

Отчет по лабораторной работе №4 :
Утилита для исследования сети и сканер
портов nmap

Дедков Сергей

2015

Содержание

1	Цель работы	2
2	Ход работы	2
2.1	Провести поиск активных хостов	2
2.2	Определить открытые порты	2
2.3	Определить версии сервисов	3
2.4	Изучить файлы nmap-services, nmap-os-db, nmapservice-probes	3
2.5	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)	6
2.6	Сохранить выводы утилиты в формате xml	8
2.7	Исследовать различные этапы и режимы работы nmap с использованием утилиты Wireshark	8
2.8	Просканировать виртуальную машину Metasploitable2 используя nmap_db из состава metasploit-framework	9
2.9	Выбрать пять записей из файла nmap-service-probes и описать их работу. Выбрать один скрипт из состава Nmap и описать его работу	11
3	Вывод	16

1 Цель работы

Определить набор и версии сервисов запущенных на компьютере в диапазоне адресов. Данная работа выполняется на ОС kali linux, используется утилита nmap.

2 Ход работы

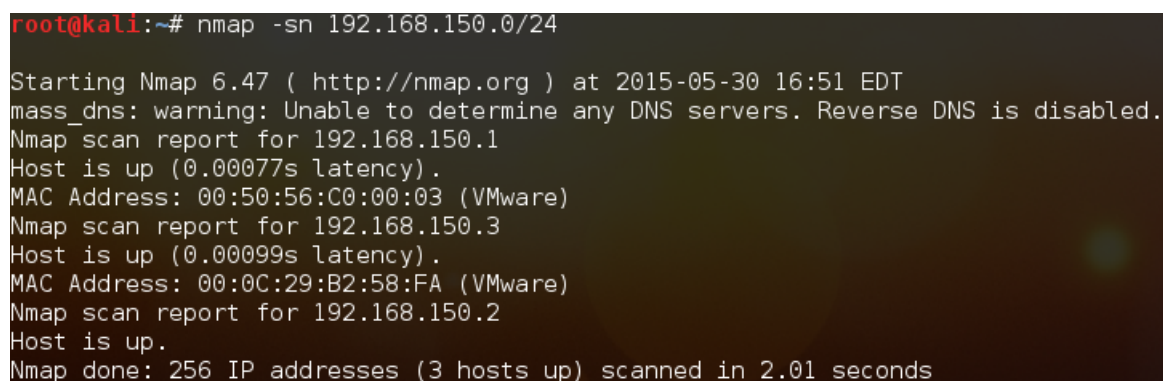
2.1 Провести поиск активных хостов

Настройки сети: в нашей сети имеется всего 3 хоста.

- Windows 8 (192.168.150.1), основная ОС
- kali linux (192.168.150.2)
- metasploitable2 (192.168.150.3)

Выведем список хостов в подсети 192.168.150.0/24

Для этого воспользуемся командой `nmap -sn 192.168.150.0/24`. (См. рисунок 1)



```
root@kali:~# nmap -sn 192.168.150.0/24

Starting Nmap 6.47 ( http://nmap.org ) at 2015-05-30 16:51 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
Nmap scan report for 192.168.150.1
Host is up (0.00077s latency).
MAC Address: 00:50:56:C0:00:03 (VMware)
Nmap scan report for 192.168.150.3
Host is up (0.00099s latency).
MAC Address: 00:0C:29:B2:58:FA (VMware)
Nmap scan report for 192.168.150.2
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 2.01 seconds
```

Рис. 1: Поиск хостов

2.2 Определить открытые порты

Просканируем порты metasploitable2.

Для определения открытых портов достаточно просто ввести `nmap 192.168.150.3` (сканируются порты до 1024). Или же воспользоваться опцией `-p`, например `nmap -p "*" 192.168.150.3`. Данной командой просканируются все порты, если необходимо задать диапазон достаточно указать его вместо `"*"`. Результат на рисунке 2.

```

root@kali:~# nmap -p "*" 192.168.150.3

Starting Nmap 6.47 ( http://nmap.org ) at 2015-05-30 17:07 E
mass_dns: warning: Unable to determine any DNS servers. Reve
Nmap scan report for 192.168.150.3
Host is up (0.00019s latency).
Not shown: 4219 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
3632/tcp  open  distccd
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: 00:0C:29:B2:58:FA (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.30 seconds

```

Рис. 2: Поиск портов

2.3 Определить версии сервисов

Чтобы определить версии сервисов необходимо воспользоваться командой nmap с ключем sV следующим образом: `nmap -sV 192.168.150.3`. Результат на рисунке 3.

