Bericht

Webapplikations-Penetrationstest - Musterprojekt

TÜV TRUST IT
TÜV AUSTRIA GmbH
TÜV AUSTRIA-Platz 1
2345 Brunn am Gebirge

Ansprechpartner:
DI Rainer Seyer, BSc
Tel: +43 5 0454 - 6231
Mobil: +43 664 60454 6231
E-Mail: rainer.seyer@tuv.at
https://www.it-tuv.com

TÜV ®

An:

Musterkunde GmbH

z.Hd. Herrn Mustermann

Musterstraße 1

A-1111 Wien

Angebots/Auftragsnummer: A21xxxxxx

Wien, 8. Oktober 2021

Berichtsversion: 1.0

Geschäftsführung:
Detlev Henze
Andreas Köberl

Sitz:
Deutschstraße 10
1230 Wien

Firmenbuchgericht / -nummer:
Wien / FN 213923 v

Bankverbindungen:
UniCredit Bank Austria AG
IBAN
AT451100009654319400
BIC BKAUATWW

UID ATU 52720505
DVR 3002478

Inhaltsverzeichnis

Management-Zusammenfassung

TÜV TRUST IT TÜV AUSTRIA GmbH (in der Folge mit „TÜV TRUST IT“ abgekürzt) wurde von Musterkunde GmbH (in der Folge mit „Musterkunde“ abgekürzt) mit der Durchführung einer technischen Sicherheitsüberprüfung beauftragt. Umfang dieser Sicherheitsüberprüfung war ein Webapplikations-Penetrationstest im **Grey-Box-Verfahren** bezogen auf die Webapplikation „**Musterprojekt**“. Ziel der Analyse war es, sicherheitsrelevante Risiken zu identifizieren, die es einem externen Angreifer erlauben würden, die Sicherheitsziele, im Besonderen die Integrität, die Vertraulichkeit und die Verfügbarkeit von IT-Systemen, Prozessen oder Kundendaten zu gefährden.

Die durchgeführte Analyse orientiert sich an internationalen Standards (u.a. OWASP Top 10) und eigener Erfahrungswerte und wurde im Zeitraum von **28. Dezember bis 31. Dezember 2021** durchgeführt.

Aus den Ergebnissen konnte ermittelt werden, dass die Musterprojekt-Applikation zwei Schwachstellen mit hohem Risiko aufweist und daher dringend verbessert werden sollte. Diese Schwachstellen betreffen den Punkt „Überprüfung von Benutzereingaben“. Durch das Ausnutzen dieser Schwachstelle ist ein Auslesen von internen und vertraulichen Daten der Applikation ebenso möglich wie das Einschleusen von Schadcode. Weiters wurden sieben Schwachstellen mit mittlerem Risiko identifiziert. Deren rasche Behebung ist ebenfalls anzuraten.

Eine Auswertung der Ergebnisse ist auf den folgenden Seiten grafisch aufbereitet. Detaillierte Informationen zu den evaluierten Schwachstellen können im Hauptkapitel „*Ergebnisse*“ eingesehen werden.

Das Sicherheitsniveau der getesteten Applikation wird mittels drei Bewertungen eingestuft. Diese Bewertungsskala wird subjektiv von den TÜV TRUST IT Auditoren nach Kundeninteressen und auftretenden Art der Schwachstellen bewertet. Hierbei wird eine Abstufung in vier Bewertungen getroffen: *Problematisch*, *Verbesserungspotential*, *Solide* und *Nicht Anwendbar*. Hierbei stellen als *Problematisch* definierte Kategorien meist konzeptionelle Probleme und für den Kunden kritische Schwachstellen dar. Die Bewertung *Verbesserungspotential* deutet auf geringere, aber ernstzunehmende Schwachstellen hin. *Solide* bedeutet, dass keine sicherheitsrelevanten Schwachstellen in diesem Bereich identifiziert werden konnten. Ist die jeweilige Kategorie im vereinbarten und durchgeführten Prüfumfang nicht enthalten oder nicht anwendbar, wird das Niveau mit *Nicht Anwendbar* deklariert und der Grund dafür in der entsprechenden Spalte vermerkt.

OWASP-Kategorie	Niveau
Überprüfung von Benutzereingaben (INPV)	Problematisch
Logische Überprüfung von Abläufen (BUSL)	Verbesserungspotential
Benutzerseitige Absicherung (CLNT)	Verbesserungspotential
Sicherer Umgang mit Informationen (INFO)	Verbesserungspotential
Sichere Serverkonfiguration (CONF)	Verbesserungspotential
Verwendung von Kryptografie (CRYP)	Verbesserungspotential
Sicheres Identitätsmanagement (IDNT)	Verbesserungspotential
Sicherer Anmeldungsprozess (ATHN)	Verbesserungspotential
Sichere Zugriffssteuerung (ATHZ)	Verbesserungspotential
Umgang mit Benutzersitzungen (SESS)	Verbesserungspotential
Korrekte Fehlerbehandlung (ERRH)	Solide

Die daraus resultierenden Risiken der Webapplikation sind nachfolgend in die Kategorien Kritisch (K), Hoch (H), Mittel (M) und Gering (G) sowie Information (I) unterteilt:

ID	CVSS-Score	Bezeichnung	OWASP-Kategorie	Seite
K1	9,4	SQL Injection	INPV	12
H1	8,0	Persistentes Cross-Site-Scripting	INPV	16
M1	6,3	Vertikale Privilegienerweiterung	ATHZ	18
M2	5,3	Benutzerenumerierung möglich	IDNT	20
M3	5,3	Einsatz schwacher Passwortrichtlinien	ATHN	22
M4	4,6	Logout-Funktionalität nicht vorhanden	SESS	23
M5	4,6	Einsatz veralteter Software	INFO	24
M6	4,3	Enumerierung von administrativen Interfaces	CONF	25
G1	3,7	Upload von Malware möglich	BUSL	27
G2	3,7	TLS 1.0-Protokoll in Verwendung	CRYP	29
G3	3,7	Verwendung verwundbarer Ciphers (“SWEET32”)	CRYP	31
G4	3,1	Applikationslogik: Datenvalidierung unzureichend	BUSL	32
G5	3,1	Webserveridentifikation	INFO	34
G6	3,1	Fehlende HTTP-Sicherheitsheader	CLNT	36
G7	2,6	Alter Softwarestand - Bibliotheken	INFO	38
I1	0,0	Unbekannte Zertifizierungsstelle	CRYP	40

1 Grundlagen und Methodik

Dieser Abschnitt beschreibt die Ausgangssituation, den Geltungsbereich sowie die Zielsetzung und die Prüf- und Bewertungsgrundlagen des durchgeführten Audits. Zudem werden die Vorgehensweise und die Art der Risiko-Bewertung erörtert.

1.1 Ausgangssituation und Zielsetzung

Musterkunde betreibt ein großes Computernetz mit vielen Webapplikationen und ist um dessen Sicherheit besorgt. Daher plant Musterkunde die Durchführung eines Penetrationstests der Musterprojekt-Webapplikation.

Die Ergebnisse der Leistungen dienen ausschließlich der Information und dem internen Gebrauch des Musterkundens. Sollen diese an andere Adressaten weitergeleitet werden, dann ist dies zuvor gesondert schriftlich zu vereinbaren und gegebenenfalls an eine Schad- und Klagloserklärung gegenüber des Musterkundens gebunden.

1.2 Prüf-/Audit-Grundlagen

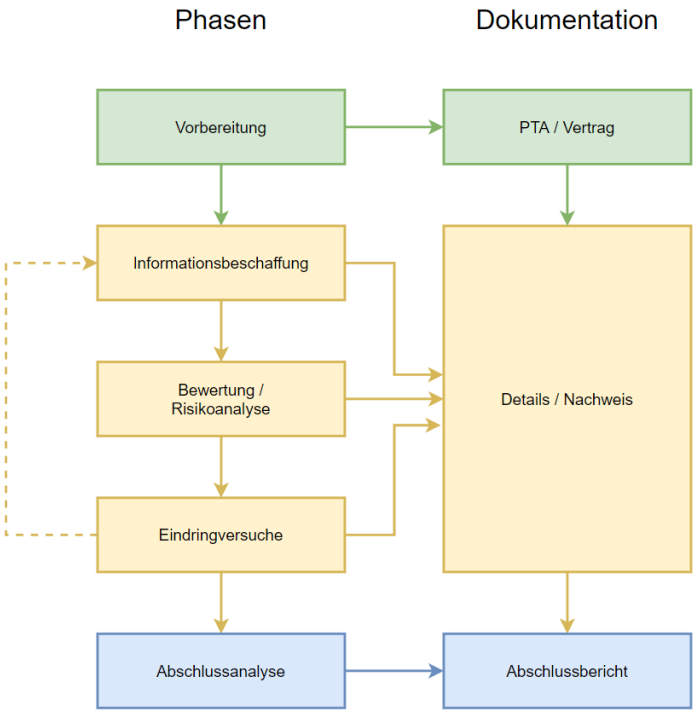
Als Prüfgrundlagen wurden verwendet:

- ISO/IEC 27001:2015, "Information security management systems"
- ISO/IEC 27002:2015, "Information technology security techniques"
- ISO/IEC 27033:2010, “Information technology – Security techniques – Network security”
- BSI-Studie „Durchführungskonzept für Penetrationstests“
- BSI IT-Grundschutz
- OWASP (Open Web Application Security Project)

1.3 Vorgehensweise

Die Grundlage für die Vorgehensweise des Penetrationstests erfolgt in Anlehnung an das Durchführungskonzept für Penetrationstests des BSI[1]. Das Vorgehen im Rahmen der sicherheitstechnischen Untersuchung umfasst typischerweise folgende Phasen:

[1] https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Studien/Penetrationstest/penetrationstest.pdf?__blob=publicationFile&v=3



Die fünf Phasen werden dabei parallel zum Penetrationstest protokolliert, damit die Überprüfung transparent und nachvollziehbar ist. Falls es zu einer Abweichung der Methodik kommen sollte, wird diese dokumentiert und begründet.

Während der in obiger Grafik aufgeführten Phasen Informationsbeschaffung und Eindringversuche wird mit dem Prüfobjekt interagiert. Dabei werden typischerweise folgende Aspekte betrachtet:

Dienste
• Reduzierung der angebotenen Dienste auf die tatsächlich benötigten Komponenten und Module • Verschlüsselung der Netzwerkkommunikation • Einsatz aktueller Software ohne bekannte sicherheitsrelevante Schwachstellen • Härtung der Dienstkonfiguration, soweit ersichtlich • Angemessenheit der Verfahren zur Benutzerauthentifizierung • Beschränkung der Zugriffsmöglichkeiten auf das System
Skripte, Funktionen und Applikationen
• Zugriffsmöglichkeiten ohne vorherige Authentisierung • Angemessenheit der Verfahren zur Benutzerauthentifizierung • Session-Handling • Zugriffsmöglichkeiten auf administrative Schnittstellen oder Teile von Applikationen, die nicht öffentlich zugänglich sein sollten

- Zugriffsmöglichkeiten auf Daten anderer authentisierter Benutzer
- Schwachstellen, die den Zugriff auf Datenbanken, das Betriebssystem oder Dateisystem ermöglichen
  - Schwachstellen, die den Zugriff auf die Session oder die Benutzerkennung eines anderen Benutzers ermöglichen
  - Fehlerbehandlung
  - Minimalisierung der Applikationsinstallation (Backup- und Testdaten, nicht benötigte Module und Skripte)

Bei der Durchführung der Sicherheitsüberprüfung werden typischerweise folgende Tools verwendet:

Tools	
• Nessus • Burp Suite • Sqlmap	• Nmap • Metasploit • Hydra

Abhängig von dem zu prüfenden Objekt kommen spezialisiertere Tools zum Einsatz. Aus diesem Grund stellt die angeführte Liste keine vollständige Auflistung dar und kann variieren.

„Denial of Service“-Angriffe werden, sofern nicht gesondert vereinbart, nicht durchgeführt, um den Produktionsbetrieb der Systeme im Untersuchungsbereich nicht zu gefährden. Falls die Anwendung eines verfügbaren Exploits zu einer „Denial of Service“-Situation führen könnte, wird dieser nicht angewendet und die Schwachstelle entsprechend dokumentiert.

1.4 Bewertungsgrundlagen und Risiko-Klassifizierung

Die identifizierten Risiken werden in folgende Risiko- und Empfehlungs-Kategorien unterteilt. Hierfür wird CVSS in der Version 3.1 als Bewertungssystem eingesetzt.

