WAZUH PLUS PLUS ADVANCED PERSISTENT THREAT DETECTION USING AI AND INTELLIGENT SIEM

Aleena Khan 20K-0358
Muhammad Abdullah 19K-1495
Siadat Ali 19K-1413

Project Supervisor: Ms. Yusra Kaleem
Project Co-Supervisor: Dr. Jawwad A. Shamsi

FAST School of Computing, National University of Computer and Emerging Sciences, Karachi Campus

May 2024

Abstract

In cybersecurity, detection of Advanced Persistent Threat (APT) attacks poses an important challenge for organizations seeking to improve their defense strategies. Our project proposes an innovative approach, integrating custom-defined rules in Wazuh Plus Plus for APT detection, with advanced machine learning (ML) algorithms tailored specifically for phishing detection.

While custom-defined rules in Wazuh Plus Plus meticulously identify APT related patterns, our hybrid ML approach solely focuses on phishing detection. This approach combines models like Logistic Regression (Random Forest Classifier) with BERT-based, including Bert and RoBERTa, enabling identification and detection of phishing malwares within emails. By analyzing sender details, thematic categories, and link counts, the system effectively discerns phishing attempts with accuracy, evaluated on classification metrics such as accuracy and recall.

Furthermore, we enhance detection efficacy by integrating insights into APT techniques, tactics and indicators of compromise (IOCs). Through finetuning and training on phishing datasets taken from GitHub repositories of Tanusree Sharma and Kostas Koutrou, we provide cybersecurity practitioners with robust tools to proactively detect APT threats, alongside ML inclusive for phishing malwares.

Acknowledgement

The completion of this project could not have been possible without the participation and assistance of the individuals contributing to it. We would like to express our appreciation to our supervisors. We would also like to express our gratitude towards our parents for their kind cooperation and encouragement at every step in our lives, which has ultimately led us to complete this project.

Contents

1 Introduction

Phishing attacks have become a significant threat to cybersecurity, involving deceptive practices to obtain sensitive information from unsuspecting users. These attacks can lead to severe financial and personal data losses, necessitating the development of robust detection mechanisms. Machine learning (ML) techniques have emerged as a powerful tool in combating phishing, offering advanced methods to identify and mitigate such threats effectively.

This report explores the application of machine learning models to detect phishing emails using a dataset labeled "CEAS_08.csv."

2 Literature Review

2.1 Phishing Attacks Detection: A Machine Learning-Based Approach

2.1.1 Introduction

Phishing attacks are a prevalent form of social engineering that targets users' emails to fraudulently acquire sensitive information. Despite numerous proposed anti-phishing techniques, challenges in efficiency and accuracy persist. This paper introduces a machine learning-based approach to enhance phishing detection accuracy.

2.1.2 Overview of Anti-Phishing Techniques

Anti-phishing methods can be broadly categorized into rule-based, whitelist/blacklist, heuristic, and hybrid techniques:

1. **Rule-Based Approaches**: These involve data mining techniques to create models that detect phishing by extracting specific rules from datasets. However, their low accuracy limits their effectiveness.

2. **Whitelist and Blacklist Approaches**: These methods compare URLs to pre-established lists of legitimate or malicious sites. While straightforward, they struggle with new or rapidly changing websites, requiring frequent updates.

3. **Heuristic Techniques**: These techniques use specific features like URLs, IP addresses, and HTML elements to identify phishing attempts. They can be more effective than blacklist methods but still have vulnerabilities.

4. **Hybrid Techniques**: Combining elements from various methods, hybrid approaches can achieve higher accuracy but often at the cost of increased complexity and computational demands.

2.1.3 Methodology

The paper focuses on leveraging machine learning to detect phishing emails. The dataset, derived from over 4,000 phishing emails targeting the University of North Dakota's email service, was used to train and test three machine learning models: Support Vector Machine (SVM), Logistic Regression (LR), and Artificial Neural Network (ANN).

2.1.4 Feature Selection

Ten relevant features were selected for the model, including:

- SSL Certificate: Verifying if the email includes a secure HTTP link.
- Certificate Authority: Checking the credibility of the SSL certificate issuer.
- Blacklist Keywords: Identifying common phishing keywords.
- Redirection URL: Detecting implicit redirections in URLs.
- Hiding Links: Identifying the use of URL shorteners or hidden links.
- Clear IP Address: Spotting links with clear IP addresses instead of domain names.

- Website Traffic: Evaluating the website's Alexa ranking.
- Age of the Webpage: Considering the lifespan of the webpage.
- Sender's Email Address: Checking for inconsistencies in the sender's domain.
- Attached File Extension: Detecting suspicious file types like .exe or .dll.

