A Prover-Anonymous and Terrorist-Fraud Resistant Distance-Bounding Protocol*

Xavier Bultel
University Clermont Auvergne
xavier.bultel@udamail.fr

Sébastien Gambs
UQAM, Montréal
gambs.sebastien@uqam.ca

David Gérault
University Clermont Auvergne
david.gerault@udamail.fr

Pascal Lafourcade
University Clermont Auvergne
pascal.lafourcade@udamail.fr

Cristina Onete
INSA/IRISA Rennes
cristina.onete@gmail.com

Jean-Marc Robert
ÉTS, Montréal
jean-marc.robert@etsmtl.ca

ABSTRACT

Contactless communications have become omnipresent in our daily lives, from simple access cards to electronic passports. Such systems are particularly vulnerable to *relay attacks*, in which an adversary relays the messages from a prover to a verifier. Distance-bounding protocols were introduced to counter such attacks. Lately, there has been a very active research trend on improving the security of these protocols, but also on ensuring strong privacy properties with respect to active adversaries and malicious verifiers.

In particular, a difficult threat to address is the *terrorist fraud*, in which a far-away prover cooperates with a nearby accomplice to fool a verifier. The usual defence against this attack is to make it impossible for the accomplice to succeed unless the prover provides him with enough information to recover his secret key and impersonate him later on. However, the mere existence of a long-term secret key is problematic with respect to privacy.

In this paper, we propose a *novel approach* in which the prover who wants to help his accomplice to authenticate does not leak his secret key but a reusable session key along with a group signature on it. This allows the adversary to impersonate him even without knowing his signature key. Based on this approach, we give the first distance-bounding protocol, called SPADE, integrating anonymity, revocability and provable resistance to standard threat models.

1. INTRODUCTION

With the accelerating convergence of our digital identities on our ubiquitous *smartphones*, developing secure authentication protocols is more important than ever. As an example, a virtual wallet including various personal credentials can be used for everyday life applications such as public transport, logistics and contactless-payment systems. Another crucial notion is to protect the privacy of the users against external eavesdroppers and legitimate entities. The canonical application for this concept is the contactless pass used for accessing public transport systems. In this context, privacy is a fundamental property in order for users to trust the system deployed.

Authentication protocols are among the most fundamental cryptographic primitives of the digital world. They enable an entity, called a *verifier*, to check the legitimacy of users (called *provers*) before giving access to a resource. The provers are assumed to possess cryptographic devices storing their secret credentials. To be secure, an authentication protocol must guarantee two properties: *correctness* and *soundness*. Correctness requires that a legitimate prover is always authenticated, while soundness demands that all illegitimate ones should be rejected by the verifier. Authentication protocols are often prone to *relay attacks* [6, 12], in which an adversary relays to the verifier the responses of a legitimate prover. This attack bypasses standard countermeasures such as encryption or digital signatures. Indeed, since the adversary forwards legitimate messages between the verifier and a legitimate prover, the verifier ends up authenticating the adversary.

Distance bounding (DB) was introduced by Brands and Chaum [8] to thwart relay attacks by allowing the verifier to estimate an upper bound on the distance between him and the prover using several *time-critical* challenge-response rounds. Assuming that trust requires physical proximity, if a prover is *outside* the close vicinity of the verifier, he should be rejected. Thus, in DB protocols, verifiers are equipped with a clock, and they measure the time between sending a challenge and receiving the corresponding response from the prover. Once the different *Round Trip Times* (RTTs) for all challenge-response rounds are measured, the verifier compares these values to a pre-existing bound $t_{\max}$ and accepts the prover if and only if: (a) the responses are correct and (b) all RTT values are below the threshold $t_{\max}$.

To be secure, a DB protocol must resist at least to: (1) *Mafia fraud* (MF), (2) *Distance fraud* (DF) and (3) *Impersonation fraud* (IF). MF resistance requires that no illegitimate *Man-in-the-Middle* (MiM) adversary can authenticate to the verifier, even in the presence of a legitimate prover with whom he can interact. DF resistance demands that no legitimate but malicious prover, located outside the verifier's trusted vicinity, should be able to authenticate. A variant of

*This work was partially supported by the "Digital trust" Chair from the University of Auvergne Foundation, by NSERC Discovery and Accelerator Supplement grants, and by the European Union through the European Regional Development Fund.

WiSec'16, July 18–20, 2016, Darmstadt, Germany.

DOI: http://dx.doi.org/10.1145/2939918.2939919

this attack, in which a distant malicious prover uses an honest prover located in the verifier's vicinity to authenticate, is called *Distance Hijacking* (DH) [11]. Finally, the IF resistance addresses the simple situation in which the malicious adversary tries to fool the verifier without any help.

Another important threat against DB protocols is the *Terrorist Fraud* (TF), in which a malicious yet legitimate prover helps a cooperative MiM accomplice to authenticate. However, one of the assumptions is that the prover wants to retain control of his secret credentials. Thus, he is willing to help his accomplice, but without giving him a better chance to authenticate in latter attempts. Since this attack assumes that the prover *knows* his credentials, it could be prevented straightforwardly if a tamper-proof component is used to prevent the prover from learning this information. Unfortunately, relying only on tamperproofness is generally not sufficient [23]. In most payment systems relying on secure smartcards, back-end fraud detection mechanisms have been deployed to prevent any massive frauds. Thus, it is preferable to consider the most pessimistic scenario in which the adversaries are assumed to be given a *white-box* access to the secret keys. In this context, the usual countermeasure against TF is to force the prover to leak parts of his long-term key if he wants to give his accomplice a fair chance to succeed.

Since DB protocols were defined for RFID tags and readers, they use shared symmetric keys between provers and the verifier (*i.e.*, the prover is authenticated using his shared secret with the verifier). However, the seminal DB protocol of Brands and Chaum [8] was based on public-key cryptography. Improvements in RFID architectures as well as the emergence of NFC smartphones have motivated recent research in DB to consider public-key cryptography [16,19,27].

A recent concern in DB protocols is *privacy*. One of the first schemes to address this concept is the Swiss-Knife protocol [20]. However, its guarantee holds only if secret keys can never be leaked, and only with respect to an external eavesdropper but not against a legitimate verifier. Currently, no precise definition of this property is given and no formalized proof exists in the literature. Furthermore, as noted by Fischlin and Onete [14], the original protocol presents a flaw, since the same secret key is used both as input to pseudo-random function and to a function XORing it with one of the time-critical response strings.

Introducing privacy with respect to the verifier raises the question of the revocability of a prover by the registration authority. Hence, before the authentication succeeds, the verifier should check whether this prover has been revoked. Indeed, if this property is not taken into account, the corruption of a prover makes the whole system vulnerable, as there is no way to distinguish whether a prover uses stolen credentials or legitimate ones. In this paper, corrupting a user simply means that the adversary is able to obtain his secret keys. Building an anonymous protocol without any revocation mechanism is trivial: the same secret key is given to all provers who become indistinguishable. However in this case, one corruption forces the update of the keys of all users, which makes this solution impractical. Our aim is to fill this gap by proposing a provably TF-resistant, prover-anonymous, secure and revocable DB protocol.

A typical scenario for our secure and anonymous DB protocol can be described as follows. In a public transport system, users relying on their NFC-enabled phones may have access to buses or subway stations if they can properly authenticate. However, users must protect their identity with respect to legitimate verifiers trying to profile them. In such a context, a TF attack is simply a user ready to lend illegally his monthly pass to someone for a single trip while he is not using it. However, this user would not accept that his accomplice can impersonate him later at will to avoid being caught (if the same nonce N_p is used successfully numerous times). Thus, the presence of a backdoor in the verifiers can play in important role to deter such frauds. In an in-depth security approach, tamper-proof protection is not sufficient in this case. Indeed, it may protect the long term private key, but it would be useless to protect the two strings used in the time-critical phase implemented directly in the network access card for efficiency. The prover should answer the challenges as fast as possible, or otherwise the verifier can estimate that the prover is further than he really is. These strings are critical for the TF attacks and can therefore be easily obtained.

Another scenario could be the following one. Consider a research center, in which authorized personnel use experimental vehicles for both private and work-related purposes. In particular, they may use the vehicles *and* open the facility doors with a contactless (near-field) smartphone application. In this context, the following properties must be guaranteed. First, neither the vehicles, nor the buildings, must be accessible to someone who is not affiliated with the research center. Second, a vehicle or a building should only be unlocked by someone which is in its close proximity, as unlocking it remotely may leave the doors open to intruders. Next, the personnel should be held responsible if they unlock a facility door or a car that results in an unauthorized intrusion causing theft or damage. Finally, since the vehicles can also be used for personal purposes, their usage by someone should not be traceable ito respect his privacy. The protocol $SPADE$ fulfils all these requirements: namely MF and impersonation resistance, DF resistance, TF resistance and revocable anonymity.

Contributions. We propose SPADE (for *Secure Prover Anonymous Distance-bounding Exchange*), the first protocol to achieve prover-anonymity with respect to the strongest possible adversaries, provable TF resistance, and revocability of corrupted provers. A *wide-strong adversary* is the strongest that one can imagine. He can actively cheat during the protocol, corrupt entities and learn the result of the authentication.

For ensuring anonymity, our construction relies on the concept of group signatures [5], which enables a member of a group to sign anonymously on behalf of the group. New members can dynamically join the group or be revoked. This is managed by a central registration authority, which has to be involved in any signature verification. In case of dispute, a trusted authority can retrieve the identity of a signer.

In addition to privacy, our main contribution is to ensure TF resistance. Most TF-resistant DB protocols achieve this property by binding the responses of the time-critical phases to a long-term secret key. This forces the provers to reveal to their MiM accomplices some bits of their secret key to authenticate, thus allowing their accomplice to impersonate a prover in latter runs of the protocol. Our approach represents a radical change in the sense that it is based on a session key, chosen by a legitimate prover and signed with

his group signature key, before being encrypted. To prevent replay attacks, the responses to the time-critical phases depend on a verifier-specific nonce. However, given a value that is reasonably close to the prover's session key, the adversary can replay the prover's signature to be authenticated on his behalf. The presence of a *backdoor*, which can be used to retrieve the information needed to impersonate a prover, should deter any prover to help potential accomplices. This was originally suggested by Fischlin and Onete [15].

Related Work. In a recent survey, Brelurut, Gérault and Lafourcade reviewed 42 DB protocols (ranging from 1993 up to 2015) [9]. Only nine of them are not broken yet, and none of them achieves at the same time provable TF resistance, prover anonymity and revocability. Lately, Gambs, Onete and Robert [16] introduced the concept of privacy against honest-but-curious and malicious verifiers that aim at profiling legitimate provers. They proposed a construction relying on the protocol proposed in [19], but in which the provers are managed as a group. Although they addressed MF, DF and IF, they did not tackle TF resistance, which is not easily compatible with privacy.

Recently, Vaudenay [28] introduced a generic construction to transform a DB protocol based on a symmetric-key scheme into one based on a public-key cryptography. His construction yields provable MF, DF and DH resistance against the strongest possible adversaries, but not TF resistance. Moreover, privacy is only limited to a MiM adversary and not with respect to a malicious verifier.

Ahmadi and Safavi-Naini [1] have proposed a privacy-preserving DB protocol that is claimed to be TF-resistant. Their protocol fixes the vulnerabilities [3] of the DBPK-log protocol [10], and provides prover anonymity. A new prover joining the system picks a secret key and obtains its blind signature from the registration authority. This signature is then used as a membership certificate. During a session of the DB protocol, the prover never reveals his secret key nor his certificate. Instead, he uses a zero-knowledge proof of knowledge to prove that he knows a valid certificate, and that its value was used to generate his responses during the protocol execution. Unfortunately, the security proofs are not yet available. Moreover, the mechanism used to ensure anonymity does not allow user revocation, which is a major limitation of this solution.

Three frameworks for analyzing the security of DB protocols have been published: Avoine, Ali Bingöl, Kardaş, Lauradoux and Martin [2], Dürholz, Fischlin, Kasper and Onete [13], and Boureanu, Mitrokotsa and Vaudenay [7]. In our analysis, the DFKO framework [13] and the compatible game-based TF resistance [15] are used. Though this framework is for a single prover and a single verifier, it can be extended to multiple provers and to capture DH attacks. Finally, the prover anonymity notion defined in [16] is used, which is compatible with the DFKO framework.

