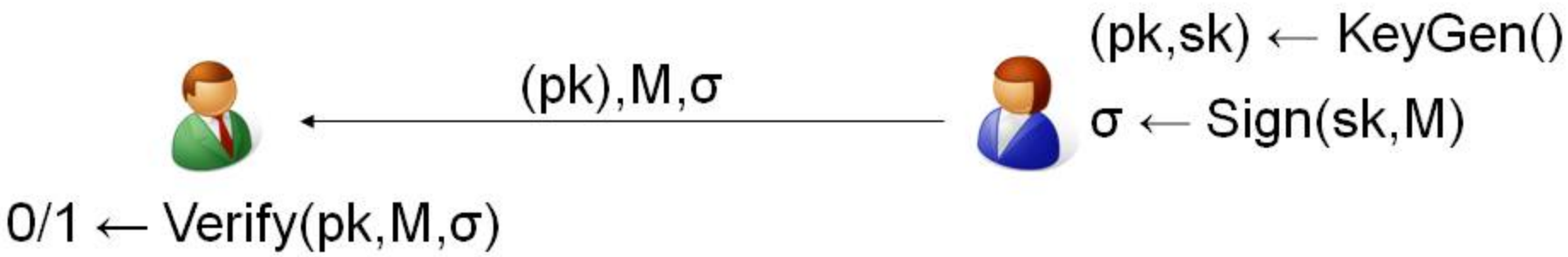
Efficient Sequential Aggregate Signed Data

