Задачу обнаружения хостов иногда называют пинг сканированием (ping scan). Целью всех этих запросов является получение ответов, указывающих, что IP адрес в настоящее время активен (используется хостом или сетевым устройством). В большинстве сетей лишь небольшой процент IP адресов активен в любой момент времени. Это особенно характерно для адресных пространств вида 10.0.0.0/8. Такие сети имеют 16 млн. IP адресов, но я видел, как они используются компаниями, в которых не более тысячи машин. Функция обнаружения хостов может найти эти машины в этом необъятном море IP адресов. Если не задано никаких опций обнаружения хостов, то Nmap посылает TCP ACK пакет на порт 80 и запрос на ICMP эхо ответ каждой целевой машине.

```
root@kali:~/Documents/lab4# nmap 192.168.0.104

Starting Nmap 7.01 ( https://nmap.org ) at 2016-03-20 19:52 EDT
Nmap scan report for 192.168.0.104
Host is up (1.0s latency).
Not shown: 992 closed ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
443/tcp   open  https
445/tcp   open  microsoft-ds
554/tcp   open  rtsp
2869/tcp  open  icslap
10243/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 3.60 seconds

root@kali:~/Documents/lab4# nmap 192.168.0.100-105

Starting Nmap 7.01 ( https://nmap.org ) at 2016-03-20 20:32 EDT

Nmap scan report for 192.168.0.100
Host is up (0.0059s latency).
All 1000 scanned ports on 192.168.0.100 are filtered

Nmap scan report for 192.168.0.101
Host is up (0.015s latency).
All 1000 scanned ports on 192.168.0.101 are filtered
```

```
Nmap scan report for 192.168.0.102
Host is up (0.0053s latency).
All 1000 scanned ports on 192.168.0.102 are filtered

Nmap scan report for 192.168.0.103
Host is up (0.081s latency).
All 1000 scanned ports on 192.168.0.103 are filtered

Nmap scan report for 192.168.0.104
Host is up (1.0s latency).
Not shown: 992 closed ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
443/tcp   open  https
445/tcp   open  microsoft-ds
554/tcp   open  rtsp
2869/tcp  open  icslap
10243/tcp open  unknown

Nmap scan report for 192.168.0.105
Host is up (0.0037s latency).
All 1000 scanned ports on 192.168.0.105 are filtered
```

Open означает, что приложение на целевой машине готово для принятия пакетов на этот порт. Filtered означает, что брандмауэр, фильтр, или что-то другое в сети блокирует порт, так что Nmap не может определить, является ли порт открытым или закрытым. Closed — не связанны в данный момент ни с каким приложением, но могут быть открыты в любой момент. Unfiltered порты отвечают на запросы Nmap, но нельзя определить, являются ли они открытыми или закрытыми.

```
root@kali:~/Documents/lab4# nmap -O 127.0.0.1

Starting Nmap 7.01 ( https://nmap.org ) at 2016-03-20 20:41 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000051s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE
```

```
5432/tcp open  postgresql
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.8 - 3.19
Network Distance: 0 hops

OS detection performed. Please report any incorrect results at https://nmap.org/
Nmap done: 1 IP address (1 host up) scanned in 2.13 seconds
```