Outline. In Section 2, we review the notions necessary to understand our work such as the adversary models for DB protocols as well as the cryptographic primitives used in our protocol. Afterwards in Section 3, we describe SPADE. In Section 4, we define the security models and prove these security properties of our protocol in Section 5. Then in Section 6, we give an analysis on how to set the parameters of the protocol depending on the security goals before concluding in Section 7.

2. PRELIMINARIES

In this section, we review the security models for DB protocols before describing the tools used in SPADE.

2.1 Distance-Bounding Models

In this section, we briefly review the notion of distance-bounding authentication, as well as the DFKO framework, which we use to prove the security of our protocol. DB protocols involve two parties: a *prover* P and a *verifier* V. In the public-key setting, let $\{\mathsf{sk}_P, \mathsf{pk}_P\}$ denote the private and public keys of P and let $\{\mathsf{sk}_V, \mathsf{pk}_V\}$ represent the ones of V. All public keys are known by all parties. During the DB protocol, P and V interact with each other, yielding the bit $OutV = 0$ for *reject* or 1 for *accept* at the end of the verifier algorithm.

Adversary Model. In the DFKO model [13], an adversary $\mathcal{A}$ interacts with a prover and the verifier in three types of sessions: (1) *prover-verifier* sessions, in which $\mathcal{A}$ eavesdrops on an honest execution between a prover and a verifier, (2) *prover-adversary* sessions, in which $\mathcal{A}$ impersonates a verifier interacting with the prover, and (3) *adversary-verifier* sessions in which $\mathcal{A}$ impersonates a prover interacting with the verifier. Protocols are sequences of exchanges between a prover and the verifier. A message from the verifier to the prover and the subsequent response back is called a *round*. Successive rounds may be combined in *phases*. If the clock is used to measure the time elapsed from the beginning of the phase to its end, this phase is called *time-critical* while otherwise it is called *lazy*. Finally, sessions run by honest parties are associated with identifiers sid.

Adversaries are quantified in terms of their *computational resources*, the number of prover-verifier sessions q_{obs} eavesdropped, the number q_{v} of verifier-adversary and the number q_{p} of prover-adversary sessions initiated, as well as the adversary's winning advantage $\epsilon(n)$. In this paper, we consider the strongest possible adversaries against privacy, which are assumed to know the final result of an authentication session (*i.e.*, accept or reject), and to be able to corrupt provers (*i.e.*, get their keys) without any restriction. Such adversaries are called *wide-strong* in the literature [29].

Oracles. Adversaries are also classified depending on the oracles they may use. These oracles perform functionalities without giving any further details on their internal information unknown to their users.

Prover Anonymity. The PA concept due to Gambs, Onete and Robert [16] extends existing privacy models used in DB protocols [18, 19]. In this case, an adversary against the prover anonymity can be either a MiM adversary, an *honest-but-curious* or even malicious verifier. His objective is to link sessions involving a given prover. This game is characterized by a hidden bit b, and the goal of the adversary is to guess this bit.

The *PA game* is defined by an adversary $\mathcal{A}$ interacting with the provers through an oracle, which blinds their identity. The adversary $\mathcal{A}$ *wins* the game if he can identify with a non-negligible advantage which prover was selected by a challenger and hidden by the oracle. The adversary's advantage is simply given by the difference between his probability of guessing the identity of the selected prover among the set of potential provers and the trivial guessing probability of one over the cardinality of that set. A protocol is *PA-resistant* if there is no such a winning adversary.

Mafia-Fraud Resistance. MF attacks are defined with respect to an active MiM adversary $\mathcal{A}$, which can interact with a prover and the verifier in several sessions. His goal is to authenticate to the verifier, but without relaying information directly from the prover (since the verifier should detect relays). The DFKO framework rules out *pure relaying* attacks, which consist in an adversary forwarding the messages between two legitimate parties. A prover-adversary session sid' used concurrently with an adversary-verifier session sid to perform pure relays is called a *tainted* session.

An adversary $\mathcal{A}$ *wins* the MF game if there is at least one adversary-verifier session that is untainted by any other prover-adversary session, and in which the verifier accepts with a non-negligible probability the adversary as legitimate. A protocol is *MF-resistant* if there is no such adversary.

Terrorist-Fraud Resistance. In their extension of the DFKO framework, Fischlin and Onete [15] consider many variations of TF attacks. We use their game-based TF-resistance notion GameTF. Intuitively, the goal of the MiM adversary $\mathcal{A}$ colluding with a malicious prover is to be authenticated by the verifier. However, the prover also wants to control his accomplice's access in future sessions, in particular, by keeping his secret key as confidential as possible. This property is formalized as a two-phase game. First, the TF adversary $\mathcal{A}$ attempts to authenticate to the verifier with non-negligible probability. In the second one, an adversary $\mathcal{B}$ takes as input the $\mathcal{A}$'s complete view and runs a MF attack. If $\mathcal{B}$ authenticates with higher probability than in a regular MF attack, $\mathcal{A}$ is said to be *helpful* to $\mathcal{B}$.

An adversary $\mathcal{A}$ *wins* the TF game if he authenticates with a non-negligible probability, but is not helpful to anyone. A protocol is *TF-resistant* if there is no such adversary.

Intuitively, a protocol is TF-resistant if a malicious prover *cannot* help his accomplice authenticate with a non-negligible probability without losing control on his credentials. Thus, a *rational* prover will avoid to perform such an attack.

Impersonation-Fraud Resistance. IF resistance mainly concerns the lazy phases of the protocol. Here, the objective of the adversary $\mathcal{A}$ is to make the verifier accept his authentication in a session in which he does not relay the lazy phases from a honest session. $\mathcal{A}$ *wins* the IF game if there is at least one adversary-verifier session in which the verifier accepts with a non-negligible probability the adversary as legitimate, and such that no prover-verifier session shares the same lazy transcript. A protocol is *IF-resistant* if there is no such adversary.

Distance-Fraud Resistance. In the case of DF attacks, the adversary $\mathcal{A}$ is a malicious prover outside the proximity of the verifier. Since this adversary is not be able to beat the verifier's clock, an adversary-verifier session is defined as *tainted* if for any time-critical phase the adversary is unable to *commit* to that round's response *before* receiving the verifier's message for that phase. In this attack, the adversary usually sends the response before receiving the challenge. We omit from the adversary's quantification the number of sessions he creates with the prover.

An adversary $\mathcal{A}$ *wins* the DF game if there is at least one adversary-verifier session that is untainted by any other prover-adversary session, and in which the verifier accepts with a non-negligible probability the adversary as legitimate. A protocol is *DF-resistant* if there is no such adversary.

2.2 Cryptographic Primitives

First, we recall the definition of a public-key encryption scheme.

Definition 1 (Public Key Encryption). *A public key encryption scheme* PKE *is defined by:*

$\mathsf{E.gen}(1^\lambda)$ *returns a public and private key pair* $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{E.enc}_{\mathsf{pk}}(m)$ *returns the ciphertext* c.

$\mathsf{E.dec}_{\mathsf{sk}}(c)$ *returns* m *such that* $\mathsf{E.dec}_{\mathsf{sk}}(\mathsf{E.enc}_{\mathsf{pk}}(m)) = m$.

The related security game is defined as follows. An adversary receives a public key pk from the key pair $(\mathsf{pk}, \mathsf{sk})$ and has access to a decryption oracle. He sends two messages (m_0, m_1) to a challenger that computes the ciphertext $c = \mathsf{E.enc}_{\mathsf{pk}}(m_b)$ for a random bit b. The adversary wins if he correctly guesses b. A PKE is $\mathsf{IND\text{–}CCA2}$ secure [24], if there is no polynomial-time winning adversary.

Group Signature. In a group signature scheme [5,25], each member of the group has a personal signing key that he uses to sign a message on behalf of the group. An entity, called a *group manager*, adds new members to the group while another entity called the *opening authority* can open a signature to reveal the identity of the signer. Some schemes allow also dynamic group structures with revocation capabilities by the group managers.

Definition 2 (Revocable Group Signature). *A revocable group signature scheme* $\mathsf{G\text{–}SIG}$ *is defined by:*

$\mathsf{G.gen}(1^\lambda)$ *returns a group/master key pair* $(\mathsf{gpk}, \mathsf{msk})$ *and sets the user list* UL *and the revoked user list* RL.

$\mathsf{G.join}_{\mathsf{msk}}(i, \mathsf{gpk}, \mathsf{UL})$ *is a protocol between a user* U_i *(using* gpk*) and a group manager GM (using* gpk *and* msk*).* U_i *interacts with GM to get his signing key* ssk_i*, while GM outputs a value* reg_i *and adds* U_i *to* UL.

$\mathsf{G.rev}_{\mathsf{msk}}(i, \mathsf{RL}, \mathsf{UL}, \mathsf{gpk})$ *computes the revocation logs* rev_i *for user* U_i*, using* $\mathsf{reg}_i, \mathsf{gpk}$ *and* msk *and moves* U_i *to* RL.

$\mathsf{G.sig}_{\mathsf{ssk}_i}(m)$ *returns a group signature* σ.

$\mathsf{G.ver}_{\mathsf{gpk}}(\sigma, m, \mathsf{RL})$ *outputs* 1 *if and only if* σ *is valid for the message* m *and the key* ssk_i *of a non-revoked user.*

$\mathsf{G.ope}_{\mathsf{msk}}(\sigma, m, \mathsf{UL}, \mathsf{gpk})$ *outputs a user identity* U_i.

The essential security property required for a group signature is *unlinkability.* It captures the idea that no polynomial-time adversary should be able to distinguish with a non-negligible probability whether two signatures have been issued by the same signer or not. The adversary is assumed to have access to oracles that add new users (honest or corrupted), corrupt or revoke a user, sign with an user's signing key, and open a signature using the opening authority key (the oracle trivially rejects the challenge signature).

The second security property is *traceability.* It ensures that no polynomial-time adversary should be able to produce a valid signature for a revoked user or a honest user with non-negligible probability (using the same oracles as for the unlinkability experiment). An adversary breaks the traceability of a signature scheme if he is able to forge a valid signature on a message of his choice. For digital signature schemes, this property is known as $\mathsf{EUF\text{–}CMA}$ secure [17].

The last security requirement is *non-frameability*, which guarantees that no adversary is able to sign on behalf of

a honest user even if he knows the key of the group manager. This property is rather strong since it protects the user against a corrupted group manager or opening authority.

Pseudorandom Function. A set PRF is a collection of polynomial-time pseudo-random functions $\{PRF_k\}_{k\in K}$ defined on a key set K, which is such that, for any polynomial-time adversary, the probability of distinguishing between outputs of PRF_k for a random key k and outputs of a truly random function is negligible.

In the random oracle model, defining $PRF_k(x) = H(k, x)$ is a simple way to construct PRF using a given cryptographic hash function H.

3. SPADE

We first describe the functionalities provided by an anonymous distance-bounding protocol, before detailing our proposition SPADE for such a protocol.

DEFINITION 3 (ANONYMOUS DB). *An* anonymous distance-bounding *protocol* DB *is defined by:*

$\mathsf{DB.gen}(\lambda)$ *sets a master key* MK *and a verification key* VK*, and sets the user list* UL *and the revoked-user list* RL.

$\mathsf{DB.join}_{\mathsf{MK}}(i, \mathsf{UL})$ *returns a prover secret key* psk_i *for* P_i*. This algorithm also outputs a value* reg_i *and adds* P_i *to* UL.

$\mathsf{DB.auth}(\mathsf{psk}_i, \mathsf{VK}, \mathsf{RL})$ *is an interactive authentication protocol between* P_i *(using* psk_i*)and* V *(using* VK *and* RL*). V returns* 1 *in case of success and* 0 *otherwise. This algorithm also outputs a transcript* trans.

$\mathsf{DB.revoke}_{\mathsf{MK}}(i, \mathsf{RL}, \mathsf{UL})$ *computes the revocation logs* rev_i *for* P_i*, using* reg_i *and* MK*, and moves* P_i *from* UL *to* RL.

