Threat Attribution System

By

Hadia Saif Khan

Maryam Haq Khattak

Syed Ameer Abdullah

Zojaja Arif

Supervised by:

Dr. Waleed Bin Shahid

Submitted to the faculty of Department of Electrical Engineering,
Military College of Signals, National University of Sciences and Technology, Islamabad,
in partial fulfillment for the requirements of B.E Degree in Electrical (Telecom) Engineering.

June 2023

In the name of ALLAH, the Most benevolent, the Most Courteous

CERTIFICATE OF CORRECTNESS AND APPROVAL

This is to officially state that the thesis work contained in this report

"Threat Attribution System"

is carried out by

Hadia Saif Khan, Maryam Haq Khattak, Syed Ameer Abdullah, and Zojaja Arif

under my supervision and that in my judgement, it is fully ample, in scope and excellence, for the degree of Bachelor of Electrical (Telecom.) Engineering in Military College of Signals, National University of Sciences and Technology (NUST), Islamabad.

Approved by

Supervisor

Dr. Waleed Bin Shahid

Department of IS, MCS

Date: April 27th 2023

DECLARATION OF ORIGINALITY

We hereby declare that no portion of work presented in this thesis has been submitted in support of another award or qualification in either this institute or anywhere else.

ACKNOWLEDGEMENTS

Allah Subhan'Wa'Tala is the sole guidance in all domains.
Our parents, colleagues, and especially our supervisors, Assistant Professor Dr. Waleed Bin Shahid and Assistant Professor Aimen Aakif, have been invaluable sources of guidance throughout our journey. Their unwavering support and mentorship have been instrumental in our growth and development. They have provided us with direction, knowledge, and wisdom, pushing us to reach our full potential.
Despite challenges, the group's unwavering dedication, mutual support, and collective perseverance enabled us to overcome obstacles and achieve exceptional outcomes. Gratitude is expressed for invaluable guidance and collaborative teamwork.

Plagiarism Certificate (Turnitin Report)

This thesis has 8% similarity index. Turnitin report endorsed by Supervisor is attached.

Hadia Saif Khan

NUST Serial no 296980

Maryam Haq Khattak

NUST Serial no 297335

Syed Ameer Abdullah

NUST Serial no 293484

Zojaja Arif

NUST Serial no 295726

Signature of Supervisor

ABSTRACT

Over the years, computers have undergone significant improvements, changing from bulky, expensive machines with few capabilities to sleek, potent gadgets that are portable. Nearly 97% of Americans, according to a survey, own a personal device. Personal computers and cellphones are far more susceptible to cyberattacks since they store private and sensitive information. Malicious software, commonly known as malware, has the ability to disrupt the entire network while stealing valuable information. Naturally, such attacks frequently go unnoticed. As a result, understanding malware attribution and threat-actor attribution is essential for spotting and evaluating criminal malicious activity.

Malware attribution is the process of mapping a cyberattack to its threat actor. An individual, group, or organization that poses a threat to a computer system or network is referred to as a threat actor. The existing attribution systems use code similarity in APT malwares to analyze Indicators of Compromise (IOC data). With the emergence of polymorphic [12] malwares, attackers generate new signatures through slight code variations. Thus, code similarity techniques are inefficient for attribution. Our project's goal is to identify the threat actors behind Windows malware using systems event logs and registry files. It uses an anomaly-based approach to identify and classify malware. To begin with, it makes use of Sysmon to produce Windows-based operating system logs. For the purpose of identifying activity, the logs are analyzed to separate harmful from non-malicious actions. Secondly, data registries are a useful tool for locating and minimizing risks in the surroundings of threat actors. A data registry is a centralized database that houses details about user preferences, system setup, application settings, and other crucial information that an operating system or application uses. Natural Language Processing algorithms are used for classification. Malware attribution system ensures data integrity and security by timely attribution of the threat actor group.

Keywords: Malware, Malicious Software, Data Privacy and Security, Personal Gadgets, Cyber-attacks, Cyber-risk, BERT, Machine Learning, Indicators of Compromise (IOC), Data Registry

Table of Contents

List of Figures

CHAPTER 1: INTRODUCTION

Chapter 1: Introduction

Security has long been a need of humans. It has cultivated a sense of stability in humans; and links to their physical and emotional development of traits. The digital world of '0s' and '1s' reflects a similar ideology.

With the advent of computers and the internet, the world has shifted from an analog to a digitally enabled globe. The rapid growth has created a vast network relaying important information and instructions; crucial for system functioning. These computer-based systems have become an essential human-life component, from the military to the healthcare systems. In today's world, hardly any office or hospital functions without these technological appliances.

Cybersecurity aims to protect valuable data from potential malicious attacks. The present-day problems demand automatic solutions that are efficient. Data breach problems due to malware should also be dealt with solutions based on the latest technology.

1.1 Overview

Cybersecurity is an under looked concept in the low-and middle-income countries. These countries have recently started investing in digital infrastructure to boost their economy and relying on it for crucial system operations. This enhanced dependency on digital infrastructure requires efficient and swift cybersecurity solutions. The lack of these solutions led to the Colonial Pipeline ransomware attack: a large-scale attack on an oil and gas company. Only a comprehensive analysis of malware attacks can prevent future hazards. A study shows that the manufacturing and finance sectors were more vulnerable to malicious attacks in 2022.

It is the need of the hour to fashion some policy or mechanism to avoid these malware attacks. Hence in our proposed system of malware attribution, we focus on prevention through a detailed analysis of malware characteristics.

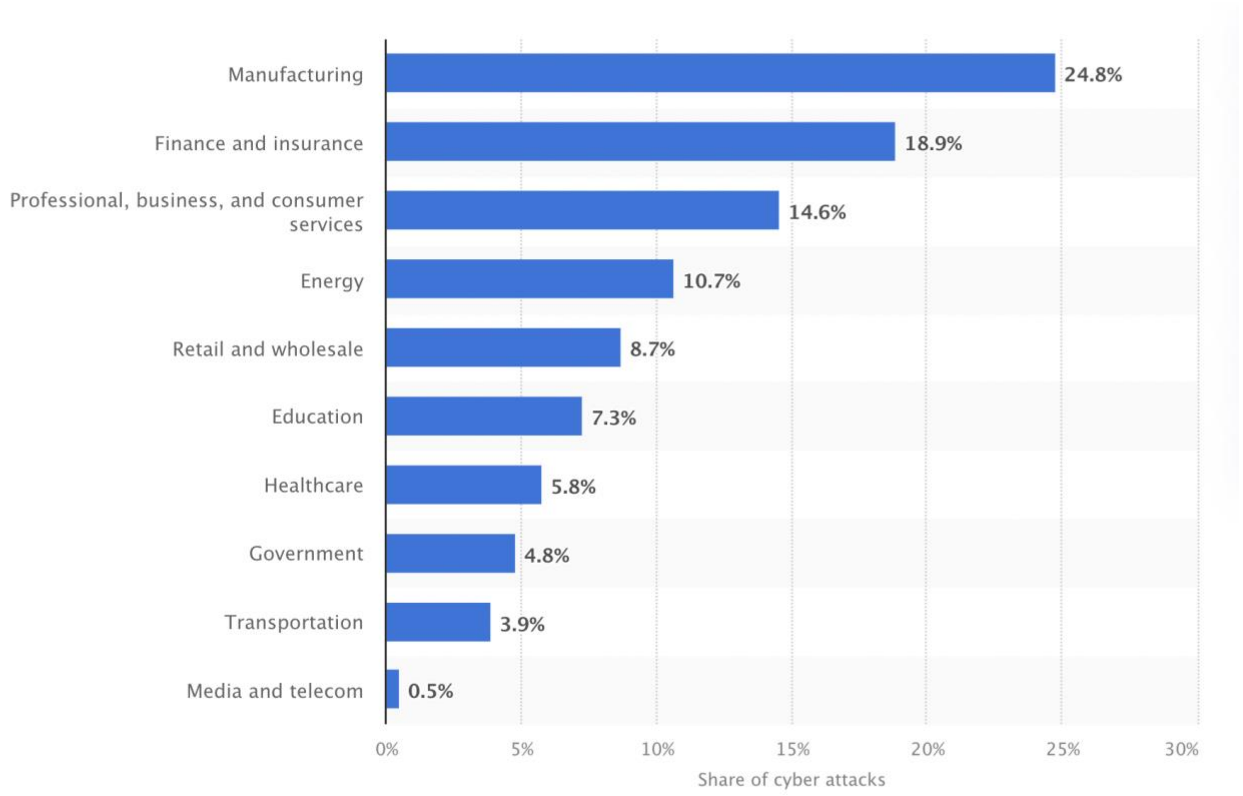


Figure 1 Distribution of cyberattacks across worldwide industries in 2022 [9]

1.2 Problem Statement

Cyberattacks can hurt a corporation by jeopardizing its systems, networks, devices, and end users' confidentiality, integrity, and availability. Malware assaults are one of the most prevalent types of cyberattacks. Malware causes a variety of harm, including the theft of private information and the disruption of network functions. They routinely bypass conventional protection tools like anti-viruses.

