
REPUBLIC OF TURKEY

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering

# A NOVEL APPROACH FOR IOT DEVICES TO COMMUNICATE DIRECTLY WHILE OPTIMIZING COMMUNICATION SYSTEM PERFORMANCE

**Bassam Abdulabbas FARHAN**

Master's Thesis

Supervisor

Asst. Prof. Dr. Sefer KURNAZ

Istanbul, 2022

# A NOVEL APPROACH FOR IOT DEVICES TO COMMUNICATE DIRECTLY WHILE OPTIMIZING COMMUNICATION SYSTEM PERFORMANCE

**Bassam Abdulabbas FARHAN**

Electrical and Computer Engineering

Master's Thesis

ALTINBAŞ UNIVERSITY

2022

This thesis titled “A NOVEL APPROACH FOR IOT DEVICES TO COMMUNICATE DIRECTLY WHILE OPTIMIZING COMMUNICATION SYSTEM PERFORMANCE” prepared by “BASSAM ABDULABBAS FARHAN” and submitted on 15/8/2022 has been **accepted unanimously** for the degree of Master of Science in Electrical and Computer Engineering.

_____

Asst. Prof. Dr. Sefer KURNAZ

Supervisor

Thesis Defense Committee Members:

| | | |
|---|---|---|
| Asst. Prof. Dr. Sefer KURNAZ | Faculty of Engineering and Architecture,<br>Altinbas University | _____ |
| Asst. Prof. Dr. Oguz KARAN | Faculty of Engineering and Architecture,<br>Altinbas University | _____ |
| Asst. Prof. Dr Serdar KARGIN | Faculty of Biomedical Engineering,<br>Beykent 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 in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not originalto this work.

Bassam Abdulabbas FARHAN

Signature

# DEDICATION

To my Parents, the reason of what I become today. To my sister & brothers. and A special thanks to Prof. Dr. Sefer KURNAZ For his support, worth notes and close follow up.

# ABSTRACT

## A NOVEL APPROACH FOR IOT DEVICES TO COMMUNICATE DIRECTLY WHILE OPTIMIZING COMMUNICATION SYSTEM PERFORMANCE

FARHAN, Bassam Abdulabbas

M.Sc., Electrical and Computer Engineering, ALTINBAŞ University,

Supervisor: Asst. Prof. Dr. Sefer KURNAZ

Date: 08/2022.

Page: 61

With the use of internet access to establish communications amongst different types of devices, other than computers, the era of Internet of Things (IoT) has emerged, demanding the internet to adopt the new requirements of these devices. The main concerns about these devices are according to the limited resources available on them, such as bandwidth, processing power and, most importantly, energy. Despite these limitations, the limited resources are implied by the main features of the IoT devices, which are the low cost and high mobility. Hence, the need for alternative communication protocols, other than the ones that are used by default by services being provided over the internet, e.g., the Hyper Text Transfer Protocol (HTTP), according to the relatively higher resources required to use these protocols. Moreover, according to the lack of public Internet Protocol (IP) addresses, these devices are normally assigned with local addresses, which are addresses that can be used to access the internet but cannot be accessed from the internet. This limitation is according to the use of Network Address Translation (NAT), which allows multiple hosts to use a single public address to establish connections to remote hosts with public addresses. Despite the ability of IoT devices to interchange information using a midpoint server with public IP, such topology increases the length of the path the packets travel through, i.e., more delay, in addition to the need of servers with huge resources to handle a huge number of IoT devices. Thus, a new method is proposed in this study based on port-hole punching, in which

a server with public address is required to only create a direct link between the IoT devices and has no further role in the communications. Hence, the proposed method has significantly reduced the time required to deliver the packets, as well as reducing the resources required from the designated server, such as the bandwidth and processing power.

**Keywords:** Internet of Things, Porthole Punching, Point-to-point Connections, Internet Protocol.

# TABLE OF CONTENTS

**Pages**

# LIST OF TABLES

**Pages**

# LIST OF FIGURES

**Pages**

# LIST OF ABBREVIATIONS

| | | |
|---|---|---|
| TCP | : | Transmission Control Protocol |
| UDP | : | User Datagram Protocol |
| IP | : | Internet Protocol |
| OSI | : | Open Systems Interconnection |
| NIC | : | Network Interface Card |
| NAT | : | Network Address Translation |
| HTTP | : | Hyper-Text Transfer Protocol |
| DNS | : | Domain Name Server |
| FTP | : | File Transfer Protocol |
| SCADA | : | Supervisory Control and Data Acquisition |
| CMS | : | Connections Management Server |
| HTTP | : | Hyper-Text Transfer Protocol |
| XML | : | Xtensible Markup Language |

# 1. INTRODUCTION

Other than computers, numerous smart gadgets have been linked to the internet in recent years as the age of smart devices has progressed. The goal of these connections is to enable these devices to communicate with one another as well as access internet services [1]. These connections allow for the collection of many forms of data from various places, allowing for a greater understanding of the environment and better judgments. The Internet of Things (IoT) is a network of connected gadgets such as vehicles, appliances, and other household items.refrigerators and even lights have been connected to the internet [2]. However, as the internet is designed and implements mainly for computers, it has become important to accommodate the rapidly growing number of these devices and propose services that are more suitable for them to use [3].

Because the addresses used to differentiate each device are limited, one of the fundamental issues of the internet is dealing with the continually rising number of devices linked to it. Despite the fact that the internet's addressing technique can support 4,294,967,296 hosts since each address is four bytes long, the fast increasing number of devices connected to the internet, per person,, imposes the need to expand this addressing method [4]. To overcome this limitation, the internet uses private subnets, where the devices in such a network have access to each other but cannot be accessed from outside the network.However, these devices still have access to the internet using a router that redirects their traffic to the corresponding hosts on the internet [5].

As the hosts in a private network are not reachable from the internet, the available addressesin the system used for the internet are split into multiple ranges. The public and private rangesare the most important, which are used to connect devices to the internet and allow them to access the services available on the internet. Public addresses are assigned to hosts that provide these services, so that, they can be reached from anywhere on the internet. Private addresses are used for hosts in the private networks, where hosts on the internet are not allowed to use the addresses in these ranges to prevent any confusion. However, the route back to the hosts that requested the services is still required, in need for internet services to be able to send answers back to these hosts [6, 7].

Each host on the internet or in a private network can provide, or make use, of one or more services, using different applications. Thus, in addition to specifying the host that the data is targeting, it is also important to specify the application running on that host that the data must be delivered to. For this purpose, a two-byte value, known as the port number, is appended with the address of the host in order to identify the application that the traffic is designated to. Port numbers of services provided on the internet are predefined, so that, the applications can access them. However, when an application accesses a service on the network, or the internet, a random port number is assigned to that application, using any unused port numberon that host. The remote computer communicates with the host requesting the service using the host's address and port number [8, 9].

Multiple devices on different private networks may have the same address, which makes it impossible for the routers that connect networks on the internet to distinguish which of thesehosts is the traffic directed to. Thus, the router that connects these private networks to the internet manipulate the addressing information in the traffic directed out of the network. Theaddress that the packet is originated from is changed to the public address of the router, whilethe port number is changed depending on the available port numbers on that address, whichare not assigned to other applications. When the traffic is transmitted to the internet's destination host, the address information retrieved by the destination host point to the public address of the router. Thus, responses are sent back to the public address of the router, which keeps track of the changes it makes to traffic so that it can deliver it to the host that initiatedthe communication [10, 11].

Hosts that belong to private networks can initiate connections to other hosts on the internet, which have public addresses, and the domains on the internet may respond to the hosts on the private network. However, hosts on the internet, as well as hosts in other private networks,cannot initiate connections to hosts on a certain private network. Thus, to establish communications, normally, between two hosts over the internet, one of these hosts must havea public address, so that, An other device creates a link that may be used to send and receive data. However, devices on different networks, especially the IoT devices, need to interchange information in order to achieve the tasks assigned to them. For this

purpose, computers with public address, known as relay or proxy servers, are used to redirecttraffic among these devices [12, 13].

Hosts that need to interchange information over the internet are connected to the proxy server,where each host establishes the connection that allows the data to be transferred back and forth. When one of the hosts needs to send data to another, data is provided to the proxy server, together with details about the device to which the proxy server must send the data.. Using the connection established by the other device, the proxy server, then, delivers those data to the intended host. However, One such design necessitates the usage of a host with relevant application transmission capacity to handle the communicationsamong an enormous number of devices. Moreover, the fact that the data must be delivered tothe proxy server before they are forwarded to the destination host imposes security threats, as these data are revealed and cached in the proxy server. Revealing sensitive information to the proxy server does not only jeopardizes its confidentiality, regarding the proxy server, it also imposes the risk of revealing and manipulating it by attackers who gain unauthorized access to the proxy server [14, 15].

Another technique is used by computers to establish communications among computers over the internet. This technique also uses a server with a public address that can be reached by all the hosts that intend to communicate with each other. However, in this method the server's role is limited to introducing the information of the remote device to each other. This method is known as porthole punching, where both devices that intend to communicate establish connections to the server in order to reveal their public address information that can be used to deliver data to them. After delivering the information of one of the hosts to the other, or both information to each other, the communications between these devices are established directly, i.e., the traffic is initiated from one device to another without being passed through the server. This method improves the security of the information being exchanged and minimizes the route it follows to reach the destination device, compared to the use of proxy, or relay, servers [16, 17].

## 1.1 PROBLEM DEFINITION

Various sorts of gadgets are linked to the internet in order to exchange data in order to complete their functions, which has created the Internet of Things. According to the limited numberof public addresses available for the hosts connected to the internet, Before connecting to the internet, the majority of gadgets are linked to private networks. Hosts on private networks can communicate with each other and any other host with a public address on the internet. However, these devices do not have the ability to communicate with hosts on other private networks. In order to establish communications among these devices, In the earlier methods, servers were utilized. These servers have a public speech available to anybody with a connection to the internet, so that, to transmit data from one host to another, the transmitting device writes the data to the proxy server, which delivers them to the destination device. This approach has three main drawbacks, which are the security of the interchanged data, the huge resources required by the proxy server and the high latency.

Revealing the data to the proxy server imposes the risk of compromising these data by attackers that gain unauthorized access to the proxy server, in addition to the risk of revealing the data to a third-party, which is the proxy server itself. Attackers who gain access to the proxy server can also manipulate the cached data, so that, Wrong data is supplied to the destination device, which has an impact on the reception device's results relative to the faults caused by the incorrect data. Moreover, as all the data flowing from one host to another must pass through the proxy server, huge resources are required to handle all these communications, especially the processing power and the bandwidth of the proxy server. As the data flowing through the proxy server increase, the resources required from the server are also increased. Otherwise, dramatic latency is imposed in order to handle this amount of data.In addition to the resources' limitation, When a proxy server is used as a middleman in connection, the data must take a longer path to reach the target host than if the data were sent straight of one site to the other.

## 1.2 THE AIM OF THE THESIS

The study seeks to propose a communication system that allows IoT devices in different, as well as the same, private network to communicate with each other using the minimum possible route. To achieve this goal, a server is required to deliver the information that can be used to contact a certain host to the other one. However, the server in the proposed method does not have any role in the communications. though when the devices that need to engage with one other find a clear connection, the server's function ends. The proposed system uses port hole punching approach to retrieve the information of the public address that maps the communications back to the intended host. This information is forwarded to the other device, which uses it to contact the requested device and interchange the data, without sending them to the server. In addition to minimizing the path, the proposed method increases the security of the communicated data, as they are not revealed to a third- party and any extension of the communications route is eliminated. Moreover, as the role of the server is terminated by establishing the connection between the communicating device, Because the resources used are unrelated to the data being transferred among these devices, a server with low resources may manage a large number of IoT devices.

