     TWO WAY AUTHENTICATION USING RASPBERRY PI

### A PROJECT REPORT

#### in partial fulfillment for the award of the degree of

# BACHELOR OF ENGINEERING

**IN**

**COMPUTER SCIENCE WITH SPECIALIZATION IN INFORMATION SECURITY**

#### Submitted by

#### MENDA MANMADHA RAO -21BCS3544

#### PENTAKOTA SRI PRANEETH - 21BCS3523

#### ARNAV GUPTA - 21BCS10946

**Under The Supervision of**

VIJAY BHARDWAJ (E12849)



## **CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413, PUNJAB**

**November, 2023**

**TWO WAY AUTHENTICATION USING RASPBERRY PI**

### A PROJECT REPORT

#### Submitted by

#### MENDA MANMADHA RAO -21BCS3544

#### PENTAKOTA SRI PRANEETH - 21BCS3523

ARNAV GUPTA - 21BCS10946

#### in partial fulfillment for the award of the degree of

**BACHELOR OF ENGINEERING**

###### IN

**COMPUTER SCIENCE WITH SPECIALIZATION IN**

**INFORMATION SECURITY**



### Chandigarh University

**NOVEMBER 2023**



**BONAFIDE CERTIFICATE**

Certified this project report “TWO WAY AUTHENTICATION USING

**RASPBERRY PI”**is the Bonafide work of “Menda Manmadha rao, Sri Praneeth, Arnav Gupta” who carried out the project work under my/our supervision.

###### 

###### SIGNATURE OF THE

###### HEAD OF THE DEPARTMENT

(**AIT-CSE)**

###### SIGNATURE OF THE

###### SUPERVISOR

(**AIT-CSE)**

Submitted for the project viva-voce examination held on

INTERNAL EXAMINER EXTERNAL EXAMINER

**TABLE OF CONTENTS**

Abstract

Graphical Abstract

Abbreviations

Symbols

Chapter 1: Introduction

### 1.1 Problem Definition

### 1.2 Problem Overview

### 1.3 Hardware Specification

### 1.4 Software Specification

### Chapter 2: Literature Survey

### 2.1 Existing System

### 2.2 Proposed System

### 2.3 Literature Review Summery

### Chapter 3. Design/Flow Process

### 3.1 System Architecture

### 3.2 Problem Formulation

### 3.3 Experiments

### Chapter 4: Results Analysis and Validation

### Chapter 5: Conclusion and Future work

### 5.1 Conclusion

### 5.2 Future work

### Chapter 6: References

# ABSTRACT

# As the digital landscape evolves, protecting the security of sensitive data becomes increasingly important. Two-factor authentication (2FA) has developed as a reliable technique for improving the security of user accounts and systems. The implementation of two-factor authentication is investigated in this study utilizing the Raspberry Pi, a versatile and cost-effective single-board computer. The proposed approach strengthens the authentication process by combining the strengths of something the user knows (password) and something the user owns (Raspberry Pi).

# The Raspberry Pi's capabilities are used by the two-factor authentication system to generate time-based one-time passwords (TOTPs) using the Time-based One-Time Password Algorithm (TOTP). This algorithm generates a unique password that changes on a regular basis, often every 30 seconds, adding an extra layer of security against unwanted access. The Raspberry Pi acts as a secure token generator, ensuring that only the legitimate user, in possession of the physical device, can successfully complete the authentication process.

# The system architecture encompasses multiple components, including a user interface for password entry, a Raspberry Pi module for TOTP generation, and a secure communication channel between the user device and the Raspberry Pi.

# The communication is established using industry-standard encryption protocols to safeguard against eavesdropping and man-in-the-middle attacks.

# The design also considers the importance of user experience, with seamless integration of the 2FA process into existing authentication workflows.

# One of the key advantages of employing Raspberry Pi for 2FA is its affordability and accessibility.

# Raspberry Pi offers a low-cost alternative to dedicated hardware tokens, making it an attractive option for small to medium-sized enterprises and individual users.

# The open-source nature of Raspberry Pi allows for flexible customization, allowing organizations to tailor their 2FA implementation to their specific security needs.

# This paper describes the technical details of the TOTP generation process on the Raspberry Pi and highlights the importance of secure key storage and regular key rotation to reduce potential security risks.

# Additionally, methods are being investigated to make the system more resistant to physical manipulation and ensure the integrity of the authentication process.

# A comprehensive evaluation of the proposed system includes performance benchmarks, security evaluations, and user feedback.

# The effectiveness of the system in defending against common attack vectors such as brute force attacks and phishing attacks is examined in detail.

# User acceptance and usability studies provide insight into the practicality and ease of use of Raspberry Pi-based 2FA systems in real-world scenarios.

# In summary, this paper presents a robust and cost-effective approach to implementing his two-factor authentication using the Raspberry Pi.

# The proposed system leverages the strengths of hardware and software elements to create a secure authentication process.

# Raspberry Pi's affordability, accessibility, and flexibility make it an attractive choice for organizations and individuals looking to increase the security of their digital assets through two-factor authentication.

# Problem Definition

# With the proliferation of internet-connected devices and services, identity theft and online account compromises have become increasingly common. Cybercriminals employ sophisticated techniques like phishing, malware, credential stuffing and brute force attacks to steal sensitive information like usernames, passwords and payment details. Traditional single-factor authentication relying only on passwords is inadequate to thwart such advanced threats. Even complex passwords can be cracked using tools or guessed during phishing. This has created an urgent need for multi-layered authentication approaches that can protect user identities and accounts more robustly without inconveniencing legitimate users.

# Problem Overview

# The inability of single-factor password-only authentication methods to prevent unauthorized access in today's complex cyber threat landscape poses serious problems. Financial losses 5 due to identity theft run into millions while data breaches erode user trust in online platforms. Compromised accounts are also misused to cause secondary damage via activities like spreading malware or scamming contacts. While multi-factor authentication provides stronger security, existing commercial solutions can be expensive and inaccessible for some. This project aims to address this problem through development of an affordable yet robust two-factor system leveraging open-source tools. By integrating SMS OTPs delivered by a Raspberry Pi server, it seeks to provide an additional layer of security beyond passwords during login at minimal cost. This could help both individuals and organizations fortify their online defence through a practical alternative.

