

Communication and Network Security

exercises

Henrik Kramselund Jereminsen
hlk@zencurity.com

April 10, 2020



Contents

1 Download Kali Linux Revealed (KLR) Book 10 min	3
2 Check your Kali VM, run Kali Linux 30 min	4
3 Bonus: Check your Debian VM 10 min	5
4 Wireshark and Tcpdump 15 min	6
5 Capturing TCP Session packets 10 min	8
6 Whois databases 15 min	10
7 Using ping and traceroute 10 min	11
8 DNS and Name Lookups 10 min	13
9 Nping check ports 10 min	14
10 Try pcap-diff 15 min	16
11 Discover active systems ping sweep 10 min	18
12 Execute nmap TCP and UDP port scan 20 min	19
13 Perform nmap OS detection 10 min	20
14 EtherApe 10 min	21
15 ARP spoofing and ettercap 20	22

CONTENTS

16 Perform nmap service scan 10 min	23
17 Nmap full scan - strategy 15 min	24
18 Reporting HTML 15 min	26
19 SSL/TLS scanners 15 min	28
20 sslstrip 15 min	29
21 mitmproxy 30 min	30
22 sslsplit 10 min	31
23 Wardriving Up to 60min	32
24 Aircrack-ng 30 min	33
25 SNMP walk 15min	34
26 Try Hydra brute force 30min	35
27 Zeek on the web 10min	36
28 Zeek DNS capturing domain names 10min	37
29 Zeek TLS capturing certificates 10min	39
30 Suricata Basic Operation 15min	40
31 Basic Suricata rule configuration 15min	42
32 Configure Mirror Port 10min	44
33 Save Suricata JSON Output in Database 30min	45

CONTENTS

34 Suricata Netflow 10min	47
35 Extending Zeek and Suricata 10min	48
36 Indicators of Compromise 15min	49
37 Bonus: VXLAN Detection	50
38 Logning med syslogd og syslog.conf 10min	51
39 Create Kibana Dashboard 15min	52
40 Fun with SSH honeypots 30min	53
41 Integrating Zeek IDS with the Elastic Stack 30min	54
42 Test an e-mail server 10min	55
43 VLANs, Routing and RPF - 2h	56
44 Monitoring - setup LibreNMS - 30 min	62
45 IDS with Zeek and Suricata - 30min	64
46 Configure port security - 30 min	66

Preface

This material is prepared for use in *Communication and Network Security workshop* and was prepared by Henrik Lund Kramshoej, <http://www.zencurity.com> . It describes the networking setup and applications for trainings and workshops where hands-on exercises are needed.

Further a presentation is used which is available as PDF from kramse@Github
Look for communication-and-network-security-exercises in the repo security-courses.

These exercises are expected to be performed in a training setting with network connected systems. The exercises use a number of tools which can be copied and reused after training. A lot is described about setting up your workstation in the repo

<https://github.com/kramse/kramse-labs>

Prerequisites

This material expect that participants have a working knowledge of TCP/IP from a user perspective. Basic concepts such as web site addresses and email should be known as well as IP-addresses and common protocols like DHCP.

Have fun and learn

Introduction to networking

IP - Internet protocol suite

It is extremely important to have a working knowledge about IP to implement secure and robust infrastructures. Knowing about the alternatives while doing implementation will allow the selection of the best features.

ISO/OSI reference model

A very famous model used for describing networking is the ISO/OSI model of networking which describes layering of network protocols in stacks.

This model divides the problem of communicating into layers which can then solve the problem as smaller individual problems and the solution later combined to provide networking.

Having layering has proven also in real life to be helpful, for instance replacing older hardware technologies with new and more efficient technologies without changing the upper layers.

In the picture the OSI reference model is shown along side with the Internet Protocol suite model which can also be considered to have different layers.

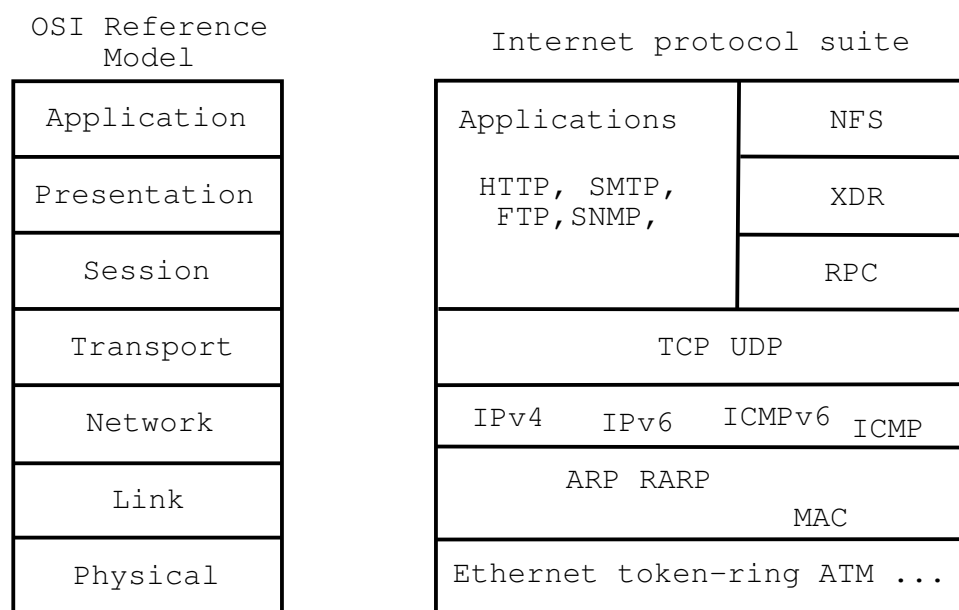


Figure 1: OSI og Internet Protocol suite

Exercise content

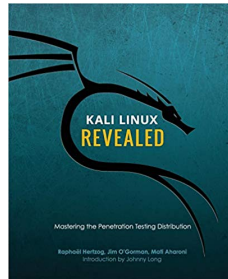
Most exercises follow the same procedure and has the following content:

- **Objective:** What is the exercise about, the objective
- **Purpose:** What is to be the expected outcome and goal of doing this exercise
- **Suggested method:** suggest a way to get started
- **Hints:** one or more hints and tips or even description how to do the actual exercises
- **Solution:** one possible solution is specified
- **Discussion:** Further things to note about the exercises, things to remember and discuss

Please note that the method and contents are similar to real life scenarios and does not detail every step of doing the exercises. Entering commands directly from a book only teaches typing, while the exercises are designed to help you become able to learn and actually research solutions.

Exercise 1

Download Kali Linux Revealed (KLR) Book 10 min



Kali Linux Revealed Mastering the Penetration Testing Distribution

Objective:

We need a Kali Linux for running tools during the course. This is open source, and the developers have released a whole book about running Kali Linux.

This is named Kali Linux Revealed (KLR)

Purpose:

We need to install Kali Linux in a few moments, so better have the instructions ready.

Suggested method:

Create folders for educational materials. Go to <https://www.kali.org/download-kali-linux-revealed-book/> Read and follow the instructions for downloading the book.

Solution:

When you have a directory structure for download for this course, and the book KLR in PDF you are done.

Discussion:

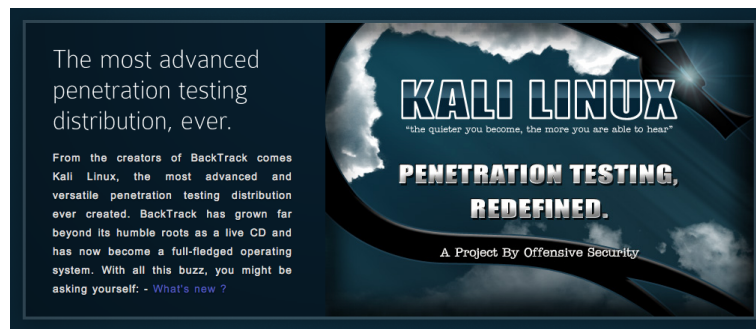
Linux is free and everywhere. The tools we will run in this course are made for Unix, so they run great on Linux.

Kali Linux is a free pentesting platform, and probably worth more than \$10.000

The book KLR is free, but you can buy/donate, and I recommend it.

Exercise 2

Check your Kali VM, run Kali Linux 30 min



Objective:

Make sure your virtual machine is in working order.

We need a Kali Linux for running tools during the course.

Purpose:

If your VM is not installed and updated we will run into trouble later.

Suggested method:

Go to <https://github.com/kramse/kramse-labs/>

Read the instructions for the setup of a Kali VM.

Hints:

If you allocate enough memory and disk you wont have problems.

Solution:

When you have a updated virtualisation software and Kali Linux, then we are good.

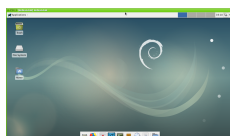
Discussion:

Linux is free and everywhere. The tools we will run in this course are made for Unix, so they run great on Linux.

Kali Linux includes many hacker tools and should be known by anyone working in infosec.

Exercise 3

Bonus: Check your Debian VM 10 min



Objective:

Make sure your virtual Debian machine is in working order.

We need a Debian Linux for running a few extra tools during the course.

This is a bonus exercise - one is needed per team that want to try these tools. Tools which need Debian are Zeek and Suricata.

Purpose:

If your VM is not installed and updated we will run into trouble later.

Suggested method:

Go to <https://github.com/kramse/kramse-labs/>

Read the instructions for the setup of a Kali VM.

Hints:

Solution:

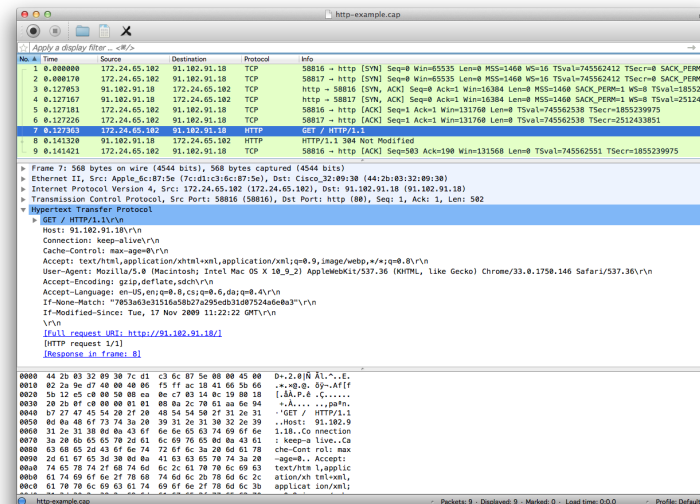
When you have a updated virtualisation software and Kali Linux, then we are good.

