



KALWEIT ITS



Penetrationstest



KALWEIT ITS GmbH
im Auftrag der

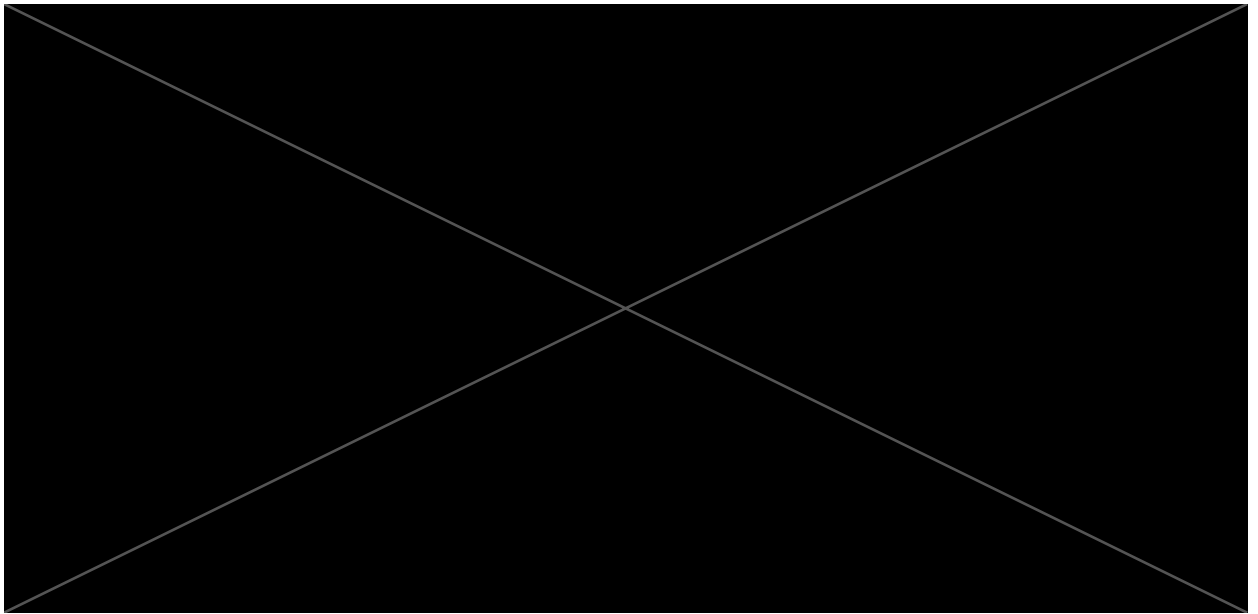


Finale Fassung – **vertraulich**

 2022 –  2022

Inhaltsverzeichnis

Vorwort	4
Einleitung	5
Management Summary	6
Projektübersicht	8
Überprüfte Applikationen	10
Risikobewertung	11
Ansatz des Penetrationstests	13
Reconnaissance	15
OSINT	15
Eingesetzte Standards und Methoden	19
Penetrationstest Zeitplan	19
Eingesetzte Sicherheitswerkzeuge	19
Detaillierte Ergebnisse des Penetrationstests	20
Penetrationstest Schwachstellen	20
Anhang	65
Verantwortlichkeiten	66
Über uns - Kalweit ITS	66



Vorwort

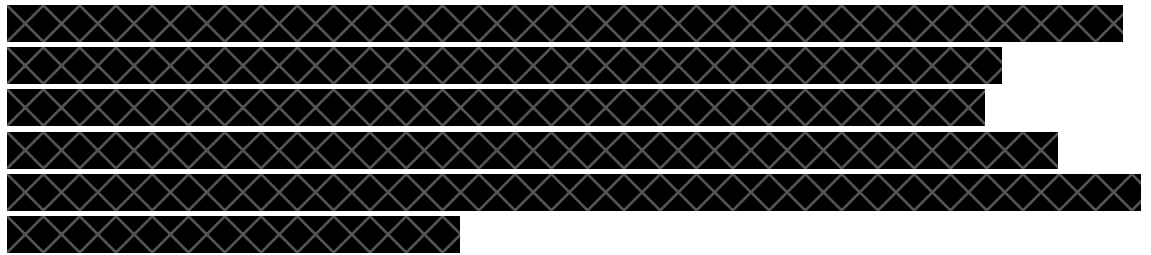
Die KALWEIT ITS GmbH vereint eine Handvoll leidenschaftlicher, neugieriger, junger Experten, die ein Ziel haben: Immer neue Wege zu finden, um in Systeme einzudringen. Mutige Ideen, pragmatisch umgesetzt, um immer wieder neue Wege zu finden, den Status Quo der IT-Sicherheit unserer Kunden auf das nächste Level zu bringen. Partnerschaftliche Treue, individuelle Herangehensweisen, Zusammenarbeit auf Augenhöhe. Das ist es, wonach wir tagtäglich Tag streben und worin wir stetig besser werden möchten.

Die in diesem Bericht dokumentierten Audittätigkeiten wurden liebevoll und mit bestmöglicher Präzision durchgeführt. Ein kleines Team von Experten hat diesen Report mit größtmöglicher Sorgfalt und den besten Absichten erstellt. Mehrfach wurde sie von mir persönlich und von meinem Team auf Fehler überprüft. Sollten Sie dennoch Änderungswünsche haben oder Korrekturen notwendig sein, zögern Sie bitte nicht, mich persönlich zu kontaktieren.



Philipp Kalweit, Geschäftsführer

Einleitung



Aufgrund dieser hohen Verantwortung gegenüber Kunden, Mitarbeitenden und Geschäftspartnern hat sich das Unternehmen dazu entschlossen eine IT-Sicherheitsüberprüfung aller aus dem Internet erreichbaren IT-Dienste durch unabhängige IT-Sicherheitsprüfer durchzuführen.

Management Summary

██ beauftragte die KALWEIT ITS GmbH, ein unabhängiges Fachberatungsunternehmen mit Schwerpunkt in Penetrationstests, mit der Durchführung eines IT-Sicherheitsaudits. Gegenstand des Tests waren alle dem Unternehmen zugehörigen IT-Systeme und Dienste, welche aus dem Internet erreichbar waren.

Der externe Test umfasste automatisierte Scans sowie eine manuelle Sicherheitsprüfung nach OWASP, PTES, sowie den Empfehlungen durch das Bundesamt für Sicherheit in der Informationstechnik (BSI), wie sie im Rahmen der Studie „Durchführungskonzept für Penetrationstests“ formuliert wurden. Die Einstufung des Penetrationstests lautet wie folgt:

Penetrationstest; abwägend (Agressivität), fokussiert (Umfang); offensichtlich (Vorgehensweise), von außen (Ausgangspunkt).

Die Projektergebnisse deuten auf eine akzeptable IT-Sicherheitslage hin. Ein Großteil der Maßnahmen zur IT-Sicherheit entspricht dem Stand der Technik sowie bekannten Best Practices in der IT-Sicherheit. Es lässt sich mutmaßen, dass das Unternehmen zum gegenwärtigen Zeitpunkt keine ganzheitliche IT-Sicherheitsstrategie (wie bspw. VdS10000) verfolgt,

Im Testzeitraum wurden drei Schwachstellen mit hohem Schweregrad gefunden, von denen keine als kritisch eingestuft wurde. Andere Schwachstellen, die identifiziert wurden, sind als mittel, gering oder rein informativ zu bewerten.

Es wird darauf hingewiesen, dass ein Verbesserungspotenzial besteht und sowohl technische als auch regulatorische Maßnahmen ständig auf Wirksamkeit und Aktualität überprüft werden sollten. Die Beurteilung der IT-Sicherheitslage bezieht sich ausschließlich auf den aktuellen Zustand der IT-Infrastruktur während des Testzeitraums. Kleine Veränderungen in informationstechnischen Prozessen können jedoch erhebliche Auswirkungen auf die allgemeine IT-Sicherheitslage haben.

Alle Daten wurden vertraulich behandelt und zu jederzeit auf europäischen Servern gespeichert.

Es wird empfohlen insbesondere folgende Handlungsempfehlungen zeitnah umzusetzen:

[CWE-1104]: Veraltete PHP-Version

Handlungsempfehlung I

Die PHP-Version 7.2.23 sollten schnellstmöglich auf den neuesten Stand aktualisiert werden. Durch die veraltete Version können Angreifer eine Reihe verschiedenener

Exploits anwenden, wodurch die Vertraulichkeit, Integrität und Verfügbarkeit von Daten und Diensten gefährdet wird.

[[CVE-2018-15473](#)]: User Enumeration Through Vulnerable OpenSSH Version

Handlungsempfehlung II

Einem potenziellen Angreifer wäre es möglich Nutzer und Dienste auf dem System zu enumerieren. Das Auflisten von Diensten stellt eine sehr hohe Gefahr dar, da diese Rückschlüsse über die Funktion sowie die auf dem Server implementierten Sicherheitsmaßnahmen zulassen.

Es wird empfohlen, die Softwarekomponenten nach Möglichkeit auf die neuesten Versionsstufen zu aktualisieren.

[[CVE-2014-8985](#)]: Security Bypass Vulnerability

Handlungsempfehlung III

Das IP-Sicherheitsfeature im Microsoft Internet Information Services (IIS) 8.0 und 8.5 verarbeitet bestimmte Regeln für Domänen nicht ordnungsgemäß.

Es wird empfohlen, alle Softwarekomponenten nach Möglichkeit auf die neuesten Versionsstufen zu aktualisieren. Somit kann die Schwachstelle behoben werden. Es sollte zudem, nach Möglichkeit, eine Patch-Management Strategie eingeführt werden.

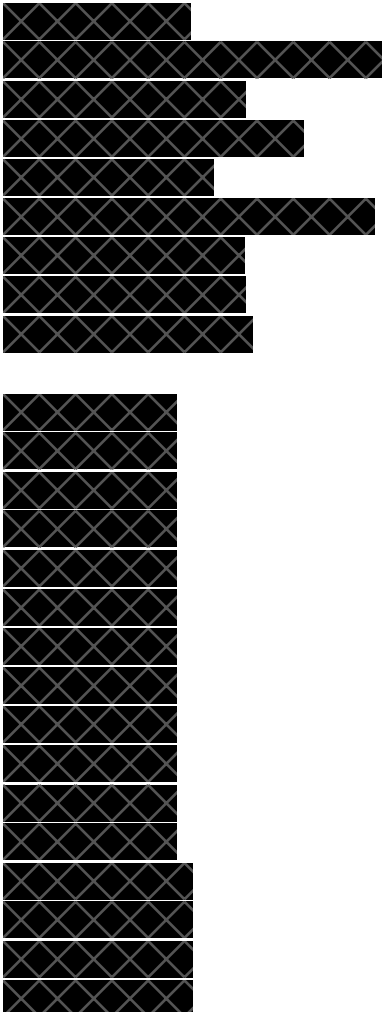

Projektübersicht


Version	Name	Datum
☒	██████████	██████████
☒	████████████████████	██████████
☒	████████████████████ ██████████████████	██████████
☒	████████████████████	██████████
☒	████████████████████ ██████████████	██████████

Projekt	[REDACTED]
Projektdauer	7 Projektstage (8 Stunden pro Projekttag)
Projektaktivitäten	Penetrationstest
Involvierte Consultants	[REDACTED]
Projektstandort	Remote
Kontaktpersonen auf Kundenseite	[REDACTED]
Projektbezogene Versicherung	Vermögensschadenhaftpflicht mit je 10.000.000,00 Euro Personen- und Sachschäden bei 5.000.000,00 Euro

Überprüfte Applikationen

Im Rahmen unseres Auftrages haben wir die folgenden Systeme und Anwendungen auf ihre Sicherheit hin überprüft:

Ziel- URLs	
Auditzeitraum	

Alle im Angebot  genannten Aspekte wurden mit der Durchführung der Sicherheitsprüfung abgegolten.

Wenn Sie weitere Fragen haben, zögern Sie nicht, uns zu kontaktieren. Wir helfen Ihnen gerne bei der Interpretation der Ergebnisse und entwickeln gemeinsam mit Ihnen sichere Lösungen für Ihre Anwendungen.

Risikobewertung

Schwachstelle	Schweregrad
[CWE-1104]: Veraltete PHP-Bibliothek	Hoch
[CVE-2021-24719]: WordPress Theme is Vulnerable	Hoch
[CVE-2018-15473]: User Enumeration Through Vulnerable OpenSSH Version	Hoch

Schwachstelle	Schweregrad
[CWE-307]: Improper Restriction of Excessive Authentication Attempts	Mittel
[CVE-2014-8985]: Security Bypass Vulnerability	Mittel
[CVE-2020-9490]: Potential Denial-of-Service	Mittel
[CWE-1104]: Use of Unmaintained Third-Party Components (Vulnerable JavaScript Dependency)	Mittel
[CWE-434]: Unrestricted Upload of File with Dangerous Type	Mittel
[CWE-200]: Ungeschützte SQL-Datenbank	Mittel

Schwachstelle	Schweregrad
[CWE-552]: Directory/File/API - Enumeration	Gering
[CWE-310]: Base64-encoded Data in Parameter	Gering
[CWE-610]: Externally Controlled Reference to a Resource in Another Sphere (Redirection from HTTP to HTTPS)	Gering

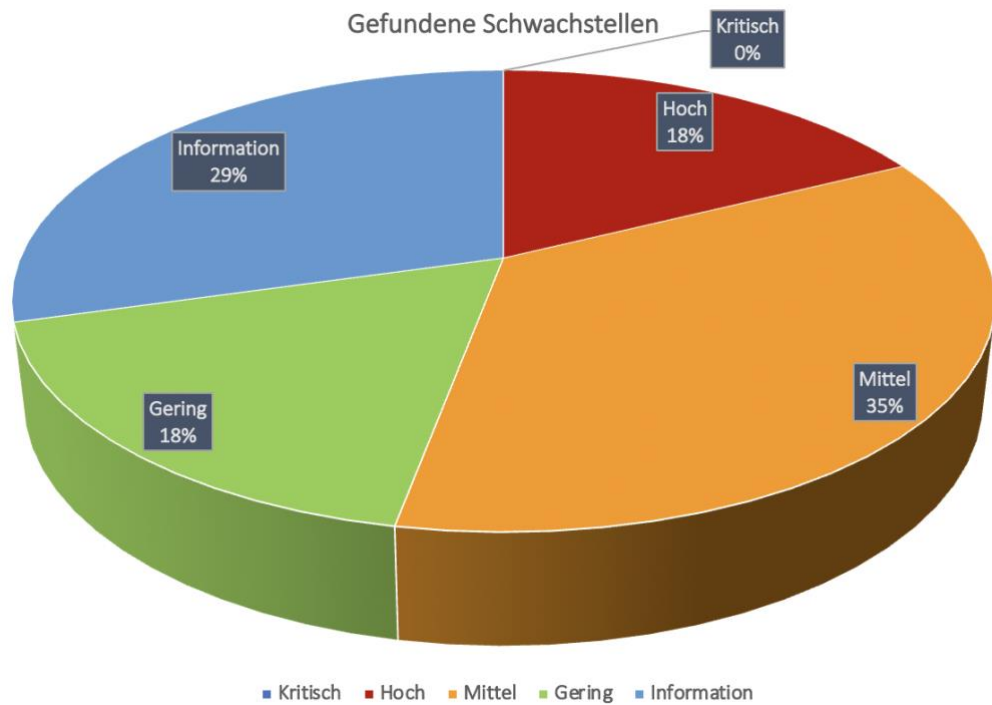
Schwachstelle	Schweregrad
[A01: 2021]: Broken Access Control	Information
[CWE-540]: Inclusion of Sensitive Information in Source Code	Information
[CWE-693]: Protection Mechanism Failure (Strict Transport Security Misconfiguration)	Information

[CWE-79]: Improper Neutralization of Input During Web Page Generation (Browser Cross-Site Scripting Filter Misconfiguration)	Information
[CWE-693]: Protection Mechanism Failure (Content Sniffing Not Disabled)	Information

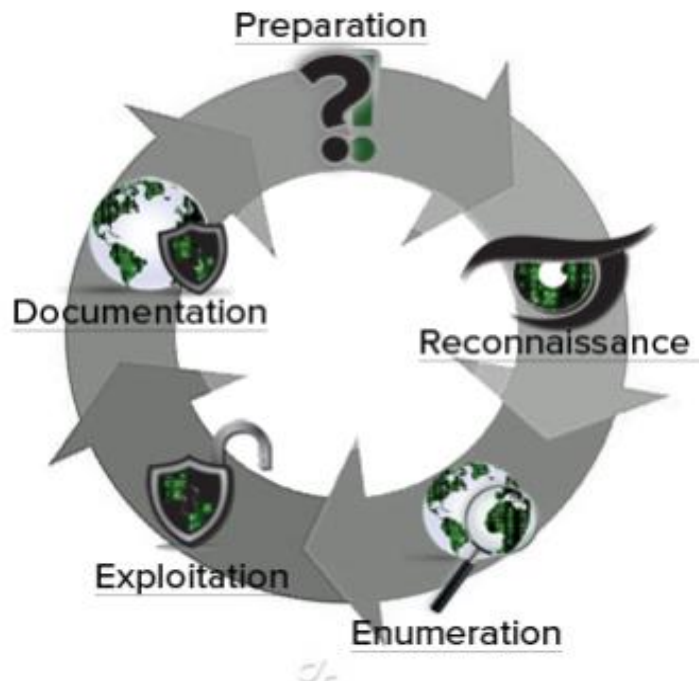
Die durchgeführten Sicherheitsprüfungen zeigen die folgenden Ergebnisse:

Risiko-Level	Kritisch	Hoch	Mittel	Gering	Information
Anzahl	0	3	6	3	5

- Information:** Dieser Befund weist nicht auf eine Schwachstelle hin, sondern enthält einen Kommentar, der auf Konstruktionsfehler und / oder auf eine unsachgemäße Implementierung hinweist.
- Gering:** Sollte behoben werden, hat aber keine Priorität.
- Mittel:** Sollte zeitnah behoben werden.
- Hoch:** Sollte mit großer Dringlichkeit behoben werden.
- Kritisch:** Sollte sofort behoben werden.



Ansatz des Penetrationstests



Phase 1: Preparation

Bevor die eigentlichen Tests und Scans beginnen können, müssen zunächst die relevanten rechtlichen Bedingungen und der genaue Umfang der Prüfung in Phase 1 (Preparation) festgelegt werden. Sobald alle organisatorischen Bedingungen ausgehandelt sind, kann die Phase 2 (Reconnaissance) beginnen.

Phase 2: Reconnaissance

In diesem Abschnitt des Penetrationstests wird das Testobjekt durch erste Analysen wie Portscans und Footprinting analysiert. Darüber hinaus werden in dieser Phase automatisierte Scan-Tools eingesetzt, um so viele Informationen wie möglich über das Testobjekt zu sammeln und erste automatisierte Angriffe auf die Systeme durchzuführen. Basierend auf den Ergebnissen der eingesetzten Tools können die ersten potenziellen Schwachstellen identifiziert und dann von den Tools aktiv ausgenutzt werden.

Phase 3: Enumeration

Die entsprechenden Scan-Ergebnisse werden ausgewertet und analysiert. Einerseits werden die bereits von den Tools ausgeführten Angriffe sichtbar und andererseits

können die gesammelten Informationen der Scans zur Ausführung manueller Angriffe in Phase 4 (Exploitation) verwendet werden.

Phase 4: Exploitation

Basierend auf den Scanergebnissen aus Phase 2 und der detaillierteren Analyse in Phase 3 (Enumeration) werden manuelle Angriffe auf das Testobjekt durchgeführt. In dieser Phase wird versucht, Schwachstellen manuell auszunutzen, z. B., um Angriffe auf die Anmeldemaske durchzuführen. Darüber hinaus werden weitere Angriffe wie XSS oder SQL-Injection durchgeführt. Phase 4 (Exploitation) nimmt einen großen Teil der verfügbaren Projektzeit in Anspruch und stellt damit den wichtigsten Projektmeilenstein dar.