# Hardware Specification

# The main hardware components are - Raspberry Pi 3/4 Model B single board computer, 8GB or higher microSD card, USB power adapter, Ethernet cable

# Software Specification

# The software tools to be used are - Raspbian GNU/Linux OS, Python 3 programming language, MySQL database to store user credentials, Twilio API for SMS integration.

# GRAPHICAL ABSTRACT

# This detailed guide is intended to provide a comprehensive understanding of the visual and textual components of a graphical overview of a research paper on two-factor authentication using Raspberry Pi.

# 1.Title and Author: The title is the focal point, so font, size, and color should be carefully considered.

# Font selection should be based on modern design principles, as readability is very important.

# For a clean, modern look, consider using a sans-serif font.

# The title should be big, bold, and grab attention immediately.

# Author's name and affiliation are after the title.

# Add a unique symbol to each author's name to highlight their role.

# For example, the first author can be represented by a lock icon to symbolize leadership.

# Contributors with hardware knowledge can be identified by her Raspberry Pi icon, which highlights specific contributions.

# 2. Central Concept: The central concept visually represents the essence of the research.

# Combining the Raspberry Pi logo with a lock or key symbol creates a powerful and recognizable image.

# Consider using a color gradient to evoke a sense of heightened security, with darker shades conveying robust protection.

# The central concept must be large enough to capture attention and convey the primary focus of the research.

# It should be strategically positioned to serve as the anchor point, around which other elements will revolve.

# 

# 3. System Architecture: Creating a detailed system architecture diagram is important to convey the complexity of the proposed solution.

# Use different symbols or icons for major components.

# The lock represents security and can represent the Raspberry Pi symbol of a single board computer, a cloud or a network symbol of a communication channel.

# The arrows connecting these elements indicate the flow of information.

# Color should be used strategically to differentiate components and increase clarity.

# Annotations provide additional context and ensure that the viewer understands the relationships between different parts of the system.

# 

# 4. Authentication Process: The authentication process is best conveyed through a visual

# sequence.

# Start with an icon that represents the user interacting with the device and entering a password.

# Mark this step with clear visual cues, such as a password entry field or a lock icon.

# Proceed to the next step with a visual representation of how the Raspberry Pi generates a time-based one-time password (TOTP).

# This can be represented by a digital clock icon that emphasizes the time-based nature of the authentication process.

# Completes the sequence with an icon indicating successful authentication.

# B.

# A green check mark or unlocked padlock.

# This visual narrative helps viewers understand the dynamic and sequential nature of authentication methods.

# 

# 5. Security Measures: Security measures are vital components of the graphical abstract.

# Use icons or symbols to represent encryption, secure key storage, and periodic key rotation.

# A shield can symbolize encryption, a secure vault for key storage, and a rotating key for periodic key rotation.

# Arrange these symbols in a visually cohesive manner, forming a protective perimeter around the central concept.

# This not only emphasizes the robust security measures in place but also guides the viewer's attention towards critical aspects of the system.

# 

# 6. Affordability and Accessibility: For this section, a detailed comparison chart using icons and symbols can effectively highlight the affordability and accessibility of Raspberry Pi compared to traditional hardware tokens.

# A color-coded system with green checkmarks for advantages and red crosses for disadvantages can provide a quick visual summary.

# Accompany the chart with a textual explanation, ensuring that viewers can easily comprehend the information.

# Use a Raspberry Pi icon with a dollar sign to indicate affordable prices, and a hardware token with a price tag to indicate high costs.

# 7. Customization: To demonstrate the flexibility of the Raspberry Pi, we need a visual representation of the customization options.

# If code snippets are part of your customization process, display them in a clean and organized manner.

# Use evenly-spaced fonts for clarity and consistency.

# If you use a graphical user interface for customization, please include screenshots with annotations highlighting key features.

# The purpose of this section is to visually convey the adaptability and ease of use of the Raspberry Pi platform.

# 

# 8. Performance Evaluation: Integrating charts or graphs into graphical summaries is essential for visualizing performance benchmarks and security assessments.

# A bar graph can effectively show the efficiency of a system compared to common attack vectors, with different bars representing different scenarios.

# Line graphs illustrate the security of a system over time and can show the system's resilience to evolving threats.

# Include concise captions and labels to help your audience quickly understand what each data point means.

# 

# 9. User Experience: Incorporate visuals that represent a positive user experience.

# Screenshots of the user interface during the verification process must be clear, high-resolution, and properly labeled.

# Convey positive feedback to your users by strategically incorporating icons and emojis.

# For example, you can place a smiley face or thumbs up icon next to a user-friendly interface element to emphasize the positive aspects of the user experience.

# Consider adding captions or annotations that describe specific interface features that contribute to a positive user experience.

# The purpose of this section is to highlight the practicality and ease of use of the proposed authentication system.

# 10. Experimental Work:

# Accessing your raspberry-pi via SSH:

# A lot of people use a Raspberry Pi at home as a file, or media, server. This is has become rather common with the launch of Raspberry Pi 4, which has both USB 3 and Gigabit Ethernet. However, when you’re setting up this sort of server you often want to run it “headless”; without a monitor, keyboard, or mouse. This is especially true if you intend tuck your Raspberry Pi away behind your television, or somewhere else out of the way. In any case, it means that you are going to need to enable Secure Shell (SSH) for remote access.

# However, it’s also pretty common to set up your server so that you can access your files when you’re away from home, making your Raspberry Pi accessible from the Internet.

# Most of us aren’t going to be out of the house much for a while yet, but if you’re taking the time right now to build a file server, you might want to think about adding some extra security. Especially if you intend to make the server accessible from the Internet, you probably want to enable two-factor authentication (2FA) using Time-based One-Time Password (TOTP).

# Updating the raspberry pi operating system:

# The first thing you should do is make sure your Raspberry Pi is up to date with the latest version of Raspbian. If you’re running a relatively recent version of the Operating-system you can do that from the command line:

# $ sudo apt-get update

# $ sudo apt-get full-upgrade

# If you’re pulling your Raspberry Pi out of a drawer for the first time in a while, though, you might want to go as far as to install a new copy of Raspbian using the new Raspberry Pi Imager, so you know you’re working from a good image.

# 

# Enabling secure shell:

# The Raspbian operating system has the SSH server disabled on boot. However, since we’re intending to run the board without a monitor or keyboard, we need to enable it if we want to be able to SSH into our Raspberry Pi.