$\mathsf{DB.open}_{\mathsf{MK}}(\mathsf{trans})$ *outputs the identity of prover* P_i.

Consider the scenario in which we have a group manager GM, a verifier V and a group of provers P_i, which can authenticate to V. During the initialization, GM uses $\mathsf{DB.gen}(\lambda)$ to produce a master key MK and a verifier key VK for V. Calling $\mathsf{DB.join}_{\mathsf{MK}}(i, \mathsf{UL})$, M also generates a secret key psk_i for each prover P_i. Using these keys, P_i runs $\mathsf{DB.auth}(\mathsf{psk}_i, \mathsf{VK}, \mathsf{RL})$ to authenticate himself to V. Afterwards, GM can add new provers using $\mathsf{DB.join}$ and revoke a prover using $\mathsf{DB.revoke}$. Finally, GM can lift the anonymity of a user by running the algorithm $\mathsf{DB.open}_{\mathsf{MK}}(\mathsf{trans})$"

The main idea behind SPADE is that the prover is authenticated anonymously as a member of an authorized group, ensuring anonymity due to the group signature scheme.

DEFINITION 4 (SPADE). *Let* $E = (\mathsf{E.gen}, \mathsf{E.enc}, \mathsf{E.dec})$ *be a PKE scheme,* $\mathsf{G} = (\mathsf{G.gen}, \mathsf{G.sig}, \mathsf{G.ver}, \mathsf{G.join}, \mathsf{G.rev}, \mathsf{G.ope})$ *be a* G–SIG *scheme and PRF be a pseudorandom-function set. The DB protocol* $(E, G, PRF)-SPADE$ *is defined by:*

$\mathsf{DB.gen}(\lambda)$ *sets the verifier keys* $(\mathsf{pk}_V, \mathsf{sk}_V) = \mathsf{E.gen}(\lambda)$ *and the signature key pair* $(\mathsf{gpk}, \mathsf{msk}) = \mathsf{G.gen}(\lambda)$*. It also returns the master key* $\mathsf{MK} = (\mathsf{msk}, \mathsf{gpk}, \mathsf{pk}_V, \mathsf{sk}_V)$ *and a verification key* $\mathsf{VK} = (\mathsf{sk}_V, \mathsf{gpk})$*, and sets the user list* UL *and the revoked-user list* RL.

$\mathsf{DB.join}_{\mathsf{MK}}(i, \mathsf{UL})$ *runs the algorithm* $\mathsf{G.join}_{\mathsf{msk}}(i, \mathsf{gpk}, \mathsf{UL})$ *to get* ssk_i *and then constructs* $\mathsf{psk}_i = (\mathsf{pk}_V, \mathsf{ssk}_i)$ *for the prover* i*. This algorithm also returns a value* reg_i *and adds* P_i *to* UL.

$\mathsf{DB.auth}(\mathsf{psk}_i, \mathsf{VK}, \mathsf{RL})$ *is described in Figure 1. The security parameter* n*, defining the number of rounds, is function of the security parameter* λ.

$\mathsf{DB.revoke}_{\mathsf{MK}}(i, \mathsf{RL}, \mathsf{UL})$ *runs* $\mathsf{G.rev}_{\mathsf{msk}}(i, \mathsf{RL}, \mathsf{UL}, \mathsf{gpk})$.

$\mathsf{DB.open}_{\mathsf{MK}}(\mathsf{trans})$ *computes* $(N_P, \sigma) = \mathsf{E.dec}_{\mathsf{sk}_V}(e)$*, in which* e *is the first message of the transcript. Afterwards, it outputs the prover* $P_i = \mathsf{G.ope}_{\mathsf{msk}}(\sigma, N_P, \mathsf{UL}, \mathsf{gpk})$.

We now detail our protocol presented in Figure 1.

Initialization Phase. First, P generates a random n-bit string N_P and signs it $\sigma = \mathsf{G.sig}_{\mathsf{ssk}_P}(N_P)$. Then, he encrypts both values $e = \mathsf{E.enc}_{\mathsf{pk}_V}(N_P, \sigma)$, with the public key of V and sends the result to V. The verifier retrieves $(N_P, \sigma) = \mathsf{E.dec}_{\mathsf{sk}_V}(e)$ and checks the signature with $\mathsf{G.ver}_{\mathsf{gpk}}(\sigma, N_P, \mathsf{RL})$. If it is invalid, he aborts the protocol. Otherwise, V returns two new random n-bit strings m and N_V to P. Finally, both P and V compute $a = PRF_{N_P}(N_V)$.

Distance-bounding Phase. P and V perform n challenge-response rounds during a time-critical phase. This is the heart of the protocol as it is used to determine if the prover is in the verifier's vicinity. At round i, V sends a bit c_i and P answers with $a_i \oplus ((N_{Pi} \oplus m_i) \wedge c_i)$. The string m prevents a malicious prover from picking $N_P = 0^n$, which would allow him to respond with a_i even before receiving the challenges. At the end of each round, V stores the RTT denoted by Δt_i.

Verification Phase. P concatenates all the challenges C and the responses R, computes $\tau = PRF_{N_P}(N_V, m, C, R)$ and sends it to V. V checks $\tau \stackrel{?}{=} PRF_{N_P}(N_V, m, C, R)$ and verifies that the Δt_i are coherent with respect to the proximity threshold to ensure that P is within an authorized distance. If all these checks succeed and all the responses are correct, V returns $OutV = 1$ for acceptance, while otherwise he returns $OutV = 0$.

Novel Approach. In contrast to most protocols in the literature, our DB protocol does not rely on a long-term shared secret between a prover and the verifier, but on a session key N_P exchanged anonymously. Long-term shared secrets constitute a serious burden to overcome to provide anonymity for the prover as these secrets can be easily used to link different sessions of a user. The radical shift that we propose can be seen as the main contribution of this paper.

SPADE is built in such a way that an adversary can replay a session key if he gets access to it (*e.g.*, during a TF). To ensure that provers protect their session keys, we introduce a *stateless backdoor* in the verifier, allowing an adversary to recover the complete session key N_P provided that he knows enough bits about it. This sets a trade-off between the malicious prover and any potential accomplice. Indeed, providing too much information to an accomplice, he may eventually impersonate the prover, which is not desirable. At the other end of the spectrum, by not giving him enough information, he may not be helpful to the prover.

The backdoor is presented in Figure 2 and its analysis is given in Section 6. If V receives the bit $b = 1$, he gets afterwards e and N'_P. Then, he extracts $(N_P, \sigma) = \mathsf{E.dec}_{\mathsf{sk}_V}(e)$ and uses $\mathsf{G.ver}_{\mathsf{gpk}}(\sigma, N_P, \mathsf{RL})$ to verify the signature. If σ is valid, he returns N_P provided that the Hamming distance $d_H(N_P, N'_P)$ is smaller than a threshold t. Otherwise, he returns 0. Since a regular MiM adversary learns nothing about N_P, his probability to recover it using the backdoor

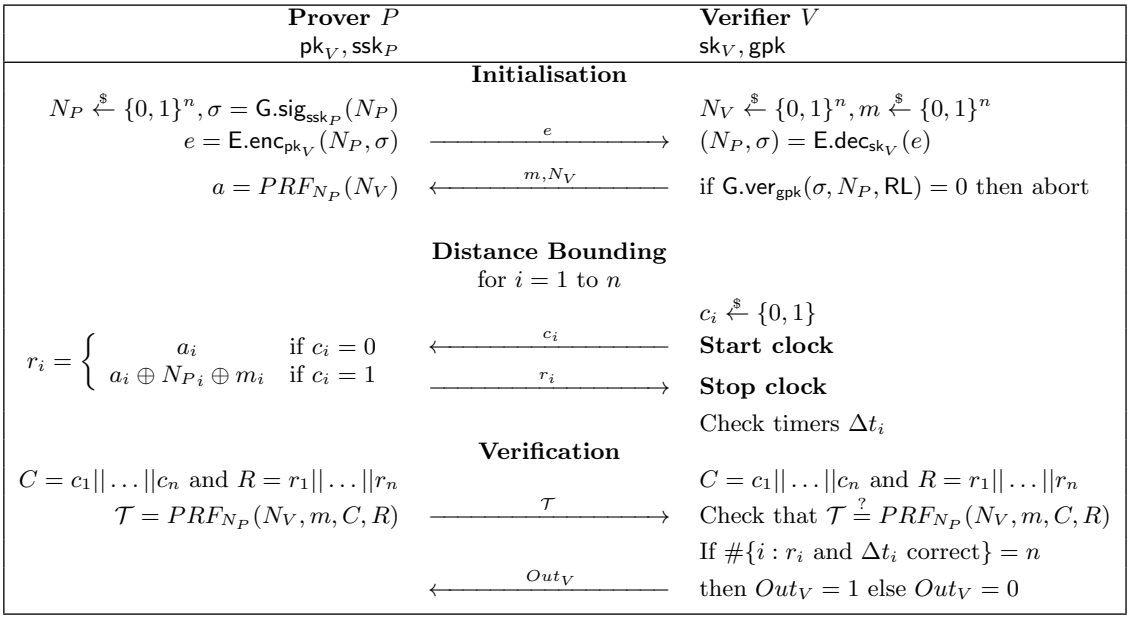

Figure 1: Anonymous TF resistant protocol from a public key encryption E , a pseudo-random-function set PRF and a group signature G, where $a||b$ is the concatenation of a and b, and $x \oplus y$ denotes the exclusive-or.

is negligible. However, the accomplice authenticating with the help of the prover should learn enough bits of N_P to use the backdoor. This new approach makes SPADE the first secure provable revocable and anonymous DB protocol.

4. SECURITY DEFINITIONS

In this section, we define the security properties of anonymous DB protocols as *games* between powerful adversaries and benign *challengers*. For more details,refer to [13, 16].

Initialization. The challenger uses $\mathsf{DB.gen}(n)$ to build the simulation environment $(\mathsf{MK}, \mathsf{VK}, \mathsf{UL}, \mathsf{RL})$ and sets four lists: two transcript lists $\mathsf{TL_v}$ for the verifier, $\mathsf{TL_p}$ for the prover P, a corrupted users list CU and a list HL representing the random oracle calls.

Oracles. The challenger can simulate these oracles:

- $\overline{\mathsf{DB}}.\mathsf{Join}^h(i)$ adds P_i by using $\mathsf{DB.join}_{\mathsf{MK}}(i, \mathsf{UL})$.
- $\overline{\mathsf{DB}}.\mathsf{Join}^c(i)$ adds a corrupted P_i using $\mathsf{DB.join}_{\mathsf{MK}}(i, \mathsf{UL})$. It also returns the secret key psk_i, and adds P_i to CU.
- $\overline{\mathsf{DB}}.\mathsf{Revoke}(i)$ runs $\mathsf{DB.revoke}_{\mathsf{MK}}(i, \mathsf{RL}, \mathsf{UL})$ on P_i.
- $\overline{\mathsf{DB}}.\mathsf{Corrupt}(i)$ simulates the corruption of P_i by returning his secret key psk_i and adding P_i to CU.
- $\overline{\mathsf{DB}}.\mathsf{Prover}(i)$ simulates a session by the honest prover P_i and adds the generated transcript to $\mathsf{TL_p}$.
- $\overline{\mathsf{DB}}.\mathsf{Verifier}(d)$ simulates a session by a honest verifier V at a distance d by delaying messages appropriately. It then appends $tainted = 0$ at the end of the transcript and adds it to the list $\mathsf{TL_v}$.
- $\overline{\mathsf{DB}}.\mathsf{Session}(i)$ simulates a session between a honest verifier V and a nearby honest P_i.
- $\overline{\mathsf{DB}}.\mathsf{Taint}(.)$ simulates an altered $\overline{\mathsf{DB}}.\mathsf{Verifier}(d)$ in which the time delay checks are bypassed. It then appends $tainted = 1$ to the transcript before adding it to $\mathsf{TL_v}$.
- $\mathsf{H}(\cdot)$ is a random oracle using a list HL. When receiving an input i such that $i \notin \mathsf{HL}$, it draws a binary string r uniformly at random, adds an entry (i, r) in HL and returns r. If $i \in \mathsf{HL}$, it returns the corresponding r.