2.1.5 Classification Techniques

The study compared the performance of three classifiers:

1. **Support Vector Machine (SVM)**: Utilizes a hyperplane to separate data into different classes. A polynomial kernel was employed for non-linear data.
2. **Logistic Regression (LR)**: A binary classification method using the sigmoid function for prediction.
3. **Artificial Neural Network (ANN)**: Implemented due to its ability to capture complex patterns in data.

2.1.6 Results and Discussion

The experimental results demonstrated that the Artificial Neural Network outperformed the other models in terms of detection probability, missed detection, false alarm rate, and overall accuracy. ANN's ability to handle complex patterns made it the most effective in identifying phishing emails.

2.1.7 Conclusion

The study concludes that machine learning, particularly ANN, provides a promising approach for phishing detection. Future work may explore further optimization of features and model parameters to enhance detection rates and reduce computational costs. This review encapsulates the significant findings and methodologies of the paper, highlighting the effectiveness of machine learning in combating phishing attacks.

2.2 Literature Review on "Detection of Phishing Attacks: A Machine Learning Approach

2.2.1 Introduction

Phishing, a prevalent form of identity theft, is a significant threat in the digital world, involving deceptive practices to obtain sensitive information like passwords and credit card details. This literature review delves into the machine learning approach proposed by Ram Basnet, Srinivas Mukkamala, and Andrew H. Sung for detecting phishing attacks, exploring its methodology, dataset, experimental results, and key findings.

2.2.2 Methodology

The authors utilize a machine learning framework to classify emails as phishing or legitimate by incorporating key structural features specific to phishing emails. They employed various machine learning algorithms to analyze a dataset of emails, using features that are indicative of phishing activities. Sixteen distinct features were used, including HTML content, IP-based URLs, domain age, number of domains and subdomains, presence of JavaScript, and form tags. These features were selected based on their relevance to common phishing techniques.

2.2.3 Dataset

The dataset comprised a collection of phishing and legitimate emails. The phishing emails were identified based on known attack patterns and sources, while legitimate emails were sourced from non-phishing email archives. The authors did not specify the exact size of the dataset, but emphasized that it included a diverse set of emails to ensure robustness in the detection model.

2.2.4 Experimental Results

Various machine learning algorithms, including decision trees, support vector machines (SVM), and k-nearest neighbors (KNN), were tested on the dataset. The performance of each algorithm was evaluated based on its accuracy in correctly classifying emails. The results indicated that certain algorithms, particularly decision trees and SVM, performed better in identifying phishing emails, demonstrating high accuracy rates. The use of novel input features significantly contributed to the detection capabilities of the model.

2.2.5 Key Findings

The study highlighted several critical insights:

- **Feature Importance**: The inclusion of structural features like HTML content, IP-based URLs, and domain age proved essential in distinguishing phishing emails from legitimate ones.
- **Algorithm Performance**: Decision trees and SVM outperformed other algorithms, showing that machine learning models could effectively classify phishing emails with high accuracy.
- **Scalability and Adaptability**: The approach demonstrated potential for scalability and adaptability, as it could incorporate new features and data to improve detection over time.

2.2.6 Conclusion

The machine learning approach proposed by Basnet, Mukkamala, and Sung presents a promising solution to the persistent problem of phishing attacks. By leveraging structural features and advanced algorithms, the model achieves significant accuracy in detecting phishing emails. This study lays the groundwork for further research and development in automated phishing attacks, emphasizing the importance of continuous improvement and adaptation to emerging phishing techniques.

2.3 Identification of Phishing Attacks using Machine Learning Algorithm

2.3.1 Introduction

Phishing attacks are a significant and growing threat in the cyber domain, aiming to deceive users into revealing sensitive information such as passwords and credit card numbers. This literature review examines the study by Dinesh P.M., Mukesh M., Navaneethan B., Sabeenian R.S., Paramasivam M.E., and Manjunathan A. on identifying phishing attacks using machine learning algorithms. The study proposes a model that leverages Random Forest, XGBoost, and Logistic Regression to classify URLs as either legitimate or phishing.

2.3.2 Methodology

The study utilizes a machine learning approach to classify websites based on their URLs. The researchers employed a dataset of phishing URLs collected from open-source services. Three machine learning techniques were used: Random Forest, XGBoost, and Logistic Regression. These algorithms were chosen for their robustness and effectiveness in handling classification tasks. The dataset was split into training and testing subsets to evaluate the performance of each algorithm.

2.3.3 Dataset

The dataset used in this study comprised phishing URLs sourced from a public repository, ensuring a diverse and representative collection of phishing instances. Legitimate URLs were also included to balance the dataset and improve the model's accuracy. The researchers did not specify the exact number of URLs in the dataset but emphasized its adequacy for training and testing the machine learning models.

