REPUBLIC OF TURKEY

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering

PROXYING INTERNET SERVICES AND MACHINE LEARNING FOR THE INTERNET OF THINGS

Mohammed Yousif ALRAWE

Doctor of Philosophy

Supervisor

Asst. Prof. Dr. Sefer KURNAZ

Istanbul, 2022

PROXYING INTERNET SERVICES AND MACHINE LEARNING FOR THE INTERNET OF THINGS

Mohammed Yousif ALRAWE

Electrical and Computer Engineering

Doctor of Philosophy

ALTINBAŞ UNIVERSITY

2022

This thesis titled PROXYING INTERNET SERVICES AND MACHINE LEARNING FOR THE INTERNET OF THINGS prepared by MOHAMMED YOUSIF ALRAWE and submitted on 17/08/2022 has been **accepted unanimously** for the degree of Phd in Electrical and Computer Engineering.

_____

Asst. Prof. Dr. Sefer KURNAZ

Supervisor

Thesis Defense Committee Members:

Asst. Prof. Dr. Sefer KURNAZ	Faculty of Engineering and Archıtecture, Altinbas University	_____
Asst. Prof. Dr. Zeynep ALTAN	Faculty of Engineering and Archıtecture, Beykent University	_____
Asst. Prof. Dr. Abdullahi Abdu Ibrahım	Faculty of Engineering and Nutural Sciences, Altinbas University	_____
Asst. Prof. Dr. Serdar Kargın	Faculty of Engineering and Nutural Sciences, Beykent University	_____
Asst. Prof. Dr. Oguz KARAN	Faculty of Engineering and Nutural Sciences, Altinbas University	_____

I hereby declare that this thesis meets all format and submission requirements of a Phd's disseration.

Submission date of the thesis to Institute of Graduate Studies: ___/___/___

I hereby declare that all information/data presented in this graduation project has been obtained in full accordance with academic rules and ethical conduct. I also declare all unoriginal materials and conclusions have been cited in the text and all references mentioned in the Reference List have been cited in the text, and vice versa as required by the abovementioned rules and conduct.

Mohammed Yousif ALRAWE

Signature

DEDICATION

To my beloved wife, Nadia, who encouraged me to urge myself to the limits. I am wholeheartedly grateful for everything she has done to me; her positive energy kept me focus on my goal whenever I felt desperate. Without her love, encouragement, and support I would not be walking on my path to accomplish my dream, to gain master degree.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my supervisor Asst. Prof. Dr. Sefer KURNAZ for guiding and helping me along the way in writing this dissertation. Discussing my progress, problems, and ideas with my supervisor Asst. Prof. Dr. Sefer KURNAZ, a couple of times every week, helped me tremendously understand the research's logic. It mades me better realize the technical need for this research work.

ABSTRACT

PROXYING INTERNET SERVICES AND MACHINE LEARNING FOR THE INTERNET OF THINGS

ALRAWE, Mohammed Yousif

M.S., Electrical and Computer Engineering, Altınbaş University,

Supervisor: Asst. Prof. Dr. Sefer KURNAZ

Date: 08 / 2022

Pages: 82

The last decades have shown a rapidly growing number of connected devices of the Internet of Things (IoT). Accordingly, more significant access is needed to the internet service to operate the devices. However, access to the services became exhaustive on IoT devices due to limited available resources. This limitation motivated proposing of lightweight protocols, (for example, constrained Application Protocol (CoAP) and MQ Telemetry Transport (MQTT). This research uses these lightweight protocols and proposes a novel system capable of proxying internet services by using their protocols and requirements for IoT devices. On IoT devices, the experiments demonstrate a significant decrease in resources consumption in terms of network bandwidth, time, memory, and energy. Consequently, the efficiency of the devices is significantly improved. Furthermore, the same (futher) tasks can be performed using the less (same) resources devices.

Keywords: IoT, Cloud Computing, HTTP, MQTT, CoAP

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

ABBREVIATIONS

IoT	:	Internet of Things
RF	:	Radio Frequency
MAC	:	Medium Access Control
QoS	:	Quality of Service
TCP	:	Transmission Control Protocol
REST	:	REpresentational State Transfer
HTTP	:	Hyper Text Transport Protocol
M2M	:	Machine-to-Machine
MQTT	:	MQ Telemetry Transport
LTE	:	Light Term Evolution
DSL	:	Digital Subscriber Line
R2L	:	Remote to Local

1. INTRODUCTION

1.1 INTRODUCTION

Since 1974, the TCP protocol (IP) invention, human life has depended on complicated computer applications and networks [1]. It includes daily computer uses, self-driving cars, distributed business networks, and industrial and medical implementations. According to [2], a proxy became a target of systems because of a continuous massive increase of the stored data. However, computer systems have a direct negative economic effect against successful proxies. Networks defense is a hot open research area because of the rise in network proxies and the increase in sophisticated and available internet technologies [3]. Therefore, knowledge of all network protocols is required by the defense. Therefore, for the proper interpretation of the collected data, capture software applied several complicated parsers. According to [4], processing an increasing of massive information needs efficient algorithms and implementation, and thus a big challenge. It is essential to process the data in real-time to avoid or lighten economic issues and a fast uncovering of past and present proxies.

Zero-day proxies are new vulnerabilities that occur daily and fastly investigated. The authors in [5] stated that detection of unknown previously threats is not possible by the strategies of signature-based detection. In the common tools, techiques of machine learning are rarely found, although the topic has been investigated in the last two decades. In the anomaly-based proxying internet, there are many false positives and issues in proper data evaluation and training [6]. In network environments, the software stack is due to significant changes in traffic patterns. To evaluate the strategies of anomaly-based proxying internet, using modern data sets is crucial. Therefore, finding the breaches of a network is significantly supported by network proxying internet. Additionally, trace the breaches back to the responsible parties. Accordingly, the appeared damage can be retrieved and isolated [7].

Furthermore, proxying internet systems' monitoring capabilities and proxy recognition have a deterrent impact on the proxy (has a significant risk of being prosecuted and discovered). A proxy can be convinced to look for else target due to a proxying internet occurrence. Due to an alert, being blocked by an analyst or monitor, unwanted attention is created for the intruder and reduces

his work [8]. An extensive and reliable information source is needed to make correct anticipation, and thus to take proxying internet benefits.

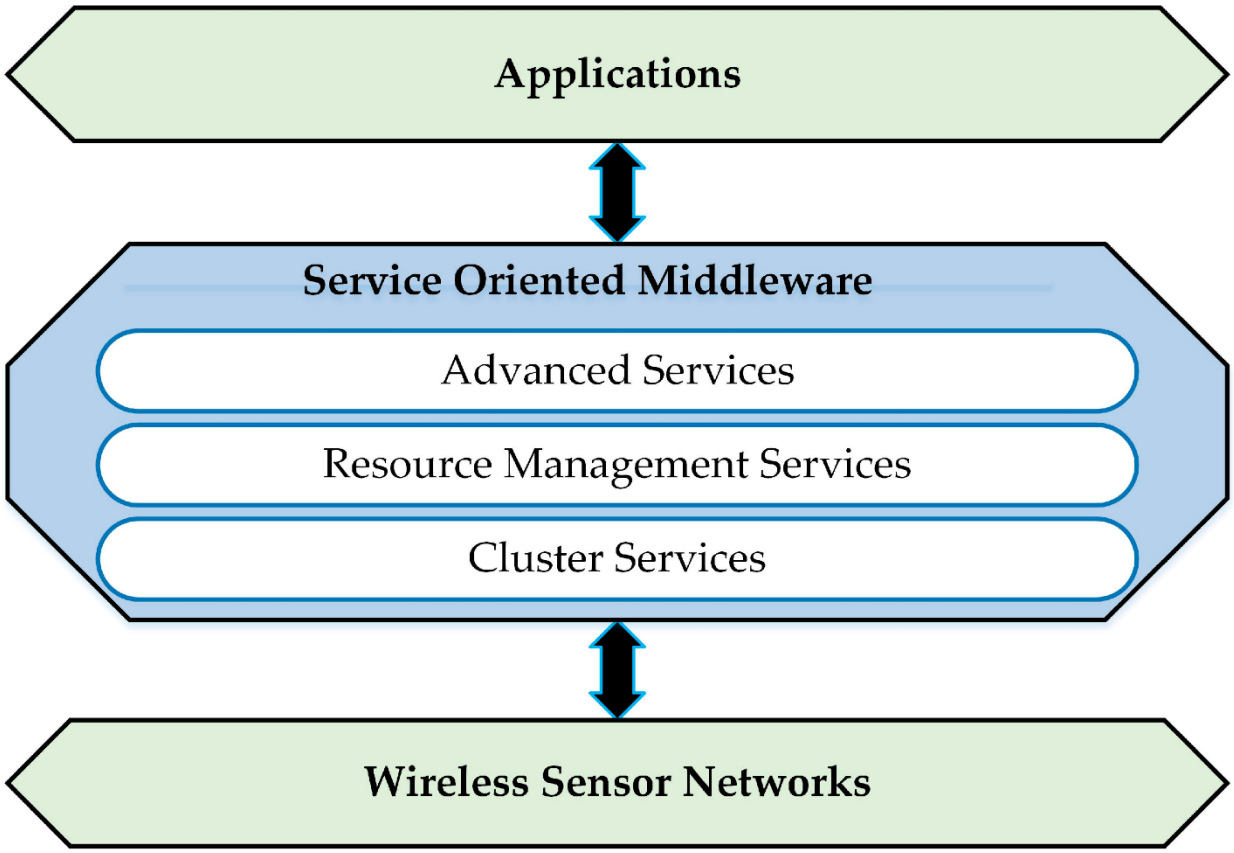


Figure 1.1: Wireless sensor networks' provisioning schema [8].

After compromising a system, access control and authentication (i.e., prevention strategies) may fail. Therefore, proxying the internet works as a second defense line and plays as an incident response base [9]. To find interest assets and do reconnaissance, time passes for the proxy after getting a foothold on a network. According to [10], when proxies lateral movement is detected, more damage is prevented although undetected of initial infection. For more forensic investigation, proxying the internet is a key to uncovering such behavior and crucial in reconstructing it. 1/3 of organizations detected themselves as an intrusion. According to a report published in 2018, 99 (101) days in 2016 (2017) is the median detection time [11]. The time is high, although the reduction time is good. For a proxy, one hundred is large to detect assets of interest and react

against the intrusion. Furthermore, it is subject to insiders (privileges are misused), although of most secure and sophisticated developed systems. According to [12], alarm trends are seen when looking at vulnerability visualization, where the yearly reported vulnerabilities are increasing rapidly, and most of them are reported as high or medium severe.

Proxying the internet is considered crucial for information security and network defense. It plays an essential role in detecting any attempt to compromise the information of a system and protect its confidentiality, integrity, and availability. In the literature, Security information and event management (SIEM) systems were proposed to perform the following: 1) Organize massive data; and 2) Interoperability provided between input sources and different data formats [13]. To a SIEM, the fed data is classified into four types: 1) Threat intelligence feeds; 2) logs generated by different systems; 3) host data from the monitored endpoints; and 4) network traffic [14]. Even if a proxy tries to obfuscate or hide its occurrence, the network's data is valuable information because each proxy needs to spread in the network and leaves traces. On the other hand, since proxies manipulate a compromised system, the host's data is not usually reliable [15].

Recently, a work stated that eighty-three percent of routers consist of vulnerable code (the research published by the American Consumer Institute Center for Citizen Research). Even after the vulnerabilities, Infrequent updates make the device, for a long period, to ve vulnerable [16]. Network security monitoring can detect malicious attempts from both compromised hardware or software components and contact the outside world. For compromise afterward indicators and to find whether there were other past trails, it allows it searches the gathered data [17]. The data must be transferred to the server (that the intruder controls) to infiltrate information or assets. Accordingly, data of the network is made to be robust evidence. Trojan's development seems to be increasing and not stopped (as a study demonstrated implemented on Remote Access Tools (RATs) history). Over the last two decades, the project studied three hundred of the most critical RATs [18]. Since new malware variants appear frequently and signatures are able only to detect known malicious software, the importance of behavior-based detection strategies is proven.

Several issues are faced by collecting features for network proxying internet. According to [20], the increasing volume of data needed to process: in corporate environments, video conferences, big file tansports, and high-resolution multimedia streaming are common scenarios. Then, many network packets (need to process) are generated. Proxies against the monitoring system are another

problem (result in crashes). Consequently, audit data is lost, or even remote compromise of the network monitor. From traffic, low-level programming languages are used to write the available solution for features collection. Automated memory management is not provided by low-level languages; therefore, it increases the proxy surface enormously [21]. Consequently, memory corruption vulnerabilities can occur, allowing for remote code proxies or denial of service.

Therefore, new protocols or continuous updates in protocols features (even implementation state is being secure or audited). It reintroduces exploitable security vulnerabilities. Additionally, to qualify for a board application, the collection sensor must be functional across all major platforms and independent of a particular architecture. To enable quick and easy integration into the stack of network monitoring, the format of the output data must be (as possible) consumable as in several systems. The comma-separated values (CSV) is the most implemented data format for machine learning and analysis of big data. The CSV does not provide any structure to the data or even preserve data kinds. For large environments, the existing solution's configuration and application are time-intensive and complex [22].

An increasing encrypting of shared traffic, entering content is denied, and analysis techniques are required (independent of payloads of clear text packets for classifying). Since most of the available tools have not been developed for a research task, the current state of them is not sufficient to meet the needs for effective and convenient tests. In client behavior or architecture, the authors in [23] stated that there should be basic differences. A further realistic scenario would be created inside the network during a penetration test, collecting network traffic and utilizing this as a dataset to check the implemented countermeasures. As a result, better outcomes are achieved. There is no tool available publicly to complete this type of independent verification effectively and reproducibly.

1.2 MOTIVATION

Due to logic errors or memory corruption vulnerabilities, many software systems are exploitable. An example is buffer overflows. A section of memory is being overfilled with data and the following section overwritten, which can lead to a crash or even remote code execution. Another common example is used after free vulnerabilities, which refer to the process of accessing memory after it has been freed, as mentioned in [24]. This can also cause a program to crash or result in the

execution of arbitrary code. The programming language and its derivatives without automated memory management suffer from these vulnerabilities due to human errors in bounds checking or compiler optimizations that eliminate bound checks. However, languages that provide such automated memory management are not pruned to proxies. For example, the Java virtual machine and object serialization has been exploited. Web browsers are a well-known target for proxies, as they implement a huge protocol stack and have many potentially vulnerable components, such as plugins and extensions, as mentioned in [25]. A commonly exploited component of web browsers is Adobe Flash, a programming language for creating animated web content.

Even big companies that invest large amounts of money in a security team and offer bug bounties for submitted vulnerabilities fail to regularly protect themselves or their users from proxies. In the recent Facebook incident, proxyers exploited a technical vulnerability to steal access tokens that would allow them to log into about 50 million people's accounts. Apple recently fixed a vulnerability in the ICMP packet error handling code of their XNU kernel, which resulted in a heap overflow and could potentially be exploited for remote code execution (CVE-2018-4407). The vulnerability exists in such a fundamental part of the operating systems networking code that anti-virus software cannot detect exploitation attempts, as mentioned in [26]. Also, even companies run by engineers aware of computer security implications (e.g., the spyware manufacturer Hacking Team) have been compromised with a zero-day exploit against an embedded device in their network in the past [27]. A new zero-day in Cisco's Adaptive Security Appliance and Firepower Threat Defense Software has recently been discovered due to an error in handling Session Initiation Protocol (SIP) packets. The vulnerability allows an proxyer to reload or crash the affected systems remotely. The proxy surface is further increased by the tools used for forensic investigation. Protocol dissectors have a huge proxy surface due to the number of supported protocols and implemented code to be maintained. Wireshark, for example, has 1400 authors, supports over 2400 protocols, and is made of a total of at least 2.5 million lines of code. Due to its complexity and dynamic nature, current software stacks are potentially vulnerable to exploitation and compromise [28]. Therefore, monitoring them is necessary to ensure rapid uncovering of breaches and increase the proxyer's efforts and costs. Finally, the network monitor might also be the target of a proxy.

1.3 PROBLEM STATEMENT

Recently, significant attention has been paid to providing digital services for more the thirty-five billion IoT devices [5]. Such as medical [6, 7, 8], agricultural [9, 10], transportation [11, 12], and smart cities [13, 14], the services empowered by utilizing IoT devices in various areas. However, than an IoT device, services are primarily developed for PCs that have extensive resources (e.g., bandwidth memory and processing). Service provision is performed by utilizing complicated protocols (such as XML and HTTP) [2, 15, 16, 17]. In the devices and utilizing the protocols, the provisions of services alter from developing communication for data exchange [18, 19], process data [8, 20], and more achievements [21]

For devices, recent services have begun to utilize other protocols (e.g., MQTT and CoAP) to decrease the overhead needed by the protocols implemented in web service [22, 23]. The CoAP protocol depends on the UDP to decrease the processing needed by the device. The UDP is very lighter than TCP (implemented by HTTP protocol). Furthermore, the CoAP protocol is utilized to access web services. It transmits (receives) data to (from) the server. The MQTT protocol is typically utilized to send and receive data among devices. Since IoT device subscribes to the topics its interest in their texts, the protocol is considered a publish-subscribe protocol [22, 24].

Despite reducing the resources needed by the devices to enter their wanted services by these current protocols, the application can be used in services of the web only (using protocols, the access is provided). Additionally, recently the services number has been increased. However, the services provided by Representational State Transfer (REST) architecture and HTTP are more in comparison with lightweight services [25, 26]. Furthermore, whenever they attempt to access a service, the IoT devices usually communicate massive redundant data as they are primarily utilized for particular duties [3, 4]. Therefore, to proxy all internet services for IoT devices, this paper proposes a novel system by utilizing a cloud server.

Solving these questions requires efficient feature collection, selection, and extraction tooling. Unfortunately, many researchers don’t publish their tools, making it hard to reproduce and validate their results and increasing the time needed for future research on the topic. Furthermore, relying on crafted datasets for verification of network security monitoring does not reflect the actual situation inside the protected network environment.

1.4 RESEARCH CONTRIBUTIONS

The suggested system contributes to the following objectives:

1. Decreasing devices' need for resources by implementing lightweight protocols (e.g., MQTT and CoAP) to communicate with the devices.

2. Based on the protocols configured for each Application Programming Interface (API), interpret the messages between the intended services and the IoT devices.

3. Storing redundant data in the server to communicate the only sensed data by the device.

4. Extract the part of the data and send it without additional data as the need of device from wed service.

1.5 THESIS ORGANIZATION

This introduction explains the motivation for network security monitoring and anomaly-based detection strategies with clusters. It outlines the impact of memory corruption on today's networking infrastructure security within wireless sensor networks. Common terminology will be explained, along with a problem definition and a task description to define the scope of this thesis. After a brief discussion of related work, requirements for network feature collection mechanisms are analyzed, followed by an evaluation of existing solutions. Afterward, the concept and implementation of the research project will be outlined in-depth, and ideas for future improvements and several use cases beyond network proxying internet. In the final evaluation, several practical appliances of the developed tool are shown, including a series of experiments on classifying malicious behavior with a wireless sensor network. After the conclusion, a look ahead at possible future developments is given.

2. RELATED WORK

This thesis work is motivated by the goal of overcoming the challenges and difficulties that proxying internet system units in wireless sensor network faces with the help clusters techniques in real time with their structure and the environmental hostility surrounds them, that is by creating a better environment solution, a solution that will avoid the shortcomings of the traditional wireless networks for proxying internet system and controlling operations. For the sake of explaining the motivation of this thesis work, a detailed review of all these difficulties and constraints is discussed in depth in the sections below; furthermore, a comprehensive review of wireless sensor network system and their advantages that this thesis work aims to overcome.

2.1 LITERATURE REVIEW

To analyze, filter, and visualize network flow data, many network traffic flow capture techniques, tools and formats have been presented in [31]. The authors in [32] provided a comprehensive review of the network's sensor placement and topologies. Additionally, they studied network incident response and security monitoring. They demonstrated the use of some common tools. To determine network vulnerabilities and anomalies at each layer, several feature selection algorithms were proposed in [33] for some classification techniques. The authors discussed challenges, presented different tools, and assessed the systems of network anomaly detection

Using sensors and network mapping, the authors in [34] proposed techniques to perform experiments in collecting and analyzing the data collection, and analyzing traffic behavior, and data analysis. Furthermore, to visualize network data, the techniques provided several ways. Moreover, the techniques dealt with identifying the application. The authors in [35] explained the need for a network proxying internet system and described the network monitor’s operations and structure. This article inspired us to conduct this research because it contained the network security monitoring aspects.

The authors in [36] utilized the KDD dataset and used several filter algorithms to reduce forty-one features to sixteen one to achieve the same detection outcomes. They utilized Stability Selection, Stepwise Regression, and Maximum Relevance (WMR). The author validated their work using Support Vector Machines (SVM) and Bayes Classifier. The article motivated us to ask about the

need for overly complex extracted features. Furthermore, a more basic feature set is key to performing the current research tests.