# The easiest way to enable SSH is from the desktop. Go to the Raspbian menu and select “Preferences > Raspberry Pi Configuration”. Next, select the “Interfaces” tab and click on the radio button to enable SSH, then hit “OK.”

# You can also enable it from the command line using systemctl:

# $ sudo systemctl enable ssh

# $ sudo systemctl start ssh

# 

# Alternatively, you can enable SSH using raspi-config, or, if you’re installing the operating system for the first time, you can enable SSH as you burn your SD Card.

# Enabling challenge response:

# Next, we need to tell the SSH daemon to enable “challenge-response” passwords. Go ahead and open the SSH config file:

# $ sudo nano /etc/ssh/sshd\_config

# 

# Enable challenge response by changing Challenge Response Authentication from the default no to yes.

# Then restart the SSH daemon:

# $ sudo systemctl restart ssh

# It’s good idea to open up a terminal on your laptop and make sure you can still SSH into your Raspberry Pi at this point — although you won’t be prompted for a 2FA code quite yet. It’s sensible to check that everything still works at this stage.

# 

# Installing two factor authentication:

# The first thing you need to do is download an app to your phone that will generate the TOTP. One of the most commonly used is Google Authenticator. It’s available for Android, iOS, and Blackberry, and there is even an open source version of the app available on GitHub.

# 

# Google Authenticator in the App Store.

# So go ahead and install Google Authenticator, or another 2FA app like Authy, on your phone. Afterwards, install the Google Authenticator PAM module on your Raspberry Pi:

# $ sudo apt install libpam-google-authenticator

# 

# Now we have 2FA installed on both our phone, and our Raspberry Pi, we’re ready to get things configured.

# Configuring two-way authentication:

# You should now run Google Authenticator from the command line — without using

# sudo — on your Raspberry Pi in order to generate a QR code:

# $ google-authenticator

# 

# Afterwards you’re probably going to have to resize the Terminal window so that the QR code is rendered correctly. Unfortunately, it’s just slightly wider than the standard 80 characters across.

# 

# Don’t move forward quite yet! Before you do anything else you should copy the emergency codes and put them somewhere safe.

# <https://www.google.com/chart?chs=200x200&chld=M|0&cht=qr&chl=otpauth://totp/manmadha@raspberry%3Fsecret%3DCDMFGL6WADSGUGC4YGP2NS7JQA%26issuer%3Draspberry>

# 

# The scanner will provided by the raspberry pi in google authentication , so we need to scan this QR code using a Google authenticator app.

# This Qr code can give the access to allowing a raspberry-pi in google authentication.

# 

# These codes will let you access your Raspberry Pi — and turn off 2FA — if you lose your phone. Without them, you won’t be able to SSH into your Raspberry Pi if you lose or break the device you’re using to authenticate.

# 

# Next, before we continue with Google Authenticator on the Raspberry Pi, open the Google Authenticator app on your phone and tap the plus sign (+) at the top right, then tap on “Scan barcode.”

# Your phone will ask you whether you want to allow the app access to your camera; you should say “Yes.” The camera view will open. Position the barcode squarely in the green box on the screen.

# 

# As soon as your phone app recognises the QR code it will add your new account, and it will start generating TOTP codes automatically.

# 

# These are the codes, when we enter the code from google authentication app they can generate the eight digit of five codes.

# Your phone will generate a new one-time password every thirty seconds. However, this code isn’t going to be all that useful until we finish what we were doing on your Raspberry Pi. Switch back to your terminal window and answer “Y” when asked whether Google Authenticator should update your .google\_authenticator file.

# Then answer “Y” to disallow multiple uses of the same authentication token, “N” to increasing the time skew window, and “Y” to rate limiting in order to protect against brute-force attacks.

# 

# You’re done here. Now all we have to do is enable 2FA.

# Enabling two factor authentication:

# We’re going to use Linux Pluggable Authentication Modules (PAM), which provides dynamic authentication support for applications and services, to add 2FA to SSH on Raspberry Pi.

# Now we need to configure PAM to add 2FA:

# $ sudo nano /etc/pam.d/sshd

# 

# Add auth required pam\_google\_authenticator.so to the top of the file. You can do this either above or below the line that says @include common-auth.

# 

# As I prefer to be prompted for my verification code after entering my password, I’ve added this line after the @include line. If you want to be prompted for the code before entering your password you should add it before the @include line.

# 

# Now restart the SSH daemon:

# $ sudo systemctl restart ssh

# 

# Next, open up a terminal window on your laptop and try and SSH into your Raspberry Pi.

# If everything has gone to plan, when you SSH into the Raspberry Pi, you should be prompted for a TOTP after being prompted for your password.

# SSH’ing into my Raspberry Pi.

# You should go ahead and open Google Authenticator on your phone, and enter the six-digit code when prompted. Then you should be logged into your Raspberry Pi as normal.

# You’ll now need your phone, and a TOTP, every time you ssh into, or scp to and from, your Raspberry Pi. But because of that, you’ve just given a huge boost to the security of your device.

# Now you have the Google Authenticator app on your phone, you should probably start enabling 2FA for your important services and sites — like Google, Twitter, Amazon, and others — since most bigger sites, and many smaller ones, now support two-factor authentication.

# 11. Conclusion: The conclusion part serves to summarize the most important findings and

# advantages of the proposed system in a visually appealing manner.

# You can use icons and symbols that reflect the key points of the deal, such as a trophy to represent an accomplishment or a handshake to represent a successful integration.

# Make sure the visual elements in this section match the overall theme and color scheme of the graphic overview.

# The conclusion should leave a lasting visual impression and highlight the positive results and contributions of the research.

# In summary, a detailed graphic summary should effectively combine visual elements with concise, informative text to convey the core concepts, methods, and results of the study.

# The breakdown shown above provides a comprehensive guide to each element, ensuring the graphical overview is both visually appealing and intellectually informative.

**ABBREVIATIONS**

**Abbreviations in raspberry pi:**

Raspberry Pi is a versatile single-board computer that has gained popularity for various applications. Here are some common abbreviations associated with Raspberry Pi:

**RPi or Pi**: Abbreviation for Raspberry Pi.

**GPIO:** General Purpose Input/Output. GPIO pins on the Raspberry Pi allow it to interact with the physical world by sending and receiving digital signals.

**USB:** Universal Serial Bus. Used for connecting peripherals such as keyboards, mice, and other devices to the Raspberry Pi.