2.4 Изучить файлы nmap-services, nmap-os-db, nmapservice-probes

Рассмотрим файл nmap-services. Для этого введем команду `vim /usr/share/nmap/nmap-services`.

Файл служит для быстрого поиска, напрмер с ключем -F. В файле в каждой строчке задаются сервисное название или сокращение, число

```

root@kali:~# nmap -p "*" -sV 192.168.150.3

Starting Nmap 6.47 ( http://nmap.org ) at 2015-05-30 17:11 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is
Nmap scan report for 192.168.150.3
Host is up (0.00024s latency).
Not shown: 4219 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  exec         netkit-rsh rexecd
513/tcp   open  login
514/tcp   open  tcpwrapped
1099/tcp  open  rmiregistry  GNU Classpath grmiregistry
1524/tcp  open  shell        Metasploitable root shell
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
3632/tcp  open  distccd      distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu
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: 00:0C:29:B2:58:FA (VMware)
Service Info: Hosts: metasploitable.localdomain, localhost, irc.Metasp

```

Рис. 3: Определение версий сервисов

порта и протокол, определенный разделом, частота порта мера того, как часто порт был найден открытым во время сканирования. Пример файла можно увидеть на рисунке 4.

Файл `nmap-os-db` содержит сотни примеров реакций ОС на `nmap`. Таким образом `nmap` определяет какая операционная система установлена на удаленной машине. Для того чтобы узнать какая ОС установлена нужно запустить `nmap` с ключем `-O`. Содержимое файла представлено на рисунке 5.

`nmap-service-probes` — это простой текстовый файл состоящий из строк, в котором хранятся тесты и сигнатуры подсистем определений версий. Строки, начинающиеся с символа "решетки"(`#`) воспринимаются как комментарии и игнорируются обработчиком. Пустые строки также не обрабатываются.

```
# Fields in this file are: Service name, portnum/protocol, open-frequency, optional comments
#
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
```

Рис. 4: Файл nmap-services

```
MatchPoints
SEQ(SP=25%GCD=75%ISR=25%TI=100%CI=50%II=100%SS=80%TS=100)
OPS(O1=20%O2=20%O3=20%O4=20%O5=20%O6=20)
WIN(W1=15%W2=15%W3=15%W4=15%W5=15%W6=15)
ECN(R=100%DF=20%T=15%TG=15%W=15%O=15%CC=100%Q=20)
T1(R=100%DF=20%T=15%TG=15%S=20%A=20%F=30%RD=20%Q=20)
T2(R=80%DF=20%T=15%TG=15%W=25%S=20%A=20%F=30%O=10%RD=20%Q=20)
T3(R=80%DF=20%T=15%TG=15%W=25%S=20%A=20%F=30%O=10%RD=20%Q=20)
T4(R=100%DF=20%T=15%TG=15%W=25%S=20%A=20%F=30%O=10%RD=20%Q=20)
T5(R=100%DF=20%T=15%TG=15%W=25%S=20%A=20%F=30%O=10%RD=20%Q=20)
T6(R=100%DF=20%T=15%TG=15%W=25%S=20%A=20%F=30%O=10%RD=20%Q=20)
T7(R=80%DF=20%T=15%TG=15%W=25%S=20%A=20%F=30%O=10%RD=20%Q=20)
U1(R=50%DF=20%T=15%TG=15%IPL=100%UN=100%RIPL=100%RID=100%RIPCK=100%RUO
IE(R=50%DFI=40%T=15%TG=15%CD=100)
```

Рис. 5: Файл nmap-os-db

Синтаксис:

- Probe <protocol> <probenamе> <probesendstring> - директива probe (тест) - указывает nmap, какие данные отправлять в процессе определения служб
- match <service> <pattern> <productname> <version> <device> <h?????> <info> <OS> - указывает nmap на то, как точно определить службу, используя полученный ответ на запрос, отправленный предыдущей директивой probe. 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

