PPAA: Peer-to-Peer Anonymous Authentication*

Patrick P. Tsang and Sean W. Smith

Department of Computer Science
Dartmouth College
Hanover, NH 03755, USA
{patrick,sws}@cs.dartmouth.edu

Abstract. In the pursuit of authentication schemes that balance user privacy and accountability, numerous anonymous credential systems have been constructed. However, existing systems assume a client-server architecture in which only the clients, but not the servers, care about their privacy. In peer-to-peer (P2P) systems where both clients and servers are peer users with privacy concerns, no existing system correctly strikes that balance between privacy and accountability.

In this paper, we provide this missing piece: a credential system in which peers are *pseudonymous* to one another (that is, two who interact more than once can recognize each other via pseudonyms) but are otherwise anonymous and unlinkable across different peers. Such a credential system finds applications in, e.g., Vehicular Ad-hoc Networks (VANets) and P2P networks.

We formalize the security requirements of our proposed credential system, provide a construction for it, and prove the security of our construction. Our solution is efficient: its complexities are independent of the number of users in the system.

Keywords: privacy, anonymous authentication, credentials, secret handshakes, VANets, reputation systems.

1 Introduction

We live in an era where human activities happen electronically more than ever. People rely heavily on computer infrastructures, such as Web applications and peer-to-peer (P2P) networks, to share information, express opinions and trade goods. It is therefore paramount to protect the privacy of the users in these infrastructures by providing them with the option of acting anonymously, unlinkably and/or unobservably.

* This work was supported in part by the National Science Foundation under grant CNS-0524695. The views and conclusions do not necessarily represent those of the sponsors.

S.M. Bellovin et al. (Eds.): ACNS 2008, LNCS 5037, pp. 55–74, 2008.

1.1 Balancing User Privacy and Accountability

It is impractical to pursue user privacy without taking accountability into consideration. Without the fear of being identified, held responsible and punished when they abuse the services, clients are likely to misbehave due to selfishness or malice, thereby disrupting system operations and harming everyone else. Accountability has traditionally been achieved through authentication mechanisms (often followed by access control and/or auditing), which verify the identity of a client who requests a service. In the classic examples of passwords, Kerberos and standard Public Key Infrastructures (PKIs), clients have to give up their privacy to be authenticated.

Anonymous Credential Systems. In the pursuit of authentication schemes that balance privacy and accountability, numerous anonymous credential systems [15, 17], and closely related schemes such as k-times anonymous authentication (k-TAA) [33, 34], offline anonymous electronic cash (e-cash) systems [20, 14] and group signatures [21, 5] have been constructed. An anonymous credential system allows a client to be authenticated by a server as a group member anonymously and unlinkably, and yet the anonymity can be revoked when certain conditions are met. Existing systems differ in their anonymity revocation mechanisms, and hence provide different balancing points between privacy and accountability for different application settings. For example, clients can be identified when they "double-spend" in an e-cash system; their authentications become linkable[1] when they are authenticated more than k times in k-TAA. In group signatures, an authority exists and is capable of arbitrarily revoking anonymity.

1.2 The Challenge: P2P Systems

All anonymous credential systems in existence today assume a client-server architecture in which only the clients, but not the servers, care about their privacy. However, in P2P systems where both clients and servers are peer users with privacy concerns, none of the existing credential systems correctly strike that balance between privacy and accountability.

More specifically, several existing anonymous credential systems provide client accountability by empowering servers to pseudonymize clients who are otherwise anonymous, and servers can thus decide whether and/or how to serve an anonymous client depending the past behavior of the client. In all such systems, however, a client must either (1) present to all servers the very same and hence linkable pseudonym, or (2) learn the identity or at least the pseudonym of a server and then present to that server a pseudonym specific to it. In the former case, client privacy is at risk because colluding servers can link connections from the same client; in the latter, server privacy is at risk because colluding clients can link connections to the same server.

[1] Two authentication runs are linkable (by some entity) *if and only if* it is possible (for that entity) to tell whether or not the two runs are executed by the same client.

We provide below two application scenarios to motivate the user's need for privacy not just as a client, but also as a server, in P2P systems. The opposing requirements of server privacy, client privacy, server accountability and client accountability in these scenarios illustrate the non-triviality of the challenge we overcome in this paper.

Vehicular Ad-Hoc Networks (VANets). To contribute to safer and more efficient roads, vehicles in VANets constantly exchange information such as road and weather conditions among each other and with roadside base stations. Research has shown that the provision of the necessary security and privacy in VANets is critical to the users who rely on these networks [30, 12].

To protect the location privacy of the drivers when information is exchanged on the road between two vehicles, both vehicles should remain anonymous among all the vehicles in communication range. Furthermore, no one should be able to link reports by the same vehicle to different other vehicles or roadside base stations. This helps prevent a vehicle from being not only pseudonymized and thus tracked, but also deanonymized through drawing inferences from multiple reports made by the vehicle [28].

From the accountability perspective, to distinguish legitimate data from rogue data, vehicles must be authenticated when reporting sensor readings. Moreover, so that repetitive reporting of the same information can be detected, vehicles should be pseudonymous to one another (that is, vehicle X can recognize some vehicle Y reporting again, without knowing anything else about Y). For instance, in VANets in which vehicles decide when to accelerate and break based on reports collected from the network, the failure to achieve these security goals can allow an attacker to paralyze traffic and/or induce accidents.

Reputation Systems for P2P Networks. The existence of selfish users in P2P networks such as those for file sharing severely degrades system performance. Adversaries can reduce the availability of specific items in P2P networks by "poisoning" [22] them, i.e., injecting lots of decoys into the network. Reputation systems provide a game-theoretic solution to these problems by introducing incentives for users to behave well. Unfortunately, reputation systems lacking privacy can also introduce disincentives to good behavior: if a reputation system reveals the pseudonym or even the identity of the serving peers, peers might refuse to serve others so as to stay anonymous.

A privacy-preserving reputation system for P2P networks where there is no (trusted) central server should have the following properties: users are pseudonymous to one another, so that a user Carol can decide whether to serve (or be served by) another user Dave based on her past experience with Dave, without knowing his actual identity. However, assuming the registration procedures make sure that users in the system can have at most one single membership, Dave shouldn't be able to start fresh after having established a bad reputation with respect to Carol, nor can he impersonate Carol for her high reputation, potentially even spoiling her reputation through misbehavior. Finally, connections

between a peer Carol and different other peers should be unlinkable, as otherwise it might be possible for someone to trace Carol by studying those connections.