Phase 5: Documentation

Schließlich werden die festgestellten Schwachstellen und Befunde ausgewertet und dokumentiert. Die Nutzungswahrscheinlichkeiten sowie der Schweregrad der Schwachstelle und andere relevante Parameter werden berücksichtigt, um eine angemessene Risikobewertung zu ermöglichen.

Reconnaissance

OSINT

Open-Source-Intelligence (OSINT) ist eine Form der nachrichtendienstlichen Informationsbeschaffung, bei der Informationen aus öffentlich zugänglichen Quellen gefunden, ausgewählt und analysiert werden, um verwertbare Erkenntnisse zu gewinnen, ohne dabei die gesetzlichen Bestimmungen zu verletzen. Für diese Methodik werden ausschließlich passive Werkzeuge eingesetzt, daher stellen die nachfolgenden Befunde reine Beobachtungen dar. Eine tatsächliche Überprüfung der Richtigkeit durch ein Proof of Concept sind nicht Bestandteil von OSINT. Im späteren Verlauf des Penetration Testes können die gefundenen Informationen jedoch genutzt werden.

BEOBACHTUNGEN

Für [REDACTED] wurden **16** Subdomains gefunden:

[REDACTED]

Für [REDACTED] wurden **11** offene Ports gefunden:

21/tcp	open	ftp
22/tcp	open	ssh
80/tcp	open	http
110/tcp	open	pop3
143/tcp	open	imap
443/tcp	open	https
465/tcp	open	smtps

Cloud Provider

[REDACTED]

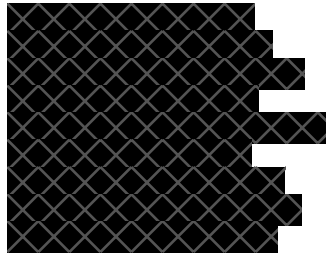
Cloud Asset

[REDACTED]

Gefundene Mailadressen

Nachfolgende Email-Adressen konnten in Datenlecks und dem Darknet gefunden werden. Es wird empfohlen die Existenz/Aktualität dieser Email-Adressen zu prüfen und Mitarbeitende der Organisation darüber aufzuklären, dass Firmen-Email-Adressen nicht für private Zwecke genutzt werden sollten. **Ferner gilt es diese Informationen auf Richtigkeit zu prüfen.**

[REDACTED]



Eingesetzte Standards und Methoden

Die Methodik zur Bewertung der Anwendungssicherheit basiert auf den folgenden Richtlinien und Standards:

[OWASP Testing Guide](#) - ein Industriestandard für Webanwendungen und verwandte Technologien. Wird bei Sicherheitstests von Webanwendungen verwendet.

[PTES](#) - eine innovative Methode für Penetrationstests, die von weltweit führenden Experten für Penetrationstests, Sicherheitstests und Social Engineering entwickelt wurde.

[OWASP Top 10](#) – Teststandard für die am häufigsten auftretenden und ausgenutzten Sicherheitslücken

Penetrationstest Zeitplan

Phase	[Redacted Content]						
Preparation							
Reconnaissance							
Enumeration							
Exploitation							

Eingesetzte Sicherheitswerkzeuge

- Burp Suite Pro [Commercial Edition]
- Kali Linux Toolkit

Detallierte Ergebnisse des Penetrationtests

Penetrationtest Schwachstellen

[CWE-1104]: Veraltete PHP-Version

Schweregrad der Schwachstelle: Hoch

Host:	Path:
[REDACTED]	

INHÄRENTE RISIKEN:

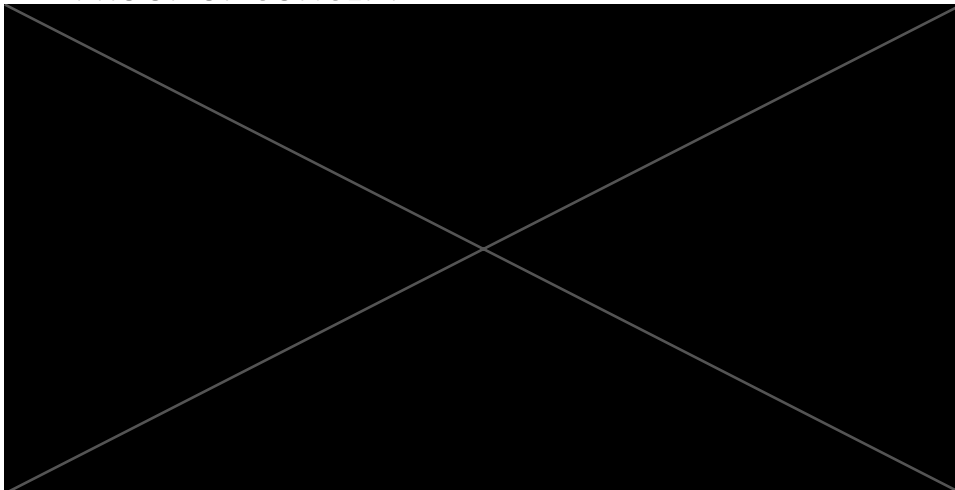
PHP-Versionen besitzen eine Lebensdauer von ca. zwei Jahren. In dieser Zeit werden weiterhin aktiv Sicherheitsupdates bereitgestellt.

Werden nach Ablauf dieser zwei Jahre neue Schwachstellen bekannt, bleiben diese in der Regel bestehen und stellen so ein hohes Risiko für die betroffenen Anwendungen dar. Der Support für PHP 7.2.23 endete am **30.11.2020**.

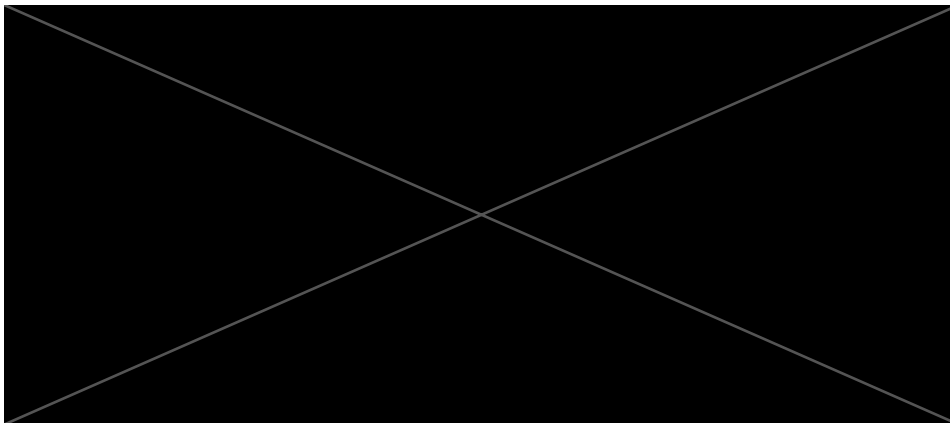
Veraltete Softwareversionen bieten Angreifern ein gutes Angriffsziel. Für die PHP-Version 7.2.23 sind eine große Zahl von Exploits und Angriffsmöglichkeiten bekannt, so etwa:

- CVE-2020-7067
- CVE-2019-11044
- CVE-2019-11047
- CVE-2019-11050

PROOF OF CONCEPT



equest



esponse

HANDLUNGSEMPFEHLUNG:

Es sollte eine Patchstrategie implementiert werden, um alle Drittanbieter-Anwendungen und -Bibliotheken stets auf dem aktuellsten Versionsstand zu halten. Durch regelmäßige Updates kann das Risiko von Angriffen auf veraltete Versionen minimiert werden.

REFERENZEN

- [CWE-1104: Use of Unmaintained Third-Party Components](#)
- [Informationen zu PHP-Versionen](#)

[CVE-2021-24719]: WordPress Theme is Vulnerable

Schweregrad der Schwachstelle: Hoch

Host:
██

INHÄRENTE RISIKEN:

Das Enfold WordPress-Theme vor 4.8.4 war anfällig für Reflected Cross-Site Scripting (XSS). Die Schwachstelle ist in den Enfold-Versionen vor 4.8.4 vorhanden, welche den Avia Page Builder verwenden. Durch XSS kann in eine URL Schadcode eingefügt werden, welche der Server aufgrund mangelnder Absicherung übernimmt und eine dynamisch generierte Website erstellt. Diese präparierte URL kann einem Opfer zugesendet werden, welches die Seite für vertrauenswürdig hält. Wenn das Opfer nun Daten auf der präparierten Website angibt, so können diese an den Angreifer weitergeleitet werden.

PROOF OF CONCEPT

- Enfold Theme < 4.8.4 = Reflected Cross-Site Scripting (XSS)
- Fixed in: 4.8.4

It is possible for an attacker to add some input text in the URL that is reflected on the response. Therefore, there is a XSS vulnerability that can be exploited crafting a customized URL which includes a link to a paginated entry:

`https://[website]/paginated-entry/?ProofOfConcept`

"ProofOfConcept" text will be reflected on the server's response. It will be included on generated page buttons.

The attacker can exploit this issue by sending the following payload:

```
?%2527%253E%253Cscript%253Eeval%2528atob%2528%2522Y29uc29sZS5sb2coZG9jdW1lbnQuY29va2llKQ%253D%253D%2522%2529%2529%253C%252Fscript%253E
```

which can be decoded as:

```
'><script>eval(atob("Y29uc29sZS5sb2coZG9jdW1lbnQuY29va2llKQ=="))</script>
```

and the Base64 payload decoded as:

```
console.log(document.cookie);
```

HANDLUNGSEMPFEHLUNG:

Wordpress, sowie auch die darauf installierten Plugins und Themes sollten regelmäßigen Updates unterzogen werden. Nach Möglichkeit sollten Updates immer zyklisch eingespielt werden, sodass keine veralteten Versionen entstehen können und alle Softwareversionen aktuell sind. Mit aktuellen Versionen sinkt beständig die Gefahr Opfer eines Angriffs zu werden.

REFERENZEN

- [CVE -CVE-2021-24719](#)

[CVE-2018-15473]: User Enumeration Through Vulnerable OpenSSH Version

Schweregrad der Schwachstelle: Hoch

INHÄRENTE RISIKEN:

Die Schwachstelle CVE-2018-15473 existiert bereits seit 2018 und stellt damit ein Risiko für die Sicherheit dar. Durch die mehrjährige Existenz einer Sicherheitslücke, können Angreifer auf ein großes Repertoire an Exploits zurückgreifen und besitzen somit verschiedene Möglichkeiten die Schwachstelle auszunutzen.

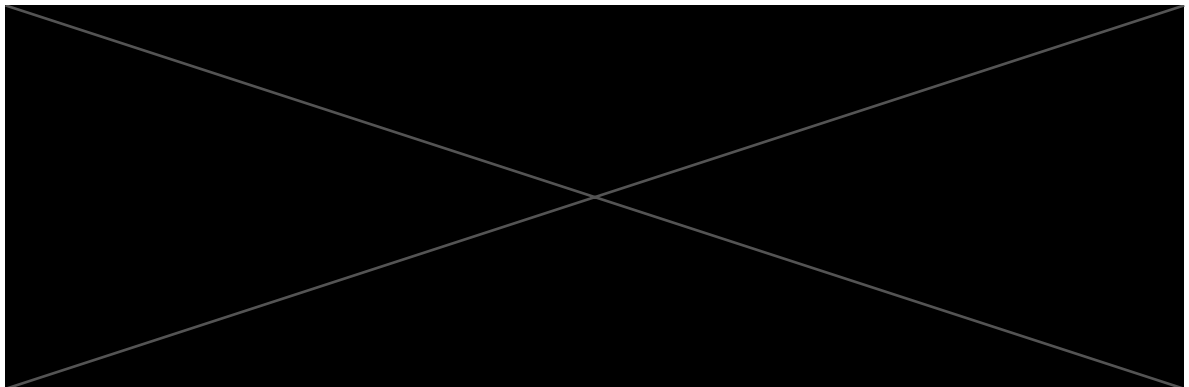
Bei dieser Schwachstelle ist es möglich Nutzer und Dienste auf dem System zu enumerieren. Diese Informationen können u.a. zur Vorbereitung von Bruteforce-Angriffen verwendet werden.

Das Auflisten von Diensten stellt eine sehr hohe Gefahr dar, da diese Rückschlüsse über die Funktion sowie die auf dem Server implementierten Sicherheitsmaßnahmen zulassen.

Eine Auflistung der Resultate der Enumeration befindet sich im Anhang in der Datei *<OpenSSH-Enumeration.pdf>*

PROOF OF CONCEPT

- Öffnen sie ein Terminal auf einem Linuxsystem und erstellen sie mit dem folgenden Befehl eine neue Datei:



- Kopieren Sie den Inhalt aus *<SSH_Enum_Exploit.txt>* (siehe Anhang) per GUI in Ihre Zwischenablage
- Drücken Sie „i“ zum Aktivieren des Schreibmodus
- Drücken Sie Ctrl+Shift+V zum Einfügen aus der Zwischenablage


```

import sys
import re
import socket
import logging
import argparse
import multiprocessing
from typing import Union
from pathlib import Path

import paramiko

assert sys.version_info >= (3, 6), "This program requires python3.6 or higher"

class Color:
    """ Class for coloring print statements. Nothing to see here, move along. """
    BOLD = '\033[1m'
    ENDC = '\033[0m'
    RED = '\033[38;5;196m'
    BLUE = '\033[38;5;75m'
    GREEN = '\033[38;5;149m'
    YELLOW = '\033[38;5;190m'

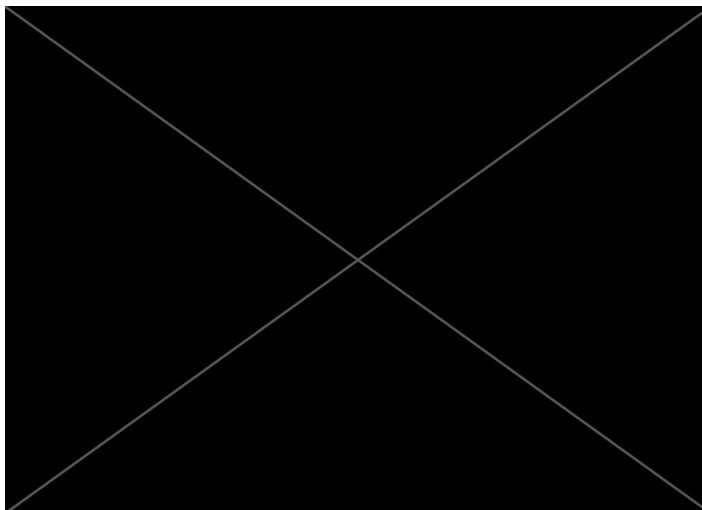
    @staticmethod
-- INSERT --

```

19,1

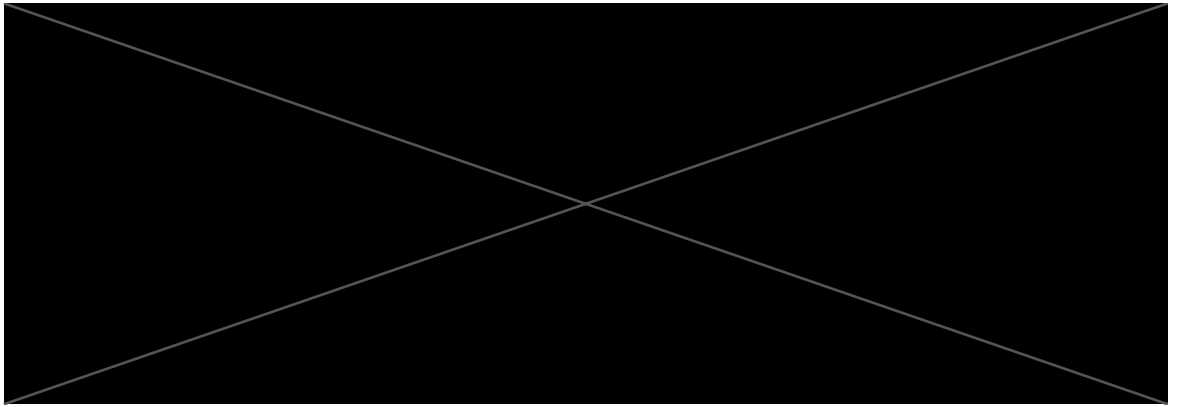
Top

- Drücken Sie ESC zum Verlassen des Schreibmodus
- Mittels <:x>- <Enter> speichern und beenden Sie ohne Rückfrage
- Wiederholen Sie diesen Schritt für eine .txt Datei, in welcher Sie eigene Nutzer/Dienste zum Testen schreiben können. Alternativ benutzen Sie gerne die von uns zur Verfügung gestellte *unix_users.txt*-Datei
- Sie sollten nun zwei Dateien in Ihrem Verzeichnis sehen



- Zurück im Terminal können Sie Ihr eben erstelltes Python-Script nun wie folgt ausführbar machen:

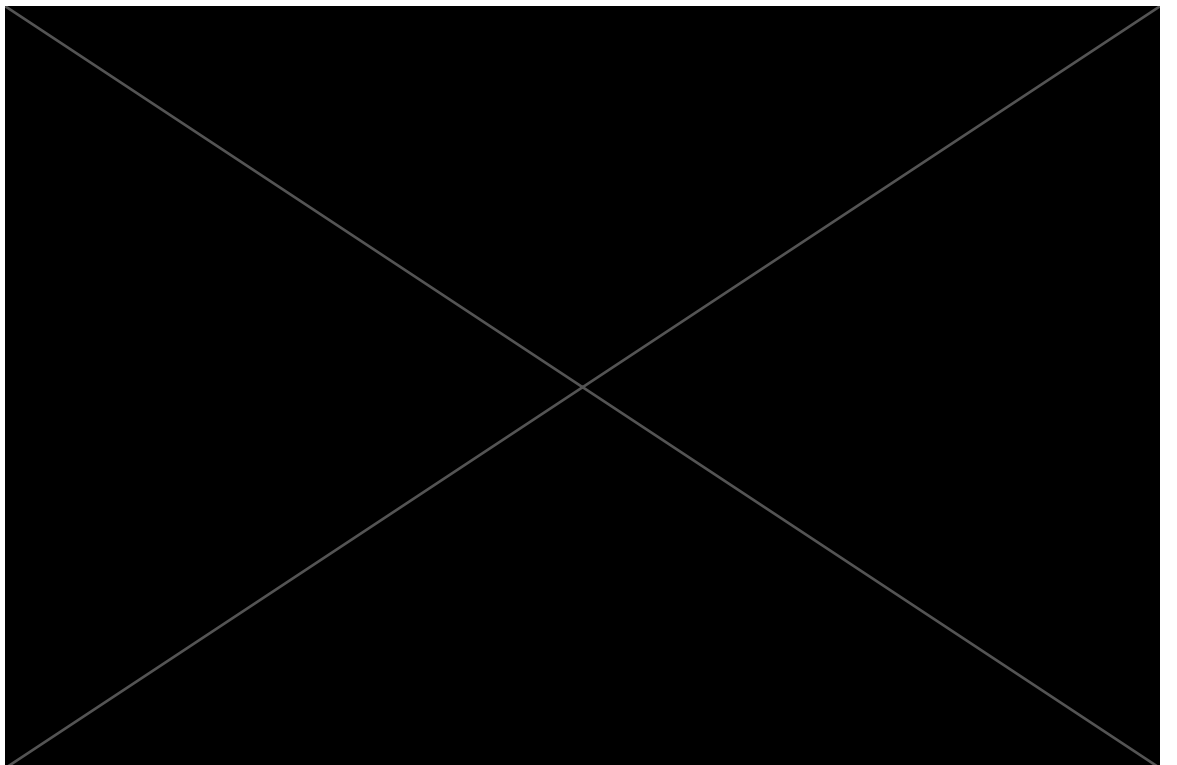
```
$ chmod +x exploit.py
```



- Der Exploit kann nun durch den folgenden Befehl gestartet werden:

```
$ python3 exploit.py -t 10 -w unix_users.txt
```

- -t Anzahl der Threads
- -w Pfad zur Wordlist (z.B. unix_users.txt)



Anhang mit Results

HANDLUNGSEMPFEHLUNG:

Es wird empfohlen, die Softwarekomponenten nach Möglichkeit auf die neuesten Versionsstufen zu aktualisieren. Beachten Sie auch, dass wir die Implementierung einer Patchverwaltungsstrategie empfehlen, um sicherzustellen, dass Sicherheitsupdates umgehend auf alle Bibliotheken von Drittanbietern in Ihrer Anwendung angewendet werden. Erwägen Sie auch, Ihre Angriffsfläche zu reduzieren, indem Sie Bibliotheken entfernen, die nicht mehr verwendet werden.

