Zero-Communication Seed Establishment for Anti-Jamming Techniques

Kim Pecina
Saarland University
Computer Science Department
pecina@cs.uni-saarland.de

Esfandiar Mohammadi
Saarland University
Computer Science Department
mohammadi@cs.uni-saarland.de

Christina Pöpper
Ruhr-University Bochum
Horst-Görtz-Institute for IT-Security
christina.poepper@rub.de

***Abstract*—In an environment perturbed by malicious interference, establishing a reliable wireless connection without shared secrets typically relies on uncoordinated spread spectrum methods, such as UFHSS or UDSSS. These methods offer a low-throughput channel for the exchange of key contributions with the goal to establish a shared secret. They, however, incur a significant delay when confronted with adversarial jamming, in particular when long public-key credentials must be exchanged. In this paper, we propose and evaluate a scheme that is orders of magnitude more efficient than prior proposals. Drawing on powerful identity-based (ID-based) cryptography and exploiting a-priori knowledge about the identities of the communication partners, the parties can immediately and without any communication derive a cryptographically strong seed (or key) for creating a high-throughput, jamming-resistant channel. Even without a-priori knowledge, our scheme still benefits from the certificate-free authentication offered by ID-based cryptography. Additionally, we define the notion of a secure seed establishment scheme for anti-jamming methods and prove that our protocol satisfies this definition. We further conduct an experimental evaluation of our approach to demonstrate its practicality: on mobile commodity hardware such as notebooks, tablet computers, and smart phones, a non-optimized implementation derives shared seeds in less than half a second.**

I. Introduction

In cellular, mobile and ad-hoc communication scenarios, wireless devices use a common frequency band and modulation scheme for communicating and exchanging messages. Due to the open nature of the wireless communication channel, an attacker can easily disturb or prevent the message exchange by jamming, also from distance. The communication partners can circumvent jamming attacks by using spread-spectrum techniques such as FHSS or DSSS [18], [27] as long as they share the same secret cryptographic material (and the attacker is not blocking the entire available frequency band at once). For both techniques, the sender and the receiver use the cryptographic material as seed for pseudo-random generators that will subsequently generate the same spreading sequences.

SENT '14, 23 February 2014, San Diego, CA, USA
Copyright 2014 Internet Society, ISBN 1-891562-36-3
http://dx.doi.org/10.14722/sent.2014.23004

A setup with shared secrets, however, is impractical for large or dynamically changing sets of communication partners. Naturally, such a shared secret setup relies on trust in all communication parties and requires dynamic re-keying in the event of a key compromise. In particular, the question arises how devices that do not share any cryptographic secrets can agree on such shared secret seeds.

Prior proposals (e.g., [19], [20], [24], [26]) require an active public-key infrastructure and (low-throughput) communication under jamming attacks: a priori knowledge about the respective communication partner is not required but the communication partners must possess valid certificates (certified public keys) issued by the trusted third party. The drawback of these proposals is that the seed establishment may incur a non-negligible delay in the order of tens of seconds [24], [25], which can be intolerable for time-critical communication as may be required in industrial control systems, smart grids, or alarm systems [30]. In particular, this delay depends on the power attributed to the attacker, which must not be underestimated. Additionally, previous protocols that establish a shared secret rely on randomized protocols that are either communication intensive (e.g., UFHSS) or processing intensive (e.g., UDSSS) [20], which, given the nature of the wireless devices, uses up precious resources such as battery power. Recent work [3] already explores reducing the amount of transmitted data by leveraging the certificate-free authorization offered by ID-based cryptography. That work, however, only considers the scenario where the communication partners do not know each other's identity a-priori.

Contribution. In this work, we address the efficient establishment of shared seeds for spread-spectrum methods. The contributions of this work are as follows:

- We investigate the use of ID-based cryptography for establishing seeds under jamming perils in communication settings where the communication partners know each other's identity. Using our scheme, the devices derive cryptographically strong seeds without having to communicate; these keys can later be used for coordinated spread-spectrum methods. This approach makes the seed establishment orders of magnitude faster than using PKI- and certificate-based uncoordinated spread-spectrum methods.
  Due to the absence of communication, the proposed method is insensitive to the attacker's transmission and reaction capabilities. Exploiting the power of ID-based

cryptography, it supports dynamically changing sets of participants without incurring any key additional management; only the identity of the communication partner is required. In practical scenarios, learning the communication partner's identity is easier than obtaining an authenticated public key or managing a shared secret.

- We characterize the security of our ID-based seed establishment scheme for anti-jamming purposes: first, we introduce the notion of a *secure seed establishment scheme* (a variant of the eCK model for authenticated key exchange); second, we prove that our proposed scheme fulfills this definition.
- We experimentally demonstrate the efficiency of our approach by evaluating running times of a prototypical implementation on commodity mobile devices, in particular smart phones and notebooks.

Outline. We begin with a presentation of the key idea of our proposal and its positioning with respect to other work (Section II). We then present our system model and motivate our attacker model (Section III). Thereafter, we present our seed establishment scheme (Section IV). Then we define the notion of a secure seed establishment scheme and prove that our scheme satisfies this notion (Section V). We continue with illustrating the efficiency of our scheme with a prototype implementation on mobile devices and commodity notebooks (Section VI). Finally, we describe related work (Section VII) and conclude (Section VIII).

II. Overview and Key Idea

This paper addresses the problem of establishing shared secrets under jamming attacks between communication partners that do not share cryptographic material. The solution space to this problem contains several dimensions: (1) the required knowledge about the communication partner (e.g., in terms of name, identity, and certificates), (2) the a-priori infrastructure setup, (3) the amount of communication needed for the establishment of the secret seed, and (4) the computational complexity and time needed by the communication parties for establishing the seed. In other words, possible solutions to this problem are determined in terms of what needs to be known, received, and computed.

The problem above has previously been addressed as *anti-jamming key-establishment dependency* [26]. In this paper, we refer to it as *seed establishment*. Existing solutions (e.g., [24]–[26]) break the dependency cycle by enabling anti-jamming communication without shared secrets, as depicted by Figure 1 (a). They achieve this by transmitting the messages of a standard (e.g., Diffie-Hellman-based) key-establishment scheme on randomized frequency channels. As part of the infrastructure setup, these schemes require a trusted third party to certify the public keys of the involved communication parties to enable authentic key contributions. These certificates must be either communicated while the key is being established, or they must be pre-shared before the communication phase. In the latter case, potentially large numbers of certificates are to be stored. In both cases, the communication over randomized frequency channels takes significant time and the overall time needed for the seed establishment is strongly dominated by the inefficient uncoordinated communication under jamming. We note that these problems are not solved by one-way communication or broadcast scenarios [20], where certified public keys are also required for digital signature and authentication purposes.

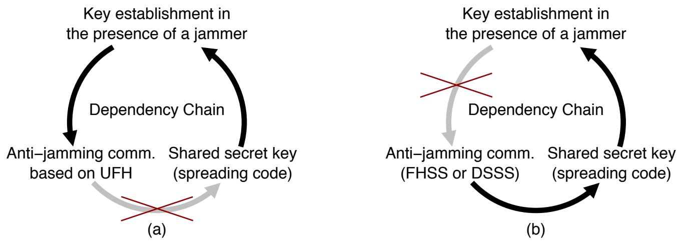

Key establishment in the presence of a jammer
Dependency Chain
Anti–jamming comm. based on UFH
Shared secret key (spreading code)
(a)
Key establishment in the presence of a jammer
Dependency Chain
Anti–jamming comm. (FHSS or DSSS)
Shared secret key (spreading code)
(b)

Figure 1. Positioning of this work. (a) Existing solutions [10], [25], [26] break the anti-jamming key-establishment dependency cycle by not relying on shared secrets. (b) Under the assumption that the communication partners have a-priori knowledge about each other's identities, they can establish a cryptographically strong key without any communication using our ID-based approach, i.e., break the dependency cycle as shown above. This significantly speeds up the setup of shared secrets and faces neither jamming nor interference.

