Nmap Scanning, Fast and Slow

Luc Gommans

October 2025
hack.lu

X41 D-Sec

Why Nmap anyway?

Nmap, Zmap, Masscan… Why choose Nmap?

1. **Data**
   Library of probes, top ports list, OS characteristics lists, scripts, …

2. **Reliability**
   Adapts speed to target’s responses

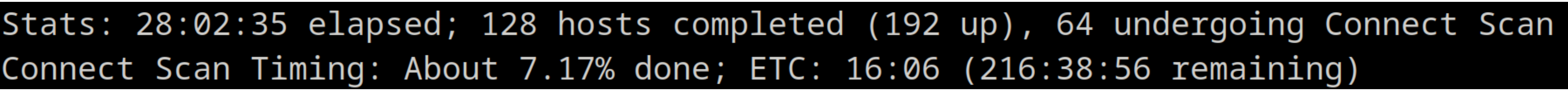
“Why is Nmap so slow”

```
$ sudo nmap -A -p- -Pn -oA site1 192.168.0.0/16
```

- ✅ Safe
- ✅ Detailed
- ✅ Reliable
- ❌ Nobody can tell you how long this will take

“Why is Nmap so slow”

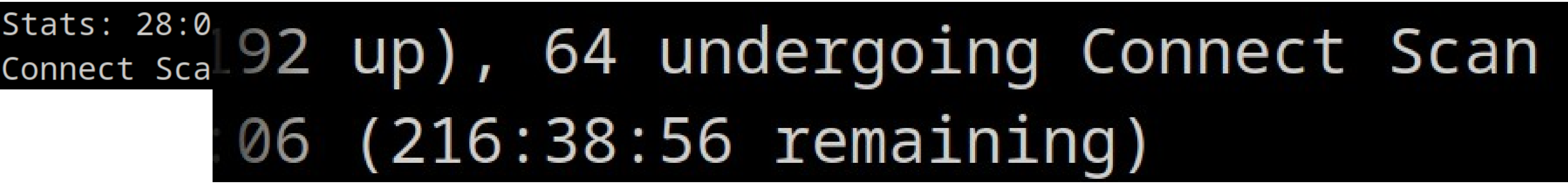
```
$ sudo nmap -A -p- -Pn -oA site1 192.168.0.0/16
```



```
Stats: 28:02:35 elapsed; 128 hosts completed (192 up), 64 undergoing Connect Scan
Connect Scan Timing: About 7.17% done; ETC: 16:06 (216:38:56 remaining)
```

“Why is Nmap so slow”

```
$ sudo nmap -A -p- -Pn -oA site1 192.168.0.0/16
```



```
Stats: 28:0
Connect Sca
```

```
192 up), 64 undergoing Connect Scan
:06 (216:38:56 remaining)
```

216 hours = 9 days
for 64 hosts!

Making Nmap faster

1 year → 1 week

- Bigger hostgroups

- Supply network capacity information

Making Nmap faster

Speed limits:

1. Your host → monitor CPU/RAM

2. Your router → connection states (NAT and/or FW)

(intermediate internet routers should be fine)

3. Their router → same issue

4. Their host → rate limiting

Making Nmap faster

1 week → 1 day

- Scratch some options for the exploratory scan

    “-A” = services probes, script scans, OS detection...

Making Nmap faster

```
$ sudo nmap -v -Pn -oA site1 --min-hostgroup 1024
  --min-rate 1024  --top-ports 200 --max-retries 1
  --randomize-hosts  192.168.0.0/16
```

✅ Safe

🟧 Detailed

✅ Reliable

✅ It will actually finish!

Making Nmap faster

```
$ sudo nmap -v -Pn -oA site1 --min-hostgroup 1024
  --min-rate 1024  --top-ports 200 --max-retries 1
  --randomize-hosts  192.168.0.0/16
```

Duration:

65k hosts × 200 ports × 2 tries = 26M probes

26M probes / 1024 packets/second = 7 hours

Making Nmap faster

```
$ sudo nmap -v -Pn -oA site1 --min-hostgroup 1024
  --min-rate 1024  --top-ports 200 --max-retries 1
  --randomize-hosts  192.168.0.0/16
```

Duration:

65k hosts × 200 ports × 2 tries = 26M probes

26M probes / 1024 packets/second = 7 hours

Remaining ports → cover at leisure during the rest of the week

Takeaways

- Go deep on small groups of hosts+ports
- The ETC is per hostgroup
- Use big hostgroups to spread the load, with -v for live info
- Tell Nmap what your network can handle

- Download: github.com/x41sec/slides (with bonus slides)

Bonus slides!

3 miscellaneous tips

OS detection (-O, included in -A):
Having Nmap running twice seems to make this always fail. In general, it’s slow and fails a lot. I’ve never found this useful. YMMV.

To get hostnames, you usually want --dns-servers <AD> with -sL (list scan). Add -Pn to avoid pinging each host.

Find which ports are in -sU --top-ports 20:

```
sort -rk3 /usr/share/nmap/nmap-services | grep '/udp\s' | head -20
```

Exclude from next scan:

```
nmap --top-ports 100 --exclude-ports 53,161,123,...
```

Command example

```
$ sudo nmap 10.0.0.0/8
   -sS -n -Pn -v -oA blah  --top-ports 50
   --min-hostgroup 1024    --randomize-hosts
   --min-rate 800          --max-retries 1
   --max-rtt-timeout 150ms --scan-delay 500ms
```

Beware: top-ports list is pretty old
Especially for UDP, where scanning is slower,
you’ll want to pick targets specific to the environment

Command example

```
$ sudo nmap 10.0.0.0/8
   -sS -n -Pn -v -oA blah  --top-ports 50
   --min-hostgroup 1024    --randomize-hosts
   --min-rate 800          --max-retries 1
   --max-rtt-timeout 150ms --scan-delay 500ms
```

Scan 1k hosts simultaneously (4× /24).
Specifying --max-hostgroup is rarely useful, but there’s no reason not to

Command example

```
$ sudo nmap 10.0.0.0/8
   -sS -n -Pn -v -oA blah  --top-ports 50
   --min-hostgroup 1024    --randomize-hosts
   --min-rate 800          --max-retries 1
   --max-rtt-timeout 150ms --scan-delay 500ms
```

```
+ Spreads load if they have small-ish networks (/23 or
so)
- Breaks resumption
```

Command example

```
$ sudo nmap 10.0.0.0/8
   -sS -n -Pn -v -oA blah  --top-ports 50
   --min-hostgroup 1024    --randomize-hosts
   --min-rate 800          --max-retries 1
   --max-rtt-timeout 150ms --scan-delay 500ms
```

Each packet has a small % chance of being lost.
No retries will thus work most of the time, but you know you’ll lose some. The odds of two unlikely events is much lower, three even lower, etc. Choose your risk level.

Command example

```
$ sudo nmap 10.0.0.0/8
   -sS -n -Pn -v -oA blah  --top-ports 50
   --min-hostgroup 1024    --randomize-hosts
   --min-rate 800          --max-retries 1
   --max-rtt-timeout 150ms --scan-delay 500ms
```

This is quite high. 50ms (20pps) is more common. Derive it from:

```
  rate / hostgroup  =  800 / 1024  =  0.78 pkts/sec/host
```

and divide by like 2-4 so Nmap has some wiggle room

Scratching the surface

We must go deeper!

- UDP scans
- Service probes and version detection
- Scripts
- IPv6
- A world beyond UDP and TCP