Anonymous Transactions in Computer Networks

by

Shlomi Dolev and Marina Kopeetsky

Technical Report #09-04

August 2009

Abstract. We present schemes for providing anonymous transactions while privacy and anonymity are preserved, providing user anonymous authentication in distributed networks such as the Internet. We first present a practical scheme for anonymous transactions while the transaction resolution is assisted by a Trusted Authority. This practical scheme is extended to a theoretical scheme where a Trusted Authority is not involved in the transaction resolution. Given an authority that generates for each player hard to produce evidence *EVID* (e. g., problem instance with or without a solution) to each player, the identity of a user U is defined by the ability to prove possession of said evidence. We use Zero-Knowledge proof techniques to repeatedly identify U by providing a proof that U has evidence *EVID*, without revealing *EVID*, therefore avoiding identity theft.

In both schemes the authority provides each user with a unique random string. A player U may produce unique user name and password for each other player S using a one way function over the random string and the IP address of S. The player does not have to maintain any information in order to reproduce the user name and password used for accessing a player S. Moreover, the player U may execute transactions with a group of players S^U in two phases; in the first phase the player interacts with each server without revealing information concerning its identity and without possibly identifying linkability among the servers in S^U. In the second phase the player allows linkability and therefore transaction commitment with all servers in S^U, while preserving anonymity (for future transactions).

1 Introduction

Due to the rapid development of the Service Oriented Architecture (SOA) and web services in supporting different multiphase business processes, the issue of providing services which preserve user anonymity, is very important nowadays. SOA is defined as a group of services that communicate with each other. The process of communication involves a number of services coordinating a common activity. The purpose of this paper is to design the new protocols for providing the multiphase transactions between users and a set of servers while maintaining the user anonymity.

We provide anonymous authentication in distributed networks. We define and present protocols for executing anonymous transactions between a user and a set of servers. Both the user privacy and anonymity are preserved during the transaction. Protocols for providing anonymous communications via insecure communication channels were proposed in, e. g. [7], [8], [11] and [3]. Our scope is beyond the actual anonymous communications assuming one of the above protocols is used to provide such a service.

Several protocols devoted to the performance of anonymous transactions have been recently proposed. For example protocols for providing anonymous credentials were proposed and analyzed in [16], [4] and [2]. The authors proposed pseudonym systems which allow users to interact with multiple organizations anonymously, using unlinkable pseudonyms. In [4] an accumulator scheme is proposed. Dynamic accumulators enable efficient revocation of credentials in the anonymous credential systems. A special P-signature scheme which enables a practical design of a non-interactive anonymous credential system is proposed in [2]. The scope of the above works is user centric, where the servers do not communicate directly. In such a user centric design the communication and computation over the user is high.

A system for anonymous personalized web browsing was proposed in [10]. A new cryptographic function, namely the Janus function that satisfies a number of properties, including anonymity, consistency, secrecy, uniqueness of an alias, and protection from creation of dossiers provides user anonymity in a computationally secure aspect. This function translates the user name (for example, e-mail address) to a unique user alias. The computation of the Janus function and the translation of the real user name to the corresponding alias is performed by the centralized Janus proxy server which is usually placed in the firewall which connects a particular Intranet to the Internet.

Our contribution. The contribution of our paper is in the design of new schemes for providing user with anonymous transactions. A transaction is defined as a two stage operation. During the first stage a player (user) U anonymously authenticates himself/herself by means of a unique pseudonym to several other players (servers) S^U. In the second stage U can prove to each sever S from S^U that the same entity interacted with all the servers and therefore has done all the preparations to execute a (composite) transaction. Each user has a one way function F. The user U may produce a unique user name and password for each other player (user or server) S using both a random seed and the IP address of S. The user U does not have to maintain any information in order to reproduce the user name and password used for accessing a server S. Moreover, the user U may execute transactions with a group of players (servers or users) S^U and prove without revealing information concerning the identity of U that U is the one that accessed each player in S^U. The pseudonym generated by a user for accessing any server in a group of servers, is unique for each user-server pair. The pseudonym uniqueness is provided by the server's IP address which is one of the parameters of the seed that generates pseudo random sequence by a given one way function.

We present two protocols for anonymous transactions. The first Anonymous Transactions Protocol ATP_1 is based on the approval of the Trusted Authority (*TA*) for the transaction resolution. The *TA* may limit the number of possible transactions carried out by the same user. The transactions are performed in two modes: unlinkable and linkable transactions. The different user's transactions can be unlinkable in the sense that different players from a set of network users do not know that the same anonymous identity requested a service from them. The transactions linkability may also be satisfied as the users get tools to verify whether they have received service from the same anonymous identity. The transactions mode is chosen by the player who accesses the *TA* in order to carry out the transactions.

The advanced theoretical protocol ATP_2 uses the *TA* in the initialization stage only, while the anonymous transactions are carried out by a player (user) without involving the *TA*. In the advanced scheme the *TA* generates a hard to produce evidence *EVID* for each user U e.g., a problem instance with or without a solution. The user's pseudonym in this scheme is based on the server's IP address, a hard to produce evidence *EVID*, and a one way function granted to the user by the *TA* during the system initialization. The pseudonym includes the anonymous user name (login), and the password. The pseudonym is created as a pseudo random sequence. This pseudo random sequence is generated by the one way function and a seed calculated by $XOR-ing$ the server's IP address and an evidence *EVID*. In order to perform the transaction resolution and prove to the group of players S^U that a user U processed all accesses needed to complete the transaction, the user performs an interactive Zero Knowledge Proof to prove that he/she knows *EVID* which is computationally hard to produce by a polynomial time adversary. The advanced ATP_2 protocol uses Zero-Knowledge proof techniques to repeatedly identify the user U without exposing the *EVID* and its solution, therefore avoiding an identity theft. Hence, the *TA* in ATP_2 is involved only once during the system initialization.

