[REDACTED]
Security Assessment Findings Report

Business Confidential

Date: [REDACTED], 2020
Project: Application Assessment
Version 1.0

Table of Contents

Confidentiality Statement

This document is the exclusive property of [REDACTED]. This document contains proprietary and confidential information. Duplication, redistribution, or use, in whole or in part, in any form, requires the consent of [REDACTED] or Joshua Mol.

[REDACTED] may share this document with auditors under non-disclosure agreements to demonstrate penetration test requirement compliance. Joshua Mol also retains the right of duplication and distribution of this document.

Disclaimer

A penetration test is considered a snapshot in time. The findings and recommendations reflect the information gathered during the assessment and not any changes or modifications made outside of that period.

Time-limited engagements do not allow for a full evaluation of all security controls. I have prioritized the assessment to identify the weakest security controls an attacker would exploit. I recommend conducting similar assessments on an annual basis by internal or third-party assessors to ensure the continued success of the controls.

Contact Information

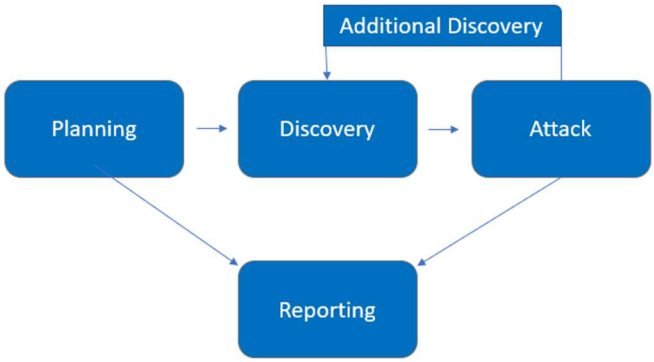
Name	Title	Contact Information
Company		
[REDACTED]	VP, Information Security (CISO)	Office: [REDACTED] Email: [REDACTED]@[REDACTED].com
[REDACTED]	Talent Acquisition, Advisor	Office: [REDACTED] Email: [REDACTED]@[REDACTED].com
[REDACTED]	Talent Acquisition Coordinator	Office: [REDACTED] Email: [REDACTED]@[REDACTED].com
Security Tester		
Joshua Mol	Penetration Tester	Office: [REDACTED] Email: DevR4ndom@gmail.com

Assessment Overview

From June [REDACTED], 2020 to June [REDACTED], 2020, Joshua Mol engaged [REDACTED] to evaluate the security posture of its infrastructure compared to current industry best practices that included an external penetration test. All testing performed is based on the NIST *SP 800-115 Technical Guide to Information Security Testing and Assessment, OWASP Testing Guide (v4), and customized testing frameworks*.

Phases of penetration testing activities include the following:

- Planning – Customer goals are gathered and rules of engagement obtained.
- Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
- Attack – Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.
- Reporting – Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.



Assessment Components

External Penetration Test

An external penetration test emulates the role of an attacker attempting to gain access to an internal network without internal resources or inside knowledge. I normally would have made attempts to gather sensitive information through open-source intelligence (OSINT), including employee information, although due to request by [REDACTED] and the scope of the project these tests were excluded from this scenario. Instead I have focused on historical breached passwords, and web application vulnerabilities that can be leveraged against external systems to gain internal network access. Scanning and enumeration were also conducted to identify potential vulnerabilities within the infrastructure of [REDACTED] regarding exploitation and manipulation of target.

Finding Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

Severity	CVSS V3 Score Range	Definition
Critical	9.0-10.0	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.
High	7.0-8.9	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Moderate	4.0-6.9	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1-3.9	Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Informational	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

Scope

Assessment	Details
External Penetration Test	[REDACTED] ([REDACTED])

- Full scope information provided in "**[REDACTED].xslx**"

Scope Exclusions

Per client request, Joshua Mol did not perform any activity outside of the specified scope, nor social engineering attacks during testing.

Client Allowances

[REDACTED] has communicated that the following allowances would be allowed during the testing period, manual and automated tools and techniques, credential compromise and reuse, privilege escalation attacks. No further allowances were communicated.

Executive Summary

Joshua Mol evaluated [REDACTED] external security posture through an external web application penetration test from June [REDACTED], 2020 to June [REDACTED], 2020. By leveraging a series of attacks, Joshua Mol found critical level vulnerabilities that allowed partial internal access to the [REDACTED] application server. It is highly recommended that [REDACTED] address these vulnerabilities as soon as possible as the vulnerabilities are easily found through basic reconnaissance and exploitable without much effort.

Attack Summary

The following table describes how Joshua Mol gained internal access, step by step:

Step	Action	Recommendation
1	Scanned for web application firewall using WafW00f	Installing a web application firewall increases the difficulty of transmitting certain data to your application. This can increase your attack readiness and deter attack before they occur.
2	Enumerated servers open ports, services, versions, and operating system through an opensource tool Nmap.	Firewalls/web application firewalls can greatly reduce the effectiveness of automated scanning tools, limiting the speed and knowledge they will produce to a potential attacker.
3	Obtained username credentials through unauthenticated views of posts authors.	Redact the usernames of authors who make public posts from view of unauthenticated users. Refrain from using usernames as login credentials instead use emails.
4	Enumerated web application through an opensource tool “dirb” using a common list of URL paths.	Using request limiting services though Cloudflare, CloudFront, and web application firewalls directory enumeration can be drastically slowed. Additionally, for increased security posture remove robots.txt, although this is not a security vulnerability these documents are unnecessary and allows attackers to quickly enumerate web applications, and possible hidden URL paths without scanning.
5	Attempted SQL injection on Login Form	Sanitization/Filtering should be implemented to detect and strip/replace special characters with safe encodings. SQL injection handling needs to be improved, although the SQL injection attempts failed responses contained unnecessary error and operating system information.

6	Performed brute force attack using Burp Suite Pro Intruder against usernames found of main page, with common weak passwords	Utilizing secure form Captcha's, and one-time form tokens will prevent automated attacks, along with account timeout's and lockout periods will increase the web applications resistance to brute force applications. Additionally, enhanced, and enforced password policies for user accounts, greatly reduces the success and timeliness of these attacks.
7	Successfully authenticated as "alice" using password: "55555". Successfully changed Alice's password to "admin"	Enable email verification of password changes to prevent unauthorized persistence attacks. Enforce the recommended password policy to enhance security of user's accounts. Verify correctness of email for password change requests.
8	Obtained usernames, local shares, and table names with discovery of Beta page /private.	Remove beta/development content disclosing operation of backend services, and credentials from production servers. Dev servers should be used for pages and applications which still need to be tested.
9	Uploaded disguised php web shell to /private Beta page created by Pat. Enabling remote code execution.	Remove beta/development pages from production site. Improve regex to check ending extension of files. Additionally, extracting data from between "magic bytes" of the file then placing it within image file that the server generates is the most secure method of allowing picture uploads.
10	Utilized web shell and local http server, uploading python_reverse_tcp to /dev/shm (RAM Disk) Getting local pseudo-shell, enabling ease of directory enumeration.	Configuring security groups and firewall policies restricting server access to initiate unsolicited connections. This would prevent reverse shells from being effective.
11	Enumerated directories, determining that pat has SSH keys for the server.	Although permissions on the id_rsa (private key) are sufficient. Pat's home directory shouldn't be able to be read by the web server. Increasing the security to disallow read access on contents with the home directory is best practice.
12	Obtained MySQL credentials hard coded within /var/www/html/private/index.php page.	Configuring config files and environment variables is a much safer way to handle sensitive information such as database credentials. Credentials should never be hard coded with scripts or pages.
13	Obtained pat's private SSH key from the MySQL database table. Obtained additional user credentials and locations to encrypted files in root directory from same logins and top-secret table	Removing hard-coded credentials from files, while using frameworks and config files should be implemented

14	Accessed web server using Pat's SSH Keys	Having password authentication along with requiring the SSH Private key would enhance security for this type of attack.
15	Generated one-time password reset link for admin using Drush	Improving security posture to prevent unauthorized access to server configured with Drush.
16	Generated SSH keys for Joshua (Attacker)	Removing access for the SSH-keygen and other SSH configuration tools that where given to both the web server and pat.

Security Strengths

Privileges, Execution Rights, Patches

During the assessment, [REDACTED] security controls for internal server were able to prevent execution of several exploits that the system appeared to be susceptible against. Resource restrictions such as insufficient privileges, locked “dpkg” and prevention of execution of GCC many exploits didn’t run as intended. Additionally, kernel exploits were prevented in this manner preventing privilege escalation on the server to root. Cron job-based escalation attempts were also denied due to insufficient privileges in editing global crontab located in /etc/crontab.

Security Weaknesses

Missing Multi-Factor Authentication

I leveraged multiple attacks against [REDACTED] TestMasheen login forms using valid credentials harvested through sourced intelligence. Successful logins included employee accounts through TestMasheen login portal and internal access via SSH Public key encryption. The use of multi-factor authentication would have prevented full access and required me to utilize additional attack methods to gain internal network access.