Gregory Neven

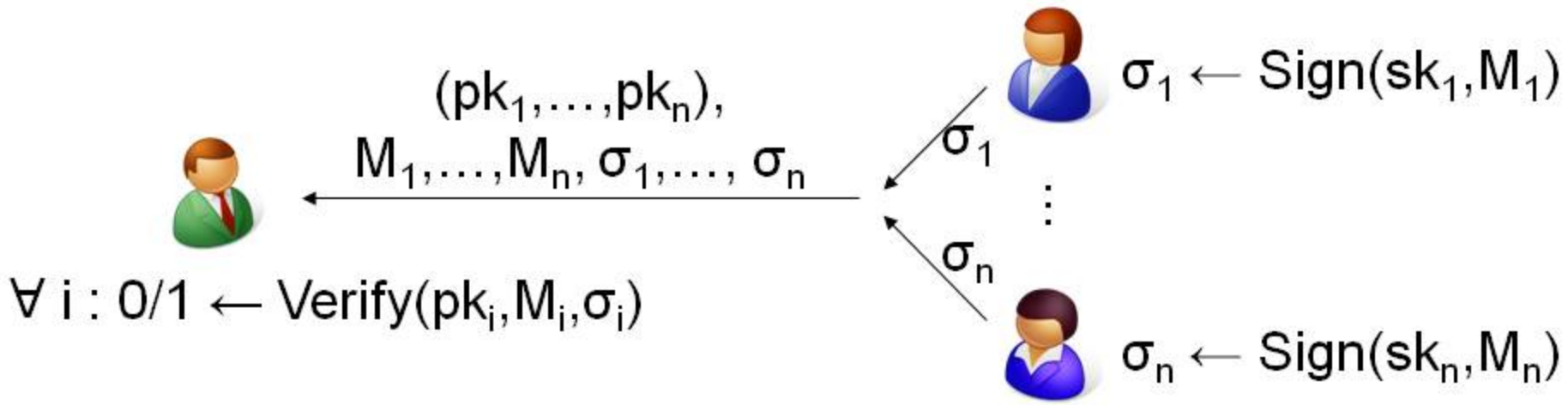
IBM Zurich Research Laboratory

work done while at K.U.Leuven

Digital signatures

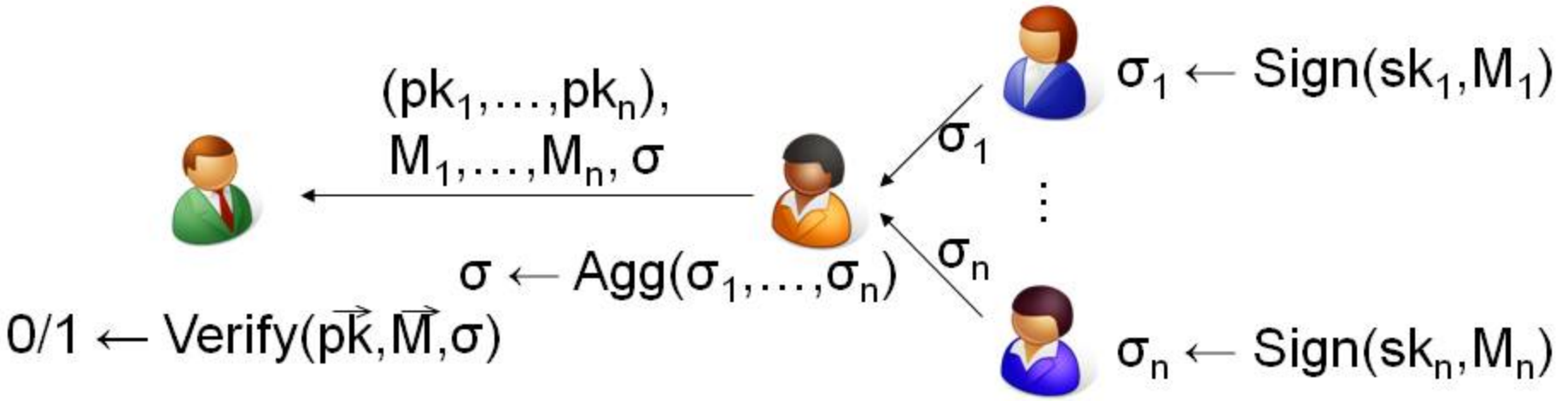


Digital signatures



Aggregate signatures (AS)

[BGLS03]