In the current cybersecurity environment, the creation of a malware and threat actor attribution system has become crucial. By linking new malware samples or campaigns to old or new campaigns, the system hopes to find and identify any novel malware samples or campaigns. The goal is to develop a system that can analyze, identify, and correlate malware capabilities with the already-existing dataset to improve threat intelligence. The system will be able to examine malware behaviour and spot trends that could be used to connect it to earlier operations.

A thorough understanding of the capabilities of threat actors and the behaviour of malware will be necessary for the system's implementation. To offer another line of defense against cutting-edge attacks, it will also need to interact with the already-existing security architecture. Overall, a company's cybersecurity posture and defense against cutting-edge threats will be greatly improved by the malware and threat actor attribution system.

1.3 Proposed Solution

The proposed system is based on an attribution engine that has been improved using several examples of current malware and is aimed to categorize unknown malware samples using machine learning techniques. To assess incoming virus behaviour and establish attribution, the engine looks into the Windows Event Logs and data registry files. It's crucial to remember that the suggested architecture is more of a method for attribution rather than a malware detection engine. The framework makes use of machine learning methods - BERT transformer algorithm, and is not restricted to a single form of malware, enabling continuous attribution.

- Collection of malware samples.
- Collection of Windows Event Logs generated by the malware samples.
- Collection of Registry Files.
- Application of NLP algorithm

1.4 Working Principle

The project mainly works on the principles of cybersecurity with machine learning algorithms. The project is divided into different modulus and every module is inter-woven with the next module. The list of modules is as under:

- Datasets and annotations
- Dataset training and processing
- Feature Extraction
- Integration

1.4.1 Datasets and annotations:

The integral part of the project is the preparation of datasets. In order to complete this project, data must be gathered from a variety of sources, including network logs, system logs, registry files and other databases pertaining to cybersecurity.

1.4.2 Dataset training and processing:

The prepared dataset is used as input to train models using machine learning for attribution. This module chooses the features that are pertinent to getting the intended result.

1.4.3 Feature Set

A feature set is a group of features that are taken from raw data in machine learning and data analysis and used to represent each data point as a numerical vector.

An attribute or characteristic of the data that is quantifiable and useful to the analysis of the problem is referred to as a feature. The frequency with which certain words or phrases appear in an email might be regarded as a feature when spam email detection is being performed.

The feature set is crucial since it serves as the foundation for the machine learning model that will be applied to predict or categorize data. The effectiveness and performance of the model may be significantly impacted by the features' relevancy and quality. As a result, one of the most important steps in the machine learning process is frequently choosing and extracting the most valuable characteristics.

1.4.4 Integration:

The different modules are then integrated in to one stand-alone entity. This stand-alone entity is essential for a compact solution.

1.5 Objectives

1.5.1 Creation of Malware Database for Researches

Building an effective malware database is a key component of creating cybersecurity solutions. Developing such a database, which would contain examples of dangerous software from various malware families, is the goal of this project. The database will also include the extracted log files produced by these malware samples, which will offer important details about how these threats behave and act.

Researchers can better understand how these threats operate and spot patterns and trends in their behaviour by gathering and analyzing data from these malware samples. This can enhance the efficacy of current cybersecurity solutions and provide information for the creation of new ones.

1.5.2 Academic Objectives:

- Understanding the underlying ideas and approaches used to analyze malware, such as code analysis, behaviour analysis, and memory forensics, entails becoming familiar with its principles and procedures.
- Investigating malware attacks include understanding how to find the source of an attack's origin and cause as well as creating efficient treatment plans.
- Studying how to reverse engineer malware to comprehend its capabilities, behaviour, and evasion tactics is necessary to become proficient in this field.
- Keeping abreast with the most recent malware trends and methods attackers employ to avoid detection and compromise systems.

1.6 Scope

The ability to identify and stop attacks in real-time makes this project applicable to a variety of fields, including e-commerce, organizations, the healthcare industry, and the nation's overall security sector. It can lessen breached privacy in banks and safeguard additional sensitive data.

The project's main goal is to attribute malware swiftly while supplying resistance against attacker evasion. Therefore, we intend to use this project to integrate our theoretical knowledge with real-world experiences to enhance the security of network systems. With its machine learning detection system and a raspberry gateway to analyze attacks, this project seeks to develop a comprehensive security solution.

For development of Threat Attribution System, we will be requiring sound knowledge of:

- Programming (Python)
- Network Security (Attacks, Defences, Security Architecture)
- Malware Disassembler, Debuggers

For attribution of malware, we will be requiring hand-on expertise of:

- Sysmon Tool
- Log Extraction and Analysis
- Sandbox Solutions
- Python Scripting

1.7 Deliverables

The deliverable of this project is a system that attributes malwares.

1.8 Relevant Sustainable Development Goals

This project aligns with SDG 9: Industry Innovation and Infrastructure. Malicious actors launch deadly cyberattacks against vital infrastructure, including businesses, humanitarian NGOs, hospitals, clinics, and labs in an effort to demand ransom payments, exfiltrate data, or otherwise disrupt operations, endangering not just the safety of the data but also the lives of the people within. The faith in international and national organizations will decrease as a result of ongoing cyberattacks that will worsen disputes. We want to provide the groundwork for a peaceful internet that is accessible to everyone, everywhere.

1.9 Structure of Thesis

Chapter 2 contains the literature review and the background and analysis study this thesis is based upon.

Chapter 3 contains the design and development of the project.

Chapter 4 contains the result of the project.

Chapter 5 contains the conclusion of the project.

Chapter 6 highlights the future work needed to be done for the commercialization of this project.

1.10 Chapter Summary

The thesis's introductory chapter gives a brief summary of the study's problem statement, objectives, methods, importance, and thesis structure. It establishes the context, highlights the importance, and sets the direction for the rest of the thesis.

CHAPTER 2: LITERATURE REVIEW

Chapter 2: Literature Review

Overview

This chapter deals with comprehensive details of Malware attribution offered worldwide, their limitations and the uniqueness of our proposed solution.

2.1 Introduction

The thesis aims to discuss the preliminary research conducted before the project's design and development phase. This section provides a concise overview of the current attribution systems.

2.2 Project Domain

Malicious software is designed to cause harm to computer systems and users by gaining unauthorized access to personal information inserting harmful code word deceiving users to extract money malware creators are always looking for vulnerabilities to exploit and they frequently update their code to evade detection by malware detection software. Malware developers often attempt to disguise their malicious code as benign to avoid detection, making recognition a challenging task.

2.3 Industrial Background

Today’s digital era greatly relies on exchanging information between different sectors and firms; therefore, information security [14] is crucial to protect any organization’s data assets.

A significant increase in the occurrence of malware attacks and ever-changing threat to the community has been observed in today’s world, which has led the researchers to adopt malware analysis to avoid such incidents in future. The major purpose of Malware Attribution is to deeply examine the malware samples, study its origin, tactics, functioning and different procedures.

Various Industries are inclining to deploy combination of techniques to protect against cyber threats while each strategy has its own advantage and disadvantages for instance the security system may use signature based detection to quickly identify known threats while also incorporating machine learning behavioral analysis to identify new and evolving threats this helps organization to detect and prevent cyber-attacks more effectively similarly another common approach is the use of indicators of compromise IOC’s which are specific behaviors or are defects associated with known malware or threat actors file extension block list can be used to prevent the execution of certain file types that are commonly used to distribute malware.

2.4 Literature Review

Numerous research and solutions have been proposed to improve the detection and analysis techniques; however, these methodologies have some shortcomings i.e., attackers have adopted new trend of IDs, similarly many evaluations are only based on limited dataset, or many approaches cannot identify a suitable number of features to train a classifier or compatibility in scaling large number of samples. This literature review aims to gain an understanding relevant to our project, Malware Attribution.

2.4.1 Malware Analysis Attribution Using Generic Information: Maggi

The article discusses a system that was developed to analyze malware using concepts from science and linguistics. To group malware into families, the system makes use of several approaches, including static, dynamic, behavioral, and functional analysis [2]. With the use of this research, defenses may be created quickly, future attacks can be predicted, and theories regarding the malware's origin can be put forth. The scope of our project is non generic and completely based on TTPs, without the application of reverse engineering. The system then compares behavioral profile of the malware with the profiles of non-malware families to determine attribution. The authors demonstrate the effectiveness of this approach through experimentation and evaluation.

Despite its promise the Maggi approach has also several limitations

- It may not be effective against highly polymorphic malware, which can change their behaviors to evade detection.
- Another issue is that the approach relies on metadata which can't be manipulated by attackers to fool the system.
- The system may produce false positives if the behavioral profile of a system of malware is like multiple known malware families.