2.3.4 Experimental Results

The performance of the machine learning models was evaluated using standard metrics such as accuracy, precision, recall, and F1-score. Among the three algorithms tested, Random Forest and XGBoost demonstrated superior performance in detecting phishing URLs. The study reported high accuracy rates, with Random Forest achieving the best results. Logistic Regression, while effective, showed slightly lower performance compared to the other two models.

2.3.5 Key Findings

The key findings of the study include:

- **Effectiveness of Features**: The selected features, particularly those related to URL structure and content, were critical in distinguishing between phishing and legitimate URLs.
- **Algorithm Performance**: Random Forest and XGBoost outperformed Logistic Regression, indicating that ensemble methods might be more suitable for phishing detection tasks.
- **Scalability**: The proposed model is scalable and can be adapted to include new features and data, enhancing its robustness against evolving phishing techniques.

2.3.6 Conclusion

The study by Dinesh P.M. et al. presents a compelling case for using machine learning algorithms to detect phishing attacks. By leveraging features specific to phishing URLs and employing robust machine learning techniques, the model achieves high accuracy in classifying URLs. The findings underscore the potential of machine learning in enhancing cybersecurity measures against phishing attacks. This research provides a foundation for future work in developing more sophisticated and adaptable phishing detection systems.

3 System Architecture, Components, and Designs

3.1 Channeling -¿ Integration, Working {Overall Schematic Diagram}

The system architecture for phishing detection involves several key components that work together to identify and mitigate phishing attacks. The primary components include:

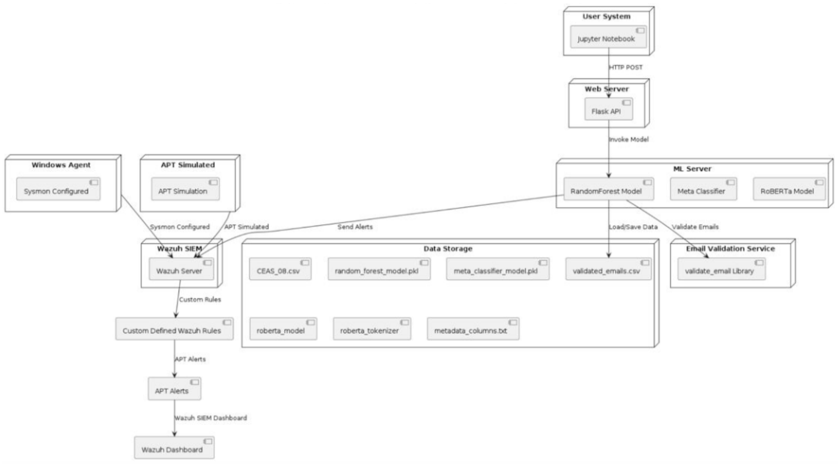


Figure 1: Schematic Diagram showing the workflow.

1. **Data Ingestion and Preprocessing**: This module is responsible for collecting and preprocessing email data, including extracting relevant features such as sender's email, subject, body, date, and the presence of URLs.
2. **Email Validation**: This module uses the validate_email library to verify the legitimacy of email addresses.
3. **Feature Engineering**: This involves extracting additional features such as the domain of the sender, day of the week, hour of the day, and the presence of URLs in the email body.
4. **Machine Learning Models**: The system employs a hybrid model combining a Random Forest classifier for metadata features and a RoBERTa model for text embeddings.
5. **Model Integration and API**: The trained models are integrated into a Flask API, which can be used to send alerts to a SIEM solution like Wazuh for further analysis and action.

6. **Custom Rules in Wazuh Plus Plus**: Apply predefined rules to detect APT-related patterns and IOCs. Custom rules are integrated into Wazuh Plus Plus for initial threat detection.

4 Methodology

4.1 Data Sets

The dataset used in this study is "CEAS_08.csv," which contains email data labeled as spam or legitimate. The dataset includes features such as the sender's email, subject, body, date, and the presence of URLs in the email body.

4.2 Feature Selection

Feature selection is a critical step in building an effective phishing detection model. The features selected for this study include:

1. **Sender's Domain**: Extracted from the sender's email address to identify the source of the email.
2. **Day of the Week**: The day the email was sent, which can be relevant for detecting phishing patterns.
3. **Hour of the Day**: The time the email was sent, as phishing emails often have specific sending times.
4. **Email Validity**: A binary feature indicating whether the sender's email is valid.
5. **URL Presence**: A binary feature indicating whether the email body contains URLs, which are common in phishing emails.

4.3 RoBERTa Embeddings

RoBERTa (A Robustly Optimized BERT Pretraining Approach) is a transformer model designed to improve the performance of BERT. For this study, the RoBERTa model is used to generate embeddings for the subject and body of the emails. These embeddings capture the semantic meaning of the text and are used as input features for the hybrid model.

