Explaining IoT Attacks: An Effective and Efficient Semi-Supervised Learning Framework

Giuseppe Cascavilla
TU/e, JADS, The Netherlands
g.cascavilla@tue.nl

Reinier Zwart
JADS, The Netherlands
r.b.w.zwart@tilburguniversity.edu

Damian A. Tamburri
TU/e, JADS, The Netherlands
d.a.tamburri@tue.nl

Alfredo Cuzzocrea
University of Calabria, Italy
alfredo.cuzzocrea@unical.it

2022 IEEE International Conference on Big Data (Big Data) | 978-1-6654-8045-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/BigData55660.2022.10020894

***Abstract*—Cyber-attacks targeting Internet-of-Things (IoT) devices are prevalent due to the limited security resources of the target devices and their often limited connectivity. Explaining such attacks is therefore greatly important to construct countermeasures. Current methods of automated IoT attack analysis require either large amounts of labelled data for classification, or use clustering methods which can be inaccurate. However, when a desired grouping of the data, as well as some prior knowledge about some observations in the data is available, approximate *semi-supervised* learning methods may be used to create accurate cluster arrangements. We therefore investigated the use of semi-supervised clustering approaches for creating accurate clusters of IoT attack sessions based on their goals and characteristic commonalities. We first manually created a ground-truth grouping of recent IoT attacks based on their goal. We differentiated the goal of each session according to the purpose of the used commands and the taken approach, resulting in a total of five classes. We then automatically constructed a feature set suitable for clustering similar IoT attack sessions using a method proposed in recent literature, and passed it to two different semi-supervised clustering algorithms using either labelled data (SeededKMeans) or pairwise constraints (PCKMeans) as prior knowledge. We found that both semi-supervised approaches were able to create accurate cluster arrangements using only small amounts of prior knowledge. Moreover, they outperformed an entirely unsupervised KMeans algorithm in terms of accuracy.**

***Index Terms*—IoT, Cybersecurity, Machine Learning, Clusterization, SeededKMeans, PCKMeans, Supervised Analysis, Semi-Supervised Analysis**

I. Introduction

With an increasing amount of devices being connected over network systems, the so-called Internet of Things (IoT) phenomenon is ever growing. IoT entails of clusters of intelligent systems which are able to compute and communicate information with each other, allowing more or less complex forms of swarm intelligence [1] towards many domain-specific goals, e.g., crop harvesting, security monitoring, and more. The ubiquity of the IoT allows it to be applied in many different sectors [2]. However, IoT devices often have limited computing, storage, power and communication resources, leaving little room for comprehensive security mechanisms [3]. Their limited security in combination with their constant connectivity make them a popular target for cyberattackers. Successful attacks on IoT devices can result in detrimental outcomes, such as data theft and Denial-of-Service [3], [4]. Due to IoT being increasingly interwoven in our everyday lives, surpassing over 20.4 billion connected devices in 2020 according to [5], it is generally acknowledged that cyber-attacks on IoT devices will become increasingly prevalent [6]. It is therefore crucial that such attacks and their approaches are efficiently studied, so that counter-mechanisms can be developed.

The IoT Kill Chain (IoTKC), created by [7], describes the commonly observed structure of IoT attacks. Vulnerable devices are first located in IoT attacks either through the sending of connection requests on open ports or through network scans. The targeted devices are then breached and entered through brute force or dictionary attacks. Once entered, information is gathered about the device. Preparations, such as modifications in the settings of the device, are then made for the downloading and installation of remote malware packages. A malware package is then downloaded using some file transfer protocol or simulated browser.

We can apply the IoTKC to the practical example of the Mirai botnet attacks. Mirai, which targeted IoT devices, continuously scanned the internet using its already infected devices to find other IoT devices to target. After the vulnerable devices were identified, the Mirai-infected devices breached and entered them by attempting combinations of commonly used default credentials. After entering the system, Mirai automatically checked if installation of its malware was possible on the targeted device, and proceeded with the infection steps of the IoTKC (i.e. prepare the device, download the Mirai malware, grant the necessary permissions, install the malware and remove traces). After infection, the targeted device became another part of the Mirai botnet, which was used to launch a large scale DDoS attack on DNS service provider [8].

To collect IoT attack data to analyse, we developed an honeypot. Honeypots are non-production computer systems designed for attracting cyber-attackers [3]. Honeypots deceive attackers by feigning some (if not all) of the services found on real devices, such as IoT devices. When the honeypot is compromised, it logs all kinds of information about the attack, e.g. the commands used in the attack.

IoT attack data logs are often large and therefore resource-costly to analyze manually. Both supervised and unsupervised Machine Learning (ML) approaches have therefore been proposed to automatize a part of this big data analysis process (e.g., [9], [10]). Supervised approaches rely on large amounts of labelled data for classification rarely available in practice

(e.g. [11], [12]. Conversely, unsupervised approaches offer an alternative procedure to grouping IoT attacks, requiring no labelled data. However, the accuracy of these results is unpredictable and cluster analysis is first needed to determine how each cluster is formed (e.g. [13], [14]). Moreover, any available prior knowledge about the data is disregarded in unsupervised approaches.

A different approach exists somewhere in the middle of the spectrum of supervised and unsupervised approaches: semi-supervised clustering.

Two main types of semi-supervised clustering algorithms exist: clustering algorithms using pairwise constraints and using partially labelled data [15], [16]. Clustering algorithms using pairwise constraints use knowledge about connections between pairs of observations when creating clusters. Clustering algorithms using partially labelled data make use of any known class labels of the observations. For example, if two observations share a class label, it logically follows that they should be assigned to the same cluster.