- `softmatch <service> <pattern> <productname> <version> <device> <h?????> <info> <OS>` - имеет аналогичный формат 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 шаблонной службе. Тестирование порта будет идти до тех пор, пока не будет найдено строгое соответствие (`match`) или не закончатся все тесты для данной службы
- `ports <portlist>` - группирует порты, которые обычно закрепляются за идентифицируемой данным тестом службой
- `sslports <sslportlist>` - аналогична директиве `ports`, только эта директива указывает порты, обычно используемые совместно с SSL
- `totalwaitms <milliseconds>` - редко используемая, т.к. указывает сколько времени (в миллисекундах) необходимо ждать ответ, прежде чем прекратить тест службы

2.5 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)

Напишем простой `tcp`-сервер, который просто ждет подключения клиента и отправляет ему сообщение. В файл `nmap-service-probes` добавим следующую строку:

```
match tcp-server m|^111| v/1.0.X/ p/Dedkov S.V./ i/It's works /
```

Код сервера:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using System.Net;
using System.Net.Sockets;
using System.Threading;

namespace ExampleTcpListener_Console
{
    class ExampleTcpListener
    {
        static void Main(string[] args)
```

```

{
    TcpListener server = null;
    try
    {
        int MaxThreadsCount = Environment.ProcessorCount * 4;
        Console.WriteLine(MaxThreadsCount.ToString());
        ThreadPool.SetMaxThreads(MaxThreadsCount, MaxThreadsCount);
        ThreadPool.SetMinThreads(2, 2);

        Int32 port = 9596;
        IPAddress localAddr = IPAddress.Parse("192.168.137.1");
        int counter = 0;
        server = new TcpListener(localAddr, port);

        server.Start();

        while (true)
        {

            Console.Write("\nWaiting for a connection... ");

            ThreadPool.QueueUserWorkItem(ObrabotkaZaprosa, server.AcceptTcpCl
            counter++;
            Console.Write("\nConnection №" + counter.ToString() + "!");

        }
    }
    catch (SocketException e)
    {
        Console.WriteLine("SocketException: {0}", e);
    }
    finally
    {
        server.Stop();
    }

    Console.WriteLine("\nHit enter to continue...");
    Console.Read();
}

static void ObrabotkaZaprosa(object client_obj)
{
    Byte[] bytes = new Byte[256];
    String data = null;

```

```

        TcpClient client = client_obj as TcpClient;

        data = null;

        NetworkStream stream = client.GetStream();

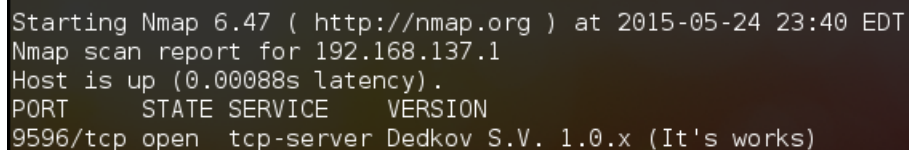
        int i;

        data = "111";
        byte[] msg = System.Text.Encoding.ASCII.GetBytes(data);
        stream.Write(msg, 0, msg.Length);

        client.Close();
    }
}

```

Таким образом теперь nmap знает, что если при пустом запросе с сервера прихот строка 111, значит нужно выводить информацию которая указана на рисунке 6.



```

Starting Nmap 6.47 ( http://nmap.org ) at 2015-05-24 23:40 EDT
Nmap scan report for 192.168.137.1
Host is up (0.00088s latency).
PORT      STATE SERVICE      VERSION
9596/tcp  open  tcp-server  Dedkov S.V. 1.0.x (It's works)

```

Рис. 6: Вывод информации о сервисе

2.6 Сохранить выводы утилиты в формате xml

Для того, чтобы вывести данные в xml файл достаточно вызвать команду nmap с ключем -oX и указать имя файла. Например:

```
nmap -sn -oX output.xml 192.168.150.1
```

Результат можно увидеть на рисунке 7:

2.7 Исследовать различные этапы и режимы работы nmap с использованием утилиты Wireshark

Wireshark — это достаточно известный инструмент для захвата и анализа сетевого трафика, фактически стандарт как для образования, так и для траблшутинга. Wireshark работает с подавляющим большинством


