Raspberry Pi Based Smart Lab For Controlling Iot Devices

1st Shashikant Thakur
Information Technology
Indian Institute of Information Technology
Prayagraj, India
IIT2020024@iiita.ac.in

2nd Nikhil Dubey
Information Technology
Indian Institute of Information Technology
Prayagraj, India
IIT2020016@iiita.ac.in

3rd Anuj Chaturvedi
Information Technology
Indian Institute of Information Technology
Prayagraj, India
IIT2020019@iiita.ac.in

4thAnoop Kumar
Information Technology
Indian Institute of Information Technology
Prayagraj, India
IIT2020020@iiita.ac.in

***Abstract*—There is nothing that makes education more engaging than a dash of technology and digitalization. In this paper, we'll talk about how to authenticate students so they may use smart devices in the Smart Lab. It is believed that a smart lab will feature a variety of IoT devices, including motion sensor-based doors and electronic devices. The Raspberry Pi will be used to authenticate pupils. A credit card-sized minicomputer called the Raspberry Pi (RPi) has many features that make it comparable to a PC. It serves as a gateway node for enrollment and verification in this study. Cryptographic techniques are applied to and are kept in an encrypted format to preserve the security of user attributes. Furthermore, the students who are connected to the switch, their device details like mac address, device type, operating system, etc will be monitored by the faculty.**

Keywords—raspberry pi, hue bridge,switch,cloud, mac address, network packets, sniffing

I. Introduction

A single board computer called the Raspberry Pi is compact. The Raspberry Pi can function as a miniature personal computer by adding peripherals like a keyboard, mouse, and display to it. Robotics applications, IoT-based applications, and real-time image/video processing are all common uses for the Raspberry Pi. Raspberry Pi is a computer that can offer all the required features or abilities at a low power consumption, abilities being slower than a laptop or desktop. With the help of the Internet of Things (IoT), which is a cutting-edge technology, objects may communicate with one another and with the rest of the world by exchanging data online and taking action on it. Although most people still consider the Internet of Things to be a relatively new notion, interconnected technology has been around for two decades. IoT consists of networked smart devices that use embedded technology to gather, store, and communicate information from their environment. Sensors, processors, and communication hardware are all components of embedded technology. Sensor data is shared among internet-connected devices and forwarded to an IoT gateway for cloud-based or local analysis. These devices also interact with one another through communication, acting on the knowledge they have learned from one another. However, in most cases, the smart device will handle all of the work in the background. Humans can engage with the devices by configuring them or retrieving acquired data. The future of human productivity at home and in the workplace is this. By automatically responding to and learning from the data they receive, IoT devices produce the highest level of ease. IoT devices are low-cost, low-power sensor technology that improve connection, give businesses easier access to infrastructure, and aid in the development of machine learning, analytics, and conversational AI. Data can be changed using cryptography methods from a readable form to a protected form and back again. Data encryption, authentication, and digital signatures are just a few significant tasks that use cryptographic methods. Steganography, hashing, simple codes, symmetric encryption, and asymmetric encryption are some prominent cryptographic techniques. Hashing is one method that is well known for maintaining message integrity. It is employed to protect extremely sensitive data and information. It gives defense against people who shouldn't have access to a message. When the sender disputes the message's transmission, digital signatures enable non-repudiation in court cases.

II. Devices Used

A. *Raspberry PI*

Raspberry Pi is a credit-card-sized computer that offers a low-cost, portable, and versatile computing solution. It runs on Linux-based operating systems and supports a wide range of programming languages. With its GPIO pins, it can interface with various electronic components, making it popular for DIY projects and educational purposes.

B. *Philips hue bridge*

The Philips Hue Bridge is a central hub that connects and controls Philips Hue smart lighting devices. It enables wireless communication between the smart bulbs and other Hue accessories, allowing users to customize lighting settings, schedule automated actions,

and control their lights remotely using the Hue mobile app or voice assistants.

C. *Philips smart bulb*

Philips smart bulbs are innovative lighting solutions that can be controlled wirelessly using the Hue Bridge. The Hue Bridge serves as a central hub, connecting the bulbs to your home network and enabling smart functionality.