Compared with the Janus system, our protocol has several advantages.
Anonymity between users, Trusted Authority, and servers in the network. Anonymity is provided between all players: a user which performs a transaction; a group of servers participating in the transaction; and a Trusted Authority.
Performing the transactions without involving a Trusted Authority. The ATP_2 protocol involves the *TA* only once during the system initialization. The future ATP_2 protocol's functionality is implemented by the user which locally computes the pseudonym and resolves the transaction by himself/herself.
Low computational cost combined with a very high security level. Our first protocol continuously uses the pseudo random sequences generation by the cryptographic one way functions and XOR calculation as a source for preserving the security level of the anonymous user's pseudonym.

Therefore, low computational power is required in comparison with the standard cryptographic techniques. The Zero-Knowledge proof performed by the user in the transaction resolution stage computationally prevents an adversary from impersonating himself/herself on behalf of the legal user. In contrast to the Janus system our protocol does not rely on a Secure Socket Layer (SSL) [18] or any other security protocol that assumes the involvement of the Trusted Authority.

Permanent validity of the secret granted to the users by the *TA*. The secret number granted by the *TA* to the user in the ATP_1 protocol and the corresponding hardly producible *EVID* in ATP_2 are valid forever, and only user encryptions of the secret value are updated during different transactions.

Linkability and unlinkability of the transactions. The encryption scheme for the pseudonym generation may be chosen adaptively on user demand. The user may choose the unlinkable mode while neither network players (users and servers) can identify that they communicated with the same anonymous identity during different transactions. The user may also perform transactions in a linkable mode while the other players acquire tools to ascertain that the same player executed with them different transactions, whereas the user's identity remains anonymous.

Compared with the recent papers on anonymous credentials, such as [16], [4] and [2] our anonymous transactions protocols ATP_1 and ATP_2 have the following benefits.

Complete anonymity to Trusted Authority. A user in our scheme gets the permission for providing the transactions in the completely anonymous manner, without using Public Key Infrastructure.

Proof of linkability. A user in ATP_1 protocol must not prove the linkability, whereas the servers verify the linkability on their own, unlike the anonymous credential schemes.

Pseudonym storage. User in our scheme does not have to remember and store his/her pseudonym to each server in the system. The pseudonym is efficiently generated by a user each time when he/she intends to access a server.

Scale to number of servers. A user in ATP_1 and ATP_2 protocols proves the transaction correctness simultaneously to a group of servers, while in the anonymous credential systems a user cannot prove the possession of the credentials simultaneously to a group of organizations.

Simplicity. ATP_1 is not based on the ZKPs as credential schemes in [16], [4] and [2].

Performing transaction by a single operation. In our model a user performs a transaction by a single operation. Hence, the transaction resolution in our model is atomic. In the anonymous credential system a user is highly involved with each organization separately for validation of his/her credential transfer (performing the transaction).

Involving of Trusted Authority. In the ATP_2 protocol the Trusted Authority is involved in the initialization stage only, while in the anonymous credential systems the TA carries out the transaction resolution.

Paper organization. The formal system description appears in Section 2. The basic protocol for anonymous transactions ATP_1 is introduced in Section 3.1. The advanced protocol, impractical in current technology, ATP_2

protocol which provides transaction resolution anonymously without involving the Trusted Authority, is presented in Section 4. Conclusions appear in Section 5.

2 Security Model for Anonymous Transactions

We consider the set of network players (users and servers), and a Trusted Authority *TA* which comprise the anonymous network. The set of network players includes two sub-sets that may intersect: the set of users $U = \{U_1, \dots, U_n\}$ and the set of servers $S = \{S_1, \dots, S_l\}$. A user $U_i \in U$ initiates and carries out an anonymous transaction with the servers from the corresponding server set S^{U_i}. The transaction is defined as a two stage operation:
(a) The first stage is the authentication stage. During this stage a user U_i anonymously authenticates himself/herself and interacts with each server S_j from the server set S^{U_i}.
(b) The second stage is the transaction resolution. In this stage U_i proves to each server from the S^{U_i} set that he/she is the identity that provided authentication and visited all the servers from S^{U_i} during the previous stage. Note that this stage is optional and depends on U_i's choice.
According to [7], we use two cryptographic primitives: *encryption* and *authentication*. *Encryption* guarantees the secrecy of messages, while *authentication* ensures that if a sender sends a message to a receiver and an adversary alters this message, then with overwhelming probability the receiver can detect this fact (see [7] for details and [13] for formal definitions). Denote the k^{th} transaction carried out by the user U_i with the servers from the corresponding server set S^{U_i} as T_i^k. Each T_i^k starts with the authentication message t_{ij} sent by U_i to each $S_j \in S^{U_i}$, and ends when each $S_j \in S^{U_i}$ confirms the transaction resolution by sending the message $r_i = Confirm_i^k$, or rejects it by sending the message $r_i = Reject_i^k$. The message t_{ij} consists of the anonymous user name and password pair (u_{ij}, p_{ij}) which is, in essence unique for a given user U_i and a server S_j. The transaction resolution is initiated by the user U_i and is performed by sending the message $Resolve_i^k$ to each $S_j \in S^{U_i}$.

The security parameter in our model is ϵ which is equal to the length of the seed c_{ij} that generates the pseudo random sequence for the anonymous user name and password. The larger ϵ is compared to the pseudo random sequence length, the higher the computational security level of the proposed scheme is.

Given the features of the proposed model, we describe the basic anonymous transactions protocol ATP_1, and the advanced ATP_2 protocol. The transaction resolution in ATP_1 is executed by the *TA* when each server S_j passes the $Resolve_i^k$ message received from U_i to the *TA*, while the *TA* confirms the transaction correctness by sending the message $r_i = Confirm_i^k$, or rejects the transaction by sending the $r_i = Reject_i^k$ to all servers from the S^{U_i} set.

