Security Lab – SIEM

Preparation

For this lab, you should be able to use almost any x86-64 Debian-based VM, such as our cloud-based VMs. Any Debian-based VM image should also work, such as stock Ubuntu or Debian. The remainder of this document assumes you are working with this VM. If you use our cloud-based VMs, there should already be a directory *siem/* in your home directory. In this case, you can skip the remainder of this section and can continue to Section 1 below. **This lab will (probably) not work on anything that's not (emulating) an x86 CPU**. If you have an ARM CPU and feel adventurous, you can try running this lab on an ARM-based Debian-based VM, but we can give no support if it doesn't work.

If you do not already have a directory *siem/* in your home directory, you need to unpack the files for this lab first. You should have access to a file called *siem.tar.gz*. Copy this to your home directory and then unpack this file using

```
tar xzvf siem.tar.gz
```

Also check if you have ansible installed using

```
apt install ansible
```

1 Introduction

In this lab you will learn how to use a SIEM platform using Splunk as an example. The goal is to get familiarize how a SIEM operates and how to use search queries to find important events in a network. Furthermore, the aim is to interpret alerts in the context of the cyber kill chain and to be able to understand a cyber attack in its entirety.

Important hint: This is a long lab, and you have three weeks to work on it. We know from previous courses that in 3-week labs, the first two weeks are usually spent doing nothing and then the last week is a frantic flurry of activity. **Don't do that.** Instead, work on the lab up to and including task 3 in the first week, do Task 4 and parts of Task 5 in the second week and the remainder in the third week. You can show completed tasks to your instructor and you will get partial points.

2 Basis for the Lab

The lab will be done with a freshly installed Splunk Enterprise instance which runs in the 60-day trial period. It can happen that you will see a prompt asking you to take a tour of different functions which Splunk offers. Please feel free to watch the tours but they can also just be ignored.

Most of the events are generated by an app called Eventgen and aren't created by any systems directly. The aim is for the events to be as realistic as possible, but it may be that certain events are a bit strange. It is advisable not to get too caught up in detailed analysis.

2.1 Installation

The installation is automated with Ansible[1] and can simply be started via the console.

1. Open the console.
2. Move to project folder by typing:

   ```
   cd siem
   ```

3. Start the installation of Splunk by using:

   ```
   sudo ansible-playbook -i inventory.yml deploy_splunk.yml
   ```

4. The installation needs to disable a system feature called «transparent huge pages». Check if this has worked by typing

[1] https://www.ansible.com/

```
cat /sys/kernel/mm/transparent_hugepage/enabled
cat /sys/kernel/mm/transparent_hugepage/defrag
```

The output should in both cases end in «[never]». If that is not the case, type

```
sudo systemctl start disable-thp
```

and check again.

5. Open the Splunk instance by typing into a browser: https://localhost:8000. Splunk uses a self-signed certificate, so swat away your browser's warning dialog.
6. Use the following login:
   a. **Username:** student
   b. **Password:** student1234

3 Task

Congratulations, you're the newest member of the ZHAW Security Operations Center (SOC) Team. Your tasks are to maintain and improve the existing security information and event management platform. ZHAW uses Splunk as their SIEM-Platform.

ZHAW issued a press release that a research group stands on the brink of a revolution for Artificial Intelligence with a new cutting-edge procedure and will soon present their findings.

Your team leader comes to you and tells you that the number of ports scans on the firewall tripled since the press release. He asks you to check the configuration of the SIEM-Platform and make sure everything gets monitored. Your predecessor left a list of To-Do tasks which you decide to take care of.

3.1 Task - Monitor new Account created

It seems your predecessor never found the time to setup the monitoring of some local logs on the Splunk-Server. We want to change that and start monitoring "/var/logs/auth.log", which records all successful or failed logins and shows if a new user was created.

First, we must allow the Splunk-User to read the logs located in "/var/logs". We give the permission by entering:

```
sudo setfacl -R -m u:splunk:rX /var/log
```

in a terminal.

> **Author's note:**
>
> We could also achieve this by giving the Splunk-User root access, but this isn't recommended. For further information read about the **Fundamental Security Principles** especially the **Principle of Least Privilege** (covered in SWS1).

You can check if the permission were set with:

```
getfacl -R /var/log
```

When this is setup, we can start monitoring the log-file in Splunk.

First login over a browser to your Splunk-Instance:

1. On the Start-Page under Settings (top right) click on **Add data.**
2. Choose **Monitor** in the second square.
3. Choose **Files & Directories** and type in the Field «/var/log/auth.log» and choose *Continuously Monitor* and click **Next.**

With “Continuously Monitor” you can actively monitor a file or directory that are local to Splunk, such as various log files or even the entire log folder “var/log”. In a production environment you would collect logs from remote machines as well using Splunk agents.

Archived log files can be uploaded to Splunk with “Index Once”. These can be analysed with Splunk for a forensic investigation, for example.

4. On the left side, in “Set Source Type” choose **Operating System → linux_secure** and click **Next.**

With "Set Source Type" we configure how the imported logs are parsed in Splunk. Splunk has various ready-made templates for different log types, but also allows you to create your own templates. Certainly, an interesting task for people who like to use regex. For this task, however, we will use a ready-made Source Type.

5. For “Input Setting”, click on **Create a new index.**

   Configure the new Index as follow:

   - **Index Name:** splunk
   - **Max Size of Entire Index:** 5000 MB

   Click on Save when done.

   In the Dropdown menu, choose the newly created Index splunk.

Indexes[2] are repositories in which the parsed log data, now converted into searchable events, are stored. The events are saved in flat files on the indexer in so-called buckets. Therefore, no third-party database is required for the indexes.

6. After the Data Input is created click “Review” and “Submit” and finally on **Start Searching.**

⚠ It can take a couple of seconds until the logs appear.

Question: Looking through the log file. Find the three admin-users created on the system in the log. Adjust the search query if necessary.

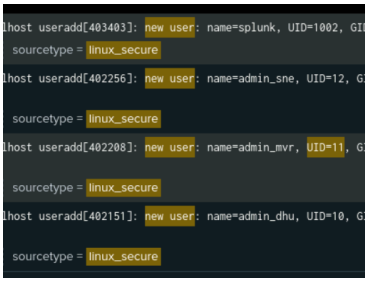
admin_dhu,
admin_mvr,
admin_sne

We want to trigger an alarm every time somebody creates a new user on the system. Adjust the search query to:

```
source=”/var/log/auth.log” host=”<hostname>” index=”splunk”
sourcetype=”linux_secure” ”new user”
```

You should see a list of new created users. Now we want to save this search query as an alarm. Every time a new user is created, we want to be informed by Splunk about it.

