

Лабораторная работа №2.
Утилита ntar

Климов Сергей

4 апреля 2016 г.

Оглавление

1	Цель работы	2
2	Описание окружения	2
3	Ход работы	3
3.1	Поиск активных хостов	3
3.2	Определение открытых портов целевой ВМ	3
3.3	Определение версий сервисов	4
3.4	Исследование служебных файлов nmap-services, nmap-os-db, nmap-service-probes	5
3.5	Добавление собственной сигнатуры в файл nmap-service-probes	6
3.6	Сохранение вывода утилиты nmap в формате XML	8
3.7	Исследование работы утилиты nmap при помощи wireshark	9
3.8	Просканировать виртуальную машину Metasploitable2 используя db_nmap из состава metasploit-framework	9
3.9	Описать работу пяти записей из файла nmap-service-probes	11
3.10	Описать работу скрипта из состава Nmap	11
4	Выводы	13

1 Цель работы

Изучение принципов работы утилиты nmap.

2 Описание окружения

Для проведения лабораторной работы было подготовлено две виртуальные машины, объединенные в общую сеть. Первая виртуальная машина (Metasploitable2), которая намеренно содержит ряд уязвимостей - целевая ВМ. Вторая виртуальная машина (Kali Linux) необходима для сканирования и поиска уязвимостей на целевой ВМ.

Целевая ВМ с ОС Metasploitable2 имеет адрес 169.254.120.103, ВМ Kali Linux имеет адрес 169.254.120.101.

Конфигурация целевой ВМ:

```
root@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:70:24:f3
          inet addr:169.254.120.103  Bcast:169.254.255.255  Mask:255.255.0.0
          inet6 addr: fe80::a00:27ff:fe70:24f3/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:11515 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2319 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1119307 (1.0 MB)  TX bytes:230624 (225.2 KB)
          Base address:0xd010 Memory:f0000000-f0020000
```

Конфигурация ВМ Kali Linux:

```
root@kali:~# ifconfig
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 169.254.120.101  netmask 255.255.0.0  broadcast 169.254.255.255
        inet6 fe80::a00:27ff:fe11:e70a  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:11:e7:0a  txqueuelen 1000  (Ethernet)
        RX packets 8532  bytes 838582 (818.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 253  bytes 20495 (20.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Проверим доступность целевой ВМ при помощи утилиты ping:

```
root@kali:~# ping 169.254.120.103
PING 169.254.120.103 (169.254.120.103) 56(84) bytes of data.
64 bytes from 169.254.120.103: icmp_seq=1 ttl=64 time=0.554 ms
64 bytes from 169.254.120.103: icmp_seq=2 ttl=64 time=0.756 ms
64 bytes from 169.254.120.103: icmp_seq=3 ttl=64 time=0.697 ms
64 bytes from 169.254.120.103: icmp_seq=4 ttl=64 time=0.740 ms
^C
--- 169.254.120.103 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.554/0.686/0.756/0.085 ms
```

Как видно из результатов, целевая ВМ доступна в сети.

3 Ход работы

Дальнейшие действия будут выполняться с использованием утилиты nmap.

3.1 Поиск активных хостов

Для поиска активных хостов воспользуемся ключем -sP. Данный флаг указывает, что нужно искать сервера и устройства подключенные к сети и работающие в данный момент. В качестве параметра зададим нашу подсеть: 169.254.120.0/24. Результат сканирования подсети на наличие активных хостов:

