

Privacy Preserving Authentication Schemes and Applications

Pranav Asokan

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science

in

Computer Engineering

Jung-Min (Jerry) Park, Chair

Yaling Yang

Haibo Zeng

May 08, 2017

Blacksburg, Virginia

Keywords: Privacy-preserving authentication, direct anonymous attestation, trusted
platform module

Copyright 2017, Pranav Asokan

Privacy Preserving Authentication Schemes and Applications

Pranav Asokan

(ABSTRACT)

With the advent of smart devices, Internet of things and cloud computing the amount of information collected about an individual is enormous. Using this meta-data, a complete profile about a person could be created - professional information, personal information like his/her choices, preferences, likes/dislikes etc. The concept of privacy is totally lost with this gamut of technology. The ability to separate one's on-line identity from their personal identity is near impossible. The conflicting interests of the two parties - service providers' need for authentication and the users' privacy needs - is the cause for this problem. *Privacy Preserving Authentication* could help solve both these problems by creating valid and anonymous identities for the users. And simply by proving the authenticity and integrity of this anonymous identity (without revealing/exposing it) the users can obtain services whilst protecting their privacy. In this thesis, I review and analyze the various types of PPA schemes leading to the discussion of our new scheme *Lightweight Anonymous Attestation with Efficient Revocation*. Finally, the application scenarios where these schemes are applicable are discussed in detail.

Privacy Preserving Authentication Schemes and Applications

Pranav Asokan

(GENERAL AUDIENCE ABSTRACT)

With the advent of smart devices, people are almost always connected to the Internet. These smart devices and applications collect information about the user on a massive scale. When all such meta-data are put together, a complete profile of the user - professional and personal information, his/her choices, preferences, likes/dislikes etc. could be created. And all this data is stored somewhere on the Internet. The concept of privacy loses its meaning as this entity knows more about the user than they do themselves. The main reason for this is the inability to separate one's on-line identity from their personal identity. Service providers need to authenticate the users - the process by which one entity is assured of the identity of the second entity it is interacting with - to ensure only valid members are allowed to use their service. This leads to invasion of the user's privacy/anonymity as authentication often needs details like address, date-of-birth, credit card details etc. *Privacy Preserving Authentication* could help solve both these problems by creating valid but anonymous identities for the users. PPA works by issuing the users a secret credential if they can prove their identity. And simply by proving the authenticity and integrity of these secret credentials (without revealing/exposing it) the users can obtain services whilst protecting their privacy. In this thesis, I review and analyze the various types of PPA schemes leading to the discussion of our new scheme *Lightweight Anonymous Attestation with Efficient Revocation*. Finally, the application scenarios where these schemes are applicable are discussed in detail.

Acknowledgments

Foremost, I would like to express my sincere gratitude to my advisor Prof. Jung-Min Park for giving me the opportunity to work on this project, for his immense knowledge and enthusiastic personality. And I truly am lucky to have had him as my advisor and mentor for my Masters study.

I would like to thank the rest of my thesis committee: Dr. Yaling Yang and Dr. Haibo Zeng for their support and encouragement.

A special mention for Dr. Vireshwar Kumar for his constant support, cheerful attitude, insightful comments, words of wisdom and, most importantly, for his brotherly friendship. His guidance and ideas played a massive role in the completion of this thesis and project.

Every result in this thesis was accomplished with the help of my fellow collaborators Noah Luther and He Li. I thank them for the stimulating discussions and suggestions. And I thank my lab-mates Pradeep Reddy and Gaurang Naik for the timely help and advise.

Last but not the least, I would like to thank my family: my parents Mr. Asokan and Dr. Sampathkumari and my sister Dr. Niranjana for everything that I am today.

Contents

- 1 Introduction** **1**
- 1.1 Privacy Preserving Authentication 2
- 1.2 Contribution 4
- 1.3 Structure of this thesis 5

- 2 Background** **6**
- 2.1 Bilinear Pairing 6
- 2.2 Zero-knowledge Proof 7
- 2.3 Group Signatures 8
- 2.4 BBS+ Signature 8
- 2.5 Trusted Platform Module 9
- 2.6 Cryptographic Assumptions 10

3	Privacy Preserving Authentication	11
3.1	Verifier Anonymous Authentication	12
3.1.1	Signer based Revocation	13
3.1.2	Verifier Local Revocation	16
3.1.3	Trusted Third Party based Revocation	18
3.2	Fully Anonymous Authentication	19
3.2.1	Direct Anonymous Attestation	20
4	Analysis of PPA schemes	27
4.1	Signer-based Revocation	27
4.2	Verifier-local Revocation	28
4.3	Trusted Third Party-based Revocation	29
4.4	Direct Anonymous Attestation	30
5	Lightweight Anonymous Attestation with Efficient Revocation (LASER)	31
5.1	Overview	32
5.2	Scheme	35
5.3	Implementation and Analysis	38

6 Applications	41
6.1 Multiple Cloud Collaborative Environment	41
6.1.1 Applicability of LASER for MCCE	42
6.2 DAA for Embedded Mobile Devices	47
6.3 Usage Statistics and Error Reports	48
6.4 PPA for VANETs	49
7 Conclusion	50
Bibliography	52

List of Figures

5.1	Comparison of offline computational overhead between B-EPID[1], CDL-EPID[2] and LASER[3]	40
5.2	Comparison of online computational overhead between B-EPID[1], CDL-EPID[2] and LASER[3]	40
6.1	Comparison of absolute and conditional unlinkability in LASER	46

Chapter 1

Introduction

Privacy is the state or condition of being free from being observed says Google. But with the advent of smart devices, cloud computing and Internet of things people are connected to the Internet almost all the time. And all kinds of information about these users are collected and stored somewhere on the Internet. And the devices they carry around all the time, helps in tracking their day to day lives. It has become easier to observe people from afar. In today's world where everything is connected, companies like Google and Facebook themselves keep enormous amount of data and meta-data about its users, privacy is an illusion. Recommendation engines log users' usage pattern, preferences and suggest what they should do next. And the user's choice helps the engine learn more about him/her. If there is a data breach at the service provider end - Yahoo attack, Sony data breach, Ashley Madison accounts hack etc - it leads to cyber crime, identity theft and other forms of privacy invasion. This leads to the case where the online identity of an individual needs to

be separated from his personal identity. But service providers need to authenticate the users and need information about their personal identity like date-of-birth, credit card information, address etc. And the use of security questions as an added layer of authentication isn't appealing as the online presence and meta-data collected about the individual is enough to crack this layer. This conflicting objectives of user's need for privacy and service providers need for authentication is the problem researchers have tried to address in recent years.

1.1 Privacy Preserving Authentication

Privacy preserving authentication schemes resolve the conflicting interests of the service providers/verifiers and the service users/signers by making use of secret credentials and (zero knowledge) proof of knowledge protocols. The user obtains secret credentials from the issuer by proving its identity or integrity of the platform. Later for signing, the user proves that it holds a valid credential and its proof is correct, which is enough for the verifier to authenticate and provide its services. Thus, user privacy preserved and the verifier also provides its services to valid users only. PPA schemes aim at unlinkable transactions, minimal information disclosure and for the service providers/verifiers - a secure and trustworthy method to authenticate the users. This helps users have a separation between their online and offline identity ensuring privacy much better than existing methods.

There are two main types of PPA schemes in literature. *Verifier Anonymous Authentication* schemes provide anonymity to the signer with respect to the verifier only. A trusted

third party or an issuer or revocation manager knows the identity of the signer and can open his signature at all times. *Fully Anonymous Authentication* schemes provide complete anonymity to the signer. Neither the verifier nor any other entity can know the identity of the signer if he opts for complete anonymity. The signatures also would completely unlinkable. The Trusted Computing Group, a security organization that focuses on PPA schemes amongst others, has emphasized on the use *Direct Anonymous Attestation* schemes for fully anonymous authentication. The DAA schemes make use of a *Trusted Platform Module*, a secure crypto-processor in a device that stores cryptographic secret keys for anonymous authentication. The TCG sets standards for these TPMs which are proven to be secure and trustworthy. Thus, PPA schemes are very much in development and could find its way in most applications in the near future.

Some applications where PPA could be applied are,

1. Multiple Cloud Collaborative Environment: for users to anonymously authenticate and obtain services from different cloud service providers.
2. Mobile embedded devices: users can subscribe to applications through their mobile network operator.
3. Application Usage Statistics and Error Reports: to protect the identity of the user when applications request for these reports from the user devices.
4. Vehicular Ad-hoc Networks: communication between vehicles and infrastructure where the focus is on the message and not the sender.

1.2 Contribution

1. Review the different types of PPA schemes and analyze the pros and cons of each scheme. Though not all schemes are practical, some application scenarios for each type of scheme are also mentioned.
2. As part of team, develop new PPA scheme (LASER) to overcome the shortcoming of existing ones. Implement and compare the new scheme with state-of-the-art schemes illustrating our advantages.
3. Explored and explain in detail the applicability of our scheme for some real-world applications. Find some other example applications for schemes in literature.