**HDMI:** High-Definition Multimedia Interface. Used for connecting the Raspberry Pi to displays or TVs.

**SD CARD**: Secure Digital card. The primary storage medium for the Raspberry Pi's operating system and data.

**OS:** Operating System. The Raspberry Pi can run various operating systems, including Raspbian (now called Raspberry Pi OS), Ubuntu, and others.

**SSH:** Secure Shell. A protocol used for secure remote access to the Raspberry command line.

**VNC:** Virtual Network Computing. A graphical desktop-sharing system that allows you to remotely control the desktop interface of the Raspberry Pi.

**Wi-Fi:** Wireless Fidelity. The Raspberry Pi can connect to Wi-Fi networks using a USB Wi-Fi dongle or built-in Wi-Fi on certain models.

**LAN:** Local Area Network. The network that the Raspberry Pi can connect to, either through Ethernet (wired) or Wi-Fi (wireless).

**SBC**: Single-Board Computer. Raspberry Pi is an example of an SBC, as it integrates all the components of a computer onto a single circuit board.

**SoC:** System on a Chip. The Broadcom BCM2835 or BCM2711 chip on the Raspberry Pi contains the CPU, GPU, and other components.

**RPi Foundation**: Raspberry Pi Foundation. The organization behind the development and promotion of the Raspberry Pi.

**Raspbian**: The former name of the official Raspberry Pi operating system, now called Raspberry Pi OS.

**NOOBS:** New Out Of the Box Software. A beginner-friendly operating system installation manager for the Raspberry Pi.

**SSH:** Secure Shell. A cryptographic network protocol used for secure remote login and other secure network services over an unsecured network.

**RTC:** Real-Time Clock. Some Raspberry Pi models do not have a built-in real-time clock, and an external RTC module may be added to keep track of time when the device is powered off.

# Abbreviations in two way authentication :

# Two-factor authentication (2FA) is a security process in which a user provides two different authentication factors to verify their identity Here are some common abbreviations related to two-factor authentication:

# 2FA: Two-Factor Authentication. The overall term for the authentication process that involves two different factors.

# OTP: One-Time Password. A password that is valid for only one login session or transaction on a computer system or other digital devices.

# SMS: Short Message Service. In the context of 2FA, it refers to the delivery of authentication codes via text messages.

# TOTP: Time-Based One-Time Password. A type of OTP that is valid for a short and predefined amount of time, often generated by apps like Google Authenticator or Authy.

# HOTP: HMAC-based One-Time Password. Similar to TOTP, but the passwords are generated based on a counter and a secret key using HMAC (Hash-based Message Authentication Code).

# U2F: Universal 2nd Factor. A standard for two-factor authentication using specialized security keys, such as USB or NFC devices.

# MFA: Multi-Factor Authentication. A broader term that encompasses any authentication method that uses two or more verification factors.

# Biometrics: The use of physical or behavioral characteristics, such as fingerprints, facial recognition, or iris scans, as a form of authentication.

# App-based 2FA: Two-factor authentication that relies on a mobile app, like Google Authenticator or Authy, to generate OTPs.

# Push Authentication: A method where a notification is sent to a mobile app, and the user can approve or deny the login attempt.

# Backup Codes: Pre-generated codes that can be used as a backup method for authentication in case the primary methods are unavailable.

# Security Key: A physical device, often in the form of a USB key, that is used for authentication.

# Biometric Authentication: The use of unique physical or behavioral attributes, such as fingerprints, facial recognition, or voice recognition, for authentication.

# Duo: Refers to Duo Security, a popular provider of two-factor authentication services.

# RSA: While RSA can refer to the encryption algorithm, in the context of 2FA, it can also refer to the RSA SecurID system, which uses tokens to generate one-time passwords.

# Abstractions in two way authentication using raspberry pi :

# When implementing two-factor authentication (2FA) using Raspberry Pi, various abstractions and components are involved in the process. Here's an abstract breakdown of the key elements:

# Raspberry Pi (RPi): The central computing device responsible for handling the authentication process. It runs the necessary software and interacts with the user and authentication factors.

# User Interface (UI): The interface through which the user interacts with the Raspberry Pi during the authentication process. This could be a command-line interface, a graphical user interface (GUI), or even a web interface.

# Authentication Factors:

# In the context of security, there are two factors that can be used to authenticate a user: something they have and something they are. "Something you have" refers to a physical device like a mobile phone or a security token. For instance, in the case of Raspberry Pi, you could use USB security keys, RFID cards, or other similar physical tokens as a means of authentication. On the other hand, "something you are" refers to biometric information, such as fingerprints or facial recognition, that can be incorporated if the Raspberry Pi has the necessary hardware. In the context of security, there are two factors that can be used to authenticate a user: something they have and something they are. "Something you have" refers to a physical device like a mobile phone or a security token. For instance, in the case of Raspberry Pi, you could use USB security keys, RFID cards, or other similar physical tokens as a means of authentication. On the other hand, "something you are" refers to biometric information, such as fingerprints or facial recognition, that can be incorporated if the Raspberry Pi has the necessary hardware.2FA Logic/Controller: The logic or controller responsible for coordinating the 2FA process. It manages the interaction between the Raspberry Pi, user interface, and authentication factors.

# OTP Generator: If using one-time passwords (OTPs) as one of the factors, there may be a component responsible for generating time-based or event-based OTPs. This could be implemented using libraries or applications like TOTP (Time-Based One-Time Password) generators.

# Communication Module: A component responsible for communication with external devices or services. For example, sending OTPs via SMS, handling push notifications, or communicating with external authentication servers.

# Security Module: Handles the encryption and secure storage of sensitive information, such as secret keys for OTP generation or user credentials.

# Authentication Server (Optional): In some setups, a dedicated authentication server may be involved, especially if the 2FA process extends beyond the local Raspberry Pi to authenticate against external services.

# Logging and Auditing: A module responsible for logging authentication events and auditing activities for security purposes.

# Error Handling: A component that manages errors and exceptions during the authentication process, providing feedback to the user and maintaining the security of the

# Configuration Management: A module allowing the user to configure and manage various aspects of the 2FA setup, such as adding or removing authentication factors.

# Abbreviations in one time password :

# When it comes to One-Time Passwords (OTPs), various abbreviations are commonly used in the context of authentication and security. Here are some key abbreviations related to One-Time Passwords:

# OTP: One-Time Password. A password that is valid for a single login session or transaction on a computer system or other digital devices.