Weak Password Policy

I have successfully performed password brute force attacks against [REDACTED] login forms, providing application access. Predictable and weak passwords such as “booboo”, “55555”, “charmed” were attempted and successful. Recommended password policy should be enforced on all users instead of optional.

Unrestricted Logon Attempts

During the assessment, I performed multiple brute-force attacks against login forms found on the external network. For all logins, unlimited attempts were allowed, which permitted an eventual successful login on the TestMasheen Drupal application. Blacklisting IP’s, locking accounts, and throttling login attempts should be used to combat this issue in addition to anti-automation techniques.

External Penetration Test Findings

Insufficient Lockout Policy – TestMasheen App (Critical)

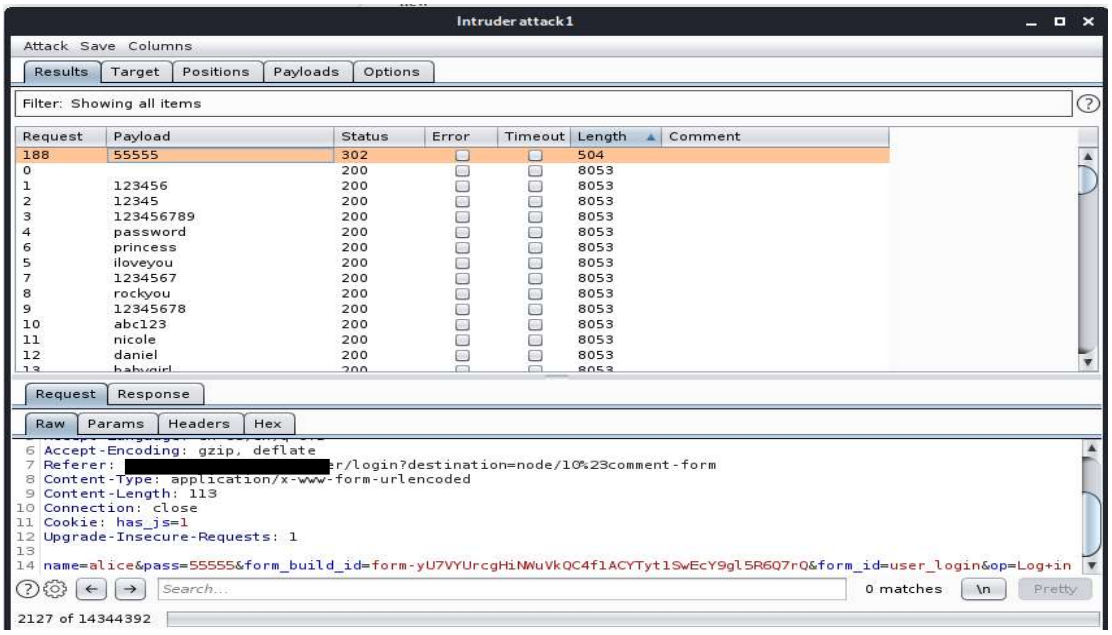
Description:	[REDACTED] allowed unlimited logon attempts against their TestMasheen Drupal Web App. This configuration allowed brute force and password guessing attacks, which I used to gain access to user's accounts.
Impact:	Critical
System:	[REDACTED]

Exploitation Proof of Concept

I had gathered historical commonly used credentials amounting to 14.5 Million total account credentials (**Note:** A full list of compromised accounts can be found in "**[REDACTED].xslx**".).

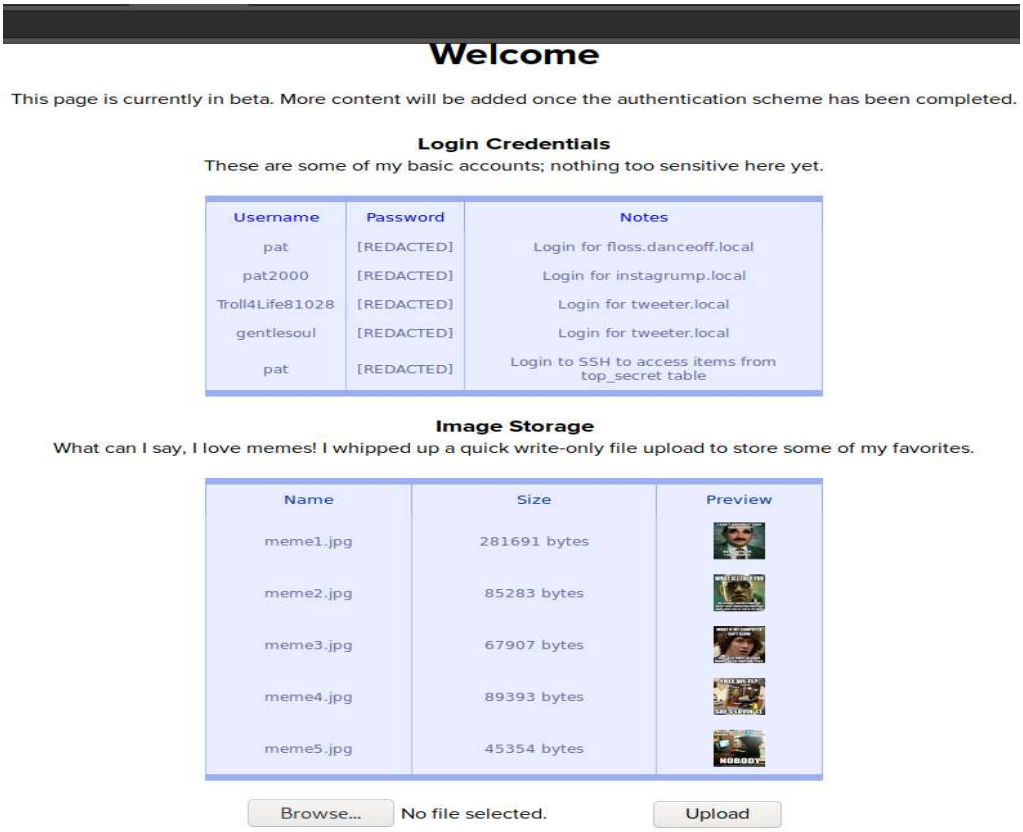
	A	B	C
	User	**Pass**	
	alice	55555	
	david	booboo	
	bob	popcorn	
	carol	charmed	
	admin	admin	*(After Reset)*

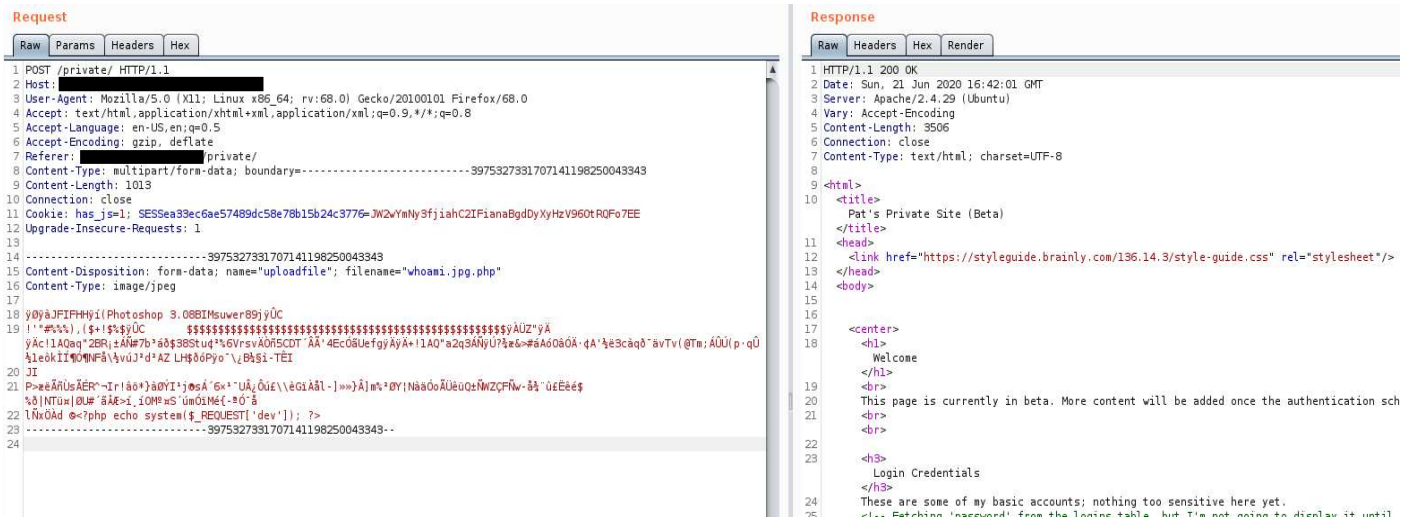
Figure 1: Sample list of breached user credentials



Development pages in Production site – TestMasheen App (Critical)

Description:	[REDACTED] TestMasheen had a index.php page in the /private URL path. This page did insufficient upload checks on images allowing me to get a web shell compromising internal infrastructure.
Impact:	Critical
System:	[REDACTED]