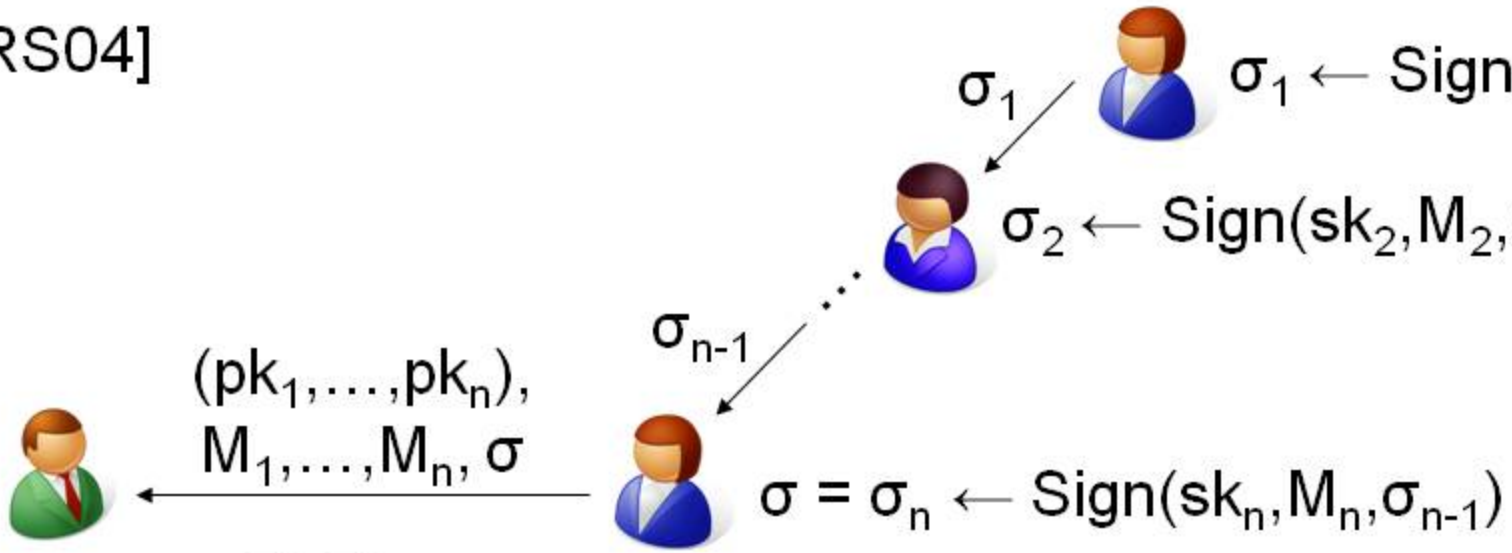


Goal: $|\sigma| < |\sigma_1| + \ldots + |\sigma_n|$, preferably constant

Motivation: certificate chains
secure routing protocols
save bandwidth (= battery life) for wireless devices

Sequential aggregate signatures (SAS)

[LMRS04]



Goal: $|\sigma| < |\sigma_1| + \ldots + |\sigma_n|$, preferably constant

Motivation: certificate chains
secure routing protocols
save bandwidth (= battery life) for wireless devices

Existing (S)AS schemes

Scheme	Type	Based on	Key model	RO
BGLS	AS	pairings	plain	Y
LMRS	SAS	RSA	plain	Y
LOSSW	SAS	pairings	KoSK	N

Drawbacks of existing schemes

- Current drawbacks of pairings (BGLS, LOSSW)
  - trust in assumptions vs. factoring, RSA
  - no standardization
  - implementations
- Rather inefficient verification (BGLS, LMRS)
  - BGLS: n pairings
  - LMRS: **certified** claw-free trapdoor permutations

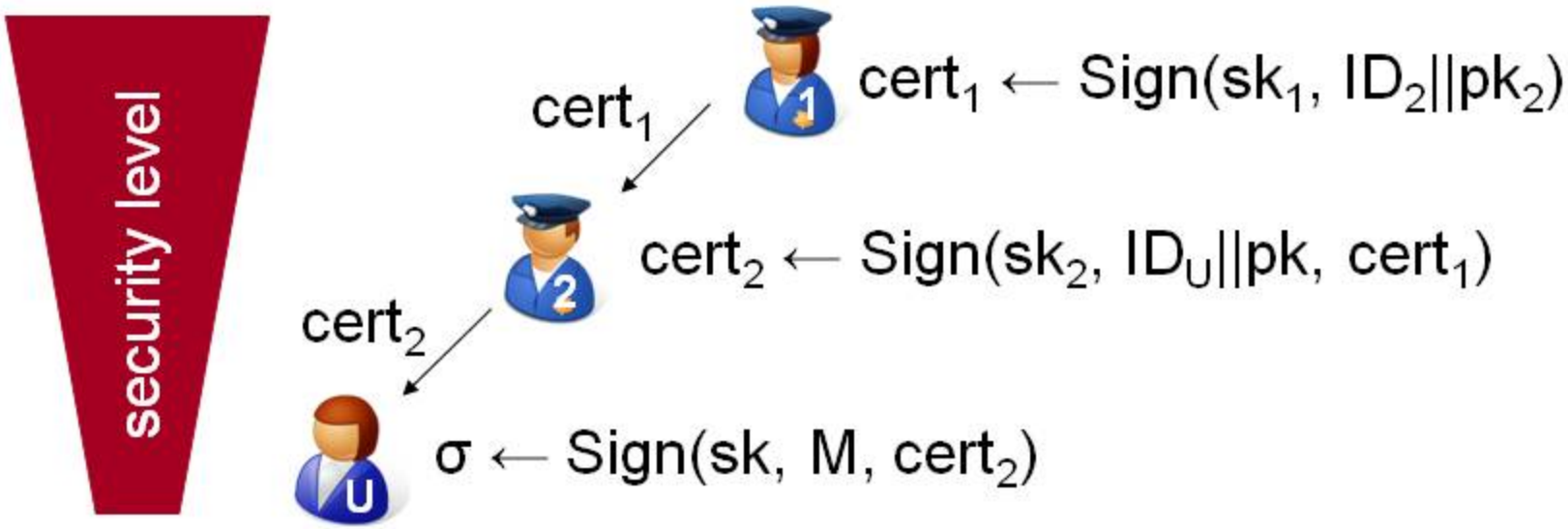
    instantiation from RSA requires $e > N$

