Yan, Zheng; Wang, Pu; Feng, Wei

A novel scheme of anonymous authentication on trust in Pervasive Social Networking

Published in:
Information Sciences

DOI:
10.1016/j.ins.2018.02.037

Published: 01/06/2018

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

A Novel Scheme of Anonymous Authentication on Trust in Pervasive Social Networking

Zheng Yan*
State Key Lab on Integrated Services Networks, School of Cyber Engineering, Xidian University, Xi'an 710071, China
Department of Communications and Networking, Aalto University, Espoo 02150, Finland
zyan@xidian.edu.cn; zheng.yan@aalto.fi

Pu Wang
State Key Lab on Integrated Services Networks, School of Cyber Engineering, Xidian University, Xi'an 710071, China
wangpu03@gmail.com

Wei Feng
State Key Lab on Integrated Services Networks, School of Cyber Engineering, Xidian University, Xi'an 710071, China
bjxsakya@qq.com

* Corresponding author: Zheng Yan (email: zyan@xidian.edu.cn)

Abstract
Pervasive Social Networking (PSN) supports various social activities at any time and in any places with the heterogeneous networks. Trust plays a crucial role in securing PSN. Authenticating trust anonymously is becoming an attractive approach to ensuring trustworthy and privacy-preserving social networking. However, the literature still lacks serious studies on this topic, especially for PSN systems. In this paper, we propose a novel scheme to authenticate PSN node trust in an anonymous and semi-distributed manner. The scheme allows one or multiple Authorized Parties (APs) to announce up-to-date aggregate lists of Integrated Node Trust (INT) for certificateless authenticating trust with anonymity, unforgeability, unlinkability and conditional traceability. In addition, multiple APs can cooperate to flexibly conduct trust authentication without significantly increasing computational overhead. Aggregate signature verification further improves scheme efficiency. Security proof, performance analysis and evaluation show that our scheme is effective with regard to security, privacy preservation, computational complexity, communication cost, efficiency, scalability and flexibility.

Keywords: Anonymous Authentication, Privacy Preservation, Social Network Security, Trust Evaluation

1 INTRODUCTION

In recent years, heterogeneous networks organized by the Internet, mobile cellular networks and self-organized Mobile Ad hoc Networks (MANETs) have received special attention due to their capabilities of establishing an instant communication platform for time-critical or mission-critical applications. As a concrete application example, Pervasive Social Networking (PSN) supports instant social activities anywhere and at any time in an intelligent and context-aware manner by switching among heterogeneous networks based on user demands. Not only people socially connected, but also strangers physically in proximity can form a social

group to perform various social activities in a pervasive way.

PSN is an essential complement to the Internet online social networking with the properties of "anywhere and anytime", thus very valuable for mobile users. It can be formed by leveraging Bluetooth or Wi-Fi to support peer-to-peer networks and relay networks without the involvement of the Internet and mobile cellular networks. PSN is especially valuable when the Internet online social networks are temporarily unavailable or costly to access. The current trend of PSN services is decentralizing since a PSN node can be both a service provider and a consumer. For example, Uber [49], Didi car-sharing [50] and eRideShare [51] help people share car riding in a convenient way. PSN can also provide instant recommendations, fast assistance, and urgent rescues in practice.

One of the most important issues in PSN is its security, trust, and privacy [38, 41]. Nowadays, social networking is widely applied into official communications, which requests advanced security guarantee and enhanced privacy preservation. However, PSN is specific in terms of the availability of heterogeneous networks and lacks trust among PSN nodes in nature. Since trust helps people overcome perceptions of uncertainty and risk, people need to evaluate the trust of communication parties in order to make a wise decision for engaging in subsequent "trusted social behaviors" [26, 38]. Obviously, one attractive and direct way to secure PSN and to preserve privacy is to anonymously authenticating trust among PSN nodes.

However, few existing studies explored this issue in the literature [8, 22, 31, 32, 42, 43, 44]. Traditional privacy enhancement techniques usually apply node pseudonyms in social networking to hide real node identities and avoid privacy tracking [22, 31, 32]. However, adopting and frequently changing pseudonyms negatively influence the efficiency of node authentication and key management, as well as trust management. Every time the pseudonym changes, an authorized party needs to generate and certify a new public/private key pair for later authentication and verification. Moreover, the trust evaluated based on old pseudonyms should be at least mapped to the new one. Otherwise, the system could probably suffer from Sybil attacks.

Anonymous authentication can ensure secure communications and simultaneously protecting user privacy by resisting modification attacks and impersonation attacks. It consists of two essential security checks: message integrity check and node identification check. Existing solutions fall into two main categories: traditional public-key-infrastructure (PKI)-based digital signature schemes [22, 32] and group signature based schemes [1, 7, 18, 20, 27]. Nevertheless, most of the existing schemes failed to satisfy the security requirements of PSN due to its specific characteristics. Group revocation is still a problem in group signature schemes. Meanwhile, negative correlation between privacy and security [11] brings an additional challenge: the more privacy achieved, the harder to gain non-repudiation and accountability [16]. To address the above issue, Yan et al. [40] proposed an anonymous authentication scheme based on trust value in PSN from the point of view that PSN nodes can authenticate each other without knowing node identities. This scheme depends on a centralized Trust Authority (TA) to issue a list of trust values to guarantee anonymous communications among nodes. And a backup token is temporarily used for extending anonymous communications when TA is not available. Based on this scheme, both user privacy and trustworthy social networking can be achieved. However, this scheme cannot support a distributed PSN topology where multiple TAs served by some individual PSN nodes exist. The limited validity period of the backup token affects the flexibility of the whole system.

At present, we are still facing a number of challenges in terms of anonymous authentication on trust. First, how to ensure the anonymity and unlinkability of message originators, but offer conditional traceability in case any disputes happen? Second, how to make the scheme compatible with PSN architecture and topology, which can be either centralized or distributed? Third, how to flexibly support trust authorization played by any number of parties? Based on our previous studies, such an authentication mechanism for PSN is still missing in the literature.

In this paper, we propose a novel scheme to anonymously authenticate node trust for securing PSN system

and assisting user social decision. It allows one or multiple Authorized Parties (APs) played by a TA or PSN nodes to announce up-to-date aggregate lists of Integrated Node Trust (INT) for certificatelessly authenticating trust. According to the position of the trust value in the aggregation list of INT, a receiver node can authenticate the sender's trust anonymously and certificatelessly. In addition, multiple APs can cooperate to flexibly achieve trust authentication without significantly increasing computational overhead with the support of aggregate signature verification.

The proposed scheme is original and differs substantially from existing schemes that either authenticate pseudonyms or apply group signature to preserve identity privacy [1, 7, 18, 20, 22, 27, 31, 32]. First, it supports building up social trust relationships in PSN by authenticating trust in an anonymous manner. Based on anonymous authentication on trust, we effectively solve the challenges of group revocation and balance between privacy and security by offering conditional traceability. Second, the previous work heavily depends on a centralized AP to guarantee anonymous communications among nodes. This seriously affects scheme flexibility. However, our scheme allows one or multiple APs played by a TS or some PSN nodes to announce up-to-date aggregate lists of INT. In case the TS is not available, the PSN nodes can still communicate with each other for social networking by negotiating and selecting some PSN nodes as APs. In this case, a recommender system based on user feedback and comments can be applied to select an AP [38, 39]. Meanwhile, multiple APs can cooperate to achieve comprehensive and sophisticated trust evaluation. The proposed scheme can adaptively fit into PSN system structure and flexibly support any number of APs. In addition, this scheme can be further applied into other application fields, e.g., Vehicular Ad hoc Networks (VANETs) and Unmanned Aerial Vehicle Networks (UAVNETs) to establish a trust relationship without invading identity privacy. Specifically, the contributions of this paper can be summarized as below:

- We propose a novel authentication scheme that supports trustworthy PSN with privacy preservation by authenticating node trust in a somehow distributed manner and verifying node signatures in an anonymous way. We advocate to replace traditional identity authentication by authenticating trust in order to preserve privacy.
- To the best of our knowledge, the proposed scheme is one of the first to realize certificateless and anonymous authentication on trust in PSN and can support multiple trust issuers. In particular, the scheme can support three kinds of application scenarios appropriate to the specific characteristics of PSN and compatible with its topology. In addition, we apply aggregate signature verification to reduce the computational and storage costs of our scheme.
- We prove the security of the proposed scheme and perform simulation-based evaluation and comparison to show its advantages, effectiveness and efficiency.

The rest of the paper is organized as follows. Section 2 gives a brief overview of related work. Section 3 introduces a system and threat model and our design goals. We describe the detailed design of the proposed scheme in Section 4, followed by security proof and performance evaluation in Section 5. Finally, a conclusion is presented in the last section.

2 Related Work

In this section, we give a brief review on related work with regard to message authentication and batch signature verification, anonymous authentication, group signature and other miscellaneous work.

2.1 Message Authentication and Batch Signature Verification

Some message authentication schemes adopted PKI and Certificate Revocation List (CRL) to authenticate and verify messages [13, 34, 47]. In a PKI system, message and identity authentication is performed by checking whether the certificate of a sender is included in a currently valid CRL and verifying the signature of message. In [47], Zhu et al. applied Hash Message Authentication Code (HMAC) to guarantee the integrity

of messages and avoid time-consuming CRL check in VANETs. They used batch group authentication to reduce computation overhead and adopted cooperative message authentication among entities to further alleviate authentication burden. Hao et al. also used cooperative authentication to reduce the number of messages that each node needs to verify in VANETs [13]. However, they did not consider revoking check time, which consumes much time in CRL checking. Wasef and Shen [36] proposed an Expedite Message Authentication Protocol (EMAP) for VANETs, which replaces CRL checking by an efficient revocation checking process through a keyed HMAC. The key used in calculating the HMAC is only shared among non-revoked nodes. This method increases the load of key management. To reduce the time of signature verification, Wasef and Shen [35] and Zhang et al. [46] both employed batch group signature verification based on the properties of bilinear pairing operation, in which a large number of messages can be authenticated in a timely manner. However, the above schemes cannot anonymously authenticate trust values. Their applicability in PSN needs further investigation [21, 33, 36, 47].

2.2 Anonymous Authentication

Pseudonym-based authentication [22, 31, 32] has been proposed to help mobile nodes communicate without revealing their real identities. However, its computation cost grows linearly with a load of communications. The public-private key pair should be updated each time when the node pseudonym is changed, thus the computational overhead increases linearly with the number of applied pseudonyms. Some studies suggested performing authentication based on a centralized party to reduce the burden of mobile nodes in MANET [38, 39], while others executed authentication at individual Wireless Body Area Network (WBAN) and VANET nodes [25, 32]. Both schemes suffer from scalability and message loss problems, as any one entity (the node or the central party) is solely responsible for the key generation and/or verification. This leads to scalability issues when the density of communications goes high and the scale of networking expands. By applying identity-based encryption for encrypting packets and group signature, Emura et al. [9] proposed a secure and anonymous communication protocol for anonymous user authentication. In our proposed scheme, we can apply either a centralized party (e.g., TS) or several PSN nodes or both to flexibly support certificateless anonymous authentication on trust in order to overcome the weakness of previous work [22, 31, 32]. Our scheme avoids verifying public key certificates and uses some proxy entities to conceal user identities.