Kritisches Risiko	Bedeutung	• Unmittelbare Gefährdung des Sicherheitsziels • Einfache Ausnutzung der Schwachstelle
	Beispiel	• Übernahme des Systems
	Aktion der Auditoren	• Sofortige Information des Kunden • Sofortiges Übermitteln einer Schwachstellenbeschreibung und Empfehlung
	Empfehlung an Kunden	• Schnellstmögliches Beheben der Bedrohung • Interne Analyse der Auswirkungen der Schwachstelle
	CVSS-Bereich	• 10,0 – 9,0
Hohes Risiko	Bedeutung	• Unmittelbare Gefährdung des Sicherheitszieles • Realistische Ausnutzung der Schwachstelle
	Beispiel	• Unerlaubter Zugriff auf administrative Funktionen oder Code-Injection
	Aktion der Auditoren	• Zeitnahe Information des Kunden • Übermittleln einer Schwachstellenbeschreibung auf Anfrage
	Empfehlung an Kunden	• Baldige Umsetzung von Behebungsmaßnahmen • Interne Analyse der Auswirkungen der Schwachstelle
	CVSS-Bereich	• 8,9 – 7,0
Mittleres Risiko	Bedeutung	• Konkrete Gefährdung des Sicherheitsziels • Aufwändige Ausnützung der Schwachstelle
	Beispiel	• Schwere Konfigurationsfehler
	Aktion der Auditoren	• Dokumentation der Schwachstelle im Bericht
	Empfehlung an Kunden	• Festlegen eines Zeitplans zur Behebung
	CVSS-Bereich	• 6,9 – 4,0
Geringes Risiko	Bedeutung	• Geringe Gefährdung des Sicherheitsziels • Komplexe Ausnutzbarkeit der Schwachstelle
	Beispiel	• Triviale Konfigurationsfehler
	Aktion der Auditoren	• Dokumentation der Schwachstelle im Bericht
	Empfehlung an Kunden	• Behebung in längerfristige Planung aufnehmen
	CVSS-Bereich	• 3,9 – 0,1
Information	Bedeutung	• Keine Bedrohung des Sicherheitsziels • Schafft Aufmerksamkeit für ein bestimmtes Verhalten
	Beispiel	• Informationen in Metadaten
	Aktion der Auditoren	• Dokumentation der Schwachstelle im Bericht
	Empfehlung an Kunden	• Analyse und Beurteilung der Information
	CVSS-Bereich	• 0,0
Behoben	Bedeutung	• Bedrohung des Sicherheitsziels wurde als Behoben verifiziert
	Aktion der Auditoren	• Dokumentation des Behebungsstatus im Bericht
	Empfehlung an Kunden	• Keine Aktionen notwendig
	CVSS-Bereich	• Alter CVSS-Wert durchgestrichen

1.5 Audit-Team

Teammitglied	Kontaktdaten	Rolle
DI Rainer Seyer, BSc	Mobil: +43 664 60454 6231 E-Mail: rainer.seyer@tuv.at	Lead Auditor
Ing. Sabrina Semper, MSc	Mobil: +43 664 60454 6241 E-Mail: sabrina.semper@tuv.at	Auditor
DI Mario Rubak, BSc	Mobil: +43 664 60454 6238 E-Mail: mario.rubak@tuv.at	Qualitätssicherung

1.6 Dokumenten-Historie

Version	Kommentar	Dateiname	Datum
V1.0	Initiale Berichtsversion	A21xxxxxx_Penetrationstest-Musterprojekt_Musterkunde_v1.0_A	31.12.2021

2 Testabdeckung

2.1 Zielsysteme

Folgende Systeme sind auf Basis der initialen Abstimmung im Prüfumfang:

Bezeichnung	Zielsystem	Prod/Test	Testtiefe
Musterprojekt	https://www.musterkunde.at	Test	Grey-Box-Test

Dies wurde in einer Permission to Attack (PtA) festgehalten.

Eine Deaktivierung der Web Application Firewall (WAF) zum Zweck des Audits wurde nicht durchgeführt, da keine WAF vorhanden war.

2.2 Ausgangslage

Ausgangslage war ein externer Angreifer, welcher über eine normale Internetverbindung auf die Applikation zugreift.

2.3 Zeitraum

Der Webapplikations-Penetrationstest wurde vor Ort im Testzeitraum vom 28. Dezember bis 31. Dezember 2021 durchgeführt. Die Berichterstellung erfolgte unmittelbar nach dem Test.

Zur Durchführung der Sicherheitstests stand ein Zeitrahmen von maximal 10 Personentagen (ohne Berichterstellung) zur Verfügung. Es wurde ein risikobasierter Ansatz verfolgt, der sicherstellt, dass risikoreiche Funktionen zuerst getestet werden (Time-Box-Ansatz).

2.4 Verwendete Rollen und Berechtigungen

Jedem der Auditoren wurden verschiedene Benutzerkonten mit unterschiedlichen Rollen zur Verfügung gestellt, mit welchen sich die Auditoren an der Webapplikation im Testzeitraum anmelden:

- User1 – Normaler Benutzer
- Admin1 - Administrator

2.5 Einschränkungen

Nicht im Audit-Umfang ist die hinter der Applikation befindliche Serverinfrastruktur.

3 Ergebnisse

K1	SQL Injection	
	Risiko	Kritisch
	CVSS-Bewertung	9,4 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:L)
	Referenz	WSTG-INPV-05
	Systeme	https://www.musterkunde.at/musterprojekt/SearchProjekt
	Vorbedingung	Keine

Allgemeine Beschreibung der Schwachstelle

SQL-Injection (abgekürzt "SQLi") bezeichnet das Ausnutzen einer Sicherheitslücke in Zusammenhang mit SQL-Datenbanken. Es handelt sich hierbei um eine Schwachstelle, die entsteht, indem ein Angreifer die Möglichkeit erhält, eine Structured Query Language (SQL)-Abfrage der Anwendung zu beeinflussen, die anschließend der Backend-Datenbank übergeben wird. Das Einschleusen der Befehle entsteht durch eine mangelhafte Maskierung oder Überprüfung der Benutzereingaben. Das Ziel eines Angreifers besteht darin, über die Anwendung, die den Zugriff auf die Datenbank bereitstellt, eigene Datenbankbefehle einzuschleusen (engl. *Injection*). Durch das Einschleusen von Befehlen wird es möglich auf Daten und Tabellen zuzugreifen und deren Inhalte zu lesen oder sogar zu manipulieren. Dies kann dazu führen, dass der Angreifer möglicherweise vertrauliche Informationen lesen oder sogar manipulieren kann. Schafft der Angreifer Zugriff auf eine Shell zu erhalten, ist sogar eine Kompromittierung des gesamten Servers möglich.

Nachweis der Schwachstelle

Es wurde eine Blind-SQL-Injection-Schwachstelle in der getesteten Webapplikation beim Parameter **projektname** des Requests *https://www.musterkunde.at/musterprojekt/SearchProjekt* identifiziert.

Folgende Anfragen demonstrieren die Machbarkeit, wobei die erste Abfrage **CASE WHEN (2>1)** ein logisches **TRUE** liefert (repräsentiert durch den HTTP-Statuscode **200** in der Serverantwort) und die zweite Anfrage **CASE WHEN (2>3)** ein logisches **FALSE** liefert (repräsentiert durch den HTTP-Statuscode **400** in der Serverantwort):

```
GET /musterprojekt/SearchProjekt/?projektname=test-
2'+AND+(SELECT+(CASE+WHEN+(2>1)+THEN+NULL+ELSE+CTXSYS.DRITHSX.SN(1,3752)+END)+FROM+DU
AL)+IS+NULL--&projektgruppe=2 HTTP/1.1
→ HTTP/1.1 200 OK
```

```
GET /musterprojekt/SearchProjekt/?projektname=test-
2'+AND+(SELECT+(CASE+WHEN+(2>1)+THEN+NULL+ELSE+CTXSYS.DRITHSX.SN(1,3752)+END)+FROM+DU
AL)+IS+NULL--&projektgruppe=2 HTTP/1.1
→ HTTP/1.1 400 Bad Request
```

Dadurch ist es möglich, durch gezielte WAHR/FALSCH-Fragen die Datenbank Schritt für Schritt auszulesen. Da dieses Vorgehen sehr zeitintensiv ist, wurden im Rahmen des Penetrationstests nur einige Daten gezielt ausgelesen, um die Machbarkeit zu demonstrieren. Das Schreiben oder Löschen von Datensetzen wurde nicht getestet, um eine Verfälschung von Datenbankinformationen zu verhindern.

Folgende Informationen wurden aus der Datenbank gewonnen:

Abbildung 1 – Datenbank-Banner mit detaillierter Versionsinformation

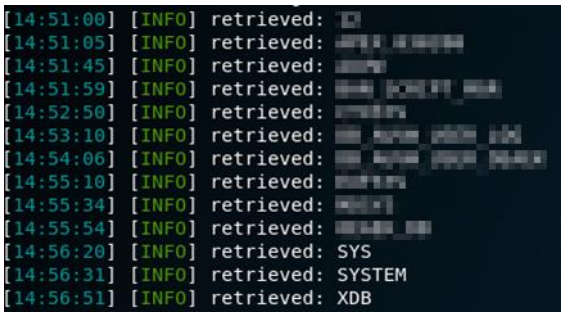


Abbildung 2 – Vorhandene Datenbanken

Abbildung 3 – Vorhandene Tabellen in der Datenbank „XX“

Abbildung 4 – Vorhandene Spalten in der Tabelle „XX_USER“ der Datenbank „XX“ (Auszug)

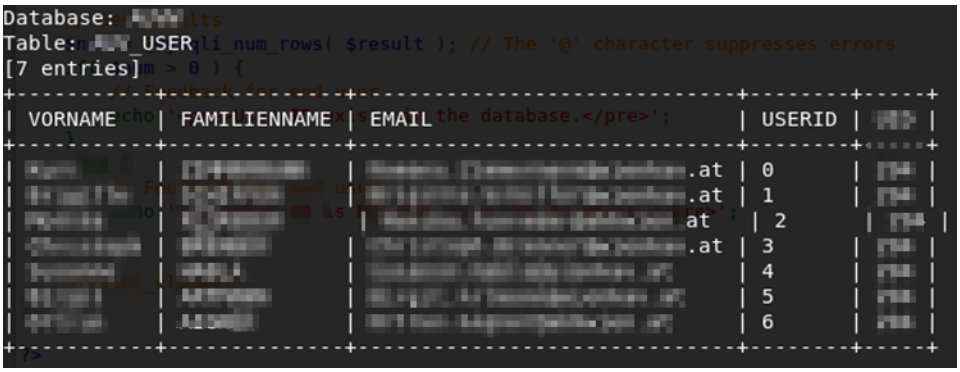


Abbildung 5 – Vorhandene Einträge in der Tabelle „XX_USER“ der Datenbank „XX“ (Auszug)

Empfehlung

Um einer Web-Anwendung keine Basis für SQL-Angriffe zu bieten, müssen alle Eingabeparameter als unsicher betrachtet und vor der weiteren Verarbeitung serverseitig strikt geprüft werden. Das Definieren von unzulässigen Eingaben (Black List) stellt keine verlässliche Methode dar, da es bereits unzählige Angriffsvektoren gibt. Empfehlenswert ist stattdessen das Festlegen von zulässigen Eingabewerten (White

List). Diese Vorgehensweise zählt zu den empfohlenen Vorgehensweisen (Best Practices) der Programmierung und sollte eingesetzt werden, um unerwartetes Programmverhalten zu verhindern.

Zusätzlich hat sich als Schutz vor SQL-Injections der Einsatz von parametrisierten Abfragen („Prepared Statements“) bewährt. Dieses Konzept ist in fast allen Programmiersprachen nutzbar.

H1	Persistentes Cross-Site-Scripting	
	Risiko	Hoch
	CVSS-Bewertung	8,0 (CVSS:3.1/AV:N/AC:L/PR:L/UI:R/S:U/C:H/I:H/A:H)
	Referenz	WSTG-INPV-01
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Authentifizierter Benutzer

Allgemeine Beschreibung der Schwachstelle

Cross-Site-Scripting (XSS) ist eine Art der Javascript-Injection. Cross-Site-Scripting tritt auf, wenn eine Webapplikation 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, Skripte indirekt an den Browser des Opfers zu senden und damit Schadcode auf der Seite des Clients auszuführen. Javascript-Injections können von Angreifern aber für eine Vielzahl von weiteren Angriffsszenarien verwendet werden.

Ein klassisches Beispiel für XSS ist die Übergabe von Parametern an ein serverseitiges Skript, das eine dynamische Webseite erzeugt. Dies kann beispielsweise das Eingabeformular einer Webseite sein, wie in Webshops, Foren, Blogs und Wikis üblich. Die eingegebenen Daten werden auf der Webseite wieder als Seiteninhalt ausgegeben, wenn die Seite von Benutzern aufgerufen wird. So ist es möglich, manipulierte Daten an alle Benutzer zu senden, sofern die Applikationslogik dies nicht serverseitig verhindert.

