Use case definition and State of the Art analysis

Document: WP1-D1.1.00
Date: August 04, 2022

Table of content

Summary 5

1. Introduction 6
2. Use case definition 6
   - 2.1. Application visibility use case 6
   - 2.2. IoT use case 9
3. State-of-the-art 13
   - 3.1. Application visibility 14
   - 3.2. IoT device detection 21
4. References 27
5. Acronyms 34
6. Appendix A: Summary Tables of Application Detection Solutions 37
7. Appendix B: Cyber-Attacks Involving and Affecting IoT Devices 41
   - 7.1. Cyber-attacks in the IoT environment 41
   - 7.2. E-IoT communications layer threat model 42
   - 7.3. Layers of monitoring and applications 43

List of Figures

Fig 1 (a) HTML data is displayed in the online embedded devices. (b) Some appropriate websites describe this device in their search engines.....22
Fig 2 ETEI architecture. Image source: [Yin-2021] 23
Fig 3 The Architecture of Sivanathan et al. Solution. Image Source: [Sivanathan-2018] 24
Fig 4 The Architecture of Pashamokhtari et al. Solution. Image Source: [Pashamokhtari-2021] 24

List of Tables

Table 1: Applications and Categories.....8
Table 2 Summary of Second-Level Application Detection Solutions. 37
Table 3 Summary of Third-Level Application Detection Solutions. 39

Change Log

Version	Submission date	Description of changes	Affected Sections
Initial Draft	08-04-2022	First version	NA

Summary

This document defines the use cases for the ENTA (Encrypted Network Traffic Analysis) project and analyzes the state-of-art solutions associated with the use cases. For each use case, its scope, a brief summary of the present state of existing solution, and proposed solutions are described. Specifically, the first use case is about encrypted network application visibility and the second use case is about IoT device discovery and rogue IoT device detection. Next, the state-of-the-art for each use case is surveyed.

Moreover, a list of references, a list of acronyms, Appendix A summarizing existing application detection solutions in two tables and Appendix B describing cyber-attacks involving and affecting IoT devices are provided.

1. Introduction

This document defines the use cases for the ENTA (Encrypted Network Traffic Analysis) project and analyzes the state-of-art solutions associated with the use cases. For each use case, its scope, a brief summary of the present state of existing solution, and proposed solutions are described in detail in Section 2. Specifically, Section 2.1 describes the first use case which is about encrypted network application visibility and Section 2.2 describes the second use case is about IoT device discovery and rogue IoT device detection. Moreover, Section **Error! Reference source not found.** presents in more detail the state-of-the-art for each use case. That is, Section **Error! Reference source not found.** provides for that of application visibility use case and Section **Error! Reference source not found.** for that of IoT device discovery and rogue IoT device detection.

A list of references is provided in Section **Error! Reference source not found.** and a list of acronyms in Section **Error! Reference source not found.**. Finally, Appendix A (Section 6) contains two summary tables of application detection solutions and Appendix B (Section 7) describes cyber-attacks involving and affecting IoT devices.

2. Use case definition

For defining the use cases for the project, we consider the following aspects:

- Scope: here, we define the objectives we want to achieve in the use case.
- Existing solutions: here, we describe briefly current solutions for each use case, technologies they use, etc.
- Solution: here, we will define how Artificial Intelligence will be leveraged in the solution, defining the inputs and the outputs, and listing relevant KPI's

2.1. Application visibility use case

2.1.1. Scope

Increased levels of data encryption pose challenges to IT organizations that are responsible managing and securing a network. Analysts need to know the Internet applications being carried by their network. The use of encrypted traffic carrying Internet applications impedes their ability to perform network planning as well. Also, encrypted traffic poses a challenge for Law Enforcement Agencies (LEA) and Intelligence organization. Regular network probes that rely on Deep Packet Inspection are less accurate and so they fail to classify. Thus, one of the Use Case of ENTA addresses this issue by providing application-level detection and visibility of encrypted traffic.

Three possible levels for application detection can be defined:

The first level of application detection is the identification of an application as belonging to one of the following categories: video streaming, audio streaming, audio chat, text messaging, gaming, file transfer, etc. Academic research in this area is mature and a number of researchers have shown that encrypted applications can be detected and classified to various categories based on temporal and spatial traffic characteristics. For example, [Auld-2007] proposes a Bayesian trained neural network

that classifies ten categories of traffic flows, based on header-derived statistics and no port or host (IP address) information, with up to 99% accuracy for data trained and tested on the same day, and 95% accuracy for data trained and tested eight months apart.

The second level is the identification of a specific application such as being a Netflix, YouTube, DailyMotion, Spotify, WhatsApp, or Zoom application. For examples, Taylor et al profile 110 of the most popular apps in the Google Play Store and are able to re-identify them with more than 99% accuracy in [Taylor-2016] and up to 96% accuracy in [Taylor-2018]. Deep Learning based encrypted application detection have also been explored in recent years. Academic researchers have reported high accuracy of application detection. For example [Akbari-2021a] have used a combination of CNN and LSTM to illustrate that service level as well as application classification can be achieved with more than 90% accuracy.

The third level is the inference of the intent/activity occurring during an application uptime. This is a difficult problem to address, particularly due to the lack of availability of labeled dataset for model training. For examples, [Aiolli-2019] identifies user activities on smartphone-based Bitcoin wallet apps and [Yan-2018] distinguishes red packets and fund transfer activities. Ability to predict a language or specific words in a VoIP conversation can be considered. [Pathmaperuma-2022] showed that using deep learning, activities on various social media applications (e.g., Facebook, Instagram, Skype, WhatsApp and Messenger) can be identified with reasonably high accuracy. Although inferring the presence of malware or cyber-attacks also belong to the third level of application visibility, this aspect of cybersecurity visibility is out of scope of the present use case scenario.

To demonstrate the proposed ENTA solution in a realistic manner, the use case to demonstrate the Internet application detection will consider only the second (application detection) and the third level (intent/activity) of application visibilities described above.

2.1.2. Present State of Encrypted Application Detection

A detail survey of all three levels of encrypted application detection is presented in Section **Error! Reference source not found.**. A significant amount of academic research and exploration of the problem have been reported by researchers. Both Machine learning and deep learning-based solutions are presented with a high accuracy of detection capabilities. However, it is not clear if results are generic i.e., similar performance numbers can be achieved on any dataset. There are two issues: First, many of these test datasets are not publicly available and second, most authors train and test on same dataset. The first issue prevents others from reproducing and experimenting with proposed techniques. All these datasets should be publicly available. The second issue should be addressed by separating train and test datasets. Achieving high accuracy under this condition can be a challenge. In addition, very few researchers have addressed application detection in real-time (or near real-time).

2.1.3. Proposed Solution

The benefits are as follows:

- The solution will be generalized, scalable and near real-time. Models created on a training dataset will be able to classify traffic from various networks.

- Knowing the user activities with respect to applications helps IT managers understand the application usage and LEA personnel takes appropriate action.
- Although the number of applications per category may be limited, the procedure that leads to each of the above solutions will be applicable to more applications per category and to other unexplored categories.

The expected improved KPIs are:

- **F1 score**: Greater than 95%
- **Number of application traffic categories:** 10 encrypted applications
- **Speed of classification**: More than 10 Gbps

For the use case, the following categories of traffic (first level traffic from the visibility perspective) types are selected for which their second and third levels of visibility will be investigated.

1. Social Networking
2. Video Streaming
3. Audio Streaming
4. Live Streaming
5. Chat
6. Download
7. Games
8. Mail
9. Search
10. Web

For each of the traffic categories, five or more application-level (second level traffic from the visibility perspective) traffic are selected. These classifications of applications are organized as in the table below.

Table 1: Applications and Categories

Categories	Applications
Social Networking	Facebook, Instagram, Twitter, Messenger, Tiktok
Video Streaming	YouTube, Netflix, Bilibili, Dailymotion, Facebook TV, Prime Video, HBO
Audio Streaming	Spotify, SoundCloud, QQ Music, Deezer, YouTube Music, Apple Music, Amazon Music
Live Streaming	Twitch, Facebook live,
Chat	Facebook Messenger, Snapchat, WhatsApp, WeChat, GoogleChatSkype, Telegram, Viber, Discord
Download	GooglePlay, AppleDownload, SCP, SFTP, Skype FTP, WhatsApp FTP, Dropbox, GoogleDrive
Games	Minecraft, Fortnite, Grand theft Auto Online, League of Legends, Valorant

Mail	Gmail, Hotmail, Yahoomail, Outlook
Search	Google, Amazon, Yahoo!, Bing, DuckDuckGo, AoL, Baidu, Ask.com, Yandex, Ecosia
Web	Google Chrome, Mozilla Firefox, Opera Web Browser, Microsoft Edge

Identification of user activities on Social Networking will be of interest. Some of these examples are:

- Upload a video
- Watch a video
- Send/receive a voice message
- Send/receive a short text mail
- Send/receive an image on a mail
- Various actions on Facebook, Instagram, WhatsApp, Messenger
- Like/dislike an image/video/comment
- Send/receive a location info
- Send/receive contact info
- Download a file/book/image/video/web page

2.2. IoT use case

2.2.1. Scope

IoT technologies are becoming more and more popular, being implemented in more diverse scenarios and at larger scales. Devices that compose a network might be setup to cover large areas of land, positioned in hard-to-access locations or remote places.

Periodic interferences, signal loss, and general wear and tear can affect the IoT devices and this in turn will be visible in the pattern of the data packages that are being periodically sent. Depending on the application, the structure of the packages themselves might change depending on the state of the devices.

Such behavior might either be ignored, offering a possible “cover” for malicious parties who wish to infiltrate the network or be wrongfully flagged as a totally compromised system by an overzealous assessment tool.

Moreover, encrypted traffic generated by IoT devices presents traffic type and content visibility challenge. Currently, there aren’t any solution that can detect if an encrypted flow traffic is being generated by an IoT device or not, and if this encrypted traffic flow is being used to execute an attack by a rogue IoT device.

In the IoT context described above, the IoT use case will demonstrate the discovery of IoT devices and the detection of rogue IoT devices. Firstly, IoT devices will be identified in term of their presence in the network and their characteristics such as type, category, and usage.

Secondly, an IoT device will be considered rogue if it has been compromised by bad actors and is being used to attack the servers inside the company network. An IoT device can become rogue in the following manners:

- When a bad actor adds a new compromised device to the network
- When a bad actor updates an IoT device software trying to gain control over the device

Some examples of the detection processes that might indicate an IoT device is being rogue are as follows:

- **Detect if an encrypted traffic flow is being generated by an IoT device**
- **Detect if a device has been relocated outside the transmission and reception range**
- **Detect if a device has been reprogrammed while they are deployed**
- **Detect if a device has low battery level**
- **Detect if an encrypted traffic flow is being used to execute an attack**

For the purpose of the use case, AI algorithms will be trained and applied on an IoT network composed of devices that use MQTT or MQTTS over Wi-Fi, IEEE 802.15.4 and developed standards from 802.15.4 such as Zigbee. The devices will consist of:

- Waspmote boards developed by Libelium
- Meshlium gateways developed by Libelium
- Raspberry Pi boards
- Smart home devices
- Smart office devices

2.2.2. Present State of IoT Challenges

A detail survey of the current state of the art of IoT solutions with respect to the described use case is presented in Section **Error! Reference source not found.**. Some existing solutions with respect to IoT discovery are presented next followed by those with respect to rogue IoT discovery.

