

Received November 25, 2021, accepted December 18, 2021, date of publication December 28, 2021, date of current version January 6, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3139065

A Novel NFC-Based Secure Protocol for Merchant Transactions

SHAIK SHAKEEL AHAMAD 

Department of Information Technology, College of Computer and Information Sciences, Majmaah University, Al Majma'ah 11952, Saudi Arabia
e-mail: ahamadss786@gmail.com; s.ahamad@mu.edu.sa


The work of Shaik Shakeel Ahamad was supported by the Deanship of Scientific Research, Majmaah University, under Project R-2021-313.

ABSTRACT The unprecedented growth of mobile applications promoted the usage of these mobile applications for payments. The current research works in mobile payments and commerce are prone to reverse-engineering attacks and lacked transport layer protection, so these research works do not ensure security. Therefore, such attacks on Mobile Payment Applications (MPA) will be successful, which leads to severe financial loss. To address these issues, we propose a secure framework incorporating a defense-in-depth approach for Near Field Communication (NFC) based mobile payment frameworks. Our defense-in-depth approach has three levels, i.e., Defense at hardware, mobile application, and communication level. We have proposed a NFC based Secure Protocol for Mobile Transaction (NSPMT) protocol and successfully verified a mobile payment protocol with BAN (Burrows, Abadi, and Needham) logic and Scyther tool, and our proposed protocol overcome multi-protocol attack, RAM (Random Access Memory) scrapping attack, DOS (Denial Of Service), DDOS (Distributed Denial Of Service), and Phlashing attacks. Our proposed mobile Payment system overcomes the known mobile application vulnerabilities, including Heartbleed and ROBOT (Return Of Bleichenbacher's Oracle Threat). Our proposed protocol ensures all the security properties and the energy and communication cost and computational cost are far less than the existing works in the literature. Finally, we have successfully implemented our protocol using kotlin language in Android Studio, with two Mobile Payment Applications (MPA) and POS Payment Application (PPA), Elliptic Curve Digital Signature Algorithm (ECDSA) is used and Advanced Encryption Standard (AES) with GCM (Galois/Counter Mode) mode is used for encryption and decryption of Customer Payment Data at MPA and PPA.

INDEX TERMS MPA, BAN logic, RAM scraping, phlashing attacks, Heartbleed and ROBOT vulnerabilities, Scyther tool, reverse-engineering attacks, Kotlin language.

I. INTRODUCTION

The unprecedented growth of smartphones promoted mobile payment services based on mobile applications as consumers are adopting cashless payments. Information and communication technology (ICT) is widely used all around the globe [32]. With the comfort and acceptance of NFC smartphones, merchants encourage consumers for NFC based proximity payments. Berg Insight predicts that the exports of Point Of Sale (POS) based on NFC will be 4.1 million by 2022 [19]. MPAs are playing a vital role in mobile payment frameworks. MPAs are not as trustworthy as the intruders can tamper the MPA, so there is a need to strengthen the

The associate editor coordinating the review of this manuscript and approving it for publication was Liang-Bi Chen .

MPA. As per our knowledge, there exists no solution to overcome the issues highlighted in the existing mobile payment research works. Ninety-eight percent of the mobile applications are reverse-engineered, and eighty-three percent of the mobile applications lacked transport layer security. Security and privacy of mobile transactions is the major hindrance to wide-spread adoption of these services. Therefore, such vulnerabilities in MPA will hinder the adoption of mobile payments. So security should be included and incorporated in the design at every phase. The main contributions of this work are:

- a) We propose a NFC based Secure Protocol for Mobile Transaction (NSPMT) incorporating a defense-in-depth approach at three levels, i.e., Defense at hardware, mobile application, and at communication levels.

- b) The proposed payment framework overcomes RAM scrapping attack, DOS, DDOS, and Phlashing attacks.
- c) MPA in our proposed payment framework overcomes the Heartbleed and ROBOT mobile application vulnerabilities.
- d) We propose a secure POS-based payment protocol and the proposed payment protocol is successfully verified with BAN logic and Scyther tool.
- e) The energy and communication cost and computational cost of our protocol are far less than the existing works in the literature.
- f) Proposed protocol was implemented using kotlin language in Android Studio.

The remaining article's organization is as follows: In Section II, we provide background and preliminaries on NFC, UICC (Universal Integrated Circuit Card), Hardware Security Module (HSM), and MPA. In Section III, we discuss the related work in the realm of secure proximity payments based on NFC. Section IV proposes a Secure Mobile Payment System based on NFC incorporating a defense in the Depth approach at the secure elements level and payment application level. Section V provides the formal verification of the proposed protocol. Section VI includes security analysis. Section VII presents the implementation and performance analysis of the proposed protocol, and Section VIII provides the conclusion of the paper.

II. BACKGROUND AND PRELIMINARIES

Customer possesses UICC in a smartphone, Merchant is an entity which sells goods or services and possesses a HSM, Bank is the financial institution of both the Customer and Merchant, MNO provides mobile network connectivity and updates Over The Air (OTA). PG acts as an adjudicator, containing evidence repository and provenance repository.

Bank and Payment Gateway (PG) is an integral part of a secure private banking network that securely exchanges messages without encryption. Customer's anonymity is ensured by Traceable anonymous certificate (TAC), MPA is in the UICC of the smartphone, MPA shares a symmetric key between the Bank (B) and the Customer (C), POS Payment Application (PPA) shares a symmetric key between the Bank (B) and the POS. NFC is a technology very much compatible with the technologies used in transport and proximity payments. It is a high-frequency radio standard helping in exchanging wireless data within a range of 10 cm. NFC is compatible with ISO/IEC 14443 standard cards and readers and also with NFC enabled smartphones. According to GlobalPlatform [11] a Secure Element (SE) is a tamper-resistant hardware device which hosts applications. ETSI project smart card platform (ETSI EP SCP) standardized UICC, a universal platform for smart card applications. UICC hosts different mobile applications and allocates separate security domains for each application, governed by the Application Owner (AO). Card's Operating System (COS) of the UICC enforces firewalls among applications restricting

applications from interfering with the working of other applications. As defined by Payment Card Industry (PCI) [17], HSM provides secure cryptographic services by implementing cryptographic logic, algorithms and processes [17]. Wireless Public Key Infrastructure (WPKI) ensures all the security properties, but the implementation of WPKI in the smartphone's memory is dangerous as the malware compromises cryptographic keys. UICC can generate and securely store the client's credentials, which includes private keys and X.509 certificates. In addition to these, digital signatures are also generated securely in UICC. All the entities in the framework trust certification Authority (CA) as it plays the role of a Trusted Service Manager (TSM) and adjudicator and its regular functions, including issuing certificates. Registration Authority (RA) verifies the entities' credentials involved in the ecosystem. The Bank acts as a RA in our proposed mobile payment framework. OCSP (Online Certificate Status Protocol) is an integral part of CA which updates the status of the revoked and compromised certificates. White Box Cryptography (WBC) is used to securely store symmetric keys and execute the symmetric encryption in MPA. UICC hosts Mobile Payment Application (MPA); MPA works with remote and NFC proximity mobile payments. It stores the keys and executes symmetric encryption using White-Box Cryptography (WBC). MPA is protected by PIN and biometric. Point Of Sale (POS) contains the HSM, which helps store cryptographic keys and execute cryptographic calculations. HSM is a Common Criteria EAL (Evaluation Assurance Level) 5 certified device ensuring the keys' security with effective, secure key management. HSM hosts PPA, PPA works with only NFC based payments, and it also stores the keys and executes symmetric encryption using WBC. PPA is protected by PIN and biometric.

III. RELATED WORK

[1] proposes a cloud-based mobile payment mechanism, it claims security, anonymity, fairness, and the reduction of computational cost. After critically reviewing, we found the following limitations in [1]

- a) There is no clarity on how the client or customer interacts with the cloud.

[2] proposes an extended version of the NFC cloud Wallet, where SE authenticates customers, but the cloud stores customer's payment information. Following are the limitations in [2]

- a) There is no clarity on how the cloud ensures the security of the payment information.
- b) There is no clarity on how the client or customer interacts with the cloud.

Isaac and Zeadally (2012) [3] proposes a payment gateway centric model for a anonymous secure payment mechanism. anonymous in a payment gateway centric model. Client and merchant exchange transaction data through a payment gateway. But the proposed work has the following limitations

- a) The client can deny Payment information as the generation of Payment information is anonymous.

[4] proposes an efficient three-party authentication protocol based on ECC (Elliptic Curve Cryptography) algorithm in mobile commerce. It claims to have fewer computation and communication costs. [5] proposes an authenticated encryption scheme based on ECC algorithm. [6] proposes a mechanism for mobile electronic transactions based on cloud computing claiming that the client and merchant do not require a shared symmetric key. The following are the drawbacks of the proposed mechanism.

- There is no clarity about the role of the Trusted Authority (TA)
- There is no clarity on how the client securely stores his credentials and payment information in the personal cloud.
- There is no clarity on how the Merchant and Bank securely store their credentials and their transaction data in the public cloud.

[23] proposes a protocol for mobile payments based on NFC by making use of SE. [24] proposes a new communication network that connects banks with its client's mobile phones. It claims that it ensures security without compromising efficiency. The proposed work has the following limitations:

- There is no clarity on how the Bank ensures security.

[5] proposes an e-payment system that is vulnerable to impersonation attacks. [25] Overcomes the shortcoming of [5] by proposing an improved authenticated encryption and e-payment schemes. It claims that the schemes are more robust and more lightweight than [5]. [26] proposes an offline mobile payment protocol which is compatible to EMV with mutual authentication using the reverse hash chain technique. [27] proposes an EMV-compatible payment protocol in order to overcome the risk in transactions. Communications security is ensured between a card and a card reader in order to overcome eavesdropping on sensitive data. In addition to these, the protocol resists impersonation attacks and avoids the security threats in EMV. [28] proposes a mobile payment protocol which is lightweight and based on Short Message Service (SMS) that ensures information security and fair exchange properties. The proposed protocol is formally proven using BAN logic and the Scyther tool. [29] proposes a lightweight and secure NFC mobile payment protocol ensuring information security and fair exchange properties for sales transaction processing. The proposed protocol is formally proven using both Burrows, Abadi, and Needham (BAN logic) and the Scyther tool. [30] proposes a secure operational model for mobile payments based on a service-oriented architecture based on a two-dimensional barcode as the payment certificate. Authors of [31] proposed a NFC mobile payment protocol based on public key cryptography.

