Bundesamt
für Sicherheit in der
Informationstechnik

EMPFEHLUNG: IT IM UNTERNEHMEN

TLS/SSL Best Practice

Aufbauend auf der Technischen Richtlinie TR-02102-2, in der kryptographische Empfehlungen und Schlüssellängen zur Verwendung mit TLS gegeben werden, enthält die vorliegende BSI-Veröffentlichung zur Cyber-Sicherheit grundsätzliche Hinweise für die serverseitige Verwendung von Transport Layer Security (TLS)[1] mit HTTPS. Dabei dient TLS dem Schutz der zwischen einem Webserver und einem Browser (Client) ausgetauschten Daten. Neben HTTPS wird TLS z. B. auch von IMAP, SMTP über STARTTLS und bestimmten VPN-Typen zur Gewährleistung von Integrität, Vertraulichkeit und Authentizität der über ein unsicheres Netz (bspw. das WWW) übertragenen Daten verwendet. Besonderer Bedarf hierfür besteht immer dann, wenn sensitive Daten übermittelt werden. Insbesondere für HTTPS ist das immer dann der Fall, wenn

- Webseiten von Besuchern auszufüllende Formulare enthalten,
- Webseiten persönliche Daten verarbeiten,
- Webseiten einen Bereich anbieten, der für Besucher nur mit Login (z. B. mit Benutzername und Passwort) erreichbar sein soll,
- für Besucher die Echtheit der Webseiteninhalte eine besondere Rolle spielt.

In dem vorliegenden Best Practice-Dokument werden zunächst gravierende *Sicherheitsprobleme* von TLS/SSL vorgestellt und anschließend eine Reihe wichtiger Empfehlungen zu dessen Einsatz zusammengefasst, unterteilt in die wesentlichen Themenbereiche *Zertifikat*, *Konfiguration* und *Validierung der Server-Konfiguration*.

1 Sicherheitsprobleme

In den letzten Jahren wurden immer wieder schwerwiegende Angriffe gegen das TLS/SSL-Protokoll bzw. gegen dessen Implementierungen bekannt. Einige davon werden nachfolgend beschrieben.

Im Jahr 2009 wurde eine TLS/SSL-Schwachstelle beschrieben, die einem Angreifer (Man-in-the-middle) einen Angriffsvektor durch clientseitige Session-Neuverhandlung (Renegotiation) bietet. Dieser Angriff funktioniert mit SSL v3 und neueren TLS-Versionen. Als Schutzmaßnahme gegen diesen Angriff sollte durch den Client initiierte Renegotiation vom Server abgelehnt (siehe dazu Abschnitt 3.3 in BSI-TR-02102-2 und Renegotiation nur auf Basis von RFC 5746 verwendet werden.

BEAST ist die Bezeichnung für einen 2011 veröffentlichten Chosen plaintext-Angriff. Er richtet sich gegen die Implementierung der Verschlüsselung mit Blockchiffren im CBC-Modus in SSL v3 und TLS 1.0. Als Gegenmaßnahme wurde vorübergehend die Verwendung der Stromchiffre RC4 (statt einer Blockchiffre im CBC-Modus) in TLS 1.0 vorgeschlagen. Im Jahr 2013 wurde jedoch ein Angriff auf RC4 in TLS veröffentlicht, mit der Folge, dass RC4 nicht mehr als sichere Alternative in TLS 1.0 angesehen werden kann.

1 TLS ist der Name einer neuen Version des zuvor als Secure Sockets Layer (SSL) bekannten Protokolls.

Somit gibt es in TLS 1.0 keine Cipher Suite mehr, die nach dem aktuellen Stand der kryptografischen Forschung ausreichende Sicherheit bietet. Eine Migration zu TLS 1.1 oder 1.2 ist infolgedessen unerlässlich.

CRIME ist ein 2012 vorgestellter Angriff auf alle TLS-Versionen, der die Kompressionsrate der Nutzdaten als Seitenkanal verwendet. Das Ziel des Angriffs ist es, eine HTTPS-Sitzung zu übernehmen, indem der Angreifer das geheime Sitzungs-Cookie in Erfahrung bringt. Als Gegenmaßnahme wird empfohlen, die TLS-Kompression abzuschalten. Client-seitig wird dies mittlerweile von der üblicherweise eingesetzten Browser-Software unterstützt. Server-seitig muss die Kompression in der Konfiguration des Webservers deaktiviert werden.

