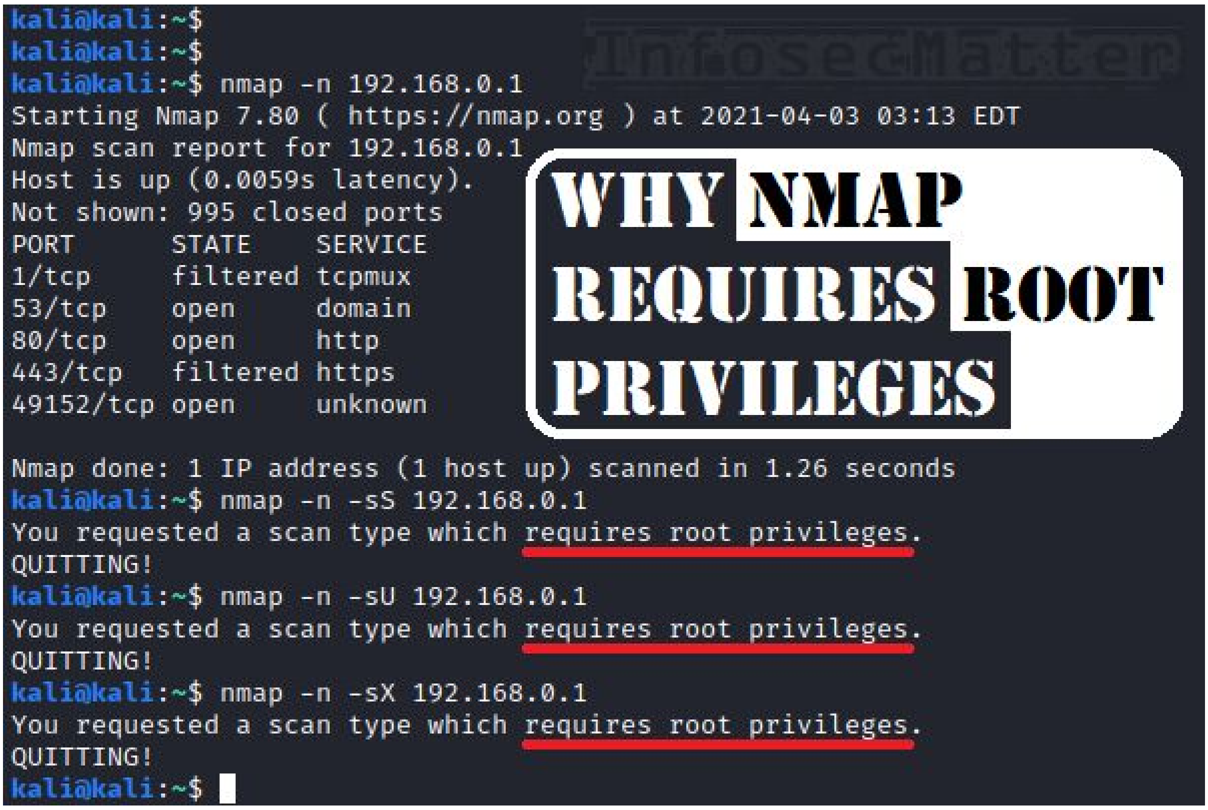
Why Does Nmap Need Root Privileges?

infosecmatter.com/why-does-nmap-need-root-privileges

April 3, 2021



```
kali@kali:~$
kali@kali:~$
kali@kali:~$ nmap -n 192.168.0.1
Starting Nmap 7.80 ( https://nmap.org ) at 2021-04-03 03:13 EDT
Nmap scan report for 192.168.0.1
Host is up (0.0059s latency).
Not shown: 995 closed ports
PORT      STATE    SERVICE
1/tcp     filtered tcpmux
53/tcp    open     domain
80/tcp    open     http
443/tcp   filtered https
49152/tcp open     unknown

Nmap done: 1 IP address (1 host up) scanned in 1.26 seconds
kali@kali:~$ nmap -n -sS 192.168.0.1
You requested a scan type which requires root privileges.
QUITTING!
kali@kali:~$ nmap -n -sU 192.168.0.1
You requested a scan type which requires root privileges.
QUITTING!
kali@kali:~$ nmap -n -sX 192.168.0.1
You requested a scan type which requires root privileges.
QUITTING!
kali@kali:~$
```

Nmap is probably the most popular network port scanner used by cyber security professionals and penetration testers today. In this blog post we are going to discuss in detail reasons why Nmap requires root (administrative) privileges for most of its functions.

Here’s the short explanation of why Nmap needs root privileges:

Although Nmap can perform basic functions with normal user privileges, in order to use its advanced scanning features, Nmap requires root privileges. This is because it needs access to raw network sockets, ability to inject network packets on the network while listening on the network interface (aka. packet sniffing) to receive the responses.

For more detailed answer, read on.

Crafting raw network packets

In order to use advanced port scanning methods such as the stealthy SYN scan, ACK, Window, NULL, Xmas or similar scan types, Nmap needs to be able to construct raw (custom) network packets and inject them directly on the network.

In other words, Nmap needs access to raw sockets and this is something non-standard, something which is out of the standard functionality of an operating system for a typical user.

