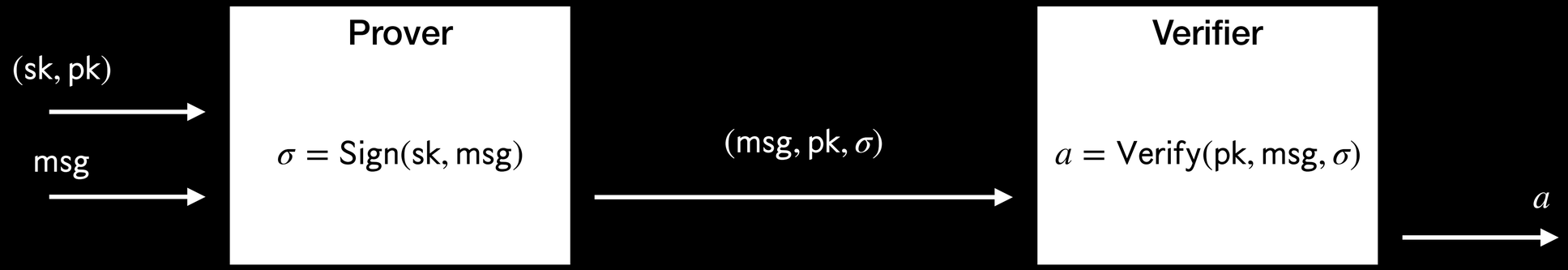
Key Blinding for Signature Schemes

draft-dew-cfrg-signature-key-blinding

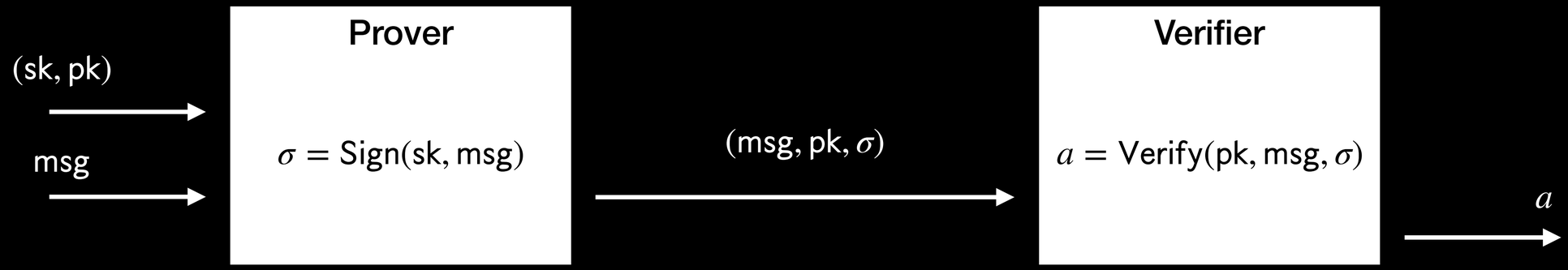
Denis, Eaton, Wood — IETF 113 — CFRG

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

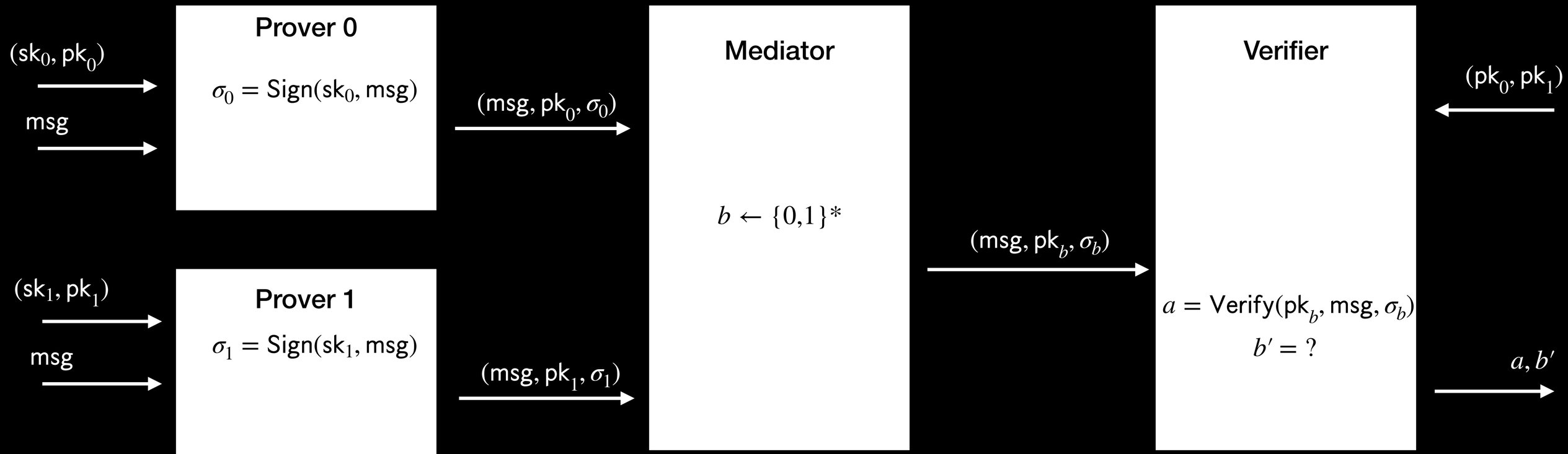