Wenn eine Patch-Management-Strategie vorhanden ist, empfehlen wir zu überprüfen, ob sie effektiv und auf dem neuesten Stand ist.

REFERENZEN

- [CVE-2018-15473](#)
- [Vulnerability Details: CVE-2018-15473](#)

[CWE-307]: Improper Restriction of Excessive Authentication Attempts

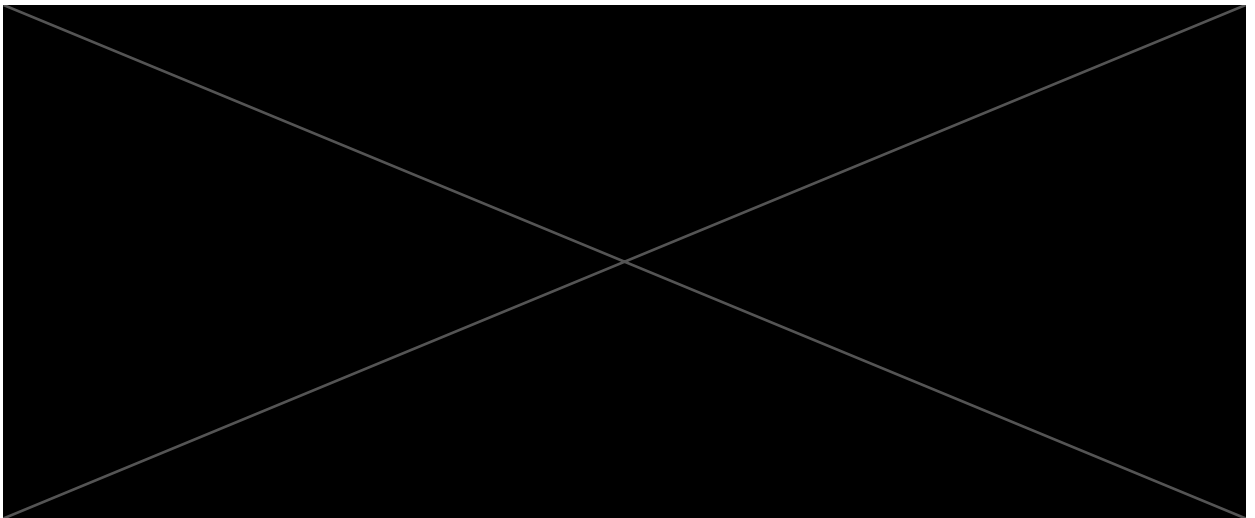
Schweregrad der Schwachstelle: **Mittel**

Host:	Pfad:
	

INHÄRENTE RISIKEN:

Wenn Login-Versuche nicht limitiert werden, können Angreifer sogenannte Brute-force Angriffe durchführen. Hierbei werden mit Standardpasswörtern, geleakten Zugangsdaten oder Wörterbüchern, so lange Enumerationsversuche durchgeführt, bis diese zum Erfolg führen. Auch wird hierdurch die Möglichkeit geschaffen, einen Denial-of-Service-Angriff durchzuführen. Dabei wird eine so enorm hohe Anzahl an Anfragen gestellt, bis die Verfügbarkeit der Anwendung eingeschränkt wird.

PROOF OF CONCEPT



HANDLUNGSEMPFEHLUNG:

Es wird empfohlen, gemäß der Best Practices der IT-Sicherheit eine Wartezeit bis zum nächsten Anmeldeversuch zu implementieren. Weiterhin sollte nach zehn oder mehr Anmeldeversuchen die temporäre Sperrung der IP erfolgen. Weiterhin sollte der Nutzeraccount eine Meldung über die Anmeldeversuche erhalten.

REFERENZEN

- [CWE-307: Improper Restriction of Excessive Authentication Attempts](#)
- [Blocking Brute Force Attacks](#)

[CVE-2014-8985]: Security Bypass Vulnerability

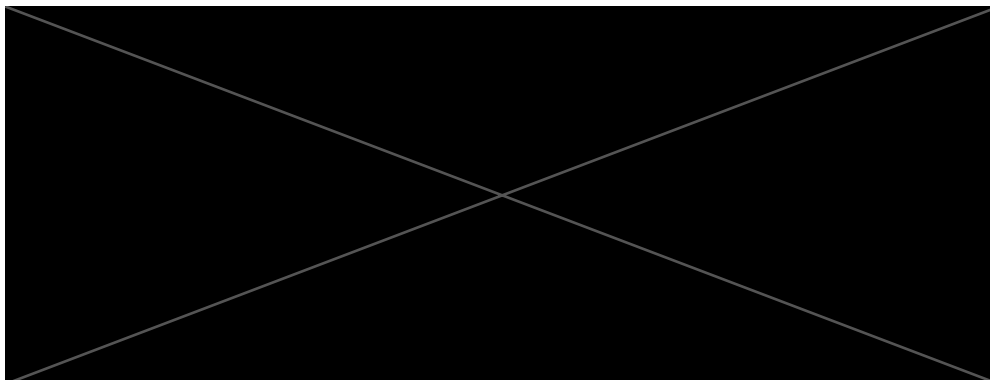
Schweregrad der Schwachstelle: **Mittel**

Host:	Pfad:
	

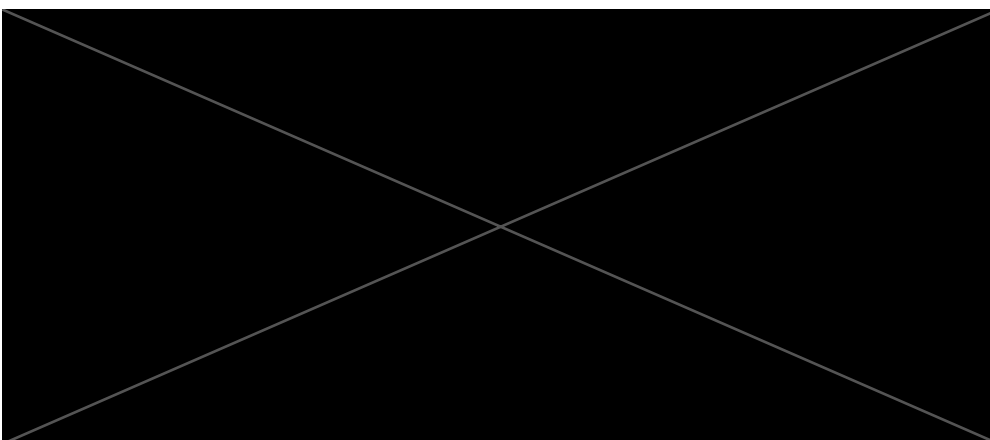
INHÄRENTE RISIKEN:

Das IP-Sicherheitsfeature im Microsoft Internet Information Services (IIS) 8.0 und 8.5 verarbeitet Wildcards und Allow/Deny-Regeln für Domänen nicht ordnungsgemäß. Dadurch können Angreifer recht einfach über einen http-Request den Regelsatz umgehen. Durch diese Schwachstelle könnten weitere Angriffe ermöglicht oder erleichtert werden. Da es sich hierbei um ein Herstellerseitiges Sicherheitsfeature handelt, sollte seine einwandfreie Funktionalität und die aller weiteren Komponenten, stets gewährleistet sein.

PROOF OF CONCEPT



Request



Response

HANDLUNGSEMPFEHLUNG:

Es wird empfohlen, alle Softwarekomponenten nach Möglichkeit auf die neuesten Versionsstufen zu aktualisieren. Somit kann die Schwachstelle behoben werden. Nach Möglichkeit sollten Updates immer zyklisch eingespielt werden, sodass keine veralteten Versionen entstehen können und alle Softwareversionen aktuell sind.

REFERENZEN

- [CVE-2014-8985 Detail](#)

[CVE-2020-9490]: Potential Denial-of-Service

Schweregrad der Schwachstelle: Mittel

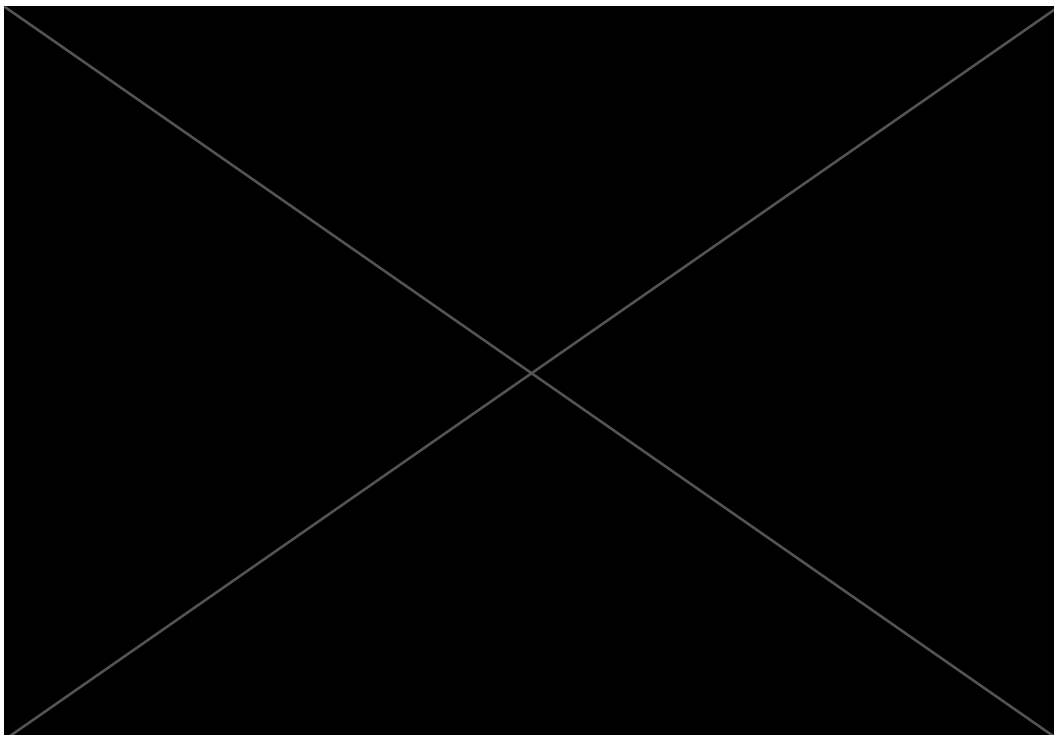
INHÄRENTE RISIKEN:

Apache Server mit der Version 2.4.38 sind von einem Fehler betroffen, welcher sowohl versehentlich als auch durch einen gezielten Angriff zum Absturz des Servers führen kann.

Dieser Fehler ist am 07.08.2020 durch Sicherheitsforscher veröffentlicht worden und ist gut dokumentiert. Es muss also davon ausgegangen werden, dass diese Schwachstelle potenziellen Angreifern bekannt ist. Das Ausnutzen dieser Sicherheitslücke, würde es einem Angreifer ermöglichen einen Denial-of-Service-Angriff gegen den betroffenen Server durchzuführen und somit die Verfügbarkeit der auf dem Server installierten Dienste erheblich einzuschränken.

PROOF OF CONCEPT

Host:

Response Header mit Serverinformationen

HANDLUNGSEMPFEHLUNG:

Die Sicherheitslücke kann behoben werden, indem „H2PushOff“ durch die http/2 Funktion konfiguriert wird. Des Weiteren sollte Apache auf die neueste Version

geupdated werden. Mit einem Update lässt sich diese Lücke schließen. Im Zuge dessen sollte eine Patchstrategie eingepflegt werden, um zyklisch immer die neuesten Versionen und Updates zu implementieren. Somit sinkt die Gefahr eines Angriffs, durch die Ausnutzung veralteter Versionen.

REFERENZEN

- [CVE-2020-9490](#)
- [Push Diary Crash on Specifically Crafted HTTP/2 Header \(CVE-2020-9490\)](#)

[CWE-1104]: Use of Unmaintained Third-Party Components (Vulnerable JavaScript Dependency)

Schweregrad der Schwachstelle: **Mittel**

INHÄRENTE RISIKEN:

Die Verwendung von JavaScript-Bibliotheken von Drittanbietern kann zu einer Reihe von DOM-basierten Schwachstellen führen, einschließlich einiger, die verwendet werden können, um Benutzerkonten via DOM-XSS zu hijacken.

Gängige JavaScript-Bibliotheken genießen in der Regel den Vorteil, dass sie stark durch Sicherheitsforscher überwacht werden. Dadurch können Fehler schnell identifiziert und gepatcht werden, was zu einem stetigen Strom von Sicherheitsupdates führt, die angewendet werden müssen. Diese Updates sollten nicht ignoriert werden, da die Verwendung einer veralteten Bibliothek schnell zu Schwachstellen führen kann. Daher ist es wichtig sicherzustellen, dass alle verfügbaren Sicherheitsupdates umgehend installiert werden.

Einige Bibliotheksschwachstellen können so jede Anwendung für Angriffe öffnen, die die Bibliothek importiert. Andere wiederum betreffen jedoch nur Anwendungen, die bestimmte Bibliotheksfeatures verwenden. Es kann schwierig sein, genau zu identifizieren, welche Bibliotheksschwachstellen auf Ihre Website zutreffen, **daher empfehlen wir, alle verfügbaren Sicherheitsupdates unabhängig davon anzuwenden.**

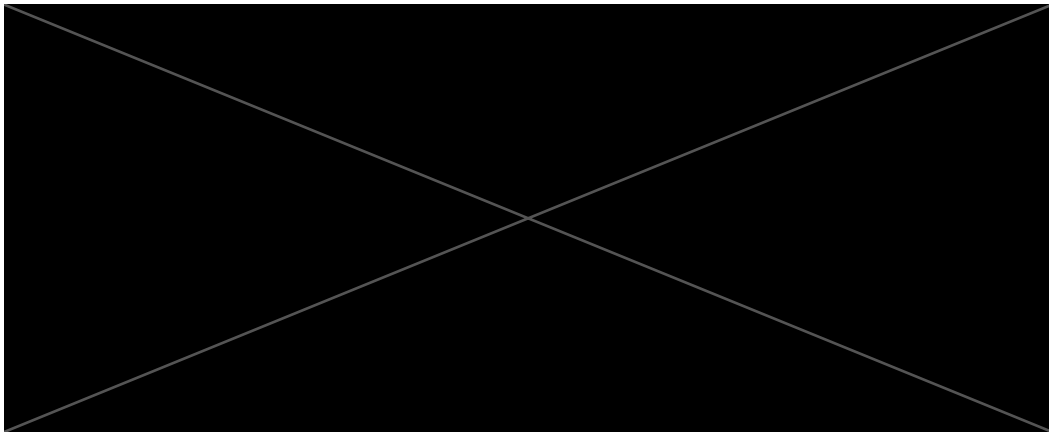
Im Zuge des Penetrationstests, wurde die folgende veraltete JavaScript-Bibliothek gefunden, welche hier mit ihren bekannten Schwachstellen aufgeführt wird:

jQuery Version 3.3.1:

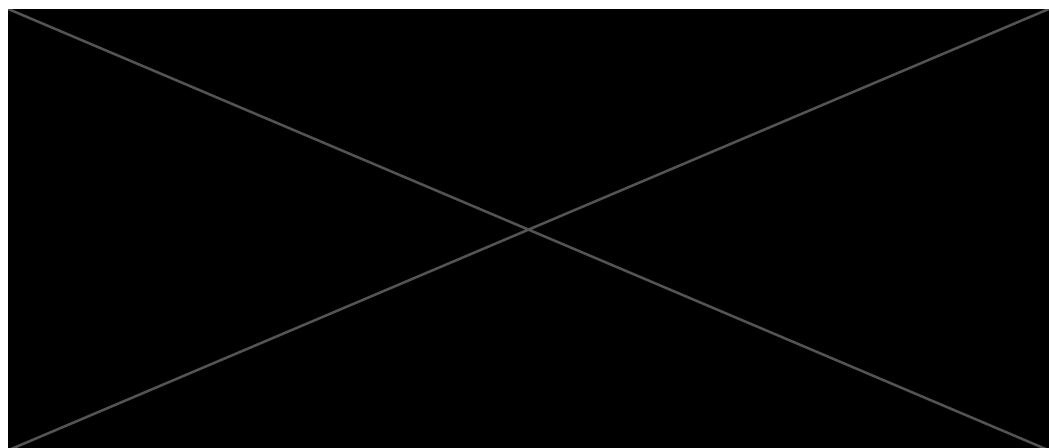
- [CVE-2019-11358]: jQuery before 3.4.0, as used in Drupal, Backdrop CMS, and other products, mishandles jQuery.extend(true, {}, ...) because of Object.prototype pollution
- [CVE-2020-11022]: Regex in its jQuery.htmlPrefilter sometimes may introduce XSS
- [CVE-2020-11023]: Regex in its jQuery.htmlPrefilter sometimes may introduce XSS

PROOF OF CONCEPT

	Host:	Path:
jQuery Version 3.3.1	[REDACTED]	



Request



Response

HANDLUNGSEMPFEHLUNG:

Es wird empfohlen, die Softwarekomponenten nach Möglichkeit auf die neuesten Versionsstufen zu aktualisieren. Es bietet sich außerdem an, eine Update-Policy zu implementieren, um sicherzustellen, dass Sicherheitsupdates umgehend für alle Bibliotheken von Drittanbietern durchgeführt werden. Das Entfernen eventuell nicht mehr genutzter Bibliotheken kann die potenzielle Angriffsfläche zudem weiter reduzieren.

REFERENZEN

- [CWE-1104: Use of Unmaintained Third-Party Components](#)
- [CWE-79: Improper Neutralization of Input During Web Page Generation \('Cross-site Scripting'\)](#)

[CWE-434]: Unrestricted Upload of File with Dangerous Type

Schweregrad der Schwachstelle: **Mittel**

Host:	Path:
	

INHÄRENTE RISIKEN:

Im Laufe des Penetrationstests ist es gelungen, den Uploadfilter der Anwendung mittels zweier unterschiedlicher Methoden zu umgehen.

Die Anwendung prüft sowohl auf Dateiendungen als auch auf sog. „Magic Bytes“.

Bei Magic Bytes handelt es sich um Signaturen in Form von Hexadezimalzahlen, welche sich in der Regel am Anfang und am Ende der Datei befinden.

Mit Hilfe dieser Signaturen ist es Anwendungen möglich Dateitypen zu erkennen und die Dateien entsprechend zu verarbeiten.

Durch das Abändern der Dateiendung bzw. des Veränderns der „Magic Bytes“ ist es möglich potenziell schädliche Dateien zu tarnen und mittels der Anwendung hochzuladen.