2.4.2 Kaspersky Threat Attribution Engine

Kaspersky Threat Attribution Engine [3] is a malware analysis tool that offers insight into the source and potential authors of the any malware. It helps prioritize high-risk attacks for prompt defensive measures by swiftly attributing a threat to a known Advanced Persistent Threat (APT). The system uses a combination of machine learning algorithms an expert analysis to identify the origins and motives of malicious actors [4]. It analyzes multiple factors, including the code and infrastructure used by the attacker, as well as victimology of the targets to generate hypothesis about the identity of attacker however like any attribution system Kaspersky threat attribution engine has its limitations.

- Firms and governments are concerned that this software might serve as a listening or monitoring agent for a foreign power, endangering the security of their data.
- Attribution can be complicated by use of false flags or use of infrastructure owned by a third party.
- Difficulty of obtaining accurate and complete data about cyber-attacks which make it challenging to identify the true source of an attack.

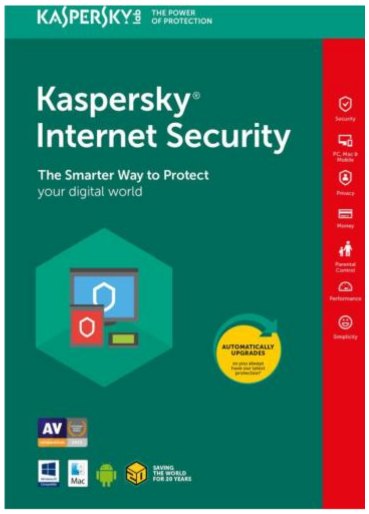

KASPERSKY THE POWER OF PROTECTION
Kaspersky Internet Security
The Smarter Way to Protect your digital world
AUTOMATICALLY UPGRADES

Figure 2 Kaspersky Internet Security

2.4.3 Advanced persistent threat analysis using Splunk.

Advanced persistent threat analysis explores the use of Splunk [5], A popular security information and event management (SIEM) tool [19], for advanced persistent threat analysis. The system outlines various data sources and collection methods that can be used to detect and investigate APT's, including log files, network traffic and endpoint data. The author also proposes A methodology for using Splunk to analyze APT's, which includes developing and tuning detection [8] rules, performing correlation an analysis conducting threat intelligence research.
However, like any APT analysis system there are some drawbacks to using Splunk.

- One major challenge is the need for skilled analysts with expertise in both cybersecurity and Splunk. The system generates a large volume of data which requires skilled analysts to process and interpret.
- Additionally, Splunk is a commercial tool and can be costly to implement and maintain, which can limit its accessibility for small and medium sized organizations.
- Another drawback is the need for continuous updates and maintenance of detection rules, as APT's are constantly evolving and adapting to new security measures.

Figure 3 Splunk

2.5 Research Based Results

2.5.1 Malware Classification with BERT

This research paper explores the use of Bidirectional Encoder Representation from Transformers [10] language model for malware classification. The paper explains that BERT can be used to learn representation of malware samples based on their textual description which can then be used to classify malwares into different families the paper proposes A methodology for fine tuning BERT for malware classification [18] and evaluates its performance on data set of over 2000 malware samples.

Following are the security flaws while studying the classification with BERT [1,18].

- One of the drawbacks of this approach is that it requires a large amount of labeled data for training which can be difficult to obtain for certain malware families.
- Another drawback is that the approach relies on textual description of malware samples which may not always be available or may not contain enough information to accurately classify the malware in addition the performance of the approach may be affected by the quality of descriptions and variability in the language [6] used to describe the malware.
- Another potential drawback is that the approach may be vulnerable to adversarial attacks where an attacker modifies the malware sample in a way that changes its classification while preserving its functionality this can be particularly problematic if the approach is used in a security system that relies on accurate malware classification to detect and prevent attacks.

2.5.2 Clustering analysis for malware behavior detection using registry data.

The research paper "Clustering analysis for malware behavior detection using registry data" [16] proposes an approach for detecting malware by analyzing the behavior of the system through registry data. The method involves clustering the registry keys and values, and then identifying abnormal pattern that may indicate the presence of malware the study found that this approach is effective in detecting malware and can be used as complementary method to traditional signature-based detection. The method first collects registry data from a target system and performs preprocessing to remove irrelevant data [20]. Then the data is clustered using K-means algorithm to identify patterns of behavior that are indicative of malware.

- Firstly, it heavily relies on the accuracy of the clustering algorithm, which can be influenced by various factors such as the choice of distance measure and number of clusters.
- The method requires access to the systems registry data which may not always be feasible in certain environments or scenarios.
- The approach relies on the assumption that malware behavior will always be reflected in registry data, which may not always be the case.
- The method is limited to detecting malware that exhibits registry related behavior and may not detect other types of malwares.
- The clustering approaches vulnerable to false positives, which may result in legitimate software being flagged as malware.

2.6 Conjecture

Existing solutions suffer from a range of issues and complications that undermine their reliability and safety. Users are seeking a more effective and innovative solution that can provide them with complete anonymity and security, align their concerns with cyber threats and vulnerabilities. As cybercrime continues to grow it is critical that users have access to a solution that can deliver a high degree of protection, safeguarding their online activities and personal information therefore the need for a digital robust and reliable solution has become increasingly urgent in today's digital age.

2.7 Chapter Summary

The literature review chapter offers a thorough analysis of previous scholarly research concerning the Maggi research paper, Kaspersky Threat Attribution System, and advanced persistent threat analysis utilizing Splunk. It examines significant themes, evaluates methodologies, and identifies gaps in research.

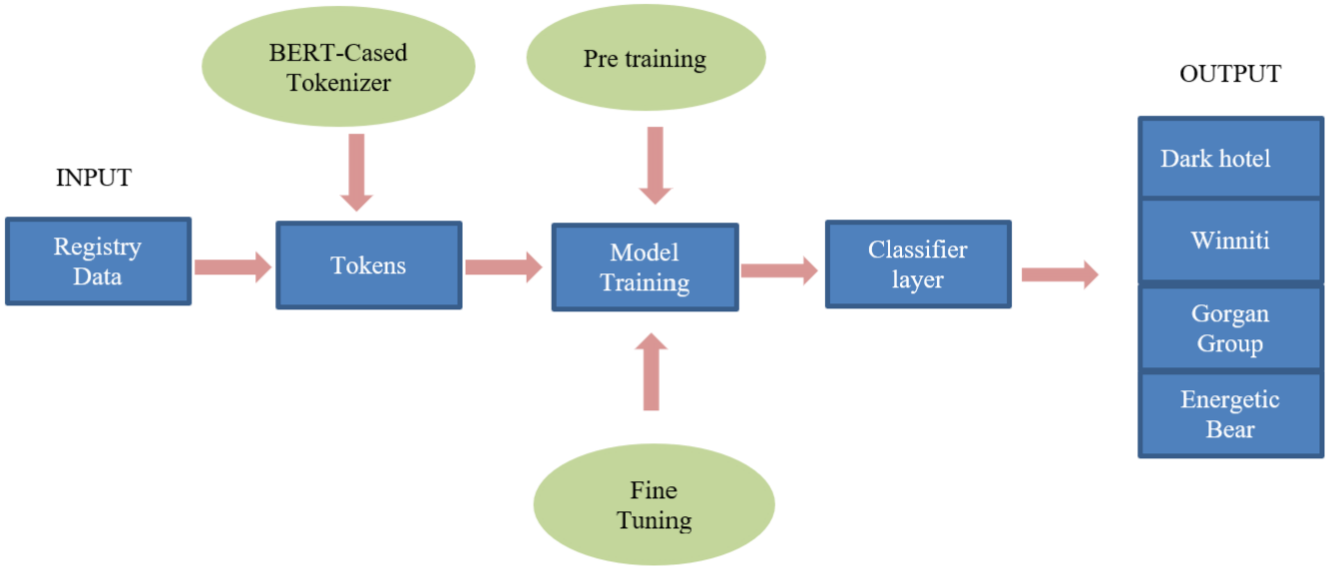
CHAPTER 3: PROJECT DESIGN AND METHODOLOGY

Chapter 3: Project Design and Methodology

3.1 Overview

This chapter covers the project's technical requirements as well as the specifics of the project's key steps as it progresses.

3.2. Block Diagram



3.3. Methodology

For our research on malware attribution, we have chosen to implement two different methodologies to collect relevant data for analysis.

- The first methodology involves the collection of windows log events by Sysmon [15], which will allow us to gather a wide range of system level events and activities.
- The second methodology involves the collection of registry data from known malware samples, which will provide us with detailed information about the specific behavior and actions of the malware.

To process and analyze the collected data we will utilize BERT, a powerful machine learning model that is specifically designed for natural language processing tasks. They collected data into bird, we will be able to extract relevant features and perform clustering analysis to identify similar behaviors and patterns among the collected data, this approach shall further be discussed in detail in Chapter 4.

