Reputation Enforced Bayesian Network Analysis Tool User Manual

Team Members:

- Jovian Peng
- Strahinja Janjusevic
- Xoua Thao
- Hayes Martin

Customer:

Patrick Dowd, Phd, Cyber Science Department

Date: 03.28.2024

Version: 0.1

Table of Contents

System Introduction

Purpose of the User's Manual

Upon the conclusion of this manual, the user should be able to understand the fundamental operations of the Reputation Enforced Bayesian Network Analysis Tool (REBAT), and be able to run REBAT on datasets. The user will be able to set up the environment to run the REBAT. Although not critical to its operation, the user may additionally gain further insight regarding the inner workings of REBAT.

General Overview of the System Developed

The REBAT allows for the tracking and prediction of threat actor behavior. Packets, incoming traffic data, into the network are filtered using a reverse Reputation method. Packets originating from an Internet Protocol (IP) address with a bad reputation are collected. The collected packets are then analyzed by referencing the MITRE ATTACK Framework to label the packet with a corresponding Tactic, Technique, and Procedure (TTP).

System Set-Up

To set up and start using the Bayesian Network Analysis Tool, follow the detailed steps outlined below. Ensure that you complete each step in the order presented to guarantee a smooth setup process.

1. Download Necessary Files

Begin by downloading the required Python scripts for the system. These are the core components that will run the Bayesian analysis:

- BayesianGenerator.py: The main script that generates the Bayesian network based on provided data.
- Final_FormatFin1.py: Used for formatting and preparing the data for analysis.
- Updated_Classify.py: Responsible for classifying data points using machine learning models.
- PcapRepFinal.py: Analyzes network traffic data (PCAP files) and integrates reputation scores.

Ensure you save these files in a dedicated directory on your computer where you intend to run the analysis.

2. Obtain API Keys

<u>IPQualityScore API Key:</u>

You need to acquire a personal API key from IPQualityScore to enable the reputation scoring functionality:

1. Visit IPQualityScore.com.
2. Register or log in to create an account.
3. Navigate to the API section and generate a new API key.
4. Once you have your API key, open the PcapRepFinal.py file in a text editor.
5. Locate the placeholder for the IPQS API Key and replace it with your actual API key.

<u>OpenAI API Key:</u>

Similarly, for the classification functionality in Updated_Classify.py, you need an API key from OpenAI:

1. Go to OpenAI's website and sign up or log in.
2. Access the API section to generate your unique API key.
3. Open the Updated_Classify.py file in a text editor.
4. Find the designated spot for the OpenAI API Key and insert your key there.

3. Environment Setup

Ensure that Python 3.7 or higher is installed on your system. Additionally, install the necessary Python packages if they are not already installed. Open a terminal or command prompt and run the following command:

```
$ pip install networkx pgmpy matplotlib requests
```

This command will install NetworkX, pgmpy, matplotlib, and requests, which are essential for running the system, there might be some other needed libraries depending on your environment that you might have to install.

4. Running the program

To run the program make sure that you have a pcapng file in the directory, put that into the PcapRepFinal.py as a variable and run, this will find the reputation of every unique IP. After that run Updated_Classify.py with the file name of the CSV that you want to store the results and target pcapng file you have chosen before. After this run BayesianGenerator.py with an attacker name as the command line argument.