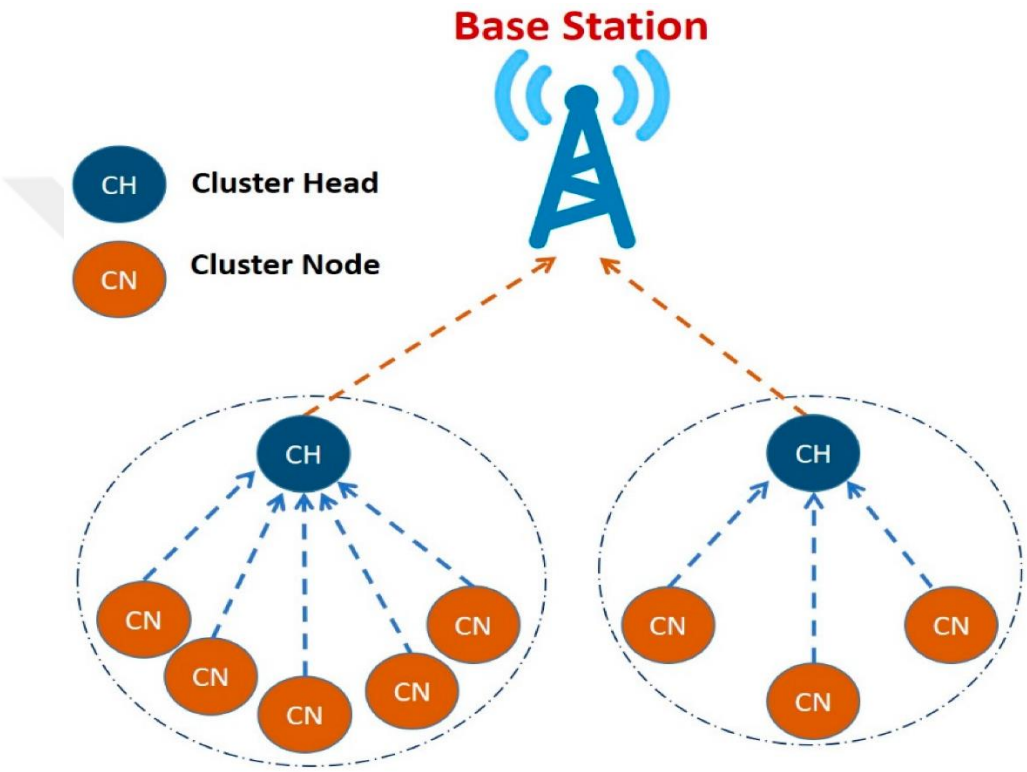


Figure 2.1: The basic schema of base station with two-cluster head and several cluster nodes [37].

Researchers in [38] presented their packet extraction tool for large volume network traces and compared the performance to existing solutions. Unfortunately, while their implementation could have been of great interest to the research community, they do not seem to have released it. The fact that their results could therefore not be evaluated, and that future research will need to continue with inefficient tooling, greatly encouraged me to publish my whole tool chain, in order to assist future researchers and allow independent verification of my results.

Researchers in [39] analyzed the state of the art for traffic flow analysis, and provided an overview and comparison of common tools and capture formats. It served as a useful introduction to flow based collection methodology.

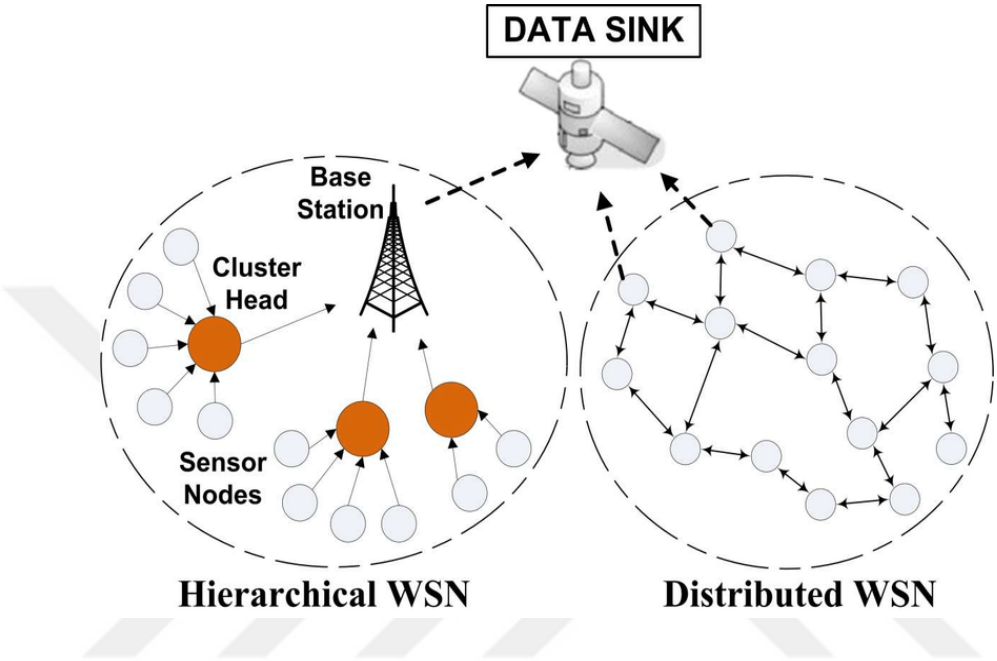


Figure 2.2: Conceptual model comparison between Hierarchical WSN and Distributed WSN [40].

2.2 FUNCTIONAL REQUIREMENT

The feature collector must be able to parse as many protocols as possible, in order to get the most detailed view on activity on the network. Additionally, the system must report on traffic that it does not understand.

2.2.1 Data Availability

For research purposes, the network monitor should make the collected data available at the lowest possible level, prior to generating summary structures. That means, protocol specific information should be offered for every protocol that can be interpreted by the monitor.

2.2.2 Abstraction Capabilities

Using the information that has been gathered in the collection stage, abstractions, summary structures, aggregate values and statistics can be generated from the collected data. A popular example for summary structures are unidirectional network flows and bidirectional connections.

2.2.3 Concurrent Design

After developing reduced-cost multi-core processors, many systems have been equipped with this design. In the beginning, it is important to consider the number of available processing cores in the design of a monitoring system.

According to [41], diving a problem into several different minor problems is known as concurrency. It gives chances to perform different parallel operations. Therefore, the group of independently performing processes describes the programming by the term concurrency. Accordingly, simultaneous execution of related operations is referred to as parallelism programming. In other words, parallelism is considered to perform many things at once, whereas concurrency deals with many things at once. Both are related but not identical, where the parallelism indicates execution, and the concurrency points to structure.

Through communication, independent executions should be arranged to enable concurrency. Network packets are received sequentially; therefore, it is no problem if they are read live from the network card or a dump file. The power of multi-core systems by parallelizing their decoding process is utilized to increase the entire processing speed. Finally, from the ground up, several systems are not developed with concurrent processing.

2.2.4 File Extraction Capabilities

Many protocols have been developed to transfer files in the literature, such as Bit-Torrent, SFTP, FTP, HTTPS, and HTTP. However, to carry file payloads, other protocols are abused. DNS protocol is an example of converting data exfiltration [42]. Without access to the utilized cryptographic or certificate key, inspection is impossible in encrypted communication (for example, HTTPS). For performing more analysis, there is a need to extract files transmitted in the network from a security monitoring side. For example, matching executables against anti-virus

(AV) databases Breaking encryption for this purpose should not be considered, because this weakens the overall system security, which is not a desirable effect.

2.2.5 Supported Input Formats

For input of offline network data, suitable formats are required to support it. Therefore, the industry standards, such as an improved version of PCAPNG and PCAP, are excellent candidates [43]. The PCAP preserves the full payloads and each single packet. Accordingly, for collecting data, it is a good input format. But, dump files' size may increase. It is a crucial problem, especially in the case of storing data over long periods. For this reason, other formats such as Cisco's NetFlow have been developed, that summarize traffic on a per flow basis without payload data, which greatly reduces size.

2.2.6 Suitable Output Formats

For the collected data, the output formate must be consumable by various frameworks, libraries, and systems and, in a human-readable format, be displayable. The CSV is often selected output format. Therefore, it is not limited to a particular character set and works just also with Unicode as with ASCII [44]. However, CSV does not preserve the character set currently in use, this must be communicated separately. Although humans are readable in their original form, CSV is not a space-efficient mechanism for exchanging or storing information. Additionally, CSV cannot represent hierarchical or object-oriented structured data, without separating sub structures from its parent context.

2.2.7 Real-Time Operation

It is important to support live capture from a network interface to monitor and operate real-time. To disc, dumping the collected data can not often be performed because of constraints of disc space (an example on embedded devices). Therefore, there must be another mechanism for data collection because capturing to disk is always optional. The following actions push proxyers to go faster in a compromised network: 1) Real-time uncovers breaches, and 2) Raise alerts. According to [45], incident response and digital forensics are required to increase the possibility of mistakes and give fewer periods for cleaning traces. Consequently, reducing the damages. Slow systems may produce alerts with a delay, then poxyers are given more periods to complete their objectives.

Real-time determination of attempted breaches makes valuable information be drawn from the network security about used techniques and targeted systems. Furthermore, it supports putting counter measures to take place.

2.3 NON-FUNCTIONAL REQUIREMENTS

In case untrusted input is parsed, it should be produced that no exploitable vulnerabilities are available in parser code. Many potential and different complex parsers have to be maintained and implemented because a large stack of protocols is the base of advanced network communication. Consequently, in the corresponding parser code, programming errors' chances may significantly increase.

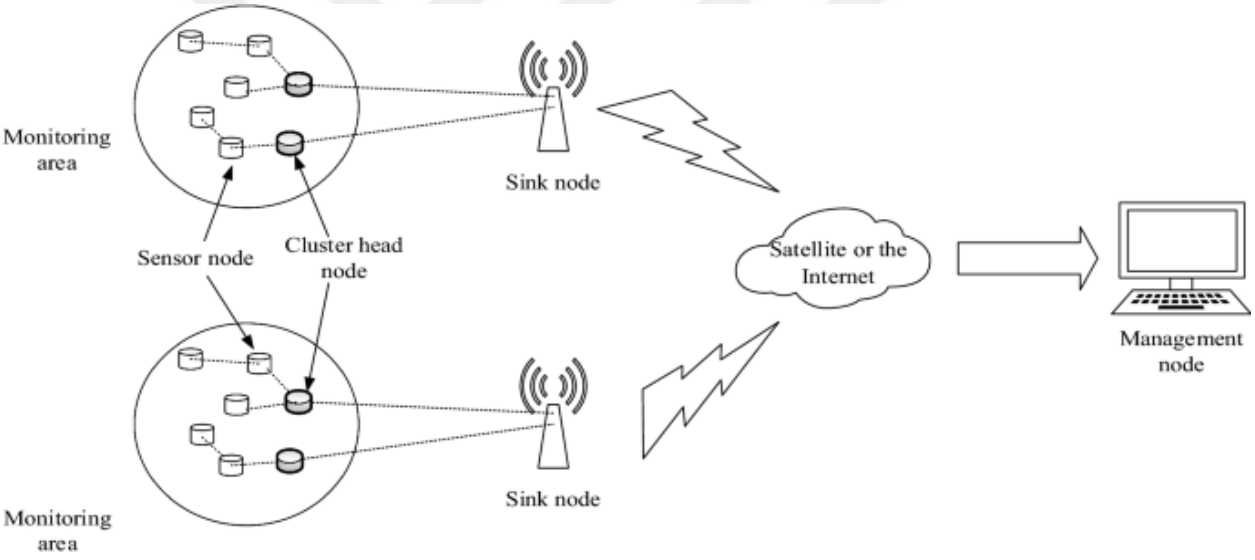


Figure 2.3: In a wireless sensor network, equality constrained optimization with management node and sensor nodes [46].

2.3.1 Memory Safety

Due to high costs, code audits are usually omitted or can only be done on the source code's fraction. Furthermore, protocols' RFC specifications are violated by vendors. It needs fault-tolerant parsers, and it should be verified fuzzing. Rather than secure memory handling, creating robust parsing logic is allowed through developing a system of a feature collection, and selecting a language with memory-safe garbage collected runtime.

2.3.2 Open Source Codebase

To support other scholars in extending or auditing the source code, the code must be well documented and accessible. Even though the code is available to a potential proxy, the network monitor's security model should not rely on secret code.

2.3.3 Scalability

To meet the needs of large-scale distributed deployments, the system must be scalable. For example, receive an input date from several network sensors. It must be performed efficiently and securely. Thus, compressing the data when it is transferred, and the communication needs to be encrypted by the network monitor to avoid eavesdropping

2.3.4 Performance

The number of packets (that must be processed and captured) continuously rises. The network monitor should be efficient to avoid losing anything and keep all the data. An alert does not automatically perform prevention of intrusion; however, its detection is a begnning, and the risk for an proxyer (to be found) is increased. An overloaded system that starts to drop packets, can lead to missing the discovery of an ongoing proxy. Performance depends heavily on the programming language used for the implementation. As mentioned before, low level system programming languages do not provide memory safety. Because of this, the usage higher level languages such as python programming for the design of security critical infrastructure, should be considered in the future. This introduces a performance penalty; however, this is a trade-off that comes with great benefits for the overall application security. A recently published paper implemented a POSIX compliant kernel in python, and measured a performance penalty ranging from 5-15% in comparison to the python implementation. Today's networks transport lots of huge traffic streams, for example video streaming such as netflix, torrent downloads and more. Because these flows cause a huge performance degradation for the monitoring system, a common practice is to detect those so called 'elephant flows' and discard them at the earliest stage possible as mentioned in [47]. However, this imposes the security risk of missing relevant information, an proxy could use the knowledge of certain traffic types being discarded to hide in a big data stream. Since audio and video players and their codecs are a common source for security vulnerabilities,

discarding these flows may lead to missing an actual intrusion, due to an exploited vulnerability in the media player. Searching the MITRE CVE database for the VLC media player for example returns 92 results as mentioned in [48]. An optimal implementation should not discard any data to meet performance goals.

2.3.5 Configurable Design

The monitor application needs to be configurable, to allow performance tweaking and adaption to special requirements, such as for example privacy regulations. Automated capture and analysis of network traffic can result in severe privacy violations, and may violate local, state and national laws. Therefore, it is necessary to obtain the required permissions for the target network as well as qualified legal advice, prior to installing any monitoring software. How to protect the user's privacy during a network security monitoring operation has become an important question for many network administrators. Since the 28th of May 2018, the General Data Protection Regulation (GDPR) further increases protection of personal data from citizens of the European Union, by drastically increasing the fines for violations as well as the scope of protected information as mentioned in [49].

2.3.6 Extensibility

Computer networks are highly dynamic environments and thus require the network monitor to be easily extensible, to integrate new protocols as fast as possible and keep up with the extremely fast development of technology. For this reason, the monitoring system should allow to add parsers for new protocols in a fast and secure manner, and should provide an interface to further extend functionality.

2.3.7 Reliability

The monitor must always be available and deliver meaningful results, even under proxy or in high load scenarios. Also, critical errors must not lead to a shutdown of the monitoring system, but should notify the administrator and rollback the system to working state as mentioned in [50]. Recovering from system failure, no matter if due to logic errors or due to a fault produced by an proxy, it is important to ensure continuous monitoring. Reporting and recovering on parsing errors or un-decodable traffic is mandatory, because the monitor will be subject to proxies.

2.3.8 Usability

The user interface of the network monitor should be powerful enough to accomplish complex tasks, but also be simple enough to be efficiently used by a human, without confusion or waste of time. Since network data is very abstract for humans, the software should provide proper visualization options, to assist humans in understanding, correlating and analyzing huge amounts of high dimensional data in a short period of time.

2.3.9 Storage Efficiency

Storage is limited and the continuously increasing amount of data that modern networks are transporting increases the amount of collected audit data. Therefore, the generated output should be as small as possible, while preserving all information relevant for the detection task as mentioned in [51]. Additionally, it should be configurable what data will be collected. In some scenarios there might be data that is not relevant for the detection task or a concrete monitoring system. If this is the case, no resources should be wasted on collecting or storing irrelevant data.

2.4 FEATURE COLLECTION IN IDS SYSTEM

Feature collection systems for proxying internet should run continually without or with minimal human supervision, and should be reliable enough to run in the background of the observed system and impose minimal overhead. The system must be fault tolerant and able to recover from errors automatically, and should preserve its own integrity in order to detect proxies against the monitor itself. Also important are aspects like re-configurability and extensibility, since dynamic environments such as computer networks often require adaptations. Deployment should be easy and fast and the system must be able to observe deviations from normal behavior. Additionally, they must provide an efficient user interface and provide visualization to assist human analysts. The named requirements for the underlying hardware also play a vital role in ensuring the monitoring systems integrity and performance. For future reference, all software requirements for modern network proxying internet systems have been listed in table 2.1.

Table 2.1: Summary of feature collection requirements [52].

Requirement	Short Description
Protocol Support Coverage	Range of supported protocols
Data Availability	Access to data on the lowest level
Abstraction Capabilities	Generation of abstractions such as flows
Concurrent Design	Efficient use of multi-core architectures
File Extraction	Extraction of files from network traffic
Suitable Output Format	Widely supported and efficient output format(s)
Real Time Operation	Live acquisition of traffic from a network interface
Supported Input Formats	Support for industry standard input formats
Memory Safety	Secure memory management
Open Source Codebase	Publicly available source code
Scalability	Operation in large scale, possibly distributed networks
Performance	Efficient processing for high volume traffic
Configurable Design	Configuration and adaption options for special requirements
Extensibility	Ease of extension for new protocols
Reliability	Reliable implementation
Usability	Ease of use for the analyst / researcher
Storage Efficiency	Low storage overhead

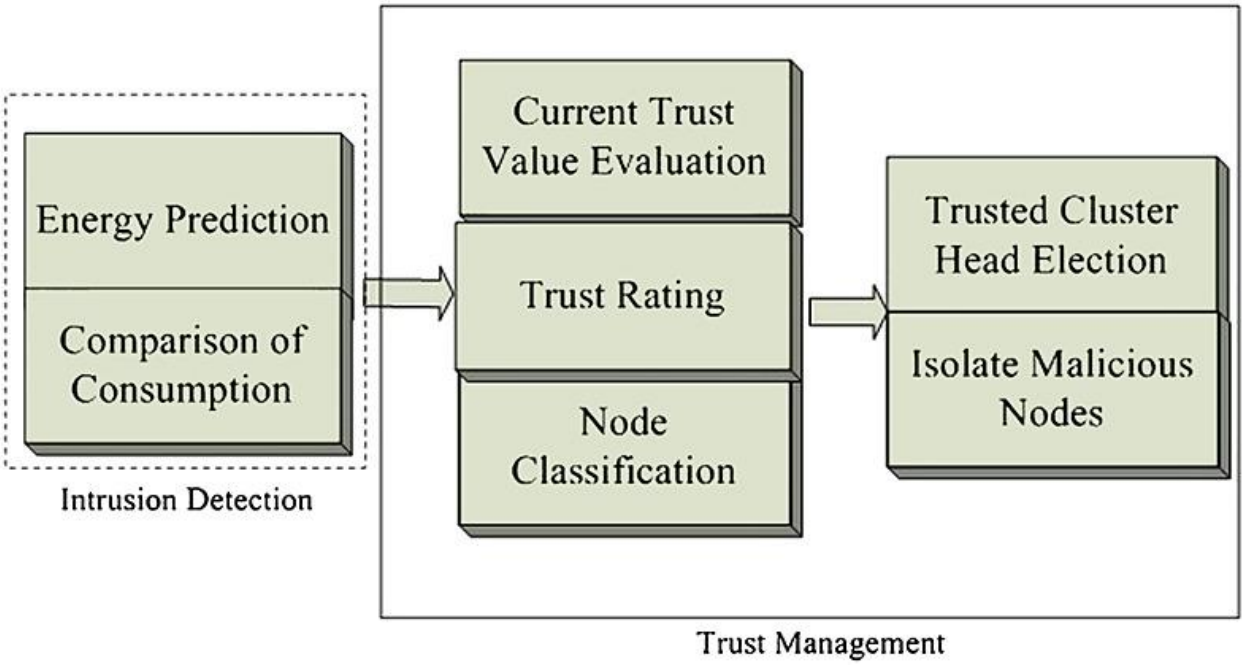


Figure 2.4: With two modules in the security scheme, network re-configurability, and extensibility for proxying internet [53].

2.5 BENEFITS OF INTERNET-OF-THINGS

In 2019 the quantity of Internet of Things (IoT) associated gadgets was 19.71 billion around the world. Before the finish of 2020 it will hit 26.66 billion as referenced in [54]. In 2025, the quantity of gadgets associated with the Internet will be 75.44 billion, making the number ascent multiple times in 10 years as referenced in [55]. These shrewd gadgets associated with the Internet are fit for creating information on its own dependent on conduct or expected outcome and offer it over the Internet. This is the thing that comprises the idea of Internet of Things (IoT). IoT is a gigantic gathering of gadgets containing sensors or actuators associated together over wired or remote organizations as referenced in [56]

Other than all the advantages and positive attributes that the remote medium has brought to IoT, there are weaknesses. There are various types of traffic inconsistencies and intermediaries. The most well-known security dangers are interlopers, which is by and large alluded, as programmer or wafer and the other is infection. This accompanies the danger of individual information being communicated to the world and cause more digital intermediaries like phishing, disavowal of-administration (DoS), Probe, Remote to Local (R2L) intermediary and so on Since these intermediaries are not normal so ordinary information stream has an example which can be pretty much the equivalent [57]. Be that as it may, when another substance will attempt to mutilate or change the information, the example will change and this is the place where the Intrusion location methods becomes an integral factor with regards to web and explicitly with regards to Internet of Things.