Ultimately our goal is to use these methodologies and tools to accurately attribute malware to specific threat actors or groups providing valuable insights into the tactics, techniques, and procedures used by these malicious actors.

Prior to implementing any approach, it is necessary to establish an appropriate virtual environment and download malware samples as a preliminary step. This entails creating a self-contained and isolated environment that enables the installation and configuration of necessary software packages and tools without interfering with the underlying operating system.

3.3.1 Setting Up a Virtual Environment

A virtual machine that provides an isolated and secure environment (sandbox) to execute malware samples. It runs in a separate system therefore protecting the host and network infrastructure. For this project, we simulated the host operating system with VMware [13] workstation pro (version 17.2), the snapshot feature of VMware is the eminent advantage . This feature enables you to record the state of a virtual machine at a particular point in time. It restores it to a previous state in case something goes wrong, creates a delta file when you take a snapshot of a virtual machine. Installing Windows operating system is the next step after the program has been installed.

3.4 Log Events Collection

3.4.1 System Monitor (Sysmon)

Windows operating system has an inbuilt service – Event Viewer, that keeps a track of every activity performed. Sysmon provides an added advantage to monitor Windows event logs and activities: network connection, process creation and termination, process ID etc. These assist in malware analysis [11] by capturing malicious events.

3.4.2 Sample Collection

Different malware samples have been collected from various sources on the internet which offer free access for research and evaluation purpose i.e. GitHub, malware bazaar [22] etc.
https://github.com/mikesiko/PracticalMalwareAnalysis-Labs
https://github.com/fabrimagic72/malware-samples
Windows event logs, that capture a variety of system events such as process launches, network activity, and security-related events, can be a crucial source of data for malware investigation. Event logs can be collected and analyzed to find abnormal activity and possible malware-related indicators of compromise (IOCs).

Sysmon logs can be generated verbose. The huge amount of data generated requires careful filtering and analysis. In order to ensure data integrity, it's necessary to that the Sysmon logs are stored securely and obscured from unauthorized entry.
For our project, we have collected a dataset of total 120 different samples from publicly available malware repositories. For the purpose of our research, we will only be focusing on 5 distinct categories of malware as it's important to remember that malware evolves continually and new variants are constantly appearing; trojans, keyloggers, spyware, rootkits and worms.

3.4.3 Automated Log Extraction

The process of gathering and examining event logs automatically, produced by the operating system and different apps running on a Windows PC is recognized as automated log extraction in Windows. Event logs can be used to troubleshoot problems and diagnose issues since they provide details about system events like errors, warnings, and informational messages.

Automated log extraction often requires collecting event logs from several sources, like Windows Event Viewer or third-party log sources, and then examining them for patterns or anomalies using specialized software tools. This procedure can assist IT specialists in locating flaws or potential security vulnerabilities before they become serious difficulties.

For our project, Python script is used to automate the process of gathering event logs produced by the operating system.

3.4.3.1 Preferred Installer Program (PIP)

The Python programming language has a tool called pip for installing packages. It is a command-line utility used to manage Python packages and their related dependencies, and additionally to install and upgrade Python packages. Pip on the Windows Command Prompt, uses command, "pip", including the package name to be installed or the location of a requirements file. Other helpful capabilities offered by Pip include the capacity to uninstall programs, list installed packages, and look for packages that are open for download. Pip is an important tool for Python programming since it makes it easier to manage and install dependencies.

We have installed two libraries through pip by using the following commands.

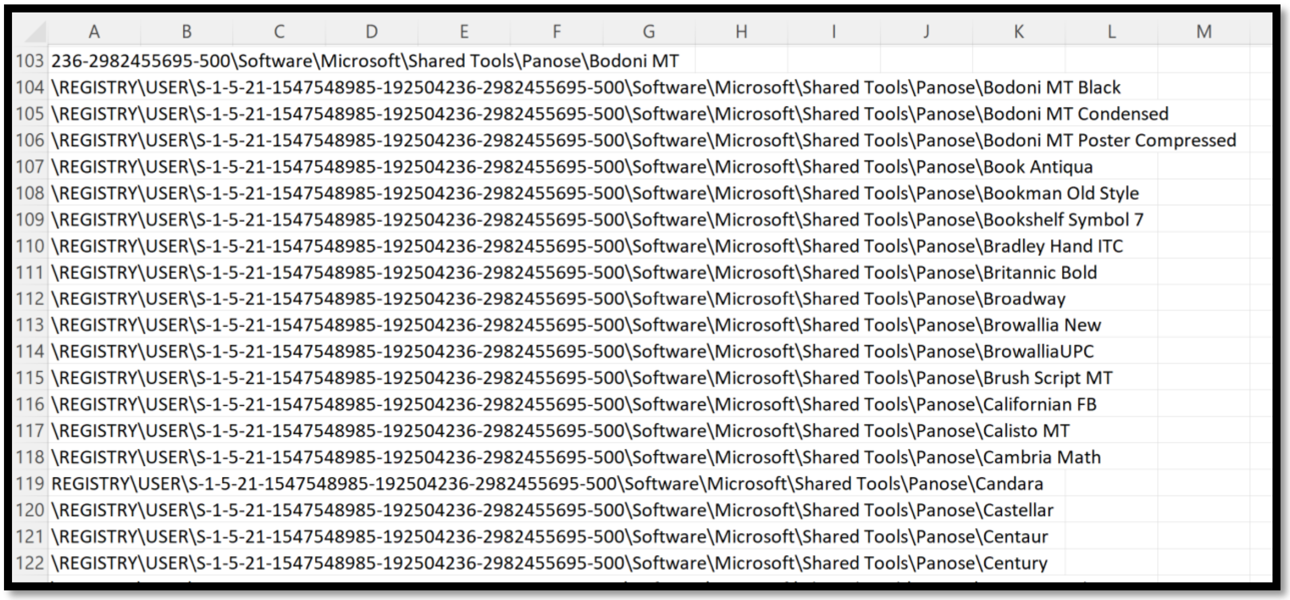
- Pip install win32evtlog.
- Pip install xmltodict

A Python package called win32evt gives users access to the Windows Event Log. You may employ it to obtain events from the log, write events to the log, and manage the log itself (like adding new event sources or deleting the log).

Python's xmltodict package offers a simple method for working with XML data. It enables you to transform data from XML to a Python dictionary and the other way around. The markup language XML (eXtensible Markup Language) is frequently used for data exchange between various systems. To specify data pieces and their connections, it uses tags. Although XML is a frequently used data format, working with it in Python can be challenging because the default library does not offer a simple method to parse XML data.

3.5 Registry Keyset Collection

Registry keys of malware samples [21] were obtained from various sources such as Virus total [17] and manual analysis, and were subsequently compiled into a structured format, in the CSV format. This process involved extracting relevant information from the registry keys, organizing and formatting the data, and then consolidating them into a single file, organizing the registry key data into a suitable format for input into Bert, which could then potentially reveal significant features or patterns in the data that could aid in understanding the characteristics and behavior of the malware.



```
103 236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Bodoni MT
104 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Bodoni MT Black
105 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Bodoni MT Condensed
106 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Bodoni MT Poster Compressed
107 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Book Antiqua
108 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Bookman Old Style
109 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Bookshelf Symbol 7
110 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Bradley Hand ITC
111 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Britannic Bold
112 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Broadway
113 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Browallia New
114 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\BrowalliaUPC
115 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Brush Script MT
116 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Californian FB
117 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Calisto MT
118 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Cambria Math
119 REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Candara
120 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Castellar
121 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Centaur
122 \REGISTRY\USER\S-1-5-21-1547548985-192504236-2982455695-500\Software\Microsoft\Shared Tools\Panose\Century
```

Figure 4 Registry Key Dataset

Four families of malware, namely Winnti, DarkHotel, Gorgon, and Energetic Bear were studies for this project.

1. Winnti: Winnti is a type of advanced persistent threat (APT) malware that mainly targets organizations in the gaming, healthcare, and technology industries. Some of the key features of Winnti include:

   - Using multiple components to avoid detection and persist on the system.
   - Being capable of downloading and executing additional malicious payloads.

- Using custom encryption to protect its communication channels.

2. DarkHotel: DarkHotel is a type of APT malware that primarily targets high-profile individuals such as executives and government officials who are staying in luxury hotels. Some of the key features of DarkHotel include:

- Being capable of exploiting zero-day vulnerabilities to infect systems.
- Using phishing emails and social engineering techniques to lure victims into downloading the malware.
- Monitoring network traffic to steal sensitive information such as login credentials and intellectual property.

3. Gorgon: Gorgon is a type of malware that is primarily used for cyber espionage and data theft. Some of the key features of Gorgon include:

- Being capable of using multiple infection vectors such as email attachments and malicious websites
- Using a modular architecture to execute different functions such as keylogging and data exfiltration.
- Using custom encryption to protect its communication channels.