    $\rightarrow$ verification = signing = n full-length exps
- Weak key setup model (LOSSW)

  plain public-key vs. knowledge of secret key (KOSK)

Drawbacks of existing schemes

- Security parameter flexibility (BGLS, LMRS, LOSSW)
  e.g. certificate chains



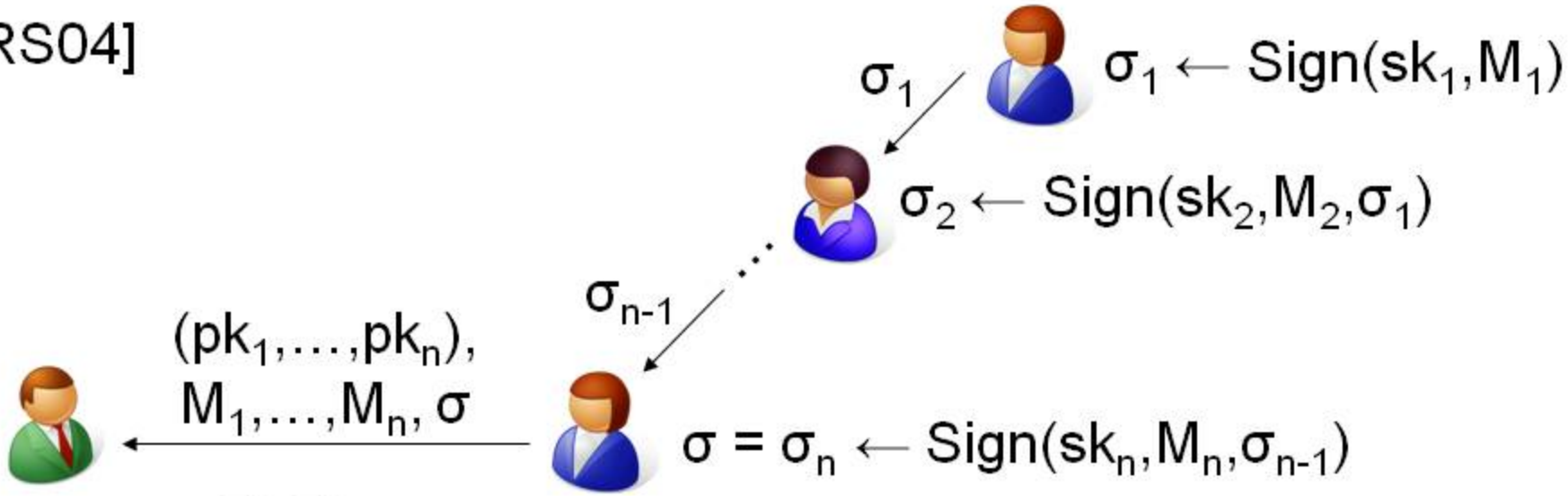
  - BGLS, LOSSW: no flexibility whatsoever
  - LMRS: increasing modulus size only
    → exact opposite of what we need
- No (S)AS schemes for currently existing keys/certificates!

