A preliminary version of this paper appears in the proceedings of the *42nd IEEE Symposium on Security and Privacy (S&P 2021)*. This is the full version.

The Provable Security of Ed25519: Theory and Practice

Jacqueline Brendel[1], Cas Cremers[1], Dennis Jackson[2], and Mang Zhao[1]

[1]CISPA Helmholtz Center for Information Security
[2]Department of Computer Science, ETH Zurich

Version 1.0.3*– October 14, 2020

Abstract

A standard requirement for a signature scheme is that it is existentially unforgeable under chosen message attacks (EUF-CMA), alongside other properties of interest such as strong unforgeability (SUF-CMA), and resilience against key substitution attacks.

Remarkably, no detailed proofs have ever been given for these security properties for EdDSA, and in particular its Ed25519 instantiations. Ed25519 is one of the most efficient and widely used signature schemes, and different instantiations of Ed25519 are used in protocols such as TLS 1.3, SSH, Tor, ZCash, and WhatsApp/Signal. The differences between these instantiations are subtle, and only supported by informal arguments, with many works assuming results can be directly transferred from Schnorr signatures. Similarly, several proofs of protocol security simply assume that Ed25519 satisfies properties such as EUF-CMA or SUF-CMA.

In this work we provide the first detailed analysis and security proofs of Ed25519 signature schemes. While the design of the schemes follows the well-established Fiat-Shamir paradigm, which should guarantee existential unforgeability, there are many side cases and encoding details that complicate the proofs, and all other security properties needed to be proven independently.

Our work provides scientific rationale for choosing among several Ed25519 variants and understanding their properties, fills a much needed proof gap in modern protocol proofs that use these signatures, and supports further standardisation efforts.

1 Introduction

The Ed25519 signature scheme was introduced in 2011 by Bernstein, Duif, Lange, Schwabe, and Yang in the paper "High-speed high-security signatures" [1]. The efficiency of the scheme has led to a global uptake in modern applications, and it is now used in TLS 1.3, SSH, Tor, ZCash, and messaging protocols based on the Signal protocol such as WhatsApp.

For modern digital signature schemes the quintessential property is *existential unforgeability under chosen message attacks* (EUF-CMA) [2]. This property basically requires that an adversary cannot construct a signature for a message that the key owner did not sign previously.

A stronger property that can be met by signature schemes, is that of *strong unforgeability under chosen message attacks* (SUF-CMA). This property additionally requires that the adversary cannot construct an alternative signature for a given signed message. A high-profile example exploiting the absence of this property is the Mt. Gox attack on Bitcoin [3]. In this attack, signatures on transactions were mauled by an adversarial recipient before being stored on the blockchain. The recipient would then claim that the transaction failed. The sender would check the blockchain, and would indeed not find their exact signature (due to mauling), conclude that it apparently failed, and start a new transfer. Yet both signatures would be valid and the recipient would thus receive the double amount. This attack would not have been possible with a strongly unforgeable (SUF-CMA) signature scheme as this notion prohibits malleability.

Additionally, in many practical systems, it is highly desirable that signature schemes resist key substitution attacks [4], [5]. In such attacks the adversary computes specific public keys, e.g., based on observed signatures of honest signers, such that these honest signatures can also be verified under the

*We provide a summary of changes in Appendix C.

adversary's new public keys. This has been shown to lead to attacks on protocols such as Let's Encrypt Certificate Issuance and SOAP's WS-Security [4].

Surprisingly, *full proofs of any of these security properties have never been given for Ed25519*. The original publications [1], [6] focused on efficiency of computation, and do not contain a precise statement on the security property that is offered by the scheme, which in the following we will refer to as Ed25519-Original. Because the scheme is said to be constructed via the Fiat-Shamir transform, it should follow that Ed25519-Original at least provides EUF-CMA security, but full details were never provided. The papers do refer to malleability and argue that is not relevant for the standard definitions of signature scheme security, but their definition of malleability does not agree with the common usage of the term. It also transpires that, whilst the source code presented alongside the paper accepts mangled signatures (hence is not SUF-CMA), the additional check included in the paper's description *but not the source code*, is actually sufficient to prove SUF-CMA, as we will show. This further adds to the confusion around the security properties enjoyed by Ed25519.

While the original papers came with a full implementation of Ed25519-Original, later implementations made various modifications. Notably, the Ed25519-IETF version that was standardized by the Internet Engineering Task Force (IETF) in [7] includes a check that is claimed to prevent malleability, thereby implicitly suggesting that Ed25519-IETF is strongly unforgeable (SUF-CMA). Later versions, such as the ones used by LibSodium [8] and ZCash [9] included additional group element checks. We return to the details of these differences in Section 4.1. This leads to the obvious question: which exact properties are actually provided by the various Ed25519 schemes? This question is especially timely as Ed25519 is currently proposed for inclusion in the USA's National Institute of Standards and Technology (NIST) standard for Digital Signature Schemes [10]–[12] and was recently included in the TLS 1.3 standard [13].

Over the last years, several published works reported security proofs of systems that use Ed25519, but require specific cryptographic security notions from their signature schemes, such as computational proofs of TLS 1.3's properties [14]–[17]. These proofs assume that Ed25519-IETF satisfies EUF-CMA, leaving a proof gap. A claimed computational proof of SSH [18] requires that all supported signature schemes provide SUF-CMA. However, SSH implementations also allow the use of the malleable Ed25519-Original, which leads to a counterexample to the security statement. The Signal protocol library implements yet another variant of the Ed25519 signature scheme. Adding to the confusion, a recent work [19] claims that the results on Schnorr signatures in prime order groups implies Ed25519-Original enjoys SUF-CMA and resistance to key substitution attacks. We will see in Section 5.4 that this is not the case.

In this work we remedy these proof gaps, and establish further properties of Ed25519 signature schemes. As we will see, the devil is in the detail: the specific details of Ed25519 (e.g., small subgroup elements, scalar clamping) require subtle tailoring of the proof details, and lead to mismatches such as the lack of SUF-CMA security for Ed25519-Original despite its similarities to Fiat-Shamir applied to the Schnorr identification scheme, which 'should' imply that SUF-CMA security holds. These subtleties also manifest themselves in the requirements of various checks. Thus, our work not only fills these highly-needed proof gaps, but also provides additional insight into the subtle differences between Ed25519 schemes which are summarized in Table 2.

Our main contributions are the following.

- We provide the first detailed proof that Ed25519-Original [1] is indeed EUF-CMA secure.
- We provide the first proof that Ed25519-IETF [7] is actually SUF-CMA secure.
- We prove that all Ed25519 schemes are resilient against key substitution attacks, and that if small subgroup keys are rejected as in LibSodium, a signature uniquely identifies a message, even for malicious keys.
- In a wider sense, our results retroactively support the standardisation of Ed25519-IETF, and support the ongoing standardisation by NIST.

Overview: In Section 2 we present related work, and Section 3 recalls background knowledge. Section 4 presents Ed25519 signature schemes and their subtle differences from Schnorr identification schemes. We provide the security proofs in Section 5 and we conclude in Section 6.

2 Related Work

2.1 History of EdDSA and Ed25519

Ed25519-Original is just one instantiation of the more general EdDSA signature scheme, which was introduced in the same paper [1], [6]. EdDSA is itself a variant of the well-known Schnorr signature scheme [20], [21]. Ed25519 is EdDSA instantiated over curve Edwards25519 [1] and remains by far the most popular instantiation of EdDSA, despite its later extension to support alternative curves [7], [22].

EdDSA instantiations such as Ed25519-Original can sign and verify signatures substantially faster than almost all other signatures schemes at similar security levels. For schemes that have comparable speeds, Ed25519-Original further provides considerably smaller signatures, producing 64-byte signatures and 32-byte public keys. Additionally, EdDSA is widely considered to provide better resistance to side-channel attacks than alternative schemes. However, the original papers [1], [6] contain no formal statements (and consequently, no actual proofs) of its security properties.

By virtue of its outstanding performance with respect to efficiency and bandwidth, EdDSA was standardised by the IETF between 2015 and 2017 [7]. In 2019, EdDSA was proposed to also be adopted as part of NIST's Digital Signature Standard (DSS) [10], [11]. In early 2020, the public call for comments was closed [12], but as of writing, no new version has appeared.

2.2 Related proofs

The Fiat-Shamir paradigm was proposed by Fiat and Shamir [23] as a generic approach to derive a secure signature scheme from a canonical identification schemes (CID). A vast body of work followed this seminal result and the aforementioned Schnorr signatures [20], [21], on which EdDSA was built, is probably one of the most famous examples of the transform's power to build efficient and provably-secure signatures. Here we merely present some of the many milestones related to Fiat-Shamir that are most relevant for our work. While the original presentation [23] lacked security proofs, Pointcheval and Stern [24], [25] closed this gap by providing proofs in the (then) relatively new random oracle model [26]. Abdalla et al. [27] indicated the minimal conditions for the underlying identification scheme to prove Fiat-Shamir transformed signatures to be EUF-CMA secure. In 2016, Kiltz et al. [28], [29] provided a concrete and modular security analysis of Fiat-Shamir signatures in both the single-user and multi-user setting, closing the tightness gap of the reduction.

The treatment of the multi-user setting is especially interesting as in practical applications there exist many different public keys for an adversary to attack. In 2002, Galbraith, Malone-Lee, and Smart [30] considered security of signatures in this multi-user setting. They showed that if an adversary were to attack N keys at once, its advantage can increase only at most by a factor of N (this is often referred to as the *generic bound*). Their second result claimed that for *Schnorr-like* signatures one can do even better and achieve a tight reduction between single-user and multi-user security. Much later, Bernstein [31] exposed a flaw in this tight proof that to this date could not be resolved. However, Bernstein [31] was able to show that one can achieve a tight reduction between single-user security of Schnorr signatures and multi-user security of *key-prefixed* Schnorr signatures. Key prefixing had been introduced earlier by Menezes and Smart [32] in the context of key-substitution attacks, where they also (controversially) claimed that it is not necessary to actively mitigate key-substitution attacks for Schnorr signatures. In contrast, the designs of Ed25519 signatures [1], [7] nevertheless employ key-prefixing. Kiltz et al. [28] show that if the underlying canonical identification scheme achieves random self-reducibility in the random oracle model, then a tight reduction between multi-user and single-user security can be achieved without key prefixing. In Appendix 5.3, we briefly discuss multi-user security in light of these results.

2.3 Computational proofs of systems that use Ed25519

Because of its performance and conjectured security, EdDSA's Ed25519 instantiation over Edwards25519 has become one of the most popular digital signature schemes, appearing in innumerable applications and protocols including TLS 1.3 [13], SSH [33], Tor, ZCash, and the Signal protocol [34].

Regarding such systems, there exists numerous security proofs which hold only when the deployed digital signatures satisfy certain conditions. For example, Bhargavan et al. [14] developed the first machine-checked cryptographic proof for TLS 1.3 draft-18 using the verification tool CryptoVerif, thereby assuming that Ed25519-IETF meets EUF-CMA. Similarly, [15] proved the security of session resumption in the TLS 1.3 draft-05 full handshakes and [16] proved the security of (a slightly modified version of) the ephemeral Diffie-Hellman handshake of TLS 1.3 with unilateral authentication. Kobeissi, Bhargavan,

and Blanchet [35] analyzed a model of the Signal protocol in CryptoVerif assuming EUF-CMA security of Signal's X-Ed25519 scheme. However, none of these schemes have actually been proven to achieve EUF-CMA security.

In 2014, SSH was proven to be secure even when the same signing key is used across multiple ciphersuites, assuming that the underlying signature is strongly unforgeable [18][1]. However, SSH implementations may use the originally-proposed version Ed25519-Original (e.g., [37]), which does *not* satisfy SUF-CMA. This yields a counterexample in their security model: mauling a signature in an otherwise honest session allows a session-key reveal on the peer, as the sessions no longer match. Thus, their proof does not apply as-is to SSH implementations that use Ed25519-Original. [19] claims that the results on Schnorr signatures in prime order groups imply that Ed25519 enjoys SUF-CMA and resistance to key substitution attacks, which, as we will see in Section 5.4 is not the case.

3 Preliminaries

In this section we introduce the necessary notation and definitions. In particular, we also review the Fiat-Shamir transform [23] which allows to transform passively-secure (interactive) identification protocols into (non-interactive) signature schemes which are secure against active adversaries.

Notation

For an integer q, we denote by $\mathbb{F}_q$ the finite field with order q. For a bit string h and an integer i, we let $h[i]$ denote the i-th bit of h. Overloading notation, we write $\underline{a}$ for the bitstring encoding of a, where a can be an integer or a curve point. We describe the details of these encodings at the start of Section 4. The algorithms in this paper run in probabilistic polynomial time (PPT), unless stated otherwise, and we write $y \xleftarrow{\$} \mathcal{A}(x)$ for the probabilistic execution of algorithm $\mathcal{A}$ on input x with output y. If $\mathcal{A}$ is a deterministic algorithm we write $y \leftarrow \mathcal{A}(x)$. Furthermore, $\mathcal{A}^{\mathcal{O}}(x)$ denotes that $\mathcal{A}$ has access to the oracle $\mathcal{O}$ during its execution on input x. By pp we denote the public parameters of the scheme and system. We assume that they are always known to the adversary $\mathcal{A}$ and thus omit pp from its explicit input. For variables x, y, we denote by $y \leftarrow x$, the assignment of value x to y and by $s \xleftarrow{\$} \mathcal{D}$ we denote the sampling of an element x from the probability distribution $\mathcal{D}$; for simplicity, we denote the uniform sampling of an element x from a set X by $x \xleftarrow{\$} X$. Security games $\mathrm{G}^{\mathsf{sec\text{-}prop}}_{\Pi,\mathcal{A}}(\mathsf{pp})$ describe the run of an adversary $\mathcal{A}$ against the security property $\mathsf{sec\text{-}prop}$ of a cryptographic scheme Π parametrized by pp. We write $\mathrm{G}^{\mathsf{sec\text{-}prop}}_{\Pi,\mathcal{A}}(\mathsf{pp}) = 1$ to mark the event in which $\mathcal{A}$ has *won* the game. We use 1 to represent the Boolean value *True*, and 0 for *False*. Lastly, $[\![statement]\!]$, we denote the Boolean evaluation of *statement*.

3.1 Cryptographic Building Blocks

Our presentation of definitions follows the style of the textbook by Katz and Lindell [38] with security defined in the *concrete setting* which explicitly specifies the amount of time and resources needed (cf. [38, Sec. 3.1]). We first introduce the notion of secure signatures schemes before defining canonical identification protocols and their security.

Definition 1 (Signature scheme)**.** *A* signature scheme $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ *is a triple of algorithms, where* KGen *is called the key generation algorithm,* Sign *is called the signing algorithm, and* Vfy *is called the verification algorithm.*

Key generation KGen *takes as input the public parameters* pp *and outputs a public key* pk *and a secret key* sk*, i.e.,* $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$.