Версии сервисов

Для определения ОС удаленного хоста, версия которой неизвестна, необходимо иметь определенную информацию о том, как ОС известных версий реагируют на определенные виды запросов, описанных выше, иначе говоря - составить "отпечаток"стека TCP/IP операционной системы. Алгоритм получения отпечатка стека TCP/IP следующий. Вначале проводится сканирование портов удаленного хоста с целью определения открытых портов и служб, функционирующих на исследуемом хосте. Затем проводится несколько тестов, поэтапно выполняющих опрос стека TCP/IP удаленного хоста с целью выявления признаков, определяющих версию служб. На основе полученных от хоста ответов составляется отпечаток, который затем сравнивается с уже имеющейся базой отпечатков, и принимается решение о типе и версии ОС исследуемого хоста.

```
root@kali:~/Documents/lab4# nmap -sV 192.168.0.104

Starting Nmap 7.01 ( https://nmap.org ) at 2016-03-20 20:42 EDT
Stats: 0:00:50 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 62.50% done; ETC: 20:43 (0:00:28 remaining)
Stats: 0:01:25 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 62.50% done; ETC: 20:44 (0:00:49 remaining)
Nmap scan report for 192.168.0.104
Host is up (1.0s latency).
Not shown: 992 closed ports
PORT      STATE SERVICE      VERSION
80/tcp    open  http
135/tcp   open  msrpc        Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows 98 netbios-ssn
443/tcp   open  https
445/tcp   open  microsoft-ds Microsoft Windows 7 or 10 microsoft-ds
554/tcp   open  rtsp?
```

```
2869/tcp  open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
10243/tcp open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
2 services unrecognized despite returning data. If you know the service/version,
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port80-TCP:V=7.01%I=7%D=3/20%Time=56EF4373%P=x86_64-pc-linux-gnu%r(GetR
SF:equest,1A,"HTTP/1\.0\x20404\x20Not\x20Found\r\n\r\n")%r(HTTPOptions,6B,
SF:"af\)\xcc\x99=\x0e\xd3\x022\x9a\xaf\x0fJ\x8e\x0e\x16\xe8\xcd\*\x84\xec\
SF:xef\xdc\x15\xbb\x02pt\xd6nU\xfar\xcfB\xa3\xd5\xe6\xe5\xa6A\xc9P\xfa\xb5
SF:g\x14\xe2k\x20\x11\x0e\xa7,\xad\xfa\xa3\xf8\t\xa6_\x84%\x12\xdb\xd0\x01
SF:>\x17\xdc\x9d\*\x13\xa8\xf9\xd6\xcf4\x15BK\x80\xf1n\x87\x8c\x8dZ\x83X\x
