Contents

Abstract

- Attacks on industrial control systems and critical infrastructure are on the rise. Important systems and devices like programmable logic controllers are at risk due to outdated technology and ad-hoc security measures.
- Virtual honeypots mitigate the unreasonable cost of hardware-replicated honeypots, these systems often suffer from a lack of authenticity due to proprietary hardware and network protocols.
- The result is a software tool which can be readily integrated into existing honeypot frame-works for improved performance finally, of the proposed proxy synchronization algorithms, templock and its minimal variant are found to provide the best overall performance.

Introduction

A honeypot is an information system resource whose value lies in unauthorized or illicit use of that resource.

They are different types of honeypot:

1. Production honeypots
2. Research honeypots
3. Database honeypot
4. Client Honeypots
5. Malware honeypots
6. High-interaction honeypots
7. Low-interaction honeypots
8. Medium-interaction honeypots

Literature Survey

- Jiang & Xu proposed the idea of catering honeypots architecture called BAIT-TRAP
- It, discussed ways at which virtual honeypots such as Honeyd can be camouflaged by designing a honey-pot that supports link latency in the order of one microsecond (μs) instead of the original one millisecond default, to avoid timing signature profiles that attackers might profile against the honeypot hence launching timing attacks against it
- Chowdhary and wagner uses machine learning technique as an approach for implementing an intelligent mechanism in dynamic honeypot .

Existing System

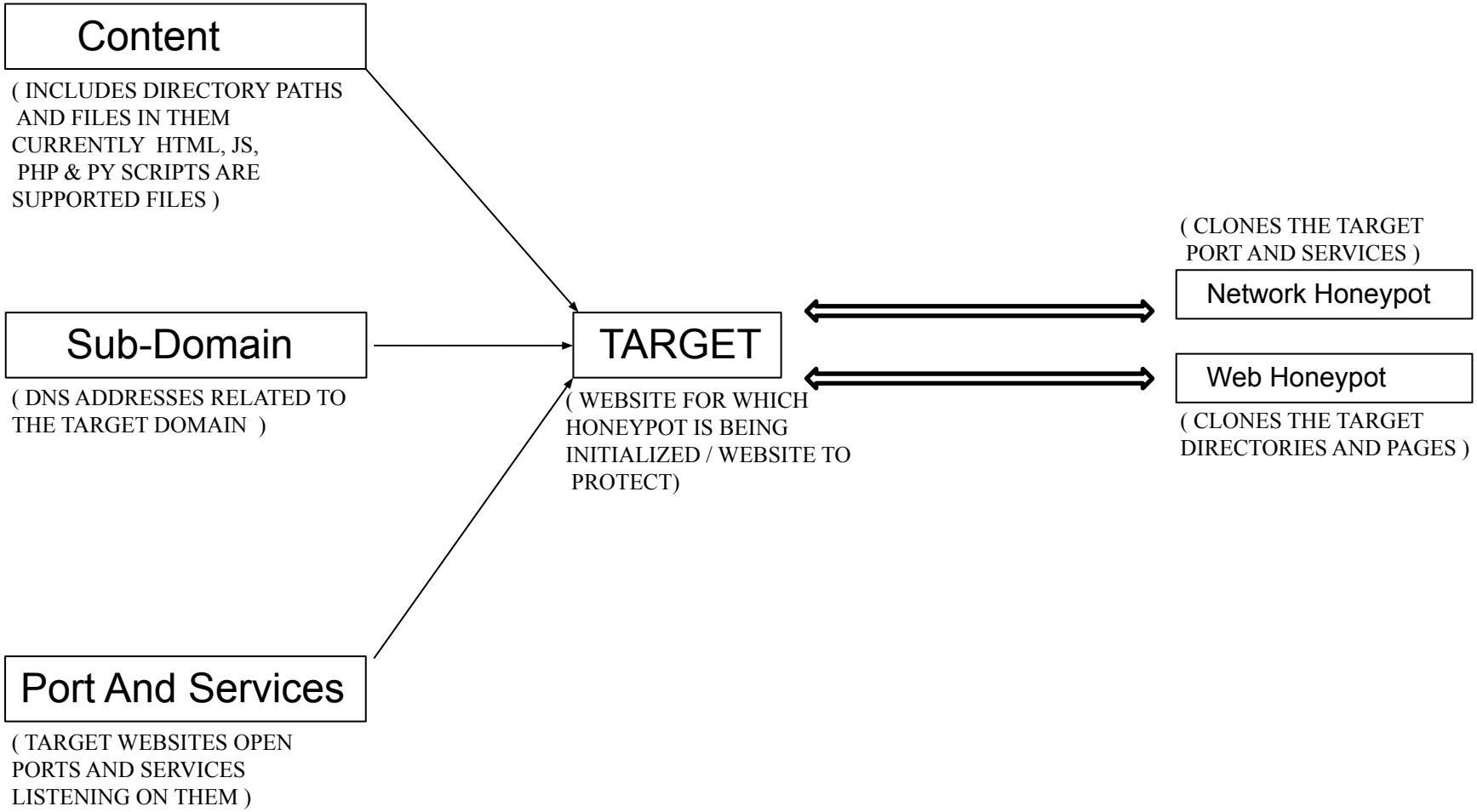
- Wordpot is a Wordpress honeypot which detects probes for plugins, themes, timthumb and other common files used to fingerprint a wordpress installation. It is developed by Gianluca Brindisi in the year 2012. Other then this no other web honeypot is created and this honeypot was also designed for wordpress only websites.
- Ironbee is a network honeypot which can only open one port mentioned by the user, it can’t clone the services running on them and can’t launch multiple instances which means we won’t be able to open multiple ports with this honeypot.