Signing Sign *takes as input a secret key* sk *as well as the message* $m \in \mathcal{M}$ *to be signed and outputs a signature* σ*, i.e.,* $\sigma \xleftarrow{\$} \mathsf{Sign}(sk, m)$*. Here,* $\mathcal{M}$ *is called the* message space.

Verification Vfy *takes as input a public key* pk*, a signature* σ *and message* $m \in \mathcal{M}$*. It deterministically outputs 1, i.e.,* $1 \leftarrow \mathsf{Vfy}(pk, \sigma, m)$ *if* σ *is a valid signature over* m *wrt.* pk*, else it outputs 0.*

We call a signature scheme $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ correct *if for all* $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$ *and all messages* $m \in \mathcal{M}$*:* $1 \leftarrow \mathsf{Vfy}(pk, \mathsf{Sign}(sk, m), m)$.

[1] The full version of the paper [36] explicitly uses the definition for strong unforgeability, even though both versions use a "euf-cma" shorthand.

The standard notion for security of signature schemes is that of (single-user) *existential unforgeability under chosen message attacks*. Intuitively, this guarantees that for a fixed public key, an adversary $\mathcal{A}$ cannot generate a valid signature on a new message, for which it has not seen a valid signature before. A stronger definition of security is that of (single-user) *strong unforgeability*, which will also play a role later in the paper. Here, the adversary is not restricted to forging signatures on new messages for a fixed public key but may also generate a signature on a message on which it has seen (other) signatures. Both of these notions can then be transferred to the multi-user setting, where there is not just a single public key generated by the challenger but multiple honestly generated keys. The adversary's goal is then to (existentially or strongly) forge a signature under *any* of these keys.

Definition 2 (EUF-CMA and SUF-CMA security)**.** *Let* $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ *be a signature scheme. Consider the security games* $\mathrm{G}^{\mathsf{euf\text{-}cma}}_{\mathcal{S},\mathcal{A}}$ *and* $\mathrm{G}^{\mathsf{suf\text{-}cma}}_{\mathcal{S},\mathcal{A}}$ *as defined in Fig. 1. We say that a signature scheme* $\mathcal{S}$ *is* (t, ϵ, Q_S)-EUF-CMA*-secure or* existentially unforgeable under chosen message attacks, *if for any adversary* $\mathcal{A}$ *running in time at most* t*, making at most* Q_S *queries to the signing oracle, the probability* $\Pr\left[\mathrm{G}^{\mathsf{euf\text{-}cma}}_{\mathcal{S},\mathcal{A}}(\mathsf{pp}) = 1\right] \leq \epsilon$.

Analogously, we say that $\mathcal{S}$ *is* (t, ϵ, Q_S)-SUF-CMA*-secure or* strongly unforgeable under chosen message attacks.

$\mathrm{G}^{\mathsf{euf\text{-}cma}}_{\mathcal{S},\mathcal{A}}(\mathsf{pp})$:
1 $\mathcal{L}_{\mathsf{Sign}} \leftarrow \emptyset$
2 $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$
3 $(m^\star, \sigma^\star) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}}(pk)$
4 **return** $[\![\mathsf{Vfy}(pk, \sigma^\star, m^\star) \wedge m^\star \notin \mathcal{L}_{\mathsf{Sign}}]\!]$

$\mathcal{O}_{\mathsf{Sign}}(m)$:
5 $\sigma \xleftarrow{\$} \mathsf{Sign}(sk, m)$
6 $\mathcal{L}_{\mathsf{Sign}} \leftarrow \mathcal{L}_{\mathsf{Sign}} \cup \{m\}$
7 **return** σ

$\mathrm{G}^{\mathsf{suf\text{-}cma}}_{\mathcal{S},\mathcal{A}}(\mathsf{pp})$:
1 $\mathcal{L}_{\mathsf{Sign}} \leftarrow \emptyset$
2 $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$
3 $(m^\star, \sigma^\star) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}}(pk)$
4 **return** $[\![\mathsf{Vfy}(pk, \sigma^\star, m^\star) \wedge (m^\star, \sigma^\star) \notin \mathcal{L}_{\mathsf{Sign}}]\!]$

$\mathcal{O}_{\mathsf{Sign}}(m)$:
5 $\sigma \xleftarrow{\$} \mathsf{Sign}(sk, m)$
6 $\mathcal{L}_{\mathsf{Sign}} \leftarrow \mathcal{L}_{\mathsf{Sign}} \cup \{(m, \sigma)\}$
7 **return** σ

Figure 1: EUF-CMA security (left) and SUF-CMA security (right) of a signature scheme $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ with differences highlighted in gray.

As mentioned before, secure signature schemes can be built from identification protocols. Identification protocols allow a so-called *prover* that holds a secret key sk to authenticate to a *verifier* who holds the corresponding public key pk. Here we only consider so-called *canonical* identification protocols, which consist of three moves: The prover P sends a commitment com to the verifier V. The verifier V then samples a random challenge ch and sends it to P. Finally, P sends a response rsp to V, whose decision is then a deterministic function of their *conversation* $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ and P's public key. More formally:

Definition 3 (Canonical identification protocol)**.** *A* canonical identification protocol $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P} = (\mathsf{P}_1, \mathsf{P}_2), \mathsf{V} = (\mathsf{V}_1, \mathsf{V}_2))$ *is a triple of algorithms:*

Key generation KGen *takes as input the public parameters* pp *and outputs a public key* pk *and a secret key* sk*, i.e.,* $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$.

The ***prover*** $\mathsf{P} = (\mathsf{P}_1, \mathsf{P}_2)$ *is a two-stage algorithm that takes as input a secret key* sk*.* P_1 *takes as input* sk *and outputs a commitment* com *as well as some state* st*.* P_2 *takes as input the challenge* ch *(sent by the verifier* V_1*) as well as state* st *and outputs a response* rsp.

The ***verifier*** $\mathsf{V} = (\mathsf{V}_1, \mathsf{V}_2)$ *is a two-stage algorithm that is initialized with a public key* pk*.* V_1 *selects a random challenge* ch *and sends it to the prover.* V_2 *takes as input the public key, the* $\mathsf{com}, \mathsf{ch}$ *and* rsp *and outputs* 1 *if it* accepts *the* conversation $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ *for* pk *or* 0 *if it* rejects.

For all $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$*, we require that if* $\mathsf{P}(sk)$ *and* $\mathsf{V}(pk)$ *interact honestly within an instance of the protocol, then the verifier accepts. I.e., for* $(\mathsf{com}, st) \xleftarrow{\$} \mathsf{P}_1(sk)$*,* $\mathsf{ch} \xleftarrow{\$} \mathsf{V}_1$*, and* $\mathsf{rsp} \xleftarrow{\$} \mathsf{P}_2(\mathsf{com}, \mathsf{ch}, st)$*, we have verifier* $\Pr[1 \leftarrow \mathsf{V}_2(pk, \mathsf{com}, \mathsf{ch}, \mathsf{rsp})] = 1$.

We sometimes denote the interactive run of the identification protocol between the prover and the verifier by $\mathsf{P} \leftrightarrows \mathsf{V}$ or $\mathsf{P}(sk) \leftrightarrows \mathsf{V}(pk)$. We write $\mathsf{Trans}\left[\mathsf{P}(sk) \leftrightarrows \mathsf{V}(pk)\right]$ to denote a conversation $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$

resulting from the interaction between P and V and identify $\mathsf{V}_2(pk, \mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ with the final decision $\in \{0,1\}$ of the verifier. If $1 \leftarrow \mathsf{V}_2(pk, \mathsf{com}, \mathsf{ch}, \mathsf{rsp})$, we say that $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ is an *accepting conversation for pk*, or simply a *valid conversation.*

Intuitively, the basic security of identification protocols is defined in terms of the inability of an adversary $\mathcal{A}$ to impersonate the prover towards an honest verifier without knowledge of the prover's secret key. This can be in the setting where $\mathcal{A}$ only has access to the public key of the prover (called IMP-KOA for security against impersonation under key-only attacks), or in the stronger setting, where $\mathcal{A}$ can observe honest conversations between the prover and the verifier (IMP-PA for security against impersonation under passive attacks):

Definition 4 (IMP-KOA and IMP-PA security)**.** *Let $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P}, \mathsf{V})$ be a canonical identification protocol and let $\mathcal{A}$ be an algorithm. Consider the security game $\mathrm{G}^{\text{IMP-KOA}}_{\mathcal{CID},\mathcal{A}}$ as defined on the left in Fig. 2. We say that $\mathcal{CID}$ is* (t,ϵ)-secure against impersonation under key-only attacks*, or simply* (t,ϵ)-IMP-KOA*-secure, if for any algorithm $\mathcal{A}$ running in time at most t the probability $\Pr\left[\mathrm{G}^{\text{IMP-KOA}}_{\mathcal{CID},\mathcal{A}}(\mathsf{pp}) = 1\right] \leq \epsilon$.*

Similarly, consider the game $\mathrm{G}^{\text{IMP-PA}}_{\mathcal{A}}$ as defined on the right in Fig. 2. We say that $\mathcal{CID}$ is (t,ϵ,Q_T)-secure against impersonation under passive attacks*, or simply* (t,ϵ,Q_T)-IMP-PA*-secure, if for any algorithm $\mathcal{A}$ running in time at most t and with at most Q_T queries to the oracle $\mathcal{O}_{\mathsf{Trans}}$, the probability $\Pr\left[\mathrm{G}^{\text{IMP-PA}}_{\mathcal{CID},\mathcal{A}}(\mathsf{pp}) = 1\right] \leq \epsilon$.*

$\mathrm{G}^{\text{IMP-KOA}}_{\mathcal{CID},\mathcal{A}}(\mathsf{pp})$:
1 $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$
2 $\mathsf{ch} \xleftarrow{\$} \mathsf{V}_1$
3 $(\mathsf{com}, st) \xleftarrow{\$} \mathcal{A}(pk)$
4 $\mathsf{rsp} \xleftarrow{\$} \mathcal{A}(\mathsf{ch}, st)$
5 **return** $[\![\mathsf{V}_2(pk, \mathsf{com}, \mathsf{ch}, \mathsf{rsp})]\!]$

$\mathrm{G}^{\text{IMP-PA}}_{\mathcal{CID},\mathcal{A}}(\mathsf{pp})$:
1 $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$
2 $\mathsf{ch} \xleftarrow{\$} \mathsf{V}_1$
3 $(\mathsf{com}, st) \xleftarrow{\$} \mathcal{A}(pk)$
4 $\mathsf{rsp} \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Trans}}}(\mathsf{ch}, st)$
5 **return** $[\![\mathsf{V}_2(pk, \mathsf{com}, \mathsf{ch}, \mathsf{rsp})]\!]$

$\mathcal{O}_{\mathsf{Trans}}$:
6 **return** $\mathsf{Trans}[\mathsf{P}(sk) \leftrightarrows \mathsf{V}(pk)]$

Figure 2: IMP-KOA and IMP-PA security of $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P}, \mathsf{V})$ against impersonating adversaries $\mathcal{A}$ with differences highlighted in gray.

To argue about the security of canonical identification protocols, it is useful to talk about the min-entropy of an identification scheme as well as the notion of *honest-verifier zero-knowledge*,or HVZK, for short. The former notion captures the unpredictability of commitments in the protocol, whereas HVZK formalizes the property that an adversary $\mathcal{A}$ gains no additional knowledge from honest interactions $\mathsf{P} \leftrightarrows \mathsf{V}$, since $\mathcal{A}$ could generate such conversations on its own.

Definition 5 (Min-entropy of identification scheme)**.** *We say that a canonical identification protocol $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P}, \mathsf{V})$ has α* bits min-entropy *if the probability over the choice $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$, that the commitment generated by $\mathsf{P}_1(sk)$ is from a distribution with at least α bits of min-entropy, is at least $1 - 2^{\alpha}$. Recall that a discrete random variable X has α bits of min-entropy, denoted by $H_\infty(X) := \alpha$, if $\max_x(\Pr[X = x]) = 2^{-\alpha}$.*

PTT: Probablistic Polynomial Time

Definition 6 (Honest-verifier zero-knowledge)**.** *Let $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P}, \mathsf{V})$ be a canonical identification protocol. We say that $\mathcal{CID}$ is* ϵ_{zk}-honest-verifier zero-knowledge*, or* ϵ_{zk}-HVZK *for short, if there exists a PPT algorithm* Sim*, called the* simulator*, such that for all $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$, the outputs of $\mathsf{Sim}(pk)$ can only be distinguished from real conversations $\mathsf{P}(sk) \leftrightarrows \mathsf{V}(pk)$ with probability at most ϵ_{zk}.*

The schemes discussed in this paper will have two important properties. They are *commitment recoverable*, which means that commitments can be efficiently and publicly computed from the public key, the challenge, and the response. Furthermore, they have *(computationally) unique responses*, meaning that for a fixed instance of the protocol and commitments and challenges, there exists at most one response such that the verifier will accept the conversation (or a second response is computationally infeasible to find). More formally:

Definition 7 ((Computationally) unique responses CUR)**.** *Let $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P}, \mathsf{V})$ be a canonical identification protocol. We say that $\mathcal{CID}$ has* ϵ_{cur}-computationally unique responses *or* CUR*, if for any*

$(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$, $\mathsf{com} \xleftarrow{\$} \mathsf{P}_1(sk)$ *and* $\mathsf{ch} \xleftarrow{\$} \mathsf{V}_1$ *the probability of an adversary being able to output two responses* rsp *and* rsp' *such that* $\mathsf{V}_2(pk, \mathsf{com}, \mathsf{ch}, \mathsf{rsp}) = 1$ *and* $\mathsf{V}_2(pk, \mathsf{com}, \mathsf{ch}, \mathsf{rsp}') = 1$ *is at most* ϵ_{cur}. *If* $\epsilon_{\mathsf{cur}} = 0$ *we say that* $\mathcal{CID}$ *has* unique responses.

3.2 The Fiat-Shamir Transform

Finally, we show the transform from secure identification protocols to secure signature schemes. In this work, we follow the approach by Abdalla et al. [27] when applying the Fiat-Shamir transform, i.e., we start from a canonical identification protocol that is secure against impersonation under passive attack and model the hash function as a random oracle (cf. Section 3.3) to show the existential unforgeability of the resulting signature scheme.

Let $\mathcal{CID} = (\mathcal{CID}.\mathsf{KGen}, \mathsf{P}, \mathsf{V})$ be an IMP-PA-secure canonical identification protocol and let $\mathsf{H} : \{0,1\}^\star \to \{0,1\}^\lambda$ be a cryptographic hash function with output length λ modeled as a random oracle. Then the signature scheme $\mathsf{FS}[\mathcal{CID}, \mathsf{H}] = (\mathsf{FS.KGen}, \mathsf{Sign}, \mathsf{Vfy})$ constructed as described in Fig. 3 is existentially unforgeable under chosen message attacks.