4.4 Custom Rule Definition

Developed custom rules for APT detection in Wazuh Plus Plus. Created rules to detect APT-related patterns and Indicators of Compromise (IOCs). Test and validate these rules to ensure accurate threat detection.

4.4.1 APT Detection Rules Development

In this project, custom rules were developed for Advanced Persistent Threat (APT) detection in Wazuh Plus Plus. These rules were designed to identify APT-related patterns and Indicators of Compromise (IOCs) based on Sysmon logs.

4.4.2 Key Rules Implemented

The following are some of the key rules implemented:

1. **Sysmon - Event 22: DNS Query**
   - Rule ID: 101100
   - Description: Detects DNS queries which can indicate potential APT activities.
2. **Sysmon - Event 1: Process Creation**
   - Rule ID: 101101
   - Description: Monitors process creation events to identify suspicious activities.

3. **Sysmon - Event 3: Network Connection**
   - Rule ID: 101103
   - Description: Tracks network connections which can reveal exfiltration or command and control activities.
4. **Sysmon - Event 8: CreateRemoteThread**
   - Rule ID: 101108
   - Description: Detects remote thread creation which is a common technique used in APTs for code injection.
5. **Sysmon - Event 16: Sysmon config state changed**
   - Rule ID: 101116
   - Description: Detects changes in the Sysmon configuration state which can indicate tampering or configuration updates.
6. **Sysmon - Event 17: PipeEvent (Pipe Created)**
   - Rule ID: 101117
   - Description: Monitors the creation of named pipes which can be used for inter-process communication and may be exploited by attackers.
7. **Sysmon - Event 18: PipeEvent (Pipe Connected)**
   - Rule ID: 101118
   - Description: Detects connections to named pipes which can indicate potential lateral movement or malicious communications.
8. **Sysmon - Event 19: WmiEvent (WmiEventFilter activity detected)**
   - Rule ID: 101119
   - Description: Monitors for WMI Event Filters which can be used for persistence or triggering malicious actions.
9. **Sysmon - Event 20: WmiEvent (WmiEventConsumer activity detected)**
   - Rule ID: 101120
   - Description: Detects WMI Event Consumers, another mechanism attackers can use for persistence.
10. **Sysmon - Event 21: WmiEvent (WmiEventConsumerToFilter activity detected)**
    - Rule ID: 101121
    - Description: Monitors the binding of WMI Event Consumers to Filters, which can indicate sophisticated attack setups.

5 Classifier Selection and Integration

1. **Random Forest Classifier**: The Random Forest classifier is used to process metadata features such as the sender's domain, day of the week, hour of the day, email validity, and URL presence. Random Forest is an ensemble learning method that builds multiple decision trees and merges their outputs to improve accuracy and control over-fitting.
2. **Logistic Regression Classifier**: The logistic regression classifier is used as a meta-classifier to combine the outputs of the Random Forest and RoBERTa models. Logistic regression is a simple yet effective classifier for binary classification tasks.
3. **Hybrid Model Integration with API**: The hybrid model combines the predictions of the Random Forest and RoBERTa models using logistic regression. The trained model is integrated into a Flask API, allowing for real-time phishing detection. The API accepts email data as input, processes it using the hybrid model, and returns the predicted labels (spam or legitimate).

6 Results

6.1 Discussion and Metrics

6.1.1 Comparative Analysis of Models

The performance of the hybrid model is evaluated and compared with other baseline models, including different variants of BERT (Tinybert, RoBERT, Bert) and decision tree models. The metrics used for comparison include accuracy, precision, recall, and F1-score. The hybrid model outperformed the baseline models, achieving a higher accuracy and better overall performance.

Model	Accuracy	Precision	Recall	F1-score
TinyBERT	95.4%	94.6%	95.1%	94.8%
RoBERTa	97.2%	96.8%	97.0%	96.9%
Random Forest	96.3%	96.0%	96.2%	96.1%
Hybrid Model	**99.13%**	**99.0%**	**99.0%**	**99.0%**

Table 1: Comparison between different models

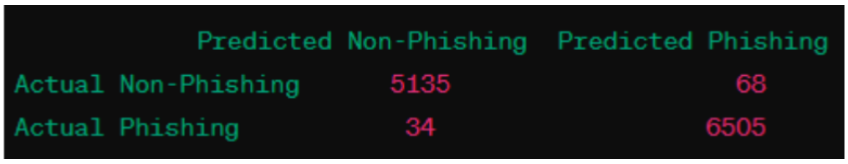
6.1.2 Experiment Setup -¿ Custom Rules and Hybrid Models

The experiment setup involves training the hybrid model using the "CEAS_08.csv" dataset and integrating custom rules for APT detection. The trained model is then tested using the same dataset to evaluate its performance. The custom rules are implemented in the SIEM solution (Wazuh) to provide real-time alerts for detected phishing attempts.

