Contextual Authentication of Users and Devices

Thesis submitted by

MR. DIVYANS MAHANSARIA

Doctor of Philosophy (Engineering)

Department of Information Technology

Faculty Council of Engineering & Technology

Jadavpur University

Kolkata, India

2025

JADAVPUR UNIVERSITY

KOLKATA-700032, INDIA

INDEX NO. 207/19/E

1. Title of the thesis :

Contextual Authentication of Users and Devices

2. Name, Designation & Institution of the Supervisor :

Name: Dr. Uttam Kumar Roy

Designation: Professor

Institution: Department of Information Technology

Jadavpur University, Kolkata -700106, India.

3 . List of Publications :

Journal Publications -

1. Mahansaria, D., Roy, U.K. (2025). One-time password authentication based on QR code and hash chain. Iran Journal of Computer Science, Springer, DOI - https://doi.org/10.1007/s42044-025-00259-3.

2. Mahansaria, D., Roy, U.K. (2024). Contextual Authentication of Users and Devices Using Machine Learning. Computing, vol. 106(12), Springer journal, pp. 4083-4107.

Conference Publications -

1. Mahansaria, D., Roy, U.K. (2025). ATM Transactions made safer using NFC and Blockchain. In: 15th International Conference Confluence 2025, Noida, India. Publisher: Springer.

2. Roy, U.K., Mahansaria, D. (2020). Two-Factor Authentication Using Mobile OTP and Multi-dimensional Infinite Hash Chains. In: Future of Information and Communication Conference (FICC 2020), San Francisco, United States. Published: Advances in Information and Communication, vol 1129. Springer.

3. Mahansaria, D., Roy, U.K. (2020). Secure Authentication Using One Time Contextual QR Code. In: Security in Computing and Communications (SSCC 2019), Kerela, India. Published: Communications in Computer and Information Science, vol 1208. Springer.

4. Mahansaria, D., Roy, U.K. (2019). Secure Authentication for ATM transactions using NFC technology. In: 53rd International Carnahan Conference on Security Technology (ICCST), Chennai, India, Published: IEEEXplore.

5. Mahansaria, D., Roy, U.K. (2019). Developing Authentication Solutions using Quick Response (QR) Code & Near Field Communication. In: ph.d forum of International Conference on Internet Technologies, Innovation, Policy and Standards (ITIPS' 2019). By India Internet Foundation (IIFON), Kolkata, India.

4. List of Patents : NIL

5. List of Presentations in National/International/Conferences/Workshops :

1. Mahansaria, D., Roy, U.K. (2025). ATM Transactions made safer using NFC and Blockchain. In: 15th International Conference Confluence 2025, Noida, India. Publisher: Springer.

2. Mahansaria, D., Roy, U.K. (2019). Secure Authentication for ATM transactions using NFC technology. In: 53rd International Carnahan Conference on Security Technology (ICCST), Chennai, India, Published: IEEEXplore.

3. Mahansaria, D., Roy, U.K. (2019). Developing Authentication Solutions using Quick Response (QR) Code & Near Field Communication. In: ph.d forum of International Conference on Internet Technologies, Innovation, Policy and Standards (ITIPS' 2019). By India Internet Foundation (IIFON), Kolkata, India.

Statement of Originality

I Divyans Mahansaria registered on 05/07/2019 do hereby declare that this thesis entitled "**Contextual Authentication of Users and Devices**" contains literature survey and original research work done by the undersigned candidate as part of Doctoral studies.

All information in this thesis have been obtained and presented in accordance with existing academic rules and ethical conduct. I declare that, as required by these rules and conduct, I have fully cited and referred all materials and results that are not original to this work.

I also declare that I have checked this thesis as per the "Policy on Anti Plagiarism, Jadavpur University, 2019", and the level of similarity as checked by iThenticate software is ___**6**_____%.

Signature of Candidate:

Divyans Mahansaria

Date :

15/01/2026

Certified by Supervisor :

15.01.2026

(Dr. Uttam Kumar Roy)

PROFESSOR
Dept. of Information Technology,
JADAVPUR UNIVERSITY
Salt Lake Campus, Block-LB/8, Sector-III
Salt Lake, Kolkata-700106, India

CERTIFICATE FROM THE SUPERVISOR/S

This is to certify that the thesis entitled "**Contextual Authentication of Users and Devices**" submitted by Shri Divyans Mahansaria, who got his name registered on 05.07.2019 for the award of Ph.D. (Eng.) degree of Jadavpur University is absolutely based upon his own work under the supervision of Prof. Uttam Kumar Roy and that neither his thesis nor any part of the thesis has been submitted for any degree/diploma or any other academic award anywhere before.

15.01.2026

Dr. Uttam Kumar Roy

Professor

Department of Information Technology

Jadavpur University, Kolkata, India

PROFESSOR
Dept. of Information Technology,
JADAVPUR UNIVERSITY
Salt Lake Campus, Block-LB/8, Sector-III
Salt Lake, Kolkata-700106, India

Acknowledgements

Completing this thesis has been an enriching experience, and it would not have been possible without the support and guidance of several individuals. I am sincerely thankful to all those whose encouragement and assistance played a crucial role throughout this journey.

I extend my heartfelt appreciation to my project advisor, **Prof. Uttam Kumar Roy**, for his invaluable guidance, consistent support, and insightful advice throughout the course of my research. His mentorship has been instrumental not only in the successful completion of this work but also in shaping my academic and professional development. This thesis would not have come to fruition without his dedication and encouragement.

Lastly, I extend my heartfelt thanks to my beloved spouse for her unwavering support and encouragement throughout this journey.

Regards Divyans Mahansaria
15/01/2026

(DIVYANS MAHANSARIA)

List of Figures

Figure 1.1: Historical timeline of some authentication methods 1
Figure 1.2: Different methods of authentication 2
Figure 1.3: Sample virtual keyboard 3
Figure 1.4: An example of a graphical password technique 3
Figure 1.5: Sample QR codes 5
Figure 1.6: Near-Field Communication 7
Figure 1.7: NFC architecture 8
Figure 1.8: Digital certificate issuance process in blockchain 9
Figure 1.9: Digital certificate validation process in blockchain 9
Figure 2.1: RSA SecurID key fob 17
Figure 3.1: A 3-dimensional hypercube having eight nodes and twelve edges 23
Figure 3.2: Registration process 24
Figure 3.3: Authentication process 26
Figure 3.4: Demonstration application of the proposed solution 32
Figure 3.5: Sample registration data 32
Figure 3.6: Sample challenge data 33
Figure 3.7: Trust boundary 33
Figure 3.8: OTP computation time for various challenge lengths 40
Figure 3.9: Time taken for challenge generation for different dimensions 41
Figure 4.1: Randomized QWERTY keyboard layout mapping 46
Figure 4.2: Password length vs average input time 53
Figure 5.1: Registration process 55
Figure 5.2: Authentication process 56
Figure 5.3: Trust boundary and the communication channels 60
Figure 6.1: Authentication process 66
Figure 6.2: Steps in IDS modeling 68
Figure 6.3: Recall for DoS/DDoS 72
Figure 6.4: Precision for DoS/DDoS 72
Figure 6.5: Accuracy for DoS/DDoS 73
Figure 6.6: F-Score for DoS/DDoS 73
Figure 6.7: Recall for brute force 76
Figure 6.8: Precision for brute force 77
Figure 6.9: Accuracy for brute force 77
Figure 6.10: F-Score for brute force 77
Figure 6.11: Trust boundary and the communication channels 80

List of Tables

Table 3.1: Comparison of the proposed scheme with related schemes 37
Table 3.2: Challenge generation time (in milliseconds) 41
Table 4.1: Symbols 45
Table 4.2: Comparison with other OTP-based authentication systems 52
Table 6.1: Confusion Matrix 69
Table 6.2: Model-built time (seconds) for DoS/DDoS 74
Table 6.3: Top 14 ranked features for DoS/DDoS 74
Table 6.4: Model-built time (seconds) for brute force 78
Table 6.5: Top 10 ranked features for brute force 79

List of Abbreviations

ATM	Automated Teller Machine
CA	Certificate Authority
CIC	Canadian Institute for Cybersecurity
CNN	Convolutional Neural Network
DIDs	Decentralized Identifiers
DKMS	Decentralized Key Management System
DoS	Denial-of-Service
DDoS	Distributed Denial-of-Service
DT algorithm	Decision Tree algorithm
FIDO2	Fast Identity Online 2
FTP	File Transfer Protocol
HCE	Host Card Emulation
HOTP	HMAC-based One-Time Password
HTTP	HyperText Transfer Protocol
IDS	Intrusion Detection System
IIoT	Industrial Internet of Things
IP	Internet Protocol
J48	Java 48 Decision Tree Algorithm
K-NN	k-Nearest Neighbour
LSTM	Long Short-Term Memory
MFA	Multi-Factor Authentication
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSA	Multistage Attack
MITM	Man-in-the-Middle
NB algorithm	Naive Bayes algorithm
NFC	Near-Field Communication
OTP	One-Time Password
OS	Operating System
P2P	Peer-to-Peer

PIN	Personal Identification Number
QR code	Quick Response code
RAM	Random Access Memory
RFC	Request for Comments
RFID	Radio Frequency Identification
RF	Radio Frequency
RF algorithm	Random Forest algorithm
RSA algorithm	Rivest–Shamir–Adleman algorithm
SE	Secure Element
SHA	Secure Hash Algorithm
SMBA	Smart Mobile Banking Application
SMS	Short Message Service
SSO	Single Sign-On
SSH	Secure Shell
SSI	Self-Sovereign Identity
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TPM	Trusted Platform Module
TEE	Trusted Execution Environment
TLS	Transport Layer Security
TOTP	Time-based One-Time Password
TZOS	Trust Zone Operating System
UDP	User Datagram Protocol
UTC	Coordinated Universal Time
VC	Verifiable Credentials
Weka	Waikato Environment for Knowledge Analysis
XML	Extensible Markup Language

ABSTRACT

Confidential data is exchanged between the user/device and the authentication server at the time of authentication to determine the legitimacy of the source requesting authentication. Various authentication techniques are vulnerable to numerous types of attacks. Safeguarding the authentication process from security attacks is of utmost importance. Different authentication methods exist depending on the system's requirements. However, no authentication process can guarantee foolproof security.

The motivation behind our research is the need to counter the ever-evolving security attacks, especially during the authentication process in a system. As part of this thesis work, an innovative authentication framework has been designed to counter security threats across diverse systems and scenarios. The framework comprises four distinct authentication solutions, each tailored to address specific security requirements for user and device authentication. A comprehensive security analysis and threat modeling have been conducted to assess the effectiveness of these solutions in enhancing the security posture of protected systems.

In one of the solutions, a novel one-time password authentication system has been developed based on Quick Response (QR) code and a multi-dimensional hash chain. A one-time password (OTP) will be generated at the client end based on a challenge thrown by the server. This challenge is represented in the geometrical coordinate system as a random path originating from the origin and leading to a point located within a multi-dimensional hypercube. A QR code has been used to facilitate the exchange of information between a client terminal and an end-user device without inputting the data manually.

Another solution introduces a unique one-time QR code-based mechanism designed to mitigate different forms of security threats during the authentication process. This method leverages context-aware authentication by embedding dynamic information within the QR code, which varies with each login attempt. Based on the encoded data, users generate a one-time password derived from their original password. The proposed technique is versatile and can seamlessly integrate into existing and emerging systems.

ATM (automated teller machine) networks are widely used worldwide for financial transactions. The most common and widely used method to authenticate at an ATM kiosk is using the ATM card and its associated PIN (personal identification number). Although this authentication method has existed for many years, it suffers from some difficulties, such as carrying an additional ATM card during use, manufacturing and securely shipping the ATM card to the end user, being prone to ATM card skimming, etc. To handle these challenges, we have designed a solution to authenticate in ATM networks using near-field communication (NFC) and blockchain technologies. The communication range for data exchange between two NFC devices or NFC cards is minimal (typically less than 4 cm), making it a good choice for exchanging sensitive data. Blockchain technology has gained recognition as a reliable solution for secure data storage and transaction verification. In the proposed solution, digital identity is stored in the blockchain to carry out two-factor authentication of the system users.

In another research, we have aimed to use the context of users and devices during authentication to detect anomalies and security-related attacks. This solution can be used in conjunction with each of the other three proposed solutions, i.e., it will complement the different solutions in enhancing the security level. In our research, in particular, denial-of-service (DoS)/distributed denial-of-service (DDoS) attacks, & brute-force techniques have been analyzed in detail using contextual information. Extensive simulations were conducted on the benchmark CIC-IDS2017 dataset using the Weka tool. The performance metrics of recall, precision, accuracy, f-score, and model-built time were computed for the four machine-learning classifiers—J48, Random Forest, Multi-Layer Perceptron, and Bayes Net—for different combinations of data splits and groups of data features.

Index Terms - Application Security; Secure Authentication; Context-based authentication; OTP; QR code; NFC; Security attacks

Contents

Introduction *1*

1.1 Authentication **1**

1.2 ATM Networks **4**

1.3 QR Code **5**

1.4 Hash Chain **6**

1.5 Near-Field Communication **7**

1.6 Blockchain **8**

1.7 Machine Learning **10**

1.8 Contextual Information **11**

1.9 Trusted Platform Module **11**

1.10 Threat Modeling **12**

1.11 Thesis Structure **12**

Literature Survey *14*

2.1 Hash Chain **14**

2.1.1 Lamport and S/Key One-Time Password System 14

2.1.2 Scheme of Bicakci and Baykal 15

2.1.3 Scheme of Eldefrawy et al. 16

2.1.4 Other Hash Chain Techniques 16

2.2 RSA SecurID Authenticator **17**

2.3 NFC-based Authentication Solutions **17**

2.4 ATM-based Authentication Solutions **18**

2.5 Blockchain-based Authentication Solution **19**

2.6 Machine Learning-based Security Solutions **20**

2.6.1 Behavioral Biometrics 20

2.6.2 DDoS/DoS Attacks 20

2.6.3 Brute Force Attacks 21

One-Time Password Authentication based on QR Code and Hash Chain *22*

3.1 Overview **22**

3.2 Proposed Solution **23**

3.2.1 Working Principle 23

3.2.2 Registration 24

3.2.3 Authentication 26

3.2.3.1 First Factor of Authentication 26

3.2.3.2 Authentication Server - Challenge Generation 26

3.2.3.3 Mobile App - Derive OTP 28

3.2.3.4 Authentication Server - Verify OTP 29

3.2.4 Sample Implementation 30

3.2.5 Demonstration of the Proposed System 31

3.3 Security Analysis and Threat Modeling **33**

3.3.1 Security Objectives 33

3.3.2 Trust Boundary ... 33
3.3.3 Threats ... 34
3.3.3.1 Attacks on the Components ... 34
3.3.3.2 Attacks on the Communication Channels ... 35
3.3.3.3 Attacks on the Authentication Protocol ... 35
3.3.4 Discussion ... 36
3.3.5 QR Code Security Considerations ... 38

3.4 Performance Evaluation and Applications ... 38
3.4.1 Memory Consumption for OTP Computation ... 39
3.4.2 Time Complexity – OTP Computation ... 40
3.4.3 Time Complexity – Challenge Generation ... 40
3.4.4 Applicability ... 41

3.5 Conclusion and Limitations ... 42

Secure Authentication using One Time Contextual QR Code ... 43

4.1 Overview ... 43

4.2 Proposed Solution ... 43
4.2.1 Registration Phase ... 44
4.2.2 Device Setup ... 44
4.2.3 Authentication Phase ... 44
4.2.3.1 Authentication Server - Generate Login Information ... 45
4.2.3.2 User - Derive OTP ... 47
4.2.3.3 Authentication Server - Verify OTP ... 48

4.3 Security Analysis, Applicability and Discussion ... 48
4.3.1 Security Analysis ... 48
4.3.2 Applicability ... 51
4.3.3 Discussion ... 51

4.4 Usability Study ... 52

4.5 Conclusion ... 53

ATM transactions made safer using NFC and Blockchain ... 54

5.1 Overview ... 54

5.2 Proposed Solution ... 54
5.2.1 Registration Process ... 55
5.2.2 Authentication Process ... 56

5.3 Discussion ... 57
5.3.1 Threat Modeling ... 59
5.3.1.1 Security Objectives ... 59
5.3.1.2 Trust Boundary ... 60
5.3.1.3 Threats ... 61

5.4 Conclusion ... 62

Contextual Authentication of Users and Devices using Machine Learning ... 64

6.1 Overview ... 64

6.2 Proposed Solution ... 64
6.2.1 Working Principle ... 64
6.2.2 User/Device Registration ... 65
6.2.3 Authentication Process ... 66
6.2.3.1 First Level of Authentication ... 66
6.2.3.2 Second Level of Authentication ... 67

6.2.4 Intrusion Detection System Modeling 67
6.2.4.1 Data Collection 67
6.2.4.2 Data Preprocessing 68
6.2.4.3 Feature Selection 68
6.2.4.4 Model Classifier 69
6.2.4.5 Model Training 69
6.2.4.6 Model Evaluation 69
6.3 Experiments and Results 70
6.3.1 DoS/DDoS Attack 71
6.3.1.1 Dataset 71
6.3.1.2 Analysis 71
6.3.1.3 Inference 74
6.3.2 Brute Force Attack 75
6.3.2.1 Dataset 75
6.3.2.2 Analysis 75
6.3.2.3 Inference 78
6.3.3 Limitations 79
6.4 Security Analysis and Threat Modeling 80
6.4.1 Security Objectives 80
6.4.2 Trust Boundary 80
6.4.3 Threats 81
6.4.3.1 Attacks on the Components 81
6.4.3.2 Attacks on the Communication Channels 82
6.4.3.3 Attacks on the Authentication Protocol 82
6.4.4 Applicability 82
6.5 Conclusion 83
Conclusions 84
7.1 Overview 84
7.2 Contributions 84
7.3 Future Works 86
References 88

Chapter 1

Introduction

1.1 Authentication

Authentication refers to the process of verifying the identity of an individual attempting to access a system or resource to ensure they are legitimate. Existing user authentication schemes are broadly classified as knowledge-based (e.g., personal identification number and password), possession-based (e.g., RSA token, SecurEnvoy, and automated teller machine cards), inheritance-based (e.g., biometrics like fingerprint, voice, and face recognition), and somebody you know (social networks).

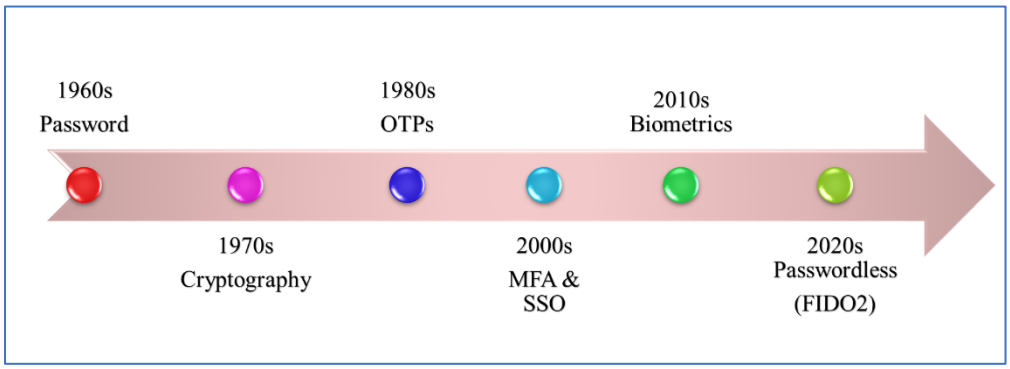


Figure 1.1: Historical timeline of some authentication methods

Figure 1.1 presents a historical timeline of some authentication methods, tracing their evolution from the 1960s to the 2020s. The journey begins with the introduction of usernames and passwords in the 1960s, invented by a professor at MIT, marking the earliest widely used form of user authentication. In the 1970s, cryptographic techniques of both

symmetric and asymmetric encryption emerged, enhancing the security of authentication systems. The concept of one-time passwords (OTPs) was first proposed in the early 1980s by Lamport, who introduced the idea of using a hash chain to generate OTPs, providing a more dynamic and secure alternative to static passwords. Moving into the 2000s, the focus shifted to multi-factor authentication (MFA) & single sign-on (SSO) technologies, aimed at strengthening security while streamlining user access across systems. The 2010s saw the rise of biometric authentication methods, leveraging unique biological characteristics such as facial features and fingerprints to authenticate users. Finally, the 2020s have ushered in the era of passwordless authentication, particularly with the development of the FIDO2 standard, aiming to enhance security while improving user convenience. After over 65 years, the study and implementation of secure authentication systems have become critical due to the massive growth in cyberattacks. Safeguarding the authentication process against evolving threats is of utmost importance. Moreover, modern authentication solutions must balance strong security with user-friendliness, ensuring that added layers of protection do not burden users, such as requiring them to remember complex passwords for every account.

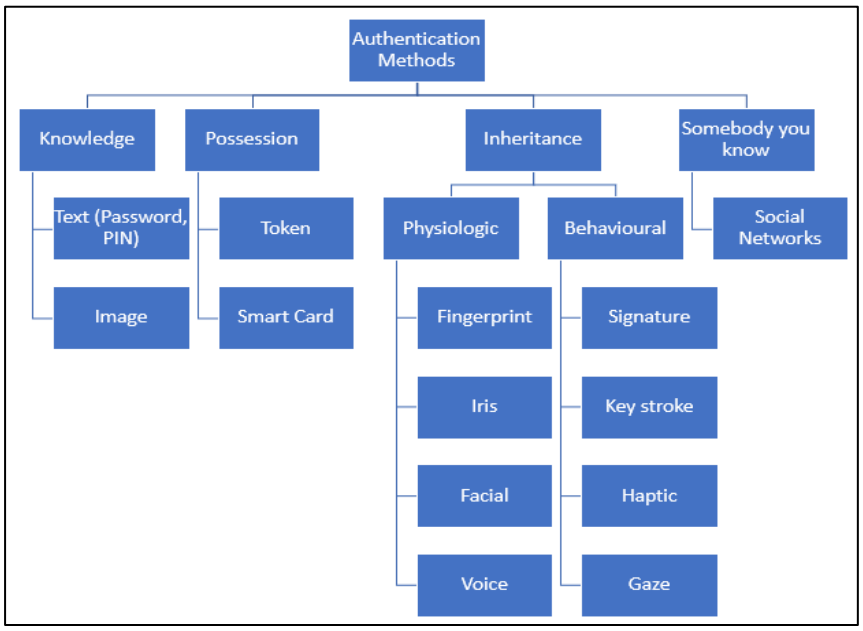


Figure 1.2: Different methods of authentication

Figure 1.2 represents the different methods of authentication.

Among these techniques, login credentials are the most common and adopted worldwide. Unfortunately, it is subject to a wide variety of attacks such as key-logger and asterisk-logger, eavesdropping, shoulder surfing, brute-force, dictionary attacks, replay attacks, Trojan Horse intrusions, phishing schemes, man-in-the-middle (MITM) attacks, SQL injection, and others.

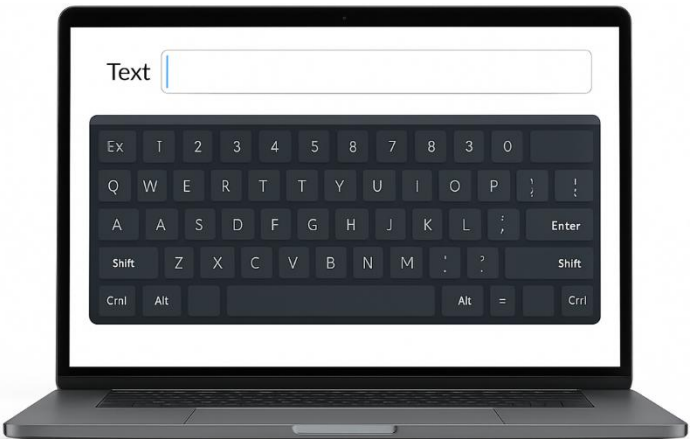


Figure 1.3: Sample virtual keyboard

A virtual or on-screen keyboard is used in some applications to prevent key-logging. A sample virtual keyboard is shown in Figure 1.3. Also, extensions to the virtual keyboard have been proposed to make it more secure [35, 84]. However, they cannot handle both shoulder surfing and screen capturing.

As a propitious alternative to traditional alphanumeric passwords, graphical passwords [104] with challenge-response [28, 64, 91] based authentication methods have been proposed to strengthen the system. An example of a graphical password technique is illustrated in Figure 1.4.

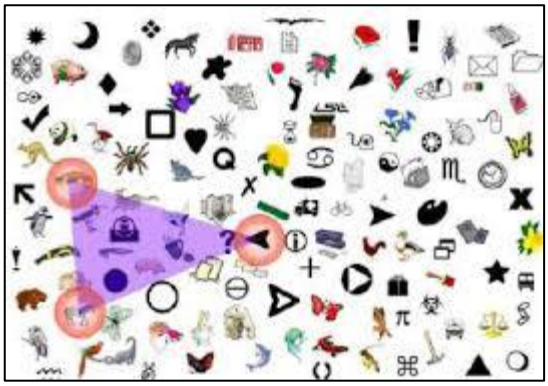


Figure 1.4: An example of a graphical password technique

However, they are still susceptible to either key-logging, mouse-logging, screen capture, or shoulder-surfing attacks [2]. Also, some cases require an additional device [2], like a

stylus, limiting usability. Compared to text-based password schemes, some graphical password schemes have less password space and thus reduce the system's entropy [64].

