

SMARTNET

by

Mohith Kune 411641

Sunandhini Medi 411648

Bishanth Thota 411672

Under the esteemed guidance of

Dr. Gowtham Reddy



DEPARTMENT OF COMPUTER SCIENCE

NATIONAL INSTITUTE OF TECHNOLOGY, ANDHRA PRADESH

TADEPALLIGUDEM - 534102, INDIA

MAY 2020

SMARTNET

Thesis submitted to
National Institute of Technology Andhra Pradesh
for the award of the degree
of
Bachelor of Technology
by

Mohith Kune *411641*

Sunandhini Medi *411648*

Bishanth Thota *411672*

Under the esteemed guidance of

Dr Gowtham Reddy



DEPARTMENT OF COMPUTER SCIENCE

NATIONAL INSTITUTE OF TECHNOLOGY, ANDHRA PRADESH

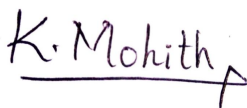
TADEPALLIGUEDEM - 534102, INDIA

MAY 2020

© 2018. All rights reserved to NIT Andhra Pradesh

DECLARATION

I declare that this written submission represents my ideas in my own words and where other's ideas or words have been included. I have adequately cited and referenced the sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.



Mohith Kune

411641

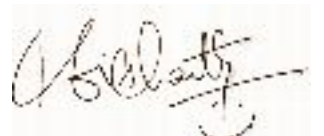
Date:



Sunandhini Medi

411648

Date:



Bishanth Thota

411672

Date:

CERTIFICATE

It is certified that the work contained in the thesis titled “**SMARTNET**,” by “**Mohith Kune**, bearing Roll No: 411641”, “**Sunandhini Medi**, bearing Roll No: 411648”, “**Bishanth Thota**, bearing Roll No: 411672” has been carried out under my/our supervision and that this work has not been submitted elsewhere for a degree*

Dr. Gowtham Reddy Alavalapati

Dept. of Computer Science and Engineering

N.I.T. Andhra Pradesh

May 2020

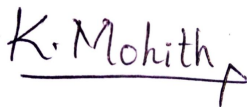
*Note: this statement is mandatory

ACKNOWLEDGEMENTS

The success and outcome of this project required a lot of guidance and assistance from many people, and we are privileged to have got this all along with the completion of my project. Everything we have done is because of such guidance and help, and we will never hesitate to thank them.

We owe my sincere gratitude to our project guide *Dr. A. Goutham Reddy*, Assistant Professor, Department of Computer Science, National Institute of Technology, Andhra Pradesh, who took keen interest and guided us all along, till the completion of our project work by providing all the necessary information.

We are grateful and lucky enough to receive consistent motivation, support, and guidance from all the staff of the Computer Science Department who have helped us to complete our project work successfully. We would also like to extend our sincere gratitude to all the teaching and non-teaching staff including that helped us in carrying out this project.



Mohith Kune

411641

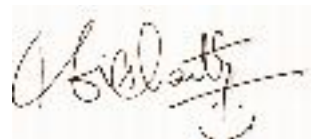
Date:



Sunandhini Medi

411648

Date:



Bishanth Thota

411672

Date:

LIST OF FIGURES

Sno	Title	Page no.
1	P2P Communication with the help of the Server	13
2	Running WebRTC server	14
3	Accessing the WebRTC server	14
4	Connecting to the WebRTC server through Desktop browser	15
5	All clients available on the network	15
6	Mobile version of the client accessing the WebRTC server	16
7	Communicating with another client	17
8	Mobile version communicating with Desktop client	18
9	Local Cache Server in LAN	20
10	Squid proxy server configuration directory	21
11	Examining the squid configuration file - showing the ACL & Authentication	21
12	Squid conf file showing the proxy server's cache configuration	22
13	Proxy server asking for authentication credentials on desktop firefox browser	22
14	Proxy server asking for authentication credentials on mobile chrome browser	23

15	Squid server access log when accessing IIIYear.pdf file from nit andhra website for the first time using desktop firefox browser.	25
16	Squid server access log when accessing IIIYear.pdf file from nit andhra website for the second time using mobile chrome browser.	25
17	Framerates for iOS, Android and Chrome	27
18	Relative performance without and with the proxy server	28

LIST OF TABLES

Table	Title	Page no.
1	Comparison between the regular proxy server and our proxy server	20
2	Squid server cache codes	24
3	Various protocols used in popular platforms	26
4	Proposed Protocols	26
5	WebRTC supported features across Android, iOS and Web	27

NOTATIONS

Notation	Meaning
VOIP	Voice Over Internet Protocol
LAN	Local Area Network
IT	Information Technology
ACL	Access Control List
P2P	Peer to Peer
WebRTC	Web Real-Time Communication
SDP	Session Description Protocol
HMAC	Hash-based Message Authentication Code
SHA-256	Secure Hash Algorithm
AES-256	Advanced Encryption Standard
OAuth2	Open Authorization
DES	Data Encryption Standard

ABSTRACT

The Internet has become the centre of everything nowadays. People around the world can communicate through the internet, thanks to VOIP [\[1\]](#) which involves low costs compared to telecommunications. However, when the communicating parties are on the same LAN, the cost of communication can be further optimized by leveraging this concept without the need of the internet. Unlike other existing solutions which use specific hardware (like VOIP phones), our project efficiently achieves this by allowing communications through the effective path using intranet infrastructure and without the use of any special hardware or third-party software like WhatsApp, zoom etc.

Institutes often opt for high bandwidth internet services intending to serve numerous users who can also happen to habitually browse or download the same content on the network. The existing infrastructure incurs huge internet bills and it can be made better to make use of bandwidth efficiently and reduce internet costs. Our project mitigates it by setting up a web proxy cache server on the LAN, which caches the most used files and serves effectively without any delay.

TABLE OF CONTENTS

<i>Content</i>	<i>Page No</i>
Title	i
Declaration	ii
Certificate	iii
Acknowledgements	iv
List of Figures	v
List of Tables	vi
Notations	vii
Abstract	viii
Table of Contents	xi

CONTENTS

1. INTRODUCTION	1
1.1. Introduction	1
1.2. Terms	2
1.2.1. VOIP	2
1.2.2. LAN	2
1.2.3. ACL	2
1.2.4. P2P	2
1.2.5. WebRTC	2
1.2.6. Nginx	3
1.2.7. SDP	3
1.2.8. SHA-1 & MD-5	3
1.2.9. Squid	3
2. PROBLEM STATEMENT	4
2.1. Problem Statement	4
2.2. Objectives	5
2.3. Expected Outcomes	5
2.3.1. Communication	5
2.3.2. Security	5
2.3.3. Cost	5
3. LITERATURE REVIEW	6
3.1. Literature Review	6
3.1.1. Voice over IP Mobile Telephony Using Wi-Fi P2P	6
3.1.2. Web Real-Time Communication (WebRTC) Technology	6
3.1.3. A Videoconferencing System Based on WebRTC Technology	6
3.1.4. DistCache: Provable Load Balancing for Large-Scale Storage Systems with Distributed Caching	6
3.1.5. Web Load Balance and Cache Optimization Design Based Nginx under High-Concurrency Environment	7
3.1.6. Web Server Performance of Apache and Nginx	7
3.1.7. Caching in Base Station with Recommendation via Q-Learning	7
3.1.8. An Improved Authenticated Group Key Transfer Protocol Based on Secret Sharing	7
3.1.9. Web Load Balance and Cache Optimization Design based Nginx Under Highconcurrency Environment	8

3.1.10. Squid Proxy Server A Practical Approach	8
4. PROPOSED APPROACH	9
4.1. Module 1	9
4.1.1. Aim	9
4.1.2. Approach	9
4.1.3. Outcomes	9
4.2. Module 2	10
4.2.1. Aim	10
4.2.2. Approach	10
4.2.3. Outcomes	10
4.3. Module 3	11
4.3.1. Aim	11
4.3.2. Approach	11
Module 1	11
Module 2	11
4.3.3. Outcomes	11
5. EXPERIMENTAL PROCEDURE	12
5.1. MODULE-1	12
5.1.1. Procedure	12
5.1.2. Implementation	14
5.2. MODULE-2	19
5.2.1. Procedure	19
5.2.1. Implementation	21
5.2.2. Squid cache codes	23
5.3. MODULE-3	26
5.3.1. Procedure	26
6. RESULTS AND DISCUSSIONS	27
6.1. Module 1	27
6.2. Module 2	28
7. SECURITY ANALYSIS	29
7.1. Module 1	29
7.1.1. Accessing media resources	29
7.1.2. SRTP weakness	29
7.1.3. Signalling protocol	30
7.1.3.1. MITM	30

7.1.3.2. Replay attack	30
7.1.3.3. Session hijacking	30
7.2. Module 2	31
7.2.1 Port number	31
7.2.2 Maximum cache object size	31
7.2.3 DNS poisoning	32
7.2.4 Authentication	32
7.2.5 Request header	32
7.2.6. Client lifetime	33
7.2.7. Logging	33
8. CONCLUSIONS AND FUTURE SCOPE	34
REFERENCES	35

1. INTRODUCTION

1.1. Introduction

It has been a dream to offer communication technology at an affordable cost for everyone in the world. The charges of early-stage voice communications were directly proportional to the distance between connecting parties. The proliferation of computing and network technologies have led the researchers to develop protocols that enable the users to communicate through the internet at low cost irrespective of their geographical locations, as the transfer of data packets is relatively cheaper than voice calls. This technology works even if the telephone lines go offline. This fundamental property has steered us towards a project that aims at providing communication services to the users of the same network without external means and lease line connections. The strategic benefit of this idea is that different divisions of an institute functioning through several buildings can communicate by just being on the same network without the internet lease line. This project intends to minimize the cost and provide more features in addition to the calls by converting the computers as phones through software.