All the research works discussed in this section are vulnerable to reverse-engineering, DOS, DDOS, Phlashing attacks, lacked transport layer protection and does not ensure end to end security and evidence cannot be

TABLE 1. List of abbreviations and notations.

Abbreviation	Description
BAN	Burrows, Abadi, and Needham
DOS	Denial Of Service
DDOS	Distributed Denial Of Service
ROBOT	Return Of Bleichenbacher's Oracle Threat
ECDSA	Elliptic Curve Digital Signature Algorithm
MPA	Mobile Payment Application
PPA	POS Payment Application
RAM	Random Access Memory
AES	Advanced Encryption Standard
GCM	Galois/Counter Mode
POS	Point Of Sale
NFC	Near Field Communication
UICC	Universal Integrated Circuit Card
HSM	Hardware Security Module
TSM	Trusted Service Manager
CA	Certifying Authority
WPKI	Wireless Public Key Infrastructure
PCI	Payment Card Industry
COS	Card Operating System
AO	Application Owner
SE	Secure Element
ISO/IEC	International Organization for Standardization/International Electro technical Commission
ECC	Elliptic Curve Cryptography
TA	Trusted Authority
EMV	Europay, MasterCard and Visa
SMS	Short Message Service
PG	Payment Gateway
OTA	Over The Air
MNO	Mobile Network Operator
WBC	White Box Cryptography
EAL 4+	Evaluation Assurance Level 4+
TLS	Transport Layer Security
OCSP	Online Certificate Status Protocol
TEE	Trusted Execution Environment
EAC	Enrolment Activation Code
SEPM	Secure Evidence Preservation Module
SPM	Secure Provenance Module
PASS	Provenance-Aware Storage System
NTP	Network Time Protocol
SPDL	Security Protocol Description Language
NSPMT	NFC based Secure Protocol for Mobile Transaction
POS	Pont Of Sale
ID _C	Identity of Customer
ID _{POS}	Identity of POS
LOC _C	Location of Customer
SK _{CB}	Shared key between C & B
SK _{POSB}	Shared key between POS & B
T _C	Time Stamp generated by the customer
T _{POS}	Time Stamp generated by POS
N _C	Nonce generated by Customer
N _{POS}	Nonce generated by POS
LOC _{POS}	Location of POS
TID	Transaction Identity
PI _C	Payment Information of the Customer

guaranteed in case of security breaches. All the research works (except [28] and [29]) discussed in this section are vulnerable to multi-protocol attacks. [1]–[6] and [23] protocols are not formally verified. All the research works discussed in this section are vulnerable to RAM scrapping attack and

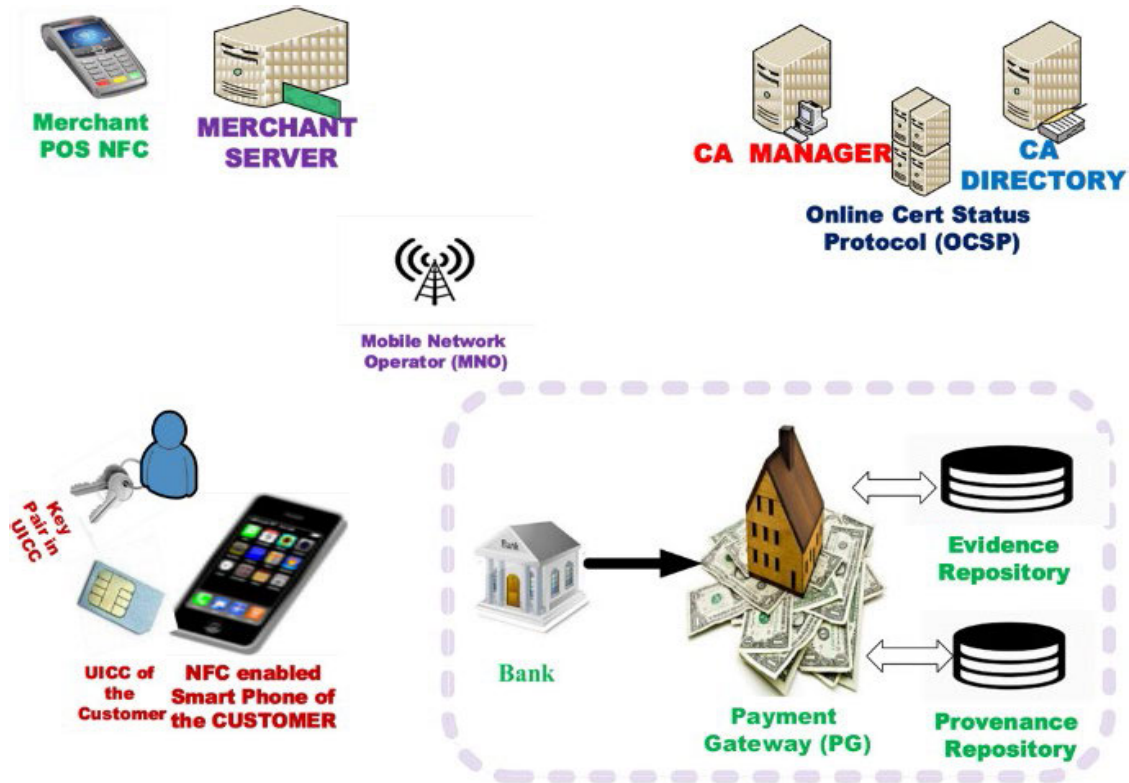


FIGURE 1. Entities involved in the proposed mobile payment systems.

fails to withstand Heartbleed and ROBOT vulnerabilities. As per our knowledge, we are the first to address Heartbleed and ROBOT vulnerabilities, RAM scrapping attack. [4]–[6], [23], [5] and [26]–[30] protocols cannot withstand insider attacks. [25]–[30] protocols cannot withstand stolen smart card attack, parallel session attack, physically stolen device attack and unauthorized key computation attacks. [4]–[6], [23], [24] and [25]–[30] protocols do not ensure communication and application security. Except [23], no work has implemented its proposed protocol in real-time. Table 1 shows the list of abbreviations used in this paper.

IV. PROPOSED MOBILE PAYMENT SYSTEM

Customer (C), Merchant (M), Bank (B), Mobile Network Operator (MNO), Payment Gateway (PG) and Certifying Authority (CA) are the entities and Wireless Public Key Infrastructure (WPKI), White Box Cryptography (WBC), Near Field Communication (NFC) are the technologies involved in the proposed secure mobile payment system.

Figure 1 depicts the entities involved in the proposed Mobile Payment System.

A. PROPOSED DEFENSE IN DEPTH APPROACH FOR SECURE MOBILE PAYMENTS

We incorporate security in our proposed framework because adding security at the end of the development phase can be very costly. So, our proposed framework incorporates a

defense in depth approach. So we propose to have Defense in Depth at two levels

- a) Securing the Secure Elements of the Customer and Merchant
- b) Securing the payment applications of the Customer and Merchant

Figure 2 depicts the proposed defense-in-depth approach for the mobile payment system.

1) DEFENSE AT HARDWARE LEVEL

The manufacturer of UICC and HSM requests CA for a Chip certificate. CA issues a chip certificate to the SE and HSM chip. EAL4+ (Evaluation Assurance Level 4+) certificate is issued by CA for integrated circuit (IC) chip. CA also issues a certificate to the operating system (OS), which excludes applications. In our proposed mobile payment system, customers possess UICC in the smartphone as a SE, and the merchant uses the Hardware Security Module in its POS. WPKI ensures end to end security as UICC and HSM have WPKI functionality to generate and securely store client’s and merchant’s credentials, including private keys and X.509 certificates. In addition to these, digital signatures are also generated securely in UICC and HSM. UICC and HSM host several mobile applications, either from the UICC and HSM issuer or from other service providers, each application defines and administers its application. UICC and HSM allocate separate security domains for

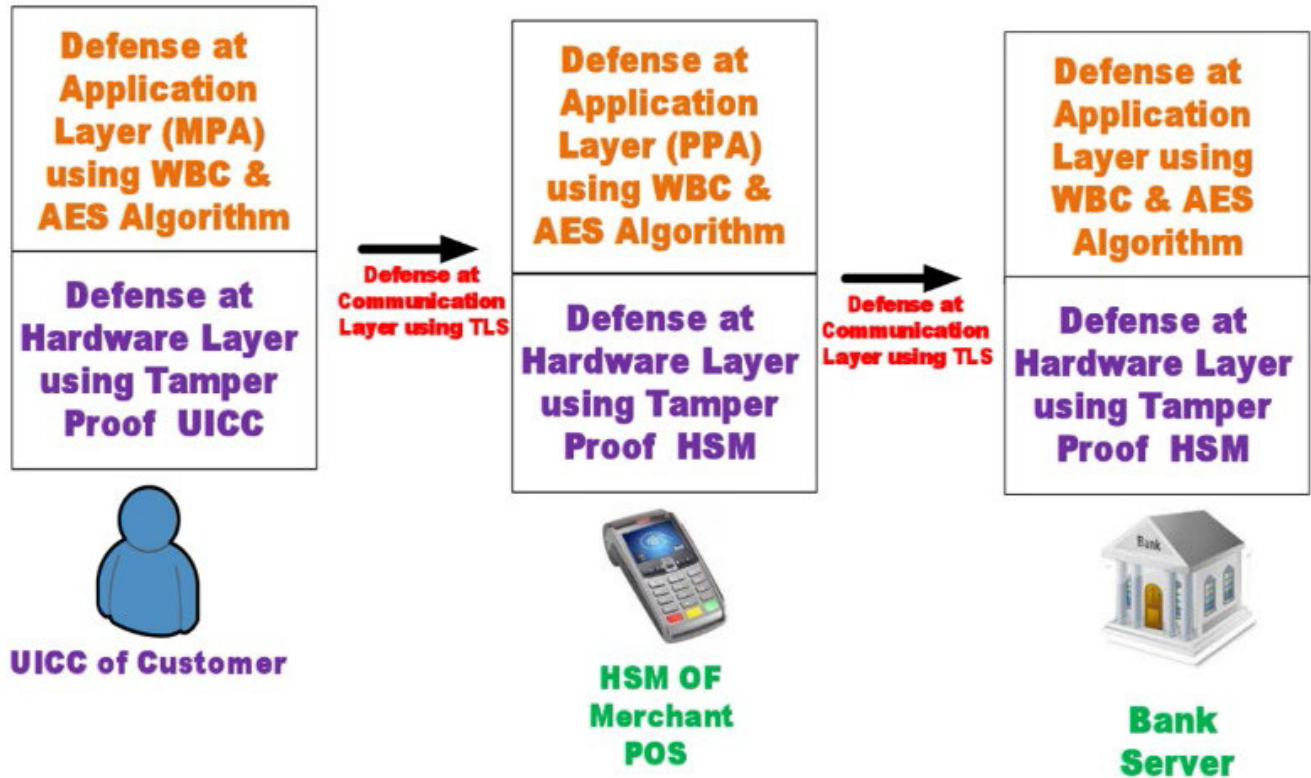


FIGURE 2. Proposed defense-in-depth approach.