Der 2013 veröffentlichte Angriff LUCKY13[2] hat das Ziel, Teile oder den gesamten Klartext einer TLS-Nachricht in Erfahrung zu bringen. Voraussetzung ist hierbei, dass die TLS-Verbindung eine Blockchiffre im CBC-Modus verwendet. Dieser Angriff ist ein Timing-Angriff, der geringe Zeit-Differenzen bei der Prüfung der HMAC-Tags von TLS-Nachrichten ausnutzt.

Wie zuvor schon BEAST richtet sich auch der 2014 unter dem Namen POODLE bekannt gewordene Angriff gegen eine Schwachstelle in SSL v3 und herstellerspezifische Implementierungen von TLS 1.0 (bspw. TLS/SSL terminierende Load-Balancer)[3]. Unter Nutzung des unzureichend gesicherten Paddings im CBC-Modus kann ein BEAST-ähnlicher Angriff durchgeführt und damit bspw. das Session-Cookie einer Sitzung ausgespäht werden. Laut den Entdeckern ist der POODLE-Angriff zudem einfacher durchführbar als BEAST. Aufgrund der Möglichkeit von Downgrade-Angriffen sollten Webserverbetreiber die Unterstützung von SSL v3 serverseitig vollständig deaktivieren. Nach Bewertung des BSI hat dies nur in Einzelfällen, wie bspw. dem Webseitenaufruf mit dem Internet Explorer 6 unter Windows XP, praktische Auswirkungen. Nur wenn die vollständige Migration zu TLS 1.1 und 1.2 unmöglich ist, kann TLS 1.0 unter bestimmten Bedingungen übergangsweise weiterverwendet werden[4].

Angriffe auf den CBC-Modus (bspw. BEAST, LUCKY13 und POODLE) können mit einem Wechsel von dem bisher in TLS/SSL üblichen Mac-then-Encrypt hin zu einem authentifizierten Verschlüsselungsverfahren verhindert werden.

Eine ergänzende Übersicht bieten auch aktuelle Zusammenfassungen der bekannt gewordenen Schwachstellen und Angriffsarten[5][6].

2 Zertifikat

2.1 Vertrauenswürdige Zertifizierungsstelle

Aufgrund der großen Zahl von Zertifizierungsstellen (Certificate Authority, CA) auf dem Markt sollte ein Anbieter sorgfältig selektiert werden. Es ist daher ratsam, die für den späteren Betrieb wesentlichen Auswahlkriterien im Vorfeld festzulegen. Zu diesen können beispielsweise gehören:

- das Vorhandensein in den CA-Listen der Browser,
- Sitz und Rechtsstand der Firma[7], geschäftliche Ausrichtung (CA-Betrieb ein zentrales Geschäftsfeld?), sowie die angebotenen CA-Dienste (OSCP, CRL),
- das Sicherheitsniveau und mitunter der Sitz des technischen Betriebs,
- Umfang und Qualität des technischen Supports.

2 http://www.isg.rhul.ac.uk/tls/Lucky13.html
3 Von betroffenen Herstellern werden Updates angeboten. Grundsätzlich gilt, dass auch TLS/SSL Libraries und Endpunkte regelmäßig aktualisiert werden müssen.
4 Für ergänzende Details und Übergangsfristen siehe BSI-TR-02102-2, Abschnitt 3.2.2.
5 „Lessons Learned From Previous TLS/SSL Attacks - A Brief Chronology Of Attacks And Weaknesses“ http://eprint.iacr.org/2013/049.pdf
6 Summarizing Known Attacks on Transport Layer Security (TLS) and DTLS“ https://www.rfc-editor.org/rfc/rfc7457.txt
7 Eingriffe fremder Nachrichtendienste in die Schlüsselerzeugung lassen sich mitunter umgehen, indem Zertifikate nationaler SSL-CAs verwendet werden.