Biometric-based (e.g., fingerprint, face, voice, iris, etc.) authentication is often used in high-end applications. Some exciting techniques have also been proposed to carry out authentication that uses characteristics such as haptic, gaze, handwriting, brain waves, etc., which are unique to the human body [15, 20, 65, 102]. These systems assure greater security but have limited applicability due to their high installation and maintenance costs and difficulty in implementation. A compromised biometric system is a significant security risk, as the user's biometrics do not change.

In the multi-factor authentication process, more than one authentication factor is deployed, and the chance of compromising all the factors simultaneously is comparatively lower than that of single-factor authentication.

Using a one-time rather than a static password has proved more secure while carrying out authentication in various scenarios [32, 81, 100]. It is not vulnerable to replay attacks. However, recently, there have been compromises on SMS OTPs sent on mobile phones [14]. Using suitable tools, the attacker could intercept the mobile traffic of an end user, including SMS. Moreover, mobile phone malware, and especially Trojans, are being used by criminals to obtain SMS messages containing OTPs [21]. Two typically used OTP implementations are HOTP (HMAC-based One-Time Password) and TOTP (Time-based One-Time Password), standardized in RFC 4226 [25] and RFC 6238 [26], respectively. HOTP is an event-based OTP algorithm that uses a shared secret key (seed) and an event counter. The seed is exchanged only once, and the client and server safely store it. TOTP employs the HOTP algorithm but replaces the event counter with a time counter. Some of the literature shows that from a secure perspective, TOTP is better than HOTP [43, 56].

1.2 ATM Networks

ATM (automated teller machine) networks are widely used worldwide for financial transactions. People all around the globe use ATM networks to withdraw cash and also carry out other banking activities. Thus, it is paramount to safeguard users and provide them with convenience while transacting using ATMs. The most common and widely used method to authenticate at an ATM kiosk is using the ATM card and its associated PIN

(personal identification number). Despite widespread use, physical cards present several security and reliability concerns during ATM network transactions. One major threat is ATM skimming, where a small skimmer device captures card data during use, often leading to card cloning, even in the case of chip-enabled cards. This has emerged as a significant security challenge. Additionally, the magnetic stripe or embedded chip in these cards can degrade over time due to frequent use, eventually rendering the card inoperative. These damaged cards need replacement. Manufacturing many cards and transporting them to the end users are both time-consuming and involve some cost.

1.3 QR Code

A Quick Response (QR) code or 2D matrix barcode can hold large amounts of data in a small space, in both horizontal and vertical directions. Figure 1.5 shows sample QR codes.

Figure 1.5: Sample QR codes

It has error correction capability and a fast response time. Recently, there has been an increase in QR code usage as it can be used to hold and retrieve information easily. In some of the proposed research work, QR code has been used to secure applications [1, 17, 27, 39, 40, 53, 62, 105, 106]. A sound authentication system must balance both security and usability.

A public QR code and some user-specific information could be merged to form a contextual QR code, providing data related to a particular context. Rouillard [45] combines context-aware QR codes from two parts: the Public Part (which is “traditional” QR code info) and the Private Part (which is XML-based context data). The private part can represent information like the user’s profile, location, device used by the user, time, type of environment, etc. The combined data is used to compute contextual messages.

1.4 Hash Chain

A hash chain is formed by repeatedly applying a cryptographic hash function to a starting value, known as the seed. Because secure hash functions are one-way functions, computing the next value in the sequence is straightforward. However, reversing the process to find a previous value from a given one is computationally impractical. The OTP schemes of Lamport [57] and S/KEY [78] are pioneering work in proposing the idea of a hash chain for authentication. The below hash chain of length N is formed by applying the hash algorithm h(.) for N times to an initial seed value s. Each element generated in the hash chain represents an OTP in the Lamport and S/KEY schemes.

$h^1(s), h^2(s), \ldots, h^{N-1}(s), h^N(s)$

A multi-dimensional hash chain is a hash chain with two or more directions. In the case of a 2-dimensional hash chain, the directions of the hash chain could be considered in the x-axis and y-axis in the coordinate geometry system. Cryptographic hash functions take input messages of variable length and map these to fixed-length output messages.

Choosing appropriate hash algorithms is crucial for maintaining the security and integrity of the generated OTPs. Some of the standard hash algorithms are SHA-256, SHA-224, SHA-512, SHA-384, [77] BLAKE2 [75], and BLAKE3 [50]. SHA-256, SHA-224, SHA-512, and SHA-384 belong to the SHA-2 family, known for their strong cryptographic properties. They provide strong levels of security, with SHA-256 providing a 256-bit output, SHA-224 offering a 224-bit output, SHA-512 producing a 512-bit output, and SHA-384 generating a 384-bit output. Designed as a faster alternative to SHA-2, BLAKE2 offers strong security with optimized performance. BLAKE3 is a modern improvement over BLAKE2, optimized for speed and parallelism while maintaining high cryptographic security. All of these hash functions exhibit high collision resistance and pre-image resistance. In cryptographic hashing, collision resistance refers to the computational infeasibility of finding two distinct inputs that produce the same hash output. Pre-image resistance is the property of a hash function such that, for a given output y, it is computationally difficult to find any input x such that $h(x) = y$.

1.5 Near-Field Communication

Near field communication (NFC) is a technique that allows two electronic devices or a device and an NFC tag to establish communication when they are in close physical proximity. Figure 1.6 depicts a smartphone communicating with a paper NFC tag, demonstrating wireless data transfer using NFC technology.

Figure 1.6: Near-Field Communication

NFC has advantages over Bluetooth, RFID, and other communication methods for carrying out sensitive transactions. NFC tag doesn't need a power source – it is passive, and a terminal can read or write information inside it. A Bluetooth tag requires power to broadcast its signal because radio frequency (RF) from another Bluetooth device is insufficient to power a Bluetooth chip. Another advantage of using NFC is the short range of communication (less than 4 cm) between NFC devices/tags, making it a good choice for exchanging sensitive data. In the case of Bluetooth, data theft may occur because of the long distance (even up to 100 meters) communication range for the data transfer. NFC is faster and easier to set up than a Bluetooth connection. In contrast to RFID, which generally supports one-way wireless communication between a passive tag and an active reader, NFC enables two-way data exchange, making it suitable for more advanced applications like card emulation and peer-to-peer (P2P) communication.

NFC technology supports three primary modes: card emulation, peer-to-peer communication, and reader/writer functionality. In card emulation mode, the NFC device appears to an external reader much the same as a traditional contactless smart card. The peer-to-peer mode enables data exchange between two NFC devices. In reader/writer mode, the device can read data from NFC-compliant tags and write data to them.

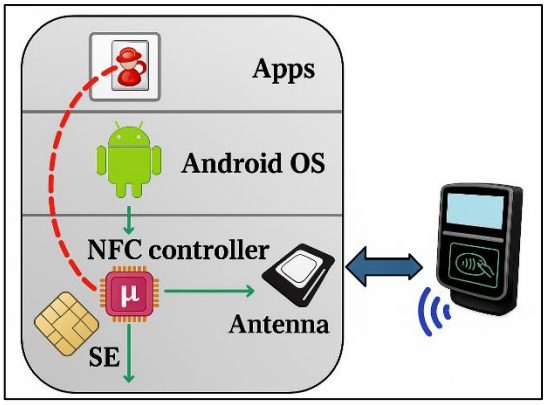


Figure 1.7: NFC architecture

The diagram shown in Figure 1.7 illustrates the architecture of Near Field Communication (NFC) on a smartphone, specifically in an Android environment. It shows the interaction between mobile apps, the Android operating system, the NFC controller, and the Secure Element (SE). Applications communicate with the Android OS, which then communicates with the NFC controller. The NFC controller is responsible for managing the communication between the mobile device and external NFC readers (such as a payment terminal) via the antenna. The Secure Element (SE), which can be an embedded chip or an external security module, securely stores sensitive payment credentials. The diagram highlights both the data flow within the device and the wireless communication with an external NFC payment reader.

1.6 Blockchain

A blockchain is a list of immutable data records linked cryptographically. It is a decentralized digital ledger designed to record transactions securely, transparently, and permanently between parties. Each block in the chain holds a timestamp, transaction details, and a cryptographic reference to the previous block, ensuring continuity and integrity. To incorporate a new block, participants must use their private keys and computational power to solve intricate cryptographic puzzles. Once a solution is verified, the block becomes part of the chain and remains permanently stored across the network. Blockchain technology provides greater fault tolerance and availability because the data is distributed across multiple nodes. These characteristics make blockchain particularly valuable in cybersecurity, where its structure helps protect sensitive data from unauthorized access. Its record-keeping system gives each node insight into any data manipulation, exposing potential cyberattacks.

Shadab et al. [96] proposed a blockchain-based framework for secure and reliable management of student academic records. The research presented a student document issuance and verification process using blockchain.

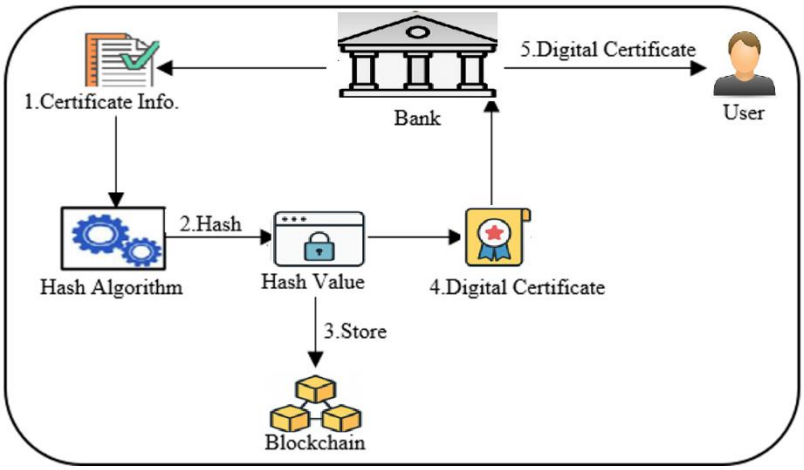


Figure 1.8: Digital certificate issuance process in blockchain

Figure 1.8 shows an example of the digital certificate issuance process in the blockchain network.

Certificate Issuance - A certificate authority (CA) issues a digital certificate to an individual or organization. The certificate contains information such as the holder's name, public key, and expiration date.

Blockchain Registration - The certificate is registered on a blockchain network by creating a transaction. The transaction includes the certificate details and is broadcast to the blockchain network.

Transaction Verification - Nodes within the blockchain network verify transactions using advanced computational processes. They ensure that each transaction is legitimate and that the certificate is correctly formatted.

Information in a blockchain is hashed and stored, meaning that before being added to a block, data is transformed into a unique cryptographic hash, which is then stored on the blockchain to maintain the accuracy and permanence of the information.

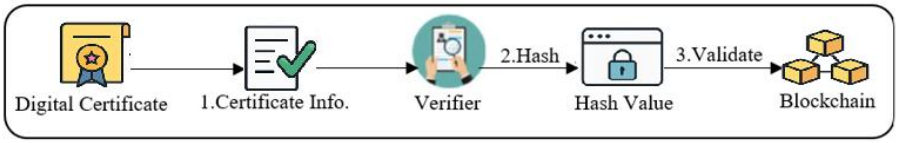


Figure 1.9: Digital certificate validation process in blockchain

Figure 1.9 shows an example of the digital certificate validation process in the blockchain network.

Certificate Validation - When a user attempts to access a system or service, the blockchain network is queried to validate the certificate. The blockchain network checks the certificate's details against the registered information.

1.7 Machine Learning

Machine learning (ML) algorithms use data to enhance the ability to perform specific tasks over time. It creates models from training data that can make predictions or decisions without the need for explicit programming. Various standard machine learning algorithms are used for different tasks. For example, machine learning can be leveraged to detect ever-evolving cyberattacks effectively [7, 8, 36]. Machine learning models can be developed using historical cyberattack data, allowing these models to learn patterns and subsequently detect or respond to similar threats. Datasets are an integral part of the field of machine learning. The machine learning models are trained and tested using the datasets. The dataset required for a specific task can be generated or obtained from publicly available repositories. A feature or attribute in a dataset represents a measurable piece of data used for analysis.

Several machine learning software programs, like Scikit Learn, PyTorch, TensorFlow, Weka, etc., are available on the market. Weka [9] is an open-source data preparation, classification, regression, clustering, association, and visualisation tool. It has an easy-to-use user interface and an extensive collection of machine-learning algorithms for performing data mining tasks. Some popular machine learning classification algorithms are J48 [42], Random Forest [61], Multi-Layer Perceptron (MLP) [76], Bayes Net [51], etc. The J48 algorithm, J standing for Java, builds decision trees based on training data and is used for classification. The random forest model is made up of multiple decision trees that work together as a collective ensemble. It combines the predictions from each tree and determines the final result based on the majority of votes. The MLP is a type of neural network that includes three layers: input, hidden, and output. The input layer receives the data to be processed, while the output layer carries out the desired task, such as classification. The hidden layers are placed between the input and output layers and serve as the computational engine of the MLP. Bayes Net or Bayesian network is a type of probabilistic graphical model that uses directed edges in a graph to represent conditional dependencies between variables.

1.8 Contextual Information

In the security domain, supplemental contextual information has been used in some applications and proposed in several research works to improve security decisions and detect attacks. Various organizations are generating incredible amounts of data, and preserving the privacy of the data is an important security aspect. Some of the research works [23, 68, 69] have proposed the use of contextual information in a system to facilitate privacy preservation. In these works, an ontology data model has been developed to bring context awareness to a privacy-preserving system. Al-Turjman et al. [33] proposed an access control approach tailored to contextual information within healthcare applications in the Industrial Internet of Things (IIoT) environment. It reduces the risks of identity theft/loss by considering the dynamics of the inhabitants' behaviors and their access and usage patterns. Francisco et al. [34] presented a novel intrusion detection system (IDS) that exploits contextual information to detect multistage attacks (MSAs). A MSA unfolds through several sequential phases or actions, each of which may appear benign when considered in isolation, but when all are executed sequentially, the attack succeeds. A novel contextual framework was introduced by Ahmed et al. [10], incorporating various attack prediction models, which were integrated with IDSs to enhance cyberattack detection. Here, the contextual framework aimed to assist in identifying both known and unknown attacks. Wilson et al. [103] proposed using a physical context-based authentication system to counter external attacks. Here, the authors showed that entities sharing a common physical context can leverage it to create a secure communication channel, thereby safeguarding one another against external threats.

1.9 Trusted Platform Module

In a general-purpose smartphone, there is often a need to store secure data such as cryptographic keys. A Trusted Platform Module (TPM) is a dedicated security chip embedded in a device's motherboard or processor and uses cryptography to store sensitive information. The usual implementation of mobile TPMs is protected environments through hardware-based trusted execution environments (TEE) [16, 38, 46], like the ARM TrustZone. In ARM TrustZone, the TEE operates a distinct and isolated TrustZone Operating System (TZOS) alongside the Android OS, and the device vendors design and

implement the cryptographic functions [6]. In the case of iOS, the Secure Enclave subsystem fabricated within the system on chip (SoC) is designed to keep sensitive data.

1.10 Threat Modeling

Threat modeling is a process by which potential threats in a system are identified, and countermeasures are defined to prevent or mitigate the effects of threats to the system. Threat modeling helps to design and assess the security of a system.

1.11 Thesis Structure

The thesis is organized into seven chapters. Chapter 1 presents an overview of the various authentication methods and security-related attacks. It also introduces the key technologies and concepts employed in this research. The remainder of this thesis is organized into six chapters:

Chapter 2 (Literature Survey) comprehensively reviews existing security solutions, specifically emphasizing authentication mechanisms. The discussion highlights the advantages and disadvantages of various authentication approaches, providing a foundation for understanding the research context. Additionally, this chapter examines relevant works related to our proposed authentication solutions, setting the stage for developing the framework presented in this thesis.

Chapter 3 presents our work on a novel authentication solution based on a QR code and a multi-dimensional hash chain. This is a One-Time Password (OTP) based solution.

Chapter 4 presents our work on a novel authentication solution leveraging contextual QR codes. The proposed approach generates an OTP based on dynamic contextual information embedded within the QR code.

The proposed work in Chapter 5 demonstrates a novel authentication solution for Automated Teller Machines (ATMs), eliminating the need for physical ATM cards. The proposed system leverages Near Field Communication (NFC) and Blockchain technologies to provide a secure, cardless, and contactless authentication experience.

In Chapter 6, we have proposed a solution in which the context of users and devices is considered during authentication to detect anomalies and security-related attacks. An independent machine learning network processes and analyses the contextual information to determine the credibility of the source requesting authentication.

The final Chapter 7 summarizes the key contributions of this research. It also outlines potential avenues for future research, highlighting opportunities to further enhance and expand the proposed authentication solutions.

Chapter 2

Literature Survey

2.1 Hash Chain

2.1.1 Lamport and S/Key One-Time Password System

Lamport [57] suggested an OTP-based scheme for user authentication in the early 1980s. Based on this idea, a prototype software implementation of the S/KEY [78] one-time password was also developed. The scheme uses a one-way function to generate a finite series of OTPs based on a chosen password. A one-way function can be efficiently computed in the forward direction, but reversing it is computationally impractical. It is applied to a chosen password a certain number of times to generate different OTPs.

Let x denote the chosen password. The finite series of OTPs is computed as -

$$F^{n-1}(x) \text{ ..... } F(F(F(x))),\ F(F(x)),\ F(x),\ x$$

The value $F^n(x)$ is initially stored at the server end. The server does not store the actual password x. In the i^{th} authentication, the client calculates the OTP as -

$$OTP_i(x) = F^{n-i}(x)$$

The client sends this OTP to the server for authentication. On receiving the OTP, the server performs a hash of it and compares it with the stored OTP of the previous login.

$$F(OTP_i(x)) = F(F^{n-i}(x))$$

If there is a match, authentication is validated, and the server replaces its stored value with the current OTP, i.e., $F^{n-i}(x)$. In the subsequent authentication, the client increments i by one and computes the next OTP. Thus, the OTPs used for authentication are $F^{n-1}(x)$ to x. Also, if the server and client are out of sync, the server could send a challenge i corresponding to which the client needs to calculate the OTP. After n authentications, a new value of x needs to be generated for further n time authentications.

Generating a new password after a certain number of authentications makes the scheme constrained. Also, the host could be impersonated, as there is no means to verify the authenticity of information sent by the host to the client. Calculating the initial values of the chain demands significant computational power, making the system impractical for resource-constrained devices. Moreover, manually typing a long OTP is very inconvenient.

2.1.2 Scheme of Bicakci and Baykal

Bicakci and Baykal [52] proposed infinite-length hash chains using a public key algorithm. Let d denote the private key, e denote the public key, s denote the seed value, and $A^N()$ denote the public key algorithm A applied N times. The hash chain of infinite length, generated from the initial seed s is –

s, A(s, d), A^2(s, d), …… A^{N-1}(s, d), A^N(s, d), ……

The client calculates OTP for the i^{th} authentication as –

$$OTP_i(s) = A^i(s, d)$$

The generated OTP is transmitted to the server for authentication. Upon receipt, the server calculates –

$$s = A^i(OTP_i(s), e)$$

If the shared secret seed matches, then the authentication will be successful. The drawback of this method is that as the number of cascaded exponentiations grows, the computational complexity also rises, thus making it challenging to implement on devices having limited computational capacity. The previous OTP can be stored on both the client and server ends during each authentication to facilitate faster algorithm implementation. However, if the last OTP is lost or damaged, a large exponentiation might be required to generate or verify the subsequent OTP. Moreover, manually typing a long OTP is very inconvenient.

2.1.3 Scheme of Eldefrawy et al.

Eldefrawy et al. [63] scheme extends Lamport's idea of a finite series of hash chain generation for authentication. In this scheme, an infinite-length hash chain is generated by employing two different one-way hash functions, $h_A()$ for updating the seed and $h_B()$ for generating the OTP.

$$OTP(x, y) = h^y_B(h^x_A(seed))|_{x:1->\infty, y:1->\infty}$$

At the time of authentication, the user first provides the current status of the OTP to the server. The current status enables the server to align its seed with the client's, ensuring both share an identical seed value. Next, the server challenges the user with new indexes (i.e., x and y values) based on which the user computes the OTP. The OTP is subsequently transmitted to the server for verification, and if found correct, the server confirms the user's identity.

This scheme lacks strong resistance against prediction-based attacks. In particular, the likelihood of correctly guessing the next challenge is $1/m^2$, where m denotes the range of x & y coordinates of a challenge. Securely storing the seed on the client end is an additional risk. Here, a chain of hash functions is applied depending on the challenge x & y, thereby increasing computational overhead during authentication. Moreover, it is very inconvenient for users to type a long OTP manually.

2.1.4 Other Hash Chain Techniques

Goyal et al. [101] developed a scheme called the N/R one-time password system. Here, the server aids the client's computation by placing breakpoints within the hash chains. Compared to Lamport's OTP scheme, the N/R system allows for significantly more client logins before requiring system reinitialization, but the constraint here is that it still involves system reinitialization. In Park's proposed work [18], a novel hash chain was developed to enable infinite OTP generation without the need for a pre-shared secret between the communicating parties. This paper addresses the weaknesses of a single long hash chain by generating multiple short hash chains, allowing deferred construction and infinite OTP generation. This delayed construction reduces the initialization time of the algorithm; however, there is an increase in the time cost of OTP generation and verification.

T/Key, a time-based one-time password system, was proposed by Kogan et al. [24]. This system aims to eliminate the server compromise attack by not storing any secrets on the server. However, OTP generation and verification efficiency are low in T/Key.

Pietro et al. [82] proposed infinite-length chains using a public key algorithm. In this scheme, the shared secret between the server and client is not required, so it is effective against server compromise attacks. However, the computational complexity of implementing a public key algorithm is much greater than that of lightweight hash functions.

2.2 RSA SecurID Authenticator

RSA SecurID [44] is a hardware device that generates time-synchronized token codes every sixty seconds. An RSA SecurID key fob is shown in Figure 2.1.

Figure 2.1: RSA SecurID key fob

A token code is computed as a cryptographic function of the current time and a shared secret between the corresponding SecurID and the authentication server. This token code acts as an OTP for the user to authenticate. The same token code is generated at both the authentication server and client end. Thus, when a client sends the token code to the authentication server, it validates the correctness of the token code.

RSA SecurID is an efficient OTP generation technique. However, its usage is limited because of the device procurement cost for each user and the need to carry an additional device for authentication. Moreover, there is a lag in setup time as the RSA SecurID device needs to be shipped to the user, and the device needs to be replaced with a new one every few years.

2.3 NFC-based Authentication Solutions

H Lee et al. [37] have proposed an authentication solution using a digital signature and NFC card emulation on Android. During authentication, the server sends a nonce to the client. The user's mobile device reads the nonce via the NFC channel, signs it using its

private key, and returns the signed nonce to the server for validation. The study emphasizes the advantages of this method over conventional password-based authentication. However, there is an added overhead associated with securely storing the private key and generating a digital signature on the mobile device for every authentication attempt.

R.M. Ranasinghe et al. [85] have proposed a device design that integrates fingerprint authentication with RFID or NFC tags to overcome some security risks using RFID and NFC technologies. This device exchanges data via an RF signal with an NFC/RFID reader or writer. A primary limitation of this proposal is the requirement for users to carry an additional sophisticated device.

2.4 ATM-based Authentication Solutions

An NFC-enabled solution has been proposed by Mandalapu et al. [3] to authenticate at an ATM. An NFC-enabled smartphone reads from an NFC tag affixed to the ATM. This launches a website on the user's smartphone, wherein to authenticate, the user needs to input a pattern password corresponding to their unique username. However, this system has notable limitations, including the continued reliance on physical ATM cards for transactions, the requirement of remembering the pattern password, and the potential delay it introduces in the authentication process at the ATM terminal.

Several recent research works have proposed using biometrics, such as fingerprints, facial recognition, etc., to authenticate into the ATM system [4, 5, 22, 86, 92]. However, implementing biometric authentication often necessitates additional hardware, such as high-resolution scanners or cameras. When a customer shares biometric information with a bank server, if it is breached, the impact is either non-recoverable or requires significant time and effort to mitigate.

R. Pote et al. [87] proposed the use of a smart mobile banking application (SMBA) for withdrawing cash from ATMs. This method does not require an ATM card to authenticate in the ATM network. The customer will enter the amount to be withdrawn in the SMBA. The bank server verifies the amount, and then the SMBA will authenticate the customer with a password. At this stage, a pre-commit of this transaction occurs in the bank database. At the ATM kiosk, the pre-committed transaction details are communicated via a QR code from the SMBA to the ATM terminal. If the data is correct, the user must enter their PIN

for further verification. Thus, users need to remember both a password and a PIN to perform a cash transaction.

In existing NFC-based authentication methods at ATMs, Host Card Emulation (HCE) is used to emulate a traditional payment card on a mobile device. HCE enables communication between the device and the ATM's NFC reader without the need for a physical card. Typically, HCE is combined with additional secure methods such as a PIN, biometric authentication, or other verification techniques to strengthen security. However, in HCE, sensitive authentication information is transferred during the authentication process, which introduces vulnerabilities. One notable risk is the potential for attacks where a rogue NFC reader is covertly placed at kiosks to capture sensitive data during a transaction [95].

