



Download

Reference Guide

Book

Docs

Zenmap GUI

In the Movies

[Nmap Network Scanning](#) / [Chapter 4. Port Scanning Overview](#) / A Quick Port Scanning Tutorial[◀ Prev](#)[Next ▶](#)

A Quick Port Scanning Tutorial

One of my goals in developing Nmap is to keep the most common usage simple, while retaining the flexibility for custom and advanced scans. This is accomplished with the command-line interface by offering dozens of options, but choosing sane defaults when they are not specified. A newbie can start out with a command as simple as **nmap <target>**. Meanwhile, advanced users sometimes specify so many options that their terminal line wraps around.

A similar balance must be struck with command output. The most important results should stick out even to the occasional user who hasn't even read the man page. Yet the output should be comprehensive and concise enough to suit professional penetration testers who run Nmap against thousands of machines daily. Users smart enough to read this book or the Nmap source code benefit from greater control of the scanner and insights into what Nmap output really means.

This tutorial demonstrates some common Nmap port scanning scenarios and explains the output. Rather than attempt to be comprehensive, the goal is simply to acquaint new users well enough to understand the rest of this chapter.

The simplest Nmap command is just **nmap** by itself. This prints a cheat sheet of common Nmap options and syntax. A more interesting command is **nmap <target>**, which does the following:

1. Converts *<target>* from a hostname into an IPv4 address using DNS. If an IP address is specified instead of a hostname this lookup is skipped.
2. Pings the host, by default with an ICMP echo request packet and a TCP ACK packet to port 80, to determine whether it is up and running. If not, Nmap reports that fact and exits. I could have specified `-Pn` to skip this test. See [Chapter 3, Host Discovery \("Ping Scanning"\)](#).
3. Converts the target IP address back to the name using a reverse-DNS query. Because of the way DNS works, the reverse name may not be the same as the *<target>* specified on the command-line. This query can be skipped with the `-n` option to improve speed and stealthiness.
4. Launches a TCP port scan of the most popular 1,000 ports listed in `nmap-services`. A SYN stealth scan is usually used, but connect scan is substituted instead for non-root Unix users who lack the privileges necessary to send raw packets.
5. Prints the results to standard output in normal human-readable format, and exits. Other output formats and locations (files) can be specified, as described in [Chapter 13, Nmap Output Formats](#). [Example 4.2](#) displays the results when `scanme.nmap.org` is used as *<target>*.

Example 4.2. Simple scan: `nmap scanme.nmap.org`

```
# nmap scanme.nmap.org

Starting Nmap ( https://nmap.org )
Nmap scan report for scanme.nmap.org (64.13.134.52)
```

```

Not shown: 994 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    closed smtp
53/tcp    open  domain
70/tcp    closed gopher
80/tcp    open  http
113/tcp   closed auth

Nmap done: 1 IP address (1 host up) scanned in 4.99 seconds

```

The first output line in [Example 4.2](#) simply gives the URL for downloading Nmap. The time Nmap started and version number are normally provided as well, though these were generally removed from this book for consistency and to avoid line wrapping.

The next line provides the target IP address (IPv4 in this case), and reverse DNS name (also known as the PTR record) if it is available. Nmap promises to show the “interesting ports”, though all ports scanned are accounted for. The ports considered most interesting because they are open or in a rarely-seen state for that host are itemized individually. When many ports are in a single non-open state, they are considered a default state, and aggregated onto a single line to avoid diluting the results with thousands of uninteresting entries. In this case, Nmap notes that 994 ports are filtered.

The interesting ports table comes next, and provides the key scan results. The columns vary depending on options used, but in this case provide the port number and protocol, state, and service protocol for each port. The service here is just a guess made by looking up the port in `nmap-services`. The service would be listed as unknown if any of the ports had no name registered in that file. Three of these ports are open and three are closed.

Finally, Nmap reports some basic timing stats before it exits. These stats are the number of targets specified, the number of those that the ping scan found to be up, and the total time taken.

While this simple command is often all that is needed, advanced users often go much further. In [Example 4.3](#), the scan is modified with four options. `-p0` asks Nmap to scan every possible TCP port, `-v` asks Nmap to be verbose about it, `-A` enables aggressive tests such as remote OS detection, service/version detection, and the Nmap Scripting Engine (NSE). Finally, `-T4` enables a more aggressive timing policy to speed up the scan.

