Enhancing Cybersecurity Intelligence through Machine Learning: Clustering and Forecasting Analysis of Honeypot Data

David Rosado Rodríguez
Agència de Ciberseguretat de Catalunya
Barcelona, Spain
drosado.pwc@ciberseguretat.cat

Santiago Romeu Sala
Agència de Ciberseguretat de Catalunya
Barcelona, Spain
sromeu@ciberseguretat.cat

Isart Canyameres Gimènez
Agència de Ciberseguretat de Catalunya
Barcelona, Spain
icanyameres@ciberseguretat.cat

Tomàs Roy Catala
Agència de Ciberseguretat de Catalunya
Barcelona, Spain
troy@ciberseguretat.cat

***Abstract*—In an era dominated by an escalating digital threat landscape, proactive cybersecurity measures are increasingly crucial. This paper utilizes a comprehensive dataset provided by the Global Cyber Alliance, comprising cyberattack records collected from distributed honeypots worldwide. The study pursues two main objectives: conducting a cluster analysis to unveil attack patterns and developing a forecasting model to estimate the number of cyberattacks. Beginning with a exploratory data analysis, the research offers insights into attack characteristics across honeypot locations, followed by the application of advanced clustering techniques to identify common patterns. For this purpose, malware hashes are also analyzed, paying special attention to those not identified or unrecognized by the VirusTotal malware database, as they pose the major risk to the organizations. The final aspect of this study is the development of a forecasting model aiming to forecast expected cyberattack occurrences, offering valuable support for resource allocation and strategic planning within cybersecurity.**

I. Introduction

Systems with publicly exposed IP addresses remain highly susceptible to mass cyberattacks that can lead to unauthorized intrusions, data leaks, ransomware infections, and other forms of malicious activity, which can have severe effects on an organization's security infrastructure. Such exposure creates vulnerable entry points that are often exploited by attackers, using advanced malware and evolving tactics that traditional security mechanisms struggle to defend against. To tackle this issue, there is an urgent need for advanced methods to analyze, detect, and mitigate the diverse and complex nature of these large-scale attacks. Our research, which leverages honeypots, aims to address this challenge by employing a combination of cluster analysis, malware hash recognition, and forecasting techniques to identify, understand, and mitigate these cyberthreats proactively.

Honeypots are sophisticated cybersecurity mechanisms strategically designed to detect, deflect, and analyze malicious activities within networks. These isolated systems are meticulously monitored and equipped to analyze attackers, providing a controlled environment for observing and understanding various cyber threats and strategies employed by adversaries [1]–[4]. Honeypots can be categorized into different types based on their deployment and functionality. Low-interaction honeypots provide a lightweight solution for detecting and studying common threats. They mimic essential services and protocols such as Secure Shell (SSH) or File Transfer Protocol (FTP) without granting attackers access to the operating system (OS). In contrast, high-interaction honeypots provide a fully immersive environment that closely imitate a production system, allowing researchers to observe sophisticated attack techniques in action. Additionally, there exists a middle ground known as medium-interaction honeypots, which strike a balance between low and high interaction. Medium-interaction honeypots offer a compromise between realism and resource consumption, providing a more comprehensive view of attacker behavior without the overhead associated with high-interaction deployments [5]–[7].

Certain companies are operating extensive networks of honeypots, known as honeyfarms, spread across various geographical locations. These networks allow them to collect valuable information on cyberattacks and provides a geographical strategic perspective, enabling comparison of cyberactivities across different countries.
For this research, we collaborate with the Global Cyber Alliance (GCA), a non-profit organization dedicated to improving online safety by reducing cyber risks. The GCA operates a honeyfarm, called Automated IoT Defense Ecosystem (AIDE), operational since late November 2021. This setup comprises 221 honeypots distributed across 55 countries, utilizing IPv4 addresses that were not previously used for honeyfarms or darknets. The honeyfarm is deployed

using the open-source software Cowrie [8], which is a medium-interaction honeypot that provides a realistic shell environment for attackers and allows capturing extensive logging's of their activity.

This paper focuses on analyzing 29 month of data, from November 2021 to April 2024, collected by the aforementioned honeyfarm. We employ state-of-the-art Machine Learning (ML) and Deep Learning (DL) methodologies to examine cyberattack records where the attacker executes at least one command, totaling over 34 million sessions. This subset of data, is highly valuable, as it allows us to extract numerous insights using ML and DL techniques based on the commands introduced by the attackers. The whole honeyfarm can be divided into five distinct groups based on various scenarios: when an attacker attempts to log in, when the attacker's credentials are incorrect (given that honeypots accept "root" as a username and any password other than "root"), when the attacker successfully logs in but does not enter any commands, when the attacker introduces commands, and when the attacker attempts to access external resources via URI. *Munteanu et al.* [9] conducted an extensive analysis of the complete dataset using this partition, examining and comparing the activity within each group. We highly recommend reading their article to gain a comprehensive understanding of the database composition.
Following the notation of the aforementioned article, our focus lies on the **CMD** (sessions with a successful login and executed commands) and **CMD+URI** (sessions with a successful login, executed commands and access to an external resource via a URI) groups. We chose to focus on this particular data because sessions that involve executed commands and access to external resources are more likely to involve serious threats, such as malware installation, data exfiltration, or system modification.

Our objective extends beyond a deep analysis; first, we employ cluster analysis techniques [10] to further partition this subset of data, facilitating the identification of attack patterns. Specifically, we utilized a spectral clustering algorithm, leveraging the embeddings of the introduced commands as input, allowing the categorization of attack instances into distinct clusters. Following the initial cluster analysis, we focused our efforts on providing valuable insights to enhance cybersecurity strategies. Malware hashes are analyzed to gain deeper insights into the characteristics and patterns of attacks. The honeyfarm provides access to the VirusTotal malware database [11], allowing us to identify instances where malware is either unknown or unrecognized. This method seeks to provide proactive strategies to address attacks that involve introducing previously unknown or unrecognized hashes into the system.
Furthermore, we apply forecasting analysis methods to predict the frequency of attacks. For this purpose, we will compare three distinct modeling approaches: Classical Models, including Autoregressive Integrated Moving Average (ARIMA) [12]; ML models, such as XGBoost [13] and Support Vector Machines (SVM) [14]; and DL models, focusing on Long Short-Term Memory (LSTM) networks [15]. By evaluating and comparing the performance of these methodologies on historical attack data, we aim to identify the most effective approach for forecasting the likelihood of future attacks within our system. This comparative analysis will enable us to select the most suitable forecasting model, empowering us to implement proactive measures to mitigate potential security threats and enhance overall system security.