```
root@kali:~# nmap -sP 169.254.120.0/24
```

```
Starting Nmap 7.01 ( https://nmap.org ) at 2016-04-03 05:24 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using -R
Nmap scan report for 169.254.120.2
Host is up (0.0017s latency).
MAC Address: 0A:00:27:00:00:15 (Unknown)
Nmap scan report for 169.254.120.100
Host is up (0.00089s latency).
MAC Address: 08:00:27:8C:08:58 (Oracle VirtualBox virtual NIC)
Nmap scan report for 169.254.120.103
Host is up (0.0011s latency).
MAC Address: 08:00:27:70:24:F3 (Oracle VirtualBox virtual NIC)
Nmap scan report for 169.254.120.101
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 1.82 seconds
```

В результате сканирования, утилита нашла 4 активных хоста. Среди них видим наши виртуальные машины с адресами 169.254.120.101 и 169.254.120.103.

3.2 Определение открытых портов целевой ВМ

Для сканирования портов хоста, утилите nmap необходимо передать адрес хоста, например:

```
root@kali:~# nmap 169.254.120.103
```

```
Starting Nmap 7.01 ( https://nmap.org ) at 2016-04-03 05:38 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using -R
Nmap scan report for 169.254.120.103
Host is up (0.00067s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
```

```

111/tcp open  rpcbind
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
512/tcp open  exec
513/tcp open  login
514/tcp open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 08:00:27:70:24:F3 (Oracle VirtualBox virtual NIC)

```

Nmap done: 1 IP address (1 host up) scanned in 0.23 seconds

Как видно из вывода, на целевой ВМ открыто множество портов, например, 21 - ftp, 22 - ssh, 23 - telnet, 80 - http, а так же ряд других портов.

3.3 Определение версий сервисов

Для определения версий сервисов воспользуемся ключем -sV и передадим адрес целевой ВМ в качестве аргумента.

```
root@kali:~# nmap -sV 169.254.120.103
```

```

Starting Nmap 7.01 ( https://nmap.org ) at 2016-04-03 05:43 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using
Nmap scan report for 169.254.120.103
Host is up (0.00092s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet       Linux telnetd
25/tcp    open  smtp         Postfix smtpd
53/tcp    open  domain       ISC BIND 9.4.2
80/tcp    open  http         Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind      2 (RPC #100000)
139/tcp   open  netbios-ssn  Samba smbd 3.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  Samba smbd 3.X (workgroup: WORKGROUP)
512/tcp   open  tcpwrapped
513/tcp   open  tcpwrapped
514/tcp   open  tcpwrapped
1099/tcp  open  rmiregistry  GNU Classpath grmiregistry
1524/tcp  open  tcpwrapped

```

```

2049/tcp open  nfs          2-4 (RPC #100003)
2121/tcp open  ftp          ProFTPD 1.3.1
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open  vnc         VNC (protocol 3.3)
6000/tcp open  X11         (access denied)
6667/tcp open  irc         Unreal ircd
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:70:24:F3 (Oracle VirtualBox virtual NIC)
Service Info: Hosts: metasploitable.localdomain, localhost, irc.Metasploitable.LAN; OSs:

```

Service detection performed. Please report any incorrect results at <https://nmap.org/submi>
Nmap done: 1 IP address (1 host up) scanned in 16.63 seconds

В результате видим, что утилита nmap вывела нам версии сервисов, запущенных на целевой ВМ.

3.4 Исследование служебных файлов nmap-services, nmap-os-db, nmap-service-probes

Служебные файлы для утилиты nmap по умолчанию располагаются в директории `"/usr/share/nmap"`.

Файл nmap-services

Файл nmap-services является реестром, где хранятся названия портов (служб на них работающих), их номера и названия протокола. <https://nmap.org/book/nmap-services.html> Файл имеет структуру таблицы со следующими столбцами: имя_сервиса, номер_порта/название_протокола, частота, комментарии. Часть данного файла:

```

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

```

Для "свободных"номеров портов, файл так же содержит записи, но они не несут никакой полезной информации, что ожидаемо, так как на этих портах запускаются пользовательские сервисы, и они не закреплены ни за одним приложением.