## 1.3 THESIS LAYOUT

The rest of the study is structured as follows:

i. Chapter two reviews earlier studies related to the methods and applications related to the topic of this thesis.
ii. Chapter three describes the proposed method, including the role of the server and the connection establishment procedures.
iii. Chapter four presents the experiments conducted to suggested approach is utilized to establish connections, assess the performance of the IoT device., as well as, their performance when using the existing proxy-based method.
iv. Chapter five discusses the results of the conducted experiments.
v. Chapter six illustrates the conclusions of the study.

# 2. LITERATURE REVIEW

Communications among devices connected to a network are established based on predefined protocols that define the formation that is used to transfer data from one host to another. These protocols are derived from the internet protocol, which defines the basic characteristics of the protocols used in the network. However, different protocols use different approaches to deliver the data from one host to another. Thus, each protocol has its own benefits that make it more appropriate for some applications than other [18, 19]. Two of the most extensively used Internet protocols are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) (IP). To demonstrate the unique features of each methodology and select the most appropriate one to establish point-to-point communications among IoT devices [20, 21]. These protocols, as well as, porthole punching are illustrated in this chapter.

## 2.1 INTERNET PROTOCOL

A packet sent across the network from one host to another travels through the five layers shown in Figure 2.1, which are chosen from the Open Systems Interconnection (OSI) model, in order to be translated to the proper format for the media used to link the host to the network. Each layer processes sent packets from top to bottom, whereas received packets are handled in reverse order [22]. The Network Interface Controller, as the name implies, is in charge of the physical layer activities (NIC), which converts the digital information coming from the link layer into electrical signals that can be transferred using the media that the NIC card is connected to. The duty of the link layer is to ensure that the transmitted information is directed to the requested host, on the hardware level, i.e., the NIC of the transmitting host communicates only with the NIC of the receiving host, or the host responsible of forwarding the transmitted message to the destination host. These two layers handle the communications at a very low level, very close to the hardware of the NIC, and are out of the scope of this study. However, the remaining layers, which are the network, transport and application layer, are of significant importance for establishing direct communications among IoT devices [23].

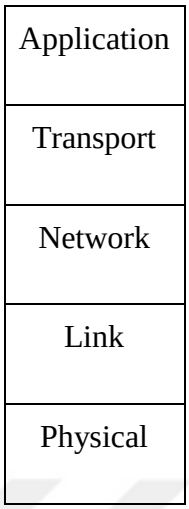


**Figure 2.1:** Communication layers used in networking

As defined by the internet protocol, each host on the network is identified using four bytes, which are known as the IP address. Such domain is established at the network layer and is used to address the data transferred across the network to its intended destination. These addresses are split into ranges based on the restricted number of usable IP addresses, which is limited by the four-byte width. Each range has a defined function. Using these ranges, it becomes possiblefor any number of hosts to be connected to each other by creating subnets. These subnets aredefined by the netmask, which specifies the range of IP addresses that a host in a certain network can communicate with, directly. The netmask is also described using four bytes, butthe values of these four bytes have different topology [8]. Some of the widely used IP rangesare [24]:

i. Public IP addresses: These addresses are unique over the internet, so that, no two hosts can have the same address. A public IP address may be in the range 1.0.0.0 to 191.255.255.255. As these addresses are unique, Any domain having a public IP address may be contacted through the web from every host.

ii. Secret Ip address is assigned to hosts who are not required to offer net access. connections. However, these hosts may access hosts that havepublic IP addresses and

communicate with them, using gateways. Three main ranges exist to be used for private IP addressing, which are 10.0.0.0 to 10.255.255.255,

iii. 172.16.0.0 to 172.31.255.255 and 192.168.0.0 to 192.168.255.255. Multiple hosts

iv. may have the same private IP address, but are nested in different subnets.

v. Loopback IP addresses: Addresses in this range point to the same host, which is normally used for testing the protocol operations and test applications on the same computer, on how to respond to requests over the network. The range reserved for these addresses is 127.0.0.0 to 127.255.255.255.

That whenever a server tries to transmit a packet, the target host's IP address is compared to the range of IP addresses in the same network, as determined by the subnet mask. The range of IP addresses that may be used is determined by comparing the bits in each byte of the subnet mask to the corresponding bits in the bytes of the IP address. be contacted directly is calculated. The first IP addressin the network is calculated by setting the bits in the IP address of the host to zeros, for eachbit that has a zero bit in the same position of the subnet mask. The last IP address in the rangeis calculated by setting the same bits to one [25]. The initial IP address of the subnet is 192.168.1.0, as illustrated in the example in Figure 2.2 as the value of the last byte of the subnetmask is 0, while the last IP address in that subnet is 192.168.1.255. This subnet can also be described as 192.168.1.0/24, as the first IP address of the subnet is 192.168.1.0, and the subnet mask consists of 24 bits with values equal to one. Thus, the first IP address of the subnet cannot be assigned to a host, as it is used to describe the range of the network. Moreover, the last IP address of the subnet is also reserved, and cannot be assigned to a host, where packets sent to this IP address are broadcasts to the network, i.e., sent to all hosts in the subnet. Thus, the last IP address is known as the broadcast IP address of the network.

| | | | | |
|---|---|---|---|---|
| IP Address | 192<br>11000000 | 168<br>10101000 | 1<br>00000001 | 10<br>00001010 |
| Subnet Mask | 255<br>11111111 | 255<br>11111111 | 255<br>11111111 | 0<br>00000000 |

**Figure 2.2:** Sample IP address and subnet mask of a host in a subnet.

If the destination host's IP address is in the same subnet as the source host, the packet is sent straight to the destination host. The IP address of the destination hosts, on the other hand, does not belong to them. The packet is transmitted to a specific device in the network, which is on the same subnet. is used to interconnect different subnet, known as the router. These routers have the ability to connected different subnets, using multiple NICs. The IP address of each NIC must be within the rangeof the subnet that it is connected to. Using tables, known as routing tables, these routers canmake the appropriate decision about the To deliver the packet to the targeted host, the packet should be passed to the next hop. If the target host's IP address is lies in the range of one of the subnets that the router is connected to, it is directly delivered to that host, otherwise, the packet is forwarded to another router [26]. This topology is the backbone of the internet, where the routing table are either statically configured in the routers, or generated automatically, by interchanging information about the subnets that each router can reach [27].

Moreover, a single host on the network may provide or access multiple services on another host, which require the identification of the application running on a certain host that the packet is targeted to. For this reason, each application is assigned with a specific number thatcan be used to distinguish that application, among many applications running on the host and accessing the network. This number is known as the port number, and have a size of two bytes, i.e., a range from zero to 65535. Thus, If a host asks that packets be sent to a specific host, the IP address of that host must be specified as well as the port number that the application is expecting the packet to arrive on. The socket is the pairing of an IP address and a port number. each connection is defined

by the source anddestination socket information [8, 28]. Normally, hosts that provide services over the networkuse predefined port number for the application that provide the service, so that, hosts can reach to those services, while random port numbers can be assigned to the applications initiating the connection, so that, the requested service can send the required information to the requesting application. Some of the well-known examples of port number are the ports number 80, for Hyper-Text Transfer Protocol (HTTP) web services, 23 for telnet, 53 for Domain Name Server (DNS) requests and 20 for File Transfer Protocol (FTP) [29].

Due to the limited amount of public IP addresses accessible, these addresses are often provided to routers that link subnets to the internet. As the routers in such topology handle all the traffic outside the subnet, they are called gateways. Hosts in the subnet sent all

packets that have destination addresses outside the range of the subnet to that gateway. However, as most of the connections to internet services require responses to be It is critical for servers outside the network to detect the path back to the host that originated the connection. Because these hosts may have private IP addresses that are replicated across networks, a subnet's gateway utilizes a method known as Network Address Translation (NAT) to mask the IP addresses of the hosts in the private network using its own public IP address [29, 30].

As illustrated earlier, normally, hosts with private IP addresses are not reachable from outside the subnet that they belong to. Thus, a response from a host on the internet cannot be sent back to the host requesting the services, as the private IP ranges do not exist in the routing tables of the routers. Many hosts may have the same IP address, which makes it impossible for the routers to distinguish the requested host. Thus, the source A host with a private The IP address and port number of an Ip vary at the channel's entrance. This gateway uses a random unused port number and its own public IP address as the source socket in thepacket. Information about this mapping is stored in a table in the gateway, known as the NATforwarding table, so that, a host on the internet sends the information back to the gateway of the network, using the socket information in the packet, instead of using the the host's private IP address. When a packet arrives at the gateway, the port number in the packet's destination socket is compared to the gateway's NAT forwarding table. Then, using the socket information of the host in the private network that originally initiated

the communication, extracted from the NAT forwarding table, to forward that packet to the intended host [31]. An example of the steps executed in the gateway to Figure 2.3 shows how to use NAT to create communication between a host on the private network and a host on the internet.

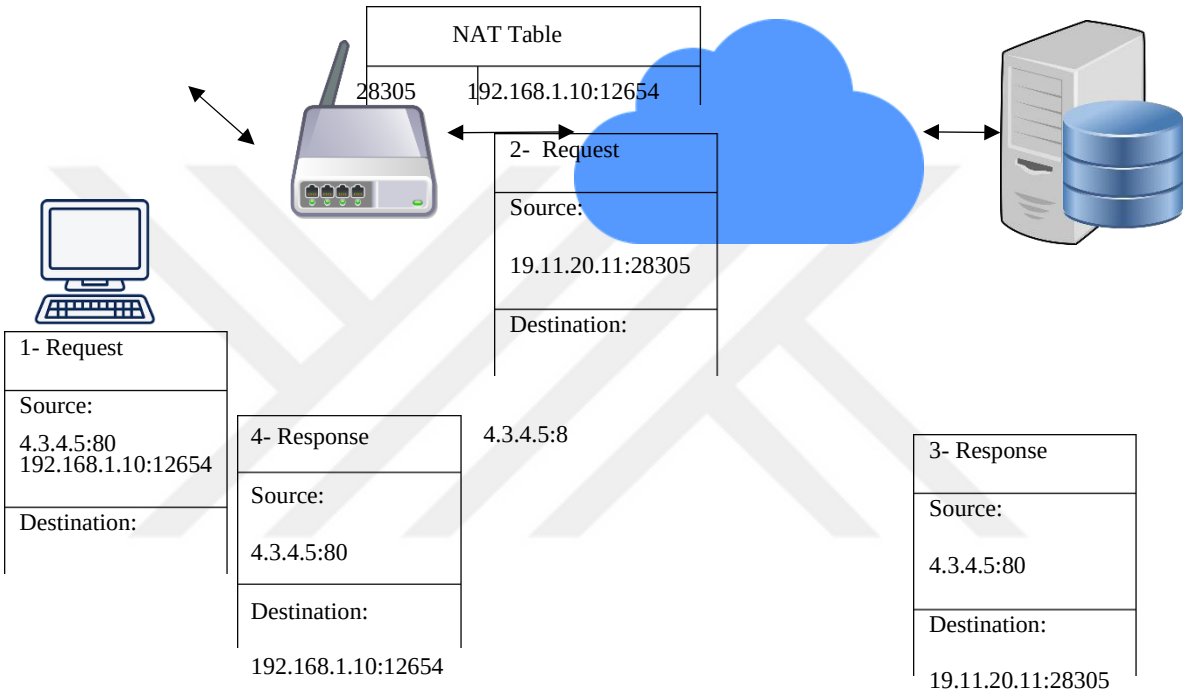


**Figure 2.3:** Sample communications with NAT.

## 2.2 Transmission Control Protocol / Internet Protocol (TCP/IP)

The TCP/IP protocol is implemented in the model depicted in Figure 2.1's transport layer, which is above the network layer and below the application layer. In order to be communicated to the target host, data transmitted by applications is separated into packets in the application layer. A three-way handshaking protocol is used to establish a connection between the source and destination sites before the packets are sent. The source host sends a connection request to the destination host, which should respond with a connection acceptance message, which the source

host confirms. Once the link has been established,the packets are sent from the source host, after being encapsulated by the TCP/IPprotocol [32]. This encapsulation adds a total of 20 bytes to each packet, shown in Figure2.4,where in addition to the standard information required by the internet protocol, information about the position of the packet in the sent data is also included. Using such

an approach, thereceiving host can rebuild the send data, even if the packets are received in a non-ordered manner. As packets may follow different routes, depending on the decision of the routers that connect the communicating hosts, some packets may go through shorter paths, while some may follow a longer one [33, 34]. Thus, these packets may not be received in the same order they are transmitted and using the position that the TCP/IP added to the packet, they are rearranged and presented to the

