VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi-590018

COMPUTER GRAPHICS LABORATORY (18CSL67)
MINI PROJECT REPORT ON

"FIREWALL"

Submitted in the partial fulfillment of the requirements for VI semester of Bachelor of Engineering

In

COMPUTER SCIENCE ENGINEERING

Submitted By

KUNAAL SHIVAKUMAR	**USN: 1BY20CS092**
MOHIT KRISHNA SAI RAM	**USN: 1BY20CS105**
MITHIL MENON	**USN:1BY20CS111**

Under the guidance of

Prof. Shankar R	**Prof. Chethana C**
Assistant Professor	**Assistant Professor**
Dept. of CSE	**Dept. of CSE**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT

YELAHANKA, BENGALURU - 560064.

2022-2023

BMS INSTITUTE OF TECHNOLOGY &MANAGEMENT

YELAHANKA, BENGALURU – 560064

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project work entitled **“FIREWALL”** is a bonafide work carried out by **Kunaal Shivakumar(1BY20CS092), Mohit Krishna Sriram (1BY20CS105) and Mithil Menon(1BY20CS111)** in partial fulfillment for *Mini Project* during the year 2021-2022. It ishereby certified that this project covers the concepts of *Computer Graphics & Visualization*. It is also certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in this report.

Signature of the Guide	**Signature of the Guide**	**Signature of the HOD**
Prof. Shanakar R	**Prof Chethana C**	**Dr. Thippeswamy G**
Assistant Professor	Assistant Professor	Professor & HOD
CSE, BMSIT&M	CSE, BMSIT&M	CSE, BMSIT&M

EXTERNAL VIVA – VOCE

Name of the Examiners	**Signature with Date**
1. ____________________	____________________
2. ____________________	____________________

INSTITUTE VISION

To emerge as one of the finest technical institutions of higher learning, to develop engineering professionals who are technically competent, ethical and environment friendly for betterment of the society.

INSTITUTE MISSION

Accomplish stimulating learning environment through high quality academic instruction, innovation and industry-institute interface.

DEPARTMENT VISION

To develop technical professionals acquainted with recent trends and technologies of computer science to serve as valuable resource for the nation/society.

DEPARTMENT MISSION

Facilitating and exposing the students to various learning opportunities through dedicated academic teaching, guidance and monitoring.

PROGRAM EDUCATIONAL OBJECTIVES

1. Lead a successful career by designing, analyzing and solving various problems in the field of Computer Science & Engineering.
2. Pursue higher studies for enduring edification.
3. Exhibit professional and team building attitude along with effective communication.
4. Identify and provide solutions for sustainable environmental development.

PROGRAM SPECIFIC OUTCOMES

1. Analyse the problem and identify computing requirements appropriate to its solution.
2. Apply design and development principles in the construction of software systems of varying complexity.

ACKNOWLEDGEMENT

We are happy to present this project after completing it successfully. This project would not have been possible without the guidance, assistance and suggestions of many individuals.

We would like to express our deep sense of gratitude and indebtedness to each and every one who has helped us make this project a success.

We heartily thank our Principal, **Dr. MOHAN BABU G N,** BMS Institute of Technology &Management, for his constant encouragement and inspiration in taking up this project.

We heartily thank our Professor and Head of the Department, **Dr. Thippeswamy G**, Department of Computer Science and Engineering, BMS Institute of Technology and Management, for his constant encouragement and inspiration in taking up this project.

We gracefully thank our Project Guide, **Prof. Shankar R and Prof. Chethana C**, Assistant Professors, Department of Computer Science and Engineering for their intangible support and for being constant backbone for our project.

Special thanks to all the staff members of Computer Science Department for their help and kind co-operation.

Lastly, we thank our parents and friends for the support and encouragement given throughout in completing this precious work successfully.

KUNAAL SHIVAKUMAR (1BY20CS081)

MOHIT KRISHNA SAI RAM(1BY20CS105)

MITHIL MENON(1BY20CS111)

ABSTRACT

- Main aim of this Mini Project is to illustrate the concepts and usage of Fire Wall in OpenGL
- A firewall can either be software-based or hardware-based and is used to help keep a network secure.
- Its primary objective is to control the incoming and outgoing network traffic by analyzing the data packets and determining whether it should be allowed through or not, based on a predetermined rule set.
- A network's firewall builds a bridge between the internal network or computer it protects, upon securing that the other network is secure and trusted, usually an external (inter)network
- We have used input devices like mouse and keyboard to interact with program.
- We have also used Solid Cube for forming a complete network setup which help to understand concept of Congestion Control very well.
- To differentiate between objects, we have used different colors for different objects.
- We have added menu which makes the program more interactive.
- In this project we have used a small Solid Cube to represent a data, which travels as data transfer from source to destination.
- We have used font family for indicating the name of objects as we can see in this project.

TABLE OF CONTENTS

TABLE OF CONTENTS

CHAPTER 01

Brief introduction to the project including the idea behind it is explained in this chapter.

