Journal of King Saud University – Computer and Information Sciences 35 (2023) 101785

HOSTED BY

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

The use of IoT-based wearable devices to ensure secure lightweight payments in FinTech applications

Sriramulu Bojjagani [a,*], Nagarjuna Reddy Seelam [b], Neeraj Kumar Sharma [a], Ravi Uyyala [c], Sree Rama Chandra Murthy Akuri [b], Anup Kumar Maurya [d]

[a] *Cyber Security Lab, Department of Computer Science and Engineering, School of Engineering and Applied Sciences (SEAS), SRM University-AP, Amaravati, 522503 India*
[b] *Department of Computer Science and Engineering, Lakireddy Bali Reddy College of Engineering, Mylavaram, JNTUK, Kakinada, A.P, India*
[c] *Department of CSE, Chaitanya Bharathi Institute of Technology (CBIT), Gandipet, Hyderabad 500075, India*
[d] *Goa Institute of Management, Goa, India*

ARTICLE INFO

Article history:
Received 27 May 2023
Revised 8 September 2023
Accepted 27 September 2023
Available online 5 October 2023

Keywords:
Wearable payments
FinTech
ECIES
ECDSA
NFC
RoR

ABSTRACT

Daily digital payments in Financial Technology (FinTech) are growing exponentially. A huge demand is for developing secure, lightweight cryptography protocols for wearable IoT-based devices. The devices hold the consumer information and transit functions in a secure environment to provide authentication and confidentiality using contactless Near-Field Communication (NFC) or Bluetooth technologies. On the other hand, Security breaches have been observed in various dimensions, especially in wearable payment technologies. In this paper, we developed a threat model in the proposed framework and how to mitigate these attacks. This study accepts the three-authentication factor, as biometrics is one of the user's most vital authentication mechanisms. The scheme uses an "Elliptic Curve Integrated Encryption Scheme (ECIES)", "Elliptic Curve Digital Signature Algorithm (ECDSA)" and "Advanced Encryption Standard (AES)" to encrypt the messages between the entities to ensure higher security. The security analysis of the proposed scheme is demonstrated through the Real-or-Random oracle model (RoR) and Scyther's widely accepted model-checking tools. Finally, we present a comparative summary based on security features, communication cost, and computation overhead of existing methods, specifying that the proposed framework is secure and efficient for all kinds of remote and proximity payments, such as mini, macro, and micro-payments, using wearable devices.

1. Introduction

Wearable payments are becoming an extension of digital payments. FinTech wearable applications refer to electronic devices embedded into items or accessories worn. They can do tasks just as a mobile or a computer can. It is a part of the emerging Internet of Things (IoT) trend, which helps optimize operations, increase productivity, boost revenue, and improve lives (Challa et al., 2017; Chen et al., 2019). Wearable technology can be categorized into two main categories: business and personal usage. Parts of physical objects are embedded with sensors and software for individual usage as activity trackers. For business-oriented purposes, wearable devices can be effectively used for transactions in global markets. The modes or channels of digital payments using smartphone applications are more. Nowadays, most people are interested in buying products through digital payments, especially wearable devices–the massive demand for wearable device payments in global payment markets. The enormous growth of payment market value for connected devices globally from 2015 to 2025 is shown in Fig. 1. It provides ten-year market forecasts of wearable transactions and transaction volume by region, device type, and technology from 2015 through 2025[1]. Wearable devices contribute toward banking and payments and provide better services than smartphone devices. Wearable devices have tremendous potential to provide better and more efficient services to end-users; for this reason, most banks are implementing wearable tech-

* Corresponding author.
E-mail address: sriramulubojjagani@gmail.com (S. Bojjagani).

Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

[1] https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/.

https://doi.org/10.1016/j.jksuci.2023.101785
1319-1578/© 2023 The Author(s). Published by Elsevier B.V. on behalf of King Saud University.

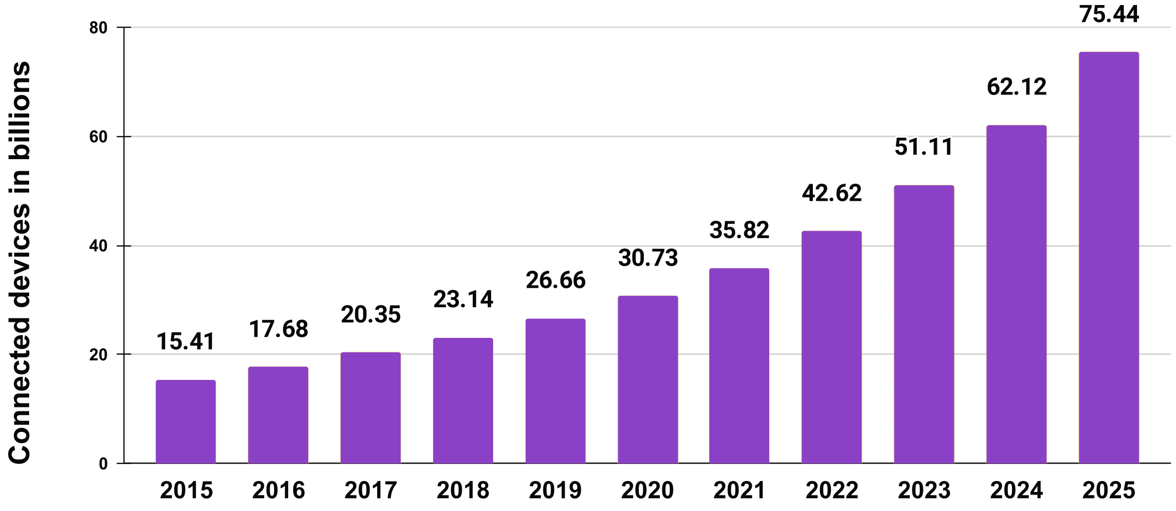


Fig. 1. IoT connected devices installed base worldwide from 2015 to 2025 (in billions)[1].

Table 1
The differences between offline, e-commerce and m-commerce approaches.

Feature	Offline	E-commerce	M-commerce
Security	C2M, C2C	2-factor authentication	3-factor authentication
Push notifications	NA	Minimum	Maximum
Location Tracking	NA	Location tracking and providing localized offers can become tedious when using e-commerce	location tracking is easy with mobile phones, as they have built-in GPS
Portability	NA	Highly not portable	Highly portable
Omnichannel	Limited	Limited	personalization and a customized shopping

nology as one of the essential Costs of Goods Sold (COGS) (Seneviratne et al., 2017). The goods are directly related to costs and expenses. COGS exempts indirect product costs such as marketing, retailing sales and overhead (Sedita et al., 2018). It is deducted from sales to calculate gross margin and gross profit. Higher COGS results in lower margins (Bezovski, 2016).

COGS differs from OPerating EXpenses (OPEX); it includes expenditures not directly tied to producing goods or services (Jason Fernando, 2022). Wearable and IoT devices have more significant usage in the medical domain in evaluating patient health conditions such as body temperature, glucose levels, heart rate, etc., transmitted these parameter values to the consumer's smartphone (Kumar and Grover, 2019). Some of the applications of wearable devices, such as a wearable, can help solve diagnostic issues with the patient through telemedicine and telehealth. Customers can directly buy products with the merchant or third-party vendors, etc. Bojjagani et al. (2022, 2023). Wearable devices to other gadgets increase productivity and efficient decision-making by FinTech, which connects the Internet of Things. Nowadays, most people are shown to be interested in purchasing products and services and banking using digital payments using wearable devices and smartphones. The wearable devices send a user's sensitive information to the mobile terminal, thus establishing secure communication. There is a requirement to design a secure protocol for macro payments between wearables and smartphones. Such schemes provide secure end-to-end payments. Wearable technology is an add-on to smart devices; the primary advantage of using macro payments with wearables is reducing the payment process time and providing handy convenience. In addition, it can be adopted in online m-commerce, e-commerce, and offline-physical stores. For the offline stores, there is no need for security between Customer-to-Merchant (C2M) and Customer-to-Customer (C2C) because of direct payment with cash. Table 1 shows the distinction between these approaches. Any related schemes use Bluetooth technology (Lo and Yohan, 2020; Diallo et al., 2014) for pairing wearable and smartphone payments, but these schemes only solve a few malicious attacks at the application level. As the wave of wearables takes hold, consumers expect their wearable devices to support secure payments, authentication mechanisms, and transit functions via contactless Bluetooth Low Energy (BLE) technologies. The NFC-based payment system is a globally accepted and convenient way to pay at compatible terminals worldwide. In this paper, the authors will use "near-field communication (NFC)" for device pairing because it is a newly emerging wireless technology for secure communication. Compared with Bluetooth technology NFC has significant features such as better communication range and less chance of interference with other devices.

An IoT-based macro payment protocol is generally divided into three stages: registration, user authentication, transaction initialization, and the notification phase. In this paper, a secure end-to-end macro payment is designed and proposed using a wearable device. Apart from that, the authors have used security analysis with the advanced simulation tools of Scyther and RoR.

1.1. Research contributions

An IoT-based macro payment system for consumers using wearable devices is designed and proposed to apply for all types of payments, such as mini, micro, and macro, reduce the protocol deployment cost, increase user convenience and provide end-to-end security between the parties, and efficient system performance. The work focuses on unsolved problems to ensure device authentication between wearable and traditional devices. Some unsolved problems are:

1. The existing schemes do not need to consider deploying trusted third parties into the payment applications.
2. Some existing schemes are (Patel et al., 2015; Kumar and Grover, 2019), which use device pairing with wearable devices using RFID and Bluetooth to communicate large, secured messages. However, the proposed protocol uses NFC as a device pairing to build a secure channel between the entities.
3. We have implemented a threat model that primarily focuses on cyber security fields of Application-level security, communication-level security, and device-level security.
4. We addressed and resolved most attacks from the communication and application levels. Because the adversary finds any loophole/vulnerability in these fields, they might exploit major attacks and cause severe damage to the financial institutes and entities involved in the protocol.
5. The security analysis of the proposed scheme is demonstrated through the RoR and broadly accepted model-checking tools of Scyther.
6. Finally, we present a comparative summary based on security features, communication cost, and computation overhead of existing methods (Das et al., 2017), specifying that the proposed framework is secure and efficient for macro micropayments using wearable devices.

Table 2 lists the acronyms used in this paper. These acronyms are used in the text, figures, and tables.

Table 2
List of acronyms used in this paper.

Name of the acronym	Full-form
ADT	Android Development Tools
AES	Advanced Encryption Standard
BLE	Bluetooth Low Energy
C2C	Customer-to-Customer
C2M	Customer-to-Merchant
COGS	Costs of Goods Sold
ECC	Elliptic Curve Cryptography
ECDLP	Elliptic Curve Discrete Logarithm Problem
ECDSA	Elliptic Curve Digital Signature Algorithm
ECIES	Elliptic Curve Integrated Encryption Scheme
EED	End-to-End Delay
FPS	Fingerprint Scanner
FinTech	Financial Technology
IoT	Internet of Things
MitM	Man-in-the-Middle
MWA	Mobile Wear App
NFC	Near-Field Communication
OPEX	OPerating EXpenses
OTA	Over-The-Air
P2P	Peer-to-Peer
PKC	Public Key Cryptography
PUF	Physical Unclonable Function
RFID	Radio Frequency Identification
RoR	Real-or-Random oracle model
RSA	Ronald Rivest Shamir Adleman
SE	Secure Element
SHA-1	Secure Hash Algorithm-1
SPDL	Software Protocol Descripriction
UPI	Unified Payments Interface
WPKI	Wireless Public Key Infrastructure

1.2. Organization of the paper

The paper is organized as follows: In Section 2, provided the existing works. Section 3 deals with the system models of network and threat model. Mathematical preliminaries of ECC are discussed in Section 4. Section 5 explains our proposed framework's architecture and protocol steps. This section includes the definition of entities involved and how messages communication between these entities using steps are shown. Section 6, the proposed protocol is formally verified with RoR and the model-checking tool of Scyther. Section 7 deals with the computation and communication costs of the proposed framework followed by an implementation using the network simulator of NS3, Section 8. Section 9 provides the conclusions and future work. Finally, the proposed protocol formal verification with Scyther written in Software Protocol Descripriction (SPDL) is available in A, and informal verification of the proposed framework using the Drozer tool [2] is explained in B.

2. Related work

Designing lightweight authentication protocols for wearable devices is a critical and challenging task. However, some of the related schemes are discussed in this section. Wong et al. (2006) proposed a secure framework for resource-constrained devices; it provides user authentication based on XOR operation and with some hash functions. The mechanism offers a symmetric process that is permitted for numerous attacks, such as Man-in-the-Middle, forgery, and replay attacks. Das et al. (2012) developed a lightweight smart-card-based user authentication. The framework also supports many security features on password protection, key agreement, dynamic node addition phase, and mutual authentication. However, the scheme is used only with certain hash computation functions and XOR operations, leading to design flaws and being susceptible to many attacks. Patel et al. (2015) proposed a Bluetooth pairing device framework. The mechanism provides digital payments using a smartphone application under an Android device called PAYTOOTH. Bluetooth is an open wireless and widely available protocol for transmitting and receiving data over short distances between two mobile phones. In the Paytooth system, Bluetooth is a connection medium between two devices. A paytooth system where people can make payments via their phone through secure Bluetooth without ever needing to carry cash with them can be developed (Patel et al., 2015). However, the authors must represent the developed app's cryptography techniques, security features, and formal verification. Hence, the mechanism leads to failing security against attacks for these reasons. Liu et al. (2016) implemented an asymmetric three-party-based authentication security framework for wearable devices. The developed scheme communicates only with the three parties, such as customers, wearable devices, and smartphones. The method provides many features for pairing the smartphone with wearable devices using Bluetooth, such as password updates, wearable device replacement, and user node addition. However, the authors must show the session keys used for transaction purposes. More information about the schemes, like benefits and disadvantages, is available in Table 3. Liu et al. (2016) developed a mechanism of "Yoking-Proof-based Authentication Protocol (YPAP)" for cloud-assisted wearable devices. A few secure HMAC code and XOR logic gates operations support the developed framework. It uses a "Physical Unclonable Function (PUF)" and lightweight cryptography operations to achieve mutual authentication between smartphones and wearable devices. In addition, the yoking proofs for simultaneous verification at the cloud server. Apart from the above, the authors used a formal security analysis of Rubins' logic for the correctness of the protocol. Generally, PUFs provide primitives for implementation, primarily for tamper detection, encryption and device fingerprinting. One widespread application is replacing Non-volatile Memory (NVM) as key storage in embedded

[2] https://labs.f-secure.com/tools/drozer/.

Table 3
The existing approaches in wearable devices: A comparative summary.

