Computest

Security test

Confidential

AFAS Software

Pocket API

Sept. 23, 2019

Version 1.0

Laurens Baardman

Computest in a nutshell

Computest, your **digital superhero**!

We are a team of passionate and experienced technical specialists. The digital world is our playground.

It is our daily challenge to make our customers' IT better, faster and more secure. We do so by testing on all areas. And we're the best at it!

Training

Advice

Testing

Review

Security

Through testing, consulting and auditing we help our clients to increase and to monitor their digital security.

- > Pentesting & vulnerability assessment
- > Hybrid infrastructure scanning
- > Code audit
- > Advice & Consultancy
- > DevOps-integration
- > Mobile app security
- > Security training

Performance

Through testing we investigate the load that applications can handle and we identify any possible bottlenecks.

- > Stress-, Load- or endurance testing
- > Performance-expertise in development teams
- > Unique loadfarm: simulate 250,000 concurrent users
- > Application Performance Monitoring (APM)
- > Integration of tests in CI/CD
- > Performance training

Functional

Through functional testing and test automation we streamline our clients' test efforts and we help to safeguard quality.

- > Functional testing
- > Setting up test automation programmes
- > Integration of testing in CI/CD
- > Designing testing in DevOps-context
- > Application monitoring 24x7
- > Tooling-independent

Marvin_, our **digital superhero**!

A test providing a snapshot in time is fine, but daily insight is better. Marvin_ provides you with continuous insight into how matters stand with the security, availability and performance of your applications. By carrying out a daily security scan, as well as monitoring every few minutes the critical paths of your web apps for availability and response times, you can always be sure how things stand with your important applications..

Contents

1.
Executive summary

1.1. Summary

Computest conducted a vulnerability assessment on the application of AFAS Pocket. The goal of this test was to determine to what extent the Pocket API and browser version are vulnerable to an attack from the internet.

Computest is of the opinion that the Pocket API itself has an outstanding security level. The API contains no vulnerabilities that could directly lead to the compromise of confidentiality or integrity of data. In this regard, the quality of the API is comparable to the connector API's that are used by AFAS Profit.

However, there are a few improvements that can be made for the web version, and a few improvements to further enhance the level of security of the API. The browser implementation uses outdated software with known vulnerabilities that should be updated to the latest version. Regarding the API, session management can be improved. Currently, authentication tokens are valid indefinitely, which means that when an attacker gets hold of a token, he has permanent access to the API, until the token is revoked in Profit.

1.2. Vulnerability impact overview

The table below provides an overview of the discovered vulnerabilities. The impact level of each vulnerability is determined by its assigned CVSS score. More information about vulnerability scores can be found in the introduction of this report.

Section	High (7.0-10.0)	Medium (4.0-6.9)	Low (0.0-3.9)	Total
Web scan	0	2	3	**5**
Total	**0**	**2**	**3**	**5**

1.3. Next steps

Medium impact

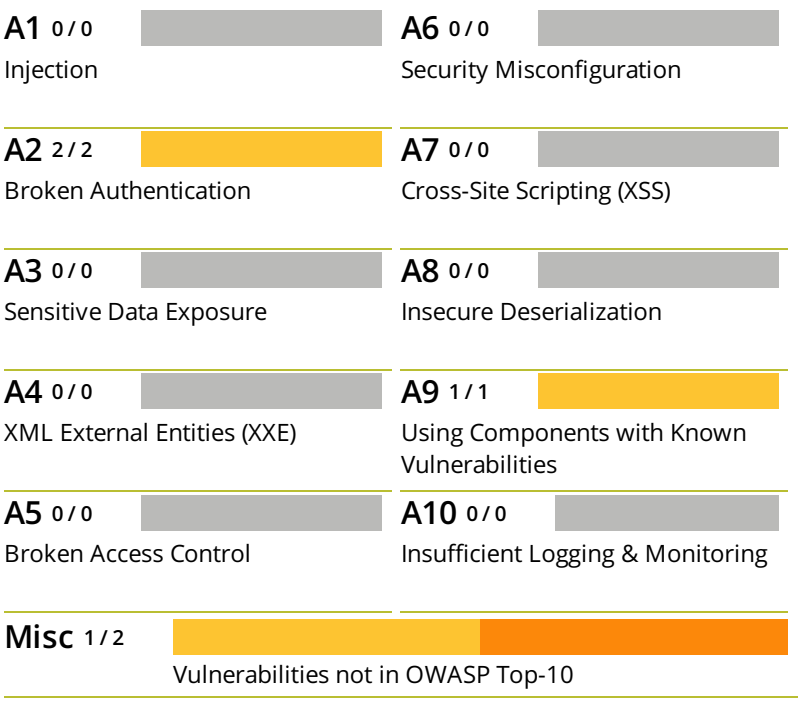
> Update outdated software with the most recent patches and updates. (**4.1.1**)
> Use the POST method for state changing requests. (**4.7.1**)

