REPUBLIC OF TÜRKİYE

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering

A NEW METHOD FOR DIRECT IOT DEVICES COMMUNICATION WITH SIMULTANEOUS OPTIMIZATION OF SYSTEM PERFORMANCE

Luma Abdulnaser ALSAMARRAY

Master's Thesis

Supervisor

Prof. Dr. Osman Nuri UÇAN

Istanbul, 2023

A NEW METHOD FOR DIRECT IOT DEVICES COMMUNICATION WITH SIMULTANEOUS OPTIMIZATION OF SYSTEM PERFORMANCE

Luma Abdulnaser ALSAMARRAY

Electrical and Computer Engineering

Master's Thesis

ALTINBAŞ UNIVERSITY

2023

This thesis titled A NEW METHOD FOR DIRECT IOT DEVICES COMMUNICATION WITH SIMULTANEOUS OPTIMIZATION OF SYSTEM PERFORMANCE prepared by LUMA ABDULNASER ALSAMARRY and submitted on 07/04/2023 has been **accepted unanimously** for the degree of Master of Science in Electrical and Computer Engineering.

Prof. Dr. Osman Nuri UÇAN
Supervisor

Thesis Defense Committee Members:

Prof. Dr. Osman Nuri UÇAN	Department of Electrical and Electronics Engineering, Altınbaş University	_____
Asst. Prof. Dr. Sefer KURNAZ	Department of Computer Engineering, Altınbaş University	_____
Assoc. Prof. Dr. Adil Deniz DURU	Department of Trainer Education, Marmara University	_____

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

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

I hereby declare that all information/data that had been presented in this graduation project was obtained in a full accordance with the ethical conduct and academic rules. I declare as well that all of the unoriginal materials and conclusions were cited in the text and all of the references that have been mentioned in the Reference List were cited in the text, and vice versa as it is required by the aforementioned rules and conducts.

Luma Abdulnaser ALSAMARRAY

Signature

DEDICATION

To my Parents, the reason of what I become today. To my sister & brothers. To my soulmate my biggest supporter my husband. A special thanks to Prof. Dr. Osman for his support, worth notes and close follow up.

ACKNOWLEDGEMENT

First, I would like to express my gratitude to my supervisor Prof. Dr. Osman Nuri UÇAN for the guide and help that he had provided to me along the path in writing the present dissertation. Discussing my progress, ideas and problems with my supervisor Prof. Dr. Osman Nuri UÇAN, a few times each week, helped me a lot to understand the research's logic. It made me better realize technical needs for the present study.

ABSTRACT

A NEW METHOD FOR DIRECT IOT DEVICES COMMUNICATION WITH SIMULTANEOUS OPTIMIZATION OF SYSTEM PERFORMANCE

ALSAMARRAY, Luma Abdulnaser
M.S., Electrical and Computer Engineering, Altınbas University,
Supervisor: Prof. Dr. Osman Nuri UÇAN

Date: April / 2023
Pages: 84

The Internet of Things (IoT) has become increasingly popular in recent years, leading to a significant growth in the number of devices that require internet access to operate. However, IoT devices have limited resources, which can make accessing internet services challenging. To overcome this limitation, researchers have proposed lightweight protocols such as Constrained Application Protocol (CoAP) and MQ Telemetry Transport (MQTT) to reduce resource consumption. This study proposes a new system that utilizes these lightweight protocols to proxy internet services, which allows IoT devices to meet their requirements. The research conducted experiments on IoT devices and found that the proposed system reduces resource consumption in terms of network bandwidth, time, memory, and energy. This improved efficiency enables IoT devices to perform the same tasks with fewer resources, making the system highly efficient. Overall, the proposed system could be a solution to the challenges posed by limited resources on IoT devices, making accessing internet services more accessible and efficient. The results of this research provide valuable insights into the potential benefits of utilizing lightweight protocols and the impact of these protocols on IoT devices' performance.

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

TABLE OF CONTENT

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

ABBREVIATIONS

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

1. INTRODUCTION

1.1 INTRODUCTION

Since 1974, 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 for the purpose of processing 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.

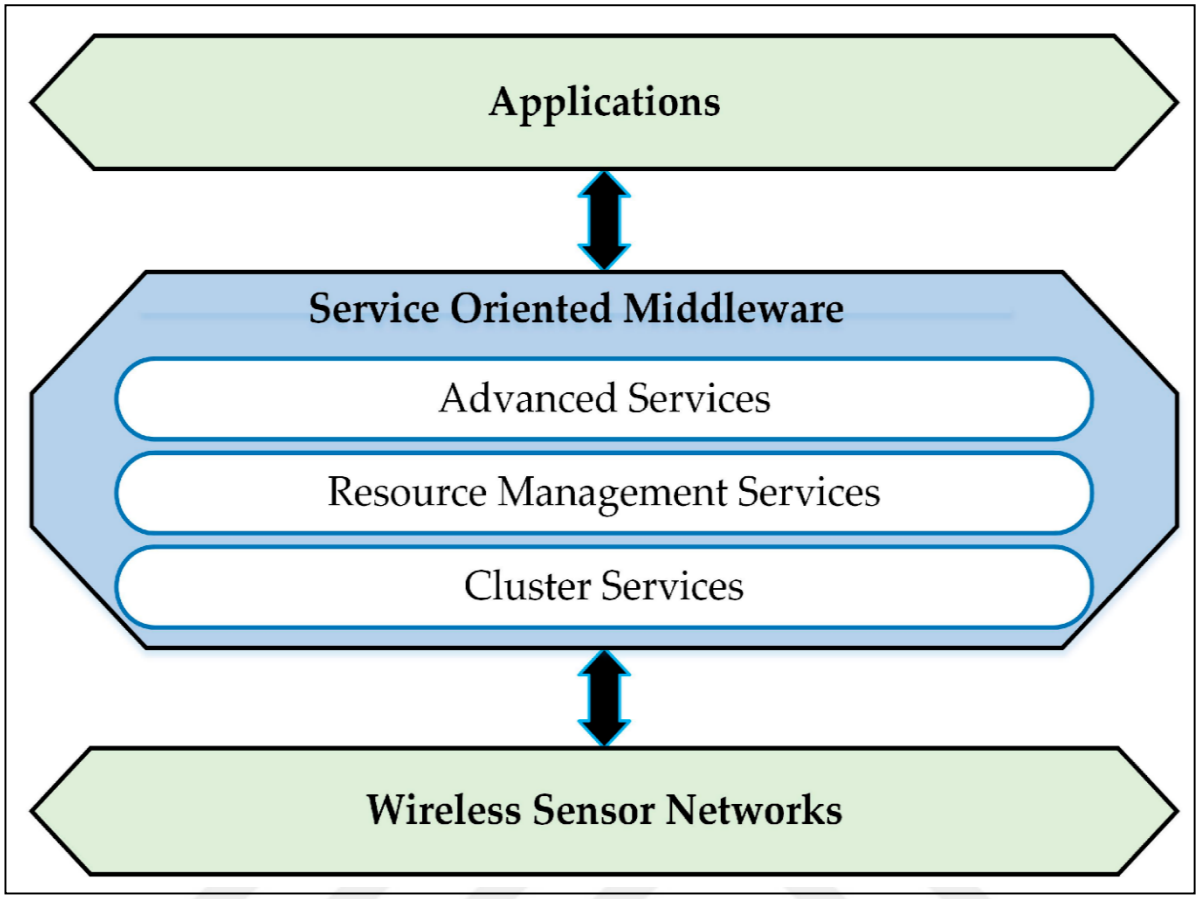


Figure 1.1: Wireless Sensor Networks' Provisioning Schema [8].

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 is of high importance in detecting any attempt to compromise the information regarding 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 American Consumer Institute Center for the 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 transports, 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) are 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, their present state isn’t enough for meeting the requirements 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 throughout penetration test, collecting network traffic and utilizing this as dataset for checking implemented countermeasures. Thus, better outcomes are achieved. There is no tool available publicly to complete this type of independent verification effectively and reproducibly.

1.2 MOTIVATION

Various software systems include memory corruption vulnerabilities or exploitable logic flaws. Among these are buffer overflows. A crash or even remote code execution may result from the overfilling of one section of memory with data while overwriting the next section. Another frequent illustration is post-free vulnerabilities, which, as described in [24], indicate to the act of accessing memory after being released. Additionally, it may lead to the execution of a program crash or arbitrary code. Those errors affect programming language and its variants without automatic management of the memory because boundaries checking errors either by humans or compiler optimizations that do away with bound checks might lead

to them. Languages with such automatic memory management, though, are not reduced to proxies. For instance, Java VM and object serialization have both been exploited. Due to their extensive protocol stack implementation and abundance of potentially weak components, like extensions and plugins, web browsers are important target for proxies [25]. Adobe Flash, a programming language for making animated web content, is an often-exploited part of web browsers. Even huge corporations that lavish a lot of cash on security team and provide bug bounty for reported flaws fail to consistently shield their users or themselves from proxies. In latest Facebook incident, proxyers used technical flaw to obtain access tokens, enabling them to enter into the accounts of nearly 50 million individuals. Apple lately patched a flaw that caused heap overflow and may possibly be used for the remote code executions (CVE2018-4407) in XNU kernel's ICMP packet error handling code. According to [26], the vulnerability is present in such a core component regarding the networking code of the OS that antivirus software is not capable of identifying the attempts of exploitation. Additionally, organizations managed by engineers who are aware of the consequences of the computer security, like spyware producer Hacking Team, have in the past had an embedded device in their network compromised by a zero-day vulnerability [27]. Because of a mistake in the handling of Session Initiation Protocol (SIP) packets. The flaw enables a remote crash or reload of the vulnerable systems by a proxyer. The forensic investigation tools expand the proxy surface even more. Because of the quantity of implemented code and supported protocols that needs to be maintained, protocol dissectors have a large proxy surface. For instance, Wireshark has 1400 writers, has a minimum of 2.5 million lines of code overall, and supports over 2400 protocols. Present software stacks could be exploited and compromised because of their dynamic nature and complexity [28]. In order to enhance the proxyer's efforts and expenditures and ensure quick discovery of breaches, they must be monitored. Lastly, network monitor could be a proxy target.

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]. CoAP protocol depends on UDP to decrease the processing needed by the device. The UDP is very lighter than TCP (implemented by HTTP protocol). Furthermore, CoAP is utilized for accessing the web services. It transmits (receives) data to (from) server. The MQTT protocol is typically utilized to send and receive data among devices. The protocol is known as a publish-subscribe protocol because Internet of Things devices subscribe to the topics that interest them in their texts [22, 24].

Despite reducing the resources needed by the devices for entering their wanted services by such 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]. Thus, this study presents a unique approach using a cloud server for proxying all internet services for IoT devices.

Effective feature extraction, selection, and collection tooling are required for solving such queries. Regrettably, a lot of researchers do not publish the tools that they use, and that makes it difficult to validate and replicate their findings and lengthens the time required for subsequent research on the subject. Additionally, depending on fabricated data-sets for the network security monitoring verification doesn't accurately reflect the actual condition inside the secured environment of the network.

1.4 RESEARCH CONTRIBUTIONS

The suggested system contributes to the following objectives:

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

ii. Interpreting messages between IoT devices and the intended service devices depending upon protocols that have been defined for every Application Programming Interface (API).

iii. Storing redundant data in server to communicate the only sensed data via device.

iv. Extract the part of the data and send it with no further data as the need of device from wed service.

1.5 THESIS ORGANIZATION

The rationale behind cluster-based anomaly-based detection methodologies and network security monitoring are explained in the presented introduction. It describes the effects of memory corruption on the security related to the networking infrastructure used by WSNs nowadays. To clarify the purpose of the presented work, common terminologies will be provided together with a task description and a problem statement. The requirements for network feature collection mechanisms are examined after a brief review of related work, and then available solutions are assessed. The implementation and concept of research project will then be thoroughly explained, along with suggestions for future developments and a number of use cases beyond network proxying internet. In the final assessment, a number of useful applications of the created tool are displayed, including a number of tests on categorizing malicious behavior using a WSN. An analysis of potential future developments follows the conclusion.

2. THEORETICAL BACKGROUND

The purpose of the presented study is to develop better environment solution which will eliminate drawbacks of conventional wireless network for proxying internet system and controlling operations. In addition, the presented work is motivated to overcome the complications and challenges that proxying internet system units in WSN faces using clusters methods in real time with their structure and hostile environment is surrounding them. They also provide an in-depth analysis of WSN systems and their benefits, which this study tries to overcome.

2.1 RELATED WORK

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] presented a thorough 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 network proxying internet system and presented the network monitor’s structure and operations. 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 with the use of Bayes Classifier and Support Vector Machines (SVM). We were inspired to inquire about the necessity of extremely complex extracted characteristics

by the study. Furthermore, a more basic feature set is key to performing the current research tests.

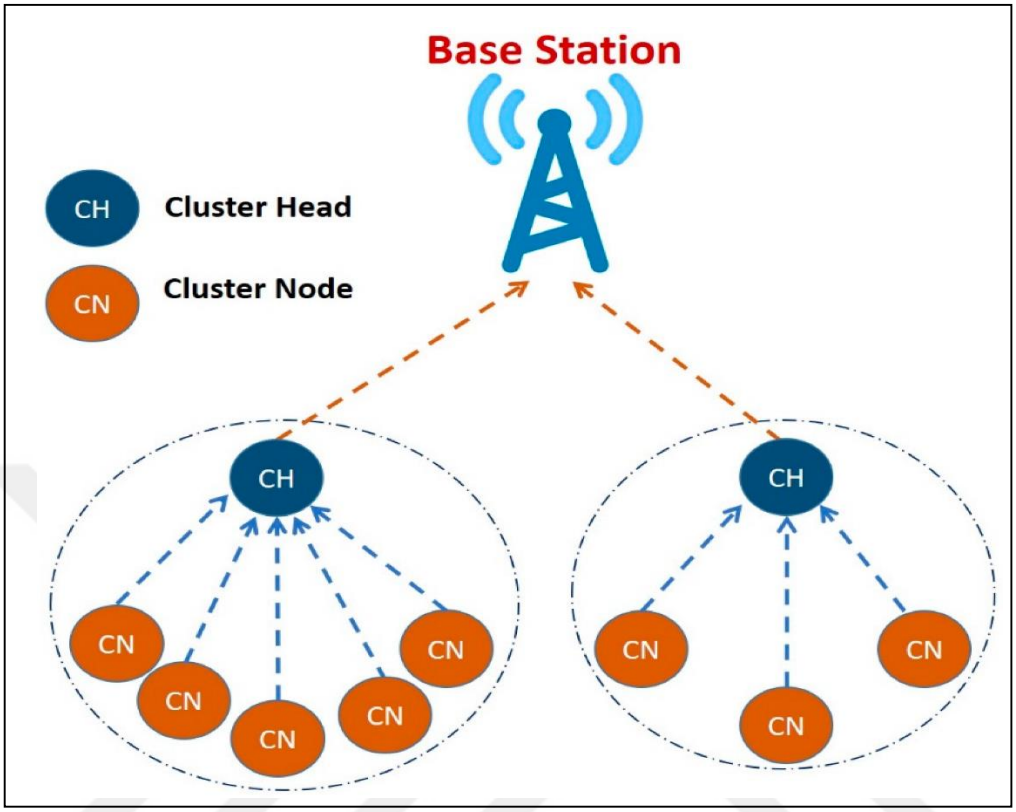


Figure 2.1: A Central Station Design with Two Heads of Clustering and Many Nodes Within [37].

They do not appear to have shared their implementation, which is unfortunate because it may have been of tremendous importance to research community [38]. The researcher was strongly motivated to publish the entire tool chain for helping future researchers and enable independent verification of the results because results couldn't, thus, be reviewed and future study will require to proceed by using inefficient tooling.

The current traffic flow analysis was examined by researchers in [39], who also included a comparison and an overview of popular tools and capture formats. The flow-based collecting mechanism was effectively introduced.

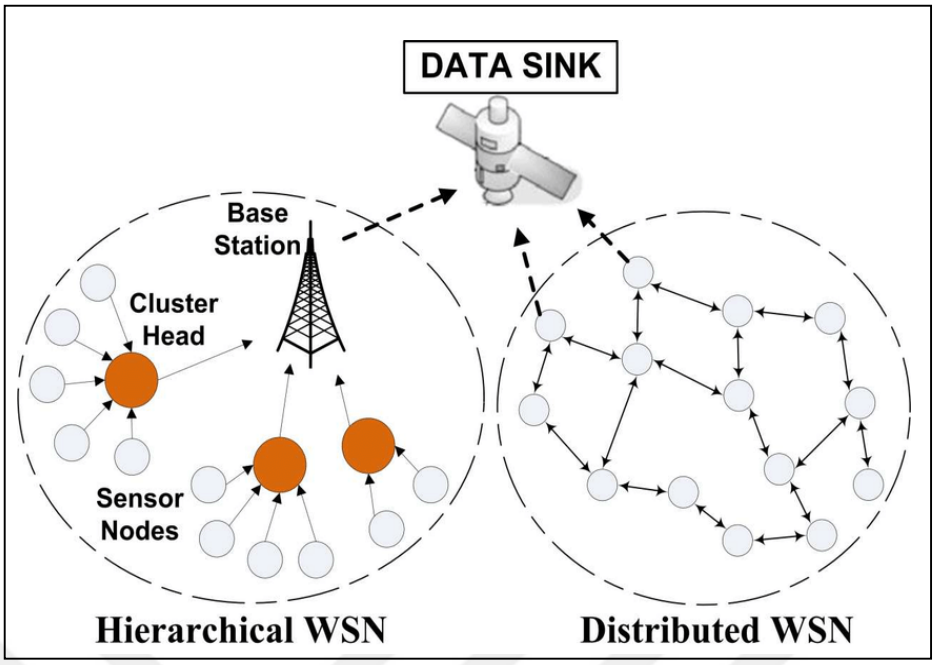


Figure 2.2: Evaluation of The Theoretical Representations Pertaining to Distributed and Hierarchical Wsns [40].