To guarantee the framework and information are simply open to approved substances like people and cycles, intelligent controls are utilized as programming safeguards. Interruption recognition and counteraction framework, passwords, access control and so on incorporate sensible controls as referenced in [58]. To screen and identify oddities or any dubious conduct, interruption discovery framework (IDS) and interruption anticipation framework (IPS) are being utilized. As various conditions and most recent advances are inclined to be malignantly intermediary, AI (ML) calculations can recognize, break down and order interlopers in network precisely and quickly as referenced in [59]. Two sorts of interruption recognition framework are there. They are: Anomaly based identification and Signature based recognition. Abnormality based recognition strategies distinguishes the intermediaries that are obscure by noticing the entire framework and the elements

in it like traffic, objects and so on Something besides ordinary conduct is recognized as a possible intermediary. Then again to distinguish explicit examples of the intermediaries that are known by exploring network information or traffic, signature based recognition techniques are incredible as referenced in [60]. Subsequent to exploring on explicit use cases, information attributes and framework needs, such a variety of qualities has affirmed that there is no exceptional technique that would give effective and exact interruption location for each dataset under the examination as referenced in [61].

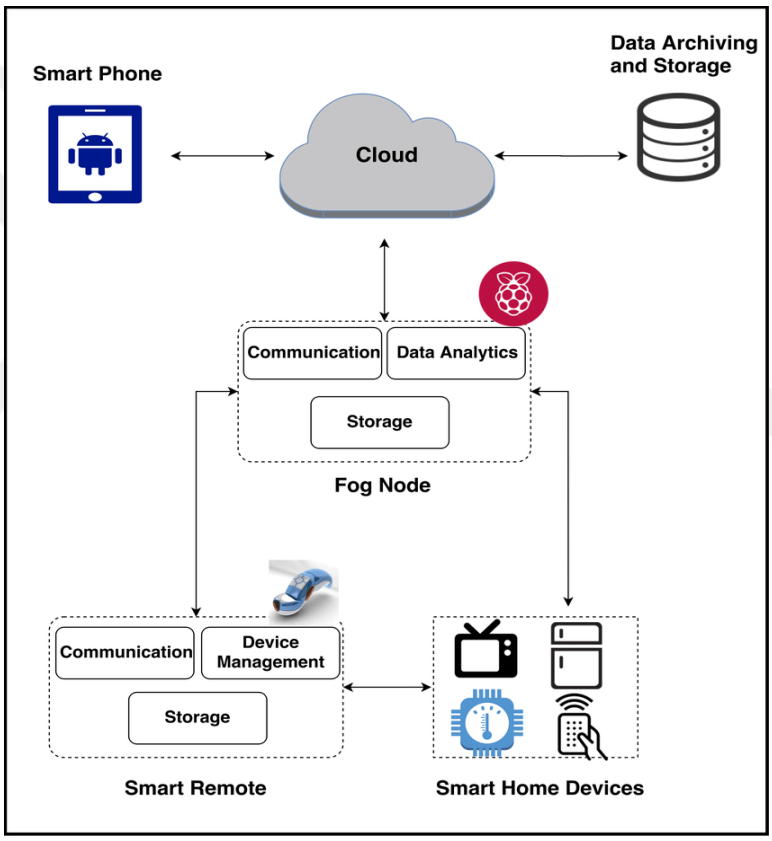


Figure 2.5: A model of intelligent proxy transformation with data archiving and storage management system inside the proxy [61].

2.6 IOT SUPPORT IN INTELLIGENT PROXY AUTOMATION

2.6.1 Advantages of Internet of Things (IoT)

IoT provides several day-to-day advantages not only for consumer's sector but for business sector as well as mentioned in [62]. This technology allows us to view real time information that was not

available before. Business can improve their production efficiency by reducing material waste and unforeseen downtime. Since Artificial Intelligence is used, it saves considerable human effort and time. Sensors can be used to build infrastructure, detect damage to infrastructure. Automated traffic control can help reduce road congestion and gadgets or sensors can be implemented on the outside to recognize change in the environment and warn us for any impending natural calamity.

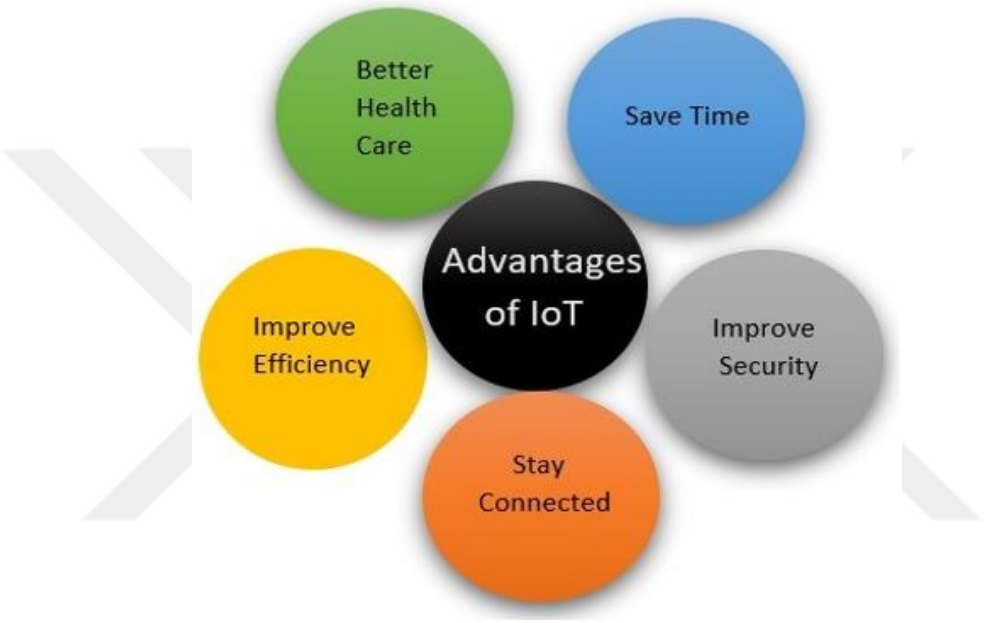


Figure 2.6: Advantages of IoT described. Source [62]

IoT makes our proxy, offices and vehicles smarter, more measurable and this in turn makes our life more pleasing. Smart speakers like Amazon's echo or Google proxy makes it very easy to set timers, get information on the go or be it to play music. Through proxy security system, we can monitor our proxy from other places remotely and can see what is happening on the inside and outside or talk to visitors. Smart lighting systems can reduce unnecessary energy consumption as it stays off if there is no one in the house or smart air conditioners will turn on when it knows you are on your way to your proxy.

2.6.2 Disadvantages of Internet of Things (IoT)

Security in IoT is a double edged sword. Now that we have a system of interconnected devices on a network it makes the system somewhat secure and efficient as mentioned in [63]. Though connected devices generate colossal amount of data and are constantly being exchanged between

devices so it in turn generates lots of internet traffic. These data can make the devices useful but this easily accessible nature of the internet brings up both security and privacy issues. These issues arise for many reasons.

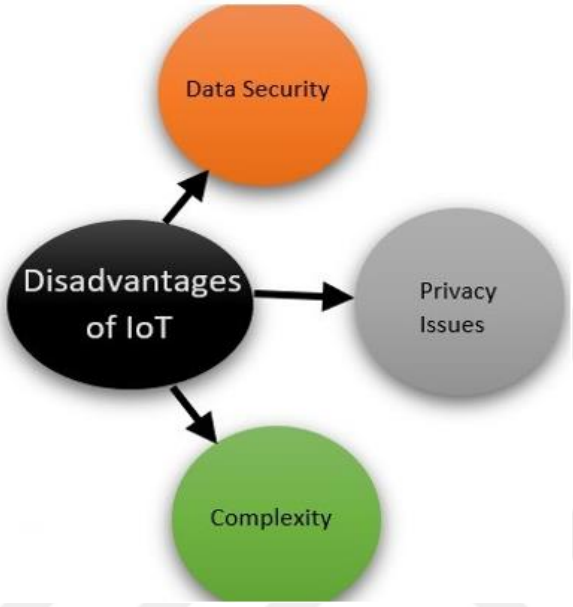


Figure 2.7: Disadvantages of IoT described. Source [62]

Imperfections in programming is found oftentimes due to the way that IoT gadgets do not have the capacity to be fixed or fixed. This makes them inclined to security hazards like DDoS intermediaries, perhaps the most well-known issue, which occurs by setting the gadget's secret key to its default secret phrase which are effectively break capable by programmers as referenced in [64]. Ransomware and Malware both depends on encryption to lock out the client's gadgets totally and takes the client's information. Protection stays among the biggest issues of IoT as information is being sent, put away and handled and being badgering by huge organizations as referenced in [65]. This very data is then being sold to various other companies which is violating our rights for privacy and data security. Proxy Invasion, one of the scariest threats that IoT can possess as most of the proxies and offices rely on automatic processes that leads to the use of IoT devices in these places. IP address of these devices are exposed to hackers who can use it to pinpoint the location and gain access to our personal data just to sell them to underground sites which are the hives of criminal activities.

2.6.3 IoT-Disconnection in Smart-Proxy

Internet of things disconnection happens in smart proxies due to many reasons. It could be because of a faulty node. Other reasons could be due to the movement of nodes, which could be a random walk to single or both nodes of each link or could be periodic due to satellites and planets movements on orbits. It is good to mention also that some cut offs occur due to the nodes periodic power saving, like low power nodes in sensors networks.

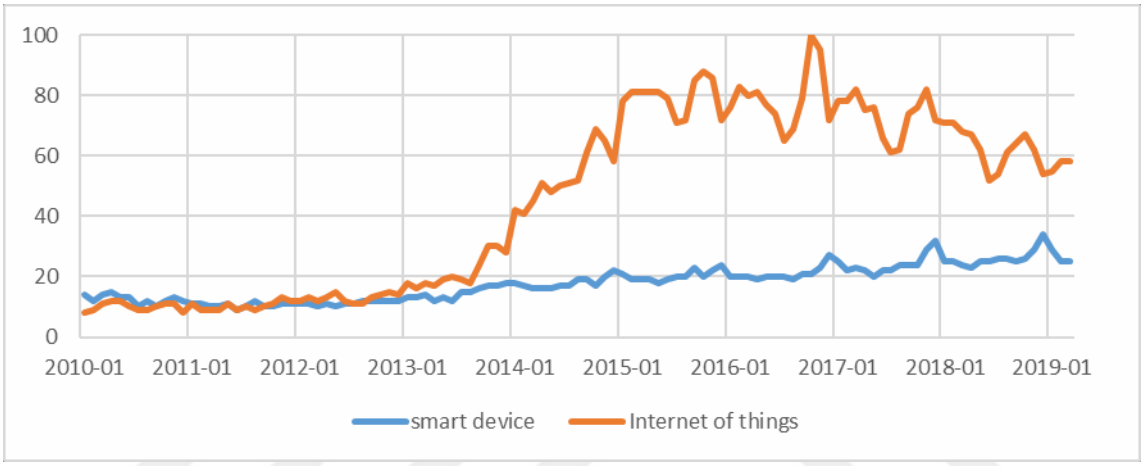


Figure 2.8: Interest over time according to Google trends since 2010 to 2019 for terms Smart device and Internet of Things [66].

2.6.4 Unexpected Lengthened Queuing in IoT Networks

In conventional IoT networks, queueing delays could be parts of seconds to seconds in extreme cases. But due to the abnormal circumstances of smart proxy queueing delays could reach days if not hours. What contributes to those longer queueing delays is the nature of smart proxies where many copies are traversing among the nodes, not to mention the retransmission that takes place when data cannot reach their destinations. Intermediary Automation is like Samsung SmartThings, however is created by Belkin Company. It gives a full shrewd intermediary framework, with out-of-the-crate savvy gadgets and applications (portable and online) to control and deal with the brilliant intermediary framework. Its engineering is marginally not quite the same as Samsung SmartThings, on the grounds that the gadgets are overseen straightforwardly from the applications, without a mediator center, since they are WiFi empowered. The framework utilizes the WiFi switch to make a shrewd intermediary organization. To control from outside the intermediary, it is important to add a center point, such a Raspberry Pi as referenced in [67]. Like Samsung

SmartThings, Belkin WeMo Proxy Automation framework doesn't give adaptability to add specially crafted gadgets, without adding a WeMo item (WEMO Maker) to associate with any remaining gadgets, regardless of whether the clients make gadgets that impart through WiFi as referenced in [68].

The setting of the web of things utilization on the planet. The urban communities count with a worldwide populace socioeconomics and assumes a significant part inside the worldwide financial. It has a place with the gathering of megacities with a populace of 10.5 million by 2025 as referenced in [69]. The situation could be either fortunate or unfortunate relying upon the whether megacities are considered as a positive or negative thing for the plan on reasonable turn of events. By endeavoring to associate "nations with the largest number of megacities" with the "World Environmental Performance Index" it tends to be gathered that there is no relationship between's nations with high level of populace living in megacities and natural manageability as referenced in [70].

3. METHODOLOGY

The components of the suggested system are as follows: network service, database, and web application. As illustrated in Figure 3.1, to achieve tasks of the IoT devices, each component communicates with another (among all components). The web application is used by IoT developers in order to design and configure the Apps that IoT devices access. Locally, an app operates and accesses outside services on the internet or even combines both. By the web application, the configurations can be stored in the server (the database management server). To get the needed task, the server is accessed whenever the network service needs it. Furthermore, the database management server stores redundant information that needs to perform a task. Therefore, to decrease the needed bandwidth to execute the App, the information is not communicated from the device.

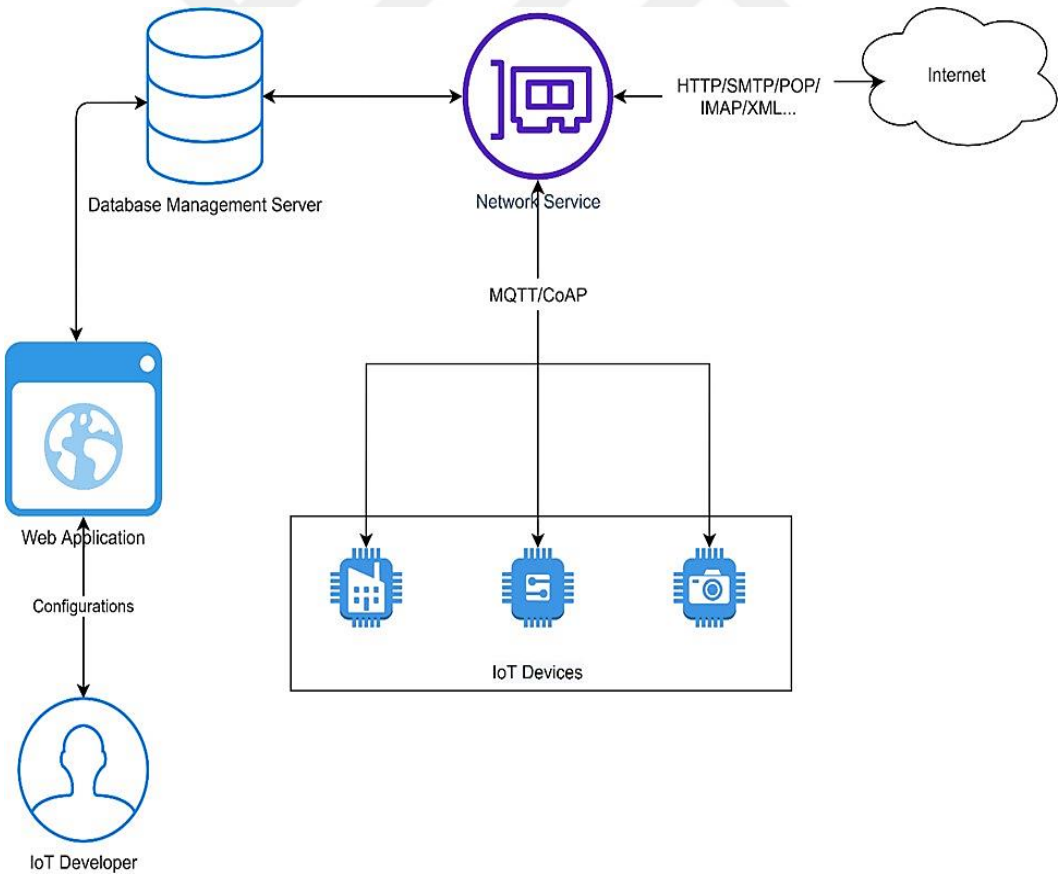


Figure 3.1: Hierarchy of the proposed system.

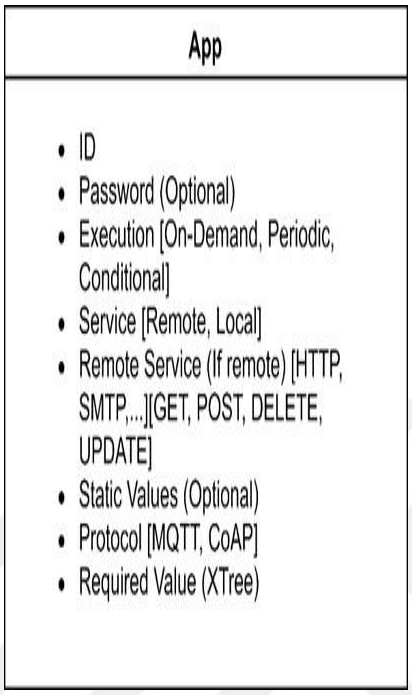


Figure 3.2: App configurations of the suggested system.

3.1 STEP OF DESIGNING AND CONFIGURATION OF APP THE APP

The system assigns a unique identifier (ID) and optional authentication password (see Figure 2). Public App is the App that does not have any password, and thus any device can access it without any authentication. Using the web interface, the developer predefines these types of Apps, which are designed particularly for a specific task. Manually, the App is triggered according to the configuration. When a predefined condition is satisfied, the device requests the triggering by using a preconfigured interval (conditionally or periodically). Using the MQTT protocol, these Apps can be accessed, and their values are updated. The App updates assigned by the device (which has subscribed to the topic) lead to automatically updating the device if Else, using the CoAP protocol, is used to access the APP when triggered and configured. Therefore execution of the App is performed, and the outcome is sent back to the device requesting the App.

Using HTTP protocol to access a service, an App has DELETE, UPDATE, POST, or GET commands based on REST architecture. Furthermore, on the remote server that provides the intended service, the App consists of any data needed to execute the command. At a particular server that needs authentication, the device requires to write a specific value. The name of the value being written and the sensed value. All the rest information can be prestored in the system with the sensed value. Therefore. the remote service can be accessed. Then, once the measured value is delivered, the sensed value is written by the device to the suggested system. In this case, the CoAP protocol is used. Lastly, XTree is compared with the response retrieved from the remote service. In the received XML response or HTTP, XTree defines the required value's position so that only that value is returned to the IoT devices. The response is returned to the device requesting the service in case of no tree is specified

3.2 ACCESSING THE APPS

To access an App, the device uses the protocol according to the App's execution kind. While, the CoAP is utilized on-demand ones, MQTT is utilized for conditional and periodic Apps.

As mentioned earlier, the protocol that can be used by the IoT device to access an app depends on the execution type of the App, where MQTT is used for periodic and conditional Apps and CoAP is used for on demand ones. By subscribing to the topic of the required task, which is defined by its ID, an IoT device can receive updates, about the value it is interested in, whenever a change is detected in its value using the MQTT protocol. Alternatively, an on-demand App is accessed using the CoAP protocol by providing all the dynamic values required from the IoT device to access the remote service, if any is de- fined in the App configurations. These values are appended to the template defined at the server at is used to access the remote service, where the response of the service is compared against the configured XTree in order to extract the required piece of information and send it back to the IoT device.

In a vast area, Wireless sensor networks (WSN) are typically utilized to assess several physical objectives. It contains many sensors (small nodes). They are used to measure the physical features of the location, such as air pollution, pressure, and temperature. WSNs implement in several areas, such as monitoring natural disasters like volcanic, earthquakes, and floods. They are utilized in medical, chemical, automated warehouses, robotics, and military applications. In nature, the WSNs

are widely used to measure mountain disasters, humidity, pressure, and temperature. Sensors are small devices and have limited memory and power. The system receives and transmits information among the sensors and some components to measure area characteristics. Identifying the position of sensors is crucial in WSNs applications. Typically, there is a limitation in using sensor data without the precise location of the sensors. Knowing the precise location of some sensors is essential in several applications. Therefore, the remaining network topology is entirely ad-hoc and random. In the literature, several survey papers have been published reviewing MQTT-related issues. Usually, the scholars investigated and addressed the per-node location information. Since our framework has a client authorization component in the home server, all the customers should have an interesting identifier that match with the one characterized in the client consents, if not they will be allocated with the default client consents. The authorizations could incorporate access privileges to explicit gadgets or to gatherings of gadgets, for example it is feasible to characterize that a client can get to the upsides of all the kitchen gadgets yet can't transform them. As of late we hear like never about web of things (IoT), fake power perception, and home computerization. These terms are comparable by they way they sound, yet they are distinctive in alternate ways. This part will give the execution subtleties IoT based wise home mechanization frameworks by and large, and that incorporates, MQTT convention for taking care of the robotization interaction with home server modules and for this reason wellspring of programming language being utilized is python programming 3.8 on boa constrictor pilot stage with both CPU and GPU usage. The basic flow diagram has been shown in Figure 3.3.

