


Article

# Threat Hunting System for Protecting Critical Infrastructures Using a Machine Learning Approach

Mario Aragonés Lozano \*, Israel Pérez Llopis and Manuel Esteve Domingo

Communications Department, Universitat Politècnica de València, 46022 Valencia, Spain; ispello0@upvnet.upv.es (I.P.L.); mesteve@dcom.upv.es (M.E.D.)

\* Correspondence: maarlo9@teleco.upv.es

**Abstract:** Cyberattacks are increasing in number and diversity in nature daily, and the tendency for them is to escalate dramatically in the foreseeable future, with critical infrastructures (CI) assets and networks not being an exception to this trend. As time goes by, cyberattacks are more complex than before and unknown until they spawn, being very difficult to detect and remediate. To be reactive against those cyberattacks, usually defined as zero-day attacks, cyber-security specialists known as threat hunters must be in organizations' security departments. All the data generated by the organization's users must be processed by those threat hunters (which are mainly benign and repetitive and follow predictable patterns) in short periods to detect unusual behaviors. The application of artificial intelligence, specifically machine learning (ML) techniques (for instance NLP, C-RNN-GAN, or GNN), can remarkably impact the real-time analysis of those data and help to discriminate between harmless data and malicious data, but not every technique is helpful in every circumstance; as a consequence, those specialists must know which techniques fit the best at every specific moment. The main goal of the present work is to design a distributed and scalable system for threat hunting based on ML, and with a special focus on critical infrastructure needs and characteristics.

**Keywords:** critical infrastructure protection; threat hunting; cyberattacks; artificial intelligence; machine learning

**MSC:** 68T07



**Citation:** Aragonés Lozano, M.; Pérez Llopis, I.; Esteve Domingo, M. Threat Hunting System for Protecting Critical Infrastructures Using a Machine Learning Approach. *Mathematics* **2023**, *11*, 3448. <https://doi.org/10.3390/math11163448>

Academic Editor: Mario Muñoz Organero

Received: 23 June 2023

Revised: 1 August 2023

Accepted: 5 August 2023

Published: 9 August 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

The evolution of IT, and more specifically the internet, has resulted in the adoption of it at every single action; as a consequence, agencies, SMEs (small and medium enterprises), and big companies and CI have to ensure that their internet connection is working properly; otherwise, any service offered by them will be left useless and that can lead to a relevant economic impact. Not only business continuity but other kinds of attacks are conducted by cybercriminals, such as information exfiltration or reputation compromise, to gain a financial benefit.

Taking into account the huge problem of cyberattacks, CI are investing enormous amounts of money in preventing them or, at least, addressing them properly by making their IT security departments bigger. In a desired scenario, no data loss, no data exfiltration, no reputation loss, and, probably the most crucial concern, no business discontinuity would happen. It is necessary to provide the specialized personnel with specific and valuable tools to avoid that they end up overwhelmed by vast amounts of near real-time data resulting in them being unable to address any kind of attacks.

To learn about the kind of traffic that is analyzed, surveys have been conducted [1] with threat hunters, which concluded that a vast amount of the actionable data belongs to harmless actions of the employees (such as DNS requests or WEB browsing). It can

relatively easily be characterizable in patterns that simplify the classification between two sets: innocuous and threatening events. As ML techniques are characterized by extracting patterns from apparently unstructured bare data sets [2], they can be used to properly classify data sets according to the potential threat they can pose, paying attention to the fact that those algorithms must be customized and adapted to the scenarios that might be faced in CI daily basis.

Moreover, as some relevant studies [3–5] point out, human cognition prediction on patterns is strongly influenced by context [6] even more so under stress conditions [7]. In normal conditions, on a day-by-day work basis, threat hunters regularly suffer those stressful conditions as they face extremely challenging situations associated with attacks with enormous amounts of data in highly dynamic scenarios where the slightest error can lead to relevant negative impacts. Furthermore, threat hunting is typically conducted in new and unknown scenarios with limited and incomplete information, embracing many unbounded factors, including in these scenarios the zero-day attacks [8].

Therefore, taking into account the before-mentioned high coupling in the context of prediction by human cognition, human bias could introduce error when discriminating in behavior between an attack to a non-attack quite similar; however, the usage of ML systems could behave more accurately than humans, avoiding their bias. Hence, with the knowledge generated by ML systems (such as feasibility scores, likelihoods, etc.), threat hunters could better understand what is happening in the operations scenario.

Another relevant element in the threat-hunting process, considered a cognitive loop, is the hypothesis generation process. Assisted by ML, threat hunters can enhance their hypothesis generation activity obtaining insights from the machine and even detecting hidden patterns and processes.

To cope with all those before-mentioned requirements and demands, and in order to process and analyze huge data sets to detect threats or at least anomalous behaviors, a reference architecture must be designed and tools must be implemented following it. The key contribution of this work is the development of a system devoted to fulfilling the stated needs highlighting the definition of useful ML techniques to generate hypotheses about what is going on from the gathered raw data.

### *Paper Layout*

The remaining paper is organized as follows: In Section 2 is described the current state-of-the-art considering existing academic and business solutions. An overview of different ML techniques suitable for solving the detected problem is reviewed in Section 3. Section 4 describes how a prototype of the proposed system has been implemented and evaluated. Finally, in Section 5, the achieved conclusions and the future work are exposed.

## **2. Motivation and Previous Work**

The application of ML techniques to threat hunting is flourishing. For instance, the work [9] deals with the application of ML to hasty time-critical systems to provide automation and promptness in response. Both refs. [10,11] are oriented to developing advanced and smart threat-hunting approaches on software-defined networks (SDNs). The field of threat hunting in the internet of things (IoT), characterized by limiting factors such as resource scarceness, has received attention from the research community in applying ML techniques to cope with those limitations. Examples of this are [12,13]. Several efforts have also been made in order to use artificial intelligence to highlight social trends in large amounts of near real-time data, for example, the work [14]. Finally, solutions can be found in the literature with a more general perspective regarding ML applications to threat hunting, such as [15,16].

On the other hand, several works try to develop a threat-hunting architecture using an ML approach. The article [17] proposes an architecture including all steps, from data collection to data visualization, mainly focused on generating indicators of compromise (IoCs) and applying ML techniques to several stages. Another relevant work is [18], which,

aligned to the previous one, attempts to develop an architecture to spot IoCs by means of applying ML, with a different approach from the one taken in the preceding work. It is important to state that none of the studied works propose methods and mechanisms to generate hypotheses from existing data. Another relevant effort in the literature is [19], which shows a complete architecture but only uses a specific social network as a data source. To end up, the work [20] takes into account all the benefits of the above-mentioned proposals and attempts to mitigate the problems detected.

In summary, all the studied works point out the challenges associated with the threat-hunting process, where it is extremely challenging to create a proper situational understanding because of the growing number of not previously known threats or the fast-paced environments with high change rate conditions, among others.

Besides academia, the industry is also making enormous efforts to apply ML techniques for threat-hunting commercial systems. Some relevant examples of commercial products that do provide it are Splunk [21], Palo Alto Firewalls [22], IBM X-Force Exchange [23,24] and Anomali ThreatStream [25].

After the previously shown survey and analysis of existing solutions and approaches in the research area, the architecture defined at [20] takes into account modularity, scalability, and security, among others; as a consequence, it is considered the best option to continue the efforts previously done and to implement a threat-hunting system for protecting CI using an ML approach following that architecture.

It is also considered important to highlight the importance of having a well-designed and reliable mechanism to authenticate the users and to cipher the data between the different components that constitute the system. As proposed in [20], several mechanisms can be considered, but more options can be found at the current state-of-the-art, for instance, [26]. In addition, regarding that the exchanged data can be considered very sensitive, it is considered relevant to implement systems that ensure the integrity of all the operations committed in the distributed system, ensuring that all of them are legitimate and none of them has been executed using old data; furthermore, these systems should also provide mechanisms to audit operations by third-party authorities publicly. To achieve this, current state-of-the-art can be found in works like [27,28] where blockchain technology is proposed.