The transaction resolution provided by ATP_2 is performed by the user U_i while U_i proves that he/she knows the hardly producible evidence *EVID* to each server from the corresponding server set S^{U_i} in the interactive ZKP. We assume a polynomial time restricted semi-honest adversary A [19]. A

can impersonate as the legal user and, therefore can get access to the server resources. A can also try to prove to the group of servers that he/she is a legal user that initiated the transaction. Nevertheless, A must follow the protocol fairly.

We define the requirements for the ATP_1 and ATP_2 protocols.

(R_1)-anonymity: If a user U_i performs a transaction, neither the network players, including the *TA* nor the servers get information about $U_i's$ real identity.

(R_2)-Anonymous authentication to *TA*: The user U_i authenticates himself/
herself to the *TA* and gets the credit for the execution of a certain number of transactions via an anonymous secure channel.

(R_3)-Pseudonym uniqueness: the pseudonym which is composed of the user's user name and password is unique for any user-server pair that participated in the transaction.

(R_4)-Pseudonym consistency: The user pseudonym is consistent for each server in the sense that the server can recognize the pseudonym in the course of the repeated user visits in the same transaction.

(R_5)-Atomicity: The transaction resolution is executed simultaneously for all servers in the corresponding servers' set.

(R_6)-Optionality of the transaction mode (linkability and unlinkability of a transaction): A user can choose for himself/herself one of two transaction modes: linkable and unlinkable. In the linkable mode the servers from the corresponding servers set can verify that the same user executed with them a certain number of transactions, whereas in the unlinkable mode the servers have no tools to identify the same anonymous identity.

The model presented and the corresponding requirements are defined as a function of the security parameter ϵ that assures the computationally secure level of the anonymous transactions.

The anonymous transactions protocol ATP_1 and the advanced ATP_2 protocol are introduced in the next sections.

3 The Anonymous Transactions Protocol ATP_1

3.1 ATP_1 Description

The ATP_1 protocol is described in Figures 1 and 2. In order to carry out a certain transaction with a group of servers S^{U_i} a user $U_i \in U$ performs the following operations:

(a) Initialization stage: The Trusted Authority *TA* grants the user U_i a secret random number Q_i. It is assumed that there is common knowledge of a one-way function F. t possible transactions are permitted by the *TA* to U_i (Figure 1, (a), Figure 2, lines 1-4). Note that anonymous communication is provided between the *TA* and the user because the *TA* gets no information about the user's identity.

(b) In order to start the k^{th} transaction T_i^k U_i locally computes $Pseudonym_{ij}$ that consists of the anonymous user name u_{ij} and password p_{ij}.
$Pseudonym_{ij} = (u_{ij} \| p_{ij}) = F((Q_i^k) \oplus IP(S_j))$ (Figure 1, (a), Figure 2, lines 5-13). Here $IP(S_j)$ is the corresponding IP address of the server S_j, and Q_i^k, $k = 1, .., t$ is the k^{th} sub-string of the Q_i string. It should be noted that

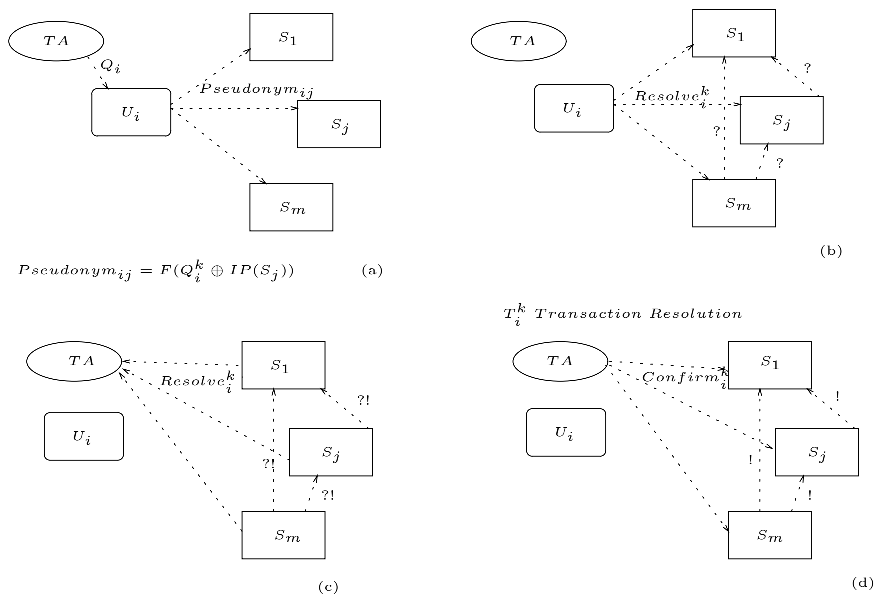


Fig. 1: ATP_1: Stages in Executing a Transaction.

if the Q_i string is short, Q_i may be used as a seed for generating a pseudo random sequence $c_F(Q_i)$ by the F function for any k^{th} transaction instead of the Q_i^k sub-string. In this case $c_F(Q_i)$ should be divided on t sequential sub-strings $c_F^1(Q_i), \dots, c_F^t(Q_i)$. Each $c_F^k(Q_i)$ will generate the user name and password pair for any k^{th} user's transaction.
Assuming that F is a proper collision-resistant one way function [19], the generated $Pseudonym_{ij}$ is unique for all interactions between U_i and any $S_j \in S^{U_i}$. Note, that the user's U_i $Pseudonym_{ij}$ is different for any server S_j from S^{U_i}, and therefore the servers do not know whether U_i is the same identity that visited any server from the corresponding server set. The question marks in Figure 1 (b) relate to this case.