1.3 Our Contributions

In this paper, we overcome the challenge posed above by proposing the concept—and giving a construction and implementation—of *Peer-to-Peer Anonymous Authentication*, or PPAA for short, a credential system in which peers are pseudonymous to individual peers but unlinkable across different peers. More specifically, we make the following contributions:

- We rigorously define the operations of PPAA and its security and privacy requirements, during which we introduce the notion of the *Linkability Context* of an authentication scheme as a tool for a more precise reasoning about the linkability property of an authentication scheme. We also formalize the threat model in which those security requirements must be satisfied.
- We provide the first construction for PPAA. Our construction is both secure and efficient. In particular, its complexities are independent of the number of users in the system. In the extended version of this paper [36], we also report empirical performance figures of a software implementation of our construction.

Paper Organization. We review the related works in Section 2 and give an overview of our solution in Section 3. Section 4 covers the preliminary materials. In Section 5, we define the security model. We present our solution and analyze its security and efficiency in Section 6. We provide some discussions in Section 7 and conclude the paper in Section 8.

2 Related Works

We review the literature for related works, and argue why they fail to solve the problem posed in this paper. We make occasional but otherwise minimal use of mathematical notation without definition for the sake of conciseness.

***k*-Times Anonymous Authentication.** *k*-TAA [33, 29, 34, 13] and related schemes such as event-oriented linkable group/ring signatures [37, 6] are close candidates in overcoming the posed challenge. In essence, when a client Alice in these schemes is being authenticated by a server Bob, she provides Bob with a tag and convinces him that the tag is correctly formed. Bob can test if two authentications are linked to the same client by examining the associated tags.

These schemes do not solve the posed problem since authentication runs by the same user to different servers are linkable. This is because a user always uses the same tag when being authenticated by any server. More specifically, the tag of client i with secret x_i has the form of $t_i = g^{x_i}$ for some global parameter g.

We point out that, while they do not address server privacy, various *k*-TAA schemes and anonymous credential systems do provide several major ingredients

for the solution we propose in this paper. For example, our proof system for group membership uses ideas from Camenisch and Lysyanskaya [17] and Boneh et al. [10]. Also, the concept of event identifiers in this paper stems from several other existing schemes [15, 33, 37].

Secret Handshakes. Secret handshake schemes (SHSs) [7, 19, 40, 4] allow any two members of the same group to authenticate each other as a group member and share a session key without revealing their group affiliations to outsiders.

In the scheme due to Xu and Yung [40], secret handshakes are anonymous and unlinkable, but members are limited to shaking hands no more than some predefined number of times. The state-of-the-art construction [4] provides anonymity and unlinkability without such a limitation. Recently, Tsudik and Xu [38] extended secret handshakes into a multi-party and privacy-conserving setting: two or more group members can anonymously and unlinkably authenticate each other such that one's group affiliation is not revealed unless every other party's membership is ensured.

All anonymous secret handshakes proposed so far [40, 4, 38] fail to solve the posed problem. As handshakes are unlinkable, a client Alice has no way to tell if the one she is shaking hands with is the same as the one behind some earlier handshakes. As a remedy, Alice may ask the person behind the handshake to reveal a secret, e.g., a random nonce, that she leaked in their last handshake. Unfortunately, this is problematic because the person does not know which secret to reveal as Alice is anonymous. Also, one could pretend to be new by "forgetting" the secrets.

3 Our Approach

In this section, we provide an overview of our approach to solve the posed challenge.

3.1 Putting Authentication Schemes into "Linkability Context"

We first introduce the notion of the *linkability context* in authentication.

Definition 1 (Linkability Context). The *Linkability Context*, or *LC* for short, of an authentication scheme is a collection of attributes that determines the linkability of authentication runs in the scheme. In particular, two authentication runs are *linkable if and only if* the two runs are executed when the attributes in the linkability context are all in the same condition. □

In k-TAA, for instance, authentication runs by the same client at the same "time" are linkable, while runs by the same client at different times, as well as those by different clients at the same time, are not linkable. The linkability context of k-TAA is thus $LC = \{\texttt{client-ID}, \texttt{time}\}$, i.e., the collection of client identity and time.

Understanding the precise linkability context of an authentication scheme helps reason about the privacy guarantees and hence implications of the scheme. At one end of the spectrum of client privacy, in conventional authentication

schemes such as those using digital signatures, any two authentication runs are linkable. The linkability context of these schemes therefore consists of nothing, i.e., $LC = \emptyset$. At the other end of the spectrum, there are schemes such as ring authentication [31, 23] in which no two authentication runs are linkable. In this case, the linkability context is the authentication run instance, i.e., $LC = \{\texttt{authen-run-ID}\}$.

Linkability Context in PPAA. A correct choice of its linkability context is the first step towards a secure PPAA construction. In our design, the linkability context in PPAA is the collection of the *unordered* pair of client and server identity, and the event for which the PPAA authentication is executed, i.e.,

$$LC = \{\{\texttt{client-ID}, \texttt{server-ID}\}, \texttt{event-ID}\}.$$

In other words, we would like to design PPAA in such a way that authentication runs are linkable *if and only if* they are executed between the same pair of peers for the same event. In the example of VANets, if one sets the event to be *"speed on Highway I-89 on June 3rd, 2008,"* then only those PPAA-authenticated speed report made by the same vehicle to the same road-side base station on Highway I-89 on June 3rd, 2008 are linkable.

3.2 Key Ideas in Our PPAA Design

An Observation. It should have become clear now that event-oriented linkable group/ring signatures and k-TAA fail as a secure PPAA construction because server identity is not in their linkability context. It would seem that one could bring server identity into the linkability context in an event-oriented linkable group/ring signature (resp. k-TAA) by mapping an event in (resp. one "time") into the identity of a server. Consequently, LC becomes $\{\texttt{client-ID}, \texttt{server-ID}\}$ and authentication runs by the same user to different servers become unlinkable. More specifically, the tag of client i with secret x_i with respect to server j has the form of $t_{i,j} = g_j^{x_i}$, where g_j is a server-specific parameter. Tags of the same client with respect to different servers are now unlinkable thanks to the underlying intractability assumption (the Decisional Diffie-Hellman assumption).

Unfortunately, to produce a tag and prove its correctness during an authentication run in the above modified scheme, a client must now ask the server for its g_j, which can be considered its pseudonym. Even if there existed a way in which a client could compute a tag for the server without knowing the pseudonym of the server, two colluding users can easily determine if they are being authenticated by the same server. In other words, using the tag design in event-oriented linkable group/ring signatures and k-TAA, it is impossible to devise a secure authentication scheme with $LC = \{\texttt{client-ID}, \texttt{server-ID}\}$.

The Need of a Novel Tag Construct. As a result, constructing a secure PPAA requires a new tag design that possesses novel features:

- Tags must be dependent on the identity of the client, the server, and the event.

