UnderDefense
CyberSecurity solutions protecting your business

Penetration Testing Report for [CLIENT]

Executive summary

This report presents the results of the “Grey Box” penetration testing for [CLIENT] Infrastructure. The recommendations provided in this report structured to facilitate remediation of the identified security risks. This document serves as a formal letter of attestation for the recent [CLIENT] Infrastructure Penetration Testing.
Evaluation ratings compare information gathered during the course of the engagement to “best in class” criteria for security standards. We believe that the statements made in this document provide an accurate assessment of [CLIENT] Infrastructure.
We highly recommend to review section of Summary of business risks and High-Level Recommendations for better understanding of risks and discovered security issues.

Scope	Security level	Grade
Infrastructure perimeter	C	Fair

UnderDefense Grading Criteria:

Grade	Security	Criteria Description
A	Excellent	The security exceeds “Industry Best Practice” standards. The overall posture was found to be excellent with only a few low-risk findings identified.
B	Good	The security meets with accepted standards for “Industry Best Practice.” The overall posture was found to be strong with only a handful of medium- and low- risk shortcomings identified.
C	Fair	Current solutions protect some areas of the enterprise from security issues. Moderate changes are required to elevate the discussed areas to “Industry Best Practice” standards
D	Poor	Significant security deficiencies exist. Immediate attention should be given to the discussed issues to address exposures identified. Major changes are required to elevate to “Industry Best Practice” standards.

F	Inadequate	Serious security deficiencies exist. Shortcomings were identified throughout most or even all of the security controls examined. Improving security will require a major allocation of resources.

Assumptions & Constraints

As the environment changes, and new vulnerabilities and risks are discovered and made public, an organization's overall security posture will change. Such changes may affect the validity of this letter. Therefore, the conclusion reached from our analysis only represents a "snapshot" in time.

Objectives & Scope

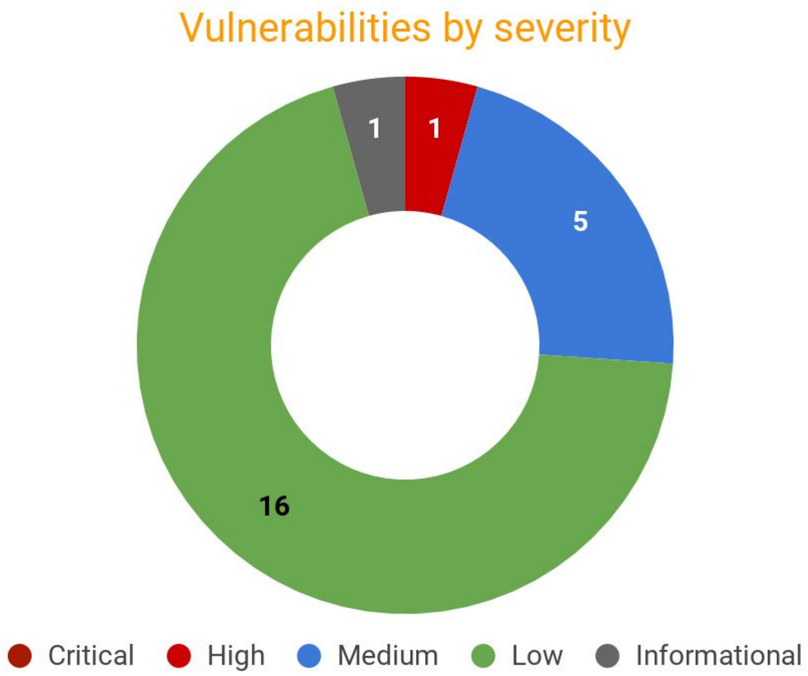
Organization	[CLIENT]
Audit type	Grey-Box Manual and Automated Infrastructure Penetration Testing
Asset URL	APPENDIX A – Scope
Audit period	Oct. 10 - Oct. 31

Consultants performed discovery process to gather information about the target and searched for information disclosure vulnerabilities. With this data in hand, we conducted the bulk of the testing manually, which consisted of input validation tests, impersonation (authentication and authorization) tests, and session state management tests. The purpose of this penetration testing is to illuminate security risks by leveraging weaknesses within the environment that lead to the obtainment of unauthorized access and/or the retrieval of sensitive information. The shortcomings identified during the assessment were used to formulate recommendations and mitigation strategies for improving the overall security posture.

Results Overview

The test uncovered a few vulnerabilities that may cause users session hijacking, sensitive data leakage, broken confidentiality and integrity and availability of the resource. Security testing activities showed that there are a lot of components with known vulnerabilities.

Identified vulnerabilities are not directly exploitable but the risk posed by these vulnerabilities can cause significant damage to the company.



Security experts performed manual security testing, which demonstrate the following results.

Severity	Critical	High	Medium	Low	Informational
# of issues	0	1	5	16	1

Severity scoring:

- Critical – Immediate threat to key business processes.
- High – Direct threat to key business processes.
- Medium – Indirect threat to key business processes or partial threat to business processes.
- Low – No direct threat exists. Vulnerability may be exploited using other vulnerabilities.
- Informational – This finding does not indicate vulnerability, but states a comment that notifies about design flaws and improper implementation that might cause a problem in the long run.

Summary of business risks

High severity issues make direct threat to the business as they can be used to:

- Using XSS attack it is possible to steal user session and get full access to user's account. It will lead to client data leakage as this vulnerability is present on application whis is used by company clients. Successful exploitation of this vulnerability will create big reputational and financial damage for company.

Medium and low severity issues can lead to:

- Attacks on communication channels and as a result on sensitive data leakage and possible modification, in other words it affects integrity and confidentiality of data transferred.
- Information leakage about system components which may be used by attackers for further malicious actions.
- Attacks on old and not patched system components with bunch of publicly known vulnerabilities.
- Enumerating existing users emails/usernames and brute forcing their passwords. Easy access to their session after exploitation of high level risks, as session token is the same for each login.
- Social engineering, intimidation or blackmail is possible via sending emails from open SMTP server, which can interrupt working process, cause credentials stealing or can be used with combination with other attacks.
- Combination of few issues can be used for successful realisation of attacks.

Informational severity issues do not carry direct threat but they can be used to gather useful information for an attacker.

High-Level Recommendations

Taking into consideration all issues that have been discovered, we highly recommend to:

- Conduct current vs. future IT/Security program review;
- Establish Secure SDLC best practices, assign Security Engineer to a project to monthly review code, conduct SAST & DAST security testing;
- Review Architecture of application;
- Deploy Web Application Firewall solution to detect any malicious manipulations;
- Continuously monitor logs for anomalies to detect abnormal behaviour and fraud transactions. Dedicate security operations engineer to this task;
- Implement Patch Management procedures for whole IT infrastructure and endpoints of employees and developers;
- Continuously Patch production and development environments and systems on regular bases with latest releases and security updates;
- Conduct annual Penetration test and quarterly Vulnerability Scanning against internal and external environment;
- Conduct security coding training for Developers;
- Develop and Conduct Security Awareness training for employees and developers;

- Develop Incident Response Plan in case of Data breach or security incidents;
- Analyse risks for key assets and resources;
- Update codebase to conduct verification and sanitization of user input on both, client and server side;
- Use only encrypted channels for communications;
- Improve server and application configuration to meet security best practises;
- Also we recommend to conduct remediation testing of both infrastructure and web applications.

Performed tests

- All set of applicable OWASP Top 10 Security Threats
- All set of applicable SANS 25 Security Threats

Criteria Label	Status
A1:2017-Injection	Meets criteria
A2:2017-Broken Authentication	Fails criteria
A3:2017-Sensitive Data Exposure	Fails criteria
A4:2017-XML External Entities (XXE)	Meets criteria
A5:2017-Broken Access Control	Meets criteria
A6:2017-Security Misconfiguration	Fails criteria
A7:2017-Cross-Site Scripting (XSS)	Fails criteria
A8:2017-Insecure Deserialization	Meets criteria
A9:2017-Using Components with Known Vulnerabilities	Fails criteria
A10:2017-Insufficient Logging&Monitoring	N/A

Performed Tests	Status
Host and service enumeration	Fails criteria
Weak passwords attack and brute-force	Fails criteria
Identification of misconfigurations	Fails criteria
Vulnerability identification and system exploitation	Fails criteria
Search Engine Discovery and Reconnaissance for Information Leakage	Fails criteria
Weak Authorization Mechanisms testing	Meets criteria
Database compromising, sensitive information stealing	Meets criteria
Outdated services	Fails criteria
S3 bucket enumeration	Meets criteria

Security tools used

- Burp Suite Pro [Commercial Edition]
- Nmap
- TestSSL
- Nikto
- Dirbuster
- GoBuster
- Arachni
- Nessus
- Hydra
- Harvester
- Sublister

Methodology

Our Penetration Testing Methodology grounded on following guides and standards:

- Penetration Testing Execution Standard
- OWASP Top 10 Application Security Risks - 2017
- OWASP Testing Guide
- SANS: Conducting a Penetration Test on an Organization
- The Open Source Security Testing Methodology

Open Web Application Security Project (OWASP) is an industry initiative for web application security. OWASP has identified the 10 most common attacks that succeed against web applications. These comprise the OWASP Top 10.

Application penetration test includes all the items in the OWASP Top 10 and more. The penetration tester remotely tries to compromise the OWASP Top 10 flaws. The flaws listed by OWASP in its most recent Top 10 and the status of the application against those are depicted in the table below.

Findings Details

Reflected Cross Site Scripting in multiple pages

SEVERITY: **High**

LOCATION:

- https://client.com/explore/categories/dir"><img src=a onerror=alert(1)>
- https://client.com/explore/categories/dir'-alert(1)-'
- https://client.com/member/login/r/dir"><script>alert(1)</script>
- https://client.com/page/dir?"><script>alert(1)</script>
- https://client.com/something1/something1/something1/something1?lexto"><script>alert(1)</script>
- https://client.com/something1/something1/something1/something1?nt6np"><script>alert(1)</script>
- https://client.com/something1/something1/something1/something1?yya23"><script>alert(1)</script>
- https://client.com/something1/something1/something1/something1/?owvx9"><script>alert(1)</script>
- https://client.com/something1/something1/something1?"><script>alert(1)</script>

ISSUE DESCRIPTION:

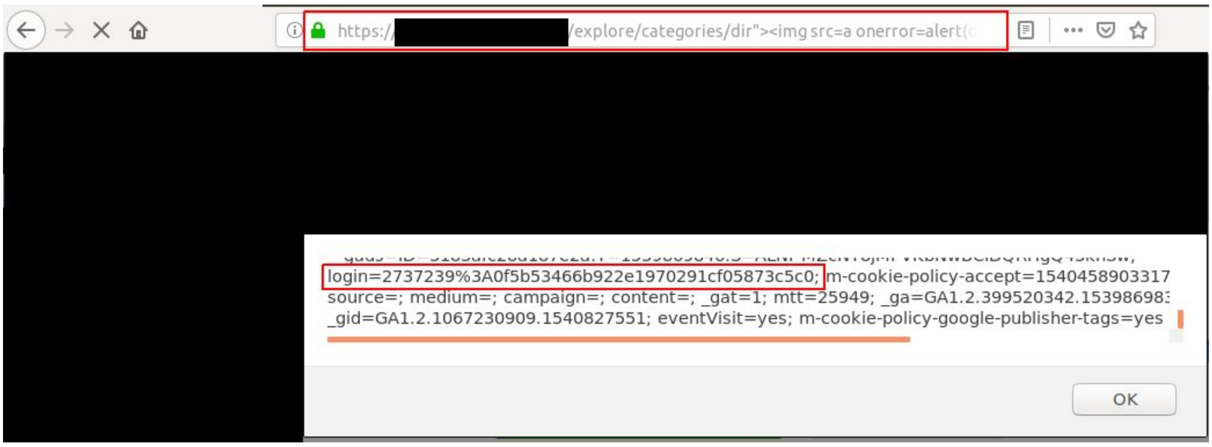
Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.

An attacker can use XSS to send a malicious script to an unsuspecting user. The end user’s browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.