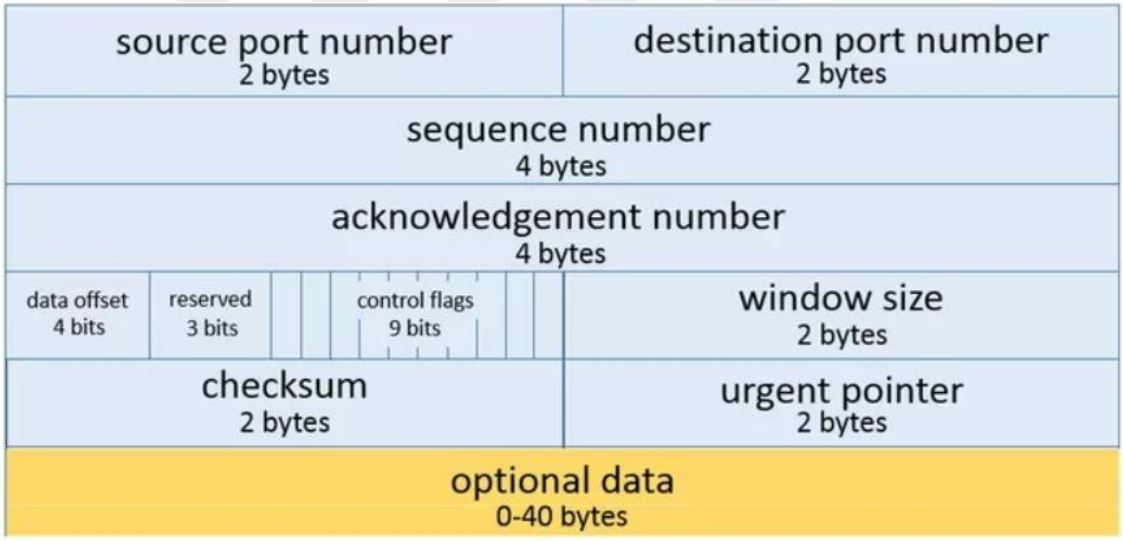


**Figure 2.4:** The structure of the TCP/IP's header [35].e application layer [35].

Moreover, each packet received by the destination host is acknowledged to the source, so that, the source host retransmits packets that haven't been recognized Although a failure to acknowledge the arrival of a packet may occur for one of two reasons, unacknowledged packets are retransmitted by the source hosts regardless of the cause since the host is unable to determine the exact cause of failure. Loss of a packet broadcast from the source to the destination, or loss of the acknowledge message sent back to the source hosts, might cause an acknowledgement failure, from the destination host. Packets retransmitted from the source host are acknowledged by the destination host, regardless of the actual reason, which is known to the destination host, because without sending back such acknowledgment, the sources host keeps sending the same packet over and over [36].

## 2.3 User Datagram Protocol

This technique is also implemented in the model presented in Figure 2.1's network layer, which transports data from the application layer to the networks layer, and the other way around as illustrated in Figure 2.5, this technique simply adds eight bytes of information to the packets. Only the source and destination port numbers, the length of the data in the packet, and the checksum of this data are included in the information., the only possible check that can be executed over a received packet is the validity of the data. This check can be executed by calculating the checksum of the data received in the packet and compare it to the checksumarrived in the packet. If the calculated checksum is different from the received one, the datais considered invalid, regardless of the possibility that the data may be received correctly butthe error occurred in the checksum of the packet. However, in most of the application, the data received in a packet that is not validated using its checksum are also forwarded to the application layer, but with a flag that indicates that the data is invalid [37].

| Source port number<br><br>2 bytes | Destination port number<br><br>2 bytes |
|---|---|
| Length | Checksum |
| Data<br><br>0-64,507 bytes | |

**Figure 2.5:** The structure of the UDP's header.

As the header of the UDP does not include any information about the position of the packetin the transmitted data, these packets are forwarded to the application layer, according to theorder they reach the destination host. Moreover, the UDP does not require any connection establishment prior to data transmission, i.e., data can be sent directly to a UDP port that is assigned to a specific application, and these data are forwarded to the application layer. However, as the amount of information added by the header of the UDP is relatively smallerthan that added by the TCP and according to the higher capacity of the UDP packet to transfer more data, it is widely used in applications that require faster data transfer, where the structure of the data is less important. An example of such an application is the Voice Over IP (VOIP) applications, where the sound is transferred from one host to another using the UDP protocol, in order to establish telephone-like communication. In such application, whena packet is received and the audio data in it are decoded and delivered to the user, any packets

prior to it become of no importance, so that, retrieving a packet that has been lost prior to thispacket has no value to the user. Thus, it is more important to make use of the higher capacityof the UDP protocol for such application [37].

## 2.4 The IEC 870-5-101 Protocol

The IEC 780-5-101 protocol is one of the widely used protocols in remote data collections systems, SCADA (Supervisory Control and Data Acquisition) systems, for example. This protocol has two types of messages, the fixed and variable length messages. Fixed length messages are normally used to send commands among the different parts of the system, while the variable length messages are normally used to send data. The popularity of this protocol is a result of the low size of the data used to encapsulate the information being sent, the need of acknowledgment per every sent packet, and the ability to detect errors in the data sent in the packet [38]. Per each variable length packet sent using this protocol, only nine bytes of data are added to the original data in that packet, as shown in Figure 2.6.

| |
|---|
| Start 68H |
| Length of data |
| Length of data (Repeated) |
| Control field |
| Address field 1 |
| Address field 2 |
| Payload data<br>.<br>.<br>. |
| Checksum |
| End 16H |

**Figure 2.6:** Structure of the IEC 870-5-101 packet [38].

Each variable length package must start with the value of 68H, which is followed by the length of that payload data repeated twice. The control field is used to append information about the communications to the packet, so that, the source device can send information to the destination device, such as whether to expect further packets or this packet is the last in the transmission. Two address fields can be used to specify the address of the destination device, as this protocol can be implemented in the transport layer in a communication system where each device can have a two-byte address. Before the packet is terminated with the value 16H, a single-byte checksum is calculated for the payload and appended to the packet, so that, the destination device can verify the incoming data in order to detect any corruption. The checksum is calculated by summing all the payload data and selecting the least significant byte of the summation results, as this byte if the most sensitive in the calculated summation against any corruptions during transmission [39].

If a package arrives at its desired target, the packet is validated using a set of rules that must be met [40]. These rules are:

i. The received packet starts with the value 68H.
ii. The value of the data length must be identical in both fields.
iii. The length field's value corresponds to the size of the payload data received in the packet.
iv. The received packet is terminated with the value 16H.
v. The checksum in the received packet is identical to the checksum calculated for the received payload data in the packet.

That whenever a packet is deemed to be legitimate, a single-byte acknowledgement message with the value E5 is sent back to the transmitter. If the transmitter doesn't get a response from the target device, the packet is retransmitted with an indication in the control field that it is a retransmitted packet. Thus, if the packet is found to be corrupted at the destination device, no acknowledgment is sent back to the transmitter, so that, corrupted packets and packets lost during communications are retransmitted by the source device.

## 2.5 PROXY SERVERS

Proxy servers, also known as relay servers, are computers that are reachable by the different devices communicating on different networks. Despite the possibility that these devices may or may not be reachable by each other, these devices must always establish communicationsthrough the relay server, as it is guaranteed to be reachable. The relay server manages the data flow from one device to another, where the destination device is defined in the data sentto the relay server. The relay server looks for the target device among the devices connected to it and transmits the data to it as soon as the data is received [41]. As shown in Figure2.7, each piece of information sent from one device to another using this approach must go through the proxy server, which also stores them in certain situations, such as logging or delivering the data when the destination device becomes available. Moreover, the figure shows that any communications between the devices A and B are required to go through theproxy server, despite both devices are on the same

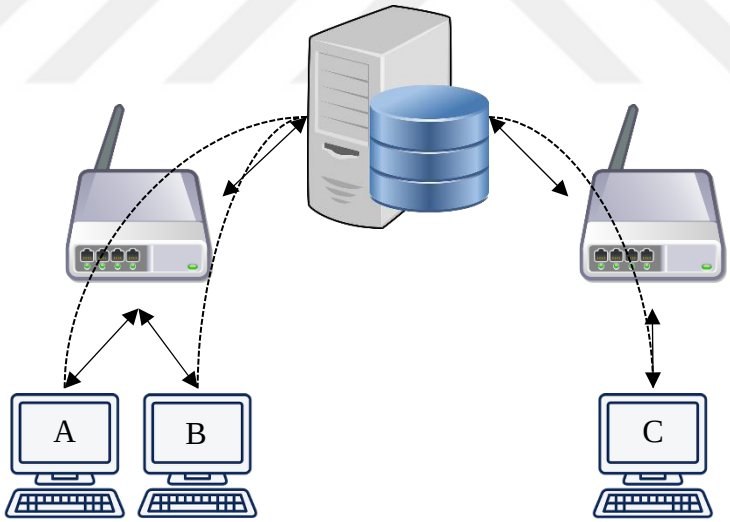


**Figure 2.7:** Illustration of proxy server topology.

Many methods are proposed to allow IoT devices to communicate using proxy servers, suchas the method proposed by Maureira et al. [42] that allows users to define variables for the IoT devices

to write values to, so that, other devices can read these values in order to deliver them from one device to another. The server also stores all the values written to the variables, so that, users can retrieve logs of the values with their timestamps. This illustrates how values

should be provided to the proxy server in order to transmit them across multiple devices, jeopardizing the security of data shared between these phones, even if it is encrypted. Moreover, this method does not provide the ability to send any confirmations between the communicating devices, unless another variable is defined for this purpose.