Secrets Database Credential Exposure – TestMasheen Server (Critical)

Description:	[REDACTED] TestMasheen had a index.php page in the /var/www/html/private URL path. Once a reverse shell is created piping the file to less allows a user to find the credentials for secrets database.
Impact:	Critical
System:	[REDACTED]

```php
<?php

// MySQL Settings
$host = "localhost";
$user = "secrets";
$pass = "A718xhrB1xYwS6K5qhe9EMYmaee49a99";
$db   = "secrets";

// Connect and error upon failure
$c = new mysqli($host, $user, $pass, $db);
if ($c->connect_error) die("Connection failed: " . $conn->connect_error);

// Set up the query
$sort = "";
if (isset($_GET['c']) and isset($_GET['d'])){
        $sort = "ORDER BY " . $_GET['c'] . " " . $_GET['d'];
}
$sql = "SELECT username, password, notes FROM logins " . $sort;
$result = $c->query($sql);
?>
```

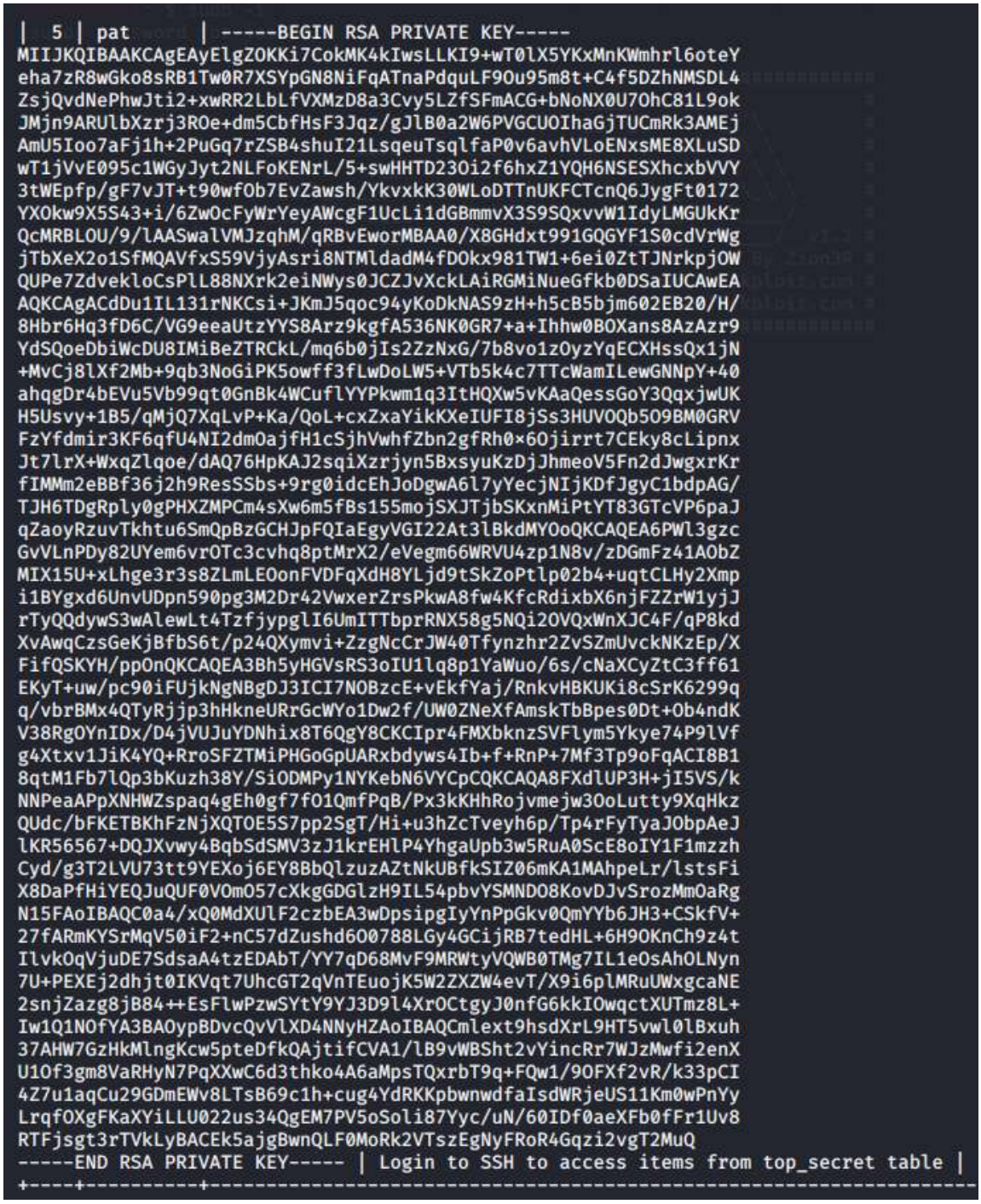
DrupalDB Database Credential Exposure – TestMasheen Server (Critical)

Description:	[REDACTED] TestMasheen contained a second php file for the main app that contained hard coded database credentials for drupaldb.
Impact:	Critical
System:	[REDACTED]

```php
*/
$databases = array (
  'default' =>
  array (
    'default' =>
    array (
      'database' => 'drupaldb',
      'username' => 'drupaluser',
      'password' => 'wutfwpbHqcORQKmw/ycCPBwIc44a5ad9',
      'host' => 'localhost',
      'port' => '',
      'driver' => 'mysql',
      'prefix' => '',
    ),
  ),
);
```

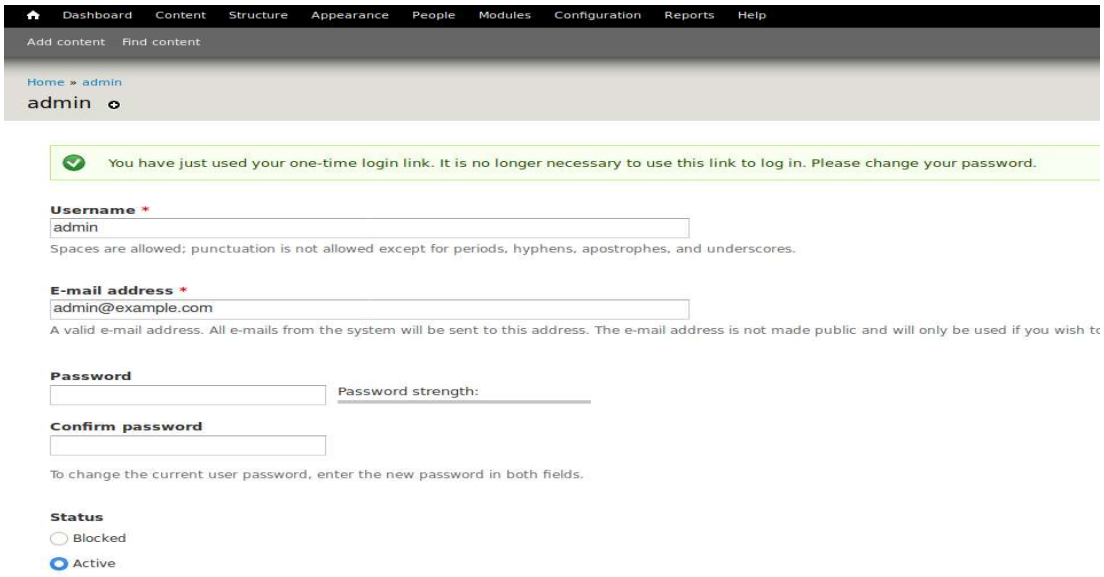
SSH Private Key Exposure – TestMasheen Server (Critical)

Description:	Private SSH keys for pat are in the secrets database under top_secret table. The keys are exposed through a simple SELECT statement using credentials shown above.
Impact:	Critical
System:	[REDACTED]



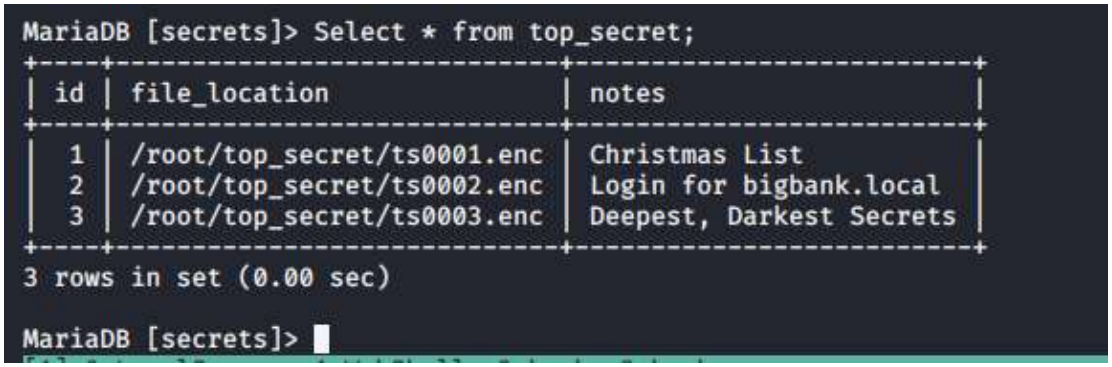
Drush Admin Password Reset – TestMasheen Server (Critical)

