Malicious Behavior Detection Using Audit Logs

Beta Release

Team ID: 19

Project ID: CS_Egcert3_2023

Presented by :

Safaa Sarhan	300389876
Esraa Mahmoud	300389401
Salma Hasanin	300389877

Safaa Sarhan - 300389876

Project Mentor
Dr. Ahmed Hamdy

Project Sponsor:
EGCERT company

Ottawa support:
Dr. Miguel Garzon

Problem definition

Increasing Cybersecurity Threats

Insufficient Detection Mechanisms

Challenges in Manual Analysis of logs

Safaa Sarhan - 300389876

Users and Benefits

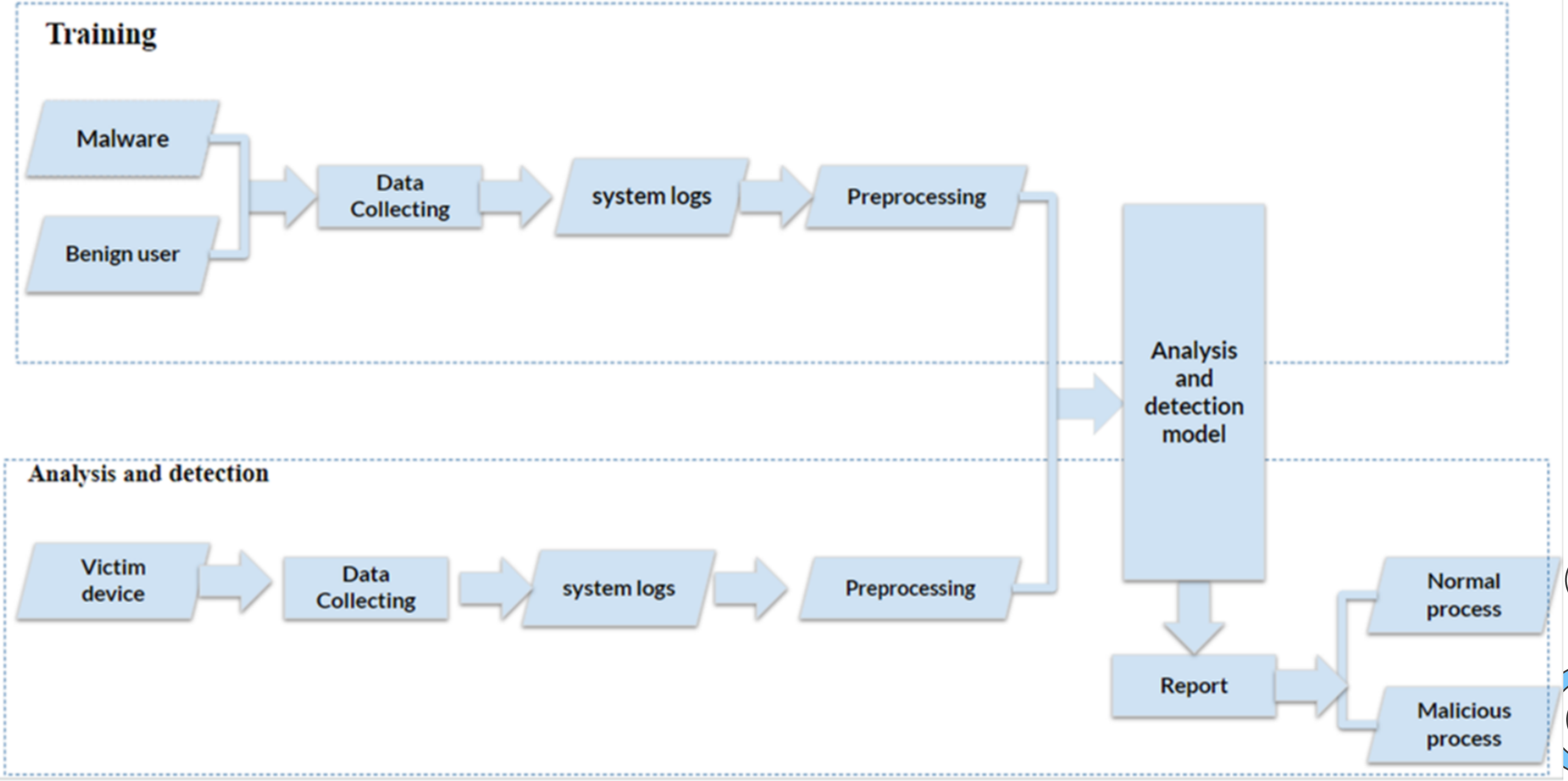
01 detect anomalies in user activity.