```

<?xml version="1.0"?>
<!DOCTYPE nmaprun>
<?xml-stylesheet href="file:///usr/bin/../share/nmap/nmap.xsl" type="text/xsl"?>
<!-- Nmap 6.47 scan initiated Sun May 24 23:49:40 2015 as: nmap -sn -oX output.xml 192.168.137.1 -->
<nmaprun scanner="nmap" args="nmap -sn -oX output.xml 192.168.137.1" start="1432525780" startstr="Sun May 24 23:49:40 2015" version="6.47" xmloutputversion="1.04">
<verbose level="0"/>
<debugging level="0"/>
<host><status state="up" reason="echo-reply" reason_ttl="128"/>
<address addr="192.168.137.1" addrtype="ipv4"/>
<hostnames>
</hostnames>
<times srtt="813" rttvar="5000" to="100000"/>
</host>
<runstats><finished time="1432525780" timestr="Sun May 24 23:49:40 2015" elapsed="0.01" summary="Nmap done at Sun May 24 23:49:40 2015; 1 IP address (1 host up) scanned in 0.01 seconds" exit="success"/><hosts up="1" down="0" total="1"/>
</runstats>
</nmaprun>

```

Рис. 7: output.xml

известных протоколов, имеет понятный и логичный графический интерфейс на основе GTK+ и мощнейшую систему фильтров. Кроссплатформенный, работает в таких ОС как Linux, Solaris, FreeBSD, NetBSD, OpenBSD, Mac OS X, и, естественно, Windows. Распространяется под лицензией GNU GPL v2. Доступен бесплатно на сайте wireshark.org.

Далее продемонстрируем простую работу с wireshark. При запуске wireshark предложит выбрать интерфейс и начать сканировать его трафик (см. рисунок 8). Выберем интерфейс eth0

Wireshark по-умолчанию выводит все пакеты, которые проходят через интерфейс eth0. Все пакеты просматривать неудобно, поэтому мы будем пользоваться фильтрами. Поставим фильтр по протоколу icmp, ip отправителя и получателя:

```
ip.src == 192.168.150.2 and ip.dst == 192.168.150.3 and icmp.
```

Введем команду для определения ОС в подсети 192.168.150.0/24

```
nmap -O 192.168.150.0/24.
```

Результат на рисунке 9. Увидим три пакета

По каждому запросу можно увидеть дополнительную информацию кликнув на нем, как, например, на рисунке 10.

2.8 Просканировать виртуальную машину Metasploitable2 используя nmap_db из состава metasploit-framework

Для работы с metasploit потребуется СУБД postgresql. Для того, чтобы ее запустить воспользуемся командой

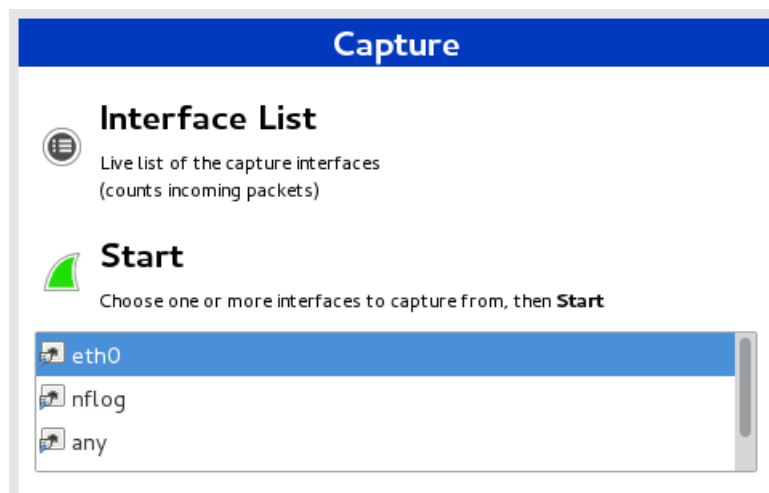


Рис. 8: wireshark выбор интерфейса и старт

Filter: : == 192.168.150.2 and ip.dst == 192.168.150.3 and Expression... Clear Apply Save						
No.	Time	Source	Destination	Protocol	Length	Info
3495	6.678055000	192.168.150.2	192.168.150.3	ICMP	162	Echo (ping) request
3499	6.703968000	192.168.150.2	192.168.150.3	ICMP	192	Echo (ping) request

Рис. 9: wireshark scan

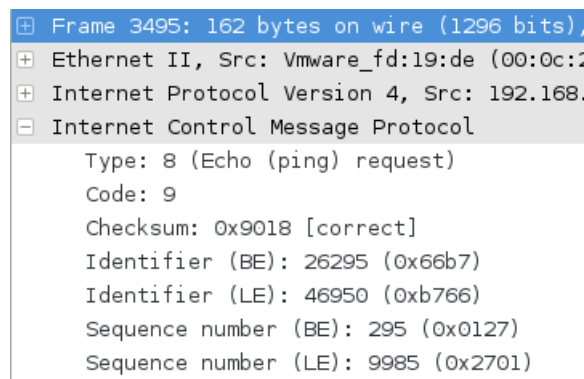


Рис. 10: wireshark info

`service postgresql start`

Затем запустим msf консоль командой `msfconsole`.

В данной консоли осуществляется работа с metasploit. Проверим соединение с БД:

`db_status`

Результат на рисунке 11.