II. Related Work

Munteanu et al. [9] conducted a thorough examination of the dataset we also utilized in our analysis. They divided the dataset into the five distinct groups previously described and used it to examine and compare the activity within each group. They delve deeply into the sessions and activity per honeypot, examining them both individually and collectively.
Kheirkhah et al. [16] examined brute-force SSH attacks detected by a low-interaction honeypot. The researchers identified three distinct groups based on the commands used: successful attempts without additional commands, scanning attacks aimed at gathering system information, and the most concerning sessions involving the download and execution of malware.
Melese & Avadhani [17] deployed a honeypot to analyze SSH attacks. In this study, the authors identified a common sequence of steps that cyberattackers typically follow when infiltrating a system. Additionally, the study aimed to identify common shell commands employed by attackers and tracked their geographic locations via IP addresses, documenting SSH client usage in each session. Notably, a significant number of attacks were found to originate from China and the United States.
Fraunholz et al. [18] conducted an investigation using data collected over three months from a honeypot. Initially, they conducted a statistical analysis focusing on login credentials and IP addresses. They then categorized the data based on the attackers' skill levels, assigning higher scores when attackers introduced predefined commands. Finally, they employed ML techniques for clustering analysis using two sets of features: login credentials and introduced commands. The findings indicated that ML-based methods effectively formed distinct clusters, while the skill-based approach faltered due to honeypot configuration issues.
Valero et al. [19] examined SSH attacks captured from honeypots. They categorized them into three groups according to threat severity and applied different supervised ML algorithms to classify them, using these threat levels as labels. The most successful model achieved an F1-score of 0.9714, validating the effectiveness of the initial clustering.
Tabari et al. [20] introduced an innovative method for constructing a honeypot ecosystem. In this study, researchers collected data and conducted a cluster analysis by creating a similarity matrix for each unique command pair. They

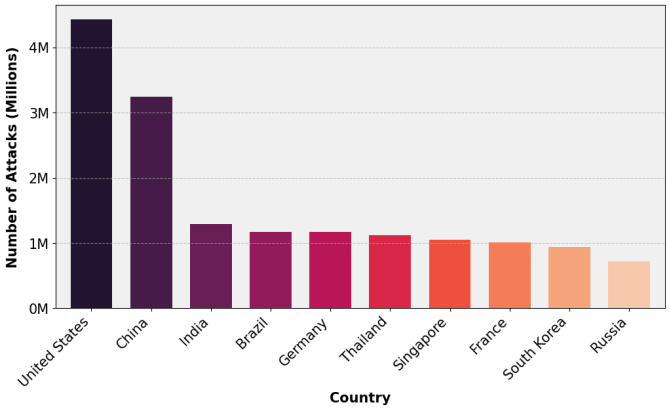

Fig. 1: Top ten countries where cyberattacks with at least one executed command originate.

Password	
1) admin	2) password
3) 1234	4) 123
5) 12345	6) 1

TABLE I: Top 5 most used successful passwords (row-major order).

employed a Gaussian Mixture Model (GMM) [21] to generate 50 distinct clusters, helping them identify the objectives of attackers within each group.

Transitioning to forecasting analysis, different statistical models, such as ARIMA [12], SARIMA [12], and GARCH [22], have been historically utilized to predict cyberattacks [23]–[27]. Recently, a specific type of DL forecasting algorithm known as LSTM has demonstrated superior performance compared to traditional models like ARIMA. According to [28], the DL approach significantly outperforms ARIMA models, reducing prediction errors by more than 80%.

In contrast to the statistical methods commonly used in research, *Fang et al.* [29] pioneer the use of DL techniques to predict cyberattacks. They employ a bi-directional Recurrent Neural Network (RNN) [30] with Long Short-Term Memory (BRNN-LSTM), demonstrating superior prediction accuracy compared to traditional statistical methods.

In summary, significant effort has been devoted to studying and analyzing honeypot data. *Munteanu et al.* [9] conducted an in-depth examination of the same dataset we are using. Other studies [16], [18]–[20] have also analyzed honeypot data, aiming to categorize it for a deeper understanding of the insights it offers. In the context of forecasting analysis, the studies [23]–[27] employ traditional statistical methods, such as ARIMA and SARIMA, to predict cyberattacks. In contrast, [29] introduces a novel approach by utilizing DL techniques for this purpose.

Our research not only covers a thorough examination of honeypot data but an in-depth examination of malware hashes.

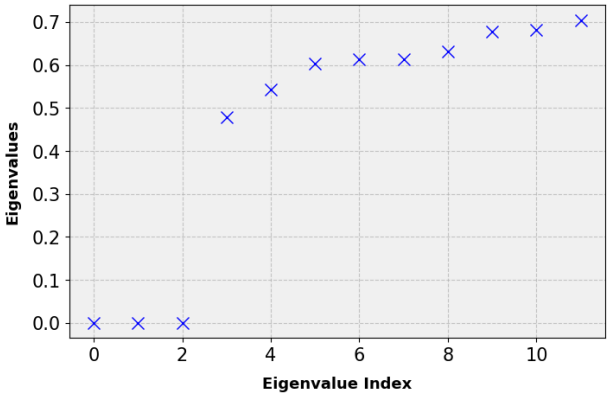

Fig. 2: Eigenvalues of L_{rw} in ascending order.

Original	Normalized
/bin/busybox ZSQCG	*/bin/busybox token*
root:Q9szJ—chpasswd	*root:token—chpasswd*

TABLE II: Command normalization examples.

We categorize this data using advanced ML techniques by using the commands used by attackers. Additionally, we explore the differences within each group of data to develop practical recommendations for effectively countering these attacks. We also investigate the forecasting task by comparing traditional statistical models, ML techniques, and DL algorithms. While traditional statistical models and DL algorithms have been studied in previous research for this task, there is no record of ML techniques being applied.

III. Honeyfarm Dataset

This research utilizes 29 month of data, spanning from late November 2021 to April 2024 collected from GCA honeyfarm. We are focusing solely on data where the attacker has executed at least one command, totaling over 34 million sessions.

The available features in the honeyfarm's database can be categorized into three main groups:

- Geographical features of the attacker, e.g. location.
- Geographical features of the honeypot.
- Behavioral data, e.g. executed commands, login credentials, and session time.

Geographical features are crucial in honeyfarms because having these sensors distributed worldwide allows us to gain valuable insights and comparisons regarding malicious activities across countries. This distribution helps in identifying the primary locations of attackers contributing to the networks. Figure 1 illustrates this information, showing that the majority of attackers are situated in the United States, China, and India.

While the chart shows where cyberattacks appear to originate, it may not accurately reflect the true locations of

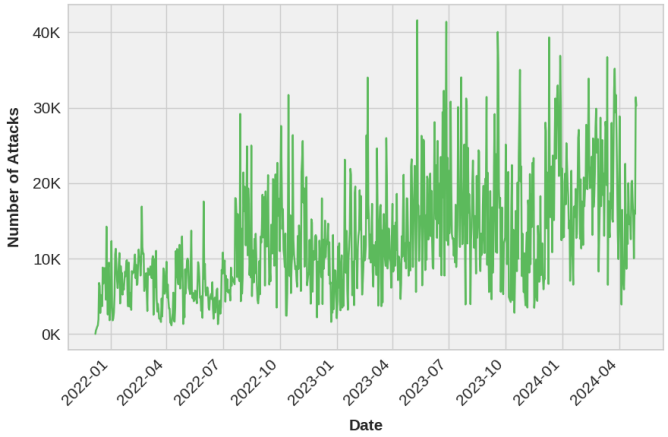

Fig. 3: Daily cyberattacks with at least one executed command in the **United States** from November 2021 to April 2024.

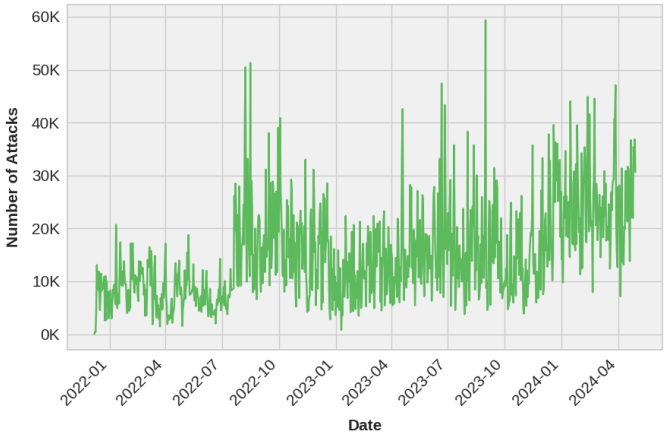

Fig. 4: Daily cyberattacks with at least one executed command in **Europe** from November 2021 to April 2024.

the attackers due to the widespread use of VPNs and the Tor network, which mask real locations. These tools route traffic through multiple relay nodes, often ending in infrastructure-heavy regions like the United States, creating a misleading concentration of attacks in these areas. Data centers and cloud services can also be exploited for malicious purposes, further inflating attack counts. For example, although Russia is known for cybercriminal activity, it ranks lower, likely due to attackers using VPNs or Tor to obscure their actual origins.

