Attack Tree DREAD Calculator

Attack Tree DREAD Calculator, short ATDC, is a Python application for generating, i.e. visualising attack trees and calculating the DREAD score for each node (leaf) in the attack tree.

It takes a JSON file as input, which needs a specific structure for the app to work successfully. The structure is as follows:

```
{
  "nodes": [
    {
      "id": "id",
      "name": "name",
    }
  ],
  "edges": [
    {
      "from": "from",
      "to": "to",
    }
  ]
}
```

The input file, i.e. the JSON file, is the basis for the visualisation, i.e., the attack tree specification.

This application relies heavily on utilising the pyvis library, which aims to build a python-based approach to constructing and visualising network graphs in the same space (pyvis.readthedocs.io, n.d.).

ATDC was inspired by Suraksha, which is a workbench for security designers intended to be embedded into the system from the very early stages of Software Development Life Cycle (SDLC)(Maurya et al., 2009).

This application applies the same methodology (DREAD) for calculating the risk associated with a node as Suraksha, meaning that a user can select the appropriate value (between 1 and 10) for each DREAD category of a single node: Damage Potential (D), Reproducibility (R), Exploitability (E), Affected Users (A) and Discoverability (D).

Once all values have been entered, a user receives a risk score for each node based on the following formula:

```
Risk = (D + R + E + A + D) / 5
```

The calculation always produces a number between 0 and 10; the higher the number, the more serious the risk (Maurya et al., 2009).

Installation

1. Clone repository

Clone the repository from GitHub to access the project.

2. Set up a virtual environment

If you use a modern IDE, such as PyCharm, the programme will automatically ask you to set up a virtual environment and do the rest. You only have to select the Python version. Once you do, it will set up the venv, active it and afterwards install the dependencies from the reqirements.txt file.

Once you have downloaded it, please navigate to the folder in which you have saved it and set up a virtual Python environment. **!! Make sure to use Python 2.7 !!**

Suppose the system does not automatically assign the venv to your project. In that case, you will need to use different syntax for activating the virtual environment depending on which operating system and command shell you're using (Yegulalp, 2022).

On Unix or macOS, using the bash shell:

```
source /path/to/venv/bin/activate
```

On Windows, using the command prompt:

```
path\to\venv\Scripts\activate.bat
```

3. Install dependencies

Now that you have set up a virtual Python environment for this project, you can start to install the dependencies. Make sure that you are located in the root folder of the project. To install the dependencies, execute the following command in the root folder.

```
pip install -r requirements.txt
```

Great! You are now ready to start using the Attack Tree DREAD Calculator App.

Usage

Python

The ATDC App takes two inputs from a user and produces one output file. The first input is the file path to the JSON file containing the attack tree specification, and the second is the name for the output file produced by the ATDC app.

To start the app, navigate to the project's root folder (in your terminal/console) and execute the following command:

```
python main.py
```

This command will start the application, and you will be presented with the following screen:

```
================================
Attack Tree DREAD Calculator App
================================
You will have to input a JSON file and name the output file, which will automa

1. Enter the file's location: [file location, e.g. attack_tree.json]
2. Enter the output file's name: [output file's name, e.g. attack_tree_result]
```

