Gone Phishing Security

Stocker Testing Report

Pen-testing Security Service, LLC

502 E BOONE AVE,

SPOKANE, WA 99258

Tel: (202) 324-3000

penetration test report for Stocker

A. Penetration Test

a. Summary and Introduction

i. This report overviews the efforts attempted to compromise the Stocker website and web server.

ii. Testing was performed by Connor Goldschmidt

b. Requirements:

i. In this report, we will walk through the steps that led up to compromising the stocker website. We have included screenshots and sources of how we performed the attack.

ii. We also have included a high level summary as well as recommendations on how to improve.

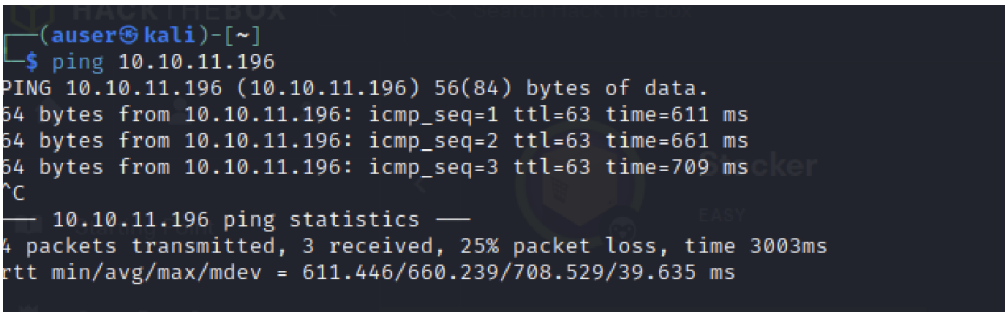
c. Objectives

i. Our main objective was to perform a penetration test for the stocker website. Specifically, we were aiming to gain root access of the webserver to gain full control of the website.

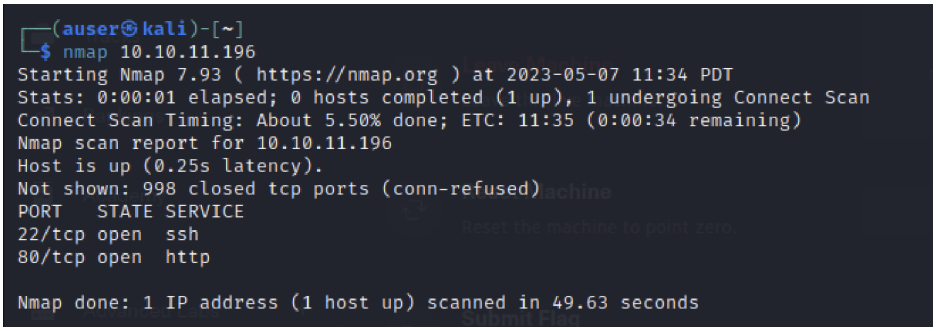
A. Initial Scan

a. Using Kali Linux, we did an initial scan of the given server and found ports 22 and 80 open suggesting that the system could potentially be hosting a website.

b.



c.



B. Basic Investigation

a. Initially we conducted an inconclusive hydra bruteforce attack on port 22 to attempt to gain access to the system via ssh.