Häufige Angriffsarten sind das Stehlen von Benutzer-Sessions, Website-Defacements und das Einstellen negativer Inhalte sowie Phishing-Angriffe und die Übernahme der Kontrolle des Browsers des Benutzers.

Hierbei wird ausgenutzt, dass dynamisch generierte Web-Seiten ihren Inhalt oft über URL (HTTP GET-Methode) oder Formulare (HTTP POST-Methode) übergebene Eingabewerte anpassen. Persistent oder gespeichert heißt dieser Typ, da der Schadcode auf dem Webserver gespeichert wird. Dadurch wird der Schadcode bei jeder Anfrage ausgeliefert.

Nachweis der Schwachstelle

Es wurden zwei Funktionen gefunden, die es möglich machten, die Inputvalidierung zu umgehen und damit persistente Cross-Site-Scripting-Payloads in der Applikation zu speichern. Diese sind nur beispielhaft für ein generelles Problem zu betrachten, welches sich durch die komplette Applikation zieht.

Als Machbarkeitsnachweis war es möglich, in der Funktion „XY“ persistente Cross-Site-Scripting-Payloads zu speichern. Diese werden beim Aufruf der jeweiligen Funktion auch ausgeführt (siehe Abbildung 6).

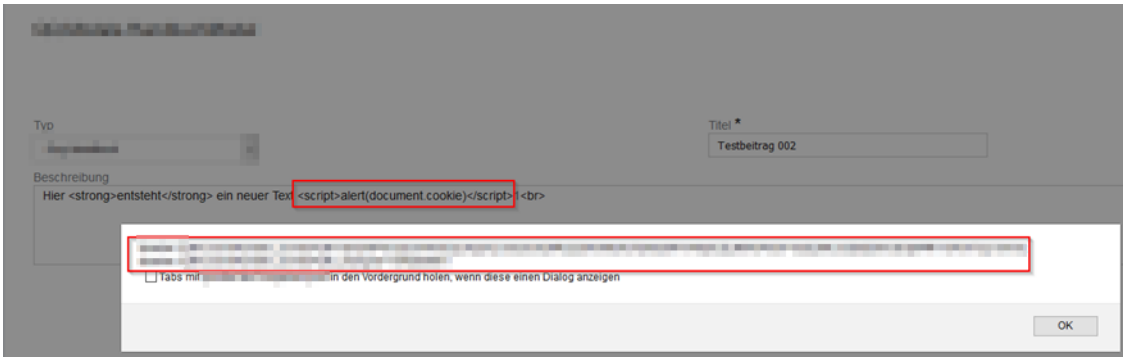


Abbildung 6 – In der Funktion „XY“ war die Eingabe von XSS-Democode direkt möglich

```
POST /musterprojekt/anlegen/1 HTTP/1.1
Host: www.musterkunde.at
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101
Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: de,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 426
Connection: close
Cookie: XXX
Upgrade-Insecure-Requests: 1

Titel=Testbeitrag+002&Beschreibung=Hier+%3Cstrong%3Eentsteht%3C%2Fstrong%3E+ein+neuer
+Text+%3Cscript%3Ealert%28document.cookie%29%3C%2Fscript%3E1%3Cbr%3E
```

Abgesendeter Request mit XSS-Democode zu Abbildung 6

Empfehlung

Um einer Web-Anwendung keine Basis für XSS-Angriffe zu bieten, müssen alle Eingabeparameter als unsicher betrachtet und vor der weiteren Verarbeitung serverseitig strikt geprüft werden. Das Definieren von unzulässigen Eingaben (Black List) stellt keine verlässliche Methode dar, da es bereits unzählige Angriffsvektoren gibt. Empfehlenswert ist stattdessen das Festlegen von zulässigen Eingabewerten (White List). Diese Vorgehensweise zählt zu den empfohlenen Vorgehensweisen (Best Practices) der Programmierung und sollte eingesetzt werden, um unerwartetes Programmverhalten zu verhindern. Außerdem sollten Nutzereingaben vor der Ausgabe auf der Webseite HTML-kodiert werden. Dabei werden Sonderzeichen (beispielsweise "<") mit den entsprechenden HTML-Entitäten (in diesem Beispiel "`<`") ersetzt, was ein Ausführen des Schadcodes verhindert.

M1	Vertikale Privilegienerweiterung	
	Risiko	**Mittel**
	CVSS-Bewertung	6,3 (CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N)
	Referenz	WSTG-ATHZ-03
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Authentifizierter Benutzer

Allgemeine Beschreibung der Schwachstelle

Die meisten Applikationen sind als Mehrbenutzersysteme mit einem Konzept für das Management von Zugriffsrechten ausgelegt. Privilegienerweiterung oder Rechteausweitung („Privilege Escalation“) bedeutet, dass ein Benutzer Rechte erhält, die er normalerweise nicht haben würde. Diese erweiterten Berechtigungen können verwendet werden, um beispielsweise Einstellungen zu manipulieren oder private Informationen anderer Benutzer abzurufen und zu verändern. Eine solche Schwachstelle tritt auf, wenn Fehler in der Applikation vorhanden sind, die eine Umgehung des Sicherheitskonzepts möglich machen. Man untscheidet zwischen horizontaler und vertikaler Rechteausweitung.

Bei der vertikalen Rechteausweitung können Benutzer der Applikation ihre Rechte erhöhen, um auf Funktionen zugreifen zu können, auf denen Ihnen im Normalfall die Rechte fehlen. Beispielsweise können normale Benutzer administrative Funktionen ausführen. Dieses Verhalten kann das Berechtigungskonzept komplett aushebeln.

Nachweis der Schwachstelle

Der zur Verfügung gestellte Benutzer **tuev8** mit der Rolle „**XY**“ kann auf die privaten Seiten aller bereitgestellten Benutzerkonten zugreifen. Dieser Zugriff ist über den Direktaufruf der Seite eines anderen Benutzers möglich, ist aber in der Weboberflächenansicht der Rolle als Option nicht abgebildet.

Die Auditoren gehen davon aus, das ein solcher Zugriff nicht möglich sein sollte. Da aber keine Benutzerhandbücher oder Spezifikationen der Applikation vorliegen, ist es zu empfehlen, das beschriebene Verhalten nochmals zu evaluieren.

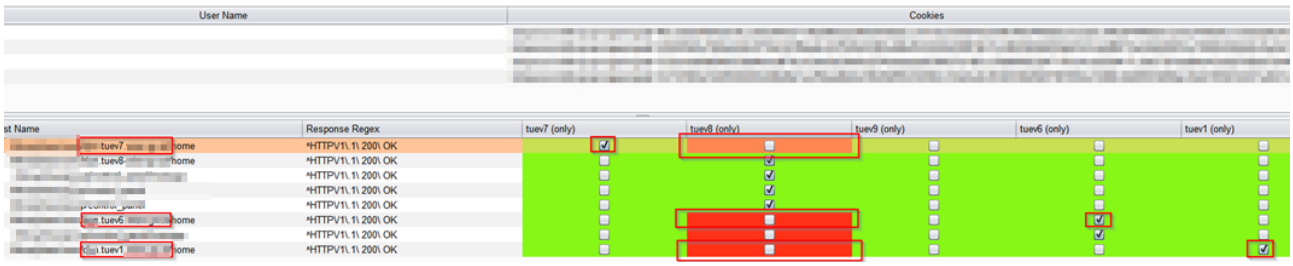


Abbildung 7 – Direktzugriff auf private Seiten alle bereitgestellten Benutzer mit dem Benutzer tuev8

Empfehlung

Es sollte gewährleistet sein, dass jeder Nutzer nur die Aktionen ausführen kann und nur die Rechte besitzt, welche er zur Ausübung seiner Tätigkeit benötigt. Ein Review der Benutzerberechtigungen, sowie eine Überprüfung des Autorisierungsverfahrens ist notwendig, um den Zugriff auf unerlaubte Daten und Funktionen zu unterbinden.

M2	Benutzerenumerierung möglich	
	Risiko	Mittel
	CVSS-Bewertung	5,3 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N)
	Referenz	WTG-IDNT-04
	Systeme	https://www.musterkunde.at/musterprojekt/Kunde/Register
	Vorbedingung	Keine

Allgemeine Beschreibung der Schwachstelle

Die Webapplikation erlaubt es, über verschiedene Rückmeldungen der Applikation festzustellen, ob der eingegebene Benutzername gültig ist. Dieses Verhalten ermöglicht es einem Angreifer, eine Liste von gültigen Benutzernamen zu erstellen. Diese Benutzernamen können dann bei weiteren Angriffen, wie beispielsweise Passwort-Bruteforcing, hilfreich sein.

Nachweis der Schwachstelle

Die Applikation liefert bei der Registrierung Hinweise, ob eine Kundennummer bereits existiert, wenn auch der Nachname des Kunden bekannt ist. Da die Kundennummern scheinbar fortlaufend vergeben werden, war es den Auditoren möglich, durch Enumerierung der Kundennummer und Benutzung gängiger Nachnamen bestehende Kundenkonten herauszufinden:

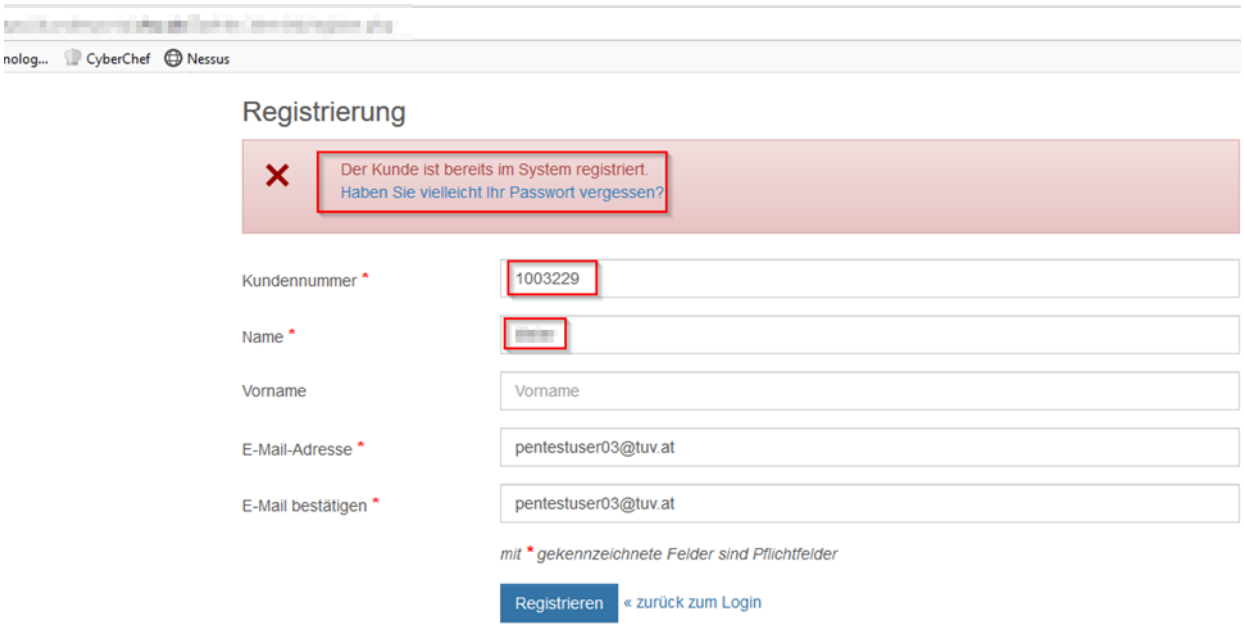


Abbildung 8 – Rückmeldung der Applikation bei einem existierenden Kunden

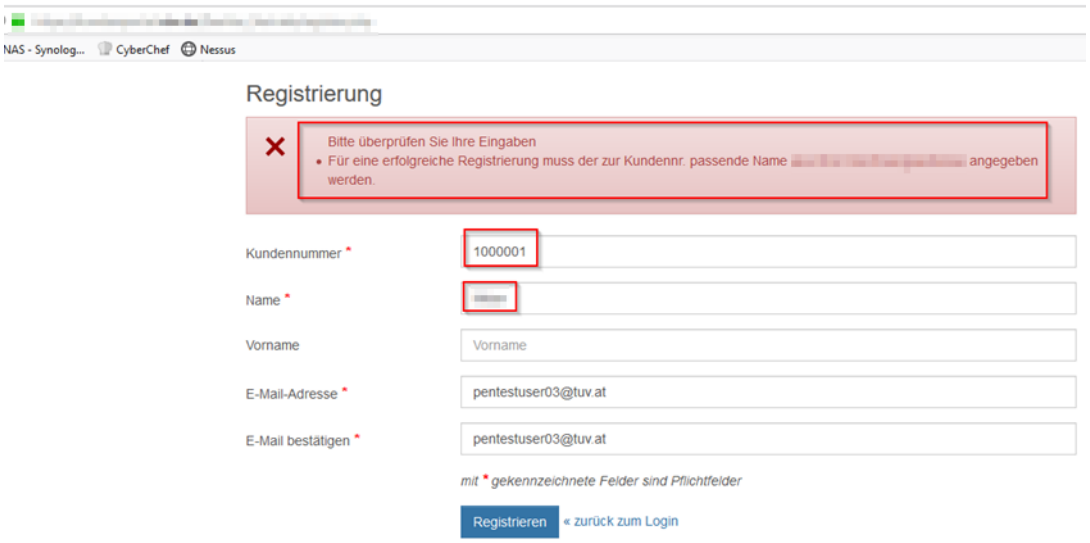


Abbildung 9 – Rückmeldung der Applikation bei einer ungültigen Kombination

Dadurch kann eine Liste an gültigen Benutzernamen erstellt werden, welche für weitere Angriffe verwendet werden kann. Folgende Benutzerkonten wurden als Beispiele im Zuge des Audits gefunden (jeweils eine Kundennummer als Beispiel pro Name):

```
Mxxxx     - 1003xxx
Mxxxx     - 1000xxx
Hxxxx     - 1001xxx
Hxxxx     - 1005xxx
Fxxxx     - 1000xxx
Wxxxx     - 1004xxx
Wxxxx     - 1000xxx
Gxxxx     - 1002xxx
```

Empfehlung

Die Applikation sollte immer mit einer generischen Fehlermeldung antworten, wenn Daten fehlen. Als Beispiel kann die Meldung „**Benutzerkonto existiert bereits oder Kombination aus Kundennummer und Name falsch**“ zurückgeliefert werden. Diese Meldung lässt keine Rückschlüsse zu, ob ein Benutzerkonto existiert.