2.2 BEHAVIORAL DEMANDS

To obtain the most comprehensive view of network activity, the feature collector should be able to parse a broad range of the protocols. The system should also report traffic that it cannot comprehend.

2.2.1 Data Availability

Prior to creating summary structures, network monitor must make collected data accessible at minimal level that is possible for the research purposes. This means that for each protocol that the monitor could interpret, protocol-specific information must be provided.

2.2.2 Abstraction Capabilities

Summary structures, abstractions, aggregate values, and statistical data could be produced from acquired data with the use of information which was acquired during the collecting stage. Bidirectional connections and unidirectional network flows are common examples of summary structures.

2.2.3 Concurrent Design

After developing reduced-cost multi-core processors, many systems have been equipped with this design. In the beginning, there is high importance in considering 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. Through parallelizing their decoding process, multi-core systems' power is used to boost the overall 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 instance, matching executables to antivirus (AV) data-bases. Breaking the encryption for that reason shouldn't be considered since it reduces system security overall, which represents a negative outcome.

2.2.5 Supported Input Format

With regard to 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 major problem, particularly when storing data for extended periods of time. Other formats, like Cisco's NetFlow, Suitable Output Format.

Output format for data that had been gathered needs to be readable through a variety of libraries, frameworks, and systems and displayable in a human-readable form. The output format of choice is frequently CSV. As a result, it isn't restricted to any one-character set and functions equally well with Unicode and ASCII [44]. The character set that is used presently should be supplied individually because CSV doesn't preserve it. Even though CSV is space-inefficient method of the storage or the exchange of the information, people are readable in their original form. Also, without separating structures from their parent environment, CSV can't represent the object-oriented and hierarchical structured data.

2.2.6 Real-Time Operation

It is important to support live capture from network interface for monitoring and operating real-time. To disc, dumping the collected data cannot 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 likelihood of mistakes and give fewer periods for the cleaning of the 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 happen.

2.2.7 Non-Functional Requirement

In case untrusted inputs are parsed, it should be produced that no exploitable vulnerabilities would be available in the code of the parser. 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.

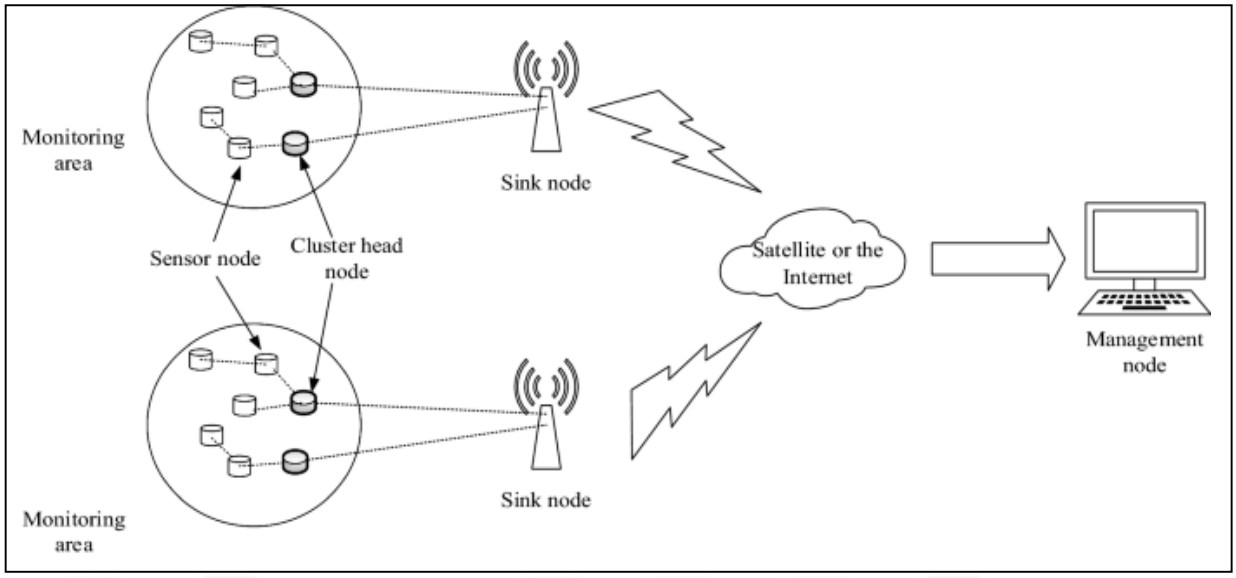


Figure 2.3: In A Wireless Sensor Network, Equality Constrained Optimization with Management Node and Sensor Nodes [46].

2.2.8 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 some language that has memory-safe garbage collected run-time.

2.2.9 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 must depend upon a secret code.

2.2.10 Scalability

In order 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 any eavesdropping.

2.2.11 Performance

The quantity of the packets (which should be examined and recorded) keeps increasing. To prevent data loss and save all data, the network monitor needs to be effective. Even though an alert doesn't prevent intrusions automatically, its identification is merely a start and increases the chance for a proxyer (to be detected). Missing the finding of an active proxy could result from a system which is overloaded and begins to drop packets. The implementation's programming language has a significant impact on performance. The programming languages of low-level systems don't offer any memory safety, as was previously mentioned. For the architecture of security-critical infrastructure, higher level languages like Python programming must be taken into account in future. Although there is a performance cost associated with this, the overall application security is greatly benefited by this trade-off. In a newly released study, POSIX compliant kernel was written by using Python, and the performance penalty assessed ranged between 5% and 15% when compared with the Python implementations. Networks in the present day carry a variety of the heavy traffic streams, including torrent downloads, Netflix-style video streaming, and others. As indicated in [47], it is customary practice to identify such "elephant flows" and eliminate them as soon as feasible since they significantly reduce the performance of monitoring system. This offers a security concern, too, since an intermediary may leverage the knowledge that specific traffic types are being ignored for the purpose of hiding in big data stream. Discarding such flows could result in missing a genuine incursion caused by a media player vulnerability, as video and audio players and associated codecs have been considered as one of the popular sources for the security flaws. For instance, looking for VLC media player in MITRE CVE database yields 92 results, as indicated in [48]. To achieve performance objectives, an ideal solution must not reject any data.

2.2.12 Configurable Design

For allowing performance tuning and adaptation to unique requirements, like privacy rules, the monitor program must be flexible. Automated network traffic collection and analysis could be illegal on a state, local, and national level and could lead to serious privacy violations. Thus, before installing any monitoring software, it is vital to acquire the needed rights for the target network and professional legal counsel. For various network administrators, the issue of how to preserve user privacy while conducting network security monitoring

operations has become crucial. Since may 28, 2018, General Data Protection Regulation (GDPR) has dramatically increased the penalties for violations and the range of protected information as described in [49], considerably enhancing the protection of personal data from EU individuals.

2.2.13 Extensibility

Computer networks represent tremendously dynamic settings; therefore, the network monitor must be simply expandable, quickly integrate new protocols, and keep up with the lightning-fast pace of technological advancement. Thus, the monitoring system must make it secure to quickly and securely install parsers for additional protocols and must offer an interface to further expand capabilities.

2.2.14 Reliability

Even in proxy or high load situations, the monitor should often be accessible and produce actionable data. Additionally, critical errors should not cause the monitoring system to shut down; instead, they should inform an administrator and restore system to a functioning state, as described by [50]. It's crucial to provide ongoing monitoring when recovering from a system failure, regardless of whether it was caused by a defect induced by a proxy or by logical errors. The monitor will be vulnerable to proxies, therefore recovering and reporting from undecodable traffic or parsing errors is required.

2.2.15 Usability

The network monitor's user interface needs to be strong enough to handle challenging tasks, yet sufficiently simple to be operated by human being with no confusions or wastage of the time. In time to help people quickly comprehend, correlate, and analyze vast amounts of high dimensional data, the software should offer appropriate visualization options because network data is quite abstract for humans.

2.2.16 Storage Efficiency

Since state-of-the-art networks can only store so much data at once, the amount of audit data that is gathered must also rise. Since the resulting output must contain all the information necessary for the detection task described in [51], it must be as compact as possible. The data that will be collected must also be customizable. Data that is unrelated to a detection

task or a specific monitoring system may exist in some instances. If so, there must be no waste of resources on gathering or storage of the irrelevant data.

2.3 COLLECTION OF THE FEATURES IN THE IDS SYSTEMS

The systems of feature collection that are used for proxying the internet must be sufficiently dependable for running continuously in background related to observed system with little overhead and with no human supervision. For identifying proxies attacking the monitor directly, the system should be fault tolerant, capable of recovering from errors automatically, and must maintain its own integrity. Extensibility and reconfigurability are crucial, as modifications are frequently needed in dynamic contexts like computer networks. The system should be capable of detecting deviations from expected behavior, and deployment must be simple and quick. They also need to offer an effective user interface and visualizations in order to help the human analysts. To guarantee performance and integrity of monitoring systems, the mentioned criteria for the underlying hardware are equally important. For future reference, table 2.1 includes a list of each software required for contemporary network proxying internet systems.

Table 2.1: Summary of The Requirements for The Feature Collection [52].

Supported Input Format	Support for the standard input formats of the industry
proper Output Format	Widely supported and efficient formats of output
Real Time Operations	Live acquisitions of the traffic from an interface of the network
Open-Source Codebase	Publicly available source code
Memory Safety	Secure management of the memory
Performance	sufficient processing for the traffic of high volume
Scalability	Operations in the large scales, potentially distributed networks
Extensibility	extension Ease for the new protocols
Configurable Design	adaption and Configuration options for the special requirement
Usability	simplicity of the utilization for analysts / researchers
Reliability	Reliable implementations
Storage Efficiency	Low storage overhead
Requirements	Short Descriptions
Coverage of Protocol Support	Range of the supported protocols
Capabilities of Abstraction	Generations of the abstractions like the flows
Availability of the Data	Access to the data on lowest levels
Extraction of Files	Extractions of the files from the network traffic
Concurrent Design	Efficient utilization of the multi-core architecture

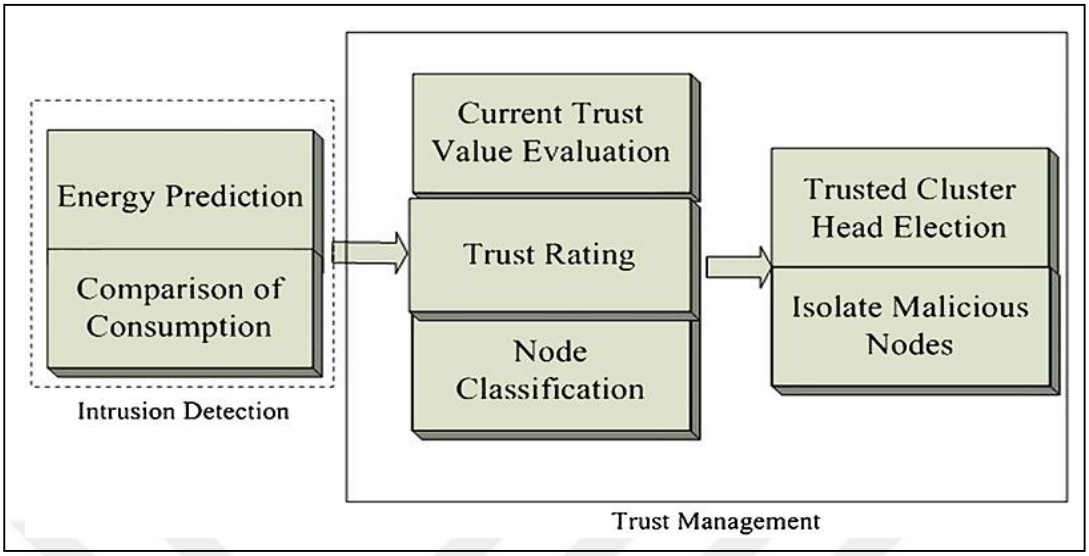


Figure 2.4: With Two Modules in Security Scheme, Network Re-Configurability, And Extensibility for Proxying Internet [53].

2.4 BENEFITS OF INTERNET-OF-THINGS

19.71 billion connected devices to the IoT were present worldwide in the year 2019. According to [54], it will reach 26.66 billion before the end of 2020. According to [55], there will be 75.44 billion gadgets connected to the Internet in 2025, which is an increase of several orders of magnitude over the previous ten years. Those Internet-connected gadgets are capable of independently producing information based upon the behaviors or the anticipated results and providing it via Internet. Which is what IoT concept is made from. According to [56], the IoT is a vast network of gadgets with actuators or sensors linked via remote or wired organizations.

In addition to all advantages and positive aspects which remote medium had added to the IoT, there are drawbacks, such as numerous kinds of the intermediaries and irregularities in the traffic. The two most well-known security risks include the infection and intrusion, and those are typically alluded to as programmers or wafer. This goes along with the risk of personal information being shared with the public and leads to an increase of digital intermediaries such as phishing, probe, Remote to Local (R2L), denial-of-service (DoS) intermediary, and so on. The ordinary information stream contains an example that could be roughly similar because such intermediaries are not typical [57]. However, the scenario will change

if another material tries to alter or modify the data, and this is the point at which intrusion location techniques become a crucial consideration for the web, and particularly for IoT.

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, which include: Signature-based recognition and Anomaly-based identification. Abnormality-based recognition strategies distinguish the intermediaries which are obscure by noticing the entire framework and 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 indicated via 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 examination as referenced in [61].

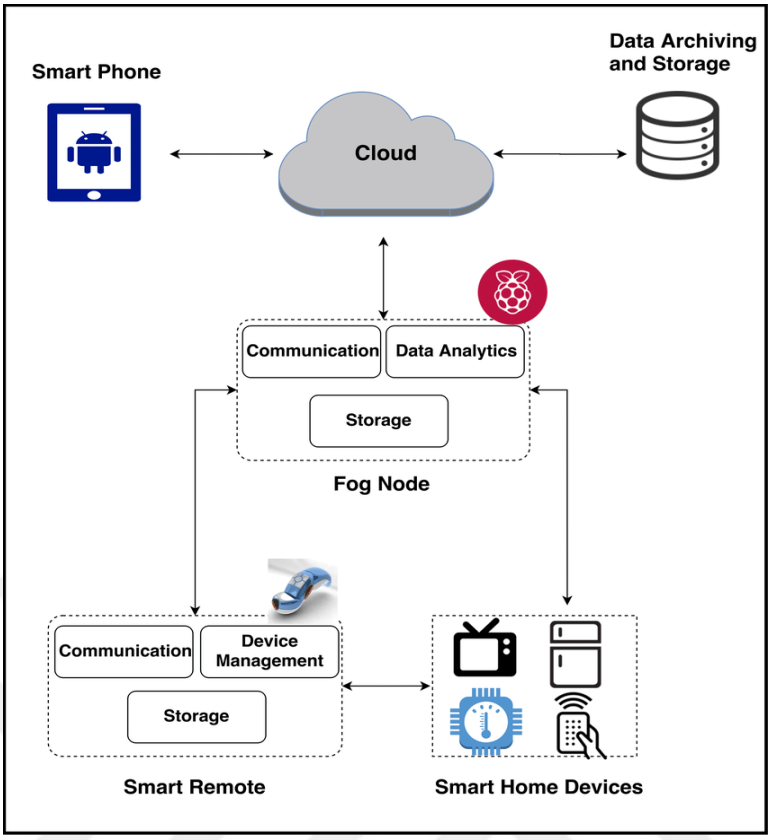


Figure 2.5: An Intelligent Proxy Transformation Model with The Storage Management and Data Archiving System Inside Proxy [61].

2.5 IOT SUPPORT IN THE INTELLIGENT PROXY AUTOMATIONS

2.5.1 IoT Advantages

The IoT offers a number of daily benefits, as indicated in [62], not just for the consumer sector, yet also for the business sector. We can now examine real-time information which was previously unavailable thanks to technology. By lowering material waste and unplanned downtime, businesses may increase the efficiency of their operations. AI is applied, which significantly reduces the effort and time required from humans. Infrastructure construction and infrastructure damage detection could both be done using sensors. Road congestion might be decreased with the aid of automated traffic control, and exterior sensors or gadgets might be installed to detect environmental changes and alert us to approaching natural disasters.

