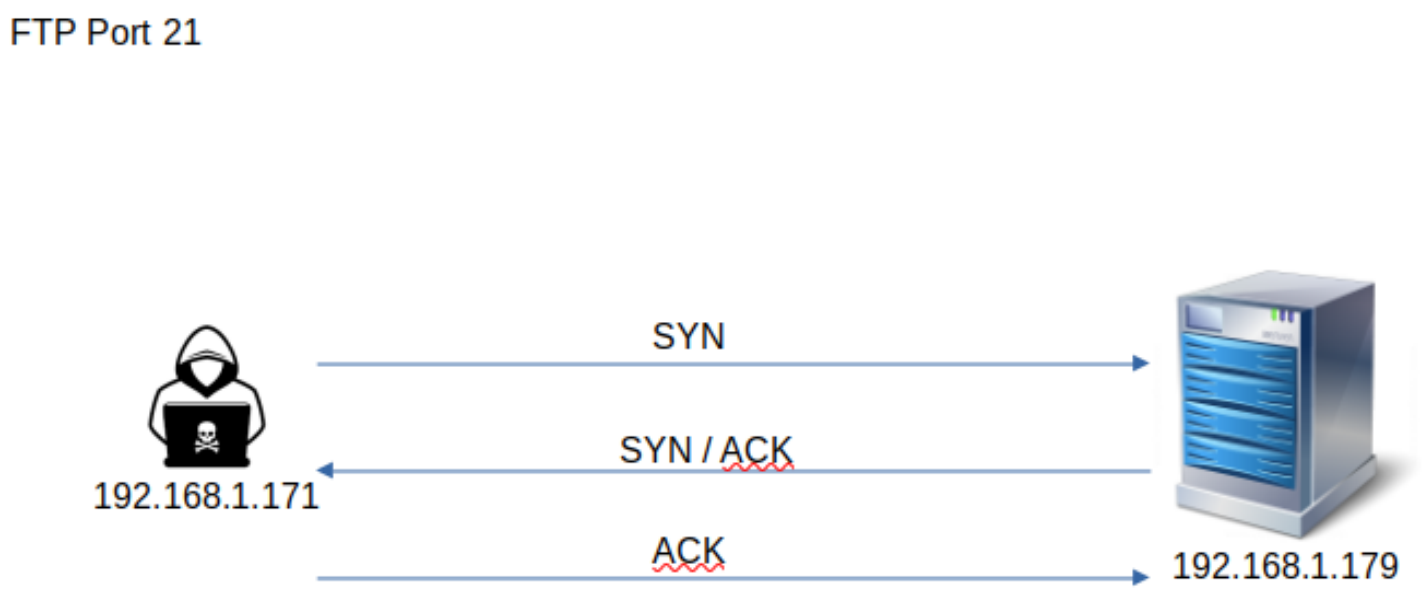
Nmap Decoy Scanning

Using nmap, it's possible to obfuscate port scans by cloaking them within decoy IP addresses. This is referred to in nmap as a decoy scan.

I'm going to demonstrate how this is done.

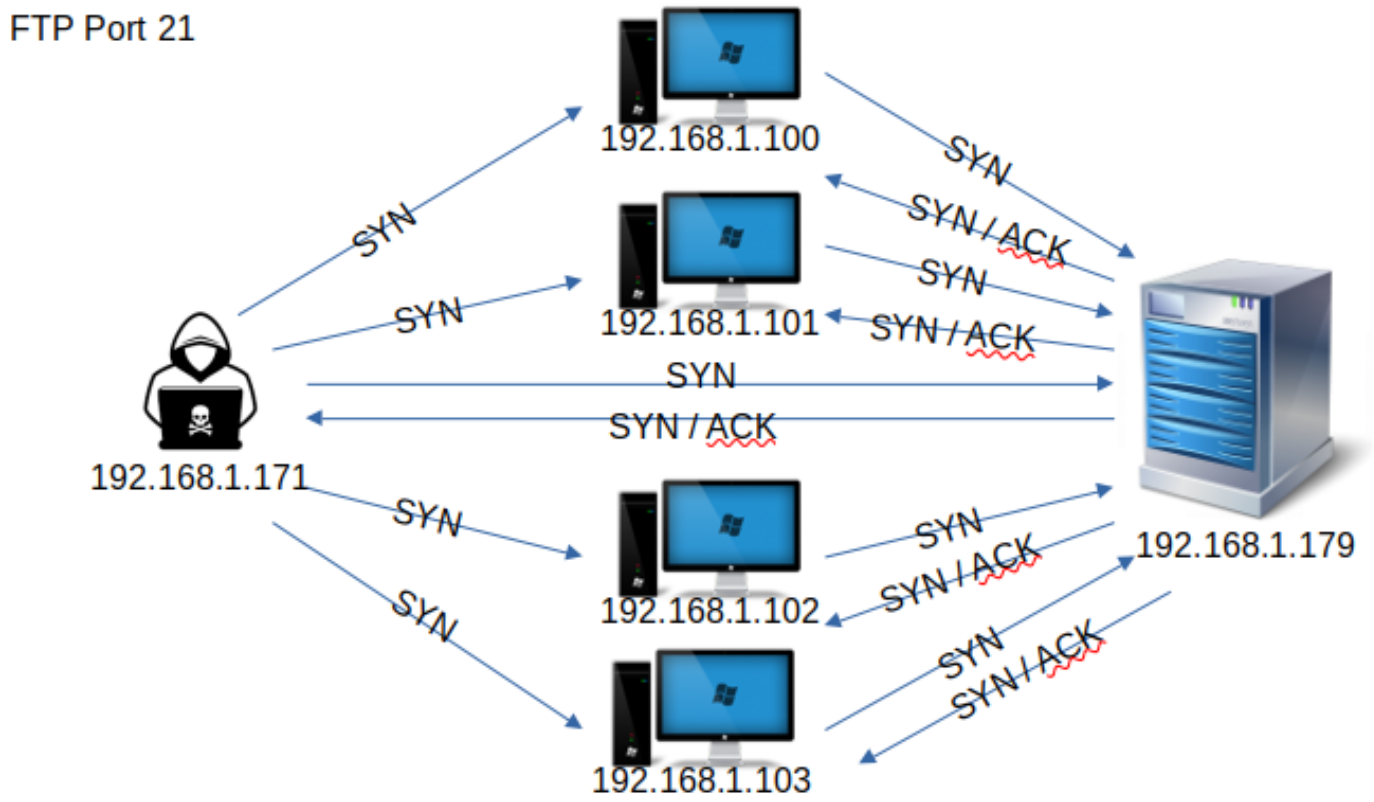
Let's say that an attacker, sitting on IP 192.168.1.171, wants to check to see if port 21 (FTP) is open on a server at IP 192.168.1.179. The typical connection process would look like the image below. The attacker sends a packet that has the SYN flag set (this is sent to initiate a connection). If the port is open, the server will respond with the SYN / ACK flags set, and the attacker's machine will complete the connection by setting the ACK flag.