# TOTP: Time-Based One-Time Password. A type of OTP that is valid for a short and predefined amount of time, often generated by apps like Google Authenticator or Authy.

# HOTP: HMAC-based One-Time Password. Similar to TOTP, but the passwords are generated based on a counter and a secret key using HMAC (Hash-based Message Authentication Code).

# SMS OTP: One-Time Password delivered via Short Message Service. This involves receiving the OTP as a text message on a mobile device.

# PIN: Personal Identification Number. Sometimes used interchangeably with OTP, especially in cases where the user needs to enter a numeric code for authentication.

# 2FA: Two-Factor Authentication. OTPs are often used as one of the factors in a two-factor authentication setup.

# MFA: Multi-Factor Authentication. OTPs may be one of multiple factors used for authentication in a multi-factor setup.

# U2F: Universal 2nd Factor. While not directly related to OTPs, it's a standard for two-factor authentication using specialized security keys for a second factor.

# Backup Code: A pre-generated code that can be used as a backup method for authentication in case the primary OTP method is unavailable.

# QR Code: Quick Response Code. Used to easily set up OTPs in applications like Google Authenticator by scanning a QR code that contains the necessary information.

# Soft Token: Refers to software-based implementations of OTP generators, typically in the form of mobile apps like Google Authenticator or Authy.

# Hard Token: A physical device, like a key fob or smart card, that generates OTPs for authentication.

# OATH: Initiative for Open Authentication. An organization that has developed standards, including TOTP and HOTP, for OTPs.

# RFC: Request for Comments. Documents that describe various Internet-related protocols and standards. The standards for TOTP and HOTP are defined in RFC documents.

# Abbreviations in ssh using Raspberry Pi:

# When working with SSH (Secure Shell) on a Raspberry Pi or any other system, you may encounter several abbreviations and commands. Here are some common ones specific to SSH on a Raspberry Pi:

# SSH: Secure Shell (the protocol used for secure remote access).

# Raspberry Pi OS: The operating system designed for Raspberry Pi.

# IP: Internet Protocol (referring to an IP address).

# DNS: Domain Name System (resolving domain names to IP addresses).

# SSH Key: A cryptographic key pair used for secure authentication.

# Pi: Short for Raspberry Pi.

# GPIO: General Purpose Input/Output (referring to the GPIO pins on a Raspberry Pi).

# VNC: Virtual Network Computing (used for remote desktop access on a Raspberry Pi).

# Sudo: Superuser does (used to execute commands with elevated privileges).

# RPi: Abbreviation for Raspberry Pi.

# USB: Universal Serial Bus (commonly used for connecting peripherals to a Raspberry Pi).

# LAN: Local Area Network (referring to the network within a specific geographical area).

# WiFi: Wireless Fidelity (referring to wireless network connectivity).

# SSID: Service Set Identifier (the name of a wireless network).

# PSU: Power Supply Unit (providing power to the Raspberry Pi).

# DHCP: Dynamic Host Configuration Protocol (automatically assigns IP addresses on a network).

# NAT: Network Address Translation (translating private IP addresses to public ones).

# AP: Access Point (a device that allows Wi-Fi devices to connect to a wired network).

# WPA/WPA2/WPA3: Wi-Fi Protected Access (security protocols for wireless networks).

# LAN CABLE: Local Area Network Cable (Ethernet cable used for wired connections).

# When working with SSH on a Raspberry Pi, you'll often use commands like:

# SSH: Connect to a remote system using SSH.

# SCP: Secure Copy (copy files securely between systems).

# Raspi-Config: Raspberry Pi configuration tool.

# Ifconfig or IP address: Check network interfaces and IP addresses.

# Sudo: Execute a command with superuser privileges.

# Hostname: Display or set the system hostname.

# SYMBOLS

# Symbols in Raspberry Pi :

Raspberry Pi is a well-known single-board computer that comes equipped with a variety of symbols and components on its board. If you are new to Raspberry Pi or just want to gain a better understanding of the device, it's important to familiarize yourself with some of the key symbols and components you might find on a typical Raspberry Pi board. By doing so, you'll be able to easily identify and troubleshoot any issues that may arise.

**1. GPIO (General Purpose Input/Output) Pins:**

- These are pins that can be used for general-purpose input or output. They are used to connect the Raspberry Pi to external devices and components.

**2. USB Ports:**

- These are used for connecting USB devices such as keyboards, mice, and external storage.

**3. Ethernet Port:**

- Used for connecting the Raspberry Pi to a network using an Ethernet cable.

**4. HDMI Port:**

- Used for connecting the Raspberry Pi to a display using an HDMI cable.

**5. MicroSD Card Slot:**

- This is where you insert the microSD card containing the operating system and data for the Raspberry Pi.

**6. Power Port:**

- Where you connect the power supply to power the Raspberry Pi.

**7. Camera and Display Connectors:**

- Raspberry Pi models may have connectors for attaching a camera module and a display.

**8. Audio/Video Jack:**

- Some models have a 3.5mm audio/video jack for connecting speakers or headphones.

**9. LEDs (Light Emitting Diodes):**

- These small lights indicate various statuses, such as power and SD card activity.

**10. SoC (System on a Chip):**

- The main processor that powers the Raspberry Pi. The specific chip may vary between models.

**11. Heat Sink and GPIO Labels:**

- Some Raspberry Pi models have a heat sink for cooling the processor, and GPIO pin labels printed on the board to help identify pin functions.

**12. Push-Button Switch:**

- A small button on the board, often used for initiating a shutdown or triggering specific actions.

**13. Polyfuse:**

- A self-resetting fuse that helps protect the Raspberry Pi from overcurrent situations.

**14. CSI (Camera Serial Interface) and DSI (Display Serial Interface) Connectors**:

- These connectors are used for connecting camera and display peripherals, respectively.

**15. Header Pins:**

- These are pins that allow you to connect additional hardware or accessories to the Raspberry Pi, such as HATs (Hardware Attached on Top) or jumper wires.

Remember that the specific symbols and components can vary between different models of Raspberry Pi, so it's essential to refer to the documentation for the specific version you are using.