On the top right click on «Save As» → Alert.



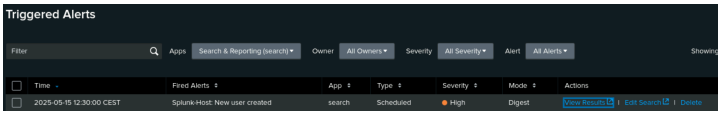
Fill out the form as follows, leave the default value on not mentioned fields:

- **Title**: Splunk-Host: New user created
- **Alert type:** Scheduled

  Run on Cron Schedule

[2] Find more infos here: https://t.ly/MJ7fn

- **Time Range:** *Relative:*
  Earliest: **5 Minutes Ago, Beginning of minute**
  Latest: **Beginning of minute**
- **Cron Expression:** */5 * * * *
- **Throttle:** checked
- **Suppress triggering for:** 5 minute(s)
- **Trigger Actions:** Add to Triggered Alerts → Severity: **High**



Save the configuration and press “View Alert”.

So, what exactly have we configured? We have created an alert that checks every 5 minutes whether a new user has been created in our log file "/var/log/auth.log". We check this by searching for the term "new user" in the log file. We have also set a throttle value of 5 minutes to ensure that the same entries do not trigger the same alarm twice.

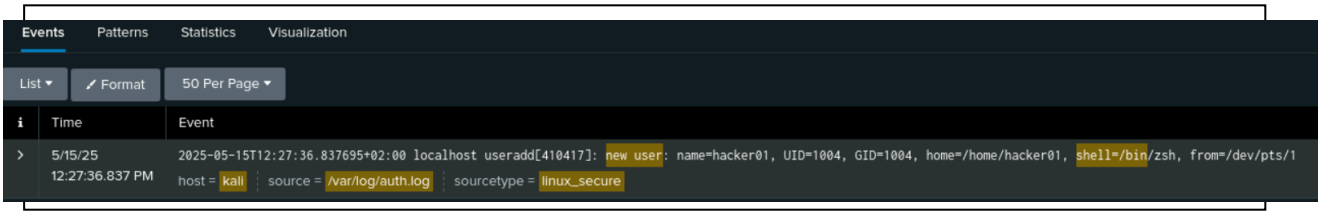
It would also be possible to monitor a log in real time. However, this is not recommended for performance reasons, as otherwise this task would occupy a core on the CPU all the time.

Let’s see if this alarm is working, shall we? Go back to the console and create a new user by entering:

```
sudo adduser --gecos “” --disabled-password hacker01
```

You can find the triggered alerts in the top bar under Activity → Triggered Alerts. If everything was done correctly, you should see an alert. **It can take up to 5 minutes until an alert appear because of the alert Cron schedule**.

Question: Make a screenshot of the triggered alert.



3.2 Splunk up your life...

... as sung by the Splunk Girls. Joke aside, here's a quick time-out of our story. We've dipped our toe in the deep end and (probably) used Splunk for the first time. To give you a little bit of a leg up for the following tasks, we will give you a little crash course in Splunk Syntax.

For all the following tasks, you will only use the "Search & Reporting" app. All necessary search queries can be entered and saved there. You will also find all Reports, Alerts and Dashboards there.

As already mentioned, events are stored in indexes in Splunk. Indexes can be searched using the command:

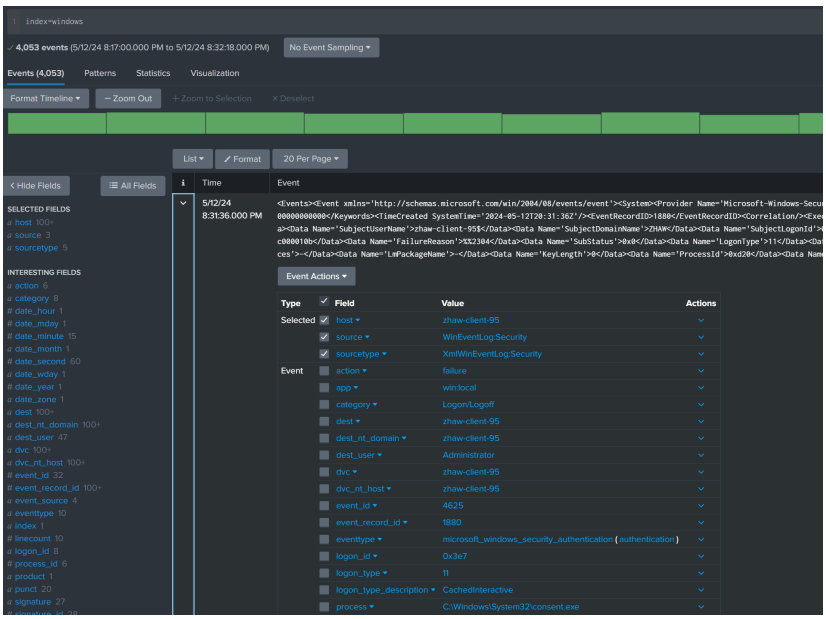
```
index=”<index_name>”
```

Splunk also has internal indexes that start with an underscore (like «_audit»).

These indexes will be relevant for this lab:

- firewall
- proxy
- windows
- mail

Splunk uses regex to generate fields from incoming events. These regex fields are defined in a configuration file. The contents of these fields can then be used in a query. It is possible to display these fields from an event directly in Splunk.



In the "Search & Reporting"-app, click on the arrow in the column with the **i**-sign and all the specific fields for this event will be displayed under Events. This will be important for the upcoming tasks.

Splunk uses SPL2 (Search Processing Language version 2) for search inputs. SPL2 is bimodal and supports both SPL and SQL search input. So good news for all SQL aficionados among you. All you need to do is write an SQL search query and get the same result as with SPL. Maybe it would be a good thing to try and remember back to the database module anyway.

Let's have a gander at a long and scary looking search query and break it down so we can see how this works.

