Key Blinding for Signature Schemes

draft-irtf-cfrg-signature-key-blinding
Setting: Single Prover

\[ \sigma = \text{Sign}(sk, msg) \]

Unforgeability: Given \((msg, pk, \sigma)\), will the Verifier conclude that the owner of \(sk\) produced \(\sigma\) with overwhelming probability? ✔
Unforgeability: Given \((\text{msg}, \text{pk}, \sigma)\), will the Verifier conclude that the owner of \(\text{sk}\) produced \(\sigma\) with overwhelming probability? ✓

... but what if one wanted the signature or public to reveal nothing about the Prover?

- Tor hidden service identity blinding protocol: Signing hidden service descriptor
- Privacy Pass rate limiting: Signing Privacy Pass token requests
- Cryptocurrency private airdrop: Computing public airdrop tokens
Setting: Multiple Provers

**Prover 0**

\[ \sigma_0 = \text{Sign}(sk_0, \text{msg}) \]

**Prover 1**

\[ \sigma_1 = \text{Sign}(sk_1, \text{msg}) \]

**Mediator**

\[ b \leftarrow \{0,1\}^* \]

**Verifier**

\[ a = \text{Verify}(pk_b, \text{msg}, \sigma_b) \]

\[ b' = ? \]
Setting: Multiple Provers

1. Unforgeability: Given \((msg, pk_b, \sigma_b)\), will the Verifier conclude that the owner of \(sk_b\) produced \(\sigma_b\) with overwhelming probability? 

2. Unlinkability: Given \((msg, pk_b, \sigma_b)\), can the Verifier determine \(b\) with probability not negligibly better than \(1/2\)?
Setting: Multiple Provers

1. Unforgeability: Given \((\text{msg}, \text{pk}_b, \sigma_b)\), will the Verifier conclude that the owner of \(\text{sk}_b\) produced \(\sigma_b\) with overwhelming probability?

2. Unlinkability: Given \((\text{msg}, \text{pk}_b, \sigma_b)\), can the Verifier determine \(b\) with probability not negligibly better than \(1/2\)?

\[
\begin{align*}
\text{(sk}_0, \text{pk}_0) & \quad \text{msg} & \quad \text{Prover 0} & \quad \sigma_0 = \text{Sign}(\text{sk}_0, \text{msg}) \\
\text{(sk}_1, \text{pk}_1) & \quad \text{msg} & \quad \text{Prover 1} & \quad \sigma_1 = \text{Sign}(\text{sk}_1, \text{msg}) \\
\text{Mediator} & \quad \text{(msg, pk}_0, \sigma_0) \quad & b \leftarrow \{0,1\}^* \\
\text{Verifier} & \quad \text{(msg, pk}_b, \sigma_b) \quad & a = \text{Verify}(\text{pk}_b, \text{msg}, \sigma_b) \\
& & b' = ? \\
\text{a, b'} & \quad & \text{(pk}_1, \text{pk}_1) \quad \text{msg}
\end{align*}
\]
Functional Requirements

Unforgeable signature scheme with the following additional properties:

1. Per-message public keys are independently distributed from long-term public keys

2. Per-message signatures do not leak any information about the long-term signing keys

Proposed solution: signature schemes with key blinding
Signature Scheme with Key Blinding

Extend digital signature schemes with three functions

1. BlindKeyGen: Produce a *blinding key*

2. BlindPublicKey: Given public key and blinding key, produce *blinded public key*

3. BlindKeySign: Sign message with secret key and secret blind

\[
\text{Verify(BlindPublicKey}(\text{pk}_S, \text{sk}_B), \text{msg}, \text{BlindKeySign}(\text{sk}_S, \text{sk}_B, \text{msg})) = 1
\]
Signature Scheme with Key Blinding

Optionally add one more function for unblinding public keys

4. UnblindPublicKey: Given blinded public key and blinding key, produce an unblinded public key

\[
\text{UnblindPublicKey}(\text{BlindPublicKey}(pk_S, sk_B), sk_B) = pk_S
\]

… how is this optionally done in practice?
Generalizing Key Blinding

Generalize BlindPublicKey (and related functions) to support a context string

BlindPublicKey(pk_S, sk_B, ctx)

where context varies based on application use case, e.g.

\[ \text{ctx} = \bot, \text{ Rate-limited privacy pass } \]

\[ \text{ctx} = (pk_R, \text{timestamp}), \text{ Tor hidden services} \]

See https://github.com/cfrg/draft-irtf-cfrg-signature-key-blinding/pull/37
Status and Next Steps

Implementation status:

PureEdDSA (RFC8032) and ECDSA key blinding extension support and test vectors

Several interoperable implementations exist

Security analysis:

Unlinkability and unforgeability analysis for EdDSA and ECDSA variants complete (under peer review)

Next steps: merge PR#37 and solicit early Crypto Panel reviews
Questions?
Key Blinding for Signature Schemes

draft-irtf-cfrg-signature-key-blinding