Penetration Test on Example Application for Example Customer AG

Final Report and Management Summary

2025-08-13

PUBLIC

X41 D-SEC GmbH
Soerser Weg 20
D-52070 Aachen
Amtsgericht Aachen: HRB19989

https://x41-dsec.de/
info@x41-dsec.de

Revision	*Date*	*Change*	*Author(s)*
1	2023-12-31	Final Report and Management Summary	Example Employee

Contents

1 Executive Summary **4**

2 Introduction **6**
2.1 Methodology 6
2.2 Findings Overview 7
2.3 Scope 8
2.4 Coverage 8
2.5 Recommended Further Tests 9

3 Rating Methodology **10**
3.1 CVSS 10
3.2 Severity Mapping 13
3.3 Common Weakness Enumeration 13

4 Results **14**
4.1 Network 14
4.2 Findings 16
4.3 Informational Notes 31

5 About X41 D-Sec GmbH **33**

Dashboard

Target	
Customer	Example Customer AG
Name	Networks of Example Customer AG
Type	Network Infrastructure
Version	As deployed between 2017-12-01 and 2017-12-10

Engagement	
Type	Gray Box Penetration Test
Consultants	3: Eric Sesterhenn, Markus Vervier and Niklas Abel
Engagement Effort	10 person-days, 2017-12-01 to 2017-12-10

Total issues found	11

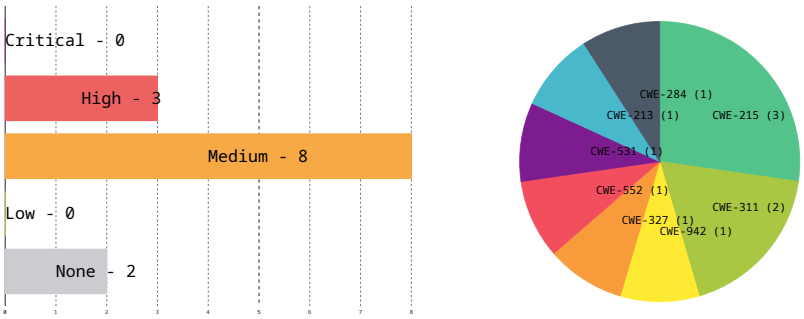

Figure 1: Issue Overview (l: Severity, r: CWE Distribution)

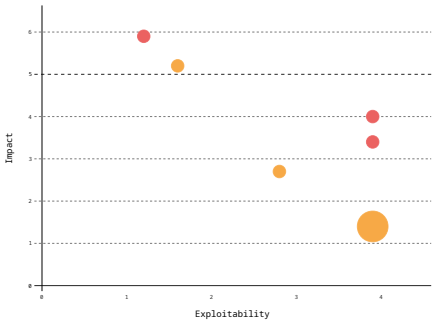

Figure 2: CVSS Impact and Exploitability Distribution

1 Executive Summary

In December 2015 X41 conducted an initial penetration test for Example Customer AG on selected external network perimeters and accessible applications. During this test 14 security relevant issues were discovered. Two issues were rated with a *high* severity, three with *medium* severity, seven with *low* severity and two with a severity of *none*. No issues with critical severity rating were discovered.

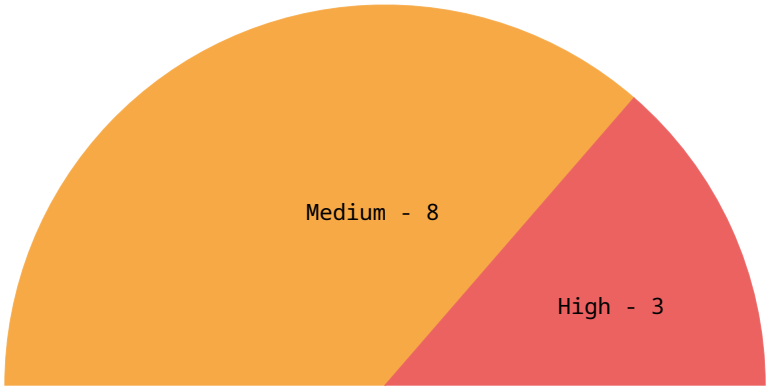

Figure 1.1: Number of Issues

The goal of this review was to estimate the security status of the <anonymized> network perimeters and related application infrastructure. Testers of X41 identified vulnerabilities in the tested systems using real-world hacking methods.

The penetration test was conducted as a gray box test over the Internet on predetermined sys-

tems named by Example Customer AG beforehand. During a gray box test, testers are only given information which they could also gain with an increased time budget by themselves.

The test was carried out by one tester in a total of four person days of active testing.

In comparison the tested infrastructure and the <anonymized> application showed a better security level than most other systems tested of comparable size and purpose.

Especially the Django-based application framework showed good resistance against attacks. Mitigations and design features employed by it were functional and prevented standard vulnerabilities such as XSS[1] quite well.

