# Mobile Payment Protocol 3D by Using Cloud Messaging

Mohammad Vahidalizadehdizaj and Avery Leider
Seidenberg School of CSIS, Pace University, Pleasantville, New York

***Abstract*—The popularity of mobile platforms makes them a good candidate for electronic payment. However, there are challenges in this field such as privacy protection, security, the bandwidth limitations of mobile networks, and the limited capabilities of mobile devices. Traditional e-commerce payment protocols were designed to keep track of traditional flows of data, are vulnerable to attacks and are not designed for mobile platforms. Also, 3D Secure, an extra security layer in modern payment methods (which is mainly to prevent fraud where the credit card is not present), is not suitable for mobile platforms because of issues like the difficulty of viewing the authentication pop-up window on a mobile device. In this paper, we propose a new private mobile payment protocol based on a client centric model that utilizes symmetric key operations. Our protocol reduces the computational cost of the Diffie-Hellman key agreement protocol by using the algebra of logarithms instead of the algebra of exponents, achieves proper privacy protection for the payer by involving mobile network operators and generating temporary identities, avoids repudiation attacks by utilizing digital signatures, avoids replay attacks by using random time-stamp generated numbers, and provides better and safer customer experience by utilizing cloud messaging instead of text messaging and pop-up windows in its extra layer of security (3 domain authentication).**

***Keywords*—E-commerce, M-commerce, Mobile Commerce, Mobile Payment, Privacy Protection, Non-repudiation, Replay Attack, 3D Secure, Verified by Visa, MasterCard SecureCode, American Express SafeKey, JCB International as J/Secure, Diffie-Hellman.***

## I. Introduction

E-commerce is any financial transaction over the Internet. Most of the time, the payer uses his credit card in this process. An e-commerce transaction involves the purchaser or card holder, the merchant, the purchaser's credit card issuer (bank), the merchant's acquirer (bank), and the certification authority for supporting secure transaction execution [1]. Most of these protocols are using a key agreement protocol for establishing a secure connection between the engaging parties [3].

Mobile commerce (m-commerce) is e-commerce activities conducted via mobile platforms. Principles of m-commerce are the same as e-commerce plus mobile network operator. Moreover, most e-commerce protocols are based on public key cryptography that is not efficient in mobile and wireless networks [2]. Some of these protocols are keeping the credit card's information on the mobile devices or using this information in transactions without proper protection. This is why they are vulnerable to attacks [4].

Mobile devices like smart phones and tablets are becoming very popular [5]. Most of these devices are light, easy to carry, and convenient to use. These devices are compatible with mobile networks that are widely available in outdoor space. Growth of m-commerce sales continues to be rapid even with the challenges that m-commerce faces like slow download times. Forrester predicted 11 percent (of whole e-commerce) growth in m-commerce between 2016 and 2020. Currently m-commerce has 35 percent of e-commerce transactions. Forrester predicted that m-commerce will be 49 percent of e-commerce in 2020. This amount is 252 billion dollars in sales [16].

## II. Background

In this section, we review existing related protocols. We review Diffie-Hellman as the popular key agreement protocol in payment protocols. We review SET, iKP, KSL, and 3D Secure protocols [2], [14]. Diffie-Hellman is the most famous key agreement protocol. Its calculations are based on algebra of exponents and modulus in arithmetic. Goal of this protocol is to generate a shared session key between two parties [3].

SET defines an open encryption and security specification. This protocol is designed to protect credit card transactions over the Internet. Initial version of SET emerged in a call for security standards by MasterCard and Visa in February 1996 [3]. SET has some problems. In the SET protocol, the cardholder is not protected from dishonest merchants that charge more than their advertised price or hackers who put up an illegal website to collect credit card information [2]. Besides, the merchant is not protected from dishonest customers who provide invalid credit card numbers or claim refund without any real cause [1]. In the SET protocol, the merchant is more vulnerable to fraud, since legislation protects the customers in most of the countries in the world [15].