We would recommend putting line after line into Splunk to see how each additional line changes the output of Splunk.

```
index=firewall TCP "fw_name=firewall-01" AND (src_ip="192.168.*" OR src_ip="10.*") AND NOT dst_port=443
| stats count, avg(bytes) as avg_bytes, values(action) as actions, values(src_port) by src_ip, dst_ip, dst_port
| rename values(src_port) as src_ports
| sort - count
| dedup src_ip
| head 100
| eval avg_bytes = round(avg_bytes, 2), note=if(dst_port>1024, "high dst port", "low dst port"), actions = lower(actions)
| fields - count
| where avg_bytes>50000
```

We are going through the query line after line:

```
source="/var/log/auth.log" host="kali" index="splunk" sourcetype="linux_secure" "new user"
```

```
① index=firewall TCP "fw_name=firewall-01" AND (src_ip="192.168.*" OR src_ip="10.*") AND NOT dst_port=443
② | stats count, avg(bytes) as avg_bytes, values(action) as actions, values(src_port) by src_ip, dst_ip, dst_port
③ | rename values(src_port) as src_ports
④ | sort - count
⑤ | dedup src_ip
⑥ | head 100
⑦ | eval avg_bytes = round(avg_bytes, 2), note=if(dst_port>1024, "high dst port", "low dst port"), actions = lower(actions)
⑧ | fields - count
⑨ | where avg_bytes>50000
```

1. Specifies the “index” from which to retrieve “firewall” events. The keyword “TCP” filters only for events which match that string. The following statements will filter the events to include the ones where the field value of “fw_name” is “firewall-01”, the field value of “src_ip” is matching “192.168.*” or “10.*” and the “dst_port” is not “443”. The asterisk “*” in Splunk is used as a classic wildcard and matches any character or characters. Looking at performance, asterisks should whenever possibly used at the end of a term and not in the beginning. Even more important, they should not be used in the middle of a term because this can provide inconsistent results.
2. Generates statistics for each combination of “src_ip”, “dst_ip” and “dst_port”, including count of events, average of the value within the “bytes” field, list of unique values for the “action” field and a list of unique values for the “src_port” field. The “bytes” field gets renamed as “avg_bytes” and “action” as “actions”.
3. Renames the field “values(src_port)” as “src_ports”.
4. With “sort and the “-“, the result will be set descending order based on the value of the “count” field. Leave “-“ away for ascending order.
5. Removes duplicate events based on the field “src_ip”. It keeps only the first occurrence (top to down) of a “src_ip” value within the table.
6. Keeps only the first 100 events respectively table rows.
7. Evaluates the expression and limits the value of “avg_bytes” to 2 decimal places. Furthermore, it creates a new field “note” where the value depends on if the “dst_port” value is greater than 1024 or not. If the value is greater the note field will have the value “high dst port” otherwise “low dst port”. Lastly, it lowers the letters of the action field values.
8. With “fields” and “-“ the “count” field will be removed from the result set.
9. Filters the result set to only include those where “avg_bytes” is greater than “50000”.

As an additional overview you can find here some important search terms and filters to find the needed information.

Syntax	**Example**	**Description**
index=<index>	index=”main”	Find events in a specific index.
stats <function1> by <field1>	stats count by src_ip	Generate statistics with a function grouped by fields.
rename <oldN> as <newN>	rename src_ip as source_ip	Rename a given field.
sort <order_sign> <field>	sort - count	Sorts the specified columns ascending (+) or descending (-).
dedup <field1>	dedup host, src_ip, port	Deduplicates events that contain the same combination of values for the specified fields.
head <number>	head 100	Shows only the latest 100

		events in the defined time range.
eval <newF>=<expression>	eval new_field=round(bytes, 2)	Evaluates an expression or function and stores the result within a new specified field.
fields <sign> <field1>	fields - host, scr_ip	Specifies which fields to keep (+) or remove (-) from the search results.
where <expression>	where name=”mike”	Filters events based on the expression and keeps only the ones which return true in the result set.
search <expression>	search “TCP” AND src_ip=”192.*”	Filters events based on keywords or field values and supports wildcards.
rex field=<field> “<regex>”	rex field=_raw “(?<newField>\w+)\d”	Extracts new fields within the existing events or fields based on a regular expression.
regex	regex “(\d+\.){3}\d+”	Keep only events which match the specified regular expression in the result set.
lookup <lookup_name> <lookupF> as <eventF> OUTPUTNEW <lookupF> as <eventF>	lookup threat_intel src_ip as ip OUTPUTNEW confidence as ioc_confidence	Populates search result events with information stored in external files (lookups) based on field values which work as key.
table <field1> <field2>	table host, src_ip	Tables the specified fields from an event into a table structure.
timechart <funktion> by <field>	timechart count by src_ip	Evaluates the function per specified field value of the BY-clause and aggregates the results over time.
xmlkv	xmlkv	Extracts fields and field values from an XML log structure at search time.
bin <field> span=<binSize>	bin _time span=15m	Converts continuous numbers into buckets of the specified bin size. All numbers within a bucket have the same value.

For further reference check the links here:

- SPL2 Search Reference: https://docs.splunk.com/Documentation/Splunk/latest/SearchReference/WhatsInThisManual

- Splunk EVAL expressions and functions:
  https://docs.splunk.com/Documentation/Splunk/latest/SearchReference/CommonEvalFunctions

3.3 Creating an Alert in Splunk

We have already created an alert in task 3.1. In the following tasks, we will create queries and save them as alerts. These alerts will be used in the next chapters of the lab. As these alerts will always have the same settings but they will be slightly different to our first alert, we will briefly go through the configuration here.

When you have wrote the search query you can save it by clicking “Save As” on the right top corner and choose “Alert”.

```
index=splunk source="/var/log/auth.log"
sourcetype="linux_secure" "new user"
```

Choose the following settings:

- **Title**: Choose your own title
- **Alert type:** Scheduled
  Run on Cron Schedule
- **Time Range:** *Advanced:*
  Earliest: **-10m@m**
  Latest: **-5m@m**
- **Cron Expression:** */5 * * * *
- **Throttle:** checked
- **Suppress triggering for:** 5 minute(s)
- **Trigger Actions:** Add to Triggered Alerts → Severity: **High**

Please ignore the Trial Warning.

You may be wondering why our time range is in the past. The reason for this is that when the events are generated, they are not immediately in the Splunk system and if a rule then searches over the last 5 minutes, it may be that the event is already older and therefore the alarm is not triggered if many events reach the system. The search window is therefore moved further back to ensure that all relevant events are found and the alarm is triggered reliably.

3.4 Task - New Scheduled Task created

The use of Scheduled Task is a popular method for hackers to establish a persistent connection to their own Command & Control Server. The system that was compromised during the initial access phase should stay accessible to the attackers even after a restart.

There are different ways to create a Scheduled Task. On Linux this can be achieved with a "cronjob" or a "systemd timer". On Windows, the Scheduled Task Service is responsible.

When working in a SOC, it is important to know the different Windows event IDs and to assess which are important and which are not in the context of cyber security.