Related work	Technology	Device pairing	Method of payment	Name of the cryptography algorithm & concept used	Advantages	Limitations
Patel et al. (2015)	Paytooth	Bluetooth	Proximity payments	NA	i. Cashless/cardless payment using a point of sale machine ii. Applicable for remote areas	i. No discussion of security mechanisms used ii. No cryptographic algorithms are used Formal security analysis proofs are missed
Liu et al. (2016)	QR code-based	Bluetooth	Proximity payments	i. Asymmetric ii. Symmetric iii. One-way hash function	i. Multi-users condition ii. Efficient revocation iii. Unforgeability, anonymity and traceability	i. There is no evidence for session keys used during the transaction. ii. Only resist known types of attacks iii. The entire app depends on third-party
Liu et al. (2016)	Yoking-proof-based	RFID	Remote & Proximity payments	i. Symmetric encryption Ex. AES ii. HMAC	i. Used for cloud assisted wearables ii. Connected two wearable devices in one session iii. Using physical unclonable function iv.lightweight cryptographic operators	i. High communication cost ii. Applicable for only RFID applications iii. The method used only hash functions and XOR operations
Das et al. (2017)	Biometric-based	Bluetooth	Proximity payments	i. AES ii. Diffie-Hellman iii. SHA-1	i. Provides Biometric authentication ii. The scheme is lightweight, because only bitwise XOR and hash function iii. Applicable for resource-constrained wearable devices	i. Requires high computation and communication overheads ii. Real-time implementation difficult
Wu et al. (2017)	Cloud-assisted	Bluetooth	Proximity payments	i. Symmetric ii. Asymmetric encryption/ decryption iii. Hash functions	i. The mechanism secure against various attacks ii. The scheme achieves mutual authentication and provides device and user anonymity	i. The scheme requires more communication overhead between the cloud server and mobile devices. ii. Difficult to implement in wearable payments iii. Time cost between entities are more compared with other existing works
Gupta et al. (2019)	IoT-based	NA	Remote payments	i. Simple XOR ii. One-way hash function	i.The scheme provides mutual authentication between the gateway, mobile terminal and the wearable sensing device ii. The framework provides formal verification using AVISPA	i. No session key updates between the parties or entities ii. High computation overhead
Kumar and Grover (2019)	Authentication-based	NA	Remote payments	i. ECC ii. Hash functions	i. The method uses ECC for providing secure authentcation. ii. The scheme analyzes security features with advanced tools of Proverif and random oracle models	i. No evidence of providing secure communication between servers, wearable devices, and application providers ii. Not implemented the scheme in wearable devices payments.
Magdum et al. (2021)	EMV-Contactless	NFC	Proximity payments	NA	i. The scheme is developed on two combined technologies: a fingerprint sensor and the other one near field communication technology. ii. The ring is designed with the three components of hardware; first, it is a cryptoprocessor chip, the second is an NFC tag and lastly biometric sensor	i. More hardware is needed ii. No evidence of formal security verification payments iii. No evidence of protocol applicable in real-time
Bojjagani et al. (2022)	IoT-based	NFC	Proximity payments	i. ECC ii. AES	i. The method effectively and efficiently used for micro-payments ii. The developed framework best fits IoT-based wearables with banks in real-time payments. iii. The scheme provides security analysis with advanced simulation tools.	i. The work is applicable for only micro-payments ii.The scheme provides only two-way authentication iii. The scheme doesn't use biometric authentication
Proposed work	Biometric & IoT-based	NFC	Remote & Proximity payments	i. ECDSA & ECIES ii. AES	i. The proposed method used for various kind of payments such as mini, micro and macro payments ii. The scheme provides three-way authentication iii, The scheme used biometric authentication ii. The developed framework best fits IoT-based wearables with banks in real-time payments. iii. The scheme provides security analysis with RoR, Scyther, and Informal verification. iv. The proposed scheme addresses the maximum number of attacks exploited by the adversary.	i. The work does not apply to Blockchain's place in wearables.

devices like smart cards and secure microcontrollers. Moreover, the improvement to overall system security provided by PUFs is still the subject of much debate (Helfmeier et al., 2014). A physical unclonable function (PUF) is a device these are potentially used in a variety of applications, from anti-counterfeiting, identification, authentication, and key generation to advanced protocols such as oblivious transfer, key exchange, key renovation, and virtual proof of reality (Gao et al., 2020). In 2017, Das et al. (2017) developed and designed a protocol for lightweight cryptography operations for wearable devices. The scheme requires high computation and communication overheads, which is challenging to implement in real-time scenarios. The framework does not consider cloud servers and mobile phone communication lightweight features. Wu et al. (2017) developed a framework for wearable devices assisted by the cloud. It is a lightweight authentication mechanism. The developed approach is resilient to attacks such as inapplicability, desynchronization, etc. However, the scheme needs more communication overheads as compared with existing methods. Gupta et al. (2019) used a lightweight anonymous user authentication and key establishment scheme for wearable devices. The framework uses simple XOR and one-way hash computation functions. It provides a formal security analysis with BAN logic and AVISPA. The technique is developed but has yet to be implemented in real-time scenarios of IoT devices. Kumar and Grover (2019) developed an authentication protocol for wearable devices using Elliptic Curve Cryptography (ECC). This mechanism provides a formal verification but no evidence of real-time implementation scenarios. Also, it fails to show secure communication between servers and wearable devices with an application provider between various entities. Magdum et al. (2021) proposed a new contactless transaction scheme for wearable devices with a biometric fingerprint feature; for this protocol, the authors showed steps for the flow of execution but no evidence of real-time implementation of wearable payments. For example, how a device communicates with the bank servers, customers, and merchants. Finally, Bojjagani et al. (2022) implemented a new scheme for IoT-based micropayments. The payment request is transmitted from the wearable device to the smartphone and then communicated to the banks. However, the method applies only to small payments; it may fail if the customer uses a credit card or net banking with wearable devices. Some of the observations found in related schemes, such as Most of the techniques use Bluetooth for device pairing, which is currently not secure for payments. The associated works fail to develop the applications in real-time implementation. Most of the schemes don't deal with payments using wearable devices. Some approaches are supported but only valid for some types of payments. Hence, this paper addresses the abovementioned problems and proposes a new scheme with an additional biometrics security feature applicable for all payments compared with Bojjagani et al. (2022).

3. System models

The network, threat and adversary models help to better understand the proposed protocol in terms of the entities involved, its execution, and security features.

3.1. Network model

For the network model, how wearable devices are connected with the various parties of application providers, banks, consumers, and merchants are shown in Fig. 2 in two channels (Seneviratne et al., 2017). The payment gateway is a trusted third-party server; the primary responsibility of this entity is to collect and send payment data from consumers to merchants. In other words, this entity identifies the amount in the customer's bank account and sends payment acceptance or rejection back to the customer.

1. The consumer orders an item using a wearable device.
2. The raised request made by the consumer is transferred to the smartphone. The smartphone establishes a secure connection with the wearable device through an NFC connection.
3. After establishing a connection, the App confirms user authentication requests by password and biometric authentication.
4. The user authentication is successful; now, the wearable device sends payment information to the consumer's smartphone via an installed wearable app.
5. The consumer initiates a payment request with any payment channels, such as net banking, credit card/debit card, or other payment gateways.
6. The consumer selects any payment channel and sends the request to the corresponding bank, such as the customer's bank, which operates and deals with consumer transactions.
7. The customer's banks validate the user data, verify the funds in their account and route the transaction to the merchant's bank via a payment gateway.
8. The payment gateway verifies the user's payment order and initiates the transaction settlement statement to the corresponding merchant bank account through a secure network (Bojjagani et al., 2022).
9. All verifications are completed, and the payment order amount is deducted from the customer's account.
10. After successful validation by the banks, the banks will send a payment order notification to the respective entities of the payment initiator and service provider.

The proposed protocol steps are neatly explained in the three phases in Section 5.

3.2. Threat model

This section discusses the security and privacy attacks in mobile payments. In Fig. 3, we have implemented a threat model that primarily focuses on cyber security fields of application-level security, communication-level security, and device-level security. Some of the points discussed following about the robust threat and adversary models:

1. We consider a robust adversary model, assuming that all entities are not fully trusted except the certificate authority. The bank and payment gateway are considered honest but curious.
2. External adversaries may eavesdrop on the communication to infer confidential information about users for payment.
3. Attackers can launch collusion attacks to falsely accuse legitimate users of double-spending.
4. An eavesdropper may attack the system without making proper payments or impersonating other legitimate users while purchasing an item online, thereby using their account.
5. In client–server communication, the customer's confidential information may be stored and forwarded to various external entities or third parties, which can face multiple system attacks. For example, the attacker can capture the nodes and analyse the network traffic by obtaining the access point identifier or through network vulnerabilities. They might exploit two types of system attacks: one is active, and the other one is passive. An active attacker can corrupt or modify the existing data, jam and even terminate access to the device. The attacker compromises the customer's sensitive data and threatens the pay-

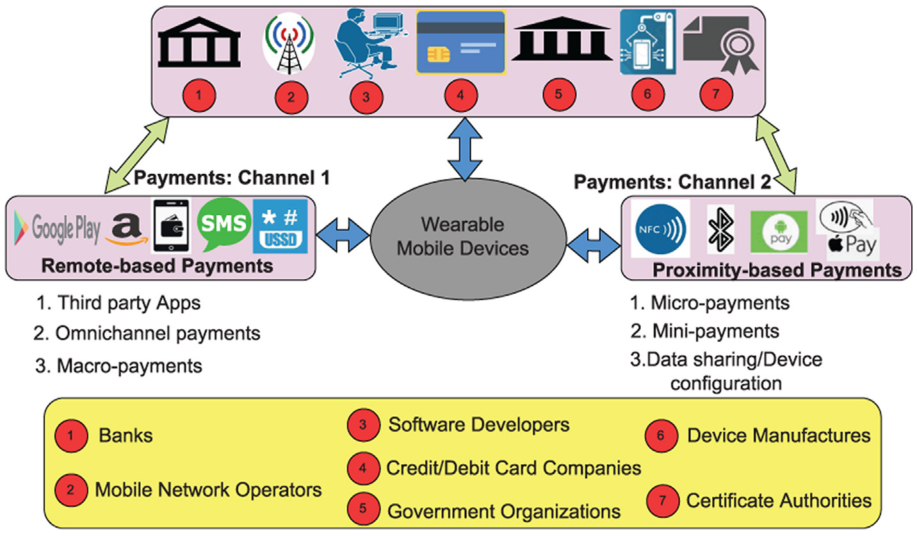


Fig. 2. Architecture for wearable devices in FinTech.

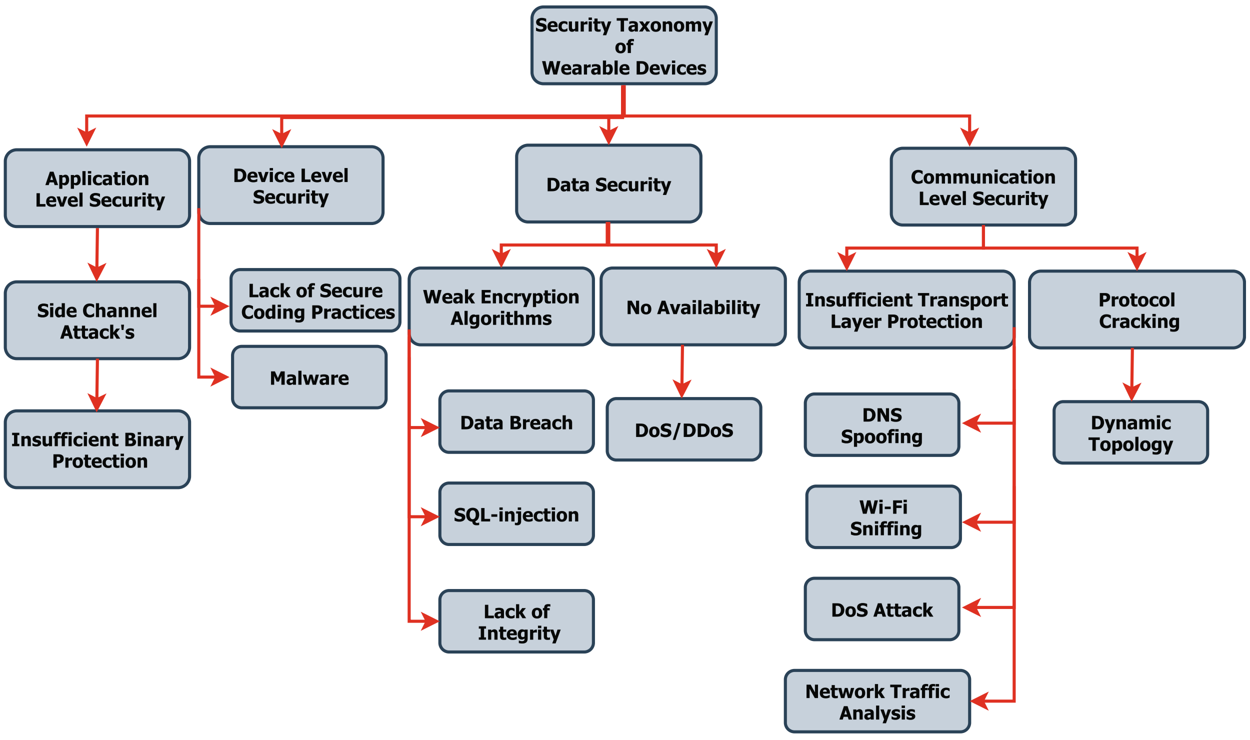


Fig. 3. A taxonomy of security risks involved in FinTech: Threat model.

ment devices' safety and integrity. On the other hand, a passive attacker can observe and capture a message from the network to compromise customer privacy data.

6. Issues in data communication: The sensitive data is transmitted and processed between various entities involved during the transmission, which creates security and privacy issues (Segura Anaya et al., 2018).
7. Issues in wearable & IoT devices: The consequences faced between wearable and IoT devices and their users.
8. Issues in stakeholders: The entities involved in this whole process, in other words, the stakeholders that are being considered in the framework (For example. Banks, payment gateway's, and consumers).
9. Issues in security tools: The user's data is protected using advanced cryptography tools (Gao et al., 2015).

In this paper, we addressed and resolved most attacks from the communication and application levels. Because the adversary finds any loophole/vulnerability in these fields, they might exploit major attacks and cause severe damage to financial institutes and entities involved in the protocol. We are using the lightweight cryptography techniques of ECDSA for signature generation and verification. Similarly, we have used ECIES as asymmetric for encryption and decryption; for symmetric, we have used an AES algorithm. Using formal and informal security analysis, our proposed model tested and verified the attacks mentioned in Fig. 3.

4. Mathematical preliminaries of ECC

This section deals mainly with the essential mathematical preliminary computations of Elliptic Curve Cryptography, ECDSA: signature generation, verification, and Elliptic Curve Discrete Logarithm Problem (ECDLP).

4.1. Functions of ECC

- Key pair generation of ECC: Over a finite field F_p select a set of elliptic curve points E such that the number of points in $E(F_p)$ is divisible by a large prime n. And it is defined by the equation $y^2 \text{mod } p = (x^3 + ax + b) mod\ p\ with\ a, b \in F_p$ and $\left(4a^3 + 27b^2\right)$mod p = 0. Now select a base point P(n) such that P $\in (F_q)$. Select a unique and unpredictable integer d, in the interval [1, n-1]. Then, compute Q = dP, where private key is d and public key is Q (Koblitz, 1987).
- Multiplication consists of point addition and point doubling operations. In other words, it involves repeated point addition and doubling. The scalar multiplication method for any positive integer k is represented by the binary form: $k = \sum_{j=0}^{l-1} K_j 2^j, K_j \in \{0, 1\}$
  For example if k = 519 = $(1000000111)_2$, it will require (W-1) = 4–1 = 3 point additions and l - 1 = 10–1 = 9 point doubling operations.
- Signature generation using ECDSA: Let the user generate M's signature message with his A's private key. Now user selects an integer k which is unique and unpredictable for the interval of [1, n-1]. Compute an operation of kP = (x1; y1), where x1 is an integer, Compute r = x_1 mod n; Compute h = H(M), where H is the "Secure Hash Algorithm (SHA-1)". Now compute s = (k^{-1} h + dr) mod n; The signature of A for message M is the integer pair (r,s) (Hankerson et al., 2006).
- Signature verification using ECDSA: Once the signature message is transmitted, the receiver B, receives it and verifies the authenticity of A's signature (r,s) over M by with the A's public key (E, P, n, Q). The receiver verifies the values of (r,s) within the interval of [1, n-1]. Now Compute w = s^{-1} mod p. Compute h = H (M), where H is the same secure hash algorithm used by A. Compute u_1 = hw mod n. Then compute u_2 = rw mod n. And now compute u_1P + u_2Q = (x_0, y_0). Finally, Compute v = x_0 mod n (Hankerson et al., 2006).