4.1 Prover Anonymity

Let the anonymity experiment $\mathsf{Exp}^{\mathsf{PA}}_{\mathcal{A},\mathsf{DB}}(\lambda)$ for an adversary $\mathcal{A}$ on a protocol DB be defined as follows.

Definition 5 (Prover Anonymity Security). *$\mathcal{A}$ has access to the DB-oracles $\mathsf{Join}^h(i)$, $\mathsf{Join}^c(i)$, $\mathsf{Revoke}(i)$, $\mathsf{Corrupt}(i)$, $\mathsf{Prover}(i)$, $\mathsf{Verifier}(d)$, $\mathsf{H}(\cdot)$ and $\mathsf{Session}(i)$. First, $\mathcal{A}$ outputs (i_0, i_1). If i_0 or $i_1 \in \mathsf{CU}$, the challenger aborts the experiment. Otherwise, he picks $b \xleftarrow{\$} \{0,1\}$. Then, $\mathcal{A}$ loses access to $\mathsf{Corrupt}(i)$ and $\mathsf{Revoke}(i)$ on identities i_0 and i_1 (the oracles return false if $\mathcal{A}$ tries). Finally, $\mathcal{A}$ has access to the DB-oracle $\mathsf{Prover}_b(\cdot)$, which runs the DB protocol as the prover i_b using key psk_{i_b} interacting with $\mathcal{A}$.*

$\mathcal{A}$ wins if and only if adv outputs b.

We define $\mathcal{A}$'s advantage on this experiment as $\mathsf{Adv}^{\mathsf{PA}}_{\mathcal{A},\mathsf{DB}}(\lambda) = |\mathsf{Pr}[\mathsf{Exp}^{\mathsf{PA}}_{\mathcal{A},\mathsf{DB}}(\lambda) = 1] - \frac{1}{2}|$ and the advantage on the PA experiment as $\mathsf{Adv}^{\mathsf{PA}}_{\mathsf{DB}}(\lambda) = \max_{\mathcal{A} \in \mathsf{Poly}(\lambda)}\{\mathsf{Adv}^{\mathsf{PA}}_{\mathcal{A},\mathsf{DB}}(\lambda)\}$. DB is PA-resistant *if $\mathsf{Adv}^{\mathsf{PA}}_{\mathsf{DB}}(\lambda)$ is negligible*[1].

In this game, the adversary has access to all the verifier-accessible information and he may interact with one of two provers chosen adversarially (the choice depends on a secret bit b). The adversary wins if he can identify the secret bit, which implies that he can distinguish transcripts produced by a given prover, thus linking the sessions of this prover.

4.2 Mafia Fraud

Let the mafia fraud experiment $\mathsf{Exp}^{\mathsf{MF}}_{\mathcal{A},\mathsf{DB}}(\lambda)$ for an adversary $\mathcal{A}$ on a protocol DB be defined as follows.

[1] $\mathsf{Poly}(\lambda)$ is the set of algorithms running polynomially in λ.

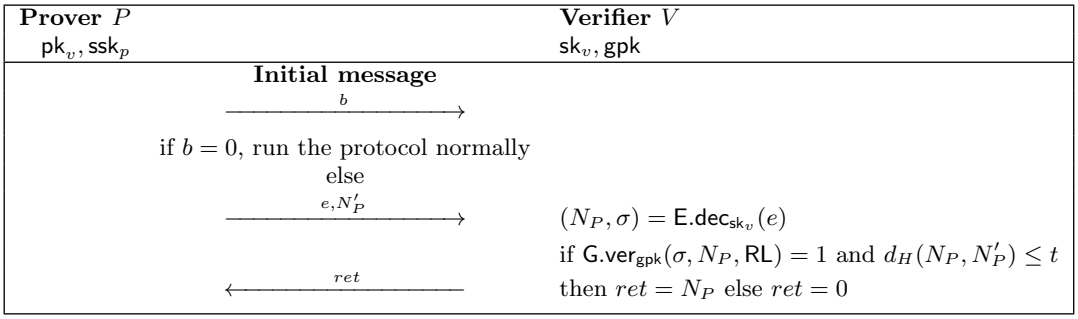

Figure 2: The backdoor mechanism. If the initial message is $b = 0$, the protocol is run normally. Otherwise, the verifier simply waits to receive a value e that he parses as (N_P, σ) and a string N'_P. If N_P and N'_P are close enough, he returns N_P.

DEFINITION 6 (MAFIA FRAUD SECURITY). *$\mathcal{A}$ has access to the* DB*-oracles* $\mathsf{Join}^h(i)$, $\mathsf{Join}^c(i)$, $\mathsf{Revoke}(i)$, $\mathsf{Corrupt}(i)$, $\mathsf{Prover}(i)$, $\mathsf{Verifier}(d)$, $\mathsf{Session}(i)$, $\mathsf{H}(\cdot)$ *and* $\mathsf{Taint}(.)$.

$\mathcal{A}$ wins if and only if $\exists$ $\mathsf{trans} \in \mathsf{TL_v}$ *such that* trans *is the concatenation of all the messages exchanged during a* DB *session and the following conditions are satisfied:*

- $OutV = 1$, $tainted = 0$ *is at the end of* trans, *and*
- $\mathsf{DB.open}_{\mathsf{MK}}(\mathsf{trans}) \notin \mathsf{CU}$.

We define $\mathcal{A}$'s advantage on this experiment as $\mathsf{Adv}^{MF}_{\mathcal{A},\mathsf{DB}}(\lambda) = \mathsf{Pr}[\mathsf{Exp}^{MF}_{\mathcal{A},\mathsf{DB}}(\lambda) = 1]$ *and the advantage on the MF experiment as* $\mathsf{Adv}^{MF}_{\mathsf{DB}}(\lambda) = \max_{\mathcal{A}\in\mathsf{Poly}(\lambda)}\{\mathsf{Adv}^{MF}_{\mathcal{A},\mathsf{DB}}(\lambda)\}$. DB *is* MF-resistant *if* $\mathsf{Adv}^{MF}_{\mathsf{DB}}(\lambda)$ *is negligible.*

In this game, the adversary is able to interact with provers and the verifier and wins if he authenticates to the verifier (in the presence of a prover) if, and only if, the attacker has not purely relayed information between the two legitimate parties.

4.3 Terrorist Fraud

A TF involves a legitimate but malicious prover and his accomplice interacting with the verifier.

First, we recall the definition of terrorist-fraud resistance given by Fischlin and Onete [15]. It is based on GameTF in which the accomplice $(q_{\mathsf{obs}}, q_{\mathsf{v}})$-adversary $\mathcal{A}$ can eavesdrop q_{obs} honest prover-verifier sessions. He can also initiate q_{v} adversary-verifier sessions (and the matching adversary-prover sessions), in which he acts as a MiM adversary, forwarding information in the lazy phases. Then, $\mathcal{A}$ may act as the prover P in the time-critical phase. In all these sessions, the cautious P and V should select new values N_P and N_V for each session. Naturally, q_{obs} and q_{v} depend on the security parameter n. During these sessions, $\mathcal{A}$ can interact with P and V as specified in the MF game. In GameTF, the notion of a *tainted* session is defined as:

DEFINITION 7 (TAINTED SESSION - TF). *A TF adversary $\mathcal{A}$ taints an adversary-verifier session* sid *if there is a prover-adversary session* sid*' such that the following events occur: (i) $\mathcal{A}$ receives a message c from the verifier in* sid*, (ii) $\mathcal{A}$ sends a message c' in session* sid*' to the prover such that $c' > c$[2], and gets a response r such that $r > c'$, and (iii) $\mathcal{A}$ forwards a message r' in* sid *such that $r' > r$.*

[2] *i.e.*, c' can be sent only after c has been received.

Thus, relaying scheduling messages is ruled out, even if they are different. This is stronger than the pure relaying definition, which requires than the same messages are forwarded.

Let $\mathsf{view}_{\mathcal{A}}$ be the internal state of $\mathcal{A}$ colluding with P.

DEFINITION 8 (HELPFUL ADVERSARY). *A TF adversary $\mathcal{A}$ against a DB protocol* DB *is said to be* helpful *to an adversary $\mathcal{B}$ in the game Π if, given $\mathsf{view}_{\mathcal{A}}$, $\mathcal{B}$ wins the game Π with a probability greater than $\mathsf{Adv}^{\Pi}_{\mathsf{DB}}(\lambda)$, his original advantage without the help of his accomplice $\mathcal{A}$.*

In the *SPADE* context, $\mathcal{B}$ may either want to play a MF attack or an IF attack. In both cases, a non-negligible help can be amplify to win these two games with probability one.

DEFINITION 9 (GAMETF-SECURITY). *$\mathcal{A}$ has access to the* DB*-oracles* $\mathsf{Join}^h(i)$, $\mathsf{Join}^c(i)$, $\mathsf{Revoke}(i)$, $\mathsf{Corrupt}(i)$, $\mathsf{Prover}(i)$, $\mathsf{Verifier}(d)$, $\mathsf{Session}(i)$ *and* $\mathsf{Taint}(.)$. *A DB protocol* DB *is $p_{\mathcal{A}} -$* GameTF*-secure if for any (q_{obs}, q_V) adversary $\mathcal{A}$ winning with probability $p_{\mathcal{A}}$, at least one of the two following conditions holds:*

- *$p_{\mathcal{A}}$ is negligible with respect to n,*
- *there exists a MF adversary $\mathcal{B}$ running $O(q_{\mathsf{obs}})$ prover-verifier sessions and $O(q_V)$ adversary-verifier sessions to which $\mathcal{A}$ is helpful.*

4.4 Impersonation Fraud

Let the IF experiment $\mathsf{Exp}^{\mathsf{IF}}_{\mathcal{A},\mathsf{DB}}(\lambda)$ for an adversary $\mathcal{A}$ on a protocol DB be defined as follows.

DEFINITION 10 (IMPERSONATION FRAUD SECURITY). *$\mathcal{A}$ has access to the* DB*-oracles* $\mathsf{Join}^h(i)$, $\mathsf{Join}^c(i)$, $\mathsf{Revoke}(i)$, $\mathsf{Corrupt}(i)$, $\mathsf{Prover}(i)$, $\mathsf{Verifier}(d)$, $\mathsf{Session}(i)$, $\mathsf{H}(\cdot)$ *and* $\mathsf{Taint}(.)$. *$\mathcal{A}$ wins if and only if $\exists$* $\mathsf{trans} \in \mathsf{TL_v}$ *such that* trans *is the concatenation of all the messages exchanged during a* DB *session, and the following conditions are satisfied:*

- $OutV = 1$,
- $\nexists t \in \mathsf{TL_p}$ *such that the lazy phases of t and* trans *are equal,*
- *and* $\mathsf{DB.open}_{\mathsf{MK}}(\mathsf{trans}) \notin \mathsf{CU}$.

Define $\mathsf{Adv}^{\mathsf{IF}}_{\mathsf{DB}}(\lambda)$ *as in Definition 6.* DB *is* IF-resistant *if* $\mathsf{Adv}^{\mathsf{IF}}_{\mathsf{DB}}(\lambda)$ *is negligible.*

4.5 Distance Fraud

Let the DF experiment $\mathsf{Exp}^{\mathsf{DF}}_{\mathcal{A},\mathsf{DB}}(\lambda)$ for a distant adversary $\mathcal{A}$ on a protocol DB be defined as follows.

DEFINITION 11 (DISTANCE FRAUD SECURITY). *$\mathcal{A}$ has access to the* DB*-oracles* $\mathsf{Join}^c(i)$ *and* $\mathsf{Verifier}(d)$.

$\mathcal{A}$ wins if and only if $\exists\ \mathsf{trans} \in \mathsf{TL_v}$ *such that* trans *is the concatenation of all the messages exchanged during a* DB *session, and the following conditions are satisfied:*

- $OutV = 1$ and $d > d_{\mathsf{max}}$ *(maximum distance allowed).*

Define $\mathsf{Adv}^{DF}_{\mathsf{DB}}(\lambda)$ *as in Definition 6.* DB *is* DF-resistant *if* $\mathsf{Adv}^{DF}_{\mathsf{DB}}(\lambda)$ *is negligible.*

In this game, the adversary is a malicious prover, who can interact with the verifier in an arbitrary manner. He wins this game if and only if he is able to send all time-critical responses before receiving the respective challenges, for each time-critical round.

