SIEM and Log Analysis exercises

Henrik Kramselund Jereminsen
hlk@zencurity.com

November 18, 2020

Contents

Preface

This material is prepared for use in *SIEM and Log Analysis course* and was prepared by Henrik Lund Kramshoej, 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 siem-log-analysis-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 Administrator's 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 Administrator's 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.

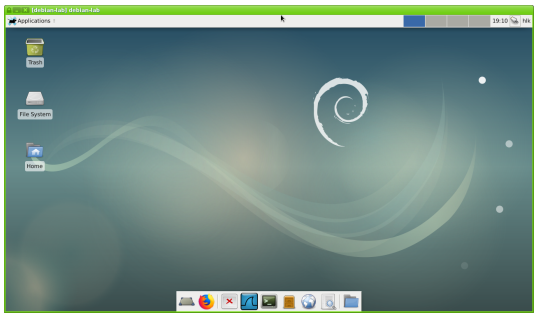
Debian Linux is a free operating system platform.

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 VM 10 min



Objective:
Make sure your virtual machine is in working order.

We need a Debian 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 Debian VM.

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

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

I suggest 50G disk, 2CPU cores and 6Gb memory for this course, if you have this.

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.

Debian Linux allows us to run Ansible and provision a whole SIEM in very few minutes.

Exercise 3

Investigate /etc 10 min

Objective:
We will investigate the /etc directory on Linux. We need a Debian Linux

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.

```
user@debian:~$ cd /etc
user@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
user@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:/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.

Also note the difference between running as root and normal user. Usually books and instructions will use a prompt of hash mark # when the root user is assumed and dollar sign $ when a normal user prompt.

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

Enable UFW firewall

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`

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

Exercise 5

Data types – IP addresses 15min

Objective:
Find out what IP-addresses really are – just a 32-bit integer for IPv4

Purpose:

Suggested method:

Hints:
Use CIDR tool online or locally

See that libraries and tools exist for IP addresses

Example, look at PostgreSQL data types

Solution:

Discussion:
Compare the bytes needed for storing a string `192.0.2.10` versus the bytes needed for a 32-bit integer.

Exercise 6

Postman API Client 20 min

Objective:
Get a program capable of sending REST HTTP calls installed.

Purpose:
Debugging REST is often needed, and some tools like Elasticsearch is both configured and maintained using REST APIs.

Suggested method:
Download the app from https://www.postman.com/downloads/

Available for Windows, Mac and Linux.

Hints:
You can run the application without signing in anywhere.

Solution:
When you have performed a REST call from within this tool, you are done.

Example: use the fake site https://jsonplaceholder.typicode.com/todos/1 and other similar methods from the same (fake) REST API

If you have Elasticsearch installed and running try: http://127.0.0.1:9200

Discussion:
Multiple applications and plugins can perform similar functions. This is a standalone app.

Tools like Elasticsearch has plugins allowing decoupling of the API and plugins. Example: https://www.elastic.co/what-is/elasticsearch-monitoring and https://www.elastic.co/what-is/open-x-pack

Exercise 7

Use Ansible to install Elastic Stack

Objective:
Run Elasticsearch

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

Suggested method:
We will run Elasticsearch, either using the method from:
`https://www.elastic.co/guide/en/elastic-stack-get-started/current/get-started-elastic-stack.html`

or by the method described below using Ansible - your choice.

Ansible used below is a configuration management tool `https://www.ansible.com/`

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
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 3-elasticstack.yml
```

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

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

Solution:
When you have a updated VM and Ansible running, 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 8

Getting started with the Elastic Stack - 60 min

Objective:
Get a working Elasticsearch, so we can do requests.

Purpose:
Elasticsearch uses REST extensively in their application.

Suggested method:
either use the *Getting started with the Elastic Stack* https://www.elastic.co/guide/en/elastic-stack-get-started/current/get-started-elastic-stack.html

OR my Ansible based approach - which some already ran.

The ansible is described in exercise 7 on 10

Hints:
We dont really need a lot in the Elasticsearch database, and you can run most tasks with zero data. Graphs will not be as pretty though.

Solution:
When you have a running Elasticsearch you are done, and ready for next exercise.

The web page for the getting started show multiple sections:

- Elasticsearch - the core engine, this must be done manually or with Ansible
- Kibana - the analytics and visualization platform
- Beats - data shippers, a way to get some data into ES
- Logstash (optional) offers a large selection of plugins to help you parse, enrich, transform, and buffer data from a variety of sources

Each describes a part and are recommended reading.

Discussion:
We could have used a lot of other servers and service, which ones would you prefer?