Identity-based cryptography. We investigate the use of ID-based cryptography for settings in which the communication partners know each other's identity. In these settings, ID-based cryptography allows us to construct a seed establishment scheme that does not need any prior communication—therefore the name *zero-communication seed establishment*. This breaks the anti-jamming key-establishment dependency cycle by not relying on communication between the involved parties, as shown in Figure 1 (b). Identity-based schemes allow for dynamic network extensions and communication to new devices without requiring modifications to already established devices.

One particular aspect in the construction of the scheme is how to overcome the static nature of ID-based established seeds. We solve this by introducing time in the seed establishment. As a consequence, two consecutively-generated seeds will be distinct. In particular, we will show that a seed for a given time is uncorrelated to all previously-established seeds.

We note that seed establishment without communication is not a property that can exclusively be achieved by means of identity-based cryptography. Since ID-based cryptography significantly reduces the storage requirements (identities require much less space than identities together with the corresponding public key and associated certificate), and the sharing of identities requires one order of magnitude fewer communication, it, however, constitutes an ideal choice.

In the setting where the communication partners do not know each other's identity, ID-based schemes still remove the necessity to store and transmit public keys and certificates because only the identity of the communication partner must be known before the payload exchange can take place [3]. Since the communication time is the major delay in anti-jamming key-establishment scenarios, ID-based schemes speed up the key-establishment process by one order of magnitude: while the exchanged certificates in UFHSS or UDSSS are several hundreds and sometimes thousands of bytes in size [26], the identities required in our setting are merely a few tens of bytes in size. If the identities need to be exchanged, this approach can be categorized as shown in Figure 1 (a).

III. Model and Notation

We next describe the system and attacker models that our scheme is based on, followed by a description of the notation that we use while detailing the scheme in Section IV.

System Model. We consider the following identity-based setting: two devices want to communicate with each other and know the identity of the communication partner. This identity consists of a unique name or string, such as an e-mail address, a MAC address, or any other unique identifier. This identifier may be publicly known and does not need to be kept secret. We do not assume any shared cryptographic material between the two parties. The advantage of storing the identity of the communication partner instead of storing shared pairwise secrets is that the identity is public, long-term, and typically rather short. More precisely, the identity usually does not change and it may even be possibly to dynamically derive it or to share it by public out-of-band channels. We let the devices have loosely synchronized clocks, e.g., via GPS, such that they can agree on overlapping time frames. Moreover, we assume that all information (on groups, generator elements, pairing, etc.) of an identity-based cryptographic scheme is public and shared among all communication parties.

Furthermore, we require the existence of a trusted central private key generator (PKG), which generates for every party A with the unique identifier ID_A a secret key sk_A. We assume that every participating party is aware of its identity and of her secret key sk_A. Typically, the distribution is done in a set-up phase where every party receives her secret key with the help of the trusted PKG. In practice, this can be achieved by the device manufacturer or by an initial registration step conducted at the commissioning of a device.

Threat Model. We grant the attacker the capabilities to record and eavesdrop on the communication between the communication parties. In particular, the attacker may use a broadband receiver or multiple partial band receivers for this purpose. We, however, impose the standard restriction that (coordinated) FHSS communication could be used to counter jamming attacks if keys were securely shared. This implies that the attacker is not powerful enough to perform a broadband jamming attack; in other words, we assume that the frequency spectrum is sufficiently large such that the attacker cannot block the complete transmission spectrum.

Furthermore, we let the attacker dynamically corrupt and in turn impersonate devices. More precisely, the attacker can corrupt any device at any time. Once a device is corrupted, the attacker obtains all cryptographic material stored on that device, thus enabling the attacker to act on behalf of that device.

Notation. We describe a setting with a symmetric pairing. In Appendix A, we detail the setting with an asymmetric pairing. Following the setup of Boneh and Franklin's identity-based encryption scheme [2], we use two groups $\mathbb{G}_1$ and $\mathbb{G}_T$ of prime order q and a symmetric bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$. By g we denote the designated generator of $\mathbb{G}_1$. In the following, we use the convention that operations in the subgroup $\mathbb{G}_1$ of the elliptic curve are written in additive notation (i.e., we write $a \cdot g$ and $g + g$ rather than g^a and $g \cdot g$, respectively) and operations in the target group $\mathbb{G}_T$ are written in the usual multiplicative form.

We let $h : \{0,1\}^* \to \mathbb{G}_1^*$ denote a cryptographic hash function that takes as input arbitrary strings and maps them to $\mathbb{G}_1^*$. Throughout the paper, we use the usual convention that $\mathbb{G}^* := \mathbb{G} \setminus \{0\}$, where 0 is the neutral element of the group $\mathbb{G}$ w.r.t. the group operation. We write $e \leftarrow_R \mathbb{G}$ to denote that the element e is drawn uniformly from the group $\mathbb{G}$.

The public key pk_A of a communication party A with name ID_A is $\mathrm{pk}_A := h(\mathrm{ID}_A) = a \cdot g$, i.e., $a := \mathrm{dlog}_g(\mathrm{pk}_A)$ is the discrete logarithm of pk_A with respect to the base g. In general, it is infeasible to compute a; we only use the existence of a to derive and show equalities between elements. The PKG computes A's secret key as $\mathrm{sk}_A := s \cdot \mathrm{pk}_A = sa \cdot g$, where $s \leftarrow_R \mathbb{Z}_q^*$ is the PKG's master key.

IV. Our Zero-Communication Seed Establishment Scheme

Our scheme establishes a cryptographically strong seed for anti-jamming techniques, such as FHSS or DSSS without any communication by using techniques from identity-based encryptions [2]. We first give an overview of our seed establishment (Section IV-A) and then present its details (Section IV-B).

A. *Overview*

Based on the ID-based setup and the communication parties knowledge of their own secret keys (sk_A for party A), the scheme that we propose allows two communication parties to output a cryptographically strong seed, i.e., a seed that is computationally indistinguishable from a randomly chosen seed. We consider the case where the parties' secret keys are solely used for the seed establishment.

The basic idea for the seed generation for two parties A and B is standard and in its core looks as follows:

$$e(\mathrm{sk}_A, \mathrm{pk}_B),$$

where sk_A is A's secret key, pk_B is B's public key, $e(\cdot,\cdot)$ is the bilinear pairing, s is the PKG's master key, g is the group generator, and a and b are given by A's and B's public key, respectively. Using the properties of the bilinear map, we derive that

$$e(\mathrm{sk}_A, \mathrm{pk}_B) = e(as \cdot g, b \cdot g) = e(g,g)^{abs}.$$

This simple seed generation has the serious drawback that only one unique seed can be generated per pair of devices. Although this seed already has strong cryptographic properties, it is insufficient for our applications scenario where devices keep their identities for long times and re-keying is cumbersome or even impossible. Being able to generate only one seed is particularly problematic because broadband attackers may be able to listen on all FHSS frequencies and can thus learn large fractions of the pseudo-random sequence. If, in future communication attempts, the communication partners keep on reusing the same seed, the attacker can successfully jam the communication by jamming the previously-learned frequency sequence.