5. SECURITY RESULTS

Now that the models have been presented, we are ready to prove the main results of our paper.

In some of the theorems, the backdoor can be used to realize the attacks (*e.g.*, a MF or an IF attack). In this case, let $p_{back}(n,t)$ denote the probability that a n-bit secret can be recovered through a polynomial number of queries to the backdoor with a threshold t. Theorem 6.1 gives the value of $p_{back}(n,t)$, for $t = \alpha n$, for some constant $\alpha > 0$.

5.1 Prover Anonymity

First, let us recall the security game for the anonymity of a revocable group signature. This generalizes the anonymity game described by Gambs, Onete and Robert [16].

Let the anonymity experiment $\mathsf{Exp}^{\mathsf{Anon}}_{\mathcal{A},\mathsf{G-SIG}}(\lambda)$ for $\mathcal{A}$ on a revocable group signature G–SIG be defined as follows.

DEFINITION 12. ***First phase:*** *The challenger creates* $(\mathsf{UL}, \mathsf{RL}, \mathsf{msk}, \mathsf{gpk})$ *using* $\mathsf{G.gen}(1^\lambda)$*, gives* gpk *to $\mathcal{A}$, and sets the lists* CU *and* Σ*. During this phase $\mathcal{A}$ has access to* G*-oracles:*

$\overline{\mathsf{G}}.\mathsf{Join}^h(i)$ *creates P_i using* $\mathsf{G.join}_{\mathsf{msk}}(i, \mathsf{gpk}, \mathsf{UL})$.

$\overline{\mathsf{G}}.\mathsf{Join}^c(i)$ *creates P_i using* $\mathsf{G.join}_{\mathsf{msk}}(i, \mathsf{gpk}, \mathsf{UL})$ *with $\mathcal{A}$ and adds him to* CU.

$\overline{\mathsf{G}}.\mathsf{Revoke}(i)$ *revokes P_i using* $\mathsf{G.rev}_{\mathsf{msk}}(i, \mathsf{RL}, \mathsf{UL}, \mathsf{gpk})$.

$\overline{\mathsf{G}}.\mathsf{Corrupt}(i)$ *returns the secret key of P_i. If $P_i \in$* UL*, it sends* ssk_i *to $\mathcal{A}$ and adds P_i to* CU.

$\overline{\mathsf{G}}.\mathsf{Sign}(i, m)$ *returns a signature σ on behalf of P_i, using* $\mathsf{G.sig}_{\mathsf{ssk}_i}(m)$ *and adds the pair (m, σ) to Σ.*

$\overline{\mathsf{G}}.\mathsf{Open}(\sigma, m)$ *opens a signature σ on m and returns P_i to $\mathcal{A}$, using* $\mathsf{G.ope}_{\mathsf{msk}}(\sigma, m, \mathsf{UL}, \mathsf{gpk})$.

Challenge: *$\mathcal{A}$ selects (i_0, i_1). If i_0 and $i_1 \in$* CU*, the challenger stops. Otherwise, he picks* $b \xleftarrow{\$} \{0,1\}$.

Second phase: *$\mathcal{A}$ cannot use* $\overline{\mathsf{G}}.\mathsf{Corrupt}(i)$ *and* $\overline{\mathsf{G}}.\mathsf{Revoke}(i)$ *on i_0 or i_1. Moreover, $\mathcal{A}$ has access to the* G*-oracle:*

$\overline{\mathsf{G}}.\mathsf{Sign}_b(i_b, m)$ *simply returns* $\mathsf{G.sig}_{\mathsf{ssk}_{i_b}}(m)$.

Afterwards, $\overline{\mathsf{G}}.\mathsf{Open}(\sigma, m)$ *rejects all signatures produced by* $\overline{\mathsf{G}}.\mathsf{Sign}_b(i_b, m)$.

Guessing phase: *$\mathcal{A}$ outputs b' and the challenger returns the Boolean value $(b = b')$.*

Define $\mathsf{Adv}^{Anon}_{\mathsf{G-SIG}}(\lambda)$ *as in Definition 5. A group signature* G–SIG *is anonymous if* $\mathsf{Adv}^{Anon}_{\mathsf{G-SIG}}(\lambda)$ *is negligible.*

THEOREM 5.1. *Let* G–SIG *be a group signature scheme such that* $\mathsf{Adv}^{Anon}_{\mathsf{G-SIG}}(\lambda)$ *is negligible. Thus, SPADE is prover-anonymous and* $\mathsf{Adv}^{PA}_{SPADE}(\lambda) \leq \mathsf{Adv}^{Anon}_{\mathsf{G-SIG}}(\lambda)$.

PROOF. Assume that there is a polynomial-time adversary $\mathcal{A}$ having a non-negligible advantage $\mathsf{Adv}^{PA}_{\mathcal{A},SPADE}(\lambda)$ on a challenger in $\mathsf{Exp}^{PA}_{\mathcal{A},SPADE}(\lambda)$. $\mathcal{A}$ can be used by an adversary $\mathcal{B}$, which is challenged in $\mathsf{Exp}^{\mathsf{Anon}}_{\mathcal{B},\mathsf{G-SIG}}(\lambda)$. Thus, $\mathsf{Adv}^{\mathsf{Anon}}_{\mathcal{B},\mathsf{G-SIG}}(\lambda) \geq \mathsf{Adv}^{PA}_{\mathcal{A},SPADE}(\lambda)$, contradicting the assumption on G–SIG.

Initially, the challenger in $\mathsf{Exp}^{\mathsf{Anon}}_{\mathcal{B},\mathsf{G-SIG}}(\lambda)$ sends the key gpk and the revoked list RL to $\mathcal{B}$, which relays it to $\mathcal{A}$, as well as a function PRF and a PKE scheme E. Thus, $\mathcal{A}$ can initialize his own experiment $\mathsf{Exp}^{PA}_{\mathcal{A},SPADE}(\lambda)$ and return the public key pk_V to $\mathcal{B}$. Then, $\mathcal{B}$ creates the empty list CU. Having access to G–SIG-oracles from his challenger, $\mathcal{B}$ can simulate the DB-oracles for $\mathcal{A}$ as follows:

$\overline{\mathsf{DB}}.\mathsf{Join}^h(i)$: $\mathcal{A}$ creates P_i. $\mathcal{B}$ relays it to $\overline{\mathsf{G}}.\mathsf{Join}^h(i)$ and adds P_i to UL.

$\overline{\mathsf{DB}}.\mathsf{Join}^c(i)$: $\mathcal{A}$ creates P_i. $\mathcal{B}$ relays it to $\overline{\mathsf{G}}.\mathsf{Join}^c(i)$, obtains the signing key ssk_i,and adds P_i to UL and CU. $\mathcal{A}$ gets ssk_i.

$\overline{\mathsf{DB}}.\mathsf{Revoke}(i)$: $\mathcal{A}$ revokes P_i. $\mathcal{B}$ relays it to $\overline{\mathsf{G}}.\mathsf{Revoke}(i)$, which updates RL and sends it to $\mathcal{B}$. He relays it to $\mathcal{A}$.

$\overline{\mathsf{DB}}.\mathsf{Corrupt}(i)$: $\mathcal{A}$ corrupts P_i. $\mathcal{B}$ relays it to $\overline{\mathsf{G}}.\mathsf{Corrupt}(i)$ and gets ssk_i. $\mathcal{B}$ adds P_i to CU and returns ssk_i to $\mathcal{A}$.

$\overline{\mathsf{DB}}.\mathsf{Prover}(i)$: $\mathcal{B}$ simulates P_i for $\mathcal{A}$ as follows:

Initialization phase: $\mathcal{B}$ picks $N_P \xleftarrow{\$} \{0,1\}^n$, sends (i, N_P) to $\overline{\mathsf{G}}.\mathsf{Sign}(i, N_P)$ and gets back σ from $\mathcal{A}$. $\mathcal{B}$ then computes $e = \mathsf{E.enc}_{\mathsf{pk}_V}(N_P, \sigma)$ and sends e to $\mathcal{A}$ and receives (m, N_V). Afterwards, he computes $a = PRF_{N_P}(N_V)$.

Distance-bounding phase: $\mathcal{B}$ uses a, N_P and m to correctly answer to the challenges c_i sent by $\mathcal{A}$.

Verification phase: $\mathcal{B}$ builds (C, R) from the challenges and responses and sends to $\mathcal{A}$ the value $PRF_{N_P}(C, R)$.

Then, $\mathcal{A}$ picks two identities i_0 and i_1 and sends them to $\mathcal{B}$. If $i_0, i_1 \notin$ CU, $\mathcal{B}$ sends (i_0, i_1) to the challenger. In this phase, $\mathcal{B}$ simulates $\overline{\mathsf{DB}}.\mathsf{Prover}_b(\cdot)$ as follows. During the initialization phase, $\mathcal{B}$ picks $N_P \xleftarrow{\$} \{0,1\}^n$ sends it to $\overline{\mathsf{G}}.\mathsf{Sign}_b(i_b, m)$ and receives σ. He then computes $e = \mathsf{E.enc}_{\mathsf{pk}_V}(N_P, \sigma)$ and sends $(0, e)$ to $\mathcal{A}$. The objective of this prover simulation is the same as $\overline{\mathsf{DB}}.\mathsf{Prover}(i)$. Finally, during the guessing phase, $\mathcal{A}$ returns b' and $\mathcal{B}$ outputs the same b'.

The experiment is perfectly simulated for $\mathcal{A}$, and consequently, $\mathcal{B}$ wins his experiment with the same probability that $\mathcal{A}$ wins his and $\mathsf{Adv}^{\mathsf{Anon}}_{\mathcal{B},\mathsf{G-SIG}}(\lambda) = \mathsf{Adv}^{PA}_{\mathcal{A},SPADE}(\lambda)$, contradicting the assumption on G–SIG. □

5.2 Mafia Fraud

THEOREM 5.2. *Let* PKE *be a* IND–CCA2 *secure encryption scheme such that* $\mathsf{Adv}^{IND-CCA2}_{\mathsf{PKE}}(\lambda)$ *is negligible and*

G–SIG *a traceable signature scheme such that* $\mathsf{Adv}^{trace}_{\mathsf{G-SIG}}(\lambda)$ *is also negligible. Let* q_p *be the number of calls to the prover oracle,* q_v *the number of calls to the verifier oracle and* $p_{back}(n,t)$ *the probability to recover* N_P *through the backdoor. Thus,* $SPADE$ *is MF-resistant in the random oracle model if the challenges are drawn uniformly at random by the verifier and*

$$\mathsf{Adv}^{MF}_{SPADE}(\lambda) \leq p_{back}(n,t) + \frac{q_\mathsf{p}^2 + q_\mathsf{v}^2 + 1}{2^n} + \mathsf{Adv}^{trace}_{\mathsf{G-SIG}}(\lambda) + q_\mathsf{p} \cdot \mathsf{Adv}^{IND-CCA2}_{\mathsf{PKE}}(\lambda)\,.$$

The proof of this result is given in Appendix A.

5.3 Terrorist Fraud

THEOREM 5.3. *Let* $p_{back}(n,t)$ *be the probability to recover* N_P *through the backdoor and* r *be the number of bits of* N_P *unknown to any potential accomplice. Thus,* $SPADE$ *is* $\max\left(p_{back}(n,t), \left(\frac{3}{4}\right)^r\right)$-GameTF-*resistant.*

Remark, if there is an adversary $\mathcal{A}$ that knows all the bits of N_P, then there exists an adversary $\mathcal{B}$ to which $\mathcal{A}$ is helpful, and who wins with probability 1 (*i.e.*, in this case, SPADE is 1-GameTF-resistant). Similarly, SPADE is trivially 1-GameTF-resistant using insecure schemes. If an adversary can break the encryption scheme, he can find N_P encrypted in e and he can use it to authenticate himself. In addition, if an adversary can forge a signature, he can choose the nonce N_P himself, sign it and use it to authenticate. Finally, in a terrorist fraud scenario, the accomplice can authenticate himself even if he does not use the help of the malicious prover. Thus, a malicious prover cannot perform any efficient TF attacks while preserving his secret key.