Unforgeability: Given (msg, pk, σ), will the Verifier conclude that the owner of sk produced σ with overwhelming probability? ✓

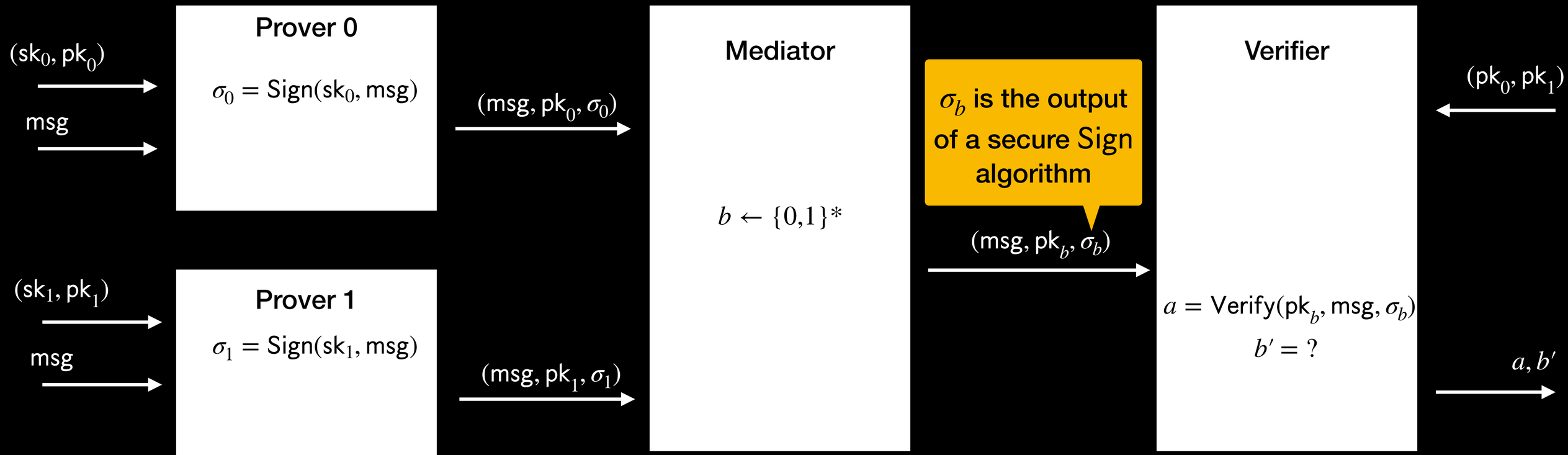
… but what if one wanted the signature or public key to reveal nothing about the Prover?