Now with the TCP connection fundamentals out of the way, let's get to it.

With a decoy scan, we're hiding our true IP address among a bunch of decoys. I like to call this a chaos scan, because that's what it looks like to an admin on the other end, trying to figure out what's going on (you'll see soon).

Here's a high-level view of what a decoy scan would look like (minus the ACK packet from the hacker).



Here the server is receiving a bunch of packets asking if port 21 is open. Naturally, it will respond to all of them, but only the hacker will receive that message. The rest of the machines are just getting SYN / ACKs that they never initiated in the first place.

For the sake of the example, I'll keep this simple for right now (don't worry, your head will be spinning soon enough).

Using nmap's -D parameter, we will specify a decoy IP to throw into the scan with our legit IP, in this case 192.168.1.1
NOTE: we need to use sudo for this scan because it requires access to raw sockets, which elevation with sudo provides.

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p 21 192.168.1.179 -D 192.168.1.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-13 15:35 EDT
Nmap scan report for 192.168.1.179
Host is up (0.00028s latency).

PORT   STATE  SERVICE
21/tcp closed ftp
MAC Address: E0:2E:0B:D8:E9:3D (Intel Corporate)

Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
```

The port is closed. Let's try again, this time with a port that we know to be open.

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p 111 192.168.1.179 -D 192.168.1.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-13 15:35 EDT
Nmap scan report for 192.168.1.179
Host is up (0.00024s latency).

PORT    STATE SERVICE
111/tcp open  rpcbind
MAC Address: E0:2E:0B:D8:E9:3D (Intel Corporate)

Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
```

Nmap was able to determine the port status. Let's see what happened on the back end.

Let's break down this packet capture.

Source	Destination	Protocol	Length	Info
192.168.1.171	192.168.1.179	TCP	58	45218 → 21 [SYN] Seq=0
192.168.1.1	192.168.1.179	TCP	58	45218 → 21 [SYN] Seq=0
192.168.1.179	192.168.1.171	TCP	60	21 → 45218 [RST, ACK] S
192.168.1.171	192.168.1.179	TCP	58	47440 → 111 [SYN] Seq=0
192.168.1.1	192.168.1.179	TCP	58	47440 → 111 [SYN] Seq=0
192.168.1.179	192.168.1.171	TCP	60	111 → 47440 [SYN, ACK]
192.168.1.171	192.168.1.179	TCP	54	47440 → 111 [RST] Seq=1

The first 2 lines show that my machine sent a SYN to the target, asking if port 21 is open. Also note that the destination doesn't only show my machine (192.168.1.171), but also the decoy we used in the scan.
The third line is the response from the server. Note this is a RST / ACK. The server is resetting the connection, because the port is not open. Also note, we can't see the traffic returned to the decoy (it's not our IP).
Lines 4 and 5 start off the same as the initial 2 lines. On line 6 the server sets the SYN / ACK flags. The port is open and the server is trying to establish a connection.
Line 7 is where an admin can narrow things down a bit (possibly). This is a default scan, which attempts to create a full connection. Therefore, my machine sends the RST to cut things off. None of the decoys would send this, unless they are active on the network. This still hides the scan, but rules out any IPs that were used that are not active in the environment.