Challenges:

Since most of the schemes proposed so far do not have or give the implementation details, figuring out the type of curve, platform and software used to implement them and have a fair basis for comparison was challenging. And with the newer smart devices becoming smaller and more resource constrained, we tried implementing our scheme on a lightweight platform to analyze its efficiency and applicability for IoT devices. After several iterations of modifying the scheme and implementing on a Raspberry Pi 3 platform, we understood the enormous overhead PPA schemes produce for the proof of knowledge protocols was too heavy to handle for the low end device. Especially, the BN256 curve (256-bits to represent the points on the curve) computations could not be done on the Raspberry Pi device. Finally, finding new and

upcoming applications where our scheme would fit in proved to be a formidable proposition. Since the common norm is to use a username-password method for most existing applications, we had to look for an application where PPA could be incorporated right from its initial stages.

1.3 Structure of this thesis

The remainder of this thesis is structured as follows: In chapter 2, the background details needed for the concepts used are explained. Chapter 3 explains about PPA in general and focuses on the first type of PPA schemes. Chapter 4 explains about schemes belonging to the second type of PPA leading to our scheme, LASER. The implementation details, results and analysis are given to conclude the chapter. Chapter 5 discusses the applications for our scheme elaborately and mentions other applications for existing PPA schemes. Chapter 6 concludes the thesis.

Chapter 2

Background

The following sections will explain some of the prominent concepts used in this thesis.

2.1 Bilinear Pairing

A pair of multiplicative cyclic groups of prime order p , \mathbb{G}_1 and \mathbb{G}_2 , is called a bilinear group pair, if there exists a group \mathbb{G}_T , and a bilinear mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with the following properties [4]:

1. Bilinearity: $e(u^\alpha, v^\beta) = e(u, v)^{\alpha \cdot \beta}$, for all $u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$, and $\alpha, \beta \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$. Here, \mathbb{Z}_p^* represents the set of integers modulo p , and $\stackrel{R}{\leftarrow}$ represents a random selection.
2. Non-degeneracy: There exists $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$ such that $e(g_1, g_2) \neq 1$, which means the map does not send all pairs in $\mathbb{G}_1 \times \mathbb{G}_2$ to the identity in G_T ; making g_1 the

generator of G_1 and g_2 the generator of G_2 .

3. Computability: There is an efficient algorithm to compute $e(u, v)$ for all $u \in \mathbb{G}_1$ and $v \in \mathbb{G}_2$.

To implement pairing-based cryptography, I used the PBC library given in [5]. This library defines almost all the functions that can be done with pairing of two same or different groups with many choices for the curve.

2.2 Zero-knowledge Proof

In cryptography, these are interactive methods for a signer to prove to the verifier that a particular statement is true without revealing anything other than the statement's truthfulness. In technical terms, as given in [6], 'a protocol which is a proof of knowledge has the zero-knowledge property if it is simulatable in the scenario - there exists an expected polynomial-time algorithm which can produce, upon input of the assertions to be proven but without interacting with the real prover, transcripts indistinguishable from those resulting from interaction with the real prover.' Zero-knowledge proofs need to have three important properties: completeness, soundness and zero-knowledge.

2.3 Group Signatures

The group signature setting was introduced by Chaum and Van Heyst [7]. Consider a group with numerous members and a group manager. The group has a single signature-verification key gpk - the group public key. Each member of this group have their own secret signing key using which they can produce a signature relative to gpk . The technical details which makes this possible include the group manager having a secret key $gmsk$ based on which it can extract the identity of the signer given a signature σ he created (traceability). Also, any other entity that doesn't know the $gmsk$ should not be able to identify the signer from his signature σ (anonymity).

Since then, many new requirements have been introduced that refine or augment the core ones. Some are unlinkability, unforgeability, collision resistance etc. [8]. Group signatures are needed for privacy preserving attestation [9]. Majority of PPA schemes are a variant of group signatures.

2.4 BBS+ Signature

The BBS+ signature scheme is defined as follows [10, 11, 12],

1. $(pk, sk) \leftarrow \text{BBS_Setup}(1^\lambda)$: The input to this key generation algorithm is the security parameter 1^λ . This algorithm selects $g_1, h_1, h_2, h_3 \leftarrow \mathbb{G}_1$, $g_2 \leftarrow \mathbb{G}_2$, $\gamma \leftarrow \mathbb{Z}_p^*$; and computes

$\omega = g_2^\gamma$. This algorithm sets the public key $\mathbf{pk} = (g_1, h_1, h_2, h_3, g_2, \omega)$, and the secret key $\mathbf{sk} = \gamma$. It outputs $(\mathbf{pk}, \mathbf{sk})$.

2. $\sigma_b \leftarrow \text{BBS_Sign}(\mathbf{pk}, \mathbf{sk}, f, x)$: The inputs to this signature generation algorithm are the public key \mathbf{pk} , the secret key \mathbf{sk} , and two messages $f, x \in \mathbb{Z}_p^*$. This algorithm selects $y, z \leftarrow \mathbb{Z}_p^*$, and computes $A = (g_1 \cdot h_1^f \cdot h_2^x \cdot h_3^y)^{1/(\gamma+z)}$. It sets the signature $\sigma_b = (A, y, z)$, and outputs σ_b .
3. $(\text{valid/invalid}) \leftarrow \text{BBS_Verify}(\mathbf{pk}, \sigma_b, f, x)$: The inputs to this signature verification algorithm are \mathbf{pk} , a purported signature σ_b , and two messages f and x . This algorithm verifies that $e(A, \omega \cdot g_2^z) = e(g_1 \cdot h_1^f \cdot h_2^x \cdot h_3^y, g_2)$. If the verification succeeds, this algorithm outputs *valid*; otherwise, it outputs *invalid*.

2.5 Trusted Platform Module

TPM is a cryptographic co-processor chip (microcontroller) that can securely store artifacts used to authenticate the platform (PC, Laptop, smart phone) [13]. TPMs can store passwords, certificates or secret keys. One of their important applications is storing platform measurements that help to ensure platform remains trustworthy. That is, the platform is not compromised or tampered with. Two ways to ensure safer computing is authentication and attestation. Authentications means ensuring that the platform can prove what it claims to be. And attestation is the process by which a platform can cryptographically prove to another platform that it is in a particular state. TPMs have been used in many applications

from secure military platforms to industrial control systems to electronic voting systems [14].

TPM contains two important functional components - a cryptographic engine that can perform encryption, digital signatures and hashing, and a special register set called the Platform Configuration Registers (PCRs). They store the representation of the state of software on the platform [14].

The latest standards - TPM 2.0 Specification - were introduced by the Trusted Computing Group in 2015. As explained in [15], TPM 2.0 supports cryptographic algorithms such as RSA, ECC, ECC-DAA, ECDH, SHA256, HMAC, AES etc. Using TCG IDs, vendors can also add new algorithms to the TPMs. 2.0 also supports various signature schemes. TPMs store different types of keys of which the most prominent ones are the Endorsement Key (EK), Attestation Identity Key (AIK), Storage Key (SK). This ensures that TPMs are secure and can provably attest and authenticate the device.

2.6 Cryptographic Assumptions

The security of LASER is proved in the random oracle model based on the discrete logarithm (DL) assumption, the decisional Diffie-Hellman (DDH) assumption [16], and the q -strong Diffie-Hellman (q -SDH) assumption [17].

Chapter 3

Privacy Preserving Authentication

Privacy preserving authentication schemes resolve the conflicting interests of the service providers/verifiers and the service users/signers. The users want privacy while the service providers need to authenticate the users to allow only valid members obtain their services. PPA enables this by making use of secret credentials of proof of knowledge protocols. The user obtains secret credentials from the issuer by proving its identity or valid platform. Later for signing, the user proves that it holds a valid credential and its proof is correct, which is enough for the verifier to authenticate and provide its services. Thus, user privacy preserved and the verifier also provides its services to valid users only.

There are two types of PPA schemes.

1. **Verifier Anonymous Attestation :**

In these schemes, the signer is anonymous only to the verifier. The issuer and/or

revocation manager knows the identity of the signer and can link signatures. When the signer gets secret credentials from the issuer, the issuer and/or revocation manager keeps record of the signer and the credential they got. In case of dispute anywhere, this entity can open the signature and reveal the identity of the signer. In this chapter, the three types of *Verifier Anonymous Attestation* schemes are explained.

2. Fully Anonymous Attestation :

The second type of Privacy Preserving Authentication schemes are called *Fully Anonymous Attestation*. Here, neither the issuer nor the verifier can reveal the identity of the platform. These schemes are preferred when the user needs complete privacy and also prevent the case of any entity being malicious or colluding with other entity to repeal the anonymity of the user. These are explained in the next chapter.

3.1 Verifier Anonymous Authentication

These VAA schemes can be divided into three types based on the kind of revocation they do. They are, (i) Signer-based revocation, (ii) Verifier local revocation and (iii) Trusted third party based revocation.

3.1.1 Signer based Revocation

In this model, the signers get the RL from the issuer and can prove to the verifier or issuer that it has not been revoked (ie. it is a valid signer). Thereby, making valid signatures using the credentials it obtained earlier.

Generally, in signer based revocation schemes, signing will have the computational complexity of $O(N)$ or $O(R)$, where N is the group size and R is the number of revoked users. $O(N)$ if the scheme is such that the group manager updates the keys/credentials after revocation and each signer downloads the latest keys/credentials to prove it is still a valid member of the group to make new signatures. $O(R)$ when the signer has to download the latest RL and prove to the issuer/verifier that it is not present in the RL and generating new signatures. Other approaches include the use of accumulators [18] or dividing large groups into sub-groups [19].