### 3. Machine Learning

Artificial intelligence, or more accurately ML, is meant to serve as a helper for threat hunters, but they are not usually specialists in data science; as a consequence, without providing specific rules or examples on which ML technique fits the best for every circumstance, ML can result in a problem instead of a helper.

A study on the topic has been conducted to give a first approach to the current state-of-the-art of ML techniques and which one may be used for every circumstance.

#### 3.1. Techniques

There are countless ML techniques that can be used for cybersecurity processing. Deep research has been conducted and the techniques that have been implemented at the prototype are explained thereupon.

##### 3.1.1. Clustering

###### K-Means

K-means is a clustering technique that suffers problems of random initialization and unexpected convergence to local minimums in its original implementation. K-means is based on the generation of  $k$  clusters. Each cluster has a centroid and the algorithm tries to find where the centroid is in a  $N$ -dimensions space to achieve the least squared Euclidean distance [29].

Several alternatives have been proposed in the last few years to solve the problems of the original implementation of K-means, such as [30,31].

### Affinity Propagation

Affinity propagation is a clustering technique that tries to find a set of data points that best exemplify the data which have been processed. According to how the algorithm is defined, it has several limitations, such as no correct definition of the parameters and oscillations. Some alternatives have been proposed to solve the previously listed problems, such as [32,33].

### Mean Shift

Mean shift is a clustering technique in which the most remarkable property is that there are no parameters to configure [34]. It works well for low-dimensional spaces, whereas for high-dimensional space combinations, other approaches must be taken into account, for example, ref. [35] in which a hierarchical approach is combined with mean shift.

### Spectral Clustering

Spectral clustering has become one of the most popular clustering techniques in recent years, becoming a substitute for traditional techniques like k-means in some scenarios.

Spectral clustering is easy to implement and the mathematics behind it can easily be solved using simple linear algebra software.

There are several algorithm variants, as stated in [36], for instance, unnormalized and normalized spectral clustering.

### Hierarchical Clustering

Hierarchical clustering is an unsupervised ML technique used for performing data exploratory analysis, which consists in obtaining the root of a tree in which leaves are the elements of a set of data [37].

It is important to highlight the parameter base distance function used to calculate the distance between elements in the space, which means the dissimilarity.

One example of hierarchical clustering could be the algorithm m-ADIC clustering [38].

### DBSCAN

DBSCAN (density-based spatial clustering of applications with noise) is a clustering method characterized by being able to discover clusters of any arbitrary shape and size, even with sets of data containing noise and outliers. The main issues of using this technique are the parameters that must be configured to work and several computational complexities [39].

To solve the above-mentioned issues, some alternatives have been developed, for example, VDBSCAN (varied DBSCAN) [40], FDBSCAN (fast DBSCAN) [41], etc.

## 3.1.2. Neural Network

### LSTM RNN

LSTM RNN (long short-term memory recurrent neural network) is a kind of neural network capable of learning more than 1000 steps, depending on the complexity of the built network [42,43].

### BiLSTM

BiLSTM (bidirectional LSTM) is a kind of neural network in which, first of all, the input data feeds the LSTM layer, and then, the training is repeated, but, in this case, the LSTM layer receives the data in the reverse order of the sequence of the input data [43,44].

### C-RNN-GAN

C-RNN-GAN (continuous recurrent neural networks with adversarial training) is a kind of recurrent neural network (RNN) whose main objective is to generate a model of the trained data. To achieve the objective, two neural networks must be implemented, one is called a generator, and the other one is called a discriminator. The generator is the one that

will be modeled according to the data, and from a kind of input data, it returns data that must be as close as possible to the training data. To know if the generated data is like the trained one, there is the discriminator, which has as input the generated data and the real data, returns the value of proximity, and, finally, it is the work of the data expert to set a threshold to distinguish between regular and irregular data [45].

### Graph Neural Networks

Complex data, like images or videos, usually are analyzed using (multi-dimensional) kernels which go through all the pixels across all the dimensions. It can be done thanks to the specific properties of the images. However, other kinds of complex data (for instance, social network data, financial data, IP network data, etc.) follow specific patterns, usually node-edge relations or graph relations, which are unsuitable for learning by using kernels. As graph-like data is very usual, enormous efforts have been made to develop specific neural networks defined as GNN (graph neural networks). Some examples of GNN are GNNExplainer, GNN-LRP, and PGExplainer [46].

### 3.1.3. Natural Language Processing

#### BERT

BERT (Bidirectional Encoder Representations from Transformers) is a model which is able to cope with NLP (natural language processing) tasks such as text classification without human supervision [43,47].

#### TF-IDF

TF-IDF (term frequency inverse document frequency) is a statistical technique that is useful to calculate how much significance has a specific word in a full text. This specific technique is useful, for example, to detect malicious URLs, etc. [48,49].

### 3.2. Comparison

After analyzing different ML techniques, a comparison highlighting the most important characteristics can be found in Table 1.

As shown in the table, all machine learning techniques are unsupervised; however, many supervised machine learning techniques are very useful, for instance, decision trees [50], etc.

Another important conclusion is that threat hunters must be helped by data analysts in order to interact with the system, or at least, they must have basic knowledge about how each ML technique works in order to set the required parameters to obtain the desired outcome.

**Table 1.** Techniques comparison.

Technique	Type	Parameters	Supervised
K-means	Clustering	Yes	No
Affinity Propagation	Clustering	Yes	No
Mean Shift	Clustering	No	No
Spectral Clustering	Clustering	Yes	No
Hierarchical Clustering	Clustering	Yes	No
DBSCAN	Clustering	Yes	No
LSTM RNN	Neural Network	Yes	No
BiLSTM	Neural Network	Yes	No
C-RNN-GAN	Neural Network	Yes	No
GNN	Neural Network	Yes	No
BERT	NLP	Yes	No
TF-IDF	NLP	Yes	No

### 3.3. Use Cases

Depending on what a threat hunter looks for, the ML techniques used and how they are ordered differ. As well as with the techniques, the combinations are countless; despite that, some implementations have been done at the prototype in order to verify that the implemented system works properly, and some of them are shown thereupon.

#### 3.3.1. TTP Discovery

MITRE ATT&CK [51–53] is an extremely useful attempt to characterize adversary behavior (mostly advanced persistent threat, or APT) by classifying it into tactics, techniques, and procedures (TTP) [54]. Comparing what is going on in the analyzed infrastructure and the previous APTs analysis, an analyst could know whether a specific APT is attacking the infrastructure.

There are some SIEMs (security information and event management) that are able to classify the events in the corresponding TTP; however, many of them still need that feature, and analysts must do that work. To alleviate analysts from that work, the usage of NLP is proposed to discover TTP, like how MITRE TRAM does [55].

#### 3.3.2. Behaviour Analysis

The developed system is able to gather data from hosts and from networking equipment to have a complete picture of the monitored infrastructure and how its users interact with other users or endpoints. Because of that holistic view, analysts are searching for threats at every possible weakness to act in a very early step of the kill chain. One technique to do it is to create a model about what is normal in the monitored infrastructure, and everything that deviates is analyzed to decide whether it is a threat or not. One main issue is that, sometimes, that divergence is very small, and a person cannot discriminate, but an ML model is more likely to succeed.

A possible solution is to model the behavior of the monitored infrastructure users using ML, and, once it is modeled, everything that diverges more than a threshold could be considered an anomaly, enabling cybersecurity analysts to check it out [56–58].

#### 3.3.3. Alert Priority

Several ML algorithms not only learn from the data gathered from the data collectors, but also from the feedback received from cybersecurity analysts. If a model is trained using the feedback generated by the cyberanalyst, it can learn, for example, which alerts have bigger priority; as a consequence, the system will be able to tag automatically the alerts by priority.

