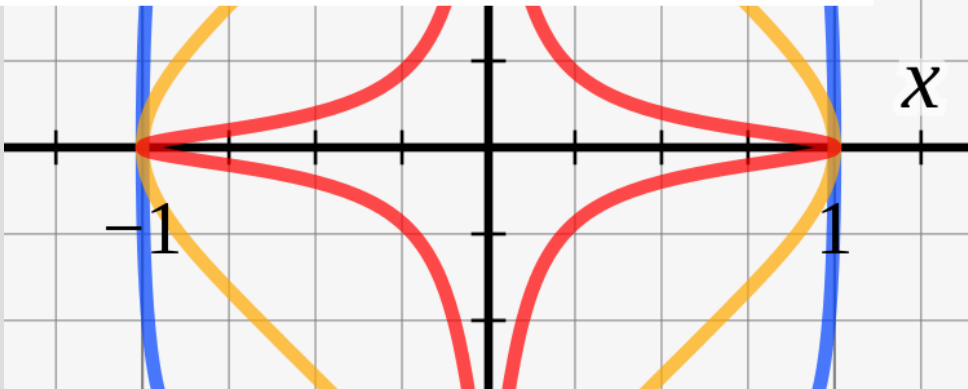


A formal Security Analysis of the EdDSA Signature Scheme

2023

Fakultät für Informatik

Aaron Kaiser



- 1 Overview
- 2 Motivation
- 3 Related work
- 4 Preliminaries
- 5 The EdDSA signature scheme
- 6 Singe- and multi-user Proofs for EdDSA
- 7 GGM proofs of the underlying assumptions
- 8 Concrete security

Overview

What is this thesis about?

- 1 Providing a tight security proof for the EdDSA signature scheme
- 2 Showing the security level of concrete instantiations of EdDSA

Results of this thesis:

- 1 EdDSA is tightly secure under Ed-DLog assumption in the single-user setting
- 2 EdDSA is tightly secure under the N-Ed-DLog-Reveal assumption in the multi-user setting
- 3 Ed25519 provides 125/124 bits of security in the single/multi-user setting
- 4 Ed448 provides 221/220 bits of security in the single/multi-user setting

Results of this thesis:

- 1 EdDSA is tightly secure under Ed-DLog assumption in the single-user setting
- 2 EdDSA is tightly secure under the N-Ed-DLog-Reveal assumption in the multi-user setting
- 3 Ed25519 provides 125/124 bits of security in the single/multi-user setting
- 4 Ed448 provides 221/220 bits of security in the single/multi-user setting

EdDSA is everywhere...



Signal



SSH



No existing tight security proof since publication in 2015

- Brendel et al. 2021 [1]: First security proof for Ed25519
- Chalkias et al. 2020 [2]: Analysis of different EdDSA implementations
- Fuchsbauer et al. 2020 [3]: Tight security proof for Schnorr Signatures using AGM

Definition

A digital signature scheme $SIG = (\text{KeyGen}, \text{Sign}, \text{Verify})$ is a tuple of algorithms.

- **KeyGen**: The key generation algorithm, which upon receiving the security parameter as input outputs a matching tuple of public and private key.
- **Sign**: The signature algorithm, which upon receiving a secret key and a message, outputs a signature for that message.
- **Verify**: The verification algorithm, which upon receiving a public key, a message and a signature, outputs 1 if the signature gets accepted and 0 otherwise.

For the digital signature scheme to be correct, it is required that

$$\forall (pk, sk) \in \text{KeyGen}(par), m \in \mathcal{M}, \sigma \in \text{Sign}(sk, m) : \text{Verify}(pk, m, \sigma) = 1$$

Game N-MU-EUF-CMA

for $i \in \{1, 2, \dots, N\}$

$(pk_i, sk_i) \leftarrow \text{KeyGen}(1^\lambda)$

$(m^*, \sigma^*) \leftarrow \mathcal{A}^{\text{Sign}(\cdot, \cdot)}(pk_1, pk_2, \dots, pk_n)$

return $\exists i \in \{1, 2, \dots, N\} : \text{Verify}(pk_i, m^*, \sigma^*) \stackrel{?}{=} 1 \wedge (pk_i, m^*) \notin M$

Oracle Sign ($i \in \{1, 2, \dots, n\}, m \in \mathcal{M}$)

$\sigma \leftarrow \text{Sign}(sk_i, m)$

$M := M \cup \{(pk_i, m)\}$

return σ

Abb. N-MU-EUF-CMA Security Game

Game N-MU-SUF-CMAfor $i \in \{1, 2, \dots, N\}$ $(pk_i, sk_i) \leftarrow \text{KeyGen}(1^\lambda)$ $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\text{Sign}(\cdot, \cdot)}(pk_1, pk_2, \dots, pk_n)$ return $\exists i \in \{1, 2, \dots, N\} : \text{Verify}(pk_i, m^*, \sigma^*) \stackrel{?}{=} 1 \wedge (pk_i, m^*, \sigma^*) \notin M$ Oracle Sign ($i \in \{1, 2, \dots, n\}, m \in \mathcal{M}$) $\sigma \leftarrow \text{Sign}(sk_i, m)$ $M := M \cup \{(pk_i, m, \sigma)\}$ return σ