PROOF OF VULNERABILITY:

The attacker can inject javascript code sent in URL directly to html code which will be executed.

https://client.com/explore/categories/dir%22%3E%3Cimg%20src=a%20onerror=alert(document.cookie)%3E

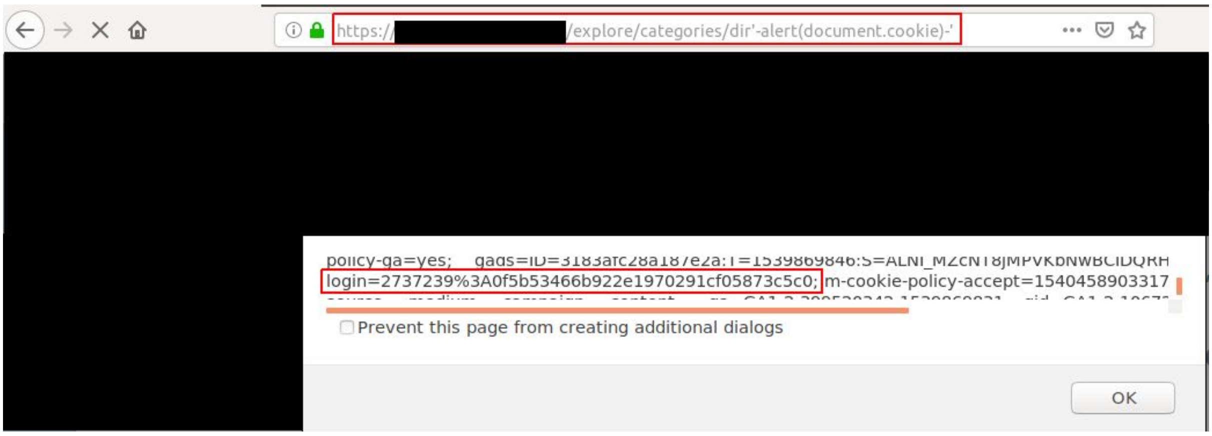


```
https://[redacted]/explore/categories/dir"><img src=a onerror=alert(
```

[redacted]

```
login=2737239%3A0f5b53466b922e1970291cf05873c5c0; m-cookie-policy-accept=1540458903317
source=; medium=; campaign=; content=; _gat=1; mtt=25949; _ga=GA1.2.399520342.15398698
_gid=GA1.2.1067230909.1540827551; eventVisit=yes; m-cookie-policy-google-publisher-tags=yes
```

https://client.com/explore/categories/dir'-alert(document.cookie)-'

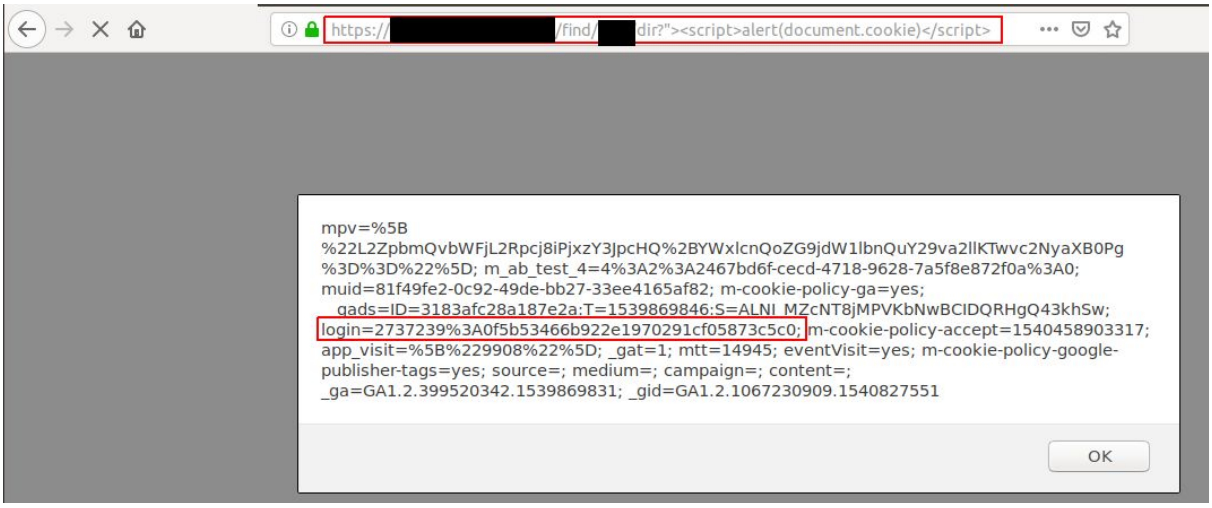


```
https://[redacted]/explore/categories/dir'-alert(document.cookie)-'
```

[redacted]

```
policy-ga=yes;   __gads=ID=3183afc28a187e2a:T=1539869846:S=ALNI_MZcNT8jMPVKbNWBCIDQRH
login=2737239%3A0f5b53466b922e1970291cf05873c5c0; m-cookie-policy-accept=1540458903317
```

☐ Prevent this page from creating additional dialogs https://www.client.com/something/dir?"><script>alert(document.cookie)</script>



```
https://[redacted]/find/[redacted]dir?"><script>alert(document.cookie)</script>
```

```
mpv=%5B
%22L2ZpbmQvbWFjL2Rpcj8iPjxzY3JpcHQ%2BYWxlcnQoZG9jdW1lbnQuY29va2llKTwvc2NyaXB0Pg
%3D%3D%22%5D; m_ab_test_4=4%3A2%3A2467bd6f-cecd-4718-9628-7a5f8e872f0a%3A0;
muid=81f49fe2-0c92-49de-bb27-33ee4165af82; m-cookie-policy-ga=yes;
__gads=ID=3183afc28a187e2a:T=1539869846:S=ALNI_MZcNT8jMPVKbNwBCIDQRHgQ43khSw;
login=2737239%3A0f5b53466b922e1970291cf05873c5c0; m-cookie-policy-accept=1540458903317;
app_visit=%5B%229908%22%5D; _gat=1; mtt=14945; eventVisit=yes; m-cookie-policy-google-
publisher-tags=yes; source=; medium=; campaign=; content=;
_ga=GA1.2.399520342.1539869831; _gid=GA1.2.1067230909.1540827551
```

RECOMMENDATIONS:

Preventing XSS requires separation of untrusted data from active browser content. This can be achieved by:

- Using frameworks that automatically escape XSS by design, such as the latest Ruby on Rails, React JS. Learn the limitations of each framework's XSS protection and appropriately handle the use cases which are not covered.
- Escaping untrusted HTTP request data based on the context in the HTML output (body, attribute, JavaScript, CSS, or URL) will resolve Reflected and Stored XSS vulnerabilities.

To filter user input sufficiently, consider XSS Prevention Cheat Sheet.

SMTP Server without authentication

SEVERITY: **Medium**

LOCATION:

- smtp client_ip:25

ISSUE DESCRIPTION:

It is possible for an attacker to connect to smtp port of the server and send emails to existing email accounts from domains:

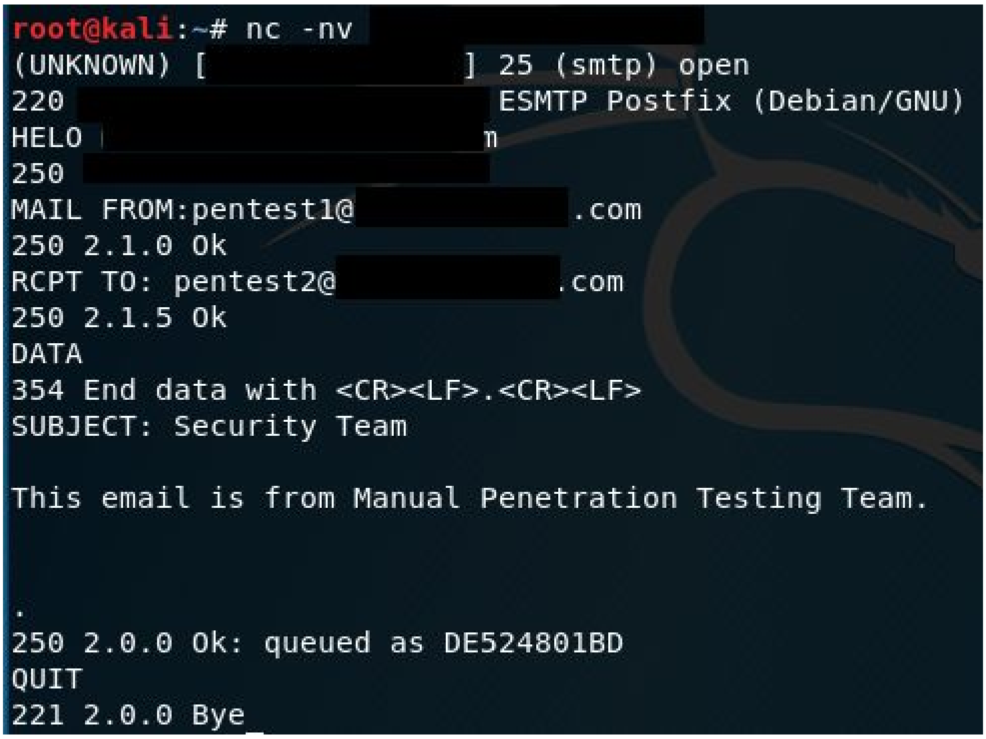
- service1.com
- service2.com
- service3.com
- service4.com
- service5.com

- service6.com
- service7.net

This issue can be used by an attacker to send phishing emails to users and conduct social engineering attacks.

PROOF OF VULNERABILITY:

Attacker has connected to the server without authentication and successfully sent email from pentest1@service1.com to pentest2@service1.com.



```
root@kali:~# nc -nv [redacted]
(UNKNOWN) [[redacted]] 25 (smtp) open
220 [redacted] ESMTP Postfix (Debian/GNU)
HELO [redacted]
250 [redacted]
MAIL FROM:pentest1@[redacted].com
250 2.1.0 Ok
RCPT TO: pentest2@[redacted].com
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
SUBJECT: Security Team

This email is from Manual Penetration Testing Team.


.
250 2.0.0 Ok: queued as DE524801BD
QUIT
221 2.0.0 Bye_
```

RECOMMENDATIONS:

Implement authentication and access control. For example, SMTP authentication requires users to supply a username and password to be able to send mail from the server. Make sure access to your servers is on a need-to-have basis and is shared with as few people as possible.

Password Brute Force Allowed

SEVERITY: **Medium**

LOCATION:

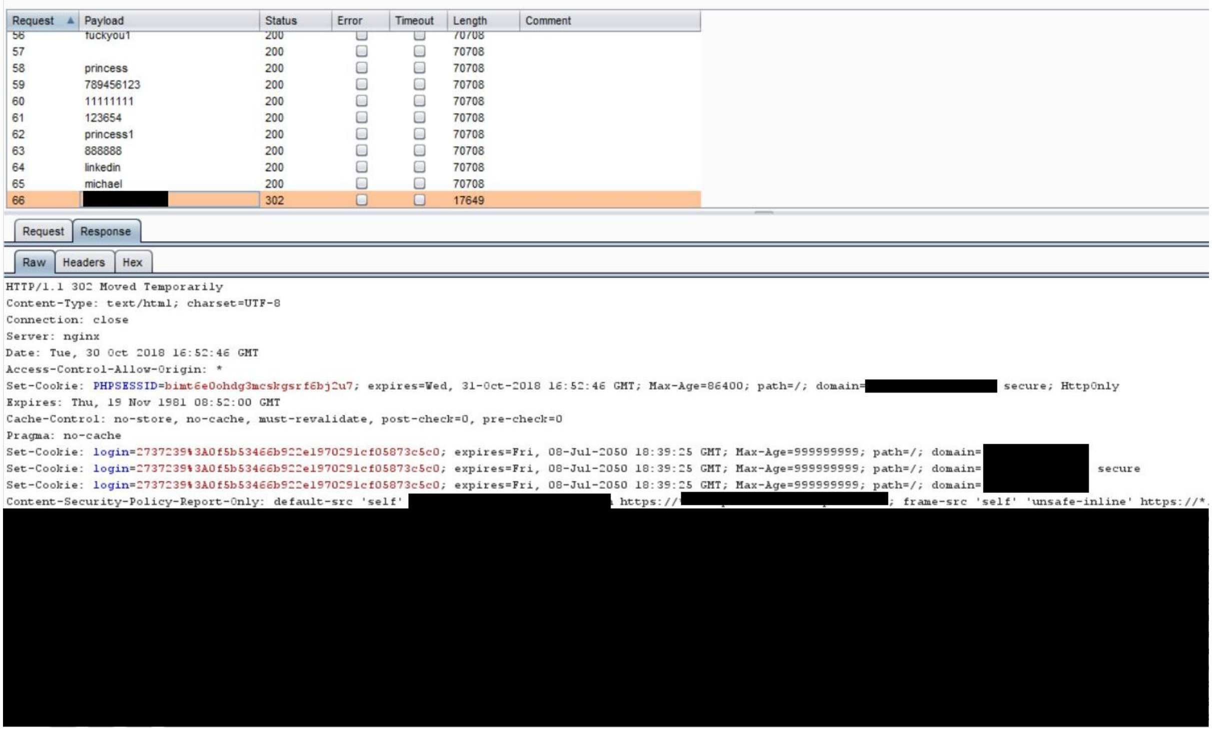
- https://client.com/member/login
- https://service1.com/signin
- https://service2.com
- ssh client_ip:22
- ssh client_ip2:4118

ISSUE DESCRIPTION:

A brute force attack can manifest itself in many different ways, but primarily consists in an attacker configuring predetermined values, making requests to a server using those values, and then analyzing the response. For the sake of efficiency, an attacker may use a dictionary attack (with or without mutations) or a traditional brute-force attack (with given classes of characters e.g.: alphanumerical, special, case (in)sensitive). Considering a given method, number of tries, efficiency of the system which conducts the attack, and estimated efficiency of the system which is attacked the attacker is able to calculate approximately how long it will take to submit all chosen predetermined values.