2.5 Blockchain-based Authentication Solution

L. Zhang et al. [58] proposed a decentralized user authentication model leveraging blockchain technology. In this framework, users register their identities on the blockchain, keep their encrypted personal data in external (off-chain) storage, and use smart contracts to assign specific access rights to various websites or applications.

Another proposed approach, known as "bubbles of trust", presents a decentralized system aimed at enhancing the secure identification and authentication of devices [67]. This model leverages the inherent security benefits of blockchain technology and creates secure virtual zones ("bubbles") where objects can mutually recognize and authenticate one another.

D. Nguyen et al. [30] proposed using blockchain technology to issue immutable digital certificates. The authors compare this work with the Blockcerts application (developed by the Massachusetts Institute of Technology, and considered one of the early adopters of blockchain for certificate issuance and verification) and a few other works to claim an improved digital certificate retrieval process. A working prototype of the system has been implemented and utilized for various short-term academic programs at the university.

2.6 Machine Learning-based Security Solutions

2.6.1 Behavioral Biometrics

Behavioral biometrics uses the user's behavior while interacting with their device. It focuses on identifying individuals based on distinct, measurable behavioral traits exhibited during routine activities. Behavioral biometrics has been implemented in some devices and systems [11, 31, 47, 59] based on machine learning.

2.6.2 DDoS/DoS Attacks

Several machine learning approaches have been used to identify distributed denial-of-service (DDoS) traffic; these approaches can be divided into two categories: supervised and unsupervised. Alkasassbeh et al. [70] used the MLP, Random Forest, & Naive Bayes algorithms on the dataset to classify various types of DDoS attacks, including HTTP-Flood, UDP-Flood, Smurf, and SIDDOS. The dataset consisted of 27 features, and it was demonstrated that the MLP classifier achieved the highest accuracy, reaching 98.63%. Parvinder et al. [79] used the J48, Naive Bayes, Random Forest, and MLP algorithms on the same data source as [70] to detect DDoS attacks. It was shown that J48 has the highest accuracy (98.64%) compared to the other three algorithms. Shreekh et al. [93] used the "Canadian Institute for Cybersecurity Intrusion Detection Systems (CIC IDS) 2017 dataset" to study the denial-of-service (DoS) attack detection accuracy of Random Forest and MLP algorithms. The dataset consists of 84 features. The experimental result showed that Random Forest performed better than MLP for different data split percentages of training and testing data. Kurniabudi et al. [54] applied the information gain feature selection technique on the CIC IDS 2017 dataset to select the most relevant attributes. Then they used various classification algorithms, including Random Tree, Random Forest, Naive Bayes, Bayes Net, and J48, towards intrusion detection. The study's results specifically showed that Random Forest achieved a maximum accuracy of 99.86% when using 22 selected features, while the J48 classifier slightly outperformed it with an accuracy of 99.87% using 52 features. In the study done by Kimmi et al. [55], ML algorithms such as logistic regression and naive bayes were employed on the CAIDA 2007 dataset to detect DoS attacks and normal scenarios. The original dataset comprising 20,090 records is categorized into a 70:20:10 ratio corresponding to 70% training, 20% testing, and 10%

validation. Here, Logistic Regression accuracy ranges from 99 to 100%, while Naive Bayes demonstrated an accuracy ranging from 98% to 99%. However, the CAIDA 2007 dataset lacks the modern forms of DoS attacks present in the CIC IDS 2017 dataset.

2.6.3 Brute Force Attacks

Stephen et al. [94] proposed a Convolutional Neural Network-based approach for detecting SSH brute-force attacks within network traffic. The CIC-IDS 2018 dataset, having 78 features, was transformed and used for experimentation, and an accuracy of 94.3%, a precision of 92.5%, a recall of 97.8%, and an F1-score of 91.8% in detecting SSH-brute force attacks were recorded. The detection of SSH and FTP brute-force attacks using the Long Short-Term Memory (LSTM) deep learning technique was investigated by Md Delwar et al. [71]. The experiment was performed on the CIC-IDS 2017 dataset with all 79 features, 80% training data, and 20% testing data to evaluate the effectiveness of the LSTM ML algorithm. Here, for SSH and FTP brute-force attacks, the accuracy of the LSTM model was derived to be 99.88%, but the execution time of the algorithm is not mentioned in the paper. Username enumeration attack detection on the SSH protocol was investigated by Abel et al. [12] using various ML algorithms - random forest (RF), naive bayes (NB), k-nearest neighbor (K-NN), and decision tree (DT). The generated dataset consists of 36,273 raw packets, each containing 25 features, and the result of the experiment showed that KNN has an accuracy of 99.93%, followed by RF with 99.92%, DT with 99.88%, and NB with 95.70%. Muhammad et al. [72] studied 5 machine learning algorithms - gaussian naive bayes, random forest, decision tree, linear support vector, and multilayer perceptron for several attacks on the CIC-IDS2018 dataset. Specifically, the brute-force attack had a significant imbalance, with only 362 malicious traffic flow instances, using just ten features, and poor detection performance was recorded across all five classifiers.

Chapter

3

One-Time Password Authentication based on QR Code and Hash Chain

3.1 Overview

This chapter will demonstrate a novel authentication technique that uses a multidimensional hash chain and a QR code. In the proposed approach, the server generates a dynamic challenge by selecting a random path within an n-dimensional hypercube, where each dimension is associated with a distinct cryptographic hash function. The client then derives the OTP by sequentially applying specified hash functions to a synchronized seed value. This multidimensional strategy significantly expands the challenge space, thereby increasing the entropy of the OTP and making it resistant to attacks such as pre-play and replay. QR codes have been used to facilitate the exchange of information between a client terminal and an end-user device without manually entering the data. The proposed algorithm differs from existing OTP systems in two key ways: using a multidimensional hash chain architecture and incorporating QR codes for secure and dynamic challenge transmission. The suggested algorithm does not require a system re-initialization and is more secure and computationally efficient than some of the other similar authentication schemes. Security analysis and threat modeling show that the proposed authentication scheme effectively enhances a secure system's security level. The approach has been

validated using synthetically generated challenge datasets and standard cryptographic hash benchmarks, with experiments conducted on an Intel Core i3-based system. The performance evaluations reveal that the challenge generation and OTP computation algorithms execute rapidly, operating within microseconds even on low-end devices.

3.2 Proposed Solution

In this section, the core concept behind the proposed solution is explained in detail. Also, the steps involved in the registration and authentication process are explained along with the algorithms.

3.2.1 Working Principle

A hypercube is defined as the n-dimensional extension of a square when n = 2 and a cube when n = 3. In Figure 3.1, a 3-dimensional hypercube (a cube with eight nodes and twelve edges) is depicted for easy visualisation and explanation of the working principle of the proposed solution.

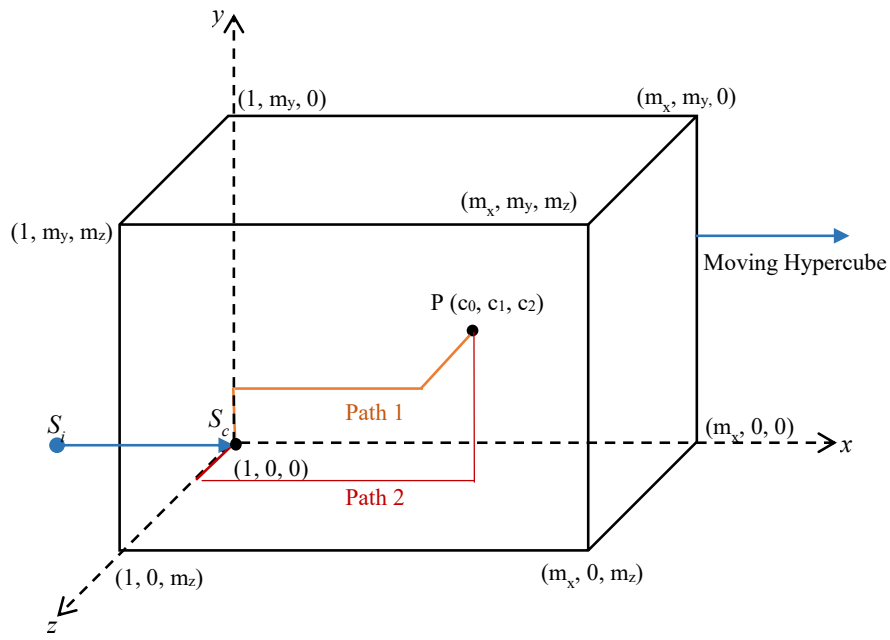


Figure 3.1: A 3-dimensional hypercube having eight nodes and twelve edges

The following representations are used:

S_i → initial seed

S_c → current seed

$m = \{m_i\}$, $0 \leqslant i < n \rightarrow$ bound of each side of the hypercube

$h = \{h_i\}$, $0 \leqslant i < n \rightarrow$ n hash functions; one along each dimension of the hypercube

$P (c_o, c_1, \ldots, c_{n-1}) \rightarrow$ coordinate point inside the hypercube

In the 3-dimensional hypercube shown in Figure 3.1, the bounds are represented by m_x, m_y, and m_z. A unique cryptographic hash function is assigned along each dimension, represented by h_i. In this thesis, a challenge-response scheme is proposed, wherein, based on a challenge thrown by the server, the response would be an OTP generated by a client. The challenge here would be a forward path from the starting coordinate of the hypercube to a random point inside the hypercube. There could be several paths for a given point inside the hypercube, and each possible path is a candidate for a challenge. For example, in Figure 3.1, two paths (Path 1 and Path 2) are sketched to the point $P (c_o, c_1, c_2)$, which lies inside the hypercube. The challenge path denotes the indices from the group of hash functions based on which OTP will be calculated. The hash functions are consecutively applied on the current seed to derive the OTP.

After every authentication, the current seed is changed by applying a cryptographic hash function to the current seed. So after k^{th} authentication, ideally, the same hash function is applied k times to the initial seed to get the current seed.

Thus, $S_c = h_o^k(S_i)$, where h_o is the hash function for the seed update.

With each authentication, the seed update essentially moves the hypercube to a new position (in the coordinate system). An extensive set of possible challenges exists, based on which a temporary authentication password needs to be derived by the client. The one-time password computed by the proposed method is different at all times.

3.2.2 Registration

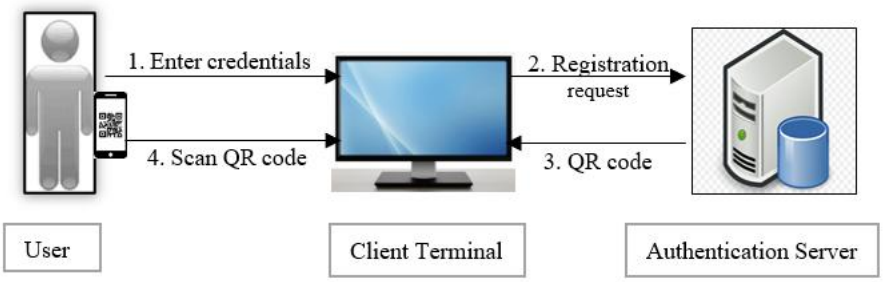


Figure 3.2: Registration process

The registration process is shown in Figure 3.2. Users should install the custom authentication application (from the Google Play Store, Apple App Store, or any other source) on their smartphone to use the proposed authentication system. The user chooses a username (u) and a strong password (pwd), the first authentication factor, and enters it in the client terminal. This information is sent to the identity verification server. The server stores the username as received and applies a hash to the password (with salt) to store the password in hash form (pwd_h) in the database. The hashing of passwords makes it difficult to derive the original password from the hashed form. The server then computes a unique initial seed (S_i) based on the user's details and the date and timestamp of registration. The server randomly chooses n different hash functions (denoted by h) for authentication. The computed seed (S_i) and the set of hash functions (h) are encoded as a QR code and sent to the client terminal.

At the server end, the following data is maintained corresponding to each of the registered clients:

$u \rightarrow$ username

$pwd_h = h(pwd) \rightarrow$ hash password

$S_i \rightarrow$ initial seed

${}^{s}S_c \rightarrow$ current seed of the server, initialised to S_i

$h = \{h_i\}, 0 \leqslant i < n \rightarrow$ n hash functions

$C_s \rightarrow$ counter, initialised to 0

Upon receiving the generated QR code at the client terminal, it is required to be scanned by the user using the custom authentication application installed on their smartphone. The mobile application decodes the QR code and retrieves the initial seed and hash functions.

In the smartphone of the user, the following data is securely stored via the custom authentication application:

$S_i \rightarrow$ initial seed

${}^{c}S_c \rightarrow$ current seed of the client, initialised to S_i

$h = \{h_i\}, 0 \leqslant i < n \rightarrow$ n hash functions

$C_c \rightarrow$ counter, initialised to 0

3.2.3 Authentication

A user needs to be authenticated to access various systems and confidential data. Thus, it is paramount to safeguard the authentication phase from vulnerable security attacks. Information exchange occurs between the authentication server and the user during the identity verification procedure to verify the user's identity. The proposed authentication process is depicted in Figure 3.3.

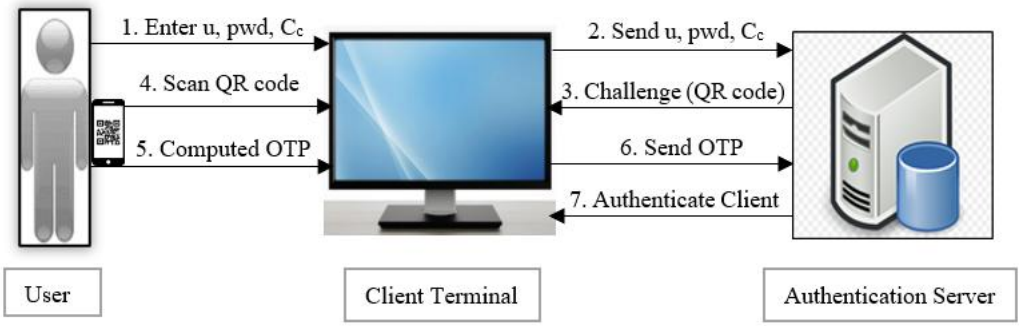


Figure 3.3: Authentication process

3.2.3.1 First Factor of Authentication

The proposed authentication mechanism uses a two-factor authentication to enhance the system's security level. The user has to enter the username (u) and the registered password (pwd) at the client terminal, which is the first authentication factor. The seed's current counter value (C_c) is read from the mobile application and input along with the credential. The authentication server receives the submitted data for validation. At the server end, the received password is hashed along with the salt (as it was done during registration).

Computed hash password, $pwd_h = h(pwd)$

The computed password (pwd_h) is compared against the password stored in the server's database corresponding to the username. If the credential is valid, the server proceeds with the generation of the challenge, or else the authentication request is invalidated. The server temporarily stores C_c, which will be used in the second authentication factor.

3.2.3.2 Authentication Server - Challenge Generation

A random point $P(c_o, c_1, c_2 \ldots c_{n-1})$ is chosen inside the n-dimensional hypercube. The challenge would be a forward path from the starting coordinate of the hypercube to the

random point selected inside the hypercube. There could be several paths for a given point inside the hypercube, and the length of any of the possible paths is given by –

$Path_l = (c_o + c_1 + c_2 \ldots + c_{n-1})$

The pseudo-code of the challenge generation algorithm is given below.

Notations used –

n → dimension of the hypercube

$m = \{m_i\}, 0 \leqslant i < n$ → bound of each side of the hypercube

path → a random path from the origin to a point inside the hypercube

Algorithm 1 *Generate Challenge*

Input: *n, m*

Output: *path*

```
path ← []
for i = 0, . . . , n − 1 do
        pick an integer c : 1 ⩽ c ⩽ m[i] randomly
        add i to path, c number of times
end for
// Use of existing shuffling algorithm
for i = length(path) − 1, . . . , 1 do
        pick a random integer k : 0 ⩽ k ⩽ i
        swap(path[k], path[i])
end for
return path
```

The input to the algorithm is the dimension of the hypercube and the bounds of each side of the hypercube. The output will be a random path from the origin to a point in the hypercube, and this output is the challenge that will be sent to the client to derive the OTP. The existing state-of-the-art shuffling algorithm proposed by Durstenfeld [83] generates a random permutation of a finite sequence, i.e., the algorithm shuffles the sequence. It has linear O(n) time complexity. This algorithm has been used in our challenge generation algorithm to generate a random path out of all the possible paths for a given point inside the hypercube. The challenge is saved at the server end against the client's username, and

the challenge generation time is recorded. This challenge is encoded as a QR code and sent to the client terminal.

3.2.3.3 Mobile App - Derive OTP

The system user reads the QR code shown at the client terminal via the custom mobile application. The challenge sent by the server is extracted from the QR code and used to derive the OTP. The pseudo-code of the OTP generation algorithm is given below.

Notations used –

path → challenge thrown by the server

${}^{c}S_c$ → current seed of the client

$h = \{h_i\}, 0 \leqslant i < n$ → n hash functions

h_o → hash function to update seed

C_c → current counter

Algorithm 2 *Generate OTP*

Input: *path,* ${}^{c}S_c$, *h*

Output: *OTP*

OTP ← ${}^{c}S_c$

for each *i* ∈ *path* ***do***

OTP = $h_i(OTP)$

end for

return *OTP*

The input to the algorithm is the challenge received from the server, the current seed value, and the hash function sequence that was saved in the mobile application during registration. The current seed value is synchronized between the client and the server so that the same OTP will be computed at both ends for a given challenge and set of hash functions. Cryptographically secure hash functions have been used to ensure the confidentiality of the OTP. The OTP is the output of the algorithm that the authentication server will validate to authenticate the user. The algorithm assigns the OTP an initial value equal to the current seed value. The path received as the challenge contains the indices of the hash functions that must be used sequentially on the current seed value to obtain the OTP.

The selected cryptographic hash functions are applied in the prescribed order:

$$\mathrm{OTP} = h_{pk}(\, h_{pk-1}(\ldots h_{p1}(\, {}^{c}S_c\,)\ldots))$$

where pi are indices representing the hash functions applied in sequence for a challenge path of length k.

The iterative application of h_i for each element in the path provides a high level of entropy, making it computationally infeasible for an attacker to predict or reverse-engineer the OTP.

After computing the OTP, the mobile app updates the counter and seed value as –

${}^{c}S_c = h_o({}^{c}S_c)$

$C_c = C_c + 1$

As per the system's requirements, there could be different ways to communicate the computed OTP from the mobile app to the client terminal. The length of the generated OTP by the hash chain is long enough and might not be practical for the user to enter manually. In one solution variant, it is proposed that the user enter only the first six OTP characters at the client terminal. The client terminal sends the username and OTP to the authentication server for verification.

This enhanced OTP generation process introduces a secure, efficient, and computationally feasible authentication mechanism. The random challenge path introduces variability in OTP derivation, eliminating predictability. The forward hashing ensures that previous OTPs cannot be reconstructed.

3.2.3.4 Authentication Server - Verify OTP

Upon receiving the OTP, the server obtains the challenge generation time corresponding to the username. The authentication request is rejected if the difference between the OTP receipt and the challenge generation time exceeds the acceptable duration limit. Otherwise, a temporary seed S_t is calculated as below –

$S_t = h_o^{(Cc-Cs)}({}^{S}S_c)$ if $(C_c - C_s) \geqslant 0$

$S_t = h_o^{Cc}(S_i)$ if $(C_c - C_s) < 0$

The current counter value of client C_c was temporarily saved at the server end during the first authentication factor. Ideally, the value $(C_c - C_s)$ would be zero for the very first authentication of a user. Hence, the temporary seed would be $S_t = {}^{S}S_c$.

Ideally, at each authentication request from the client, the C_c value should be one greater than C_s. So, $S_t = h_o({}^{S}S_c)$, i.e., the seed update hash function should be applied once on the current seed to obtain the temporary seed. If the server counter loses its value due to a server crash or any other failure, it could start from an initial value of 0, and ${}^{S}S_c$ would be initialized to S_i. In that case, the seed update hash function would be applied C_c times on the current seed of the server to get the temporary seed. If $C_c < C_s$ (e.g., mobile loses C_c and starts from 0), the server can still calculate S_t from the initial seed S_i.

The authentication server generates the OTP similarly to the client. The stored challenge corresponding to the username is retrieved. It contains the indices of the hash functions that must be sequentially applied to the temporary seed value to get the OTP. As per the implementation of the system, the combination of the first six characters of the computed OTP is matched against the OTP received from the client. If it does not match, the authentication fails; otherwise, if it matches, the authentication is successful, and the server updates the current seed and counter as –

${}^{S}S_c = S_t$ and $C_s = C_c$

The new values of the current seed and server counter will be used in the subsequent authentication.

3.2.4 Sample Implementation

The detailed example below shows how the initial challenge path is formed, randomized through shuffling, and then used to apply different hash functions to the seed sequentially, resulting in a robust one-time password.

A. Scenario setup:

A set of 3 secure hash functions is considered:

h = {SHA-256, BLAKE2, SHA-512}

A 3-dimensional hypercube is considered where:

- Dimension 0 (associated with SHA-256) has a bound of 3
- Dimension 1 (associated with BLAKE2) has a bound of 2
- Dimension 2 (associated with SHA-512) has a bound of 3

B. Example random selections (within each bound):

- Dimension 0: Randomly pick $c_0 = 2$ (i.e., apply SHA-256 twice)

- Dimension 1: Randomly pick $c_1 = 1$ (i.e., apply BLAKE2 once)
- Dimension 2: Randomly pick $c_2 = 3$ (i.e., apply SHA-512 three times)

C. Generate the unshuffled challenge path:

For each dimension, the dimension index is added the number of times given by c_i.

- Dimension 0: two occurrences → [0, 0]
- Dimension 1: one occurrence → [1]
- Dimension 2: three occurrences → [2, 2, 2]

Concatenating these, the initial (unshuffled) path is: [0, 0, 1, 2, 2, 2]

D. Shuffle the path using Durstenfeld's algorithm:

The shuffling process randomizes the order of hash function applications. An example of random choice after shuffling:

Final shuffled path: [2, 2, 0, 2, 0, 1]

E. OTP generation using the final challenge path:

The current seed is denoted by ${}^{c}S_c$. The hash functions for each dimension are:

- Index 0: SHA-256
- Index 1: BLAKE2
- Index 2: SHA-512

Now, the hash functions are applied in the order specified by the final challenge path [2, 2, 0, 2, 0, 1]:

BLAKE2 (SHA-256 (SHA-512 (SHA-256 (SHA-512 (SHA-512 (${}^{c}S_c$))))))

The resulting value is the final OTP used for authentication.

3.2.5 Demonstration of the Proposed System

This section provides a hands-on illustration of how the proposed system is implemented. A sample demonstration of the application built based on the proposed solution is shown in Figure 3.4.

Figure 3.4: Demonstration application of the proposed solution

A. Registration

During registration, the server generates and sends an initial seed and n distinct hash functions to the client. The user sets up the mobile application with this information. Sample registration data is shown in Figure 3.5. A set of 4 hash functions - SHA-1, MD5, SHA-512, and BLAKE2 is shown here. It is separated from the initial seed by a pipe.

Figure 3.5: Sample registration data

Initial seed & list of hash names separated by | :

YL!4pnSXsEO70RDGl9ssErGoQdnvGF4jBOTR5r9fYXbr4QeHIipCjR$dX1tsw!S6U4uVPkXnpdXdV3JpP6VYMb|SHA-1 MD5 SHA-512 BLAKE2

B. Authentication

The sample challenge generated during the authentication process is shown in Figure 3.6. The challenge represents indices from a set of 4 hash functions configured during the registration phase.

Figure 3.6: Sample challenge data

Challenge:

0100110220102010000233012011111102100202100111210000021213103012 01

The mobile application iteratively applies the designated hash functions to the current seed to compute the OTP.

Computed OTP: bfe9d633782090f118c38123642deb2c

3.3 Security Analysis and Threat Modeling

In this section, the threat model is established together with the security goals, trust limits, and potential threats that exploit vulnerabilities in the system. In addition, a discussion on the security aspects of QR codes is provided.

3.3.1 Security Objectives

A secure system must fulfil numerous security requirements to ensure stability and resiliency. Identification represents a primary requirement in any confidential system. It is finding the identity of a person or an application requesting access to a secure system. Any anonymous entity should be denied access to a secure system. Non-repudiation is another essential security requirement. It ensures that an entity cannot repudiate the execution of a specific action. The transactions carried out by an authorised user should be accountable.

3.3.2 Trust Boundary

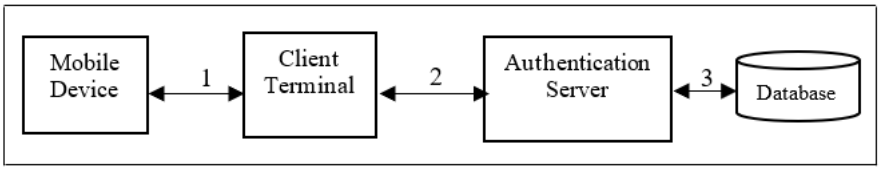


Figure 3.7: Trust boundary

The proposed authentication scheme has four major components (mobile device, client terminal, authentication server, and database) and three major communication channels (marked by numbers 1 to 3) for transferring data, as shown in Figure 3.7.