The cyberattacker establishes a Transmission Control Protocol (TCP) connection either through SSH using port 22 or Telnet using port 23 to gain access to the system, as these are the connection methods configured by Cowrie within the honeyfarm environment. When a connection is established successfully, the system records a new entry in the database. This entry documents various behavioral aspects of the attack, such as the attack's start and end times, the IP and port of both the honeypot and the client, the commands executed by the attacker, and any credentials (username/password) utilized by the attacker. If SSH is employed, the system additionally logs the SSH version used by the client.

Out of the over 34 million entries in the database, 93.8% represent sessions utilizing SSH, while the remaining 6.2 % are attributed to connections made via Telnet. Additionally, the most commonly utilized SSH version among the attackers is *SSH-2.0-Go*, accounting for 62.2% of cases, followed by *SSH-2.0-libssh-0.9.6* at 15.2%.
Each session initiated via SSH or Telnet is terminated either by the client closing the TCP connection or by the honeypot timing out, with a configured timeout duration of three minutes. The vast majority of attacks, approximately 97%, are terminated by the client within one minute. This suggests that either the attacker identifies the honeypot before the configured timeout and promptly ceases activity, or the attacker merely executes a few commands to scan the system before departing, possibly with intentions to revisit later.

When it comes to the credentials used by the intruders, remember that the username only allows the string *root*, and for successful login, the password must be something other than *root*. Table I displays the top 5 most successful passwords. Notice that they are quite predictable and expected.

When looking at the commands most frequently used by intruders, we notice a significant trend in attacks primarily focused on scanning operations to gather system information. Specifically, the most prevalent activity observed on the honeypot involves a command that serves to test the functionality of the compromised console by printing the string *ok*. This command is: *echo -e "\\x6F\\x6B"*. Commands like *uname -a* or *uname -s -v -n -r -m* are also commonly found in the database. These commands retrieve system information such as the OS and the processor architecture.

IV. Methods and Algorithms

We have two main research goals:

1) To perform a cluster analysis on the data. This will help us to group attack instances into different clusters, allowing us to understand the patterns of malicious activities better.
2) To develop forecasting models that can predict the frequency of attacks based on previous information.

A. Cluster Analysis

Cyberattacks detected by the honeyfarm are not labeled, posing a challenge for analysis. Employing unsupervised learning methods to categorize the data could offer clarity and enhance our understanding of its nature.

We suggest a Spectral Clustering approach that utilizes the commands introduced by attackers to categorize the data into distinct clusters [31]. Spectral Clustering is a clustering method that utilizes a similarity matrix of the data to identify

clusters. For a given set of data points, $x_1, \ldots, x_n$, and a defined measure of similarity, $s_{ij} \geq 0$, between all pairs of data points x_i and x_j, we can represent the data using a similarity graph $G = (V, E)$. Each vertex v_i in this graph represents a data point x_i and each vertex is connected by an edge weighted by s_{ij}. Clustering using this framework involves creating a partition such that the edges between different groups have very low weights (meaning that points in different clusters are dissimilar from each other) and the edges within a group have high weights (meaning that points within the same cluster are similar to each other).

The general approach consists of applying a standard clustering technique to a pertinent set of eigenvectors derived from the Laplacian matrix of the similarity matrix. While there are different definitions for the Laplacian matrix, we opted for the random walk normalized Laplacian, which is commonly used in the field [32]. If we represent the similarity matrix as $S = (s_{ij})$, the random walk normalized Laplacian is defined as follows:

$$L_{rw} = I - D^{-1}S \quad \text{where} \quad D_{ij} = \begin{cases} \sum_k s_{ik} \text{ if } i = j, \\ 0 \text{ otherwise.} \end{cases}$$

In summary, the algorithm proceeds through the following steps:

1) Compute the random walk normalized Laplacian from the similarity matrix.
2) Compute the first k eigenvectors and consider the normalized matrix formed by these.
3) Use a standard clustering technique (e.g., K-means).

Our analysis employs a cosine similarity matrix derived from the embeddings of commands, generated through Term Frequency-Inverse Document Frequency (TF-IDF) vectorization [33]. Specifically, TF-IDF is applied to the attacker-introduced commands, enabling the computation of the cosine similarity between each command pair. However, computing a 34 million by 34 million matrix by computing the cosinus of the angle between each pair of command embeddings is computationally expensive. To mitigate this, we decided to reduce the dataset to a three-month period, from May to July 2023, resulting in approximately 5 million records, which is still substantial. We also only consider unique command attacks to avoid redundant computations. Additionally, by normalizing some commands to ensure variability, we end up with a set of 3299 unique commands for cosine similarity computation. The normalization process retains all essential information, as demonstrated in Table II with examples of normalized commands.

Finally, we applied the Spectral Clustering algorithm mentioned before to the three-month dataset. To determine the optimal number of clusters, we use the eigengap heuristic technique on the Laplacian matrix [31]. This approach involves several steps: first, computing the eigenvalues of the Laplacian; second, arranging them from smallest to largest; and finally, identifying the point where a gap occurs. Figure 2 illustrates such a gap between the 3rd and 4th eigenvalues, where the difference $|\lambda_4 - \lambda_3|$ is relatively large. Hence, in our method, we choose 3 as the number of clusters.

Once we have successfully grouped the three-month data into the three distinct clusters, our next step is to extend these results to the two-year dataset we have. To achieve this, we employ a Random Forest (RF) supervised learning approach [34], using the clusters identified through spectral clustering as labels. By doing so, we train a model capable of accurately classifying the data into these predefined groups. With an 80-20 train-test split, the model is trained using the TF-IDF vectorization of the commands embeddings. Our model achieves an impressive f1-score of 0.967 on the test set, indicating its robustness in classifying new cyberattack records. Consequently, we can effectively label the two-year dataset and establish three distinct groups within the database based on the clustering results.

B. Forecasting Analysis

Let us transition to the primary focus of this study: the forecasting analysis [12]. Our objective is to forecast the daily count of cyberattack records detected by a specific region. For this task, we utilize 29 month of data, from November 2021 to April 2024. By aggregating data from multiple honeypots within a country or continent and treating them collectively as a unified sensor, we aim to anticipate the volume of cyberattacks that a country or continent might encounter in the coming days. In this study, we analyze time series data collected from the United States (US) and Europe (EU), as illustrated in Figure 3 and Figure 4 respectively.

1) Classical approach: Classical models such as ARIMA [12], SARIMA [12] or GARCH [22] have been historically utilized to predict the daily frequency of cyberattacks [23]–[27]. These traditional statistical models are particularly useful when the statistical properties of the series does not change over time (stationarity) and does not have any outliers present. However, most real-world datasets exhibit non-stationarity. To address this, differencing should be considered to transform the non-stationary data into a stationary form. We choose to employ the Dickey-Fuller test to assess the stationarity of our series [35]. The test examines whether a unit root exists in an autoregressive (AR) time series model, with the null hypothesis suggesting the presence of a unit root, and the alternative hypothesis suggesting stationarity.

In our study of honeyfarm data, we utilize two classical forecasting models: ARIMA [12] and Prophet [36]. In the ARIMA classical model, three parameters are essential: (p, d, q), where p represents the autoregression order, d signifies the differencing degree, and q indicates the moving average order. To determine the appropriate value for d, we conduct the Dickey-Fuller test on both the US and EU datasets, employing a significance level of $\alpha = 0.05$. The test results reveal that the US time series is non-stationary, while the EU time series is stationary. Consequently, we set $d = 1$ for the US data and $d = 0$ for the EU data. To determine the optimal values for p and q, we conduct a grid search

to identify the combination of parameters that minimize the Akaike Information Criterion (AIC) [37].