each application, governed by the Application Owner (AO). Respective Operating Systems enforces firewalls among applications restricting applications from interfering with the working of other applications. MPAs are customized by the Application Issuer using Over The Air (OTA) technology. So UICC and HSM cannot be tampered and securely protects the private keys and payment applications. Bank (B) acts as a Registration Authority (RA); it registers all the Customers (C) and Merchants (M) and helps in getting certificates from Certification Authority (CA). CA maps the certificate identity and chip certificate to the uniqueness of the entities in the ecosystem. The Bank contains the database of the customers and merchants.

2) DEFENSE AT APPLICATION LEVEL

MPA is in the UICC of the smartphone; MPA shares a symmetric key between the Bank (B) and the Customer (C). PPA is in the HSM of the Merchant's POS. PPA shares a symmetric key between the Bank (B) and the POS. Using the procedure given in (Kungpisdan *et al.*, 2003) [8], the generation of new session keys is possible using hashing algorithms given in (Kungpisdan *et al.*, 2003) [8].

This system uses WBC [7], which ensures the security of secret and session keys in the MPA. The Bank updates the keys in the MPA of SE on the smartphone. The symmetric key is protected in the payment applications using

WBC. The adversaries cannot extract the key from the payment application despite knowing the encryption algorithms and key lengths. WBC provides Trusted Execution Environment (TEE) in the payment application. The Bank is the Application Provider (AP) of both MPA and PPA, and CA verifies the authenticity of both MPA and PPA. MPA and PPA overcome reverse engineering attacks by binary code obfuscation, flow relocation, stripping debugging information, and by encrypting strings and resources in the code. In addition to these, the Bank enforces Self-Signing Restriction on MPA and PPA, and the Bank's private key attests the code of the MPA and PPA, so MPA and PPA overcome reverse-engineering attacks.

3) DEFENSE AT COMMUNICATION LEVEL

Our proposed Mobile Payment framework's security does not rely on network layer security as it is susceptible to eavesdropping attacks. All the entities involved in our proposed Mobile Payment framework use CA-signed certificates. The entities involved in the ecosystem implement certificate pinning, ensuring communication security. Bank detects DoS attacks from activity profiling, change-point detection, and wavelet-based signal analysis detection techniques. The Bank only communicates with the legitimate subscribers. Bank only establishes a secure connection with the legitimate subscribers by establishing a secure channel with the payment

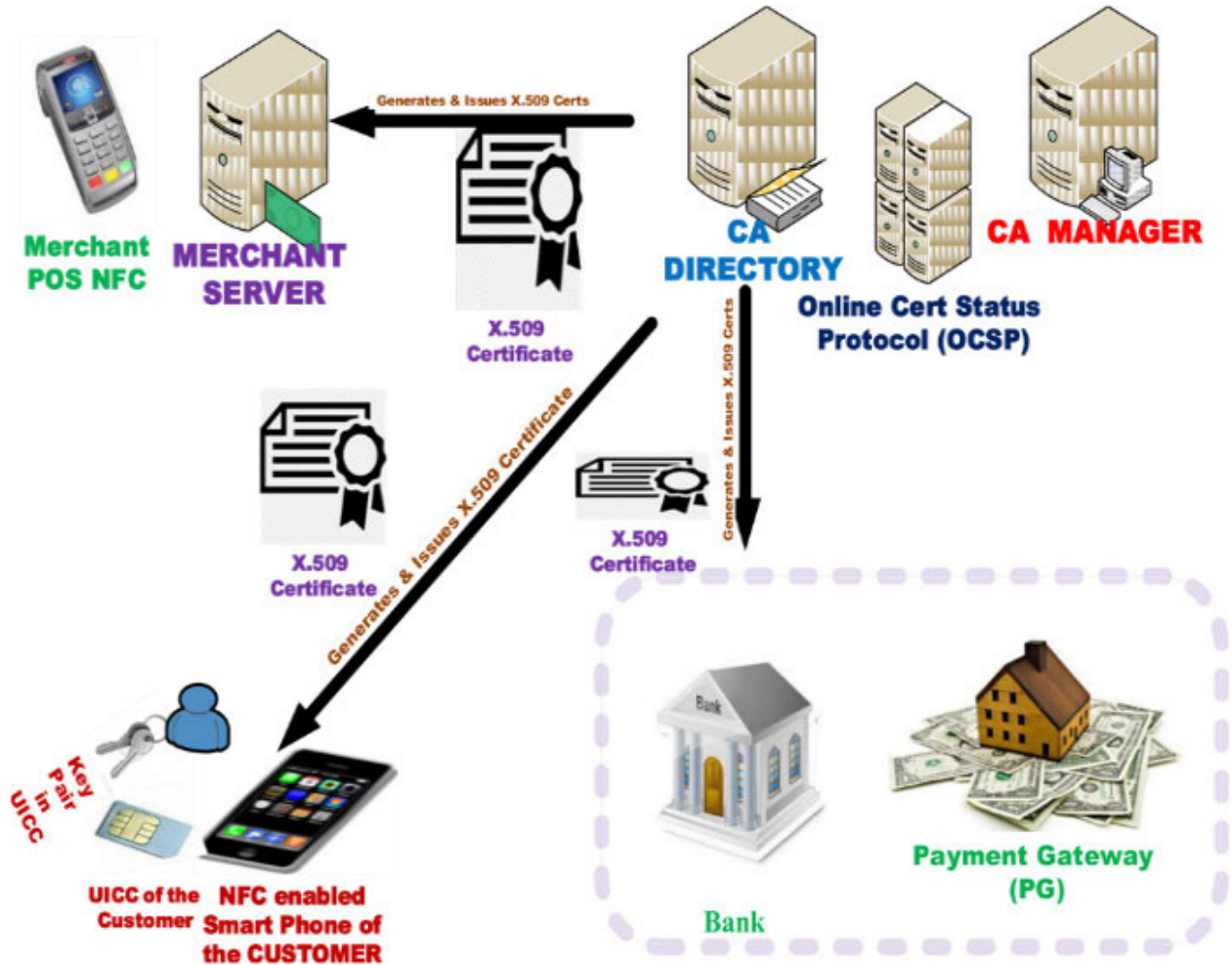


FIGURE 3. Generation & issuance of certificates by the CA.

application in UICC of smartphone and HSM of POS using TLS (Transport Layer Security) protocol at the communication layer. Bank updates the security of the payment applications Over The Air (OTA) by patch management.

B. GENERATION AND ISSUANCE OF CERTIFICATES BY THE CA

This section explains the generation and issuance of X.509 certificates to merchants, customers, and banks. Figure 3 depicts the generation and issuance of certificates by the CA. CA issues the following certificates:

- a. Chip certificate: EAL4+ (Evaluation Assurance Level 4+) certificate is issued by CA for HSM.
- b. OS certificate: SE and HSM has its OS certificates SE.
- c. Application certificate: Based upon the Bank’s recommendations, CA issues certificates to both MPAs and PPAs after verifying the applications’ authenticity.
- d. Client and Merchant certificate: Bank acts as a Registration Authority (RA), the Bank verifies the credentials of these entities, and if the verification is successful, it recommends CA to issue certificates to these entities.

- e. CA issues Traceable Anonymous Certificates to all the entities [18].
- f. CA manager manages all the day to day operations of the CA.
- g. CA directory keeps all the certificates in its directory.
- h. OCSP (Online Certificate Status Protocol) server updates the status of the revoked and compromised certificates.

We propose three algorithms in this sub-section; they are Algorithm 1: Generation and Issuance of X. 509 certificates to the customer by the CA, Algorithm 2: Generation and Issuance of X. 509 certificates to the merchant by the CA, and Algorithm 3: Generation and Issuance of X. 509 certificates to the Bank by the CA.

C. PROPOSED SECURE MOBILE PAYMENT PROTOCOL

Our proposed, secure mobile payment protocol has three steps. Figure 4 depicts the steps involved in the proposed Mobile Payment protocol. Table 2 shows the notations used in the proposed protocol.

Step 1: Customer (C) selects and collects all the items in the store and comes to the POS for paying the bill. The merchant calculates the bill amount and displays it at the counter. Merchant shows a unique Merchant ID (MID) at the counter for the convenience of the customers.

The customer encrypts the filled-in MPA with the symmetric key shared between the Customer (C) and the Bank (B). Customer (C) sends the encrypted message to the POS using NFC link.

Step 1: C → POS: {MS1}SK_{CB}

MS1: {ID_C, ID_{POS}, AMT, PI_C, Nc T_C, LOC}

Step 2: After receiving the encrypted message from the Customer, POS fills his/her PPA with Customer ID (ID_C), Amount (AMT), Transaction ID (TID) remaining attributes including ID_{POS} (ID of Merchant), LOC_{POS} (Location of the POS), T_{POS} (Timestamp of the POS), N_{POS} (Nonce of the POS) then defaults attributes. Filled in PPA along with {MS1}SK_{CB} is encrypted (with the symmetric key shared between the POS and the Bank (B)) and sent to the Bank.