IBM developed the iKP ($i = 1, 2, 3$) family of protocols. These protocols are designed to implement credit card transactions between customer and merchant [2]. Each one of the members of the family differs from the others from the aspect of level of complexity and security. iKP is the direct ancestor of SET. These protocols have been used in

Mohammad Vahidalizadehdizaj is a PhD candidate in Computer Science Department, Pace University, New York, NY. Email: m.vahidalizadeh@gmail.com

Avery Leider is an assistant adjunct professor in Computer Science Department, Pace University, New York, NY. Email: aleider@pace.edu

Internet since 1996. iKP protocols are unique because of longevity, security, and relative simplicity of the underlying mechanisms. These protocols are based on public key cryptography. They differ from each other in the aspect of the number of principals who possess the public key pairs [1], [2].

KSL is a payment protocol for e-commerce in fixed networks like the Internet. This protocol is not proper for mobile platforms because of its heavy computation and communication costs [3]. The idea of KSL is to reduce the number of people who possess their own key pair in SET protocol. In this protocol all principals except the customer should have their own certificates, resulting in lighter client side computation[1]. KSL protocol is a nonce based protocol. KSL is an alternative of Kerberos. The drawback of Kerberos is that it uses timestamps. So, it requires at least one loose synchronization between timestamp generator and participants. Principals of KSL protocol are customer, merchant, payment gateway, and financial service provider [5].

3D Secure is an extra security layer for online credit and debit card transactions in order to prevent the fraud of the credit card not being present. Arcot Systems was first to develop this protocol. The name 3D came from 3 domains: the acquiring domain, the issuer domain, and the interoperability domain [14]. There are some vulnerability issues in 3DS. In 3DS, there is a pop-up window or inline frame that is coming from a source that is not a familiar domain. It is very hard for the customer to find out if this window is a phishing scam or if it is coming from the bank. Both the man in the middle attack and phishing scam is possible in this pop-up window step. Mobile users may experience issues to see the pop-up or iframe [14]. Sometimes a 3D Secure confirmation code is required. If the 3DS implementation confirmation code is sent as a text message, customer may be unable to receive it based on the country that he is in. Also, this may cause trouble for the people who change their cellphone number regularly [14].

## III. Proposed Protocol

In this section, we introduce a new payment protocol that is better for mobile platforms. Our mobile payment protocol is based on client centric model. In this section, we introduce an improved version of Diffie-Hellman key agreement protocol. Our improved key agreement protocol is based on the algebra of logarithms and modulus arithmetic. Our intention is to make shared key generation process satisfactory for mobile platforms by reducing computational cost. We also try to reduce the size of temporary results in our computations. You can see our key agreement protocol in table I.

You can see the proof of correctness of our key agreement protocol in below. Suppose, there exists two parties A and B. We compare the key that is generated by A to the key that is generated by B. If they are equal, our key agreement protocol is correct. In this proof, we use rules of algebra of logarithms and algebra of modulus arithmetic. We suppose $a_i$ is $a$ mod $p_i$ and $b_i$ is $b$ mod $p_i$. Integer $a$ is represented by r-tuple $(a_1, ..., a_r)$. In this kind of representation, residues should be calculated by multiple divisions. So, $a_i = a$ mod $p_i$ $(1 \leqslant i \leqslant r)$. Then, based on theorem 2.1, we have the following [2]:

| | Action | Description |
|---|---|---|
| 1 | Alice and Bob agree on three numbers a, p, and q | q is a large prime number<br>$a = p^n$<br>p = 2, 3, ..., n<br>n, u, and v = 1, 2, 3, ..., n |
| 2 | Alice picks a secret number u | Alice's secret number = u<br>u mod q is not zero |
| 3 | Bob picks a secret number v | Bob's secret number = v<br>v mod q is not zero |
| 4 | Alice computes her public number<br>$A = ((u \bmod q) \times log_p a) mod\, q$ | Alice's public number = A<br>$= (log_p a^{(u \bmod q)}) \bmod q$ |
| 5 | Bob computes his public number<br>$B = ((v \bmod q) \times log_p a) mod\, q$ | Bob's public number = B<br>$= (log_p a^{(v \bmod q)}) \bmod q$ |
| 6 | Alice and Bob exchange their public numbers | Alice knows u, a, p, q, A, and B<br>Bob knows v, a, p, q, A, and B |
| 7 | Alice computes<br>$k_a = ((u \bmod q) \times B) \bmod q$ | $k_a = ((u \bmod q) \times (log_p a^{(v \bmod q)})) mod\, q$<br>$k_a = (log_p a^{(v \bmod q) \times (u \bmod q)}) mod\, q$<br>$k_a = (log_p a^{((u \times v) \bmod q)}) \bmod q$ |
| 8 | Bob computes<br>$k_b = ((v \bmod q) \times A) mod\, q$ | $k_b = ((v \bmod q) \times (log_p a^{(u \bmod q)})) \bmod q$<br>$k_b = (log_p a^{(u \bmod q) \times (v \bmod q)}) mod\, q$<br>$k_b = (log_p a^{((v \times u) \bmod q)}) \bmod q$ |
| 9 | By the law of algebra, Alice's $k_a$ is the same as Bob's $k_b$, or $k_a = k_b = k$ | Alice and Bob both know the secret value k |