While ARIMA is widely recognized, Prophet, developed by Facebook's data science team in 2018, may not enjoy the same level of recognition. Prophet employs a Bayesian framework wherein the algorithm calculates the posterior distribution of the model parameters rather than relying on singular point estimates. This approach enables the algorithm to generate probabilistic forecasts, providing a measure of uncertainty surrounding the central forecast. At its core, Prophet represents time series data by combining trend, seasonality, and noise elements.

2) Machine Learning approach: We aim to implement both ML and DL models in order to compare their performance and determine whether this Artificial Intelligence (AI) approach outperforms the classical methods mentioned above. For these methodologies, we must transform the forecasting task into a ML problem by defining the daily cyberattack count as the target variable and integrating pertinent features related to the problem. We use three different type of features:

1) **Date-related features**: These encompass temporal data such as the month, year, day of the year, week of the year, quarters, and whether it is a working day.
2) **Lagged features**: These involve historical counts of cyberattacks, which are valuable for forecasting. By incorporating information on cyberattack counts from earlier times (e.g., $t - i, \geq 1$), we can enhance the prediction at a given time.
3) **Statistical features**: These consist of statistical characteristics derived from past observations, including the mean, standard deviation, variance, minimum, and maximum of the preceding $j \geq 1$ days.

These features are considered as parameters to be tuned in the model, aiming to identify the optimal values for i and j that enhance the performance of the ML or DL model for each time series.

To ensure that our models can generalize well, we divide the dataset into three parts: training, validation, and testing. The testing set consists of the most recent month's data, the validation set contains the data from the month before that, and the training set comprises the rest of the data. The validation set is utilized for fine-tuning the hyperparameters of our ML models. We employ Optuna [38], a hyperparameter optimization framework, to automate this process by defining a search space of hyperparameters and specifying an objective function (e.g., maximize accuracy or minimize loss). We train various models to conduct further comparisons and select the best one. These models include XGBoost [13], Support Vector Machines (SVM) [14], Random Forest (RF) [34], Gradient Boosting (GD) [39] and Elastic Net Regression (ENR) [40].

3) Deep Learning approach: Utilizing the previously mentioned consistent data partitions across training, validation, and testing sets, we employ a DL forecasting approach to predict

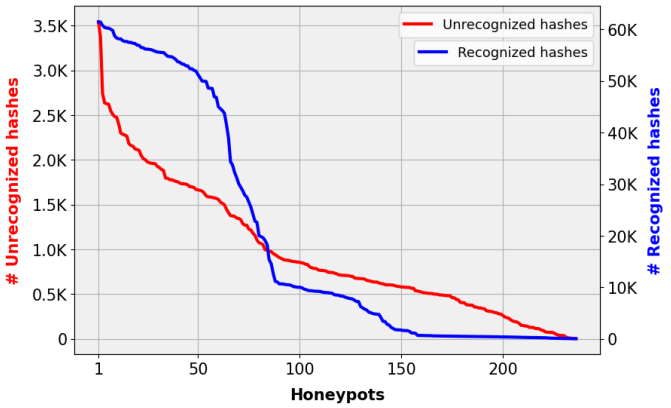

Fig. 5: Number of *recognized* (right axis) and *unrecognized hashes* (left axis) per honeypot, sorted by activity.

Group 1	Group 2	Group 3
grep name	*/bin/busybox*	*sh*
echo \\x6F\\x6B	*cp /bin/echo*	*chmod 777*
cat /proc/cpuinfo	*enable system shell*	*rm-rf*
uname -a	*cat /proc/mounts*	*tftp*
uname -s-v-n-r-m	*rm*	*wget*

TABLE III: Top 5 commands identified in each group.

the daily occurrences of cyberattacks. Recent studies demonstrate the applicability of DL models in forecasting tasks, showcasing their superiority over traditional statistical models [28], [41]. Notably, Long Short-Term Memory (LSTM) networks [15] consistently yield promising results in such forecasting endeavors. LSTMs, a type of Recurrent Neural Network (RNN) [42], are specifically designed to address the shortcomings of traditional RNNs in capturing and learning long-term dependencies within sequential data. Despite numerous RNN-based architectures developed for temporal forecasting [43], [44], older RNN variants often encounter challenges in effectively learning long-range dependencies [45] due to issues with exploding and vanishing gradients [46], attributed to their infinite lookback window. Consequently, LSTMs were devised to mitigate these limitations by enhancing gradient flow within the network. While more intricate networks, such as the bi-directional RNN [30] with LSTM proposed by *Fang et al.* [29], can be employed for forecasting the number of attacks, utilizing a straightforward LSTM model is adequate to achieve satisfactory results.

To train the model, we first organize the data into sequences of tensors, each containing three elements. The label for each sequence is set as the next value in the series. Additionally, we scale the data to enhance model performance. Once the data is arranged sequentially, we construct a simple LSTM model. This model consists of two LSTM layers, each consisting of 64 neurons, with dropout used for regularization. After the LSTM layers, a dense layer with 32 neurons is added, leading to the final output layer.

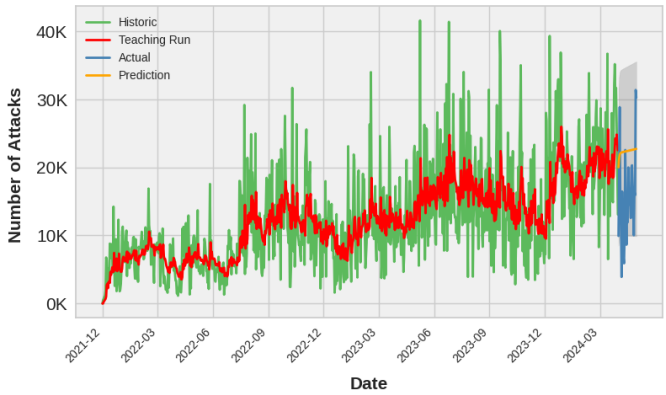

Fig. 6: **United States** time series forecasting with ARIMA model.

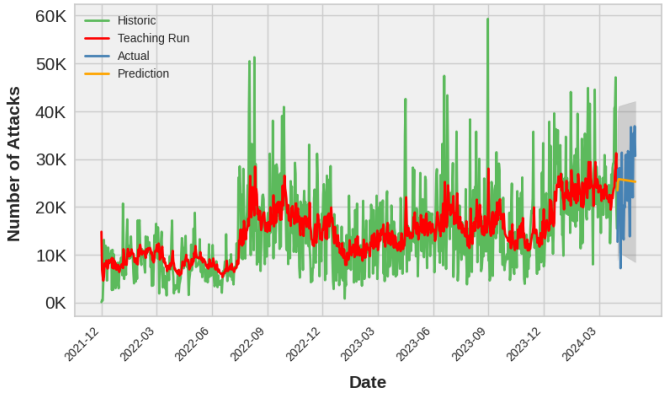

Fig. 7: **Europe** time series forecasting with ARIMA model.

4) Metrics evaluation: Let $(y_1, \ldots, y_N)$ be the observed values and $(\hat{y}_1, \ldots, \hat{y}_N)$ be the predicted values. In order to evaluate the accuracy of the ML and DL framework, we propose using the following widely used metrics [23], [29].

- Mean Square Error (MSE): $MSE = \sum_{i=1}^{N}(y_i - \hat{y}_i)^2/N$.
- Mean absolute percentage error (MAPE): $MAPE = \sum_{i=1}^{N}|(y_i - \hat{y}_i)/y_i|/N$.