1.1 Brief Introduction

- In the field of computer graphics and visualization, the CGV (Computer Graphics and Visualization) Mini Project focuses on the implementation of a firewall using OpenGL. A firewall is a security system designed to monitor and control incoming and outgoing network traffic, ensuring the protection of a computer network from unauthorized access and potential threats.
- The purpose of this mini project is to develop a visual representation of a firewall using the OpenGL (Open Graphics Library) framework. OpenGL is a powerful and widely-used graphics library that provides a set of functions for rendering 2D and 3D graphics. By utilizing OpenGL, we can create a visually appealing and interactive representation of a firewall system.
- The main objectives of this mini project include:
- Designing and implementing a graphical user interface (GUI) for the firewall using OpenGL.
- Creating a realistic representation of a computer network with various nodes, connections, and datapackets.
- Implementing firewall rules and policies to control the flow of network traffic.
- Visualizing the network traffic in real-time, highlighting the packets that are allowed or blocked by the firewall.
- Providing user interaction to modify firewall rules and policies dynamically.
- By accomplishing these objectives, we aim to enhance understanding and visualization of firewall operations in a computer network environment. This mini project serves as an educational tool to explore the concepts of network security and demonstrate the importance of firewalls in safeguarding computer networks.
- Throughout the development process, we will leverage the capabilities of OpenGL to create a visually appealing and interactive experience. Users will be able to observe the flow of network traffic, monitor the actions of the firewall, and make modifications to the firewall rules based on their preferences.

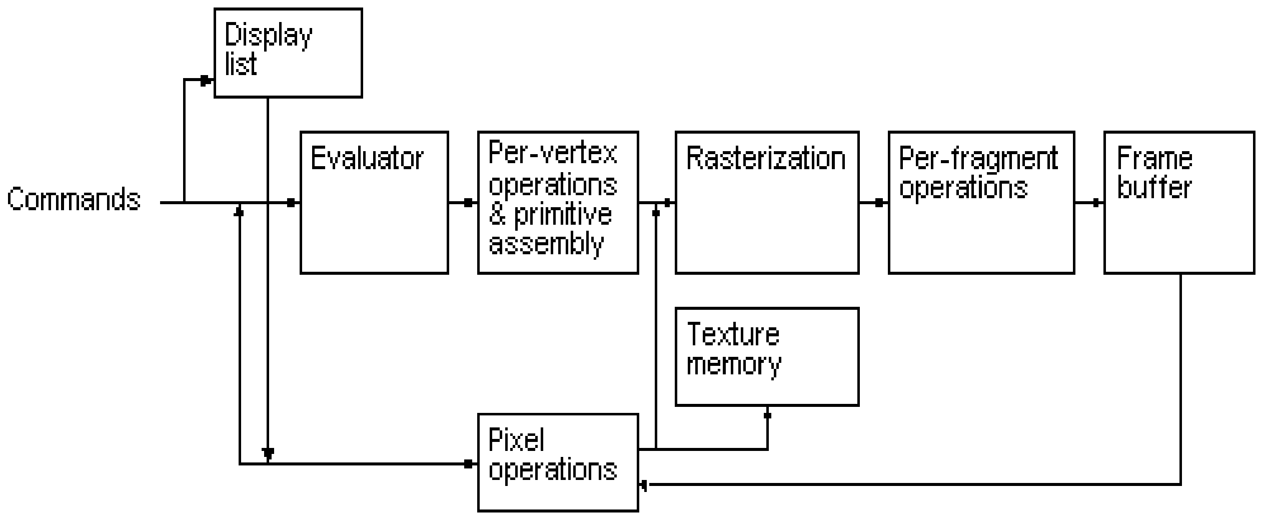


Fig: 1.4 OpenGL Block Diagram

1.2 Motivation

The Motivation of this project are summarized below:

- **Network Security Awareness:** Cybersecurity threats are a significant concern for individuals, businesses, and organizations alike. By delving into firewalls, this mini-project aims to raise awareness about the importance of network security and equip individuals with knowledge to protect their networks from potential threats.
- **Practical Implementation**: The mini-project focuses on providing practical knowledge of configuring and managing firewalls. By exploring firewall rules, best practices, and real-world examples, participants gain hands-on experience in implementing effective network security measures.
- **Risk Mitigation**: Implementing firewalls is an essential risk mitigation measure for organizations. By understanding the concepts and best practices related to firewalls, participants can contribute to improving the security posture of their organizations, thereby reducing the likelihood and impact of potential security incident.

1.3 Scope

- **Network Security**: The firewall would play a crucial role in securing the network infrastructure of theCGV project.

- **Access Control**: The firewall would enforce access control policies, determining which network resources and services are accessible to different users or components within the CGV mini project.

- **Traffic Filtering**: The firewall can filter network traffic based on various criteria such assource/destination IP addresses, port,numbers, protocols, or specific content patterns.

- **Intrusion detection and prevention:** Firewalls can incorporate intrusion detection and prevention systems (IDPS) to detect and block unauthorized or malicious activities.

- **Load balancing:** Graphics projects often involve distributing the workload across multiple servers or nodes for scalability and performance reasons. Firewalls can be configured to support load balancing.

- **Intrusion detection and prevention:** Firewalls can incorporate intrusion detection and prevention systems (IDPS) to detect and block unauthorized or malicious activities.