```
msf > db_status
[*] postgresql connected to msf3
```

Рис. 11: db_status

Таблицы БД: hosts, services, vulns, loot и notes. В каждой храниться соответствующая информация. Для заполнения этих таблиц автоматизированно, можно использовать db_nmap. Так же можно использовать какую-то любую утилиту для сканирования, экспортировать результаты её работы в XML-файл, а потом - импортировать его в метасплloit. Это можно сделать, используя db_import внутри меню метасплloit.

Выполним команду: db_nmap -sV 192.168.150.0/24

Вывод будет такой же как и при использовании команды nmap. Теперь после того как была просканирована наша сеть можно посмотреть, что было записано в БД: hosts, services. Результаты на рисунках 12 и 13.

```
msf > hosts

Hosts
=====

address      mac                name  os_name  os_flavor  os_sp  purpose  info  comment
-----
192.168.150.1 00:50:56:c0:00:03   Unknown  Unknown  device
192.168.150.3 00:0c:29:b2:58:fa   Unknown  Unknown  device
```

Рис. 12: Таблица хостов

2.9 Выбрать пять записей из файла nmap-service-probes и описать их работу. Выбрать один скрипт из состава Nmap и описать его работу

- Первая запись

Возьмем самую первую запись probe - эта запись теста с отправкой null-запроса. В данной записи будет отправляться пустой запрос по протоколу TCP. С ожиданием ответа в 6 секунд(директива totalwaitms).

- Вторая запись

Второй записью рассмотрим match после probe null-запроса. Если пользователь укажет ключ -sV при использовании nmap и после отправки нулевого теста с сервера придет выражение подходящее под mSxf5xc6x1a тогда в колонке SERVICE при выводе информации он увидит наименование сервиса lc-server, а в колонке VERSION 1C:Enterprise business management server.

```
msf > services

Services
=====

host      port  proto  name      state  info
-----
192.168.150.1 135  tcp    msrpc     open   Microsoft Windows RPC
192.168.150.1 139  tcp    netbios-ssn open   netbios-ssn
192.168.150.1 445  tcp    netbios-ssn open   netbios-ssn
192.168.150.1 1028 tcp    msrpc     open   Microsoft Windows RPC
192.168.150.1 2869 tcp    http      open   Microsoft HTTPAPI httpd 2.0 SSDP/UPnP
192.168.150.3 512  tcp    exec      open   netkit-rsh rexecd
192.168.150.3 22   tcp    ssh       open   OpenSSH 4.7p1 Debian 8ubuntu1 protocol 2
192.168.150.3 23   tcp    telnet    open   Linux telnetd
192.168.150.3 25   tcp    smtp      open   Postfix smtpd
192.168.150.3 53   tcp    domain    open   ISC BIND 9.4.2
192.168.150.3 80   tcp    http      open   Apache httpd 2.2.8 (Ubuntu) DAV/2
192.168.150.3 111  tcp    rpcbind   open   2 RPC #100000
192.168.150.3 139  tcp    netbios-ssn open   Samba smbd 3.X workgroup: WORKGROUP
```

Рис. 13: Таблица сервисов