IoT Discovery and Identification

An IoT device can be recognized at varying levels of granularity, from a device-category to a device-type, and a specific device-instance. A device-category reflects a general grouping of similar functionality devices such as a light bulb. A device-type is a specific device model within a general device category such as light bulb and monochrome. Lastly, device-instances distinguish different devices belonging to the same device-type. Most existing solutions focus on fingerprinting the device categories or types rather than the device instances. Some of these solutions are described next:

- [Siby-2017] develops IoTScanner as a framework to passively analyze network traffic using frame header information during specific examination time windows to distinguish device types based on the observed traffic patterns. However, such an approach is prone to misclassifying identical device-types that exhibit variations in traffic patterns.
- [Apthorpe-2017] performs device type fingerprinting by analyzing metadata like IP packet headers, TCP packet headers, and send/receive rates instead of packet contents.
- [Miettinen-2017] proposes IoTSentinel, a framework for fingerprinting and identifying device-types using many packets header-based features. It provides an average accuracy rate of 50-100%. However, packet headers are prone to spoofing attacks.

- [Bezawada-2018] improves the IoTSentinel approach for profiling device categories and types using additional features like TCP window size, entropy, and payload lengths, reporting a mean identification rate of 93-99%. These approaches commonly impose high computational cost and hence delayed classification time. They require deep packet inspection functionalities or specialized hardware accelerators that render them impractical and unscalable for encrypted traffic analysis.

Rogue IoT Detection

To improve the above solutions and to detect rogue IoT devices, other approaches incorporate machine learning/deep learning (ML/DL) to handle raw data and derive latent features automatically or seamlessly integrate with feature engineering methods to predict network activities. For example, [Riyaz-2018] leverages deep convolutional neural network (CNN) approach to classify wireless devices in 2.4-GHz channels and contrast the performance with two other techniques (i.e., SVM and logistic regression).

Other existing approaches aim to monitor devices to detect rogue and malicious activities. They are broadly classified into the following three categories: specification-based, statistical-based, and anomaly-based.

Specification-based solutions, such as [Surendar-2016] and [Hamza-2020], monitor and inspect devices based on rules that characterize either benign or malicious activities. In this sense, they can be used to detect deviations from the benign profile as malicious or pinpoint patterns conforming to the specified malicious profile. Similarly, *statistical-based* solutions such as [Sehatbakhsh-2018] aim to statistically model the network and identify devices that operate under abnormal scenarios.

However, deriving distinctive profiles/features manually is a tedious task, and relying on static profiles that does not reflect the dynamic changing of activity patterns or network flow statistics cannot scale to the heterogeneity of devices in IoT environments.

Anomaly-based solutions learn normal behavior from the monitored network traffic and detect variations as anomalous activities and potential security vulnerabilities. ML/DL techniques started to receive attention for anomaly detection. For example, [Parwez-2017] employs a CNN autoencoder to model the normal behavior of the network, without considering anomalous patterns during the training, which renders it ineffective against complex attack scenarios and produces many false alarms.

Although tools for analyzing traffic such as Wireshark exist, they all require manual analysis from experts to be effective and as such, there is no generalized tool for solving the proposed issue to our knowledge. Most software focuses on establishing the security strengths of a network architecture, but these tools fall short when assessing a real-time situation.

Also, there are some tools that can be used for detecting rogue IoT devices, such as Armis or zvelo. These tools try to identify the device or monitor the IoT devices, without specifying the underlying methods.

2.2.3. Solution for this use case

We aim to explore how to secure IoT ecosystems against rogue devices by automatically identifying and discovering connected devices and pinpointing compromised devices, focusing on encrypted

network traffic. This will be achieved by leveraging ML/DL techniques to analyze passively collected traffic traces and wireless signals, available to network operators and surveillance agents.

We plan to explore the identification of commercial IoT devices that commonly appear in smart home/office environments ranging from smart cameras, speakers, doorbells, lightbulbs, and sensors, along with several non-IoT devices such as printers and laptops. We are also interested in considering a wide range of prevalent attacks that belong to two main categories: direct and reflection. The former includes ARP spoofing, TCP SYN flooding, Fraggle (UDP flooding), and Ping of Death attacks. The latter covers SNMP, SSDP, TCP SYN, and Smurf attacks.

After reviewing existing benchmark datasets, we identified three recent publicly available datasets[1] that align with our choice of IoT devices and attack vectors. The IoT traffic traces [Sivanathan-2018] and IoT IPFIX records [Pashamokhtari-2021] datasets are suitable for identifying IoT devices as they simulate benign activities, while the IoT benign and attack traces dataset [Hamza-2019] is suitable to detect rogue devices launching the attacks mentioned above.

Our solution will be capable of detecting most of the IoT devices and identifying the rogue IoT devices that use encryption, if any exists. In comparison, Armis is capable of just detecting devices in the network and check if the device is using encryption or not; and zvelo is just a monitoring tool for the devices, which include discovery, identification, and profiling. None of which can detect if a device is rogue or not when the IoT traffic is encrypted.

Also, the solution will be able to be applied to most networks as it will not be dependent on the actual information that is being transmitted but on the overall behavior of the traffic. This also means that encrypted traffic can be analyzed just as effectively as normal traffic.

AI, a key part of our solution, will be trained to extract the most important features of the input and it will label the traffic flow between generated or not by an encrypted IoT device, and if the device generating the traffic flow is rogue or not. The input will be the packets sent by IoT devices and the output will be the label generated to classify the traffic flow.

The data features will be extracted in several layers. For the first layer the inputs of the system will be the quantity, frequency, and timing of the information that is being transmitted over the network. This data will create a prediction of future behavior for a network which will act as a "profile" of that network. Changes that are not predicted will account for a "state change" of the network. This new change will be flagged and then reanalyzed with a more complex algorithm which will act as a second layer. The flagged event will be sent forward to be assessed by an administrator who can acknowledge it as a known issue or investigate further (e.g., send a team to investigate the physical device). As such, the first layer will be able to make a depth 0 decision while the second one will advance that decision to depth 1. More details on the KPIs are described next followed by the list of expected improvement in KPIs.

The KPIs we considered for this challenge are as follows:

[1] https://iotanalytics.unsw.edu.au/index

- **Relative computing power to network size and number of devices monitored.** With the development of AI and the implementation of machine learning algorithms to solve more and more problems, the computing power demand is starting to increase, draining more and more resources. Efficiency in these types of algorithms is an emerging problem.
- **Successful detection of state changes.** As different situations occur that affect IoT devices they will be reflected in the network, and we define them as "state changes" for the whole architecture.
- **Depth understanding of a state change.** A state change indicates that something happened within the network, but without knowing what that something is. We define the simple detection of the state change as depth 0, distinction between human intervention, transmission noise, or nature intervention as depth 1 and finally a more complex understanding of what the change means for the data transmitted (e.g., A malicious party has compromised a device and requests made by that device can pose a security threat) as depth 2.
- **Accuracy in depth understanding**. By this we define the correct identification of a situation at a certain depth level.
- **Relative implementation between encrypted and unencrypted data**. The proposed solution will analyze encrypted as well as non-encrypted data. The relative implementation indicates discrepancies between the 2 situations. For example, if for non-encrypted data we will have an accuracy in depth understanding of 50% and for encrypted data 25%, the relative implementation will be 0.5, but if in both scenarios the accuracy will be 50% then the relative implementation will be 1.
- **Number of automated device detection:** Automated IoT discovery is the first step towards detection of rogue IoT devices. Most of the devices should be discovered automatically. Existing solutions do not auto-discover encrypted IoT devices.
- **Accuracy in rogue device detection:** Rogue device detection accuracy is important measure for validation of approach developed in this project. The measurement will be performed on a testbed with many IoT devices and a few randomly placed rogue devices aiming for 80+% accuracy**.**

The expected improved KPIs are as follows:

- **Number of automated device detection:** Discovery of 90% of devices
- **Accuracy in rogue device detection:** Accuracy more than 80%
- **Relative computing power to network size and the number of devices monitored:** greatly decreased.
- **Successful detection of state changes:** 95%
- **Depth understanding of a state change:** 1
- **Accuracy in depth understanding:** 80%
- **Relative implementation between encrypted and unencrypted data:** 1

3. State-of-the-art

This section presents the state-of-the-art solutions with respect to the two ENTA use cases described above. Section 3.1 presents those SotA solutions with respect to the application visibility, while Section 3.2 with respect to the IoT device detection.

3.1. Application visibility

3.1.1. Introduction

Before reviewing the state-of-the art of encrypted network traffic analysis (ENTA), we first describe the key aspects of ENTA we are pursuing. Various information can be obtained in analysis encrypted network traffic. Examples of such information are network protocols used, service categories offered, applications supported, and contents exchanged. Network traffic can flow in various types of networks such as wireline, wireless, and heterogeneous networks. Here, we will be dealing mainly with protocols supporting mobile wireless applications such as Wi-Fi and associated encryption tools. In analyzing network traffic, we will be concerned with identifying specific set of applications and inferring their activities/intents. Thus, identifying a network traffic as carrying a service category traffic, which we considered as 1st-level application detection, such a video streaming, video chat, text chat or audio chat, is not our main concern. This is like identifying higher-level applications that can support multiple specification applications. Some examples these high-level applications are Facebook, Viber, WhatsApp, or Snapchat. Instead, we are interested in identifying at the level of Facebook text chat or Facebook video chat. For this type of identification, we call it the 2nd-level application detection. We also define a 3rd-level application detection. This is the identification of specific activities/intents occurring in a 2nd-level application.

Thus, in this SotA review of ENTA solutions, we are concerned with the second- and third level of mobile application detection/identification/classification. In these solutions, we are interested mainly in the Machine Learning and Deep Learning solutions proposed, explored, or evaluated, the input data characteristics extracted from the encrypted network traffic carrying the applications and the level application detection output. In some cases, the associated pre-processing steps are also described, and specific applications are listed. However, we do not describe here the data collection process, the training process, the experimental setup and performance results.

Next, we review the state-of-the-art of second- and third-level application detection solutions in Sections 3.1.2 (21 articles) and 3.1.3, (12 articles) respectively. For each of these sections, we consider the application set coverage, and the proposed solutions and their input data characteristics. We conclude the review by listing non-exhaustively some salient solution features of the reviewed articles.

3.1.2. State of the art of Second Level Application Detection Solutions

The articles with second-level of application detection solutions reviewed here are **[Aceto-2020], [Akbari-2021a, Akbari-2021b]**, *[Alan-2016]*, *[Al-Obaidy-2019]*, *[Alshammari-2011]*, **[Cui-2019]**, [DraperGil-2016], [Hajjar-2015], **[Hou-2019]**, [Khatouni-2019], *[Khatouni-2021]*, **[Lotfollahi-2020]**, *[Moore-2005]*, *[Muehlstein-2017]*, *[Papadogiannaki-2018]*, [Taylor-2016], *[Taylor-2018]*, [Wang-2015], **[Wang-2018]** and *[Zhang-2011]*

Those articles in bolt and italic are described in more detail next while all of the above articles are summarized in Table 2. Moreover, the articles in bolt use Deep Learning-based solutions and the rest use Machine Learning-based solutions.

Application Set Coverage

In these articles, the number of applications considered ranges from as few as four apps [Papadogiannaki-2018] to as many as 1595 apps [Alan-2016]. Most of them are in the tens of apps [Aceto-2020, Akbari-2021a, Akbari-2021b, Al-Obaidy-2019, Alshammari-2011, DraperGil-2016, Hajjar-2015, Hou-2019, Khatouni-2019, Khatouni-2021, Moore-2005, Muehlstein-2017, Wang-2015, Zhang-2011]. Only [Talyor-2016, Taylor2018] considers apps on the order of 100.

Most articles consider only apps. However, some consider both services/traffic and apps within services [Akbari-2021a, Akbari-2021b, Al-Obaidy-2019, Alshammari-2011, Cui-2019, DraperGil-2016, Hou-2019, Lotfollahi-2020, Papadogiannaki-2018] while [Zhang-2011] only considers traffic categories such as browsing, chatting, downloading, etc.