Question: Find and note the needed event ID to monitor if a scheduled task is created.
Hint: You can look up event ids here:

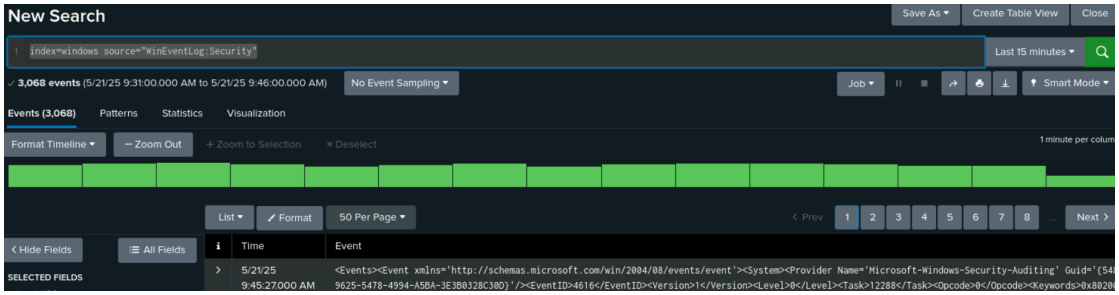
- https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/default.aspx?i=j

Windows	4698	A scheduled task was created

Create an overview to see what Windows Security events are in Splunk. Limit the search query to the source “WinEventLog:Security”.

Question: Build the query to find the events. Note your query and the found events here:

```
index=windows source="WinEventLog:Security"
```



Create the query to display when a scheduled task has been created. The output should be in table format with the columns *host, event id, signature, Author, Command* and *Enabled*. To do this, use the event ID found in the previous task:

Hints:

- The event will be in xml format. Find a command to extract the xml fields and field value.
- You can test your query with other events which are in xml format.
- If no events are available, similar alternative events can also be used to create the query.

```
index=* source="WinEventLog:Security" EventCode=4698
| xmlkv
| table host, EventCode, signature, Author, Command, Enabled
```

```
index=* source="WinEventLog:Security"
| stats count by EventCode
| sort - count
```

Save your query as an alert as described in chapter 3.3.

Note: Normally, baselining would also be necessary here to keep the “False-Positive”-rate low. There are admin accounts that are probably authorised to create a scheduled task. Also, there are legitimate paths that can be used within a scheduled task. When whitelisting such correlation rules, however, care should be taken to ensure that new blind spots are not introduced. For example, an admin account with authorisation to create schedule tasks can also be taken over by an attacker.

3.5 Task - Port Scanning Rule

The next item on the To-Do list is to recognize port scanning in the LAN. Attackers try to detect vulnerable services in the network using simple port scans, such as ping or vanilla scans.

Ping-Scans use the ICMP protocol to identify systems in the network. ICMP pings are normally logged by a firewall. However, this depends on the respective configuration of the firewall.

```
index=firewall sourcetype=eventgen:firewall:kv
(src_ip="10.*" OR src_ip="172.16.*" OR src_ip="192.168.*")
(dst_ip="10.*" OR dst_ip="172.16.*" OR dst_ip="192.168.*")
| stats dc(dst_port) as unique_ports values(dst_port) as ports
values(action) as actions values(protocol) as protocols
sum(bytes) as total_bytes by src_ip, dst_ip
| where unique_ports > 20
| sort - unique_ports
```

The Vanilla-Scan is a little more sophisticated. The Vanilla-Scan attempts to connect to all 65,536 ports of a system. In a normal TCP Three-way handshake[3], a SYN flag is sent to the server and if the client receives a SYN-ACK response, it replies with an ACK. The handshake is done, and the TCP-Connection is established. An attacker uses this to learn which ports are available and open and which are not. One way for attackers to remain undetected would be, for example, not to respond with an ACK after the SYN-ACK. Firewalls normally only log fully initialized connections.

Other port scan options would be XMAS/FIN, FTP bounce and Sweep Scan, which are more difficult to detect and aren't part of this lab.

We want to concentrate on discovering Vanilla-Scans. The problem is that in our network, we cannot distinguish between legitimate and malicious TCP handshakes when we look at them individually. We therefore need to look at the TCP handshakes in correlation. We assume that it is unlikely that a system would normally try to access more than 20 ports in a 15-minute time frame of another system. We try to detect these anomalies in our network with Splunk.
Note: It could be that a network has services running which legitimately do Port-Scans on the network like Nessus. Such servers would need to be whitelisted by the rule.

We now want to build the query in Splunk. For that open the "Search & Reporting"-App in Splunk.

Here some **hints** how to build the query:

- Choose the firewall index and define the specific host and sourcetype.
- We only searching for Vanilla-Scans inside our network so only choose the local networks. The local networks are 10.0.0.0/8, 172.16.0.0/12 and 192.168.0.0/16. The source and destination IP-Address should be in these ranges.
- We need a statistic for how many unique destination ports from a destination IP Address were targeted by a source IP Address. Also, we want to see which destination ports were contacted, which action was taken (ACCEPT or DENY), which protocol were used (TCP or UDP) and the sum of bytes which were transferred.
- Filter the results that we only see events were more than 20 unique destination ports were contacted.

Question: Note your final query to find Vanilla-Scans:

New Search

```
index=firewall sourcetype=eventgen:firewall:kv
(src_ip="10.*" OR src_ip="172.16.*" OR src_ip="192.168.*")
(dst_ip="10.*" OR dst_ip="172.16.*" OR dst_ip="192.168.*")
| stats dc(dst_port) as unique_ports values(dst_port) as ports values(action) as actions values(protocol) as protocols sum(bytes) as total_bytes by src_ip, dst_ip
| where unique_ports > 20
| sort - unique_ports
```

Save your query as an alert as described in chapter 3.3.

3.6 Task - Brute-Force and Password Spraying

Attackers try to gain access to accounts where the password is unknown or where only password hashes are currently available. Methods like OS credential dumping would be one possibility, another would be a password brute force. Password brute force involves repetitive or iterative attempts to obtain the password of an account, for example using a dictionary attack. It should be noted that this can be done both offline if the password hashes are known or online against a live system.

There are various forms of brute force attacks. *Password Guessing* is probably the simplest, in which, a list of the most frequently used passwords is taken and login attempts are made against an account to see if that account uses any of these passwords. Another method is *Password Cracking*, which involves comparing a pre-compiled list of passwords with password hashes that have been found in a system. To counteract this, passwords should always have a different *salt*[4] value. This already in-

[3] As discussed in module KT
[4] Discussed in ITS

creases the hurdle for an attacker enormously, as the salt is also required to create a pre-compiled password list. *Credential Stuffing* is a way of using already known credential details to gain access to an account from the same user on a different platform. It is not uncommon for users to use the same password for several services or even to use the same password for most of their user accounts.