Discussion:

Linux is free and everywhere. The tools we will run in this course are made for Unix, so they run great on Linux.

Exercise 4

Wireshark and Tcpcdump 15 min



Objective:

Try the program Wireshark locally your workstation, or tcpcdump

You can run Wireshark on your host too, if you want.

Purpose:

Installing Wireshark will allow you to analyse packets and protocols

Tcpcdump is a feature included in many operating systems and devices to allow packet capture and saving network traffic into files.

Suggested method:

Run Wireshark or tcpcdump from your Kali Linux

The PPA book page 41 describes Your First Packet Capture.

Hints:

PCAP is a packet capture library allowing you to read packets from the network. Tcpcdump uses libpcap library to read packet from the network cards and save them. Wireshark is a graphical application to allow you to browse through traffic, packets and protocols.

Both tools are already on your Kali Linux, or do: `apt-get install tcpcdump wireshark`

Solution:

When Wireshark is installed sniff some packets. We will be working with both live traffic and saved packets from files in this course.

If you want to capture packets as a non-root user on Debian, then use the command to add a Wireshark group:

```
sudo dpkg-reconfigure wireshark-common
```

and add your user to this:

```
sudo gpasswd -a $USER wireshark
```

Dont forget to logout/login to pick up this new group.

Discussion:

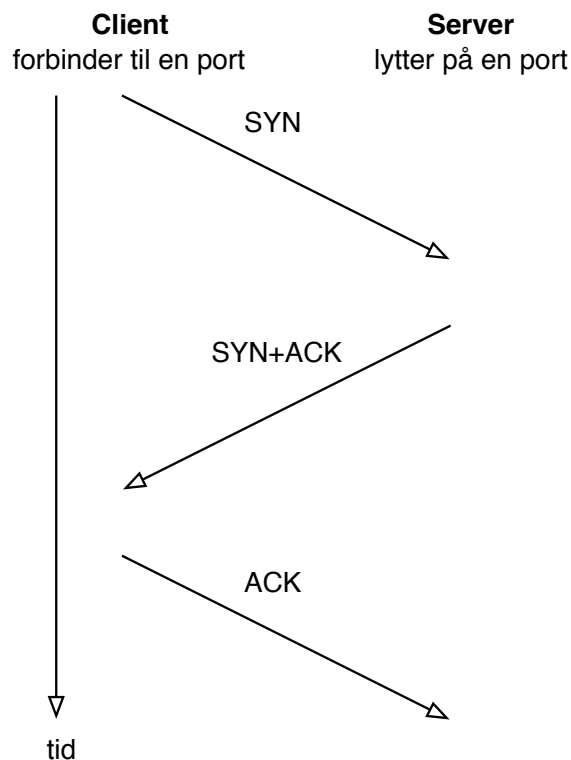
Wireshark is just an example other packet analyzers exist, some commercial and some open source like Wireshark

We can download a lot of packet traces from around the internet, we might use examples from

<https://www.bro.org/community/traces.html>

Exercise 5

Capturing TCP Session packets 10 min



Objective:

Sniff TCP packets and dissect them using Wireshark

Purpose:

See real network traffic, also know that a lot of information is available and not encrypted.

Note the three way handshake between hosts running TCP. You can either use a browser or command line tools like cURL while capturing

```
curl http://www.zencurity.com
```

Suggested method:

Open Wireshark and start a capture

Then in another window execute the ping program while sniffing

or perform a Telnet connection while capturing data

Hints:

When running on Linux the network cards are usually named `eth0` for the first Ethernet and `wlan0` for the first Wireless network card. In Windows the names of the network cards are long and if you cannot see which cards to use then try them one by one.

Solution:

When you have collected some TCP sessions you are done.

Discussion: Is it ethical to collect packets from an open wireless network?

Also note the TTL values in packets from different operating systems

Exercise 6

Whois databases 15 min

Objective:

Learn to lookup data in the global Whois databases

Purpose:

We often need to see where traffic is coming from, or who is responsible for the IP addresses sending attacks.

Suggested method:

Use a built-in command line, like: `host www.zencurity.dk` to look up an IP address and then `whois` with the IP address.

Hints:

Another option is to use web sites for doing Whois lookups <https://apps.db.ripe.net/db-web-ui/#/query> or their RIPEStat web site which can give even more information <https://stat.ripe.net/>

Solution:

When you can find our external address and look it up, you are done.

Discussion:

Whois databases are global and used for multiple purposes, the ones run by the Regional Internet Registries ARIN, RIPE, AfriNIC, LACNIC og APNIC have information about IP addresses and AS numbers allocated.

Exercise 7

Using ping and traceroute 10 min

Objective:

Be able to do initial debugging of network problems using commands ping and traceroute

Purpose:

Being able to verify connectivity is a basic skill.

Suggested method:

Use ping and traceroute to test your network connection - can be done on Windows and UNIX.

Hints:

```
$ ping 10.0.42.1
PING 10.0.42.1 (10.0.42.1) 56(84) bytes of data.
64 bytes from 10.0.42.1: icmp_seq=1 ttl=62 time=1.02 ms
64 bytes from 10.0.42.1: icmp_seq=2 ttl=62 time=0.998 ms
^C
--- 10.0.42.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.998/1.012/1.027/0.034 ms
```

Don't forget that UNIX ping continues by default, press ctrl-c to break.

Do the same with traceroute.

Solution:

Run both programs to local gateway and some internet address by your own choice.

Discussion:

Note the tool is called tracert on Windows, shortened for some reason.

ICMP is the Internet Control Message Protocol, usually used for errors like host unreachable. The ECHO request ICMP message is the only ICMP message that generates another.

The traceroute programs send packets with low Time To Live (TTL) and receives ICMP messages, unless there is a problem or a firewall/filter. Also used for mapping networks.

Bonus:

Whats the difference between:

- **traceroute** and **traceroute -I**
- NB: traceroute -I is found on UNIX - traceroute using ICMP pakket
- Windows tracert by default uses ICMP
- Unix by default uses UDP, but can use ICMP instead.
- Lots of traceroute-like programs exist for tracing with TCP or other protocols

Exercise 8

DNS and Name Lookups 10 min

Objective:

Be able to do DNS lookups from specific DNS server

Purpose:

Try doing DNS lookup using different programs

Suggested method:

Try the following programs:

- nslookup - UNIX and Windows, but not recommended
`nslookup -q=txt -class=CHAOS version.bind. 0`
- dig - syntax `@server domain query-type query-class`
`dig @8.8.8.8 www.example.com`
- host - syntaks `host [-l] [-v] [-w] [-r] [-d] [-t querytype] [-a] host [server]`
`host www.example.com 8.8.8.8`

Hints:

Dig is the one used by most DNS admins, I often prefer the host command for the short output.

Solution:

Shown inline, above.

Discussion:

The nslookup program does not use the same method for lookup as the standard lookup libraries, results may differ from what applications see.

What is a zone transfer, can you get one using the host command?

Explain forward and reverse DNS lookup.

Exercise 9

Nping check ports 10 min

Objective:

Show the use of Nping tool for checking ports through a network

Purpose:

Nping can check if probes can reach through a network, reporting success of failure.
Allows very specific packets to be sent.

Suggested method:

Run the command using a common port like Web HTTP:

```
root@KaliVM:~# nping --tcp -p 80 www.zencurity.com
```

```
Starting Nping 0.7.70 ( https://nmap.org/nping ) at 2018-09-07 19:06 CEST
```

```
SENT (0.0300s) TCP 10.137.0.24:3805 > 185.129.60.130:80 S ttl=64 id=18933 iplen=40 seq=2984847972 win=1480
RCVD (0.0353s) TCP 185.129.60.130:80 > 10.137.0.24:3805 SA ttl=56 id=49674 iplen=44 seq=3654597698 win=16384 <mss
SENT (1.0305s) TCP 10.137.0.24:3805 > 185.129.60.130:80 S ttl=64 id=18933 iplen=40 seq=2984847972 win=1480
RCVD (1.0391s) TCP 185.129.60.130:80 > 10.137.0.24:3805 SA ttl=56 id=50237 iplen=44 seq=2347926491 win=16384 <mss
SENT (2.0325s) TCP 10.137.0.24:3805 > 185.129.60.130:80 S ttl=64 id=18933 iplen=40 seq=2984847972 win=1480
RCVD (2.0724s) TCP 185.129.60.130:80 > 10.137.0.24:3805 SA ttl=56 id=9842 iplen=44 seq=2355974413 win=16384 <mss
SENT (3.0340s) TCP 10.137.0.24:3805 > 185.129.60.130:80 S ttl=64 id=18933 iplen=40 seq=2984847972 win=1480
RCVD (3.0387s) TCP 185.129.60.130:80 > 10.137.0.24:3805 SA ttl=56 id=1836 iplen=44 seq=3230085295 win=16384 <mss
SENT (4.0362s) TCP 10.137.0.24:3805 > 185.129.60.130:80 S ttl=64 id=18933 iplen=40 seq=2984847972 win=1480
RCVD (4.0549s) TCP 185.129.60.130:80 > 10.137.0.24:3805 SA ttl=56 id=62226 iplen=44 seq=3033492220 win=16384 <mss
```

```
Max rtt: 40.044ms | Min rtt: 4.677ms | Avg rtt: 15.398ms
Raw packets sent: 5 (200B) | Rcvd: 5 (220B) | Lost: 0 (0.00%)
Nping done: 1 IP address pinged in 4.07 seconds
```

Hints:

A lot of options are similar to Nmap

Solution:

When you have tried it towards an open port, a closed port and an IP/port that is filtered you are done.

Discussion:

A colleague of ours had problems sending specific IPsec packets through a provider. Using a tool like Nping it is possible to show what happens, or where things are blocked.