Solutions and Their Input Data Characteristics

Here, we review solutions and corresponding input data characteristics for the second level application detection solutions.

Most proposed solutions for the second level application detection use Machine Learning (ML) techniques. Some are just direct applications of existing ML techniques while others are some combinations of existing ML techniques. Some uses Deep Learning (DL) techniques. For the input data characteristics, most consider flow features, statistics, and time series of bidirectional traffic flows.

Next, we describe those ML-based solutions mentioned at the beginning of Section 3.1.2 and their corresponding input data characteristics.

- [Alan-2016] evaluated three existing supervised machine learning methods. The first method uses similarity measure (SM) based on the Jaccard's coefficient on traffic bursts which are groups of contiguous incoming or outgoing packets within a TCP connection and are rounded to the nearest 32 bytes. The second method uses Gaussian Naïve Bayes (GNB) classifier on packet sizes of traffic sample with negative values indicating incoming packet sizes. The third method uses Multinomial Naïve Bayes (MNB) classifier using packet sizes as in the second method where term frequency – inverse document frequency transformation and normalization are applied to feature vectors. The proposed solutions assume the launch time traffic characteristics are available, specifically, the packet sizes of apps during their launch time.

- [Al-Obaidy-2019] evaluated four ML based solution to classify social media applications. The four ML algorithms are Support Vector Machine (SVM), Naïve Bayesian (NB), C4.5, and Multi-Layer Perceptron. The inputs to all algorithms consist of 14-tuple bidirectional traffic flow features such as flow duration, number of packets, data rate, and the ratio of bytes per packet for each flow.

- [Alshammari-2011] evaluated three machine learning algorithms (AdaBoost, C4.5, and Genetic Programming (GP)) to identify Skype and SSH tunnels. The three algorithms use packet header-based features and flow-based features.

- [Khatouni-2021] evaluated two ML-based frameworks, undertook feature engineering for optimal features selection, and explored the generality of the solutions in terms of network conditions. Thirteen ML algorithms (Random Forest, Decision Tree, Complement Naïve Bayes, Multinomial Naïve Bayes, k-Nearest Neighbours, Bernoulli Naïve Bayes, Linear Support Vector Machine, Ridge Regression, Nearest Centroid, Support Vector Machine, and Linear Models with Stochastic Gradient Descent) are evaluated on NIMs2018, NIMS2019, and PRI2019 datasets. Varying number of service-based and network-based features are extracted.

- [Moore-2005] applied a Naïve Bayes estimator to classify applications in the network traffic using header-derived discriminator.

- [Muehlstein-2017] used the Support Vector Machine with Radial Basis Function as the kernel function to perform a three-tuple <OS, Browser, Application> classification based on flow features and time series.

- [Papadogiannaki-2018] used a pattern matching-based solution to identify Over-the-Top applications and services.

- [Taylor-2018] used Random Forest classifier on selected flow-based features/statistics to identify smartphone applications. Note that ambiguous flows are detected and relabeled to enhance the classifier accuracy.

- [Zhang-2011] evaluated two hierarchical ML-solutions (SVM-based and RBFN-based) to classify 7 traffic categories using flow statistics of the MAC-layer traffic such as data rate, frame size, frame interarrival time, frame size distribution, and number of frames. This solution may not be easy to scale.

Here, we describe those DL-based solutions mentioned at the beginning of Section 3.1.2 and their corresponding input data characteristics.

- [Aceto-2020] summarized a list of articles adopting DL for traffic identification and classification. The DL solutions consisting of Deep Neural Network, AutoEncoder, Stacked AutoEncoder, 1D-Convolution Neural Network, 2D-Convolutional Neural Network, and Long Short-Term Memory. The authors proposed a DL framework that is composed of instance of elementary learning layers such as dense (AutoEncoder and Deep Belief Neural Networks), convolutional (1-D and 2-D Convolutional Neural Networks), pooling (Max-pooling and Average pooling), and recurrent (Long Short-Term Memory and Gated Recurrent Unit) layers. The authors evaluated three classifiers. The first is the ML-based state-of-the-art Random Forest (Base-ML), taking as input 40 handcrafted input features, namely the best-ranked statistics (i.e., min, max, mean, standard deviation, variance, mean absolute deviation, skewness, kurtosis, and percentiles) on the basis of the Gini impurity score, calculated on the sequences of upstream, downstream, and bidirectional IP packet sizes. The other two are different DL-based implementations of their framework. For the first implement, an optimized 1D-CNN fed with the first N = 784 bytes of L4 payload, being the current DL baseline (Base-DL). For the second implementation, a Multiple Input Modalities-DL hybrid architecture using

both the recommended unbiased input sets, namely the first N = 576 bytes of L4 payload (first modality) and four informative fields[2] of the first Np = 12 packets (second modality). For the first modality, they adopt two ‘‘light” 1D-convolutional layers (16 and 32 filters and rectifier activations), each followed by a 1D max-pooling layer, and one dense layer (256 nodes). For the second modality, they use a Gated Recurrent Unit (64 nodes) and one dense layer (256 nodes). Lastly, the intermediate outputs of the two branches are stacked and fed to a shared dense layer (128 nodes).

- [Akbari-2021a, Akbari-2021b] used a multi-set input DL architecture called tripartite neural network architecture to classify traffic services and applications. While each input set is processed by a separate DL architecture, their respective outputs are processed by a common NN to determine the classification outcome. The authors extracted the TLS handshake bytes, flow time-series, and standard flow statistics as the three distinct input sets to be fed respectively to a CNN-based, LSTM-based, and a fully connected network-based architectures.

- [Hou-2019] developed the SS-Infer (Smart Spying-Infer) architecture which integrates multiple traditional neural networks. Convolutional Neural Networks is used to learn the spatial dependency features of the encoded data. The CNN output is fed to Long Short-Term Memory to learn the temporal dependency features. The combined extracted spatial-temporal features and the flow features extracted directly from the network traffic are then fed to a dense layer and a softmax layer to obtain the final classification. The flow features consist of header information and statistical information. The encoded data is the One-Hot Encoding of the first 100 bytes, middle 100 bytes and last 100 bytes of a packet data payload, considering only those payloads with 300 or more bytes.

- [Cui-2019] proposed a session-packets-based encrypted network traffic classification model using capsule neural networks (CapsNet), called SPCaps. It is basically a One-Dimensional Convolution Neural Networks (1D-CNN) feeding on processed and then subsequently encoded network traffic features (a 784 byte-matrix of 28*28 size). The initial traffic data consists of packets each with its header info, byte length, and the start time. The processed feature data considers the location of fixed strings and order between packets and is differentiated in terms of pcap-sessions and session-packets. In addition to the 1-D CNN, CNN+LSTM is also evaluated to exploit the spatial-temporal features of the encrypted data. SAE is also evaluated.

- [Lotfollahi-2017] proposed “Deep Packet” to classify traffic into service categories and into application names using Stacked Autoencoder and Convolutional Neural Network in parallel using flow IP header information and the first 1489 bytes of each IP packet.

[2] IP packet size, direction, inter-arrival time, and TCP window size (set to zero for UDP packets).

- [Wang-2018] developed three encrypted network traffic classifiers based on three DL schemes, Multi-Layer Perceptron, Stacked Autoencoder, and Convolutional Neural Networks using 1480 bytes of each raw input packet after removing Ethernet header, MAC address, frame types and other information not useful for classification and padding/truncating to the same size.

3.1.3. State of the art of Third Level Application Detection Solutions

The articles with third level of application detection solutions reviewed here are *[Aiolli-2019]*, [Brissaud-2018], *[Brissaud-2019]*, [Conti-2015], *[Conti-2016]*, *[Liu-2019]*, [Mari-2021], [Park-2016], *[Pathmaperuma-2020]*, **[Pathmaperuma-2022]**, [Wright-2010], and *[Yan-2018]*.

Those articles in bolt and italic are described in more detail next while all of the above articles are summarized in Table 3. Moreover, only [Mari-2021] and [Pathmaperuma-2022] use DL-based solutions while the rest use ML-based solutions.

Coverage of Intents and Activities

In these articles, the number of activities ranges from as few as four in one application [Yan-2018, Mari-2021] to as many as 92 in 8 applications [Pathmaperuma-2022]. Most consider social media applications [Conti-2016, Pathmaperuma-2020, Pathmaperuma-2022] while others consider very specialized applications [Aiolli-2019] (bitcoin transactions), [Brissaud-2018, Brissaud-2019] (keywords association with thumbnails), [Liu-2019] (walking direction), [Park-2016] (KakaoTalk activities), [Wright-2010] (phrases in encrypted VoIP conversations) and [Yan-2018] (fund transfers in WeChat app).

Solutions and Their Input Data Characteristics

Here, we review solutions and corresponding input data characteristics for the third level application detection solutions.

Most proposed solutions for the third level application detection use Machine Learning (ML) techniques. Some are just direct applications of existing ML techniques while others are some combinations of existing ML techniques. Some uses Deep Learning (DL) techniques. For the input data characteristics, most consider flow features, statistics, and time series of bidirectional traffic flows, while some consider thumbnail statistics [Brissaud-2018, Brissaud-2019], search keywords [Brissaud-2018, Brissaud-2019], video frame statistics [Mari-2021], and packet subsequence [Wright-2010].

Next, we describe those DL- and ML-based solutions mentioned at the beginning of Section 3.1.3 and their corresponding input data characteristics.

- [Aiolli-2019] developed a four-stage ML-based solutions using either SVM or RF at each stage to classify a flow to be a Bitcoin app or not; to classify if it is an Android app or an iOS app; to identify the specific apps; and finally, to identify the action taking place. Flows are extracted by pre-processing the input traffic and converted into time series. From the time series, statistical features are extracted to be training/test sequences. The statistics used are length of the series, minimum, maximum, mean, median, mode, variance, skewness, kurtosis, and percentile at 25%, 50% and 75%.

- [Brissaud-2019] proposed H2Classifier to detect user actions (search keywords) that has been observed in previously monitored Web service traffic (requested web content). Assuming the thumbnails' sizes are representative of a searched keyword, the authors leverage the Kernel Density Estimation (KDE) to estimate the density function of the sizes to be associated with the keyword, creating a signature for the corresponding keyword. H2Classifier is an improvement over the KDE-based solution. It leverages the random forest learning technique. One forest is used to handle one web service and to predict one class for each monitored keyword and one class for all other unknown keywords. The features used by H2Classifier are connection statistics, burst information, count of different sizes, and more.

- [Conti-2016] proposed a framework consisting of two components: the "pre-processor" and the "traffic classifier". The pre-processor filters out the relevant portion of the input traffic and converts it to a set of time series: (i) a time series is obtained by considering the bytes transported by incoming packets only; (ii) another one is obtained by considering bytes transported by outgoing packets only; (iii) a third one is obtained by combining (ordered by time) bytes transported by both incoming and outgoing packets. Similar time series are grouped into the same cluster. The traffic classifier consists of a hierarchical clustering algorithm that composes clusters as one moves up the hierarchy with distance between clusters optimized based on the total cost of an optimal warping path as the distance metric, and random forest algorithm that locates the different set of clusters associated with different user actions.

- [Liu-2019] explored six machine learning algorithms (Random Forest, Gradient Boosting Decision Tree. Decision Tree, Naïve Bayes, Logistic Regression, K-Nearest Neighbors) in their ability to identify eight basic activities of daily living from encrypted video surveillance traffic. The input feature data is obtained from the processed time-series traffic size data of cameras traffic. The initial time-series is processed to remove I frames, and the resulting time series is divided into m segments, each containing one action or no action.