One solution is to use a fresh nonce for every seed [3]. Such an approach, however, inherently requires communication. Another remedy is to let each device store a state for every communication partner it interacted with, e.g., the state of the pseudo-random generator. In both solutions, two communicating parties do not re-use frequencies and an attacker cannot gain an advantage by eavesdropping and storing the used communication frequencies. Storing a state for every peer, however, is expensive and would destroy the storage advantage gained by deploying ID-based cryptography.

We choose storage over communication but significantly relax the storage requirements and only stipulate that devices store a state for every communication peer for a certain short time frame, e.g., two hours. In this way, within a time frame, the frequencies derived are pseudo-random. Across two time frames, our seed establishment scheme ensures that the seeds derived in different time frames are uncorrelated, i.e., information obtained in one time frame cannot be used to obtain information in a future time frame.

For A and B to agree on a time frame tf, we exploit loosely synchronized clocks. The time frames may vary in length and depend on the precision of the synchronicity of the devices' clocks and on the storage available on the devices. It is, however, crucial that the time frames are absolute (such as "January 24, 2014, 2 p.m. - 4 p.m., UTC+1") and not periodic (such as "today, 2 p.m. - 4 p.m."): periodically reoccurring time frames cause established seeds to be equal and, therefore, enable an attacker that recorded previously-used frequencies to mount a successful jamming attack. For reasons of simplicity, we make the time frame enumerable, i.e., there is an initial time frame and for every time frame tf there is a successor $\mathit{tf}+1$.

Inspired by Ryu et al. [21], a first approach might be to derive the shared seed as $e(\mathrm{sk}_A, \mathrm{pk}_B)^{f(\mathit{tf})}$ for two parties A and B, and a time frame tf where f is a mapping from strings to exponents. While this approach can be modified to derive keys that provide forward-secrecy, these seeds are insecure in our setting because we assume that the attacker can learn all seeds: merely using $e(\mathrm{sk}_A, \mathrm{pk}_B)^{f(\mathit{tf})}$ allows the attacker to compute $e(\mathrm{sk}_A, \mathrm{pk}_B)$, since f, tf, and the prime group order are publicly known and an attacker can simply compute the tf-th root of $e(\mathrm{sk}_A, \mathrm{pk}_B)^{f(\mathit{tf})}$. As a result, this approach allows the attacker to predict seeds that are used in future communications.

We remedy this problem and propose the following seed establishment protocol. It combines the efficiency of the approach by Ryu et al. and, at the same time, prevents attackers from deriving future seeds. The core idea is to use a second cryptographic hash function[1] $f : \{0,1\}^* \to \{0,1\}^{2\eta}$ and compute the seed

$$\mathcal{S}_{(A,B)} := f(e(\mathrm{sk}_A, \mathrm{pk}_B), \mathit{tf}) = f(e(g,g)^{abs}, \mathit{tf})$$

as the established shared session seed, where η denotes the security parameter and $f(x,y)$ denotes the application of f to the standard encoding of the pair that contains x and y.

[1]To achieve η bit security, we choose the output length to be 2η. This accommodates the general attack against hash functions based on the birthday paradox.

B. The Seed Establishment Scheme

Formally, let GGen be a *group generator* algorithm that takes as input a security parameter η and draws a randomly chosen generator for a group $\mathbb{G}_1$ of prime order q in order of 2^η, a description of a non-degenerate bilinear mapping $e : \mathbb{G}_1 \to \mathbb{G}_T$, and two hash functions $f : \{0,1\}^* \to \{0,1\}^{2\eta}$ and $h : \{0,1\}^* \to \mathbb{G}_1$ such that $e(g,g)$ is a generator for a multiplicative group $\mathbb{G}_T$ for the designated generator g. Note that $\mathbb{G}_T$ is also of prime-order q. Then GGen outputs $\langle q, g, e, f, h \rangle$.

The PKG's setup phase algorithm $\mathrm{MGen}(1^\eta)$. The master key generation algorithm MGen takes as input a security parameter η. $\mathrm{MGen}(1^\eta)$ runs the group generator algorithm GGen obtaining $\langle q, g, e, f, h \rangle$ and chooses a random master key $s \in \mathbb{Z}_q^*$ uniformly at random. Let $\mathit{par} := \langle q, g, e, f, h \rangle$ be the global parameters. Then, $\mathrm{MGen}(1^\eta)$ outputs $\langle \mathit{par}, s \rangle$.

A user's setup algorithm $\mathrm{Gen}(\mathit{par}, \mathrm{ID}_A, s)$. The setup algorithm Gen for a user A takes as input the global parameters par, the user's unique name ID_A, and the master secret key s. $\mathrm{Gen}(\mathit{par}, \mathrm{ID}_A, s)$ outputs $\mathrm{sk}_A := s \cdot \mathrm{pk}_A = s \cdot h(\mathrm{ID}_A) \in \mathbb{G}_1^*$.

The seed establishment procedure $\mathrm{Seed}(\mathit{par}, \mathrm{sk}_A, \mathrm{ID}_B, \mathit{tf})$. The seed establishment algorithm Seed run by user A to establish a seed with user B takes as input the global parameters par, the secret key of the caller sk_A, the unique name ID_B of the peer B, and the time frame tf. $\mathrm{Seed}(\mathit{par}, \mathrm{sk}_A, \mathrm{ID}_B, \mathit{tf})$ outputs $f(e(\mathrm{sk}_A, \mathrm{pk}_B), \mathit{tf})$.

For the sake of illustration, we depict the correctness of our construction and show that A and B derive a shared seed:

$$\begin{aligned}\mathcal{S}_{(A,B)} &:= f(e(\mathrm{sk}_A, \mathrm{pk}_B), \mathit{tf}) = f(e(sa \cdot g, b \cdot g), \mathit{tf}) \\ &= f(e(g,g)^{abs}, \mathit{tf}) = f(e(sb \cdot g, a \cdot g), \mathit{tf}) \\ &= f(e(\mathrm{sk}_B, \mathrm{pk}_A)), \mathit{tf}) =: \mathcal{S}_{(B,A)}.\end{aligned}$$

Enabling duplex communication. Duplex communication can be enabled by using different seeds for the two directions. The idea is to encode the direction as input to the outer hash function f: let A and B be the communicating parties. Then, we use for the communication from A to B the key $\mathcal{S}^{\rightarrow}_{(A,B)} := f(e(\mathrm{sk}_A, \mathrm{pk}_B), \mathit{tf}, \mathrm{ID}_A, \mathrm{ID}_B)$ and for the communication from B to A the seed $\mathcal{S}^{\rightarrow}_{(B,A)} := f(e(\mathrm{sk}_B, \mathrm{pk}_A), \mathit{tf}, \mathrm{ID}_B, \mathrm{ID}_A)$.