PROOF OF VULNERABILITY:

User is not locked after 65 attempts of password enumeration and successfully logged in after it.



Request ▲	Payload	Status	Error	Timeout	Length	Comment
56	fuckyou1	200	□	□	70708	
57		200	□	□	70708	
58	princess	200	□	□	70708	
59	789456123	200	□	□	70708	
60	11111111	200	□	□	70708	
61	123654	200	□	□	70708	
62	princess1	200	□	□	70708	
63	888888	200	□	□	70708	
64	linkedin	200	□	□	70708	
65	michael	200	□	□	70708	
66	[redacted]	302	□	□	17649	

```
HTTP/1.1 302 Moved Temporarily
Content-Type: text/html; charset=UTF-8
Connection: close
Server: nginx
Date: Tue, 30 Oct 2018 16:52:46 GMT
Access-Control-Allow-Origin: *
Set-Cookie: PHPSESSID=bimt6e0ohdg3mcskgsrf6bj2u7; expires=Wed, 31-Oct-2018 16:52:46 GMT; Max-Age=86400; path=/; domain=[redacted] secure; HttpOnly
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Set-Cookie: login=2737239%3A0f5b53466b922e1970291cf05873c5c0; expires=Fri, 08-Jul-2050 18:39:25 GMT; Max-Age=999999999; path=/; domain=[redacted]
Set-Cookie: login=2737239%3A0f5b53466b922e1970291cf05873c5c0; expires=Fri, 08-Jul-2050 18:39:25 GMT; Max-Age=999999999; path=/; domain=          secure
Set-Cookie: login=2737239%3A0f5b53466b922e1970291cf05873c5c0; expires=Fri, 08-Jul-2050 18:39:25 GMT; Max-Age=999999999; path=/; domain=
Content-Security-Policy-Report-Only: default-src 'self' [redacted] https://[redacted]; frame-src 'self' 'unsafe-inline' https://*.
[redacted]
```

RECOMMENDATIONS:

There are a number of techniques for preventing brute force attacks:

- account lockout policy;
- progressive delays;
- use a challenge-response test to prevent automated submissions of the login page (CAPTCHA);
- IP address lock-out.

Details on how to prevent this attack you can find here:

- https://www.owasp.org/index.php/Blocking_Brute_Force_Attacks

Session Fixation

SEVERITY: **Medium**

LOCATION:

- https://www.client.com
- https://service1.com

ISSUE DESCRIPTION:

User can use the same session token after logout. Attacker can repeat request with token that should be marked as invalidated.

PROOF OF VULNERABILITY:

Request made after Logout with the same cookie value.

```
curl -i -s -k  -X $'GET' \
                            -H       $'Host:       www.client.com'       -H       $'Cookie:
login=2737239%3A0f5b53466b922e1970291cf05873c5c0;        '        -H       $'X-Forwarded-For:
spoofed.s6zuzg64yc4wqc6vs0nxrfqjmasci86x.burpcollaborator.net' \
    -b $'login=2737239%3A0f5b53466b922e1970291cf05873c5c0' \
    $'https://www.client.com/member/account-preferences'
```

```
curl -i -s -k  -X $'GET' \
    -H $'Host: service1.com' -H $'Cookie: sid=91iqik6qtblp0vsu9b5j7fgal0;' \
    -b $'sid=91iqik6qtblp0vsu9b5j7fgal0' \
    $'https://service1.com/account'
```

RECOMMENDATIONS:

The logout function should be prominently visible to the user, explicitly invalidate a user's session and disallow reuse of the session token. Server should provide new session id to user browser after logout.

Insufficient session expiration

SEVERITY: **Medium**

LOCATION:

- https://www.client.com

ISSUE DESCRIPTION:

Session is active after more than 50 hours of user inactivity. Insufficient session expiration weakness is a result of poorly implemented session management. This weakness can arise on design and implementation levels and can be used by attackers to gain an unauthorized access to the application.

When handling sessions, web developers can rely either on server tokens or generate session identifiers within the application. Each session should be destroyed after the user clicks the Log off button, or after a certain period of time (called timeout). Unfortunately, coding errors and server misconfigurations may influence session handling process, which can result in an unauthorized access.

Session expiration is comprised of two timeout types:

- Inactivity – such timeout is the amount of idle time allowed before the session is invalidated.

- Absolute – such timeout is defined by the total amount of time a session can be valid without re-authentication.

The lack of proper session expiration may increase the likelihood of success of certain attacks. Long expiration time increases an attacker's chance of successfully guessing a valid session ID. The longer the expiration time, the more concurrent open sessions will exist at any given time. The larger the pool of sessions, the more likely it will be for an attacker to guess one at random. Although a short session inactivity timeout does not help if a token is immediately used, the short timeout helps to insure that the token is harder to capture while it is still valid.

RECOMMENDATIONS:

A Web application should invalidate a session after a predefined idle time has passed (a timeout) and provide the user the means to invalidate their own session (log out); this helps to keep the lifespan of a session ID as short as possible and is necessary in a shared computing environment, where more than one person has unrestricted physical access to a computer. More information:

- https://www.owasp.org/index.php/Session_Timeout

Weak Account Preferences Update functionality

SEVERITY: **Medium**

LOCATION:

- https://www.client.com

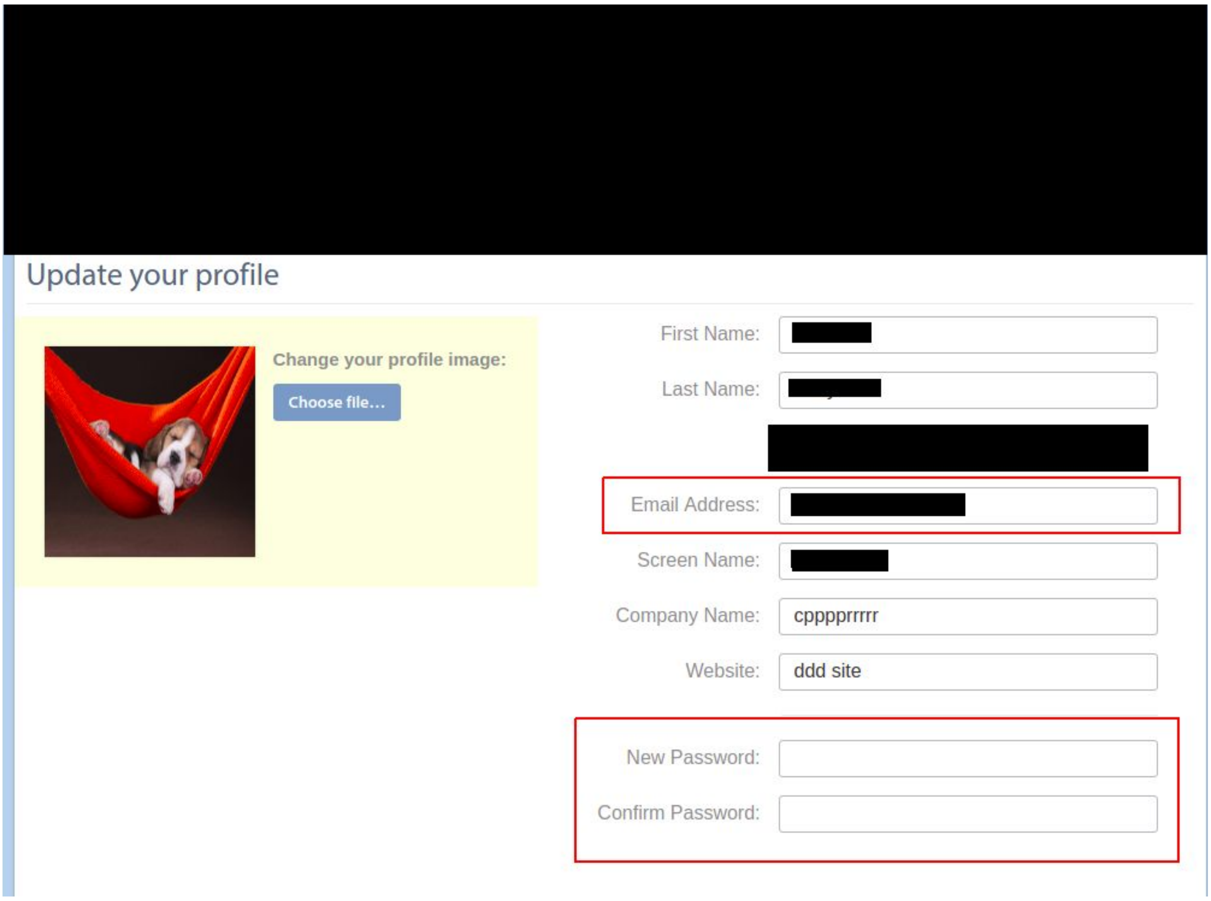
ISSUE DESCRIPTION:

Security-sensitive actions should ask for an additional authentication attempt. Mere logging in to the service should not enable the attacker to perform sensitive actions.

This application allows authenticated user to change password and email to new one without additional security checks and proper validation of account owner. So it is possible for attacker to change password without knowing old one.

PROOF OF VULNERABILITY:

User can update email and password without any authorization whether user is account owner.



[redacted]

Update your profile

Change your profile image:

First Name: [redacted]

Last Name: [redacted]

[redacted]

Email Address: [redacted]

Screen Name: [redacted]

Company Name: cpppprrrr

Website: ddd site

New Password:

Confirm Password:

RECOMMENDATIONS:

Implement additional check for sensitive actions of user. Such as password change and email address update. Most common solution is to ask for additional authentication for account settings change.

The additional authentication step can be:

- Give the password again.
- Email confirmation.
- SMS confirmation.
- Give another two-factor authentication token.

Possible SWEET32 vulnerability

SEVERITY: **Low**

LOCATION:

- deals.client.com
- login.clien.com
- service1.com
- service2.net
- sservice4.com
- wiki.client.com
- jira.client.com

ISSUE DESCRIPTION:

Legacy block ciphers having a block size of 64 bits are vulnerable to a practical collision attack when used in CBC mode. All versions of the SSL/TLS protocols that support cipher suites which use 3DES as the symmetric encryption cipher are affected.

PROOF OF VULNERABILITY:

TestSSL results.

```
 Testing vulnerabilities

...
 POODLE, SSL (CVE-2014-3566)               not vulnerable (OK)
 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)
                                                 make sure you don't use this certificate

elsewhere with SSLv2 enabled services
...
```

RECOMMENDATIONS:

Consider using following guide to secure web server.

