

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра САПР

МЕТОДИЧЕСКОЕ ПОСОБИЕ
«Сканирование сети при помощи Nmap»

Студенты гр. 9302

Ширнин К.В.

Квитко Д.В.

Преподаватель

Горячев А.В.

Санкт-Петербург

2022

Оглавление

1.	Установка Nmap	3
1.1	Установка Nmap в Ubuntu и Debian	3
1.2	Установка Nmap на CentOS и Fedora	3
1.3	Установка Nmap на macOS	3
1.4	Установка Nmap в Windows	4
2.	Синтаксис Nmap	5
3.	Справка по Nmap	7
4.	Как пользоваться Nmap для сканирования портов в linux	9
4.1	Базовые возможности	13
4.2	Определяем название и версию сервиса на порте	15
4.3	Определяем имя и версию ОС	16
4.4	Повышаем скорость сканирования	16
4.5	Скрываем следы	17
4.6	Обходим IDS и брандмауэры	19
4.7	Используем Nmap для обнаружения машин в сети	22
5.	Заключение	25

1. Установка Nmap

Nmap — это многоплатформенная программа, которую можно установить во всех основных операционных системах. Первоначально он был выпущен как инструмент только для Linux, а позже был перенесен на другие системы, такие как BSD, Windows и macOS.

Если вы предпочитаете графический интерфейс, а не командную строку, Nmap также имеет графический пользовательский интерфейс под названием Zenmap.

Официальные бинарные пакеты доступны для загрузки со страницы загрузки Nmap. Процедура установки проста и зависит от вашей операционной системы.

1.1 Установка Nmap в Ubuntu и Debian

Nmap доступен из репозитория Ubuntu и Debian по умолчанию. Чтобы установить его, запустите:

```
sudo apt update  
sudo apt install nmap
```

1.2 Установка Nmap на CentOS и Fedora

На CentOS и других производных от Red Hat запускаются:

```
sudo dnf install nmap
```

1.3 Установка Nmap на macOS

Пользователи macOS могут установить Nmap, загрузив установочный пакет «.dmg» с сайта Nmap или через Homebrew:

```
brew install nmap
```

1.4 Установка Nmap в Windows

Версия Nmap для Windows имеет некоторые ограничения и, как правило, немного медленнее, чем версия для UNIX.

Самый простой вариант установки Nmap в Windows — это загрузить и запустить самоустанавливающийся exe-файл.

Вы можете запустить Nmap в Windows либо из командной строки, либо запустив программу Zenmap. Для получения дополнительной информации о том, как использовать Nmap в Windows, ознакомьтесь с инструкциями по использованию после установки.

2. Синтаксис Nmap

Команда запуска Nmap очень проста для этого достаточно передать ей в параметрах целевой IP адрес или сеть, а также указать опции при необходимости:

```
$ nmap опции адрес
```

Рассмотрим основные опции.

- **-sL** — просто создать список работающих хостов, но не сканировать порты nmap;
- **-sP** — только проверять доступен ли хост с помощью ping;
- **-PN** — считать все хосты доступными, даже если они не отвечают на ping;
- **-sS/sT/sA/sW/sM** — TCP сканирование;
- **-sU** — UDP сканирование nmap;
- **-sN/sF/sX** — TCP NULL и FIN сканирование;
- **-sC** — запускать скрипт по умолчанию;
- **-sI** — ленивое Idle сканирование;
- **-p** — указать диапазон портов для проверки;
- **-sV** — детальное исследование портов для определения версий служб;
- **-O** — определять операционную систему;

- **-T[0-5]** — скорость сканирования, чем больше, тем быстрее;
- **-D** — маскировать сканирование с помощью фиктивных IP;
- **-S** — изменить свой IP адрес на указанный;
- **-e** — использовать определенный интерфейс;
- **—spoof-mac** — установить свой MAC адрес;
- **-A** — определение операционной системы с помощью скриптов.

Теперь, когда мы рассмотрели все основные опции, давайте поговорим о том, как выполняется сканирование портов `ntar`.

3. Справка по Nmap

nmap [Тип(ы) сканирования] [Опции] {определение цели}

ОПРЕДЕЛЕНИЕ ЦЕЛИ СКАНИРОВАНИЯ:

Можно использовать сетевые имена, IP адреса, сети и т.д.