When performing hyperparameter tuning on the ML/DL-based approach, we select the model that minimizes the MSE.

V. Results

In this chapter, we analyze the results obtained from the clustering and forecasting analysis.

A. Cluster Analysis

Let us examine the three distinct groups formed by applying spectral clustering to the text embeddings produced using TF-IDF in the commands introduced by the attackers.

The method produces three distinct groups, with one being dominant, comprising 91% of the records, while the remaining 9% are distributed evenly among the other two groups. Upon examining the attacks within each cluster, distinct patterns of behavior emerge among the groups. Table III presents the top 5 commands identified in each cluster. Commands like *echo -e "\\x6F\\x6B"*, *uname -a*, or *uname -s -v -n -r -m* are predominantly associated with group 1, which focuses on gathering system information exclusively. Conversely, commands such as *chmod 777*, *wget*, or *sh* are primarily observed in group 3, indicative of activities involving altering file permissions, downloading potentially malicious files from the web, or executing shell scripts containing potential malware. It is noteworthy that group 3 poses a significantly greater threat compared to group 1, as its activities involve attempting to install malware or manipulate system file permissions. Additionally, there exists a middle group characterized by commands like */bin/busybox*, which execute custom functionalities via BusyBox. This middle group encompasses attacks focused not only on acquiring system information but also on attempting system modifications. In summary, groups can be categorized as follows:

- **Group 1**: Represents the **lowest threat** level. Activities mainly involve gathering system data and conducting scans.
- **Group 2**: Falls into an **intermediate threat** level. Activities include gathering system data as well as modifying file permissions.
- **Group 3**: Poses the **highest threat** level. Activities involve downloading and running files with root permissions, potentially introducing malicious software.

Notice that about 91% of the data falls under the lowest threat level, suggesting that the majority of activities in this honeypot involve system scans. This trend was previously identified in our analysis of the honeyfarm dataset. We observed that, approximately 97% of the attacks conclude within a minute, with the predominant activity being the collection of system information through command execution. Despite the large volume of attacks in this group, it contains only 187,958 unique source IP addresses, representing about 50% of the total IP addresses involved. This relatively low number (taking into account that 91% of the data falls in that group) suggests that many attacks may be repeated or distributed across a subset of IP addresses. In contrast, the medium and high-threat level categories, each constituting 4.5% of overall attacks, have a more concentrated set of unique IP addresses—181,640 for medium-threat and 22,109 for high-threat. This concentration indicates a more targeted or deliberate strategy by attackers in these groups, with each IP address potentially associated with more impactful or sophisticated attack vectors.

To further analyze the data within each threat group, we aimed to differentiate between mass-produced and basic attacks, and those that are more tailored and sophisticated. It is expected that the lowest threat group would be associated with a higher proportion of basic attacks, while the intermediate and highest threat groups would involve more sophisticated attacks. Our analysis confirms this pattern: in the lowest threat

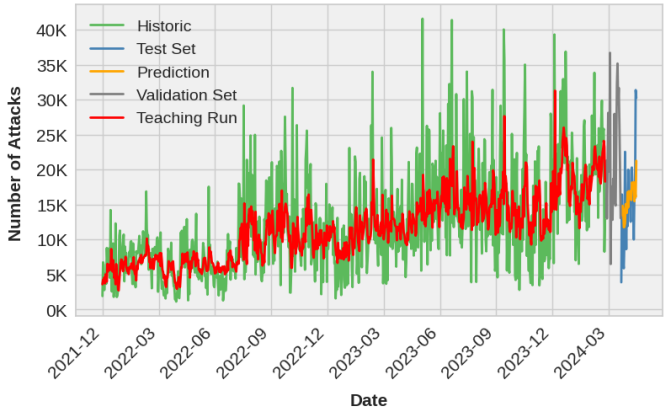

Fig. 8: **United States** time series forecasting with SVM model.

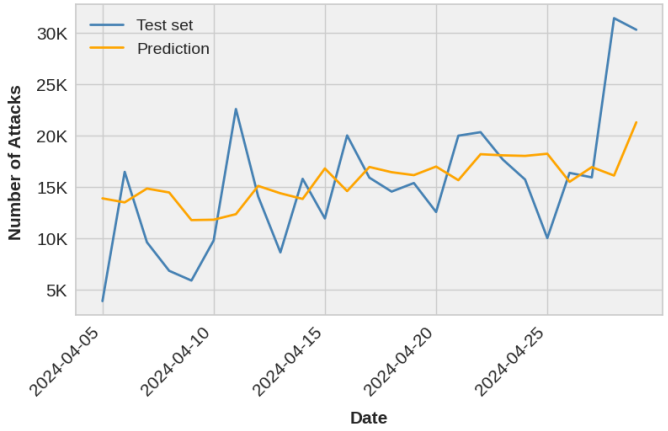

Fig. 9: Zoomed-in view of the last month for the **United States** time series forecasting using the SVM model.

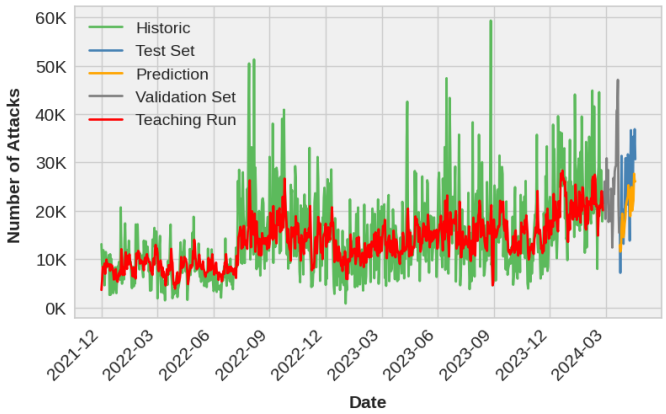

Fig. 10: **Europe** time series forecasting with SVM model.

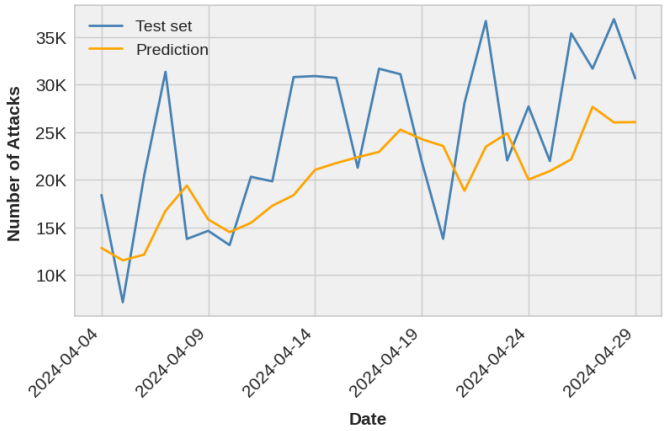

Fig. 11: Zoomed-in view of the last month for the **Europe** time series forecasting using the SVM model.

group, 69.85% of the attacks share the same procedures, indicating untargeted attacks. In contrast, only 3.25% and 9% of the attacks in the intermediate and highest threat groups, respectively, use the same procedures, suggesting the use of more innovative and targeted techniques.

B. Malware hashes

In the second cluster, and primarily in the third cluster generated by the spectral clustering algorithm, a unique identifier derived from the binary content of malware, called hash, is created. The honeyfarm provides access to the VirusTotal database's responses [11] for every hash since October 2022. Our research focuses on analyzing hashes that, at the time of the attack, were not flagged as malicious by any antivirus engine in the VirusTotal database. While hashes recognized as malicious by VirusTotal have well-established detection methods, ensuring user protection, the unrecognized hashes represent a blind spot in current cybersecurity measures, posing a potential threat that go unnoticed. By studying these hashes, we aim to uncover vulnerabilities, improve detection strategies, and enhance the overall security to better protect against emerging and sophisticated threats.