Description:	Using the reverse shell from the faulty development page, we now have access to Drush which get us the ability to generate one-time links to reset passwords for any user unauthenticated.
Impact:	Critical
System:	[REDACTED]



Confidential Information Location Exposure – TestMasheen Server (High)

Description:	Accessing the Secrets database using credentials hardcoded in a server php file an attacker can target files encrypted files with banking information and other data that may be valuable.
Impact:	High
System:	[REDACTED]



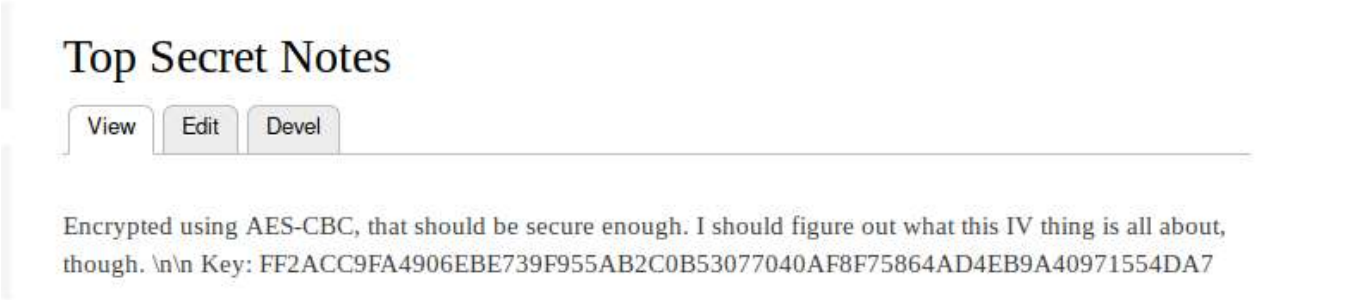
User Hash Exposure– TestMasheen Server (High)

Description:	Within the Drupal database using credentials hardcoded into a script an attacker can dump the password hashes off all the users including the admin.
Impact:	High
System:	[REDACTED]

```
MariaDB [drupaldb]> SELECT * FROM users;
+-----+-------+-------------------------------------------------------------+---------------------+------+-----------+------------------+------------+------------+------------+--------+---------------+----------+---------+------+
| uid | name  | pass                                                        | mail                | theme | signature | signature_format | created    | access     | login      | status | timezone      | language | picture | init |
+-----+-------+-------------------------------------------------------------+---------------------+------+-----------+------------------+------------+------------+------------+--------+---------------+----------+---------+------+
|   0 |       | NULL                                                        |                     |      |           | NULL             |          0 |          0 |          0 |      0 | NULL          |          |       0 |
|   1 | admin | $S$D4wU1QD4jOMd5Zgx6zhf0ToQURpOXEHiJXxxBsqv3kQZkefYvIdf | admin@example.com   |      |           | NULL             | 1537553460 |          0 | 1537553460 |      1 | UTC           |          |       0 | admin@example.com
|   2 | alice | $S$D0XyqcQsXeI0RfFNMBE.eJbiYXCacG4Wgk6Ws0B5ZtGyRX8A6zdO | devr4ndom@gmail.com |      |           | filtered_html    | 1592744464 | 1592784184 | 1592784184 |      1 | America/New_York |          |       0 |
|   3 | bob   | $S$D7ONqcFgFVImAxh2CYFG5RdMRfMFfTK8mXJzmEhwssSE4NzGU73j | hello@hotmail.com   |      |           | filtered_html    | 1592744464 | 1592784567 | 1592784567 |      1 | America/New_York |          |       0 |
|   4 | carol | $S$D9hnbUc5KTGNBSEkpPhfV3MBYcD/pisHe7v8Nwn/YgjB1J776aS6 | hel@hel.ciom        |      |           | filtered_html    | 1592744465 | 1592844721 | 1592843510 |      1 | America/New_York |          |       0 |
|   5 | david | $S$Dm2fVPMwWEwq98D9ofH3.KiWLFDpmvuQa8cXagWEzyUPpZP4PZJL | alice@gmail.com     |      |           | filtered_html    | 1592744465 | 1592784538 | 1592784335 |      1 | America/New_York |          |       0 |
+-----+-------+-------------------------------------------------------------+---------------------+------+-----------+------------------+------------+------------+------------+--------+---------------+----------+---------+------+
6 rows in set (0.00 sec)
```

Admin AES-CBC 256 Key Exposure– TestMasheen App (High)

Description:	In the admin section of the TestMasheen accounts page an unpublished post titled: “Top Secret Notes” contains an AES-CBC key with possibly no initialization vector.
Impact:	High
System:	[REDACTED]



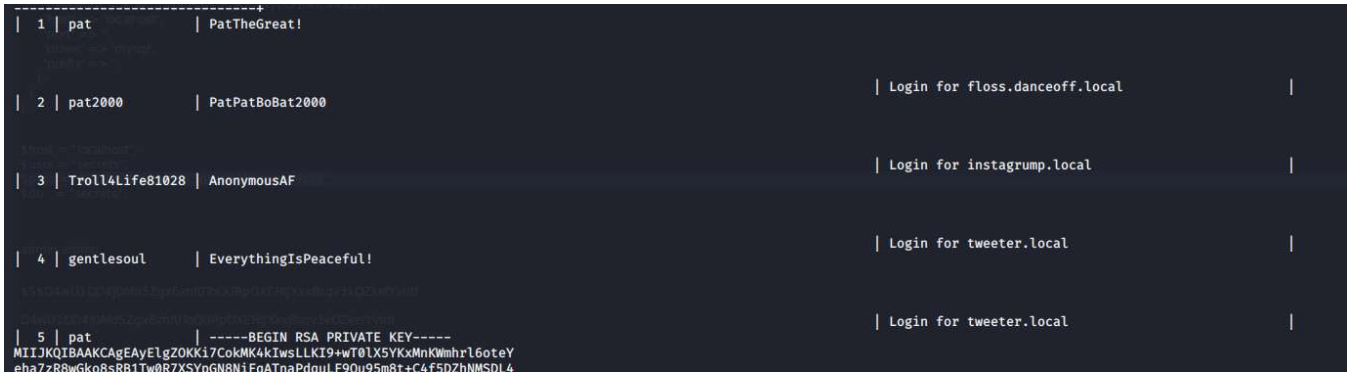
Login Form Token Exposure– TestMasheen App (High)

Description:	Tokens for the login form are concealed only by a “hidden” attribute. Tokens are also reusable giving access to brute force attacks.
Impact:	High
System:	[REDACTED]

```
<input type="hidden" name="form_build_id" value="form-xW195ADK4zfWxOS6JJj1Jb6bCxb-XJqrlzBrKs8gQZ4"> event
<input type="hidden" name="form_token" value="pVjiVzQxCmGuKKW94TeA5Yj2YInA4HBrLYyLDgoRrBY"> event
<input type="hidden" name="form_id" value="user_profile_form"> event
▼<fieldset id="edit-picture" class="form-wrapper">
```

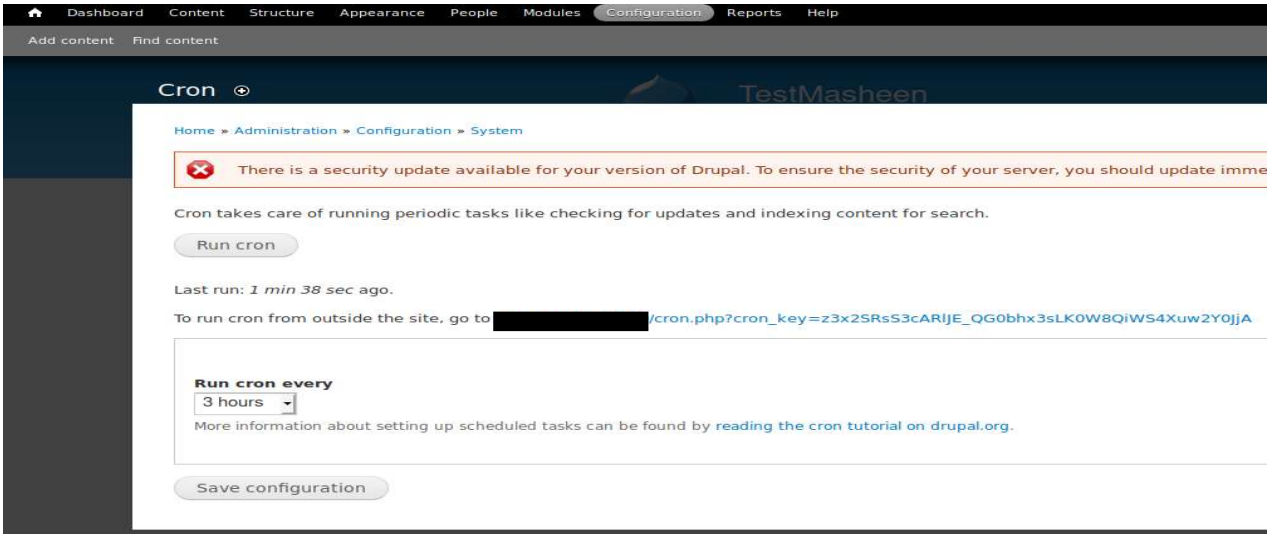
Plain Text Credential Exposure – TestMasheen App (High)