One variant of a brute force attack is *Password Spraying*. This involves checking one or more passwords for several accounts. This is particularly successful when accounts use simple passwords and is also an effective tactic as it is easy to execute.

We want to configure a correlation rule. For that, we need to monitor the authentication events. We will limit ourselves here to Windows events, but this would also apply to Linux systems or application logins.

Question: Find the Windows events that show whether a login attempt has succeeded or failed. Note the event IDs:

Windows 4624 An account was successfully logged on
Windows 4625 An account failed to log on

To be able to monitor a *Password Spraying*-attack, we need to make the following assumption:

In our network, it is highly unlikely ***for more than 10 different users*** *to try to log in from* ***one and the same IP Adress*** *in a* ***period of 15 minutes****.*

Please note that this may be different in other environments.

To create the query, we need the stats function, which we have already used in the previous tasks. Among other things, use the functions *distinct count* (dc(<field>) and *list* (list(<field>). The field *_time* is also required. Find a function that allows you to divide time spans into *buckets*.

Question: Write the query based on our assumption and note it here:

```
index=* source="WinEventLog:Security" sourcetype="XmlWinEventLog:Security"
"EventID>4624<"
                    index=* source="WinEventLog:Security" EventCode=4625
                    | bin _time span=5m
                    | stats dc(Account_Name) as unique_users, list(Account_Name)
                    as users by _time, src_ip
                    | where unique_users > 3
```

And once again save the query as an alert as described in chapter 3.3.

3.7 Task - Detection of threat activity and data exfiltration

With Command & Control, attackers attempt to communicate with the systems under their control. Attackers try to simulate normal traffic to evade detection. To do this, they use application layer protocols such as DNS, file transfer, mail or web. With HTTP(S), data can be hidden in headers, for example, while normal traffic is simulated at the same time. The command & control channel can also be used for data exfiltration. The attackers steal sensitive and valuable data, which is then offered for sale.

The detection of threat activity in a SIEM can be achieved with the correlation of so-called threat intelligence feeds. Various providers on the market compile such feeds, which summarize known attacker domains, IPs and hashes, for example. This data is then integrated into a SIEM and correlated across all available log sources where possible.

This means, for example, that all log data in which an IP is present is correlated with IP addresses from the threat intelligence feed. If an IP address from the feed appears in the logs, a corresponding alarm is generated, which must be checked for legitimacy by a SOC employee.

To prevent the task from becoming too complex, we limit ourselves to a domain threat intelligence feed. The feed can be viewed in Splunk with the following command in the "Search & Reporting"-App:

```
| inputlookup threat_intel
```

Here is a brief description of the individual columns:

- **Confidence:** Shows how certain it is that this IoC (Indication of Compromise) is real and not a false positive. The higher the number, the more certain it is that the IoC is real.
- **Description:** Describes the individual IoC.
- **Domain:** Domain that represents a threat.
- **IP:** IP Address that represents a threat.
- **Share_level:** TLP handling restrictions which apply for that intel.
- **Sources:** Source-Feed from where the information is provided.
- **URL:** URL that represents a threat.

Question: Our goal is to write a query that monitors the outgoing traffic and checks whether a connection to a suspicious domain has been established.

First find the correct index for the task and then create a list with the columns *count*, *sum_bytes*, *methods by user*, *clientIP* and *uri_domain*. Use the **lookup** command to connect the created list to our threat intelligence feed. Use the appropriate column as a link for this. From the threat intelligence feed we want the columns *confidence*, *description* and *sources* to appear in our logs. At the end, restrict the list so that no entries are displayed that have a confidence lower than 60 or no matching connection to the threat intelligence feed. Find a way to remove them from the result set.

Hints:

- Remember that you can rename columns.
- You need to use regex to get the domain from the events.

Note your final query and save it as an alert as described in chapter 3.3:

```
index=proxy_logs OR index=network_logs
| stats count, sum(bytes) as sum_bytes, values(method) as methods by user, clientIP, uri_domain
| lookup threat_intel domain as uri_domain OUTPUT confidence, description, sources
| where isnotnull(confidence) AND confidence >= 60
| table user, clientIP, uri_domain, count, sum_bytes, methods, confidence, description, sources
```

3.8 A Hard Day's Night

Congratulations, you have made it. All the tasks on your to-do list have been ticked off and you have brought the monitoring system back up to scratch. If something were to happen, you would see it immediately. The bad guys don't stand a chance against your monitoring. But luckily, it's Friday and the weekend is coming up. What could possibly happen at the weekend?

4 Task

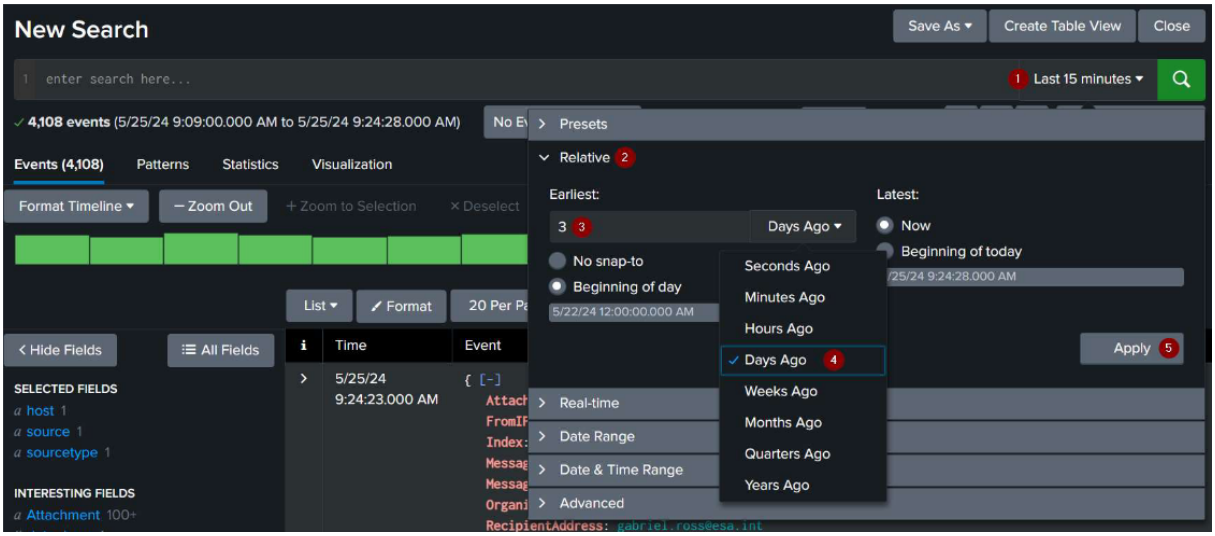
Others would go out at the weekend see friends and family, but you are not one of them. No! You spend the weekend familiarizing yourself with the Splunk indexes and visualization of data. Because you are Splunk and Splunk is you…

OK, let us keep it real. Splunk not only allows us to collect and analyse events, but we can also visualize data with Splunk. With this feature, dashboards can be created that help to see important correlations immediately. In this lab, we will not focus on dashboards, but we want to show a few ways in which Splunk can provide an effective insight into the data.