We decided to use semi-supervised clustering methods to create accurate groupings of IoT attack sessions with similar goals. This would offer a relatively cheap (i.e. in terms of resource costs) alternative to grouping using classification, since large amounts of labelled data would not be required. Moreover, the resulting cluster arrangements may prove to be more accurate than unsupervised clustering methods. Quickly and cheaply creating accurate groupings of IoT attack sessions based on their goal can be beneficial for the development of counter-mechanisms, since we can then more efficiently study similarities and dissimilarities between recent approaches of attacks with the same or different goals. We therefore pose the following main research question (**RQ**) in this paper:

RQ: *Can semi-supervised methods be used to accurately cluster IoT attack sessions according to their goals?*

To formulate an approach which can be used to provide an answer to the posed research question, we divide the question into three sub-questions. First-of-all need to identify common goals of the attacks and establish a ground-truth grouping that accurately describes the sessions and their goals. We can then compare this ground-truth grouping to the cluster labels of any used cluster models to determine accuracy. Moreover, any used semi-supervised models rely on the availability of prior knowledge, which we make available by establishing the desired grouping of the data. We therefore pose **sRQ1**:

sRQ1: *How can IoT attack sessions be grouped based on their goals?*

Second, as far as we are aware semi-supervised clustering, in the context of IoT attack analysis, has not yet been performed. Therefore, applicable semi-supervised methods first needs to be identified and properly applied. We consequently need to consider which models to use, how to obtain a suitable feature set, and how to obtain suitable subsets of prior knowledge from the ground-truth grouping we established which the models can use. We consequently pose **sRQ2**:

sRQ2: *How can we apply semi-supervised methods for clustering IoT attack sessions according to their goals?*

Last, we need to identify the effectiveness of the semi-supervised approaches. We therefore have to find the semi-supervised models which output the most accurate cluster labels when compared to the established ground-truth labels. Moreover, we need to compare their accuracy to a fully unsupervised approach, since using semi-supervised methods of unsupervised approaches only makes sense if the clustering results are more accurate. We therefore pose **sRQ3**:

sRQ3: *How accurate are the predicted labels of the semi-supervised clustering methods when compared to the class labels?*

We structure the rest of this paper as follows: first of all, we analyze the available literature on IoT attack analysis and describe the problem that follows. We then provide each of the materials and methods we used in Section III. In Section IV, we describe the results of the used clustering approaches. In Section V we discuss our findings and provide a lessons learned. Lastly in Section VII we discuss some limitations and our future works.

II. Related work

In this section, we review the work related to the analysis of (IoT) attacks to identify the current research gap in the field. We systematically analyzed the work, starting with the type of approach used, i.e. manual or automatized using machine learning. We divided the automated approaches in supervised and unsupervised approaches and describe related work using each type of approach separately. Moreover, we determined the overarching end goal for each approach, i.e. what the researchers wished to achieve. It should be noted here that this line of research has a very long tradition, even developed in the related-context of data stream analytics (e.g., [17], [18]).

A. Supervised analysis

[12] analyzed the TCP flow of attacks, i.e. the sending and receiving of packets between two systems. They labelled each attack as not-so-severe (i.e. the attacker successfully entered the system but did not execute any commands) or severe (i.e. the attacker successfully breached the device and executed at least one command). With the TCP flow as a feature set, they then used four different machine learning algorithms in order to classify the severity of the attacks: J48 Decision Tree (DT), Naive Bayes learner, Logistic Regression and Support Vector Machine. Results showed highest accuracy for the DT algorithm. In [19] authors created a system capable of classifying the threat level of attack sessions. They collected data from a Cowrie honeypot, which they labelled according to their threat level. Then used an A-priori rule generation algorithm (see [20]) to generate rules based on the association between used commands and the attack threat levels, so that certain commands would be associated with sessions with a higher or lower threat level. Using the feature set and the threat level labels, they trained four classification algorithms

to classify the threat level of the attacks. Random Forest model showed the best accuracy (0.998).

[21] mimicked a smart factory environment in a honeypot to collect botnet attack information. They used 10 features related to botnet intrusion types to classify attacks as one of 4 attack types: DDoS, DoS, reconnaissance and theft. Using Random Forest (RF) classification, 96% accuracy was achieved in the classification of these methods using the features.

B. Unsupervised analysis

In [11] authors used a clustering approach to identify features useful for classifying IoT attacks. They used two different feature sets: a feature set based on the credentials (i.e. username/password combinations) the attackers used to try and access the honeypot, and a feature set based on the commands used in the attack session. Both methods were able to generate distinct clusters and were therefore considered good indicators of different attacks. Differently, authors from [13] analyzed unique attack patterns in a IoT attack dataset collected with Cowrie. They used three dimensions based on the actions of the attacker for their analysis: depth of the interaction, behaviour of the attacker and utilisation of resources. Using these dimensions, they extracted 20 features which described the actions performed in an attack by an attacker (e.g. send connection request). They clustered the attacks using seven different clustering algorithms, three of which showed approximate distribution: Expectation Maximization (EM), KMeans and FilteredClusterer (FC). However, training a RF classifier algorithm on the clustering results showed only five features were relevant for the classification process.

In [22] authors clustered unique commands used in IoT attack sessions based on the similarity of their used commands to create clusters of similar goals. They represented each command as arrays containing counts of each word used in the command and calculated the cosine similarity between every pair of commands. The commands were then clustered using Gaussian Mixture Model (GGM) clustering algorithm.

Authors in [14] stated that most of the approaches in the IoT attack analysis literature required extensive domain knowledge for manually correlating commands to create feature sets for a clustering task. They proposed an automated method of feature construction using an autoencoder, which could automatically learn the semantic similarity between commands used in IoT attack sessions. They first collected data from a Cowrie honeypot, they then created an initial feature set consisting of binary values for the presence of a command keyword (e.g. `rm`, `wget`) in a session, as well as the total amount of commands used and the total session time. The feature set was then condensed using an autoencoder model and clustered using different algorithms, such as KMeans. By manually inspecting the results of the KMeans clusters, they were able to observe differences and similarities between the clusters of attacks, as well as the different levels of changes between the attacks in a cluster.

C. Problem statement

We found that manual analysis was often done to achieve categorizations of the IoT attack data, while automatized approaches were either used for classification (supervised) or clustering (unsupervised) of the data. The manual approaches were time-intensive due to having to manually analyze the available data. Our proposed automated supervised and unsupervised learning approaches offer a quicker alternative to the analysis of big data coming from log file of attacks perpetrated to IoT devices. However, the supervised approaches relied on large amounts of labelled data for a classification task. In practice, large amounts of labelled data might not always be available. The unsupervised approaches did not use any labelled data, which made them unpredictable in terms of accuracy, since cluster arrangements were formed entirely unsupervised. Lastly, none of the proposed automatized approaches considered situations where a desired grouping of the data is known and some prior knowledge about the data is available to be leveraged, albeit not enough for classification tasks.

III. Research methods

We propose using a semi-supervised approach for clustering IoT attack sessions with similar goals in order to achieve improved accuracy over currently used fully unsupervised approaches to IoT attack clustering. We first give an overview of the experimental configuration used for this study, after which we elaborate on each of the used methods in more detail in later sections.

A. Experimental configuration

In order to perform our study, we first collected recent IoT attack data using a Cowrie honeypot and removed sessions without command input, resulting in a dataset of 2.115 unlabelled IoT attack sessions.