Пример: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254

-iL <имя_входного_файла>: Использовать список хостов/сетей из файла

-iR <количество_хостов>: Выбрать произвольные цели

--exclude <хост1[,хост2][,хост3],...>: Исключить хосты/сети

--excludefile <имя_файла>: Исключить список из файла

ОБНАРУЖЕНИЕ ХОСТОВ:

-sL: Сканирование с целью составления списка - просто составить список целей для сканирования

-sn: Пинг сканирование - отключить сканирование портов

-Pn: Рассматривать все хосты будто бы они онлайн -- пропустить обнаружение хостов

-PS/PA/PU/PY[список_портов]: TCP SYN/ACK, UDP или SCTP обнаружение данных портов пингование заданных портов

-PE/PP/PM: Пингование с использованием ICMP эхо запросов, запросов временной метки и сетевой маски

-PO[список_протоколов]: Пингование с использованием IP протокола

-n/-R: Никогда не производить DNS разрешение/Всегда производить разрешение [по умолчанию: иногда]

--dns-servers <сервер1[,сервер2],...>: Задать собственные DNS сервера

--system-dns: Использовать системный DNS преобразователь

--traceroute: Отслеживать путь к хосту

РАЗЛИЧНЫЕ ПРИЕМЫ СКАНИРОВАНИЯ:

-sS/sT/sA/sW/sM: TCP SYN/с использованием системного вызова Connect()/ACK/window/Maimon сканирования

-sU: UDP сканирование

-sN/sF/sX: TCP Null, FIN и Xmas сканирования

--scanflags <флаги>: Задать собственные TCP флаги

-sI <зомби_хост[:порт]>: "ленивое" (Idle) сканирование

-sY/sZ: SCTP INIT/COOKIE-ECHO сканирования

-sO: Сканирование IP протокола

-b <FTP ретранслирующий хост>: FTP bounce сканирование

ОПРЕДЕЛЕНИЕ ПОРТОВ И ПОРЯДКА СКАНИРОВАНИЯ:

-p <диапазон_портов>: Сканирование только определенных портов

Пример: -p22; -p1-65535; -p u:53,111,137,t:21-25,80,139,8080,s:9

--exclude-ports <port ranges>: Exclude the specified ports from scanning

-F: Быстрый режим - Сканировать меньше портов чем при сканировании по умолчанию

-r: Сканировать порты последовательно - не использовать случайный порядок портов

--top-ports <количество_портов>: Сканировать <количество_портов> наиболее распространенных портов

--port-ratio <рейтинг>: Сканировать порты с рейтингом большим чем <рейтинг>

ОПРЕДЕЛЕНИЕ СЛУЖБ И ИХ ВЕРСИЙ:

-sV: Исследовать открытые порты для определения информации о службе/версии

--version-intensity <уровень>: Устанавливать от 0 (легкое) до 9 (пробовать все запросы)

--version-light: Ограничиться наиболее легкими запросами (интенсивность 2)

--version-all: Использовать каждый единичный запрос (интенсивность 9)

--version-trace: Выводить подробную информацию о процессе сканирования (для отладки)

СКАНИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ СКРИПТОВ:

- SC:** эквивалентно опции **--script=default**
- script=<Lua скрипты>:** <Lua скрипты> это разделенный запятыми список директорий, файлов скриптов или категорий скриптов
- script-args=<имя1=значение1,[имя2=значение2,...]>:** передача аргументов скриптам
- script-args-file=имя_файла:** передать скрипту NSE аргументы в файле
- script-trace:** Выводить все полученные и отправленные данные
- script-updatedb:** Обновить базу данных скриптов.
- script-help=<Lua скрипты>:** Показать помощь о скриптах.
<Lua скрипты> разделённый запятой список файлов скриптов или категорий скриптов.

ОПРЕДЕЛЕНИЕ ОС:

- O:** Активировать функцию определения ОС
- osscan-limit:** Использовать функцию определения ОС только для "перспективных" хостов
- osscan-guess:** Угадать результаты определения ОС

ОПЦИИ УПРАВЛЕНИЯ ВРЕМЕНЕМ И ПРОИЗВОДИТЕЛЬНОСТЬЮ:

Опции, принимающие аргумент <время>, задаются в миллисекундах, пока вы не добавите 's' (секунды), 'm' (минуты), или 'h' (часы) к значению (напр. 30m).

- T[0-5]:** Установить шаблон настроек управления временем (больше - быстрее)
- min-hostgroup/max-hostgroup <кол_хостов>:** Установить размер групп для параллельного сканирования
- min-parallelism/max-parallelism <кол_хостов>:** Регулирует распараллеливание запросов
- min-rtt-timeout/max-rtt-timeout/initial-rtt-timeout <время>:** Регулирует время ожидания ответа на запрос
- max-retries <количество_попыток>:** Задаёт максимальное количество повторных передач запроса
- host-timeout <время>:** Прекращает сканирование медленных целей
- scan-delay/--max-scan-delay <время>:** Регулирует задержку между запросами
- min-rate <число>:** Посылать запросы с интенсивностью не меньше чем <число> в секунду
- max-rate <число>:** Посылать запросы с интенсивностью не больше чем <число> в секунду

ОБХОД ФАЙЕРВОЛОВ/IDS И СПУФИНГ:

- f; --mtu <значение>:** фрагментировать пакеты (опционально с заданным значением MTU)
- D <фикт_хост1,фикт_хост2[,ME],...>:** Маскировка сканирования с помощью фиктивных хостов
- S <IP_адрес>:** Изменить исходный адрес
- e <интерфейс>:** Использовать конкретный интерфейс
- g/--source-port <номер_порта>:** Использовать заданный номер порта
- proxies <url1,[url2],...>:** Переправлять подключения через прокси HTTP/SOCKS4
- data <hex string>:** Append a custom payload to sent packets
- data-string <string>:** Дописать пользовательскую ASCII строку к отправляемым пакетам
- data-length <число>:** Добавить произвольные данные к посылаемым пакетам
- ip-options <опции>:** Посылать пакет с заданным ip опциями
- ttl <значение>:** Установить IP поле time-to-live (время жизни)

--spoof-mac <MAC_адрес/префикс/название производителя>: Задать собственный MAC адрес

--badsum: Посылать пакеты с фиктивными TCP/UDP контрольными суммами

ВЫВОД РЕЗУЛЬТАТОВ:

-oN/-oX/-oS/-oG <файл>: Выводить результаты нормального, XML, s|<rIpt kIddi3, и GrePable формата вывода, соответственно, в заданный файл

-oA <базовое_имя_файла>: Использовать сразу три основных формата вывода

-v: Увеличить уровень вербальности (используйте -vv или более для усиления эффекта)

-d: Увеличить уровень отладки (используйте -dd или более для усиления эффекта)

--reason: Отобразить причину, по которой порт в конкретном состоянии

--open: Показывать только открытые (или возможно открытые) порты

--packet-trace: Отслеживание принятых и переданных пакетов

--iflist: Вывести список интерфейсов и роутеров (для отладки)

--append-output: Добавлять в конец, а не перезаписывать выходные файлы

--resume <имя_файла>: Продолжить прерванное сканирование

--stylesheet <путь/URL>: Устанавливает XSL таблицу стилей для преобразования XML вывода в HTML

--webxml: Загружает таблицу стилей с Nmap.Org

--no-stylesheet: Убрать объявление XSL таблицы стилей из XML

РАЗЛИЧНЫЕ ОПЦИИ:

-6: Включить IPv6 сканирование

-A: Активировать функции определения ОС и версии, сканирование с использованием скриптов и трассировку

--datadir <имя_директории>: Определяет место расположения файлов Nmap

--send-eth/--send-ip: использовать сырой уровень ethernet/IP

--privileged: Предполагать, что у пользователя есть все привилегии

--unprivileged: Предполагать, что у пользователя нет привилегий для использования сырых сокетов

-V: Вывести номер версии

-h: Вывести эту страницу помощи.

ПРИМЕРЫ:

```
nmap -v -A scanme.nmap.org
nmap -v -sn 192.168.0.0/16 10.0.0.0/8
nmap -v -iR 10000 -Pn -p 80
```

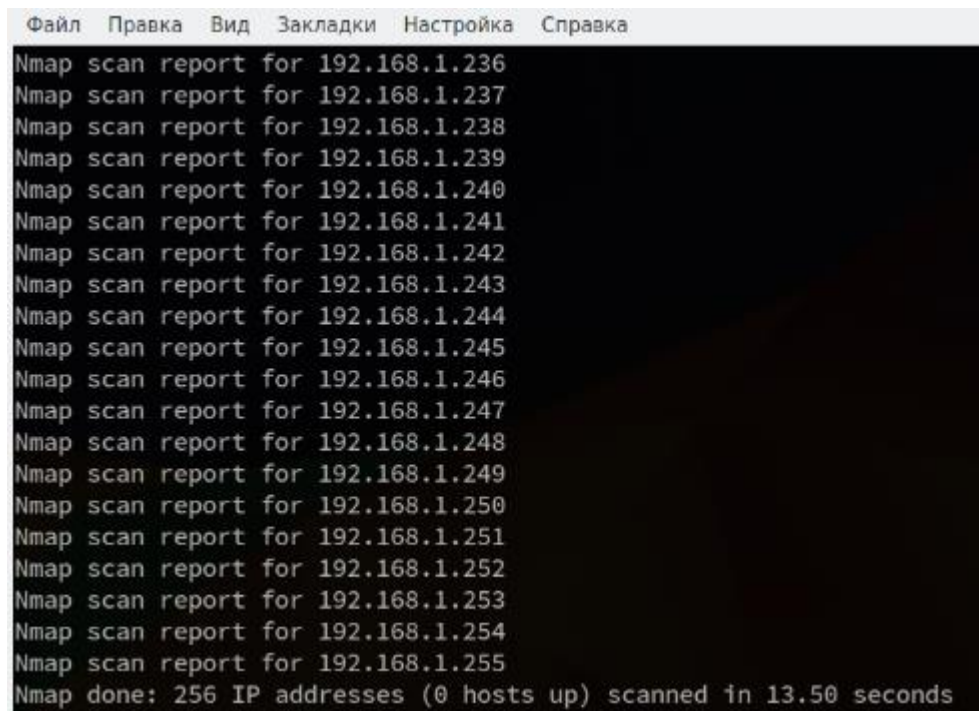
4. Как пользоваться Nmap для сканирования портов в linux

