Provable Security Analysis of Butterfly Key Mechanism Protocol in IEEE 1609.2.1 Standard

Alexandra Boldyreva
Georgia Institute of Technology
USA
sasha@gatech.edu

Virendra Kumar
Qualcomm
USA
virendra@qti.qualcomm.com

Jiahao Sun
USA

ABSTRACT

The paper provides the first provable security analysis of the Butterfly Key Mechanism (BKM) protocol from IEEE 1609.2.1 standard. The BKM protocol specifies a novel approach for efficiently requesting multiple certificates for use in vehicle-to-everything (V2X) communication. We define the main security goals of BKM, such as vehicle privacy and communication authenticity. We prove that the BKM protocol, with small modifications, meets those security goals. We also propose a way to significantly improve the protocol's efficiency without sacrificing security.

KEYWORDS

V2X communication, butterfly key mechanism, digital certificates, vehicle privacy, unforgeability, provable security

1 INTRODUCTION

1.1 Motivation

V2X. Vehicle-to-vehicle (V2V) communication, where vehicles exchange messages with other vehicles (e.g., vehicle's speed, heading, braking status, etc.), along with other types of vehicle communications, such as vehicle-to-infrastructure (V2I) and vehicle-to-pedestrian (V2P), collectively known as vehicle-to-everything (V2X), have the potential to significantly improve safety and efficiency of our transportation system. The U.S. Department of Transportation (DOT) – National Highway Traffic Safety Administration (NHTSA) estimated that when fully deployed, V2V communications can help prevent up to 592,000 crashes and save up to 1,083 lives per year [22]. In 2017, the DOT proposed a rule [1] to mandate the inclusion of V2V technology in light vehicles in the US. Even though the proposed rule didn't materialize into a mandate, in their recent draft V2X deployment plan [5], the DOT has set short term (2024 – 2026), medium term (2027 – 2029) and long term (2030 – 2034) goals that include a fully deployed national highway system, 6 vehicle manufacturers and 20 vehicle models to be V2X capable.

V2X communication has its own unique challenges and requirements:

- Privacy: Vehicles need protection from being tracked as they are continuously sending sensitive information.
- Trust: Vehicles need assurance that incoming messages are from genuine senders that have been allowed to participate in V2X.
- Resource constraints: Vehicles have limited resources in terms of connectivity, compute, storage, etc.

IEEE Standards. To address the above, the IEEE Std 1609.2 [2] specifies digital certificate formats and the IEEE Std 1609.2.1 [4] specifies certificate management protocols. Both these standards are the de facto specifications for securing V2X communications in the US [3], and form the bases for specifications elsewhere including Europe and China.

Vehicles are issued digital certificates that they use to digitally sign messages so that the receiver of those messages can verify the signatures and be sure that the messages are coming from a genuine sender. Those certificates are designed to be pseudonym, i.e., the certificates do not contain any identifying information, and instead contain permissions to send certain types of messages. Vehicles are also provisioned with not one but several concurrently valid and seemingly unrelated certificates, so that they don't need to use any particular certificate for a prolonged period.

Butterfly Key Mechanism. The IEEE Std 1609.2.1 specifies a novel approach for requesting multiple certificates efficiently through a cryptographic protocol called the *Butterfly Key Mechanism* (BKM).

We briefly and informally describe the protocol, also see Figure 1 for the pictorial description. The protocol involves three parties: an end entity (EE), e.g., a vehicle, a registration authority (RA), and an authorization certificate authority (ACA). The EE has two caterpillar secret-public key pairs and two keys for pseudorandom functions, where one is used for signing and the other for encryption. Using the caterpillar secret keys and the pseudorandom function keys, the EE creates two sets of cocoon secret keys, one set for signing and the other for encryption. The RA, who gets the caterpillar public keys and the pseudorandom function keys from the EE over a secure channel, can create the corresponding sets of cocoon public keys for signing and encryption.

Next, the RA permutes the cocoon public key sets from a large number of EEs, and sends the permuted cocoon public keys to the ACA. The ACA picks a random offset for each caterpillar signing key, creates the corresponding butterfly public key and a digital certificate for it, encrypts all of these under the cocoon public key for encryption, signs the ciphertexts, and sends those back to the RA. The RA "un-permutes" the responses and forwards them to the EE. The EE verifies the signatures, decrypts the ciphertexts using its cocoon secret key for encryption, and obtains the offsets. Using these, the EE computes the butterfly secret and public keys and verifies the certificates for the latter. We provide the protocol details in Section 3.3.

Proceedings on Privacy Enhancing Technologies YYYY(X), 1–18

https://doi.org/XXXXXXX.XXXXXXX

The butterfly key mechanism (cf. clause 9.3 in [4], and [14, 27]) has the following unique privacy and efficiency features:

- The certificate requester needs to make just one request to obtain essentially unlimited number of seemingly unrelated pseudonym certificates.
- The certificate provider that includes two distinct entities, registration authority and certificate authority, can't tell if any two pseudonym certificates belong to the same EE or not, as long as the registration and certificate authorities do not collude with each other or with other entities.

Despite its importance and being on the verge of a massive deployment, the Butterfly Key Mechanism (BKM) protocol, standardized in IEEE Std 1609.2.1 has not been formally analyzed. The works [14, 27] describe the protocol along with the intended security and privacy properties, but do not provide a formal analysis. Simplicio et al. [25] suggests an efficiency improvement so that EEs could avoid sending the encryption keys to the RAs as those keys could be derived by the parties using the same key expansion mechanism. Their security analysis is very informal.

1.2 Our Contributions

We provide the first provable security analysis of the Butterfly Key Mechanism protocol.

Protocol Syntax and Description. We start with defining the Butterfly Key Mechanism (BKM) protocol's functionality (syntax) in Section 3.2. Our syntax follows the flow of the BKM as specified in IEEE Std 1609.2.1, hereafter referred as IEEE BKM, but is general enough to capture the modifications we suggest and some new protocols.

We then describe (in Section 3.3) the IEEE BKM following our syntax.

Security Definitions. Next, we provide the security definitions for BKM protocols. Our security definitions capture two security goals BKM protocols aim to achieve for end entities – privacy (anonymity/unlinkability) and unforgeability (authenticity). For both of these goals we consider different threat models depending on which parties are corrupted. We assume that EEs are honest (though we do discuss what happens if some of their keys get compromised). We consider the cases when either the RA or the ACA are corrupted. We also consider the strongest case when both RA and ACA are corrupted and colluding. In all these cases, we make the following assumptions, which follow the setup in practice. The communication between EEs and the RA is private and authenticated, and the communication between the RA and the ACA is authenticated. EEs have what are called enrollment certificates, which they use to sign the requests to the RA. This helps prevent impersonation attacks on EEs. The caterpillar keys of EEs are always honestly generated and thus have the right distribution and are independent from each other. Each EE is communicating with a single RA, and each RA may talk to several ACAs. For each corruption case, we consider attackers who learn all the public information and everything the corrupted party knows. We assume the attacker is active and can deviate from the protocol.

The goal of the attacker in the privacy experiment is to distinguish whether two butterfly public keys belong to the same EE or two different EEs. The goal of the attacker in the unforgeabilty experiment is to forge a signature under any butterfly public key.

Security Analysis. Our main contribution is the security analysis of IEEE BKM.

Privacy. We start with the goal of privacy. If both ACA and RA are corrupted, then no privacy can be achieved by a BKM protocol[1]. This is because the attacker (by corrupting the ACA) will know the butterfly public keys and the corresponding certificates of the EE, and (by corrupting the RA) to which EE the certificates are returned. In particular, the attacker can link public keys and certificates to EEs (whether they belong to the same EE or not, and moreover, which public key is whose). Hence no EE privacy can be achieved. And of course, the certificates cannot be trusted as they may have been produced by the attacker on behalf of the ACA.

If only the RA is corrupted, then EE's privacy depends on how the corrupted RA deals with ACAs. If there are multiple ACAs and EEs the RA works with, then privacy in the strong sense cannot hold as a malicious RA can do the following attack. Say, the corrupted RA gets requests from EE_1 and EE_2, and each is expanded into several cocoon keys using the expansion function. Then the RA can send ACA_1 all requests from EE_1 and ACA_2 all requests from EE_2. Later, the attacker will not be able to tell the EEs' keys apart, but it will be able to tell their certificates apart, as they will be signed by different ACAs.

If there is a single ACA for all the EEs that the RA services, then privacy can hold. If there are multiple ACAs, and if RA sends fractions of requests from each EE to several ACAs, then privacy can hold within each EE-ACA batch. In practice, however, it is extremely unlikely that an RA will use more than one ACA to generate certificates. The most likely scenario is where a vehicle manufacturer contracts a Security Credential Management System (SCMS) provider, so there will be a one-to-one mapping between the RA and the ACA. Our definition captures both cases.

To prove that privacy holds, we would like to use the fact that the underlying public-key encryption scheme, ECIES is secure. But the standard security (indistinguishability under chosen-plaintext attack or IND-CPA) is not immediately sufficient for us. Since the attacker breaking the protocol will see ciphertexts created under different but related cocoon keys (they are related because they correspond to the same caterpillar key and the attacker knows that key and the cocoon extensions), we need to rely on the security of the base encryption scheme under a weak version of the notion of related-key attack (RKA) [11, 12], where the related keys are created by adding random offsets to a single key.

It turns out that proving security requires an additional non-standard notion of security for the base encryption scheme, the property that captures inability to create ciphertexts which are valid wrt different keys. Such a property is called *robustness*. Robustness of asymmetric encryption was studied by Abdalla et al. and Farshim et al. [6, 17]. However, as we discuss in Section 5.2, their security definitions are not immediately suitable for us. We provide the definition of robustness capturing our setting, that we call *target robustness*.

[1] It may be possible to construct a protocol with a slightly different functionality but serving the same general security goals using group signatures [16] or fair blind signatures [26], but efficiency is likely going to be an issue.

Theorem 5.1 provides the formal bound stating that the generic IEEE BKM (based on arbitrary underlying schemes) achieves privacy against the corrupted RA in the ideal cipher and random oracle model, assuming that the base encryption scheme is IND-CPA secure against additive RKA and is also target-robust, and the signature scheme used by the ACA is UF-CMA secure.

Unfortunately, ECIES, the base encryption scheme used by IEEE BKM is not known to be target-robust or secure in this sense of additive RKA. The good news is that we can prove ECIES is secure in these senses. In Theorem 6.1 we show that ECIES is IND-CPA under additive RKA assuming hardness of the Hash Diffie-Hellman problem and the IND-CPA security of the underlying symmetric encryption scheme. We also prove that ECIES, with small modifications, satisfies target-robustness in the random oracle model (cf. Theorem 6.2).

In case of corrupted ACAs, EEs' privacy does hold assuming that the honest RA sends each ACA the permuted requests containing equal portion of each EE's cocoon keys or there is only one ACA for each EE. Theorem 5.2 states the result.

Unforgeability. We now turn to the goal of unforgeability/authenticity. Consider the strongest model when both the RA and ACA are corrupt. The adversary cannot compute the butterfly secret keys as it does not know the caterpillar secret keys. But to formally prove unforgeability we face the same issue we faced to prove privacy, in that the standard (unforgeability under chosen-message attack or UF-CMA) security of the base signature is not enough for the proof to go through. What one would need is stronger security for the base signature scheme such as unforgeability under related-key attack (UF-RKA [11]). Such a notion requires unforgeability to hold even if the attacker can observe signatures under related keys. The relation we are concerned with is a specific one, where random offsets are added to the secret key.

In order to obtain BKM unforgeability under such additive RKA attacks we need a very simple modification to the protocol. The change is as follows. In the current design, the ACA picks the offset r, and sends it (encrypted) to EE via RA. We propose to use $H(r, cpk)$ instead of r in butterfly key generation, where H is a hash function and cpk is the cocoon public key of EE. The intuition for this modification is to prevent a corrupted ACA to gain any advantage by picking r maliciously. Applying the hash makes the result look random despite the choice of r, as long as hash inputs do not repeat. The use of the cocoon public key is to prevent repeated inputs. Even if the malicious ACA chooses the same r, the cocoon keys will be distinct as the outputs of expansion functions will be distinct (with overwhelming probability). Theorem 5.3 states that the unforgeability of the generic (based on an arbitrary signature scheme) IEEE BKM, with slight modification, reduces to the additive RKA security of the underlying signature scheme, in the ideal cipher and random oracle models.

But is ECDSA signature scheme the protocol uses secure under (additive) RKA? ECDSA has not been proven to be (additive) RKA secure until the very recent work by Groth and Shoup [21], and their results fortunately can be used to complete arguing security of IEEE BKM.

Since we could prove unforgeability in the strongest corruption model, we do not focus on the weaker models. We observe, however, that in the case of honest ACA (and corrupted RA), unforgeability holds without the protocol's modification.

We note that the modifications we suggest are only needed for the proofs to go through, and we do not know of attacks on the protocol without the modifications.

Efficiency Improvement. We propose a simple change to the protocol that yields a significant efficiency improvement. More specifically, we propose that the ACA can re-use randomness when encrypting certificate responses using ECIES under different cocoon encryption keys. Observe that the first part of an ECIES ciphertext is vG, where $v \in \mathbb{Z}_q$ has to be picked at random for each encryption. We show that the ACA can re-use the same v across all ciphertexts. This will result in reducing the computation in half (as the ACA will have to perform only $N + 1$ scalar elliptic curve multiplications as opposed to $2N$, where N is the number of encryptions the ACA performs) and significantly reducing the communication (since vG can be sent only once to each EE).

Kurosawa, Bellare et al. and Barbosa and Farshim [8, 10, 24] studied the problem of secure randomness re-use. Bellare et al. [10] defined the security notion for asymmetric *multi-recipient* encryption schemes (MRES) and proved that randomness can be safely re-used across multiple DHIES encryptions under different public keys. Their result applies to ECIES as well, however, we cannot use their result as is. The reason is the public keys in our application are related, and the results of [8, 10] do not cover this case. We show that the MRES with ECIES and randomness re-use is secure in the setting with the related keys, assuming hardness of the Oracle Diffie-Hellman problem and IND-CPA security of the underlying symmetric encryption scheme (cf. Theorem 7.2). Finally, we show that this is what we need for unlinkability of the modified IEEE BKM.

2 NOTATION AND PRELIMINARIES

2.1 Notation

For $l \in \mathbb{N}$ we denote by 1^l the string of l "1" bits. $a_1 \| ... \| a_n$ denotes the string encoding of $a_1, ..., a_n$ from which $a_1, ..., a_n$ are uniquely recoverable, e.g., concatenation. We use the bold font $\mathbf{x}$ to denote the list $(x_1, \ldots, x_n)$ for any x. We assume that the number of elements in the list is clear from the context. And then $x[i] = x_i$ is the ith element in $\mathbf{x}$. If S is a set then $x \xleftarrow{\$} S$ denotes that x is selected uniformly at random from S. If $\mathcal{A}$ is a randomized algorithm then $y \xleftarrow{\$} \mathcal{A}(x_1, x_2, ...)$ denote the operation of running $\mathcal{A}$ on inputs $x_1, x_2, ...$ and assigning output to y. For the syntax of any interactive protocol (algorithm) $\mathcal{I}$ executed between party A and party B, we use the convention: $(\mathsf{output}_A, \mathsf{output}_B) \leftarrow [\mathcal{I}_A(\mathsf{input}_A), \mathcal{I}_B(\mathsf{input}_B)]$. If $\mathcal{A}$ is an algorithm, then $\mathcal{A} \Rightarrow x$ means that $\mathcal{A}$ outputs string x at the end of its execution. By efficient we mean algorithms that run in (expected) polynomial-time in the length of their inputs, and make polynomial number of queries of polynomial length.

2.2 Preliminaries

We recall the cryptographic primitives the BKM protocol uses and their security definitions.

Digital Signatures. A digital signature scheme $\mathcal{DS}$, associated with the message space MsgSp, consists of four algorithms $(\mathcal{G}, \mathcal{K}, \mathcal{S}, \mathcal{V})$. The global info generation algorithm $\mathcal{G}$ takes as input the security parameter 1^λ and outputs the global info I. (For Diffie-Hellman-based schemes the global info may include the group description and the generator of the group.) The key generation algorithm $\mathcal{K}$ takes as input the global information I that contains the security parameter and possibly some other information and returns a pair of public-secret key (pk, sk). The signing algorithm takes a secret key sk and message m then returns a signature σ. The verification algorithm takes a public key pk, a message m, and a signature σ, then returns a bit b indicating whether the signature is valid. Correctness of the scheme requires that for any I output by $\mathcal{G}(1^\lambda)$, $(pk, sk) \xleftarrow{\$} \mathcal{K}(I)$, and any $m \in \mathsf{MsgSp}$, $\mathcal{V}(pk, m, \mathcal{S}(sk, m)) = 1$.