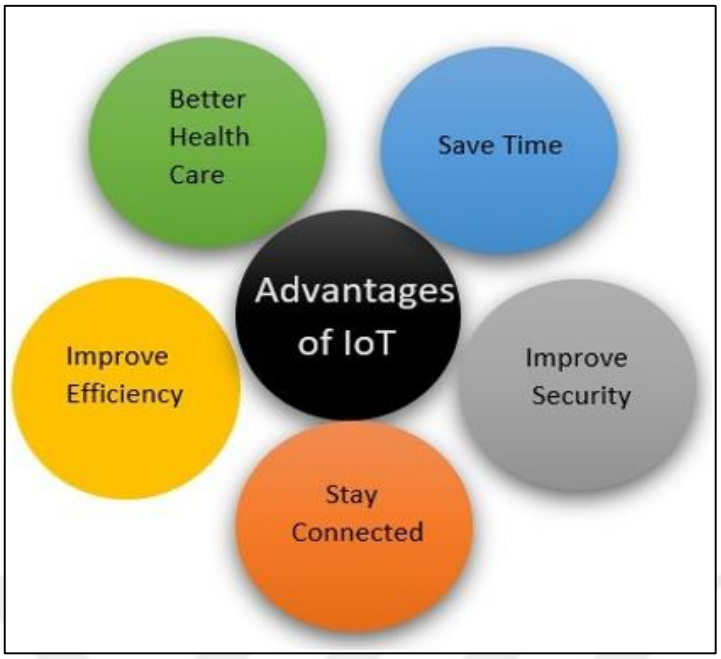


Figure 2.6: Advantages of IoT [62]

IoT improves the quality of our lives by making our homes, workplaces, and vehicles more measurable and smarter. It is incredibly simple to obtain information on the go, set timers, or play music with smart speakers like Google Proxy or Amazon's Echo. We could remotely monitor our proxy from other locations and see what's occurring both outside and inside, as well as interact with visitors thanks to the proxy security system. Smart air conditioners will turn on in the case where it detects that you are enroute to your proxy, and smart lighting systems will switch off if no one is home, reducing excessive energy consumption.

2.5.2 IoT Advantages

IoT security has two disadvantages. As indicated in [63], the fact that there is currently an interconnected device system on a network helps to make system effective and secure to some extent. Although the connected devices produce enormous amounts of the data and are continually exchanging data, this leads to a large volume of internet traffic. The devices may become more valuable with such data, yet the internet's openness raises concern about privacy and security. There are several causes for such problems.

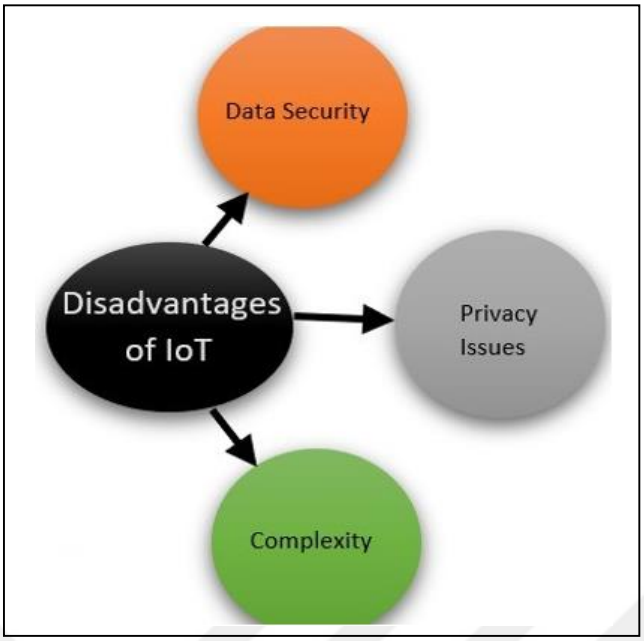


Figure 2.7: Dis Advantages of IoT[62**]**

2.5.3 IoT-Disconnection in Smart-Proxy

Because IoT gadgets cannot be fixed, programming errors are frequently detected in them. As a result, they are more vulnerable to security risks such as DDoS intermediaries, which is likely the most well-known problem caused through setting device's secret key to the default secret phrase that the programmers may actually break, as mentioned in [64]. Both malware and ransomware use encryption to completely lock out its client' gadgets and steal their data. Protection continues to be one of the main IoT challenges as data is transferred, stored, processed, and harassed by the large corporations, as it has been stated by [65]. Our rights to data security and privacy are violated when this data is sold to different businesses. One of the most significant risks associated with IoT is proxy invasion, as the majority of proxies and workplaces depend on automatic procedures that encourage using IoT devices in such settings. Those devices' IP addresses are available to the hackers, who could utilize them in order to determine the device location and acquire access to the personal information in order to sell it to darknet markets that are data of the criminal activity.

2.5.4 IoT-Disconnection in Smart-Proxy

Disconnection from the IoT occurs in smart proxies for a variety of causes. It can be due to a faulty node. Other causes may be related to node movement, which may occur randomly to one or both nodes of each link or periodically as a result of satellite and planet movements on their orbits. The fact that certain cutoffs happen as a result of periodic power savings by nodes, such as low power nodes in sensor networks, is also important to note.

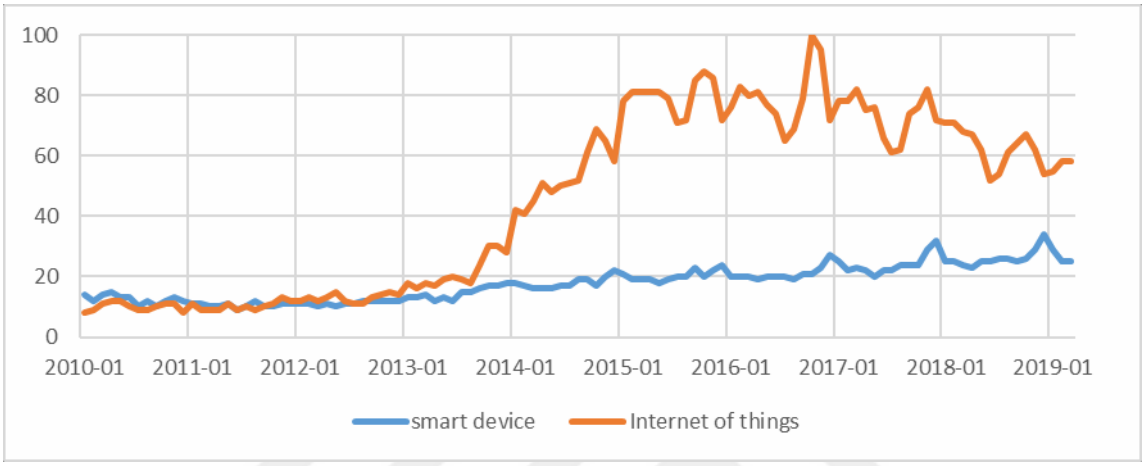


Figure 2.8: The Phrases "Smart Devices" and "IoT" Have Shown an Increase In Popularity Over Time, According to Google Trends Between 2010 And 2019[66].

2.5.5 Unexpected Lengthened Queuing in the IoT Networks

In the severe circumstances, queueing delay in typical IoT networks might range from a few seconds of a second to seconds. However, because of the unusual circumstances surrounding smart proxy queueing, delays might last days or even hours. The nature of smart proxies, in which several copies are transiting across the nodes, as well as the retransmission that occurs in the case when data cannot reach their destinations, are what contribute to these longer queueing delays. Similar to the Samsung Smart Things, Intermediary Automation was advanced by the Belkin Company. Which had provided a complete smart intermediary framework along with pre-built smart applications and gadgets (online and portable) for managing and controlling the smart intermediate framework. Its tech is just a little different from the Samsung Smart Things since, because gadgets are WiFi enabled, they can be controlled directly from applications without the need for a mediator center. Also, the framework creates a clever intermediary organization by using WiFi switch. It is crucial to have a central point, like a Raspberry Pi as mentioned in [67], to enable control from outside the intermediate. Belkin WeMo Proxy Automation Framework, such as Samsung

SmartThings, does not offer the flexibility to add specially created gadgets without first adding WeMo item (WEMO Maker) for connecting with any additional gadgets, in spite of whether the customers create gadgets that communicate via WiFi as mentioned in [68]. The environment in which the web of things is used on the plant. Urban areas have a tremendous socioeconomic impact on the world's population and play a vital role in global finance. It fits in with the group of megacities that [69] mentions will have 10.5 million residents by 2025. Depending on whether megacities are viewed as a negative or positive factor for the strategy on a realistic course of events, the situation may be either fortunate or bad. Through the attempt to relate "nations with largest number of the megacities" to "World Environmental Performance Index," it is generally concluded that there isn’t any connection between the countries that have a high proportion of their populations that live in the megacities and their capability for managing their natural resources, as mentioned in [70].

3. METHODOLOGY

Components of the suggested system include: 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 a server (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 isn't communicated from the device.

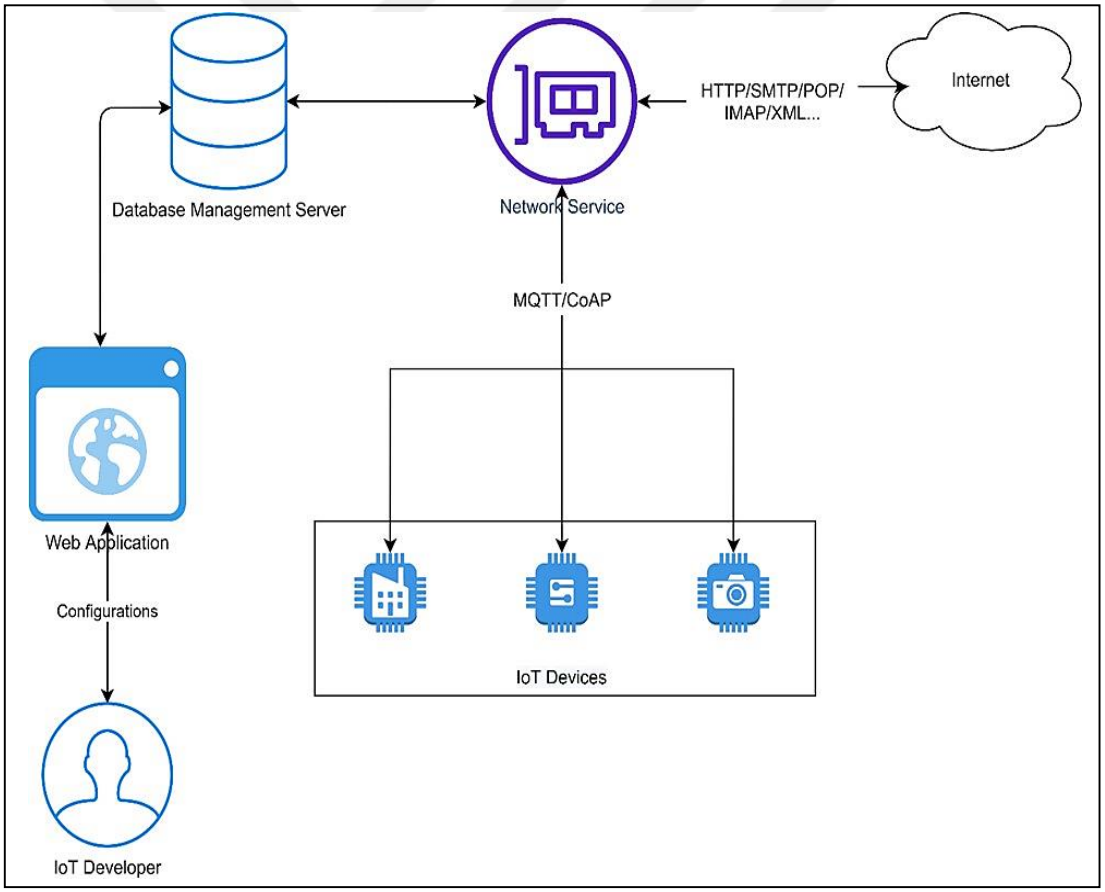


Figure 3.1: Hierarchy of Suggested System

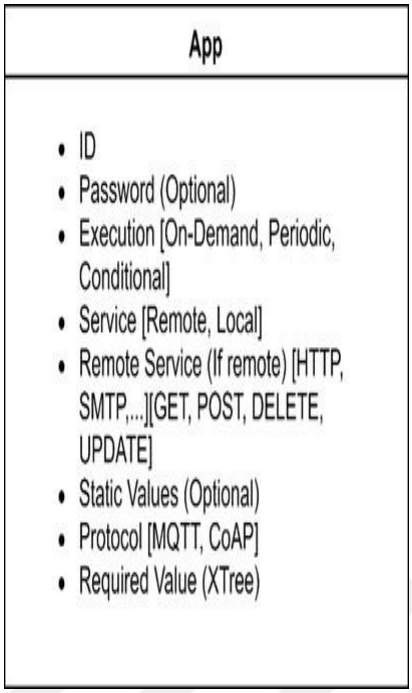


Figure 3.2: App Configurations of The Suggested System.

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 triggering by utilizing 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 CoAP protocol, is utilized for accessing APP when triggered and configured. Therefore, execution of an App has been performed, and the outcome is sent back to device that requests the App.

An App has UPDATE, DELETE, GET, or POST commands depending on REST architecture and uses HTTP protocol for accessing the service. The App also includes any data that is required in order to carry out command on remote server which delivers targeted service. The device needs to write some specified value to some specific server which necessitates the authentication. Both detected value and the name of the value

being written. The system could prestore the remaining data together with the sensed value. As a result, it is possible to access remote service. The sensed value is after that written by the device to the proposed system after the measured value has been given. The CoAP protocol is applied here. Finally, response that has been obtained from remote service has been compared to the XTree. The position of needed value is defined by XTree in received XML response or HTTP so that only that value would be sent back to IoT devices. In the case where no trees have been specified, response is sent back to device which had requested service.

3.2 ACCESSING 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 was previously indicated, the protocol that an IoT device could utilize for accessing an app is based on how the app is executed; periodic and conditional apps utilize MQTT, while on-demand apps utilize CoAP. The IoT device could get updates on value that it's interested in each case that a change has been detected in its value with the use of the protocol of MQTT through the subscription to the appropriate task topic that has been identified by the ID. An on-demand App may be accessible with the use of CoAP protocol through giving all dynamic values that are needed from IoT device for accessing remote service, in the case where any has been defined in the specifications of the App. Those values have been added to template that has been specified at server that has been utilized for accessing remote service, in which the response of service has been put to comparison with predefined XTree for extracting the necessary piece of information and transmit 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 for measuring the physical features of the location, like 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. All customers must have an interesting identification that matches the one described in client consents because our framework has a client authorization component in the home server. If they don't, they'll be given the default client consents. The authorizations might be including access right to certain devices or group of the devices, for example, there is a possibility to state that a client might access benefits of all kitchen appliances but can’t change them. We have recently heard more regarding home computerization, the IoT, and fake power perception than ever before. While the way such terms sound makes them similar, there are other ways in which they differ. This section will provide performance information for IoT-based smart home automation systems generally. This includes the protocol of the MQTT for managing robotization interaction with the modules of the home server. Python 3.8 on the boa constrictor pilot platform with GPU and CPU usage is used as the language of choice for this section. Figure 3.3 displays the fundamental flow diagram.

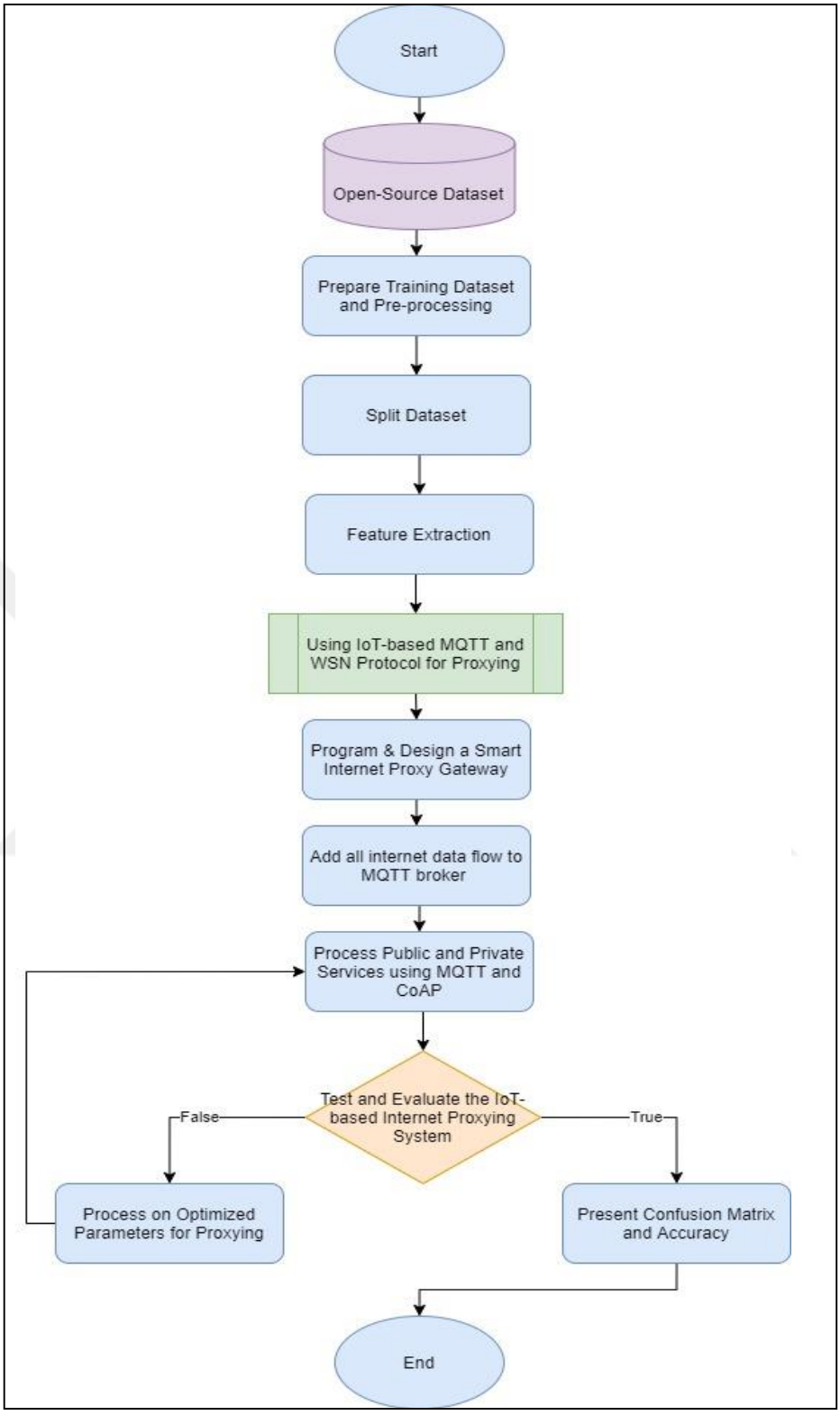


Figure 3.3: The Flow Diagram of Proxying Internet System in Internet-of-Things.

