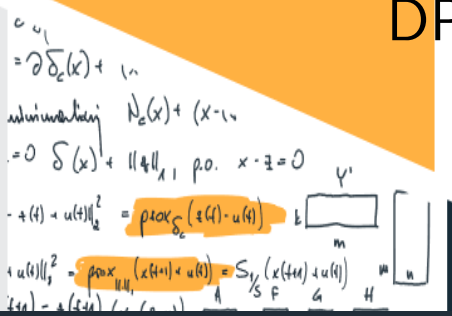
Kleptographic SETUP against Pseudonym Unlinkability or: check your EDIW Wallet

Mirosław Kutyłowski[2]
joint work with Przemysław Błaśkiewicz[1], Anna Lauks-Dutka[1], and Przemysław Kubiak[2]

[1] Wrocław University of Science and Technology
[2] NASK – National Research Institute, Poland

DPM'26 presentation



D'où venons nous? Où allons nous?

SCIENCE NASK

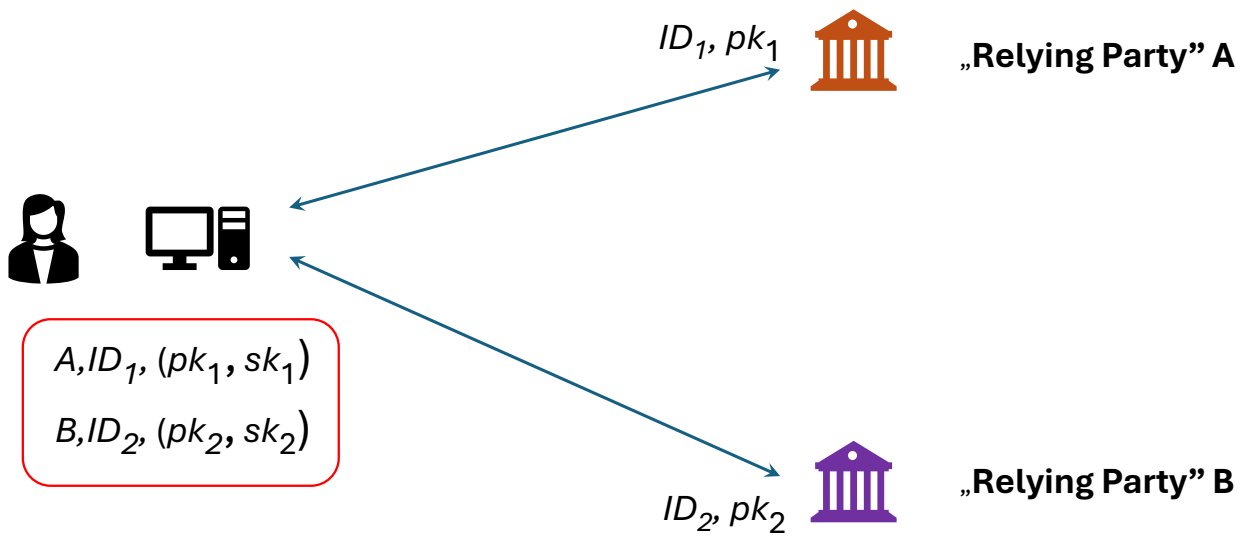
eIDAS Legal Framework, CIR 2024/2979

Article 14

Pseudonyms

1. Wallet units shall support the generation of pseudonyms for wallet users in compliance with the technical specifications set out in Annex V.

2. Wallet units shall support the generation, upon the request of a wallet-relying party, of a pseudonym which is specific and unique to that wallet-relying party and provide this pseudonym to the wallet-relying party, either standalone or in combination with any person identification data or electronic attribute attestation requested by that wallet-relying party.



WebAuthn - the standard chosen in Annex B

Registration (details omitted):

1. the wallet **randomly generates a key pair** (sk, pk)
2. . . . and signs the hash of the challenge obtained from the Relying Party
3. the Relying Party stores (*pseudonym*, pk)

