On the Security of Two-Round Multi-Signatures

Manu Drijvers[1], Kasra Edalatnejad[2], Bryan Ford[2], Eike Kiltz[3], Julian Loss[3], Gregory Neven[1], Igors Stepanovs[4]

[1] DFINITY, [2] EPFL , [3] Ruhr-University Bochum, [4] UCSD

To appear at S&P 2019, full version on ePrint 2018/417

Multi-signatures

$(pk_1,sk_1) \leftarrow Kg$ | $(pk_2,sk_2) \leftarrow Kg$ | $(pk_3,sk_3) \leftarrow Kg$

$\mathrm{Sign}((pk_1,pk_2,pk_3), sk_1, m) \rightarrow \sigma$ $\leftrightarrow$ $\mathrm{Sign}((pk_1,pk_2,pk_3), sk_2, m) \rightarrow \sigma$ $\leftrightarrow$ $\mathrm{Sign}((pk_1,pk_2,pk_3), sk_3, m) \rightarrow \sigma$

$$\mathrm{Verify}((pk_1,pk_2,pk_3), m, \sigma) = 1$$

Every signer must agree to sign m

Goal: **short signature** (preferably ≈ single signature, definitely << N signatures)

efficiently verifiable

Multi-signatures

$(pk_1, sk_1) \leftarrow Kg$ $(pk_2, sk_2) \leftarrow Kg$ $(pk_3, sk_3) \leftarrow Kg$

$Sign((pk_1, pk_2, pk_3), sk_1, m) \leftrightarrow Sign((pk_1, pk_2, pk_3), sk_2, m) \leftrightarrow Sign((pk_1, pk_2, pk_3), sk_3, m)$

$\rightarrow \sigma$ $\rightarrow \sigma$ $\rightarrow \sigma$

Key aggregation: $apk \leftarrow KAgg(pk_1, pk_2, pk_3)$

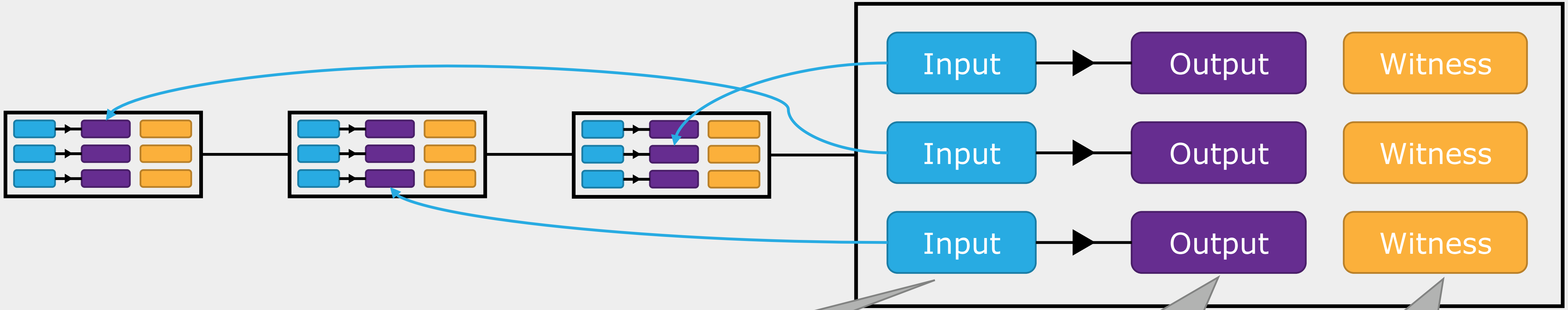
$Verify(apk, m, \sigma) = 1$

Every signer must agree to sign m

<u>**Goal:**</u> **short signature** (preferably ≈ single signature,

efficiently verifiable definitely << N signatures)