b.

```
┌──(auser㉿kali)-[~]
└─$ hydra -l root -P /usr/share/wordlists/metasploit/unix_passwords.txt 10.10.11.196 ssh -t 4
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non
-binding, these **s ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-05-07 11:38:17
[DATA] max 4 tasks per 1 server, overall 4 tasks, 1009 login tries (l:1/p:1009), ~253 tries per task
[DATA] attacking ssh://10.10.11.196:22/
[STATUS] 24.00 tries/min, 24 tries in 00:01h, 985 to do in 00:42h, 4 active
[STATUS] 27.00 tries/min, 81 tries in 00:03h, 928 to do in 00:35h, 4 active
[STATUS] 23.43 tries/min, 164 tries in 00:07h, 845 to do in 00:37h, 4 active
[STATUS] 23.67 tries/min, 284 tries in 00:12h, 725 to do in 00:31h, 4 active
[STATUS] 23.29 tries/min, 396 tries in 00:17h, 613 to do in 00:27h, 4 active
[STATUS] 23.14 tries/min, 509 tries in 00:22h, 500 to do in 00:22h, 4 active
[STATUS] 23.11 tries/min, 624 tries in 00:27h, 385 to do in 00:17h, 4 active
[STATUS] 23.25 tries/min, 744 tries in 00:32h, 265 to do in 00:12h, 4 active
[STATUS] 23.24 tries/min, 860 tries in 00:37h, 149 to do in 00:07h, 4 active
[STATUS] 23.18 tries/min, 881 tries in 00:38h, 128 to do in 00:06h, 4 active
[STATUS] 23.18 tries/min, 904 tries in 00:39h, 105 to do in 00:05h, 4 active
[STATUS] 23.18 tries/min, 927 tries in 00:40h, 82 to do in 00:04h, 4 active
[STATUS] 23.17 tries/min, 950 tries in 00:41h, 59 to do in 00:03h, 4 active
[STATUS] 23.14 tries/min, 972 tries in 00:42h, 37 to do in 00:02h, 4 active
[STATUS] 23.09 tries/min, 993 tries in 00:43h, 16 to do in 00:01h, 4 active
[STATUS] 22.93 tries/min, 1009 tries in 00:44h, 1 to do in 00:01h, 1 active
1 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-05-07 12:22:26
```

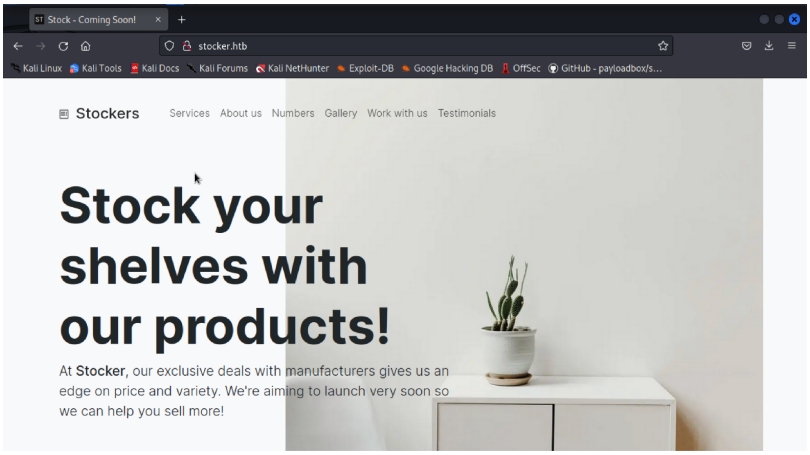
c. Next we drew our attention towards the possible website and found that the server was hosting stocker.htb

d.

```
File  Actions  Edit  View  Help
127.0.0.1       localhost
127.0.1.1       kali

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
10.10.11.210 only4you.htb
10.10.11.196 http://stocker.htb
~
~
~
~
```

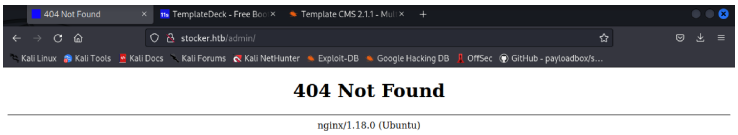
e. Here we added the webpage to our host file and found the stocker website below.

f.



C. Website investigation

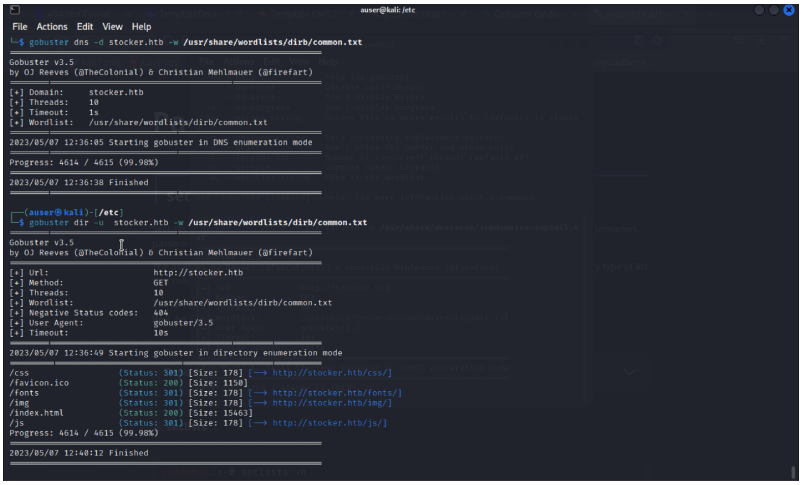
a. Our main approach was to find alternative domains/pages that could be hidden in the website.



b.

c. GoBuster was used in attempts to find further dns information. However, these results were initially inconclusive.

d.



e.