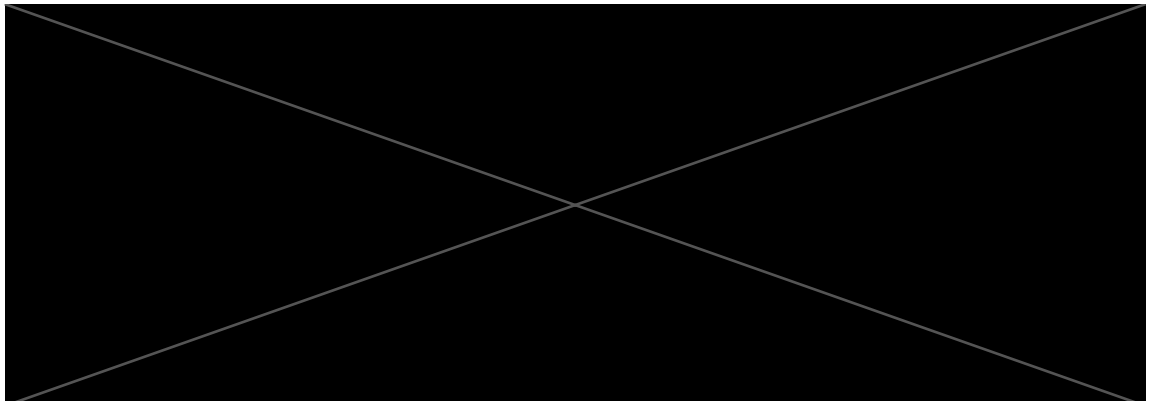
Im weiteren Verlauf kann ein Angreifer dann auf eine solche Datei zuzugreifen und die jeweilige Tarnung zu entfernen und so die hochgeladene Datei als Ausgangspunkt für weitere Angriffe auf das betroffene System verwenden.

Im Folgenden werden die beiden Vorgehensweisen ausführlich beschrieben:

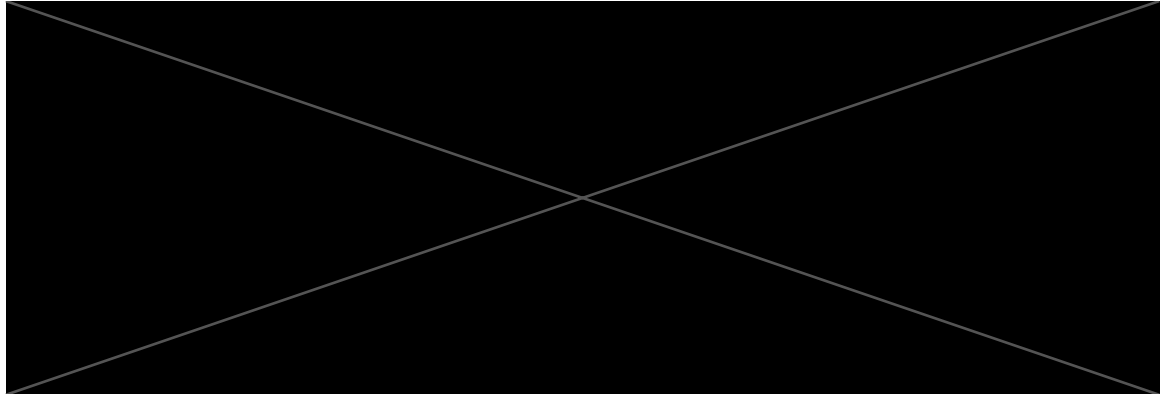
PROOF OF CONCEPT I

STEPS TO REPRODUCE

1. Öffnen Sie ein Terminal auf einem Linuxsystem.
2. Erstellen Sie mit dem folgenden Befehl eine .h-Datei:

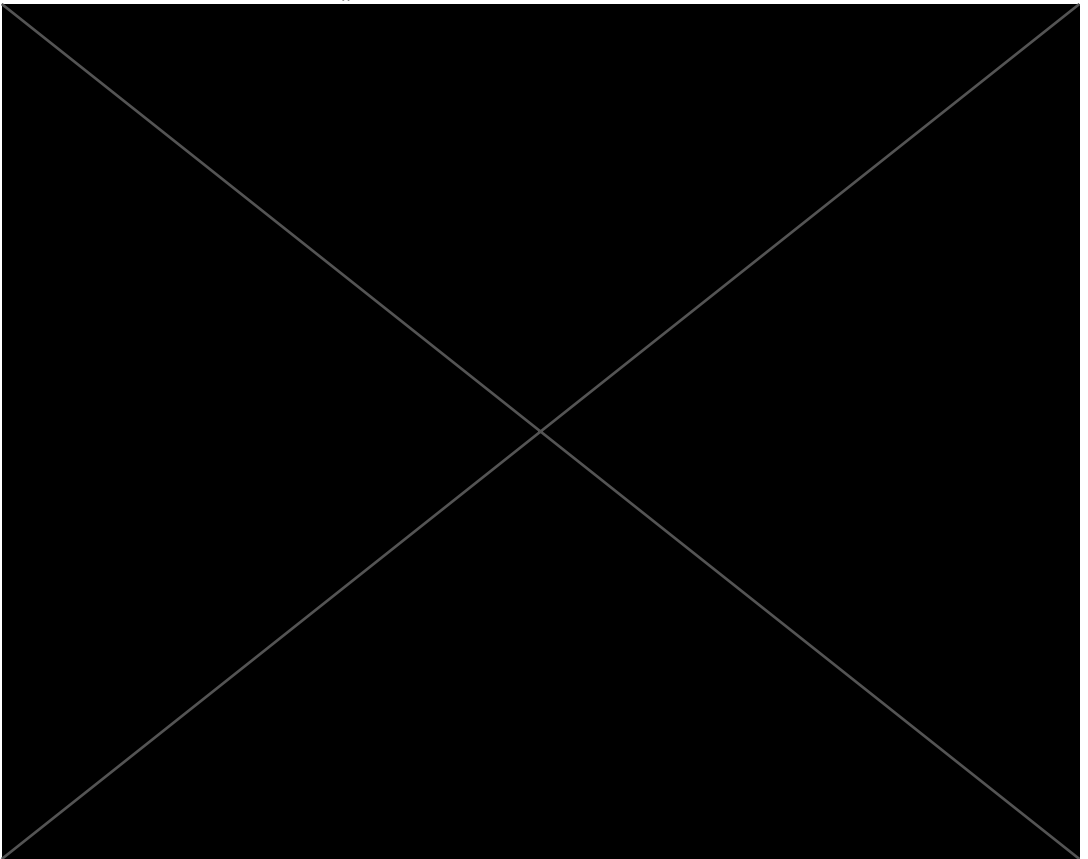


3. Durch Betätigen der Taste „i“ (im Folgenden <i>) gelangen Sie in den Bearbeitungsmodus. Geben Sie nun einen Beispielttext ein. Durch Betätigen der Escape-Taste (i.F. <Esc>) verlassen Sie den Bearbeitungsmodus. Geben sie nun nacheinander die Zeichenfolge „:x“ (i.F. <:x>) ein und drücken anschließend die Enter-Taste i.F. <Enter> . Dies speichert und schließt die Datei.

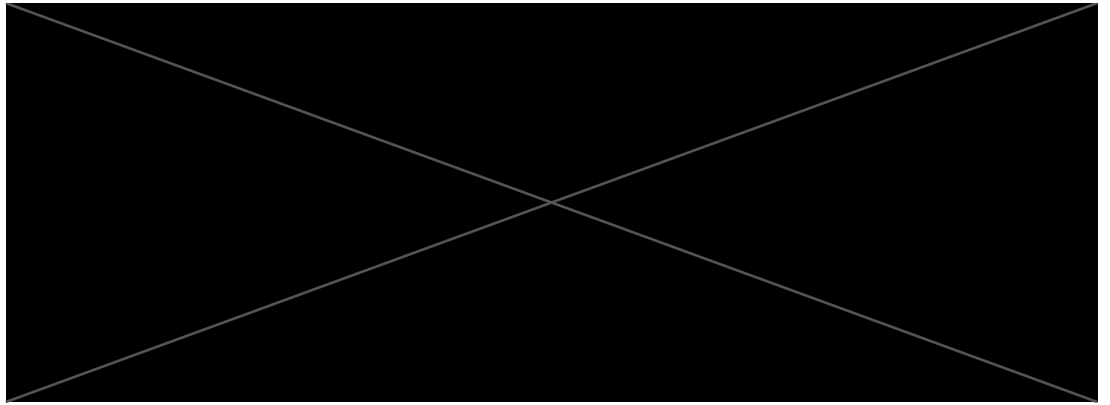


4. Laden sie die Datei nun in der Anwendung hoch.

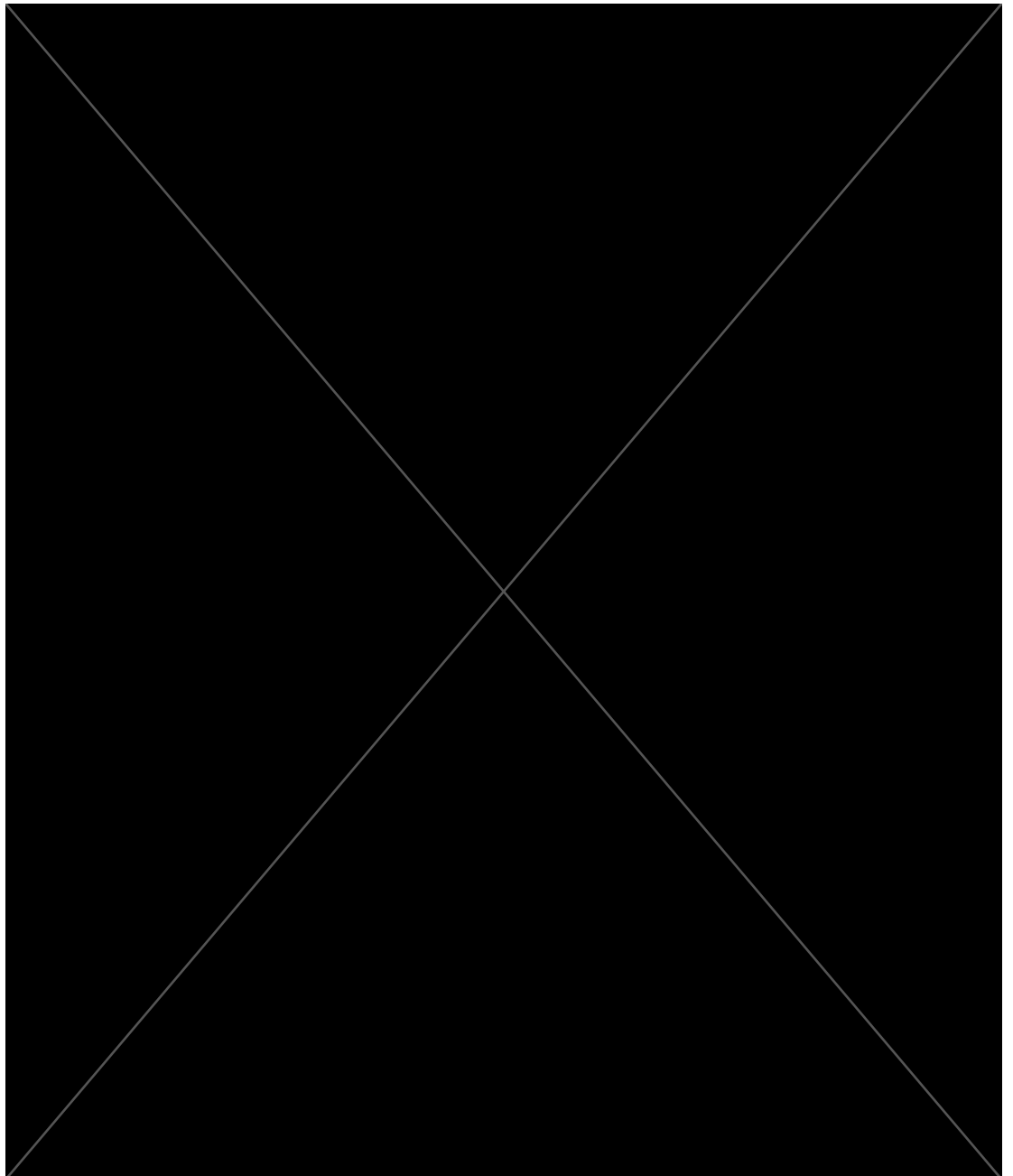
5. Ihnen wird der Fehler „Invalid File Type“ angezeigt.



6. Benennen sie die Datei nun durch folgenden Befehl in ProofOfConcept.php.rtf um:



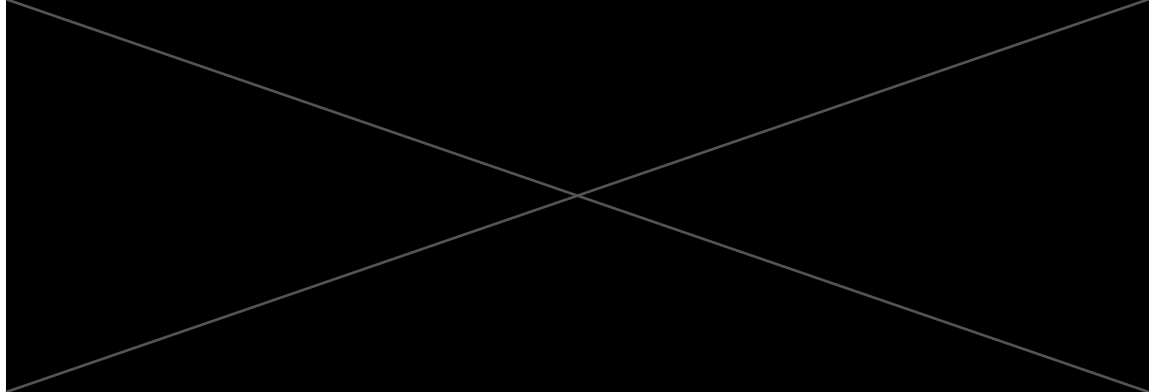
7. Wenn sie die nun umbenannte Datei hochladen, wird diese nach einer kurzen Prüfung als valide eingestuft.



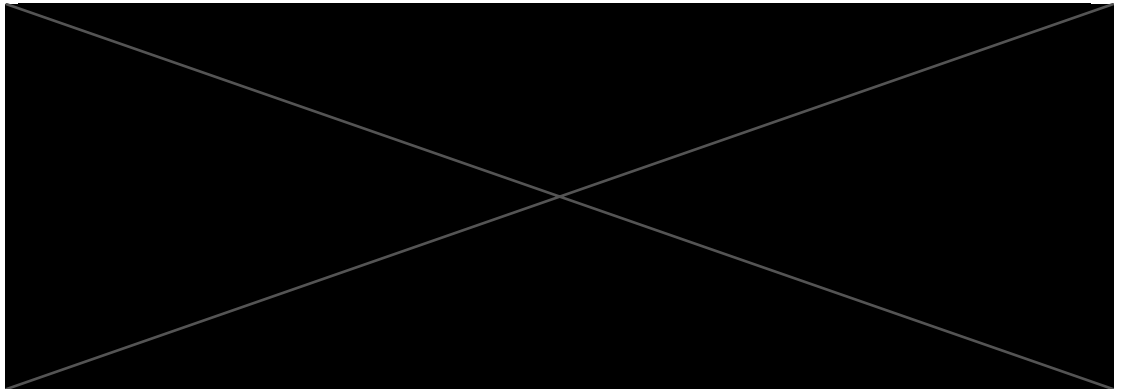
PROOF OF CONCEPT II

STEPS TO REPRODUCE

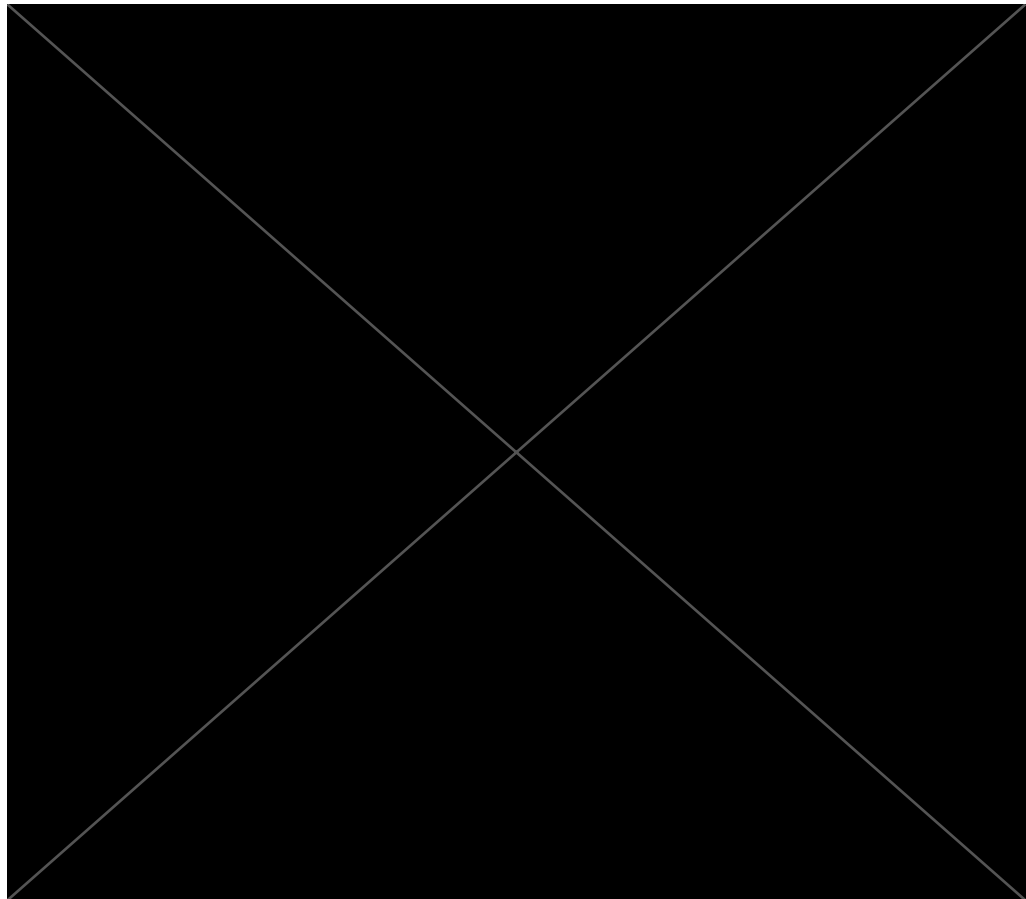
1. Öffnen Sie ein Terminal auf einem Linux System.
2. Erstellen Sie mit dem Befehl `fol` einen Befehl eine `. -Datei:`



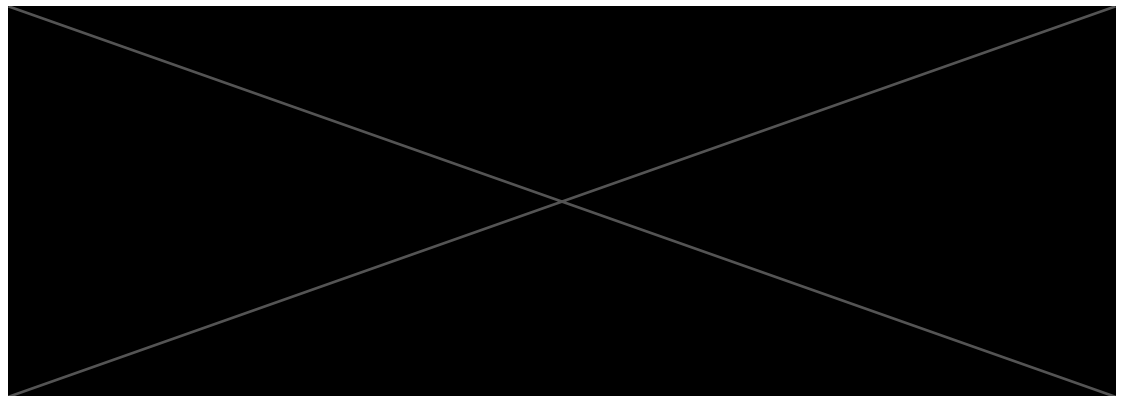
3. Fügen Sie den Power Shell Code aus der im Anhang befindlichen Datei `XXXXXXXXXXPayload.txt` ein (im Terminal: Strg-Umschalt-V), dieser Code wird von der Anwendung **nicht** ausgeführt. Führt man den Code manuell aus, sendet er lediglich eine OOB-Anfrage an einen unserer Server. Der Code soll hier lediglich als Beispiel dienen.