High-Level View of System

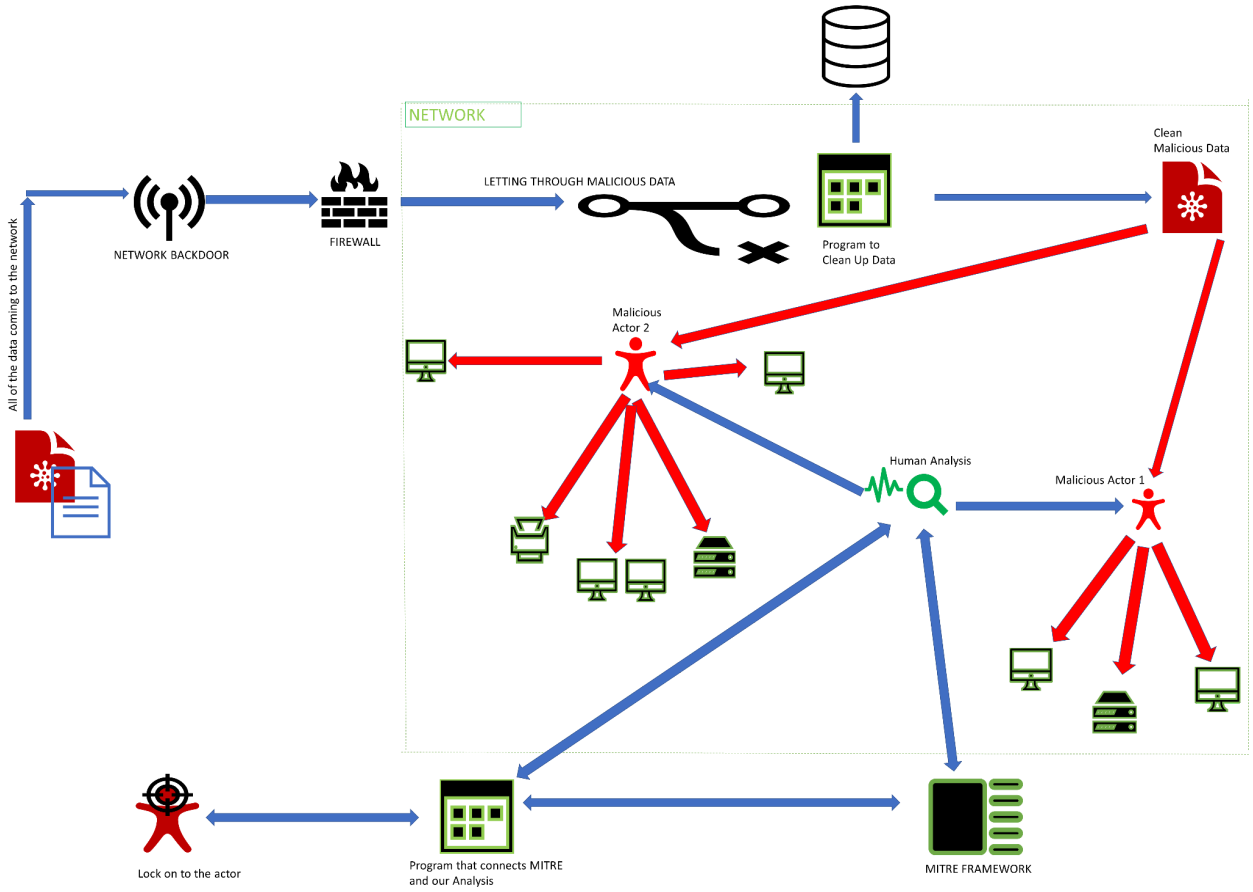


Figure 1a. High-Level Diagram of the System

The Reputation Enforced Bayesian Analysis Tool (REBAT) allows for the tracking and prediction of threat actor behavior. Figure 1a is a diagram of traffic entering a host network. Each component that REBAT interacts with is labeled within the diagram. Incoming traffic data is allowed through the firewall. Using the reverse Reputation method, packets are labeled for further analysis. Packets originating from an Internet Protocol (IP) address with a bad reputation are collected. The collected packets are then analyzed by referencing the MITRE ATT&CK Framework to label the packet with a corresponding Tactic, Technique, and Procedure (TTP). Once labeled, the actor is associated with the TTP and the information is inserted into the Bayseian network.

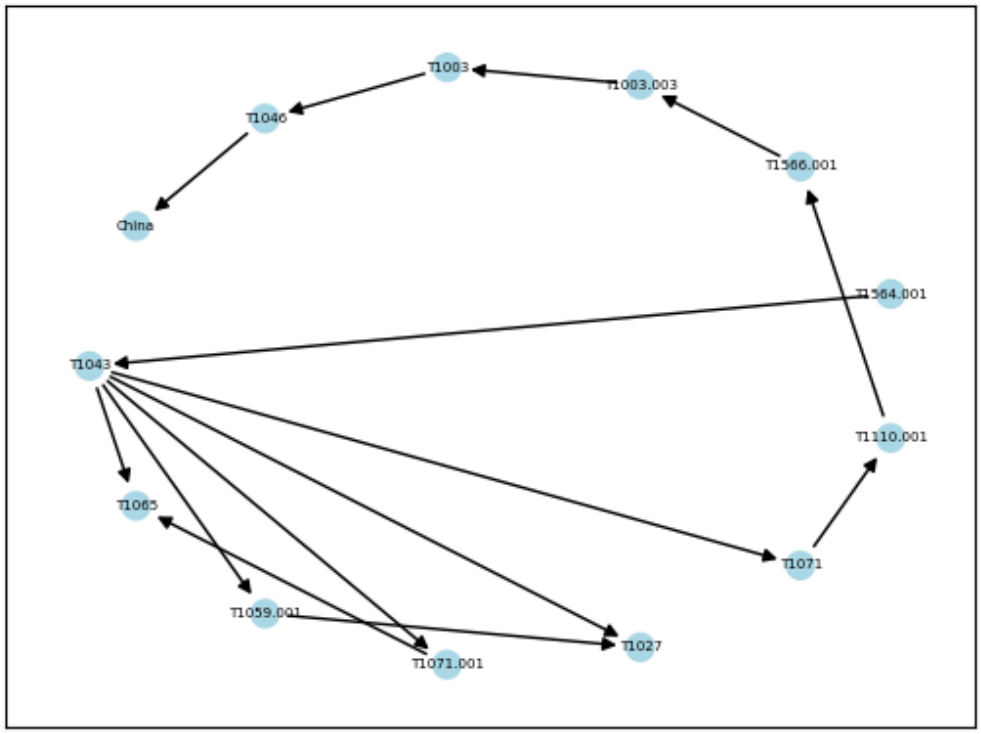
Quick-Start Tutorial

This section serves to give examples of the input and outputs of Reputation Enforced Bayesian Analysis Tool (REBAT) in use and offers a snapshot of the usage of REBAT.