The issue with highest severity and impact was an exposure of Elastic Search credentials via public debugging messages on the staging system. Other issues rated with high severity were related to external requests for <anonymized> web resources being allowed and weak encryption algorithms for HTTPS[2] connections. Additionally X41 was able to discover non-public pricing information via brute forcing of web addresses. Besides that, other minor issues were found.

As many of the discovered vulnerabilities were related to debugging information and error pages found on the staging system, X41 recommends to lock down the staging system. In general staging system should have identical security settings and content access rules as production systems.

In summary the security level of the <anonymized> application and related infrastructure was good. The amount of issues discovered was low for an initial penetration test. X41 recommends to fix or mitigate the discovered issues and conduct repeated tests in order to keep the security level high.

[1] Cross-site Scripting
[2] HyperText Transfer Protocol Secure

2 Introduction

X41 reviewed network perimeters and related application infrastructure which are involved in <censored>.

They are considered sensitive because <censored>.

Attackers could, for example, try to attack vulnerable, internet-facing services or applications to gain access to the internal company network or the application's sensitive user data.

2.1 Methodology

The review was based on a penetration test against the staging systems a network scan of associated infrastructure.

A manual approach for penetration tests and for code reviews is used by X41. This process is supported by tools such as static code analyzers and industry standard web application security tools[1].

X41 adheres to established standards for source code reviewing and penetration testing. These are in particular the *CERT Secure Coding*[2] standards and the *Study - A Penetration Testing Model*[3] of the German Federal Office for Information Security.

The flow of network scans is shown in figure 2.1. The available systems are explored using automated scanning in the reconnaissance phase, after which possible attacks are considered in the enumeration phase. To validate that an attack is applicable and effective, exploitation is attempted. Each finding is documented directly to not lose track of information.

[1] https://portswigger.net/burp
[2] https://wiki.sei.cmu.edu/confluence/display/seccode/SEI+CERT+Coding+Standards
[3] https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Studies/Penetration/penetration_pdf.pdf?__blob=publicationFile&v=1

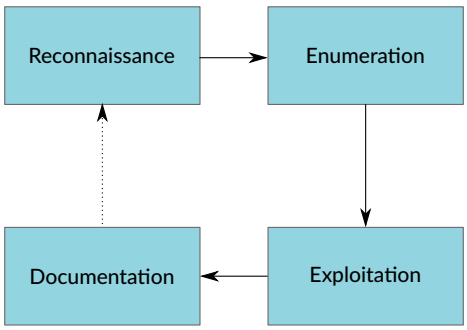

Figure 2.1: Penetration Test Methodology

2.2 Findings Overview

DESCRIPTION	SEVERITY	ID	REF
Credentials Exposed via Error Page	HIGH	SHRTNM-PT-23-01	4.2.1
Wildcard CORS Headers	HIGH	SHRTNM-PT-23-02	4.2.2
Weak SSL / TLS Settings Enabled on TCP Port 443	MEDIUM	SHRTNM-PT-23-03	4.2.3
Insecure JavaScript Resource Loading	MEDIUM	SHRTNM-PT-23-04	4.2.4
Nonpublic Price Information Revealed	MEDIUM	SHRTNM-PT-23-05	4.2.5
Cookies Without the Secure Flag	HIGH	SHRTNM-PT-23-06	4.2.6
Hostname of Development System Leaked	MEDIUM	SHRTNM-PT-23-07	4.2.7
API Documentation Exposed via OPTIONS Request Method	MEDIUM	SHRTNM-PT-23-08	4.2.8
Permissive Route for Blog URLs	MEDIUM	SHRTNM-PT-23-09	4.2.9
Server and OS Version Information Exposed	MEDIUM	SHRTNM-PT-23-10	4.2.10
NTP Service Exposed to the Internet	MEDIUM	SHRTNM-PT-23-11	4.2.11
No or Misconfigured Strict Transport Security	NONE	SHRTNM-PT-23-100	4.3.1
Cookies Without the HttpOnly Flag	NONE	SHRTNM-PT-23-101	4.3.2

Table 2.1: Security-Relevant Findings

2.3 Scope

X41 was provided with the IP ranges of the target systems, as listed below.

- `0.0.0.0/0`

The objective was to scan external and internal IP ranges for exposed hosts and any exposed and potentially vulnerable services.

X41 was further provided with credentials to access the staging systems:

- `username`

Further, a Signal chat group was created for direct communication between the developers and the testers.

2.4 Coverage

A security assessment attempts to find the most important or sometimes as many of the existing problems as possible, though it is practically never possible to rule out the possibility of additional weaknesses being found in the future.

The time allocated to X41 for this assessment was sufficient to yield a good coverage of the given scope.

X41 tested the components for OWASP[4] top 10 vulnerabilities[5] and in addition looked for security best practices not covered by OWASP. The web application hosts and used CDNs[6] were scanned for TLS[7]/SSL[8] configuration issues. After enumerating common attacks, targeted attacks were performed specific to the web application components like authentication issues related to using JWT[9]. JavaScript modules were checked for known security issues.