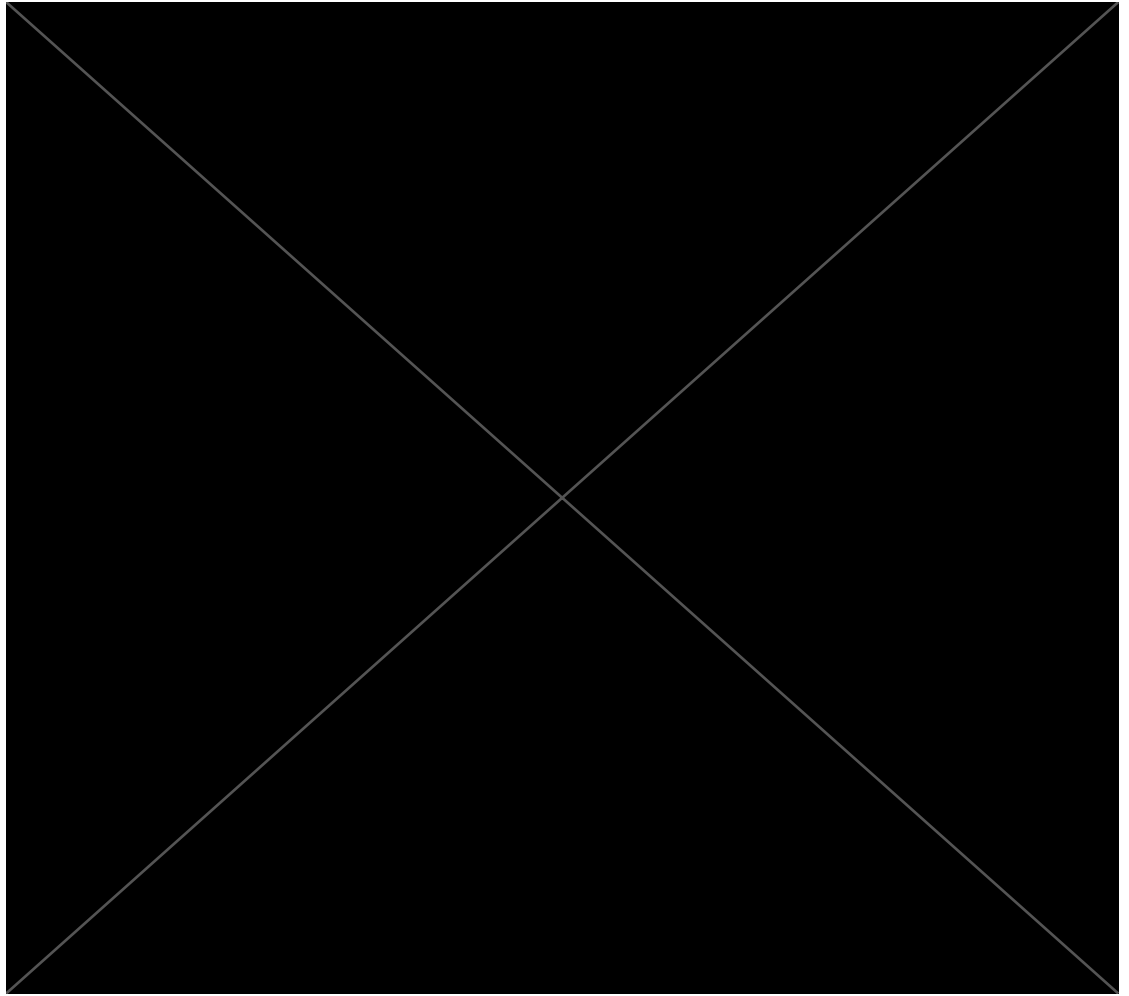
4. Speichern und schließen sie die Datei mit `<:x>-<Enter>`.
5. Laden sie die Datei nun hoch (sie wird einige Sekunden überprüft).



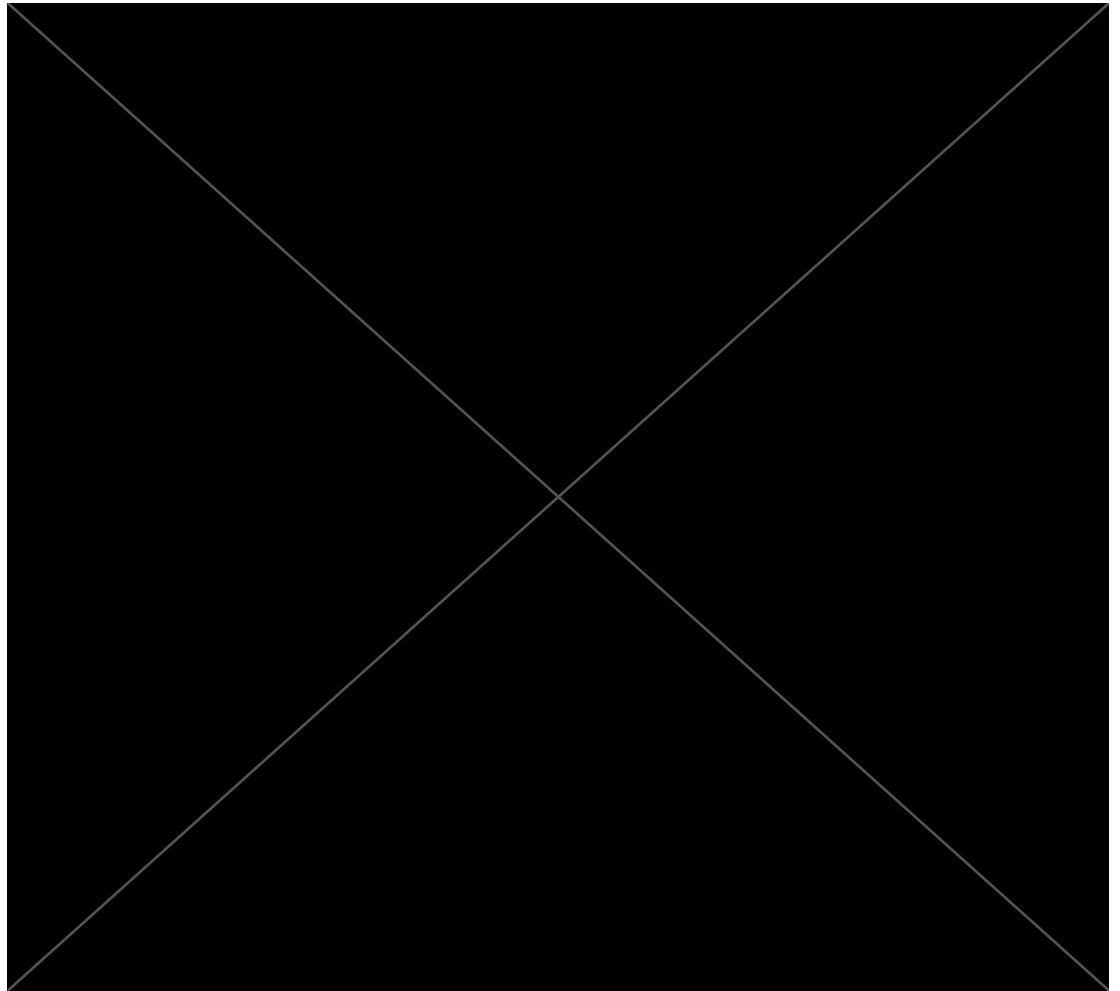
6. Der Server überprüft, ob es sich um eine valide .jpg-Datei handelt. Da dies nicht der Fall ist, verwirft er die Datei.
7. Öffnen Sie die Datei nun in einem Hex Editor:



8. Verändern Sie nun die ersten und letzten Bytes in die benötigten „Magic Bytes“, welche eine .jpg-Dateien ausweisen:
Anfang: FFD8 FFDB
Ende: FFD9

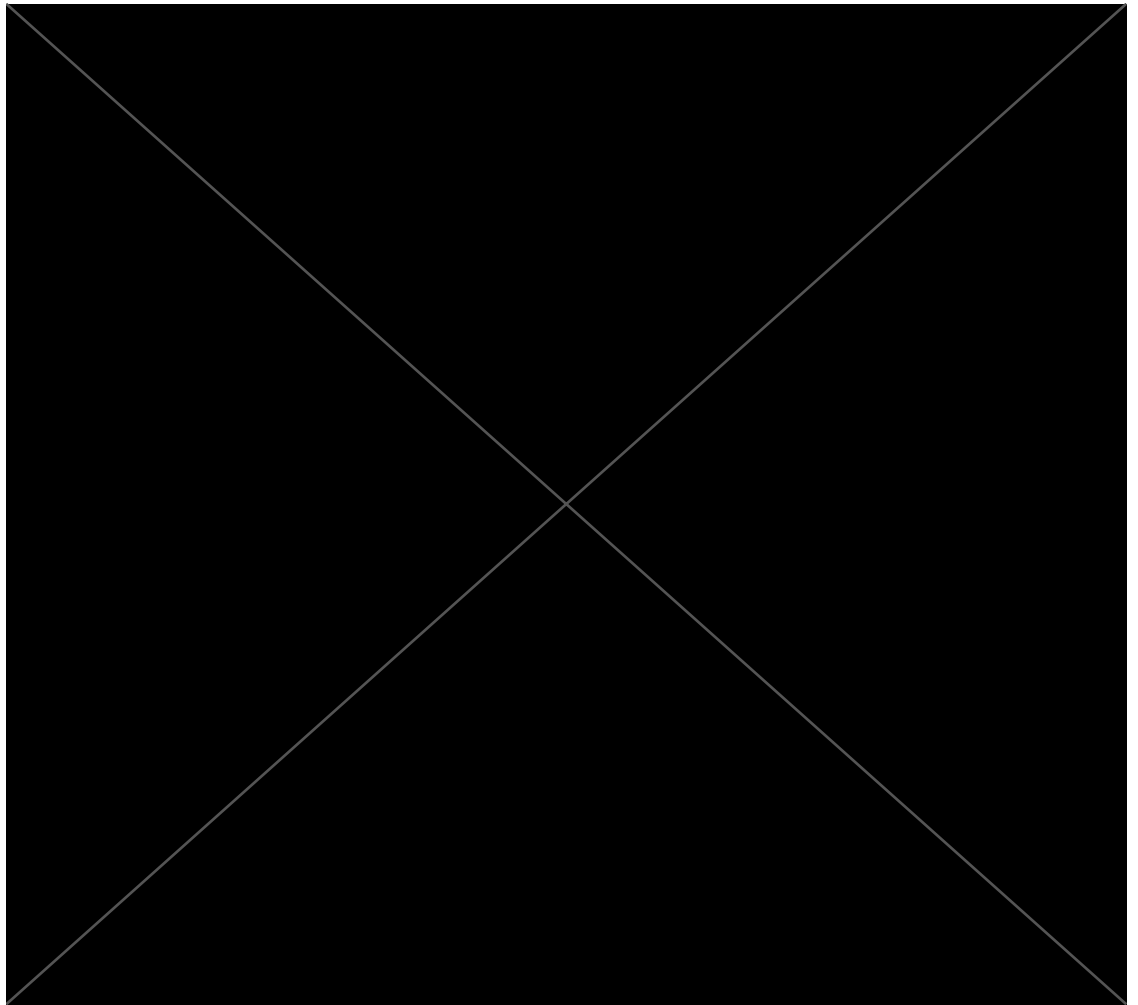


Vorher

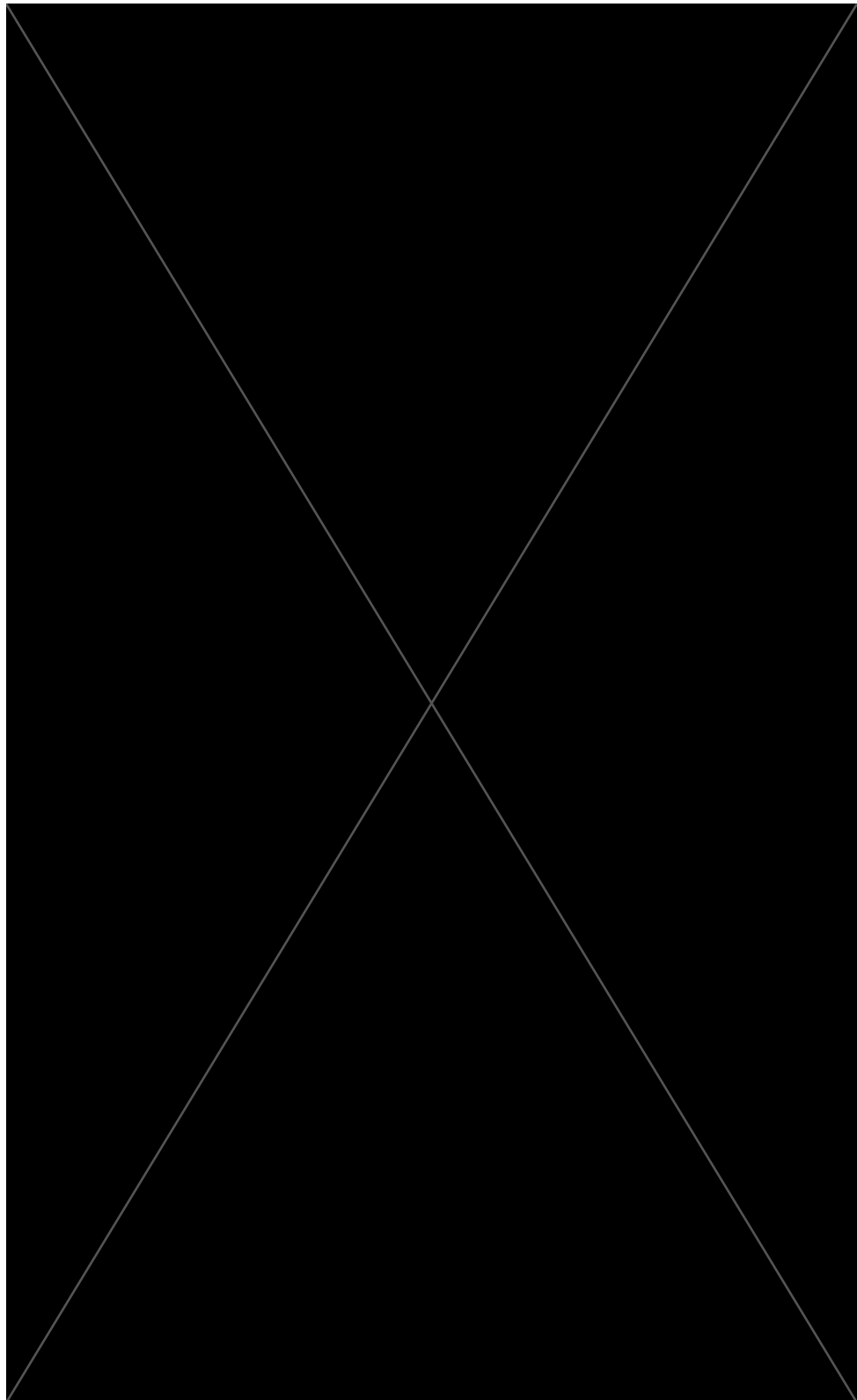


Nachher

9. Speichern Sie Ihre Änderungen mit Strg-O und bestätigen Sie den Dialog mit „y“. Geben Sie anschließend ProofOfConcept.jpg ein und bestätigen Sie abermals mit <Enter>.



10. Laden Sie die Datei nun erneut hoch, sie wird abermals einige Sekunden überprüft.



11. Diesmal wird sie jedoch vom Server akzeptiert.

HANDLUNGSEMPFEHLUNG:

Der Uploadfilter sollte eine Inspektion vornehmen, die über die Prüfung der Dateiendung und der „Magic Bytes“ hinausgeht.

Eine Möglichkeit die Prüfung des Dateinamens sicherer zu gestalten und die im ersten Proof of Concept dargestellte Umgehung zu verhindern ist die Verwendung von Regular Expressions. Hier ein Beispiel:

```
<String ohne Punkte><1xPunkt><ext(WL)>
```

Wobei mit WL eine Whitelist der erlaubten Dateiendungen gemeint ist.

Es sollte immer angenommen werden, dass alle Eingaben bösartig sind. Daher sollte eine "Accept known good"-Eingabevalidierungsstrategie verwendet werden. Dies bedeutet, dass eine Liste mit akzeptablen Eingaben den Input prüft und alles was nicht den Spezifikationen entspricht, verwirft oder umwandelt.

Bei der Eingabevalidierung sollten alle potenziell relevanten Eigenschaften wie Länge, Art und Syntax der Eingabe mit den erwarteten Eingaben übereinstimmen.

Beim Upload einer Datei sollte z.B. überprüft werden, ob es sich um eine gültige Datei handelt. Wenn es sich etwa um ein .jpg handelt, kann mittels Funktionen wie `getimagesize()` ermittelt werden, ob es sich um ein gültiges Bild handelt.

MIME Sniffing ist eine weitere Möglichkeit die Dateien einer genaueren Prüfung zu unterziehen und sicherzustellen, dass nur valide Dateien auf den Server geladen werden. Für weitere Informationen zu diesem Thema haben wir einige Links in den Referenzen bereitgestellt.

Zudem sollte jede Datei, die von Dritten auf Ihre Server hochgeladen wird einer Malware-Prüfung unterzogen werden.