SF:e9\x06\?\xe4\x05r\xbb0\xe1\x9e\xf7<}\x8a\xf3\x08")%r(RTSPRequest,5C,"\x
SF:e78\xc5\x7f\xd9{i\xcd\xcf\xea5\x86W\xbf\x81W\xd6b\x0bZ\xd1\x93\xb8\x20\
SF:xdc\x7f\xa6H\xf59Vx\xea\x1e\x99\x19\$\x06\x12\x87!\x8e\x94z\xd0\xd6\x7f
SF:f\xe89\x16\x0ftU\x82\x8b\xc01\xae\xc7\xcc\xcd\x9a\xc3\x98\)F\x7f\$E\xb2
SF:\xfbp!\xde7\|\xbd\xca3H\x19v\xef\xd45\xe2k\x20\x11\x0e\xa7")%r(FourOhFo
SF:urRequest,1A,"HTTP/1\.0\x20404\x20Not\x20Found\r\n\r\n")%r(RPCCheck,60,
SF:"\xc5\xe4\xd4\x08\x19\xf2\xc1U\x0c}\xf6\xb6\rF\x0f\x7f\xb7\x98\xd1;\xf7
SF:_\x1e\xc9t\xb4h\xd1\x9f\xf9\xed{\xde\xa7F\xea\t\xa0\xe7a\xbf<\xf0\xa8\x
SF:87AU\xc4\x03\xd8i\x86\xbfd\x85\xf2;\xb0a\x1ew\xbc\xfd\ns\x88Y\xb6/\x14u
SF:\"\xab‘QN\xe7l\xed:\xe38I\xe6\x9f\xc4eR\x1b\x10A~W\x1c\xddj")%r(DNSVers
SF:ionBindReq,6E,"\xd87h\x96\x9cT\?~\x9a\x0f\x91\xc4\x15\x9f\x8b\x01C\xc5\
SF:x91w\+p\xbam\xe7H\x9e\xc8\xc3\xa8\x95\xbbC\xff\xc5\x20\x86\x92\xf5\x01\
SF:xc5\xf7\x89m\xf4\xd1\x85\xbb\x96\x8f\xf4\xd5\x02\x0b@\xb1\.GLM\x1aC\x18
SF:\xa9\xc6\xff\xa4\xc52{\xf0\xa1\^\xb7\xfc=J\xb3\xc8\x99\xf6oT\xb5b\xeb\x
SF:a0\x91\x8e’\xac-z#x\x89&\xdf\x04\xa5\x92\[P\x81\xbe\x97\\\x1d\xaa\x93")
SF:%r(DNSStatusRequest,3C,"\x87\x12AZ=\xa4\xaf\xdfi\x1b\n\xa2\xaeM\x9fd@\x
SF:cc\x04\xed\xddo\xecce\*{\x11\x16n%\t\xca\+\x067>\x93I\x87mq\xce\xbe\xa9
SF:g\xf0\xf8\x10A~W\x1c\xddjS\xe89\x16\x0f")%r(SSLSessionReq,44,"\xe1E\x95
SF:\xd8\x19\xff\xc0\"\x15\xd1\xbc\xa5\(\xc2\xcd\xdc\x0b\xdf\t\xdf@\xe9\x80
SF:\x059\xd2\x7f\xbd\t8\xba\x0b\xbb\xee\xf7}’-\xa3h\x8b\xa1\x81\xd9\x9d\x8
SF:b\xe8\x9d\xbd\xca3H\x19v\xef\xd45\xe2k\x20Z\x97\xdfA\x10\x1a\xc3\x13")%
SF:r(TLSSessionReq,55,"\x9b\xc4\xcfF<g\xc4\xe7\xefU\$7p\x02\xc3\xdd\xaa\xe
SF:0i\x96\]7\x7f\xfb\\\.\x1d\xf5\x86\xa7\x18/\xa4\x9f\x98\xac\x9d:\x83\x84
SF:\xdf\xe7\xd9\x07P6\x9d\xb0l\xed:\xe38I\xe6\x9f\xc4eR\x1b\x10A~W\x1c\xdd
SF:jS\xe89\x16\x0ftU\x82\x8b\xc0\xc62\+\xdf\xe2\x9d\xa0\(");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port443-TCP:V=7.01%I=7%D=3/20%Time=56EF4378%P=x86_64-pc-linux-gnu%r(SSL
SF:SessionReq,46,"\xeeE\^\xc9\xf8k\xc7\xfa\0\xcd\xc5\x96\x85U\xc2\xc2\x88‘
SF:\xb7\xed03\xc0bXTv:\r5Kuh\xdcF\xb1\(\xea\$\x1dc\x11\x1c\xf1J\xb5\x02\xa
SF:3\xf7<}\x8a\xf3\x08\xd96\xaf\x94\xf5\xa2\+\xe0r\x17J\xc7\x14YUE")%r(TLS
SF:SessionReq,57,"\x92_\x0e\x8a\x98\x9b\x04t\xb5v\x03\xbc\x83B\xeeJ\x8e\xa