```

unknown 249/tcp 0.000050
unknown 250/tcp 0.000138
unknown 251/tcp 0.000125
unknown 252/tcp 0.000088

```

Файл nmap-os-db

Данный файл содержит сигнатуры ответов различных операционных систем при сканировании утилитой nmap. Это необходимо для того, что бы узнать какая операционная система находится на данном хосте. Пример файла nmap-os-db:

```
root@kali:~# head /usr/share/nmap/nmap-os-db -n92906 | tail -n10
# Arris Interactive, L.L.C. MODEL: TM402G SW_REV: 6.1.77D.D11PLUS
# bizhub C203
Fingerprint VxWorks
Class Wind River | VxWorks || general purpose
CPE cpe:/o:windriver:vxworks auto
SEQ(SP=FE-10A%GCD=1-6%ISR=FC-10E%TI=I%CI=I%II=I%SS=S%TS=8)
OPS(O1=M5B4NWOONNT11%O2=M5B4NWOONNT11%O3=M5B4NWOONNT11%O4=M5B4NWOONNT11%O5=M5B4NWOONNT11%O6=M5B
WIN(W1=4000%W2=4000%W3=4000%W4=4000%W5=4000%W6=4000)
ECN(R=Y%DF=Y%T=3B-45%TG=40%W=4000%O=M5B4NWO%CC=N%Q=)
T1(R=Y%DF=Y%T=3B-45%TG=40%S=0%A=S+%F=AS%RD=0%Q=)
```

Файл nmap-service-probes

Данный файл содержит сигнатуры для определения сервисов, прослушивающих тот или иной порт. Как правило, это относится к известным службам, например SMTP - почтовый сервис. Данные о сервисах задаются при помощи нескольких директив:

- Exclude `<port specification>`
- Probe `<protocol>` `<probenam>` `<probestring>`
- match `<service>` `<pattern>` [`<versioninfo>`]

3.5 Добавление собственной сигнатуры в файл ntpar-service-probes

Для того, что бы добавить собственную сигнатуру, создадим небольшой сервер, который мы будем идентифицировать при помощи утилиты `nslookup`. Код сервера представлен ниже:

```
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#define DEF_PORT 19931
int main(int argc, char** argv) {
    char str[100];
    char *sendStr="sergServ 0.3";
    struct sockaddr_in listenerInfo;
    listenerInfo.sin_family = AF_INET;
```

```

listenerInfo.sin_port = htons(DEF_PORT);
listenerInfo.sin_addr.s_addr = htonl(INADDR_ANY);
int listener = socket(AF_INET,SOCK_STREAM,0);
if(listener < 0) {
perror("Can't create socket to listen: ");
exit(1);
}
int res = bind(listener,(struct sockaddr *) &listenerInfo,sizeof(listenerInfo));
if(res < 0) {
perror("Can't bind socket");
exit(1);
}
res = listen(listener,5);
if(res) {
perror("Error while listening:");
exit(1);
}
int client = accept(listener,NULL,NULL);
while(1) {
bzero( str, 100);
recv(client,str, 100, 0);
printf("Message from client - %s",str);
send(client, sendStr, (int)strlen(sendStr), 0);
}
return 0;
}

```

Сервер работает на порте 19931. Серверу отправляется строка 'qqSerg' и он отвечает строкой со своим именем и версией: 'sergServ v0.3'.

Для определения данного сервера в файл nmap-service-probes были добавлены следующие строки:

```

####Test sergServ
Probe TCP sergServ q|qqSerg|
rarity 1
ports 19931
match sergServ m|sergServ (...+?)| v/$1/

```

Запустим на испытуемой виртуальной машине данный сервер и попробуем определить его при помощи утилиты nmap:

```
root@kali:~# nmap 169.254.120.101 -p 19931 -sV
```

```

Starting Nmap 7.01 ( https://nmap.org ) at 2016-04-03 22:49 MSK
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using
Nmap scan report for 169.254.120.101
Host is up (0.000067s latency).
PORT      STATE SERVICE  VERSION
19931/tcp  open  sergServ 0.3

```

Service detection performed. Please report any incorrect results at <https://nmap.org/submi>

Nmap done: 1 IP address (1 host up) scanned in 6.91 seconds

Как видно из вывода, утилита корректно определила наш сервер.

3.6 Сохранение вывода утилиты nmap в формате XML

Для сохранения вывода утилиты nmap в формате XML необходимо при запуске указать ключ '-oX':