TABLE I: Our Key Agreement Protocol

$$(a_1, ..., a_r) \times (b_1, ..., b_r) = (a_1 b_1 \bmod p_1, ..., a_r b_r \bmod p_r)$$

In our case, we only have one prime number that is $q$. In addition, based on algebra of logarithm, we have the following [4]:

$$log_b(m^n) = n log_b(m)$$

So, let's suppose we have two parties A and B. We follow our protocol to generate a key for each one of them. Then, by the rules that we just mentioned, we prove that the key that is generated for A is equal to the key that is generated for B. Besides, as you see in table II, results of the calculations in both sides (A and B) are equal.

A:
$k_a = ((u \bmod q) \times (log_p a^{(v \bmod q)})) mod\, q$
$k_a = (log_p a^{(v \bmod q) \times (u \bmod q)}) mod\, q$
$k_a = (log_p a^{((u \times v) \bmod q)}) \bmod q$

B:
$k_b = ((v \bmod q) \times (log_p a^{(u \bmod q)})) \bmod q$
$k_b = (log_p a^{(u \bmod q) \times (v \bmod q)}) mod\, q$
$k_b = (log_p a^{((v \times u) \bmod q)}) \bmod q$

As you can see the results in both sides are equal. So, Alice and Bob now have a shared key. They can use this key for symmetric encryption. Note that, this key never traveled via the network during the key generation steps. Also our middle keys never travel via the network. We kept security strength of Diffie-Hellman and reduced its computational cost. We can use our improved key agreement protocol instead of

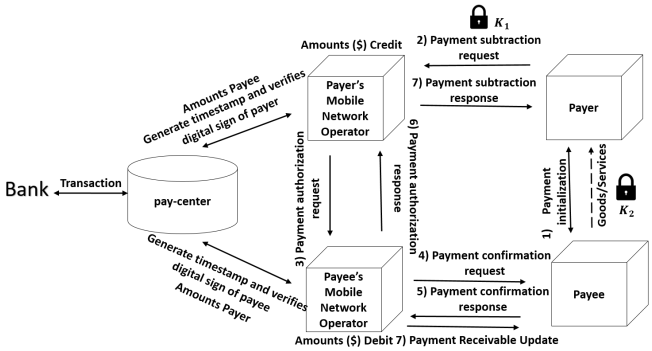


Fig. 1: Proposed mobile payment protocol

| Symbols | Descriptions |
|---|---|
| MNO | Mobile Network Operator |
| {payer, payee, payerś MNO, payeeś MNO} | A set of engaging parties, which includes Payee, Payer, Payeeś MNO and Payerś MNO |
| Pay Center | Time Stamp and Digital Sign center |
| PN | Phone Number of Party P |
| PIN | Party P selected this password identification number |
| ID | Identity of Party P, which identifies Party P to MNO |
| AI | Account Information of Party P, which including credit limit for each transaction and type of account |
| $R_1$ | Random and times-tamp generated number by Payer act as Payer's pseudo-ID |
| $R_2$ | Random and time-stamp generated number to protect against replay attack |
| $K_1$ | Shared key between payer and his mobile network operator |
| $K_2$ | Shared key between payer and payee |
| AMOUNT | Payment transaction amount and currency |
| DESC | Payment Description, which may include delivery address, purchase order details and so on |
| TID | The Identity of transaction |
| $TID_{Req}$ | Request for TID |
| $PID_{Req}$ | Request for P identity |
| Req | Request |
| MX | The message M encrypted with key X |
| H(M) | The one way hash function of the message M |
| i | Used to identify the current session key of $X_i$ and $Y_i$ |
| $K_{p\text{-}p}$ | The secret key shared between Payerś MNO and Payeeś MNO. |
| Success/Fail | The status of registration, whether success or failed |
| Yes/No | The status of transaction, whether approved or rejected |
| Received | Payment receivable update status, which may include the received payment amount |
| $Pr_P$ | Private key of party P |
| $Pu_P$ | Public key of party P |
| CK | client key: a key that is necessary for decoding $X_i$ and $Y_i$ sets on client side |
| $CK_{Req}$ | Request for client key |
| T | Current date and time |