Between October 2022 and November 2023, we identified 8,593 unique file hashes generated by honeypot clients. The frequency of these hashes in our database varies widely, with only one unique hash accounting for 90% of all observations. This suggests that most of the hashes entering to the database are unique, indeed, 72.5% of the total appears only once, highlighting a potential risk of being unknown. This necessitates reliance on external resources like VirusTotal to evaluate these new hashes. Notably, only 20% of the hashes are flagged as malicious by at least one security vendor in the VirusTotal results, from now on, *recognized hashes*, leaving 80% unrecognized, from now own, *unrecognized hashes*. Analyzing these two subsets of data separately, we found that the *unrecognized hashes* were generated by 22,951 unique clients, resulting in 226,275 new records. In contrast, the *recognized hashes* were generated by 80,695 unique clients, producing 4,559,048 records. Notice that there's a 10x relationship for the *unrecognized hashes* and a 55x relationship for the *recognized hashes*. These findings imply that the *unrecognized hashes* are more targeted with respect

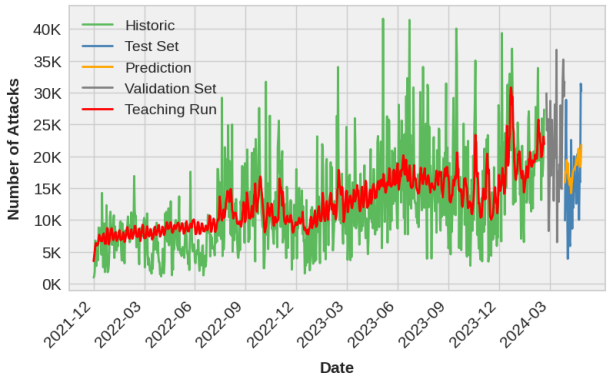

Fig. 12: **United States** time series forecasting with LSTM network.

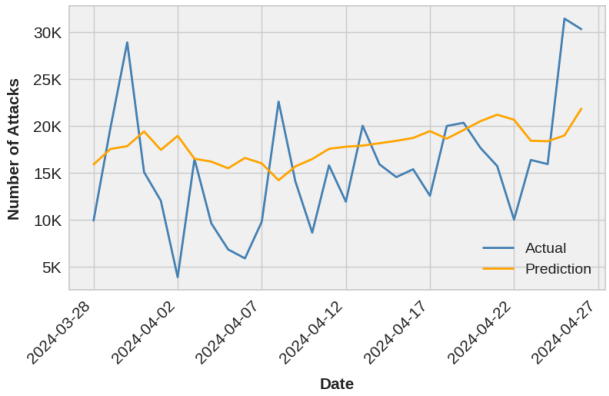

Fig. 13: Zoomed-in view of the last month for the **United States** time series forecasting using the LSTM network.

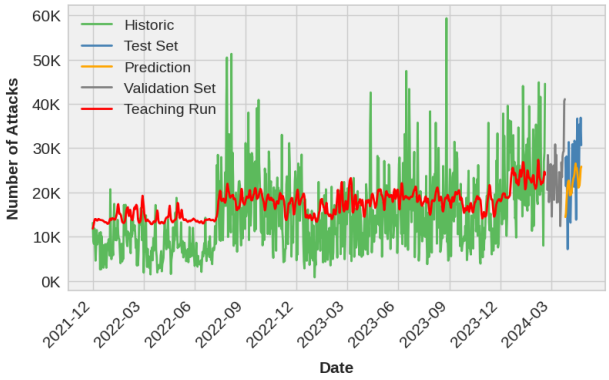

Fig. 14: **Europe** time series forecasting with LSTM network.

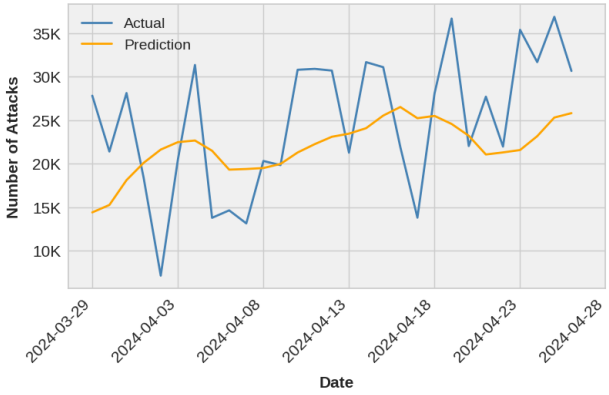

Fig. 15: Zoomed-in view of the last month for the **Europe** time series forecasting using the LSTM network.

to source IP addresses compared to the *recognized hashes*.

We observed a different pattern in the honeyfarm activity for each subset of data. Figure 5 shows the number of *recognized* and *unrecognized hashes* per honeypot, sorted by activity. The curve for *unrecognized hashes* falls smoother, whereas the curve for *recognized hashes* drops off significantly after the 80th honeypot. Precisely, beyond the 110th honeypot, the sensors capture only 6% of the *recognized hashes*, compared to the 23% of the *unrecognized hashes*. This indicates that while *recognized hashes* are concentrated in a specific number of honeypots, *unrecognized hashes* are distributed across a greater number of them, suggesting the benefit of deploying a larger honeyfarm.

Upon analyzing the behavior of 22,951 clients within the context of *unrecognized hashes*, we found that a small subset of clients were responsible for generating a significantly large number of hashes, indicating untargeted massive attacks. In contrast, the vast majority of clients submitted four or fewer hashes, implying more focused or deliberate attacks. Specifically, only 300 clients out of the total 22,951 contributed more than four hashes, while the remaining 98.7% generated four or fewer. We suggest to block these little group of high-activity IP addresses to avoid massive campaigns of *unrecognized hashes*.

Finally, to gain deeper insights into these *unrecognized hashes*, we applied the same spectral clustering technique to this specific subset of data. The results led to the formation of three distinct clusters, categorizing various attacks into three main groups, each suggesting a different pattern of behavior. The general breakdown of these groups is as follows:

1) **Group 1**: This group focuses on changing the system password to *root*, scan the system, create temporary files, execute them, and then remove these files to hide traces. This group accounts for 16.7% of the total attacks.
2) **Group 2**: These attacks focuses on downloading and executing malware-infected scripts from external sources. After executing these scripts, they take measures to cover their tracks and hide traces. This group accounts for 0.7% of the total attacks.
3) **Group 3**: This group manipulates the existing SSH configuration to add a new SSH public key, allowing unauthorized access without a password. They also gather system information, download and execute malware-infected scripts from external sources, and then clean up to hide traces. This group accounts for 82.6% of the total attacks.

Given the most frequently observed activities on the honeyform for *unrecognized hashes*, we give some security measures for each group.

1) **Group 1**: Organizations should enforce strict password policies, including the use of complex, regularly updated passwords. Implementing Multi-Factor Authentication (MFA) and disabling direct root login via SSH enhances security by requiring additional verification steps for privileged access. Continuous monitoring of system logs and file integrity can help detect unauthorized changes and swiftly respond to suspicious activities. Additionally, limiting executable permissions in critical directories and conducting regular security audits further strengthens defenses against unauthorized system alterations and potential malware execution.
2) **Group 2**: Network segmentation and robust firewall configurations can isolate critical systems from potentially compromised external networks, reducing the risk of malware infiltration. Implementing advanced endpoint protection solutions, such as Endpoint Detection and Response (EDR), that include behavior-based detection can help identify and block malicious script execution in real-time.
3) **Group 3**: Organizations should prioritize SSH hardening by disabling password-based authentication and enforcing strong, certificate-based authentication methods. Implementing centralized logging and real-time monitoring of SSH access attempts enables rapid detection and response to unauthorized access attempts. Regular audits of SSH configurations and authorized key management are crucial to identifying and revoking unauthorized access promptly.