```
nmap -oX - 169.254.120.103 > out.xml
```

В результате запуска, утилитой nmap будут просканированы все порты хоста с адресом 169.254.120.103, и вывод будет сохранен в формате XML. Содержимое файла out.xml:

```
<?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 07:42:02 2016 as: nmap -oX - 169.254.120.103 -->
<nmaprun scanner="nmap" args="nmap -oX - 169.254.120.103" start="1459683722" startstr="Sun
<scaninfo type="syn" protocol="tcp" numservices="1000" services="1,3-4,6-7,9,13,17,19-26,3
<verbose level="0"/>
<debugging level="0"/>
<host starttime="1459683722" endtime="1459683723"><status state="up" reason="arp-response"
<address addr="169.254.120.103" addrtype="ipv4"/>
<address addr="08:00:27:70:24:F3" addrtype="mac" vendor="Oracle VirtualBox virtual NIC"/>
<hostnames>
</hostnames>
<ports><extraports state="closed" count="977">
<extrareasons reason="resets" count="977"/>
</extraports>
<port protocol="tcp" portid="21"><state state="open" reason="syn-ack" reason_ttl="64"/><se
<port protocol="tcp" portid="22"><state state="open" reason="syn-ack" reason_ttl="64"/><se
<port protocol="tcp" portid="23"><state state="open" reason="syn-ack" reason_ttl="64"/><se
<port protocol="tcp" portid="25"><state state="open" reason="syn-ack" reason_ttl="64"/><se
<port protocol="tcp" portid="53"><state state="open" reason="syn-ack" reason_ttl="64"/><se
<port protocol="tcp" portid="80"><state state="open" reason="syn-ack" reason_ttl="64"/><se
<port protocol="tcp" portid="111"><state state="open" reason="syn-ack" reason_ttl="64"/><s
<port protocol="tcp" portid="139"><state state="open" reason="syn-ack" reason_ttl="64"/><s
<port protocol="tcp" portid="445"><state state="open" reason="syn-ack" reason_ttl="64"/><s
<port protocol="tcp" portid="512"><state state="open" reason="syn-ack" reason_ttl="64"/><s
<port protocol="tcp" portid="513"><state state="open" reason="syn-ack" reason_ttl="64"/><s
<port protocol="tcp" portid="514"><state state="open" reason="syn-ack" reason_ttl="64"/><s
<port protocol="tcp" portid="1099"><state state="open" reason="syn-ack" reason_ttl="64"/><
<port protocol="tcp" portid="1524"><state state="open" reason="syn-ack" reason_ttl="64"/><
<port protocol="tcp" portid="2049"><state state="open" reason="syn-ack" reason_ttl="64"/><
<port protocol="tcp" portid="2121"><state state="open" reason="syn-ack" reason_ttl="64"/><
<port protocol="tcp" portid="3306"><state state="open" reason="syn-ack" reason_ttl="64"/><
<port protocol="tcp" portid="5432"><state state="open" reason="syn-ack" reason_ttl="64"/><
<port protocol="tcp" portid="5900"><state state="open" reason="syn-ack" reason_ttl="64"/><
```

```

<port protocol="tcp" portid="6000"><state state="open" reason="syn-ack" reason_ttl="64"/>
<port protocol="tcp" portid="6667"><state state="open" reason="syn-ack" reason_ttl="64"/>
<port protocol="tcp" portid="8009"><state state="open" reason="syn-ack" reason_ttl="64"/>
<port protocol="tcp" portid="8180"><state state="open" reason="syn-ack" reason_ttl="64"/>
</ports>
<times srtt="284" rttvar="152" to="100000"/>
</host>
<runstats><finished time="1459683723" timestr="Sun Apr  3 07:42:03 2016" elapsed="0.27" su
</runstats>
</nmaprun>

```

3.7 Исследование работы утилиты nmap при помощи wireshark

Запустим утилиту Wireshark, затем просканируем целевую ВМ по адресу 169.254.120.103.