Bitcoin blockchain and transactions



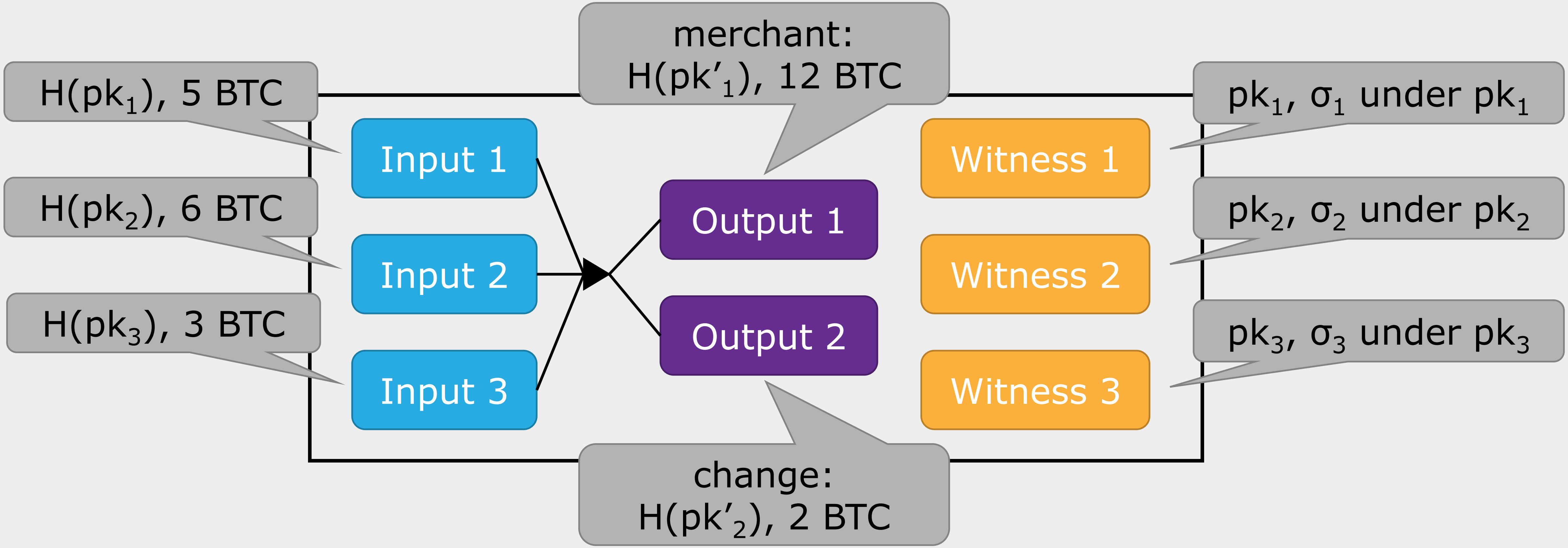
Pointer to sender = unspent output with
addr_{in} = H(pk)
$\text{amount}_{\text{in}}$ = 1 BTC

recipient address & amount
addr_{out} = H(pk')
$\text{amount}_{\text{out}}$ = 1 BTC