4. Energetic Bear: Energetic Bear is a type of malware that is primarily used for cyber espionage and sabotage against organizations in the energy and industrial sectors. Some of the key features of Energetic Bear include:

- Being capable of infecting both Windows and Linux-based systems
- Using social engineering techniques such as spear-phishing to infect targeted systems.
- Using custom protocols and encryption to avoid detection and protect its communication channels.

3.6 Machine Learning Algorithm

In this project, we plan to employ the BERT algorithm, a cutting-edge natural language processing model, as a key tool in our analysis. Specifically, we intend to use BERT to analyse registry keys extracted from malware samples, with the goal of identifying potential patterns or relationships between the keys that could shed light on the origin, authorship, or behaviour of the malware.

To achieve this goal, we will describe in detail in Chapter 4 of our thesis how we plan to leverage the power of BERT to analyse the registry key data. This will include a discussion of the specific techniques we will use to pre-process the data and prepare it for input into the BERT model, as well as the criteria we will use to evaluate the results of our analysis.

By incorporating BERT into our malware attribution project, we aim to improve the accuracy and efficiency of our analysis, and to gain new insights into the behavior and characteristics of the malware samples we are studying. Ultimately, our goal is to contribute to the field of cybersecurity by advancing our understanding of how malware operates and who may be responsible for it.

3.7 Chapter Summary

The research methodology chapter outlines the study's data gathering procedures, analysis methods, and research design. It describes the methodical process used to compile and examine data in order to guarantee the validity and reliability of the study. A detailed plan for carrying out the study is provided in the chapter.

CHAPTER 4: Machine Learning in Threat Attribution

Chapter 4: Machine Learning in Threat Attribution

4.1 Chapter Overview

This chapter presents the methodology employed for attributing malware to their respective threat groups using machine learning techniques. Specifically, we utilize Windows event registries and apply deep learning algorithms, including BERT, to generate embeddings and classify malware into four major threat groups: Dark Hotel, Winniti, Gorgon Group, and Energetic Bear. This chapter outlines the steps involved in preprocessing the data, training the machine learning model, and evaluating its performance.

4.2 Machine Learning Algorithm

4.2.1 Natural Language Processing

Word embeddings, a type of rich information extracted from sentences of a language using NLP (natural language processing) techniques, can be used to fill in the gaps in phrases or determine the meaning of a sentence. The relationship between each word in a phrase and every other word is extracted by NLP models. In the input dataset it is given, the model combines together keywords with similar meanings and maps them onto a space with greater dimensions. NLP models can perform a variety of categorization and prediction tasks with the aid of this knowledge.

To create embeddings for malware samples, NLP models can be applied to the field of malware recognition. The malware samples from the same group would share similar characteristics. Classifiers might make use of this data to put malware samples from the same family in a single category.

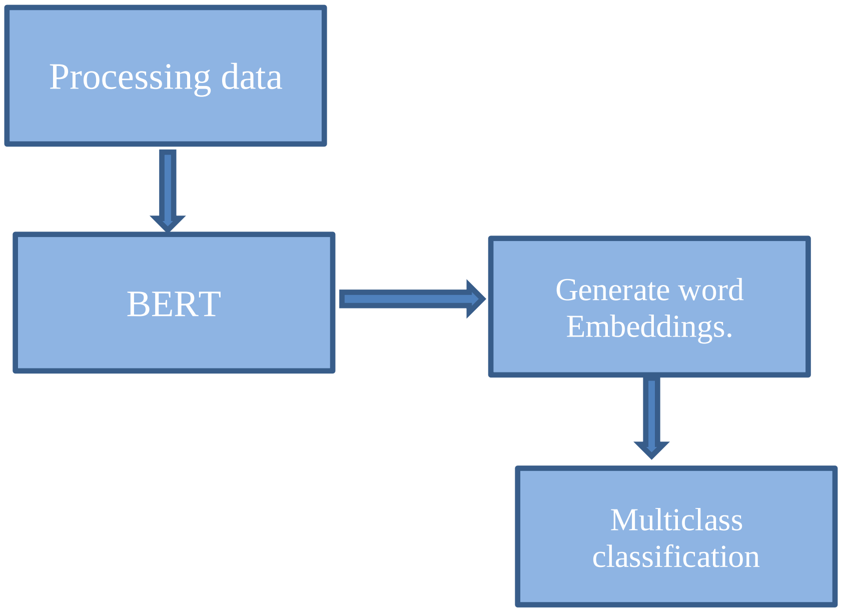


Figure 5 Bert Algorithm

4.2.2 Word Embedding

Word embeddings [7], which capture the significance and implications of the language they are employed in, are numerical representations of words or sentences.

4.2.3 BERT

BERT is a transformer-based NLP model that handles language-based tasks like sentiment classification, masked word prediction, and other classification tasks [1]. The architecture is nothing more than a collection of skilled Transformer Encoders. By also considering the context in which a word was used, a technique known as contextualized word embeddings, BERT is able to provide the word embedding for a specific word. This is how BERT works:

1. Data Preprocessing: Before the raw data is input into the BERT model, it must first undergo preprocessing. To do this, the text must be tokenized, put into numerical format, and divided into train and test sets.
2. Fine-tuning BERT: BERT is a model that has already been trained using a lot of text data. We must adjust it for our particular goal in order to use it for malware investigation. This includes employing methods like transfer learning to train the model on our dataset.

3. The BERT model can be used to extract characteristics from the samples of malicious and benign software in our dataset once it has been tweaked. The samples are placed through the BERT model, and one of the layers' outputs are extracted.
4. Following the features' extraction from the samples, we may utilize them to train a classifier. This entails classifying the samples as malicious or benign using supervised learning techniques like decision trees, logistic regression, or support vector machines.
5. Metrics like accuracy, precision, recall, and F1 score can be used to assess the classifier's performance after it has been trained. Cross-validation is one method we can use to make sure our findings are reliable.
6. At last, we can use the learned classifier to identify and stop malware attacks in a network security system or antivirus programme.

4.3 Data Preprocessing

Data pre-processing is a crucial step in machine learning and data analysis, as it involves transforming raw data into a suitable format for further analysis. This section discusses the initial steps involved in pre-processing a dataset containing Windows event log data related to malware instances. The goal is to prepare the data for subsequent analysis, ensuring accurate and reliable results in malware detection.

1. Importing the necessary libraries: To begin the data pre-processing, several libraries are imported. These libraries include:
    - TensorFlow: A popular open-source machine learning framework.
    - TensorFlow Hub: A library that provides pre-trained models and reusable components.
    - TensorFlow Text: An extension of TensorFlow specifically designed for text-based tasks.
    - Pandas: A powerful data manipulation and analysis library in Python.
    - NumPy: A fundamental package for scientific computing in Python.
2. Loading the dataset: The dataset is loaded from a CSV (Comma Separated Values) file, which contains Windows event log data associated with malware instances. This dataset

likely consists of various features or columns that describe the characteristics of each event log entry, such as event ID, timestamp, source, and message.

3. Pre-processing to remove newline characters: In some cases, the dataset might contain newline characters ("\n") within the text data, which can interfere with subsequent analysis or modelling. Therefore, a pre-processing step is performed to remove these newline characters from the dataset. This can be done using appropriate string manipulation techniques or regular expressions, ensuring that the data remains intact and meaningful.
4. Examining the distribution of malware labels: It is essential to understand the distribution of the malware labels within the dataset. This step helps ensure that the dataset is balanced, meaning that each label class has a sufficient number of instances for accurate model training and evaluation. By analysing the distribution, it becomes possible to identify any class imbalances or biases that might affect the overall performance of the malware detection system.

4.4 Data Splitting

Dataset splitting allows for the evaluation and validation of models on unseen data. We will discuss the process of splitting a dataset containing Windows event log data related to malware instances into training and testing sets. The dataset is divided using the train_test_split function from the scikit-learn library. The aim is to create a balanced and representative split, ensuring accurate model assessment for malware detection.

Our dataset is divided into two subsets: the training set and the testing set. This separation is necessary to evaluate the performance of the model on unseen data, simulating real-world scenarios.

1. Importing the necessary libraries: To perform the dataset splitting, we utilize the train_test_split function from the scikit-learn library. Scikit-learn is a widely used machine learning library in Python that provides various tools for data manipulation, pre-processing, and modelling.

2. Splitting the dataset: The train_test_split function takes the dataset as input and splits it into two subsets based on a specified ratio. In this case, the dataset is divided into 80% for training and 20% for testing. This ratio can be adjusted depending on the specific requirements of the analysis.

3. Stratification for label preservation: Preserving the distribution of malware labels is essential to ensure that both the training and testing sets have a representative distribution of classes. It helps prevent biases and ensures that the model learns to generalize well across different malware categories.