The work of Toru Nakanishi, Hiroki Fujii, Yuta Hira and Nobuo Funabiki - '*Revocable Group Signature Schemes with Constant Costs for Signing and Verifying*' [20] presents a way to sign in $O(1)$ computational complexity. The data related to revocation, fetched by the signer from the issuer is $O(R)$ in size.

Overview

In a conventional group signature scheme, when a member joins he sends a one-way function of his secret key, $f(x)$, to the group manager. The manager returns a membership certificate

by signing on the $f(x)$. Then the group signature consists of an encryption function using the manager's public key on the $f(x)$ and a signature proof of knowledge (SPK) on the message M . When opening the message, the manager decrypts to check the $f(x)$ of sender when joining.

In [20], a new revocation mechanism for realizing constant computational complexity in signing is added. The modifications are: (i) changing the membership certificate to include a *UID* which is the ID of the member, (ii) having a revocation list - a list of revoked *UIDs* sorted and a SPK for this RL. This SPK consists of (i) certificate to prove the correctness of its signature and (ii) certificate to prove the signer's absence in RL. Any non-revoked member can prove this SPK and make valid signatures. On the other hand, a user whose *UID* is present in the RL cannot produce a valid second certificate of the SPK and hence cannot make signatures. Since the RL is sorted, inequality operations are only needed to prove non-revocation and thus signing (using SPK) has complexity $O(1)$. The size of signature is $O(1)$ and RL is $O(R)$. The overhead is the revocation complexity of the group manager. Each revocation requires $O(R)$ computation and group public key becomes $O(N)$ size because of the SPK inequations.

A secure SBR group signature scheme must satisfy three requirements,

1. Non-frameability: The signature of an honest member cannot be computed by other members or even the group manager.
2. Traceability: There can be no signatures produced that cannot be traced back to any

valid member of the group by any entity. An adversary should not be able to forge a signature that cannot be traced back to one of the members in his coalition.

3. Anonymity: An adversary cannot find the signer or link to other signatures from a particular signature σ , message M .

Scheme

Setup : This probabilistic initial setup algorithm, on input 1^l outputs the public parameters *param*.

KeyGen : This probabilistic key generation algorithm for group manager, on input N that is the maximum number of members and *param*, outputs the group public key *gpk* and the manager's secret key *msk*.

Join : Interactive protocol between probabilistic algorithms of user and manager where the user joins the manager's group w.r.t *gpk*. User on input *gpk* obtains his secret key *usk*[*i*] and the manager on inputs *gpk*, *msk* outputs *reg*[*i*] the registration log of the *i*-th user. *usk* denotes list of all users' secret keys, *reg* is the users' registration log and *i* denotes the user's ID.

Revoke : This probabilistic algorithm on inputs *gpk*, a time counter *t*, set of revoked member IDs outputs a revocation list RL_t at time *t*.

Sign : Probabilistic algorithm on inputs *gpk*, *usk*[*i*], *t* and RL_t and signed message *M* outputs signature σ .

Verify : Deterministic algorithm for verification. On inputs gpk, t, σ, M outputs ‘*valid*’ or ‘*invalid*’.

Open : Deterministic Algorithm on inputs gpk, msk, reg, t, σ and M outputs i , which indicates the signer of σ .

3.1.2 Verifier Local Revocation

In this model the revocation messages/lists are sent only to the signature verifiers; the signers are stateless. This is better than sending to signers because the number of verifiers in most cases are less than the number of signers. Consequently, there is no need to contact every signer when a user is revoked. Just passing the update to the verifiers will do.

One of the foremost schemes to implement verifier local revocation was the ‘*Group Signatures with Verifier-Local Revocation*’ by Dan Boneh and Hovav Shacham [21]. Their main motivation for using VLR is that it simplifies revocation and when signing functionality is implemented in a tamper-resistant module, allowing signers to be stateless ensures added robustness and security. Therefore, VLR is advantageous for privacy preserving attestation in the trusted computing environment.

Overview

This scheme implements verifier-local group signatures by providing the Revocation List (RL) as an additional argument to the signature verification algorithm. A signer can be revoked if his secret key is known to others or the tamper-resistance property of the TPM

chip in his device is broken (losing privacy by design). In this case, the manager can add that particular signer's token in the RL. This RL is sent to all the verifiers. Thereby, when a verifier gets a signature, it checks the RL to see whether this message originated from a signer who has been revoked. If yes, this message can be discarded by the verifier. Signatures from this signer will no longer be accepted. And all his signatures become linkable now. On the contrary, a verifier can never trace back a signature to an unrevoked user.

[21] has the special property that given the secret key of a user, it is simple to derive his revocation token as it is the left half of the secret key. Hence, any secret key that has been compromised, known to others can be easily added to the RL. This potentially eliminates the need for a trusted revocation authority (Trusted Third Party).

A secure VLR group signature scheme must satisfy three requirements,

1. **Correctness:** For all parameters generated during the generation algorithm, every signature generated by a user verify as valid, except when the user is revoked.
2. **Traceability:** There can be no signatures produced that cannot be traced back to any valid member of the group by any entity. An adversary should not be able to forge a signature that cannot be traced back to one of the members in his coalition.
3. **Selfless-Anonymity:** A group member can tell whether he generated a particular signature σ , but if he didn't he learns nothing else about the origin of σ .

Scheme

KeyGen(n) : This randomized algorithm takes as input the number of members of the group n (the number of user keys to be generated). It outputs the group public key \mathbf{gpk} , an n -element vector of user keys $\mathbf{gsk} = (\mathbf{gsk}[1], \mathbf{gsk}[2], \dots, \mathbf{gsk}[n])$, and an n -element vector of user revocation tokens \mathbf{grt} , similarly indexed.

Sign($\mathbf{gpk}, \mathbf{gsk}[i], M$) : The randomized signing algorithm takes as input the group public key \mathbf{gpk} , a user private key $\mathbf{gsk}[i]$ and a message $M \in \{0, 1\}^*$, and returns a signature σ .

Verify($\mathbf{gpk}, RL, \sigma, M$) : The verification algorithm takes as input the group public key \mathbf{gpk} , a set RL of revocation tokens (each an element of \mathbb{G}_1), a purported signature σ and a message $M \in \{0, 1\}^*$ and proceeds in two phases. First, it ensures that the signature σ is valid; then it ensures that σ was not generated by a revoked user. It accepts only if both conditions hold.

3.1.3 Trusted Third Party based Revocation

TTP-based schemes were one of the first developed for revocation of malicious users. It was initially proposed by the Trusted Computing Group (TCG) for revocation of rogue TPMs [22]. In TTP-based revocation, there is a fourth entity along with issuer, signer and verifier referred to by many terms such as revocation manager, privacy certificate authority. This is the trusted third party (TTP). They know the secret key/credential of all the users and manage revocations in case of compromised secret, tampered/stolen device or adversary attack.

In [23], Privacy Certification Authority (CA) is the trusted third party. All the users are equipped with TPM based devices. Each TPM generates an RSA key pair called *endorsement key (EK)*. The privacy CA is assumed to know the EKs of all valid TPMs. When the TPM has to authenticate itself to a verifier, it generates another RSA key pair called the Attestation Identity Key (AIK). The TPM sends the AIK public key to the privacy CA to authenticate it with respect to the EK. The privacy CA, on find the EK in it's list, issues a certificate for this AIK public key. Now, the signer can use this certificate to authenticate itself to the verifier w.r.t this AIK. Two ways to detect a rogue TPM in this scheme, (i) if the EK of a TPM is extracted and distributed, then the privacy CA can check for the corresponding public key of this exposed EK and revoke the TPM; (ii) if the privacy CA gets many requests that are authorized using the same EK, it may reject these requests and revoke the TPM. The specific parameters are set depending on the environment and application.

3.2 Fully Anonymous Authentication

The second type of Privacy Preserving Authentication schemes are called *Fully Anonymous Authentication*. Here, neither the issuer nor the verifier can reveal the identity of the platform. These schemes are preferred when the user needs complete privacy and also prevent the case of any entity being malicious or colluding with other entity to repeal the anonymity of the user. Foremost and one of the most important concept that embraces full anonymous authentication is the *Direct Anonymous Attestation* scheme.

3.2.1 Direct Anonymous Attestation

The need for DAA can be explained through this example, as given in [1], “ Consider the following authentication problem: a hardware device (e.g., a graphics chip, a mobile device, a smart phone, or a processor) wants to authenticate to a service provider that it is a genuine hardware device instead of a software simulator, so that the service provider can send a protected resource (e.g., one-time password or high definition media) to the device. One possible solution is that the hardware manufacturer assigns each device a unique device certificate. The device can authenticate to the service provider by showing the device certificate. However, such solution raises a serious privacy concern as the device certificate can uniquely identify the device.” DAA solves this problem. It is a cryptographic protocol that preserves the privacy of the user by decoupling the information about the platform’s configuration and the identity of the platform’s user [3]. In DAA, a platform consists of the host and a secure and dedicated crypto-processor called *Trusted Platform Module (TPM)*. This TPM is designed to secure the platform by integrating its cryptographic keys into the hardware (non-extractable in conventional methods). With the help of host, the TPM generates anonymous signatures, using this secret key (and its public key pair), corresponding to the current configuration of the platform. The Trusted Computing Group (TCG) standardized the RSA-based DAA in TPM specification 1.2 [24] and later included Elliptic Curve Cryptography (ECC)-based DAA in TPM specification 2.0 [25]. In the next sections, I explain the state-of-the-art DAA schemes and then present the new DAA scheme we proposed that

overcomes the shortcomings of existing schemes.