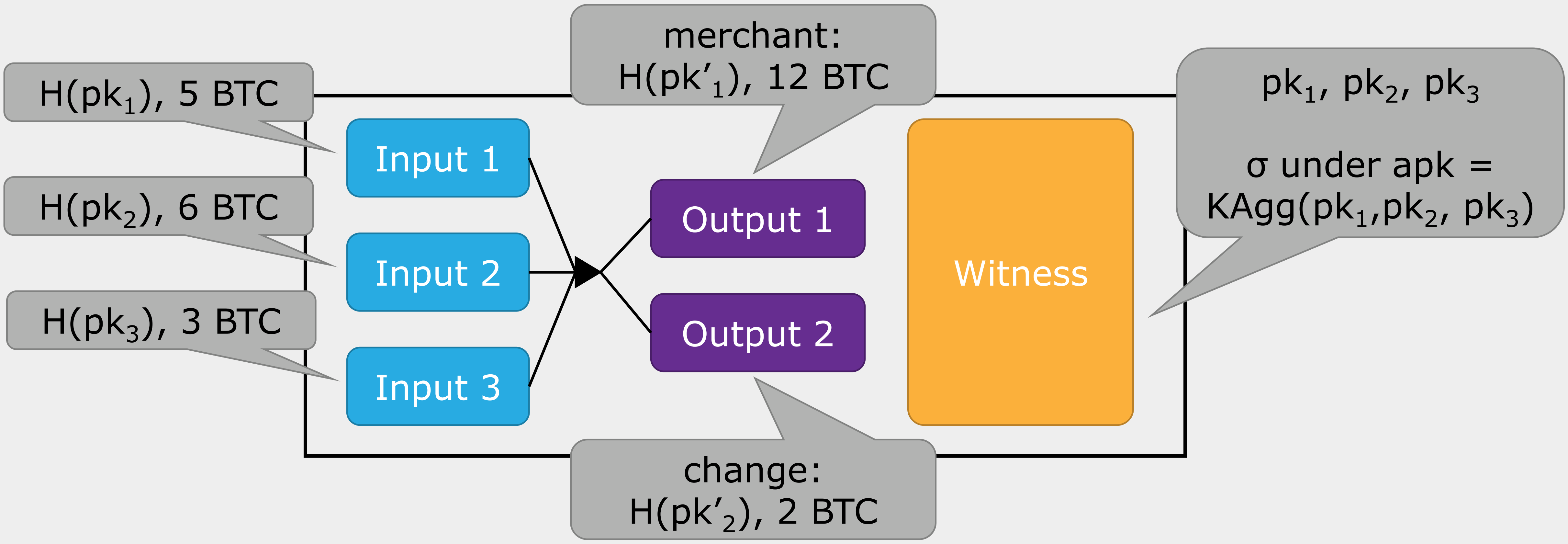
Witness data

pk, σ under pk

Multi-input/output transactions



Multi-input/output transactions



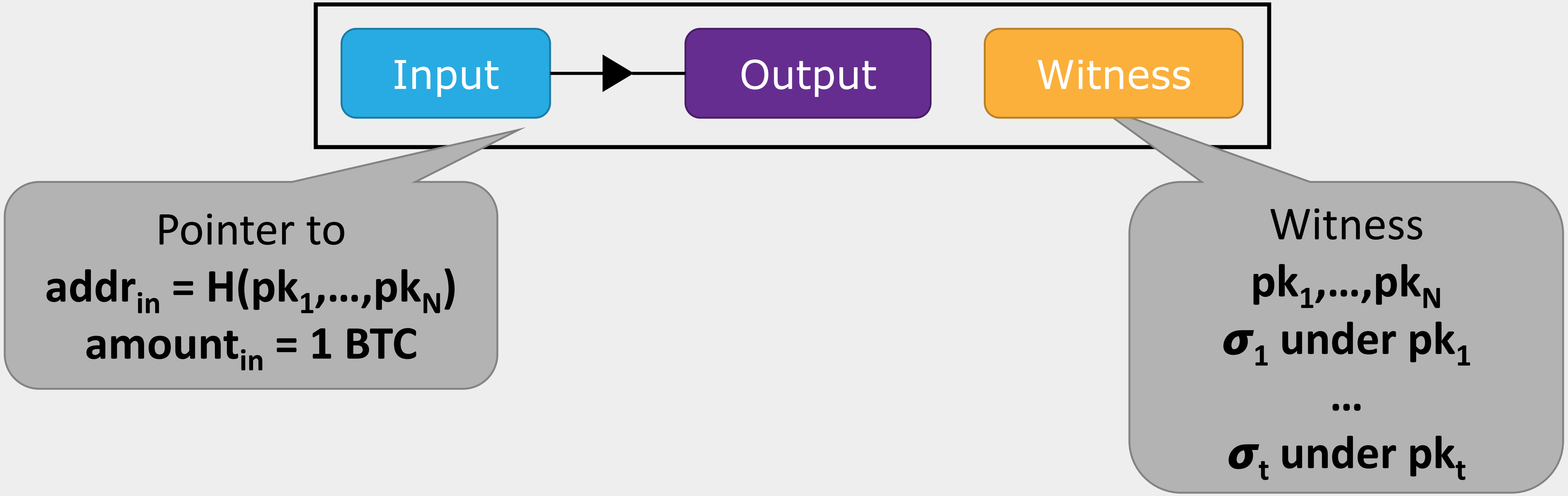
Goal: save on network/storage/verification load (currently 200GB)

more transactions per block (block size is constant)

Multi-Sig addresses

Address requiring signatures from multiple keys (t-out-of-N)