3.3 ENVIRONMENT OF WIRELESS SENSOR NETWORKS

By utilizing sensor-based interaction with the environment, we set WSNs apart from other wired, conventional, or even wireless networks. WSNs often have minimal infrastructure, if any at all. They are made up of several sensor nodes, ranging in number from some tens to thousands. Together, such sensors measure and monitor a certain area to get data regarding the environment. Most of the time, we are interested in using low-cost sensors, which have few processing components. In this chapter, methods for localization processes for proxying internet systems within WSNs are categorized and evaluated. Depending on a variety of factors, they could be divided into distinct categories. In this work, the problem is viewed as a distributed systems problem. Also, we presumptively include the possibility that not the entire network might have a direct connection. Consequently, one of the most important categories in terms of hardware limitations.

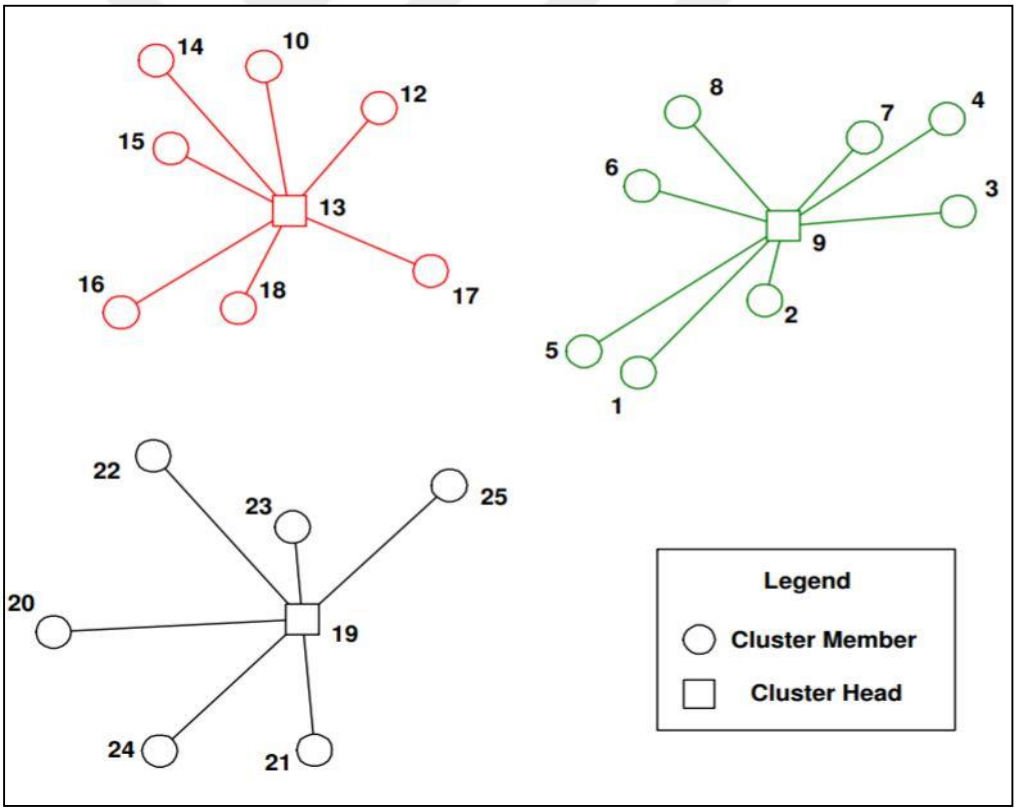


Figure 3.4: Topology of WSN Following Organization into Cluster.

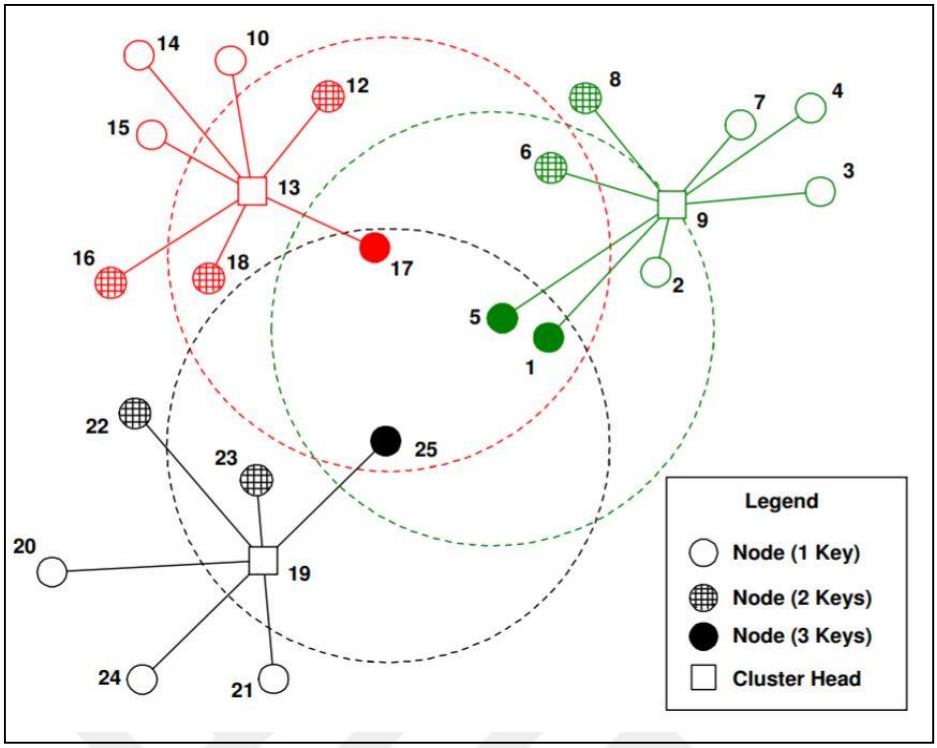


Figure 3.5: Topology of WSN Throughout the Phase of Key Setup.

3.4 DESCRIPTION OF DATA-SET

Internet techniques for the anomaly-based proxying suffer from a dearth of the reliable data-sets for the verification as well as testing. The majority of the data-sets are unreliable and outdated for evaluation, according to analyses of 11 datasets which have been available since the year 2017 by the Canadian Institute of Cybersecurity. While a few of such datasets fall short in terms of diversity and volume of network traffic, others lack the most recent proxy patterns and/or packet payload data anonymization, which restricts study potential. A few lack the metadata information and feature set as well. Researchers' system for inserting proxies into present datasets was published in 2015, presenting an intriguing method to the development of realistic datasets. Even though their tool wasn't employed in this study, it offers an intriguing alternative for future research. The studies will make use of CIC-IDS-2017 dataset, which was selected as being the most recent and well-documented dataset [54]. The dataset comprises of PCAP files needed for creation and the CSV flow records produced by CICFlowMeter. Data hs been captured for 5 days and no, and the dataset has not been anonymized. The bundled PCAP dumpfiles have a combined size of 48.9 GB. On working days of Tuesday, Thursday, Wednesday, and Friday, proxies were run in the afternoon and

morning. DDoS and DoS attacks, Heartbleed, Web proxies, SSH and FTP brute force attacks, infiltration, port scans (Friday), botnet traffic, and DoS attacks are among the proxies that have been put into place. Only valid traffic is present on Monday because it is a typical day. The associated research article and the CIC website both contain comprehensive information on the dataset and the testbed design.

3.5 RANGE-BASED LOCALIZATION IN MQTT

These methods rely on point-to-point or angle-to-point data. For determining the angle and separation between two nodes, the following techniques are the most popular:

3.6 RECEIVED SIGNAL STRENGTH INDICATORS (RSSI)

This method is based upon the comparison of a signal's power or strength at receiver node to that at the transmitter node. In comparison to the other approaches that will be discussed, this method uses RF signals, is more prevalent, and is considerably less expensive (requires no additional equipment). In terms of calibration and configuration, it is also easy and reasonably priced. Its sensitivity to noise is one of its biggest flaws, though. The accuracy of distance estimation might be very poor, especially for long distance cases.

3.6.1 Angle of Arrival (AoA)

The method, as its name implies, has been utilized in order to determine the angle at which source's signal is emitted. Each one of the nodes in this method typically requires at least two antennas. Its applicability in networks with small, inexpensive sensors is constrained by this necessity.

3.6.2 Time of Arrival (ToA)

IOT can be defined as one of the traditional techniques for localizing the indoor environment. The speed related to a radio signal is the same as the light speed (when the audio signals have been utilized for the localization, sound speed has been considered rather than speed of the light). Distance can be calculated between receiver and sender if we could calculate the signal's travel time and divide it by the speed of light.

The synchronization regarding the time between the nodes is one of the most crucial factors in this method [71]. Utilizing two different types of signals is one method for resolving synchronization problems in acoustic ToA estimation. The sender simultaneously emits an ultrasound signal and a radio signal. Receiver receives radio signal and initiates timer. The timer is stopped in a case where it receives ultra-sound signal. It could calculate differences between them as a result. High range precision is achieved using this technology, yet it requires additional hardware and uses a higher level of the energy.

3.6.3 Time Difference of Arrival (TDoA)

Although check time is essentially the same as the ToA approach, the locations are estimated using the relative times of arrival between nodes. We do away with the requirement to know precise time of signal emission at source by utilizing TDoA [72].

3.7 RANGE-FREE LOCALIZATION IN MQTT

In place of the more expensive Range-Based techniques, such schemes are used since they are more affordable [73]. Those methods lack information on distances and angles of the paths between nodes. In terms of the architecture of the algorithm, available techniques have been categorized into two blocks, which are:

3.7.1 Centralized Architecture

A node is chosen to serve as a monitor for many nodes. It ought to serve as a strong central station for the others. The chosen node serves as the neighbors' coordinator. Simplicity regarding implementation and removal of the computing process in each node are two benefits of such algorithms. However, we will lose a portion of the region if we lose the monitor node.

3.7.2 Decentralized Architectures

We require certain hubs in the conventional centralized algorithms, and getting the data can be challenging. Decentralized algorithms divide the region into numerous clusters, and in each of them, simultaneous transmission between all nodes is possible. As a result, we have greater reliability when put to comparison with the previous structure.

In large-scale ad-hoc networks, adding a GPS device to every node could increase the size and cost of the sensors [74]. It also has an impact on power consumption, which lowers network performance and lifespan. Therefore, adding a space-based satellite navigation system which could give each node its location might not be a good idea. Additionally, this factor categorizes the WSN techniques into two primary groups.

3.7.3 Anchor Based

Just a small position of nodes could be accurately located in several ways. GPS can often be used to accomplish this. We might also be aware of a node's exact location in the area. Those nodes are known as beacon or anchor nodes.

3.7.4 Anchor Free

We could not be able to have sensors with known precise geographic locations due to cost, usage, and environmental restrictions. Put differently, the network is devoid of anchor nodes. In such circumstances, localization techniques that rely on distance data can just determine the relative positions regarding the network's sensors [75]. In real environment, we can lose certain nodes or receive inaccurate data from the nodes. As a result, each node has a random and undesirable variable which could have an impact on the outcome. Each one of the nodes is independent of the random variable used to model the error. Another category regarding network localization algorithms exists as well. Depending on the level of detail in the data that the sensors collected, the category is separated into two primary categories of techniques.

3.7.5 Fine-Grained Algorithms

Those techniques make use of precise data throughout communication, like the distance from a reference node measured by ToA or RSSI. These algorithms typically make use of a variety of technologies, including infrared, radio frequency, and ultrasound signals. The positioning of network nodes is precise when there are no measurement errors, as is the case with fine-grained algorithms.

3.7.6 Coarse-Grained Algorithms

In essence, the localization regarding unknown nodes is done using less precise information using such algorithms. Those algorithms consistently use methods like hop-count to determine the distances between network's sensors. The division of MQTT localization algorithms into several groups is shown in Figure 3.4. The noise is frequently reliant on the interfaces and devices, particularly in range-based ideas. Therefore, a large number of them only operate within isolated environments.

i. They attempt to calculate angle or distance between 2 nodes.

ii. To determine where the sensors are, they attempt to integrate the measured distances or angles.

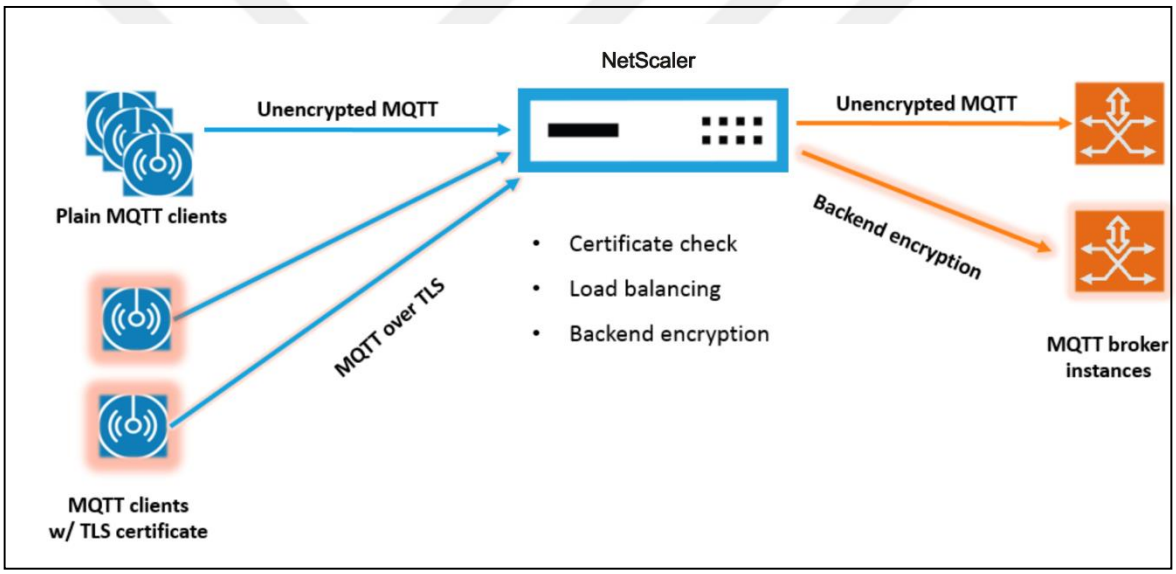


Figure 3.6: Classification of Approaches in The MQTT Broker.

The following six measures are used to assess how well localization algorithms perform:

i. Accuracy: The algorithms look for the measurement value that comes the closest to the actual value. Therefore, the most crucial element in the localization methods is precision or accuracy.

ii. Coverage: Every node has a number of neighbors. It might be transmitted to them. The nodes could communicate data over short distances. Every type of connecting technology has a specific operating range. Due to several factors like battery life,

antenna power, and propagation conditions, the effective coverage varies. Therefore, we require a sufficient node density and number of anchor nodes to achieve improved performance and coverage.

iii. Complexity: In contrast to networks that just depend on the connectivity information, range-based networks need that the ranging information be transferred between sensors, requiring devices to have more complex hardware (range-free). The network's earlier design is more complex.

iv. Scalability: The perfect algorithm can be made larger. It is a characteristic of every localization algorithm. It ought to be appropriate to use in more complicated scenarios. Put differently, the algorithm doesn't need to alter as the network gets bigger.

v. Robustness: All signals in the outside environment are noisy with random error, therefore we anticipate that this will not unduly impair the algorithm's performance [76]. Maybe we lost certain nodes and a few network networks. We constantly work with distances, angles, and time when using range-based algorithm. As a result, if we do not use a robust and dependable algorithm, we will eventually encounter several errors.

vi. Cost: The cost of each sensor is one of the most crucial factors in WSNs. We could either lose or save a lot of money by making a little adjustment to the nodes' software and hardware. Energy consumption is one of the elements that affects price. The costs may be reduced if the sensors used less energy. However, in the case where the proposed algorithm has the ability to quickly determine the sensor positions, a long-lasting battery is not required [77].

The program code has a feature that only activates when a smart intermediary change is introduced to the device. As a result, in this smart intermediary work, our gadget declares and establishes the fundamental characteristics connected to the components sticks, WiFi network attributes, intermediary server address, and MQTT agent address. It after that uses WiFi to transmit the enrollment messages to intermediary server for the purpose of registering itself including each of its elements and attributes. The gadget eventually establishes a connection with the MQTT agent and purchase’s themes related to its constituent parts. The capacity, which has been suggested by its name, is one that will run continuously to the point

where the board has been reset or switched off, which allows the program to adapt and respond to any focus. MQTT convention circle has been implemented in such capacity, which verifies the existence of new messages of the MQTT from desired IoT-based MQTT representative for execution regarding the intermediary computerization framework for each primary circle cycle.