REFERENCE:

https://www.openssl.org/blog/blog/2016/08/24/sweet32/

Possible BEAST vulnerability

SEVERITY: **Low**

LOCATION:

- deals.client.com
- login.clien.com
- service1.com
- service2.net
- sservice4.com
- wiki.client.com
- jira.client.com
- chat.client.com
- nf-chat.client.com
- zdt.client.com
- kbill.service1.com
- kbill.service2.com
- account.service3.com
- account.client.com

ISSUE DESCRIPTION:

The SSL protocol, as used in certain configurations in Microsoft Windows and Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Opera, and other products, encrypts data by using CBC mode with chained initialization vectors, which allows man-in-the-middle attackers to obtain plaintext HTTP headers via a blockwise chosen-boundary attack (BCBA) on an HTTPS session, in conjunction with JavaScript code that uses (1) the HTML5 WebSocket API, (2) the Java URLConnection API, or (3) the Silverlight WebClient API, aka a "BEAST" attack.

PROOF OF VULNERABILITY:

TestSSL results.

```
Testing vulnerabilities

...                                         https://censys.io/ipv4?q=123 could help
you to find out
 LOGJAM (CVE-2015-4000), experimental      not vulnerable (OK): no DH EXPORT ciphers, no
DH key detected
```

```
 BEAST (CVE-2011-3389)                    TLS1: ECDHE-RSA-AES128-SHA
                                                ECDHE-RSA-AES256-SHA
                                                AES128-SHA AES256-SHA
                                                 VULNERABLE -- but also supports higher
protocols  TLSv1.1 TLSv1.2 (likely mitigated)
...
```

RECOMMENDATIONS:

Disable TLS 1.0 and have users connect using TLS 1.1 or TLS 1.2 protocols which are immune to the BEAST attack. TLS 1.0 is now considered insecure and disabling the protocol improves the overall security.

REFERENCE:

https://www.acunetix.com/blog/articles/tls-ssl-cipher-hardening

Possible LUCKY13 vulnerability

SEVERITY: **Low**

LOCATION:

- deals.client.com
- login.clien.com
- service1.com
- service2.net
- sservice4.com
- wiki.client.com
- jira.client.com
- chat.client.com
- nf-chat.client.com
- zdt.client.com
- kbill.service1.com
- kbill.service2.com
- account.service3.com
- account.client.com

ISSUE DESCRIPTION:

The TLS protocol 1.1 and 1.2 and the DTLS protocol 1.0 and 1.2, as used in OpenSSL, OpenJDK, PolarSSL, and other products, do not properly consider timing side-channel attacks on a MAC check requirement during the processing of malformed CBC padding, which allows remote attackers to conduct distinguishing attacks and plaintext-recovery attacks via statistical analysis of timing data for crafted packets, aka the "Lucky Thirteen" issue.

PROOF OF VULNERABILITY:

TestSSL results.

```
Testing vulnerabilities
```

```
...                                                 https://censys.io/ipv4?q=123 could help
you to find out
 LOGJAM (CVE-2015-4000), experimental      not vulnerable (OK): no DH EXPORT ciphers, no
DH key detected

 LUCKY13 (CVE-2013-0169), experimental       potentially VULNERABLE, uses cipher block
chaining (CBC) ciphers with TLS. Check patches
 RC4 (CVE-2013-2566, CVE-2015-2808)        no RC4 ciphers detected (OK)
...
```

RECOMMENDATIONS:

Avoid using TLS in CBC-mode and to switch to using AEAD algorithms.

REFERENCE:

https://blog.cloudflare.com/new-ssl-vulnerabilities-cloudflare-users-prot/

Possible RC4 vulnerability

SEVERITY: **Low**

LOCATION:

- login.client.com

ISSUE DESCRIPTION:

The RC4 algorithm, as used in the TLS protocol and SSL protocol, does not properly combine state data with key data during the initialization phase, which makes it easier for remote attackers to conduct plaintext-recovery attacks against the initial bytes of a stream by sniffing network traffic that occasionally relies on keys affected by the Invariance Weakness, and then using a brute-force approach involving LSB values, aka the "Bar Mitzvah" issue.

PROOF OF VULNERABILITY:

Nessus results for login.client.com.

```
List of RC4 cipher suites supported by the remote server :

  High Strength Ciphers (>= 112-bit key)

    RC4-MD5                  Kx=RSA         Au=RSA      Enc=RC4(128)             Mac=MD5
    RC4-SHA                  Kx=RSA         Au=RSA      Enc=RC4(128)             Mac=SHA1

The fields above are :

  {OpenSSL ciphername}
  Kx={key exchange}
  Au={authentication}
  Enc={symmetric encryption method}
  Mac={message authentication code}
  {export flag}
```

RECOMMENDATIONS:

Reconfigure the affected application, if possible, to avoid use of RC4 ciphers. Consider using TLS 1.2 with AES-GCM suites subject to browser and web server support. More information: https://support.microsoft.com/en-us/help/2868725/microsoft-security-advisory-update-for-disabling-rc4

Possible BREACH vulnerability

SEVERITY: **Low**

LOCATION:

- api.service1.com
- api.client.net
- service2.com
- service3.net
- wiki.client.com
- jira.client.com

ISSUE DESCRIPTION:

This web application is potentially vulnerable to the BREACH attack.

An attacker with the ability to:

- Inject partial chosen plaintext into a victim's requests
- Measure the size of encrypted traffic
- can leverage information leaked by compression to recover targeted parts of the plaintext.

BREACH (Browser Reconnaissance & Exfiltration via Adaptive Compression of Hypertext) is a category of vulnerabilities and not a specific instance affecting a specific piece of software.

To be vulnerable, a web application must:

- Be served from a server that uses HTTP-level compression
- Reflect user-input in HTTP response bodies
- Reflect a secret (such as a CSRF token) in HTTP response bodies

PROOF OF VULNERABILITY:

TestSSL results.

```
Testing vulnerabilities

 Heartbleed (CVE-2014-0160)                not vulnerable (OK), no heartbeat extension
 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)                              potentially  NOT  ok,  uses  gzip  HTTP
compression. - only supplied "/" tested
                                                  Can be ignored for static pages or if no
secrets in the page
 POODLE, SSL (CVE-2014-3566)                not vulnerable (OK)
 TLS_FALLBACK_SCSV (RFC 7507)               Downgrade attack prevention supported (OK)
 ...
```

RECOMMENDATIONS:

The mitigations are ordered by effectiveness (not by their practicality - as this may differ from one application to another).

- Disabling HTTP compression
- Separating secrets from user input
- Randomizing secrets per request
- Masking secrets (effectively randomizing by XORing with a random secret per request)
- Protecting vulnerable pages with CSRF
- Length hiding (by adding random number of bytes to the responses)
- Rate-limiting the requests

Possible Secure Client-Initiated Renegotiation

SEVERITY: **Low**

LOCATION:

- servrice1.com
- service2.com
- service3.net

ISSUE DESCRIPTION:

This vulnerability consists of two vectors of attacks:

- CVE-2009-3555: This vulnerability allows a “man-in-the-middle” attacker to inject data into an HTTPS session and execute requests on behalf of the victim. Refer to CVE-2009-3555 for more details.
- Denial of Service (DoS): Establishing a secure SSL connection requires more processing power on the server, around 15 times, than on the client. An attacker can exploit this processing-power property along with renegotiation to trigger hundreds of handshakes in the same TCP connection; an assault can bring down a 30Gb-link server using only a laptop and DSL connection.

PROOF OF VULNERABILITY:

TestSSL results.

```
 Testing vulnerabilities

 Heartbleed (CVE-2014-0160)                not vulnerable (OK), no heartbeat extension
 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)                            potentially  NOT  ok,  uses  gzip  HTTP
compression. - only supplied "/" tested
```

RECOMMENDATIONS:

There are several steps to mitigate the threat of renegotiation attacks:

1. Renegotiation is not required by the majority of sites. Disable SSL renegotiation support on the server.
2. If disabling renegotiation is not possible due to business needs such as ecommerce, then allow only secure renegotiation and limit the number of SSL handshakes, or

upgrade server resources by adding products like an SSL accelerator.
Do not support insecure renegotiation.

3. If disabling renegotiation is not possible due to a legacy server not having the option to disable renegotiation, then limit the number of SSL handshakes, or upgrade server resources by adding products like an SSL accelerator.
4. Information on how to limit the number of SSL handshakes can be found here.
5. Amazon Web Services Elastic Load Balancing does not support disabling client-initiated renegotiation. As an alternative solution, you can use port 443 as TCP rather than HTTPS so that all requests are passed to the server and also disable renegotiation on the server.

User and E-mail Enumeration

SEVERITY: **Low**

LOCATION:

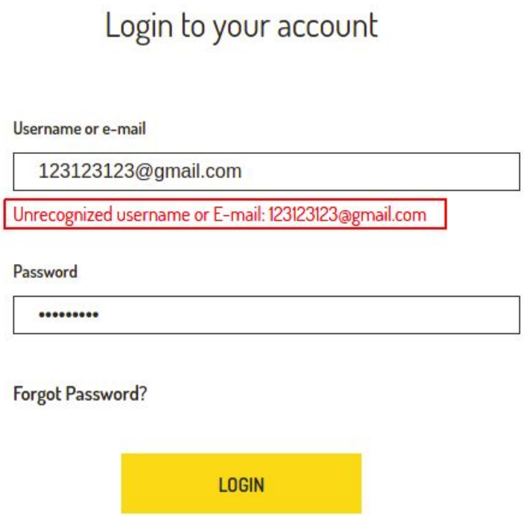
- service1.com
- service2.com

ISSUE DESCRIPTION:

User enumeration is when a malicious actor can use brute-force to either guess or confirm valid users in a system. User enumeration is often a web application vulnerability, though it can also be found in any system that requires user authentication. Two of the most common areas where user enumeration occurs are in a site's login page and its ‘Forgot Password' functionality. We have been able to find user enumeration vulnerability on ‘Login’ and ‘Forgot Password’ functionality which allows attacker to enumerate existing users.

PROOF OF VULNERABILITY:

User enumeration at login page of service1.com :



Login to your account

Username or e-mail

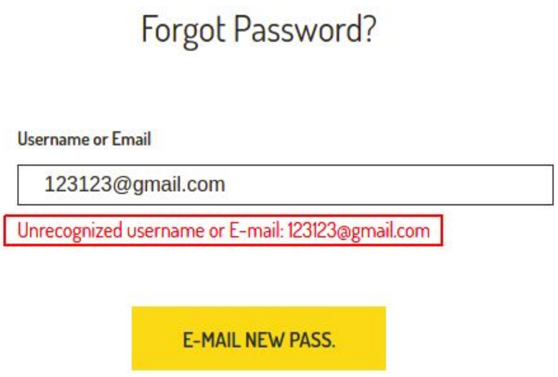
123123123@gmail.com

Unrecognized username or E-mail: 123123123@gmail.com

Password

•••••••••

User enumeration at forgot password page of service2.com:



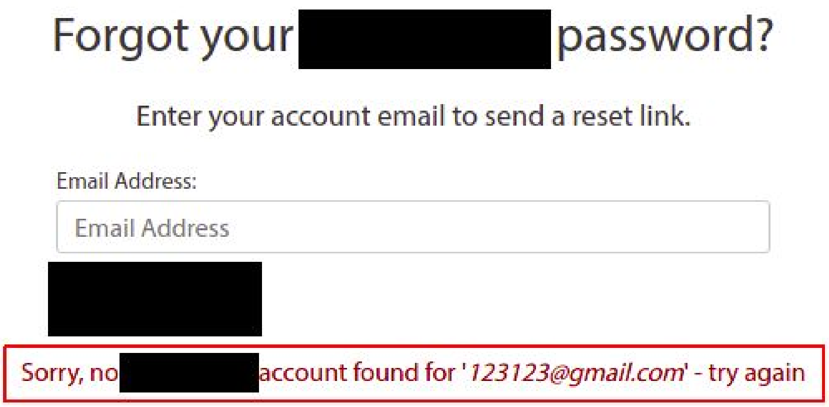
Forgot Password?

Username or Email

123123@gmail.com

Unrecognized username or E-mail: 123123@gmail.com

User enumeration found at forgot password page of service1.com



Forgot your [redacted] password?