02 Improved threat detection and incident response

03 Reduced risk of successful attacks

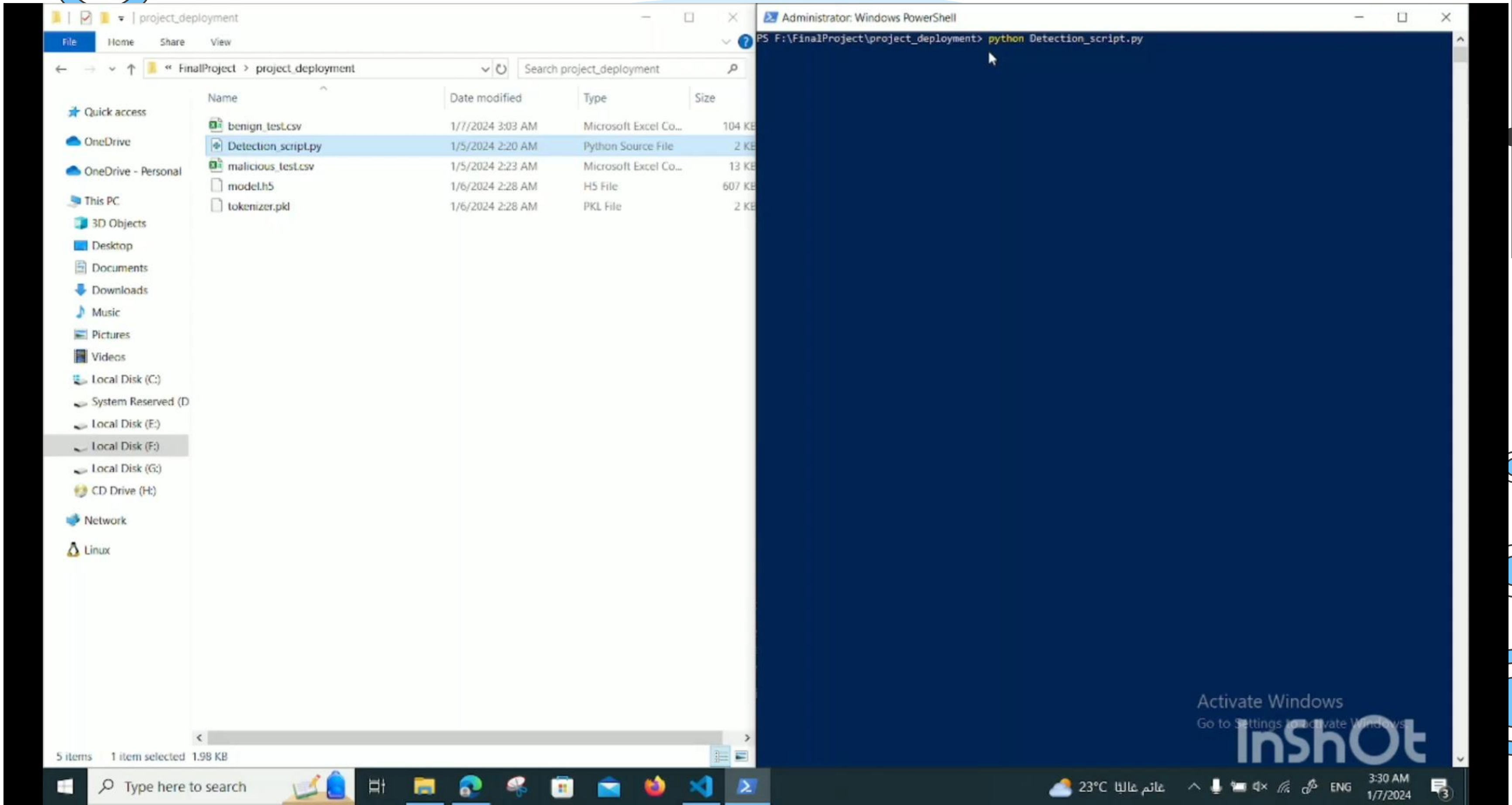
Safaa Sarhan - 300389876

The proposed log analysis and detection method



Our Demo

Safaa Sarhan - 300389876



Name	Date modified	Type	Size
benign_test.csv	1/7/2024 3:03 AM	Microsoft Excel Co...	104 KB
Detection_script.py	1/5/2024 2:20 AM	Python Source File	2 KB
malicious_test.csv	1/5/2024 2:23 AM	Microsoft Excel Co...	13 KB
model.h5	1/6/2024 2:28 AM	H5 File	607 KB
tokenizer.pkl	1/6/2024 2:28 AM	PKL File	2 KB

5 items 1 item selected 1.98 KB

```
PS F:\FinalProject\project_deployment> python Detection_script.py
```

Safaa Sarhan - 300389876

01

Data collection

Why sysmon?

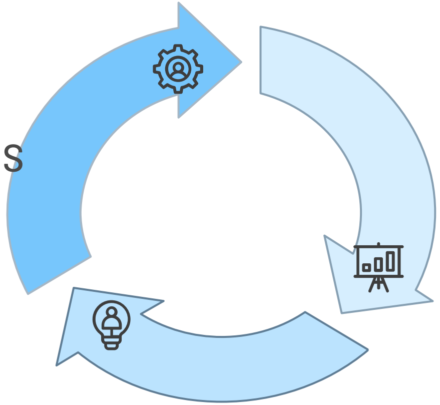