4.1 Time range picker

In Splunk, you can customize the time span in which you receive events. This is important if you want to search for certain events, e.g., if there was an incident and you need to investigate what exactly happened. You will certainly have to adjust the time span in the following tasks.

The time range can be adjusted as follows:

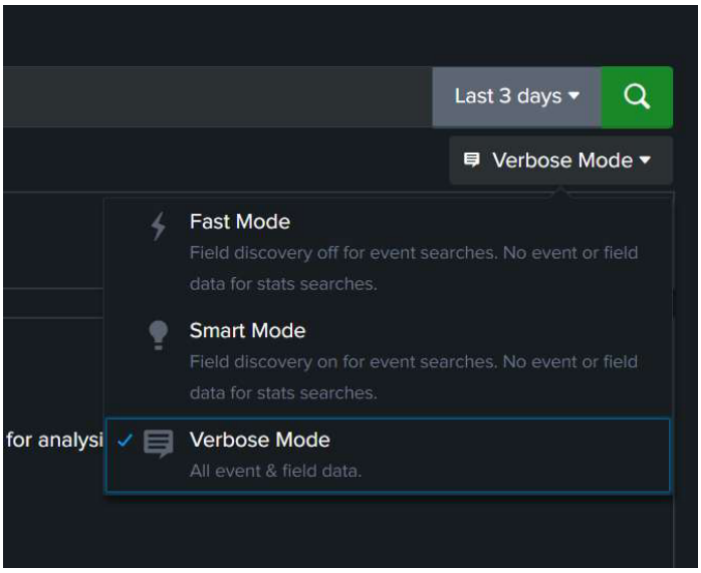


1. Click on the time range picker, right from the search bar (Default Last 15 minutes).
2. Choose the Relative preset option.
3. Type the value.
4. Select Days Ago, from the drop-down list.
5. Click Apply.

The time range picker will be important for the next tasks. To have enough data points, we recommend that the time range is set to 3 days. However, you are free to adjust this value if a search should take too long or if there are too few data points. It is best to play around a little with the time range picker.

4.2 Verbose mode

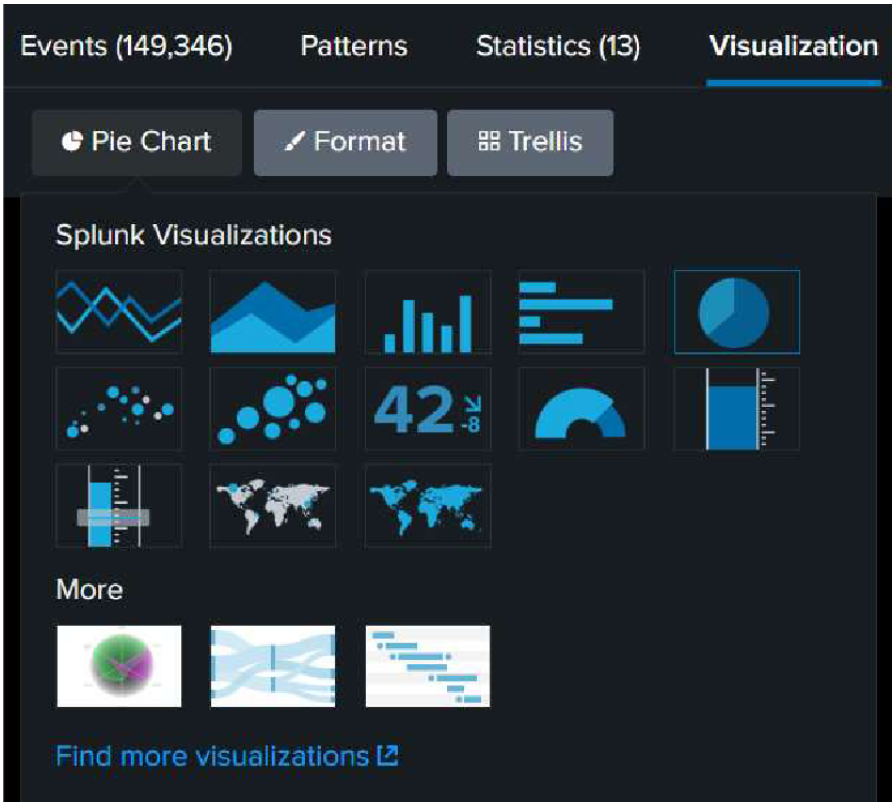
Splunk offers three different search modes. The differences can be read here: `https://docs.splunk.com/Documentation/Splunk/9.2.1/Search/Changethesearchmode`. For the lab we only need the verbose mode.



Change the display from Smart Mode to Verbose Mode so that all events are displayed. This setting can be set below the time range picker.

4.3 Visualisation

Splunk offers a variety of visualizations. We have already gotten to know Events list and Tables. However, Splunk can also display data in charts, gauges, or maps.



The various visualizations can be selected under the **Visualization** tab.

Let us try to create a few helpful visualizations with our data.

4.4 Task – Choropleth Map

So, what is a choropleth map? Simply put, it is a heat map calibrated on a world map. This allows us to show where we get the most internet traffic from, or where our firewall is most accessed or attempted to be accessed from. Such visualizations can be helpful, for example, to support management

in deciding whether a country needs to be geo-blocked if too much unwanted traffic is coming from there.

First, we need to prepare our data. Select the index firewall and write a query to filter out all traffic from the internal network as source IP. As a quick reminder, the internal ranges are 10.0.0.0/8, 172.16.0.0/12 and 192.168.0.0/16.

Question: Write down the query

```
index=firewall sourcetype=eventgen:firewall:kv
| regex src_ip!="^(10\.|192\.168\.|172\.(1[6-9]|2[0-9]|3[0-1]))"
| iplocation src_ip
| stats count by Country
| sort - count
| geom geo_countries allFeatures=True featureIdField=Country
```

The next step is to add geographical information to our source IP addresses. Splunk has the iplocation function that does this for us using data from a third-party database. Add the following line to the query:

```
| iplocation src_ip
```

Through *iplocation*, we have obtained information about the city, country, latitude and longitude for the individual IP addresses, among other things.