Proposed System

- Our project plan is to combine a web honeypot and network honeypot to make our local host machine's services and address to look exactly like the target website.
- First step will be to enumerate all the target's open ports and services running on them and also enumerate the web directories and files in them(html, php, js).
- We have designed a network honeypot which can open multiple ports according to the enumerated open ports from the target address.
- Npot also has a feature to check if the enumerated pages are valid or not and if they are valid then it will create the same folder structure in local machine and place the files(html, php, js) accordingly.
- For wordpress website we have created a separate program which is forked from wordpot, we have initialize a template generator which records the original valid templates with selenium and then use these templates in our program.
- At last all these activities will be recorded in log files and also the live traffic analysis can be done.

Design Methodology of Proposed Work

- This is a Linux based honeypot. Python ,ruby and shell scripting is used to develop Npot.
- The network based honeypot is designed to open the same ports is the local machine as on the target address. Then it also clones the services running on the port.
- Requests made to these ports are recorded in a separate log file and also displayed in terminal for live analysis.
- For wordpress related targets we have designed a forked version of wordpot honeypot.
- Net data is a tool used to monitor the local system's resources.
- The activities of network based honeypot and web based honeypot will be saved in separate log files and also the live activity will be displayed in seperate terminals.
- A firewall is also use to secure networks.


Content
(INCLUDES DIRECTORY PATHS AND FILES IN THEM CURRENTLY HTML, JS, PHP & PY SCRIPTS ARE SUPPORTED FILES)
Sub-Domain
(DNS ADDRESSES RELATED TO THE TARGET DOMAIN)
Port And Services
(TARGET WEBSITES OPEN PORTS AND SERVICES LISTENING ON THEM)
TARGET
(WEBSITE FOR WHICH HONEYPOT IS BEING INITIALIZED / WEBSITE TO PROTECT)
(CLONES THE TARGET PORT AND SERVICES)
Network Honeypot
Web Honeypot
(CLONES THE TARGET DIRECTORIES AND PAGES)

Resources and Tools

Hardware and Software Requirements:

Hardware

Processor: 2GHz

RAM: 4GB or Above

ROM: 500 MB

Graphics: 2GB

Software

OS Linux/Ubuntu

Language Used Python3, Ruby

Tools Required PIP, Code Editor, Virtual Apache Server, Net Data,Namp,Dirbuster

Terms Used In Flowcharts

1. Target port enumeration – Nmap is a tool which we will use to enumerate the target port and services listening on them.

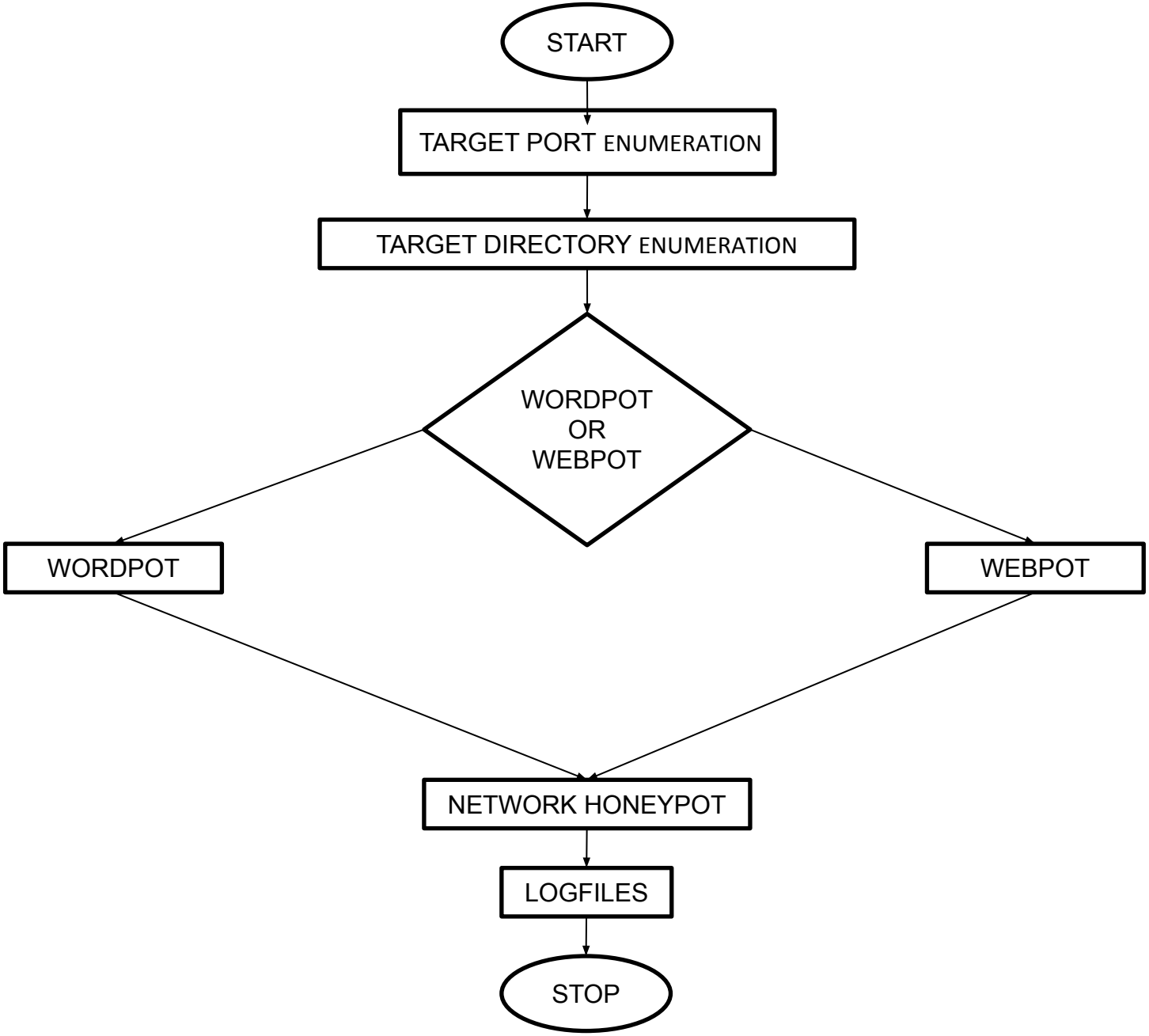
2. Target Directory enumeration – Dirbuster is tool which will be used to list the avail avail directory of the target website.