enable collection of detailed information : various system events, processes, network connections..etc.

benign data : collected from our daily used devices

malicious data *: Simulated almost 300 attacks from MITER ATT&CK matrix on a VM*

Safaa Sarhan - 300389876

Malicious logs



Simulate attack
Using atomic red teams

Collect logs
as XML files using sysmon tool

Return to the VM fresh snapshot
To prevent log overlapping from previous attack

Safaa Sarhan - 300389876

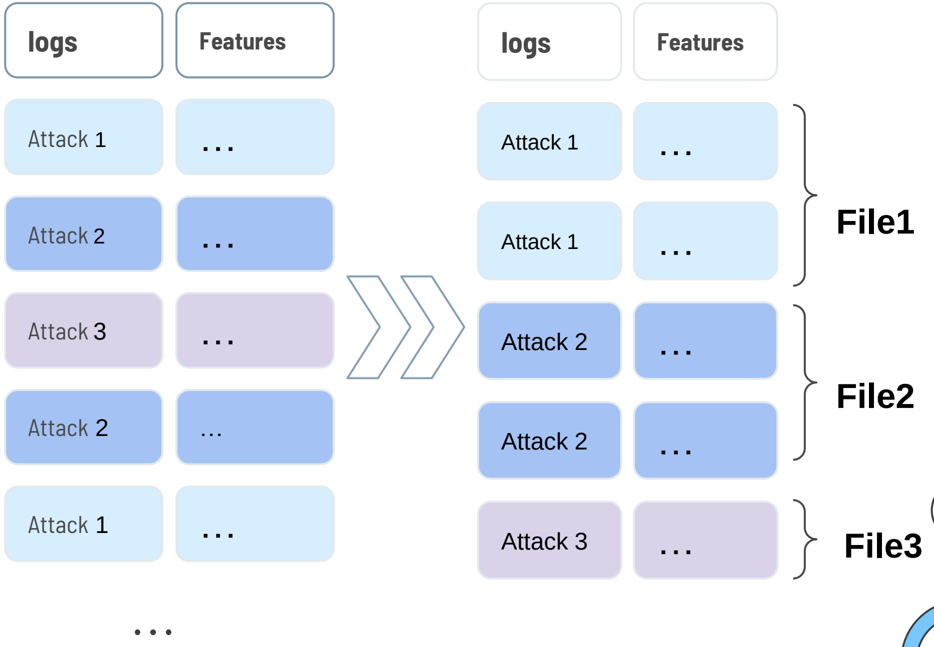
Why this cycle
and not simulate all attacks in one VM?

Why..?

analyse pattern behaviour

Some attacks generate logs after a while

it can interface with the logs of the next attack



logs	Features
Attack 1	...
Attack 2	...
Attack 3	...
Attack 2	...
Attack 1	...