Step 2: POS → B: {MS2}SK_{POSB}

MS2: {ID_C, ID_{POS}, AMT, T_{POS}, TID, N_{POS}, LOC_{POS}, {MS1}SK_{CB}}

Step 3: After receiving {MS2}SK_{POSB} from the POS, Bank decrypts {MS2}SK_{POSB} and recovers MS2. Bank comes to know about the Customer ID (ID_C) from the MS2 and decrypts {MS1}SK_{CB} and recovers MS1. Bank compares AMT, ID_{POS}, and LOC in MS1 and MS2, checks the timestamps in MS1 and MS2; if all the checks are successful, it then transfers the AMT in the merchant's account.

1) SECURE EVIDENCE PRESERVATION MODULE (SEPM)

SEPM is a part of the Payment Gateway (PG); its function is to collect and store the evidence in the evidence repository. PG collects the evidence from transaction data, registry logs, and timestamps with the Network Time Protocol (NTP) from network logs. It stores the evidence in the repository according to transaction identity.

2) SECURE PROVENANCE MODULE (SPM)

A provenance-aware storage system (PASS) provides search for provenance as it a depository which manages the storage. PASS helps SPM in finding and storing the evidence.

V. FORMAL VERIFICATION OF THE PROTOCOL

A. FORMAL VERIFICATION OF THE PROTOCOL USING BAN LOGIC

A security protocol exchanges messages which are encrypted using cryptographic mechanisms (Muhammad *et al.*, 2006) [12]. BAN logic [13], [14] analyzes the security of the protocol.

Assumptions for the analysis and verification of the proposed protocol

(a) Assumptions about keys and secrets:

'X' is a set of entities having {C, POS, and B}. CA issues certificates to all the entities involved in the system, and all the entities have their keys (AS1, AS2).

Algorithm 1 Generation and Issuance of X. 509 Certificates to the Customer by the CA

Step 1: MNO gets UICC from a UICC manufacturer containing an OS certificate and chip certificate (both issued by the CA) and CA's certificate.

Step 2: MNO verifies both the certificates; if the certificates' verification is successful, it allocates UICC to its customers after verifying its credentials.

IF Verification (OS certificate and Chip certificate)

UNSUCCESSFUL {

/ Both OS and Chip certificates are compromised*/*

Go to Step 3)

Else {

/ allocates UICC to its customers after*

successful verification of customer's credentials/*

Exit)

Step 3: Enrolment Activation Code (EAC) is issued by the CA to the customer based on the recommendation of MNO.

Step 4: After generating his keys (both public key and private key), the customer sends an encrypted (with the public key of the CA) message containing his public key, along with EAC and a digital signature generated on the EAC using the private key of the customer.

Step 5: CA decrypts the received message, checks the EAC, and verifies the digital signature generated on EAC.

IF Verification (of EAC and digital signature on EAC)

UNSUCCESSFUL {

/ CA will not issue a certificate to the customer*/*

Else {

/ CA will issue X.509 certificate to the customer and*

sends the certificate URL to the customer and MNO/*

Exit)

Algorithm 2 Generation and Issuance of X. 509 Certificates to the Merchant by the CA

Step 1: Bank gets POS (with HSM) from the POS manufacturer containing an OS certificate and chip certificate (both issued by the CA) and CA's certificate.

Step 2: Bank verifies both the certificates; if the certificates' verification is successful, it allocates POS (with HSM) to the merchant after verifying its credentials.

IF Verification (OS certificate and Chip certificate)

UNSUCCESSFUL {

/ Both OS and Chip certificates are compromised*/*

Go to Step 3)

Else {

/ allocates POS (with HSM) to the merchant after*

successful verification of merchant's credentials/*

Exit)

Step 3: Enrolment Activation Code (EAC) is issued by the CA to the merchant based on the recommendation from the bank.

Step 4: After generating his keys (both public key and private key), the merchant sends an encrypted (with the public key of the CA) message containing his public key, along with EAC and a digital signature generated on EAC by the private key of the merchant

Step 5: CA decrypts the received message and checks the EAC, and verifies the digital signature generated on EAC.

IF Verification (of EAC and digital signature on EAC)

UNSUCCESSFUL {

/ CA will not issue a certificate to the merchant*/*

Else {

/ CA will issue X.509 certificate to the merchant and*

sends the certificate URL to the merchant and Bank/*

Exit)

AS1. CA believes $(\forall S \in \{C, POS \text{ and } B\} \vdash \overset{K_X}{\rightarrow} X)$

Certification Authority CA believes that all the stakeholders have their public keys to communicate.

AS2. $X \in \{C, POS \text{ and } B\}$ believes $\overset{K_{ca}}{\mapsto} CA$). All the entities in the system possess the public key and x.509 certificate of CA.

(b) Assumptions about freshness:

These assumptions specify the freshness of quantities.

AS3. POS believes freshness N_C ; if POS sees quantity N_C in a message, the POS can conclude that it is a replay message.

AS4. B believes freshness N_{POS} ; if B sees quantity N_{POS} in a message, the B can conclude that it is a replay message.

(c) Assumptions about Timeliness:

All the entities involved believe that the nonce generated by them are unique and fresh. These assumptions are about the certificate's validity and timestamping.

AS5. T_C is the timestamp generated by the C, ensuring timeliness.

AS6. T_{POS} is the timestamp generated by the POS, ensuring timeliness.

(d) Assumptions about trust:

The following assumptions are about the trust levels of all the entities.

AS7. $(\forall X, Q \in \{C, POS, \text{ and } B\})$, X believes CA controls $K_{ca} \mapsto Q$). All the entities trust the certification authority.

AS8. $\forall \text{ belief } X$, CA believes (Bank controls (X believes Y)). CA trusts the Bank that Bank to relay the Bank's beliefs.

Verification of our proposed Protocol using BAN logic:

Step 1: C \rightarrow POS: {MS1}SK_{CB}

MS1: {ID_C, ID_{POS}, AMT, PI_C, N_C T_C, LOC}

Step 2: POS \rightarrow B: {MS2}SK_{POSB}

MS2: {ID_C, ID_{POS}, AMT, T_{POS}, TID, N_{POS}, LOC_{POS}, {MS1}SK_{CB}}

Step 3: B \rightarrow POS & C: {MS3}

MS3: {TID, AMT, Success}

Step 1: C \rightarrow POS: {MS1}SK_{CB}

MS1: {ID_C, ID_{POS}, AMT, PI_C, N_C T_C, LOC}

POS decrypts the received {MS1}SK_{CB} from the assumptions **AS1, AS2, AS3 & AS5**

POS believes {MS1}SK_{CB} statement (1)

POS verifies the public key of C (**AS7**) received from C, If the verification is successful, then

POS believes C said {MS1}SK_{CB} statement (2)

POS believes fresh T_C from AS5. statement (3)

POS believes fresh N_S from AS3. statement (4)

So from the statements 1 to 4

POS believes {MS1}SK_{CB}

Step 2: POS \rightarrow B: {MS2}SK_{POSB}

MS2: {ID_C, ID_{POS}, AMT, T_{POS}, TID, N_{POS}, LOC_{POS}, {MS1}SK_{CB}}

B decrypts the received {MS2}SK_{POSB} from the assumptions **AS1, AS2, AS3, AS6 & AS7**

B believes {MS2}SK_{POSB} statement (5)

B verifies the public key of POS (**AS7**) received from POS If the verification is successful, then

B believes POS said {MS2}SK_{POSB} . statement (6)

B believes fresh T_{POS}T_G from AS6. statement (7)

Algorithm 3 Generation and Issuance of X. 509 Certificates to the Bank by the CA

Step 1: Bank gets HSM from the HSM manufacturer containing an OS certificate and chip certificate (both issued by the CA) and CA's certificate.
Step 2: Bank verifies both the certificates; if the certificates' verification is successful, it allocates HSM.

IF Verification (OS certificate and Chip certificate)

UNSUCCESSFUL {

/ Both OS and Chip certificates are compromised*/*

Go to Step 3}

Else {

/ allocates HSM to the bank after successful verification by the bank*/*

Exit}

Step 3: Bank sends its credentials to the CA; after successful verification of the Bank's credentials, the CA generates and issues enrolment activation code (EAC) to the Bank.

Step 4: After generating his keys (both public key and private key), Bank sends an encrypted (with the public key of the CA) message containing his public key, along with EAC and a digital signature generated on EAC by the private key of the Bank.

Step 5: CA decrypts the received message and checks the EAC, and verifies the digital signature generated on EAC.

IF Verification (of EAC and digital signature on EAC)

UNSUCCESSFUL {

/ CA will not issue a certificate to the Bank*/*

Else {

/ CA will issue X.509 certificate to the Bank and sends the certificate URL to the*

Bank/*

Exit}

B believes fresh N_{POS}N_G from AS4. statement (8)
So, from statements 5 to 8

B believes {MS2}SK_{POSB}

Step 3: B \rightarrow POS & C: {MS3}

MS3: {TID, AMT, Success}

Bank sends SMS confirmation message MS3 containing the transaction's outcome to the POS and Customer without encryption.

B. FORMAL VERIFICATION OF THE PROTOCOL USING THE SCYTHYR TOOL

Proposed protocol is written in Security Protocol Description Language (SPDL); SPDL is a language for the Scyther simulation tool [15] & [16]; it verifies the security of protocols. This tool defines the roles of the entities in our proposed system. All the entities involved in the framework experience three types of attacks; the first attack is integrity attack, the second is authentication attack and the last one is confidentiality attack.

Scyther is tool used in verifying, falsifying, and analyzing the security properties of a protocol. Table 2 maps security objectives with security properties. Table 3 shows the parameters used in the Scyther verification tool. Table 4 shows the outcome of automated security claims using the Scyther verification tool. It is the only tool which has the ability to verify multi-protocol attacks. Appendix presents the code used in NSPMT (NFC based Secure Protocol for Mobile Transaction) in SPDL.