**Symbols in Two way Authentication:**

Two-factor authentication (2FA) is a security process in which a user provides two different authentication factors to verify their identity. These factors typically fall into three categories: something you know, something you have, and something you are. Here are common symbols and representations associated with two-factor authentication:

**1. Key Symbol:**

- A key is often used to represent the "something you have" factor, such as a physical key, keycard, or digital key.

**2. Lock Symbol:**

- A lock can symbolize security and is often associated with the protection of information. It may be used to represent the authentication process.

**3. Password/Pin Symbol**:

- An alphanumeric symbol, often in the form of asterisks or dots, represents the "something you know" factor, such as a password or PIN.

**4. Smartphone Symbol:**

- A smartphone is commonly used as a device for 2FA. Icons representing a mobile device, especially with a key or lock, may signify the use of a mobile app for authentication.

5. SMS or Message Symbol:

- A speech bubble or message icon can represent the delivery of authentication codes via SMS or other messaging services.

6. Biometric Symbols:

- Fingerprints, eye scans, or other biometric icons represent the "something you are" factor in 2FA.

7. Clock/Time Symbol:

- A clock or timer icon may indicate that a time-sensitive code is part of the authentication process, such as a Time-based One-Time Password (TOTP).

8. Checkmark/Verified Symbol:

- A checkmark or a symbol indicating verification is often used to confirm that the user has successfully passed the authentication process.

9. Grid or Matrix Symbol:

- Some 2FA systems use a grid of characters or numbers that the user must reference during the authentication process.

10. QR Code Symbol:

- A QR code may represent the setup or scanning of a code for authentication using a mobile app.

11. Multiple Layers/Steps Symbol:

- A representation of layered security or multiple steps, such as a multi-step staircase or layers of shields, can indicate that more than one factor is required.

These symbols are often used in user interfaces, mobile apps, or documentation to communicate the two-factor authentication process effectively. It's worth noting that the symbols used may vary between different platforms and services, so users should be familiar with the specific symbols associated with the 2FA method they are using.

**Chapter 1: Introduction:**

Traditional username and password based single-factor authentication is no longer sufficient in today's digital landscape where cyber threats are constantly evolving. Simple passwords are easily cracked or phished, exposing user accounts and private information to risk. As more services move online, there is a growing need for robust authentication mechanisms to securely and conveniently identify legitimate users. Two-factor authentication addresses this need by requiring two separate credentials - something you know (password) and something you have (mobile phone) - to access sensitive systems and data. This project aims to develop an affordable yet secure two-factor authentication solution using Raspberry Pi as the backend server and one-time passwords (OTPs) delivered over SMS as the second factor. The goals are to prototype the system, assess its usability for end users and test the security against potential threats. This will contribute to strengthening organizational network perimeter protection and user account safety through a practical and cost-effective multi-factor approach.

* 1. Problem Definition

With the proliferation of internet-connected devices and services, identity theft and online account compromises have become increasingly common. Cybercriminals employ sophisticated techniques like phishing, malware, credential stuffing and brute force attacks to steal sensitive information like usernames, passwords and payment details. Traditional single-factor authentication relying only on passwords is inadequate to thwart such advanced threats. Even complex passwords can be cracked using tools or guessed during phishing. This has created an urgent need for multi-layered authentication approaches that can protect user identities and accounts more robustly without inconveniencing legitimate users.

* 1. Problem Overview

The inability of single-factor password-only authentication methods to prevent unauthorized access in today's complex cyber threat landscape poses serious problems. Financial losses 5 due to identity theft run into millions while data breaches erode user trust in online platforms. Compromised accounts are also misused to cause secondary damage via activities like spreading malware or scamming contacts. While multi-factor authentication provides stronger security, existing commercial solutions can be expensive and inaccessible for some. This project aims to address this problem through development of an affordable yet robust two-factor system leveraging open-source tools. By integrating SMS OTPs delivered by a Raspberry Pi server, it seeks to provide an additional layer of security beyond passwords during login at minimal cost. This could help both individuals and organizations fortify their online defence through a practical alternative.

* 1. Hardware Specification

The main hardware components are - Raspberry Pi 3/4 Model B single board computer, 8GB or higher microSD card, USB power adapter, Ethernet cable

* 1. Software Specification

The software tools to be used are - Raspbian GNU/Linux OS, Python 3 programming language, MySQL database to store user credentials, Twilio API for SMS integration.

**Chapter 2: Literature survey:**

2.1 Existing System

Several two-factor authentication systems currently exist in the market. Popular commercial options like Google Authenticator, Microsoft Authenticator and Authy utilize mobile apps to generate time-based one-time passwords (TOTP). Hardware security keys from Yubico provide near-field communication (NFC)-based OTPs. Banking institutions commonly use SMS-delivered one-time passcodes as the second factor. However, app-based solutions require compatible devices. Hardware security keys involve additional purchase costs. Text messages as a second factor pose privacy and portability issues. Furthermore, both consumer and organizational implementations of existing multi-factor options can be expensive to set up and manage at scale. This project addresses such limitations by proposing an open-source, 6 Raspberry Pi-based authentication-server integrated with cost-effective SMS OTPs. It aims to offer a practical and affordable alternative for small teams.

2.2 Proposed System

The proposed system seeks to develop a secure yet low-cost two-factor authentication solution leveraging open-source tools and technologies. At the core will be a Raspberry Pi single board computer functioning as the authentication server. It will be programmed using Python to generate and store one-time passwords (OTPs), interact with a MySQL database for user credentials, and integrate with the Twilio API for sending SMS messages. When a user attempts to login, the Raspberry Pi server will generate a random numeric OTP and send it via SMS to the phone number registered with their account. The user is then prompted to enter this received OTP in addition to their username and password on the login webpage or application interface. If the details match, authentication will be approved and access granted. Otherwise, login attempts are blocked after a set number of tries to prevent brute force attacks. Activity logs will aid auditing compromised accounts. The entire system aims to provide robust yet affordable multi-factor security for capacity-constrained teams.