...

logs	Features	
Attack 1	...	File1
Attack 1	...	
Attack 2	...	File2
Attack 2	...	
Attack 3	...	File3

Esraa Mahmoud - 300389401

02

Data preprocessing

Esraa Mahmoud - 300389401

How to process the data for anomaly detection?

specific sequence of benign logs

≠

1 normal log

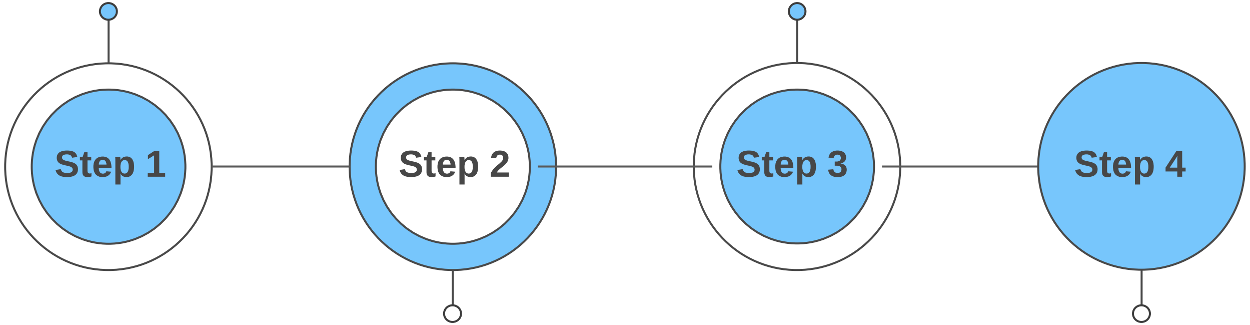
the analysis will be done by collection not one log “Pattern analysis”