4.2. Functions of ECDSA

1. The "Elliptic Curve Digital Signature Algorithm (ECDSA)" technique has been widely used in many domain areas, including Internet of Things (IoT) devices, wireless networks, Grid computing, and Internet of Vehicles (IoV), etc., for the implementation of secure protocols (Madhusudhan and Shashidhara, 2020).

2. The same security level is achieved with the ECC compared to Ronald Rivest Shamir Adleman (RSA) but with fewer bits. The security levels were tested and verified with real-time scenarios (Mahto et al., 2016).

3. The performance computations on key generation, sign generation and verification using ECDSA are less than the RSA algorithm (Levy, 2015).

With the above functionalities of ECDSA and RSA asymmetric algorithms, we can conclude that our proposed protocol is best suited for mobile applications. Because the key length is occupied by fewer bits and performance, computation costs are also meagre. Security is a significant concern for payment applications and maintaining Public Key Cryptography (PKC) schemes in a constrained environment.

4.3. Elliptic Curve Discrete Logarithm Problem (ECDLP)

The ECDLP (Hankerson and Menezes, 2011) is the fundamental assumption for elliptic curve-based protocols. Based on the infeasibility of computing, the discrete logarithms on elliptic curves are defined over finite fields. Let two points, P and Q, on an elliptic curve E over a finite field $P, Q \in E(\mathbb{F}_q)$ with an order b, if Q = r.P for some $r \in Z * b$, it is computational hard to find r. It relates the problem to elliptic curve cryptography and pairing-based cryptography.

5. Proposed protocol

This section provides our proposed framework for the macro payment protocol using a wearable device. The notations used for this proposed protocol scheme are listed in Table 5.

5.1. Initial assumptions and conditions

1. The consumer raised the request for payment using a wearable device. The wearable device and smartphone should be connected or paired with the NFC technology.
2. The registered users (merchants and customers) should maintain valid accounts with their respective banks for transactions with mobile/digital payments.
3. Every consumer uses their signed certificates from a trusted third party, i.e., a certification authority (CA), to perform macro payments.
4. Every consumer has their own "Wireless Public Key Infrastructure (WPKI)" certificate, which is stored in a Secure Element (SE) in the wearable device.
5. All parties involved in the proposed protocol possess their public keys and digital certificates.
6. The banks provide and personalise mobile payment applications Over-The-Air (OTA).

5.2. Payment gateway's used in our protocol

Some of the Payment gateway used in our protocol: Amazon Pay and Google Pay: Many companies have tie-ups with Amazon Payments. Worldwide, a total of 200 million Amazon Prime members [3], making it a successful payment amazon payment and a desirable payment gateway option. Amazon Pay is also reasonably customizable, with several plugins available, including some for use with BigCommerce.

In our proposed protocol, we developed Amazon payments with customer Unified Payments Interface (UPI) with the help of API. UPI APIs are a set of APIs provided by the National Payments Corporation of India (NPCI), which allow the transfer of amounts between merchant and customer account holders. Many payment apps, including public and private sector banks, Amazon Pay, Google Pay, Phone Pay, and Paytm, use the UPI platform to transfer money. Some of the UPI APIs used in our proposed framework are shown in Table 4. The security recommendations checklist or indicative guidelines are defined within each organization for development, deployment, and opening APIs for partners. These security policies are applied to the platform, like activities where to perform and corresponding actions.

[3] https://www.bigcommerce.com/articles/omnichannel-retail/how-to-sell-on-amazon-for-beginners/.

Table 4
API from standard entities.

S. No.	API Name	Description
1	Virtual Payment Address (VPA)	This API finds whether a particular user's UPI-ID is valid. It also verifies customer information matches the registered account.
2	Merchant Collect	The merchant uses this API and initiates the money request and sends to a customer in the form of a UPI collect notification.
3	Check Transaction Status	This API is used to check the status of a transaction, whether it was successful or with the response code. It works based on fetching status against the Transaction ID (TID).
4	Merchant Cash Back	A payment API will send money from a merchant account to a customer via UPI. It can be used for cash-backs, refunds, disbursements etc.

Table 5
List of notations used in the proposed protocol.

Sl. No.	Name of the parameter	Notation description
1	$\mathscr{C}, \mathscr{M}, \mathscr{WD}$	Consumer, Merchant/Service provider, Wearable Device
2	$\mathscr{CB}, \mathscr{MB}, \mathscr{PG}$	Consumer's Bank, Merchant's Bank, Payment Gateway
3	$\mathscr{A}$	Adversary
4	ID_X	Identity of X; Where $X\epsilon\{C, WD, CB, MB, PG, M\}$
5	N_X	Nonce generated by any participant of X.
6	$Cert_X$	The certification of entity X; which is signed by the private key of a certification authority
7	TID_X	The entity X transaction ID
8	SK_X	The entity X secret key
9	$SIGN_X^Y(M)$	Represents the signed message, which is generated by an entity X, and to be verified by entity Y.
10	K_X^+	Public key of X where $X\epsilon\{C, CB, MB, M, WD\}$
11	K_X^-	Private key of X where $X\epsilon\{C, CB, MB, M, WD\}$
12	PI	Payment information
13	POI_X	Payment order information used by the entity X, where $X\epsilon\{C, WD\}$
14	$hash(K, M)$	Hashed message with a key
15	AMT_X	Amount of X
16	SK_{XY}	Session key shared between the entites of X and Y
17	TS_X	Time stamp generated by the entity X
18	$(PI)_{K_{X,Y}}$	the entities X and Y use the payment information, and it is encrypted with the symmetric key
19	$P\|\|Q$	Data concatenation of P and Q
20	M	Message
21	AN_X	Entity X account numbers, where $X\epsilon\{C, M\}$
22	MOB_X	Consumer's or Merchant's mobile number
23	$NPIN_C$	Non-repudiation PIN used by the consumer
24	BIO_i'	A fresh biometric captured template
25	BIO_i	Stored biometric captured template
26	ACC_NO	Consumers' account number
27	TC_i, TC_j_1	Temporal credentials
28	MAC	Message authentication code

5.3. Detailed protocol

This section provides our proposed framework for the macro payment protocol using a wearable device. Our solution consists of three phases: (1) Registration phase, (2). Login phase (3). Transaction initialization phase. In addition, we provided the dynamic wearable device, which is used to add a new wearable or replace the devices. The following are brief explanations of the three phases step-wise.

5.3.1. Registration phase:

Initially, the registered user/consumer $\mathscr{C}$ should establish secure pairing using Near-Field communication (NFC) between the devices such as wearable $\mathscr{WD}$ and smartphone $\mathscr{SP}$; The whole process is called device pairing. Some of the points for novelty in the secure NFC-based lightweight payment system are mentioned as follows:

- NFC technology enables integrating services from a wide range of applications into one single smartphone (Coskun et al., 2013).
- According to the following works (Coskun et al., 2015; Tabet and Ayu, 2016; Haselsteiner and Breitfuß, 2006), the NFC works effectively against MitM, data leakage, eavesdropping, network traffic analysis, and other types of attacks. Most surveys and studies on NFC security agreed that developing a secure channel is the most efficient security mechanism for securing data in an NFC-based system.
- Our proposed framework is implemented in the business-centric architecture of payment systems, so the keys are used between the entities and shared between the devices with pairing using NFC. But as long as security is a concern, the entities should register with the trusted authority, and the framework achieves mutual authentication. It best fits resource-constrained devices such as the security elements embedded into NFC-enabled devices.

Step 1: For wearable device (WD_i) Before the registration phase, the NFC-enabled wearable device $(\mathscr{WD})$ is paired with the consumer's $(\mathscr{C})$ smartphone. After devices are paired, the $(\mathscr{WD})$ sends a constructed message of M1, which is encrypted with his key of sk_{cb}.

- The constructed message m1 has the details of $\{WD_{ID}, N_{WD}\}$.
- The message M1
  $\{hash(M1)\}SK_{CB}$, which is a static signature of the hash. The whole message is encrypted with their session key (SK_{CB}) for guarantees the integrity of M1.
- $Cert_{CB}$ is the customer's bank certification signed by a secret session key of the CA Bojjagani and Sastry (2019). Used to obtain the public key of customer's bank (K_C^+) to verify hashed message of the $\{hash(M1)\}SK_{CB}$.
- $Cert_{WD}$ is the certificate of $\mathscr{WD}$ signed by SK_{CB}, and it will be used by the $\mathscr{C}$ to obtain the public key of PK_{WD}^+.

$\mathscr{WD} \rightarrow \mathscr{C}$: $M1\{hash(M1)\}SK_{CB}, Cert_{CB}, Cert_{WD}$.

Step 2: For Consumer $(\mathscr{C}_i)$ The message M1 received from the wearable device WD_i, and it is verified by the wearable device user by:

IF (Verfication (M1) == TRUE). Successfully verified the consumer, confidentiality, authentication and Integrity of M1 Goto - Step 2

Else

Reject M1 because it is compromised.

Exit.

If it is successful, then the C construct the new message of M2. The consumer uses the secret key of SK_{CB}, provided by the customer's bank and sent to the wearable device.

- The consumer constructed message M2 has the details of $\{C_{ID}, N_C, MOB_C\}$.
- The consumer certificate $Cert_C$ is signed by the SK_{CB}. The wearable device $\mathscr{WD}$ will use this to get the public key of PK_C^+.
- The static signature of the hash of M2
  $\{hash(M2)\}SK_{CB}$. It is generated by the private key (Sk_{CB}) and guarantees the integrity of M1.

$\mathscr{C} \rightarrow \mathscr{WD}$: $M2\{hash(M2)\}SK_{CB}, Cert_{CB}, Cert_C$.

Now, the consumer and wearable devices get the public keys for exchanging secure messages and performing transactions with the entities. To determine the personal identities of the user and wearable devices, use the nonce messages. Similarly, the banks also get

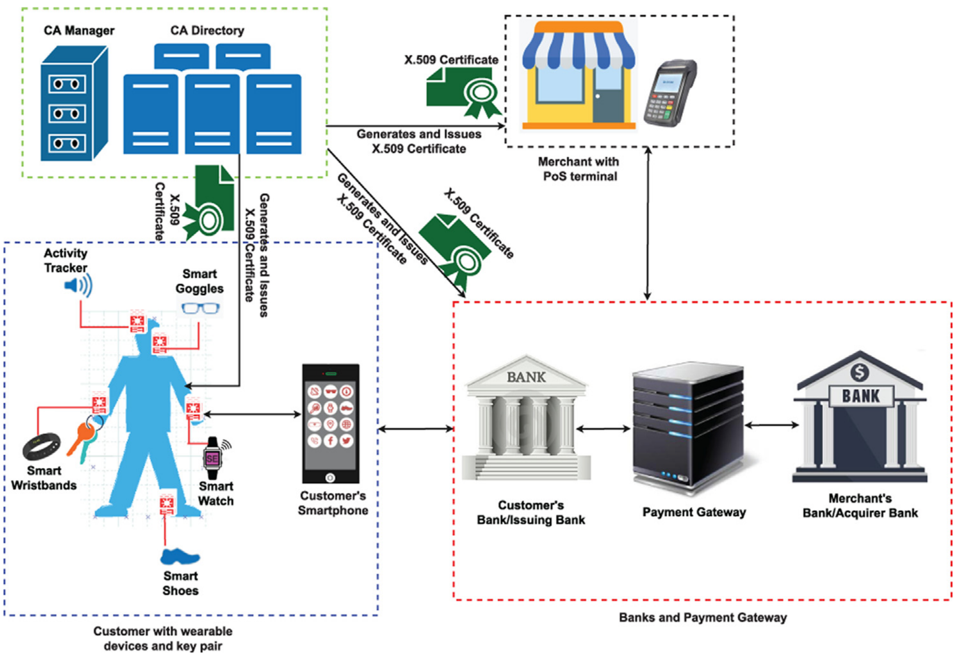

CA Manager
CA Directory
X.509 Certificate
Generates and Issues X.509 Certificate
Merchant with PoS terminal
X.509 Certificate
Generates and Issues X.509 Certificate
X.509 Certificate
Generates and Issues X.509 Certificate
Activity Tracker
Smart Goggles
Smart Wristbands
Smart Watch
Smart Shoes
Customer's Smartphone
Customer with wearable devices and key pair
BANK
BANK
Customer's Bank/Issuing Bank
Payment Gateway
Merchant's Bank/Acquirer Bank
Banks and Payment Gateway

Fig. 4. Registration Phase.

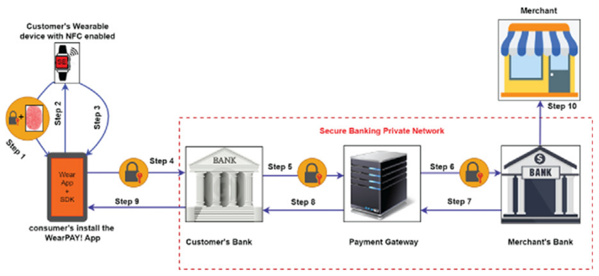


Fig. 5. A sequence of steps between the entities using the wearable device.

the public and private keys from the trusted third parties of certification authorities (CA); this paper needs to mention those details. The registration phase is shown in Fig. 4. In the diagram, the bidirectional arrows indicate the communication between entities. The single side arrow suggests that the participating entities get the X.509 certificates from trusted authorities such as the Certification Authority (CA).

5.3.2. Login phase

Once the registration phase is over, the entities of $\mathscr{WD}_i, \mathscr{C}$ get the keys to encrypt and decrypt messages from the customer's bank. The consumer initiates a wearable payment and sends user credentials and biometric personalization information using the Fingerprint Scanner (FPS) or camera. The mobile, wearable application verifies the user's authentication; these verification details are available in Algorithm 1. A secure end-to-end macro payment protocol for mobile payments using an IoT-based wearable device environment (as shown in Fig. 5) is proposed, suitable for wearable devices.

Step 1: For Wearable Device $\mathscr{WD}_i$

In this login phase, the owner of the Wearable Device user first inputs the $\langle ID_{WD}||N_{WD}||TS_{WD}||BIO_C||NPW\rangle$, and selects a public key of customer K_C^+ for encryption purposes. Moreover, the consumer C_i imprints biometric information BIO_C' such as fingerprint using an FPS and sends the captured Fingerprint Template (FPT) to the Mobile Wear App (MWA). The FPT is stored in the mobile wear application.

$\mathscr{WD} \rightarrow \mathscr{C}$: $\left\{M1,\ SIGN_C^{WD}\ (M1)\right\}K_C^+$.