1.4 Problem Statement

Our program must:

- The problem statement for the CGV mini-project on firewalls is to design and implement an effective firewall solution for a network infrastructure that addresses the following objectives:
- **Network Segmentation:** Develop firewall rules and policies to create distinct security zones within the network, preventing unauthorized access between these zones.
- **Access Control:** Define and enforce firewall rules to control inbound and outbound network traffic, allowing only authorized communication while blocking malicious or suspicious activities.
- **Threat Prevention:** Implement firewall configurations and settings that effectively detect and block common network threats, such as malware, intrusion attempts, and unauthorized access.
- **Monitoring and Logging:** Enable comprehensive logging and monitoring of network traffic, including firewall events, to facilitate the identification and analysis of security incidents or anomalies.
- **Scalability and Performance:** Ensure that the firewall solution can handle the network's traffic load without compromising performance, considering factors such as packet filtering, deep packet inspection, and network throughput.
- The goal of this mini-project is to provide a practical understanding of firewall technologies and their application in network security. By addressing the problem statement, participants will gain hands-on experience in configuring, managing, and optimizing firewalls to enhance the security posture of a network infrastructure.

1.5 Proposed System

Proposed System for CGV Mini-Project: Firewall

The proposed system for the CGV mini-project on firewalls aims to implement an effective firewall solution for network security. The system will include the following components and functionalities:

1. Firewall Hardware/Software Selection: Selecting an appropriate firewall hardware or software solution based on the network's requirements, such as the number of users, expected traffic volume, and budget constraints.

2.Network Architecture Design: Designing a network architecture that incorporates the firewall solution effectively. This includes determining the placement of the firewall within the network topology to ensure optimal security and performance.

3.Firewall Rule Configuration: Defining and configuring firewall rules based on the network's security policies and objectives. This involves specifying the allowed and blocked traffic based on criteria such as source and destination IP addresses, port numbers, protocols, and application-level inspections.

4.Intrusion Detection and Prevention: Implementing intrusion detection and prevention mechanisms within the firewall to detect and block malicious activities, including unauthorized access attempts, malware, and suspicious network traffic.

5.Network Segmentation: Creating distinct security zones within the network using firewall rules and policies. This involves dividing the network into logical segments and controlling the traffic flow between them to prevent unauthorized access and contain potential security breaches.

6.Logging and Monitoring: Enabling comprehensive logging and monitoring capabilities within the firewall system to track network traffic, firewall events, and security incidents. This will facilitate real-time analysis, threat detection, and incident response.

7.Regular Updates and Maintenance: Implementing a maintenance plan to ensure that the firewall system remains up-to-date with the latest security patches, firmware updates, and configuration adjustments. Regular monitoring and fine-tuning of the firewall rules will also be conducted to optimize performance and adapt to evolving threats.

8.Documentation and Reporting: Creating documentation that outlines the firewall configuration, rules, and policies implemented in the network. Generating periodic reports on firewall activities, traffic patterns, and security incidents to facilitate auditing, compliance, and ongoing network security assessment.

The proposed system aims to provide a robust and well-configured firewall solution that enhances the network's security posture, mitigates threats, and aligns with industry best practices. By implementing and managing this system, participants will gain practical knowledge and skills in firewall technologies and network security management.

1.6 Limitations

- **Simplified Environment:** The mini-project may utilize a simplified network environment, which may not reflect the complexity and scale of real-world network infrastructures.
- **Hardware/Software Constraints**: Depending on the resources available for the mini-project, there may be limitations in terms of the hardware or software firewall solutions that can be used.
- **Lack of Real-Time Network Traffic:** The mini-project may not provide access to live network traffic, which can impact the ability to analyze and respond to real-time security incidents
- **Scalability**: If your project involves a distributed firewall system or multiple firewall instances, scalability becomes a consideration.
- **Maintenance and support:** Incorporating graphics into a firewall project may require ongoing maintenance and support. Graphics libraries or frameworks might require updates.
- **Usability:** Firewalls are primarily designed for managing network traffic and enforcing security policies. While visualizations can enhance the user experience, it's crucial to consider the usability aspects.

CHAPTER 02

LITERATURE SURVEY

- ✓ **https://ieeexplore.ieee.org/document/7444779**

Firewalls are security devices used to apply an organization's security policy. However, commercial firewalls, such as Juniper Networks and Cisco ASA firewall devices, are mainly designed to be used by networking and security professionals, are not very appropriate for the academia environment, and lack simplicity. In addition, commercial firewalls are usually considered high-cost hardware devices. However, academic institutions usually have tight budget and few financial resources for purchasing networking and security devices for their laboratories. To overcome the aforementioned limitation of commercial firewalls, an educational firewall hardware device, called Edu-Firewall, is discussed and demonstrated. Edu-Firewall main objectives are to offer advanced educational security functions that are not commonly available in current commercial firewalls, and an easy-to-use friendly graphical interface to configure the firewall and manipulate the filtering rules. In addition, Edu-Firewall's objective is to offer an affordable educational device, compared to the available commercial firewall devices. Edu-Firewall allows students to implement hands-on lab exercises on firewalls and better anatomize network traffic filtering concepts and firewall configuration, and contributes to enhance students hands-on security

- ✓ **https://ieeexplore.ieee.org/document/6567326**

Firewall is an important tool for protecting the security of the network from attacking between trusted and untrusted networking. It detects and filters the packets that pass through itself. However, the security levels depend on verification measures which usually follow by organization policies or rules. If some organizations want a strict usability to access information on the untrusted network, the firewall rules are therefore complicated by the policies. Basically, the policy usually consists of a group of six conditions, and is divided into two parts, namely predicate and decision as follows:

Network security is usually protected by a firewall, which checks in-out packets against a set of defined policies or rules. Hence, the overall performance of the firewall generally depends on its rule management. For example, the performance can be decreased when there are firewall rule anomalies. The anomalies may happen when two sets of firewall rules are overlapped or their decision parts are both an acceptance and a denial simultaneously. In this paper, we propose a new paradigm of the firewall design, consisting of two parts: (1) Single Domain Decision firewall (SDD) -a new firewall rule management policy that is certainly not conflicts, and (2) the Binary Tree Firewall (BTF) -a data structure and an algorithm to fast check the firewall rules. Experimental results have indicated that the new design can fix conflicting anomaly and increase the speed of firewall rule checking from O(N 2) to O(log 2 N).

