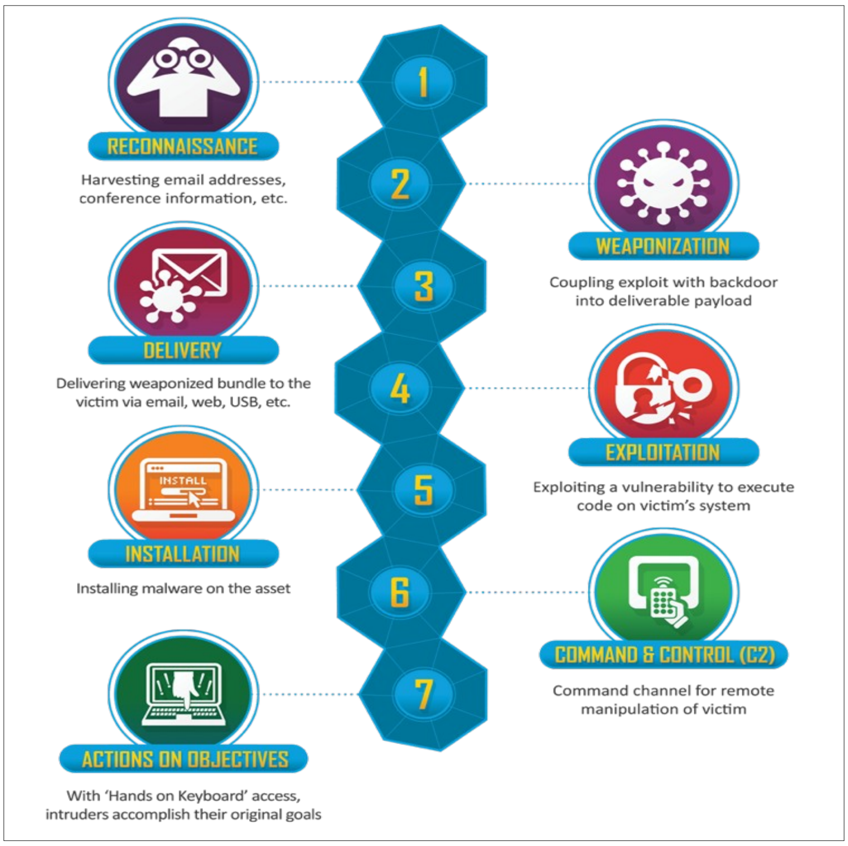
SIEM
SECURITY INFORMATION & EVENT MANAGMENT

Splunk Workshop

I've used Splunk to investigate a scenario using the **"2016 Boss of the SOC" dataset**, to identify an ***A**dvanced **P**ersistent **T**hreat* within the network. They have defaced an external website, and the goal is to find out who is responsible for this.
Splunk and SIEM were some of the topic of the *Cyber Security Bootcamp* run by the University Study of Bristol in 2021-2022. Upon complition of the program I had a good understanding of the SIEM field and a fair knowledge of Splunk as investigative tool. I was capable of distinguish and use the different software's components: indexer, fowarder, search head, apps. Ive applied this knowldge on further workshops and exercises.

The hands-on exercise for this workshop is based on the BOTS data set that was developed in 2016 and used for the first iteration of Boss of the SOC. This workshop is designed to provide a hands-on walk through using Splunk as an investigative tool.
We will be using **Lockheed Martin's Kill Chain** to break down each attack and report it accordingly.

Lockheed Martin's Kill Chain
Source: Tryhackme.com/room/bpsplunk


1
RECONNAISSANCE
Harvesting email addresses, conference information, etc.
2
WEAPONIZATION
Coupling exploit with backdoor into deliverable payload
3
DELIVERY
Delivering weaponized bundle to the victim via email, web, USB, etc.
4
EXPLOITATION
Exploiting a vulnerability to execute code on victim's system
5
INSTALL
INSTALLATION
Installing malware on the asset
6
COMMAND & CONTROL (C2)
Command channel for remote manipulation of victim
7
ACTIONS ON OBJECTIVES
With 'Hands on Keyboard' access, intruders accomplish their original goals

1. Introduction

Siem

Security information and event management (SIEM) is a field within the field of computer security, where software products and services combine security information management (SIM) and security event management (SEM). They provide real-time analysis of security alerts generated by applications and network hardware.