It is assumed that the mobile device component is uncompromised as long as it is in the user's possession. The confidential information (initial seed and current seed) is stored by the custom authentication application in a mobile device's secure storage in encrypted form. The client terminal, especially in a public place, is vulnerable due to widespread viruses and malicious programs like keyloggers. It is assumed that the authentication server and the database are secure components, and their security is not directly covered in the scope of our discussion. However, the password is hashed (with salt) and persisted in the database.

During authentication, the client terminal exposes the challenge in the form of a QR code that is scanned by the mobile device. The user manually enters the OTP generated by the mobile device in the client terminal. Communication between a client terminal and the authentication server occurs via the Internet and is secured using Transport Layer Security (TLS)/Secure Sockets Layer (SSL). The network pathway between the authentication server and the database is assumed to be secure. Finally, it is also assumed that the registration process is safe.

3.3.3 Threats

The threats could be related to the system's components, communication channels, and authentication protocol.

3.3.3.1 Attacks on the Components

In an adverse scenario, a smartphone can be stolen. Mobile phones have become an indispensable part of daily life, so a stolen mobile phone will be quickly noticed, and users can reset the seed in the authentication server. Also, a criminal cannot easily break into a stolen smartphone if protected using a password, pattern, or biometrics. Moreover, the current seed and password (first authentication factor) are both required to generate OTP.

A hacker could exploit several ways to crash a server, such as by overloading it with traffic. The current seed and counter value could go out of synchronization in a system crash.

However, the proposed authentication solution will continue to work once the authentication server is back. The initial seed is stored by the server in the persistent storage, so after the server crashes, the current seed can be recalculated from the initial seed and counter value sent by the mobile phone.

Using a one-time rather than a static password is more secure for authentication. The client terminal could be exploited by malicious programs like key loggers, asterisk loggers, etc., especially if it is a shared computer. OTP is beneficial against key and asterisk loggers, as the captured temporary password is not valid for later use. Moreover, during authentication, a client terminal does not expose or store confidential information, such as a seed, based on which OTPs can be computed.

3.3.3.2 Attacks on the Communication Channels

SSL or TLS is a standard security technology for establishing an encrypted communication channel between a server and a client. Therefore, the packets cannot be modified or snooped during transmission if the certificate is valid. During authentication, when the mobile device scans the QR code, its communication range with the client terminal is small. Moreover, the QR code is dynamic and does not expose confidential data. Thus, the system's security cannot be breached even if an adversary obtains the QR code.

3.3.3.3 Attacks on the Authentication Protocol

Firstly, the proposed scheme is compared with SMS-based OTP, which is widely used in many secure applications. Recently, there have been compromises on SMS OTPs sent on mobile phones [74, 90]. Using suitable tools, an attacker could intercept the mobile traffic of an end user, including SMS. This would reveal the OTP (via SMS) at the interception point. The proposed method of authentication described in this thesis is quite different and does not rely on mobile networks. Also, the mobile device does not require internet or any other network connection.

In a replay or playback attack, a message that a legitimate user earlier sent is fraudulently repeated. In the proposed solution, the dynamically generated QR code contains a challenge used to derive an OTP. A replay of the already used authentication information, i.e., OTP,

will be rejected by the server, thus making the authentication system secure against replay attacks.

An attacker can impersonate the host and try to determine the initial values of the hash chain by issuing a minor challenge to the client. These values can then be exploited to compute subsequent OTPs (if dependent on previous entries) and impersonate the client to the host. The proposed scheme uses a multi-dimensional forward hashing technique, eliminating this attack type.

In a pre-play attack, an attacker tries to guess the next challenge, then impersonates the legitimate server and presents to the client the predicted challenge. After receiving the corresponding OTP generated by the client in response, the attacker saves it. When the actual server issues the challenge later, the attacker reuses the captured OTP to gain unauthorized access. In the proposed solution, the set of possible challenges for a point inside a hypercube depends on both the hypercube's dimension and the point's range. For instance, in a two-dimensional space, when each dimension spans a range of 2, the maximum number of hash computations is 2+2=4, and the number of distinct paths for point P (2,2) is 6. Similarly, in a two-dimensional space, when each dimension spans a range of 3, the maximum number of hash computations is 3+3=6, and the number of distinct paths for point P (3,3) is 20. The dimensions and range can be set according to the system's security needs.

Shoulder surfing involves the use of direct visual observation to obtain sensitive information, such as watching over a person's shoulder during data entry. This type of attack can also be executed from a distance using tools like binoculars or other visual augmentation devices to discreetly capture information. In the proposed solution, shoulder surfing at the client terminal can reveal a one-time password valid only for the particular authentication, as it changes in subsequent authentications.

3.3.4 Discussion

The comparison of the proposed work with some of the other authentication techniques that use chaining is presented in Table 3.1.

Table 3.1: Comparison of the proposed scheme with related schemes

Features↓ Schemes→	Lamport & S/Key	Goyal et al.	Yeh et al.	Eldefrawy et al.	Bicakci et al.	Proposed scheme
Chaining Method	Hashing	Hashing	Hashing	Hashing	PKC	Hashing
Eavesdropping Protection	Yes	Yes	No	Yes	Yes	Yes
Auth. before re-initialization	Low	Moderate	Low	Unlimited	Unlimited	Unlimited
Pre-play attack resistance	Low	Low	Low	Moderate	High	High
Client computation	High	High	High	Moderate	Very High	Low

Some OTP schemes like Lamport [57], S/KEY system [78], Goyal et al. [101], and Yeh et al. [98] use a hash chain backwards. These systems necessitate re-initialization after a predefined number of authentications, and the number of hash functions executed per authentication is initially very high. Additionally, the Yeh et al. [98] scheme cannot deal with eavesdropping because sensitive information is passed back and forth during authentication. Our proposed authentication method uses a forward hashing technique to compute an OTP based on the challenge thrown by the server. It eliminates the need to re-initialize the system, even after a substantial number of authentications.

Eldefrawy et al. [63] suggested a similar scheme of infinite-length hash chains that calculates an OTP based on a challenge. However, our proposed algorithm is superior to Eldefrawy's work regarding pre-play attacks. An extensive set of possible challenges exists that can be configured based on the dimension and bounds of the sides of the n-dimensional hypercube. Unlike Eldefrawy's work, the user does not need to manually type a long OTP during authentication in the proposed solution.

Hash chains yielded better performance when compared to the schemes of infinite-length chains using a public key algorithm (like that of Bicakci and Baykal [52] and Pietro et al. [82]). Specifically, a significant (approx. 200 times) slowdown was recorded when encrypted using the RSA public key.

The custom authentication application installed in the mobile device does not require internet or other network connectivity. The application calculates OTP based on the challenge read from the client terminal via the QR code. It is advantageous over some

mobile authenticator apps that require internet connectivity to perform authentication or generate an OTP [19].

3.3.5 QR Code Security Considerations

Our system uses QR codes to transmit challenge data between the authentication server and the mobile application. Some of the security measures to mitigate potential vulnerabilities with QR code usage are -

- **Ephemeral Nature and Limited Data Exposure**

The QR codes in the proposed system are dynamically generated for each authentication session. They do not contain long-term secrets such as the initial seed, which remains securely stored on the mobile device. This transient nature significantly reduces the impact of any potential interception.

- **Short-Range Communication**

QR code scanning typically occurs within a close physical range, making it inherently difficult for an attacker to intercept the QR code without proximity to the user.

- **Secure Generation and Transmission**

The QR code generation occurs at the server end and is securely transmitted to the client terminal via TLS/SSL communication protocol. This secure channel protects the QR code's payload from tampering during transmission or display.

- **Replay Protection**

The generated QR code has one-time content. The system tracks whether a particular QR code has already been used. This prevents an attacker from reusing it.

3.4 Performance Evaluation and Applications

This section analyses the proposed algorithm's storage and computational costs and applicability. The computer used to record the performance had the following specifications:

Intel® Core™ i3-5010U CPU running at 2.10 GHz, 2 cores, 4 logical processors, 256 KB of L2 cache, and 3072 KB of L3 cache.

3.4.1 Memory Consumption for OTP Computation

The observations were made using a sample Java method to generate OTP, which utilized standard cryptographic hash libraries.

Notations used –

challenge → array representing challenge, i.e., a path

sc → current seed

h → array representing a sequence of hash functions

Sample method –

```
static byte[] generateOTP(int[] challenge, String sc, String[] h) {
        byte[] bytes = sc.getBytes(StandardCharsets.UTF_8);
        try {
                MessageDigest md;
                for(int i=0; i<challenge.length; i++) {
                        md = MessageDigest.getInstance(h[challenge[i]]);
                        md.update(bytes);
                        bytes = md.digest();
                }
                return bytes;
        } catch(Exception e) {
                return null;
        }
    }
```

When tested with six standard hash algorithms: SHA-512, SHA-384, SHA-256, SHA-224, BLAKE3, and BLAKE2 with a challenge length of 1,000, the memory consumption was about 194 KB of RAM. This low memory footprint demonstrates the suitability of the proposed system for devices with limited resources. The storage efficiency is crucial for resource-constrained mobile and IoT devices.

3.4.2 Time Complexity – OTP Computation

Time complexity refers to the computational time an algorithm requires to execute, expressed as a function of the input size. The graph for the OTP generation algorithm between the number of hash functions applied, i.e., challenge length, and the time taken (in milliseconds) to compute OTP, is plotted in Figure 3.8.

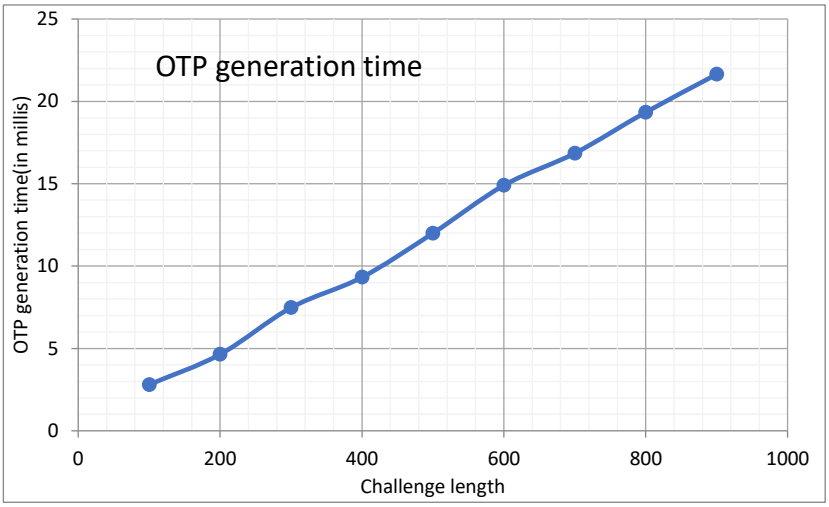


Figure 3.8: OTP computation time for various challenge lengths

The results indicate a linear relationship between the challenge length and OTP computation time. For shorter challenges, the OTP is computed in microseconds; as the challenge length increases, the computation time scales predictably. The linear trend confirms that the algorithm's time complexity is O(k), where k denotes the number of hash operations applied. The low computation overhead indicates that the system remains responsive even when many hash operations are required.

3.4.3 Time Complexity – Challenge Generation

The time taken by the authentication server to generate challenges of different lengths is shown in Table 3.2. It can be seen that testing on a low-end i3-based computer incurs a duration of approx. 197 microseconds (last row), even for a reasonably significant challenge of length 10,000. The authentication server is a high-end machine with a hardware configuration better than the test machine, so the challenge generation overhead should be insignificant.

Table 3.2: Challenge generation time (in milliseconds)

m	n=2	n=4	n=6	n=8	n=10
100	0.01185	0.01238	0.01606	0.01867	0.02317
200	0.00809	0.01616	0.02493	0.03178	0.04052
300	0.01210	0.02493	0.03519	0.04825	0.05883
400	0.01578	0.03529	0.05272	0.06203	0.07847
500	0.01909	0.03962	0.05882	0.07835	0.09836
600	0.02336	0.04753	0.07019	0.09551	0.12001
700	0.02757	0.05572	0.08256	0.10967	0.13719
800	0.03072	0.06298	0.09462	0.12480	0.15795
900	0.03594	0.07062	0.10632	0.14139	0.17887
1000	0.03940	0.07807	0.11727	0.15853	0.19758

The graph for the challenge generation algorithm between the dimension range and the time taken (in milliseconds) to generate the challenge is plotted in Figure 3.9. Different color-coded lines represent different values of n for an n-dimensional hypercube.

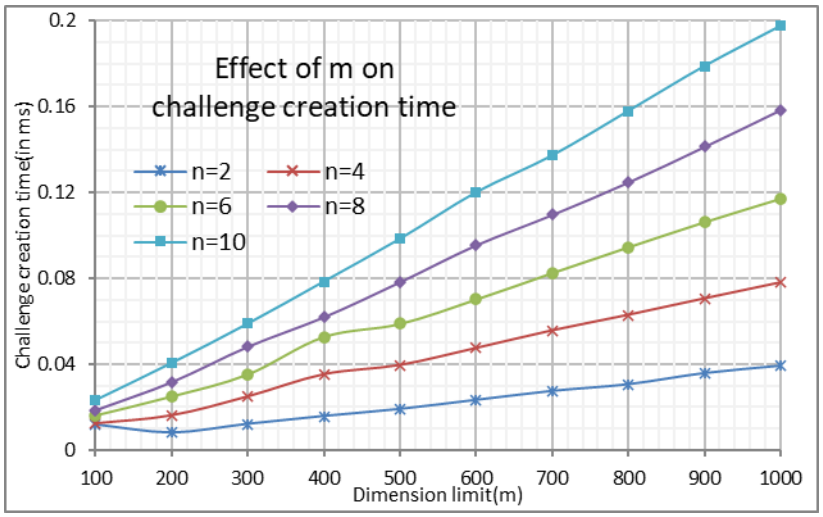


Figure 3.9: Time taken for challenge generation for different dimensions

These findings indicate that the challenge generation overhead is negligible, even as the complexity (number of dimensions and bound values) increases. This low overhead ensures that the authentication server can handle high request volumes without becoming a bottleneck.

3.4.4 Applicability

It has been demonstrated in the performance analysis that the challenge generation process on a low-end Intel i3 system takes only a few milliseconds, even for longer challenge paths. This indicates that the server-side overhead is minimal. In a large-scale deployment, traditional scalability techniques such as load balancing and horizontal scaling of the

authentication server cluster can further ensure that the system remains responsive under high user loads.

The architecture of the proposed system is inherently adaptable to various environments and applications. The system can seamlessly integrate into diverse operational settings, from enterprise networks to IoT ecosystems, by leveraging standard cryptographic hash functions and ubiquitous QR code technology. The mobile application component, which is platform-independent, allows the solution to function in environments with or without consistent internet connectivity. Moreover, the security parameters, such as the dimensions and bounds of the hypercube used in the multidimensional hash chain, can be tuned to meet different security requirements. This flexibility makes the system suitable for specific application needs.

3.5 Conclusion and Limitations

The solution discussed in this section proposes a two-factor (knowledge-based and possession-based) authentication system where a smartphone generates OTP from a multi-dimensional infinite-length hash chain. The complex challenge is defined as an arbitrary path from the origin to a randomly selected point within a dynamically shifting multi-dimensional hypercube. To avoid manually typing long challenges, it is encoded as Quick Response (QR) codes that can be read using a mobile device's camera. Security analysis and threat modeling show that the proposed system effectively protects against vulnerable attacks that may occur at the time of authentication. Performance testing and analysis of the algorithm show that it could be used to carry out authentication using devices with limited resources.

While the proposed authentication scheme is an effective solution, certain limitations exist. The current performance analysis is based on an Intel Core i3-based system. Evaluating the system across a broader range of devices, including high-load server environments, is necessary to ensure consistent performance and responsiveness. While the security analysis addresses common threats, emerging attack vectors (e.g., those exploiting quantum computing or side-channel vulnerabilities) have not been comprehensively assessed. A more formal and extensive security verification process would be beneficial to establish the system's robustness fully. To fully understand user needs and concerns, further research is necessary, including user studies and surveys that assess perceptions of the system's convenience, security, and overall usability.

Chapter

4

Secure Authentication using One Time Contextual QR Code

4.1 Overview

This chapter will demonstrate a novel authentication solution proposed to counter various security breaches during the authentication process. A one-time contextual QR code is generated during each authentication request received from the system users. Contextual information present in the QR code is a random ID, which is unique for a particular login. The information present in the QR code is encrypted. During authentication, a device capable of scanning a QR code, such as a usual smartphone, with custom software developed for the proposed scheme is required. The custom software will decode and subsequently decrypt the contents of the QR code. After decryption, a piece of information will be shown to the user, using which the user needs to derive the OTP corresponding to their actual password.

4.2 Proposed Solution

This section provides a detailed explanation of the underlying working principle of the proposed solution. For a better understanding, we divide the solution into three phases: registration, device setup, and authentication.

4.2.1 Registration Phase

The first-time user should register to set up the authentication credentials. The registration process is similar to most current systems that use a username and password to authenticate. The hashing algorithm is the standard approach to storing passwords securely [66]. The hash values of the chosen username and password are stored in the database to maintain secrecy. It is assumed that registration is carried out in a secure environment and that the authentication credentials are not stolen during registration. Alternatively, the system administrator can register new users as per the organizational policy and share the authentication credentials with the users. The user can change the registration password at any time.

The following notations are used:
U_{plain} → plaintext username,
P_{plain} → plaintext password,
h() → one-way hash function,
U_h → hash value of username,
P_h → hash value of password
Thus, $U_h = h(U_{plain})$ and $P_h = h(P_{plain})$

The values U_h and P_h are stored against the respective user details in the database.

4.2.2 Device Setup

At the time of login, the user should have a device (such as a smartphone) capable of scanning a QR code. Custom software, which has been specifically developed for the proposed solution, needs to be installed on this device. The software installation on any device is a one-time activity. The custom software contains a configured cryptographic algorithm used to decrypt the information received from the authentication server.

4.2.3 Authentication Phase

During the authentication process, information exchange occurs between the authentication server and the user to verify the user's identity.

4.2.3.1 Authentication Server - Generate Login Information

The responsibility of the authentication server is to facilitate authentication for a user who attempts to access a secured system. On receiving a login request from the user, specific tasks are performed on the server end.

A mapping is generated between the alphanumeric characters and symbols (using which a password can be formed) and a pre-defined set of generic elements. In one of the proposed mappings, each cell of the QWERTY keyboard layout contains a mapping character. The alphabet cells of the QWERTY keyboard layout will contain a random alphabet mapping; the numerical cells of the QWERTY keyboard layout will contain a random numerical mapping; and the symbol cells of the QWERTY keyboard layout will contain a random symbol mapping. Thus, all 26 alphabets (a-z), numerals (0-9), and 14 chosen symbols of the QWERTY keyboard layout will contain a mapping element. The 14 chosen symbols are presented in Table 4.1.

Table 4.1: Symbols

Symbol	Name	Symbol	Name
!	Exclamation	*	Asterisk
@	At sign	(	Open Parenthesis
#	Hash	)	Close Parenthesis
$	Dollar sign	-	Minus
%	Percentage	+	Plus
^	Caret	_	Underscore
&	Ampersand	=	Equal

An example of the generated mapping is shown in Figure 4.1. Here, the elements in black are the usual keyboard elements of the QWERTY keyboard layout. The elements in red are the mapping elements against each usual element of the QWERTY keyboard layout.

Figure 4.1: Randomized QWERTY keyboard layout mapping

The authentication server also generates a unique random id for the particular authentication request and makes an entry of it into a database with the status as active. Also, the current timestamp in UTC (Coordinated Universal Time) is stored against the random id.

The concatenated value of generated mapping information and a random id is encrypted using the private key of the authentication server. This encrypted data is then encoded into a QR code. This QR code is sent to the end user who requested a login.

The following notations are used:

M → mapping information,

R → random id,

K_{PR} → private key,

K_{PU} → public key,

A → public-key encryption algorithm,

$E_A()$ → encrypted value using A,

$D_A()$ → decrypted value using A,

E_{QR} → encoded information in QR code,

D_{QR} → decoded information from QR code

Thus,

$$E_{QR} = Encode(E_A(K_{PR}, (M||R)))$$

This information is sent to the end user as a QR code.

4.2.3.2 User - Derive OTP

In case of authentication for applications on devices other than mobile phones, such as desktops or laptops, a user scans the QR code using custom software installed on a QR code scanning device. In the case of applications accessed on mobile phones, there would be an additional link beside the QR code, and when the link is clicked, the custom software will launch. Here, the QR code data will be passed internally to custom software.
The custom software first decodes the QR code.

$$D_{QR} = \text{Decode}(E_{QR})$$

$$D_{QR} = E_A(K_{PR}, (M||R))$$

The custom software uses the pre-defined public-key algorithm and the stored public key to decrypt the decoded information to obtain the mapping and the random id.

$$D_A(K_{PU}, (D_{QR})) = (M||R)$$

Custom software has the function of separating M and R from the concatenated values of M and R. The user enters the username chosen at the time of registration. The OTP to be entered needs to be derived from the mapping information presented to the user. In the case of mapping information shown in Figure 4.1, the user needs to find the corresponding mapping character for each element of the user's actual password. The sequence of the mapped characters is the derived OTP. In case of an uppercase letter present in the actual password, the corresponding mapped letter will be in uppercase. Similarly, for lowercase letters present in the actual password, the corresponding mapped letter will be in lowercase.

Example: Suppose the password of a user is "Data@1". The OTP corresponding to "Data@1" from Figure 4.1 can be derived as follows – For "D" the mapping character is "J", for "a" the mapping character is "l", for "t" the mapping character is "y", for "a" the mapping character is "l", for "@" the mapping character is "(" and for "1" the mapping character is "0". Thus, the derived OTP as per the mapping is "Jlyl(0".

On submitting the authentication information, the username (U_{plain}), OTP, and QR code are posted to the authentication server.

4.2.3.3 Authentication Server - Verify OTP

Upon receiving the authentication request, the authentication server first checks whether the user account is locked, and if it is found locked, the authentication request is denied. Suppose a user attempts to log in using invalid information for a pre-determined number of consecutive times. In that case, the account will be locked, and additional verification needs to be done to unlock the account.

The received QR code is decoded and then decrypted using the public-key algorithm. Mapping information and the random id are obtained from the decrypted data. This process is the same as that carried out at the custom software end to obtain mapping information and the random id from the QR code. The authentication server checks whether the random id (obtained from the QR code) has status as active in the database. If it is active, the status is changed to inactive, or else the authentication request is rejected. The timestamp stored in the database against the random id is obtained. The difference between this timestamp and the current time (in UTC) is calculated. If it exceeds the set time limit, then the authentication request is rejected.

The actual password is derived from the user-entered OTP and the mapping information extracted from the QR code. The hash values of the username and derived password are matched against the values (from the registration phase) in the server database. If the values match, the authentication is successful; otherwise, the authentication request is rejected.

4.3 Security Analysis, Applicability and Discussion

4.3.1 Security Analysis

Firstly, we compare the proposed scheme with SMS-based OTP, which is widely used in many secure applications. Recently, there have been compromises on SMS OTP sent to mobile phones [21]. Using suitable tools, the mobile traffic of an end user, including SMS, could be intercepted by an attacker. This would reveal the OTP (via SMS) at the interception point. Also, mobile phone malware capable of receiving SMS containing OTPs is becoming a rising threat. Upon reception, in the background, the SMS could be forwarded to an attacker and deleted from the mobile phone to hide the fact that the SMS containing OTP ever arrived at the infected phone. Our proposed authentication solution

does not send the OTP over a mobile network, so it is resistant to capturing mobile traffic. Unlike the SMS-based OTP system, the one-time password in the proposed scheme is not distributed as plaintext. Mapping information in an encrypted QR code is sent to the user, using which the user derives the OTP from the actual password known only to the user and the authentication server. Thus, malware-infected mobile devices used for scanning a QR code cannot reveal the authentication credentials.

Phishing attacks aim to steal sensitive information by impersonating a trusted entity through electronic means. Typically, users are tricked into entering details on a fraudulent website that has been developed identically to a legitimate one. The QR code used in the proposed solution can only be generated at the authentication server end, which has the private key of the public-key encryption algorithm. The custom scanning software will not resolve any other form of QR code; hence, mapping information will not be shown to the user. Also, there is an upper bound on the time the user should be authenticated after QR code generation. So, a QR code captured by an attacker to deceive the user later will not be significant. Thus, with the proposed scheme, phishing attacks on authentication information will be challenging to achieve.

Using a one-time password rather than a static password has proved useful while carrying out authentication. It is an efficient method to increase system security. In the proposed method of authentication, the password entered by the user is valid only for a particular authentication request. In subsequent logins, the mapping information changes; hence, the password to be entered also changes. Using OTP makes a system secure against key-logger software running in the background to track the keys struck on a keyboard and an asterisk logger showing hidden passwords masked behind asterisks. A password-stealing Trojan could acquire the password cached in browser memory in a compromised system. However, this kind of attack can be prevented using OTP, as the user never enters the actual password.