V. Security Analysis

In this section, we characterize the security of seed establishment schemes by introducing the notion of a *secure seed establishment scheme for anti-jamming methods*. We first informally describe the attacker model and then formally define the challenger; this definition precisely describes how we formalize the attacker for our threat model from Section III. Thereafter, we show that the scheme described in Section IV indeed constitutes a secure seed establishment scheme.

```
upon (initialize)
    $\langle par, s\rangle \leftarrow \mathrm{MGen}(1^\eta)$;
    $tf := 0$;
    send $par$ to $\mathcal{E}$

upon (nextTimeframe)
    $tf := tf + 1$;
    send $tf$ to $\mathcal{E}$

upon (initialize, $\mathrm{ID}_P$)
    if $registered[\mathrm{ID}_P] = \perp$ then
        $\mathrm{sk}_P \leftarrow \mathrm{Gen}(par, s, \mathrm{ID}_P)$;
        set $registered[\mathrm{ID}_P] := 1$
        send ready to $\mathcal{E}$

upon (compromise, $\mathrm{ID}_P$)
    if $\mathrm{ID}_P \notin challengeSet$ then
        $c[\mathrm{ID}_P] := 1$;
        send $\mathrm{sk}_P$ to $\mathcal{E}$

upon (computeSeed, $\mathrm{ID}_P, \mathrm{ID}_Q$)
    if $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf] = \perp$ then
        $d_{tf} \leftarrow \mathrm{Seed}(par, \mathrm{sk}_P, \mathrm{ID}_Q, tf)$
        $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf] := d_{tf}$
    send $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf]$ to $\mathcal{E}$

upon (challenge, $\mathrm{ID}_P, \mathrm{ID}_Q$)
    set $challengeSet := \{\mathrm{ID}_P, \mathrm{ID}_Q\}$
    if $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf] = \perp \wedge$
            $c[\mathrm{ID}_P] = \perp \ \wedge \ c[\mathrm{ID}_Q] \perp$
    then
        if $b = 1$ then
            $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf] \leftarrow_R \mathbb{G}_T$
        else
            $d_{tf} \leftarrow \mathrm{Seed}(par, \mathrm{sk}_P, \mathrm{ID}_Q, tf)$
            $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf] := d_{tf}$
        send $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf]$ to $\mathcal{E}$
```

The actions compromise, computeSeed, and challenge require that the involved parties ID_P and, if applicable, ID_Q are registered. For the sake of readability, we do not explicitly write these two **if** statements into the code.

Figure 2. Pseudo-randomness seed challenger $\mathrm{PRSCh}_b(1^\eta)$

A. Preliminary Remarks

Intuitively, we use a standard reduction technique to show that any attacker that wins against the challenger can be used to break the well-known decisional bilinear Diffie-Hellman problem. Since we assume that this problem is hard, we conclude that no attacker exists.

Our security notion for a seed establishment closely resembles the security notion of a secure key exchange in the eCK model [13]. On a high level, the main difference to the eCK model is that we do not allow any communication between the participating parties. In the spirit of the eCK model, we introduce the notion of a *session*. One session corresponds to one time frame. Our security definition for the seed establishment requires that the established seed is pseudo-random. More precisely, we require that the seeds of one session are indistinguishable from randomly chosen bit strings, in particular the seeds should be unpredictable based on past sessions.

Here and throughout the remainder of this paper, we use the convention that variables such as *seed* are written in lower-case letters. Some variables are parameterized, for instance, by user identifier or by the time frame. We use square brackets to denote the parameters, e.g., $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf]$. Initially, all variables for all parameters are set to $\perp$. Functions are denoted by capitalized strings with the usual parenthesis notation, e.g., $\mathrm{Seed}(par, \mathrm{sk}_P, \mathrm{ID}_Q, tf)$.

B. The Notion of a Secure Seed Establishment

We consider a probabilistic and polynomially bounded (ppt) attacker $\mathcal{E}$. The attacker determines the number of parties involved in the game. This number can be dynamically changed. We grant the attacker the knowledge of all seeds that were established in the system, i.e., all seeds from all concurrent sessions. We stress that this is an unrealistically strong attacker: a real-life attacker is able to use a broadband receiver to determine the frequency sequence used in a communication. Our attacker, however, not only knows all these sequences but also the input to the pseudo-random generator that resulted in these sequences. Furthermore, we grant the attacker the power to arbitrarily compromise users at any point in time. These users will immediately release their secret key. The only exception are the users that are challenged by the attacker. These users cannot be compromised since this trivially breaks any security property.

Challenger definition. In Figure 2, we formally define the seed challenger $\mathrm{PRSCh}_b(1^\eta)$ against the pseudo-randomness of the seed establishment scheme. The attacker can start the game by sending (initialize) and the attacker can advance to the next time frame by sending (nextTimeframe). We let the attacker dynamically decide at which point in time an honest party P performs its initial setup phase by sending (initialize, ID_P).

The variable $registered[\mathrm{ID}_P]$ keeps track of parties that finished their setup phase. By sending (compromise, ID_P), the attacker can adaptively compromise a party P. The variable $c[\mathrm{ID}_P]$ keeps track of compromised parties. Compromising a party is only possible if the attacker has not previously issued a challenge involving that party.

The attacker can let parties P and Q compute the seed for the current time frame by sending (computeSeed, $\mathrm{ID}_P, \mathrm{ID}_Q$). The newly established seed is immediately send to the attacker. Finally, the attacker can request a challenge seed for two honest (i.e., non-compromised) parties P and Q of her choice by sending (challenge, $\mathrm{ID}_P, \mathrm{ID}_Q$). At this point, the parameter

b of the challenger comes into play: if $b = 0$, i.e., if the challenger is in the pseudo-randomness game, the challenger sends the correctly computed seed; otherwise, $b = 1$, i.e., if the challenger is running the random game, and the challenger sends a value chosen uniformly at random. The attacker wins if she is able to determine the bit b with non-negligible probability.

C. Security proof

In the following, we show that the attacker $\mathcal{E}$ cannot win in the challenge defined above. In the following, we let $\mathcal{E}(1^\eta)^M$ denote the execution of $\mathcal{E}(1^\eta)$ with oracle access to the machine M, i.e., can send requests to M and receives responses from M.

Definition 1: Let PRSCh be defined as in Figure 2. We say that a tuple of ppt algorithms $(\mathrm{GGen}, \mathrm{MGen}, \mathrm{Seed})$ is a *secure seed establishment scheme* if for all ppt attackers $\mathcal{E}$ the following difference is negligible in η:

$$|\Pr[z \leftarrow \mathcal{E}(1^\eta)^{\mathrm{PRSCh}_0(1^\eta)} : z = 1] - \Pr[z \leftarrow \mathcal{E}(1^\eta)^{\mathrm{PRSCh}_1(1^\eta)} : z = 1]|.$$

We base the security of our scheme on the commonly used bilinear version of the decisional Diffie-Hellmann problem.

The decisional bilinear Diffie-Hellmann (DBDH) problem. Let GGen be a group generator algorithm for $\mathbb{G}_1$ as described in Section IV-B. We say that the *DBDH problem is hard for* GGen if for every pair $(\mathrm{ID}_A, \mathrm{ID}_B) \in \{0,1\}^*$ and every probabilistic, polynomial-time (ppt) attacker $\mathcal{E}$ the following difference is negligible in η:

$$\begin{aligned} &|\Pr[\langle q, g, e, f, h\rangle, \leftarrow \mathrm{GGen}(1^\eta), a, b, s \leftarrow \mathbb{Z}_q^* \\ &\quad z \leftarrow \mathcal{E}(1^\eta, q, g, e, a \cdot g, b \cdot g, s \cdot g, e(g,g)^{abs}) : z = 1] \\ &- \Pr[\langle q, g, e, f, h\rangle, \leftarrow \mathrm{GGen}(1^\eta), a, b, s, r \leftarrow \mathbb{Z}_q^*, \\ &\quad z \leftarrow \mathcal{E}(1^\eta, q, g, e, a \cdot g, b \cdot g, s \cdot g, e(g,g)^{r}) : z = 1]|. \end{aligned}$$

Theorem 1: Let GGen be the group generator algorithm as defined in Section IV-B. If $h : \mathbb{G}_1 \to \mathbb{G}_T$ and $f : \{0,1\}^* \to \{0,1\}^\eta$ are random oracles and if DBDH is hard for GGen, then $(\mathrm{GGen}, \mathrm{MGen}, \mathrm{Seed})$ is a secure seed establishment scheme.