A great worry often institutes evident is growing communication costs. One of the reasons for this is intranet formation requires a special hardware setup along with a leased line, and another reason is opting high bandwidth internet services with an aim to serve numerous users who can also happen to habitually browse or download the same content. This practice is being perceived as a significant problem for which several firms have started to Figure out the solutions, which includes some adaptations to the network in order to regulate the bills. One of such solutions is to cache the data files that are frequently accessed. This project targets at extending this caching system with the feature users tracking across the LAN and reliable communication. In today's world, privacy and security have become a leading concern. This project has features like authentication and robust data sharing by deploying efficient cryptographic techniques for maximum security.

1.2. Terms

This section discusses the most important terms used throughout this report.

1.2.1. VOIP

Voice over Internet Protocol is a new technology that is developed to overcome the drawbacks of traditional tele-communication technology. It helps people use internet services as a medium for communication. It transfers the audio and video from one place to another in the form of packets using internet protocol(IP) rather than by traditional circuit transmissions of the PSTN.

1.2.2. LAN

A local area network (LAN) is a computer network within a limited range. It is used mostly in houses, schools, laboratories, universities, etc. The range of the LAN depends on many factors, such as access points, obstacles etc.

1.2.3. ACL

ACL is a set of permissions attached to an entity. it specifies the extent of access and limitations on a resource for the users or system processes. For example, ACL is used for limiting certain users accessing some websites on a network. ACL is also used to separate users into different groups.

1.2.4. P2P

Peer-to-peer (P2P) technology is one of the best inventions in Computer Science. In the P2P network, there is no centralized server serving the requests. The best example of P2P networking is torrents. All peers are equally important and the data is distributed among the peers. In this project, we use this P2P technology for communication between two users.

1.2.5. WebRTC

WebRTC (Web Real-Time Communication) is a free, open-source project by Google. It helps web browsers to achieve real-time communication (RTC) easily by using simple application programming interfaces (APIs) and without the need of any third-party plugins or

tools. As of today, all browsers support WebRTC. It allows audio and video communication to work inside web pages by allowing direct peer-to-peer communication.

1.2.6. Nginx

Nginx is a web server which can also be used as a reverse proxy, load balancer, mail proxy and HTTP cache. It accelerates content and application delivery, improves security, facilitates availability and scalability for the busiest websites on the Internet.

1.2.7. SDP

The Session Description Protocol (SDP) is used for establishing multimedia communication sessions. It is used for the session announcement and session invitation. Its mainly used for streaming media applications, like VoIP and video conferencing. SDP does not deliver any media streams itself but is used between endpoints for negotiation of network metrics, media types, and other associated properties.

1.2.8. SHA-1 & MD-5

SHA-1 (Secure Hash Algorithm 1) and MD5 are cryptographic hash function algorithms that give message digest from a given input. These functions are popularly used for storing hashes of the passwords, during authentication, verification(checksums), etc.

1.2.9. Squid

Squid is a proxy server capable of forwarding and caching requests. It is used for increasing the response time, caching repeated requests. It is so powerful that it can also be used as a resource for sharing files across the network. It is mainly used for serving HTTP and FTP requests. It includes limited support for several other protocols like SSL, TLS and HTTPS.

2. PROBLEM STATEMENT

2.1. Problem Statement

There have been numerous everyday circumstances that have driven us towards this project. Among the various circumstances, the most popular ones are as below.

1. Assume you are an employee/student of an institution working in a particular division and want to reach out to the IT team that is on another floor/block for some help on a technical issue. You may call/message them using external means like telephone, WhatsApp and several others. Since both are on the same network, we can leverage this concept to provide communication free of cost.
2. A faculty may want to have reservations on communicating with others concerning timings and designations. For instance, he might want to accept the calls from/to his colleagues throughout the working hours a day, whereas he may want to accept the calls from his students only during a particular session of the day. Henceforth, this project proposes to use the access control list (ACL) mechanism to restrict communications.
3. Imagine when all of your colleagues have to download the same files to work. It is often a waste of bandwidth and internet charges to route every request until the respective server. As well, several users of the same network often download the same content knowingly/unknowingly. For instance, on a university network, students may download a Netflix/Amazon TV show and as an admin of the network, you can know what a particular student is viewing or downloading what content. This project provides a means for monitoring users on a network and a local caching mechanism to intelligently download the content which could cut down the costs drastically.

2.2. Objectives

1. Communication among the users of the same network without external means like the internet and telephones.
2. Lessen the internet bills by effective usage with the help of the local cache server.
3. Secure the communications by deploying suitable cryptographic practices

2.3. Expected Outcomes

2.3.1. Communication

1. The users of the network will be able to make voice calls by logging in to their accounts at the local server.
2. Users can also be able to video chat.

2.3.2. Security

1. The Network administrator is able to monitor students' activities across the network.
2. Secure communication between any two systems in the Network

2.3.3. Cost

1. The local cache server can store the once downloaded file, and when the same request appears, it redirects the client to download from its local storage.
2. The same server also maintains a cache that helps to re-resolve the requests that are already once resolved in the past, which may thereby reduce the internet charges.

3. LITERATURE REVIEW

3.1. Literature Review

3.1.1. Voice over IP Mobile Telephony Using Wi-Fi P2P

This paper proposes an algorithm to use IP addresses as mobile numbers for contacting other mobile devices using P2P technology. This paper also proposes a way to provides voice calls and SMS services on LAN and allow devices to communicate with one another without any cost through P2P technology.[\[1\]](#)

3.1.2. Web Real-Time Communication (WebRTC) Technology

This paper describes the WebRTC technology and its client-server implementation and signalling systems for P2P audio, video and data communications. WebRTC supports real-time communication without using any third-party software or plugins[\[2\]](#). In this project, we use WebRTC technology for achieving our goals.

3.1.3. A Videoconferencing System Based on WebRTC Technology

This paper analyzes the impact of WebRTC technology on the CPU and memory requirements for different software and hardware configurations of end-point devices. It also compares the technology with some other existing rivals.[\[3\]](#)

3.1.4. DistCache: Provable Load Balancing for Large-Scale Storage Systems with Distributed Caching

This paper proposes a new mechanism for distributed caching for providing load balancing for large-scale storage systems. DistCache co-designs cache allocation with cache

topology and query routing. This paper shows different ways for implementing the load balancing[\[4\]](#).

3.1.5. Web Load Balance and Cache Optimization Design Based Nginx under High-Concurrency Environment

This paper shows detailed analysis on Nginx and proves that Nginx based solutions are efficient and highly suitable for load balancing, and efficient deployment in a highly concurrent environment by testing on a virtual BBS application server cluster environment.[\[5\]](#). In this project, we recommend Nginx for deploying the software into production.

3.1.6. Web Server Performance of Apache and Nginx

This paper shows the systematic literature review comparing different web servers like Apache, IIS, Nginx and Lighttpd and several more. It shows the results of different web servers in terms of response time, CPU utilization and memory usage. It concludes that both Apache and Nginx are powerful, flexible, and capable. Nevertheless, Nginx performed better than Apache based on major performance measures that include response time, CPU utilization and Memory usage.[\[6\]](#)

3.1.7. Caching in Base Station with Recommendation via Q-Learning

This paper leverages the human-behaviour related information for boosting the system throughput and improving user experience with reduced cost and optimal cache replacement policy. This paper recommends a policy for higher long-term system reward than existing policies without recommendation.[\[7\]](#)

3.1.8. An Improved Authenticated Group Key Transfer Protocol Based on Secret Sharing

This paper proposes an improved authenticated key transfer protocol based on Shamir's secret sharing algorithm. The proposed approach achieves key confidentiality and provides key

authentication by broadcasting a single authentication message to all members. Furthermore, the scheme proposed in this paper resists against both insider and outsider attacks.[\[8\]](#)

3.1.9. Web Load Balance and Cache Optimization Design based Nginx Under Highconcurrency Environment

In this paper, a virtual Discuz BBS application server cluster environment was designed according to the topology in a real network, together with a set of corresponding load balancing and Web caching optimization solutions are designed and simulated. This paper proves that solutions based on Nginx for are not only feasible but also very suitable and efficient for deployment in a highly concurrent environment. [\[10\]](#)

3.1.10. Squid Proxy Server A Practical Approach

Proxy Networks are useful where there is a shortage of connection, but a need to share a connection with multiple personal computers. They help to speed up the internet surfing, it provides a mechanism that enables to hide the IP-Address of the client pc so it can surf anonymously. The squid proxy also provides secure control over the network and bandwidth sharing [\[9\]](#).