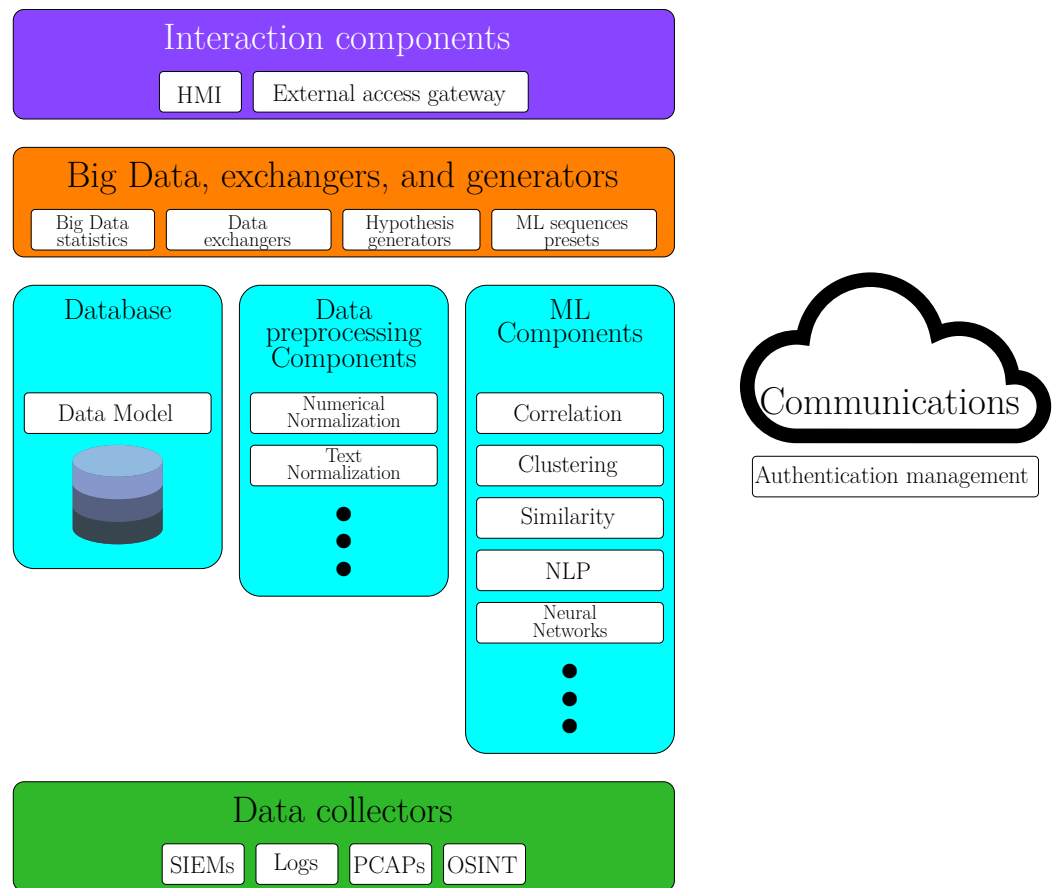
## 4. Prototype

Once the architecture has been chosen, the next step will be to develop the system prototype to evaluate its performance and to decide whether the solution is useful for protecting CI using a machine learning approach.

The developed prototype implements the architecture defined at [20], which is described according to Figure 1. The chosen programming language was Python because it is widely used in ML projects, and there are a lot of specific libraries like Scikit-Learn, Keras, Tensorflow, or PyTorch which do the work with ML techniques easily and, in addition, they are optimized to work with GPU CUDA which results in short processing times.

Following the proposed architecture, each component has been considered an autonomous service in charge of a non-complex specific task that provides functionalities to other services and/or interacts with other services to solve complex tasks.

After each service had been developed and tested, a docker image of the component was created to be deployed in the system. Each component has specific configurations set using the container's environmental variables.



**Figure 1.** Implemented architecture. Obtained from Mario Aragonés Lozano [20], with permission from MDPI, 2023.

After all the components had been developed and containerized, they were deployed using docker-compose. Another alternative would be to use Kubernetes, but for the purpose of resource evaluation, it was considered that deploying the system using docker-compose would be enough.

*4.1. Prototype Evaluation*

After developing the prototype, it was evaluated, simulating real conditions of usage. Our virtualization infrastructure consists of a cyber range running VMware vSphere 6 Enterprise Plus and with the following resources:

- 16 CPU × 2.10 GHz;
- 140 GB RAM;
- 5 TB HDD;

At that cyber range, several components had been deployed during the evaluation phase. First, the developed system with all the components simulating a real distributed scenario was deployed. The following resources were assigned to those components:

- 6 CPU × 2.10 GHz;
- 32 GB RAM;
- 500 GB HDD;

After that, a CI network was deployed consisting of the following elements:

- Main router;
- WAN network:
  - Attackers;
- Prototype network:

- Data collectors;
- Pre-processing components;
- ML components;
- ML sequence preset component;
- Big Data components;
- HMI component;
- ...
- Industrial network:
  - Modbus TCP devices;
  - Profinet devices;
  - KNX devices;
  - ...
- Critical Infrastructure network:
  - Hosts with Linux;
  - Hosts with Windows;
  - Domain Controllers;
  - ...
- DMZ:
  - Apache servers;
  - NGINX servers;
  - DNS servers;
  - Exchange servers;
  - ...

The final schema of deployed elements (the prototype components and the simulated CI network) can be found in Figure 2.

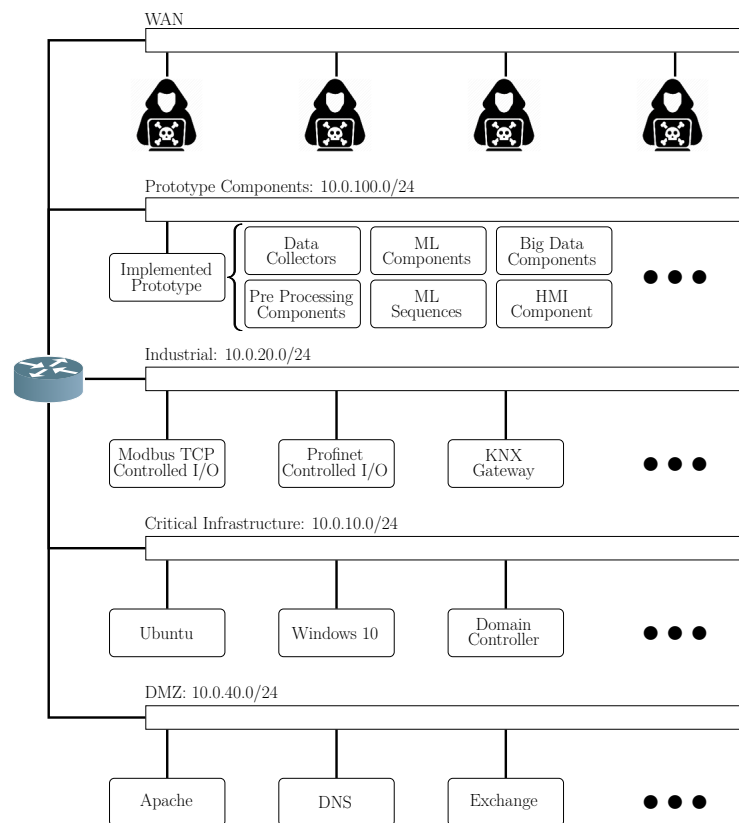


Figure 2. Monitored infrastructure schema.



Once all the components were deployed, all the hosts were configured with real conditions, and several sensors and external triggers simulated a realistic scenario (for example, sensor emulating gates opening/closing, sensor simulating temperature changes, external endpoints making requests to the servers of the DMZ, users interacting with windows server, users interacting with Linux servers, etc.).

Having both the system prototype and the scenario to protect deployed, the final step was to validate and verify the system prototype. The validation was conducted by executing machine learning techniques and ensuring the desired results. The verification was decomposed into two phases and consisted of simulating real attacks against the monitored infrastructure.