Proof outline. The full proof shows by contraposition that $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_0(1^\eta)}$ and $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_1(1^\eta)}$ are indistinguishable. For every attacker $\mathcal{E}$, we construct a reduction M that breaks the DBDH problem with non-negligible probability if $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_0(1^\eta)}$ and $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_1(1^\eta)}$ are distinguishable with non-negligible probability, i.e., if the following difference is non-negligible:

$$|\Pr[z \leftarrow \mathcal{E}(1^\eta)^{\mathrm{PRSCh}_0(1^\eta)} : z = 1] - \Pr[z \leftarrow \mathcal{E}(1^\eta)^{\mathrm{PRSCh}_1(1^\eta)} : z = 1]|.$$

The full proof can be found in Appendix B.

VI. Experimental Evaluation

We implemented our seed establishment scheme as detailed in Section IV-B and conducted an experimental evaluation to investigate the practical feasibility of our approach.

A. Prototype Implementation

The implementation was done in Java since Java runs on virtually all commodity personal computers and on Android embedded devices, thus conveniently allowing us to study the efficiency with only one implementation. For the mathematical operations, we relied on the jPBC library [4]. This library is written entirely in Java without resorting to external native-code libraries for speedup. As hash function to compute the seed, we use SHA-256 as provided by the standard Java API via the `MessageDigest` class. For the time frames, we conveniently used the current system time, accessible via the Java API call `System.currentTimeMillis`. Of course, a concrete implementation would use a more coarse-grained time resolution. Since this does not affect the seed computation time we omitted the code that computes such time frames in our implementation.

The implementation is general and does not exploit any form of parallelism (further discussed below). The source code and an executable Android APK package and an executable Java archive are freely available [17].

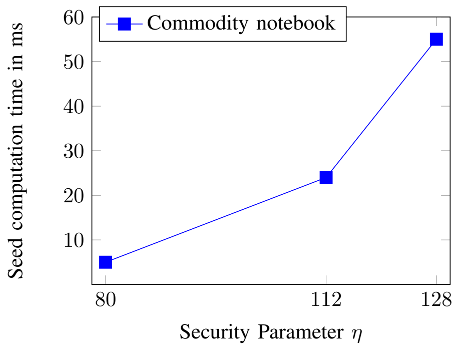
B. Experimental Evaluation

We conducted an experimental evaluation of the implementation. We let the implementation run on a commodity notebook with a 3.3 GHz quad-core processor and 4 GB RAM, on a Galaxy Nexus phone that is equipped with a 1.2 GHz dual-core processor and 1 GB RAM, and on a Galaxy Note 10.1 tablet that is outfitted with a 1.4 GHz quad-core processor and 2 GB RAM. On each device, we used multiple security parameters to show that our scheme is also feasible for larger, more secure elliptic curves. We used elliptic curves that offer a security parameter of 80 bits, 112 bits, and 128 bits. More precisely, we used symmetric TYPE-I curves [7] with an embedding degree of $k = 2$ that are coined Type-A curves in jPBC [4]. The curve parameters are specified in Table I. We selected a symmetric pairing.[2] In the following, we shed more light on the selection of the curve parameters.

Since we rely on the discrete logarithm problem for the security of our scheme, we require that this problem is at least as hard in the target group $\mathbb{G}_T$ as it is in $\mathbb{G}_1$. The group $\mathbb{G}_1$ is an r bit prime-order subgroup of an elliptic curve $E(\mathbb{F}_q)$ over the finite field $\mathbb{F}_q$, where q is a p bit prime. Since the embedding degree of our curves is $k = 2$, $\mathbb{G}_T$ is a subgroup of the field extension $\mathbb{F}_{q^2}$. Koblitz and Menezes [12] recommend that for achieving a security parameter η, the field extension $\mathbb{F}_{q^k}$ should be as large as RSA groups with that security parameter and that r should equal 2η. Intuitively, the algorithms for attacking RSA are as sophisticated as those for attacking the discrete logarithm problem in r bit subgroups of $E(\mathbb{F}_q)$ and in $\mathbb{F}_{q^k}$.

We applied these recommendations and derive the curves as follows: we used the curve provided with jPBC that offers 80 bit security (called "`a.param`") and we generated curves with 112 bit and with 128 bit security ourselves using the jPBC

[2] Asymmetric pairings may result in a higher embedding degree and, as a consequence, smaller curves and potentially better performance (see Appendix A). We are, however, not aware of a library with support for asymmetric pairings on Type-II curves that runs on commodity hardware as well as embedded devices out of the box.



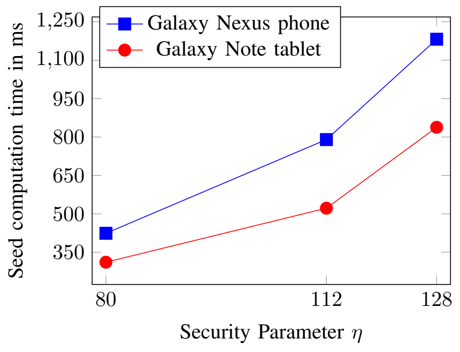


Figure 3. Experimental evaluation.

RSA	η	r	p	$\lvert\mathbb{F}_{q^2}\rvert$
1024	80	160	512	1024
2048	112	229	1099	2198
3072	128	260	1599	3198

Table I. ELLIPTIC CURVE PARAMETERS USED IN THE PROTOTYPE IMPLEMENTATION.

function `TypeACurveGenerator(r, p)`. Since the curve generation algorithm sometimes deviates from the desired parameters, we added a margin to guarantee at least the desired group sizes. Table I summarizes the RSA group sizes for the security parameter η, the bit size r of $\mathbb{G}_1$, the bit size p of the base field $\mathbb{F}_q$ of the elliptic curve, and the bit size of the field extension $\mathbb{F}_{q^2}$. For more details and explanations, we refer the reader to extensive literature on the topic [2], [7], [12], [21].

For each of the three curves, we evaluated the schemes as follows: we randomly chose PKG secret keys. For each such key, we set up the public keys for two parties A and B and we ran the seed establishment scheme 100 times, each time with a different time frame. We report the average time that it took party A and B to compute the seed in Figure 3.

C. Results and Discussion

Figure 3 shows that the seed establishment protocol is fast enough to be used on commodity hardware as well as embedded devices. On a normal notebook, the computation of the seed constitutes only a very minor overhead: it takes less than 55 ms even for the highest security parameter. On the two Android devices, the time required for the seed computation ranges between 311 ms to 838 ms and 424 ms to 1182 ms for the quad-core tablet and the dual-core phone, respectively.

The graphs indicate that the computation time does not grow linearly in the security parameter. The reason is that we use elliptic curves that are defined over finite fields and finite field extensions. The operations in these finite fields have a super-linear complexity in the length of the security parameter. This super-linear complexity of the basic operations is reflected in the computation time of the pairing.

