

# Penetration Test - Example Report

19.08.2021 — 26.08.2021

**Target Company**  
Some Address 1  
12345 City

Author

**Roman Hergenreder**  
roman.hergenreder@ilume.de

On behalf of

ILUM:E INFORMATIK AG  
tel. +49 6131-25 00 6-0  
fax +49 6131-25 00 6-29  
info@ilume.de

## Document History

Date	Author	Comment
19. Aug 2021	Roman Hergenreder	Initial Version
23. Aug 2021	Vasily Gibin	Approved

# Contents

<b>1</b>	<b>Management Summary</b>	<b>2</b>
<b>2</b>	<b>Introduction</b>	<b>3</b>
2.1	Abbreviations . . . . .	3
2.2	Glossary . . . . .	5
2.3	Motivation . . . . .	6
2.4	Methodology . . . . .	6
2.4.1	OWASP Top 10 . . . . .	6
2.4.2	Used Tools . . . . .	8
<b>3</b>	<b>Overview</b>	<b>9</b>
3.1	Structure . . . . .	9
3.2	Results . . . . .	9
3.3	Severity . . . . .	10
3.3.1	CVSS 0.1 - 3.9: <b>Low</b> . . . . .	10
3.3.2	CVSS 4.0 - 6.9: <b>Medium</b> . . . . .	10
3.3.3	CVSS 7.0 - 8.9: <b>High</b> . . . . .	10
3.3.4	CVSS 9.0 - 10.0: <b>Critical</b> . . . . .	10
<b>4</b>	<b>Target: some-domain.com</b>	<b>12</b>
4.1	Ports and Services . . . . .	12
4.2	Unauthenticated SQL-Injection in /product.php . . . . .	13
4.3	Unnecessary Open Port: 3306 . . . . .	15
4.4	Reflected Version in HTTP Header . . . . .	16
4.5	Privilege Escalation in cron backup script . . . . .	17
<b>5</b>	<b>Target: 192.168.178.123</b>	<b>18</b>
5.1	Ports and Services . . . . .	18
5.2	Vulnerabilities . . . . .	18
<b>6</b>	<b>Final Words</b>	<b>19</b>

## Management Summary

illum:e informatik ag (hereinafter referred to as "we") were tasked with performing a penetration test towards *Target Company*. The agreed scope includes a server running a web application (IP-Address: 1.2.3.4) and a windows client machine (Hostname: client.corp) used by the Office. During the penetration test, we have found multiple severe vulnerabilities on the web server allowing an attacker to completely compromise the server and potentially steal private data, cause a denial of service or harm the system in any other way.

However for the windows client computer no vulnerabilities were found.

# Introduction

## 2.1 Abbreviations

Short	Name	Definition
<b>HTTP</b>	Hypertext Transfer Protocol	Protocol for requesting and transferring files and other data, commonly used by browsers and APIs
<b>HTTPS</b>	Hypertext Transfer Protocol Secure	Same as HTTP, but additionally end-to-end encryption is used
<b>HTML</b>	Hypertext Markup Language	Type of code to structure elements displayed in browsers.
<b>CVE</b>	Common Vulnerabilities & Exposures	Reference-method for publicly known vulnerabilities and exposures
<b>CVSS</b>	Common Vulnerability Scoring System	Scoring system for CVEs to categorize and assess vulnerabilities
<b>API</b>	Application Programming Interface	Interface provided by an application to communicate with other applications (e.g. browser and website)
<b>DNS</b>	Domain Name System	Protocol to resolve names, addresses and other information
<b>XSS</b>	Cross-Site Scripting	An attacker can inject html and javascript code and execute code on the client side
<b>SSTI</b>	Serverside Template Injection	Allows an attacker to inject code which is evaluated by template engines and possibly grants remote code execution.
<b>XXE</b>	eXternal Entity injection	An attacker can send crafted XML documents to an application which can leak secret information or lead to remote code execution

**SQLi**

SQL-Injection

An attacker can abuse the sql database to execute manipulated sql queries which can lead to secret information leakage and remote code execution

**PrivEsc**

Privilege Escalation

A vulnerability in which an attacker can escalate their privilege to either another user on the same level (horizontal privilege escalation) or to an user with higher privileges.

## 2.2 Glossary

Name	Definition
<b>Name resolution</b>	Using the Domain Name System (DNS) mainly IP-Addresses of domain names or for the reverse case domain names assigned to IP-Addresses are resolved. Furthermore, additional data, such as mail and service configurations or domain identifiers (for Google, letsencrypt, etc.) can be exchanged.
<b>Red Team</b>	Group that plays the role of an enemy or competitor, and provides security feedback from that perspective. See also: Blue Team, Purple Team

## 2.3 Motivation

As the digitization is growing rapidly, more and more IT systems are getting targeted by hackers and other criminals. However, most of the attacks are not detected fast enough, and some of them are not detected at all. The average time between an security incident and it's detection is more than 200 days. In this time, all the customer data and company secrets are leaked, internal networks are infiltrated and there is great financial damage.