CHAPTER 03

SYSTEM REQUIREMENTS AND SPECIFICATIONS

3.1 SOFTWARE REQUIREMENTS

1. **Operating System**: Microsoft Windows XP, Microsoft Windows 7 and above. An operating system (OS) is the program that, after being initially loaded into the computer by a boot program, manages all of the other application programs in a computer. The application programs make use of the operating system by making requests for services through a defined application program interface (API). In addition, users can interact directly with the operating system through a user interface, such as a command-line interface (CLI) or a graphical UI (GUI).

2. **Code Blocks installed.**

   Code::Blocks is a free, open-source cross-platform IDE that supports multiple compilers including GCC, Clang and Visual C++. It is developed in C++ using wx Widgets as the GUI toolkit. Using a plugin architecture, its capabilities and features are defined by the provided plugins. Currently, Code::Blocks is oriented towards C, C++, and Fortran. It has a custom build system and optional Make support.

3. **Compiler used: VC++ 6.0 compiler:**

   Microsoft Visual /C++ (**MSVC**) is a compiler for the C, C++ and C++/CX programming languages by Microsoft. MSVC is proprietary software; it was originally a standalone product but later became a part of Visual Studio and made available in both trialware and freeware forms.

4. Language used: Visual C++
5. Free Glut Software

FreeGLUT is an open-source alternative to the OpenGL Utility Toolkit (GLUT) library. GLUT (and hence FreeGLUT) allows the user to create and manage windows containing OpenGL contexts on a wide range of platforms and also read the mouse, keyboard and joystick functions. FreeGLUT is intended to be a full replacement for GLUT, and has only a few differences.

3.2 HARDWARE REQUIREMENTS

6. Processor: Intel Core^{TM}Core i3-32 bit
7. Processor Speed: 2.9 GHz
8. RAM Size: 8GB DDR3
9. Graphics-2GB
10. Cache Memory -2MB
11. Graphics Card
12. INPUT devices such as mouse, keyboard, etc.

CHAPTER 04

SYSTEM ANALYSIS

4.1 INITIALIZATION

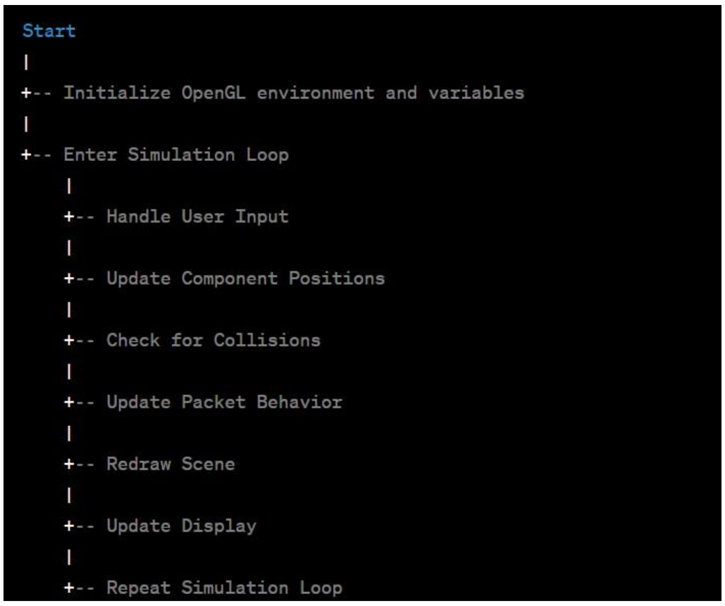
- Initialize to interact with the Windows.
- Initialize the display mode that is double buffer and RGB colour system.
- The code likely contains an initialization function that sets up the OpenGL environment and initializes necessary variables.
- This function may perform tasks such as creating a window, specifying the viewport, and setting up any required OpenGL configurations.
- Additionally, it may initialize variables related to the network components, suchas their initial positions, velocities, and states.

4.2 DISPLAY

- Introduction page of “FIREWALL”.
- Menus are created and depending on the value returned by menus.
- Suitable operations are performed.
- The code includes a rendering function responsible for drawing and displaying thenetwork components and packets on the screen.
- The rendering function utilizes OpenGL functions to create 3D objects and applytransformations.
- It may iterate over the network components and packets, drawing them using appropriate OpenGL calls and specifying their positions.