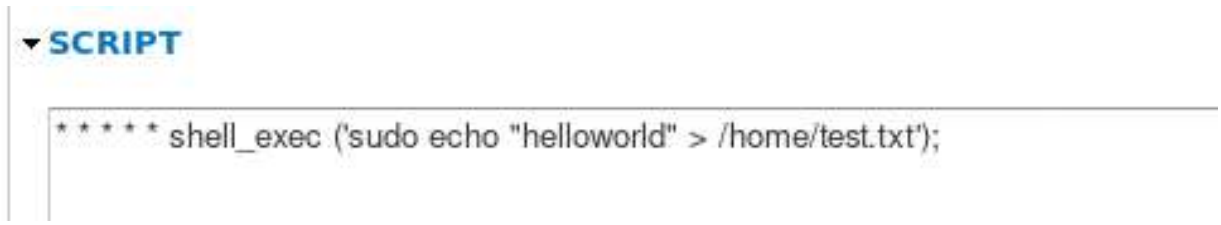
Description:	Accessing the Secrets database using credentials hardcoded in a server php file an attacker can harvest login creds and the associated place to use them.
Impact:	High
System:	[REDACTED]



Cron Key Access – TestMasheen App (Moderate)

Description:	Within the admin console of the web application a cron scheduler is present along with a cron key. This could allow an attacker to run scheduled commands authenticated as admin.
Impact:	Moderate
System:	[REDACTED]



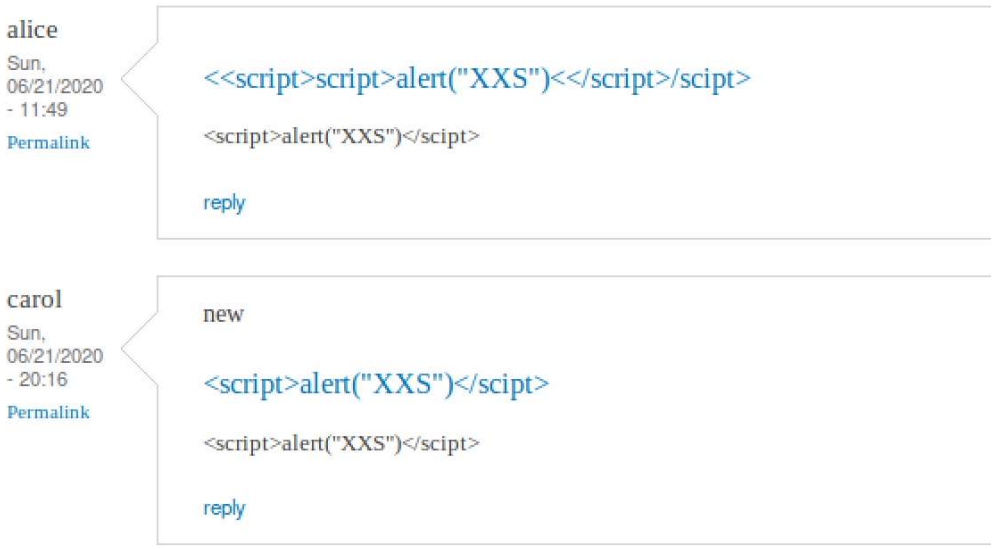


Pat Home Directory Read Exposure – TestMasheen Server (Moderate)

Description:	Server side the www-data service has read permissions to the user pat's home directory. This gives attackers greater information about who to target in their attack.
Impact:	Moderate
System:	[REDACTED]

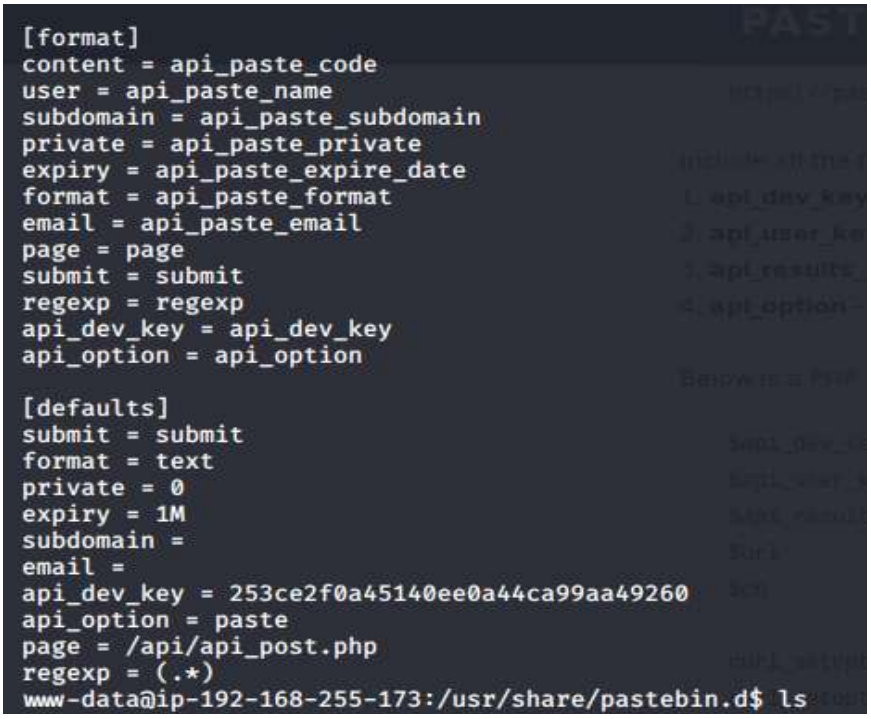
Insufficient Script Tag Filtering – TestMasheen App (Moderate)

Description:	While making a post to a drupal node the filtered HTML option doesn't sufficiently filter <script> tags if organized correctly.
Impact:	Moderate
System:	[REDACTED]



Pastebin Dev API Key Exposure – TestMasheen Server (Moderate)

Description:	Within the /usr/share/pastebin.d directory a URL and API key are hard coded into the pastebin.com.conf file. This could possibly lead to impersonation attacks although don't affect the server's integrity.
Impact:	Moderate
System:	[REDACTED]



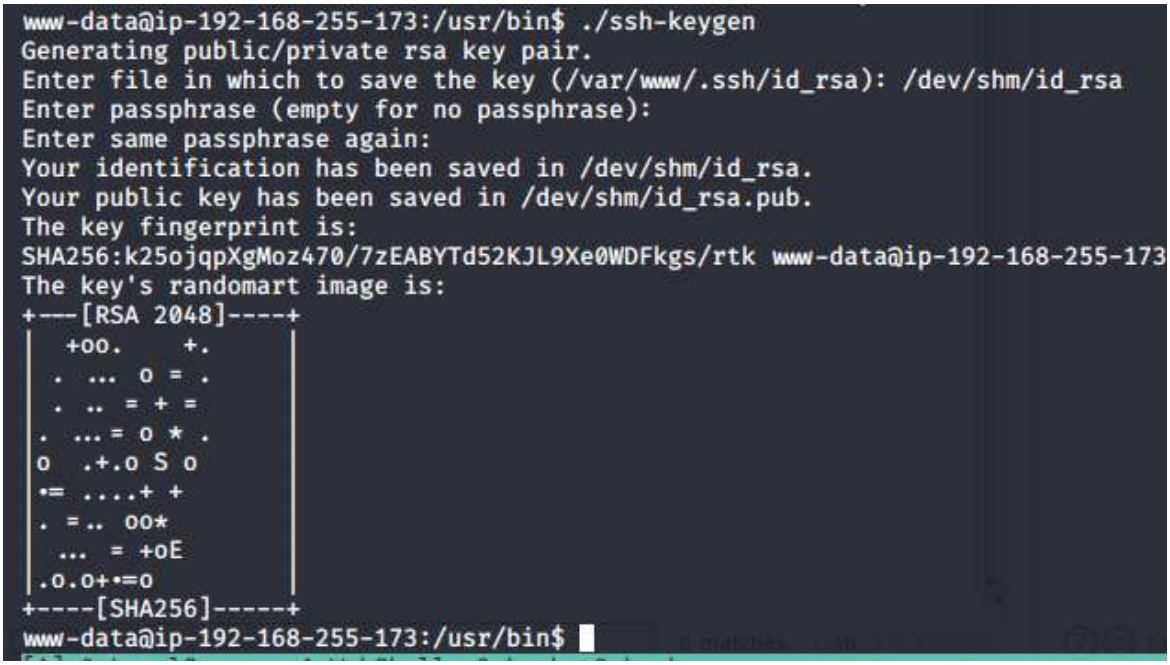
User Password Reset – TestMasheen App (Moderate)