For example, the web servers were tested for common misconfigurations and debugging and error messages inspected for sensitive data.

Suggestions for next steps in securing this scope can be found in section 2.5.

[4] Open Web Application Security Project
[5] `https://owasp.org/www-project-top-ten/`
[6] Content Delivery Networks
[7] Transport Layer Security
[8] Secure Sockets Layer
[9] JSON Web Token

2.5 Recommended Further Tests

X41 recommends to test the production system as well because there might be subtle differences that have an impact on the security of the application.

3 Rating Methodology

Security vulnerabilities are given a purely technical rating by the testers when they are discovered during a test. Business factors and financial risks for Example Customer AG are beyond the scope of a penetration test, which focuses entirely on technical factors. However, technical results from a penetration test may be an integral part of a general risk assessment. A penetration test is based on a limited time frame and only covers vulnerabilities and security issues which have been found in the given time, there is no claim for full coverage.

The CVSS[1] is used to score all findings relevant to security. The resulting CVSS score is mapped to qualitative ratings as shown below.

3.1 CVSS

All findings relevant to security are rated by the testers using the CVSS industry standard version 3.1, revision 1.

Vulnerabilities scored with CVSS get a numeric value based on several metrics ranging from 0.0 (least worst) to 10.0 (worst).

The score captures different factors that express the impact and the ease of exploitation of a vulnerability among other factors. For a detailed description of how the scores are calculated, please see the CVSS version 3.1 specification.[2]

The metrics used to calculate the final score are grouped into three different categories.

[1] Common Vulnerability Scoring System
[2] `https://www.first.org/cvss/v3-1/cvss-v31-specification_r1.pdf`

The *Base Metric Group* represents the intrinsic and fundamental characteristics of a vulnerability that are constant over time and user environments. It captures the following metrics:

- Attack Vector (AV)
- Attack Complexity (AC)
- Privileges Required (PR)
- User Interaction (UI)
- Scope (S)
- Confidentiality Impact (C)
- Integrity Impact (I)
- Availability Impact (A)

The *Temporal Metric Group* represents the characteristics of a vulnerability that change over time but not among user environments. It captures the following metrics:

- Exploitability (E)
- Remediation Level (RL)
- Report Confidence (RC)

The *Environmental Metric Group* represents the characteristics of a vulnerability that are relevant and unique to a particular user's environment. It includes the following metrics:

- Attack Vector (MAV)
- Attack Complexity (MAC)
- Privileges Required (MPR)
- User Interaction (MUI)
- Confidentiality Requirement (MCR)
- Integrity Requirement (MIR)
- Availability Requirement (MAR)
- Scope (MS)
- Confidentiality Impact (MC)

- Integrity Impact (MI)
- Availability Impact (MA)

A CVSS vector defines a specific set of metrics and their values, and it can be used to reproduce and assess a given score. It is rendered as a string that exactly reproduces a score.

For example, the vector `CVSS:3.1/AV:N/AC:H/PR:L/UI:R/S:C/C:H/I:L/A:N` defines a base score metric with the following parameters:

- Attack Vector: Network
- Attack Complexity: High
- Privileges Required: Low
- User Interaction: Required
- Scope: Changed
- Confidentiality Impact: High
- Integrity Impact: Low
- Availability Impact: None

In this example, a network-based attacker performs a complex attack after gaining access to some privileges, by tricking a user into performing some actions. This allows the attacker to read confidential data and change some parts of that data.

The detailed scores are the following:

Metric	Score
CVSS Base Score	6.5
Impact Sub-Score	4.7
Exploitability Sub-Score	1.3
CVSS Temporal Score	Not Available
CVSS Environmental Score	Not Available
Modified Impact Sub-Score	Not Available
Overall CVSS Score	6.5

CVSS vectors can be automatically parsed to recreate the score, for example, with the CVSS calculator provided by FIRST, the organization behind CVSS: `https://www.first.org/cvss/calculator/3.1#CVSS:3.1/AV:N/AC:H/PR:L/UI:R/S:C/C:H/I:L/A:N`.

3.2 Severity Mapping

To help in understanding the results of a test, numeric CVSS scores are mapped to qualitative ratings as follows:

Severity Rating	CVSS Score
NONE	0.0
LOW	0.1–3.9
MEDIUM	4.0–6.9
HIGH	7.0–8.9
CRITICAL	9.0–10.0

3.3 Common Weakness Enumeration

The CWE[3] is a set of software weaknesses that allows vulnerabilities and weaknesses in software to be categorized. If applicable, X41 gives a CWE ID for each vulnerability that is discovered during a test.

CWE is a very powerful method for categorizing a vulnerability. It gives general descriptions and solution advice on recurring vulnerability types. CWE is developed by *MITRE*.[4] More information can be found on the CWE site at `https://cwe.mitre.org/`.