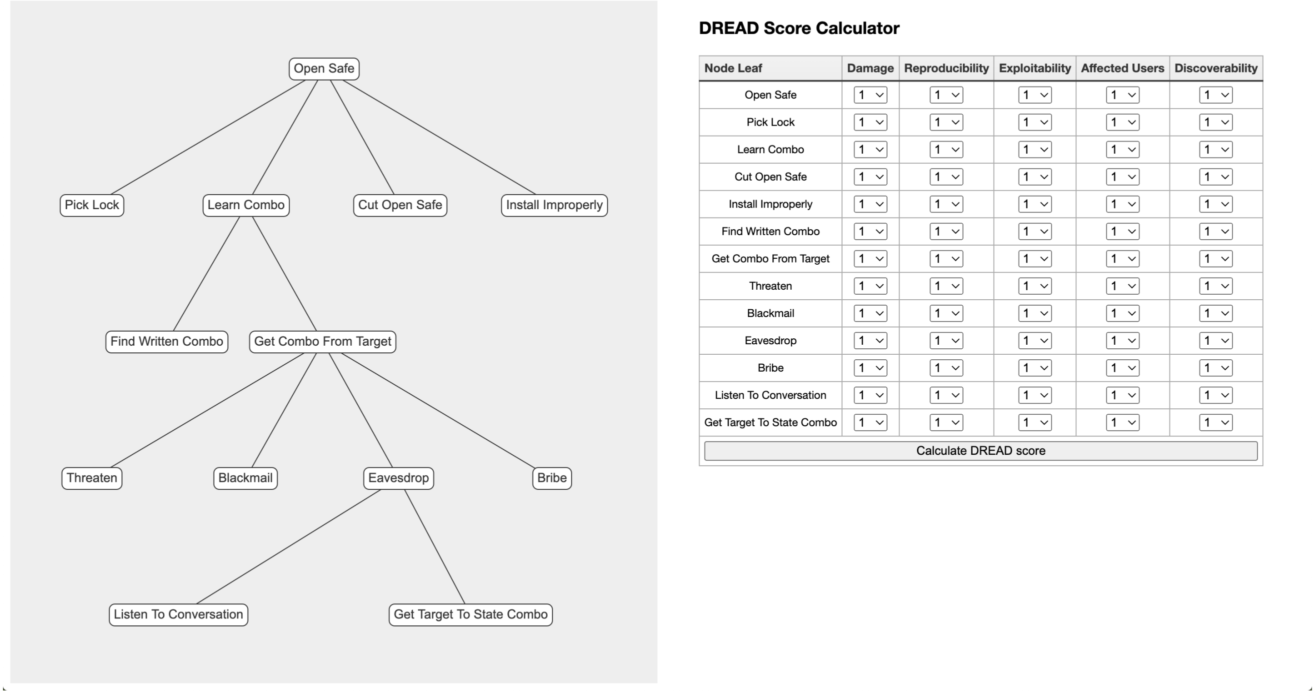
Once you have entered the required inputs, the app will automatically start a browser and open the output file. From this moment on, the Python part of the app has finished executing.

JavaScript / HTML / Browser

Now that you find yourself in the internet browser, you will see the generated attack tree on the left side of the screen and the DREAD score calculator on the right side. This table will be populated by the attack tree's node (leaf) values. For each node, there will be the option to enter the DREAD values (1-10).

The following action you can undertake is changing the default values of 1 for each node (DREAD categories) and adapting them to your liking.

Once you have done that, click on the "Calculate DREAD score" button, and a new table will be generated with DREAD scores for each node, which can be seen in the example below:



DREAD Score Calculator

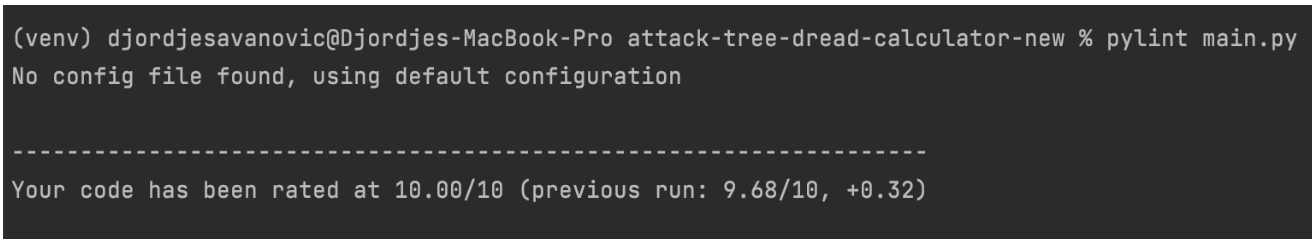
Node Leaf	Damage	Reproducibility	Exploitability	Affected Users	Discoverability
Open Safe	1	1	1	1	1
Pick Lock	1	1	1	1	1
Learn Combo	1	1	1	1	1
Cut Open Safe	1	1	1	1	1
Install Improperly	1	1	1	1	1
Find Written Combo	1	1	1	1	1
Get Combo From Target	1	1	1	1	1
Threaten	1	1	1	1	1
Blackmail	1	1	1	1	1
Eavesdrop	1	1	1	1	1
Bribe	1	1	1	1	1
Listen To Conversation	1	1	1	1	1
Get Target To State Combo	1	1	1	1	1

Figure 1.

Linter output

```
(venv) djordjesavanovic@Djordjes-MacBook-Pro attack-tree-dread-calculator-new
No config file found, using default configuration

------------------------------------------------------------------
Your code has been rated at 10.00/10 (previous run: 9.68/10, +0.32)
```



```
(venv) djordjesavanovic@Djordjes-MacBook-Pro attack-tree-dread-calculator-new % pylint main.py
No config file found, using default configuration

------------------------------------------------------------------
Your code has been rated at 10.00/10 (previous run: 9.68/10, +0.32)
```

Figure 2.

Demonstration of Application

For the demonstration of the ATDC application, I will use a demo attack tree from the Henriksen(2018) article on utilising draw.io for generating attack trees.