Step 2: For Consumer $(\mathscr{C})$

After receiving the $(M1)$ from the $\mathscr{WD}_i$, the customer decrypts $M1$ using their secret key. Now, the consumer validates the $M1$ by the following. **if**(Verification $(SIGN_C^{WD}\ (M1)) == 1)$ **then**

```
    "Authenticity and Integrity of M1 is not compromised"
    Now C constructs $M2$ and send to the $\mathcal{WD}_i$:
    $C \rightarrow \mathcal{WD}: \left\{M2,\ SIGN_{WD}^{C}\ (M2)\right\} K_{WD}^{+}$
    Where M2= $\langle BIO_C||ID_{WD}||N_{WD}||TS_{WD}||ID_C||N_C||TS_C\rangle$
else
    Reject $\mathcal{WD}_i$
    "Authenticity and Integrity of M1 is compromised"
```

Step 3: For Wearable Device (WD_i) After receiving the $M2$ from the $\mathscr{C}$, the $\mathscr{WD}$ decrypts $M2$ and recovers M2. $\mathscr{WD}$. The wearable device compares the consumers' nonce, timestamp values and message2 $M2$ authentication and integrity. If all verifications are successful, the device initiates the payment order and transfers it to the mobile wear application. **if** (Verification ($SIGN_{WD}^{C}$ ($M2$)) == 1) **then**

```
    "Authenticity and Integrity of M2 is not compromised"
    The device $\mathcal{WD}_i$ constructs a new message which is about the payment order information (POI) called M3 and sends it to the C:
    $\mathcal{WD} \rightarrow C : \left\{ M3,\ SIGN_{C}^{WD}\ (M3) \right\} K_C^{+}$
    Where $M3$ is:
    $\langle ID_{WD} \| ID_C \| AMT_{WD} \| POI_{WD} \| N_{WD} \| TID_{WD} \| TS_{WD} \rangle$
else
    Reject $C$
    "Authenticity and Integrity of M2 is compromised"
```

5.3.3. Transaction initialization phase

After receiving the payment order information from the wearable device, the consumer placed a new order and transferred it to the issuing bank through the secure banking network.

Step 4: For Customer $\mathscr{C}$ The $\mathscr{C}$ receiving message from the $\mathscr{WD}_i$. The $\mathscr{C}$ decrypts and recovers the message3. And also, the $\mathscr{C}$ verifies the message authentication and integrity. **if** (Verification ($SIGN_{C}^{WD}$ ($M3$)) == 1) **then**

```
if (Verification ($SIGN_C^{WD}$ (M3)) == 1 ) then
    "Authenticity and Integrity of M3 is not compromised"
    If all verifications are successful, the C constructs a new message and sends it to their bank with $POI_C$, including their bank account details of PIN and registered mobile number for payments, and sends it to the customers' bank.
    $C \rightarrow \mathcal{CB} : \left\{ M4,\ SIGN_{IB}^{C}\ (M4) \right\} K_{CB}^{+}$
    Where M4= $\left\langle ID_C \| N_C \| TS_C \| AMT_C \| TID_C \| (PI)_{K_{C,CB}} \| N_{WD} \right\rangle$,
    and $\langle POI_{WD} \| ID_{WD} \| TID_{WD} \| TS_{WD} \rangle$
    where $PI = \langle MOB_C \| N_C \| AMT_C \| POI_C \| ACC\_NO_C \| NPIN \rangle$
else
    Reject $\mathcal{WD}_i$
    "Authenticity and Integrity of M3 is compromised"
```

The consumer bank (CB) verifies users' account details and wearable device data. This step is necessary for authorizing the consumer details; the verification details of the consumer's details are shown in Algorithm 2.

Step 5: For Consumer Bank (***CB***) The message $M4$ received from the Consumer; the bank (CB) verified the authentication and integrity of $M4$.

if (Verification ($SIGN_{CB}^{C}$ ($M4$)) == 1) **then**

```
    "Authenticity and Integrity of M4 is not compromised"
    The consumer bank (CB) constructs $M5$ and send to the payment gateway (PG)
    $\mathcal{CB} \rightarrow \mathcal{PG} : \{ MAC0,\ M5 \} SK_{CP}$
    Where $MAC0 = hash(SK_{CP}, M5)$.
    Where $M5$=
    $\langle N_{CB} \| TS_{CB} \| ID_{WD} \| ID_C \| POI_C \| AMT_C \| TID_C \| SK_{CP} \rangle$
else
    Reject $C$
    "Authenticity and Integrity of M4 is compromised"
```

Step 6: For Payment Gateway (***PG***) The message $M5$ was received from the customer's bank; the PG verified the authentication and integrity of $M5$. If all verifications are successful, the (***PG***) will construct a message called ***M6*** and send it to the merchant's bank.

if (Verification ($MAC0$ ($M5$)) == 1) **then**

```
    "Authenticity and Integrity of M5 is not compromised"
    Now **(PG)** constructs **M6** and send to the AB:
    $\boldsymbol{\mathcal{PG} \rightarrow \mathcal{MB} : \{ MAC1,\ M6 \} SK_{PM}}$
    Where $\boldsymbol{MAC1 = hash(SK_{PM}, M6)}$
    Where M6=
    $\boldsymbol{\langle ID_{WD} \| ID_C \| POI_C \| TID_C \| AMT_C \| TS_{CB} \| SK_{PM} \rangle}$
else
    Reject $\mathcal{CB}$
    "Authenticity and Integrity of M5 is compromised"
```

Step 7: For Merchants' Bank (***MB***) After receiving the $M6$ from the payment gateway, the merchant bank (MB) decrypts the message, verifies the message authentication, and validates the payment information (PI), consisting of secure data about the merchant. The merchant banks conform to the received information from the customer.

if (Verification ($MAC1$ ($M6$)) == 1) **then**

```
    "Authenticity and Integrity of M6 is not compromised"
    The merchant banks (MB) constructs **M7** and send it to the PG
    $\boldsymbol{\mathcal{MB} \rightarrow \mathcal{PG} : \{ MAC2,\ M7 \} SK_{MP}}$
    Where $\boldsymbol{MAC2 = hash(SK_{MP}, M7)}$,
    Where M7=
    $\boldsymbol{\langle ID_M \| AMT_M \| TID_M \| ID_{WD} \| ID_C \| SK_{MP} \rangle}$, and
    $\boldsymbol{\left\langle (PI)_{K_{M,MB}} \| SMS \right\rangle}$
else
    Reject $\mathcal{PG}$
    "Authenticity and Integrity of M6 is compromised"
```

Step 8: For Payment Gateway (***PG***) After receiving the $M7$ from the merchant bank (MB), the PG decrypts the message and verifies the message's authentication and integrity.

```
if (Verification (MAC2 (M7)) == 1) then
    "Authenticity and Integrity of M7 is not compromised"
    PG makes a message $M8$ and sends it to the CB:
    $\mathcal{PG} \rightarrow \mathcal{CB} : \{ MAC3,\ M8 \} SK_{PC}$
    Where $MAC3 = hash(SK_{PC}, M8)$,
    Where M8=$\langle TID \| AMT \| ID_C \| ID_{WD} \| ID_M \| SK_{PC} \| SMS \rangle$
else
    Reject $AB$
    "Authenticity and Integrity of M7 is compromised"
```

Once customer and merchant banks validate the payment order information, the successful information is sent via SMS as a public channel to the consumer and beneficiary and encrypted data using a secure medium.

Step 9: Customer Bank (***CB***) Once the Message $M8$ is received from the gateway, such as PG, the consumer bank (CB) will decrypt it using the secret session key of MessageMessage $M8$. Now, CB verifies its message integrity and authentication of the payment gateway.

```
    "Authenticity and Integrity of M8 is not compromised"
    CB constructs $M9$ and send to the $C$:
    $\mathcal{CB} \rightarrow C : \left\{ M9,\ SIGN^{C}_{CB}\ (M9) \right\} K^{+}_{C}$
    Where M9= $\langle TID, AMT, ID_C, ID_M, ID_{WD}, SMS \rangle$
    $\xrightarrow[To\ Customer:\ Via\ Secure\ Channel]{\left\{ M9,\ SIGN^{C}_{CB}\ (M9) \right\} K_C}$
    $\xrightarrow[To\ Consumer:\ Via\ Public\ Channel]{\{ SMS\ Notification \}}$
else
    Reject $PG$
```

Step 10: Merchants' Bank (***MB***) The merchant bank (MB) sent a notification message to the merchant (M). After message verification, it also builds a message of $M10$ and sends it to the merchant via a secure channel and SMS as a notification.

$\mathcal{MB} \rightarrow \mathcal{M} : \left\{ M10,\ SIGN^{M}_{MB}\ (M10) \right\} K^{+}_{M}$

Where M10 = $\{TID, AMT, ID_C, ID_{WD}, ID_M, SMS\}$.

Algorithm 1. The mobile wearable App verifies the user's authentication

Input: A customer initiates the personalization of the mobile wear app (MWA), for this user keeps their finger on the fingerprint scanner using a mobile phone. The scanner verifies the user's current biometric with the older biometric stored in the device.

Output: The Mobile wear app (MWA) verifies the user's data and confirms the process for payment order information.

```
if $BIO'_C == BOI_C$ then
    return True;
    //The customer authorized;
else
    //The customer unauthorized;
    if ($NPW_C == NPW_{MWA}$) then
        return True;
        //Customer authorization successful;
    else
        //Customer authorization unsuccessful;
        if ($TID_{WD}, TID_C = Successful$) then
            return True;
            //Valid Transaction_ID;
            if (($N_{WD}, N_C$) = $Authorized$) then
                return True;
                //Nonce validated;
            else
                return False;
                //Incorrect Nonce;
            end
        else
            return False;
            //Invalid Transaction_ID ;
        end
    end
end
```

Algorithm 2. Authentication verification by the consumer's bank

```
Input: {$POI_{WD}, ID_{WD}, Amount_{WD}, TS_{WD}, TID_{WD}$}
Output: Consumer's Bank authorizes the payment order information
if ($ID_{WD}$ == $ID_C$ Authorized) then
    return True;
    //WD and C authorized and proceed to payment;
else
    //WD and C unauthorized;
    if ($POI_{WD}$ == $POI_C$ & $Amount_{WD}$ == $Amt_C$) then
        return True;
        //Payment order successfully validated;
    else
        //Invalid payment order;
        if ($TID_{WD}$ == $TID_C$) then
            return True;
            //Valid Transaction_ID;
            if (($N_{WD}, N_C$) = Authorized) then
                return True;
                //Nonce validated;
            else
                return False;
                //Incorrect Nonce;
            end
        else
            return False;
            //Invalid Transaction_ID;
        end
    end
end
```

5.3.4. Dynamic WD addition

It allows the consumer to add a new wearable device and replace devices that have been misplaced or stolen. Only the primary consumer has the right to replace an existing wearable device. For deployment of a new wearable device, say WD_j^{new} in the existing network, the certificate authority does the following steps:

Step 1: The certificate authority chooses a unique identity $ID_{WD_j}^{new}$ and a unique master key $MK_{WD_j}^{new}$ for WD_j^{new}. Now the CA computes the session key $SK_{C-WD_j}^{new} = hash(K||MK_{WD_j}^{new}||BIO_{C_j}^{new})$ using its secret key K, the corresponding temporal secret credentials TC with the biometrics of the user BIO.

$TC_{j_1}^{new} = hash(SK_{C-WD_j}^{new}||ID_{WD_j}^{new}||BIO_{C_j}^{new})$ for $(i = 1, 2, 3, \ldots . n_C)$.

Step 2: Finally, the TA stores the information $(ID_{WD_j}^{new}, BIO_{C_j}^{new}, TC_{j_1}^{new})$ into SN_j^{new}'s memory before its deployment.

6. Security analysis

Wang et al. (2014) observed that the formal security verification methods such as " Random/Real Oracle Model (Abdalla et al., 2005), Scyther (Cremers, 2008)" are used to determine the various attacks such as Man-in-the-Middle, Replay and other associated attacks. But, still, these methods may not identify some of the hidden vulnerabilities in the system. Hence, guaranteeing the soundness of security protocols remains an open challenge. Based on this consideration, our proposed framework must also verify the security properties to meet the designed protocol using formal and informal verification methods to ensure that the proposed framework runs in a secure environment with high probability.

In the threat model Section 3.2, we defined some of the vulnerabilities with the attacks. The security features helped to build the security analysis of our proposed protocol. These features are classified into application-level, communication-level, and device-level security.

1. Application-level: It is mainly responsible for the app developer during the development of an application. The app developer should take care of secure coding practices. During the application's design, the app leads to vulnerabilities such as side-channel attacks, lack of binary protection, SQL injection, etc.
2. Communication-level: The information is shared between the client and server during the data communication stage. When the solution transmits its data, it must traverse the mobile device's carrier network and the internet. Threat agents might exploit vulnerabilities to intercept sensitive data travelling across the wire. The following threat agents exist: An adversary performs node capture or network traffic analysis, monitored Wi-Fi, network spoofing, DNS spoofing, etc. It leads to insufficient transport layer protection.
3. Device-level: An adversary that has attained a lost/stolen mobile device or wearable device; malware or another repackaged app acting on the adversary's behalf that executes on the mobile device. Examples: The threat agents might perform data breaches, insecure data storage, etc.

6.1. Formal verification

Formal verification is a traditional method of proof of the security terms to some precisely expressed notion of functionally correct. These security protocols have been analyzed and designed heuristically. There are some reasons we need formal verification of secure design protocols.

1. An improperly designed protocol could be vulnerable to an "active" saboteur, who may impersonate another user and alter or replay the message. A protocol might be compromised in a complex way (Dolev and Yao, 1983).

2. After designing, many existing approaches need to follow the formal verification methods, causing a violation of the security features. Without applying the standard verification techniques, there is no guarantee that the protocol is accessible from dangerous attacks of MitM, replay, etc.
3. Formal methods are the only reliable way to achieve security and privacy in computer systems. By modelling computer systems and adversaries, formal techniques can prove that a network is immune to entire classes of attacks (provided the models' assumptions are satisfied (Chong et al., 2016).
4. The designed protocol fails; for example, if the key gets compromised, the adversary exploits an attack on the protocol. Formal verification methods such as Scyther and Tamarin provide the attack graph for tracing the attack.

From the above discussion, it is necessary to use formal verification methods to identify the security violations in the proposed protocol. These methods ensure the protocol is run safely even though the active attacker is present in the network. Our proposed protocol uses three formal verification approaches RoR and Scyther.

6.2. Proposed protocol verification using Scyther

The proposed framework is simulated using the Scyther tool, written in "Security Protocol Description Language (SPDL)" (Cremers, 2009; Cremers, 2006). The simulation is to verify the security features of authentication, confidentiality, and integrity. The tool defines the entities as roles in the proposed protocol, and Scyther is a tool to verify, falsify, and analyze a protocol's security features. Scyther Cremers (2009) is a mechanism for verifying the essential characteristics of security properties under the perfect cryptography hypothesis. The attacker cannot perform security attacks over the encrypted messages with the Scyther tool unless they know the decryption key. In the Dolev-Yao model (Dolev and Yao, 1983), the attacker had total control over the communication entities. The adversary can capture any messages and modify or delete any transmissions over the network if they can construct new things from his knowledge.

Claims verification: It describes how to formalize security features in Scyther using match and claim. This subsection describes how to formalize security features in Scyther using match and claim. The match can be used in two ways. One is based on equality constraints, i.e., the event match specified in the interval of (m1, m2) can be executed if m1 equals m2. Another is value assignment, similar to the "=" in C programming. Let us assume that m is variable and v is the value for a match(m, v) denotes assigning value v to variable m.

Scyther tool uses the claims to specify the security requirements. Some claims are Nisynch, Secret, Niagree, Alive, Weakagree and Commitment.