4. PROPOSED APPROACH

As discussed in the previous chapter the project is solely divided into three modules. In this chapter, we will briefly discuss the project modules and the proposed approach. In the next chapters, we will see the detailed experimental approach for each module. The project's prerequisite is that all users must be in the same network.

4.1. Module 1

4.1.1. *Aim*

Identification and communication among the users of the same network without external means like the Internet and Telephones.

4.1.2. *Approach*

Consider an organization network to which many users are connected. To communicate with other people, one has to login to the server that is also on the same network. This server keeps track of all logged-in users and their respective IP addresses along with the access control list (ACL) with it. The user interacts through a web browser and requests the server for communicating with the other user by providing his details using the session description protocol (SDP). The server then sends the message of the first user to the receiver who in turn provides the server with his details using SDP. The server then sends the receiver's description to the first user. Both users now are fully aware of their presence on the network; they can now communicate without the server via peer to peer (p2p) communication. We use modified web real-time communication (WebRTC) protocol[\[2\]](#) to achieve our goals that may also include ACL.

4.1.3. *Outcomes*

1. A user need not remember anything except the credentials.
2. ACL takes care of unimportant communications for a user.
3. The protocol and the software hide all the complex details.
4. A user can use either mobile or computer in the network, however, login is mandatory.

5. All the logins and communications are protected from eavesdroppers using encryption techniques. This will be covered more in the *third module*.
6. A login session is invalidated when the user leaves the software.

4.2. Module 2

4.2.1. Aim

Reduction of the internet bill by effective usage of the internet with the help of the local cache server.

4.2.2. Approach

The same server used in the previous module acts as a proxy server in this module. All connections to the internet are routed through this server, which also does other functions apart from caching. It maintains policies like domain whitelist, caching policies and several more, which can be conFIGured anytime. It doesn't cache files that are not from whitelisted domains, and also stores the activities of users like browsing history, downloaded files and so on; so that the characteristics of a student in a university can be predicted for official purposes. The proxy server can, in turn, have multiple slave nodes that can handle connections from thousands of users (scaling).

4.2.3. Outcomes

1. Bandwidth usage is reduced and Internet charges are controlled.
2. Users can be served in no time provided sufficient cache storage at the Server.
3. We can also monitor users based on authentication information from module-1
4. We can filter content that can be viewed by a user, depending on the role. For example, a student can be given restricted access to websites like YouTube, Vimeo, etc.

4.3. Module 3

4.3.1. Aim

To secure the communications between any two parties by deploying suitable cryptographic techniques.

4.3.2. Approach

The communication between any two peers must be protected. We mean to use a lightweight authenticated key agreement protocol using existing standard cryptographic methods to secure the channel between them. Throughout the above modules, we have identified different scenarios where security should be exclusively provided to the end-users.

The various scenarios are:

1. Module 1

- a. The database at the server which has sensitive information of clients.
- b. The communication between the client and the server during
 - i. Authentication process
 - ii. Signalling other peers
- c. P2P communication between two Clients.

2. Module 2

- a. The client sessions with the proxy server.
- b. The data transfer from the cache storage at the proxy server to the client.

4.3.3. Outcomes

1. The data flown between any two peers is encrypted.
2. The response time the server may get increased due to repeated use of encryption techniques.

5. EXPERIMENTAL PROCEDURE

5.1. MODULE-1

5.1.1. Procedure

Consider a scenario where two clients, say client-1, client-2 are on the same network. When the client-1 wants to communicate with the client-2, the client-1 has to know the path to client-2. Somehow the client-1 has to signal the client-2 that he wants to initiate a call. However both the clients don't know how to reach one another. We need to provide a mechanism to signal the clients. Due to this reason, we shall install a server on the same network that is known to every client on the network which has many functionalities which we will see throughout this report. In order to send call signals, client-1 now uses this server as a signalling mechanism. However, client-1 has to authenticate himself with the server before using its signalling mechanism to send or get calls from others. Every client has to remain logged in order to send or receive calls. This server maintains a database in which all the information of its clients like their usernames, IP addresses are stored as part of the authentication mechanism so that clients necessarily don't know the IP address of each other to establish communication.

The client-1 has to initiate a session with client-2 where both can stream other's video and audio. With the server being active, the client-1 can retrieve all the active usernames on the network. The client-1 chooses to call client-2 by creating an offer through SDP and forwards the offer object to the server hoping to get transferred to the client-2. The offer object includes information about MediaStreamTracks, codec, and other options supported by the client's browser. The Server then forwards the offer of client-1 to the client-2. After receiving the offer the client-2 may choose to accept or reject the offer. Let's say client-2 wants to accept the offer from client-1. The client-2 does so by generating an answer object (through the SDP) and forwarding it to the server. The server, in turn, forwards the answer of client-2 to client-1. As the two clients have exchanged offer-answer they know each other's details necessary to establish p2p communication. This project uses custom features in addition to WebRTC technology [\[3\]](#) to provide this p2p communication between two client browsers.

On a university level, for example, a student can know the username of a faculty and can call at inappropriate times. Further a faculty may choose to answer student calls in a particular session of a day. In order to address these issues we use ACL which can segregate users on a network into different Pools with a priority, and it works this way:

1. A person cannot establish communication with other people in a pool having higher priority.
2. A person can establish communication only with the people in the same pool and with the people in the pools that have lower priority than his pool.
3. A person may choose to be available or unavailable for a period of time.

A client can have only one active session at a time. Multiple instances of an account can be suspended to prevent identity theft. When a user tries to authenticate from a device, the server accepts the connection only if the user has no active logins. A client may choose to close the browser or forgets to logout. However, the server automatically logs out the user and changes the status to *inactive* so that other clients can be aware of his status. This makes room for other people to use the same device in the near future.

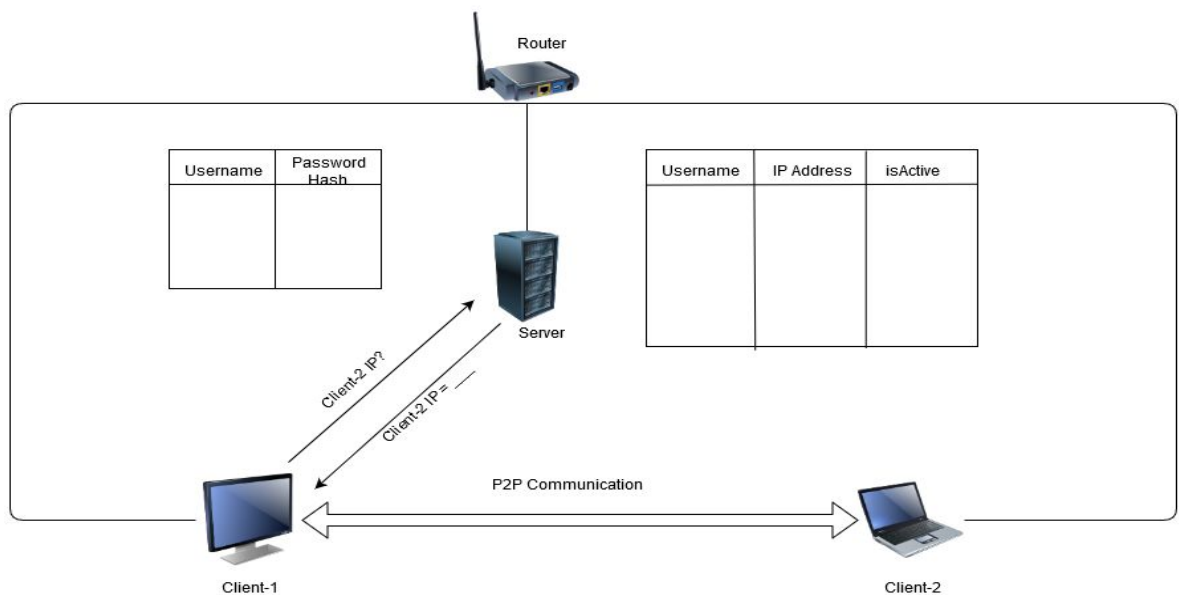


Figure 1: P2P Communication with the help of the Server

5.1.2. Implementation

We have developed two types of software for this module, for the server and the client. The server runs the WebRTC signalling server code. And the client can access the services by using the mobile application or the client can also use the web browser. We used Debian/Ubuntu on the server and the WebRTC implementation is done in Flutter

```
server@server-VirtualBox:~$ cd SMARTNET-server/
server@server-VirtualBox:~/SMARTNET-server$ ls
bin  configs  web
server@server-VirtualBox:~/SMARTNET-server$ ./bin/server-linux-amd64
2020-05-24T09:58:42+05:30 INF Flutter WebRTC Server listening on: 0.0.0.0:8086
2020/05/24 09:59:19 http: TLS handshake error from 192.168.1.134:44638: remote error: tls: unknown certificate
```