Now, let's get even more stealthy. Remember the chaos I mentioned? Here it comes.

Keeping within the same subnet, I'll add a bunch of random IP addresses (comma separated).

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p 111 192.168.1.179 -D 192.168.1.1,192.168.1.73,192.168.1.3,192.168.1.72,192.168.1.71,192.168.1.67,192.168.1.76,192.168.1.77,192.168.1.84,192.168.1.200,192.168.1.93,192.168.1.215,192.168.1.220
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-13 16:38 EDT
Nmap scan report for 192.168.1.179
Host is up (0.00028s latency).

PORT    STATE SERVICE
111/tcp open  rpcbind
MAC Address: E0:2E:0B:D8:E9:3D (Intel Corporate)

Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds
```

Here's the back end.

Source	Destination	Protocol	Length	Info
192.168.1.179	192.168.1.171	TCP	60	111 → 47509 [SYN, A
192.168.1.171	192.168.1.179	TCP	54	47509 → 111 [RST] S
192.168.1.179	192.168.1.72	TCP	60	111 → 47509 [SYN, A
192.168.1.179	192.168.1.71	TCP	60	111 → 47509 [SYN, A
192.168.1.179	192.168.1.67	TCP	60	111 → 47509 [SYN, A
192.168.1.179	192.168.1.76	TCP	60	111 → 47509 [SYN, A
192.168.1.179	192.168.1.77	TCP	60	111 → 47509 [SYN, A
192.168.1.179	192.168.1.84	TCP	60	111 → 47509 [SYN, A
192.168.1.200	192.168.1.179	TCP	58	47509 → 111 [SYN] S
192.168.1.93	192.168.1.179	TCP	58	47509 → 111 [SYN] S
192.168.1.215	192.168.1.179	TCP	58	47509 → 111 [SYN] S
192.168.1.220	192.168.1.179	TCP	58	47509 → 111 [SYN] S
192.168.1.179	192.168.1.93	TCP	60	111 → 47509 [SYN, A
192.168.1.67	192.168.1.179	TCP	60	47509 → 111 [RST] S
192.168.1.73	192.168.1.179	TCP	60	47509 → 111 [RST] S
192.168.1.84	192.168.1.179	TCP	60	47509 → 111 [RST] S
192.168.1.71	192.168.1.179	TCP	60	47509 → 111 [RST] S
192.168.1.72	192.168.1.179	TCP	60	47509 → 111 [RST] S
192.168.1.76	192.168.1.179	TCP	60	47509 → 111 [RST] S
192.168.1.77	192.168.1.179	TCP	60	47509 → 111 [RST] S
192.168.1.93	192.168.1.179	TCP	60	47509 → 111 [RST] S

If I didn't tell you my IP from the start, would you know which machine is mine?

Before I set up the decoys, I ran a ping sweep so that I could use IPs that I know to be active, this hides me within the RST packets and makes my machine indestinguishable from the others that are active.

But wait...there's more.

Nmap also has another option for its decoy scans: RND:(count). What this does is allows nmap to randomly generate however many IPs you specifiy. These won't be in the same subnet, they'll be public, so this is better for an external scan. Here's a quick example. Let's let nmap generate 20 random IPs as decoys.

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p 111 192.168.1.179 -D RND:20
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-13 16:43 EDT
Nmap scan report for 192.168.1.179
Host is up (0.00026s latency).

PORT    STATE SERVICE
111/tcp open  rpcbind
MAC Address: E0:2E:0B:D8:E9:3D (Intel Corporate)

Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds
```

Here we have a number of external IPs probing at port 111. This is a perfect method for wreaking havoc on firewall logs. This highlights the importance of how security professionals really need to understand these concepts so that they're better equipped with how to react and weed out the noise.

Source	Destination	Protocol	Length	Info
192.168.1.171	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
13.56.234.206	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
155.49.181.157	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
79.134.117.67	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
68.103.92.193	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
111.9.178.88	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
32.238.105.174	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
205.51.84.201	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
11.158.69.147	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
199.234.200.18	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
6.211.185.162	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
139.200.238.142	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
116.155.4.155	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
192.168.1.179	192.168.1.171	TCP	60	111 → 35724 [SYN, ACK] S
192.168.1.179	13.56.234.206	TCP	60	111 → 35724 [SYN, ACK] S
192.168.1.179	155.49.181.157	TCP	60	111 → 35724 [SYN, ACK] S
192.168.1.179	79.134.117.67	TCP	60	111 → 35724 [SYN, ACK] S
192.160.117.226	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
24.86.63.160	192.168.1.179	TCP	58	35724 → 111 [SYN] Seq=0
192.168.1.179	68.103.92.193	TCP	60	111 → 35724 [SYN, ACK] S
192.168.1.179	111.9.178.88	TCP	60	111 → 35724 [SYN, ACK] S
192.168.1.179	32.238.105.174	TCP	60	111 → 35724 [SYN, ACK] S
192.168.1.179	205.51.84.201	TCP	60	111 → 35724 [SYN, ACK] S

So there you have it. That's decoy scans in a nutshell.