- [Pathmaperuma-2020] evaluated three ML-solutions (Bayes Naïve, Random Forest, J48) in their ability to identify 51 in-app activities for three social media applications. The input data consists of only 802.11 data frames. The input traffic data is also segmented into various sizes to simulate different observation windows and opportunities. The two main features considered are frame length measured in bytes and frame inter arrival time measured in seconds. Other features consist of frame statistics and directions.

- [Pathmaperuma-2022] proposed a framework that can fully identify in-app activities based on partially observed activity-related encrypted traffic. The framework uses a CNN-based solution to convert network traffic flows information into images to identify in-app activities while also detect unknow data. Similar to [Pathmaperuma-2020], only 802.11 data frames are used, and the input traffic data is also segmented into various sizes of 1, 0.5, 0.2, and 0.1 second. Each side channel data consisting of frame length, data length, and inter arrival time forms a vector. The vectors from all side channels then form a matrix. After some normalization process such

as ensuring it is a Grayscale data of 28x28x1 (784), it becomes suitable for input to the CNN classifier.

- [Yan-2018] proposed a classifier based on Random Forest to distinguish red packet and fund transfer transactions from two other popular WeChat activities. Some key processing steps consist of time series transformation using packet length, timestamp, and TCP flag; and segmenting traffic into bursts, where a burst is defined to be a group of consecutive packets, where the inter-arrival time of two consecutive packets is within a predetermined threshold time period. The features extracted are overall statistics, packet length, number of TCP handshakes, and inbound and outbound statistics.

3.1.4. Salient Features of Reviewed Solutions

In this section, we conclude the SotA review, listing non-exhaustively some salient features of the reviewed solutions.

Instead of combining all input data features into a single set, [Akbari-2021a, Akbari-2021b] partitions them into three distinct input feature sets each of which is fed to a different DL/ML algorithm. The rational here is to exploit optimally the feature sets and enable generalization of classification objectives.

Instead of designing a novel architectural solution, [Aceto-2020] described a framework that can handle single or multiple types of input and perform single or multiple task classification such as the traffic type and the specific application generating the traffic.

Instead of identifying activities in social media applications, [Brissaud-2018, Brissaud-2019] identifies thumbnails being searched, [Liu-2019] infers activities in video traffic, [Mari-2021] infers walking directions in video traffic, [Wright-2010] detects spoken phrases in encrypted Voice over IP conversations, and [Yan-2018] infers fund transfers in WeChat.

Instead of just identifying applications, [Muehlstein-2017] also identifies operating system and browser.

Instead of using typical ML- and DL- based solutions, [Papadogiannaki-2008] uses a pattern language to describe packet trains for the purpose of fine-grained identification of application-level events in encrypted network traffic.

[Pathmaperuma-2022] proposes a DL- based solution that can differentiate between known (previously trained) and unknown (previously untrained) in-app activities and identify known in-app activity type. Moreover, their solution is also robust in handling partially captured traffic data.

[Taylor-2018] claims their solution can maintain high accuracy of identifying applications even after six months of being trained. Moreover, their solution fingerprinting capability persists to varying extents across devices and app versions.

3.2. IoT device detection

3.2.1. Introduction

The objective ENTA pursues with this use case is to be able to detect IoT devices that are connected to the corporate networks and have been compromised, causing them to be a threat for the corporate security. Also, these devices are known as rogue IoT device. To achieve this objective, we are going to analyze the encrypted network stream sent by these devices, detect if these streams are generated by IoT devices, and determine if that IoT device has been compromised.

Thus, in this SotA review of ENTA solutions we are concerned with IoT device detection/classification. In these solutions, we are interested mainly in the Machine Learning and Deep Learning proposed solutions, specifically those whose input data is encrypted; and the input data extracted characteristics is from the network data. Next, we review the state-of-the-art for automatic device detection, that is, the discovery of IoT devices and the detection of rogue IoT devices. We conclude the review by listing non-exhaustively some salient solution features of the reviewed articles. Moreover, we will describe in Appendix B (Section 7) those issues that affect the integrity of IoT devices such as cyberattacks that involves IoT devices, cyberattacks that affects E-IoT devices, and commercial IoT monitoring tools.

3.2.2. Automated Discovery of Encrypted IoT devices and Detection of Rogue IoT devices

The first part of the SotA is focused on reviewing solutions for detecting IoT devices connected to company networks and detecting if the IoT device is compromised and being used to execute attacks inside the company network.

Automated discovery of encrypted IoT devices

Automatic device discovery and annotation on a large-scale is an open problem in IoT. The Acquisitional Rule-based Engine (ARE) is proposed, which generates rules automatically to discover and annotate IoT devices without using training data. These rules are built with the help of leveraging application-layer response data originating from IoT devices, described in websites for device annotations. In the ARE framework, a transaction is a mapping between a unique response and a product description. ARE collects the transaction set by extracting relevant terms from response data such as search queries to find crawling websites. It uses the association algorithm that generates rules for IoT device annotations. So far, the types of IoT devices found were accessible IoT devices, compromised IoT devices, hundreds of thousands of cases in which IoT devices are not perfectly secured and vulnerable. For more information, see [Feng-2018].

The fingerprinting-based discovery was used for operating systems for more than two decades, using a set of input data together with a classification function to provide safety of a device in a network. The input data has a pair of interrogations and responses that IoT devices use. A classification model maps the input data and the class labels using the training data. This method requires a large training dataset. So far, no training data has been developed for IoT devices.

The Banner-grabbing Discovery method is used instead of fingerprinting for IoT device discovery, which is suited when dealing with many devices and a lack of training data. First, textual information is extracted from application-layer data to label an IoT device. [Antonakakis-2017] used the Nmap banner rules to analyze devices from CENSYS and Honeypot online.

Fig 1 (a) HTML data is displayed in the online embedded devices. (b) Some appropriate websites describe this device in their search engines.

[Meidan-2017] proposes a method to identify unauthorized types of IoT devices connected to the network based on the continuous classification of individual device traffic. To achieve this, over several months, traffic data from various IoT devices deployed in a lab is collected and tagged. To make the prediction, a model based on *Random Forest* is used. In this study, a perfect detection of unauthorized IoT devices is obtained in the test set. These encouraging results were obtained for televisions, plugs, and motion sensors, demonstrating generalizability across specific models and devices.

[Zahid-2022] proposes using HDNN to identify IoT devices that have been connected to the network, classifying network stream between being generated by an IoT device or not. To do this, a dataset of almost a million samples is generated, which, after removing the least significant features, is made up of 44 features from each sample. This architecture comprises two DNN networks*:* the first oversees identifying whether the device is an IoT device, and the second oversees identifying to which class the detected device belongs. The proposed solution can achieve an average precision of 0.9179, surpassing some models that have achieved good results, such as *Random Forest* or *Decision Tree Classifier.*

[Yin-2021] proposes creating a system responsible for extracting the essential characteristics of network traffic and learning to identify IoT devices. To do this, it uses a model formed by 1) convolutional networks in charge of feature extraction and 2) BiLSTM memory modules. This model salient feature is the use of a CNN before making the actual prediction, which allow the model to learn the spatial and temporal features extracted by the convolutional networks, without involving the researcher. For the data to be introduced into the model, it must be previously preprocessed, converting the network packets into network flows as shown in the figure below.

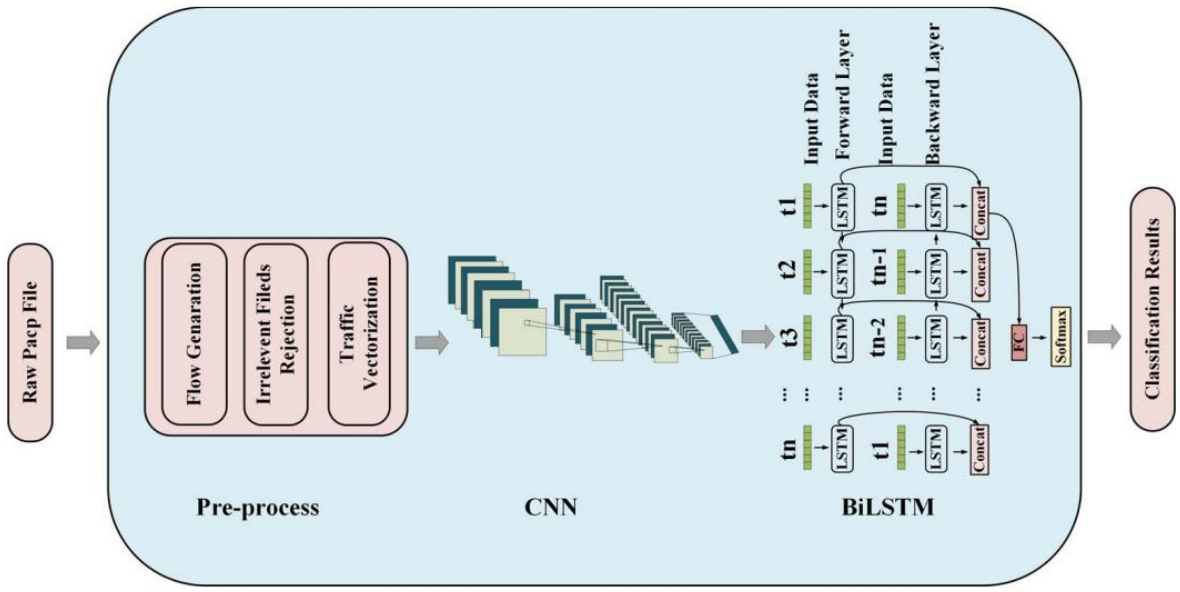

Fig 2 ETEI architecture. Image source: [Yin-2021]

[Sivanathan-2018] employs a two-stage hierarchical architecture for their classifier, as depicted in the following figure below. The authors feed multi-valued attributes to the Stage-0 classifier as a matrix representing a “bag of words”. The matrix contains M rows for the labeled instances and N columns for the unique words, where the cells contain the number of occurrences of such unique words in each instance. Their work identified 356, 421, and 54 unique words for domain names, remote port numbers, and cipher suite strings. Furthermore, they combined all corresponding words for non-IoT devices under a column named “others”. Each classifier of Stage-0 generates two outputs: a tentative class and a confidence level. These outputs, along with other single-valued quantitative attributes (i.e., flow volume, duration, rate, sleep time, DNS, NTP intervals), are fed into a Stage-1 classifier that predict the final output (i.e., the device identification with a confidence level). They employed Naive Bayes Multinomial for the Stage-0 classifier and Random Forest for the Stage-1 classifier. The authors collected a total of 50,378 labeled instances captured from different IoT and non-IoT devices generating traffic from either triggered user interactions or autonomously generated activities. The captured instances are randomly split as 70% for training and 30% for testing. The proposed solution achieved a detection accuracy of 99.88%, with a minimal value of RRSE at 5.06 %.