3. Wordpot – A forked version of the original Wordpot with some changes for better cloning purposes.

4. Webpot – A web honeypot which will be used to clone non-wordpress based websites. Working principle for both wordpot and webpot are similar.

5. Network honeypot – A network based honeypot used for cloning the open ports and the services according to nmap enumeration results.

6. Logfile – All the possible unknown requests will be stored here.


START
TARGET PORT ENUMERATION
TARGET DIRECTORY ENUMERATION
WORDPOT
OR
WEBPOT
WORDPOT
WEBPOT
NETWORK HONEYPOT
LOGFILES
STOP

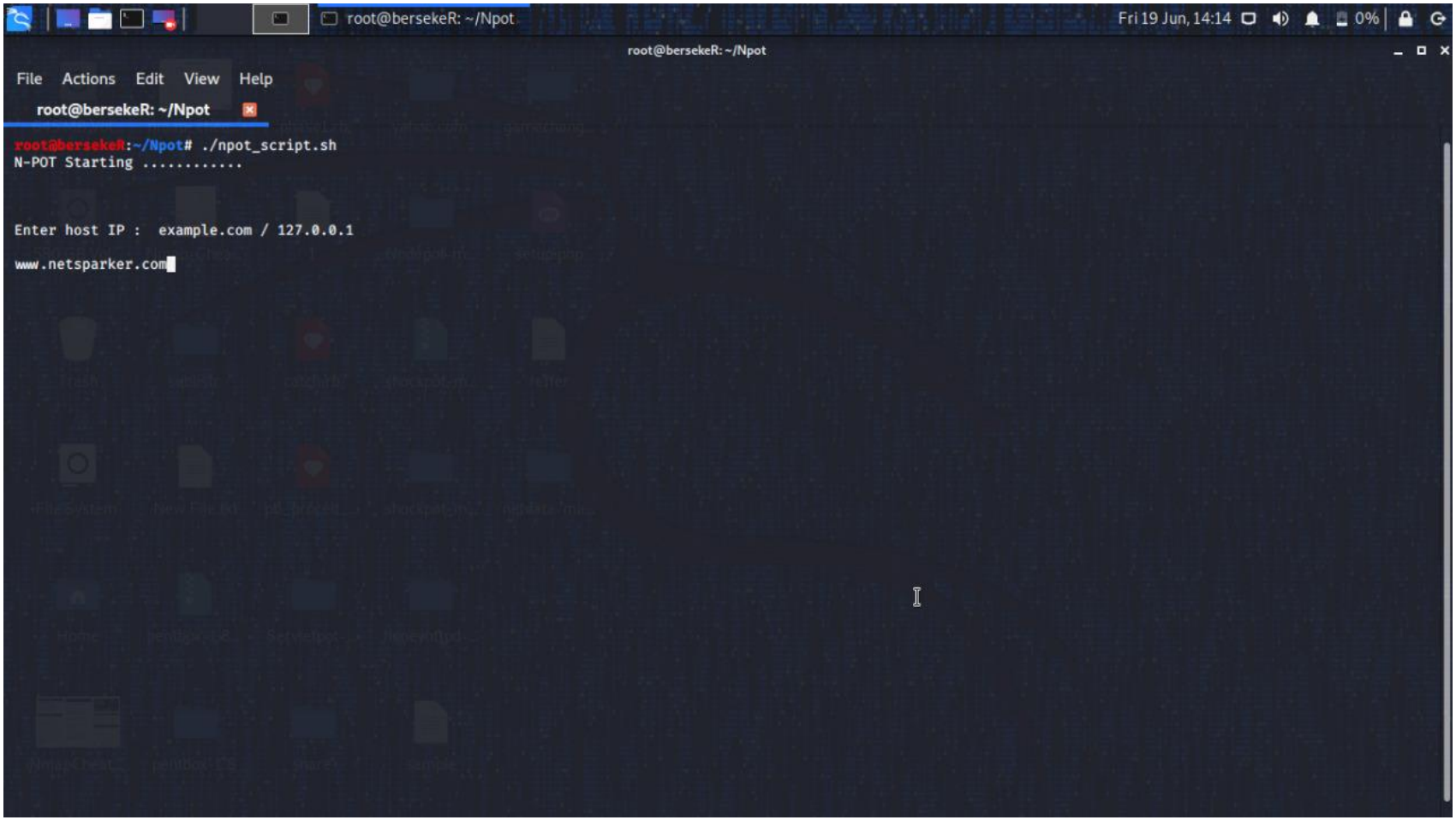
Implementation

- We have used apache web server for the local host machine and as mentioned previously the entire honeypot Npot is divided in two parts network honeypot and web honeypot.
- When the attacker will interact with our honeypot, we have applied VPN blocking or proxy blocking using third party.
- We have implemented Nmap tool to enumerate the target's open ports and the services listening on them.
- Dirbuster tool has been implemented to enumerate the website directory
- For wordpress based websites we have created a separate program which is originally forked from wordpot.
- We will be using sublistr tool to enumerate all the target's DNS record and copying them to the local host DNS record.