Another method is proposed by Banaie et al. [43] based on the use of a proxy server to accomplish communications among IoT devices. However, this method emphasizes more on the receiving device and data validation, in order to deliver the most up to date data to that device. This method uses a hybrid proxy model to queue the information incoming to the proxy server, so that, the least possible delay in values update is achieved. If the requested data is found to be outdated, A demand to transfer the most current version of the desired information is sent back to the source device. This illustrates the heavy load applied by the communicated data over the proxy server, as it must handle these data by receiving them from the source devices, store them and forward them to the destination devices.

Jin and Kim develop an IoT proxy that is based on virtual resources to interpret communications among IoT devices that use different protocols and services available on the internet. This allows the heterogeneous platform, applications and devices that comprise the internet of things to communicate with each other, regardless of their implementation. The proxy server uses the resources of the Open Connectivity Foundation to allow all these devices to connect to the proxy server and communicate with the other devices or the web services. The clients can also use consistent discovery interface in order to explore the existing devices and services on the internet that can be communicated or exploited.

## 2.6 PORT HOLE PUNCHING

As illustrated in this chapter, when a device communicates with another on a network, both devices need to know the socket information of the other device in order to transfer information. The device that is initiating the communications use a predefined socket of a device on the network that has a known address and is listening for connections on the predefined port. The socket information of the client is contained in every packet transmitted by the client, so that, the listening server can send back the information requested

from the client, or any other connection-related information. Packets outgoing from a private network to another are masqueraded at the gateway of the network, as the address of such devices is not reachable from remote networks. The gateway replaces the socket information of the source device is replaced utilizing the gateway's IP address and an arbitrary destination address that hasn't been allocated to any other sites on the network, by the gateway. Information about the original socket information and the replaced one is stored in the NAT table at the gateway, so that, when data are received on the new socket information, these data are forwarded to the original device that initiated the communication [44].

As soon as a new socket is assigned at the gateway for a specific port on a certain device, the data received on Regardless of the source socket information in the packet, it is routed to the selected device. Thus, in Figure 2.8, if the device A sends data to the device B, which lies in another subnet, and the socket information of the received packet is shared with the device C, any data incoming to that socket from device C are still forwarded to the device A. Using this approach, Device C has been able to communicate with device A directly, despite that these devices are in different networks and have private IP addresses. This approach is known as porthole punching [45].

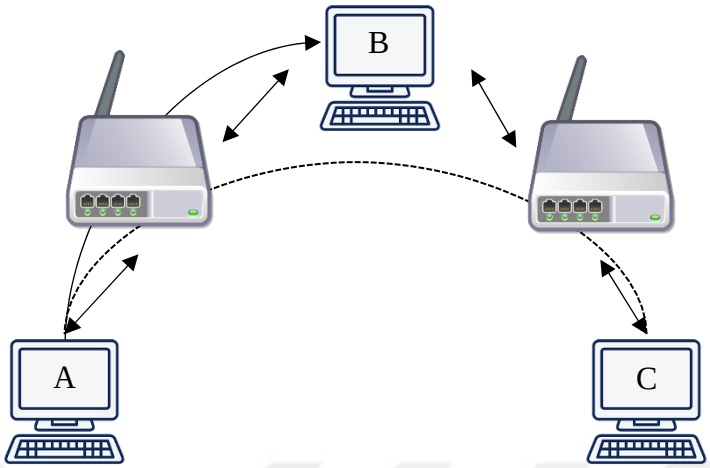


**Figure 2.8:** Illustration of the porthole punching approach.

The use of porthole punching with the TCP protocol is more complex to achieve, comparedto the UDP protocol, which required more of the limited resources on the IoT devices. This complexity is a result of the handshaking required by the TCP protocol to establish a

connection between the communicating devices, prior to any data communications, where the socket information of both devices is used during the handshaking. Moreover, the complex header added by the TCP protocol also includes sequence numbers that are used torearrange the received packet, in case the data are not received in the order they are sent in [45, 46]. Thus, must applications that employ port hole punching rely on using UDP protocol, as no connection establishments are required and the information is forwarded to the hosts in the connection regardless of the source socket information in the packet.

Using port hole punching, the path between devices from different networks is minimized, compared to the use of the proxy server, which improves the communication speed by eliminating any unnecessary hops in the path. When a proxy server is used to establish the communications, In reaching its target device, every packet must first transit via the proxy server, in most situations lengthens the route between them. Furthermore, because the proxy server reveals details sent from

hole punched increases the security of the item by allowing it being sent to the targeted system. give a clear picture by avoiding the proxy server and reducing the number of hops to the shortest possible, reducing the security threats posed by intruders and hackers.

The method proposed by Sung Woo Cho [47] uses UDP port hole punching to communicate sensitive and personal information among devices in a social network. Instead of the existingsocial networking methods, such as Facebook, Yahoo and Twitter, this method does not usea centralized computer to store and exchange this information. The server in this method onlyexchanges network socket information among related devices, which is used by these devices to communicate the required data. Using such an approach, the communicated data is never delivered to the server, which improves the security of the personal and sensitive data being communicated. This application illustrates the importance of employing UDP hole punching regarding the security of the communicated data [48].

An update distribution system is proposed by Herry et al. [49] that employs UDP port hole punching to transfer a large amount of data between different devices on different networks. Instead of using a single host for all computers to download these updates, any host that has the update also has the ability of communicating with the host requesting the update using porthole punching. Using such an approach, the load on the single host that has these updatesis reduced, as the load is distributed over different networks or hosts. This application illustrates the reduction in the bandwidth consumption from the main server that is providing the server, as well as the resources required to handle these connections. Thus, better servicesare provided to the end users, and less resources are required from the main server, which reduces the expenses required for the bandwidth and resources.

# 3. METHODOLOGY

The method proposed in this study relies on a computer that is reachable by all the IoT devices to manage the connections among different IoT devices, hence, the server is denoted as the Connections Management Server (CMS). The main task required from the server is to establish point to point connections among the IoT device, where these connections are used for data communications directly from one IoT device to another, without sending any data to the CMS. However, IoT devices would still keep a communication link in order to exchange orders with it.

## 3.1 CONNECTIONS MANAGEMENT SERVER

In order to create point-to-point connections with other IoT devices, any IoT device that desires to utilize the proposed approach must first establish and maintain a connection with the CMS using TCP/IP protocol. This connection is never used to interchange users' data but it is mandatory for the CMS to send command and receive requests from the IoT devices. An IoT device is registered to the CMS using a device ID and a secret key, which are used to authenticate the IoT device to the server when establishing the TCP/IP connection. After the connection is established and the device is authenticated, it is the device'sresponsibility to maintain that connection, by checking the status of that connection and establish a new connection in case the older connection's state changes to disconnected.

When a connection to a certain IoT device is required, the source device sends a request to the CMS asking for the connection information of the destination device. This request must be initiated from an authenticated device and includes the ID of the destination device and its secret key as well, to deny any unauthorized access. As soon as the request is received from the source device, using the TCP/IP connection, the credentials of the destination device are authenticated against the database of the registered devices. If the requested device exists in the database, the CMS scans for that device in the list of devices connected to the server. Then, the CSM sends a new connection request to the destination device, if it is found connected to the server. Otherwise, The source device receives a failure message.

Whereas if IoT destination device is found in the list of devices connected to the CMS, the server starts listening for new UDP connection on any unused port number. In order to protect the CMS from attacks that attempt to connect to any available listening port, which denies the legitimate IoT device from making use of the service provided Each seeking to bring causes the server to create a random string. This random string must be unique among the strings created for the devices presently connected to the server, and it is delivered to the destination device together with the CMS's expected port number.When the destination device receives a connection request from the CMS, through the TCP/IP connection, with the random string generated for the new connection The target IoT device initiates a new UDP connection using the port number on which the CMS is listening for a connection. The random string obtained from the CMS over the TCP/IP connection is transmitted back to the server to verify the connected device's validity. The CMS examines the source IP address in the received UDP packet, which must be equal to the source IP address in the TCP connection, as soon as the connection is established and the string is received. If the IP addresses are same, the CMS is activated. validates the received string againstthe one sent to the IoT device, which proves that it is the same IoT device that the connectionis requested from, as it is possible to have multiple devices sharing the same public IP address.

The source socket information in the received packet, which includes the IP address and port number, is obtained by the server and transmitted to the IoT device that requested the connection when a new UDP connection is verified at the CMS. As the necessary entries in the NAT tables are already made by the routers responsible of connecting the IoT device to the internet, when the device has sent the string to the assigned UDP port at the CMS, the route back to the device is already established using the socket information provided by the CMS. Thus, the IoT device that has requested the connection can use to socket information receivedfrom the CMS to send the data to the requested device directly, without forwarding them to the server. Thus, the UDP port that the CMS server has used to wait for a connection from the requested device is no longer needed and can be closed or used with other devices.

**Algorithm 3.1**: summarizes the steps required to establish point to point connections using theproposed method.

| | |
|---|---|
| **Algorithm:** Creating an IoT stage communication. This method establishes internet connections amongst IoT devices. The asking device is represented by the The desired IoT device is represented by the Interval scale, whereas the function f represents the desired Iot system.<br>**Input:** As in an IoT device with ID=y, a connection request is sent to a Device with ID=x. Cx and Cy interfaces are intended to be used to communicate those devices to the server. | |
| Step1: | *//Initialization*<br>*As a string, use the socket /Storage of the asked device's socket details, to be transmitted to the asking one.*<br>*Verify to see if the device you're looking for is linked to the service that manages connections.*<br>*If x can sometimes be accessed via Cx, then*<br>*Strings like a one-of-a-kind string* |
| Step2: | */Instruct the desired device to use the existing connection to establish a new connection.*<br>*Deliver a message to machine x to establish a new communication to*<br>On any CMS-accessible ports (P), the CMS uses Like for verification. |
| Step3: | */Wait again for requested object to interact with the CMS to use the CMS's new bridge.*<br>*Nevertheless (On line P, there is no new connection from x)*<br>*Using port P, look for a new connections from x.* |
| Step4: | / Can get external port data for the user's includes a pair to line P.<br>*Connection Access data about the distant socket for the connection made via port P.* |

| | |
|---|---|
| Step5: | If *Socket* and *AS* are valid: |
| Step 6: | Even if everything fails, notify the asking device that the desired item is unavailable.<br><br>Convey the signal "Unattainable" to device y through Cy. |

## 3.2 POINT TO POINT CONNECTIONS ESTABLISHMENT

After illustrating the role of the CMS server in retrieving and providing the information required to establish a point to point connection by the IoT device, it is important to describe how such connections are established, from the IoT devices' point of view. First, each IoT device connects to the CMS using the TCP/IP protocol and provide its authentication credentials. Devices that are not willing to communicate with others are also required to connect and authenticate to the CMS, so that, if another IoT device requires communications with them, the required device can be reached by the server. Each IoT device is also required to monitor the state of that connection, so that, new connections are established if the current connection is lost for any reason.

When a connection to another IoT device is required, the requesting devices sends the login credentials of the requested device to the CMS server, in order to prove the right to communicate with that device. The requesting IoT device, then, receives the socket information, The desired IoT device's public IP address and destination port are specified by the visible Id number, as well as the device id The inquiring equipment transmits the data necessary for transmission to the requested item using the socket information acquired.

from the content management system. After the data have been transmitted to the destination device, the connection can be disposed as it is no longer required. Algorithm 3.2 illustrates the key steps required to by the proposed method from the requesting device to send data to another IoT device.