In a replay or playback attack, a message that a legitimate user sent earlier is fraudulently repeated. In our implementation, the dynamically generated QR code contains a unique random id, and the database on the server is used to track whether the random id is active. Authentication is allowed only if the random id is active and marked as inactive after its usage in an authentication. So, the server will reject a replay of the already used

authentication information, thus making the authentication system secure against replay attacks.

Shoulder surfing involves the use of direct visual observation to obtain sensitive information, such as watching over a person's shoulder during data entry. This type of attack can also be executed from a distance using tools like binoculars or other visual augmentation devices to discreetly capture information. In the proposed solution, shoulder surfing on the keyboard by directly observing through eyes or video recording can just reveal a temporary password valid for the particular authentication. The mapping information displayed to the user is on the handheld device, which is close to the user's body. So it is nearly impossible to shoulder surf both on the handheld device's screen and the system's keyboard simultaneously. Thus, it is resilient against shoulder surfing.

Brute force and a dictionary attack on passwords use all possible combinations of passwords and an exhaustive list of dictionary words, respectively, to crack the password of a system. In our implementation, any combination of alphabets (a–z and A–Z), numbers (0–9), and 14 special symbols can form the password.

Total possible passwords of length L, $N = (76)^L$
For L = 3, $N = (76)^3 = 4.39 * 10^5$
For L = 4, $N = (76)^4 = 3.34 * 10^7$

Thus, the password space is vast. Also, the system locks out the user account after a few invalid attempts. Thus, obtaining the user's credentials is nearly impossible using brute force and dictionary attacks. This aligns with most of the existing username and password-based authentication systems.

In SQL injection, a malicious script is written as the user input, and it gets executed against the database to compromise the stored information. In the proposed scheme, the server processes the received data before executing the database query. The hash values of the username and the derived password from OTP are verified against the database. So, SQL injection in the username and password fields will not be significant.

In a pre-play attack, an attacker prepares for the attack in advance by predicting the next challenge to be used for authentication. In the proposed scheme, the generated QR code is

encrypted by the authentication server's private key. The custom software will not resolve any other form of QR code, thus making it infeasible for an attacker to generate the next challenge.

Thus, the above discussion illustrates the usefulness of the proposed solution in countering various cyberattacks during the authentication process.

4.3.2 Applicability

At the time of authentication, a device capable of scanning a QR code, such as a smartphone, is required. Smartphones are already in use worldwide by a vast number of mobile phone users, and the usage of smartphones shows an increasing trend in the near future. Additional devices like token generators, chip-based cards, along with readers, biometric devices, etc., are not required. Thus, the infrastructure cost and the lag time to set up an authentication system are reduced to a large extent. This increases the applicability of the proposed scheme.

4.3.3 Discussion

Table 4.2 presents a comparative analysis between the proposed system and existing OTP-based authentication mechanisms.

Mobile SMS-based OTP systems are commonly used to authenticate users, especially during transactions. However, it is prone to malicious software installed on a mobile device. Also, there is a mobile network dependency to receive the SMS. Lamport & S/Key [57, 78] and Bicakci and Baykal [52] schemes require system re-initialization after a certain number of authentications. They are prone to pre-play attacks as well. Lamport & S/Key, Bicakci and Baykal, and Eldefrawy [63] schemes perform a high computation at the client end, and additionally, there is an overhead to store secrets. RSA SecurID generates a token in a dedicated hardware device. Thus, no extra resources are required from the client to generate the OTP. However, in the case of RSA SecurID, the device procurement cost is high, and it needs to be replaced after a certain lifetime. There is a lag time when setting up the RSA SecurID system for a user, as a dedicated hardware device needs to be tagged and shipped to the user. The proposed system of password entry eliminates all the mentioned problems. It is highly usable and can be easily integrated with existing password entry systems.

Table 4.2: Comparison with other OTP-based authentication systems

Metric↓ System→	SMS-based OTP	Lamport and S/Key	Bicakci and Baykal	Eldefrawy et al.	RSA SecurID	Proposed scheme
Usability	Very high	Low	Low	Low	High	High
Initial setup time	Small	Medium	Medium	Medium	High	Small
Device cost	Low	Low	Low	Low	High	Low
Operational cost	Medium	Low	Low	Low	Low	Low
Number of authentications before re-initialization	Unlimited	Low	Medium	Unlimited	High	Unlimited
Resistance to pre-play attack	High	Low	Medium	Medium	High	Very high
Client computational requirements	Not applicable	High	High	High	Not applicable	Low
Additional overhead to store secret	No	Yes	Yes	Yes	No	No
Mobile network dependency	Yes	No	No	No	No	No
Prone to SMS capture malware installed on mobile device	Yes	No	No	No	No	No
Type of encryption algorithm	Hash function	Hash function	Public key	Hash function	Hash function	Public key

4.4 Usability Study

A usability study has been conducted to determine the error rate and the time taken by the users to authenticate using the proposed method. The study was performed on 50 participants who are regular users of computers and mobile phones. The participants were given a presentation describing the new authentication method in detail. Also, the participants had hands-on experience with the new method of authentication. All the participants were given the same passwords for the experiment. The passwords varied in length from 3 to 8 elements, and each of the passwords had a minimum of one number or symbol along with the alphabet (a–z, A–Z).

The average value of the entry time of OTP after the appearance of mapping information on the user's handheld device is shown in Figure 4.2.

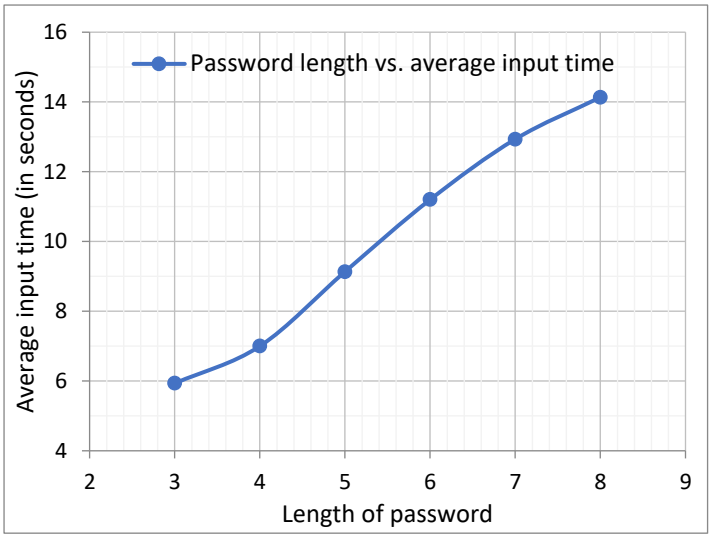


Figure 4.2: Password length vs average input time

It can be seen from the graph that, on average, a password of length three characters was derived and entered in 6 s, and a password of length eight characters was derived and entered in 14 s. We believe that the entry time taken by the users will reduce to some extent once they use the new authentication method a considerable number of times.

We found that invalid OTPs were entered in 5% of the total authentication attempts of all the users. This result may be because the proposed authentication method is still new to users. We believe the authentication failure percentage will decrease as users become comfortable using the new authentication method.

4.5 Conclusion

In this chapter, we have proposed an innovative solution for generating OTPs using mapping information and context-aware QR codes. The discussion in the earlier section establishes that it can counter various kinds of security attacks on a system. In the proposed solution, there is a trade-off between the time taken to authenticate a user and the system's security, so depending on the need, it can be deployed in applications where more significant security is desired. However, the new method is easy to integrate and adapt to existing systems.

Chapter

5

ATM transactions made safer using NFC and Blockchain

5.1 Overview

The proposed work in this chapter demonstrates a novel authentication solution to authenticate at ATMs. The authentication system does not rely on the use of physical ATM cards. During authentication, a user must have an NFC-supported smartphone with a custom application installed. The custom application uses mobile TPM to store some user-specific private information. A one-time password (OTP) must be generated in the custom application before reaching the ATM kiosk. When operating in card-emulation mode, an NFC-enabled device mimics the behavior of a contactless smart card. In this configuration, the radio frequency (RF) field is generated by the NFC reader, not the mobile device itself. At the ATM kiosk, the user needs to swipe the smartphone (in NFC card-emulation mode) in front of the NFC reader to authenticate. Blockchain has been used to store digital identity to carry out two-factor authentication of the system's users. The security strength of the proposed system has been discussed using threat modeling.

5.2 Proposed Solution

In this section, the registration and authentication processes are described in detail.

5.2.1 Registration Process

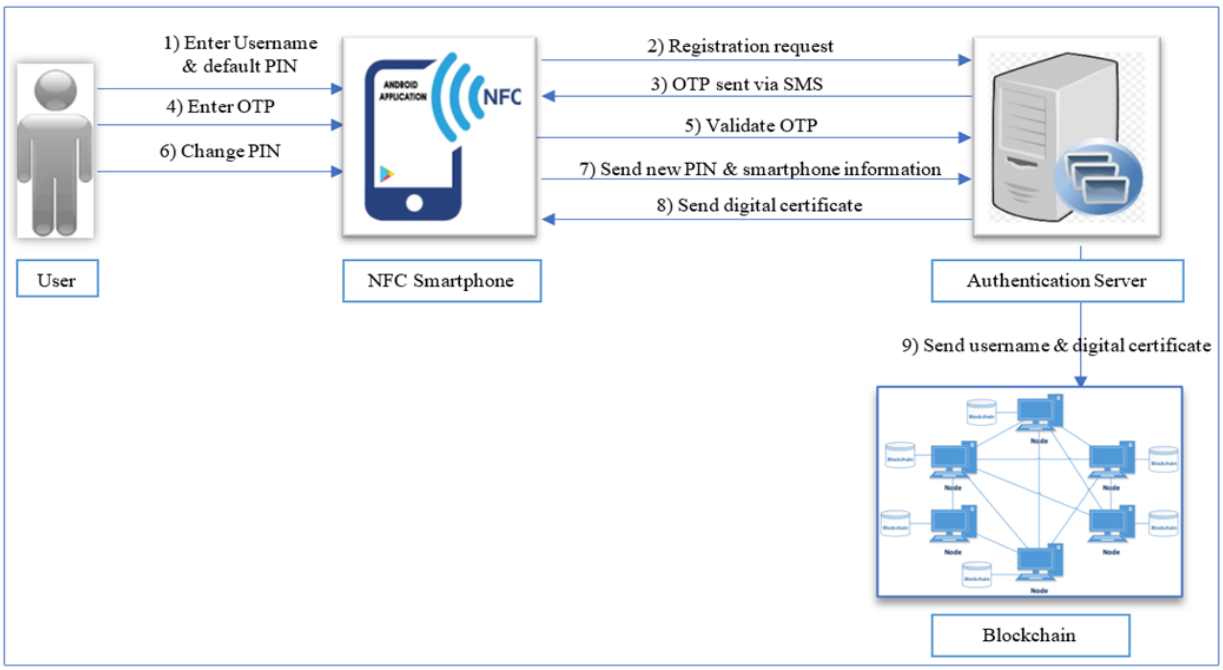


Figure 5.1: Registration process

The registration process is shown in Figure 5.1. In line with the current banking process deployed by some banks for ATM card registration, a username and default PIN are securely communicated to the end user of the banking system. A custom authentication application has been developed, which the user needs to install on their mobile phone. The custom application can be downloaded from a public repository like the Google Play Store or an internal banking website. A username and default PIN should be entered in the custom authentication application to register with the authentication server. At the time of registration, an OTP is delivered via SMS to the mobile number associated with the user's account to further validate the system's new user. The user needs to select a new PIN, i.e., change the default PIN during registration. If the authentication server accepts registration, the user's smartphone-specific unique information such as mobile device ID, manufacturing company (like Apple, Samsung, etc.), brand model (iPhone 12 Pro Max, Vivo X60 Pro, etc.), mobile operating system (Android, iOS etc.) and others are sent to the server and stored against the username of the user. The authentication server issues a digital certificate and transmits it to the user's smartphone via a secure channel. The issued digital certificate is securely stored by the custom application in the mobile TPM. Also, a copy of the same digital certificate is tagged to the user and stored in the blockchain. This completes the registration process.

5.2.2 Authentication Process

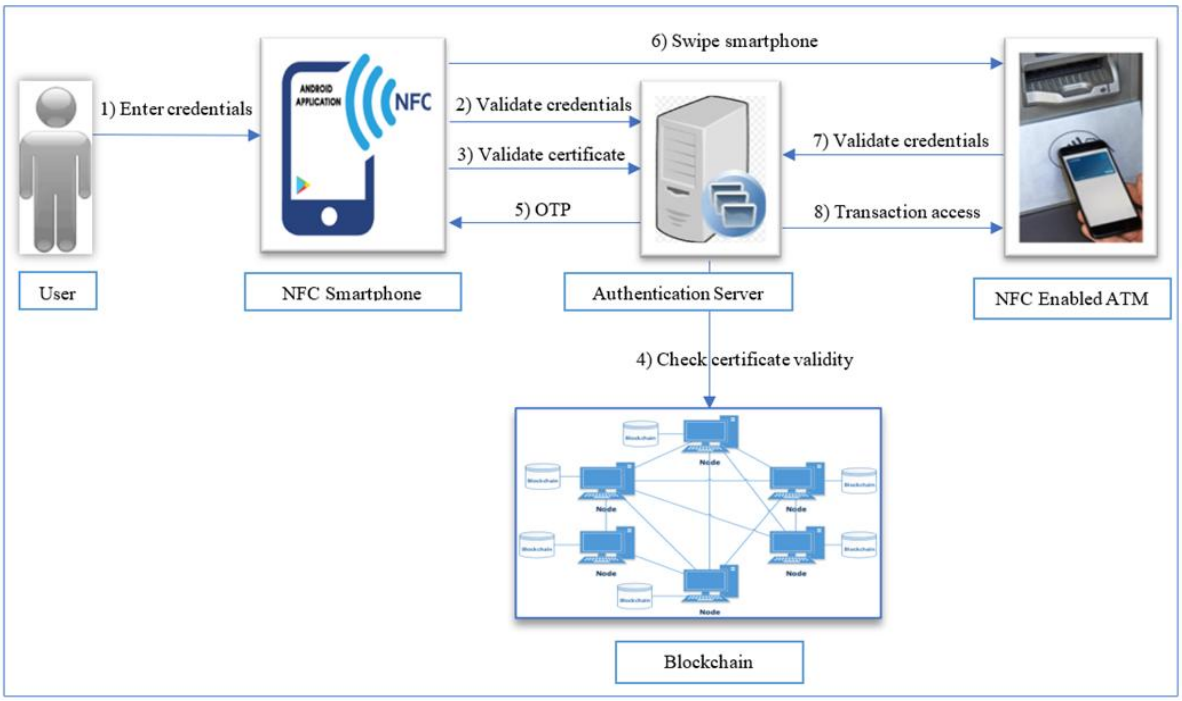


Figure 5.2: Authentication process

The authentication process is shown in Figure 5.2. A user must enter the username and the corresponding PIN (chosen during the registration process) in the custom authentication app registered on their smartphone. When a mobile device sends an authentication request, the authentication server receives both the user's credentials and device-specific information. On receiving the data, the authentication server validates it for correctness. Suppose the credential entered by the user is valid. In that case, the digital certificate (issued by the authentication server during registration) is sent via the custom authentication app to the authentication server for validation. The digital identity is verified in the blockchain network. After successfully validating the information, the authentication server sends an OTP to the user's device, and the custom authentication app securely stores the OTP. The validity of the OTP is for a preset time duration (for example, 15 minutes) from the generation time. The stored OTP is hidden from the user. This generated OTP can be used once during its validity duration to authenticate at the ATM kiosk.

Before reaching the ATM kiosk, the user can log in to the custom authentication app and start the NFC card emulation mode. The NFC card emulation mode will make the user's mobile phone act like an NFC card. An NFC reader can read the information exposed via the NFC by placing the mobile phone in close contact with an NFC reader. To authenticate

at the ATM kiosk, users must tap their smartphone against the NFC reader installed at the terminal. When the mobile phone comes in contact with the NFC reader, the authentication data from the custom authentication app is read by the NFC reader through a secure channel. The authentication server verifies the authenticity of the information and grants transaction access to the user.

5.3 Discussion

Most of the latest smartphones have built-in NFC support. In card emulation mode, the NFC-enabled device functions as a virtual contactless card. An external NFC reader can access the emulated NFC card. In the proposed system, a dedicated physical card or extra device (apart from an NFC-supported smartphone) is not required during authentication. This eliminates the overhead of manufacturing and shipping physical ATM cards.

A custom authentication application has been developed, which the user needs to install on their mobile phone. The custom application can be downloaded from a public repository like the Google Play Store or an internal banking website. Ideally, the app needs to be installed only once on a device. During registration, the user's smartphone-specific unique information such as mobile device ID, manufacturing company (like Apple, Samsung, etc.), brand model (iPhone 12 Pro Max, Vivo X60 Pro, etc.), mobile operating system (Android, iOS, etc.) and others are sent to the server and stored against the username of the user. The device-specific unique information is an additional piece of information (apart from the credentials) that the authentication server uses to validate the source requesting authentication.

At a later point, a user can change the NFC device (smartphone) as required. The installed app or the internal banking website can generate a new default PIN. After generating a new default PIN, the custom app automatically deregisters from the current device. The default PIN can now be used to register the app on the new device.

Blockchain technology raises a new paradigm, which puts the user at the centre of managing their own digital identity; this idea is behind one of the digital movements called self-sovereign identity (SSI). This new decentralised digital identity model gives the user the power to manage and share their identity. SSI revolves around the concept of Decentralised Identifiers (DIDs) and Verifiable Credentials (VCs), which is essentially the

heart of the SSI. We use physical credentials such as our identity cards, like a passport or license; the idea behind the Verifiable Credential is to create a digital credential like the physical one. The DID is a unique identifier that associates a person with a VC to identify the individual uniquely. In a digital world, a certificate associated with a user can be seen as a Verifiable Credential. Managing the key associated with the user is one of the critical areas of security. In the blockchain world, we can use a Decentralised Key Management System (DKMS) to provide highly scalable key distribution, verification, and recovery.

In our proposed system, a digital certificate corresponding to each user is generated at the time of registration. This digital certificate is securely transmitted to the registered user and acts as the SSI for the user. This certificate is stored securely on the user's mobile device. Specifically, the mobile TPM, an added layer of protection to keep sensitive user data, has been used in our system. Also, a copy of the certificate corresponding to the user is securely recorded on the blockchain. During the authentication process, the user's digital certificate is validated before providing them access to the ATM network. The validation of the digital certificate, apart from the credentials, makes it a second factor for authentication. Authenticating the genuineness of the user from an independent blockchain system further strengthens the system's security.

In ATM networks, a well-defined set of users uses the system. The financial transactions are usually carried out within an organization's systems or among its trusted partners. Thus, private blockchain is a better choice over public blockchain networks. In contrast to a public blockchain, a private blockchain is a restricted network that restricts access and participation to authorized individuals within an organization. It offers many advantages, such as improved security measures, increased scalability, compliance with regulatory standards, comparatively lower operational costs, and better intellectual property protection.

An NFC relay attack is a sophisticated exploit primarily targeting contactless payment systems by leveraging NFC technology. In this attack, criminals generally use two NFC-enabled devices, one to intercept and another to amplify the signals between the victim's NFC device and, particularly, a payment terminal. This manipulation of the transaction process highlights the serious nature of the threat. In our proposed work, as discussed earlier, mobile TPM has been used to store user OTP and digital certificates. The use of TPM is one of the mitigation strategies against NFC relay attacks, as confidential

information is not exposed openly. The user activates the NFC card emulation mode only at ATM kiosks, and the exchange of messages between a smartphone and the ATM terminal takes place via a secure, encrypted channel.

In the proposed system, a smartphone configured to emulate an NFC card has proven advantageous over physical ATM card usage. If a criminal obtains an ATM card, then it is easy to read the information written on it or skim the ATM card. A smartphone can be protected by various authentication methods, such as passwords, biometrics, etc., making it difficult for criminals to break into the smartphone. Manufacturing many cards and transporting them to the end users are both time-consuming and involve some cost. In the case of smartphones, the authentication setup is much quicker as it is software-based. Mobile phones have become an indispensable part of daily life, so a stolen mobile phone will be easily noticed, whereas a user might not see a stolen ATM card for many days.

5.3.1 Threat Modeling

This section determines the threat model, including the security objectives, trust boundaries, and possible threats.

5.3.1.1 Security Objectives

A secure system must fulfil numerous security requirements to ensure stability and resiliency. Identification represents a primary requirement in any confidential system. It is the act of finding out the identity of a person or application requesting access to a secure system. Any anonymous entity should be denied access to a secure system. Nonrepudiation is another essential security requirement. It ensures that an entity cannot repudiate the execution of a specific action. The transactions carried out by an authorised user should be accountable.

5.3.1.2 Trust Boundary

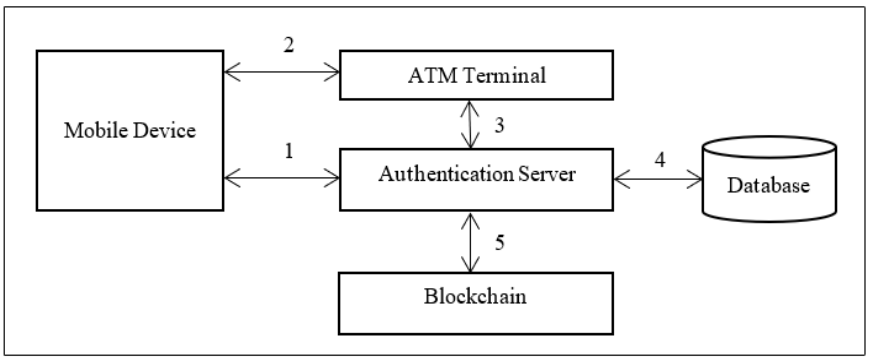


Figure 5.3: Trust boundary and the communication channels

The proposed authentication scheme has five major components (mobile device, ATM terminal, authentication server, blockchain, and database) and five major communication channels (marked by numbers 1 to 5) for transferring data, as shown in Figure 5.3.

The user's smartphone details are captured and stored in the authentication server during registration. We assume that the mobile device component is uncompromised as long as it is in the user's possession. It is assumed that the ATM terminal, authentication server, and database are secure components, as securing these components is taken care of by the existing ATM systems. Blockchain has emerged as a secure technology that safeguards data against unauthorized access, minimizes the risk of fraud, and reduces the likelihood of data breaches or tampering. It is increasingly used in different sectors to avoid fraud and to increase data protection. In our proposed solution, we have used a private blockchain. The financial entity running a private blockchain can easily change the rules of a blockchain, nullify transactions, and perform other desired functions. The validators are known in a private blockchain, so any risk of a 51% attack is eliminated. A private blockchain enhances privacy by limiting read access to authorized participants only.

TLS and SSL are widely used cryptographic protocols that ensure secure communication across computer networks. In the proposed system, communication between a mobile device and the authentication server occurs via the Internet and is secured using TLS or SSL. The security of the communication channel between an ATM terminal and the authentication server, and the channel between the authentication server and the database, is assumed to be taken care of by the existing ATM systems, and it is not in the scope of our discussion. A mobile device communicates with an ATM terminal via NFC technology.

The authentication server establishes a secure channel via TLS or SSL to communicate with the blockchain system.

5.3.1.3 Threats

In today's cybersecurity landscape, there are numerous streams of potential threats. The threats look for vulnerabilities in the environment that they can exploit. In system and network security, threats can be mitigated by properly using security features and procedures. The threats could be on the components, communication channels, and the authentication protocol of the system.

Attacks on the components:

The proposed authentication scheme has five major components: mobile device, ATM terminal, authentication server, blockchain, and database. We assume that the mobile device component is uncompromised as long as it is in the user's possession. However, a mobile device such as a smartphone can be stolen in an adverse scenario. Smartphones have become essential to daily life, so smartphone theft will quickly get noticed. The custom authentication app can be easily unregistered from the stolen device via online or phone banking. A new default PIN needs to be generated to register a fresh app on the new mobile device. Also, smartphone data can be wiped off remotely using applications with elevated privileges. As mentioned earlier, the ATM terminal, authentication server, database, and blockchain network are assumed to be secure, and the security of these individual components is not in the scope of our discussion.

Attacks on the communication channels:

As depicted in Figure 5.3, there are five primary communication channels in the proposed system, marked by numbers 1 to 5. The data exchange between a mobile device and the authentication server and between the authentication server and the blockchain network is protected by TLS. Therefore, the packets cannot be modified or snooped during transmission if the certificate is valid. A mobile device communicates with an ATM terminal via an NFC channel. As stated earlier, the communication distance and the lifetime of an NFC channel are short, thus making it difficult for an attacker to compromise. As mentioned earlier, the communication channels between an ATM terminal and the

authentication server and between the authentication server and the database are assumed to be secure.

Attacks on the authentication protocol:

An ATM card and corresponding PIN commonly validate a user at an ATM kiosk. ATM skimming, along with the subsequent cloning of cards (including chip-based cards), has emerged as a security concern. Also, attackers exploit methods, such as shoulder surfing by direct observation, hidden cameras, or other vision-enhancing devices, to steal the PIN corresponding to the user's ATM card. In the system proposed in this chapter, NFC card emulation mode in a mobile device is considered an alternative to ATM cards during authentication. The mobile application generates an OTP, which is used to authenticate the user. The authenticity of the user and the device requesting the OTP are validated before generating the OTP. A replay attack involves a malicious actor capturing data from a secure network transmission and then replaying it as authentic information. In contrast to static passwords, since the OTP is valid only for one login session, it is resistant to replay attacks. The blockchain system validates the user's digital certificate to determine whether the source is genuine. The blockchain system will reject the digital certificate generated by any malicious source.