- Tags are linked *if and only if* they are produced by the same (unordered) pair of peers, and during the same event.
- Peers must be able to produce tags and prove their correctness in zero-knowledge through interacting with the other peers and without knowing the identity of the other peers.

In Section 6, we present such a tag design and how we use it to construct a secure PPAA.

4 Preliminaries

We provide the technical background necessary for understanding the rest of this paper.

Notations. A function $f(\lambda)$ is negligible if for all polynomial $p(\lambda)$, $f(\lambda) < 1/p(\lambda)$ holds for all sufficiently large λ. A function is non-negligible if it is not negligible. The probability $\Pr[E]$ of an event E is overwhelming (in some parameter λ) if $1 - \Pr[E]$ is negligible (in λ).

Let λ be a sufficiently large security parameter. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be cyclic groups of prime order p with $|p| = \lambda$ such that group operation is efficiently computable. Let g_0 and h_0 be generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively such that there is an efficiently computable isomorphism ψ from $\mathbb{G}_2$ to $\mathbb{G}_1$ with $\psi(h_0) = g_0$.

We say that $(\mathbb{G}_1, \mathbb{G}_2)$ is a bilinear group pair if there exists an efficiently computable map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, where $\mathbb{G}_T$ is also a cyclic group of prime order p, such that: $\hat{e}(A^x, B^y) = \hat{e}(A, B)^{xy}$ for all $A \in \mathbb{G}_1$, $B \in \mathbb{G}_2$ and $x, y \in \mathbb{Z}_p$, and $\hat{e}(g_0, h_0) \neq 1$.

Complexity Assumptions. The security of our solution to be presented later in this paper relies on the validity of the DDH assumption in $\mathbb{G}_1$ and the q-SDH assumption on bilinear group pair $(\mathbb{G}_1, \mathbb{G}_2)$, which we define as the following.

- *The Decisional Diffie-Hellman (DDH) problem* in $\mathbb{G}_1$: On input of a quadruple $(g_0, g_0^a, g_0^b, g_0^c) \in \mathbb{G}_1^4$, where $a, b \in_R \mathbb{Z}_p$, and $c = ab$ or $c \in_R \mathbb{Z}_q$ equally likely, output 1 if $c = ab$ and 0 otherwise. We say that *the DDH assumption* in $\mathbb{G}_1$ holds if no probabilistic polynomial time (PPT) algorithm has non-negligible advantage over random guessing in solving the DDH problem in $\mathbb{G}_1$.
- *The q-Strong Diffie-Hellman (q-SDH) problem* in $(\mathbb{G}_1, \mathbb{G}_2)$: On input of a $(q+2)$-tuple $(g_0, h_0, h_0^x, h_0^{x^2}, \cdots, h_0^{x^q}) \in \mathbb{G}_1 \times \mathbb{G}_2^{q+1}$, where $x \in_R \mathbb{Z}_p$, output a pair $(A, c) \in \mathbb{G}_1 \times \mathbb{Z}_p$ such that $A^{(x+c)} = g_0$. We say that *the q-SDH assumption* in $(\mathbb{G}_1, \mathbb{G}_2)$ holds if no PPT algorithm has non-negligible advantage in solving the q-SDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$.

The q-SDH assumption was introduced and proven to hold in generic groups [32] by Boneh and Boyen [9]. The DDH assumption in $\mathbb{G}_1$ is the also known as the *eXternal Diffie-Hellman (XDH)* assumption in $(\mathbb{G}_1, \mathbb{G}_2)$ [14, 10]. The validity of the XDH assumption implies that ψ is computationally one-way. The assumption

is known is be false on supersingular curves [25], but is conjectured to hold for the Weil or Tate pairing on MNT curves with embedded degree greater than 1 and $\mathbb{G}_1$ defined over the ground field [10].

Proofs of Knowledge. In a *Zero-Knowledge Proof-of-Knowledge (ZKPoK)* protocol [26], a prover convinces a verifier that some statement is true without the verifier learning anything except the validity of the statement. Σ-protocols are a special type of three-move ZKPoK protocols. They can be converted into non-interactive *Signature Proof of Knowledge (SPK)* schemes that are secure in the *Random Oracle (RO)* Model [8] (in the sense of Indistinguishability against chosen-message attacks, or IND-CMA [27]).

In many anonymous credential systems, a client uses an SPK scheme to prove in zero-knowledge to a server her possession of a credential issued by the Group Manager when being authenticated by a server. The SPK schemes differ in these systems, which accounts for the differences in privacy and accountability guarantees and complexity assumptions. The SPK schemes we will use in our solution are based on the ZKPoK protocol due to Boneh and Boyen [10].

We follow the notation introduced by Camenisch and Stadler [18] for the various ZKPoK protocols. For example, $PK\{(x) : y = g^x\}$ denotes a ZKPoK protocol that proves the knowledge of an integer x such that $y = g^x$ holds, where y and g are elements of some group $G = \langle g \rangle$. Using this notation, a ZKPoK protocol can be described by just pointing out its aim while hiding all the details. Moreover, we denote by $SPK\{(x) : y = g^x\}(M)$ the SPK scheme converted from the above ZKPoK protocol.

5 Model

This section formalizes PPAA. The entities involved in PPAA are the Group Manager (GM) and a set of peer users, or simply peers. The GM is responsible for registering peers. A peer can be a client, a server, or both. Clients are interested in accessing services provided by servers and servers are willing to serve the clients, as long as their privacy and accountability requirements are satisfied.

5.1 System Operations

Operations that take place in PPAA include the GM setting up the system (Setup) and registering peers into the system (Registration), and peers authenticating one another (Authentication) and testing if two authentication runs are linked (Linking). We highlight that only Setup and Registration involve a centralized authority, namely the GM; Authentication requires no centralized authority, which is a crucial property necessary for PPAA to be applicable to P2P systems with scalability.

The syntax for these operations are given as follows.

- Setup is a *Probabilistic Poly-Time (PPT)* algorithm invoked by the GM. On input a sufficiently large security parameter λ, the algorithm outputs GM's

secret key `gsk` and the group public key `gpk`. The GM stores `gsk` privately and publishes `gpk` to the public. `gpk` is an implicit input to all the algorithms below.

