomcat Servlet Containers“, Listing 47: mögliche Angaben zu ciphers in conf/server.xml
[16] https://www.owasp.org/index.php/O-Saft

## bM 6.1 Notfallvorsorge für einen Tomcat-Server

Der Ausfall eines Tomcat-Servers als integraler Bestandteil eines Fachverfahrens kann signifikante Auswirkungen haben. Daher ist eine Vorgehensweise für den Notfall wichtig. Im Rahmen der Notfallvorsorge ist zu betrachten, wie die Folgen eines Ausfalls minimiert werden können und welche Aktivitäten im Falle eines Ausfalls durchzuführen sind.

Hierfür kann auf die Ergebnisse einer Business Impact Analyse aus dem Baustein B 1.3 zurückgreifen.

Folgende Aspekte müssen dabei berücksichtigt werden:

- Notfallplanung für die Tomcat-Instanz
- Notfallplanung für den physikalischen Server, auf dem die Instanz läuft (verantwortlich: Outsourcing-Dienstleister)
- Datensicherung der Tomcat-Instanz
- Wiederanlaufplanung mit Reihenfolge der Wiederinbetriebnahme
- Regelmäßige Notfallübungen

Prüffragen:

- Existiert ein Notfallplan für die Tomcat-Server-Instanzen, der die fachlichen Verfügbarkeitsanforderungen der Fachanwendung berücksichtigt?
- Existieren Notfallpläne für die physikalischen Server?
- Existiert ein Datensicherungsplan?

## bM 6.2 Datensicherung einer Tomcat-Server-Instanz

Die Datensicherung der einzelnen Tomcat-Instanzen ist zu konzeptionieren. Das Konzept muss sicherstellen können, dass die Fachanwendung zeitgerecht wiederhergestellt werden können. In der Konzeption ist zu berücksichtigen:

- Verfügbarkeitsanforderungen der Webanwendungen (u.a. besondere Auszahlungstermine)
- Datenvolumen für die Datensicherung
- Änderungsvolumen pro Tag/Woche/Monat

Für die Datensicherung jeder Tomcat-Instanz ist ein Sicherungsplan zu dokumentieren. Dieser Sicherungsplan sollte die folgenden Angaben enthalten:

- Art der Datensicherung (Voll, Inkrementell, etc.)
- Intervall der Datensicherung
- Zurückspielen der Datensicherung (Recovery der Daten)
- Vorhaltefrist der Datensicherung

Prüffragen:

- Existiert ein Datensicherungskonzept für die Tomcat-Instanzen?
- Sind in diesem Datensicherungskonzept die maximal tolerierbaren Ausfallzeiten der zugrundeliegenden Fachanwendungen berücksichtigt?
- Sind die Art und der Umfang der Datensicherung dokumentiert?
- Wurde ein fehlerfreies Zurückspielen der Datensicherung (Recovery) getestet?

## 7. Kreuzreferenztabelle

Die Kreuzreferenztabelle stellt dar, welche Maßnahme(n) gegen welche Gefährdung(en) wirkt.

| Zyklus/Siegel | | | bG 1.2 | G 1.1 | bG 2.1 | G 2.7 | G 2.9 | G 2.111 | bG 3.1 | bG 3.2 | G 3.3 | G 3.38 | bG 4.1 | bG 4.2 | bG 4.3 | bG 4.4 | G 4.20 | G 4.22 | G 4.39 | G 5.18 | G 5.28 | G 5.48 | G 5.143 | G 5.171 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| bM 2.1 | PK | - | | | X | X | X | | | X | | | | | | | X | | X | | | X | | |
| bM 2.2 | PK | - | | | | | | X | X | | | | | | | | | | | | | | X | |
| bM 2.3 | PK | - | | | | X | X | X | X | | X | | | | | | | | | | | | | |
| bM 2.4 | PK | - | X | | | X | | | | | | X | | X | X | X | | X | X | | X | | | X |
| M 2.35 | BT | - | | | | | | | | | | | | | X | X | | X | X | | | | | |
| bM 3.1 | BT | - | | X | | | | | X | | X | | | | | | | | | X | | | | |
| bM 3.2 | BT | - | | X | | | | | X | | | | | | | | | | | | | | | |
| bM 4.1 | UM | - | | | | | | | | X | | | X | | | | | | | | X | | X | |
| bM 4.2 | BT | - | | | | | X | X | X | X | X | X | | | X | X | X | X | X | | X | | X | |
| bM 4.3 | UM | - | | | X | X | | | | X | | | | | | | | | | | X | X | X | |
| bM 4.4 | UM | - | X | | | X | | | | X | | | X | | | | X | | | | X | | | X |
| bM 4.5 | UM | - | X | | | X | | | | X | | | | | | | X | | X | | X | | X | X |
| bM 4.6 | UM | - | | | | X | | | X | | | X | | | | | | | | | X | | | |
| bM 4.7 | UM | - | | | | X | | | | | | X | | | | | | | | X | X | X | | |
| bM 4.8 | UM | - | X | | | X | | X | | | | | | | | | | | | | | | X | |
| bM 4.9 | UM | | X | | | X | | | | | | X | | | | | | | | | | | | |
| bM 4.10 | BT | | | | | | | | | | | | | | | X | | X | X | | | | | |
| bM 5.1 | UM | | | | X | X | | | | | | | | | | | | | | | | | X | |
| M 5.10 | PK | - | X | | | X | | | | | X | X | | | X | | | | | | | | | |
| bM 6.1 | NV | - | X | | | | | | | | | X | | | | | X | | X | | X | | X | X |
| bM 6.2 | NV | - | | | | | | | | | | X | | | | | | | X | | | | | |