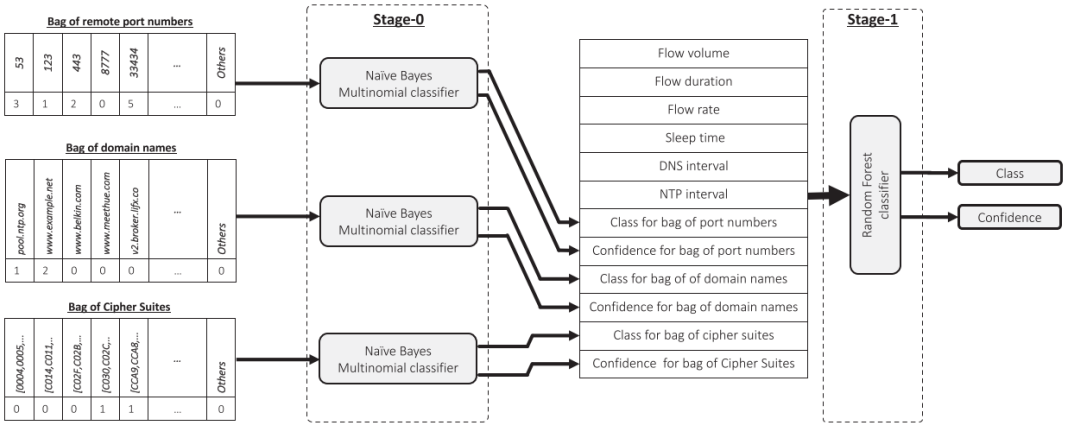

Fig 3 The Architecture of Sivanathan et al. Solution. Image Source: [Sivanathan-2018]

[Pashamokhtari-2021] takes a different direction to analyze IPFIX records, a legacy flow-based telemetry data typically collected from the edge of ISP networks. The proposed analysis was conducted on three million records emitted from a testbed of 26 IoT devices, which were parsed into 28 flow-level features to characterize the network behavior of these devices. In particular, they employed a multi-class model leveraging random forest to identify the IoT device types in home networks based on the extracted features from their post-NAT IPFIX records. They followed 10-fold cross-validation to assess the model accuracy, where the model hits 96% accuracy across all classes.

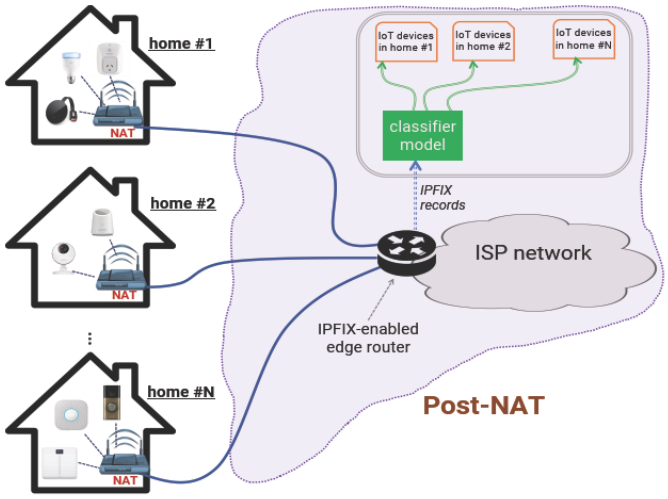

Fig 4 The Architecture of Pashamokhtari et al. Solution. Image Source: [Pashamokhtari-2021]

Detection of rogue devices in a network

Attackers cannot access a legitimate device to capture traffic and find patterns when they want to steal sensitive information.

In recent work, the authors build a system that automatically finds the type of IoT device using machine learning (ML) and limiting the communication of exposed devices for reduced damage over the network. Because the proposed protocol looks at each device's MAC addresses to find a new device that wants to authenticate with the network, there is a risk of accepting spoofed devices. Further, it

needs to be discussed the case when a previously authenticated device gets compromised and the case when an IoT device re-authenticates each time when it is trying to rejoin the network. [Bremler-2019]'s team focuses on finding differences between IoT and Non-IoT (NoT) devices by using ML classifiers to assign the relevant security policies to the IoT devices. See [Gupta-2022] for more details.

The Secure Multicontroller SDN Blockchain Model framework has recently been proposed to provide reputation and consensus mechanisms. Using the reputation method, the system monitors the rogues and adjusts their time of consumption. Upon evaluation, the detection rate of the flow rule injections was 100%. Dynamic fading factor adjustment reached the needed detection duration. See [Janani-2022] for more details.

[Hamza-2019] proposes a solution that relies on MUD specifications as an RFC standard embraced by giant manufacturers such as Google and Cisco. Despite their benefits in providing access control lists (ACLs) that allow network operators to regulate the network traffic, they cannot provide robust protection against volumetric attacks, such as ARP spoof, TCP SYN flooding, Fraggle, Ping of Death, and SSDP/SNMP/TCP/ICMP reflection. Thus, the proposed solution consists of several components: an SDN switch, a MUD engine in conjunction with an App on the SDN controller, a MUD collector, and a combination of anomaly-based and specification-based threat detectors. These components work in harmony to dynamically manage the flow-table rules inside the switch while monitoring the network activity of traffic flows for each device. This solution aims to detect if an IoT device is part of a volumetric attack and identify the traffic flows involved in the attack. This is achieved by applying machine learning techniques to learn the behavioral pattern of MUD rules at two stages: coarse-grained (per device) and fine-grained (per flow). Then, they leverage the trained model to detect attacks exhibiting anomalous deviations from the expected traffic patterns. In other words, the anomaly detection is conducted through three steps: 1) feature reduction using Principal Component Analysis (PCA); 2) clustering using X-means; and 3) outlier detection using boundary detection and Markov Chain. The solution achieved an accuracy of 89.7% of all attacks across all IoT devices.

[Ullah-2022] identifies the two basic classes of abnormal activity identification in a system or application: intrusion and compromised operation. To try to distinguish these classes, it is proposed to use FFN. To check the effectiveness of the trained model, tests are carried out with several data sets to check its ability to identify intrusions and devices that have been compromised, obtaining an average accuracy of 97% between all data sets. This data set includes data from various *botnets,* where the proposed model can identify the various *botnets* with an accuracy greater than 98%.

3.2.3. Salient features of reviewed documents

In this section, we conclude the SotA review, listing non-exhaustively some salient features of the reviewed solutions.

[Meidan-2017] proposed to use RF classifier for classifying labeled data obtained from a test set formed by smart TV, plugs and motion sensors.

[Zahid-2022] proposed to use HDNN with two different models: one for identification of IoT devices and another one for classifying the detected device. After cleansing the dataset, 44 features where selected.

[Yin-2021] presents a novel approach due to the use of CNN model to process the dataset and extract important features to be fed to the BiLSTM network. The BiLSTM network oversees identifying IoT devices receiving as an input the extracted featured by the CNN network.

[Sivanathan-2018] uses a 2-stage classifier: in the first stage using the bag of word approach to convert the data into their numeric representation using NBM; in the second stage the classification occurs using RF. Multiple unique groups were identified for the domain name, port number and cipher suite strings.

[Pashamokhtari-2021] identifies IoT devices using IPFIX records, obtained from edge ISP network. This dataset formed 28 flow level characteristics that are fed to a multi-level RF classifier.

[Ullah-2022] identifies two types of attacks: intrusion and compromised operation. Using FFN, [Ullah-2022] obtains an accuracy of 97% for all datasets increasing up to 98% when only using botnet datasets.

[Salman-2022] proposes a 4-step process to identify IoT devices and detect abnormal behavior in devices connected to the network using RF.

[Geetha-2022] identifies botnets by extracting features manually and feeding then into a model formed by two layers: a BiLSTM layer and a neuron layer in charge of classifying the traffic.

After researching the current state of art for IoT device detection and rogue IoT device detection, it is found that nowadays, there is much scientific work done for detecting IoT devices and classifying them, obtaining good results when not using encrypted communications. Also, all commercially available tools cannot detect IoT devices or determine if the devices have been compromised when using encrypted communications.

4. References

[Aceto-2020]	Aceto, Giuseppe, Domenico Ciuonzo, Antonio Montieri, and Antonio Pescapé. "Toward Effective Mobile Encrypted Traffic Classification through Deep Learning." Neurocomputing (Amsterdam) 409 (2020): 306–315.
[Aiolli-2019]	Fabio Aiolli, Mauro Conti, Ankit Gangwal, and Mirko Polato. 2019. "Mind your wallet's privacy: identifying Bitcoin wallet apps and user's actions through network traffic analysis," In Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing. ACM, 1484–1491.
[Akbari-2021a]	Iman Akbari, Mohammad A. Salahuddin, Leni Ven, Noura Limam, Raouf Boutaba, Bertrand Mathieu, Stephanie Moteau, and Stephane Tuffin., A Look Behind the Curtain: Traffic Classification in an Increasingly Encrypted Web. Proc. ACM Meas. Anal. Comput. Syst. 5, 1, Article 04 (March 2021), https://doi.org/10.1145/3447382
[Akbari-2021b]	I. Akbari, "A Look Behind the Curtain: Traffic Classification in an Increasingly Encrypted Web," Master of Mathematics in Computer Science Thesis, University of Waterloo.
[Alan-2016]	Hasan Faik Alan and Jasleen Kaur. 2016. "Can Android applications be identified using only TCP/IP headers of their launch time traffic?" In Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks. ACM, 61–66.
[Al-Obaidy-2019]	F. Al-Obaidy, S. Momtahen, M. F. Hossain and F. Mohammadi, "Encrypted Traffic Classification Based ML for Identifying Different Social Media Applications," 2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE), 2019, pp. 1-5, doi: 10.1109/CCECE.2019.8861934.
[Alshammari-2011]	Alshammari, Riyad, and A. Nur Zincir-Heywood. "Can Encrypted Traffic Be Identified Without Port Numbers, IP Addresses and Payload Inspection?" Computer networks (Amsterdam, Netherlands: 1999) 55, no. 6 (2011): 1326–1350.
[Antonakakis-2017]	Antonakakis M., April T., Bailey M., Bernhard M., Bursztein E., Cochran J., and Kallitsis M., "Understanding the mirai botnet," Paper presented at the 26th {USENIX} Security Symposium ({USENIX} Security 17), (2017)
[Apthorpe-2017]	N. Apthorpe, D. Reisman, and N. Feamster, "A Smart Home is No Castle: Privacy Vulnerabilities of Encrypted IoT Traffic", Tech. Report, ArXiv, [Online]. Available: https://arxiv.org/abs/1705.06805.
[Auld-2007]	T. Auld, A. W. Moore and S. F. Gull, "Bayesian Neural Networks for Internet Traffic Classification," in IEEE Transactions on Neural Networks, vol. 18, no. 1, pp. 223-239, Jan. 2007, doi: 10.1109/TNN.2006.883010.

[Bezawada-2018]	B. Bezawada, M. Bachani, J. Peterson, H. Shirazi, I. Ray and I. Ray, “Behavioral Fingerprinting of IoT Devices”, in ASHES ’18: Proc. of the 2018 Workshop on Attacks and Solutions in Hardware Security, 2018, pp. 41-50.
[Bremler-2019]	Anat Bremler-Barr, Haim Levy, and Zohar Yakhini, “IoT or NoT: Identifying IoT Devices in a Short Time Scale,” arXiv:1910.05647v1 [cs.NI] 12 Oct 2019.
[Brissaud-2018]	P. Brissaud, J. François, I. Chrisment, T. Cholez and O. Bettan, "Passive Monitoring of HTTPS Service Use," 2018 14th International Conference on Network and Service Management (CNSM), 2018, pp. 219-225.
[Brissaud-2019]	P. -O. Brissaud, J. Francçis, I. Chrisment, T. Cholez and O. Bettan, "Transparent and Service-Agnostic Monitoring of Encrypted Web Traffic," in IEEE Transactions on Network and Service Management, vol. 16, no. 3, pp. 842-856, Sept. 2019, doi: 10.1109/TNSM.2019.2933155.
[Conti-2015]	Mauro Conti, Luigi V. Mancini, Riccardo Spolaor, and Nino Vincenzo Verde. 2015. “Can't You Hear Me Knocking: Identification of User Actions on Android Apps via Traffic Analysis,” In Proceedings of the 5th ACM Conference on Data and Application Security and Privacy (CODASPY '15). Association for Computing Machinery, New York, NY, USA, 297–304. https://doi-org.proxy.library.carleton.ca/10.1145/2699026.2699119
[Conti-2016]	M. Conti, L. V. Mancini, R. Spolaor and N. V. Verde, "Analyzing Android Encrypted Network Traffic to Identify User Actions," in IEEE Transactions on Information Forensics and Security, vol. 11, no. 1, pp. 114-125, Jan. 2016, doi: 10.1109/TIFS.2015.2478741.
[Cui-2019]	S. Cui, B. Jiang, Z. Cai, Z. Lu, S. Liu, and J. Liu, "A Session-Packets-Based Encrypted Traffic Classification Using Capsule Neural Networks," 2019 IEEE 21st International Conference on High-Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Zhangjiajie, China, 2019, pp. 429-436.
[DraperGil-2016]	Draper-Gil, G., Lashkari, A., Mamun, M. and Ghorbani, A. “Characterization of Encrypted and VPN Traffic using Time-related Features,” In Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP 2016), pages 407-414.
[Feng-2018]	Feng, Xuan, et al. "Acquisitional Rule-based Engine for Discovering {Internet-of-Things} Devices." 27th USENIX Security Symposium (USENIX Security 18). 2018.
[Geetha-2022]	Geetha, K., and SH Brahmananda. "Network traffic analysis through deep learning for detection of an army of bots in health IoT network." International Journal of Pervasive Computing and Communications (2022).