Die Technische Richtlinie TR-03145 beschreibt Anforderungen für den sicheren Betrieb einer Zertifizierungsstelle[8]. Anhand darauf aufbauender Audits und Zertifizierungen können zukünftig ebenfalls CAs ausgewählt werden.

2.2 Extended-Validation-Zertifikat

Extended-Validation (EV) Zertifikate bieten gegenwärtig die beste Möglichkeit, sich Besuchern der eigenen Webseite als vertrauenswürdige Organisation zu präsentieren. Vor der Ausstellung des Zertifikats muss der Antragsteller gemäß der vom CA/Browser Forum, einem Zusammenschluss von Zertifizierungsstellen und Browser-Entwicklern, festgelegten Vergabekriterien überprüft werden[9]. Um dem Nutzer zu zeigen, dass er sich auf einer legitimen und authentischen Webseite befindet, färbt sich die Navigationsleiste moderner Browser beim Einsatz eines EV-SSL-Zertifikats grün. Insbesondere Anbieter von Webseiten, über die monetäre Transaktionen durchgeführt werden können, sollten diese Art von Zertifikaten verwenden.

2.3 Ausreichende Schlüssellänge und Schutz des privaten Schlüssels

Zertifikate haben meist eine mehrjährige Gültigkeit. Das CA/Browser Forum sieht vor, dass die Schlüssellängen von EV-SSL-Zertifikaten bestimmten Mindestanforderungen genügen sollen. Für die Nutzung des häufig verwendeten Verschlüsselungsverfahrens RSA empfiehlt das BSI in der Technischen Richtlinie TR-02102-1 „Kryptographische Verfahren: Empfehlungen und Schlüssellängen“ gegenwärtig die Verwendung von Schlüsseln mit einer Länge von mindestens 2048 Bit[10]. Nach Einschätzung des BSI bietet diese Schlüssellänge momentan noch einen ausreichenden Schutz. Bei der Beantragung oder Verlängerung eines EV-SSL-Zertifikats sollten Anbieter von Webseiten darauf achten, dass die entsprechende Mindestschlüssellänge eingehalten wird. Zum Schutz vor unauthorisiertem Zugriff muss der private Schlüssel außerdem sicher aufbewahrt werden. Dies kann bspw. unter Nutzung einer Hardwareeinheit zur Speicherung und Anwendung des jeweiligen Schlüssels realisiert werden.

2.4 Common Name-Eintrag

Beim Aufbau einer TLS/SSL-Verbindung wird vom Webbrowser geprüft, ob der Common Name[11] des übermittelten Zertifikats mit der URL der aufgerufenen Webseite übereinstimmt. Dabei sollten Wildcard-Zertifikate (z. B. für *.bund.de) zur Absicherung unterschiedlicher Subdomains vermieden werden.

8 https://www.bsi.bund.de/DE/Publikationen/TechnischeRichtlinien/tr03145/index_htm.html
9 https://cabforum.org/extended-validation
10 https://www.bsi.bund.de/DE/Publikationen/TechnischeRichtlinien/tr02102/index_htm.html
11 Der *Common Name* eines Webservers ist der vollständige DNS-Name (*Fully Qualified Domain Name*), über den er im Web erreichbar ist. Unqualifizierte Name (z. B. 'www'), lokale Namen (z.B. 'localhost') oder private IP-Adressen (z. B. 192.168.1.1) entsprechen nicht Spezifikation und sind daher zu vermeiden.