Lindell [23] formally defined the requirements of anonymous authentication, which has been widely studied in VANETs and WBAN. Lin and Li proposed a cooperative message authentication scheme for VANETs [21]. When a vehicle generates an integrated signature, other vehicles can verify an evidence token from a Trusted Authority (TA) in order to know whether the vehicle truly verifies the messages it claimed. The work in [45] introduces a batch verification scheme for communications between vehicles and Roadside Units (RSUs). However, it requires the involvement of RSUs in message authentication. Shao, et al. [34] proposed an authentication protocol for VANETs in a decentralized group model by using a new group signature scheme to achieve a threshold authentication. All of the above anonymous authentication methods in VANETs depend on TA or RSUs to some degree. They cannot support node authentication in a distributed manner. He et al. [14] proposed an anonymous authentication scheme in WBANs to meet security requirements in specific medical applications. Wu et al. [37] presented an anonymous authentication scheme in WBANs that can resist various kinds of attacks. However, mobility and communication range of WBANs is far smaller than PSN, so existing solutions in WBAN may not be appropriate to be applied into PSN.

2.3 Group Signature

Group signature can also be used for anonymous authentication [1, 7, 18, 20, 22, 27, 31, 32, 34, 35]. Chaum and Van Heyst firstly introduced this concept in 1991 [7]. Group signature allows a member of a group to anonymously sign a message on behalf of the group and can achieve anonymity, unforgeability, traceability,

and unlinkability. But the group signature needs a group manager in charge of adding/removing group members and responsible for revealing an original signer in case of any disputes. It was recently applied in VANETs [34, 35]. However, it may not be suitable to be applied in PSN. First, a group of instant social communications is generally established in a temporal and dynamic way. It is hard to select a trustworthy node to play as a group manager. Second, PSN switches its network access automatically according to practical demands. The network access adaptation requests that keys for anonymous authentication should be managed in a distributed way by PSN nodes. But considering online social networking structure and the need to trace node identities in case of disputes, a centralized party is also preferred for key management [10]. System design expects a hybrid and holistic solution. Third, efficient revocation is still a challenge in the field of group signature studies [10].

2.4 Others

Trust can be applied to support anonymous authentication. An enhanced distributed reputation system was proposed for anonymous authentication in MANET [6]. The distributed reputation system monitors activities of nodes along a routing path and evaluates their levels of trust. The system provides an efficient mechanism for misbehavior detection and guarantees anonymity by using reputation to find a trustworthy route in MANET.

Certificateless authentication on trust has been seldom studied in the literature. Guan et al. proposed a Joint Authentication and Topology Control (JATC) scheme to increase resource utilization and throughput capacity of a network [12]. The authentication protocol is based on hash chains and Merkle Tree. Liu et al. [25] presented two certificateless remote authentication protocols to preserve the privacy of Wireless Body Area Network (WBAN) users. Nevertheless, there is little research work about anonymous authentication based on trust.

Our past work considers how to control PSN communication data access based on node trust levels [42, 43, 44]. We assumed that the trust level issued by an authorized party could be authenticated. But we did not consider how to provide anonymous authentication based on node trust. The work presented in this paper solves the open issue that has not been investigated in our previous work [10, 40]. This paper is a continuous study in order to overcome its shortcomings as described in Section 1.

3 Problem Statements

3.1 System and Threat Model

We consider a PSN system involving two kinds of entities, as shown in Fig. 1, the PSN nodes that interact with each other for instant social communications; a centralized Trusted Server (TS), which has sufficient capabilities and is trustworthy to provide social networking services, identity management and trust management. In addition, TS or some PSN nodes (e.g., Wi-Fi access points and base stations) that are more stable and reliable than other nodes can serve as Authorized Parties (APs). APs evaluate trust values of nodes with sufficient information about nodes. As integrity and privacy of some instant social communications are crucial, it is important to authenticate node trust in an anonymous way for trustworthy communications and privacy preservation. To save computation and storage costs, PSN nodes may resort to TS in the cloud [19] through the Internet to manage real identities, pseudonyms, keys and trust relationships. In case the TS is not available, the PSN nodes can still communicate with each other by making some PSN nodes play as APs.

We assume that TS is reliable and trustworthy for preserving privacy of nodes due to business incentives. The communications between the nodes and TS are secure by applying an existing security protocol such as OpenSSL. Storages in PSN nodes and TS are secure. PSN nodes may not be trustworthy. Anonymity in PSN is expected because some nodes may maliciously track other nodes' private information, e.g., locations. In addition, each node prefers communicating with trustworthy nodes. The work presented in this paper focuses

on anonymous authentication on trust. We assume trust can be evaluated as a concrete value based on our previous work about context-aware trust/reputation generation in PSN [38, 41].

3.2 Why Applying TS

Applying TS fits the system topology of PSN. PSN can be supported by either online or ad hoc social-networking services. A trusted server normally offers the online social networking. The ad hoc social networking is self-maintained by the PSN nodes. However, nodes can also communicate with the TS when the Internet connectivity is available. Applying TS is compatible with the online social networking structure and can support instant social networking. The most important is that TS is able to extract the real identity of a node for the purpose of sanction and penalty when a dispute occurs. In addition, setting up TS makes the designed system easily adopted by business stakeholders. For example, a cloud social service provider can operate the TS to offer PSN services with trust, security and privacy management.

3.3 Design Goal

For anonymous authentication on trust in PSN, our design should achieve the following security and performance goals: 1) privacy preservation and anonymous authentication on trust; 2) unforgeability on message signature; 3) anonymity and unlinkability with regard to node identification and recognition; 4) conditional traceability in case disputes; 5) low computational complexity and overhead; 6) improved scalability to support large-scale PSN and flexibility to handle various PSN scenarios. In what follows, we define some basic properties used for performance evaluation on privacy preservation.

***Anonymity*:** The standard [48] defines: "*[anonymity] ensures that a user may use a resource or service without disclosing its identity. The requirements for anonymity provide protection of the user identity.*" Accordingly, we consider the anonymity as the property that the authentication cannot reveal any information of a node's real identity.

Unlinkability: From [30]: "*[unlinkability] ensures that a user may make multiple uses of resources or services without others being able to link these uses together. It requires that users and/or subjects are unable to determine whether the same user caused certain specific operations in the system.*" In our scheme, unlinkability is defined that an adversary cannot recognize whether two messages linked with the identical trust values are sent from the same nodes.

Conditional traceability: Refer to [1, 7], "*full-traceability ensures that all signatures, even those created by the collision of multiple users and a group manager, can be used to trace to a member of forging coalition.*" In our scheme, traceability is supported only when the TS is involved in the PSN.

4 The Proposed Scheme

4.1 System Structure

Fig. 1 illustrates a system structure of PSN. At each PSN node, a Pervasive Social Networker (e.g., a Facebook mobile client) provides a user interface of social networking. One of its important functions is to authenticate other nodes' trust values during PSN communications. A Communicator communicates with AP in a secure way. It requests and receives the INT from AP. The AP also distributes an up-to-date aggregate list of INT with its signature. A Trust Evaluator evaluates the node trust that is evolved based on the social data locally accumulated [38, 41]. A Trust Processor generates a one-off public-private key pair based on its current INT. A Dataset stores all data related to the above functions in the node in a safe manner (e.g., based on Trusted Computing Platform technology). In addition, a Node Profile Manager is used to maintain node user's personal information. It can communicate with TS to register the node into the system and manage related credentials.

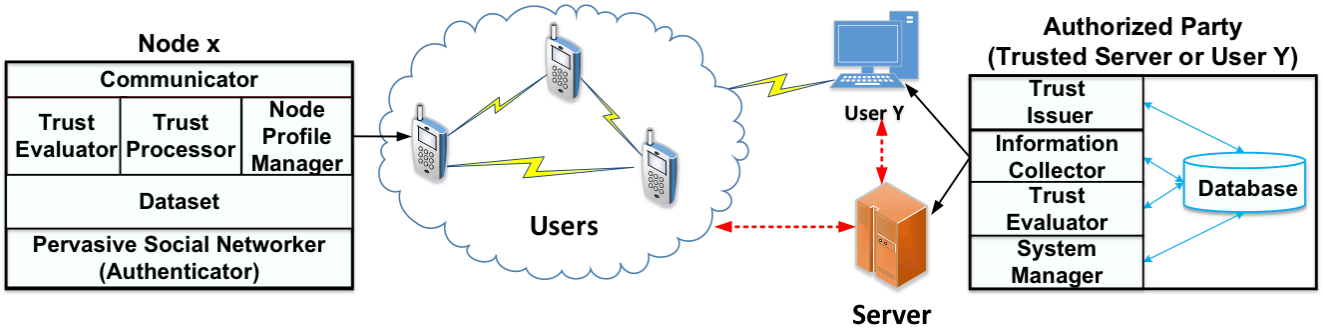

Fig.1 System Structure

At the AP, a Trust Evaluator assesses the trust values of nodes and detects malicious ones. A Trust Issuer issues an aggregate list that contains INT hash values of all nodes to PSN nodes periodically or by request. An Information Collector collects and processes social networking records from the nodes and saves results into a Database. In case that the AP is TS, a System Manager handles node registration, manages keys and system credentials. The Database of TS also saves the trust value of each node, its real identity and long-term key pair. Non-TS APs can communicate with TS in a more stable and reliable way than normal nodes.

The trust values are generated based on the social networking records collected by APs. The TS that performs as one of APs, assigns an initial trust value to a node during its registration. All APs evaluate trust values of other nodes initiated from their initial values when the trust values are going to expire or when a node requests trust value update. If the trust value is re-issued, the AP inserts its corresponding INT into the up-to-date aggregate list at a right position after removing its old value. Then, the AP announces the list to all nodes. All APs synchronously broadcast their up-to-date aggregate lists of INT.

4.2 Preliminaries and Notations

In this sub-section, we introduce the preliminaries of our scheme. For easy reference, we summarize the notations used in this paper in Table 1.

Bilinear map works as the basis of our proposed scheme. Based on the properties of bilinear pairing operations (Weil [4] or Tate pairing [29]), we design the scheme to avoid verifying public key certificates and achieve aggregate signature verification.

Let $\mathbb{G}$ be a cyclic additive group generated by P and $\mathbb{G}_T$ be a cyclic multiplicative group. $\mathbb{G}$ and $\mathbb{G}_T$ have the same prime order q, i.e., $|\mathbb{G}| = |\mathbb{G}_T| = q$. Let $e: \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map, which satisfies the following properties [4, 12, 13]:

- *Bilinear*: For all P, Q, $R \in \mathbb{G}$, and $a, b \in \mathbb{Z}_p$, $e(Q, P+R) = e(P+R, Q) = e(P, Q) \cdot e(R, Q)$. In particular, $e(aP, bP) = e(P, bP)^a = e(aP, P)^b = e(P, P)^{ab}$.
- *Non-degenerate*: There exist $P, Q \in \mathbb{G}$ such that $e(P, Q) \neq 1_{\mathbb{G}_T}$.
- *Computable*: There is an efficient algorithm to compute $e(P, Q)$ for any P, $Q \in \mathbb{G}$.

Assume that the inversion and multiplication in $\mathbb{G}$ can be computed efficiently. To prove the security of our scheme, we need the following intractable problems:

- *Computational Diffie-Hellman problem (CDH)*: Given $P, aP, bP \in \mathbb{G}$ for $a, b \in \mathbb{Z}_p$, compute abP.
- *Decision Diffie-Hellman problem (DDH)*: The DDH problem in $\mathbb{G}$ is to distinguish between the distributions <P, aP, bP, abP> and <P, aP, bP, cP> where a, b, c are randoms in $\mathbb{Z}_p$ and P is a random in $\mathbb{G}$.