TABLE II: Notations

Diffie-Hellman in our mobile payment protocol in order to make it suitable for mobile platforms.

Our proposed mobile payment protocol's principals are payer, payee, mobile network operator, MPI, ACS, payer's credit card issuer (bank), payee's acquirer (bank), and certification authority for supporting secure transaction execution. Our protocol works with two sets of keys. First set should be shared between payer and his mobile network operator. The second set should be shared between payee and his mobile network operator. Our protocol consists of two-sub protocols that are registration and payment protocols. Payer and payee must register with their own mobile network operators at the beginning. Payer and his mobile network operator should generate a session key by running our improved key agreement protocol. You can see our notations in table II.

The rest of this section is defining our new mobile payment protocol that we implement in seven steps. At the beginning, payer encrypts registration details such as account information, payer's identity, and his phone number with his shared key. This information should be sent to payer's mobile network operator.

payer $\Rightarrow$ payerś MNO:
$\{PN_{Payer}, ID_{Payer}, AI_{payer}\}K1$

As we mentioned earlier, there are several challenges in designing our payment protocol. One of these challenges is to prevent privacy violation of payer. Most of the current payment protocols are providing identity protection from eavesdroppers. But, they don't provide identity protection from merchant. One of our goals is to avoid possible identity or privacy violation in our payment protocol. In order to overcome the issue of privacy violation, we want to involve mobile network operators in the payment process. Besides, we want to generate temporary identity for our customers in order to provide proper privacy protection for them. We generate this temporary identity based on our customer's phone number and his password identification number. Note that, this ID will be generated after a successful authentication.

During the registration process, payer has to set his password identification number ($PIN_{Payer}$) in order to access his mobile wallet application. This implementation uses two factor authentication that is an important principle for mobile device access control. Two factor authentication authenticates users in two steps. The first step is authentication with mobile wallet application on his mobile that is something that he has. The second step is password authentication that is something that he knows. Then, $ID_{Payer}$ will be computed by hashing payer's phone number ($PN_{Payer}$) and password identification number ($PIN_{Payer}$).

$ID_{Payer}$ = $PN_{Payer}$ + Hash($PN_{Payer}$,$PIN_{Payer}$)

Then, payer's mobile network operator decodes the message with his shared key ($K_1$). Payer's mobile network operator stores necessary information into its database. If registration process is successful, payer's mobile network operator will send confirmation message to inform payer about the result. Confirmation message is also encrypted with the session key ($K_1$).

Payer's MNO $\Rightarrow$ Payer: {Success/ Failed} Encrypted with $K_1$

After registration, payer receives mobile wallet application through email or downloads it from his mobile network operator website. Mobile wallet application has symmetric key generation and payment software. After successful installation, a set of symmetric keys ($X = \{X_1, X_2, ..., X_n\}$) will be generated. They will be stored in payer's mobile device and will be sent to his mobile network operator. Payee must go through the similar registration process with his mobile network operator. This enables him to receive the