1. Nisynch (Non-injective synchronization): All the transmission processes and sessions taken in the network between the entities are to adhere to all the security specifications of the proposed protocol. All participating entities shall adhere to being synchronized in their current state. In the proposed protocol, it is expressed in a claim: (WD, Nisynch) means that the entity WD is sure that the communication partner sends all messages; on the other hand, the messages are received by another communication partner.
2. Alive: A secure authentication between the two intended parties. It targets the performance of specific activities by an intended communication partner–for example, claim (C; Alive).
3. Secret claim: It aims to build confidentiality features between the two communication parties. For example, claim (MB, Secret, PI) means MB claims that PI must be unknown to the attacker.
4. Commitment is used to avoid the impersonation attack from the adversary. It's a promise between two entities. For example, Claim (WD; Commit; C; n) means that role WD promises n to role C.
5. Niagree (Non-injective agreement on messages): Non-injective agreement with a role on a set of data items can be defined by inserting the appropriate signal claims.
6. Reachable claim: It is used to verify the specific claim is reachable; it is indicated by at least one trace pattern if it exists. This claim is also used to verify if there are existing errors within the specification of the modelled protocol (Cremers, 2006; Dalal et al., 2010).

In SPDL programming, which accepts input for the Scyther tool, the claims regarding security are added at the end of every role; these claims are necessary for the entities to determine whether the protocol passed for verification as expected and whether goals are reached shown in Table 6. The entities involved in the proposed protocol are mapped with the scyther security properties validated and verified. Table 7 shows the mapping of security properties with the scyther tool security services; from this table, we can identify how to control and fight various adversary attacks on the proposed protocol. The proposed protocol was described and simulated using the Scyther tool, written in SPDL, available in A. The Scyther tool successfully ran and executed the code; the verification of claim procedure and automatic claim verification procedure results are shown in Fig. A.10 and Fig. A.11, respectively.

6.3. Proposed protocol proof using Real-or-Random (RoR) oracle

In this section, we used one of the widely accepted and proven security formal verification methods of the Real-or-Random oracle model proposed by Abdalla et al. (2005). It helps the security protocol's designers verify the proposed framework achieves polynomial time security against an adversary ($\mathscr{A}$) to violate the security features.

6.3.1. Communication model

For the communication model, we introduced the number of protocol participants, oracles, adversaries, protocol execution, etc.

1. Protocol participants: In our proposed protocol, six entities are participating. Still, for RoR, we mainly used three entities because for Acquiring and issuing bank and payment gateway

Table 6
The security goals described to analyze the proposed protocol (Wearable) using Scyther.

1.	$claim\_M1(M, Secret, prk(MB))$
2.	$claim\_M2(M, Secret, PI)$
3.	$claim\_WD1(WD, Secret, Nwd)$
4.	$claim\_WD2(WD, Secret, POIwd)$
5.	$claim\_CB1(CB, Secret, SKpi)$
6.	$claim\_CB2(CB, Niagree)$
7.	$claim\_CB3(CB, Nisynch)$
8.	$claim\_PG1(PG, Secret, SKpi)$
9.	$claim\_PG2(PG, Secret, SKap)$
10.	$claim\_MB1(MB, Secret, POIc)$
11.	$claim\_MB2(MB, Secret, SKpa)$
12.	$claim\_MB3(MB, Niagree)$

Table 7
Mapping security properties with scyther security services.

Security Claims	Claims verification with Scyther security properties
Confidentiality	claim_MB1(MB,Secret, POI_c) claim_CB (CB,Secret,SK_pi) claim_C(C,Secret,PI)
Authentication	NiAgree, Alive
Non-Repudiation	With the Agreement between entities satisfy the properties of messages aliveness and authenticity
Availability	Prove correctness of protocol for an unbounded number of sessions
Detect adversary attacks on MitM, replay, and protocol cracking	NiSynch

collectively called server (the set of all χ_S^j), consumer χ_C^i, and device χ_D^k, either mobile or wearable with the instances of i, j, and k.

2. Protocol execution: With the help of Oracle queries, the reaction between adversary $\mathscr{A}$ and the protocol participants is possible. It models the $\mathscr{A}$ capacity or potential for a real-time attack. During the RoR execution, the $\mathscr{A}$ may generate various concurrent instances of a legitimate user.
3. Oracle partnering: The instances of χ^i and χ^j are said to be partner oracles if and only if satisfied the following conditions:
   - The combination of χ^i and χ^j instances are in the acceptance state.
   - Both χ^{t_1} and χ^{t_2} mutually authenticate and share the standard session id "sid", which is used for the transcript of all communicated messages between the oracles.
   - Both χ^i and χ^j satisfy the partner identification, so these are mutual partners of each other.
   - No instances other than χ^i and χ^j accept with the partner recognition matching to χ^{t_1} and χ^{t_2}.
4. Random oracle: The random oracle defined for h(.), the $\mathscr{A}$ have, computes the one-way cryptographic hash function h(.) against collision resolution. Let's consider the following: an adversary generates a challenge with message m_i, with a random oracle challenger computes $r_i = h(m_i)$ and stores it in the list, say L, and it initialised with zero value with a pair of (m_i, r_i).
5. Adversary $\mathscr{A}$: An $\mathscr{A}$ is an attacker who commands the public communication links or channels. In IoT-based fintech applications, an $\mathscr{A}$ with robust potential for developing secure frameworks is also helpful for producing robust protocols with the security validation for the proposed lightweight authentication protocols. The adversary can also inject their messages and capture the live messages over the communication channel for our proposed framework. Furthermore an $\mathscr{A}$ has the following execution queries:
   - $\mathscr{RQ}$ (χ^t): The reveal query, provides the current session key used $SK_{i,j}$ generated by χ^t to an $\mathscr{A}$.
   - $\mathscr{EQ}$ $(\chi_C^i, \chi_S^j, \chi_D^k)$: The execute query models passive attacks in which an adversary snoops or monitors the legitimate executions between the consumer instance χ_C^i and the bank server instance χ_S^j, which is described as an eavesdropping attack.
   - $\mathscr{SQ}$ $(\chi_C^i, \chi_S^j, m_i)$: The send query is described as an active attack over the network model. The active attacker may capture the messages (m_i), inject or alter them or create new ones, and then send them to the legitimate parties.
   - $\mathscr{TQ}$ (χ_C^i, χ_S^j): The test query is represented by the semantic security of the session key of $SK_{i,j}$ between the parties of consumer and bank servers. If no session key for the instance of χ_C is defined or $\mathscr{RQ}$ was asked of the partner, it returns an undefined null symbol $(\perp)$.
   - $\mathscr{CDQ}(X_N^t)$: This attack is modelled as a "Capture Device Query", where adversary $\mathscr{A}$ performs to access a device's secret parameters and stolen devices by running this query.
6. Semantic security in the RoR: According to the Abdalla et al. (2005) RoR requirements model, the $\mathscr{A}$ allow asking the various queries which are mentioned earlier, such as test, execute and send. The adversary cannot find the revealed oracle for that; it is better to provide as many test queries as it wants to perform in various instances. All test queries return the same value for the random bit b. That means the keys returned by the $\mathscr{TQ}$ are either all real or all random (Wazid et al., 2017). $\mathscr{A}$ can make different $\mathscr{TQ}$ to χ_C^i or χ_j^j. Overall, if an $\mathscr{A}$ get a guessed bit b', the adversary successfully chases the game if the condition b' = b is met. Let $\mathscr{SUCC}$ represent the event in which the attacker is successful. The RoR-wear-advantage Adv_ρ^{WKA} of $\mathscr{A}$ in breaking the semantic security of our proposed framework of wearable-key-agreement (WKA), say ρ for deriving the $SK_{i,j}$ between the entities of consumer C_i and the bank server S_j is given by $Adv_\rho^{WKA} = 2 * Pr[\mathscr{SUCC}] - 1$. In RoR sense, let ρ is secure if $Adv_\rho^{WKA} \leqslant \psi$, where $\psi > 0$ is a sufficiently small real number.

Theorem 1: If $\mathscr{A}$ is a polynomial time attacker running against the proposed wearable-key-agreement (WKA) protocol within a limited time t. $\mathscr{SQ}$ denotes the send queries, $\mathscr{EQ}$ represents the execute query, PD is a uniformly distributed password dictionary, and b is many bits, and Let $q_{|hash|}$ determines the range space of hash $(|hash|)$ queries against the Consumer χ_C^i, χ_S^j and Adv_ρ^{ECDLP} defines the advantage of $\mathscr{A}$ of breaking the discrete logarithm problem of $\mathscr{A}$ for our proposed framework described as the following equation.

$$Adv_\rho^{ECDLP}(\mathscr{A}) \leqslant \frac{q^2}{2^{b_{|hash|}}} + max(|\mathscr{SQ}|, (\frac{1}{PD}, \rho f_p)) + 2 * Adv_\rho^{ECDLP} + (\frac{1}{2^{b_r}}).$$

In the above equation, b_h is the size of the return value of the hash, which is generated by an attacker $\mathscr{A}$ in bits and l_r is the random nonce generated by WKA. Send query $|\mathscr{SQ}|$ is used as the PD, the finite size of the password, and ρf_p describes the Adversary probability of false positive occurrence.

Proof. The proposed framework is secured if the $Adv_\rho^{ECDLP}(\mathscr{A})$ is ignored with the RoR model. The same work is also defined in Wazid et al. (2017), and we define the two games Gm_0 to Gm_1, to prove that our proposed scheme is secure. Finally, the event we described as $\mathscr{SUCC}$ is the correct guess for the bit b of each game Gm_i with the help of test query $\mathscr{TQ}$ by an $\mathscr{A}$.

Gm_0: The game Gm_0 is modelled as a legitimate attack by the attacker $\mathscr{A}$ on our proposed framework WKA. The bit is chosen as the b of the adversary for the initial game.

$$Adv_\rho^{WKA}(\mathscr{A}) = |2 * Pr[SUCC_0 - 1]|. \tag{1}$$

Gm_1: Let an adversary $\mathscr{A}$ capture the legitimate messages between the parties of Consumer C and the bank server B with the execute query, i.e. $\mathscr{EQ}$ $(\chi_C^i, \chi_S^j, \chi_D^k)$ oracle. Before completing the round, the adversary queries the TQ oracle. The outcome of this game determines whether the session key used between the entities of consumer and bank, such as $SK_{i,j}$, is legitimate or simply a random number generated. Note that the session key is generated with the help of two entities of consumer C_C and Server PG_P; in this case, we used the payment gateway. SK= $(hash(SK_{CP}||M5))$, where $M5$=.

$\langle N_{CB}||TS_{CB}||ID_{WD}||ID_C||POI_C||AMT_C||TID_C||SK_{CP}\rangle$. To compute the SK_{CP}, the $\mathscr{A}$ should require the message (M4); moreover, they need a payment information PI, the key which is shared between the consumer and payment gateway K_CP. Apart from that, the adversary needs personal identities to compute the SK. As a result, the $\mathscr{A}$ chance to apply attacks on M4 and M5 will not support executing the MitM and eavesdropping attacks. Hence, the proposed protocol is free from attacks, robust and secure. Thus, the winning probability of $\mathscr{A}$ does not increase in Gm_1, so we formulated the equation as

$$Pr[SUCC_0] = Pr[SUCC_1]. \tag{2}$$

Gm_2: The game Gm_2 is an active attack by $\mathscr{A}$ to trick a trustworthy entity into accepting an illicit message. It does this by imitating send and hash requests. $\mathscr{A}$ can transmit and execute unlimited hash queries $(q_{|hash|})$ to produce hash collisions. Still, since every message contains the current timestamp and a random number, this is impossible in a polynomial amount of time. Therefore, the following conclusion occurs:

$$|Pr[Succ_2] - Pr[Succ_1]| \leqslant \frac{(q^2_{|hash|})}{2.|hash|} \tag{3}$$

Gm_3: In this game, $\mathscr{A}$ executes a $CDQ(X^t_N)$ query to perform a device capture attack and obtains all of the secret parameters it has been holding. $\mathscr{A}$ catches or steals a device and utilises all of its information. Secret key K_X is encrypted using the one-way hash function h (.), and secret key SK_X is not kept. Using a password dictionary attack (PDA), $\mathscr{A}$ tries to guess Kn using the formulas $a_n = ID_X \oplus h(SK_X||K_X)$ and $b_n = SK_X \oplus K_X \oplus a_n$. The one-way collision-resistant hash function makes it difficult for $\mathscr{A}$ to use a password dictionary attack and impossible to determine the SK_X of a node.

$$|Pr[Succ_3] - Pr[Succ_2]| \leqslant \frac{SQ}{2.|PD|} \tag{4}$$

$\mathscr{A}$ runs every Oracle query to circumvent our protocol's semantic security. After $TQ(X^t_N)$ query, A can only guess the bit b_i to win the game. It results in $|Pr[Succ_3]| = \frac{1}{2}$.

We obtain the following equation for the game Gm_i by applying Eqs. 1 and 2:

$$Adv^{WKA}_{\rho} = 2.|Pr[Succ_1] - Pr[Succ3]| \tag{5}$$

Using the inequality of triangles, we have

$$|Pr[Succ_1] - Pr[Succ3]| \leqslant |Pr[Succ_1] - Pr[Succ2]| + |Pr[Succ_2] - Pr[Succ3]|$$

From Eqs. (3)–(5), we get.

$$\frac{1}{2}Adv^{WKA}_{\rho} \leqslant \frac{(q^2)}{2.|hash|} + \frac{SQ}{2.|PD|}$$

We get the following equation by multiplying both sides by 2.

$$Adv^{WKA}_{\rho} \leqslant \frac{(q^2)}{|hash|} + \frac{SQ}{2.|PD|}.$$

6.4. Informal security analysis

The following propositions illustrate that our proposed framework can withstand various security attacks.

Proposition 1. The proposed protocol is secure against replay attacks.

Proof: Assume that the adversary $\mathscr{A}$ tries to intercept the message from legitimate parties and resends a captured message to the client or server victim. To impersonate during login and perform the replay message using various tools. For example $M3$ is:

$\langle ID_{WD}||ID_C||AMT_{WD}||POI_{WD}||N_{WD}||TID_{WD}||TS_{WD}\rangle$. The attacker captures many session tokens originating from a server, which need not be of a particular client, and tries to learn/predict the pattern of tokens. Once successful learning is achieved, an attacker can create any session token and perform various unauthorized operations on the server during the login, registration and transaction phases. However, replay messages are easily identified by validating the timestamps TS_{WD} and nonce/fresh N_{WD} messages, so servers easily detect and do not proceed further.

Proposition 2. The proposed protocol is secure against SQL injection attacks.

Proof: A successful SQL injection exploit can read sensitive information from the database. Sometimes, the adversary finds weak tables in the database and tries to read or modify the sensitive data. The proposed framework is developed and implemented in Android applications. It does not allow permissions (Felt et al., 2011) to inject SQL queries via input data from the client to the data application. Because we perform static analysis on the application, no content providers exist on the attack surface. Hence, from this static security analysis, we claim that the application is free from SQL injection. The framework is verified with the Drozer tool [4], and the results for this security test are shown in Appendix B.

Proposition 3 The proposed framework resists the MitM attack.

Proof: The adversary's main objective is to learn the network traffic, capture the user's sensitive information, incorporate their information, or modify the victims' data. To detect the MitM attack using an informal static security strategy, the user must install Burp's CA certificate into the Android device. A Burp proxy can generate a self-signed certificate (Fahl et al., 2012), and any application that can receive these certificates leads to a MitM attack. On the other hand, such a configuration is possible only when the receiving host does not check the authenticity of the certificate it receives. The application has not received outdated, fake, or self-signed certificates in our proposed protocol. So, this protocol is safe from the MitM and other associated attacks, such as session prediction and denial of service attacks. Our proposed protocol has achieved the maximum security features discussed in Section 3.2 from the overall security analysis. Table 8 for the security features achieved and addressed various attacks exploited by the adversary for the proposed framework using the RoR, Scyther and informal verification.