The first step, the validation, consisted in creating ML systems by concatenating several ML techniques. After ML systems were created, they were executed using real data gathered from the data collectors. After every execution, the results and the logs generated were analyzed to fine-tune the ML techniques defined until obtaining the desired result.

Then, the system was attacked by pentesters using Kali Linux to generate attacks against the simulated infrastructure. Several kinds of attacks were executed following the process explained thereupon. First of all, network scanings were executed in order to discover assets, their services, and their corresponding vulnerabilities in a stealthy and noisy approach. After that, denial of service (DoS) attacks were also made against network equipment and hosts. Not only DoS attacks, but also distributed denial of service (DDoS) attacks were executed. The next step was to launch man-in-the-middle attacks, for instance, arp spoofing, etc. Other interesting attacks were those tagged as brute force against services like SSH, HTTP, MySQL, etc., to get user enumerations, passwords, etc. To continue, several malware software were created targeting specific hosts and they were established at the host using previously generated reverse shells. They were downloaded and executed, giving us useful information about whether the attack would be detected before it would be executed or not. Phishing campaigns were also created using USBs and emails. Some other attacks against the network and hosts were those exploiting the previous vulnerability detected and bad configurations of the devices. Also, web attacks and SQL injection attacks were executed, among others.

In order to evaluate those attacks, two approaches were followed. On the one hand, threat hunters analyzed the attacks in real-time using the developed system, and on the other hand, all the data were given to threat hunters for conducting a forensic analysis.

Once the evaluation with pentesters using Kali Linux had finished, the next step was to evaluate the system using MITRE CALDERA [59].

With MITRE CALDERA we were able to generate specific threat profiles and used them against the simulated network to analyze those attacks using the developed prototype. The first thing to do was to install MITRE CALDERA agents at the simulated hosts. Afterward, adversary profiles containing the steps a cyberattacker would execute were created. In our specific case, we tried to simulate known steps made by some APTs in different attack campaigns trying to evaluate how useful the system would be if those steps were made to the monitored system. After being defined those actions to reproduce, we executed the campaigns against those agents previously installed.

As well as before, the evaluation of the system was made in real-time, simulating an attack. In addition, using the collected data, threat hunters could conduct a forensic analysis in a post-phase stage.

### Prototype Evaluation Results

The first step to evaluate the prototype was to conduct several validation challenges. To execute those validation challenges, ML systems were generated, each trying to solve different problems that threat hunters face daily. In particular, the evaluation results that will be shown thereupon consist of a cluster of logs in which the temporary value of the inputs is not taken into account; there are other validated systems that have also taken into account the proximity between events.

This ML system (Figure 3) consisted of a concatenation of, firstly, pre-processing components which normalized those logs that did not follow a standard structure. After that, NLP techniques obtained properties like words distribution of the samples due to logs can be viewed as textual data [60]. Finally, clustering techniques were executed to obtain the results.

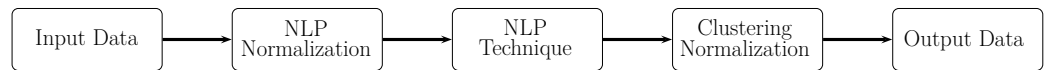


Figure 3. Validation ML system.

After several iterations of fine-tuning the parameters of TF-IDF and spectral clustering techniques that best fit the data requirements, the results were considered useful. A two-dimensional sample was generated from a multidimensional output to visually analyze the result, as shown in Figures 4 and 5.

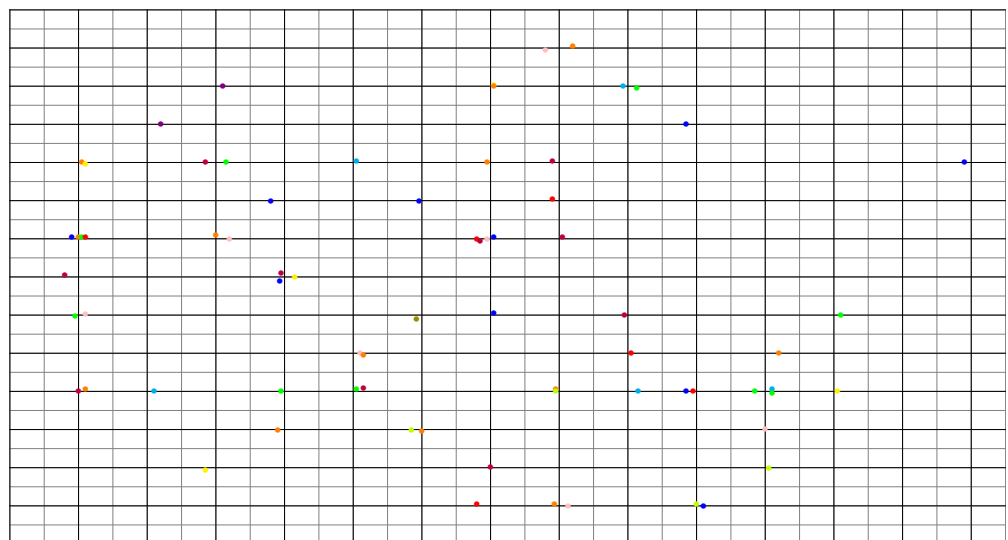


Figure 4. Validation ML system: 2D cluster plot.

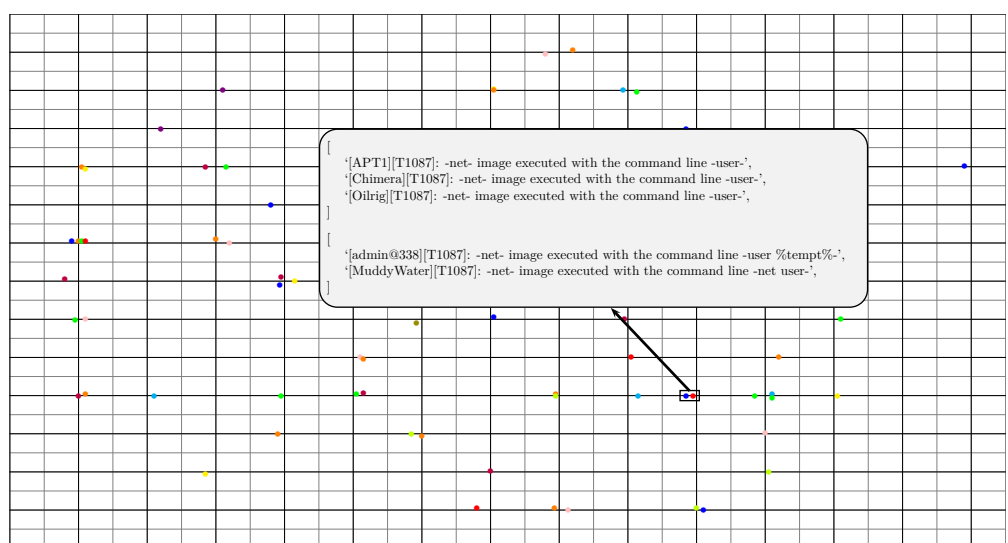


Figure 5. Validation ML system: 2D cluster plot with legend.

As can be seen, the ML system is able to process the data by achieving the desired result, which is clustering it following an NLP approach and grouping the detected logs along with the associated APT together.

The next step of the evaluation process was the first stage of the verification process, in which the monitored infrastructure was attacked by pentesters using Kali Linux. At this step, several ML systems were generated to detect the attacks using different kinds of perspectives. One of them was the usage of C-RNN-GAN (Figure 6) with the analyzed network data for two different goals; first of all, to generate a model of the baseline data of the monitored infrastructure, i.e., the network traffic generated by devices and people by normal usage; and secondly, once the model is generated, to discriminate the new data between baseline and deviant to notify threat hunters when there is something unusual.