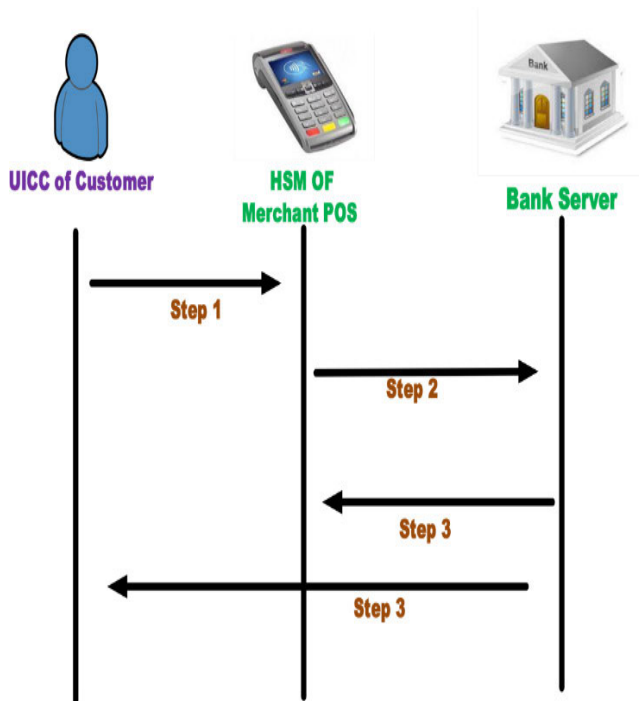


FIGURE 4. Steps involved in the proposed mobile payment protocol.

Attack Model: In our proposed system, we use Secure Elements in Customer (C), HSM in POS of Merchant (M), and Bank (B), so these devices cannot tamper, and the messages transmitted are encrypted, so all the security properties are ensured thereby, ensuring the end to end security.

VI. SECURITY ANALYSIS

This section provides the security analysis of the proposed protocol. Table 5 shows the comparative analysis of NSPMT with the related work.

- Confidentiality:** Data exchanged among the participants in the framework are encrypted using session keys thereby ensuring confidentiality.
- Mutual authentication:** WPKI is a part of both the device and MPA, which authenticates the entities using certificates. Bank personalizes Payment Applications (PA) in the Customer (C) and Merchant (M), i.e., the Bank shares a separate symmetric key with Customer (C) and Merchant (M), thereby ensuring mutual authentication.
- Integrity:** Intruder will not be able to access or modify the messages. In addition to this, the encrypted message also contains timestamps and nonce, ensuring timeliness and uniqueness properties. So, the intruder cannot modify the messages, thereby ensuring the integrity of the exchanged messages.
- Accountability:** Figure 4 depicts the steps involved in the protocol containing all the entities involved; PG (Payment Gateway) ensures accountability property, collecting evidence from the Bank. The proposed framework implements WPKI. Bank updates the MPA

of the Customer and PPA of the POS Over The Air (OTA). PG maintains the Evidence Repository (ER) and Provenance Repository (PR), ensuring accountability property. So the proposed protocol provides accountability.

- Defense in Depth:** Our proposed framework incorporates protection in-depth at the SE level and payment application level. WPKI provides application security, and the TLS protocol provides communication security. If the symmetric key is compromised, bank updates the symmetric key in the payment application.
- Overcomes Heartbleed and ROBOT Vulnerabilities:** Heartbleed, and the recent ROBOT [9], [10]. Our proposed mobile payment system uses newer versions of TLS certificates signed by the CA. So our proposed mobile payment system overcomes these vulnerabilities.
- Fake Terminal and Mobile Application:** An intruder cannot reverse engineer MPA and PPA as both (MPA and PPA) overcomes reverse engineering attacks by binary code obfuscation, flow relocation, stripping debugging information, and by encrypting strings and resources in the code, in addition to these Bank enforces Self-Signing Restriction on MPAs and PPAs and codes of these applications is attested by the Bank’s private key, so our proposed mobile payment framework overcomes reverse-engineering attacks.
- Tampering Configuration:** The workflow of the configuration file in MPA will be modified, so this attack will not be fruitful in our proposed mobile payment system as both MPA and PPA overcome this attack by flow relocation, stripping debugging information, and by encrypting strings and resources in the code, in addition to these Bank enforces Self-Signing Restriction on MPAs and PPAs and codes of these applications is attested by the Bank’s private key, so our proposed mobile payment framework overcomes tampering configuration attack.
- RAM Scraping:** This attack is also known as memory scraping or memory parsing attack, which retrieves payment information and symmetric keys from the memory of MPA. RAM Scraping or Memory Parsing attack will not be successful from the SE or MPA of either the customer’s smartphone or the HSM of the POS as SE and HSM are tamper-resistant. At the same time, MPA and PPA adopt WBC.

TABLE 2. Mapping between security objectives with security properties.

ROLE	Weak-agree	Secrecy	Alive	Nisynch	Niagree	Comment
Customer (C)	Pass	Pass	Pass	Pass	Pass	Pass
Merchant (M)						
Bank (B)						

TABLE 3. Setting parameters for the Scyther verification tool.

Verification Parameters		Advanced Parameters	
Title	Value	Title	Value
Maximum number of runs	100	Search Pruning	Find all attacks
Matching Type	Typed matching	Maximum number of patterns per claim	100

TABLE 4. Mapping between security claims and scyther security services.

Claims of Security	Scyther security services that verify claims
Authentication property	Alive, WeakAgree, NiAgree
Detecting replay the attack, Man In The Middle attack and reflection attack	NiSynch
Confidentiality of the messages exchanged	claim_B1 (B, Secret, SKcb); claim_B2 (B, Secret, SKposb);
Non-repudiation property	Agreement (NiAgree, NiSynch) between the entities, Aliveness, and Authenticity using WeakAgree

- 10) **Dishonest Merchant:** If the Merchant tries to cheat the Bank or Customer by overspending or double-spending the customer’s payment information, he will not be able to do so as the customer encrypts the message received with a symmetric key shared between the Customer and Bank. If the merchant tries to reuse the customer’s message, he will not be successful because it contains timestamps and nonce generated by the customer.
- 11) **Transaction Sniffing:** Transaction sniffing is not possible in our proposed mobile payment protocol as our proposed protocol ensures communication and application security.
- 12) **POS Security:** HSM is a tamper-resistant hardware device that protects the merchants’ credentials, ensuring protection against hardware attacks. POS’s HSM hosts PPA, PPA contains a symmetric key shared with the Bank. HSM securely stores its credentials (private key) and cannot be accessed by unauthorized entities.
- 13) **Payment Secrecy:** WBC and symmetric keys provide payment secrecy.
- 14) **Multi-Protocol Attack:** It is a type of attack in which one protocol interferes with the functioning of the other protocol. This attack will not be successful in our proposed system as we have successfully verified our proposed protocol using the Scyther tool.
- 15) **Man-in-The Middle Attack:** Data exchanged among the participants in the framework are encrypted using session keys, in addition to this message also contains timestamps and nonce thereby overcoming Man In The Middle Attack.
- 16) **Replay Attack:** Our proposed framework overcomes the Replay attack as the participants in the framework

are encrypted using session keys, in addition to this message also contains timestamps and nonce.

- 17) **Impersonation Attack:** Our proposed system overcomes impersonation attacks. The attacker fails to generate session keys.
- 18) **Parallel Session Attack:** Intruder will not be successful in starting a parallel session in our proposed framework. The Bank establishes a secure tunnel based on TLS protocol and by certificate pinning. In addition to this, encryption ensures application security using the AES algorithm. So, the Attacker/Intruder cannot have a new parallel session in our proposed framework. In addition to these, Scyther verifies our proposed protocol. So, our proposed framework overcomes or withstands parallel session attacks.
- 19) **Physically Stolen Device Attack:** If an adversary steals a smartphone or POS, he will not be able to extract customers’ or merchant’s credentials as the smartphone does not store Customer’ information.
- 20) **Resistance against Unauthorized Key Computation:** Attacker will not be able to compute session keys as the attacker does not have private keys of the Bank, Customer, and POS.
- 21) **Resistance Against Stolen Verifier Attack:** An intruder cannot get any relevant information or verifier from MPA, PPA, SE, and HSM as these implement WPKI and WBC. So our proposed system is safe against stolen verifier attack.
- 22) **DoS and DDoS Attacks:** DOS attacks are detected by the bank using change-point detection, activity profiling and wavelet-based signal analysis detection techniques and moreover the Bank only communicates with the legitimate subscribers. Bank only establishes a secure connection with the legitimate subscribers.
- 23) **Permanent Denial of Service (DoS) Attacks:** These attacks are also known as “Phlashing” attacks causing permanent damage to the hardware. Using the “Bricking a system” method, an attacker sends fraudulent hardware updates to bring down the servers. This attack will not be fruitful in our proposed system as the Bank only establishes a secure connection with the legitimate subscribers.
- 24) **Resists Stolen Smart card:** If the intruder steals the SE’s of the Customer and Merchant, the intruder cannot use the SE. Tampering SE is not possible, and it does not contain any information. So our proposed system resists stolen smart card attack;
- 25) **Outsider attack:** The intruder cannot read the transmitted messages as the messages are in encryption form. So our proposed protocol overcomes outsider attack.
- 26) **Insider attack:** Assume that a disgruntled employee of Merchant or Bank tries to extract shared symmetric keys from the payment application. He will not succeed as the WBC protects symmetric keys.

TABLE 5. Comparative analysis of NSPMT with related work.

Protocols \ Features	[1]	[2]	[3]	[4]	[5]	[6]	[23]	[29]	[30]	NSPMT (Our Proposed)
Confidentiality	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Mutual Authentication	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓
Integrity	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Accountability	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
Defense in Depth	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
Overcomes Heartbleed & ROBOT vulnerabilities	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
Fake Terminal & Mobile Application	✗	✗	✗	✗	✗	✗	✓	NA	✗	✓
Tampering Configuration	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
RAM Scrapping	✗	✗	✗	✗	✗	✗	✗	NA	✗	✓
Dishonest Merchant	✗	✓	✗	✗	✗	✗	✓	✗	✗	✓
Transaction Sniffing	✗	✗	✗	✗	✗	✗	✓	✗	✗	✓
POS Security	✗	✗	✗	✗	✗	✗	✓	NA	✗	✓
Payment Secrecy	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Multi-Protocol Attacks	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓
MITM Attack	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Replay Attack	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Impersonation Attack	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Parallel Session Attack	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
Physically Stolen Device Attack	✗	✗	✗	✓	✗	✗	✗	✗	✗	✓
Resistance Against Unauthorized Key Computation	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
Resistance Against Stolen Verifier Attack	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
DOS & DDOS Attack	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
Permanent DOS Attack	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
Resists Stolen Smart Card Attack	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
Outsider Attack	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓
Insider Attack	✓	✓	✓		✗	✗	✗	✗	✗	✓
Formal Verification	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓

27) **Formal Verification:** Proposed payment protocol is successfully verified using BAN logic and Scyther tool, so our protocol overcomes all the known attacks.