4.3 FLOW CHART

- The code likely includes a simulation loop that updates the state of the network components and packets over time.
- The simulation loop controls the flow of the program and repeatedly executes a setof instructions.
- Within each iteration of the simulation loop, the following steps might occur:
- Handle user input: Check for user interactions, such as keyboard or mouse input, and update the state of the simulation accordingly.
- Update component positions: Move the network components based on their velocities or any other defined rules.
- Check for collisions: Detect and handle collisions between packets and network components. This might involve checking for overlaps or proximity between objects.
- Update packet behavior: Determine the behavior of packets based on defined rules or protocols. This could involve changing their directions, speeds, or triggering specific events.
- Redraw the scene: Call the rendering function to redraw the network components and packets with their updated positions and states.
- Update display: Refresh the display to show the newly rendered scene on the screen.



```
Start
|
+-- Initialize OpenGL environment and variables
|
+-- Enter Simulation Loop
    |
    +-- Handle User Input
    |
    +-- Update Component Positions
    |
    +-- Check for Collisions
    |
    +-- Update Packet Behavior
    |
    +-- Redraw Scene
    |
    +-- Update Display
    |
    +-- Repeat Simulation Loop
```

Fig: 3.3 Flow Chart

CHAPTER 05

SYSTEM IMPLEMENTATION

5.1 STRUCTURE

- ✓ void setFont (void *font);
- ✓ void drawstring (float x,float y,float z,char *string);
- ✓ voidstroke_output(GLfloat x, GLfloat y, char *format,...);
- ✓ void server();
- ✓ void plane();
- ✓ void wall();
- ✓ void wall();
- ✓ void user();
- ✓ void pc();
- ✓ void port();
- ✓ void packet();
- ✓ void port(float x1,float y1);
- ✓ void display();
- ✓ void spring();
- ✓ void myinit();
- ✓ void info();
- ✓ int main(int argc,char** argv);

5.2 ANALYSIS

FUNCTIONS

A function is a block of code that has a name and it has a property that it is reusable that is it can be executed from as many different points in a c program as required. The partial code of various function that have been used in the program are:

5.2.1 doInit

```
void doInit()
{
   glClearColor(0.6,0.7,0.8,0.0);
   glColor3f(.0,1.0,1.0);
  glViewport(0,0,640,480);
  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();

   gluPerspective(30.0f,(GLfloat)640/(GLfloat)480,0.1f,200.0f);
}
```

This function is used to initialize the graphics window.GlMatrixMode(GL_PROJECTION), glLoadIdentity() are used to project the output on to the graphics window.

5.2.2 DISPLAY

```
void display ()
{
 glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
glLoadIdentity();
glTranslatef(0.0f,0.0f,-13.0f);
stroke_output(-2.0, 1.7, "s -> Start");
stroke_output(-2.0, 1, "q--> quit");

GLfloat mat_ambient[]={0.0f,1.0f,2.0f,1.0f};
GLfloat mat_diffuse[]={0.0f,1.5f,.5f,1.0f};
GLfloat mat_specular[]={5.0f,1.0f,1.0f,1.0f};
GLfloat mat_shininess[]={50.0f};
glMaterialfv(GL_FRONT,GL_AMBIENT,mat_ambient);
glMaterialfv(GL_FRONT,GL_DIFFUSE,mat_diffuse);
glMaterialfv(GL_FRONT,GL_SPECULAR,mat_specular);
glMaterialfv(GL_FRONT,GL_SHININESS,mat_shininess);
 GLfloat lightIntensity[]={3.7f,0.7f,0.7f,1.0f};
GLfloat light_position[]={0.0f,5.5f,4.0f,0.0f};

glLightfv(GL_LIGHT0,GL_POSITION,light_position);
glLightfv(GL_LIGHT0,GL_DIFFUSE,lightIntensity);

GLfloat lightIntensity1[]={3.7f,0.7f,0.7f,1.0f};
GLfloat light_position1[]={0.0f,-5.5f,4.0f,0.0f};

glLightfv(GL_LIGHT1,GL_POSITION,light_position1);
glLightfv(GL_LIGHT1,GL_DIFFUSE,lightIntensity1);



glEnable(GL_COLOR_MATERIAL);
glFlush();
glutSwapBuffers();
}
```

- The display function is responsible for rendering a scene using OpenGL. It starts by clearing the color and depth buffers using glClear(GL_COLOR_BUFFER_BIT| GL_DEPTH_BUFFER_BIT)`. Next, it sets up the view transformation by calling glLoadIdentity() to reset the current matrix and then glTranslatef() to move the scene in the -z direction.

- The function then uses stroke_output() to render text on the screen, displaying the options "s -> Start" and "q -> quit" at specific positions.

- After that, it sets up the material properties using glMaterialfv(), specifying the ambient, diffuse, specular, and shininess values.

- The lighting is set up by specifying the light position and intensity using glLightfv(). Two light sources are defined using `GL_LIGHT0` and `GL_LIGHT1`.

- Finally, glEnable(GL_COLOR_MATERIAL)` enables the use of glColor to specify material properties.

- The function ends by calling glFlush() to ensure all OpenGL commands are processed, glutSwapBuffers() to swap the front and back buffers for double buffering, and hence, displaying the rendered scene on the screen.

- In summary, the `display` function sets up the scene, renders text, applies material and lighting properties, and displays the scene using OpenGL functions.

5.2.3 PORT

```
void port(float x1,float y1){

 glPushMatrix();
 glTranslatef(x1,y1,-
 5.85);
 glScaled(0.4,0.45,0.0
 1);
 glutSolidSphere(1.0,40,40);

 glPushMatrix();
 glColor3f(0,0,0);
 glTranslatef(0,0,1);
 glScaled(1,1,0.01);
 glutSolidSphere(0.8,40,4
 0);glPopMatrix();

 glPopMatrix();

 }
```

- The port function is used to render a port object consisting of two solid spheres. It takes the coordinates x1 and y1 as parameters to position the port in the 3D space.

- The function first saves the current matrix state and then translates the object to the specified coordinates. It then applies scaling transformations to the object to adjust its size.