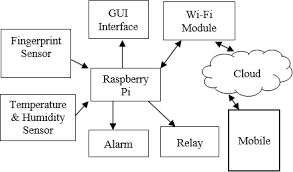
2.3 Literature Review Summary

A literature Using a systematic literature review, an exhaustive exploration of the research topic was made to provide the objective summary of current studies related to the research topic. Systematic literature described as a qualitatively and quantitatively identifying, merging, and assessing all available data to produce results related to a specific research question. This entire work is done on the Linux based embedded computer called raspberry pi, in which database creation and management using postgre sql, web page creation using PHP language, fingerprint reader access, authentication and recognition using python were entirely done on raspberry pi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Author** | **Year** | **Title** | **Key finding** | **Methodology** |
| Eben Upton | 2012 | Raspberry pi | To learn programming skills, build hardware projects, do home automation, implement Kubernetes clusters and edge computing, and even use them in industrial applications. | Project planning, operating system,documentation, program development,power management,security network connection. |
| Eben Upton | 2015 | Raspberry  Pi 2 model B | The raspberry pi 2 B model is Provide with the same pi as before but now with double the ram and much faster processor. | Server and network services, Media-center, home automation and IoT, programming development, security privacy, gaming, custom projects. |
| Eben Upton | 2018 | Raspberry pi 3 model A+ | The raspberry pi 3 model A+ is the single board computer (BCM2837B0). It can be sufficient for many tasks and lightweight applications and it can built-in wireless connectively. | Getting started, operating system installation, configuration, remote access, updating software, GPIO pins, prototyping power management, backup and security, documentation. |
| Eben Upton | 2019 | Raspberry Pi 4 model B | The raspberry pi 4 model B is powered by a Broadcom (BCM2711), it can be significant of performance improvements over earlier models. | Get the hardware, prepare the microSD card, connect peripherals, boot-up, initial setup, software installation, project development, explore projects and communications. |

**Chapter 3: Design flow/Process:**

3.1 **SYSTEM ARCHITECTURE**



The fingerprint sensor in our suggested model is the GT511-C3, which has its own database. After enrollment and verification, it promptly notifies the Raspberry Pi. The DHT22, employed as the temperature and humidity sensor, transmits readings to the cloud every second. A buzzer functions as an alarm, activating when the current readings exceed predefined threshold values. Simultaneously, the relay is switched off, turning off the LED to indicate system deactivation. The Raspberry Pi acts as the central hub, facilitating communication with all system components. Equipped with its Wi-Fi module, it continuously uploads temperature and humidity values to the cloud. The Raspberry Pi creates a graphical user interface (GUI) for user interaction The fingerprint sensor in our suggested model is the GT511-C3, which has its own database. After enrollment and verification, and, if surpassed, triggers the LED while deactivating the relay. • Fingerprint Sensor (Biometric Factor): Utilizes a high-quality fingerprint sensor such as GT511C3 with a 32-bit CPU and storage capacity for fingerprint templates. The sensor interfaces with Raspberry Pi, which supports Python for effective integration. Responsible for capturing and authenticating user fingerprints. • DHT22 Sensor (Environmental Factor): Incorporates a DHT22 sensor to measure and provide accurate digital outputs for both temperature and humidity. Offers superior precision compared to alternatives like DHT11. Enhances the overall security of the system by incorporating environmental data as an additional authentication factor. • Raspberry Pi: Serves as the central processing unit and communication hub for the system. Communicates with the fingerprint sensor, DHT22 sensor, and other system components. Executes the authentication process by comparing the provided fingerprint data and environmental readings against stored templates and thresholds. • Firebase (Cloud Storage and Authentication): Implements Firebase for cloud storage to securely store and retrieve user data, including fingerprint templates. Manages OTP (One-Time Password) generation and validation. gives a real-time information on the humidity information through the associated mobile app. • Two-Factor Authentication Process: User initiates the authentication process by presenting a fingerprint. Fingerprint data is processed by Raspberry Pi for verification against stored templates. Simultaneously, DHT22 sensor readings are obtained for environmental authentication. If both fingerprint and environmental factors pass validation, access is granted. In the case of a successful authentication, the user is notified through the mobile app. OTPs may also be used for an additional layer of authentication, with Firebase managing their generation and validation.

**3.2 PROBLEM FORMULATION**

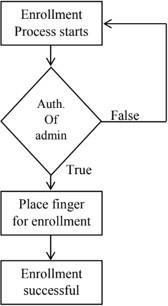
In today's rapidly evolving digital landscape, ensuring secure access to sensitive systems and data is paramount. However, traditional username-password authentication mechanisms are susceptible to various cyber threats such as phishing, brute force attacks, and password leaks. To address these vulnerabilities, researchers and practitioners have turned to more advanced 8 authentication methods, including two-way authentication. Two-way authentication, also known as two-factor authentication (2FA), involves the use of two distinct factors to verify a user's identity, enhancing security by requiring something the user knows and something the user possesses. The challenge lies in developing efficient and user-friendly two-way authentication systems that can effectively thwart unauthorized access attempts while maintaining usability and convenience. While there has been significant progress in this area, several issues persist. Integration Complexity: Implementing two-way authentication often requires the integration of disparate technologies and methods, leading to complexity in system design and deployment. Usability vs. Security: Striking a balance between security and user experience is crucial. Overly complex authentication processes can discourage users, leading to non-compliance or the adoption of less secure alternatives. Cost-Effectiveness: Organizations need cost-effective solutions that can be deployed at scale without imposing exorbitant expenses, particularly for small and medium sized enterprises. Adaptability: Authentication methods should be adaptable to different contexts and user preferences, accommodating a range of factors such as biometrics,

tokens, and mobile devices. Evolving Threat Landscape: Cyber threats continually evolve, necessitating authentication systems that can keep pace with emerging attack vectors and vulnerabilities. Privacy Concerns: Biometric authentication methods raise privacy concerns, necessitating robust encryption and secure storage practices. Integration with Legacy Systems: Integrating modern two-way authentication with existing legacy system can pose compatibility challenges. The goal of this research is to address these challenges by exploring innovative approaches to implement two-way authentication using Raspberry Pi. This cost-effective and versatile platform offers potential solutions to the aforementioned issues, providing a foundation for developing secure and user-friendly authentication system that can adapt to various use cases and security requirements. These credentials will be hardcoded into the Python scripts on each device. Once setup is complete, Pi 2 will run the client script to send a authentication request to Pi 1's server with its hardcoded username and password.es

**3.3 EXPERIMENTS**

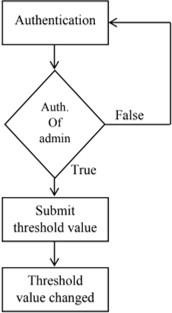
The Biometric authentication system designed have been prototyped for an industrial application to manipulate temperature and humidity values only on fingerprint authentication. The authentication system has three Process that are Enrollment, Authentication and Remote Authentication. The Buzzer is Turned ON and Relay is turned OFF when © 2023 IJNRD | Volume 8, Issue 1 January 2023 | ISSN: 2456-4184 | IJNRD.ORG IJNRD2301223 International Journal of Novel Research and Development (www.ijnrd.org) c173 current temperature and humidity readings exceed threshold levels.