logs	Features
Benign 1	…
Benign 2	…
Benign 3	…
Benign 4	…

Malicious

Esraa Mahmoud - 300389401

Our Process



Step 1

data cleaning

Step 2

Select most important features to use

Step 3

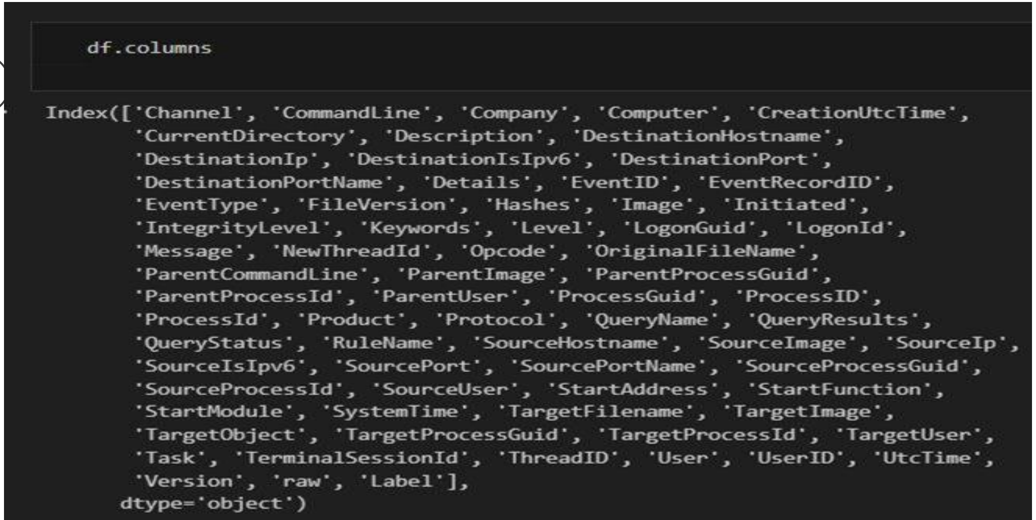
Combine each log features in one column

Step 4

Combine the logs of each file in one row

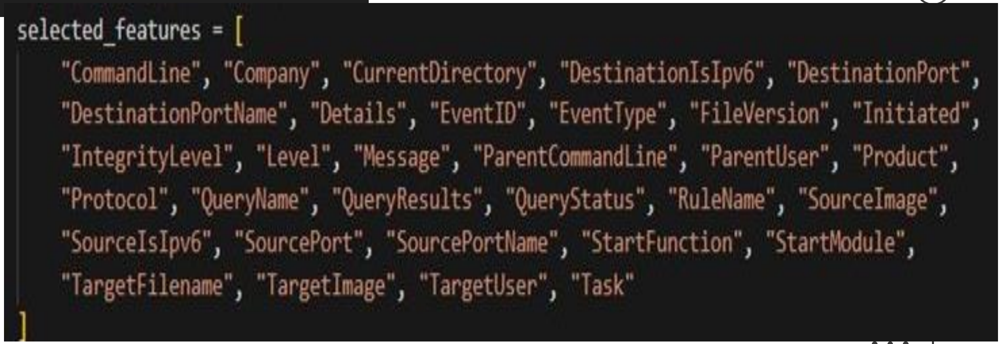
Esraa Mahmoud - 300389401

Selected features



```
df.columns
```

```
Index(['Channel', 'CommandLine', 'Company', 'Computer', 'CreationUtcTime',
       'CurrentDirectory', 'Description', 'DestinationHostname',
       'DestinationIp', 'DestinationIsIpv6', 'DestinationPort',
       'DestinationPortName', 'Details', 'EventID', 'EventRecordID',
       'EventType', 'FileVersion', 'Hashes', 'Image', 'Initiated',
       'IntegrityLevel', 'Keywords', 'Level', 'LogonGuid', 'LogonId',
       'Message', 'NewThreadId', 'Opcode', 'OriginalFileName',
       'ParentCommandLine', 'ParentImage', 'ParentProcessGuid',
       'ParentProcessId', 'ParentUser', 'ProcessGuid', 'ProcessID',
       'ProcessId', 'Product', 'Protocol', 'QueryName', 'QueryResults',
       'QueryStatus', 'RuleName', 'SourceHostname', 'SourceImage', 'SourceIp',
       'SourceIsIpv6', 'SourcePort', 'SourcePortName', 'SourceProcessGuid',
       'SourceProcessId', 'SourceUser', 'StartAddress', 'StartFunction',
       'StartModule', 'SystemTime', 'TargetFilename', 'TargetImage',
       'TargetObject', 'TargetProcessGuid', 'TargetProcessId', 'TargetUser',
       'Task', 'TerminalSessionId', 'ThreadID', 'User', 'UserID', 'UtcTime',
       'Version', 'raw', 'Label'],
      dtype='object')
```



```
selected_features = [
    "CommandLine", "Company", "CurrentDirectory", "DestinationIsIpv6", "DestinationPort",
    "DestinationPortName", "Details", "EventID", "EventType", "FileVersion", "Initiated",
    "IntegrityLevel", "Level", "Message", "ParentCommandLine", "ParentUser", "Product",
    "Protocol", "QueryName", "QueryResults", "QueryStatus", "RuleName", "SourceImage",
    "SourceIsIpv6", "SourcePort", "SourcePortName", "StartFunction", "StartModule",
    "TargetFilename", "TargetImage", "TargetUser", "Task"
]
```

Esraa Mahmoud - 300389401

03

Analysis and detection model

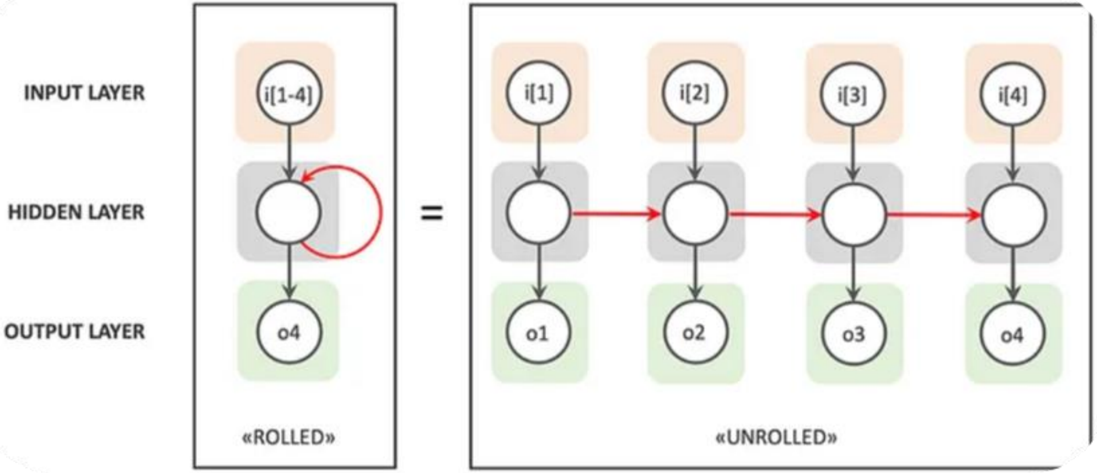
Esraa Mahmoud - 300389401

RNN Model

Recurrent Neural Networks model (RNN)

designed to process sequential data

takes some specific input and returns output (Many to one)