7. Comparative study

This section deals with the analysis of the proposed work in terms of computational cost, security attacks, functionalities, and communication cost. The performance analysis of the proposed work is compared with some of the existing approaches, such as Kumar and Grover (2019); Lo and Yohan (2020); Das et al. (2017); Gupta et al. (2019); Bojjagani et al. (2022); Das et al. (2017); Santosa and Budiyanto (2019); Yohan et al. (2016); Kim et al. (2019). The security attacks resolved by various relevant methods are shown in Table 9.

7.1. Comparison of security features

Cryptography functionalities of the proposed protocol are compared with various related schemes, such as Kumar and Grover (2019); Lo and Yohan (2020); Das et al. (2017); Gupta et al. (2019); Bojjagani et al. (2022); Das et al. (2017); Santosa and Budiyanto (2019); Yohan et al. (2016); Kim et al. (2019) and the

[4] https://labs.f-secure.com/tools/drozer/.

Table 8
List of the security properties achieved for the proposed protocol using RoR, Scyther, and Informal security analysis.

Sno.	Name of the security features to achieve	Using RoR	Using Scyther	Informal Verification
1	Confidentiality	✓	✓	✓
2	Authentication	✓	✓	✓
3	Prevent Man-in-the-Middle (MitM)	✓	✓	✓
4	Resist Replay	✓	✓	✓
5	Protect from Protocol-cracking	✓	✓	NA
6	Non-Repudiation	✓	✓	✓
7	Availability	✓	✓	X
8	Safe from Capture Device	✓	✓	X
9	Prevent SQL-Injection	X	X	✓
10	Communication-level Security	✓	✓	X
11	Application-level Security	✓	✓	✓
12	Network Traffic Analysis	✓	✓	X

Table 9
Cryptography attribute features.

CF	Kumar and Grover (2019)	Lo and Yohan (2020)	Das et al. (2017)	Gupta et al. (2019)	Bojjagani et al. (2022)	Das et al. (2017)	Santosa and Budiyanto (2019)	Kim et al. (2019)	PP
CAF1	✓	✓	✓	✓	✓	✓	✓	✓	✓
CAF2	NA	✓	X	NA	NA	NA	✓	✓	✓
CAF3	X	X	X	X	X	X	X	X	✓
CAF4	✓	✓	✓	✓	✓	✓	✓	✓	✓
CAF5	NA	✓	X	NA	✓	NA	NA	✓	✓
CAF6	✓	NA	✓	✓	✓	X	NA	✓	✓
CAF7	✓	X	✓	✓	✓	✓	✓	X	✓
CAF8	X	X	X	X	X	X	X	✓	✓
CAF9	X	X	X	X	X	X	X	X	✓
CAF10	X	X	X	✓	X	X	X	✓	✓
CAF11	✓	✓	X	X	✓	X	X	✓	✓
CAF12	X	✓	✓	X	X	X	X	✓	✓
CAF13	✓	X	✓	X	✓	X	X	✓	✓

CAF1: Resistance to Reply attack; CAF2: The system supports the session key in the presence of the CK-attacker model; CAF3: Secure against MitM; CAF4: Supports anonymity and identity privacy-preserving; CAF5: Resistant to password guessing attacks; CAF6: Untraceability; CAF7: Resistance to Insider attack; CAF8: DoS attack protection; CAF9: Prevents node capture attack; CAF10: Secure against impersonation attack; CAF11: Supports mutual authentication and key agreement; CAF12: Formal verification using Scyther; CAF13: Formal security analysis using RoR model; ✓: The scheme is secure or supports a feature, X: The scheme is insecure or doesn't support a feature; NA: Not Available; PP: Proposed Protocol.

results are presented in the Table 9. An earlier Section 3.2 discussed the Threat model taxonomy in the IoV environment. The threat taxonomy helps analyze the proposed security protocol and evaluate the threat model's well-known vulnerabilities. Most of these threats are discussed in RoR and formal verification using Scyther model-checking tools.

7.2. Computation costs comparison

Computation cost refers to the time the computer requires to perform a single round of computation of a cryptographic operation. The time taken by the computer varies according to the mathematical complexity of the cryptographic process involved. The proposed scheme is designed based on the hash function and ECC public key cryptography, which are lightweight operations. We have used the computation costs calculation of the well-known and familiar works of Lee et al. (2013), He et al. (2015) for the computation costs. We used symbols to denote the cryptographic operations and compare computational costs. The Ex_{or} was omitted for this calculation because the overhead for this operation is very low. T_e : modular exponentiation; T_a : ECC point addition; T_{bp} : bilinear pairing; T_{sm} : ECC scalar multiplication; T_{fe} : biometric fuzzy extractor computation time; T_{sym} : symmetric encryption/decryption; T_{mac} - message authentication code for either MAC or HMAC; T_h : hash function execution time. In Table 10, shown as the computation cost comparison with existing works proposed by Liu et al. (2016); Liu et al. (2016); Das et al. (2017); Wu et al. (2017); Bojjagani et al. (2022); Sun et al. (2008). However, the existing works of Das et al. (2017), Liu et al. (2016) have lesser computation costs. But still, they lack security and functionality features and have high communication costs.

7.3. Communication cost

Table 10 summarises the computation cost analysis of various existing schemes. During the comparison, we counted protocol execution time between the parties. Many current approaches have just developed the protocol but are not used and are designed for payments. Communication cost is the number of bits exchanged among the communication entities in a single authentication round. It is desirable to have minimum communication cost and a minimum number of messages exchanged between the parties. Because fewer messages exchanged would help in faster authentication of parties, we consider some of the identities ECC point addition and multiplication P= (P_x, P_y) with a "(160 + 160 = 320 bits)", and AES-256 used for symmetric encryption

Table 10
Performance comparison.

Scheme	Wearable Device (WD)	User/ Mobile Device	Banks (CB/AB)	Payment Gateway (PG)/Trusted Authority
Kumar and Grover (2019)	$5T_h$+$2T_{xor}$+$1T_{ae}$+$1T_{se}$ +$1T_{sd}$	$16T_h$+$12T_{xor}$+$2T_{ae}$+$1T\_ad$+ $1T\_se$+$1T\_sd$	NA	NA
Lo and Yohan (2020)	$1T_{puf}$+$4T_{xor}$+$2T_h$+$2T_v$	NA	$3T_h$+$4T_{xor}$+ $1T_{se}$+$1T_{sd}$	$1T_{puf}$+$1T_h$+$1T_{sd}$
Liu et al. (2016)	$2T_{se}$+$3T_{xor}$+$2T_h$+$1T_v$	$8T_{xor}$+$2T_{sd}$+$2T_h$+$1T_v$	NA	NA
Das et al. (2017)	$4T_h$+$1T_{xor}$+$1T_{se}$+$1T_{fe}$ +$1T_{sd}$	$13T_h$+$10T_{xor}$+ $1T_{ae}$+$1T_{fe}$+$1T_{sd}$	NA	NA
Santosa and Budiyanto (2019)	$3T_{se}$+$2T_{sd}$+$7T_h$	$3T_{se}$+$3T_{sd}$+$7T_h$	$3T_{se}$+$3T_{sd}$+$7T_h$	$3T_{se}$+$1T_{sd}$+$6T_h$
Bojjagani et al. (2022)	$2T_{ae}$+$1T_{ad}$+$2T_g$+$1T_v$	$3T_{ad}$+$1T_{ae}$+$2T_g$+$3T_v$+ $1T_{se}$	$1T_{sd}$+$1T_h$+$1T_{se}$+ $2T_{sd}$+$2T_v$+$1T_h$+ $1T_{se}$+$1T_{ae}$	$2T_h$+$2T_{sd}$+$2T_{se}$
Amin and Biswas (2016)	$5T_h + 3T_{xor}$	$12T_h + 7T_{xor}$	NA	$15T_h + 7T_{xor}$
Gope and Hwang (2016)	$3T_h + 1T_{xor}$	$14T_h + 7T_{xor}$	NA	$26T_h + 12T_{xor}$
Proposed	$1T_{ae}$+$1T_{ad}$+$1T_g$+$1T_v$	$2T_{ad}$+$1T_{ae}$+$2T_g$+$2T_v$+ $1T_{se}$	$1T_h$+$1T_{se}$+$2T_{sd}$+$2T_v$+$1T_h$+$1T_{ae}$	$2T_h$+$2T_{sd}$+$2T_{se}$

Table 11
Communication overhead analysis.

Scheme	Required Messages	Required bits
Wu et al. (2017)	5	4800
Sun et al. (2008)	3	4512
Amin and Biswas (2016)	6	4096
Gope and Hwang (2016)	4	3184
Adavoudi-Jolfaei et al. (2019)	4	3696
Liu et al. (2016)	7	2720
Liu et al. (2016)	3	2528
	4	1504
Das et al. (2017)	3	1696
Bojjagani et al. (2022)	5	1024
Proposed protocol	5	916

and decryption, hash digest (SHA-1) requires 160 bits and for timestamp needed 160 bits (Gallagher and Director, 1995). Table 11 shows the communication cost details for our scheme and the related works.

8. Implementation

This section deals with the proposed protocol framework design, network simulation, and technologies for pilot project development.

8.1. Practical implementation using NS3 simulator

This section verifies and tests the proposed protocol using the NS3 network simulator on the Ubuntu operating system with the latest version of 18.04.3 LTS to compute the network performance metrics. The simulation is done during the login, registration, and transaction initialization phases. For the simulation environment, we have created three network scenarios;

- The scenario one has the one Customer C_i/Mobile device MD_j and with five Wearable Devices WD_k
- The scenario two has the Customer C_i/Mobile device MD_j and with eight Wearable Devices WD_k
- The scenario four has the one Customer C_i/Mobile device MD_j and with ten Wearable Devices WD_k

Each network scenario uses the following messages:
$M1 = \langle ID_{WD}, N_{WD}, TS_{WD}, BIO_C, NPW \rangle$. From (WD to C).
$M2 = \langle BIO_C, ID_{WD}, N_{WD}, TS_{WD}, ID_C, N_C, TS_C \rangle$. From (C to WD) $M3$ is: $\langle ID_{WD}, ID_C, AMT_{WD}, POI_{WD}, N_{WD}, TID_{WD}, TS_{WD} \rangle$. From (WD to C).
$M4$= $\left\langle ID_C, N_C, TS_C, AMT_C, TID_C, (PI)_{K_{C,CB}}, N_{WD} \right\rangle$ From (C to CB). Messages 1 to 4 require 256 bits, 128, 256 and 128 bits, respectively. The simulation and evaluation of the proposed protocol's impact on various network performance parameters are shown in Table 12. The metrics measured regarding end-to-end delay, message loss ratio, throughput, and message delivery ratio are briefly mentioned below.

Table 12
Parameters used in network simulation.

Name of the Parameter	Desrciption
Operating system	Ubuntu 18.04.3 LTS
Simulator Tool with version	NS-3.25
Wireless Technology	IEEE 802.11
Simulation time	1700 s
Channel bandwidth	10Mbps
Number of WD_i	Scenario 1:5 wearable devices Scenario 2:8 wearable devices Scenario 4:10 wearable devices
Communication range of WD_i	50 m
Number of C_i/MT_i	1
Mobile Terminal communication range	100 m
Number of Network scenarios	1 to 3

1. End-to-End Delay (EED): It is defined as the message to reach the destination being calculated as the average time to get the goal from a specific source. It can be represented mathematically with the following expression:

EED =.

$$(T_{Msg\_recd_n} - T_{Msg\_sent_n})/Msg_p \cdot \sum_{i=1}^{Msg\_sent_n} \cdot \left(t_{nMsg}^{sign}\right) \cdot \left(t_{xMsgy}^{trans}\right) \cdot \left(t_{xMsgy}^{verify}\right)$$

The calculation of EED consists of the signing time, verifying, and transmission time. In the above expression, the parameter is the time the messages are delivered to the number of messages sent by the device n, and Msg_p is the total number of messages. t_{nMsg}^{sign} represents the message Msg signing time by device n, and the second parameter t_{xMsgy}^{trans} transmission time of Msg sent from the customer x to wearable y. The third parameter is t_{xMsgy}^{verify}, proving the time of Msg received from device x to customer y, and lastly, for buffer length of the vehicle (Buf_Len_k). The EED values of the three network scenarios are 0.21456s, 0.86985s, and 0.94451s, respectively. From these values, it is clear that the number of wearable devices is increasing, and the value of EED is also increasing. **2. Message Loss Ratio:** It is measured as the total number of messages lost per unit of time, which is defined as the message loss rate in applications from a network security perspective due to the limited buffer size and other network issues involved in message loss. Another problem is that the receiver will verify the signature when it receives the signing message. It might consume time; however, the buffer may overflow if messages arrive more than the processing rate. This paper does not count the packet loss due to network issues such as low bandwidth and wireless medium. Thus, mathematically, it can be formulated for the total number of network scenarios as MsgLoss $=1/T_d \cdot \sum_{i=1}^{Msg\_sent_n} (Msg^n_{drop}/\sum_{k=1}^{k_n} Msg^n_{arrive})$ The parameter Msg^n_{drop} indicates the messages dropped in an application buffer for device n, similarly denotes the Msg^n_{arrive}. The total number of messages received by the user k. The packet loss ratio for the three network scenarios is 0.0335, 0.00555, and 0.00216. The packet loss rate is meager in these three network scenarios.

8.2. Application demonstration

We have developed a prototype framework for macro payments with wearable devices using Android mobile phones. The list of devices with the software mentioned in our proposed development of macro payment is available in Table 13. Liu et al. (2016) described that most wearable devices run the Android operation system. So, we chose to write an app running on Android OS. The following are application development details.

- Eclipse Android Development Tools (ADT) is a plugin for the IDE for Android SDK
- Crypto libraries (OpenSSL library, along with Bouncy Castle and Crypto-JS)

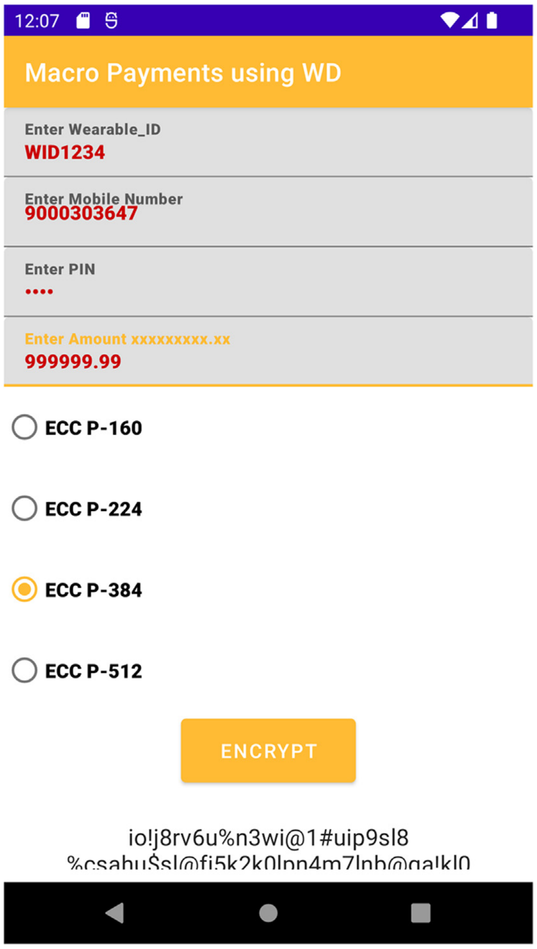
Table 13
List of software and hardware used in the proposed protocol.