**Algorithm 3.2:** Establishing point to point connection for data transmission.

| **Algorithm:** Passing packets of data from x to y IOT through connection | |
|---|---|
| Step1: | */ a fresh link attempt to the CMS server for the needed item.*<br>*Obtain connectivity data as from CMS for item (y).* |
| Step2: | // To use the data from the CMS, access to the required device.<br>Utilize Info to communicate to an IoT device. |
| Step3: | /Send all content to the specified device to use the connections based on the connection evidence gleaned from of the CMS, without contacting the CMS again. In the messages set, for each message:<br>Deliver the package to y immediately. |
| Step4: | /Close the desired device's connections.<br>y's connexion should be terminated. |

A requested IoT device, on the other hand, is only required to send the random string sent from the CMS, through the TCP/IP connection, back to the server using UDP protocol and the port assigned for that device by the server, which is sent alongside with the string through the TCP connection. As soon as the server authenticates the received information, the requested IoT device receives the data directly from the IoT devices that requested the communications. The socket information that permits this device to interact with the device that requested the communications becomes accessible as soon as the first packet is transmitted to the requested device. This information may be derived from the received packet's socket information, which is found in the source's socket information. As a result, data may be sent back and forth between these devices. These three stages are shown in Figure 3.1 shows how to link two Iot technologies in a moment in time link.

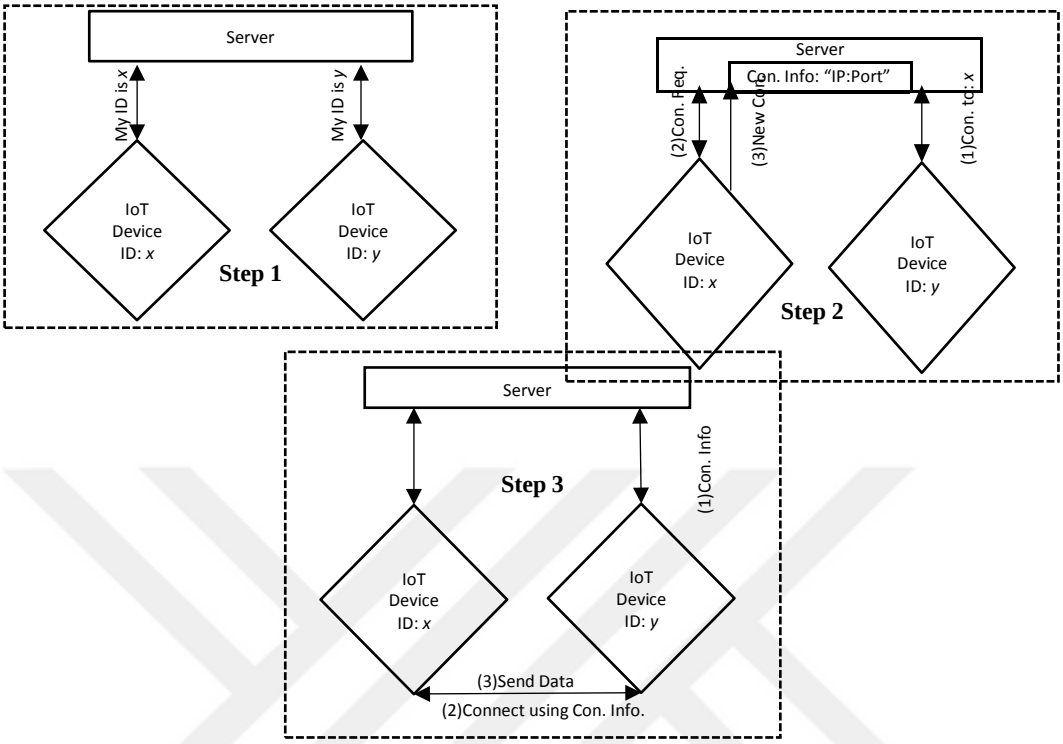


**Figure 3.1:** This figure depicts the three steps required to establish a node link among Iot systems using the proposed technique.

Moreover, as there is a significant chance that the IoT devices that require exchanging information placed in the same environment, and most probably to be connected to the same subnet, another approach is also utilised in the suggested technique, which tries connections first using the desired IoT device's local socket details. As such information isnot known to the CMS, this information is retrieved by the requested device and delivered tothe server with the random string used for authentication. The server forwards both the localand public socket information to the requesting device. The asking device then uses the local socket details to try to connect to the other device. If the host isunreachable or the port is not listening for connections, the requesting IoT device attempts toconnect using the public socket information. Although such approach may add another step to the connection establishment procedure, communications using the local network can reduce the latency of data delivery, eliminate unnecessary processing by the gateway and improve the security of the communicated data, as it never reaches the public network. Moreover, devices that communicate using the local network become independent of the

public IP address of the router, so that, losing connectivity to the internet does not interrupt these communications.

### 3.3 APPLICATION-LAYER PROTOCOLS

As the IoT devices communicate with the CMS and each other using TCP and UDP protocols, a predefined format is required, so that, the received information can be interpreted by the receiver. As the TCP/IP protocol guarantees the delivery of the information communicated between two devices intact, i.e., complete and in the same order they are transmitted in, a very simple protocol is implemented to transfer the commands and requests between the IoT devices and the CMS. The information being sent from one host to another using the TCP/IP connection are predefined, so that, no definition is required for the information in the protocol. Thus, two symbols are selected to indicate the beginning and the end of each piece of information, where the symbol < is placed before the first alphanumerical character and the symbol > is places after the last one.

This configuration decreases the amount of processing needed by IoT devices, since using more complicated protocols like Hyper-Text Transfer Protocol (HTTP) or eXtensible Markup Language (XML) takes more processing and therefore more resources. from the IoT device. Figure 3.2 shows the formation required from IoT devices to send their authentication credentials and the data received from the CMS to initiate a new connection, where the top message represents the message sent from the IoT device, with ID ‘IoT1’ and secret key ‘Secret-Key1’, to the CMS for authentication. The middle message is an example of a requests for a connection establishment on port 5261 from the CMS to the IoT device, using the random string ‘RandomString12’ for authentication. The bottom message is an example of a message sent to an IoT device that requested communication with the device ‘IoT1’, providing the device's IP address and port number.

<IoT1><Sercret-Key1>

<5261><RandomString123>

<35.28.16.41><3128>

**Figure 3.2:** Examples of the messages communicated among the devices in the proposed method.

Top: Authentication credentials from an IoT device to the CMS; Middle: new connection request

Moreover, as some applications require communicating sensitive data, where the completion and arrangement of the data are important, and as the communications between the IoT devices is implemented using UDP protocol, it is important to employ an application layer protocol that ensures the arrival of the transmitted data as it is sent. For this purpose, the IEC780-5-101 protocol can be used, which is illustrated in Section (2.1.3). The transmitting IoT device waits for a confirmation message from the receiving one indicating that the data in the last transmission has been received and validated. Next, the transmitting device transmitsthe next packet, until the entire data is sent. The last packet created in the application layer, for the payload data, is marked at the control field, so that, the receiving host can recognize that this packet represents the end of the data being transferred. The data from the received packets are extracted and appended before being forwarded to the application that isdesignated to interact with these data.

## 3.4 SUMMARY

Although the IoT devices rely on the CMS to establish communications among them, the server has no role in the data exchanging among these devices. The customer's role changes when a clear link is created among Internet of things which need to exchange information function ends. This approach improves the speed of the communications among the devices, reduces the resources required from the server and secures the data flowing from one IoT device to another, as they are not revealed to the server. According to the very low resources required to manage the communications among any number of IoT devices, the proposed method can be run using on-

premises, as well as, cloud servers.

The IoT devices can establish the communication in a simple three-step procedure, by authenticating to the CMS, requesting the socket information of a certain IoT device then connect to that device. The communications are formed using the Tcp / ip protocol, which lacks the capacity to check and rearrange the received data to match the broadcast data in the event of any changes occurring during transmission. As a result, for applications that transfer sensitive data that must be received precisely as it is provided.

# 4. EXPERIMENTAL RESULTS

Despite the illustrated benefits of using the proposed method in establishing point to point communications among the IoT devices, It is critical to guarantee that the suggested approach has no negative impact on the IoT device's efficiency. The performance of the IoT devices in communicating data can be described using the energy consumption, time and total amount of data interchanged in conveying the user's transmitted signalThus, three experimentsare conducted, one using the existing proxy-server-based technique and two using the proposed method, to evaluate the performance of IoT devices while transmitting a set amount of information from one to another

For evaluation purpose, two of the popular ESP8266-E12 microcontrollers with embedded WIFI modules, which are widely used in IoT devices, are used in the implementation of the experiments. For accurate evaluation, all servers that are used in the experiments to establish communications among the IoT devices are implemented in an Amazon Web Services cloud server, with one virtual CPU running at 2.5GHz frequency and2GB of memory. The IoT devices are powered through an energy meter, to calculate the consumed energy, The gateway device collects the quantity of data sent from and to the IoT devices. Timers are also connected to the IoT device, which are controlled by one of the pins of the ESP8266-E12, so that, the timer is started before data transmission and halted as soon as all the data is transferred to the other device. The servicesthat run on the server are implemented using The Microsoft Visual Studio development environment allows you to use the C# coding language. Applications that run on Connected systems and allow data to be transferred from one device to another are implemented using C++ programming language, using the Arduino integrated development environment.

## 4.1 EXPERIMENT A – USING RELAY SERVER

This research looks at the present approach of employing a relay, also known as a bypass, system to control the flow of electricity send data across numerous linked devices. The structure used for this analysis is shown in Table 4.1, Both results were used as control values for comparing the effects of the various approaches.

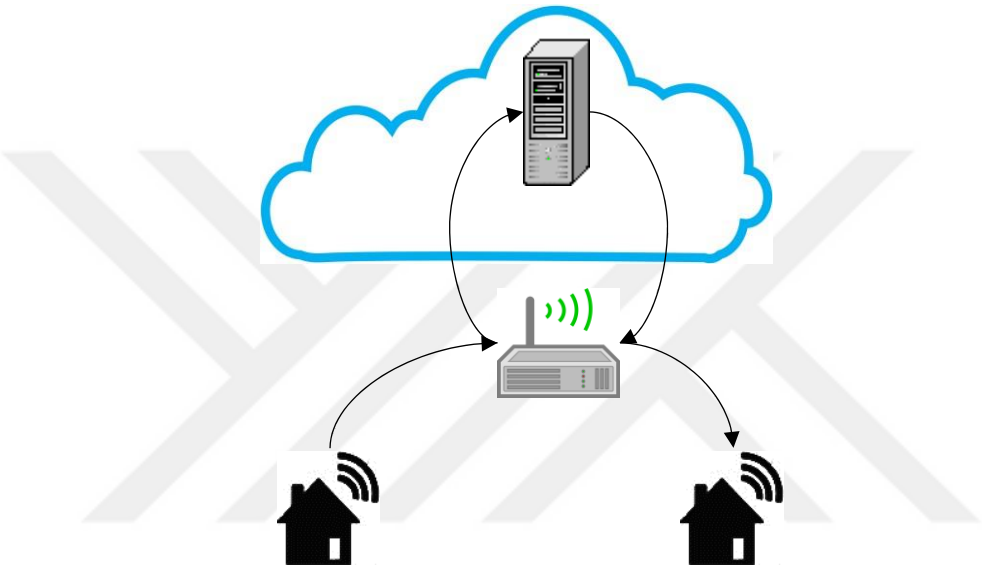

**Figure 4.1:** The design for using a gateway network to transfer data from one Iot system to some other [50].

Any packet sent from one IoT device to another must pass via the relay server, as indicated in the diagram. This lengthens the journey that the data must go and jeopardizes the data's security since it must be given to the relay server. Table 4.1 summarizes the outcomes of this investigation.