For security, recall the following security experiment $\mathbf{Exp}^{\mathsf{uf\text{-}cma}}_{\mathcal{DS}}(\mathcal{A})$ associated with $\mathcal{DS}$ and adversary $\mathcal{A}$. First, keys are generated: $I \xleftarrow{\$} \mathcal{G}(1^\lambda)$, $(pk, sk) \xleftarrow{\$} \mathcal{K}(I)$. Then $\mathcal{A}$ is given pk and access to the oracle $\mathcal{S}_{sk}(\cdot) = \mathcal{S}(sk, \cdot)$. In the end, $\mathcal{A}$ outputs a message-signature pair (m, σ). $\mathbf{Exp}^{\mathsf{uf\text{-}cma}}_{\mathcal{DS}}(\mathcal{A})$ returns 1 iff $\mathcal{V}(pk, m, \sigma) = 1$ and m is in MsgSp and was not queried to the $\mathcal{S}_{sk}(\cdot)$ oracle. The advantage $\mathbf{Adv}^{\mathsf{uf\text{-}cma}}_{\mathcal{DS}}(\mathcal{A})$ is defined as $\Pr\left[\mathbf{Exp}^{\mathsf{uf\text{-}cma}}_{\mathcal{DS}}(\mathcal{A}) \Rightarrow 1\right]$.

Asymmetric Encryption. An asymmetric encryption scheme $\mathcal{AE}$, associated with the message space MsgSp, is defined by four algorithms $(\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$. The global info generation algorithm $\mathcal{G}$ takes as input the security parameter 1^λ and outputs the global info I. The key generation algorithm $\mathcal{K}$ takes as input the global information I and returns a pair of public-secret key (pk, sk). The encryption algorithm $\mathcal{E}$ takes a public key pk and message m to return a ciphertext c. The decryption algorithm takes a secret key sk and a ciphertext c to return a plaintext m. Correctness of the scheme requires that for any any I output by $\mathcal{G}(1^\lambda)$, $(pk, sk) \xleftarrow{\$} \mathcal{K}(I)$, and any $m \in \mathsf{MsgSp}$, $\mathcal{D}(sk, \mathcal{E}(pk, m)) = m$. We will define the security notions we need later in the paper.

Symmetric Encryption. A symmetric encryption scheme $\mathcal{SE}$, associated with the message space MsgSp, consists of three algorithms $(\mathcal{K}, \mathcal{E}, \mathcal{D})$. The key generation algorithm $\mathcal{K}$ returns a secret key k. The encryption algorithm $\mathcal{E}$ takes key k and message m to return a ciphertext c. The decryption algorithm takes key k and a ciphertext c to return a plaintext m. Correctness of the scheme requires that for any $k \xleftarrow{\$} \mathcal{K}$ and any $m \in \mathsf{MsgSp}$, $\mathcal{D}(k, \mathcal{E}(k, m)) = m$.

For security, we recall the ind-cpa definition. Let $\mathsf{LR}(\cdot, \cdot, b)$ denote the function that on inputs m_0, m_1 returns m_b. For an adversary $\mathcal{A}$, consider the experiments $\mathbf{Exp}^{\mathsf{ind\text{-}cpa}\text{-}b}_{\mathcal{SE}}(\mathcal{A})$. First, the key is generated as $k \xleftarrow{\$} \mathcal{K}$. Then $\mathcal{A}$ is given access to the oracle $\mathcal{E}(k, \mathsf{LR}(\cdot, \cdot, b))$. We require that each query (m_0, m_1) that $\mathcal{A}$ makes to its oracle satisfies $|m_0| = |m_1|$. Finally, $\mathcal{A}$ outputs a bit d, and the experiment returns 1 iff $b = d$. The ind-cpa advantage $\mathbf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{SE}}(\mathcal{A})$ is defined as $\Pr\left[\mathbf{Exp}^{\mathsf{ind\text{-}cpa\text{-}0}}_{\mathcal{SE}}(\mathcal{A}) \Rightarrow 0\right] - \Pr\left[\mathbf{Exp}^{\mathsf{ind\text{-}cpa\text{-}1}}_{\mathcal{SE}}(\mathcal{A}) \Rightarrow 1\right]$.

3 THE BUTTERFLY KEY MECHANISM PROTOCOL

In this section we first informally describe the IEEE BKM protocol. Next, we formally define the syntax (functionality) of a BKM protocol. This is necessary for the formal security analysis. Next, we formally specify the cryptographic core of the IEEE BKM, following the syntax.

3.1 Overview of IEEE BKM

The IEEE BKM protocol involves three parties: an end entity (EE), a registration authority (RA), and an authorization certificate authority (ACA). Figure 1 informally presents the main steps of the protocol.

3.2 Protocol Syntax

The Butterfly Key Mechanism protocol ($\mathcal{BKM}$) is an interactive protocol involving three parties: an end entity (EE), a registration authority (RA), and an authorization certificate authority (ACA). It is associated with two digital signature schemes $\mathcal{DS}_1 = (\mathcal{G}_1, \mathcal{K}_1, \mathcal{S}_1, \mathcal{V}_1)$, $\mathcal{DS}_2 = (\mathcal{G}_2, \mathcal{K}_2, \mathcal{S}_2, \mathcal{V}_2)$ with message spaces $\mathsf{MsgSp}_1, \mathsf{MsgSp}_2$. ($\mathcal{DS}_1$ and $\mathcal{DS}_2$ can be the same schemes.) The protocol consists of the following algorithms and interactive subprotocols:

Caterpillar key generation (CKG). The algorithm is executed by the EE. It takes the security parameter and outputs a caterpillar key pair and an expansion key : $(pk_{\mathsf{cp}}, sk_{\mathsf{cp}}, k_{\mathcal{EXP}}) \xleftarrow{\$} \mathcal{CKG}(1^\lambda)$.

ACA signing key generation (ACAKG). The algorithm is run by the ACA. It takes the security parameter and outputs a pair of signing keys : $(pk_{\mathsf{aca}}, sk_{\mathsf{aca}}) \xleftarrow{\$} \mathcal{ACAKG}(1^\lambda)$.

Cocoon key expansion (CKE). This is an interactive subprotocol between the EE and the RA. The EE takes as inputs the expansion key and the caterpillar secret key, and at the end of the interaction outputs a list of cocoon secret keys. The RA inputs the expansion key and the caterpillar public key, and outputs a list of cocoon public keys: $[(\boldsymbol{pk_{cc}}, \boldsymbol{sk_{cc}}), \boldsymbol{pk_{cc}}] \xleftarrow{\$} [\mathcal{CKE}_{\mathsf{EE}}(sk_{\mathsf{cp}}, k_{\mathcal{EXP}}), \mathcal{CKE}_{\mathsf{RA}}(pk_{\mathsf{cp}}, k_{\mathcal{EXP}})]$. In practice the expansion key is sent by the EE via a secure channel.

Butterfly key generation (BKG). This is an interactive subprotocol between the RA and the ACA. The RA takes input the ACA's public signing verification key and the cocoon public keys and outputs the certificate response. The ACA takes its signing key and has no output: $[\boldsymbol{rsp}, \perp] \xleftarrow{\$} [\mathcal{BKG}_{\mathsf{RA}}(pk_{\mathsf{aca}}, \boldsymbol{pk_{cc}}), \mathcal{BKG}_{\mathsf{ACA}}(sk_{\mathsf{aca}})]$

Butterfly key reconstruction (BKR). This is an interactive subprotocol between the EE and the RA. The EE takes inputs the cocoon secret keys and the ACA's signing verification key and outputs lists of butterfly public and secret keys, as well as a list of certificates. The RA takes input the response from ACA in $\mathcal{BKG}$ and outputs nothing: $[(\boldsymbol{pk_{bf}}, \boldsymbol{sk_{bf}}, \boldsymbol{cert}), \perp] \xleftarrow{\$} [\mathcal{BKR}_{\mathsf{EE}}(\boldsymbol{sk_{cc}}, pk_{\mathsf{aca}}), \mathcal{BKR}_{\mathsf{RA}}(\boldsymbol{rsp})]$.

Correctness. Informally, correctness requires that at the end of the protocol the EE obtains valid key pairs for the signature scheme and valid certificates for the public keys. More precisely, we require that for any λ, $(pk_{\mathsf{cp}}, sk_{\mathsf{cp}}, k_{\mathcal{EXP}})$ output by $\mathcal{CKG}(1^\lambda)$,

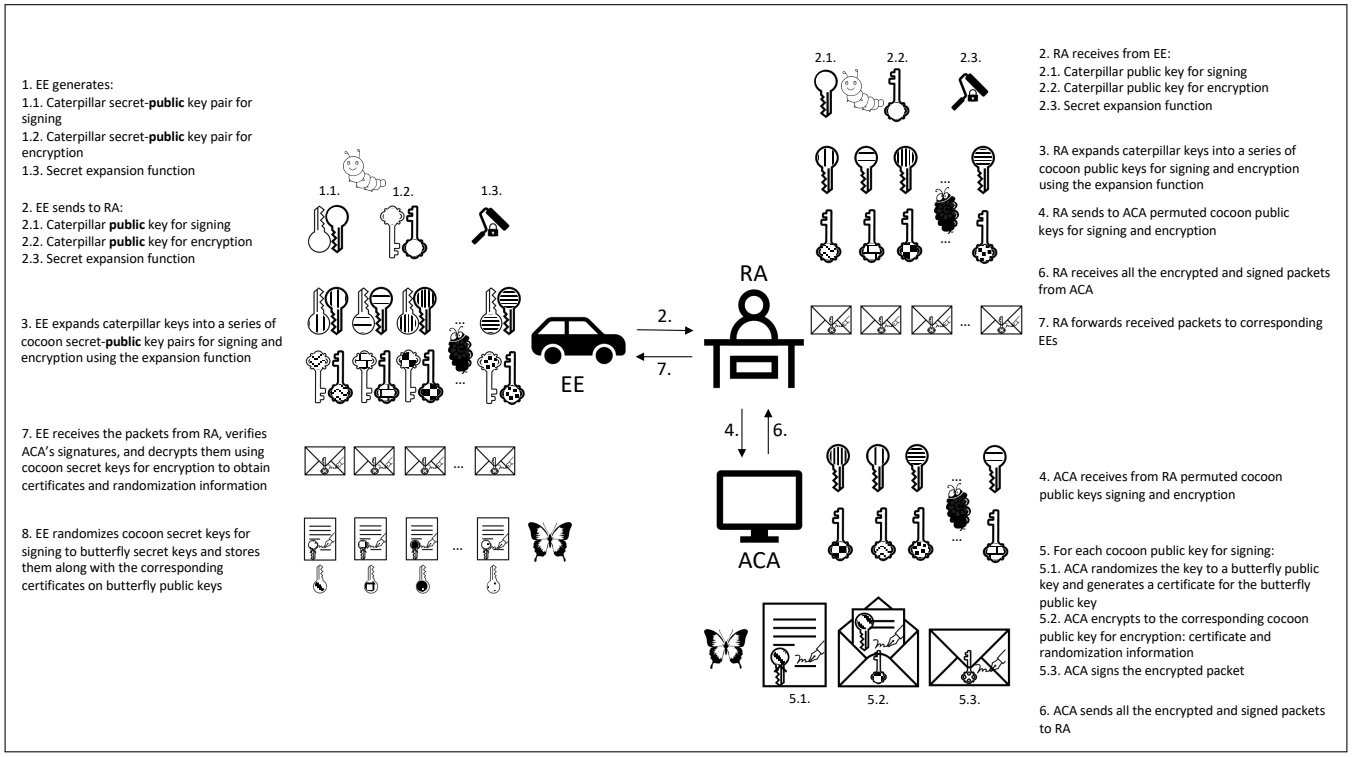

Figure 1: Eight main steps of IEEE BKM protocol, ran between EE, RA and ACA.

$(pk_{\text{aca}}, sk_{\text{aca}})$ output by $\mathcal{ACAKG}(1^\lambda)$, and for the sequential executions of interactive protocols $\mathcal{CKE}, \mathcal{BKG}$ and $\mathcal{BKR}$, where $(\boldsymbol{pk_{bf}}, \boldsymbol{sk_{bf}})$ denote the butterfly keys produced by the EE at the end of $\mathcal{BKR}$, we have that for any message $m \in \text{MsgSp}_1$, $\mathcal{DS}_1.\mathcal{V}_1(\boldsymbol{pk_{bf}}[i], m, \mathcal{DS}_1.\mathcal{S}_1(\boldsymbol{sk_{bf}}[i], m)) = 1$. Also, we require that $\boldsymbol{pk_{bf}}[i] \in \text{MsgSp}_2$ and $\mathcal{DS}_2.\mathcal{V}_2(pk_{\text{aca}}, \boldsymbol{pk_{bf}}[i], \boldsymbol{cert}[i]) = 1$ for any $1 \le i \le n$. Here for simplicity we ignore the auxiliary information that the certificates usually contain, and focus only on the public butterfly keys certification.

3.3 IEEE-BKM Description

General Description. We now specify the IEEE Butterfly Key Certificate protocol using our syntax. We first present the protocol for general (but elliptic-curve Diffie-Hellman based) schemes, and later discuss the particular instantiations the protocol uses.

Let $\mathcal{DS}_1 = (\mathcal{G}_1, \mathcal{K}_1, \mathcal{S}_1, \mathcal{V}_1)$, $\mathcal{DS}_2 = (\mathcal{G}_2, \mathcal{K}_2, \mathcal{S}_2, \mathcal{V}_2)$ be signature schemes and let $\mathcal{AE} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be an asymmetric encryption scheme. $\mathcal{DS}_1$ and $\mathcal{AE}$ are elliptic-curve Diffie-Hellman based schemes, meaning that their key generation algorithms are as follows. The global info generation algorithm outputs the global information $I = (\mathbb{G}, G, q)$, where $\mathbb{G}$ is the group of points on the elliptic curve of prime order q, generated by $G \in \mathbb{G}$. The key generation algorithm outputs a pair of keys (pk, sk), where sk is a random element of $\mathbb{Z}_q$ and $pk = x \cdot G$. Let $\mathcal{EXP}\colon \{0,1\}^k \times \{0,1\}^l \to \mathbb{Z}_q$ be a function family that we refer to as the *expansion function*. Let $\mathcal{H}\colon \{0,1\}^* \to \mathbb{Z}_q$ be a hash function. For simplicity of notation we assume that public keys contain the global information and that the parties are stateful.

∘ $\underline{\mathcal{CKG}}$:
$I_s \xleftarrow{\$} \mathcal{DS}_1.\mathcal{G}_1(1^\lambda)$; $I_e \xleftarrow{\$} \mathcal{AE}.\mathcal{G}(1^\lambda)$;
$(pk_{\text{cp}}, sk_{\text{cp}}) \xleftarrow{\$} \mathcal{DS}_1.\text{Kg}_1(I_s)$; $k_{\mathcal{EXP}} \xleftarrow{\$} \{0,1\}^k$
$(epk_{\text{cp}}, esk_{\text{cp}}) \xleftarrow{\$} \mathcal{AE}.\text{Kg}(I_e)$; $ek_{\mathcal{EXP}} \xleftarrow{\$} \{0,1\}^k$
Return $(pk_{\text{cp}} \| epk_{\text{cp}},\ sk_{\text{cp}} \| esk_{\text{cp}},\ k_{\mathcal{EXP}} \| ek_{\mathcal{EXP}})$

The $\mathcal{CKG}$ (caterpillar key generation) algorithm specifies how the (caterpillar and expansion) keys initially possessed by an EE are generated.

∘ $\underline{\mathcal{ACAKG}}$:
$I \xleftarrow{\$} \mathcal{DS}_2.\mathcal{G}_2(1^\lambda)$; $(pk_{\text{aca}}, sk_{\text{aca}}) \xleftarrow{\$} \mathcal{DS}_2.\text{Kg}_2(I)$
Return $(pk_{\text{aca}}, sk_{\text{aca}})$

The $\mathcal{ACAKG}$ (ACA key generation) algorithm specifies how the keys initially possessed by the ACA are generated. The ACA's public key is assumed to be publicly known.

∘ $\underline{\mathcal{CKE}}$: The protocol is presented in Figure 2.