Abb. N-MU-SUF-CMA Security Game

Game N-MU-EUF-NMA

for $i \in \{1, 2, \dots, N\}$

$(pk_i, sk_i) \leftarrow \text{KeyGen}(1^\lambda)$

$(m^*, \sigma^*) \leftarrow \mathcal{A}(pk_1, pk_2, pk_n)$

return $\exists i \in \{1, 2, \dots, N\} : \text{Verify}(pk_i, m^*, \sigma^*) \stackrel{?}{=} 1$

Abb. N-MU-EUF-NMA Security Game

Random Oracle Model (ROM)

- Hash functions are modeled as public oracle
- Oracle behaves like a true random function
- Challenger can observe all inputs
- Challenger can program the random oracle

Algebraic Group Model (AGM)

- Adversary has to provide a representation of all group elements

Generic Group Model (GGM)

- GGM hides all group-specific representation of group elements
- Adversary works with random labels instead of actual group elements

- Paper by Bernstein et al. [4, 5]
- RFC 8032 [6]
- FIPS 186-5 [7]

The EdDSA Signature Scheme

Parameter	Description
q	An odd prime power q . EdDSA uses an elliptic curve over the finite field $\mathbb{F}_q$.
b	An integer b with $2^{b-1} > q$. The bit size of encoded points on the twisted Edwards curve.
$Enc(\cdot)$	A $(b - 1)$ -bit encoding of elements in the underlying finite field.
$H(\cdot)$	A cryptographic hash function producing $2b$ -bit output.
c	The cofactor of the twisted Edwards curve.
n	The number of bits used for the secret scalar of the public key.
a, d	The curve parameter of the twisted Edwards curve.
B	A generator point of the prime order subgroup of E .
L	The order of the prime order subgroup.
$H'(\cdot)$	A prehash function applied to the message prior to applying the Sign or Verify procedure.

Tab. Parameter of the EdDSA signature scheme

The EdDSA Signature Scheme

Parameter	Description
b	An integer b with $2^{b-1} > q$. The bit size of encoded points on the twisted Edwards curve.
$H(\cdot)$	A cryptographic hash function producing $2b$ -bit output.
c	The cofactor of the twisted Edwards curve.
n	The number of bits used for the secret scalar of the public key.
B	A generator point of the prime order subgroup of E .
L	The order of the prime order subgroup.

Tab. Parameter of the EdDSA signature scheme

The EdDSA Signature Scheme

KeyGen

$$k \leftarrow \{0, 1\}^b$$

$$(h_0, h_1, \dots, h_{2b-1}) := H(k)$$

$$s \leftarrow 2^n + \sum_{i=c}^{n-1} 2^i h_i$$

$$A := sB$$

$$\text{return } (\underline{A}, k)$$

The EdDSA Signature Scheme

Sign(k, m)

$(h_0, h_1, \dots, h_{2b-1}) := H(k)$

$s \leftarrow 2^n + \sum_{i=c}^{n-1} 2^i h_i$

$(r'_0, r'_1, \dots, r'_{2b-1}) := H(h_b | \dots | h_{2b-1} | m)$

$r := \sum_{i=0}^{2b-1} 2^i r'_i$

$R := rB$

$\text{ch} := H(\underline{R} | \underline{A} | m)$

$S := (r + \text{ch} \cdot s) \pmod{L}$

return $\sigma := (\underline{R}, S)$

The EdDSA Signature Scheme

Sign(k, m)

$$(h_0, h_1, \dots, h_{2b-1}) := H(k)$$

▷ Recover secret scalar

$$s \leftarrow 2^n + \sum_{i=c}^{n-1} 2^i h_i$$

$$(r'_0, r'_1, \dots, r'_{2b-1}) := H(h_b | \dots | h_{2b-1} | m)$$

$$r := \sum_{i=0}^{2b-1} 2^i r'_i$$

$$R := rB$$

$$\mathbf{ch} := H(\underline{R} | \underline{A} | m)$$

$$S := (r + \mathbf{ch} \cdot s) \pmod{L}$$

$$\text{return } \sigma := (\underline{R}, S)$$

The EdDSA Signature Scheme

Sign(k, m)