[Google-2022]	Security Whitepapers - Data Loss Prevention - Preventing Data Exfiltration, https://cloud.google.com/docs/security/data-loss-prevention/preventing-data-exfiltration
[Gupta-2022]	Gupta, Kaustubh. "Machine Learning-Based Device Type Classification for IoT Device Re-and Continuous Authentication." (2022).
[Hajjar-2015]	Amjad Hajjar, Jawad Khalife, Jesús Díaz-Verdejo, "Network traffic application identification based on message size analysis," Journal of Network and Computer Applications, Volume 58, 2015, Pages 130-143, ISSN 1084-8045, https://doi.org/10.1016/j.jnca.2015.10.003. (https://www.sciencedirect.com/science/article/pii/S1084804515002167)
[Hamza-2019]	A. Hamza, H. H. Gharakheili, T. Benson, V. Sivaraman, "Detecting Volumetric Attacks on IoT Devices via SDN-Based Monitoring of MUD Activity", *ACM SOSR*, USA, Apr 2019.
[Hamza-2020]	A. Hamza, D. Ranathunga, H. H. Gharakheili, T. A. Benson, M. Roughan, and V. Sivaraman, "Verifying and monitoring IoTs network behavior using MUD profiles," IEEE Trans. Dependable Secure Comput., May 27, 2020
[Hou-2019]	T. Hou, T. Wang, Z. Lu and Y. Liu, "Smart Spying via Deep Learning: Inferring Your Activities from Encrypted Wireless Traffic," 2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP), 2019, pp. 1-5, doi: 10.1109/GlobalSIP45357.2019.8969428.
[Janani-2022]	Janani, K., and S. Ramamoorthy. "A Secure Multicontroller SDN Blockchain Model for IoT Infrastructure." Cyber Security, Privacy and Networking. Springer, Singapore, 2022. 321-338.
[Khatouni-2019]	A. S. Khatouni and N. Zincir-Heywood, "Integrating Machine Learning with Off-the-Shelf Traffic Flow Features for HTTP/HTTPS Traffic Classification," 2019 IEEE Symposium on Computers and Communications (ISCC), 2019, pp. 1-7, doi: 10.1109/ISCC47284.2019.8969578.
[Khatouni-2021]	Safari Khatouni, Ali, Nabil Seddigh, Biswajit Nandy, and Nur Zincir-Heywood. "Machine Learning Based Classification Accuracy of Encrypted Service Channels: Analysis of Various Factors." Journal of network and systems management 29, no. 1 (2021).
[Lashkari-2009]	Arash Habibi Lashkari, Mir Mohammad Seyed Danesh and B. Samadi, "A survey on wireless security protocols (WEP, WPA and WPA2/802.11i)," 2009 2nd IEEE International Conference on Computer Science and Information Technology, 2009, pp. 48-52, doi: 10.1109/ICCSIT.2009.5234856.
[Liu-2019]	X. Liu, J. Wang, Y. Yang, Z. Cao, G. Xiong and W. Xia, "Inferring Behaviors via Encrypted Video Surveillance Traffic by Machine Learning," 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th

	International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2019, pp. 273-280, doi: 10.1109/HPCC/SmartCity/DSS.2019.00051.
[Lotfollahi-2020]	Lotfollahi, M., Jafari Siavoshani, M., Shirali Hossein Zade, R. et al. Deep packet: a novel approach for encrypted traffic classification using deep learning. Soft Comput 24, 1999–2012 (2020). https://doi.org/10.1007/s00500-019-04030-2
[Mari-2021]	D. Mari et al., "Looking Through Walls: Inferring Scenes from Video-Surveillance Encrypted Traffic," ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2021, pp. 2595-2599, doi: 10.1109/ICASSP39728.2021.9414391.
[Meidan-2017]	Meidan, Yair, et al. "ProfilIoT: a machine learning approach for IoT device identification based on network traffic analysis." Proceedings of the symposium on applied computing. 2017.
[Miettinen-2017]	M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A. R. Sadeghi, and S. Tarkoma, "IoT Sentinel: Automated device-type Identification for Security Enforcement in IoT", *Proc. of 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, IEEE, 2017.
[Moore-2005]	Andrew W. Moore and Denis Zuev. 2005. Internet traffic classification using bayesian analysis techniques. SIGMETRICS Perform. Eval. Rev. 33, 1 (June 2005), 50–60. https://doi.org/10.1145/1071690.1064220
[Muehlstein-2017]	Muehlstein, J., Zion, Y., Bahumi, M., Kirshenboim, I., Dubin, R., Dvir, A. and Pele, "Analyzing HTTPS encrypted traffic to identify user's operating system, browser and application," In 14th IEEE Annu. Conf. Consum. Commun. Netw., pp. 1-6., 2017.
[Papadogiannaki-2018]	Papadogiannaki, Eva, Constantinos Halevidis, Periklis Akritidis, and Lazaros Koromilas. "OTTer: A Scalable High-Resolution Encrypted Traffic Identification Engine." In Research in Attacks, Intrusions, and Defenses, 315–334. Cham: Springer International Publishing, 2018.
[Papadogiannaki-2021]	Papadogiannaki, Eva, and Sotiris Ioannidis. "A survey on encrypted network traffic analysis applications, techniques, and countermeasures." ACM Computing Surveys (CSUR) 54.6 (2021): 1-35.
[Park-2016]	Park, Kyungwon, and Hyoungshick Kim. "Encryption Is Not Enough: Inferring User Activities on KakaoTalk with Traffic Analysis." In Information Security Applications, 254–265. Cham: Springer International Publishing, Springer Computer Science eBooks 2016 English/International Scholars Portal Books: Springer-Computer Science 2016
[Parwez-2017]	M. S. Parwez, D. B. Rawat, and M. Garuba, "Big data analytics for user-activity analysis and user-anomaly detection in mobile wireless network," IEEE Trans. Ind. Informat. , vol. 13, no. 4, pp. 2058–2065, Aug. 2017.

[Pashamokhtari-2021]	A. Pashamokhtari, N. Okui, Y. Miyake, M. Nakahara, H. H. Gharakheili, "Inferring Connected IoT Devices from IPFIX Records in Residential ISP Networks", *IEEE LCN*, Virtual Conference, Oct 2021.
[Pathmaperuma-2020]	Pathmaperuma, Madushi H.; Rahulamathavan, Yogachandran; Dogan, Safak; Kondoz, Ahmet (2020): In-app activity recognition from Wi-Fi encrypted traffic. Loughborough University. Conference contribution. https://hdl.handle.net/2134/11366099.v1
[Pathmaperuma-2022]	M. H. Pathmaperuma et al, "CNN for User Activity Detection Using Encrypted In-App Mobile Data," Future Internet, vol. 14, (2), pp. 67, 2022. DOI: http://dx.doi.org/10.3390/fi14020067
[Puche-2019]	Luis Puche Rondon, Leonardo Babun, Kemal Akkaya, and A. Selcuk Uluagac. 2019. HDMI-walk: attacking HDMI distribution networks via consumer electronic control protocol. In Proceedings of the 35th Annual Computer Security Applications Conference (ACSAC '19). Association for Computing Machinery, New York, NY, USA, 650–659. https://doi.org/10.1145/3359789.3359841
[Riyaz-2018]	S. Riyaz, K. Sankhe, S. Ioannidis, and K. Chowdhury, “Deep learning convolutional neural networks for radio identification,” IEEE Commun. Mag., vol. 56, no. 9, pp. 146–152, Sep. 2018.
[Salman-2022]	Salman, Ola, et al. "A machine learning based framework for IoT device identification and abnormal traffic detection." Transactions on Emerging Telecommunications Technologies 33.3 (2022): e3743.
[Sehatbakhsh-2018]	N. Sehatbakhsh, M. Alam, A. Nazari, A. Zajic, and M. Prvulovic, “Syndrome: Spectral analysis for anomaly detection on medical IoT and embedded devices,” in Proc. IEEE Int. Symp. Hardw. Orient. Security Trust (HOST), Washington, DC, USA, pp. 1–8, 2018.
[Siby-2017]	Sandra Siby, Rajib Ranjan Maiti, and Nils Tippenhauer. 2017. IoTScanner: Detecting and Classifying Privacy Threats in IoT Neighborhoods. arXiv preprint arXiv:1701.05007, 2017.
[Sivanathan-2018]	A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath and V. Sivaraman, "Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics", *IEEE Transactions on Mobile Computing*, August 2018.
[Surendar-2016]	M. Surendar and A. Umamakeswari, “InDReS: An Intrusion Detection and Response System for Internet of Things with 6LoWPAN”, in IEEE WiSP-NET, Chennai, India, Mar. 2016.
[Taylor-2016]	Vincent F Taylor, Riccardo Spolaor, Mauro Conti, and Ivan Martinovic. 2016. “Appscanner: Automatic fingerprinting of smartphone apps from encrypted

	network traffic," In 2016 IEEE European Symposium on Security and Privacy (EuroS&P). IEEE, 439–454.
[Taylor-2018]	Vincent F Taylor, Riccardo Spolaor, Mauro Conti, and Ivan Martinovic. 2018. "Robust smartphone app identification via encrypted network traffic analysis," IEEE Transactions on Information Forensics and Security 13, 1 (2018), 63–78.
[Ullah-2022]	Ullah, Imtiaz, and Qusay H. Mahmoud. "An Anomaly Detection Model for IoT Networks based on Flow and Flag Features using a Feed-Forward Neural Network." 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC). IEEE, 2022.
[Vaccari-2021]	Vaccari, Ivan, et al. "Exploiting Internet of Things Protocols for Malicious Data Exfiltration Activities." IEEE Access 9 (2021): 104261-104280.
[Vanhoef-2017]	Mathy Vanhoef and Frank Piessens. 2017. Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17). Association for Computing Machinery, New York, NY, USA, 1313–1328. https://doi.org/10.1145/3133956.3134027
[Wang-2015]	Q. Wang, A. Yahyavi, B. Kemme and W. He, "I know what you did on your smartphone: Inferring app usage over encrypted data traffic," 2015 IEEE Conference on Communications and Network Security (CNS), 2015, pp. 433-441, doi: 10.1109/CNS.2015.7346855.
[Wang-2018]	P. Wang, F. Ye, X. Chen and Y. Qian, "Datanet: Deep Learning-Based Encrypted Network Traffic Classification in SDN Home Gateway," in IEEE Access, vol. 6, pp. 55380-55391, 2018.
[Wright-2010]	Charles V. Wright, Lucas Ballard, Scott E. Coull, Fabian Monrose, and Gerald M. Masson. 2010. Uncovering Spoken Phrases in Encrypted Voice over IP Conversations. ACM Trans. Inf. Syst. Secur. 13, 4, Article 35 (December 2010), 30 pages. https://doi-org.proxy.library.carleton.ca/10.1145/1880022.1880029
[Yan-2018]	F. Yan et al., "Identifying WeChat Red Packets and Fund Transfers Via Analyzing Encrypted Network Traffic," 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), 2018, pp. 1426-1432, doi: 10.1109/TrustCom/BigDataSE.2018.00198.
[Yin-2021]	Yin, Feihong, et al. "IoT ETEI: End-to-end IoT device identification method." 2021 IEEE Conference on Dependable and Secure Computing (DSC). IEEE, 2021.
[Zahid-2022]	Zahid, Hafiz Muhammad, et al. "A Framework for Identification and Classification of IoT Devices for Security Analysis in Heterogeneous Network." Wireless Communications and Mobile Computing 2022 (2022).