Enter your account email to send a reset link.

Email Address:

Email Address

[redacted]

Sorry, no [redacted] account found for '*123123@gmail.com*' - try again

RECOMMENDATIONS:

Provide less verbose responses in the functionality. The website should display the same generic message regardless if the username/email address exists or not. A message such as ‘Further instructions have been sent to your email address’ or similar. For more detailed information please consider using the link below:
https://blog.rapid7.com/2017/06/15/about-user-enumeration/

Apache Tomcat/8.5.6 multiple vulnerabilities

SEVERITY: **Low**

LOCATION:

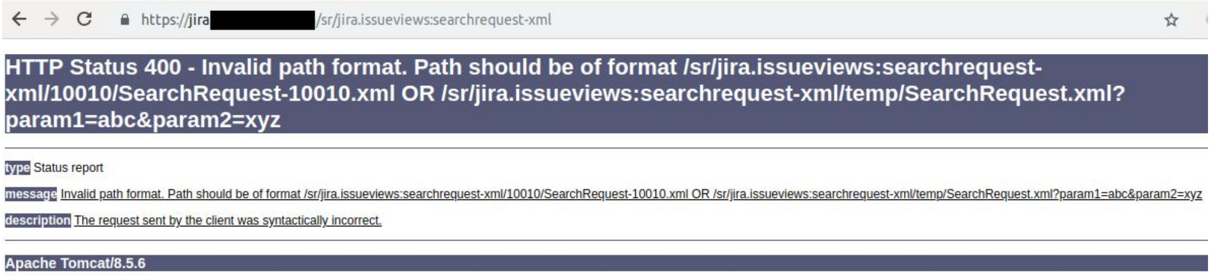
- jira.client.com

ISSUE DESCRIPTION:

It was detected that there is version of Apache Tomcat with number of vulnerabilities. Unpatched known vulnerabilities are a serious risk that may cause direct or non-direct impact on server security.

PROOF OF VULNERABILITY:

Error message that exposes Apache Tomcat version which is vulnerable for several known vulnerabilities.



https://jira[redacted]/sr/jira.issueviews:searchrequest-xml

HTTP Status 400 - Invalid path format. Path should be of format /sr/jira.issueviews:searchrequest-xml/10010/SearchRequest-10010.xml OR /sr/jira.issueviews:searchrequest-xml/temp/SearchRequest.xml?param1=abc&param2=xyz

type Status report

message Invalid path format. Path should be of format /sr/jira.issueviews:searchrequest-xml/10010/SearchRequest-10010.xml OR /sr/jira.issueviews:searchrequest-xml/temp/SearchRequest.xml?param1=abc&param2=xyz

description The request sent by the client was syntactically incorrect.

Apache Tomcat/8.5.6

List of vulnerabilities from CVE database:

- CVE-2018-8037
- CVE-2018-8034
- CVE-2018-8014
- CVE-2018-1336
- CVE-2018-1305
- CVE-2018-1304
- CVE-2017-12617
- CVE-2017-7675
- CVE-2017-7674
- CVE-2017-5664
- CVE-2017-5651
- CVE-2017-5650
- CVE-2017-5648
- CVE-2017-5647
- CVE-2016-8745
- CVE-2016-8735
- CVE-2016-6817
- CVE-2016-6816

RECOMMENDATIONS:

There should be a patch management process in place to:

- Remove unused dependencies, unnecessary features, components, files, and documentation.
- Continuously inventory the versions of both client-side and server-side components (e.g. frameworks, libraries) and their dependencies using tools like versions, DependencyCheck, retire.js, etc. Continuously monitor sources like CVE and NVD for vulnerabilities in the components. Use software composition analysis tools to automate the process. Subscribe to email alerts for security vulnerabilities related to components you use.
- Only obtain components from official sources over secure links. Prefer signed packages to reduce the chance of including a modified, malicious component.
- Monitor for libraries and components that are unmaintained or do not create security patches for older versions. If patching is not possible, consider deploying a virtual patch to monitor, detect, or protect against the discovered issue.

Every organization must ensure that there is an ongoing plan for monitoring, triaging, and applying updates or configuration changes for the lifetime of the application or portfolio.

OpenSSH 6.0p1 multiple vulnerabilities

SEVERITY: **Low**

LOCATION:

- service3.com

ISSUE DESCRIPTION:

It was detected that there is version of OpenSSH with number of vulnerabilities. Unpatched known vulnerabilities are a serious risk that may cause direct or non-direct impact on server security.

PROOF OF VULNERABILITY:

Nmap scan result:

```
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 6.0p1 Debian 4+deb7u7 (protocol 2.0)
| ssh-hostkey:
|   1024 66:08:6b:e9:b8:5c:f2:c6:ce:1a:a3:84:7b:21:d2:7e (DSA)
|   1024 e1:5c:e5:de:e1:78:04:5e:1d:6f:2f:88:f6:79:64:b5 (RSA)
|_  256 b1:9b:1a:60:05:b7:d6:65:df:91:56:c1:ee:a3:9e:32 (ECDSA)
```

List of vulnerabilities from CVE database:

- CVE-2018-15473
- CVE-2017-15906
- CVE-2016-10708
- CVE-2016-0778
- CVE-2016-0777

RECOMMENDATIONS:

There should be a patch management process in place to:

- Remove unused dependencies, unnecessary features, components, files, and documentation.
- Continuously inventory the versions of both client-side and server-side components (e.g. frameworks, libraries) and their dependencies using tools like versions, DependencyCheck, retire.js, etc. Continuously monitor sources like CVE and NVD for vulnerabilities in the components. Use software composition analysis tools to automate the process. Subscribe to email alerts for security vulnerabilities related to components you use.
- Only obtain components from official sources over secure links. Prefer signed packages to reduce the chance of including a modified, malicious component.
- Monitor for libraries and components that are unmaintained or do not create security patches for older versions. If patching is not possible, consider deploying a virtual patch to monitor, detect, or protect against the discovered issue.

Every organization must ensure that there is an ongoing plan for monitoring, triaging, and applying updates or configuration changes for the lifetime of the application or portfolio.

SSH Server CBC Mode Ciphers Enabled

SEVERITY: Low

LOCATION:

- service4.com

ISSUE DESCRIPTION:

CBC Mode Ciphers are enabled on the SSH Server.

PROOF OF VULNERABILITY:

Nessus scan results:

```
The following client-to-server Cipher Block Chaining (CBC) algorithms
are supported :

  3des-cbc
  aes128-cbc
  aes192-cbc
  aes256-cbc
  blowfish-cbc
  cast128-cbc
  rijndael-cbc@lysator.liu.se

The following server-to-client Cipher Block Chaining (CBC) algorithms
are supported :

  3des-cbc
  aes128-cbc
  aes192-cbc
  aes256-cbc
  blowfish-cbc
  cast128-cbc
  rijndael-cbc@lysator.liu.se
```

RECOMMENDATIONS:

Disable CBC Mode Ciphers by modifying /etc/ssh/sshd_config file and use CTR Mode Ciphers.

SSH Weak MAC Algorithms Enabled

SEVERITY: **Low**

LOCATION:

- service3.com

ISSUE DESCRIPTION:

It was detected that the remote SSH server is configured to use Weak MAC Algorithms.

PROOF OF VULNERABILITY:

Nessus scan results:

```
The following client-to-server Message Authentication Code (MAC) algorithms
are supported :

  hmac-md5
  hmac-md5-96
  hmac-sha1-96
  hmac-sha2-256-96
  hmac-sha2-512-96


The following server-to-client Message Authentication Code (MAC) algorithms
```

```
are supported :

  hmac-md5
  hmac-md5-96
  hmac-sha1-96
  hmac-sha2-256-96
  hmac-sha2-512-96
```

RECOMMENDATIONS:

Based on the SSH security checks result you may want to disable these obsolete MAC algorithms. But before that you could check the current allowed algorithms using the command below:

# sshd -T | grep "\(ciphers\|macs\)"

Jira Sensitive Information Leakage

SEVERITY: **Low**

LOCATION:

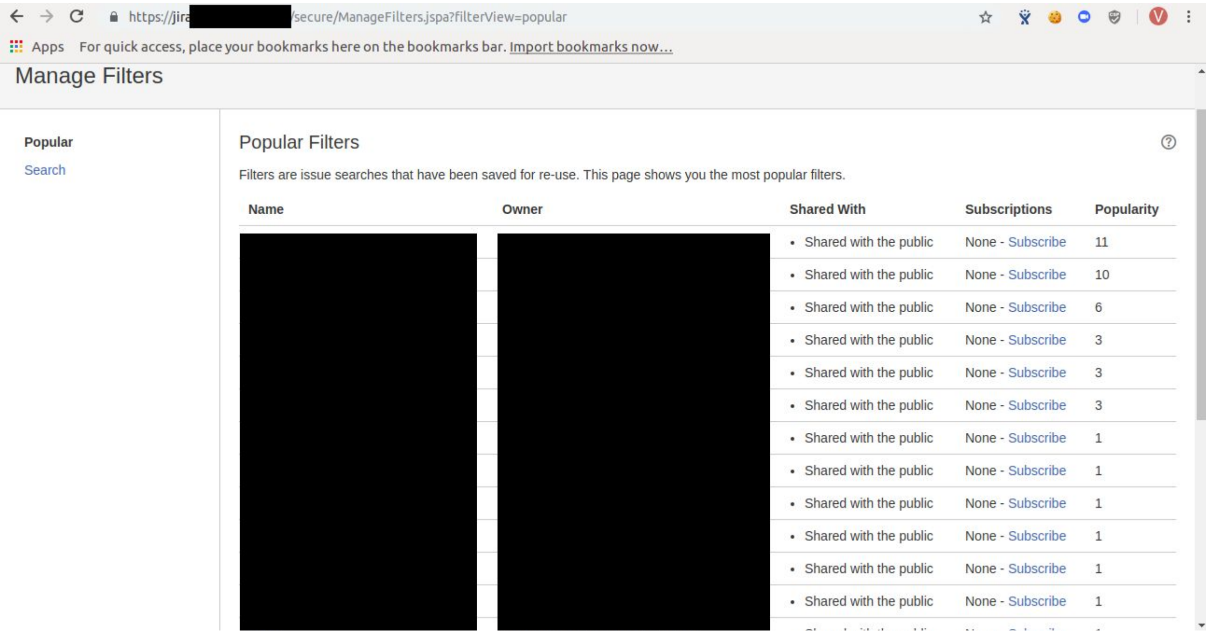
- https://jira.client.com/secure/ManageFilters.jspa?filterView=popular
- https://jira.client.com/secure/ContactAdministrators!custom.jspa

ISSUE DESCRIPTION:

Anonymous user at Jira can access sensitive information like usernames, First and Last names. This information can be used by attacker to make further steps more precise.

PROOF OF VULNERABILITY:

Attacker could access information like Issue names and Owner usernames which are valid and can be used by attacker to perform further steps like brute force or phishing.



https://jira[redacted]/secure/ManageFilters.jspa?filterView=popular

Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now…

Manage Filters

Popular
Search

Popular Filters

Filters are issue searches that have been saved for re-use. This page shows you the most popular filters.