Even though the embedded devices used in our evaluation are equipped with powerful CPUs, they show a significantly worse performance for the following reasons: jPBC internally uses the BigInteger class. This class uses normal registers to compute arbitrary-precision modular operations. Since the CPUs in both handheld devices are 32 bit processors, these devices have an inherent disadvantage when compared to 64 bit processors of commodity notebooks. Furthermore, a study conducted by Oracle [9] suggests that the Dalvik virtual machine, the back-end virtual machine that runs the Android applications, is slower than native Java VMs.

Generality and parallelism. The implementation uses the jPBC library out of the box. This implementation is written to be portable and consequently does not exploit any optimized number theory library. Even though jPBC is thread-safe, our implementation cannot exploit parallelism because the implementation consists of two sequentially-dependent jPBC API calls. A specialized, more low-level implementation potentially improves efficiency by actively exploiting the features of a multi-core architecture.

VII. RELATED WORK

Identity-based cryptography. The identity-based cryptography paradigm was initiated by Adi Shamir [23]. Since then, an impressive corpus of literature on different kinds of identity-based cryptography techniques has been published. The work closest to ours are pairing-based key establishment schemes (e.g., [5], [11], [21], [22]). In all of these schemes, the established key k is used to directly encrypt or directly mac the subsequent communication. In particular, the established key k must be strong enough to provide forward secrecy or unconditional anonymity, i.e., even if the long-term keys used to establish k are broken in the future, k is still secure. In our case, however, we do not need that seeds remain secret in the future or that seeds remain secure if one of the two parties is compromised. We only require that two consecutively generated seeds are different. Therefore, we can resort to the conceptually much simpler and computationally much more efficient solution based on time-frames and hash functions. After establishing the seed to enable a jamming-resistant communication, any key establishment protocol can be used to establish a session key to secure the successive communication.

Our seed establishment scheme is inspired by the key setup of the encryption scheme by Boneh and Franklin [2]. Our approach, however, significantly deviates from their approach since they build an asymmetric encryption scheme while we aim at establishing a shared seed in a non-interactive manner.

The concept of combining strings and cryptographic material as input to hash functions has, for instance, been explored in the form of service-specific pseudonyms [16], of domain pseudonyms [8], and of scope-exclusive pseudonyms [1]. The aim of all of these approaches is to create pseudonyms that are linkable within the context described by the string but unlinkable outside of that context. We do not establish pseudonyms. In fact, the identity of two communication parties can be publicly known. Instead, we focus on establishing cryptographically strong seeds and we enforce that consecutively established seeds differ.

Seed establishment and non-interactive key establishment. Our security notion for a seed establishment closely resembles the security notion of a secure key exchange in the eCK model [13]. The main difference to the eCK model is that our goal is to abstain from any communication between the participating parties and that we do not require forward secrecy. Even though a seed establishment scheme for anti-jamming techniques does not need to provide forward secrecy, an attacker should not be able to mount replay attacks. We introduce time frames and require that even for the same peer the seed establishment scheme produces a completely uncorrelated seed for every time frame.

On the theory side, non-interactive key establishment has recently been re-investigated by [6]. In this work, the authors provide security models for this primitive and explore their relationships. On the more practical side, the use of identity-based key establishment protocols for wireless ad-hoc communication in adversarial settings has recently been modeled by Capar *et al.* [3]. The authors analyze the communication and energy costs of existing security protocols in wireless settings and propose a protocol that reduces both costs. In contrast to our scheme, their protocol relies on transmitting a nonce for guaranteeing the freshness of a seed, which make zero-communication seed establishment impossible. Moreover, they do not consider specific implementation details for realizing identity-based key establishment.

Keyless spread-spectrum communication. Apart of the mentioned works on spread-spectrum communication without shared secrets such as [25], [26] for FHSS-based solutions and [10], [19], [24] for DSSS-based solutions, a number of further schemes have been proposed to improve their efficiency and applicability. To improve the robustness of keyless DSSS-based schemes to reactive jammers, delayed seed-disclosure [14] and randomized differential DSSS [15] were proposed. In [28], Wang *et al.* model UFHSS transmissions as a multi-armed bandid problem and derive optimality results for adaptive UFHSS. In [29], Xiao *et al.* take the pairwise problem to network (multi-hop) settings and propose solutions for collaborative broadcast. All these proposals focus on the communication side and how to make the reception of messages more efficient. They do not improve on the required length of messages that must be exchanged for establishing a shared secret (or seed) as we do in this paper. This idea might seem simple but it can nevertheless be very effective.

A general problem in this context (not specific to this paper) is how a receiver will know that it should be receiving messages. In scenarios where messages are sent regularly, this can be predicted by the receiver. Other settings require out-of-band channels or the receiver's initiative when all communication is suddenly lost (due to jamming attacks).

VIII. Conclusion

In this paper, we presented a zero-communication seed establishment scheme that derives cryptographically strong seeds for anti-jamming spread spectrum methods. Once derived, these seeds are used as input to a pseudo-random generator that in turn is used to create a high-throughput, jamming-resistant channel. The proposed scheme combines powerful identity-based cryptographic methods with sophisticated anti-jamming techniques to facilitate fast and reliable wireless communication in an adversarial setting. Cryptographically strong seeds are derived by exploiting a-priori knowledge of the communication partners. Even if no a-priori knowledge about the partners is available, our method significantly reduces the required communication over low-throughput channels by one order of magnitude.

We proved that our scheme derives cryptographically strong seeds. In particular, the derived seeds are computationally indistinguishable from random numbers against a very strong attacker that knows all established seeds and that can dynamically create and corrupt parties. More precisely, we formally defined the notion of a secure seed establishment scheme for anti-jamming methods and we showed that our scheme satisfies this notion in the presence of such a strong attacker. Furthermore, we also conducted an experimental evaluation to show the feasibility of our approach. The results confirm that the seed establishment is efficient enough to run on commodity mobile hardware such as notebooks, tablet computers, and smart phones. An optimized implementation that exploits hardware-specific features such as the readily-available multi-core architectures of these devices can be expected to improve the efficiency further.

Acknowledgements.
This work was supported by the German research foundation (DFG) through the Emmy Noether program and the Cluster of Excellence on Multimodal Computing and Interaction (MMCI), and by the German Federal Ministry of Education and Research (BMBF) through funding for the Center for IT-Security, Privacy and Accountability (CISPA). We thank the reviewers for their helpful comments.

References

[1] "Attribute-based Credentials for Trust EU Project," https://abc4trust.eu/.

[2] D. Boneh and M. Franklin, "Identity-Based-Encryption from the Weil Pairing," *SIAM Journal on Computing*, vol. 32, no. 3, pp. 586–615, 2003.

[3] C. Capar, D. Goeckel, K. G. Paterson, E. A. Quaglia, D. Towsley, and M. Zafer, "Signal-flow-based analysis of wireless security protocols," *Information and Computation*, vol. 226, pp. 37–56, 2013.

[4] A. De Caro, "jPBC Library," http://libeccio.dia.unisa.it/projects/jpbc/downsload.html.

[5] R. Dupont and A. Enge, "Provably secure non-interactive key distribution based on pairings," *Discrete Applied Mathematics*, vol. 154, no. 2, pp. 270–276, Feb. 2006.

[6] E. S. V. Freire, D. Hofheinz, E. Kiltz, and K. G. Paterson, "Non-interactive key exchange," in *Public Key Cryptography*, 2013, pp. 254–271.

[7] S. D. Galbraith, K. G. Paterson, and N. P. Smart, "Pairings for Cryptographers," *Discrete Applied Mathematics*, vol. 156, no. 16, pp. 3113–3121, Sep. 2008.