Figure-2: running WebRTC server

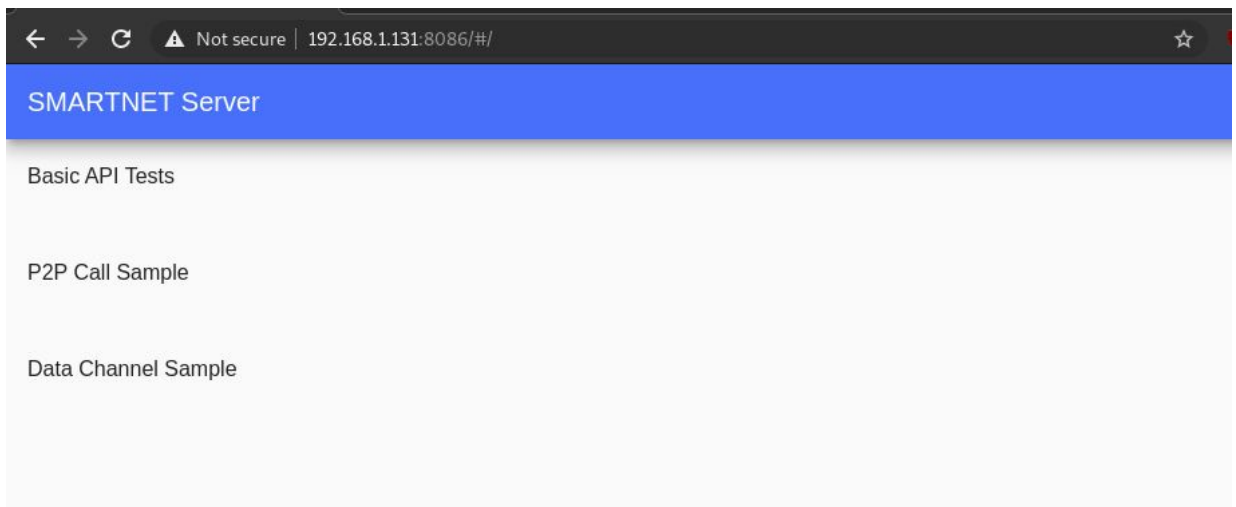


Figure-3: accessing the WebRTC server

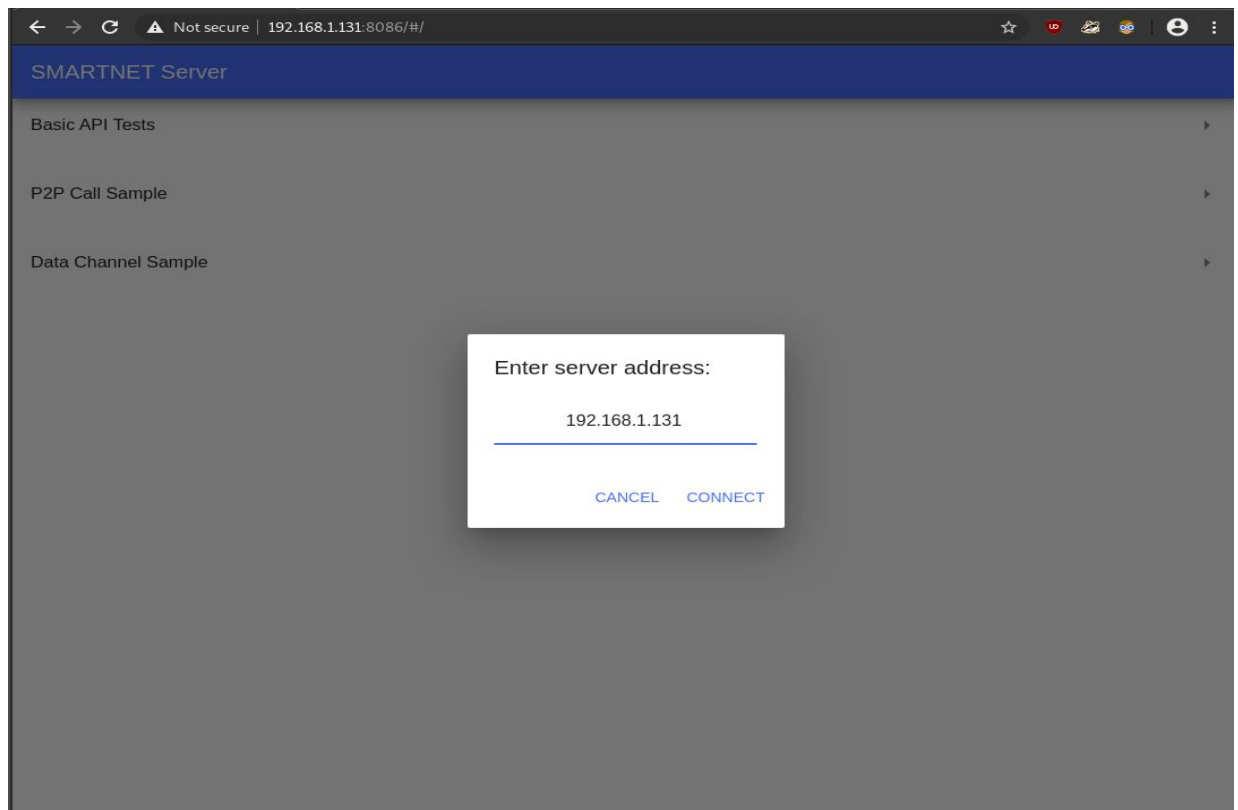


Figure-4: connecting to the WebRTC server through Desktop browser



Figure-5: all clients available on the network

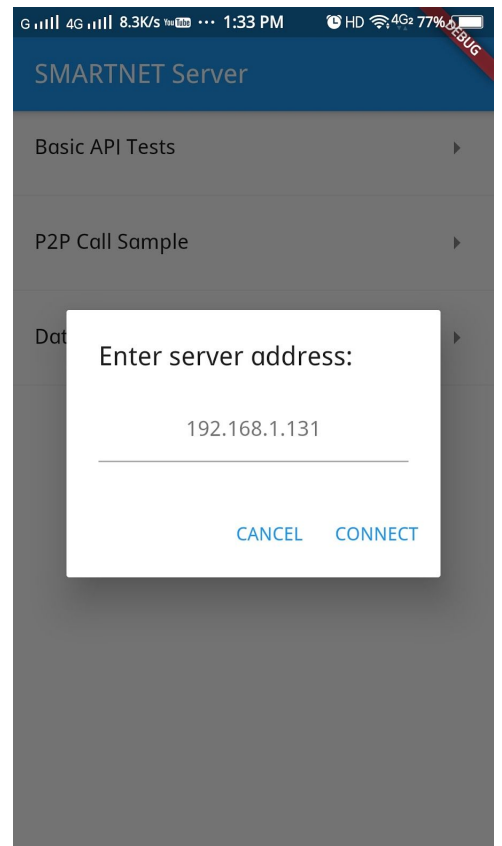
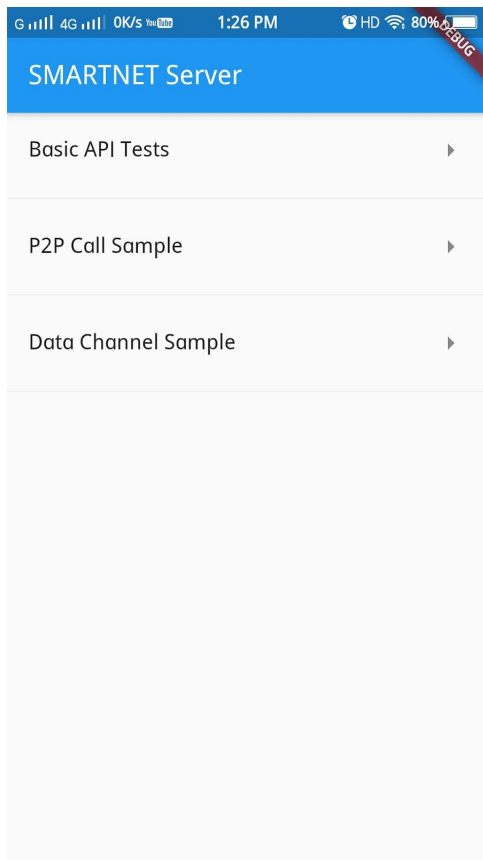