$$(h_0, h_1, \dots, h_{2b-1}) := H(k)$$

$$s \leftarrow 2^n + \sum_{i=c}^{n-1} 2^i h_i$$

$$(r'_0, r'_1, \dots, r'_{2b-1}) := H(h_b | \dots | h_{2b-1} | m)$$

▷ Calculate commitment

$$r := \sum_{i=0}^{2b-1} 2^i r'_i$$

$$R := rB$$

$$\text{ch} := H(\underline{R} | \underline{A} | m)$$

$$S := (r + \text{ch} \cdot s) \pmod{L}$$

$$\text{return } \sigma := (\underline{R}, S)$$

The EdDSA Signature Scheme

Sign(k, m)

$$(h_0, h_1, \dots, h_{2b-1}) := H(k)$$

$$s \leftarrow 2^n + \sum_{i=c}^{n-1} 2^i h_i$$

$$(r'_0, r'_1, \dots, r'_{2b-1}) := H(h_b | \dots | h_{2b-1} | m)$$

$$r := \sum_{i=0}^{2b-1} 2^i r'_i$$

$$R := rB$$

$$\text{ch} := H(\underline{R} | \underline{A} | m)$$

▷ Calculate challenge

$$S := (r + \text{ch} \cdot s) \pmod{L}$$

$$\text{return } \sigma := (\underline{R}, S)$$

The EdDSA Signature Scheme

Sign(k, m)

$$(h_0, h_1, \dots, h_{2b-1}) := H(k)$$

$$s \leftarrow 2^n + \sum_{i=c}^{n-1} 2^i h_i$$

$$(r'_0, r'_1, \dots, r'_{2b-1}) := H(h_b | \dots | h_{2b-1} | m)$$

$$r := \sum_{i=0}^{2b-1} 2^i r'_i$$

$$R := rB$$

$$\text{ch} := H(R | A | m)$$

$$S := (r + \text{ch} \cdot s) \pmod{L}$$

▷ Calculate response

$$\text{return } \sigma := (\underline{R}, S)$$

The EdDSA Signature Scheme

Verify($\underline{A}, \sigma := (\underline{R}, S), m$)
return $2^c SB \stackrel{?}{=} 2^c R + 2^c H(\underline{R}|\underline{A}|m)A$

The EdDSA Signature Scheme

Signature Parsing

- Strict parsing: Reject all bitstring representations of $S > L$
- Lax parsing: Allow all bitstring representations of S and work with $S \pmod{L}$

The EdDSA Signature Scheme

Encoding of Group Elements

- Decoding function ensures that point is on curve
- Multiple bitstrings might map to the same point on the curve

The EdDSA' Signature Scheme

KeyGen

$(h_0, h_1, \dots, h_{2b-1}) \leftarrow \{0, 1\}^{2b}$

$s \leftarrow 2^n + \sum_{i=c}^{n-1} 2^i h_i$

$A := sB$

return $(\underline{A}, k := (s, h_b | \dots | h_{2b-1}))$

Sign $(k := (s, h_b | \dots | h_{2b-1}), m)$

$(r'_0, r'_1, \dots, r'_{2b-1}) := RF(h_b | \dots | h_{2b-1} | m)$

$r := \sum_{i=0}^{2b-1} 2^i r'_i$

$R := rB$

$S := (r + sH(\underline{R} | \underline{A} | m)) \pmod{L}$

return $\sigma := (\underline{R}, S)$

Abb. Generic description of the algorithms KeyGen, Sign and Verify used by the EdDSA' signature scheme

Theorem

Let $\mathcal{A}$ be an adversary against SUF-CMA security of the EdDSA signature scheme. Then

$$\text{Adv}_{\text{EdDSA}', \mathcal{A}}^{\text{SUF-CMA}}(\lambda) \leq \text{Adv}_{\text{EdDSA}, \mathcal{A}}^{\text{SUF-CMA}}(\lambda) + \frac{2(q_h + 1)}{2^b}.$$

Theorem (Security of EdDSA with strict parsing in the single-user setting)

Let $\mathcal{A}$ be an adversary against the SUF-CMA security of EdDSA with strict parsing, making at most q_h hash queries and q_o oracle queries, and $\mathbb{G}$ be a group of prime order L . Then,

$$\text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{SUF-CMA}}(\lambda) \leq \text{Adv}_{E, n, c, L, \mathcal{B}}^{\text{Ed-DLog}} + \frac{2(q_h + 1)}{2^b} + \frac{q_o q_h + q_o}{2^{-\log_2(\lceil \frac{2^{2b}-1}{L} \rceil 2^{-2b})}}$$

Theorem (Security of EdDSA with lax parsing in the single-user setting)

