Dubex:
MANAGING RISK. ENABLING GROWTH.

Web Application Penetration test of app.champ.dk for Champ ApS

MARCH 2020

DUBEX A/S

1 Introduction

The purpose of a web application penetration test is to gain a realistic picture of threats and vulnerabilities that may affect the confidentiality, integrity and availability of a web application.

Champ ApS, henceforth the Customer, contracted Dubex to perform a web application penetration tests of app.champ.dk and related web services, henceforth the System. The System was available and tested on the URL https://app.champ.dk.

The customer goals of this test were to

- Evaluate the overall security stance of its platform and infrastructure.
- Verify that robust and adequate access controls were in place to prevent unauthorized access to the systems or the data they hold.

This report describes the observations made by Dubex' security consultants during penetration tests, performed in February 2020, and how they could be abused. Focus is around the overall findings and providing advice on how to mitigate or prevent the risks related to the findings.

Report content includes selected screenshots, error message examples, directory listings and other findings. However, the report should not be considered an exhausted list of findings but a prioritized list of the findings to which fixes will provide the most value to the Customer and their business. As supplement to the report Dubex can provide detailed reports from tools such as Burp Suite Pro, Nmap and others that contain explicit technical details.

2 Content

3 Findings and observations

The System follows current best practices for secure development of web-based applications and the underlying infrastructure does follow current best practices for secure web application server configuration.

The Customer's goal with performing the tests were to

- *Evaluate the overall security stance of its platform and infrastructure.*
  The overall security stance of the systems was found to be adequate with a high level of applied technical security measures.
- *Verify that robust and adequate access controls were in place to prevent unauthorized access to the systems or the data they hold.*
  The access controls were tested and found adequate. It was not possible to bypass the authentication or authorization mechanisms during testing.

Dubex made a few observations around system configuration that could be further hardened, namely:

- Partial implementation of Content-Security-Policy.
- HTTP headers with system metadata.

It was not possible to abuse these observations, but they may be helpful to an attacker if other vulnerabilities are discovered.

3.1 Recommendations

Dubex recommends that the following mitigating actions are performed on the System:

- Implement the HTTP security header Content-Security-Policy in block mode.
  The header was under implementation during testing and was set to report-only mode. Once this implementation has been finished it is expected to increase the security level of the System even further.
- De-activate all HTTP headers that leak metadata about technology infrastructure.
  Specific system metadata may aid an attacker in finding suitable exploits against a system.

4 Test scope

The Customer wished to test the following components of the System, which are all in scope.

SERVER NAME / DNS	INTERNAL IP	EXTERNAL IP (NAT)
APP.CHAMP.DK	-	13.80.21.25
UPLOAD.CHAMP.DK	-	13.80.21.25

The application layer includes a web application located at https://app.champ.dk and an accompanying file upload service located at https://upload.champ.dk.

An initial vulnerability scan was performed and succeeded by targeted, manual penetration tests. Testing was performed over the internet against the System's production environment. As a part of these tests Dubex was provided with a set of user credentials to test authentication and authorization. As a part of testing the public attack surface of the System is determined as it would be seen and possibly abused by a motivated attacker.

Initially, the System's URL structure was mapped as an authenticated user to save time and provide a basis for unauthenticated testing. Subsequently, the URLs discovered were tested as both authenticated and authenticated users.

5 Test execution

After the Customer's accept of the agreed scope of testing Dubex began the reconnaissance phase as described in the Penetration Testing Execution Standard.

5.1 Intelligence Gathering

Based on the information provided by the Customer about the System's public URLS an initial port scan and manual inspection of the System's public pages were performed.

From Dubex' test systems a thorough port scan with Nmap was performed to identify active services and potential network access. The scan returned the following results:

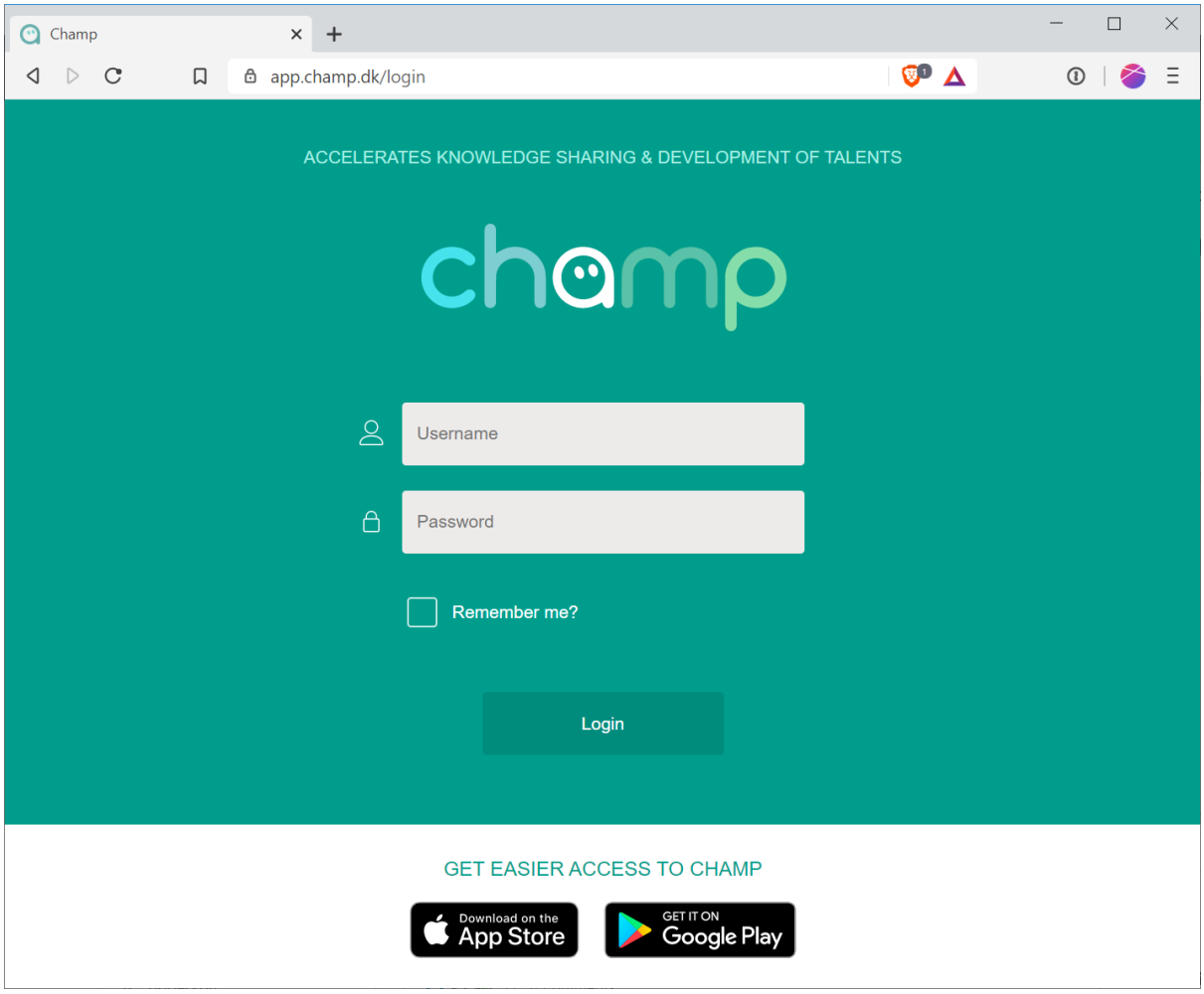
```
# Nmap 7.80 scan initiated Mon Feb 10 09:35:45 2020 as: nmap -sV -sC -sS -Pn -T5 -p- -vvv -oA
./quick.txt app.champ.dk
Nmap scan report for app.champ.dk (13.80.21.25)
Host is up, received user-set (0.00063s latency).
Scanned at 2020-02-10 09:35:45 CET for 506s
Not shown: 65469 filtered ports
Reason: 65469 no-responses
PORT     STATE  SERVICE    REASON          VERSION
80/tcp   open   http       syn-ack ttl 114 Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
| http-server-header:
|   Microsoft-HTTPAPI/2.0
|_  Microsoft-IIS/10.0
|_http-title: Did not follow redirect to https://app.champ.dk/
443/tcp  open   ssl/http   syn-ack ttl 114 Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-favicon: Unknown favicon MD5: EE2A3E55C9E76A5CAF750FCBD7552651
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
| http-robots.txt: 1 disallowed entry
|_/
| http-server-header:
|   Microsoft-HTTPAPI/2.0
|_  Microsoft-IIS/10.0
| http-title: Champ
|_Requested resource was https://app.champ.dk/login
| ssl-cert: Subject: commonName=app.champ.dk
| Subject Alternative Name: DNS:app.champ.dk
| Issuer: commonName=Let's Encrypt Authority X3/organizationName=Let's Encrypt/countryName=US
| Public Key type: rsa
| Public Key bits: 3072
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2020-01-05T08:00:09
| Not valid after:  2020-04-04T08:00:09
| MD5:   daad 7844 ce8b f686 d444 7917 6e4d b94a
| SHA-1: 804f ec59 9ed2 6516 abe0 11d3 4b80 a329 aa3f f008
| -----BEGIN CERTIFICATE-----
|… Shortened for brevity
|_-----END CERTIFICATE-----
|_ssl-date: 2020-02-10T08:44:11+00:00; 0s from scanner time.
| tls-alpn:
|   h2
|_  http/1.1
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: 0s

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Mon Feb 10 09:44:11 2020 -- 1 IP address (1 host up) scanned in 506.59 seconds
```