- Registration is a two-party multi-round protocol between the $\mathsf{Register}^P$ PPT algorithm invoked by a peer and the $\mathsf{Register}^{GM}$ PPT algorithm invoked by the GM. The additional input to $\mathsf{Register}^{GM}$ is the GM's secret key `gsk`. Upon successful termination of a protocol run, $\mathsf{Register}^P$ outputs a credential, which the peer stores privately, and by doing so becomes a registered peer in the system.
- Authentication is a two-party multi-round protocol between the $\mathsf{Authenticate}^I$ PPT algorithm invoked by a registered peer Alice (as the *Initiator*, i.e. the one who initiates the protocol) and the $\mathsf{Authenticate}^R$ PPT algorithm invoked by another registered peer Bob (as the *Responder*). The common input to both parties is an event identifier `eid` upon which they have already agreed.[2] The additional inputs to $\mathsf{Authenticate}^I$ and $\mathsf{Authenticate}^R$ are Alice's credential and Bob's credential, respectively.
  A protocol run terminates successfully *if and only if* both algorithms output a tag, in which case we say that the authentication is *successful* and that Alice and Bob are mutually authenticated with one another, during an event with identifier `eid`. When we say that a peer Carol is involved in an authentication without specifying her role, then Carol can be either the initiator or the responder in that authentication.
- Linking is a (possibly probabilistic) poly-time algorithm any peer can invoke. On input two tags $\mathtt{tag}_1$ and $\mathtt{tag}_2$, the algorithm outputs a boolean value of either `linked` or `not-linked`.
  In the former (resp. the latter) case, the two tags, and also the two successful authentication runs from which the tags are resulted, are said to be *linked* (resp. *not linked*).
  Semantically, a peer Carol uses this algorithm to pseudonymize other peers with which she has mutually authenticated: for any two successful authentication runs during the same event, she thinks she is mutually authenticating with the same peer *if and only if* the two authentication runs are linked.

Any construction of PPAA must be correct:

Definition 2 (Correctness). An PPAA construction is correct if it has *authentication correctness* and *linking correctness*:

- *Authentication Correctness.* If all entities in PPAA are honest (i.e. they all follow the system's specification), then, with overwhelming probability, any authentication between any two registered peers is successful.
- *Linking Correctness.* If all entities in PPAA are honest, then, with overwhelming probability, in any two successful authentication involving any registered peer Carol, the two tags output by Carol are linked *if and only if*, in those two authentications, both the event identifiers and the other peers involved are identical. □

[2] In the VANet example given in Section 3.2, the `eid` can be `20080603||I-89||speed`.

5.2 Security Requirements

Roughly speaking, a PPAA construction is secure if it satisfies the following security requirements. (A formal definition can be found in the extended version of this paper [36].)

Mis-authentication Resistance. Mis-authentication occurs when two peers successfully complete mutual authentication, but only one of them is an honest and registered peer. A secure PPAA construction must be resistant to mis-authentication.

For example, this property prevents vehicles in VANets from believing (malicious) data from rogue sensors.

Peer Accountability. To subvert peer accountability, a coalition of $n \geq 1$ registered but malicious peer(s) attempts to run more than n successful mutual authentication involving the same honest peer Carol during the same event such that the tags Carol outputs in those authentication are all pairwise unlinked. A secure PPAA construction requires that no adversary can succeed in such an attempt.

In the example of P2P networks, this prevents a peer from starting fresh after having established a bad reputation with respect to another peer.[3]

Peer Privacy. To subvert the privacy of an honest peer Carol involved in an authentication potentially executed with a malicious peer, the adversary, potentially with the GM's help, attempts to:

- deanonymize Carol in individual protocol runs, and/or
- pseudonymize Carol in protocol runs with different peers and/or during different events.

A secure PPAA construction requires that no adversary can succeed in any of the above attempts.

As an example, this ensures that communications of a vehicle in VANets with different other vehicles or roadside base stations cannot be linked.

Framing Resistance. An honest peer Carol is *framed* when another honest peer Dave thinks that he is mutually authenticating with the same peer in two successful authentication runs, even though Carol is involved in exactly one of them. A secure PPAA construction requires that no adversary, even with the help of the GM, can frame an honest peer.

In the example of P2P reputation systems, this makes sure that peers can't impersonate other peers with high reputation.

6 Our Solution

We begin this section with a presentation of our first attempt to construct PPAA, which, although insecure by itself, illustrates our tag design as the core component of our full and secure PPAA construction. Then we proceed to present our actual PPAA construction. In the extended version of this paper [36], we discuss our implementation of the construction and its empirical performance evaluation.

[3] This assumes that a peer can't register more than once. We will discuss this issue further in Section 7.

6.1 Our First Attempt

We call our first attempt Basic-PPAA.

Parameters. Let $\mathbb{G}_1$ be a group as described in Section 4 in which the DDH assumption holds. Let $H : \{0,1\}^* \to \mathbb{G}_1$ be a secure cryptographic hash function. Event identifiers are strings of any length.

Credentials. Each user is given by the GM one credential $\texttt{cred}$ in the form $\texttt{cred} = (A, x, y) \in \mathbb{G}_1 \times \mathbb{Z}_p^2$, where $x, y \in_R \mathbb{Z}_p$ and A is distinct in all credentials.

Tags. In Basic-PPAA, a tag is the output of a function f that takes as inputs the credential of an initiating peer $\texttt{cred}_1 = (A_1, x_1, y_1)$, the credential of a responding peer $\texttt{cred}_2 = (A_2, x_2, y_2)$ and an event identifier $\texttt{eid}$. The function is defined as follows:

$$f : (\texttt{cred}_1, \texttt{cred}_2, \texttt{eid}) \mapsto \texttt{tag} \doteq \{\tau_1, \tau_2\}, \text{ where } \begin{cases} \tau_1 = A_1^{x_2} H(\texttt{eid})^{y_1}, \\ \tau_2 = A_2^{x_1} H(\texttt{eid})^{y_2}. \end{cases}$$

Thus, a tag is a set of two $\mathbb{G}_1$ elements.

The Skeleton Protocol. The following steps describe a protocol run between an initiating peer Alice with credential $\texttt{cred}_1 = (A_1, x_1, y_1)$ and a responding peer Bob with credential $\texttt{cred}_2 = (A_2, x_2, y_2)$ during an event with identifier $\texttt{eid}$. When the protocol terminates, Alice and Bob output a tag.

1. Alice $\to$ Bob: $\langle U_1, V_1 \rangle = \langle A_1^{r_1}, H(\texttt{eid})^{r_1} \rangle$, where $r_1 \in_R \mathbb{Z}_p$.
2. Bob $\to$ Alice: $\langle U_2, V_2, W_2 \rangle = \langle A_2^{r_2}, H(\texttt{eid})^{r_2}, U_1^{x_2} V_1^{y_2} \rangle$, where $r_2 \in_R \mathbb{Z}_p$.
3. Alice $\to$ Bob: $\langle W_1, \tau_1 \rangle = \langle U_2^{x_1} V_2^{y_1}, W_2^{1/r_1} \rangle$.
4. Bob $\to$ Alice: $\langle \tau_2 \rangle = \langle W_1^{1/r_2} \rangle$.
5. Alice and Bob both output $\texttt{tag} = \{\tau_1, \tau_2\} = f(\texttt{cred}_1, \texttt{cred}_2, \texttt{eid})$ and terminate.