[Zhang-2011]	Fan Zhang, Wenbo He, Xue Liu, and Patrick G. Bridges. 2011. Inferring users' online activities through traffic analysis. In Proceedings of the fourth ACM conference on Wireless network security (WiSec '11). Association for Computing Machinery, New York, NY, USA, 59–70. https://doi-org.proxy.library.carleton.ca/10.1145/1998412.1998425

5. Acronyms

Acronym	Meaning
1D-CNN	One-Dimensional Convolutional Neural Network
2D-CNN	Two-Dimensional Convolutional Neural Network
ACL	Access Control List
AdaBoost	Adaptive Boosting
AE	AutoEncoder
AI	Artificial Intelligence
ARE	Acquisitional Rule-based Engine
ARP	Address Resolution Protocol
Bi-GRU	Bidirectional Gated Recurrent Network
BiLSTM	Bidirectional Long Short-Term Memory
BNB	Bernoulli Naive Bayes
C4.5	A decision tree-based classification algorithm
CCTV	Closed Circuit Television
CEC	Consumer Electronic Control
ClaSP	Closed Sequential Patterns algorithm
CNB	Complement Naive Bayes
CNN	Convolutional Neural Network
ConvNet	CNN
DDoS	Distributed Denial of Service
DL	Deep Learning
DNN	Deep Neural Network
DNS	Domain Name Server
DoS	Denial of Service
DRM	Digital Right Management
DT	Decision tree
DTC	DT classifier
DTMC	Discrete Time Markov Chain
E-IoT	Enterprise Internet of Things
ENTA	Encrypted Network Traffic Analysis
EU	European Union
FCNN	Fully Connected Neural Network
FFN	Feed Forward Network
FPM	Frequent Pattern Mining
FTP	File Transfer Protocol
GBDT	Gradient Boosting Decision Tree
GBPS	Gigabits per seconds
GDPR	General Data Protection Regulation
GMM	Gaussian Mixture Model
GNB	Gaussian Naïve Bayes
GP	Genetic Programming
GRU	Gated Recurrent Unit
HC	Hierarchical Clustering
HDMI	High-Definition Multimedia Interface
HDNN	Hierarchical Deep Neural Network

HMM	Hidden Markov Model
HTTP	Hyper Text Transfer Protocol
ICMP	Internet Control Message Protocol
ID	Identification
IETF	Internet Engineering Task Force
IIoT	Industrial Internet of Things
IoMT	Internet of Medical Devices
IoT	Internet of Things
IP	Internet Protocol
IPFIX	IP Flow Information Export
ISP	Internet Service Provider
IT	Information Technologies
J48	J48 algorithm is one of the most widely used machine learning algorithms to examine the data categorically and continuously. (https://medium.com/@nilimakhanna1/j48-classification-c4-5-algorithm-in-a-nutshell-24c50d20658e)
KDE	Kernel Density Estimation
KNN	K-Nearest Neighbours
KPI	Key Performance Indicator
LEA	Law Enforcement Agencies
LR	Logistic Regression
LSGD	Linear Models with Stochastic Gradient Descent (LSGD).
LSTM	Long Short-Term Memory
LSVM	Linear Support-Vector Machine
MAC	Media Access Control
ML	Machine Learning
MLP	Multi-Layer Perceptron
MNB	Multinomial Naïve Bayes
MQTT	Message Queue Telemetry Transport
MQTTS	Message Queue Telemetry Transport Secure
MUD	Manufacturer Usage Description
NAT	Network Address Translation
NB	Naïve Bayes
NBM	Naïve Bayes Multinomial
NC	Nearest Centroid
Nmap	Network Mapper
NN	Neural Network
NoT	Non-IoT
NTP	Network Time Protocol
NVR	Network Video Recorder
OS	Operating System
OT	Operational Technology
OTT	Over-The-Top
PCA	Principle Component Analysis
PR	Passive-Aggressive
PTM	Packet Train Matching
RBF	Radial Basis Function

RBFN	Radial Basis Function Network
RC-4	Rivest Cipher 4, a stream cipher Ron's Code 4
ResNet	Residual neural network
RF	Random Forest
RFC	Request for Comments
RNN	Recurrent Neural Network
RR	Classifier using Ridge Regression (RR)
RRSE	Root Relative Squared Error
SAE	Stacked AutoEncoder
SCM	Supervised Category Mapping
SCP	Secure Copy Protocol
SDN	Software Defined Network
SFTP	Secure File Transfer Protocol
SM	Similarity measure using Jaccard's coefficient
SMS	Short Message Service
SNMP	Simple Network Management Protocol
SotA	State-of-the-Art
SPCaps	Session-packets-based encrypted network traffic classification model using capsule neural networks (CapsNet)
SS	Smart Spying
SSDP	Simple Service Discovery Protocol
SSH	Secure Shell
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
VAE	Variational Autoencoder
VoIP	Voice over IP
VPN	Virtual Private Network
WEP	Wired equivalent privacy
Wi-Fi	Wireless Fidelity
WPA	Wi-Fi Protected Access

6. Appendix A: Summary Tables of Application Detection Solutions

Appendix A contains tables summarizing the review of the second- and third-level application detection solutions. For each table, we indicate the authors of the solutions, the level of application detection, the size and type of application set, support for real-time deployment, solution types, traffic object, and input data characteristics.

Note that the acronyms used in this section is spelled out in Section 5. Some entries for the application level have appended a + or - sign to indicate that the applications being detected may be considered to be more or less than the defined level. The – sign is usually due to the mixing of service and application categories, considering high-level applications (e.g., Chat or just Facebook) than just specific application (e.g., Facebook Chat). The + sign is usually due to some indication of user activities being inferred. Although Table 2 is supposed to list only those of the second-level application detection solutions, we include [Cui-2019] because of its DL-based solutions.

From the tables, few solutions support or describe real-time deployment, except [Zhang-2011] which explicitly indicated suitability for real-time deployment. Most solutions use one aggregated set of extracted features unlike [Akbari-2021a, Akbari-2021b] which partitions the feature set into three and feeds each one to a different algorithm/solution.

Table 2 Summary of Second-Level Application Detection Solutions.

Authors	Application Level	Application Set	Real-time Support	Classifier	Traffic Object	Input Data
[Aceto-2020]	2nd	Two-app set, 49-app set, and 45-app set	Not addressed	DL (DNN, AE, SAE, 1D-CNN, 2D-CNN, LSTM, bi-GRU	F/BF	L4 payload [N Bytes] K fields [N_p packets]
[Akbari-2021a, Akbari-2021b]	2nd+	8 service categories, 5 Google services, 19 service/app classes	Not addressed	DL+ML (LSTM, CNN, MLP)	BF	Handshake headers, flow time-series, flow statistics, respectively for each of the solution component
[Alan-2016]	2nd	1595 apps	Not addressed	ML (SM, GNB, MNB)	Bidirectional TCP/IP headers	32-64 initial packets
[Al-Obaidy-2019]	2nd	5 social media sub-classes	Not addressed	ML (SVM, MLP, NB, C4.5)	BF	Flow-based features/statistics

[Alshammari-2011]	2nd-	2 traffic tunnel types, 5 to 11 services over each tunnel	Not addressed	ML (C4.5, AdaBoost, GP)	F	Flow-based features and packet header feature sets
[Cui-2019]	1st	12 traffic classes with one or more apps each	Not addressed	DL (SPCaps, 1D-CNN, CNN+LSTM, SAE)	BF	Flow-based features and packet header feature sets
[DraperGil-2016]	2nd	14 traffic categories of 7 apps in regular and VPN sessions	Not addressed	ML (C4.5, KNN)	BF	Flow-based time-related features
[Hajjar-2015]	2nd-	18 apps	Not addressed	ML (GMM, DTMC)	BF	Flow-based statistics
[Hou-2019]	2nd+	7 services each of which having two or more apps	Not addressed	DL (CNN, LSTM)	BF	Flow features and statistics
[Khatouni-2019]	2nd	11 apps	Not addressed	ML (RF, DT, CNB, MNB, KNN, BNB, LSVM, RR, NC, SVM, PR, MLP, LSGD)	BF	Flow-based features
[Khatouni-2021]	2nd	9 apps	Not addressed	ML (RF, DT, CNB, MNB, KNN, BNB, LSVM, RR, NC, SVM, PR, MLP, LSGD)	BF	Service and network-based features
[Lotfollahi-2020]	2nd+	18 apps and 13 traffic categories	Not addressed	DL (CNN, SAE)	F	Flow header info and first 1489 bytes of each IP packet
[Moore-2005]	2nd-	10 apps categories	Not addressed	ML (NB)	F	Header derived discriminators

[Muehlstein-2017]	2nd	3 OSes, 5 Browsers, 8 apps	Not addressed	ML (SVM, RBF)	BF	Flow features and time series, Features specific to SSL and browsers
[Papadogiannaki-2018]	2nd+	4 OTT Android apps and 3 services	Evaluated	ML (PTM, FPM, ClaSP)	F	Flow features and statistics
[Taylor-2016]	2nd	110 apps	Support online mode	ML (SVM, RF)	F, Burst	Flow-based features/statistics
[Taylor-2018]	2nd	110 apps	Not addressed	ML (RF with ambiguity elimination)	F, Burst	Flow-based features/statistics
[Wang-2015]	2nd	13 apps	Not addressed	ML (RF)	BF	Frame Statistics (802.11 frames)
[Wang-2018]	2nd	15 apps	Not addressed	DL (MLP, SAE, CNN)	F	Flow-based features
[Zhang-2011]	2nd+	7 traffic categories	Support real-time deployment	ML (SVM, NN, RBF)	BF	Flow statistics, MAC-layer traffic,

Table 3 Summary of Third-Level Application Detection Solutions.