```
# This is the NULL probe that just compares any banners given to us
#####NEXT PROBE#####
Probe TCP NULL q||
# Wait for at least 6 seconds for data. It used to be 5, but some
# smtp services have lately been instituting an artificial pause (see
# FEATURE('greet_pause') in Sendmail, for example)
totalwaitms 6000
```

Рис. 14: Запись 1

```
match lc-server m|^S\x55\x56\x1a{| p/1C:Enterprise business management server/
```

Рис. 15: Запись 2

- Третья запись

В данной записи на сервер отправляется запрос по протоколу TCP в котором передается информация. Так же указан список портов и ssl-портов, по которым нужно осуществлять сканирование.

- Четвертая запись

Здесь отправляется запрос по протоколу UDP для проверки RPC.

- Пятая запись

Здесь отправляется запрос по протоколу UDP для проверки sql, через порт 1434.

```
#####NEXT PROBE#####
Probe TCP GetRequest q|GET / HTTP/1.0\r\n\r\n|
rarity 1
ports 1,70,79,80-85,88,113,139,143,280,497,505,514,515,540,554,591,620,631,783,888,898,
900,901,993,995,1026,1080,1042,1214,1220,1234,1311,1314,1344,1503,1610,1611,1830,1900,2
001,2002,2030,2064,2160,2306,2396,2525,2715,2869,3000,3002,3052,3128,3280,3372,3531,368
9,3872,4000,4444,4567,4660,4711,5000,5427,5060,5222,5269,5280,5432,5800-5803,5900,6103,
6346,6544,6600,6699,6969,7002,7007,7070,7100,7402,7776,8000-8010,8080-8085,8088,8118,81
81,8443,8880-8888,9000,9001,9030,9050,9080,9090,9999,10000,10001,10005,11371,13013,1366
6,13722,14534,15000,17988,18264,31337,40193,50000,55555
sslports 443,4443
```

Рис. 16: Запись 3

```
#####NEXT PROBE#####
Probe UDP RPCCheck q|\x72\xFE\x1D\x13\0\0\0\0\0\0\0\0\x02\0\x01\x86\xA0\0\x01\x97\x7C\0\0\
rarity 1
ports 17,88,111,407,500,517,518,1419,2427,4045,10000,10080,12203,27960,32750-32810,38978
```

Рис. 17: Запись 4

```
#####NE
Probe UDP Sqlping q|\x02|
rarity 6
ports 1434
```

Рис. 18: Запись 5

- Скрипт

Рассмотрим следующий скрипт `skypev2-version.nse`. Его можно найти в папке с отчетом.

Первым делом в нем объявлены переменные импортированные из библиотек:

```
local comm = require "comm"
local nmap = require "nmap"
local shortport = require "shortport"
local string = require "string"
local U = require "lpeg-utility"
```

Далее следует описание скрипта в переменной `description`:

```
description = [[
Detects the Skype version 2 service.
]]
```

Далее описание в комментарии в формате NSEDoc.

```
---
-- @output
-- PORT    STATE SERVICE VERSION
-- 80/tcp  open  skype2  Skype
```

Далее имя автора, лицензия, категория:

```
author = "Brandon Enright"
license = "Same as Nmap--See http://nmap.org/book/man-legal.html"
categories = {"version"}
```

Далее правило порта(вместо него можно задавать правило хоста)
- функция возвращающая true или false. Если возвращает true, то выполняется функция заданная в переменной action. В данном случае из кода понятно, в каких случаях выполняется это условие.