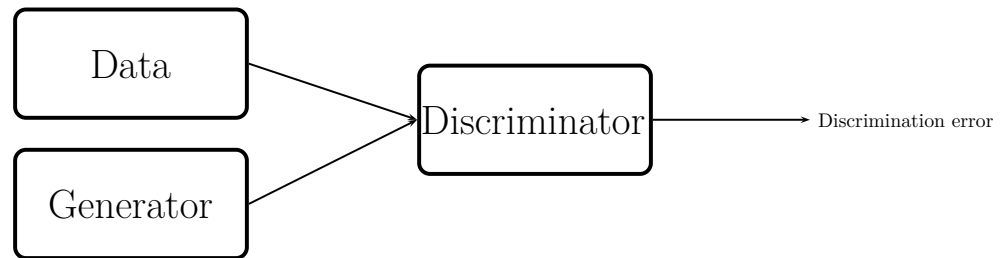


Figure 6. C-RNN-GAN.

To evaluate the results, we used the ROC curve [61] because the area under the curve is a perfect measure of the ability of a system to discriminate whether a specific condition is satisfied or not. The results obtained across the modifications of neural networks’ hidden layers for fine-tuning the C-RNN-GAN neural networks generated the ROC curves in Figure 7.

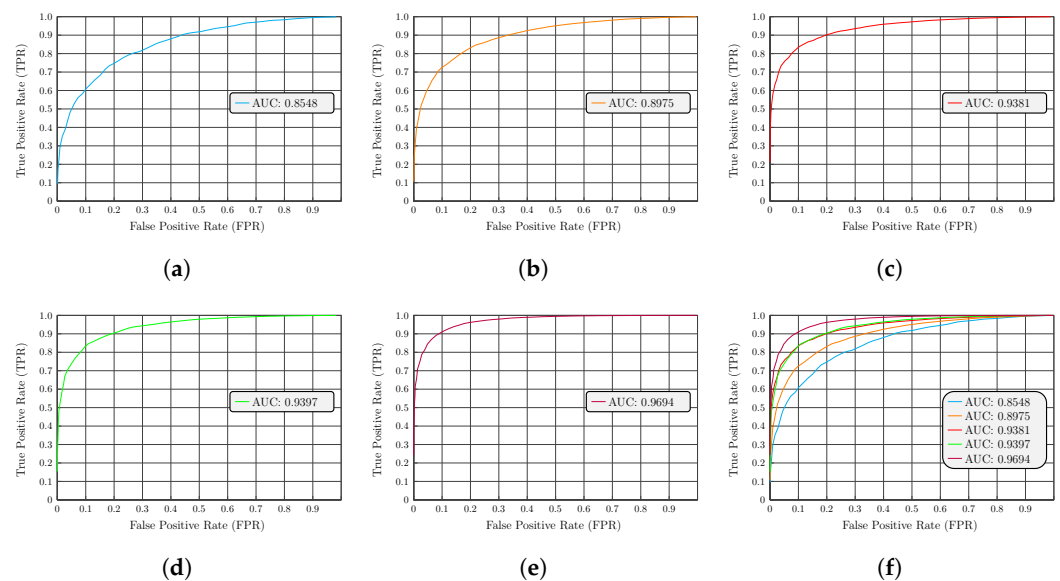


Figure 7. C-RNN-GAN: ROC curve (AUC: area under the ROC curve, the higher the better). (a) First iteration; (b) second iteration; (c) third iteration; (d) fourth iteration; (e) fifth iteration; (f) all together.

The last step of the evaluation process was to attack the monitored infrastructure as an APT would do. The main purpose of this step was to know how capable the developed system would be to detect a hypothetical very stealthy and complex attack. To help us in the execution of this step of the evaluation process, the attacks were executed using the tool developed by MITRE called MITRE CALDERA. With this tool, red teams can model the behavior of APTs and execute those attacks against several hosts efficiently.

MITRE CALDERA has no APT modeled, but several repositories with APT models can be found online. For the purpose of the evaluation of the implemented system, it

was decided to use a model of APT29 [62] because it was considered comprehensive. A screenshot of the MITRE CALDERA attack model can be shown in Figure 8 and the steps of one simulated attack are defined thereupon:

1. RTLO Start Sandcat (T1036);
2. PowerShell (T1086);
3. Automated Collection (T1119);
4. Data from staged file (T1074) and Exfiltration over C2 Channel (T1041);
5. Staging Monkey PNG;
6. UAC Bypass via Backup Utility;
7. Registry Cleanup for UAC Bypass Technique;
8. Planting Modified Sysinternals Utilities;
9. Process Discovery;
10. Artifact Cleanup—Delete Files;
11. Persistent Service—1 and 2;
12. Credentials in Files (T1081)—Chrome;
13. Credentials in Files (T1081)—Private Keys Extraction;
14. Staging Files for PowerShell Module Imports;
15. Screen Capturing;
16. Automated Collection (T1119)—Clipboard (T1115);
17. Automated Collection (T1119)—Input Capture (T1417);
18. Data from Staged File (T1074) and Exfiltration over C2 Channel (T1041);
19. Remote System Discovery (T1018);
20. Identifying Current User on Other Machines;
21. Copy Sandcat File;
22. Startup Folder Persistence Execution;
23. Artifact Cleanup.

The screenshot shows the MITRE CALDERA interface for the 'ATT&CK Eval APT29 - Day 1.A' adversary model. The interface includes a sidebar with navigation options like 'agents', 'abilities', 'operations', and 'PLUGINS'. The main area displays the profile name and a table of attack steps.

Ordering	Name	Tactic	Technique	Executors	Requires	Unlocks	Payload	Cleanup
1	1.A - RTLO Start Sandcat (T1036)	execution	RTLO Override	■				✕
2	1.B - PowerShell (T1086)	execution	PowerShell	■				✕
3	2.A - Automated Collection (T1119)	collection	Automated Collection	■				✕
4	2.B.1 - Data from staged file (T1074) and Exfiltration over C2 Channel (T1041)	exfiltration	Exfiltration Over Command and Control Channel	■				✕
5	3.A - Staging monkey PNG	defense-evasion	masquerading	■				✕
6	3.B - UAC Bypass via Backup Utility	privilege-escalation	Bypass User Account Control	■				✕
7	3.B - Registry Cleanup for UAC Bypass Technique	defense-evasion	Modify Registry	■				✕
8	4.A - Planting Modified Sysinternals Utilities	stage-capabilities	Upload, install, and configure software/tools	■				✕
9	4.B.1 - Process Discovery	discovery	Process Discovery	■				✕
10	4.B.2 - Artifact Cleanup - Delete Files	defense-evasion	File Deletion	■				✕
11	4.C - Loading Stage-2 & Performing Discovery	discovery	System Information Discovery	■				✕

Figure 8. MITRE CALDERA: APT29 adversary model.

The MITRE CALDERA attacks were analyzed using GNN following the approach proposed at [63]. All the detected information received was modeled using graphs to understand the evolution of attacks, not only taking into account the stages of the attack, but also discovering lateral movements inside the network, being able to know which was the entry point and how the attack evolved over time. Using as reference [63], the graph is modeled taking into account both, intrahost relations and interhost relations, as it is explained in Figure 9.