Name of the Divice with software used for proposed framework	Description
Wearable Device (WD)	Name: Samsung Galaxy5 LTE Processor: Dual-core 1.18 GHz Exynos W920 RAM: 1.5 GB, Internal Storage:16 GB Operating System: Wear OS; Display: 1.2-inch 396X396 Connectivity: Near Field Communication; Sensors: Samsung BioActive
Mobile Device (MD)	Name: Nexus 6 API Level: 31; RAM 3 GB; Internal Storage: 64 GB Processor: 2.7 GHz Quad-core - Snapdragon 805 Operating system: Android 7.0 (Nougat)
Wearable Application	Package Name:com.siri. MacroPayment_WD Language: Kotlin SDK: API 24: Android 7.0 (Nougat) Connectivity: Near Field Communication Application size:7.5 MB Hashing: SHA1, Signature: ECDSA; Public Key Certificate: X.509 V3 Asymmetric encryption & Decryption: ECIES, Key Size:384 bits Symmetric encryption & decryption: AES, Key Size:256 bits
Server Description	CPU:2 * Xeon 2.4G/32G RAM Database: PHP and MySQL 5.8 HTTP server: XAMPP 8.1.17

8.2.1. Peer-to-Peer (P2P) secure payment demonstration

We demonstrate a P2P communication model with encryption and decryption of customer and merchant data for clear understanding. For this demonstration purpose, we used two mobile devices. Mobile A (Mobile No 9000303647) acts as a customer with a macro payment encryption application embedded with the public keys of the Bank. We have used 4 EC for examination purposes, one with P-160, P-224, P-384 and P-512. Mobile B (Mobile No 8639923497) is a merchant installed with a decryption application with private keys.

- For this purpose, the SMS service provided by the voice gateway is used to connect with the Xeon server using the shortcode 56677
- To settle the accounts between users and merchants in Fintech with the help of payment gateways/switches
- The mobile numbers should register with corresponding banks and telecom operators for payment transactions and SMS services
- On the other hand, the Xeon server has a decryption application.
- INFINET, called Indian Financial Network, established a secure HTTPS communication between the Xeon server and voice gate-


12:07

Macro Payments using WD

Enter Wearable_ID
WID1234

Enter Mobile Number
9000303647

Enter PIN
••••

Enter Amount xxxxxxxxx.xx
999999.99

ECC P-160 [ ]
ECC P-224 [ ]
[✓ ECC P-384]
ECC P-512 [ ]

ENCRYPT

io!j8rv6u%n3wi@1#uip9sl8
%csahu$sl@fi5k2k0lpn4m7lnb@galkl0

Fig. 6. The customer enters his details and message, then select the P-384 public key.

way. This link is between the server and the bank.
- For the credit card/debit card pay applications, the wearable device may represent cards transacting on any payment network (such as American Express, Discover, Mastercard, Visa, Maestro, Cirrus or others), each of which has its corresponding applet in the secure element (Alliance, 2016).

In this paper, for the practical implementation part, we did not show all the screenshots of the registration and login phases, but which are most relevant and essential to the customer and merchant are represented. The three parties (Mobile Device, Wearable Device and Customer) have never been authenticated, so they must initialise their information. Mobile device needs to store biometric data from the user; meanwhile, the user sets a password in a wearable device. The default setting is that the customer's ID is the legitimate primary user. Before pairing the wearable device with a mobile terminal, it must authenticate with the customer. If it is the first time wearable and mobile devices of respective customers authenticate each other, they must accomplish this setup process. After successful authentication, the customer can be seen as a trusted device, providing a secure channel. Once the devices are authentically successful, The mobile devices need to be installed APK and started to run this app. Step 1: The customer runs a macro payment application to enter the details of Wearable-ID, mobile customer number, pin and amount. After this, they choose any curve for an encrypted message shown in Fig. 6 to the Xeon server through shortcode 56677.

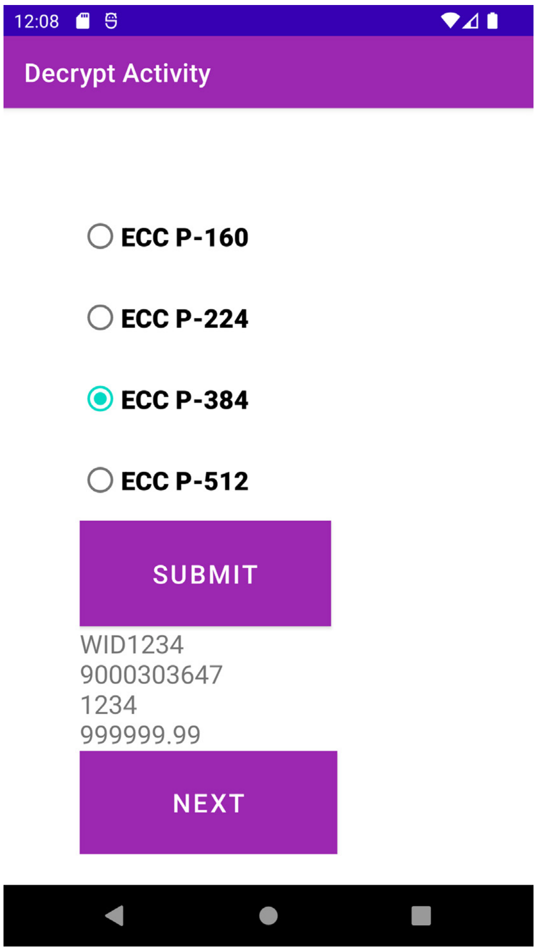
Step 2: The server receives the encrypted message and executes the decryption application activity. The merchant runs the decryption activity and selects the same curve for encryption, such as the P-384 private key. After decryption, the decrypted message is shown in Fig. 7.

Step 3: After decrypting the message, the user presses the "NEXT" button to communicate the payment gateway.

Step 4: Now, the merchant fills the remaining fields of merchant payment process details of Virtual Payment Address (VPA) UPI-id, mobile number, and purpose of payment and then selects any payment gateway, as shown in Fig. 8.

Step 5: Now, the gateway routed the payment process to respective transaction services provided by the banks. After the transaction is completed, the notification messages are sent as SMS to the customer and merchant entities as alert notifications.

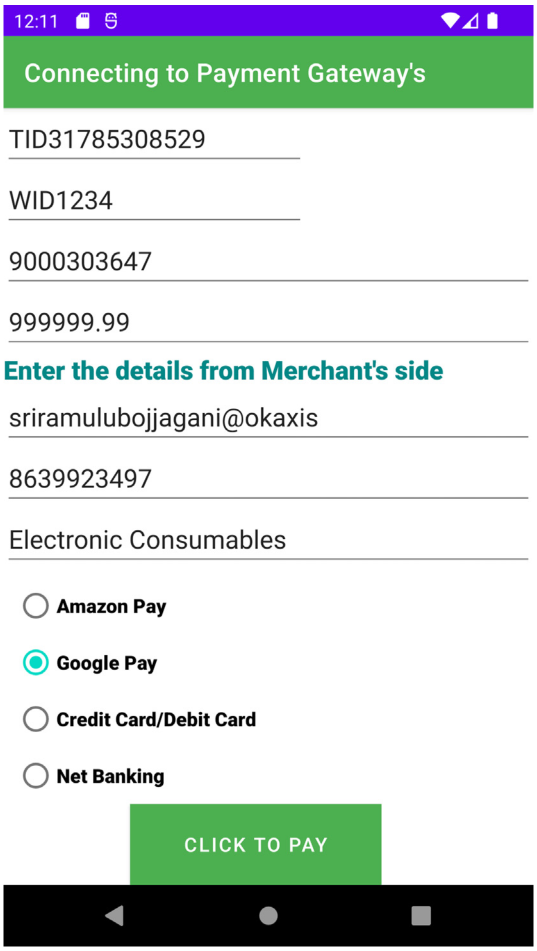
From these steps, we know that the security features of confidentiality, authentication and data Integrity are achieved. The application demonstration is shown in Fig. 9.



[✓ ECC P-384]

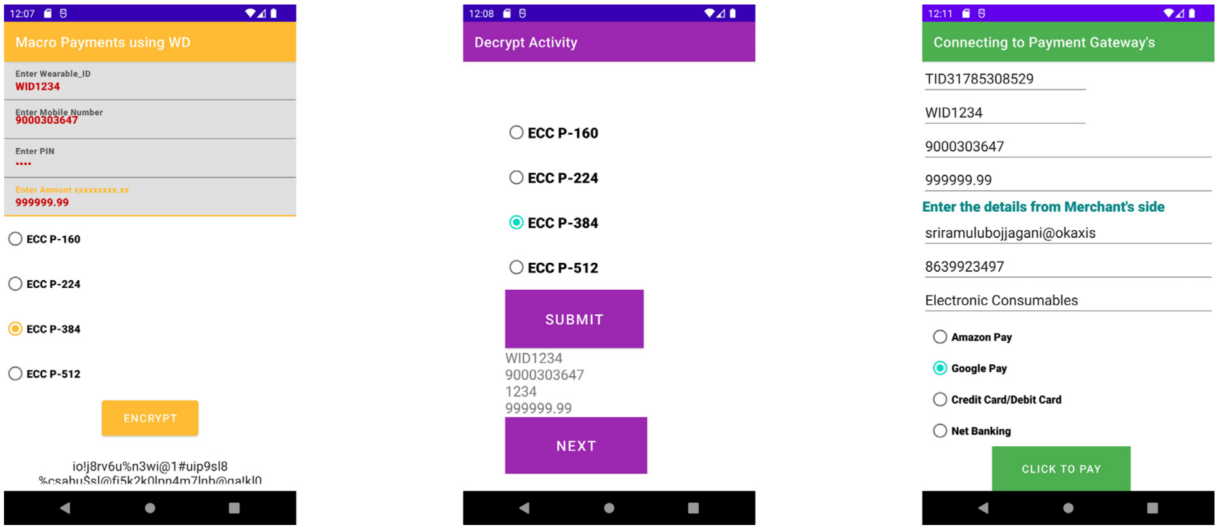
WID1234
9000303647
1234
999999.99

Fig. 7. The user runs the decryption algorithm and selects the P-384 private key.



TID31785308529
WID1234
9000303647
999999.99
sriramulubojjagani@okaxis
8639923497
Electronic Consumables
[✓ Google Pay]

Fig. 8. The merchant enters their details and clicks on any one payment gateway's.


12:07

Macro Payments using WD

Enter Wearable_ID: WID1234

Enter Mobile Number: 9000303647

Enter PIN: ••••

Enter Amount xxxxxxxxx.xx: 999999.99

ECC P-160 [✓ ECC P-384] ECC P-224 ECC P-512

ENCRYPT

io!j8rv6u%n3wi@1#uip9sl8
%csahu$sl@fi5k2k0lpn4m7lnb@qalkl0

12:08

Decrypt Activity

ECC P-160 ECC P-224 [✓ ECC P-384] ECC P-512

SUBMIT

WID1234
9000303647
1234
999999.99

NEXT

12:11

Connecting to Payment Gateway's

TID31785308529
WID1234
9000303647
999999.99

Enter the details from Merchant's side

sriramulubojjagani@okaxis
8639923497
Electronic Consumables

Amazon Pay [✓ Google Pay] Credit Card/Debit Card Net Banking

CLICK TO PAY

Fig. 9. Application demonstration.

9. Conclusions

Wearable devices in the financial sector have enormous potential to provide better services and end consumers. There is a demand for developing wearable apps for wearable devices to provide a secure environment between the devices, which helps the financial sector meet the ever-demanding consumer needs and become valued banks ahead of competitors. This paper addresses the security issues in wearable payments with the help of a threat model. It uses NFC for device pairing and ECC for encrypting the messages between the entities, and we have used biometrics as an additional authentication factor in secure payments. The proposed protocol analyses using widely accepted formal verification mechanisms of RoR and the model-checking tool of Scyther, revealing that it is free from the maximum possible attacks and makes the protocol run securely. Additionally, the proposed framework reduces communication and computation costs overhead compared to the existing schemes.

As part of our future work, we will focus on Blockchain integration because Blockchain changes wearable technology from comfort to need. It removes intermediaries from the situation and makes the process significantly quicker and easier.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Formal verification using Scyther: SPDL specification

We specify our proposed framework characteristics in “Security Protocol Description Language” (SPDL). To implement protocols in the Scyther tool, Cremers (2006). This protocol mainly describes five roles: Wearable Device (WD), Customer, Customer’s Bank (CB), Payment Gateway (PG), Merchant’s Bank (MB), and Merchant (M). The following code specifications in SPDL:

```
usertype IDwd,Nwd,TSwd,IDc,TSc,Nc,AMTc,
PI,POIc,TIDwd,TIDc,TIDm;
usertype POIwd,AMTwd,IDm,Nib,TSib,AMTm;
const puk:Function;
secret prk:Function;
inversekeys (puk,prk);
usertype TimeStamp,Success,failure,SMS;
hashfunction H1,H2,H3;
protocol Wearable(WD,C,CB,MB,PG,M).
{
role WD.
{
fresh nwd: Nonce;
send_1(WD,C,IDwd,Nwd,TSwdprk(WD)puk(C));
read_2(C,WD,IDwd,IDc,Nc,TSc,Nwd,TSwdprk(C)puk(WD));
send_3(WD,C,{{POIwd,AMTwd,IDwd,IDc,Nwd,
TSwd,TIDwd}prk(WD)}puk(C)); read_11(C,WD,{{SMS,IDc,IDwd}
prk(C)}puk(WD));
claim_WD1 (WD,Secret, Nwd);
claim_WD2 (WD,Secret, POIwd);
claim_WD3 (WD,Secret, IDwd);
claim_WD4 (WD,Secret, IDc);
claim_WD5(WD,Secret, prk(WD));
}
role C.
{
fresh nc: Nonce;
const Kcib:SessionKey;
read_1(WD,C,IDwd,Nwd,TSwdprk(WD)puk(C));
send_2(C,WD,IDwd,IDc,Nc,TSc,Nwd,TSwdprk(C)puk(WD));
read_3(WD,C,{{POIwd,AMTwd,IDwd,IDc,Nwd,
TSwd,TIDwd}prk(WD)}puk(C));
send_4(C,CB,{{(POIc,AMTc,IDc,TIDc,TSc,
IDwd,Nwd,TIDwd,POIwd,TSwd,
PIKcib)}prk(C)}puk(CB));
read_9(CB,C,{{TIDc,AMTc,IDc,IDwd,
IDm,SMS}prk(CB)}puk(C));
send_11(C,WD,SMS,IDc,IDwdprk(C)puk(WD));
claim_C1(C,Secret,Kcib);
claim_C2(C,Secret,Nc);
claim_C3(C,Secret,PI);
claim_C4(C,Secret,TIDc);
claim_C5(C,Secret,POIc);
claim_C6(C,Niagree);
claim_C7(C,Nisynch);
}
role CB.
{
const SKip,SKpi,Kcib:SessionKey;
read_4(C,CB, {{(POIc,AMTc,IDc,TIDc,TSc,IDwd,Nwd,
TIDwd,POIwd,TSwd, PIKcib)}prk(C)}puk(CB));
send_5(CB,PG,{H1(SKip),
(TIDc,POIc,AMTc,IDwd,IDc,Nib,TSib),
TIDc,POIc,AMTc,IDwd,IDc,Nib,TSib}SKip);
send_9(CB,C,{{TIDc,AMTc,IDc,IDwd,IDm,SMS}prk(CB)}puk(C));
read_8(PG,CB,{H3(SKpi),
(TIDc,AMTc,IDc,IDm,SMS),TIDc,AMTc,IDc,IDm,SMS}SKpi);
claim_CB1(CB,Secret, Skpi);
claim_CB2(CB,Secret, SKip);
claim_CB3(CB,Secret, PI);
claim_CB4(CB,Secret, POIc);
claim_CB5(CB,Niagree); claim_CB6(CB,Nisynch);
}
role PG.
{
const SKap,SKpa,Kmab,SKip,SKpi:SessionKey;
read_5(CB,PG,{H1(SKip),
(TIDc,POIc,AMTc,IDwd,IDc,Nib,TSib),
TIDc,POIc,AMTc,IDwd,IDc,Nib,TSib}SKip);     send_6(PG,MB,{H2
(SKpa),
(TIDc,POIc,AMTc,IDwd,IDc,TSib)}SKpa);
read_7(MB,PG,{TIDm,AMTm,IDm,IDc,IDwd,
PIKmab,SMS}SKap);
send_8(PG,CB,{H3(SKpi),
(TIDc,AMTc,IDc,IDm,SMS),TIDc,AMTc,IDc,IDm,SMS}SKpi);
claim_PG1(PG,Secret, SKpi);
claim_PG2(PG,Secret,SKap);
claim_PG3(PG,Secret,SKpa);
claim_PG4(PG,Secret, PI);
claim_PG5(PG,Niagree);
claim_PG6(PG,Nisynch);}
role MB.
{
const SKap,SKgi,SKpa,Kmab,Kcib,SKip:SessionKey;
read_6(PG,MB,{H2(SKpa), (TIDc,POIc,AMTc,
IDwd,IDc,TSib)}SKpa);
send_7(MB,PG,{TIDm,AMTm,IDm,IDc,IDwd,
PIKmab,SMS}SKap);
send_10(MB,M,{{PIKmab,SMS}prk(MB)}puk(M));
claim_MB1(MB,Secret, POIc);
claim_MB2(MB,Secret, SKpa);
claim_MB3(MB,Secret, PI);
claim_MB4(MB,Niagree);
claim_MB5(MB,Nisynch);
}
role M.
{
const SKap,SKgi,Kmab,SKip,Kcib:SessionKey;
read_10(MB,M,{{{PI}Kmab,SMS}prk(MB)}puk(M));
claim_M1(M,Secret, prk(MB));
claim_M2(M,Secret, PI);
claim_M3(M,Niagree);
claim_M4(M,Nisynch);
}}
```