Low impact

> Revoke the session on the server whenever a user logs out. (**4.7.3**)
> Make sure that user sessions expire periodically. (**4.7.16**)
> Use anti content sniffing headers on all pages. (**4.9.3**)

2.

OWASP TOP 10 Overview



Medium	1
Misc Test for cross-site request forgery (CSRF)	4.3

The OWASP Top 10 is a popular project by the OWASP community, that aims to chart the most prevalent security risk in web applications. Every few years the list is revised, so that the most current risks are included.

The items on the OWASP Top 10 list aren't practical vulnerabilities, but categories of risk. For this reason, Computest cannot conduct a test for the "OWASP Top 10 items", but the vulnerabilities can be linked to the relevant OWASP Top 10 categories. This page shows the mapping from the vulnerabilities that were found during this security test to the respectable OWASP Top 10 items they are related to.

Not all items that are tested for can be mapped to an OWASP Top 10 item. This is obvious, since the Top 10 is only a "Top" list, and other vulnerability categories carry risk just as well. Those items are shown at the bottom of the table.

The OWASP Top 10 list from 2017 has been used for the mapping. This list is published at https://www.owasp.org/index.php/Top_10-2017_Top_10.

4.
Web scan

The goal of this test is to determine whether the systems in scope of this test are vulnerable to an attack from the internet. The conducted test is based on the Certified Secure Web Application Security Test Checklist.

#	Certified Secure Web Application Security Test Checklist	Result
1.0	**Deployment**	
1.1	Test for missing security updates	✗
1.2	Test for unsupported or end-of-life software versions	✓
1.3	Test for HTTP TRACK and TRACE methods	✓
1.4	Test for extraneous functionality	✓
1.5	Test the server using the Server Security Test Checklist	✓
2.0	**Information Disclosure**	
2.1	Test for extraneous files in the document root	✓
2.2	Test for extraneous directory listings	✓
2.3	Test for accessible debug functionality	✓
2.4	Test for sensitive information in log and error messages	✓
2.5	Test for sensitive information in robots.txt	●
2.6	Test for sensitive information in source code	✓
2.7	Test for disclosure of internal addresses	✓
3.0	**Privacy and Confidentiality**	
3.1	Test for sensitive information stored in URLs	✓
3.2	Test for unencrypted sensitive information stored at the client-side	✓
3.3	Test for sensitive information stored in (externally) archived pages	✓
3.4	Test for content included from untrusted sources	✓
3.5	Test for caching of pages with sensitive information	✓
3.6	Test for insecure transmission of sensitive information	✓
3.7	Test for non-SSL/TLS pages on sites processing sensitive information	✓
3.8	Test for SSL/TLS pages served with mixed content	✓
3.9	Test for missing HSTS header on full SSL sites	✓
3.10	Test for known vulnerabilities in SSL/TLS	✓

#	Certified Secure Web Application Security Test Checklist	Result
3.11	Test for weak, untrusted or expired SSL certificates	✓
3.12	Test for the usage of unproven cryptographic algorithms	✓
3.13	Test for the incorrect usage of cryptographic primitives	✓
4.0	**State Management**	
4.1	Test for client-side state management	✓
4.2	Test for invalid state transitions	✓
5.0	**Authentication and Authorization**	
5.1	Test for missing authentication or authorization	✓
5.2	Test for client-side authentication	✓
5.3	Test for predictable and default credentials	✓
5.4	Test for predictable authentication or authorization tokens	✓
5.5	Test for authentication or authorization based on obscurity	✓
5.6	Test for identifier-based authorization	✓
5.7	Test for acceptance of weak passwords	✓
5.8	Test for plaintext retrieval of passwords	✓
5.9	Test for missing rate limiting on authentication functionality	✓
5.10	Test for missing re-authentication when changing credentials	✓
5.11	Test for missing logout functionality	✓
6.0	**User Input**	
6.1	Test for SQL injection	✓
6.2	Test for path traversal and filename injection	✓
6.3	Test for cross-site scripting	✓
6.4	Test for system command injection	✓
6.5	Test for XML injection	✓
6.6	Test for XPath injection	✓
6.7	Test for XSL(T) injection	✓
6.8	Test for SSI injection	✓
6.9	Test for HTTP header injection	✓
6.10	Test for HTTP parameter injection	✓
6.11	Test for LDAP injection	✓
6.12	Test for dynamic scripting injection	✓
6.13	Test for regular expression injection	✓