From the existing literature we can see that DAA has one type of revocation only so far. It is the signer-verifier based revocation. The signer and the verifier have to collaborate to complete the revocation check procedure. This happens online and is the obstacle preventing deployment of these schemes. In this section, I explain two state-of-the-art DAA schemes which I implemented for comparison with our scheme.

3.2.1.1 Intel's Enhanced Privacy ID

Ernie Brickell and Jiangtao Li proposed “*Enhanced Privacy ID from Bilinear Pairing for Hardware Authentication and Attestation*” [1], a DAA scheme that augments the security notion of DAA by introducing additional revocation capabilities. The original usage of DAA was for anonymous authentication of TPM. Initially, these TPMs could be revoked only if their secret key extracted by an adversary was published on the Internet or spread somehow. Then, the revocation manager on knowing this secret key would add it to the revocation list. But in the case of hardware tampering, adversary not publishing the secret key but using for malicious purposes would go undetected. Serious issue of misusing the supposedly secure hardware module this was. Hence, [1] came up with the signature-based revocation concept. The revocation manager can now revoke a device based on the signatures that were signed by the private key of the device without reducing anonymity properties. There were several revisions of the EPID scheme and the latest most efficient model is supported by the TPM 2.0 standard and ISO standardized [26].

Overview

In an EPID scheme, there are four types of entities: an issuer, a revocation manager, platforms, and verifiers. The issuer could be the same entity as the revocation manager. The issuer is in charge of issuing membership to platforms, i.e., each platform obtains a unique private key from the issuer through a join process. A platform can prove membership to a verifier by creating a signature using its private key. The verifier can verify membership of the platform by verifying the signature, but he cannot learn the identity of the platform. One important feature of EPID is that nobody besides the platform knows the platform's private key and nobody can trace the signatures created by the platform if the platform has not been tampered. Yet an EPID scheme has to be able to revoke a platform if the platform's private key has been corrupted. There are two types of revocations in EPID: (1) private-key based revocation in which the revocation manager revokes a platform based on the platform's private key, and (2) signature based revocation in which the revocation manager revokes a platform based on the signatures created by the platform. EPID chooses not to give the revocation manager traceability capability (as used in group signature schemes) to ensure maximum privacy to unrevoked platforms.

EPID enables platform authentication but not user authentication. Thus, there is no need to revoke the EPID key inside the device in case of device ownership changing as it is still a valid platform. The other disadvantage of this scheme is that the revocation method is expensive and hence with large revocation lists, practical implementation is a problem. But the unique property of EPID is to being able to revoke a private key that generated a signature without

opening the signature.

EPID scheme is secure if it satisfies the following three requirements:

1. Correctness: every signature generated by a platform can be verified as valid, except when the platform is revoked.
2. Anonymity for Unrevoked platforms: An adversary should not be able to determine which one of two private keys was used in generating a signature produced by an unrevoked platform.
3. Unforgeability: Given that all the private keys of an adversary is revoked, he should not be able to forge a valid signature.

Scheme

Setup : The issuer takes a security parameter 1^k as input and issues a group public key gpk and an issuing private key isk . **Join** : Interactive protocol between issuer and platform where issuer is given gpk, isk and platform give gpk . Platform outputs sk its secret key.

Sign : On input of the group public key gpk , a private key sk , a message m , and a signature based revocation list sRL , this sign algorithm outputs $NULL$ if sk has been revoked in sRL , or outputs a signature σ otherwise. The sRL is used by the platform to prove that it has not been revoked in sRL , i.e., it has not created any of the signatures in sRL .

Verify : On input of the group public key gpk , a message m , a private-key based revocation list pRL , a signature based revocation list sRL , and a signature σ , this verify algorithm

outputs *valid, invalid, orrevoked*. The verifier and signer use the same *sRL*, only then the verification would pass. **Revoke** : A revocation manager as private key *sk* to the *pRL* given the *gpk*. And on inputs *gpk, m, σ onm*, updates *sRL* by placing σ to *sRL*.

For private key based revocation, the *pRL* is not sent to the platform, thus making this part *verifier local revocation*. Once a platform is on the *pRL*, it loses its privacy and all its signatures can be linked. Thereby, an issuer cannot revoke a platform, link signatures without knowing the private key of the platform. This upholds the full anonymous attestation property of DAA.

3.2.1.2 IBM's Enhanced Privacy ID

Overview

Jan Camenisch, Manu Drijvers and Anja Lehmann came up with “*Anonymous Attestation Using the Strong Diffie-Hellman Assumption Revisited*” [2] presenting a qSDH-based DAA scheme with support for attributes and signature-based revocation. They pointed out the flaws in the security proofs of existing schemes, including [1], and came up with a scheme that uses BBS+ credentials, type-3 pairing setting and a new way to prove knowledge of credentials making [2] more efficient than any existing DAA scheme. They have incorporated the idea of signature-based revocation from [1] and membership credentials including attributes along with the TPM key from [27] and made the proofs secure while making the scheme efficient. They also present the model with split up of work between the TPM and

host making it closer to a practical implementation.

The proofs of this scheme are more efficient to generate as it works completely in \mathbb{G}_1 , avoiding the less efficient groups $\mathbb{G}_2, \mathbb{G}_T$ and the pairing operations. And verification requires only two pairing operations, the rest of the computation takes place in \mathbb{G}_1 .

Scheme

IssuerSetup : The issuer creates a key pair of BBS+ signature scheme registers the public key with the certificate authority. **JoinRequest** : The join protocol runs between the issuer and a platform. The platform authenticates to the issuer and, if the issuer allows the platform to join with certain attributes, obtains a credential that subsequently enables the platform to create signatures. A unique sub-session identifier distinguishes several join sessions that might run in parallel. **JoinProceed** : The join session is complete when the issuer receives an explicit input telling it to proceed with the join session identifier and issues attributes *attrs*. **SignRequest** : The sign protocol runs between the TPM and host of the platform. After joining, together they can sign a message *m* with respect to a basename *bsn*, attribute predicate and signature-based revocation list *sRL*. Again, a unique sub-session identifier is used to allow for multiple sign sessions. **SignProceed** : Interactive protocol between the TPM and host. TPM proves knowledge of the credential and the host randomizes the credential and then completes the ZK proof. Then the platform computes ZK proof for each entry in the *sRL*, proving its credential is not present in it. Finally, the platform outputs the signature σ , where the length of the signature is linearly dependent on the size of the *sRL*. **Verify** : The verifier checks whether a signature σ on message *m* with respect to basename

bsn , attribute disclosure, private key revocation list RL and signature revocation list SRL is valid. **Link** : The link algorithm allows the verifier/issuer to check whether two signatures σ, σ' , on messages m, m' respectively, that were generated for the same basename bsn were created by the same TPM.

Since [2] split the workload between TPM and host, it is easier to implement this scheme closer to real-world application scenario. This scheme is considered the state-of-the-art benchmark for comparison with our scheme LASER.

Chapter 4

Analysis of PPA schemes

In this chapter, I analyze the pros and cons of the PPA schemes discussed in the previous chapter. One common trend among all the schemes is that, they are not practical to be deployed in a real-world setting. This analysis of existing schemes help us identify their shortcomings and understand what properties are important for a PPA scheme to be practical. And using this knowledge, we designed a new scheme which will be explained in the next chapter.

4.1 Signer-based Revocation

Traditional signer based revocation group signatures schemes were not suitable for large groups because of the large $O(N), O(R)$ complexity during signing. But [20] overcomes this by making the signing sufficiently fast even for larger or dynamic groups. But the

compensation is the long public key with $O(N)$ size. Though the public key is distributed once only during joining, when there are millions of members the long time of downloading may be considered a disadvantage. The authors have suggested an extension to the scheme which makes the public key size $O(\sqrt{N})$ but signing has extra costs. Using this extension, for 112-bit security with a million members, the public key comes around to 100 kilobytes in size. And since the users download it once only, this may seem acceptable on PCs and smart phones.

The signer fetching $O(R)$ size data RL is a cause for concern in most SBR schemes. Along with the linear dependency of public key size with group size, the communication cost is high [28]. This is where VLR schemes are considered than SBR schemes.

One possible application area for SBR schemes are authentications for web services where public key is distributed once when joining. And, these schemes are also useful in ID management, anonymous credential systems.

4.2 Verifier-local Revocation

In VLR schemes, signature verification time grows linearly to the number of revoked users. But in general, it is desirable to have VLR system where verification time is constant. Boneh presents a slight modification, to the scheme at the cost of partial anonymity, where a table lookup will suffice to check whether a user is present in RL. He justifies saying *for some applications, the trade-off between partial linkability and efficient revocation might be*

acceptable.

One big advantage of this scheme is that signatures are short: only 141 bytes for a standard security level. They are shorter than group signatures built from strong-RSA assumption, BBS short group signatures [29]. This is made possible by making use of *elliptic curves*.

The biggest disadvantage being *backward linkability*. All signatures produced by a revoked member are linkable. Anonymity of signatures produced before the user was revoked is also compromised after revocation. In the case of a member leaving voluntarily or member's secret key being stolen, anonymity of signatures should be ensured [30]. But this scheme falls short here.