4. Evaluation on unseen data: By separating the dataset into training and testing sets, we can train the model on the training set and evaluate its performance on the testing set. This evaluation provides insights into the model's ability to generalize to new, unseen instances and helps assess its effectiveness in detecting malware.

4.5 Embedding generation with BERT

Embedding generation plays a crucial role in natural language processing tasks, including malware detection. This will focus on the process of generating embeddings using the BERT (Bidirectional Encoder Representation Transformers) pre-trained model provided by TensorFlow Hub. The BERT model enables the creation of semantic representations of text data, allowing machine learning models to understand the patterns and attributes of malware instances.

1. Utilizing the BERT Pre-trained Model: BERT is a powerful language model that has been pre-trained on a massive amount of text data. By leveraging the pre-trained BERT model provided by TensorFlow Hub, we can benefit from its learned representations and apply them to specific tasks, such as malware detection.

2. Creating an Input Layer for Text: To generate embeddings, we need to provide the text data as input to the BERT model. Therefore, an input layer is created specifically for text, allowing us to feed the raw text data into the BERT model.

3. Passing through the BERT Encoder: After pre-processing, the text data is passed through the BERT encoder. The BERT encoder consists of multiple transformer layers, which capture the contextual information and relationships between words in the text. The encoder performs deep bidirectional learning, considering the context both before and after each word. As a result, the BERT encoder generates high-quality contextualized word embeddings.

4. Generating Embeddings: The output of the BERT encoder is a sequence of embeddings representing each word in the text. These embeddings capture the semantic representation of the text, encoding important information about the meaning and context of the words. By leveraging these embeddings, the machine learning model can better understand the underlying patterns and attributes of the malware instances, leading to improved detection performance.

4.6 Model Architecture

In our pursuit of accurately classifying malware based on threat groups, we have adopted a deep learning approach. This methodology incorporates the utilization of BERT (Bidirectional Encoder Representations from Transformers) embeddings as the model's input. BERT embeddings are pre-trained language representations that capture the contextual information of words and sentences, making them highly effective in comprehending the distinctive characteristics of malware.

To preprocess the textual data, we employ a dedicated preprocessor model called Bert_en_uncased_preprocess. This preprocessor is specifically designed to complement the BERT architecture, which has demonstrated exceptional performance across various natural language processing tasks. By employing this preprocessor, we effectively tokenize, normalize, and encode the input text, facilitating the extraction of significant features for subsequent analysis.

The encoder model is constructed using the small_bert/bert_en_uncased_L-2_H-128_A-2/2 architecture. This architecture comprises two hidden layers, each having a hidden size of 128, as well as two attention heads. This selection of a compact yet powerful encoder model allows us to capture the intricate relationships and dependencies present within the textual data, enabling the generation of precise representations of the input data.

Throughout the training phase, we leverage the Adam optimizer, a widely used optimization algorithm in the field of deep learning. The Adam optimizer dynamically adjusts the learning rate for each parameter, leading to effective model optimization. In our case, we set the learning rate to 0.1, allowing the model to make substantial updates during training. Additionally, we incorporate a decay mechanism with a decay rate of 0.96 and decay steps of 2. This approach facilitates model convergence by gradually reducing the learning rate over time, ensuring a balance between exploration and exploitation.

To ensure effective model training, we conduct multiple epochs, specifically 100 epochs. This extended training period provides the model with ample exposure to the training data, enabling it to learn complex patterns and generalize well to unseen instances. During training, we employ the sparse categorical cross-entropy loss function, which is widely used in multi-class classification tasks such as our malware classification problem. This loss function effectively measures the discrepancy between the predicted labels and the true labels of the malware samples, guiding the model towards optimal performance.

By effectively combining the BERT embeddings, the chosen encoder model architecture, the Adam optimizer with a learning rate schedule, and the sparse categorical cross-entropy loss function, we have developed a robust and accurate model for classifying malware according to threat groups. The BERT embeddings capture vital contextual information and provide a comprehensive representation of the malware samples. The dense output layer, with its corresponding number of output classes and threat categories, allows for precise predictions for each threat group. The utilization of the Adam optimizer with a learning rate schedule ensures effective parameter adjustments during training, facilitating optimal learning and adaptation. Ultimately, this comprehensive approach equips us with a powerful tool to classify malware accurately and address associated security concerns.

Here is the list of the model parametres:

Preprocessor Model =Bert_en_uncased_preprocess

Encoder Model =small_bert/bert_en_uncased_L-2_H-128_A-2/2

Hidden Layers=2

Hidden Size=128

Attention Heads=2

Epochs=100

Optimizer=Adam

Learning rate=0.1

Decay_steps=2

Decay_rate=0.96

Loss= Sparse Categorical CrossEntropy

4.7 Model Training and Evaluation

We use our deep learning model to implement the task of classifying malware into their respective threat groups and conduct a thorough training and evaluation process. Our labelled dataset is first split into training and testing sets to make sure the data distribution reflects the real-world circumstances we want to address.

The fit function is used to feed the training set into the model to start the training process. Utilizing methods like backpropagation and gradient descent, this function iteratively optimizes the model's parameters in an effort to reduce loss and boost the model's ability to predict the future. To track the model's development and avoid overfitting, we add the validation data, which serves as a distinct subset of the training set, throughout training. We can examine the model's capacity to generalize to new cases and, if necessary, make improvements by periodically reviewing the model's performance on the validation data.

The number of times the model iterates over the complete training dataset depends on how long we train the model for, or the number of epochs. In this instance, we train the model over 100 epochs, enabling it to discover intricate linkages and patterns in the data. This prolonged training period enhances the model's robustness and accuracy.

The model's performance is assessed when training is finished using the testing set, which consists of labelled instances that the model has never seen before. We evaluate the generalization capacity of the model and evaluate its performance in real-world scenarios by measuring the model's accuracy on an independent dataset. This evaluation allows us to evaluate the model's performance and determine how well it can categories malware according to related threat groups.

Additionally, our trained model is an effective tool for identifying the threat group of previously unknown malware infections. We can generate predictions about the corresponding threat groups by feeding brand-new, previously undiscovered malware samples into the trained algorithm. Due to our capacity to swiftly and effectively identify potential hazards linked to malware that hasn't been seen before, proactive and efficient cybersecurity actions are made possible.

The model is trained on the labelled training set, performance is monitored using validation data, and accuracy is assessed using an independent testing set in our comprehensive approach. We can confidently forecast the threat category of undiscovered malware instances using our trained model, enabling us to proactively strengthen security precautions and counteract prospective attacks.

4.8 Chapter Summary

This chapter has presented a comprehensive methodology for attributing malware to their respective threat groups using machine learning techniques. By leveraging Windows event registries and employing BERT embeddings, we have demonstrated the effectiveness of deep learning algorithms in classifying malware. The results obtained from this methodology contribute to enhancing our understanding of malware attribution and provide valuable insights for cybersecurity professionals in combating evolving threats.

Overall, this chapter highlights the significance of machine learning in the field of malware analysis and emphasizes the potential for further advancements in the application of deep learning models for cybersecurity purposes.

CHAPTER 5: RESULTS

Chapter 5: Results

5.1 Overview

The research study's findings are presented in the results chapter along with data analysis, statistical testing, and result interpretation. This chapter adds to the research goals and body of knowledge in the topic.

Over the years, data scientists and researchers have created a wide variety of model interpretability techniques. Listed below are the results of the BERT algorithm used in this project.

5.2 Model Accuracy

A particular sort of visualization that demonstrates how the accuracy of a machine learning model alters over time or with various parameters is a model accuracy graph. One of the most used metrics for assessing a machine learning model's performance is accuracy. Out of all outcomes, it calculates the proportion of accurately predicted outcomes. The following formula is used to determine accuracy:

Accuracy = (Number of Correct Predictions) / (Total Predictions)

The graph below demonstrates that accuracy of 0.85, or 85%, is attained.