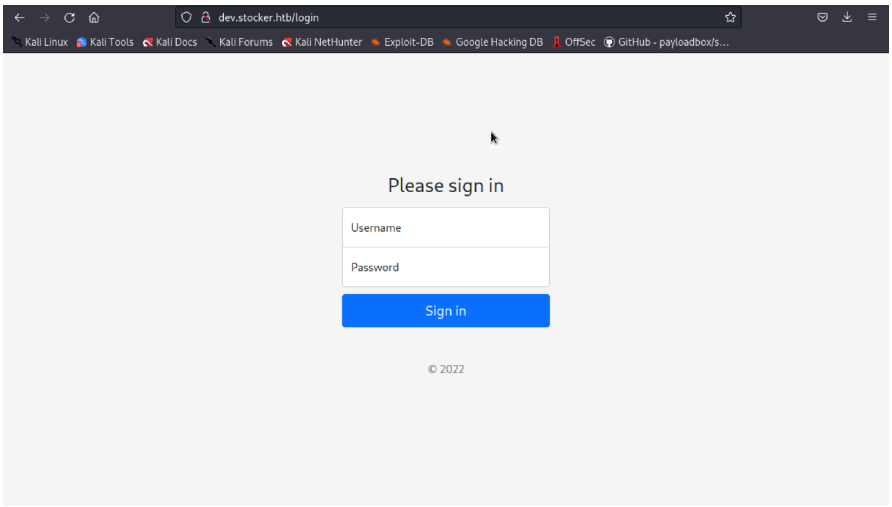
f. Next Virtual hosts were examined and initially did not yield results. However, after discovering the –append-domain flag “dev.stocker.htb” was found.

g.

```
└─$ gobuster vhost -u stocker.htb -w /usr/share/dnsrecon/subdomains-top1mil.t
xt
===============================================================
Gobuster v3.5
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:          http://stocker.htb
[+] Method:       GET
[+] Threads:      10
[+] Wordlist:     /usr/share/dnsrecon/subdomains-top1mil.txt
[+] User Agent:   gobuster/3.5
[+] Timeout:      10s
[+] Append Domain:   false
===============================================================
2023/05/07 16:29:32 Starting gobuster in VHOST enumeration mode
===============================================================
Progress: 114595 / 114607 (99.99%)
===============================================================
2023/05/07 17:18:31 Finished
===============================================================
```

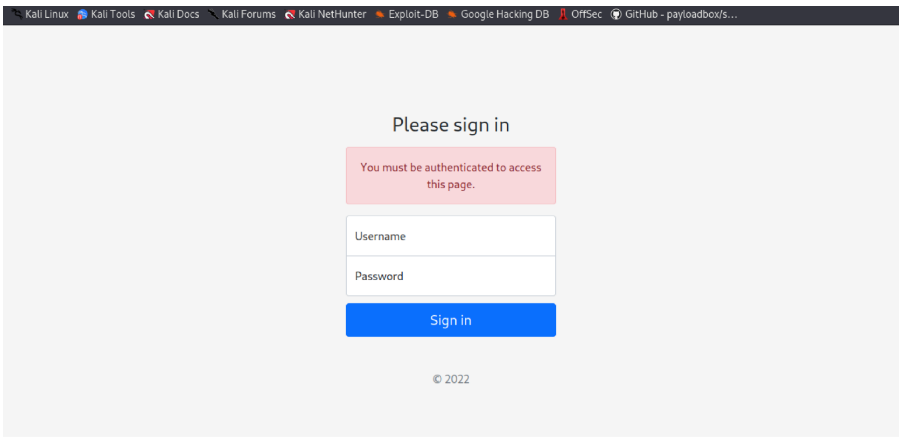
h.

i.



j. Once found we initially tested the login with default credentials “root” and “password” with no success.

k.