| |
|---|
| Phase 1: Payment Initialization:<br>Payer ⇒ Payee: $R_1, TID_{Req}, PayeeID_{Req}$<br>Payee ⇒ Payer: $\{ID_{Payee}, TID, ID_{MNO}\}k_2$ |
| Phase 2: Payment Subtraction Request Payer:<br>Payer ⇒ Payer's MNO: { $ID_{Payee}, ID_MNO, R_1$, TID, AMOUNT, DATE, $R_2$,<br>H ($ID_{Payee}, ID_{MNO}, R_1$, TID, AMOUNT, DATE,$R_2$), {$R_2$, DESC } $K_2$} $X_i, i, ID_{Payer}$<br>Payer's MNO ⇒ pay-center:<br>H [ $\{ID_{Payee}, ID_{MNO}, R_1$, TID, AMOUNT, DATE, $R_2$, H ($ID_{Payee}, ID_{MNO}, R_1$, TID, AMOUNT, DATE, $R_2$),<br>{$R_2$, DESC } $K_2$}$X_i, i, ID_{Payer}$]<br>pay-center ⇒ Payer's MNO: generates TimeStamp1 and verifies Payer's digital signature |
| Phase 3: Payment Authorization Request:<br>Payer's MNO ⇒ Payee's MNO: $R_1, ID_{Payee}$ , TID,<br>AMOUNT, DATE, {$R_1$, DESC } $K_2$ |
| Phase 4: Payment Confirmation Request:<br>Payee's MNO ⇒ Payee: {$R_1$,TID, AMOUNT, DATE, {$R_1$, DESC } $K_2, R_2$,<br>H ($R_1$,TID, AMOUNT, DATE, {$R_1$, DESC } $K_2, R_2$), H ($K_{pp}$)}$y_i, i$ |
| Phase 5: Payment Confirmation Response:<br>Payee ⇒ Payee's MNO: { Yes/No, $R_2$, H ($K_{pp}$, H ($R_1$, TID, AMOUNT, DATE,<br>{$R_1$, DESC } $K_2, R_2$), { Yes/No, TID, AMOUNT, DATE } $K_2$} $Y_{i+1}$ |
| Phase 6: Payment Authorization Response:<br>Payee's MNO ⇒ pay-center: H ({ Yes/No, $R_2$, H ($K_{pp}$),<br>H ($R_1$, TID, AMOUNT, DATE, {$R_1$, DESC } $K_2, R_2$), { Yes/No, TID, AMOUNT, DATE } $K_2$} $Y_{i+1}$)<br>pay-center ⇒ Payee's MNO: generates TimeStamp2 and verifies Payee's digital signature<br>Payee's MNO ⇒ Payer's MNO: Yes/No, TID, AMOUNT, DATE, { Yes/No, TID, AMOUNT, DATE } $K_2$ |
| Phase 7: Payment Subtraction Response:<br>Payer's MNO ⇒ Payer: { Yes/No, $R_2$, H ($K_{p\text{-}p}$), H ($ID_{Payee}, ID_MNO, R_1$,<br>TID, AMOUNT, DATE, $R_2$ ), { Yes/No, TID, AMOUNT, DATE } $K_2$} $X_{i+1}$<br>Payee's MNO ⇒ Payee: { Received, $R_2$, H ($K_{p\text{-}p}$), H ($R_1$, TID, AMOUNT,<br>DATE, {$R_1$, DESC } $K_2, R_2$)} $Y_{i+1}$ |

TABLE III: Our mobile payment protocol

payment amount. Payee generates a set of symmetric keys ($Y = \{Y_1, Y_2, ..., Y_n\}$) with his mobile network operator. These keys will be stored into payee's terminal and his mobile network operator's database.

In our protocol, if a person captures details of a payment transaction, he will not be able to use the message again, since all messages are encrypted in our protocol. Besides, these messages include random time-stamp generated numbers in order to protect our protocol from replay attacks. If someone steals the payment device, he can access ($X$ or $Y$) the shared keys. Therefore, the thief can decode the payment messages and use them for illegal payment. To address this issue, all keys ($X$ and $Y$) are encrypted in client device (with his key). Note that, this key is only viewable by his mobile network operator. Client does the following steps in order to obtain the client key.

P ⇒ P's Mobile Network Operator: {$PN_P$, Current Date and Time,$CK_{Request}$}$Pu_{P's\ MNO}$
P's MNO ⇒ P: {CK} Encrypted with $Pu_P$

Current payment protocols support transaction privacy protection from eavesdroppers. However, they don't support transaction privacy protection from bank. So, it is obvious that who is paying how much to whom for ordering what items in each transaction. Also, some credit card issuers provide categorized spending charts (ex. merchandise, dining, and travel) for their customers. So, the financial institution or bank knows details of the transaction. We want to provide transaction privacy protection in our protocol. For this purpose, we encrypt transaction's details before sending it to pay-center (payment gateway).