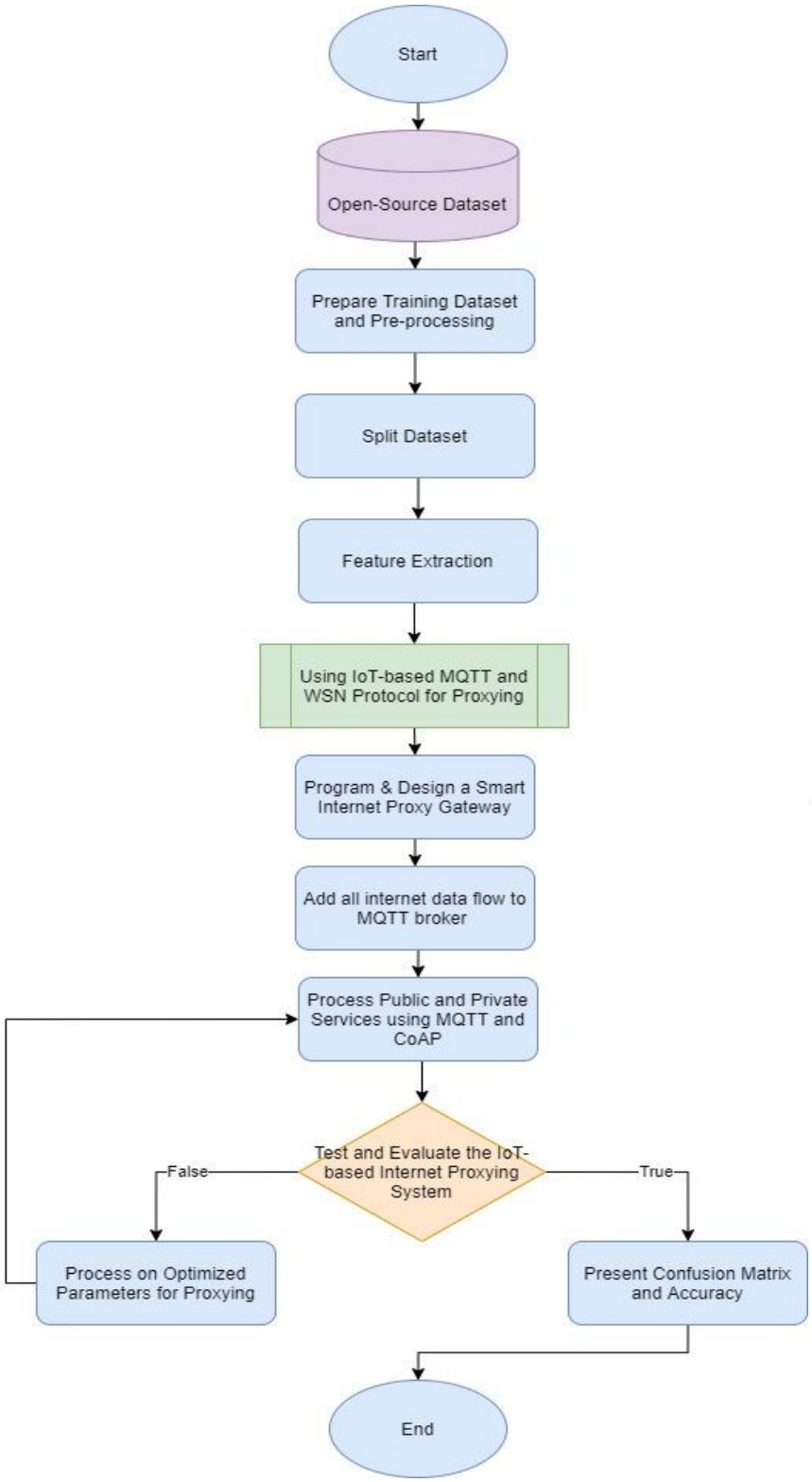


Figure 3.3: The flow diagram of proxying internet system in internet-of-things.

3.3 WIRELESS SENSOR NETWORK

We distinguish WSNs from other conventional or wired even wireless networks through sensor based interaction with the environment. WSNs typically possess little (or sometimes no) infrastructure. They consist of a number of sensor nodes which can range from few tens to thousands. These sensors work together to monitor and measure a specific region to obtain information about the environment. In most cases, we are interested in the usage of inexpensive sensors, which intrinsically have limited processing components. This chapter focuses on classification and evaluation of the approaches proposed in the literature for the localization processes for proxying internet system within wireless sensor network. They can be classified in different categories based on various criteria. In the thesis, we consider the problem as distributed systems problem. We also assume the direct connection may not be established across the whole network. Thus, one of the most significant categorization with regard to hardware limitation.

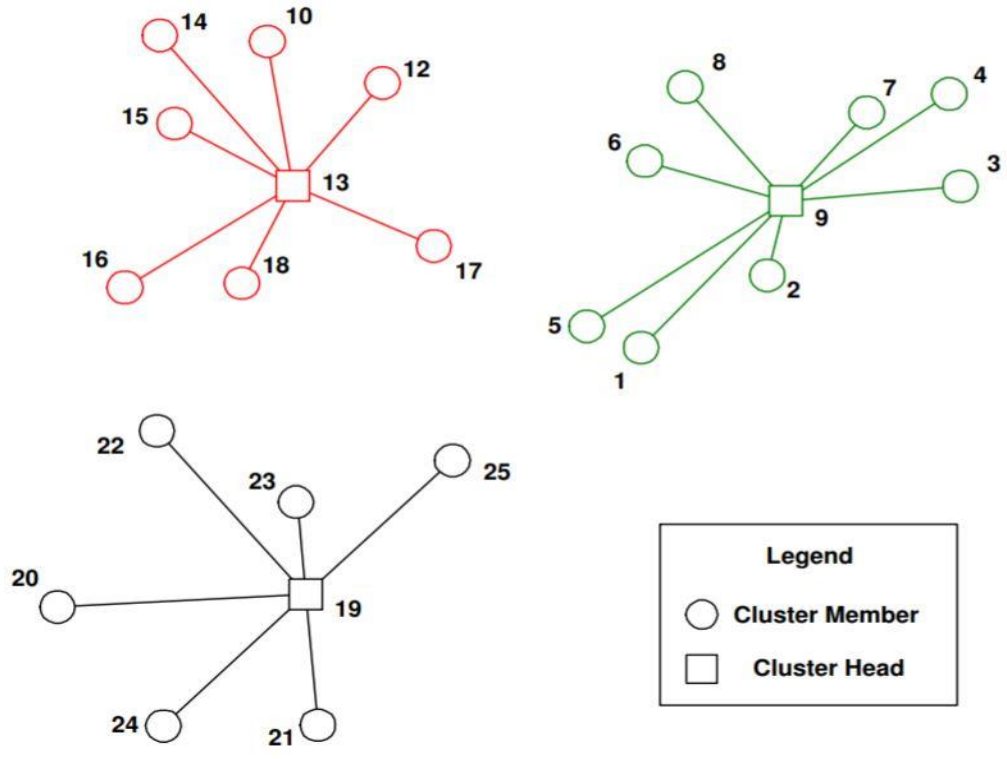


Figure 3.4: Topology of wireless sensor network after organization into cluster.

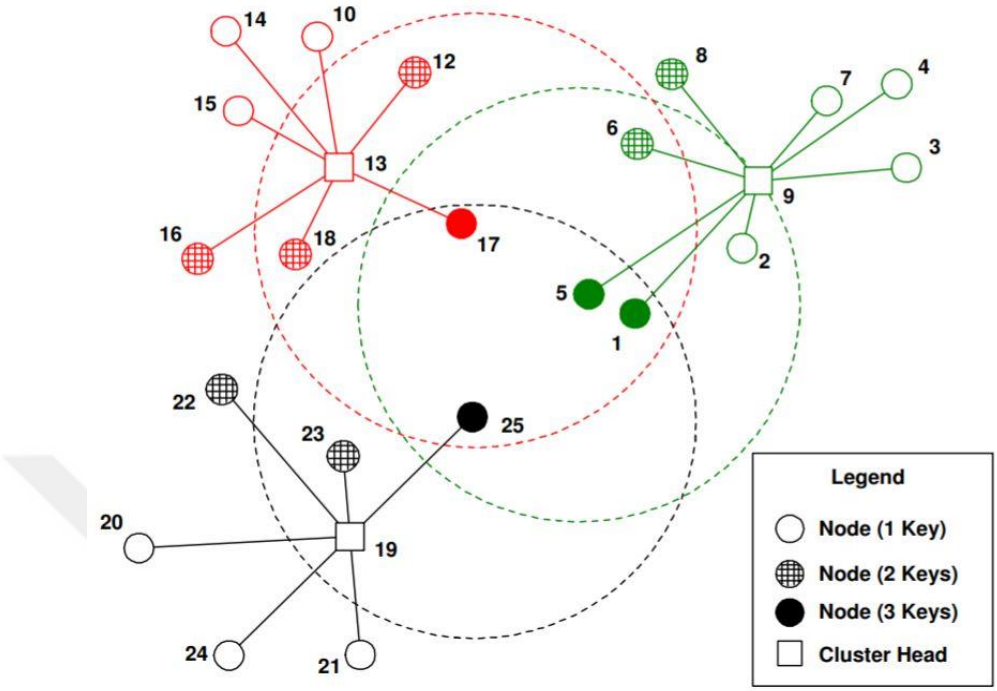


Figure 3.5: Topology of wireless sensor network during the key setup phase.

3.4 DATASET DESCRIPTION

Anomaly-based proxying internet approaches suffer from the lack of reliable datasets for testing and verification. Evaluations of the eleven datasets that exist since 2017, conducted by the Canadian Institute of Cybersecurity, have shown that most datasets are out of date and unreliable for evaluation purposes. While some of these datasets fail to provide sufficient diversity and volume of network traffic, some do not contain latest proxy patterns or anonymize packet payload data, which limits research capabilities. Additionally, some are also lacking feature set and metadata information. An interesting approach to- wards the creation of realistic datasets was presented in 2015, when researchers released their framework for injecting proxies into existing datasets. Although their tool was not used in this research, it is a promising option for future experiments. The CIC-IDS-2017 dataset has been chosen as the most up to date and most extensively documented dataset, and will be used for the experiments [54]. The dataset consists of CSV flow records generated with CICFlowMeter, and the PCAP files used for creation. Data was captured for a total of five days and no anonymization has been performed on the dataset. The total size of all included PCAP dumpfiles is 48.9 GB. Proxies have been carried out on the working

days Tuesday, Wednesday, Thursday and Friday, in both morning and afternoon. The implemented proxies include DoS & DDoS (Wednesday), Heartbleed (Wednesday), Web Proxies (Thursday), FTP Brute Force & SSH Brute Force (Tuesday), Infiltration (Thursday), Botnet traffic and Port Scans (Friday). Monday is the normal day and includes only legitimate traffic. Detailed information on the dataset and the testbed architecture can be found in the corresponding research paper and on the CIC website.

3.5 RANGE-BASED LOCALIZATION IN MQTT

These methods use the point-to-point distance or angle information. The most popular techniques for estimating the distance and the angle between two nodes are:

3.5.1 Received Signal Strength Indicators (RSSI)

This technique relies on comparing the strength or the power of the received signal with the one sent at the transmitter node. This approach works with RF signals, it is relatively cheaper (no additional devices) and used more than the other techniques that will be described. It is also simple and inexpensive in the configuration and calibration process. However, one of its most important drawbacks is its sensitivity to noise. The distance estimation accuracy can be very bad specially for large distances.

3.5.2 Angle of Arrival (AoA)

As the name suggests, the technique is used to find the angle of the emitted signal from the source. This approach normally needs at least two antennas for each node. This requirement limits its usability in the networks with small and low-cost sensors.

3.5.3 Time of Arrival (ToA)

One of the classical methods to localize the indoor environment is ToA. The speed of the radio signal is equal to the speed of light (in the cases where audio signals are used for localization, we consider the sound speed instead of the light speed). If we are able to find the time that the signal has traveled, by dividing it with the speed of light we can estimate the distance between the sender and the receiver.

One of the most important issues to be considered in this technique is the time synchronization between the nodes [71]. One of the ways to overcome synchronization problems in acoustic ToA estimation is to use two types of signals. The sender emits the radio signal and ultrasound propagation at the same time. The receiver receives the radio signal and starts the timer. When it receives the ultrasound signal it stops the timer. Thus it can compute the difference between them. This method achieves the high ranging precision but it needs additional hardware and consumes more energy.

3.5.4 Time Difference of Arrival (TDoA)

The measurement procedure for this technique is technically the same as the ToA method, but the relative time of arrivals between nodes are used to estimate the locations. By using the TDoA, we eliminate the need to know the exact signal emission time at the source [72].

3.6 RANGE-FREE LOCALIZATION IN MQTT

These schemes are applied as cost-effective alternative to the more expensive Range-Based approaches [73]. In these approaches, there is no information about the distance and the angle of paths between the nodes. From an algorithm architecture perspective, we divide the available methods in two blocks:

3.6.1 Centralized Architectures

We select a node as a monitor for several nodes. It should be a powerful central base station for others. The selected node works as a coordinator for neighbors. The advantages of these algorithms are the easy implementation and the elimination of the computation process in each node. But if we lose the monitor node, we will lose a part of the region.

3.6.2 Decentralized Architectures

In the traditional centralized algorithms, we need some hubs and the data may be difficult to obtain. In decentralized algorithms the region is divided in many clusters and in each of them the transmission is possible between all nodes together. Thus, compared to the former structure, we have more reliability.

Adding a GPS device to each node in large-scale ad-hoc networks, may increases the price and the size of the sensors [74]. As well, it affects the consumption of the power and reduce the network lifetime and its performance. Thus it may be not a good idea to add a space-based satellite navigation system that can provide the location to each node. This consideration also divides the approaches in WSN into two main categories

3.6.3 Anchor Based

In many approaches we can obtain the exact position of only a few nodes. Sometimes it is possible by using GPS. As well, we may know the ab- solute position of a node in the region. These nodes are called anchor or beacon node.

3.6.4 Anchor Free

Based on the cost, consumption and environment constraint we might not be able (or not interested) to have sensors with known exact geographical location. In other words, there are no anchor nodes in the network. In these cases, localization schemes based on distance information are only able to find the relative positions of the sensors in the network [75].

In the real environment, we may get incorrect data from the nodes or we may lose some nodes. Thus there is a random and unwanted variable for each node that can affect the solution. The random variable modeling the error is independent for each node.

There is also another category for network localization algorithm. This category is divided the methods in two main groups of methods based on the granularity of data gotten by the sensors.

3.6.5 Fine-Grained Algorithms

These methods use accurate information during the communication such as the distance from a reference node determined using RSSI or ToA measurements. In general, this type of algorithms uses several technologies, such as ultrasound, infrared, or radio frequency signals. In the absence of measurement errors, fine-grained algorithms provide exact network nodes positioning.

3.6.6 Coarse-Grained Algorithms

Basically, these algorithms use less accurate in- formation for the localization of unknown nodes. These algorithms always, estimate the distances between the sensors in the network with some techniques such as hop-count. Figure 3.4 shows the possible categories to divide the localization algorithms in MQTT.

Especially in the range-based proposals the noise is typically dependent on the devices and interfaces. Thus many of them work only in isolated environment.

- They try to estimate the distance or angle of two nodes.
- They try to combine the achieved distances or angles to estimate the locations of the sensors.

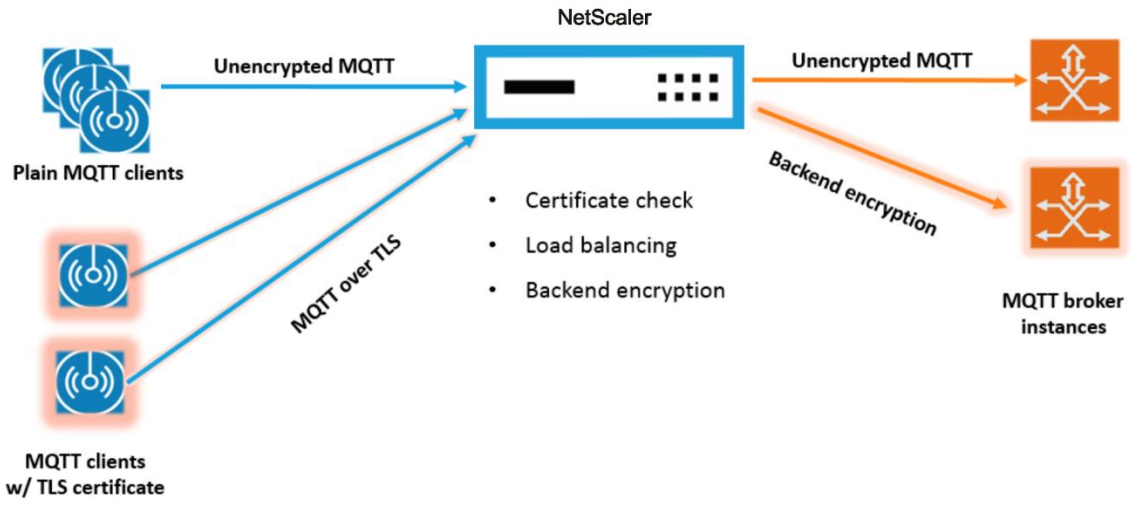


Figure 3.6: Classification of the approaches in MQTT broker.

There are six metrics which determine the high level of performance in localization algorithms:

1. Accuracy: The algorithms try to find the closest value of measurement to the actual value. Thus accuracy or precision is the most important factor in the localization techniques.

2. Coverage: Each node has several neighbors. It can have transmission with them. Of course, the nodes can exchange data on short distances. Each connection technology operates in a special range. The effective coverage varies due to many criteria such as propagation conditions, antenna,

battery power and etc. So for achieving the better coverage and better performance we need an adequate density of nodes and the number of anchor nodes.

3. Complexity: In networks where the ranging information is sent between the sensors (range-based) the devices need to have more complex hardware com- pared to networks which rely only on the connectivity information (range-free). The former architecture of the network is more complex.

4. Scalability: The ideal algorithm has the ability to be enlarged. It is one of the properties of each localization algorithm. It should be suitable to apply for larger situations. In other words, when the network gets larger, the algorithm does not need to change.

5. Robustness: In the outdoor environment all signals are noisy with a random error, and we expect to not unduly affect the performance of algorithm [76]. Perhaps, we lost some nodes and some parts of the networks. In the range based algorithm, we always work with the angles, distances, and time. Thus, if we do not apply a the reliable and robust algorithm we will face a lot of errors at the end.

6. Cost: One of the most important criteria in the wireless sensor networks is the price of each sensor. With a small change in hardware and software of nodes we may save or lose a lot of money. One of the factors that effects on the price is energy consummation. If the sensors consume less amount of energy, we could cut the cost. On the other hand, if our algorithm can find the position of sensors rapidly we do not need long life battery [77].

The program code has a capacity, that just runs during the gadget introduction of wise intermediary change. Hence, in this clever intermediary work our gadget in states and sets the underlying qualities related with the parts sticks, the WiFi network properties, the intermediary server address and the MQTT agent address. Then, at that point, it sends the enrollment messages to the intermediary server, utilizing WiFi, to enlist itself and all of its parts and properties. At last, the gadget associates with the MQTT agent and buys in the themes related with its parts. The capacity, as its name recommends, is a capacity that will be executed ceaselessly until the board is turned off or reset, permitting our program to change and react in every emphasis. We have executed in this capacity the MQTT convention circle that checks, for every primary circle cycle, assuming

there are new MQTT messages from the preferred IoT-based MQTT representative for the execution of intermediary computerization framework.

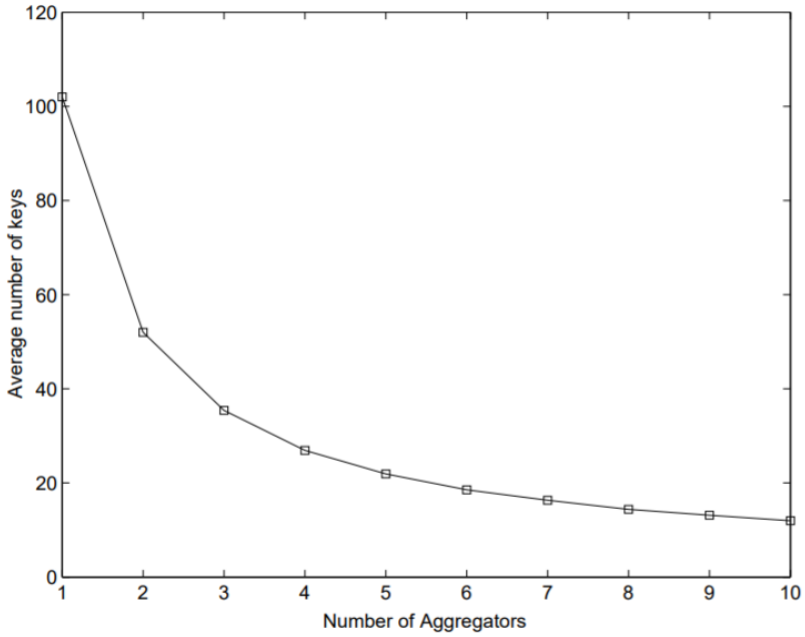


Figure 3.7: The 100 keys used as an aggregator for nodes in MQTT.

