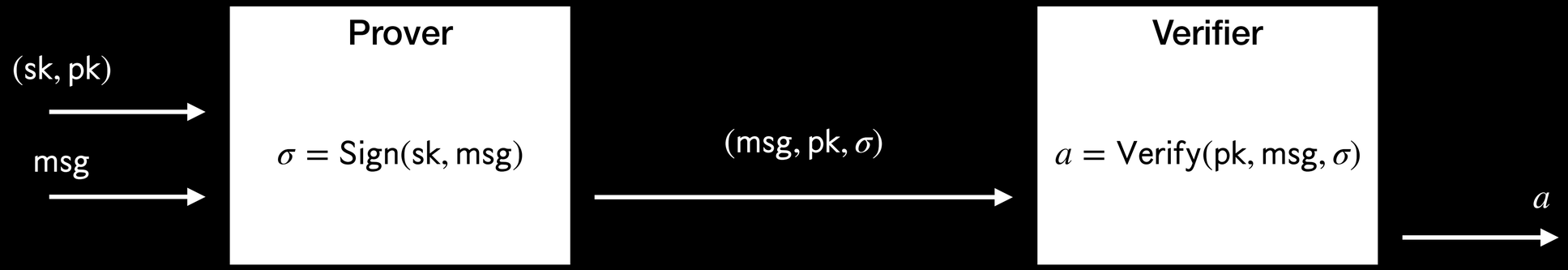
Key Blinding for Signature Schemes

`draft-irtf-cfrg-signature-key-blinding`

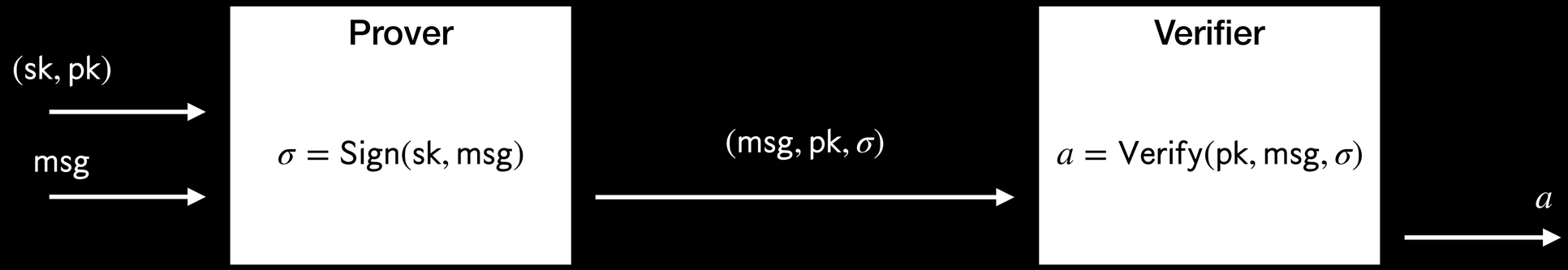
Denis, Eaton, Wood — IETF 114 — CFRG

Setting: Single Prover



Unforgeability: Given (msg, pk, σ), will the Verifier conclude that the owner of sk produced σ with overwhelming probability? ✔