D. Wi-fi switch

A Wi-Fi switch enables wireless connectivity by controlling the transmission and reception of radio signals for devices in a network. It acts as a central hub, allowing multiple devices to connect and communicate with each other over the airwaves, providing internet access and network functionality.

III. Technologies & Tools used

Before you begin to format your paper, first write and save the content as a separate text file. Complete all content and organizational editing before formatting. Please note sections A-D below for more information on proofreading, spelling and grammar.

Keep your text and graphic files separate until after the text has been formatted and styled. Do not use hard tabs, and limit use of hard returns to only one return at the end of a paragraph. Do not add any kind of pagination anywhere in the paper. Do not number text heads-the template will do that for you.

A. *Node.js*

Node.js is a powerful, open-source runtime environment that allows developers to execute JavaScript code on the server-side. It is built on Chrome's V8 JavaScript engine, enabling high-performance and scalability. Node.js provides a non-blocking, event-driven architecture, allowing for efficient handling of concurrent requests. With its extensive package ecosystem, including npm, developers can easily access and utilize a wide range of pre-built modules and libraries. Node.js has gained popularity due to its versatility, making it suitable for building various applications, such as web servers, APIs, real-time applications, and even desktop applications. It has revolutionized server-side development, enabling JavaScript to be used both in the browser and on servers.

B. *React.js*

React.js is a popular JavaScript library for building user interfaces. It follows a component-based approach, allowing developers to create reusable UI elements. React efficiently updates and renders components by using a virtual DOM, which minimizes actual DOM manipulations. It employs a declarative syntax, where developers describe how the UI should look based on the application's state, and React takes care of updating the UI when the state changes. React also supports server-side rendering, making it suitable for both client-side and server-side rendering scenarios. With its vast ecosystem and strong community support, React simplifies the development of interactive, scalable, and efficient web applications.

C. *Socket(network)*

A socket in networking is a software interface that allows programs to communicate over a network. It acts as an endpoint for sending or receiving data between different devices connected to the network. Sockets provide a standardized mechanism for applications to establish network connections, exchange data, and close connections when the communication is complete. They define the necessary protocols, such as TCP/IP or UDP, and handle details like addressing, port numbers, and data buffering. Sockets enable a wide range of network-based applications, including web browsing, file transfers, email, and real-time communication, by providing a reliable and flexible communication channel.

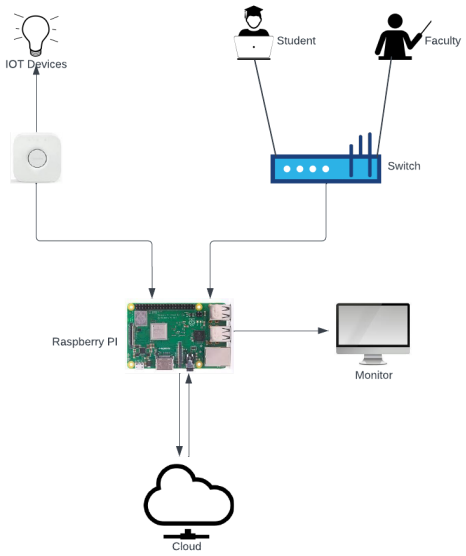
D. *Flask*

Flask is a lightweight and versatile web framework for Python, designed to create web applications quickly and efficiently. With its simplicity and flexibility, Flask empowers developers to build scalable and modular web applications. It follows the WSGI standard, allowing seamless integration with various web servers. Flask provides essential functionalities for routing, template rendering, and handling HTTP requests and responses. Its modular nature allows easy integration of extensions for database connectivity, form handling, authentication, and more. Flask embraces the concept of microservices, encouraging the development of small and self-contained applications. Its extensive documentation, active community, and vast ecosystem of plugins make Flask an excellent choice for developing web applications of any size or complexity.