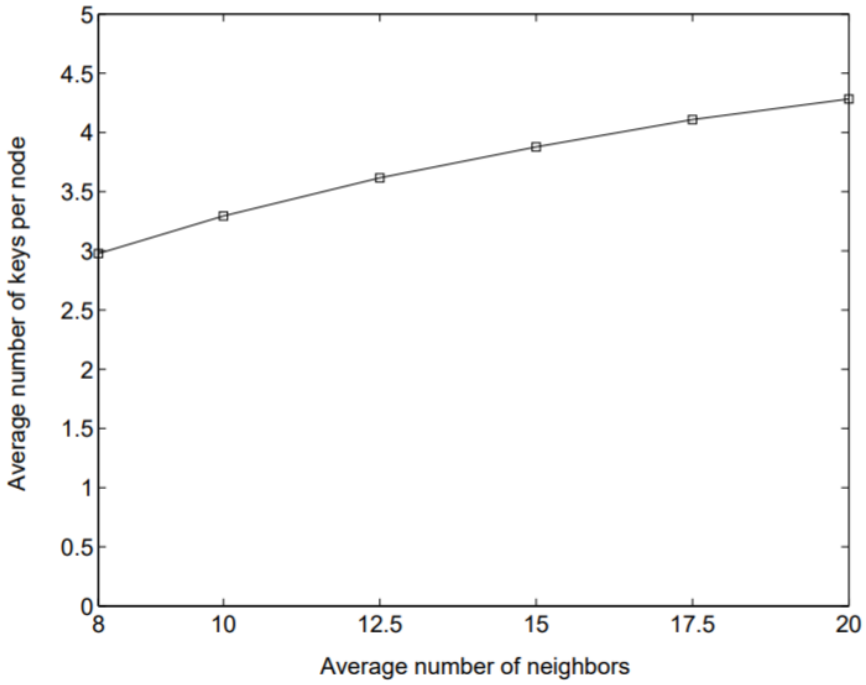


Figure 3.8: The keys held by sensor nodes for 20 neighbors in the MQTT.

3.7 IMPLEMENTATION

In the early ninetieth of last century, the python software foundation released Python as a compiled imperative and statistically kind systems programming language [78]. It is syntactically similar to Go but borrowed numerous features from other languages to boost readability and efficiency. Python is renowned for its incredibly rapid compilation time and creation of statically linked binaries, which are independent of any libraries in the execution environment. The Communicating Sequential Processes (CSP) study influenced Python's current version, 3.8, which includes a concurrent execution and coordination model of asynchronous processes. A go procedure is an asynchronous process, not to be confused with an operating system (OS) thread. MQTT in Python is multiplexed onto OS threads as necessary. If a WSN block occurs, the relevant OS thread also blocks, but no other blocks are impacted. MQTTs are less computationally costly than threads and dynamically allocate resources as required. Python's channels provide lightweight inter-WSN communication for synchronization and messaging. This design choice was motivated by the notion of memory sharing via communication, as opposed to memory sharing through communication.

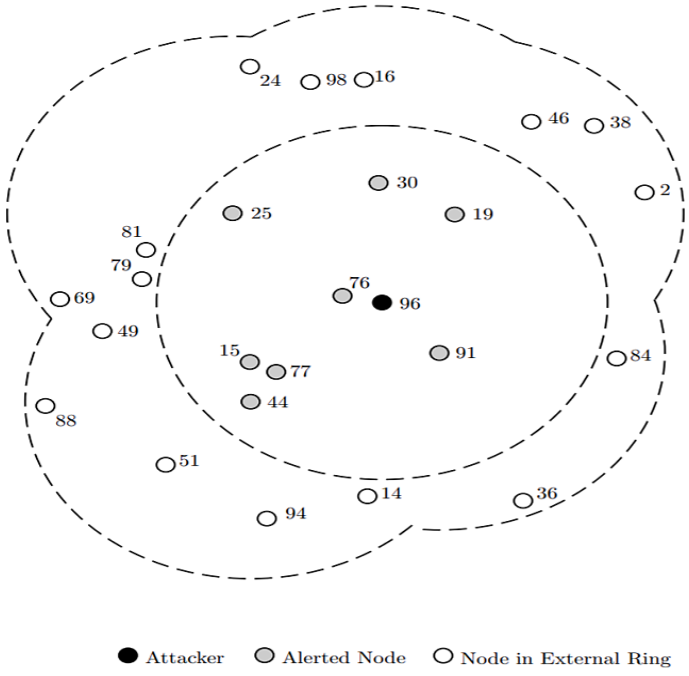


Figure 3.9: The ringing concept in wireless sensor network for proxy's identification.

Additionally, multi-way concurrent control is provided through select statements, which are used to receive data from channels. Another important aspect of python is its rich platform and architecture support, which will be described in detail below. Besides that, the language offers a garbage collected runtime, and stack management, in order to combat memory corruptions [78]. However, this should not be confused with the virtual machine approach of the python runtime. Although python is an object oriented programming language, it provides interfaces and the ability to define methods on structures. Python also enforces a uniform programming style for formatting the source code, which greatly helps increasing readability across code from different authors. It is noteworthy that python provides functionality to circumvent the type system and runtime checks, by reading and writing to arbitrary memory addresses via the unsafe package.

Table 3.1: MQTT protocol sub structures

Name	**Description**
Dot11QOS	IEEE 802.11 Quality Of Service
Dot11HTControl	IEEE 802.11 HTC information
Dot11HTControlVHT	IEEE 802.11 HTC information
Dot11HTControlHT	IEEE 802.11 HTC information
Dot11HTControlMFB	IEEE 802.11 HTC information
Dot11LinkAdapationControl	IEEE 802.11 HTC information
Dot11ASEL	IEEE 802.11 HTC information
LLDPChassisID	Link Layer Discovery Protocol information
LLDPPortID	Link Layer Discovery Protocol information
LinkLayerDiscoveryValue	Link Layer Discovery Protocol information
LLDPSysCapabilities	Link Layer Discovery Protocol information
LLDPCapabilities	Link Layer Discovery Protocol information
LLDPMgmtAddress	Link Layer Discovery Protocol information
LLDPOrgSpecificTLV	Link Layer Discovery Protocol information
IPv4Option	IPv4 option
ICMPv6Option	ICMPv6 option
TCPOption	TCP option
DNSResourceRecord	Domain Name System resource record
DNSSOA	Domain Name System start of authority record
DNSSRV	Domain Name System service record
DNSMX	Mail exchange record
DNSQuestion	Domain Name System request for a single domain
DHCPOption	DHCP v4 option
DHCPv6Option	DHCP v6 option
IGMPv3GroupRecord	IGMPv3 group records for a membership report
IPv6HopByHopOption	IPv6 hop by hop extension TLV option
IPv6HopByHopOptionAlignment	Hop By Hop Option Alignment

Table 3.2:MQTT Layer Encoder Fields.

Layer	NumFields	Fields
TCP	22	Timestamp, SrcPort, DstPort, SeqNum, AckNum, DataOffset, FIN, SYN, RST, PSH, ACK, URG, ECE, CWR, NS, Window, Checksum, Urgent, Padding, Options, PayloadEntropy, PayloadSize
UDP	7	Timestamp, SrcPort, DstPort, Length, Checksum, PayloadEntropy, PayloadSize
IPv4	17	Timestamp, Version, IHL, TOS, Length, Id, Flags, FragOffset, TTL, Protocol, Checksum, SrcIP, DstIP, Padding, Options, PayloadEntropy, PayloadSize
IPv6	12	Timestamp, Version, TrafficClass, FlowLabel, Length, NextHeader, HopLimit, SrcIP, DstIP, PayloadEntropy, PayloadSize, HopByHop
DHCPv4	16	Timestamp, Operation, HardwareType, HardwareLen, HardwareOpts, Xid, Secs, Flags, ClientIP, YourClientIP, NextServerIP, RelayAgentIP, ClientHWAddr, ServerName, File, Options
DHCPv6	7	Timestamp, MsgType, HopCount, LinkAddr, PeerAddr, TransactionID, Options
ICMPv4	5	Timestamp, TypeCode, Checksum, Id, Seq
ICMPv6	3	Timestamp, TypeCode, Checksum
ICMPv6Echo	3	Timestamp, Identifier, SeqNumber
ICMPv6NeighborSolicitation	3	Timestamp, TargetAddress, Options
ICMPv6RouterSolicitation	2	Timestamp, Options
DNS	18	Timestamp, ID, QR, OpCode, AA, TC, RD, RA, Z, ResponseCode, QDCount, ANCount, NSCount, ARCount, Questions, Answers, Authorities, Additional
ARP	10	Timestamp, AddrType, Protocol, HwAddressSize, ProtAddressSize, Operation, SrcHwAddress, SrcProtAddress, DstHwAddress, DstProtAddress
Ethernet	6	Timestamp, SrcMAC, DstMAC, EthernetType PayloadEntropy, PayloadSize

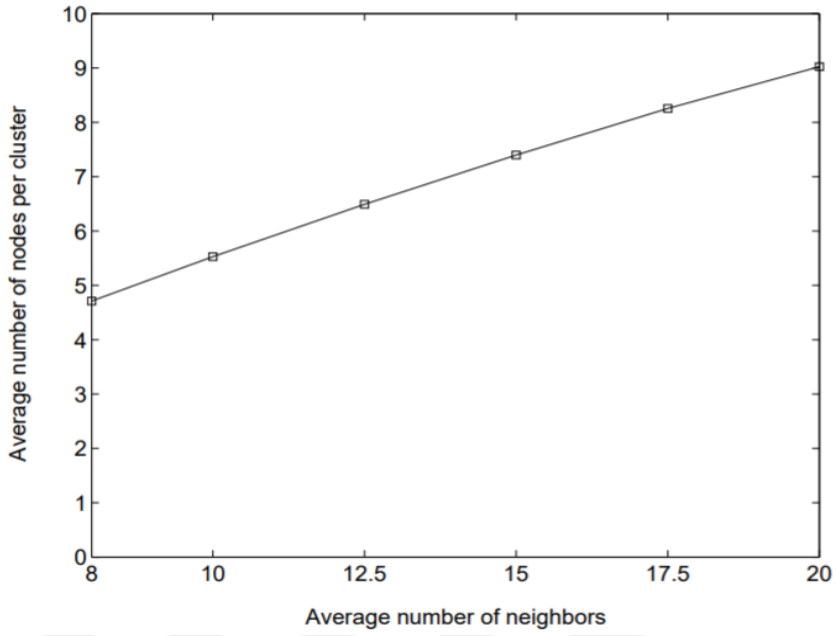


Figure 3.10: Wireless sensor network has average number of nodes in clusters.

3.8 FLOWS AND CONNECTIONS IN MQTT

Flow audit records in MQTT are unidirectional, whereas Connections and Layer Flows are bidirectional. Flows and Connections are continuously flushed, to keep the memory usage as low as possible. Timeout intervals can be configured separately for both types. The first packet seen decides about source and destination of a Connection or Flow.

Table 3.3: Table that describes the different networks with data transfer rate.

Name	Layer	Description
Flow	All	Unidirectional Multi-Layer Flow
Connection	All	Bidirectional Multi-Layer Flow
LinkFlow	Link	Bidirectional Link Layer Flow
NetworkFlow	Network	Bidirectional Network Layer Flow
TransportFlow	Transport	Bidirectional Transport Layer Flow

The outside clients should be verified to get to the accessible intermediary server administrations. The verification ought to be finished utilizing advanced authentications. At whatever point customers need to associate with a public assistance, they ought to validate utilizing their own authentications. The correspondence ought to likewise be finished utilizing HTTPS, which makes the discussion with the web server completely scrambled. As we have examined in the subsection

User Permissions Management there are two kinds of client ideas: Individual Users and User Roles. Along these lines, the computerized authentications could be created for a client job or in any event, for explicit clients. The intermediary server will utilize this data in the validation stage to distinguish which client profile it will utilize.

For Flows and Connections, the following identical information is preserved:

Flow / Connection	Example
Timestamp First Seen (seconds.micro)	1499257434.003136
Link Layer Protocol	Ethernet
Network Layer Protocol	IPv4
Transport Layer Protocol	TCP
Application Layer Protocol	HTTP
Source Mac Address	00:0c:28:9f:16:1e
Destination Mac Address	00:0c:28:c9:60:ce
SrcIP	173.15.11.103
SrcPort	1873
DstIP	95.100.238.75
DstPort	80
Size in bytes	922
Number of Packets	6
Timestamp Last Seen (seconds.micro)	1499257551.553088
Duration (nanoseconds)	117549952000

Table 3.4: MQTT flows and connections for proxying internet.

In the arrangement we propose, the gadgets correspondence is made utilizing MQTT convention because of every one of its benefits. This is connected with its effortlessness, the simple execution and in light of the fact that it is upgraded for high-idleness and problematic organizations. Subsequently, it is compulsory to have a MQTT merchant (a MQTT server) that permits this correspondence. The correspondence transport we use is a WiFi organization and the MQTT dealer likewise gives security systems, for example, utilizing endorsements to validate gadgets. The public administrations are RESTful web-benefits, that burns-through/produces JSON

demands/reactions. With this methodology we ensure adaptability and opportunity to execute various applications, in any stage, without similarity concerns. It very well may be a portable application, a smartwatch, an internet browser application or significantly another IoT framework. The main necessities are that customers ought to have the option to associate with the web and conjure these administrations utilizing HTTP convention with substance in JSON design information.

In our design the utilization of web-administrations, makes the framework more straightforward to reach out with different administrations that we should execute later on, without changing the administrations that are now evolved. The public administrations were created by a Service-Oriented Architecture (SOA) where they are" secret elements" for clients and the administrations are free from one another.

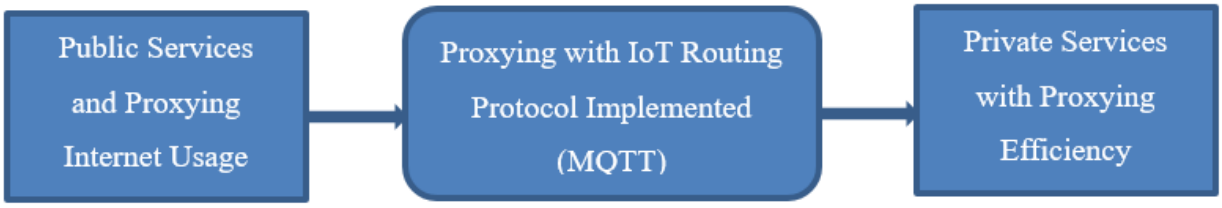


Figure 3.11: The basic processing of proxying system with IoT routing protocol involving the public and private services based on MQTT broker routing protocol.

Another extraordinary benefit utilizing RESTful web-administrations are connected with the framework versatility. The server end of REST is stateless, which implies that the servers don't need to store state or information across demands. Subsequently, the correspondence between servers are negligible, making it exceptionally versatile. Additionally, the RESTful web-administrations are illustrative, which implies that is feasible to give a decent burden balancer on the server-side that can undoubtedly course the solicitations to the right servers, as indicated by the URLs and the HTTP techniques, for example, the GET solicitations could go to a gathering of servers while POSTs go to an alternate one. As indicated by the accepted procedures of RESTful webservices engineering plan, we have utilized the HTTP techniques to separate the accessible sorts of tasks. GET tasks are utilized to recover information in a read-just manner; DELETE activities to erase assets, that can incorporate properties, parts or even gadgets; POST tasks are

utilized to make new assets; and PUT are utilized to refresh assets. The motivation behind these assistances will be portrayed in the accompanying subsections.

We utilize the MQTT convention to permit correspondence between gadgets, because of all of its benefits. Be that as it may, there is one extraordinary disservice in a Pub-Sub engineering: the endorsers should know, deduced, which points are accessible to buy in. This is a major inconvenience and it is totally in conflict with our framework's objective, that is for the most part an exceptionally nonexclusive and truly adaptable framework, that permits the expansion and the evacuation of gadgets specially appointed.

Along these lines, to exploit MQTT convention and furthermore to give adaptability in our framework we make an Internal Devices Management System that cooperates with the MQTT representative and deals with every one of the gadgets associated, their parts and their properties. With this framework, the gadgets don't have to know, prior to being conveyed in our organization, what are the gadgets accessible and their properties. This is an extra framework yet it is important to permit gadgets to know, after their arrangement and with the framework ready for action, what are the organization properties and the gadgets associated.

The inside gadgets the executive’s framework ought to consistently be synchronized with the genuine framework data; in this way it is compulsory that gadgets in their introduction interface with the intermediary server (in the individual administrations) to enlist their data. A more complete model is represented in the Examples of Usage segment, however here is a short clarification regarding how this interior gadget the executive’s framework works: We have a crystal fixture (a gadget) that we need to associate in our savvy intermediary framework. During the instatement the gadget should enroll itself in the intermediary server utilizing the private help. /programming interface/de-bad habit/gadgets, and register their parts and properties. Thus, the crystal fixture will likewise make a solicitation to/programming interface/gadget/parts telling that it contains a Lamp (part) and this light has a Mode (property) that permits the qualities ON or OFF. Utilizing this instrument, the Internal Devices Management framework will consistently realize the organization express, the gadgets associated and their properties. It permits to add new gadgets to the organization impromptu without resetting the whole organization or refreshing the gadgets code. To accomplish the positive knowledge, our framework gives a Complex Actions Management System that permits the production of streams of execution as per explicit conditions,

for example, gadget parts' states. Despite the fact that we didn't execute this component in our framework, it very well may be carried out utilizing a methodology that comprises in a spellbinding language that permit clients to depict the perplexing activities. A surveying framework will check occasionally in the event that the properties' qualities match with any conditions characterized and, assuming they do, the framework will execute the comparing activities.

3.8.1 LinkFlow in MQTT

In order to further decouple Flows, they are also exported for each layer, e.g: LinkFlow, NetworkFlow, TransportFlow. As mentioned previously, Layer Flows are bidirectional, their UID is the same for both directions. Note that in the following graphics the UID field was omitted. A LinkFlow describes communication on layer 1 of the Internet Protocol Suite. It contains the hardware adresses of the communicating devices, and connection statistics.

Link Flow	Example
TimestampFirst (seconds.micro)	1499255388.191380
TimestampLast (seconds.micro)	1499284556.475471
Protocol	Ethernet
SrcMAC	00:0c:28:9f:16:1e
DstMAC	00:0c:28:c9:60:ce
NumPackets	52
Size (bytes)	1423
Duration (nanoseconds)	29168284091000

Table 3.5: MQTT Link Layer Flow for proxying internet.

3.8.2 Network Flow in MQTT

A NetworkFlow describes communication on layer 2 of the Internet Protocol Suite. It contains the IP addresses of the communicating devices, and connection statistics.

Network Flow	Example
TimestampFirst (seconds.micro)	1499283872.749692
TimestampLast (seconds.micro)	1499284166.267369
Protocol	IPv4
SrcIP	192.168.10.8
DstIP	185.11.128.203
NumPackets	137
Size (bytes)	30160
Duration (nanoseconds)	293517677000

Table 3.6: MQTT Network Layer Flow for proxying internet.

3.8.3 Transport Flow in MQTT

A TransportFlow describes communication on layer 3 of the Internet Protocol Suite. It contains the ports of the communicating devices, and connection statistics.

Transport Flow	Example
TimestampFirst (seconds.micro)	1499279129.560185
TimestampLast (seconds.micro)	1499282267.760859
Protocol	TCP
SrcPort	60846
DstPort	443
NumPackets	156
Size (bytes)	118942
Duration (nanoseconds)	3138200674000

Table 3.7: MQTT Transport Layer Flow for proxying internet.

3.9 BUILDIND PROXYING INTERNET

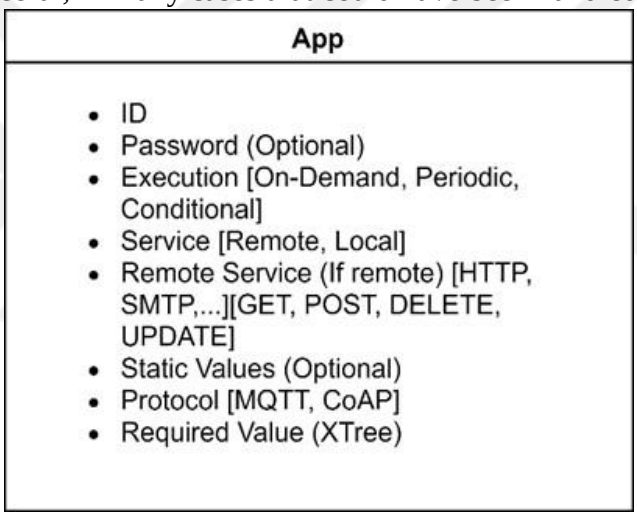
In concurrent programming, shared resources need to be synchronized, in order to guarantee their state when modifying or reading them. If access is not synchronized, race conditions occur, which will lead to faulty program behavior. To avoid this and detect race conditions early in the development cycle, the go toolchain offers compiling the program with the race detector enabled. This will let the application crash with stack traces to assist the developer in debugging, if a data race occurs. Programs with active race detection are slower by the factor of 10 to 100. To compile a python program with the race detection enabled the **-race** flag must be added to the compilation command.

To add support for a new protocol or custom abstraction the following steps need to be performed. First, a type definition of the new audit record type must be added to the *netcap.proto* protocol buffers definitions, as well as a *Type* enumeration following the naming convention with the *NC* prefix. After recompiling the protocol buffers, a file for the new encoder named after the protocol must be created in the *encoder* package. The new file must contain a variable created with *CreateLayerEncoder* or *CreateCustomEncoder* depending on the desired encoder type. Depending on the choice of the encoder type, the new variable must be added to the *customEncoderSlice* in *encoder/customEncoder* or *layerEncoderSlice* in *encoder/layerEncoder*. Next, the interface for conversion to CSV must be implemented in the *types* package, by creating a new file with the protocol name and implementing the *CSVHeader() []string*, *CSVRecord() []string*, *NetcapTimestamp() string* functions of the *types.CSV* interface. If the new protocol contains sub-structures, functions to convert them to strings need to be implemented as well.

Finally, the *InitRecord(typ types.Type) (record proto.Message)* function in *netcap.go* needs to be updated, to initialize the structure for the new type.