```

```
SF:5\$>D\x83\xfb\xe0\xfe}\xb3>\x8b\xba\x16\xf3\x89\xde\xe0\xde\xce\xda\xdb
SF:\x14\xf3P\(\|\xe6\xabmQ\x0b@\xb1\.GLM\x1aC\x18\xa9\xc6\xff\xa4\xc52{\xf
SF:0\xa1\^\xb7\xfc=J\xb3\xc8\x99\xf6oT\xb5\x9f\xd7\xe4\\HT\x92\xb7")%r(SSL
SF:v23SessionReq,42,"K\x92\x83\xac#\x9a0\n\xb3\xd8'r\xf1\xed\xca\xdc\xd4\x
SF:c4\xe0f&\xc1\xf4\xe2\x98\x169V\xbe\x92\+\x0cN\xb5\x10@\0\x1br\x0e\xcdB\
SF:$\xe1yMy\xf6\xc8\x99\xf6oT\xb5b\xeb\xa0\x91\x8e'\xac-z#x\x89")%r(GetReq
SF:uest,1A,"HTTP/1\.0\x20404\x20Not\x20Found\r\n\r\n")%r(HTTPOptions,5C,"/
SF:\xcc\xa4B\x05\x909\xe1\^\xb7\xc9aO\xda\x12E\xb0\.\xa0\x19\x04\xad\x1d\x
SF:05\xfa\x18V\*\xf2\xca\x15\xba\xf3\xc9i\xba\x8c\xce\xc9\xe1\xd1Y\^4\x81\
SF:x13\xbc>\xa8\xf9\xd6\xcf4\x15BK\x80\xf1n\x87\x8c\x8dZ\x83X\xe9\x06\?\xe
SF:4\x05r\xbb0\xe1\x9e\xf7<}\x8a\xf3\x08\xd96\xaf\x94\xf5\xa2\+\xe0\xd1\xc
SF:eg")%r(RTSPRequest,4E,"\xd7j\x83}\x07\x15\xb2\xa3\xc0\x93P\xb8p\x7f\xb2
SF:42\x80\x11y\x8eZ\x01\xb1\xc2\xb5\^\x1a-\xedv\x02\xdf\xda\x9b\xc8\x06\xf
SF:4\xf4\xdeUN-u\x10\xe1\x814\x20\x11\x0e\xa7,\xad\xfa\xa3\xf8\t\xa6_\x84%
SF:\x12\xdb\xd0\x01>\x17\xdc\x9d\*\x13\xa8\xf9\xd6\xcf4\x15")%r(RPCCheck,3
SF:0,"\xad\xa6\xb5\x19w\xb4\xcf\xfe\xba\xf8\xcf\xd0u\xa8\xdd~\x9elRB\xdd\x
SF:1f\x16\xd9\xb5R\xe5\x02a}\xe7\x18\xdf\?~\xe0o\x9b\x10a}\xad\xf7G\xc1\xd
SF:3k\x02")%r(DNSVersionBindReq,53,"7w\xf9\xa0\x7fbQ~o\x8bC\x87o\\\xc5\xfd
SF:\x19k\xaf\xf3\xa3\xb6R5X\xf4\xf5\x92\x9e\xb3~\x8b\xa5\xbf\xec0v\]\xf2\$
SF:\x0c\)\x15Q\xdc\)C@\?\xe4\x05r\xbb0\xe1\x9e\xf7<}\x8a\xf3\x08\xd96\xaf\
SF:x94\xf5\xa2\+\xe0\xd1\xceg\xecm\xbac\xb8\xc9f\x1fD\xe5")%r(DNSStatusReq
SF:uest,30,"yL\xd0\xe1f\xe3=r\xa1\xb6\x86Kv\xc4\xf5!6p\xabc\xc4\^I\t\xca\x
SF:fc\xaa\x88\x1ez\*\xa9j\x042\x81x\xea\xa5\xd0\xd2\x1ese\xaf\xf9k\x97")%r
SF:(Kerberos,54,"\xca\xee\x85F\xc0\xdc\xa7\xa7F\xfb\xb8\xea'R2\xe9\x90\xe0
SF:\xda\xd5Q\xf2\x18N\xb0B\xe29\xc2\+\x9e0;\x18\xb8I\x8a\^\xfb\x05\xa2w\xb
SF:9@\xc0'\x1be\x1ew\xbc\xfd\ns\x88Y\xb6/\x14u\"\xab‘QN\xe7l\xed:\xe38I\xe
SF:6\x9f\xc4e\x89\|\n\xbfb\x06\xbe\xb8");
Service Info: OSs: Windows, Windows 98; CPE: cpe:/o:microsoft:windows, cpe:/o:mi

Service detection performed. Please report any incorrect results at https://nmap
Nmap done: 1 IP address (1 host up) scanned in 123.06 seconds
```