To answer **sRQ1**, we needed to provide every session with a specific label based on their goal, which both established a ground-truth grouping the cluster algorithms needed to aim for, as well as a set of prior knowledge semi-supervised algorithms could borrow from. We used two methods from previous literature (i.e. [22], [23]) to manually divide the sessions in two groups based on their goal. We then inspected approach of each of the sessions assigned to both groups in order to create a more specific label set, resulting in 5 unique classes. We developed a tool for labelling the sessions according to these 5 classes, which we evaluated using inter-rater reliability, and labelled all sessions accordingly.

To answer **sRQ2**, we developed semi-supervised approaches to clustering our IoT attack data. We first automatically extracted a feature set suitable for the clustering task using an autoencoder, following a state-of-the-art procedure proposed in recent literature [14]. We initially constructed 72 features from the data based on command keywords, the total session duration and the total amount of commands used in the session. We then used an autoencoder to extract an efficient

representation of the feature set, resulting in a final feature set of 20 features.

We then selected two state-of-the-art semi-supervised clustering algorithms, which used different types of prior knowledge: a SeededKMeans algorithm [24], which used partially labelled data, and a PCKMeans algorithm [25], which used pairwise constraints.

Using the ground-truth labels we established to answer **sRQ1**, we obtained small subsets of prior knowledge for the semi-supervised algorithms to use. We randomly sampled small subsets of ground-truth labelled sessions from each class for the SeededKMeans algorithm. As advised by [25], we used an active learning scheme for finding sets of informative pairwise constraints for the PCKMeans algorithm, determining a pairwise constraint of a pair by comparing their ground-truth labels.

To answer **sRQ3**, we considered different configurations of each of the semi-supervised models to determine their accuracy at variable amounts of available prior knowledge and different hyperparameters (if relevant). Using variable amounts of prior knowledge for the models is interesting, since in practice the amount of available prior knowledge will most likely vary heavily as well, and knowing which model performs better at what level of prior knowledge can be beneficial in these situations. We determined accuracy by comparing the predicted cluster labels of the models to the ground-truth labels we established to answer **sRQ1**. We selected the Adjusted Mutual Information (AMI) metric to determine cluster accuracy, which we calculated for each model configuration. To show effectiveness of the semi-supervised models over an unsupervised approach, we ran different configurations of a KMeans approach using the same feature set and calculated the accuracy of its cluster labels.

B. Data collection

Since this study relies on recent IoT attack data and no (recent) datasets were, as far as we were aware, available online, we collected new data was for the purpose of this study. We used a Cowrie honeypot [26], which can log brute force attacks and shell interactions performed by attackers. It can simulate the services found on IoT devices [13], [14], [19] and is the honeypot which is most commonly expanded upon [3]. We therefore chose Cowrie over alternatives.

We deployed Cowrie on a virtual machine in a Microsoft Azure cloud environment. The virtual machine ran on a a Debian 10 distribution, which is required for installation of Cowrie [26]. We installed Cowrie via T-pot [27], a honeypot platform which allows for easy installation of multiple honeypots at once. Even though T-pot hosts many different honeypots in its arsenal by default, we disabled all other honeypots than Cowrie after installation to limit server costs.

We collected the IoT attack data over three time periods, for a total of 54 days: from January 6th 2022 until February 15th 2022 (10 days), from January 19th 2022 until March 22nd 2022 (32 days) and from April 8th 2022 until April 19th 2022 (12 days). Due to the limited time available for this thesis, a longer data collection period was not possible. We stored the data from each day in a `.JSON` format and retrieved the new data every day from the honeypot virtual environment so that it could be stored locally. Cowrie, by default, collects attacker IP-addresses. We assumed that the far majority (if not all) of these IP-addresses were run through a VPN-client, due to the attacker wishing to hide their identity when performing the (illegal) attack. However, in order to ensure none of the IP-addresses could somehow still be used to identify individual attackers, we removed collected IP-addresses from the data immediately.

We combined the rest of the individual `.JSON` files from each days' worth of collected honeypot attacks and cast it into a `.CSV` format using Python [28] and the Pandas module for Python [29]. The unedited dataset contained 111.106 unique sessions. Since the goal of this study was to cluster sessions containing command data, we discarded all sessions lacking command data. A quick inspection of the dataset showed disconnection after a failed username/password combination as a common explanation for early disconnection of the attacker. After discarding empty sessions, 2.118 sessions remained in the dataset.

We inspected the data once more for outliers. On average, sessions would take $\sim$ 26 seconds. We decided to removed session containing duplicated commands, a session that took over 1292 seconds, and a session containing unusual input, e.g. `Accept-Encoding: gzip` and `User-Agent: libwww-perl/6.58`. A total of 2.115 sessions remained in the final dataset after these alterations.

C. Labelling procedure

We first created a ground-truth grouping of the data to compare any created cluster arrangements to. We chose to group the attacks sessions based on their goal, since being able to quickly determine the goal of an attack session can prove useful in the development of countermeasures [22].

We chose to determine the goal of the attack sessions based on the purpose of the commands used in the session, since the purpose of the used commands can determine the goal of the session. [22], [23]. We considered two suitable approaches from the existing literature for the labelling procedure: the approach by [22] and the approach by [23]. We identified some overlap between these approaches. For example, both distinguished commands used for fingerprinting (e.g. `cat` or `uname`) and commands with which some malicious activity was performed (e.g. `curl` or `wget`). Therefore, we initially identified each session as being part of one of the following two categories:

- **Fingerprinting:** Sessions in which only fingerprinting commands are used to gain information about the system, e.g. gathering information about available services on the device.
- **Malicious Activity:** Sessions in which malicious acts are performed which can harm the system, e.g. downloading and execution of payloads or blocking existing users from accessing the device.

We manually analyzed the goal of the commands used in each attack session and assigned them a corresponding label. We display the distribution of these labels among the sessions in Figure 1. Most sessions in the dataset were initially labelled as 'Malicious Activity' sessions ($n = 1263$). In the remaining sessions only fingerprinting commands were executed, and these sessions were therefore labelled as 'Fingerprinting'. ($n = 852$). We assigned only one label per session, and considered Malicious Activity sessions more severe due to their negative consequences for the system. Therefore, in case both labels were applicable to a session, we chose to label the session as Malicious Activity.