So that none of this happens, a security audit is done, in the best case repeatedly. We as a "Red Team" take one approach: We put ourselves in the role of an attacker and offensively penetrate the target network to find security flaws before an attacker does.

## 2.4 Methodology

We utilized a widely adopted approach based on the "Information Systems Security Assessment Framework" [1] to performing penetration testing that is effective in testing how well the systems of *Target Company* are secure. According to the framework the penetration test is separated into three phases: Planning & Preparation, Assessment and Reporting & Clean-up. For the assessment phase we proceeded the following steps:

1. *Information Gathering & Network Mapping:*  
We collected information about the target systems using various scanning tools and publicly available sites, as well as identified possible vulnerabilities.
2. *Penetration:*  
Using the gathered information we performed several attack scenarios and tried to exploit the running applications.
3. *Gaining Access & Privilege Escalation:*  
Once we got successfully gained access we tried to escalate internal privileges to test the infrastructure's security.
4. *Enumerating Further:*  
Possible vulnerabilities which are only accessible from the inside were detected here.
5. *Covering Tracks:*  
After the penetration test process, we eliminate all signs of compromise including temporarily created accounts and back doors.

### 2.4.1 OWASP Top 10

When it comes to testing web applications we mainly focus on vulnerabilities which are specified in the "OWASP Top 10" [7]. It describes a standard of the ten most important vulnerabilities specifically in web applications and it is updated frequently. In the version published in 2017 the following vulnerabilities are described:

1. *Injection:*  
Occurs when untrusted data is passed to an interpreter as part of a command or query.
2. *Broken Authentication:*  
Allows an attacker to bypass functions related to authentication or session management.
3. *Sensitive Data Exposure:*  
Unprotected data can be easily accessed without any or insufficient permission checks.
4. *XML External Entities (XXE):*  
A kind of injection where poorly configured XML-processors evaluate external references such as files or code.
5. *Broken Access Control:*  
Restrictions related to access control are broken and can be exploited or bypassed to access data unpredictably.
6. *Security Misconfiguration:*  
Insufficient security configurations can be used in combination with other flaws to leverage access or steal private data.
7. *Cross-Site Scripting (XSS):*  
Untrusted data is included in the Hypertext Markup Language document without sanitizing and can lead to session hijacking or code execution on the client side.
8. *Insecure Deserialization:*  
Serialized untrusted data is passed to an internal deserializer and can lead to code execution.
9. *Using Components with Known Vulnerabilities:*  
Software which is not kept up-to-date can often contain publicly known security vulnerabilities.
10. *Insufficient Logging & Monitoring:*  
Good Logging and Monitoring is important to mitigate attacks quickly. However, most breach studies show time to detect a security incident is over 200 days.



## 2.4.2 Used Tools

During the security assessment the following tools have been used:

- smbmap
- Burp Suite
- impacket
- metasploit
- Internal Security Tools

# Overview

## 3.1 Structure

In the following section, a high level overview of the penetration test results and their meaning are shown and explained. After that each analyzed target is described in a separate chapter including a list of running services, vulnerabilities identified and possible mitigations.

## 3.2 Results

During the penetration test, we have found a total of **4 vulnerabilities** on **2 targets** with an average CVSS-Score of **6.0**. On figure 3.1 you can see the distribution of the vulnerabilities severity.