1. Run

```
strajo22@WK5CG02180JZMID:/mnt/c/Users/m243006/Desktop/Capstone-20240118T195438Z-001/Capstone/Progs$ python3 PcapRepFinal.py

Processing Try.pcapng:
192.168.1.27
192.168.1.1
```



```
Cycle detected: [('T1046', 'T1046', 'forward')]
this is the cycle [('T1046', 'T1046', 'forward')]
Removing node: T1046
Node T1046 has been removed.
No cycle detected.
'China' node is isolated; it has no incoming edges.
Added edge from 'T1046' to 'China'
Model is correctly specified.
['T1564.001']
1. Input a single TTP in the format 'TXXXX' or 'TXXXX.
2. Input a sequence of TTPs seperated by comma
3. Print the existing HashTavle
Enter your choice or type 'quit' to exit: 2
Enter a sequence of TTPs separated by commas: T1059.00
This is the result
+----------+--------------+
| China    |   phi(China) |
+==========+==============+
| China(0) |       0.0000 |
+----------+--------------+
| China(1) |       1.0000 |
+----------+--------------+
```

2. Run

Running PcapRepFin.py

```
64          pcap_file_path = "N-0.pcapng"
65          print(f"\nProcessing {pcap_file_path}:")
66          print_ips_and_reputation_from_pcap(pcap_file_path, unique_ips)
67
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Processing N-0.pcapng:

IP Address: 192.168.159.1
Reputation: 70 (Fraud Score)
192.168.159.1

IP Address: 192.168.159.131
Reputation: 70 (Fraud Score)
```

Running UpdatedClassify.py

```
<tr><td>

<p><label for="login">Login:</label><br />
<input type="text" id="login" name="login"></p>

</td>

<td width="5"></td>

<td>

<p><label for="email">E-mail:</label><br />
<input type="text" id="email" name="email" size="30"></p>

</td></tr>

<tr><td>
```

Getting stored into CSV

```
IP,Label,Description,Technique Number,Payload

192.168.159.1,False,,,"GET /bWAPP/ HTTP/1.1
Host: 192.168.159.131
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Sa
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-ex
Referer: http://192.168.159.131/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=81904e520814f311d8f6e10bbfd1a5fe
```

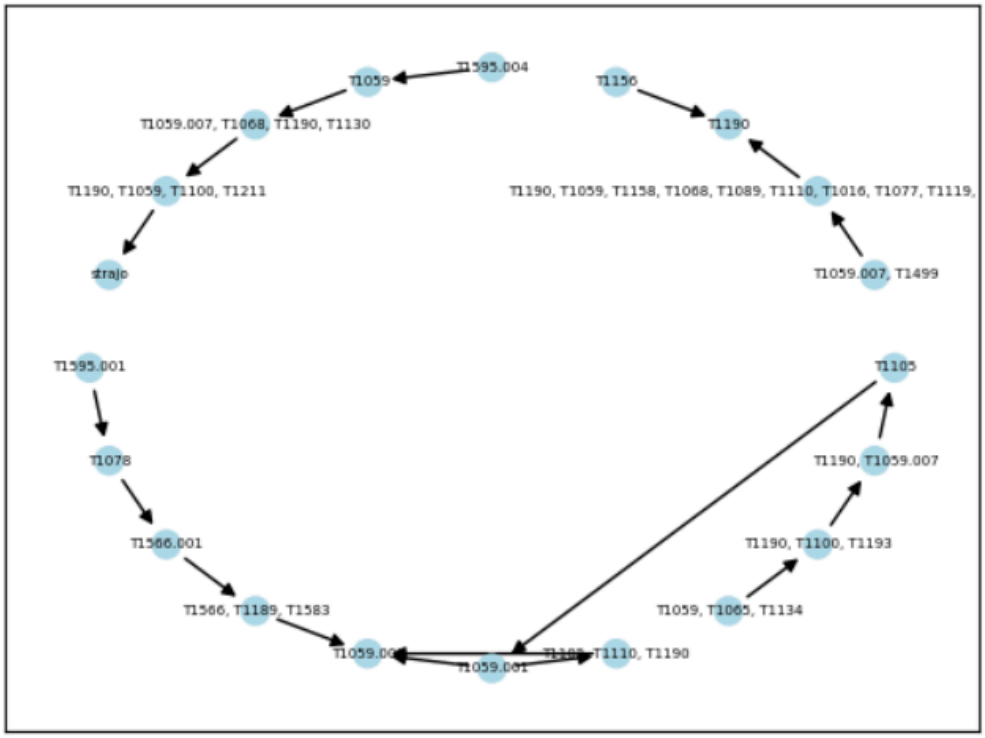
Running BayesianGenerator on the results

```
strajo22@WK5CG02180JZMID:/mnt/c/Users/m243006/Desktop/Capstone-20240118T195438Z-001/Capstone/Progs$ python3 BayesianGenerator.py strajo
['T1046', 'T1046', 'T1046', 'T1046', 'T1595.001', 'T1078', 'T1566.001', 'T1566, T1189, T1583', 'T1059.007', 'T1059.001', 'T1059.007', 'T105
9, T1065, T1134', 'T1190, T1100, T1193', 'T1190, T1059.007', 'T1105', 'T1059.001', 'T1059.007', 'T1059.007, T1499', 'T1190, T1059, T1158, T
1068, T1089, T1110, T1016, T1077, T1119, T1105, T1041, T1490', 'T1190', 'T1059.001', 'T1102, T1110, T1190', 'T1059.007', 'T1156', 'T1190',
'T1595.004', 'T1059', 'T1059.007, T1068, T1190, T1130', 'T1190, T1059, T1100, T1211', 'T1190', 'T1059.007', 'strajo']
['T1046', 'T1595.001', 'T1078', 'T1566.001', 'T1566, T1189, T1583', 'T1059.007', 'T1059.001', 'T1059, T1065, T1134', 'T1190, T1100, T1193',
 'T1190, T1059.007', 'T1105', 'T1059.007, T1499', 'T1190, T1059, T1158, T1068, T1089, T1110, T1016, T1077, T1119, T1105, T1041, T1490', 'T1
190', 'T1102, T1110, T1190', 'T1156', 'T1595.004', 'T1059', 'T1059.007, T1068, T1190, T1130', 'T1190, T1059, T1100, T1211']
"JavaScript"
"Exploit Public-Facing Application Mitigation"
"Remote File Copy Mitigation"
"Spearphishing Attachment"
"Network Service Scanning Mitigation"
```