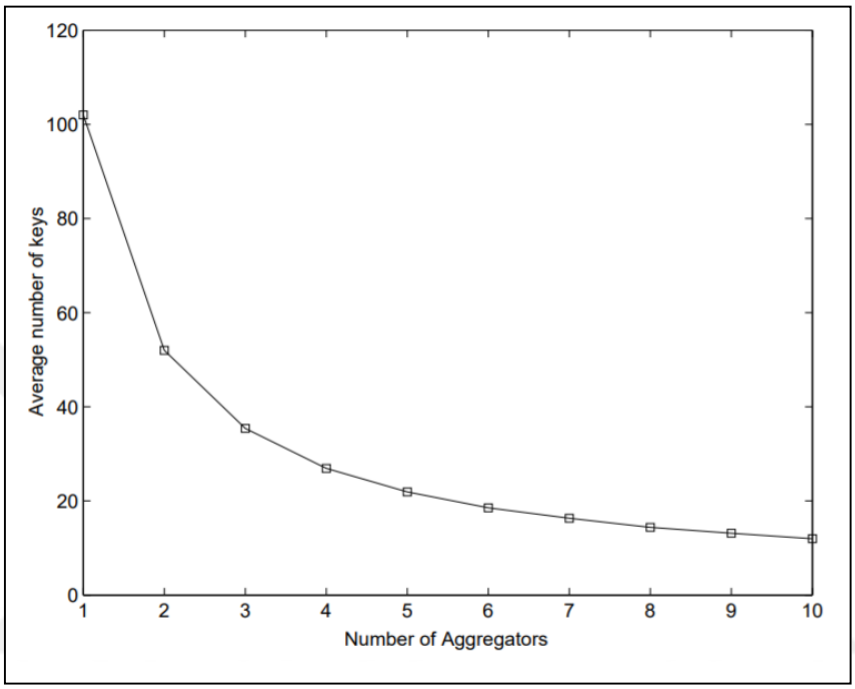


Figure 3.7: The 100 Keys That Have Been Utilized as Node Aggregator in the MQTT.

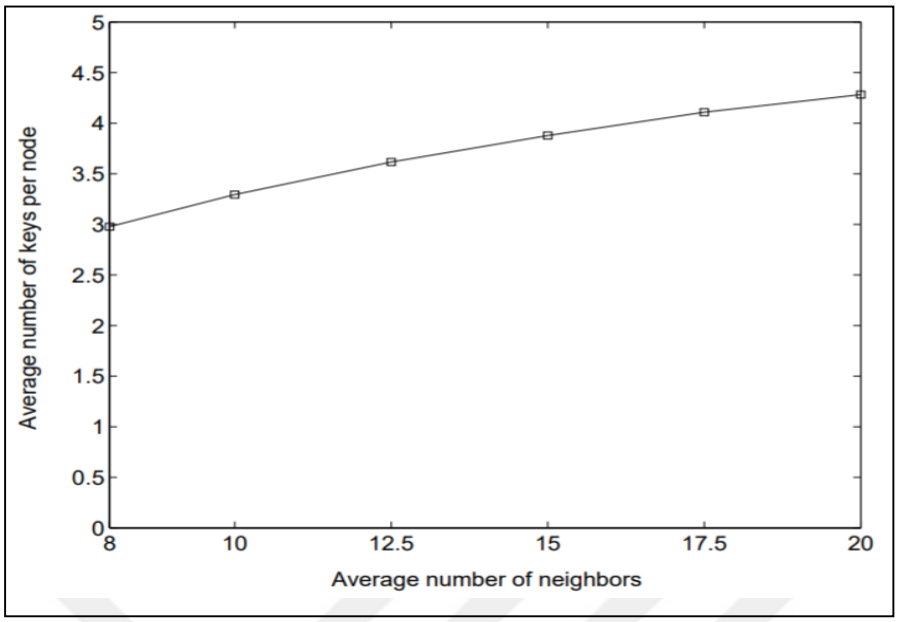


Figure 3.8: Keys That Are Held by the Sensor Nodes for 20 Neighbors in MQTT.

3.8 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. P Python has been well known due to its exceptionally fast compilation time and ability to produce statically linked binaries that are not dependent on execution environment's libraries. The current version of Python, 3.8, which has a model of concurrent executions and coordination of asynchronous processes, has been influenced by CSPs (i.e., Communicating Sequential Processes) study. The OS thread is not to be confused with a go procedure, which is an asynchronous process. Python multiplexes MQTT onto OS threads as required. The relevant OS thread also blocks when a WSN block happens, yet no other blocks are affected. MQTTs dynamically allocate resources as need and are less computationally expensive compared to threads. 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.

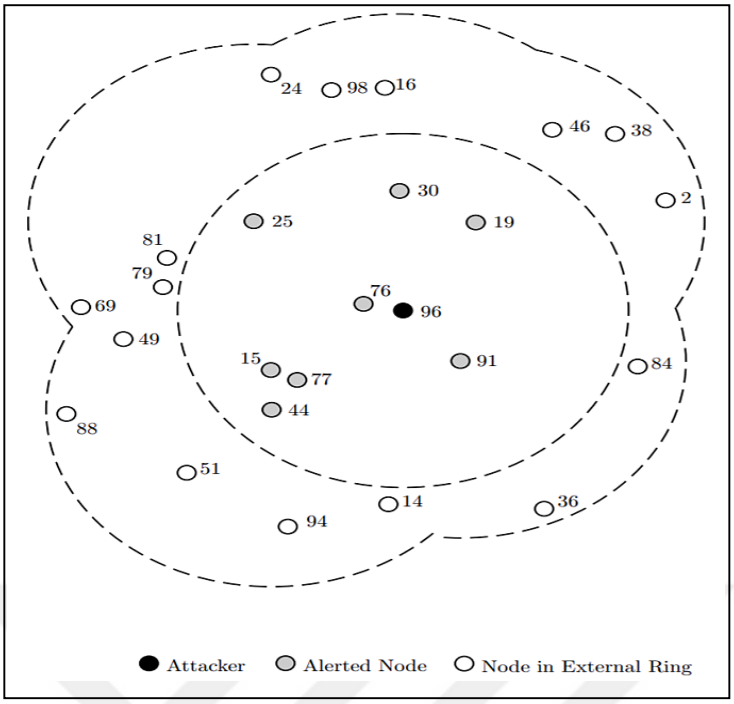


Figure 3.9: Ringing Concept in the WSN for Proxy's Identification.

Choose statements, which are utilized for receiving data from channels, offer multi-way concurrent control. Python's extensive support of architecture and platform that will be covered in more detail later, is another crucial feature. In addition, the language provides stack management and a garbage collected runtime to prevent memory corruptions [78]. This must not be mistaken with Python runtime's VM method, though. Python offers interfaces and the flexibility to define techniques on structures despite being an object-oriented programming language. Python mandates a uniform programming style as well for presenting source code, and it significantly aids in improving readability when comparing codes from various authors. It should be noted that Python offers the ability to read and write to arbitrary memory addresses through the unsafe module, hence evading runtime checks and type system. The relevant OS thread also blocks when a WSN block happens, yet no other blocks are affected. MQTTs dynamically allocate resources as need and are less computationally expensive compared to threads. 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.

Table 3.1: Sub Structures of the MQTT Protocol

Name	Description
Dot11 QOS	IEEE802.11 Quality of Service
Dot11 HTControlVHT	information of IEEE802.11 HTC
Dot11 HTControl	information of IEEE802.11 HTC
Dot11 HTControlMFB	information of IEEE802.11 HTC
Dot11 HTControlHT	information of IEEE802.11 HTC
Dot11 ASEL	information of IEEE802.11 HTC
Dot11 LinkAdapationControl	information of IEEE802.11 HTC
LinkLayerDiscoveryValue	ICMPv6 option
LLDP ChassisID	
IPv4 Option	
LDP PortID	IGMPv3 group records for a report of membership
LLDP Capabilities	Information of Link Layer Discovery Protocol
ICMPv6 Option	TCP option
LLDP SysCapabilities	DNS start of authority record
LLDP OrgSpecificTLV	resource record of the Domain Name System
IGMPv3 GroupRecord	Mail exchange record
LLDP MgmtAddress	DNS service record
TCP Option	DHCP v4 option
DNSSOA	DNS for domain of signal
DNS ResourceRecord	DHCP v6 option
DNSMX	Hop By Hop Option Alignment
DNSSRV	IPv6 hop by hop extension TLV option
DHCP Option	
DNS Question	
DHCPv6 Option	
IPv6 HopByHopOptionAlignment	
IPv6 HopByHopOption	

Table 3.2: Encoder Fields of the MQTT Layer.

Layer	Num-Fields	Field
TCP	2	Time-stamp, DstPort, FIN, Data Offset, SrcPort, SeqNum, AckNum, RST, SYN PSH, URG, ACK, PayloadEntropy, ECE, NS, CWR, Window, Checksum, Urgent, Padding, Options.
IPv4	7	TTL, Checksum, DstIP, SrcIP, Padding, PayloadEntropy, Options, Payload Size
UDP		Time-stamp, DstPort, SrcPort, Length, Checksum, PayloadSize, PayloadEntropy.
DHCPv4	6	Time-stamp, HardwareType, Operation, RelayAgentIP, HardwareOpts, HardwareLen, Xid, Flags, Secs, ClientIP, NextServerIP, YourClientIP, ServerName, ClientH-WAddr, File, Option
IPv6	2	Time-stamp, TrafficClass, Version, FlowLabel, NextHeader, Length, SrcIP, HopLimit, PayloadEntropy, DstIP, Hop-ByHop, PayloadSize.
ICMPv6		Timestamp, Checksum, TypeCode,
DHCPv6		Time-stamp, PeerAddr, HopCount, MsgType, LinkAddr, Options, Transaction ID
ICMPv4		Time-stamp, Checksum, TypeCode, Id, Seq
ICMPv6NeighborSolicitation		Time-stamp, TargetAddress, Options
ICMPv6 Echo		Time-stamp, SeqNumber, Identifier
DNS	8	Time-stamp, OpCode, ID, QR, NSCount,
ICMPv6RouterSolicitation		Time-stamp, Options
ARP	0	Timestamp, Protocol, AddrType, HwAddressSize, Operation, ProtAddressSize, SrcProtAddress, DstHwAddress, SrcHwAddress, DstProtAddress

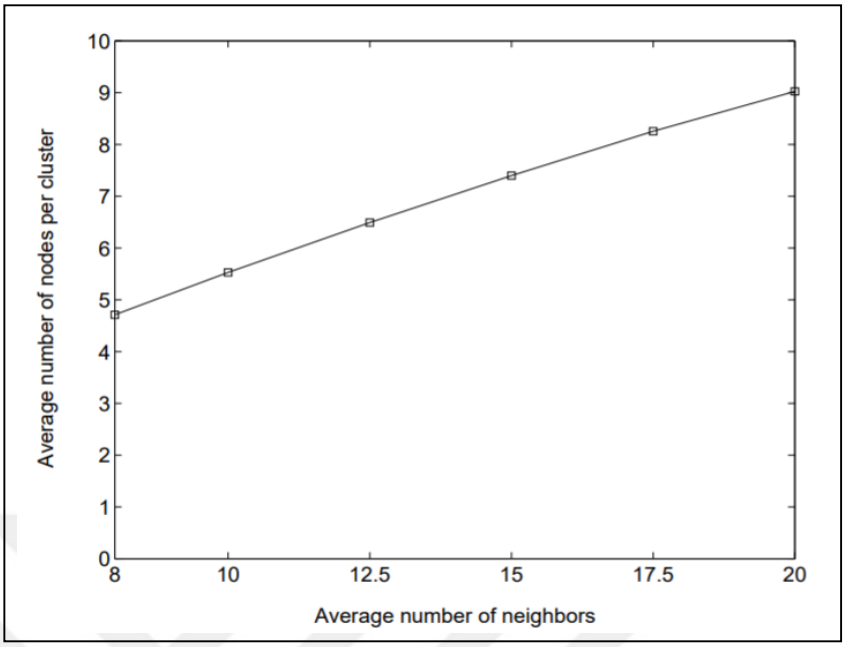


Figure 3.10: WSN That Uses Clusters with An Average Number of Nodes.

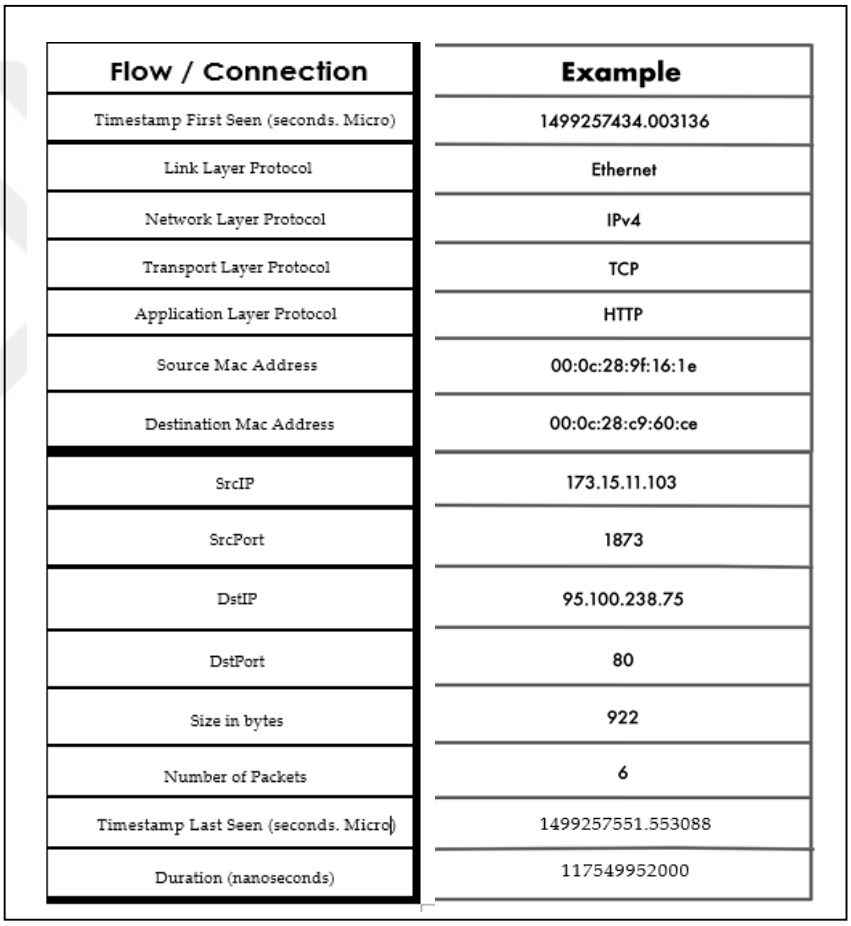
3.9 PROCESS OF MQTT

While the Connections and the Layer Flows are bidirectional, MQTT Flow Audit Records are unidirectional. To reduce the usage of memory used, Connections and Flows are flushed regularly. Timeout intervals could be set for each category independently. The destination and source of a Flow or Connection are determined by the first packet seen.

Table 3.3: Networks with Varying Data Transport Rate

Name	**Layer**	**Description**
Flow	All	Uni-directional Multi-Layer Flow
LinkFlow	Link	Bi-directional Link Layer Flow
Connection	All	Bi-directional Multi-Layer Flow
TransportFlow	Transport	Bi-directional Transport Layer Flow
NetworkFlow	Network	Bi-directional Network 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, making the discussion with web server completely scrambled. As it has been examined in sub-section User Permissions Management there are two kinds of client ideas, which are: Individual Users and User Roles. Along these lines, computerized authentications might be created for client job or in any event, for explicit clients. The intermediary server will utilize that data in validation stage in order to distinguish the client profile that it will utilize.



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

Figure 3.11: 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’s compulsory having an MQTT merchant (in other words,

MQTT server) which permits that correspondence. The correspondence transport that has been utilized is a WiFi organization and MQTT dealer likewise gives security systems, for example, utilizing endorsements to validate gadgets. The public administrations are RESTful web-benefits, which burns-through/produces JSON demands/reactions. With that methodology we ensure adaptability and opportunity to execute various applications, in any stage, without any similarity concerns. It very well may be a portable application, smartwatch, an internet browser or significantly another IoT framework. The main necessities are that the 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 SOA (i.e. Service-Oriented Architecture) where they're "secret elements" for clients and administrations are free from each other.

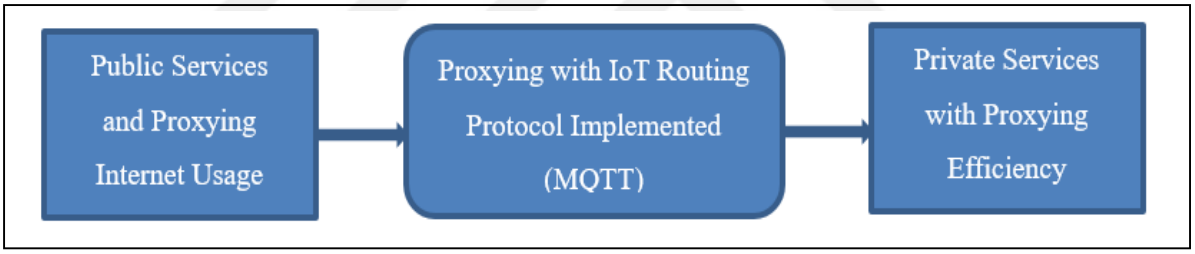


Figure 3.12: Processing at The Most Fundamental Level for An Iot Proxying System Using MQTT Broker Routing Protocol for Both Private and Public Services.