Our contributions

- Generalization of SAS to SASD
- SASD scheme with
  - instantations from low-exponent RSA and factoring
  - efficient signing (1 exp + O(n) mult) and verification (O(n) mult)
  - full flexibility in modulus size
  - compatible with existing RSA/Rabin keys and certificates
- Pure SAS scheme with same properties
- Generalization of multi-signatures to multi-signed data (MSD)
- Non-interactive MSD scheme from RSA and factoring (no pairings)

Sequential aggregate signatures

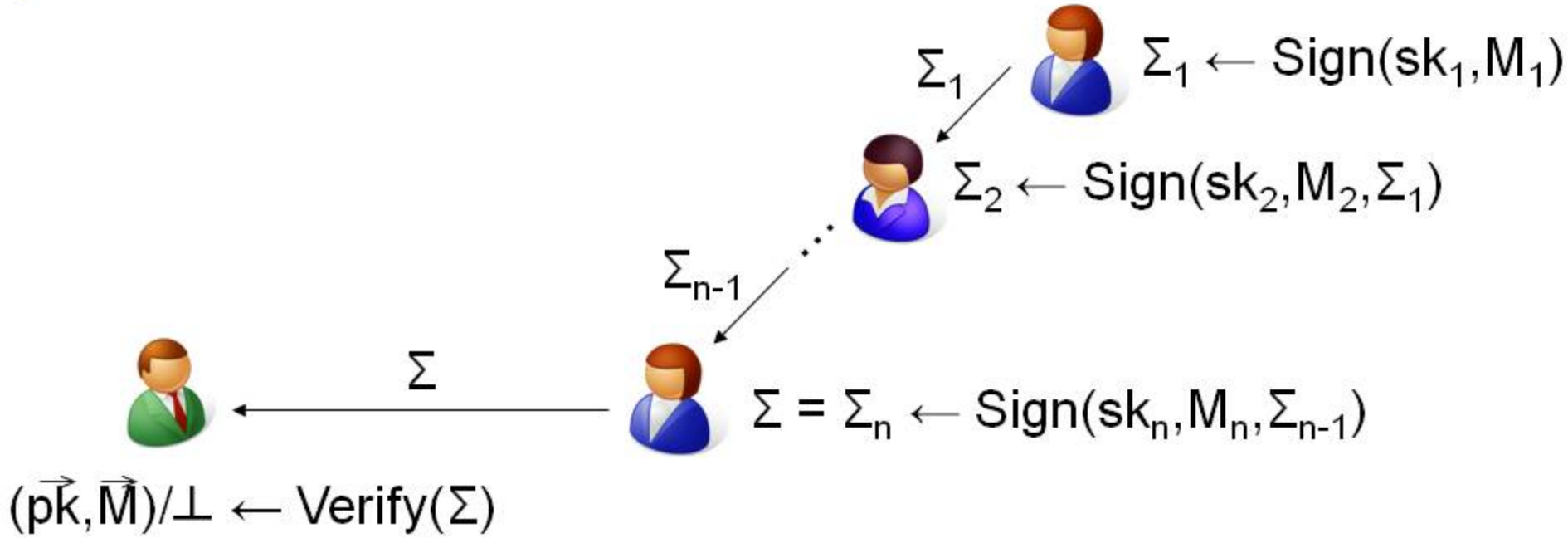
[LMRS04]



$0/1 \leftarrow \mathrm{Verify}(\vec{pk},\vec{M},\sigma)$

Goal: $|\sigma| < |\sigma_1| + \ldots + |\sigma_n|$

Sequential aggregate signed data (SASD)