[3] Common Weakness Enumeration
[4] `https://www.mitre.org`

4 Results

This chapter describes the results of this test. Following general observations including enumerated services and other collected information about the tested systems, the security-relevant findings are documented in Section 4.2. Additionally, findings without a direct security impact are documented in Section 4.3.

4.1 Network

This section describes findings related to network services, network infrastructure and other discoveries related to networking.

4.1.1 Identified Network Services

HOST	PORT	SERVICE	VERSION
<anonymized>	22/tcp	ssh	OpenSSH 6.6.1p1 Ubuntu 2ubuntu2
<anonymized>	80/tcp	http	nginx 1.4.6 (Ubuntu)
<anonymized>	123/udp	ntp	NTP v4 (secondary server)
<anonymized>	443/tcp	ssl/http	nginx 1.4.6 (Ubuntu)
<anonymized>	3022/tcp	ssh	OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.3 (Ubuntu Linux; protocol 2.0)

Table 4.1: Discovered services using network port scanning.

The services displayed in table 4.1 were discovered using version 7.01 of the *NMAP*[1] tool.

[1] `https://nmap.org`

The following command lines were used:

- nmap -sS -v -v -p 0-65535 -A -sC -iL ../targets.txt -oA <anonymized>-nmap
- nmap -sU -T4 -v -v -A -sC -iL ../targets.txt -oA <anonymized>-nmap-udp-fast

The scanning targets were:

- <anonymized>
- <anonymized>
- <anonymized>
- Network range <anonymized>

The scanning source was a public network block on the Internet (0.0.0.0/24).

A full TCP port range was scanned on all tested hosts using the TCP SYN scan[2] scanning method.

A standard UDP scanning was performed on the most common ports due to time constraints and the unreliable nature of UDP[3] port scanning in general.

4.1.2 DNS Names

The DNS[4] information and host names shown in table 4.2 were discovered using metadata, DNS reverse lookups and other sources such as search engines and passive DNS.

Hostname	IP
<anonymized>	<anonymized>
<anonymized>	<anonymized>
<anonymized>	<anonymized>
<anonymized>	<anonymized>

Table 4.2: Discovered DNS information.

[2] https://nmap.org/book/man-port-scanning-techniques.html
[3] User Datagram Protocol
[4] Domain Name System

4.2 Findings

Security relevant findings are documented in the following.

4.2.1 SHRTNM-PT-23-01: Credentials Exposed via Error Page

Severity:	HIGH
CVSS v3.1 Total Score:	8.6
CVSS v3.1 Vector:	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:N/A:N
CWE:	215 – Information Exposure Through Debug Information
Affected Component:	

4.2.1.1 Description

It was discovered that configuration data leaks via an error page. This information includes credentials for *Elastic Search*, local variable values and other sensitive data.

When accessing the URL `<anonymized>`, an error page like the one shown in figure 4.1 is displayed. This behavior is also present on other pages triggering an exception.

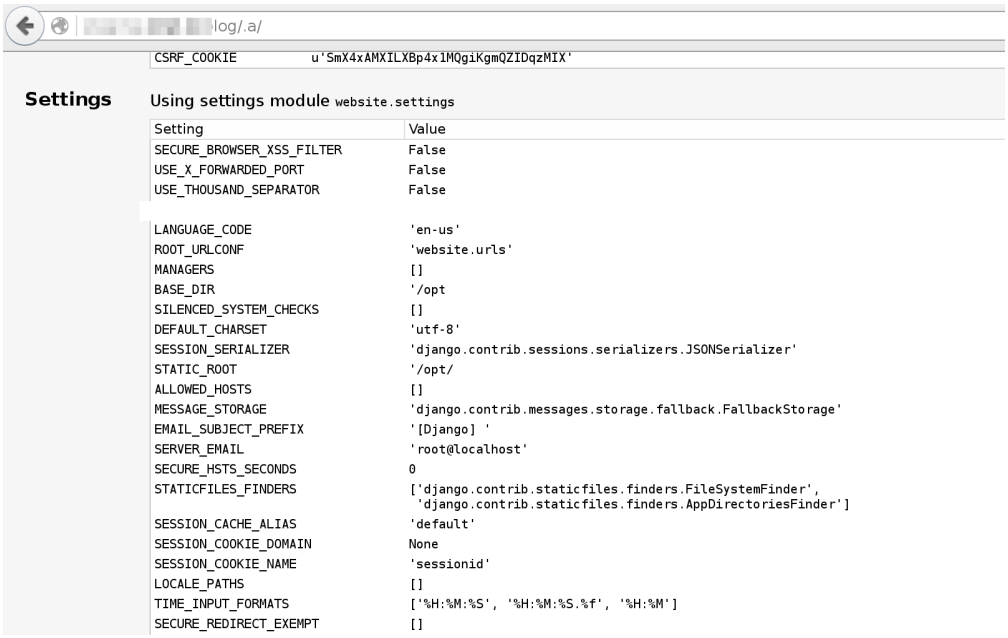

Figure 4.1: Error Page Exposing Sensitive Data

Confidentiality, integrity and availability of data belonging to the exposed Elastic Search account might be affected.