VII. IMPLEMENTATION AND PERFORMANCE ANALYSIS OF THE PROPOSED PROTOCOL

This section highlights the implementation details and performance analysis of the proposed protocol.

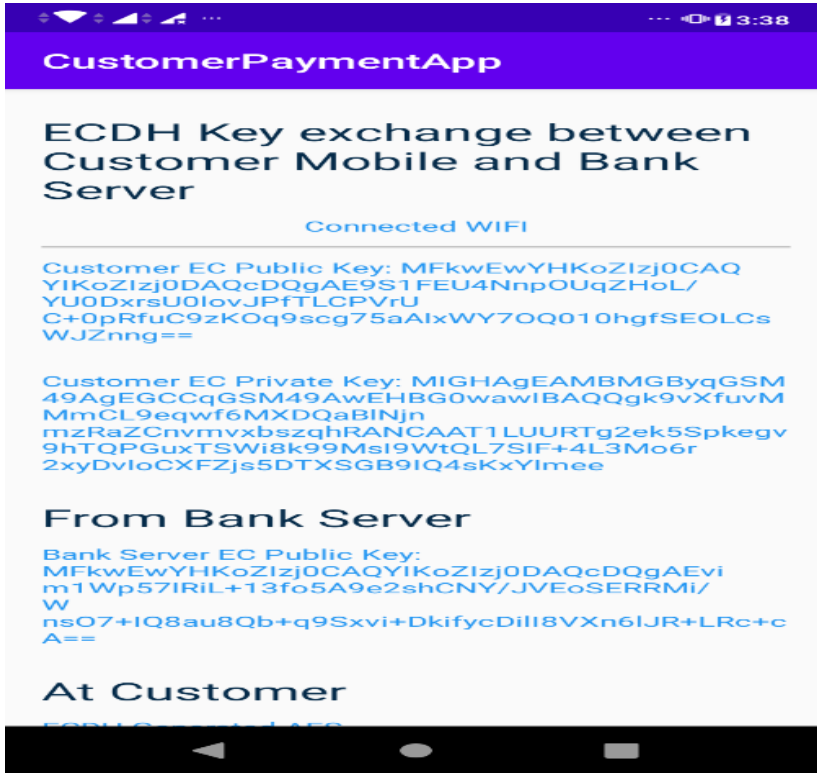


FIGURE 5. ECDH key exchange between customers' MPA.

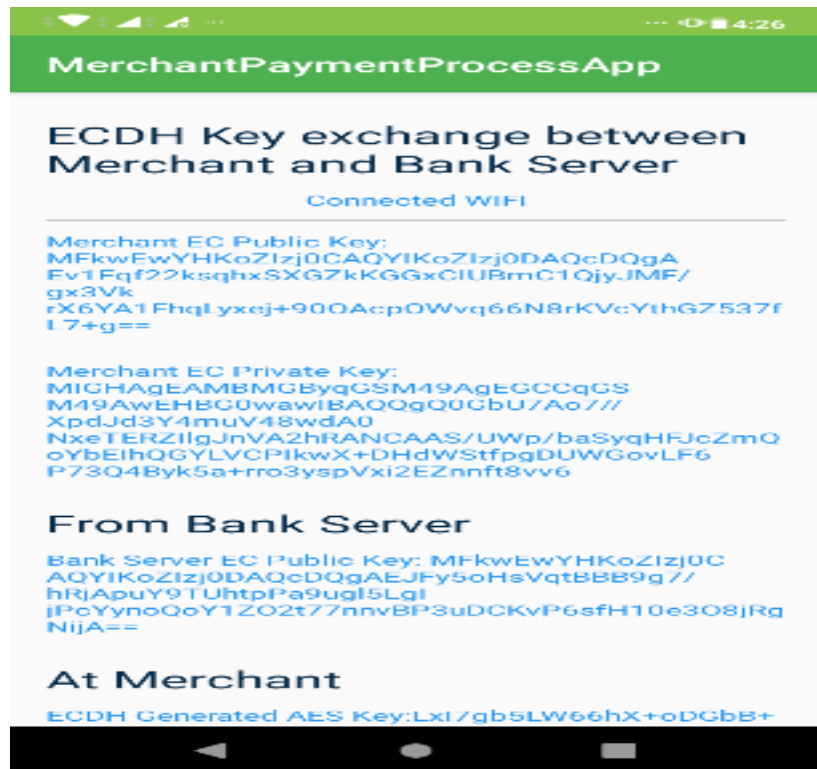


FIGURE 6. ECDH key exchange between merchant's PPA and Bank server.

A. IMPLEMENTATION OF THE PROPOSED PROTOCOL

We Implemented ECDH Key exchange and AES encryption for proximity payments using Kotlin language in Android

Studio (Kotlin interoperates fully with Java). There are two applications MPA and PPA, in our research work. ECDSA (digest algorithm used is SHA-256), an algorithm

TABLE 6. Environmental parameters.

Environment	Parameters
Customer Mobile	NFC Enabled Android Mobile
	Snapdragon 632
	3GB RAM
Merchant's POS	Android v9.0 (minimum Android v6.0)
	NFC Enabled Android Mobile
	Snapdragon 632
Bank server	3GB RAM
	Android v9.0 (minimum Android v6.0)
	Linux CentOS 7.8.2003
	Intel i7 9700k
	4GB RAM
	80GB SSD
	Nginx Server 1.18.0
	PHP 7.2.31
MariaDB 5.5.65	
Java (OpenJDK 1.8.0_252)	

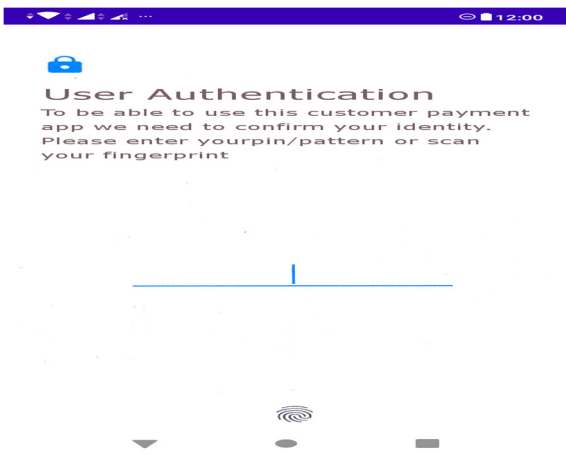


FIGURE 7. Customer authentication.

is used in generating digital signature, and the AES algorithm for encrypting messages. Kotlin language implements our protocol in real-time. Kotlin interoperates fully with Java. We created an EC key pair (NIST P-256 aka secp256r1) at customer-bank and merchant-bank by using ECDH, we created a shared AES secret key. AES with GCM (Galois/Counter Mode) used for encryption and decryption of Customer Payment Data at MPA and PPA. Table 6 shows the environmental parameters used in the implementation of the proposed protocol.

Figure 5 depicts the ECDH key exchange process between the Customer's MPA and the bank server. Using the customer's private key and Bank Server's public key, the Customer's MPA generates a shared AES key. Similarly, the bank server also generates a shared AES key by using the customer's public key and the bank server's private key. Figure 6 depicts the ECDH key exchange process between the Merchant's PPA and the bank server. Using

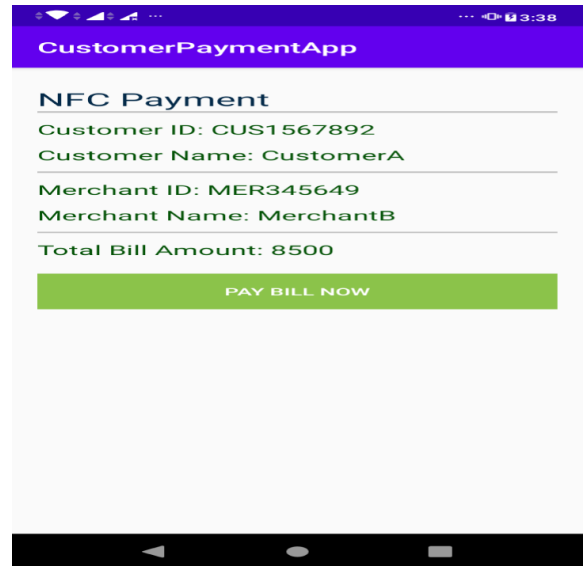


FIGURE 8. MPA showing the amount and merchant details.