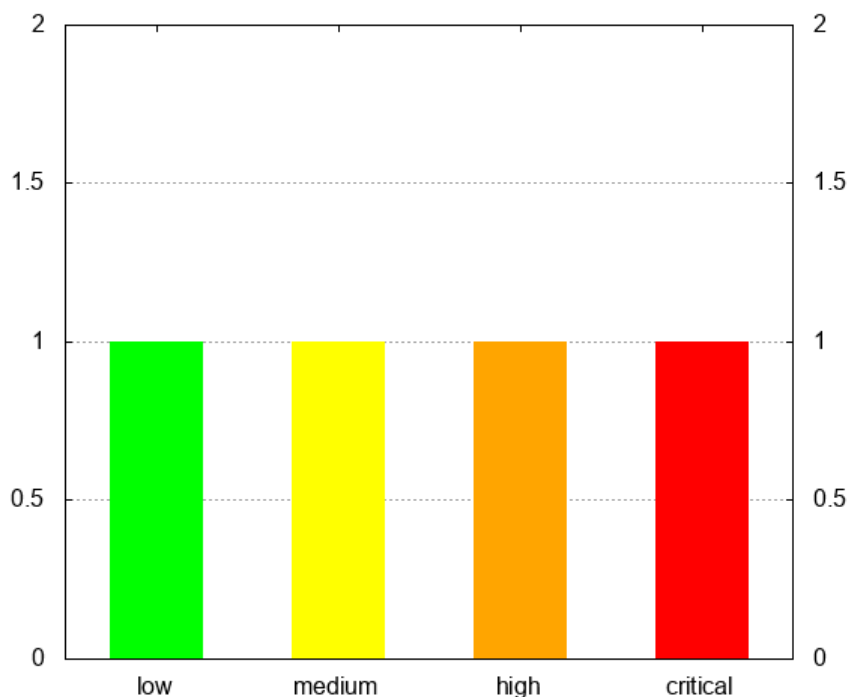


Figure 3.1: Vulnerability Distribution

The following sections describe, in which way vulnerabilities are categorized and which effect they could have when exploited. Additionally, we refer to the CVSS [4] to classify the severity of found vulnerabilities according a standardized scoring system.

## 3.3 Severity

The final score relies on different properties such as the exploitability, the availability, required privileges and user interactions, impact and more. Existing vulnerabilities (so called CVEs), especially for common software like operating systems and web servers, can often be found in public databases [5, 3, 2, 6]. This allows us to quickly check, whether someone reported a security issue in used software before and therefore report it quickly. However, sometimes attackers might use vulnerabilities, which have not been reported yet. These are usually referred to as “Zero-Day exploits“ or just “0-Days”, as 0 days have passed since publicly reported.

Severity	Score
Low	0.1 - 3.9
Medium	4.0 - 6.9
High	7.0 - 8.9
Critical	9.0 - 10.0

Table 3.1: CVSS Scores

### 3.3.1 CVSS 0.1 - 3.9: Low

Vulnerabilities marked as “low” usually do not have a direct attack vector but can be used to gain information about a target system and might be used in combination with other vulnerabilities to successfully take over a system. Those vulnerabilities are usually configurations which can be hardened, such as “sensitive loggings“, “debug modes” or insufficient security settings for cookies, Hypertext Transfer Protocol Secure servers and more. For this category it is not absolutely necessary to take actions.

### 3.3.2 CVSS 4.0 - 6.9: Medium

“Medium” vulnerabilities often allow attackers to perform actions they are usually not intended to perform. Such actions can lead to unexpected behavior and also grants them the ability to take further actions. These actions are usually called “footholds”, as they are the first step to gain control over the whole system. Such vulnerabilities are often security bypasses or possibilities to get information about the system behind including code access and access to internally used software. This category usually requires some mitigations.

### 3.3.3 CVSS 7.0 - 8.9: High

Vulnerabilities with a CVSS score rated “high” allows attackers to exploit software in a way that they can execute code, exfiltrate private data and possibly harm the complete environment. However, the attack vector is difficult or additional privileges are required to exploit the system. Nevertheless, such vulnerabilities necessarily require actions to be taken to mitigate the issues.

### 3.3.4 CVSS 9.0 - 10.0: Critical

“Critical” vulnerabilities require immediate actions as the system is at high risk to be exploited. Such exploits usually do not require complex attack vectors or special privileges to be executed.

Often vulnerabilities in this category are publicly known which additionally allows attackers, even with little knowledge, to cause great damage.

# Target: some-domain.com

This chapter includes the full report for the target specified in the following table.

Primary Address	1.2.3.4
Additional Addresses	5.6.7.8,2001:db8:3333:4444:cccc:dddd:eeee:ffff
Domain Names	some-domain.com,additional-domain.net
Operating System	x86_64 GNU/Linux, Ubuntu 20.04.4 LTS

## 4.1 Ports and Services

During the penetration test the following open ports and their corresponding services were identified:

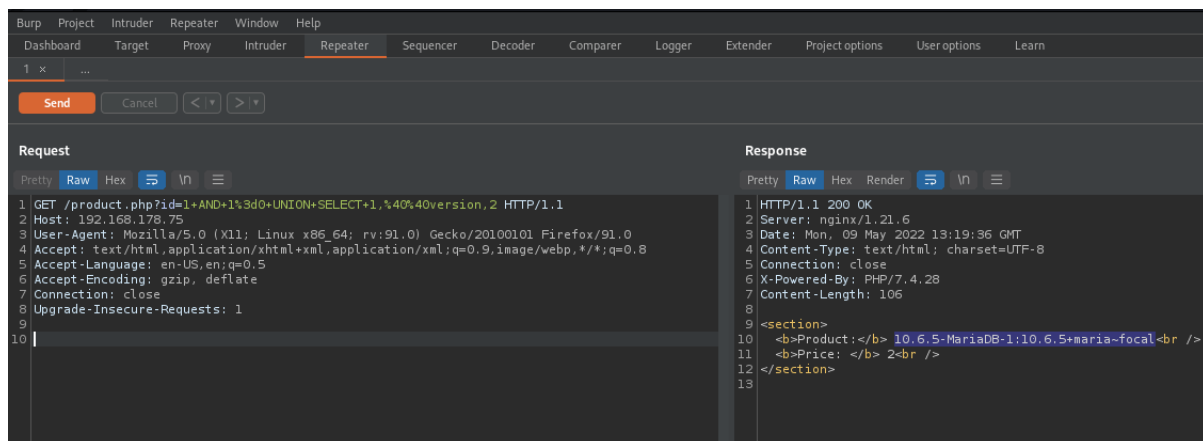
Protocol	Port	Identified Service
TCP	22	OpenSSH 8.2p1 Ubuntu 4ubuntu0.4 (Ubuntu Linux; protocol 2.0)
UDP	53	NLnet Labs NSD
TCP	443	Apache httpd
TCP	3306	MySQL 5.5.5-10.6.5-MariaDB-1:10.6.5+maria~focal

## 4.2 Unauthenticated SQL-Injection in /product.php

<b>Severity:</b> 9.4 Critical	<b>Status:</b> open
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:L	
<b>SQL-Injection (SQLi):</b> An attacker can abuse the sql database to execute manipulated sql queries which can lead to secret information leakage and remote code execution	
<b>Arbitrary File Read:</b> An attacker can access any file on the disk due to path traversal or local file inclusion.	

### Description

The route `/product.php` served by the apache web server to display product information by a given ID is vulnerable to an unauthenticated SQL-Injection, which allows attackers to access the entire database as well as files readable by the database service. This includes sensitive files such as configuration files with passwords or log files.



```
1 GET /product.php?id=1+AND+1%3d0+UNION+SELECT+1,%40%40version,2 HTTP/1.1
2 Host: 192.168.178.75
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Upgrade-Insecure-Requests: 1
9
10

1 HTTP/1.1 200 OK
2 Server: nginx/1.21.6
3 Date: Mon, 09 May 2022 13:19:36 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: close
6 X-Powered-By: PHP/7.4.28
7 Content-Length: 106
8
9 <section>
10 <b>Product:</b> 10.6.5-MariaDB-1:10.6.5+maria-focal<br />
11 <b>Price:</b> 2<br />
12 </section>
13
```