If you have access to Azure, you can try Azure REST API Reference
https://docs.microsoft.com/en-us/rest/api/azure/

Exercise 9

Making requests to Elasticsearch - 15-75min

Objective:
Use APIs for accessing Elasticsearch data, both internal and user data.

Purpose:
Learn how to make requests to an API.

Suggested method:
Go to the list of exposed Elasticsearch REST APIs:
https://www.elastic.co/guide/en/elasticsearch/reference/current/rest-apis.html

The Elasticsearch REST APIs are exposed using JSON over HTTP.

Select a category example, Cluster APIs, then select Nodes Info APIs. This will show URLs you can use:

```
# return just process
curl -X GET "localhost:9200/\_nodes/process?pretty"
# same as above
curl -X GET "localhost:9200/\_nodes/_all/process?pretty"

curl -X GET "localhost:9200/_nodes/plugins?pretty"

# return just jvm and process of only nodeId1 and nodeId2
curl -X GET "localhost:9200/\_nodes/nodeId1,nodeId2/jvm,process?pretty"
# same as above
curl -X GET "localhost:9200/\_nodes/nodeId1,nodeId2/info/jvm,process?pretty"
# return all the information of only nodeId1 and nodeId2
curl -X GET "localhost:9200/\_nodes/nodeId1,nodeId2/_all?pretty"
```

When you can see this works, then feel free to install X-Pack and monitoring plugins

Hints:
Pretty Results can be obtained using the pretty parameter.

> When appending ?pretty=true to any request made, the JSON returned will be pretty formatted (use it for debugging only!). Another option is to set ?format=yaml which will cause the result to be returned in the (sometimes) more readable yaml format.

Lots of tutorials exist for accessing Elasticsearch

A couple of examples:

- https://aws.amazon.com/blogs/database/elasticsearch-tutorial-a-quick-start-guide/
- https://www.digitalocean.com/community/tutorials/how-to-install-elasticsearch-logstash-and-kibana-elastic-stack-on-ubuntu-18-04

Solution:
When you have seen examples of the API, understand the references with underscore, like `_nodes` and pretty printing you are done.

I recommend playing with Elasticsearch plugins and X-pack.
https://www.elastic.co/downloads/x-pack

Note: In versions 6.3 and later, X-Pack is included with the default distributions of Elastic Stack, with all free features enabled by default.

Also Kibana can be used for creating nice dashboards and become applications more or less.

Discussion:
You can also try calling the REST API from Python

Similar to what we did previously in this course:

```
#!/usr/bin/env python
import requests
r = requests.get('https://api.github.com/events')
print (r.json());
```


Exercise 10

Use a XML library in Python up to 60min

Objective:
Try using a programing library in the Python programming language.

Purpose:
See how easy it is to produce functionality by re-using existing functions and features available in a popular language.

Suggested method:
Start by getting an XML file. Suggested method is to boot your Linux and run a command like `nmap -p 80,443 -A -oA testfile www.zencurity.com`. Output should be testfile.xml and two other files, grepable output `testfile.gnmap` and text output `testfile.nmap`.

Then using Python import a library to parse XML and print a few values from the XML, or all of them.

Recommended values to print from the file:

- Nmap version
- Date of the Nmap run, note either use start and convert from Unix time or startstr which is a string
- Nmaprun args - aka the command line
- Host address
- Ports like from the <port protocol="tcp" portid="443">
- Anything you feel like

Hints:
One option is to use the Python ElementTree XML API:
`https://docs.python.org/2/library/xml.etree.elementtree.html`

Also - use Python3!

Solution:
When you can read a file and process it using Python3.

Improvements, you might consider:

- Use Python3 to run the Nmap process

- Create command line parameters for the program, making it more useful
- Pretty print using formatted output

Discussion:
Many examples contain code like this:

> Getting child tag's attribute value in a XML using ElementTree
>
> Parse the XML file and get the root tag and then using [0] will give us first child tag. Similarly [1], [2] gives us subsequent child tags. After getting child tag use `.attrib[attribute_name]` to get value of that attribute.

```
>>> import xml.etree.ElementTree as ET
>>> xmlstr = '<foo><bar key="value">text</bar></foo>'
>>> root = ET.fromstring(xmlstr)
>>> root.tag
'foo'
>>> root[0].tag
'bar'
>>> root[0].attrib['key']
'value'
```

Source:
https://stackoverflow.com/questions/4573237/how-to-extract-xml-attribute-using-python-elementtree

What is the point of referring to a specific numbered child, when we specifically have the tags?!

What happens if the XML output changes a bit, so another tag is before the expected one! Dont trust Stackoverflow, unless you want a stack overflow ☺.