Implementation

- Net Data tool will be used to monitor the local machine's resources incase of a DOS attack scenario it will be useful.
- Net Data will be operated on port 5555 by default.
- The live traffic for the webserver of the local machine will be displayed in a terminal for live analysis and with filtering all the known/valid request all other request will be saved in a log file for further analysis and these unknown/invalid request will not be processed as well.
- A firewall will be deployed with modified rules to isolate the local host from the entire network.

Results with snap shots/Test Cases etc.



This above screenshot is the starting of the script .
Here we have used www.netsparker.com
For example as a host to initialize the honey-pot .

Results with snap shots/Test Cases etc.

```
root@bersekeR: ~/Npot
File  Actions  Edit  View  Help
root@bersekeR: ~/Npot

        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 118739  bytes 51101283 (48.7 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 118739  bytes 51101283 (48.7 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

usb0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.42.182  netmask 255.255.255.0  broadcast 192.168.42.255
        inet6 fe80::44b:92ff:fee9:c823  prefixlen 64  scopeid 0x20<link>
        ether 06:4b:92:e9:c8:23  txqueuelen 1000  (Ethernet)
        RX packets 18  bytes 2253 (2.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 18  bytes 3164 (3.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0


Running as user "root" and group "root". This could be dangerous.
1. usb0
2. lo (Loopback)
3. any
4. eth0
5. nflog
6. nfqueue
7. bluetooth0
8. ciscodump (Cisco remote capture)
9. dpauxmon (DisplayPort AUX channel monitor capture)
10. randpkt (Random packet generator)
11. sdjournal (systemd Journal Export)
12. sshdump (SSH remote capture)
13. udpdump (UDP Listener remote capture)

enter the interface to monitor
```

The above screenshot shows the options for the interface to listen on.
Here usb0 is selected as it is connected to internet.
This is important as while listing to the webserver and other ports the interface name is needed.

Results with snap shots/Test Cases etc.

```
13. udpdump (UDP Listener remote capture)

enter the interface to monitor
1
Scanning network for open ports and services  ..................
::::::::::::::::::::::::::::::::::::::::::::

Deploying Nmap ....

scan results:
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-19 14:14 HKT
Nmap scan report for www.netsparker.com (52.1.25.52)
Host is up (0.32s latency).
rDNS record for 52.1.25.52: ec2-52-1-25-52.compute-1.amazonaws.com
Not shown: 998 filtered ports
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 30.30 seconds
Launching Network Honeypot .............
Please wait we are cloning ports and services.....

cloning ports
Port 443 is cloned.

::::::::::::::::::
monitoring this port.


Logs saved into scanresults.txt
Warning: locale not supported by Xlib, locale set to C

Choose any one option :

1. Host is using wordpress
2. Other
2
```

```
Running as user "root" and group "root". This could be dangerous.
Capturing on 'usb0'
```

This screenshot shows the result of nmap scan on netsparker.com, below the output of nmap we can se that the ports are cloned

Except port 80, on the right side of the screen we can see the monitoring window for port 443 .

```
root@bersekeR: ~/Npot   tshark -i 1 -f"tcp port 44...   Screenshot   Fri 19 Jun, 14:16   0%
File  Actions  Edit  View  Help
root@bersekeR: ~/Npot

2020-06-19 14:16:08 (101 KB/s) - 'index.html' saved [34798/34798]

:::::::::::::::::::::::
disovered directories
/get-demo/
/get-demo/
/pricing/
/web-vulnerability-scanner/customers/
/blog/
/get-demo/
/get-demo/
/features/advanced/overview/
/integrations/
/pricing/
/case-studies/
/web-applications-advisories/
/resources/white-paper/
/blog/web-security/false-positives-web-application-security/
/blog/web-security/sql-injection-cheat-sheet/
/blog/web-security/getting-started-web-application-security/
/blog/web-security/content-security-policy/
/about/
/contact/
/support/
/jobs/
/resources/
/resellers/
/privacy-policy/
/data-protection-policy/
/sitemap/
:::::::::::::::::::::::::::::
Starting the Cloning process

URL transformed to HTTPS due to an HSTS policy
--2020-06-19 14:16:14--  https://www.netsparker.com/get-demo/
Resolving www.netsparker.com (www.netsparker.com)... 52.1.25.52
Connecting to www.netsparker.com (www.netsparker.com)|52.1.25.52|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 18820 (18K) [text/html]
```