The $\mathcal{CKE}$ (cocoon key expansion) protocol is executed between the EE and RA. The EE sends the public caterpillar and expansion keys to the RA via a secure channel. Both parties use the expansion keys to expand the public caterpillar keys into multiple cocoon public keys. The EE can also expand the secret caterpillar keys into

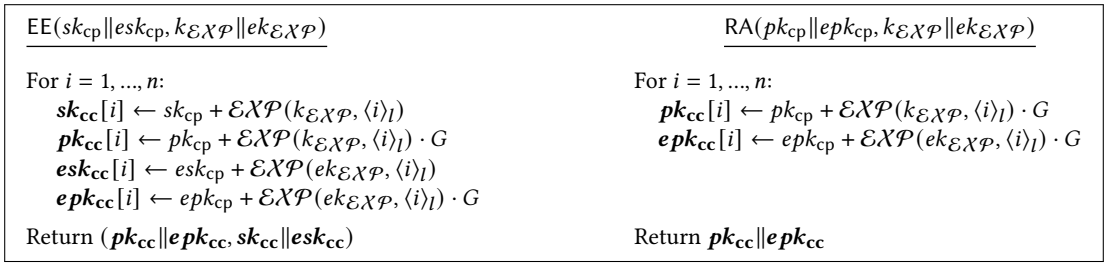

Figure 2: $\mathcal{CKE}$ algorithm. Here $\langle i \rangle_i$ means number i represented as l bits.

multiple cocoon secret keys. We assume that $n \leq 2^l$.

∘ $\underline{\mathcal{BKG}}$: The protocol is presented in Figure 3. There P_n is a set of all permutations on n elements.

The $\mathcal{BKG}$ (butterfly key generation) protocol is executed between the RA and ACA. The RA randomly permutes the EE's cocoon public keys for signing and encryption and sends them to the ACA. The ACA expands the signing cocoon keys with random offsets into butterfly keys and certifies them. It then encrypts each butterfly key, the certificate and the random offset under the EE's public cocoon encryption key. The ACA also signs each ciphertext, and sends them all to the RA. The RA "un-permutes" the ciphertexts.

∘ $\underline{\mathcal{BKR}}$: The protocol is presented in Figure 4.

The $\mathcal{BKR}$ (butterfly key reconstruction) protocol is between the EE and the RA. The RA sends the EE the ciphertexts. The EE verifies the ACA's signatures and decrypts the ciphertexts using the secret cocoon keys for decryption. It then uses the offsets to expand the secret signing cocoon keys into the butterfly secret keys. It also verifies the validity of the certificates and that the secret keys match the public keys. The correctness follows from correctness of the base schemes. Even though the butterfly and cocoon keys are not computed using the respected key generation algorithms, they still could be output by those algorithms, and hence the correctness follows.

IEEE-BKM Instantiations. IEEE BKM uses AES as $\mathcal{EXP}$[2], ECDSA as $\mathcal{DS}_1$, $\mathcal{DS}_2$, and ECIES as $\mathcal{AE}$ with the CCM mode as the underlying symmetric encryption. The IEEE 1609.2.1 standard specifies several mechanisms for secure channels including TLS1.2, TLS1.3 and ISO/TS21177.

4 BKM SECURITY DEFINITIONS

In this Section we formally define the security notions for the two main security goals of the BKM protocol: privacy (anonymity/unlinkability) and authenticity (unforgeability). We treat each goal separately. For each goal, we consider different scenarios of corrupted parties. The EE is always honest[3]. The strongest model will assume that both RA and ACA are corrupted. The weaker models treat the cases when either RA or ACA are corrupted.

[2]The standard also makes use of some XOR operations, but those are not relevant to our analyses.

[3]In Section 7 we consider a modification to the protocol where EEs share some information, and hence it makes sense to consider the case of compromised EEs there.

Let $\mathcal{BKM} = (\mathcal{CKG}, \mathcal{ACAKG}, \mathcal{CKE}, \mathcal{BKG}, \mathcal{BKR})$ be a BKM protocol associated with two digital signature schemes $\mathcal{DS}_1 = (\mathcal{G}_1, \mathcal{K}_1, \mathcal{S}_1, \mathcal{V}_1)$, $\mathcal{DS}_2 = (\mathcal{G}_2, \mathcal{K}_2, \mathcal{S}_2, \mathcal{V}_2)$ with message spaces MsgSp_1, MsgSp_2.

In the security following security definitions we assume that the adversary is stateful, i.e., it can preserve state between invocations. We do not specify states explicitly. If the attacker was previously given some inputs, it can use those inputs in further stages.

4.1 Privacy

We formalize security in terms of end entity (EE) privacy. In other words, we define anonymity (unlinkability) of EEs. The adversary should not be able to tell to which EE a (butterfly) public key belongs to. Note that this goal is not applicable in the standard PKI setting, where digital certificates bind public keys and *public* identities together. In our setting, the certificates intentionally do not contain information about the key owner.

I. Honest EE, corrupted RA and ACA. In this strongest model, no EE privacy is possible. Since the attacker communicates to EEs on behalf of corrupted RA and ACA, it can later link public keys and certificates to EEs (whether they belong to the same EE or not, and moreover, whose public key is whose). More precisely, the attacker will know the butterfly public keys of EE and the certificates, and to which EE the certificates are returned. Hence no EE privacy can be achieved.

II. Honest EE and ACA, corrupted RA. If the RA is corrupted, EE's privacy depends on how the corrupted RA deals with the ACAs. If there are more than one ACA and EE the RAs works with, then no privacy holds in the strong sense as a malicious RA can do the following attack. Say, the corrupted RA gets requests from EE_1 and EE_2, and each is expanded into several cocoon keys using the expansion function. Then the CKE can send ACA_1 all requests from EE_1 and ACA_2 all requests from EE_2. Later, the attacker will not be able to tell the EEs' keys apart, but it will be able to tell their certificates apart, as they will be signed by different ACAs with different public keys.

If all requests from each EE are sent to a single ACA, then privacy can hold. If an RA sends fractions of requests from each EE to several ACAs, then privacy can hold within each batch. In practice, it is extremely unlikely that an RA will use more than one ACA to generate a single batch of certificates. The most likely scenario is where a vehicle manufacturer contracts a Security Credential Management System (SCMS) provider, so there will be a one-to-one

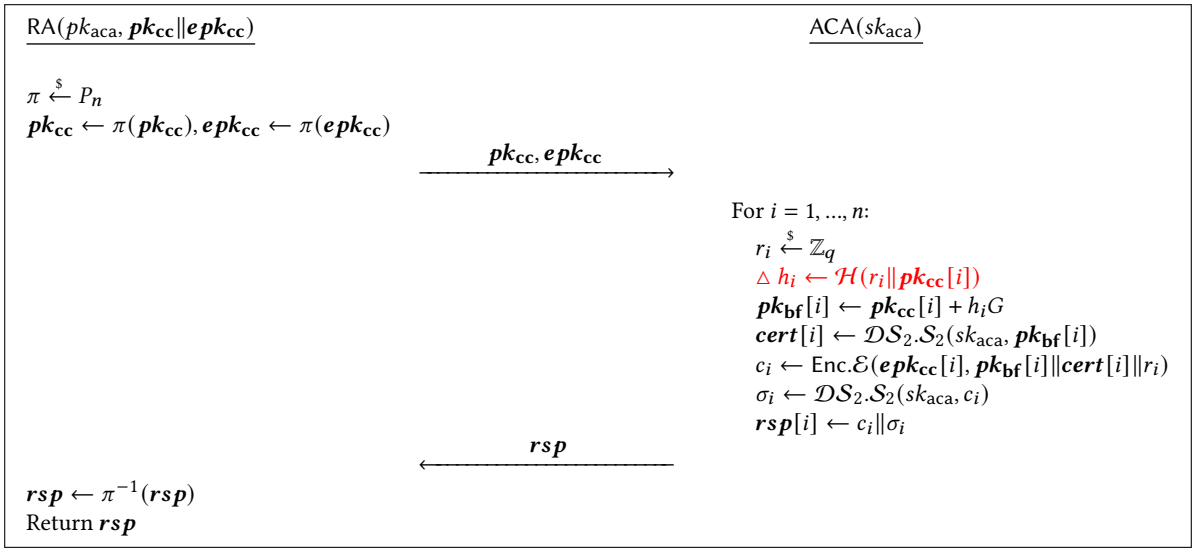

Figure 3: $\mathcal{BKG}$ algorithm. The red font (also marked with △ symbol) indicates the change we suggest. In the current protocol, $h_i = r_i$.

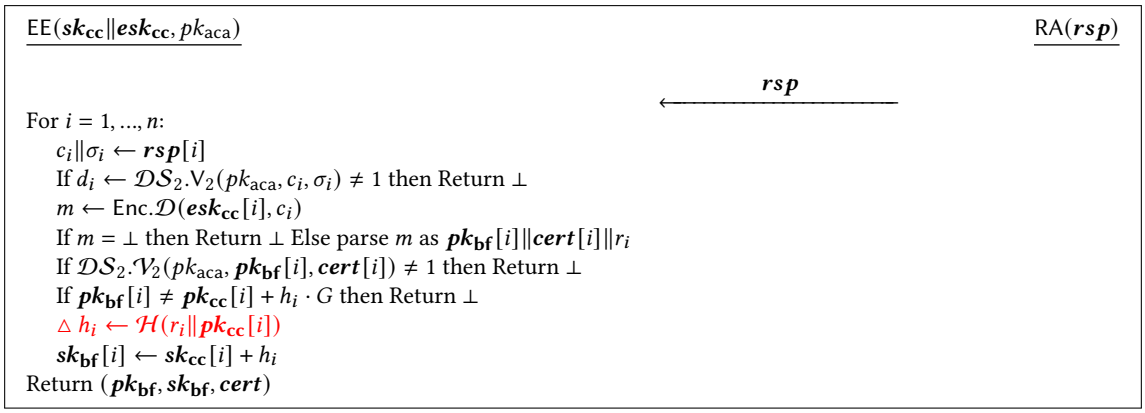

Figure 4: $\mathcal{BKR}$ algorithm. The red font (also marked with △ symbol) indicates the change we suggest. In the current protocol, $h_i = r_i$.

mapping between the RA and the ACA. Our definition (implicitly) covers both cases.

The experiment $\mathbf{Exp}^{\text{priv-cra}}_{\mathcal{BKM}}$ is defined in Figure 7. The definition models a malicious RA, whose goal is to link the butterfly keys of the same EE together. The adversary interacts with two honest EEs on behalf of the RA and learns everything that the RA knows. It also interacts with the honest ACA. We also let the adversary know both caterpillar public keys. We then give the attacker a pair of public butterfly keys, and the certificates, which either belong to the same EE or two different EEs. The goal of the adversary is figure out which case it is.

The privacy advantage is defined as

$$\mathbf{Adv}^{\text{priv-cra}}_{\mathcal{BKM}}(\mathcal{A}) = 2 \cdot \Pr\left[\mathbf{Exp}^{\text{priv-cra}}_{\mathcal{BKM}}(\mathcal{A}) \Rightarrow 1\right] - 1 \,.$$

Note that the definition does not consider multiple ACAs for simplicity. (One could do so, but that would not make the definition stronger). If multiple ACAs are used in practice, then the definition ensures that the EEs' keys certified *by each* ACA are unlinkable.

III. Honest EE and RA, corrupted ACA. The definition models a malicious ACA whose goal is to link two butterfly keys of the same EE together. The experiment $\mathbf{Exp}^{\text{priv-caca}}_{\mathcal{BKM}}$ associated with attacker $\mathcal{A}$ is defined in Figure 5. There are two EEs that interact with the honest RA. The adversary learns everything that the ACA knows, namely, the permuted cocoon public keys, and the ACA's secret signing key. We also let the adversary know both caterpillar public keys. Note that the adversary does not get to intervene into the EE-RA communication because they talk over a secure channel. We then give the attacker a pair of public cocoon keys, which either belong to the same EE or two different EEs. The goal of the adversary is figure out which case it is. Note that the ACA can always link a butterfly public signing key to the corresponding

cocoon key, so it is enough for us only focus on the cocoon key unlinkability.

The privacy advantage is defined as

$$\mathbf{Adv}^{\text{priv-caca}}_{\mathcal{BKM}}(\mathcal{A}) = 2 \cdot \Pr\left[\mathbf{Exp}^{\text{priv-caca}}_{\mathcal{BKM}}(\mathcal{A}) \Rightarrow 1\right] - 1 \ .$$

Recall that P_{2n} denotes the set of all permutations on $2n$ elements.

We note that in this definition the adversary does not have to participate in any interactive protocol or have any oracle access, as it has already got all the information it can get.

$\underline{\mathbf{Exp}^{\text{priv-caca}}_{\mathcal{BKM}}(\mathcal{A})}$

$\pi \xleftarrow{\$} P_{2n}$; $b \xleftarrow{\$} \{0,1\}$

$(pk^A_{\text{cp}}, sk^A_{\text{cp}}, k^A_{\mathcal{EXP}}) \xleftarrow{\$} \mathcal{CKG}(1^\lambda)$

$(pk^B_{\text{cp}}, sk^B_{\text{cp}}, k^B_{\mathcal{EXP}}) \xleftarrow{\$} \mathcal{CKG}(1^\lambda)$

$(pk_{\text{aca}}, sk_{\text{aca}}) \xleftarrow{\$} \mathcal{ACAKG}(1^\lambda)$

$[(\boldsymbol{sk_{cc}}^A, \boldsymbol{pk_{cc}}^A), \boldsymbol{pk_{cc}}^A] \xleftarrow{\$} [\mathcal{CKE}_{\text{EE}}(sk^A_{\text{cp}}, k^A_{\mathcal{EXP}}), \mathcal{CKE}_{\text{RA}}(pk^A_{\text{cp}}, k^A_{\mathcal{EXP}})]$

$[(\boldsymbol{sk_{cc}}^B, \boldsymbol{pk_{cc}}^B), \boldsymbol{pk_{cc}}^B] \xleftarrow{\$} [\mathcal{CKE}_{\text{EE}}(sk^B_{\text{cp}}, k^B_{\mathcal{EXP}}), \mathcal{CKE}_{\text{RA}}(pk^B_{\text{cp}}, k^B_{\mathcal{EXP}})]$

$\boldsymbol{pk_{cc}} \leftarrow \pi(\boldsymbol{pk_{cc}}^A, \boldsymbol{pk_{cc}}^B)$

$pk_1 \xleftarrow{\$} \{pk^A_{\text{cc}1}, ..., pk^A_{\text{cc}n}\}$

If $b = 0$:

  $pk_2 \xleftarrow{\$} \{pk^A_{\text{cc}1}, ..., pk^A_{\text{cc}n}\} \setminus \{pk_1\}$

Else:

  $pk_2 \xleftarrow{\$} \{pk^B_{\text{cc}1}, ..., pk^B_{\text{cc}n}\}$

$d \xleftarrow{\$} \mathcal{A}(pk^A_{\text{cp}}, pk^B_{\text{cp}}, sk_{\text{aca}}, \boldsymbol{pk_{cc}}, pk_1, pk_2)$

Return 1 iff $(b = d)$

Figure 5: The experiment $\mathbf{Exp}^{\text{priv-caca}}_{\mathcal{BKM}}(\mathcal{A})$ for defining privacy with corrupted ACA.

4.2 Unforgeability

A BKM protocol lets EEs obtain public-secret key pairs to be used for signing. Accordingly, the protocol must guarantee that attackers cannot forge $\mathcal{DS}_1$ signatures on behalf of EEs. In addition, the public key certificates issued by the ACA must also be unforgeable. But the latter goal is straightforward and follows from the standard security of $\mathcal{DS}_2$ signature scheme. Hence, we focus on studying the former goal.

We first consider the strongest definition where only the EE is honest.

I. Honest EE, corrupted RA and ACA. The adversary learns all information known to the RA and ACA and gets to interact with the honest EE on their behalf. The adversary wins if it produces a new valid forgery for one of the EE's butterfly signing keys. Figure 6 defines the security experiment $\mathbf{Exp}^{\text{uf-cma}}_{\mathcal{BKM}}(\mathcal{A})$ in detail, and the unforgeability advantage of the attacker $\mathcal{A}$ is defined as

$$\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{BKM}}(\mathcal{A}) = \Pr\left[\mathbf{Exp}^{\text{uf-cma}}_{\mathcal{BKM}}(\mathcal{A}) \Rightarrow 1\right] .$$

Remark. We note that the corrupted ACA in possession of the ACA's secret key can issue the certificates for any public key. We still consider the corrupted ACA and study its inability to forge signatures on behalf of honest EEs. This is similar to the situation with the standard PKI: a corrupted CA can create certificates on any key and frame any user, but it still should not be able to forge signatures under the legitimate keys users certified.

II. Honest EE and ACA, corrupted RA. The security definition in this case is strictly weaker than the above for the case of corrupted ACA and RA. Since the BKM protocol can achieve the stronger latter definition, we do not consider the former in detail. But the security guarantees are much stronger in this setting because the ACA is honest and all certificates are hence trusted.