C. Forecasting Analysis

Let us analyze the results of the forecasting analysis using three distinct approaches: Classical models, ML models, and DL models. We will assess our models by utilizing time series data from both the US and Europe, as illustrated in Figure 3 and 4.

1) Classical approach: As previously discussed, we employed ARIMA [12] and Prophet [36] methodologies to forecast the quantity of expected cyberattacks. Remember that the time series data for the US is non-stationary, whereas the time series data for the EU is stationary. Consequently, the ARIMA model parameters are set accordingly, with a differencing parameter d of 1 for the US data and 0 for the EU data. Subsequent optimization through grid search yields (p, q) values of $(2, 1)$ for the US series and $(1, 3)$ for the EU series. However, even with these adjustments, both ARIMA models generate forecasts that are uninformative and unrealistic. The predicted values remain constant, as depicted in Figures 6 and 7. Although the confidence interval includes the majority of the test set, the predictions fail to capture the underlying data patterns, highlighting the limitations of the ARIMA approach in forecasting these particular datasets.
The situation is identical with Prophet- the forecasted values persist as constants throughout the test set, resulting in uninformative and unrealistic forecasts.

Region/Model	ARIMA	Prophet	SVM	LSTM
USA	82.29	55.78	41.54	53.18
Europe	38.75	35.17	27.57	31.99

TABLE IV: Comparison of the MAPE metric across all studied approaches and regions.

Region/Model	ARIMA	Prophet	SVM	LSTM
USA	9366.07	6807.90	5878.53	6703.82
Europe	7894.05	7318.87	7830.71	7911.17

TABLE V: Comparison of the RMSE metric across all studied approaches and regions.

2) Machine Learning approach: We began by adjusting the parameters i and j, which dictate the window length of lagged and statistical features for each time series model. Specifically, for the US time series, (i, j) is configured as $(5, 5)$, whereas for the EU time series, it is set to $(4, 5)$. Several ML models were trained, and Support Vector Machines (SVM) [14] emerged as the top performer. The results generated by SVM are illustrated in Figures 8, 9, 10 and 11. Notably, although these models may not accurately predict the peaks of the series, they demonstrate proficiency in forecasting the overall trend of the series, a particularly intriguing aspect within the cybersecurity domain.

3) Deep Learning approach: As before, we identified the best parameters (i, j) to determine the window length of lagged and statistical features for each time series model. Although it might seem unnecessary to include lagged features since LSTM inherently handles them, our investigation revealed that incorporating lagged features sometimes improves model performance. For the US time series, (i, j) is set to $(0, 5)$, meaning no lagged features are added. However, for the EU time series, (i, j) is set to $(1, 4)$. By implementing the LSTM architecture mentioned in the previous chapter, we achieved comparable results to the ML approach, as shown in Figures 12, 13, 14 and 15. As the ML approach, accurately capturing the peaks of a time series remains a significant challenge.

In summary, Table VI display the parameters (i, j) used for each model and time series.

4) Model comparison: Table IV and V displays the MAPE and the RMSE metrics respectively, for each approach under study. Combining metric evaluation with visual forecasting, it becomes evident that classical models perform poorly in this task, often generating unrealistic, constant forecasts. It is important to combine both metric evaluation with visual forecasting, indeed, Table V shows Prophet as the top-performing model for Europe based on RMSE, when it actually predict a constant line averaging the number of cyberattacks. On the other hand, both ML and DL approaches

Parameter selection	SVM	LSTM
USA	(5,5)	(0,5)
Europe	(4,5)	(1,4)

TABLE VI: Parameters (i, j) used for each model and time series, where i represents the number of lagged features, and j represents the number of lagged statistical features.

demonstrate strong overall performance. Leveraging lagged and statistical features, the ML approach adeptly captures time series patterns, producing forecasts that mirror the underlying trend. Notably, forecasting the peaks of the series presents a significant challenge, with daily fluctuations of up to 15K cyberattacks, posing complexities for any predictive model. The LSTM approach achieves comparable results to the ML approach, effectively capturing the trend of the series but struggling with accurately predicting the peaks. However, when focusing solely on metrics, the ML approach slightly outperforms the LSTM approach.

Table IV demonstrates that the MAPE metric is consistently lower (and therefore better) for all models in the Europe time series compared to the USA time series. In contrast, Table V shows that, in general, the RMSE metric is higher (and therefore worse) for the Europe time series than for the USA. This difference is primarily due to the magnitude of cyberattacks. In Europe, the attack values are typically larger than in the USA. As a result, even if the absolute errors between predicted and actual values are substantial, the relative (percentage) error is smaller because of the larger actual values. A similar effect occurs with the RMSE: since the actual cyberattack values in Europe are larger, even small relative prediction errors can translate into significant absolute errors, given RMSE's sensitivity to large errors.

We strongly believe that with a larger data time window, such as 9-10 years, LSTMs would outperform ML methods. This is because LSTMs are specifically designed to handle larger sequential data, allowing them to capture long-term dependencies and complex temporal patterns more effectively. In contrast, ML methods might struggle with the increased dimensionality and require extensive feature engineering to achieve similar results.

VI. Conclusions

In this study, we analyze 29 month of honeypot data focusing only on instances where attackers execute commands, totaling over 34 million sessions. Our exploration reveals the United States, China, and India as primary sources of attacks. Notably, 97% of attacks terminate within a minute, indicating swift detection or brief reconnaissance, and, common passwords like *admin*, *password*, and *1234* underscore predictable credentials for brute-force tactics.

We further utilize cluster analysis to categorize attack instances into distinct groups, enhancing our understanding of malicious activities. Employing spectral clustering with a cosine similarity matrix derived from command embeddings via TF-IDF vectorization, we uncover three clusters reflecting different threat levels. The analysis delineates a low-threat tier primarily focused on system data gathering, an intermediate tier involving data gathering alongside file permission modifications, and a high-threat tier featuring file downloads and root permission executions. Notably, 91% of records belong to the lowest threat group, indicating that system scans dominate the activity within the honeypot environment. The attack vectors on the lowest threat groups tend to be repetitive, suggesting they are the result of mass-produced and untargeted attacks. In contrast, the high-threat group experienced more tailored and targeted attacks that were not repeated, indicating these were likely manual attacks.

We examined malware hashes and the responses provided by the VirusTotal malware database. Out of over 8,500 unique hashes analyzed, we found that 80% were unrecognized by VirusTotal, indicating a significant risk to systems. In the case of *unrecognized hashes*, the relationship between the observed hashes and source IP addresses is narrower, indicating that these attacks are more targeted than those involving *recognized hashes*. Furthermore, we noted that not all honeypots experienced the same level of activity. Indeed, beyond the 110th honeypot, the sensors capture only 6% of the *recognized hashes*, compared to the 23% of the *unrecognized hashes*. This indicates that while *recognized hashes* are concentrated in a specific number of honeypots, *unrecognized hashes* are distributed across a greater number of them, suggesting the benefit of deploying a larger honeyfarm. However, while we conclude that deploying a larger network of honeypots is beneficial, since unrecognized hashes are distributed across a greater number of them, we have not been able to determine which is the optimal location for deploying these honeypots. No discernible patterns or differences were observed in the effectiveness of honeypot deployment based on location. We leave this study for a further research.