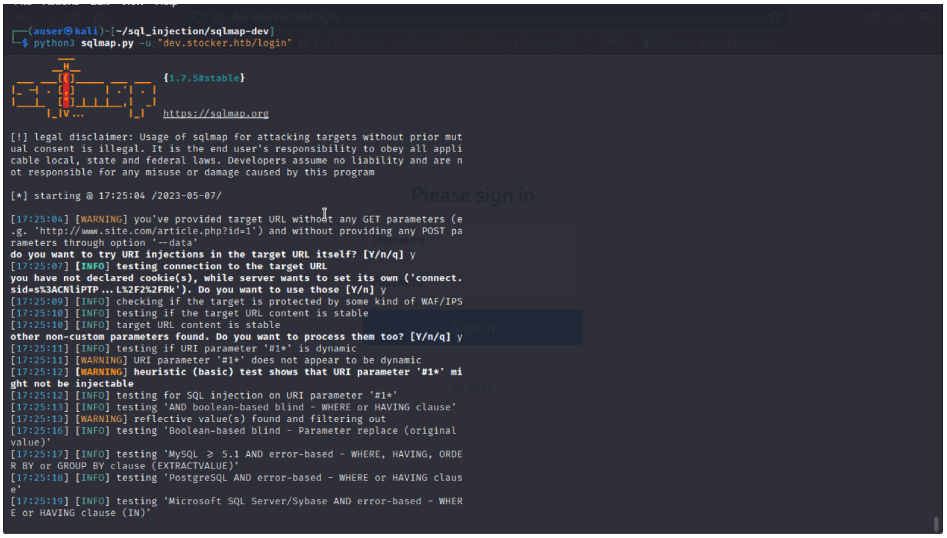


D. SQL Injection

a. After discovering this login page we began scanning for SQL injection vulnerabilities. First we used sqlmap to scan for basic sql vulnerabilities.

b. However, these tests were inconclusive.

c.



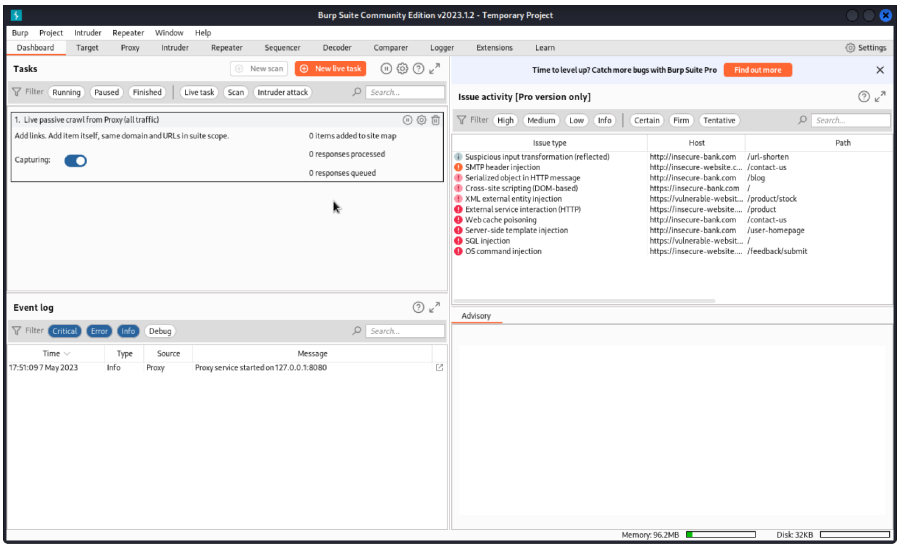
d. We then found a recommended similar attack known as NOSQL Injection. An overview regarding this attack can be found here (OWASP).

e. As a result we shifted our approach to focus on http requests.

E. Compromising the site

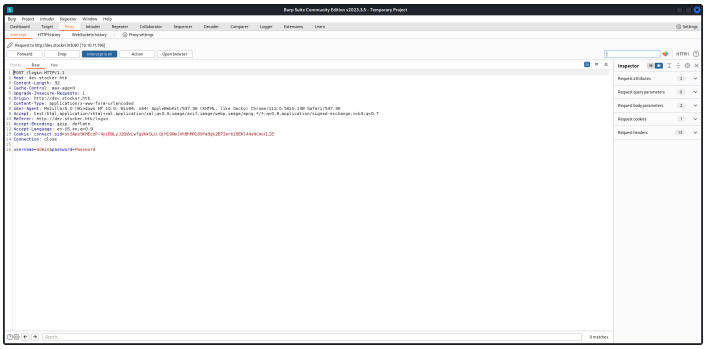
a. After discovering that the website could be potentially vulnerable to a NOSQL injection we used burp suite to analyze our packets.

b.