IV. Proposed Architecture

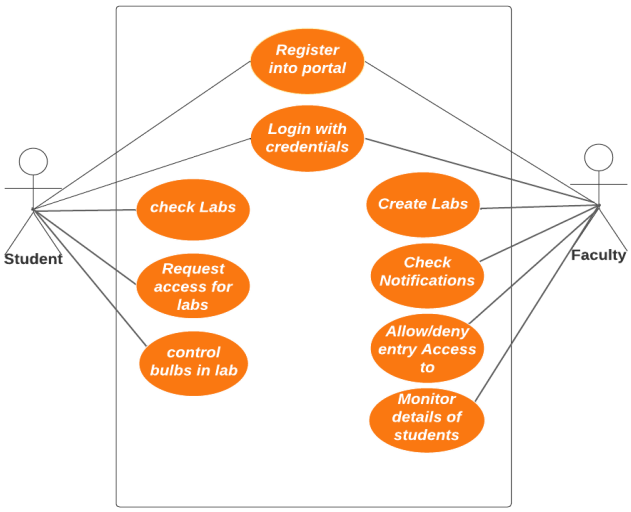
A. *Architecture diagram :*

In our architecture, we have two entities: students and faculties, who will be connected through the switch. The bulb will be connected with a hue bridge which will be connected to the raspberry pi. The bulb will be controlled with the philips hue bridge API. All the databases will be in the cloud (google cloud). Finally faculty can monitor the device details of students present in the lab.



(fig 5) Architecture Diagram of Project

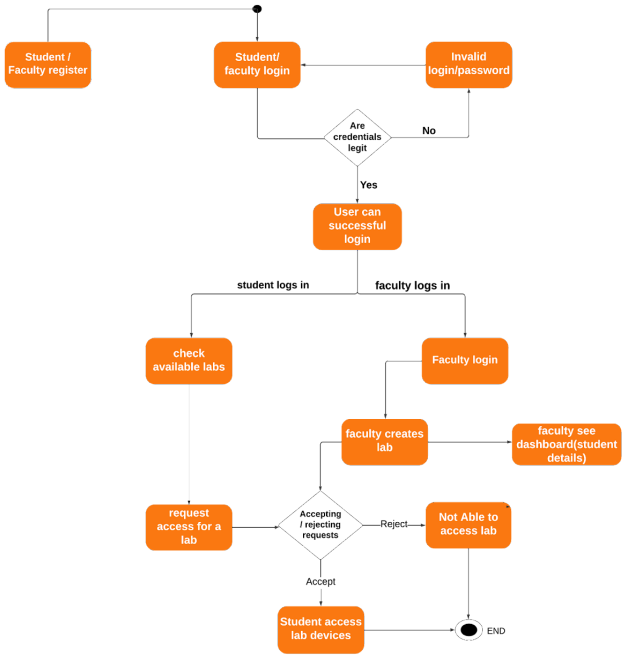
B. *Use case diagram :*



(fig 6) Use case Diagram

We have two entities student and faculty , students can perform action like register into portal , login with credentials, check labs , request access for labs ,control bulbs in the labs, and faculty can performs actions like register with phone number and password , then login with phone number and password, create a lab, check the notification , allow / deny access to the lab then finally monitoring student currently present in the lab.

C. Activity Diagram



(fig 7) Activity Diagram

- The process starts with taking the login credentials of faculty and students.
- Faculty and Students have their own signup page to register through the website.
- For login we use multiple jwt authentication(It is a token-based stateless authentication mechanism).
- After a faculty logs in the jwt token is used as an authentication mechanism for accessing all the features in the website.
- Faculty has the privilege to create a new lab and activate the lab.
- Students after registering can apply to join their respective labs faculty can reject and accept the students request.
- After activating the lab respective students and faculty have the access to control the iot devices present in the lab.

V. Security Challenges and Vulnerabilities of the IoT

The weakest parts of a system. As the number of IoT devices is rapidly growing, the resource limitations of IoT devices lead to the use of lightweight security algorithms and the security of certain devices is likely neglected. These devices become the weakest parts of an IoT network, Low control over updates. Often, users have a shallow understanding of the internal mechanisms of IoT devices and little knowledge about how to handle online updates, opening up opportunities for security attacks by various malware.

A. *Protecting passwords*

The Hue Bridge allows you to connect and control up to 50 lights and accessories. By default all connections to the Hue Bridge are done over TLS, after the negotiation of the Bridge certificate being verified to the expected format and subject contents.The Bridge certificate is self-signed, so this will cause issues when validating it normally. The library will process the certificate, validate the issuer and the subject and if happy will then allow the connection over TLS with the Hue Bridge.When using the remote API functionality of the library, the certificate is validated normally as the https://api.meethue.com site has an externally valid certificate and CA chain.There is an option to connect over HTTP using createInsecureLocal() as there are some instances of use of the library against software the pretends to be a Hue Bridge. Using this method to connect will output warnings on the console that you are connecting in an insecure way.