$\underline{\mathsf{FS.KGen}(\mathsf{pp})}$:	$\underline{\mathsf{Sign}(sk, m)}$:	$\underline{\mathsf{Vfy}(pk, m, \sigma)}$:
1 $(pk, sk) \xleftarrow{\$} \mathcal{CID}.\mathsf{KGen}(\mathsf{pp})$	3 $(\mathsf{com}, st) \xleftarrow{\$} \mathsf{P}_1(sk)$	8 $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp}) \leftarrow \sigma$
2 **return** (pk, sk)	4 $\mathsf{ch} \xleftarrow{\$} \mathsf{H}(\mathsf{com}, m)$	9 **return** $[\![\mathsf{ch} = \mathsf{H}(\mathsf{com}, m) \wedge \mathsf{V}_2(pk, \mathsf{com}, \mathsf{ch}, \mathsf{rsp})]\!]$
	5 $\mathsf{rsp} \xleftarrow{\$} \mathsf{P}_2(\mathsf{ch}, st)$	
	6 $\sigma \leftarrow (\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$	
	7 **return** σ	

Figure 3: Signature scheme $\mathsf{FS}[\mathcal{CID}, \mathsf{H}] = (\mathsf{FS.KGen}, \mathsf{Sign}, \mathsf{Vfy})$ resulting from the (transcript variant of) Fiat-Shamir applied to the canonical identification protocol $\mathcal{CID} = (\mathcal{CID}.\mathsf{KGen}, \mathsf{P}, \mathsf{V})$.

Note that there are different variants of the Fiat-Shamir transform in terms of how the signatures are constructed. The transform shown in Fig. 3 is of the *transcript variant* as used, e.g., by Pointcheval and Stern [25], where the signature consists of the entire conversation of the identification scheme.

Schnorr signatures as described in Appendix A are of the *challenge variant* where the signature consists only of the challenge and the response, i.e., $\sigma \leftarrow (\mathsf{ch}, \mathsf{rsp})$. This requires that there exists an algorithm that can reconstruct the commitment com from the public key, the challenge, and the response. Further signatures that are of the challenge variant are the original work by Fiat and Shamir, GQ signatures [39] and Okamoto signatures [40]. An in-depth treatment of these variants, including a third variant, the *commitment variant*, can be found in Backendal et al. [41]. As we will later see, Ed25519 signatures are a deterministic variant of Schnorr signatures but in the commitment-variant of the Fiat-Shamir transform. In order to make Fiat-Shamir signatures as described in Fig. 3 deterministic, P_1 is *derandomized* by using $\mathsf{H}(sk, m)$ as randomness during commitment generation.

3.3 The Random Oracle Methodology

When applying the Fiat-Shamir transform, we do this in the so-called *random oracle model*. This model was first introduced by Bellare and Rogaway [26] and enabled security proofs for many efficient schemes that previously had eluded the provable security paradigm. It does so by representing a hash function in a cryptographic scheme as an idealized random function (the *random oracle*). With this idealization in place, an adversary can *only* evaluate the hash function H on input x, if it queries this random oracle on x. It is no longer able to simply evaluate the hash function locally. In particular, this allows us to "peek" at the adversary's inputs to the hash function, a property of the model that is often referred to as *extractability*. When queried on input x, the random oracle then returns uniformly random answers from the range of H for each input. For each new query a fresh uniformly random output is sampled, but just like for real hash functions, repeated queries are answered *consistently*, i.e., the same inputs yield the same outputs. In the Fiat-Shamir transform another (optional) property of the random oracle is used, namely its *programmability*: if the adversary queries some input x for the first time, we can set the value $\mathsf{H}(x)$ to a specific, freely-chosen output value y as long as it is correctly distributed and does not collide with previously set outputs.

Note that when our proofs are in the random oracle model for hash function H, the security notions introduced previously get the extra query parameter Q_{H} of the maximal number of queries the adversary makes to the random oracle. For example, for a signature scheme in the random oracle model we will speak of $(t, \epsilon, Q_s, Q_{\mathsf{H}})$-EUF-CMA security instead of (t, ϵ, Q_s)-EUF-CMA security.

3.4 Elliptic Curves

Lastly we briefly recap the main theory of elliptic curves relevant for this paper. For a more in-depth treatment of specific concepts and constructions, we refer the interested reader to, e.g., [42], [43]. We begin by defining elliptic curves over a finite field $\mathbb{F}_q$, which are the most common types of elliptic curves in cryptography:

Definition 8 (Elliptic curve)**.** *Let $q \geq 5$ be a prime. An* elliptic curve E *defined over the finite field $\mathbb{F}_q$ is an equation of the form $y^2 = x^3 + ax + b$. with $a, b \in \mathbb{F}_q$ such that $4a^3 + 27b^2 \neq 0$.*

In elliptic-curve cryptography, the group in question is the set of points on the elliptic curve E.

Definition 9 (Points on E)**.** *Let $E(\mathbb{F}_q)$ be the set of pairs $(x, y) \in \mathbb{F}_q \times \mathbb{F}_q$ satisfying the elliptic curve equation. Let $\mathcal{O}$ denote a special point, the so-called* point at infinity*. Then the set $E(\mathbb{F}_q) := \{(x, y) | x, y \in \mathbb{F}_q \wedge y^2 = x^3 + ax + b\} \cup \{\mathcal{O}\}$ denotes the* points on the elliptic curve *E.*

With an adequately defined addition operation "+" $E(\mathbb{F}_q)$ forms a group with neutral element $(0, 1)$. The multiplication of a curve point P with an integer n is defined as adding P n times to itself, i.e., $nP := \underbrace{P + P + \cdots + P}_{n \text{ times}}$ where $0P := \mathcal{O}$.

For brevity, we often write E instead of $E(\mathbb{F}_q)$ if the underlying field is clear from context.

Further Definitions

The number of points on an elliptic curve E over $\mathbb{F}_q$ is called the *order* of the curve and is denoted by $|E(\mathbb{F}_q)|$. We call an element B that generates a cyclic subgroup the *base point* and write $P \in \langle B \rangle$ to indicate that P is an element of the subgroup generated by B. For an element B, we overload notation and write $|B|$ to denote its order, i.e., the smallest integer n such that $nB = \mathcal{O}$. If B generates a subgroup of $E(\mathbb{F}_q)$, we define the *cofactor* to be the integer $\frac{|E(\mathbb{F}_q)|}{|B|}$.

Cofactor here

3.4.1 Different Forms of Curves

There exist several different forms of elliptic curve equations, such as *Weierstraß*, *Montgomery* or *Edwards* form. Most relevant for this paper are *twisted Edwards curves* [44] which are defined over a finite field $\mathbb{F}_q$ with $q > 3$ prime with additional parameter a (the *twist*) via the curve equation $E_{TEd} : ax^2 + y^2 = 1 + dx^2y^2$, where $a, d \in \mathbb{F}_q$ with $a, d \neq 0$ and $a \neq d$.

Addition "+" on $E_{TEd}(\mathbb{F}_q)$ is defined as follows. Let $P = (x_1, y_1), Q = (x_2, y_2) \in E_{TEd}(\mathbb{F}_q)$, then $P + Q = (x_3, y_3)$ is defined as

$$x_3 = \frac{x_1y_2 + x_2y_1}{1 + dx_1x_2y_1y_2}, \qquad y_3 = \frac{y_1y_2 - ax_1x_2}{1 - dx_1x_2y_1y_2}.$$

Note that if a is a square in $\mathbb{F}_q$ and d is non-square in $\mathbb{F}_q$, then the addition operation is complete and $(E_{TEd}(\mathbb{F}_q), +)$ is a group with neutral element $(0, 1)$. The inverse $-P$ of a point $P = (x, y) \in E_{TEd}(\mathbb{F}_q)$ is $(-x, y)$.

The twisted Edwards curve E underlying the Ed25519 constructions, which we discuss in more detail in the next section, is birationally equivalent to `curve25519` introduced by Bernstein [45], which is of the Montgomery form and due to its efficient arithmetic implementation yields very performant constructions. `curve25519` is defined over the finite field $\mathbb{F}_q$ with $q = 2^{255} - 19$ prime via the curve equation `curve25519`: $y^2 = x^3 + 486662x^2 + x$.

3.4.2 The Elliptic Curve Discrete Logarithm Problem

ECDLP - Elliptic Curve Discrete ..

The security of Ed25519 signatures will ultimately be reduced to a common complexity-theoretic hardness assumption in cryptography: the discrete logarithm problem.

Definition 10 (ECDLP)**.** *Let E be an elliptic curve defined over a finite field $\mathbb{F}_q$ and let $B \in E(\mathbb{F}_q)$ be a point of order n. Let $P \in \langle B \rangle$. Then the* elliptic curve discrete-log problem *is to find an integer $0 \leq k < n$ such that $P = kB$. We say that E is (t, ϵ)-*hard on E *if for any algorithm $\mathcal{A}$ running in time at most t the probability of solving* ECDLP *is at most ϵ.*

4 Ed25519 Signatures

In this section we describe how the Ed25519 [1], [7] signature scheme operates in detail, unravel its relationship with Schnorr signatures and why proofs for Schnorr are not directly applicable to Ed25519. We also describe several of the proposed variants of Ed25519, which target stronger security properties than provided by the original formulation.

We define the generic signature scheme Ed25519 in Fig. 4. Part of the generic scheme description, highlighted on line 13 in Fig. 4, is replaced in the variant schemes. We summarise these variations in Table 2 and discuss them further below. The various parameters common to all variants are listed in Table 1. These parameters include those necessary to define the elliptic curve E over which the signature scheme operates and the hash function used.

Encodings Integers mod L are encoded as as b-bit strings in little endian format. Elliptic curve points (x, y) are encoded as a $(b-1)$-bit little-endian encoding of y, followed by a sign bit which is 1 if and only if x is negative. We note an oft-omitted property of the encoding scheme: the field element that represents y is encoded as a $(b-1) = 255$-bit string, but the size of the field is $q = 2^{255} - 19$, yielding a larger space of encodings than actual elements. Similarly, the S part of the signature is expected to be a b-bit integer, but is necessarily reduced mod L prior to use in the signature verification. These details turn out to have substantial consequences when showing security of the scheme and we will highlight this in the respective proofs. In places, the original presentation [1] and its accompanying source code disagree on the necessary tests, e.g., the range of L. As the source code is the basis of the most popular Ed25519 implementations, we treat it as authoritative.

4.1 Variants of Ed25519

Recall that we refer to the original, and currently most widely deployed formulation of Ed25519 as Ed25519-Original. Several alternative specifications have also been published which largely maintain wire compatibility with Ed25519-Original, but substantially alter the security properties of the scheme. In particular, variants have been published by the IETF [7], NIST [46], and the ZCash Foundation [9]. Furthermore, LibSodium [8], one of the most popular cryptographic libraries, uses a set of additional checks that render it a de-facto standard.

Another variant is the use of Ristretto encodings [47] for Ed25519. Whilst this appears promising, the draft RFC [48] is still under active development and we do not analyze it here.

We note that the Signal Foundation [34] have also proposed a variant with enhanced resistance to side-channel attacks and fault resistance during signature generation. However, this variant operates similarly to Ed25519-Original with regards to signature verification and consequently we do not treat it separately.

4.1.1 Pre-Hashing Variants

The IETF-standardised RFC 8032 [7] and the NIST draft standard [46] support a *pre-hashing* mode for their variants. This mode allows implementations to sign large messages whilst only needing to perform a single pass over the message. The signed message value m is replaced with the pre-hash $\mathsf{PH}(m)$, where PH

Parameter	Description	Instantiation for Ed25519
q	The finite field size q	$2^{255} - 19$
n	Secret scalars are $n+1$-bits	254
b	The public key bit-length $2^{b-1} > q$	256
a, d	Curve parameters in $\mathbb{F}_q$	$-1, -121665/121666$
B	A generator of the prime order subgroup of E	$(x, 4/5), x > 0$
c	The $\log_2$ of curve cofactor	$c = 3$
L	The prime generator order $LB = 0$ and $2^c L = \lvert E \rvert$	$2^{252} + 27742\ldots8493$
H	Secure hash function producing $2b$-bit output	SHA-512
E	Curve equation in Twisted Edwards form	$x^2 + y^2 = 1 + dx^2y^2$

Ed25519 curve parameters here

Table 1: Parameters for Ed25519 signatures as described in Fig. 4 on the following page.

Ref	Scheme	Variant Specific Checks	EUF-CMA (Thm 3)	SUF-CMA (Thm 4)	S-UEO (Thm 5)	M-S-UEO (Thm 6)	MBS (Thm 7)
[49]	Ed25519-Original	$S \in \{0, ..., 2^b - 1\}$	✓	×	✓	×	×
[7]	Ed25519-IETF	$S \in \{0, ..., L - 1\}$	✓	✓	✓	×	×
[8]	Ed25519-LibS	$S \in \{0, ..., L - 1\} \wedge \lvert R\rvert \geqslant L \wedge \lvert A\rvert \geqslant L$	✓	✓	✓	✓	✓

Table 2: Ed25519 Schemes. To form each variant, replace the highlighted section in Fig. 4 with the text presented here. Note that $2^b - 1 > L$ (see Table 1) so the latter checks are stricter. EUF-CMA: Existential Unforgeability; SUF-CMA: Strong Unforgeability; S-UEO and M-S-UEO denote resilience against key substitution attacks; MBS: Message Bound Security, ensuring a signature verifies a unique message, even for malicious keys.

KGen(pp):
1 $k \xleftarrow{\$} \{0,1\}^b$
2 $h \leftarrow \mathsf{H}(k)$
3 $s \leftarrow 2^n + \sum_{i=c}^{n-1} 2^i h[i]$
4 $A \leftarrow sB$
5 **return** $(\underline{A}, k)$

Sign(k, m):
6 $h \leftarrow \mathsf{H}(k)$
7 $s \leftarrow 2^n + \sum_{i=c}^{n-1} 2^i h[i]$
8 $r \leftarrow \mathsf{H}(h[b], \ldots, h[2b-1], m)$
9 $R \leftarrow rB$
10 $S \leftarrow (r + \mathsf{H}(\underline{R}, \underline{A}, m)s) \mod L$
11 **return** $\sigma = (\underline{R}, \underline{S})$

Vfy$(\underline{A}, \sigma = (\underline{R}, \underline{S}), m)$:
12 Check $R, A \in E$
13 Variant Specific Checks
14 **return**
Checks succeed $\wedge$ $[\![2^c SB = 2^c R + 2^c \mathsf{H}(\underline{R}, \underline{A}, m)A]\!]$

Figure 4: Generic description of the Ed25519 signature scheme algorithms KGen, Sign, and Vfy. Note that the highlighted line (13) varies depending on the version of Ed25519 and the appropriate check is listed in Table 2.

.. where PH ..
is a hash function. The IETF specification explicitly recommends against the use of this mode, stressing it is included only to support legacy signing interfaces. Consequently, we do not discuss it further.

4.1.2 Bounds Checking Variants

Some variants of Ed25519 require an additional check on the received alleged signatures during verification. In particular, this is enforced by the IETF standard [7] and proposed in the NIST draft standard [46]. In these variants, implementers are required to reject signatures whose S parameter is equal to or larger than L, where L is the order of the prime-order subgroup. Contrastingly, Ed25519-Original implementations merely check S is a 256-bit integer. We refer to such implementations as Ed25519-IETF. This is claimed to have a substantial impact and achieve strong unforgeability. We investigate this further in Section 5 but already note that, contrastingly to Ed25519-IETF variants, Ed25519-Original implementations simply reduce the received signature element S mod L during signature verification, thereby immediately ruling out SUF-CMA security.