The group that possesses such a map e is called a bilinear group, on which the DDH problem is easy to solve while the CDH problem is believed hard [4]. For example, given P, aP, bP, $cP \in \mathbb{G}$ and any a, b,

$c \in \mathbb{Z}_p$, there exists an efficient algorithm to determine $ab = c \bmod q \Leftrightarrow e(P, cP) = e(aP, bP)$. But the CDH in $\mathbb{G}$ can still be hard while there is no algorithm that can compute $abP \in \mathbb{G}$ with non-negligible probability within polynomial time.

In our scheme, all PSN nodes must register themselves into TS before joining the PSN. The TS is in charge of checking node identity and providing a long-term public/private key pair for each node. In addition, the TS sets up the system parameters and preloads registered nodes with public parameters (PK).

TABLE 1
NOTATIONS DESCRIPTION

Symbols	Description
PK	The system public parameters shared among all system entities;
(PK_{TS}, SK_{TS})	The public/private key pair of TS;
s_x	The secret between TS and node x;
N_x	The node x;
(PK_x, SK_x)	The long-term public/private key pair of N_x;
TV_x	The short-lived trust value of N_x;
T_TV_x	The validity period of TV_x;
AC_TV_x	The authentication code of TV_x for generating a unique INT value $h(TV_x, AC\_TV_x)$;
$Cert\_PK_x$	The certificate of PK_x issued by TS;
(U_x, V_x)	The one-off public/private pair key of N_x;
DEK_x	The session key between node N_x and TS;
$h()$	The one-way hash function, e.g., SHA-1;
$H_1()$	The hash function, such that $H_1: \mathbb{G} \to \{0,1\}^n$;
$H_2()$, $H_3()$	The map-to-group function, such that $H_2: \{0,1\}^n \to \mathbb{G}$, $H_3: \{0,1\}^* \to \mathbb{G}$;
Ha, Ha_AP_y	The up-to-date aggregate list of INT hashes issued by TS/AP_y.

4.3 Scheme Algorithms

The proposed scheme contains a number of algorithms to realize its basic functionalities. Concretely, after system setup and node registration, APs issue the INT value and the aggregate list of INT hashes (in short, the aggregate list) to each PSN node. Based on the INT, the node generates its one-off key pair for signing its messages. The receiver node verifies the message either individually or in an aggregate measure. In what follows, we introduce the main algorithms of the proposed scheme.

***SystemSetup*:** Give security parameter $k \in Z^+$, the CDH parameter [3] generator $\mathcal{G}$ generates prime q, cyclic additive group $\mathbb{G}$, cyclic multiplicative group $\mathbb{G}_T$ of order q, integer field $\mathbb{Z}_p$ and admissible bilinear map $e: \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. TS chooses a random generator $P \in \mathbb{G}$, generates random $SK_{TS} \in \mathbb{Z}_p$ and sets $PK_{TS} = SK_{TS} \cdot P \in \mathbb{G}$. Meanwhile, it chooses some cryptographic hash function $h: \{0,1\}^* \to \{0,1\}^n$, $H_1: \mathbb{G} \to \{0,1\}^n$, $H_2: \{0,1\}^n \to \mathbb{G}$ $H_3: \{0,1\}^* \to \mathbb{G}$. We view h, H_1, H_2 as random oracles. The system parameter is $PK = \{q, \mathbb{G}, \mathbb{G}_T, e, n, P, PK_{TS}, h, H_1, H_2, H_3\}$. The master-key of TS is SK_{TS}.

***NodeRegistration*(N_x):** PSN node N_x registers at TS with its real identity. The TS chooses random $SK_x \in \mathbb{Z}_p$ as the long-term private key of N_x and sets its long-term public key $PK_x = SK_x \cdot P \in \mathbb{G}$. TS also provides certificate $Cert\_PK_x = PK_x \cdot SK_{TS}$ on PK_x. This certificate is only used when the node requests the TS to issue its trust value.

***IssueTrustValuebyTS*(N_x)**: If the period of the current trust value is expired or the node requests, the TS re-evaluates N_x's trust value and issues it to N_x with an authentication code AC_TV_x. For issuing the trust value, node N_x sends a random number r_1 and its certificate $Cert\_PK_x$ to TS. After authenticating PK_x, TS chooses r_2 as its share to establish a shared session key DEK_x between N_x and itself. This process can

be achieved by adopting a Diffie-Hellman key agreement protocol. After that, TS sends parameters $\{h(TV_x||AC\_TV_x),\ T\_TV_x,\ s_x,\ Q_x = s_x \cdot P\}$ to N_x, where T_TV_x is the expiration time of TV_x.

***IssueTrustValuebyNode*(N_x)**: Node N_y can also play as an AP (AP_y), which can evaluate the trust values of other nodes in PSN. Similarly, by adopting a Diffie-Hellman key agreement protocol, AP_y can issue $\{h(TV_x\_AP_y||AC\_TV_x\_AP_y), T\_TV_x\_AP_y\}$ to N_x, where $T\_TV_x\_AP_y$ is the expiration time of $TV_x\_AP_y$. In this case, N_y can also be authenticated with $Cert\_PK_y$ by node N_x

***One-offKeyPairGeneration1*($h(TV_x||AC\_TV_x)$)**: With $h(TV_x||AC\_TV_x)$, N_x can generate a one-off anonymous public/private key pair (U_x and V_x respectively). Algorithm 1 describes the one-off anonymous key pair generation only for a single AP.

ALGORITHM 1

ONE-OFF ANONYMOUS KEY PAIR GENERATION FOR A SINGLE AP

Input: $i_x = h(TV_x||AC\_TV_x)$
Output: $U_x(U1_x$ and $U2_x)$ and $V_x(V1_x$ and $V2_x)$

i) Computes the one-off public key U_x as
$U1_x = i_x \cdot a \cdot Q_x$; $\quad U2_x = h(i_x) \oplus H_1(i_x \cdot a \cdot Q_x)$
where parameter a is a random nonce and $\oplus$ is an XOR operation;
ii) Compute the corresponding private key V_x as
$V1_x = U1_x \cdot s_x$; $\quad V2_x = H_2(U1_x||U2_x) \cdot s_x$

In order to generate a unique key pair, N_x could change the random nonce "a" each time for a new one-off public/private key pair. Thus, even for the same "$i_x = h(TV_x||AC\_TV_x)$", it is possible to generate different key pairs for achieving advanced privacy, e.g., using distinct key pairs for signing and verifying distinct messages. But this strategy could consume more computational resources. In practice, the PSN node can set its own key pair update policy according to its privacy demand. The lower requirement of privacy is, the longer the period for updating keys and a lower computational cost can be achieved.

***One-offKeyPairGeneration2*($\{h(TV_x\_AP_y||AC\_TV_x\_AP_y)\}$)**: If there are multiple APs in the system. With all $\{h(TV_x\_AP_y||AC\_TV_x\_AP_y)\}$, N_x can generate an anonymous one-off public/private key pair (vector $\overrightarrow{U_x\_AP}$ and V_x_AP respectively) to sign its messages. Algorithm 2 describes the one-off anonymous key pair generation when multiple APs exist in the system. The same as the previous algorithm, N_x can change the random nonce "a" each time for gaining high security.

ALGORITHM 2

ONE-OFF ANONYMOUS KEY PAIR GENERATION FOR MULTIPLE APS

Input: $\overrightarrow{i_x\_AP} = (i_{x_{AP_1}}, i_{x_{AP_2}}, \cdots, i_{x_{AP_Y}}) = (h(TV_x\_AP_1||AC\_TV_x\_AP_1), \cdots, h(TV_x\_AP_Y||AC\_TV_x\_AP_Y))$
Output: $\overrightarrow{U_x\_AP}$ ($\overrightarrow{U1_x\_AP}$ and $\overrightarrow{U2_x\_AP}$) and $V_x\_AP(V1_x\_AP$ and $V2_x\_AP)$

i) Compute one-off public key $\overrightarrow{U_x\_AP}$ as
$\overrightarrow{U1_x\_AP} = \overrightarrow{i_x\_AP} \cdot a \cdot Q_x$ $\quad \overrightarrow{U2_x\_AP} = \overrightarrow{h(i_x\_AP)} \oplus H_1(\overrightarrow{i_x\_AP} \cdot a \cdot Q_x)$
where parameter a is a random nonce and $\oplus$ is an XOR operation; and
$\overrightarrow{U1_x\_AP} = (U1_x\_AP_1, U1_x\_AP_2, \cdots, U1_x\_AP_Y), \overrightarrow{U2_x\_AP} = (U2_x\_AP_1, U2_x\_AP_2, \cdots, U2_x\_AP_Y)$
ii) Compute corresponding private key V_x_AP as
$V1_x\_AP = |\overrightarrow{U1_x\_AP}| \cdot s_x$ $\quad V2_x\_AP = H_2(|\overrightarrow{U1_x\_AP}|||\overrightarrow{U2_x\_AP}|) \cdot s_x$
where, $|\overrightarrow{U1_x\_AP}| = U1_x\_AP_1 + U1_x\_AP_2 + \cdots + U1_x\_AP_Y$, $|\overrightarrow{U2_x\_AP}| = U2_x\_AP_1 + U2_x\_AP_2 + \cdots + U2_x\_AP_Y$

***GenerateSignature* (m, V_x *or* V_x_AP)**: When node N_x wants to send message m, it computes signature $sign_x(m) = V1_x + H_3(m) \cdot s_x + V2_x$ or $sign_x(m) = V1_x\_AP + H_3(m) \cdot s_x + V2_x\_AP$ on m using private key V_x or V_x_AP respectively. Then N_x sends m to other nodes with the following format $\{U_x||m||Q_x||sign_x(m)\}$ or $\{\overrightarrow{U_x\_AP}||m||Q_x||sign_x(m)\}$.

***AggregateSignature* ($\{m||sign_x(m)\}$)**: By applying the concept of aggregate signatures introduced by

Boneh et al. [3], we can combine multiple signatures by computing $\sigma = \prod_{i=1}^{K} sign_i(m_i) \in \mathbb{G}$ for getting a single aggregate signature at a receiver. It is especially attractive for mobile devices for reducing the computation cost, e.g., sensors and mobile phones.

AggregateListofTrustValues1 **(**$\{h(TV_x||AC\_TV_x)\}$**)**: TS periodically generates the list Ha of the INT hashes: $Ha = \{h(Q_1||h(i_1)), \cdots, h(Q_x||h(i_x)), \cdots\}$. Then it issues $\{Ha||sign_{TS}(Ha)\}$ to all nodes.

AggregateListofTrustValues2 **(** $\{h(TV_x\_AP_y||AC\_TV_x\_AP_y)\}$ **)**: In case there are multiple APs, AP_y periodically generates $Ha\_AP_y = \{h(Q_1||h(i_1\_AP_y)), \cdots, h(Q_x||h(i_x\_AP_y)), \cdots\}$. Then it issues $\{Ha\_AP_y||sign_{AP_y}(Ha\_AP_y)\}$ signed with its private key to all nodes.

TrustValueListUpdate **(**Ha *or* Ha_AP_y**)**: Each time AP re-evaluates trust to get a new trust value, it inserts the hashes of new INT values into Ha or Ha_AP_y at right positions. Similarly, when a trust value reaches its expiry time, its corresponding INT is removed from the list in AP. The newly updated Ha or Ha_AP_y is then issued to all PSN nodes. In addition, all APs synchronously broadcast their aggregate trust lists periodically.