We now create a statistic for all countries and count how many IP addresses come from which country. Because we are fancy, we sort the list by number of hits.

Add the following commands to the query:

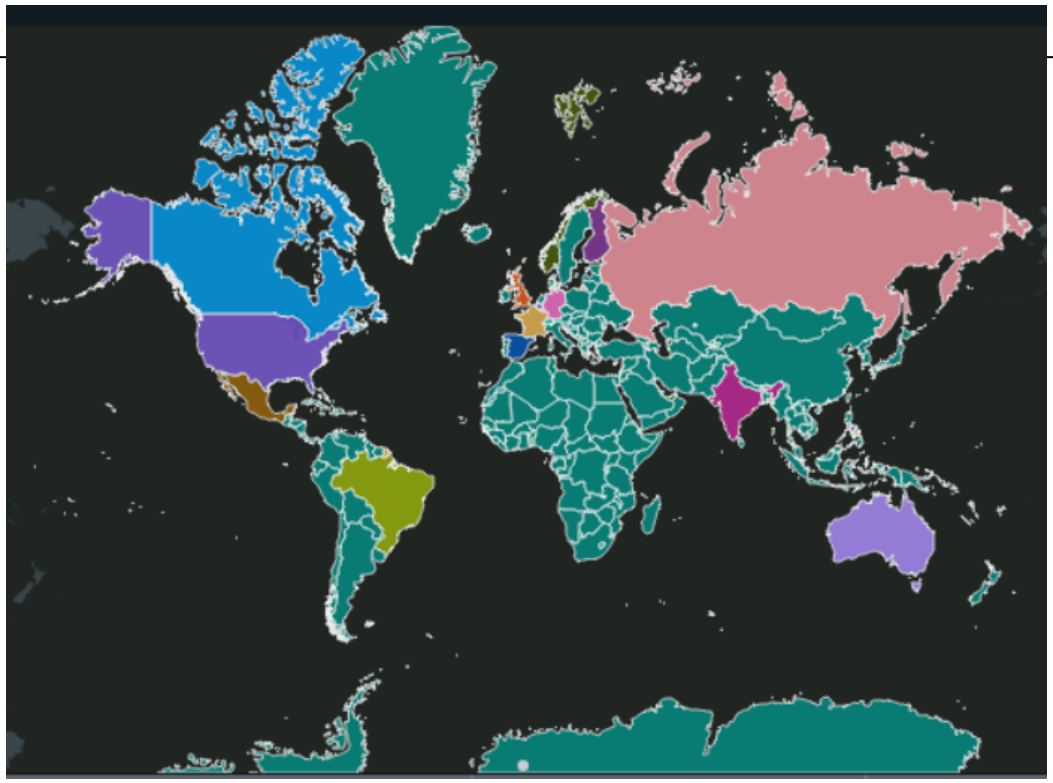
```
| stats count by Country
| sort - count
```

To map this data to our choropleth map, we need to add geographic data structures for polygon geometry. Splunk already has a function with the name *geom* for this. Add the following command to the query:

```
| geom geo_countries allFeatures=True featureIdField=Country
```

When the query is complete, go to Visualization and select the Choropleth Map.

Question: Take a screenshot of your map



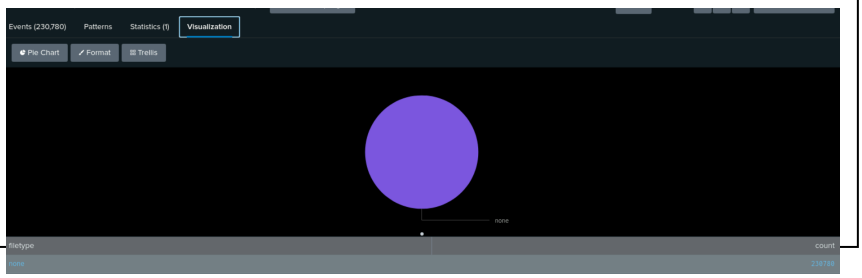
4.5 Task - Pie Chart

In the index Mail we see all incoming and outgoing e-mails from the ZHAW domain. The traffic is always displayed in the same format. We also see that an e-mail either has an attachment, the file-name, or no attachment, which is symbolized with “-”.

Question: Create statistics on the attachments of emails received by employees. Create an overview of how many emails have an attachment and how many do not. Group the attachments by file extension, such as PDF, DOCX, XLSX etc. Create a pie chart and take a screenshot of it.

Hint: You will need to use regex to filter out the file endings. Remember, you can set a variable in the regex right away.

```
index=mail
| eval has_attachment=if(attachment=".", "no attachment", "has attachment")
| eval filetype=if(match(attachment, "(?i)\.(pdf|docx|xlsx|pptx|txt|zip)$"), lower(replace(attachment,
".*\.(?i)([^\.]+)$", "\1")), "none")
| stats count by filetype
```