(c) In order to convince each server from the S^{U_i} set that U_i is the same identity that visited any server from the server set, U_i opens the part Q_i^k of the secret Q_i, or the sub-string $c_F^k(Q_i)$ of the pseudo random sequence $c_F(Q_i)$, respectively used in k^{th} transaction by sending the message $Resolve_i^k = (Q_i^k) \oplus IP(S_j)$ (or $Resolve_i^k = c_F^k(Q_i) \oplus IP(S_j)$, respectively) to all servers. This stage is provided by the user for himself/herself (Figure 1 (b), Figure 2, lines 14-16). The question marks in Figure 1 (b), denote that the servers do not know yet that they interacted with the same anonymous user.

(d) Transaction resolution is performed simultaneously when each S_j sends the Q_i^k (or $c_F^k(Q_i)$) to TA revealed from the $Resolve_i^k$ message which was previously received from U_i. TA verifies whether Q_i^k (or $c_F^k(Q_i)$) is the correct Q_i^{th} sub-string, or correct sub-string generated by F from the secret number Q_i. If so, then the TA sends $Confirm_i^k$ message to each S_j from S^{U_i} and resolves the k^{th} transaction T_i^k. Otherwise, the TA sends $Reject_i^k$

```
    $ATP_1$. Protocol for User $U_i$

1:  Initialization:
2:  Get $(Q_i,\ t)$
3:  from $TA$
4:  int $k = 1$
5:      Start $k^{th}$ transaction $T_i^k$
6:      User's $U_i$ authentication
7:      to any $S_j \in S^{U_i}$
8:          for $j = 1..m$
9:              $F_{ij} = F(Q_i^k \oplus IP(S_j))$
10:             $F_{ij} = (u_{ij} \| p_{ij})$
11:             $Pseudonym_{ij} = (u_{ij} \| p_{ij})$
12:             Send $t_{ij} = Pseudonym_{ij}$ to $S_j$
13:     End user's $U_i$ authentication

14:     $k^{th}$ Transaction Resolution
15:         $Resolve_i^k = F(Q_i^k \oplus IP(S_j))$
16:         Send $Resolve_i^k$ to all $S_j$ from $S^{U_i}$
17:         If $r_i$=$Confirm_i^k$
18:             return
19:             $Transaction\ Resolution$
20:         else
21:             return
22:             $Transaction\ Failure$
23: if $k \prec t$ $k := k + 1$
24: else Last Transaction
```

```
    $ATP_1$. Protocol for Server $S_j$

1:
2:
3:
4:
5:
6:
7:
8:
9:
10:
11:
12: Upon reception of $t_{ij}$
13:     Confirm $U_i^{th}$ authentication

14: $k^{th}$ Transaction Resolution
15:     Upon reception of $Resolve_i^k$
16:
17:         If $TA(Resolve_i^k)$=$Confirm_i^k$
18:             Send $r_i = Confirm_i^k$ to $U_i$
19:
20:         If $TA(Resolve_i^k) = Reject_i^k$
21:             Send $r_i = Reject_i^k$ to $U_i$
22:
23: if $k \prec t$ $k := k + 1$
24: else Last Transaction
```

Fig. 2: Anonymous Transactions Protocol ATP_1.

message to each S_j, and rejects T_i^k (Figure 1 (c), (d), Figure 2, lines 17-22). The question marks that appear together with the exclamation marks, in Figure 1 (c), denote that the servers may verify that they intercated with the same anonymous user. The exclamation marks in Figure 1 (d) denote that the transaction resolution has been carried out simultaneously and each server has been convinced that the same anonymous user carried out the transaction.

3.2 Linkable and Unlinkable Transactions

Linkable mode. Let us assume that a certain user U_i wishes that his/her different transactions T_i^k and T_i^l, $k \prec l$ with the same group of servers S^{U_i} will be linkable in the sense that after executing the T_i^k transaction any S_j from S^{U_i} can be convinced that the T_i^k and T_i^l transactions have been performed by the same anonymous user. We propose to use the authentication scheme suggested by Lamport [14]. In this case k^{th} encryption of the secret Q_i used in T_i^k transaction equals to $F^{t-k}(Q_i)$,and the corresponding user name and password pair is $(u_{ij}, p_{ij}) = F(F^{t-k}(Q_i) \oplus IP(S_j))$. It should be noted that the one way function F generates the pseudo random sequence $F(Q_i)$ from the seed Q_i. t denotes the maximum number of the transactions permitted to the user U_i by the TA. $F^{t-k}(Q_i)$ denotes $t-k$ sequential applications of the one way function F on the secret seed Q_i. The value revealed by the user U_i in the $Resolve_i^k$ message is, consequently the internal seed $F^{t-k}(Q_i) \oplus IP(S_j)$ used to compute (u_{ij}, p_{ij}). Assume that the user U_i has provided T_i^k, $T_i^{k+1}, \ldots,$ T_i^l sequential transactions with the servers from the S^{U_i} set. U_i remains linkable after any transaction that follows the T_i^k transaction because each $S_j \in S^{U_i}$ can verify by repeatedly applying $l-k$ times the one way function F on Q_i that Q_i^{th} encryptions $F^{t-k}(Q_i)$ and $F^{t-l}(Q_i)$ used in the k^{th} and l^{th} transactions, respectively satisfy the following equality $F^{t-k}(Q_i) = F^{l-k}(F^{t-l}(Q_i))$.
Unlinkable mode.We propose the following encryption scheme in order to ensure the user's U_i unlinkability during his/her transactions with the group of servers from the S^{U_i} set. In this case the TA grants U_i the pair (Q_i, W_i) of secret seeds. The U_i^{th} user's user name and password in the T_i^k transaction are calculated in the following manner $(u_{ij}, p_{ij}) = F(F^{t-k}(Q_i) \oplus F^k(W_i))$. In doing so, the pseudo random sequence $F^k(W_i)$, generated by the secret seed W_i by the k sequential compositions of the one way function F, provides the one way "lock" $F^{t-k}(Q_i)$ of the encrypted secret Q_i.