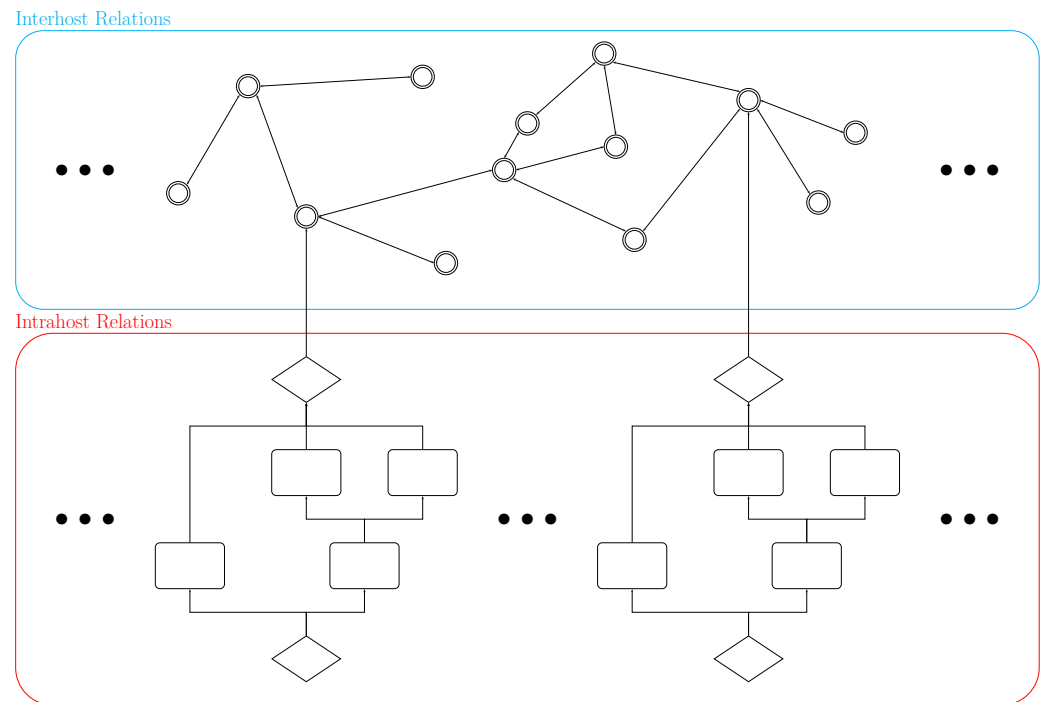


Figure 9. GNN relations.

After training the GNN model, the graph modeled what the system would do after the action of a user (for instance, after visiting a web page, the computer will translate the DNS name into an IP address, then the browser will ask the IP for the host of the web page, etc.) and all the anomalies of the normal behavior will follow irregular paths of the graph, will warn the threat hunters about were to find for threats.

At the work used as a reference, they used ROC curves to analyze the results (as we have done before). However, to enhance the results, we propose to analyze them by using a relation between the probability of an action to be triggered against which action had been finally done by the user.

In order to highlight which edges are less susceptible to be taken, we use the following equation:

$$M = p_m * N \tag{1}$$

where,

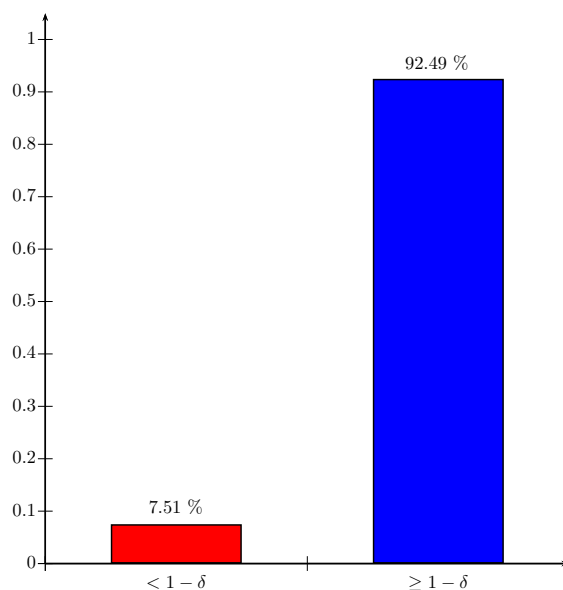
- $p_m$  is the probability of the edge to be taken;
- $N$  is the number of available edges;

Some examples of the application of the given formula would be:

1. There are 2 available edges and the given possibility for the detected edge is  $\frac{99}{100}$ , the result would be  $M = 1.98$ ;
2. There are 100 available edges and the given possibility for the detected edge is  $\frac{1}{100}$ , the result would be  $M = 1$ ;
3. There are 2 available edges and the given possibility for the detected edge is  $\frac{1}{100}$ , the result would be  $M = 0.02$ .

What can be concluded from the given formula is that, in normal conditions, the value for  $M$  will be  $1 \pm \delta$ , and, the values lower than the  $1 - \delta$  will be considered anomalies.

After running the attacks with MITRE CALDERA, the obtained results generated the histogram in Figure 10.



**Figure 10.** MITRE CALDERA: Histogram.

Some examples of unusual paths detected by the GNN would be:

- HTTP requests without previous DNS requests.
- Processes started without a known parent.
- Running processes with no footprint in the file system.
- Outbound persistent connections.

#### 4.2. Prototype Improvements

After conducting all the evaluation phases we were able to confirm that the developed and implemented prototype was able to successfully conduct the security analysis and protection of the monitored infrastructure.

Nonetheless, there is always room for performance improvements and that is something that we also keep in mind. The libraries used, despite that they are for Python, make extensive inner use of highly-optimized C++ libraries for most of the processing being the Python part sort of a wrapper on them; notwithstanding, those libraries are always improving their performance. In addition, all the libraries we used are open-source, and if we detect any kind of improvement we (or anyone) can create requests providing code enhancements to the maintainer. During the development phase, we have been discussing fully moving all the code to C++ instead of this mixture between Python and optimized C++; however, we have found out that the benefits provided are much lesser than the introduced complexities.

Despite that, the evaluation was done in a very limited in-scope scenario with some particular resources. As future work could be interesting to verify the developed prototype in more limited resources machines or with worse quality data than the used for our testing scenarios to detect and solve issues in performance.

## 5. Conclusions

After conducting deep research on the current state-of-the-art systems capable of helping threat hunters by means of artificial intelligence (more specifically, ML) for the protection of CI, it was found that none of the surveyed works take into account all the difficulties that cyberprofessionals face on their daily job as a whole. Therefore, to fill the detected gaps, a system for protecting CI using an ML approach has been developed.

To know exactly which are the needs of cyberprofessionals, surveys were conducted. After analyzing the answers to those surveys, one of the obtained conclusions was that

the vast majority of the actionable data were related to harmless actions of the employees, which, in addition, was repetitive and followed patterns, making it perfectly fit for the usage of ML techniques.

After that, research on the current state-of-the-art was conducted and it was selected the architecture defined at *threat hunting architecture using a machine learning approach for critical infrastructures protection* [20] in which a scalable, big data oriented and online-configurable architecture for protecting CI using ML techniques was defined. One key point of this architecture was the ability to define ML systems to detect anomalies as well as to define hypotheses from the generated data, which fit the main needs detected at the surveys. There were also analyses of which ML techniques could be useful for usage with a TH system.

Then, the architecture was implemented. To continue, several ML techniques were defined inside the corresponding module, for instance, NLP, C-RNN-GAN, GNN, among others. Using the defined techniques, ML systems were created by combining several of them. Each one was in charge of solving some specific problems and all combined were in charge of protecting the simulated infrastructure implemented to test the system. For each one of the components, validation tests were conducted to ensure that they worked as expected.

After that, a simulated infrastructure was deployed trying to emulate the hosts and network equipment of a real CI to use it as a verification scenario. Using this infrastructure, verification processes were conducted to ensure that the implemented prototype was capable of solving most of the TH problems detected at the previously conducted surveys.

From the validation, we were able to verify that each component, individually, was able to return the expected output for the requested inputs. On the other hand, the verification processes' results prove that the overall system was useful to detect anomalies and give an anomaly factor to prioritize some attacks over others.

After finishing that evaluation phase, it was concluded that the developed prototype could detect simple to complex attacks focused on CI using an ML approach in the scope of the evaluation scenarios.

In addition, to enrich the current state-of-the-art, an equation to detect anomalies in graph-based data by means of graph-based neural networks was proposed at the verification step.

The work is not finished here, but we are continuing it by testing the implemented prototype with scarce resources machines and how to enhance our ML techniques and our ML pipelines to make them robust against lower quality and poorer datasets. As a research line, we are starting to work on applying the architecture and the implemented prototype to hybrid (physical and cyber) scenarios by detecting anomalies not only in the cyber domain, but also in the physical domain, and how the possible effects of the events of one domain can be propagated to the other and the other way round.