#	Certified Secure Web Application Security Test Checklist	Result
6.14	Test for data property/field injection	✓
6.15	Test for protocol-specific injection	✓
6.16	Test for expression language injection	✓
7.0	**Sessions**	
7.1	Test for cross-site request forgery (CSRF)	✗
7.2	Test for predictable CSRF tokens	✓
7.3	Test for missing session revocation on logout	✗
7.4	Test for missing session regeneration on login	✓
7.5	Test for missing session regeneration when changing credentials	✓
7.6	Test for missing revocation of other sessions when changing credentials	✓
7.7	Test for missing Secure flag on session cookies	✓
7.8	Test for missing HttpOnly flag on session cookies	✓
7.9	Test for non-restrictive domain on session cookies	✓
7.10	Test for non-restrictive or missing path on session cookies	✓
7.11	Test for predictable session identifiers	✓
7.12	Test for session identifier collisions	✓
7.13	Test for session fixation	✓
7.14	Test for insecure transmission of session identifiers	✓
7.15	Test for external session hijacking	✓
7.16	Test for missing periodic expiration of sessions	✗
8.0	**File Uploads**	
8.1	Test for storage of uploaded files in the document root	●
8.2	Test for execution or interpretation of uploaded files	●
8.3	Test for uploading outside of designated upload directory	●
8.4	Test for missing size restrictions on uploaded files	●
8.5	Test for missing type validation on uploaded files	●
9.0	**Content**	
9.1	Test for missing or non-specific content type definitions	✓
9.2	Test for missing character set definitions	✓
9.3	Test for missing anti content sniffing measures	✗
10.0	**XML Processing**	
10.1	Test for XML external entity expansion	✓

#	Certified Secure Web Application Security Test Checklist	Result
10.2	Test for external DTD parsing	
10.3	Test for extraneous or dangerous XML extensions	
10.4	Test for recursive entity expansion	
11.0	**Miscellaneous**	
11.1	Test for missing anti-clickjacking measures	
11.2	Test for open redirection	
11.3	Test for insecure cross-domain access policy	
11.4	Test for missing rate limiting on e-mail functionality	
11.5	Test for missing rate limiting on resource intensive functionality	
11.6	Test for inappropriate rate limiting resulting in a denial of service	
11.7	Test for application- or setup-specific problems	

5.
Conclusion

Computest has performed a vulnerability assessment on the AFAS Pocket API and the web interface for the Pocket app. The goal of this assessment was to determine to what extent this platform is vulnerable to an attack from the internet.

Computest is of the opinion that the Pocket API has an outstanding level of security. It is clear that AFAS holds security in a high regard.

However, there are several improvements that can be made for both the API and web interface. The API can be improved in terms of session management. Currently, authentication tokens are valid indefinitely and not revoked automatically. Moreover, tokens remain constant over sessions, which means that logging out of the application does not revoke the token. Computest recommends providing a way for users to invalidate a sessions and obtain a new token for the API.

The web interface uses outdated software. This particular AngularJS version is end-of-life and should be updated immediately. Also, anti-content-sniffing measures could further improve the security for Internet Explorer users.

Computest makes the following recommendations, ordered by priority:

Score	Description	Chapter
4.3	Use unique form tokens for data modifications.	**4.7.1**
4.3	Update outdated software with the most recent patches and updates.	**4.1.1**
2.6	Use anti content sniffing headers on all pages.	**4.9.3**
2.6	Make sure that user sessions expire periodically.	**4.7.16**
2.6	Revoke the session on the server whenever a user logs out.	**4.7.3**