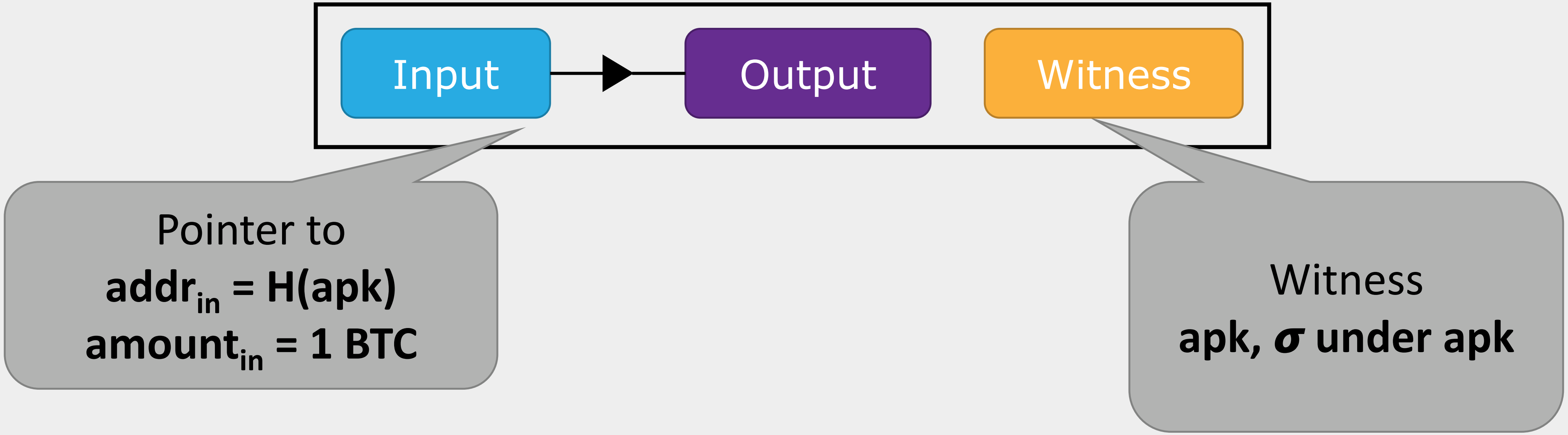
e.g., joint accounts, additional security, fair exchange/escrow



Multi-Sig addresses

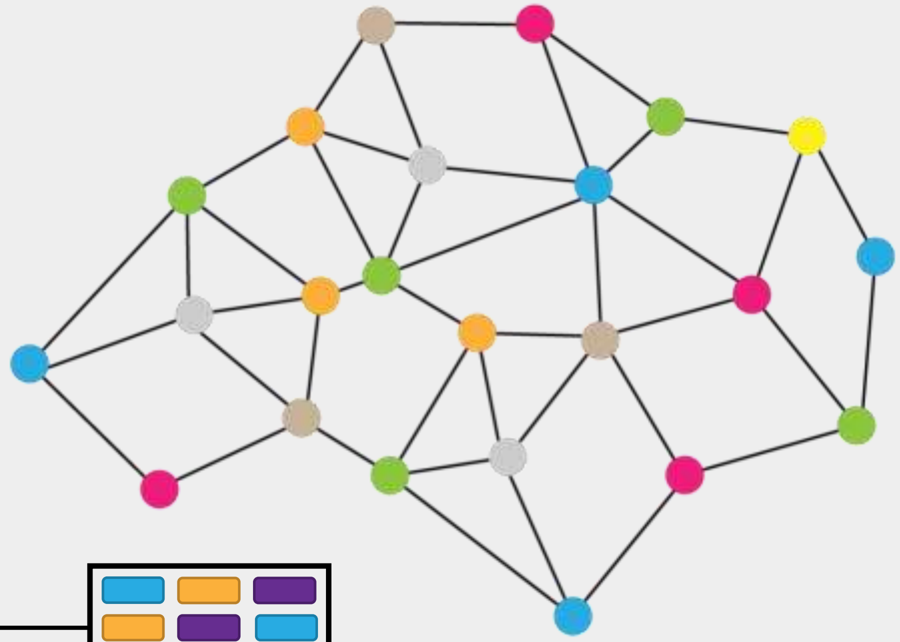
N-out-of-N case using multi-signatures

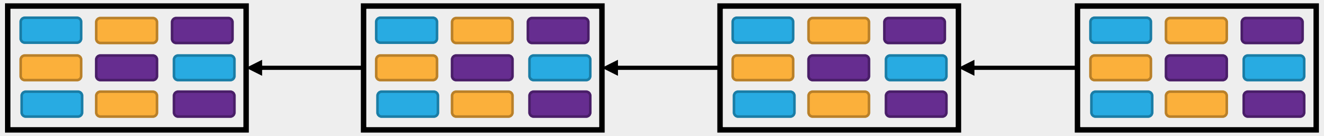
Transparent to verifier!