Note that the use of the different seeds Q_i^1, $Q_i^2, \ldots,$ Q_i^t in the different transactions divided from the secret string Q_i, also provide the transactions unlinkability. The reason is that the different independent seeds, e. g. Q_i^k and Q_i^l result in the different independent pseudo random sequences $F(Q_i^k)$ and $F(Q_i^l)$, respectively.

The security parameter ϵ equals the length of the seed Q_i or W_i which are used as the arguments of the F function for generating the pseudo random sequence. Note that the larger ϵ is, the higher the encryption level provided in the transactions is.

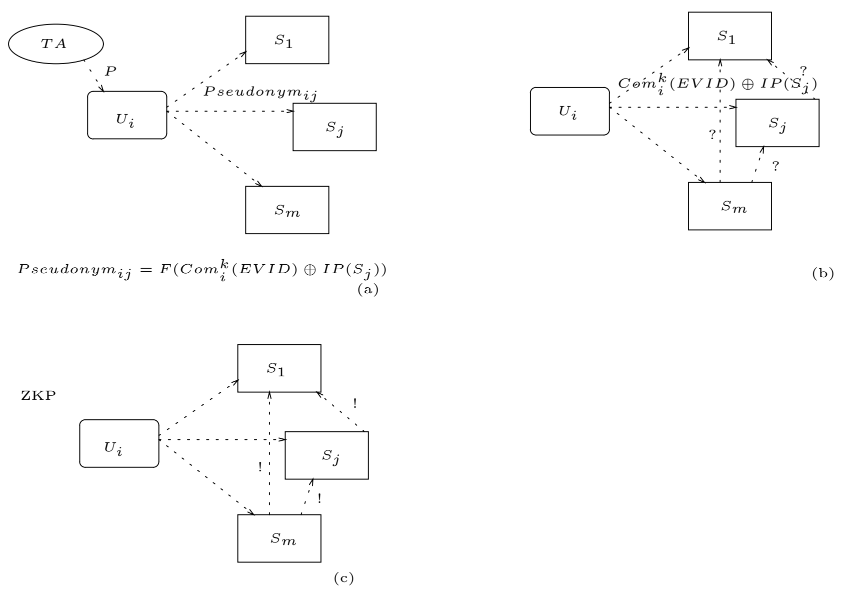
The following Theorem proves that ATP_1 satisfies the model requirements and provides the anonymous transactions in the computationally secure manner.

Theorem 1 ATP_1 satisfies the following requirements:
(R_1)-anonymity;
(R_2)-Anonymous authentication to *TA*;
(R_3)-Pseudonym uniqueness;
(R_4)-Pseudonym consistency;
(R_5)-Atomicity;
(R_6)-Optionality of the transaction mode (linkability and unlinkability of a transaction).
ATP_1 is computationally secure regarding the security parameter ϵ that determines the computational security level.
The proof of Theorem 1 is presented in the Appendix.

4 The Anonymous Transactions Protocol ATP_2

4.1 ATP_2 Description

The advanced anonymous transactions protocol ATP_2 enables the user to carry out and resolve the anonymous transactions by himself/herself. Hence, the *TA* is involved in the system only once during the initialization stage. The idea is to provide a user with an evidence which is hard to produce, say a very long prime number [6]. Only a powerful entity may have enough resources to obtain such evidences. ATP_2 is described in Figures 3 and 4. The user's U_i transaction with the servers from the corresponding set S^{U_i} is executed in the following way.



TA
P
S_1
$Pseudonym_{ij}$
U_i
S_j
S_m
$Pseudonym_{ij} = F(Com_i^k(EVID) \oplus IP(S_j))$
(a)
$Com_i^k(EVID) \oplus IP(S_j)$
?
(b)
ZKP
!
(c)

Fig. 3: ATP_2: Stages in Executing a Transaction.

```
$ATP_2$. Protocol for User $U_i$

1:  Initialization:
2:  Get $EVID$
3:  from $TA$
4:  int $k = 1$
5:  Start $k^{th}$ transaction $T_i^k$
6:      User's $U_i$ authentication
7:      to any $S_j \in S^{U_i}$
8:          for $j = 1..m$
9:              $F_{ij} = F(Com_i^k(EVID)$
                $\oplus IP(S_j))$
10:             $F_{ij} = (u_{ij} \| p_{ij})$
11:             $Pseudonym_{ij} = (u_{ij} \| p_{ij})$
12:             Send $t_{ij} = Pseudonym_{ij}$ to $S_j$
13:     End user's $U_i$ authentication
14:
15:     $k^{th}$ Transaction Resolution
16:         $Resolve_i^k = Com_i^k(EVID)$
17:         Send $Resolve_i^k$ to all $S_j$ from $S^{U_i}$
18:         Provide ZKP that $U_i$ knows $EVID$
19:         If $r_i = Confirm_i^k$
20:             return
21:             $Transaction\ Resolution$
22:         else
23:             return
24:             $Transaction\ Failure$
25: $k := k + 1$
26: if $k \prec t$ $k := k + 1$
27: else Last Transaction
```

```
$ATP_2$. Protocol for Server $S_j$

1:
2:
3:
4:
5:
6:
7:
8:
9:

10:
11:
12:
13: Upon reception of $t_{ij}$
14:     Confirm $U_i^{th}$ authentication
15: $k^{th}$ Transaction Resolution
16:     Upon reception of $Resolve_i^k$
17:     Provide ZKP $U_i$ knows $EVID$
18:         If ZKP correct
19:             Send $r_i = Confirm_i^k$ to $U_i$
20:         else
21:             Send $r_i = Reject_i^k$ to $U_i$
22:
23:
24:
25 : $k := k + 1$
26: if $k \prec t$ $k := k + 1$
27: else Last Transaction
```

Fig. 4: Anonymous Transactions Protocol ATP_2.

(a) Initialization Stage: As in the the ATP_1 case, a particular commonly known proper one way function F generates the pseudo random sequence for the anonymous user's pseudonym. An evidence $EVID$ which is hard to produce, is also granted to U_i. After the initialization stage the TA leaves the system forever (Figure 3 (a) and Figure 4, lines 1-3).

