Portscanning

Author: Lily Li
Conferred with: jeff Ondich

Part 1 Sources:
https://www.nirsoft.net/whois_servers_list.html
https://en.wikipedia.org/wiki/WHOIS
Part 2 Sources:
https://accedian.com/blog/dns-query-main-types/

1. Passive information gathering

A: Domain name:
mandaringardennorthfield.com

B: IP Address:
137.22.198.41

C: Domain Expiration Date:
Sun, 16 Oct 2022 16:40:10 +0000

D: More Information
With command

whois mandaringardennorthfield.com

We get:
1: information of the Registrar (the organization that registered the domain name)
2: information of the Registrant (individual who registers domain names through registrars). In this case, Mandarin Garden's contact information is hidden given they used domain privacy service,

I. Information of the Registrar, VeriSign, the authoritative registry for the *.com* domains.

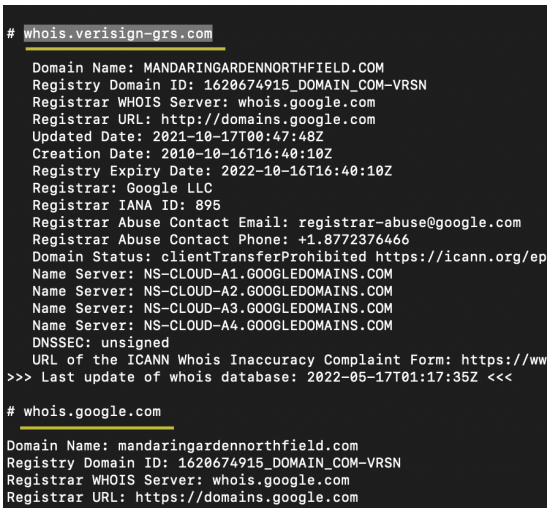
VeriSign Global Registry Services
address: 12061 Bluemont Way, Reston Virginia 20190, United States
phone: +1 703 925-6999
fax-no: +1 703 948 3978
e-mail: info@verisign-grs.com
created: 1985-01-01
changed: 2017-10-05

source: IANA

Other information such as name servers, ds-rdata….etc is not listed.

ii.Information of the Registrant, Mandarin Garden.

Mandarin Garden used Domain Privacy Services, which hides their public contact information in the WHOIS system and displays the information of a proxy. Here you can see the proxy registrant is *Contact Privacy Inc. Customer 7151571251.* I believe we can understand it as *Contact Privacy Inc* is the organization providing the Privacy Service and *7151571251* is a number assigned to Mandarin Garden.



Interestingly, when we make the query, we get results back from 2 WHOIS servers, *whois.google.com*, and *whois.verisign-grs.com*, both providing information about registered domains. The following information is from the *whois.google.com* server.

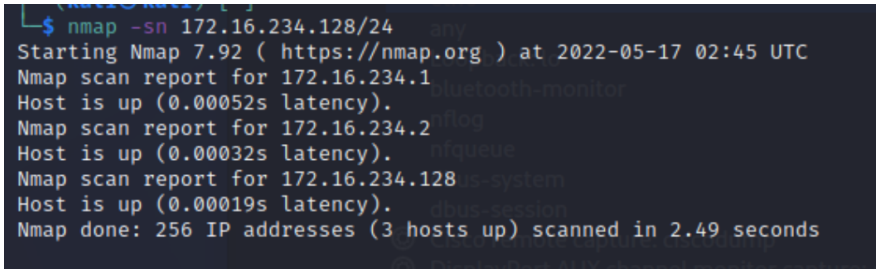
Registry Registrant ID:
Registrant Name: Contact Privacy Inc. Customer 7151571251
Registrant Organization: Contact Privacy Inc. Customer 7151571251
Registrant Street: 96 Mowat Ave
Registrant City: Toronto
Registrant State/Province: ON
Registrant Postal Code: M4K 3K1
Registrant Country: CA
Registrant Phone: +1.4165385487
Registrant Phone Ext:
Registrant Fax:
Registrant Fax Ext:
Registrant Email: https://domains.google.com/contactregistrant?domain=mandaringardennorthfield.com
… etc

Here you can see the proxy also replaces mandarin garden's private email with an anonymous email that can be routed to them, so you could still contact them. The command also returns.