1. Enrollment Process In Enrollment Process Admin has to give authorization to allow others to get enrolled into the system.



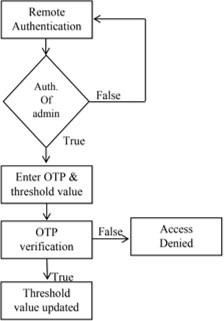
(Registration)

1. Authentication Process In Authentication process the admin of the system can directly manipulate the threshold settings by performing fin- gerprint authentication.



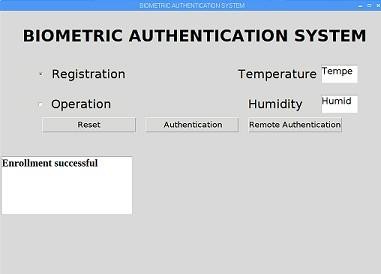
(Authentication)

1. Remote Authentication Process In Remote Authentication process the admin when not in place can give access to other person who has his fingerprint registered by sending a 10 digit OTP code. The Code is Random and unique generated using the admin’s mobile App which is sent as SMS to the person for access and simultaneously is sent to cloud for Verification. This Code will be valid only for short time in our project it is 30secs, post that it is replaced with garbage values.

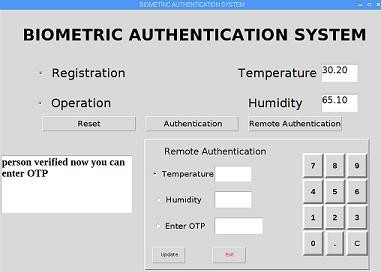


**(Remote Authentication)**

**Chapter 4: Results analysis and validation:**



**(Enrollment window)**



**(Remote authentication window)**

**Chapter-5: Conclusion and Future work:**

**5.1 Conclusion:**

The Raspberry Pi provides a low cost, customizable platform for delivering hardware-backed two-factor authentication to both on-premise and remote services. By acting as the shared secret TOTP generator, it adds a physical security token beyond just passwords or software authenticators. Implementing the TOTP generator on the Pi adds an extra layer of assurance that one-time passwords are only generated from authorize d hardware in possession of the legitimate user. Software-only solutions are vulnerable to spoofing or interception of secrets. Pairing the Pi with client-side verification modules allows strong two-factor authentication to be applied across multiple different types of services, from SSH access to VPN logins to web applications. The modular design makes the solution adaptable.

**5.2 Future work:**

Future work in blood banking via cloud computing holds immense potential for further advancements and improvements in the field. Here are some areas that can be explored:

1. Advanced Analytics and Decision Support Integration of advanced analytics cloud-based blood banking systems enable predictive analysis, trend. identification, and decision support. This can help in optimizing blood inventory management, predicting patient 34 needs, and improving resource allocation.

2. Internet of Things (IoT) Integration: Incorporating IoT devices, such as smart sensors and wearable devices, into the blood banking ecosystem can provide real- time data on temperature, storage conditions, and transportation logistics. This can enhance traceability, ensure quality control, and reduce the risk of blood wastage.

3. Blockchain Technology: Implementing blockchain technology in blood banking can enhance security, transparency, and traceability of blood supply chains. Blockchain can help authenticate donor records, track blood units from collection to transfusion, and ensure the integrity of data across multiple stakeholders.

4. Mobile Applications and Telemedicine: Developing mobile applications and telemedicine platforms integrated with cloudbased blood banking systems can facilitate remote blood donor registration, appointment scheduling, and result notifications. This can improve accessibility, donor engagement, and overall convenience for both donors and healthcare professionals.

5. Visual Reality (VR) and Augmented Reality (AR): Leveraging VR and AR technologies can enhance training programs for blood bank staff, allowing them to practice complex procedures in a virtual environment. AR can also assist in real time blood unit identification during transfusions, reducing the risk of errors.

6. Data Sharing and Collaboration: Establishing standardized protocols and frameworks for secure data sharing and collaboration among blood banks, healthcare providers, and research institutions can facilitate knowledge sharing, research collaborations, and the exchange of best practices.

**Chapter-6: References**

[1] Matt Richardson and Shawn Wallace – A bestseller that is great for beginners to get up and running with basic projects using a Pi.

[2] Simon Monk covers a wide variety of projects with clear explanations of electronics and coding principles.

[3] Divil jain, Dr. P.S Ramkumar, and DR. K.V.S.S.S.S Sairam, “IoT based Biometric Access Control System”, International Journal of Innovative Research Science, Engineering and Technology (IJIRSET), vol. 5 Issue9, pp. 555–559, May 2016.

[4] Dhvani Shah, D.K.Bharadi, V.A.Kaul, V.J.Amrutia, S., “End-toEnd Encryption Based Biometric SaaS: Using Raspberry Pi as a Re- mote Authentication Node”, IEEE sponsored 1st International Conference on Computing, Communication, Control and Automation (IC-CUBEA) February 2015, pg. 52 59.

[5] Vijayasanthi. R, Radha N, Jayashree M, Sindhuja P Fingerprint Authentication using Raspberry Pi based on IoT, International conference on Algorithm, Methodology, Models and Applications in Emerging Technologies (ICAMMAET).

[6] A. K. Jain, L. Hong, S. Pankanti, R. Bolle, An identity authentication system using fingerprints, Proceedings of the IEEE, vol. 85, no. 9, pp. 1365- 1388, September 1997.

[7] Archana S. Shinde, Varsha Bendre, An Embedded Fingerprint Authentication System, 2015 International Conference on Computing Communication Control and Automation.

[8] Adventures in Minecraft by David Whale and Martin O’Hanlon- Explore coding and engineering concepts within the popular games.

[9] Hello Raspberry Pi! By Ryan Heitz, Ben Everard step by step fundamentals of Linux, Python, Hardware interfacing for educators.

[10] Dan Aldred – perfect for younger learners, it holds their hands through creative toy builds.

[11]https://precisebiometrics.com/prod ucts/fingerprint-recognition-software