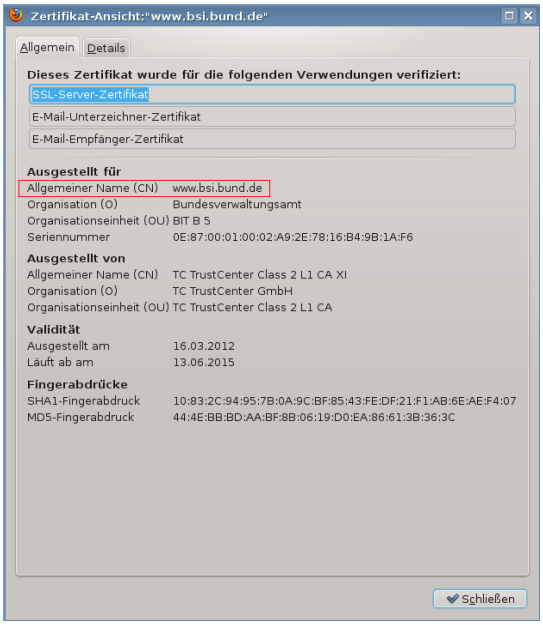

Abbildung 1: Common Name des BSI-Zertifikats

Sollen mehrere Subdomains mit TLS/SSL abgesichert werden, also beispielsweise https://bsi.bund.de und https://bmi.bund.de, so sind idealerweise auch verschiedene Zertifikate zu verwenden.

Ist ein Zertifikat zur Absicherung mehrerer Subdomains vorgesehen, können diese gemäß Spezifikation in das Erweiterungsfeld Subject Alternate Names (SANs) eingetragen werden[12]. Da Webbrowser bei fehlender Übereinstimmung eine Zertifikatswarnung anzeigen, muss sichergestellt werden, dass die im Zertifikat enthaltenen Namen tatsächlich zu den URLs passen.

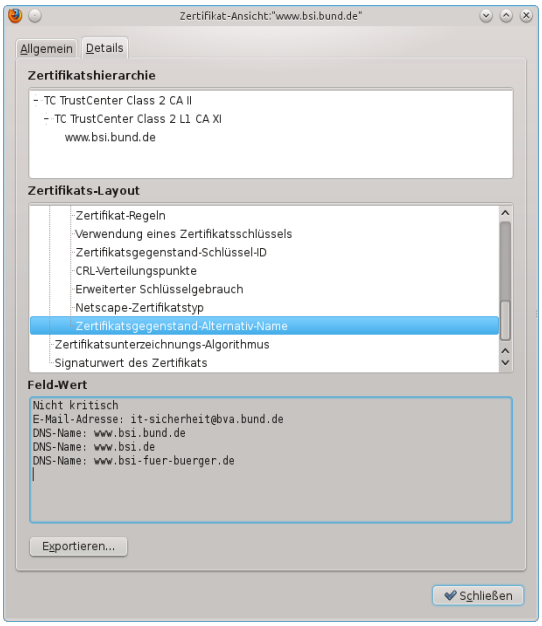

Abbildung 2: Alternativ-Namen des BSI-Zertifikats

2.5 SHA-2 Fingerprint

Um die Echtheit eines Zertifikats zu garantieren, signieren Zertifizierungsstellen üblicherweise den Fingerprint eines ausgestellten Zertifikats. Ein Fingerprint ist der mit einem bestimmten Algorithmus berechneter Hash-Wert eines öffentlichen Schlüssels. Wenn es einem Angreifer gelingen kann, für ein Zertifikat einen identischen Hash-Wert zu erzeugen, ist dessen Authentizität nicht mehr eindeutig an seinem Fingerprint zu erkennen. Das hieße aber auch, dass die

12 http://tools.ietf.org/html/rfc6125#page-32

Authentizität einer Webseite nicht mehr eindeutig an ihrem Zertifikat erkannt werden kann. Da mit der in die Jahre gekommenen Hash-Funktion SHA-1 ein solches Szenario nicht mehr ausgeschlossen werden kann, ist deren Verwendung nicht mehr empfehlenswert. Verschiedene Browserhersteller haben darauf reagiert und angekündigt, Zertifikate mit einem SHA-1-Fingerprint in Kürze nicht mehr als vertrauenswürdig einzustufen[13][14][15].