Using RESTful web-administrations has the additional outstanding benefit of their framework flexibility. REST is stateless at the server end, meaning that servers don't need to maintain state or information over the requests. Therefore, communications between the servers are minimal, giving it a wide range of applications. Furthermore, because RESTful web-administrations are demonstrative, it is possible to present good server-side load balancer which could undoubtedly route requests to the appropriate servers in accordance with URL and HTTP techniques. For example, GET requests may go to a set of the servers, whereas the POST requests may go to another server. Based on acknowledged practices of the REST-

ful web-services engineering plan, we had separated many types of the tasks that are available using the HTTP approaches. The following tasks have been utilized: GET to obtain the information in a read-just manner; DELETE to delete assets, which might be including properties, parts, or even tasks; POST for the creation of new assets; and PUT to update assets. Accompanying subsections will provide a description of the motivation for such assistances.

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 the proposed framework we make Internal Devices Management System which cooperates with 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.

It is necessary for devices to interface with the intermediary server (in the individual administrations) to enlist their data during their introduction since the internal devices the executive's framework should always be synchronized with the real framework data. Here is a brief explanation of how this interior gadget of the executive's framework functions before moving on to the more thorough model that is shown in the Examples of Usage segment: We need to incorporate a crystal fixture (a device) into our savvy intermediary framework. Using the private assistance provided by the /programming interface/de-bad habit/gadgets during installation, the gadget must register its properties and parts as well as enroll itself in the intermediary server. As a result, crystal fixture will also send a request to programming interface, gadget, or components, which permits turning characteristics OFF or ON. Internal Devices Management framework will constantly understand organization express, associ-

ated devices, and their properties using this tool. Without having to reset the entire organization or update the gadgets' code, it permits the impromptu addition of new gadgets to the organization. This framework provides Complex Actions Management System which permits creating execution streams in accordance with explicit conditions, such as the states of gadget parts, to achieve the positive knowledge. 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 that have been characterized and, assuming that they do, the framework will execute comparing activities.

3.9.1 Link Flow in MQTT

Table 3.4: MQTT Link Layer Flow for Internet Proxying.

Link Flows
NumPackets
DstMAC
Protocol
SrcMAC
TimestampLast (seconds.micro)
TimestampFirst (seconds.micro)
Size (bytes)
Duration (nanoseconds)

Flows are additionally exported for each layer, such as NetworkFlow, LinkFlow, and Transport Flow, for further decoupling them. The Layer Flows are bi-directional, and their UID is same in the two directions, as was already indicated. Take note that UID field was left out of the next graphics. A LinkFlow is a term used for describing Internet Protocol Suite layer 1 communication. It includes connection statistics as well as the hardware addresses of the connecting devices.

3.9.2 Network Flow in MQTT

A Net workflow of Internet Protocol Suite depicts communication at layer 2. It also includes IP addresses and connection statistics of the connecting devices.

Table 3.5: Using MQTT Network Layer Flow as An Internet Proxy.

Network Flows	Example
Timestamp First (seconds. Micro)	1499255388.191380
Timestamp Last (seconds. Micro)	1499284556.475471
Protocol	Ethernet
SrcIP	00:0c:28:9f:16:1e
DstIP	00:0c:28:c9:60:ce
NumPackets	52
Size (bytes)	1423
Duration (nanoseconds)	29168284091000

3.9.3 Transport Flow in MQTT

A TransportFlow of Internet Protocol Suite describes layer 3 communication. It includes statistics on connections as well as the ports for communicating devices.

Table 3.6: MQTT Transport Layer Flow for Proxying Internet.

Transport Flows	Example
Timestamp First (seconds. Micro)	1499279129.560185
Timestamp Last (seconds. Micro)	1499282267.760859
Protocol	TCP
SrcPort	60846
DstPort	443
NumPackets	156
Size (bytes)	118942
Duration (nanoseconds)	3138200674000

3.10 BUILDING PROXYING INTERNET

Shared resources must be synchronized in concurrent programming for ensuring their status when being read or modified. Race conditions develop if access is not synced, which will result in flawed program behavior. The go toolchain supports the compiling of program by race detector enabled to prevent that and identify the race conditions early in development cycle. In a case when a data race take place, which will allow application to crash with the stack traces in order to aid developer in the process of the debugging. Active race detection slows down programs by a factor of 10 to 100. The -race flag should be introduced to the command of the compilation for compiling Python program with race detection being enabled.

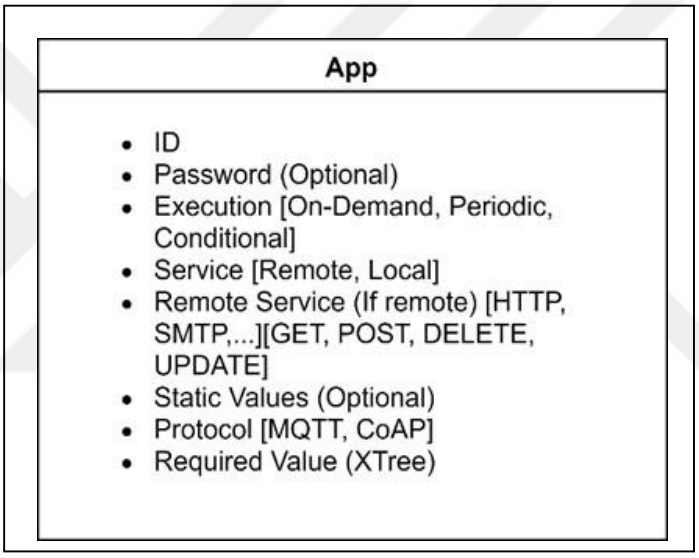


Figure 3.13: Configurations of An App in The Proposed System.

Processing time is significantly influenced by the setup, traffic type, and performance of the system. One might find the processing timings in the experiments with the use of default setup with 4096byte buffers, all 40+ encoders, compression, 1,000 workers, and packet buffer size of 100 on development computer (current MacBook Pro with 32GB 2400MHz DDR-4, running Windows 10, 2,9GHz Intel Core i-9). It's possible that the project has bugs that are still undiscovered. The program panics frequently when an error might have been handled in a more graceful manner so that a stack trace might be used to help with debugging. Python should only be used for non-research tasks till more robust error handling and unit tests are implemented. Lines of code (LoC) statistics for Python v. 3.8 are shown in the table below:

Table 3.7: Metrics Contained in The Files for The Python Code Used to Proxy the Internet.

Packages	Languages	File	Blank	Comments	Code
collector	Python	13	273	371	882
label	Python	15	365	413	1,266
types	Python	38	232	465	1,721
cmd	Python	3	57	64	282
encoder	Python	48	559	957	3,299
netcap.proto	XML	1	93	100	660
utils	Python	2	39	44	156
total	Python	126	1,693	2,520	8,296

For facilitating future development through other researchers and for making implementation details understandable, Python source code includes a significant amount of descriptive comments (80 to 100 lines of the comments on many code lines as of v. 3.8).

Table 3.8: Metrics for The Input Packet Processing for Internet Proxies.

Files	Num Packets	Sizes	Time of Processing	Extracted Data
Tuesday-Working-Hours.pcap	11551954	11GB	11m11.918614271 s	440MB
Thursday-Working-Hours.pcap	9322025	8.3GB	8m39.274062535 s	369MB
Wednesday-Working-Hours.pcap	13788878	13GB	12m58.134988695 s	536MB
Friday-WorkingHours.pcap	9997874	8.8GB	9m0.653867719 s	410MB

3.11 ELEMENTS OF PROXYING INTERNET

The proxying internet components may be summarized as in: Users: are those who will engage with or inhabit installation.

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

ii. Sensing Devices: which have been installed IoT devices which serve as data generators. Their data is transferable, consumable, and collectible for future uses.

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

iv. Gateway Devices: They are IoT devices whose only purpose is to transmit data between the Internet and an installation. In setups where other IoT devices aren't Internet-enabled and can't interact in a direct way with the rest of system, they play a vital role.

v. Units of Measurements: It is related to standard that has been utilized to quantify observable occurrence.

vi. 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.

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

viii. 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 connection is crucial for security and management needs, and it may be used for managing the complete installation.

ix. Sensing Attributes: term "sensing attributes" refers to actuator as well as sensing devices and is utilized in order to describe the kinds of data that every device creates or the kinds of devices that it controls. Information on physical phenomenon, measurement unit, or even whether data.

x. Access: The ability of a user to study the data presented via a collection of physical locations or sensing devices or to operate an actuator device is specified by this element. Based on the type of user, these permissions could be restricted. For example, students and instructors might just need access to information regarding their classroom, yet might just look at restricted information related to other school facilities. Similar to this, as elements such as HVAC system are just managed through building

administrators or designated teachers (principal), students could not be permitted to operate the devices of the actuator in their classrooms.

The QoS internal framework, which is responsible for each message's delivery confirmation, is another MQTT feature that we utilize. In this framework, gadgets are allowed to purchase/distribute messages at any of the three QoS levels which are available. However, the recommendation of the present study is that basic messages must be having the highest possible QoS level, which is 2. These grantees that messages are delivered and that they are also non-repeatable, meaning that messages aren't sent twice or more. An alarm might serve as an example of a message with the QoS of level 2. In such case, we need the framework to keep attempting to communicate a message to the point where it finally achieves desired result. QoS levels for each theme have been defined during gadget enrolment stage and can be adjusted thereafter. They're distinguished by a certain characteristic that presents extraordinary adaptability to manage QoS levels as determined by the most grained level of the gadgets. In the case where the level of the QoS hasn't been specified, a default value of 0 is used (the most minimal and quickest one).

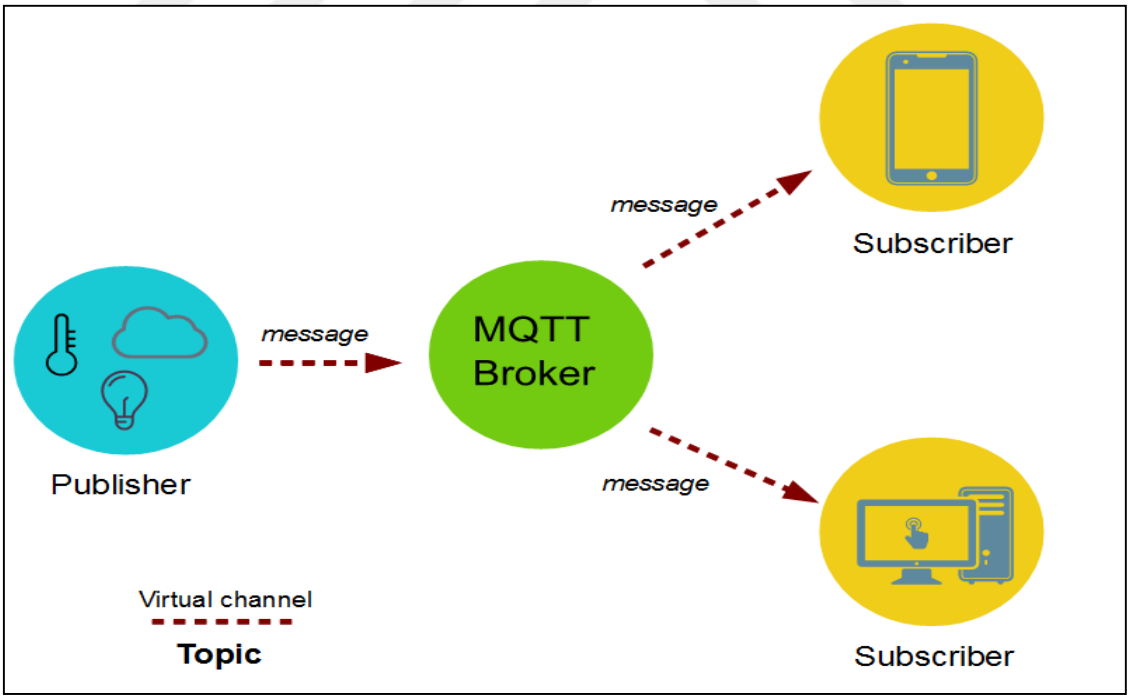


Figure 3.14: General Sequencing of The MQTT Broker For Management of Actions.

The communication between gadgets is also made possible by the use of a framework with extra module (i.e. server) for MQTT expert, which permits a need for the intermediary server's outer "work." This is useful to reduce the workload on intermediary server and, assuming the clients need to associate gadgets with better computational capabilities, that might handle a few "complex" tasks on their own, they might have gadgets in charge of tasks directly from other gadgets, without any intermediary server that acts as a delegate in such interaction. The disservice of such methodology represents the extra intricacy that it adds to deal with those intricate activities, for certain activities characterized at the intermediary server level and others straightforwardly in the gadgets.

Additionally, we make use of MQTT stored messages in our execution. Those messages are described in MQTT convention, and we utilize them to inform other gadgets when a gadget is turned off and to keep the most recent message. The last option feature enables a recently connected device to quickly determine the most recent state of certain characteristics or a component of another device, without having to wait for another message to receive a newly updated state. The framework comes with just one client job by default: The highest level of authorizations has been granted by an IoT-based MQTT merchant: it enables changing the values of all components, gadgets, and properties; creating new burden balancers in jobs and assigning them to specific pieces of skilled intermediaries; creating new rooms.

4. RESULTS

The suggested system is constructed with the use of Python and an Amazon LightSail web server to demonstrate advantages of utilizing it for supporting the IoT devices. The Flask web system is utilized to develop the web application which is utilized for creating as well as configuring Apps, while Paho-MQTT python library is utilized for implementing MQTT broker and aicoap for the CoAP protocol. After that, an ESP-32 device has been utilized for the purpose of accessing the same services directly or by utilizing the suggested system, employing both the current techniques and libraries. The total device of memory, energy, time, and data supplied and received via ESP32 is used to gauge its performance. One of ESP32's pins must be set to high right prior to the appropriate task is executed and low as soon as it is finished in order for measurements to be accurate. Depending on the condition regarding such pin, energy and time are measured. In first experiment, the suggested system has been utilized in order to transfer the data between the IoT devices and is contrasted with utilizing the well-known ThingSpeak service to transfer the same amount of data. The App set for such task is configured for updating a point at ThingSpeak server with the use of HTTP request as well as updating all of the IoT devices which have been subscribed to such value right when it changes because ThingSpeak service permits logging data as well. As a result, the App has been set to conditionally update via MQTT protocol if the value changes. The comparison depends on ending and receiving 1, 5, and 10B regarding payload data from an IoT device to another, while directly or through suggested approach storing the value in ThingSpeak server in the two instances.

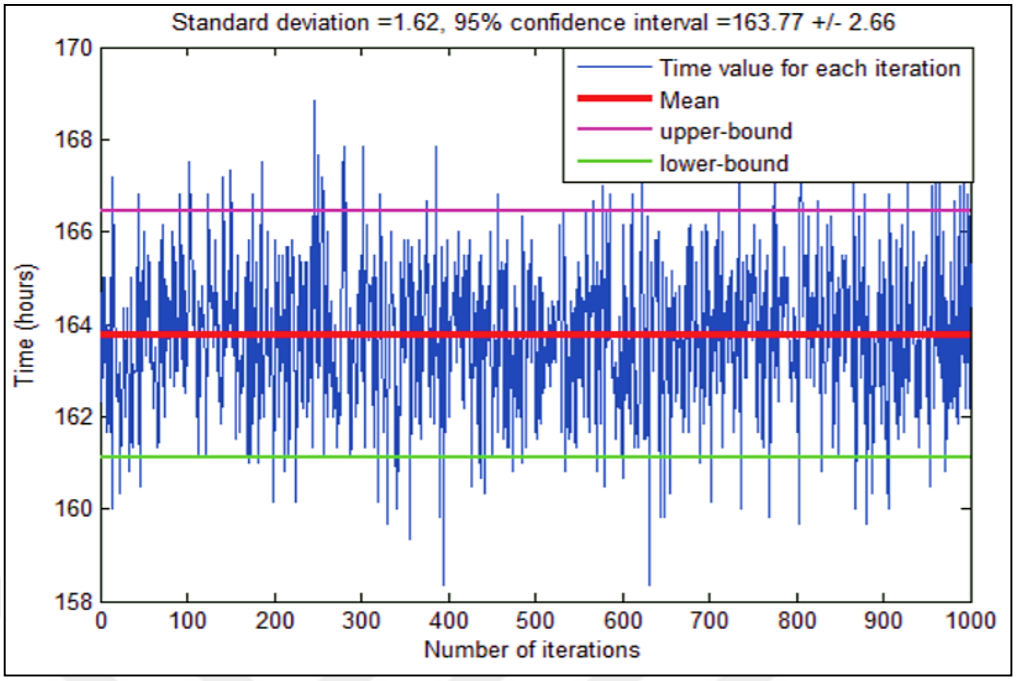


Figure 4.1: Conveyance Plot of Recreation Time in Simulation for Proxying Internet.