(b) In order to initiate the T_i^k transaction, U_i visits the servers from the S^{U_i} set and authenticates himself/herself to each $S_j \in S^{U_i}$ by means of the anonymous $Pseudonym_{ij} = (u_{ij}, p_{ij})$. Here u_{ij} and p_{ij} are the anonymous user name and password, as in the ATP_1 case. Now the anonymous U_i^{th} identity is computed as $Pseudonym_{ij} = (u_{ij}, p_{ij}) = F(Com_i^k(EVID) \oplus IP(S_j))$. The question marks in Figure 3 (b) mean that the servers do not know whether they are interacting with the same anonymous user. As in the ATP_1 case, $IP(S_j)$ is the S_j^{th} IP address, and F is the one way function provided to the user by the TA during the initialization stage. The parameter $Com_i^k(EVID)$ is the k^{th} commitment to the instance $EVID$ [12]. Compared to the ATP_1 protocol, the computationally hard evidence $EVID$ is granted to U_i by the TA instead of a secret random number. As in the ATP_1 case the message sent by U_i to each S_j is $t_{ij} = (u_{ij}, p_{ij})$ (Figure 3 (a) and Figure 4, lines 4-13).

(c) In order to resolve the T_i^k transaction, U_i opens and sends each $S_j \in S^{U_i}$ in the $Resolve_i^k$ message the $Com_i^k(EVID)$ string (Figure 3 (b) and Figure 4, lines 15-17). The question marks in Figure 3 (b) denote that the servers do not know yet that they interacted with the same anonymous user.

The T_i^k transaction is resolved by the user when U_i provides the computationally secure Interactive ZKP [12] that U_i is the same anonymous identity that knows $EVID$ (Figure 3 (c) and Figure 4, lines 15-24). The exclamation marks in Figure 3 (c) relate to the transaction resolution.

The following Theorem proves that ATP_2 protocol satisfies the following requirements and provides the anonymous transactions in the computationally secure manner without involving the TA.

Theorem 2 ATP_2 satisfies the following requirements:
(R_1)-anonymity;
(R_2)-Anonymous authentication to TA;
(R_3)-Pseudonym uniqueness;
(R_4)-Pseudonym consistency;
(R_5)-Atomicity.
The ATP_2 protocol's computational security level is determined by the security parameters of the ZKP performed by the user to the servers from the corresponding servers' set.
The proof of Theorem 2 is presented in the Appendix.

4.2 Possible ZKPs of Primality

In order to provide a user's permanent identity, the TA must generate a hard to produce evidence $EVID$ and its solution, so it is computationally unfeasible to produce $EVID$ and, hence to guess $EVID$ in an adversarial manner. We suggest making use of the difficulty of the prime numbers generation and the primality proof. As a matter of fact, the generation of large prime numbers is an extremely hard problem ([6]), and therefore only an extremely powerful authority such as a government is able to produce large primes. The proposed hard evidence $EVID$ is a very large prime number P.

Assume that the purpose of the user U_i is to convince the servers from the S^{U_i} set that U_i anonymously interacted with each $S_j \in S^{U_i}$ and provided a k^{th} transaction T_i^k in a legitimate manner. The user U_i acts as follows: he/she computes a commitment to P for T_i^k transaction and proves in the $T_i^{k\ th}$ second phase that P is a very large prime. We suggest the use of the ZKP proofs of primality proposed in [5] or [15]. The ZKP protocols presented in these papers are based on the randomized primality tests which provide a very high confidence level of the committed number being prime. The main advantage of these schemes is that the target number is not revealed, and only its commitment participates in the ZKP. Based on these results, the prime evidence P granted to the user by the TA is valid forever for any number of possible transactions. Hence, only commitment to P is updated for a new transaction. In ([5]) the first efficient statistical ZKP protocol for proof that a committed number is pseudo prime is presented. A prover (in our context a user) performs the correct computation of a modular addition, modular multiplication, or a modular exponentiation, where all values including the modulus are committed but not publicly available ([5]). The authors implement the randomized Lehmann primality test in Zero Knowledge (see [1]). The computational security power is based on the hardness of the discrete logarithm problem in finite groups ([19]).

The authors of [15] propose a more efficient ZKP of primality. The primality certificate is investigated based on the proof that a given number has only one prime factor and that it is square free. The algebraic settings are: Let $G = \langle g \rangle$ be a finite group of large known order Q and let h be a second group generator while the discrete logarithm $log_g h$ is unknown to the prover-user and the verifier-server. A commitment scheme in [5] and [15] is as follows: in order to commit to any element x participated in the ZKP the prover-user chooses a random number r_x and sends the commitment to x $c_x = g^x h^{r_x}$ to the verifier-sender. Given $c_x = g^x h^{r_x}$ it is computationally infeasible for the verifier or any other adversary to obtain any information about x, while it is infeasible to find different pairs $(x, r_x$ and $x', r_{x'}$ such that $c_x = g^x h^{r_x} = g^{x'} h^{r_{x'}}$ unless the prover can compute $log_h g$ (see [5], [15]). The ZKP of primality in [15] is provided in two phases: firstly, the prover proves that a committed in $c_P = g^P h^{r_P}$ number P has only one prime factor; and next that P is square free. The ZKP is based on the quadratic residues and the LaGrange theorem [1]. For example, in order to perform the secret modular computation $\forall\ a \in Q\ a^P = a(mod\ P)$, the basic secret modular multiplication, exponentiation, and quadratic residue computations are provided. The Lehmann primality test in [5] is also based on these building blocks. The ZKP proves that P is in a given range, and its upper bound may be as large as desired as presented in [15], which suits our model.

The application of ATP_2 requires the users and the servers to deal with computations that are very long and, therefore requires them to be powerful machines as well.