Дальше рассмотрим примеры nmap. Сначала давайте рассмотрим как найти все подключенные к сети устройства, для этого достаточно использовать опцию -sL и указать маску нашей сети. В примере это 192.168.1.1/24. Маску вашей локальной сети вы можете найти, выполнив команду:

```
ip addr show
```

Команда на сканирование сети nmap будет выглядеть вот так:

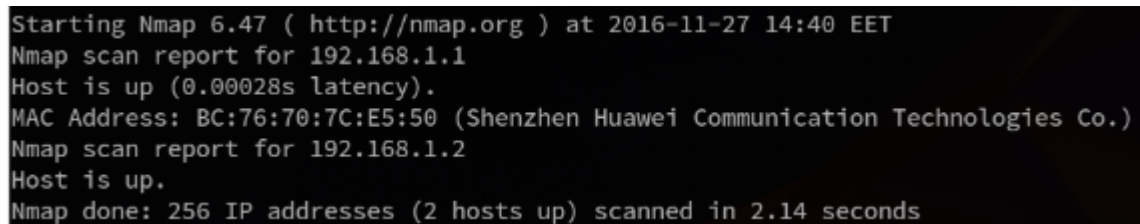
```
nmap -sL 192.168.1.1/24
```



```
Файл  Правка  Вид  Закладки  Настройка  Справка
Nmap scan report for 192.168.1.236
Nmap scan report for 192.168.1.237
Nmap scan report for 192.168.1.238
Nmap scan report for 192.168.1.239
Nmap scan report for 192.168.1.240
Nmap scan report for 192.168.1.241
Nmap scan report for 192.168.1.242
Nmap scan report for 192.168.1.243
Nmap scan report for 192.168.1.244
Nmap scan report for 192.168.1.245
Nmap scan report for 192.168.1.246
Nmap scan report for 192.168.1.247
Nmap scan report for 192.168.1.248
Nmap scan report for 192.168.1.249
Nmap scan report for 192.168.1.250
Nmap scan report for 192.168.1.251
Nmap scan report for 192.168.1.252
Nmap scan report for 192.168.1.253
Nmap scan report for 192.168.1.254
Nmap scan report for 192.168.1.255
Nmap done: 256 IP addresses (0 hosts up) scanned in 13.50 seconds
```

Иногда это сканирование может не дать никаких результатов, потому что некоторые операционные системы имеют защиту от сканирования портов. Но это можно обойти, просто использовав для сканирования `ping` всех `ip` адресов сети, для этого есть опция `-sn`:

```
nmap -sn 192.168.1.1/24
```



```
Starting Nmap 6.47 ( http://nmap.org ) at 2016-11-27 14:40 EET
Nmap scan report for 192.168.1.1
Host is up (0.00028s latency).
MAC Address: BC:76:70:7C:E5:50 (Shenzhen Huawei Communication Technologies Co.)
Nmap scan report for 192.168.1.2
Host is up.
Nmap done: 256 IP addresses (2 hosts up) scanned in 2.14 seconds
```

Как видите, теперь программа обнаружила активные устройства в сети. Дальше мы можем сканировать порты `nmap` для нужного узла запустив утилиту без опций:

```
sudo nmap 192.168.1.1
```

```
Starting Nmap 6.47 ( http://nmap.org ) at 2016-11-27 14:44 EET
Nmap scan report for 192.168.1.1
Host is up (0.00059s latency).
Not shown: 996 filtered ports
PORT      STATE SERVICE
21/tcp    open  ftp
23/tcp    open  telnet
25/tcp    open  smtp
80/tcp    open  http
MAC Address: BC:76:70:7C:E5:50 (Shenzhen Huawei Communication Technologies Co.)
Nmap done: 1 IP address (1 host up) scanned in 691.35 seconds
```

Теперь мы можем видеть, что у нас открыто несколько портов, все они используются каким-либо сервисом на целевой машине. Каждый из них может быть потенциально уязвимым, поэтому иметь много открытых портов на машине небезопасно.

Чтобы узнать более подробную информацию о машине и запущенных на ней сервисах вы можете использовать опцию `-sV`. Утилита подключится к каждому порту и определит всю доступную информацию:

```
sudo nmap -sV 192.168.1.1
```

```
Starting Nmap 6.47 ( http://nmap.org ) at 2016-11-27 14:59 EET
Nmap scan report for 192.168.1.1
Host is up (0.00050s latency).
Not shown: 996 filtered ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp?
23/tcp    open  telnet?
25/tcp    open  smtp?
80/tcp    open  http    RomPager UPnP/1.0
```

На нашей машине запущен `ftp`, а поэтому мы можем попытаться рассмотреть эту службу подробнее с помощью стандартных скриптов `nmap`. Скрипты позволяют проверить порт более детально, найти возможные уязвимости. Для этого используйте опцию `-sC` и `-p` чтобы задать порт:

```
sudo nmap -sC 192.168.56.102 -p 21
```