4.2.1.2 Solution Advice

X41 recommends to disable debugging for all externally accessible systems to prevent leakage of sensitive data. We further recommend to configure the staging system with the same security mitigations and settings as the production system.

4.2.2 SHRTNM-PT-23-02: Wildcard CORS Headers

Severity:	HIGH
CVSS v3.1 Total Score:	7.3
CVSS v3.1 Vector:	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:L
CWE:	942 – Overly Permissive Cross-domain Whitelist
Affected Component:	

4.2.2.1 Description

It was discovered that the configured wildcard CORS[5] headers allow external sites to access content and possibly sensitive information on `<anonymized>` and `<anonymized>`.

CORS allows requests to resources originating from external domains. It defines exceptions to the same origin policy which normally denies external domains to access local resources. This is a security feature to protect secret and sensitive information from unauthorized access.

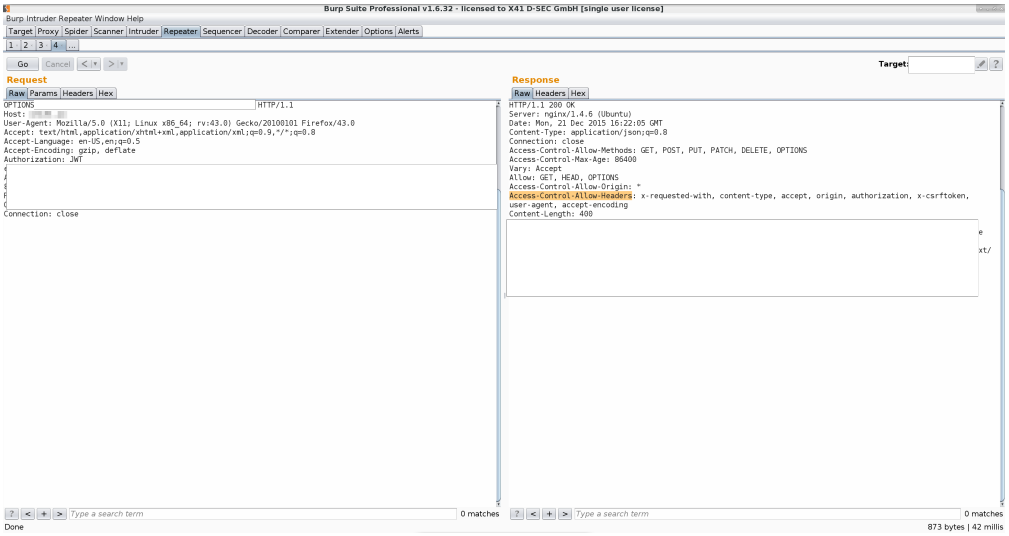

Figure 4.2: Wildcard CORS Header

As shown in figure 4.2, the HTTP header *`Access-Allow-Origin`* was set to the wildcard value `*`. This allows, for example, JavaScript code on external sites to make requests and retrieve responses to resources hosted on `<anonymized>`.

An attacker might use this to retrieve sensitive content or other information such as session cookies or CSRF tokens.

[5] Cross-Origin Resource Sharing

4.2.2.2 Solution Advice

X41 recommends to only allow requests from external sites. Wildcard headers should be avoided. Instead, X41 recommends to collect a whitelist of allowed and trusted domains and set CORS headers for them individually.

4.2.3 SHRTNM-PT-23-03: Weak SSL / TLS Settings Enabled on TCP Port 443

Severity:	MEDIUM
CVSS v3.1 Total Score:	6.8
CVSS v3.1 Vector:	CVSS:3.0/AV:A/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N
CWE:	327 – Use of a Broken or Risky Cryptographic Algorithm
Affected Component:	

4.2.3.1 Description

Weak SSL / TLS settings were discovered on <anonymized> port 443 which would allow an attacker to break the encryption or conduct MITM[6] attacks on <anonymized> customers.

```
1     Accepted  SSLv3  256 bits  ECDHE-RSA-AES256-SHA
2     Accepted  SSLv3  256 bits  DHE-RSA-AES256-SHA
3     Accepted  SSLv3  256 bits  DHE-RSA-CAMELLIA256-SHA
4     Accepted  SSLv3  256 bits  AECDH-AES256-SHA
5     Accepted  SSLv3  256 bits  AES256-SHA
6     Accepted  SSLv3  256 bits  CAMELLIA256-SHA
7     Accepted  SSLv3  128 bits  ECDHE-RSA-AES128-SHA
8     Accepted  SSLv3  128 bits  DHE-RSA-AES128-SHA
9     Accepted  SSLv3  128 bits  DHE-RSA-SEED-SHA
10    Accepted  SSLv3  128 bits  DHE-RSA-CAMELLIA128-SHA
11    Accepted  SSLv3  128 bits  AECDH-AES128-SHA
12    Accepted  SSLv3  128 bits  AES128-SHA
13    Accepted  SSLv3  128 bits  SEED-SHA
14    Accepted  SSLv3  128 bits  CAMELLIA128-SHA
15    Accepted  SSLv3  128 bits  ECDHE-RSA-RC4-SHA
16    Accepted  SSLv3  128 bits  AECDH-RC4-SHA
17    Accepted  SSLv3  128 bits  RC4-SHA
18    Accepted  SSLv3  128 bits  RC4-MD5
19    Accepted  SSLv3  112 bits  ECDHE-RSA-DES-CBC3-SHA
20    Accepted  SSLv3  112 bits  EDH-RSA-DES-CBC3-SHA
21    Accepted  SSLv3  112 bits  AECDH-DES-CBC3-SHA
22    Accepted  SSLv3  112 bits  DES-CBC3-SHA
```