B. *Connecting IOT devices*

The Hue Bridge allows you to connect and control up to 50 lights and accessories.
By default all connections to the Hue Bridge are done over TLS, after the negotiation of the Bridge certificate being verified to the expected format and subject contents.
The Bridge certificate is self-signed, so this will cause issues when validating it normally. The library will process the certificate, validate the issuer and the subject and if happy will then allow the connection over TLS with the Hue Bridge.When using the remote API functionality of the library, the certificate is validated normally as the https://api.meethue.com site has an externally valid certificate and CA chain.There is an option to connect over HTTP using createInsecureLocal() as there are some instances of use of the library against software the pretends to be a Hue Bridge. Using this method to connect will output warnings on the console that you are connecting in an insecure way.

C. *Authenticating users*

A token-based stateless authentication method is JWT authentication. The server doesn't have to totally rely on a data store (or) database to save session information because it is frequently used as a client-side-based stateless session. JWTs are normally signed and encrypted, though they can be encrypted. A JSON object can be used to securely communicate data between parties utilizing the open standard known as JSON Web Token. JWT is used for stateless authentication protocols for users and providers, which allows for client-side session maintenance rather than server-side session storage.

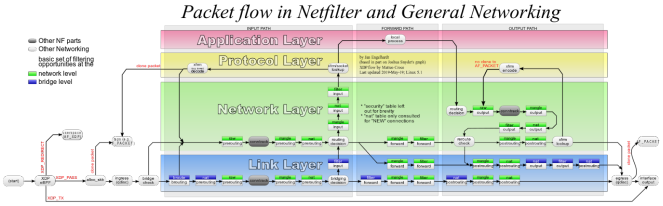
VI. Monitoring student device (connected to switch)

For monitoring student's device details who are connected to the switch/router, we have to make a packet capturing program which will capture all packets and save the information whenever the student tries to login in the portal. For this we will use a raw socket from which all the network packets will pass and get analyzed if it contains the student's login attempt flag. If the packet contains the flag, then the student's mac address along with his roll number will get saved in a data structure.
The student's credentials will be checked by the backend server of the website and if it is correct then it will send a request to our packet capture server which will provide the student's mac address of the student, which will get saved into the database for further analysis.
For handling requests from the backend server, a web server is running in the packet capturing server(on another thread), which will handle requests from the backend server.

VI(a). Packet capturing mechanism

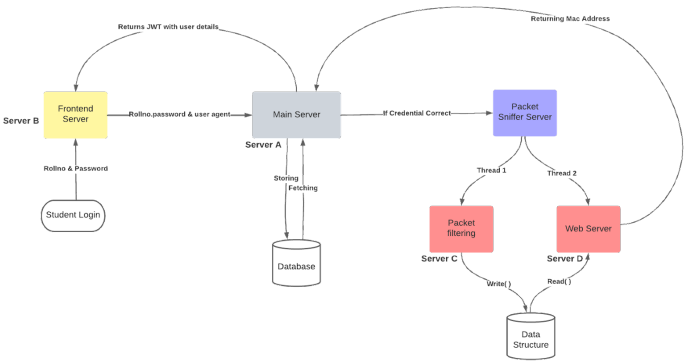
The linux kernel simply duplicates the packets as soon as it receives them from the physical layer (for incoming packets) or just before sending them out to the physical layer (for outgoing packets). One copy of each packet is sent to your socket (if we use ETH_PH_ALL then you are listening on all interfaces, but we can also bind to a particular one). After a copy is sent to your socket, the other copy then continues being processed like it normally would (e.g. identifying and decoding the protocol, checking firewall rules, etc).
Here fig 8. shows flow of packet inside linux system or netfilter. Netfilter is a framework provided by the Linux kernel that allows various networking-related operations to be implemented in the form of customized handlers.
We have used AF_PACKETS which will clone all packets passing through the kernel.