Figure-6: The mobile version of the client accessing the WebRTC server

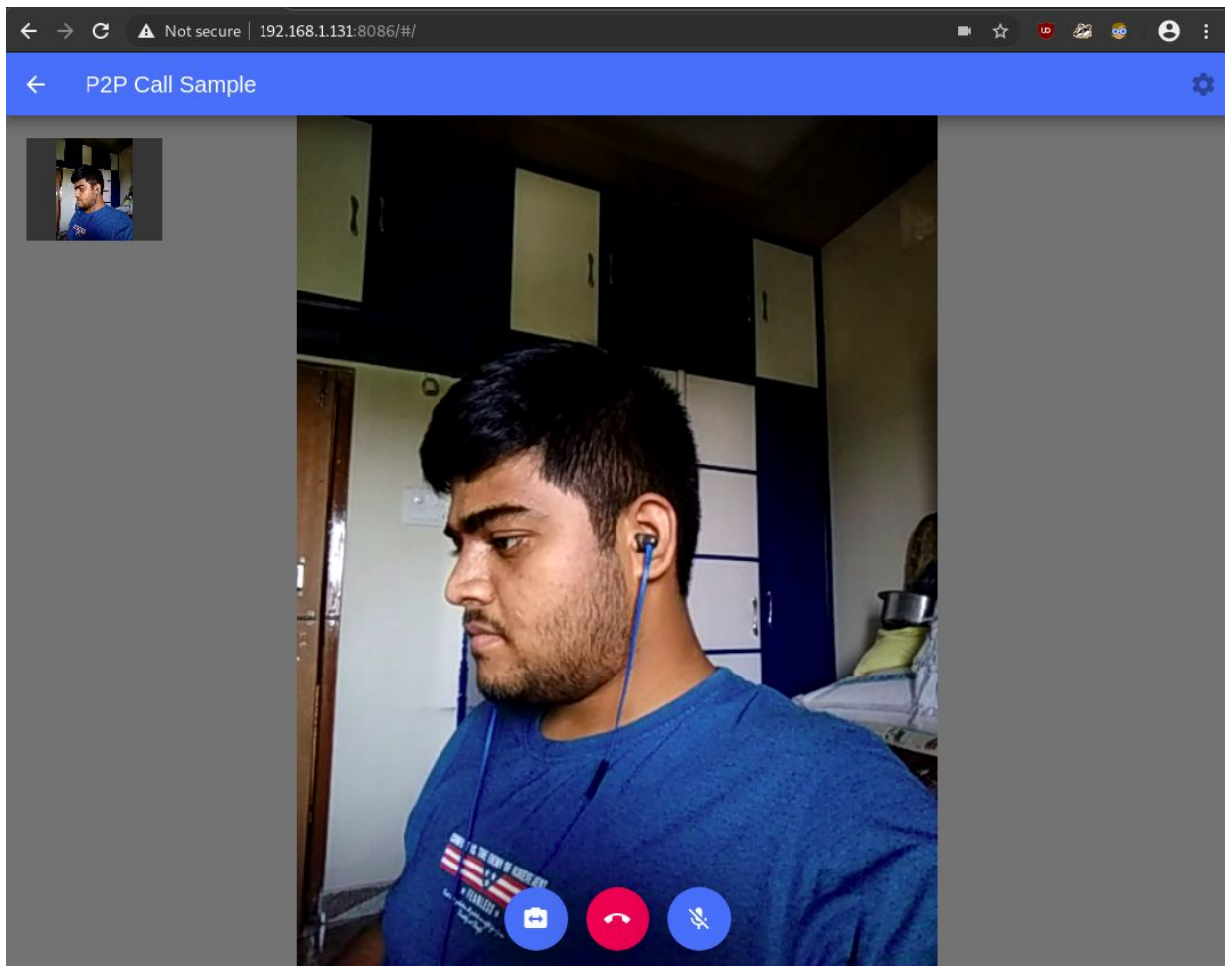


Figure-7: The desktop version communicating with the mobile client

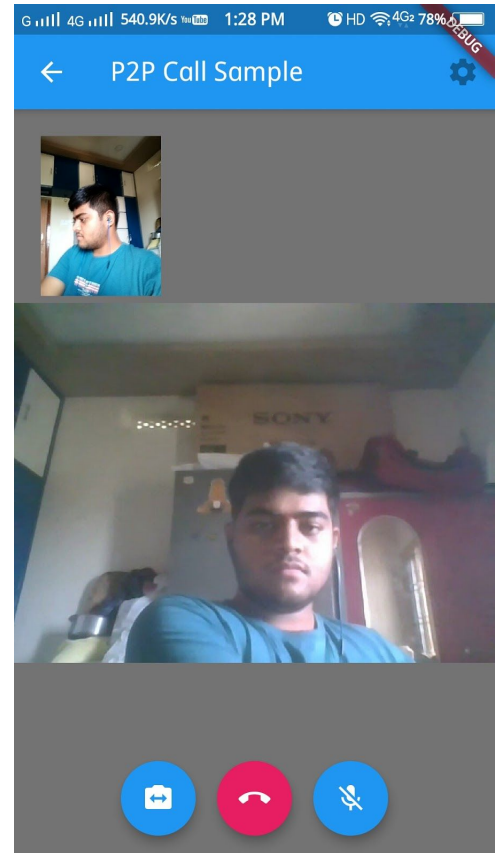
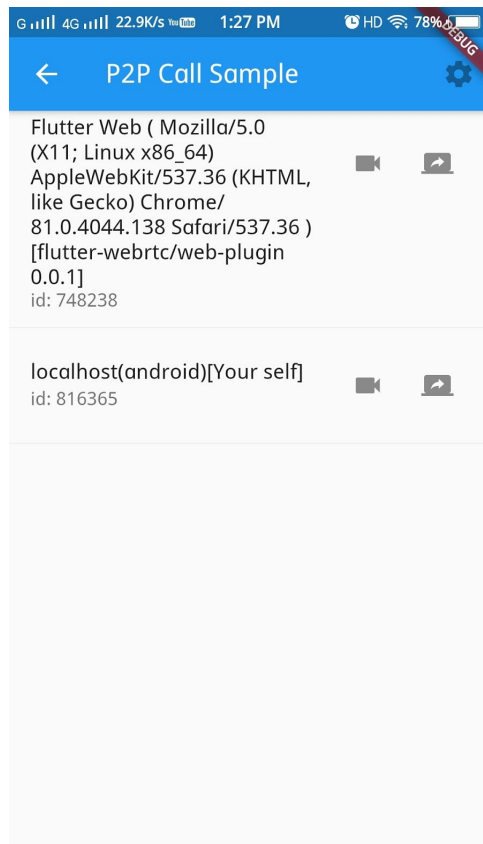


Figure-8: The mobile version communicating with Desktop client

5.2. MODULE-2

5.2.1. Procedure

First, we present our primary way of accomplishing this module later we will propose a way to scale it. Consider a scenario where the clients connected in a network. All connections from the clients to the internet pass through the gateway router in the network. We also connect the same server from the first module. This server now acts as a proxy server and every connection is now routed from it. In addition to being a proxy server and masking the user from the internet, it acts as a local cache server. Throughout this report, we use terms, the proxy server and the local cache server interchangeably to mention the server we use in this project.

Whenever a client wants to access a website the connection is routed through the proxy server, which in turn requests on behalf of the client. And while responding to the client the proxy server caches static, media files such as images, videos, javascript, CSS, pdf files, etc. So that when other clients request the same website the proxy server intelligently serves some of the files from cache storage, instead of routing to the internet. This makes it an incredibly elegant solution to reducing costs. In addition to this, the proxy server also maintains the record of every user and their browsing activity, downloaded files by leveraging the authentication mechanism from module-1.

To brief things the functions of our Proxy server are as below:

1. To cache the media, static files like Html, javascript, CSS, pdf, pictures, etc based on policies.
2. Ties the browsing activity of a user to his account on the server.

The proxy server uses a set of configurable policies while caching the data from the internet. User sensitive information like credentials, personal files from google and other domains, etc are ignored. It also uses a policy for whitelisting domains and can be controlled by the network admins. In addition, it also has policies for invalidating cache based on properties like time, the size of the cached file, etc. This makes it a smarter move to keep the cache updated.