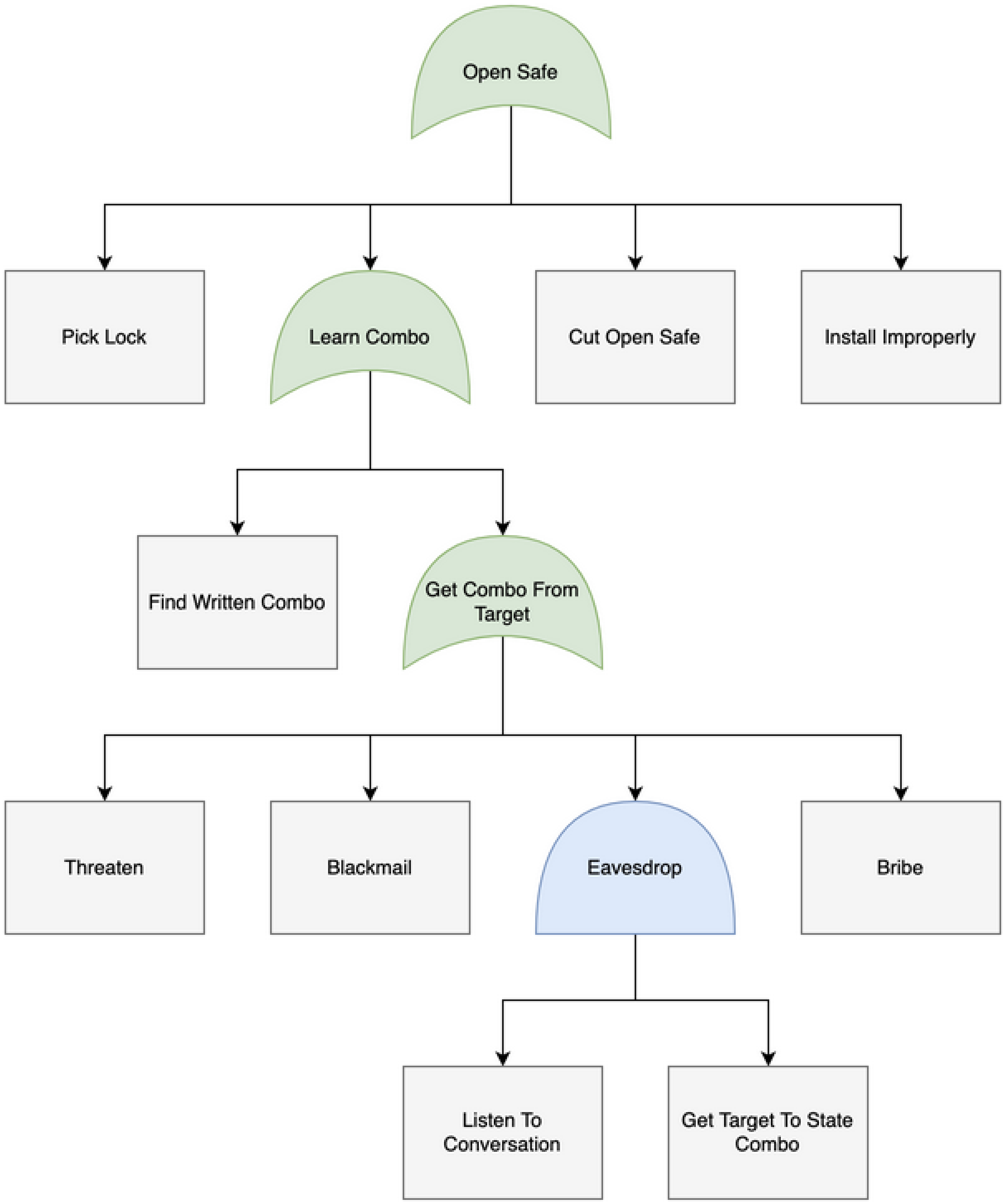
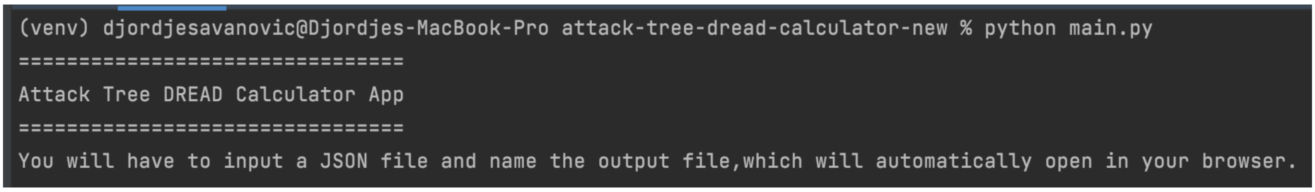


Figure 3: An attack tree exploring how to open a safe. (Henriksen, 2018)

First, a user needs to create the attack tree specification. In this case, that file can be found in the root folder of this project - it is called `attack_tree.json`.