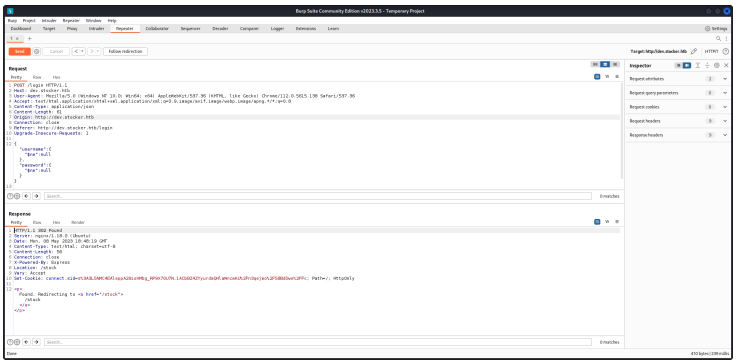


c. Here we use the intercept mode to intercept our http requests to the website.

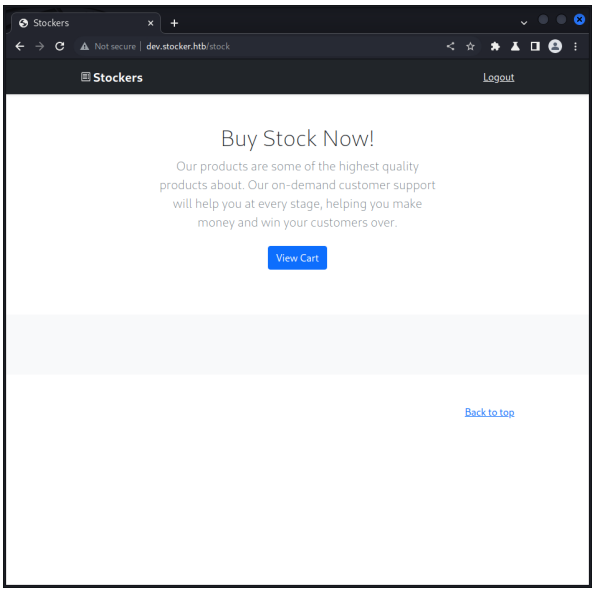
d.



e. Knowing that the site was vulnerable to a NOSQL attack we modified the post request to inject sql code into the website.

f. 

g. As a result we were able to bypass the login screen and were redirected to the /stock page.

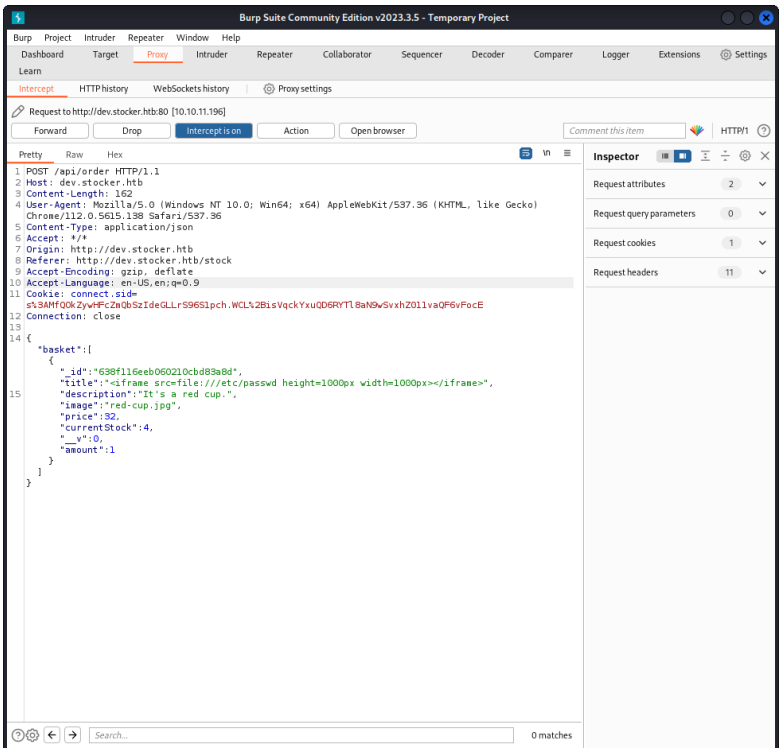
h. 