Не зависимо от того, насколько технически грамотно реализована система определения версий, от нее не будет никакого толка, пока не наберется внушительная база отпечатков различных сервисов. В настоящее время существует база, которая содержит десятки тысяч отпечатков различных операционных систем и устройств. Чтобв определить, какой службе и версии соответствует отпечаток, проводятся специальные тесты. Если служба отвечает на один или более тестов, а Nmap не может определить ее, он выведет отпечаток службы наподобие этого:

```
SF:uest,30,"yL\xd0\xe1f\xe3=r\xa1\xb6\x86Kv\xc4\xf5!6p\xabc\xc4\^I\t\xca\x
```

```
SF:9@\xc0'\x1be\x1ew\xbc\xfd\ns\x88Y\xb6/\x14u\"\xab‘QN\xe7l\xed:\xe38I\xe
```

Это значит, что такой службы нет в базе.

Изучене файлов nmap-services, nmap-os-db, nmap-service-probes

nmap-services Содержит в себе все возможные порты, свыше 2200 называний общеизвестных служб, соответсвующие некоторым портам, котором напротив каждого номера обнаруженного порта nmap укажет возможное назначение этого порта: относится ли он к почтовому серверу (SMTP), веб-серверу (HTTP) или к службе DNS

```
at 4.txt
# THIS FILE IS GENERATED AUTOMATICALLY FROM A MASTER - DO NOT EDIT.
# EDIT /nmap-private-dev/nmap-services-all IN SVN INSTEAD.
# Well known service port numbers -*- mode: fundamental; -*-
# From the Nmap Security Scanner ( http://nmap.org )
#
# $Id: nmap-services 35292 2015-10-02 07:52:30Z fyodor $
#
# Derived from IANA data and our own research
#
# This collection of service data is (C) 1996-2011 by Insecure.Com
# LLC.  It is distributed under the Nmap Open Source license as
# provided in the COPYING file of the source distribution or at
# http://nmap.org/data/COPYING .  Note that this license
# requires you to license your own work under a compatable open source
# license.  If you wish to embed Nmap technology into proprietary
# software, we sell alternative licenses (contact sales@insecure.com).
# Dozens of software vendors already license Nmap technology such as
# host discovery, port scanning, OS detection, and version detection.
# For more details, see http://nmap.org/book/man-legal.html
#
# Fields in this file are: Service name, portnum/protocol, open-frequency, optio
#
tcpmux 1/tcp 0.001995 # TCP Port Service Multiplexer [rfc-1078]
tcpmux 1/udp 0.001236 # TCP Port Service Multiplexer
compressnet 2/tcp 0.000013 # Management Utility
compressnet 2/udp 0.001845 # Management Utility
compressnet 3/tcp 0.001242 # Compression Process
compressnet 3/udp 0.001532 # Compression Process
unknown 4/tcp 0.000477
rje 5/udp 0.000593 # Remote Job Entry
unknown 6/tcp 0.000502
```

```
echo 7/sctp 0.000000
echo 7/tcp 0.004855
echo 7/udp 0.024679
unknown 8/tcp 0.000013
discard 9/sctp 0.000000 # sink null
discard 9/tcp 0.003764 # sink null
discard 9/udp 0.015733 # sink null
unknown 10/tcp 0.000063
systat 11/tcp 0.000075 # Active Users
systat 11/udp 0.000577 # Active Users
unknown 12/tcp 0.000063
daytime 13/tcp 0.003927
daytime 13/udp 0.004827
unknown 14/tcp 0.000038
netstat 15/tcp 0.000038
unknown 16/tcp 0.000050
qotd 17/tcp 0.002346 # Quote of the Day
qotd 17/udp 0.009209 # Quote of the Day
msp 18/udp 0.000610 # Message Send Protocol
chargen 19/tcp 0.002559 # ttytst source Character Generator
chargen 19/udp 0.015865 # ttytst source Character Generator
ftp-data 20/sctp 0.000000 # File Transfer [Default Data]
ftp-data 20/tcp 0.001079 # File Transfer [Default Data]
ftp-data 20/udp 0.001878 # File Transfer [Default Data]
ftp 21/sctp 0.000000 # File Transfer [Control]
ftp 21/tcp 0.197667 # File Transfer [Control]
ftp 21/udp 0.004844 # File Transfer [Control]
ssh 22/sctp 0.000000 # Secure Shell Login
ssh 22/tcp 0.182286 # Secure Shell Login
ssh 22/udp 0.003905 # Secure Shell Login
telnet 23/tcp 0.221265
telnet 23/udp 0.006211
priv-mail 24/tcp 0.001154 # any private mail system
priv-mail 24/udp 0.000329 # any private mail system
smtp 25/tcp 0.131314 # Simple Mail Transfer
smtp 25/udp 0.001285 # Simple Mail Transfer
rsftp 26/tcp 0.007991 # RSFTP
nsw-fe 27/tcp 0.000138 # NSW User System FE
nsw-fe 27/udp 0.000395 # NSW User System FE
unknown 28/tcp 0.000050
msg-icp 29/tcp 0.000025 # MSG ICP
```

```
msg-icp 29/udp 0.000560 # MSG ICP
unknown 30/tcp 0.000527
```

nmap-service-probes

После того как какие-либо TCP и/или UDP были обнаружены, Nmap начинает "опрашивать"эти порты, чтобы определить, какие же приложения (службы) их действительно используют. База данных nmap-service-probes содержит запросы для обращения к различным службам и соответствующие выражения для распознавания и анализа ответов. Nmap пытается определить протоколо службы (напр. FTP, SSH, Telnet, HTTP), имя приложения (e.g. ISC BIND, Apache httpd, Solaris telnetd), номер версии, имя хоста, тип устройства (напр. принтер, роутер), семейство ОС (напр. Windows, Linux) и иногда различные детали типа возможно ли соединится с X сервером, версию протокола SSH