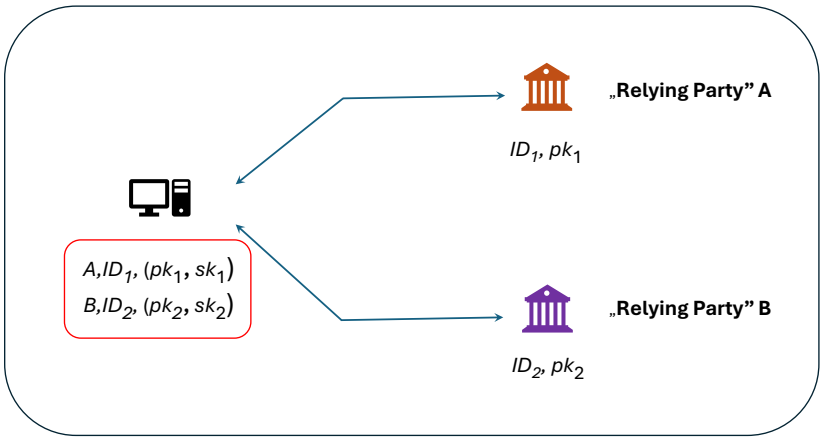
Authentication:

Challenge-response protocol: **sign a challenge** with sk

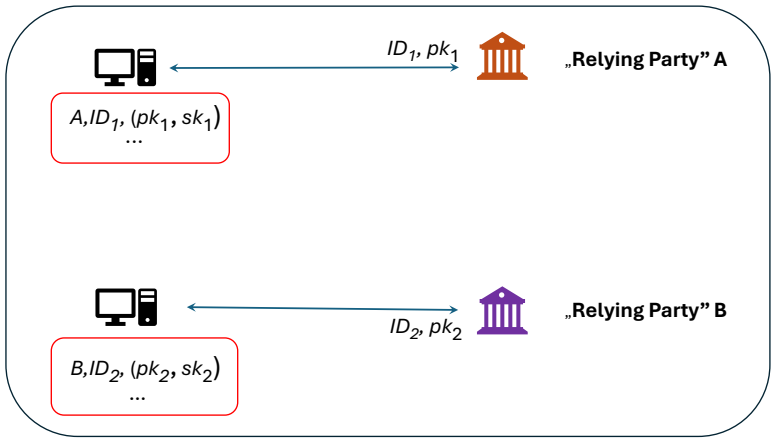
Unlinkability

Left or right?

The Relying Parties are curious whether pk_1 and pk_2 correspond to the same wallet