Let $\mathcal{A}$ be an adversary against the EUF-CMA security of EdDSA with lax parsing, making at most q_h hash queries and q_o oracle queries, and $\mathbb{G}$ be a group of prime order L . Then,

$$\text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{EUF-CMA}}(\lambda) \leq \text{Adv}_{E, n, c, L, \mathcal{B}}^{\text{Ed-DLog}} + \frac{2(q_h + 1)}{2^b} + \frac{q_o q_h + q_o}{2^{-\log_2(\lceil \frac{2^{2b}-1}{L} \rceil 2^{-2b})}}$$

$$\text{Ed-DLog} \xRightarrow{\text{AGM}} \text{Ed-IDLOG} \xRightarrow{\text{ROM}} \text{EUF-NMA} \xRightarrow{\text{ROM}} \text{SUF-CMA}_{\text{EdDSA sp}}$$

$$\text{Ed-DLog} \xRightarrow{\text{AGM}} \text{Ed-IDLOG} \xRightarrow{\text{ROM}} \text{EUF-NMA} \xRightarrow{\text{ROM}} \text{EUF-CMA}_{\text{EdDSA lp}}$$

Single-User Security

EUf-NMA $\stackrel{\text{ROM}}{\Rightarrow}$ SUf-CMA_{EdDSA_{sp}}/EUf-CMA_{EdDSA_{lp}}

Theorem ([1])

Let $\mathcal{A}$ be an adversary against SUf-CMA, making at most q_h hash queries and q_o oracle queries, and let $\mathbb{G}$ be a group of prime order L . Then,

$$\text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{SUf-CMA}}(\lambda) \leq \text{Adv}_{\mathbb{G}, \mathcal{B}}^{\text{EUf-NMA}}(\lambda) + \frac{q_o q_h}{2^{-\log_2(\lceil \frac{2^{2b}-1}{L} \rceil 2^{-2b})}}.$$

Single-User Security

EUf-NMA $\stackrel{\text{ROM}}{\Rightarrow}$ SUf-CMA_{EdDSA_{sp}}

Proof Idea:

- Simulate signatures without private key
 - 1 Choose commitment and solution uniformly at random
 - 2 Calculate corresponding challenge
 - 3 Program random oracle to output that challenge for the signature
- Forward random oracle queries to challenger
- A valid signature forgery in the SUf-CMA game is also a valid forgery in the EUf-NMA game

Single-User Security

Ed-IDLOG $\stackrel{\text{ROM}}{\Rightarrow}$ EUF-NMA

Game Ed-IDLOG

$$a \leftarrow \{2^{n-1}, 2^{n-1} + 2^c, \dots, 2^n - 2^c\}$$

$$A := aB$$

$$s^* \leftarrow \mathcal{A}^{\text{Chall}(\cdot)}(A)$$

$$\text{return } \exists(R^*, \text{ch}^*) \in Q : R^* = 2^c s^* B - 2^c \text{ch}^* A$$

Oracle *Chall* ($R_i \in \mathbb{G}$)

$$\text{ch}_i \leftarrow \{0, 1\}^{2b}$$

$$Q := Q \cup \{(R_i, \text{ch}_i)\}$$

$$\text{return ch}_i$$

Abb. Ed-IDLOG

Single-User Security

Ed-IDLOG $\stackrel{\text{ROM}}{\Rightarrow}$ EUF-NMA

Theorem

Let $\mathcal{A}$ be an adversary against EUF-NMA. Then,

$$\text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{EUF-NMA}}(\lambda) = \text{Adv}_{\mathbb{G}, \mathcal{B}}^{\text{Ed-IDLOG}}(\lambda).$$

Single-User Security

Ed-IDLOG $\stackrel{\text{ROM}}{\Rightarrow}$ EUF-NMA

Proof Idea:

- Forward random oracle queries to *Chall* oracle
- A valid signature forgery provides a valid solution for Ed-IDLOG

Single-User Security

Ed-DLog $\xRightarrow{\text{AGM}}$ Ed-IDLOG

Game Ed-DLog

$a \leftarrow \{2^{n-1}, 2^{n-1} + 2^c, \dots, 2^n - 2^c\}$

$A := aB$

$a' \leftarrow \mathcal{A}(A)$

return $a \stackrel{?}{=} a'$

Abb. Ed-DLog

Single-User Security

Ed-DLog $\xRightarrow{\text{AGM}}$ Ed-IDLOG

Theorem

Let $\mathcal{A}$ be an adversary against Ed-IDLOG with $\mathbb{G}$ being a cyclic group of prime order L , making at most q_o oracle queries. Then

$$\text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{Ed-IDLOG}}(\lambda) \leq \text{Adv}_{\mathbb{G}, \mathcal{B}}^{\text{Ed-DLog}}(\lambda) + \frac{q_o}{2^{-\log_2(\lceil \frac{2^{2b}-1}{L} \rceil 2^{-2b})}}.$$

Single-User Security

Ed-DLog $\xRightarrow{\text{AGM}}$ Ed-IDLOG

Proof Idea:

- Adversary provides a valid solution: $R^* = 2^c s^* B - 2^c ch^* A$
- Adversary also provides: $R^* = r_1 B + r_2 A$
- Rewrite equations: $A = (2^c s^* - r_1)(r_2 + 2^c ch^*)^{-1} B$

Theorem (Security of EdDSA with strict parsing in the multi-user setting)

Let $\mathcal{A}$ be an adversary against the N -MU-SUF-CMA security of EdDSA with strict parsing, receiving N public keys and making at most q_h hash queries and q_o oracle queries, and $\mathbb{G}$ be a group of prime order L . Then,

$$\text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{MU-SUF-CMA}}(\lambda) \leq \text{Adv}_{E, n, c, L, \mathcal{B}}^{\text{Ed-OMDL}} + \frac{2(q_h + 1)}{2^b} + \frac{q_o q_h + q_o N}{2^{-\log_2(\lceil \frac{2^{2b}-1}{L} \rceil 2^{-2b})}}$$

Theorem (Security of EdDSA with lax parsing in the multi-user setting)

Let $\mathcal{A}$ be an adversary against the MU-EUF-CMA security of EdDSA with lax parsing, receiving N public keys and making at most q_h hash queries and q_o oracle queries, and $\mathbb{G}$ be a group of prime order L . Then,

$$\text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{MU-EUF-CMA}}(\lambda) \leq \text{Adv}_{E, n, c, L, \mathcal{B}}^{\text{Ed-OMDL}} + \frac{2(q_h + 1)}{2^b} + \frac{q_o q_h + q_o N}{2^{-\log_2(\lceil \frac{2^{2b}-1}{L} \rceil 2^{-2b})}}$$

$$\text{Ed-OMDL} \xRightarrow{\text{AGM}} \text{Ed-IDLOG} \xRightarrow{\text{ROM}} \text{MU-EUF-NMA} \xRightarrow{\text{ROM}} \text{MU-SUF-CMA}_{\text{EdDSA}_{\text{sp}}}$$
$$\text{Ed-OMDL} \xRightarrow{\text{AGM}} \text{Ed-IDLOG} \xRightarrow{\text{ROM}} \text{MU-EUF-NMA} \xRightarrow{\text{ROM}} \text{MU-EUF-CMA}_{\text{EdDSA}_{\text{lp}}}$$

Multi-User Security

Ed-OMDL $\stackrel{\text{AGM}}{\Rightarrow}$ Ed-IDLOG

Game Ed-OMDL

for $i \in \{1, 2, \dots, N\}$

$a_i \leftarrow \{2^{n-1}, 2^{n-1} + 2^c, \dots, 2^n - 2^c\}$

$A_i := a_i B$

$(a'_1, a'_2, \dots, a'_N) \leftarrow \mathcal{A}^{DL(\cdot)}(A_1, A_2, \dots, A_N)$

return $(a_1, a_2, \dots, a_N) \stackrel{?}{=} (a'_1, a'_2, \dots, a'_N)$

Oracle $DL(j \in \{1, 2, \dots, N\})$

return $\{a_i | i \in \{1, 2, \dots, N\} \setminus \{j\}\}$

▷ *max. one query*

Abb. Ed-OMDL

Proof Idea:

- Similar to single-user proof
- Query discrete logarithms of all but one challenge group element A_i
- Construct a representation $R^* = r_1 B + r_2 A_i$
- Calculate discrete logarithm of A_i

Theorem

Let n and c be positive integers. Consider a twisted Edwards curve E with a cofactor of 2^c and a generating set consisting of $(B, E_2, \dots, E_m)$. Among these, let B be the generator of the largest prime order subgroup with an order of L . Let $\mathcal{A}$ be a generic adversary making at most q_g group operations. Then,

$$\text{Adv}_{E,n,c,L,\mathcal{A}}^{\text{Ed-DLog}} \leq \frac{(q_g + 3)^2 + 1}{2^{n-1-c}}.$$

Theorem

Let n , N , c be positive integers. Consider a twisted Edwards curve E with a cofactor of 2^c and a generating set consisting of $(B, E_2, \dots, E_m)$. Among these, let B be the generator of the largest prime order subgroup with an order of L . Let $\mathcal{A}$ be a generic adversary against Ed-OMDL receiving N group elements as challenge and making at most q_g group operations queries. Then,

$$\text{Adv}_{E,n,c,L,\mathcal{A}}^{\text{Ed-OMDL}} \leq \frac{2(q_g + N + 2)^2 + 1}{2^{n-1-c}}.$$

Lemma (Schwartz-Zippel lemma [8])