Мы выполняли скрипт по умолчанию, но есть еще и другие скрипты, например, найти все скрипты для `ftp` вы можете командой:

```
sudo find /usr/share/nmap/scripts/ -name '*.nse' | grep ftp
```

Затем попытаемся использовать один из них, для этого достаточно указать его с помощью опции `—script`. Но сначала вы можете посмотреть информацию о скрипте:

```
sudo nmap --script-help ftp-brute.nse
```

```
Starting Nmap 6.47 ( http://nmap.org ) at 2016-11-27 15:10 EET

ftp-brute
Categories: intrusive brute
http://nmap.org/nsedoc/scripts/ftp-brute.html
  Performs brute force password auditing against FTP servers.

  Based on old ftp-brute.nse script by Diman Todorov, Vlatko Kosturjak and Ron Bowes.
```

Этот скрипт будет пытаться определить логин и пароль от FTP на удаленном узле. Затем выполните скрипт:

```
sudo nmap --script ftp-brute.nse 192.168.1.1 -p 21
```

```
Starting Nmap 6.47 ( http://nmap.org ) at 2016-11-27 15:11 EET

sergiy@dhcppc0:~> nmap --script ftp-brute.nse 192.168.1.1 -p 21

Starting Nmap 6.47 ( http://nmap.org ) at 2016-11-27 15:11 EET
Nmap scan report for 192.168.1.1
Host is up (0.00057s latency).
PORT      STATE SERVICE
21/tcp    open  ftp
| ftp-brute:
|   Accounts
|     admin:admin - Valid credentials
|   Statistics
|     Performed 370 guesses in 75 seconds, average tps: 6
|
|_ ERROR: Too many retries, aborted ...

Nmap done: 1 IP address (1 host up) scanned in 75.88 seconds
```

В результате скрипт подобрал логин и пароль, `admin/admin`. Вот поэтому не нужно использовать параметры входа по умолчанию.

Также можно запустить утилиту с опцией -A, она активирует более агрессивный режим работы утилиты, с помощью которого вы получите большую часть информации одной командой:

```
sudo nmap -A 192.168.1.1
```

```
Starting Nmap 6.47 ( http://nmap.org ) at 2016-11-27 15:17 EET
Nmap scan report for 192.168.1.1
Host is up (0.00066s latency).
Not shown: 996 filtered ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp?
|_ftp-anon: ERROR: Script execution failed (use -d to debug)
|_ftp-bounce: no banner
23/tcp    open  telnet?
25/tcp    open  tcpwrapped
|_smtp-command: Couldn't establish connection on port 25
80/tcp    open  tcpwrapped
MAC Address: BC:76:70:7C:E5:50 (Shenzhen Huawei Communication Technologies Co.)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: switch|phone|VoIP adapter
Running (JUST GUESSING): Cisco embedded (86%), Nokia Symbian OS (85%)
OS CPE: cpe:/h:cisco:catalyst_1900 cpe:/o:nokia:symbian_os cpe:/h:cisco:ata_188_voip_gateway
Aggressive OS guesses: Cisco Catalyst 1900 switch (86%), Nokia 3600i mobile phone (85%), Cisco ATA 188 VoIP adapter (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop

TRACEROUTE
HOP RTT      ADDRESS
1   0.66 ms  192.168.1.1

OS and Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 48.31 seconds
```

Обратите внимание, что здесь есть почти вся информация, которую мы уже видели раньше. Ее можно использовать чтобы увеличить защиту этой машины.

4.1 Базовые возможности

Перед тем как перейти к обсуждению продвинутых возможностей Nmap, давай уделим немного внимания использованию Nmap на базовом уровне. Конечно, можно сказать, что говорить тут не о чем, — достаточно всего лишь

натравить Nmap на нужный хост, и спустя некоторое время он выдаст на экран информацию об открытых портах:

```
nmap 192.168.0.1
```

Это действительно так, но стоит иметь в виду две особенности реализации Nmap. Первая: запущенный с правами обычного пользователя Nmap крайне неэффективен. Весь процесс сканирования при этом фактически сводится к попытке установить полноценное соединение с каждым из портов. В случае протокола TCP это значит, что Nmap пошлет на удаленную сторону пакет SYN; если запрошенный порт открыт, машина ответит пакетом SYN/ACK, после чего Nmap отправит пакет ACK и только потом закроет соединение с помощью пакета FIN.

Весь этот процесс осуществляется с помощью системного вызова connect() и, по сути, целиком ложится на сетевой стек ОС. В результате страдает как производительность сканирования, так и скрытность (сервисы, висящие на сканируемых портах, будут активно логировать соединения).