One possible application area for VLR schemes are mobile environments where the mobile hosts anonymously communicate with the servers [30]. The computational costs of verifiers depend on number of revoked users; signer costs are low.

4.3 Trusted Third Party-based Revocation

The major drawback of using a TTP is that it needs to be involved in every transaction [22]. The TTP should be available for long periods of time and still be secure as a CA that operates offline. The other disadvantage being the case of the TTP and verifier colluding or TTP records being compromised/revealed. The verifier or any adversary can identify the TPMs from the signatures and hence no anonymity. This is true for all TTP-based schemes and hence the dearth of TTP-based revocation for privacy preserving authentication.

4.4 Direct Anonymous Attestation

The existing DAA schemes share a common disadvantage. Since both [1, 2] are *signer-verifier based revocation* schemes, signing and verifying operations increase linearly with the size of the revocation list. The signer obtains the RL from the issuer before creating a signature. Now, for each entry in this list, the signer has to prove that it is not its secret key that has been revoked. And the verifier checks all these proofs and decides whether has been revoked or not. This happens in the online stage and this makes the scheme impractical to deploy in applications. If there are a million revoked entries in the list, generating these proofs take enormous amount of time. Add to that the communication overhead in sending the massive signature to the verifier. And if this has to happen for every signature generated, the overhead cost would surpass all other costs.

The suggestion provided in these schemes to overcome this problem, is to revert to a *rekey-based revocation* system. That is, as the size of the RL crosses a threshold value, reset the group and issue new credentials to all the unrevoked members. At first sight this may seem possible, but in large networks issuing new credentials at regular intervals to the unrevoked members again results in massive communication overhead.

Since these are the only two revocation methods used in DAA and both seem to be inefficient, the need for a new revocation method or at least a modification to the existing methods is necessary. Only then DAA schemes could be used in the real-world setting.

Chapter 5

Lightweight Anonymous Attestation with Efficient Revocation (LASER)

“Lightweight Anonymous Attestation with Efficient Revocation” [3] is the DAA scheme we propose for it is implementation efficient in a practical setting and scalable with the increasing size of revocation list. It overcomes the two most important hurdles of DAA schemes by moving the heavy computation overhead to the offline side and also taking into consideration the fact that not all applications need absolute anonymity. For some applications, the trade-off between partial anonymity and efficient revocation for practical implementation might be acceptable [21].

The major hindrance in the widespread adoption of DAA schemes is that, for each entry in the revocation list the platform needs to generate a proof of non-revocation of its secret

key with respect to the entry and include it as part of its signature. Therefore, three things increase linearly with the size of revocation list: (i) computational complexity of generating a signature (ii) computational complexity of verifying the signature (iii) communication overhead in terms of the length of the signature. Since these process happen online (during the signing-verifying phase) the time taken for each signature process is high. These prevent most applications from deploying DAA schemes for privacy preserving authentication. LASER overcomes these problems by changing the system as explained in the next section.

5.1 Overview

In LASER, the platform obtains a membership credential **memCre** from the issuer by proving in ZK its secret key. But, unlike the existing DAA schemes, the platform does not use **memCre** to generate signatures on messages sent to the verifier. Instead, the TPM and the host run another protocol to obtain signing credentials which are used for signing messages. The *signCre* obtaining protocol happens as, (i) the TPM and the host generate the proof of non-revocation of the TPM's secret key **tsk** corresponding to a basename-based revocation list **baseRL**, and send it to the issuer; (ii) the issuer checks the revocation status of the platform against the revocation lists; (iii) If the verification is successful, the issuer sends a *signing credential* **signCre_j** to the platform. The platform may run this protocol for m_s number of times to obtain m_s number of signing credentials, **signCre_j**, $\forall j \in [1, m_s]$. Each signing credential **signCre_j** contains a set of m_a *alias credentials* **aliasCre_{jk}**, $\forall k \in [1, m_a]$, where

each alias credential is a signature on \mathbf{tsk} using \mathbf{isk} . Each $\mathbf{aliasCre}_{jk}$ is also associated with an *alias token* x_{jk} .

In the signature generation protocol in LASER, the TPM and the host of the platform use an alias credential $\mathbf{aliasCre}_{jk}$ to generate a signature for the verifier. In the signature, the platform reveals the alias token x_{jk} associated with the alias credential $\mathbf{aliasCre}_{jk}$. The verifier runs the signature verification algorithm to check the validity of the signature and the revocation status of the alias token x_{jk} received from the platform. The revocation status of x_{jk} is determined by searching through an *alias token-based revocation list* \mathbf{atRL} . The \mathbf{atRL} contains all the revoked alias tokens, and is published by the issuer. Thus, this *online* step is similar to the *verifier-local revocation* schemes.

The platform does not need to generate any proof of knowledge of non-revocation of its TPM's secret key in the signatures on messages sent to the verifier. Thereby, most of the burden of the revocation check procedure in LASER is shifted from the on-line signature generation to the off-line acquisition of signing credentials phase. This unique feature of LASER brings about a number of important practical advantages. First, during the signature generation protocol, the platform is *not* burdened with any computations related to the revocation check procedure, and thus results in a significant reduction in the computational complexity of signature generation. Second, the length of the signature communicated to the verifier is constant, and does not grow proportionally with the length of the revocation list. Third, LASER enables the verifier to employ a computationally efficient search procedure to check the revocation status of the platform that has generated a given signature,

and hence results in a significant reduction in the computational complexity of signature verification protocol. These advantages are especially important when a DAA scheme needs to be deployed in a network with a large number of nodes [3].

The four security properties of LASER are,

1. *Correctness*: This property requires that if a signature is honestly generated by a platform, any entity can correctly verify the authenticity of the signature.
2. *Adaptable Anonymity*: This notion consists of the following properties—*anonymity* and *adaptable unlinkability*.
 - (a) *Anonymity*: This property requires that no entity (including the issuer) is able to identify the platform which has generated a given signature.
 - (b) *Adaptable unlinkability*: This notion requires that the platform is able to adaptably control whether or not two signatures can be linked. Here, linking two signatures means that they are proved to be generated by the same platform. For any two signatures, the platform may select *one* of the following three properties of adaptable unlinkability.
 - i. *Absolute unlinkability*: The two signatures cannot be linked either by the issuer or by the verifier.
 - ii. *Conditional unlinkability*: The two signature cannot be linked by the verifier, but can be linked by the issuer. This is a new notion of unlinkability in a DAA scheme introduced in this paper.

- iii. *No unlinkability*: The two signature can be trivially linked by the verifier as well as the issuer.
- 3. *Traceability*: This property requires that no colluding set of platforms (even consisting of the entire group) can create a valid signature that does not belong to any platform.
- 4. *Non-frameability*: This property requires that no colluding set of entities (including the issuer) can create a valid signature that belongs to any one of the non-colluding platforms.

5.2 Scheme

Setup

This protocol is performed by the issuer. On input of 1^λ , the issuer's secret key isk (known only to the issuer) and group public key gpk are outputs.

Join

This protocol is performed among the TPM, the host and the issuer. The inputs to the TPM and the host are gpk . The inputs to the issuer are gpk and isk . In this protocol, the TPM generates a TPM's secret key tsk , and outputs a TPM's public key tpk and a key handle $hd1$ which specifies the location of the tsk in the secure memory of TPM. The tsk is known only to the TPM. Also, the host outputs a platform's membership credential $memCre$.

SignCreGen

This protocol is performed among the TPM, the host and the issuer. The inputs to the TPM are `gpk` and `tsk`. The inputs to the host are `gpk`, `hdl`, `tpk`, `memCre`, and a basename-based revocation list `baseRL`. The inputs to the issuer are `gpk`, `isk`, `baseRL`, and a database registry `reg`. Further, for each $i \in [1, m_r]$, where m_r is the number of tuples in `baseRL`, the host and the TPM generate a signature σ_i . The honestly generated signatures $\sigma_i, \forall i \in [1, m_r]$, act as the proof that the TPM's secret key `tsk` is not revoked. In this procedure, the signature σ_i presents the proof of inequality of discrete logarithms. For each $i \in [1, m_r]$, the issuer checks the revocation status of the platform by verifying the validity of signature σ_i . This protocol outputs the platform's signing credential

$$\text{signCre}_j = (\text{aliasCre}_{j_1}, \dots, \text{aliasCre}_{j_{m_a}}).$$

SelectAliasCre

After the TPM and the host run the `SignCreGen` protocol with the issuer for m_s number of times to obtain m_s signing credentials represented as `signCre` = (`signCre`₁, ..., `signCre` _{m_s}).

Then the host generates a signing credential usage list `signUL`, and an alias credential usage list `aliasUL`. This selection algorithm is performed by the host. The inputs to this algorithm are `signCre`, `rul`, `signUL`, and `aliasUL`. This algorithm outputs an alias credential `aliasCre` _{j_k} .

Sign

This signature generation protocol is performed between the TPM and the host. The inputs to the TPM are `gpk` and `tsk`. The inputs to the host are `gpk`, `hdl`, `tpk`, `aliasCre` _{j_k} , and a

message to be signed $M \in \{0,1\}^*$. This protocol outputs a signature σ_s which presents the proof of knowledge of a valid alias token x_{jk} , a basename and a corresponding public-name associated with f , and the BBS+ signature (A_{jk}, y_j, z_{jk}) on (f, x_{jk}) .