PROOF. First, let us assume there is a polynomial-time $(q_{\mathsf{obs}}, q_\mathsf{v})$-adversary $\mathcal{A}$ that can win the TF game with a non-negligible probability with the help of his malicious prover. Then, we can construct an adversary $\mathcal{B}$ that can always win later on MF or IF games using $\mathcal{A}$'s view, contradicting Theorems 5.2 and 5.4. *A fortiori*, $\mathcal{A}$ can also do so.

To fool the verifier in a TF attack, $\mathcal{A}$ must get from P prior to the time-critical phase one of these two:

- Two n-bit strings $\mathbf{c}^0$ and $\mathbf{c}^1$ representing respectively the responses to the 0-challenges and the 1-challenges.
- An algorithm A to generate these strings.

If A is *stateless* (*i.e.*, the response to a challenge does not depend on the previous ones), these two are equivalent. For simplicity, the former case is used. Hence, $\mathcal{A}$ receives the strings $(\mathbf{c}^0, \mathbf{c}^1)$, representing his internal $\mathsf{view}_\mathcal{A}(\mathsf{sid})$. They are defined as:

Case 1 :	$\mathbf{c}_i^0 =$	a_i	and	$\mathbf{c}_i^1 =$	$a_i \oplus (N_{Pi} \oplus m_i)$	
Case 2 :	$\mathbf{c}_i^0 =$	a_i	and	$\mathbf{c}_i^1 =$	$\perp$	or
	$\mathbf{c}_i^0 =$	$\perp$	and	$\mathbf{c}_i^1 =$	$a_i \oplus (N_{Pi} \oplus m_i)$	
Case 3 :	$\mathbf{c}_i^0 =$	$\perp$	and	$\mathbf{c}_i^1 =$	$\perp$	

Under the assumptions that (1) the same values of N_P or N_V are never chosen twice by a cautious prover and an honest verifier, and that (2) the function PRF is pseudorandom, the values a_i can be seen as random values. Thus Case 2 cannot leak any information on N_P.

FACT 5.1. *For any round, the probability that* $\mathcal{A}$ *responds correctly to the challenge (let* p_i *denote such an event) is* 1 *in the first case,* $\frac{3}{4}$ *in the second one, and* $\frac{1}{2}$ *in the last one.*

If the objective of P and $\mathcal{A}$ is to fool the verifier, Case 2 should be preferred to Case 3, even though some bits of N_P may leak in the process. In fact, if Case 2 is chosen for r bits, half of them would leak during the time-critical phase for a winning session. These bits are the missing bits that had to be guessed successfully.

LEMMA 5.1. *Assume that a malicious prover* P *provides to his accomplice* $\mathcal{A}$ *two strings* $(\mathbf{c}^0, \mathbf{c}^1)$ *such that only one answer is known for* r *rounds (*Case 2*). Thus,* $\mathcal{A}$ *can fool a verifier in the time-critical phase of* $SPADE$ *only with probability*

$$\Pr(\mathsf{P}_0 \wedge \cdots \wedge \mathsf{P}_n | r) = 1^{n-r} \cdot \left(\frac{3}{4}\right)^r .$$

This requires that the challenges are independent and identically distributed and thatN_P has been randomly selected.

Other strategies can be used by P and $\mathcal{A}$ but they would leak more bits in the process.

First, remark that the trivial case in which $\mathcal{A}$ sends a successful query to the backdoor of V is discarded. This happens only with probability $p_{back}(n,t)$. Let suppose now that $\mathcal{A}$ has won the TF game with a non-negligible[3] probability $p_\mathcal{A}$. Consider the adversary-verifier session sid^* for which $\mathcal{A}$ has fooled V. This has happened with probability at least $\frac{p_\mathcal{A}}{q_\mathsf{v}}$. In such a case, $\mathcal{A}$ has successfully guessed the missing answers, which have been requested (*i.e.*, on average $\frac{r}{2}$ such queries). Since this happened, $(\frac{3}{4})^r$ should be greater than the non-negligible $\frac{p_\mathcal{A}}{q_\mathsf{v}}$. Hence,

$$\exists c, \forall n_c, \exists n > n_c, \left[\left(\frac{3}{4}\right)^r > \frac{n^{-c}}{q_\mathsf{v}} > n^{-c'}\right] .$$

since $q_\mathsf{v} \in n^{O(1)}$. Thus, r should be in $O(\log n)$.

If an adversary $\mathcal{B}$ gets the internal $\mathsf{view}_\mathcal{A}(\mathsf{sid}^*)$ and has eavesdropped to all the communications involving P, $\mathcal{A}$, and V, he would get e and N'_P such that $d_H(N_P, N'_P) \in O(\log n)$. Thus, $\mathcal{B}$ (as well as $\mathcal{A}$ himself) would be able to retrieve N_P directly through the backdoor of V and eventually be able to do a MF or an IF on behalf of P with probability one. This concludes the proof that $\mathcal{A}$ is very helpful for $\mathcal{B}$! □

5.4 Impersonation Fraud

THEOREM 5.4. *Let* PKE *be a* IND–CCA2 *secure encryption scheme such that* $\mathsf{Adv}^{IND-CCA2}_{\mathsf{PKE}}(\lambda)$ *is negligible and* G–SIG *a traceable signature scheme such that* $\mathsf{Adv}^{trace}_{\mathsf{G-SIG}}(\lambda)$ *is negligible. Let* q_p *be the number of calls to the prover oracle,* q_v *be the number of calls to the verifier oracle,* q *be the number of (different) digests generated by* H *and* $p_{back}(n,t)$ *the probability to recover* N_P *through the backdoor.* $SPADE$ *is IF-resistant in the random oracle model and*

$$\mathsf{Adv}^{IF}_{SPADE}(\lambda) \leq p_{back}(n,t) + \frac{q_\mathsf{p}^2 + q_\mathsf{v}^2 + q^2 + 1}{2^n} + \mathsf{Adv}^{trace}_{\mathsf{G-SIG}_n}(\lambda) + q_\mathsf{p} \cdot \mathsf{Adv}^{IND-CCA2}_{\mathsf{PKE}}(\lambda)\,.$$

The proof is very similar to the one given in Appendix A.

[3]Formally, $p_\mathcal{A}$ is such that $\exists c, \forall n_c, \exists n > n_c, p_\mathcal{A} > n^{-c}$.

5.5 Distance Fraud

THEOREM 5.5. *If m is drawn from a uniform distribution by the verifier, then SPADE is DF-resistant, and*

$$\mathsf{Adv}^{DF}_{SPADE}(\lambda) \leq \left(\frac{3}{4}\right)^n .$$

PROOF. To defeat the time-bound for each round i, the far-away prover must send r_i before receiving c_i. Since c_i is unpredictable, the prover cannot determine in advance whether he must respond a_i or $a_i \oplus N_{Pi} \oplus m_i$. Hence, if these two possible responses are different, he must guess c_i. On the other hand, if the two possible responses are equal, he succeeeds in passing the round with probability 1. Due to the uniform distribution of m, and the fact it is picked by V after the prover has committed to N_P with e, $Pr[a_i \oplus N_{Pi} \oplus m_i = a_i] = \frac{1}{2}$. From this, we deduce an upper bound on the probability of success for a given round: $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4}$. Since they are n independent rounds, the probability of success is at most $\frac{3}{4}^n$. □

5.6 Impersonation and Multiple Verifiers

Traditionally, security models for DB protocols assume that there is only one verifier. To extend these models to support numerous verifiers, we have mainly two options, which are to consider that verifiers are *honest-but-curious* or *malicious*. Unfortunately, in the latter case, SPADE as presented does not prevent the impersonation of a prover by a malicious verifier. Indeed, knowing N_P and its anonymous signature by P, the malicious verifier can simply reuse this information to another verifier. However, a simple modification can ensure that SPADE is secure in this broader context. Let assume that the verifier V_i has a public certificate with an identifier id_{V_i} and a public key sk_{V_i}. Thus, each nonce N_P can be associated to the appropriate identifier id_{V_i}. In fact, P would sign and encrypt $N_P||\mathsf{id}_{V_i}$.

If the malicious verifier ends up with the ciphertext e for another verifier, he cannot to retrieve N_P or its signature due to security of the encryption scheme. Furthermore, even if he is able to obtain a from P simply by sending only 0-challenges, the one-way property of the functions in PRF would prevent the verifier to retrieve N_P from N_V and a. Thus, the malicious verifier is limited to a classical MF against P and the legitimate verifier. The formal security model of this generalization is an important problem and will be addressed in future work.

6. THE PRESENCE OF THE BACKDOOR

The objective of the backdoor in the verifier is mainly to deter any prover to help potential accomplices. Remark that this mechanism is stateless for the verifier, as he simply has to decrypt the initial message of the protocol to retrieve the information needed to impersonate a prover. We further analyze the impact of the backdoor in this section.

The probabilities to detect MF or TF attacks depend on the proximity threshold t. There is clearly a trade-off between these two probabilities, as one increases and the other one decreases in function of t. Unfortunately, there is no *optimal* value for t, rather it depends of the security requirements – and the underlying threat models. At the end of this section, we review the main scenarios that we envision.

6.1 Querying the Backdoor

Let us assume that a verifier V_{s}, having a n-bit secret s, can be queried a polynomial number of times with strings $x \in \{0,1\}^n$. If a query is *close* to s (*i.e.*, if $d_H(\mathsf{s}, x) \leq t$, for some $t < \frac{n}{2}$), the verifier simply returns his secret. Otherwise, he simply outputs 0.

Since the number of potential queries is 2^n and the number of strings at Hamming distance at most t of s is simply $\sum_{k=0}^{t} \binom{n}{k}$,

LEMMA 6.1. *The probability that the i^{th} random query is successful (let $\mathcal{Q}_i$ denote such an event) is*

$$\mathbf{p} = \Pr[\mathcal{Q}_i|n,t] = 2^{-n} \times \sum_{k=0}^{t} \binom{n}{k} .$$

6.2 Best Strategy to Retrieve the String s

As described in Section 5.3, the accomplice receives from a malicious prover two strings $(\mathbf{c}^0, \mathbf{c}^1)$ to help him to fool the verifier in a TF attack. These strings would clearly allow the accomplice to know the values of n and $r = \alpha n$. However, he can easily estimate the proximity threshold t by sending queries with increasing Hamming weight.

Let assume that $d_H(\mathbf{c}^0 \oplus \mathbf{c}^1, \mathbf{s}) = r$ such that $t < r < \frac{n}{2}$. It can be shown that the best strategy for an adversary is to flip the minimal number of bits to transform $\mathbf{c}^0 \oplus \mathbf{c}^1$ into $\mathbf{s}$. Thus, assume that the adversary selects $r - t$ bits, complements them and submits the result to V_{s}.

LEMMA 6.2. *Given n, t and r, the probability that the i^{th} random query to retrieve s by flipping exactly $r - t$ bits of a given chain $\mathbf{c}^0 \oplus \mathbf{c}^1$ such tjat $d_H(\mathbf{s}, \mathbf{c}^0 \oplus \mathbf{c}^1) = t$ is given by*

$$\Pr[\mathcal{Q}_i|\mathbf{c}^0 \oplus \mathbf{c}^1, n, t, r] = \frac{\binom{r}{r-t}}{\binom{n}{r-t}}, \text{ by symmetry } \frac{\binom{n-r+t}{t}}{\binom{n}{n-r}} .$$

6.3 Simple Case in which $t = \alpha n$ and $r = (\alpha + \epsilon)n$

Consider that the backdoor provided by the verifier is such that $t = \alpha n$, for some $\alpha > 0$. The probability of obtaining s with a query (Lemma 6.1) can be bounded by

$$\Pr[\mathcal{Q}_i|n,t] \approx \frac{O(1)}{2^{n(1-H(\alpha))} \cdot \sqrt{n}} ,$$

in which $H(\alpha) = \alpha \log \frac{1}{\alpha} + (1-\alpha) \log \frac{1}{(1-\alpha)}$ (see [22], Prob. 9.42). This probability increases as α increases in $[0 \cdots \frac{1}{2})$.