Applications beyond Bitcoin

- Collective signing by co-thorities (e.g., CoSi [STV+16])
- Distributed random beacons (e.g., RandHound [SJK+16])
- Notarization in blockchains
  - cryptocurrencies (e.g., ByzCoin [KJG+16])
  - distributed ledgers
    (e.g., OmniLedger [KJG+17], Ziliqa, Harmony)





Overview of this talk

- Brief history of multi-signatures
- Attacks on existing two-round schemes
- Secure schemes
- Conclusion

Brief history of multi-signatures

“Plain” Schnorr multi-signatures

$pk_1 = g^{sk1}$		$pk_2 = g^{sk2}$		$pk_3 = g^{sk3}$
$r_1 \leftarrow_R Z_q$ $t_1 \leftarrow g^{r1}$	$\leftrightarrow$	$r_2 \leftarrow_R Z_q$ $t_2 \leftarrow g^{r2}$	$\leftrightarrow$	$r_3 \leftarrow_R Z_q$ $t_3 \leftarrow g^{r3}$
$t \leftarrow t_1 \cdot t_2 \cdot t_3$ $c \leftarrow H(t,m)$		$t \leftarrow t_1 \cdot t_2 \cdot t_3$ $c \leftarrow H(t,m)$		$t \leftarrow t_1 \cdot t_2 \cdot t_3$ $c \leftarrow H(t,m)$
$s_1 \leftarrow r_1 + c \cdot sk_1 \bmod q$	$\leftrightarrow$	$s_2 \leftarrow r_2 + c \cdot sk_2 \bmod q$	$\leftrightarrow$	$s_3 \leftarrow r_3 + c \cdot sk_3 \bmod q$
$s \leftarrow s_1+s_2+s_3 \bmod q$ $\sigma \leftarrow (c, s)$		$s \leftarrow s_1+s_2+s_3 \bmod q$ $\sigma \leftarrow (c, s)$		$s \leftarrow s_1+s_2+s_3 \bmod q$ $\sigma \leftarrow (c, s)$

$$apk \leftarrow pk_1 \cdot pk_2 \cdot pk_3$$

$$\text{Check} \quad c = H(g^s \cdot apk^{-c}, m)$$

Problem 1: Rogue-key attacks

$pk_1 = g^{sk1}$

$pk_2 = g^{sk2} / pk_1$

$apk = pk_1 \cdot pk_2 = g^{sk2}$

can compute signatures under apk by himself!

Known remedies:

- Knowledge of secret key (KOSK) assumption
- Interactive key generation [MOR01]
- Per-signer challenges [BN06]
- Proofs of possession added to pk [RY07,BCJ08]
- MuSig key aggregation: $apk \leftarrow \Pi \; pk_i^{H(pki, \{pk1,\dots,pkN\})}$ [MPSW18]

Problem 2: Signature simulation

pk_1 pk_2

$c, s_1 \leftarrow_R Z_q$

$t_1 \leftarrow g^{s1}\, pk_1^{-c}$ $\longrightarrow t_1$

$t \leftarrow t_1 \cdot t_2$ $\longleftarrow t_2$

$c \leftarrow H(t,m)$

cannot program random oracle,
because adversary knows t before simulator does

Multi-signatures from discrete logarithms

Scheme	Rounds	Rogue keys	Signature simulation
MOR [MOR01]	2	interactive key generation	sequential attacks only
BN [BN06]	3	per-signer challenges	preliminary round $H(t_i)$
~~BCJ-1 [BCJ08]~~	~~2~~	~~per-signer challenges~~	~~homomorphic equivocable (HE) commitments~~
~~BCJ-2 [BCJ08]~~	~~2~~	~~proofs of possession~~	
~~MWLD [MWLD10]~~	~~2~~	~~per signer challenges~~	~~witness indinstinguishable keys~~
~~CoSi [STV+16]~~	~~2~~	~~proofs of possession~~	~~(no security proof)~~
~~MuSig-1 [MPSW18a]~~	~~2~~	~~MuSig key aggregation~~	~~DL oracle in one-more DL assumption~~
mBCJ [this work]	2	proofs of possession	per-message HE commitments
BDN-DL, MuSig-2 [BDN18, MPSW18b]	3	MuSig key aggregation	preliminary round $H(t_i)$
BDN-DLpop [BDN18]	3	proofs of possession	preliminary round $H(t_i)$
BLS [Bol03,RY07]	1	KOSK / proofs of possession	pairings
BDN-P [BDN18]	1	MuSig key aggregation	pairings