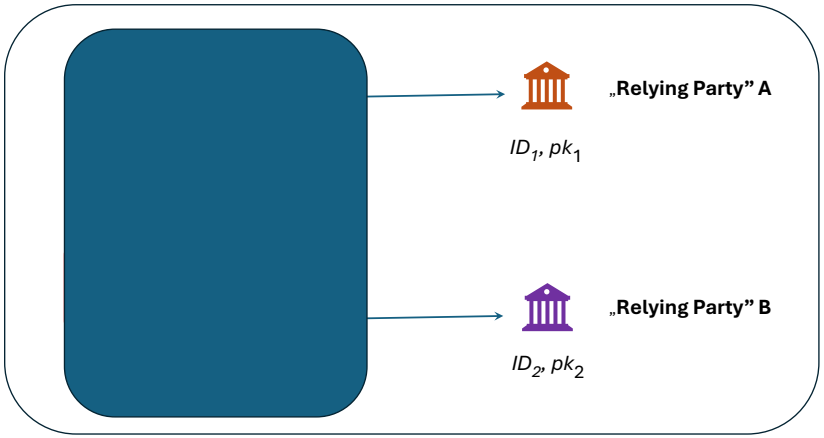


or

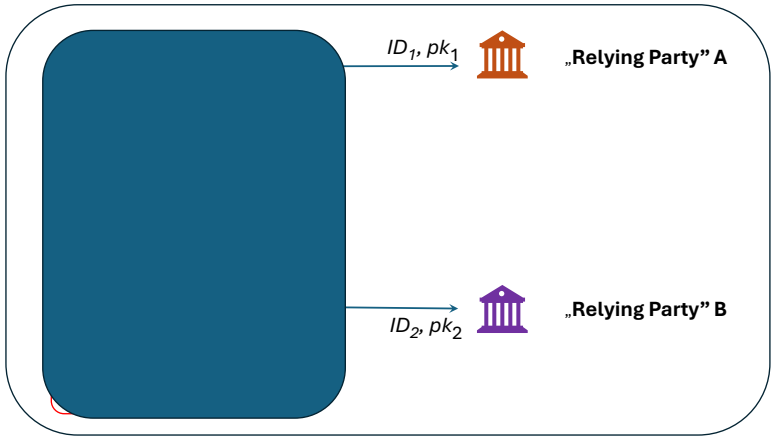


Unlinkability

What they really see



or



And, allegedly, the keys have been chosen independently at random

Impossible to distinguish **if the wallets follow the protocol**

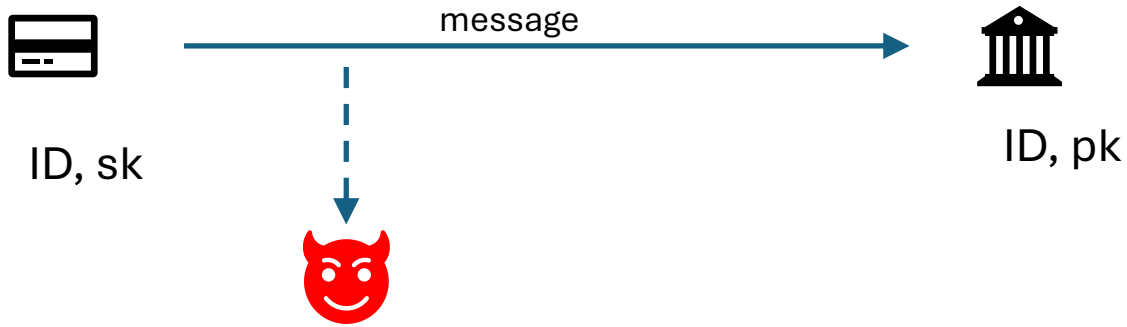
D'où venons nous? Où allons nous?

SCIENCE NASK

Black-Box Device

Kleptographic Black Box Device

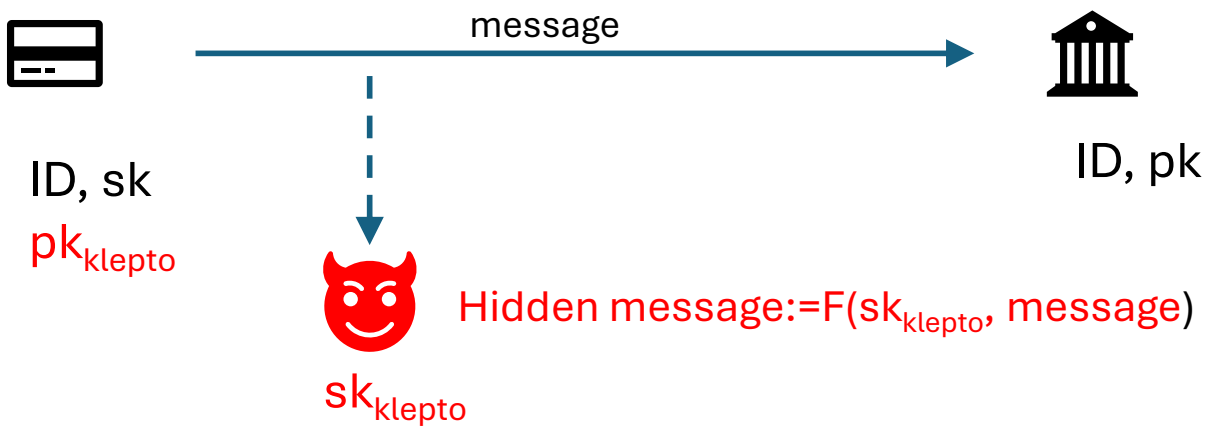
- the device appears to execute protocol P
- ... but in fact it executes malicious protocol klepto(P) such that
  -
  -
  -



Black-Box Device

Kleptographic Black Box Device

- the device appears to execute protocol P
- ... but in fact it executes malicious protocol klepto(P) such that
  - nobody but Adv holding kleptographic secret key sk_{klepto} can detect the difference,
  - the device does not store sk_{klepto},
  - klepto(P) provides a covert channel from the device to Adv



Black-Box Device

Anamorphic Black Box Device

Same as a kleptographic device, but the covert channel is undetectable even if the secret keys of P are revealed.

Law Enforcement versus Wallet Misuse

What to do in case of criminal activity of a wallet?

Law enforcement needs to de-anonymize the user and blacklist the other pseudonyms associated with the same wallet.

Big Brother

The wallet provider delivers wallets that enable linking the pseudonyms while pretending to execute the original protocol.

SCIENCE NASK

Law Enforcement versus Wallet Misuse

What to do in case of criminal activity of a wallet?

Law enforcement needs to de-anonymize the user and blacklist the other pseudonyms associated with the same wallet.

Big Brother

The wallet provider delivers wallets that enable linking the pseudonyms while pretending to execute the original protocol.