Secondly, a user needs to start the app by typing in the following command:

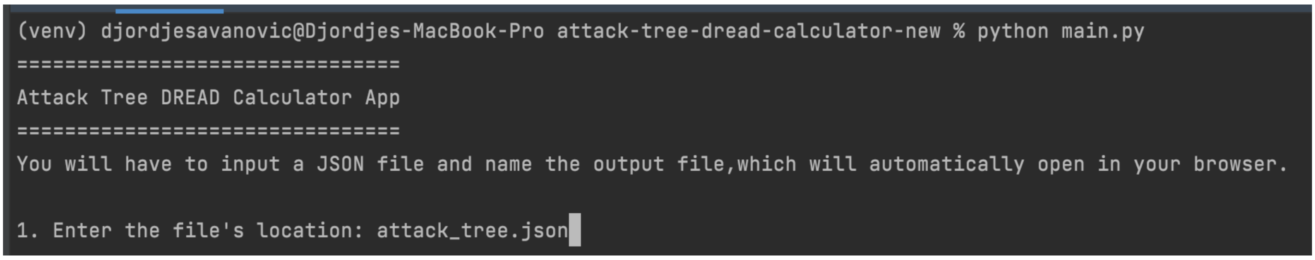
```
python main.py
```



```
(venv) djordjesavanovic@Djordjes-MacBook-Pro attack-tree-dread-calculator-new % python main.py
================================
Attack Tree DREAD Calculator App
================================
You will have to input a JSON file and name the output file,which will automatically open in your browser.
```

Figure 4.

Once the app runs, users must type the file path to the attack tree specification.

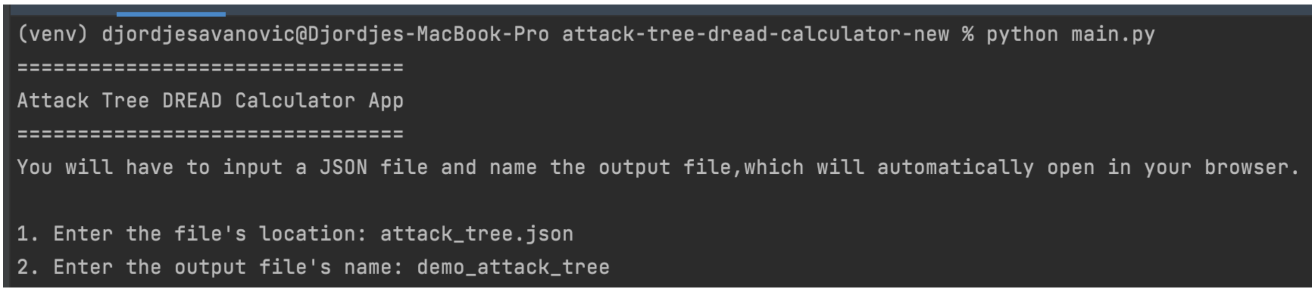


```
(venv) djordjesavanovic@Djordjes-MacBook-Pro attack-tree-dread-calculator-new % python main.py
================================
Attack Tree DREAD Calculator App
================================
You will have to input a JSON file and name the output file,which will automatically open in your browser.

1. Enter the file's location: attack_tree.json
```

Figure 5.

Secondly, they need to enter the name of the output file:



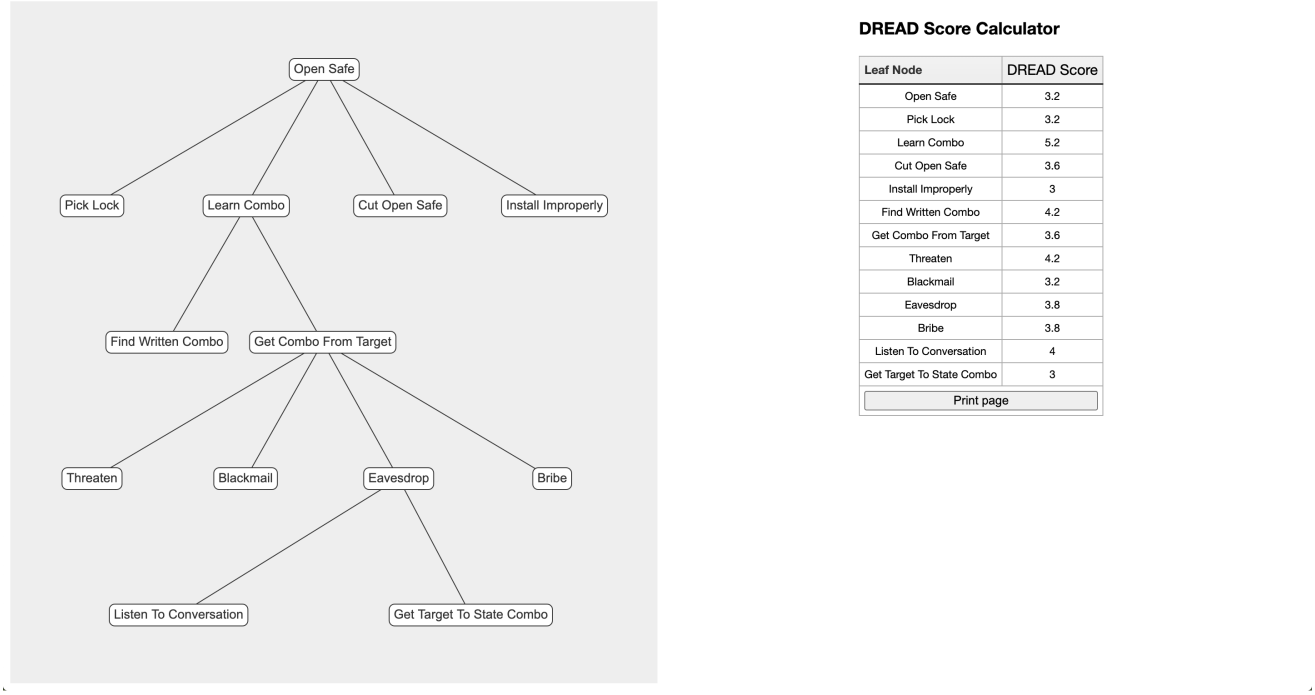
```
(venv) djordjesavanovic@Djordjes-MacBook-Pro attack-tree-dread-calculator-new % python main.py
================================
Attack Tree DREAD Calculator App
================================
You will have to input a JSON file and name the output file,which will automatically open in your browser.

1. Enter the file's location: attack_tree.json
2. Enter the output file's name: demo_attack_tree
```

Figure 6.

Thirdly, they need to press enter and wait for the browser window to open, where they will be presented with the screen, as seen in Figure 1.

Once they are in the browser window they can enter values for each DREAD category of each node and press the "Calculate DREAD risk" button. They should end up with something like this:



DREAD Score Calculator

Leaf Node	DREAD Score
Open Safe	3.2
Pick Lock	3.2
Learn Combo	5.2
Cut Open Safe	3.6
Install Improperly	3
Find Written Combo	4.2
Get Combo From Target	3.6
Threaten	4.2
Blackmail	3.2
Eavesdrop	3.8
Bribe	3.8
Listen To Conversation	4
Get Target To State Combo	3

Figure 7.