Listing 4.1: SSL Ciphers

As shown in listing 4.1, export and anonymous cipher suites are enabled on TCP[7] port 443 as

[6] Machine-in-the-middle Attack
[7] Transmission Control Protocol

well as SSL protocol version 3. Furthermore, the weak DES[8] encryption algorithm is accepted by the server.

These cipher suites are considered insecure. Some of them (such as DES-based cipher suites) could even be broken by amateurs with a limited budget.

Anonymous cipher suites provide no checking of integrity and trust. This means that despite the use of encryption an attacker could still conduct a MITM attack.

The SSL protocol version 3 is affected by an attack called *POODLE*[9] which may allow an active MITM to decrypt content transferred via an SSL protocol version 3 connection.

4.2.3.2 Solution Advice

X41 recommends to follow security best practices and to only use cipher suites and algorithms considered strong and secure. For the nginx web server the default settings are sufficient. X41 further recommends to check the current configuration against the recommended one from nginx found at: `https://nginx.org/en/docs/http/configuring_https_servers.html`

[8] Data Encryption Standard
[9] `https://poodle.io/`

4.2.4 SHRTNM-PT-23-04: Insecure JavaScript Resource Loading

Severity:	MEDIUM
CVSS v3.1 Total Score:	6.1
CVSS v3.1 Vector:	CVSS:3.0/AV:A/AC:L/PR:N/UI:N/S:C/C:L/I:L/A:N
CWE:	311 – Missing Encryption of Sensitive Data
Affected Component:	

4.2.4.1 Description

Arbitrary JavaScript code execution via insecure loading of resources and their evaluation could allow an attacker to execute script code in the context of `<anonymized>`.

Figure 4.3: Insecure JavaScript Resource

A script resource on the URL[10] `<anonymized>` is loaded via unencrypted HTTP[11] connections as shown in figure 4.3. An attacker who is able to launch a MITM on a network might inject their own script content. It would then be executed in the context of the <anonymized> website.

4.2.4.2 Solution Advice

X41 recommends to only load scripts from trusted sources and via HTTPS[12] connections.

[10] Uniform Resource Locator
[11] HyperText Transfer Protocol
[12] HyperText Transfer Protocol Secure

4.2.5 SHRTNM-PT-23-05: Nonpublic Price Information Revealed

Severity:	MEDIUM
CVSS v3.1 Total Score:	5.3
CVSS v3.1 Vector:	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N
CWE:	552 – Files or Directories Accessible to External Parties
Affected Component:	

4.2.5.1 Description

Not yet public price information is accessible via the staging system without authentication.

```
1 <anonymized>
```

Listing 4.2: Price Information

As shown in figure 4.2, price information of Example Customer AG is accessible via the URL `<anonymized>` without authentication.

4.2.5.2 Solution Advice

X41 recommends to only allow authorized users access to content that is not ready for disclosure. To this end, authentication mechanisms such as client certificates or HTTP-Basic-Auth could be used.

4.2.6 SHRTNM-PT-23-06: Cookies Without the Secure Flag

Severity:	HIGH
CVSS v3.1 Total Score:	7.1
CVSS v3.1 Vector:	CVSS:3.1/AV:A/AC:H/PR:N/UI:R/S:U/C:H/I:H/A:H
CWE:	311 – Missing Encryption of Sensitive Data
Affected Component:	https://<anonymized>

4.2.6.1 Description

The following pages do not set the secure[13] flag for the cookies provided:

- `https://<anonymized>` - Cookie *example*

The secure flag prevents the browser from sending the cookie via unencrypted connections such as HTTP. This might allow an attacker to get access to these cookies which could get facilitated to observe or modify its content. Further information is available from OWASP[14].

4.2.6.2 Solution Advice

X41 recommends to set the secure flag for all cookie values.

[13] `https://en.wikipedia.org/wiki/Secure_cookies`
[14] `https://www.owasp.org/index.php/SecureFlag`

4.2.7 SHRTNM-PT-23-07: Hostname of Development System Leaked

Severity:	MEDIUM
CVSS v3.1 Total Score:	5.3
CVSS v3.1 Vector:	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N
CWE:	531 – Information Exposure Through Test Code
Affected Component:	

4.2.7.1 Description

It was discovered that hostname and port of the development system leak via a test page at `<anonymized>`.