Why does it rule out that security property?

4.1.3 Point and Bound Checking Variants

Some popular implementations of Ed25519, such as LibSodium [8], perform the bounds check on S values in addition to point validation. Specifically, Ed25519-LibS ensures that received elements are canonically encoded and have order $\geqslant L$. This check means that certain R values, which are low order points on the curve, are rejected during verification as well as low order public keys. Additionally, R values and public keys which have y-coordinates above q are also rejected to ensure unique encodings. We see the impact of these decisions in Section 5.4.

4.2 Differences from Schnorr Signatures

Clamping here

Ed25519 has several notable features which differentiate it from traditional Schnorr signatures. The original Schnorr signature scheme and its underlying identification protocol can be recalled in Appendix A. In particular, private keys in Ed25519 are *clamped* to a specific format, signature nonces are chosen deterministically, and signed messages are prefixed with public keys. As indicated before, these differences impact the security of the overall signature scheme and our analysis. We next discuss these alterations in more detail.

4.2.1 Group Structure

Ed25519's curve is of order $8L$ for L a large prime defined in Table 1. Contrastingly, Schnorr signatures are typically constructed over prime order groups and implementations are assumed to reject the identity element.

Non-prime order groups contain a more complex group structure than prime order groups. In particular, non-prime order groups entail the presence of additional subgroups, whose elements lie outside the intended prime order subgroup. Performing group operations on these elements can lead to surprising results, including confinement under exponentiation, where they map to a small range of elements and leakage, where performing exponentiation with a private scalar leaks information about that scalar. The original paper [1] allowed Ed25519 implementations to optionally include multiplication by the cofactor in the verification equation. Including the cofactor makes the verification function strictly more permissive and we assume it is present so that our proofs carry over to the cofactorless case.

Proofs about systems defined over prime order groups do not necessarily hold if the system is implemented with non-prime order groups, even when the proof does not explicitly rely on the prime order structure. For example, proofs typically assume that any group element can be written as the exponentiation of a fixed generator, or that exponentiation of an element is uniformly distributed over the group. In the non-prime order case, neither assumption is true in general.

4.2.2 Group Element Checks

The question of how values should be decoded and parsed is often omitted from academic papers. Ed25519-Original and Ed25519-IETF are of particular interest in this regard as it is explicitly argued that elements do not need to be checked to ensure they belong to the prime order subgroup. This means any group element, including small order elements, the identity element and elements of order $8L$ will be accepted. We discussed variants that mandate this check in Section 4.1.3. The related Diffie-Hellman function X25519 also omits these checks which has previously lead to otherwise avoidable attacks on protocols employing it [50].

A related issue is whether elements are checked to ensure they lie on the intended curve, rather than its twist. Whilst X25519 does not reject elements on the twist, Ed25519-Original explicitly mandates that points are checked to ensure they do belong to E. This is checked during the decompression of received points prior to signature verification. We assume all implementations uphold this requirement as otherwise point addition during signature verification is not necessarily defined.

4.2.3 Private Key Clamping

In part due to the non-prime order nature of the curve and the lack of group element validation, Ed25519 mandates the use of *key clamping* which involves the bitwise manipulation of private keys prior to use in signing. The rationale behind this requirement has been the subject of much debate [51], [52]. The original Ed25519 paper defines private keys, without discussion, such that a high bit is always set and three low bits are cleared. All subsequent variations have kept the same requirement. There are two rationales for clamping:

- Setting the high bit ensures that some deficient point multiplication implementations, which have variable execution time with respect to the position of the highest set bit in the scalar, become constant time [53].
- Clearing the low bits ensures that the scalar is a multiple of the cofactor. This ensures that the result of applying the scalar to any group element results in an element in the prime order subgroup. This avoids key leakage attacks, although these attacks are not relevant for Ed25519 signature schemes as private keys are never applied to adversary-provided group elements. However, as implementers may wish to re-use keys in both X25519 and Ed25519, this choice provides defence in depth.

As we will see later, the use of clamping complicates our security proofs. This is because not every element in the prime order subgroup is also a valid public key as produced by the key generation algorithm. Consequently, when providing reductions which must manipulate public keys, e.g., blinding them, there is a small chance an invalid public key is produced and the reduction must abort.

4.2.4 Key Prefixing

Unlike traditional Schnorr signatures, Ed25519 uses key prefixing, where the signature scheme, prior to signing or verification, prepends the public key to the message. This choice has also been the subject of much debate [28], [31], [54], [55] as to whether it provides a substantial improvement in security. Much of the discussion has revolved around multi-user security and whether key prefixing improves security in the presence of an adversary who is satisfied to break some subset of witnessed multiple keys, rather than one key in particular. We discuss the multi-user security of Ed25519 in more detail in Appendix 5.3. It transpires that key prefixing also has benefits when considering lesser-known multi-user security properties such as message key substitution attacks, as we show in Section 5.4.

4.2.5 Deterministic Nonce Generation

Signature schemes require the use of a nonce with each signed message. This has historically been an area prone to subtle implementation mistakes leading to critical real world vulnerabilities [56]. Ed25519 uses *deterministic signing* which removes the need for fresh random numbers during the signing process. This does not lead to any particular consequences for our security analysis since we model the key derivation function as a random oracle. However, it is well known not to reduce security [57].

5 The Security of Ed25519

We now present our security results for Ed25519-Original and Ed25519-IETF signatures. As suggested by earlier works that informally discuss the security of these schemes, we use the Fiat-Shamir approach. However, as elaborated in Section 4.2, there exist marked differences between Schnorr signatures and Ed25519 signatures such that the established security results for Schnorr do not hold without careful adjustment to the Ed25519 setting. We close this gap by proving both the existential unforgeability of Ed25519-Original and show that due to the additional check on the value S, Ed25519-IETF and Ed25519-LibS achieve strong unforgeability. As is common for signature proofs, we assume idealized versions of hash functions (random oracles), but do not make any strong assumptions on the properties of the underlying elliptic curve group. Tighter security bounds may be possible in the so-called *generic group model* (cf., e.g., [28], [58]), however we explicitly want to explore security in a setting where the adversary may take advantage of, e.g., encoding details.

We recall that, in the following, E refers to the twisted Edwards curve underlying Ed25519 (cf. Table 1) and analogously $E(\mathbb{F}_q)$ denotes the set of elements on the curve which forms a group with point addition, as defined in Section 3.

5.1 Existential Unforgeability of Original Ed25519

We start by showing EUF-CMA security of Ed25519-Original specifically by means of the Fiat-Shamir transform. We first define an appropriate canonical identification protocol $\mathcal{CID}$ in Fig. 5 to which the transform can be applied, then show that $\mathcal{CID}$ satisfies the necessary prerequisites in Theorem 1 and 2 and finally apply the transform in Theorem 3 to establish existential unforgeability of the resulting signature scheme. We note that with the additional check in Line 17 of $\mathcal{CID}$, the proof of EUF-CMA security directly carries over to Ed25519-IETF and with the further check described in Table 2 also to Ed25519-LibS.

To get from $\mathcal{CID}$ in Fig. 5 to the Ed25519-Original in Fig. 4, we apply a variant of Fiat-Shamir, denoted by $\mathsf{FS}^{\mathsf{kp}}_{\mathsf{det}}$, that captures *deterministic signing* and *key-prefixing*.

Deterministic signing is achieved by deterministically deriving the randomness r of P_1 in the signing algorithm via $r \leftarrow \mathsf{H}(h[b], ..., h[2b-1], m)$, where m is the message to be signed, and, as before, $h[i]$ denotes the i-th bit of h. The de-randomization of signing by computing the randomness deterministically as some $\mathsf{H}(sk, m)$ is common and in our case does not impact the security of the resulting signatures, assuming H is at least a pseudorandom function (cf., e.g., [57]). Where is h defined?

Recall that key-prefixing, on the other hand, describes the derivation of the challenge ch by the prover during signature generation as $\mathsf{H}(\underline{R}, \underline{A}, m)$ instead of $\mathsf{H}(\underline{R}, m)$, i.e., by also including the public key into the hash function. It will become clear from the proof that this additional input to the random oracle H does not impact security, since this solely relies on $\underline{R}$ being sufficiently unpredictable in honest signature generations.

KGen(pp):
1 $k \xleftarrow{\$} \{0,1\}^b$
2 $h \leftarrow \mathsf{H}(k)$
3 $s \leftarrow 2^{b-2} + \sum_{i=3}^{b-3} 2^i h[i]$
4 $A \leftarrow sB$
5 **return** $(\underline{A}, k)$

$\mathsf{P}_1(k)$:
6 $h \leftarrow \mathsf{H}(k)$
7 $s \leftarrow 2^{b-2} + \sum_{i=3}^{b-3} 2^i h[i]$
8 $st \leftarrow s$
9 $r \xleftarrow{\$} \{0,1\}^{2b}$
10 $R \leftarrow rB$
11 **return** $(\underline{R}, st)$

V_1:
12 $\mathsf{ch} \xleftarrow{\$} \{0,1\}^{2b}$
13 **return** ch

$\mathsf{P}_2(\mathsf{ch}, st)$:
14 $S \leftarrow (r + \mathsf{ch} \cdot st) \mod L$
15 **return** $\underline{S}$

$\mathsf{V}_2(\underline{A}, \underline{R}, \underline{S}, \mathsf{ch})$:
16 Check $R, A \in E$
17 Variant Specific Checks
18 **return** $[\![8SB = 8R + 8\mathsf{ch}A]\!]$

Figure 5: Canonical identification protocol $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P} = (\mathsf{P}_1, \mathsf{P}_2), \mathsf{V} = (\mathsf{V}_1, \mathsf{V}_2))$ underlying Ed25519-Original in Fig. 4. Note that the highlighted line (17) varies depending on the version of Ed25519 and the appropriate check is listed in Table 2 on page 10.

HVZK: Honest Verifier Zero Knowledge

5.1.1 Security of the underlying identification protocol

We show that the underlying $\mathcal{CID}$ is secure against impersonation attacks by passive adversaries (IMP-PA security, cf. Definition 4). We do this in a two-step process by first showing in Theorem 1 that $\mathcal{CID}$ is secure against impersonating adversaries that only have access to the public key (IMP-KOA security). To lift this result to the setting of passive impersonating adversaries, we then only need to be able to simulate queries to the oracle $\mathcal{O}_{\mathsf{Trans}}$, which can be achieved by the HVZK property of $\mathcal{CID}$. Thus, in Lemma 1 we show that $\mathcal{CID}$ is HVZK and, combined with the IMP-KOA security, we achieve IMP-PA security.

Theorem 1 ($\mathcal{CID}$ is IMP-KOA secure)**.** *Let $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P}, \mathsf{V})$ be the identification protocol as defined in Fig. 5. If* ECDLP *is $(t, \epsilon_{\mathsf{ecdlp}})$-hard on $E(\mathbb{F}_q)$, then $\mathcal{CID}$ is (t', ϵ')-*IMP-KOA *secure, where $t \approx 2t'$ and $\frac{2^{b-5}}{L}\left(\epsilon' - \frac{1}{L}\right)^2 \leq \epsilon_{\mathsf{ecdlp}}$.*

ECDLP: Elliptic Curve Discrete Log Problem

Proof Sketch. The full proof can be found in the Appendix B.1 We notice the first marked difference to proofs of Schnorr signatures. While in the latter, we have a straightforward reduction to ECDLP using the *rewinding technique* [24], we now need to account for the secret key clamping in Ed25519. The clamping causes that an element $A \in E(\mathbb{F}_q)$ is not necessarily a valid public key output of KGen and thus cannot be relayed by the reduction to the IMP-KOA adversary $\mathcal{A}$, resulting in the loss of a factor of $\frac{2^{b-5}}{L}$. As in the Schnorr setting, the reduction exploits the property that from two valid conversations $(\underline{R}^\star, \mathsf{ch}_1, \underline{S}_1)$ and $(\underline{R}^\star, \mathsf{ch}_2, \underline{S}_2)$ for the public key $\underline{A}$ with $\mathsf{ch}_1 \neq \mathsf{ch}_2 \pmod L$, we can extract the value $s = \frac{S_1 - S_2}{\mathsf{ch}_1 - \mathsf{ch}_2} \mod L$ such that $A = sB$. The *Reset Lemma* [59] which is the analogue of the *Forking Lemma* [24] for identification protocols instead of signatures, ensures that two such conversations can be found with probability at least $(\epsilon' - \frac{1}{L})^2$, where ϵ' is the success probability of $\mathcal{A}$. We note the hardness of ECDLP on $E(\mathbb{F}_q)$ as used in Ed25519 carries over from the respective hardness on curve25519 due to their birational equivalence [6], [7]. □

As mentioned before, we now lift this result to show that an adversary cannot impersonate the prover, even when given access to an oracle $\mathcal{O}_{\mathsf{Trans}}$, that outputs valid accepting conversations $\mathsf{Trans}[\mathsf{P} \rightleftarrows \mathsf{V}]$. This is due to the fact, that the underlying canonical identification protocol $\mathcal{CID}$ is ϵ_{zk}-HVZK, i.e., there exists a simulator Sim which takes as input only the public key pk outputs conversations $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ which are ϵ_{zk}-indistinguishable from real conversations $\mathsf{P}(sk) \leftrightarrows \mathsf{V}(pk)$. The simulator $\mathsf{Sim}(pk)$ for $\mathcal{CID}$ is defined in Fig. 6 and exploits the fact that commitments can be recovered from the public key, the challenge, and the response.

Lemma 1. *Let $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P}, \mathsf{V})$ be as defined in Fig. 5. Then $\mathcal{CID}$ is* HVZK *with $\epsilon_{\mathsf{zk}} = 0$.*

Theorem 2 ($\mathcal{CID}$ is IMP-PA secure)**.** *Let $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P}, \mathsf{V})$ be the ϵ_{zk}-*HVZK *and (t, ϵ)-*IMP-KOA*-secure identification protocol as defined in Fig. 5. Then $\mathcal{CID}$ is (t', ϵ', Q_T)-*IMP-PA*-secure with $t \approx t'$ and $\epsilon' \leq \epsilon$.*

Proof. We have shown in Lemma 1 that $\mathcal{CID}$ is ϵ_{zk}-HVZK with $\epsilon_{\mathsf{zk}} = 0$. Since $\mathsf{Sim}(pk)$ uses public information only, any resulting conversations could also have been computed by $\mathcal{A}$ itself. $\mathcal{A}$ therefore

HVZK: Honest Verifier Zero Knowledge

IMP-PA: Impersonation attacks by Passive Adversaries

IMP-KOA: impersonating adversaries that only have access to the public key

existential unforgeability under chosen message attacks (EUF-CMA)

strong unforgeability under chosen message attacks (SUF-CMA)

$\mathsf{Sim}(\underline{A})$:
1 $\mathsf{ch} \xleftarrow{\$} \{0,1\}^{2b}$
2 $\tilde{s} \xleftarrow{\$} \{0,1\}^{2b}$
3 $S \leftarrow \tilde{s} \mod L$
4 $R \leftarrow (SB - \mathsf{ch}A) \mod L$
5 **return** $(\underline{R}, \mathsf{ch}, \underline{S})$

Figure 6: Simulator Sim for $\mathcal{CID}$.