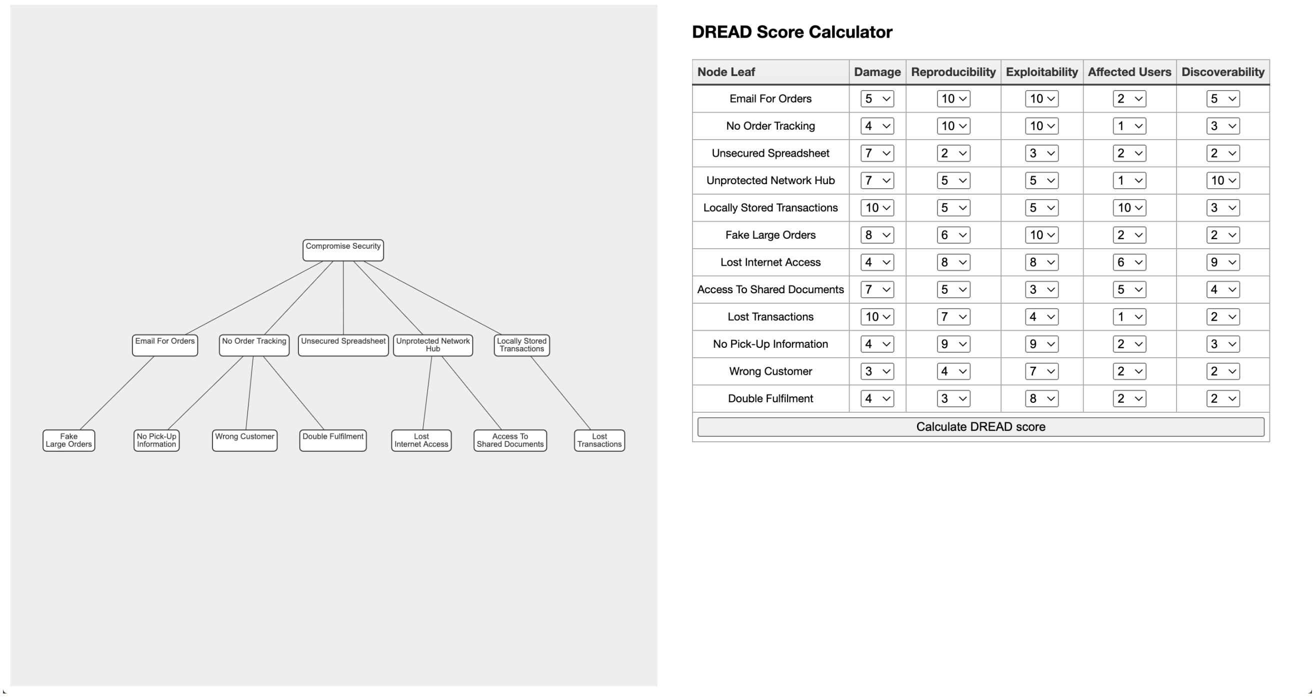
Summary

The aim of this exercise was to compare the risk states of the company Pampered Pets before and after the process of digitalisation. This was achieved by visualising attack trees and then calculating risks scores based on the DREAD methodology for each node on the attack tree.

In the images below, we can see that based on the values inserted in the DREAD categories of nodes of both attack trees, one can conclude that the overall risk score is a bit higher in the "after digitalisation" scenario - to be precise, it is 0.15 higher.

Before the process of digitalisation, the overall risk score came to **5.19** and after the aforementioned process, it amounted to **5.34**. Based on the score, I can recommend the company maintain their *status quo*.

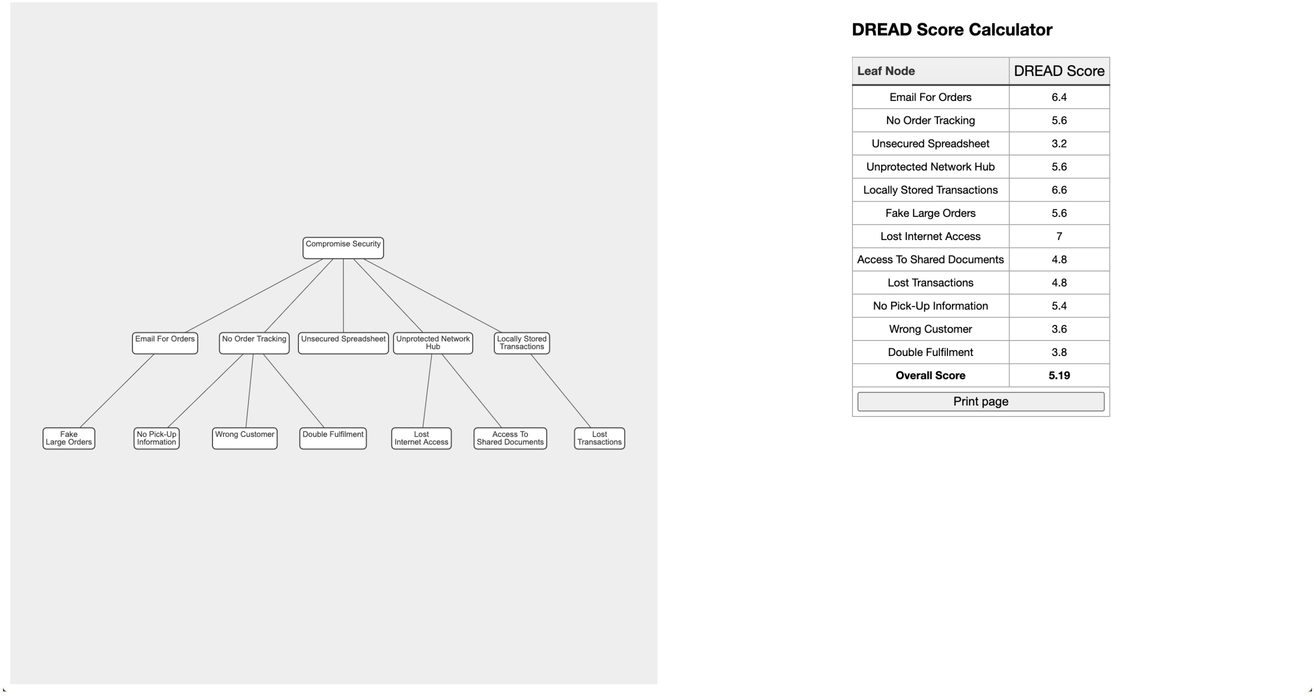
In order to better understand the values inserted, I suggest visiting the following web page, which explains all categories and the magnitude of their values: EC-Council.