[8] "Identity Mixer," http://idemix.wordpress.com/.

[9] "Java SE Embedded Performance Versus Android 2.2," https://blogs.oracle.com/javaseembedded/entry/how_does_android_22s_performance_stack_up_against_java_se_embedded, accessed March 21, 2013.

[10] T. Jin, G. Noubir, and B. Thapa, "Zero pre-shared secret key establishment in the presence of jammers," in *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*. ACM, 2009, pp. 219–228.

[11] A. Kate, G. M. Zaverucha, and I. Goldberg, "Pairing-Based Onion Routing," in *Proc. Privacy Enhancing Technologies Symposium (PETS'07)*, ser. Lecture Notes in Computer Science, vol. 4776. Springer-Verlag, 2007, pp. 95–112.

[12] N. Koblitz and A. Menezes, "Pairing-based Cryptography at High Security Levels," pp. 13–36, 2005.

[13] B. LaMacchia, K. Lauter, and A. Mityagin, "Stronger security of authenticated key exchange," in *Proceedings of the 1st international conference on Provable security*, ser. ProvSec'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 1–16.

[14] A. Liu, P. Ning, H. Dai, Y. Liu, and C. Wang, "Defending DSSS-based Broadcast Communication against Insider Jammers via Delayed Seed-Disclosure," in *Proc. of the Computer Security Applications Conference*. ACM, 2010, pp. 367–376.

[15] Y. Liu, P. Ning, H. Dai, and A. Liu, "Randomized differential DSSS: Jamming-resistant wireless broadcast communication," in *Proceedings of INFOCOM*. IEEE, 2010, pp. 1–9.

[16] M. Maffei, K. Pecina, and M. Reinert, "Security and privacy by declarative design," in *Proc. IEEE Symposium on Computer Security Foundations (CSF'13)*, 2013, pp. 81–96.

[17] K. Pecina, E. Mohammadi, and C. Pöpper, "Project Page: A Zero-Communication Seed Agreement for Anti-Jamming Techniques," http://www.sps.cs.uni-saarland.de/zcaj/.

[18] R. A. Poisel, *Modern Communications Jamming Principles and Techniques*. Artech House Publishers, 2004.

[19] C. Pöpper, M. Strasser, and S. Capkun, "Jamming-resistant broadcast communication without shared keys," in *Proceedings of the 18th USENIX Security Symposium*. The USENIX Association, 2009, pp. 231–247.

[20] C. Pöpper, M. Strasser, and S. Capkun, "Anti-jamming broadcast communication using uncoordinated spread spectrum techniques," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 5, pp. 703–715, 2010.

[21] E.-K. Ryu, E.-J. Yoon, and K.-Y. Yoo, "An Efficient ID-Based Authenticated Key Agreement Protocol from Pairings," in *Proc. Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications (NETWORKING'04)*, ser. Lecture Notes in Computer Science, vol. 3042. Springer-Verlag, 2004, pp. 1458–1463.

[22] R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems based on Pairings," in *Proc. Symposium on Cryptography and Information Security (SCIS'00)*, 2000.

[23] A. Shamir, "Identity-based Cryptosystems and Signature Schemes," in *Proc. Advances in Cryptology (CRYPTO'84)*, ser. Lecture Notes in Computer Science, vol. 7. Springer-Verlag, 1984, pp. 47–53.

[24] D. Slater, P. Tague, R. Poovendran, and B. J. Matt, "A coding-theoretic approach for efficient message verification over insecure channels," in *Proceedings of the ACM Conference on Wireless Network Security (WiSec)*. ACM, 2009, pp. 151–160.

[25] M. Strasser, C. Pöpper, and S. Capkun, "Efficient uncoordinated FHSS anti-jamming communication," in *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*. ACM, 2009, pp. 207–218.

[26] M. Strasser, C. Pöpper, S. Capkun, and M. Cagalj, "Jamming-resistant key establishment using uncoordinated frequency hopping," in *IEEE Symposium on Security and Privacy (S&P)*, 2008, pp. 64–78.

[27] D. Torrieri, *Principles of Spread-Spectrum Communication Systems*, 2nd ed. New York: Springer, 2011.

[28] Q. Wang, P. Xu, K. Ren, and X. Li, "Towards optimal adaptive UFH-based anti-jamming wireless communication," *Journal on Selected Areas in Communications*, vol. 30, no. 1, pp. 16–30, 2012.

[29] L. Xiao, H. Dai, and P. Ning, "Jamming-resistant collaborative broadcast using uncoordinated frequency hopping," *IEEE Transactions on Information Forensics & Security*, vol. 7, no. 1, pp. 297–309, 2012.

[30] K. Xu, Q. Wang, and K. Ren, "Joint UFH and power control for effective wireless anti-jamming communication," in *Proceedings of the IEEE INFOCOM*, 2012, pp. 738 –746.

Appendix

A. Seed Establishment with an Asymmetric Pairing

The seed establishment scheme in a setting with an asymmetric pairing works as follows: the pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \mapsto \mathbb{G}_T$, $\mathbb{G}_1 \neq \mathbb{G}_2$ is defined over a Type-II elliptic curve [7]. These curves have an efficiently-computable isomorphism $\phi : \mathbb{G}_2 \mapsto \mathbb{G}_1$. We require that all values reside in $\mathbb{G}_2$ and the hash function $h : \{0,1\}^* \mapsto \mathbb{G}_2$. The public key and private keys are still computed as before. The seed computation however, becomes slightly more involved:

$$\begin{aligned}\mathcal{S}_{A,B} &:= f(e(\phi(\mathrm{sk}_A), \mathrm{pk}_B), \mathit{tf})\\ \mathcal{S}_{B,A} &:= f(e(\phi(\mathrm{sk}_B), \mathrm{pk}_A), \mathit{tf})\end{aligned}$$

To show the correctness of this computation, i.e., to show that $\mathcal{S}_{A,B} = \mathcal{S}_{B,A}$, we observe that $\phi(\hat{g}) := g$ for some generators $\hat{g}$ of $\mathbb{G}_2$ and g of $\mathbb{G}_1$ and that $\phi(x \cdot \hat{g}) = x \cdot g$ since ϕ is an isomorphism. Furthermore, we recall that $\mathrm{sk}_A = s \cdot \mathrm{pk}_A = s \cdot a \cdot \hat{g}$ and $\mathrm{sk}_B = s \cdot \mathrm{pk}_B = s \cdot b \cdot \hat{g}$ for some a, b, and the PKG secret s. Then

$$\begin{aligned}\mathcal{S}_{A,B} &:= f(e(\phi(\mathrm{sk}_A), \mathrm{pk}_B), \mathit{tf})\\ &= f(e(\phi(s \cdot a \cdot \hat{g}), b \cdot \hat{g}), \mathit{tf})\\ &= f(e(s \cdot a \cdot \phi(\hat{g}), b \cdot \hat{g}), \mathit{tf})\\ &= f(e(\phi(\hat{g}), \hat{g})^{s \cdot a \cdot b}, \mathit{tf})\\ &= f(e(s \cdot b \cdot \phi(\hat{g}), a \cdot \hat{g}), \mathit{tf})\\ &= f(e(\phi(s \cdot b \cdot \hat{g}), a \cdot \hat{g}), \mathit{tf})\\ &= f(e(\phi(\mathrm{sk}_B), \mathrm{pk}_A), \mathit{tf})\\ &=: \mathcal{S}_{B,A}\end{aligned}$$