THEOREM 6.1. *An adversary can achieve a MF or an IF with the involuntary assistance of the verifier V_{s} through $n^{O(1)}$ queries to V_{s} with probability*

$$\Pr[\cup_i \mathcal{Q}_i|n,t] \leq \sum_i \frac{O(1)}{2^{n(1-H(\alpha))} \cdot \sqrt{n}} = \frac{n^{O(1)}}{2^{n(1-H(\alpha))}} .$$

This follows from the union bound and the independence of queries.

An adversary can also get enough information to achieve a MF or an IF attack from the accomplice. Consider the situation in whcich the adversary has retrieved a string at Hamming distance $r = (\alpha + \epsilon)n$ of $\mathbf{s}$, for some $0 < \alpha < (\alpha + \epsilon) \leq \frac{1}{2}$. Thus, the equation of Lemma 6.2 can be

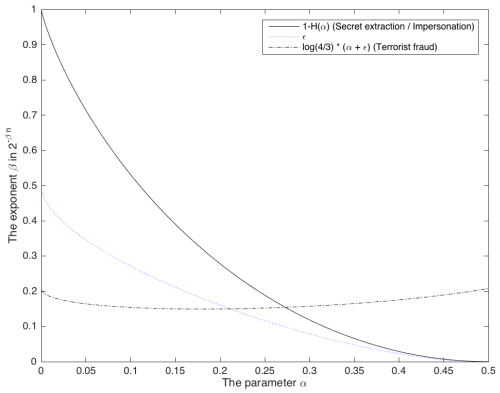

Figure 3: Plotting with respect to α the exponents of the probability in Lemma 5.1 ($\alpha+\epsilon$) and Theorem 6.1 ($1-H(\alpha)$), and the value of ϵ making $1-H(\alpha)=\nu(\alpha,\epsilon)$.

rewritten as

$$\Pr[\mathcal{Q}_i|\mathbf{c}^0 \oplus \mathbf{c}^1, n, \alpha n, (\alpha+\epsilon)n] = \frac{\binom{\alpha n+\epsilon n}{\alpha n}}{\binom{\epsilon n+(1-\epsilon)n}{\epsilon n}}$$
$$\approx \sqrt{\frac{(\alpha+\epsilon)(1-\epsilon)}{\alpha}}\left[\frac{(\alpha+\epsilon)^{\alpha+\epsilon}(1-\epsilon)^{1-\epsilon}}{\alpha^{\alpha}}\right]^n = \frac{O(1)}{2^{\nu(\alpha,\epsilon)n}} ,$$

in which $\nu(\alpha,\epsilon) = (\alpha+\epsilon)\log\frac{1}{\alpha+\epsilon} + (1-\epsilon)\log\frac{1}{1-\epsilon} - \alpha\log\frac{1}{\alpha}$ (see [21], Section 1.2.6).

THEOREM 6.2. *An adversary can achieve a MF or IF attack with the help of an accomplice of a TF attack through $n^{O(1)}$ queries to V_s with probability*

$$\Pr[\cup_i \mathcal{Q}_i|\mathbf{c}^0 \oplus \mathbf{c}^1, n, \alpha n, (\alpha+\epsilon)n] \leq \sum_i \frac{O(1)}{2^{\nu(\alpha,\epsilon)n}} = \frac{n^{O(1)}}{2^{\nu(\alpha,\epsilon)n}} .$$

6.4 Different Threat Models

In the following scenarii, the backdoor threshold t is set to αn (for some constant $0 < \alpha < 0.5$). An honest prover would like to have the lowest backdoor threshold as possible ($\alpha \to 0$) since it would ensure the protection of his secret and minimize the probability that an adversary retrieves some useful information directly from the verifier (as in Theorem 6.1). However, a malicious prover would take profit of such a small value. A relatively small value of $r = (\alpha+\epsilon)n$ would protect his secret while minimizing the probability of Lemma 5.1. Depending on the threats, the parameters (α,ϵ) are defined differently.

Honest prover. This prover will not attempt any TF attack and thus third parties represent the only adversaries. Hence, t can be set by the verifier to a small value such as $t = 0.01\cdot n$, giving a probability in $O\left(\frac{n^{O(1)}}{2^{0.92n}}\right)$ of extracting the secret and having a successful MF or IF attack (as in Theorem 6.1).

Malicious and suspicious prover. This prover may attempt to do a TF attack while doing his best to protect his secret. His accomplice should not obtain any advantage over the backdoor, more precisely Theorems 6.1 and 6.2 probabilities should be equal (ϵ must be chosen accordingly). In Figure 3, the intersection of the exponent $\alpha+\epsilon$ (terrorist fraud detection probability) and the exponent $1-H(\alpha)$ (secret extraction probability) gives the equilibrium. At this point, both probabilities are equal and in $O\left(\frac{n^{O(1)}}{2^{0.3715}}\right)$, for $t = 0.273\cdot n$ (chosen by V) and $\epsilon = 0.0985$ (chosen by P). Notice that these intersection points have been obtained through numerical approximation.

Malicious prover having some trust in his accomplice. This prover may attempt to perform a TF. However, he accepts that his accomplice is able to impersonate him with a better probability than any other party. The prover may chose ϵ ten times smaller than expected in the previous scenario. This would increase the success probability of his TF attack. Hence, the verifier would have the responsibility to increase α, increasing implicitly all the success probabilities of the attacks. By plotting these curves as in Figure 3, we obtain $O\left(\frac{n^{O(1)}}{2^{0.3028}}\right)$, for $t = 0.2945\cdot n$ (chosen by V).

7. CONCLUSION

Table 1: Summary of SPADE security properties.

Properties	Security probabilities
PA	$\mathsf{Adv}^{\mathsf{Anon}}_{\mathsf{G-SIG}}(\lambda)$
MF	$p_{back}(n,t) + \frac{q_p^2+q_v^2+1}{2^n} + \mathsf{Adv}^{trace}_{\mathsf{G-SIG}}(\lambda)$ $+q_p \cdot \mathsf{Adv}^{IND-CCA2}_{\mathsf{PKE}}(\lambda)$
TF	$\max\left(p_{back}(n,t), \left(\frac{3}{4}\right)^r\right)$
IF	$p_{back}(n,t) + \frac{q_p^2+q_v^2+q^2+1}{2^n} + \mathsf{Adv}^{trace}_{\mathsf{G-SIG}n}(\lambda)$ $+q_p \cdot \mathsf{Adv}^{IND-CCA2}_{\mathsf{PKE}}(\lambda)$
DF	$\left(\frac{3}{4}\right)^n$

Considering the widespread development of contactless technologies, we believe that it is crucial to develop provably secure DB protocols, which address privacy issues to limit the ability of tracking users. In this paper, we have proposed SPADE, a provably TF-resistant prover-anonymous DB protocol, which uses group signatures to hide the prover's identity, even against a potentially malicious verifier. While our construction is provably resistant to all known attacks against DB protocols (see Table 1 for a summary), the backdoor introduced to obtain the TF-resistance lowers the resistance of the protocol to other threats. This is a frequent problem when designing provably TF-resistant protocols.

In addition to building the first protocol ensuring these properties, we have introduced a promising new approach to ensure TF resistance. In essence, the information leaked to an accomplice during a TF is no longer a long-term secret key but rather a temporary session key. Such a session key can then be used by the accomplice to authenticate. This novel approach opens the door for further research on terrorist fraud resistance.

Acknowledgments

We would like to thank the anonymous reviewers and our sheperd, Bart Preneel, who by their comments helped us to improve the quality of the paper.

8. REFERENCES

[1] A. Ahmadi and R. Safavi-Naini. Privacy-preserving distance-bounding proof-of-knowledge. In *ICICS 2014*, LNCS 8958, pages 74–88. Springer, 2015.

[2] G. Avoine, M. Ali Bingöl, S. Kardaş, C. Lauradoux, and B. Martin. A formal framework for analyzing RFID distance bounding protocols. *Journal of Computer Security - Special Issue on RFID System Security*, 19(2):289–317, 2010.

[3] A. Bay, Boureanu I, A. Mitrokotsa, I. Spulber, and S. Vaudenay. The Bussard-Bagga and other distance-bounding protocols under attacks. In *Inscrypt 2012*, LNCS 7763, pages 371–391. Springer, 2012.

[4] M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In *EUROCRYPT 2000*, LNCS 1807, pages 259–274, 2000.

[5] M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT 2003*, LNCS 2656, pages 614–629. Springer, 2003.

[6] S. Bengio, G. Brassard, Y. G. Desmedt, C. Goutier, and J.-J. Quisquater. Secure implementation of identification systems. *Journal of Cryptology*, 4(3):175–183, 1991.

[7] I. Boureanu, A. Mitrokotsa, and S. Vaudenay. Secure and lightweight distance-bounding. In *LightSec 2013*, LNCS 8162, pages 97–113. Springer, 2013.

[8] S. Brands and D. Chaum. Distance-bounding protocols. In *EUROCRYPT '93*, LNCS 765, pages 344–359. Springer, 1993.

[9] A. Brelurut, D. Gérault, and P. Lafourcade. Survey of distance bounding protocols and threats. In *FPS 2015*, LNCS 9482, pages 29–49. Springer, 2015.

[10] L. Bussard and W. Bagga. Distance-bounding proof of knowledge to avoid real-time attacks. In *Security and Privacy in the Age of Ubiquitous Computing*, IFIP, pages 222–238. Springer, 2005.

[11] C. Cremers, K. Rasmussen, B. Schmidt, and S. Čapkun. Distance hijacking attacks on distance bounding protocols. In *Symp. on Security and Privacy*, pages 113–127. IEEE, 2012.

[12] Y. G. Desmedt, C. Goutier, and S. Bengio. Special uses and abuses of the Fiat-Shamir passport protocol. In *Proc. of Advances in Cryptology – CRYPTO'87*, LNCS 293, pages 21–39. Springer, 1988.

[13] U. Dürholz, M. Fischlin, M. Kasper, and C. Onete. A formal approach to distance-bounding RFID protocols. In *Information Security*, LNCS 7001, pages 47–62. Springer, 2011.

[14] M. Fischlin and C. Onete. Subtle kinks in distance-bounding: an analysis of prominent protocols. In *WISec*, pages 195–206. ACM Press, 2013.

[15] M. Fischlin and C. Onete. Terrorism in distance bounding: Modeling terrorist fraud resistance. In *ACNS 2013*, LNCS, pages 414–431. Springer, 2013.

[16] S. Gambs, C. Onete, and J.-M. Robert. Prover anonymous and deniable distance-bounding authentication. In *AsiaCCS'14*, pages 501–506. ACM Press, 2014.

[17] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.

[18] J. Hermans, A. Pashalidis, F. Vercauteren, and B. Preneel. A new RFID privacy model. In *ESORICS 2011*, LNCS 6879, pages 568–587. Springer, 2011.

[19] J. Hermans, Peeters R, and C. Onete. Efficient, secure, private distance bounding without key updates. In *WiSec 2013*, pages 207–218. ACM Press, 2013.

[20] C. H. Kim, G. Avoine, F. Koeune, F.-X. Standaert, and O. Pereira. The Swiss-Knife RFID distance bounding protocol. In *ICISC 2008*, LNCS 5461, pages 98–115. Springer, 2008.

[21] D. E. Knuth. *The Art of Computer Programming: Fundamental Algorithms, 3^{rd}ed.* Addison Wesley, 1997.

[22] D. E. Knuth, R. L. Graham, and O. Patashnik. *Concrete mathematics, 2^{nd} ed.* Adison Wesley, 1994.

[23] S. Mangard, E. Oswald, and T. Popp. *Power analysis attacks: Revealing the secrets of smart cards*. Springer Science & Business Media, 2008.

[24] S. Micali, C. Rackoff, and B. Sloan. The notion of security for probabilistic cryptosystems. *SIAM Journal on Computing*, 17(2):412–426, 1988.

[25] T. Nakanishi, H. Fujii, Y. Hira, and N. Funabiki. Revocable group signature schemes with constant costs for signing and verifying. In *PKC 2009*, LNCS 5443, pages 463–480. 2009.

[26] V. Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint 2004/332, 2004.