- The first sphere represents the main body of the port, while the second smaller sphere represents the opening of the port. The smaller sphere is positioned inside the main sphere and given a different color.

- Finally, the function restores the previous matrix state, ensuring that the transformations applied to the port object do not affect subsequent objects rendered in the scene.

5.2.4 SPRING

```
void spring(){
    glPushMatrix(); glColor3f(0,1,1);
    glTranslatef(-2.3,-2.5,-5.3);
    if(z>=9.6 && z<=10.2){
    glScaled(0.3,-1.5+z/10,0.3);
}
glScaled(0.4,1.5,0.4);
glRotatef(90,1,0,0);
glutWireTorus(0.25,0.4,34);
glPopMatrix();
}

void switch1(){
glPushMatrix(); glColor3f(0,1,1.2);
glTranslatef(-2,-2,5);
glScaled(1.5,0.5,2);
glutSolidCube(0.5);
glPopMatrix();
}
```

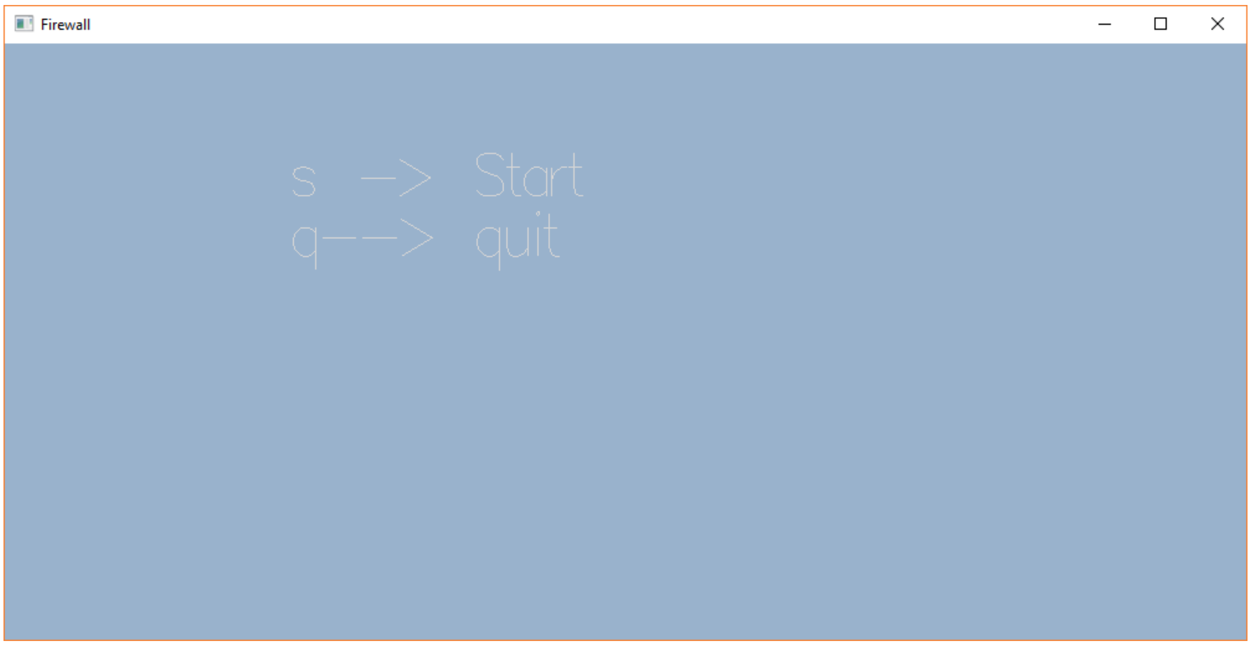
- The spring function renders a wireframe spring object. It sets the color to cyan, translates and scales the object accordingly, and then rotates it. Finally, it uses glutWireTorus() to render the wireframe torus.
- The `switch1` function renders a solid cube object. It sets the color to a shade of blue, translates the object to a specific position, scales it non-uniformly to achieve desired proportions, and then uses glutSolidCube() to render the solid cube.