Setting: Single Prover

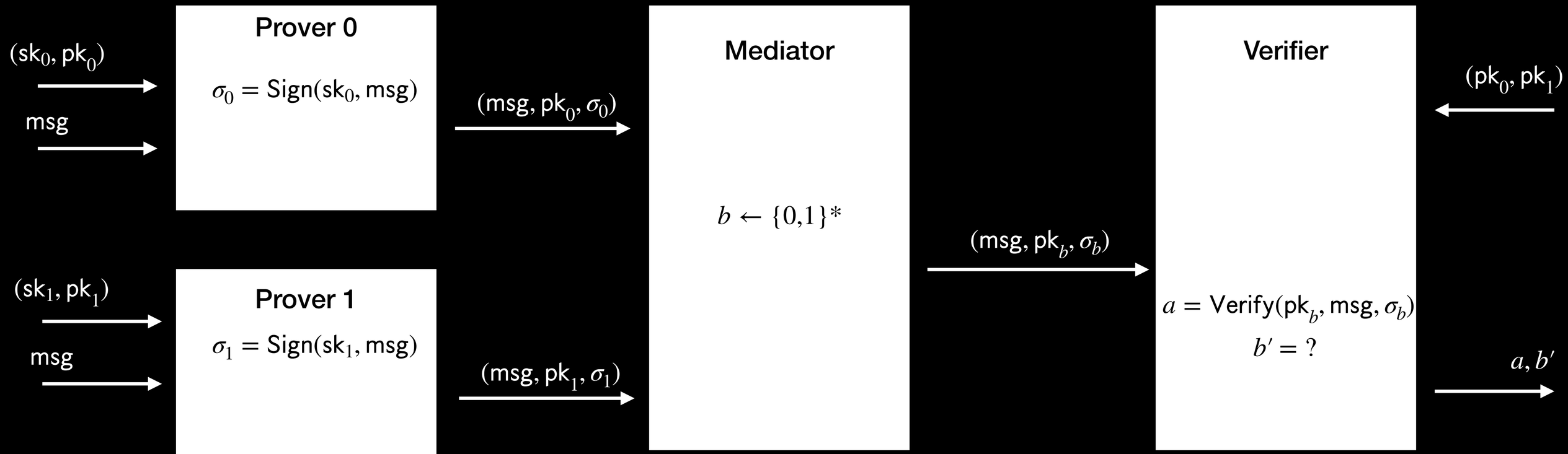


Unforgeability: Given (msg, pk, σ), will the Verifier conclude that the owner of sk produced σ with overwhelming probability? ✔

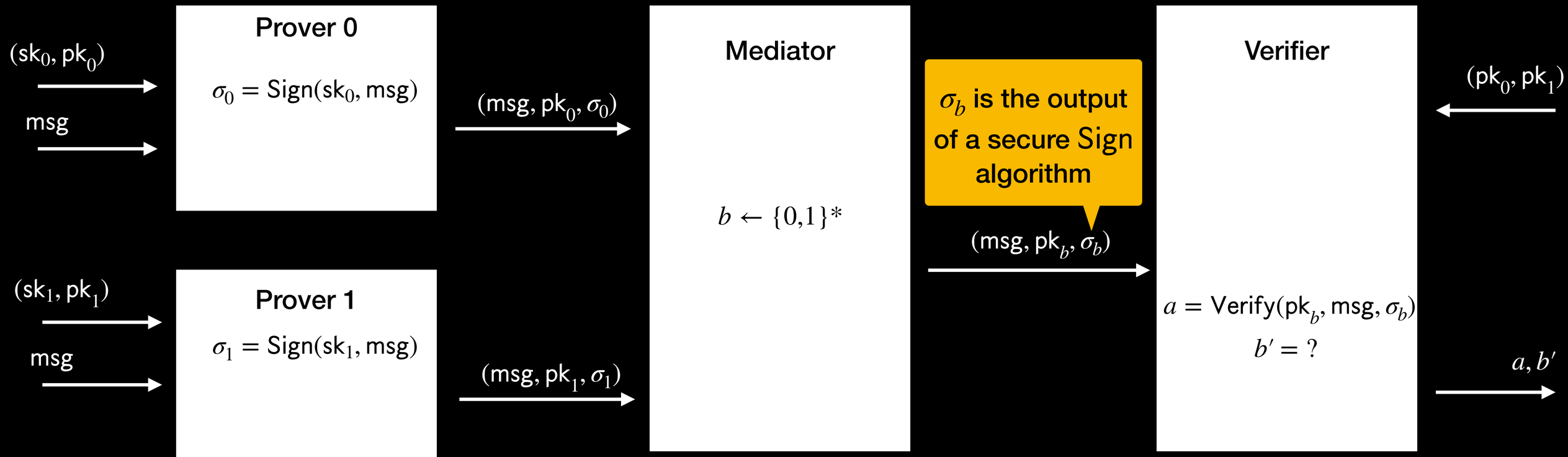
… but what if one wanted the signature or public to reveal nothing about the Prover?