Appendix B. Informal verification

Static security testing using Drozer: The proposed protocol is verified with the drozer tool, a static security analysis tool primarily used at application-level security. The wearable app initially connected with the mobile device with a laptop/desktop. Now drozer is running on the mobile device with root permissions. After successfully launching the device with the laptop, we will send commands over the attack surface to identify the application or code-level vulnerabilities. For the ideal case, on the attack surface, it is shown as 0's for the activities, broadcast receivers, content providers and services. The following output is shown for our proposed model on the attack surface. dz> run app.package.attacksurface com.siri.MacroPayment WD.

```
Attack Surface:
2 activities exported.
0 broadcast receivers exported.
0 content providers exported.
0 services exported.
is debuggable.
dz> run app.provider.info -a com.siri.MacroPayment WD.
Package: com.siri.MacroPayment WD.
“No matching providers” dz> run scanner.provider.finduris -a
com.siri.MacroPayment WD.
Scanning com.siri.MacroPayment WD.
“No accessible content URIs found”.
dz> run scanner.provider.injection -a com.siri.MacroPayment
WD.
Scanning com.siri.MacroPayment WD...
“Not vulnerable: No non-vulnerable URIs found.
Injection in Projection: No vulnerabilities found.
Injection in Selection: No vulnerabilities found”.
dz> run app.service.info -a com.siri.MacroPayment WD: com.
siri.MacroPayment WD.
“No exported services”.
```

References

Abdalla, M., Fouque, P.-A., Pointcheval, D., 2005. Password-based authenticated key exchange in the three-party setting. In: International Workshop on Public Key Cryptography. Springer, pp. 65–84.

Adavoudi-Jolfaei, A., Ashouri-Talouki, M., Aghili, S.F., 2019. Lightweight and anonymous three-factor authentication and access control scheme for real-time applications in wireless sensor networks. Peer-to-Peer Network. Appl. 12 (1), 43–59.

Alliance, S.C., 2016. Contactless emv payments: Benefits for consumers, merchants and issuers, Smart Card Alliance, US, 4–6.

Amin, R., Biswas, G., 2016. A secure light weight scheme for user authentication and key agreement in multi-gateway based wireless sensor networks. Ad Hoc Netw. 36, 58–80.

Bezovski, Z., 2016. The future of the mobile payment as electronic payment system. Eur. J. Bus. Manage. 8 (8), 127–132.

Sriramulu Bojjagani, V.N., Sastry, V.N., 2019. A secure end-to-end proximity NFC-based mobile payment protocol. Comput. Stand. Interfaces, Elsevier 66,. https://doi.org/10.1016/j.csi.2019.04.007 103348.

Bojjagani, S., Rao, P., Vemula, D.R., Reddy, B.R., Lakshmi, T.J., 2022. A secure iot-based micro-payment protocol for wearable devices. Peer-to-Peer Network. Appl., 1–26

Bojjagani, S., Sastry, V., Chen, C.-M., Kumari, S., Khan, M.K., 2023. Systematic survey of mobile payments, protocols, and security infrastructure. J. Ambient Intell. Humanized Comput. 14 (1), 609–654.

Challa, S., Wazid, M., Das, A.K., Kumar, N., Reddy, A.G., Yoon, E.-J., Yoo, K.-Y., 2017. Secure signature-based authenticated key establishment scheme for future iot applications. IEEE Access 5, 3028–3043.

Chen, Y., Xu, W., Peng, L., Zhang, H., 2019. Light-weight and privacy-preserving authentication protocol for mobile payments in the context of iot. IEEE Access 7, 15210–15221.

Chong, S., Guttman, J., Datta, A., Myers, A., Pierce, B., Schaumont, P., Sherwood, T., Zeldovich, N., 2016. Report on the nsf workshop on formal methods for security, arXiv preprint arXiv:1608.00678.

Coskun, V., Ozdenizci, B., Ok, K., 2013. A survey on near field communication (nfc) technology. Wireless Personal Commun. 71, 2259–2294.

Coskun, V., Ozdenizci, B., Ok, K., 2015. The survey on near field communication. Sensors 15 (6), 13348–13405.

Cremers, C.J.F., 2006. Scyther: Semantics and verification of security protocols. Eindhoven University of Technology Eindhoven, Netherlands.

Cremers, C.J., 2008. The scyther tool: Verification, falsification, and analysis of security protocols. In: International Conference on Computer Aided Verification. Springer, pp. 414–418.

Cremers, C., 2009. The scyther tool: Automatic verification of security protocols.

Dalal, N., Shah, J., Hisaria, K., Jinwala, D., et al., 2010. A comparative analysis of tools for verification of security protocols. Int'l J. Commun., Network Syst. Sci. 3 (10), 779.
Das, A.K., Sharma, P., Chatterjee, S., Sing, J.K., 2012. A dynamic password-based user authentication scheme for hierarchical wireless sensor networks. J. Network Comput. Appl. 35 (5), 1646–1656.
Das, A.K., Wazid, M., Kumar, N., Khan, M.K., Choo, K.-K.R., Park, Y., 2017. Design of secure and lightweight authentication protocol for wearable devices environment. IEEE J. Biomed. Health Informat. 22 (4), 1310–1322.
Das, A.K., Zeadally, S., Wazid, M., 2017. Lightweight authentication protocols for wearable devices. Comput. Electr. Eng. 63, 196–208.
Diallo, A., Al-Khateeb, W.F.M., Olanrewaju, R., Sado, F., 2014. A secure authentication scheme for bluetooth connection. In: 2014 International Conference on Computer and Communication Engineering. IEEE, pp. 60–63.
Dolev, D., Yao, A., 1983. On the security of public key protocols. IEEE Trans. Infor. Theory 29 (2), 198–208.
Fahl, S., Harbach, M., Muders, T., Baumgärtner, L., Freisleben, B., Smith, M., 20012. Why eve and mallory love android: An analysis of android ssl (in) security. In: Proceedings of the 2012 ACM Conference on Computer and Communications Security, pp. 50–61.
Felt, A.P., Chin, E., Hanna, S., Song, D., Wagner, D., 2011. Android permissions demystified. In: Proceedings of the 18th ACM Conference on Computer and Communications Security, pp. 627–638.
Gallagher, P., Director, A., 1995. Secure hash standard (shs). FIPS PUB 180, 183.
Gao, Y., Li, H., Luo, Y., 2015. An empirical study of wearable technology acceptance in healthcare. Ind. Manage. Data Syst.
Gao, Y., Al-Sarawi, S.F., Abbott, D., 2020. Physical unclonable functions. Nat. Electron. 3 (2), 81–91.
Gope, P., Hwang, T., 2016. A realistic lightweight anonymous authentication protocol for securing real-time application data access in wireless sensor networks. IEEE Trans. Ind. Electron. 63 (11), 7124–7132.
Gupta, A., Tripathi, M., Shaikh, T.J., Sharma, A., 2019. A lightweight anonymous user authentication and key establishment scheme for wearable devices. Comput. Netw. 149, 29–42.
Hankerson, D., Menezes, A., 2011. Elliptic curve discrete logarithm problem.
Hankerson, D., Menezes, A.J., Vanstone, S., 2006. Guide to Elliptic Curve Cryptography. Springer Science & Business Media.
Haselsteiner, E., Breitfuß, K., 2006.Security in near field communication (nfc)-strengths and weaknesses.
He, D., Zeadally, S., Xu, B., Huang, X., 2015. An efficient identity-based conditional privacy-preserving authentication scheme for vehicular ad hoc networks. IEEE Trans. Inf. Forensics Secur. 10 (12), 2681–2691.
Helfmeier, C., Boit, C., Nedospasov, D., Tajik, S., Seifert, J.-P., 2014. Physical vulnerabilities of physically unclonable functions. 2014 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, pp. 1–4.
Jason Fernando, 2022. Cost of Goods Sold (COGS), Investopedia, 11th Aug 2022, https://www.investopedia.com/terms/c/cogs.asp (Accessed: 2023-01-25).
Kim, M., Lee, J., Yu, S., Park, K., Park, Y., 2019. A secure authentication and key establishment scheme for wearable devices. In: 2019 28th International Conference on Computer Communication and Networks (ICCCN). IEEE, pp. 1–2.
Koblitz, N., 1987. Elliptic curve cryptosystems. Mathe. Comput. 48 (177), 203–209.
Kumar, D., Grover, H.S., et al., 2019. A secure authentication protocol for wearable devices environment using ecc. J. Infor. Sec. Appl. 47, 8–15.
Lee, C.-C., Chen, C.-T., Wu, P.-H., Chen, T.-Y., 2013. Three-factor control protocol based on elliptic curve cryptosystem for universal serial bus mass storage devices. IET Comput. Digital Techn. 7 (1), 48–55.
Levy, S., 2015. Performance and security of ecdsa. Comput. Sci.
Liu, S., Hu, S., Weng, J., Zhu, S., Chen, Z., 2016. A novel asymmetric three-party based authentication scheme in wearable devices environment. J. Netw. Comput. Appl. 60, 144–154.
Liu, W., Liu, H., Wan, Y., Kong, H., Ning, H., 2016. The yoking-proof-based authentication protocol for cloud-assisted wearable devices. Pers. Ubiquit. Comput. 20 (3), 469–479.
Lo, N.-W., Yohan, A., 2020. Ble-based authentication protocol for micropayment using wearable device. Wireless Pers. Commun., 1–22
Madhusudhan, R., Shashidhara, R., 2020. A secure anonymous authentication protocol for roaming service in resource-constrained mobility environments. Arabian J. Sci. Eng. 45, 2993–3014.
Magdum, A., Sivaraman, E., Honnavalli, P.B., 2021. Contactless transaction using wearable ring with biometric fingerprint security feature. In: Computer Networks and Inventive Communication Technologies. Springer, pp. 653–666.
Mahto, D., Khan, D.A., Yadav, D.K., 2016. Security analysis of elliptic curve cryptography and rsa. In: Proceedings of the World Congress on Engineering, vol. 1, pp. 419–422.
Patel, R., Kunche, A., Mishra, N., Bhaiyat, Z., Joshi, R., 2015. Paytooth-a cashless mobile payment system based on bluetooth. Int. J. Comput. Appl. 120 (24).
Santosa, G.B., Budiyanto, S., 2019. New design of lightweight authentication protocol in wearable technology. Telkomnika 17 (2), 561–572.
Sedita, J., Perrella, S., Morio, M., Berbari, M., Hsu, J.-S., Saxon, E., Jarrahian, C., Rein-Weston, A., Zehrung, D., 2018. Cost of goods sold and total cost of delivery for oral and parenteral vaccine packaging formats. Vaccine 36 (12), 1700–1709.
Segura Anaya, L., Alsadoon, A., Costadopoulos, N., Prasad, P., 2018. Ethical implications of user perceptions of wearable devices. Sci. Eng. Ethics 24, 1–28.
Seneviratne, S., Hu, Y., Nguyen, T., Lan, G., Khalifa, S., Thilakarathna, K., Hassan, M., Seneviratne, A., 2017. A survey of wearable devices and challenges. IEEE Commun. Surv. Tutor. 19 (4), 2573–2620.
Sun, D.-Z., Huai, J.-P., Sun, J.-Z., Zhang, J.-W., Feng, Z.-Y., 2008. A new design of wearable token system for mobile device security. IEEE Trans. Consum. Electron. 54 (4), 1784–1789.
Tabet, N.E., Ayu, M.A., 2016. Analysing the security of nfc based payment systems. In: 2016 International Conference on Informatics and Computing (ICIC). IEEE, pp. 169–174.
Wang, D., He, D., Wang, P., Chu, C.-H., 2014. Anonymous two-factor authentication in distributed systems: Certain goals are beyond attainment. IEEE Trans. Dependable Secure Comput. 12 (4), 428–442.
Wazid, M., Das, A.K., Odelu, V., Kumar, N., Susilo, W., 2017. Secure remote user authenticated key establishment protocol for smart home environment. IEEE Trans. Dependable Secure Comput. 17 (2), 391–406.
Wong, K.H., Zheng, Y., Cao, J., Wang, S., 2006. A dynamic user authentication scheme for wireless sensor networks. IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'06), vol. 1. IEEE, pp. 8–pp.
Wu, F., Li, X., Xu, L., Kumari, S., Karuppiah, M., Shen, J., 2017. A lightweight and privacy-preserving mutual authentication scheme for wearable devices assisted by cloud server. Comput. Electr. Eng. 63, 168–181.
Yohan, A., Lo, N.-W., Randy, V., Chen, S.-J., Hsu, M.-Y., 2016. A novel authentication protocol for micropayment with wearable devices. In: ACM Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication, Danang, Viet Nam, January 04–06, 2016, 2016, pp. 1–7.