Совсем по-другому Nmap ведет себя, когда запущен с правами root:

```
sudo nmap 192.168.0.1
```

В этом случае он полностью берет на себя формирование пакетов и управление соединением. Подключение к открытым портам будет выглядеть так: Nmap посылает SYN, машина отвечает SYN/ACK, Nmap посылает FIN, разрывая наполовину открытое соединение (это называется TCP SYN сканирование). В результате сервис, висящий на порте, не логирует попытку соединения, а Nmap способен обнаружить брандмауэр, который просто отбрасывает SYN-пакеты вместо того, чтобы отправить в ответ SYN/ACK (порт открыт) или FIN (порт закрыт), как это должна делать операционная

система по умолчанию. Кроме того, этот способ сканирования намного более производительный.

Вторая особенность Nmap заключается в том, что на самом деле он сканирует не весь диапазон портов (65 536), а только 1000 портов типовых служб, определенных в файле `/usr/share/nmap/nmap-services`. Так что, если кто-то повесит сервис на нестандартный порт, которого просто нет в этом файле, Nmap его не увидит. Изменить подобное поведение можно при помощи такой команды:

```
sudo nmap -sS -sU -p 1-65535 192.168.0.1
```

Nmap будет использовать сканирование типа TCP SYN и UDP-сканирование для всего диапазона портов.

4.2 Определяем название и версию сервиса на порте

Одна из интересных особенностей Nmap в том, что он способен не только определить состояние порта (открыт, закрыт, фильтруется), но и идентифицировать имя демона/службы на этом порте, а в некоторых случаях даже его версию. Для этого Nmap может применять несколько разных техник, например подключиться к порту 80 и послать HTTP-запрос для идентификации имени и версии веб-сервера либо использовать информацию о том, как сервис отвечает на те или иные пакеты и запросы.

Все правила идентификации служб и их версий определены в файле `/usr/share/nmap/nmap-service-probes`, а заставит Nmap их применить флаг `-sV`:

```
sudo nmap -sV 192.168.0.1
```

Причем есть возможность даже усилить попытки Nmap определить службу с помощью флага `—version-all`:

```
sudo nmap -sv -version-all 192.168.0.1
```

Однако обычно это не повышает качество распознавания.

4.3 Определяем имя и версию ОС

Наверное, это одна из самых известных функций Nmap. Отправляя машине нестандартные пакеты и сопоставляя ее ответ (время ответа, значения полей TTL, MTU, ACK и многое другое) с «базой отпечатков ОС» (/usr/share/nmap/nmap-os-db), Nmap способен достаточно точно определить запущенную на машине ОС. Все, что нужно сделать, — это запустить Nmap с флагом -O:

```
sudo nmap -O 192.168.0.1
```

Однако далеко не всегда Nmap способен на 100% правильно угадать ОС. Если сделать это не получится, Nmap выведет на экран наиболее близкие к правильному варианты, заботливо снабдив их «процентом попадания»: 90%, 82%...

Более того, можно пойти еще дальше и воспользоваться флагом -A, чтобы заставить Nmap попытаться определить версию ОС, версию служб и даже провести более детальный анализ служб с помощью скриптов NSE (о них позже):

```
sudo nmap -A 192.168.0.1
```

4.4 Повышаем скорость сканирования

Nmap известен своей феноменальной производительностью, которая продолжает улучшаться вот уже двадцать лет. Сканер применяет множество техник и хаков, многопоточный режим с динамически изменяемым количеством портов на поток, умеет использовать несколько ядер процессора.

Но даже с целым вагоном оптимизаций в своем коде по умолчанию Nmap не такой быстрый, каким мог бы быть на самом деле.

Nmap поддерживает десяток флагов, позволяющих тонко контролировать такие параметры, как задержка между попытками подключения к порту или количество попыток подключения. Разобраться с ними с наскоку довольно тяжело, поэтому в Nmap есть набор преднастроенных шаблонов агрессивности сканирования. Всего их шесть (от 0 до 5), а сделать выбор можно с помощью опции -T:

```
sudo nmap -T3 192.168.0.1
```

В данном случае мы выбрали шаблон номер 3. Это дефолтовое значение, своеобразный компромисс между скоростью и точностью сканирования в медленных сетях. Однако в современных условиях, когда минимальная скорость проводного доступа в сеть уже перешагнула за 30 Мбит/с, лучшим выбором будет -T4 или даже -T5. Последний стоит применять только в стабильных сетях без провалов скорости.

Более низкие значения предназначены для обхода систем обнаружения вторжений. Например, -T0 отключает многопоточное сканирование и устанавливает задержку между пробами портов в пять минут; потратив весь день (ночь) на сканирование, ты можешь надеяться, что сама попытка сканирования будет не замечена (тем более что Nmap перебирает порты в случайном порядке). Шаблон -T1 — более быстрый режим с задержкой в 15 с, -T2 — 0,4 с, -T3 — 10 мс, -T4 — 5 мс.