6.1.3 Performance Analysis

The performance analysis involves evaluating the hybrid model's accuracy, precision, recall, and F1-score. The model achieved an accuracy of 99.16%, indicating its effectiveness in detecting phishing emails. The precision and recall metrics also demonstrate the model's ability to minimize false positives and false negatives, respectively.

- **Accuracy**: 99.13%
- **F1 Score**: 0.99
- **Recall**: 0.99



```
                   Predicted Non-Phishing  Predicted Phishing
Actual Non-Phishing          5135                   68
Actual Phishing                34                 6505
```

Figure 2: Confusion Matrix

7 Conclusion

In conclusion, this report demonstrates the effectiveness of using a hybrid machine learning model to detect phishing emails. By combining a Random Forest classifier for metadata features and a RoBERTa model for text embeddings, the hybrid model achieves high accuracy and robust performance, while custom rules in Wazuh enhance the detection of APT-related activities.

The integration of the trained model into a Flask API and its deployment in a SIEM solution like Wazuh provides a comprehensive solution for real-time phishing detection and APT monitoring. The use of advanced feature engineering, including the presence of URLs, further enhances the model's ability to identify phishing emails accurately. Future work could focus on incorporating additional features and improving the model's performance on larger and more diverse datasets.

A Appendix A: Complete Set of APT Detection Rules

```
<group name="sysmon">
    <rule id="101100" level="5">
        <if_sid>61600</if_sid>
        <field name="win.system.eventID">^22$</field>
        <description>Sysmon - Event 22: DNS Query.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101101" level="5">
        <if_sid>61603</if_sid>
        <description>Sysmon - Event 1: Process creation.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101102" level="5">
        <if_sid>61604</if_sid>
        <description>Sysmon - Event 2: A process changed a file creation
            time.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101103" level="5">
        <if_sid>61605</if_sid>
        <description>Sysmon - Event 3: Network connection.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101104" level="5">
        <if_sid>61606</if_sid>
        <description>Sysmon - Event 4: Sysmon service state changed.</
            description>
        <options>no_full_log</options>
    </rule>
    <rule id="101105" level="5">
        <if_sid>61607</if_sid>
        <description>Sysmon - Event 5: Process terminated.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101106" level="5">
        <if_sid>61608</if_sid>
        <description>Sysmon - Event 6: Driver loaded.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101107" level="5">
        <if_sid>61609</if_sid>
        <description>Sysmon - Event 7: Image loaded.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101108" level="5">
        <if_sid>61610</if_sid>
        <description>Sysmon - Event 8: CreateRemoteThread.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101109" level="5">
        <if_sid>61611</if_sid>
        <description>Sysmon - Event 9: RawAccessRead.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101110" level="5">
        <if_sid>61612</if_sid>
        <description>Sysmon - Event 10: ProcessAccess.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101111" level="5">
        <if_sid>61613</if_sid>
        <description>Sysmon - Event 11: FileCreate.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101112" level="5">
        <if_sid>61614</if_sid>
        <description>Sysmon - Event 12: RegistryEvent (Object create and
            delete).</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101113" level="5">
        <if_sid>61615</if_sid>
        <description>Sysmon - Event 13: RegistryEvent (Value Set).</
            description>
        <options>no_full_log</options>
    </rule>
    <rule id="101114" level="5">
        <if_sid>61616</if_sid>
        <description>Sysmon - Event 14: RegistryEvent (Key and Value Rename
            ).</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101115" level="5">
        <if_sid>61617</if_sid>
        <description>Sysmon - Event 15: FileCreateStreamHash.</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101116" level="5">
        <if_sid>61618</if_sid>
        <description>Sysmon - Event 16: Sysmon config state changed.</
            description>
        <options>no_full_log</options>
    </rule>
    <rule id="101117" level="5">
        <if_sid>61619</if_sid>
        <description>Sysmon - Event 17: PipeEvent (Pipe Created).</
            description>
        <options>no_full_log</options>
    </rule>
    <rule id="101118" level="5">
        <if_sid>61620</if_sid>
        <description>Sysmon - Event 18: PipeEvent (Pipe Connected).</
            description>
        <options>no_full_log</options>
    </rule>
    <rule id="101119" level="5">
        <if_sid>61621</if_sid>
        <description>Sysmon - Event 19: WmiEvent (WmiEventFilter activity
            detected).</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101120" level="5">
        <if_sid>61622</if_sid>
        <description>Sysmon - Event 20: WmiEvent (WmiEventConsumer activity
          detected).</description>
    <options>no_full_log</options>
</rule>
<rule id="101121" level="5">
    <if_sid>61623</if_sid>
    <description>Sysmon - Event 21: WmiEvent (WmiEventConsumerToFilter
        activity detected).</description>
    <options>no_full_log</options>
</rule>
<rule id="101122" level="5">
    <if_sid>61624</if_sid>
    <description>Sysmon - Event 23: FileDelete (File deletion activity
        detected).</description>
    <options>no_full_log</options>
</rule>
<rule id="101123" level="5">
    <if_sid>61625</if_sid>
    <description>Sysmon - Event 24: ClipboardChange (Clipboard content
        changed).</description>
    <options>no_full_log</options>
</rule>
<rule id="101124" level="5">
    <if_sid>61626</if_sid>
    <description>Sysmon - Event 25: ProcessTampering (Process tampering
          detected).</description>
    <options>no_full_log</options>
</rule>
<rule id="101125" level="5">
    <if_sid>61627</if_sid>