Properties. The tags and the skeleton protocol given above have the following desirable properties:

1. Two tags $\texttt{tag} = f(\texttt{cred}_1, \texttt{cred}_2, \texttt{eid})$ and $\texttt{tag}' = f(\texttt{cred}'_1, \texttt{cred}'_2, \texttt{eid}')$ are the same *if and only if* $\{\texttt{cred}_1, \texttt{cred}_2\} = \{\texttt{cred}'_1, \texttt{cred}'_2\}$ and $\texttt{eid} = \texttt{eid}'$, with overwhelming probability.
2. The protocol view of Alice $\langle \texttt{cred}_1, \texttt{eid}, r_1, U_2, V_2, W_2, \tau_2 \rangle$ can be simulated (computationally indistinguishably) by Alice if she is given $\texttt{tag}$. In other words, Alice learns no knowledge other than $\texttt{tag}$ from running the skeleton protocol. Similarly, Bob learns no knowledge other than $\texttt{tag}$ from running the skeleton protocol.
3. The tag produced by a peer Alice for another peer Bob during an event is indistinguishable from the tag produced by any peer for Bob during a different event; it is also indistinguishable from the tag produced by Alice for a different peer during the same event.

The validity of these properties are straightforward provided that the DDH assumption in $\mathbb{G}_1$ holds. We thus omit the proof.

Remark. If not all entities are honest, Basic-PPAA results in an insecure PPAA construction. For instance, users can be authenticated without asking the GM for a credential, dishonest users may use an arbitrary credential instead of the one given by the GM to get away from being linked and a malicious GM can frame clients.

6.2 Our PPAA Construction

We now enumerate our PPAA construction. It can be thought of as the result of securing Basic-PPAA by adding to it all necessary mechanisms to force the entities to behave honestly, such as by accompanying each step in the skeleton protocol with a SPK scheme that proves the correctness of the step.

Parameters. In addition to those in Basic-PPAA, our PPAA construction has the following parameters. Let $\mathbb{G}_2$ be a group as described in Section 4 such that $(\mathbb{G}_1, \mathbb{G}_2)$ is a bilinear group pair in which the q-SDH assumption holds. Let ℓ be a sufficiently large security parameter of size polynomial in λ. Let $g_1, \ldots, g_5 \in \mathbb{G}_1$ be generators of $\mathbb{G}_1$ such that the relative discrete logarithms among $g_1, \ldots, g_5$ and g_0 (from Section 4) are unknown. Let $\hat{H} : \{0,1\}^* \rightarrow \mathbb{Z}_p$ be a secure cryptographic hash function. $\hat{H}$ is utilized by the various SPKs in the construction.

Setup. The GM randomly chooses $\gamma \in_R \mathbb{Z}_p$ and computes $w = h_0^\gamma \in \mathbb{G}_2$. The group secret key is $\texttt{gsk} = (\gamma)$ and the group public key is $\texttt{gpk} = (w)$.

Registration. At the successful termination of a run of this protocol between a user Alice and the GM, Alice obtains a credential `cred` in the form of $\texttt{cred} = (A, e, x, y, z) \in \mathbb{G}_1 \times \mathbb{Z}_p^4$ such that $A^{e+\gamma} = g_0 g_1^x g_2^y g_3^z$. The private input to the GM is his group secret key `gsk`. The protocol proceeds as follows.

1. The GM sends $\langle N_0 \rangle$ to Alice, where $N_0 \in_R \{0,1\}^\ell$ is a random challenge.
2. Alice sends $\langle C, \Pi_0 \rangle$ to the GM, where $C = g_1^x g_2^y g_3^{z'} \in \mathbb{G}_1$ is a commitment of $(x, y, z') \in_R \mathbb{Z}_p^3$ and Π_0 is a signature proof of knowledge of

$$SPK \left\{ (x, y, z') : C = g_1^x g_2^y g_3^{z'} \right\} (M)$$

   on message $M = N_0 || C$, which proves the correctness of C. We will refer to the above SPK as SPK_0.
3. The GM terminates with `failure` if the verification of Π_0 returns `invalid`. Otherwise the GM sends $\langle A, e, z'' \rangle$ to Alice, where $e, z'' \in_R \mathbb{Z}_p$ and

$$A = (g_0 C g_3^{z''})^{\frac{1}{e+\gamma}} \in \mathbb{G}_1$$

4. Alice computes $z = z' + z''$. She terminates with `failure` if $\hat{e}(A, w h_0^e) \neq \hat{e}(g_0 g_1^x g_2^y g_3^z, h_0)$. Otherwise she outputs $\texttt{cred} = (A, e, x, y, z)$ as her credential.