Verification **(**U_x, m, Q_x, $sign_x(m)$**)**: When a node receives messages, the receiver first computes $h(i_x) = U2_x \oplus H_1(U1_x)$ extracted from U_x and calculates $h(Q_x||h(i_x))$ to authenticate the trust value of N_x based on its position of in list. If there are multiple APs in the system, the receiver computes the vector $\overrightarrow{h(\imath_x\_AP)}$ from $\overrightarrow{U_x\_AP}$ as $\overrightarrow{h(\imath_x\_AP)} = \overrightarrow{U2_x\_AP} \oplus \overrightarrow{H_1(U1_x\_AP)}$. Then, it calculates $h(Q_x||h(i_x\_AP_y))$ and checks their position in Ha_AP_y, respectively.

Once the genuineness of the trust values of senders is confirmed, the receiver undergoes signature verification. With the system public parameters, the receiver can verify the signature of the sender by checking whether the following equations are satisfied

$$e(sign_x(m), P) = e(U1_x + H_3(m) + H_2(U1_x||U2_x), Q_x)$$

In the case of multiple APs, the verification is performed as below:

$$e(sign_x(m), P) = e\left(\left|\overrightarrow{U1_x\_AP}\right| + H_3(m) + H_2(\left|\overrightarrow{U1_x\_AP}\right| \parallel \left|\overrightarrow{U2_x\_AP}\right|, Q_x\right)$$

In particular, aggregate signature verification is supported in our scheme. By using the *AggregateSignature* algorithm, K distinct messages from K distinct nodes can be aggregated into one signature $\sigma \leftarrow \prod_{i=1}^{K} sign_{N_i}(m_i)$ that can be collectively verified as below, refer to Section 5.1 for details.

$$\mathrm{e}(\sigma, P) = \prod_{i=1}^{K} \left(e(U1_i + H_3(m_i) + H_2(U1_i||U2_i), Q_i)\right)$$

With the aggregate signature verification, the performance of trust authentication and signature verification can be greatly improved.

4.4 Anonymous Authentication Protocols

There are three main PSN scenarios: 1) only one AP performed by TS; 2) only one AP performed by a PSN node while TS is offline; 3) multiple APs performed by some nodes while TS is offline or by both nodes and TS. The protocols of anonymous authentication on trust in the above three scenarios are illustrated in Fig. 2, Fig. 3, and Fig. 4, respectively. The main processes of each protocol are identical. What different is the concrete algorithms called in the same steps. In principle, the protocols take the aforementioned algorithms as the basis, in which TS first sets up the system and generates essential parameters for each registered node. Then AP, performed by TS or a PSN node, evaluates trust and issues the INT to other nodes to allow them to generate the one-off anonymous key pairs for signature generation and verification. Specifically, a message receiver can authenticate sender trust by referring to the aggregate list and apply aggregate signature verification to improve efficiency. It is evident that AP only issues trust values and the aggregate list. It is not directly involved in the process of signing and verifying messages. The concrete procedures are described below.

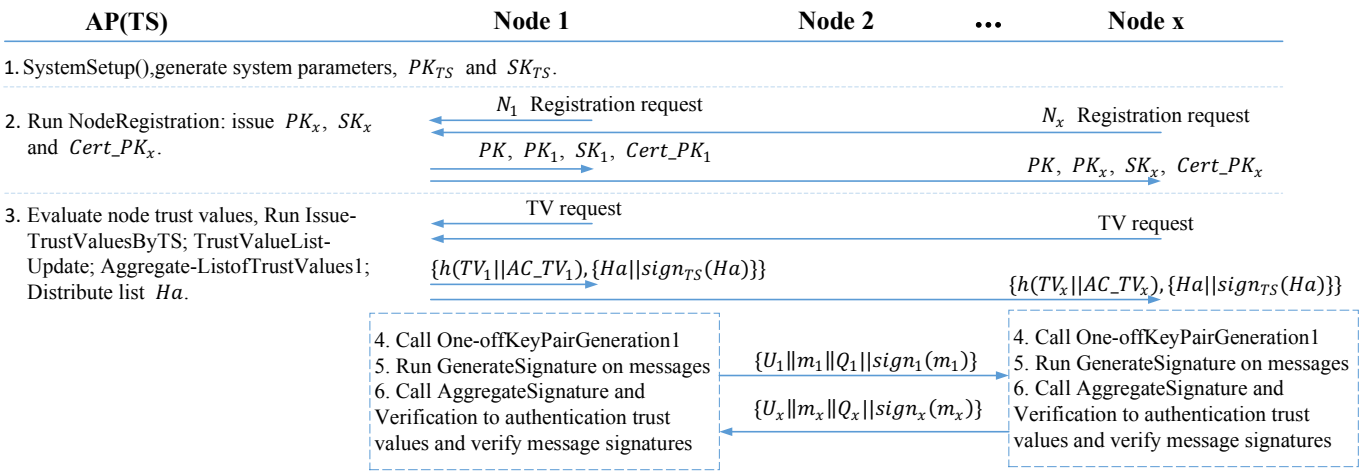

Fig.2 The procedure of anonymous authentication on trust with one AP performed by TS

Step 1: At system initiation, TS calls *SystemSetup* to generate system parameter and the public/private key pair of TS: PK_{TS} and SK_{TS}.

Step 2: Each PSN node N_x needs first to register into the system (TS) with its unique real identity by calling *NodeRegistration*. TS generates a long-term node public/private key pair PK_x and SK_x and issues PK, $Cert\_PK_x$, PK_x and SK_x to the node in a secure way.

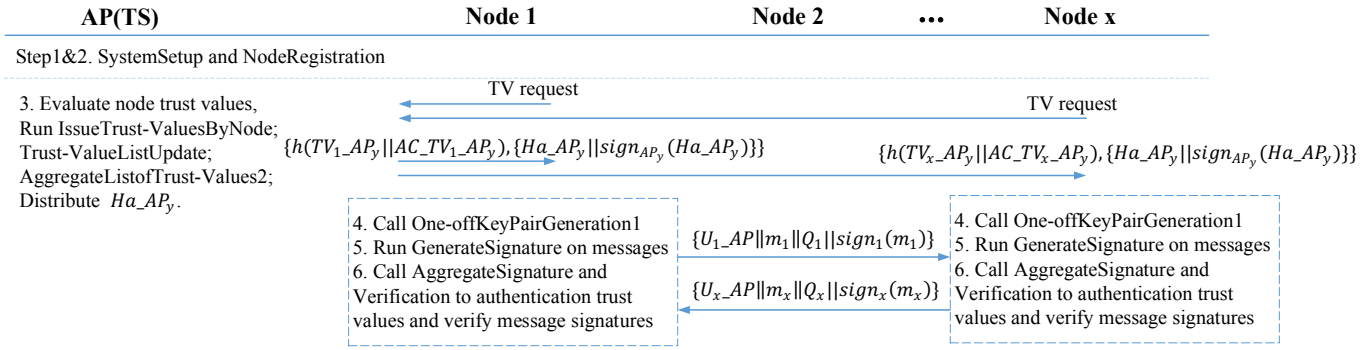

Fig.3 The procedure of anonymous authentication on trust with one AP performed by a PSN node

Step 3: After joining PSN, node N_x could request each AP (performed by TS or PSN node) in the system to issue an initial trust value with a certain validity period. The AP can also check the validity period of a previously trust value to decide whether it should be re-issued. In the first scenario (shown in Fig. 2), where the AP is only performed by TS, the algorithm *IssueTrustValuebyTS* is applied based on a real identity to issue an initial or a new INT value. While in the second scenario (shown in Fig. 3), where the AP is only performed by a PSN node, the AP calls *IssueTrustValuebyNode* based on the one-off public key of other nodes to issue a new INT value. For the above two cases, AP sets the validity period of the new INT value and updates the aggregate list. Then it calls *AggregatedListofTrustValues1* (performed by TS) or *AggregatedListofTrustValues2* (performed by PSN nodes) to generate an up-to-date aggregate list and announces it to all nodes.

Step 4: After getting the new INT values, the PSN node runs *One-offKeyPairGeneration1* (if there is one AP) or *One-offKeyPairGeneration2* (if there are multiple APs) to generate a one-off anonymous public/private key pair.

Step 5: With the one-off private key, the node can sign its messages by calling *GenerateSignature*.

Step 6: The receiver can anonymously authenticate trust values of the sender through signature verification. Note that if a node receives many messages from distinct nodes over a period, it would apply aggregate signature verification to collectively verify them by calling *AggregateSignature* before running *Verification*.

Note that if there are multiple APs in the system (Fig. 4), the procedure after Step 4 can be implemented in another way as described below. After getting the new INT values from different APs, the PSN node can run *One-offKeyPairGeneration1* to generate a one-off public/private key pair. In this way, the node generates the same number of one-off key pairs as the number of APs. With each one-off private key, the node can sign one message by calling *GenerateSignature* and then aggregate these signatures into a single one by calling *AggregateSignature*. Applying aggregate signature verification, the receiver can effectively verify the message signatures. In addition, after getting to know the existence and positions of INT hashes in all aggregate lists, the receiver can analyze the trust values of the sender.

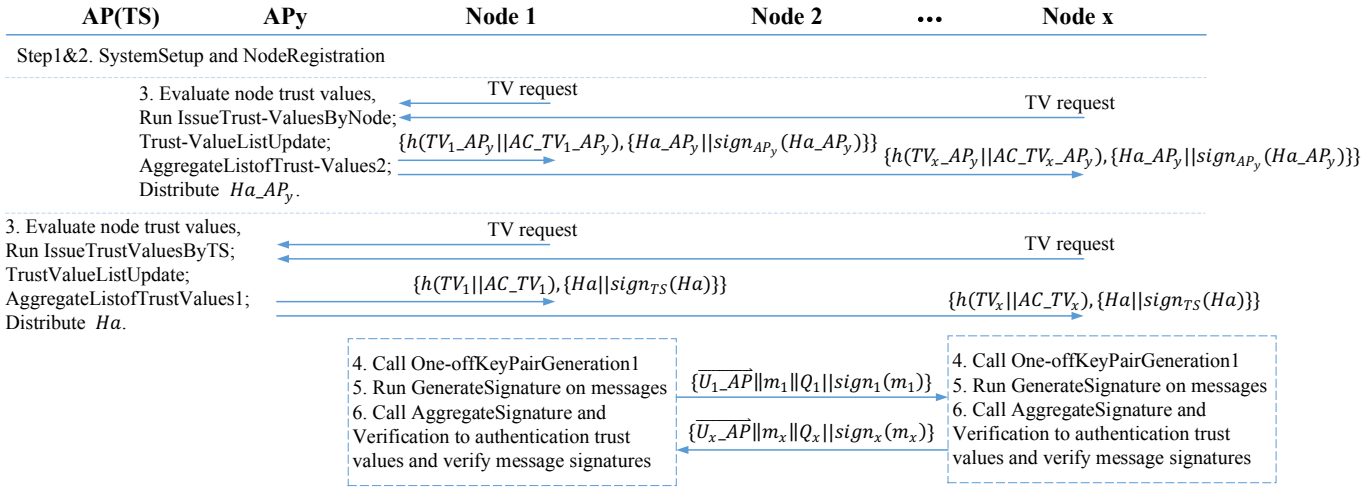

Fig.4 The procedure of anonymous authentication on Trust with multiple APs

4.5 Trust Value Verification and Calculation

Obviously, accurate trust evaluation can be performed with the support of TS since it holds the real identities of nodes. APs performed by nodes can only conduct trust evaluation based on one-off public keys of other nodes. The initial trust value, evolved based on node social behaviors and activities, is issued by TS at node registration. TS is able to figure out the INT of a node and tracks its real identity through collaborating with the APs without disclosing the real identity of the node to any other PSN nodes. Concrete examples of trust evaluation in PSN can be found in [38, 39, 41].