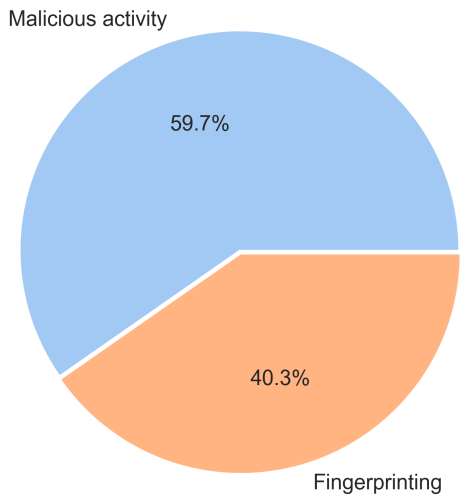

Fig. 1. Pie chart of labelled attacks in collected honeypot data.

This initial categorisation of the attack sessions was rather broad, and our aim was to create a more specific description of each of the sessions to better describe each session. We therefore developed a new labelling tool by inspected the sessions belonging to each of the two initial categories and differentiated them based on their taken approach. For a sake of space in Table I we provide an overview of the labelled we defined based on the different Iot attacks.

To assess the usefulness of the labelling tool, we used inter-rater reliability determined using Cohen's Kappa [30]. Near perfect agreement was achieved between two different raters, i.e. the researcher and a domain expert, who manually labelled the sessions using this tool ($\kappa = 0.979$). We discussed the labels on which disagreement existed, after which we achieved a final labelling of the dataset.

The final labelling of the data included 889 SSH Attack, 42 Payload Transfer, 332 Payload Attack, 465 Busybox-FP and 387 Basic-FP sessions. The distribution of the labels was rather unbalanced. We recognized it is normally good practice to address such class imbalances before using machine learning tasks. However, although class prevalence is often known when performing classification tasks due to the availability of fully labelled data, such knowledge is most likely unavailable in practical situations requiring unsupervised or semi-supervised clustering tasks, since they are performed when either no or only a small amount of labelled data is available respectively. We therefore chose not to address the imbalance in the classes for the experiment we performed in this thesis.

D. Feature construction

To prepare the IoT attack session data for the clustering models, we first extracted a feature set from which we then automatically constructed an efficient feature representation using an autoencoder model, following the approach proposed by [14]. We selected this approach since it is able to automatically construct a feature set suitable for clustering similar IoT attacks without requiring manual feature correlation, which in turn requires extensive domain knowledge and can be time-consuming. We use the following sub-sections to first describe the basics of autoencoder models, and then to elaborate on followed procedure.

1) Procedure: We first automatically extracted an initial feature set from the data using Python. We created a list of command keywords used in the sessions by inspecting the data, defining command keywords as keywords which shaped the action performed by a command, e.g. `wget` (for retrieving contents from a web server) or `cd` (for changing directories). We considered commonly accessed file directories command keywords as well. These directories (e.g. `/tmp`, `bin/busybox`) held information about which part of the system an attack targeted, which we suspected could have been another way to differentiate between the goals attack sessions. Since all commands followed Unix shell syntax (or similar), we used the `shlex` module (available in base Python) to automatically separate the keywords in each command.

We indicated the presence of a command keyword in a session with a binary value, i.e. 1 indicating the presence of a command in a session and 0 if the command was not present. We then automatically assigned the binaries for each command keyword to each individual session. Lastly, we used session duration and total amount of commands as additional features, similar to [14]. We extracted a total of 72 features. We show the most and least commonly used command keywords in the sessions in Figure 2.

Secondly, we used the Keras module [31] to construct an autoencoder model in Python, which could be used for dimensionality reduction of the original feature set. We first used K-fold cross-validation over 8 folds on the original feature set. Using this method, we allowed the test set of each of the train/test combinations to be unseen data on which we could test the autoencoder composition. Moreover, we scaled each test/train set combination using the MinMaxScaler found in the `scikit-learn` module [32]. We first fit the scaler on each train set and applied it to each train and test set individually, so that we calculated the maximum and minimum value for each input feature using only the training set and not the full dataset to prevent data leakage.

We then determined the quality of each autoencoder composition in terms of their reconstruction loss, for which we used the reconstruction Mean Squared error (rMSE). rMSE

Label	Typical commands	Main goal	Number of sessions	Description of label
SSH Attack	`rm, .ssh, mkdir`	Malicious activity	889	Sessions in which the `.ssh` folder is removed and new `.ssh` folder is added. An attack such as this will allow an attacker to open an SSH channel to the system at a later moment. All other users are locked out.
Payload Transfer	`scp`	Malicious Activity	42	Sessions in which malicious files (from a `/tmp` folder) are seemingly transferred from a remote system to the honeypot using an `scp` command.
Payload Attack	`wget, curl, busybox, chmod`	Malicious activity	332	Sessions in which some sort of malicious file is retrieved from an external location (using `wget`, `curl`, `/bin/Busybox`, or via a Payload Transfer session) (and executed).
Busybox-FP	`busybox, dd, while`	Fingerprinting	465	Sessions in which the `/bin/Busybox` directory is directly used to gather system intelligence.
Basic-FP	`uname, nproc, ps, ls`	Fingerprinting	387	Sessions in which merely information is gathered about the honeypot system by the attacker. For example, a single `echo` command is sent, most likely to see how the system responds. Another example is using the `cat` command to see if the /bin/Busybox directory exists.

TABLE I: Tool for labelling IoT attack sessions according to their goal.

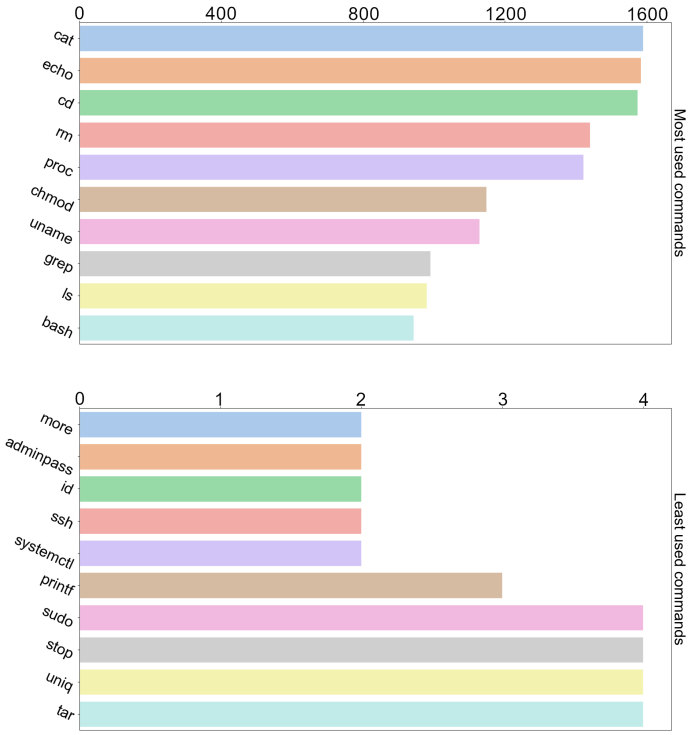

Fig. 2. Most and least used commands in the collected honeypot data.