F. Compromising Stock Webpage

a. After discovering the XSS PDF vulnerability, the iframe html tag was used to exploit the /etc/passwd file to discover users. After this, the user “angoose” was discovered.

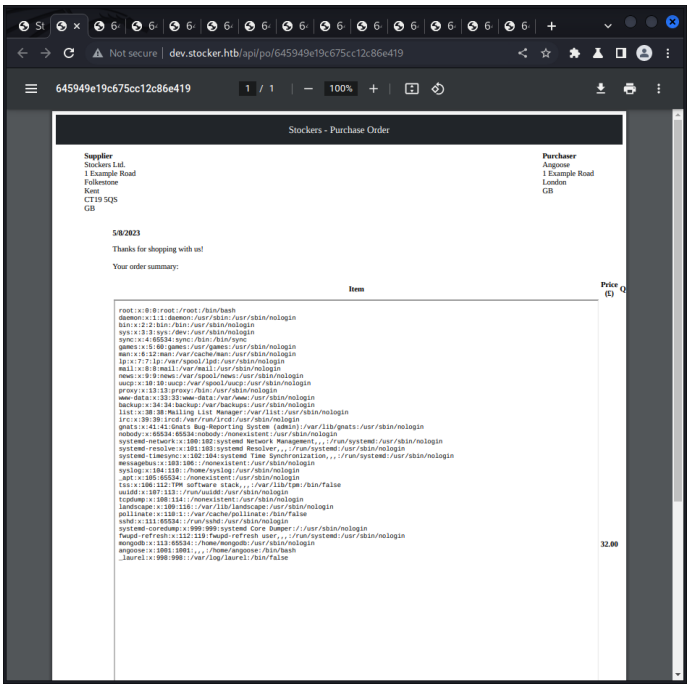
b. This PDF vulnerability was discovered in triskele labs article here

c.



d. After using this XSS (cross-site scripting) attacking technique we were able to locate the users associated with the server. Specifically we attacked the user “angoose”.

e.



f. With this user discovered we ran an inconclusive hydra attack on the open ssh port with the user angoose specified.

g.

```
┌──(cgoldschmidt㉿kali)-[~]
└─$ hydra -l angoose -P /usr/share/wordlists/metasploit/unix_passwords.txt 10.10.11.196 ssh t -4
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or fo
r illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-05-08 12:17:05
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1009 login tries (l:1/p:1009), ~64 tries per task
[DATA] attacking ssh://10.10.11.196:22/t
[STATUS] 128.00 tries/min, 128 tries in 00:01h, 883 to do in 00:07h, 14 active
[STATUS] 98.67 tries/min, 296 tries in 00:03h, 715 to do in 00:08h, 14 active
[STATUS] 92.29 tries/min, 646 tries in 00:07h, 365 to do in 00:04h, 14 active
1 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-05-08 12:28:33
```

h. Nmap was rerun to determine the version and type of web server that was being run for stocker.

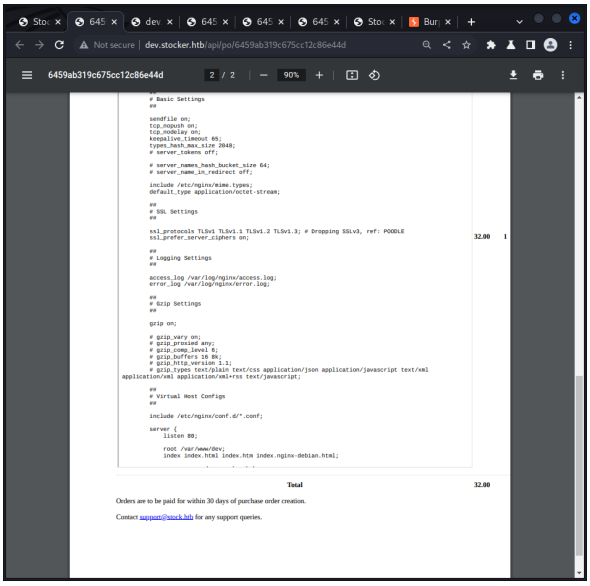
i.

```
File  Actions  Edit  View  Help

┌──(cgoldschmidt㉿kali)-[~]
└─$ sudo nmap -sV 10.10.11.196
[sudo] password for cgoldschmidt:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-08 18:51 PDT
Stats: 0:00:09 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 50.00% done; ETC: 18:51 (0:00:06 remaining)
Nmap scan report for stocker.htb (10.10.11.196)
Host is up (0.21s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
80/tcp open  http    nginx 1.18.0 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.48 seconds
```