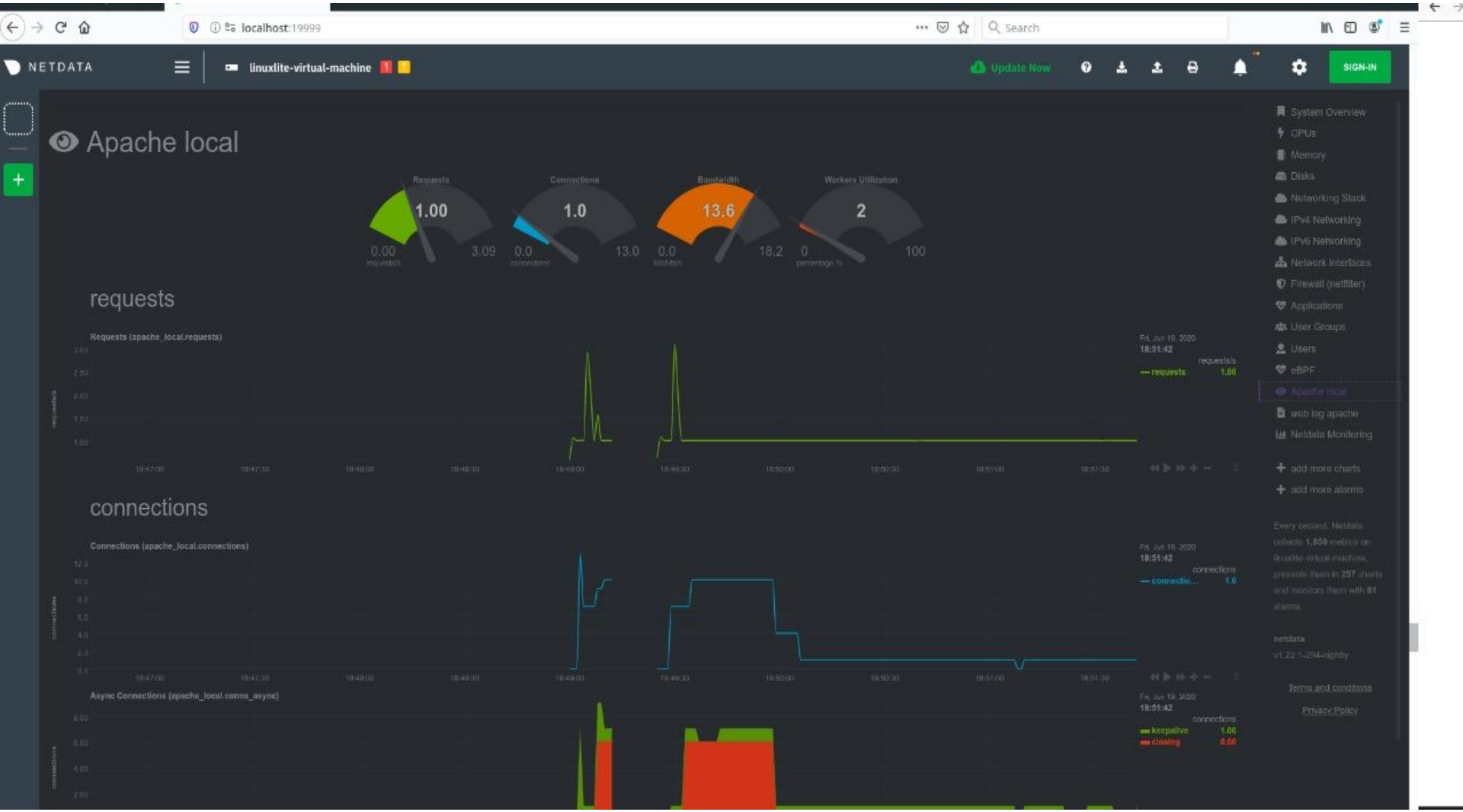
```
tshark -i1 -f"tcp port 443"
 986 35.404120057   52.1.25.52 â 192.168.42.182 TCP 1354 443 â 36380 [ACK] Seq=71533 Ack=878 Win=261632
 Len=1300 [TCP segment of a reassembled PDU]
 987 35.404143222 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [ACK] Seq=878 Ack=72833 Win=77184 Le
n=0
 988 35.404166107   52.1.25.52 â 192.168.42.182 TCP 1354 443 â 36380 [ACK] Seq=72833 Ack=878 Win=261632
 Len=1300 [TCP segment of a reassembled PDU]
 989 35.404186806 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [ACK] Seq=878 Ack=74133 Win=79744 Le
n=0
 990 35.404200229   52.1.25.52 â 192.168.42.182 TCP 1354 443 â 36380 [ACK] Seq=74133 Ack=878 Win=261632
 Len=1300 [TCP segment of a reassembled PDU]
 991 35.404208017 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [ACK] Seq=878 Ack=75433 Win=82304 Le
n=0
 992 35.404241647   52.1.25.52 â 192.168.42.182 TCP 1354 443 â 36380 [ACK] Seq=75433 Ack=878 Win=261632
 Len=1300 [TCP segment of a reassembled PDU]
 993 35.404250969 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [ACK] Seq=878 Ack=76733 Win=84992 Le
n=0
 994 35.404283409   52.1.25.52 â 192.168.42.182 TCP 1354 443 â 36380 [ACK] Seq=76733 Ack=878 Win=261632
 Len=1300 [TCP segment of a reassembled PDU]
 995 35.404294569 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [ACK] Seq=878 Ack=78033 Win=87552 Le
n=0
 996 35.404324394   52.1.25.52 â 192.168.42.182 TCP 1354 443 â 36380 [ACK] Seq=78033 Ack=878 Win=261632
 Len=1300 [TCP segment of a reassembled PDU]
 997 35.404334367 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [ACK] Seq=878 Ack=79333 Win=90112 Le
n=0
 998 35.404364210   52.1.25.52 â 192.168.42.182 TCP 1354 443 â 36380 [ACK] Seq=79333 Ack=878 Win=261632
 Len=1300 [TCP segment of a reassembled PDU]
 999 35.404371574 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [ACK] Seq=878 Ack=80633 Win=92800 Le
n=0
1000 35.404364533   52.1.25.52 â 192.168.42.182 TCP 1354 443 â 36380 [ACK] Seq=80633 Ack=878 Win=261632
 Len=1300 [TCP segment of a reassembled PDU]
1001 35.404384973 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [ACK] Seq=878 Ack=81933 Win=95360 Le
n=0
1002 35.404401659   52.1.25.52 â 192.168.42.182 TCP 1354 443 â 36380 [ACK] Seq=81933 Ack=878 Win=261632
 Len=1300 [TCP segment of a reassembled PDU]
1003 35.404408638 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [ACK] Seq=878 Ack=83233 Win=97920 Le
n=0
1004 35.404442583   52.1.25.52 â 192.168.42.182 TLSv1.2 698 Application Data
1005 35.404450085 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [ACK] Seq=878 Ack=83877 Win=100480 L
en=0
1006 35.405472352 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [FIN, ACK] Seq=878 Ack=83877 Win=100
480 Len=0
1007 35.437346174 192.168.42.182 â 52.1.25.52   TCP 74 36382 â 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
 SACK_PERM=1 TSval=4121652259 TSecr=0 WS=128
1008 35.688905770   52.1.25.52 â 192.168.42.182 TCP 54 443 â 36380 [FIN, ACK] Seq=83877 Ack=879 Win=261
632 Len=0
1009 35.688951857 192.168.42.182 â 52.1.25.52   TCP 54 36380 â 443 [ACK] Seq=879 Ack=83878 Win=100480 L
en=0
1010 35.728900467   52.1.25.52 â 192.168.42.182 TCP 66 443 â 36382 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len
=0 MSS=1300 WS=256 SACK_PERM=1
1011 35.728957854 192.168.42.182 â 52.1.25.52   TCP 54 36382 â 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0
1012 35.730109896 192.168.42.182 â 52.1.25.52   TLSv1 571 Client Hello
```