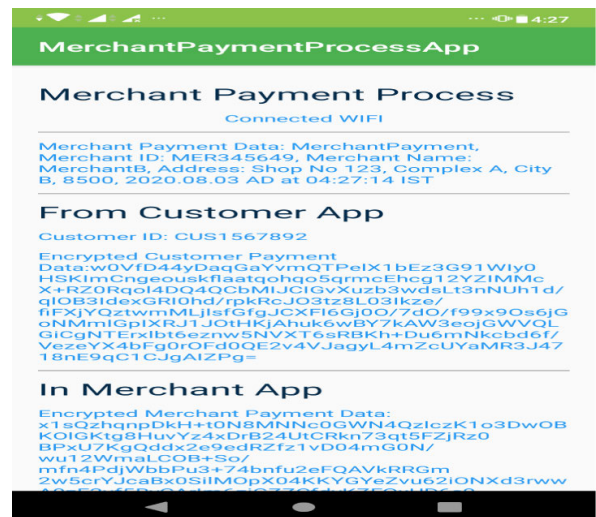


FIGURE 9. PPA showing merchant transaction and payment status.

the merchant's private key and Bank Server's public key merchant's MPA generates a shared AES key. Similarly, the bank server also generates a shared AES key using the merchant's public key and bank server's private key. Figure 7 shows the user authentication screenshot. To complete the payment, the customer needs to authenticate either through PIN or Fingerprint. Only after successful authentication, the customer is allowed to use MPA. Figure 8 shows the MPA screen containing merchant ID, merchant name, and total bill amount, which is encrypted and sent to POS via NFC. Figure 9 shows the screen containing merchant transactions, customer ID, and customer encrypted payment data. After successful payment completion at the bank server, this screen receives a successful message from the bank server, shows success status to the merchant, and sends a successful message to the customer MPA. Figure 10

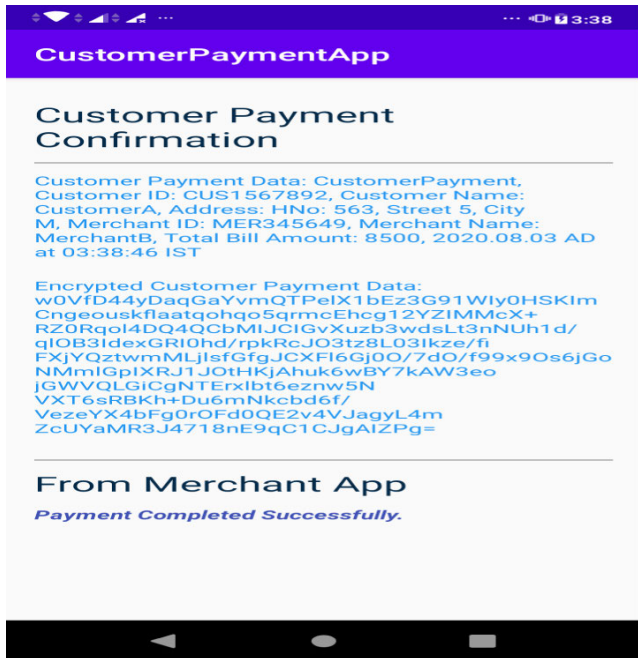


FIGURE 10. MPA showing plain and encrypted transaction and payment status.

TABLE 7. Computational costs of the proposed protocol.

Protocol	Computation cost of the Client (C) in seconds	Computation cost of Merchant (M) in seconds	Computation cost Bank (B) in seconds	Overall computation cost in seconds
Jen-Ho Yang an et al. [1]	3TS+2TH (0.3917)	2TS+1TH (0.261)	2TS (0.2606)	0.9133
Pourghomi, P et al. [2]	7TS (0.9121)	3TS (0.3903)	8TS (1.0424)	2.3448
Isaac, J et al. [3]	4 TS +4 TH (0.5228)	5 TS+2TH (0.6523)	3 TS (0.3909)	1.5660
Jen-Ho Yang a et al. [4]	5ECPM+2 TS (0.265675)	5ECPM+2TS (0.265675)	2ECPM+4TS (0.52323)	1.05458
Jen-Ho Yang et al. [5]	4ECPM+2 TS (0.26466)	3ECPM+1TS (0.133345)	1ECPM+2TS (0.261615)	0.65962
Our Proposed	1 TS (0.1303)	1TS (0.1303)	2TS (0.2606)	0.5212 Seconds

shows the plain and encrypted transaction and payment status from PPA. Our implementation details and code are here: <https://github.com/ShaikShakeelAhamad/merchant-customer-payment>

B. PERFORMANCE ANALYSIS OF THE PROPOSED PROTOCOL

Table 8 presents the computational cost analysis of the proposed scheme against related schemes. The notations used in the table are TH and TS, which denotes the complexity of time for computing the one-way hash function (TH) and symmetric encryption/decryption (TS) operation.

The performance analysis shows the effectiveness of the proposed protocol as it utilizes one-way hash and symmetric encryption/decryption functions, both of which are the least

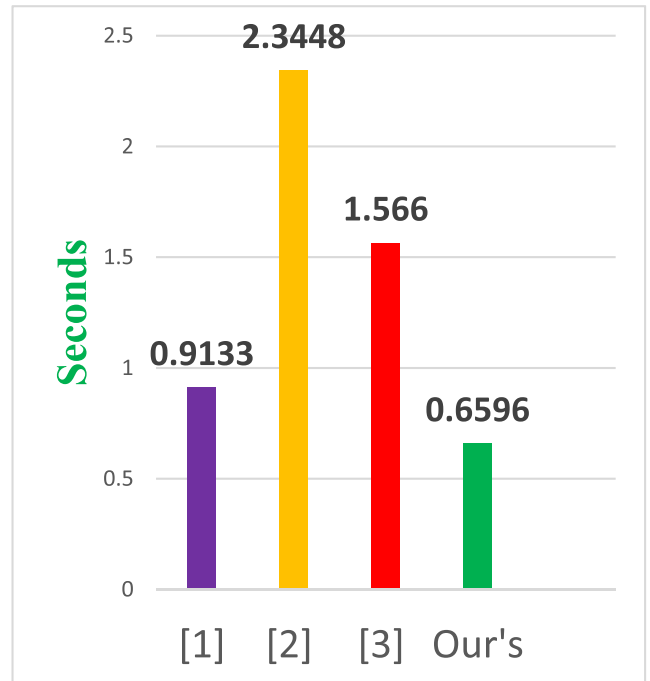


FIGURE 11. Bar chart for computational costs of the proposed protocol.

TABLE 8. Energy costs for the proposed protocol.

Protocol	Energy cost for Client (C) in Micro Joules	Energy cost for Merchant (M) in Micro Joules	Energy cost for Bank (B) in Micro Joules	Overall Energy cost in Micro Joules
Jen-Ho Yang a et al. [1]	3TS+2TH (5.15)	2TS+1TH (3.18)	2TS (2.42)	10.75
Pourghomi, P et al. [2]	7TS (8.47)	3TS (3.63)	8TS (9.68)	21.78
Isaac, J et al. [3]	4 TS +4 TH (7.88)	5 TS+2TH (7.57)	3 TS (3.63)	19.08
Jen-Ho Yang a et al. [4]	5ECPM+2 TS (2895.17)	5ECPM+2 TS (2895.17)	2ECPM+4 TS (1161.94)	6952.28
Jen-Ho Yang et al. [5]	4ECPM+2 TS (2316.62)	3ECPM+1TS (1736.86)	1ECPM+2TS (580.97)	4634.45
Our Proposed	1SyE (1*1.21=1.21 μJ)	1SyE (1*1.21=1.21 μJ)	2SyE (2*1.21=2.42 μJ)	4.84 μJ

expensive than other cryptographic functions. As presented in [5], the time complexities (in seconds) are TH = 0.0004 and TS = 0.1303. One ECPM (Elliptic Curve Point Multiplication) is equal to 0.001015 seconds from [22]. Clearly, our proposed work shows better performance (see Figure 11).

Table 8 compares our proposed protocol with the related works in terms of consumption of energy. According to [5] the consumption of energy to generate AES encryption/decryption (ES) is 1.21 Micro Joules/byte and for generating hash code (EH) using SHA-1 algorithm is 0.76 Micro Joules. The energy consumption of One ECPM (Elliptic Curve Point Multiplication) is equal to 578.55 Micro Joules from [22]. As shown in the table, our work is better than other works (Figure 12).

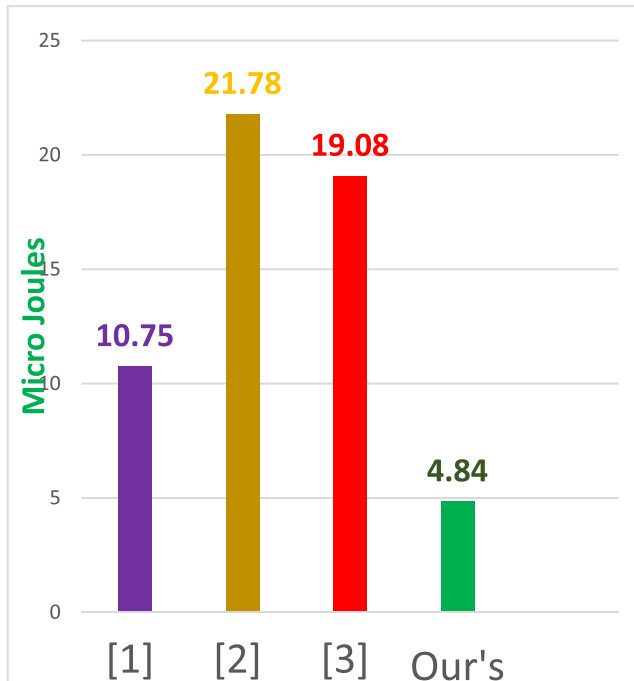


FIGURE 12. Bar chart for energy costs of the proposed protocol.

VIII. CONCLUSION

This paper proposes a NFC based Secure Protocol for Mobile Transaction (NSPMT) protocol incorporating Defense in Depth approach; our Defense-in-depth approach has three levels, i.e., Defense at the hardware level, Defense at the application level, and Defense at the communication level. The proposed protocol has been successfully verified using BAN logic and using the Scyther tool. The proposed protocol overcomes all the known attacks, including multi-protocol attacks. Our proposed protocol has less energy consumption, communication cost, and computational cost than the existing works. In addition to this, the proposed system overcomes RAM scrapping attack, DOS, DDOS, and Phlashing attacks. Our proposed mobile Payment system overcomes the known mobile application vulnerabilities such as Heartbleed and ROBOT. NSPMT protocol has been successfully implemented using kotlin language in Android Studio, with two Mobile Payment Applications (MPA) and POS Payment Application (PPA), Elliptic Curve Digital Signature Algorithm (ECDSA) is used in generating and verifying digital signatures and Advanced Encryption Standard (AES) with GCM (Galois/Counter Mode) mode is used for encryption and decryption of Customer Payment Data at MPA and PPA. NSPMT protocol ensures all the security properties.