Getting Probabilities:

```
['T1595.001', 'T1059, T1065, T1134', 'T1059.007, T1499', 'T1156', 'T1595.004']
1. Input a single TTP in the format 'TXXXX' or 'TXXXX.YYY'
2. Input a sequence of TTPs seperated by comma
3. Print the existing HashTavle
Enter your choice or type 'quit' to exit: 1
Enter a TTP: T1059.007
+-----------+---------------+
| strajo    |   phi(strajo) |
+===========+===============+
| strajo(0) |        0.5000 |
+-----------+---------------+
| strajo(1) |        0.5000 |
+-----------+---------------+
```

Network is built:



3. Run

Run PcapRepFin.py on the merged_output file

```
64          pcap_file_path = "merged_output.pcapng"
65          print(f"\nProcessing {pcap_file_path}:")
66          print_ips_and_reputation_from_pcap(pcap_file_path, unique_ips)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS wsl

```
IP Address: 172.20.6.1
Reputation: 70 (Fraud Score)
172.20.6.1
```

Run Updated_Classify.py on the file

```
strajo22@WK5CG02180JZMID:/mnt/c/Users/m243006/Desktop/Capstone-20240118T195438Z-001/Capstone/Progs$ python3 Updated_Classify.py
Timestamp: 2015-08-06 03:08:21.431389 Packet Length: 78 bytes
Ethernet Frame: Source MAC: a4:5e:60:e5:da:95 Destination MAC: 44:1e:a1:c2:36:46 Ethernet Type: 2048
IP: Source IP: 172.20.6.1 Destination IP: 42.96.141.35 IP Protocol: 6 TTL: 64 DF (Don't Fragment): 1 MF (More Fragments): 0 Fragment Offset
: 0
TCP: Source Port: 51500 Destination Port: 80 TCP Flags: SYN No TCP data
```

Get the new CSV

```
42.96.141.35,True,"This packet is malicious as it contains a meta refresh tag that redirects to a suspicious URL, whi
Content-Type: text/html
Content-Length: 206
Connection: close
```

Run BayesanGenerator with new CSV file as the target