**Author Contributions:** Writing—original draft, M.A.L., I.P.L. and M.E.D. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the European Commission's Project PRAETORIAN (Protection of Critical Infrastructures from Advanced Combined Cyber and Physical Threats) under the Horizon 2020 Framework (Grant Agreement No. 101021274).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data analyzed in this study were synthetically generated. Data sharing is not applicable to this article.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

APT	Advanced persistent threat
BERT	Bidirectional encoder representations from transformers
BiLSTM	Bidirectional LSTM
C-RNN-GAN	Continuous recurrent neural networks with adversarial training
CI	Critical infrastructures
DBSCAN	Density-based spatial clustering of applications with noise
DDoS	Distributed denial of service
DoS	Denial of service
FDBSCAN	Fast DBSCAN
GNN	Graph neural networks
IoC	Indicators of compromise
IoT	Internet of things
IP	Internet protocol
IT	Information technology
LSTM RNN	Long short-term memory recurrent neural network
ML	Machine learning
NLP	Natural language processing
RNN	Recurrent neural network
SDN	Software-defined networks
SIEM	Security information and event management
SME	Small and medium enterprise
TF-IDF	Term frequency inverse document frequency
TTP	Tactics, techniques, and procedures
VDBSCAN	Varied DBSCAN

## References

1. PRAETORIAN. D3.1 Transitioning Risk Management. PRAETORIAN H2020 Project Deliverables. 2021, *in press*.
2. Li, J.H. Cyber security meets artificial intelligence: A survey. *Front. Inf. Technol. Electron. Eng.* **2018**, *19*, 1462–1474. [[CrossRef](#)]
3. Falandays, J.B.; Nguyen, B.; Spivey, M.J. Is prediction nothing more than multi-scale pattern completion of the future? *Brain Res.* **2021**, *1768*, 147578. [[CrossRef](#)]
4. Federmeier, K.D. Thinking ahead: The role and roots of prediction in language comprehension. *Psychophysiology* **2007**, *44*, 491–505. [[CrossRef](#)] [[PubMed](#)]
5. Riegler, A. The role of anticipation in cognition. In Proceedings of the AIP Conference Proceedings, Liege, Belgium, 7–12 August 2000; American Institute of Physics: Melville, NY, USA, 2001; Volume 573, pp. 534–541.
6. Slattery, T.J.; Yates, M. Word skipping: Effects of word length, predictability, spelling and reading skill. *Q. J. Exp. Psychol.* **2018**, *71*, 250–259. [[CrossRef](#)] [[PubMed](#)]
7. Lehner, P.; Seyed-Solorforough, M.M.; O'Connor, M.F.; Sak, S.; Mullin, T. Cognitive biases and time stress in team decision making. *IEEE Trans. Syst. Man Cybern.-Part A Syst. Humans* **1997**, *27*, 698–703. [[CrossRef](#)]
8. Bilge, L.; Dumitraş, T. Before we knew it: An empirical study of zero-day attacks in the real world. In Proceedings of the 2012 ACM Conference on Computer and Communications Security, Raleigh, NC, USA, 16–12 October 2012; pp. 833–844.
9. Jahromi, A.N.; Hashemi, S.; Dehghantanha, A.; Parizi, R.M.; Choo, K.K.R. An enhanced stacked LSTM method with no random initialization for malware threat hunting in safety and time-critical systems. *IEEE Trans. Emerg. Top. Comput. Intell.* **2020**, *4*, 630–640. [[CrossRef](#)]
10. Schmitt, S. *Advanced Threat Hunting over Software-Defined Networks in Smart Cities*; University of Tennessee at Chattanooga: Chattanooga, TN, USA, 2018.
11. Schmitt, S.; Kandah, F.I.; Brownell, D. Intelligent threat hunting in software-defined networking. In Proceedings of the 2019 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 11–13 January 2019; pp. 1–5.
12. HaddadPajouh, H.; Dehghantanha, A.; Khayami, R.; Choo, K.K.R. A deep recurrent neural network based approach for internet of things malware threat hunting. *Future Gener. Comput. Syst.* **2018**, *85*, 88–96. [[CrossRef](#)]
13. Raju, A.D.; Abualhaol, I.Y.; Giagone, R.S.; Zhou, Y.; Huang, S. A survey on cross-architectural IoT malware threat hunting. *IEEE Access* **2021**, *9*, 91686–91709. [[CrossRef](#)]
14. Xu, Z.; Qian, M. Predicting Popularity of Viral Content in Social Media through a Temporal-Spatial Cascade Convolutional Learning Framework. *Mathematics* **2023**, *11*, 3059. [[CrossRef](#)]
15. Homayoun, S.; Dehghantanha, A.; Ahmadzadeh, M.; Hashemi, S.; Khayami, R. Know abnormal, find evil: Frequent pattern mining for ransomware threat hunting and intelligence. *IEEE Trans. Emerg. Top. Comput.* **2017**, *8*, 341–351. [[CrossRef](#)]