Как принято в файлах ОС UNIX, nmap-service-probes состоит из строк. Строки, начинающиеся с символа «hash» (#) воспринимаются как комментарии и игнорируются обработчиком. Пустые строки также не обрабатываются. Строки, подлежащие обработке, должны содержать следующие директивы:

Probe <protocol> <probename> <probesendstring>

Пример:

```
Probe TCP GetRequest q|GET / HTTP/1.0\r\n\r\n|
Probe UDP DNSStatusRequest q|\0\0\x10\0\0\0\0\0\0\0\0\0|
Probe TCP NULL q||
```

Директива «probe» (тест) указывает Nmap, какие данные отправлять в процессе определения служб. Аргументы этой директивы следующие:

Protocol – тип протокола. Может быть указан один из протоколов TCP или UDP. Nmap будет использовать только те тесты, тип протокола которых совпадает с рабочтм протоколом проверяемой службы.

Probename – название теста. Используется в отпечатке службы для указания, на какой тест был получен ответ. Название может быть произвольным (удобным для пользователя).

Probestring –строка, используемая для тестового запроса. Должна начинаться и заканчиваться символом-ограничителем «q». Между ограничителями находится непосредственно сама строка, передаваемая в качестве теста. Эта строка имеет формат, аналогичный строкам языков C или Perl, и может содержать стандартные escape-последовательности:

```
\\ \0 \a \b \f \n \r \t \v \xHH
```

. В последнем примере показано, что тестовая строка может быть пустой. Это и есть тот самый «нуль-тест», при котором данные на порт не отправляются.

match <service> <pattern> [versioninfo]

Пример:

```
match domain m|^\0\0\x90\x04\0\0\0\0\0\0\0\0|
match argus m|^\x80\x01\0\x80\0\x80\0\0\xe5az\xcb\0\0\0\0J.............\x02\0\
match ssh m|^SSH-([\d.]+)-OpenSSH[_-]([\w.]+)\r?\n|i p/OpenSSH/ v/$2/ i/protocol
```

Директива «match» указывает Nmap на то, как точно определить службу, используя полученный ответ на запрос, отправленный предыдущей директивой. Эта директива используется в случае, когда полученный ответ полностью совпадает с шаблоном. При этом тестирование порта считается законченным, а при помощи дополнительных спецификаторов Nmap строит отчет о названии приложения, номере версии и дополнительной информации, полученной в ходе проверки. Директива имеет следующие аргументы:

Service – название службы, для которой приведен шаблон. Например, ssh, smtp, http, или SNMP.

Pattern – шаблон, с которым должен совпадать полученный ответ. Формат шаблона аналогичен принятому в языке Perl, и имеет следующий синтаксис: «m/[regex]/[opts]». Литерал «m» указывает на начало строки. Прямой слэш ('/') является разделителем, вместо которого может быть подставлен любой печатаемый символ (при этом вместо второго слэша должен быть подставлен такой же символ). Regex – это регулярное выражение, принятое в языке Perl. В настоящее время поддерживаются только две опции – это 'i' (снимает чувствительность выражения к регистру) и 's', включающая символ перевода строки в спецификаторе типа '.' .

Versioninfo – это поле имеет следующий формат: v/vendorproductname/version/info/, где слэш может быть заменен любым разделителем. Любое из трех полей может быть пустым. Кроме этого, поле само может быть пустым, и это означает, что дополнительная информация о службе отсутствует. Поле vendorproductname содержит название производителя и имя службы, например, «Sun Solaris rexecd», «ISC Bind named», или «Apache httpd». Поле version содержит «номер» версии (в кавычках потому, что может обозначаться не числовым значением, а напротив, состоять из нескольких слов). Поле info содержит дополнительную полезную информацию, которая может пригодиться на этапе сканирования (например, номер протокола сервера ssh).

softmatch <service> <pattern>

Примеры:

```
softmatch ssh m|^SSH-([\d.]+)-| i/protocol $1/
softmatch ppp m|^\x7e\xff\x7d\x23.*\x7e|
```