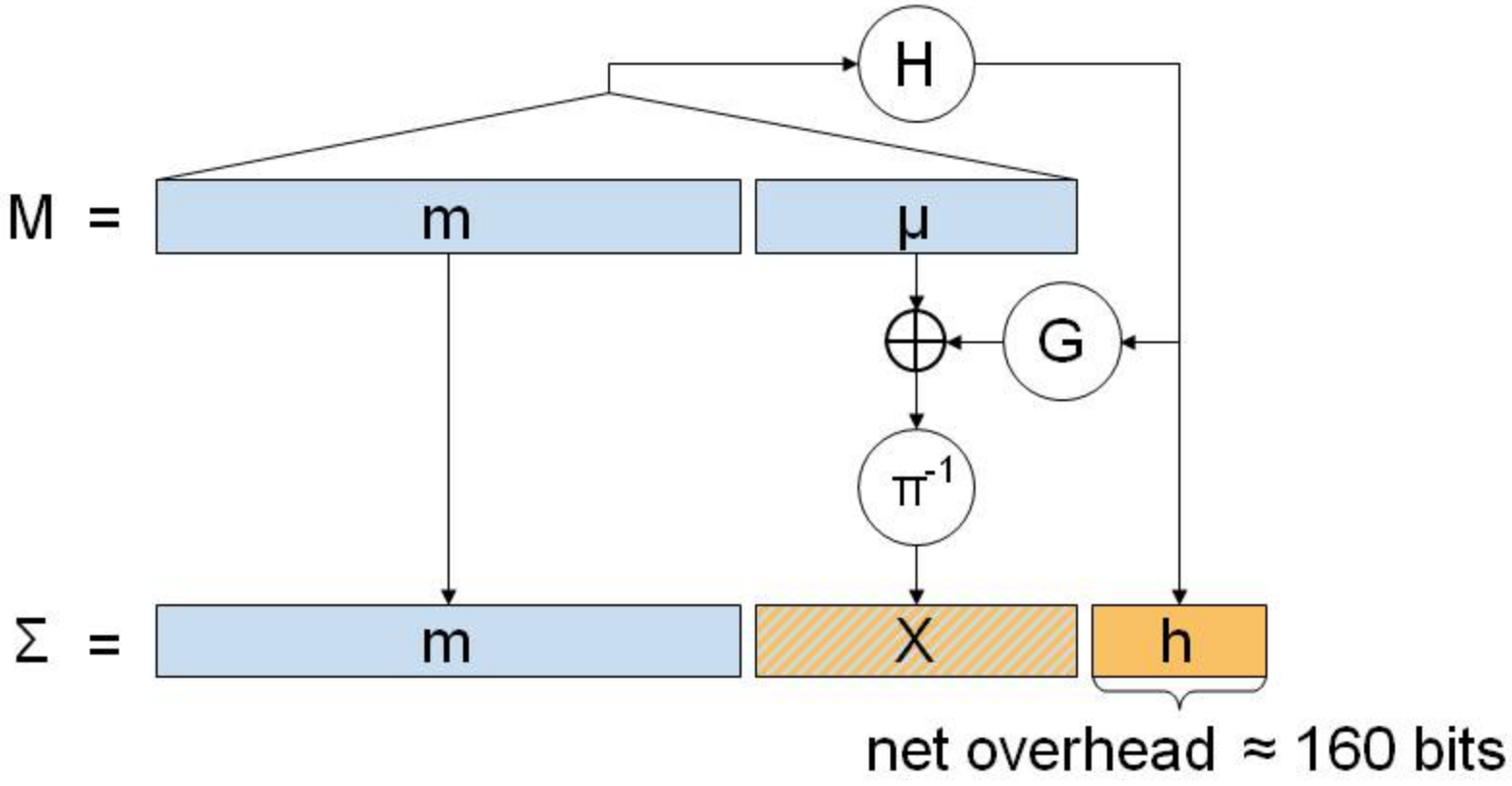


Goal: minimize "net overhead" $|\Sigma| - |M_1| - \ldots - |M_n|$

SASD scheme intuition