**Table 4.1:** Observations of the an IoT device utilizing a relaying server to convey information.

| | | | Data (Bytes) | | |
|---|---|---|---|---|---|
| | **Energy(mWh)** | **Time (Sec)** | **Sent** | **Received** | **Total** |
| **1K** | 0.734 | 6.478 | 1863.00 | 572.00 | 2435.00 |
| **100K** | 3.886 | 44.244 | 103615.00 | 2403.00 | 106018.00 |
| **1M** | 12.716 | 122.342 | 1572864.00 | 14538.00 | 1587402.00 |
| **Avg** | **5.78** | **57.69** | **559447.33** | **5837.67** | **565285.00** |

## 4.2 EXPERIMENT B – USING CMS AND PUBLIC SOCKET INFORMATION

In this research, channel punched is accomplished utilizing connection data obtained first from CMS side of the connection, i.e., no knowledge well about IoT device's local socket details is accessible. Even though the transmitting and recipient Connected technologies should be on the same private network, connection is handled thru the network's internet gateway, It connects the local network to the internet using a conventional gateway. Links are broken if the gateway loses its internet connection, for instance, owing to a connection reset, and the Website must frequently re knowledge here about how to contact the target server.The topology used in the stage 2 of the this test is shown in Figure 4.2, where first phase, obtaining information from the CMS server for the target device, is only done again.

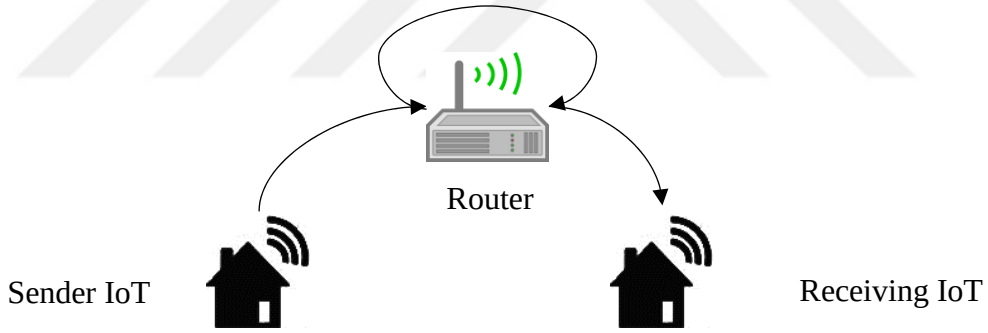


**Figure 4.2:** Representation of the structure utilized to transport data to use the suggested approach through public socket port punching.

As shown in the figure, each packet of information needs to be delivered to the gateway of the network, as these packets are directed to the public IP address of the subnet, which exists in the router. Then, the router uses the NAT table in order to recognize the host that the packet should be delivered to. Thus, in addition to the dependency on the public IP address, this approach also

increases the processing required by the gateway router to deliver the packet. To scenarios are experimented using this topology. The first scenario does not use any protocols at the applications layer, i.e., data is sent and received using the UDP protocol without any modification. Table 4.2 displays the data obtained from the transmitting IoT device.

**Table 4.2:** Observations of IoT devices exchanging information on the network socket using port punching and no application layer protocol.

| | | | Data (Bytes) | | |
|---|---|---|---|---|---|
| | **Energy (mWh)** | **Time (Sec)** | **Sent** | **Received** | **Total** |
| **1K** | 0.53 | 3.15 | 1161.00 | 24.00 | 1185.00 |
| **100K** | 2.33 | 26.12 | 102537.00 | 24.00 | 102561.00 |
| **1M** | 6.36 | 61.07 | 1048713.00 | 24.00 | 1048737.00 |
| **Avg** | **3.07** | **30.11** | **384137.00** | **24.00** | **384161.00** |

The IEC 870-5-101 scheme is based at the application level in the later case to verify that the data is received precisely as it was delivered. Packets are sent one by one, where a confirmation is required from the receiver to ensure that the transmitted packet is received and validated, before the next packet is transmitted. The last packet in the transmission is marked at the control field, so that, the receiving device can detect the end of the transmission, append the received packets and forward it to the application responsible of processing the received data. Table 4.3 depicts the data collected from the transmitting device in order to complete the transmission of data.

**Table 4.3:** Observations of such an IoT device sending data over a public socket employing port punching and the IEC protocol on the application level.

| | | | Data (Bytes) | | |
|---|---|---|---|---|---|
| | **Energy (mWh)** | **Time (Sec)** | **Sent** | **Received** | **Total** |
| **1K** | 0.63 | 5.21 | 1186.00 | 49.00 | 1235.00 |
| **100K** | 3.51 | 35.72 | 104947.00 | 2434.00 | 107381.00 |
| **1M** | 10.58 | 95.18 | 1073386.00 | 24696.00 | 1098082.00 |
| **Avg** | 4.91 | 45.37 | 393173.00 | 9059.67 | 402232.67 |

The findings demonstrate that using the application layer protocol enhanced the IoT device's usage of resources. The additional time needed for processing the data, splitting them into packets and encapsulating these packets before being transmitted over the network has also increased the energy consumption, compared to transmitting the same amount of data using the application layer protocol is not required. Furthermore, the data added to the payload data has also increased the amount of transmitted data, which also increases the energy consumption, according to the extra power required by the network module to transmit these data. The findings also reveal that the quantity of data received by the transmitting device is now dependent on the amount of data sent, which is a result of the confirmation messages sent back from the receiving to the transmitting devices. However, the overall energy consumption and time required to send the same amount of data has remained below those required by using the relay server topology.

## 4.3 EXPERIMENT C - USING CMS AND LOCAL SOCKET INFORMATION

The goal of this experiment is to evaluate the performance of Iot systems while they are on the same subnet and share local socket data over the CMS, using the suggested technique. As a result, the requested IoT device's confidential socket information is sent to the CMS server, where it

Because it cannot be retrieved by the CMS, it gets known to the CMS host. Such data is provided to the requesting machine alongside public port details, which can be seen in Figure 4.3, As a result, the enquiring machine tries to connect towards the output port using secret network data at first. If the desired device cannot be reached using this information, the connection is established using the public socket information.However, as the IoT devices in this experiment are connected to the subnet, the private socket information is used to communicate their data.

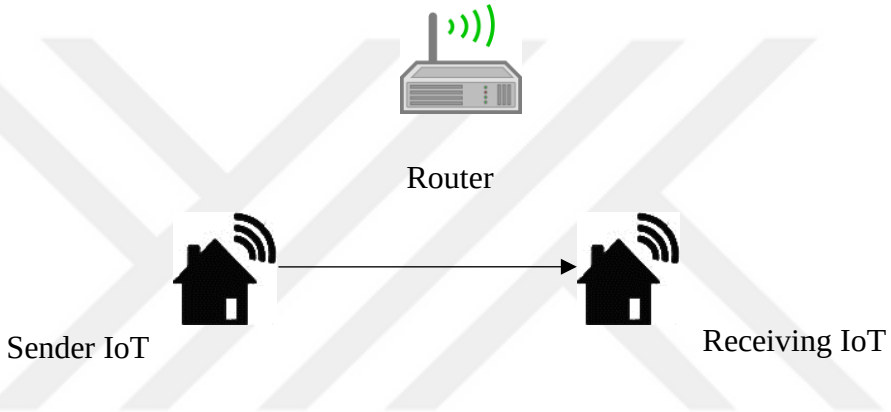


**Figure 4.3:** The architecture used to send data utilizing a private socket port punching is shown in this diagram.

As the figure shows, the data is transmitted directly between the communicating IoT devices, without being forwarded to the gateway of the network after retrieving the local socket information from the CMS. Thus, the communications between these devices do not rely on the gateway router or the public IP address, hence, losing internet connectivity or even losing the gateway router does not interrupt the communications between the devices. The performance measures collected from the transmitting device, Table 4.4 shows the results without utilizing the application layer protocol.

**Table 4.4**: Observations of the an Iot system sending data over a local connection via point punch.

| | | | **Data (Bytes)** | | |
|---|---|---|---|---|---|
| | **Energy (mWh)** | **Time (Sec)** | **Sent** | **Received** | **Total** |
| **1K** | 0.17 | 1.48 | 1161.00 | 38.00 | 1199.00 |
| **100K** | 0.81 | 11.08 | 102537.00 | 38.00 | 102575.00 |
| **1M** | 3.20 | 24.87 | 1048713.00 | 38.00 | 1048751.00 |
| **Avg** | **1.39** | **12.48** | **384137.00** | **38.00** | **384175.00** |

The performance of the IoT device transmitting the same data, The service layer's use of the IEC 870-5-101 protocol is also assessed. Table 4.5 summarizes these measurements for the transmitting device.

**Table 4.5:** Observations of an IoT device sending data via a closed connection utilizing port bashing and the IEC protocol at the protocol stack.

| | | | Data (Bytes) | | |
|---|---|---|---|---|---|
| | **Energy (mWh)** | **Time (sec)** | **Sent** | **Received** | **Total** |
| **1K** | 0.24 | 1.71 | 1186.00 | 63.00 | 1249.00 |
| **100K** | 1.31 | 13.68 | 104947.00 | 2448.00 | 107395.00 |
| **1M** | 4.25 | 38.97 | 1073386.00 | 24710.00 | 1098096.00 |
| **Avg** | **1.93** | **18.12** | **393173.00** | **9073.67** | **402246.67** |

The findings reveal that confidential socket data may be used to create conversations among IoT devices. in the same subnet has been able to reduce the difference between the resources required when the IEC protocol is used in the application layer, and when it is not used. Moreover, The amount of bytes received by the asking End devices has also grown marginally, according to the data, which is a result of sending the local, in addition to the public, socket information by the CMS. However, this information is sent once, which illustrates the slight increment, but it has significantly reduced the resources consumption atthe transmitting IoT device.

# 5. DISCUSSION

The average performance of the transmitting device using the existing relay server method, as well as the different scenarios implemented using the proposed method are summarized in Table 5.1. This summary allows comparing these results to each other and the existing method, to illustrate the improvement in performance and the suitable scenarios that can be implemented for different applications.

**Table 5.1:** Summary of the transmitting device's average performance in the conducted experiments.

| | | | | Data (Bytes) | | |
|---|---|---|---|---|---|---|
| | | **Energy (mWh)** | **Time (Sec)** | **Sent** | **Received** | **Total** |
| **Relay Server** | | 5.78 | 57.69 | 559447 | 5837.67 | 565285 |
| **The Proposed Method** | **Public Socket.** | 3.07 | 30.11 | 384137 | 24 | 384161 |
| | **Public Socket + IEC.** | 4.91 | 45.37 | 393173 | 9059.67 | 402233 |
| | **Private Socket.** | 1.39 | 12.48 | 384137 | 38 | 384175 |
| | **Private Socket + IEC.** | 1.93 | 18.12 | 393173 | 9073.67 | 402247 |

Based on this summary, Figure 5.1 illustrates the average energy consumed by the transmitting IoT device, for all the evaluated approaches, to transmit the exact same amount of data. These dialogues are completed in the time it takes to transport data from one IoT device to another. Figure 5.2 depicts the time it takes to transmit the same data from one IoT device to another under different scenarios and approaches. layer has increased the energy consumption, compared to the applying the same topologies without the IEC protocol in the application layer.