REFERENZEN

- [CWE-434: Unrestricted Upload of File with Dangerous Type](#)
- [Regex: Learn, Build & Test Regular Expressions](#)
- [MIME Sniffing](#)
- [DetectContentType](#)
- [File Upload Protection – 10 Best Practices for Preventing Cyber Attacks](#)

[CWE-200]: Ungeschützte SQL-Datenbank

Schweregrad der Schwachstelle: **Mittel**

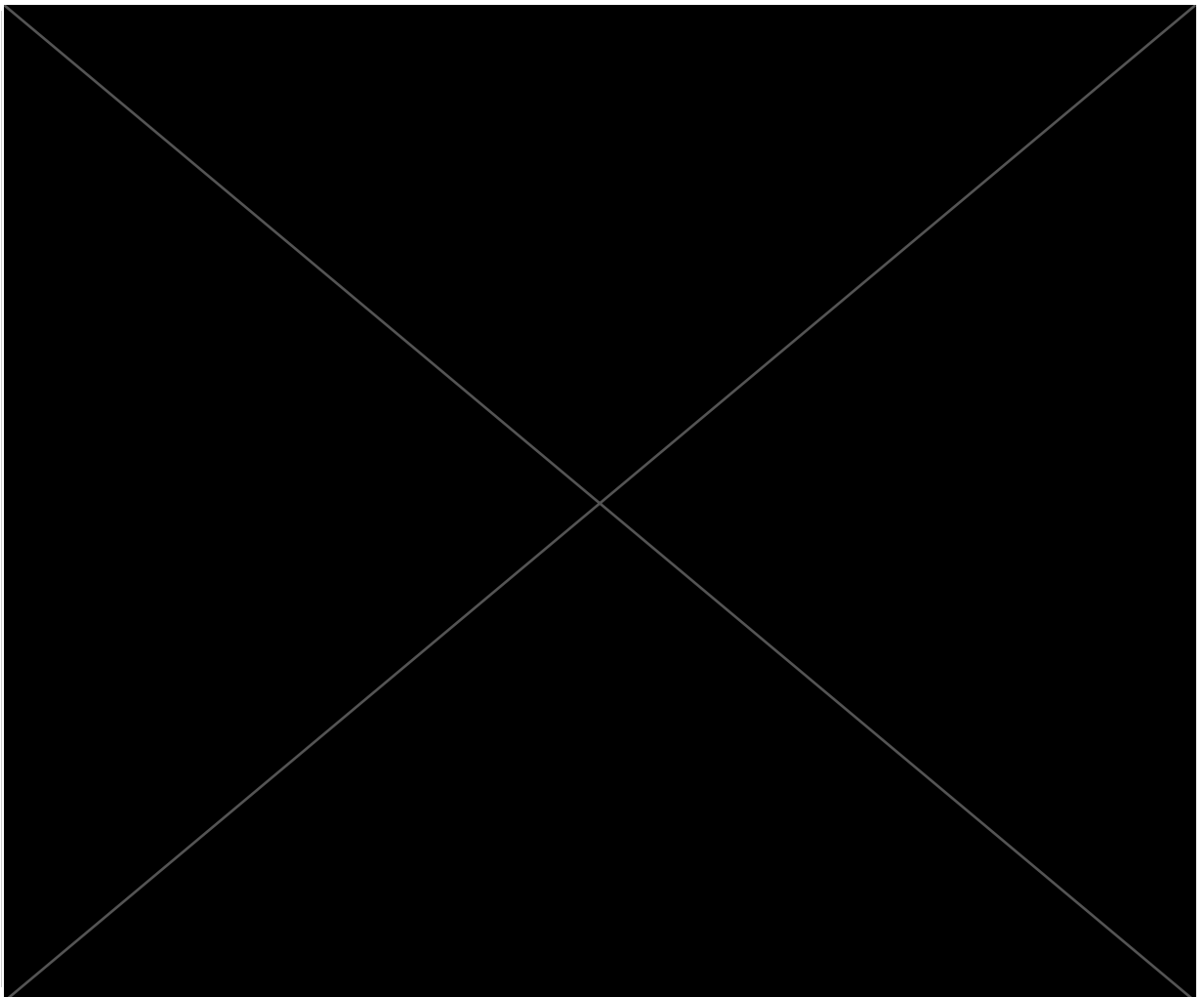
Host IP	Port
XXXXXXXXXXXXXXXXXXXX	

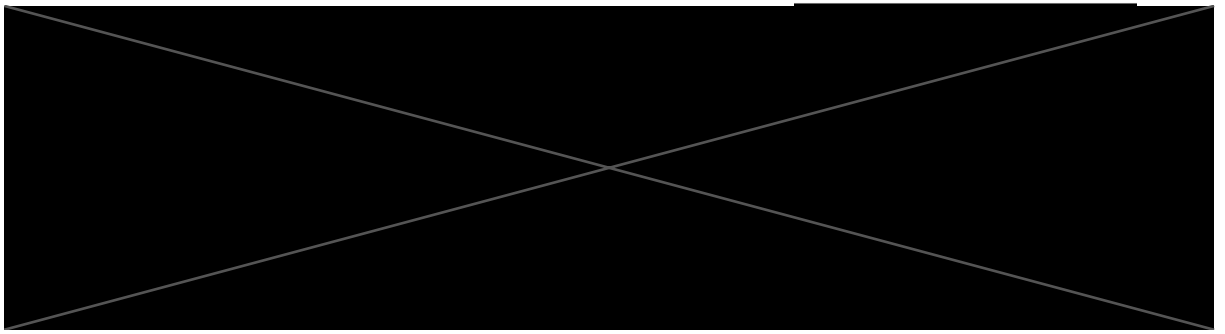
INHÄRENTE RISIKEN:

Unzureichend gesicherte SQL-Datenbanken können es einem potenziellen Angreifer ermöglichen, unautorisiert an Daten zu gelangen. Durch mangelnde Sicherheitsmaßnahmen kann die Struktur der Datenbank erfasst werden, wodurch weitere Informationen freigelegt werden können.

In diesem Falle konnten wir keine der üblichen Sicherheitspraktiken wie die Limitierung von Anfragen, das Verschleiern der Versionsnummern und Capabilities der Datenbank oder das Entfernen der Standardnutzer feststellen.

PROOF OF CONCEPT





HANDLUNGSEMPFEHLUNG:

Die Datenbank kann besser geschützt werden, wenn Anfragen an diese begrenzt werden und nur autorisierte Requests beantwortet werden. Weitere Möglichkeiten die Datenbank abzusichern, bestehen darin, die Datenbank hinter einer Firewall zu verbergen und eine Policy einzuführen.

Policy beschreibt in diesem Zusammenhang sowohl Passwortrichtlinien als auch das Rollenvergabe- und Berechtigungssystem. Im Zuge der Rollenvergabe sollten auch Standardbenutzer entfernt werden, da diese Enumerationversuche begünstigen können.

REFERENZEN

- [CWE-200: Exposure of Sensitive Information to an Unauthorized Actor](#)
- [MySQL Enum](#)
- [11 Steps to Secure SQL in 2022](#)

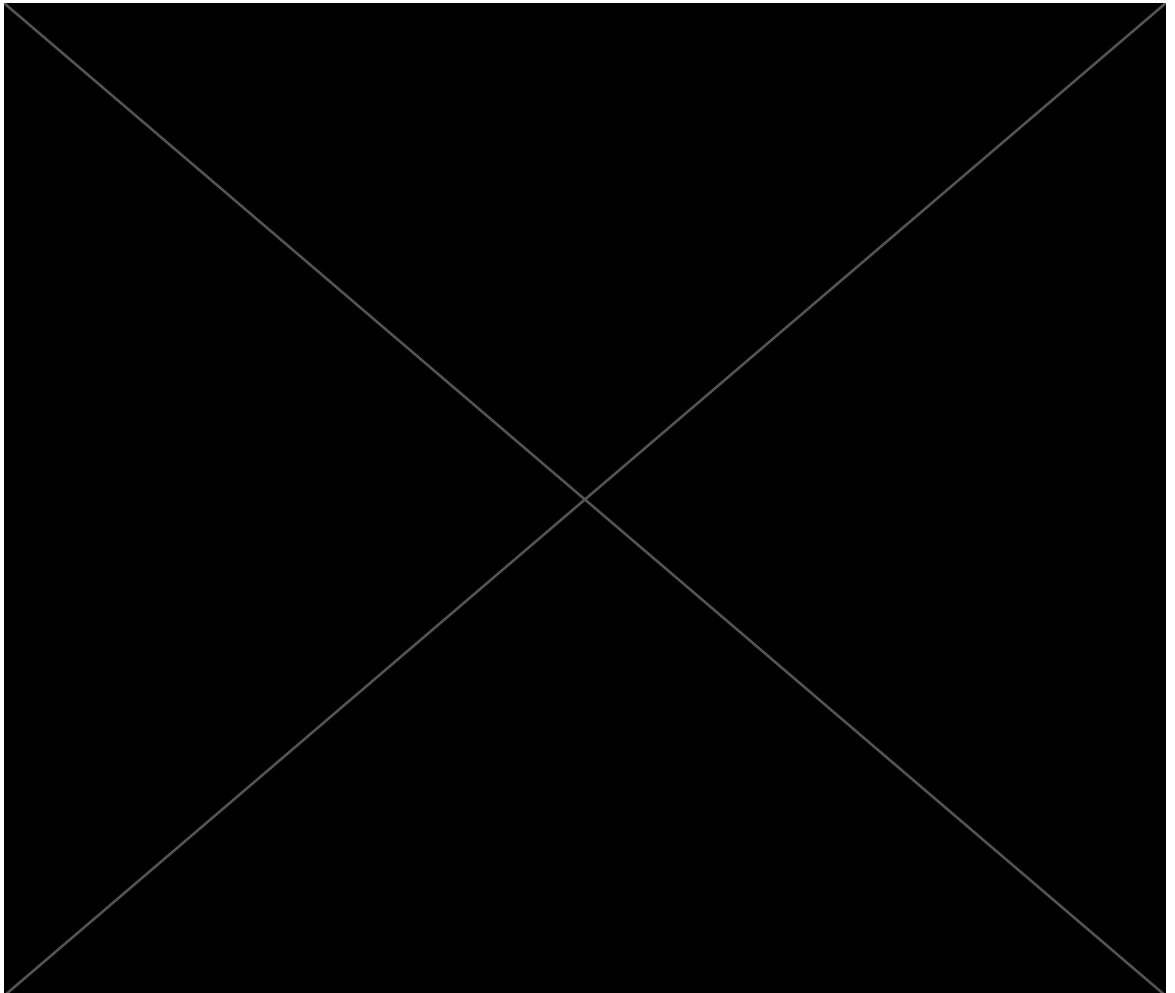
[CWE-552]: Directory/File/API - Enumeration

Schweregrad der Schwachstelle: Gering

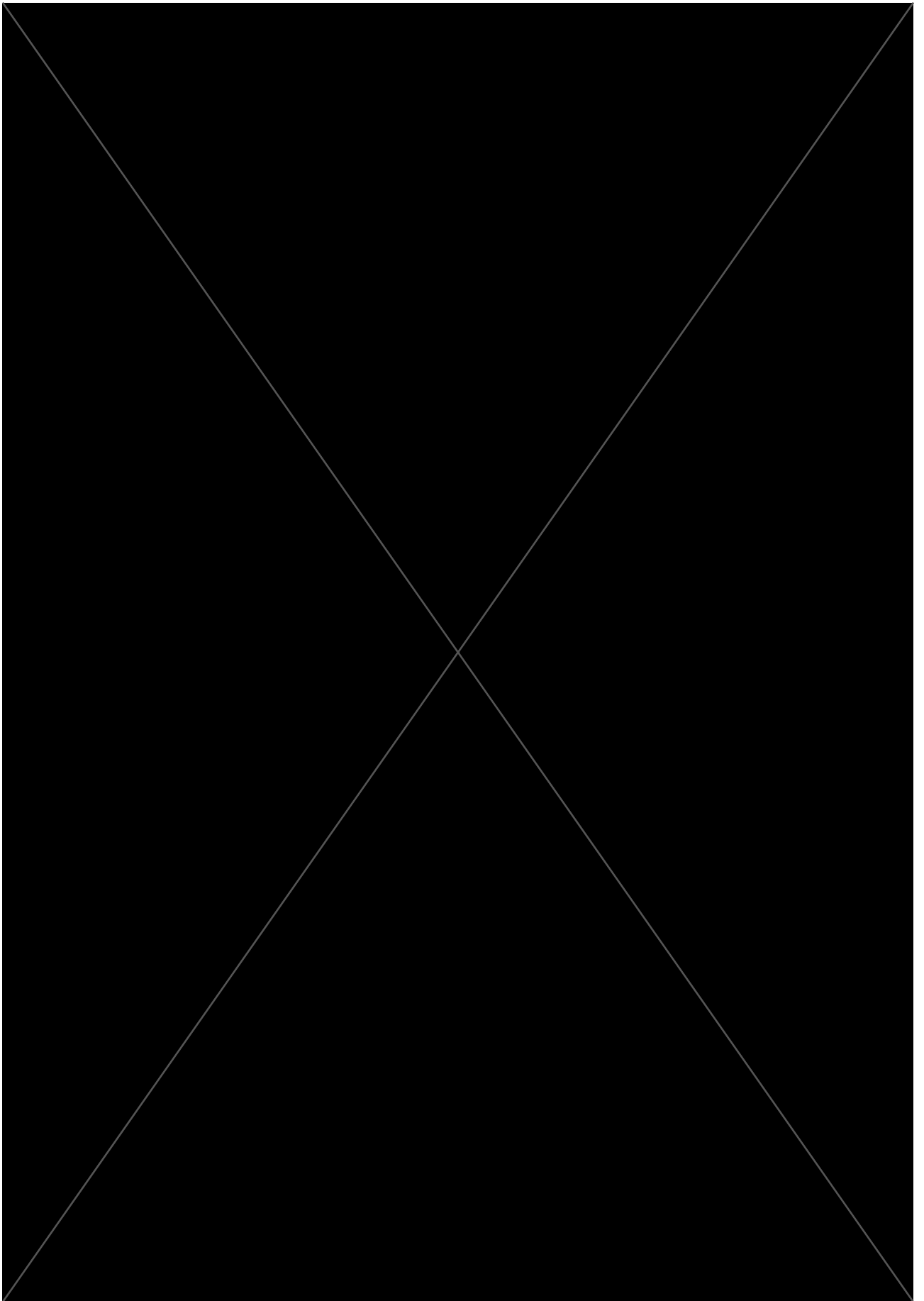
INHÄRENTE RISIKEN:

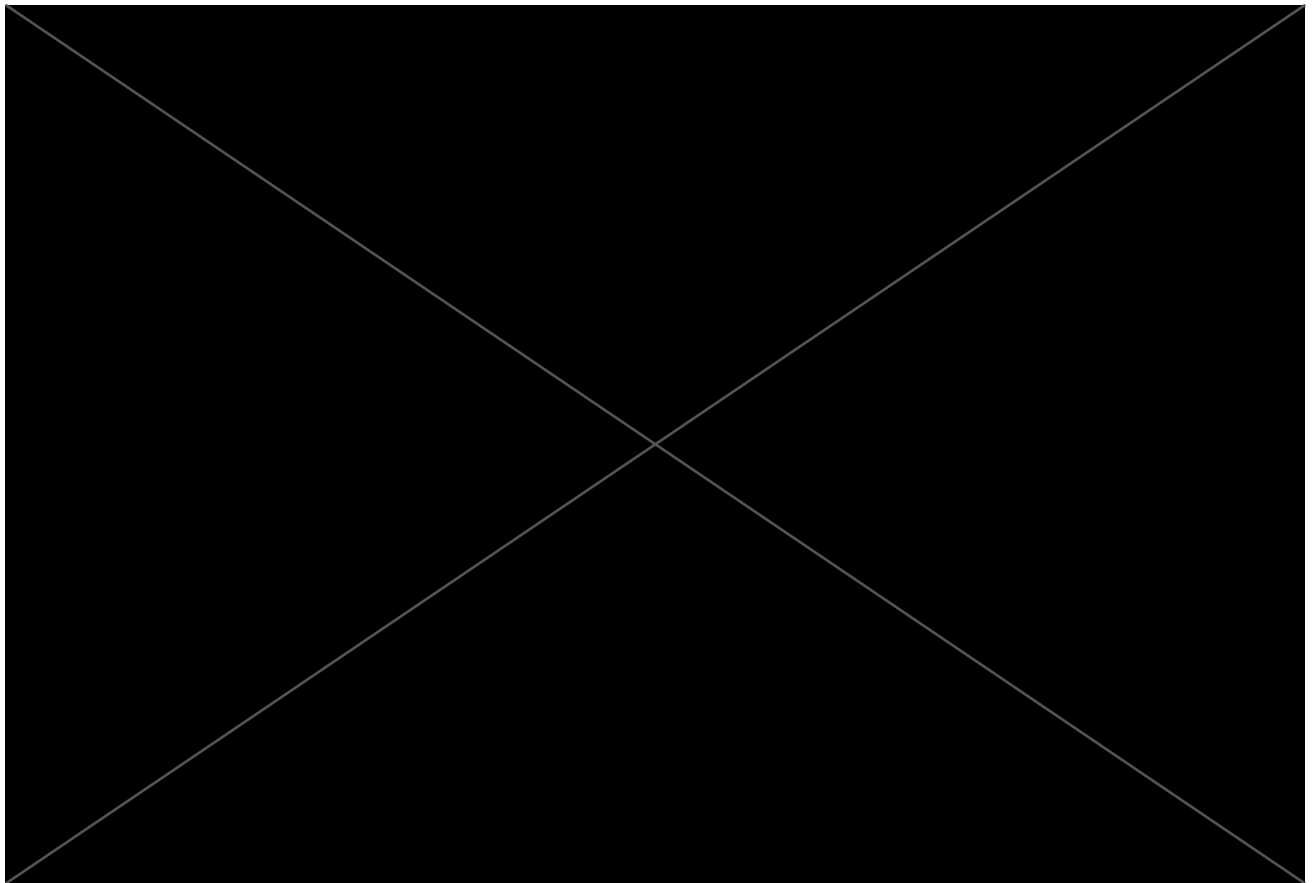
Es ist möglich das Directory des Webservers zu enumerieren. Dadurch können alle Verzeichnisse und Pfade offengelegt werden. Somit kann ein Angreifer Einblicke in die Struktur des Webservers gewinnen und Informationen über potenzielle Angriffsziele generieren. Schlimmstenfalls findet der Angreifer einen Pfad, welcher ohne Authentifizierung aufrufbar ist und sensible Daten enthält.

PROOF OF CONCEPT



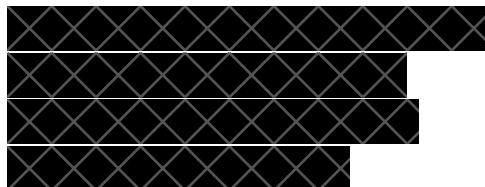
NTLM Directories:





Eine Übersicht der durch die Enumeration ermittelten Informationen finden sie im Anhang in den folgenden Dokumenten:

- Fuzzing_Full.pdf



- Extrahierbare_JS_JSON-Files.pdf

HANDLUNGSEMPFEHLUNG:

Verzeichnisstrukturen sollten für Dritte niemals enumerierbar oder klar ersichtlich sein. Daher sollten diese verborgen und Anfragen für Directory Listing geblockt werden.

Es wird zudem empfohlen, die Servereinstellungen so zu konfigurieren, dass keine detaillierten Informationen über Fehler und Benutzerkontodaten angezeigt werden. Sobald eine Mail eingegeben wird, sollte der Server eine generische Rückmeldung geben. Dies sollte nach dem [Minimalprinzip](#) geschehen.

Stellen Sie sicher, dass die Fehlermeldungen nur minimale Details enthalten, die für die Zielgruppe nützlich sind. Nur so kann die Einhaltung des [Minimalprinzips](#) gewährleistet werden. Die Meldungen sollten nicht die Methoden offenlegen, die zur Ermittlung des Fehlers verwendet wurden. Angreifer können detaillierte Informationen nutzen, um

ihren ursprünglichen Angriff zu verfeinern oder zu optimieren und so ihre Erfolgchancen zu erhöhen.

Wenn Fehler detailliert erfasst werden müssen, sollten diese in Logdateien festgehalten werden. Diese Logdateien wiederum müssen ebenfalls vor dem Zugriff durch Unbefugte geschützt werden. Hochsensible Informationen wie Passwörter sollten jedoch auch nicht in Logdateien gespeichert werden.

Vermeiden Sie inkonsistente Meldungen, die einem Angreifer versehentlich Aufschluss über den internen Status geben könnten, z.B. ob ein Benutzerkonto existiert oder nicht.