B. Full Proof of Theorem 1

Theorem 1. *Let* GGen *be the group generator algorithm as above. If $h : \mathbb{G}_1 \to \mathbb{G}_T$ and $f : \{0,1\}^* \to \{0,1\}^\eta$ are random oracles and if DBDH is hard for* GGen*, then for all ppt attackers $\mathcal{E}$ the following difference is negligible in η:*

$$\begin{aligned}&|\Pr[z \leftarrow \mathcal{E}(1^\eta)^{\mathrm{PRSCh}_0(1^\eta)} : z = 1] -\\ &\Pr[z \leftarrow \mathcal{E}(1^\eta)^{\mathrm{PRSCh}_1(1^\eta)} : z = 1]|.\end{aligned}$$

Proof: We show by contraposition that $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_0(1^\eta)}$ and $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_1(1^\eta)}$ are indistinguishable. We construct for every attacker $\mathcal{E}$ a reduction M that breaks the DBDH problem with non-negligible probability if $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_0(1^\eta)}$ and $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_1(1^\eta)}$ are distinguishable with non-negligible probability, i.e., if the following difference is non-negligible:

$$\begin{aligned}&|\Pr[z \leftarrow \mathcal{E}(1^\eta)^{\mathrm{PRSCh}_0(1^\eta)} : z = 1] -\\ &\Pr[z \leftarrow \mathcal{E}(1^\eta)^{\mathrm{PRSCh}_1(1^\eta)} : z = 1]|.\end{aligned}$$

Let p be the polynomial of $\mathcal{E}$. The reduction M_b communicates with $\mathcal{E}$ and outputs as a guess z whatever $\mathcal{E}$ outputs as a guess. For a given challenge tuple $(q, g, e, a \cdot g, b \cdot g, s \cdot g, e(g,g)^z)$ and bilinear mapping e, where $z = (abs)$ for $b = 0$ and z is chosen uniformly at random for $b = 1$, M_b is constructed as follows:

```
**Upon initialization**
  uniformly draw a secret key $s'$
  let $par := \langle q, g, e \rangle$
  draw two random numbers $i, j \leftarrow \{1, \dots, p(\eta)\}$
  let $\mathrm{pk}_i := a \cdot g$ and $\mathrm{pk}_j := b \cdot g$
  let $c := 0$ and $cheated := \emptyset$

**Upon a random oracle query for** $h(m)$
  **if** $c \in \{i, j\}$ **then**
    program $h(m) := \mathrm{pk}_c$
    store $cheated := cheated \cup \{m\}$
  **else**
    let $r \leftarrow \mathbb{Z}_q^*$
    store $exponents[m] := r$
    program $h(m) := r \cdot g$
  increment $c := c + 1$
  return $h(m)$

**upon** $(\mathsf{initialize}, \mathrm{ID}_P)$
  **if** $\mathrm{ID}_P \notin cheated$ **then**
    $\mathrm{sk}_P \leftarrow \mathrm{Gen}(par, s', \mathrm{ID}_P)$
  send ready to $\mathcal{E}$

**upon** $(\mathsf{compromise}, \mathrm{ID}_P)$
  **if** $\mathrm{ID}_P \notin challengeSet$ and $\mathrm{ID}_P \notin cheated$ **then**
    $c[\mathrm{ID}_P] := 1$; send $\mathrm{sk}_P$ to $\mathcal{E}$
  **else**
    abort and halt

**upon** nextTimeframe
  $tf := tf + 1$; send $tf$ to $\mathcal{E}$

**upon** $(\mathsf{computeSeed}, \mathrm{ID}_P, \mathrm{ID}_Q)$
  **if** $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf] = \perp$ **then**
    **if** $\mathrm{ID}_P, \mathrm{ID}_Q \in cheated$ **then**
      $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf] := f(e(g,g)^z, tf)$
    **else if** $\mathrm{ID}_P \in cheated$ **then**
      $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf]$
      $:= f(e(s \cdot g, h(\mathrm{ID}_Q))^{exponents(m)}, tf)$
    **else if** $\mathrm{ID}_Q \in cheated$ **then**
      $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf]$
      $:= f(e(h(\mathrm{ID}_P), s \cdot g)^{exponents(m)}, tf)$
    **else**
      $d_{tf} \leftarrow \mathrm{Seed}(par, \mathrm{sk}_P, \mathrm{ID}_Q, tf)$
      $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf] := d_{tf}$
  send $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf]$ to $\mathcal{E}$

**upon** $(\mathsf{challenge}, \mathrm{ID}_P, \mathrm{ID}_Q)$
  set $challengeSet := \{\mathrm{ID}_P, \mathrm{ID}_Q\}$
  **if** $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf] = \perp \wedge$
        $c[\mathrm{ID}_P] = \perp = c[\mathrm{ID}_Q]$
  **then**
    **if** $\mathrm{ID}_P, \mathrm{ID}_Q \in cheated$ **then**
      $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf] := e(g,g)^z$
    **else**
      $d_{tf} \leftarrow \mathrm{Seed}(par, \mathrm{sk}_P, \mathrm{ID}_Q, tf)$
      $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf] := d_{tf}$
  send $seed[\mathrm{ID}_P, \mathrm{ID}_Q, tf]$ to $\mathcal{E}$
```

We have to show that if the attacker distinguishes $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_0(1^\eta)}$ from $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_1(1^\eta)}$ with non-negligible probability then the reduction M also wins in the DBDH game with non-negligible probability.

We prove this statement in three steps. First, we show that $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_b(1^\eta)}$ is indistinguishable from $\mathcal{E}(1^\eta)^{\mathrm{M}_b}$ as long as M_b does not abort and halt. Second, we show that the probability that M_b does not abort and halt is noticeable. Third, we show that the attacker cannot predict when M_b does abort and halt.

As a first step, we show the indistinguishability as long as M_b does not abort and halt. We observe that the output of the random oracle is random and uncorrelated, and that for the parties not in *cheated* will behave exactly like the challenger. Moreover, for every session in which only one of the parties in *cheated* participates, we have

$$\begin{aligned} seed(\mathrm{ID}_P, \mathrm{ID}_Q, tf) &:= f(e(h(\mathrm{ID}_P), s \cdot g)^{exponents(m)}, tf) \\ &= f(e(h(\mathrm{ID}_P), s \cdot g)^r, tf) \\ &= f(e(h(\mathrm{ID}_P), r \cdot g)^s, tf) \\ &= f(e(h(\mathrm{ID}_P), h(\mathrm{ID}_Q))^s, tf) \\ &= f(e(g,g)^{ars}, tf) \text{ where } a \cdot g = h(\mathrm{ID}_P) \end{aligned}$$

Hence, as long as M_b does not abort and halt $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_b(1^\eta)}$ is indistinguishable from $\mathcal{E}(1^\eta)^{\mathrm{M}_b}$.

As a second step, we show that the probability that M_b does not abort and halt is noticeable. This statement directly follows from the fact that M_b randomly picks the parties that are in *cheated* from a set of polynomial size.

As a third step, we show that the attacker cannot predict when M_b aborts and halts. This statement directly follows from the fact that the elements in *cheated* are randomly chosen.

Consequently, if the attacker distinguishes $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_0(1^\eta)}$ from $\mathcal{E}(1^\eta)^{\mathrm{PRSCh}_1(1^\eta)}$ with non-negligible probability then the reduction M also wins in the DBDH game with non-negligible probability. This contradicts the DBDH assumption and the statement follows.

■