Enabling linking pseudonyms might be necessary, but it is a delicate issue. We skip the discussion and focus on feasibility.

SCIENCE NASK

D'où venons nous? Où allons nous?

SCIENCE NASK

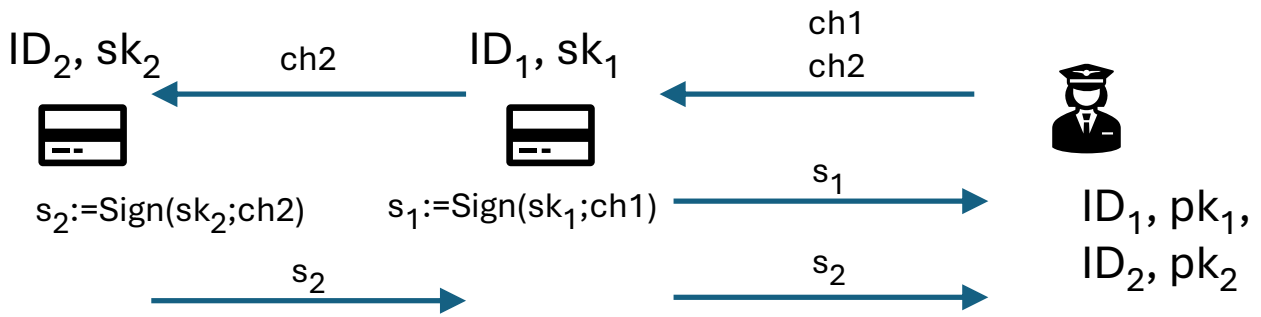
Proof of possession of 2 pseudonyms

How to show that sk_1 and sk_2 are on the same wallet?

- Self-declaration is insufficient
- ... as it can lead to identity theft

Undeniable evidence needed

- a wallet cannot claim to hold sk_i without holding it
- two different wallets cannot pretend to be the same wallet



Proof of possession of 2 pseudonyms

Solution via extractability

- the wallet proves possession of sk_1 by signing a challenge
- possession of sk_2 follows from extractability:
  - sk_2 **can be derived by the signer who knows** sk_1 **and the ephemeral private key used to create the signature**

D'où venons nous? Où allons nous?

SCIENCE NASK

Schnorr signature creation

Require: signing private key sk, message M, random nonce k

procedure $\mathtt{EC-Schnorr}(\mathrm{sk}, M, k)$

$R \leftarrow k \cdot G$

$e \leftarrow \mathrm{Hash}(R_x, R_y, M)$ where R_x, R_y are x and y-coordinate of R

$s \leftarrow k + \mathrm{sk} \cdot e \bmod q$

return signature (s, e)

ECDSA Signature Creation with a Random Nonce

Require: signing private key sk, message M, random nonce k

procedure ECDSA(sk, M, k)

$R \leftarrow k \cdot G$

$k_{\text{inv}} \leftarrow k^{-1} \bmod q$

$s \leftarrow k_{\text{inv}} \cdot (R_x \cdot \text{sk} + \text{Hash}(M)) \bmod q$ with $R_x = x$-coordinate of R

return signature (R_x, s)

- the signer must know sk to create a valid signature
- so the signer can extract k used for signature creation

Creating ECDSA with hidden Schnorr

Require: signing keys $\mathsf{sk}_1, \mathsf{sk}_2$, message M, random ephemeral u with $U = u \cdot G$ known to Adv, shared key dkey, internal counter ctr

procedure `lnk_ECDSA`(sk_1, sk_2, M, u, dkey, ctr, U)

▷ *Start of Phase 1*

increment ctr

$S \leftarrow \mathsf{Hash}(\mathsf{dkey}, \mathsf{ctr})$

$e \leftarrow \mathsf{Hash}(U_x, U_y, S, M)$

$k \leftarrow u + \mathsf{sk}_2 \cdot e \bmod q$

▷ *Start of Phase 2*

$R \leftarrow k \cdot G$

$k_{\mathsf{inv}} \leftarrow k^{-1} \bmod q$

$s \leftarrow k_{\mathsf{inv}} \cdot (R_x \cdot \mathsf{sk}_1 + \mathsf{Hash}(M)) \bmod q$

return signature (R_x, s)

Verification by Adv

procedure lnk–ECDSA(sk_1, sk_2, M, u, dkey, ctr, U)