III. Honest EE and RA, corrupted ACA. It is important to remember that the corrupted ACA can always certify keys of its choice. Yet the definition can still ensure unforgeability under the butterfly keys accepted by EEs. Again, the security definition is strictly weaker than that in the case of corrupted ACA and RA. Since the BKM protocol can achieve the stronger one, we do not treat the weaker definition separately.

5 SECURITY ANALYSIS OF THE GENERIC IEEE BKM PROTOCOL

In this section we study whether IEEE BKM satisfies the definitions of privacy and unforgeability defined above, and under which conditions.

As the butterfly signing keys are derived from the same caterpillar signing key and the cocoon encryption keys are derived from the same caterpillar encryption key, rather than independently generated, we cannot prove security of the protocol assuming the standard security definitions for the encryption and signature schemes. Accordingly, we start with recalling the security definitions for encryption and signatures in the presence of related keys. For us, a weak version of the related-key attack security, where one considers a particular additive relation function, is sufficient. Please see the Introduction for the prior work references regarding related-key attacks.

5.1 Security for Signatures and Encryption in Presence of Related Keys

Signatures in the presence of related keys. A signature key derivation function specifies how the related keys are generated.

Signature Key Derivation Function. Let $\mathcal{DS} = (\mathcal{G}, \mathcal{K}, \mathcal{S}, \mathcal{V})$ be a digital signature scheme. Let (sPKExp, sSKExp) denote a pair of key expansion algorithms associated with $\mathcal{DS}$ and the randomness space Coins. sSKExp takes a secret key sk and randomness $r \in$ Coins, then outputs the derived secret key sk'; sPKExp takes a public key pk and randomness $r \in$ Coins, then outputs the derived public key pk'. For correctness, we require that for all I output by $\mathcal{G}(1^\lambda)$, all (pk, sk) output by $\mathcal{DS}.\mathcal{K}(I)$, for all randomness $r \in$ Coins, and for all $m \in$ MsgSp, let $sk' \leftarrow$ sSKExp$(sk; r)$, and $pk' \leftarrow$ sPKExp$(pk; r)$, we have

$$\Pr\left[\ \mathcal{DS}.\mathcal{V}\big(pk', \mathcal{DS}.\mathcal{S}(sk', m)\big) \Rightarrow 1\ \right] = 1.$$

Unforgeability under weakly related key attacks. Let $\mathcal{DS} = (\mathcal{G}, \mathcal{K}, \mathcal{S}, \mathcal{V})$ be a digital signature scheme and let (sSKExp, sPKExp) be a pair of key expansion functions with randomness space Coins. Consider the experiment defined in Figure 8. We note that the adversary is not given the expanded public keys explicitly because it can compute them itself. The adversary's advantage $\mathbf{Adv}^{\text{uf-wrka}}_{\mathcal{DS},(\text{sPKExp,sSKExp})}(\mathcal{A})$ is $\Pr\left[\mathbf{Exp}^{\text{uf-wrka}}_{\mathcal{DS}}(\mathcal{A}) \Rightarrow 1\right]$.

$\underline{\mathbf{Exp}^{\text{uf-cma}}_{\mathcal{BKM}}(\mathcal{A})}$

$(pk_{\text{cp}}, sk_{\text{cp}}, k_{\mathcal{EXP}}) \xleftarrow{\$} \mathcal{CKG}(1^\lambda)$; $(pk_{\text{aca}}, sk_{\text{aca}}) \xleftarrow{\$} \mathcal{ACAKG}(1^\lambda)$
$\mathcal{A} \leftarrow (pk_{\text{cp}}, k_{\mathcal{EXP}}, sk_{\text{aca}}, pk_{\text{aca}})$
For $i = 1, ..., n$: $M[i] \leftarrow \emptyset$
$[EEout, Aout] \xleftarrow{\$} [\mathcal{CKE}_{\text{EE}}(sk_{\text{cp}}, k_{\mathcal{EXP}}), \mathcal{A}]$
Parse $EEout$ as $(\boldsymbol{sk_{cc}}, \boldsymbol{pk_{cc}})$
If does not parse, return $\perp$
$[EEout, Aout] \xleftarrow{\$} [\mathcal{BKR}_{\text{EE}}(\boldsymbol{sk_{cc}}, pk_{\text{aca}}), \mathcal{A}]$
Parse $EEout$ as $(\boldsymbol{sk_{bf}}, \boldsymbol{pk_{bf}}, \boldsymbol{cert})$
If does not parse, return $\perp$
$(m, \sigma, j) \xleftarrow{\$} \mathcal{A}^{\text{Sign}(\cdot,\cdot)}(\boldsymbol{pk_{bf}}, \boldsymbol{cert})$
Return 1 iff $m \notin M[j]$ and $\mathcal{DS}.\mathsf{V}(\boldsymbol{pk_{bf}}[j], m, \sigma) = 1$

$\underline{\text{Sign}^{\mathcal{DS}}(m, j)}$

$\sigma \xleftarrow{\$} \mathcal{DS}.\mathcal{S}(\boldsymbol{sk_{bf}}[j], m)$
$M[i] \leftarrow M[i] \cup \{m\}$
Return σ

Figure 6: The experiment $\mathbf{Exp}^{\text{uf-cma}}_{\mathcal{BKM}}(\mathcal{A})$ for defining unforgeablility with corrupted RA and ACA.

$\underline{\mathbf{Exp}^{\text{priv-cra}}_{\mathcal{BKM}}(\mathcal{A})}$

$b \xleftarrow{\$} \{0, 1\}$
$(pk^A_{\text{cp}}, sk^A_{\text{cp}}, k^A_{\mathcal{EXP}}) \xleftarrow{\$} \mathcal{CKG}(1^\lambda)$
$(pk^B_{\text{cp}}, sk^B_{\text{cp}}, k^B_{\mathcal{EXP}}) \xleftarrow{\$} \mathcal{CKG}(1^\lambda)$
$(pk_{\text{aca}}, sk_{\text{aca}}) \xleftarrow{\$} \mathcal{ACAKG}(1^\lambda)$
$[EEout, Aout] \xleftarrow{\$} [\mathcal{CKE}_{\text{EE}}(sk^A_{\text{cp}}, k^A_{\mathcal{EXP}}), \mathcal{A}(pk^A_{\text{cp}}, k^A_{\mathcal{EXP}}, pk_{\text{aca}})]$
Parse $EEout$ as $(\boldsymbol{sk_{cc}}^A, \boldsymbol{pk_{cc}}^A)$
If does not parse, return $\perp$
$[EEout, Aout] \xleftarrow{\$} [\mathcal{CKE}_{\text{EE}}(sk^B_{\text{cp}}, k^B_{\mathcal{EXP}}), \mathcal{A}(pk^B_{\text{cp}}, k^B_{\mathcal{EXP}})]$
Parse $EEout$ as $(\boldsymbol{sk_{cc}}^B, \boldsymbol{pk_{cc}}^B)$
If does not parse, return $\perp$
$[Aout, \perp] \xleftarrow{\$} [\mathcal{A}, \mathcal{BKG}_{\text{ACA}}(sk_{\text{aca}}, \boldsymbol{pk_{cc}}^A, \boldsymbol{pk_{cc}}^B)]$
$[EEout, Aout] \xleftarrow{\$} [\mathcal{BKR}_{\text{EE}}(sk^A_{\text{cc}1}, ..., sk^A_{\text{cc}n}, pk_{\text{aca}}), \mathcal{A}]$
Parse $EEout$ as $(sk^A_{\text{bf}1}, ..., sk^A_{\text{bf}n}, pk^A_{\text{bf}1}, ..., pk^A_{\text{bf}n}, cert^A_1, ..., cert^A_n)$
If does not parse, return $\perp$
$[EEout, Aout] \xleftarrow{\$} [\mathcal{BKR}_{\text{EE}}(sk^B_{\text{cc}1}, ..., sk^B_{\text{cc}n}, pk_{\text{aca}}), \mathcal{A}]$
Parse $EEout$ as $(sk^B_{\text{bf}1}, ..., sk^B_{\text{bf}n}, pk^B_{\text{bf}1}, ..., pk^B_{\text{bf}n}, cert^B_1, ..., cert^B_n)$
If does not parse, return $\perp$
$(pk_1, cert_1) \xleftarrow{\$} \{(pk^A_{\text{bf}1}, cert^A_1), ..., (pk^A_{\text{bf}n}, cert^A_n)\}$
If $b = 0$:
$(pk_2, cert_2) \xleftarrow{\$} \{(pk^A_{\text{bf}1}, cert^A_1), ..., (pk^A_{\text{bf}n}, cert^A_n)\} \setminus \{(pk_1, cert_1)\}$
Else:
$(pk_2, cert_2) \xleftarrow{\$} \{(pk^B_{\text{bf}1}, cert^B_1), ..., (pk^B_{\text{bf}n}, cert^B_n)\}$
$d \xleftarrow{\$} \mathcal{A}(pk^A_{\text{cp}}, pk^B_{\text{cp}}, pk_1, pk_2, cert_1, cert_2)$
Return 1 iff $b = d$

Figure 7: The experiment $\mathbf{Exp}^{\text{priv-cra}}_{\mathcal{BKM}}(\mathcal{A})$ for defining privacy with corrupted RA.

Encryption in the presence of related keys.

Encryption Key Derivation Function. Let $\mathcal{AE} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be an public-key encryption scheme. Let $(\mathsf{eSKExp}, \mathsf{ePKExp})$ be a pair of key expansion algorithms associated with $\mathcal{AE}$ and the randomness space Coins. eSKExp takes a secret key sk and randomness r, then returns the expanded secret key sk'; similarly, ePKExp takes a public key pk and randomness r, then outputs the expanded public key pk'. Correctness requires that for all I output by $\mathcal{G}(1^\lambda)$, all (pk, sk) output by $\mathcal{AE}.\mathcal{K}(I)$, for all randomness $r \in \text{Coins}$, and for all $m \in \mathsf{MsgSp}$, let $sk' \leftarrow \mathsf{eSKExp}(sk; r)$, and $pk' \leftarrow \mathsf{ePKExp}(pk; r)$ we have that

$$\Pr\left[\ \mathcal{AE}.\mathcal{D}\big(sk', \mathcal{AE}.\mathcal{E}(pk', m)\big) \Rightarrow m\ \right] = 1.$$

IND-CPA under related key attack. Let $\mathcal{AE} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be a public key encryption scheme, $(\mathsf{ePKExp}, \mathsf{eSKExp})$ be a pair of key expansion functions. The security experiment is presented in Figure 9. Again, the public keys are not explicitly given to $\mathcal{A}$ as they can be computed. Note that the coins are generated honestly, as for our proofs we do not require security agagist maliciously-generated coins. For a stateful attacker $\mathcal{A}$, we define its advantage $\mathbf{Adv}^{\text{ind-cpa-wrka}}_{\mathcal{AE},(\mathsf{ePKExp},\mathsf{eSKExp})}(\mathcal{A})$ as $2\Pr\left[\mathbf{Exp}^{\text{ind-cpa-wrka}}_{\mathcal{AE},(\mathsf{ePKExp},\mathsf{eSKExp})}(\mathcal{A}) \Rightarrow 1\right] - 1$.

5.2 Target Robustness

It turns out that security of the IEEE BKM protocol requires an additional non-standard notion of security for the base encryption scheme, called *robustness* [6, 17]. Robustness captures the inability to create ciphertexts which are valid wrt different keys. Abdalla et al. [6] defined robustness and proved that DHIES (and hence ECIES), with small modification, is robust. However, taking a closer look we can see that the robustness definition in [6] is for properly generated keys, in that pk'_i has to be valid. But in our experiment there is no restriction on the validity of the public keys the malicious RA sends to the ACA. Farshim et al. [17] revisited the definitions of robustness and provided stronger robustness definitions which take into account public keys which can be invalid. However, the adversary in their definition is required to output the corresponding secret keys. This is a problem for us since the malicious RA in our experiment will not provide the secret key in any way. Luckily, we are only concerned about the ciphertext being valid under one of the two keys. Strictly speaking, the other key in our experiment is not generated according to the base encryption scheme key generation algorithm. But we can deal with this discrepancy easily in the ideal cipher model. Below we provide the definition of robustness capturing this setting, that we call *target robustness* (trob). Later, we prove that ECIES, with small modifications, satisfies target robustness.

Target Robustness Definition. Let $\mathcal{AE} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be a public key encryption scheme. Figure 10 defines the experiment $\mathbf{Exp}^{\text{trob}}_{\mathcal{AE}}(\mathcal{A})$ associated with an adversary $\mathcal{A}$. The adversary's advantage, $\mathbf{Adv}^{\text{trob}}_{\mathcal{DS}}(\mathcal{A})$ is defined as $\Pr\left[\mathbf{Exp}^{\text{trob}}_{\mathcal{AE}}(\mathcal{A}) \Rightarrow 1\right]$.

$\underline{\mathbf{Exp}^{\text{uf-wrka}}_{\mathcal{DS}}(\mathcal{A})}$

$I \xleftarrow{\$} \mathcal{DS}.\mathcal{G}(1^\lambda)$
$(pk, sk) \xleftarrow{\$} \mathcal{DS}.\mathcal{K}(I)$
$Ql \leftarrow \emptyset$
$(a_1, ..., a_n) \xleftarrow{\$} \mathcal{A}(pk)$
For $i = 1, ..., n$:
  $sk_i \leftarrow \text{sSKExp}(sk; a_i)$
  $pk_i \leftarrow \text{sPKExp}(pk; a_i)$
$(m, \sigma, i) \xleftarrow{\$} \mathcal{A}^{\text{Sign}(\cdot,\cdot)}$
Return 1 iff $(i, m) \notin Ql$ and
  $\mathcal{DS}.\mathcal{V}(pk_i, m, \sigma) = 1$

$\underline{\text{Sign}^{\mathcal{DS}}(m, i)}$

$\sigma \xleftarrow{\$} \mathcal{DS}.\mathcal{S}(sk_i, m)$
$Ql \leftarrow Ql \cup \{(i, m)\}$
Return σ

Figure 8: The experiment $\mathbf{Exp}^{\text{uf-wrka}}_{\mathcal{DS},(\text{sPKExp,sSKExp})}(\mathcal{A})$.

$\underline{\mathbf{Exp}^{\text{ind-cpa-wrka}}_{\mathcal{AE},(\text{ePKExp,eSKExp})}(\mathcal{A})}$

$b \xleftarrow{\$} \{0, 1\}$; $I \xleftarrow{\$} \mathcal{G}(1^\lambda)$
$(pk, sk) \xleftarrow{\$} \mathcal{AE}.\mathcal{K}(I)$
$(a_1, \ldots, a_n) \xleftarrow{\$}$ Coins
For $i = 1, \ldots n$:
  $sk_i \leftarrow \text{eSKExp}(sk; a_i)$
  $pk_i \leftarrow \text{ePKExp}(pk; a_i)$
$d \xleftarrow{\$} \mathcal{A}^{\text{Enc}(\cdot,\cdot,\cdot)}(pk, a_1, ..., a_n)$
Return 1 iff $(b = d)$

$\underline{\text{procedure Enc}(m_0, m_1, i)}$

If $|m_0| = |m_1|$:
  $c \xleftarrow{\$} \mathcal{AE}.\mathcal{E}(pk_i, m_b)$
Else:
  $c \leftarrow \perp$
Return c

Figure 9: The experiment $\mathbf{Exp}^{\text{ind-cpa-wrka}}_{\mathcal{AE},(\text{ePKExp,eSKExp})}(\mathcal{A})$.

$\underline{\mathbf{Exp}^{\text{trob}}_{\mathcal{AE}}(\mathcal{A})}$

$I \xleftarrow{\$} \mathcal{G}(1^\lambda)$
$(pk, sk) \xleftarrow{\$} \mathcal{K}(I)$
$(pk', m) \leftarrow \mathcal{A}(pk)$
$c \leftarrow \mathcal{E}(pk', m)$
Return 1 iff $pk \neq pk'$ and $\mathcal{D}(sk, c) \neq \perp$

Figure 10: Experiment $\mathbf{Exp}^{\text{trob}}_{\mathcal{AE}}(\mathcal{A})$ for defining target robustness (TROB).

5.3 Generic IEEE BKM Security

We start with analyzing the generic IEEE BKM (based on generic signature and encryption schemes). In the next section we analyze the specific instantiations of the base schemes.