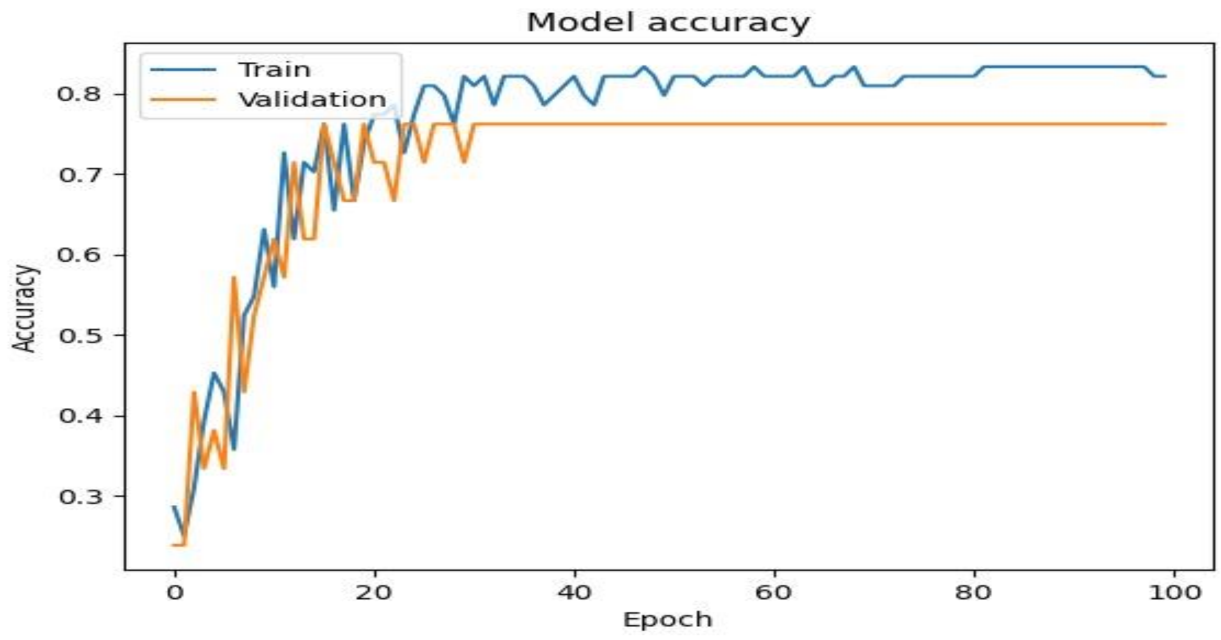


Figure 6 Model Accuracy Curve

5.3 Model Loss

A sort of visualization called a "model loss graph" demonstrates how a machine learning model's loss varies over time or in response to varied parameter values. Loss is a metric for gauging a model's accuracy in predicting the right result from a given input. It calculates the discrepancy between production as projected and output as actually produced.

Lower train and validation loss levels can be seen in the graph below. But the latter outperforms the former by only a slight difference. When this occurs, the model may be overfit and have problems extending to new data. Cross-validation, diversified data collection, model complexity reduction, and hyperparameter modification can all be utilized to resolve this issue. With the aid of these methods, model performance can be enhanced by lowering overfitting, enhancing generalization, and closing the gap between training and validation loss.

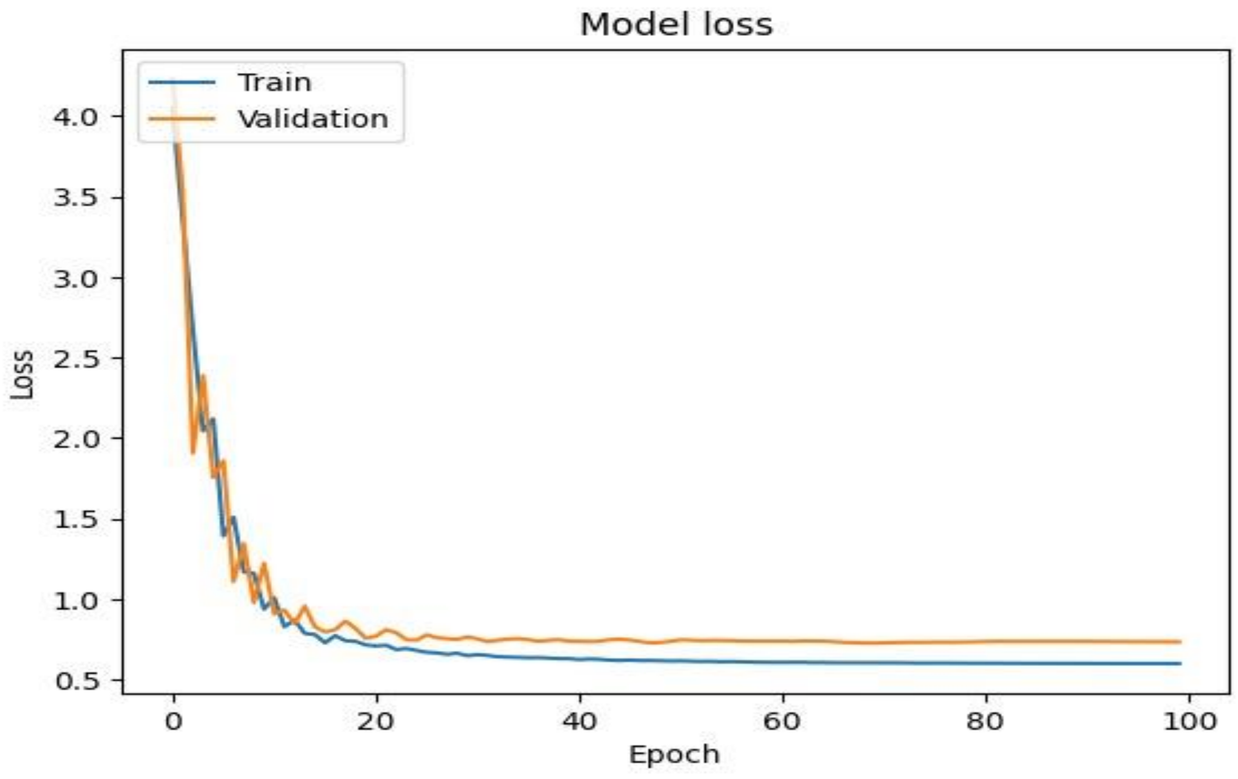


Figure 7 Model Loss Curve

5.4 Confusion Matrix

The confusion matrix is a table that contrasts the actual class labels in the dataset with the anticipated class labels of a classification model. Information on true positives, true negatives, false positives, and false negatives are provided. Accuracy, precision, recall, specificity, and the F1 score are among the evaluation metrics that are computed using these variables. The confusion matrix aids in highlighting flaws in the model, locating inaccuracies, and suggesting improvements. It helps with modifying the classification threshold, resolving class imbalances, and identifying misclassification tendencies. The confusion matrix, in general, is a useful tool for assessing and improving classification models since it offers a thorough understanding of their performance and enables reasoned decision-making for optimization.

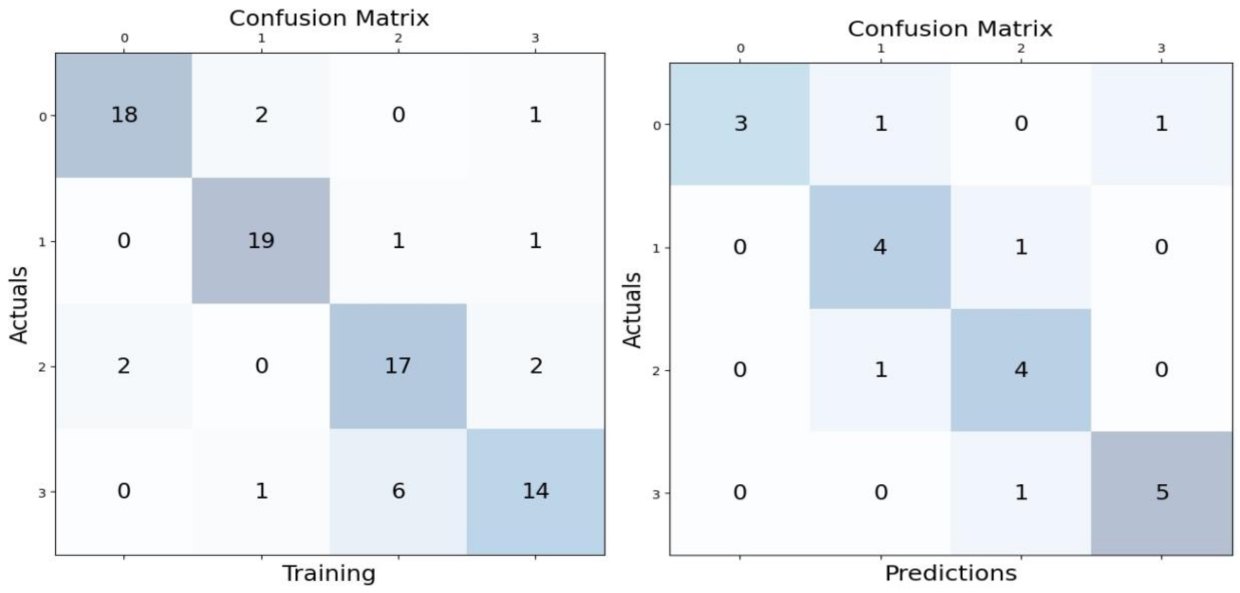


Figure 8 Confusion Matrix for Training and Validation Process

5.5 Chapter Summary

In the results chapter, the classification model is evaluated using the model loss, model accuracy, and confusion matrix metrics. These metrics, which provide information on the model's functionality, accuracy, and mistake types, serve as a reference for the model's interpretation and potential improvements.