Functions	Regular proxy server	Our Proxy server
Content filtering	yes	yes
Data caching	yes	yes
Specific user monitoring	no	yes
Located on the same network	no	yes

Table-1: Local Cache Server in LAN

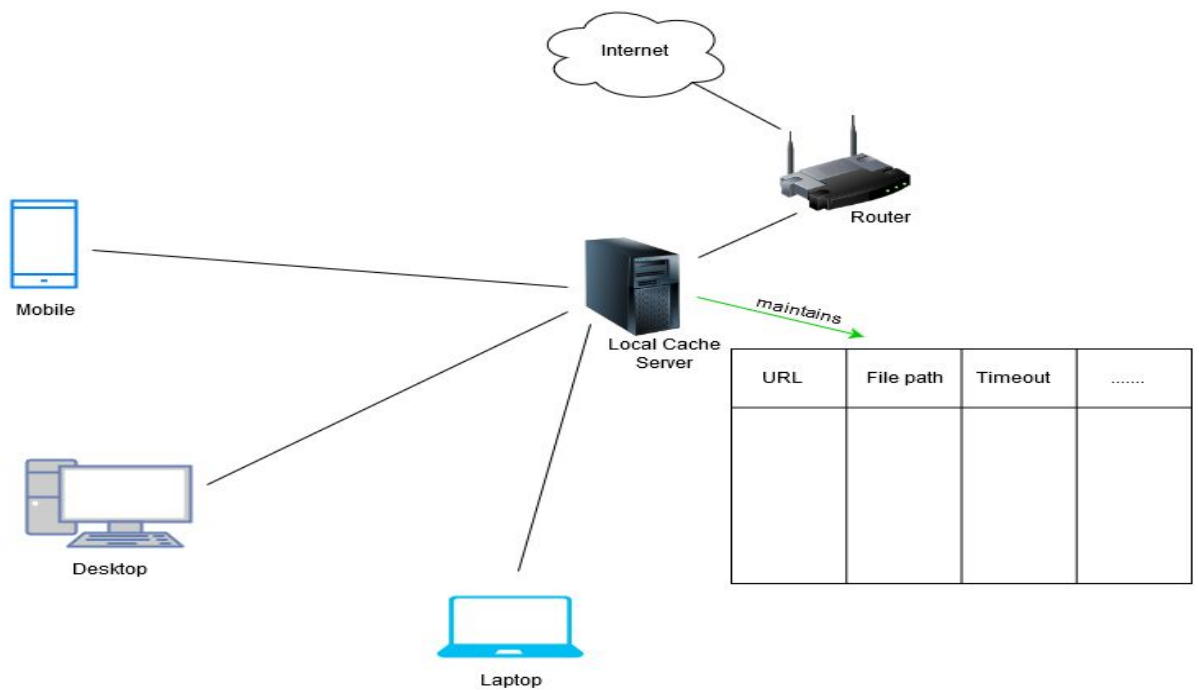


Figure-9: Local Cache Server in LAN

At a university level, however, due to the increase in the number of connections the proxy server may not achieve the desired performance levels. We deploy multiple servers(slave nodes) to serve many clients which are in turn controlled by a master node. This incredibly boosts the performance of the proxy server as the load is now distributed among multiple slave nodes. Each slave node is just a clone of the proxy server itself. This makes the proxy server to become more reliable than in the former case. We propose to use squid [9] server to choose the Nginx web server for load balancing [6] due to its advantages over the Apache web server.

5.2.1. Implementation

In this module, we have set up a proxy server which is capable of authenticating the users and caching the files. We have used a squid server in order to achieve our goals. Below are the screenshots of the results.

```
root@server-VirtualBox:/etc/squid# ls
blocked_sites.acl  errorpage.css  passwd  squid.conf
```

Figure-10: Squid proxy server configuration directory

```
GNU nano 2.9.3 /etc/squid/squid.conf
acl blocked_url dstdomain "/etc/squid/blocked_sites.acl"
http_access deny blocked_url

# acl whitelist dstdomain .google.com
# http_access allow whitelist

auth_param basic program /usr/lib/squid/basic_db_auth \
    --user server --password 123456 --plaintext --persist
# auth_param basic program /usr/lib/squid/basic_ncsa_auth /etc/squid/passwd
# auth_param basic credentialsttl 30 minutes
# auth_param basic casesensitive on
auth_param basic realm SMARTNET project - Authenticate yourself!
acl squid_users proxy_auth REQUIRED
http_access allow squid_users
```

Figure-11: examining the squid configuration file - showing ACL & Authentication

```
# cache_dir scheme directory size[MB] L1 L2 [options]
cache_dir ufs /var/spool/squid 1000 16 256
maximum_object_size 100 MB
```

Figure-12: The squid conf file showing the proxy server's cache configuration

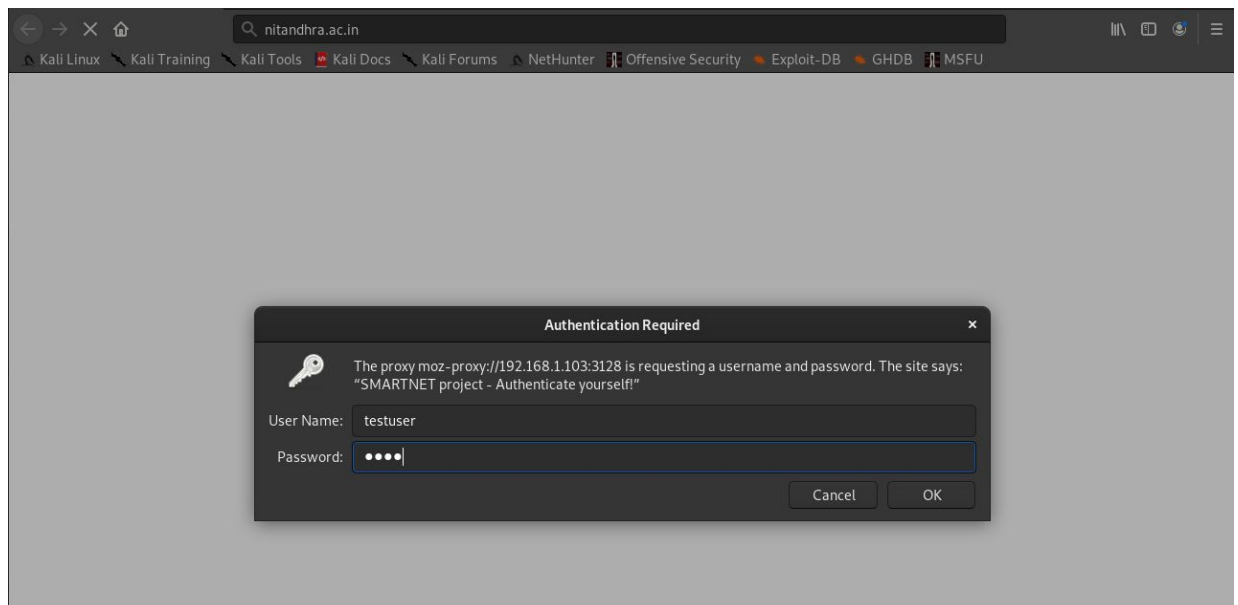


Figure-13: The squid proxy server asking for authentication credentials on desktop firefox browser

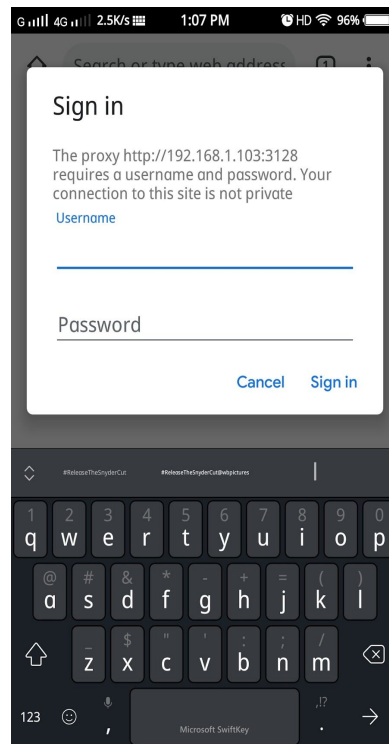


Figure-14: The proxy server asking for authentication credentials on mobile chrome browser

5.2.2. Squid cache codes

The squid proxy server maintains different log files for access, cache requests. The cache result code indicates how a request is handled by the squid proxy server. Some of the important codes used by the squid proxy server are described below.

Code	Description
TCP_HIT	It means the requested content is already in the cache and the squid proxy server doesn't have to forward the request.
TCP_MISS	It means the content is not in the cache and the squid proxy server forwards the request.