Figure 3.10. Configurations of an App in the proposed system.

Processing time depends heavily on the type of traffic, configuration and system performance. Processing times on the development machine (a recent MacBook Pro with 32 GB 2400 MHz DDR4, 2,9 GHz Intel Core i9, running Windows 10), using the default configuration with 4096 byte buffers, compression, all 40+ encoders, 1000 workers and a packet buffer size of 100, can be found in the experiments. The project may contain bugs, that have not yet been discovered. Error handling is not very graceful, in many cases that could have been handled otherwise, the program



panics in order to assist in debugging with a stack trace. Until there are further unit tests and the error handling is more robust, using python for other purposes than research is recommended. The following shows a summary of statistics regarding lines of code (LoC) in python version 3.8 (excluding generated code for the protocol buffers):

Table 3.8: Python code statistics for proxying internet within files.

Package	Language	Files	Blank	Comment	Code
collector	Python	13	273	371	882
types	Python	38	232	465	1721
label	Python	15	365	413	1266
encoder	Python	48	559	957	3299
cmd	Python	3	57	64	282
utils	Python	2	39	44	156
netcap.proto	XML	1	93	100	660
total	Python	126	1693	2520	8296

Python source code contains many descriptive comments (80-100 lines of comments on several lines of code as of version 3.8), in order to ease future development by other researchers and to make implementation details comprehensible.

Table 3.9: Statistics for processing input packets for proxying internet.

File	Num Packets	Size	Processing Time	Extracted Data
Tuesday-WorkingHours.pcap	11551954	11 GB	11m11.918614271s	440 MB
Wednesday-WorkingHours.pcap	13788878	13 GB	12m58.13498869s	536 MB
Thursday-WorkingHours.pcap	9322025	8.3 GB	8m39.274062535s	369 MB
Friday-WorkingHours.pcap	9997874	8.8 GB	9m0.653867719s	410 MB

3.10ELEMENTS OF PROXYING INTERNET

The proxying internet components may be summarized as in the following:

Users: Users are those who will engage with or inhabit the installation.

Sensing Devices: These devices are installed IoT devices that serve as data generators. Their data is transferable, consumable, and collectible for future use.

Actuator Devices: These are IoT devices that may be commanded to cause a physical change. They consist of components such as light switches and the thermostat setting in an HVAC system.

Gateway Devices: They are IoT devices whose only purpose is to transmit data between the Internet and an installation. They play a significant function in installations where the other IoT devices are not Internet-enabled and cannot connect directly with the rest of the system.

Observed Phenomena: It refers to the physical events seen by the actuator devices or the digital sensor.

Units of Measurements: It is related to the standard used to quantify an observable occurrence.

Locations They refer to the logical and physical classifications that may be given to any previously described parts. They may characterize a building, a classroom unit, or a school and a more general collection of users throughout various school communities.

Similarly, the connections between the previously described components are specified as follows:

Ownership: It specifies the relationship between a collection of sensor, actuator, or gateway devices and physical locations over which users have complete authority and rights. This relationship is essential for administrative and security purposes and can be utilized to manage the entire installation.

Access: It specifies the right of a user to examine the data provided by a group of sensing devices or physical locations or to manipulate an actuator device. Such permissions may be restricted depending on the user type. For instance, instructors and students may only require access to information about their classroom, but they are only permitted to examine restricted information about the rest of their school facilities. Similarly, students may not be permitted to manipulate actuator devices in their classroom since features like the HVAC system are controlled only by building administrators or approved teachers (e.g., the principal).

Sensing Attributes: Sensing attributes refer to the sensing and actuator devices and are used to define what type of information each one of the devices generates, or what type of device it controls. The can be information about the physical phenomenon or the measurement unit that is measured or even whether the data generated are raw values or need a post processing to be valid.

Another MQTT include that we use is the QoS interior framework that is liable for the conveyance confirmation of each message. In our framework we permit gadgets to buy in/distribute messages from the three accessible QoS levels, yet our proposal is that basic messages ought to have the most extreme QoS level: 2, that ensures that messages are conveyed and furthermore it is a non-repeatable activity, which means messages are not sent twice or more occasions. One illustration of a message with a level 2 of QoS, could be an alarm. For this situation, we need the framework to continue to attempt to convey the message until it at long last arrives at the right objective. The QoS levels of every theme are characterized in the gadget enrollment stage and they could be effectively changed thereafter. They are characterized to a particular property, which gives incredible adaptability to deal with the QoS levels as indicated by the most fined grained level of

gadgets. In the event that the QoS level isn't characterized, the default esteem is 0 (the most minimal and quickest one).

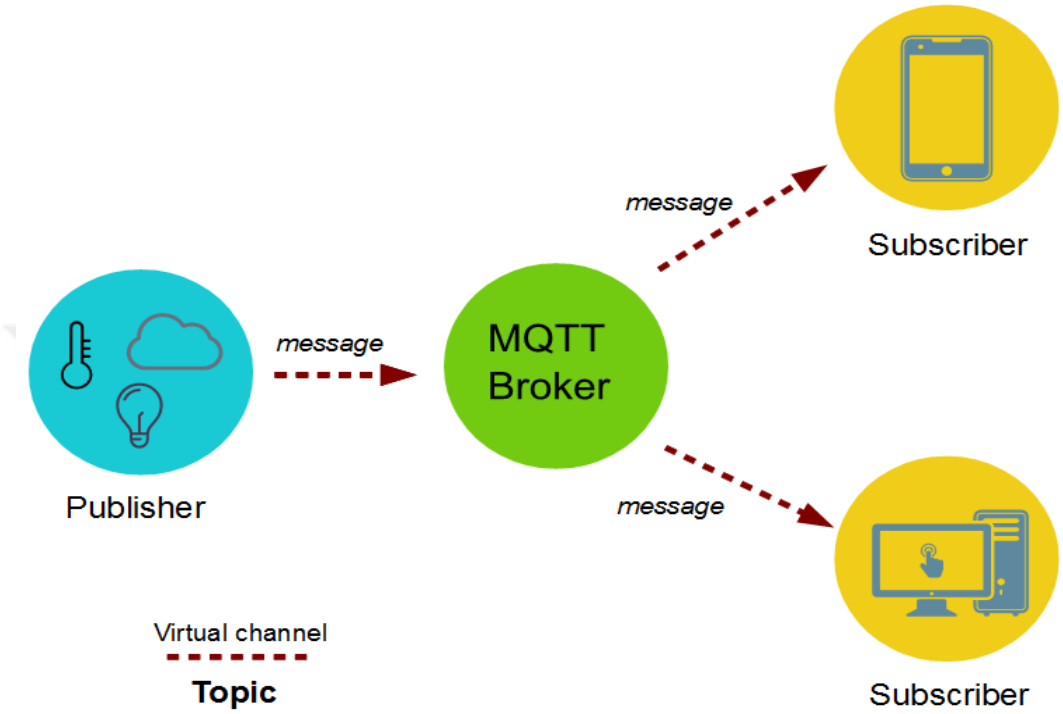


Figure 3.12: The general sequencing of MQTT broker for managing the actions.

Utilizing a framework with an outer module (server) for the MQTT specialist likewise permits the correspondence between gadgets without the extra" work" to the intermediary server. This is valuable to diminish the payload in the intermediary server and, assuming the clients need to associate gadgets that have better computational assets, that can deal with a few" complex" activities for themselves, they can have gadgets overseeing activities straightforwardly from different gadgets, without the intermediary server going about as a delegate in this interaction. The disservice of this methodology is the extra intricacy that it adds to deal with these intricate activities, for certain activities characterized at the intermediary server level and others straightforwardly in the gadgets.

In our execution we likewise exploit the MQTT held messages. These messages are characterized in the MQTT convention and we use them with the reason to let different gadgets known when a gadget is disengaged and furthermore to store the last message. The last option highlight permits that an as of late associated gadget can know promptly which is the last condition of a particular

property or part of other gadget, without trusting that another message will get a refreshed state. The framework has, per default, one client job: An IoT-based MQTT merchant has doled out the greatest authorizations level: it permits to change every one of the gadgets, parts and properties values; to make new burden balancer in jobs and relegate them to explicit piece of keen intermediary; to make new rooms and furthermore to make new consents with the assistance of MQTT dealer with low mistake rate.

4. RESULTS

In order to illustrates the benefits of using the proposed system to support IoT devices, the system is implemented using Python programming language using an Amazon LightSail web server. The web application that is used to create and configure the Apps is implemented using Flask web system, whereas the Paho-MQTT python library is used to implement the MQTT broker and aicoap for the CoAP protocol. Then, an ESP32 de- vice is used to access the same services using both the existing methods and libraries, i.e. directly, or by using the proposed system. The performance of the ESP32 is measured by measuring the total energy, memory, time and data sent and received by the device. For accurate measurements, one of the pins of the ESP32 is set to high just before the execution of the required task and is set to low as soon as it ends, to indicate the start and end of the task. The time and amount of energy are measured based on the state of this pin. In the first experiment, the pro- posed system is used to communicate data between IoT devices using the proposed system and compared to communicating the same amount of data using the popular ThingSpeak service. As the ThingSpeak service also allows logging data, the App that is created for this task is set to update a point at the ThingSpeak server using HTTP request and update all IoT devices that are subscribed to this value as soon as it changes. Hence, the App is set to update conditionally, i.e. using the MQTT protocol, as soon as the value changes. The comparison is conducted based on sending and receiving 1, 5 am 10 bytes of payload data from one IoT device to another, while storing the value in the ThingSpeak server in both cases, directly or using the proposed method.

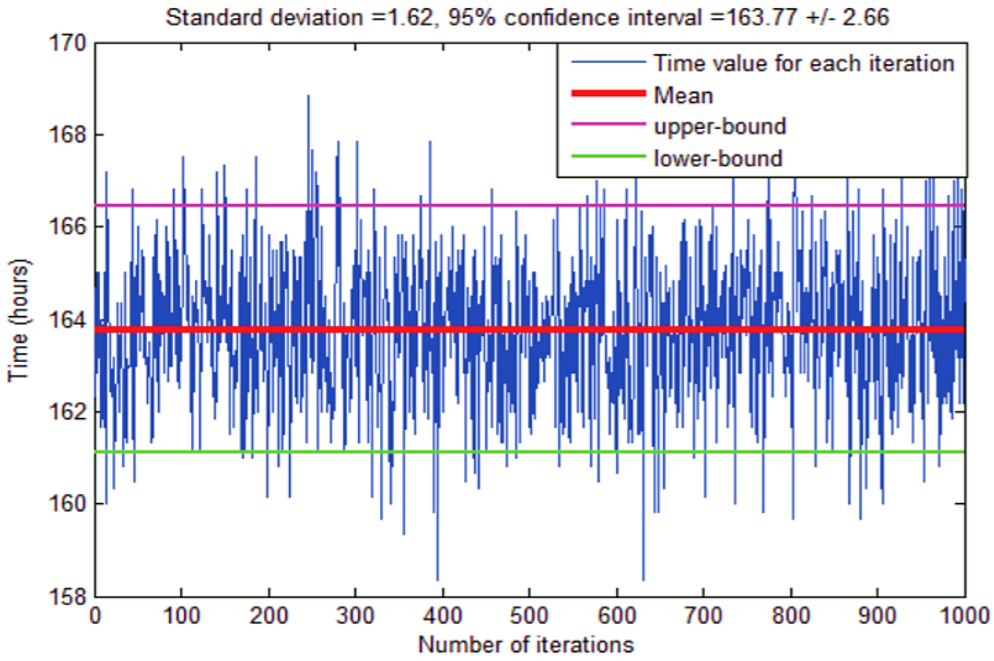


Figure 4.1: Conveyance plot of the recreation time in the simulation for proxying internet.

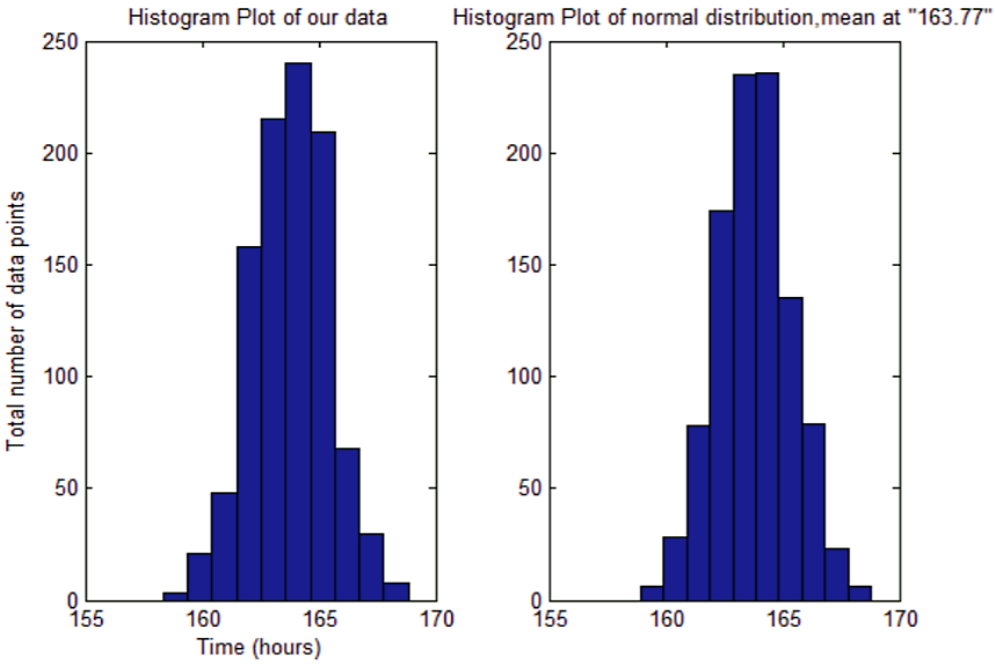


Figure 4.2: Histogram plot of the reproduction time contrasted with the ordinary conveyance for data points in the WSN.

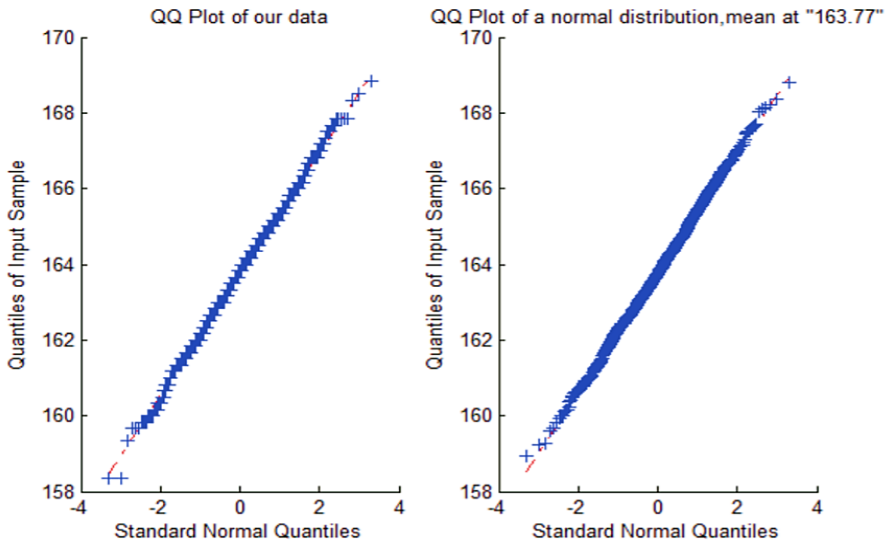
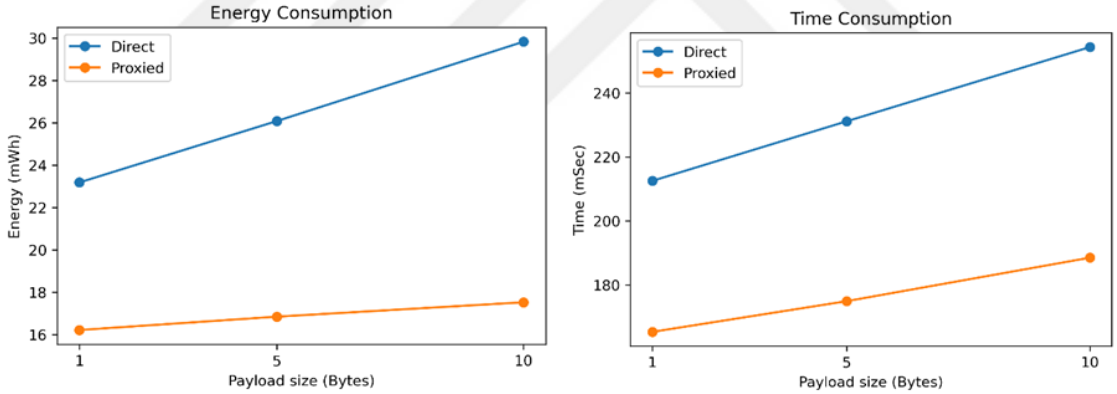
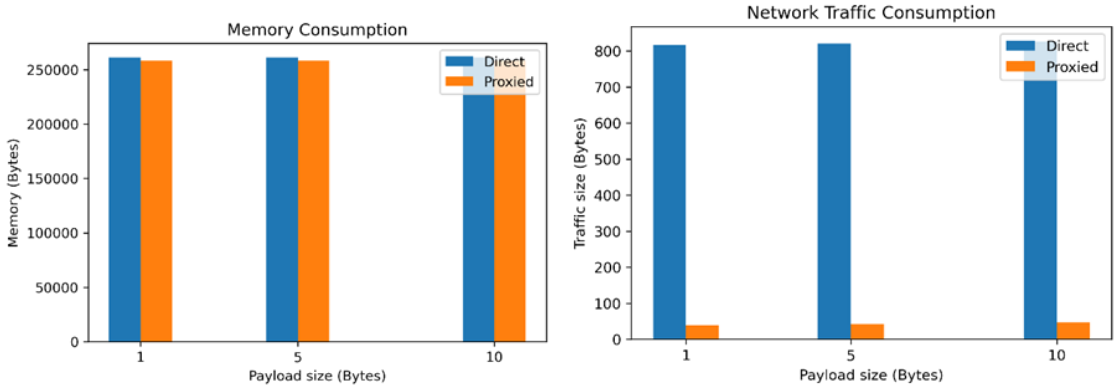


Figure 4.3: Standard normal quantiles represented in the simulation with time in contrast with the normal distribution.

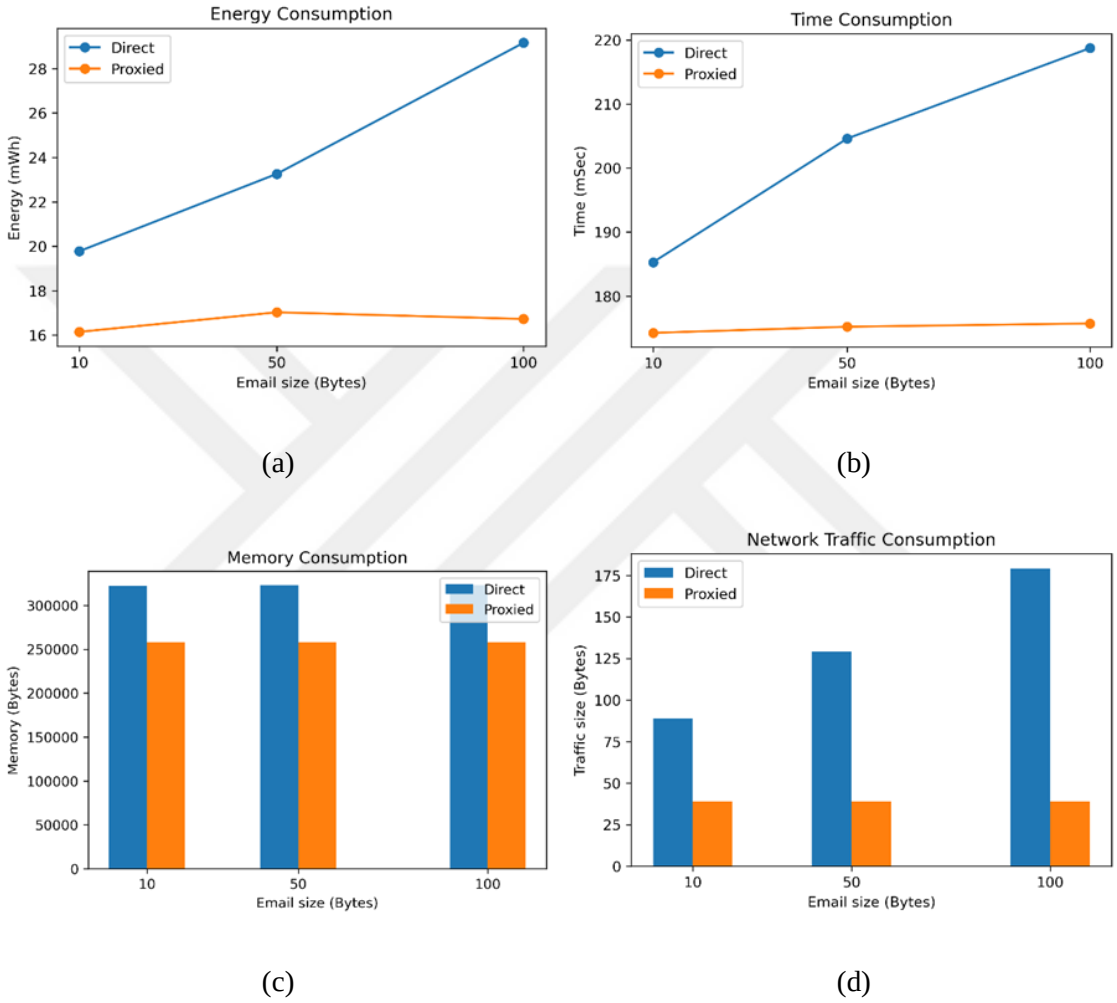


(a) (b)



(c) (d)

Figure 4.4: IoT devices' resources consumption (communicate data directly by the suggested system): (a) Energy; (b) Time; (c) Memory; (d) Network traffic consumption.