Als Alternative könnte bei der vorliegenden Applikation der richtige Vorname verpflichtend als dritter Parameter für einen erfolgreichen Vergleich herangezogen werden. Dadurch wäre ein Erraten richtiger Kombinationen um ein vielfaches schwieriger.

M3	Einsatz schwacher Passwortrichtlinien	
	Risiko	Mittel
	CVSS-Bewertung	5,3 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N)
	Referenz	WSTG-ATHN-07
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Keine

Allgemeine Beschreibung der Schwachstelle

Der Einsatz einer guten Passwortrichtlinie setzt ein Mindest-Sicherheitsniveau für alle Benutzer und senkt die Wahrscheinlichkeit von erfolgreichen Wörterbuch- oder Brute-Force-Angriffen. Ist eine Passwortrichtlinie hingegen mangelhaft konfiguriert oder nicht vorhanden, können Benutzer beliebige Passwörter setzen. Meist sind diese einfach und leicht zu erraten, wie beispielsweise *1111*, *1234* oder *password*.

Nachweis der Schwachstelle

In der getesteten Webapplikation wurde erfolgreich folgendes schwaches Passwort gesetzt:

```
111111
```

Folgende unzureichende Mindestanforderungen an das Passwort waren in der getesteten Applikation gegeben:

- Mindestens 6 Zeichen
- Keine Anforderung an die Komplexität

Empfehlung

Der Einsatz einer Passwortrichtlinie sollte eine Mindestlänge von 8 Zeichen erfordern und den Komplexitätsvoraussetzungen entsprechen. Diese sind meist als drei von vier Zeichengruppen (Großbuchstaben, Kleinbuchstaben, Ziffern und Sonderzeichen) definiert. Nähere Informationen zu diesem Thema gibt es etwa beim BSI[2] oder der NIST [3].

Ist ein erhöhtes Sicherheitsniveau erforderlich, beispielsweise bei administrativen Logins, sollte die Passwortrichtlinie dafür verschärft werden oder das Hinzufügen eines zweiten Authentifizierungsfaktors erwogen werden.

[2] https://www.bsi.bund.de/DE/Themen/Verbraucherinnen-und-Verbraucher/Informationen-und-Empfehlungen/Cyber-Sicherheitsempfehlungen/Accountschutz/Sichere-Passwoerter-erstellen/sichere-passwoerter-erstellen_node.html
[3] https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63b.pdf

M4	Logout-Funktionalität nicht vorhanden	
	Risiko	Mittel
	CVSS-Bewertung	4,6 (CVSS:3.1/AV:A/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:N)
	Referenz	WSTG-SESS-06
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Keine

Allgemeine Beschreibung der Schwachstelle

Applikationen bieten im Normalfall eine Möglichkeit für den Benutzer, sich auszuloggen und damit die Session sicher zu beenden. Ist diese Funktionalität nicht vorhanden oder nicht sicher integriert, hat der Benutzer keine Möglichkeit, eine gültige Session zu beenden. Somit kann eine solche Session bis zum Ende ihrer Gültigkeitsdauer von einem Angreifer übernommen werden, entsprechende Angriffsvektoren vorausgesetzt.

Nachweis der Schwachstelle

Die Applikation bietet zwar eine Logout-Funktionalität, diese invalidiert die Session aber nicht korrekt. Nach dem Klick auf Logout reicht ein Betätigen des Zurück-Buttons des Browsers, um die Applikation wieder als eingeloggter Benutzer verwenden zu können.

Empfehlung

Jede Applikation sollte eine Logout-Funktionalität implementieren. Diese sollte so gestaltet sein, dass die Session nach erfolgtem Logout wirklich nicht mehr verwendet werden kann (terminiert wird).

M5	Einsatz veralteter Software	
	Risiko	Mittel
	CVSS-Bewertung	4,6 (CVSS:3.1/AV:N/AC:H/PR:L/UI:R/S:U/C:L/I:L/A:L)
	Referenz	WSTG-INFO-08
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Authentifizierter Benutzer

Allgemeine Beschreibung der Schwachstelle

Es wurden Softwareversionen identifiziert, welche nicht dem aktuellen Stand entsprechen. Der Einsatz veralteter Software kann neben einer konkreten Gefährdung der IT-Sicherheit auch ein Anzeichen für Schwächen im Patchmanagementprozess sein.

Da Applikationsversionen unter anderem mittels Banner-Informationen erhoben wurden und diese nicht immer zuverlässig sind, wäre eine manuelle Prüfung der Zielsysteme notwendig, um die Aktualität der genannten Softwarepakete sicherzustellen. Den Auditoren der TÜV TRUST IT ist es außerdem nicht immer möglich, sogenannte „Backported Security Patches“ zu ermitteln, weshalb das Zutreffen der Schwachstellen gesondert zu überprüfen sind.

Nachweis der Schwachstelle

Folgende Softwareversion der im Einsatz befindlichen Oracle-Datenbank wurde im Zuge der Ausnutzung der SQL-Injection-Schwachstelle (siehe *SQL Injection*) entdeckt:

```
Oracle Database 11g Enterprise Edition Release 11.2.0.4.0 - 64bit Production
```

Diese Version ist seit Jänner 2019 im kostenpflichtigen Extended Support[4]. Außerdem ist sie bereits von einigen bekannten Schwachstellen betroffen[5].

Empfehlung

Es sollte geprüft werden, ob ein kostenpflichtiger Extended Support-Vertrag für die weitere Versorgung mit Updates besteht. Da dieser Extended Support Ende 2020 ausläuft, wird eine rechtzeitige Aktualisierung auf eine neuere Version der Datenbank empfohlen.

[4] https://mikedietrichde.com/2017/01/27/release-dates-oracle-database-12-2-0-1-on-prem-extended-support-waiving-for-oracle-11-2-0-4-12-1-0-2/
[5] https://www.cvedetails.com/vulnerability-list/vendor_id-93/product_id-18751/version_id-193713/Oracle-Database-11.2.0.4.html

M6	Enumerierung von administrativen Interfaces	
	Risiko	Mittel
	CVSS-Bewertung	4,3 (CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:N/I:L/A:N)
	Referenz	WSTG-CONF-05
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Keine

Beschreibung

Das erfolgreiche Auffinden von administrativen Interfaces kann dazu führen, dass Angreifer Brute-Force oder Wörterbuch-Angriffe auf diese durchführen. Bei Verwendung eines Standardpasswortes oder einem schwach gewählten Passwort kann ein Angreifer so innerhalb kürzerster Zeit administrativen Zugriff erlangen. Außerdem kann es passieren, dass vorhandene Admin-Zugänge gesperrt werden und dadurch auch legitime Benutzer ausgesperrt werden.

Details

Ein SAP-Webadmin-Portal konnte auf folgender URL gefunden werden. Dieses ist durch eine HTTP-Authentifizierung geschützt.

```
https://www.musterkunde.at/musterprojekt/sap/admin/public/default.html
```

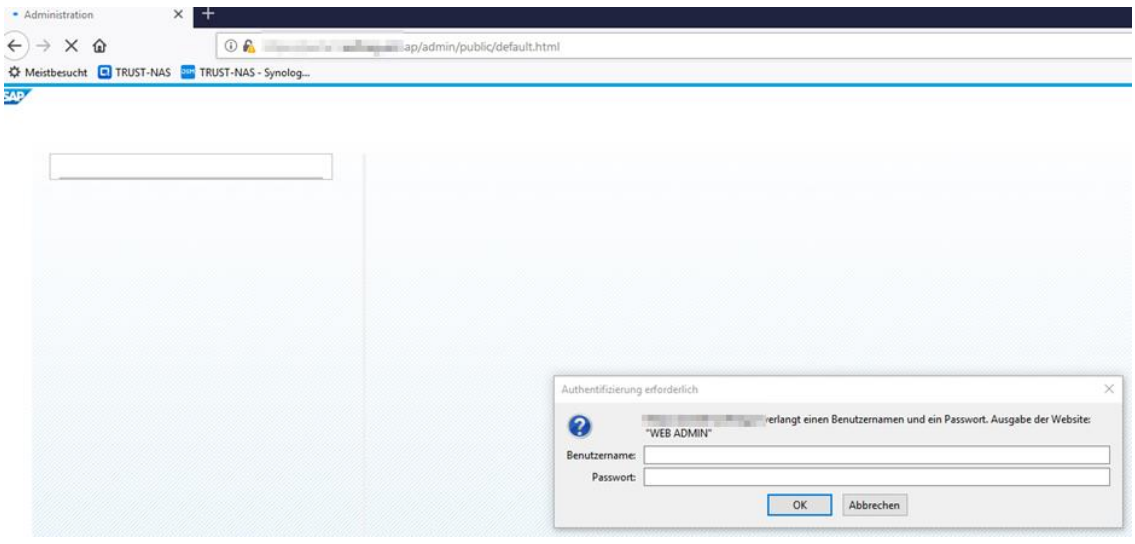


Abbildung 10 – Es konnte ein SAP-Webadmin-Portal gefunden werden

Ein Test mit üblichen Benutzername und Passwortkombinationen führte zu keinem gültigen Login. Es lässt sich aber nicht ausschließen, dass ein Angreifer mit ungleich höheren Ressourcen hier Erfolg haben kann.

Empfehlung

Es sollte überprüft werden, ob administrative Interfaces sowie Konfigurations-Übersichtsseiten über das Internet zugänglich sein müssen. Sollte das der Fall sein, sollte die Einführung von Zugangsbeschränkungen, wie beispielsweise IP-Filtering, oder die Verwendung eines VPNs erwogen werden.

G1	Upload von Malware möglich	
	Risiko	Gering
	CVSS-Bewertung	3,7 (CVSS:3.1/AV:N/AC:H/PR:L/UI:R/S:U/C:L/I:L/A:N)
	Referenz	WSTG-BUSL-09
	Systeme	https://www.musterkunde.at/musterprojekt/Dokument/Upload
	Vorbedingung	Authentifizierter Benutzer

Allgemeine Beschreibung der Schwachstelle

Es ist möglich, Dateien in die getestete Webapplikation hochzuladen. Im Zuge des Audits werden deshalb gezielt präparierte Dateien hochgeladen. Dabei kommen die EICAR-Virustestdateien[6]- zum Einsatz. Es handelt es sich um Testdateien, mit deren Hilfe die Funktionalität von Anti-Virenprogramme getestet werden kann. Die Dateien selbst stellen keine Gefahr dar, sollten aber von jedem Anti-Viren-Programm entdeckt und behandelt werden.

Prinzipiell besteht beim Fehlen einer Virenüberprüfung von hochgeladenen Dateien die Gefahr, dass schädliche Dateien am Server ausgeführt werden, was zur Kompromittierung des Systems führen kann, oder dass diese Dateien von anderen Benutzern der Applikation heruntergeladen werden, was zur Kompromittierung deren Systeme führen kann.

Nachweis der Schwachstelle

Es wurde festgestellt, dass der Upload der „EICAR“ -Virustestdatei nicht erkannt und unterbunden wird, was darauf hindeutet, dass keine Virenüberprüfung stattfindet.