Директива softmatch имеет формат, аналогичный директиве match. Основное отличие заключается в том, что после совпадения принятого ответа с одним из шаблонов softmatch, тестирование будет продолжено с использованием только тех тестов, которые относятся к определенной шаблоном службе. Тестирование порта будет идти до тех пор, пока не будет найдено строгое соответствие («match») или не закончатся все тесты для данной службы. Аргументы те же самые, только отсутствует versioninfo.

ports <portlist> и sslports <portlist>

Пример:

```
ports 21,23,35,43,79,98,110,113,119,199,214,264,449,505,510,540,587,616,628,666,
sslports 989,990,992,995
```

Эта директива группирует порты, которые обычно закрепляются за идентифицируемой данным тестом службой. Синтаксис представляет собой упрощенный формат опции '-p'. Директива sslports указывает порты, обычно используемые совместно с SSL

totalwaitms <milliseconds>

Пример:

```
totalwaitms 5000
```

Редко используемая директива. Она указывает, сколько времени (в миллисекундах) необходимо ждать ответ, прежде чем прекратить тест службы.

nmap-os-db

Одна из наиболее известных функциональных возможностей Nmap это удаленное определение ОС на основе анализа работы стека TCP/IP. Nmap посылает серию TCP и UDP пакетов на удаленный хост и изучает практически каждый бит в ответах. После проведения дюжины тестов, таких как TCP ISN выборки, поддержки опций TCP, IP ID выборки, и анализа продолжительности процедуры инициализации, Nmap сравнивает результаты со своей nmap-os-db базой данных, состоящей из более чем тысячи известных наборов типичных результатов для различных ОС и, при нахождении соответствий, выводит информацию об ОС. Каждый набор содержит свободное текстовое описание ОС и классификацию, в которой указаны название производителя (напр. Sun), название ОС

(напр. Solaris), поколение ОС (напр. 10), и тип устройства (). OS, and a classification which provides the vendor name (e.g. Sun), underlying OS (e.g. Solaris), OS generation (e.g. 10), and device type (для общих целей, роутер, коммутатор (switch), игровая консоль и т.д.).

Пример:

```
# Windows 10 build 10240
Fingerprint Microsoft Windows 10 build 10240
Class Microsoft | Windows | 10 | general purpose
CPE cpe:/o:microsoft:windows_10 auto
SEQ(SP=104-10E%GCD=1-6%ISR=106-110%TI=I%CI=I%II=I%SS=S%TS=A)
OPS(O1=M5BCNW8ST11%O2=M5BCNW8ST11%O3=M5BCNW8NNT11%O4=M5BCNW8ST11%O5=M5BCNW8ST11%
WIN(W1=2000%W2=2000%W3=2000%W4=2000%W5=2000%W6=2000)
ECN(R=Y%DF=Y%T=7B-85%TG=80%W=2000%O=M5BCNW8NNS%CC=N%Q=)
T1(R=Y%DF=Y%T=7B-85%TG=80%S=O%A=S+%F=AS%RD=0%Q=)
T2(R=Y%DF=Y%T=7B-85%TG=80%W=0%S=Z%A=S%F=AR%O=%RD=0%Q=)
T3(R=Y%DF=Y%T=7B-85%TG=80%W=0%S=Z%A=O%F=AR%O=%RD=0%Q=)
T4(R=Y%DF=Y%T=7B-85%TG=80%W=0%S=A%A=O%F=R%O=%RD=0%Q=)
T5(R=Y%DF=Y%T=7B-85%TG=80%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)
T6(R=Y%DF=Y%T=7B-85%TG=80%W=0%S=A%A=O%F=R%O=%RD=0%Q=)
T7(R=Y%DF=Y%T=7B-85%TG=80%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)
U1(DF=N%T=7B-85%TG=80%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)
IE(DFI=N%T=7B-85%TG=80%CD=Z)
```

Добавить новую сигнатуру службы в файл nmap-service-probes (для этого создать минимальный tcp server, добиться, чтобы при сканировании nmap указывал для него название и версию)

Исходный код tcp-сервера

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <stdio.h>
#include <string.h>
#include <unistd.h>

int main(){
    char str[100];
char *resp = "Hello, Server 1.0\n";
    int listen_fd, comm_fd;

    struct sockaddr_in servaddr;
```

```
    listen_fd = socket(AF_INET, SOCK_STREAM, 0);

    bzero( &servaddr, sizeof(servaddr));