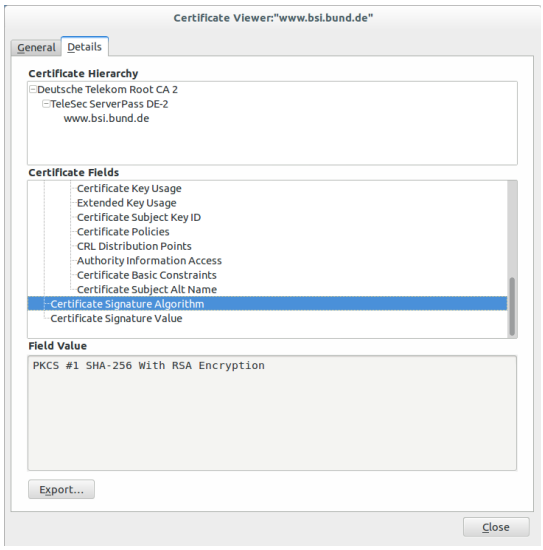

Abbildung 3: Signatur Algorithmus des BSI-Zertifikats

Wie auch bei den Cipher Suites gilt, dass die schnellstmögliche Migration zu SHA-2 (SHA-256, SHA-384 bzw. SHA-512) empfohlen wird. Von diesem Sachverhalt betroffene Webseitenbetreiber sollten sich bei ihrer Zertifizierungsstelle erkundigen, wann entsprechende Zertifikate mit einem auf SHA-2 basierenden Hash-Wert ersetzt und ob diese für Neuausstellungen automatisch verwendet werden.

3 Konfiguration

3.1 HTTP Strict Transport Security

Bei der TLS/SSL-Konfiguration sollte vermieden werden, dass Webseiten aus gemischten Inhalten bestehen. Als Webseite mit gemischtem Inhalt wird eine Seite bezeichnet, die zwar Verschlüsselung nutzt, dabei aber auch unverschlüsselte Inhalte (z. B. JavaScript-, CSS-Dateien oder Bilder) einbindet. Ein eine Man-in-the-Middle-Position nutzender Angreifer kann die Übertragung einer einzelnen unverschlüsselten Datei ausnutzen, um eine HTTPS-Session zu kapern. Da Webseiten mit gemischten Inhalten zudem üblicherweise Browser-Warnungen erzeugen, wird dadurch die Benutzerfreundlichkeit verschlechtert.

In diesem Kontext wird für die sichere serverseitige Konfiguration von HTTPS die Verwendung von HTTP Strict Transport Security (HSTS) angeraten. Bei HSTS handelt es sich um eine Sicherheitseinstellung mit der Aufrufe von ungesicherten Bereichen einer Webseite unterbunden werden. Letztlich geschieht dies durch das serverseitige Versenden eines Response Headers. Diese instruieren den Browser, alle nachfolgenden Webseitenaufrufe über HTTPS zu versenden. Das Vorgehen bei der Konfiguration unterscheidet sich je nach Webserver. Auf verschiedenen Webseiten existieren Anleitungen zur HSTS-Konfiguration, bspw. für Apache[16], IIS[17] und nginx[18].

13 http://blogs.technet.com/b/pki/archive/2013/11/12/sha1-deprecation-policy.aspx
14 http://googleonlinesecurity.blogspot.com/2014/09/gradually-sunsetting-sha-1.html
15 https://blog.mozilla.org/security/2014/09/23/phasing-out-certificates-with-sha-1-based-signature-algorithms/
16 Apache: https://www.owasp.org/index.php/HTTP_Strict_Transport_Security
17 IIS: http://www.iis.net/configreference/system.webserver/httpprotocol/customheaders
18 Nginx: https://scotthelme.co.uk/setting-up-hsts-in-nginx/

3.2 Sichere Protokolle

Von den existierenden TLS/SSL-Protokollversionen (SSL v2, SSL v3, TLS 1.0, TLS 1.1 und TLS 1.2) werden momentan die Varianten TLS 1.1 und TLS 1.2 als ausreichend sicher eingestuft (sofern richtig konfiguriert, siehe BSI-TR-02102-2. Aufgrund der noch nicht flächendeckenden Verbreitung dieser Versionen kann ggf. die übergangsweise Verwendung von TLS 1.0 in Ergänzung zu TLS 1.1 und TLS 1.2 erwogen werden. Ratsam ist dies allerdings nur, falls eine sofortige Migration zu TLS 1.1 oder TLS 1.2 unmöglich ist. In diesem Fall müssen aber Gegenmaßnahmen gegen die bekannten Angriffe auf TLS 1.0 getroffen werden. Für weitere Details und die Länge des tolerablen Übergangszeitraums wird auf BSI-TR-02102-2, Abschnitt 3.2.2, verwiesen. Von der Verwendung von SSL v2 und SSL v3 wird grundsätzlich abgeraten.