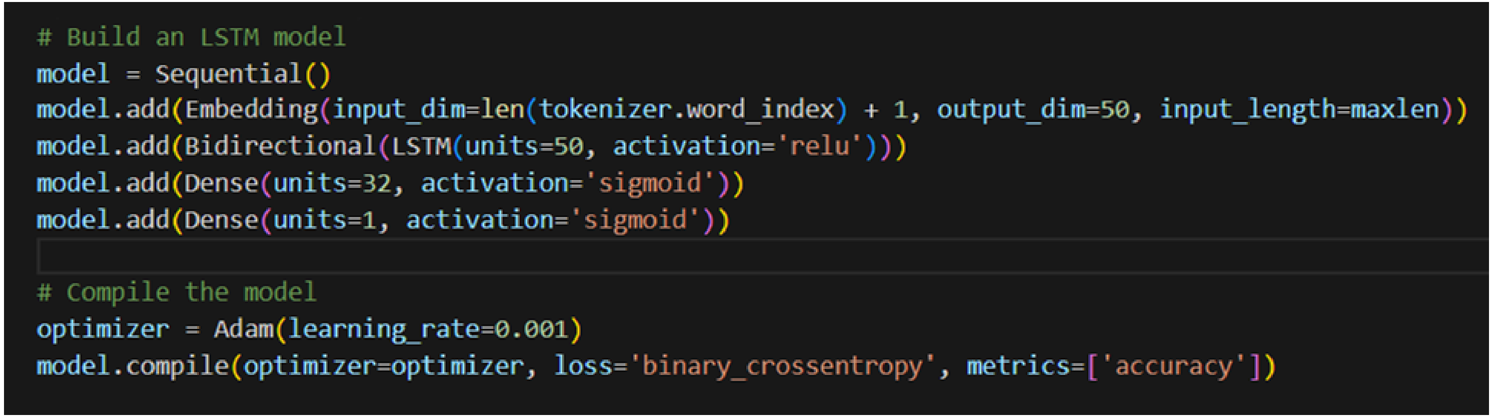


libraries: Pandas, NumPy and TensorFlow.

Building the RNN Model:

Contains of :

1. embedding layer
2. bidirectional LSTM layer
3. 2 dense layers with a 'sigmoid' activation function



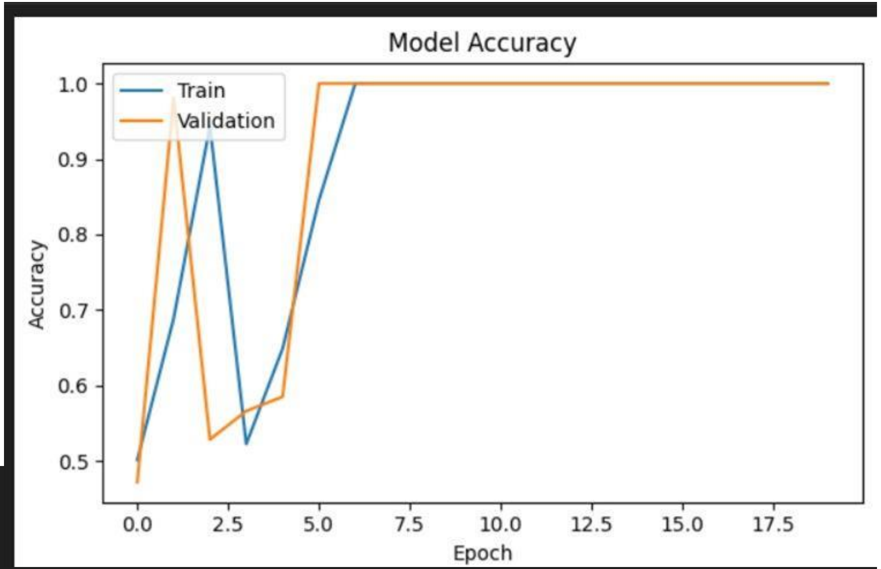
```
# Build an LSTM model
model = Sequential()
model.add(Embedding(input_dim=len(tokenizer.word_index) + 1, output_dim=50, input_length=maxlen))
model.add(Bidirectional(LSTM(units=50, activation='relu')))
model.add(Dense(units=32, activation='sigmoid'))
model.add(Dense(units=1, activation='sigmoid'))


# Compile the model
optimizer = Adam(learning_rate=0.001)
model.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy'])
```