is the average squared difference between the data in the input layer, and the reconstructed data in the output layer. It therefore serves as a good measure for assessing the ability of the autoencoder model to reconstruct the input data [14], with a lower rMSE indicating a better reconstruction. We calculated rMSE for each of the 8 train/test combinations. We trained each fold of each autoencoder composition for 1000 epochs, with a batch size of 100. We considered the following autoencoder compositions:

- **1 hidden layers:** C(15), C(20)
- **3 hidden layers:** C(40, 20, 40), C(32, 16, 32), C(20, 15, 20), C(20, 10, 20), C(20, 5, 20)
- **5 hidden layers:** C(32, 16, 8, 16, 32), C(32, 20, 10, 20, 32), C(40, 25, 15, 25, 40)

Learning curves of each of the models showed similar rMSE of the train and test set over the set amount of epochs, giving no indication of over-fitting. We display the mean rMSE of all folds for each autoencoder composition in Figure 3. As displayed, we found best mean rMSE for C(20), followed by C(40, 20, 40).

We chose the C(40, 20, 40) composition over the C(20) composition for final feature construction, since it showed close rMSE to the C(20) composition, and due to the ability of deep autoencoders to learn complex, nonlinear mappings of the data [33]. We then once more trained the C(40, 20, 40) composition autoencoder using the entire feature set, which was again scaled using the MinMaxScaler.

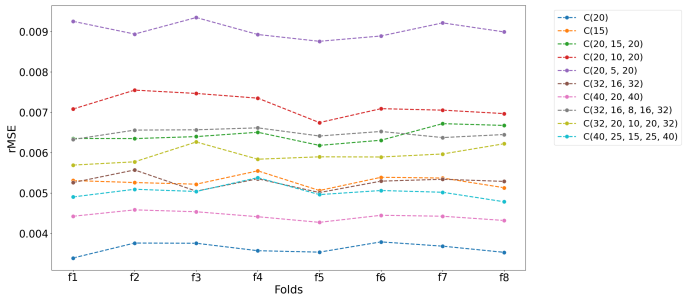

Fig. 3. Reconstruction mean squared error for proposed autoencoder compositions.

E. Selecting semi-supervised methods

We chose two semi-supervised algorithms for clustering of the IoT attack sessions: SeededKMeans [24], which uses partially labelled data, and PCKMeans, which uses pairwise constraints [25]. We chose these methods for three reasons: first, because they have been shown to outperform unsupervised algorithms in terms of accuracy [24], [25]. We therefore hoped to improve on the accuracy of the current unsupervised approaches of IoT attack analysis, consequently building on the state-of-the-art. Second, because the performance of the

models could be considered state-of-the-art in the field of semi-supervised clustering as far as we were aware [15], [16]. Third, because they utilized different types of prior knowledge to aid in the clustering task, and only one of the two types of data might be available in practice. We use the following subsections to describe how each of the two models incorporate prior knowledge.

F. Obtaining sets of prior knowledge

Each of the chosen semi-supervised models required a set of specific prior knowledge for aid in the clustering process, i.e. partially labelled data (SeededKMeans) or pairwise constraints (PCKMeans). For the SeededKMeans algorithm, no advised approaches for obtaining such information can be derived from the literature. We therefore obtained this information by randomly sampling attack sessions from each label from the labelled set created previosly.

As recommended by [25], we used an active learning scheme to obtain a set of informative pairwise constraints for the PCKMeans model. [25] proposed a 'farthest-first" active learner themselves, which was later improved by [34] in their MinMax active learner. We therefore used a MinMax active learner to find pairwise constraints for the PCKMeans model.

G. Clustering procedure

We clustered the collected IoT attack sessions using the two selected semi-supervised clustering methods: SeededKMeans and PCKMeans, to determine the accuracy of their predicted cluster labels. We implemented a SeededKMeans model and PCKMeans model in Python using the `active-semi-supervised-learning` module [35]. Moreover, we used an unsupervised KMeans approach, which we implemented using the `scikit-learn` Python module [32], to show contrast between the outcome of the unsupervised and semi-supervised approaches. We chose KMeans for the comparison, since it is commonly used as a clustering algorithm in other work concerning IoT attack analysis [11], [13], [14].

We selected a metric for accuracy of the predicted cluster labels to compare cluster arrangements of each model. We chose Adjusted Mutual Information [36]. Adjusted Mutual Information (AMI) is an adaption of the commonly used Mutual Information (MI) metric [37], [38], which measures the similarity between the predicted cluster labels to target cluster labels.

We considered different configurations of the clustering models and calculated the accuracy of each model to find the most accurate configurations. Since we wished to find 5 classes in the data according to the Table I. Moreover, we set the maximum amount of iterations for all models to 100.

For the SeededKMeans model, we used subsets of partially labelled data of sizes 50 (i.e. 10 samples per label), 100 (i.e. 20 samples per label), 150 (i.e. 30 samples per label) and 200 (i.e. 40 samples per label). This amounted to around 2%, 5%, 7%, and 9% of the total amount of data. We set the seed for the sampling procedure at 0 for reproducibility, which meant the same subset would be sampled every time the model was executed.

For the PCKMeans model, we allowed the MinMax active learner the following amounts of maximum allowed queries: 50, 100, 150, and 200. For the PCKMeans algorithm itself we used different penalty sizes for violated constraints: 0.5, 1, 1.5, and 2. We used grid search to create cluster arrangements of each maximum query/penalty size combination. We set the random seed to 0 for reproducibility, which meant the active learner would generate the same set of pairwise constraints every time the model was run. Moreover, this meant the PCKMeans algorithm would chose the same centroids for initialization every time it was run.

Lastly, for the KMeans models we used KMeans++ [39] for initialization, with the random state set to 0 for reproducibility purposes. This meant the algorithm would output the same cluster arrangement every time the model was executed. We used different numbers of initialization, namely 5, 10, 15 and 20.

IV. Cluster results

In this section, we provide the model configurations which resulted in the most accurate cluster labels when compared to the class labels using AMI. We display the highest AMI for each best model configuration in Figure 4 for comparison.