Description:	Persistence methods can be executed with knowledge to user affected, 2FA should be enabled to ensure attacker persistence isn't successful.
Impact:	Moderate
System:	[REDACTED]

```
POST /user/2/edit HTTP/1.1
Host: [redacted]
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: [redacted]/user/2/edit
Content-Type: multipart/form-data; boundary=---------------------------706362498392484758207918873 2
Content-Length: 1415
Connection: close
Cookie: has_js=1; SESSea33ec6ae57489dc58e78b15b24c3776=nu_emNzQxZIgDx6SOQc2ZdAY2TaAT_7k19gwThyfSF4
Upgrade-Insecure-Requests: 1

-----------------------------706362498392484758207918873 2
Content-Disposition: form-data; name="current_pass"

55555
-----------------------------706362498392484758207918873 2
Content-Disposition: form-data; name="mail"

devr4ndom@gmail.com
-----------------------------706362498392484758207918873 2
Content-Disposition: form-data; name="pass[pass1]"

admin
-----------------------------706362498392484758207918873 2
Content-Disposition: form-data; name="pass[pass2]"

admin
-----------------------------706362498392484758207918873 2
Content-Disposition: form-data; name="form_build_id"

form-flHdX5meyOo8wHm1aek2eTdjonDu5ZvDu-XO7oOQX2U
-----------------------------706362498392484758207918873 2
```

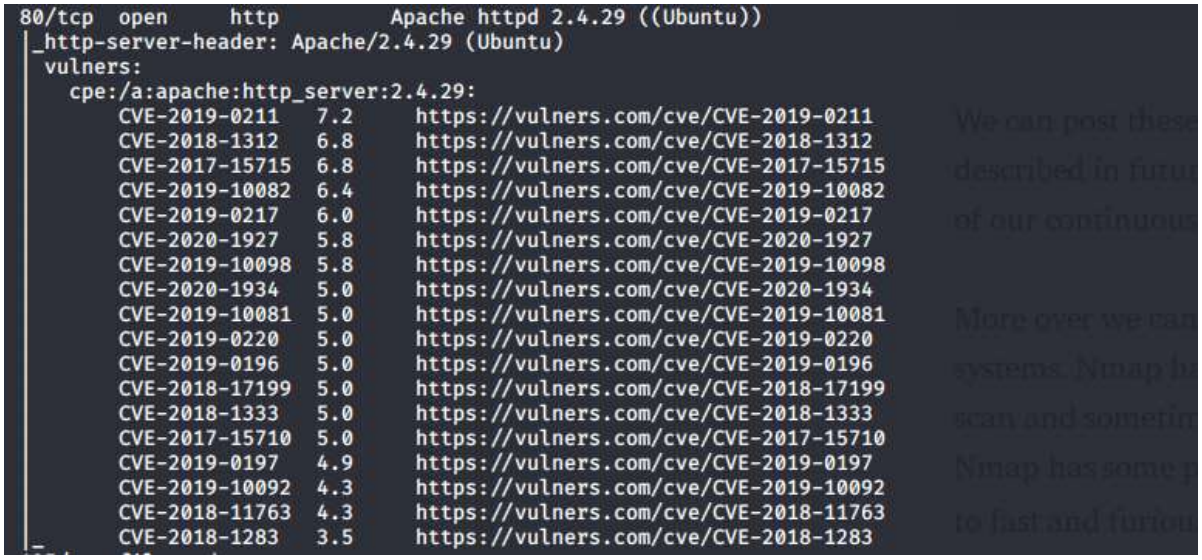
SSH-keygen – TestMasheen Server (Moderate)

Description:	Attackers can generate persistence SSH keys, then implant them in the know-hosts file when elevated to root.
Impact:	Moderate
System:	[REDACTED]



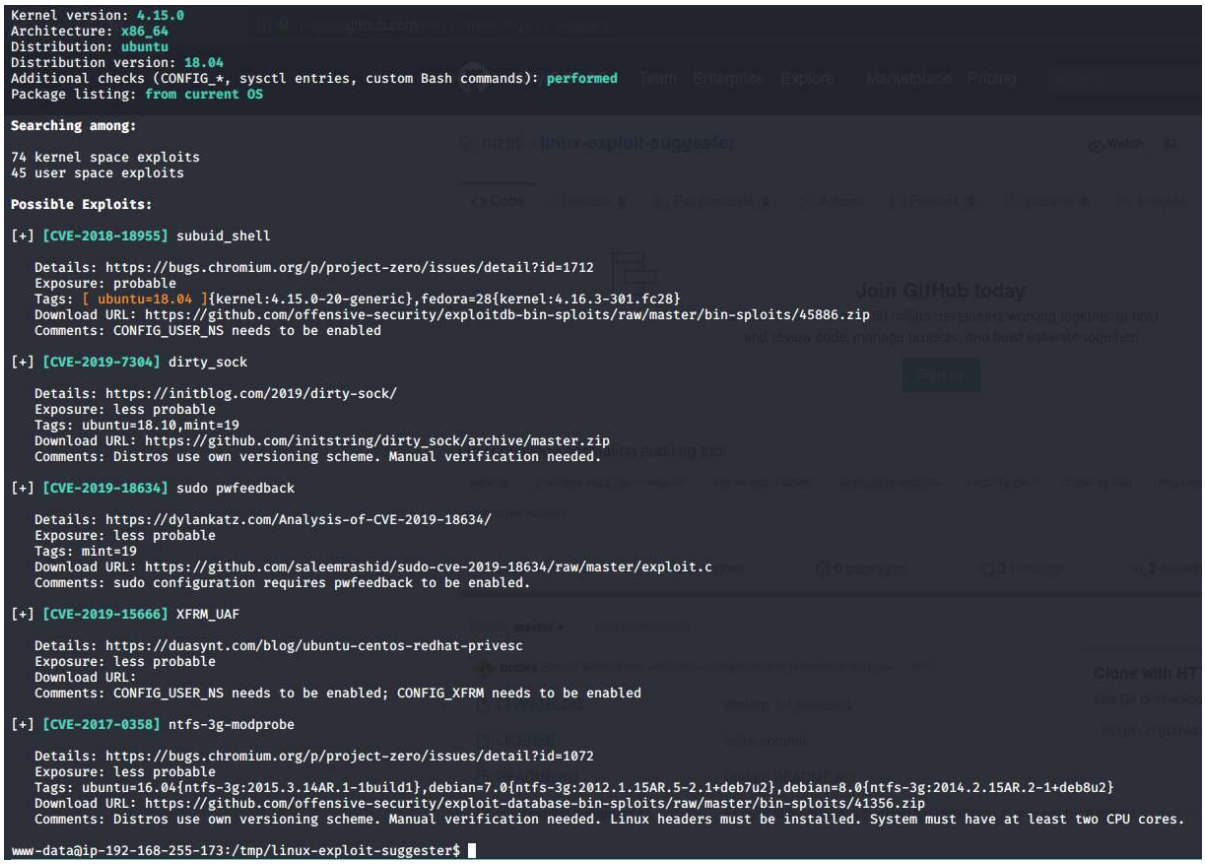
Nmap Vuln Scan – TestMasheen Server (Moderate)

Description:	Outdated Apache server, multiple CVE's can be used to exploit.
Impact:	Moderate
System:	[REDACTED]



Exploit Suggestions Scan – TestMasheen Server (Low)

Description:	www-data service has access to git, and wget allowing download of scripts to enumerate for vulnerabilities.
Impact:	Low
System:	[REDACTED]

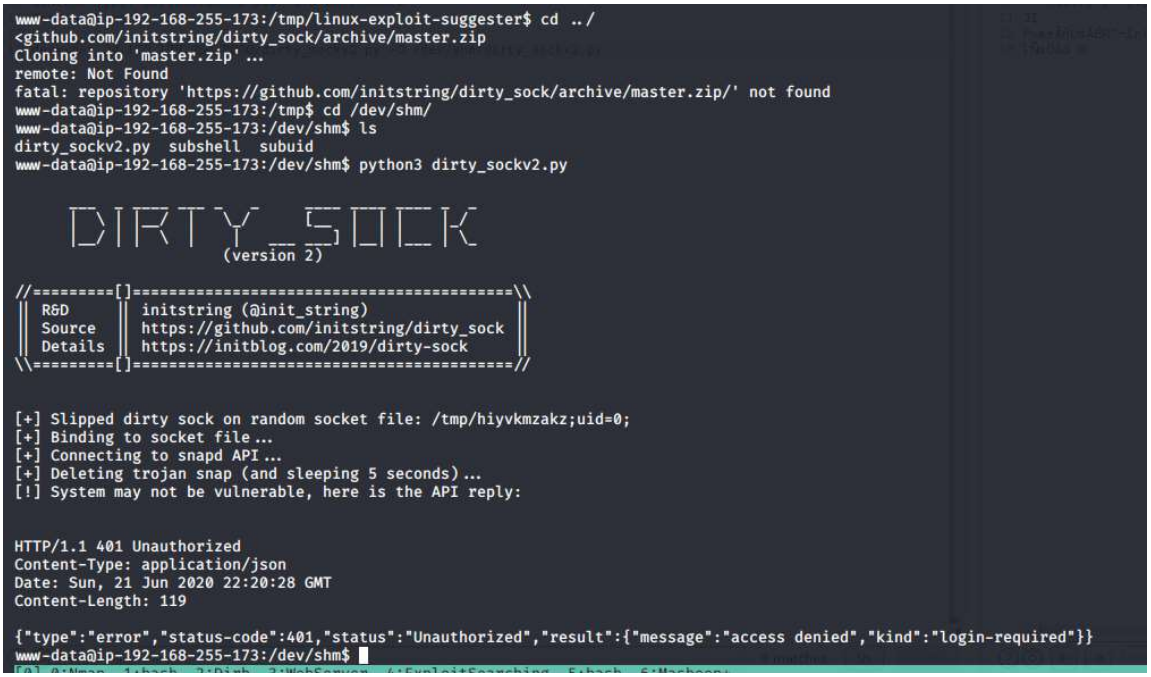


HTML Developer Comment – TestMasheen App (Low)