Things like changing the TTL may provoke ICMP messages, like this:

```
root@KaliVM:~# nping --tcp -p 80 --ttl 3 www.zencurity.com
```

```
Starting Nping 0.7.70 ( https://nmap.org/nping ) at 2018-09-07 19:08 CEST
```

```
SENT (0.0303s) TCP 10.137.0.24:37244 > 185.129.60.130:80 S ttl=3 id=60780 iplen=40 seq=1997801125 win=1480
RCVD (0.0331s) ICMP [10.50.43.225 > 10.137.0.24 TTL=0 during transit (type=11/code=0) ] IP [ttl=62 id=28456 iplen=7
SENT (1.0314s) TCP 10.137.0.24:37244 > 185.129.60.130:80 S ttl=3 id=60780 iplen=40 seq=1997801125 win=1480
RCVD (1.0337s) ICMP [10.50.43.225 > 10.137.0.24 TTL=0 during transit (type=11/code=0) ] IP [ttl=62 id=28550 iplen=7
SENT (2.0330s) TCP 10.137.0.24:37244 > 185.129.60.130:80 S ttl=3 id=60780 iplen=40 seq=1997801125 win=1480
RCVD (2.0364s) ICMP [10.50.43.225 > 10.137.0.24 TTL=0 during transit (type=11/code=0) ] IP [ttl=62 id=28589 iplen=7
SENT (3.0346s) TCP 10.137.0.24:37244 > 185.129.60.130:80 S ttl=3 id=60780 iplen=40 seq=1997801125 win=1480
RCVD (3.0733s) ICMP [10.50.43.225 > 10.137.0.24 TTL=0 during transit (type=11/code=0) ] IP [ttl=62 id=29403 iplen=7
SENT (4.0366s) TCP 10.137.0.24:37244 > 185.129.60.130:80 S ttl=3 id=60780 iplen=40 seq=1997801125 win=1480
RCVD (4.0558s) ICMP [10.50.43.225 > 10.137.0.24 TTL=0 during transit (type=11/code=0) ] IP [ttl=62 id=30235 iplen=7
```

```
Max rtt: 38.574ms | Min rtt: 2.248ms | Avg rtt: 13.143ms
```

```
Raw packets sent: 5 (200B) | Rcvd: 5 (360B) | Lost: 0 (0.00%)
```

```
Nping done: 1 IP address pinged in 4.07 seconds
```

Exercise 10

Try pcap-diff 15 min

Objective:

Try both getting an utility tool from Github and running an actual useful tool for comparing packet captures.

Purpose:

Being able to get tools and scripts from Github makes you more effective.

The tool we need today is <https://github.com/isginf/pcap-diff> **Suggested method:** Git clone the repository, follow instructions for running a packet diff.

Try saving a few packets in a packet capture, then using tcpdump read and write a subset - so you end up with two packet captures:

```
sudo tcpdump -w icmp-dump.cap
// run ping in another window, which probably creates ARP packets
// Check using tcpdump
sudo tcpdump -r icmp-dump.cap arp
reading from file icmp-dump.cap, link-type EN10MB (Ethernet)
10:06:18.077055 ARP, Request who-has 10.137.0.22 tell 10.137.0.6, length 28
10:06:18.077064 ARP, Reply 10.137.0.22 is-at 00:16:3e:5e:6c:00 (oui Unknown), length 28
10:06:24.776987 ARP, Request who-has 10.137.0.6 tell 10.137.0.22, length 28
10:06:24.777107 ARP, Reply 10.137.0.6 is-at fe:ff:ff:ff:ff:ff (oui Unknown), length 28
// Write the dump - but without the ARP packets:
sudo tcpdump -r icmp-dump.cap -w icmp-dump-no-arp.cap not arp
```

With these pcaps you should be able to do:

```
sudo pip install scapy
git clone https://github.com/isginf/pcap-diff.git
cd pcap-diff/

$ python pcap_diff.py -i ../icmp-dump.cap -i ../icmp-dump-no-arp.cap -o diff.cap
Reading file ../icmp-dump.cap:
Found 23 packets

Reading file ../icmp-dump-no-arp.cap:
Found 19 packets

Diffing packets:

Found 2 different packets

Writing diff.cap
// Try reading the output packet diff:

$ sudo tcpdump -r diff.cap
```

```
reading from file diff.cap, link-type EN10MB (Ethernet)
10:06:24.777107 ARP, Reply 10.137.0.6 is-at fe:ff:ff:ff:ff:ff (oui Unknown), length 28
10:06:24.776987 ARP, Request who-has 10.137.0.6 tell 10.137.0.22, length 28
```

Note: I ran these on a Debian, so I needed the sudo, if you run this on Kali there is no need to use sudo.

Hints:

Git is one of the most popular software development tools, and Github is a very popular site for sharing open source tools.

Solution:

When you or your team mate has a running pcap-diff then you are done

Discussion:

I often find that 90% of my tasks can be done using existing open source tools.

Discover active systems ping sweep 10 min



Exercise 12

Execute nmap TCP and UDP port scan 20 min

Objective:

Use nmap to discover important open ports on active systems

Purpose:

Finding open ports will allow you to find vulnerabilities on these ports.

Suggested method:

Use `nmap -p 1-1024 server` to scan the first 1024 TCP ports and use Nmap without ports. What is scanned then?

Try to use `nmap -sU` to scan using UDP ports, not really possible if a firewall is in place.

If a firewall blocks ICMP you might need to add `-Pn` to make nmap scan even if there are no Ping responses

Hints:

Sample command: `nmap -Pn -sU -p1-1024 server` UDP port scanning 1024 ports without doing a Ping first

Solution:

Discover some active systems and most interesting ports, which are 1-1024 and the built-in list of popular ports.

Discussion:

There is a lot of documentation about the nmap portscanner, even a book by the author of nmap. Make sure to visit <http://www.nmap.org>

TCP and UDP is very different when scanning. TCP is connection/flow oriented and requires a handshake which is very easy to identify. UDP does not have a handshake and most applications will not respond to probes from nmap. If there is no firewall the operating system will respond to UDP probes on closed ports - and the ones that do not respond must be open.

When doing UDP scan on the internet you will almost never get a response, so you cannot tell open (not responding services) from blocked ports (firewall drop packets). Instead try using specific service programs for the services, sample program could be `nsping` which sends DNS packets, and will often get a response from a DNS server running on UDP port 53.

Exercise 13

Perform nmap OS detection 10 min

Objective:

Use nmap OS detection and see if you can guess the brand of devices on the network

Purpose:

Getting the operating system of a system will allow you to focus your next attacks.

Suggested method:

Look at the list of active systems, or do a ping sweep.

Then add the OS detection using the option `-O`

Better to use `-A` all the time, includes even more scripts and advanced stuff See the next exercise.

Hints:

The nmap can send a lot of packets that will get different responses, depending on the operating system. TCP/IP is implemented using various constants chosen by the implementors, they have chosen different standard packet TTL etc.

Solution:

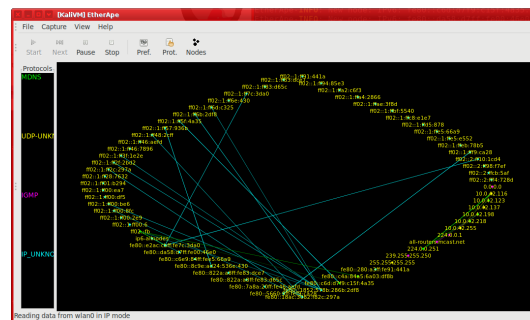
Use a command like `nmap -O -p1-100 10.0.45.45` or `nmap -A -p1-100 10.0.45.45`

Discussion:

nmap OS detection is not a full proof way of knowing the actual operating system, but in most cases it can detect the family and in some cases it can identify the exact patch level of the system.

Exercise 14

EtherApe 10 min



EtherApe is a graphical network monitor for Unix modeled after ethernan. Featuring link layer, IP and TCP modes, it displays network activity graphically. Hosts and links change in size with traffic. Color coded protocols display. Node statistics can be exported.

Objective:

Use a tool to see more about network traffic, whats going on in a network.

Purpose:

Get to know the concept of a node by seeing nodes communicate in a graphical environment.

Suggested method:

Use the tool from Kali

The main page for the tool is: <https://etherape.sourceforge.io/>

Hints:

Your built-in network card may not be the best for sniffing. Borrow one from Henrik.

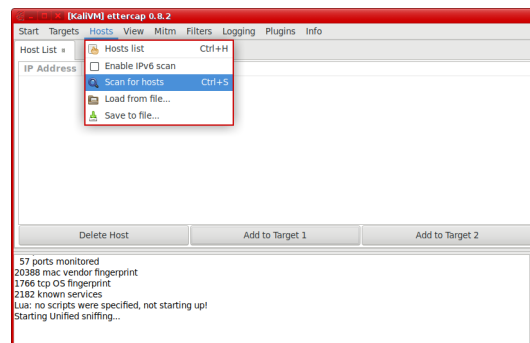
Solution:

When you have the tool running and showing data, you are done.

Discussion:

Exercise 15

ARP spoofing and ettercap 20



Objective:

Use a tool to see more about network traffic, whats going on in a network.

Purpose:

Start the tool, do a scan and start sniffing between your laptop and the router.

Suggested method:

1. Start the tool using ettercap --gtk to get the graphical version.
2. Select menu Info, Help - and read about unified and bridged sniffing.
3. Start Unified sniffing from Sniff, Unified sniffing - select your network card.
4. Select Hosts - Scan

You should be able to see some hosts. Then the next step would be to initiate attacks - which are menu-driven and easy to perform.

Hints:

We might be messing to much with the traffic, so attacks wont succeed. Some coordination is needed.

Solution:

When you can scan for hosts and realize how easy that was, you are done.

Discussion:

How many admins know about ARP spoofing, ARP poisoning?

Exercise 16

Perform nmap service scan 10 min

Objective:

Use more advanced features in Nmap to discover services.

Purpose:

Getting more intimate with the system will allow more precise discovery of the vulnerabilities and also allow you to select the next tools to run.

Suggested method:

Use `nmap -A` option for enabling service detection and scripts

Hints:

Look into the manual page of nmap or the web site book about nmap scanning

Solution:

Run nmap and get results.

Discussion:

Some services will show software versions allowing an attacker easy lookup at web sites to known vulnerabilities and often exploits that will have a high probability of success.