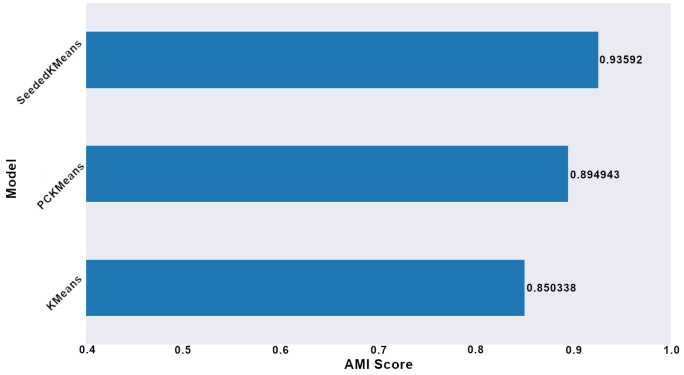

Fig. 4. AMI for best model configurations

We calculated the overall highest AMI for the SeededKMeans model cluster arrangement which used a sample size of 200 labelled sessions (i.e. 40 from each class, $\approx 9\%$ of the full dataset) for supervision (AMI≈ 0.926, see Figure 4). We display the cluster arrangement for this model in Figure 5. By investigating distribution of the IoT attack over the clusters, we found all Busybox-FP sessions assigned to cluster 0 of this cluster arrangement. Cluster 1 contained mostly SSH Attack sessions, although three Basic-FP sessions were assigned to this cluster as well. Cluster 2 almost exclusively contained Payload attack sessions, but we also found two Basic-FP sessions in this cluster. Cluster 3 consisted of mostly Basic-FP sessions, as well as twenty-three SSH Attack and twenty-nine Payload Attack sessions. Cluster 4 contained all Payload Transfer sessions.

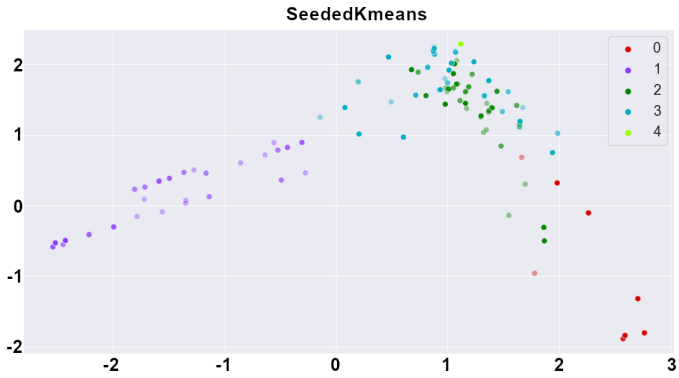

Fig. 5. Cluster arrangement of SeededKMeans clustering

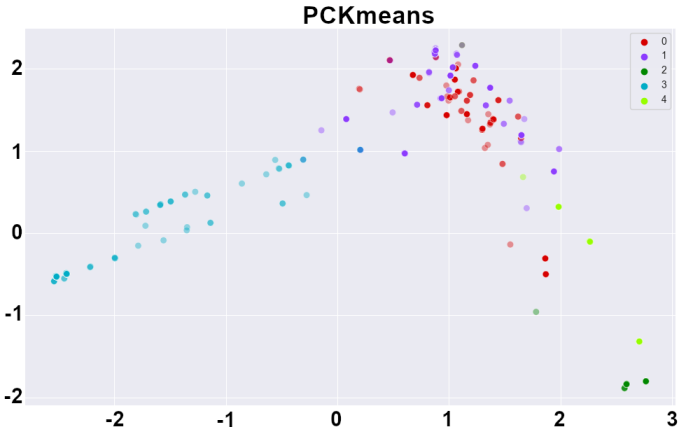

Fig. 6. Cluster arrangement of PCKMeans with MinMax active learning

The highest achieved AMI score for the PCKMeans model was found by setting the maximum amount of queries at 200 and setting the penalty for violated pairwise constraints at 2 (AMI$\approx$ 0.895, see Figure 4). We show the resulting cluster arrangement in Figure 6. By deeply analyzing the retrieved results, we see found that cluster 0 of this cluster arrangement contained most Payload Attack sessions. Cluster 1 contained most Basic-FP sessions, as well as all but one Payload Transfer sessions, twenty SSH attack and twenty-one Payload attack sessions. Cluster 2 contained most Busybox-FP sessions. We found the majority of SSH attack sessions in cluster 3, as well as two Basic-FP sessions. Lastly, cluster 4 contained mostly Busybox-FP sessions, as well as a single Payload Transfer session.

We found the highest achieved AMI score for the unsupervised KMeans model at 5 initializations of the model (AMI$\approx$ 0.850, see Figure 4). We show the created cluster arrangement in Figure 7. Cluster 0 of this arrangement mostly contained Payload attack sessions, as well as all Payload Transfer sessions and 6 Basic-FP sessions. Cluster 1 contained mostly Basic-FP sessions, as well as some SSH attack sessions and Payload Attack sessions. Cluster 2 contained most SSH

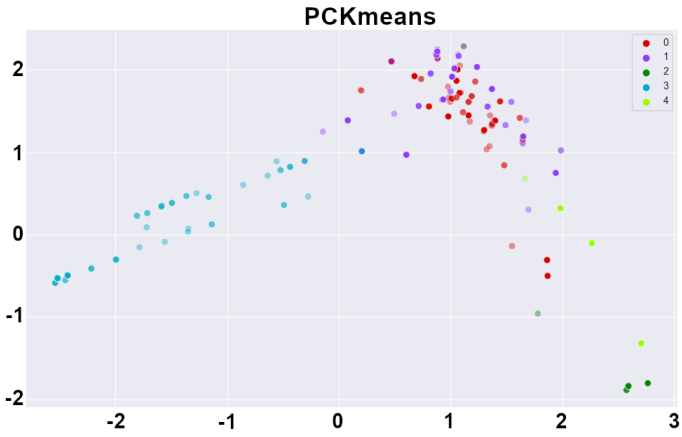

Fig. 7. Cluster arrangement of KMeans

Attack sessions, although we found three Basic-FP sessions in this cluster as well. We found most Busybox-FP sessions in cluster 3, as well as ten SSH Attack sessions. Lastly, we found a combination of some Payload Attack sessions, as well as a single Busybox-FP session and five Basic-FP sessions in cluster 4.

V. Discussion

In this section, we discuss some interesting findings of the results. First of all, both semi-supervised models were able to achieve high accuracy at different configurations. Moreover, both models were able to achieve higher accuracy than the best unsupervised KMeans model configuration, even when using lower amounts of prior knowledge. This implies the semi-supervised approaches can create accurate cluster arrangements which are able to beat the accuracy of unsupervised approaches, even if only a small amount of prior knowledge is available to be leveraged by the models.