Splunk

Splunk is a software platform to search, analyze and visualize the machine-generated data gathered from the websites, applications, sensors, devices etc. which make up your IT infrastructure and business

Source: Google.com

BOTS! 'Boss of the SOC' Security Operations Center competition

Boss of the SOC (BOTS) is a Capture the Flag-esque (CTF) competition that is played in teams of up to four players and hosted by Splunk and mnemonic. The game features contestants playing the role of a SOC Security Analyst. Participants are challenged to answer questions about security-related scenarios

2. Execution

2.1 Reconnaissance

As we are working with preloaded data the first thing to do is to set the data frame to *All time* instead of *Last 24hrs.* Our server is **imreallynotbatman.com**

2.1.1 What Ip address is scanning our web server?

Our first step is to start examinating all the event which occured on our webserver.

The data set's index we'll be working on is *botsv1,* and we must include it in our search

```
index="botsv1"  imreallynotbatman.com
| stats  count  by  src_ip
```

With these simple commands we are asking the database to show us a statistic view of sources Ip counts. With the '|' we can add commands to the query. As result we can see 3 Ip addresses with 90% of activity coming from the Ip.

- ***40.80.148.42***

We now need to validate our scan and to do so we analyse the ***Suricata*** *sourcetype* (Add **sourcetype = suricata** to the search query). Because Suricata has IDS Signatures in it, we can scroll the fields and discover additional informations.
From the Suricata's Signatures we can confirm that the **Ip 40.80.148.42** is scanning our webserver.

2.1.2 What Web scanner scanned the server?
Now we know on which Ip address we can focus on and we redirect our attention on the the web traffic redifining our search with ***sourcetype = "stream_http"*** instead of suricata. In addition we look at the *headers* to try discover other information.

```
index="botsv1"  imreallynotbatman.com  src_ip=40.80.148.42
sourceype="stream:http"
| stats  count  by  src_headers
```

By simply looking at the headers we see the presence of ***Acunetix Web Vulnerabilities Scanner.***

2.1.3 What Content Managment System is our server using?
We first find out what's our server Ip address. We can do this by selecting the *dest* field on the sidebar during our search. Now we need to analyse the web traffic on our server and narrow the results focusing on the URL. This would allow us to see which files and directories structure are requested.

```
index="botsv1"  dest=192.168.250.70  sourcetype="stream:http"
| stats  count  by  url
```

We can now see the directories structure and we can confirm that ***Joomla*** is our server CMS. Knowing the system, and by so the vulnerabilities, we are trying to protect is priority.

2.2 Exploitation
At this stage we must find out which vulnerabilities have exploited and what sort of attack have been executed.

2.2.1 What Ip address is performing a bruteforce attack against the server?
We now want to focus on the *http* data traffic, and more in detail, on a specific ***http method: POST***. Why? We know that, most likely, a bruteforce attack would be executed with a http post method in various attempts to "guess" the password.

```
index="botsv1"  dest=192.168.250.70  sourcetype="stream:http"
http_method= Post
| stats  count  by  url
```

As result of this search query two Ip addresses call our attention: the Ip address who performed the scanning 40.80.148.42, and counts for the most of the activity, and another IP address: 23.22.163.114.
Despite the majority of activity coming from the first IP address we must confirm the result with further queries. To do so we have to look at where input of *username* and *passwords* were attempted.

```
index="botsv1"  dest=192.168.250.70  sourcetype="stream:http" http_method=
Post  form_data=*username*password*
| stats  count  by  url
```

The **form_data** field contains information being passed from the client browser to the web server. If we search the form_data, we can use wildcards to look for values that contain the strings *username* and *password* within the field. Again we only need to create a statistic table of our results to see that more than %99 of activity in this case come from the **IP 23.22.163.114**.

2.2.2 What was the first password attempted?
To answer this question we need another variable which is the **_time** variable. The data set will be shown from the youngest to the oldest and so we need to **reverse** our result. Moreover, to have a tabular view of our data will use the command **table**.

```
index="botsv1"  sourcetype="stream:http"  form_data=*username*password*
| table  time  form_data
| reverse
```

As result of our search we can now see that the first password attempted was **12345678**.