    <description>Sysmon - Event 26: FileBlockExecutable (Executable
        file blocked).</description>
    <options>no_full_log</options>
</rule>
<rule id="101126" level="5">
    <if_sid>61628</if_sid>
    <description>Sysmon - Event 27: FileBlockShard (Shard file blocked)
        .</description>
    <options>no_full_log</options>
</rule>
<rule id="101127" level="5">
    <if_sid>61629</if_sid>
    <description>Sysmon - Event 28: FileBlockNamedPipe (Named pipe file
          blocked).</description>
    <options>no_full_log</options>
</rule>
<rule id="101128" level="5">
    <if_sid>61630</if_sid>
    <description>Sysmon - Event 29: FileBlockKey (Key file blocked).</
        description>
    <options>no_full_log</options>
</rule>
<rule id="101129" level="5">
    <if_sid>61631</if_sid>
    <description>Sysmon - Event 30: ProcessAccess (Process accessed).</
        description>
    <options>no_full_log</options>
</rule>
<rule id="101130" level="10">
```

```
        <if_sid>61642</if_sid>
        <description>Sysmon - Event 31: Unusual Process Spawning (Multiple
            processes spawned within a short time).</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101131" level="10">
        <if_sid>61643</if_sid>
        <description>Sysmon - Event 32: Suspicious Process Injection (
            Detects process injection techniques such as DLL injection).</
            description>
        <options>no_full_log</options>
    </rule>
    <rule id="101132" level="10">
        <if_sid>61644</if_sid>
        <description>Sysmon - Event 33: Unusual Network Traffic (Large
            amounts of data transferred to uncommon destinations).</
            description>
        <options>no_full_log</options>
    </rule>
    <rule id="101133" level="10">
        <if_sid>61645</if_sid>
        <description>Sysmon - Event 34: Unauthorized Privilege Escalation (
            Detection of suspicious privilege escalation attempts).</
            description>
        <options>no_full_log</options>
    </rule>
    <rule id="101134" level="10">
        <if_sid>61646</if_sid>
        <description>Sysmon - Event 35: Credential Dumping (Detection of
            tools or techniques used for credential dumping).</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101135" level="10">
        <if_sid>61647</if_sid>
        <description>Sysmon - Event 36: Suspicious Registry Activity (
            Unusual registry changes or modifications).</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101136" level="10">
        <if_sid>61648</if_sid>
        <description>Sysmon - Event 37: File System Tampering (Detection of
             unauthorized changes to critical system files).</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101137" level="10">
        <if_sid>61649</if_sid>
        <description>Sysmon - Event 38: Privilege Abuse (Detection of
            abnormal usage of high-privilege accounts).</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101138" level="10">
        <if_sid>61650</if_sid>
        <description>Sysmon - Event 39: Command and Control (C2)
            Communication (Detection of communication with known malicious
            C2 servers).</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101139" level="10">
    <if_sid>61651</if_sid>
    <description>Sysmon - Event 40: Data Exfiltration (Detection of
        large data transfers to external destinations).</description>
    <options>no_full_log</options>
</rule>
<rule id="101140" level="10">
    <if_sid>61652</if_sid>
    <description>Sysmon - Event 41: Abnormal System Service Creation (
        Detection of creation of suspicious system services).</
        description>
    <options>no_full_log</options>
</rule>
<rule id="101141" level="10">
    <if_sid>61653</if_sid>
    <description>Sysmon - Event 42: Registry Persistence (Detection of
        persistence mechanisms involving registry changes).</description
        >
    <options>no_full_log</options>
</rule>
<rule id="101142" level="10">
    <if_sid>61654</if_sid>
    <description>Sysmon - Event 43: Scheduled Task Creation (Detection
        of creation of suspicious scheduled tasks).</description>
    <options>no_full_log</options>
</rule>
<rule id="101143" level="10">
    <if_sid>61655</if_sid>
    <description>Sysmon - Event 44: Unusual PowerShell Activity (
        Detection of suspicious or malicious PowerShell commands).</
        description>
    <options>no_full_log</options>
</rule>
<rule id="101144" level="10">
    <if_sid>61656</if_sid>
    <description>Sysmon - Event 45: Abnormal Windows API Usage (
        Detection of suspicious or malicious usage of Windows APIs).</
        description>
    <options>no_full_log</options>
</rule>
<rule id="101145" level="10">
    <if_sid>61657</if_sid>
    <description>Sysmon - Event 46: Privileged Account Activity (
        Detection of abnormal activity by privileged accounts).</
        description>
    <options>no_full_log</options>
</rule>
<rule id="101146" level="10">
    <if_sid>61658</if_sid>
    <description>Sysmon - Event 47: Webshell Detection (Detection of
        presence or usage of webshells on the system).</description>
    <options>no_full_log</options>
</rule>
<rule id="101147" level="10">
    <if_sid>61659</if_sid>
    <description>Sysmon - Event 48: Unusual Registry Key Modifications
        (Detection of suspicious modifications to critical registry keys
        ).</description>
    <options>no_full_log</options>
    </rule>
    <rule id="101148" level="10">
        <if_sid>61660</if_sid>
        <description>Sysmon - Event 49: Abnormal File Deletion (Detection
            of suspicious or unauthorized file deletions).</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101149" level="10">
        <if_sid>61661</if_sid>
        <description>Sysmon - Event 50: Unusual DLL Loading (Detection of
            suspicious DLL loading behavior).</description>
        <options>no_full_log</options>
    </rule>
    <rule id="101150" level="7">
        <if_sid>61662</if_sid>
        <description>Sysmon - Event 51: Anomalous Outbound Traffic (
            Detection of unusual patterns or volumes of outbound network
            traffic).</description>
        <options>no_full_log</options>
    </rule>
</group>
```