Example 4.3. More complex: `nmap -p0- -v -A -T4 scanme.nmap.org`

```

# nmap -p0- -v -A -T4 scanme.nmap.org

Starting Nmap ( https://nmap.org )
Completed Ping Scan at 00:03, 0.01s elapsed (1 total hosts)
Scanning scanme.nmap.org (64.13.134.52) [65536 ports]
Discovered open port 22/tcp on 64.13.134.52
Discovered open port 53/tcp on 64.13.134.52
Discovered open port 80/tcp on 64.13.134.52
SYN Stealth Scan Timing: About 6.20% done; ETC: 00:11 (0:07:33 remaining)
Completed SYN Stealth Scan at 00:10, 463.55s elapsed (65536 total ports)
Completed Service scan at 00:10, 6.03s elapsed (3 services on 1 host)
Initiating OS detection (try #1) against scanme.nmap.org (64.13.134.52)
Initiating Traceroute at 00:10
64.13.134.52: guessing hop distance at 9
Completed SCRIPT ENGINE at 00:10, 4.04s elapsed
Host scanme.nmap.org (64.13.134.52) appears to be up ... good.
Nmap scan report for scanme.nmap.org (64.13.134.52)
Not shown: 65530 filtered ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 4.3 (protocol 2.0)
25/tcp    closed smtp
53/tcp    open  domain   ISC BIND 9.3.4
70/tcp    closed gopher
80/tcp    open  http     Apache httpd 2.2.2 ((Fedora))

```

```
|_HTML title: Go ahead and ScanMe!  
113/tcp closed auth  
Device type: general purpose  
Running: Linux 2.6.X  
OS details: Linux 2.6.20-1 (Fedora Core 5)  
Uptime guess: 2.457 days (since Thu Sep 18 13:13:24 2008)  
TCP Sequence Prediction: Difficulty=204 (Good luck!)  
IP ID Sequence Generation: All zeros  
  
TRACEROUTE (using port 80/tcp)  
HOP RTT ADDRESS  
[First eight hops cut for brevity]  
9 10.36 metro0.sv.svcolo.com (208.185.168.173)  
10 10.29 scanme.nmap.org (64.13.134.52)  
  
Nmap done: 1 IP address (1 host up) scanned in 477.23 seconds  
Raw packets sent: 131432 (5.783MB) | Rcvd: 359 (14.964KB)
```

Nmap certainly provided the requested verbosity in [Example 4.3](#)! Fortunately the extra output is easy to understand. The first 13 new lines are runtime information letting the user know what is happening as she stares expectantly at the terminal, hoping for good news. What constitutes good news depends on whether she is a systems administrator who has to fix problems, a pen-tester who needs some issues to report on, or a black-hat cracker trying to exploit them. About a dozen similar lines were removed for brevity. The “discovered open port” lines provide as-it-happens notification of open ports so that she can start banging on them before the scan even finishes. The “scan timing” line provides a completion time estimate, so she knows whether to keep staring at the screen or have lunch. Since network conditions (latency, congestion, bandwidth, etc.) and packet filtering rules vary so much, the same scan options may take 30 seconds to complete against one host and 45 minutes against another. If you want the current time estimate while scanning, just press enter.

The port table shows no new ports. All the extra ports scanned are in the filtered state, raising the filtered port total from 994 to 65,530. While there are no new itemized ports, the entries have changed. A new VERSION column provides the application name and version details for the listening service. This comes from service detection, one of the features enabled by the -A option. Another feature of service detection is that all of the service protocols in the SERVICE column have actually been verified. In the previous scan, they were based on the relatively flimsy heuristic of an nmap-services port number lookup. That table lookup happened to be correct this time, but it won't always be.

Another feature added by -A is the Nmap Scripting Engine, which is discussed in depth in [Chapter 9, Nmap Scripting Engine](#). The only script shown here is HTML title. Dozens of other scripts exist, but none found useful output for this machine. The traceroute results were also added by -A. This option is more efficient and more powerful than most traceroute programs since probes are performed in parallel and Nmap uses scan results to determine a favorable probe type (TCP packets to port 80 in this case).

Most of the remaining new lines come from OS detection (also enabled by -A), which is discussed in depth in [Chapter 8, Remote OS Detection](#). The final line shows that all this extra info came at a price—the scan took almost 100 times longer than [Example 4.2, “Simple scan: nmap scanme.nmap.org”](#) to complete (477 seconds compared to 5).

Site Search



Nmap Security Scanner

Ref Guide

Npcap packet capture

User's Guide

API docs

Security Lists

Nmap Announce

Nmap Dev

Security Tools

Vuln scanners

Password audit

Install Guide

Docs

Download

Nmap OEM

About

About/Contact


Privacy


Advertising


Nmap Public Source License


Download

Npcap OEM









Full Disclosure

Open Source Security

BreachExchange

Web scanners

Wireless

Exploitation

https://nmap.org/book/port-scanning-tutorial.html

4/4