- 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
- BIP32 wallets

Setting: Multiple Provers

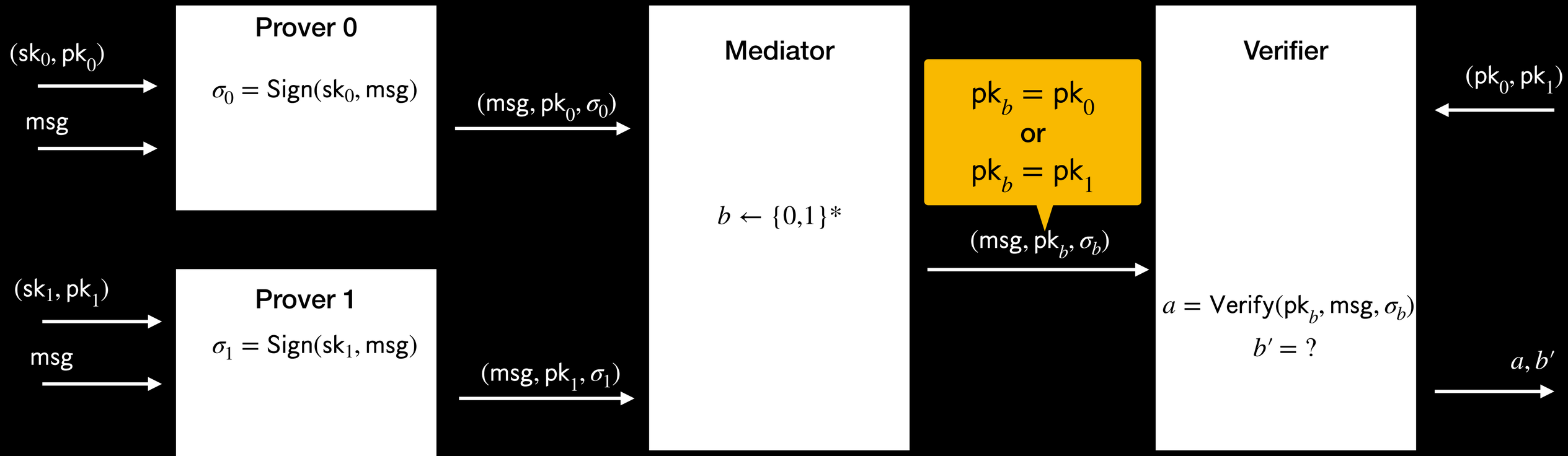


Setting: Multiple Provers



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 two functionalities

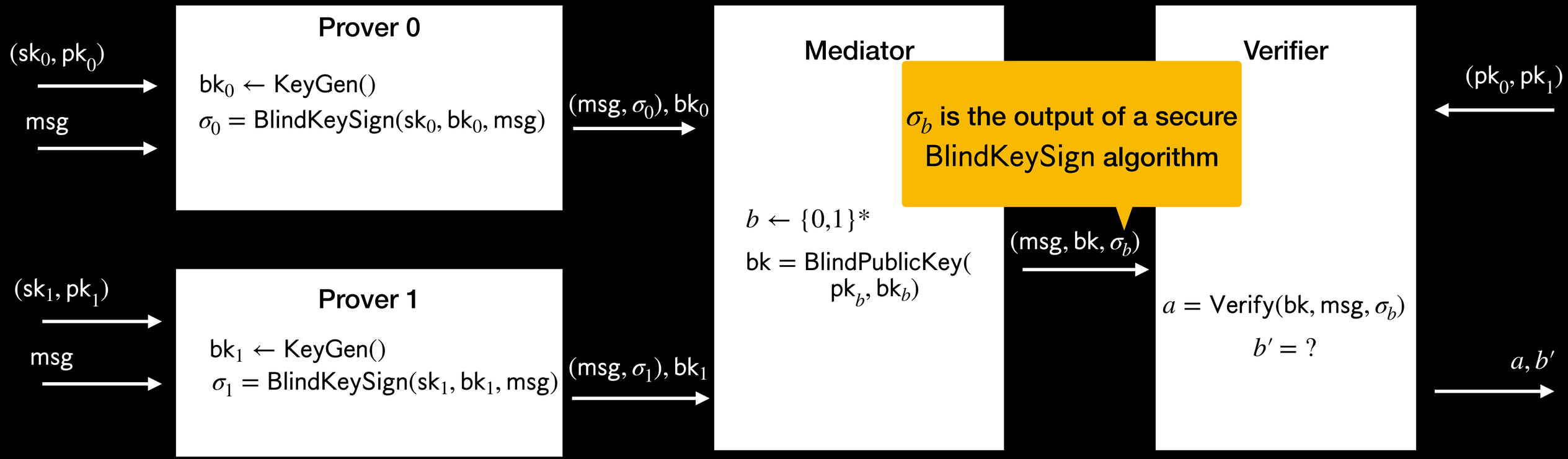
1. BlindPublicKey and UnblindPublicKey: Given public key and blinding key, produce *blinded public key* that is independent from input public key