The trust value of a node can be analyzed by verifying the existence and position of $h(Q_x||h(i_x))$ or $h(Q_x||h(i_x\_AP_y))$ in the aggregate list (Ha or Ha_AP_y). If one AP only holds a right to issue a certain level of trust, the existence of $h(Q_x||h(i_x))$ or $h(Q_x||h(i_x\_AP_y))$ in this AP's aggregate list implies the trust level of the node. Since the INT values are arranged in a sorted list (e.g., in an ascending order), it is easy for a node to compare the trust values of the nodes during message authentication. Suppose there are Y aggregate lists in the system, node N_x's INT value is positioned at P_x_y in list y. If there is no trust value issued for N_x in list y, $P_x\_y = 0$. Thus, trust value TL_x of node N_x can be calculated as:

$$TL_x = \sum_{y=1}^{Y} P_x\_y / P_y, \tag{4.5.1}$$

where Y is the total number of AP, P_y is the total length of list y. Herein, we classify the real trust values announced by AP into a limited number of trust levels. This design aims to achieve unlinkability, refer to Section 5.1.

5 SECURITY ANALYSIS & PERFORMANCE EVALUATION

In this section, we analyze and prove our scheme to show whether it achieves our design goals. First, its security is proved theoretically in order to show its correctness, unforgeability, anonymity, unlinkability and conditional traceability. Second, we analyze its performance to exhibit its merits with regard to computational complexity, communication cost, scalability and flexibility. Finally, we implement the proposed scheme to further reveal its real operation performance and compare with other schemes based on simulations.

5.1 Security Proof

Theorem 1. *The proposed scheme of anonymous authentication on trust (AAT) is correct.*

Proof 1. It is essential to prove that the designed scheme works correctly in all possible PSN scenarios. We first prove that the proposed scheme is correct in the case that there is only one AP performed by TS or Node. Node N_x calls the algorithm *One-offKeyPairGeneration1* to generate U_x and V_x based on i_x. When N_x wants to send message m, it computes $sign_x(m) = V1_x + H_3(m) \cdot s_x + V2_x$ and sends $\{U_x||m||Q_x||sign_x(m)\}$ to other nodes. The receiver node first computes $h(i_x)$ from U_x as below:$h(i_x) = \{h(i_x) \oplus H_1(i_x \cdot a \cdot Q_x)\} \oplus H_1(i_x \cdot a \cdot Q_x) = U2_x \oplus H_1(U1_x)$. Then, it computes $h(Q_x||h(i_x))$ and checks its existence in list $Ha = \{h(Q_1||h(i_1)), h(Q_2||h(i_2)), \cdots, h(Q_x||h(i_x)), \cdots\}$.

If the position of $h(Q_x||h(i_x))$ in list Ha is ahead of a certain threshold set by the receiver, the receiver treats N_x as trustworthy and then verifies the signature of message m. Concretely, it checks whether $e(sign_x(m), P)$ is equal to $e(U1_x + H_3(m) + H_2(U1_x||U2_x), Q_x)$, which is proved as below:

$e(sign_x(m), P) = e(V1_x + H_3(m) \cdot s_x + V2_x, P) = e(U1_x \cdot s_x + H_3(m) \cdot s_x + H_2(U1_x||U2_x) \cdot s_x, P)$
$= e(U1_x + H_3(m) + H_2(U1_x||U2_x), Q_x)$

Note that Q_x is shared by N_x with all other nodes. But it is computationally hard to know s_x even though P and Q_x are available. This based on the theory of Elliptic Curve Discrete Logarithm Problem (ECDLP) [5].

Proof 2: We then prove that the scheme is correct in the case that there are multiple APs. Herein, node N_x calls the algorithm *One-offKeyPairGeneration2* to generate one-off anonymous public/private keys ($\overrightarrow{U_x\_AP}$ and V_x_AP) based on vector $\{i_x\_AP_1, \ldots i_x\_AP_Y\}$. When N_x sends message m, it computes $sign_x(m) = V1_x\_AP + H_3(m) \cdot s_x + V2_x\_AP$ and sends $\{\overrightarrow{U_x\_AP}||m||Q_x||sign_x(m)\}$ to other nodes. The receiver first computes $\overrightarrow{h(i_x\_AP)}$ from $\overrightarrow{U_x\_AP}$ as below:

$$\overrightarrow{h(i_x\_AP)} = (h(i_x\_AP_1), h(i_x\_AP_2), \cdots, h(i_x\_AP_Y))$$
$$= (U2_x\_AP_1, \cdots, U2_x\_AP_Y) \oplus (H_1(U1_x\_AP_1) \cdots, H_1(U1_x\_AP_Y))$$
$$= (U2_x\_AP_1 \oplus H_1(U1_x\_AP_1), \cdots, U2_x\_AP_Y \oplus H_1(U1_x\_AP_Y)).$$

Then the receiver can calculate $h(Q_x||h(i_x\_AP_y))$ and obtain its position P_x_y in list Ha_AP_y. With a number of Y lists in the system, the receiver can gain the aggregated trust value TL_x by applying Formula (4.5.1). If TL_x exceeds a certain threshold, node N_x is trusted by the receiver. Then the message signature can be further verified as Proof. 1.

Proof 3: Finally, we prove that the scheme is correct when we apply the aggregated signature to improve the efficiency of the scheme. One situation is that a node generates the same number of one-off key pairs as the number of APs and signs its message with each one-off private key by calling *GenerateSignature*. Another situation is that a node receives distinct messages that signed by one node with distinct private keys. For both of the above situations, *AggerateSignature* is called to combine all signatures into a single aggregated signature $\sigma \leftarrow \sum_{i=1}^{K} sign_i(m_i)$. After computing $veri_i(m_i) = U1_i + H_1(m_i) + H_1(U1_i||U2_i)$ for all messages, the node can verify all signatures by checking $e(\sigma, P)$ is equal to $\prod_{i=1}^{K} e(veri_i(m_i), Q_i)$. With the properties of the bilinear map, we demonstrate the above equation can be achieved as below:

$$e(\sigma, P) = e(\sum_{i=1}^{K} sign_i(m_i), P) = \prod_{i=1}^{K} e(veri_i(m_i), Q_i).$$

Theorem 2. (**Unforgeability**) *Let prime order group* $\mathbb{G}$ *be a* (τ, t', ϵ')*-CDH group that means no algorithm* $\mathcal{B}(t', \epsilon')$ *can break CDH on it. Then the proposed scheme Anonymous Authentication Trust (AAT) scheme is* $\left(t, \epsilon, q_U, q_{H_2}, q_{H_3}, q_S\right)$*-secure against existential forgery on adaptive chosen message attack, where* $t = t' - c_{\mathcal{B}}\left(q_U + q_{H_2} + q_{H_3} + q_S\right)$ *and* $\epsilon = eq_S\epsilon'$*, and* $c_{\mathcal{B}}$ *is a constant and* e *is the base of the natural logarithm.*

Game. Adversary $\mathcal{A}$ has advantage ϵ and time t against our AAT scheme. Suppose $\mathcal{A}$ makes q_U queries to U-queries, q_{H_2} queries to H_2-queries, q_{H_3} queries to H_3-queries, and q_S queries to S-queries. Then there is an algorithm $\mathcal{B}$ that can solve the CDH problem with advantage at least ϵ/eq_S and running time is $t + c_{\mathcal{B}}\left(q_U + q_{H_2} + q_{H_3} + q_S\right)$.

Proof. Algorithm $\mathcal{B}$ is given the parameter $\langle q, \mathbb{G}, \mathbb{G}_T, e\rangle$ and a random instance $\langle P, aP, bP\rangle$ of the CDH problem, i.e. P is a random generator of $\mathbb{G}$ and a, b are random in Z_q^* where q is the order of $\mathbb{G}$ and $\mathbb{G}_T$. Let $D = abP \in \mathbb{G}$ be the solution to the CDH problem. Algorithm $\mathcal{B}$ finds it by interacting with $\mathcal{A}$ as follows:

Setup: Algorithm $\mathcal{B}$ creates the public key $K_{pub} = \langle q, \mathbb{G}, \mathbb{G}_T, P, H_2, H_3\rangle$ and gives it to $\mathcal{A}$. Here H_2 and H_3 are random oracles controlled by $\mathcal{B}$ as described below.

***U*-queries:** Algorithm $\mathcal{A}$ can query public key U at any time. To respond to these queries, algorithm $\mathcal{B}$ maintains a list of tuple $\langle U1_j, U2_j, x_j\rangle$ referred as U^{list}. Initially, it is empty. $\mathcal{B}$ picks random $x_i \in Z_q^*$, $U2_i \in \{0,1\}^n$ and sets $U1_i = x_iP \cdot aP$. Then it adds tuple $\langle U1_i, U2_i, x_i\rangle$ into U^{list} and responds to $\mathcal{A}$ with $\langle U1_i, U2_i\rangle$ in query i.

***H_2*-queries:** To respond to these queries about random oracle H_2, algorithm $\mathcal{B}$ maintains list H_2^{list} in tuple $\langle U1_j, U2_j, y_j, H_{2j}\rangle$. $\mathcal{B}$ picks random $y_i \in Z_q^*$ and sets $H_{2i} = h(U1_i \parallel U2_i) = y_iP$. Then it adds tuple $\langle U1_i, U2_i, y_i, H_{2i}\rangle$ into U^{list} and responds to $\mathcal{A}$ with H_{2i} in query i.

***H_3*-queries:** $\mathcal{A}$ can issue queries to random oracle H_3. To respond to these queries, algorithm $\mathcal{B}$ maintains a list of tuple $\langle r_j, m_j, H_{3j}\rangle$, referred as H_3^{list}. Initially, it is empty. To respond to query m_i, algorithm $\mathcal{B}$ does the following:

1. If query m_i already appears in H^{list} in the tuple $\langle r_i, m_i, H_{3i}\rangle$, then $\mathcal{B}$ responds with $H_3(m_i) = H_{3i}$.
2. Otherwise, $\mathcal{B}$ just generates random bit $b_i \in \{0,1\}$, that $\Pr[b_i = 1] = \xi$ for some ξ that will be determined later.
3. $\mathcal{B}$ picks random number $r_i \in Z_q^*$. If $b_i = 0$, it then sets $H_{3i}(m_i) = H_{3i} = r_iP$. If $b_i = 1$, it then sets $H_3(m_i) = H_{3i} = bP \cdot r_iP$. Afterwards, $\mathcal{B}$ adds the tuple $\langle r_i, m_i, H_{3i}\rangle$ to H_3^{list} and responds to $\mathcal{A}$ with $H_3(m_i) = H_{3i}$. Note that H_{3i} is uniform in $\mathbb{G}$ and is independent of $\mathcal{A}$'s current view as required.