REFERENZEN

- [CWE-552: Files or Directories Accessible to External Parties](#)
- [Directory Listing des Apache-Webservers deaktivieren](#)

[CWE-310]: Base64-encoded Data in Parameter

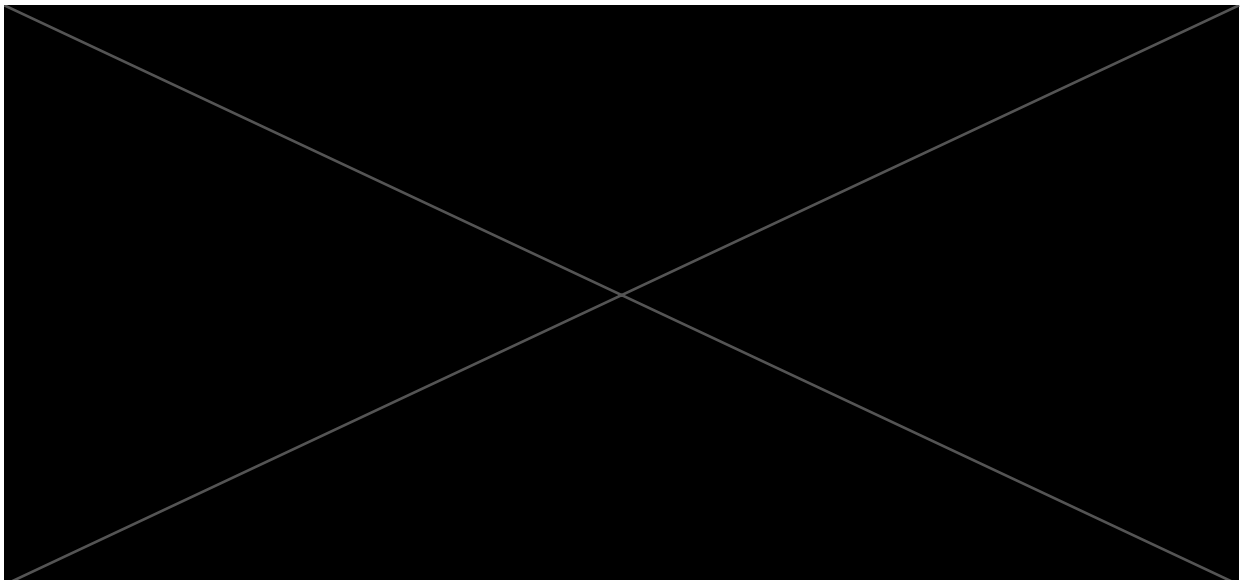
Schweregrad der Schwachstelle: Gering

INHÄRENTE RISIKEN:

Anwendungen verschlüsseln streckenweise Parameter mit Base64, um sie vor dem Benutzer zu verbergen oder den Transport von Binärdaten zu erleichtern. Das Vorhandensein von Base64-kodierten Daten weist oftmals auf sicherheitsrelevante Informationen oder Funktionen hin, welche für einen potenziellen Angreifer von Interesse sein könnten.

PROOF OF CONCEPT

Host:	Path:
	



Response Decrypte

HANDLUNGSEMPFEHLUNG:

Base64 ist nicht mehr zeitgemäß und sollte durch andere State-of-the-Art Hashfunktionen, wie beispielsweise SHA265, ersetzt werden. Die Daten sollten außerdem überprüft werden, um sicherzugehen, dass keine sensiblen Daten enthalten sind, welche eine zusätzliche Angriffsfläche darstellen könnten.

REFERENZEN

- [CWE CATEGORY: Cryptographic Issues](#)
- [Passworthashing - Aber sicher!](#)

[CWE-610]: Externally Controlled Reference to a Resource in Another Sphere (Redirection from HTTP to HTTPS)

Schweregrad der Schwachstelle: Gering

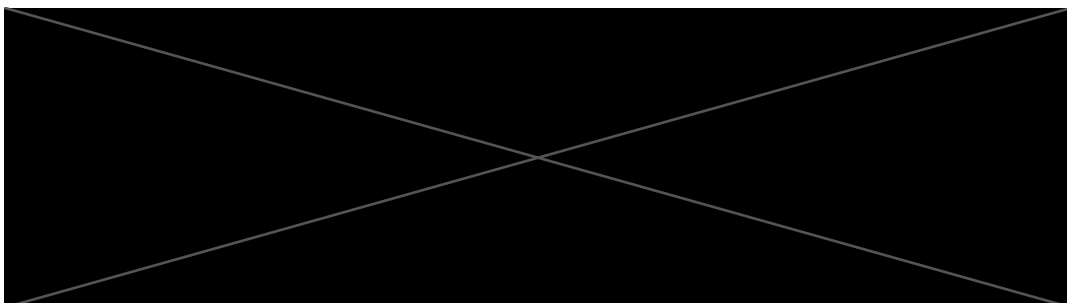
INHÄRENTE RISIKEN:

Die Umleitung auf eine HTTPS-URL wird über das unsichere HTTP-Protokoll übertragen. Dadurch wird die Umleitung selbst anfällig für Man-in-the-Middle-Angriffe. Ein Angreifer könnte den Benutzer zu einer etwas anderen HTTPS-URL umleiten, die sich unter seiner Kontrolle befindet, oder die Verbindung unverschlüsselt halten, indem sie auf HTTP heruntergestuft und zwischen Client und Server weitergeleitet wird.

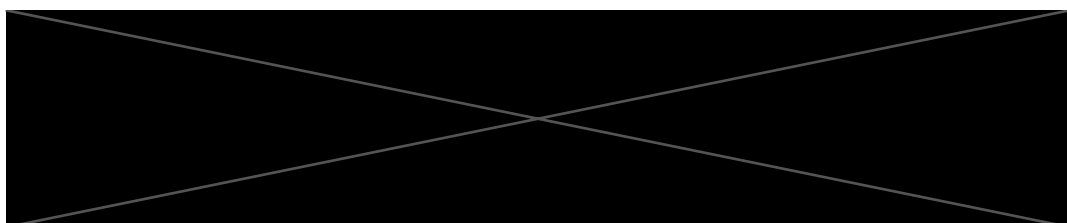
Die Schwachstelle wurden an folgenden Orten gefunden:

Host:	Path:
[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]

PROOF OF CONCEPT



Request



Response

HANDLUNGSEMPFEHLUNG:

Die Verwendung von HTTP sollte in Webanwendungen mit hohem Schutzbedarf auf ein Minimum beschränkt werden. Benutzer, die die Webanwendung über HTTP aufrufen, z.B. indem sie nur den Domainnamen in die URL-Leiste ihres Browsers eingeben, sollten direkt auf eine sichere HTTPS-URL umgeleitet werden. Alle HTTPS-Ressourcen sollten einen Strict-Transport-Security-Header bereitstellen, der sicherstellt, dass der Browser für einen bestimmten Zeitraum nur HTTPS verwendet. Die Syntax für diesen Header lautet wie folgt:

Strict-Transport-Security: max-age=<seconds>[; includeSubDomains]

Die Expire Time gibt den Zeitrahmen in Sekunden vor, in welchem der Browser des Benutzers das HTTPS-Protokoll verwenden soll und sollte daher recht hoch gewählt werden (mehrere Monate). Es kann sich außerdem anbieten die zusätzliche Flag „*includeSubDomains*“ zu verwenden um diese Einstellungen für alle Subdomains der Webseite zu übernehmen. Bis auf die initiale Umleitung sollte die Anwendung komplett mit HTTPS genutzt werden.

REFERENZEN

- [CWE-610: Externally Controlled Reference to a Resource in Another Sphere](#)
- [95% of HTTPS servers vulnerable to trivial MITM attacks](#)
- [HTTP Strict Transport Security](#)

[A01: 2021]: Broken Access Control

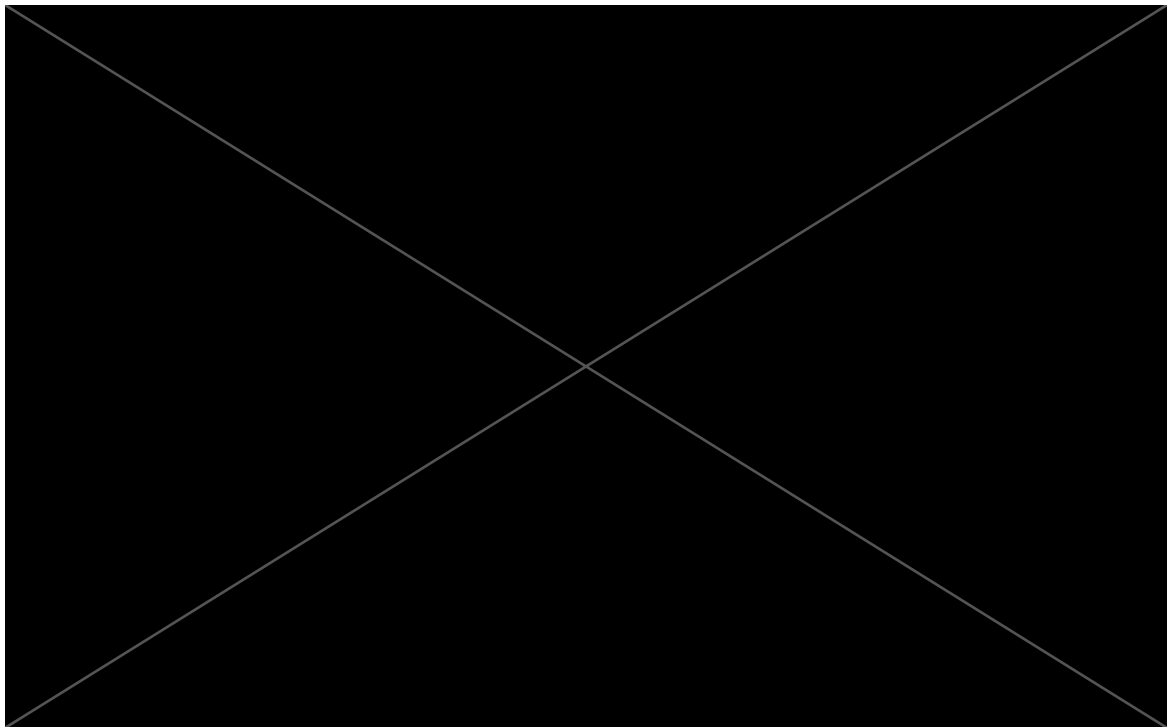
Schweregrad der Schwachstelle: **Information**

Host:	Pfad:
[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]
[REDACTED]	[REDACTED]

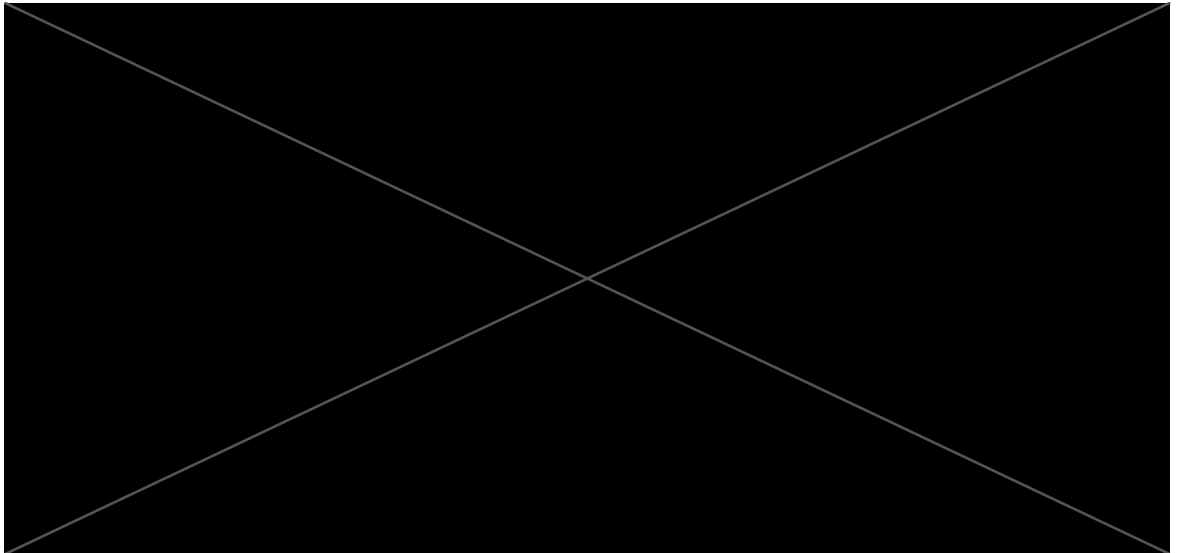
INHÄRENTE RISIKEN:

Wenn Login-Interfaces nicht geschützt oder versteckt sind, so können Angreifer Einblicke in die genutzten Funktionen und Software des Unternehmens bekommen. Außerdem können bei fehlender Limitierung der Login-Versuche Brute-force Angriffe durchgeführt werden. Weiterhin könnten bereits gefundene/gestohlene Anmeldedaten genutzt werden, um sich in diesen Login Pages zu authentifizieren und so Zugriff auf sensible Daten zu erhalten.

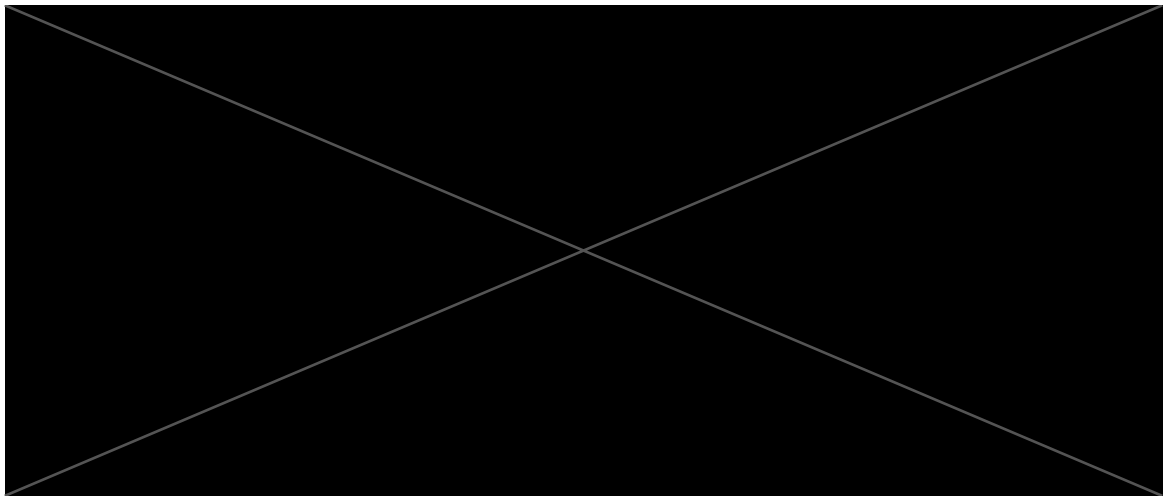
PROOF OF CONCEPT



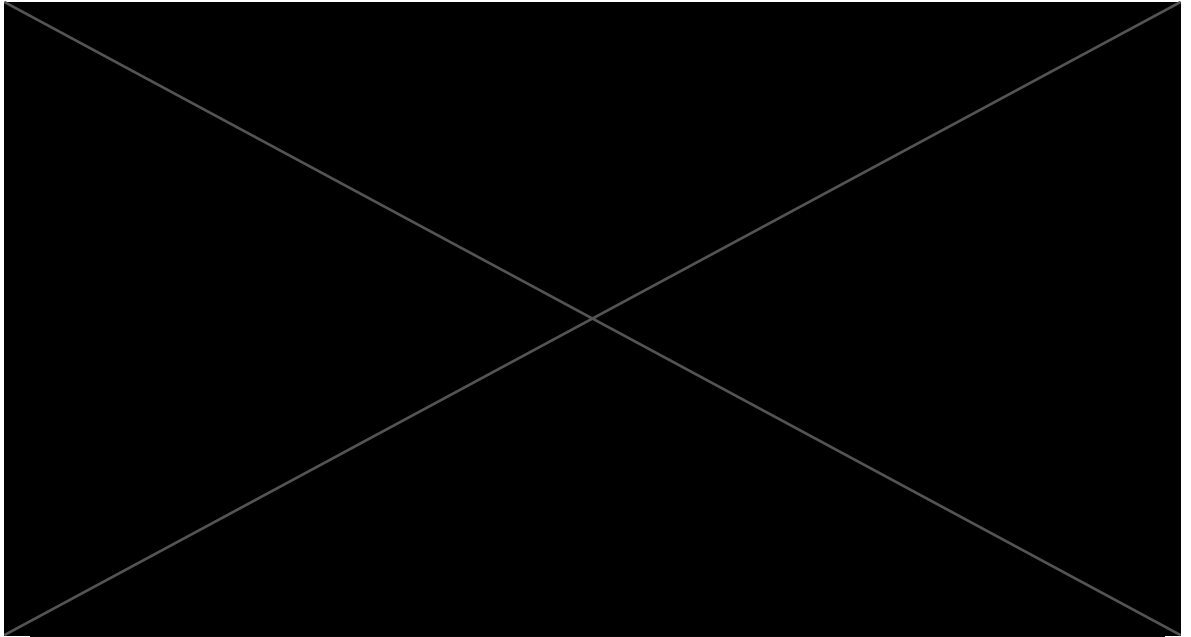
Login [REDACTED]



Login Outlook



Login SharePoint



Login WordPress

HANDLUNGSEMPFEHLUNG:



Login Seiten sollten nicht unter den standardisierten URLs erreichbar sein. Stattdessen sollten sie Seiten unter nicht offensichtlichen Namen erreichbar sein. Beispielsweise könnte die WordPress Login Seite unter XXXXXXXXXX hinterlegt werden. Ebenfalls kann ein serverseitiger Schutz durch die .htaccess Datei erfolgen. Darin kann die Datei wp-login.php durch ein weiteres Passwort geschützt werden.

REFERENZEN

- [Poor Web Security Practices: Unsecured Login Forms](#)
- [WordPress Login schützen](#)

[CWE-540]: Inclusion of Sensitive Information in Source Code

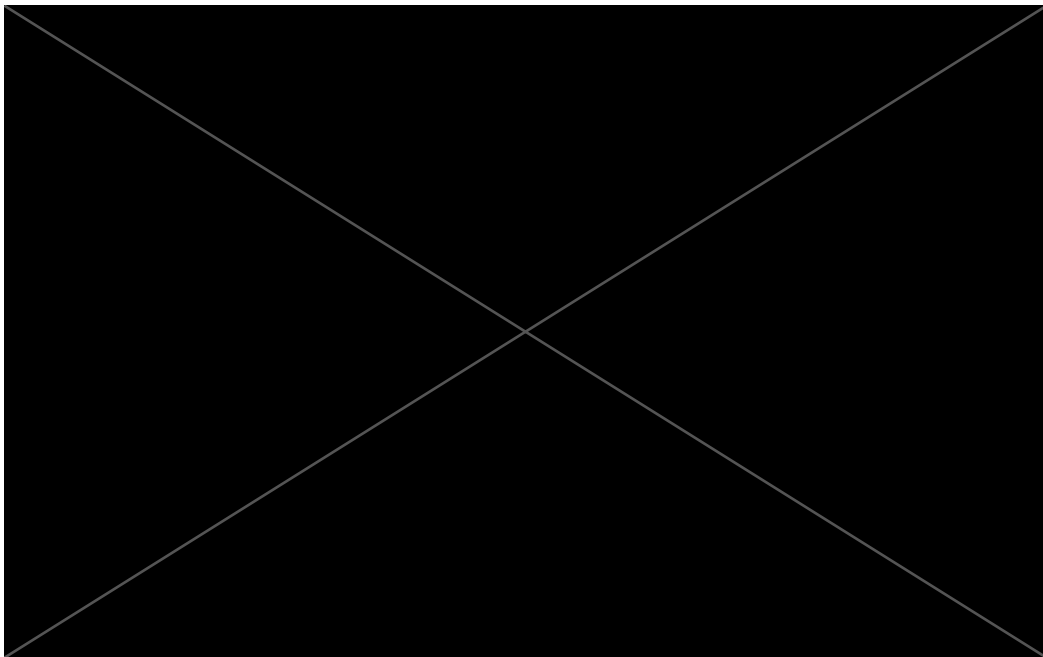
Schweregrad der Schwachstelle: [Information](#)

Host	Path
	

INHÄRENTE RISIKEN:

Informationen, welche im Quelltext liegen, aber keine Funktion erfüllen, stellen eine Gefahr dar. Angreifer können im schlimmsten Fall die Logik von Skripten oder Funktionen nachvollziehen und äußerst nützliche Informationen wie Code-Fehler oder Logins und Passwörter extrahieren. Auch in Konfigurationsdateien wie der /web.config, sollten keine sensiblen Daten enthalten sein.