Make sure you know the difference between a vulnerability which is discovered, but not really there, a false positive, and a vulnerability not found due to limitations in the testing tool/method, a false negative.

A sample false positive might be reporting that a Windows server has a vulnerability that you know only to exist in Unix systems.

Exercise 17

Nmap full scan - strategy 15 min

Objective:

Write down your Nmap strategy, and if needed create your own Nmap profile in Zenmap.

Purpose:

Doing a port scan often requires you to run multiple Nmap scans.

Suggested method:

Use Zenmap to do:

1. A few quick scans, to get web servers and start web scanners/crawlers
2. Full scan of all TCP ports, -p 1-65535
3. Full or limited UDP scan, `nmap -sU --top-ports 100`
4. Specialized scans, like specific source ports

Hints:

Using a specific source ports using `-g/--source-port <portnum>`: Use given port number with ports like FTP 20, DNS 53 can sometimes get around router filters and other stateless Access Control Lists

Solution:

Run nmap and get results.

Discussion:

Recommendation it is highly recommended to always use:

`-iL <inputfilename>`: Input from list of hosts/networks
`-oA outputbasename`: output in all formats, see later

Some examples of real life Nmaps I have run recently:

```
dns-scan: nmap -sU -p 53 --script=dns-recursion -iL targets -oA dns-recursive
bgpscan: nmap -A -p 179 -oA bgpscan -iL targets
dns-recursive: nmap -sU -p 53 --script=dns-recursion -iL targets -oA dns-recursive
php-scan: nmap -sV --script=http-php-version -p80,443 -oA php-scan -iL targets
scan-vtep-tcp: nmap -A -p 1-65535 -oA scan-vtep-tcp 185.129.60.77 185.129.60.78
```

```
snmp-10.x.y.0.gnmap: nmap -sV -A -p 161 -sU --script=snmp-info -oA snmp-10xy 10.x.y.0/19
snmpscan: nmap -sU -p 161 -oA snmpscan --script=snmp-interfaces -iL targets
sshscan: nmap -A -p 22 -oA sshscan -iL targets
vncscan: nmap -A -p 5900-5905 -oA vncscan -iL targets
```

Exercise 18

Reporting HTML 15 min

Nmap Scan Report - Scanned at Fri Sep 7 18:35:54 2018						
Scan Summary www.zencurity.com (185.129.60.130)						
Scan Summary						
Nmap 7.70 was initiated at Fri Sep 7 18:35:54 2018 with these arguments: <code>nmap -oA zencurity-web www.zencurity.com</code>						
Verbosity: 0; Debug level 0						
Nmap done at Fri Sep 7 18:35:59 2018; 1 IP address (1 host up) scanned in 4.90 seconds						
185.129.60.130 / www.zencurity.com						
Address						
• 185.129.60.130 (ipv4)						
Hostnames						
• www.zencurity.com (user)						
Ports						
The 998 ports scanned but not shown below are in state: filtered						
• 998 ports replied with: no-responses						
Port	State (toggle closed [0] filtered [0])	Service	Reason	Product	Version	Extra info
80	tcp open	http	syn-ack			
443	tcp open	https	syn-ack			

Objective:

Show the use of XML output and convert to HTML

Purpose:

Reporting data is very important. Using the oA option Nmap can export data in three formats easily, each have their use. They are normal, XML, and grepable formats at once.

Suggested method:

First run Nmap, with output, then process it

```
sudo nmap -oA zencurity-web www.zencurity.com
xsltproc zencurity-web.xml > zencurity-web.html
```

Hints:

Nmap includes the stylesheet in XML and makes it very easy to create HTML.

If the tool xsltproc is not installed, then install it: `apt-get install xsltproc`

Solution:

Run XML through xsltproc, command line XSLT processor, or another tool

Discussion:

Options you can use to change defaults:

--stylesheet <path/URL>: XSL stylesheet to transform XML output to HTML
--webxml: Reference stylesheet from Nmap.Org for more portable XML

Also check out the Ndiff tool

```
hlk@cornerstone03:~$ ndiff zencurity-web.xml zencurity-web-2.xml
-Nmap 7.70 scan initiated Fri Sep 07 18:35:54 2018 as: nmap -oA zencurity-web www.zencurity.com
+Nmap 7.70 scan initiated Fri Sep 07 18:46:01 2018 as: nmap -oA zencurity-web-2 www.zencurity.com

www.zencurity.com (185.129.60.130):
PORT      STATE SERVICE VERSION
+443/tcp  open  https
```

(I ran a scan, removed a port from the first XML file and re-scanned)

Exercise 19

SSL/TLS scanners 15 min

Objective:

Try the Online Qualys SSLabs scanner <https://www.ssllabs.com/> Try the command line tool sslscan checking servers - can check both HTTPS and non-HTTPS protocols!

Purpose:

Learn how to efficiently check TLS settings on remote services.

Suggested method:

Run the tool against a couple of sites of your choice.

```
root@kali:~# sslscan --ssl2 web.kramse.dk
Version: 1.10.5-static
OpenSSL 1.0.2e-dev xx XXX xxxx

Testing SSL server web.kramse.dk on port 443
...
  SSL Certificate:
Signature Algorithm: sha256WithRSAEncryption
RSA Key Strength:    2048

Subject: *.kramse.dk
AltNames: DNS:*.kramse.dk, DNS:kramse.dk
Issuer:  AlphaSSL CA - SHA256 - G2
```

Also run it without --ssl2 and against SMTPTLS if possible.

Hints:

Originally sslscan is from <http://www.titania.co.uk> but use the version on Kali, install with apt if not installed.

Solution:

When you can run and understand what the tool does, you are done.

Discussion:

SSLscan can check your own sites, while Qualys SSLabs only can test from hostname

Exercise 20

sslstrip 15 min

Objective:

sslstrip <https://moxie.org/software/sslstrip/>

Purpose:

Read about the tool, and lets try to run it - on a single VM.

This tool provides a demonstration of the HTTPS stripping attacks that I presented at Black Hat DC 2009. It will transparently hijack HTTP traffic on a network, watch for HTTPS links and redirects, then map those links into either look-alike HTTP links or homograph-similar HTTPS links. It also supports modes for supplying a favicon which looks like a lock icon, selective logging, and session denial. For more information on the attack, see the video from the presentation below.

Suggested method:

Make sure tool is installed, Then run it and intercept your own traffic from the same system.

You may run this on the wireless and try intercepting others.

Hints:

IF you are using wireless - most likely, then make sure to run on the same channel/AP/frequency. Either switch everything to 2.4GHz and have only one AP or just do the mitm on a single host - run browser and mitmproxy on the same VM.

Solution:

When you have intercepted some traffic you are done, we will spend at least 30-45 minutes doing various mitm related stuff.

Discussion:

Exercise 21

mitmproxy 30 min

Objective:

mitmproxy <https://mitmproxy.org/>

mitmproxy is a free and open source interactive HTTPS proxy

Purpose:

Try running a mitm attack on your phone or another laptop.

Suggested method:

Make sure tool is installed

Then use the command line interface to verify it is working, then switch to the Web interface for playing with the tool.

Hints:

IF you are using wireless - most likely, then make sure to run on the same channel/AP/frequency. Either switch everything to 2.4GHz and have only one AP or just do the mitm on a single host - run browser and mitmproxy on the same VM.

Solution:

When you have intercepted some traffic you are done, we will spend at least 30-45 minutes doing various mitm related stuff.

Discussion:

Exercise 22

sslsplit 10 min

Objective:

Read about sslsplit <https://www.roe.ch/SSLsplit> - transparent SSL/TLS interception system

Purpose:

This tool has a lot of features described on the home page.

Overview

SSLsplit is a tool for man-in-the-middle attacks against SSL/TLS encrypted network connections. It is intended to be useful for network forensics, application security analysis and penetration testing.

SSLsplit is designed to transparently terminate connections that are redirected to it using a network address translation engine. SSLsplit then terminates SSL/TLS and initiates a new SSL/TLS connection to the original destination address, while logging all data transmitted. Besides NAT based operation, SSLsplit also supports static destinations and using the server name indicated by SNI as upstream destination. SSLsplit is purely a transparent proxy and cannot act as a HTTP or SOCKS proxy configured in a browser.

Suggested method:

If we were to use this tool, we would redirect traffic using "firewalls"/routers

- SSLsplit currently supports the following operating systems and NAT engines:
- FreeBSD: pf rdr and divert-to, ipfw fwd, ipfilter rdr
- OpenBSD: pf rdr-to and divert-to
- Linux: netfilter REDIRECT and TPROXY
- Mac OS X: ipfw fwd and pf rdr

We wont run this tool, but beware such tools exist

Hints:

Specifically read the section *SSLsplit implements a number of defences against mechanisms* - and think about the consequences to a regular user.

Solution:

When you feel you have a idea about what this tool can do then you are done.

Discussion:

Should tools like this even exist?

Exercise 23

Wardriving Up to 60min

Objective:

Try putting a network card in monitor mode and sniff wireless networks.

Purpose:

See that wireless networks dont encrypt MACs addresses and other characteristics - what can be found just by turning on the radio.

Suggested method:

Insert USB wireless card, make sure your VM has USB 2.0 Hub and allow VM to control the card.

Start monitor mode - maybe card is not wlan0!:

```
airmon-ng start wlan0
```

Start airodump-ng:

```
airodump-ng wlan0mon
```

See the data

Hints:

Selecting a specific channel can be done using `-channel` and writing captured packets can be done using `-w`

Solution:

When you have an overview of nearby networks you are done.

Discussion:

Lots of information is available on the internet. One recommended site is: <http://www.aircrack-ng>

Exercise 24

Aircrack-ng 30 min

Objective:

See the program aircrack-ng being used for cracking WEP and WPA-PSK keys.

Purpose:

Some methods previously used to protect wireless networks should not be used anymore.

Suggested method:

Get access to a WEP encrypted dump of wireless network traffic and break encryption.
Get access to a WPA handshake and try cracking it.

Hints:

Kali includes the aircrack-ng program and some test data in
`/pentest/wireless/aircrack-ng/test`

Solution:

When you have cracked a network from either testdata or real nearby - our lab network.