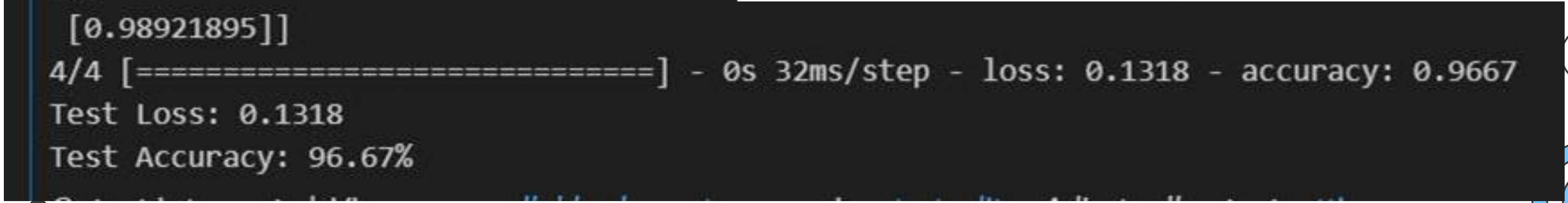
Esraa Mahmoud - 300389401

Model accuracy

In each iteration:

validation and accuracy are monitored to check for overfitting and model's performance.





```
 [0.98921895]
...
 [0.98921895]]
4/4 [==============================] - 0s 32ms/step - loss: 0.1318 - accuracy: 0.9667
Test Loss: 0.1318
Test Accuracy: 96.67%
```

Salma Hasanin - 300389877

04

Deployment

Deployment Plan

01

Extract the XML log

The end-user able to use windows sysmon logs , or a one query log at a time.

02

Anomaly Detection

The end-user can query for anomaly in any of those two files.

Salma Hasanin - 300389877

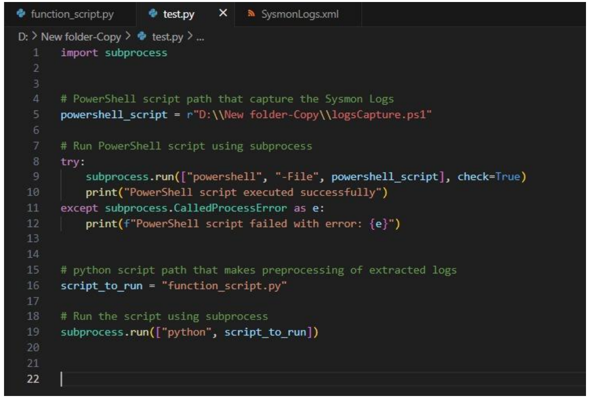
Deployment Plan

1

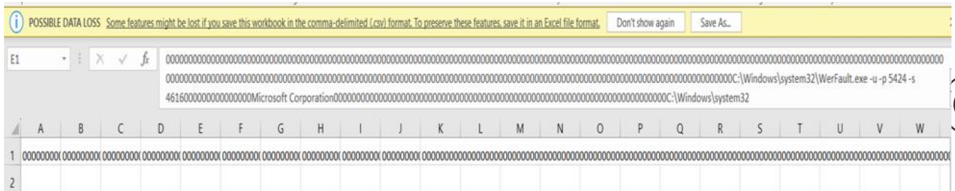
Since **XML** logs requires a lot of effort to be processed, in our deployment we configure a one united script to deals with more than XML format .