Most operating systems do not have such functionalities built-in and that’s why it is necessary to install additional library on the system (WinPcap or Npcap on Windows and Libpcap on Linux) in order to use Nmap.

Additionally, in most cases these advanced Nmap scan methods do not establish a network connection at all, e.g. there is no typical TCP session. Therefore, Nmap needs to be able to listen on the network interface (aka. packet sniffing) in order to receive the replies.

This brings us to the 2nd most important reason.

Listening on network interface (sniffing)

As hinted in the previous paragraph, Nmap has to be able to listen on the network interface in order to receive the incoming packets. This is used practically in all advanced network scans, OS fingerprinting methods, some of the host discovery techniques and others.

For instance, here’s what Nmap tells us if we try to run IP Protocol ping (host discovery method) as a normal user:

```
kali@kali:~$ nmap -n -PO 192.168.0.1
Sorry, IPProto Ping (-PO) only works if you are root (because we need to read raw
responses off the wire)
QUITTING!
```

Without being able to listen (sniff) on the network interface, Nmap simply cannot capture the packets that it needs to capture in order to discern alive systems on the network.

Performing UDP port scanning

This may sound surprising, but Nmap requires root privileges in order to do any UDP port scanning.

Here’s what we see if we attempt to do UDP port scan as a normal user:

```
kali@kali:~$ nmap -n -sU 192.168.0.1
You requested a scan type which requires root privileges.
QUITTING!
```

This is because UDP is a stateless protocol and for a proper diagnostics, Nmap needs to listen on the network interface (again, packet sniffing) to find out what’s actually happening.

It also needs to be able to receive ICMP messages in order to really diagnose whether an UDP port is open (either by receiving nothing or by receiving some response coming to the source port) or if it is closed (ideally by receiving ICMP port unreachable message).

Binding to privileged network ports

In certain cases, Nmap may need to bind on a local network interface and start listening for incoming connections on TCP or UDP port which is below 1024.

Binding on ports below 1024 is a privileged operation which has been historically restricted only for administrative users for security reasons. Although this is based on an obsolete security model, it is still enforced today on all popular operating systems. Therefore, a normal user cannot assign these ports without root access.

Running certain NSE scripts

There are some Nmap Scripting Engine (NSE) scripts that require root privileges for their operation, although most of them do not need any special privileges and can run perfectly fine as a normal user.

In fact, from 604 NSE scripts that are included in the current Nmap release (link), only 24 of them require root privileges.

Here's a list of all NSE scripts requiring root privileges:

All other NSE scripts (574) run perfectly fine as a normal user (without root).

Note that NSE scripts are located in the `/usr/share/nmap/scripts/` directory on most Linux systems (e.g. Kali Linux).

Scan methods requiring root privileges

Here's an overview of all port scanning methods that Nmap supports and whether they require root privileges or not:

Option	Port scanning method	Requires root
-sT	TCP Connect scan	no
-sS	TCP SYN / stealth / half-open scan	yes
-sA	TCP ACK scan	yes
-sW	TCP Window scan	yes
-sM	TCP Maimon scan (FIN/ACK flags)	yes
-sN	TCP Null scan (no flags)	yes
-sF	TCP FIN scan	yes
-sX	TCP Xmass scan (all flags)	yes
-sI	TCP Zombie / Idle scan	yes
-sY	SCTP INIT scan	yes
-sZ	SCTP COOKIE ECHO scan	yes
-sU	UDP scan	yes
-sO	IP protocol scan	yes

Here's an overview of all host discovery methods and whether they require root privileges or not:

Option	Host discovery method	Requires root
-PR	ARP Ping	no
-PE	ICMP Echo (ping)	yes
-PP	ICMP Timestamp	yes
-PM	ICMP Netmask request	yes
-PO	IP Protocol ping	yes

As you can see, the choices we have as a normal user are quite limited – we can use only 1 port scanning method and 1 host discovery method. Although this may seem very limiting, we should still be able get the job done in the end of the day without any problem.

In fact, in certain attack simulations and penetration testing scenarios, port scanning using standard operating system functionalities (aka. TCP Connect scan, which establishes a full TCP connection) is perfectly fine.

For example, see my network port scanner written in PowerShell described here.

Running Nmap without root privileges

It is worth to mention that we can make Nmap run without root privileges and in the same time support all the advanced features and port scanning methods.

All we need to do is to use the Linux process capabilities and assign these 3 capabilities to the Nmap binary:

- CAP_NET_RAW
- CAP_NET_ADMIN
- CAP_NET_BIND_SERVICE

Here’s how to do it:

```
sudo setcap cap_net_raw,cap_net_admin,cap_net_bind_service+eip /usr/bin/nmap
```

From now on, we can run Nmap as a normal unprivileged user like this:

```
nmap --privileged -sS 192.168.0.1
```

Note that we have to inform Nmap via the `--privileged` flag that it has all the necessary capabilities even though we are not root.

More detailed information about running Nmap without root can be found here.

SHARE THIS

TAGS | ICMP | Linux | Linux capabilities | Nmap | Packet sniffing | Portscan | Raw sockets | TCP | UDP