***S*-queries:** $\mathcal{B}$ simulates the signature oracle by responding to the signature query of any message m_j by maintaining a list of tuple $\langle m_j, H_{3j}, \sigma_j\rangle$. We refer to this list as S^{list} that is initially empty. When $\mathcal{A}$ queries oracle S with message m_i, $\mathcal{B}$ responds to query as follows:

1. If query m_i already appears in S^{list} in $\langle m_i, H_{3i}, \sigma_i\rangle$, then algorithm $\mathcal{B}$ responds with σ_i.
2. Otherwise, $\mathcal{B}$ checks whether $\langle U1_i, U2_i, x_i\rangle$, $\langle U1_i, U2_i, y_i, H_{2i}\rangle$ and $\langle r_i, m_i, H_{3i}\rangle$ exist. If not, $\mathcal{B}$ firstly conducts U-queries to get $\langle U1_i, U2_i, x_i\rangle$, H_2-queries to get $\langle U1_i, U2_i, y_i, H_{2i}\rangle$ and H_3-queries to get $\langle r_i, m_i, H_{3i}\rangle$. If $b_i = 0$, $\sigma_i = ar_iP + aU1_i + aH_{2i}$. If $b_i = 1$, it then sets $\sigma_i = *$, a placeholder value. Then it adds tuple $\langle m_i, H_{3i}, \sigma_i\rangle$ to list S^{list} and responds with σ_i.

Challenge: Let m_i be a signature query issued by algorithm $\mathcal{A}$. Algorithm $\mathcal{B}$ runs the above algorithms for responding to S-queries to obtain $\sigma_i \in \mathbb{G}$. Note that $\mathcal{B}$ can run $\mathcal{A}$ with public key K_{pub} to get $\langle P, aP, H_{3i}, \sigma_i\rangle$, which can be converted into a valid Diffie-Hellman tuple.

Claim: $\mathcal{A}$ halts, either admitting failure or obtaining a forged signature $\langle m', \sigma'\rangle$, where $m' = m_{i^*}$, for some i^* on which $\mathcal{A}$ had not queried a signature. If $\mathcal{A}$ succeeds in forging, $\mathcal{B}$ output "**success**" to solve CDH problem; otherwise, it outputs "**failure**". And in the game model, $\mathcal{A}$ behaves exactly as it would like. Thus,

$$\mathrm{Adv}_{\mathcal{B}} = \Pr\left[\mathcal{B}^{\mathcal{A}}(P, aP, bP) = \text{success}: a, b \in Z_q^*\right]$$

$$= \Pr\left[\text{Verify}(U, m', \sigma') = \text{valid}: \begin{matrix}(U,V) \leftarrow OneoffKeyGen \\ (m', \sigma') \leftarrow \mathcal{A}\ (U)\end{matrix}\right] = \epsilon$$

With a modification, if $\mathcal{A}$ fails to create a forgery, $\mathcal{B}$ also fails. But if $\mathcal{A}$ succeeds in finding a forgery on m_{i^*}, $\mathcal{B}$ claims success only if $b_{i^*} = 1$, and $\mathcal{A}$ makes q_S signature oracle queries on the message for which $b_i = 0$ (for which $b_i = 1$, $\mathcal{A}$ declares failure and halts immediately) with indices $i_1, i_2, \cdots, i_{q_S}$, then $\text{Adv}'_{\mathcal{B}} = \text{Adv}_{\mathcal{B}} \cdot \Pr[b_{i^*} = 1] \cdot \Pr\left[b_{i_j} = 0, j = 1,2, \cdots, q_S\right] = \xi(1-\xi)^{qs}\epsilon$.

So, if $\mathcal{B}$ claims success and in addition, output $(\sigma' - a \cdot x_{i^*}P \cdot aP - a \cdot y_{i^*}P)/(r_{i^*}P)$, where i^* is the index of message m' for which $\mathcal{A}$ outputs forged signature σ'. If $b_{i^*} = 1$, it means that $H_{3i^*} = bP \cdot r_{i^*}P$. If σ' is a valid signature on $m' = m_{i^*}$, then $\langle P, aP, H_{i^*}, \sigma' \rangle$ must be a valid Diffie-Hellman tuple, so σ' must equal to $a(H_{3i^*} + U1_{i^*} + H_{2i^*}) = abP \cdot r_{i^*}P + a \cdot x_{i^*}P \cdot aP + a \cdot y_{i^*}P$. Thus, in every instance on which $\mathcal{B}$ claims to succeed, it also outputs $(\sigma' - aU1_{i^*} - aH_{2i})/(r_{i^*}P) = abP$, which is indeed the answer to the CDH problem in $\mathbb{G}$.

The algorithm $\mathcal{B}$ thus uses the signature forger $\mathcal{A}$ to solve CDH problem with advantage ϵ' and time t'. The function $\xi(1-\xi)^{qs}\epsilon$ is maximized at $\xi = 1/(1+q_S)$, where it has the value：

$$\frac{1}{1+q_S}\left(1 - \frac{1}{1+q_S}\right)^{qs} \cdot \epsilon = \frac{1}{q_S}\left(1 - \frac{1}{1+q_S}\right)^{qs+1} \cdot \epsilon.$$

For large q_S, $(1 - 1/(1+q_S))^{qs+1} \approx 1/e$. Meanwhile, $\mathcal{B}$'s running time includes the running time of $\mathcal{A}$. The additional overhead imposed by $\mathcal{A}$ is dominated by the need to evaluate group multiplication for each signature and hash request from $\mathcal{B}$. Anyone such multiplication may be computed by using at most $c_{\mathcal{B}}$ time units on $\mathbb{G}$ (see [12]). $\mathcal{B}$ may need to answer as many as $q_U + q_{H_2} + q_{H_3} + q_S$ such requests. So, its overall running time is $t + c_{\mathcal{B}}(q_U + q_{H_2} + q_{H_3} + q_S)$.

To summarize the above proof, if there exists a forger algorithm $\mathcal{A}$ that $(t, \epsilon, q_U, q_{H_2}, q_{H_3}, q_S)$ breaks our AAT signature scheme on $\mathbb{G}$, then there exists algorithm $\mathcal{B}(t', \epsilon')$ that can break CDH, where $t' = t + c_{\mathcal{B}}(q_U + q_{H_2} + q_{H_3} + q_S)$ and $\epsilon' = \epsilon/(eq_S)$.

Conversely, if group $\mathbb{G}$ be a (τ, t', ϵ') -CDH group, then there exists no algorithm that $(t, \epsilon, q_U, q_{H_2}, q_{H_3}, q_S)$ breaks the AAT signature scheme, where $t = t' - c_{\mathcal{B}}(q_U + q_{H_2} + q_{H_3} + q_S)$ and $\epsilon = eq_S\epsilon'$.

Theorem 3. (Anonymity) *The proposed scheme achieves privacy preservation and anonymous authentication in PSN.*

Proof. The real identity of node N_x is preserved within the TS. The one-off public key U_x used in message authentication is generated from INT value i_x given by AP (nodes or TS), which has no trace of the real identity. In addition, N_x changes the random nonce each time when it generates (U_x, V_x), so the one-off key pair can be unique for each message. Thus, the node privacy can be safely preserved because the TS is fully trusted. Additionally, a node cannot gain any information about the real identity from message package $\{U_x||m||Q_x||sign_x(m)\}$.

In terms of anonymous authentication on trust, the TS periodically distributes Ha signed by its private key SK_{TS} to the PSN nodes. The position of $h(Q_x||h(i_x))$ inside Ha indicates the trust value of N_x but does not leak the real identity of N_x and the real trust value. A node can rely on U_x and its linked trust value to conduct trust authentication and further signature verification. Thus, the proposed scheme provides anonymous authentication on trust with identity privacy protection.

Theorem 4. (Unlinkability) *The proposed anonymous authentication scheme satisfies unlinkability.*

Proof. Unlinkability [6] means that an adversary cannot distinguish nodes based on their communications. This means that all messages generated by a node should not leak any information to an adversary. In the proposed scheme, the U_x of each node is generated absolutely based on the INT value issued by AP. The INT value in the aggregate list (Ha or Ha_AP_y) is divided into different discrete levels that are mapped to the corresponding position in the list. It is possible that the trust value of a number of nodes will fall into the

same trust levels. By classifying the real trust values into a limited number of trust levels, we can make a range of INT value $i_x = h(TV_x||AC\_TV_x)$ for a group of nodes with the same trust level. Thereby, a message receiver cannot judge that two or more messages are sent from the same node during trust authentication even though it verifies that same i_x exists in Ha or Ha_AP_y. In addition, public key U_x is computed at node N_x, where "a" is a random element in the integer field $\mathbb{Z}$ that could be changed by the node for every different message. This guarantees a unique public key is generated each time in terms of a PSN activity. Moreover, the trust value of a node cannot be retrieved from its hash value because of the irreversible property of one-way hash function [2]. Therefore, a receiver cannot link any two public keys and signatures signed by their corresponding private keys together.

Theorem 5. (Conditional Traceability) *The proposed scheme AAT achieves conditional traceability.*

Proof. An identity disclosure is performed only when solving a dispute. In this case, with the accused message $\{U_x||m||Q_x||sign_x(m)\}$ from U_x, TS can extract the long-term public key of the responsible node and its real identity to sanction any penalties based on legal considerations. Therefore, the scheme preserves conditional traceability, which is one of the acceptable and desired properties in PSN. But if TS is not involved in the PSN, such a dispute cannot be solved. Thus, we suggest that for crucial PSN communications, TS should be involved in order to guarantee system safety and at the same time preserve node privacy.

5.2 Performance Analysis

In this section, we analyze the performance of our proposed scheme in terms of computational complexity, communication cost, scalability and flexibility.

Computational Complexity

We analyze the computational complexity of our scheme by only considering time-consuming operations in the algorithms. Table 2 summarizes the computational complexity of the algorithms of the proposed scheme. We assume that the sender generates a new one-off key pair for each message. Thus, both the number of messages transmitted and the number of APs in the PSN system affect the performance of our scheme. But message number impact is inevitable in almost all schemes. We can see that the advantage of our scheme is its performance is not directly impacted by the number of nodes in PSN, especially for anonymous authentication on trust.

TABLE 2
COMPUTATION COMPLEXITY

Algorithms	Computational Complexity
SystemSetup	$O(1)$
NodeRegistration	$O(1)$
IssueTrustValueByTS / IssueTrustValueByNode	$O(1)$
AggregateListofTrustValues1&2	$O(1)$
One-offKeyPairGeneration1	$O(M)$
One-offKeyPairGeneration2	$O(M \times Y)$
GenerateSignature	$O(M)$
AggregateSignature	$O(M)$
Verification	$O(M)$
AggregateVerification	$O(M)$

N: The number of nodes; M: the number of messages; Y: the number of APs

Communication Cost

In our scheme, the communication cost mainly consists of three parts after system setup and node registration: the distribution of the aggregate list, INT value issuing and message exchange, as summarized in Table 3. Ha or Ha_AP_y contains N hash values and $sign_{TS}(Ha)$ is an element in $\mathbb{G}$ with 128 bytes.

Since the size of the hash value is small (only 20 bytes for SHA-1), the total length of the aggregate list is 20N+128 bytes.

The package of INT value $\{h(TV_x||AC\_TV_x), T\_TV_x, s_x, Q_x\}$ issued by TS is composed of four parts: a hash code (SHA-1), a digit, an element in $\mathbb{Z}_p$ and an element in $\mathbb{G}$. Their length are respectively 20 bytes, 4 bytes, 20 bytes, and 128 bytes. Thus, its total size is 172 bytes. The package $\{h(TV_x\_AP_y, AC\_TV_x\_AP_y), T\_TV_x\_AP_y\}$ issued by node APs has a total length of 24 bytes.