The above screenshot shows that the directories of netsparker.com have been enumerated and cloning process has also started.

```
root@bersekeR: ~/Npot   tshark -i 1 -f"tcp port ...   tshark -i 1 -f"tcp port ...   [www.netsparker.com...   Fri 19 Jun, 14:19   0%

root@bersekeR: ~/Npot
File  Actions  Edit  View  Help
root@bersekeR: ~/Npot

2020-06-19 14:18:33 ERROR 404: Not Found.

mv: cannot move 'index.html' to '/var/www/html/company/netsparker/index.html': No such fi
mkdir: cannot create directory '/var/www/html/netsparker': File exists
URL transformed to HTTPS due to an HSTS policy
--2020-06-19 14:18:33--  https://www.netsparker.com/netsparker
Resolving www.netsparker.com (www.netsparker.com)... 52.1.25.52
Connecting to www.netsparker.com (www.netsparker.com)|52.1.25.52|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://www.netsparker.com/web-vulnerability-scanner/ [following]
--2020-06-19 14:18:34--  https://www.netsparker.com/web-vulnerability-scanner/
Reusing existing connection to www.netsparker.com:443.
HTTP request sent, awaiting response... 200 OK
Length: 33533 (33K) [text/html]
Saving to: 'index.html'

index.html                                  100%[=========================================

2020-06-19 14:18:35 (101 KB/s) - 'index.html' saved [33533/33533]


Finished recreating directory structure


Finished cloning webpages
.........................
Finished Cloning
::::::::::::::::

Montitoring the web-server on port 80

::::::::::::::::
 Netdata started

2020-06-19 14:18:35: netdata INFO  : MAIN : SIGNAL: Not enabling reaper

scanning for additional hidden directories

Finished setting up Honeypot
```

```
tshark -i1 -f"tcp port 80"
342 46.722111826 192.168.42.182 à 52.1.25.52   HTTP 180 GET /~test HTTP/1.1
343 47.040586856   52.1.25.52 à 192.168.42.182 HTTP 1191 HTTP/1.1 301 Moved Permanently  (text/html)
344 47.040753673 192.168.42.182 à 52.1.25.52   TCP 54 43644 à 80 [ACK] Seq=15522 Ack=138128 Win=209152
Len=0
345 47.041334268 192.168.42.182 à 52.1.25.52   HTTP 179 GET /~tmp HTTP/1.1
346 47.360715173   52.1.25.52 à 192.168.42.182 HTTP 1189 HTTP/1.1 301 Moved Permanently  (text/html)
347 47.361222530 192.168.42.182 à 52.1.25.52   HTTP 180 GET /~user HTTP/1.1
348 47.690544665   52.1.25.52 à 192.168.42.182 HTTP 1191 HTTP/1.1 301 Moved Permanently  (text/html)
349 47.691116100 192.168.42.182 à 52.1.25.52   HTTP 185 GET /~webmaster HTTP/1.1
350 48.001586068   52.1.25.52 à 192.168.42.182 HTTP 1201 HTTP/1.1 301 Moved Permanently  (text/html)
351 48.002173743 192.168.42.182 à 52.1.25.52   HTTP 179 GET /~www HTTP/1.1
352 48.330507597   52.1.25.52 à 192.168.42.182 HTTP 1189 HTTP/1.1 301 Moved Permanently  (text/html)
353 48.330957374 192.168.42.182 à 52.1.25.52   HTTP 176 GET /0 HTTP/1.1
354 48.654601238   52.1.25.52 à 192.168.42.182 HTTP 1183 HTTP/1.1 301 Moved Permanently  (text/html)
355 48.655110208 192.168.42.182 à 52.1.25.52   HTTP 177 GET /00 HTTP/1.1
356 48.961511372   52.1.25.52 à 192.168.42.182 HTTP 1185 HTTP/1.1 301 Moved Permanently  (text/html)