[27] S. Vaudenay. Proof of proximity of knowledge. Cryptology ePrint 2014/695, 2014.

[28] S. Vaudenay. Private and secure public-key distance bounding; application to NFC payment. In *Financial Cryptography*, LNCS 8975, pages 207–216. Springer, 2015.

[29] Serge Vaudenay. On privacy models for RFID. In *Proc. of Advances in Cryptology – Asiacrypt'07*, LNCS 4883, pages 68–87. Springer, 2007.

A. Mafia-Fraud Resistance Proof

Let the traceability experiment $\mathsf{Exp}^{\mathsf{trace}}_{\mathcal{A},\mathsf{G-SIG}}(\lambda)$ for $\mathcal{A}$ on a revocable group signature scheme G–SIG be defined as:

Definition 13. *The initialization is described in Def. 12.* ***Guessing phase:*** *$\mathcal{A}$ outputs a message m^* and a signature σ^*. Then, the challenger outputs 1 if and only if:*

- $(m^*, \sigma^*) \notin \Sigma$ and $\mathsf{G.ver}_{\mathsf{gpk}}(\sigma^*, m^*, \mathsf{RL}) = 1$,
- $\mathsf{G.ope}_{\mathsf{msk}}(\sigma^*, m^*, \mathsf{UL}, \mathsf{gpk}) \notin \mathsf{CU} \setminus \mathsf{RL}$.

i.e., $\mathcal{A}$ has been able to produce a valid and fresh signature for a message without using the key of a corrupted user. Define $\mathsf{Adv}^{\mathit{trace}}_{\mathsf{G-SIG}}(\lambda)$ as in Def. 6. A group signature G–SIG *is traceable if $\mathsf{Adv}^{\mathit{trace}}_{\mathsf{G-SIG}}(\lambda)$ is negligible.*

Proof. of Theorem 5.2 This proof is built as a series of games [26]. The idea is to go from the initial security game $G0$ to a final game for which no successful polynomial-time adversary can exist. Each game Gi is associated with the probability $Pr[Gi]$ that an adversary wins the game. The proof makes small steps from one game to the next one, such that $|Pr[Gi] - Pr[G(i+1)]|$ is negligible. Proving the difference property for each step, and the fact that the winning probability for the last game is negligible prove that no adversary can win $G0$ with a non-negligible probability.

Proof strategy. We first rule out the use of the backdoor returning the provers' secret keys. Then, we force the verifier and the provers to never reuse their respective values N_V and N_P twice, and rule out a group signature forgery by the adversary. The last transition consists in replacing the initial message e by the encryption of a random message, completely independent from the actual value N_P used in the protocol. This game can be seen as the classical shared-secret DB protocol. The secret N_P is shared offline between the prover and the verifier, and the adversary $\mathcal{A}$ does not have any more access to a ciphertext containing this secret. This game $\mathcal{A}$ cannot be won with a non-negligible probability by any adversary. We build the following sequence of games, reducing progressively the capabilities of the adversaries. This follows from the difference lemma presented by Shoup [26].

In the following, WL denotes a list in which each $\mathsf{Prover}(i)$ stores every generated tuple (e, N_P, σ). It allows the provers and the verifier oracles to exchange their shared secrets.

Game $G1$. The backdoor is deactivated. $\mathcal{A}$ loses a (negligible) probability (Theorem 6.1) of recovering N_P from the backdoor. From the difference lemma [26], $|Pr[G1] - Pr[G0]| \leq p_{back}(n,t)$.

Game $G2$. The oracle $\mathsf{Prover}(i)$ uses different N_P values for each of the q_p calls. $\mathcal{A}$ loses an advantage of at most $q_\mathsf{p}^2/2^n$. This follows from the binomial expansion of the probability that a given value has been selected zero or once and the union bound. Thus, $|Pr[G2] - Pr[G1]| \leq q_\mathsf{p}^2/2^n$.

Game $G3$.The oracle $\mathsf{Verifier}$ uses different N_V values for each of the q_v calls. Thus, $|Pr[G3] - Pr[G2]| \leq q_\mathsf{v}^2/2^n$.

Game $G4$. The oracle $\mathsf{Verifier}$ aborts if $\mathsf{G.ver}_{\mathsf{gpk}}(\sigma, N_P, \mathsf{RL}) = 1$, no tuple of WL contains σ and $\mathsf{G.ope}_{\mathsf{msk}}(\sigma, N_P, \mathsf{UL}, \mathsf{gpk}) \notin \mathsf{CU}$. This happens if $\mathcal{A}$ produces a fresh signature σon a value N_P without using the key of a corrupted user. Since $\mathsf{G\text{–}SIG}$ is a traceable group signature scheme, $|Pr[G4] - Pr[G3]| \leq \mathsf{Adv}^{trace}_{\mathsf{G\text{–}SIG}}(\lambda)$.

Game $G5$. The oracle $\mathsf{Prover}(i)$ sends an encrypted value N_{P0} unrelated to the actual value N_{P1} used in the protocol *by both parties*. The oracles are modified as follows:

- $\mathsf{Prover}(i)$ sets $N_{P0}, N_{P1} \xleftarrow{\$} \{0,1\}^n$, $\sigma_0 = \mathsf{G.sig}_{\mathsf{psk}}(N_{P0})$, $\sigma_1 = \mathsf{G.sig}_{\mathsf{psk}}(N_{P1})$, and $e = \mathsf{E.enc}_{\mathsf{pk}_v}(N_{P0}, \sigma_0)$. It adds (e, N_{P1}, σ_1) to a list WL. It then sends e, but uses N_{P1}.
- $\mathsf{Verifier}$ acts as in $G4$, except for the N_P computation:
  - If $(e, N_{P1}, \cdot) \in \mathsf{WL}$, it uses N_{P1}. This is the main case, which corresponds to non-corrupted provers.
  - If $(e, \cdot, \cdot) \notin \mathsf{WL}$, it computes $(N_P^*, \sigma^*) = \mathsf{E.dec}_{\mathsf{sk}_v}(e)$ and uses N_P^*.

Thus, $\mathcal{A}$ loses the possibility of recovering the value of N_P from the ciphertext e. However, this advantage is negligible since the PKE encryption scheme is $\mathsf{IND\text{–}CCA2}$ secure.

Consider the extension of the $\mathsf{IND\text{–}CCA2}$ security concept allowing a polynomially-bounded number q_p of challenges [4], with a single public/private key pair. During this experiment, the challenger picks a bit b and gives an oracle $\mathsf{LREnc}_\mathsf{b}^{\mathsf{pk}_\mathsf{v}}(\mathsf{m}_0, \mathsf{m}_1)$, which returns the encryption of m_b. This oracle can be queried at most q_p times. The challenger also provides a decryption oracle that deciphers any ciphertext, except if it was generated by $\mathsf{LREnc}_\mathsf{b}^{\mathsf{pk}_\mathsf{v}}(\cdot,\cdot)$. The distinguisher wins if he properly guesses b using the oracles.

Let us assume that there exists an efficient adversary $\mathcal{A}$ for the game $G5$. Thus, an efficient distinguisher $\mathcal{B}$ can be defined for the $\mathsf{IND\text{–}CCA2}$ experiment using $\mathcal{A}$.

Initialization $\mathcal{B}$ sets up the environment (except the PKE setting). He creates two user lists UL and RL, and a group signature scheme setup with $\mathsf{G.gen}(1^\lambda)$, obtaining a couple $(\mathsf{gpk}, \mathsf{msk})$. He adds to UL n_p provers using $\mathsf{DB.join}_{\mathsf{msk}}(i, \mathsf{UL})$.

Simulation $\mathcal{B}$ simulates a $SPADE$ environment for $\mathcal{A}$ with the DB-oracles of $G4$, with the following modifications to the prover and verifier oracles. When the oracle $\mathsf{Prover}()$ is called, it sets $e = \mathsf{LREnc}_\mathsf{b}^{\mathsf{pk}_\mathsf{v}}(\mathsf{N_{P0}}||\sigma_0, \mathsf{N_{P1}}||\sigma_1)$, adds (e, N_{P1}, σ_1) to WL and uses N_{P1} internally. The oracle $\mathsf{Verifier}$ acts as in $G4$ if it receives e such that $(e, N_{P1}, \sigma_1) \in \mathsf{WL}$. Otherwise, it decrypts e with the provided decryption oracle. Finally, $\mathcal{B}$ returns the result of the authentication $OutV$ to the challenger, which is 1 if the verifier accepts, and 0 otherwise.

If $b = 1$, $e = \mathsf{E.enc}_{\mathsf{pk}_v}(N_{P1}, \sigma_1)$, $(e, N_{P1}, \sigma_1) \in \mathsf{WL}$ and both parties use N_{P1} internally. This games corresponds exactly to the game $G4$. Otherwise, $e = \mathsf{E.enc}_{\mathsf{pk}_v}(N_{P0}, \sigma_0)$ while the N_{P1} is used. This simulates the game $G5$.

Let $\mathcal{B}_0$ denote the event that the distinguisher outputs 0, and $\mathcal{B}_1$ the event that it outputs 1. Thus, $Pr[\mathcal{B}_1|b=1] = Pr[G4]$ and $Pr[\mathcal{B}_0|b=0] = 1 - Pr[G5]$, since $\mathcal{B}$ returns the result of the authentication to the distinguisher. The winning probability of $\mathcal{B}$ is then $Pr[\mathcal{B}_1 \wedge b = 1] + Pr[\mathcal{B}_0 \wedge b = 0]$, which is equal to $Pr[G4] \cdot \frac{1}{2} + (1 - Pr[G5]) \cdot \frac{1}{2} = \frac{1}{2} + \frac{Pr[G4] - Pr[G5]}{2}$. Now assume that $Pr[G4] - Pr[G5]$ is non negligible. Then $\mathcal{B}$ has a non-negligible advantage on the extended $\mathsf{IND\text{–}CCA2}$ game. However, the advantage of $\mathcal{B}$ cannot be more than q_p times the advantage on the original $\mathsf{IND\text{–}CCA2}$ experiment. Thus, $|Pr[G5] - Pr[G4]| \leq q_\mathsf{p} \cdot \mathsf{Adv}^{IND-CCA2}_{\mathsf{PKE}}(\lambda)$, which is negligible by hypothesis.

The final game. The final step is to prove that the probability to win $G5$ is negligible. Since $\mathcal{A}$ cannot purely relay messages without using $\mathsf{DB.Taint}(.)$ and invalidating the session, for each round j,

- If $\mathcal{A}$ sends c'_j to $\mathsf{Prover}(i)$ before getting c_j from $\mathsf{Verifier}$, he would be wrong with probability $\frac{1}{2}$.
- If $\mathcal{A}$ sends c'_j to $\mathsf{Prover}(i)$ after receiving c_j from $\mathsf{Verifier}$, he must send r'_j before receiving r_j since pure relay is not allowed. He would be wrong with probability $\frac{1}{2}$.

In the first case, a wrongly guess challenge invalidates the final message τ. Hence, $\mathcal{A}$ must recompute this message to succeed. He cannot do so without guessing N_P, and his success probability is no more than $\left(\frac{1}{2}\right)^n$. In the second case, the authentication fails if $\mathcal{A}$ sends a single wrong response. Thus, his success probability is again no more than $\left(\frac{1}{2}\right)^n$. Since $a = \mathsf{H}(N_P, N_V)$ is a truly random value, during the time-critical phase, $\mathcal{A}$ cannot do better than guessing the answer of each challenging round.Thus, $Pr[G5] \leq (\frac{1}{2})^n$. Finally, $\mathcal{A}$ cannot determine τ except with a negligible probability (because he would have to guess the value used as N_P).

From the sequence of games, let $\mathcal{A}^{MF} = 1$ denote the event that $\mathcal{A}$ wins the MF experiment, $Pr[\mathcal{A}^{MF} = 1] \leq p_{back}(n,t) + \frac{q_\mathsf{p}^2 + q_\mathsf{v}^2 + 1}{2^n} + \mathsf{Adv}^{trace}_{\mathsf{G\text{–}SIG}n}(\lambda) + q_\mathsf{p} \cdot \mathsf{Adv}^{IND-CCA2}_{\mathsf{PKE}}(\lambda)$ □