Discussion:

There is a lot of information available about aircrack-ng at the web site:
<http://www.aircrack-ng.org/>

Another tool available is pyrit and cpyrit which can break WPA-PSK using CUDA enabled graphic cards - instead of 100s of keys/second this may allow 10000s keys/second.

Hashcat also is able to crack WPA <https://hashcat.net/hashcat/>

Exercise 25

SNMP walk 15min

Objective:

Run SNMP walk on a switch, `snmpwalk` see information from device.

Lots of basic information helps defenders - AND attackers.

Purpose:

SNMP is a default management protocol used in almost any network.

Suggested method:

Run `snmpwalk` using community string `public`. Command is: `snmpwalk -v 2c -c public system`

Hints:

Community strings `private` and `public` used to be default for network devices. Today professional devices have no SNMP configured by default, but lots of networks install the community "public" anyway.

Solution:

When you have done `snmpwalk` for at least one device.

Discussion:

Which part of the information is most relevant to defenders, and attackers.

Also remember on internal LAN segments, use `Nmap`:

`snmp-10.x.y.0.gnmap: nmap -sV -A -p 161 -sU --script=snmp-info -oA snmp-10xy 10.x.y.0/19`

`snmpscan: nmap -sU -p 161 -oA snmpscan --script=snmp-interfaces -iL targets`

Exercise 26

Try Hydra brute force 30min

Objective:

Try a brute force program named hydra/Xhydra.

You decide which service to attack, SSH or SNMP are good examples.

Purpose:

Learn that some protocols allow brute forcing.

Suggested method:

Make a short list of usernames and a short list of passwords and use hydra to brute force your way into a system. Use the editor `kate`, using `kate users.txt` and `kate pass.txt` followed by a command similar to this:

```
$ hydra -V -t 1 -L users.txt -P pass.txt 10.0.45.2 ssh
```

If you want to attack SNMP try with a small list of community names: `public`, `private`, `kramse`, etc.

Hints:

When learning tools create a nice environment and check that things are working before trying to hack. So with brute forcing an account, create and test it!

Solution:

When you have found working credentials for at least one service, like SNMP and done SNMPwalk

Discussion:

The hydra program can brute force a lot of different protocols and also allow a lot of tuning.

The hydra program does an online brute force attack, in some cases you can get access to data like password databases, or hash values that can be cracked in off-line brute force attacks.

Exercise 27

Zeek on the web 10min

Objective:

Try Zeek Network Security Monitor - without installing it.

Purpose:

Show a couple of examples of Zeek scripting, the built-in language found in Zeek Network Security Monitor

Suggested method:

Go to <http://try.bro.org/> and try a few of the examples.

Hints:

The exercise *The Summary Statistics Framework* can be run with a specific PCAP.

```
192.168.1.201 did 402 total and 2 unique DNS requests in the last 6 hours.
```

Solution:

You should read the example *Raising a Notice*. Getting output for certain events may be interesting to you.

Discussion:

Zeek Network Security Monitor is an old/mature tool, but can still be hard to get started using. I would suggest that you always start out using the packages available in your Ubuntu/Debian package repositories.

They work, and will give a first impression of Zeek. If you later want specific features not configured into the binary packet, then install from source.

Also Zeek uses a broctl program to start/stop the tool, and a few config files which we should look at. From a Debian system they can be found in `/etc/bro` :

```
root@NMS-VM:/etc/bro# ls -la
drwxr-xr-x  3 root root  4096 Oct  8 08:36 .
drwxr-xr-x 138 root root 12288 Oct  8 08:36 ..
-rw-r--r--  1 root root  2606 Oct 30  2015 broctl.cfg
-rw-r--r--  1 root root   225 Oct 30  2015 networks.cfg
-rw-r--r--  1 root root   644 Oct 30  2015 node.cfg
drwxr-xr-x  2 root root  4096 Oct  8 08:35 site
```

Exercise 28

Zeek DNS capturing domain names 10min

Objective:

We will now start using Zeek on our systems.

Purpose:

Try Zeek with example traffic, and see what happens.

Suggested method packet capture file:

Note: a dollar sign is the Linux prompt, showing the command after

```
$ cd
$ wget http://downloads.digitalcorpora.org/corpora/network-packet-dumps/2008-nitroba/nitroba.pcap
$ mkdir $HOME/bro; cd $HOME/bro; bro -r ../nitroba.pcap
... bro reads the packets
~/bro$ ls
conn.log  dns.log  dpd.log  files.log  http.log  packet_filter.log
sip.log  ssl.log  weird.log  x509.log
$ less *
```

Use :n to jump to the next file in less, go through all of them.

Suggested method Live traffic:

Make sure Zeek is configured as a standalone probe and configured for the right interface. Linux used to use eth0 as the first ethernet interface, but now can use others, like ens192 or enx00249b1b2991.

```
root@NMS-VM:/etc/bro# cat node.cfg
# Example BroControl node configuration.
#
# This example has a standalone node ready to go except for possibly changing
# the sniffing interface.

# This is a complete standalone configuration.  Most likely you will
# only need to change the interface.
[bro]
type=standalone
host=localhost
interface=eth0
...
```

Hints:

There are multiple commands for showing the interfaces and IP addresses on Linux. The old way is using `ifconfig -a` newer systems would use `ip a`

Note: if your system has a dedicated interface for capturing, you need to turn it on, make it available. This can be done manually using `ifconfig eth0 up` **Solution:** When you either run Zeek using a packet capture or using live traffic

Running with a capture can be done using a command line such as: `bro -r traffic.pcap`

Using `broctl` to start it would be like this:

```
// install bro first
kunoichi:~ root# broctl
Hint: Run the broctl "deploy" command to get started.
```

```
Welcome to BroControl 1.5
Type "help" for help.
```

```
[BroControl] > install
creating policy directories ...
installing site policies ...
generating standalone-layout.bro ...
generating local-networks.bro ...
generating broctl-config.bro ...
generating broctl-config.sh ...
...
```

```
// back to Broctl and start it
[BroControl] > start
starting bro
// and then
kunoichi:bro root# cd /var/spool/bro/bro
kunoichi:bro root# tail -f dns.log
```

You should be able to spot entries like this:

```
#fields ts      uid      id.orig_h      id.orig_p      id.resp_h      id.resp_p      proto  trans_i
      query  qclass  qclass_name    qtype  qtype_name    rcode  rcode_name    AA    TC    RD
1538982372.416180 CD12Dc1SpQm42QW4G3 10.xxx.0.145 57476 10.x.y.141 53 udp 20383 0.045021 www.dr.dk
1 C_INTERNET 1 A 0 NOERROR F F T T 0 www.dr.dk-v1.edgekey.net,e16198.b.akamaiedge.net,2.17.212.93
60.000000,20409.000000,20.000000 F
```

Note: this show ALL the fields captured and dissected by Zeek, there is a nice utility program `bro-cut` which can select specific fields:

```
root@NMS-VM:/var/spool/bro/bro# cat dns.log | bro-cut -d ts query answers | grep dr.dk
2018-10-08T09:06:12+0200 www.dr.dk www.dr.dk-v1.edgekey.net,e16198.b.akamaiedge.net,2.17.212.93
```

Discussion:

Why is DNS interesting?

Exercise 29

Zeek TLS capturing certificates 10min

Objective:

Run more traffic through Zeek, see the various files.

Purpose:

See that even though HTTPS and TLS traffic is encrypted it often show names and other values from the certificates and servers.

Suggested method:

Run Zeek capturing live traffic, start https towards some sites. A lot of common sites today has shifted to HTTPS/TLS.

Hints:

use broctl start and watch the output directory

```
root@NMS-VM:/var/spool/bro/bro# ls *.log
communication.log  dhcp.log  files.log  known_services.log  packet_filter.log  stats.log
stdout.log  x509.log  conn.log  dns.log  known_hosts.log  loaded_scripts.log  ssl.log
stderr.log  weird.log
```

We already looked at dns.log, now check ssl.log and x509.log

```
root@NMS-VM:/var/spool/bro/bro# grep dr.dk ssl.log
1538983060.546122 CtKYZ625cq3m3jUz9k 10.xxx.0.145 49932 2.17.212.93 443 TLSv12 TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 secp256r1 www.dr.dk F -h2 T FzmZCt3o9EYcmNaxIi,FKXcmxQHT3znDDMSj (empty) CN=*.dr.dk,O=DR,L=Copenhagen S
CN=GlobalSign Organization Validation CA - SHA256 - G2,O=GlobalSign nv-sa,C=BE --ok
1538983060.674217 CLjZo51fzuTcvPT0lg 200xxxxb:89b0:5cbf 49933 2a02:26f0:2400:2a1::3f46 443 TLSv12
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 secp256r1 asset.dr.dk F -h2 TFEpW9a1IFe6NTUZNpb,FwV50B4CHIwF1CPlu
(empty) CN=*.dr.dk,O=DR,L=Copenhagen S,ST=Copenhagen,C=DK CN=GlobalSign Organization Validation CA - SH
sa,C=BE --ok
```

Solution:

When you have multiple log files with data from Zeek, and have looked into some of them. You are welcome to ask questions and look into more files.

Discussion:

How can you hide that you are going to HTTPS sites?

Hint: VPN

Exercise 30

Suricata Basic Operation 15min

Objective:

Start using Suricata IDS engine with some traffic.

Purpose:

Show how to get started, meet some obstacles - missing files etc.