2. Host detection

1: Local Network

A: Kali's IP (nmap -sn 172.16.234.12/24)



I-II. IP addresses on Kali local network and entities:

172.16.234.1 (name server Kali created)
172.16.234.2 (name server Kali created)
172.16.234.128 (Kali)

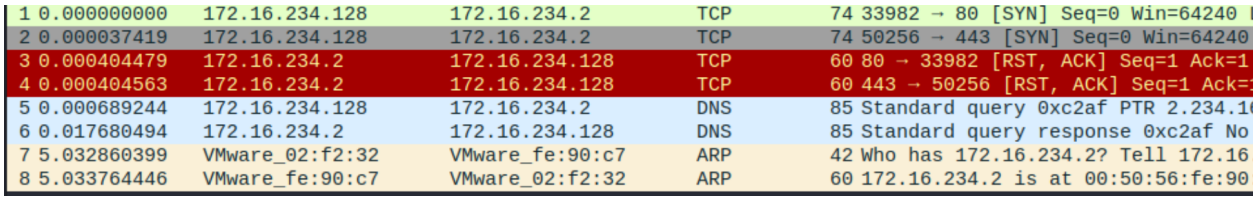
Although I was also running Metasploitable on my laptop, Metaslpoitable and Kali did not choose the same private network IP range (Metasploitable IP address is 10.0.2.15) and the only Device that showed up on nmap was Kali itself. I talked with Jeff and we came to a conclusion that for some reason VMware decided to create a local name server, that is what 172.16.234.1/172.16.234.2 are, and 172.16.234.128 is the Kali linux virtual machine itself.

III. Steps did nmap take

In the case on my kali server, for each possible IP address,
0: TCP Handshake attempt.
1: A DNS request is sent from local (172.16.234.128) to the name server (172.16.234.2), to ask for the domain name associated with the IP address.
2: The name server then sends back a response, either with an answer or not.
3: In order to get the MAC address, which is associated with the actual device, my device broadcasts to all devices on the local network a ARP request. "Who has the target IP address? Tell my IP address".

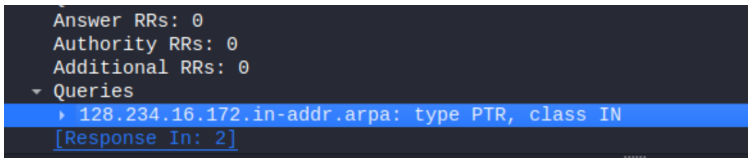
4: All devices but the target device, associated with the target IP address, ignores the ARP request. The target IP address responds my device (my IP address) with it's unique MAC address.

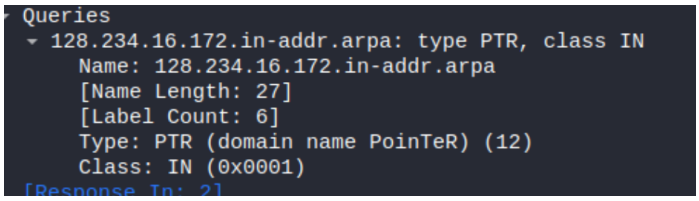
IV. Example: candidate 172.16.234.2



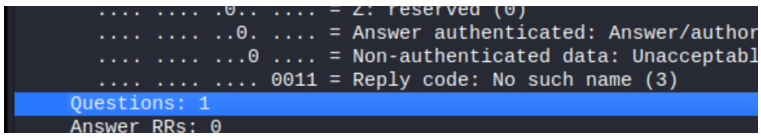
First of all an attempt for a TCP handshake is initiated from me to the target. In this case, the target doesn't listen on port 80 (HTTP) or port 443(HTTPS), so the target sends back a [RST,ACK]. After talking to Jeff, we believe this means that: the host exists, just it's not listening on the requested port.