5.4 Conclusion

An ATM is a self-service electronic device that enables users to perform transactions without visiting a physical bank branch. It is a convenient method to meet users' transaction requirements, such as deposits, cash withdrawals, bill payments, etc. Millions of ATMs are deployed all around the globe. Therefore, ensuring user security and offering ease of use during ATM transactions is of utmost importance. Using debit cards or similar types of cards at ATMs can present issues, such as ATM skimming, damaged magnetic strips of cards, lag time to set up the system users, longer time to authenticate at the ATM kiosks, etc. In this chapter, we have proposed a novel solution to authenticate users at ATM kiosks. Here, a smartphone supporting the NFC feature (present in most modern smartphones) is required during authentication into the ATM system. The discussion of the advantages of using the proposed system shows that it is a valuable alternative to the existing authentication method. Blockchain technology has been leveraged to offload the server-side computation and act as a second factor authenticator to validate the digital certificate

of the system users. Security analysis and threat modeling show that the proposed system effectively protects against vulnerable attacks that may occur at the time of authentication.

Chapter

6

Contextual Authentication of Users and Devices using Machine Learning

6.1 Overview

In this chapter, apart from the usual authentication methods currently in practice, the context of users and devices is considered during authentication to detect anomalies and security-related attacks. An independent machine learning network processes and analyses the contextual information to determine the credibility of the source requesting authentication.

6.2 Proposed Solution

This section details the core concept behind the proposed solution. Additionally, the steps involved in processing context information during authentication are detailed.

6.2.1 Working Principle

Context analysis of the user/device will be carried out in the background during authentication. The context analysis will facilitate the construction of a historical data-based classifier/prediction model to derive the credibility score of the user/device during

authentication. A machine learning network will be built with a continuously evolving new dataset. The machine learning analysis could be carried out in a central on-cloud network where the technology stack can be easily updated without affecting end users.

The context of the user/device varies from one system to another. Hence, the contextual parameters can be set as per a system's requirements. Some of the contextual parameters that can be considered are:

1. Source network address reputation - Network address/Internet Protocol (IP) reputation intelligence enhances online security by identifying and blocking sources associated with malware, as well as potentially harmful or suspicious content. Organizations can restrict authentication attempts originating from an IP address or a subnet associated with known bad actors.
2. Device Software - Software/programs in the end user machine, such as old security patches of the operating system, absence of anti-malware software, presence of malicious software like keyloggers, etc., may indicate a compromised system.
3. Location information - Context-aware user verification can track a user's physical location to analyse the possibility of an authentication request from that location. Combining the physical location of login attempts with a user's historical access patterns can enhance the detection of unauthorized activities. For example, if a user authenticates at 9 a.m. from Canada and again tries to authenticate at 10 a.m. from Dubai on the same day, that would be an improbable travel event and raise suspicion.
4. Behavioral analysis - A user's interaction patterns with a device, such as keystroke dynamics, mouse movements, and similar behaviors, can be recorded and analyzed over time to derive the usage patterns of the particular user. During later uses, deviations outside the established behavioral patterns can raise suspicion.
5. Network information to detect network anomalies.

6.2.2 User/Device Registration

There are various existing methods of authentication to authenticate a user/device, such as knowledge-based (e.g., personal identification number, password), possession-based (e.g., RSA token, automated teller machine cards), inheritance-based (e.g., biometrics like a fingerprint, face recognition), and somebody you know (social networks). At the time of registration, the secret information corresponding to the chosen authentication scheme is

stored at the server end to validate a client's identity requesting access to resources. The proposed method of context-based authentication can be adapted to the various existing authentication schemes. Context-based processing will be carried out in the background, and it can be hidden from users (or potential attackers) of a secured system.

6.2.3 Authentication Process

The authentication process depicted in Figure 6.1 caters to both user authentication and device authentication.

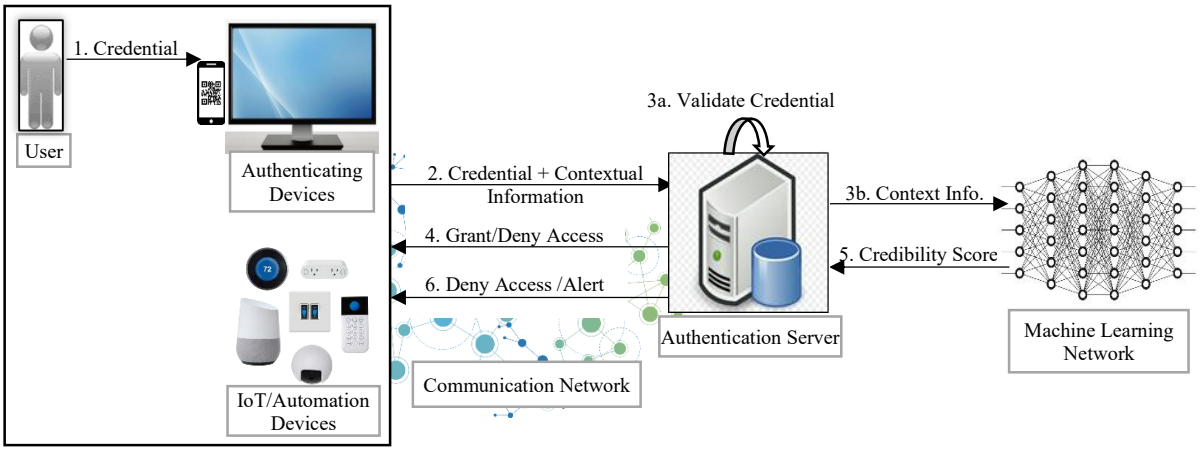


Figure 6.1: Authentication process

6.2.3.1 First Level of Authentication

The authentication process shown in Figure 6.1 uses credentials (like username and password) to authenticate the users. However, the core framework of the proposed system will be the same for other types of authentication techniques as well. In the case of user authentication, the contextual information of both the user and their device is captured and communicated to the authentication server along with the user credentials. When standalone devices are being authenticated, secret information corresponding to the device, such as the device's digital certificate and contextual knowledge, is communicated to the authentication server.

The authentication server verifies the user/device based on the received secret information corresponding to the user/device. The private data is validated against the information stored in the authentication server during registration. If user/device validation fails, the

authentication server flags the request as invalid. On the other hand, if the secret information is correct, the user is granted access to the system/resources.

6.2.3.2 Second Level of Authentication

In the background, the authentication server sends the contextual information obtained from the user/device to an independent machine learning network. This network is an isolated entity that performs contextual processing on the received data. The machine learning network can reside in a central on-cloud network where the technology stack can be easily updated without affecting end users. This data offloading strategy facilitates the analysis of large volumes of data on powerful computational resources of a secure ML network to improve detection performance. Various contextual parameters, such as source network address reputation, device software information, location information, the behavioral pattern of the users, network information, etc., can be considered depending on the system's needs. The result of the contextual analysis is a credibility score of the user/device, which helps to determine the level of authenticity of the entity requesting authentication. Based on the credibility score, the authentication server can invalidate an authentication request or alert the user about a suspicious authentication attempt.

6.2.4 Intrusion Detection System Modeling

An anomaly-based intrusion detection system has been modelled and analyzed in detail to detect security attacks based on the contextual information of users and devices. In particular, various machine learning models have been generated and tested to detect DoS/DDoS and brute-force attacks. The various steps involved in building and testing the ML models are shown in Figure 6.2.

6.2.4.1 Data Collection

In our research, the "Canadian Institute for Cybersecurity (CIC) Intrusion Detection Systems (IDS)" dataset, which is CIC-IDS2017 [41], is used for the analysis of DoS/DDoS attacks and brute force attacks. This dataset contains benign and common attacks that resemble actual real-world data. The dataset was generated to build realistic background traffic. It uses the B-Profile system [41] to model abstract patterns of human interaction and generates naturalistic, benign background traffic. The dataset contains different data

files representing each attack: DoS/DDoS and brute-force. In a data file, the rows represent the data points, and the columns represent the features.

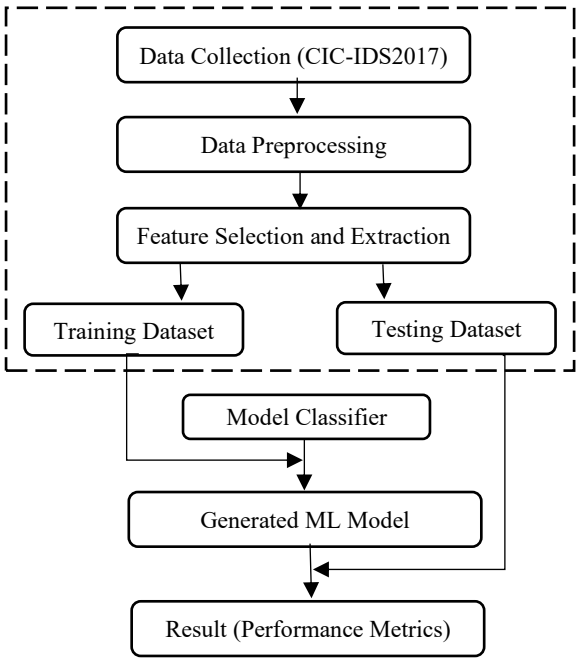


Figure 6.2: Steps in IDS modeling

6.2.4.2 Data Preprocessing

The second step in building the model is data preprocessing. In this step, the data quality is ensured by using techniques like managing missing values, addressing outliers, transforming the data into an appropriate format, etc. In the data files of DoS/DDoS and brute-force attacks, 79 features characterize each of the data points. Out of the 79 features, it has been observed that 'Fwd Header Length' is a duplicate. This duplicate feature has been removed from the dataset, and analysis has been carried out using 78 features.

6.2.4.3 Feature Selection

In the feature selection step, a subset of the attributes is chosen from the dataset and considered for training and testing the ML model. The CIC-IDS2017 dataset consists of 78 features, and an exhaustive analysis is carried out using all the dataset's features. Apart from complete dataset analysis, a feature selection technique has been applied to rank the features and consider the selective top-ranking features for the study. The feature extraction technique transforms the data from a higher to a lower-dimensional space. Fewer features require less computation time, and model accuracy may also improve due to less misleading

data [73, 99]. Various feature ranking methods are available, and in our research, the InfoGainAttributeEval ranking method has been used to rank the dataset's features. To make feature selection, the InfoGainAttributeEval algorithm measures how each feature contributes to decreasing the overall entropy.

6.2.4.4 Model Classifier

A classifier is the ML algorithm used to generate ML models. Some of the popular machine learning classification algorithms like J48, Random Forest, Multi-Layer Perceptron (MLP), and Bayes Net have been studied in our research.

6.2.4.5 Model Training

Model training involves using the selected ML algorithm and the training dataset to learn the patterns in the data. The outcome of this step is a generated ML model. In one experimental case, the data files of the CIC-IDS2017 dataset for DDoS/DoS attacks and brute-force attacks are split into 50% data to train the model and the remaining 50% for testing the model. In another experimental case, the same data files are divided into 40% data to train the model and the remaining 60% for testing the model.

6.2.4.6 Model Evaluation

During model evaluation, the performance of the trained model is assessed using the test dataset. The machine learning classifiers are evaluated using the confusion matrix, which summarizes the number of incorrect and correct predictions made by a classification model as a table. [107] (Table 6.1).

Table 6.1: Confusion Matrix

		Predicted	
		Positive	***Negative***
True	***Positive***	TP	FN
	Negative	FP	TN

Notations used -

TP = True positive (positive labeled record classified as a positive record)

TN = True Negative (negative labeled record classified as a negative record)

FP = False Positive (positive labeled record classified as a negative record)
FN = False Negative (negative labeled record classified as a positive record)
Accuracy, Precision, Recall, and F-Score are studied collectively to determine the best-fit classifier for a particular security attack.

- Accuracy reflects the rate at which attack instances are correctly classified.

$$Accuracy\ (A) = \frac{TN + TP}{TN + FN + FP + TP} \qquad (Eq. 6-1)$$

- Precision represents the proportion of true positive predictions among all positive predictions made by the model, including both true positives and false positives.

$$Precision\ (P) = \frac{TP}{TP + FP} \qquad (Eq. 6-2)$$

- Recall measures the proportion of true positive attack predictions out of all actual attack cases in the dataset.

$$Recall\ (R) = \frac{TP}{TP + FN} \qquad (Eq. 6-3)$$

- The harmonic mean of a system's recall and precision values is represented by the F-score.

$$F - score\ (F) = \frac{2\ \times\ (R \times P)}{(R + P)} \qquad (Eq. 6-4)$$

6.3 Experiments and Results

In this section, the various experiments conducted to detect DoS/DDoS and brute-force attacks are described in detail. The ML models are generated with the help of the Weka tool. The different performance metrics, including accuracy, precision, recall, f-score, and model-built time, are analyzed for the generated ML models.

System configuration: All the experiments have been carried out on the same laptop so that performance time can be compared across the experiments. System configuration: Processor - 12th Gen Intel® Core™ i5-1235U @ 1.30 GHz, RAM - 16.0 GB, System type - 64-bit Windows operating system, x64-based processor

Assumptions: The data is assumed to be available as one flat file or relation, where a fixed number of attributes describe each data point. It is further assumed that the training dataset represents a representative population sample so that the generated model is not biased. The CIC-IDS2017 dataset used in this research contains benign and different kinds of modern DoS/DDoS and brute-force attacks, which are assumed to resemble real-world data.

6.3.1 DoS/DDoS Attack

This type of cyber-attack aims to disrupt the normal functioning of a host system, rendering a machine or network resource inaccessible to its legitimate users. It is usually accomplished by flooding the target system with excessive traffic, thereby exhausting its resources and obstructing the processing of legitimate requests.

6.3.1.1 Dataset

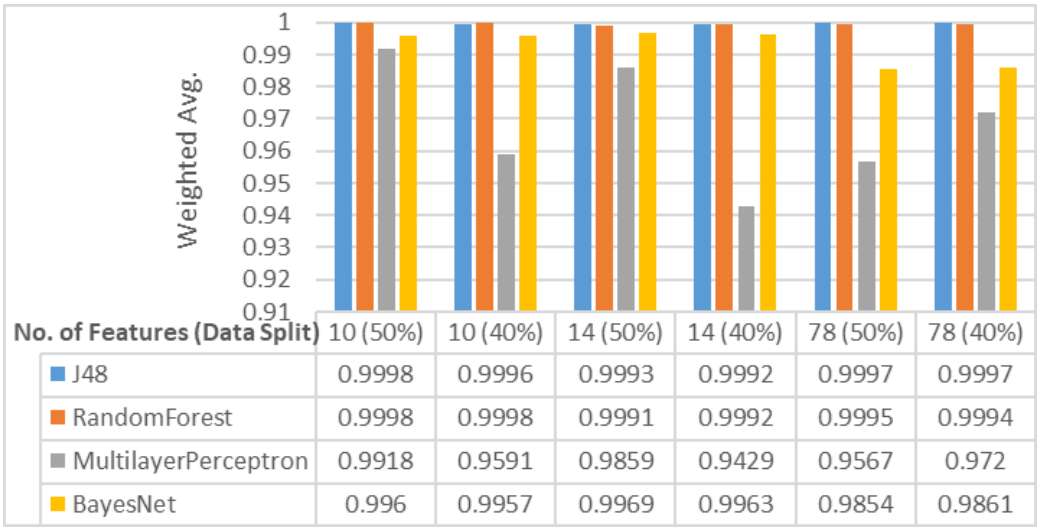
In the CIC-IDS2017 dataset, the data file representing the traffic of benign and DoS/DDoS attacks has been used for analysis. The dataset comprises 78 features and 6,92,703 records labeled Benign, DoS slowloris, DoS Slowhttptest, DoS Hulk, DoS GoldenEye, and Heartbleed. In this chapter, for analysis, the various categories of DoS attacks have been represented collectively as DOS attacks. Thus, the transformed dataset has two types of labels - BENIGN and DOS. Out of the total 6,92,703 records, there are 4,40,031 instances of BENIGN and 2,52,672 instances of DOS.

6.3.1.2 Analysis

Analysis of the DoS/DDoS attacks has been carried out on the popular machine learning algorithms - J48, Random Forest, Multi-Layer Perceptron (MLP), and Bayes Net using the Weka tool. A part of the dataset is used to train the classifier and generate a model, while the remaining data is used to evaluate the model's performance. The output produced by the Weka tool is a confusion matrix (along with other details) for each unique combination of the testing dataset and the classifier. The DoS/DDoS attack-based data file of the CIC-IDS2017 dataset is used for a 50% and 40% data split, i.e., 50% and 40% instances, respectively, to train the model, and the remaining for testing the model.

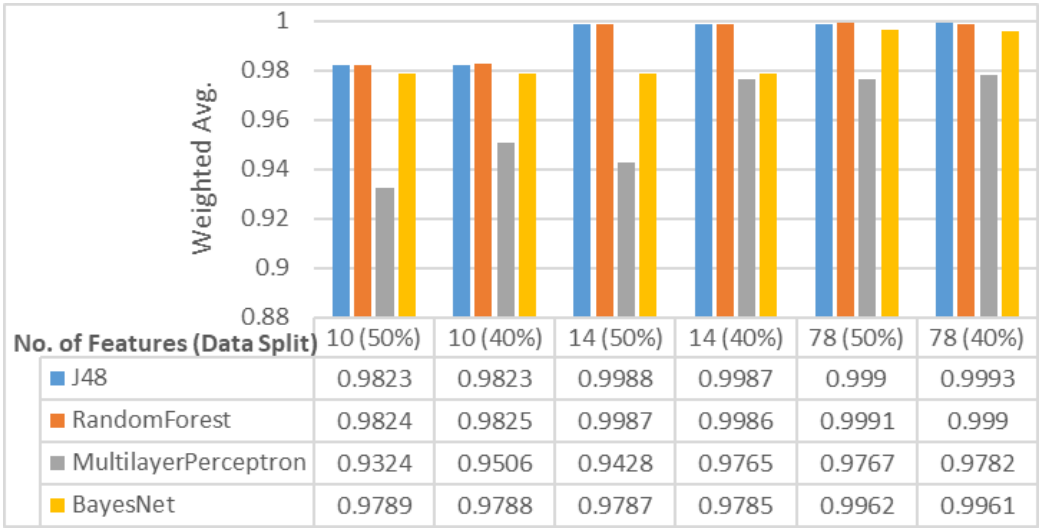
The dataset consists of 78 features. An exhaustive analysis of the DoS/DDoS attacks is carried out on the complete dataset with 78 features. To study the most significant features out of the 78 features, the InfoGainAttributeEval ranking method has been used to rank the 78 features. An analysis of the DoS/DDoS attacks is carried out on the dataset of the top 14 and top 10 ranked features. A comparative study of the dataset having 78 features, 14 features, and 10 features is done for various metrics, including performance metrics.

The accuracy, precision, recall, and f-score are computed using equations 6-1, 6-2, 6-3, and 6-4 on the generated confusion matrix for J48, Random Forest, MLP, and Bayes Net for 50% and 40% data split and different feature groups. Figures 6.3, 6.4, 6.5, and 6.6 present graphical representations and corresponding data tables summarizing the computed values of recall, precision, accuracy, and f-score, respectively.



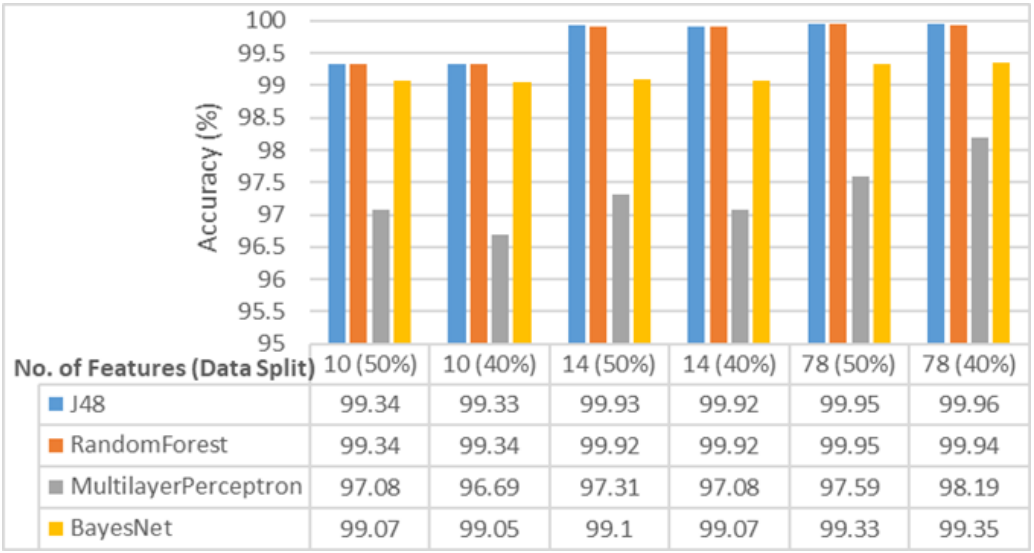
No. of Features (Data Split)	10 (50%)	10 (40%)	14 (50%)	14 (40%)	78 (50%)	78 (40%)
J48	0.9998	0.9996	0.9993	0.9992	0.9997	0.9997
RandomForest	0.9998	0.9998	0.9991	0.9992	0.9995	0.9994
MultilayerPerceptron	0.9918	0.9591	0.9859	0.9429	0.9567	0.972
BayesNet	0.996	0.9957	0.9969	0.9963	0.9854	0.9861

Figure 6.3: Recall for DoS/DDoS



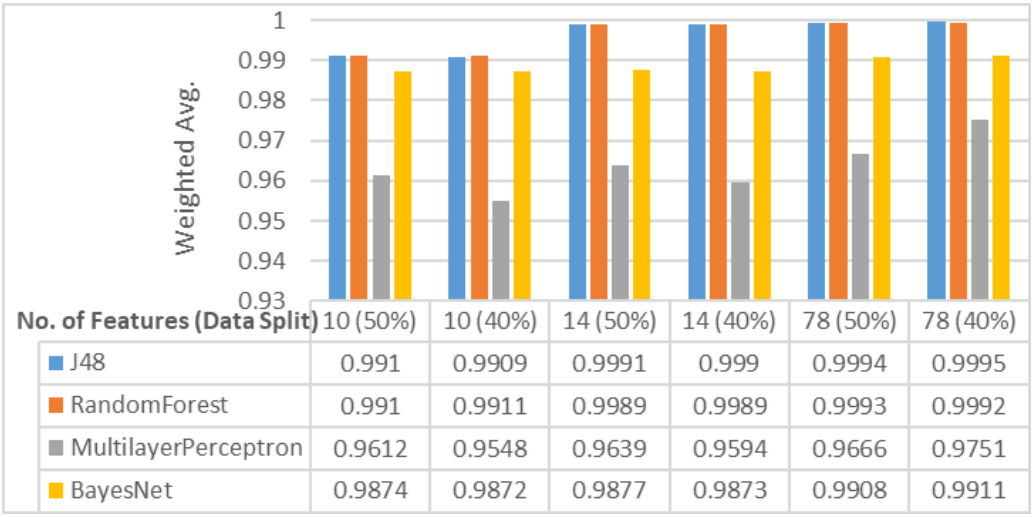
No. of Features (Data Split)	10 (50%)	10 (40%)	14 (50%)	14 (40%)	78 (50%)	78 (40%)
J48	0.9823	0.9823	0.9988	0.9987	0.999	0.9993
RandomForest	0.9824	0.9825	0.9987	0.9986	0.9991	0.999
MultilayerPerceptron	0.9324	0.9506	0.9428	0.9765	0.9767	0.9782
BayesNet	0.9789	0.9788	0.9787	0.9785	0.9962	0.9961

Figure 6.4: Precision for DoS/DDoS



No. of Features (Data Split)	10 (50%)	10 (40%)	14 (50%)	14 (40%)	78 (50%)	78 (40%)
J48	99.34	99.33	99.93	99.92	99.95	99.96
RandomForest	99.34	99.34	99.92	99.92	99.95	99.94
MultilayerPerceptron	97.08	96.69	97.31	97.08	97.59	98.19
BayesNet	99.07	99.05	99.1	99.07	99.33	99.35

Figure 6.5: Accuracy for DoS/DDoS



No. of Features (Data Split)	10 (50%)	10 (40%)	14 (50%)	14 (40%)	78 (50%)	78 (40%)
J48	0.991	0.9909	0.9991	0.999	0.9994	0.9995
RandomForest	0.991	0.9911	0.9989	0.9989	0.9993	0.9992
MultilayerPerceptron	0.9612	0.9548	0.9639	0.9594	0.9666	0.9751
BayesNet	0.9874	0.9872	0.9877	0.9873	0.9908	0.9911

Figure 6.6: F-Score for DoS/DDoS

The model-built time (in seconds) for the combinations of features and ML algorithms for 50% and 40% of the data split is shown in Table 6.2.