Name	Owner	Shared With	Subscriptions	Popularity
[redacted]	[redacted]	• Shared with the public	None - Subscribe	11
[redacted]	[redacted]	• Shared with the public	None - Subscribe	10
[redacted]	[redacted]	• Shared with the public	None - Subscribe	6
[redacted]	[redacted]	• Shared with the public	None - Subscribe	3
[redacted]	[redacted]	• Shared with the public	None - Subscribe	3
[redacted]	[redacted]	• Shared with the public	None - Subscribe	3
[redacted]	[redacted]	• Shared with the public	None - Subscribe	1
[redacted]	[redacted]	• Shared with the public	None - Subscribe	1
[redacted]	[redacted]	• Shared with the public	None - Subscribe	1
[redacted]	[redacted]	• Shared with the public	None - Subscribe	1
[redacted]	[redacted]	• Shared with the public	None - Subscribe	1
[redacted]	[redacted]	• Shared with the public	None - Subscribe	1

Application error exposes sensitive information like application components and their versions, this information can be useful for attacker to get better understanding of application behavior and perform precise attacks.

```
Technical details
Log's referral number: c620c25c-177b-40d1-9c6c-6799f384e373
Cause
Referer URL: Unknown
java.lang.IllegalArgumentException: No command 'custom' in action
java.lang.IllegalArgumentException: No command 'custom' in action
        at              webwork.action.ActionSupport.invokeCommand(ActionSupport.java:429)
[webwork-1.4-atlassian-30.jar:?]
        at                    webwork.action.ActionSupport.execute(ActionSupport.java:157)
[webwork-1.4-atlassian-30.jar:?]
        at  com.atlassian.jira.action.JiraActionSupport.execute(JiraActionSupport.java:63)
[jira-api-7.3.8.jar:?]
        at
webwork.interceptor.DefaultInterceptorChain.proceed(DefaultInterceptorChain.java:39)
[webwork-1.4-atlassian-30.jar:?]
        at
webwork.interceptor.NestedInterceptorChain.proceed(NestedInterceptorChain.java:31)
[webwork-1.4-atlassian-30.jar:?]
        at    webwork.interceptor.ChainedInterceptor.intercept(ChainedInterceptor.java:16)
[webwork-1.4-atlassian-30.jar:?]
        at
webwork.interceptor.DefaultInterceptorChain.proceed(DefaultInterceptorChain.java:35)
[webwork-1.4-atlassian-30.jar:?]
        at  webwork.dispatcher.GenericDispatcher.executeAction(GenericDispatcher.java:225)
[webwork-1.4-atlassian-30.jar:?]
        at  webwork.dispatcher.GenericDispatcher.executeAction(GenericDispatcher.java:154)
```

```
[webwork-1.4-atlassian-30.jar:?]
        at
com.atlassian.jira.web.dispatcher.JiraWebworkActionDispatcher.service(JiraWebworkActionD
ispatcher.java:147) [classes/:?]
        at                  javax.servlet.http.HttpServlet.service(HttpServlet.java:729)
[servlet-api.jar:?]
        at
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.
java:230) [catalina.jar:8.5.6]
        at
org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:165
) [catalina.jar:8.5.6]
        at         org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
[tomcat-websocket.jar:8.5.6]
        at
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.
java:192) [catalina.jar:8.5.6]
        at
org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:165
) [catalina.jar:8.5.6]
        at
com.atlassian.jira.web.filters.JiraLastFilter.lambda$doFilter$0(JiraLastFilter.java:39)
[classes/:?]
        at
com.atlassian.jira.web.filters.steps.ChainedFilterStepRunner.doFilter(ChainedFilterStepR
unner.java:74) [classes/:?]
        at  com.atlassian.jira.web.filters.JiraLastFilter.doFilter(JiraLastFilter.java:36)
[classes/:?]
        at
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.
java:192) [catalina.jar:8.5.6]
        at
org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:165
) [catalina.jar:8.5.6]
        at
com.atlassian.jira.web.filters.XContentTypeOptionsNoSniffFilter.doFilter(XContentTypeOpt
ionsNoSniffFilter.java:20) [classes/:?]
        at
com.atlassian.core.filters.AbstractHttpFilter.doFilter(AbstractHttpFilter.java:32)
[atlassian-core-5.0.8.jar:?]
        at
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.
java:192) [catalina.jar:8.5.6]
        at
org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:165
) [catalina.jar:8.5.6]
        at
com.atlassian.core.filters.HeaderSanitisingFilter.doFilter(HeaderSanitisingFilter.java:3
7) [atlassian-core-5.0.8.jar:?]
        at
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.
java:192) [catalina.jar:8.5.6]
```

RECOMMENDATIONS:

By removing permission to search issues for ‘anyone’ anonymous user will not have access for this information.

Application should properly handle errors with intention not to leak any sensitive information about application itself. Consider using following links for performing proper error handling:

- https://www.owasp.org/index.php/Error_Handling_Cheat_Sheet
- https://cwe.mitre.org/data/definitions/209.html

SSH Weak Algorithms Supported

SEVERITY: **Low**

LOCATION:

- service4.com

ISSUE DESCRIPTION:

It was detected that the remote SSH server is configured to use the Arcfour stream cipher. RFC 4253 advises against using Arcfour due to an issue with weak keys.

PROOF OF VULNERABILITY:

Nessus scan results:

```
The following weak server-to-client encryption algorithms are supported :

  arcfour
  arcfour128
  arcfour256

The following weak client-to-server encryption algorithms are supported :

  arcfour
  arcfour128
  arcfour256
```

RECOMMENDATIONS:

Based on the SSH security checks result you may want to disable these obsolete encryption algorithms or ciphers. But before that you could check the current allowed ciphers using the command below:

```
# sshd -T | grep "\(ciphers\|macs\)"
```

HSTS missing from HTTPS server

SEVERITY: **Low**

LOCATION:

- https://account.client.com
- https://chat.client.com
- https://www.client.com
- http://wiki.client.com

ISSUE DESCRIPTION:

The remote HTTPS server is not enforcing HTTP Strict Transport Security (HSTS). The lack of HSTS allows downgrade attacks, SSL-stripping man-in-the-middle attacks, and weakens cookie-hijacking protections.

RECOMMENDATIONS:

Configure the remote web server to use HSTS.
More information you can find here:

- https://www.owasp.org/index.php/HTTP_Strict_Transport_Security_Cheat_Sheet

Cookie without Secure and HTTPOnly flags set

SEVERITY: **Low**

LOCATION:

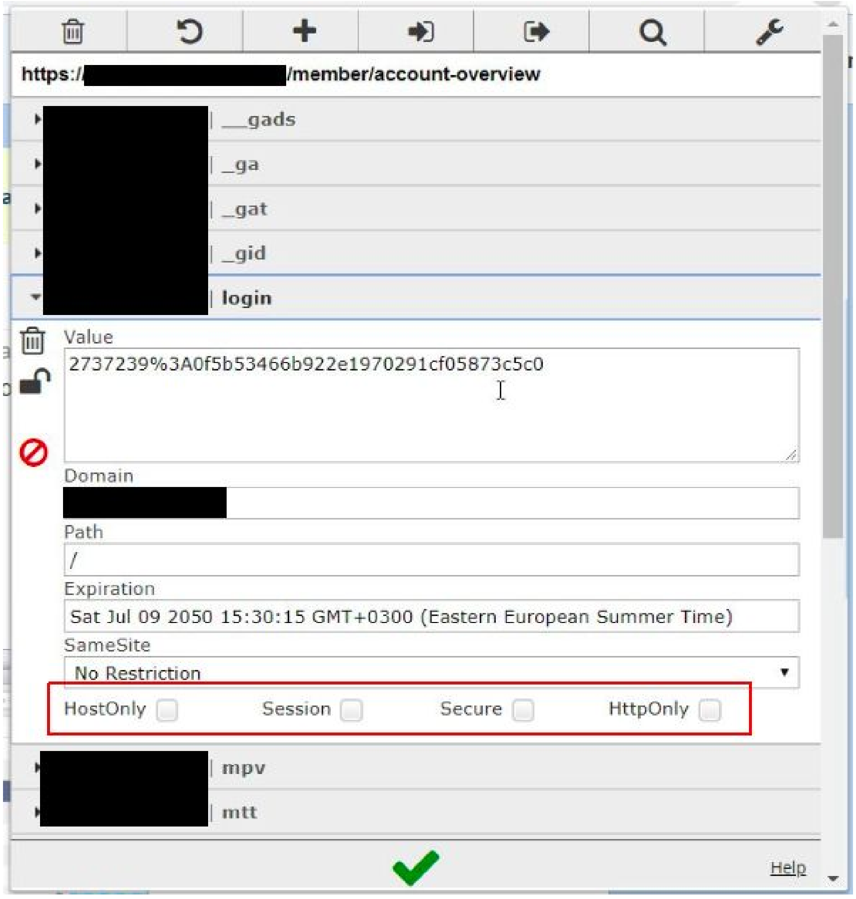
- https://www.client.com
- https://service1.com

ISSUE DESCRIPTION:

Session cookies login and sid are set without Secure and HTTPOnly flag. Secure flag forces browser not to send cookie over insecure channel (use HTTPS instead of HTTP). login and sid cookies are the most critical and the only one that is required to execute requests to a server. According to our testing, the rest cookies are optional, and we did not observe any server-side validation for them. HTTPOnly flag ensures that an attacker cannot steal cookie with Javascript on a client side.

PROOF OF VULNERABILITY:

Session cookies for www.client.com user without Secure and HttpOnly.



https://[redacted]/member/account-overview

[redacted] | __gads
| _ga
| _gat
| _gid
| login

Value
2737239%3A0f5b53466b922e1970291cf05873c5c0

Domain
[redacted]

Path
/

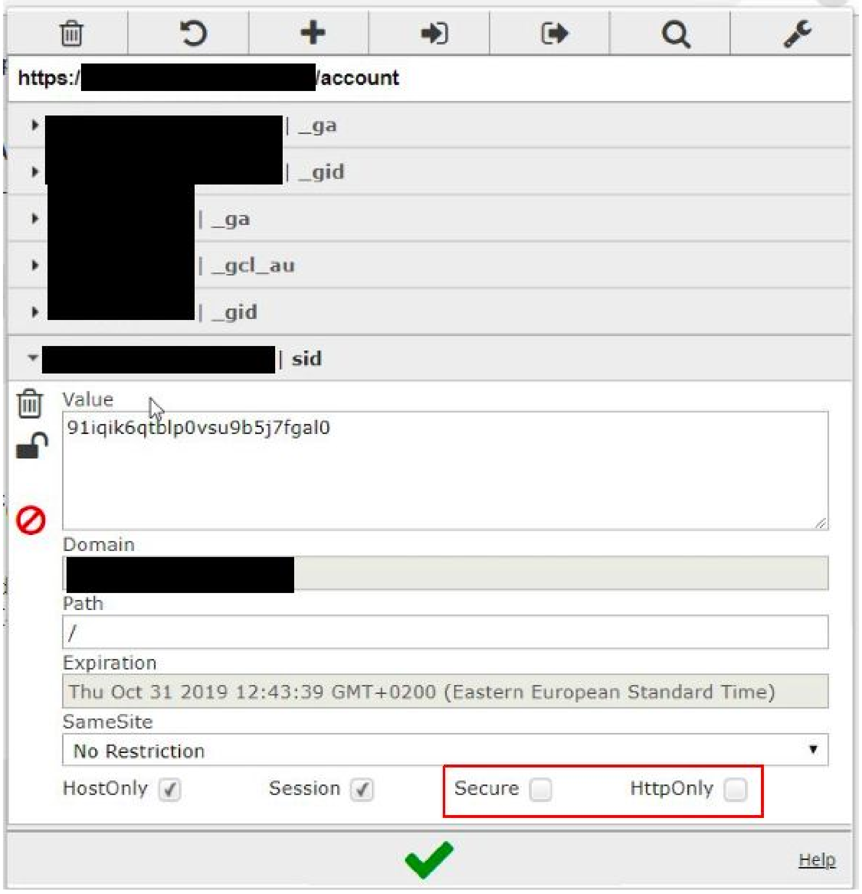
Expiration
Sat Jul 09 2050 15:30:15 GMT+0300 (Eastern European Summer Time)

SameSite
No Restriction ▾

HostOnly □ Session □ Secure □ HttpOnly □

[redacted] | mpv
| mtt

Session cookies for service1.com user without Secure and HttpOnly.



https:/[redacted]/account

[redacted] | _ga
| _gid
[redacted] | _ga
| _gcl_au
| _gid
[redacted] | sid

Value: 91iqik6qtblp0vsu9b5j7fgal0

Domain: [redacted]

Path: /

Expiration: Thu Oct 31 2019 12:43:39 GMT+0200 (Eastern European Standard Time)

SameSite: No Restriction

HostOnly ☑ Session ☑ Secure ☐ HttpOnly ☐

RECOMMENDATIONS:

Ensure that Web Server sets Secure and HttpOnly flags on session cookies.