3.3 Sichere Cipher Suites

Beim Aufbau einer TLS/SSL-Verbindung (Handshake) einigen sich Client und Server auf eine sogenannte Cipher Suite; diese legt die zu verwendenden Algorithmen (Verschlüsselung, Schlüsselvereinbarung, MAC-Sicherung usw.) für die aufzubauende TLS-Verbindung fest. Für empfohlene Cipher Suites wird auf Abschnitt 3.2. der BSI-TR-02102-2, verwiesen. Zu der Verwendung anderer Cipher Suites wird nicht geraten.

Es ist die Verwendung von Cipher Suites mit *Forward Secrecy* (auch Perfect Forward Secrecy) zu bevorzugen. Mit Forward Secrecy werden private Schlüsselpaare des Servers bei der Erzeugung des Session Keys so verwendet, dass ein Angreifer mit Kenntnis des privaten Serverschlüssels nicht rückwirkend alle verwendeten Session Keys berechnen kann. Damit können zuvor aufgezeichnete TLS/SSL-Verbindungen nicht nachträglich von einem Angreifer entschlüsselt werden. In der bereits mehrfach erwähnten BSI-TR-02102-2 werden explizit Cipher Suites mit dieser Eigenschaft empfohlen (siehe Tabelle 1 in Abschnitt 3.2.1).

3.4 Vollständige Zertifikatskette

Da für die Prüfung der hierarchischen Zertifikatskette durch den Webbrowser auch alle Zwischen-Zertifikate benötigt werden, reicht das SSL-Zertifikat des Servers alleine nicht aus. Deshalb muss der Server beim Verbindungsaufbau alle erforderlichen Zertifikate an den Client senden. Dazu wird die Zertifikatskette im Webserver entsprechend hinterlegt. Das Vorgehen bei der Konfiguration unterscheidet sich je nach Webserver.

Zu beachten ist außerdem, dass neben fehlenden auch abgelaufene oder gesperrte CA-Zertifikate die Prüfung der Zertifikatskette ungültig machen. Nur wenn alle benötigten Zertifikate gültig sind und beim Verbindungsaufbau übertragen wurden, ist eine erfolgreiche Prüfung der Zertifikatskette möglich.

3.5 OCSP Stapling

Client-seitig wird die Gültigkeit eines X.509-Zertifikats des Webservers in der Regel durch die Nachfrage bei der herausgebenden Zertifizierungsstelle geprüft. Hierfür wird häufig das OCSP-Protokoll verwendet. Da die Zertifizierungsstelle darüber aber auch Kenntnis über die Verbindungsdetails des mit dem Server kommunizierenden Clients erhält, kann es ratsam sein, diese private Information zu schützen. Dies kann über das als OCSP Stapling bekannte Verfahren geschehen. Dabei erfragt ein Webserver die OCSP-Information seines Zertifikats selbst bei der entsprechenden CA und schickt es als Teil des TLS/SSL-Handshakes an den Client. Dort wird die Gültigkeitsprüfung vom Browser vollzogen.

Da OCSP Stapling mittlerweile von Webservern, bspw. von Apache, nginx[19] und IIS[20], und von neueren Browser-Versionen auch clientseitig unterstützt wird, wird dessen Nutzung angeraten.