Theorem 5.1 (Privacy, Corrupted RA). *Let $\mathcal{BKM}$ be the IEEE Butterfly Key Certificate protocol as defined in Section 3.3, associated with signatures schemes $\mathcal{DS}_1$, $\mathcal{DS}_2$, asymmetric encryption $\mathcal{AE}$, the expansion function $\mathcal{EXP}$ and the hash function $\mathcal{H}$. Let $\mathcal{AE}$ be an elliptic-curve Diffie-Hellman based scheme (cf. the explanation in Section 3.3). Let* $(\text{ePKExp}_+, \text{eSKExp}_+)$ *be the additive keys expansion functions for encryption described in Figure 11 associated with $\mathcal{AE}$ and Coins $= \{0, 1\}^*$. If $\mathcal{EXP}$ is modeled as an ideal cipher and H is modeled as random oracles, then for every efficient adversary $\mathcal{A}$ there exist efficient $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$ such that*

$$\begin{aligned}\mathbf{Adv}^{\text{priv-cra}}_{\mathcal{BKM}}(\mathcal{A}) \quad &\leq \quad 2\mathbf{Adv}^{\text{ind-cpa-wrka}}_{\mathcal{AE},(\text{ePKExp}_+,\text{eSKExp}_+)}(\mathcal{B}) \\ &+ \quad 2\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{DS}_2}(\mathcal{C}) + 4n \cdot \mathbf{Adv}^{\text{trob}}_{\mathcal{AE}}(\mathcal{D}) + negl(\lambda)\ .\end{aligned}$$

The proof is in Appendix B.1. Recall that the above result captures privacy of EEs' certificates from the same ACA.

$\underline{\text{eSKExp}_+(sk; a)}$
$sk' \leftarrow sk + a$
Return sk'

$\underline{\text{ePKExp}_+(pk; a)}$
$pk' \leftarrow pk + a$
Return pk'

Figure 11: Additive key expansion functions for $\mathcal{AE}$.

$\underline{\text{sSKExp}_+(sk; a)}$
$sk' \leftarrow sk + H'(a)$
Return sk'

$\underline{\text{sPKExp}_+(pk; a)}$
$pk' \leftarrow pk + H'(a)G$
Return pk'

Figure 12: Additive key expansion functions for $\mathcal{DS}_1$.

Theorem 5.2 (Privacy, Corrupted ACA). *Let $\mathcal{BKM}$ be the IEEE Butterfly Key Certificate protocol be as defined in Section 3.3, associated with signatures schemes $\mathcal{DS}_1$, $\mathcal{DS}_2$, asymmetric encryption $\mathcal{AE}$, the expansion function $\mathcal{EXP}$ and the hash function $\mathcal{H}$. If $\mathcal{EXP}$ is modeled as an ideal cipher, then for every efficient adversary $\mathcal{A}$,*

$$\mathbf{Adv}^{\text{priv-caca}}_{\mathcal{BKM}}(\mathcal{A}) \leq negl(\lambda) .$$

The proof is in Appendix B.2.

Theorem 5.3 (Unforgeability, Corrupted ACA and RA). *Let $\mathcal{BKM}$ be the IEEE Butterfly Key Certificate protocol be as defined in Section 3.3, associated with signatures schemes $\mathcal{DS}_1$, $\mathcal{DS}_2$, asymmetric encryption $\mathcal{AE}$, the expansion function $\mathcal{EXP}$ and the hash function $\mathcal{H}$. Let $\mathcal{DS}_1$ be an elliptic-curve Diffie-Hellman based scheme (cf. the explanation in Section 3.3). Let $(\text{sPKExp}_+, \text{sSKExp}_+)$ be the additive keys expansion functions for signatures described in Figure 12 associated with $\mathcal{DS}_1$, Coins $= \{0,1\}^*$ and the hash function $\mathcal{H}' \colon \{0,1\} \to Z_q$. If $\mathcal{EXP}$ is modeled as an ideal cipher and H, H' are modeled as random oracles, then for every efficient adversary $\mathcal{A}$ there exist an efficient adversary $\mathcal{B}$, such that*

$$\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{BKM}}(\mathcal{A}) \leq \mathbf{Adv}^{\text{uf-wrka}}_{\mathcal{DS}_1,(\text{sPKExp}_+,\text{sSKExp}_+)}(\mathcal{B}) .$$

The proof is in Appendix B.3.

Remark. In the case of honest EE and RA, and corrupted ACA, we can drop the ideal blockcipher assumption on cocoon key expansion functions in Theorem 5.3 as the adversary no longer participates in the cocoon key expansion stage. In the case of honest EE and ACA, and corrupted RA, Theorem 5.3 holds without the protocol's modification, since all random offsets are honestly generated by the honest ACA.

6 SECURITY ANALYSIS OF THE IEEE-BKM INSTANTIATION

In this section we conclude the security analysis of IEEE-BKCP protocol. Recall that IEEE-BKCP uses AES as $\mathcal{EXP}$, ECDSA as $\mathcal{DS}_1, \mathcal{DS}_2$, and ECIES as $\mathcal{AE}$ with the underlying symmetric encryption AES-CCM. It is common in the security analyses to assume that AES is an ideal cipher and the hash is a random oracle. ECDSA has been proven to be uf-cma secure under various idealized assumptions on the underlying primitives [15, 18, 19]. The CCM mode has been analyzed in [20, 23]. But it remains to study ind-cpa-wrka and trob security of ECIES and uf-wrka security of ECDSA.

6.1 ECIES

The asymmetric encryption scheme $\mathcal{ECIES}$ is an elliptic-curve variant of DHIES [7]. It is associated with a symmetric encryption scheme $\mathcal{SE}$[4]. Let $\mathcal{HK} : \mathbb{G} \to \{0,1\}^{hl}$ be a hash function. The Kg algorithm of the asymmetric encryption scheme $\mathcal{ECIES}$ returns the elliptic curve parameters $I = (\mathbb{G}, G, q)$. Figure 13 recalls the rest of the scheme's algorithms. Let the length the key for $\mathcal{SE}$ be hl. The message space of $\mathcal{ECIES}$ is that of $\mathcal{SE}$.

[4]DHIES was designed to also use a message authentication code (MAC). The use of MACs is not essential to us as we do not consider IND-CCA security. Moreover, IEEE BKM's implementation uses ECIES with the associated symmetric scheme in CCM mode, which is an authenticated encryption scheme.

To prove ind-cpa-wrka security of ECIES we will need to rely on Hash Diffie-Hellman (HDH) assumption [7]. We recall the HDH problem.

HDH Problem. Let $\mathcal{G}$ be the global info generation and let $\mathcal{H}$ be a hash function. The experiment $\mathbf{Exp}^{\text{hdh}}_{\mathcal{G}}(\mathcal{A})$ is defined in Figure 14. The advantage of the adversary $\mathcal{A}$, $\mathbf{Adv}^{\text{hdh}}_{\mathcal{G}}(\mathcal{A}) = 2 \cdot \Pr\left[\mathbf{Exp}^{\text{hdh}}_{\mathcal{G}}(\mathcal{A}) \Rightarrow 1\right] - 1$.

Theorem 6.1 (ind-cpa-wrka security of $\mathcal{ECIES}$). *Let $\mathcal{ECIES}$ be the encryption scheme recalled in Figure 13, with global info generation $\mathcal{G}$. The scheme is associated with a symmetric encryption scheme $\mathcal{SE}$, and hash $\mathcal{HK} : \mathbb{G} \to \{0,1\}^{hl}$. Let $(\text{ePKExp}_+, \text{eSKExp}_+)$ be the additive encryption key expansion functions described in Figure 11. Then for any efficient adversary $\mathcal{A}$ there exist efficient adversaries $\mathcal{B}$ and $\mathcal{D}$ such that*

$$\mathbf{Adv}^{\text{ind-cpa-wrka}}_{\mathcal{ECIES},(\text{ePKExp}_+,\text{eSKExp}_+)}(\mathcal{A}) \leq n \cdot \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(\mathcal{B}) + 2\mathbf{Adv}^{\text{hdh}}_{\mathcal{HK},\mathcal{G}}(\mathcal{D}) .$$

The proof is in Appendix B.4.

IEEE BKM uses AES-CCM mode as $\mathcal{SE}$. The CCM mode has been proven secure in [20, 23] assuming AES is a PRF.

We now prove that ECIES with minor modifications (ECIES* in Figure 13) is TROB-secure.

Theorem 6.2 (trob security of $\mathcal{ECIES}$). *Let $\mathcal{ECIES}^*$ be the modified ECIES recalled in Figure 13. The scheme is associated with $\mathcal{SE}$ being the CCM mode and hash $\mathcal{HK}$. Then for any efficient adversary $\mathcal{A}$, if the blockcipher used in CCM modeled as an ideal cipher, we have*

$$\mathbf{Adv}^{\text{trob}}_{\mathcal{ECIES}^*}(\mathcal{A}) \leq \text{negl}(\lambda) .$$

Proof. There are two possibilities for a ciphertext $vG \| e$ created under one public key X_1 to be valid under a different public key X_2. Let $k^1_{\mathcal{SE}} \leftarrow \mathcal{HK}(vX_1)$, $k^2_{\mathcal{SE}} \leftarrow \mathcal{HK}(vX_2)$.

(1) $\mathcal{HK}(vX_1) = \mathcal{HK}(vX_2)$
(2) $\mathcal{SE}.\mathcal{D}(k^2_{\mathcal{SE}}, e) \neq \perp$

The first condition has only a negligible probability of occurring if $\mathcal{HK}$ is a random oracle and $vX_1 \neq vX_2$. And the latter is true since $X_1 \neq X_2$, and $V \neq \mathbf{1}$, $X_1 \neq \mathbf{1}$ and $X_2 \neq \mathbf{1}$, where $V = vG$.

The second condition has only a negligible probability of occurring since in the ideal cipher model, the blockcipher under a different key is an independent random permutation, and the MAC, which is part of the CCM's mode construction would have only a negligible probability of being valid. □

6.2 ECDSA

Despite its wide use, the ECDSA signature scheme has not been proven secure against related key attacks until recently. Groth and Shoup [21] showed that ECDSA is secure when keys are computed via additive key derivation function, in the elliptic curve generic group model. We use their result to bound $\mathbf{Adv}^{\text{uf-wrka}}_{\mathcal{ECDSA},(\text{sPKExp}_+,\text{sSKExp}_+)}(\mathcal{A})$, in the random oracle model.

Theorem 6.3 (uf-wrka security of $\mathcal{ECDSA}$, from [[21]]). *Let $\mathcal{ECDSA}$ be the ECDSA signature scheme (recalled in the Appendix A), whose $\mathcal{G}$ algorithm returns $I = (\mathbb{G}, G, q)$. Let*

$\underline{\mathcal{ECIES}.\mathcal{K}((\mathbb{G},G,q))}$
$u \xleftarrow{\$} \mathbb{Z}_q$
$pk \leftarrow uG$
△ If $pk = \mathbf{1}$ then return $\perp$
$sk \leftarrow u$
Return (pk, sk)

$\underline{\mathcal{ECIES}.\mathcal{E}(pk, m)}$
$v \xleftarrow{\$} \mathbb{Z}_q$
$V \leftarrow vG$
$x \leftarrow v \cdot pk$
$k_{\mathcal{SE}} \leftarrow \mathcal{HK}(x)$
$e \xleftarrow{\$} \mathcal{SE}.\mathcal{E}(k_{\mathcal{SE}}, m)$
Return $V \| e$

$\underline{\mathcal{ECIES}.\mathcal{D}(sk, c)}$
$V \| e \leftarrow c$
△ If $V = \mathbf{1}$ then return $\perp$
$x \leftarrow sk \cdot V$
$k_{\mathcal{SE}} \leftarrow \mathcal{HK}(x)$
If $\perp = \mathcal{SE}.\mathcal{D}(k_{\mathcal{SE}}, e)$
  Return $\perp$
$m \leftarrow \mathcal{SE}.\mathcal{D}(k_{\mathcal{SE}}, e)$
Return m

Figure 13: ECIES encryption scheme does not include lines in red. The modified ECIES* scheme includes lines in red (also marked with △ symbol.)

$\underline{\mathbf{Exp}^{\mathsf{hdh}}_{\mathcal{G}}(\mathcal{A})}$
$b \xleftarrow{\$} \{0,1\}$
$(\mathbb{G}, g, q) \xleftarrow{\$} \mathcal{G}(1^\lambda)$
$I \leftarrow (\mathbb{G}, g, q)$
$u \xleftarrow{\$} \mathbb{Z}_q; U \leftarrow uG$
$v \xleftarrow{\$} \mathbb{Z}_q; V \leftarrow vG$
If $b = 0$ then $Z \xleftarrow{\$} \{0,1\}^{hl}$
Else $Z \leftarrow \mathcal{H}(uvG)$
$d \leftarrow \mathcal{A}^{\mathcal{H}}(I, U, V, Z)$
Return 1 iff $b = d$

Figure 14: Experiment $\mathbf{Exp}^{\mathsf{hdh}}_{\mathcal{G}}(\mathcal{A})$ for hdh problem.

$(\mathsf{sPKExp}_+, \mathsf{sSKExp}_+)$ *be the additive expansion functions described in Figure 12 where H is the random oracle with the range* $\mathbb{Z}_q$*. For simplicity we assume that* $\mathcal{ECDSA}$ *uses the same H. Then in the elliptic curve generic group model (EC-GGM), and in the random oracle model, for any efficient adversary* $\mathcal{A}$*,*

$$\mathbf{Adv}^{\mathsf{uf\text{-}wrka}}_{\mathcal{ECDSA},(\mathsf{sPKExp}_+,\mathsf{sSKExp}_+)}(\mathcal{A}) \leq (8+o(1))q_s \cdot q_H^2/q + O(q_s^2/q)\ ,$$

where q_s *is the number of signing or group queries and* q_H *is the number of random oracle queries the adversary makes.*

Theorems 5.1, 5.2, 5.3, 6.1, 6.2, 6.3 together complete the security results for the IEEE BKM protocol.

7 EFFICIENCY IMPROVEMENT FOR IEEE BKM

Proposed modification. We propose a simple change to the BKM protocol that yields a significant efficiency improvement. More specifically, we propose that the ACA can re-use randomness when encrypting certificate responses using ECIES under different cocoon encryption keys. Recall that the first part of an ECIES ciphertext is vG, where $v \in \mathbb{Z}_q$ has to be picked at random for each encryption. We show that the ACA can re-use the same v across all ciphertexts. This will result in reducing the computation in half (as the ACA will have to perform only $N+1$ scalar elliptic curve multiplications as opposed to $2N$, where N is the number of encryptions the ACA performs) and significantly reducing the communication (since vG can be sent only once to each EE).

Kurosawa, Bellare et al. and Barbosa and Farshim [8, 10, 24] studied the problem of secure randomness re-use. Bellare et al. [10] defined the security notion for asymmetric *multi-recipient* encryption schemes (MRES) that, unlike the notion in [24], lets the attacker to corrupt some users and learn their secret keys. They also proved that randomness can be safely re-used across multiple DHIES encryptions under different public keys. Their result applies to ECIES as well, however, we cannot use their result as is. The reason is the public keys in our application are related, and the results of [8, 10] do not cover this case. Their results apply only to the case of independently-created keys, and in general, security does not extend to the case of related keys. We show that the MRES with ECIES and randomness re-use is secure in the setting with the related keys, assuming hardness of the Oracle Diffie-Hellman problem and IND-CPA security of the underlying symmetric encryption scheme (cf. Theorem 7.2). Finally, we show that this is what we need for unlinkability of the modified IEEE BKM.

Multi-Recipient Encryption Scheme with Related Keys (RK-MRES). Let $\mathcal{AE} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be an asymmetric encryption scheme with the message space MsgSp. Let (ePKExp, eSKExp) be the key expansion functions. The associated RK-MRES is defined by the following algorithms. $\mathcal{RKMRES}.\mathcal{K}$ on input 1^λ returns the global info I. $\mathcal{RKMRES}.\mathcal{K}$ takes input global info I and does the following. $(pk, sk) \xleftarrow{\$} \mathcal{AE}.\mathcal{K}(I)$, $(a_1, ..., a_n) \xleftarrow{\$}$ Coins, For $i = 1, ..., n$: $\boldsymbol{sk}[i] \leftarrow \mathsf{eSKExp}(sk; a_i)$, $\boldsymbol{pk}[i] \leftarrow \mathsf{ePKExp}(pk; a_i)$, and returns $(\boldsymbol{pk}, \boldsymbol{sk})$. $\mathcal{RKMRES}.\mathcal{E}$ takes inputs $\boldsymbol{pk}$ and $\boldsymbol{m}$ and returns $\mathbf{c}$. $\mathcal{RKMRES}.\mathcal{D}$ is the same as $\mathcal{AE}.\mathcal{D}$.