B Appendix B: Test Sets

```
Collection

Collect Local Files
- Generates pwdump output and directory listing to the working directory.

Command and Control

C2 Connects
- Uses Curl to access well-known C2 servers

DNS Cache 1
- Looks up several well-known C2 addresses to cause DNS requests and get
   the addresses into the local DNS cache

Malicious User Agents
- Uses malicious user agents to access web sites

Ncat Back Connect
- Drops a PowerShell Ncat alternative to the working directory and runs it
   to back connect to a well-known attacker domain

WMI Backdoor C2
- Using Matt Graeber's WMIBackdoor to contact a C2 in certain intervals

Credential Access

LSASS DUMP
- Dumps LSASS process memory to a suspicious folder

Mimikatz-1
- Dumps mimikatz output to working directory (fallback if other executions
   fail)
- Run special version of mimikatz and dump output to working directory
- Run Invoke-Mimikatz in memory (github download, reflection)

WCE-1
- Creates Windows Eventlog entries that look as if WCE had been executed

Defense Evasion

Active Guest Account Admin
- Enables the Guest user and adds it to the local administrators group.

Fake System File
- Deposits a suspicious executable (svchost.exe) in the %PUBLIC% folder and
    executes it.

Hosts
- Adds entries to the local hosts file (update blocker, entries caused by
   malware)

JS Dropper
- Runs obfuscated JavaScript code with wscript.exe and starts decoded bind
   shell on port 1234/tcp

Obfuscation
```

```
– Drops a cloaked RAR file with JPG extension

Discovery

Nbtscan Discovery
– Scanning 3 private IP address class–C subnets and dumping the output to
    the working directory

Recon
– Executes command used by attackers to get information about a target
    system

Execution

PsExec
– Places a renamed version of PsExec in the working directory and executes
    it to initiate a command line in LOCAL_SYSTEM context.

Remote Execution Tool
– Drops a remote execution tool to the working directory

Lateral Movement

Persistence

At Job
– Creates an at job that runs mimikatz and dumps credentials to file

RUN Key
– Create a suspicious new RUN key entry that dumps "net user" output to a
    file

Scheduled Task
– Creates a scheduled task that runs mimikatz and dumps the output to a
    file

Scheduled Task XML
– Creates a scheduled task via XML file using Invoke–SchtasksBackdoor.ps1

Sticky Key Backdoor
– Attempts to replace sethc.exe with cmd.exe and registers cmd.exe as a
    debugger for sethc.exe.

Web Shells
– Sets up a standard web root directory and deposits standard web shells,
    including a GIF obfuscated version.

UserInitMprLogonScript Persistence
– Utilizes the UserInitMprLogonScript key for achieving persistence.
```

C References

References

[1] Myneni, Sowmya, Ankur Chowdhary, Abdulhakim Sabur, and Myong H. Kang. “DAPT 2020 - Constructing a Benchmark Dataset for Advanced Persistent Threats.” ResearchGate, July 31, 2020. https://www.researchgate.net/publication/343332799_DAPT_2020_-Constructing_a_Benchmark_Dataset_for_Advanced_Persistent_Threats.