Let L be a prime number and $P \in \mathbb{F}_L[X_1, \dots, X_n]$ be a non-zero polynomial of total degree $d \geq 0$ over a field $\mathbb{F}_L$. Let S be a finite subset of $\mathbb{F}_L$ and let x be selected uniformly at random from S . Then

$$\Pr[P(x) = 0] \leq \frac{d}{|S|}.$$

Game G_0 / G_1 / G_2 / G_3 / G_4

for $i \in \{1, 2, \dots, N\}$

$a_i \leftarrow \{2^{n-1}, 2^{n-1} + 2^c, \dots, 2^n - 2^c\}$

$A_i := a_i B$

$(a'_1, a'_2, \dots, a'_N) \leftarrow \mathcal{A}^{GOp(\cdot, \cdot, \cdot), DL(\cdot)}(Enc(B), Enc(E_2), \dots, Enc(E_m), Enc(A_1), \dots, Enc(A_N))$

return $(a_1, a_2, \dots, a_N) \stackrel{?}{=} (a'_1, a'_2, \dots, a'_N)$

Oracle $DL(j \in \{1, 2, \dots, N\})$

return $\{a_i | i \in \{1, 2, \dots, N\} \setminus \{j\}\}$

Oracle $GOp(x, y \in S, b \in \{0, 1\})$

return $Enc(\sum^{-1}[x] + (-1)^b \sum^{-1}[y])$

Procedure $Enc(X \in E)$

if $\sum[X] = \perp$ **then**

$\sum[X] \leftarrow \{0, 1\}^{\lceil \log_2(|E|) \rceil} \setminus S$

$S := S \cup \{\sum[X]\}$

return $\sum[X]$

Abb. G_0

Game G_0 / G_1 / G_2 / G_3 / G_4

for $i \in \{1, 2, \dots, N\}$

$a_i \leftarrow \{2^{n-1}, 2^{n-1} + 2^c, \dots, 2^n - 2^c\}$

$A_i := (a_i, 0, \dots, 0)$

$(a'_1, a'_2, \dots, a'_N) \leftarrow \mathcal{A}^{GOp(\cdot, \cdot, \cdot), DL(\cdot)}(Enc(B), Enc(E_2), \dots, Enc(E_m), Enc(A_1), \dots, Enc(A_N))$

return $(a_1, a_2, \dots, a_N) \stackrel{?}{=} (a'_1, a'_2, \dots, a'_N)$

Oracle $DL(j \in \{1, 2, \dots, N\})$

return $\{a_i | i \in \{1, 2, \dots, N\} \setminus \{j\}\}$

Oracle $GOp(x, y \in S, b \in \{0, 1\})$

return $Enc(\sum^{-1}[x] + (-1)^b \sum^{-1}[y])$

Procedure $Enc(X \in \mathbb{Z}_L \times \mathbb{Z}_{ord(E_2)} \times \dots \times \mathbb{Z}_{ord(E_n)})$

If $\sum[X] = \perp$ then

$\sum[X] \leftarrow \{0, 1\}^{\lceil \log_2(|E|) \rceil} \setminus S$

$S := S \cup \{\sum[X]\}$

return $\sum[X]$

Abb. G_1

Game G_0 / G_1 / G_2 / G_3 / G_4

for $i \in \{1, 2, \dots, N\}$

$a_i \leftarrow \{2^{n-1}, 2^{n-1} + 2^c, \dots, 2^n - 2^c\}$

$P_i := Z_i$

$A_i := (P_i, 0, \dots, 0)$

$(a'_1, a'_2, \dots, a'_N) \leftarrow \mathcal{A}^{GOp(\cdot, \cdot, \cdot), DL(\cdot)}(Enc(B), Enc(E_2), \dots, Enc(E_m), Enc(A_1), \dots, Enc(A_N))$

return $(a_1, a_2, \dots, a_N) \stackrel{?}{=} (a'_1, a'_2, \dots, a'_N)$

Oracle $DL(j \in \{1, 2, \dots, N\})$

return $\{a_i | i \in \{1, 2, \dots, N\} \setminus \{j\}\}$

Oracle $GOp(x, y \in S, b \in \{0, 1\})$

return $Enc(\sum^{-1}[x] + (-1)^b \sum^{-1}[y])$

Procedure $Enc(X \in \mathbb{Z}_L[Z_1, \dots, Z_N] \times \mathbb{Z}_{ord(E_2)} \times \dots \times \mathbb{Z}_{ord(E_n)})$