(a) (b)

(c) (d)

Figure 4.5: IoT devices' resources consumption (send emails directly by the suggested system): (a) Energy; (b) Time; (c) Memory; (d) Network traffic consumption.

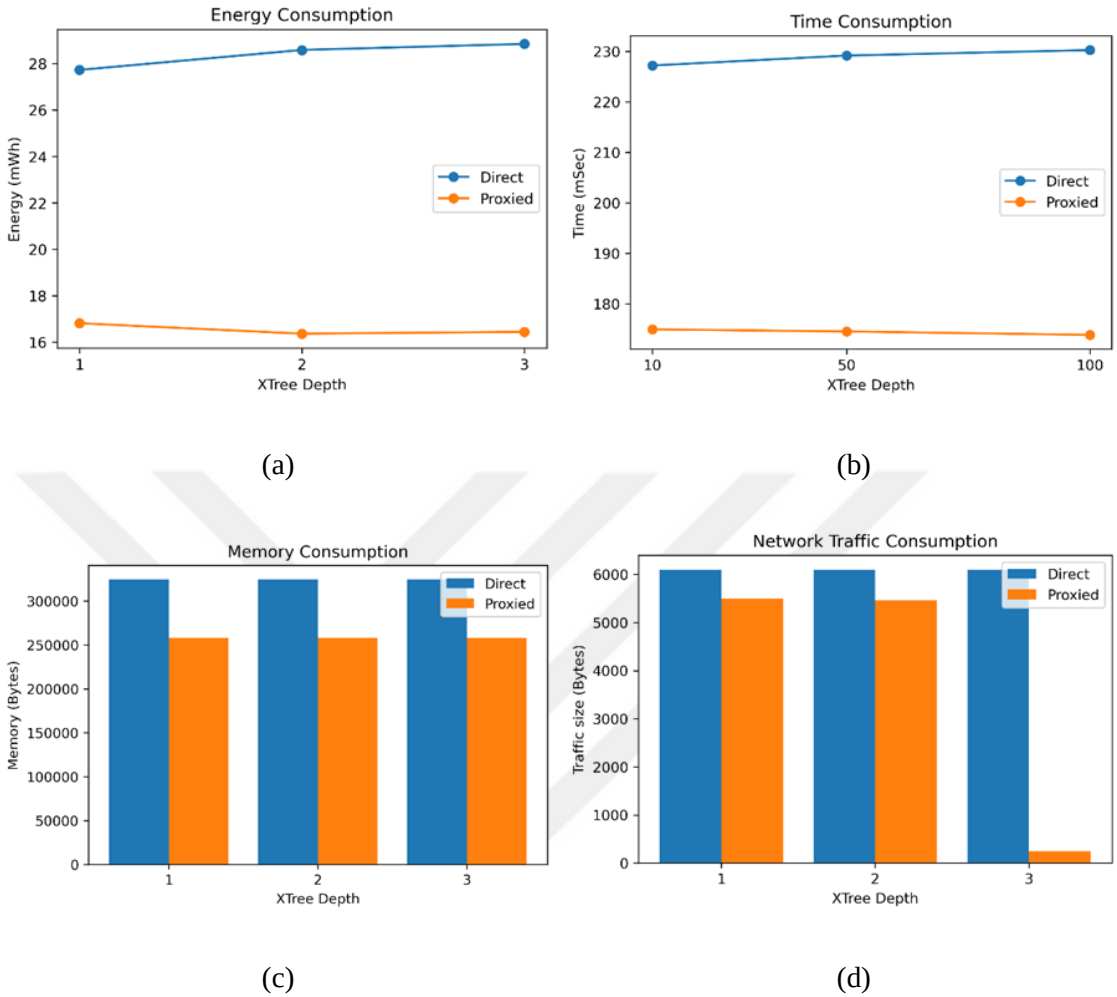


Figure 4.6: IoT devices' resources consumption (dierect an XML web service acces by the suggested system). (a) Energy; (b) Time; (c) Memory; (d) Network traffic consumption.

The theoretically good false positive rate of 0.0015 in practice would still mean that more than 1 in 1000 normal data points would be falsely labeled as a proxy. If a data point is generated every 100 milliseconds, that would give us a false alarm every 100 seconds, and this is not practical. One data point by itself, however, could hardly be considered a proxy, where a continuous stream of packets is usually sent. To that regard, this MQTT approach could be implemented, where an alarm only goes off if there is a majority of proxy data points within a window of some specified size.

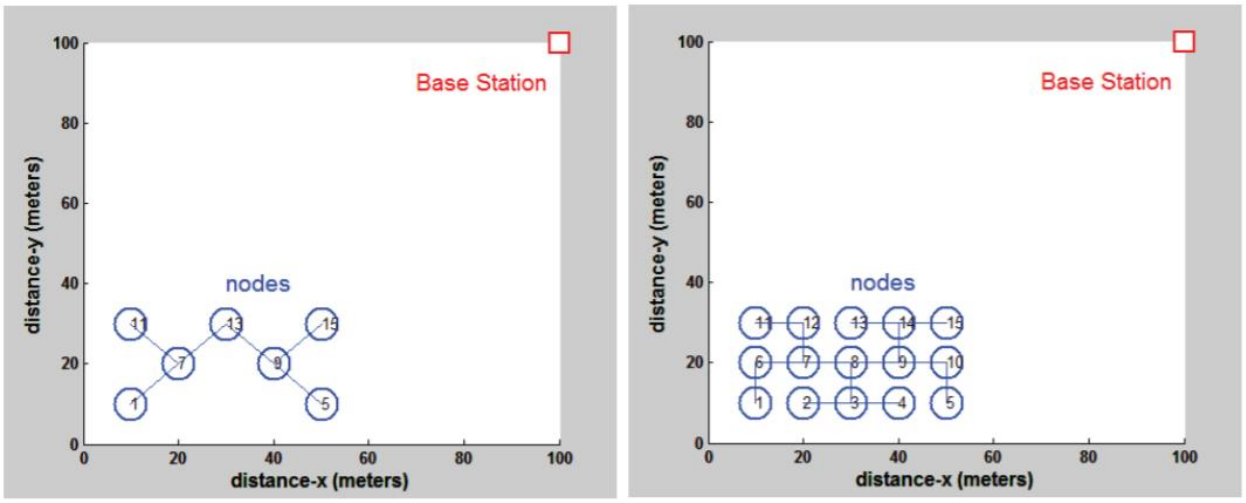


Figure 4.7: Diverse organization geographies of nodes in wireless sensor network with 7 and 15 nodes in base station.

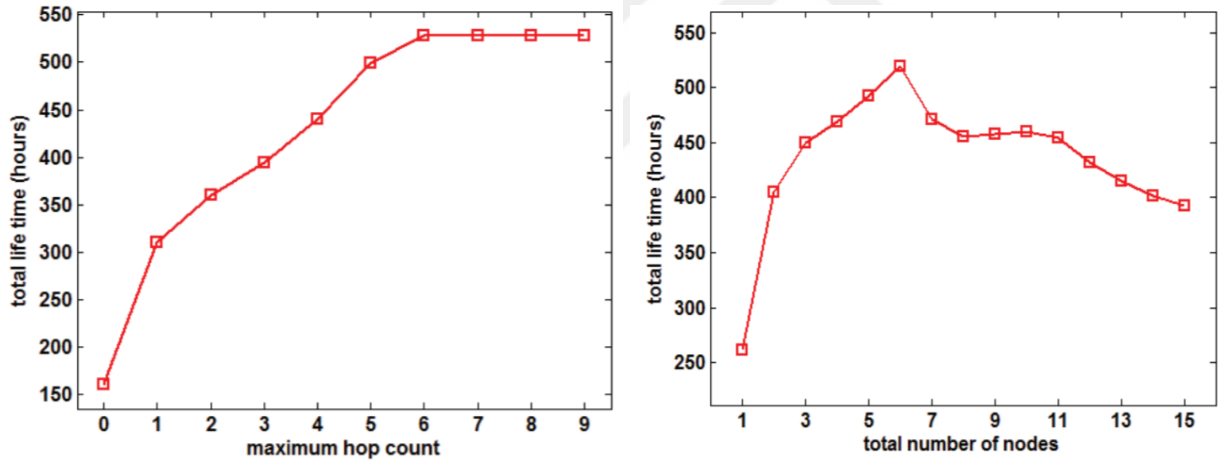


Figure 4.8: Maximum hops and total number of nodes in a network vs. total life-time of the network.

Our framework was intended to empower the use of any low-compelled gadget type as far as memory and handling power. In this way our necessities are negligible and the intricacy of our framework is in the intermediary server side, that is the reason a gadget ought to just be worried about its own parts and properties, considerably diminishing their equipment prerequisites.

The base necessities are introduced and essentially a gadget can associate with the web by means of WiFi: it ought to have carried out the TCP/IP convention and have the option to perform HTTP demand/reactions because of MQTT Protocol and intermediary server tasks.

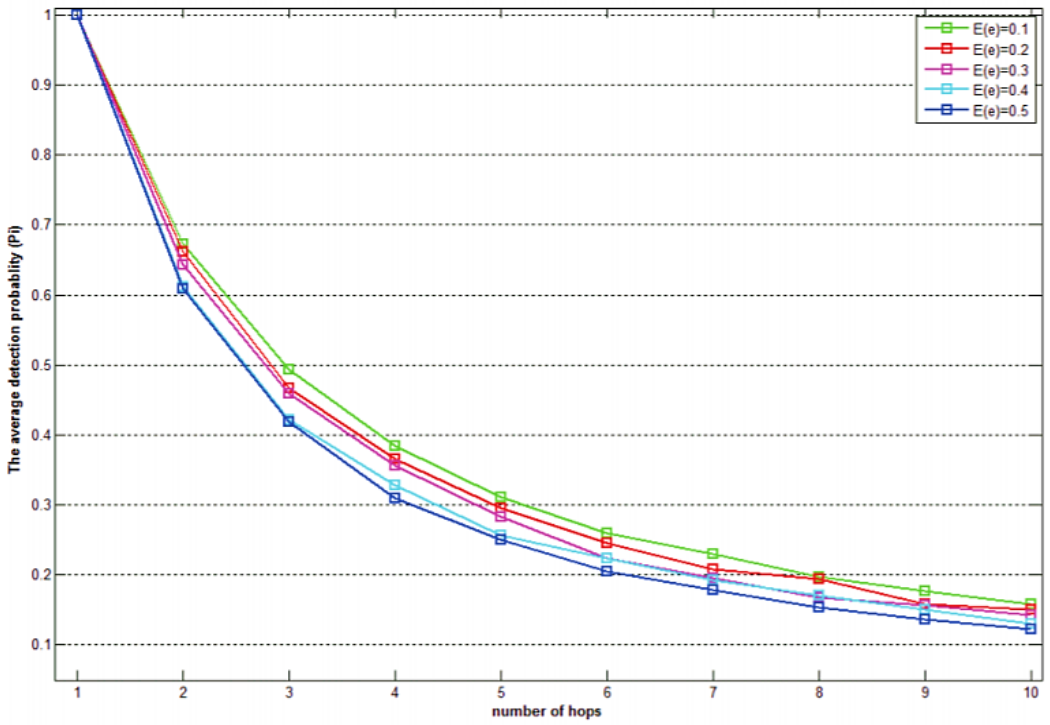


Figure 4.9: Impact of cluster size on the location likelihood of the proxying internet system for different packet misfortune rates while the rest rate is 60% in MQTT.

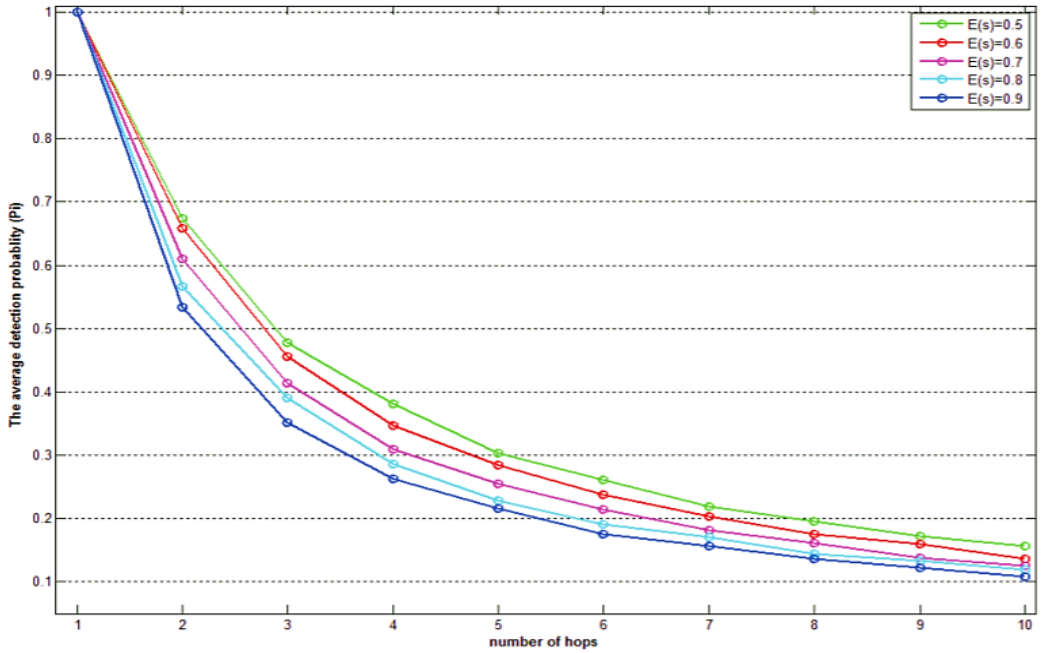


Figure 4.10: Impact of cluster size on the location likelihood of the proxying internet system for different rest rates while the packet misfortune rate is 30% in MQTT.

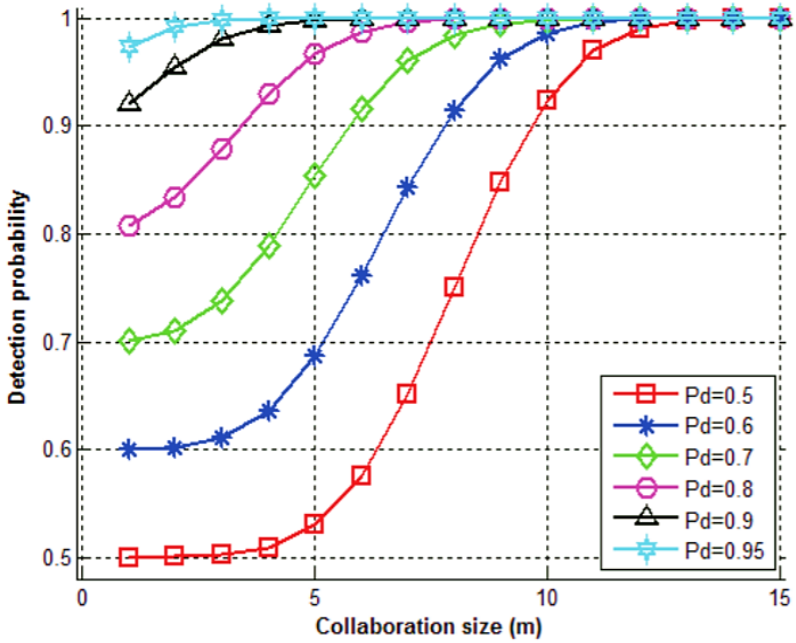


Figure 4.11: Compiling the MQTT with the race detector enabled for detection probability (P_D) vs. collaboration size(m) for proxying internet.

This framework permits to utilize any customer in any stage: it tends to be a cell phone application, a savvy application, a website page or whatever else. The main necessity is that the application should have the option to execute HTTP convention, on the grounds that the customers ought to have the option to perform RESTful API calls. Likewise, to the gadgets, assuming the customer has a JSON library the necessary code will be more straightforward, however this isn't obligatory.

The customers in our framework could be from two distinct sorts: other IoT frameworks, and in this situation they don't give any UI (it is an ordinary situation of M2M correspondence) or they are intended to offer client collaboration, and in this situation they ought to be planned agreeing the best convenience rehearses. In our framework execution we have not made any UI (is out-of-scope), we utilized just the order line to test the customer REST API calls, which demonstrates that the customer insignificant necessities.

5. DISCUSSION

For experiments conducted in this study, all WSN intrusions were detected inside nodes and clusters. As dummy variables, nodes are encoded. A new column is created for each unique entry, and the original column is deleted from the dataset afterward. To numeric values, booleans are encoded1 for true and 0 for false. For experiment 1, the first run is stopped after twelve hours after hanging for more than seven hours at the dummy variable encoding the Answers columns from Tuesday-WorkingHours-1/DNS labeled.csv. From 1 to 0.5, the sample size is dropped to decrease the data amount needed to be processed. Accordingly, half of the existed data will be utilized.

Nevertheless, The following run was likewise unable to finish in a proper length of time and was halted after eight hours, at the exact location as previously. After beginning the third run and collecting just 20% of the available data, the experiment was concluded after more than 29 hours. The first experiment was done using an early version of clusters and was never replicated. It is included here for completeness and because subsequent experiments were modified based on the results of this one. After sampling, the entries marked with"-" consisted of only one label type. Therefore, the Keras library was closed since no prediction was feasible.

Table 5.1: The classification of different proxies with respective count.

Classification	Count
Generic Protocol Command Decode	6449
Attempted Information Leak	3
Not Suspicious Traffic	80
Potential Corporate Privacy Violation	12
Potentially Bad Traffic	25
Misc Proxy	18

Table 5.2: Classification results experiment of intrusions in WSN network.

File	Num Records	Size	Labels	Exec Time	Validation Score
Connection labeled.csv	284166	51 MB	2119	50s	0.9919065381096488
DNS labeled.csv	700447	144 MB	33	3h 27m	0.9999428946692174
Ethernet labeled.csv	11551954	880 MB	3556	6m 49s	0.9996693201846267
Flow labeled.csv	647294	112 MB	4852	4m 2s	0.9904835470415573
HTTP labeled.csv	45852	14 MB	5609	21s	0.9637554585152839
IPv4 labeled.csv	11469736	1.0 GB	3555	-	-
NTP labeled.csv	15507	2.1 MB	18	2s	0.9987113402061856
TCP labeled.csv	10710230	1.5 GB	3504	-	-
TransportFlow labeled.csv	91861	5.8 MB	11	8s	0.9999059354717336
UDP labeled.csv	787015	44 MB	51	27s	0.9999491753703845

Using the dummy variable method for encoding strings delivered a good detection accuracy of around 99%, however it is impractical for operating on large datasets, due to the very high time for encoding of features. Because each unique string will lead to the creation of a new column, the dataset grows in width tremendously. For the evaluation of CIC IDS 2017 this strategy of encoding strings might have been possible, but regarding the size of modern datasets, another approach is needed. Dropping columns with a high amount of unique entries (such as DNS Answers, IP addresses) was not considered, because this first series of experiments aims to create a baseline of detection accuracy with the complete data that cluster delivers. Entries with"-" have failed when entering the training phase, because after sampling and encoding, the subset did only contain normal labels and none of the malicious types. For experiments, encoding will be slightly modified in order to reduce the time needed for the encoding stage. All numeric values will be clustered like before. Booleans are encoded to numeric values, 0 for false, 1 for true. Instead of choosing dummy variables, strings will be encoded as indices this time. This keeps the original column in the dataset and creates a new one containing a unique integer for each unique string. Originally this was used to encode only the result column, the following experiment evaluates the use of this string encoding method for the whole CIC IDS 2017 dataset. Experiments have a total processing time of over 8 hours. Processing time for encoding is now much better. Detection accuracy on HTTP traffic declined, most notably on the Working-Hours dump. Others slightly increased, for example the Working-Hours/Connection labeled.csv detection accuracy increased from 98.19% to 98.68%. Accuracy for detecting flows stayed almost the same. Classification results are comparable to the ones from the previous run and in this case do not seem to be influenced by the higher amount of unique labels. This is quite interesting, because classifying specific proxies could generate more specific alerts, which would provide more value to the analyst.

Results indicate that effective categorization of malicious or undesired activity inside WSN is attainable using a feature set consisting primarily of collective characteristics and just a few produced features. In addition, the classification of individual protocols yielded better results than the classification of summary structures such as flows and links. The time required to encode the data to a numeric feature vector highly depends on the encoding approach used for strings

(alphanumeric values). Encoding texts as indices outperformed encoding them as dummy variables but needed more time for training the deep neural network. Instead of proxy classes, proxy descriptions are used as labels that deliver comparable outcomes and must be able for more tests. Due to the possibility of multiple warnings for the same audit data, a strategy must be devised to manage this situation. Collecting all classifications and combining them into a single label has shown excellent results and should be studied further using proxy descriptions rather than proxy classes.

Table 5.3: The comparison of proposed method of implementation with existing technique.