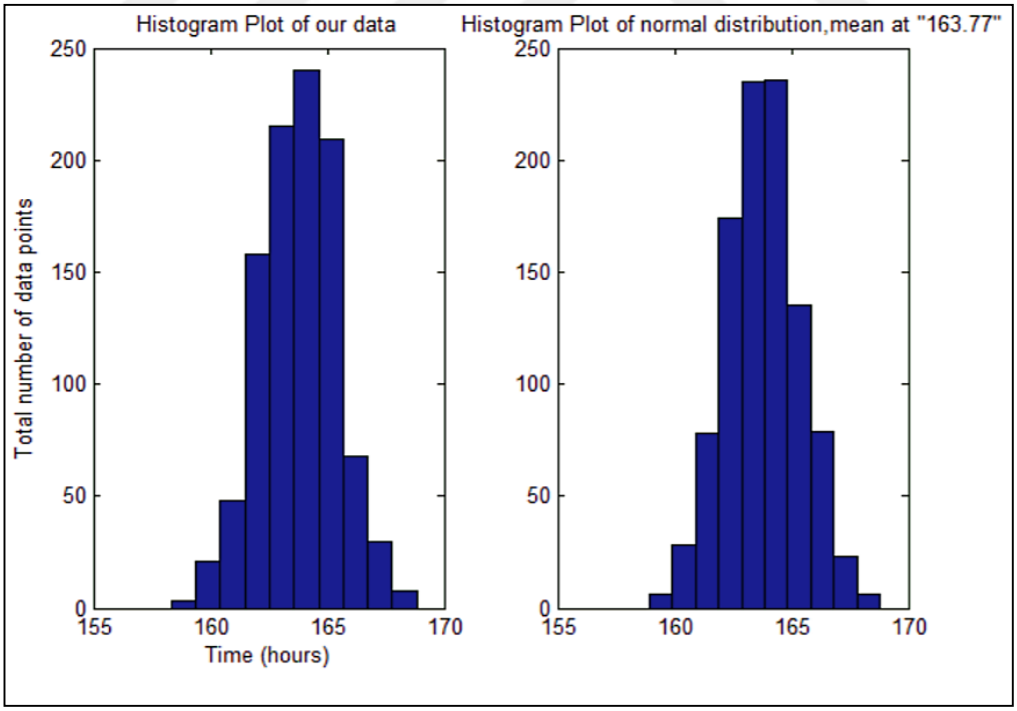


Figure 4.2: Histogram Plot of Reproduction Time That Has Been Contrasted with An Ordinary Conveyance for The Data Points In WSN.

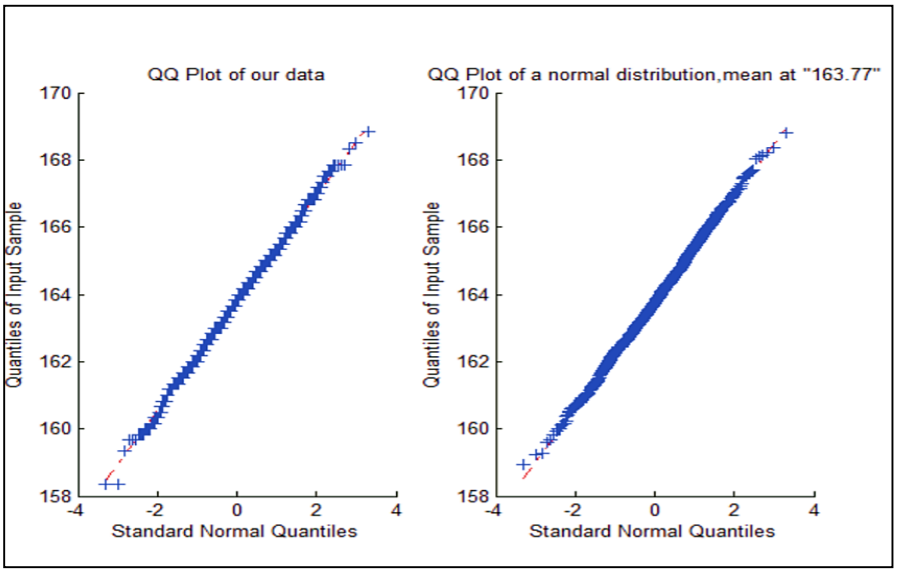


Figure 4.3: Standard Normal Quantiles That Have Been Represented in Simulation with The Time Unlike Normal Distribution.

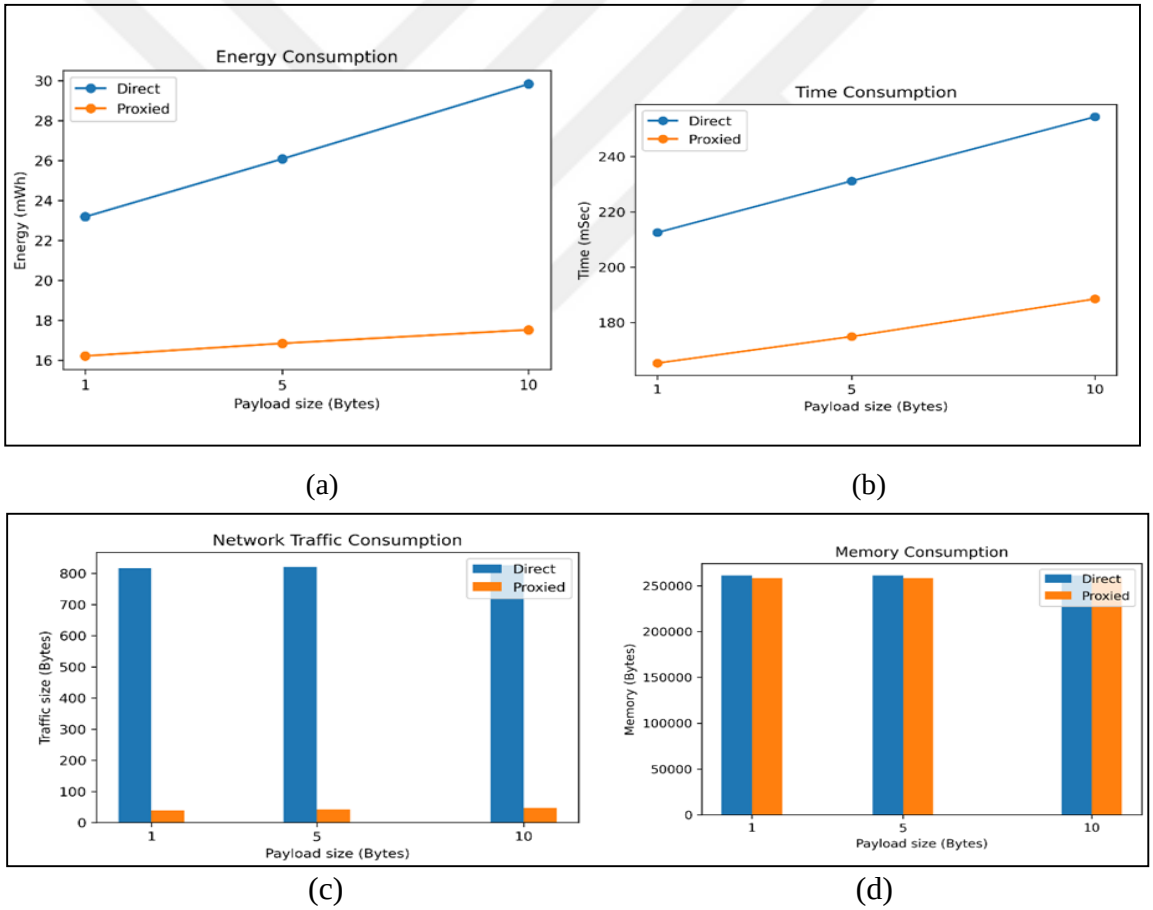
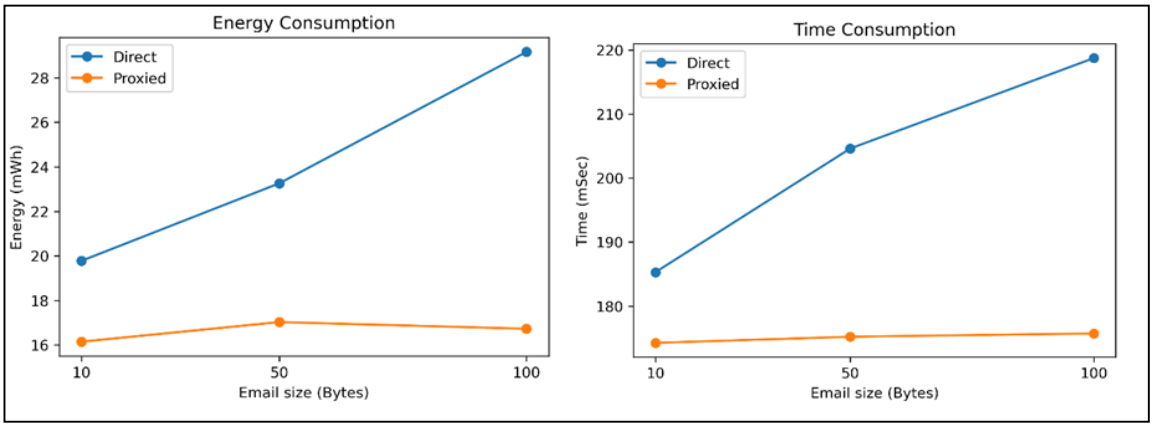
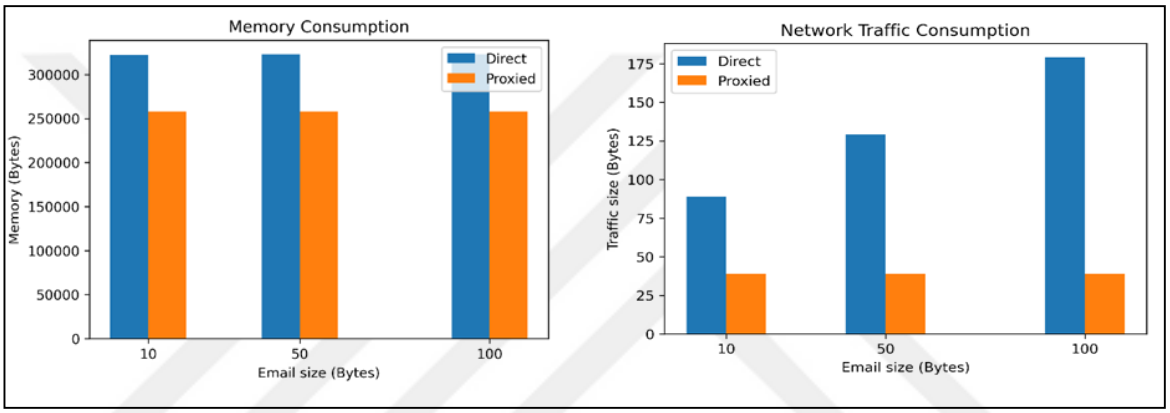


Figure 4.4: IoT Devices' Resources Consumption (Communicate Data Directly by The Suggested System): (a) Energy; (b) Time; (c) Memory; (d) Network traffic consumption.

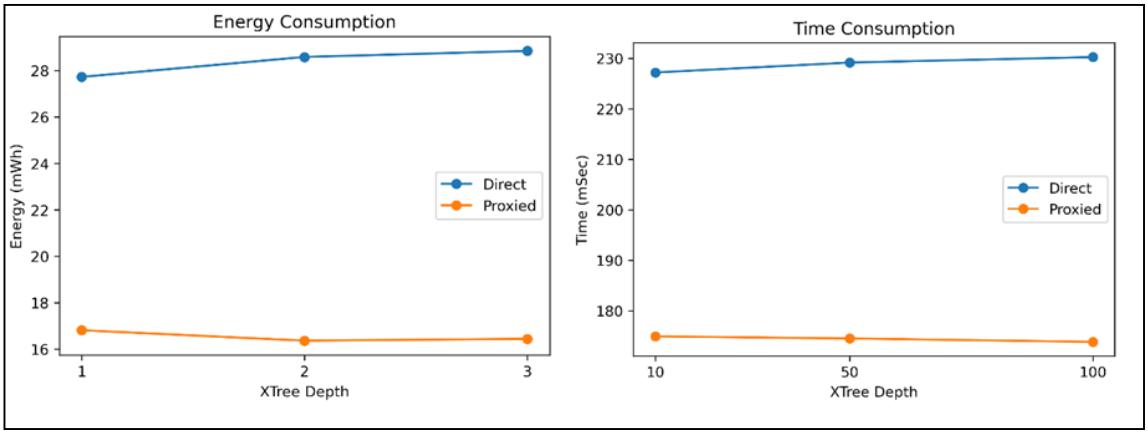


(a) (b)

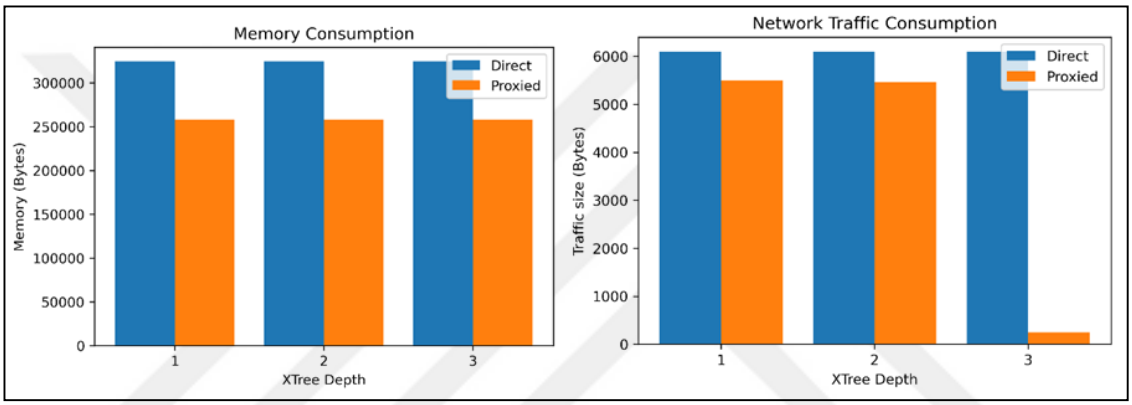


(c) (d)

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



(a) (b)



(c) (d)

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.

Over 1 in 1,000 normal data points might be incorrectly classified as proxy, even with theoretically good false positive rate that equals to 0.00150. In the scenario if a data point was to be produced every 100 milliseconds, which is not practical, a false alert can be given every 100 seconds. A single data point, on the other hand, scarcely qualifies as a proxy in situations in which packets are often sent in a continuous stream. This MQTT method might be used, in which an alarm only sounds if the bulk of the proxy data points are present inside a certain size window.

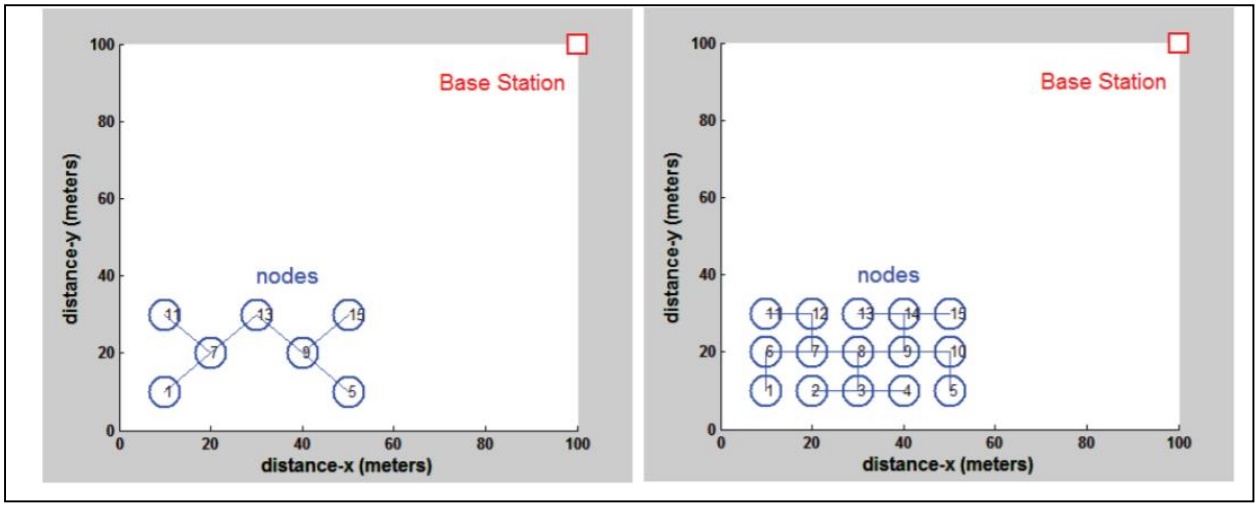


Figure 4.7: Various Organization Geographies of The Nodes in The WSN With 7 & 15 Nodes In BS.