Packet flow in Netfilter and General Networking



(fig 8) flow of packets in linux system

VI(B). MAC ADDRESS CAPTURING :



(fig 9) mechanism for capturing mac address

Methodology for mapping student mac address with student roll no.

- Making a packet sniffer and web server .
- first thread :
    - Capturing packets and filtering tcp packets containing login information.
    - Storing login information along with mac address in a data structure.

- second thread :
    - Creating a web server for providing mac address information for successfully login students.

Monitoring student mac address(connected to switch):
We have total 4 servers running,
Server A- Backend server(Node.js) : This server receives requests from the frontend (user interface) and communicates with databases, APIs, and other external systems to deliver the requested information or perform necessary operations.

Server B- Frontend server(React.js): server is responsible for handling client requests and delivering web content to users' browsers. It processes user input, retrieves data from backend servers, and generates HTML, CSS, and JavaScript code to be rendered on the client side, enabling interactive and user-friendly web experiences. If a student logins then a login flag is raised which will pass through server A and server C.

Server C- Packet Sniffer: It intercepts data packets flowing across the network, allowing us to monitor and analyze network activities, including capturing and inspecting the contents of individual packets for getting the mac address of students. If this server receives a login flag in a packet, then it stores the corresponding MAC address with roll no. in a data structure.

Server D- HTTP Server :This server handles the request from server B about student roll no and returns back the MAC address of the corresponding student.

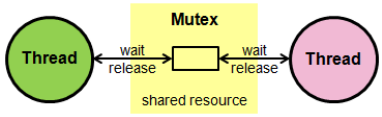
When student enters his login information, server A check the credentials, and if it is correct it sends request to server D for the MAC address of the students, server D reads MAC address corresponding to that roll no and returns to that server A. Server A stores that MAC address into the database for further analysis.

VI(C).PROBLEM OF SYNCHRONIZATION

If server D is reading from the data structure and at the same time, if server C is changing the data, then a problem of synchronization will arise. To avoid this we use mutex() lock. A mutex, short for mutual exclusion, is a locking mechanism that ensures only one thread can access the shared data at a time. When a thread wants to access the data structure, it acquires the mutex lock, performs its operations, and releases the lock when it's done. This prevents race conditions and data corruption caused by multiple threads accessing and modifying the shared data structure simultaneously.

Working of mutex:

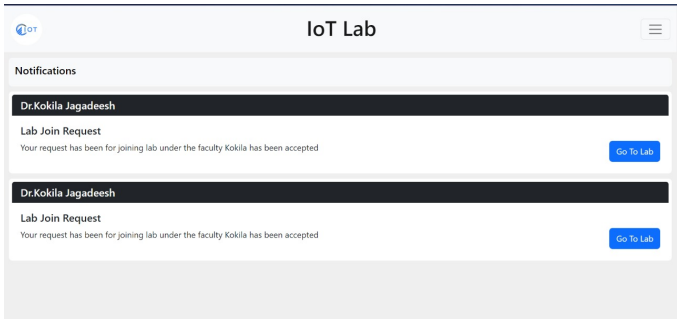
- Initialization: A mutex is created and initialized. This involves allocating memory for the mutex object and setting its initial state.
- Locking: When a thread wants to access a shared resource, it first requests a lock on the mutex. If the mutex is currently unlocked, the thread acquires the lock and continues executing. If the mutex is already locked by another thread, the requesting thread is typically blocked or put to sleep until the mutex becomes available.
- Unlocking: When a thread finishes using the shared resource, it releases the lock on the mutex, allowing other threads to acquire it. This step is crucial to ensure that other threads waiting for the resource can proceed.
- Thread Synchronization: The mutex provides synchronization by ensuring that only one thread can hold the lock at a time. This guarantees that the critical section of code, the part where the shared resource is accessed, is executed exclusively and prevents simultaneous access by multiple threads.