Secondly, by inspecting the amount of observations per class per cluster, we noticed the SeededKMeans algorithm was the only model able to assign all Payload Transfer sessions to a separate cluster. We suspect other models were unable to achieve this, due to the limited amount of available examples for this class. These results imply the SeededKMeans is able create accurate cluster arrangements, even when classes are imbalanced.

Third, by inspecting the AMI of all attempted configurations of the PCKMeans model, we found the model showed high accuracy at a lower maximum amount of queries (¡ 200) when specific penalties for violating pairwise constraints were set. We found some of the PCKMeans configurations even beat the SeededKMeans configurations using comparable sizes of partially labelled data as prior knowledge. For example, at 50 queries and a penalty size of 2, PCKMeans achieved AMI$\approx$ 0.882, while SeededKMeans using a sample size of 50 labelled examples for supervision achieved AMI$\approx$ 0.844. Note that the achieved AMI of the PCKMeans model surpassed the AMI of the best unsupervised KMeans model as well

(AMI$\approx$ 0.850). This implies that, even when only a couple of oracle queries are allowed, the PCKMeans semi-supervised approach can outperform both the SeededKMeans and the unsupervised approach if combined with the right penalty for violated pairwise constraints.

Fourth, we noticed that in the SeededKMeans approach only five other sessions were wrongfully clustered. Three Basic-FP sessions were assigned to cluster 1, which contained mostly SSH Attack sessions. Moreover, 2 Basic-FP sessions were assigned to cluster 2, which contained mostly Payload attack sessions. Closer inspection of these sessions revealed that these sessions were unfinished versions of SSH Attack and Payload Attack sessions respectively. This both implies that some attack sessions were prematurely ended, and that the SeededKMeans algorithm was still able to cluster the prematurely ended attack sessions with their finished versions.

Fifth, we noticed that SSH Attack sessions were by far the most prominent in our dataset. We find most of these attacks use a similar approach. These results imply a current trend in IoT attacks concerning in which attackers target the SSH connection of the system. Moreover, it reveals a common attack pattern used for the attack.

Lastly, by manually inspecting the different attacks and their labels in the data, we found the `/bin/busybox` command was commonly used in both Busybox-FP and Payload Attack sessions. These results show the popularity of busybox exploitation for different attacker goals. This implication is even more confirmed by investigating the Basic-FP sessions found in the data, many of which checked for the existence of the `/bin/busybox` directory on the system, e.g. using the `cat` command.

Using the results from our proposed approach, we derive some design principles for the application of semi-supervised clustering techniques to future IoT attack data. We summarize these findings here:

- We advice using semi-supervised clustering for IoT attack analysis over unsupervised clustering whenever a desired grouping of the IoT attacks is known, and a small amount of prior knowledge about the attacks is or can be made available to be leveraged.
- We suggest using the SeededKMeans algorithm if possible, i.e. if a reasonable amount of explicitly labelled examples from each class is or can be made available, since it can offer the best results and since it performs well even if classes are imbalanced. Moreover, it can correctly cluster unfinished versions of attacks.
- We suggest using the PCKMeans model in combination with a MinMax active learner if a reasonable amount of explicit examples of each class is difficult to obtain for a SeededKMeans approach, since PCKMeans can achieve high accuracy even at lower amounts of prior knowledge used and the MinMax active learner will find and query informative examples automatically.
- We find the clustering algorithms can cluster together uncommon attack approaches, which may be useful for detecting zero-day attack approaches, i.e. approaches which have not yet been observed and therefore have not yet been properly studied.
- We find that many recent IoT attacks target the SSH connection to take control of the system or exploit the busybox for different malicious goals, and therefore consider those aspects of IoT devices to currently be most vulnerable.

VI. Conclusion

IoT devices are frequently targeted by cyber attacks due to their degree of connectivity in a network and due to their limited security capabilities. Efficient analysis of the aspects of these attacks is important as a first step in developing counter-mechanisms. One important aspect of IoT attacks is understanding their goal, and being able to quickly classify attacks with similar goals be beneficial for analyzing them. We therefore contributed to the existing body of work by introducing semi-supervised approaches to clustering big data IoT attacks, improving on the accuracy of state-of-the-art.

Using the semi-supervised models, we achieved more accurate cluster arrangements than an unsupervised KMeans approach, which is commonly used in related literature. We found overall best performance for the SeededKMeans model, which achieved an AMI of 0.926 while using only 200 labelled samples (i.e. 40 from each class, $\approx 9\%$ of the full dataset) as prior knowledge.

Using our findings, we show that semi-supervised approaches can be used to accurately cluster IoT attack sessions based on their goal. We provided a set of design principles that followed from our approach, which can be useful for future work wishing to use a semi-supervised approach to clustering IoT attack sessions. Moreover, we provide multiple related research directions future work can built upon.

VII. Limitations & Future Work

Although we attempted to keep the amount of limitations to a minimum, we admit the complexity of the topics addressed in this study have resulted in some key limitations. We discuss these limitations first in this section. We then identify some directions for future work.

One of the key limitations of this study is the limited diversity of sessions in the data. We noticed many sessions were very similar in terms of their approach. A high degree of similar attacks could have caused bias in the creation of the cluster arrangements, which may explain why some more uncommon attacks were clustered wrong. Furthermore, the dataset contained only a small amount of Payload Transfer sessions ($n = 42$). We chose not to address this class imbalance, since we assumed that in practical situations where semi-supervised clustering approaches are applicable, knowledge of the exact class prevalence will most likely not be available. Although we found that the SeededKMeans algorithm did not suffer under this class imbalance, it might explain why the PCKMeans and unsupervised KMeans model were unable to assign Payload Transfer sessions to a separate cluster. Another limitation is represented by scalability of main algoritmhs,

which could be tackled by considering data approximation paradigms (e.g., [40], [41]).

As future work we plan to develop a more advanced honeypot software. This is due to the fact that we suspect that some attackers were able to identify the honeypot used in this research, which can explain why the early termination of some attacks. Moreover, we plan to look into identifying the scalability of the proposed method. As the amount of data in a to-be-clustered dataset increases, so might the amount of prior knowledge needed for effective semi-supervised clustering. Future work can therefore focus on applying the semi-supervised clustering approaches proposed in this thesis to larger datasets to identify the amount of prior knowledge required in order to outperform unsupervised methods.