5 Conclusions and Extensions

We define a framework for providing anonymous transactions in computer networks. Two schemes are proposed and analyzed. The first scheme is based on an approval of each transaction by a third party, namely Trusted Authority for the transaction resolution. This scheme is flexible in the sense that the *TA* may limit the maximal number of the transactions permitted to a user by the *TA*; the transactions can be performed in both the linkable and unlinkable mode. The transactions mode is chosen by the user for himself/herself. The advanced scheme uses the *TA* in the initialization stage only, while the anonymous transactions are provided by the user without involving the *TA*. Nevertheless, this scheme is mainly of theoretical interest. The reason is that it is based on computationally expensive Zero Knowledge Proof techniques. Hence, this scheme is implementable only by very powerful computers.

References

1. E. Bach, J. Shallit, "Algorithmic Number Theory“, Volume 1: Efficient Algorithms, MIT Press, 1996.
2. M. Belenkiy, M. Chase, M. Kohlweiss, A. Lysyanskaya, "Non-Interactive Anonymous Credentials“, *IACR Cryptology ePrint Archive*, Report 2007/384.
3. J. Camenisch, A. Lysyanskaya, "A Formal Treatment of Onion Routing“, *CRYPTO 2005*, pages 169-187.
4. J. Camenisch, A. Lysyanskaya, "Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials“, *Proceedings of Crypto 2002*, Volume 2442 of LNCS.
5. J. Camenisch, M. Michels, "Proving in Zero Knowledge that a Number is the Product of Two Safe Primes“, *Basic Research in Computer Science (BRICS) Report Series*, 1998.
6. Cooperative Computing Awards, http://w2.eff/org/awards/coop.php
7. A. Beimel, S. Dolev, "Buses for Anonymous Message Delivery“, *Journal of Cryptology*, Volume 16, pages 25-39, 2003.
8. S. Dolev, R. Ostrovsky, "Xor-Trees for Efficient Anonymous Multicast and Reception“, *ACM Transactons on Information and Syste Security*, Volume 3, Number 2, pages 63-84,2000.
9. U. Feige, A. Fiat, A. Shamir, "Zero-Knowledge Proofs of Identity“, *Journal of Cryptology*, Volume 1, Number 2, Springer New York, 1988.
10. E. Gabber, P. Gibbons, Y. Matias, A. Mayer, "How to Make Personalized Web Browsing Simple, Secure, and Anonymous“, *Financial Cryptography: First International Conference*, Fc '97, Anguilla, 1997.
11. P. Golle, M. Jakobsson, A. Juels and P. Syverson, "Universal Re-encryption for Mixnets“, *The Cryptographer's Track at RSA conference*, pages 163-178, San Francisco, CA, USA, 2004.
12. O. Goldreich, *"Foundations of Cryptography“*, Volume 1, Cambridge University Press, 2003.
13. O. Goldreich, *"Foundations of Cryptography“*, Volume 2, Cambridge University Press, 2003.
14. L. Lamport, "Password Authentication with Insecure Communication“, *Communications of the ACM*, Volume 24, Number 11, 1981.
15. T. V. Lee, K. Q. Nguyen, V. Varadharajan, "How to Prove that a Committed Number is Prime“, *Lecture Notes In Computer Science*, Volume 1716, Proceedings of the International Conference on the Theory and Applications of

Cryptology and Information Security Advances in Cryptology table, pages: 208 - 218, 1999.
16. A. Lysyanskaya, R. L. Rivest, A. Sahai, S. Wolf, "Pseudonym Systems", *Selected Areas in Cryptography*, pages 184-199, 2001.
17. M. Naor, R. Ostrovsky, R. Venkatesan, M. Yung, "Perfect Zero-Knowledge Arguments for NP Using any One-Way Permutation", *Advances in Cryptology-Crypto 92 Proceedings*, Lecture Notes in Computer Science 740, Springer-Verlag, 1993.
18. W. Stallings, *"Network Security Essentials: Applications and Standards"*, Prentice Hall, 2007.
19. D. R. Stingson, *"Cryptography. Theory and Practice"*, Chapman and Hall/CRC, Third edition, 2006.

6 Appendix

Theorem 1 ATP_1 satisfies the following requirements:
(R_1)-anonymity;
(R_2)-Anonymous authentication to TA;
(R_3)-Pseudonym uniqueness;
(R_4)-Pseudonym consistency;
(R_5)-Atomicity;
(R_6)-Optionality of the transaction mode (linkability and unlinkability of a transaction).
ATP_1 is computationally secure regarding the security parameter ϵ that determines the computational security level.
Proof:

Assume that the purpose of the user $U_i \in U$ is to carry out t transactions at most with the network servers from the S^{U_i} set. Assume next that the TA permitted the user U_i t possible transactions at most with the groups of servers, and granted U_i a secret number Q_i and a one way function F. According to the ATP_1 protocol the corresponding message generated for any l^{th} transaction, $l \prec t$ with a server $S_j \in S^{U_i}$ is $t_{ij} = (u_{ij}, p_{ij})$ while $(u_{ij} \| p_{ij}) = F((Q_i^l) \oplus IP(S_j))$ (Figure 2, lines 1-4, ATP_1 Protocol for User U_i). Let us prove that ATP_1 satisfies the requirements.

(R_1)-anonymity: For any $U_i \in U$ the anonymity of the generated pseudonym $(u_{ij} \| p_{ij})$ is provided by the one way function F and the corresponding encryption $F(Q_i^l)$ of the Q_i^l sub-string of the secret number Q_i. The security parameter ϵ which is equal to the length of the seed Q_i^l, determines the computational security level. The larger ϵ is, the higher is the $ATP_1's$ security level.
Assume that the goal of the adversary A is to break the user's U_i pseudonym sent in t_{ij} message and to reveal the secret string Q_i or its sub-string Q_i^l (Figure 2, lines 6-13, ATP Protocol for User U_i). In order to succeed in his/her attempt, A must calculate F^{-1} from t_{ij}. Since F is a proper publicly known one way function, the calculation of its inverse is computationally infeasible [19].