[6] http://www.eicar.org/86-0-Intended-use.html

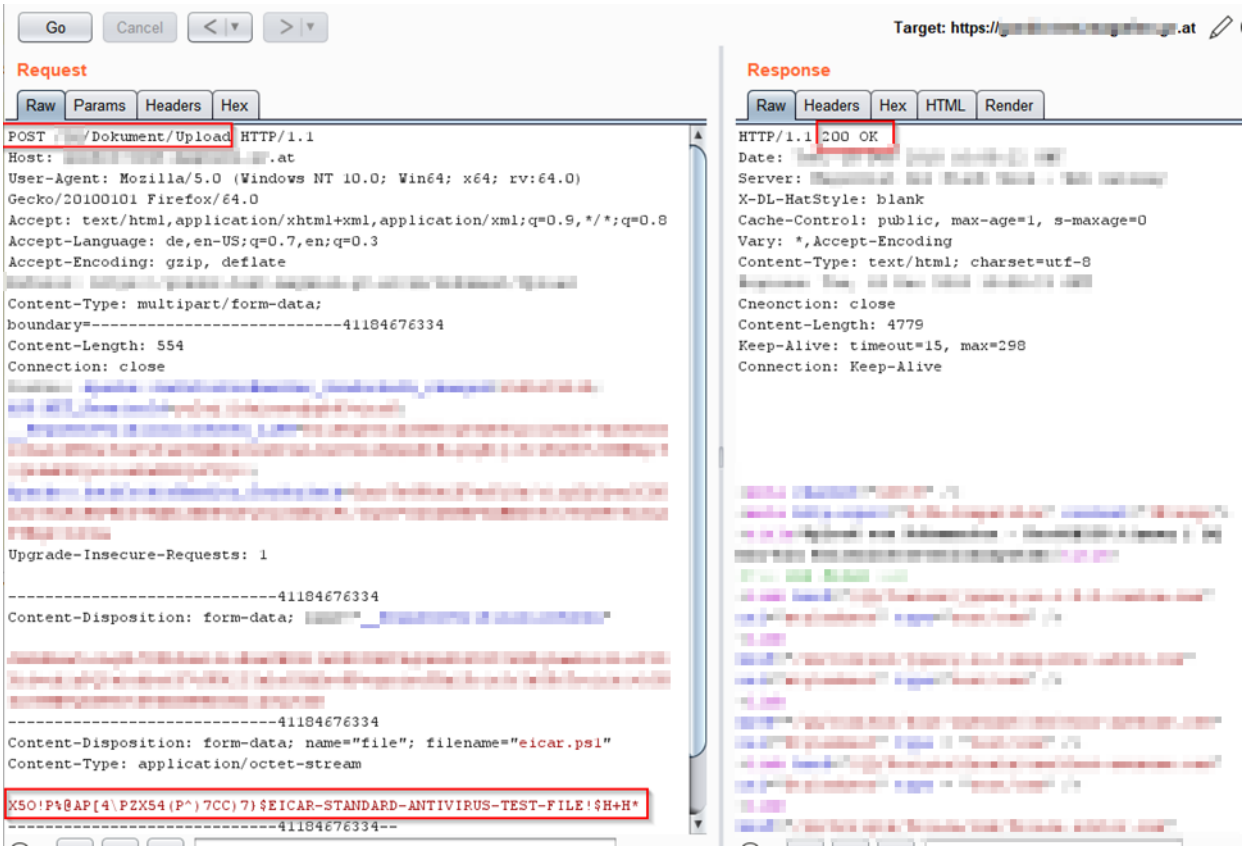


Abbildung 11 – Erfolgreicher Upload einer EICAR-Testdatei

Empfehlung

Es wird empfohlen, Dateien beim Upload von einem Anti-Virenprogramm prüfen zu lassen und den Upload bei Erkennung einer Signatur zu unterbinden.

G2	TLS 1.0-Protokoll in Verwendung	
	Risiko	Gering
	CVSS-Bewertung	3,7 (CVSS:3.1/AV:A/AC:H/PR:L/UI:N/S:U/C:L/I:L/A:N)
	Referenz	WSTG-CRYP-01
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Man-in-the-Middle-Position

Allgemeine Beschreibung der Schwachstelle

Transport Layer Security (TLS) – früher bekannt als Secure Socket Layer (SSL) - wird heutzutage vor allem im Zusammenhang mit HTTP eingesetzt. Durch die Anwendung von HTTP über TLS ist eine sichere Verbindung zwischen Webserver und Webbrowser sichergestellt. Damit können Daten authentifiziert, integritätsgesichert und vertraulich ausgetauscht werden. Das SSL-Protokoll existiert in der Version 1.0, 2.0 und 3.0. Das TLS-Protokoll gibt es in der Version 1.0, 1.1 und 1.2. Dabei stellt TLS 1.0 eine direkte Weiterentwicklung von SSL 3.0 dar.

Generell wird stets empfohlen, die höchstmögliche TLS-Version zu verwenden, um eine nach dem heutigen Stand der Technik starke Verschlüsselung der Kommunikation zu gewährleisten.

Nachweis der Schwachstelle

Folgende TLSv1.0-Ciphers werden vom Server unterstützt:

```
ECDHE-RSA-AES256-SHA
DHE-RSA-AES256-SHA
AES256-SHA
ECDHE-RSA-AES128-SHA
DHE-RSA-AES128-SHA
AES128-SHA
DES-CBC3-SHA
```

Empfehlung

Die Server sollten so konfiguriert sein, dass sie die neuesten Protokollversionen unterstützen, um sicherzustellen, dass sie nur die stärksten Algorithmen und Verschlüsselungscodes verwenden. Genauso wichtig ist es, Algorithmen und Verfahren zu deaktivieren, welche als gebrochen oder unsicher gelten und somit von Angreifern missbraucht werden könnten, um übertragene Informationen trotz eingesetztem SSL/TLS mitzulesen.

Das Deaktivieren von TLS v1.0 sollte zeitnahe in Betracht gezogen werden. Diese Empfehlung wurde vom BSI[7] ausgesprochen und wird auch vom PCI Data Security Standard (DSS) ab dem 30. Juni 2018 gefordert.

[7] https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR02102/BSI-TR-02102-2.pdf?__blob=publicationFile&v=4 (Version 2018-01)

G3	Verwendung verwundbarer Ciphers (“SWEET32”)	
	Risiko	Gering
	CVSS-Bewertung	3,7 (CVSS:3.1/AV:A/AC:H/PR:L/UI:N/S:U/C:L/I:L/A:N)
	Referenz	WSTG-CRYP-01
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Man-in-the-Middle-Position

Allgemeine Beschreibung der Schwachstelle

Der Server erlaubt die Verwendung von Block Ciphers mit 64-Bit-Blöcken (DES oder 3DES-Ciphers) und/oder Ciphers mittlerer Stärke (<112 Bit Schlüssellänge). Diese Ciphers sind mehrfach verwundbar, unter anderem auf die sogenannte „SWEET32“-Schwachstelle (**CVE-2016-2183**), welche es einem Angreifer mit ausreichenden Ressourcen erleichtert, Teile der verschlüsselten Daten zu entschlüsseln und somit beispielsweise an eine authentifizierte Session zu gelangen.

Nachweis der Schwachstelle

Folgende verwundbare Ciphers werden vom Server unterstützt:

```
DES-CBC3-SHA (TLSv1.0)
DES-CBC3-SHA (TLSv1.1)
DES-CBC3-SHA (TLSv1.2)
```

Empfehlung

Es wird empfohlen Algorithmen und Verfahren zu deaktivieren, welche als gebrochen oder unsicher gelten und somit von Angreifern missbraucht werden könnten, um zum Beispiel übertragene Informationen trotz eingesetztem SSL/TLS mitzulesen. Generell ist es empfehlenswert, in der SSL/TLS Konfiguration DES durch AES zu ersetzen.

G4	Applikationslogik: Datenvalidierung unzureichend	
	Risiko	Gering
	CVSS-Bewertung	3,1 (CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:U/C:N/I:N/A:L)
	Referenz	WSTG-BUSL-01
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Authentifizierter Benutzer

Allgemeine Beschreibung der Schwachstelle

Eine Applikation muss sicherstellen, dass nur logisch korrekte Daten eingegeben und verarbeitet werden können. Als Beispiel sollte bei einer Datumseingabe geprüft werden, ob die Eingabe dem Datumsformat entspricht und ein gültiges Datum beinhaltet. Sind solche Überprüfungen nicht implementiert, kann es passieren, dass die Applikation bei Fehleingaben abstürzt oder unerwartet reagiert.

Durch unzureichende Datenvalidierung sind verschiedene Angriffsvektoren möglich. So können beispielsweise unzulässige Daten eingegeben werden oder die Applikation zum Absturz gebracht werden.

Nachweis der Schwachstelle

Es ist möglich, logisch ungültige Daten in der Applikation zu setzen. Darunter zählt unter anderem ein Geburtsdatum, welches in der Zukunft liegt und eine ungültige E-Mail-Adresse ohne „@“. Bei manchen Parametern kann durch Senden des Requests auch die clientseitige Validierung umgangen werden, wie zum Beispiel bei der E-Mail-Adresse. Weiters wurde festgestellt, dass „null“ als allgemein gültiger Wert für erforderliche Eingabefelder akzeptiert wird, zum Beispiel für das Geburtsdatum.

Dieses Verhalten könnte zu Problemen und unvorhergesehenen Auswirkungen im Backend der Applikation führen, wie zum Beispiel Abstürze oder einer falschen Weiterverarbeitung der Daten.

Dieses Verhalten wurde bei folgendem POST-Request entdeckt:

```
POST /musterprojekt/einbringen
```

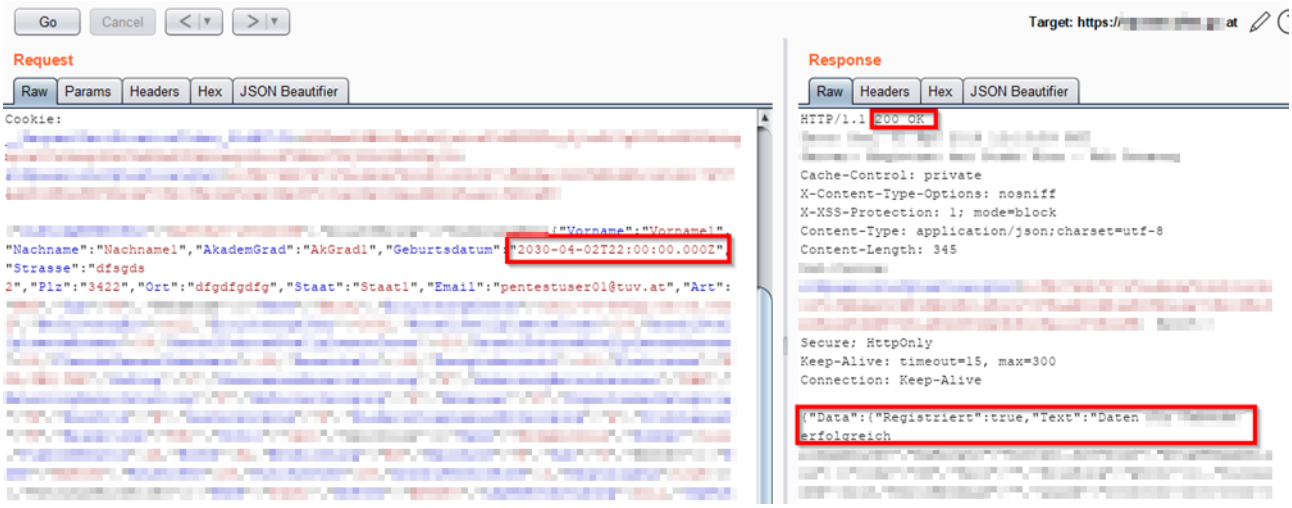


Abbildung 12 – Setzen eines Geburtsdatums in der Zukunft

Empfehlung

Alle Dateneingaben sollten auf ihre semantische und logische Plausibilität geprüft werden, bevor sie von der Applikation verarbeitet oder gespeichert werden. Es sollte außerdem darauf geachtet werden Überprüfungen serverseitig durchzuführen.

G5	Webserveridentifikation	
	Risiko	Gering
	CVSS-Bewertung	3,1 (CVSS:3.1/AV:N/AC:H/PR:L/UI:N/S:U/C:L/I:N/A:N)
	Referenz	WSTG-INFO-02
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Authentifizierter Benutzer

Allgemeine Beschreibung der Schwachstelle

Die Preisgabe von Informationen über den eingesetzten Webserver (inkl. Versionsnummer) sowie anderer verwendeter Technologien oder Frameworks ermöglicht es einen Angreifer, Verwundbarkeiten zu identifizieren und diese gegebenenfalls auszunützen.