The next challenge is to support non-repudiation. We should make sure that after a successful payment, payer or payee can't deny the transaction. For this purpose, we utilize digital signatures. Pay-center is responsible for generating time-stamps and verifying digital signatures in our protocol. Our proposed payment protocol has seven phases. In our protocol, we verify digital signatures twice. In phase 2, pay-center verifies payer's digital signature and generates our first time-stamp. In phase 6, pay-center generates the second time-stamp and verifies payee's digital signature. Because of these two verifications, we can support non-repudiation in our payment protocol.

Most of the current payment protocols support the feature

of preventing replay attacks in payment protocols. We also support this feature, since it is an essential and fundamental feature of a payment protocol. If we don't have a mechanism to prevent replay attacks, the payment transaction may be used again by an eavesdropper. If an eavesdropper captures one of the transactions, he can manipulate the transaction and use it again for illegal purposes. We also have another restriction about our keys that prevents replay attack. This restriction will be discussed later.

In our protocol, we have a mechanism to prevent replay attacks. We have two random and time-stamp generated numbers. The first one is payer's pseudo-ID. The second one is to prevent replay attacks. We include this number in our messages in phases 2, 4, 5, 6, and 7 of our payment protocol in order to prevent replay attacks. In this case, a person cannot use a transaction for the second time, since the time-stamp is not matched with the current time. As we mentioned earlier, our proposed payment protocol is composed of seven phases as illustrated in figure 1. You can see these seven phases with their details in table III. We designed these steps for mobile platform. These phases should be implemented properly as a payment protocol.

In Summary, payer sends the subtraction request to his mobile network operator. His mobile network operator sends the request to payee's mobile network operator. Payee's mobile network operator sends the request to payee and receives his response. Payee's mobile network operator sends the reply to payer's mobile network operator. If payee accepts the request, payer's mobile network operator will initiate the transaction through the payment gateway (pay-center). If payee rejects the request, payer's mobile network operator will inform the payer about the denial. After a successful transaction pay-center informs mobile network operators about the successful result. Then, they inform their clients about the result of their transaction.

After successful completion of these seven phases, payee will release or deliver the purchased goods or services. As we mentioned earlier, one of the challenges in mobile payment is to prevent replay attacks. To prevent replay attacks, payer's mobile network operator and payee's mobile network operator make sure that symmetric keys ($X_i$ and $Y_i$) have not been used before processing the current payment transaction. Mobile network operators will keep a list of generated secret keys and expire used symmetric keys from the list. Payer and payee may receive an update notification from their mobile network operators when their key is expired. To update their secret keys, they should connect to their mobile network operator and generate a new session key ($K_1$) by running our key agreement protocol. Then, they generate a new set of secret keys ($X$ and $Y$) with a new session key ($K_1$) in offline mode.

We utilize cloud messaging in order to add an extra layer of security to our payment protocol. Our intention is to prevent card not present fraud in our payment protocol. For this purpose, we try to improve 3D Secure features for authenticating a customer. Note that, 3DS is utilizing behavioral data to understand which transactions are suspicious. This protocol will not prompt all the customers, but it only prompts transactions with risk score shows higher than threshold. There is a behavior model available that assigns each transaction a score based on different factors [12]. If this risk score is less than threshold, the system allows the customer to continue the transaction. But, if this score is higher than threshold, customer will be prompted by a pop-up window or inline frame in the middle of transaction [12].

We borrow all 3DS features for this extra security level, but instead of showing the extra layer as pop-up window or inline frame, we utilize cloud messaging for this purpose [5]. We prevent several issues by utilizing cloud messaging instead of pop-up window (ex. difficulty in viewing pop-up windows in mobile devices, difficulty in receiving one time passwords and confirmation codes via text message, and unable to see the source of pop-up window). We will push a notification to our customer to make sure that he is the legitimate customer. Note that, all mobile operating systems are equipped with cloud messaging tools. The benefit of this approach is that the customer recognizes the source of push notification. Also, if sending a one-time password is required, this approach will be better, since it doesnt have the limitations of text messaging or emails. Note that, texting and sending email from one country to the other one may be temporarily unavailable or limited, but cloud messaging only require access to Internet [14].