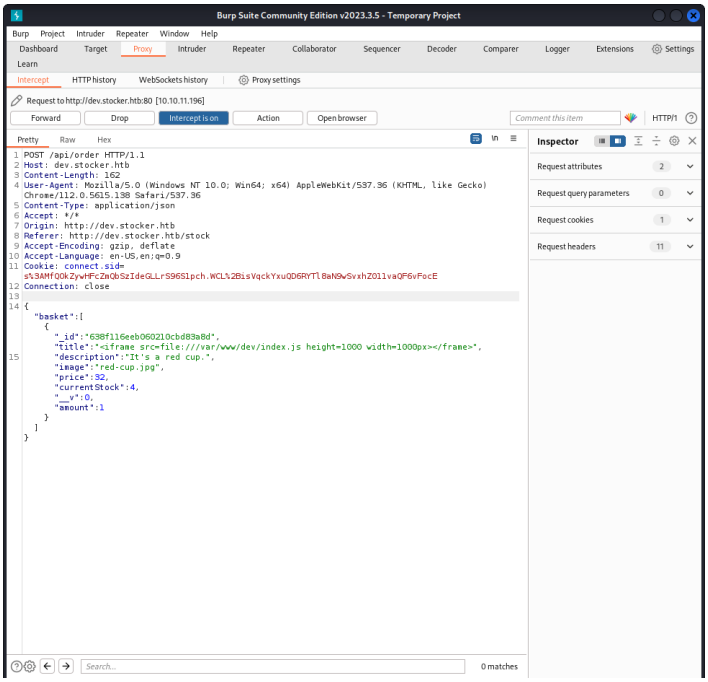
j. After making the determination that nginx version v1.18.0 was being run, we resorted to the config files to determine where the service operates from. (source [plesk])

k. We resent the post request with the desired injection and received this output.

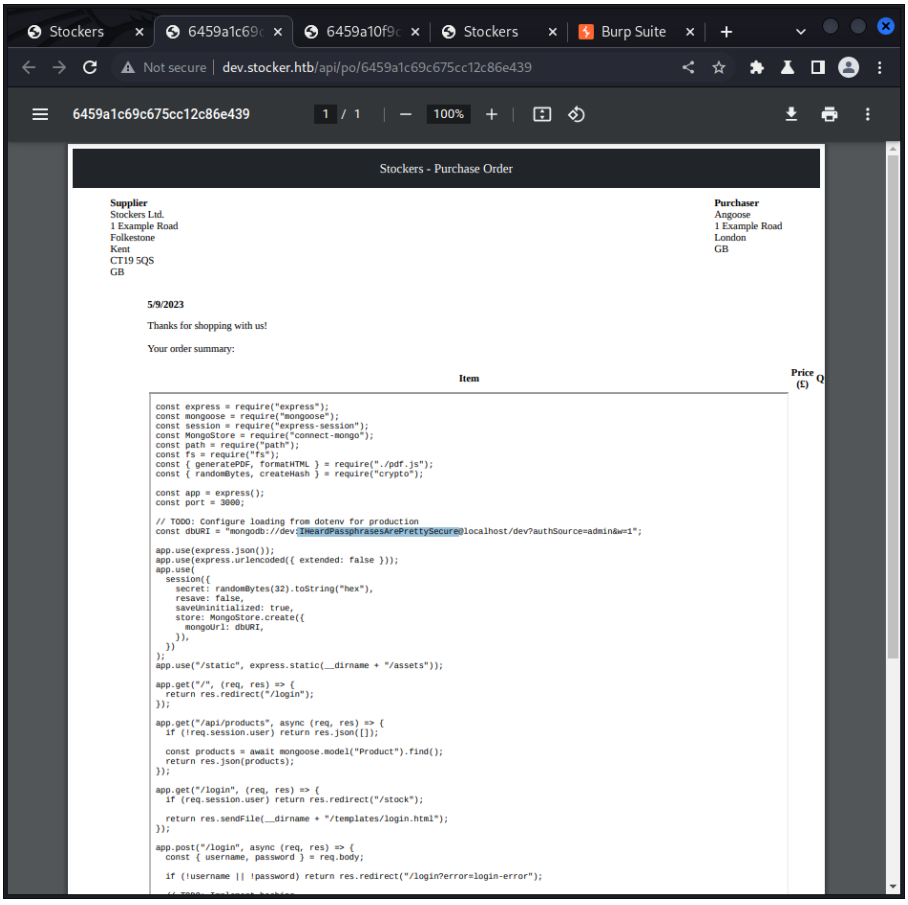
l.



