Computer Standards & Interfaces 000 (2017) 1–6

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Analysis of a mobile payment protocol with outsourced verification in cloud server and the improvement

Yongjian Liao*, Yichuan He, Fagen Li, Shijie Zhou

University of Electronic Science and Technology of China, Chengdu, China

ARTICLE INFO

Keywords:
Mobile payment
Signature
Bilinear maps
Securely outsourcing computation
Cloud computing

ABSTRACT

Today, mobile payment is becoming one of the most frequently used approach to provide payment services under business and financial organization via mobile devices, such as smart phone, ipad. However, the limited resources of the mobile devices cause that it can not perform large-scale computing. Thus, it is a better way to outsource securely some computation of mobile payment to the untrusted cloud server. Recently, Qin et al. proposed a mobile payment protocol with outsourced verification in the untrusted cloud server. In this paper, we firstly show that their protocol exists two issues: one is an unreasonable construction, which causes their protocol not to be implemented; the other is that there is a colluding attack of customers and the untrusted cloud server at outsourced verification phase, which causes the verification of their protocol to be insecure. Next, we improve their protocol and analyze the security of our improved protocol.

1. Introduction

More and more financial-services apps and the availability of mobile device drive the growth of mobile payment services. As one of the modern components of mobile payment services, mobile wallet provides a very convenient way to allow the clients to conduct the payment via their mobile devices from anywhere and anytime. Obviously, it is possible that mobile payment is becoming one of the most popular payment methods in the near future. However, mobile devices, such as smart phone and ipad, which are limited-resource, can not perform large-scale computing. Thus an easy and convenient method is to outsource some complex computation of a mobile payment protocol to an untrusted cloud server.

Recently, Qin et al.[2] proposed an efficient privacy-preserving mobile payment protocol with outsourced verification in untrusted cloud server. There were four main entities directly involving in the interactive protocol. A payment service provider(PSP), a customer, a merchant and an untrusted cloud server. A payment service provider generates the pseudo public/private key of entities(the customer client and the merchant). The customer wants to buy goods or services of the merchant. The merchant needs to sell some goods or services to the customer. The untrusted cloud server provides some outsourced computing to reduce computation cost of the merchant(or the customer) in payment phase. According to the practical security requirements, the protocol must satisfy the following security properties: unforgeability, anonymity, traceability and non-repudiation. The unforgeability property guarantees that any payment and receipt are not forged; the anonymity property guarantees that the merchant(or the customer) does not know the real identity of the customer(or the merchant); the traceability property guarantees that the PSP knows the real identity of entities of transactions from the payment and the receipt.

However, aim to the Qin et al.'s construction, it is not enough for the protocol to only satisfy the above security properties. This is because the cloud server is untrusted, and the value replying from the cloud server may be "false" which can cheat the merchant(or customer). We describe a practical attack in the following scenario, which is called a colluding attack. A customer Alice wants to buy an Apple Mac Book Air of the merchant Bob, which needs 1700 dollars. When both of them agree on this price, Alice signs *Payment* to generate her "signature" $\tilde{\sigma}$, which includes a transaction identity, price to be paid and some pseudo identities of Alice and Bob. Then Alice sends *Payment* and $\tilde{\sigma}$ to Bob, and at the same time she also sends *Payment* and $\tilde{\sigma}$ to the cloud server and pays 700 dollars to the cloud server in order to let the cloud server help her to cheat Bob. Bob first generates $\tilde{\sigma}'$ by simply randomizing the $\tilde{\sigma}$, and then sends $\tilde{\sigma}'$ to the cloud server. At last, when the cloud server receives $\tilde{\sigma}'$ and $\tilde{\sigma}$ even if $\tilde{\sigma}$ is invalid, it also can compute the values needed by Bob from $\tilde{\sigma}'$ and $\tilde{\sigma}$ if the construction of generating $\tilde{\sigma}'$ in outsourced verification phase is too simple. Since there is no verification mechanism for the outsourced verification of the untrusted cloud server in order to reduce the computation cost of Bob, it is possible that Bob will accept the invalid signature $\tilde{\sigma}$. Finally, Alice pays 700 dollars to buy the Apple

* Corresponding author.
E-mail address: liaoyj@uestc.edu.cn (Y. Liao).

https://doi.org/10.1016/j.csi.2017.09.008
Received 4 January 2017; Received in revised form 24 September 2017; Accepted 24 September 2017
Available online xxx
0920-5489/

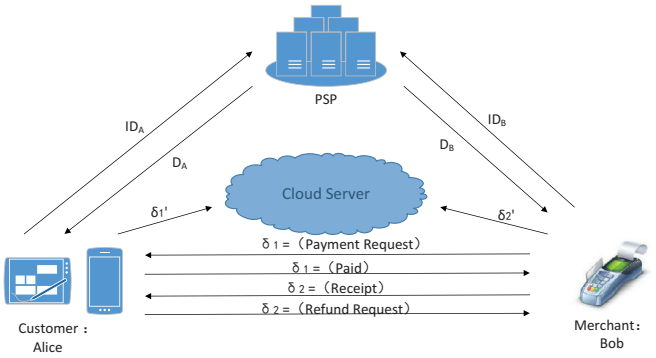

Fig. 1. System model.

Mac Book Air which worths 1700 dollars, but Bob loses the device and gets nothing. Unfortunately, the protocol of Qin et al.is insecure on the colluding attack.

In the paper, we firstly point out that the construction of their protocol is unreasonable, which causes the protocol not to be implemented. Then we show their protocol is not secure under the colluding attack of client and untrusted cloud server at outsourced verification phase. Finally, we improve their protocol and analyze the security of our improved protocol.