Correctness of the scheme requires that the following experiment always returns 1. $I \xleftarrow{\$} \mathcal{RKMRES}.\mathcal{K}(1^\lambda)$, $(\boldsymbol{pk}, \boldsymbol{sk}) \xleftarrow{\$} \mathcal{RKMRES}.\mathcal{K}(I)$, $\boldsymbol{m} \xleftarrow{\$}$ MsgSp, $\mathbf{c} \xleftarrow{\$} \mathcal{RKMRES}.\mathcal{E}(\boldsymbol{m})$, $i \xleftarrow{\$} \{1, \ldots, n\}$, $\boldsymbol{m}[i] \leftarrow \mathcal{RKMRES}.\mathcal{D}(\mathbf{c}[i])$.

Randomness re-using ECIES RK-MRES. We are interested in a specific WK-MRES scheme, $\mathcal{MR}\text{-}\mathcal{ECIES}$, obtained from $\mathcal{ECIES}$ (cf. Figure 13). by using the same coins to encrypt different messages in the message vector, and associated with the additive key expansion functions $(\mathsf{eSKExp}_+, \mathsf{ePKExp}_+)$. We present the construction explicitly for clarity. Let $\mathcal{ECIES}$.Coins denote the randomness set for $\mathcal{ECIES}$. $\mathcal{MR}\text{-}\mathcal{ECIES}.\mathcal{G}(1^\lambda)$ runs $\mathcal{ECIES}.\mathcal{G}(1^\lambda)$. The rest of the algorithms are in Figure 15. The scheme is correct by correctness of $\mathcal{ECIES}$ and of the expansion functions.

Security analysis. To assess security of this scheme, we need to introduce a new security definition that takes into account both randomness re-use and related keys. We adapt the security definition from [10] to accommodate related keys. The adversary knows (or can compute) all public keys and how the keys are related. It

$\underline{\mathcal{MR}\text{-}\mathcal{ECIES}.\mathcal{K}(I)}$
$(pk, sk) \xleftarrow{\$} \mathcal{ECIES}.\mathcal{K}(I)$
For $i = 1, ..., n$
$a_i \xleftarrow{\$} \mathbb{Z}_q$
$\boldsymbol{pk}[i] \leftarrow \mathsf{ePKExp}_+(pk; a_i)$
$\boldsymbol{sk}[i] \leftarrow \mathsf{eSKExp}_+(sk; a_i)$
Return $(\boldsymbol{pk}, \boldsymbol{sk})$

$\underline{\mathcal{MR}\text{-}\mathcal{ECIES}.\mathcal{E}(\boldsymbol{pk}, \boldsymbol{m})}$
$r \xleftarrow{\$} \mathcal{ECIES}.\mathrm{Coins}$
For $i = 1, ..., n$
$\mathbf{c}[i] \leftarrow \mathcal{ECIES}.\mathcal{E}(\boldsymbol{pk}[i], \boldsymbol{m}[i]; r)$
Return $\mathbf{c}$

$\underline{\mathcal{MR}\text{-}\mathcal{ECIES}.\mathcal{D}(sk, c)}$
$m \leftarrow \mathcal{ECIES}.\mathcal{D}(sk, c)$
Return m

Figure 15: Randomness re-using multi-recipient encryption scheme with related keys, based on public key encryption scheme $\mathcal{ECIES}$ and key expansion functions ($\mathsf{ePKExp}_+, \mathsf{eSKExp}_+$).

is allowed to learn some secret keys, and is trying to get some information about the messages encrypted under the other public keys.

Definition 7.1. Let $\mathsf{MRPKE} = (\mathcal{G}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be a multi-recipient asymmetric encryption scheme based on public key encryption scheme $\mathcal{AE}$ and key expansion functions (ePKExp, eSKExp). We define the experiment $\mathbf{Exp}^{\text{mr-cpa-wrka}}_{\mathsf{MRPKE}}(\mathcal{A})$ in Figure 16. The attacker's advantage, $\mathbf{Adv}^{\text{mr-cpa-wrka}}_{\mathsf{MRPKE}}(\mathcal{A})$ is $2\Pr\left[\mathbf{Exp}^{\text{mr-cpa-wrka}}_{\mathsf{MRPKE}}(\mathcal{A}) \Rightarrow 1\right] - 1$.

$\underline{\mathbf{Exp}^{\text{mr-cpa-wrka}}_{\mathsf{MRPKE}}(\mathcal{A})}$
$b \xleftarrow{\$} \{0,1\}$
$(pk, sk) \xleftarrow{\$} \mathcal{AE}.\mathcal{K}(I)$
$(pk', sk') \xleftarrow{\$} \mathcal{AE}.\mathcal{K}(I)$
$\mathbf{a} \xleftarrow{\$} \mathrm{Coins}$
$\mathbf{a}' \xleftarrow{\$} \mathrm{Coins}$
For $i = 1, ..., n$
$\boldsymbol{sk}[i] \leftarrow \mathsf{eSKExp}(sk; \mathbf{a}[i])$
$\boldsymbol{pk}[i] \leftarrow \mathsf{ePKExp}(pk; \mathbf{a}[i])$
$\boldsymbol{sk}'[i] \leftarrow \mathsf{eSKExp}(sk'; \mathbf{a}'[i])$
$\boldsymbol{pk}'[i] \leftarrow \mathsf{ePKExp}(pk'; \mathbf{a}'[i])$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}) \xleftarrow{\$} \mathcal{A}(pk, pk', \mathbf{a}, \mathbf{a}', sk')$
If $|\mathbf{m}_0| \neq |\mathbf{m}_1|$ then return $\perp$
If $|\mathbf{m}_0| = |\mathbf{m}_1| = |\mathbf{m}'|$ is not n then return $\perp$
$\mathbf{c} \xleftarrow{\$} \mathsf{MRPKE}.\mathcal{E}(\boldsymbol{pk}, \mathbf{m}_b)$
$\mathbf{c}' \xleftarrow{\$} \mathsf{MRPKE}.\mathcal{E}(\boldsymbol{pk}', \mathbf{m})$
$d \xleftarrow{\$} \mathcal{A}(\mathbf{c}, \mathbf{c}')$
Return 1 iff $(b = d)$

Figure 16: Experiment for mr-cpa-wrka security of RK-MRES.

We now analyze the randomness re-using ECIES-based RK-MRES. Security relies on the Oracle Diffie-Hellman (ODH) problem [7], so we recall it here. It was proven in [7] that in the random oracle model Strong Diffie-Hellman (SDH) implies ODH.

ODH Problem. Let $\mathcal{G}$ be the global info generation and let $\mathcal{H}$ be a hash function. The experiment $\mathbf{Exp}^{\mathsf{odh}}_{\mathcal{G}}(\mathcal{A})$ is defined in Figure 17. The advantage of the adversary $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{odh}}_{\mathcal{G}}(\mathcal{A}) = 2 \cdot \Pr\left[\mathbf{Exp}^{\mathsf{odh}}_{\mathcal{G}}(\mathcal{A}) \Rightarrow 1\right] - 1$.

$\underline{\mathbf{Exp}^{\mathsf{odh}}_{\mathcal{G}}(\mathcal{A})}$
$b \xleftarrow{\$} \{0,1\}$
$(\mathbb{G}, g, q) \xleftarrow{\$} \mathcal{G}(1^\lambda)$
$I \leftarrow (\mathbb{G}, g, q)$
$u \xleftarrow{\$} \mathbb{Z}_q; U \leftarrow uG$
$v \xleftarrow{\$} \mathbb{Z}_q; V \leftarrow vG$
If $b = 0$ then $Z \xleftarrow{\$} \{0,1\}^{hl}$
Else $Z \leftarrow \mathcal{H}(uvG)$
$d \leftarrow \mathcal{D}^{\mathcal{H}, O_v(\cdot)}(I, U, V, Z)$
Return 1 iff $b = d$

$\underline{O_v(X)}$:
If $X = uG$ then return $\perp$
Return $H(vX)$

Figure 17: Experiment $\mathbf{Exp}^{\mathsf{odh}}_{\mathcal{G},\mathcal{H}}(\mathcal{A})$ for odh problem.

Theorem 7.2 ($\mathcal{MR}$-$\mathcal{ECIES}$ Security). *Let $\mathcal{MR}$-$\mathcal{ECIES}$ be the randomness re-using $\mathcal{ECIES}$-based RK-MRES scheme defined in Figure 16 and associated with the additive key expansion functions ($\mathsf{eSKExp}_+, \mathsf{ePKExp}_+$) described in Figure 11. Let $\mathcal{G}$ be the global info generation algorithm of the underlying $\mathcal{ECIES}$ scheme ($\mathcal{ECIES}$ is recalled in Figure 13). Let $\mathcal{SE}$ be the symmetric encryption scheme and let $\mathcal{HK}$ be the hash underlying $\mathcal{ECIES}$. Then, in the random oracle model, for any efficient adversary $\mathcal{A}$, there exist efficient adversaries $\mathcal{B}$ and $\mathcal{C}$ such that*

$$\mathbf{Adv}^{\text{mr-cpa-wrka}}_{\mathcal{MR}\text{-}\mathcal{ECIES}}(\mathcal{A}) \leq n \cdot \mathbf{Adv}^{\mathsf{odh}}_{\mathcal{G},\mathcal{HK}}(\mathcal{B}) + n \cdot \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(\mathcal{C}) \ .$$

The proof is in Appendix B.5.

Finally, we observe that the use of $\mathcal{MR}$-$\mathcal{ECIES}$ instead of ECIES only affects unlinkability in case of the corrupted RA. However, the statement of Theorem 5.1 still holds for $\mathcal{MR}$-$\mathcal{ECIES}$ and its security wrt to the mr-cpa-wrka notion.

8 CONCLUSIONS

We analyzed the Butterfly Key Mechanism protocol from IEEE 1609.2.1 standard, using the provable security approach. We formalized the goals of end entity privacy and their signing keys unforgeability. We considered different corruption scenarios and proved that the IEEE-BKM protocol, with small modifications satisfies our definitions under the appropriate assumptions on the building blocks. We also proposed a way to significantly improve the protocol's efficiency without sacrificing security. An interesting direction for future work will be studying ways to achieve post-quantum security of the protocol, following [13].

REFERENCES

[1] 2017. Federal Motor Vehicle Safety Standards; V2V Communications, National Highway Traffic Safety Administration (NHTSA), U.S. Department of Transportation (DOT). Federal Regester.

[2] 2020. IEEE Standard for Wireless Access in Vehicular Environments–Security Services for Application and Management Messages. IEEE 1609.2-2022, Active Standard. https://standards.ieee.org/ieee/1609.2/10258/.

[3] 2020. Security Credential Management System (SCMS) Proof–of–Concept Implementation End-Entity (EE) Requirements and Specifications Supporting SCMS Software Release 1.2.1. Department of Transportation. https://www.its.dot.gov/research_areas/cybersecurity/scms/index.html.

[4] 2021. IEEE Standard for Wireless Access in Vehicular Environments (WAVE) - Certificate Management Interfaces for End Entities. IEEE 1609.2.1-2022, Active Standard. https://standards.ieee.org/ieee/1609.2.1/10728/.

[5] 2023. Saving Lives with Connectivity: A Plan to Accelerate V2X Deployment. The U.S. Department of Transportation. https://its.dot.gov/research_areas/emerging_tech/pdf/Accelerate_V2X_Deployment.pdf.

[6] Michel Abdalla, Mihir Bellare, and Gregory Neven. 2010. Robust Encryption. In *Theory of Cryptography*, Daniele Micciancio (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 480–497.

[7] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. 2001. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In *Topics in Cryptology — CT-RSA 2001*, David Naccache (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 143–158.

[8] Manuel Barbosa and Pooya Farshim. 2007. Randomness Reuse: Extensions and Improvements. In *Cryptography and Coding*, Steven D. Galbraith (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 257–276.

[9] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. 2000. Public-Key Encryption in a Multi-user Setting: Security Proofs and Improvements. In *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding (Lecture Notes in Computer Science, Vol. 1807)*, Bart Preneel (Ed.). Springer, 259–274. https://doi.org/10.1007/3-540-45539-6_18

[10] Mihir Bellare, Alexandra Boldyreva, and Jessica Staddon. 2003. Randomness Reuse in Multi-recipient Encryption Schemeas. In *PKC (Lecture Notes in Computer Science, Vol. 2567)*. Springer, 85–99. https://doi.org/10.1007/3-540-36288-6_7

[11] Mihir Bellare, David Cash, and Rachel Miller. 2011. Cryptography Secure against Related-Key Attacks and Tampering. In *Advances in Cryptology – ASIACRYPT 2011*, Dong Hoon Lee and Xiaoyun Wang (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 486–503.

[12] Mihir Bellare and Tadayoshi Kohno. 2003. A Theoretical Treatment of Related-Key Attacks: RKA-PRPs, RKA-PRFs, and Applications. In *Advances in Cryptology — EUROCRYPT 2003*, Eli Biham (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 491–506.

[13] Nina Bindel, Sarah McCarthy, Geoff Twardokus, and Hanif Rahbari. 2022. Drive (Quantum) Safe! â€" Towards Post-Quantum Security for V2V Communications. *IACR Cryptol. ePrint Arch.* (2022), 483. https://eprint.iacr.org/2022/483

[14] Benedikt Brecht, Dean Therriault, André Weimerskirch, William Whyte, Virendra Kumar, Thorsten Hehn, and Roy Goudy. 2018. A Security Credential Management System for V2X Communications. *IEEE Transactions on Intelligent Transportation Systems* 19, 12 (2018), 3850–3871. https://doi.org/10.1109/TITS.2018.2797529

[15] Daniel R. L. Brown. 2005. Generic Groups, Collision Resistance, and ECDSA. *Des. Codes Cryptogr.* 35, 1 (2005), 119–152. https://doi.org/10.1007/S10623-003-6154-Z

[16] David Chaum and Eugène van Heyst. 1991. Group Signatures. In *Advances in Cryptology — EUROCRYPT '91*, Donald W. Davies (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 257–265.

[17] Pooya Farshim, Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. 2013. Robust Encryption, Revisited. In *Public-Key Cryptography – PKC 2013*, Kaoru Kurosawa and Goichiro Hanaoka (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 352–368.

[18] Manuel Fersch, Eike Kiltz, and Bertram Poettering. 2016. On the Provable Security of (EC)DSA Signatures. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi (Eds.). ACM, 1651–1662. https://doi.org/10.1145/2976749.2978413

[19] Manuel Fersch, Eike Kiltz, and Bertram Poettering. 2017. On the One-Per-Message Unforgeability of (EC)DSA and Its Variants. In *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 10678)*, Yael Kalai and Leonid Reyzin (Eds.). Springer, 519–534. https://doi.org/10.1007/978-3-319-70503-3_17

[20] Pierre-Alain Fouque, Gwenaëlle Martinet, Frédéric Valette, and Sébastien Zimmer. 2008. On the Security of the CCM Encryption Mode and of a Slight Variant. In *Applied Cryptography and Network Security, 6th International Conference, ACNS 2008, New York, NY, USA, June 3-6, 2008. Proceedings (Lecture Notes in Computer Science, Vol. 5037)*, Steven M. Bellovin, Rosario Gennaro, Angelos D. Keromytis, and Moti Yung (Eds.). 411–428. https://doi.org/10.1007/978-3-540-68914-0_25

[21] Jens Groth and Victor Shoup. 2022. On the Security of ECDSA with Additive Key Derivation and Presignatures. In *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 13275)*, Orr Dunkelman and Stefan Dziembowski (Eds.). Springer, 365–396. https://doi.org/10.1007/978-3-031-06944-4_13

[22] John Harding, Gregory H. Powell, Rebecca Yoon, Joshua Fikentscher, Charlene T Doyle, Dana Sade, Mike Lukuc, Jim Simons, and Jing Wang. 2014. Vehicle-to-Vehicle Communications: Readiness of V2V Technology for Application. https://api.semanticscholar.org/CorpusID:107533679

[23] Jakob Jonsson. 2002. On the Security of CTR + CBC-MAC. In *Selected Areas in Cryptography, 9th Annual International Workshop, SAC 2002, St. John's, Newfoundland, Canada, August 15-16, 2002. Revised Papers (Lecture Notes in Computer Science, Vol. 2595)*, Kaisa Nyberg and Howard M. Heys (Eds.). Springer, 76–93. https://doi.org/10.1007/3-540-36492-7_7

[24] Kaoru Kurosawa. 2002. Multi-recipient Public-Key Encryption with Shortened Ciphertext. In *Public Key Cryptography*, David Naccache and Pascal Paillier (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 48–63.