A therefore ..

learns nothing from the interaction of $\mathsf{P} \leftrightarrows \mathsf{V}$ via $\mathcal{O}_{\mathsf{Trans}}$ (replaced by Sim). Thus, for canonical identification protocols that are HVZK, IMP-PA security is equivalent to IMP-KOA security and we have ave $\epsilon' \leq \epsilon$ and $t \approx t'$ plus the running time of the Sim at most Q_T times. Since t is dominated by t', we simply write $t \approx t'$. □

5.1.2 Applying the Fiat-Shamir transform

Now that we have shown that the identification protocol satisfies the prerequisites for the Fiat-Shamir transform, we can apply the transform to show that Ed25519 is existentially unforgeable. We state the theorem in terms of the most basic version Ed25519-Original. For Ed25519-IETF and thus Ed25519-LibS the proof below only needs to account for the difference in the verification algorithm.

Theorem 3 (Ed25519-Original is EUF-CMA secure)**.** *Let* $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P}, \mathsf{V})$ *be the* $(t, \epsilon, Q_T, (Q_\mathsf{H} + Q_\kappa + Q_\beta))$-IMP-PA-*secure identification protocol as defined in Fig. 5 with* $\mathsf{H} : \{0,1\}^\star \to \{0,1\}^{2b}$ *modeled as a programmable random oracle and* α *bits min-entropy. Then* $\mathsf{Ed25519\text{-}Original} = \mathsf{FS}^{\mathsf{kp}}_{\mathsf{det}}[\mathcal{CID}, \mathsf{H}]$ *as defined in Fig. 4 is* $(t', \epsilon', Q_S, (Q'_\mathsf{H} + Q_\kappa + Q_\beta))$-EUF-CMA *secure, where*

$$t \approx t' \quad \text{and} \quad \epsilon' \leq Q'_\mathsf{H} \cdot (\epsilon + Q_S Q'_\mathsf{H} 2^{-\alpha} + Q_\kappa 2^{-b})$$

for $Q_S = Q_T$ *and* $Q'_\mathsf{H} = Q_\mathsf{H} + 1$*, where* Q_κ *and* Q_β *refer to random oracle queries of a certain format (details in proof.)*

Proof. In the following we assume without loss of generality, that the adversary $\mathcal{A}$ never queries the same message twice to the oracle $\mathcal{O}_{\mathsf{Sign}}$, since Ed25519-Original is deterministic and thus $\mathcal{A}$ does not gain any advantage in doing so.

Assume there exists a $(t', \epsilon', Q_S, (Q'_\mathsf{H}+Q_\kappa+Q_\beta))$-adversary $\mathcal{A}$ against the unforgeability of the signature scheme Ed25519-Original. We show that this immediately implies a successful $(t, \epsilon, Q_T, (Q_\mathsf{H} + Q_\kappa + Q_\beta))$ adversary $\mathcal{B}$ against the IMP-PA security of the underlying identification scheme $\mathcal{CID}$. The reduction $\mathcal{B}$ receives as input a public key $\underline{A}$. $\mathcal{B}$ then runs $\mathcal{A}$ on input $\underline{A}$ as follows.

We note that the relevant random oracle queries of $\mathcal{A}$ can take three distinct and distinguishable forms: the most relevant to the reduction are those of the form $(\underline{\mathsf{com}}, \underline{A}, m)$, i.e., a b-bit string followed by the encoding $\underline{A}$, and some arbitrary-length bit-string m. The second distinct case are those queries of length exactly b bits. Any other query can be interpreted as a query of the form (β, m) with β a b-bit string and m some arbitrary-length bit-string.

Random oracle queries of the form κ: Let κ be a b-bit string. Let Q_κ be the maximum number of queries of the form κ that $\mathcal{A}$ makes to H. The reduction $\mathcal{B}$ simply forwards these queries to H and returns the answer to $\mathcal{A}$.

Random oracle queries of the form (β, m): Let β be a bit string of length b and let Q_β be the number of queries of the form (β, m) that $\mathcal{A}$ makes to H. Again, $\mathcal{B}$ simply relays these queries between H and $\mathcal{A}$.

Random oracle queries of the form $(\underline{\mathsf{com}}, \underline{A}, m)$: Let Q'_H be the (maximal) number of queries of this form that $\mathcal{A}$ makes to the random oracle H. $\mathcal{B}$ guesses the query $i \in \{1, 2, \ldots, Q'_\mathsf{H}\}$ for which $\mathcal{A}$ will eventually output the signature forgery, resulting in a loss of a factor Q'_H.

For every of the other $Q_\mathsf{H} = Q'_\mathsf{H} - 1$ queries $j \in \{1, 2, \ldots Q'_\mathsf{H}\}$ with $j \neq i$ to H, $\mathcal{B}$ simply relays the queries between $\mathcal{A}$ and H.

When the adversary $\mathcal{A}$ asks the i-th query, say on $(\underline{\mathsf{com}}', \underline{A}, m')$, $\mathcal{B}$ forwards $\underline{\mathsf{com}}'$ as its commitment to its own challenger. The challenger will then send a challenge $\mathsf{ch}^\star \xleftarrow{\$} \{0,1\}^{2b}$ to $\mathcal{B}$, which $\mathcal{B}$ returns as response to $\mathcal{A}$.

Signing queries: For every of the Q_S signing queries of $\mathcal{A}$, $\mathcal{B}$ runs its own $\mathcal{O}_{\mathsf{Trans}}$ oracle to obtain an accepting conversation $(\underline{R}, \mathsf{ch}, \underline{S})$. In order for $(\underline{R}, \underline{S})$ to be a valid signature on m, $\mathcal{B}$ must ensure that $\mathsf{ch} = \mathsf{H}(\underline{R}, \underline{A}, m)$, i.e., the reduction programs the random oracle on these values to return ch.

Note that for each commitment value $\underline{R}$ that was output by $\mathcal{O}_{\mathsf{Trans}}$, the probability that the value $\mathsf{H}(\underline{R}, \underline{A}, m)$ had been set by a previous query of $\mathcal{A}$ to the random oracle H, and thus that $\mathcal{A}$ could detect the inconsistency in the patched random oracle, is upper bounded by $Q'_{\mathsf{H}} 2^{-\alpha}$, where α is the min-entropy of the identification scheme. The distribution of commitments has α bits of min-entropy for $\alpha = -\log_2\left(\lceil \frac{2^{2b}-1}{L} \rceil \cdot 2^{-2b}\right)$ due to the bias introduced by sampling first a uniformly random $2b$-bit string and then reducing it modulo L.

If this happens, the reduction $\mathcal{B}$ aborts, the probability of which is thus upper bounded by $Q_S Q'_{\mathsf{H}} \cdot 2^{-\alpha}$.

Furthermore, note that this also implies that the value $\underline{R}$ provided by the simulation via $\mathcal{O}_{\mathsf{Trans}}$ is with high probability different from the deterministic value $\underline{R}'$ with $R' = r'B$ that would be generated in the real signing process of the message m with secret key k belonging to the public key $\underline{A}$. However, $\mathcal{A}$ is not able to compute the deterministic commitment value $\underline{R}'$ by itself unless it can guess the correct value k to determine $h[b], ...h[2b-1]$ of $h \leftarrow \mathsf{H}(k)$ and thus r', the probability of which is bounded by $Q_\kappa 2^{-b}$.

Note that it does not help $\mathcal{A}$ in detecting the simulation to guess the values $h[b], ...h[2b-1]$, as $\mathcal{A}$ has no way of checking that these are the correct values leading to the "real" r' without also guessing k.

Existential Forgery: At some point $\mathcal{A}$ terminates with a forgery output $(m^\star, \sigma^\star = (\underline{R}^\star, \underline{S}^\star))$ with $\underline{R}^\star = \mathsf{com}'$ and $m^\star = m'$. If this is not the case, $\mathcal{B}$ aborts with probability $\frac{1}{Q'_{\mathsf{H}}}$ since it has wrongly guessed the index i for which the forgery will take place. Assuming this is a valid forgery in the EUF-CMA game, it holds that m' has not been queried to $\mathcal{O}_{\mathsf{Sign}}$. In particular this means that the output of $\mathsf{H}(\mathsf{com}', \underline{A}, m')$ has not been re-programmed in $\mathcal{A}$'s view by $\mathcal{B}$ to a value other than $\mathsf{ch}^\star$. Furthermore, $\mathsf{V}_2(\underline{A}, \mathsf{com}', \mathsf{ch}^\star, \underline{S}^\star) = 1$ holds such that when $\mathcal{B}$ forwards $\underline{S}^\star$ to its own challenger as final output of its game, $\mathcal{B}$ will also be successful.

The running time t of $\mathcal{B}$ is that of $\mathcal{A}$ plus the time it takes to query the random oracle H $(Q'_{\mathsf{H}} + Q_\kappa + Q_\beta)$ times, the time it takes to query its challenger, and to query $\mathcal{O}_{\mathsf{Trans}}$ $Q_T = Q_S$ times. As before, we write $t \approx t'$ since the running time of the reduction is dominated by the running time t' of $\mathcal{A}$. If $\mathcal{A}$ outputs a forgery with probability ϵ', then $\mathcal{B}$ will be able to impersonate the prover with probability $\frac{\epsilon'}{Q'_{\mathsf{H}}} - Q_S Q'_{\mathsf{H}} \cdot 2^{-\alpha} - Q_\kappa 2^{-b}$. □

5.2 Strong Unforgeability of Standardized Ed25519

Our previous results confirm that for a target public key, the adversary is not able to forge a signature on a message m for which it has not seen valid signatures beforehand. In a real-world scenario, the security provided by existential unforgeability may be insufficient, as we have mentioned before, e.g., regarding Bitcoin transaction security or SSH multi-ciphersuite security. Another commonly named example is that of blocking certain public-key certificates. This could be achieved by storing the hash of the certificate in a list and comparing incoming certificates with this list. Here, a certificate can simply be viewed as a signature over a message, i.e., the contents of the certificate. An adversary wanting to bypass this blocking mechanism may create a new valid signature on the certificate, thereby altering its hash value that made the certificate efficiently recognizable by the filter. This is not prevented by existential unforgeability.

The security notion that bars adversaries from forging new signatures on known (message,signature)-pairs is that of *strong* unforgeability, or SUF-CMA security, which is closely related to the concept of *malleability*. Malleable signatures retain their validity even if they are slightly changed, for example, by some bits being flipped. Obviously such signature schemes cannot hope to achieve strong unforgeability.

As mentioned earlier, Ed25519-Original without the check of $S \in \{0, \ldots, L-1\}$ during signature verification is not strongly unforgeable as any $S' \leftarrow S + mL$ with integer m also satisfies the verification equation.

For Ed25519-IETF and Ed25519-LibS, this is avoided by additionally requiring that the decoded S already be reduced modulo L, leading to the rejection of values $S' \leftarrow S + mL$ during signature verification. The property that results from this additional check on the $\mathcal{CID}$ level is that of (computationally) unique responses of the identification protocol. Recall that this property guarantees that for a given commitment com and ch in the interaction $\mathsf{P} \leftrightarrows \mathsf{V}$, there exists (at most) one response rsp such that $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ is an accepting conversation (or a second response is only possible to find with probability at most ϵ_{cur}).

For the identification protocol underlying Ed25519-IETF, we in fact even have $\epsilon_{\mathsf{cur}} = 0$, i.e., for all $(\underline{A}, k) \xleftarrow{\$} \mathcal{CID}.\mathsf{KGen}$, $(R = rB, s) \leftarrow \mathsf{P}_1(k; r)$ and $\mathsf{ch} \xleftarrow{\$} \{0,1\}^{2b}$, there exists only at (most) one valid response S. This confirms the RFC's argumentation that "Ed25519 [...] signatures are not malleable due to the verification check that decoded S is smaller than L" [7]. On an Ed25519-IETF signature level this then means that given a (message,signature)-pair $(m, (\underline{R}, \underline{S}))$ there exists no second signature $(\underline{R}, \underline{S}')$ with $\underline{S}' \neq \underline{S}$ as we will show in the next theorem. Furthermore, we will argue that there also cannot exist a second valid signature $(\underline{R}', \underline{S}')$ with $\underline{R}' \neq \underline{R}$ with $\underline{S}'$ different or equal to $\underline{S}$.

Theorem 4 (Ed25519-IETF is SUF-CMA secure)**.** *Let* Ed25519-IETF *be the* $(t, \epsilon, Q_S, Q_{\mathsf{H}})$-EUF-CMA *secure signature scheme derived by applying the Fiat-Shamir transform to the identification protocol* $\mathcal{CID}$ *given in Fig. 5 with the check in Line 17. Then* Ed25519-IETF *is* $(t', \epsilon', Q'_S, Q'_{\mathsf{H}})$-SUF-CMA *secure with* $t \approx t'$ *and* $\epsilon' \leq \epsilon$.

Proof. We recall that the games of EUF-CMA and SUF-CMA only differ in the winning condition for the adversary $\mathcal{A}$: EUF-CMA forbids that the adversary has queried the signing oracle $\mathcal{O}_{\mathsf{Sign}}$ on the message $m^\star$ for which it outputs the forgery, whereas SUF-CMA allows this and only requests that the signature forgery $\sigma^\star$ differs from the signature σ that was output by $\mathcal{O}_{\mathsf{Sign}}(m^\star)$.

We therefore focus on the case that the adversary $\mathcal{A}$ on input an Ed25519-IETF public key $\underline{A}$ outputs a valid strong forgery $(m^\star, \sigma^\star = (\underline{R}^\star, \underline{S}^\star))$ with probability ϵ' such that there exists an entry $(m^\star, \sigma' = (\underline{R}', \underline{S}')) \in \mathcal{L}_{\mathsf{Sign}}$ of recorded $\mathcal{O}_{\mathsf{Sign}}$ queries of $\mathcal{A}$, such that $\sigma^\star \neq \sigma'$. Since encodings are deterministic and unique [2], we omit them in the following discussion. Naturally, there are two ways in which $\sigma^\star = (\underline{R}^\star, \underline{S}^\star) \neq \sigma' = (\underline{R}', \underline{S}')$ for the same message $m^\star$:

Case 1 $R^\star \neq R'$**:** In this case, we can immediately build a reduction $\mathcal{B}$ against the EUF-CMA security of Ed25519-IETF. The reduction $\mathcal{B}$ gets as input a public key $\underline{A}$ and invokes the SUF-CMA adversary $\mathcal{A}(\underline{A})$. For any of the maximal Q'_S signing queries of $\mathcal{A}$ on message m, $\mathcal{B}$ simply uses the strategy of the simulator Sim (cf. Fig. 6) to obtain valid conversations $(\underline{R}, \mathsf{ch}, \underline{S})$ and patches the random oracle H to return ch on input $(\underline{R}, \underline{A}, m)$. As before, this programming ensures that the response $(\underline{R}, \underline{S})$ to $\mathcal{A}$ is a valid signature from $\mathcal{A}$'s point of view. The at most Q'_{H} random oracle queries of $\mathcal{A}$ are simulated by $\mathcal{B}$ relaying queries to the "real" random oracle H on any inputs that had not been patched by a signature query, the latter are answered consistently with the patching. Note that to run the simulation of $\mathcal{A}$, $\mathcal{B}$ has made no signing query to its own $\mathcal{O}_{\mathsf{Sign}}$. Thus, once $\mathcal{A}$ outputs its strong forgery $(m^\star, \sigma^\star)$, $\mathcal{B}$ can immediately output the same pair as its existential forgery. Note that since $R^\star \neq R'$, H has not been patched by $\mathcal{B}$ on $(\underline{R}^\star, \underline{A}, m^\star)$.