2.2.3 What was the correct password for admin access to the content management system?
We must extract a specific password from the form data field. We want also count and sort the data by number of attempts. It's resonable to believe that while in a bruteforce attack each word we'll be used once, the correct password will be used twice to gain access.
It's time to introduce the command **rex**. Rex stands for regular expression and it'll extract values and create new fields we can work with.

```
index="botsv1"  sourcetype="stream:http"  form_data=*username*password*
dest_ip=192.168.250.70
| rex  field = form_data  "passwd=(?<userpassword>\w+)?
```

The *username* field will show the password **batman** with a count of 2.
We can also further uo our investigation trying

By adding the **values** function to our stats command, we can see that the brute force attack came from 23.22.63.114 but it appears that the actual penetration with the correct password came from 40.80.148.42.

2.2.4 How many seconds elapsed between the time the brute force password scan identified the correct password and the compromised login rounded to 2 decimal places?
The **transaction** command gives us an easy way to group events together based on one or more fields and returns a field called duration that calculates the difference between the first and last event in a transaction.

```
index=botsv1  sourcetype=stream:http
| rex field=form_data "passwd=(?<userpassword>\w+)"
|search userpassword=batman
| transaction userpassword
| table duration
```

We can use the **eval** command to round it to two decimal places: the result will be **92.17 seconds**.

2.2.5 How many unique passwords were attempted in the brute force attempt?
Calculating the number of unique passwords helps understand the spread of the password attempts. Perhaps understanding the passwords attempted might give us intelligence to look for future attacks of a similar nature. With the use of the **distinct count** (**dc**) function within the stats command we can count each word once so we can exclude the repetition of the word batman.

```
index=botsv1   sourcetype=stream:http   form_data=*username*passwd*
| rex field=form_data   "passwd=(?<userpassword>\w+)"
| stats dc(userpassword)
```

2.3 Installation

We now need to investigate if and what has been installed into the asset.

2.3.1 What is the name of the executable uploaded by P01s0n1vy?
A clue in the question indicates that the name of the file was an executable. We could take that literally and search for .exe. We also know that because the adversary was uploading to our web server, stream:http may be useful as well. If we compare the *sourcetypes stream:http* and *suricata* we can see files with the destination Ip of our server. Finally, the file has been probably uploaded via *Http method Post:*

```
index=botsv1   sourcetype="suricata"   dest_ip="192.168.250.70"
http_method=POST   *.exe
  | stats count by fileinfo.filename
```

One of the two results appears to be associated with Microsoft Front Page extensions and by so the executable file we are looking for is **3791.exe**

2.3.2 What is the MD5 hash of the executable uploaded?

We now move from what's happening on our network and start interrogating what that partticular file is doing on the endpoint. To gather the right informations we now look at another sourcetype: **sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational**. The **Sysmon** captures information on how that file is being used on our OS. We can then create a statistic view of the **value(MD5)**.

```
index=botsv1  3791.exe  sourcetype=
sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational
CommandLine=3791.exe
 | stats values(MD5)
```

Now we can clearly see the value of MD5 Hash:
AAE3F5A29935E6ABCC2C2754D12A9AF0

2.4 Action on objectives

On this stage we need discover what actions the intruder was able to perform on the objectives.

2.4.1 What is the name of the file that defaced the imreallynotbatman.com website?

As we previously saw, we can see the http and the suricata logs with traffic destination set on our web server. But what happen if we set our web server as source? We see that it's communicating with some external Ip addresses. One of these is **23.22.163.114**. Normally, a web server wouldn't start the communication but in this case it does as it has been compromise.

```
index=botsv1  src=192.168.250.70  sourcetype=suricata  dest_ip=23.22.63.114
| stats count by url
```

Nearly all the values have an http.urI and we can see by looking at the results that the file **/poisonivy-is-coming-for-you-batman.jpeg** has been uploaded to the server.

2.5 Command and Control

We now need to identify command channels used for remote manipulation.

2.5.1 What fully qualified domain name (FQDN) is associated with this attack?

As we know the name of the file, it'll be very easy to answer this question, the answer is contained in the same log. We only need to include the file name in our query. Sourcetype set on http.

```
index=botsv1 dest=23.22.63.114 "poisonivy-is-coming-for-you-batman.jpeg" src=192.168.250.70 sourcetype=stream:http
```

If we scroll down the events we can see the domain name associated with the attcker’s Ip address.

Resources

1.Institute of Coding Skills Workshop
http://www.cems.uwe.ac.uk/~pa-legg/resources/ioc/week3.html
2. Cyber security learning platform
https://tryhackme.com/room/bpsplunk
3. Splunk Documentation
https://docs.splunk.com/Documentation
4. W3 School
https://www.w3schools.com/html/