TABLE 3
COMMUNICATION OVERHEAD

Communication Items	**Size (byte)**
The aggregate list (e.g., $\{Ha||sign_{TS}(Ha)\}$)	20N+128
$\{h(TV_x||AC\_TV_x), T\_TV_x, s_x, Q_x\}$	172
$\{h(TV_x\_AP_y, AC\_TV_x\_AP_y), T\_TV_x\_AP_y\}$	24
$\{U_x||m||Q_x||sign_x(m)\}$	404 without m
$\{\overrightarrow{U_x\_AP}||m||Q_x||sign_x(m)\}$	148Y+256 without m

Node N_x sends message m to other nodes with package $\{U_x||m||Q_x||sign_x(m)\}$ or $\{\overrightarrow{U_x\_AP}||m||Q_x||sign_x(m)\}$. U_x consists of $U1_x$ and $U2_x$, where $U1_x$ is an element in $\mathbb{G}$, and $U2_x$ is an array of 20 bytes. Q_x and $sign_x(m)$ both are elements in $\mathbb{G}$. And Q_x is not always attached to the message if it is shared in the past. Message m of node N_x cannot be avoided. We can compress the element in $\mathbb{G}$ into a buffer of 128 bytes data. Thus, the total package length is 404 bytes excluding m. If there are a number of Y APs in the system, the length of $\{\overrightarrow{U_x\_AP}||m||Q_x||sign_x(m)\}$ (excluding message m) is 148 Y+256 bytes because each $U_x\_AP_y$ in $\overrightarrow{U_x\_AP} = (U_x\_AP_1, \dots, U_x\_AP_Y,)$ contains $U1_x\_AP_y$ (128 bytes) and $U2_x\_AP_y$ (20 bytes). From Table 3, we can see that the communication cost of message exchange is light and acceptable.

Scalability and Flexibility

First, a public key certificate is not required in the proposed scheme since the public keys can be authenticated from the aggregate list distributed by APs to achieve certificateless authentication on trust. The proposed scheme only requires each AP signs Ha or Ha_AP_y, which is verified only once for all PSN message exchanges. Therefore, verification overhead can be dramatically reduced by excluding certificate verification on every PSN message. Obviously, this design greatly benefits system scalability since the number of nodes does not influence the performance of anonymous authentication on trust. The computational complexity of a node is only related to the number of messages. The extension of network scale has no influence on the computation cost of a single node.

Second, the scheme supports signature aggregation by combining any number of signatures signed by different private keys. Owing to aggregate signature verification, a number of message signatures can be verified together with a low computation cost. This is especially applicable for mobile devices with limited resources.

Third, the scheme can flexibly support anonymous authentication on trust in PSN in a centralized or distributed way. Multiple APs that are either fixed or mobile devices can be deployed in various ways in practice and can operate either independently or cooperatively, in either a centralized or distributed manner. Although the trust value is not directly indicated by APs, a receiver can still figure out the trust level of a sender node. In addition, APs can issue an aggregate list that only contains the INT of the nodes whose trust levels are above some threshold. In this way, it is convenient for the node to decide whether a node is trustworthy. If the node can authenticate the trust levels from more than one AP, it can have more confidence in the trust of the authenticated node. Optionally, we can tailor the scheme by setting APs with different rights to issue different levels of trust.

Based on the above discussion, we can see that our scheme can provide improved scalability due to certificateless verification. It is also flexible to support various deployment strategies with regard to trust policies and PSN topology, thus appropriate to the specific characteristics of PSN.

5.3 Performance Evaluation

We performed simulations to reveal its efficiency by comparing with other schemes. We implemented the proposed scheme in C language using a PBC library. The scheme was implemented on a desktop (running 32-bit Ubuntu Linux 14.04, equipped with Intel(R) Core(TM) i3-3220 CPU 3.30GHz, 4.0G RAM). In our implementation, the map-to-point hash operation spends 6.28 milliseconds and the multiplication operation between $\mathbb{G}$ and $\mathbb{Z}_p$ costs 2.83 milliseconds, the pairing operation consumes 3.29 milliseconds. The above three operations considered in our evaluation are the most resource-consuming operations in the scheme.

Table 4 summarizes the average execution time of 200 running times of each basic algorithm. The computation overheads of *NodeRegistration*, *IssueTrustValueByTS*, and *AggregateListofTrustValues1&2* are low because the above algorithms are conducted at TS with sufficient computation capacities. Besides we notify that the operation time of the *One-offKeyPairGeneration1* (20.93 milliseconds) is much shorter than the RSA key pair generation (about 60 milliseconds). Although signature verification time is longer than RSA, the proposed scheme achieves anonymous authenticate on trust and its efficiency can be improved by applying aggregate signature verification.

TABLE 4
OPERATION TIME OF BASIC ALGORITHMS

Algorithms	Operation Time(millisecond)
SystemSetup	13.68
NodeRegistration	5.76
IssueTrustValueByTS	9.58
AggregateListofTrustValues1&2	9.12
One-offKeyPairGeneration1	20.93
One-offKeyPairGeneration2 (Y=4)	49.50
GenerateSignature	8.87
Verification	19.26
AggregateVerification (M=100)	15.58× *M*

Our scheme is different from the reviewed existing schemes [19, 36, 47] that took advantages of group signature, HMAC or CRL to authenticate nodes, rather than trust values with anonymity. We further implemented two schemes for anonymous trust authentication based on Group Signature (GS) and Attribute-Based Signature (ABS), respectively. We compared the main operation time of our scheme with the two schemes in Table 5. We found that the proposed scheme performs best with regard to System Setup and Signature Generation. Signature Verification also performs a bit better than other two schemes. Aggregate Signature Verification performs better than GS scheme, while ABS scheme cannot support this feature. Considering the drawbacks of GS (e.g., revocation issue) for PSN, our scheme performance exceeds other two schemes.

TABLE 5
OPERATION TIME OF BASIC ALGORITHMS AND COMPARISON WITH OTHER SCHEMES (UNIT: MILLISECOND)

Algorithms	Our Scheme	GS	ABS
System Setup	13.68	18.80	19.87
Node Registration	5.76	7.20	4.23
One-off Key Pair Generation	20.93	1.20	27.23
Signature Generation	8.87	22.30	47.84
Signature Verification	19.26	19.90	26.33

Aggregate Signature Verification	15.58*M*	18.16*M*	N.A.

Fig. 5 shows the operation time of the algorithms, such as *SystemSetup*, *NodeRegistration*, *IssueTrustValuebyTS*, *IssueTrustValuebyNode* and *AggregatedListofTrustValues1&2*. We observe that node registration time is increased linearly with the number of nodes, but it only happens once in the system initialization. The cost of *IssueTrustValuebyTS* and *IssueTrustValuebyNode* increases almost linearly with the number of nodes. *IssueTrustValuebyNode* operates faster than *IssueTrustValuebyTS* because it only contains one hash operation, without time-consuming multiplication operations. The computation cost of *AggregateListofTrustValues1&2* is mainly caused by one map-to-point hash and one multiplication operation at signing. The generation of Ha or Ha_AP_y is very efficient, which is not influenced much by the number of nodes. Note that these algorithms are mainly run by TS or AP (a capable node), the execution performance is reasonable even for a large-scale PSN. The result conforms to our theoretical analysis. The curves of *SystemSetup* and *AggregateListofTurstValue1&2* shown in Fig. 5 are very close because their practical operation time is very similar.

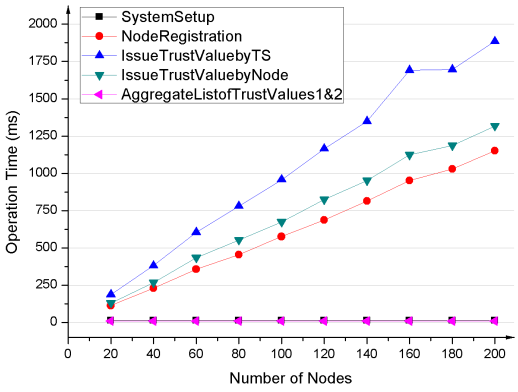

Fig. 5 The operation time of basic algorithms at TS or AP

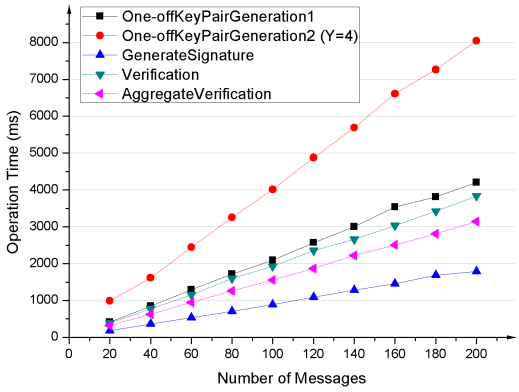

Fig. 6 The operation time of *One-offKeyPairGeneration1*, *One-offKeyPairGeneration2* (Y=4), *GenerateSignature*, *Verification* and *AggregateVerification* at nodes

Fig. 6 shows the operation time of the algorithms run at nodes. The computation cost of *One-offKeyPairGeneration1* increases linearly with the number of messages if the node generates different key pairs for different messages in order to achieve stringent privacy. In practice, the node can use the same key pair for a set of messages (e.g., in a group conversation) in order to gain efficiency. The computation cost of *GenerateSignature* also increases linearly with the number of messages. If there are multiple APs (e.g., Y=4) in the system, *One-offKeyPairGeneration2* will consume more time to compute a key pair. The key generation time is almost linearly increased with the number of messages. The computation cost of *GenerateSignature* in the case of multiple APs is almost the same as that when only one AP exists. This is because the one-off private keys for signing messages in the above two cases are exactly the same in terms of key structure.

Finally, we evaluated the operation time of signature verification that contains two parts: trust authentication and signature verification, also shown in Fig. 6. Based on our analysis, trust authentication only contains XOR operation to compute the hash of INT value and check its existence in the aggregate list. These operations execute very fast compared with the operations performed in signature verification. The time spent on signature verification is linearly increased with the number of messages the node receives. However, applying the aggregate signature verification can save half of pairing operations in signature verification. Comparing the operation time of two signature verification methods, we can see that applying the aggregate signature verification can obviously improve verification efficiency. Although the operation time of aggregate signature

verification still increases linearly with the number of messages, its growth is slower than normal verification. This result is especially attractive for mobile devices with limited resources that play as the nodes of PSN.

6 Conclusion

We proposed a novel authentication scheme that supports trustworthy PSN by authenticating node trust and verifying node signatures in an anonymous way. It can be flexibly deployed in a centralized or distributed manner because the AP can be played by either a centralized trusted server or PSN nodes. Meanwhile, the scheme also allows multiple APs to simultaneously issue trust values of all nodes. The proposed scheme achieves anonymous authentication on trust with unforgeability, anonymity, unlinkability and conditional traceability. Applying aggregate signature verification can further improve its efficiency. The performance analysis and evaluation showed that our scheme is effective and efficient with regard to computational complexity, communication cost, scalability and flexibility. For the future work, we will embed the proposed scheme into a PSN prototype system based on smartphones [15], further demonstrate its applicability and investigate its social and user acceptance. In addition, we will explore the practical use of the scheme in Vehicular Ad hoc Networks (VANETs) and Unmanned Aerial Vehicle Networks (UAVNETs).

Acknowledgements

This work is sponsored by the NSFC (grants 61672410 and U1536202), the Academy of Finland (grant 308087), the National Key Research and Development Program of China (grant 2016YFB0800704), the Project Supported by Natural Science Basic Research Plan in Shaanxi Province of China (Program No. 2016ZDJC-06), the Fundamental Research Funds for the Central Universities (JBG161509), and the 111 project (grants B16037 and B08038). The corresponding author is Zheng Yan.

References

[1] D. Boneh, X. Boyen, H. Shacham, Short Group Signatures, in: Advances in Cryptology - CRYPTO 2004, Springer, 2004, pp 41-55.