m. With this information we reran the iframe injection with the root index.js

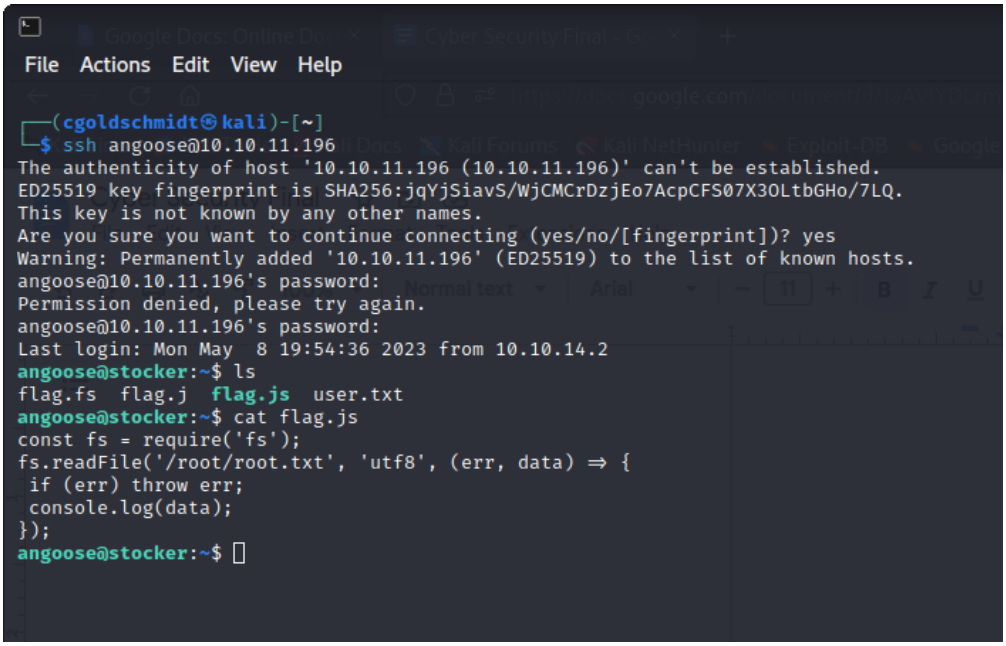
n. 

o. As a result we discovered the password IHeardPassphrasesArePrettySecure.

p. 

q. We then simply connected to the server by login in with the user angoose via ssh.

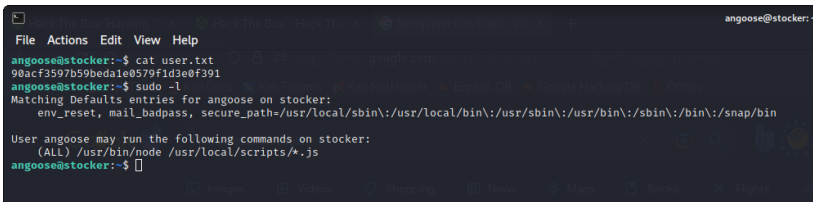
r.



G. Gaining sudo privileges

a. Using the command sudo -l we discovered the user “angoose” was able to execute all files with extension .js as a root user.

b.



c. Using path traversal manipulation, we were able to gain sudo privileges by traversing the associated path’s (reference the owasp article here)

d.

e. After executing the JS file we were able to gain “root” access a retrieve the flag

Recommendations

- We recommend rebuilding the stocker website entirely. The vulnerabilities associated with the NOSQL injection along with XSS make attacks incredibly powerful. We recommend hiring security professionals to design a website that will prevent these attacks.

- Risk-Analysis - based on these vulnerabilities we would put this risk at a high level and recommend these issues be addressed immediately.