Case 2 $S^\star \neq S'$**:** This leaves the possibility that $R^\star = R'$, but $S^\star \neq S'$, i.e., the strongly forged signature is of the form $(\underline{R}', \underline{S}^\star)$. We will argue that this also is not possible, as this contradicts the uniqueness of the underlying identification protocol: For $\mathcal{CID}$ it holds that there is only (at most) one valid response $\underline{S}$ for all $(\underline{A}, k) \xleftarrow{\$} \mathcal{CID}.\mathsf{KGen}$, $(R = rB, s) \leftarrow \mathsf{P}_1(k; r)$ and $\mathsf{ch} \xleftarrow{\$} \{0,1\}^{2b}$. Assume otherwise, i.e., there exist $S \neq S'$ such that both $(\underline{R}, \mathsf{ch}, \underline{S})$ and $(\underline{R}, \mathsf{ch}, \underline{S}')$ are valid conversations wrt. the public key $\underline{A}$. To pass verification via V_2 it must hold that $S, S' \in \{0, ..., L-1\}$ and furthermore $8SB = 8R + 8\mathsf{ch}A$ and $8S'B = 8R + 8\mathsf{ch}A$, or, equivalently, $8SB = 8S'B$, contradicting the assumption that $S \neq S'$. Since verification of an Ed25519-IETF signature $(\underline{R}, \underline{S})$ on message m for public key $\underline{A}$ is just executing the verifier V_2 on input $(\underline{A}, \underline{R}, \mathsf{H}(\underline{R}, \underline{A}, m^\star), \underline{S})$ it is clear by the same argument that there cannot be a strong forgery with $S^\star \neq S'$.

To conclude, since the probability of computing non-unique responses in $\mathcal{CID}$ is 0 and the signature scheme does not admit existential forgeries with non-negligible probability, the probability of an adversary $\mathcal{A}$ succeeding against the strong unforgeability is also negligible. □

5.3 Multi-User Security

We recall that a flaw in the tight reduction from multi-user security of signatures to the single-user case in [30] was exposed by Bernstein [31], who then was able to give an alternative tight reduction from the multi-user security of *key-prefixed* Schnorr to the single-user security of standard Schnorr. This result was taken as a justification for the much-debated employed key prefixing in Ed25519 signatures. Shortly after

[2] Note that there is a very small probability that there exist two different encodings $\underline{R}_1, \underline{R}_2$ such that they decode to the same element R. This is due to the fact that elements in $E(\mathbb{F}_q)$ are encoded as b bit strings with a $(b-1)$-bit encoding for the y coordinate, plus one bit for the sign of x. Thus the entire valid encoding space for the y coordinate encompasses integers from 0 to $2^{b-1} - 1 = 2^{255} - 1$, whereas $\mathbb{F}_q$ contains only the integers from 0 to $2^{255} - 20$. Nevertheless, Case 1 in the reduction also captures this.

Clamping

the result by Bernstein, Kiltz et al. [28] were able to provide a tight reduction in the random oracle model for general Fiat-Shamir signatures, assuming the property of random self-reducibility of the underlying identification protocol, further fueling the debate (though at this time the IETF standardisation of Ed25519-IETF had already been completed). Interestingly, when trying to apply either of the above results to Ed25519 signatures specifically, several peculiarities arise. The result by Bernstein [31] is transferable to Ed25519 signatures, but loses tightness. As explained in [31, Sec. 5.3] this is due to the clamping of secret keys in Ed25519 which yields an additional failure case in the reduction. The more general result by Kiltz et al. [28] on the other hand is not applicable at all, although Ed25519 is a Fiat-Shamir transformed signature scheme. This is precisely due to the key prefixing as this prohibits the achievement of the necessary random self-reducibility property. Consequently, only the non-tight bounds in [31, Sec. 5.3] apply to Ed25519.

5.4 Key Substitution Attacks

Key Substitution Attacks (KSA) were first introduced by Blake-Wilson and Menezes [60] and later formalized by Smart and Menezes [32]. Informally, KSA cover the scenario where an adversary learns one or more (message,signature)-pairs for a given public key, and wishes to find a different public key and message such that one of the valid signatures verify under the adversary's new public key. Maliciously generated public keys fall outside the traditional notions for signature security such as existential unforgeability. However, these attacks have practical consequences in real-world contexts: examples include an attack on the popular Let's Encrypt Certificate Issuance protocol that allowed an attacker to impersonate any website, compromise of confidentiality in the WS-Security Standard, and attacks on the well known Station to Station protocol [4].

Comparatively few publications have investigated key substitution attacks and how they apply to different signature schemes. Consequently, many signature schemes are vulnerable. For example, [32] described a KSA on the Gennaro-Halevi-Rabin signature scheme [61] and the standard-model secure scheme by Boneh and Boyen [62] also proved to be vulnerable [63]. While these schemes are more of an academic interest, [60], [63] also showed that under certain conditions the widely-deployed RSA, DSA, and ECDSA signatures are insecure against KSA adversaries. Menezes and Smart also highlighted the relevance of key-substitution [32] to the multi-user setting.

In [19], a practical scheme for email authentication is proposed that requires the underlying signature scheme to be resistant to key substitution attacks. In this paper, it is stated that Schnorr signatures in prime order groups achieve SUF-CMA and resistance to key substitution attacks. The paper goes on to claim these results transfer to a variant of Ed25519 without key prefixing. We have already seen that this is not true in general and we also point out that it is not correct for their modified form of Ed25519. In particular, absent key prefixing, an adversary can submit a mangled public key lying outside the prime order group which is a distinct bitstring from the 'honest' signature yet passes the verification checks. Later in this section we consider Ed25519 with key-prefixing and find the opposite result, that this attack is provably prevented.

Exclusive ownership!

In the following, we investigate the resistance of Ed25519 against various *exclusive ownership* definitions from [5] which rule out multiple key substitution attacks. Theorem 5 shows that Ed25519 achieves this stronger version, cf. Definition 11, where the adversary is allowed to adaptively query the signing oracle to learn (message,signature)-pairs of its choice and may choose which signature to attack. Furthermore, we show in Theorem 6 that Ed25519 has so-called *message-bound signatures* (cf. Definition 12), i.e., that there exist no two distinct messages for which the same signature would verify with respect to a given (potentially maliciously generated) public key. Lastly, Theorem 7 shows that if small order elements are rejected, even *malicious* strong universal exclusive ownership guarantees are provided. We give the proofs for Theorems 5, 6, and 7 in Appendix B. Firstly, we find that an adversary cannot substitute an alternative public key to verify against an honest party's signature in any of the Ed25519 variants we have discussed.

Definition 11 (Strong Universal Exclusive Ownership). *Let* $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ *be a signature scheme. Consider the security games* $\mathrm{G}^{\mathsf{S\text{-}UEO}}_{\mathcal{S},\mathcal{A}}$ *as defined in Fig. 7. We say that a signature scheme* $\mathcal{S}$ *is* (t, ϵ, Q_S)-S-UEO*-secure or* achieves strong universal exclusive ownership *if for any adversary* $\mathcal{A}$ *making at most* Q_S *queries to the signing oracle, the probability* $\Pr\left[\mathrm{G}^{\mathsf{S\text{-}UEO}}_{\mathcal{S},\mathcal{A}}(\mathsf{pp}) = 1\right] \leq \epsilon$.

Theorem 5 (Ed25519-Original achieves S-UEO). *Let* $\mathsf{Ed25519} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ *be as defined in Fig. 4, where* $\mathsf{H} : \{0,1\}^\star \to \{0,1\}^{2b}$ *is modelled as a random oracle. Then* Ed25519 *is* $(\epsilon, Q_S, Q_\mathsf{H})$-S-UEO *secure,* $\epsilon \leq 2 \cdot Q_\mathsf{H} \cdot \lceil \frac{2^{2b}}{L} \rceil \cdot 2^{-2b}$, *where* Q_S *and* Q_H *are the maximum numbers of queries to* $\mathcal{O}_{\mathsf{Sign}}$ *and* H.

$\mathrm{G}^{\mathsf{S\text{-}UEO}}_{\mathcal{S},\mathcal{A}}(\mathsf{pp})$:

1 $(pk, sk) \xleftarrow{\$} \mathsf{KGen}(\mathsf{pp})$
2 $((m,\sigma), pk', m') \xleftarrow{\$} \mathcal{A}^{\mathcal{O}_{\mathsf{Sign}}}(pk)$
3 **return** $[\![(m,\sigma) \in \mathcal{L}_{\mathsf{Sign}} \wedge pk \neq pk' \wedge \mathsf{Vfy}(pk', \sigma, m')]\!]$

$\mathcal{O}_{\mathsf{Sign}}(m)$:

4 $\sigma \xleftarrow{\$} \mathsf{Sign}(sk, m)$
5 $\mathcal{L}_{\mathsf{Sign}} \leftarrow \mathcal{L}_{\mathsf{Sign}} \cup \{(m,\sigma)\}$
6 **return** σ

The adversary comes up with another key which can use the same signature on another message

Figure 7: Security game S-UEO

$\mathrm{G}^{\mathsf{MBS}}_{\mathcal{S},\mathcal{A}}(\mathsf{pp})$:

1 $(pk, \sigma, m, m') \xleftarrow{\$} \mathcal{A}()$
2 **return** $[\![m \neq m' \wedge \mathsf{Vfy}(pk, \sigma, m) \wedge \mathsf{Vfy}(pk, \sigma, m')]\!]$

The adversary comes up with another message that works with the same signature and public key

Figure 8: Security game MBS

We also find that, even when the signer is dishonest, Ed25519 schemes which reject public keys and signatures with low order elements, ensure that for a particular public key, signatures can only verify under a single message. However, if low order elements are accepted, an adversary can submit a low order element as their public key and any value for their signature such that $SB = R$. The resulting signature verifies under any message. This was pointed out in [1] but deemed unproblematic.

Definition 12 (Message Bound Signatures)**.** *Let* $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ *be a signature scheme. Consider the security games* $\mathrm{G}^{\mathsf{MBS}}_{\mathcal{S},\mathcal{A}}$ *as defined in Fig. 8. We say that a signature scheme* $\mathcal{S}$ *is* (ϵ)-MBS*-secure or* achieves message bound signatures *if for any adversary* $\mathcal{A}$ *the probability:* $\Pr\left[\mathrm{G}^{\mathsf{MBS}}_{\mathcal{S},\mathcal{A}}(\mathsf{pp}) = 1\right] \leq \epsilon$.

Theorem 6 (Ed25519-LibS achieves MBS)**.** *Let* $\mathsf{Ed25519} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ *be as defined in Fig. 4 and the hash function* $H : \{0,1\}^* \to \{0,1\}^{2b}$ *is a random oracle. If the small subgroup elements are rejected, then* $\mathcal{S}$ *is* (ϵ')-MBS*-secure with* $\epsilon' \leq \lceil \frac{2^{2b}}{L} \rceil \cdot 2^{-2b} \cdot (Q_H + 2)^2$*, where* Q_H *is the maximal number of queries to the random oracle.*

Strong Universal Exclusive Ownership

We now consider a stronger variant of S-UEO where the adversary collaborates or compromises with the signer, in order to generate a signature valid under two distinct public keys. Provided small subgroup elements are rejected, this property also holds. However, there is a straightforward attack if they are accepted where the adversary chooses its public key to be two distinct low order elements.

Definition 13 (Malicious Strong Universal Exclusive Ownership)**.** *Let* $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ *be a signature scheme. Consider the security games* $\mathrm{G}^{\mathsf{M\text{-}S\text{-}UEO}}_{\mathcal{S},\mathcal{A}}$ *as defined in Fig. 9. We say that a signature scheme* $\mathcal{S}$ *is* (ϵ)-M-S-UEO*-secure or* malicious strong universal exclusive ownership *if for any adversary* $\mathcal{A}$ *the probability:* $\Pr\left[\mathrm{G}^{\mathsf{M\text{-}S\text{-}UEO}}_{\mathcal{S},\mathcal{A}}(\mathsf{pp}) = 1\right] \leq \epsilon$.

Theorem 7 (Ed25519-LibS achieves M-S-UEO)**.** *Let* $\mathcal{S} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ *be as defined in Fig. 4, with the* Ed25519-LibS *variant and the hash function* $H : \{0,1\}^* \to \{0,1\}^{2b}$ *is a random oracle. Then* $\mathcal{S}$ *is* (ϵ')-M-S-UEO*-secure with* $\epsilon' \leq \frac{\lceil 2^{2b}/L \rceil}{2^{2b}} \cdot {Q_h}^2$.

6 Conclusion

We proved that Ed25519 achieves its goal of existential unforgeability (EUF-CMA), as is assumed by many published works. While Ed25519 seems similar to Fiat-Shamir applied to the Schnorr identification scheme,

$\mathrm{G}^{\mathsf{M\text{-}S\text{-}UEO}}_{\mathcal{S},\mathcal{A}}(\mathsf{pp})$:

1 $(pk, pk', \sigma, m, m') \xleftarrow{\$} \mathcal{A}()$
2 **return** $[\![pk \neq pk' \wedge \mathsf{Vfy}(pk, \sigma, m) \wedge \mathsf{Vfy}(pk', \sigma, m')]\!]$

The adversary controls both public keys, two messages, and one signature.

Figure 9: Security game M-S-UEO

the devil is in the detail. We took into account the non-prime order group, the clamping of private scalars and many other details.

Moreover, we also proved that Ed25519-IETF achieves SUF-CMA. We proved that all Ed25519 schemes resilient against key substitution attacks, however, we also showed that rejecting small order elements does yield additional properties, enabling Ed25519-LibS to achieve an even stronger form of key substitution resilience as well as message bound security.

Our results, summarized in Table 2 on page 10, thereby provide not only theoretical foundations, but also meaningful insights for choosing among the variants.

Acknowledgements: We thank the anonymous reviewers as well as Mihir Bellare, Steven Galbraith, and Eike Kiltz for their helpful comments and discussions.

References

[1] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B. Yang, "High-speed high-security signatures," in *CHES*, ser. Lecture Notes in Computer Science, vol. 6917, Springer, 2011.