Verify

This verification algorithm takes the group public key \mathbf{gpk} , a purported signature σ_s , a message M , and an alias token-based revocation list \mathbf{atRL} as inputs. This algorithm verifies: (1) whether the signature is honestly generated, and (2) the revocation status of the alias token used to generate the signature. If both of these verification steps (as shown below) are successful, this algorithm outputs the value *valid*; otherwise it outputs the value *invalid*.

Revoke

This is the signature-based revocation algorithm run by the issuer. The inputs to this algorithm are \mathbf{gpk} , \mathbf{reg} , a purported signature σ_s associated with an alias token x_{jk} , a message M , \mathbf{baseRL} , and \mathbf{atRL} . This algorithm outputs the updated revocation lists, \mathbf{baseRL}' and \mathbf{atRL}' .

Link

This signature linking algorithm is performed by the issuer. The inputs to this algorithm are \mathbf{gpk} , \mathbf{reg} , two signatures σ_s^* and σ_s' corresponding to alias tokens $x_{j^*k^*}$ and $x'_{j'k'}$, and corresponding messages M^* and M' . This algorithm outputs the value *true* if the issuer links the two signatures σ_s^* and σ_s' to the same platform; otherwise it outputs the value *false*.

Identify

This signature tracing algorithm takes `gpk`, a purported signature σ_s , a message M , and a TPM's secret key `tsk*` as inputs. This protocol outputs the value `true` if σ_s is proved to have been generated using `tsk* = f*`; otherwise it outputs the value `false`.

5.3 Implementation and Analysis

I have implemented the three schemes [1, 2, 3] on a PC using the OpenSSL and PBC libraries in C language. All the entities - TPM, host, issuer and verifier - were simulated on the same PC, so the communication overhead is negligible in this implementation. Since these are ECC-based DAA schemes, I chose the Barreto-Naehrig's 256-bit curve which is standardized by the Trusted Computing Group. Specifically, we use the "Type F" internal described in the pairing-based cryptography library which is constructed on the curve of the form $y^2 = x^3 + 3$ with embedding degree 12 where the lengths of an element in \mathbb{Z}_p^* and \mathbb{G}_1 are 256 bits and 512 bits, respectively [5]. With this curve, the DAA provides a level of security which is approximately the same level of security as a symmetric key encryption with a key size of 128 bits, or an RSA signature with a modulus size of 3072 bits.

The 2 plots generated show the comparison of computational complexity of the three schemes in the offline and online mode. All the operations which can be pre-computed or stored, and do not need to be generated in real time are classified as offline operations. The offline operations include the computations at the TPM, the host and the issuer for establishing

the platform's membership and/or signing credentials. The operations which need to be performed in real time are classified as online operations. The online operations include the computations at the TPM and the host for generating the signature, and the computations at the verifier for verifying the signature.

The figure 5.1 shows that LASER is at a huge disadvantage when compared to B-EPID and CDL-EPID. This is because of the ZK-proof of non-revocations being included in the communication with issuer. This helps in efficient online computation of the scheme, as seen in figure 5.2, with reasonable size of signature. Since the online stage is important to applications than the offline stage, this trade-off is practical and makes the implementation of DAA scheme plausible for applications requiring privacy preserving authentication. In the online phase, with high number of revoked users, B-EPID and CDL-EPID have enormous computation overhead which makes generation and verification of signature cumbersome along with the communication overhead of signature length. One suggestion to avoid this problem of large revocation lists, as given in B-EPID and CDL-EPID, is to reset the group and issue new credentials to unrevoked platforms. This isn't practical for DAA in large networks or applications with stringent latency requirements. LASER has constant signature generation and verification time while having constant length. This makes LASER suitable for applications described in the next chapter.

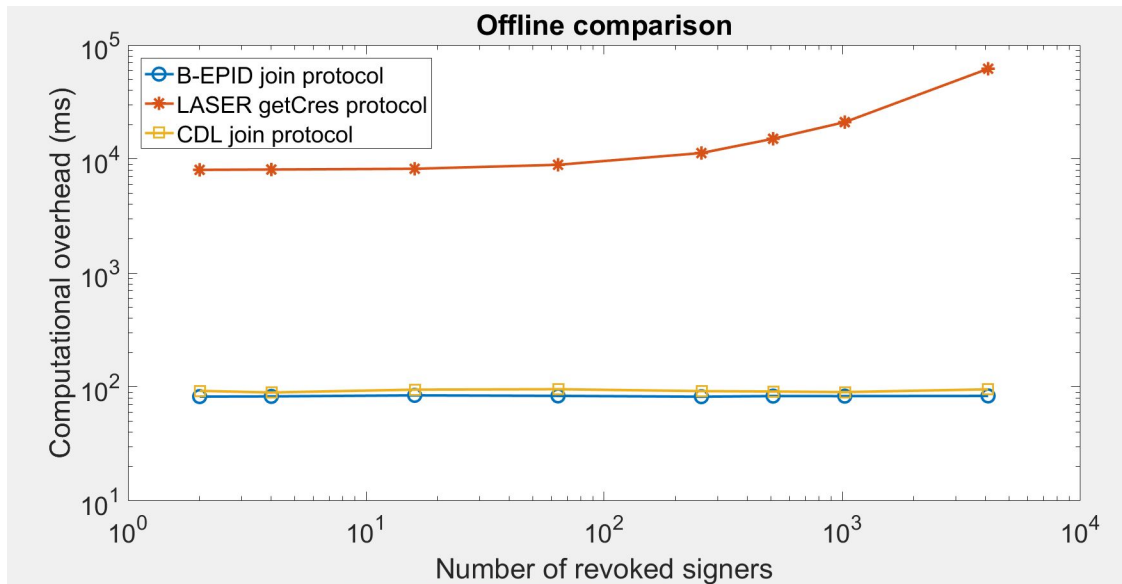


Figure 5.1: Comparison of offline computational overhead between B-EPID[1], CDL-EPID[2] and LASER[3]

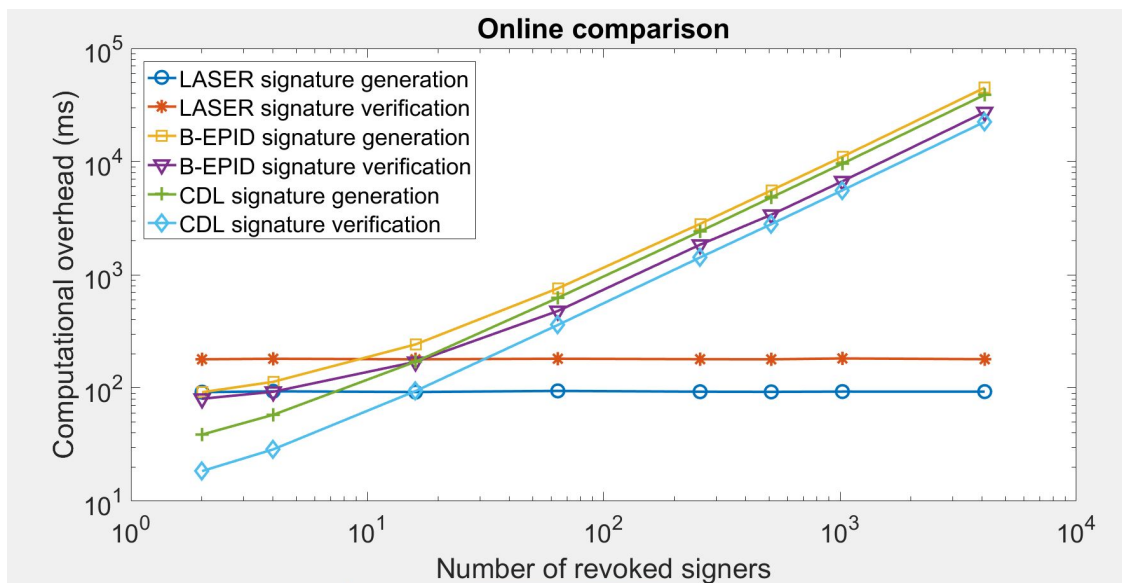


Figure 5.2: Comparison of online computational overhead between B-EPID[1], CDL-EPID[2] and LASER[3]

Chapter 6

Applications

6.1 Multiple Cloud Collaborative Environment

This is an emerging area in cloud computing where security and privacy are important features. Conceptually similar to e-commerce platform where Cloud Service Providers (SP) list their services and users can subscribe/register to different services. The platform on which the providers and users meet is the Cloud Service Broker's (CSB) platforms. [31] defines CSB as a “cloud service partner that negotiates relationships between cloud service customers and cloud service providers”. Such a collaborative environment is different from the tradition setup where all resources, services belong to a single cloud provider.

[32] gives a general setup of this multiple cloud collaborative environment. CSPs register their services with the brokering service. Users can choose from these services on their own or

specify their requirements and preferences for selection of a reliable and trustworthy service by the CSB. Then, the service User (SU) negotiates service level agreement with the CSP. This three way relationship is based on the cooperation and trust of all parties. Hence, to ensure security and privacy, privacy preserving authentication schemes could be made use of here.

The key requirements for a privacy preserved multiple cloud collaborative environment are,

1. Unlinkability: There must not be any link between cloud user's details and its subscriptions, requirements or preferences such that it could be used for malicious purposes.
2. Anonymization: All the identifying details must be anonymized to preserve the privacy of collaborating entities.
3. Control and Audit: The brokering service must not perform maliciously and be bound by loyalty to a single cloud service provider.