357 48.961985500 192.168.42.182 à 52.1.25.52   HTTP 177 GET /01 HTTP/1.1
358 49.281544240   52.1.25.52 à 192.168.42.182 HTTP 1185 HTTP/1.1 301 Moved Permanently  (text/html)
359 49.282150724 192.168.42.182 à 52.1.25.52   HTTP 177 GET /02 HTTP/1.1
360 49.610507194   52.1.25.52 à 192.168.42.182 HTTP 1185 HTTP/1.1 301 Moved Permanently  (text/html)
361 49.611076795 192.168.42.182 à 52.1.25.52   HTTP 177 GET /03 HTTP/1.1
362 49.920585684   52.1.25.52 à 192.168.42.182 HTTP 1185 HTTP/1.1 301 Moved Permanently  (text/html)
363 49.920887779 192.168.42.182 à 52.1.25.52   HTTP 177 GET /04 HTTP/1.1
364 50.254618104   52.1.25.52 à 192.168.42.182 HTTP 1185 HTTP/1.1 301 Moved Permanently  (text/html)
365 50.255089146 192.168.42.182 à 52.1.25.52   HTTP 177 GET /05 HTTP/1.1
366 50.560549046   52.1.25.52 à 192.168.42.182 HTTP 1185 HTTP/1.1 301 Moved Permanently  (text/html)
367 50.561123983 192.168.42.182 à 52.1.25.52   HTTP 177 GET /06 HTTP/1.1
368 50.879542025   52.1.25.52 à 192.168.42.182 HTTP 1185 HTTP/1.1 301 Moved Permanently  (text/html)
369 50.879744040 192.168.42.182 à 52.1.25.52   HTTP 177 GET /07 HTTP/1.1
370 51.201498496   52.1.25.52 à 192.168.42.182 HTTP 1185 HTTP/1.1 301 Moved Permanently  (text/html)
371 51.201958538 192.168.42.182 à 52.1.25.52   HTTP 177 GET /08 HTTP/1.1
372 51.521529111   52.1.25.52 à 192.168.42.182 HTTP 1185 HTTP/1.1 301 Moved Permanently  (text/html)
373 51.522318012 192.168.42.182 à 52.1.25.52   HTTP 177 GET /09 HTTP/1.1
374 51.840521069   52.1.25.52 à 192.168.42.182 HTTP 1185 HTTP/1.1 301 Moved Permanently  (text/html)
375 51.841246311 192.168.42.182 à 52.1.25.52   HTTP 176 GET /1 HTTP/1.1
376 52.161556570   52.1.25.52 à 192.168.42.182 HTTP 1183 HTTP/1.1 301 Moved Permanently  (text/html)
377 52.162102056 192.168.42.182 à 52.1.25.52   HTTP 177 GET /10 HTTP/1.1
378 52.480502389   52.1.25.52 à 192.168.42.182 HTTP 1185 HTTP/1.1 301 Moved Permanently  (text/html)
379 52.480914580 192.168.42.182 à 52.1.25.52   HTTP 178 GET /100 HTTP/1.1
380 52.814507989   52.1.25.52 à 192.168.42.182 HTTP 1187 HTTP/1.1 301 Moved Permanently  (text/html)
381 52.814788104 192.168.42.182 à 52.1.25.52   HTTP 179 GET /1000 HTTP/1.1
382 53.102561897   52.1.25.52 à 192.168.42.182 HTTP 1189 HTTP/1.1 301 Moved Permanently  (text/html)
383 53.103107707 192.168.42.182 à 52.1.25.52   HTTP 179 GET /1001 HTTP/1.1
384 53.394694144   52.1.25.52 à 192.168.42.182 HTTP 1189 HTTP/1.1 301 Moved Permanently  (text/html)
385 53.395263511 192.168.42.182 à 52.1.25.52   HTTP 178 GET /101 HTTP/1.1
386 53.704534143   52.1.25.52 à 192.168.42.182 HTTP 1187 HTTP/1.1 301 Moved Permanently  (text/html)
387 53.704914201 192.168.42.182 à 52.1.25.52   HTTP 178 GET /102 HTTP/1.1
388 53.992489723   52.1.25.52 à 192.168.42.182 HTTP 1187 HTTP/1.1 301 Moved Permanently  (text/html)
389 53.992900721 192.168.42.182 à 52.1.25.52   HTTP 178 GET /103 HTTP/1.1
390 54.400500673   52.1.25.52 à 192.168.42.182 HTTP 1187 HTTP/1.1 301 Moved Permanently  (text/html)
391 54.401132823 192.168.42.182 à 52.1.25.52   HTTP 177 GET /11 HTTP/1.1
```