CHAPTER06

INTERPRETATION OF RESULTS

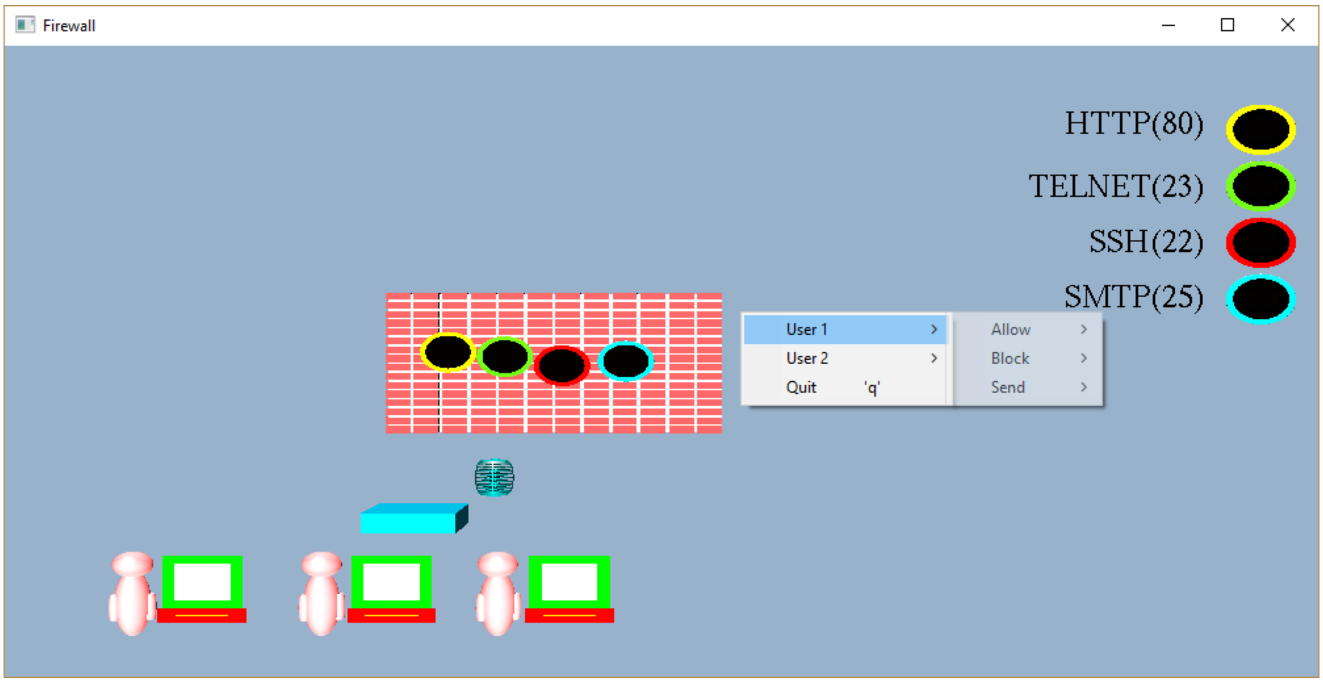
6.1 SNAPSHOTS

6.1.1 After Running the Program



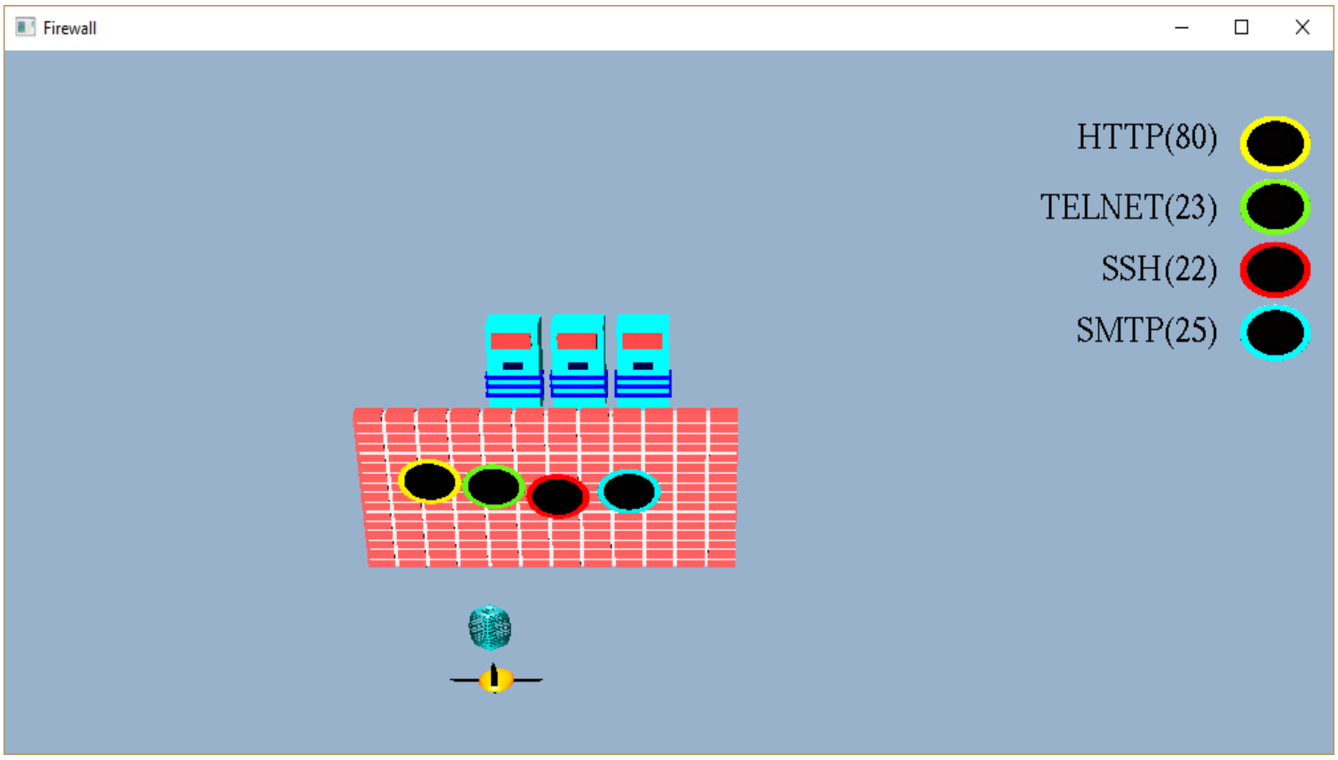
The above snapshot shows the screen displayed when the program gets Executed.