Figure 4.4: Test Page Source Code

As shown in figure 4.4, an attacker could learn the URL (`<anonymized>`) of a development system from a test page located at `<anonymized>`.

4.2.7.2 Solution Advice

X41 recommends to remove all test scripts and test code from staging and production systems.

4.2.8 SHRTNM-PT-23-08: API Documentation Exposed via OPTIONS Request Method

Severity:	MEDIUM
CVSS v3.1 Total Score:	5.3
CVSS v3.1 Vector:	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N
CWE:	213 – Intentional Information Exposure
Affected Component:	

4.2.8.1 Description

The *OPTIONS* request method shows detailed API[15] documentation.

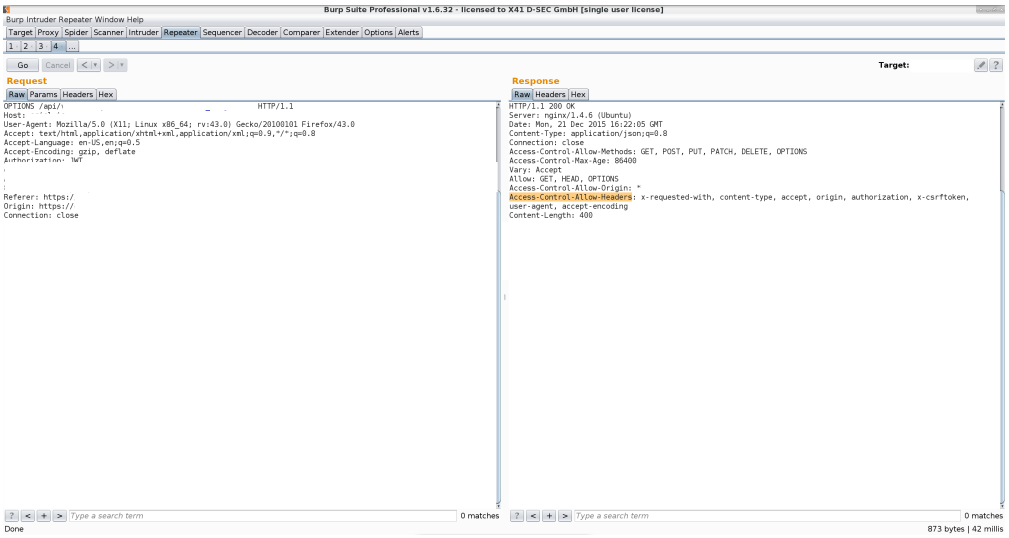

Figure 4.5: Detailed API Documentation via OPTIONS Request

As displayed in figure 4.5, the following request provides an attacker information about the API and exposed attack surface.

```
1 OPTIONS /api/<anonymized> HTTP/1.1
```

Listing 4.3: Request to Options API

[15] Application Programming Interface

4.2.8.2 Solution Advice

X41 recommends to only expose API documentation to trusted parties in order to restrict the information on the API call attack surface.

4.2.9 SHRTNM-PT-23-09: Permissive Route for Blog URLs

Severity:	MEDIUM
CVSS v3.1 Total Score:	5.3
CVSS v3.1 Vector:	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N
CWE:	215 – Information Exposure Through Debug Information
Affected Component:	

4.2.9.1 Description

Accessing any URL ending with `blog/` on <anonymized> and <anonymized> results in an empty blog template being shown. An example is the URL `https://<anonymized>/testingblog/`. This may allow an attacker to access resources ending with `blog/` that may not be intended to be exposed.

4.2.9.2 Solution Advice

X41 recommends to restrict the route to the blog.

4.2.10 SHRTNM-PT-23-10: Server and OS Version Information Exposed

Severity:	MEDIUM
CVSS v3.1 Total Score:	5.3
CVSS v3.1 Vector:	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N
CWE:	215 – Information Exposure Through Debug Information
Affected Component:	

4.2.10.1 Description

The version and type of the web server running was discovered via the Server HTTP header on any valid request to `<anonymized>` and `<anonymized>`:

```
1 Server: nginx/1.4.6 (Ubuntu)
```

Listing 4.4: Server Header

This may allow an attacker to conduct further attacks specific to the used software revisions and operating system.

4.2.10.2 Solution Advice

X41 recommends to disable the server header or set it to a generic value. For nginx it is recommended to use the *server_tokens: off* directive[16].

[16] https://nginx.org/en/docs/http/ngx_http_core_module.html#server_tokens

4.2.11 SHRTNM-PT-23-11: NTP Service Exposed to the Internet

Severity:	MEDIUM
CVSS v3.1 Total Score:	5.3
CVSS v3.1 Vector:	CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:L
CWE:	284 – Improper Access Control
Affected Component:	

4.2.11.1 Description

As shown in section 4.1, an NTP[17] service is exposed externally to the Internet via UDP port 123 on IP address <anonymized>. While there was no known flaw identified in the exposed service, several security vulnerabilities were published in the past regarding NTP software. Additionally, it is possible to amplify DDoS[18] attacks[19] using exposed NTP services.