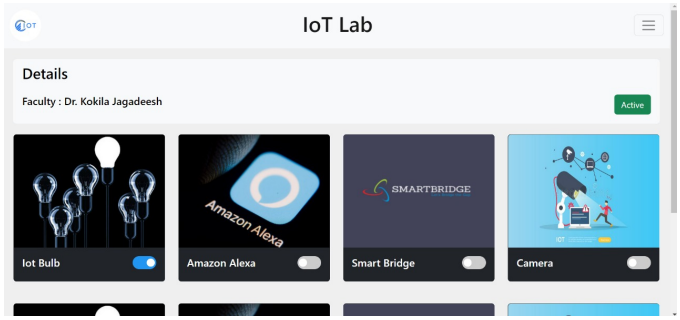
fig(10) working of mutex

VII. Results

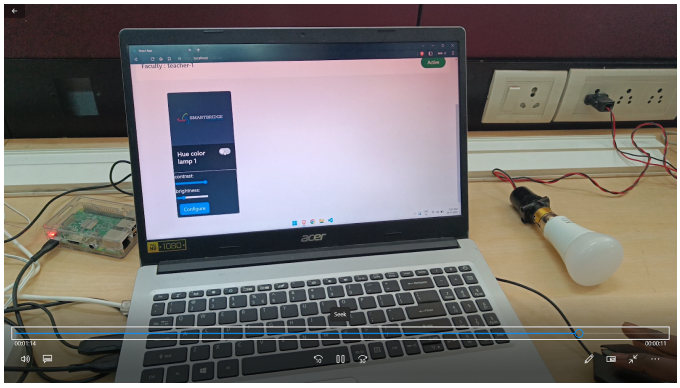
(fig 11) Homepage of portal



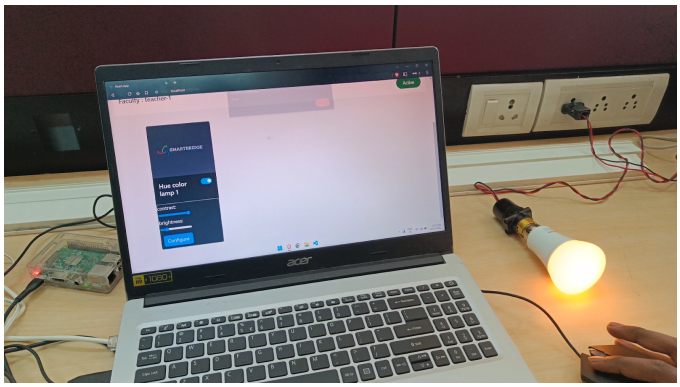
(fig 12) notifications page



(fig 13) lab overview



(fig 14) bulb in off state



(fig 15) bulb in on state

(fig 16) student mac addresses

Acknowledgment

We would like to acknowledge the contributions and support of numerous individuals who have made this research paper possible. Firstly, We are deeply grateful to our supervisor Dr. J. Kokila mam and Dr. Abhishek Vaish Sir, whose guidance, expertise, and invaluable feedback have shaped this work. Additionally, We extend our gratitude to our colleagues and friends for their encouragement and thoughtful discussions throughout the research process. Their contribution has been instrumental in the successful completion of this research.

References

[1] André Glória, Francisco Cercas, Nuno Souto, Design and and implementation of an IoT gateway to create smart environments(2017)

[2] Frank Mueller1, A Library Implementation of POSIX Threads under UNIX , Florida State University.

[3] Olaniyi A. Ayeni, Elijah Oyekunle, Omoyele A. Odeniyi, Design and Implementation of a Packet Sniffing Library, School of Computing, Federal University of Technology, Akure, Ondo State. Nigeria

[4] Leslie F. Sikos, Packet analysis for network forensics: A comprehensive survey, Edith Cowan University, Australia

[5] Dr. Charu Gandhi1, Gaurav Suri2, Rishi P. Golyan3, Pupul Saxena4, Bhavya K. Saxena5, Packet Sniffer – A Comparative Study, Department of computer science, JIIT, Noida-201307

[6] Jens Heuschkel, Tobias Hofmann, Thorsten Hollstein, Joel Kuepper, Introduction to RAW-sockets

[7] Blaise Barney, POSIX Threads Programming, Lawrence Livermore National Laboratory

[8] https://man7.org/linux/man-pages/man7/raw.7.html

[9] https://inc0x0.com/tcp-ip-packets-introduction/tcp-ip-packets-3-manually-create-and-send-raw-tcp-ip-packets/