APPENDIX

*/*A Novel NFC based Secure Protocol for Mobile Transactions (NSPMT)*/*

```
const pk: Function;
secret sk: Function;
inversekeys (pk, sk);
```

```
usertype Timestamp;
usertype TID, CID, POSID, TC, NC, AMT, LOCC, PIC,
LOCPOS, TPOS, NPOS;
// Protocol description
protocol NSPMT(C, POS, B)
{
  role C
  {
    const NC: Nonce;
    const SKcb: SessionKey;
    const SKposb: SessionKey;
    /*Authentication and Key Agreement Protocol*/
    send_1 (C, POS, {CID, POSID, TC, NC, AMT, PIC,
LOCC} SKcb);
    claim_C1 (C, Secret, SKcb);
    claim_C2 (C, Secret, NC);
    claim_C3 (C, Niagree);
    claim_C4 (C, Nisynch);
  }
  role POS
  {
    const NPOS: Nonce;
    var NPOS: Nonce;
    const SKcb: SessionKey;
    const SKposb: SessionKey;
    read_1 (C, POS, {CID, POSID, TC, NC, AMT, PIC,
LOCC} SKcb);
    send_2 (POS, B, {CID, POSID, AMT, TPOS, NPOS, TID,
LOCP} SKposb,
{CID, POSID, TC, NC, AMT, PIC, LOCC} SKcb);
    claim_POS1 (POS, Secret, SKcb);
    claim_POS2 (POS, Secret, SKposb);
    claim_POS3 (POS, Secret, NPOS);
    claim_POS4 (POS, Niagree);
    claim_POS5 (POS, Nisynch);
  }
  role B
  {
    const NB: Nonce;
    var NPOS: Nonce;
    var NC: Nonce;
    const SKcb: SessionKey;
    const SKposb: SessionKey;
    read_2 (POS, B, {CID, POSID, AMT, TPOS, NPOS, TID,
LOCP} SKposb,
{CID, POSID, TC, NC, AMT, PIC, LOCC} SKcb);
    claim_B1 (B, Secret, SKcb);
    claim_B2 (B, Secret, SKposb);
    claim_B3 (B, Niagree);
    claim_B4 (B, Nisynch);
  }
}
// An untrusted agent, with the compromised key
const e: Agent;
untrusted e;
compromised sk(e);
```

ACKNOWLEDGMENT

The authors gratefully acknowledge the editor and the reviewers' helpful comments and suggestions, which have improved the presentation. Shaik Shakeel Ahamad would like to thank the Deanship of Scientific Research at Majmaah University for supporting this work under Project No. R-2021-313.

COMPETING INTERESTS

The authors have declared that no competing interests exist.

REFERENCES

- J.-H. Yang and P.-Y. Lin, "A mobile payment mechanism with anonymity for cloud computing," *J. Syst. Softw.*, vol. 116, pp. 69–74, Jun. 2016.
- P. Pourghomi, M. Q. Saeed, and G. Ghinea, "A secure cloud-based NFC mobile payment protocol," *Int. J. Adv. Comput. Sci. Appl.*, vol. 5, no. 10, pp. 24–31, 2014.
- J. T. Isaac and S. Zeadally, "An anonymous secure payment protocol in a payment gateway centric model," *Proc. Comput. Sci.*, vol. 10, pp. 758–765, Jan. 2012.
- J.-H. Yang and C.-C. Chang, "An efficient three-party authenticated key exchange protocol using elliptic curve cryptography for mobile-commerce environments," *J. Syst. Softw.*, vol. 82, no. 9, pp. 1497–1502, Sep. 2009.
- J.-H. Yang, Y.-F. Chang, and Y.-H. Chen, "An efficient authenticated encryption scheme based on ECC and its application for electronic payment," *Inf. Technol. Control*, vol. 42, no. 4, pp. 315–324, Dec. 2013.
- J.-H. Yang, "An electronic transaction mechanism using mobile devices for cloud computing," *Wireless Pers. Commun.*, vol. 94, no. 3, pp. 713–724, Jun. 2017.
- M. Beunardeau, A. Connolly, R. Geraud, and D. Naccache, "White-box cryptography: Security in an insecure environment," *IEEE Secur. Privacy*, vol. 14, no. 5, pp. 88–92, Sep. 2016.
- S. Kungpisdan, B. Srinivasan, and P. D. Le, "Lightweight mobile credit-card payment protocol," in *Proc. INDOCRYPT*, vol. 2904, New Delhi, India, 2003, pp. 295–308.
- H. Böck, J. Somorovsky, and C. Young, "Return of Bleichenbacher's Oracle threat (ROBOT)," in *Proc. 27th USENIX Conf. Secur. Symp.*, Baltimore, MD, USA: USENIX Assoc., Aug. 2018, pp. 15–17.
- The Heartbleed Bug*. Accessed: Aug. 9, 2021. [Online]. Available: <http://heartbleed.com>
- Introduction to Secure Elements*. Accessed: Feb. 9, 2021. [Online]. Available: <https://globalplatform.org/wp-content/uploads/2018/05/Introduction-to-Secure-Element-15May2018.pdf>
- S. Muhammad, Z. Furqan, and R. K. Guha, "Understanding the intruder through attacks on cryptographic protocols," in *Proc. 44th Annu. Southeast Regional Conf. (ACM-SE)*, 2006, pp. 667–672.
- M. Abadi, M. Burrows, C. Kaufman, and B. Lampson, "Authentication and delegation with smart-cards," *Sci. Comput. Program.*, vol. 21, no. 2, pp. 93–113, Oct. 1993.
- M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *ACM Trans. Comput. Syst.*, vol. 8, no. 1, pp. 18–36, 1990.
- C. J. F. Cremers, "Scyther: Semantics and verification of security protocols," Ph.D. dissertation, Dept. Comput. Sci., Eindhoven Univ. Technol., Eindhoven, The Netherlands, 2006, doi: 10.6100/IR614943.
- C. Cremers and P. Lafourcade, "Comparing state spaces in automatic security protocol verification," Ph.D. dissertation, Dept. Comput. Sci., Eindhoven Univ. Technol., Eindhoven, The Netherlands, 2007.
- Payment Card Industry (PCI) Hardware Security Module (HSM)*. Accessed: Aug. 9, 2021. [Online]. Available: <https://www.pcisecuritystandards.org/documents/PCI%20HSM%20Security%20Requirements%20v1.0%20final.pdf>
- RFC 5636 Traceable Anonymous Certificate*. Accessed: Aug. 8, 2021. [Online]. Available: <https://tools.ietf.org/html/rfc5636>
- NFC-Ready POS Terminals to Hit 8 10 Globally by 2022*. Accessed: Aug. 15, 2021. [Online]. Available: <https://www.electran.org/publication/transactiontrends/nfc-ready-pos-terminals-to-hit-8-in-10-globally-by-2022>
- L. Xu and F. Wu, "Cryptanalysis and improvement of a user authentication scheme preserving uniqueness and anonymity for connected health care," *J. Med. Syst.*, vol. 39, no. 2, pp. 1–9, Feb. 2015, doi: 10.1007/s10916-014-0179-x.
- S. L. Javan and A. G. Bafghi, "An anonymous mobile payment protocol based on SWPP," *Electron. Commerce Res.*, vol. 14, no. 4, pp. 635–660, Dec. 2014.
- M. H. Ibrahim, S. Kumari, A. K. Das, and V. Odelu, "Jamming resistant non-interactive anonymous and unlinkable authentication scheme for mobile satellite networks," *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 5563–5580, Dec. 2016.
- I. Turk, P. Angin, and A. Cosar, "RONFC: A novel enabler-independent NFC protocol for mobile transactions," *IEEE Access*, vol. 7, pp. 95327–95340, 2019, doi: 10.1109/ACCESS.2019.2929011.
- M. Obaid, Z. Bayram, and M. Saleh, "Instant secure mobile payment scheme," *IEEE Access*, vol. 7, pp. 55669–55678, 2019, doi: 10.1109/ACCESS.2019.2913430.
- S. A. Chaudhry, M. S. Farash, H. Naqvi, and M. Sher, "A secure and efficient authenticated encryption for electronic payment systems using elliptic curve cryptography," *Electron. Commerce Res.*, vol. 16, no. 1, pp. 113–139, 2016, doi: 10.1007/s10660-015-9192-5.
- J.-N. Luo and M.-H. Yang, "EMV-compatible offline mobile payment protocol with mutual authentication," *Sensors*, vol. 19, no. 21, p. 4611, Oct. 2019.
- M.-H. Yang, "Security enhanced EMV-based mobile payment protocol," *Sci. World J.*, vol. 2014, pp. 1–19, Jan. 2014, doi: 10.1155/2014/864571.
- C. Thammarat and W. Kurutach, "A secure fair exchange for SMS-based mobile payment protocols based on symmetric encryption algorithms with formal verification," *Wireless Commun. Mobile Comput.*, vol. 2018, pp. 1–21, Jul. 2018, doi: 10.1155/2018/6953160.
- C. Thammarat and W. Kurutach, "A lightweight and secure NFC-based mobile payment protocol ensuring fair exchange based on a hybrid encryption algorithm with formal verification," *Int. J. Commun. Syst.*, vol. 32, no. 12, Aug. 2019, Art. no. e3991, doi: 10.1002/dac.3991.
- T.-K. Chang, "A secure operational model for mobile payments," *Sci. World J.*, vol. 2014, pp. 1–14, Oct. 2014, doi: 10.1155/2014/626243.
- F. S. M. Tafti, S. Mohammadi, and M. Babagoli, "A new NFC mobile payment protocol using improved GSM based authentication," *J. Inf. Secur. Appl.*, vol. 62, pp. 1–10, Nov. 2021.
- M. Baza, N. Lasla, M. M. E. A. Mahmoud, G. Srivastava, and M. Abdallah, "B-ride: Ride sharing with privacy-preservation, trust and fair payment atop public blockchain," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 1214–1229, Apr. 2021.



SHAIK SHAKEEL AHAMAD received the Ph.D. degree in computer science (cyber security) from the University of Hyderabad and the Institute for Development and Research in Banking Technology (IDRBT), Hyderabad, India, in the realm of secure mobile payment protocols and formal verification. He is currently working as an Assistant Professor with the Department of Information Technology, College of Computer and Information Sciences, Majmaah University, Al Majma'ah, Saudi Arabia. He is also the Certified EC Council Instructor (CEI), EC Council Certified Security Analyst (ECSA), Computer Hacking Forensic Investigator (CHFI), Certified Threat Intelligence Analyst (CTIA), and Certified Application Security Engineer (CASE)–Java. His research interests include cyber security, cloud computing, secure mobile payments, blockchain technology, secure smart grids, and security and privacy in healthcare 4.0. He is serving as a review committee member for many ISI indexed journals.

...