Authors	Application Level	Application Set	Realtime Support	Classifier	Traffic Object	Input Data
[Aiolli-2019]	3rd	9 Bitcoin wallet apps up to 7 activities each	Not addressed	ML (SVM, RF)	BF, Burst	Flow statistics, time series
[Brissaud-2018]	3rd	115,500 distinct keywords	Not addressed	ML (KDE)	BF	Thumbnail-keyword pairs
[Brissaud-2019]	3rd	User actions (keywords) in 5 web services	Briefly described resource requirement	ML (KDE, RF)	Flow, Burst	Connection statistics, flow features and statistics, web page-keyword pairs
[Conti-2015]	3rd	User actions in 3 apps	Not addressed	ML (HC, RF)	BF	Flow time series
[Conti-2016]	3rd	User actions in 7 apps	Not addressed	ML (HC, RF)	BF	Flow time series

[Liu-2019]	3rd	8 activities in encrypted video surveillance traffic	Not addressed	ML (RF, GBDT, DT, NB, LR, KNN)	F	Time series, Frequency domain features and traffic rate changes
[Mari-2021]	3rd	Walking directions in real videos	Simulated	DL (CNN-RNN)	F	Video frame statistics
[Park-2016]	3rd	11 activities of KakaoTalk app	Not addressed	ML (RF)	F	Flow statistics
[Pathmaperuma-2020]	3rd	51 in-app activities for 3 social media apps	Not addressed	ML (BN, RF, J48)	BF	Flow features and statistics (802.11 frames)
[Pathmaperuma-2022]	3rd	92 in-app activities for 8 apps	Not addressed	DL (CNN)	F	Frame features and statistics (802.11 frames)
[Wright-2010]	3rd	2000 phrases in encrypted VoIP conversations	Not addressed	ML (HMM)	F	Sub-sequences of packets to phrase matching
[Yan-2018]	3rd	Fund transfers in WeChat app	Not addressed	ML (RF)	BF	Flow statistics and time series

7. Appendix B: Cyber-Attacks Involving and Affecting IoT Devices

Appendix B describes in subsection 7.1 how IoT devices could be involved unwittingly in cyber-attacks; in subsection 7.2 how vulnerabilities of communication protocol could affect the integrity of IoT devices; and in subsection 7.3 how various software used by IoT devices such as service software, monitoring software, and configuration software could affect the maintenance of their security.

7.1. Cyber-attacks in the IoT environment

This subsection will review the most common attacks being carried by compromised IoT devices and the threat these attacks represent for the company.

Data exfiltration

This is about personal data shared or accidentally downloaded or as part of malicious activity. These data can be passwords, personal information, or proprietary data. The Internet of Things environment consists of many wirelesses connected devices which can be vulnerable to an accidental download of sensitive data. Many IoT devices exchange messages from various sources, therefore, being exposed to data exfiltration.

Data exfiltration can take place during Man-in-the-Middle and Sinkhole attacks. At a network level, protocol tunneling threats can occur in wireless networks, and any wireless protocol is vulnerable. The Mirai attack, in 2017 in which billions of IoT things were affected by a distributed attack to the DynDNS. See [Vaccari-2021] for more information.

Some examples of data exfiltration are unwanted data transmission, sensitive data leaked through business email, from secure computers to untrusted third parties or to insecure private systems, as plain text in SMS or email, or file attachment. Known targets include email addresses, business forecasts, and databases. Within Google cloud infrastructure, Digital Rights Management (DRM) tools can secure the files by adding access permissions and encryption. Screenshots with sensitive information can be watermarked as in the case of a list of passwords, user IDs etc. This is done with dynamic watermarking. See [Google-2022] for more information.

Distributed Denial of Service

A distributed DoS or DDoS is about sending a large number of service requests to several targeted computers to consume the resources of a network and its computers. Usually, DDoS is launched by botnet or zombie computers. To overcome this situation, these attacks were first classified depending on their scope and nature. The flooding DDoS attacks were classified depending on network and application layers.

[Salman-2022] analyzes how to identify the type of traffic and the type of devices connected to the network using methods based on machine learning to classify and detect malicious traffic. The characteristics of the network traffic depending on the device's specifications, so they propose systems based on machine learning and thus extract only characteristics of the type of device and the traffic when the device connects to the network for the first time. The proposed model consists of four tasks: 1) feature extraction, 2) IoT device identification, 3) traffic type identification, and 4) intrusion detection. The models studied by [Salman-2022] to solve the proposed problems are *Random Forest*

and learning-based methods, such RNN, *ResNet* and *ConvNet*. The results obtained between IoT and non-IoT traffic are feasible, where accuracy of 99.93% and *F1-score* of 0.9985 are achieved with the *Random Forest* model.

[Geetha-2022] proposes to perform a preliminary analysis to select the most representative features of the dataset used to train the model. Such a model uses BiLSTM layers, which allow, once the packets have been processed to extract the characteristics indicated by the researchers, to learn both the spatial and temporal characteristics of the data used as input. After these layers, two layers of neurons are added to the model, allowing the final output to be generated. This model can obtain an accuracy of 84.8% to detect attacks from the most well-known botnets.

7.2. E-IoT communications layer threat model

Wi-Fi attacks

Wi-Fi communication has been an active research topic due to its widespread appeal and uses in many connected devices. Several Wi-Fi attacks have been documented in various publications, surveys, and technical documents. Furthermore, attacks may be affected by installed hardware, firmware, security (e.g., WEP, WPA), and implementation. A survey conducted by [Lashkari-2009] revealed flaws in Wi-Fi security mechanisms. This work notes that WEP is vulnerable to attacks (e.g., packet forgery, replay attacks, de-authentication) and vulnerabilities such as improper key management and RC-4 algorithm problems.

Even with improvements, WPA and WPA2 can be vulnerable to attacks (e.g., brute force attacks). A handshake capture attack is a related WPA/WPA2 attack. An attacker can intercept the communication handshake and attempt brute force or dictionary attacks on the captured handshake. [Vanhoef-2017] propose critical re-installation attacks against WPA/WPA2, in which attackers can force a Wi-Fi network to reuse old keys, risking network confidentiality.

Finally, some flaws in WPA3 have been discovered as a newer security mechanism. As a result, denial-of-service attacks, connection deprivation attacks, and handshake attacks can jeopardize WPA3 security. Because many E-IoT devices communicate via Wi-Fi, any Wi-Fi attacks may impede the confidentiality, integrity, and availability of E-IoT and E-IoT-integrated components.

Attacks on the HDMI Protocol

HDMI is one of the essential video distribution connections, and it contains several protocols that can be dangerous to E-IoT systems. [Puche-2019] demonstrated in HDMI-Walk that the CEC protocol could be used to gain arbitrary control of CEC-supported device functions.

The authors specifically demonstrated how CEC could be used with HDMI distributions to attack multiple HDMI devices. The HDMI-Walk attacks also demonstrated that an attacker could control devices, transfer data, cause DoS conditions, eavesdrop, and otherwise harm HDMI networks via a single point of connection or compromised device. The researchers used an HDMI-capable distribution to carry out all of the attacks. The first attack used the inserted device to gather information about all connected HDMI devices, returning information such as language, model number, power state, and running version. Two more attacks demonstrated that CEC could be used for eavesdropping and facilitating existing attacks.

Finally, two DoS attacks were demonstrated in HDMI-Walk. In the first attack, the attacker device was set up to detect televisions turning on via CEC broadcast and shut them down before they started. The second DoS attack took advantage of television input change and overwhelmed displays via CEC, rendering them inoperable. Furthermore, the HDMI-Walk authors noted that CEC propagation is not apparent and difficult to mitigate, resulting in networks that are not visible to the user. The NCC group also published relevant work on HDMI sub-protocols that identified CEC-based fuzzing vulnerabilities and other viable threats via HDMI.

7.3. Layers of monitoring and applications

In this subsection, the different applications and services monitoring IoT devices are reviewed. With this analysis, a deeper knowledge about what characteristic ENTA platform needs is obtained.

Services for IoT Software

For configuration and maintenance, E-IoT systems make use of several software services. E-IoT uses common application services and proprietary tools used by E-IoT vendors, such as Control4's composer and Crestron's Simpl. The available software services may differ from one system to the next. While well-known, documented software services such as File Transfer Protocol (FTP), Secure Shell (SSH), and Telnet communication are available in E-IoT systems, E-IoT solutions may also run unknown proprietary services. Because many E-IoT systems are closed-source, documentation and details of these proprietary services are mostly unavailable. As a result, among the few sources of information on these services are online operating manuals and troubleshooting guides. In contrast, well-known and frequently used services are easier to locate.

File transfer, for example, is required for E-IoT tasks such as firmware upgrades, image uploads, and vendor software configuration. As a result, FTP is one of the accepted file transfer services, and Secure FTP is used for more secure communication. Diagnostics and configuration are other E-IoT requirements. Thus, integrators must communicate directly with the E-IoT system. As integrators use secure shell clients such as PuTTy to connect to diagnose and configure E-IoT systems and system components via services such as Telnet or SSH, secure shell services may be used for diagnostics and configuration.

Monitoring of IoT devices

Currently, a large set of tools allows monitoring all the equipment that connects to the company network, among which are well-known monitoring solutions such as Elasticsearch. These tools rely on installing programs that collect data from the devices they monitor and send it for processing. This need is complex to satisfy in IoT devices, given the great need for resources that these programs require. That is why traditional tools cannot be used to solve the current problem.

Some tools can be used to monitor the IoT devices. Many companies have and offer information about the devices they monitor.

Nozomi networks is a company that offers various cybersecurity solutions, including tools that are specific to IoT devices. Its tools can offer visibility into company networks, automatically performing security audits and generating accurate reports on the risks of each device with the help of AI. These tools can be classified as active network monitoring tools. These tools do not indicate that they use

artificial intelligence for anything other than reporting, nor do they indicate whether they are effective in detecting cyberattacks that are being carried out from IoT devices. In addition, it is necessary to deploy sensors in all the headquarters of the company to collect the data for later processing in the company's cloud.

Rhebo offers specific cybersecurity solutions for IoT devices that, to be effective, need to be embedded in the IoT device itself. Its solution offers the possibility to protect devices from cyberattacks, tampering, and error states. This solution can be classified as a network device monitoring and observability tool. However, this solution does not explain the methods used to prevent attacks on encrypted networks or provide visibility to new devices that may be added to the network.

Checkpoint has a cybersecurity solution based on continuous monitoring and network segmentation that allows you to discover IoT devices connected to the company network and can segment the network and block attacks on these devices in real time. For their solution to work, you need to install their IoT device agent on the devices where you want full functionality. These tools do not specify what methods they use to detect IoT devices and if they are being used to carry out attacks within the company network.

Palo Alto offers an IoT cybersecurity solution that brings visibility, prevention, and zero-trust policies. This tool offers the ability to identify and classify IoT, IoMT, OT and IT devices and display up to 50 unique attributes including the device's physical location, even if the device has recently connected to the company network. It also offers the ability to perform vulnerability scanners on an ongoing basis. Although this company offers the ability to detect IoT devices connected to the network, it is not specified that they can identify IoT devices that are compromised and being used to carry out attacks.

Armis is a company that offers security solutions for companies containing IoT, OT, and IT devices. The Armis platform can discover all the devices on the network, offering a large amount of information on each of the located devices, performing only passive network scanners. The platform can perform device security scanners and issue alerts on newly detected devices. Armis does not provide any information about the methods used by its platform to identify devices within the network.

Bastille is a company dedicated to analyzing the radio waves of companies in search of threats, being able to scan the frequency range that goes from 60Mhz to 6Ghz and discover all the devices that are operating within the range of the enterprise, with the ability to place devices on the office floor plan and find out if the device is transmitting information or not.

Configuration of E-IoT

Aside from software, the configuration of E-IoT systems can impact on the overall security of the system. Some E-IoT users may require remote access to E-IoT system features. Furthermore, remote access benefits integrators by allowing them to provide remote technical support, particularly in moving installations such as yachts. As a result, E-IoT vendors and integrators allow remote access via a variety of methods. While each system's configuration varies, most E-IoT systems are accessed remotely via subscription services, virtual private networks (VPNs), or port forwarding. Some vendors provide subscription services, allowing clients and integrators to connect remotely to an E-IoT system (e.g., Control4's 4Sight).

VPNs are another popular solution many vendors recommend, allowing users remote access to the E-IoT network and equipment. As a result, vendors will recommend routers with VPN functionality. Finally, because E-IoT devices (e.g., controllers and CCTV NVRs) frequently use ports for control and configuration, integrators frequently port forward these devices to enable remote access.