6.1.1 Applicability of LASER for MCCE

Consider a scenario where there are 500,000 subscribers to a Cloud Service Broker. And these subscribers pay monthly to get access to various services. Likewise, assume there are over 100 Cloud Service Providers who have partnered with this CSB. It could be the case that the CSB and CSP have prior agreement on the partnership and Service Users (SU) pay to CSB. These services could be recommendation engines, media streaming services, real-time information providers, storage services etc. Now, since the CSB acts as the central nervous

system to this body of services, every time a user subscribes to the service the CSB can know a little bit about more about the user. And with all the subscriptions and usage pattern, the CSB can collect an almost complete profile of the SU. This is invasion of privacy since his data could be sold, used to track the SU or figure out their actions. In a situation like this, privacy of the SU needs to be preserved. On the other hand, the CSB needs to keep track of the valid user details and subscriptions they are entitled to use based on their package. Only then, illegal users can be denied access. Thus, protecting privacy and availability of services to only valid users are the two main necessities of this application. One could use anonymous credential systems where an user gets anonymous credentials which he/she can utilize to authenticate to the different services. But the case of SU sharing or distributing their credentials to others who aren't allowed in the system becomes a problem. Therefore, the anonymous credentials need to be tied to the valid SU. Here we can use the Direct Anonymous Attestation scheme making use of the in-built Trusted Platform Module in the SU's device. And LASER would fit in perfectly to all the parties involved.

Let the CSB be the issuer, SU is the signer (platform containing host and TPM) and CSPs are the verifiers. In the **Setup** phase, the CSB sets the group parameters to be used by all entities. In the **Join** phase, the SU registers with the CSB and becomes a subscribed user. It gets membership credentials and the CSB records the user details and subscription charges. In the next step, **SignCreGen**, the SU authenticates itself anonymously with zero-knowledge proof of knowledge of its **memCre** to the CSB as a valid, subscribed user and gets signing credentials anonymously. The CSB cannot link the signing credentials to the user or the

membership credentials. In this phase, the SU also proves to the CSB that its profile is not listed in any of the revocation lists by, again, proving using ZK-SPK. Once the CSB verifies the signatures generated by the SU for correctness and validity, issues the signing credentials. Depending on the needs and demands of the CSB and SUs, the number of alias credentials associated with each signing credential could be set to a fixed number and generated in the `SelectAliasCre` step. Now using these credentials the SU can authenticate itself to any of the CSP without revealing its identity and obtain their services, produce signatures in the `Sign` phase. CSPs do the `Verify` step and CSBs are responsible for the `Revoke`, `LinkandIdentify` steps. Revocations happen if unauthorized SUs try to access any service, authorized SU's secret credentials are compromised/lost/stolen, SUs breaching service level agreement etc.

There are four ways in which the SU can use its signing credentials and alias credentials. Depending on the anonymity and unlinkability desired, the SU can choose one of these four methods or a mix of any/all of the four. Assume the SU is subscribed to 10 CSPs and contact all of them with at least one signature a day for a month of 30 days (monthly subscription).

1. **Absolute Unlinkability:** The SU obtains 300 signing credentials with 1 alias credential in each of them. Thus, for the 300 signatures throughout the month (10 per day to the 10 CSPs for 30 days) the SU can use a different signing credential. No CSP can link the signatures they get on different days from this SU. Different CSPs cannot link the signatures they receive even if they collude. And the CSB also cannot link any signature. This provides absolute unlinkability to all the signatures generated by the SU.

2. Absolute and Conditional Unlinkability:

(a) Case 1: The SU gets 30 signing credentials with 10 alias credentials in each of them. The SU can use one signing credential per day and one alias credential within that signing credential per CSP for the day. Thereby, none of the CSPs can link the signatures but the CSB can link the signatures produced by the SU for that day. This means, the signatures generated in a day are conditionally unlinkable and signatures generated on different days are absolutely unlinkable.

(b) Case 2: The SU gets 10 signing credentials with 30 alias credentials in each of them. Here, the SU can assign a different signing credential per CSP and use different alias credentials within the same signing credential for different days to the same CSP. In this case, the CSB can link signatures generated for the same CSP by the SU but cannot link signatures generated for different CSPs. And the CSPs can never link any of the signatures. This means, the signatures generated for a CSP are conditionally unlinkable and the signatures generated for different CSPs are absolutely unlinkable.

3. Conditional Unlinkability: The SU gets one signing credential with 300 alias credential from the CSB. And for each of the 300 signatures, use a different alias credential. Thus, all the signatures are conditionally unlinkable as only the CSB can link them.

4. Linkable Signatures: In this scenario, the SU produces many signatures with the same alias credential of a signing credential because it wants the CSP to link its signatures.

This could be useful in a scenario where the SU wants the recommendation service to learn his preferences, choices or usage. Thus, the CSP can link different signatures and keep track of the SU patterns.

As can be seen in figure 6.1, the choice of unlinkability changes the efficiency and complexity of the computational overhead. This is because of the number of signing credentials obtained by the user. While in other schemes, the user needs to obtain a new signing credential each time he generates a signature (in CDL-EPID, for 300 signatures, the line is same as $m_s = 300$ here), in LASER that problem is alleviated to a certain degree because of the use of alias credentials. This trade-off between efficiency and partial anonymity works well in many application scenarios. Also, note that the expensive portion happens in the offline portion. The online signature overhead remains constant.

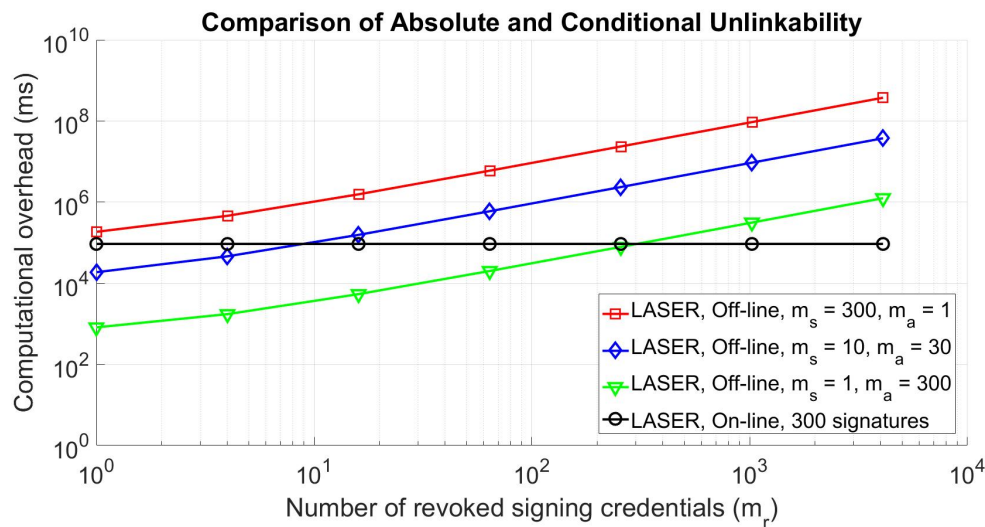


Figure 6.1: Comparison of absolute and conditional unlinkability in LASER

And as explained in the previous chapter, the efficiency and communication overhead varies depending on the number of signing credentials the SU needs. But in any case, in the long run, LASER performs better than other DAA schemes. And for some applications, trade-off between partial linkability and efficient revocation works better [21]. Also, this trade-off makes the practical implementation of a DAA scheme possible.

6.2 DAA for Embedded Mobile Devices

As explained in [33], DAA could be used for applications in embedded mobile devices. This is conceptually similar to the multiple cloud collaboration environment example explained in the previous section. Here, the mobile network operator acts as the issuer and content service providers are the verifiers. The mobile devices of the user are the platforms. Since, all mobile devices comes with embedded secure hardware (Mobile Trusted Module), DAA could be applicable here just like in the previous example. LASER would fit in perfectly as the revocation list size would be large in case on mobile devices. The content service providers partner with the mobile network operators to provide services through their applications, for which the users pay monthly. Since users carry their mobile devices everywhere, location privacy is all the more important and DAA becomes necessary if they don't want to be tracked. Using LASER, the type of unlinkability could be chosen application-wise by the user to protect their privacy and security while also maximizing the efficiency of the service.

6.3 Usage Statistics and Error Reports

Microsoft collects usage statistics and error reports from its customers for almost all of its applications [34, 35, 36]. This is on by default, but the customer can choose to turn this option off. As part of the *Customer Experience Improvement Program*, Microsoft analyzes the usage pattern of the customer to improve its product and provide regular updates. These reports are sent to the MS servers when the customer device is connected to the Internet and the application is not in use. Information like IP address, operating system version, browser/application version, hardware ID, regional and language settings etc are logged into a file and sent along with a Globally Unique Identifier (GUID) Microsoft assigns to each customer. Though it says only non-personal data are collected, some reports unintentionally include individual identifiers like serial number of a device.

These reports cannot be reviewed by the user because of the encoding MS enforces. Thereby, in such a case instead of pseudonym-based anonymity (use of GUID), the use of a DAA scheme will be appropriate. Since the reports are generated specific to the applications in a particular device, the TPM in that device can be used to sign anonymously. During OS or application installation, patch/update installation new credentials can be stored. And for each application, a different signing credential could be used to prevent linking of signatures between reports from different applications. Since MS uses the same GUID for all the reports from a customer, it is possible to link and understand the usage pattern of a customer from the meta-data. The use of DAA schemes could avoid this situation.