4.2.11.2 Solution Advice

X41 recommends to disable the NTP service or alternatively implement a firewall rule to disable network access to UDP port 123 from the Internet.

[17] Network Time Protocol
[18] Distributed Denial of Service
[19] `https://support.ntp.org/bin/view/Main/SecurityNotice`

4.3 Informational Notes

The following observations do not have a direct security impact or affect functionality and other topics that are not directly related to security.

4.3.1 SHRTNM-PT-23-100: No or Misconfigured Strict Transport Security

Affected Component:	https://example.com

4.3.1.1 Description

The following hosts do not use or use misconfigured HSTS[20]. This header helps prevent visitors from connecting to the page unencrypted, which could allow an attacker to intercept and modify information.

- `https://example.com`

The HSTS flag requests the user agent to subsequentely enforce encrypted HTTPS connections, such that a MITM cannot make the user agent use HTTP.

Further information about HSTS can be found on the OWASP website[21].

4.3.1.2 Solution Advice

X41 recommends to set the HSTS header (*`Strict-Transport-Security`*), so that the browser enforces all accesses to the host are performed via HTTPS.

[20] HTTP Strict Transport Security
[21] `https://cheatsheetseries.owasp.org/cheatsheets/HTTP_Strict_Transport_Security_Cheat_Sheet.html`

4.3.2 SHRTNM-PT-23-101: Cookies Without the HttpOnly Flag

Affected Component:	https://example.com

4.3.2.1 Description

The following pages do not set the *HttpOnly*[22] flag for the cookies provided:

- `https://example.com` - Cookie *<an>*

The *HttpOnly* flag prevents JavaScript programs to access the cookie value. This prevents attackers from abusing XSS[23] attacks to steal cookies with sensitive information such as session tokens.

4.3.2.2 Solution Advice

X41 recommends to set the *HttpOnly* flag for all cookie values.

[22] `https://blog.codinghorror.com/protecting-your-cookies-httponly/`
[23] Cross-site Scripting

5 About X41 D-Sec GmbH

X41 D-Sec GmbH is an expert provider for application security and penetration testing services. Having extensive industry experience and expertise in the area of information security, a strong core security team of world-class security experts enables X41 D-Sec GmbH to perform premium security services.

X41 has the following references that show their experience in the field:

- Source code audit of the Git source code version control system[1]
- Review of the Mozilla Firefox updater[2]
- X41 Browser Security White Paper[3]
- Review of Cryptographic Protocols (Wire)[4]
- Identification of flaws in Fax Machines[5,6]
- Smartcard Stack Fuzzing[7]

The testers at X41 have extensive experience with penetration testing and red teaming exercises in complex environments. This includes enterprise environments with thousands of users and vendor infrastructures such as the Mozilla Firefox Updater (Balrog).

Fields of expertise in the area of application security encompass security-centered code reviews, binary reverse-engineering and vulnerability-discovery. Custom research and IT security consulting, as well as support services, are the core competencies of X41. The team has a strong technical background and performs security reviews of complex and high-profile applications such as Google Chrome and Microsoft Edge web browsers.

X41 D-Sec GmbH can be reached via `https://x41-dsec.de` or `mailto:info@x41-dsec.de`.

[1] `https://x41-dsec.de/security/research/news/2023/01/17/git-security-audit-ostif/`
[2] `https://blog.mozilla.org/security/2018/10/09/trusting-the-delivery-of-firefox-updates/`
[3] `https://browser-security.x41-dsec.de/X41-Browser-Security-White-Paper.pdf`
[4] `https://www.x41-dsec.de/reports/Kudelski-X41-Wire-Report-phase1-20170208.pdf`
[5] `https://www.x41-dsec.de/lab/blog/fax/`
[6] `https://2018.zeronights.ru/en/reports/zero-fax-given/`
[7] `https://www.x41-dsec.de/lab/blog/smartcards/`

Acronyms

API Application Programming Interface 26
CDN Content Delivery Network 8
CORS Cross-Origin Resource Sharing 18
CSRF Cross-Site Request Forgery
CVSS Common Vulnerability Scoring System 10
CWE Common Weakness Enumeration 13
DDoS Distributed Denial of Service 30
DES Data Encryption Standard 21
DNS Domain Name System 15
HSTS HTTP Strict Transport Security 31
HTTP HyperText Transfer Protocol 22
HTTPS HyperText Transfer Protocol Secure 22
IP Internet Protocol
JWT JSON Web Token 8
MITM Machine-in-the-middle Attack 20
NTP Network Time Protocol 30
OWASP Open Web Application Security Project 8
SSL Secure Sockets Layer 8
TCP Transmission Control Protocol 20
TLS Transport Layer Security 8
UDP User Datagram Protocol 15
URL Uniform Resource Locator 22
XSS Cross-site Scripting 32