TCP_REFRESH_HIT	The requested content is in the cache and the squid server got an up-to-date response from the actual server on sending an if-modified-since request.
TCP_REF_FAIL_HIT	The requested content is in the cache and the squid server failed to get a response from the actual server on sending an if-modified-since request. In this case, the old content is served to the client.
TCP_REFRESH_MISS	The requested content is in the cache and the squid server got the updated content from the actual server on sending an if-modified-since request.
TCP_DENIED	The client request is denied by the squid server.
TCP_REFRESH_UNMODIFIED	The requested content is in the cache and the squid server received the unmodified response from the actual server.

Table-2: squid server cache codes

```
1590305364.826      0 192.168.1.134 TCP_SWAPFAIL_MISS_ABORTED/000 0
GET http://nitandhra.ac.in/favicon.ico testuser
HIER_DIRECT/61.0.228.111 -

1590305398.867    2271 192.168.1.134 TCP_MISS/200 355993 GET
http://nitandhra.ac.in/dept/ece_old/Students/IIIIYear.pdf testuser
HIER_NONE/- application/pdf

1590305398.867    2777 192.168.1.134 TCP_MISS/200 355994 GET
http://nitandhra.ac.in/dept/ece_old/Students/IIIIYear.pdf testuser
HIER_DIRECT/61.0.228.111 application/pdf
```

Figure-15: squid server access log when accessing IIIIYear.pdf file from nit andhra website for the first time using desktop FireFox browser.

```
1590305823.438      0 192.168.1.101 TCP_DENIED/407 4502 GET
http://nitandhra.ac.in/dept/ece_old/Students/IIIIYear.pdf -
HIER_NONE/- text/html

1590305847.044    7518 192.168.1.101 TCP_REFRESH_UNMODIFIED/200
356002 GET http://nitandhra.ac.in/dept/ece_old/Students/IIIIYear.pdf
testuser HIER_DIRECT/61.0.228.111 application/pdf
```

Figure-16: squid server access log when accessing IIIIYear.pdf file from nit andhra website for the second time using mobile chrome browser.

5.3. MODULE-3

5.3.1. Procedure

While using the aforementioned services the data privacy and security of a client must be protected. Below are the techniques used by popular social media platforms like WhatsApp, Facebook, Telegram, etc.

Apps	Authentication	Data transfer
WhatsApp	HMAC - SHA256	AES-256
Telegram	SHA-1	AES-256
Twitter	OAUTH2	CAST-256

Table-3: Various protocols used in popular platforms

The above implementations involved these protocols at the core.

Authentication	SHA256, MD5, SHA1
Data transfer	AES-128, DES

Table-4: proposed protocols

6. RESULTS AND DISCUSSIONS

This project is more application-oriented than theoretical, which means much importance is given to the implementation area and those results can be seen in the previous chapter. New options and features can be built on this project. In this chapter, we will discuss some observations.

6.1. Module 1

We have tested this module across different devices for different features. We observed that MediaRecording is not supported in case of android and iOS. However, this is not the limitation of this project, but the limitation of the WebRTC protocol.

Feature	Android	iOS	Web
Audio/Video	✓	✓	✓
Data Channel	✓	✓	✓
Screen Capture	✓	✓	✓
MediaRecorder	⚠	⚠	✓

Table-5 WebRTC supported features across Android, iOS and Web

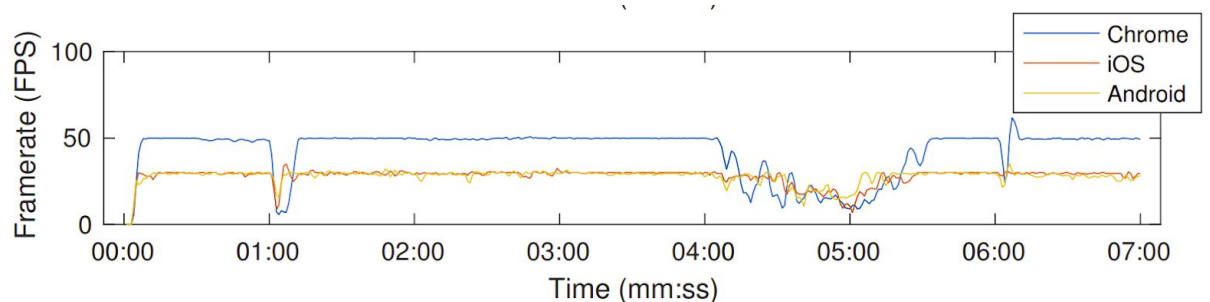


Figure-17 frame rates for iOS, Android and Chrome

6.2. Module 2

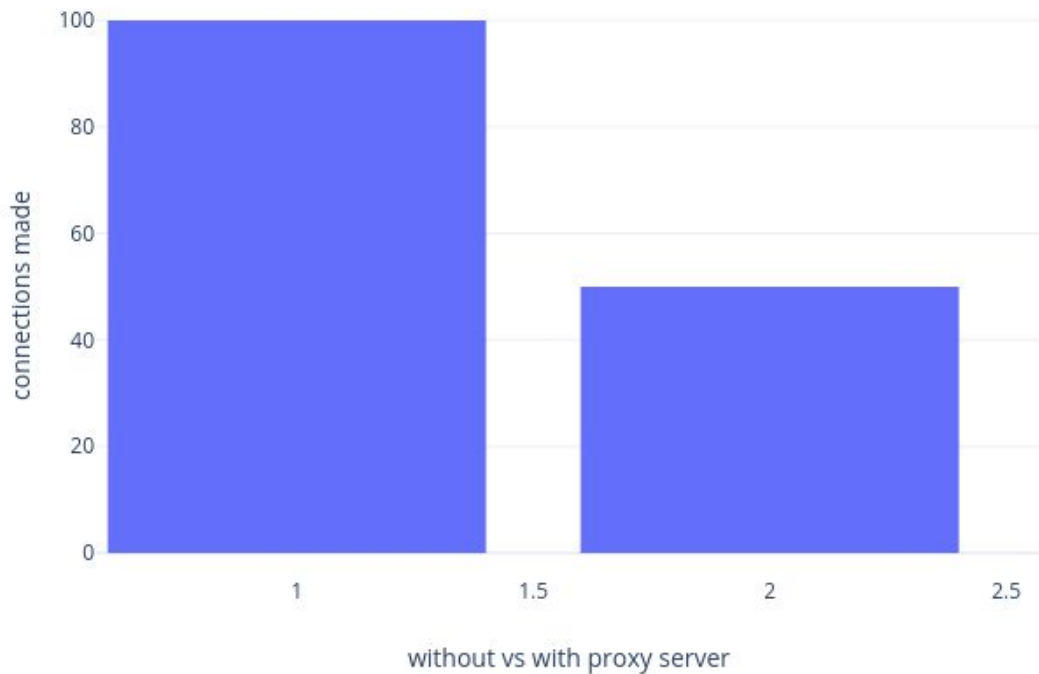


Figure-18 relative performance without and with the proxy server

Figure-18 shows the presence of the proxy server on the network. The use of the proxy server has drastically cut down the number of connections made to outside the LAN. However, the graph shown is only an approximation and better-customized results can be obtained once the project is deployed on the real network.

7. SECURITY ANALYSIS

Security should always be considered when developing, installing any software or a protocol. Although the project achieves the objectives promised, this section gives a brief security analysis of the whole project. We will describe the security issues and suggest some ways to evade the popular attacks like DoS, phishing, session-hijacking, etc.

7.1. Module 1

This module provides P2P communication by using WebRTC protocol. Primarily the WebRTC protocol has an evolutionary advantage over the typical VOIP protocol without encryption. Previously, most of the softwares using VOIP consider security as an optional parameter. However, WebRTC forces using only encrypted communications. It is considered as the most secure VOIP solution as it no longer considers the security as an optional parameter. There are many attacks possible on the WebRTC protocol, but we will only discuss those that fit our module's use-case (communication inside LAN only).

7.1.1. *Accessing media resources*

As part of the WebRTC protocol the user media streams (audio & video) are requested first and then the communication takes place. However, today's browser ask the users whether to remember their preference of media streams for future communications. An adversary can use this vulnerability and see the sensitive information on the user's screen directly without the user's consent when a call is connected. Similar problem may also arise when a media resource is being used by the other tab(recording, playing video) in the browser. Hence, we strongly suggest not to remember the user's preference and show pop-up to ask the user every time when requesting the media streams.

7.1.2. *SRTP weakness*

Secure Realtime Transfer Protocol(Secure RTP) is used by WebRTC protocol to add encryption over RTP. However, the encryption is added to all parts of the message except the header. Often times it is desirable, depending on the use case, to keep the header as a secret

since it can reveal information about the other user, if an adversary is eavesdropping. This helps the adversary to know about the other user whom the first user is communicating with. This is a drawback of WebRTC protocol and hence a security issue in our project. But it is recommended to hide the other user's details from the header by writing custom headers till the WebRTC protocol receives an update on this issue.