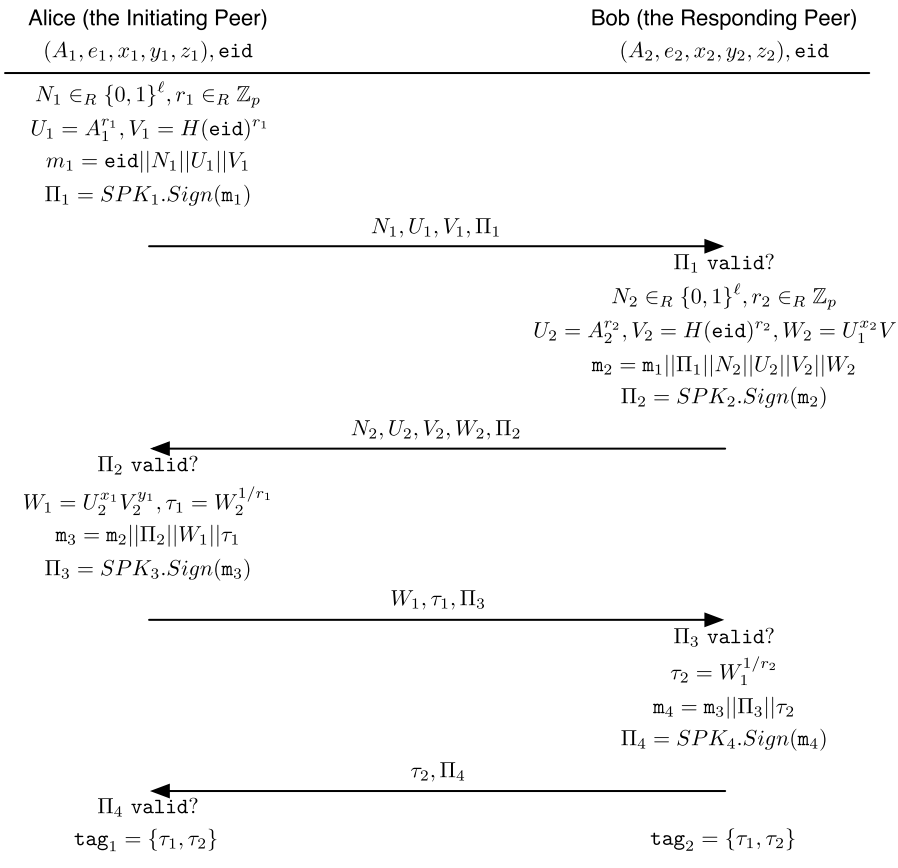

Fig. 1. The Authentication Protocol

We remark that the security of the system requires that no two instances of the Registration protocol may run concurrently. To enforce this rule, the GM registers users one after the other.

Authentication. Alice (as the initiator) and Bob (as the responder) would like to mutually authenticate with each other during an event with identifier $\texttt{eid} \in \{0,1\}^*$. The common input to both Alice and Bob is eid. The private input to Alice and Bob is their own credentials $(A_1, e_1, x_1, y_1, z_1)$ and $(A_2, e_2, x_2, y_2, z_2)$ respectively. The following describes the steps in the 4-round protocol for authentication.

1. Alice sends $\langle N_1, U_1, V_1, \Pi_1 \rangle$ to Bob, where:
   - $N_1 \in_R \{0,1\}^\ell$, $r_1 \in_R \mathbb{Z}_p$,
   - $U_1 = A_1^{r_1} \in \mathbb{G}_1$, $V_1 = H(\texttt{eid})^{r_1} \in \mathbb{G}_1$, and
   - Π_1 is a signature proof of knowledge of

$$SPK\left\{(A, e, x, y, z, r) : \begin{array}{c} A^{e+\gamma} = g_0 g_1^x g_2^y g_3^z \quad \wedge \\ U_1 = A^r \ \wedge \ V_1 = H(\texttt{eid})^r \end{array}\right\}(M)$$

on message $M = \mathtt{m}_1 = \mathtt{eid}||N_1||U_1||V_1 \in \{0,1\}^*$, which Alice can produce using her knowledge of $(A_1, e_1, x_1, y_1, z_1, r_1)$. We will refer to the above SPK as SPK_1.

2. Bob terminates with `failure` if verification of Π_1 returns `invalid`. Otherwise he sends $\langle N_2, U_2, V_2, W_2, \Pi_2 \rangle$ to Alice, where:
   - $N_2 \in_R \{0,1\}^\ell$, $r_2 \in_R \mathbb{Z}_p$,
   - $U_2 = A_2^{r_2} \in \mathbb{G}_1$, $V_2 = H(\mathtt{eid})^{r_2} \in \mathbb{G}_1$, $W_2 = U_1^{x_2} V_1^{y_2}$, and
   - Π_2 is a signature proof of knowledge of

$$SPK \left\{ \begin{array}{c} A^{e+\gamma} = g_0 g_1^x g_2^y g_3^z \quad \wedge \\ (A, e, x, y, z, r) : V_2 = H(\mathtt{eid})^r \ \wedge \ U_2 = A^r \ \wedge \\ W_2 = U_1^x V_1^y \end{array} \right\} (M)$$

   on message $M = \mathtt{m}_2 = \mathtt{m}_1||\Pi_1||N_2||U_2||V_2||W_2 \in \{0,1\}^*$, which Bob can produce using his knowledge of $(A_2, e_2, x_2, y_2, z_2, r_2)$. We will refer to the above SPK as SPK_2.

3. Alice terminates with `failure` if verification of Π_2 returns `invalid`. Otherwise she sends $\langle W_1, \tau_1, \Pi_3 \rangle$ to Bob, where:
   - $W_1 = U_2^{x_1} V_2^{y_1} \in \mathbb{G}_1$, $\tau_1 = W_2^{1/r_1} \in \mathbb{G}_1$, and
   - Π_3 is a signature proof of knowledge of

$$SPK \left\{ \begin{array}{c} A^{e+\gamma} = g_0 g_1^x g_2^y g_3^z \quad \wedge \\ (A, e, x, y, z, r) : U_1 = A^r \ \wedge \ V_1 = H(\mathtt{eid})^r \ \wedge \\ W_1 = U_2^x V_2^y \ \wedge \ W_2 = \tau_1^r \end{array} \right\} (M)$$

   on message $M = \mathtt{m}_3 = \mathtt{m}_2||\Pi_2||W_1||\tau_1 \in \{0,1\}^*$, which Alice can produce using her knowledge of $(A_1, e_1, x_1, y_1, z_1, r_1)$. We will refer to the above SPK as SPK_3.

4. Bob terminates with `failure` if verification of Π_A returns `invalid`. Otherwise he sends $\langle \tau_2, \Pi_4 \rangle$ to Alice, where:
   - $\tau_2 = W_1^{1/r_2}$, and
   - Π_4 is a signature proof of knowledge of

$$SPK \left\{ (r) : W_1 = \tau_2^r \ \wedge \ V_2 = H(\mathtt{eid})^r \right\} (M)$$

   on message $M = \mathtt{m}_4 = \mathtt{m}_3||\Pi_3||\tau_2 \in \{0,1\}^*$, which Bob can produce using his knowledge of (r_2). We will refer to the above SPK as SPK_4.

   Bob outputs $\mathtt{tag}_2 = \{\tau_1, \tau_2\}$ and terminates.
5. Alice terminates with `failure` if verification of Π_4 returns `invalid`. Otherwise she outputs $\mathtt{tag}_1 = \{\tau_1, \tau_2\}$ and terminates.

Figure 1 is a diagrammatic representation of the protocol.

Linking. On input two tags $\mathtt{tag}_1, \mathtt{tag}_2$, this algorithm returns `linked` if they are equal and `not-linked` otherwise.

6.3 SPK Instantiation

The instantiation of SPK_0 to SPK_4 and their computational costs in terms of the number of pairing computation and multi-exponentiations (multi-EXPs)[4] can be found in the extended version of this paper [36].

6.4 Analysis

Our PPAA construction has correctness, which is a straightforward consequence of the correctness of the skeleton protocol and the correctness of the various SPK schemes. We omit the proof for conciseness.