[25] Marcos A. Simplicio, Eduardo Lopes Cominetti, Harsh Kupwade Patil, Jefferson E. Ricardini, and Marcos Vinicius M. Silva. 2018. The Unified Butterfly Effect: Efficient Security Credential Management System for Vehicular Communications. In *2018 IEEE Vehicular Networking Conference (VNC)*. 1–8. https://doi.org/10.1109/VNC.2018.8628369

[26] Markus Stadler, Jean-Marc Piveteau, and Jan Camenisch. 1995. Fair Blind Signatures. In *Advances in Cryptology — EUROCRYPT '95*, Louis C. Guillou and Jean-Jacques Quisquater (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 209–219.

[27] William Whyte, André Weimerskirch, Virendra Kumar, and Thorsten Hehn. 2013. A security credential management system for V2V communications. In *2013 IEEE Vehicular Networking Conference, Boston, MA, USA, December 16-18, 2013*. IEEE, 1–8. https://doi.org/10.1109/VNC.2013.6737583

ACKNOWLEDGMENTS

We thank the anonymous reviewers for very useful comments, William Whyte for the discussions, and Jens Groth and Victor Shoup for clarifications on [21]. Alexandra Boldyreva was sponsored in part by Qualcomm and by the National Science Foundation under Award No.1946919. Jiahao Sun was supported by the National Science Foundation under Award No.1946919 while at Georgia Tech. Zichen Giu was supported by Zurich Information Security and Privacy Center (ZISC).

A ECDSA

Figure 18 recalls the ECDSA signature scheme.

B PROOFS

B.1 Proof of Theorem 5.1

We consider a series of "hybrid" games $G_0, \ldots G_5$ associated with adversary $\mathcal{A}$. Let $\Pr_j$ denote the probability of game G_j returning 1, for $0 \le j \le 5$.

G_0 is identical to $\mathbf{Exp}^{\text{priv-cra}}_{\mathcal{BKM}}(\mathcal{A})$.

By definition, we have

$$\Pr_0 = \frac{1}{2}\mathbf{Adv}^{\text{priv-cra}}_{\mathcal{BKM}}(\mathcal{A}) + \frac{1}{2} \ .$$

G_1 is identical to G_0 except in G_1 the encryption cocoon keys (which are computed by honest EE as part of $EEout$ and $\boldsymbol{sk}_{\mathbf{cc}}{}^A$, $\boldsymbol{sk}_{\mathbf{cc}}{}^B$ in Figure 7) are computed via random expansions a_i:

$\boldsymbol{esk}_{\mathbf{cc}}[i] \leftarrow esk_{\mathsf{cp}} + a_i$,

$\boldsymbol{epk}_{\mathbf{cc}}[i] \leftarrow epk_{\mathsf{cp}} + a_i \cdot G$.

In the ideal cipher model, $\mathcal{EXP}(ek_{\mathcal{EXP}}, \cdot)$ is modeled as a random permutation. The only difference between games is that in one game all expansion tweaks are chosen uniformly at random and in the other game they are outputs of a random permutation

$\underline{\mathcal{ECDSA}.\mathcal{K}((\mathbb{G},G,q))}$
$u \xleftarrow{\$} \mathbb{Z}_q$
$pk \leftarrow uG$
$sk \leftarrow u$
Return (pk, sk)

$\underline{\mathcal{ECDSA}.\mathcal{S}(sk,m)}$
$z \xleftarrow{\$} \mathcal{H}(m)$
$t \xleftarrow{\$} \mathbb{Z}_q$
$(e_x, e_y) \leftarrow tG$
$r \leftarrow e_x \mod q$
If $r = 0 \mod q$
  Go to step 2
$s \leftarrow t^{-1}(z + r \cdot sk)$
If $s = 0 \mod q$
  Go to step 2
Return (r, s)

$\underline{\mathcal{ECDSA}.\mathcal{V}(pk,\sigma)}$
$(r, s) \leftarrow \sigma$
If $r, s \notin \mathbb{Z}_q$
  Return 0
$w \leftarrow s^{-1} \mod q$
$z \xleftarrow{\$} \mathcal{H}(m)$
$u_1 \leftarrow zw \mod q$
$u_2 \leftarrow rw \mod q$
$(e_x, e_y) \leftarrow u_1 \cdot G + u_2 \cdot pk$
If $(e_x, e_y) = (0, 0)$
  Return 0
Return $r = e_x \mod q$

Figure 18: $\mathcal{ECDSA}$ signature scheme.

on non-repeating inputs. Therefore, the only way the attacker can distinguish the games if there is a collision between random tweaks. But this only happens with negligible probability. Hence we have

$$\Pr_1 - \Pr_0 \leq \mathsf{negl}(\lambda) .$$

G_2 is identical to G_1 except in G_2, if $\mathcal{A}$ modifies at least one ciphertext in the responses it got from the ACA as part of $\mathcal{BKG}$ stage, i.e., there is at least one ciphertext sent to the EE that the ACA did not send, and the EE does not reject, then the game aborts.

Note that each ciphertext is signed by the ACA, hence if the attacker sends a new ciphertext to the EE, and the EE does not reject, it means that the adversary succeeded in creating a forged signature. It is straightforward to construct an efficient attacker breaking the UF-CMA security of the signature scheme used by the ACA. And hence

$$\Pr_2 - \Pr_1 \leq \mathbf{Adv}^{\text{uf-cma}}_{\mathcal{DS}_2}(C) .$$

G_3 is identical to G_2 except in G_3, if $\mathcal{A}$ sends at least one cocoon public key to ACA that is computed not according to the protocol (distinct from what EE computes), and EE later does not reject, then the game aborts.

Here we consider an event when the adversary, who is given a set of cocoon public keys $pk_1, \ldots, pk_n$ can modify a public key pk_i into pk_i' for some $1 \leq i \leq n$ so that a ciphertext of some message under pk_i' will be deemed as valid wrt pk_i.

We construct $\mathcal{D}$ breaking target robustness as follows. $\mathcal{D}$ is given pk and runs A. $\mathcal{D}$ picks bit d at random and also picks $2n$ random tweaks $a_1, \ldots, a_{2n}$ from $\mathbb{Z}_q$, and picks j at random from $\{1, \ldots, 2n\}$. If $d = 0$, then it sets $pk^A_{\text{cp}} \leftarrow pk - a_j G$, and generates pk^B_{cp} honestly. It sets $sk^A_{\text{cc}\,i} \leftarrow pk + a_i G$ for $1 \leq i \leq n$ and $sk^B_{\text{cc}\,i} \leftarrow pk^B_{\text{cp}} + a_i G$ for $n+1 \leq i \leq 2n$. If $d = 1$, then tt sets $pk^B_{\text{cp}} \leftarrow pk - a_j G$, and generates pk^A_{cp} honestly. It sets $sk^B_{\text{cc}\,i} \leftarrow pk + a_i G$ for $1 \leq i \leq n$ and $sk^A_{\text{cc}\,i} \leftarrow pk^A_{\text{cp}} + a_i G$ for $n+1 \leq i \leq 2n$. $\mathcal{D}$ simulates the rest of the experiment properly.

Note that all the keys have the right distribution. With probability $1/2n$, A will modify the caterpillar key with index j that is equal to $pk - a_j G + a_j G = pk$ into some pk'. $\mathcal{D}$ will output pk' and the ciphertext C it created under pk. If the game aborts, then $\mathcal{D}$ wins its own game. We have

$$\Pr_3 - \Pr_2 \leq 2n \cdot \mathbf{Adv}^{\text{trob}}_{\mathcal{AE}}(\mathcal{D}) .$$

G_4 is identical to G_3 except in G_4 the ACA encrypts random plaintexts. Here we claim that there is an efficient adversary $\mathcal{B}$ so that

$$\Pr_4 - \Pr_3 \leq \mathbf{Adv}^{\text{ind-cpa-wrka}}_{\mathcal{AE},(\text{ePKExp}_+,\text{eSKExp}_+)}(\mathcal{B}) .$$

To justify the above, we construct an adversary $\mathcal{B}$ that attacks uf-wrka security of the encryption scheme, using $\mathcal{A}$ as a subroutine. Adversary $\mathcal{B}$ is given the global information $I = (\mathbb{G}, G, q)$. It is also given the public key $pk = x \cdot G$ and random offsets (tweaks) $t_1, \ldots, t_\ell$. The attacker also has access to oracle $\textsc{Enc}(\cdot,\cdot,\cdot)$.

$\mathcal{B}$ runs $(pk_{\text{aca}}, sk_{\text{aca}}) \xleftarrow{\$} \mathcal{ACAKG}(I)$, $(pk^A_{\text{cp}}, sk^A_{\text{cp}}) \xleftarrow{\$} \mathsf{ECKG}(I)$, $(pk^B_{\text{cp}}, sk^B_{\text{cp}}) \xleftarrow{\$} \mathsf{ECKG}(I)$; $k^A_{\mathcal{EXP}} \xleftarrow{\$} \{0,1\}^k$, $k^B_{\mathcal{EXP}} \xleftarrow{\$} \{0,1\}^k$, $epk^A_{\text{cp}} \leftarrow pk$, $y \xleftarrow{\$} \mathbb{Z}_q$, $epk^B_{\text{cp}} \leftarrow pk + yG$, and gives $(I, pk^A_{\text{cp}}, pk^B_{\text{cp}}, epk^A_{\text{cp}}, epk^B_{\text{cp}}, k^A_{\mathcal{EXP}}, k^B_{\mathcal{EXP}}, pk_{\text{aca}})$ to $\mathcal{A}$. The attacker is not given $ek^A_{\mathcal{EXP}}$ or $ek^B_{\mathcal{EXP}}$ because in this game the cocoon keys are "expanded" with random values. $\mathcal{B}$ computes $epk_{\text{cc}}{}^A_i \leftarrow pk_{\text{cp}} + t_i G$ and $epk_{\text{cc}}{}^B_i \leftarrow epk^B_{\text{cp}} + (t_{i+n} - y)G$ for $1 \leq i \leq n$.

Next $\mathcal{B}$ simulates the ACA in the $\mathcal{BKG}$ interaction. $\mathcal{B}$ computes the responses for both parties A and B according to the algorithm with the only difference in how the ciphertexts are computed. Instead of computing $c_i \leftarrow \mathsf{Enc}.\mathcal{E}(epk^A_{\text{cc}\,i}, pk_{\text{bf}\,i} \| cert_i \| r_i)$, $\mathcal{B}$ makes a query to its oracle $\textsc{Enc}(pk_{\text{bf}\,i} \| cert_i \| r_i, M, i)$ for a random M with the same length as $pk_{\text{bf}\,i} \| cert_i \| r_i$. And instead of $c_i \leftarrow \mathsf{Enc}.\mathcal{E}(epk_{\text{cc}}{}^b_i, pk_{\text{bf}\,i} \| cert_i \| r_i)$, $\mathcal{B}$ makes a query to its oracle $\textsc{Enc}(pk_{\text{bf}\,i} \| cert_i \| r_i, M, i+n)$ for a random M with the same length as $pk_{\text{bf}\,i} \| cert_i \| r_i$.

Finally, $\mathcal{B}$ flips bit $b \xleftarrow{\$} \{0,1\}$, and computes the challenge butterfly key pairs according to G_4. When $\mathcal{A}$ outputs its guess d, $\mathcal{A}$ outputs 1 iff $d = b$.

We now claim that the simulation is perfect for $\mathcal{A}$ in that its view is like in the respectful games: when the challenge bit of $\mathcal{B}$ is 1, then the view of $\mathcal{A}$ is as in G_4, and the challenge bit of $\mathcal{B}$ is 0, then the view of $\mathcal{A}$ is as in G_3. Everything is simulated properly in the obvious way except perhaps for the caterpillar and cocoon encryption public keys ciphertexts. Recall that $epk^A_{\text{cp}} = pk$ and epk^B_{cp} is computed as $pk + yG$. The latter also has the right uniform distribution since y is picked at random. Finally, the ciphertexts are all computed under the proper cocoon public keys. The ciphertext

for i's cocoon public key for EE A is computed via calling the encryption oracle with the third input index i, i.e., the ciphertext is encrypted under $pk + t_iG = epk_{\text{cp}}^A + t_iG = epk_{\text{cc}_i}^A$ for $1 \le i \le n$. And the ciphertext for i's cocoon public key for EE B is computed via calling the encryption oracle with the third input index $i+n$, i.e., the ciphertext is encrypted under $pk + t_iG =$ for $n+1 \le i \le 2n$. which is right since $epk_{\text{cc}_i}^B = epk_{\text{cp}}^B + (t_{i+n} - y)G = pk + yG + (t_{i+n} - y)G = pk + t_{i+n}G$ for $1 \le i \le n$, which is the same as $pk + t_iG$ for $n+1 \le i \le 2n$.

G_5 is identical to G_4 except in G_5, if $\mathcal{A}$ makes a random oracle query on any $r_i \| pk_{\text{cc}\,i}$, then the experiment aborts.

Note that since the view of the attacker in G_5 is independent from any r_i, unless the adversary makes a random oracle query containing it. This accounts to finding at least one pre-image of a hash in the random oracle model and the probability of that is negligible.

$$\Pr_5 - \Pr_4 \le \mathsf{negl}(\lambda) \ .$$

Since the view of the attacker is independent from the challenge bit, we claim that

$$\Pr_5 = \frac{1}{2} \ .$$

Finally, we observe that

$$\Pr_0 = \frac{1}{2}\mathbf{Adv}^{\text{priv-cra}}_{\mathcal{BKM}}(\mathcal{A}) + \frac{1}{2} = \sum_{j=0}^{4}(\Pr_j - \Pr_{j+1}) + \Pr_5 \ ,$$

and the statement of the theorem follows from the above bounds.

B.2 Proof of Theorem 5.2.

The adversary sees two caterpillar public keys and two cocoon public keys, which come either from the same EE or from two different EEs. The cocoon keys are created by modifying (additevely) the caterpillar keys using the expansion function $\mathcal{EXP}$ and the expansion key, different for each EE. Since in the ideal cipher model, each expansion function is modeled as a random permutation, then the only way the adversary could distinguish two worlds (determine the challenge bit b) is when two cocoon public keys coming from different EE happen to be same, since the keys of the same EE cannot be the same. But the probability if the former event is related to the birthday bound and is negligible. □

B.3 Proof of Theorem 5.3

We construct an adversary $\mathcal{B}$ that attacks uf-wrka security of the signature scheme $\mathcal{DS}$, using $\mathcal{A}$ as a subroutine. Let q_h be the number of random oracle queries $\mathcal{A}$ makes. Adversary $\mathcal{B}$ is given the global information $I = (\mathbb{G}, G, q)$, where $\mathbb{G}$ is the group of points on the elliptic curve of prime order q, generated by $G \in \mathbb{G}$. It is also given the public key $pk = x \cdot G$. $\mathcal{B}$ outputs random offsets (tweaks) $a_1, \ldots, a_n$ and computes $z_i \leftarrow H(a_i)$ for $1 \le i \le n$. The attacker has access to oracle $\textsc{Sign}(\cdot,\cdot)$.

$\mathcal{B}$ runs $(pk_{\text{aca}}, sk_{\text{aca}}) \xleftarrow{\$} \mathcal{ACAKG}(I)$, and gives $(I, pk, sk_{\text{aca}}, pk_{\text{aca}})$ to $\mathcal{A}$. $\mathcal{A}$ is not given $k_{\mathcal{EXP}}$ or $ek_{\mathcal{EXP}}$ because the permutation underlying the expansion keys is modeled as the ideal cipher. This means that for each key, the resulting function is modeled as a random permutation the adversary is given access to. Accordingly, $\mathcal{B}$ simulates computations $\mathcal{EXP}(k_{\mathcal{EXP}}, \cdot)$ and $\mathcal{EXP}(ek_{\mathcal{EXP}}, \cdot)$ for the interaction with the honest EE in $\mathcal{CKE}$ via oracles O_1, O_2 as follows. $\mathcal{B}$ picks two random permutations π_1, π_2 and random offsets u_i, v_i, where $u_i \ne u_j$ and $v_i \ne v_j$ for $1 \le i, j \le n$. It then then computes $pk_{\text{cc}\,i} \leftarrow pk_{\text{cp}} + u_iG$ and $epk_{\text{cc}\,i} \leftarrow epk_{\text{cp}} + v_iG$ and when $\mathcal{A}$ queries oracle O_1 with i, $\mathcal{B}$ returns u_i; when $\mathcal{A}$ queries oracle O_2 with i, $\mathcal{B}$ returns v_i.

Next adversary $\mathcal{B}$ simulates EE in $\mathcal{BKG}$ interaction for $\mathcal{A}$. Here for simplicity we assume that $\mathcal{A}$ makes only one random oracle query containing each $pk_{\text{cc}\,i}$. (There is no reason for the adversary to query more for each butterfly key computation. But if it does, then $\mathcal{B}$ would have to "program" a random one with a_i, and the resulting bound would have a factor of the maximum number of random oracle queries per key.) To answer $\mathcal{A}$'s random oracle queries, $\mathcal{B}$ does the following. If the random oracle query by $\mathcal{A}$ does not contain any $pk_{\text{cc}\,i}$, then $\mathcal{B}$ uses its own random oracle to provide the answer. If $\mathcal{A}$ makes a random oracle query $r \| pk_{\text{cc}\,i}$, $\mathcal{B}$ returns $h_i \leftarrow z_i - u_i$.