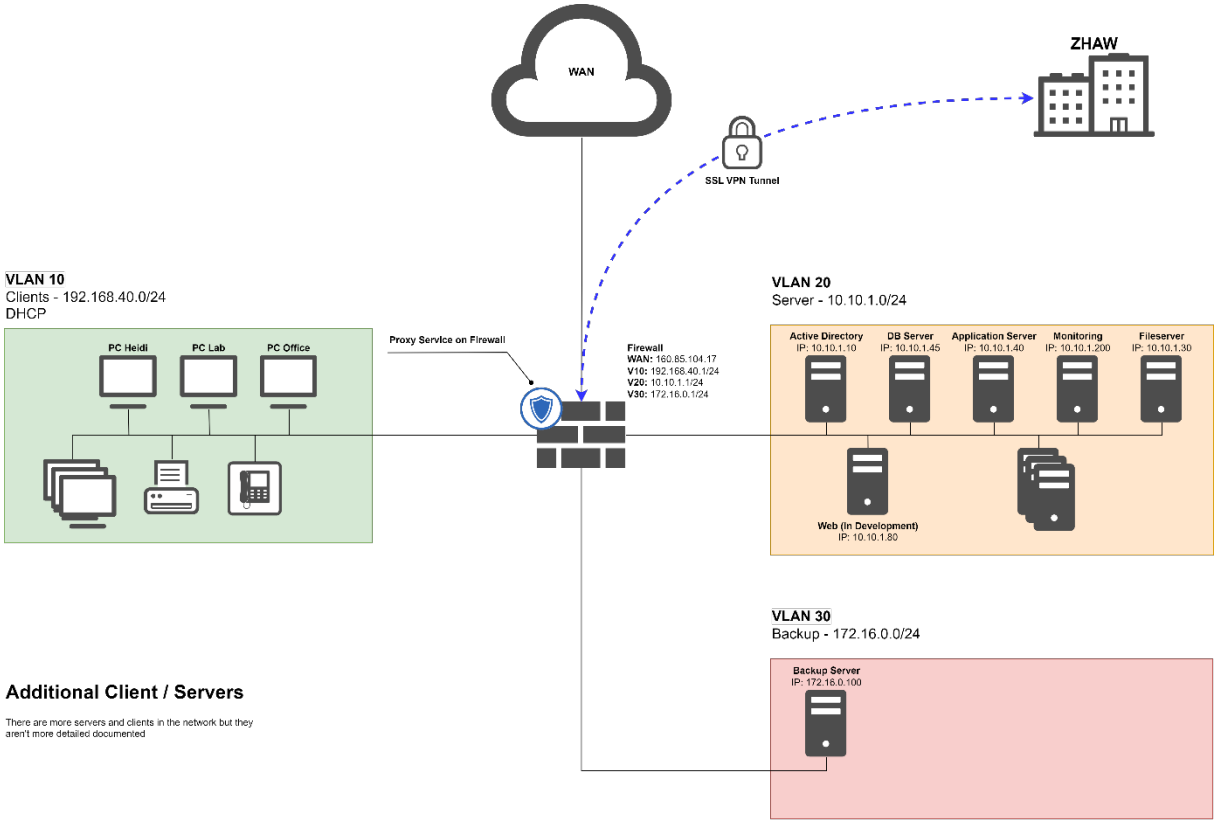


5 Task

After a good and relaxing weekend, you come back to the office. Your team leader comes rushing towards you with a bright red face. It happened over the weekend, shocking: Attackers have gained access to the system. All the data is encrypted and nobody can work any more. He gives you the task of checking the monitoring and finding out exactly what has happened. You check your Splunk installation but fortunately it has not been taken over by the attackers. Thanks to Linux and, of course, one of your strong passwords that you always use everywhere. You get to work to find out exactly what happened.

5.1 Overview

For a better overview, you take out the network plan and a user list. You can't be sure if both are still up to date.



Username	Role
stefan	Team Leader, research team
thomas	Research team
michael	Back office
andreas	Research team
markus	Research team
daniel	Research team
martin	Research team
peter	Research team
alexander	Research team
heidi	Back office
matthias	Research team
dominik	Research team
nicole	Research team
sabine	Research team
andrea	Research team
sarah	IT Administration
julia	Research team
sandra	Research team
melanie	Research team
claudia	Back office
jan	Research team
david	Research team
lukas	Research team
patrick	Research team
manuel	Research team
claude	Research team
marcel	Research team
marco	Research team
oliver	Research team
benjamin	Research team
marius	Research team
daniel	IT Administration
marc	IT Administration
monika	Back office
simon	Research team

5.2 Intermezzo - Prepare Splunk

Please run the following commands on the console:

1. Open a console.
2. Move to the project folder by typing:

   ```
   cd siem
   ```

3. Deploy the attack by executing the following command:

   ```
   sudo ansible-playbook -i inventory.yml deploy_splunk.yml \
       --tags deploy_attack
   ```

Creating the events for the attack takes time, probably about 15 minutes. **Wait about 15 minutes after the restart before continuing with the Lab**. This would be a good time to take a break. The reason for this is that all events must first be created correctly. If no alerts are visible after 15 minutes, the VM should be restarted. Also check if the crontab settings and time ranges were set correctly in the previous tasks.

All created alerts should have gone off. **If no or only a few alerts have gone off, you may not have waited long enough.** Go get a coffee. If an alert has not gone off, you can simply customize the query in the search bar and set the time span to 3 days ago (adjustable on the right side of the search bar), then you should see if events are displayed now. It is no longer necessary to create new alerts, as these would not go off anyway, as the events are already in Splunk.

5.3 Task - What happened?

You can find your triggered alerts at the top right under *Activity*.

So, what has happened now? With your alerts, you should be able to answer the following questions without any problems. If you have problems coming up with the right idea, ask yourself what a large data exfiltration event would look like. You could also ask yourself if we would show you all these Splunk features in Section 4 just for the fun of it.

Question: Has data been stolen? If so, how can we recognize this and where was it sent?

Ich denke nicht, weil
Actions = DENY

Question: Which users, clients or servers have been compromised? Make sure you know the host-name and IP address of the systems that have been compromised.

client = 192.168.40.96

hostname = firewall-01

6 Task

We now know what happened, but how could it happen? How did the hackers get into the system?

Follow the tracks in the events and find the point of entry. What *processes* had to take place for this attack to be successful? One of the triggered alerts should give you a good starting point. Consider whether the attack exploited a vulnerability in one of the systems or whether social engineering was used to gain access to the network.

Question: Note the path from the entry point until the connection to the Command & Control Server has been established:

Hints:

- Consider all indexes.
- Could a process create a scheduled task?

Port Scan von 192.168.40.96 an 192.168.40.28, um zu schauen welche Ports akitv sind! Grosse Bytes anzahl weist auf Datenaustausch hin!

Question: Where have weaknesses been identified in the network and what needs to be improved in the future to prevent such an attack from happening again? List as many vulnerabilities as possible:

Question: Try to connect the individual attacks to the phases of the Cyber Kill Chain. We should now have enough information to fill out all the steps:

7 Conclusion

What a day. You hand in your report to your boss in which you can show exactly where the weak points were. The only positive thing is that the hackers didn't get to the backup and couldn't delete it. It should therefore be possible to continue working soon. However, you don't know what the hackers will do with the stolen data. The future will show whether a ransom demand will arrive after all. Fortunately, this is not your problem to decide.

8 Clean-Up

To stop Splunk after completing the lab, execute the following command in the console:

1. Open a console.
2. Move to project folder by typing:
   ```
   cd siem
   ```
3. Execute the following command:
   ```
   sudo ansible-playbook -i inventory.yml deploy_splunk.yml --tags dis-
   able_autostart
   sudo systemctl disable disable-thp
   ```

9 Lab Points

For **3 Lab Points** you must create all queries in task 3.

You get **1 Lab Point** if you finished task 4.

For another **2 Lab Points**, you must answer all questions in task 5 and 6.

Show your filled-in sheet to your instructor. They may ask you questions when inspecting your sheet.