The united script

Final shape of data



```python
import subprocess


# PowerShell script path that capture the Sysmon Logs
powershell_script = r"D:\\New folder-Copy\\logsCapture.ps1"

# Run PowerShell script using subprocess
try:
    subprocess.run(["powershell", "-File", powershell_script], check=True)
    print("PowerShell script executed successfully")
except subprocess.CalledProcessError as e:
    print(f"PowerShell script failed with error: {e}")


# python script path that makes preprocessing of extracted logs
script_to_run = "function_script.py"

# Run the script using subprocess
subprocess.run(["python", script_to_run])


```



Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format.

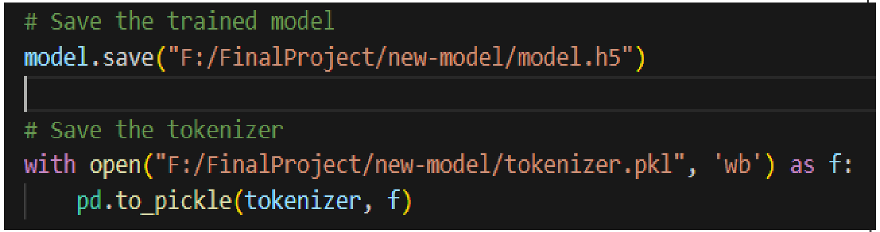
```
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000C:\Windows\system32\WerFault.exe -u -p 5424 -s
4616000000000000000Microsoft Corporation0000000000000000000000000000000000000000000000000000000000000C:\Windows\system32
```

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	00000000(	00000000(	00000000(	00000000(	00000000(	00000000(	00000000(	00000000(	00000000(	00000000(	0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000												
2																							

Salma Hasanin - 300389877

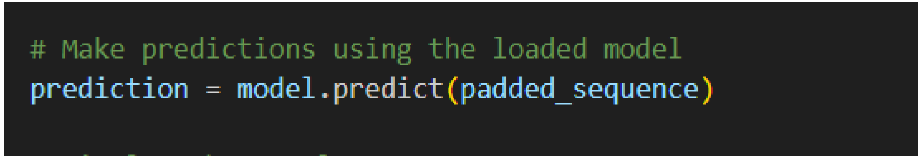
Deployment Plan

1 We saved the model in a pickle file



```
# Save the trained model
model.save("F:/FinalProject/new-model/model.h5")

# Save the tokenizer
with open("F:/FinalProject/new-model/tokenizer.pkl", 'wb') as f:
    pd.to_pickle(tokenizer, f)
```

2 Feeding the model with the final shape of data to be predicted .



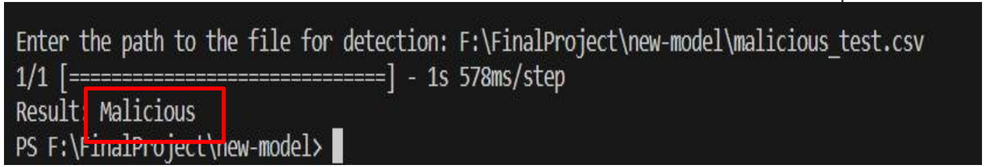
```
# Make predictions using the loaded model
prediction = model.predict(padded_sequence)
```

Salma Hasanin - 300389877

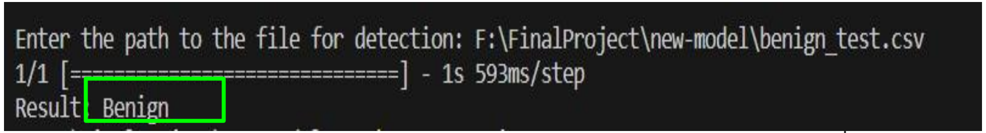
Deployment Plan

3

test a new log file -> “malicious” or “benign”



```
Enter the path to the file for detection: F:\FinalProject\new-model\malicious_test.csv
1/1 [==============================] - 1s 578ms/step
Result: Malicious
PS F:\FinalProject\new-model> 
```



```
Enter the path to the file for detection: F:\FinalProject\new-model\benign_test.csv
1/1 [==============================] - 1s 593ms/step
Result: Benign
```

4

CMD interface enable end-users / security Analysts to test logs from their environments with minimal installation effort , and from only one script.

Salma Hasanin - 300389877

Future Work

1 **Generate reports and dashboards to provide insights into the overall security posture.**

2 **Improve the model with more threats types**

Thanks!

Do you have any questions?