## IV. Performance Evaluation

In this section, we want to Compare execution time of our protocol with DH (Diffie-Hellman). For this experiment, we had a virtual machine with 7 GB of memory, and two CPUs (each CPU had two cores). The operating system was windows seven 64 Bit. We ran our protocol against Diffie-Hellman hundred times to see the difference. In our experiment, we run two protocols against each others 100 times and we kept stacked time of these experiments. You can see the result of these experiments in figure 2. Horizontal axis is showing number of our experiments and vertical axis is showing the stacked time based on milliseconds. After hundred experiments, average execution time of our protocol was 34,930 milliseconds and average execution time of the original version was 63,116 milliseconds. In the next experiment, we run the protocols against each others 30,000 times. We wanted to see the result for large number of iterations. Our prime number was 7. You can see the result of these experiments in figure 3. After 30,000 experiments, average execution time of our protocol was 3,899 milliseconds and average execution time of the original version was 6,545 milliseconds.In summary, our protocol was almost twice as fast as Diffie-Hellman. Besides, we compare our proposed payment protocol with existing payment protocols. Table IV shows the result of these comparisons.

## V. Conclusion

In this paper, we introduced a new payment protocol that is compatible with mobile platform. We decreased

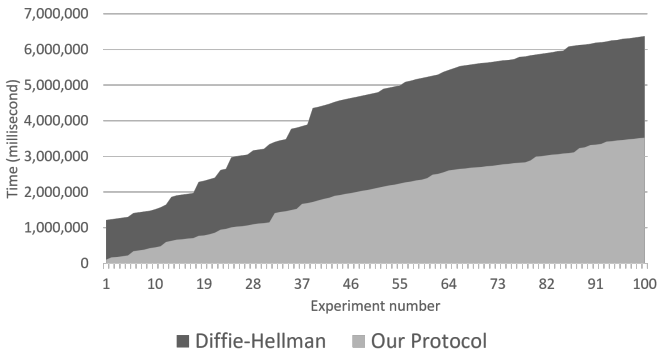


Fig. 2: Performance evaluation: prime = 7 and 100 iterations

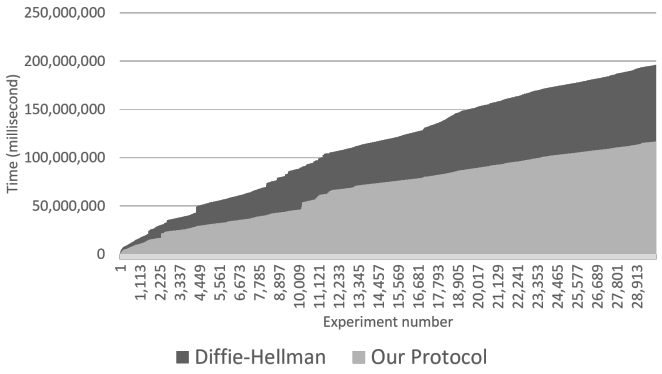


Fig. 3: Performance evaluation: prime = 7 and 30,000 iterations

computational cost of generating a shared key between two parties. Based on our experiments, our protocol is almost twice as fast as Diffie-Hellman. We defined two different random and time-stamp generated numbers in order to avoid replay attacks. We utilized digital signature in order to support non-repudiation in our protocol. We defined and extra security layer to prevent card not present fraud. We utilized cloud messaging to implement this extra security layer in order to solve original 3DS issues.