Let $X = (P, x_2, \dots, x_n)$

$P = P \cup \{P\}$

$X := (P(\vec{a}), x_2, \dots, x_n)$

If $\sum[X] = \perp$ **then**

$\sum[X] \leftarrow \{0, 1\}^{\lceil \log 2(|E|) \rceil} \setminus S$

$S := S \cup \{\sum[X]\}$

return $\sum[X]$

Abb. G_2

Oracle DL($j \in \{1, 2, \dots, N\}$)

```
for  $P_i \in P$  ▷  $G_3$   
  Let  $P_i = R_i + S_i$ ,  $R_i \in \mathbb{Z}_L[Z_1, \dots, Z_{j-1}, Z_{j+1}, \dots, Z_N]$ ,  $S_i \in \mathbb{Z}_L[Z_j]$   
   $R := R \cup \{R_i\}$   
  if  $\exists R_i, R_j \in R : R_i(\vec{a}) = R_j(\vec{a}) \wedge R_i \neq R_j$   
     $bad_1 := true$   
  abort ▷  $G_4$   
for  $P_i \in P$   
   $\sum[R_i(\vec{a}) + S_i] = \sum[P_i]$   
   $P_i := R_i(\vec{a}) + S_i$   
return  $\{a_i | i \in \{1, 2, \dots, N\} \setminus \{j\}\}$ 
```

Abb. $G_3 - G_4$

Game G_4 / G_5 / G_6 / G_7 / G_8

for $i \in \{1, 2, \dots, N\}$

$a_i \leftarrow \{2^{n-1}, 2^{n-1} + 2^c, \dots, 2^n - 2^c\}$

$P_i := Z_i$

$A_i := (P_i, 0, \dots, 0)$

$(a'_1, a'_2, \dots, a'_N) \leftarrow \mathcal{A}^{GOp(\cdot, \cdot, \cdot), DL(\cdot)}(Enc(B), Enc(E_2), \dots, Enc(E_m), Enc(A_1), \dots, Enc(A_N))$

if $\exists P_i, P_j \in \mathbf{P} : P_i(\vec{a}) = P_j(\vec{a}) \wedge P_i \neq P_j$

▷ G_5

$bad_2 := true$

abort

▷ G_6

return $(a_1, a_2, \dots, a_N) \stackrel{?}{=} (a'_1, a'_2, \dots, a'_N)$

Oracle $DL(j \in \{1, 2, \dots, N\})$

...

Oracle $GOp(x, y \in \mathbf{S}, b \in \{0, 1\})$

return $Enc(\sum^{-1}[x] + (-1)^b \sum^{-1}[y])$

Procedure $Enc(X \in \mathbb{Z}_L[Z_1, \dots, Z_N] \times \mathbb{Z}_{ord(E_2)} \times \dots \times \mathbb{Z}_{ord(E_n)})$

Let $X = (P, x_2, \dots, x_n)$

$\mathbf{P} = \mathbf{P} \cup \{P\}$

$X := (P(\vec{a}), x_2, \dots, x_n)$

If $\sum[X] = \perp$ then

$\sum[X] \leftarrow \{0, 1\}^{\lceil \log_2(|E|) \rceil} \setminus \mathbf{S}$

$\mathbf{S} := \mathbf{S} \cup \{\sum[X]\}$

return $\sum[X]$

Abb. $G_5 - G_6$

Game G_4 / G_5 / G_6 / G_7 / G_8

for $i \in \{1, 2, \dots, N\}$

$a_i \leftarrow \{2^{n-1}, 2^{n-1} + 2^c, \dots, 2^n - 2^c\}$

$P_i := Z_i$

$A_i := (P_i, 0, \dots, 0)$

$(a'_1, a'_2, \dots, a'_N) \leftarrow \mathcal{A}^{GOp(\cdot, \cdot, \cdot), DL(\cdot)}(Enc(B), Enc(E_2), \dots, Enc(E_m), Enc(A_1), \dots, Enc(A_N))$

if $\exists P_i, P_j \in \mathbf{P} : P_i(\vec{a}) = P_j(\vec{a}) \wedge P_i \neq P_j$

$bad_2 := true$

abort

return $(a_1, a_2, \dots, a_N) \stackrel{?}{=} (a'_1, a'_2, \dots, a'_N)$

Oracle $DL(j \in \{1, 2, \dots, N\})$

...

Oracle $GOp(x, y \in \mathbf{S}, b \in \{0, 1\})$

return $Enc(\sum^{-1}[x] + (-1)^b \sum^{-1}[y])$

Procedure $Enc(X \in \mathbb{Z}_L[Z_1, \dots, Z_N] \times \mathbb{Z}_{ord(E_2)} \times \dots \times \mathbb{Z}_{ord(E_n)})$