REFERENCES

[1] S. Garnier, J. Gautrais, and G. Theraulaz, "The biological principles of swarm intelligence," *Swarm Intelligence*, vol. 1, no. 1, pp. 3–31, June 2007. [Online]. Available: http://dx.doi.org/10.1007/s11721-007-0004-y

[2] A. E. Omolara, A. Alabdulatif, O. I. Abiodun, M. Alawida, A. Alabdulatif, H. Arshad *et al.*, "The internet of things security: A survey encompassing unexplored areas and new insights," *Computers & Security*, vol. 112, p. 102494, 2022.

[3] 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.

[4] M. Abomhara and G. M. Køien, "Cyber security and the internet of things: vulnerabilities, threats, intruders and attacks," *Journal of Cyber Security and Mobility*, pp. 65–88, 2015.

[5] M. F. Razali, M. N. Razali, F. Z. Mansor, G. Muruti, and N. Jamil, "Iot honeypot: A review from researcher's perspective," in *2018 IEEE Conference on Application, Information and Network Security (AINS)*. IEEE, 2018, pp. 93–98.

[6] M. Golling and B. Stelte, "Requirements for a future ews-cyber defence in the internet of the future," in *2011 3rd International Conference on Cyber Conflict*. IEEE, 2011, pp. 1–16.

[7] J. Haseeb, M. Mansoori, and I. Welch, "A measurement study of iot-based attacks using iot kill chain," in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2020, pp. 557–567.

[8] N. Woolf, "Ddos attack that disrupted internet was largest of its kind in history, experts say," Oct 2016. [Online]. Available: https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet

[9] K. Li, H. Jiang, L. T. Yang, and A. Cuzzocrea, Eds., *Big Data - Algorithms, Analytics, and Applications*. Chapman and Hall/CRC, 2015.

[10] A. Cuzzocrea, "Analytics over big data: Exploring the convergence of datawarehousing, OLAP and data-intensive cloud infrastructures," in *37th IEEE COMPSAC 2013, Kyoto, Japan, July 22-26, 2013*, 2013, pp. 481–483.

[11] 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)*, 2017, pp. 1–7.

[12] G. K. Sadasivam, C. Hota, and B. Anand, "Classification of ssh attacks using machine learning algorithms," in *2016 6th International Conference on IT Convergence and Security (ICITCS)*. IEEE, 2016, pp. 1–6.

[13] J. Haseeb, M. Mansoori, H. Al-Sahaf, and I. Welch, "Iot attacks: Features identification and clustering," in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2020, pp. 353–360.

[14] J. Haseeb, M. Mansoori, Y. Hirose, H. Al-Sahaf, and I. Welch, "Autoencoder-based feature construction for iot attacks clustering," *Future Generation Computer Systems*, vol. 127, pp. 487–502, 2022.

[15] E. Bair, "Semi-supervised clustering methods," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 5, no. 5, pp. 349–361, 2013.

[16] J. E. Van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, vol. 109, no. 2, pp. 373–440, 2020.

[17] A. Cuzzocrea, F. Furfaro, E. Masciari, D. Saccà, and C. Sirangelo, "Approximate query answering on sensor network data streams," *GeoSensor Networks*, vol. 49, 2004.

[18] A. Cuzzocrea, F. Furfaro, S. Greco, E. Masciari, G. M. Mazzeo, and D. Saccà, "A distributed system for answering range queries on sensor network data," in *3rd IEEE PerCom 2005 Workshops, 8-12 March 2005, Kauai Island, HI, USA*, 2005, pp. 369–373.

[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] M. Gupta, S. Kochhar, P. Jain, and P. Nagrath, "Hybrid recommender system using a-priori algorithm," in *Proceedings of International Conference on Sustainable Computing in Science, Technology and Management (SUSCOM), Amity University Rajasthan, Jaipur-India*, 2019.

[21] S. Lee, A. Abdullah, N. Jhanjhi, and S. Kok, "Classification of botnet attacks in iot smart factory using honeypot combined with machine learning," *PeerJ Computer Science*, vol. 7, p. e350, 2021.

[22] 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.

[23] E. Kheirkhah, S. P. Amin, H. J. Sistani, and H. Acharya, "An experimental study of ssh attacks by using honeypot decoys," *Indian Journal of Science and Technology*, vol. 6, no. 12, pp. 5567–5578, 2013.

[24] S. Basu, A. Banerjee, and R. Mooney, "Semi-supervised clustering by seeding," in *In Proceedings of 19th International Conference on Machine Learning (ICML-2002*. Citeseer, 2002.

[25] S. Basu, A. Banerjee, and R. J. Mooney, "Active semi-supervision for pairwise constrained clustering," in *Proceedings of the 2004 SIAM international conference on data mining*. SIAM, 2004, pp. 333–344.

[26] M. Oosterhof, "Cowrie ssh and telnet honeypot," Aug 2019. [Online]. Available: https://www.cowrie.org/

[27] T-Pot, "T-pot - the all in one multi honeypot plattform," 2016. [Online]. Available: https://github.com/telekom-security/tpotce

[28] G. Van Rossum and F. Drake, "Python 3 reference manual createspace," *Scotts Valley, CA*, 2009.

[29] W. McKinney *et al.*, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, vol. 445, no. 1. Austin, TX, 2010, pp. 51–56.

[30] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and psychological measurement*, vol. 20, no. 1, pp. 37–46, 1960.

[31] F. Chollet *et al.* (2015) Keras. [Online]. Available: https://github.com/fchollet/keras

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[33] W. Jia, M. Sun, J. Lian, and S. Hou, "Feature dimensionality reduction: a review," *Complex & Intelligent Systems*, pp. 1–31, 2022.

[34] P. K. Mallapragada, R. Jin, and A. K. Jain, "Active query selection for semi-supervised clustering," in *2008 19Th international conference on pattern recognition*. IEEE, 2008, pp. 1–4.

[35] DataMole, "Daactive semi-supervised clustering algorithms for scikit-learn," Oct 2018. [Online]. Available: https://github.com/datamole-ai/active-semi-supervised-clustering

[36] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *The Journal of Machine Learning Research*, vol. 11, pp. 2837–2854, 2010.

[37] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[38] J. Kreer, "A question of terminology," *IRE Transactions on Information Theory*, vol. 3, no. 3, pp. 208–208, 1957.

[39] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," Stanford, Tech. Rep., 2006.

[40] A. Cuzzocrea, "Overcoming limitations of approximate query answering in OLAP," in *Ninth IDEAS 2005, 25-27 July 2005, Montreal, Canada*, 2005, pp. 200–209.

[41] A. Cuzzocrea, F. Furfaro, and D. Saccà, "Hand-olap: A system for delivering OLAP services on handheld devices," in *6th ISADS 2003, 9-11 April 2003, Pisa, Italy*, 2003, pp. 80–87.