Testing for vulnerabilities

Exploitation techniques

- Step 1: information gathering
- Step 2: services enumeration
- Step 3: Exploitation
- Step 4: Persistence
- Step 5: Post-attack enumeration
- Step 6: Cleanup

Exploitation

- "Exploit" – sequence of instructions or data that takes advantage of a vulnerability to achieve inappropriate behavior
  - Example: Remote Code Execution (RCE)
  - Example: Local Privilege Escalation (LPE)
  - Example: Information Disclosure (ID)
- Exploit != payload
- The reason why we do pentesting
  - The presence of vulnerability is generally sufficient to act
  - If the vulnerability can also be exploited, so much the better
- The step that (probably) gives control over the vulnerable system or leads to the compromise of the organization

Exploitation

- Vulnerabilities must already be identified
  - Info gathering and service enumeration
- In this phase, we need to know exactly what exploit to use
  - We just have to choose the right payload, depending on the exploit and according to what we want to achieve, depending on the operating system and hardware
- In general, we apply the "lowest hanging fruit" principle – we exploit the most critical vulnerabilities
  - We aim to gain control over the vulnerable system
    - Not necessarily with an RCE-type vulnerability!
  - Once a system is compromised, we can do lateral movement

Exploitation

- To exploit a vulnerability, we need an exploit
- Sources of exploits:
  - Metasploit
  - exploit-db
  - packetstormsecurity
  - Own development/manual operation
- Most of the time, we will find exploits in Metasploit or on exploit-db
  - But not always
- Depending on the client's preferences, custom exploits can be developed
  - … requires skills & time

TYPES OF VULNERABILITIES

- Types of exploits == types of vulnerabilities
- Every vulnerability is exploited in a specific way
- Even identical vulnerabilities can be exploited differently, depending on the application, specific conditions, etc.
  - Example: use-after-free in Internet Explorer vs. use-after-free in an image viewer
- An exploit also depends a lot on the mitigations on the target system
  - ASLR bypass, DEP/NX bypass, CFG bypass, etc.

TYPES OF VULNERABILITIES

- Binary:
  - Memory corruption, Race-conditions, Input not validated properly, etc.
- Web:
  - XSS, CSRF, LFI, RFI, RCE, SQLi, etc.
- Generic:
  - Weak credentials/default, ID, arbitrary directory traversals, etc.
- Result:
  - RCE – remote code execution
  - NON-RCE – does not offer remote code execution

TYPES OF PAYLOADS

- In general, we speak of "payload" in the case of RCE
- The payload represents the code that will be executed because of the exploitation
  - Generally, this is a "shellcode"
- Sometimes, "staged" shellcodes are used – several stages
  - If the shellcode is very large
- Sometimes, a "drive-by-download" attack is used – the shellcode downloads and executes a binary

TYPES OF PAYLOADS

- Metasploit contains a huge list of payloads
- Each payload lends itself to a certain type of exploit, operating system, and processor architecture (in the case of binary payloads)
- Depending on the OS, metasploit offers payloads for:
  - Windows x86/x64
  - OSX
  - Solaris
  - Linux
- There are also agnostic payloads:
  - Python
  - PHP
  - Java
  - Perl

TYPES OF PAYLOADS

- Binary Payloads
  - Used to exploit binary vulnerabilities
  - Windows, Linux, OSX, Android, etc.
  - X86, x64, MIPS, ARM, SPARC, etc.
  - Choose the payload carefully according to the OS and CPU!
    - On a router, you'll probably use a MIPS payload for Linux
    - On a smartphone, you'll probably use an ARM payload for Android
  - Choose the payload according to what you want to do

TYPES OF PAYLOADS

- Generic Payloads/Scripts/Commands:
  - Especially useful in the Web sphere
  - PHP, Ruby, Perl, Python, etc.
  - They must be chosen according to the scripting language present on the server

TYPES OF PAYLOADS

- The main criterion for choosing the payload is the desired effect
- Metasploit offers a lot of different types
- If there isn't what you need in Metasploit – write by hand
- The most common types of payloads:
  - Bind – listens for connections on a predefined port
  - Reverse – connects to a predefined host:port
  - Exec – launches an application into execution
  - Download & exec – download file from predefined location + Exec
  - Loadlibrary – loads a DLL
  - Adduser – add a new user

TYPES OF PAYLOADS

- Some payloads have different variations:
  - Bind: bind_hidden_ipknock_tcp, bind_tcp_rc4, bind_tcp_uuid, etc.
  - Reverse: reverse_tcp_allports, reverse_tcp_dns, reverse_tcp_rc4, etc.
- Some payloads can be delivered in different forms:
  - Meterpreter
  - Dllinject
  - Staged
  - VNC
- Sometimes, we want to write a custom payload, for different reasons
  - It doesn't exist in Metasploit, we want to avoid detections, etc.