Next adversary $\mathcal{B}$ simulates EE in $\mathcal{BKR}$. When $\mathcal{A}$ queries oracle $\textsc{Sign}$, $\mathcal{B}$ fowards the query to its own oracle $\textsc{Sign}^{\mathcal{DS}}$, and forwards the oracle's response back to $\mathcal{A}$. When $\mathcal{A}$ provides the responses at the end of $\mathcal{BKG}$ stage, $\mathcal{B}$ verifies them using pk_{aca}. In the end, adversary $\mathcal{B}$ obtains the output of $\mathcal{A}$ and uses that forgery as its own output.

We now claim that $\mathcal{B}$ wins whenever $\mathcal{A}$ wins. First, we observe that the simulation is perfect for $\mathcal{A}$ in that its view is like in the actual experiment. The EE's key it is given has the right distribution of a random key (because the same key given to $\mathcal{B}$ has the same distribution). The expansion values u_i, v_i are computed correctly under the ideal cipher model. Under the random oracle model, the BKE's offsets z's also have the right (uniform) distribution.

Since u_i is selected at random by adversary $\mathcal{B}$, and z_i are the outputs of the random oracle, the simulated outputs of $\mathcal{H}$, $h_i = z_i - u_i$ also appear random to adversary $\mathcal{A}$, with the same uniform distribution as from a true random oracle.

At the end of the simulation, if adversary $\mathcal{A}$ successfully forged a signature σ for message m under secret key $sk_{\text{bf}\,i} = sk_{\text{cc}\,i} + h_i = (sk + u_i) + (z_i - u_i) = sk + z_i$, then (σ, m) can also be used as a valid forgery for adversary $\mathcal{B}$ in the uf-wrka game.

And clearly, if $\mathcal{A}$ is efficient, then $\mathcal{B}$ is efficient.

□

B.4 Proof of Theorem 6.1

Consider the following sequence of $n+1$ "hybrid" games associated with an adversary $\mathcal{A}$. Let $\Pr_j$ denote the probability of game G_j returning 1, for $0 \le j \le n$.

Note that by definition G_0 is ind-cpa-wrka game for $\mathcal{ECIES}$ and $(\mathsf{ePKExp}_+, \mathsf{eSKExp}_+)$. And then we have that

$$\begin{aligned} \Pr_0 &= \Pr\left[\ \text{ind-cpa-wrka}^{\mathcal{A}} \Rightarrow 1\ \right] \\ &= \frac{\mathbf{Adv}^{\text{ind-cpa-wrka}}_{\mathcal{ECIES},(\mathsf{ePKExp}_+,\mathsf{eSKExp}_+)}(\mathcal{A})}{2} + \frac{1}{2} \ . \end{aligned}$$

Now note that

$$\Pr_0 = \sum_{j=0}^{n-1}(\Pr_j - \Pr_{j+1}) + \Pr_n \ .$$

Game G_j

$(uG, u) \xleftarrow{\$} \mathcal{ECIES}.\mathcal{K}((\mathbb{G}, G, q))$
$(a_1, ..., a_n) \xleftarrow{\$} \mathcal{A}(uG)$
For $i = 1, ..., n$:
  $sk_i \leftarrow u + a_i$
  $pk_i \leftarrow (u + a_i)G$
$b \xleftarrow{\$} \{0, 1\}$
$b' \xleftarrow{\$} \mathcal{A}^{\text{Enc}(\cdot,\cdot,\cdot)}(a_1, ..., a_n)$
Return $(b' = b)$

Enc(m_0, m_1, i)

If $|m_0| \neq |m_1|$ then Return $\perp$:
If $i \leq j$:
  $v \xleftarrow{\$} \mathbb{Z}_q$
  $V \leftarrow vG$
  $(k_{\mathcal{SE}}, k_{\mathcal{MAC}}) \xleftarrow{\$} \{0, 1\}^{hlen}$
  $e \xleftarrow{\$} \mathcal{SE}.\mathcal{E}(k_{\mathcal{SE}}, m_b)$
  $t \leftarrow \mathcal{MAC}(k_{\mathcal{MAC}}, e)$
  Return $V \| e \| t$
Else:
  $c \leftarrow \mathcal{ECIES}.\mathcal{E}(pk_i, m_b)$
Return c

Figure 19: "Hybrid" games G_j for $0 \leq j \leq n$ for the proof of Theorem 6.1.

The statement of the theorem follows from the above equations and the following claims.

Lemma B.1. *For any $0 \leq j \leq n-1$ and any efficient adversary $\mathcal{A}$ there exists an efficient adversary $\mathcal{D}$ such that*

$$(Pr_j - Pr_{j+1}) \leq \mathbf{Adv}^{\text{hdh}}_{\mathcal{HK},\mathcal{G}}(\mathcal{D}) .$$

Proof. Recall that the hdh attacker $\mathcal{D}$ is given the public group info $(\mathbb{G}, G, q)$ and also $U = uG, V = vG, Z$, where u, v are random elements in $\mathbb{Z}_q$ and Z is either $H(uvG)$ or a random bitstring. To figure out which case it is $\mathcal{B}$ uses $\mathcal{A}$.

$\mathcal{D}$ picks a random bit b, offsets $a_1, \ldots, a_n$ from $\mathbb{Z}_q$ at random and runs $\mathcal{A}$ on $pk = U - a_j G$ and $a_1, \ldots, a_n$. $\mathcal{D}$ computes $pk_i \leftarrow (a_i - a_j)G + U$ for all $0 \leq i \leq n$.

To answer $\mathcal{A}$'s oracle query Enc(m_0, m_1, i), $\mathcal{D}$ follows the algorithm outlined on the right side in Figure 19 for all $i \neq j$. When $i = j$, then $\mathcal{D}$ uses Z in place of the output of H, and the rest is unchanged.

Finally, $\mathcal{D}$ outputs 0 if $\mathcal{A}$ guesses b correctly.

To analyze $\mathcal{D}$, note that the public keys of $\mathcal{A}$ are $(a_1 - a_j)G + U, \ldots, U, \ldots, (a_n - a_j)G + U$ have the right distribution. If $\mathcal{D}$'s challenge bit b is 0, i.e., Z is random, then the view of $\mathcal{A}$ in the simulated experiment is exactly as in G_j. And if $b = 1$, i.e., $Z = H(uvG) = H(v(u + a_j - a_j))G)$, then the view of $\mathcal{A}$ in the simulated experiment is exactly as in G_{j+1}.

Now,

$$\begin{aligned}\mathbf{Adv}^{\text{hdh}}_{\mathcal{HK},\mathcal{G}}(\mathcal{D}) &= \Pr\left[Exp\ \text{hdh}^0 \Rightarrow 0\right] - \Pr\left[Exp\ \text{hdh}^1 \Rightarrow 0\right] \\ &= \Pr_j - \Pr_{j+1} .\end{aligned}$$

Clearly, if $\mathcal{A}$ is efficient, then $\mathcal{D}$ is efficient. □

Lemma B.2. *For any any efficient adversary $\mathcal{A}$ there exists an efficient adversary $\mathcal{B}$ such that*

$$Pr_n \leq \frac{n \cdot \mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(\mathcal{B})}{2} + \frac{1}{2} .$$

Proof. In G_n, messages are encrypted using the symmetric encryption scheme under the keys, which are random and independent from other information the adversary sees. Hence we can construct an adversary that breaks security of the symmetric encryption scheme. Since several random keys are involved, for simplicity we use the ind-cpa security definition in the multi-user setting, n-ind-cpa, [9], and their result about the relation to the security in the standard (single-user) setting.

The n-ind-cpa attacker $\mathcal{B}_n$ is given n left-right encryption oracles associated with n randomly generated keys, that it can query on pairs of messages m_0, m_1 of equal length and they return encryptions, under the corresponding key, of m_b, where b is the challenge bit flipped at random at the beginning of the experiment. $\mathcal{B}_n$ has to guess b.

$\mathcal{B}_n$ simulates G_n for $\mathcal{A}$, by following the code of G_n with the only difference in answering the oracle query of $\mathcal{A}$ on (m_0, m_1, i), $k_{\mathcal{SE}}$ is not used, and instead of computing $e \xleftarrow{\$} \mathcal{SE}.\mathcal{E}(k_{\mathcal{SE}}, m_b)$, $\mathcal{B}_n$ queries its own ith left-right encryption oracle and the result is assigned to e.

When $\mathcal{A}$ returns a bit, $\mathcal{B}_n$ returns the same bit.

We claim that $\mathcal{B}_n$ wins whenever $\mathcal{A}$ wins, because $\mathcal{B}_n$ simulates G_n for $\mathcal{A}$ perfectly. In G_n encryptions are computed using randomly picked keys in the experiment, while B_n uses its own left-right encryption oracles, which are also under the randomly generated keys. And the rest is the same. Clearly, if $\mathcal{A}$ is efficient, then B_n is efficient.

To obtain the statement in the claim it remains to recall the result from [9] stating that security in the standard single-user setting implies security in the multi-user setting, with the multiplicative factor n in the security loss in the concrete relation between the adversaries. □

□

B.5 Proof of Theorem 7.2.

Consider a series of games $G_0, ..., G_n$ associated with adversary $\mathcal{A}$. Let $\Pr_j$ denote the probability of game G_j returning 1, for $0 \leq j \leq n$.

We first observe that when $j = 0$, then all ciphertexts are computed properly, according to the the $\mathcal{MR}$-$\mathcal{ECIES}$ scheme, matching mr-cpa-wrka experiment. When $j = n$, then n challenge ciphertexts are computed using the random keys for symmetric encryption. Hence,

$$\frac{1}{2}\mathbf{Adv}^{\text{mr-cpa-wrka}}_{\mathcal{MR}\text{-}\mathcal{ECIES}}(\mathcal{A}) + \frac{1}{2} = \Pr_0 = \sum_{j=0}^{n-1}(\Pr_j - \Pr_{j+1}) + \Pr_n .$$

The statement of the theorem will follow from the following two claims.

Game G_j

$b \xleftarrow{\$} \{0,1\}$
$(pk, sk) \xleftarrow{\$} \mathcal{AE}.\mathcal{K}(I)$
$(a_1, ..., a_n) \xleftarrow{\$}$ Coins
For $i = 1, ..., n$
$sk_i \leftarrow \mathsf{eSKExp}(sk; a_i)$
$pk_i \leftarrow \mathsf{ePKExp}(pk; a_i)$
$(pk', sk') \xleftarrow{\$} \mathcal{AE}.\mathcal{K}(I)$
$(b_1, ..., b_n) \xleftarrow{\$}$ Coins
For $i = 1, ..., n$
$sk'_i \leftarrow \mathsf{eSKExp}(sk; b_i)$
$pk'_i \leftarrow \mathsf{ePKExp}(pk; b_i)$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}) \xleftarrow{\$} \mathcal{A}(pk, pk', a_1, ..., a_n, b_1, ..., b_n, sk')$
If $|\mathbf{m}_0| \neq |\mathbf{m}_1|$ then return $\perp$
If $n \neq |\mathbf{m}_0|$ then return $\perp$
If $n \neq |\mathbf{m}|$ then return $\perp$
  $v \xleftarrow{\$} \mathbb{Z}_q$
  $V \leftarrow vG$
For $i = 1, ..., n$
If $i \leq j$:
  $(k_{\mathcal{SE}}, k_{\mathcal{MAC}}) \xleftarrow{\$} \{0,1\}^{hlen}$
Else:
  $(k_{\mathcal{SE}}, k_{\mathcal{MAC}}) \leftarrow H(sk_i \cdot V)$
  $e \xleftarrow{\$} \mathcal{SE}.\mathcal{E}(k_{\mathcal{SE}}, \mathbf{m}_b[i])$
  $t \leftarrow \mathcal{MAC}(k_{\mathcal{MAC}}, e)$
  $\mathbf{c}[i] \leftarrow (V \| e \| t)$
  $(k_{\mathcal{SE}}, k_{\mathcal{MAC}}) \leftarrow H(sk'_i \cdot V)$
  $e \xleftarrow{\$} \mathcal{SE}.\mathcal{E}(k_{\mathcal{SE}}, \mathbf{m}[i])$
  $t \leftarrow \mathcal{MAC}(k_{\mathcal{MAC}}, e)$
  $\mathbf{c}[i+n] \leftarrow (V \| e \| t)$
$b' \xleftarrow{\$} \mathcal{A}(\mathbf{c})$
Return $(b' = b)$

Figure 20: "Hybrid" games for the proof of Theorem 7.2.

Lemma B.3. *For any efficient $\mathcal{A}$ and $0 \leq j \leq n-1$ there is an efficient $\mathcal{B}$ such that*

$$(Pr_j - Pr_{j+1}) \leq \mathbf{Adv}^{\mathsf{odh}}_{\mathcal{G},\mathcal{HK}}(\mathcal{B}) .$$

Proof. We present adversary $\mathcal{B}$ in Figure 21. The "neighboring" hybrid games G_j and G_{j+1} differ only in whether the keys are picked at random and computed using the hash, and this the reduction to ODH here insures that these games are indistinguishable to the adversary. □

Lemma B.4. *For any efficient $\mathcal{A}$ there is an efficient C such that*

$$Pr_n \leq n \cdot \frac{1}{2}\mathbf{Adv}^{\mathsf{ind\text{-}cpa}}_{\mathcal{SE}}(C) + \frac{1}{2} .$$

Proof. Since all challenge messages are encrypted using the symmetric encryption scheme with randomly chosen keys, then it is straightforward to show a reduction to IND-CPA security of the scheme. The factor of n is from the reduction from the ind-cpa security in the multi-user setting to the single-user setting, similarly to the proof in [9]. □

Adversary $\mathcal{B}^{O_v(\cdot)}(I, V, U, Z)$

$b \xleftarrow{\$} \{0,1\}$
$(a_1, ..., a_n) \xleftarrow{\$}$ Coins
$(b_1, ..., b_n) \xleftarrow{\$}$ Coins
$pk \leftarrow U - a_j$
$(pk', sk') \xleftarrow{\$} \mathcal{AE}.\mathcal{K}(I)$
For $i = 1, ..., n$
$pk_i \leftarrow \mathsf{ePKExp}(pk; a_i)$
$sk'_i \leftarrow \mathsf{eSKExp}(sk'; b_i)$
$pk'_i \leftarrow \mathsf{ePKExp}(pk'; b_i)$
$(\mathbf{m}_0, \mathbf{m}_1, \mathbf{m}) \xleftarrow{\$} \mathcal{A}(pk, pk', a_1, ..., a_n, b_1, ..., b_n, sk')$
If $|\mathbf{m}_0| \neq |\mathbf{m}_1|$ then return $\perp$
If $|\mathbf{m}_0| = |\mathbf{m}_1| = |\mathbf{m}'|$ is not n then return $\perp$
For $i = 1 \ldots, n$
  If $i = j$ then $(k_{\mathcal{SE}}, k_{\mathcal{MAC}}) \leftarrow Z$
  If $i < j$ then $(k_{\mathcal{SE}}, k_{\mathcal{MAC}}) \leftarrow O_v(pk + a_i G)$
  If $i > j$ then $(k_{\mathcal{SE}}, k_{\mathcal{MAC}}) \xleftarrow{\$} \{0,1\}^{hlen}$
  $e_i \xleftarrow{\$} \mathcal{SE}.\mathcal{E}(k_{\mathcal{SE}}, \mathbf{m}_b[i])$
  $t_i \leftarrow \mathcal{MAC}(k_{\mathcal{MAC}}, e_i)$
  $\mathbf{c}[i] \leftarrow V \| e \| t$
For $i = 1 \ldots, n$
  $(k_{\mathcal{SE}}, k_{\mathcal{MAC}}) \leftarrow H((sk' + b_i)V)$
  $e'_i \xleftarrow{\$} \mathcal{SE}.\mathcal{E}(k_{\mathcal{SE}}, \mathbf{m}[i])$
  $t'_i \leftarrow \mathcal{MAC}(k_{\mathcal{MAC}}, e_i)$
  $\mathbf{c}'[i] \leftarrow V \| e \| t$
$b' \xleftarrow{\$} \mathcal{A}(\mathbf{c}, \mathbf{c}')$
Return $(b' = b)$

Figure 21: Adversary $\mathcal{B}$ for the proof of Theorem 7.2.