4.5 Скрываем следы

Увеличение задержки между пробами портов — не единственный способ скрыть, что машина сканируется. Еще один интересный метод — одновременно запустить несколько потоков сканирования, подменяя

обратный IP-адрес во всех случаях, кроме одного. Смысл здесь в том, чтобы запутать IDS и администратора машины. В логах IDS окажется сразу несколько попыток сканирования с разных адресов, среди которых будет только один настоящий.

Использовать такой метод довольно просто:

```
sudo nmap -D адрес1, адрес2, адрес3 192.168.0.1
```

Можно указать сколько угодно фиктивных адресов или позволить Nmap сгенерировать рандомные адреса за тебя (в данном случае десять адресов):

```
sudo nmap -D RND:10 192.168.0.1
```

Однако тут необходимо иметь в виду, что рандомные адреса часто будут указывать на несуществующие или отключенные машины. IDS и хороший админ смогут отсеять их, чтобы вычислить реальный адрес.

Более сложный способ — организовать так называемое Idle-сканирование. Это очень интересная техника, которая базируется на трех простых фактах:

- При выполнении SYN-сканирования удаленная сторона посылает пакет SYN/ACK в случае, если порт открыт, и пакет RST, если нет.
- Машина, получившая незапрошенный пакет SYN/ACK, должна отвечать пакетом RST, а при получении незапрошенного RST — игнорировать его.
- Каждый IP-пакет, отправленный машиной, имеет IPID, а многие ОС при отправке пакета просто увеличивают IPID.

Сама техника заключается в том, чтобы найти неактивную сетевую машину, которая просто ничего не делает (Idle), но при этом находится в рабочем

состоянии и способна отвечать на сетевые запросы. Более того, машина должна работать на древней ОС, которая увеличивает IPID пакетов вместо рандомизации, как современные ОС. Сделать это можно с помощью все тех же флагов -O -v Nmap (строка IP ID Sequence Generation в выводе) либо с помощью Metasploit Framework (это удобнее и быстрее):

```
> use auxiliary/scanner/ip/ipidseq
> set RHOSTS 192.168.0.1-192.168.0.255
> run
```

Далее запускаем сканирование портов:

```
sudo nmap -sI IP-Idle-машины 192.168.0.1
```

На каждую пробу порта Nmap сначала будет посылать запрос Idle-машине, записывать IPID пакета, затем посылать SYN-пакет жертве, подменяя обратный адрес на IP Idle-машины, затем снова посылать запрос Idle-машине и сверять IPID с ранее сохраненным. Если IPID увеличился со времени прошлой проверки, значит, машина посылала пакеты, а, как мы знаем из второго пункта выше, это означает, что она ответила пакетом RST. Это, в свою очередь, говорит, что проверяемый порт жертвы открыт. Если IPID не увеличился, значит, порт закрыт.

В современном мире, где уже не осталось Windows 95, это действительно сложно реализуемая техника, но она позволяет полностью отвести от себя подозрения о сканировании. IDS обвинит в сканировании Idle-машину.

4.6 Обходим IDS и брандмауэры

Сегодняшние IDS и брандмауэры намного умнее тех, что существовали во времена, когда в Nmap появились средства борьбы с ними. Поэтому многие приведенные здесь техники могут уже не работать, но это не повод ими не пользоваться, в Сети полно доисторического оборудования.

Начнем с того, что даже без дополнительных опций Nmap уже способен хоть и не обойти, но обнаружить брандмауэр. Происходит так потому, что при SYN-сканировании состояние открыт/закрыт определяется путем анализа ответа машины: SYN/ACK — открыт, FIN — закрыт. Однако брандмауэры, чтобы минимизировать процессорные ресурсы, зачастую просто дропают пакеты, адресуемые фильтруемым портам (даже при настройке iptables в Linux стандартная практика — это дропнуть пакет с помощью -j DROP). Nmap отслеживает, при пробе каких портов не было получено ответа, и помечает эти порты filtered.

Еще одна техника обнаружения брандмауэра заключается в том, чтобы заставить Nmap генерировать «невероятные пакеты», такие как пакеты без единого флага (-sN), FIN-пакеты (-sF) и Xmas-пакеты, содержащие флаги FIN, PSH и URG (-sX). RFC описывает все эти ситуации, поэтому любое расхождение с RFC Nmap интерпретирует как наличие брандмауэра.

Многие брандмауэры можно обойти и точно определить, фильтруется порт или нет. Для этого можно использовать ACK-сканирование:

```
sudo nmap -sA 192.168.0.1
```

Теория здесь следующая: брандмауэр должен отбивать все **новые** TCP-подключения к порту, но также обязан не препятствовать прохождению пакетов в рамках уже установленных соединений. Простой способ сделать это — отбивать все SYN-пакеты (используется для установки соединения), но не мешать ACK-пакетам (используется для отправки пакетов в рамках уже открытого соединения).