Table 6.2: Model-built time (seconds) for DoS/DDoS

No. of Features (Data Split)	10 (50%)	10 (40%)	14 (50%)	14 (40%)	78 (50%)	78 (40%)
J48	11.9	14.08	46.44	42.4	320.62	309.52
RandomForest	137.41	137.17	169.31	173.25	847.28	807.71
MultilayerPerceptron	289.97	285.31	687.76	465.97	25381.67	17675.83
BayesNet	5.18	4.37	9.12	8.49	56.18	36.22

6.3.1.3 Inference

It is observed from Figures 6.3, 6.4, 6.5, and 6.6 that for all the combinations of features and data split, the values of recall, precision, accuracy, and f-score are higher for J48 and Random Forest algorithms than MLP and Bayes Net algorithms. Between MLP and Bayes Net, it is observed that Bayes Net performs better than MLP in terms of recall, precision, accuracy, and f-score metrics. It can be seen that with both 14 and 78 features, the J48 and Random Forest algorithms achieve greater than 99% values of recall, precision, accuracy, and f-score. As depicted in Table 6.2, all the algorithms take much less time with 10 and 14 features as compared to 78 features. Thus, considering various metrics, including performance metrics, 14 features instead of all 78 features, is good to consider when building the machine learning model to detect DoS/DDoS attacks. It can be seen from Table 6.2 that the J48 algorithm performs slightly better than the Random Forest algorithm for both 50% and 40% of the data split. Thus, in the proposed authentication solution, the ML algorithm model created using the J48 algorithm and considering 14 features is chosen to detect DoS/DDoS attacks. The top 14 ranked features and entropy generated using the InfoGainAttributeEval ranking method are shown in Table 6.3.

Table 6.3: Top 14 ranked features for DoS/DDoS

Entropy	Rank	Attribute Name
0.761	1	"Init_Win_bytes_forward"
0.678	2	"Packet Length Mean"
0.676	3	"Average Packet Size"
0.672	4	"Bwd Packet Length Mean"
0.672	5	"Avg Bwd Segment Size"
0.668	6	"Destination Port"
0.666	7	"Max Packet Length"

Entropy	Rank	Attribute Name
0.663	8	"Total Length of Bwd Packets"
0.663	9	"Subflow Bwd Bytes"
0.641	10	"Bwd Packet Length Max"
0.618	11	"Flow IAT Max"
0.618	12	"Packet Length Variance"
0.618	13	"Packet Length Std"
0.605	14	"Fwd Packet Length Max"

The work in [13] and [48] demonstrates that “Packet Size” is one of the essential features in DoS/DDoS attacks. Furthermore, the work described in [60] and [80] experimentally shows that packet size distribution is useful in detecting DDoS attacks. In Table 6.3, 10 features (“Packet Length Mean”, “Average Packet Size”, “Bwd Packet Length Mean”, “Avg Bwd Segment Size”, “Max Packet Length”, “Total Length of Bwd Packets”, “Bwd Packet Length Max”, “Packet Length Variance”, “Packet Length Std” and “Fwd Packet Length Max”) out of the 14 features are directly proportional to the packet size. This further strengthens the outcome features in Table 6.3 as it is in sync with other research works.

6.3.2 Brute Force Attack

The brute force method relies on exhaustive attempts to determine valid login credentials through trial and error. It is usually done using automated tools and scripts, iterating through every possible combination of credentials.

6.3.2.1 Dataset

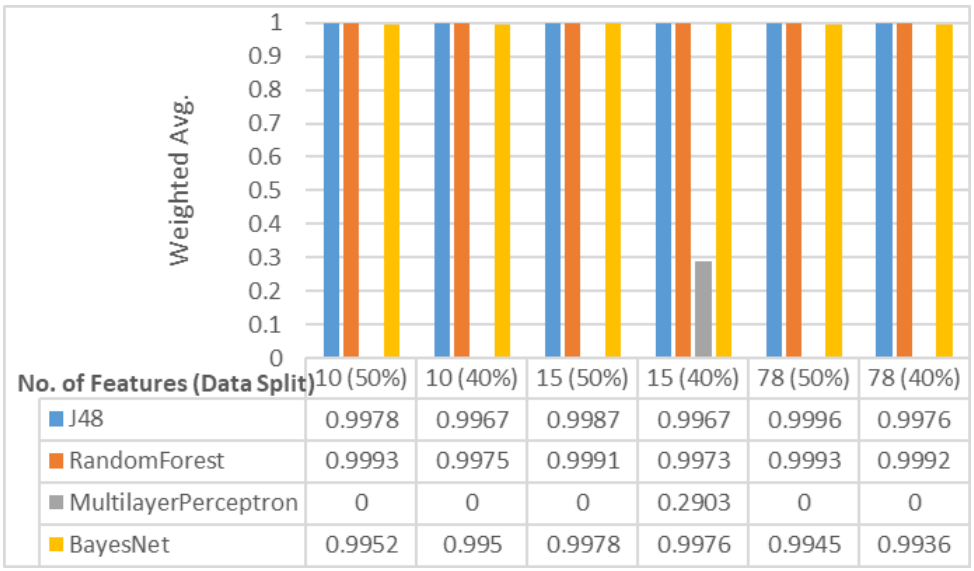
In the CIC-IDS2017 dataset, the data file representing the traffic of benign and brute force attacks has been used for analysis. The dataset comprises 78 features and 4,45,909 records labeled Benign, FTP-Patator, and SSH-Patator. In this chapter, for analysis, the various categories of brute force attacks have been represented collectively as BRUTEFORCE attacks. Thus, the transformed dataset has two types of labels - BENIGN and BRUTEFORCE. Out of the total 4,45,909 records, there are 4,32,074 instances of BENIGN and 13,835 instances of BRUTEFORCE.

6.3.2.2 Analysis

Analysis of the brute force attack has been carried out on the popular machine learning algorithms - J48, Random Forest, Multi-Layer Perceptron (MLP), and Bayes Net using the Weka tool. A part of the dataset is used to train the classifier and generate a model, while the remaining data is used to evaluate the model's performance. The output produced by the Weka tool is a confusion matrix (along with other details) for each unique combination of the testing dataset and the classifier. The brute force attack-based data file of the CIC-IDS2017 dataset is used for a 50% and 40% data split, i.e., 50% and 40% instances, respectively, to train the model, and the remaining for testing the model.

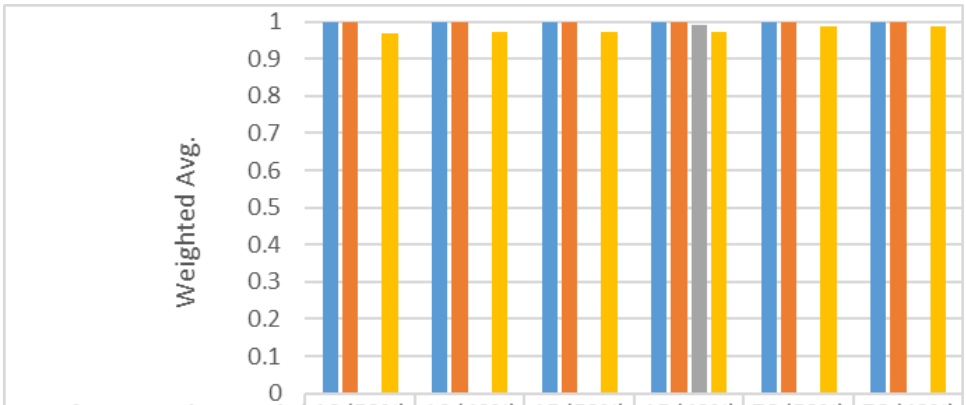
The dataset consists of 78 features. An exhaustive analysis of the brute force attack is carried out on the complete dataset with 78 features. To study the most significant features out of the 78 features, the InfoGainAttributeEval ranking method has been used to rank the 78 features. An analysis of the brute force attack is carried out on the dataset of the top 15 and top 10 ranked features. A comparative study of the dataset having 78 features, 15 features, and 10 features is done for various metrics, including performance metrics.

The accuracy, precision, recall, and f-score are calculated using Equations 6.1, 6.2, 6.3, and 6.4 on the generated confusion matrix for J48, Random Forest, MLP, and Bayes Net for 50% and 40% data split and different feature groups. Figures 6.7, 6.8, 6.9, and 6.10 present graphical representations and corresponding data tables summarizing the computed values of recall, precision, accuracy, and f-score, respectively.



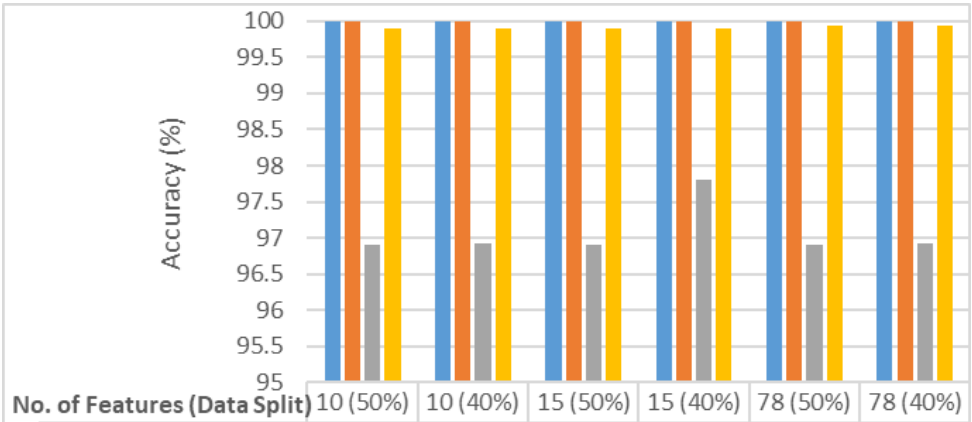
No. of Features (Data Split)	10 (50%)	10 (40%)	15 (50%)	15 (40%)	78 (50%)	78 (40%)
J48	0.9978	0.9967	0.9987	0.9967	0.9996	0.9976
RandomForest	0.9993	0.9975	0.9991	0.9973	0.9993	0.9992
MultilayerPerceptron	0	0	0	0.2903	0	0
BayesNet	0.9952	0.995	0.9978	0.9976	0.9945	0.9936

Figure 6.7: Recall for brute force



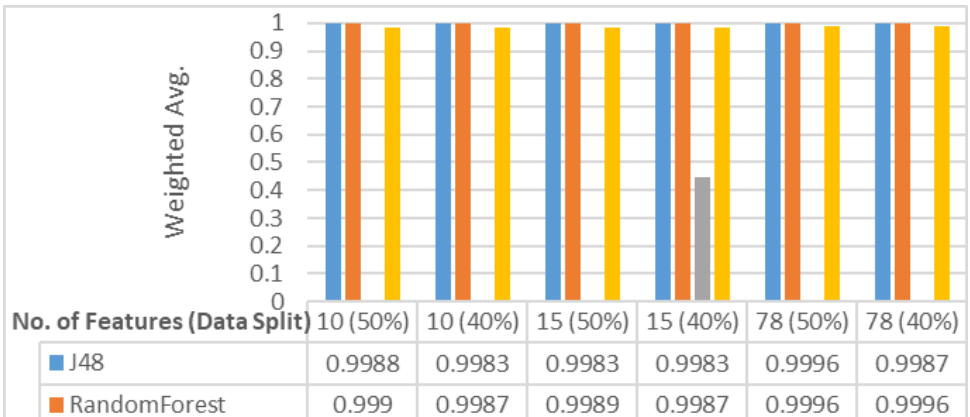
No. of Features (Data Split)	10 (50%)	10 (40%)	15 (50%)	15 (40%)	78 (50%)	78 (40%)
J48	0.9997	0.9999	0.998	0.9999	0.9997	0.9999
RandomForest	0.9987	1	0.9987	1	1	1
MultilayerPerceptron	0	0	0	0.9893	0	0
BayesNet	0.9692	0.9707	0.9704	0.9711	0.987	0.9865

Figure 6.8: Precision for brute force



No. of Features (Data Split)	10 (50%)	10 (40%)	15 (50%)	15 (40%)	78 (50%)	78 (40%)
J48	99.99	99.99	99.99	99.99	100	99.99
RandomForest	99.99	99.99	99.99	99.99	100	100
MultilayerPerceptron	96.91	96.92	96.91	97.8	96.91	96.92
BayesNet	99.89	99.89	99.9	99.9	99.94	99.94

Figure 6.9: Accuracy for brute force



No. of Features (Data Split)	10 (50%)	10 (40%)	15 (50%)	15 (40%)	78 (50%)	78 (40%)
J48	0.9988	0.9983	0.9983	0.9983	0.9996	0.9987
RandomForest	0.999	0.9987	0.9989	0.9987	0.9996	0.9996
MultilayerPerceptron	0	0	0	0.4489	0	0
BayesNet	0.982	0.9827	0.9839	0.9841	0.9907	0.99

Figure 6.10: F-Score for brute force

The model-built time (in seconds) for the combinations of features and ML algorithms for 50% and 40% of the data split is shown in Table 6.4.

Table 6.4: Model-built time (seconds) for brute force

No. of Features (Data Split)	10 (50%)	10 (40%)	15 (50%)	15 (40%)	78 (50%)	78 (40%)
J48	2.76	2.41	3.9	3.7	36.25	24.46
RandomForest	64.92	71.97	73.27	72.87	181.2	150.55
MultilayerPerceptron	155.71	402.3	409.1	434.69	10755.35	23803.72
BayesNet	3.55	3.29	5.62	5.06	34.45	27.69

6.3.2.3 Inference

It is observed from Figures 6.7, 6.8, 6.9, and 6.10 that for almost all the combinations of features and data split, the values of recall, precision, accuracy, and f-score are higher for J48 and Random Forest algorithms than MLP and Bayes Net algorithms. The metrics for the Bayes Net algorithm are close to J48 and Random Forest algorithms, but MLP shows a very low value (0 in many cases) for recall, precision, and f-score. It can be seen that with 10, 15, and 78 features, J48 and Random Forest algorithms achieve greater than 99% values of recall, precision, accuracy, and f-score. As depicted in Table 6.4, the algorithms perform best with 10 features, followed by 15 and 78 features. Thus, considering various metrics, including performance metrics, 10 features instead of 15 or 78 features, is good to consider when building the machine learning model to detect brute-force attacks. It can be seen from Table 6.4 that the J48 algorithm performs slightly better than the Random Forest algorithm for both 50% and 40% of the data split. Thus, in the proposed authentication solution, the ML algorithm model created using the J48 algorithm and considering 10 features can be chosen to detect brute-force attacks. The top 10 ranked features and entropy generated using the InfoGainAttributeEval ranking method are shown in Table 6.5.

Table 6.5: Top 10 ranked features for brute force

Entropy	Rank	Attribute Name
0.174	1	“Destination Port”
0.15	2	“Packet Length Mean”
0.148	3	“Packet Length Variance”
0.148	4	“Packet Length Std”
0.147	5	“Average Packet Size”
0.146	6	“Fwd Packet Length Mean”
0.146	7	“Avg Fwd Segment Size”
0.145	8	“Max Packet Length”
0.139	9	“Init_Win_bytes_backward”
0.137	10	“Fwd Packet Length Std”

The work in [49] and [97] demonstrates that “Packet Size” is one of the essential features in brute-force attacks. The work described in [88] experimentally shows that packet size distribution is useful in detecting brute-force attacks. In Table 6.5, 8 features (“Packet Length Mean”, “Packet Length Variance”, “Packet Length Std”, “Average Packet Size”, “Fwd Packet Length Mean”, “Avg Fwd Segment Size”, “Max Packet Length”, and “Fwd Packet Length Std”) out of the 10 features are directly proportional to the packet size. This further strengthens the outcome features in Table 6.5, as it is in sync with other research works.

6.3.3 Limitations

In this chapter, the CIC-IDS2017 dataset has been used to perform the experiments and analysis. This dataset draws the attention of many researchers as it is comprehensive and represents threats not addressed by the older datasets [89]. However, there is a lack of availability and a need for more comprehensive datasets in IDS that could be used to study different types of security attacks. One of the other limitations we faced during the experiments was the long execution time of the ML algorithms for some cases. In the case of experiments on DoS/DDoS attacks, the model-built time for MLP was approximately seven hours for 78 features with a 50% data split and about five hours for 78 features with a 40% data split (as per Table 6.2). In the case of experiments on brute-force attacks, the model-built time for MLP was approximately three hours for 78 features with a 50% data split and about six hours and thirty minutes for 78 features with a 40% data split (as per Table 6.4).

6.4 Security Analysis and Threat Modeling

This section determines the threat model, including the security objectives, trust boundaries, and possible threats. Additionally, the applicability of the proposed solution in real-world scenarios has been discussed.

6.4.1 Security Objectives

To maintain stability and resilience, a system must meet a range of essential security requirements. Identification represents a primary requirement in any confidential system. It is the act of finding out the identity of the person or application requesting access to a secure system. Any anonymous entity should be denied access to a secure system. Nonrepudiation is another essential security requirement. It ensures that an entity cannot repudiate the execution of a specific action. The transactions carried out by an authorized user should be accountable.

6.4.2 Trust Boundary

The proposed authentication scheme has four major components (authenticating device, authentication server, machine learning network, and database) and three major communication channels (marked by numbers 1-3) for transferring data, as shown in Figure 6.11.

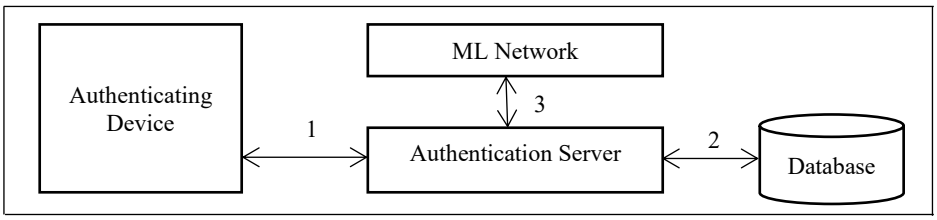


Figure 6.11: Trust boundary and the communication channels

Authenticating devices can be of different types, such as smartphones, laptops, etc., for various authentication methods. They represent a vulnerable component that can be exploited to carry out DoS/DDoS, brute force, and several other kinds of attacks. It is assumed that the authentication server and the database are secure components, as securing these components is taken care of by existing authentication systems. The ML network is a central on-cloud network where the technology stack can be easily updated without

affecting end users. It can be accessed only by the authentication server through the exchange of credentials. Various machine learning models are deployed and updated in the ML network as required. The cloud service provider manages the security of the cloud network, and it is not within the scope of discussion in this chapter.

SSL or TLS is a standard security technology for establishing an encrypted communication channel between a server and a client. In the proposed system, the communication between an authenticating device and the authentication server occurs via the Internet and is secured using TLS or SSL. Existing authentication systems are presumed to ensure the secure communication between the authentication server and the database. The authentication server establishes a secure channel via TLS or SSL to communicate with the ML network.

6.4.3 Threats

In today's cybersecurity landscape, there are numerous streams of potential threats. The threats look for vulnerabilities in the environment that they can exploit. In system and network security, threats can be mitigated using security features and procedures. The threats could be on the components, communication channels, and system authentication protocol.

6.4.3.1 Attacks on the Components

The proposed authentication scheme has four major components: an authenticating device, an authentication server, an ML network, and a database. There are various types of authenticating devices depending on the different authentication methods. The machines are geographically dispersed and are vulnerable to several types of attacks. Once a device has been compromised, it can be exploited to carry out further attacks such as DoS/DDoS, brute force, etc. In the proposed research work, the contextual information of the users and devices, such as network address, presence of various kinds of software in the machine, location information, analysis of packets originating from the device, behavioral analysis of the users using the devices, etc. are considered as input parameters to determine the genuineness of the authenticating devices and the users. Thus, the characteristics of an untrusted device are analyzed during the authentication process to determine any negative pattern. As mentioned earlier, the authentication server, database, and ML network are

assumed to be secure, and the security of these individual components is not within the scope of this chapter.

6.4.3.2 Attacks on the Communication Channels

During authentication, sensitive information is exchanged through the communication channel; thus, safeguarding the communication channel is of utmost importance. As depicted in Figure 6.11, there are three primary communication channels in the proposed system, which are marked by numbers 1-3. Communication between an authenticating device and the authentication server, as well as between the authentication server and the ML network, is protected by TLS/SSL. TLS/SSL certificates have varying periods of validity, and as long as the certificate is valid, the communication is carried out securely. TLS/SSL helps to protect against data theft, identity theft, eavesdropping, man-in-the-middle attacks, etc., as it encrypts all data in transit. As mentioned earlier, existing authentication systems are presumed to ensure the secure communication between the authentication server and the database, and it is not in the scope of this chapter.

6.4.3.3 Attacks on the Authentication Protocol

The volume of attacks during the authentication process and thereafter is increasing in the threat landscape. There could be several kinds of attacks, such as brute force, dictionary, DoS/DDoS, phishing, and man-in-the-middle attacks, among others, that an attacker can exploit. In our research, apart from the usual authentication methods currently in practice, the context of users and devices is considered during authentication to detect anomalies and security-related attacks. An independent machine learning network processes and analyses the contextual information to determine the credibility of the source requesting authentication.

6.4.4 Applicability

In the proposed solution, contextual analysis of the data is carried out in an on-cloud ML network. This significantly reduces the load and processing time of the authentication server and facilitates the implementation of the proposed solution in real-world scenarios, including resource-constrained infrastructures. The methodology and algorithms developed in our research can be used to detect intrusion in many applications that attempt to identify

constantly changing threats and attacks. The contextual information can be used to analyse different types of malware. Specifically, the developed ML model for DoS/DDoS attacks can detect such attacks in web applications, remote systems accessible on the network, etc. Likewise, the developed ML model for brute-force attacks can be used to detect such attacks in protected resources and systems, including web applications, which require authentication.

6.5 Conclusion

In this chapter, we have proposed an authentication solution that uses the context of users and devices to detect anomalies in security-related attacks. In particular, DoS/DDoS attacks and brute-force attacks have been analyzed in detail using contextual information. The experiments were performed on the CIC-IDS2017 dataset, containing data about different security attacks. The Weka tool has been used to perform the experiments and analysis. The ML models have been generated on four machine learning classifiers - J48, Random Forest, MLP, and Bayes Net, as well as different dataset splits and feature group combinations. To study the most significant features out of the total of 78 features, the InfoGainAttributeEval ranking method has been used to rank the 78 features. In the case of DoS/DDoS attacks, the experiments and results inferred that the ML model created using the Random Forest algorithm and 14 features was superior. In the case of brute-force attacks, the experiments and results inferred that the Random Forest algorithm and the J48 algorithm with 10 features are superior to the other ML models in detecting brute-force attacks. Security analysis and threat modeling have been performed to show the effectiveness of the proposed authentication scheme in enhancing a secure system's security level.

Chapter 7

Conclusions

7.1 Overview

This chapter summarizes the contribution of the present research work in the field of cybersecurity with particular emphasis on authentication. In addition, future research directions have been stated to enhance the proposed system's robustness further and address emerging challenges in the complex cybersecurity landscape.

The security attacks on different authentication mechanisms, particularly the use of authentication credentials (e.g., username and password), are increasing to an alarming extent. Thus, deploying appropriate solutions during authentication is crucial to protect both users and systems from potential security threats. In this thesis, four distinct authentication solutions cater to different security requirements, ensuring a comprehensive approach to safeguarding users and systems. The machine learning-based contextual authentication framework offers versatility, applicable to both user and device authentication. A thorough security analysis and threat modeling have been conducted to assess the effectiveness of these solutions in enhancing the security posture of protected systems.

7.2 Contributions

This thesis presents a novel authentication framework designed for both user and device authentication. The proposed authentication solutions effectively combat various security

threats across diverse authentication scenarios. The contributions of this research work are summarized below:

1. Chapter 3 presents a novel authentication solution based on a QR code and a multi-dimensional hash chain. This solution addresses prevalent security issues in several related works mentioned in the chapter. The working principle and algorithms are discussed in detail, and security analysis and threat modeling show that the proposed system effectively protects against vulnerable attacks that may occur at the time of authentication. The discussion concludes that it is useful in countering security attacks, including replay or playback attacks, host impersonation attacks, pre-play attacks, shoulder surfing, server crash attacks, stolen devices, man-in-the-middle attacks, non-repudiation attacks, key-logger, asterisk logger, and others. The proposed scheme, with respect to certain parameters, turns out to be more advantageous than SMS-based OTP, which is widely used in many secure applications. Performance testing and analysis of the algorithm show that it can be used for authentication on devices with limited resources.
2. Chapter 4 presents a novel authentication solution using contextual QR codes. The proposed solution computes a one-time password for each user login. It is compared to other state-of-the-art OTP-based authentication systems, such as SMS-based OTP, Lamport and S/Key scheme, and RSA SecurID, and is found to be advantageous in terms of several parameters. Security analysis establishes that the proposed solution can counter various security attacks, such as phishing attacks, key loggers, asterisk loggers, Trojan attacks, replay or playback attacks, shoulder surfing, brute-force and dictionary attacks, SQL injection, and pre-play attacks. Additionally, the proposed solution is easy to integrate into existing secure systems. A usability study was conducted to evaluate the trade-off between authentication time and system security, allowing for deployment based on system requirements.
3. Chapter 5 proposes a novel authentication solution for ATM networks. Currently, the most common and widely used method to authenticate at an ATM kiosk is using an ATM card and its associated PIN. However, this method has drawbacks, such as the need to carry an additional ATM card, manufacturing and securely shipping the ATM card, vulnerability to ATM card skimming, etc. The proposed solution utilizes NFC and blockchain technologies and eliminates the need for physical ATM cards. The solution addresses issues present in existing related research works. Security

analysis and threat modeling show that the proposed system effectively protects against vulnerable attacks that may occur at the time of authentication. The device-specific unique information serves as an additional validation factor, alongside credentials, to authenticate the requesting source. Furthermore, digital identity is stored in the blockchain to facilitate the two-factor authentication of system users.