- 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

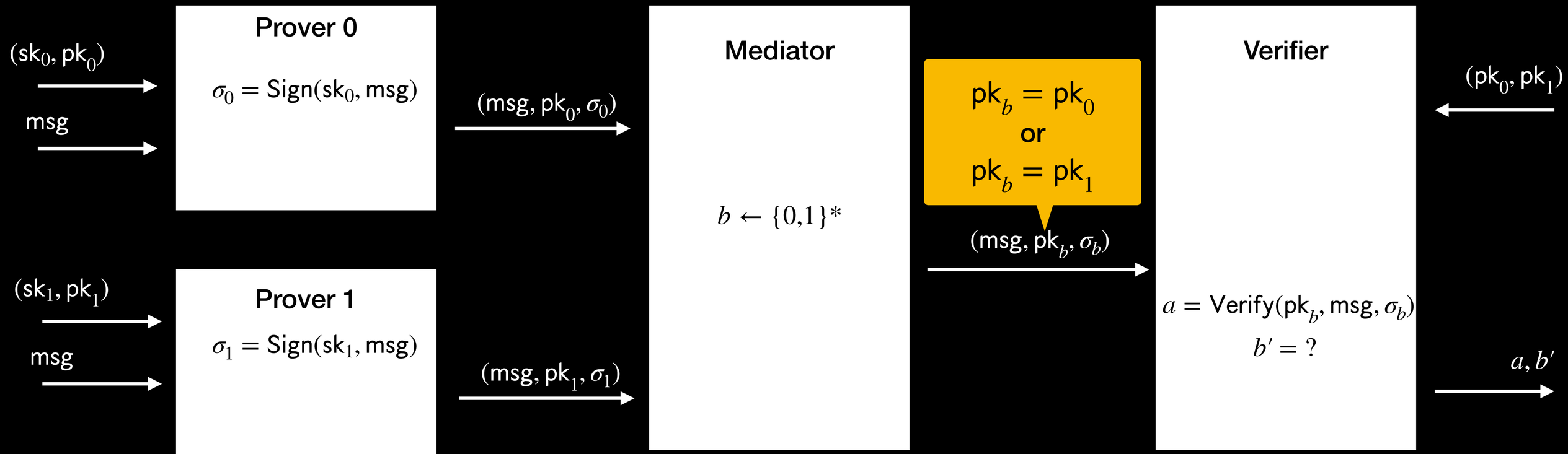


Setting: Multiple Provers



1. Unforgeability: Given $(\text{msg}, \text{pk}_b, \sigma_b)$, will the Verifier conclude that the owner of sk_b produced σ_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?

Setting: Multiple Provers



1. Unforgeability: Given $(\text{msg}, \text{pk}_b, \sigma_b)$, will the Verifier conclude that the owner of sk_b produced σ_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? ✗

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

$$\text{Verify}(\text{BlindPublicKey}(\text{pk}_S, \text{sk}_B), \text{msg}, \text{BlindKeySign}(\text{sk}_S, \text{sk}_B, \text{msg})) = 1$$

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*

$$\text{UnblindPublicKey}(\text{BlindPublicKey}(\text{pk}_S, \text{sk}_B)), \text{sk}_\text{B}) = \text{pk}_\text{S}$$

… how is this *optionally* done in practice?

Generalizing Key Blinding

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

$$\text{BlindPublicKey}(\text{pk}_S, \text{sk}_B, \text{ctx})$$

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

- $\text{ctx} = \perp$, Rate-limited privacy pass
- $\text{ctx} = (\text{pk}_R, \text{timestamp})$, 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`

Denis, Eaton, Wood — IETF 114 — CFRG