However, there are some parts of this research that can be improved. We will try to reduce the computation and communication costs of key agreement protocol. We try to improve the performance of 3DS behavioral model. So, we should only show our extra security layer for transactions that are more likely a threat. We should extend our protocol in order to support wider range of devices especially popular devices in Internet of things. We will try to utilize other cloud technologies in our payment protocol.

| Key Points | SET | iKP | KSL | MPCP 3D | 3DS |
|---|---|---|---|---|---|
| Identity Protection from Merchant | N | N | N | Y | N |
| Identity Protection from Eavesdrop | Y | Y | Y | Y | - |
| Transaction Privacy Protection from Eavesdrop | Y | Y | Y | Y | - |
| Transaction Privacy Protection from Bank | N | N | N | Y | - |
| Supporting Non-repudiation by Digital Signature | N | N | N | Y | - |
| Compatible with Mobile Devices | N | N | N | Y | - |
| preventing card not present fraud | N | N | N | Y | - |
| preventing man in the middle attack and phishing scam | - | - | - | Y | N |
| No geographical restrictions in the extra security layer | - | - | - | Y | N |
| customer can see the source of extra security layer | - | - | - | Y | N |
| Is the implementation of 3D compatible with mobile devices? | - | - | - | Y | N |

TABLE IV: Comparison

## REFERENCES


[1] Vahidalizadehdizaj, M., A. Moghaddam, R., Momenebellah, S., "New mobile payment protocol: Mobile pay center protocol (MPCP)," *International Conference on Electronics Computer Technology*, vol.2, no., pp.74,78, 8-10 April 2011.

[2] Vahidalizadehdizaj, M., Moghaddam, R.A., Momenebellah, S., "New mobile payment protocol: Mobile pay center protocol 2 (MPCP2) by using new key agreement protocol: VAM," *IEEE Pacific Rim Conference on Computers and Signal Processing*, vol., no., pp.12-18, 23-26 Aug. 2011.

[3] Vahidalizadehdizaj, M., "New mobile payment protocol: Mobile pay center protocol 4 (MPCP4) by using new key agreement protocol: VAC2," *IEEE International Conference on Electronics Computer Technology*, vol.2, no., pp.67-73, 8-10 April 2011.

[4] Vahidalizadehdizaj, M., Geranmaye, M., "New Mobile Payment Protocol: Mobile Pay Center Protocol 5 (MPCP5) by using New Key Agreement Protocol: VG1," *IEEE International Conference on Computer Modeling and Simulation*, vol.2, no., pp.246-252, 2011.

[5] Vahidalizadehdizaj, M., Geranmaye, M., "New Mobile Payment Protocol: Mobile Pay Center Protocol 6 (MPCP6) by Using New Key Agreement Protocol: VGC3," *IEEE International Conference on Computer Modeling and Simulation*, vol.2, no., pp.253-259, 2011.

[6] Vahidalizadehdizaj, M., Tao, L., "A new mobile payment protocol (GMPCP) by using a new key agreement protocol (GC)," *IEEE International Conference on Intelligence and Security Informatics*, vol., no., pp.169-172, 27-29 May 2015.

[7] Vahidalizadehdizaj, M., Tao, L., " A New Mobile Payment Protocol (GMPCP) By Using A New Group Key Agreement Protocol (VTGKA)," *IEEE International Conference on Computing, Communication and Networking Technologies*, 2015.

[8] "Verified by Visa", visaeurope.com, 2016. Available: https://www.visaeurope.com.

[9] "MasterCard SecureCode", mastercard.us, 2016. Available: https://www.mastercard.us.

[10] "American Express SafeKey", americanexpress.com, 2016. Available: https://network.americanexpress.com.

[11] "JCB International J/Secure", global.jcb, 2016. Available: https://http://www.global.jcb/.

[12] I. Warren, A. Meads, S. Srirama, T. Weerasinghe and C. Paniagua, "Push Notification Mechanisms for Pervasive Smartphone Applications," *IEEE Pervasive Computing*, vol. 13, no. 2, pp. 61-71, Apr.-June. 2014.

[13] EMV Migration Forum, "Near-Term Solutions to Address the Growing Threat of Card-Not-Present Fraud: Card-Not-Present Fraud Working Committee White Paper", April 2015.

[14] Paul, D., Hongrui, G., Kannan, S., CA Technologies, Advanced Analytics and Data Science, "3D-Secure Authentication using Advanced Models", October 2014.

[15] SET Secure Electronic Transaction Specification, Version 1.0, May 31, 1997.

[16] "Forrester:Mobile And Tablet Commerce Forecast 2015 To 2020", Forrester.com, 2016. Available: https://www.forrester.com.

[17] Bidgoli, H., "The Internet Encyclopedia". Hoboken, N.J.: John Wiley and Sons, 2004.