Details on HTTPOnly and Secure flags configuration:

- https://www.owasp.org/index.php/HttpOnly
- https://www.owasp.org/index.php/SecureFlag

Insecure Software Version

SEVERITY: **Low**

LOCATION:

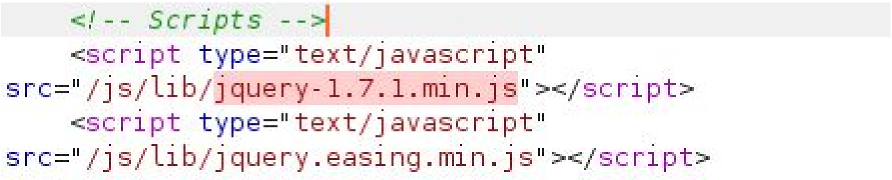
- https://service1.com/
- https://service3.com/vendor/bootstrap/js/bootstrap.min.js
- https://service4.com/vendor/jquery/jquery-2.2.4.min.js
- https://wiki.client.com/s/202a4eb254bd339a734425f2367bc-CDN/en_GB/8201/7d89a43db36102ceb73a83aa0af8aa0a285a/84659ac4c1ebd2d0be562d3fdacb4d/_/download/contextbatch/js/_super/batch.js

ISSUE DESCRIPTION:

When new vulnerabilities are discovered in software, it is important to apply patches and update to a version of the software for which the vulnerability is fixed. Attackers can use known vulnerabilities in their purposes, so security patches should be deployed as soon as they are available.

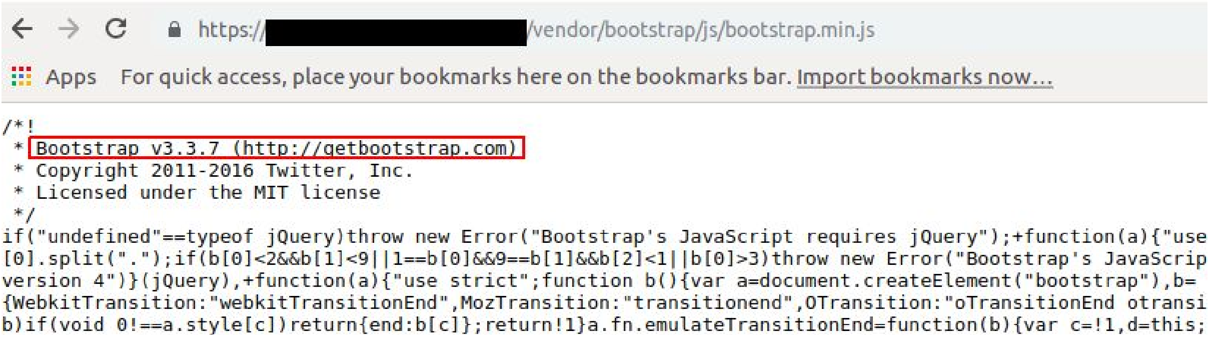
PROOF OF VULNERABILITY:

Vulnerable js lib in serevice1.com:



```
    <!-- Scripts -->
    <script type="text/javascript"
src="/js/lib/jquery-1.7.1.min.js"></script>
    <script type="text/javascript"
src="/js/lib/jquery.easing.min.js"></script>
```

Vulnerable js lib in service3.com:

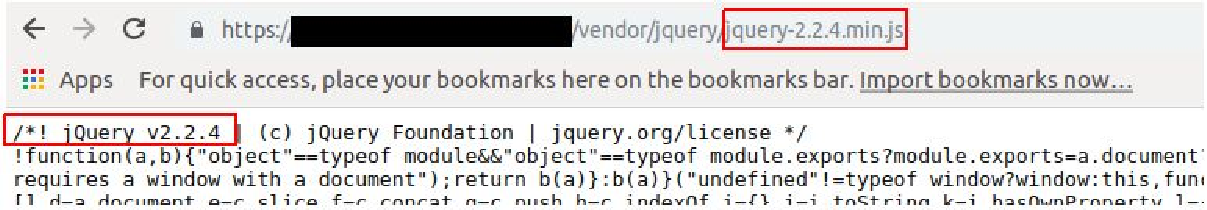


https://[redacted]/vendor/bootstrap/js/bootstrap.min.js

Apps
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

```
/*!
 * Bootstrap v3.3.7 (http://getbootstrap.com)
 * Copyright 2011-2016 Twitter, Inc.
 * Licensed under the MIT license
 */
if("undefined"==typeof jQuery)throw new Error("Bootstrap's JavaScript requires jQuery");+function(a){"use
[0].split(".");if(b[0]<2&&b[1]<9||1==b[0]&&9==b[1]&&b[2]<1||b[0]>3)throw new Error("Bootstrap's JavaScrip
version 4")}(jQuery),+function(a){"use strict";function b(){var a=document.createElement("bootstrap"),b=
{WebkitTransition:"webkitTransitionEnd",MozTransition:"transitionend",OTransition:"oTransitionEnd otransi
b)if(void 0!==a.style[c])return{end:b[c]};return!1}a.fn.emulateTransitionEnd=function(b){var c=!1,d=this;
```

Vulnerable js lib in service4.com:

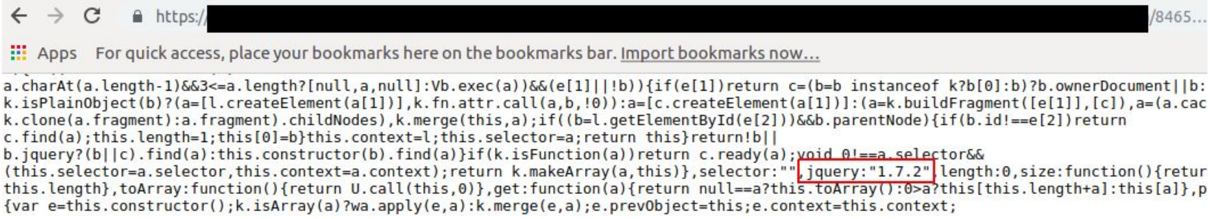


https://[redacted]/vendor/jquery/jquery-2.2.4.min.js

Apps
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

```
/*! jQuery v2.2.4 | (c) jQuery Foundation | jquery.org/license */
!function(a,b){"object"==typeof module&&"object"==typeof module.exports?module.exports=a.document
requires a window with a document");return b(a)}:b(a)}("undefined"!=typeof window?window:this,fun
[],d=a.document,e=c.slice,f=c.concat,g=c.push,h=c.indexOf,i={},j=i.toString,k=i.hasOwnProperty,l=
```

Vulnerable js lib in wiki.client.com:



https:/[redacted]/8465...
Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

```
a.charAt(a.length-1)&&3<=a.length?[null,a,null]:Vb.exec(a))&&(e[1]||!b)){if(e[1])return c=(b=b instanceof k?b[0]:b)?b.ownerDocument||b:
k.isPlainObject(b)?(a=[l.createElement(a[1])],k.fn.attr.call(a,b,!0)):a=[c.createElement(a[1])]:(a=k.buildFragment([e[1]],[c]),a=(a.cac
k.clone(a.fragment):a.fragment).childNodes),k.merge(this,a);if((b=l.getElementById(e[2]))&&b.parentNode){if(b.id!==e[2])return
c.find(a);this.length=1;this[0]=b}this.context=l;this.selector=a;return this}return!b||
b.jquery?(b||c).find(a):this.constructor(b).find(a)}if(k.isFunction(a))return c.ready(a);void 0!==a.selector&&
(this.selector=a.selector,this.context=a.context);return k.makeArray(a,this)},selector:"",jquery:"1.7.2",length:0,size:function(){retur
this.length},toArray:function(){return U.call(this,0)},get:function(a){return null==a?this.toArray():0>a?this[this.length+a]:this[a]},p
{var e=this.constructor();k.isArray(a)?wa.apply(e,a):k.merge(e,a);e.prevObject=this;e.context=this.context;
```

RECOMMENDATIONS:

Update outdated software and always keep it up-to-date.

Leaked data on several exposed databases

SEVERITY: **Informational**

ISSUE DESCRIPTION:

The definition of data leakage is the unauthorized transmission of data from within an organization to an external destination or recipient. Data leakage threats usually occur via the web and email, but can also occur via mobile data storage devices such as optical media, USB keys, and laptops. There are a lot of possible threats which can lead to sensitive data exposure like breach of Third-Party service providers or phishing attack.

We have found leaked information including corporate emails exposed on several data breaches.

PROOF OF VULNERABILITY:

List of exposed mails and where it found:

Mails	Leaked Database
user1@client.com	Apollo
user2@service3.com	Apollo
user3@service1.com	Apollo
user3@client.net	Apollo

support@client.net	Onliner Spambot
user4@service2.com	River City Media Spam List
support@service2.com	Onliner Spambot

RECOMMENDATIONS:

Employees should be aware of the risks of reusing corporate email and passwords for non-work related purposes. In case there is need for creating account on third party services using corporate emails, employees should create new password for each service. We recommend to use password managers.

APPENDIX A – Scope

DNS Name/ IP Address
LIST OF IP AND URL ADDRESSES

APPENDIX B – List of targets and services

DNS Name	IP	Open Ports and Services
DNS NAMES LIST	IP ADDRESS	PORT & SERVICE
	IP ADDRESS	PORT & SERVICE

APPENDIX C – List of performed tests

Security risks	Description	Status
Host and service enumeration	Network enumeration is the process of extracting services its versions, machine names, network resources, and other information from a system. All the gathered information can be used by attackers to identify the vulnerabilities or weak points in system security and then focus on its exploitation.	**Fails criteria** Some tested hosts showed services versions, opened ports which can be easily identified by network scanners.
Weak passwords attack and brute-force	Login pages without any defense can lead to users enumeration and breaking clients passwords. Weak password requirements allow users to create weak passwords which cause possibility to compromise them by using default or custom dictionary of passwords.	**Fails criteria** Web application does not implement sufficient measures to prevent multiple failed authentication attempts within in a short time frame, making it more

		susceptible to brute force attacks.
Identification of misconfigurations	Security misconfiguration can happen at any level of an application stack, including the platform, web server, application server, database, framework, and custom code. This could range from failing to set a useful security header on a web server, to forgetting to disable default platform functionality that could grant administrative access to an attacker.	**Fails criteria** The software generates an error message that includes sensitive information about its environment, users, and associated data.
Vulnerability identification and system exploitation	It is the possibility of identification of known vulnerabilities and their further exploitation.	**Fails criteria** It was possible to enumerate several known vulnerabilities such as: OpenSSH user enumeration, SMTP without authentication and Reflected XSS, which were successfully exploited.
Search Engine Discovery and Reconnaissance for Information Leakage	The easiest way to get sensitive information about service and its users is to google. Attackers may get sensitive information from Google using advanced search terms that help users to search the index of a specific website, specific file type and some interesting information from unsecured Websites. Such technics can uncover sensitive information such as email addresses and lists, login credentials, sensitive files, website vulnerabilities, and even financial information (e.g. payment card data)	**Fails criteria** Users emails and passwords was leaked in data breaches.
Weak Authorization Mechanisms testing	Assuming a user with a given identity, authorization is the process of determining whether that user can access a given resource, based on the user's privileges and any permissions or other access-control specifications that apply to the resource. When access control checks are not applied consistently – or not at all – users	**Meets criteria** No issues with this type of risks were identified.

	are able to access data or perform actions that they should not be allowed to perform. This can lead to a wide range of problems, including information exposures, denial of service, and arbitrary code execution.	
Database compromising, sensitive information stealing	Some web applications do not properly protect sensitive data, such as credit cards, tax IDs, and authentication credentials. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data deserves extra protection such as encryption at rest or in transit, as well as special precautions when exchanged with the browser.	**Meets criteria** No issues with this type of risks were identified.
Outdated services	Unpatched services provide possibility for crimes to use security holes and compromise the entire system, steal users data or denial of service. If your technology is not always up to date, your risk is constantly increasing at exponential rates.	**Fails criteria** Vulnerable version of Openssh and Apache Tomcat with several known vulnerabilities were found.
S3 bucket enumeration	Amazon Web Services (AWS) provides some of the most powerful and robust infrastructure for modern web applications. As with all new functionality on the web, new security considerations inevitably arise. One of common vulnerabilities are misconfigured bucket access which granted permission to read information.	**Meets criteria** No issues with this type of risks were identified.