DREAD Score Calculator

Node Leaf	Damage	Reproducibility	Exploitability	Affected Users	Discoverability
Email For Orders	5	10	10	2	5
No Order Tracking	4	10	10	1	3
Unsecured Spreadsheet	7	2	3	2	2
Unprotected Network Hub	7	5	5	1	10
Locally Stored Transactions	10	5	5	10	3
Fake Large Orders	8	6	10	2	2
Lost Internet Access	4	8	8	6	9
Access To Shared Documents	7	5	3	5	4
Lost Transactions	10	7	4	1	2
No Pick-Up Information	4	9	9	2	3
Wrong Customer	3	4	7	2	2
Double Fulfilment	4	3	8	2	2

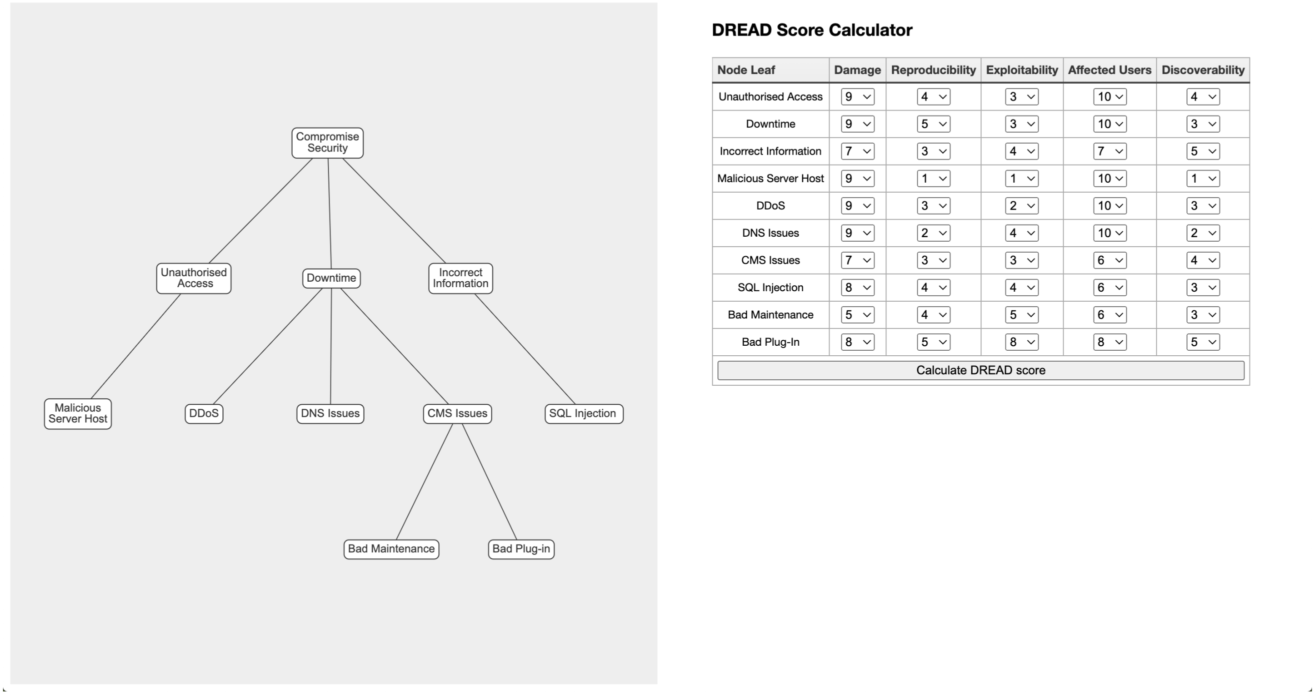
Figure 8: Values inserted in DREAD categories - Scenario: before digitalisation



DREAD Score Calculator

Leaf Node	DREAD Score
Email For Orders	6.4
No Order Tracking	5.6
Unsecured Spreadsheet	3.2
Unprotected Network Hub	5.6
Locally Stored Transactions	6.6
Fake Large Orders	5.6
Lost Internet Access	7
Access To Shared Documents	4.8
Lost Transactions	4.8
No Pick-Up Information	5.4
Wrong Customer	3.6
Double Fulfilment	3.8
Overall Score	**5.19**