```
+----------+--------------+
| china    |   phi(china) |
+==========+==============+
| china(0) |       0.5000 |
+----------+--------------+
| china(1) |       0.5000 |
+----------+--------------+
1. Input a single TTP in the format 'TXXXX' or 'TXXXX.YYY'
2. Input a sequence of TTPs seperated by comma
3. Print the existing HashTavle
Enter your choice or type 'quit' to exit:
```

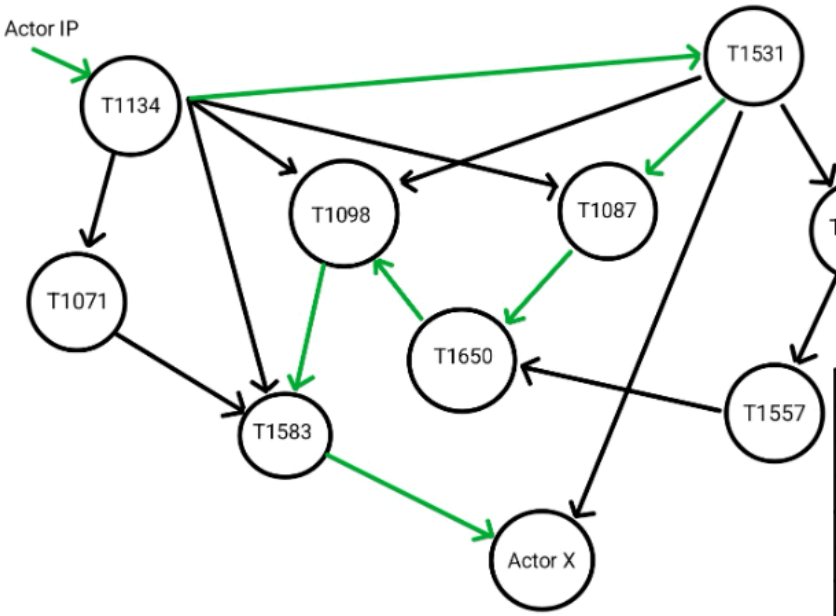
Detailed View of System

REBAT uses a systematic refinement of acquired data in order to increase the likelihood of obtaining malicious content. The initial step for data collection requires the disabling or bypass of the firewall to allow for malicious traffic to generate malicious data to be tracked. An example PCAP that's known to contain malicious activity would suffice.

Focusing on the originating IP of the packets, REBAT uses a reverse Reputation method to filter and label packets as malicious. The Reputation method uses a community database made from numerous individuals to identify bad IPs, or IPs that have a reputation of malicious activity. Instead of filtering out traffic that originates from IPs with a bad reputation, the traffic from those IPs are instead labeled for further analysis. The next step is to analyze the traffic to filter out noise and identify threat actor behavior (i.e. tradecraft) using the MITRE ATT&CK framework. The MITRE ATT&CKframework is a collection of tactics and techniques designed to identify attack attribution and objectives, and assess an organization's risk. An analysis performed by CHATGPT will assess the packet's contents and assign a TTP to the packet. The next step is to attribute a sequence of threat actor tradecraft as a tagged actor, and then log the changes in the tagged threat actor tradecraft to achieve target continuity. See figure 2a for the data flow of packets.

A Bayesian network is a probabilistic graphical model that uses a directed acyclic graph to represent variables and their conditional dependencies. Using a Bayesian network, the threat actor's profile is tracked by the insertion and updating of nodes within the network. With existing TTPs performed by the threat actor, predictions of the next action branch off of existing nodes with associated probabilities. In the case of an unpredicted TTP, that new TTP will be inserted into the Bayesian Network and prediction nodes will then be generated off of the new node. As the threat actor performs actions, probabilities are constantly updated as new nodes may be generated or when the actor performs a predicted TTP. See figure 2a for an example Bayesian Network consisting of TTPs for a threat actor.