Figure 4.1: Reflected\_SQL\_Injection.png

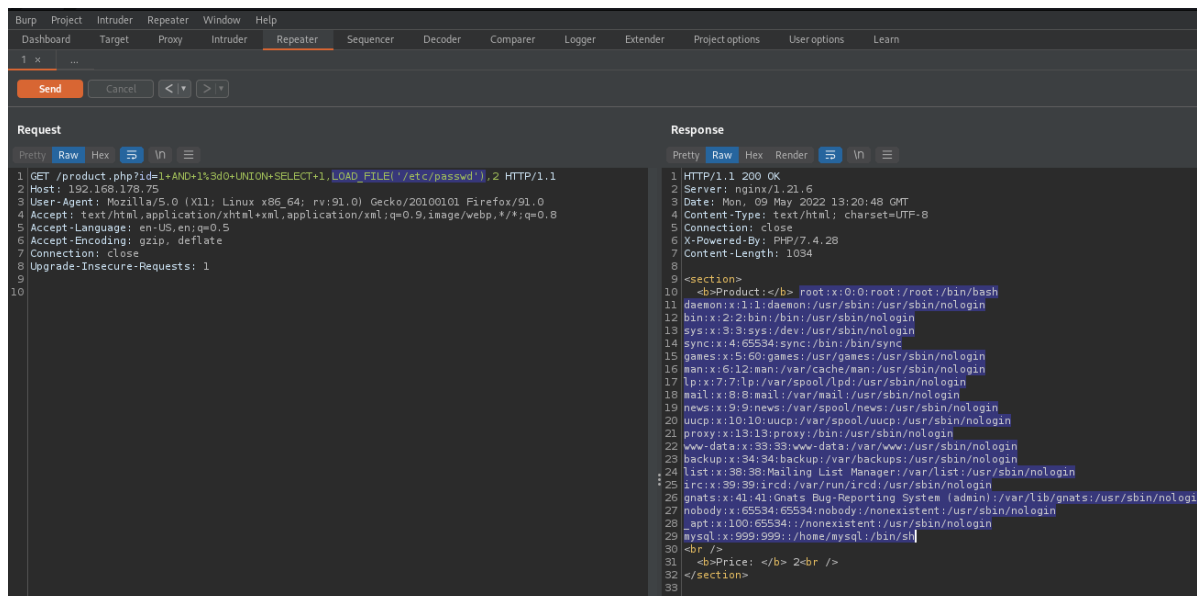


Figure 4.2: SQL\_Injection\_File\_Read.png

## Advices

Sanitize all untrusted input coming from the user. In this case, the GET parameter 'id' has to be sanitized or checked prior to passing to the database. Additionally it is highly recommended to use prepared statements, so that the database can distinguish between the query and parameters.

For more information, go to:

- <https://www.php.net/manual/en/mysql.quickstart.prepared-statements.php>
- <https://www.phptutorial.net/php-tutorial/php-sanitize-input/>

It is also recommended to grant the database user as few rights as possible, i.e. prohibit file access if not necessary needed.

## 4.3 Unnecessary Open Port: 3306

<b>Severity:</b> 2.5 Low	<b>Status:</b> open
<b>Open Port:</b> An unnecessarily publicly exposed port which can be blocked by a firewall or other security infrastructure	

### Description

The mysql database service is bound publicly on port 3306. This port can be usually blocked externally inside the firewall as the web applications runs on the same server. It is recommended to access database only locally.



## 4.4 Reflected Version in HTTP Header

<b>Severity:</b> 5.0 Medium	<b>Status:</b> open
<b>Sensitive Information:</b> Publicly accessible sensitive information due to weak permission configuration or sensitive application logging	

### Description

The apache web service returns an unnecessary http header indicating the PHP version used.

## 4.5 Privilege Escalation in cron backup script

<b>Severity:</b> 7.2 High	<b>Status:</b> open
CVSS:3.1/S:C/C:H/I:H/A:H/UI:R/PR:H/AC:H/AV:L	
<b>Weak Credentials:</b> An authentication form relies on weak credentials and can be easily bruteforced or cracked by an attacker	
<b>Service Misconfiguration:</b> An internal misconfigured service can lead to escalation of privileges	
<b>Privilege Escalation (PrivEsc):</b> A vulnerability in which an attacker can escalate their privilege to either another user on the same level (horizontal privilege escalation) or to an user with higher privileges.	

### Description

An insecurely configured cron job was found which periodically creates a backup of the web application directory. Due to the misconfiguration an local attacker can abuse the backup script to access files owned by any user including the root user. In an exemplary attack an attacker can access the `/root/.ssh` folder for private keys or the `/etc/shadow` file to get access to the root's password hash.

## Target: 192.168.178.123

This chapter includes the full report for the target specified in the following table.

Primary Address	192.168.178.123
Additional Addresses	client.corp
Domain Names	
Operating System	Windows 10

### 5.1 Ports and Services

During the penetration test the following open ports and their corresponding services were identified:

Protocol	Port	Identified Service
TCP	445	SMB

### 5.2 Vulnerabilities

For this target no vulnerabilities were found.

## Final Words

After the completion of the penetration test, we have cleaned up all temporary files, backdoors and users created during and used during the test. Another penetration test will be scheduled to verify that all found vulnerabilities are fully remediated and no other issues are found.

## References

- [1] Open Information Security Services Group (OISSG). "Information Systems Security Assessment Framework (ISSAF)". In: URL: <https://untrustednetwork.net/files/issaf0.2.1.pdf>.
- [2] CVE Details. "CVE security vulnerability database. Security vulnerabilities, exploits, references and more". In: URL: <https://www.cvedetails.com/>.
- [3] CVE mitre. "Search CVE List". In: URL: [https://cve.mitre.org/cve/search\\_cve\\_list.html](https://cve.mitre.org/cve/search_cve_list.html).
- [4] National Institute of Standards and Technology. "Common Vulnerability Scoring System". In: URL: <https://nvd.nist.gov/vuln-metrics/cvss>.
- [5] National Institute of Standards and Technology. "National Vulnerability Database". In: URL: <https://nvd.nist.gov/vuln/search>.
- [6] Offensive Security. "Exploit Database - Exploits for Penetration Testers, Researchers, and Ethical Hackers". In: URL: <https://www.exploit-db.com/>.
- [7] Open Web Application Security Project. "OWASP Top 10". In: URL: <https://owasp.org/www-project-top-ten/>.