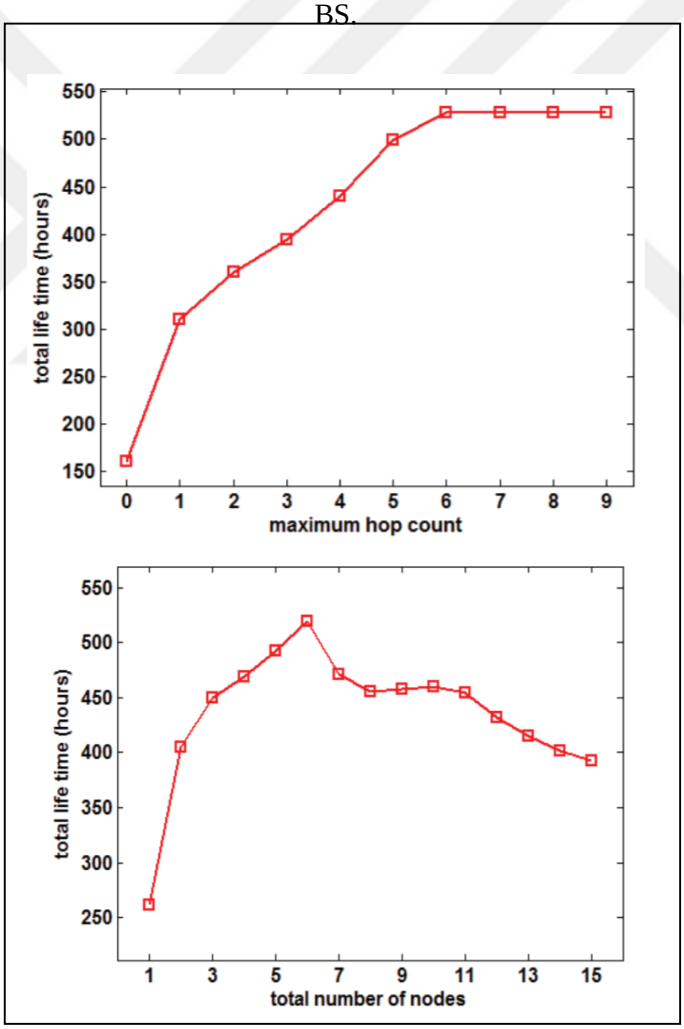


Figure 4.8: Maximal Hops and Total Number of The Nodes in The Network Vs. The Total Life-time of Network.

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.

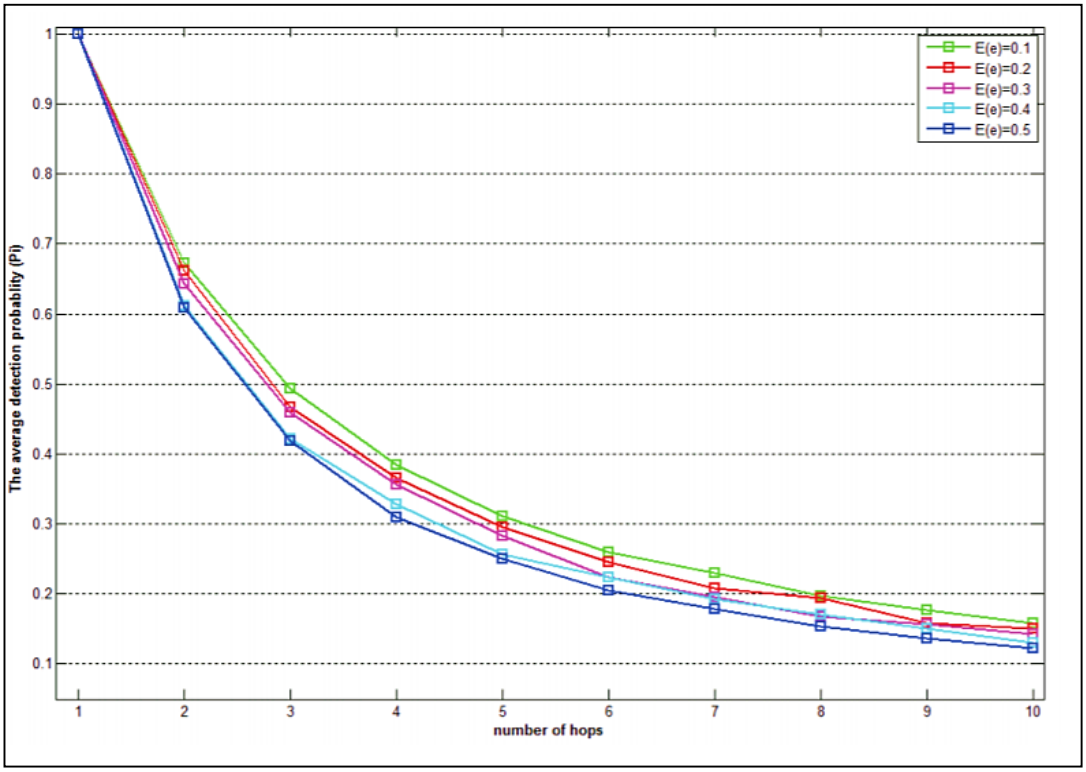


Figure 4.9: Effect of The Size of The Cluster on Location Likelihood of Proxying Internet System for A Variety of The Rates of The Packet Misfortune While the Rest of The Rate Has Been 60% In MQTT.

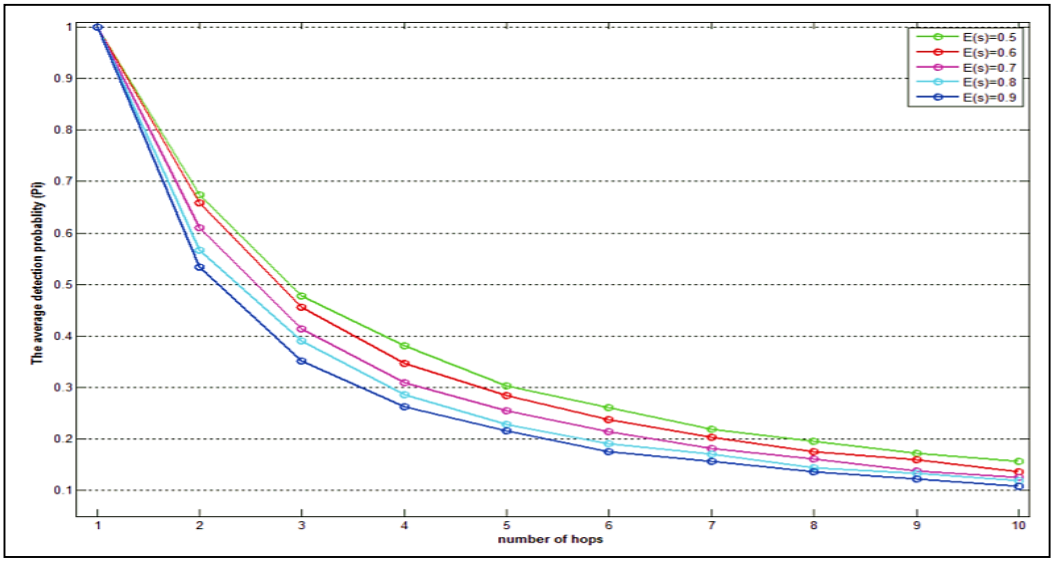


Figure 4.10: Effect of The Size of The Cluster on Location Likelihood of Proxying Internet System for A Variety of The Rest Rates While Packet Misfortune Rate Has Been 30% In MQTT.

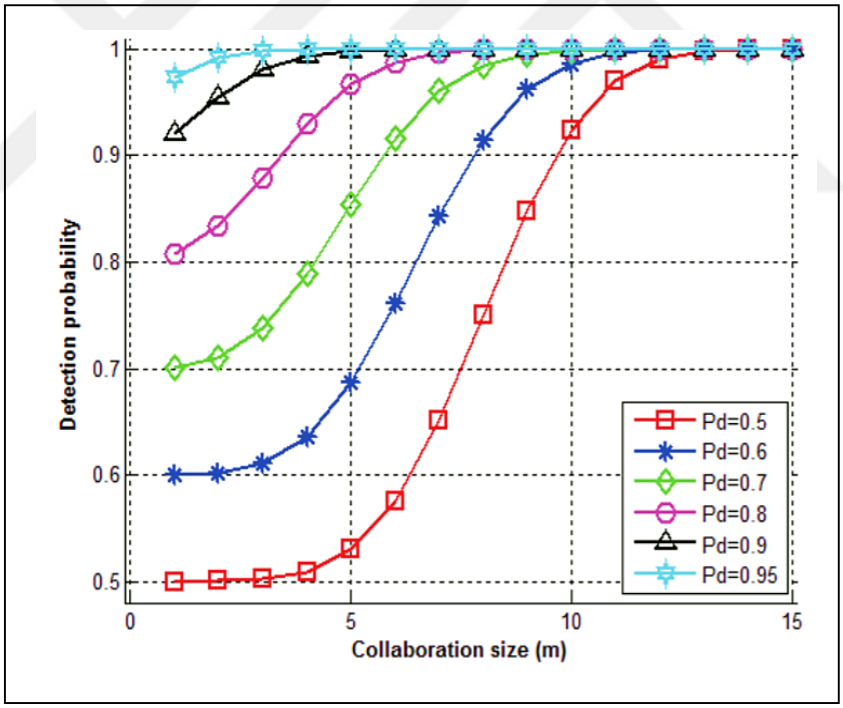


Figure 4.11: Compiling MQTT With Race Detector That Has Been Enabled for The Probability of Detection (PD) Vs. Size of Collaboration (M) For the Proxying Internet.

This framework enables the use of any customer in any stage, whether it be a mobile app, a website page, a smart app, or anything else. The primary requirement is that the program have the capability to use HTTP convention, on the grounds that customers must have the

capability to use RESTful API calls. Similar to gadgets, the required code will be simpler if the customer has a JSON library, yet this is not required. The customers in the framework can be of two different types: either they are designed to provide client collaboration, in which case they should be planned in accordance with the best convenience practices, or they are other IoT frameworks, in which case they do not provide any UI (it represents a typical M2M correspondence situation). Only the order line was used for testing the customer's REST API calls during the execution of the framework (UI is out of scope), demonstrating the customer's minimal requirements.

5. DISCUSSION

All WSN intrusions for the experiments used in this paper have been found inside nodes and clusters. Nodes have been encoded as dummy variables. Each unique entry is given its own column, and the original column is afterwards removed from dataset. Booleans are converted to numeric values by encoding 0 for false and 1 for true. For experiment 1, following hanging for no less than 7 hrs at dummy variable encoding Answers columns from Tuesday-WorkingHours-1/DNS labeled.csv, the first run is terminated after 12 hours. For the reduction of the amount of the data which needs being processed, the size of the sample is decreased from 1 to 0.50. As a result, half of the data already available will be used. The second run, however, was also unable to complete in the allotted time and was stopped at the same spot after 8 hrs as the first. The experiment took more than 29 hours to complete after starting the 3rd run and only gathering 20% of available data. The initial experiment, which used early version of the clusters, has been never repeated. It is included here for the completeness' sake and since findings from this experiment informed changes made to later ones. The entries with a "-" included just one label type following sampling. Due to the fact that no prediction has been feasible, the Keras library was shut down.

Table 5.1: Classifications of Various Proxies with The Respective Count.

Classifications	Count
Generic Protocol Command Decode	6449
Not Suspicious Traffic	80
Attempted Information Leak	3
Possibly Bad Traffic	25
Possible Corporate Privacy Violations	12
Misc Proxy	18

Table 5.2: Results of The Classification of Intrusions in The WSN Networks.

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

The dummy variable technique for the encoding of the strings had produced very good detection accuracy of almost 99%, but as a result of the extremely long time for feature encoding, it is not feasible for use with big datasets. The dataset significantly increases in size as a result of adding a new column for each new string. This method of string encoding may have been feasible for CIC IDS 2017 examination, yet given the size of contemporary datasets, a different solution is required. Dropping columns containing a lot of distinctive entries (like DNS Answers or IP addresses) was not taken into consideration since the goal of this initial series of studies is to establish a base-line of detection accuracy using all of data which the cluster provides. Since the subset following encoding and sampling just contained normal labels and none of malicious types, entries with "-" failed when they reached the training step. To cut down on the amount of time required for the encoding stage during tests, encoding will be somewhat altered. Like before, clustering will be used for all numeric values. Booleans have a 1 for true and a 0 for false value when encoded as values. This time, strings will be encoded as indices rather than dummy variables. In the dataset, the original column is kept, and a new one that contains distinct integer for every distinct string is created. The next experiment assesses the usage regarding such string encoding approach on the entire CIC IDS 2017 dataset, whereas originally it was just utilized for encoding the result column. The processing time for experiments is greater than eight hours. Encoding processing time has been improved. The Working-Hours dump was the HTTP traffic with the lowest detection accuracy. Working-Hours/Connection label, for instance, somewhat increased. From 98.19% to 98.68%, the accuracy of csv detection rose. The accuracy of detecting flows remained essentially unchanged. The classification results are comparable to those from the prior test and don't appear to be affected by the more distinctive labels in this instance. This is pretty intriguing since categorizing particular proxies can result in more focused alerts that might be more useful to the analyst.

According to the results, it is possible to effectively categorize malicious or undesirable behaviour within WSN utilizing a feature set that consists mostly of collective characteristics and a small number of created features. Furthermore, classification of individual protocols produced more accurate results in comparison with the classification of summary structures like links and flows. Time which is spent encoding strings (alphanumeric values) has a significant impact on how long it takes to convert the data to a numeric feature vector. Although it took more time to train the DNN, encoding texts as indices fared better than encoding them

as dummy variables. Proxy descriptions, which should be able to withstand more testing, are utilized as labels that offer comparable results rather than proxy classes. A plan should be developed to deal with the likelihood of repeated warnings for the same audit data. Good results have been obtained by the collection of all of the classifications and integrating them into one label; this method must be investigated more utilizing proxy descriptions instead of proxy classes.

Table 5.3: The Comparison of The Suggested Approach of Implementation with Existing Method

ARTICLES	**TECHNIQUES**	**RATE of ACCURACY**
[79]	(SVMs)	89.970%
[80]	Convolutional Neural Networks (CNNs)	96.730%
Proposal	MQTT based IoT Protocol	98.680%

It has been crucial to create some clear, flexible standard which might ensure the security and interoperability in the organization and control regarding such framework. With its affordable arrangement and simple execution, this framework enables any client to deliver it through his intermediary. Our solution makes use of Wi-Fi network, which the most of clients already have in their intermediaries, as communication technology. It also enables the reuse of current gadgets by using only a simple microcontroller, such as an Arduino, to connect such gadgets to the web. This is a perplexing framework, despite the fact that it has built-in measured style continuing decoupling between the modules, which allows the clients to alter the settings of the framework any time they need to, and removing or adding gadgets once the framework has been finished and functional, much like a "fitting and-play" framework. It is a framework whose primary audience is rather diverse, where it includes common clients without specialized knowledge who must unavoidably buy gadgets then integrate those gadgets into their framework or DIY Makers that must create their own IoT intermediary machines. The framework is robust and has security features whilst being adaptable, allowing clients to in-slow it with confidence. Those security gadgets are the result of planned engineering that is simple for the gadgets inside the organization, and they allow for

the distance of two different types of organizations: the external network, in which clients interface remotely for controlling the gadgets after being properly verified, and the internal organization, which is only used for intermediary organization correspondence and organizational purposes.

The center part of our framework is the brilliant intermediary server that is a gadget with seriously handling power assets, for example, MQTT-based-framework or python-based framework. This part includes 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 an MQTT representative for inside correspondence, which permits light-weight and straightforward messages amongst the gadgets. There's a backup data set as well that contains all system information, including related devices, their components, and statuses, as well as the layout of house (rooms), enrolled users, authorized user jobs, and their authorizations.

We discovered throughout our evaluation that there's a slight postponement in delivery of messages, associated with message generating time around the organization and to the number of the parts contained as well. None-the-less, such time span is brief, and perplexing activity (when a client modifies an estimation of the property) needs 1.5sec on average to be handled via system on the moment of the reception, which makes sense given required IoT gadget resources. The design of the proposed framework incorporates an instrument for overseeing clients and their consents. We proposed an answer, in any case, this was an in-

strument 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. Those security components are simple for users to use, but although such advantage was taken into account when designing the proposed solution, there was no opportunity for executing and testing it properly, which is why, there was no opportunity to show that it is sufficient to provide advantageous security.

6. CONCLUSIONS

6.1 CONCLUSIONS

The IoT has begun to draw a lot of interest recently due to the quickly expanding use of internet for establishing communications as well as access services by many sorts of devices. Numerous present services continue to use outdated protocols, including HTTP, in spite of the development and widespread use of lightweight, new protocols for IoT devices. In spite of the substantial support that such services could offer to the IoT devices, utilizing the present protocols for accessing them would exhaust the few resources on IoT devices. Therefore, a new architecture has been suggested in the present work to use the lightweight MQTT and CoAP protocols to proxy all of the internet services to the IoT devices. The suggested method executes all the operations necessary to access a particular service, obtains necessary information from their responses, and delivers them to the IoT device, if necessary, using the significantly more resources available through cloud computing than are available through IoT devices. The evaluation findings of the suggested approach demonstrate that while using various services, considerable resource consumption reductions have been made. This reduction is due to CoAP and MQTT protocols having less overhead than, for instance, the traditional SMTP and HTTP protocols. The decrease is due to storing redundant data in the suggested system and just appending dynamic values which are detected by IoT device. This reduces communication of any superfluous data. Lastly, because no extra libraries need to be compiled and saved on memory of IoT devices, the memory requirements of these devices have also been greatly lowered. As a result, the suggested approach can considerably boost any IoT device's efficiency by offering all internet services while utilizing the fewest resources.

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.