After the attempt for TCP, a DNS is sent from me (172.16.234.128) to the name server (172.16.234.2) which in this case also turned out to be the candidate.
I asked for a reverse query requesting the domain name corresponding to the IP address (172.16.234.2) I provide. The type PTR of this query identifies the purpose of the request.





A response is sent back, with no-such-name reply code (0011) and 0 Answer PRs, meaning no answers (so no domain name)



And then, an ARP request is broadcasted out, for the MAC address related to the target IP address (172.16.234.2). Note address in the "Target Mac Address" below. That special (00:00:00_00:00:00) is the MAC broadcast address, which goes to every device on the local network.

```
Opcode: request (1)
Sender MAC address: VMware_02:f2:32 (00:0c:29:02:f2:32)
Sender IP address: 172.16.234.128
Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
Target IP address: 172.16.234.2
```

A response is sent back from the target (172.16.234.2) with it's MAC address.

```
Opcode: reply (2)
Sender MAC address: VMware_fe:90:c7 (00:50:56:fe:90:c7)
Sender IP address: 172.16.234.2
Target MAC address: VMware_02:f2:32 (00:0c:29:02:f2:32)
```

```
8 5.033764446   VMware_fe:90:c7   VMware_02:f2:32   ARP   60 172.16.234.2 is at 00:50:56:fe:90:c
```

Note: if I attempt to *nmap -sn [my own IP address]*, the process is the same, just that TCP handshake will not be initiated.

B: 137.22.4.0/24 network

I-II. IP addresses on Kali local network and entities:

A very long list of IP addresses (76 hosts in total)
Most probably student/faculty/lab computers in the network, or maybe also some webservers.
(We were able to find Jeff's computer too)

137.22.4.5
137.22.4.15
137.22.4.17
137.22.4.21
137.22.4.30
137.22.4.31
137.22.4.32
137.22.4.34
137.22.4.35
137.22.4.37
137.22.4.38
137.22.4.39
137.22.4.40
137.22.4.41....

```
Nmap scan report for 137.22.4.5
Host is up (0.013s latency).
Nmap scan report for 137.22.4.15
Host is up (0.0074s latency).
Nmap scan report for 137.22.4.17
Host is up (0.0045s latency).
Nmap scan report for 137.22.4.21
Host is up (0.0085s latency).
Nmap scan report for 137.22.4.30
Host is up (0.0081s latency).
Nmap scan report for 137.22.4.31
Host is up (0.0081s latency).
Nmap scan report for 137.22.4.32
Host is up (0.0094s latency).
Nmap scan report for 137.22.4.34
Host is up (0.0095s latency).
Nmap scan report for 137.22.4.35
Host is up (0.0084s latency).
Nmap scan report for 137.22.4.37
Host is up (0.0088s latency).
Nmap scan report for 137.22.4.38
Host is up (0.0088s latency).
Nmap scan report for 137.22.4.39
Host is up (0.011s latency).
Nmap scan report for 137.22.4.40
Host is up (0.011s latency).
Nmap scan report for 137.22.4.41
Host is up (0.011s latency).
Nmap scan report for 137.22.4.42
Host is up (0.0094s latency).
Nmap scan report for 137.22.4.43
Host is up (0.0083s latency).
Nmap scan report for 137.22.4.46
Host is up (0.011s latency).
Nmap scan report for 137.22.4.49
Host is up (0.0096s latency).
```

III. Steps did nmap take

Very similar to the Kali scenario, just no ARP request/replies (No MAC address needed), given its a remote network

0: TCP handshake attempt

1: A DNS request is sent from local (172.16.234.128) to the name server (172.16.234.2), to ask for the domain name associated with the IP address.