Security risks	Description	Status
A1:2017-Injection	Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.	**Meets criteria** No issues with this type of risks were identified.
A2:2017-Broken Authentication	Application functions related to authentication and session management are often implemented incorrectly,	**Fails criteria**

	allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.	User can use the same session token after logout.
A3:2017-Sensitive Data Exposure	Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.	**Fails criteria** Anonymous user at Jira can access sensitive information like usernames.
A4:2017-XL External Entities (XXE)	Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.	**Meets criteria** No issues with this type of risks were identified.
A5:2017-Broken Access Control	Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.	**Meets criteria** No issues with this type of risks were identified.
A6:2017-Security Misconfiguration	Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.	**Fails criteria** The software generates an error message that includes sensitive information about its environment, users, and associated data.
A7:2017-Cross-Site Scripting (XSS)	XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with	**Fails criteria** Reflected XSS was found in web page. These weakness

	user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.	does not require a lot of technical skills and could be used to steal users sensitive data.
A8:2017-Insecure Deserialization	Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.	**Meets criteria** No issues with this type of risks were identified.
A9:2017-Using Components with Known Vulnerabilities	Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.	**Fails criteria** Vulnerable version of Openssh and Apache Tomcat were found.
A10:2017-Insufficient Logging & Monitoring	Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.	**N/A**

APPENDIX D – Risk Rating Methodology

Our risk rating methodology is based on The OWASP Risk Rating Methodology:

- https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology

OWASP Risk Assessment Calculator can be found here:

- https://www.security-net.biz/files/owaspriskcalc.html

Reflected Cross Site Scripting – High

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	High reward [9]
Opportunity	Special access or resources required [4]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Automated tools available [9]
Ease of Exploit	Difficult [3]
Awareness	Public knowledge [9]
Intrusion Detection	Logged and reviewed [3]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Extensive seriously corrupt data [7]
Loss of Availability	Extensive primary services interrupted [7]
Loss of Accountability	Attack completely anonymous [9]
Business Impact Factors	
Financial damage	Significant effect on annual profit [7]

Reputation damage	Brand damage [9]
Non-Compliance	High profile violation [7]
Privacy violation	Hundreds of people [5]

SMTP Server without authentication – Medium

Likelihood factors	
Threat Agent Factors	
Skills required	Some technical skills [6]
Motive	Possible reward [4]
Opportunity	No access or resources required [9]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Automated tools available [9]
Ease of Exploit	Easy [5]
Awareness	Public knowledge [9]
Intrusion Detection	Logged and reviewed [3]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Not Applicable [0]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Attack possibly traceable to individual [7]
Business Impact Factors	
Financial damage	Minor effect on annual profit [3]
Reputation damage	Loss of goodwill [5]
Non-Compliance	Minor violation [2]

Privacy violation	Not Applicable [0]

Password Brute Force – Medium

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Possible reward [4]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Difficult [3]
Ease of Exploit	Theoretical [1]
Awareness	Obvious [6]
Intrusion Detection	Active detection in application [1]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Extensive seriously corrupt data [7]
Loss of Availability	Extensive primary services interrupted [7]
Loss of Accountability	Attack completely anonymous [9]
Business Impact Factors	
Financial damage	Significant effect on annual profit [7]
Reputation damage	Brand damage [9]
Non-Compliance	High profile violation [7]
Privacy violation	Hundreds of people [5]

Session Fixation – Medium

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Possible reward [4]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Difficult [3]
Ease of Exploit	Theoretical [1]
Awareness	Obvious [6]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Extensive seriously corrupt data [7]
Loss of Availability	Extensive primary services interrupted [7]
Loss of Accountability	Attack completely anonymous [9]
Business Impact Factors	
Financial damage	Significant effect on annual profit [7]
Reputation damage	Brand damage [9]
Non-Compliance	High profile violation [7]
Privacy violation	Hundreds of people [5]

Insufficient session expiration – Medium

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Possible reward [4]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Difficult [3]
Ease of Exploit	Theoretical [1]
Awareness	Obvious [6]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Extensive seriously corrupt data [7]
Loss of Availability	Extensive primary services interrupted [7]
Loss of Accountability	Attack completely anonymous [9]
Business Impact Factors	
Financial damage	Significant effect on annual profit [7]
Reputation damage	Brand damage [9]
Non-Compliance	High profile violation [7]
Privacy violation	Hundreds of people [5]

Weak Account Preferences Update functionality – Medium

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Possible reward [4]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Difficult [3]
Ease of Exploit	Theoretical [1]
Awareness	Obvious [6]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Extensive seriously corrupt data [7]
Loss of Availability	Extensive primary services interrupted [7]
Loss of Accountability	Attack completely anonymous [9]
Business Impact Factors	
Financial damage	Significant effect on annual profit [7]
Reputation damage	Brand damage [9]
Non-Compliance	High profile violation [7]
Privacy violation	Hundreds of people [5]

Possible SWEET32 vulnerability – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Practically impossible [1]
Ease of Exploit	Theoretical [1]
Awareness	Hidden [4]
Intrusion Detection	Logged and reviewed [3]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Attack possibly traceable to individual [7]
Business Impact Factors	
Financial damage	Significant effect on annual profit [7]
Reputation damage	Brand damage [9]
Non-Compliance	Clear violation [5]
Privacy violation	One individual [3]

Possible BEAST vulnerability – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Practically impossible [1]
Ease of Exploit	Theoretical [1]
Awareness	Hidden [4]
Intrusion Detection	Logged and reviewed [3]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Attack possibly traceable to individual [7]
Business Impact Factors	
Financial damage	Significant effect on annual profit [7]
Reputation damage	Brand damage [9]
Non-Compliance	Clear violation [5]
Privacy violation	One individual [3]

Possible LUCKY13 vulnerability – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Practically impossible [1]
Ease of Exploit	Theoretical [1]
Awareness	Hidden [4]
Intrusion Detection	Logged and reviewed [3]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Attack possibly traceable to individual [7]
Business Impact Factors	
Financial damage	Significant effect on annual profit [7]
Reputation damage	Brand damage [9]
Non-Compliance	Clear violation [5]
Privacy violation	One individual [3]

Possible RC4 vulnerability – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Practically impossible [1]
Ease of Exploit	Theoretical [1]
Awareness	Hidden [4]
Intrusion Detection	Logged and reviewed [3]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Attack possibly traceable to individual [7]
Business Impact Factors	
Financial damage	Significant effect on annual profit [7]
Reputation damage	Brand damage [9]
Non-Compliance	Clear violation [5]
Privacy violation	One individual [3]

Possible BREACH vulnerability – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Practically impossible [1]
Ease of Exploit	Theoretical [1]
Awareness	Hidden [4]
Intrusion Detection	Logged and reviewed [3]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Attack possibly traceable to individual [7]
Business Impact Factors	
Financial damage	Significant effect on annual profit [7]
Reputation damage	Brand damage [9]
Non-Compliance	Clear violation [5]
Privacy violation	One individual [3]

Possible Secure Client-Initiated Renegotiation - Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Practically impossible [1]
Ease of Exploit	Theoretical [1]
Awareness	Hidden [4]
Intrusion Detection	Logged and reviewed [3]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Not Applicable [0]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Minimal primary services interrupted [5]
Loss of Accountability	Attack possibly traceable to individual [7]
Business Impact Factors	
Financial damage	Minor effect on annual profit [3]
Reputation damage	Minimal damage [1]
Non-Compliance	Not Applicable [0]
Privacy violation	Not Applicable [0]

User and E-mail Enumeration – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Special access or resources required [4]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Easy [7]
Ease of Exploit	Difficult [3]
Awareness	Obvious [6]
Intrusion Detection	Logged and reviewed [3]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Minimal non-sensitive data disclosed [2]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Attack fully traceable to individual [1]
Business Impact Factors	
Financial damage	Damage costs less than to fix the issue [1]
Reputation damage	Minimal damage [1]
Non-Compliance	Not Applicable [0]
Privacy violation	One individual [3]

Apache Tomcat/8.5.6 multiple vulnerabilities – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Not Applicable [0]
Vulnerability Factors	
Easy of Discovery	Not Applicable [0]
Ease of Exploit	Not Applicable [0]
Awareness	Public knowledge [9]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Not Applicable [0]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Not Applicable [0]
Business Impact Factors	
Financial damage	Not Applicable [0]
Reputation damage	Not Applicable [0]
Non-Compliance	Not Applicable [0]
Privacy violation	Not Applicable [0]

OpenSSH 6.0p1 multiple vulnerabilities – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Not Applicable [0]
Vulnerability Factors	
Easy of Discovery	Not Applicable [0]
Ease of Exploit	Not Applicable [0]
Awareness	Public knowledge [9]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Not Applicable [0]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Not Applicable [0]
Business Impact Factors	
Financial damage	Not Applicable [0]
Reputation damage	Not Applicable [0]
Non-Compliance	Not Applicable [0]
Privacy violation	Not Applicable [0]

SSH Server CBC Mode Ciphers Enabled – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Practically impossible [1]
Ease of Exploit	Theoretical [1]
Awareness	Obvious [6]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Not Applicable [0]
Business Impact Factors	
Financial damage	Minor effect on annual profit [3]
Reputation damage	Minimal damage [1]
Non-Compliance	Minor violation [2]
Privacy violation	One individual [3]

SSH Weak MAC Algorithms Enabled – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Practically impossible [1]
Ease of Exploit	Theoretical [1]
Awareness	Obvious [6]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Not Applicable [0]
Business Impact Factors	
Financial damage	Minor effect on annual profit [3]
Reputation damage	Minimal damage [1]
Non-Compliance	Minor violation [2]
Privacy violation	One individual [3]

Jira Sensitive Information Leakage – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Advanced computer user [4]
Motive	Low or no reward [1]
Opportunity	Some access or resources required [7]
Population Size	Intranet Users [4]
Vulnerability Factors	
Easy of Discovery	Easy [7]
Ease of Exploit	Theoretical [1]
Awareness	Hidden [4]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Not Applicable [0]
Business Impact Factors	
Financial damage	Minor effect on annual profit [3]
Reputation damage	Minimal damage [1]
Non-Compliance	Minor violation [2]
Privacy violation	Hundreds of people [5]

SSH Weak Algorithms Supported – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Practically impossible [1]
Ease of Exploit	Theoretical [1]
Awareness	Obvious [6]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Not Applicable [0]
Business Impact Factors	
Financial damage	Minor effect on annual profit [3]
Reputation damage	Minimal damage [1]
Non-Compliance	Minor violation [2]
Privacy violation	One individual [3]

HSTS missing from HTTPS server – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Difficult [3]
Ease of Exploit	Theoretical [1]
Awareness	Obvious [6]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Not Applicable [0]
Business Impact Factors	
Financial damage	Minor effect on annual profit [3]
Reputation damage	Loss of goodwill [5]
Non-Compliance	Minor violation [2]
Privacy violation	Hundreds of people [5]

Cookie without Secure and HTTPOnly flags set – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Authenticated users [6]
Vulnerability Factors	
Easy of Discovery	Difficult [3]
Ease of Exploit	Difficult [3]
Awareness	Obvious [6]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Extensive critical data disclosed [7]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Not Applicable [0]
Business Impact Factors	
Financial damage	Minor effect on annual profit [3]
Reputation damage	Loss of goodwill [5]
Non-Compliance	Minor violation [2]
Privacy violation	Hundreds of people [5]

Insecure Software Version – Low

Likelihood factors	
Threat Agent Factors	
Skills required	Security penetration skills [1]
Motive	Low or no reward [1]
Opportunity	Full access or expensive resources required [0]
Population Size	Not Applicable [0]
Vulnerability Factors	
Easy of Discovery	Not Applicable [0]
Ease of Exploit	Not Applicable [0]
Awareness	Public knowledge [9]
Intrusion Detection	Not Applicable [0]
Impact factors	
Technical Impact Factors	
Loss of confidentiality	Not Applicable [0]
Loss of Integrity	Not Applicable [0]
Loss of Availability	Not Applicable [0]
Loss of Accountability	Not Applicable [0]
Business Impact Factors	
Financial damage	Not Applicable [0]
Reputation damage	Not Applicable [0]
Non-Compliance	Not Applicable [0]
Privacy violation	Not Applicable [0]