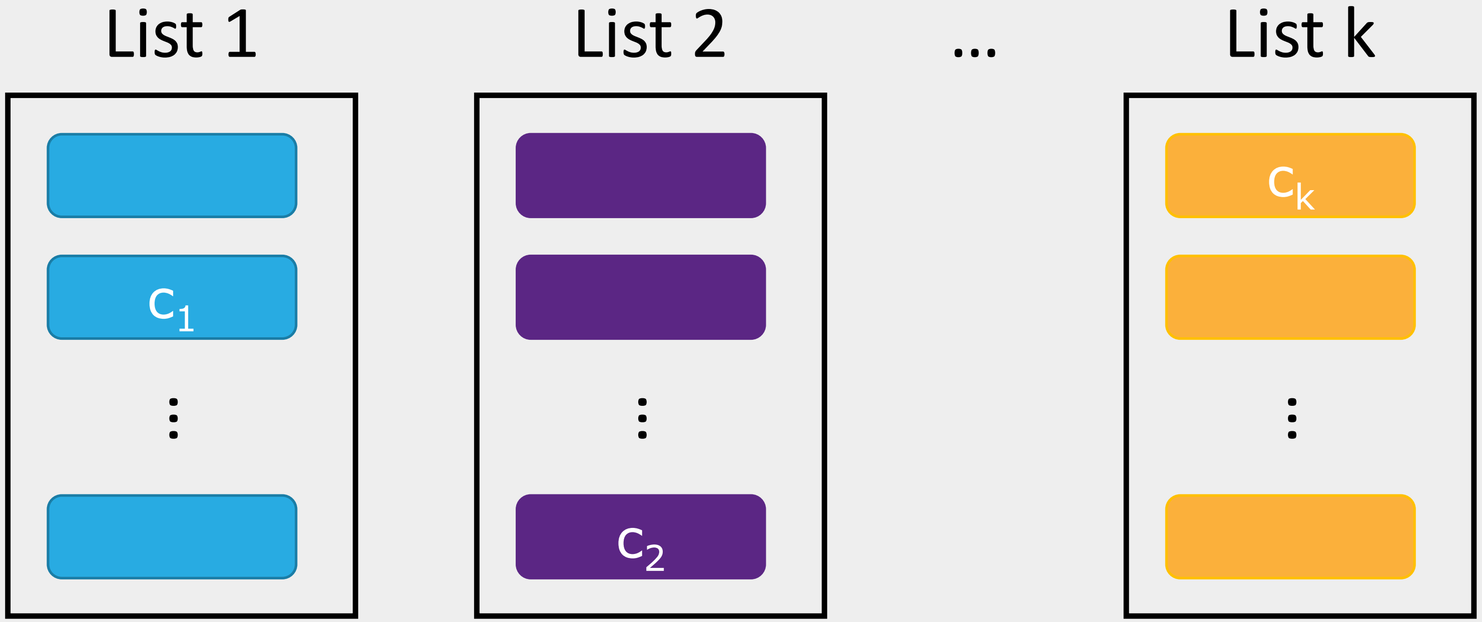
Attacks and non-provability

Wagner's generalized birthday attack [W02]

k-sum problem in Z_q:

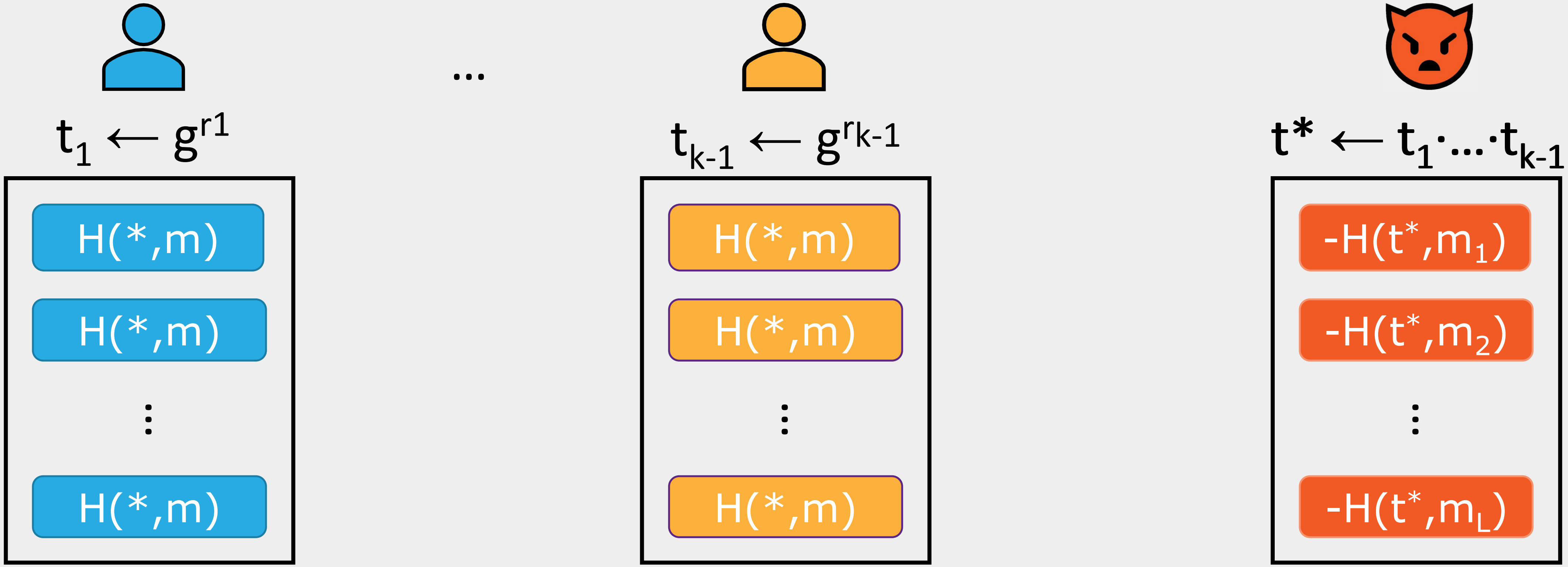
Given k lists of random elements in Z_q

Find $(c_1,\ldots,c_k)$ in lists such that $c_1 + \ldots + c_k = 0 \bmod q$



Subexponential solution: Solved for $k = 2^{\sqrt{n}}$ in time $O(2^{2\sqrt{n}})$ where $n = |q|$.

Application to “plain” Schnorr and CoSi



$s_1 \leftarrow r_1 + c_1 \cdot sk^* \bmod q$

$s_{k-1} \leftarrow r_{k-1} + c_{k-1} \cdot sk^* \bmod q$

$c_1 + \ldots + c_{k-1} = c^* \bmod q$

$s^* \leftarrow s_1 + \ldots + s_{k-1} \bmod q$

$$pk^* = g^{sk^*}$$

$$g^{s^*} = g^{\Sigma si} = g^{\Sigma ri + \Sigma ci \cdot sk^*} = \Pi t_i \cdot pk^{*c^*} = t \cdot pk^{*c^*}$$

Attacks on two-round multi-signature schemes

- Attack applies to all previously* known two-round schemes
  - BCJ-1 and BCJ-2
  - MWLD
  - CoSi
  - MuSig-1
- Sub-exponential but practical

  (for 256-bit q)
  - 15 parallel signing queries: 2^{62} steps
  - 127 parallel signing queries: 2^{45} steps
- Prevented by increasing |q|

  …any hope for provable security?

* before first version of this paper

Non-provability of two-round schemes

Theorem: One-more discrete logarithm problem is hard

$\Downarrow$

BCJ/MWLD/CoSi/MuSig-1 cannot be proved secure
under one-more discrete logarithm

(through algebraic black-box reductions in random-oracle model)

Essentially excludes all known proof techniques (including rewinding)
under likely assumptions.

Subtle flaws in proofs of BCJ/MWLD/MuSig-1
(CoSi was never proved secure)

Secure schemes

Modified BCJ multi-signatures

$pk_i = g^{sk_i}$ + PoP

KAgg: Check PoPs, $apk \leftarrow \Pi pk_i$

$(g_2, h_1, h_2) \leftarrow H'(m)$

$r, \alpha_1, \alpha_2 \leftarrow_R Z_q$

$t_{i,1} \leftarrow g_1^{\alpha 1} h_1^{\alpha 2}$

$t_{i,2} \leftarrow g_2^{\alpha 1} h_2^{\alpha 2} g_1^{r}$

$t_{i,1}, t_{i,2}$ ↔

$t_1 \leftarrow \Pi t_{i,1}$; $t_2 \leftarrow \Pi t_{i,2}$

$c \leftarrow H(t_1, t_2, \Pi pk_i, m)$

$s_i \leftarrow r + c \cdot sk_i + \Sigma s_i \bmod q$

$s_i, \alpha_{i,1}, \alpha_{i,2}$ ↔

$s \leftarrow \Sigma s_i \bmod q$

$\alpha_1 \leftarrow \Sigma \alpha_{i,1} \bmod q$

$\alpha_2 \leftarrow \Sigma \alpha_{i,2} \bmod q$

$\sigma \leftarrow (t_1, t_2, s, \alpha_1, \alpha_2)$

Verify: $c \leftarrow H(t_1, t_2, apk, m)$

Check $t_1 = g_1^{\alpha 1} h_1^{\alpha 2}$

and $t_2 = g_2^{\alpha 1} h_2^{\alpha 2} g_1^{s} apk^{-c}$

Efficiency

Sign: 1 $mexp^2$ + 1 $mexp^3$

plain Schnorr: 1 exp

Verify: 3 $mexp^2$

plain Schnorr: 1 $mexp^2$

Signature size: 160 B

plain Schnorr: 64 B

Large-scale deployment of mBCJ

- 16,384 signers generate signature within 2 seconds
- 20% bandwidth increase, 75% computation increase