The web server of the System responds to http requests on TCP port 80 (HTTP) and TCP port 443 (HTTPS) but no further network ports. Traffic to port 80 is redirected to port 443 per best practices.

The front page of the System displays a login page that does not directly provide clues about technology or application server choices.



Source code inspection and HTTP headers indicate that the System utilizes Microsoft's .NET Core technology.

This information is used for initial threat modelling that serves as a basis for further vulnerability analysis and manual penetration testing.

5.2 Threat Modelling

Based on the System description and identification of open ports the most likely threats against the System are estimated to be employees with system access and improper implementation of access control to System components.

The following test scenarios were deemed relevant to begin with:

- Unauthenticated access to the System via the Internet.
- A user with system access.
- A user that is tricked into performing an activity using a malicious link or similar.

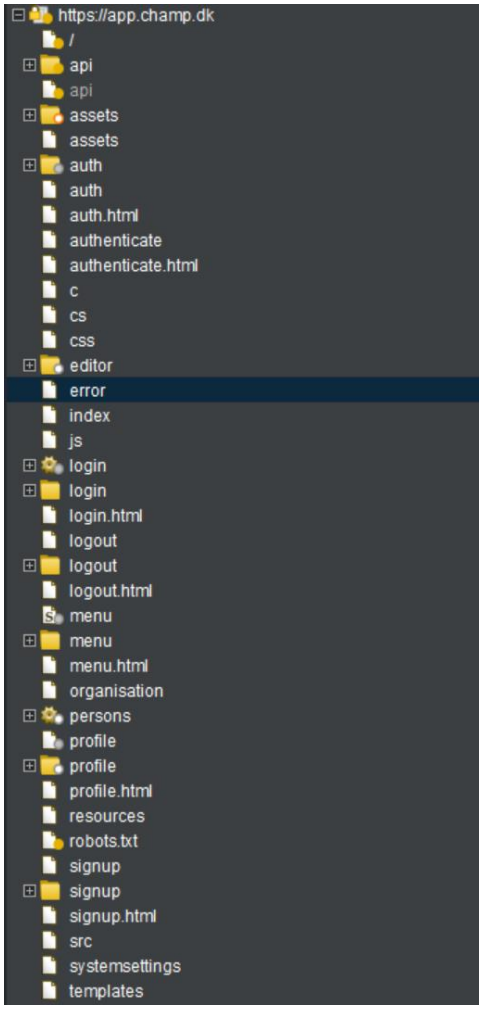
5.3 Vulnerability Analysis

Based on the collected information and threat modelling testing was initiated in late February 2020.

Per agreement with the Customer testing began using the provided test credentials to the System and subsequently without authentication. This approach was recommended to perform a comprehensive initial mapping of the application based on an authenticated user. Authentication mechanisms for discovered endpoints were then tested in-depth.

Furthermore, a network vulnerability scan was performed with Nmap scripts and manual tests for typical vulnerabilities in the discovered server and service. None of theses indicated the presence of known vulnerabilities.

After the mapping of network services the System's application layer was thoroughly analyzed using Burp suite, a web application attack proxy. Burp was also used for content discovery scans to identify further hidden, but publicly available, content. None was found. Based on discovered URLs a web application vulnerability scan was performed with Burp Suite and results were manually verified. An example of Burps site tree navigation can be seen below.



Figur 1 Tree structure of content discovered using Burp Suite

5.4 Findings and observations

Through vulnerability scanning and manual testing two observations were made around system security configuration that could be improved:

- Partial implementation of Content-Security-Policy.
- HTTP headers with system metadata.

The two observations are further described below.