Discuss how to solve problems, why do we miss them, how to fix

Suggested method packet capture file:

Note: a dollar sign is the Linux prompt, showing the command after

```
$ cd
$ wget http://downloads.digitalcorpora.org/corpora/network-packet-dumps/2008-nitroba/nitroba.pcap
$ mkdir $HOME/suricata;cd $HOME/suricata;  suricata -r ../nitroba.pcap -c /etc/suricata/suricata.yaml -
l .
... Suricata reads the packets
~/suricata$ ls
eve.json  fast.log  stats.log
$ less *
```

Suggested method live capture:

Make sure the config file /etc/suricata/suricata.yaml has the right interface eth0 - or maybe ens192?. Check using ifconfig -a

Try starting the service

```
hlk@debian:~$ sudo service suricata start
hlk@debian:~$ cd /var/log/suricata/
hlk@debian:/var/log/suricata$ ls
eve.json  fast.log  stats.log  suricata.log
hlk@debian:/var/log/suricata$

hlk@debian:/var/log/suricata$ tail -3 suricata.log
8/10/2018 -- 17:15:58 - <Warning> - [ERRCODE: SC_ERR_AFP_CREATE(190)] - Can not open iface 'eth0'
8/10/2018 -- 17:15:58 - <Warning> - [ERRCODE: SC_ERR_AFP_CREATE(190)] - Can not open iface 'eth0'
8/10/2018 -- 17:16:19 - <Warning> - [ERRCODE: SC_ERR_AFP_CREATE(190)] - Can not open iface 'eth0'
```

Yeah my network card is called ens33, and I should replace eth0 with ens33 in the config file.

```
perl -pi -e "s/eth0/ens33/g" /etc/suricata/suricata.yaml
```

```
hlk@debian:/var/log/suricata$ sudo service suricata stop
hlk@debian:/var/log/suricata$ sudo service suricata start
hlk@debian:/var/log/suricata$ tail -3 suricata.log
8/10/2018 -- 17:23:20 - <Warning> - [ERRCODE: SC_ERR_NO_RULES(42)] - No rule files match the pattern /e
worm.rules
8/10/2018 -- 17:23:20 - <Warning> - [ERRCODE: SC_ERR_NO_RULES(42)] - No rule files match the pattern /e
8/10/2018 -- 17:23:20 - <Notice> - all 2 packet processing threads, 4 management threads initialized, e
```

Hints:

https://redmine.openinfosecfoundation.org/projects/suricata/wiki/Quick_Start_Guide and
https://redmine.openinfosecfoundation.org/projects/suricata/wiki/Basic_Setup

Solution:

When you can start and stop suricata, and it only complains about missing rules, you are done with this exercise.

Discussion:

What was the main problems in this exercise?

Exercise 31

Basic Suricata rule configuration 15min

Objective:

See the Suricata configuration files, and get some rules.

The best IDS is nothing without good rules.

Purpose:

The rules make Suricata useful, and we will learn how to get a ruleset installed, and to keep it updated.

Suggested method:

Check the file `/etc/suricata/suricata-oinkmaster.conf`

It contains this:

```
hlk@debian:/etc/suricata$ cat suricata-oinkmaster.conf
# This is a Debian specific config file for oinkmaster crafted for suricata,
# you should read oinkmaster documentation to modify this file.
# This config is loaded by default from the suricata-oinkmaster-updater binary
# which is called daily from a cronjob by default

skipfile local.rules
skipfile deleted.rules
skipfile snort.conf
use_external_bins = 0

url = https://rules.emergingthreats.net/open/suricata-3.0/emerging.rules.tar.gz
```

Then try running the oinkmaster program in "dry run" with `-c`

```
root@debian:~# oinkmaster -i -c -C /etc/suricata/suricata-oinkmaster.conf -o /etc/suricata/rules/
Loading /etc/suricata/suricata-oinkmaster.conf
Downloading file from https://rules.emergingthreats.net/open/suricata-3.0/emerging.rules.tar.gz... done
Archive successfully downloaded, unpacking... done.
Setting up rules structures... done.
Processing downloaded rules... disablesid 0, enablesid 0, modifiesid 0, localsid 0, total rules 26212
```

If the output looks OK, then re-run without `-c` and let it update files.

```
root@debian:~# oinkmaster -i -C /etc/suricata/suricata-oinkmaster.conf -o /etc/suricata/rules/
...
```

```
[+] Added files (consider updating your snort.conf to include them if needed):
-> botcc.portgrouped.rules
-> botcc.rules
-> BSD-License.txt
-> ciarmy.rules
...
-> emerging-chat.rules
-> emerging-current_events.rules
-> emerging-deleted.rules
-> emerging-dns.rules
-> emerging-dos.rules
-> emerging-exploit.rules
-> emerging-ftp.rules
Do you approve these changes? [Yn]
```

Hints:

You need to restart Suricata for the rules to be found. In the example below I remove the long log with errors, and restart:

```
root@debian:~# service suricata stop
root@debian:~# rm /var/log/suricata/suricata.log
root@debian:~# service suricata start
root@debian:~# cat /var/log/suricata/suricata.log
8/10/2018 -- 17:45:19 - <Notice> - This is Suricata version 3.2.1 RELEASE
```

Solution:

When you have the ruleset downloaded and Suricata is happy when starting you are done with this exercise.

In a real deployment it is advised to automate the update of rules, and also some rules are probably not needed in your environments, YMMV. We will not go through all the rules provided.

Discussion:

Emerging Threats is a well-known ruleset provider, with commercial support.

Whenever there is a new internet wide security incident there are people providing IDS rules in Snort or Suricata format. Since Suricata can read snort rules, this is a good way to add up-to-date rules to your installation.

Note: we haven't mentioned it, but the config files for both Zeek and Suricata allows one to specify your home network.

Checkout the files: Zeek configuration in `/etc/bro/networks.cfg` and Suricata main config `/etc/suricata/suricata.yaml`

```
vars:
  # more specific is better for alert accuracy and performance
  address-groups:
    HOME_NET: "[192.168.0.0/16,10.0.0.0/8,172.16.0.0/12]"
    #HOME_NET: "[192.168.0.0/16]"
```


Exercise 32

Configure Mirror Port 10min

Objective:

Mirror ports are a way to copy traffic to Suricata and other devices - for analyzing it. We will go through the steps on a Juniper switch to show how. Most switches which are configurable have this possibility.

Purpose:

We want to capture traffic for multiple systems, so we select an appropriate port and copy the traffic. In our setup, we select the uplink port to the internet/router.

It is also possible to buy passive taps, like a fiber splitter, which then takes part of the signal, and is only observable if you look for signal strength on the physical layer.

Suggested method:

We will configure a mirror port on a Juniper EX2200-C running Junos.

```
root@ex2200-c# show ethernet-switching-options | display set
set ethernet-switching-options analyzer mirror01 input ingress interface ge-0/1/1.0
set ethernet-switching-options analyzer mirror01 input egress interface ge-0/1/1.0
set ethernet-switching-options analyzer mirror01 output interface ge-0/1/0.0
set ethernet-switching-options storm-control interface all
```

Hints:

When checking your own devices this is often called SPAN ports, Mirror ports or similar.

https://en.wikipedia.org/wiki/Port_mirroring

Cisco has called this Switched Port Analyzer (SPAN) or Remote Switched Port Analyzer (RSPAN), so many will refer to them as SPAN-ports.

Solution:

When we can see the traffic from the network, we have the port configured - and can run any tool we like. Note: specialized capture cards can often be configured to spread the load of incoming packets onto separate CPU cores for performance. Capturing 100G and more can also be done using switches like the example found on the Zeek web site using an Arista switch 7150.

Discussion:

When is it ethical to capture traffic?

Exercise 33

Save Suricata JSON Output in Database 30min

Objective:

Configure a system to read the output files from Suricata EVE logging and save into database system.

This will enable us to use browser based methods and dashboards to analyse more efficiently.

Purpose:

Flat files show that we can collect data, but processing big files when trying to solve problems or handling security incidents is slow. Using databases and document stores like Elasticsearch can help a lot.

Suggested method:

Open the Suricata config file `suricata.yml` and make sure `eve-log` is turned on. <https://suricata.readthedocs.io/en/suricata-4.0.5/output/eve/eve-json-output.html>

```
# Extensible Event Format (nicknamed EVE) event log in JSON format
- eve-log:
    enabled: yes
```

It might be enabled by default. So you can run the (complex) playbook to install database and supporting tools:

```
ansible-playbook -v elasticstack.yml
```

Run the playbook, that installs:

- Logstash for reading the EVE JSON log
- Elasticsearch the database / document store
- Kibana for showing data
- Nginx, because we really should put this in front of Kibana

Hints:

Logstash and Elastic stack are a great way to get started with dashboarding.

However, running a big installation is harder than it looks. Make sure to have multiple servers and good monitoring.

Solution:

When we have a few running installations we are done. Kibana should be available on port 5601 on localhost (127.0.0.1) only though!

Using Firefox visit Kibana on <http://127.0.0.1:5601> first time you need to select `logstash-*` as a default index. Note: Kibana is an advanced and powerful tool in itself.

Don't be discouraged if something goes wrong, there are a lot of moving pieces.

A very common problem is the permissions to read files, from logstash log:

```
[2018-10-05T18:22:33,105][WARN ][filewatch.tailmode.handlers.createinitial] failed to open
/var/log/suricata/eve.json: #<Errno::EACCES: Permission denied - /var/log/suricata/eve.json>,
["org/jruby/RubyFile.java:366:in `initialize'", "org/jruby/RubyIO.java:1154:in `open'",
"/usr/share/logstash/vendor/bundle/jruby/2.3.0/gems/logstash-input-file-4.1.6
/lib/filewatch/watched_file.rb:204:in `open'"]
```

Discussion:

Making dashboard are an art form. We will NOT start creating beautiful dashboards.

There are a lot of Dashboards available, such as:

<https://github.com/StamusNetworks/KTS6>

Note: they require Suricata 4.1+ so we cannot use them immediately.

If you want, there is a SELKS LiveCD dedicated to suricata which also includes more tools for administration of rules and getting alerts:

<https://www.stamus-networks.com/open-source/>

Exercise 34

Suricata Netflow 10min

Objective:

Configure Suricata to do netflow logging

Purpose:

In some cases we don't know what traffic we need to analyze, but if we collect netflow data - summary data about every connection. We can go back and check for specific types of traffic, based on ports, length etc.

Suggested method:

uncomment netflow in the config file `/etc/suricata/suricata.yaml` by removing the `"#"` in front of this line:

```
#- netflow
```

and restart Suricata.

Hints:

Netflow logging allows efficient logging of summary data, which can be very useful.

Solution:

When you have configured Suricata for netflow, you are done.

Discussion:

Specialized tools exist for collecting and visualizing netflow data. If you have nothing, then Suricata may be a good start.

Exercise 35

Extending Zeek and Suricata 10min

Objective:

Sometimes Zeek and Suricata by themselves will not be enough.

Investigate how to extend Zeek and Suricata, by some examples.

Purpose:

See examples of scripts and rules, evaluate the complexity.