PROOF OF CONCEPT



Source Code

HANDLUNGSEMPFEHLUNG:

Alle sensiblen Daten (Benutzeranmeldeinformationen, API-Schlüssel usw.) sollten aus dem Programmcode ausgeschlossen werden, um einen Angreifer daran zu hindern, mit diesen Daten auf das System zuzugreifen. Zudem sollten generell Informationen und Kommentare aus dem Quelltext entfernt werden, sowie auch Funktionen und Skripte welche nicht mehr von Bedeutung sind. Sourcecode sollte Dritten nur in einer bereinigten Form zugänglich sein.

REFERENZEN

- [CWE-540: Inclusion of Sensitive Information in Source Code](#)


[CWE-693]: Protection Mechanism Failure (Strict Transport Security Misconfiguration)

Schweregrad der Schwachstelle: [Information](#)



INHÄRENTE RISIKEN:

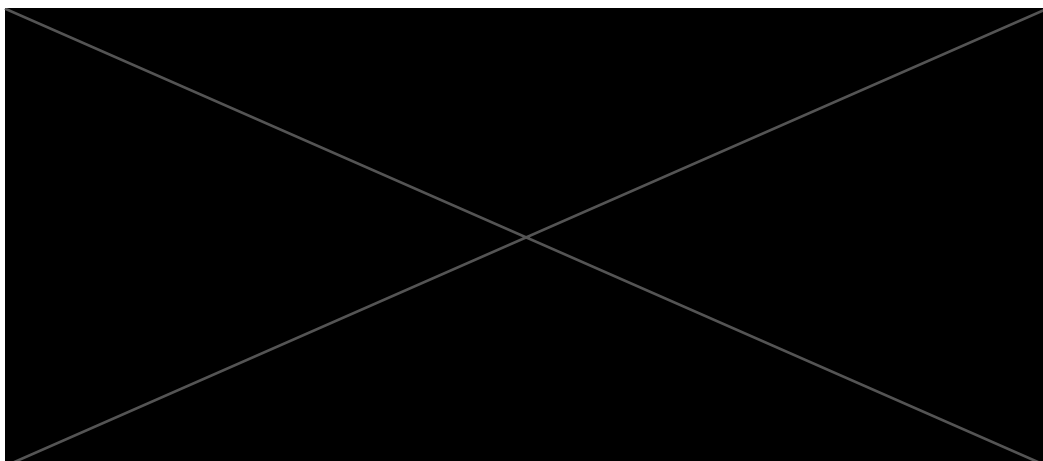
Die HTTP Strict Transport Security-Richtlinie definiert einen Zeitrahmen, in dem ein Browser über HTTPS eine Verbindung zum Webserver herstellen muss. Ohne eine Strict Transport Security-Richtlinie kann die Webanwendung gegen mehrere Angriffe anfällig sein:

- Wenn die Webanwendung die Verwendung von HTTP und HTTPS mischt, kann ein Angreifer Seiten im ungesicherten Bereich der Anwendung manipulieren oder Umleitungsziele so ändern, dass der Wechsel zur gesicherten Seite nicht erfolgt oder so, dass der Angreifer zwischen Client und Server verharrt.
- Wenn es keinen HTTP-Server gibt, könnte ein Angreifer im selben Netz einen HTTP-Server simulieren und den Benutzer durch einen Social-Engineering-Angriff dazu bringen, auf eine präparierte URL zu klicken. Der Schutz ist nur für die vorgegebene Zeit wirksam. Das mehrfache Vorkommen dieses Headers kann zu undefiniertem Verhalten in Browsern führen und sollte vermieden werden.

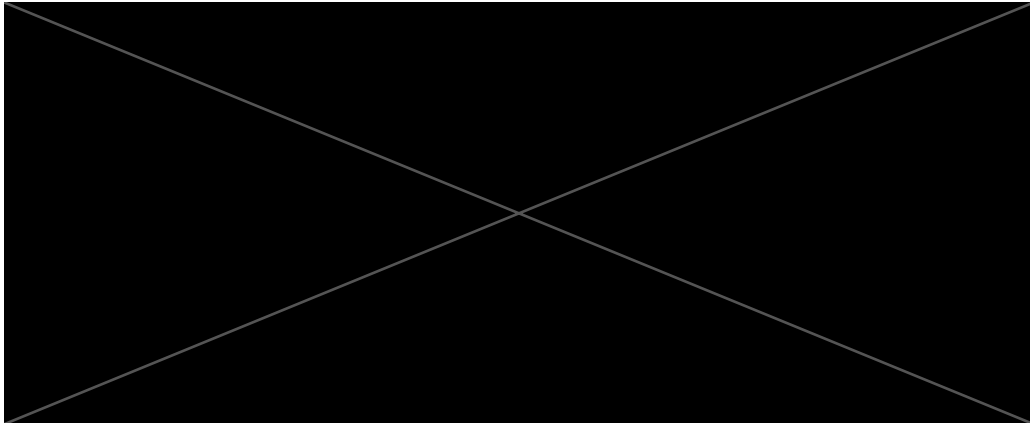
Es folgt ein Proof of Concept zur Veranschaulichung der Schwachstelle, ein Dokument mit allen betroffenen Pfaden finden Sie im Anhang unter 

PROOF OF CONCEPT

Host:	Path:
	



Request



Response

HANDLUNGSEMPFEHLUNG:

Die Verwendung von HTTP sollte in Webanwendungen mit hohem Schutzbedarf auf ein Minimum beschränkt werden. Benutzer, die die Webanwendung über HTTP aufrufen, z.B. indem sie nur den Domainnamen in die URL-Leiste ihres Browsers eingeben, sollten direkt auf eine sichere HTTPS-URL umgeleitet werden. Alle HTTPS-Ressourcen sollten einen Strict-Transport-Security-Header bereitstellen, der sicherstellt, dass der Browser für einen bestimmten Zeitraum nur HTTPS verwendet. Die Syntax für diesen Header lautet wie folgt:

Strict-Transport-Security: max-age=<seconds>[; includeSubDomains]

Die Expire Time gibt den Zeitrahmen in Sekunden vor, in welchem der Browser des Benutzers das HTTPS-Protokoll verwenden soll und sollte daher recht hoch gewählt werden (mehrere Monate). Es kann sich außerdem anbieten die zusätzliche Flag „*includeSubDomains*“ zu verwenden, um diese Einstellungen für alle Subdomains der Webseite zu übernehmen. Bis auf die initiale Umleitung sollte die Anwendung komplett mit HTTPS genutzt werden.

REFERENZEN

- [CWE-693: Protection Mechanism Failure](#)
- [HTTP Strict Transport Security](#)

[CWE-79]: Improper Neutralization of Input During Web Page Generation (Browser Cross-Site Scripting Filter Misconfiguration)

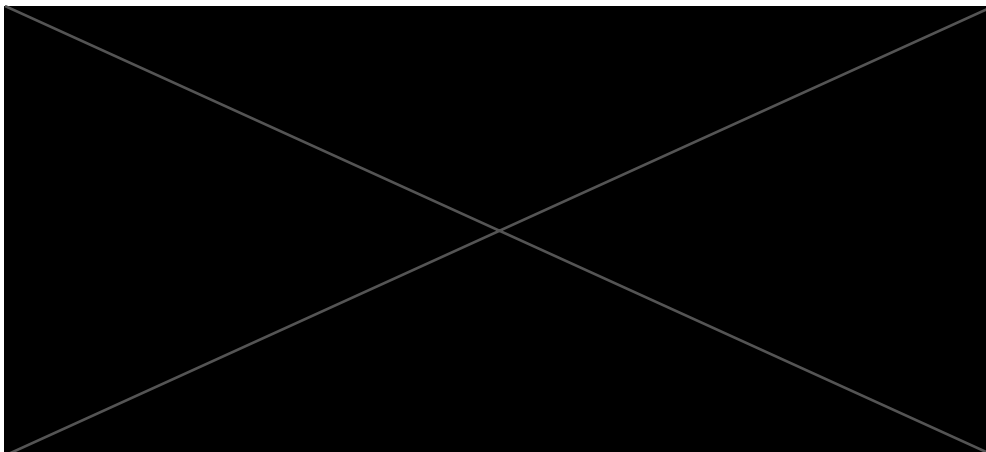
Schweregrad der Schwachstelle: [Information](#)

INHÄRENTE RISIKEN:

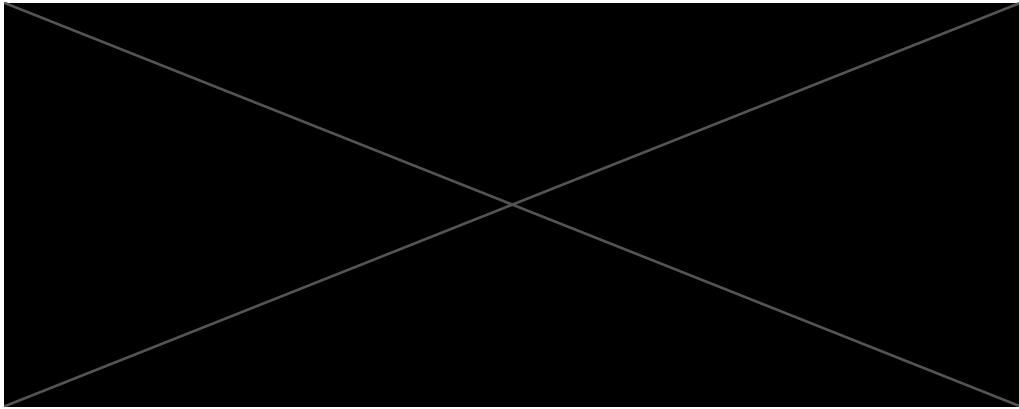
Cross-Site-Scripting-Filter (XSS) in Browsern prüfen, ob die URL mögliche XSS-Payloads enthält und ob diese auf der Response-Seite gespiegelt werden. Wird eine solche Bedingung erkannt, wird der injizierte Code derart verändert, dass er nicht mehr ausgeführt werden kann, um einen erfolgreichen XSS-Angriff zu verhindern. Der Nachteil dieser Filter ist, dass der Browser keine Möglichkeit hat, zwischen Codefragmenten, die von einer anfälligen Webanwendung bei einem XSS-Angriff reflektiert wurden und solchen, die bereits auf der Seite vorhanden sind, zu unterscheiden. In der Vergangenheit wurden diese Filter von Angreifern verwendet, um JavaScript-Code auf der angegriffenen Webseite zu deaktivieren. Manchmal sind die XSS-Filter selbst in gewisser Weise verwundbar, sodass Webanwendungen, die ordnungsgemäß gegen XSS-Angriffe geschützt waren, unter bestimmten Bedingungen anfällig wurden.

Es folgt ein Proof of Concept zur Veranschaulichung der Schwachstelle, ein Dokument mit allen betroffenen Pfaden finden Sie im Anhang unter [<XSS-Misconfig.pdf>](#)

PROOF OF CONCEPT



Request



Response

HANDLUNGSEMPFEHLUNG:

Es wird empfohlen, den XSS-Filter des Browsers anzuweisen, die Webseite niemals zu laden, wenn ein XSS-Angriff erkannt wird. In den meisten Fällen können Sie den von XSS-Filtern bereitgestellten Schutz nutzen, indem Sie den folgenden Antwortheader implementieren:

X-XSS-Protection: 1; mode=block

Wenn dieser Header gesetzt ist, rendern Browser, die einen XSS-Angriff erkennen lediglich eine leere Seite, anstatt zu versuchen, das injizierte Skript zu auswerten. Dieses Verhalten führt wesentlich seltener zu neuen Sicherheitsproblemen und sollte daher und aufgrund der einfachen Umsetzung implementiert werden.

REFERENZEN

- [CWE-79: Improper Neutralization of Input During Web Page Generation \('Cross-site Scripting'\)](#)
- [Browser Cross-Site Scripting Filter Disabled](#)

[CWE-693]: Protection Mechanism Failure (Content Sniffing Not Disabled)

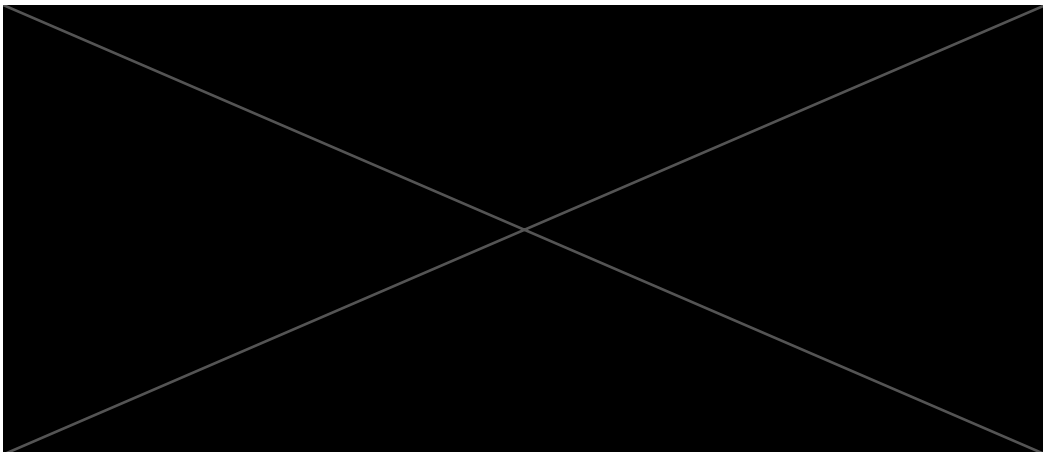
Schweregrad der Schwachstelle: [Information](#)

INHÄRENTE RISIKEN:

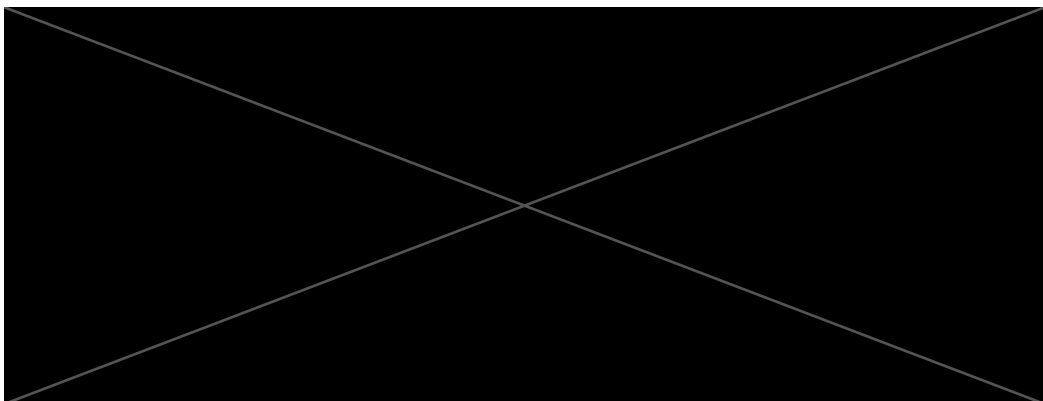
Es gab keinen HTTP-Header "X-Content-Type-Options", in dem der Wert *nosniff* in der Antwort festgelegt war. Das Fehlen dieses Headers führt dazu, dass bestimmte Browser versuchen, den Inhaltstyp und die Codierung der Antwort zu bestimmen, selbst wenn diese Eigenschaften korrekt definiert sind. Dies kann die Webanwendung anfällig für Cross-Site Scripting (XSS)-Angriffe machen. Beispielsweise behandeln der Internet Explorer und Safari Antworten mit dem Inhaltstyp `text/plain` als HTML, wenn sie HTML-Tags enthalten.

Es folgt ein Proof of Concept zur Veranschaulichung der Schwachstelle, ein Dokument mit allen betroffenen Pfaden finden Sie im Anhang unter [<Content_Sniffing.pdf>](#)

PROOF OF CONCEPT



Request



Response

HANDLUNGSEMPFEHLUNG:

Folgender HTTP-Header sollte in allen Antworten gesetzt werden, welche Nutzerangaben enthalten:

X-Content-Type-Options: nosniff

REFERENZEN

- [CWE-693: Protection Mechanism Failure](#)
- [Content Sniffing Not Disabled](#)

Anhang

Anhang 1. OpenSSH-Enumeration.pdf

Anhang 2. SSH_Enum_Exploit.txt

Anhang 3. 

Anhang 4. Fuzzing_Full.pdf

Anhang 5. 

Anhang 6. 

Anhang 7. 

Anhang 8. 

Anhang 9. Extrahierbare_JS_JSON-Files.pdf

Anhang 10. STS-Misconfig.pdf

Anhang 11. XSS-Misconfig.pdf

Anhang 12. Content_Sniffing.pdf

Verantwortlichkeiten

Dieses Dokument enthält vertrauliche Informationen, die ausschließlich für den dedizierten Kunden bestimmt sind. Jede Form der Veröffentlichung an Dritte ist ohne ausdrückliche Genehmigung von KALWEIT ITS nicht gestattet. Dieses Dokument stellt keine Verpflichtung von KALWEIT ITS gegenüber dem Auftraggeber dar und dient ausschließlich der Information und Berichterstattung über bereits erbrachte Leistungen. Insbesondere ist KALWEIT ITS ohne zusätzliche Vereinbarung nicht verpflichtet, festgestellte Schwachstellen zu beseitigen oder diesen Bericht aufgrund von Vorfällen, die nach der Erstellung dieses Berichts eingetreten sind, zu ändern. Jede Art von Dienstleistung, die KALWEIT ITS für den Kunden erbringt, bedarf einer zusätzlichen schriftlichen Vereinbarung. Wir wissen, dass die in diesem Dokument gemachten Aussagen und Empfehlungen den Grundsätzen der Best Practice entsprechen, übernehmen jedoch keine Gewähr für die Vollständigkeit, Richtigkeit und Zuverlässigkeit der Angaben. Die in diesem Bericht formulierten Ergebnisse beruhen auf dem oben beschriebenen Prinzip.

Über uns - Kalweit ITS

KALWEIT ITS steht für die exzellente Verknüpfung von Cybersecurity und Strategie. Als Beratungsmanufaktur für IT-Sicherheit helfen wir unseren Kunden dabei, IT-Risiken und damit verbundene Kosten zu senken: Durch effektive IT-Sicherheit. In interdisziplinären Teams challengen wir durch holistische Sicherheitsüberprüfungen den Status-Quo unserer Kunden und schaffen Sicherheiten im Rahmen der Unternehmensberatung dort, wo auch wirkliche IT-Risiken sind. Ebenso, dass sie auch von der Unternehmenskultur angenommen werden.

Neue Perspektiven, mutige Ideen, pragmatisch umgesetzt.
KALWEIT ITS – **Dafür stehen wir.**

Kalweit ITS GmbH
Esplanade 40
20354 Hamburg



+49 40 285 301 250



hello@kalwe.it
<https://kalweit-its.de>



Ein Mitgliedsunternehmen von
Hamburg@work
Das Netzwerk für eine neue Welt
DigitalCluster.Hamburg

Teilnehmer der
Allianz für
Cyber-Sicherheit 



 **KALWEIT ITS**