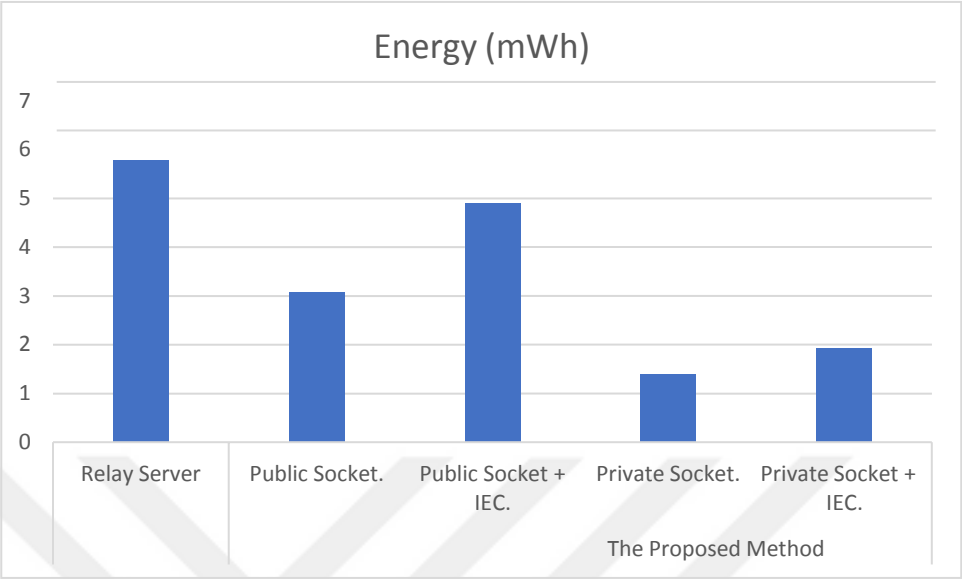


**Figure 5.1**: Illustration of the average energy consumption by the transmitting IoT device in the evaluated topologies.

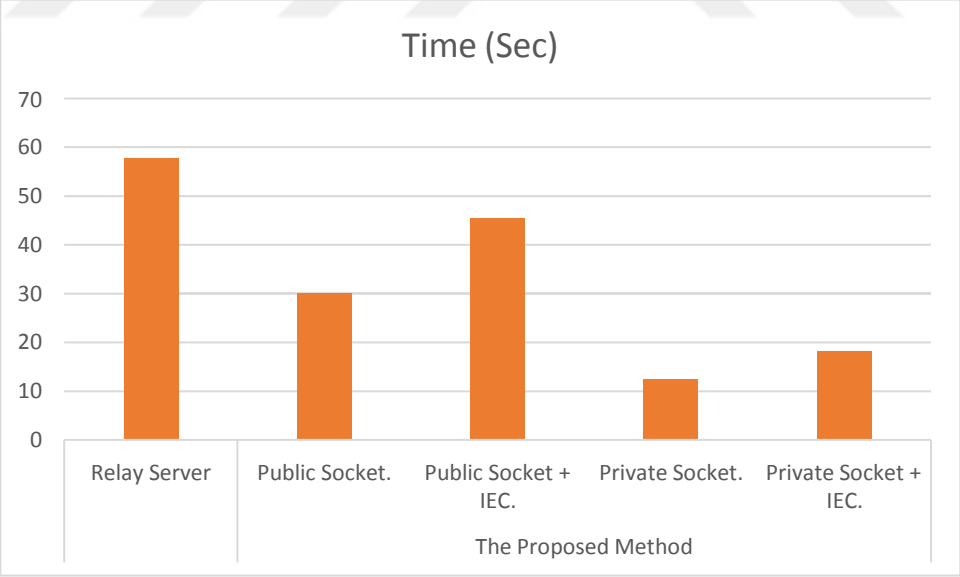


**Figure 5.2:** Average time required to transfer the data in the evaluated scenarios.

Figure 5.2 shows the amount of time it takes to transmit the same data from one IoT device to another using various methodologies and This comparison shows that in addition to the security improvement imposed by the proposedmethod, Because of the restricted connection of such units in specific application, the amount of data sent and received by the receiving End devices is indeed important to monitor. The numbers of bytes delivered and received by the transmitting IoT device on an average basis.. Even the extra time required to prepare the data for transmission, using the IEC 870-5-101 protocol, the communications have been faster than using the proxy server, despite the use of the gateway's public IP address. Moreover, the use of local socket information to connect to the requested device, directly, has also shown significant impact over the communications speed. Thus, despite the slight difference that the addition of the local socket information exchange poses to the communications, the difference in speed is significant, which makes this addition efficient.

Because of the restricted connection of the these devices in specific application, the volume of data moved from and to the transmitting IoT device is also vital to track. The numbers of bytes delivered and received by the transmitting IoT device on an average basis, as well as their totals are illustrated in Figure 5.3.

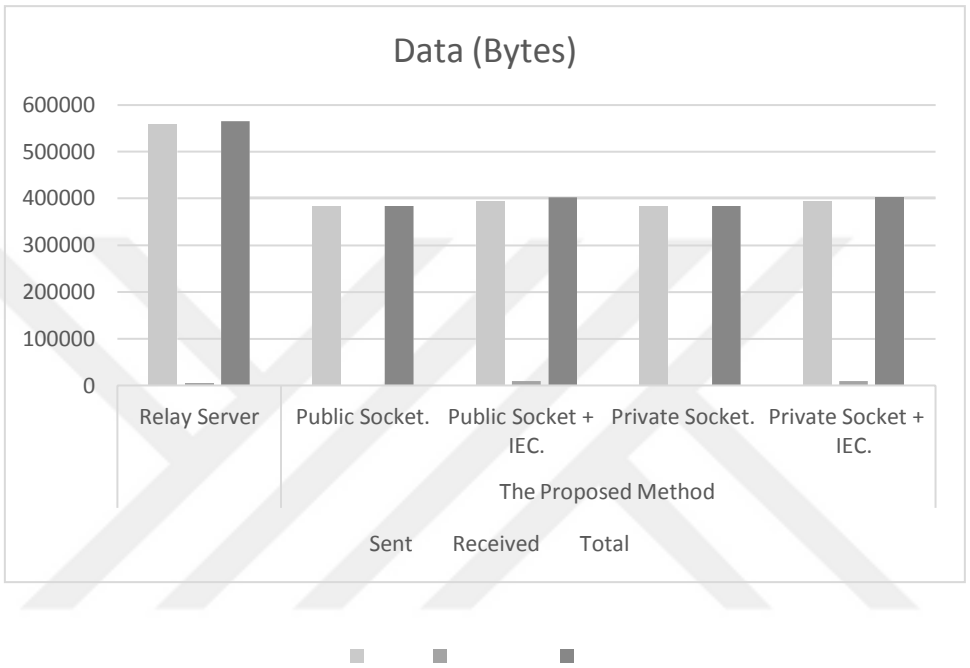


**Figure 5.**3: Illustration of the average number of sent, received and total bytes.

The comparison illustrates that the suggested solution uses the UDP protocol rather than the TCP/IP protocol to interact via the proxy server, the amount of additional data required to communicate the same payload data has been significantly reduced. This communication data reduction improves the performance of the IoT device on two aspects.

It reduces the bandwidth consumption, which can be very limited in certain connections, and the power required to transfer the data to the next hop, which is the WIFI access point in the conducted experiments. Moreover, despite the slight size addition that the use of IEC 870-5-101 introduces to the communications, which has is very small compared to the use of the proxy server and to the sent data, comparing the scenarios that use this protocol with those that do not show that it is relatively high. Thus, it is recommended to limit the implementation of this protocol in the application layer to only applications that require transferring sensitive huge data. According to the high capacity of the UDP packet, it is most likely that the entire data is sent in one packet, which eliminates the risk of receiving the data in a different arrangement than the one they are sent in.

A further significant advantage of this form method is the ability to regulate data flow among communicating devices. When a proxy server is used, different policies are applied to deny any abusive use of the provided services, which can be exploited by attackers to deny legitimate users from accessing these services. One of these policies in the limited number of read or write commands in a certain interval of time, which imposes a limit to applications that required real-time monitoring. The method proposed by Sharmad Pasha [51] uses the proxy service provided by ThingSpeak in order to transfer information from remote IoT devices and analyze the retrieved data using Matlab. The use of the proposed method in such an application can significantly improve its performance by controlling the number of samples collected from the remote IoT devices per second. As the CMS has no role in the communications, It is unaffected by the amount of the data or the frequency in which it is collected, allowing users to choose the rate that is best for their purposes.

Another application proposed by Bharadwaj et al. [52] monitors the vital signs of patients in order to find the appropriate diagnosis. The ECG and the patient's body temperature are the most often monitored signals. These data are sent to the doctors through a proxy server, which also limits the rate of data transmission, as well as the speed of communications. According to the importance of communication speed in such application, the use of the proposed method can significantly improve the diagnosis decisions made by the doctors, by allowing more data in a shorter time.

However, according to the sensitivity of the data communicated in this application, It is suggested that the IEC 870-5-101 be implemented in this system's application layer. So that, the received information is accurate and ordered inthe same order they are transmitted in.

A smart surveillance system based on the use of IoT is proposed by Chandana and Javeed [53]. This system also uses ThingSpeak’s proxy server to establish the communications between the IoT devices that are collecting the information from remote sites and the observer of the system. In order to notify the observation, the data is available to the proxy server, and according to the sensitive security of the information communicated in such system, revealing and storing the information on the proxy server poses security threats to the users of the system. An intruder that can hack into the proxy server can manipulate the data stored on the server, before being delivered to the observer, as well as the risk of compromising such sensitive data, which may enable intruders to collect and analyze these data. Thus, It is feasible to increase security of interactions to use the suggested strategy., as the information is exchanged directly between the remote devices and the observer. However, encrypting the data before being sent to the observer can improve the security of the communications, so that, hackers that are passively listening to the communications cannot interpret the intercepted packets.

# 6. CONCLUSION

The number of devices that use the internet to establish their communications is growing rapidly in the recent years. Different types of devices are being connected to the internet, which is mainly designed to be used by computers, This has ushered in the Internet of Things era. To connect with one another and share data, the majority of such gadgets rely on the internet. The restricted amount of public IP addresses, on the other hand, necessitates these devices to be linked to private local networks that are connected to the internet through a gateway router. This architecture prevents these devices from being accessed from outside the private local network to which they belong. Supplying an internet Connection to every device connected to the internet saturates the available number of addresses, due to the restricted size of these numbers. As a result, these devices interact via a single server with a public IP address, known as a proxy server, since its address is visible anywhere on the internet.

The use of proxy servers imposes security and performance issues to the IoT devices using it to interchange data. The security issues are imposed by the existence of a third-party in the communications between two devices, which is the proxy server. As the data is revealed and cached in the proxy server, before being sent to the destination host, any attacks on the proxyserver may compromise the confidentiality of these data, as well as, the ability to manipulatethem when an attacker gains unauthorized access to the proxy server. Moreover, as the proxyserver needs to handle all the data passing from one host to another, huge resources are required, especially the processing power and bandwidth. Overloading the proxy server by an enormous amount of data to be transferred may result in significant latency in the deliveryprocess. Furthermore, using a proxy server as a connection midway lengthens the path that data must travel to reach its target.