    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htons(INADDR_ANY);
    servaddr.sin_port = htons(11089);

    bind(listen_fd, (struct sockaddr *) &servaddr, sizeof(servaddr));
    listen(listen_fd, 10);

    comm_fd = accept(listen_fd, (struct sockaddr*) NULL, NULL);
    while(1){
        bzero( str, 100);
        read(comm_fd,str,100);
        printf("Echoing back - %s",str);
        write(comm_fd, resp, strlen(resp)+1);
    }
}
```

```
root@kali:~/Documents/lab4# nmap -sV 127.0.0.1

Starting Nmap 7.01 ( https://nmap.org ) at 2016-04-03 17:15 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000020s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE     VERSION
5432/tcp  open  postgresql  PostgreSQL DB
11089/tcp open  HelloServer HelloServer 1.0
```

Сохранить вывод утилиты в формате xml

```
root@kali:~/Documents/lab4# nmap -oX out.xml -sV 127.0.0.1

Starting Nmap 7.01 ( https://nmap.org ) at 2016-04-03 17:35 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000020s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE     VERSION
5432/tcp  open  postgresql  PostgreSQL DB
11089/tcp open  HelloServer HelloServer 1.0
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE nmaprun>
<?xml-stylesheet href="file:///usr/bin/../share/nmap/nmap.xsl" type="text/xsl"?>
<!-- Nmap 7.01 scan initiated Sun Apr  3 17:35:01 2016 as: nmap -oX out.xml -sV
<nmaprun scanner="nmap" args="nmap -oX out.xml -sV 127.0.0.1" start="1459719301"
<scaninfo type="syn" protocol="tcp" numservices="1000" services="1,3-4,6-7,9,13,
<verbose level="0"/>
<debugging level="0"/>
<host starttime="1459719301" endtime="1459719307"><status state="up" reason="loc
<address addr="127.0.0.1" addrtype="ipv4"/>
<hostnames>
<hostname name="localhost" type="PTR"/>
</hostnames>
<ports><extraports state="closed" count="998">
<extrareasons reason="resets" count="998"/>
</extraports>
<port protocol="tcp" portid="5432"><state state="open" reason="syn-ack" reason_t
<port protocol="tcp" portid="11089"><state state="open" reason="syn-ack" reason_
</ports>
<times srtt="2" rttvar="0" to="100000"/>
</host>
<runstats><finished time="1459719307" timestr="Sun Apr  3 17:35:07 2016" elapsed
</runstats>
</nmaprun>
```

Исследовать различные этапы и режимы работы nmap с использованием утилиты Wireshark

На рисунке ниже представлен результат анализа сканирования порта 445. Сначала на хост с IP-адресом 192.169.0.104 отправляется эхо-запрос для того, чтобы удостовериться в доступности хоста. После того, как мы получили ответ, отправляется запрос с пробой, соответствующей 445 порту из файла nmap-service-probes:

```
# SMB Negotiate Protocol
##############################NEXT PROBE##############################
Probe TCP SMBProgNeg q|\0\0\0\xa4\xff\x53\x4d\x42\x72\0\0\0\0\x08\x01\x40\0\0\0\
rarity 4
ports 42,88,135,139,445,660,1025,1027,1031,1112,3006,3900,5000,5009,5432,5555,56
```

В ответ целевой хост посылает сообщение, соответствующее шаблону:

```
match microsoft-ds m|^\0\0\0.\xffSMBr\0\0\0\0\x88\x01@\0\0\0\0\0\0\0\0\0\0\0\0
```

Результат:

```
root@kali:~/Documents/lab4# nmap -sV -p445 192.168.0.104

Starting Nmap 7.01 ( https://nmap.org ) at 2016-04-03 21:12 EDT
Nmap scan report for 192.168.0.104
Host is up (0.00100s latency).
PORT    STATE SERVICE      VERSION
445/tcp open  microsoft-ds Microsoft Windows 7 or 10 microsoft-ds
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows_7

Service detection performed. Please report any incorrect results at https://nmap
Nmap done: 1 IP address (1 host up) scanned in 6.86 seconds
```

-sV - определение версий -A - механизм определения версий и определение ОС (-O) -T4 - указывает Nmap использовать более агрессивную временную политику сканирования (с меньшими временными затратами) -F - включает режим сканирования только тех портов, которые перечислены в файле nmap-services