[2] S. Goldwasser, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM Journal on Computing*, vol. 17, no. 2, 1988.

[3] E. Klitzke. (2017). "Bitcoin transaction malleability." https://eklitzke.org/bitcoin-transaction-malleability, (Accessed June 1, 2020).

[4] D. Jackson, C. Cremers, K. Cohn-Gordon, and R. Sasse, "Seems legit: Automated analysis of subtle attacks on protocols that use signatures," in *ACM Conference on Computer and Communications Security*, ACM, 2019.

[5] T. Pornin and J. P. Stern, "Digital signatures do not guarantee exclusive ownership," in *Applied Cryptography and Network Security*, J. Ioannidis, A. Keromytis, and M. Yung, Eds., Springer, 2005.

[6] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B. Yang, "High-speed high-security signatures," *J. Cryptographic Engineering*, vol. 2, no. 2, 2012.

[7] S. Josefsson and I. Liusvaara. (2017). "RFC 8032: Edwards-Curve Digital Signature Algorithm (EdDSA)." https://tools.ietf.org/html/rfc8032 (Accessed March 9th, 2020).

[8] *Jedisct1/libsodium*, https://github.com/jedisct1/libsodium (Accessed May 31th, 2020).

[9] D. Hopwood, S. Bowe, T. Hornby, and N. Wilcox. (2020). "ZCash Protocol Specification Version 2020.1.14." https://github.com/zcash/zips/blob/master/protocol/protocol.pdf (Accessed September 15, 2020).

[10] National Institute of Standards and Technology. (2019). "Digital Signature Standard (DSS)." https://csrc.nist.gov/publications/detail/fips/186/5/draft (Accessed May 31th, 2020).

[11] National Institute of Standards and Technology, "The Digital Signature Standard," *Communications of the ACM*, vol. 35, no. 7, 1992.

[12] National Institute of Standards and Technology. (2019). "Request for Comments on FIPS 186-5 and SP 800-186." https://www.federalregister.gov/documents/2019/10/31/2019-23742/request-for-comments-on-fips-186-5-and-sp-800-186, (Accessed June 1, 2020).

[13] E. Rescorla. (2018). "RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3." https://tools.ietf.org/html/rfc8446 (Accessed May 29th, 2020).

[14] K. Bhargavan, B. Blanchet, and N. Kobeissi, "Verified models and reference implementations for the TLS 1.3 standard candidate," in *2017 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2017.

[15] B. Dowling, M. Fischlin, F. Günther, and D. Stebila, "A cryptographic analysis of the TLS 1.3 handshake protocol candidates," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 2015.

[16] M. Kohlweiss, U. Maurer, C. Onete, B. Tackmann, and D. Venturi, "(De-) constructing TLS 1.3," in *International Conference on Cryptology in India*, Springer, 2015.

[17] K. Bhargavan, C. Brzuska, C. Fournet, M. Green, M. Kohlweiss, and S. Zanella-Béguelin, "Downgrade resilience in key-exchange protocols," in *2016 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2016.

[18] F. Bergsma, B. Dowling, F. Kohlar, J. Schwenk, and D. Stebila, "Multi-ciphersuite security of the Secure Shell (SSH) protocol," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 2014.

[19] F. Günther and B. Poettering, "Linkable message tagging: Solving the key distribution problem of signature schemes," in *Information Security and Privacy*, E. Foo and D. Stebila, Eds., Springer, 2015.

[20] C.-P. Schnorr, "Efficient identification and signatures for smart cards," in *Conference on the Theory and Application of Cryptology*, Springer, 1989.

[21] C.-P. Schnorr, "Efficient signature generation by smart cards," *J. Cryptol.*, vol. 4, no. 3, Jan. 1991.

[22] D. J. Bernstein, S. Josefsson, T. Lange, P. Schwabe, and B.-Y. Yang, "EdDSA for more curves," *Cryptology ePrint Archive*, 2015.

[23] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Advances in Cryptology — CRYPTO' 86*, A. M. Odlyzko, Ed., Springer, 1987.

[24] D. Pointcheval and J. Stern, "Security proofs for signature schemes," in *Advances in Cryptology — EUROCRYPT '96*, U. Maurer, Ed., Springer, 1996.

[25] D. Pointcheval and J. Stern, "Security Arguments for Digital Signatures and Blind Signatures," *Journal of Cryptology*, vol. 13, no. 3, Jun. 2000.

[26] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *Proceedings of the 1st ACM Conference on Computer and Communications Security*, ser. CCS '93, Fairfax, Virginia, USA: ACM, 1993.

[27] M. Abdalla, J. H. An, M. Bellare, and C. Namprempre, "From Identification to Signatures via the Fiat-Shamir Transform: Minimizing Assumptions for Security and Forward-Security," in *Advances in Cryptology — EUROCRYPT 2002*, L. R. Knudsen, Ed., Springer, 2002.

[28] E. Kiltz, D. Masny, and J. Pan, "Optimal security proofs for signatures from identification schemes," in *Advances in Cryptology – CRYPTO 2016*, M. Robshaw and J. Katz, Eds., Springer, 2016.

[29] E. Kiltz, D. Masny, and J. Pan, *Optimal security proofs for signatures from identification schemes*, Cryptology ePrint Archive, Report 2016/191, https://eprint.iacr.org/2016/191, 2016.

[30] S. Galbraith, J. Malone-Lee, and N. Smart, "Public key signatures in the multi-user setting," *Information Processing Letters*, vol. 83, no. 5, 2002.

[31] D. J. Bernstein, "Multi-user Schnorr security, revisited," *IACR Cryptology ePrint Archive*, vol. 2015, 2015.

[32] N. Smart and A Menezes, "Security of signature schemes in a multi-user setting," *Designs, Codes and Cryptography*, vol. 33, Aug. 2004.

[33] S. Moonesamy. (2015). "RFC 7479: Using Ed25519 in SSHFP Resource Records." https://tools.ietf.org/html/rfc7479 (Accessed May 29th, 2020).

[34] T. Perrin. (2016). "The XEdDSA and VXEdDSA Signature Schemes." https://signal.org/docs/specifications/xeddsa/ (Accessed March 9th, 2020).

[35] N. Kobeissi, K. Bhargavan, and B. Blanchet, "Automated verification for secure messaging protocols and their implementations: A symbolic and computational approach," in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, IEEE, 2017.

[36] F. Bergsma, B. Dowling, F. Kohlar, J. Schwenk, and D. Stebila, *Multi-ciphersuite security of the secure shell (ssh) protocol*, Cryptology ePrint Archive, Report 2013/813, https://eprint.iacr.org/2013/813, 2013.

[37] OpenBSD. (2013). "Ed25519.c." https://anongit.mindrot.org/openssh.git/tree/ed25519.c, (Accessed June 1, 2020).

[38] J. Katz and Y. Lindell, *Introduction to Modern Cryptography (Second Edition)*. Chapman & Hall CRC, 2014.

[39] L. C. Guillou and J.-J. Quisquater, "A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory," in *Advances in Cryptology — EUROCRYPT '88*, Springer, 1988.

[40] T. Okamoto, "Provably secure and practical identification schemes and corresponding signature schemes," in *Advances in Cryptology — CRYPTO' 92*, E. F. Brickell, Ed., Springer, 1993.

[41] M. Backendal, M. Bellare, J. Sorrell, and J. Sun, "The Fiat-Shamir Zoo: Relating the Security of Different Signature Variants," in *Secure IT Systems - 23rd Nordic Conference, NordSec 2018,Proceedings*, N. Gruschka, Ed., ser. LNCS, vol. 11252, Springer, 2018.

[42] D. Hankerson and A. Menezes, "Elliptic curve cryptography," in *Encyclopedia of Cryptography and Security*, H. C. A. van Tilborg and S. Jajodia, Eds. Springer, 2011.

[43] H. Cohen, G. Frey, R. Avanzi, C. Doche, T. Lange, K. Nguyen, and F. Vercauteren, Eds., *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. Chapman and Hall/CRC, 2005.

[44] D. J. Bernstein, P. Birkner, M. Joye, T. Lange, and C. Peters, "Twisted Edwards Curves," in *Progress in Cryptology – AFRICACRYPT 2008*, S. Vaudenay, Ed., Springer, 2008.

[45] D. J. Bernstein, "Curve25519: New Diffie-Hellman speed records," in *Public Key Cryptography - PKC 2006*, M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, Eds., Springer, 2006.

[46] L. Chen, D. Moody, A. Regenscheid, and K. Randall, "Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters," National Institute of Standards and Technology, Tech. Rep. Federal Information Processing Standard (FIPS) 186-5 (Draft), 2019.

[47] *Ristretto - The Ristretto Group*, https://ristretto.group/ristretto.html (Accessed May 29th, 2020).

[48] J. Grigg, G. Tankersley, H. de Valence, I. Lovecruft, and F. Valsorda, *The ristretto255 Group*, https://tools.ietf.org/html/draft-irtf-cfrg-ristretto255-00 (Accessed May 29th, 2020).

[49] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, *Ed25519: high-speed high-security signatures*, https://ed25519.cr.yp.to/, (Accessed March 09, 2020).

[50] C. Cremers and D. Jackson, "Prime, order please! revisiting small subgroup and invalid curve attacks on protocols using diffie-hellman," in *2019 IEEE 32nd Computer Security Foundations Symposium (CSF)*, IEEE, 2019.

[51] J. W. Bos, C. Costello, P. Longa, and M. Naehrig, "Selecting elliptic curves for cryptography: An efficiency and security analysis," *Journal of Cryptographic Engineering*, vol. 6, no. 4, Nov. 2016.

[52] Modern Crypto Mail Archive, *[Curves] Ed25519 "Clamping" and Its Effect on Hierarchical Key Derivation*, https://moderncrypto.org/mail-archive/curves/2017/000874.html, (Accessed: June 04, 2020).

[53] IETF Mail Archive, *[Cfrg] Does the Curve25519 Montgomery Ladder Always Work?* https://mailarchive.ietf.org/arch/msg/cfrg/pt2bt3fGQbNF8qdEcorp-rJSJrc/, (Accessed: June 04,2020).

[54] IETF Mail Archive, *Re: [Cfrg] EC signature: next steps*, https://mailarchive.ietf.org/arch/msg/cfrg/TOWH1DSzB-PfDGK8qEXtF3iC6Vc/, (Accessed: June 04, 2020).

[55] IETF Mail Archive, *Re: [Cfrg] EC signature: next steps*, https://mailarchive.ietf.org/arch/msg/cfrg/af170b6OrLyNZUHBMOPWxcDrVRI/, (Accessed: June 04, 2020).

[56] D. F. Aranha, F. R. Novaes, A. Takahashi, M. Tibouchi, and Y. Yarom, "Ladderleak: Breaking ecdsa with less than one bit of nonce leakage," in *ACM SIGSAC Conference on Computer and Communications Security*, ACM Association for Computing Machinery, 2020.

[57] M. Bellare, B. Poettering, and D. Stebila, "From identification to signatures, tightly: A framework and generic transforms," in *Advances in Cryptology – ASIACRYPT 2016*, J. H. Cheon and T. Takagi, Eds., Springer, 2016.

[58] M. Bellare and W. Dai, *The multi-base discrete logarithm problem: Non-rewinding proofs and improved reduction tightness for identification and signatures*, Cryptology ePrint Archive, Report 2020/416, https://eprint.iacr.org/2020/416, 2020.

[59] M. Bellare and A. Palacio, "GQ and Schnorr Identification Schemes: Proofs of Security against Impersonation under Active and Concurrent Attacks," in *Advances in Cryptology — CRYPTO 2002*, M. Yung, Ed., Springer, 2002.

[60] S. Blake-Wilson and A. Menezes, "Unknown Key-Share Attacks on the Station-to-Station (STS) Protocol," in *Public Key Cryptography*, Springer, 1999.

[61] R. Gennaro, S. Halevi, and T. Rabin, "Secure hash-and-sign signatures without the random oracle," in *Advances in Cryptology — EUROCRYPT '99*, J. Stern, Ed., Springer, 1999.

[62] D. Boneh and X. Boyen, "Short signatures without random oracles," in *Advances in Cryptology - EUROCRYPT 2004*, C. Cachin and J. L. Camenisch, Eds., Springer, 2004.

[63] J.-M. Bohli, S. Röhrich, and R. Steinwandt, "Key substitution attacks revisited: Taking into account malicious signers," *International Journal of Information Security*, vol. 5, no. 1, 2006.

[64] K. Chalkias, F. Garillot, and V. Nikolaenko, *Taming the many EdDSAs*, Cryptology ePrint Archive, Report 2020/1244, https://eprint.iacr.org/2020/1244, 2020.

A Schnorr Signatures

Schnorr signatures are a prime example of the way secure signatures can be constructed via the Fiat-Shamir transform from secure identification protocols. [21]. They achieve short, efficient signatures that are provably secure in the random oracle model, assuming the hardness of the discrete logarithm problem in the underlying group. The underlying Schnorr identification scheme, as well as the resulting signature scheme when the Fiat-Shamir transform is applied, are depicted in Fig. 10. Here, $\mathsf{pp} = (\mathbb{G}, q, g)$ denote the public parameters of the scheme where $\mathbb{G}$ is a cyclic group of prime order q with generator g.

$\underline{\mathsf{KGen}(\mathsf{pp})}$:
1 $a \xleftarrow{\$} \mathbb{F}_q$
2 $A \leftarrow g^a$
3 **return** (A, a)

$\underline{\mathsf{P}_1(a)}$:
4 $r \xleftarrow{\$} \mathbb{F}_q$
5 $R \leftarrow g^r$
6 **return** (R, a)

$\underline{\mathsf{P}_2(\mathsf{ch}, a)}$:
7 $S \leftarrow (R + \mathsf{ch} \cdot a) \mod q$
8 **return** S

$\underline{\mathsf{V}_1}$:
9 $\mathsf{ch} \xleftarrow{\$} \mathbb{F}_q$
10 **return** ch

$\underline{\mathsf{V}_2(A, R, \mathsf{ch}, S)}$:
11 $R' \leftarrow g^S \cdot A^{-\mathsf{ch}}$
12 **return** $[\![R' = R]\!]$

$\underline{\mathsf{KGen}(\mathsf{pp})}$:
1 $a \xleftarrow{\$} \mathbb{F}_q$
2 $A \leftarrow g^a$
3 **return** (A, a)

$\underline{\mathsf{Sign}(a, m)}$:
4 $r \xleftarrow{\$} \mathbb{F}_q$
5 $R \leftarrow g^r$
6 $\mathsf{ch} \leftarrow \mathsf{H}(R, m)$
7 $S \leftarrow (R + \mathsf{ch} \cdot a) \mod q$
8 **return** $\sigma \leftarrow (\mathsf{ch}, S)$

$\underline{\mathsf{Vfy}(A, m, \sigma = (\mathsf{ch}, S))}$:
9 $R' \leftarrow g^S \cdot A^{-\mathsf{ch}}$
10 $\mathsf{ch}' \leftarrow \mathsf{H}(R', m)$
11 **return** $[\![\mathsf{ch}' = \mathsf{ch}]\!]$

Figure 10: Schnorr signature scheme $\mathcal{S} = \mathsf{FS}[\mathcal{CID}, \mathsf{H}] = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vfy})$ (right)) resulting from the Fiat-Shamir transform applied to the Schnorr identification protocol $\mathcal{CID} = (\mathsf{KGen}, \mathsf{P}, \mathsf{V})$ (left).