```
portrule = function(host, port)
  return (port.number == 80 or port.number == 443 or
    port.service == nil or port.service == "" or
    port.service == "unknown") -- условия по портам
  and port.protocol == "tcp" and port.state == "open" -- условия по протоколу
  and port.version.name_confidence < 10 -- доверие
  and not(shortport.port_is_excluded(port.number,port.protocol)) -- порт не исключен
  and nmap.version_intensity() >= 7 -- версия интенсивности
end
```

Далее приведен код функции action:

```
action = function(host, port)
  local result, rand
  -- Did the service engine already do the hard work?
  if port.version and port.version.service_fp then
    -- Probes sent, replies received, but no match.
    result = U.get_response(port.version.service_fp, "GetRequest")
    -- Loop through the ASCII probes most likely to receive random response
    -- from Skype. Others will also receive this response, but are harder to
    -- distinguish from an echo service.
    for _, p in ipairs({"HTTPOptions", "RTSPRequest"}) do
      rand = U.get_response(port.version.service_fp, p)
      if rand then
        break
      end
    end
  end
  return result
end
```

```

        end
    end
end
local status
if not result then
    -- Have to send the probe ourselves.
    status, result = comm.exchange(host, port,
        "GET / HTTP/1.0\r\n\r\n", {bytes=26})

    if (not status) then
        return nil
    end
end

if (result ~= "HTTP/1.0 404 Not Found\r\n\r\n") then
    return
end

-- So far so good, now see if we get random data for another request
if not rand then
    status, rand = comm.exchange(host, port,
        "random data\r\n\r\n", {bytes=15})

    if (not status) then
        return
    end
end

if string.match(rand, "[^%s!-~].*[^%s!-~].*[^%s!-~]") then
    -- Detected
    port.version.name = "skype2"
    port.version.product = "Skype"
    nmap.set_port_version(host, port)
    return
end
return
end
end

```

После прохождения всех проверок, обозначается версия и продукт (после комментария Detected).

Для использования скрипта нужно ввести либо ключ -sC, либо – script=<имя_скрипта>.

Для примера была просканирована основная ОС. Двумя способа-

ми. Результаты можно увидеть на рисунках 19 и 20.

```
root@kali:/usr/share/nmap/scripts# nmap 192.168.150.1 -p-

Starting Nmap 6.47 ( http://nmap.org ) at 2015-05-31 01:35 EDT
Nmap scan report for 192.168.150.1
Host is up (0.00052s latency).
Not shown: 65515 closed ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
902/tcp    open  iss-realsure
912/tcp    open  apex-mesh
1025/tcp   open  NFS-or-IIS
1026/tcp   open  LSA-or-nterm
1027/tcp   open  IIS
1028/tcp   open  unknown
1031/tcp   open  iad2
1074/tcp   open  warmspotMgmt
1203/tcp   open  unknown
1433/tcp   open  ms-sql-s
2383/tcp   open  ms-olap4
2869/tcp   open  icslap
3050/tcp   open  gds_db
5357/tcp   open  wsdapi
27017/tcp  open  unknown
41535/tcp  open  unknown
```

Рис. 19: Результаты сканирования без использования скрипта

3 Вывод

В ходе проделанной работы были изучены основы работы с nmap, немного изучен сниффер wireshark, а так же запись в БД metasploit посредством команды db_nmap. Ранее я не был знаком с подобными программами, поэтому не с чем сравнивать. В общем интересно было посканировать хосты и порты. Понравилось так же что можно писать собственные скрипты, но для того, чтобы подробно с ними разобраться нужно больше времени.


```
root@kali:/usr/share/nmap/scripts# nmap 192.168.150.1 -p- --script=skypev2-version.nse

Starting Nmap 6.47 ( http://nmap.org ) at 2015-05-31 02:03 EDT
Nmap scan report for 192.168.150.1
Host is up (0.00075s latency).
Not shown: 65515 closed ports
PORT      STATE SERVICE
80/tcp    open  http
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
902/tcp   open  iss-realsecure
912/tcp   open  apex-mesh
1025/tcp  open  NFS-or-IIS
1026/tcp  open  LSA-or-nterm
1027/tcp  open  IIS
1028/tcp  open  unknown
1031/tcp  open  iad2
1074/tcp  open  warmspotMgmt
1203/tcp  open  unknown
1433/tcp  open  ms-sql-s
2383/tcp  open  ms-olap4
2869/tcp  open  icslap
3050/tcp  open  gds_db
5357/tcp  open  wsdapi
27017/tcp open  unknown
41535/tcp open  skype2
```

Рис. 20: Результаты сканирования с использованием скрипта