6.4 PPA for VANETs

One of the most popular applications for PPA schemes are the *Vehicular Ad-hoc Networks*. In VANETs, vehicles communicate with each other and also with road side units. Since the messages are only important than the identity of the sender in almost all the cases, PPA schemes fit in seamlessly. DAA schemes may not be applicable here because of latency constraints (since TPMs are resource constrained devices, computations take a lot of time) and hence groups signatures are used primarily. Each vehicle could sign their messages anonymously and transmit. And in case of dispute, a trusted third party can open the signature. Verifier anonymous attestation is more applicable for this purpose. And since the messages cannot be linked, tracking of the user isn't possible.

Chapter 7

Conclusion

In this thesis, we established the need for privacy in today's world. The use of *Privacy Preserving Authentication* schemes may involve significant overhead and complexity but should be made mandatory to protect the privacy of the people. I reviewed the existing PPA schemes (both verifier anonymous and fully anonymous schemes) and mentioned the application scenarios where each scheme might have been applicable. After analyzing the pros and cons of the various schemes, I pointed out why almost all of them are impractical to implement in any application with reasonable demands. Then, I elaborately described the new LASER scheme we developed that overcomes the shortcomings of the existing methods. After a thorough and comprehensive analysis of existing PPA schemes, the authors of [37] recently concluded that *revocation remains the major performance bottleneck of modern PPA schemes*, and that further research is urgently needed to design schemes offering better scalability with regard to revocation. And in LASER we proposed a practical and scalable

revocation algorithm that works well with absolute and conditional anonymity scenarios. Finally, I explained in detail an application where our scheme would fit in before concluding with other application areas where our scheme and, in general, other PPA schemes could be used to ensure privacy of the user.

Future work could include the deployment of LASER in a real-world test scenario and analyze the efficiency and security of the scheme. Also, to get PPA schemes to be deployed in all applications, make the people (customers and citizens) understand the importance of privacy and security and ask for it.

Bibliography

- [1] E. Brickell and J. Li, “Enhanced privacy id from bilinear pairing for hardware authentication and attestation,” *International Journal of Information Privacy, Security and Integrity* 2, vol. 1, no. 1, pp. 3–33, 2011.
- [2] J. Camenisch, M. Drijvers, and A. Lehmann, “Anonymous attestation using the strong diffie hellman assumption revisited,” in *International Conference on Trust and Trustworthy Computing*. Springer, 2016, pp. 1–20.
- [3] V. Kumar, “Transmitter authentication in dynamic spectrum sharing,” Ph.D. dissertation, Virginia Tech, 2017.
- [4] F. Zhang, R. Safavi-Naini, and W. Susilo, “An efficient signature scheme from bilinear pairings and its applications,” in *International Workshop on Public Key Cryptography*. Springer, 2004, pp. 277–290.
- [5] B. Lynn, “Pairing-based cryptography library,” 2015, [Online; accessed 19-April-2017].
[Online]. Available: <https://crypto.stanford.edu/pbc/>

- [6] B. Ewanick, “Zero knowledge proof,” 2011.
- [7] D. Chaum and E. Van Heyst, “Group signatures,” in *Workshop on the Theory and Application of Cryptographic Techniques*. Springer, 1991, pp. 257–265.
- [8] M. Bellare, D. Micciancio, and B. Warinschi, “Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions,” in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2003, pp. 614–629.
- [9] E. Brickell, “An efficient protocol for anonymously providing assurance of the container of a private key,” *Submitted to the Trusted Computing Group*, 2003.
- [10] M. H. Au, W. Susilo, and Y. Mu, “Constant-size dynamic k-TAA,” in *Proceedings of the 5th International Conference on Security and Cryptography for Networks (SCN)*, 2006, pp. 111–125.
- [11] D. Boneh, X. Boyen, and H. Shacham, “Short group signatures,” in *Advances in Cryptology - CRYPTO*, 2004, pp. 41–55.
- [12] J. Camenisch and A. Lysyanskaya, “Signature schemes and anonymous credentials from bilinear maps,” in *Advances in Cryptology - CRYPTO*, 2004, pp. 56–72.
- [13] T. C. Group, “Trusted platform module (tpm) summary,” 2008, [Online; accessed 19-April-2017]. [Online]. Available: <https://trustedcomputinggroup.org/trusted-platform-module-tpm-summary>

- [14] J. D. Osborn and D. C. Challener, “Trusted platform module evolution,” *Johns Hopkins APL Technical Digest (Applied Physics Laboratory)*, vol. 32, no. 2, pp. 536–543, 2013.
- [15] J. Wang, Y. Shi, G. Peng, H. Zhang, B. Zhao, F. Yan, F. Yu, and L. Zhang, “Survey on key technology development and application in trusted computing,” *China Communications*, vol. 13, no. 11, pp. 70–90, 2016.
- [16] D. Boneh, “The decision Diffie-Hellman problem,” in *Proceedings of the Third International Symposium on Algorithmic Number Theory*, 1998, pp. 48–63.
- [17] D. Boneh and X. Boyen, “Short signatures without random oracles and the SDH assumption in bilinear groups,” *Journal of Cryptology*, vol. 21, no. 2, pp. 149–177, 2008.
- [18] J. Camenisch and A. Lysyanskaya, “Dynamic accumulators and application to efficient revocation of anonymous credentials,” in *Annual International Cryptology Conference*. Springer, 2002, pp. 61–76.
- [19] T. Nakanishi, F. Kubooka, N. Hamada, and N. Funabiki, “Group signature schemes with membership revocation for large groups,” in *Australasian Conference on Information Security and Privacy*. Springer, 2005, pp. 443–454.
- [20] T. Nakanishi, H. Fujii, Y. Hira, and N. Funabiki, “Revocable group signature schemes with constant costs for signing and verifying,” in *International Workshop on Public Key Cryptography*. Springer, 2009, pp. 463–480.

- [21] D. Boneh and H. Shacham, “Group signatures with verifier-local revocation,” in *Proceedings of the 11th ACM conference on Computer and communications security*. ACM, 2004, pp. 168–177.
- [22] E. Brickell, J. Camenisch, and L. Chen, “Direct anonymous attestation,” in *Proceedings of the 11th ACM conference on Computer and communications security*. ACM, 2004, pp. 132–145.
- [23] T. C. P. Alliance, “Trusted computing platform alliance (tpca) main specification version 1.1 a,” *TCG Public Review*, 2001.
- [24] Trusted Computing Group, “TPM main specification,” <http://www.trustedcomputinggroup.org/tpm-main-specification/>, accessed: Apr 24, 2017.
- [25] —, “TPM library specification,” <http://www.trustedcomputinggroup.org/tpm-library-specification/>, accessed: Apr 24, 2017.
- [26] “Information technology - security techniques - anonymous digital signatures - part 2: Mechanism using a group public key,” International Organization for Standardization, Standard ISO/IEC 20008-2, 2013.
- [27] L. Chen and R. Urian, “Daa-a: direct anonymous attestation with attributes,” in *International Conference on Trust and Trustworthy Computing*. Springer, 2015, pp. 228–245.

- [28] B. Libert, T. Peters, and M. Yung, “Scalable group signatures with revocation,” in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2012, pp. 609–627.
- [29] D. Boneh and X. Boyen, “Short signatures without random oracles,” in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2004, pp. 56–73.
- [30] T. Nakanishi and N. Funabiki, “Verifier-local revocation group signature schemes with backward unlinkability from bilinear maps,” in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2005, pp. 533–548.
- [31] M. Guzek, A. Gniewek, P. Bouvry, J. Musial, and J. Blazewicz, “Cloud brokering: Current practices and upcoming challenges,” *IEEE Cloud Computing*, vol. 2, no. 2, pp. 40–47, 2015.
- [32] N. Komninos and A. K. Junejo, “Privacy preserving attribute based encryption for multiple cloud collaborative environment,” in *Utility and Cloud Computing (UCC), 2015 IEEE/ACM 8th International Conference on*. IEEE, 2015, pp. 595–600.
- [33] C. Wachsmann, L. Chen, K. Dietrich, H. Löhr, A.-R. Sadeghi, and J. Winter, “Lightweight anonymous authentication with tls and daa for embedded mobile devices,” in *International Conference on Information Security*. Springer, 2010, pp. 84–98.
- [34] Microsoft, “Microsoft privacy statement,” 2017, [Online; accessed 23-April-2017]. [Online]. Available: <https://privacy.microsoft.com/en-us/privacystatement/>

- [35] —, “Microsoft customer experience improvement program,” 2009, [Online; accessed 23-April-2017]. [Online]. Available: <https://www.microsoft.com/products/ceip/EN-US/default.aspx>
- [36] —, “Privacy statement for the microsoft error reporting service,” 2010, [Online; accessed 23-April-2017]. [Online]. Available: <https://privacy.microsoft.com/en-US/microsoft-error-reporting-privacy-statement>
- [37] M. Manulis, N. Fleischhacker, F. Günther, F. Kiefer, and B. Poetterring, “Group signatures: Authentication with privacy,” *Bundesamt für Sicherheit in der Informationstechnik, Bonn, Germany, Tech. Rep*, 2012.