Suggested method:

Get a patch from Henrik for VXLAN support in Suricata. The patch does not need to be installed, but how big is it, how complex is it, could you or your organisation to something similar?

Zeek scripts extend the basic engine, and are a big part of the eco-system. Some 1000s of script lines are already included. Do you have a specific need to analyze in your network which could be implemented in this?

Hints:

Earlier it was quite hard to write C programs for creating and analyzing network traffic. Today we can use the Zeek scripting and Suricata rules to analyze traffic using highly efficient engines.

Solution:

Which tool is easiest to expand, what are you missing from them?

Discussion:

To repeat:

Whenever there is a new internet wide security incident there are people providing IDS rules in Snort or Suricata format. Since Suricata can read snort rules, this is a good way to add up-to-date rules to your installation.

Would you be able to write a rule for something attacking your network?

Exercise 36

Indicators of Compromise 15min

Objective:

Indicators of Compromise is a term used for artifacts observed in networks or systems which indicate that a system was compromised.

This could be a known DNS domain where a specific malware is downloaded from, a specific file name downloaded, a TCP connection to a malware control and command server.

https://en.wikipedia.org/wiki/Indicator_of_compromise

Purpose:

The purpose of this exercise is to look at the data gathered and to start planning how one could use this with IOCs to perform after-the-fact analysis of your network. Goal is to answer how an attack got in, when was the first device compromised etc.

Suggested method:

Look at the data provided by Zeek and Suricata, list the files again. Which parts will be of greatest interest in your networks? Could some of these facts have helped prevent, restrict, limit or otherwise improve your security stance?

Suricata and Zeek can include data from other sources, check the intel module

- Zeek documentation Intel framework
<https://docs.zeek.org/en/stable/frameworks/intel.html>
- Suricata reputation support
<https://suricata.readthedocs.io/en/suricata-4.0.5/reputation/index.html>

Hints:**Solution:**

Done, when you have seen the exercise on the web, not performed it, only looked: and the exercise <https://www.zeek.org/current/exercises/intel/index.html>

Discussion:

Would this need to be updated every day to have value? How do we demonstrate return on investment and benefit from looking at traffic?

Exercise 37

Bonus: VXLAN Detection

Objective:

One recent addition to many networks are cloud environments using tunneling and encapsulation to connect islands of containers and virtual systems.

One such protocol named VXLAN can be used without the network people being involved, which can be bad for security. Also it would be easy for an attacker which have compromised a system to use this for exfiltration of data.

So, do you have any VXLAN traffic in your network?

Purpose:

The main idea of this exercise is to talk about unknown traffic, that which you dont even know exist in your network. Some networks have tunnels and IPv6, but the network and security might not be fully aware of this.

Suggested method:

VXLAN traffic will most likely use the default port 4789, which is not used by much other traffic.

VXLAN is also UDP packets, so analysing if a few endpoints use a LOT of UDP might reveal interesting stuff.

Hints:

The conn.log might show you interesting things about such traffic.

Solution:

We dont have a VXLAN tunnel, but it is very easy to add a VXLAN interface to a Linux server, and start sending data out.

Discussion:

Which protocols are the most dangerous, and why?

Exercise 38

Logging med syslogd og syslog.conf 10min

Objective:

See how server syslog is configured on regular Unix/Linux

Purpose:

The main idea of this exercise is to understand how easy network connected systems can send log data.

Suggested method:

Log into your local Linux systems or network devices, see how syslog is configured.

Hints:

Look in the config file, may be in /etc/syslog or /etc/syslog-ng/syslog-ng.conf

Sample output from old-skool syslogd

```
*.err;kern.debug;auth.notice;authpriv.none;mail.crit    /dev/console
*.notice;auth,authpriv,cron,ftp,kern,lpr,mail,user.none /var/log/messages
kern.debug;user.info;syslog.info                        /var/log/messages
auth.info                                               /var/log/authlog
authpriv.debug                                         /var/log/secure
...
# Uncomment to log to a central host named "loghost".
#*.notice;auth,authpriv,cron,ftp,kern,lpr,mail,user.none @loghost
#kern.debug,user.info,syslog.info                        @loghost
#auth.info,authpriv.debug,daemon.info                  @loghost
```

Solution:

When you understand how to configure syslog from a couple of devices and has looked up which protocol and port it uses. (default is 514/udp)

Discussion:

There are syslog senders for Windows too.

Exercise 39

Create Kibana Dashboard 15min

Objective:

See Kibana and understand how it is configured.

Purpose:

Kibana is a very popular system for creating dashboards from data in elasticsearch.

Learning how to create and import dashboards is a good exercise.

Suggested method:

Revisit exercise 33 and make sure you have Elasticsearch and Kibana running.

Kibana should be available on port 5601 on localhost (127.0.0.1) only though!

Using Firefox visit Kibana on <http://127.0.0.1:5601> first time you need to select `logstash-*` as a default index. Note: Kibana is an advanced and powerful tool in itself.

Try loading the ones from: <https://github.com/StamusNetworks/KTS6>

The commands are similar to

```
git clone https://github.com/StamusNetworks/KTS6.git
cd KTS6
bash load.sh
```

Hints:

Logstash and Elastic stack are a great way to get started with dashboarding.

However, running a big installation is harder than it looks. Make sure to have multiple servers and good monitoring.

Solution:**Discussion:**

Making dashboard are an art form. We will NOT start creating beautiful dashboards.

If you want, there is a SELKS LiveCD dedicated to suricata which also includes more tools for administration of rules and getting alerts:
<https://www.stamus-networks.com/open-source/>

Exercise 40

Fun with SSH honeypots 30min

Objective:

Try a honeypot software.

Purpose:

SSH bruteforcing is a huge problem on the internet. Creating a honeypot and then blocking attackers, might prove to be a good strategy.

Suggested method:

Install and run the following software:

<https://github.com/cowrie/cowrie>

Multiple methods exist, docker or installing using the documentation:

<https://cowrie.readthedocs.io/en/latest/INSTALL.html>

Bonus: get the data inserted into Elasticsearch, see documentation link below.

Hints:

Multiple projects exist for simulating different protocols and services.

Solution:

When your team has a running honeypot and you can see the log, you are done.

Discussion:

How should you react to attacks?

Reporting attacks often dont result in any action taken, but always make sure to write a formal non-threatening email / complaint. In most cases the one receiving the complaint is NOT the attacker.

Cowrie is a fork of another honeypot Kippo.

Multiple scripts are available for processing data: Kippo-Graph, Kippo-Scripts

See for instance the information from:

- <https://cowrie.readthedocs.io/en/latest/elk/README.html>
- <https://cowrie.readthedocs.io/en/latest/graylog/README.html>
- <https://cowrie.readthedocs.io/en/latest/kippo-graph/README.html>

Exercise 41

Integrating Zeek IDS with the Elastic Stack 30min

Objective:

Integrate Zeek with Logstash, and elastic stack

Purpose:

Show how logstash filters and mutates fields from log files.

Suggested method:

Follow the instructions for reading the Zeek log files with Logstash into ES:
<https://logz.io/blog/bro-elk-part-1/>

Highly recommended to do this in multiple steps.

First, make sure it reads the data - using filebeat is one way.

Second, when data show up in ES and Kibana, notice how it is not split into fields.

Third, insert the if-statement containing bro-conn into the logstash configuration.

Hints:

Logstash make it easy to match and mutate fields, make sure the current config and real config are in sync with the names and fields.

Solution:

When your team have Zeek data to show in Kibana you are done.

Discussion:

Which log files from Zeek are most interesting to digest?

Exercise 42

Test an e-mail server 10min

Objective:

Try communicating with SMTP commands.

Purpose:

Learn how basic the protocol is.

Suggested method:

Use Kali and the telnet command line tool, if not available try nc - netcat.

```
[hlk]$ telnet mail.zencurity.com 25
Connected.
Escape character is '^]'.
220 server ESMTP Postfix
  helo test
250 server
  mail from: postmaster@pentest.dk
250 Ok
  rcpt to: root@pentest.dk
250 Ok
  data
354 End data with <CR><LF>.<CR><LF>
  skriv en kort besked
.
250 Ok: queued as 91AA34D18
quit
```

Hints:

My servr should actually not allow you to send the email, lets hope it doesn't

Solution:

When you have tried sending a few emails you are done.

Discussion:

How do we enforce new versions of protocol - as old as SMTP?!

Exercise 43

VLANs, Routing and RPF - 2h

Objective:

Configure a network and connect it to the local internet exchange. We will learn about IPv4 routing, and internet exchanges. We will also be able to configure blocking and firewall rules.

Note: it is recommended to do this in teams with two laptops. One laptop stays connected to the local wifi with internet access, for searching. The other one runs a Debian server.

The one running a Debian server will be configured as a router! YMMV but having an USB Ethernet adapter that can be dedicated and inserted into the VM will probably be the most efficient and ensure that VLAN tagging - required - are working.

You MAY try using the built-in Ethernet in your laptop for this purpose and use *bridging* to connect it to the network.

Note: the network does NOT use NAT! You can use NAT on your router, but it is recommended NOT to. If you enable NAT on your router it will be harder to connect back into your network - need to use features like port forwarding.

Purpose:

The network built for this training allows you to configure network devices without affecting production networks. You can make all the mistakes and no manager will ask you to write incident reports.

Your team will have a prefix of IPv4 /16, example 10.14.0.0/16. The same ID 14 in this example is used for other purposes too.

All your devices must end up in this network.

There are multiple networks including the other teams, all are in 10/8.

The uplink expects your connection to the rest of the networks are done with a switch using the internet exchange model with a peering network of 10.10.10.0/24. The core network will initially have a static route pointing all of your prefix to your peering address.

Your router will need an interface in this range, to be able to communicate to the internet and the other networks.

Suggested method:

Follow the instructions below for each part. Use your team and ask questions - learn.