B Full Proofs

B.1 Proof for Theorem 1

Proof. Assume there exists an adversary $\mathcal{A}$ against the (t', ϵ')-IMP-KOA security of $\mathcal{CID}$. From this we then construct another adversary $\mathcal{B}$, the reduction, that can solve the $(t, \epsilon_{\mathsf{ecdlp}})$-ECDLP on $E(\mathbb{F}_q)$. Since we assume ECDLP to be hard to break with non-negligible probability $\epsilon_{\mathsf{ecdlp}}$ in any reasonable amount of time t, this then leads to a contradiction, yielding IMP-KOA security. $\mathcal{B}$ is constructed as follows: $\mathcal{B}$ gets as input the base point $B \in E(\mathbb{F}_q)$ of order L, as well as a random point $A \in \langle B \rangle$. Then, $\mathcal{B}$ runs $\mathcal{A}$ on input $\underline{A}$.

We first analyze the probability of the received value A falling into the subset of correctly distributed public keys to invoke $\mathcal{A}$. Note that the public key $\underline{A}$ in the identification protocol is computed as $A \leftarrow sB$ with an $s \in \{2^{b-2}, 2^{b-2} + 8, \ldots, 2^{b-1} - 8\}$. We claim that for any public keys $A_1 = s_1 B, A_2 = s_2 B$ with $s_1, s_2 \in \{2^{b-2}, 2^{b-2} + 8, \ldots, 2^{b-1} - 8\}$, $A_1 = A_2$ if and only if $s_1 = s_2$. Note that for above s_1, s_2, there must exist $i_1, i_2 \in \{0, ..., 2^{b-5} - 1\}$ such that $s_1 = 2^{b-2} + 8i_1$ and $s_2 = 2^{b-2} + 8i_2$. Since $L > 2^{b-5}$ is a prime, it holds that

$$\begin{aligned}
A_1 = A_2 &\Leftrightarrow s_1 B = s_2 B \\
&\Leftrightarrow s_1 = s_2 \mod L \\
&\Leftrightarrow 2^{b-2} + 8i_1 = 2^{b-2} + 8i_2 \mod L \\
&\Leftrightarrow i_1 = i_2 \mod L \\
&\Leftrightarrow i_1 = i_2 \\
&\Leftrightarrow 2^{b-2} + 8i_1 = 2^{b-2} + 8i_2 \\
&\Leftrightarrow s_1 = s_2
\end{aligned}$$

The above claim indicates that the cardinality of the set of valid public keys equals 2^{b-5}. Recall that the point $A \in \langle B \rangle$ is uniformly at random. The probability of $\underline{A}$ being a valid public key from $\mathcal{A}$'s view is therefore bounded by $\frac{2^{b-5}}{L}$. In particular, substituting the instantiation of Ed25519 for the corresponding parameters, it holds that $\frac{2^{b-5}}{L} = \frac{2^{251}}{2^{252}+27742...8493} \approx \frac{1}{2}$, which is obviously non-negligible.

Challenge: At some point $\mathcal{A}$ outputs a commitment $\underline{R}^\star$ to its challenger. $\mathcal{B}$ then chooses a random challenge $\mathsf{ch}_1 \xleftarrow{\$} \{0,1\}^{2b}$ and sends ch_1 to $\mathcal{A}$. Finally, $\mathcal{A}$ terminates with output $\underline{S}_1$.

Resetting the adversary: Then, $\mathcal{B}$ resets $\mathcal{A}$'s internal state back to the point just after which it generated $\underline{R}^\star$ and returns a newly sampled challenge value $\mathsf{ch}_2 \xleftarrow{\$} \{0,1\}^{2b}$ to $\mathcal{A}$ with $\mathsf{ch}_1 \neq \mathsf{ch}_2 \mod L$.

Finally, again, $\mathcal{A}$ will output a response $\underline{S}_2$. $\mathcal{B}$ verifies whether $(\underline{R}^\star, \mathsf{ch}_1, \underline{S}_1)$ and $(\underline{R}^\star, \mathsf{ch}_2, \underline{S}_2)$ both are accepting conversations with $\mathsf{ch}_1 \neq \mathsf{ch}_2 \pmod L$, and aborts if this condition is not satisfied.

By the so-called *Reset Lemma* [59], we know that if $\mathcal{A}$ can find an $\underline{S}_1$ such that $(\underline{R}^\star, \mathsf{ch}_1, \underline{S}_1)$ is an accepting conversation with probability ϵ', then a reset of $\mathcal{A}$ with the same random tape will output an accepting conversation $(\underline{R}^\star, \mathsf{ch}_2, \underline{S}_2)$ for $\mathsf{ch}_1 \neq \mathsf{ch}_2 \pmod L$ with probability at least $(\epsilon' - \frac{1}{L})^2$.

$\mathcal{B}$ then outputs $s = \frac{S_1 - S_2}{\mathsf{ch}_1 - \mathsf{ch}_2} \mod L$.

Assume that $(\underline{R}^\star, \mathsf{ch}_1, \underline{S}_1)$ and $(\underline{R}^\star, \mathsf{ch}_2, \underline{S}_2)$ are accepting conversations with $\mathsf{ch}_1 \neq \mathsf{ch}_2 \pmod L$. In particular, it holds that $S_i \in \{0, ..., L-1\}$ and $8S_iB = 8R^\star + 8\mathsf{ch}_iA$ for $i \in \{1, 2\}$, which implies that

$$\begin{aligned} & 8(S_1 - S_2)B = 8(\mathsf{ch}_1 - \mathsf{ch}_2)A \\ \Leftrightarrow \quad & (S_1 - S_2) \cdot (\mathsf{ch}_1 - \mathsf{ch}_2)^{-1} B = A \end{aligned}$$

Therefore, $s = \frac{S_1 - S_2}{\mathsf{ch}_1 - \mathsf{ch}_2} \mod L$ is the desired solution to the ECDLP instance (B, A). Regarding the time complexity, it holds that $t \approx 2t'$, as $\mathcal{B}$ rewound $\mathcal{A}$'s internal state once. Finally, we can deduce that the probability of $\mathcal{B}$ successfully extracting the discrete logarithm is at least $\frac{2^{b-5}}{L}(\epsilon' - \frac{1}{L})^2$. □

B.2 Proof for Lemma 1

Proof. We must show that the conversations $(\underline{R}, \mathsf{ch}, \underline{S}) \xleftarrow{\$} \mathsf{Sim}(\underline{A})$ are distributed identically to $\mathsf{Trans}[\mathsf{P}(k) \leftrightarrows \mathsf{V}(\underline{A})]$ in honest executions of $\mathcal{CID}$.

In the following let $(\mathsf{com}, \mathsf{ch}, \mathsf{rsp})$ be a valid honest execution between the prover and the verifier. It holds that com is the encoding of an element rB in the elliptic curve group with $r \xleftarrow{\$} \{0,1\}^{2b}$, $\mathsf{ch} \xleftarrow{\$} \{0,1\}^{2b}$ and rsp is the encoding of an element in $\{0, ..., L-1\}$ of the form $(r + \mathsf{ch} \cdot s) \mod L$, with s implicitly fixed by the decoding of $\underline{A} = \underline{sB}$.

Clearly, the challenges are distributed identically in both conversations. The (decoded) simulated responses $S \leftarrow \tilde{s}$ with $\tilde{s} \xleftarrow{\$} \{0,1\}^{2b}$ are also distributed identically to real responses $\mathsf{rsp} = (r + \mathsf{ch} \cdot s) \mod L$, with $r, \mathsf{ch} \xleftarrow{\$} \{0,1\}^{2b}$. The same holds for the (decoded) simulated commitments $R \leftarrow (SB - \mathsf{ch}A) \mod L = (S - \mathsf{ch} \cdot s)B \mod L$ and the real commitments com, since the latter are in the elliptic curve group of the form $\mathsf{com} \leftarrow rB$ with $r \xleftarrow{\$} \{0,1\}^{2b}$, which is equivalent to $r'B$ with $r' \leftarrow r \mod L$.

□

B.3 Proof for Theorem 2

Proof. We have shown in Lemma 1 that $\mathcal{CID}$ is ϵ_{zk}-HVZK with $\epsilon_{\mathsf{zk}} = 0$. Since $\mathsf{Sim}(pk)$ uses public information only, any resulting conversations could also have been computed by $\mathcal{A}$ itself. $\mathcal{A}$ therefore learns nothing from the interaction of $\mathsf{P} \leftrightarrows \mathsf{V}$ via $\mathcal{O}_{\mathsf{Trans}}$ (replaced by Sim). Thus, for canonical identification protocols that are HVZK, IMP-PA security is equivalent to IMP-KOA security and we have ave $\epsilon' \leq \epsilon$ and $t \approx t'$ plus the running time of the Sim at most Q_T times. Since t is dominated by t', we simply write $t \approx t'$. □

B.4 Proof for Theorem 5

Proof. Assume there exists an adversary $\mathcal{A}$ that can break the (ϵ, Q_S)-S-UEO security of Ed25519. This means that for an honestly generated key pair $(\underline{A}, k) \xleftarrow{\$} \mathsf{KGen}$, given the public key $\underline{A}$, $\mathcal{A}$ can output $\big((m, \sigma = (\underline{R}, \underline{S})), \underline{A}', m'\big)$ such that:

1. $\sigma \leftarrow \mathsf{Sign}(k, m)$ for one of the Q_S signing queries of $\mathcal{A}$. In particular, we have $8SB = 8R + 8\mathsf{H}(\underline{R}, \underline{A}, m)A$.
2. $\underline{A'} \neq \underline{A}$, which means that $A' \neq A$.
3. Verification $\mathsf{Vfy}(\underline{A'}, \sigma, m')$ holds, i.e., $8SB = 8R + 8\mathsf{H}(\underline{R}, \underline{A'}, m')A'$.

Let $\mathsf{ch} \leftarrow \mathsf{H}(\underline{R}, \underline{A}, m)$ and $\mathsf{ch}' \leftarrow \mathsf{H}(\underline{R}, \underline{A'}, m')$. Observing property 2, properties 1) and 3) can only hold simultaneously if and only if one of the following (distinct) cases arises:

Case 1: It holds that $SB = R$ in property 1). Then $\mathcal{A}$ can simply output a low order point $\underline{A'}$, i.e., with $|A'| \leq L$ and $m' = m$, which causes property 3) to also collapse to $SB = R$, irrespective of the value ch'. This can only happen in property 1) if $\mathsf{ch} = 0 \pmod L$ or $\mathsf{ch} = s^{-1} \pmod L$ for $A = sB$. But since H is a random oracle, this happens only with probability at most $2 \cdot Q_H \cdot \left(\lceil \frac{2^{2b}}{L} \rceil \cdot 2^{-2b}\right)$. So in the following we have $SB \neq R$.

Case 2 $\mathcal{A}$ can guess $A' \neq A$ with $|A'| \geq L$ and m' such that $\mathsf{ch}'A' = \mathsf{ch}A$. But, again, since H is a random oracle, the probability of this succeeding, accounting for the adversary's ability to repeat the process, is bounded by $Q_H \cdot \lceil \frac{2^{2b}}{L} \rceil \cdot 2^{-2b}$.

□

B.5 Proof for Theorem 6

Proof. Let $\mathcal{A}$ denote an adversary against (ϵ')-MBS security of Ed25519. We then give the concrete upper bound of ϵ' in the random oracle model. Assume that $\mathcal{A}$ terminates with $(pk = \underline{A}, \sigma = (\underline{R}, \underline{S}), m, m')$ and wins the MBS experiment in Fig. 8. Then, it holds that $m \neq m'$, $8SB = 8R + 8\mathsf{H}(\underline{R}, \underline{A}, m)A$, and $8SB = 8R + 8\mathsf{H}(\underline{R}, \underline{A}, m')A$, which further implies that $8\mathsf{H}(\underline{R}, \underline{A}, m)A = 8\mathsf{H}(\underline{R}, \underline{A}, m')A$. Note that A is not a small subgroup element, it must hold that

$$\mathsf{H}(\underline{R}, \underline{A}, m) = \mathsf{H}(\underline{R}, \underline{A}, m') \mod L, \quad m \neq m'. \tag{1}$$

Obviously, we have $\epsilon' \leq \Pr[\text{Eq.}(1) \text{ holds}]$. Note that the random oracle in the MBS experiment will evaluate at most $(Q_H + 2)$ different inputs, where at most Q_H ones are queried by $\mathcal{A}$ and two are queried by the challenger for final verifications. Moreover, Eq. (1) holds only if there exists two outputs of the random oracle on different inputs such that the outputs are congruent modulo L, which occurs with probability from above bounded by $\lceil \frac{2^{2b}}{L} \rceil \cdot 2^{-2b} \cdot (Q_H + 2)^2$. Hence, it holds that $\epsilon' \leq \lceil \frac{2^{2b}}{L} \rceil \cdot 2^{-2b} \cdot (Q_H + 2)^2$. □

B.6 Proof for Theorem 7

Proof. It follows from the verification equation that:

$$8\mathsf{H}(R, pk', m')pk' = 8\mathsf{H}(R, pk, m)pk$$

It then follows that from the rejection of small subgroup elements that:

$$\mathsf{H}(R, pk', m')a' = \mathsf{H}(R, pk, m)a \mod L \tag{2}$$

As a is in the range $1, \ldots, L$ and thus coprime to L it follows that

$$\mathsf{H}(R, pk', m')a'(a)^{-1} = \mathsf{H}(R, pk, m) \mod L \tag{3}$$

We fix a, m', a', then for a particular m, $\mathsf{H}(R, pk, m)$ is in the range $0, \ldots, 2^{2b}$ of which there are at most $\lceil 2^{2b}/L \rceil$ values such that the equation holds. Consequently a given guess has probability $\frac{\lceil 2^{2b}/L \rceil}{2^{2b}}$ of fulfilling the equation. However, the adversary can also vary m' and consequently perform a collision attack. Notice that the adversary can make up to Q_h queries and consequently the overall probability of success is bounded above by $\frac{\lceil 2^{2b}/L \rceil}{2^{2b}} \cdot {Q_h}^2$ □

C Summary of changes

- **v1.0 – July 2020**: Initial release.
- **v1.0.1 – July 2020**: Minor fixes.
- **v1.0.2 – October 2020**: Various minor fixes and presentation improvements, notably better clarifying properties and checks in Table 1 and Figure 4.

  Based on feedback from Steven Galbraith, slightly improved the bound of Theorem 1, while including a previously missed factor 2 for the high bit.
- **v1.0.3 – October 2020**: Based on feedback from the authors of [64], fixed a typographical mistake in Table 2 and clarified the optional step of multiplying by the cofactor when verifying signatures in §4.2.1.