5.4.1 Partial implementation of Content-Security-Policy.

The System implements most modern HTTP security headers according to current best practice. These headers provide good basic protection against a number of attack vectors, XSS and CSRF in particular.

One of the headers, Content-Security-Policy, was implemented in report-only mode during testing. As such it does not enable the enhanced security controls against malicious content injection that modern browsers otherwise enable.

Dubex recommends fully implementing Content-Security-Policy in block mode and restrict third-party content inclusion to a minimum. This can be time-consuming but can significantly increase the overall protection level against client-side injection attacks.

The following description of Content-Security-Policy and a future recommendation, Feature-Policy, comes from the site https://securityheaders.com which also provides accessible testing tools.

Content-Security-Policy	Content Security Policy is an effective measure to protect your site from XSS attacks. By whitelisting sources of approved content, you can prevent the browser from loading malicious assets.
Feature-Policy	Feature Policy is a new header that allows a site to control which features and APIs can be used in the browser.

These headers should be implemented to the furthest extent possible.

5.4.2 HTTP headers with system metadata

The System is configured to send HTTP headers with metadata about its underlying server and application framework. This is not a vulnerability in its own right but may aid and attacker in identifying vulnerable components during an attack.

These metadata headers were observed while communicating with the System:

- Server: Microsoft-HTTPAPI/2.0
- X-Powered-By: ServiceStack/5.60 NetCore/Windows

Such header should be removed unless they serve a specific purpose and are used actively in the System. This can usually be done centrally using web.config configuration settings.

5.4.3 Further testing

This section describes some attempted attacks that failed due to correct server configuration and safe coding practices. The attacks included but were not limited to:

- Authentication bypass attempts.
- Authorization bypass attempts.
- Code injection in different document types and endpoints
- Session token manipulation
- Web socket data manipulation (Not in use)

- HTTP method tampering.

As the System provides endpoints that can accept XML input special focus was on finding XML External Entity (XXE) vulnerabilities and deserialization attacks using malicious code injection in benign service requests. The System appears to be well protected against these types of attack through the .NET frameworks generic input validation and correct us of XML input validation.

5.4.3.1 Test of cipher suites used in the encryption of https connections

The System has been tested for support for old and insecure SSL and TLS encryption cipher suites. Such ciphers may allow an attacker to perform Man-In-The-Middle attacks.

The System *does not* support old and insecure cipher suites and its transport-level security level can be said to be high.

```
nmap --script ssl-enum-ciphers -p 443 app.champ.dk
Starting Nmap 7.80 ( https://nmap.org ) at 2020-02-27 12:11 CET
Nmap scan report for app.champ.dk (13.80.21.25)
Host is up (0.013s latency).

PORT    STATE SERVICE
443/tcp open  https
| ssl-enum-ciphers:
|   TLSv1.2:
|     ciphers:
|       TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (ecdh_x25519) - A
|       TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (ecdh_x25519) - A
|       TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (dh 2048) - A
|       TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (dh 2048) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (ecdh_x25519) - A
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (ecdh_x25519) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (ecdh_x25519) - A
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (ecdh_x25519) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 2048) - A
|       TLS_RSA_WITH_AES_256_GCM_SHA384 (rsa 3072) - A
|       TLS_RSA_WITH_AES_128_GCM_SHA256 (rsa 3072) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA256 (rsa 3072) - A
|       TLS_RSA_WITH_AES_128_CBC_SHA256 (rsa 3072) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 3072) - A
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 3072) - A
|     compressors:
|       NULL
|     cipher preference: server
|     warnings:
|       Key exchange (dh 2048) of lower strength than certificate key
|_  least strength: A

Nmap done: 1 IP address (1 host up) scanned in 1.70 seconds
```

It is recommended that review and hardening of supported cipher suites are done on a regular basis and deprecated cipher suites are removed.

5.5 Exploitation and Post-Exploitation

As no vulnerabilities were found abusable, directly or indirectly, no exploitation or post-exploitation was possible. Dubex was not able to gain further access to the Customers infrastructure and/or data access.

6 Conclusion

The system adheres to current best practices for secure development of web applications and is seen as having a high level of security and corresponding low risk-level.

No vulnerabilities were found in the System that could be directly or indirectly abused to gain access to the Customers infrastructure, systems or data.

During vulnerability scans and penetration tests some observations of possible improvements to the security level of the application were made. A minor information level configuration issue and an unfinished implementation of Content-Security-Policy, that will further strengthen the System's security stance. It has not been possible to abuse these findings, but they may aid an attacker in an attack against future vulnerabilities.