19 Apache, nginx: https://www.digitalocean.com/community/tutorials/how-to-configure-ocsp-stapling-on-apache-and-nginx
20 Microsoft: „Configure OCSP Stapling“ http://technet.microsoft.com/en-us/library/hh826044(v=ws.10).aspx

3.6 Encrypt-then-MAC und Authenticated Encryption

Einige der zuvor beschriebenen Angriffe auf den CBC-Modus (z.B. BEAST, LUCKY13, POODLE) können mit einem Wechsel von dem bisher in TLS/SSL üblichen Mac-then-Encrypt zu einem *authentifizierten Verschlüsselungsverfahren* verhindert werden. Dies kann grundsätzlich auf zwei Arten geschehen:

Theoretisch kann unter TLS v1.1 und TLS v1.2 das *Encrypt-then-MAC (EtM)*-Verfahren auf Basis von RFC 7366 verwendet werden. Da es sich dabei um eine relativ neue Spezifikation handelt, wurde die Implementierung des Verfahrens aber in den meisten TLS/SSL-Bibliotheken noch nicht umgesetzt. Der Einsatz wird aber generell empfohlen, sobald geeignete Implementierungen zur Verfügung stehen.

Eine ähnliche Funktionalität wie EtM bieten auch unter *Authenticated Encryption* bekannte Cipher Suites. Da ab TLS v1.2 Authenticated Encryption unterstützt, sollten entsprechende Cipher Suites (z.B. AES-GCM) eingestellt werden.

Ergänzend zu den in diesem Abschnitt genannten Empfehlungen bieten Mozilla[21], OWASP[22] und Qualys[23] konkrete Konfigurationshinweise für TLS an. Praktische Hinweise zur sichereren Konfiguration von TLS/SSL in verschiedenen Einsatzumgebungen werden auch von der „Applied Crypto Hardening"-Initiative zur Verfügung gestellt[24].

4 Validierung der Server-Konfiguration

Die Auswirkungen von serverseitigen Konfigurationsänderungen lassen sich nicht immer mit Bestimmtheit vorhersagen. Auch Software-Updates führen mitunter zu überraschenden Änderungen. Deswegen ist es ratsam, die TLS/SSL-Konfiguration vor der Freigabe zur Nutzung auf Fehler zu prüfen und den Status in periodischen Abständen zu validieren. Frei verfügbare Online-Scanner werden z. B. von SSL Labs[25] angeboten. Auch existieren eine Reihe von Online-Scannern, mit denen eine Implementation ausschließlich auf bestimmte Verwundbarkeiten, wie bspw. Heartbleed oder POODLE, getestet werden kann.

Automatisierte Tests sind allerdings mit Vorsicht zu genießen. Wenn ein Test nicht alle Cipher Suites testet, kann es vorkommen, dass unsichere Cipher Suites konfiguriert sind, ohne dass dies in den Ergebnissen des Scans sichtbar wird. Es ist daher ratsam, die genauen Details des dokumentierten Testverfahrens zu prüfen. Bei der Verwendung von Online-Scannern sollte man sich auch darüber im Klaren sein, dass die Ergebnisse des Tests unter Umständen öffentlich sichtbar werden bzw. inkorrekt sein können.

Mit der eigenen Validierung, bspw. unter Verwendung der von OWASP referenzierten Tests[26] oder eines gegen die serverseitigen TLS/SSL-Richtlinien von Mozilla testenden Analyseprogramms[27], können lokale Analysen der eigenen Konfiguration durchgeführt werden. Damit können Tests angepasst und das Risiko einer Veröffentlichung von Testergebnissen auf einer Webseite minimiert werden.

21 Mozilla: „Security / Server Side TLS" https://wiki.mozilla.org/Security/Server_Side_TLS
22 OWASP: „OWAS Transport Layer Protection Cheat Sheet" https://www.owasp.org/index.php/Transport_Layer_Protection_Cheat_Sheet
23 Qualys SSL Labs: „SSL/TLS Deployment Best Practices" https://www.ssllabs.com/projects/best-practices/
24 https://bettercrypto.org/static/applied-crypto-hardening.pdf
25 https://www.ssllabs.com/ssltest/
26 OWASP: „Testing for Weak SSL/TLS Ciphers, Insufficient Transport Layer Protection (OWASP-EN-002)" https://www.owasp.org/index.php/Testing_for_Weak_SSL/TLS_Ciphers,_Insufficient_Transport_Layer_Protection_%28OTG-CRYPST-001%29
27 https://jve.linuxwall.info/blog/index.php?post/2014/10/09/Automated-configuration-analysis-for-Mozilla-s-TLS-guidelines