Die Quelle dieser Informationen können sowohl HTTP-Server-Header als auch Fehlermeldungen der Applikation sein.

Nachweis der Schwachstelle

Die Fehlermeldungen der Applikation liefern genaue Informationen über die verwendete Version von ASP.NET. Dabei konnten zwei verschiedene Versionen identifiziert werden:

```
ASP.Net: 4.7.3221.0
ASP.Net: 4.7.3282.0
```

Dieselben Fehlermeldungen haben im Laufe des Audits verschiedene Versionen angezeigt. Dies deutet auf Loadbalancing mit verschiedenen Patchständen hin. Dies konnte im Zuge des Audits allerdings nicht verifiziert werden und sollte daher näher untersucht werden.

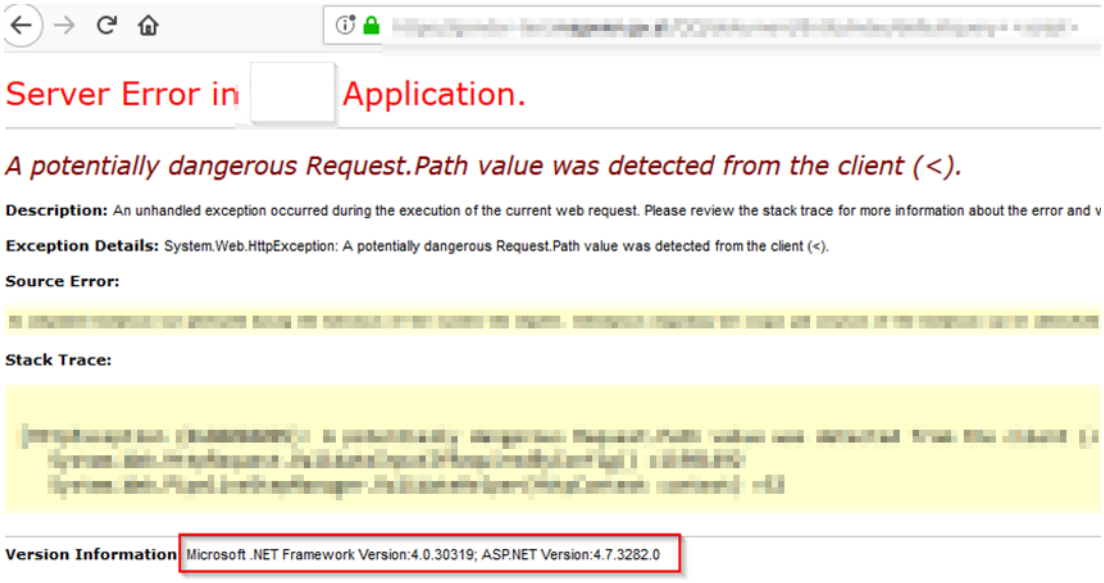


Abbildung 13 – ASP.Net-Version in einer Fehlermeldung

Empfehlung

Es sollte darauf geachtet werden, dass möglichst wenige Informationen über den eingesetzten Webserver und weitere verwendete Produkte (wie im konkreten Fall ASP.NET) preisgegeben werden, um Angriffspotentiale zu minimieren.

Weiters sollte geprüft werden, ob bei der Verwendung von Loadbalancing alle Server denselben Patchstand aufweisen.

G6	Fehlende HTTP-Sicherheitsheader	
	Risiko	Gering
	CVSS-Bewertung	3,1 (CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:U/C:N/I:L/A:N)
	Referenz	WSTG-CONF-07
	Systeme	https://www.musterkunde.at
	Vorbedingung	Benutzerinteraktion

Allgemeine Beschreibung der Schwachstelle

Durch das Setzen einiger HTTP-Sicherheitsheader kann die Sicherheit der Webapplikation maßgeblich erhöht werden. So verhindert etwa der HTTP-Strict-Transport-Security (HSTS)-Header, dass potentielle Angreifer ihre Opfer auf keine ungesicherte Seite der gleichen Domäne umleiten können.

Nachweis der Schwachstelle

Folgende HTTP-Header werden von der Applikation gesetzt:

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: application/json; charset=utf-8
Server: Not_Avail
X-AspNetMvc-Version: 5.2
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Content-Length: 658
Strict-Transport-Security: max-age=31536000
```

Empfehlung

Folgende Header sollten gesetzt werden, um das Sicherheitsniveau der Anwendung zu verbessern:

- Um unverschlüsselte Verbindungen zu vermeiden, empfiehlt es sich im HTTP-Response-Header das „`Strict-Transport-Security`“ Flag mitzusenden. Damit wird dem Browser mitgeteilt, dass in Zukunft für eine definierte Dauer (`max-age`) ausschließlich verschlüsselte Verbindungen für diese Domäne verwendet werden dürfen. Es wird Empfohlen den `max-age` Parameter auf längere Zeiträume wie zwei Jahre (63072000) zu setzten. Darüber hinaus sollte überlegt werden die Parameter `includeSubDomains` und `preload` zusätzlich zu verwenden[8].
- Der HTTP-Header "`X-Content-Type-Options`" verhindert, dass Webbrowser selbstständig versuchen, den Inhalt der vom Server zurückgelieferten Antwort zu interpretieren, falls diese nicht oder fehlerhaft definiert sind. Dieses Verhalten kann zu einer Anfälligkeit auf Cross-Site-Scripting

[8] https://infosec.mozilla.org/guidelines/web_security#http-strict-transport-security

(XSS)-Angriffe beitragen und sollte daher durch Setzen dieses Headers vermieden werden. Der HTTP-Header sollte in allen Antworten gesetzt werden, vorrangig aber bei Antworten, die Nutzereingaben entgegennehmen oder ausgeben.

```
X-Content-Type-Options: nosniff.
```

- Um Clickjacking-Angriffe zu verhindern, kann man im HTTP-Response-Header den `X-FRAME-Options`-Header auf den Wert "`SAMEORIGIN`" oder "`DENY`" setzen. Dadurch wird die Darstellung in Frames gänzlich verboten (`DENY`) oder nur von derselben Domäne erlaubt (`SAMEORIGIN`). Hierbei sollte jedoch beachtet werden, dass dieser Header als veraltet gilt und durch die `frame-ancestors` Direktive des `Content-Security-Policy`-Headers ersetzt wird. Da jedoch noch nicht alle Browser unterstützt werden, können beide Header auch parallel verwendet werden[9].

- Um feingranular zu steuern, von welchen Quellen Inhalte und Skripte zugelassen sind, empfiehlt es sich den `Content-Security-Policy`-Header zu setzen. Der Einsatz von „`unsafe-inline`“ sollte hierbei vermieden werden. Beispielhaft erlaubt der unterhalb dargestellte Header sämtlichen Content von der eigenen Seite, Bilder von sämtlichen Quellen, Medien wie Audio oder Video nur von media1.com und media2.com und Scripte nur von der angegebenen, spezifizierten Quelle.
  `Content-Security-Policy: default-src 'self'; img-src *; media-src media1.com media2.com; script-src userscripts.example.com`

  Um zu verhindern dass eine beliebige Domain den Inhalt einrahmt wird die frame-ancestors Direktive empfohlen, auch wenn diese noch nicht von allen Browsern unterstützt wird[10].

  ```
  Content-Security-Policy: frame-ancestors 'none'
  ```

- Das Zwischenspeichern von Webinhalten wird als Usability-Feature von Applikationen unterstützt. Dies kann jedoch von lokalen Angreifern (z.B. geteilter Arbeitsrechner oder öffentlich zugänglicher Computer) ausgenutzt werden, um Einsicht in möglicherweise sensible Daten zu erhalten.

  Um den Zugriff der in der History gespeicherten Daten, welche über die Schaltfläche „Zurück“ erreichbar sind, zu verhindern, kann die Seite über HTTPS aufgerufen oder der `Cache-Control`-Header verwendet werden[11].

  ```
  Cache-Control: must-revalidate
  ```

  Um den Zugriff von sensiblen Daten im Browser-Cache zu verhindern, sollte der `Cache-Control`-Header wie folgt gesetzt werden:

  ```
  Cache-Control: no-cache, no-store
  Expires: 0
  Pragma: no-cache
  ```

[9] https://infosec.mozilla.org/guidelines/web_security#x-frame-options
[10] https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Security-Policy/frame-ancestors
[11] https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/04-Authentication_Testing/06-Testing_for_Browser_Cache_Weaknesses

G7	Alter Softwarestand - Bibliotheken	
	Risiko	Gering
	CVSS-Bewertung	2,6 (CVSS:3.1/AV:N/AC:H/PR:L/UI:R/S:U/C:N/I:L/A:N)
	Referenz	WSTG-INFO-08
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Verwendung der verwundbaren Funktion

Allgemeine Beschreibung der Schwachstelle

Der Einsatz von veralteten Bibliotheken, wie etwa jQuery, kann in der integrierten Webapplikation zu einer Vielzahl von Schwachstellen führen. Diese spiegeln sich meistens auf Grund fehlender Eingabeüberprüfung in XSS-Attacken wieder. Eine solche Schwachstelle könnte von einem Angreifer missbraucht werden, um unter anderen Benutzer-Sessions zu stehlen oder den Browser des Benutzers zu übernehmen.

Das bloße Einbinden von Bibliotheken stellt meist keine Schwachstelle dar. Diese treten erst auf, wenn sich die verwundbare Funktion im Einsatz befindet.

Nachweis der Schwachstelle

Folgende veraltete und verwundbare Versionen von Bibliotheken wurden von der Webapplikation eingebunden. Es sollte geprüft werden, ob die verwundbaren Stellen auch in Verwendung sind:

```
/musterprojekt/Scripts/jquery-3.3.1
/musterprojekt/Scripts/jquery.easyui1.5.1.min.js
/musterprojekt/Scripts/bootstrap.min.js
```

Veraltete Softwareversionen	Weiterführende Links
JQuery (http://jquery.com/) Version: **3.3.1**	https://snyk.io/test/npm/jquery/3.3.1
JQuery Easy UI Version: **1.5.1**	Keine bekannten Schwachstellen, aber veraltete Version
Bootstrap Version: **4.1.3**	https://snyk.io/test/npm/bootstrap/4.1.3

Empfehlung

Alte Versionen von Bibliotheken sollten auf den aktuellen Stand gebracht werden, um das Risikio von benutzerseitigen Schwachstellen zu minimieren. Sollte dies nicht möglich sein, ist es empfehlenswert, kompensierende Maßnahmen zu treffen.

I1	Unbekannte Zertifizierungsstelle	
	Risiko	Information
	CVSS-Bewertung	N/A
	Referenz	WSTG-CRYP-001
	Systeme	https://www.musterkunde.at/musterprojekt
	Vorbedingung	Man-in-the-Middle-Position

Allgemeine Beschreibung der Schwachstelle

Die ersichtliche X.509 Zertifikatskette für diesen Dienst ist von keiner anerkannten, vertrauenswürdigen Zertifizierungsstelle (CA) signiert worden.

SSL-Zertifikate werden eingesetzt, um eine Verschlüsslung zu garantieren. Obwohl selbstsignierte SSL-Zertifikate auch Log-in- und andere persönliche Anmeldeinformationen verschlüsseln, veranlassen sie die meisten Webserver, eine Sicherheitswarnung anzuzeigen, da das Zertifikat nicht von einer vertrauenswürdigen Zertifizierungsstelle verifiziert wurde. Das Hauptproblem von selbstsignieren Zertifikaten sind aus Security-Sicht Man-in-the-Middle (MitM)-Angriffe, da sich jeder ein selbstsigniertes Zertifikat ausstellen und verwenden kann.

Nachweis der Schwachstelle

Das verwendete Serverzertifikat wird als nicht sicher eingestuft, da es nicht bis zu einer bekannten Zertifizierungsstelle rückverfolgbar ist:

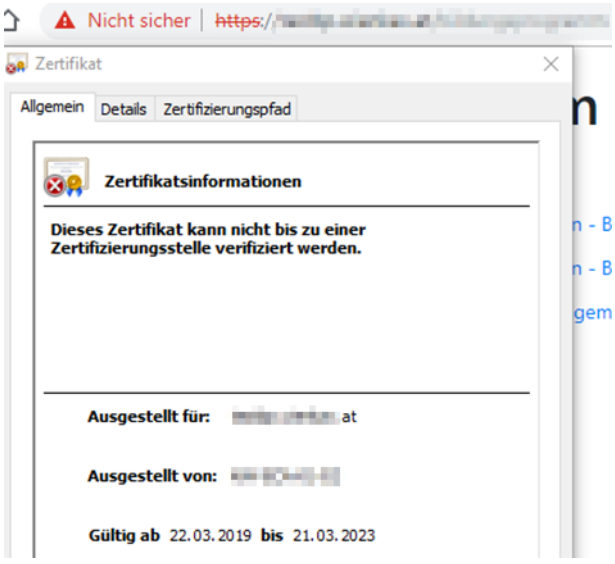


Abbildung 14 – Unsicheres Serverzertifikat

Da es sich beim getesteten Server um ein Testsystem handelt, wurde dies nur als „Empfehlung“ und nicht als Risiko eingestuft.

Empfehlung

Es sollte darauf geachtet werden, dass am Produktivsystem ein sicheres Zertifikat in Verwendung ist.

4 Allgemeine Hinweise

Das technische Sicherheitsaudit basiert auf technischen Sicherheitsanalysen der in Kapitel *Testabdeckung* genannten Systeme. Das Sicherheitsaudit war daher, sowohl aus Sicht des Geltungsbereichs als auch bezogen auf den Zeithorizont begrenzt.

Im Zuge des technischen Sicherheitsaudits wurden die sicherheitsrelevanten Informationen direkt im Rahmen der Analysetätigkeit ermittelt. Prinzipbedingt lassen sich hier ausschließlich Schwachstellen aufdecken, nicht jedoch Beweise für deren Nichtgegebenheit erbringen.

Die Leistungen des TÜV stützen sich auf die zur Verfügung gestellten Informationen und Dokumentationen. Die Auditoren gehen davon aus, dass diese vollständig und richtig sind. Auch werden alle daraus bezogenen Auskünfte zunächst als grundsätzlich Wahr angenommen.

Die im Bericht dargestellten Sachverhalte beziehen sich ausschließlich auf die im Prüfzeitpunkt vorgefundene Prüfsituation. Alle nachträglichen Änderungen an dieser Situation und der damit verbundenen Sachverhalte bleiben, soweit nicht ausdrücklich erwähnt, unberücksichtigt.

Trotz größtmöglicher Sorgfalt ist es daher möglich, dass nicht alle Sicherheitsaspekte analysiert bzw. alle vorhandenen Schwachstellen gefunden werden konnten. TÜV TRUST IT schließt daher jede Haftung für vorhandene und nicht erkannte Risiken aus.

Die Ergebnisse der Untersuchung entbinden den Kunden in keiner Weise von der Weiterverfolgung der Sicherheitsziele. Der Kunde ist in jedem Fall für die durchgeführten Maßnahmen zur Behebung von Schwachstellen und Sicherstellung der Sicherheitsziele selbst verantwortlich.

Jede Haftung von TÜV TRUST IT für eventuelle Schäden, die aus einer falschen Anwendung der hier gegebenen Informationen resultieren, wird ausgeschlossen.

Abbildungsverzeichnis

Anhänge

A.1 Permission to Attack

TÜV TRUST IT
TÜV AUSTRIA Group

„Permission to Attack“
(Einverständniserklärung / Haftungsausschluss)

Auftraggeber - Unternehmensname und Ansprechpartner:

Firma
Adresse
xxxx Stadt

Auftragnehmer - Unternehmen, welches die Sicherheitsüberprüfung durchführt:

TÜV TRUST IT TÜV AUSTRIA GmbH
TÜV AUSTRIA-Platz 1
2345 Brunn am Gebirge

Der Auftraggeber erteilt dem Auftragnehmer die Berechtigung zum Zweck einer Sicherheitsüberprüfung auf in Netzwerkressourcen und -systemen gespeicherte und verarbeitete Informationen zuzugreifen. („Permission to Attack“). Diese Berechtigung ermöglicht es dem Auftragnehmer alle zumutbaren Mittel, wie sie der Auftraggeber definiert, für eine erfolgreiche Durchführung der Dienstleistung einzusetzen, einschließlich der Verwendung von automatisierten Tools und Skripten bezüglich der folgenden IP-Adressen, Applikationen und Systeme:

Bezeichnung	Zielsysteme	Umgebung (Prod, Test, QS, ...)
tbd	tbd	Prod

*) Falls nichts anderes vereinbart, siehe „Ziele“ / “Einschränkungen“ unterhalb

Diese Vereinbarung gilt zeitlich befristet ab xx.xx.xxxx, 00.00 Uhr bis zur Übergabe des betreffenden Abschlussberichts.

Die Sicherheitsanalysen werden bei externen Untersuchungen von folgenden IP-Adressen aus durchgeführt:
212.183.39.185
212.183.39.186
212.183.39.187
212.183.39.188
212.183.39.189
212.183.39.190

Vereinbarte Ziele der Überprüfung:

Beim Penetrationstest erfolgt eine Suche nach Sicherheitsschwachstellen in den genannten Systemen. Dabei werden folgende Parameter festgelegt:

Parameter	Festlegung
Informationsbasis	Black-Box, Grey-Box, White-Box
Aggresivität	Passiver Scan, Aktiver Scan, Vorsichtiger Penetrationstest, Aggressiver Penetrationstest
Umfang	Vollständig, Begrenzt, Fokussiert (Time-Boxed)
Vorgehensweise	Verdeckt, Offensichtlich
Technik	Netzwerkzugang, physischer Zugang, Social-Engineering Ev. Freischaltungen (WAF, IDS/IPS, …)
Ausgangspunkt	Intern, Extern

A20xxxxxx_Permission-to-Attack.docx Seite 1 von 2

Einschränkungen bei der Überprüfung:

Es erfolgen keine gezielten DoS-Attacken auf Produktionssysteme.

Haftungsausschluss

Der Auftraggeber versichert hiermit, dass er die vollumfänglichen und uneingeschränkten Rechte an den aufgeführten IT Systemen innehat.

Der Auftragnehmer haftet für Schäden des Auftraggebers nur, soweit diese durch vorsätzliche oder grob fahrlässige Handlungen, oder durch die Verletzung wesentlicher Vertragspflichten verursacht worden sind. Der Auftraggeber erklärt, gegen eventuelle Datenverluste geeignete und dem Stand der Technik entsprechende Vorkehrungen getroffen zu haben sowie, sofern weitere Dienstleister von dem Penetrationstest betroffen sind, deren Einverständnis vor Beginn des Penetrationstests eingeholt zu haben. Der Auftraggeber bestätigt, dass der Auftragnehmer in seinem Auftrag tätig wird und es sich insofern nicht um strafrechtlich relevante Rechtsgutsverletzungen handelt. Änderungen oder Ergänzungen dieser Bestimmung, sowie zwischen den Parteien getroffene Vereinbarungen, bedürfen der Schriftform. Für die Änderung des Schriftformerfordernisses bedarf es ebenfalls der Schriftform. Mündliche Nebenabreden werden nicht getroffen. Sollten einzelne Teile dieser Vereinbarung ganz oder teilweise unwirksam oder nichtig sein oder werden, so wird dadurch die Wirksamkeit der Vereinbarung im Übrigen nicht berührt. Die Parteien verpflichten sich, die unwirksame oder nichtige Bestimmung durch eine wirksame Bestimmung zu ersetzen, die dem gewollten wirtschaftlichen Zweck am nächsten kommt.
Wegen Beschränkungen der zeitlichen, finanziellen und personellen Ressourcen besteht keine Gewährleistung für den Auftraggeber, dass alle vorhandenen Fehler und Schwachstellen vom Auftragnehmer gefunden werden.

Kontaktpersonen beim Auftragnehmer

Organisatorische Kontakte bzw. Projektleiter der Vertragspartner

Auftraggeber	Auftragnehmer
Hr./Fr. Tel: +43 E-Mail:	DI Rainer Seyer, BSc Tel: +43 664 60454 6231 E-Mail: rainer.seyer@tuv.at

Technische IT-Notfallkontakte

Auftraggeber	Auftragnehmer
Hr./Fr. Tel : +43 E-Mail:	DI Rainer Seyer, BSc Tel: +43 664 60454 6231 E-Mail: rainer.seyer@tuv.at
	DI Mario Rubak, BSc Tel: +43 664 60454 6238 E-Mail: mario.rubak@tuv.at

_____ _____

Ort, Datum Unterschrift (firmenmäßige Zeichnung)

_____

Name in Blockbuchstaben
Auftraggeber

A.2 Applikation - Erscheinungsbild

Dieser Abschnitt gibt einen Überblick, wie die Applikation zum Testzeitpunkt ausgesehen hat und welche Menüpunkte verfügbar waren, um eine Referenz für etwaige, zukünftige Tests zu schaffen:

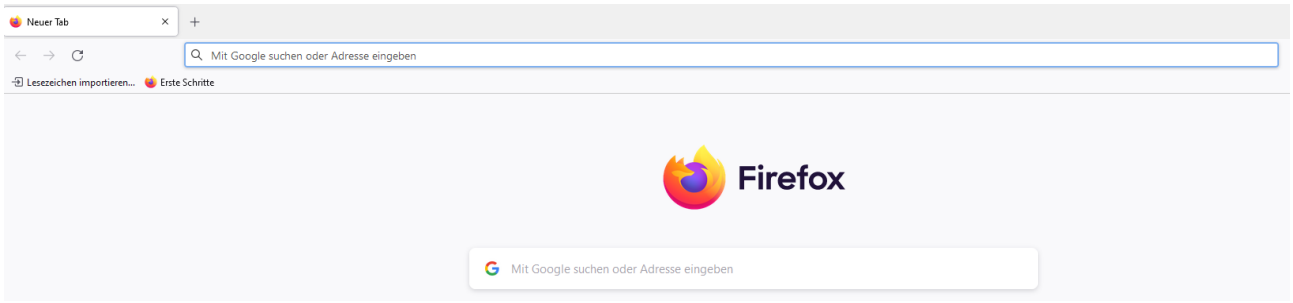


Abbildung 15 – Erscheinungsbild der Applikation

A.3 Port- und Serviceliste

Folgendes Listing ist eine Ausgabe von Nmap, um den Stand der Infrastruktur zum Testzeitpunkt zu dokumentieren:

```
Nmap scan report for www.musterkunde.at (1.2.3.4)
Host is up (0.066s latency).