ENCODING PAYLOADS

- Depending on the vulnerability and exploit, payloads must be coded to:
  - Avoid unwanted characters
  - Avoid detecting some antiviruses
- Encoding applies to binary payloads, and usually involves encrypting them
- There are encoders for: x86/x64, MIPS, Sparc, PPC

Exploitation techniques - RCE

- Once a vulnerable service is discovered, we exploit it
- We need to determine what exploit and what payload we will use
- The payload type depends on:
  - What we want to achieve
  - The system concerned
- Sometimes, we want to avoid antivirus detections or firewall alerts
  - Therefore, we will adjust the type of payload used
- In principle, any type of payload can help us achieve our goal
  - … whatever it is
- We will mainly consider RCE-type vulnerabilities

Exploitation techniques - RCE

- Bind payload
  - Start a listener on a predefined port
  - We will be able to connect to that port and get a shell
  - Disadvantage: firewall, filtered port, port already used, etc.

```
msf payload(meterpreter_reverse_tcp) > use payload/windows/shell_bind_tcp
msf payload(shell_bind_tcp) > show options

Module options (payload/windows/shell_bind_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LPORT     4444             yes       The listen port
   RHOST                      no        The target address

msf payload(shell_bind_tcp) >
```

Exploitation techniques - RCE

- Reverse payload:
  - It will connect to a predefined host:port and provide a shell
  - Disadvantage: firewall

```
msf payload(shell_reverse_tcp) > show options

Module options (payload/windows/shell_reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  none             yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     127.0.0.1        yes       The listen address
   LPORT     80               yes       The listen port

msf payload(shell_reverse_tcp) > _
```

Exploitation techniques - RCE

- Exec payload:
  - It allows us to execute commands/applications
  - We can add users, modify firewall rules, etc.
- Download & exec payload:
  - Much greater control compared to exec, but more intrusive
  - Downloads an app from a predefined link, and runs it
  - Payload typically used by malware

Exploitation techniques - RCE

- Sometimes, we use a bind/reverse payload that doesn't seem to work
  - Because of some firewall rules, for example
- How do we find out if the exploit really works?
  - Example: We use an exec payload that pings our machine
- In general, there are ports accessible from any system
  - HTTP, HTTPS, FTP, etc.
  - It is recommended that the reverse payload connect to such a port

Exploitation techniques - RCE

- If the system has an active RDP/SSH service, we can use an adduser payload and then connect to RDP/SSH
  - If the RDP/SSH service is not active, we can activate it
    - If we have root/admin rights
- If the system has a web server, we can add a PHP backoor that interprets commands
  - The file will be located in the directory from which the web pages are served
  - By accessing it, we can send commands to the system
- If the system has an FTP server, we can add a new user with access to the entire FS
  - Sometimes it is not possible, if we do not have full rights and the server is running as root

Exploitation techniques - RCE

- In the case of an RCE present in web services, there are several alternatives
  - There are many methods of exploiting web RCEs
  - It depends on which engine is used by the server
  - Depends on the vulnerability
- Example: Local File Inclusion (LFI)
  - If we can force data to be written to disk in a known location, it is trivial
  - Usually, we can inject commands/scripts into logs (e.g. Apache log)
  - Sensitive files can be accessed (passwd, shadow, SAM hives)
- Example: Remote File Inclusion (RFI)
  - It's trivial to exploit: we include a source file from us on the server

Exploitation techniques - RCE

- Example: Weak/default credentials in a web application
  - Maybe you can modify existing source files, upload plugins, etc.
  - Can gain access to the system directly as root/as a less privileged user
- Example: SQLi
  - xp_cmd_shell

Exploitation techniques – Non RCE

- NON-RCE vulnerabilities; we can consider all vulnerabilities that do not confer arbitrary code execution directly as non-rce:
  - XSS, CSRF, SQLi, ID, weak/default credentials, etc.
  - A large part of the vulnerabilities are not represented by the RCE
- If we can't execute code on the victim, it's more difficult to compromise it
  - … Is it?
- It all depends on the vulnerability and skills of the attacker + what we want to achieve
- Sometimes, however, they can be exploited to compromise the victim
- Even if they cannot lead to the victim's compromise, they can still be useful

Exploitation techniques – Non RCE

- Example: SQLi:
  - Retrieved information from the database
  - Add records to the database
- Example: XSS
  - We get cookie/sessionid from victims
- Example: CSRF
  - We can change credentials, add new users, etc.
- Example: Directory traversal
  - Obtain passwd/shadow, SAM, confidential files

METERPRETER

- Payload that provides a complete working environment on the compromised machine
- Meterpreter exists in multiple forms:
  - Binary: x86/x64
  - Script: PHP, Python
- On different operating systems:
  - Android, Linux, Windows
- And using different techniques:
  - reverse_http, reverse_tcp, bind_tcp

METERPRETER

- The main disadvantage:
  - It is detected by antivirus/IDS/IPS/etc.
- The main advantage:
  - Lots of useful commands and ease of working with it
  - Compatibility with most exploits

METERPRETER

- Additional Metasploit modules can be run
  - run Module
- A shell can be created
  - shell
- Can run common commands
  - ps, ls, cat, pwd, cwd, cd, etc.
- NT/LM hashes can be dumped
  - hashdump
- Sometimes, privilege escalation can be done
  - getsystem

METERPRETER

- Screen-shots
  - screengrab
- Migration to other processes
  - migrated
- Turning on webcams:
  - webcam_snap
- File upload/download

LATERAL MOVEMENT

- Lateral movement = compromising other systems in the network
- Most of the time, an externally exposed system (internet) has multiple network interfaces
- Once a system is compromised, there is the possibility of using it as a pivot for lateral movement
- Once such a system is compromised, we can start compromising other systems in that network
  - The current system becomes pivotal
- If access has been gained inside the organization, everything becomes much simpler

CONCLUSIONS

- Exploiting vulnerabilities is the main step in the pentesting procedure
- In principle, we can consider two broad categories of vulnerabilities:
  - RCE – those that offer Remote Code Execution
  - NON-RCE – those that do not offer Remote Code Execution
- Both have compromising potential
- The mining technique is a function of many variables:
  - Vulnerability
  - Operating System
  - CPU
  - Security mitigations present
  - Purpose