[2] D. Boneh, M. Franklin, Identity-Based Encryption from the Weil Paring, in: Advances in Cryptology - CRYPTO 2001, Springer, 2001, pp. 213-229.

[3] D. Boneh, C. Gentry, B. Lynn, H. Shacham, Aggregate and Verifiably Encrypted Signatures from Bilinear Maps, in: Advances in Cryptology - EUROCRYPT 2003, Springer, 2003, pp. 416-432.

[4] D. Boneh, B. Lynn, H. Shacham, Short Signature from the Weil Pairing, in: Advances in Cryptology - ASIACRYPT 2001, Springer, 2001, pp 514-532.

[5] J.W. Bos, M.E. Kaihara, T. Kleinjung, A.K. Lenstra1, P.L. Montgomery, PlayStation 3 computing breaks 2^60 barrier 112-bit prime ECDLP solved, 2009, http://lacal.epfl.ch/112bit_prime.

[6] E. Cesena, H. Löhr, G. Ramunno, A. R. Sadeghi, D. Vernizzi, Anonymous authentication with TLS and DAA, in: International Conference on Trust and Trustworthy Computing, Springer, 2010, pp.47-62.

[7] D. Chaum, E. Van Heyst, Group Signatures, in: Workshop on the Theory and Application of of Cryptographic Techniques, Springer, 1991, pp. 257-265.

[8] T. Ciszkowski, Z. Kotulski, Distributed Reputation Management in Collaborative Environment of Anonymous MANETs, in: EUROCON 2007 - The International Conference on "Computer as a Tool", IEEE, 2007, pp.1028-1033.

[9] K. Emura, A. Kanaoka, S. Ohta, K. Omote, T. Takahashi, Secure and Anonymous Communication Technique: Formal Model and Its Prototype Implementation, IEEE Transactions on Emerging Topics in Computing, 4 (2016) 88-101.

[10] W. Feng, Z. Yan, H.M. Xie, "Anonymous authentication on trust in pervasive social networking based on group signature", IEEE Access, 5 (2017) 6236-6246.

[11] E. Fonseca, A. Festag, R. Baldessari, R. Aguiar, Support of anonymity in vanets-putting pseudonymity in practice, in: IEEE Wireless Communications and Networking Conference (WCNC'07), IEEE, 2007, pp.3400-3405.

[12] Q. Guan, F.R. Yu, S. Jiang, V.C Leung. Joint Topology Control and Authentication Design in Mobile Ad hoc Networks with Cooperative Communications, IEEE Transactions on Vehicular Technology, 61 (2012) 2674-2685.

[13] Y. Hao, Y. Cheng, C. Zhou, W. Song., A Distributed Key Management Framework with Cooperative Message Authentication in VANETs, IEEE Journal on Selected Areas in Communications, 29 (2011) 616-629.

[14] D. He, S. Zeadally, N. Kumar, and J.H. Lee, Anonymous authentication for wireless body area networks with provable security, IEEE Systems Journal, 99 (2016) 1-12.

[15] C.Y. Huang, Z. Yan, N. Li, M. J. Wang, Secure Pervasive Social Communications based on Trust in a Distributed Way, IEEE Access, 4 (2017), 9225-9238.
[16] H. Jayasree, A. Damodaram, Anonymity and accountability in web-based transactions, Advanced Computing: an International Journal, 3 (2012) 171-182.
[17] P. Kelley, R. Brewer, Y. Mayer, L. Cranor, N. Sadeh, An Investigation into Facebook Friend Grouping, in: IFIP Conference on Human-Computer Interaction (INTERACT'11), Springer, 2011, pp.216-233.
[18] Y. Lee, S. Han, S. Lee, B. Chung, D. Lee, Anonymous Authentication System Using Group Signature, in: International Conference on Complex, Intelligent and Software Intensive Systems (CISIS'09), IEEE, 2009, pp.1235-1239.
[19] P. Li, J. Li, Z.G. Huang, C.Z. Gao, W.B. Chen, K. Chen, "Privacy-preserving outsourced classification in cloud computing," Cluster Computing, (2017) 1-10.
[20] B. Libert, T. Peters, M. Yung, Group Signatures with Almost-for-Free Revocation, in: Advances in Cryptology - CRYPTO 2012, Springer, 2012, pp. 571-589.
[21] X. Lin, X. Li, Achieving Efficient Cooperative Message Authentication in Vehicular Ad hoc Networks, IEEE Transactions on Vehicular Technology, 62 (2013) 3339-3348.
[22] X. Lin, X. Sun, X. Wang, C. Zhang, P.H. Ho, X. Shen, TSVC: Timed Efficient and Secure Vehicular Communications with Privacy Preserving, IEEE Transactions on Wireless Communications, 7 (2008) 4987-4998.
[23] Y. Lindell, "Anonymous Authentication," Journal of Privacy and Confidentiality, 2 (2007) 35-63
[24] J. K. Liu, C. K. Chu, S. S. M. Chow, X. Huang, M. H. Au, J. Zhou, Time-Bound Anonymous Authentication for Roaming Networks, IEEE Transactions on Information Forensics and Security (TIFS), 10 (2015) 178-189.
[25] J. Liu, Z. Zhang, X. Chen, K. Kyung Sup, Certificateless Remote Anonymous Authentication Schemes for Wireless Body Area Networks, IEEE Transactions on Parallel and Distributed Systems, 25 (2014) 332-342.
[26] S. Lyu, J. Liu, M. Tang, Y. Xu, J. Chen, Efficiently Predicting Trustworthiness of Mobile Services Based on Trust Propagation in Social Networks, Mobile Networks and Applications, 20 (2015) 840-852.
[27] M.S.I. Mamun, A. Miyaji, Secure VANET Applications with a Refined Group Signature, in: 12th Annual International Conference on Privacy, Security and Trust (PST), IEEE, 2014, pp.199-206.
[28] M. Manulis, N. Fleischhacker, F. Günther, F. Kiefer, B. Poettering, Group Signatures: Authentication with Privacy, Bundesamt fur Sicherheit in der Informationstechnik, Bonn, Germany, Tech. Rep, (2012)
[29] A. Miyaji, M. Nakabayashi, S. Takano, New Explicit Conditions of Elliptic Curve Traces for FR-reduction, IEICE Transactions on Fundamentals, 84 (2001) 1234-1243.
[30] A. Pfitzmann, M. Hansen "Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management - A Consolidated Proposal for Terminology." Version v0, 31, p. 15, 2008.
[31] M. Raya, J.P. Hubaux, The security of VANETs, in: Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks, ACM, 2005, pp.11-21.
[32] M. Raya, J.P. Hubaux, Securing Vehicular Ad hoc Networks, Journal of Computer Security, 15 (1) (2007) 39-68.
[33] F. Sato, H. Takahira, T. Mizuno, Message Authentication Scheme for Mobile Ad hoc Networks, in: Proc. 11th International Conference on Parallel and Distributed Systems, IEEE, 2005, pp.50-56.
[34] J. Shao, X. Lin, R. Lu, C. Zuo, A Threshold Anonymous Authentication Protocol for VANETs, IEEE Transactions on Vehicular Technology, 65 (2016) 1711-1720.
[35] A. Wasef, X. Shen, Efficient Group Signature Scheme Supporting Batch Verification for Securing Vehicular Networks, in: Proc. IEEE International Conference on Communication (ICC'10), IEEE, 2010, pp.1-5.
[36] A. Wasef, X. Shen, EMAP: Expedite Message Authentication Protocol for Vehicular Ad hoc Networks, IEEE Transactions on Mobile Computing, 12 (2013) 78-89.
[37] L. Wu, Y. Zhang, L. Li, and J. Shen, Efficient and anonymous authentication scheme for wireless body area networks, Journal of medical systems, 40 (2016) 1-12.
[38] Z. Yan, Y. Chen, Y. Shen, A Practical Reputation System for Pervasive Social Chatting, Journal of Computer and System Sciences, 79 (2013) 556-572.
[39] Z. Yan, Y. Chen, Y. Shen, PerContRep: A Practical Reputation System for Pervasive Content Services, Journal of Supercomputing, 70 (2014) 1051-1074.
[40] Z. Yan, W. Feng, P. Wang, Anonymous Authentication for Trustworthy Pervasive Social Networking, IEEE Transactions on Computational Social Systems, 2 (2015) 88-98.
[41] Z. Yan, V. Niemi, Y. Chen, P. Zhang, R. Kantola, Towards Trustworthy Mobile Social Networking, Mobile Social Networking: An Innovative Approach, A. Chin and D. Zhang, eds., Part of the series Computational Social Sciences, New York: Springer, 2013, pp.195-235.
[42] Z. Yan, M. Wang, Protect Pervasive Social Networking based on Two Dimensional Trust Levels, IEEE Systems Journal, 11(1) (2017) 207-218.
[43] Z. Yan, M. Wang, V. Niemi, R. Kantola, Secure Pervasive Social Networking based on Multi-Dimensional Trust Levels, in: IEEE Conference on Communications and Network Security (CNS'13), IEEE, 2013, pp.100-108.

[44] Z. Yan, M. Wang, P. Zhang, A Scheme to Secure Instant Community Data Access Based on Trust and Contexts, in: IEEE International Conference on Computer and Information Technology (CIT'14), IEEE, 2014, pp.646-651.
[45] C. Zhang, R. Lu, X. Lin, P.H. Ho, X. Shen, An Efficient Identity-Based Batch Verification Scheme for Vehicular Sensor Networks, in: IEEE INFOCOM 2008 - The 27th Conference on Computer Communications, IEEE, 2008, pp.246-250.
[46] L. Zhang, Q. Wu, A. Solanas, J. Domingo-Ferrer, A Scalable Robust Authentication Protocol for Secure Vehicular Communications, IEEE Transactions on Vehicular Technology, 59 (2010) 1606-1617.
[47] X. Zhu, S. Jiang, L. Wang, H. Li, Efficient Privacy-Preserving Authentication for Vehicular Ad hoc Networks, IEEE Transactions on Vehicular Technology, 63 (2014) 907-919.
[48] Information technology Security techniques Evaluation criteria for IT security, ISO/IEC JTC 1/SC 27 IT Security techniques, 2014.
[49] Uber, https://www.ubr.com.
[50] Didi car-sharing in China, http://www.xiaojukeji.com.
[51] eRideShare, http://www.erideshare.com.

BRIEF BIOGRAPHIES OF AUTHORS:

Zheng Yan is currently a professor at the Xidian University, Xi'an, China and a visiting professor at the Aalto University, Espoo, Finland. She authored more than 160 peer-reviewed publications and solely authored two books. She is the inventor and co-inventor of 60+ patents and patent applications. Her research interests are in trust, security and privacy. Prof. Yan serves as an associate editor of Information Sciences, IEEE Internet of Things Journal, Information Fusion, JNCA, IEEE Access, Security and Communication Networks etc. journals, an organization and program committee member for numerous international conferences and workshops. She is a senior member of the IEEE.

Pu Wang received the BSc degree in Telecommunications Engineering from Xidian University, Xi'an, China, 2011. He is currently a Ph.D. student major in information security at the Xidian University, Xi'an, China. His research interests are in information security, trust management in Internet of Things and cloud computing.

Wei Feng received the BSc degree in Telecommunications Engineering from Xidian University, Xi'an, China, 2011. He is currently a Ph.D. student major in information security at the Xidian University, Xi'an, China. His research interests are in information security, privacy preservation and trust management in social networking.