PORT     STATE    SERVICE       VERSION
21/tcp   open     ftp           ProFTPD
22/tcp   open     ssh           ProFTPD mod_sftp (protocol 2.0)
23/tcp   closed   telnet
25/tcp   open     smtp
80/tcp   open     http          Apache httpd
110/tcp  open     pop3          Courier pop3d
139/tcp  filtered netbios-ssn
443/tcp  open     ssl/http      Apache httpd
445/tcp  filtered microsoft-ds
3389/tcp filtered ms-wbt-server
```

A.4 OWASP WSTG

Dieses Kapitel liefert alle Kategorien des OWASP Web Security Testing Guide[12] in der Version 4.2 sowie die von TÜV Trust IT verwendeten, ergänzenden Kategorien.

WSTG-ID	Bezeichnung
Sicherer Umgang mit Informationen (INFO)	
WSTG-INFO-01	Sammeln von Informationen über Suchmaschinen
WSTG-INFO-02	Webserveridentifikation
WSTG-INFO-03	Metadateien: Preisgabe von Informationen
WSTG-INFO-04	Applikations-Enumerierung am Webserver
WSTG-INFO-05	Informationen in Kommentaren oder Metadaten
WSTG-INFO-06	Identifizierung von Einstiegspunkten der Applikation
WSTG-INFO-07	Identifizierung von Angriffsvektoren
WSTG-INFO-08	Identifizierung veralteter Software
WSTG-INFO-09	Webapplikation: Identifizierungsmerkmale
WSTG-INFO-10	Applikationsarchitektur
Sichere Serverkonfiguration (CONF)	
WSTG-CONF-01	Netzwerk/Infrastruktur-Konfigurationsfehler
WSTG-CONF-02	Applikations-Konfigurationsfehler
WSTG-CONF-03	Unsicherer Umgang mit Dateitypen
WSTG-CONF-04	Sensitive Informationen in alten oder nicht referenzierten Dateien
WSTG-CONF-05	Enumerierung von administrativen Interfaces
WSTG-CONF-06	HTTP TRACE/TRACK-Methoden im Einsatz
WSTG-CONF-07	HTTP Strict Transport Security
WSTG-CONF-08	RIA Cross Domain Policy
WSTG-CONF-09	Unsichere Dateirechte
WSTG-CONF-10	Übernahme von Sub-Domains
WSTG-CONF-11	Cloud-Speicherung
WSTG-CONF-12	Überprüfen der Content Security Policy

[12] https://owasp.org/www-project-web-security-testing-guide/stable/

Sicheres Identitätsmanagement (IDNT)	
WSTG-IDNT-01	Rollendefinitionen
WSTG-IDNT-02	Unsichere Benutzerregistrierung
WSTG-IDNT-03	Unsichere Account-Provisionierung
WSTG-IDNT-04	Benutzerenumierung möglich
WSTG-IDNT-05	Benutzernamen einfach erratbar
Sicherer Anmeldeprozess (ATHN)	
WSTG-ATHN-01	Unverschlüsselte Übertragung von Authentifizierungsdaten
WSTG-ATHN-02	Verwendung von Standard-Zugangsdaten
WSTG-ATHN-03	Fehlender Lock-Out-Mechanismus
WSTG-ATHN-04	Umgehung der Authentifizierung
WSTG-ATHN-05	Schwäche in der Passwortspeicherung
WSTG-ATHN-06	Zwischenspeicherung sensibler Webinhalte
WSTG-ATHN-07	Einsatz schwacher Passwortrichtlinien
WSTG-ATHN-08	Einsatz schwacher Sicherheitsfragen
WSTG-ATHN-09	Schwäche in Passwortänderung
WSTG-ATHN-10	Schwächere Authentifizierung in alternativen Kanälen
Sichere Zugriffssteuerung (ATHZ)	
WSTG-ATHZ-01	Directory Traversal/File Include-Schwachstelle
WSTG-ATHZ-02	Umgehung des Autorisierungsschemas
WSTG-ATHZ-03	Privilegienerweiterung
WSTG-ATHZ-04	Unsichere direkte Objektreferenz
Umgang mit Benutzersitzungen (SESS)	
WSTG-SESS-01	Umgehung des Session-Managements
WSTG-SESS-02	Fehlende Cookie-Attribute
WSTG-SESS-03	Session Fixation
WSTG-SESS-04	Unzureichender Schutz des Session-Token
WSTG-SESS-05	Cross Site Request Forgery (CSRF)
WSTG-SESS-06	Problem in Logout-Funktion
WSTG-SESS-07	Lange Session-Laufzeit
WSTG-SESS-08	Session Puzzling
WSTG-SESS-09	Session Hijacking

Überprüfung von Benutzereingaben (INPV)	
WSTG-INPV-01	Reflected Cross Site Scripting (XSS)
WSTG-INPV-02	Stored Cross Site Scripting (XSS)
WSTG-INPV-04	HTTP Parameter Pollution
WSTG-INPV-05	SQL Injection
WSTG-INPV-06	LDAP Injection
WSTG-INPV-07	XML Injection
WSTG-INPV-08	SSI Injection
WSTG-INPV-09	XPath Injection
WSTG-INPV-10	IMAP/SMTP Injection
WSTG-INPV-11	Code Injection
WSTG-INPV-12	Command Injection
WSTG-INPV-13	Format String Injection
WSTG-INPV-14	Incubated Vulnerability
WSTG-INPV-15	HTTP Splitting Smuggling
WSTG-INPV-16	HTTP Incoming Requests
WSTG-INPV-17	Host Header Injection
WSTG-INPV-18	Server-side Template Injection
WSTG-INPV-19	Server-side Request Forgery
Korrekte Fehlerbehandlung (ERRH)	
WSTG-ERRH-01	Detaillierte Fehlermeldungen
WSTG-ERRH-02	Preisgabe von Stack Traces
Verwendung von Kryptografie (CRYP)	
WSTG-CRYP-01	Unsichere kryptografische Verfahren
WSTG-CRYP-02	Padding Oracle
WSTG-CRYP-03	Unverschlüsselte Übertragung sensibler Daten
WSTG-CRYP-04	Verwendung schwacher Verschlüsselung

Logische Überprüfung von Abläufen (BUSL)	
WSTG-BUSL-01	Applikationslogik: Datenvalidierung unzureichend
WSTG-BUSL-02	Manipulation von Requests
WSTG-BUSL-03	Applikationslogik: Integritätsprüfungen unzureichend
WSTG-BUSL-04	Informationsgewinnung durch Verarbeitungszeitunterschiede
WSTG-BUSL-05	Limitierung der Häufigkeit der Verwendung einer Funktion
WSTG-BUSL-06	Umgehung von Workflows
WSTG-BUSL-07	Fehlender Schutz gegen missbräuchliche Verwendung
WSTG-BUSL-08	Erlaubte Arten von Dateianhängen
WSTG-BUSL-09	Upload von Malware möglich
Benutzerseitige Absicherung (CLNT)	
WSTG-CLNT-01	DOM-Based Cross Site Scripting (XSS)
WSTG-CLNT-02	Ausführung von JavaScript
WSTG-CLNT-03	HTML Injection
WSTG-CLNT-04	Open Redirection
WSTG-CLNT-05	CSS Injection
WSTG-CLNT-06	Client-side Resource Manipulation
WSTG-CLNT-07	Cross Origin Resource Sharing (CORS)
WSTG-CLNT-08	Cross Site Flashing
WSTG-CLNT-09	Verwundbarkeit auf Clickjacking
WSTG-CLNT-10	Verwundbarkeit in Websockets
WSTG-CLNT-11	Web Messaging
WSTG-CLNT-12	Unsichere Verwendung des Browser Storage
WSTG-CLNT-13	Cross Site Script Inclusion
API-Überprüfung (APIT)	
WSTG-APIT-01	Unsichere Verwendung von GraphQL

A.5 CVSS

Dieses Kapitel liefert alle Metriken des für die Schwachstellenbewertung verwendeten Common Vulnerability Scoring System [13] in der Version 3.1:

Metrik-Gruppe Base Score

Der Base Score (Basiswert) gibt Auskunft über die Charakteristiken einer Schwachstelle, die unabhängig von der Zeit oder der Umgebung sind. Dieser Wert setzt sich aus folgenden Elementen zusammen:

Wert	Erklärung und mögliche Werte
AV – Attack Vector **Angriffsvektor**	Dieser Wert definiert den Kontext, in dem die Ausnutzung einer Schwachstelle möglich ist. Mögliche Werte sind: o **N** Network (**Netzwerk**): Die Komponente, welche von der Schwachstelle betroffen ist, ist vom Netzwerk aus erreichbar. o **A** Adjacent (**Angrenzend**): Die Komponente, welche von der Schwachstelle betroffen ist, ist vom selben Netzwerksegment aus erreichbar. o **L** Local (**Lokal**): Die Komponente, welche von der Schwachstelle betroffen ist, ist nur von lokalen Systembenutzern erreichbar. o **P** Physical (**Physisch**): Die Komponente, welche von der Schwachstelle betroffen ist, ist nur mit direktem, physischen Zugriff erreichbar.
AC– Attack Complexity **Angriffskomplexität**	Dieser Wert gibt an, wieviel Informationen und Voraussetzungen vorhanden sein müssen, um einen Angriff zu ermöglichen. Mögliche Werte sind: o **L** Low (**Gering**): Es existieren keine speziellen Voraussetzungen für einen Angriff. o **H** High (**Hoch**): Es müssen spezielle Voraussetzungen gegeben sein, um einen Angriff durchführen zu können.

[13] https://www.first.org/cvss/v3.1/specification-document

Wert	Erklärung und mögliche Werte
PR – Privileges Required **Benötigte Berechtigungen**	Dieser Wert definiert das Level an Berechtigungen, welche für die Durchführung eines Angriffes notwendig sind. Mögliche Werte sind: o **N** None (**Keine**): Das Ausnutzen der Schwachstelle ist ohne Autorisierung möglich. o **L** Low (**Gering**): Das Ausnutzen der Schwachstelle ist mit den Rechten eines normalen Benutzers möglich. o **H** High (**Hoch**): Das Ausnutzen der Schwachstelle ist nur mit erhöhten Rechten möglich.
UI – User Interaction **Benutzerinteraktion**	Dieser Wert gibt an, ob eine Benutzerinteraktion für einen Angriff notwendig ist. Mögliche Werte sind: o **N** None (**Nicht notwendig**): Das Ausnutzen der Schwachstelle ist ohne die Interaktion eines Benutzers möglich. o **R** Required (**Notwendig**): Zum Ausnutzen der Schwachstelle ist es notwendig, dass ein Benutzer Aktionen setzt.
S – Scope **Prüfumfang**	Dieser Wert gibt an, ob durch einen Angriff abgesehen von der verwundbaren Komponente auch andere Komponenten betroffen sind. Mögliche Werte sind: o **U** Unchanged (**Nicht verändert**): Ein Angriff betrifft nur die verwundbare Komponente. o **C** Changed (**Verändert**): Ein Angriff kann auch andere, nicht verwundbare Komponenten beeinträchtigen.
C – Confidentiality **Vertraulichkeit**	Dieser Wert definiert die Auswirkung auf die Vertraulichkeit von Informationsressourcen im Falle eines Angriffes. Mögliche Werte sind: o **N** None (**Keine**): Kein Verlust der Vertraulichkeit. o **L** Low (**Gering**): Es ist mit einem Teilverlust vertraulicher Informationen zu rechnen. o **H** High (**Hoch**): Es ist mit einem vollständigen Verlust vertraulicher Informationen zu rechnen.

Wert	Erklärung und mögliche Werte
I – Integrity **Integrität**	Dieser Wert definiert die Auswirkung auf die Integrität von Informationsressourcen im Falle eines Angriffes. Mögliche Werte sind: o **N** None (**Keine**): Kein Verlust der Integrität. o **L** Low (**Gering**): Es ist mit einem Teilverlust der Integrität von Informationen zu rechnen. o **H** High (**Hoch**): Es ist mit einem vollständigen Verlust der Integrität von Informationen zu rechnen.
A – Availability **Verfügbarkeit**	Dieser Wert definiert die Auswirkung auf die Verfügbarkeit von Informationsressourcen im Falle eines Angriffes. Mögliche Werte sind: o **N** None (**Keine**): Kein Verlust der Verfügbarkeit. o **L** Low (**Gering**): Es ist mit einem Teilverlust der Verfügbarkeit von Informationen zu rechnen. o **H** High (**Hoch**): Es ist mit einem vollständigen Verlust der Verfügbarkeit von Informationen zu rechnen.

Metrik-Gruppe Temporal Score

Der Temporal Score (Zeitwert) gibt Auskunft über die Charakteristiken einer Schwachstelle, welche sich im Lauf der Zeit unter anderem aufgrund von verfügbaren Patches oder Exploits ändern können. Dieser Wert modifiziert den vorher berechneten Base Score und setzt sich aus folgenden Elementen zusammen:

Wert	Erklärung und mögliche Werte
E – Exploit Code Maturity **Verfügbarkeit des Exploit Codes**	Dieser Wert gibt an, wie wahrscheinlich ein Angriff durch die Verfügbarkeit eines Exploits ist. Mögliche Werte sind: o **X** Not Defined (**Nicht definiert**): Keine Angabe und daher keine Veränderung des Base Score. o **U** Unproven (**Nicht bewiesen**): Die Exploitverfügbarkeit ist nicht bekannt oder nicht gegeben. o **P** Proof-of-Concept (**Machbarkeitsbeweis**): Ein Machbarkeitsnachweis ist verfügbar, aber nicht zur großflächigen Ausnutzung geeignet. o **F** Functional (**Funktional**): Ein Exploitcode ist verfügbar und in den meisten Fällen funktionsfähig. o **H** High (**Hoch**): Ein Exploitcode ist verfügbar und immer funktionsfähig oder nicht notwendig.
RL – Remediation Level **Status der Behebung**	Dieser Wert gibt an, ob es bereits Möglichkeiten der Behebung der Schwachstelle, etwa durch Patches, gibt. Mögliche Werte sind: o **X** Not Defined (**Nicht definiert**): Keine Angabe und daher keine Veränderung des Base Score. o **O** Official Fix (**Offizielle Behebung**): Es gibt eine komplette Behebung der Schwachstelle durch den Hersteller. o **T** Temporary Fix (**Temporäre Behebung**): Es gibt eine Behebung der Schwachstelle durch den Hersteller, diese ist allerdings nur temporär. o **W** Workaround (**Workaround**): Es gibt eine inoffizielle Möglichkeit, die Schwachstelle zu beheben, dies kann aber zu Beeinträchtigungen führen. o **U** Unavailable (**Nicht verfügbar**): Es ist keine Maßnahme zur Behebung der Schwachstelle.

Wert	Erklärung und mögliche Werte
RC – Report Confidence **Plausibilität des Berichtes**	Dieser Wert gibt an, wie gesichert die Existenz der Schwachstelle ist und wie plausibel die technischen Details dazu sind. Mögliche Werte sind: o **X** Not Defined (**Nicht definiert**): Keine Angabe und daher keine Veränderung des Base Score. o **U** Unknown (**Nicht bekannt**): Es sind wenige technische Details bekannt und die Reproduzierbarkeit ist nicht gegeben. o **R** Reasonable (**Akzeptabel**): Es sind einige, aber nicht alle technischen Details bekannt und die Reproduzierbarkeit ist nur teilweise gegeben. o **C** Confirmed (**Bestätigt**): Technische Details existieren und eine Reproduzierbarkeit ist gegeben.