A threat actor's profile can be saved in a "pickle" file and later referenced for analysis. Loading or switching the context of the Bayesian network will allow the user to query information regarding the threat actor. Once a threat actor is loaded into the Bayesian Network, the actor's profile may be updated as necessary as the TTPs the actor takes are analyzed. See figure 2c for information regarding how REBAT loads threat actors onto the Bayesian Network.



ID	Technique
T1134	Access token manipulation
T1531	Account access removal
T1087	Account discovery
T1098	Account manipulation
T1650	Acquire access
T1583	Acquire infrastructure
T1595	Active scanning
T1557	Adversary in the middle
T1071	Application Layer protocol

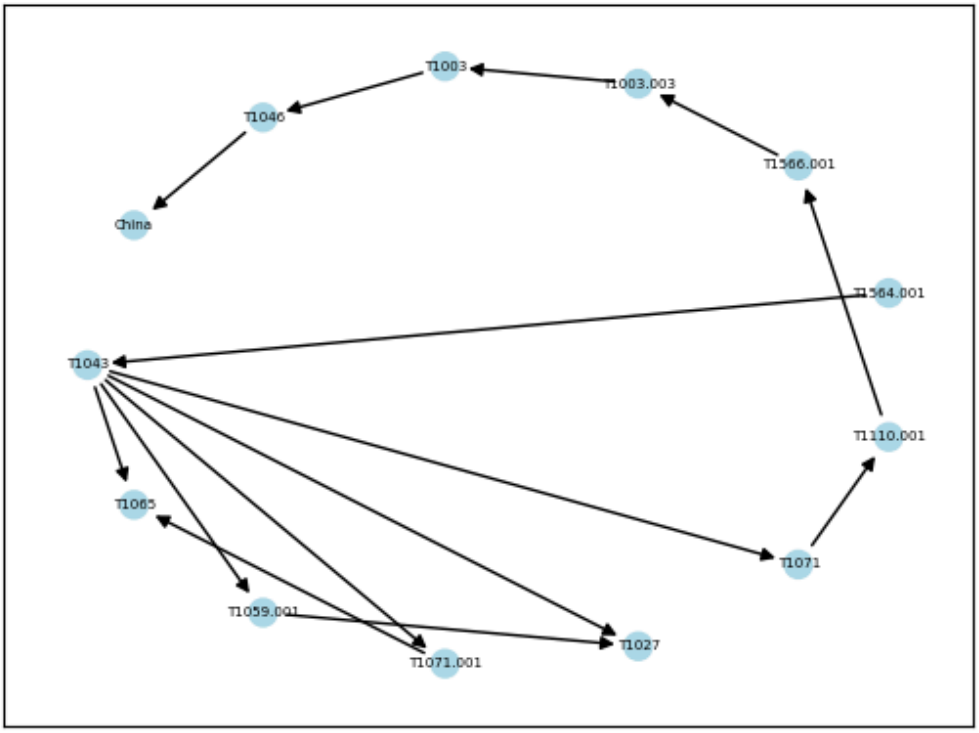


Figure 2a. Network of TTPs : Data Flow Diagram and its Implementation

The first diagram is a data flow diagram that shows how the Bayesian network will be implemented. For each actor IP address, the tactics, techniques, and procedures (TTPs) will be identified by their MITRE Attack ID and technique and be implemented as nodes in the Bayesian network. As a result, multiple TTPs being identified from a specific actor will be modeled as a path in the network. Right below the data flow diagram is the Bayesian Network generated by REBAT on a dataset.

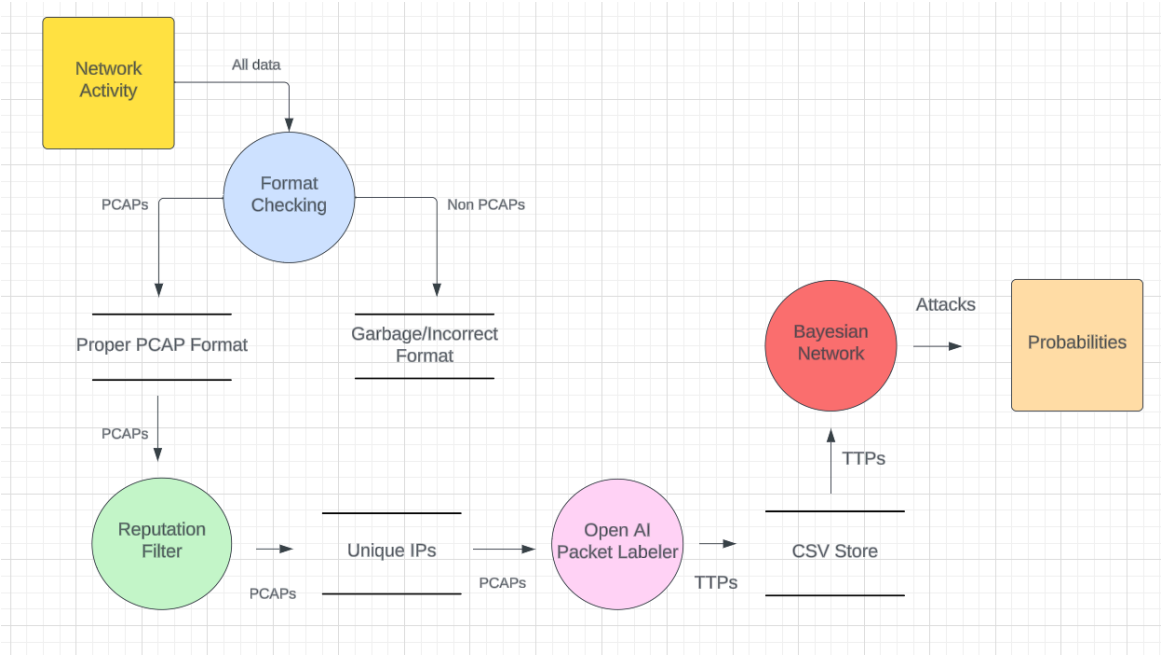


Figure 2b. Path of Packets: Data Flow Diagram

This data flow diagram presents an overview of the entire project from the raw data to the output of attack probabilities. The diagram shows the path of the data and how it is processed via format checking, filtered via the reputation method, and ultimately sorted, labeled, and stored by TTPs in the Bayesian network which ultimately outputs the probabilities.

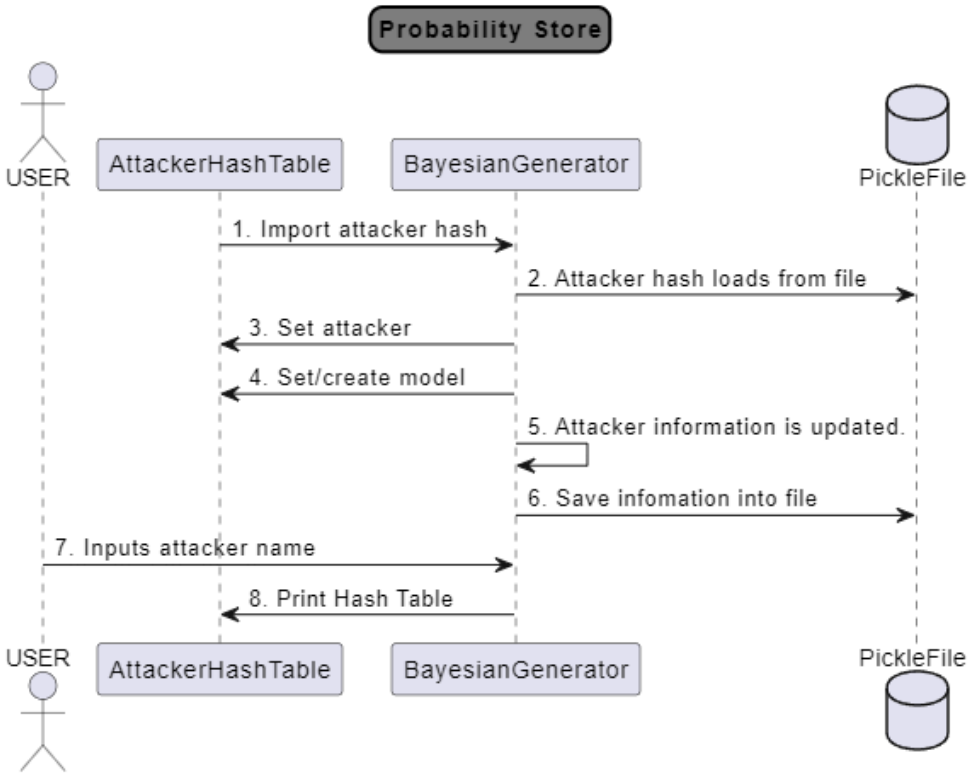


Figure 2c. Bayesian Generator: Sequence Diagram

The Bayesian Generator Sequence Diagram (Figure 2c) models the steps taken to create or update an actor's information, the TTPs associated with the actor. The actor is selected via their associated hash and their date is then loaded into the context of the Bayesian Network. Additions and modifications to the Bayesian Network are then saved and can be later queried.