Description:	A developer made a comment on a php page about retrieving information but not displaying it, this is an indication to an attacker that this php file is valuable.
Impact:	Low
System:	[REDACTED]

DirtySockV2 Exploit – TestMasheen Server (Low)

Description:	Dirty sock sideloads snap that contains an install-hook that generates a new local user. This instance isn't vulnerable although doesn't have potentially vulnerable components.
Impact:	Low
System:	[REDACTED]



Change Log Exposure – TestMasheen App (Informational)

Description:	The Drupal Changelog.txt files read permissions were not changed allowing attacker to determine the exact Version and patches applied to the instance.
Impact:	Informational
System:	[REDACTED]

```
7. Revoke documentation file permissions (optional).

   Some administrators suggest making the documentation files, especially
   CHANGELOG.txt, non-readable so that the exact version of Drupal you are
   running is slightly more difficult to determine. If you wish to implement
   this optional security measure, from a Unix/Linux command line you can use
   the following command:

     chmod a-r CHANGELOG.txt
```

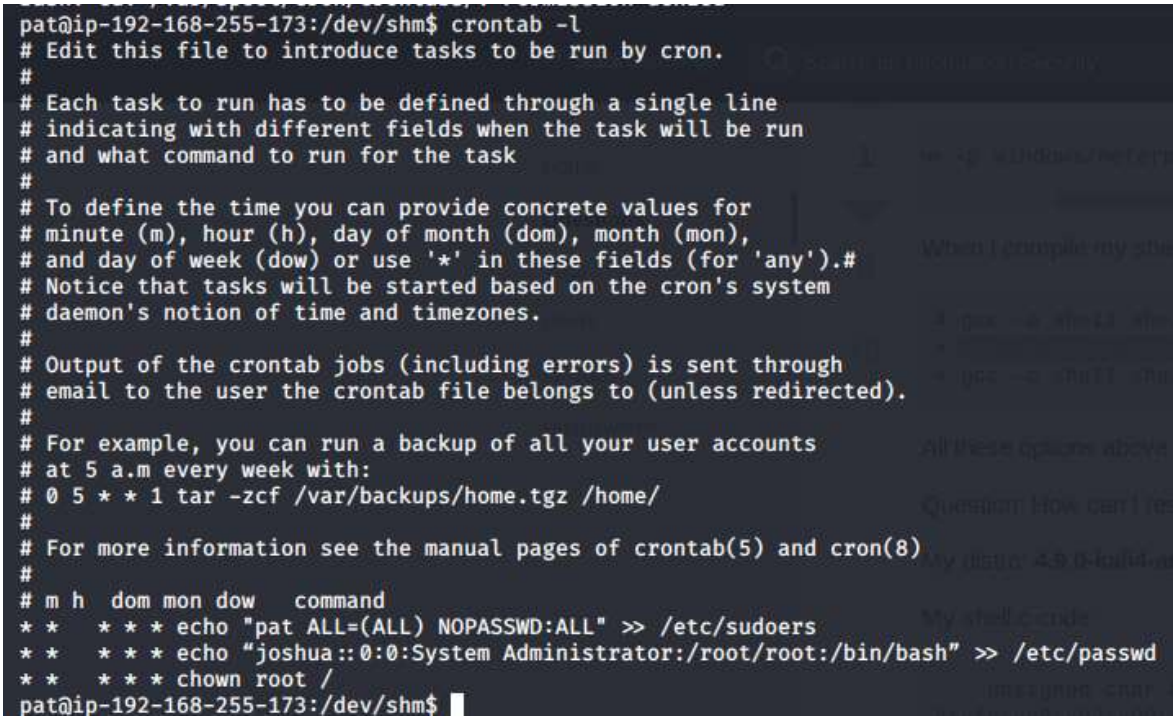
SudoInject V1, V2, V3 – TestMasheen Server (Informational)

Description:	SudoInject is a vulnerability in Ubuntu operating systems that makes use of an inactive sudo token to then become root. This instance isn't vulnerable.
Impact:	Informational
System:	[REDACTED]

```
pat@ip-192-168-255-173:/dev/shm$ sudo gcc
[sudo] password for pat:
pat@ip-192-168-255-173:/dev/shm$ sh sudoinject.sh
Injecting process 6476 → sh
sh: echo: I/O error
Injecting process 6493 → bash
sh: echo: I/O error
cat: /proc/6895/comm: No such file or directory
Injecting process 6895 →
sh: echo: I/O error
[sudo] password for pat:
Sorry, try again.
[sudo] password for pat:
Sorry, try again.
[sudo] password for pat:
sudo: 3 incorrect password attempts
pat@ip-192-168-255-173:/dev/shm$ sudo -i
[sudo] password for pat:
Sorry, try again.
[sudo] password for pat:
Sorry, try again.
[sudo] password for pat:
sudo: 3 incorrect password attempts
pat@ip-192-168-255-173:/dev/shm$
```

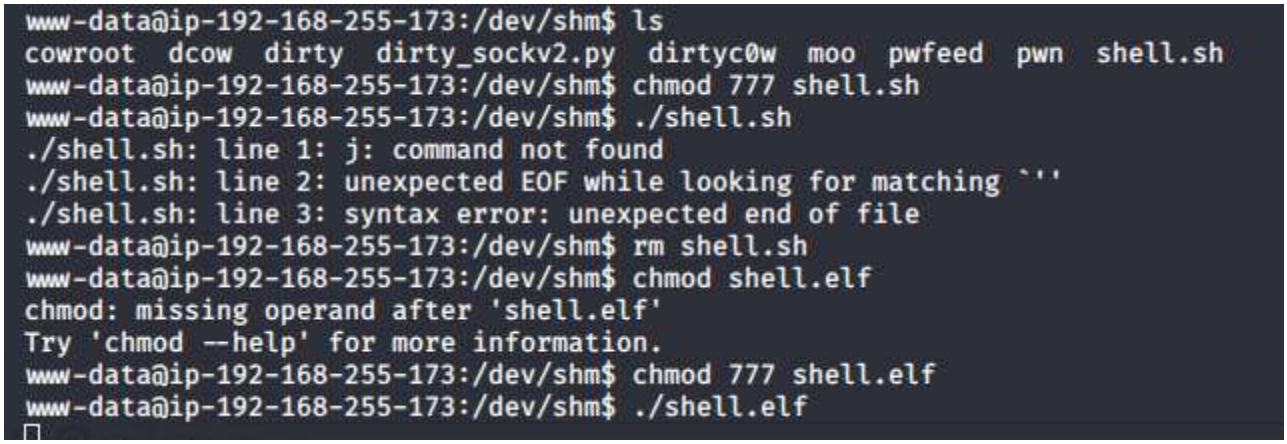
User Crontab privilege escalation – TestMasheen Server (Informational)

Description:	An attempt to use pat crontab to elevate privileges failed. This system doesn't appear to be vulnerable.
Impact:	Informational
System:	[REDACTED]