16. Neto, A.J.H.; dos Santos, A.F.P. Cyber threat hunting through automated hypothesis and multi-criteria decision making. In Proceedings of the 2020 IEEE International Conference on Big Data (Big Data), Atlanta, GA, USA, 10–13 December 2020; pp. 1823–1830.
17. Gonzalez-Granadillo, G.; Faiella, M.; Medeiros, I.; Azevedo, R.; Gonzalez-Zarzosa, S. ETIP: An Enriched Threat Intelligence Platform for improving OSINT correlation, analysis, visualization and sharing capabilities. *J. Inf. Secur. Appl.* **2021**, *58*, 102715. [[CrossRef](#)]
18. Azevedo, R.; Medeiros, I.; Bessani, A. PURE: Generating quality threat intelligence by clustering and correlating OSINT. In Proceedings of the 2019 18th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), Rotorua, New Zealand, 5–8 August 2019; pp. 483–490.
19. Alves, F.; Ferreira, P.M.; Bessani, A. OSINT-based Data-driven Cybersecurity Discovery. In Proceedings of the 12th Eurosys Doctoral Conference, Porto, Portugal, 23 April 2018; pp. 1–5.
20. Aragonés Lozano, M.; Pérez Llopis, I.; Esteve Domingo, M. Threat hunting architecture using a machine learning approach for critical infrastructures protection. *Big Data Cogn. Comput.* **2023**, *7*, 65. [[CrossRef](#)]
21. Reed, J. Threat Hunting with ML: Another Reason to SMLE. Available online: [https://www.splunk.com/en\\_us/blog/platform/threat-research-at-splunk-using-smle.html](https://www.splunk.com/en_us/blog/platform/threat-research-at-splunk-using-smle.html) (accessed on 22 June 2023).
22. Liang, J.; Kim, Y. Evolution of Firewalls: Toward Securer Network Using Next Generation Firewall. In Proceedings of the 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 26–29 January 2022; pp. 0752–0759.
23. IBM X-Force Exchange. Available online: <https://exchange.xforce.ibmcloud.com/> (accessed on 3 March 2023).
24. The Security Immune System: An Integrated Approach to Protecting Your Organization. 2018. Available online: <https://www.midlandinfosys.com/pdf/qradar-siem-cybersecurity-ai-products.pdf> (accessed on 3 March 2023).
25. Anomali ThreatStream: Automated Threat Intelligence Management at Scale. Available online: <https://www.anomali.com/products/threatstream> (accessed on 3 March 2023).
26. Chinnasamy, P.; Deepalakshmi, P.; Dutta, A.K.; You, J.; Joshi, G.P. Ciphertext-policy attribute-based encryption for cloud storage: Toward data privacy and authentication in AI-enabled IoT system. *Mathematics* **2021**, *10*, 68. [[CrossRef](#)]
27. Wei, P.; Wang, D.; Zhao, Y.; Tyagi, S.K.S.; Kumar, N. Blockchain data-based cloud data integrity protection mechanism. *Future Gener. Comput. Syst.* **2020**, *102*, 902–911. [[CrossRef](#)]
28. Chinnasamy, P.; Albakri, A.; Khan, M.; Raja, A.A.; Kiran, A.; Babu, J.C. Smart Contract-Enabled Secure Sharing of Health Data for a Mobile Cloud-Based E-Health System. *Appl. Sci.* **2023**, *13*, 3970. [[CrossRef](#)]
29. Hossain, M.; Abufardeh, S. A New Method of Calculating Squared Euclidean Distance (SED) Using pTree Technology and Its Performance Analysis. In Proceedings of the CATA, Honolulu, HI, USA, 18–20 March 2019; pp. 45–54.
30. Ahmed, M.; Seraj, R.; Islam, S.M.S. The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics* **2020**, *9*, 1295. [[CrossRef](#)]
31. Zhao, W.L.; Deng, C.H.; Ngo, C.W. k-means: A revisit. *Neurocomputing* **2018**, *291*, 195–206. [[CrossRef](#)]
32. Givoni, I.; Chung, C.; Frey, B.J. Hierarchical affinity propagation. *arXiv* **2012**, arXiv:1202.3722.
33. Wang, K.; Zhang, J.; Li, D.; Zhang, X.; Guo, T. Adaptive affinity propagation clustering. *arXiv* **2008**, arXiv:0805.1096.
34. Derpanis, K.G. Mean shift clustering. *Lect. Notes* **2005**, *32*, 1–4.
35. DeMenthon, D.; Megret, R. *Spatio-Temporal Segmentation of Video by Hierarchical Mean Shift Analysis*; Computer Vision Laboratory, Center for Automation Research, University of Maryland: College Park, MD, USA, 2002.
36. Von Luxburg, U. A tutorial on spectral clustering. *Stat. Comput.* **2007**, *17*, 395–416. [[CrossRef](#)]
37. Nielsen, F.; Nielsen, F. Hierarchical clustering. In *Introduction to HPC with MPI for Data Science*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 195–211.
38. Murtagh, F.; Contreras, P. Algorithms for hierarchical clustering: An overview. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2012**, *2*, 86–97. [[CrossRef](#)]
39. Khan, K.; Rehman, S.U.; Aziz, K.; Fong, S.; Sarasvady, S. DBSCAN: Past, present and future. In Proceedings of the Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2014), Bangalore, India, 17–19 February 2014; pp. 232–238.
40. Liu, P.; Zhou, D.; Wu, N. VDBSCAN: Varied density based spatial clustering of applications with noise. In Proceedings of the 2007 International Conference on Service Systems and Service Management, Chengdu, China, 9–11 June 2007; pp. 1–4.
41. Zhou, S.; Zhou, A.; Jin, W.; Fan, Y.; Qian, W. FDBSCAN: A fast DBSCAN algorithm. *J. Softw.* **2000**, *11*, 735–744.
42. Staudemeyer, R.C.; Morris, E.R. Understanding LSTM—A tutorial into long short-term memory recurrent neural networks. *arXiv* **2019**, arXiv:1909.09586.
43. Singh, K.; Grover, S.S.; Kumar, R.K. Cyber Security Vulnerability Detection Using Natural Language Processing. In Proceedings of the 2022 IEEE World AI IoT Congress (AIIoT), Seattle, WA, USA, 6–9 June 2022; pp. 174–178.
44. Siami-Namini, S.; Tavakoli, N.; Namin, A.S. The performance of LSTM and BiLSTM in forecasting time series. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; pp. 3285–3292.
45. Mogren, O. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. *arXiv* **2016**, arXiv:1611.09904.

46. Yuan, H.; Yu, H.; Gui, S.; Ji, S. Explainability in graph neural networks: A taxonomic survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 5782–5799. [[CrossRef](#)]
47. González-Carvajal, S.; Garrido-Merchán, E.C. Comparing BERT against traditional machine learning text classification. *arXiv* **2020**, arXiv:2005.13012.
48. Das, M.; Kamalanathan, S.; Alphonse, P. A Comparative Study on TF-IDF Feature Weighting Method and Its Analysis Using Unstructured Dataset. In Proceedings of the COLINS, Lviv, Ukraine, 22–23 April 2021; pp. 98–107.
49. Lakshmanarao, A.; Babu, M.R.; Krishna, M.B. Malicious URL Detection using NLP, Machine Learning and FLASK. In Proceedings of the 2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES), Chennai, India, 24–25 September 2021; pp. 1–4.
50. Safavian, S.R.; Landgrebe, D. A survey of decision tree classifier methodology. *IEEE Trans. Syst. Man Cybern.* **1991**, *21*, 660–674. [[CrossRef](#)]
51. Al-Shaer, R.; Spring, J.M.; Christou, E. Learning the associations of mitre att & ck adversarial techniques. In Proceedings of the 2020 IEEE Conference on Communications and Network Security (CNS), Avignon, France, 29 June–1 July 2020; pp. 1–9.
52. Alexander, O.; Belisle, M.; Steele, J. *MITRE ATT&CK for Industrial Control Systems: Design and Philosophy*; The MITRE Corporation: Bedford, MA, USA, 2020.
53. Ahmed, M.; Panda, S.; Xenakis, C.; Panaousis, E. MITRE ATT&CK-driven cyber risk assessment. In Proceedings of the 17th International Conference on Availability, Reliability and Security, Vienna, Austria, 23–26 August 2022; pp. 1–10.
54. Cole, E. *Advanced Persistent Threat: Understanding the Danger and How to Protect Your Organization*; Syngress: Waltham, MA, USA, 2012.
55. Orbinato, V.; Barbaraci, M.; Natella, R.; Cotroneo, D. Automatic Mapping of Unstructured Cyber Threat Intelligence: An Experimental Study. *arXiv* **2022**, arXiv:2208.12144.
56. Karbab, E.B.; Debbabi, M. Maldy: Portable, data-driven malware detection using natural language processing and machine learning techniques on behavioral analysis reports. *Digit. Investig.* **2019**, *28*, S77–S87. [[CrossRef](#)]
57. Saad, S.; Traore, I.; Ghorbani, A.; Sayed, B.; Zhao, D.; Lu, W.; Felix, J.; Hakimian, P. Detecting P2P botnets through network behavior analysis and machine learning. In Proceedings of the 2011 Ninth Annual International Conference on Privacy, Security and Trust, Montreal, QC, Canada, 19–21 July 2011; pp. 174–180.
58. G. Martín, A.; Fernández-Isabel, A.; Martín de Diego, I.; Beltrán, M. A survey for user behavior analysis based on machine learning techniques: Current models and applications. *Appl. Intell.* **2021**, *51*, 6029–6055. [[CrossRef](#)]
59. Mohamed, N. Study of bypassing Microsoft Windows Security using the MITRE CALDERA framework. *F1000Research* **2022**, *11*, 422. [[CrossRef](#)] [[PubMed](#)]
60. Li, W. *Automatic Log Analysis Using Machine Learning: Awesome Automatic Log Analysis Version 2.0*; Uppsala University: Uppsala, Sweden, 2013.
61. Hoo, Z.H.; Candlish, J.; Teare, D. What Is an ROC Curve? *Emerg. Med. J.* **2017**, *34*, 357–359. [[CrossRef](#)] [[PubMed](#)]
62. Long, M. Adversary Emulation Library. Available online: [https://github.com/center-for-threat-informed-defense/adversary\\_emulation\\_library](https://github.com/center-for-threat-informed-defense/adversary_emulation_library) (accessed on 22 June 2023).
63. Li, Z.; Cheng, X.; Sun, L.; Zhang, J.; Chen, B. A hierarchical approach for advanced persistent threat detection with attention-based graph neural networks. *Secur. Commun. Netw.* **2021**, *2021*, 9961342. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.