Security. The security of our construction hinges on the correctness of the skeleton protocol and the security properties of the various SPK schemes surrounding it. We now state the following theorem. (Its proof is sketched in the extended version of this paper [36].)

Theorem 1 (Security). Our proposed PPAA construction is secure in the random oracle model if the XDH assumption and the q-SDH assumption hold in $(\mathbb{G}_1, \mathbb{G}_2)$. □

Complexities. Our solution scales extremely well: all operations have constant computational and communication complexities, regardless on the number of peers, events and authentication runs. Registration is a one-time process per user in the system. Linking involves only an equality testing of two sets of two $\mathbb{G}_1$ elements.

Authentication is the dominating operation, thus we provide a more detailed analysis on its costs. Alice, the initiating peer, needs to do an SPK_1 and an SPK_3 signing, and an SPK_2 and an SPK_4 verification. The number of $\mathbb{G}_1$ multi-EXPs, $\mathbb{G}_T$ multi-EXPs and pairings are 24, 10 and 4 respectively. Some of these operations can be precomputed before the the start of an authentication; with precomputation, those numbers become 10, 4 and 2 respectively. Bob, the responding peer, needs to an SPK_2 and an SPK_4 signing, and an SPK_1 and an SPK_3 verification. The number of $\mathbb{G}_1$ multi-EXPs, $\mathbb{G}_T$ multi-EXPs and pairings are 22, 11 and 5 respectively. With precomputation, they become 14, 8 and 4. In addition to these calculation, Alice and Bob also need to compute several $\mathbb{G}_1$ multi-EXPs during the protocol.

Table 1 summarizes the computational costs for Alice and Bob.

7 Discussion

Resilience to Sybil Attacks. Sybil attacks [24] are attacks during which an individual entity masquerades as multiple simultaneous identities. Any authentication mechanisms including PPAA must defend Sybil attacks launched against

[4] A multi-EXP computes the product of exponentiations faster than performing the exponentiations separately. We assume that one multi-EXP operation multiplies up to 3 exponentiations.

Table 1. Timing complexity of the Authentication protocol

	Number of Operations (without precomputation)		
	$\mathbb{G}_1$ multi-EXPs	$\mathbb{G}_T$ multi-EXPs	Pairings
Alice (the Initiator)	12 (28)	4 (10)	2 (4)
Bob (the Responder)	16 (26)	8 (11)	4 (5)

user registration. Approaches exist to ensure that only legitimate users can register and that no legitimate user can register more than once. They include *trusted certification* such as X.509 [1], *resource testing*, where resources could be IP addresses or "friendship" in social networks or PGP-like web of trust, *recurring costs* imposed by cryptographic puzzles or CAPTCHAs [39], and *trusted devices* with certain degree of tamper-resistance, such as Trusted Platform Modules (TPMs) [35].

The practicality of the above approaches depends on the application scenarios. In the example of VANets, the Department of Motor Vehicles (DMV) can play the role of the GM with little overhead. Additionally, the makers of the vehicles can install a trusted device preloaded with a credential in each of vehicle they manufacture. In the example of P2P systems over a public network such the Internet, demonstrating the possession of IP addresses is a pragmatic and thus more popular approach, even though it does not have the highest resilience to Sybil attacks.

Revocation. Any practical authentication mechanism must allow for credential revocation. In the settings of PPAA, one might want to revoke a credential because the peer user in possession of that credential is compromised or misbehaving. For example, in VANets, the credential issued to a vehicle should be revoked when the vehicle is reported to have been stolen. Revocation allows for easier identification and thus tracking of stolen vehicles while maintaining the privacy of other vehicles as stolen cars with revoked credentials can no longer be anonymously authenticated by, e.g. a highway toll booth.

Our construction of PPAA can be modified in a straightforward manner to allow for credential revocation by adopting existing standard techniques [16, 11]: Alice and Bob verifiably encrypt part of their credentials during SPK_1 and SPK_2 respectively during the authentication under the public key of an entity usually referred to as the Revocation Manager. Now in addition to the original authentication, Alice and Bob have to convince one another that they have not been revoked. In the approach of verifier-local revocation [11], each user keeps a list of revoked users; in the approach of dynamic accumulators [16], each non-revoked user updates their credential when someone else's has been revoked.

Authenticated Key-exchange. The authentication protocol in PPAA can be easily turned into an authenticated Diffie-Hellman key-exchange. Specifically, Alice additionally includes in $\mathtt{m}_1$ an element g_0^a with $a \in_R \mathbb{Z}_p$ in Step 1 of the authentication protocol, while Bob additionally includes in $\mathtt{m}_2$ an element g_0^b with $b \in_R \mathbb{Z}_p$ in Step 3. When the protocol terminates, both of them can derive a shared session

key as $g_0^{ab} = (g_0^a)^b = (g_0^b)^a$. Since $\mathtt{m}_1$ and $\mathtt{m}_2$ are signed with SPK_1 and SPK_2 respectively, Alice and Bob can use the session key to establish a confidential channel with the same privacy and accountability guarantees as in PPAA.

Blending Secret handshakes into PPAA. As discussed, anonymous SHSs such as Ateneise et al.'s [4] do not provide the linkability desired by the servers. On the other hand, PPAA leaks the initiating peer's group affiliation to any responding peer who might not be a group member. Hence, each of them has its advantage over the other. Fortunately, one can enjoy the advantages of both by composing the two schemes. Specifically, two group members first execute an anonymous secret handshake to authenticate the group membership of one another and establish a secure channel, then they execute an PPAA authentication within that channel.

Furthermore, carrying out PPAA authentication within a secure channel has the additional benefit of preventing eavesdroppers from linking authentication traffic.

Fairness. In our PPAA construction, a malicious responding peer Bob might decide to stop after receiving Alice's protocol message at step 3 of the authentication protocol so that he could learn Alice's tag without Alice being able to learn his. The revealing of tags between Alice and Bob is thus not guaranteed to be fair in our construction.

Borrowing ideas from optimistic fair exchange [2, 3], one could augment fairness to PPAA by modifying it as follows. Alice requires Bob to additionally send a verifiable encryption of r_2 under the public key of some Trusted Third Party (TTP) in step 2 also that in case Bob stops before step 4, Alice can still reconstruct the tag with the help of the TTP. However, such a modification puts Bob's privacy at risk, as the collusion between Alice the TTP can identity Bob. We leave the exploration of how to provide fairness without sacrificing privacy as future work.

8 Conclusion

In this paper, we have introduced *Peer-to-Peer Anonymous Authentication* (PPAA), a credential system that correctly balances user privacy and accountability in P2P systems where not just clients but also servers are concerned with their privacy. We have shown that such a credential system finds applications in many P2P systems such as VANets. We have presented the first PPAA construction, which is both secure and very efficient.