The rest of this paper is organized as follows. In Section 2 we recall the system model and security requirement of the protocol and the bilinear pairing. Then we recall Qin et al.'s mobile payment protocol and prove it isn't secure in Section 3. We propose our improved protocol and analyze its security and efficiency in Section 4. Finally, we conclude the paper in Section 5.

2. Preliminaries

2.1. System model

In this section, in order to make the mobile payment protocol with outsourced verification (MPP-OV) in cloud server [2] be clear. We simplify their complex system model and omit the entities which are not necessary to directly use in MPP-OV protocol. The MPP-OV protocol includes the following four entities. The interactions of the four entities are described in Fig. 1.

- Client (or Customer). An entity, Alice, is one who wants to purchase goods or services provided by a merchant.
- Merchant: An entity, Bob, is a merchant who wants to sell goods or services to the client Alice.
- Payment Service Provider (PSP). A trusted entity, which generates the pseudo identity and corresponding partial private key of entities, is responsible for the security and privacy of the payment information.
- Untrusted Cloud Server Verification Provider (CSVP). An untrusted entity carries out outsourcing the computation of verification to reduce the computation overhead of merchants or clients.

Then, we recall the MPP-OV protocol, which is an interactive protocol divided into three phases:*Setup* and *Key Generation Phase, Payment Transaction Phase* and *Outsourced Verification Phase*.

- *Setup* and *Key Generation Phase*. The PSP takes as input a security parameter k to generate the public parameters *Params*, a master key mk and the description of a finite signature space and a finite message space. And it keeps mk secretly. Then it takes as input the real identity $ID \in \{0,1\}^l$ of the client[1] and the master key mk, and outputs a partial private key SK_{ID} and a pseudo identity P_{ID} of the client and a public key PK generated by his/her self, where l is a positive integer.
- *Payment Transaction Phase*. When the client Alice and the merchant Bob agree on goods or services and its amount paid, Alice will sign them by using her partial private key and private key generated by herself to perform the payment transaction. At the end of payment transaction phase, Alice receives an acknowledgment of a receipt, which is signed by Bob.
- *Outsourced Verification Phase*. This phase is not an independent phase. Due to the limited resources of mobile devices, the merchant Bob adopts cloud server-aided verification technique to verify the validity of a signature of the client Alice in *Payment Transaction Phase* of the protocol. That is to say, the untrusted cloud server could help Bob (or Alice) to compute some value in order to reduce the computation overhead of verification of Bob (or Alice).

In order to maintain mobile payment security, the protocol should be able to satisfy the following requirements [2]:

- Unforgeability: Only legal users can make transactions. In other words, no one can impersonate any user to submit a fake payment or a fake or illegal receipt.
- Anonymity: The identities of users must be kept confidential.
- Traceability: The merchant cannot deny the received payment, while the customer cannot deny her confirmed payment. Otherwise, the PSP can be used to trace them.
- Non-repudiation: The merchant cannot repudiate the origin and the correctness of the receipt information. Also no customer can deny his/her confirmed payment.

2.2. Bilinear map

In this section, we firstly recall some concepts about a bilinear map(or pairing) below. Let G_1 and G_2 be an additive cyclic group and a multiplicative cyclic group of the prime order p respectively. And P is a generator of G_1. A map e: $G_1 \times G_1 \to G_2$ is called an admissible bilinear map [1] if it satisfies the following properties:

- Bilinear: For any $P \in G_1$ and $a, b \in \mathbb{Z}_p$, $e(aP, bP) = e(P, P)^{ab}$.
- Non-degenerate: $e(P, P) \neq 1$, where 1 is the identity element of G_2.
- Computable: There is an efficient algorithm to compute $e(P, P)$ for any $P \in G_1$.

The bilinear Diffie–Hellman (BDH) problem in (p, G_1, G_2, e) is described as follows:

Given P, aP, bP, cP for random elements $a, b, c \in \mathbb{Z}_p^*$, there is a probabilistic polynomial-time (PPT) algorithm which outputs $e(P, P)^{abc}$.

Defintion 1. Suppose $\mathcal{A}$ is a PPT algorithm. It outputs $e(P, P)^{abc} \in G_2$ with the advantage:

$$Adv_{\mathcal{A}}^{\mathrm{BDH}}(k) = \Pr[\mathcal{A}(P, aP, bP, cP) = e(P, P)^{abc} \ : \ P, aP, bP, cP \in G_1].$$

We say that the BDH assumption holds if for any PPT algorithm $\mathcal{A}$, its advantage $Adv_{\mathcal{A}}^{\mathrm{BDH}}(k)$ is negligible in the parameter k.

The computational Diffie-Hellman (CDH) problem in G_1 is described as follows:

Given $P, aP, bP \in G_1$ for random elements $a, b \in \mathbb{Z}_p^*$, there is a PPT algorithm which outputs abP.

Defintion 2. Suppose $\mathcal{A}$ is a PPT algorithm. It outputs $abP \in G_1$ with the advantage:

$$Adv_{\mathcal{A}}^{\mathrm{CDH}}(k) = \Pr[\mathcal{A}(P, aP, bP) = abP \ : \ P, aP, bP \in G_1].$$

We say that the CDH assumption holds if for any PPT algorithm $\mathcal{A}$, its advantage $Adv_{\mathcal{A}}^{\mathrm{CDH}}(k)$ is negligible in the parameter k.

[1] In the paper [2], the client's real identity $ID \in \{0,1\}^*$, this construction has some deficiency. Because in *Setup* and *Key Generation Phase* of their protocol the PSP used $ID \oplus H(\cdot)$ to hide the identity ID for some hash function $H(\cdot)$. In general, the length of the output of hash function $H(\cdot)$ is constant. Thus, $ID \in \{0,1\}^*$ can cause binary operation $\oplus$ not to perform.