In this thsis, a strategy for allowing interaction between Internet of things is suggested. This method uses a server with a public IP address that all IoT devices connect to it in order to reveal their public socket information. When communications to a certain IoT device is requested, the server sends the socket information to the requesting device. The asking item uses this data to provide the tracking data. This approach is known as

porthole punching and relies on the UDP protocol to transfer the data from one host to another. As most of the IoT devices that require data exchange belong to the same local network, the local socket information is sent from the requested device to the server, which delivers it to the requesting device. This information is used to establish communications first, then, the public socket information is used, in case the private connection fails.

The proposed method is evaluated and compared to the existing proxy-based method, using different scenarios. Two possibilities employ the requesting device's public connection details to initiate connections, with and without the IEC protocol implementation in the application layer, while the other two scenarios use the private socket information to establish the connection. In addition to the obvious benefits of using the proposed method, which is the lower server resources and improved security, the results have shown significant by transmitting data using the suggested technique, the performance of the IoT device improves. Unlike the usage of a proxy server, the server's function in the proposed technique is limited to forwarding the desired device's port information to the requesting one. As a result, the resources used are not proportional to the quantity of the data being sent.Moreover, as the communications are established directly, data are not revealed to the serverand the route they travel through is minimized, which reduces the attack risks. In addition, the time and energy required from the IoT device have significantly dropped when the proposed method is used to communicate, instead of proxy servers, which are very importantto the IoT devices according to their very limited resources.

Encrypting data solutions will be deployed and tested in upcoming work to protect connections amongst IoT devices. Using the described approach and encrypting the information, the security can be significantly improved, as the encrypted data is not revealed to any third-party and the route between the communicated devices is minimized. However, the use of data encryption techniques may impact the performance of the IoT devices, according to the complexity of these techniques and the limited resources of available on the IoT devices.

# REFERENCES


[1] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 2019.

[2] Massive Internet of Things for Industrial Applications: Addressing Wireless IIoT Connectivity Challenges and Ecosystem Fragmentation", Ieeexplore.ieee.org, [Online]. Available: *https://ieeexplore.ieee.org/abstract/document/7883984.* [Accessed: 26- Aug] 2022.

[3] C. Carthern, W. Wilson, R. Bedwell, and N. Rivera, "The Network Layer with IP," in Cisco Networks, ed: Springer, pp. 49-68.2015.

[4] P. Srisuresh and K. Egevang, "Traditional IP network address translator (Traditional NAT)," 2070-1721, 2000.

[5] Massive Internet of Things for Industrial Applications: Addressing Wireless IIoT Connectivity Challenges and Ecosystem Fragmentation", *Ieeexplore.ieee.org*, 2022. [Online]. Available: *https://ieeexplore.ieee.org/abstract/document/7883984*. [Accessed: 26- Aug] 2022.

[6] K. R. Fall and W. R. Stevens, TCP/IP illustrated, volume 1: The protocols: addison-Wesley, 2011.

[7] M. Hassan and R. Jain, High performance TCP/IP networking: Prentice Hall Upper Saddle River, NJ, 2003.

[8] K. Egevang and P. Francis, "The IP network address translator (NAT)," 2070-1721, 1994.

[9] R. Penno, S. Perreault, M. Boucadair, S. Sivakumar, and K. Naito, "Updates to Network Address Translation (NAT) Behavioral Requirements," 2070-1721, 2016.

[10] A. Dunkels, J. Alonso, T. Voigt, H. Ritter, and J. Schiller, "Connecting wireless sensornets with TCP/IP networks," *in International Conference on Wired/Wireless Internet Communications*, pp. 143-152.2004.

[11] C. Raiciu, D. Niculescu, M. Bagnulo, and M. J. Handley, "Opportunistic mobility with multipath TCP," *in Proceedings of the sixth international workshop on MobiArch*, pp. 7-12.2011.

[12] V. Seničar, B. Jerman-Blažič, and T. Klobučar, "Privacy-enhancing technologies—approaches and development," *Computer Standards & Interfaces*, vol. 25, pp. 147-158, 2003.

[13] D. Fifield, N. Hardison, J. Ellithorpe, E. Stark, D. Boneh, R. Dingledine, et al., "Evading censorship with browser-based proxies," *in International Symposium on Privacy Enhancing Technologies Symposium*, pp. 239-258.2012.

[14] K. Jeong and R. Figueiredo, "Self-configuring software-defined overlay bypass for seamless inter-and intra-cloud virtual networking," in Proceedings of the 25th ACM International Symposium on High-Performance Parallel and Distributed Computing,pp. 153-164.2016.

[15] A. Poulovassilis, F. Xhafa, and T. O'Hagan, "Event-based awareness services for P2P groupware systems," Informatica, vol. 26, pp. 135-157, 2015.

[16] K. T. Nguyen, M. Laurent, and N. Oualha, "Survey on secure communication protocols for the Internet of Things," Ad Hoc Networks, vol. 32, pp. 17-31, 2015.

[17] I. Mahgoub and M. Ilyas, Sensor network protocols: CRC press, 2016.

[18] C. Pakanati, M. Padmavathamma, and N. R. Reddy, "Performance comparison of tcp, udp, and tfrc in wired networks," *in Computational Intelligence & Communication Technology (CICT)*, IEEE International Conference on, pp. 257-263.2015.

[19] A. Arcuri, G. Fraser, and J. P. Galeotti, "Generating TCP/UDP network data for automated unit test generation," *in Proceedings of the 10th Joint Meeting on Foundations of Software Engineering*, pp. 155-165.2015.

[20] H. Zimmermann, "OSI reference model--The ISO model of architecture for open systems interconnection," *IEEE Transactions on communications*, vol. 28, pp. 425-432, 1980.

[21] W. Ford, Computer Communications Security: principles, standard protocols, and techniques: PTR Prentice Hall, 1994.

[22] M. Cotton and L. Vegoda, "Special use ipv4 addresses," 2070-1721, 2010.

[23] J. C. Mogul and J. Postel, "*Internet standard subnetting procedure,"* 2070-1721, 1985.

*[24]* R. P. Draves, C. King, S. Venkatachary, and B. D. Zill, "Constructing optimal IP routing tables," *in INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*. IEEE, pp. 88-97.1999.

[25] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear, "Address allocation for private internets," 2070-1721, 1996.

[26] W. R. Stevens, B. Fenner, and A. M. Rudoff, UNIX Network Programming: The Sockets Networking API vol. 1: Addison-Wesley Professional, 2004.

[27] C. Trivedi, H. J. Trussell, A. A. Nilsson, and M.-Y. Chow, "Implicit traffic classification for service differentiation," *North Carolina State University. Center for Advanced Computing and* Communication2002.

[28] J. Li, D. Li, Y. Huang, Y. Cheng, and R. Ling, "Quick NAT: High performance NAT system on commodity platforms," *in Local and Metropolitan Area Networks (LANMAN), IEEE International Symposium on*, pp. 1-2.2017.

[29] A. Stephenson and D. Namiot, "On data transfer between mobile web clients,"
a. *International Journal of Open Information Technologies*, vol. 3, pp. 30-40, 2015.

[30] Y. Wang, T. T. Gamage, and C. H. Hauser, "Security implications of transport layer protocols in power grid synchrophasor data communication," *IEEE Transactions on Smart Grid*, vol. 7, pp. 807-816, 2016.

[31] S. Kota, M. Goyal, R. Goyal, and R. Jain, "Multimedia satellite networks and TCP/IP traffic transport," arXiv preprint arXiv:1603.08020, 2016.

[32] S. R. Pokhrel, M. Panda, H. L. Vu, and M. Mandjes, "TCP performance over Wi-Fi: Joint impact of buffer and channel losses," *IEEE Transactions on Mobile Computing, vol. 15, pp. 1279-1291, 2016.*

[33] J. Postel, "Transmission control protocol," 2070-1721, 1981.

[34] M. Allman, "On the generation and use of TCP acknowledgments," *ACM SIGCOMM Computer Communication Review*, vol. 28, pp. 4-21, 1998.

[35] J. Postel, "User datagram protocol," 2070-1721, 1980.

[36] I. Norwegian, "870-5-101 User Conventions, Revision no. 2.0," ed.

[37] J. Makhija and L. Subramanyan, "Comparison of protocols used in remote monitoring: DNP 3.0, IEC 870-5-101 & Modbus," *Electronics Systems Group, IIT Bombay, India*, Tech. Rep, 2003.

[38] D.-j. Kang and R. J. Robles, "Compartmentalization of protocols in SCADA communication," *International Journal of Advenced Science and Technology*, vol. 8, pp. 27-36, 2009.

[39] D. Guinard, V. Trifa, and E. Wilde, "A resource oriented architecture for the web of things," *in Internet of Things (IOT)*, pp. 1-8.2010.

[40] M. A. G. Maureira, D. Oldenhof, and L. Teernstra, "ThingSpeak–an API and Web Service for the Internet of Things," Retrieved7/11/15World WideWeb, http://www. Mediate chnology. *leiden. edu/images/uploads/docs/wt2014_* thing speak. pdf, 2011.

[41] F. Banaie, J. Misic, and V. B. Misic, "Priority-Based Caching Policy at a Hybrid IoT Proxy," *in IEEE International Conference on Communications (ICC)*, pp. 1-6.2018.

[42] F. Audet and C. Jennings, "Network address translation (NAT) behavioral requirements for unicast UDP," 2070-1721, 2007.

[43] B. Ford, P. Srisuresh, and D. Kegel, "Peer-to-Peer Communication Across Network Address Translators," in USENIX Annual Technical Conference, General Track, pp. 179-192.2005.

[44] S. N. Srirama and M. Liyanage, "Tcp hole punching approach to address devices in mobile networks," *in 2nd International Conference on Future Internet of Things and Cloud (FiCloud)*, pp. 90-97.2014.

[45] S. W. Cho, "P2P-based Mobile Social Networks," in P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), *10th International Conference on*, pp. 141-145. 2015.

[46] D. Seah, W. K. Leong, Q. Yang, B. Leong, and A. Razeen, "Peer NAT proxies for peer-to-peer games," in *Proceedings of the 8th Annual Workshop on Network and Systems Support for Games*, p. 6.2009.

[47] H. Herry, E. Band, C. Perkins, and J. Singer, "Peer-to-Peer Secure Updates for Heterogeneous Edge Devices," 2018.

[48] C. Stergiou, K. E. Psannis, B.-G. Kim, and B. Gupta, "Secure integration of IoT and cloud computing," *Future Generation Computer Systems*, vol. 78, pp. 964-975, 2018.

[49] S. Pasha, "ThingSpeak based sensing and monitoring system for IoT with Matlab Analysis," *International Journal of New Technology and Research (IJNTR)*, vol. 2, pp. 19-23, 2016.

[50] K. Bharadwaj, R. Dhawan, M. K. Ray, and P. Mahalakshmi, "Wi-Fi-Based Low-Cost Monitoring of ECG and Temperature Parameters Using Arduino and ThingSpeak," in *Advances in Systems, Control and Automation*, ed: Springer, 2018, pp. 637-646.2018.

[51] R. Chandana, S. Jilani, and S. Javeed Hussain, "Smart surveillance system using thing speak and Raspberry Pi," *International Journal of Advanced Research in Computer and Communication Engineering*, vol.4,pp.2.