Но есть одна тонкость. Дело в том, что есть так называемые stateful-брандмауэры. Они умеют отслеживать состояние соединения и проверяют такие поля пакетов, как IP-адрес и номер последовательности TCP, чтобы

отслеживать, какие пакеты действительно пришли в рамках открытого ранее соединения, а какие были отправлены Nmap в рамках АСК-сканирования (iptables в Linux может работать в обоих режимах, но по умолчанию он не stateful, это более производительный вариант).

Выяснить, какой тип брандмауэра используется, можно, выполнив SYN-сканирование и сразу за ним — АСК-сканирование:

```
$ sudo nmap -sS 192.168.0.1  
$ sudo nmap -sA 192.168.0.1
```

Если во втором случае порты, отмеченные во время SYN-сканирования как filtered, стали unfiltered, значит, перед тобой не stateful-брандмауэр.

Кроме того, можно попробовать обойти брандмауэр с помощью изменения номера исходящего порта:

```
$ sudo nmap --source-port 53 192.168.0.1
```

Это эксплуатация старой как мир ошибки настройки брандмауэра, которая заключается в том, что админ открывает доступ всему входящему трафику (включая протокол TCP) с порта 53, чтобы позволить приложениям беспрепятственно выполнять DNS-запросы. Сегодня такое встречается редко, но, как показывает практика, некомпетентность со временем не исчезает.

Кроме всего прочего, в Nmap есть средства для скрытия факта сканирования от глаз брандмауэров и IDS:

```
sudo nmap -f 192.168.0.1
```

В этом случае Nmap будет разбивать пакеты на крохотные фрагменты размером 8 байт. Делает он это в надежде на то, что брандмауэр или IDS не сможет собрать пакет из фрагментов и проанализировать его заголовок (по

причине плохой реализации или в угоду производительности) и просто пропустит пакет или отбросит.

```
sudo nmap -mtu 16 192.168.0.1
```

Та же история, только с возможностью контролировать размер пакета (в данном случае 16). Можно использовать против брандмауэров и IDS, которые умеют ловить факты сканирования с помощью Nmap, анализируя размер фрагмента.

```
sudo nmap -data-length 25 192.168.0.1
```

Добавляет в конец пакета указанное количество рандомных байтов. Цель та же, что и в предыдущем случае: обмануть IDS, которая может быть способна обнаружить сканирование, анализируя размер пакета (Nmap всегда посылает пакеты длиной 40 байт при использовании протокола TCP).

4.7 Используем Nmap для обнаружения машин в сети

Хотя Nmap известен именно как сканер портов, это также отличный инструмент для обнаружения машин в сети. Его можно натравить на любое количество хостов и буквально за несколько секунд получить результат пинга тысяч хостов. Вот только пингует хосты он совсем не так, как всем известная утилита ping. По умолчанию перед началом сканирования портов Nmap посылает несколько пакетов, чтобы удостовериться в доступности хоста:

- ICMP Echo request — аналог того, как работает ping;
- SYN-пакет на порт 443;
- ACK-пакет на порт 80;
- ICMP timestamp request.

Так много способов проверки необходимы для обхода брандмауэров и ситуаций, когда, например, в ОС или сетевом оборудовании включен запрет отвечать на запросы ICMP Echo (сегодня это частая практика).

Проверку доступности легко отключить, используя опцию -PN, о чем сам Nmap сообщит, если не сможет удостовериться в доступности порта:

```
sudo nmap -PN 192.168.0.1
```

Обычно в этом мало смысла, а вот обратная операция, то есть отключение сканера портов, очень даже полезна для проверки доступности множества хостов:

```
sudo nmap -sn 192.168.0.1-255
```

Эта команда заставит Nmap просканировать адреса с 192.168.0.1 по 192.168.0.255. Ее более удобный аналог:

```
sudo nmap -sn 192.168.0.*
```

А так можно попросить Nmap просканировать всю подсеть:

```
sudo nmap -sn 192.168.0.0/24
```

Ну или записать необходимые адреса в файл и попросить просканировать их:

```
sudo nmap -sn -iL /путь/до/файла
```

Если опустить флаг -sn, Nmap будет не просто проверять доступность хостов, но еще и сканировать порты.

Самых техник пингования также довольно много. Nmap поддерживает определение доступности хоста с помощью послышки SYN-пакета на указанный порт:

```
sudo nmap -sn -PS80 192.168.0.1
```

АСК-пакета:

```
sudo nmap -sn -PA80 192.168.0.1
```

UDP-пакета:

```
sudo nmap -sn -PU53 192.168.0.1
```

ICMP Echo request:

```
sudo nmap -sn -PE 192.168.0.1
```

ICMP timestamp:

```
sudo nmap -sn -PP 192.168.0.1
```

Все их можно комбинировать:

```
sudo nmap -sn -PE -PS443 -PA80 -PP 192.168.0.1
```


5. Заключение

Nmap — это инструмент с открытым исходным кодом, который в основном используется администраторами сети для обнаружения узлов и сканирования портов.

Обратите внимание, что в некоторых странах сканирование сетей без разрешения является незаконным.