```
Enter your choice or type 'quit' to exit: 3
Model for Johan:
+--------------+----------+----------+
| T1040        | T1040(0) | T1040(1) |
+--------------+----------+----------+
| T1005.002(0) | 1.0      | 0.0      |
+--------------+----------+----------+
| T1005.002(1) | 0.0      | 1.0      |
+--------------+----------+----------+
+--------------+--------------+--------------+
| T1005.002    | T1005.002(0) | T1005.002(1) |
+--------------+--------------+--------------+
| T1210.004(0) | 1.0          | 0.0          |
+--------------+--------------+--------------+
| T1210.004(1) | 0.0          | 1.0          |
+--------------+--------------+--------------+
+--------------+--------------+--------------+
| T1210.004    | T1210.004(0) | T1210.004(1) |
+--------------+--------------+--------------+
| T1046.003(0) | 0.5          | 0.5          |
+--------------+--------------+--------------+
```

Functional Requirements Trace Table

The table below contains the list of requirements promised to or required by the customer. Each requirement consists of test cases and their expected results, use cases, design artifacts used for modeling, an effort value based on complexity, time of completion, and the customer's acknowledgement of completion.

Functional Requirement	Set of Acceptance Plan Test Cases	Build	Use Cases	Design Artifacts	Effort Value	Testing Status & Milestone Number	Customer's Initials
1. Reputation Classification System: IP addresses from the PCAP parsed data shall be passed through the reputation database and data packet IPs with poor reputation scores shall be stored appropriately. Primary: MIDN Martin	1.1 An IP address from a PCAP data packet is assigned a poor reputation score after being passed through the reputation database. Expected result → Data packet IP address is stored into a text file (Normal).	1	Insert data into System Classify PCAP packets as malicious	Data Packet Classification: Data Flow Diagram	13	Completed for Milestone 4	pd
2. Parse PCAP: System shall label and store PCAP data as true (Malicious) or false (Not Malicious).	2.1 A properly formatted malicious PCAP data packet is presented. Expected result → The PCAP is	1	Classify PCAP packets as malicious Format PCAP data	PCAP Formatting: Data Flow Diagram	8	Completed for Milestone 4	pd

Primary: MIDN Thao	marked true and stored appropriately (Normal).						
3. Packet Mapping: Techniques, tactics, and procedures (TTPs) are extracted from "Malicious" labeled data via human analysis. Primary: MIDN Janjusevic	3.1. Data is labeled as "Malicious" by the packet classification system. Expected result → TTPs are extracted from malicious data.	1	Extract TTPs from packet data	TTP Extraction: Data Flow Diagram	5	Completed for Milestone 4	pd
4. MITRE ATT&CK framework API: System shall perform queries on the MITRE ATT&CK framework. Primary: MIDN Peng	4.1 A query is given to the API. Expected result → Associated data is returned. (Normal)	1	Query Data from database	MITRE Query: Entity Relationship Diagram	5	Completed for Milestone 5	pd
5. Data Store: System shall store Data based on IP addresses; stored in	5.1 IP address is stored in data structure based off of its determined reputation	2	Log Data	IP address storage: Data Flow Diagram	5	Completed for Milestone 7	pd

structure depending on threat level and reputation Primary: MIDN Martin	Expected result → IPs with good reputation are stored in one instance of a structure and IPs with bad reputation are stored in a separate structure (Normal)						
6. Bayesian Network Verification: System shall verify node is in proper format for Bayesian Network insertion Primary: MIDN Janjusevic	6.1 Node data is received in proper format. Expected result → Node put in queue for addition to network. (Normal) 6.2 Node data is received and not in proper format. Expected result → Node is discarded and error message is printed. Move on to next node (Normal)	3	Build Bayesian Network	Bayesian Network Node: Network Diagram	13	Completed for Milestone 7	pd
7. Bayesian Network Insertion:	7.1 Existing node data is received.	3	Build Bayesian Network	Insert Network Node:	8	Completed for Milestone 7	pd

System shall insert nodes from the database into the Bayesian Network. Primary: MIDN Janjusevic	Expected result → Node weight is increased. (Normal) 7.2 New node data is received. Expected result → New node is added to the network. (Normal) 7.3 Improper Bayesian data format is received. Expected result → System shall return an appropriate error message. (Abnormal)			Network Diagram			
8. Implementation of Bayesian Network: System shall run a live Bayesnian Primary: MIDN Janjusevic	8.1 Data received is identified as an existing node in the network. Expected result → Node weight is adjusted. (Normal) 8.2 Data received is identified as a new node.	4	Determine Threat Actor Tactic, Technique, or Plan	Complete Bayesian Network: Network Diagram	13	Completed for Milestone 8	pd

	Expected result: → A new node with the attack and its unique TTPs is added to the network. (Normal) 8.3 Improper Bayesian unit data format is received. Expected result → System shall return an appropriate error message. (Abnormal)						