To gain additional insights, we applied cluster analysis to the *unrecognized hashes* and identified three distinct groups, each representing a common pattern of attack when a hash enters the system:

1) **Group 1**: This group focus on changing system passwords to "root," scanning systems, and executing temporary files. In order to counter these tactics, we suggest to enforce strict password policies, including the use of complex, regularly updated passwords, as well as the implementation of Multi-Factor Authentication (MFA) and disabling root login via SSH.
2) **Group 2**: These attacks focus on downloading and executing malware-infected scripts. We suggest network segmentation and robust firewalls configuration to isolate critical systems from potentially compromised external networks. Implementing advanced endpoint protection solutions, such as Endpoint Detection and Response

(EDR), that include behavior-based detection can help identify and block malicious script execution in real-time.

3) **Group 3**: This group focus on manipulating the SSH configuration to gain unauthorized access. In this case, we recommend SSH hardening by disabling password-based authentication and enforcing strong, certificate-based authentication methods.

Concluding our study, we conduct a forecasting analysis aimed at predicting cyberattack frequencies per region. We evaluate various methodologies including Classical, ML and DL models. Results demonstrate the superior performance of ML and DL models over the Classical approach, attributed to the incorporation of lagged and statistical features. The ML approach effectively captures time series patterns, reflecting the underlying trend in its forecasts. While the LSTM approach produces comparable results to the ML approach with the 29-month data, the authors believe that with a larger data time window, such as 9-10 years, LSTMs would outperform ML methods.

None of the models successfully predicted the peaks in the series. The series exhibits daily fluctuations of up to 5K cyberattacks, making accurate prediction challenging for any of the approaches. To improve peak prediction, further refinement in feature engineering or model hyperparameter tuning may be considered.

REFERENCES

[1] L. Spitzner, "The honeynet project: Trapping the hackers," *IEEE Security & Privacy*, vol. 1, no. 2, pp. 15–23, 2003.

[2] N. Provos *et al.*, "A virtual honeypot framework." in *USENIX Security Symposium*, vol. 173, no. 2004, 2004, pp. 1–14.

[3] I. Mokube and M. Adams, "Honeypots: concepts, approaches, and challenges," in *Proceedings of the 45th annual southeast regional conference*, 2007, pp. 321–326.

[4] J. Franco, A. Aris, B. Canberk, and A. S. Uluagac, "A survey of honeypots and honeynets for internet of things, industrial internet of things, and cyber-physical systems," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2351–2383, 2021.

[5] M. Nawrocki, M. Wählisch, T. C. Schmidt, C. Keil, and J. Schönfelder, "A survey on honeypot software and data analysis," *arXiv preprint arXiv:1608.06249*, 2016.

[6] N. Kambow and L. K. Passi, "Honeypots: The need of network security," *International Journal of Computer Science and Information Technologies*, vol. 5, no. 5, pp. 6098–6101, 2014.

[7] R. B. Khoshnaw, "A comparative analysis between low and high level honeypots," in *Conference of Cihan University-Erbil on Communication Engineering and Computer Science*, 2017, p. 19.

[8] "Cowrie. cowrie on github," https://github.com/cowrie/cowrie, 2019.

[9] C. Munteanu, S. J. Saidi, O. Gasser, G. Smaragdakis, and A. Feldmann, "Fifteen months in the life of a honeyfarm," in *Proceedings of the 2023 ACM on Internet Measurement Conference*, 2023, pp. 282–296.

[10] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009.

[11] "Virustotal," https://www.virustotal.com/, 2023.

[12] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[13] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[14] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, pp. 273–297, 1995.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] E. Kheirkhah, S. M. P. Amin, H. A. Sistani, and H. Acharya, "An experimental study of ssh attacks by using honeypot decoys," *Indian Journal of Science and Technology*, pp. 5567â–5578, 2013.

[17] S. Z. Melese and P. Avadhani, "Honeypot system for attacks on ssh protocol," *International Journal of Computer Network and Information Security*, vol. 8, no. 9, p. 19, 2016.

[18] D. Fraunholz, D. Krohmer, S. D. Anton, and H. D. Schotten, "Investigation of cyber crime conducted by abusing weak or default passwords with a medium interaction honeypot," in *2017 International Conference on Cyber Security And Protection Of Digital Services (Cyber Security)*. IEEE, 2017, pp. 1–7.

[19] J. M. J. Valero, M. G. Pérez, A. H. Celdrán, and G. M. Pérez, "Identification and classification of cyber threats through ssh honeypot systems," in *Handbook of Research on Intrusion Detection Systems*. IGI global, 2020, pp. 105–129.

[20] A. Z. Tabari, X. Ou, and A. Singhal, "What are attackers after on iot devices? an approach based on a multi-phased multi-faceted iot honeypot ecosystem and data clustering," *arXiv preprint arXiv:2112.10974*, 2021.

[21] D. A. Reynolds *et al.*, "Gaussian mixture models." *Encyclopedia of biometrics*, vol. 741, no. 659-663, 2009.

[22] R. F. Engle, "Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation," *Econometrica: Journal of the econometric society*, pp. 987–1007, 1982.

[23] M. Zuzčák and P. Bujok, "Using honeynet data and a time series to predict the number of cyber attacks," *Computer Science and Information Systems*, vol. 18, no. 4, pp. 1197–1217, 2021.

[24] Z. Zhan, M. Xu, and S. Xu, "Characterizing honeypot-captured cyber attacks: Statistical framework and case study," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 11, pp. 1775–1789, 2013.

[25] Z. Yong, T. Xiaobin, and X. Hongsheng, "A novel approach to network security situation awareness based on multi-perspective analysis," in *2007 International Conference on Computational Intelligence and Security (CIS 2007)*. IEEE, 2007, pp. 768–772.

[26] D. Kwon, J. W.-K. Hong, and H. Ju, "Ddos attack forecasting system architecture using honeynet," in *2012 14th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. IEEE, 2012, pp. 1–4.

[27] Z. Zhan, M. Xu, and S. Xu, "Predicting cyber attack rates with extreme values," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 8, pp. 1666–1677, 2015.

[28] S. Siami-Namini, N. Tavakoli, and A. S. Namin, "A comparison of arima and lstm in forecasting time series," in *2018 17th IEEE international conference on machine learning and applications (ICMLA)*. IEEE, 2018, pp. 1394–1401.

[29] X. Fang, M. Xu, S. Xu, and P. Zhao, "A deep learning framework for predicting cyber attacks rates," *EURASIP Journal on Information security*, vol. 2019, pp. 1–11, 2019.

[30] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[31] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, pp. 395–416, 2007.

[32] R. Merris, "Laplacian matrices of graphs: a survey," *Linear algebra and its applications*, vol. 197, pp. 143–176, 1994.

[33] A. Aizawa, "An information-theoretic perspective of tf–idf measures," *Information Processing & Management*, vol. 39, no. 1, pp. 45–65, 2003.

[34] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[35] D. A. Dickey and W. A. Fuller, "Distribution of the estimators for autoregressive time series with a unit root," *Journal of the American statistical association*, vol. 74, no. 366a, pp. 427–431, 1979.

[36] S. J. Taylor and B. Letham, "Forecasting at scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018.

[37] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Selected papers of hirotugu akaike*. Springer, 1998, pp. 199–213.

[38] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2623–2631.

[39] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[40] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 67, no. 2, pp. 301–320, 2005.
[41] B. Lim and S. Zohren, "Time-series forecasting with deep learning: a survey," *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2194, p. 20200209, 2021.
[42] L. R. Medsker, L. Jain *et al.*, "Recurrent neural networks," *Design and Applications*, vol. 5, no. 64-67, p. 2, 2001.
[43] D. D. Lee, P. Pham, Y. Largman, and A. Ng, "Advances in neural information processing systems 22," *Tech Rep*, 2009.
[44] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "Deepar: Probabilistic forecasting with autoregressive recurrent networks," *International journal of forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020.
[45] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.
[46] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.