$$\mathsf{UnblindPublicKey}(\mathsf{BlindPublicKey}(\mathsf{pk}_S, \mathsf{bk})), \mathsf{bk}) = \mathsf{pk}_S$$

2. BlindKeySign: Sign message with signing key and blinding key, producing a signature that is independent of the input signing key.

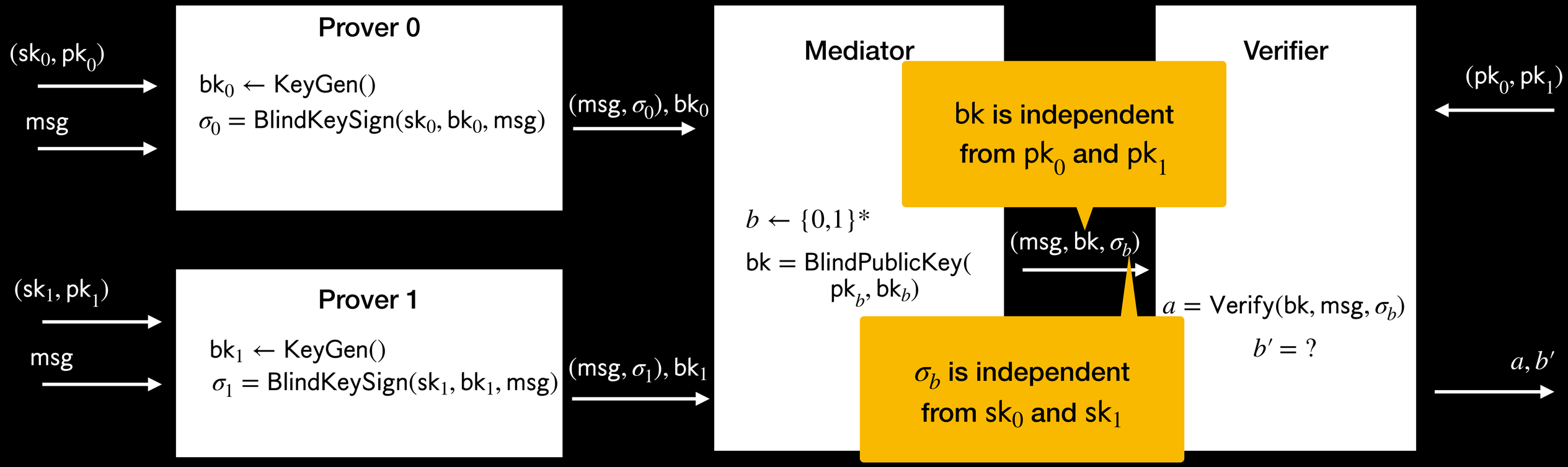
$$\mathsf{Verify}(\mathsf{BlindPublicKey}(\mathsf{pk}_S, \mathsf{bk}), \mathsf{msg}, \mathsf{BlindKeySign}(\mathsf{sk}_S, \mathsf{bk}, \mathsf{msg})) = 1$$

Setting: Multiple Provers



1. Unforgeability: Given (msg, bk, σ_b), will the Verifier conclude that the owner of sk_b produced σ_b with overwhelming probability? ✔
2. Unlinkability: Given (msg, bk, σ_b), can the Verifier determine b with probability not negligibly better than 1/2?

Setting: Multiple Provers



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

2. Unlinkability: Given (msg, bk, σ_b), can the Verifier determine b with probability not negligibly better than 1/2? ✓

Current Status

Implementation status:

- PureEdDSA (RFC8032) and ECDSA key blinding extension support and test vectors
- Several interoperable implementations exist

Unlinkability and unforgeability analysis for EdDSA and ECDSA variants is underway

1) Interest in working on signature schemes with key blinding support?

2) Interest in adopting this document?