Task 1: Getting familiar with Wazuh

Questions:

● Describe the steps involved to install Wazuh and generate alerts.

Initially my experience with virtual machines was outdated, so I asked for the software you used to be sure the one chosen is correct and works properly.

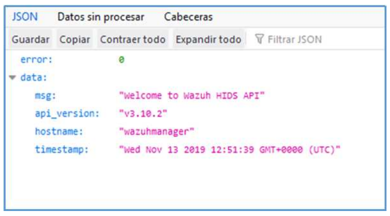
Creating a virtual machine with CentOS using vagrant was surprisingly easy.

Once the virtual machine is ready, Installing following instructions in Wazuh webpage is not a complex task, the steps are clearly explained.

I found a problem importing the template for Elasticsearch, the json file in `https://raw.githubusercontent.com/wazuh/wazuh/v3.10.2/extensions/elasticsearch/7.x/wazuh-template.json` is unreachable, cause an incorrect certificate, I can't obtain it for other ways, I tried using web browser, and no exception can be added.

In this dead-end, I make the decision to obtain the developing box you provided in your webpage, this way I have all software installed.

With everything installed, I started all services without problems and can obtain messages using a host web browser.



```
  error:            0
▼ data:
    msg:            "Welcome to Wazuh HIDS API"
    api_version:    "v3.10.2"
    hostname:       "wazuhmanager"
    timestamp:      "Wed Nov 13 2019 12:51:39 GMT+0000 (UTC)"
```

In other hand I can´t go furthermore creating alerts and events to be catched by manager, I have a basic knowledge of sysadmin and can't perform security issues.

With more time, and better equipment (I'm running a virtual machine with a very poor performance due to my host machine), I will learn enough to test deeply the software .

Task 2: Extracting alert information

Without exploring deep enough and creating alerts in manager I can´t obtain a proper alert file to parse it and create the script wanted. In this situation I obtained the default alert file and create one script that will be an approximation to the one required.

Is the file Myscript.js located in Task2 folder. I used the alerts.json file to elaborate it.

Task3: Angular

This is the larger one.

It has a component to make the menu of 4 tabs, each tab covers one of the requirements of this task.

- The first one is the one using the directive, has a *ngIf directive who erase a button from the DOM when another button its pressed more than 10 times

- The second tab has various buttons, all of them send a message, all types of messages are different (alert, snakbar, console) all are controlled by the same service called message-service, the type of message is selected by an argument.

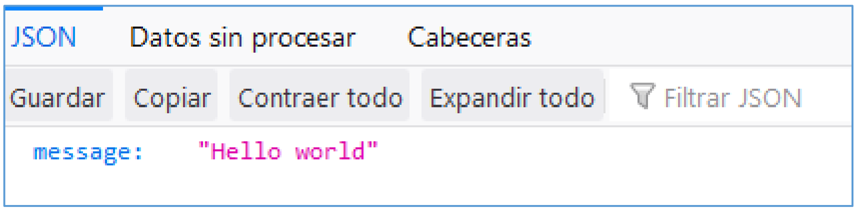
- The third one is the one uses an External API. I choose use the API from famous web comic xkcd, I start using JSON api but switched to JSONP to avoid errors caused by CORS security policy. Every comic has sequential ID, input the ID (number) click the button and the date of published, the number and the image of the comic (with alt text) appears.

- The four one is a dynamic table, inserting any string will build a table instantly, | will change the column and || will change the row. You can use this as example Hoy|Mañana|Viernes||Soleado|Mayormente soleado|Parcialmente nublado||19°C|17°C|12°C||E 13 km/h|E 11 km/h|S 16 km/h. For security all the characterse inside the marking tags < and > will be erased, to avoid code injection.

  *The dependence and modules are missing to make the file lighter, npm install will install them automatically.

Task4 Node.js

The small server is formed by two files, the app.js file, the main of the server, and routes.js, which defines the GET endpoint with the message to sent. To execute Node app.js



```
message:    "Hello world"
```

Task5 and Task 6 ares skipped

Task 7: Linux

- List all system users.
  - cut -d: -f1 /etc/passwd
- Create an empty file.
  - Touch MyNewFile
- Modify the owner for that file.
  - chown newowner MyNewFile
- Print last 5 entries from a log file.
  - tail –n 5 logfile.log
- Print the very first line from a log file.
  - Head –n 5 logfile.log
- Enable and disable a service.
  - Systemctl enable/disable myService
- Restart a service.
  - Systemctl restart myService
- Find all occurrences of a word recursively in all files from a directory.
  - grep -r myWord folder
- Extract all lines from a log file that include a certain word.
  - Grep myWord logfile.log > fileWithWord.txt