CHAPTER 6: CONCLUSION AND ENHANCEMENTS

Chapter 6: Conclusion and Enhancements

6.1 Overview

The research findings are outlined in the conclusion and future works chapter, with an emphasis on the most important contributions and insights. It addresses the research objectives, evaluates the study's consequences, and proposes prospective directions for additional study. It also provides recommendations and guidelines for further research.

6.2 Conclusion

In this thesis, we discussed a malware attribution system that analyses malicious events and identify malwares on basis of it. Our proposed system has an advantage over other traditional systems since attackers are using new ID trends, and many evaluations are also based on small datasets or fail to find enough features to scale enough samples or a sufficient number of features to train a classifier. Techniques used in our proposed system; sample collection, log extraction, and Machine Learning Natural Language Processing (NLP) algorithms, that by default generates word embeddings for the dataset, and classifies the malwares; are briefly explained including their working and importance, The purpose of increasing productivity and overcoming problems in existing solutions is being achieved by using the modern techniques. Additionally, the objectives creation of malware database.

The degree of accuracy of training data, the complexity of the malware samples, and the particular task being carried out (for example, binary classification or multi-class classification) all affect how accurate BERT is in analyzing malware. The algorithm shows a 75% accuracy. It's crucial to keep in mind that 75% accuracy could not be enough for some applications, like protecting key infrastructure or maintaining national security. We have also used registry files as the dataset, that analyzes and classifies malwares into the threat actor group. The process of locating the individuals or entities behind a cyberattack is known as threat attribution. Analysis of the attackers' artefacts, such as registry files, is one method for threat attribution. Windows operating systems keep configuration settings and other system data in a single database called the registry. Registry file analysis can provide crucial information about the attack, including the infrastructure used by the attackers as well as the tools and methods they employed. The algorithm shows an 85% accuracy.

6.3 Enhancements and Future Work

Future milestones that need to be achieved to commercialize this project are the following.

1. When malware infects a machine or interacts with a command-and-control server, it frequently displays particular behaviour patterns. By improving our sequential data, such as network traffic logs or system call traces, can assist find these patterns and comprehend how the malware behaves.
2. Initially, the machine learning algorithm – BERT, was trained on a dataset of 500 samples. In future, the enhancement of the dataset would lead to increased accuracy of classification.
3. In the rapidly expanding field of machine learning, novel strategies and algorithms are continually being created i.e., the most recent generative models include Autoregressive Models, Variational Autoencoders (VAEs), and Generative Adversarial Networks (GANs).
4. Data from network logs, malware samples, and open-source information would be extracted and combined. Our Malware Attribution System will advance through feature selection, automated data preparation, and fusion techniques.

6.4 Chapter Summary

The current section summarizes the entire documentary by highlighting the most crucial aspects of the project, Threat Attribution System, and finishes the thesis by suggesting further work to be done in order to broaden the scope of our research.

References and Work Cited

[1] K. Clark, U. Khandelwal, O. Levy. and C.D. Manning. What does bert look at? An analysis of bert's attention. arXiv preprint arXiv:1906.04341, 2019.

[2] Arun Lakhotia, Suresh Golconda University of Louisiana. Malware analysis and attribution using generic information MAAGI, 2012 7th International Conference on Malicious and Unwanted Software.

[3] kaspersky.com/en/business-security/enterprise/threat-attribution-engine-datasheet.pdf

[4] kaspersky.com/enterprise-security/cyber-attack-attribution-tool

[5] Harikrishnan V N,Gireesh Kumar T TIFAC-CORE in Cyber Security, Advanced persistent threat analysis using SPLUNK, 2017 Amrita School of Engineering, India

[6] Y. Awad, M. Nassar, and H. Safa. Modeling malware as a language. In 2018 IEEE International Conference on Communications, ICC, pages 1–6, 2018.

[7] Chandak et al. A Comparison of Word2Vec, HMM2Vec, and PCA2Vec for Malware Classification. In Malware Analysis Using Artificial Intelligence and Deep Learning (pp. 287-320). Springer, Cham, 2021.

[8] O. Hachinyan. Detection of malicious software on based on multiple equations of API-calls sequences. In 2017 IEEE Conference of Russian Roung Researchers in Electrical and Electronic Engineering, EIConRus, pages 415– 418, 2017.

[9] statista.com/statistics/1315805/cyber-attacks-top-industries-worldwide

[10] huggingface.co/docs/transformers/model_doc/bert

[11] Moser, C. Kruegel, and E. Kirda, "Exploring Multiple Execution Paths for Malware Analysis," 2007.

[12] Bruschi, D., Martignoni, L., Monga, M.: Using code normalization for fighting self-mutating malware. In: Proceedings of International Symposium on Secure Software Engineering, IEEE (March, 2006) http://homes.dico.unimi.it/~monga/listpub.html.

[13] vmware.com/products/workstation-player.html

[14] Nieminen, M. (2012). Information Security Management in Small and Medium-sized Enterprises [PDF]. Retrived from diva-portal.org/smash/get/diva2:535850/FULLTEXT01.pdf

[15] github.com/SwiftOnSecurity/sysmon-config

[16] Rosli, N.A., Yassin, W., Faisal, M.A., & Selamat, S.R. (2017). Clustering Analysis for Malware Behaviour Detection using Registry Data.

[17] virustotal.com/gui/home/upload

[18] Alvares, Joel Lawrence, "Malware Classification with BERT" (2021)

[19] htps://www.comparitech.com/net-admin/siem-tools/

[20] "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding "Authors: Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, Published: 2019.

[21] "Characterization of Malware Behavior using Windows Registry Data", Michael Cohen, Tim Daly, and Seán McSweeney, Published 2012.

[22] https://bazaar.abuse.ch

Malware attribution

ORIGINALITY REPORT

8%	5%	4%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	"ECAI 2020", IOS Press, 2020 Publication	2%
2	Submitted to Eaton Business School Student Paper	1%
3	scholarworks.sjsu.edu Internet Source	1%
4	www.researchgate.net Internet Source	<1%
5	Submitted to Higher Education Commission Pakistan Student Paper	<1%
6	Submitted to Glasgow Caledonian University Student Paper	<1%
7	repository.up.ac.za Internet Source	<1%
8	pureadmin.qub.ac.uk Internet Source	<1%
9	"Malware Analysis Using Artificial Intelligence and Deep Learning", Springer Science and	<1%

	Business Media LLC, 2021 Publication	
10	Submitted to Nottingham Trent University Student Paper	<1%
11	rstudio-pubs-static.s3.amazonaws.com Internet Source	<1%
12	Submitted to Imperial College of Science, Technology and Medicine Student Paper	<1%
13	"Natural Language Processing and Chinese Computing", Springer Science and Business Media LLC, 2019 Publication	<1%
14	Submitted to BITS, Pilani-Dubai Student Paper	<1%
15	Submitted to University of Wales Swansea Student Paper	<1%
16	escholarship.org Internet Source	<1%
17	www.mdpi.com Internet Source	<1%
18	Submitted to National Institute of Business Management Sri Lanka Student Paper	<1%
19	Submitted to University of Stirling Student Paper	<1%

20	codeclimate.com Internet Source	<1 %
21	mdpi-res.com Internet Source	<1 %
22	Submitted to University of Westminster Student Paper	<1 %
23	www.slideshare.net Internet Source	<1 %
24	open.uct.ac.za Internet Source	<1 %
25	research-repository.griffith.edu.au Internet Source	<1 %
26	www.arxiv-vanity.com Internet Source	<1 %
27	David Ramamonjisoa, Hidemaru Ikuma, Riki Murakami. "Filtering Relevant Comments in Social Media Using Deep Learning", 2022 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), 2022 Publication	<1 %
28	Nor Zakiah Gorment, Ali Selamat, Ondrej Krejcar. "Chapter 41 A Recent Research on Malware Detection Using Machine Learning Algorithm: Current Challenges and Future	<1 %

Works", Springer Science and Business Media LLC, 2021
Publication

29	Shashank Kedia, Aditya Mantha, Sneha Gupta, Stephen Guo, Kannan Achan. "Generating Rich Product Descriptions for Conversational E-commerce Systems", Companion Proceedings of the Web Conference 2021, 2021 Publication	<1 %
30	bank-security.medium.com Internet Source	<1 %
31	kth.diva-portal.org Internet Source	<1 %
32	preprints.jmir.org Internet Source	<1 %
33	"Cybersecurity for Artificial Intelligence", Springer Science and Business Media LLC, 2022 Publication	<1 %
34	Communications in Computer and Information Science, 2015. Publication	<1 %
35	Haidar Safa, Mohamed Nassar, Wael Al Rahal Al Orabi. "Benchmarking Convolutional and Recurrent Neural Networks for Malware Classification", 2019 15th International	<1 %

	Wireless Communications & Mobile Computing Conference (IWCMC), 2019 Publication	
36	www.irjmets.com Internet Source	<1%
37	ethesis.nitrkl.ac.in Internet Source	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		