6.1.2 Firewall Menu



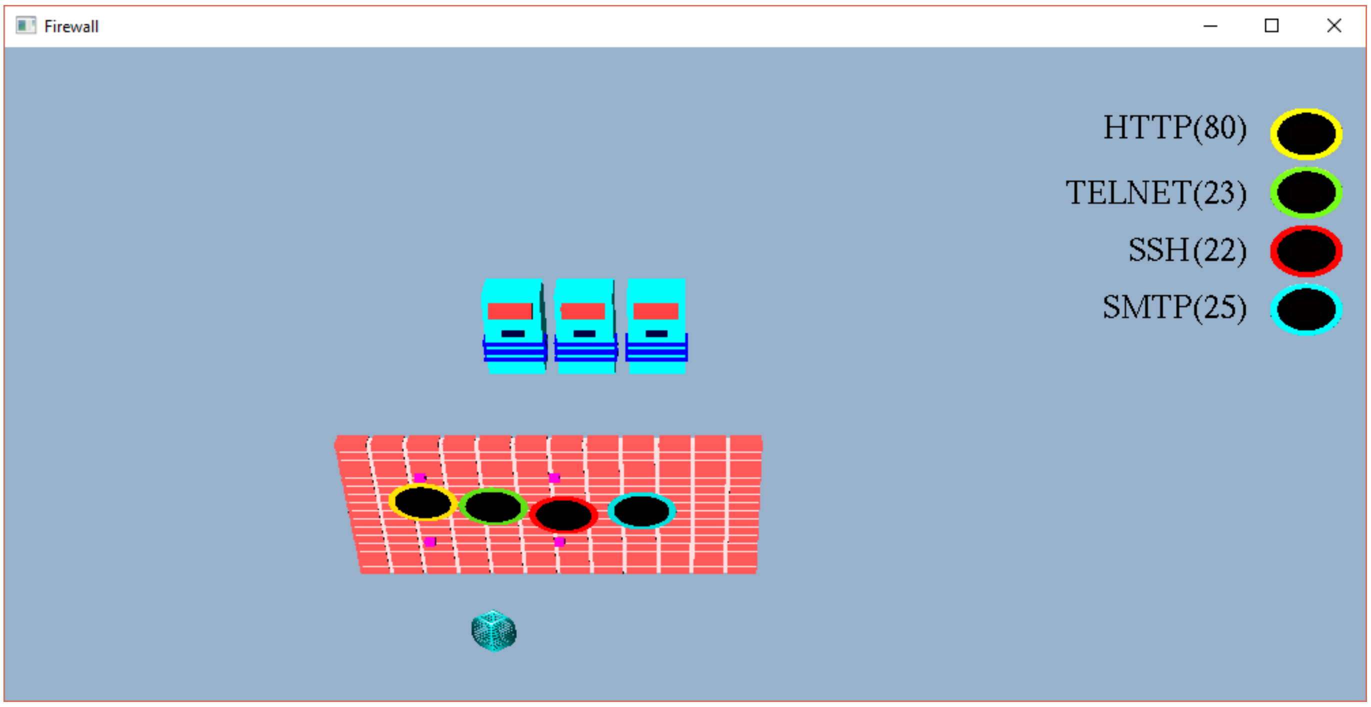
The above snapshot shows the Firewall Menu 1st is one user, 2nd option is user2, 3rd option is Quit.

6.1.3 The Firewall



The above snapshot shows user selecting telnet protocol and moving towards the firewall.

6.1.4 Firewall Output



CHAPTER 07

CONCLUSION

In conclusion, the Firewall CGV mini project aimed to develop a firewall system using the CGV (Computer Graphics Visualization) library. The project involved implementing various functionalities such as initializing the system, displaying a user interface, and performing firewall operations.

The system was designed to provide a graphical representation of a firewall, allowing users to interact with it. The code included functions for displaying various elements such as start and quit options, springs, switches, and ports. Each element was rendered using OpenGL functions to create a visual representation on the screen.

The implementation of the code involved utilizing OpenGL functions to set up the environment, apply transformations, and render the objects. The functions defined in the code performed tasks such as translating, scaling, rotating, and rendering 3D shapes.

Overall, the project successfully demonstrated the use of CGV and OpenGL in creating a firewall system with a graphical user interface. It provided a visual representation of different firewall components and allowed users to interact with them.

FUTURE ENHANCEMENTS

There are several potential future enhancements that can be implemented in the Firewall CGV mini project:

1. **Enhanced User Interface**: Improving the user interface by adding more interactive elements, intuitive controls, and visual feedback can enhance the user experience. This can include features like tooltips, status indicators, and interactive buttons for configuring firewall rules.

2. **Rule Management System:** Implementing a rule management system that allows users to add, edit, and delete firewall rules dynamically can provide more flexibility and control over network traffic filtering. This can include features like rule prioritization, rule grouping, and rule templates.

3. **Real-Time Traffic Monitoring**: Adding real-time traffic monitoring capabilities to the firewall system can provide valuable insights into network activity. This can involve displaying graphical representations of network traffic, statistics, and alerts for suspicious or unusual activity.

4. **Logging and Reporting:** Implementing a logging and reporting system that records firewall events and generates comprehensive reports can be beneficial for auditing and troubleshooting purposes. This can include features like log filtering, log visualization, and report generation.

REFERENCES

- http://www.opengl.org/
- http://www.academictutorials.com/graphics/graphics-flood-fill.asp
- https://asianlegacylibrary.org/
- https://chat.openai.com/
- https://ieeexplore.ieee.org/
- http://jerome.jouvie.free.fr/OpenGl/Lessons/Lesson3.php