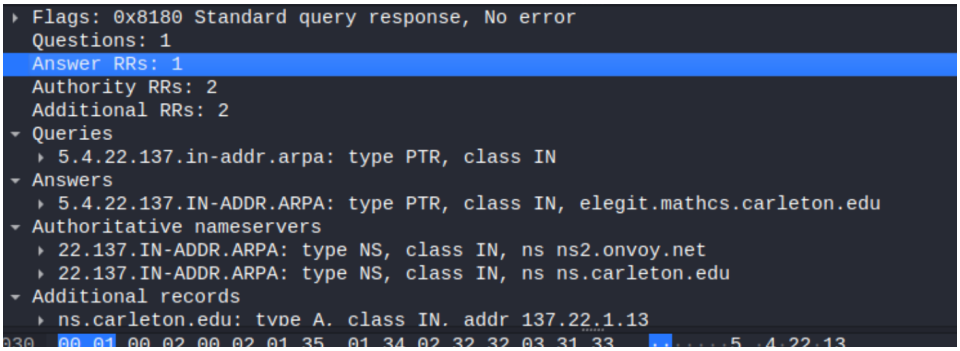
2: The name server then sends back a response, either with an answer or not.

IV. Example: candidate 137.22.4.5

1	0.000000000	172.16.234.128	137.22.4.5	TCP	74 40988 → 80 [SYN] Seq=0 Win=64246
2	0.000083237	172.16.234.128	137.22.4.5	TCP	74 34506 → 443 [SYN] Seq=0 Win=6424
3	0.012291580	137.22.4.5	172.16.234.128	TCP	60 80 → 40988 [SYN, ACK] Seq=0 Ack=
4	0.012291663	137.22.4.5	172.16.234.128	TCP	60 443 → 34506 [SYN, ACK] Seq=0 Ack
5	0.012333115	172.16.234.128	137.22.4.5	TCP	54 40988 → 80 [ACK] Seq=1 Ack=1 Win
6	0.012354778	172.16.234.128	137.22.4.5	TCP	54 34506 → 443 [ACK] Seq=1 Ack=1 Wi
7	0.012371275	172.16.234.128	137.22.4.5	TCP	54 40988 → 80 [RST, ACK] Seq=1 Ack=
8	0.012400354	172.16.234.128	137.22.4.5	TCP	54 34506 → 443 [RST, ACK] Seq=1 Ack
9	0.013091535	172.16.234.128	172.16.234.2	DNS	83 Standard query 0xfe04 PTR 5.4.22
10	0.019943858	172.16.234.2	172.16.234.128	DNS	222 Standard query response 0xfe04 P

First of all, TCP handshake. Surprisingly, (137.22.4.5) target does listen on port 80 and 443! So it is a web server! And handshake is completed and connection is established.

Then a DNS request is sent from me (172.16.234.128) to the name server (172.16.234.2) again for a reverse query (get domain name from IP address) with format very similar to the previous example. The difference is the DNS response, it turns out that this IP address does have a domain name: *elegit.mathcs.carleton.edu*



3. Port Scan

A: Metasploitable ports:

Three columns: Port number / status/ corresponding service

```
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
```

B: Database servers:

PostgreSQL :5432

MySQL: 3306

C: RSA ssh-hostkey Value and Purpose:

56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3

The host key is the ssh server's public key. The host key is used by the client to authenticate the server’s identity when communicating with it (probably by decrypting the authentication message sent from server)

Source: https://www.vandyke.com/solutions/host_keys/host_keys.pdf

D: Novel Port: 21 (FTP)

Purpose: File Transfer Protocol, one of the oldest Internet protocols for file transfer operations. Having an open FTP-port can be dangerous as it is like a drop-box for nasty software. Management responsibility is required if you want to open this port. You need to be sure that the contents of the incoming directory is not available for outgoing download without explicit permission.

Source: grc.com/port_21.htm