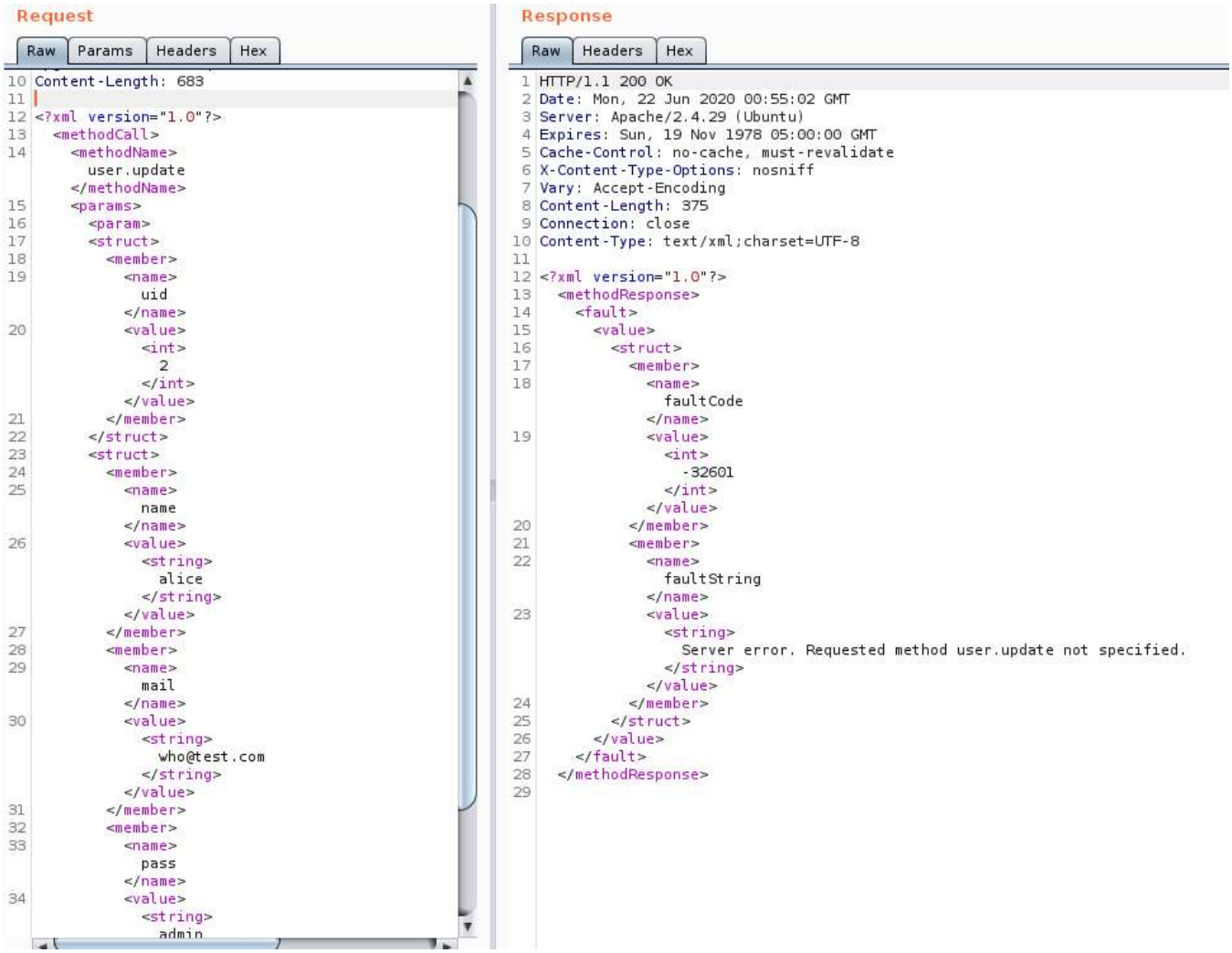
Dirty Cow Exploitation – TestMasheen Server (Informational)

Description:	An attempt was made to use DirtyCow to elevate privileges to root but failed. This system doesn't appear to be vulnerable.
Impact:	Informational
System:	[REDACTED]



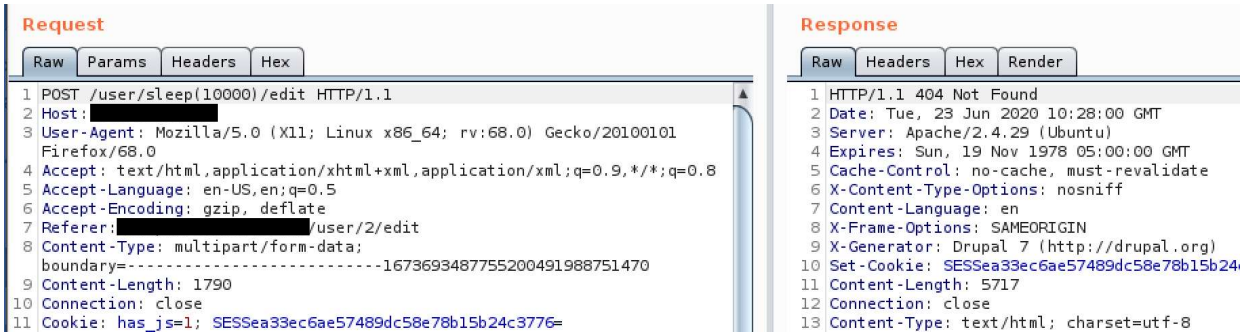
XMLRPC – TestMasheen Server (Informational)

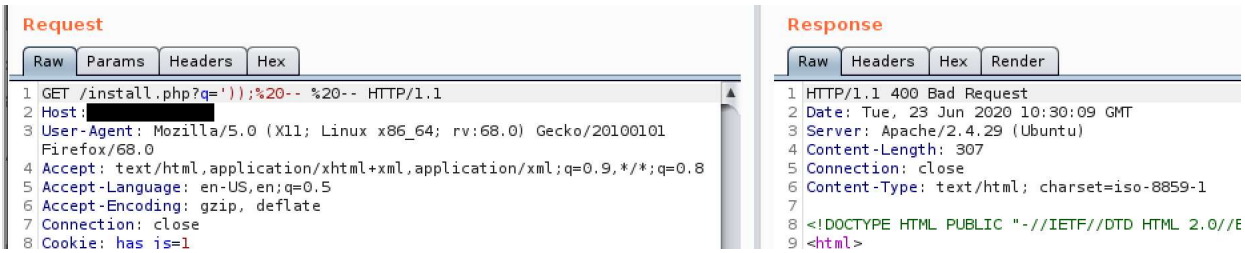
Description:	An attempt to use XMLRPC command API. API didn't respond to commands issued, possibly due to outdated documentation.
Impact:	Informational
System:	[REDACTED]

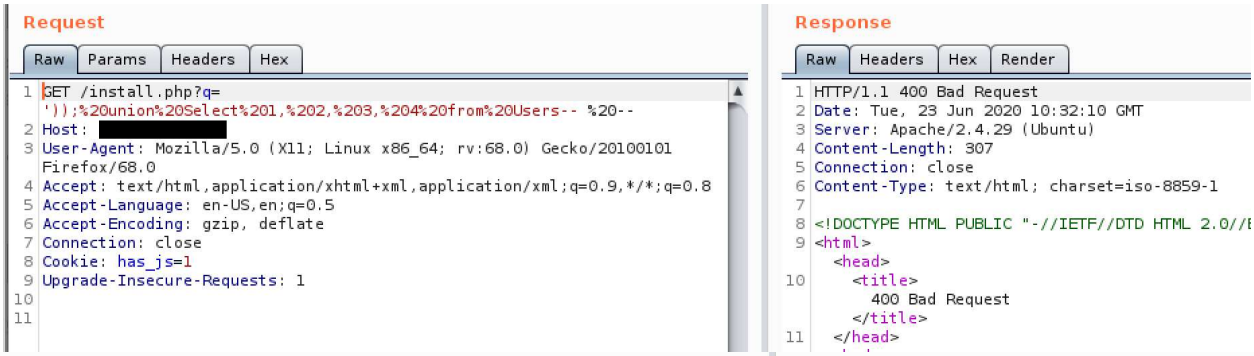


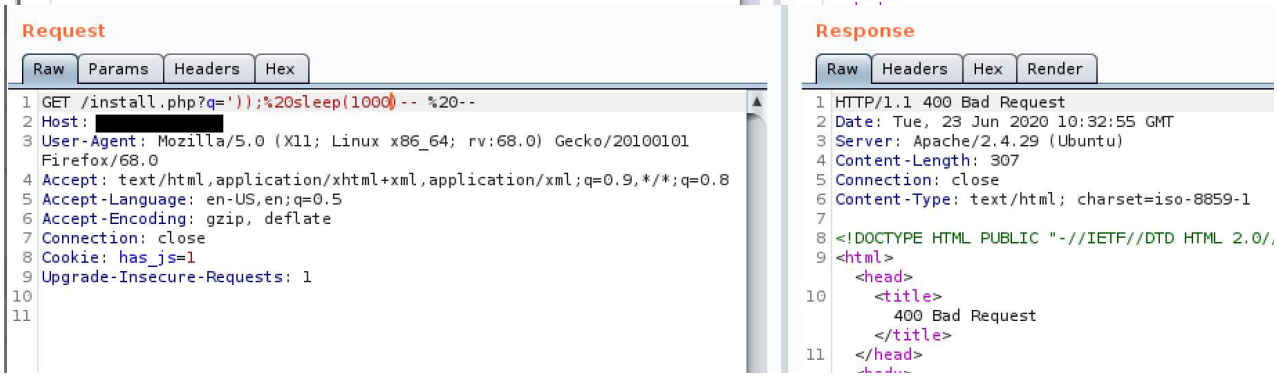
SQL Injection Attempts (Informational)

Description:	Attempts were made to utilize SQL injection although site didn't seem vulnerable even to blind SQL injection.
Impact:	Informational
System:	[REDACTED]









Last Page