9. Malicious Data Reviewer: System shall output probabilities of potential threat actor actions. Primary: MIDN Thao	9.1 Query for node probabilities is inputted. Expected result → Queried node probabilities are outputted. (Normal) 9.2 Improper query format is received. Expected result → System shall return an appropriate error message. (Abnormal)	4	Build Bayesian Network	Probability Calculator: Decision Tree	13	Completed for Milestone 8	pd

10. Probability Store: System stores probability of address/actor committing malicious attack in a hash table Primary: MIDN Martin	10.1 Address is given to be stored in structure Expected result → New entry is created in hashmap with key being the threat actor and value is the Bayesian model for that threat actor (Normal) 10.2 Set of TTPs received is not in the network.. Expected result → System shall return an appropriate error message. (Abnormal)	5	Insert data into system.	Data Packet Classification: Data Flow Diagram	8	Completed for Milestone 9	pd
11. Look up Threat Actor: System shall output probabilities of potential threat actors given a set of movements performed.	11.1 Given a set of movements, return the probability that the given threat actor carried out those movements. Expected	6	Query Data from database	Threat Actor Identifier: Data Flow Diagram	8	Completed for Milestone 9	pd

Primary: MIDN Janjusevic	result → Given a set of movements, the system returns the Threat actor probability.. (Normal) 11.2 No Malicious Activity Detected Expected result → System shall return an appropriate error message. (Abnormal)						

Customer Meeting Summary:

During our customer meeting on 27 March 2024, we went through our program with our customer as well as the entire user's manual, explaining each part of the manual in detail. Overall, our customer was very pleased and satisfied with the content of the user's manual and the functionality and result of our code. After pointing out certain points of improvement, (mentioned in the customer's feedback section) he also suggested that each of the members individually play and experiment with the system in order to uncover more possible deficiencies either corrective or perfective.

Index

User Manual Appendices

Quick Reference Card

To run the program:

1. Make sure that you have a pcapng file in the directory
2. Put that into the PcapRepFinal.py as a variable and run, this will find the reputation of every unique IP.
3. Run Updated_Classify.py with the file name of the CSV that you want to store the results and target pcapng file you have chosen before.
4. Run BayesianGenerator.py with an attacker name as the command line argument.

System Requirements

1. A computer
2. A network or PCAP data
3. Python 3.10.12

Installation

To set up and start using the Bayesian Network Analysis Tool, follow the detailed steps outlined below. Ensure that you complete each step in the order presented to guarantee a smooth setup process.

1. Download Necessary Files

Begin by downloading the required Python scripts for the system. These are the core components that will run the Bayesian analysis:

- BayesianGenerator.py: The main script that generates the Bayesian network based on provided data.
- Final_FormatFin1.py: Used for formatting and preparing the data for analysis.
- Updated_Classify.py: Responsible for classifying data points using machine learning models.
- PcapRepFinal.py: Analyzes network traffic data (PCAP files) and integrates reputation scores.

Ensure you save these files in a dedicated directory on your computer where you intend to run the analysis.

2. Obtain API Keys

<u>IPQualityScore API Key:</u>

You need to acquire a personal API key from IPQualityScore to enable the reputation scoring functionality:

6. Visit IPQualityScore.com.
7. Register or log in to create an account.
8. Navigate to the API section and generate a new API key.
9. Once you have your API key, open the PcapRepFinal.py file in a text editor.
10. Locate the placeholder for the IPQS API Key and replace it with your actual API key.

<u>OpenAI API Key:</u>

Similarly, for the classification functionality in Updated_Classify.py, you need an API key from OpenAI:

5. Go to OpenAI's website and sign up or log in.
6. Access the API section to generate your unique API key.
7. Open the Updated_Classify.py file in a text editor.
8. Find the designated spot for the OpenAI API Key and insert your key there.

3. Environment Setup

Ensure that Python 3.7 or higher is installed on your system. Additionally, install the necessary Python packages if they are not already installed. Open a terminal or command prompt and run the following command:

```
$ pip install networkx pgmpy matplotlib requests
```

This command will install NetworkX, pgmpy, matplotlib, and requests, which are essential for running the system, there might be some other needed libraries depending on your environment that you might have to install.

4. Running the program

To run the program make sure that you have a pcapng file in the directory, put that into the PcapRepFinal.py as a variable and run, this will find the reputation of every unique IP. After that run Updated_Classify.py with the file name of the CSV that you want to store the results and target pcapng file you have chosen before. After this run BayesianGenerator.py with an attacker name as the command line argument.

Developer's Information

MIDN 1/C Strahinja Janjusevic, m243006@usna.edu
MIDN 1/C Jovian Peng, m245028@usna.edu
MIDN 1/C Xoua Thao, m246384@usna.edu
MIDN 1/C Hayes Martin, m243960@usna.edu