Acknowledgments

The authors would like to thank Man Ho Au for his suggestion on an initial tag design, the anonymous reviewers for their insightful reviews, and the members in the Security Reading Group at Dartmouth College (SRG@Dartmouth)[5] who discussed this paper for their helpful feedback.

[5] `https://wiki.cs.dartmouth.edu/srg/`

References

1. Adams, C., Farrell, S.: Internet X.509 Public Key Infrastructure Certificate Management Protocols. Internet Engineering Task Force: RFC 2510 (1999)
2. Asokan, N., Schunter, M., Waidner, M.: Optimistic protocols for fair exchange. In: ACM Conference on Computer and Communications Security, pp. 7–17 (1997)
3. Asokan, N., Shoup, V., Waidner, M.: Optimistic fair exchange of digital signatures (extended abstract). In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 591–606. Springer, Heidelberg (1998)
4. Ateniese, G., Blanton, M., Kirsch, J.: Secret Handshakes with Dynamic and Fuzzy Matching. In: NDSS, The Internet Society (2007)
5. Ateniese, G., Camenisch, J., Joye, M., Tsudik, G.: A practical and provably secure coalition-resistant group signature scheme. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 255–270. Springer, Heidelberg (2000)
6. Au, M.H., Susilo, W., Yiu, S.-M.: Event-oriented *k*-times revocable-iff-linked group signatures. In: Batten, L.M., Safavi-Naini, R. (eds.) ACISP 2006. LNCS, vol. 4058, pp. 223–234. Springer, Heidelberg (2006)
7. Balfanz, D., Durfee, G., Shankar, N., Smetters, D.K., Staddon, J., Wong, H.-C.: Secret Handshakes from Pairing-Based Key Agreements. In: IEEE Symposium on Security and Privacy, pp. 180–196. IEEE Computer Society Press, Los Alamitos (2003)
8. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: Proceedings of the 1st ACM conference on Computer and communications security, pp. 62–73. ACM Press, New York (1993)
9. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
10. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M.K. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
11. Boneh, D., Shacham, H.: Group signatures with verifier-local revocation. In: ACM Conference on Computer and Communications Security, pp. 168–177. ACM Press, New York (2004)
12. Calandriello, G., Papadimitratos, P., Hubaux, J.-P., Lioy, A.: Efficient and robust pseudonymous authentication in vanet. In: VANET 2007: Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks, pp. 19–28. ACM Press, New York (2007)
13. Camenisch, J., Hohenberger, S., Kohlweiss, M., Lysyanskaya, A., Meyerovich, M.: How to win the clonewars: efficient periodic n-times anonymous authentication. In: ACM Conference on Computer and Communications Security, pp. 201–210. ACM Press, New York (2006)
14. Camenisch, J., Hohenberger, S., Lysyanskaya, A.: Compact e-cash. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 302–321. Springer, Heidelberg (2005)
15. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (2001)
16. Camenisch, J., Lysyanskaya, A.: Dynamic accumulators and application to efficient revocation of anonymous credentials. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 61–76. Springer, Heidelberg (2002)

17. Camenisch, J., Lysyanskaya, A.: A signature scheme with efficient protocols. In: Cimato, S., Galdi, C., Persiano, G. (eds.) SCN 2002. LNCS, vol. 2576, pp. 268–289. Springer, Heidelberg (2003)
18. Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups (extended abstract). In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 410–424. Springer, Heidelberg (1997)
19. Castelluccia, C., Jarecki, S., Tsudik, G.: Secret handshakes from ca-oblivious encryption. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 293–307. Springer, Heidelberg (2004)
20. Chaum, D.: Blind signatures for untraceable payments. In: CRYPTO, pp. 199–203 (1982)
21. Chaum, D., van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991)
22. Christin, N., Weigend, A.S., Chuang, J.: Content availability, pollution and poisoning in file sharing peer-to-peer networks. In: ACM Conference on Electronic Commerce, pp. 68–77. ACM Press, New York (2005)
23. Dodis, Y., Kiayias, A., Nicolosi, A., Shoup, V.: Anonymous identification in ad hoc groups. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 609–626. Springer, Heidelberg (2004)
24. Douceur, J.R.: The sybil attack. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, pp. 251–260. Springer, Heidelberg (2002)
25. Galbraith, S.D., Rotger, V.: Easy Decision-Diffie-Hellman Groups. LMS Journal of Compuation and Mathematics 7, 201–218 (2004)
26. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM J. Comput. 18(1), 186–208 (1989)
27. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Comput. 17(2), 281–308 (1988)
28. Krumm, J.: Inference attacks on location tracks. In: LaMarca, A., Langheinrich, M., Truong, K.N. (eds.) Pervasive 2007. LNCS, vol. 4480, pp. 127–143. Springer, Heidelberg (2007)
29. Nguyen, L., Safavi-Naini, R.: Dynamic k-times anonymous authentication. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 318–333. Springer, Heidelberg (2005)
30. Raya, M., Hubaux, J.-P.: Securing vehicular ad hoc networks. Journal of Computer Security 15(1), 39–68 (2007)
31. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001)
32. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg (1997)
33. Teranishi, I., Furukawa, J., Sako, K.: k-times anonymous authentication (extended abstract). In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 308–322. Springer, Heidelberg (2004)
34. Teranishi, I., Sako, K.: k-times anonymous authentication with a constant proving cost. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 525–542. Springer, Heidelberg (2006)
35. TPM Work Group. TCG TPM Specification Version 1.2 Revision 103. Technical report, Trusted Computing Group (2007)
36. Tsang, P.P., Smith, S.W.: PPAA: Peer-to-peer anonymous authentication (extended version). Technical Report TR2008-615, Department of Computer Science, Dartmouth College (April 2008)

37. Tsang, P.P., Wei, V.K., Chan, T.K., Au, M.H., Liu, J.K., Wong, D.S.: Separable linkable threshold ring signatures. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT 2004. LNCS, vol. 3348, pp. 384–398. Springer, Heidelberg (2004)
38. Tsudik, G., Xu, S.: A flexible framework for secret handshakes. In: Danezis, G., Golle, P. (eds.) PET 2006. LNCS, vol. 4258, pp. 295–315. Springer, Heidelberg (2006)
39. von Ahn, L., Blum, M., Hopper, N.J., Langford, J.: Captcha: Using hard ai problems for security. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 294–311. Springer, Heidelberg (2003)
40. Xu, S., Yung, M.: k-anonymous secret handshakes with reusable credentials. In: ACM Conference on Computer and Communications Security, pp. 158–167. ACM Press, New York (2004)