ARTICLE	TECHNIQUE	ACCURACY
[79]	Support Vector Machine (SVM)	89.97%
[80]	Convolutional Neural Network (CNN)	96.73%
Proposed	MQTT based IoT Protocol	98.68%

It was important to make a straightforward, more adaptable convention that would ensure interoperability and security in the control and organization of this framework. Our framework offers a cheap arrangement, with low execution costs and simple establishment, that permits any client to send it in his intermediary. Our answer utilizes the WiFi network as correspondence innovation, since the majority of the clients as of now have in their intermediaries and furthermore permits to reuse existing gadgets by adding just a basic microcontroller, like an Arduino, that associates these gadgets to the web. This is a mind boggling framework, in spite of the fact that it is inherent a measured style continuing decoupling between modules, permitting clients to change the framework settings at whatever point they need, adding or eliminating gadgets after the framework is carried out and right now functional, similar to a "fitting and-play" framework. It is a framework whose main interest group is very wide, including standard clients without specialized information, that absolutely need to purchase the gadgets and consolidate them into their framework, or Do-It-Yourself (DIY) Makers, who need to make their own IoT intermediary machines. In spite of being flexible, our framework is hearty and offers security components so clients can in-slow down it with certainty. These security instruments result from the planned engineering that is straightforward to the gadgets inside the organization, and permits separating two sorts of organizations: the outside network, where clients interface from a distance to control

the gadgets subsequent to being appropriately confirmed, and the inner organization limited to intermediary gadgets correspondence and organization purposes.

The center part of our framework is the brilliant intermediary server that is a gadget with seriously handling power assets, for example, a python based framework or a MQTT-based framework. This part contains all the framework security and the executive's instruments, for example, the client authorization the board, that permits to make clients and client jobs with consents allotted at various levels. It is feasible to characterize authorizations at various degrees of order (room, gadgets or properties) which gives incredible adaptability in overseeing gets to the gadgets.

With respect to gadgets permitted, the prerequisites are negligible since the framework the executives is done on a more significant level, necessitating that gadgets just deal with their own parts and can interface with a WiFi organization. Our framework additionally offers a component that permits the formation of entryways to associate with easier gadgets, that can't interface with a WiFi organization. Since the framework engineering was intended to be exceptionally versatile, there is no restriction with respect to the amount of gadgets that clients can associate. It is planned concurring a help arranged engineering with RESTful web-administrations, free of one another, with a MQTT representative for inside correspondence, that permits lightweight and straightforward messages between gadgets. There is likewise an assistant data set that stores all the framework data, like associated gadgets, their parts and status, the design of the house (rooms), enrolled clients, permitted client jobs and their authorizations.

During our assessment, we saw that there is a little postponement in the messages' conveyance, which is identified with the message engendering time over the organization and furthermore to the measure of parts included. Notwithstanding, this span is short, and the more perplexing activity (when a client changes a property estimation) requires 1.5 seconds on normal to be handled by the framework upon receipt, which is sensible considering the obliged assets of IoT gadgets. The design of our framework incorporates an instrument for overseeing clients and their consents. We proposed an answer, in any case, this was an instrument that was not carried out and tried in our undertaking, so we can't exhibit a substantial assessment in regards to the adequacy and great working of our answer. Our framework furnishes security related with the outer gets to, expecting clients to utilize secure correspondences conventions, HTTPS, and verification with computerized authentications. These security components are easy to be utilized by clients, nonetheless, albeit

this usefulness is thought about in the design of our answer, we have not had the option to execute and test it appropriately, so we have not had the option to demonstrate that it is satisfactory to give the advantageous security.

6. CONCLUSION

6.1 CONCLUSION

With the rapidly growing use of the internet to establish com- medications and access services by different types of devices, the Internet of Things has started to gain significant attention in recent years. Despite the design and use of newly designed light-weight protocols for IoT devices, an enormous number of existing services still use older protocols, such as HTTP. De- spite the significant support that these services can provide to IoT devices, accessing them using existing protocols is exhaustive to the limited resources available on IoT devices. Hence, a new system is proposed in this paper to proxy all internet ser- vices to IoT devices using the lightweight MQTT and CoAP protocols. The proposed system makes use of the relatively larger resources available via cloud computing, compared to IoT devices, to execute all the actions required to access a certain service and extracts the required information from their responses and deliver them to the IoT device, if required. The evaluation results of the proposed method show that significant reduction in resources consumption has been achieved when accessing different types of services. This reduction is according to the less overhead imposed by the MQTT and CoAP proto- cols, compared to the standard HTTP and SMTP protocol, for example. The reduction is also a result of reducing the communication of any redundant data, by storing such data in the pro- posed system and only append dynamic values that are sensed by the IoT device. Finally, the memory required from IoT de- vices has also been significantly reduced as no additional libraries are required to be compiles and stored on the IoT de- vice's memory. Hence, the use of the proposed method can significantly increase the efficiency of any IoT device by providing all the internet services using minimum resources consumption.

6.2 RECOMMENDATIONS

In future work, the efficiency behind including image processing techniques in the proposed framework is going to be evaluated. Despite the possible improvement in the efficiency, according to the less processing required from the IoT device, communicating image data can also be exhausting for these devices, according to the enormous amount of data in these images. Compressing these images to reduce the data also requires intensive processing. Thus, it is

important to evaluate the efficiency under different circumstances and make a proper decision whether to include such techniques or not.

i. In future work, the efficiency behind proxying image processing techniques is going to be evaluated.

ii. Despite the significant reduction in processing at the IoT device, transmitting the huge amount of data in images can limit the efficiency of such proxying.

iii. With limited bandwidth and relatively high power consumption for data transmission, processing images locally at the IoT device can be more efficient.

REFERENCES

[1] Bamakan, S.M.H.; Wang, H.; Yingjie, T.; Shi, Y. An effective proxying internet framework based on MCLP/SVM optimized by timevarying chaos particle swarm optimization. Neurocomputing 2016, 199, 90–102.

[2] Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," IEEE Transactions on Computers, vol. 65, no. 10, pp. 2986–2998, 2016.

[3] Ghazy, R. A. Ghazy, E.-S. M. EL-Rabaie, M. I. Dessouky, N. A. El-Fishawy, and F. E. Abd El-Samie, "Efficient techniques for attack detection using different features selection algorithms and classifiers," *Wireless Personal Communications*, vol. 100, no. 4, pp. 1689–1706, 2018.

[4] Aljawarneh, S.; Aldwairi, M.; Yassein, M.B. Anomaly-based proxying internet system through feature selection analysis and building hybrid efficient model. J. Comput. Sci. 2018, 25, 152–160.

[5] Kim, "A feature selection approach to find optimal feature subsets for the network Intrusion Detection System," *Cluster Computing*, vol. 19, no. 1, pp. 325–333, 2016.

[6] Salo, F.; Nassif, A.B.; Essex, A. Dimensionality reduction with IG-PCA and ensemble classifier for network proxying internet. Comput. Netw. 2019, 148, 164–175.

[7] Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD Cup 99 data set," *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009.

[8] .J. Rene Beulah and D. Shalini Punithavathani, "A hybrid feature selection method for improved detection of wired/wireless network intrusions," *Wireless Personal Communications*, vol. 98, no. 2, pp. 1853–1869, 2017.

[9] Bostani, H.; Sheikhan, M. Hybrid of binary gravitational search algorithm and mutual information for feature selection in proxying internet systems. Soft Comput. 2017, 21, 2307–2324.

[10] Acharya, N.; Singh, S. An IWD-based feature selection method for proxying internet system. Soft Comput. 2017, 22, 4407–4416.

[11] Alabdallah, A.; Awad, M. Using weighted support vector machine to address the imbalanced classes problem of proxying internet system. KSII Trans. Internet Inf. Syst. 2018, 12, 5143–5158.

[12] Akashdeep, S.; Manzoor, I.; Kumar, N. A feature reduced proxying internet system using ANN classifier. Expert Syst. Appl. 2017, 88, 249–257.

[13] A. AKYOL, M. HACIBEYOĞLU, and B. KARLIK, "Design of multilevel hybrid classifier with variant feature sets for Intrusion Detection System," IEICE Transactions on Information and Systems, vol. E99.D, no. 7, pp. 1810–1821, 2016.

[14] Bhattacharya, S.; Selvakumar, S. LAWRA: A layered wrapper feature selection approach for network proxy detection. Secur. Commun. Netw. 2015, 8, 3459–3468.

[15] Panda, M.; Abraham, A.; Patra, M.R. Hybrid intelligent systems for detecting network intrusions. Secur. Commun. Netw. 2015, 8, 2741–2749.

[16] Ahmad, I.; Basheri, M.; Iqbal, M.J.; Rahim, A. Performance comparison of support vector machine, random forest, and extreme learning machine for proxying internet. IEEE Access 2018, 6, 33789–33795.

[17] Aburomman, A.A.; Reaz, M.B. A survey of proxying internet systems based on ensemble and hybrid classifiers. Comput. Secur. 2017, 65, 135–152.

[18] Lilakiatsakun, W.; Somwang, P. Anomaly traffic detection based on PCA and SFAM. Int. Arab J. Inf. Technol. 2017, 12, 253–260.

[19] A. AKYOL, M. HACIBEYOĞLU, and B. KARLIK, "Design of multilevel hybrid classifier with variant feature sets for Intrusion Detection System," *IEICE Transactions on Information and Systems*, vol. E99.D, no. 7, pp. 1810–1821, 2016.

[20] Li, L.J.; Yu, Y.; Bai, S.S.; Hou, Y.; Chen, X.Y. An effective two-step proxying internet approach based on binary classification and kNN. IEEE Access 2018, 6, 12060–12073.

[21] Demir, N.; Dalkilic, G. Modified stacking ensemble approach to detect network intrusion. Turk. J. Electr. Eng. Comput. Sci. 2018, 26, 418–433.

[22] Kamarudin, M.H.; Maple, C.; Watson, T.; Safa, N.S. A LogitBoost-based algorithm for detecting known and unknown web proxies. IEEE Access 2017, 5, 26190–26200.

[23] Tian, Y.J.; Mirzabagheri, M.; Bamakan, S.M.H.; Wang, H.D.; Qu, Q. Ramp loss one-class support vector machine: A robust and effective approach to anomaly detection problems. Neurocomputing 2018, 310, 223–235.

[24] Kabir, E.; Hu, J.K.; Wang, H.; Zhuo, G.P. A novel statistical technique for proxying internet systems. Future Gener. Comput. Syst. 2018, 79, 303–318.

[25] Ahmim, A.; Derdour, M.; Ferrag, M.A. An proxying internet system based on combining probability predictions of a tree of classifiers. Int. J. Commun. Syst. 2018, 31, 1–14.

[26] Aburomman, A.A.; Reaz, M.B. A novel weighted support vector machines multiclass classifier based on differential evolution for proxying internet systems. Inf. Sci. 2017, 414, 225–246.

[27] Yan, B.H.; Han, G.D. LA-GRU: Building combined proxying internet model based on imbalanced learning and gated recurrent unit neural network. Secur. Commun. Netw. 2018, 1, 1–13.

[28] N. Naik, P. Jenkins, Web protocols and challenges of web latency in the web of things, in: 2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN), IEEE, pp. 845–850.

[29] S. Chen, X. Zhu, H. Zhang, C. Zhao, G. Yang, K. Wang, Efficient privacy preserving data collection and computation offloading for fog-assisted iot, IEEE Transactions on Sustainable Computing (2020).

[30] S. Kumar, V. K. Chaurasiya, A strategy for elimination of data redundancy in internet of things (iot) based wireless sensor network (wsn), IEEE Sys- tems Journal 13 (2) (2018) 1650–1657.

[31] S. Balakrishna, M. Thirumaran, V. K. Solanki, IoT sensor data integration in healthcare using semantics and machine learning approaches, Springer, 2020, pp. 275–300.

[32] M. Shen, Y. Deng, L. Zhu, X. Du, N. Guizani, Privacy-preserving image retrieval for medical iot systems: A blockchain-based approach, IEEE Network 33 (5) (2019) 27–33.

[33] A. Rahman, M. S. Hossain, N. A. Alrajeh, and F. Alsolami, "Adversarial examples—security threats to covid-19 deep learning systems in medical IOT devices," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9603–9610, 2021.

[34] Z. Yang, Q. Zhou, L. Lei, K. Zheng, W. Xiang, An iot-cloud based wear- able ecg monitoring system for smart healthcare, Journal of medical sys- tems 40 (12) (2016) 286.

[35] W.-J. Hu, J. Fan, Y.-X. Du, B.-S. Li, N. Xiong, E. Bekkering, Mdfc–resnet: An agricultural iot system to accurately recognize crop dis- eases, IEEE Access 8 (2020) 115287–115298.

[36] L. Zhou, Z. Qiu, Y. He, Application of wechat mini-program and wi-fi soc in agricultural iot: A low-cost greenhouse monitoring system, Trans- actions of the ASABE 63 (2) (2020) 325–337.

[37] F. Zantalis, G. Koulouras, S. Karabetsos, and D. Kandris, "A review of Machine Learning and IOT in smart transportation," *Future Internet*, vol. 11, no. 4, p. 94, 2019.

[38] D. Rahbari, M. Nickray, Low-latency and energy-efficient scheduling in fog-based iot applications, Turkish Journal of Electrical Engineering Computer Sciences 27 (2) (2019) 1406–1427.

[39] H. Nasiri, S. Nasehi, and M. Goudarzi, "Evaluation of distributed stream processing frameworks for IOT applications in Smart Cities," *Journal of Big Data*, vol. 6, no. 1, 2019.

[40] M.-D. Gonza´lez-Zamar, E. Abad-Segura, E. Va´zquez-Cano, E. Lo´pez- Meneses, Iot technology applications-based smart cities: Research analy- sis, Electronics 9 (8) (2020) 1246.

[41] W. Vogels, Web services are not distributed objects, IEEE Internet com- puting 7 (6) (2003) 59–66.

[42] W. Xu, J. Offutt, J. Luo, Testing web services by xml perturbation, in: 16th IEEE International Symposium on Software Reliability Engineering (ISSRE'05), IEEE, pp. 10 pp.–266.

[43] R. Bonetto, N. Bui, V. Lakkundi, A. Olivereau, A. Serbanati, M. Rossi, Secure communication for smart iot objects: Protocol stacks, use cases and practical examples, in: 2012 IEEE international symposium on a world of wireless, mobile and multimedia networks (WoWMoM), IEEE, pp. 1–7.

[44] M. A. A. Razali, M. Kassim, N. A. Sulaiman, S. Saaidin, A things- peak iot on real time room condition monitoring system, in: 2020 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS), IEEE, pp. 206–211.

[45] V. Babu, V. Rajan, Flood and earthquake detection and rescue using iot technology, in: 2019 International Conference on Communication and Electronics Systems (ICCES), IEEE, pp. 1256–1260.

[46] M. Lavassani, S. Forsstro¨m, U. Jennehag, T. Zhang, Combining fog com- puting with sensor mote machine learning for industrial iot, Sensors 18 (5) (2018) 1532.

[47] P. P. Ray, A survey of iot cloud platforms, Future Computing and Infor- matics Journal 1 (1-2) (2016) 35–46.

[48] N. Naik, Choice of effective messaging protocols for iot systems: Mqtt, coap, amqp and http, in: 2017 IEEE international systems engineering symposium (ISSE), IEEE, pp. 1–7.

[49] T. Yokotani, Y. Sasaki, Comparison with http and mqtt on required net- work resources for iot, in: 2016 international conference on control, elec- tronics, renewable energy and communications (ICCEREC), IEEE, pp. 1–6.

[50] D. Thangavel, X. Ma, A. Valera, H.-X. Tan, C. K.-Y. Tan, Performance evaluation of mqtt and coap via a common middleware, in: 2014 IEEE ninth international conference on intelligent sensors, sensor networks and information processing (ISSNIP), IEEE, pp. 1–6.

[51] D. Bastos, M. Shackleton, and F. El-Moussa, "Internet of things: A survey of technologies and security risks in smart home and City Environments," *Living in the Internet of Things: Cybersecurity of the IoT - 2018*, 2018.

[52] M. Noura, M. Atiquzzaman, M. Gaedke, Interoperability in internet of things: Taxonomies and open challenges, Mobile Networks and Applica- tions 24 (3) (2019) 796–809.

[53] Idhammad, M.; Afdel, K.; Belouch, M. Semi-supervised machine learning approach for DDoS detection. Appl. Intell. 2018, 48, 3193–3208.

[54] Mohammadi, S.; Namadchian, A. A new deep learning approach for anomaly base IDS using memetic classifier. Int. J. Comput. Commun. 2017, 12, 677–688.

[55] Imamverdiyev, Y.; Abdullayeva, F. Deep learning method for denial of service proxy detection based on restricted boltzmann machine. Big Data-US 2018, 6, 159–169.

[56] Ma, T.; Wang, F.; Cheng, J.; Yu, Y.; Chen, X. A hybrid spectral clustering and deep neural network ensemble algorithm for proxying internet in sensor networks. Sensors (Basel) 2016, 16, 1701.

[57] Shamshirband, S.; Daghighi, B.; Anuar, N.B.; Kiah, M.L.M.; Patel, A.; Abraham, A. Co-FQL: Anomaly detection using cooperative fuzzy Q-learning in network. J. Intell. Fuzzy Syst. 2015, 28, 1345–1357

[58] Al-Qatf, M.; Yu, L.S.; Al-Habib, M.; Al-Sabahi, K. Deep learning approach combining sparse autoencoder with SVM for network proxying internet. IEEE Access 2018, 6, 52843–52856.

[59] Hussain, J.; Lalmuanawma, S.; Chhakchhuak, L. A two-stage hybrid classification technique for network proxying internet system. Int. J. Comput. Int. Syst. 2016, 9, 863–875.

[60] Li, L.J.; Yu, Y.; Bai, S.S.; Cheng, J.J.; Chen, X.Y. Towards effective network proxying internet: A hybrid model integrating Gini index and GBDT with PSO. J. Sens. 2018, 6, 1–9.

[61] He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), PIEAS, Islamabad, Pakistan, 26–27 August 2016; pp. 770–778.

[62] Almomani, I.; Alromi, A. Integrating Software Engineering Processes in the Development of Efficient Proxying internet Systems in Wireless Sensor Networks. Sensors 2020, 20, 1375. https://doi.org/10.3390/s20051375.

[63] Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein.; et al. Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. 2016, 115, 1–37.

[64] Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. Comput. Sci. 2016, 9, 1–14.

[65] M. A. Simplício, P. S. L. M. Barreto, C. B. Margi, and T. C. M. B. Carvalho, "A survey on key management mechanisms for distributed wireless sensor networks," *Computer Networks*, vol. 54, no. 15, pp. 2591–2612, 2010.

[66] He, K.; Zhang, X.; Ren, S.; Sun, J. Identity mappings in deep residual networks. In Proceedings of the 2017 European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 11–14 October 2015; pp. 630–645.

[67] Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic Minority Over-Sampling Technique. J. Artif. Intell. Res. 2017, 16, 321–357.

[68] Wu, K.H.; Chen, Z.G.; Li, W. A novel proxying internet model for a massive network using convolutional neural networks. IEEE Access 2018, 6, 50850–50859.

[69] Le, T.T.H.; Kim, Y.; Kim, H. Network proxying internet based on novel feature selection model and various recurrent neural networks. Appl. Sci.-Basel 2019, 9, 1392.

[70] Panda, M.; Abraham, A.; Patra, M.R. Discriminative multinomial naive Bayes for network proxying internet. In Proceedings of the 2017 Sixth International Conference on Information Assurance and Security, Atlanta, GA, USA, 23–25 August 2017; pp. 5–10.

[71] Zhang, W., Han, D., Li, KC. et al. Wireless sensor network proxying internet system based on MK-ELM. Soft Comput 24, 12361–12374 (2020). https://doi.org/10.1007/s00500-020-04678-1.

[72] Gogoi, P.; Bhuyan, M.H.; Bhattacharyya, D.; Kalita, J.K. Packet and flow based network intrusion dataset. Contemp. Comput. 2017, 306, 322–334.

[73] Yin, C.; Zhu, Y.; Fei, J.; He, X. A deep learning approach for proxying internet using recurrent neural networks. IEEE Access 2017, 5, 21954–21961.

[74] Singh, R.; Kumar, H.; Singla, R.K. An proxying internet system using network traffic profiling and online sequential extreme learning machine. Expert Syst. Appl. 2015, 42, 8609–8624.

[75] Yang, Y.Q.; Zheng, K.F.; Wu, C.H.; Niu, X.X.; Yang, Y.X. Building an effective proxying internet system using the modified density peak clustering algorithm and deep belief networks. Appl. Sci.-Basel 2019, 9, 238.

[76] Kayacik, H.G.; Zincir-Heywood, A.N.; Heywood, M.I. Ahierarchical SOM-based proxying internet system. Eng. Appl. Artif. Intell. 2017, 20, 439–451.

[77] Tsang, C.H.; Kwong, S.; Wang, H. Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly proxying internet. Pattern Recognit. 2017, 40, 2373–2391.

[78] Afzali, S.F.; Cotton, J.S.; Mahalec, V. Urban community energy systems design under uncertainty for specified levels of carbon dioxide emissions. Appl. Energy 2020, 259, 114084.

[79] Zhang, B.; Hu, W.; Cao, D.; Huang, Q.; Chen, Z.; Blaabjerg, F. Deep reinforcement learning–based approach for optimizing energy conversion in integrated electrical and heating system with renewable energy. Energy Convers. Manag. 2019, 202, 112199.

[80] Guangjie, H., Shen, W., Duong, T. Q., Guizani, M., and Hara, T. (2014), A proposed security scheme against Denial of Service proxies in cluster- based wireless sensor networks, Security Comm. Networks, 7, 2542– 2554, doi: 10.1002/sec.373.