(R_2)-Anonymous authentication to TA: During the initialization stage the TA communicates with U_i via a secure anonymous channel (Figure 2, lines 1-3, ATP_1 Protocol for User U_i). That means that the TA does not get any information about U_i^{th} identity. Moreover, the next user's transactions are resolved by the TA based only on a secret Q_i and the encryption function

F. As a matter of fact, the sub-string Q_i^l of Q_i and F contain no information about a user's identity (Figure 2, lines 14-23, ATP_1 Protocol for Server S_j).

(R_3)-Pseudonym uniqueness: The user's U_i pseudonym for any l^{th} transaction T_i^l is generated by the one way function F from the seed $Q_i^l \oplus IP(S_j)$. The seed's uniqueness is satisfied by the server's S_j IP address $IP(S_j)$ which is one of $Q_i^l \oplus IP(S_j)$ seed components. Assuming that F is a proper collision-resistant one way function ([19]), the generated pseudo random sequence is unique for any (U_i, S_j) pair.

(R_4)-Pseudonym consistency: The user's pseudonym generated by the user U_i for visiting any server $S_j \in S^{U_i}$ is the same for any repeated visit S_j during the same transaction. Hence, the user's anonymous pseudonym is consistent for any transaction.

(R_5)-Atomicity: The T_i^l transaction resolution is initiated upon U_i^{th} demand and is performed by the *TA* upon reception of the same message $Resolve_i^l = F(Q_i^l)$ from all the servers participated in the T_i^l transaction. The *TA* simultaneously verifies that $F(Q_i^l)$ is the proper encryption of the secret's Q_i sub-string Q_i^l and sends the r_i=$Confirm_i^l$ message to all servers from S^{U_i} set in parallel (Figure 2, lines 18-19, ATP_1 Protocol for Server S_j). If $F(Q_i^l)$ has been corrupted for at least one server S_j, a transaction reject is performed by the *TA* simultaneously to all servers from S^{U_i} by sending the message $r_i = Reject_i^l$ (Figure 2, lines 14-18, ATP_1 Protocol for Server S_j).

(R_6)-Optionality of the transaction mode:
Linkability of the transactions: Assume that the user U_i chooses the linkable mode. In this case U_i calculates $t+1$ sequential encryptions
$F(Q_i),\ F(F(Q_i)),\ \ldots, F^t(Q_i)$ of the secret number Q_i ([14]). Assume that the user U_i has provided $T_i^k,\ T_i^{k+1}, \ldots,\ T_i^l$ sequential transactions with the servers from the S^{U_i} set. U_i remains linkable after any transaction that follows T_i^k because each $S_j \in S^{U_i}$ can verify that the secret encryption $F^{t-k}(Q_i)$ used in the k^{th} transaction equals to the $F^{t-k}(Q_i) = F^{l-k}(F^{t-l}(Q_i))$, by repeating application of an one way function F on Q_i.
Unlinkability of the transactions: Assume that the user U_i has been granted a pair of secret seeds (Q_i, W_i). The U_i^{th} user user name and password in the T_i^k transaction are calculated as $(u_{ij}, p_{ij}) = F(F^{t-k}(Q_i) \oplus F^k(W_i))$. Assume that the purpose of a certain server S_j from the corresponding server set is to reveal whether the same anonymous identity interacted with the server before in a number of transactions, say k^{th} and l^{th}, $k \prec l$ transactions. In order to trace a user, S_j has to link the corresponding user names and passwords used in the k^{th} and l^{th} transactions, respectively. In order to compare the pseudo random sequences generated by the one way function F from the seeds $F^{t-k}(Q_i) \oplus F^k(W_i)$ and $F^{t-l}(Q_i) \oplus F^l(W_i)$, S_j must invert the external envelope F of the different seeds given. Based on the assumption that F is a one way function, its inversion is computationally infeasible by the polynomial time restricted adversary A. In such a way the pseudo random sequence $F^k(W_i)$, generated by the secret number W_i by the k sequential compositions of the one way function F with the initial seed W_i, provides the one way "lock" $F^{t-k}(Q_i)$ of the encrypted secret Q_i.
■

Theorem 2 ATP_2 satisfies the following requirements:
(R_1)-anonymity;

(R_2)-Anonymous authentication to TA;
(R_3)-Pseudonym uniqueness;
(R_4)-Pseudonym consistency;
(R_5)-Atomicity.
The ATP_2 protocol's computational security level is determined by the security parameters of the ZKP performed by the user to the servers from the corresponding servers' set.
Proof: The anonymous user's pseudonym in the ATP_2 is constructed as in the ATP_1 case. The only difference is that instead of the sequential k^{th} encryption $E_k(Q_i)$ of the secret number Q_i the commitment $Com_i^k(EVID)$ to the hard evidence $EVID$ serves as one of the arguments of the one way function F, while the second argument $IP(S_j)$ remains the same: $t_{ij} = F(Com_i^k(EVID) \oplus IP(S_j))$ (Figure 4, lines 5-13, Protocol for User U_i). Therefore, the requirements $(R_1) - (R_5)$ are satisfied as in the ATP_1 protocol (see proof of Theorem 1 from the previous section).
(R_5)-*Atomicity:* T_i^l th transaction resolution is initiated and performed by U_i by sending each $S_j \in S^{U_i}$ simultaneously the commitment to the hard instance $Com_i^k(EVID)$ in the $Resolve_i^l$ message. The transaction is resolved simultaneously while all the servers accept the ZKP that U_i surely knows the committed instance $EVID$. The T_i^l th transaction is rejected simultaneously by all servers if at least one server $S_m \in S^{U_i}$ does not accept the ZKP (Figure 4, lines 15-21, Protocol for Server S_j).

■