7.1.3. Signalling protocol

The flexibility of using the WebRTC protocol is that it leaves the software developers to use their own signalling mechanism. Signalling mechanism is not offered by WebRTC and it is up to the developers to design their own signalling system and SIP (Session Initiation Protocol) is the most popular choice. The SIP is direct descendant of SMTP and HTTP, which use unencrypted communications.

The WebRTC protocol cannot guarantee security if the signalling protocol uses unencrypted messages. An adversary can directly see the unencrypted messages of SIP and the consequences are devastating as can lead to numerous attacks like identity theft, data breach, MITM, etc.

7.1.3.1. MITM

Initial SIP messages can be intercepted by the adversary and then perform MITM during the course of communication. This attack is so popular and is hard to detect because it does not have any warning. However, one can detect it if they have little knowledge. The communicating users in the presence of MITM attack experiences a small delay which can then confirm that the communication is compromised.

7.1.3.2. Replay attack

An adversary can do replay attack on to the server, asking re-initiate the call with second user faking the identity of first user. This request is similar to the previous request received by the second user. A well-designed protocol would be intelligent enough to dodge this attack.

7.1.3.3. Session hijacking

The cookies containing the session ID given by the server to the user's browser contains crucial information. An adversary can employ an attack against the targeted user and copy the

cookies, hence gaining the control over the existing session of a legitimate user. We suggest rejecting any connection using a session ID which is currently in use to prevent this attack.

Even though the above attacks are possible on SIP, we can also encrypt the signalling layer for better security. That is, the signalling protocol even if not using encryption the messages sent from signalling layer can be encrypted by using secure websockets(wss://) instead of websockets(ws://). So, in order to prevent the above attack we have use secure websockets and it is recommended to develop custom signalling protocol on top of the secure websockets for better security.

7.2. Module 2

The security of the module-2 depends on the squid configuration file. There are a many parameters that can be configured. The default configuration provides minimal security to get started. However, if the project needs to be deployed on a real network, we propose below steps to make the system more secure.

7.2.1 Port number

The default port number on which squid service runs is 3128. An adversary can perform an attack when the default port number is publicly known. Hence, it is recommended to change the port to other available port number available from a large range. This makes it a difficult task for an adversary to port-scan the server as port scanning over such a huge range takes a lot of time. Moreover, if the proxy server has multiple network interfaces it is recommended to listen only on the IP address of the interface on which the whole university network is present, thereby rejecting the connections from other interfaces for better security.

```
http_port 54321
http_port 172.30.10.1:54321
```

7.2.2 Maximum cache object size

The squid proxy server also allows to configure the maximum cacheable size of the object through parameters *max_object_size* and *min_object_size*. These parameters are used to achieve optimal hit ratio. An adversary can choose to perform DoS attack by overloading the cache object size and can slow down the proxy server for other users which happens only when

there is no upper limit set on the cacheable object size. The parameter *max_object_size* is helpful in this case to prevent an adversary to overload the object size and carryout DoS attack. Other parameters like *quick_abort_min*, *quick_abort_max* can be used to tell the proxy server to stop caching those requests which are cancelled as those requests might be sent by an adversary to perform DoS attack.

```
maximum_object_size <size>
minimum_object_size <size>
```

7.2.3 DNS poisoning

A planned adversary can also poison the DNS server used by the proxy server. And a compromised DNS is used by the adversary to redirect the users to phishing pages and the credentials of user can be compromised. Though poisoning DNS is difficult but there is always a scope for an adversary. However, we can choose the DNS by configuring in the squid configuration file.

```
dns_nameservers 8.8.8.8
```

7.2.4 Authentication

Since authentication is the core part of the project, it is important to secure the users and their accounts as their activity is associated with it. The parameters *authenticate_ttl* and *authenticate_ip_ttl* can be used for achieving better security. The parameter *authenticate_ttl* specifies the duration time during which the server remembers client authentication information. This forces the client to re-authenticate again after that duration. Another version of this parameter, *authenticate_ip_ttl* denotes the maximum duration an IP address is allotted to a user. By configuring these parameters the attacks like session-hijacking can be prevented.

```
authenticate_ttl <time>
authenticate_ip_ttl <time>
```

7.2.5 Request header

Every web request has a header associated with it. It is observed that most of the DoS attacks exploit this header option by send huge requests with larger header size than the server can handle. In order to prevent this attack we suggest using the parameter

request_header_max_size to set an upper limit on the header size of a request. Hence, the chance of DoS attack is decreased as the proxy server now drops the requests with header size greater than this value.

```
Reuquest_header-max_size <size>
```

7.2.6. *Client lifetime*

Allowing a client to stay connected to the network for a long time may not be a good option and it depends on the use case. For a company it is good to disconnect the clients on the network during non-working hours. This is actually a good option to consider, because they may be spy devices installed by an employee and it would be an excellent choice to disconnect them during non-working hours. Moreover, an adversary can use this vulnerability and open numerous socket connections to the server and thereby hanging up the server processes causing DoS attack. This helps to improve the security of the network. The proxy server can be configured with *client_lifetime* parameter with a value of maximum working hours. A variation of the former parameter *pconn_timeout* can also server the same function as *client_lifetime* except it applies only to the idle clients.

```
client_lifetime <max_time>  
pconn_timeout <max_time>
```

7.2.7. *Logging*

The proxy server maintains different log files such as access.log, cache.log, etc. The log files are the key for the improvement of the proxy server. They can reveal information about cache hits, cache misses, websites being visited etc. A malicious activity can be detected from the log files. It is recommended to back up the log files periodically as the old logs are overridden when the size of the file grows in order to save the old logs. And we recommend a constant monitoring of these log filed to detect any anomalies for better security. However, it can also be automated by using machine learning algorithms.

8. CONCLUSIONS AND FUTURE SCOPE

This project is an embodiment of numerous research works. It is more application oriented than theoretical, which means there is always room for improvements. New features can always be developed on top of this project. We have succeeded in developing the software to demonstrate the project goals. However, it is not recommended to deploy this software directly on the network since there can be some improvements made in the configuring the server.

When enormous requests are encountered the server becomes bottleneck and scaling is required. The server can be expanded to handle a large number of requests through load balancing. Tools like Nginx can be used for load-balancing to increase performance. Another area of research work can be carried out in the cache replacement policy. Machine learning concepts like Neural Networks can be used for enhancing the performance of cache replacement policies.

The project's concept is so simple that it can lead to numerous interpretations. The project research work can be expanded into different fields such as machine learning, blockchain, cryptography, etc.

REFERENCES

- [1] Kbar, G., Mansoor, W., & Naim, A. (2010, September). Voice over IP mobile telephony using WIFI P2P. In *2010 6th International Conference on Wireless and Mobile Communications* (pp. 268-273). IEEE.
- [2] Sredojev, B., Samardzija, D., & Posarac, D. (2015, May). WebRTC technology overview and signalling solution design and implementation. In *2015 38th international convention on information and communication technology, electronics and microelectronics (MIPRO)* (pp. 1006-1009). IEEE.
- [3] Bestak, R., & Hlavacek, J. (2017, June). A videoconferencing system based on webrtc technology. In *International Conference on Computer Networks* (pp. 245-255). Springer, Cham.
- [4] Liu, Z., Bai, Z., Liu, Z., Li, X., Kim, C., Braverman, V., & Stoica, I. (2019). Distcache: Provable load balancing for large-scale storage systems with distributed caching. In *17th {USENIX} Conference on File and Storage Technologies ({FAST} 19)* (pp. 143-157).
- [5] Chi, X., Liu, B., Niu, Q., & Wu, Q. (2012, July). Web load balance and cache optimization design based nginx under high-concurrency environment. In *2012 Third International Conference on Digital Manufacturing & Automation* (pp. 1029-1032). IEEE.
- [6] Kunda, D., Chihana, S., & Sinyinda, M. Web Server Performance of Apache and Nginx: A Systematic. (pp. 43-52).
- [7] Guo, K., Yang, C., & Liu, T. (2017, March). Caching in base station with recommendation via Q-learning. In *2017 IEEE Wireless Communications and Networking Conference (WCNC)* (pp. 1-6). IEEE.
- [8] Harn, L., & Lin, C. (2010). Authenticated group key transfer protocol based on secret sharing. *IEEE transactions on computers*, 59(6), 842-8
- [9] Roy, S., & Mishra, A. Squid Proxy Server A Practical Approach.
- [10] Chi, X., Liu, B., Niu, Q., & Wu, Q. (2012, July). Web load balance and cache optimization design based nginx under high-concurrency environment. In *2012 Third International Conference on Digital Manufacturing & Automation* (pp. 1029-1032). IEEE.