▷ *Start of Phase 1*

increment ctr
$S \leftarrow \mathrm{Hash}(\mathrm{dkey}, \mathrm{ctr})$
$e \leftarrow \mathrm{Hash}(U_x, U_y, S, M)$
$k \leftarrow u + \mathrm{sk}_2 \cdot e \bmod q$

▷ *Start of Phase 2*

$R \leftarrow k \cdot G$
$k_{\mathrm{inv}} \leftarrow k^{-1} \bmod q$
$s \leftarrow k_{\mathrm{inv}} \cdot (R_x \cdot \mathrm{sk}_1 + \mathrm{Hash}(M)) \bmod q$
return signature (R_x, s)

Verification process

(a) check (R_x, s) as ECDSA, R recalculated in this way

(b) recalculate e

(c) verify that $k \leftarrow u + \mathrm{sk}_2 \cdot e \bmod q$ by testing $R \stackrel{?}{=} U + e \cdot \mathrm{PK}_2$

Verification by Adv

Conclusion

1. the signer knows sk_1 as (R_x, s) is a valid ECDSA
2. consequently, the signer also knows k used for ECDSA
3. as the signer knows e and u, while $k = u + e \cdot \mathrm{sk}_2 \bmod q$ (tested via $R \stackrel{?}{=} U + e \cdot \mathrm{PK}_2$), he can extract sk_2

The signer has access to both sk_1 and sk_2

SCIENCE NASK

Point of view of other parties

procedure lnk–ECDSA(sk_1, sk_2, M, u, dkey, ctr, U)

▷ *Start of Phase 1*

increment ctr

$S \leftarrow \mathrm{Hash}(\mathrm{dkey}, \mathrm{ctr})$
$e \leftarrow \mathrm{Hash}(U_x, U_y, S, M)$
$k \leftarrow u + \mathrm{sk}_2 \cdot e \bmod q$

▷ *Start of Phase 2*

$R \leftarrow k \cdot G$
$k_{\mathrm{inv}} \leftarrow k^{-1} \bmod q$
$s \leftarrow k_{\mathrm{inv}} \cdot (R_x \cdot \mathrm{sk}_1 + \mathrm{Hash}(M)) \bmod q$
return signature (R_x, s)

Point of view – security games approach

Replace S with a random value

Point of view of other parties

procedure lnk–ECDSA(sk_1, sk_2, M, u, dkey, ctr, U)

▷ *Start of Phase 1*

$S \leftarrow \text{random}()$

$e \leftarrow \text{Hash}(U_x, U_y, S, M)$
$k \leftarrow u + \mathsf{sk}_2 \cdot e \bmod q$

▷ *Start of Phase 2*

$R \leftarrow k \cdot G$
$k_{\text{inv}} \leftarrow k^{-1} \bmod q$
$s \leftarrow k_{\text{inv}} \cdot (R_x \cdot \mathsf{sk}_1 + \text{Hash}(M)) \bmod q$
return signature (R_x, s)

Point of view – security games approach

Replace e with a random value

Point of view of other parties

procedure lnk–ECDSA(sk_1, sk_2, M, u, dkey, ctr, U)

▷ *Start of Phase 1*

$e \leftarrow \mathsf{random}()$

$k \leftarrow u + \mathsf{sk}_2 \cdot e \bmod q$

▷ *Start of Phase 2*

$R \leftarrow k \cdot G$
$k_{\mathsf{inv}} \leftarrow k^{-1} \bmod q$
$s \leftarrow k_{\mathsf{inv}} \cdot (R_x \cdot \mathsf{sk}_1 + \mathsf{Hash}(M)) \bmod q$
return signature (R_x, s)

Point of view – security games approach

Replace k with a random value

Point of view of other parties

procedure lnk–ECDSA(sk_1, sk_2, M, u, dkey, ctr, U)

▷ *Start of Phase 1*

$k \leftarrow \mathsf{random}()$

▷ *Start of Phase 2*

$R \leftarrow k \cdot G$
$k_{\mathsf{inv}} \leftarrow k^{-1} \bmod q$
$s \leftarrow k_{\mathsf{inv}} \cdot (R_x \cdot \mathsf{sk}_1 + \mathsf{Hash}(M)) \bmod q$
return signature (R_x, s)

Point of view – security games approach

Phase 2 – a regular ECDSA signature

D'où venons nous? Où allons nous?

SCIENCE NASK

Conclusion

a kleptographic device **can prove** to Adv that it knows both sk_1 and sk_2 for WebAuthn with ECDSA, EC-Schnorr, ...

THANK YOU FOR YOUR ATTENTION

SCIENCE NASK