4. Chapter 6 proposes a solution that considers the context of users and devices during authentication to detect anomalies and security-related attacks. An anomaly-based intrusion detection system is modelled and analyzed in detail. The machine learning models are experimentally generated for DoS/DDoS and brute force attacks. An analysis of accuracy, precision, recall, f-score, and execution time shows that these models outperform those generated in existing research works. Security analysis and threat modeling demonstrate the effectiveness of the proposed authentication scheme in enhancing the security level of secure systems. The methodology and algorithms developed in this research can be applied to detect intrusion in various applications that aim to identify constantly evolving threats and attacks.

7.3 Future Works

This thesis has addressed several security attacks that can be mitigated using the proposed authentication solutions. However, security attacks are constantly evolving and becoming more sophisticated. As part of future research work, an exhaustive analysis should be conducted on other types of security attacks that have not been addressed and discussed within the scope of the proposed solutions. Furthermore, the adaptability of the proposed authentication framework in different use cases should be studied to determine its applicability in various systems and services.

A QR code has been utilized in two proposed solutions to facilitate information transfer. Studies have been conducted to analyse the factors determining a user's intention to use the QR code [29]. Future work should focus on designing more user-oriented QR codes, thereby enhancing the user experience.

Chapter 4 presents a solution to derive One-Time Passwords (OTPs) based on dynamically changing mapping information. A usability study was conducted to determine the error rate and the time taken by users to authenticate using the proposed method. Future work should

involve increasing the number of participants and conducting several iterations of authentication over a considerable period to provide a holistic result of the usability study.

In Chapter 6, a solution has been proposed that considers the context of users and devices during authentication to detect anomalies and security-related attacks. Apart from the mentioned contexts, various other types of contexts should be further explored in detail. Additionally, future work should involve a detailed analysis of several other kinds of attacks (apart from DoS/DDoS and brute force) during authentication, different datasets, and their group of features to make the system more robust.

References

[1] A. T. Shah, V. R. Parihar, “Overview and an approach for QR-code based messaging and file sharing on android platform in view of security”, International Conference on Computing Methodologies and Communication (ICCMC), Erode, India (2017).

[2] A. V. Kayem, “Graphical Passwords -- A Discussion”, 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA), Crans-Montana, Switzerland (2016).

[3] A. Mandalapu, D. Deepa, L. Raj, A. Dev, “An NFC featured three level authentication system for tenable transaction and abridgment of ATM card blocking intricacies”, IEEE International Conference and Workshop on Computing and Communication (IEMCON), Vancouver, Canada (2015).

[4] A. Muley, V. Kute, “Prospective solution to bank card system using fingerprint”, IEEE 2nd International Conference on Inventive Systems and Control (ICISC), Coimbatore, India (2018).

[5] A. Kowshika, P. Sumathi, K. S. Sandra, A. S. Kumar, R. Gokulkrishnan, “FACEPIN: Face Biometric Authentication System For ATM Using Deep Learning”, NVEO Journal, Vol. 9(1), pp. 1859-1872 (2022).

[6] A. Shakevsky, E. Ronen, A. Wool, “Trust Dies in Darkness: Shedding Light on Samsung’s TrustZone Keymaster Design”, 31st USENIX Security Symposium, Boston, USA (2022).

[7] A. L. Buczak, E. Guven, “A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection”, IEEE Communications Surveys & Tutorials, Vol. 18(2), pp.1153-1176 (2016).

[8] A. Halbouni, T. S. Gunawan, M. H. Habaebi, “Machine Learning and Deep Learning Approaches for CyberSecurity: A Review”, IEEE Access, Vol. 10, pp.19572-19585 (2022).

[9] A. Desai, S. Rai, “Analysis of Machine Learning Algorithms using Weka”, International Conference and Workshop on Emerging Trends in Technology, Vol. ICWET2012(13), pp.32-37 (2012).

[10] A. AlEroud, G. Karabatis, “Using Contextual Information to Identify Cyber-Attacks”, Information Fusion for Cyber-Security Analytics, Vol. 691, pp.1-16 (2017).

[11] A. A. Bello, H. Chiroma, A. Y. Gital, “Machine learning algorithms for improving security on touch screen devices: a survey, challenges and new perspectives”, Neural Computing and Applications, Vol. 32, pp.13651-13678 (2020).

[12] A. Z. Agghey, L. J. Mwinuka, S. M. Pandhare, M. A. Dida, J. D. Ndibwile, “Detection of Username Enumeration Attack on SSH Protocol: Machine Learning Approach”, Symmetry, Vol. 13(11), pp.2192 (2021).

[13] A. P. Pandian, S. Smys, “DDOS attack detection in telecommunication network using machine learning”, Journal of Ubiquitous Computing and Communication Technologies, Vol. 1(1), pp.33-44 (2019).

[14] A. Aparicio, M. M. Mart´ınez-Gonz´alez, V. Carde˜noso-Payo, “App-based detection of vulnerable implementations of OTP SMS APIs in the banking sector”, Wireless Networks, Vol. 30, pp.6451–6464 (2024).

[15] B. Malek, M. Orozco, A. E. Saddik, “Novel Shoulder-Surfing Resistant Haptic-based Graphical Password”, EuroHaptics 2006 Conference, Paris, France (2006).

[16] B. McGillion, T. Dettenborn, T. Nyman, N. Asokan, “Open-TEE - An OpenVirtual Trusted Execution Environment”, 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Helsinki, Finland (2015).

[17] C. Huang, Y. Zhang, L. Lin, W. Wang, S. Wang, “Mutual authentications to parties with QR-code applications in mobile systems”, International Journal of Information Security, Vol. 16(5), pp.525-540 (2017).

[18] C. Park, “One-time password based on hash chain without shared secret and re-registration”, Computers & Security, Vol. 75, pp.138-146 (2018).

[19] C. Ozkan, K. Bicakci, “Security Analysis of Mobile Authenticator Applications”, IEEE International Conference on Information Security and Cryptology (ISCTURKEY), Ankara, Turkey (2020).

[20] C. Borkotoky, S. Galgate, S. B. Nimbekar, “Human computer interaction harnessing P300 potential brain waves for authentication of individuals”, ACM 1st Bangalore Annual Compute Conference (COMPUTE 2008), Bangalore, India (2008).

[21] C. Mulliner, R. Borgaonkar, P. Stewin, J. P. Seifert, “SMS-Based One-Time Passwords: Attacks and Defense”, Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2013), LNCS, Vol. 7967, pp.150-159 (2013).

[22] C. M. Manish, N. Chirag, H. R. Praveen, M. J. Darshan, D. K. Vali, “Card-Less ATM Transaction using Biometric and Face Recognition - A Review”, International Journal for Research in Applied Science & Engineering Technology (IJRASET), Vol. 8(7), pp.1493-1498 (2020).

[23] C. Perera, A. Zaslavsky, P. Christen, D. Georgakopoulos, “Context Aware Computing for The Internet of Things: A Survey”, IEEE Communications Surveys & Tutorials, Vol. 16(1), pp.414-454 (2014).

[24] D. Kogan, N. Manohar, D. Boneh, “T/Key: second-factor authentication from secure hash chains”, ACM SIGSAC Conference on Computer and Communications Security, Dallas, USA (2017).

[25] D. M’Raihi, M. Bellare, F. Hoornaert, D. Naccache, O. Ranen, “HOTP: An HMAC-Based One-Time Password Algorithm”, Internet Engineering Task Force (IETF) RFC 4226 (2005).

[26] D. M’Raihi, S. Machani, M. Pei, J. Rydell, “TOTP: Time-Based One-Time Password Algorithm”, Internet Engineering Task Force (IETF) RFC 6238 (2011).

[27] D. Conde-Lagoa, E. Costa-Montenegro, F. J. Gonzalez-Castano, F. Gil-Castineira, “Secure eTickets based on QR-codes with user-encrypted content”, Digest of Technical Papers International Conference on Consumer Electronics (ICCE), Las Vegas, USA (2010).

[28] D. Gupta, “A new approach of authentication in graphical systems using ASCII submission of values”, 13th International Wireless Communications and Mobile Computing Conference (IWCMC), Valencia, Spain (2017).

[29] D. H. Shin, J. Jung, B. H. Chang, “The psychology behind QR codes: User experience perspective”, Computers in Human Behavior, Vol. 28(4), pp.1417-1426 (2012).

[30] D. Nguyen, D. Nguyen-Duc, N. Huynh-Tuong, H. Pham, “CVSS: A Blockchainized Certificate Verifying Support System”, ACM SoICT 2018: The Ninth International Symposium on Information and Communication Technology, Danang, Vietnam (2018).

[31] D. J. Gunn, et al., “Touch-Based Active Cloud Authentication Using Traditional Machine Learning and LSTM on a Distributed Tensorflow Framework”, International Journal of Computational Intelligence and Applications, Vol. 18(4), pp.1950022 (2019).

[32] F. Binbeshr, L. Y. Por, M. L. M. Kiah, A. A. Zaidan, M. Imam, “Secure PIN-Entry Method Using One-Time PIN (OTP)”, IEEE Access, Vol. 11, pp.18121-18133 (2023).

[33] F. Al-Turjman, S. Alturjman, “Context-sensitive access in industrial Internet of Things (IIoT) healthcare applications”, IEEE Transactions on Industrial Informatics, Vol. 14(6), pp.2736-2744 (2018).

[34] F. J. Aparicio-Navarro, K. G. Kyriakopoulos, I. Ghafir, S. Lambotharan, J. A. Chambers, “Multi-Stage Attack Detection Using Contextual Information”, IEEE Military Communications Conference (MILCOM), Los Angeles, USA (2018).

[35] G. Shangfu, L. Jun, S. Yizhen, “Design and Implementation of Anti-Screenshot Virtual Keyboard Applied in Online Banking”, International Conference on E-Business and E-Government (ICEE), Guangzhou, China (2010).

[36] G. Apruzzese, M. Colajanni, L. Ferretti, A. Guido, M. Marchetti, “On the effectiveness of machine and deep learning for cyber security”, IEEE 10th International Conference on Cyber Conflict (CyCon), Tallinn, Estonia (2018).

[37] H. Lee, W. C. Hong, C. H. Kao, C. M. Cheng, “A User-Friendly Authentication Solution Using NFC Card Emulation on Android”, IEEE 7th International Conference on Service-Oriented Computing and Applications, Matsue, Japan (2014).

[38] H. Raj, et al., “fTPM: A Software-Only Implementation of a TPM Chip”, 25th USENIX Security Symposium, Texas, USA (2016).

[39] I. Barron, H. J. Yeh, K. Dinesh, G. Sharma, “Dual modulated QR codes for proximal privacy and security”, IEEE Transactions on Image Processing, Vol. 30, pp.657-669 (2020).

[40] I. Tkachenko, W. Puech, C. Destruel, O. Strauss, J. Gaudin, C. Guichard, “Two-level QR code for private message sharing and document authentication”, IEEE Transactions on Information Forensics and Security, Vol. 11(3), pp.571-583 (2016).

[41] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani, “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization”, International Conference on Information Systems Security and Privacy (ICISSP 2018), Funchal, Portugal (2018).

[42] I. H. Witten, E. Frank, M. A. Hall, C. J. Pal, "Data Mining: Practical Machine Learning Tools and Techniques", Morgan Kaufmann, 4th Edition (2016).

[43] J. X. Zhao, “Research and Design on an Improved TOTP Authentication”, Applied Mechanics and Materials, Vol. 411-414, pp.595-599 (2013).

[44] J. Brainard, et al., “Fourth-factor authentication: somebody you know”, 13th ACM Conference on Computer and Communications Security, Virginia, USA (2006).

[45] J. Rouillard, “Contextual QR codes”, Third International Multi-Conference on Computing in the Global Information Technology (ICCGI 2008), Athens, Greece (2008).

[46] J. Ekberg, K. Kostiainen, N. Asokan, “The Untapped Potential of Trusted Execution Environments on Mobile Devices”, IEEE Security & Privacy, Vol. 12(4), pp.29-37 (2014).

[47] J. M. Ackerson, R. Dave, N. Seliya, “Applications of recurrent neural network for biometric authentication & anomaly detection”, Information, Vol. 12(7), pp.272 (2021).

[48] J. Wang, L. Wang, “SDN-defend: a lightweight online attack detection and mitigation system for DDoS attacks in SDN”, Sensors, Vol. 22(21), pp.8287 (2022).

[49] J. Luxemburk, K. Hynek, T. Cejka, “Detection of HTTPS Brute-Force Attacks with Packet-Level Feature Set”, IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC), NV, USA (2021).

[50] J-P. Aumasson, S. Neves, J. O'Connor, Z. Wilcox, "The BLAKE3 Hashing Framework", Internet Engineering Task Force (2024).

[51] J. Pearl, "Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference", Morgan Kaufmann Publishers (1988).

[52] K. Bicakci, N. Baykal, “Infinite length hash chains and their applications”, Eleventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE’02), Pittsburgh, USA (2002).

[53] K. Saranya, R. S. Reminaa, S. Subhitsha, “Modern applications of QR-code for security”, 2nd IEEE International Conference on Engineering and Technology (ICETECH), Coimbatore, India (2016).

[54] Kurniabudi, et al., “CICIDS-2017 Dataset Feature Analysis With Information Gain for Anomaly Detection”, IEEE Access, Vol. 8, pp.132911-132921 (2020).

[55] K. Kumari, M. Mrunalini, “Detecting Denial of Service attacks using machine learning algorithms”, Journal of Big Data, Vol. 9(1) (2022).

[56] L. Lumburovska, J. Dobreva, S. Andonov, H. M. Trpcheska, V. Dimitrova, “A Comparative Analysis of HOTP and TOTP Authentication Algorithms. Which one to choose?”, Security & Future, Vol. 5(4), pp.131-136 (2021).

[57] L. Lamport, “Password authentication with insecure communication”, Communications of the ACM, Vol. 24(11), pp.770-772 (1981).

[58] L. Zhang, H. Li, L. Sun, Z. Shi, Y. He, “Towards Fully Distributed User Authentication with Blockchain”, IEEE Symposium on Privacy-Aware Computing (PAC), Washington, USA (2017).

[59] L. Pryor, R. Dave, J. Seliya, E. S. Boone, “Machine Learning Algorithms In User Authentication Schemes”, IEEE International Conference on Electrical, Computer and Energy Technologies (ICECET), Cape Town, South Africa (2021).

[60] L. Zhou, M. Liao, C. Yuan, Z. Sheng, H. Zhang, “DDOS attack detection using packet size interval”, 11th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2015), Shanghai, China (2015).

[61] L. Breiman, "Random Forests", Machine Learning, Vol. 45(1), pp.5-32 (2001).

[62] M. B. Krishna, A. Dugar, “Product authentication using QR codes: A mobile application to combat counterfeiting”, Wireless Personal Communications, Vol. 90, pp.381-398 (2016).

[63] M. Eldefrawy, M. Khan, K. Alghathbar, T. Kim, H. Elkamchouchi, “Mobile one-time passwords: two-factor authentication using mobile phones”, Journal of Security and Communication Networks, Vol. 5, pp.508-516 (2012).

[64] M. M. Potey, C. A. Dhote, D. H. Sharma, “Secure authentication for data protection in cloud computing using color schemes”, International Conference on Computational Systems and Information Systems for Sustainable Solutions (CSITSS), Bangalore, India (2016).

[65] M. Kumar, T. Garfinkel, D. Boneh, T. Winograd, “Reducing shoulder-surfing by using gaze based password entry”, 3rd Symposium on Usable Privacy and Security (SOUPS 2007), Pittsburgh, USA (2007).

[66] M. Singh, D. Garg, “Choosing Best Hashing Strategies and Hash Functions”, IEEE International Advance Computing Conference, Patiala, India (2009).

[67] M. H. Hammi, B. Hammi, P. Bellot, A. Serhrouchni, “Bubbles of Trust: A decentralized blockchain-based authentication system for IoT”, Computer & Security, Vol. 78, pp.126-142 (2018).

[68] M. Gheisari, H. E. Najafabadi, J. A. Alzubi, et. al., “OBPP: an ontology-based framework for privacy-preserving in IoT-based smart city”, Future Generation Computer Systems, Vol. 123, pp.1-13 (2021).

[69] M. Tao, J. Zuo, “Multi-layer cloud architectural model and ontology-based security service framework for IoT-based smart homes”, Future Generation Computer Systems, Vol. 78(3), pp.1040-1051 (2018).

[70] M. Alkasassbeh, G. A. Naymat, A. B. Hassanat, M. Almseidin, “Detecting Distributed Denial of Service Attacks Using Data Mining Techniques”, International Journal of Advanced Computer Science and Applications (ijacsa), Vol. 7(1) (2016).

[71] M. D. Hossain, H. Ochiai, F. Doudou, Y. Kadobayashi, “SSH and FTP brute-force attacks detection in computer networks: LSTM and machine learning approaches”, IEEE 5th International Conference on Computer and Communication Systems (ICCCS), Shanghai, China (2020).

[72] M. Ilyas, S. A. Alharbi, “Machine learning approaches to network intrusion detection for contemporary internet traffic”, Springer Computing, Vol. 104, pp.1061-1076 (2022).

[73] M. Ahsan, R. Gomes, M. M. Chowdhury, K. E. Nygard, “Enhancing Machine Learning Prediction in Cybersecurity Using Dynamic Feature Selector”, Journal of Cybersecurity and Privacy, Vol. 1(1), pp.199-218 (2021).

[74] M. Bartlomiejczyk, I. E. Fray, “Device Risk Analysis Protocol for SMS-Based OTP Authentication”, IEEE Access, Vol. 12, pp.123177–123192 (2024).

[75] M-J. Saarinen, J-P. Aumasson, "The BLAKE2 Cryptographic Hash and Message Authentication Code (MAC)", RFC 7693 (2015).

[76] M. W. Gardner, S. R. Dorling, "Artificial neural networks (the multilayer perceptron)-a review of applications in the atmospheric sciences", Atmospheric Environment, Vol. 32(14–15), pp.2627–2636 (1998).

[77] National Institute of Standards and Technology (NIST), "Secure Hash Standard (SHS)", Federal Information Processing Standards Publication (FIPS) 180-4 (2015).

[78] N. Haller, “The S/KEY one-time password system”, ISOC Symposium on Network and Distributed System Security, San Diego, USA (1994).

[79] P. S. Saini, S. Behal, S. Bhatia, “Detection of DDoS Attacks using Machine Learning Algorithms”, IEEE 7th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India (2020).

[80] P. Du, S. Abe, “IP Packet Size Entropy-Based Scheme for Detection of DoS/DDoS Attacks”, IEICE Transactions on Information and Systems, Vol. 91(5), pp.1274-1281 (2008).

[81] R. H. Khan, J. Miah, “Performance Evaluation of a new one-time password (OTP) scheme using stochastic petri net (SPN)”, IEEE World AI IoT Congress (AIIoT), Seattle, USA (2022).

[82] R. D. Pietro, L. V. Mancini, A. Durante, V. Patil, “Addressing the shortcomings of one-way chains”, ACM Symposium on Information, Computer and Communications Security (ASIACCS '06), Taipei, Taiwan (2006).

[83] R. Durstenfeld, “Algorithm 235: Random permutation”, Communications of the ACM, Vol. 7(7), pp.420–421 (1964).

[84] R. Malutan, C. Grosan, “Web authentication methods using single sign on method and virtual keyboard”, Conference Grid, Cloud and High Performance Computing in Science (ROLCG), Cluj-Napoca, Romania (2015).

[85] R. M. Ranasinghe, G. Z. Yu, “RFID/NFC device with embedded fingerprint authentication system”, 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, China (2017).

[86] R. Selvakumar, S. Logesh, V. S. Maha, S. Maniraj, K. A. Praveen, “Face Biometric Authentication System for ATM using Deep Learning”, IEEE 3rd International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India (2022).

[87] R. Pote, S. Kulkarni, “Securing cash withdrawal from ATM with the help of Smart Mobile Banking Application”, IEEE Interdisciplinary Research in Technology and Management (IRTM), Kolkata, India (2022).

[88] R. Hofstede, M. Jonker, A. Sperotto, A. Pras, “Flow-Based Web Application Brute-Force Attack and Compromise Detection”, Journal of Network and Systems Management, Vol. 25, pp.735-758 (2017).

[89] R. Panigrahi, S. Borah, “A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems”, International Journal of Engineering & Technology, Vol. 7, pp.479-482 (2018).

[90] S. Ma, R. Feng, J. Li, Y. Liu, S. Nepal, Diethelm, E. Bertino, R. H. Deng, Z. Ma, S. Jha, “An empirical study of SMS one-time password authentication in Android apps”, ACM 35th Annual Computer Security Applications Conference (ACSAC ’19), San Juan, USA (2019).

[91] S. Wiedenbeck, J. Waters, L. Sobrado, J. C. Birget, “Design and evaluation of a shoulder-surfing resistant graphical password scheme”, Advanced Visual Interface (AVI 2006), Venezia, Italy (2006).

[92] S. M. Aljuaid, A. S. Ansari, “Automated Teller Machine Authentication Using Biometric”, Computer Systems Science & Engineering, Tech Science, Vol. 41(3), pp.1009-1025 (2022).

[93] S. Wankhede, D. Kshirsagar, "DoS Attack Detection Using Machine Learning and Neural Network", IEEE Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India (2018).

[94] S. Wanjau, G. Wambugu, G. Kamau, "SSH-Brute Force Attack Detection Model based on Deep Learning", International Journal of Computer Applications Technology and Research, Vol. 10(1), pp.42-50 (2021).

[95] S. Chabbi, N. E. Madhoun, L. Khamer, "Security of NFC Banking Transactions: Overview on Attacks and Solutions", 6th Cyber Security in Networking Conference (CSNet), Rio de Janeiro, Brazil (2022).

[96] S. Alam, H. A. Y. Ayoub, R. A. A. Alshaikh, A. H. H. AL-Hayawi, "A Blockchain-based framework for secure Educational Credentials", Turkish Journal of Computer and Mathematics Education (TURCOMAT), Vol. 12(10), pp.5157–5167 (2021).

[97] T. Lee, L. Chang, C. Syu, "Deep Learning Enabled Intrusion Detection and Prevention System over SDN Networks", IEEE International Conference on Communications Workshops (ICC Workshops), Dublin, Ireland (2020).

[98] T. Yeh, H. Shen, J. Hwang, "A Secure One-Time Password Authentication Scheme Using Smart Cards", IEICE Transactions on Communications, Vol. E85-B(11), pp. 2515-2518 (2002).

[99] U. Jayasinghe, G. M. Lee, T. W. Um, Q. Shi, "Machine Learning Based Trust Computational Model for IoT Services", IEEE Transactions on Sustainable Computing, Vol. 4(1), pp.39-52 (2019).

[100] V. N. H. Kollipara, S. K. Kalakota, S. Chamarthi, S. Ramani, P. Malik, M. Karuppiah, "Timestamp Based OTP and Enhanced RSA Key Exchange Scheme with SIT Encryption to Secure IoT Devices", Journal of Cyber Security and Mobility, Vol. 12(1), pp.77-102 (2023).

[101] V. Goyal, A. Abraham, S. Sanyal, S. Han, "The N/R one-time password system", International Conference on Information Technology: Coding and Computing (ITCC'05), Las Vegas, USA (2005).

[102] W. Xu, J. Tian, Y. Cao, S. Wang, "Challenge-Response Authentication Using In-Air Handwriting Style Verification", IEEE Transactions on Dependable and Secure Computing, Vol. 17(1), pp.51-64 (2020).

[103] W. Melo, R. Machado, L. Carmo, "Using Physical Context-Based Authentication against External Attacks: Models and Protocols", Security and Communication Networks, Vol. 2018(1), pp.1-14 (2018).

[104] X. Suo, Y. Zhu, G. S. Owen, "Graphical passwords: a survey", 21st Annual Computer Security Applications Conference (ACSAC'05), Tucson, USA (2005).

[105] Y. Kouraogo, G. Orhanou, S. Elhajji, "Advanced security of two-factor authentication system using stego QR code", International Journal of Information and Computer Security, Vol. 12(4), pp.436-449 (2020).

[106] Y. W. Kao, et al., "Physical access control based on QR code", International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, Beijing, China (2011).

[107] Y. Liu, Y. Zhou, S. Wen, C. Tang, "A Strategy on Selecting Performance Metrics for Classifier Evaluation", International Journal of Mobile Computing and Multimedia Communications, Vol. 6(4), pp.20-35 (2014).

15.01.2026

PROFESSOR
Dept. of Information Technology,
JADAVPUR UNIVERSITY
Salt Lake Campus, Block-LB/8, Sector-III
Salt Lake, Kolkata-700106, India

Divyans Mahansaria
15/01/2026