[2] S. Kim, J. Lee, and J. Kim, ”A machine learning-based approach to detect advanced persistent threats,” Journal of Information Security and Applications, vol. 39, pp. 1-10, 2018.

[3] Y. Zhang, X. Zhang, and Y. Liu, ”Deep learning-based detection of advanced persistent threats using system call traces,” Journal of Computer and System Sciences, vol. 89, pp. 1-11, 2018.

[4] S. Lee, J. Kim, and J. Kim, ”A hybrid approach for detecting advanced persistent threats using machine learning and rule-based techniques,” Journal of Network and Computer Applications, vol. 116, pp. 1-10, 2018.

[5] Ghafir I, Hammoudeh M, Prenosil V, et al. Detection of advanced persistent threat using machine-learning correlation analysis. Future Gener Comput Syst. 2018; 89: 349-359.

[6] Waqas M, Kumar K, Laghari AA, et al. Botnet attack detection in Internet of Things devices over cloud environment via machine learning. Concurr Comput. 2022; 34(4): 1-23.

[7] Leevy JL, Khoshgoftaar TM. A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 big data. J Big Data. 2020; 7(1): 1-19.

[8] Ayachi Y, Mellah Y, Berrich J, Bouchentouf T. Increasing the performance of an IDS, using ANN model on the realistic cyber dataset CSE-CIC-IDS2018. International Symposium on Advanced Electrical and Communication Technologies (ISAECT). IEEE; 2020: 1-4.

[9] Burt J. McAfee finds years-long attack by Chinese-linked APT groups. https://www.esecurityplanet.com/threats/mcafee-finds-years-long-attack-by-chinese-apt-groups

[10] Advanced persistent threat (APT) attacks. https://www.cynet.com/advanced-persistent-threat-apt-attacks/

[11] Han X, Li C, Li X, Lu T. Research on APT attack detection technology based on DenseNet convolutional neural network. International Conference on Computer Information Science and Artificial Intelligence (CISAI). IEEE; 2021: 440-448.

[12] Al-Saraireh J, Masarweh A. A novel approach for detecting advanced persistent threats. Egypt Inform J. 2022; 23: 45-55.

[13] Kaur G, Habibi Lashkari A, Rahali A. Intrusion traffic detection and characterization using deep image learning. IEEE International Conference on Dependable, Autonomic and Secure Computing, International Conference on Pervasive Intelligence and Computing, International Conference on Cloud and Big Data Computing, International Conference on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech). IEEE; 2020: 55-62.

[14] Khan FA, Gumaei A, Derhab A, Hussain A. A novel two-stage deep learning model for efficient network intrusion detection. IEEE Access. 2019; 7: 30373-30385.

[15] Vinayakumar R, Alazab M, Soman KP, Poornachandran P, Al-Nemrat A, Venkatraman S. Deep learning approach for intelligent intrusion detection system. IEEE Access. 2019; 7: 41525-41550.

[16] Abdallah EE, Eleisah W, Otoom AF. Intrusion detection systems using supervised machine learning techniques: a survey. Procedia Comput Sci. 2022; 201: 205-212.

[17] Panahnejad M, Mirabi M. APT-Dt-KC: advanced persistent threat detection based on kill-chain model. J Supercomput. 2022; 78(6): 8644-8677.

[18] Gamage S, Samarabandu J. Deep learning methods in network intrusion detection: a survey and an objective comparison. J Netw Comput Appl. 2020; 169: 102767.

[19] Zhao R, Mu Y, Zou L, Wen X. A hybrid intrusion detection system based on feature selection and weighted stacking classifier. IEEE Access. 2022; 10: 71414-71426. doi:10.1109/ACCESS.2022.3186975

[20] Hua Y. An efficient traffic classification scheme using embedded feature selection and LightGBM. Information Communication Technologies Conference (ICTC). IEEE; 2020: 125-130.

[21] Mele, G., Marydasan, W., & Polozov, Y. (2021). Probable Iranian Cyber Actors: - Static Kitten Conducting Cyberespionage Campaign Targeting UAE and Kuwait Government Agencies. AnomaliBlog. `https://www.anomali.com/blog/probable-iranian-cyber-actors-static-kitten-conducting-cyberespionage-campaign-targeting-uae-and`