Let $X = (P, x_2, \dots, x_n)$

$\mathbf{P} = \mathbf{P} \cup \{P\}$

$\mathbf{X} := (P(\vec{a}), x_2, \dots, x_n)$

if $\sum[X] = \perp$ then

$\sum[X] \leftarrow \{0, 1\}^{\lceil \log_2(|E|) \rceil} \setminus \mathbf{S}$

$\mathbf{S} := \mathbf{S} \cup \{\sum[X]\}$

return $\sum[X]$

Abb. G_7

Game G_4 / G_5 / G_6 / G_7 / G_8

for $i \in \{1, 2, \dots, N\}$

$a_i \leftarrow \{2^{n-1}, 2^{n-1} + 2^c, \dots, 2^n - 2^c\}$

$P_i := Z_i$

$A_i := (P_i, 0, \dots, 0)$

$(a'_1, a'_2, \dots, a'_N) \leftarrow \mathcal{A}^{GOp(\cdot, \cdot, \cdot), DL(\cdot)}(Enc(B), Enc(E_2), \dots, Enc(E_m), Enc(A_1), \dots, Enc(A_N))$

for $i \in \{1, 2, \dots, N\}$

if $a_i = \perp$

$a_i \leftarrow \{2^{n-1}, 2^{n-1} + 2^c, \dots, 2^n - 2^c\}$

if $\exists P_i, P_j \in \mathbf{P} : P_i(\vec{a}) = P_j(\vec{a}) \wedge P_i \neq P_j$

$bad_2 := true$

abort

return $(a_1, a_2, \dots, a_N) \stackrel{?}{=} (a'_1, a'_2, \dots, a'_N)$

Oracle $DL(j \in \{1, 2, \dots, N\})$

for $i \in \{1, 2, \dots, N\} \setminus \{j\}$

$a_i \leftarrow \{2^{n-1}, 2^{n-1} + 2^c, \dots, 2^n - 2^c\}$

...

Oracle $GOp(x, y \in \mathbf{S}, b \in \{0, 1\})$

return $Enc(\sum^{-1}[x] + (-1)^b \sum^{-1}[y])$

Procedure $Enc(X \in \mathbb{Z}_L[Z_1, \dots, Z_N] \times \mathbb{Z}_{ord(E_2)} \times \dots \times \mathbb{Z}_{ord(E_n)})$

...

Abb. G_8

Definition (Success Ratio [9])

Let adversary $\mathcal{A}$ be an adversary with runtime $\text{Time}(\mathcal{A})$ and advantage $\text{Adv}_{\mathcal{A}}$. Its success ratio is defined as following:

$$SR(\mathcal{A}) = \frac{\text{Adv}_{\mathcal{A}}}{\text{Time}(\mathcal{A})}.$$

Definition (Bit Security [9])

A cryptographic scheme has κ bit security if the success ratio of all adversaries with a runtime $\text{Time}(\mathcal{A}) \leq 2^{\kappa}$ is upper bounded by $2^{-\kappa}$.

$$\begin{aligned}
SR(\mathcal{A}) &\leq \frac{\text{Adv}_{\mathbb{G}, \mathcal{A}}^{\text{SUF-CMA}}(\lambda)}{\text{Time}(\mathcal{A})} \\
&\leq \frac{\text{Adv}_{E, n, c, L, \mathcal{B}}^{\text{Ed-DLog}} + \frac{2(q_h+1)}{2^b} + \frac{q_o q_h + q_o}{2^{-\log_2(\lceil \frac{2^{2b}-1}{L} \rceil 2^{-2b})}}}{\text{Time}(\mathcal{A})} \\
&\leq \frac{\frac{(q_g+3)^2+1}{2^{n-1-c}} + \frac{2(q_h+1)}{2^b} + \frac{q_o q_h + q_o}{2^{-\log_2(\lceil \frac{2^{2b}-1}{L} \rceil 2^{-2b})}}}{\text{Time}(\mathcal{A})} \\
&\leq \frac{(2^{125} + 3)^2 + 1}{2^{250} 2^{125}} + \frac{2(2^{125} + 1)}{2^{256} 2^{125}} + \frac{2^{64} 2^{125} + 2^{64}}{2^{252} 2^{125}} \\
&\approx 2^{-125} + 2^{-316} + 2^{-189} \\
&\approx 2^{-125}
\end{aligned}$$

Thank you!
Questions?



J. Brendel, C. Cremers, D. Jackson, and M. Zhao, “The provable security of Ed25519: Theory and practice,” in *2021 IEEE Symposium on Security and Privacy*, (San Francisco, CA, USA), pp. 1659–1676, IEEE Computer Society Press, May 24–27, 2021.



K. Chalkias, F. Garillot, and V. Nikolaenko, “Taming the many EdDSAs.” Cryptology ePrint Archive, Report 2020/1244, 2020.
<https://eprint.iacr.org/2020/1244>.



G. Fuchsbauer, A. Plouviez, and Y. Seurin, “Blind schnorr signatures and signed ElGamal encryption in the algebraic group model,” in *Advances in Cryptology – EUROCRYPT 2020, Part II* (A. Canteaut and Y. Ishai, eds.), vol. 12106 of *Lecture Notes in Computer Science*, (Zagreb, Croatia), pp. 63–95, Springer, Heidelberg, Germany, May 10–14, 2020.