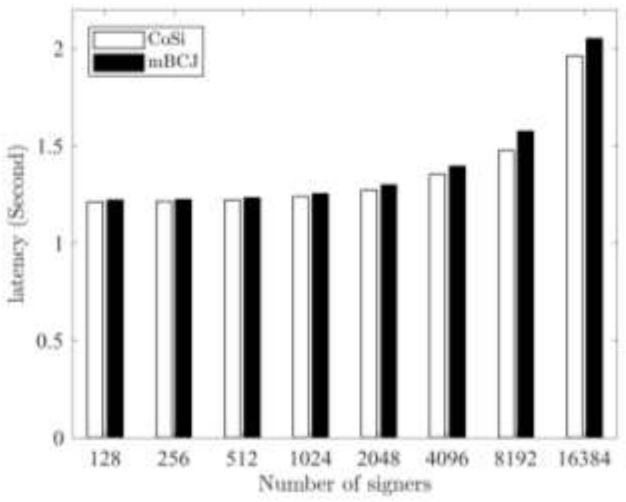


Fig. 4. Comparing end-to-end latency of CoSi and mBCJ signing with varying amounts of signers.

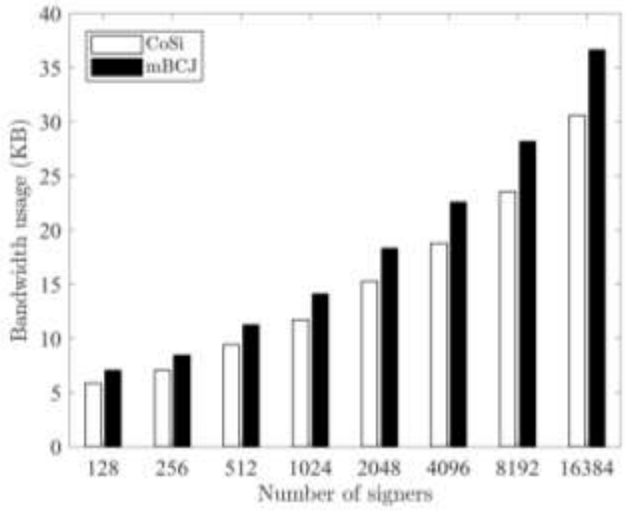


Fig. 5. Bandwidth consumption (sent and received combined) of CoSi and mBCJ with varying amounts of signers.

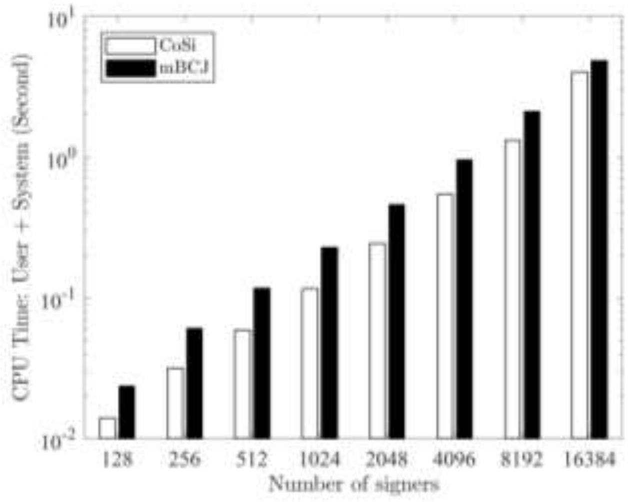


Fig. 6. CPU time (User + System) of CoSi and mBCJ with varying amounts of signers.

Other secure schemes

- Three-round scheme [BDN18, MPSW18b]
  (most likely fix for BitCoin)

- Non-interactive scheme from BLS (pairings) [BLS01,Bol03,RY07,BDN18]
  (fix for RandHound/Omniledger and Harmony)

Lessons learned

Lessons learned

- Provable security! 🤔
- Review security proofs! 🤔

- Proofs can be subtle, especially forking
- Tool support for checking proofs?
- Don’t drop steps that look like they’re “just to make the proof work”

- Provable security is not perfect, but best tool we have

Thank you!

ia.cr/2018/417

DFINITY