This screenshot shows the cloning process is completed, net data has started and on the right hand side we can see the panel for monitoring on webserver has been spawned as well.



The above screenshot shows net data is working properly on tits assigned port.

Netsparker | Web Applic... root@berseker: ~/Npot Fri 19 Jun, 14:21 0%

Netsparker | Web Application Security Solution - Mozilla Firefox

Netsparker | Web Applic

127.0.0.1

Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB GHDB MSFU

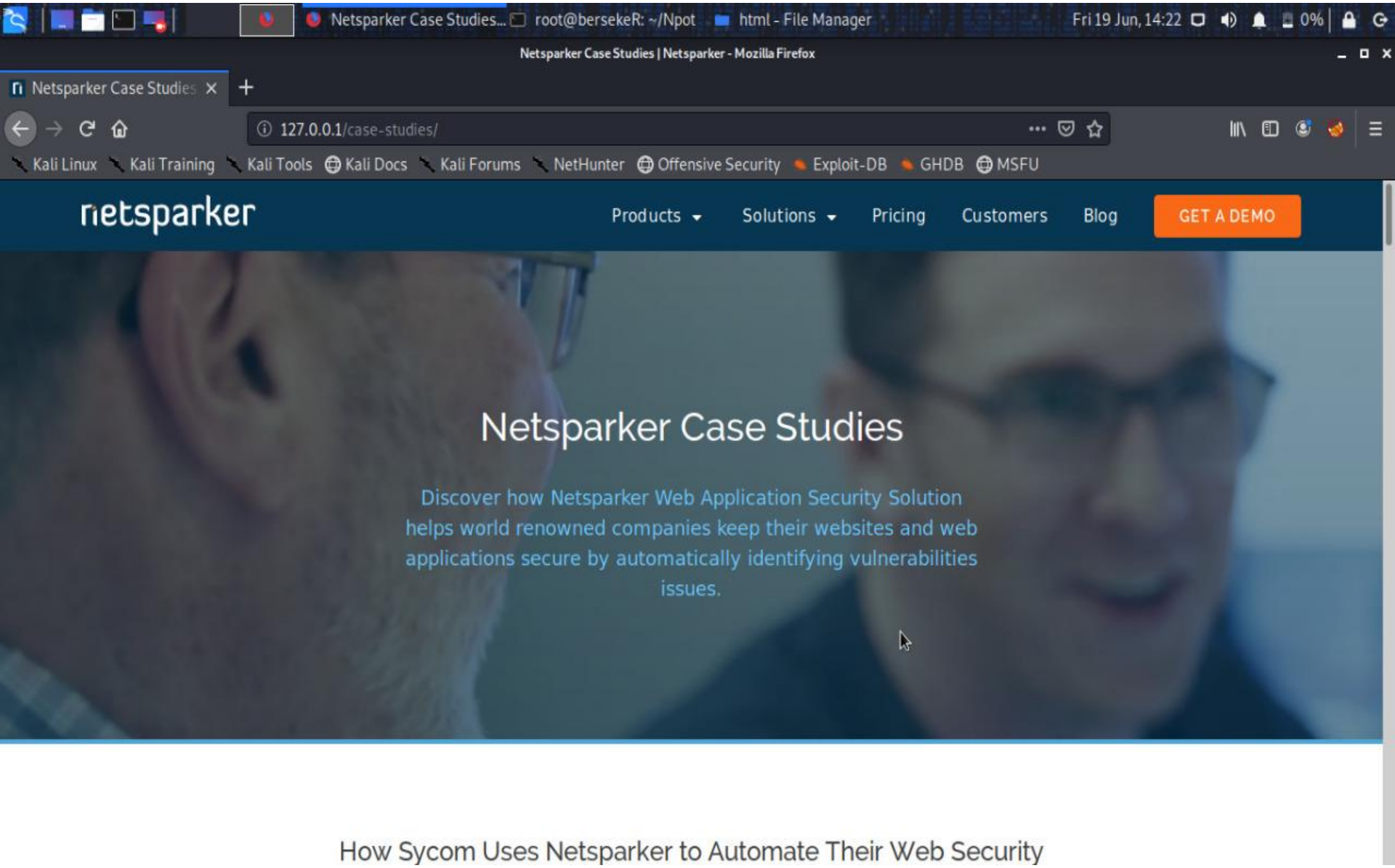
netsparker

Products Solutions Pricing Customers Blog GET A DEMO

A single platform for **all your web security needs**

Netsparker is a fully integrated, scalable, multi-user web application with built-in workflow and reporting tools.

GET A DEMO



The above two screenshot shows that the website is cloned successfully on our local host.
If we check the URL bar of these two screenshots, it clearly shows that the web-directories and contents are cloned successfully.

```
1
Scanning network for open ports and services  ..................
::::::::::::::::::::::::::::::::::::::::::::

Deploying Nmap ....

scan results:
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-19 14:14 HKT
Nmap scan report for www.netsparker.com (52.1.25.52)
Host is up (0.32s latency).
rDNS record for 52.1.25.52: ec2-52-1-25-52.compute-1.amazonaws.com
Not shown: 998 filtered ports
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 30.30 seconds
Launching Network Honeypot .............
Please wait we are cloning ports and services.....

cloning ports
Port 443 is cloned.

::::::::::::::::::
monitoring this port.



Logs saved into scanresults.txt
Warning: locale not supported by Xlib, locale set to C

Choose any one option :

1. Host is using wordpress
2. Other
2
STARTING web-honeypot ....................
```

```
root@bersekeR:~# nmap 127.0.0.1
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-19 14:26 HKT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000090s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 0.14 seconds
root@bersekeR:~#
```

In this above screenshot on the right terminal we have executed nmap on our local host , and on the left side we have nmap
Results for the host address. Comparing both of them we can successfully conclude that our honeypot has not only cloned port
But the services as well.

Conclusion

- In this paper we have proposed a honeypot system for websites. This honeypot mainly focuses on cloning the target as much as possible as by doing this it will confuse the attacker and also works as an excellent bait system.
- To summarize everything Npot is a combination of a network honeypot and web honeypot, network honeypot is used to mimic web services and open ports of the target while the web honeypot is used to mimic the directory structure of the target website and also clones the content avail avail in the target directories.
- Future goal of this project will be to clone more sophisticated web application, build a ML algorithm that can differentiate between false positive and true positive attacks with help of text classification and pattern recognition algorithms, we would also add an additional features of cloning security certificates.
- This project has also been uploaded on github as a contribution towards open source community.