```
root@kali:~# nmap 169.254.120.103
```

Как показано на рисунке 1. Изначально nmap посылает на существующие порты TCP-пакеты с установленным флагом SYN, что означает установление соединения. Если при этом сканируемый порт отправляет ответ с установленными флагами [RST, ACK], значит соединение невозможно - порт закрыт.

Если после отправки TCP-пакета с флагом SYN сканируемый порт отправляет ответ также с установленным флагом SYN, это означает, что заданный порт открыт. Это также видно на рисунке 1.

5	4.2536449794	169.254.120.101	169.254.120.103	TCP	58 47249 → 53 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
6	4.253668886	169.254.120.101	169.254.120.103	TCP	58 47249 → 993 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
7	4.253689179	169.254.120.101	169.254.120.103	TCP	58 47249 → 139 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
8	4.253707772	169.254.120.101	169.254.120.103	TCP	58 47249 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
9	4.253725310	169.254.120.101	169.254.120.103	TCP	58 47249 → 1720 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
10	4.253751472	169.254.120.101	169.254.120.103	TCP	58 47249 → 256 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11	4.253822077	169.254.120.101	169.254.120.101	TCP	60 3306 → 47249 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
12	4.253920975	169.254.120.101	169.254.120.103	TCP	54 47249 → 3306 [RST] Seq=1 Win=0 Len=0
13	4.253939568	169.254.120.103	169.254.120.101	TCP	60 53 → 47249 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
14	4.253947174	169.254.120.101	169.254.120.103	TCP	54 47249 → 53 [RST] Seq=1 Win=0 Len=0
15	4.254311790	169.254.120.103	169.254.120.101	TCP	60 993 → 47249 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
16	4.254338660	169.254.120.103	169.254.120.101	TCP	60 139 → 47249 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
17	4.254346778	169.254.120.101	169.254.120.103	TCP	54 47249 → 139 [RST] Seq=1 Win=0 Len=0
18	4.254362199	169.254.120.103	169.254.120.101	TCP	60 443 → 47249 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
19	4.254366552	169.254.120.103	169.254.120.101	TCP	60 1720 → 47249 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
20	4.254371113	169.254.120.103	169.254.120.101	TCP	60 256 → 47249 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
21	4.254862103	169.254.120.103	169.254.120.101	TCP	60 113 → 47249 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
22	4.256228994	169.254.120.103	169.254.120.101	TCP	60 111 → 47249 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
23	4.256261811	169.254.120.101	169.254.120.103	TCP	54 47249 → 111 [RST] Seq=1 Win=0 Len=0
24	4.256280140	169.254.120.103	169.254.120.101	TCP	60 22 → 47249 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
25	4.256287451	169.254.120.101	169.254.120.103	TCP	54 47249 → 22 [RST] Seq=1 Win=0 Len=0
26	4.256300289	169.254.120.103	169.254.120.101	TCP	60 1025 → 47249 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
27	4.256305947	169.254.120.103	169.254.120.101	TCP	60 3389 → 47249 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
28	4.256310006	169.254.120.103	169.254.120.101	TCP	60 80 → 47249 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
29	4.257024622	169.254.120.101	169.254.120.103	TCP	58 47249 → 24800 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
30	4.257071430	169.254.120.101	169.254.120.103	TCP	58 47249 → 1434 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
31	4.257091188	169.254.120.101	169.254.120.103	TCP	58 47249 → 5101 [SYN] Seq=0 Win=1024 Len=0 MSS=1460

Рис. 1: Вывод утилиты Wireshark.

3.8 Просканировать виртуальную машину Metasploitable2 используя db_nmap из состава metasploit-framework

Перед началом сканирования необходимо включить postgresql и выполнить команду msfdb init для инициализации базы данных:

```

root@kali:~# service postgresql start
root@kali:~# msfdb init
Creating database user 'msf'
Enter password for new role:
Enter it again:
Creating databases 'msf' and 'msf_test'
Creating configuration file in /usr/share/metasploit-framework/config/database.yml
Creating initial database schema

```

После чего необходимо запустить msfconsole и можно использовать любую из команд, описанных выше, но вместо nmap можно использовать db_nmap. Результаты работы будут записаны в базу данных, тем самым обеспечивая экономию времени при сканировании портов.

```

root@kali:~# msfconsole

```

```

Metasploit Park, System Security Interface
Version 4.0.5, Alpha E
Ready...
> access security
access: PERMISSION DENIED.
> access security grid
access: PERMISSION DENIED.
> access main security grid
access: PERMISSION DENIED....and...
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!
YOU DIDN'T SAY THE MAGIC WORD!

```

Save 45% of your time on large engagements with Metasploit Pro
 Learn more on <http://rapid7.com/metasploit>

```

      =[ metasploit v4.11.7-                               ]
+ -- --=[ 1518 exploits - 877 auxiliary - 259 post           ]
+ -- --=[ 437 payloads - 38 encoders - 8 nops               ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

```

```

msf > db_nmap 169.254.120.103
[*] Nmap: Starting Nmap 7.01 ( https://nmap.org ) at 2016-04-04 00:12 MSK
[*] Nmap: 'mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled
[*] Nmap: Nmap scan report for 169.254.120.103
[*] Nmap: Host is up (0.0010s latency).
[*] Nmap: Not shown: 977 closed ports
[*] Nmap: PORT      STATE SERVICE
[*] Nmap: 21/tcp    open  ftp

```

```

[*] Nmap: 22/tcp open ssh
[*] Nmap: 23/tcp open telnet
[*] Nmap: 25/tcp open smtp
[*] Nmap: 53/tcp open domain
[*] Nmap: 80/tcp open http
[*] Nmap: 111/tcp open rpcbind
[*] Nmap: 139/tcp open netbios-ssn
[*] Nmap: 445/tcp open microsoft-ds
[*] Nmap: 512/tcp open exec
[*] Nmap: 513/tcp open login
[*] Nmap: 514/tcp open shell
[*] Nmap: 1099/tcp open rmiregistry
[*] Nmap: 1524/tcp open ingreslock
[*] Nmap: 2049/tcp open nfs
[*] Nmap: 2121/tcp open ccproxy-ftp
[*] Nmap: 3306/tcp open mysql
[*] Nmap: 5432/tcp open postgresql
[*] Nmap: 5900/tcp open vnc
[*] Nmap: 6000/tcp open X11
[*] Nmap: 6667/tcp open irc
[*] Nmap: 8009/tcp open ajp13
[*] Nmap: 8180/tcp open unknown
[*] Nmap: MAC Address: 08:00:27:70:24:F3 (Oracle VirtualBox virtual NIC)
[*] Nmap: Nmap done: 1 IP address (1 host up) scanned in 0.40 seconds

```

3.9 Описать работу пяти записей из файла nmap-service-probes

```

Probe UDP AndroMouse q|AMSNIFF|
rarity 9
ports 8888
match AndroMouse m|^GOTBACK$|s p/AndroMouse Android remote mouse server/

```

Директива Probe указывает на то, какое сообщение необходимо отправить для идентификации сервиса. В данном случае, сервис - AndroMouse, используемый протокол - UDP, отправляется следующая строка:

AMSNIFF

Строка с директивой rarity указывает частоту, с которой от сервиса можно ожидать возвращения корректных результатов. В данном случае - 9. Директива ports указывает на порты, используемые данным сервисом. Директива match необходима при распознавании сервиса на основе ответов на строку, отправленную предыдущей директивой Probe.

3.10 Описать работу скрипта из состава Nmap

Рассмотрим скрипт imap-capabilities. Данный скрипт получает информацию об IMAP mail сервере и выводит список поддерживаемых команд из RFC 3501. Пример использования:

```
root@kali:~# nmap -script=imap-capabilities imap.yandex.ru
```

```
Starting Nmap 7.01 ( https://nmap.org ) at 2016-04-04 01:27 MSK
```

```
Nmap scan report for imap.yandex.ru (93.158.134.124)
```

```
Host is up (0.0023s latency).
```

```
Other addresses for imap.yandex.ru (not scanned): 213.180.193.124 213.180.204.124 77.88.21
```

```
Not shown: 998 filtered ports
```

```
PORT      STATE SERVICE
```

```
143/tcp   open  imap
```

```
|_imap-capabilities: CHILDREN XLIST MOVEA0001 ID NAMESPACE IMAP4rev1 STARTTLS ENABLE LITER
```

```
993/tcp   open  imaps
```

```
|_imap-capabilities: UIDPLUS AUTH=PLAIN CHILDREN CAPABILITY XLIST MOVEA0001 OK ID NAMESPACE
```

```
Nmap done: 1 IP address (1 host up) scanned in 5.87 seconds
```

Код скрипта:

```
local imap = require "imap"
```

```
local shortport = require "shortport"
```

```
local stdnse = require "stdnse"
```

```
local table = require "table"
```

```
description = [[
```

```
Retrieves IMAP email server capabilities.
```

```
IMAP4rev1 capabilities are defined in RFC 3501. The CAPABILITY command  
allows a client to ask a server what commands it supports and possibly  
any site-specific policy.
```

```
]]
```

```
---
```

```
-- @output
```

```
-- 143/tcp open  imap
```

```
-- |_ imap-capabilities: LOGINDISABLED IDLE IMAP4 LITERAL+ STARTTLS NAMESPACE IMAP4rev1
```

```
author = "Brandon Enright"
```

```
license = "Same as Nmap--See https://nmap.org/book/man-legal.html"
```

```
categories = {"default", "safe"}
```

```
portrule = shortport.port_or_service({143, 993}, {"imap", "imaps"})
```

```
local function fail (err) return stdnse.format_output(false, err) end
```

```
action = function(host, port)
```

```
    local helper = imap.Helper:new(host, port)
```

```
    local status = helper:connect()
```

```
    if ( not(status) ) then return fail("Failed to connect to server") end
```

```

local status, capa = helper:capabilities(host, port)
if( not(status) ) then return fail("Failed to retrieve capabilities") end
helper:close()

if type(capa) == "table" then
    -- Convert the capabilities table into an array of strings.
    local capstrings = {}
    local cap, args
    for cap, args in pairs(capa) do
        table.insert(capstrings, cap)
    end
    return stdnse.strjoin(" ", capstrings)
elseif type(capa) == "string" then
    stdnse.debug1("'s' for %s", capa, host.ip)
    return
else
    return "server doesn't support CAPABILITIES"
end
end
end

```

В данном скрипте проверяются два стандартных порта для подключения к IMAP серверу - 143 и 993(ssl). Далее вызывается вспомогательная функция, которая запрашивает список поддерживаемых команд сервером, затем происходит перевод их в строку для вывода пользователю.

4 Выводы

В ходе данной лабораторной работы было произведено ознакомление с утилитой nmap. Был получен опыт ее применения для сканирования открытых портов и доступных хостов, а также для определения версий сервисов. Также были рассмотрены основные служебные файлы, которые используются для работы данной утилиты: файлы конфигураций и скрипты. Было рассмотрено расширение db_nmap, которое позволяет сохранять результаты сканирования в базу данных, тем самым увеличивая скорость работы утилиты.