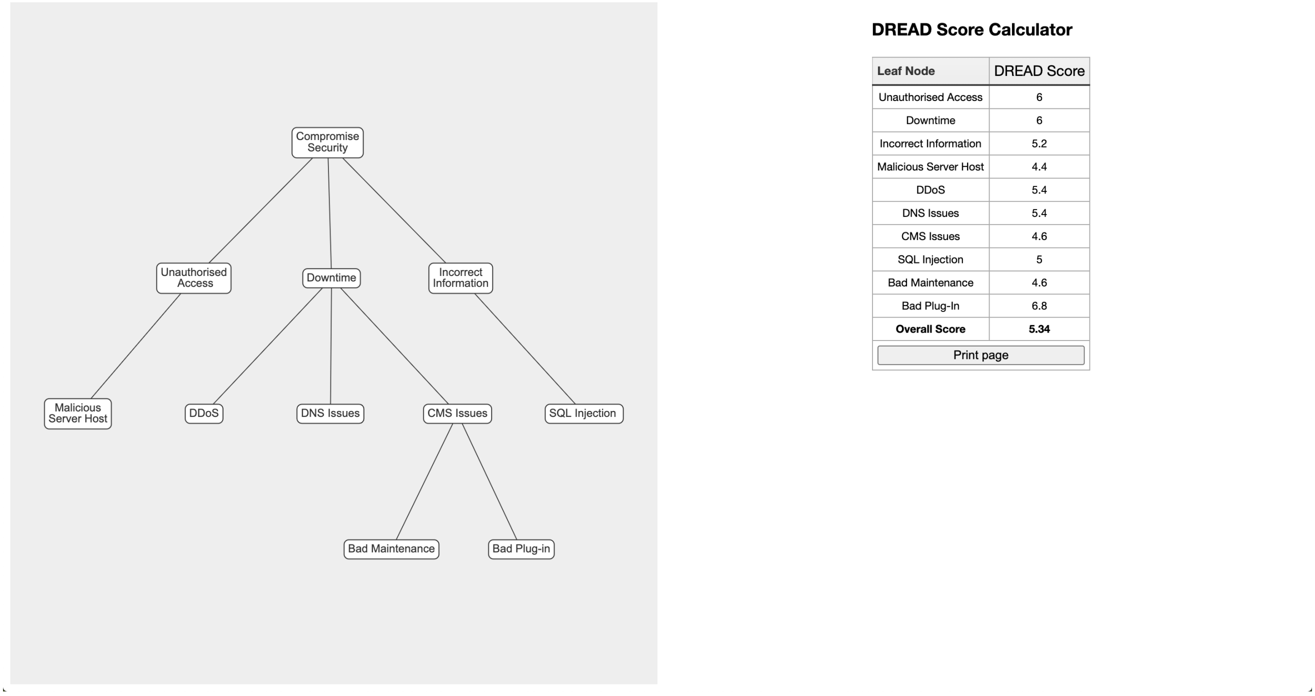
Figure 9: DREAD Risk Score - Scenario: before digitalisation



DREAD Score Calculator

Node Leaf	Damage	Reproducibility	Exploitability	Affected Users	Discoverability
Unauthorised Access	9	4	3	10	4
Downtime	9	5	3	10	3
Incorrect Information	7	3	4	7	5
Malicious Server Host	9	1	1	10	1
DDoS	9	3	2	10	3
DNS Issues	9	2	4	10	2
CMS Issues	7	3	3	6	4
SQL Injection	8	4	4	6	3
Bad Maintenance	5	4	5	6	3
Bad Plug-In	8	5	8	8	5

Figure 10: Values inserted in DREAD categories - Scenario: after digitalisation



DREAD Score Calculator

Leaf Node	DREAD Score
Unauthorised Access	6
Downtime	6
Incorrect Information	5.2
Malicious Server Host	4.4
DDoS	5.4
DNS Issues	5.4
CMS Issues	4.6
SQL Injection	5
Bad Maintenance	4.6
Bad Plug-In	6.8
Overall Score	**5.34**

Figure 11: DREAD Risk Score - Scenario: after digitalisation

References

Henriksen, M. (2018) Draw.io for threat modelling. Michael Henriksen - Freelance Security Engineer & Consultant. Available from: https://michenriksen.com/blog/drawio-for-threat-modeling/ [Accessed 13 October 2022].

Maurya, S., Jangam, E., Talukder, M. & Pais, A.R. (2009) 'Suraksha: A security designers' workbench', Hack.in. National Institute of Technology Karnataka, Kanpur, 17-19 March 2009. 59–66.

pyvis.readthedocs.io. (n.d.) Introduction — pyvis 0.1.3.1 documentation. Available from: https://pyvis.readthedocs.io/en/latest/introduction.html [Accessed 10 October 2022].

Yegulalp, S. (2022) Virtualenv and venv: Python virtual environments explained. InfoWorld. Available from: https://www.infoworld.com/article/3239675/virtualenv-and-venv-python-virtual-environments-explained.html [Accessed 26 October 2022].