Enable routing

Make sure routing is actually enabled on your device! Sysctl will show the current setting

Use your favourite editor and activate the line in /etc/sysctl.conf:

```
user@debian-lab:~$ grep forward /etc/sysctl.conf
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

Note: requires a reboot to pick up this change, or set using sysctl.

Changing it can be done using sysctl or echo:

```
# sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1

# echo 1 > /proc/sys/net/ipv4/ip_forward
```

Check using:

```
# sysctl net.ipv4.ip_forward
net.ipv4.ip_forward = 1
```

also install the vlan package using: apt-get install vlan

Network Cards

Before configuring your Debian as a router, make sure to remove / deactivate any network cards to your virtualisation host

You should only have the USB Ethernet configured, as a USB device in the virtual machine. Name may vary - on my server it became: enx00249b1b2991

So your configuration should be similar to this:

```
root@NMS-VM:~# ifconfig -a
```

```

enx00249b1b2991: flags=4098<BROADCAST,MULTICAST> mtu 1500
    ether 00:24:9b:1b:29:91 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 1355 bytes 556600 (543.5 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1355 bytes 556600 (543.5 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Initial steps to get started

1. Cable a laptop to any port, except the first one - port 1/0/1
2. Take control of the switch, default settings are in place, see label on bottom and use Ethernet preferably
3. Add VLAN with ID: 2770 Name: KramslX
4. Configure the Uplink port with VLAN tag: 2770
5. Configure the first port with VLAN tag: 2770
If you keep the port in VLAN 1 you should be able to connect to the switch management via this port using tagged or untagged, as you prefer ☺
6. Configure VLAN interface on your router with the peering IP in the peering LAN 10.10.10.0/24 - if your team ID is 14, then your peering IP is 10.10.10.14/24
7. Add 10.10.10.1 as your default gateway
8. Use 10.0.45.1 as DNS resolver
9. Ping 10.10.10.1 from your router

VLAN Config

VLAN ID: (2-4094, format: 2,4-5,8)
VLAN Name: (1-16 characters)

Untagged Ports

Port: (Format: 1/0/1, input or choose below)

UNIT1

LAGS

☐ Select All

1

2

3

4

5

6

7

8

9

10

Selected

Unselected

Not Available

Tagged Ports

Port: (Format: 1/0/1, input or choose below)

UNIT1

LAGS

☐ Select All

1

2

3

4

5

6

7

8

9

10

Selected

Unselected

Not Available

Cancel

Create

When your router can ping 10.10.10.1, and possibly the outside world - congratulate yourself! You have just used VLAN tagging successfully!

If your other laptop can also ping your router from the training network, awesome!

Tools to use for debugging, traceroute, nping, arp

PS The "peering LAN" uses tags to avoid the multiple switches with the same initial configuration to interfere. Only allowing VLAN tagged packets ensure only "configured devices" are let in. Real internet exchanges have lots of rules about which devices you may connect and which traffic is allowed.

Your network config should look something like this:

```
$ cat /etc/network/interfaces
# interfaces(5) file used by ifup(8) and ifdown(8)
# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d

auto enx00249b1b2991
iface enx00249b1b2991 inet static
    address 192.168.0.2
    netmask 255.255.255.0
```



```

auto vlan2770
iface vlan2770 inet static
vlan-raw-device enx00249b1b2991
address 10.10.10.14
netmask 255.255.255.0
    gateway 10.10.10.1

```

Note: you should probably also add an IP for the management of the router as shown. If you do NOT activate the base network card, the VLAN sub interfaces will not come online!

Implement your VLAN and IP plan

Now that your router is connected continue by adding your VLANs and IPs.

1. Add your VLANs using the names from your plan.
If you have no plan then use VLAN 100 LAN, VLAN 200 Wifi
2. Add ports to your VLAN - may be done while defining them too.
You may have a VLAN 100 which is intended for LAN traffic then add this as untagged on a port and tagged to the router port 1/0/1
3. Add your VLAN interfaces and IPs to the router
Again, if VLAN 100 is the LAN, add an interface to router with this tag and IP 10.14.100.1 - some prefer the first for routers, others the last .254
4. When adding the interface and port for the Wifi - make the port towards the AP untagged in your VLAN initially. The wifi APs are configured with only one SSID and will just bridged this traffic to the Ethernet port.
Using VLAN tagging for the wifi, multiple SSIDs require reconfiguration of the wifi AP - you can do this though.

Check using laptop that you can ping your router, before trying to ping other parts of the network. From the router you should also provide DHCP service - to avoid the clients needing a static IP.

If you are new to routing it may be recommended to have no firewall rules initially. If the routing parts work without trouble then enable firewall.

Advanced variants

1. Remove the default route from your router, replace by a BGP connection to the router 10.10.10.1 using BGP with default settings, no password
When the BGP connection is working, ask trainer to remove static route

2. Reverse Path Forwarding (RPF) check can be added using Access Control Lists ACLs
3. Run a DMZ with a web server and block everything except 80/tcp and 443/tcp

Hints:

Use your team, search the internet, ask for help.

This is a complex routing exercise, if you are new to routing, take it easy.

Solution:

When you have a network that allows you to surf from inside your own network and one that allows traffic from the training network into your network - you are done.

Discussion:

How much in this exercise is similar to real networks?

I would say a lot. The exercise emulates a large part of the technologies used in real networks, including layer 3 routing having a peering internet exchange and a transit provider. Also the lab networks allows for playing with layer 2 features such as port security settings.

Exercise 44

Monitoring - setup LibreNMS - 30 min

Objective:

See and work with LibreNMS - for a little while.

Purpose:

Show that initial setup of LibreNMS is quite easy, and immediately gives some results.

Suggested method:

Choose installation method from:

<https://docs.librenms.org/Installation/>

Prebuilt docker and VM images will be fastest.

Hints:

As they themselves write on the web page <https://www.librenms.org/>

LibreNMS is a fully featured network monitoring system that provides a wealth of features and device support.

If you feel this is too much work, ask for login to the central one used for the core network.

Solution:

When you have played around with LibreNMS you are done.

It is recommended to look at:

- How basic information about devices are presented, from devices when added - and nothing more.
See how to add a device and add your own. <https://docs.librenms.org/Support/Adding-a-Device/>
- How SNMP location is used to categorize devices and provide maps, see <https://docs.librenms.org/Extensions/World-Map/>
- How protocols like LLDP allow LibreNMS to make maps, see <https://docs.librenms.org/Extensions/Network-Map/>
- How port description can be used for describing ports, <https://docs.librenms.org/Extensions/Interface-Description-Parsing/>

Most of this happens with very little effort. Just configure devices consistently and they will be presented nicely.

Discussion:

Have you configured port description? Try it!

Real networks have policies for port description settings.

Exercise 45

IDS with Zeek and Suricata - 30min

Objective:

Add your Suricata and Zeek server to the network, use it to monitor your traffic.

Purpose:

Show how an IDS box is added to a network.

Suggested method:

Configure a mirror port - select a high numbered port not in use.

Then configure source ports, the ones in current use and destination to the selected high port.

If using the TP-Link T1500G-10PS then this link should describe the process:

https://www.tp-link.com/en/configuration-guides/mirroring_traffic/?configurationId=18210

Which describe:

1. Choose the menu MAINTENANCE > Mirroring
2. Select Edit for the Mirror Session 1
3. In the Destination Port Config section, specify a destination port for the mirroring session, and click Apply
4. In the Source Interfaces Config section, specify the source interfaces and click Apply

Using the command line would be similar to this:

```
Switch#configure
Switch(config)#monitor session 1 destination interface gigabitEthernet 1/0/7
Switch(config)#monitor session 1 source interface gigabitEthernet 1/0/1-4 both
Switch(config)#monitor session 1 source cpu 1 both
```

Hints:

Selecting a port away from the existing ones allow easy configuration of the source to be like, Source ports 1-4 and destination 7 - and easily expanded when port 5 and 6 are activated.

Solution:

When your team has configured a mirror port and seen traffic using your IDS or just tcpdump you are done.

Discussion:

How would you monitor a larger network?

Cisco has a feature named RSPAN

Remote SPAN (RSPAN): An extension of SPAN called remote SPAN or RSPAN. RSPAN allows you to monitor traffic from source ports distributed over multiple switches, which means that you can centralize your network capture devices. RSPAN works by mirroring the traffic from the source ports of an RSPAN session onto a VLAN that is dedicated for the RSPAN session. This VLAN is then trunked to other switches, allowing the RSPAN session traffic to be transported across multiple switches. On the switch that contains the destination port for the session, traffic from the RSPAN session VLAN is simply mirrored out the destination port.

Source: <https://community.cisco.com/t5/networking-documents/understanding-span-rspan-and-erspan/ta-p/3144951>

Exercise 46

Configure port security - 30 min

Objective:

See the options available for port security on the local switch

Purpose:

Even small cheap switches like the ones used in the training have options that can help protect a network. Such as:

- Max Learned Number of MAC

Specify the maximum number of MAC addresses that can be learned on the port. When the learned MAC address number reaches the limit, the port will stop learning. It ranges from 0 to 64. The default value is 64.

- Exceed Max Learned Trap

Enable Exceed Max Learned, and when the maximum number of learned MAC addresses on the specified port is exceeded, a notification will be generated and sent to the management host.

- Learn Address Mode

Select the learn mode of the MAC addresses on the port. Three modes are provided:

Delete on Timeout: The switch will delete the MAC addresses that are not used or updated within the aging time. It is the default setting.

Delete on Reboot: The learned MAC addresses are out of the influence of the aging time and can only be deleted manually. The learned entries will be cleared after the switch is rebooted.

Permanent: The learned MAC addresses are out of the influence of the aging time and can only be deleted manually. The learned entries will be saved even the switch is rebooted.

- Status when exceeded

Drop: When the number of learned MAC addresses reaches the limit, the port will stop learning and discard the packets with the MAC addresses that have not been learned.

Forward: When the number of learned MAC addresses reaches the limit, the port will stop learning but send the packets with the MAC addresses that have not been learned.

Suggested method:

Visit the settings page, configure port security and try running the `macof` mac overflow tool from a laptop connected on that port.

Hints:

There may be a configuration guide at:

https://www.tp-link.com/en/configuration-guides/configuring_port_security/?configurationId=18227

Solution:

When you have seen the setting, either on the available devices, or searched the internet and found the options available on your favourite device, you are done.

Discussion:

Port security is often neglected, so a small cabling issue in a single device in an office may take down the whole office location network.

Think about port security when installing new networks, and turn it on immediately before allowing people to connect, as most managers are wary about enabling it afterwards.