Step 1. Full-domain hash with message recovery

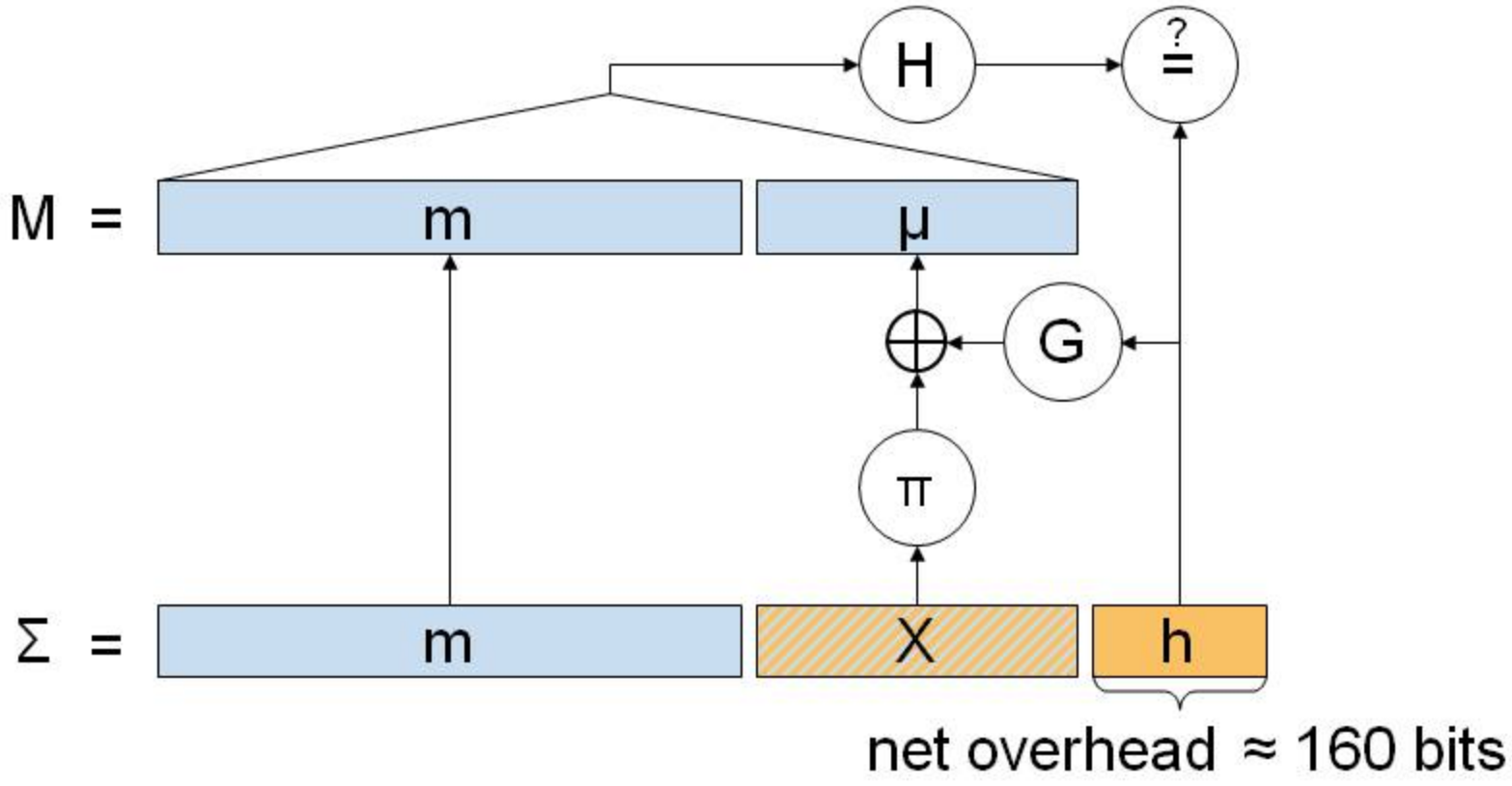
Trapdoor permutation π, message $M = m||\mu$



SASD scheme intuition

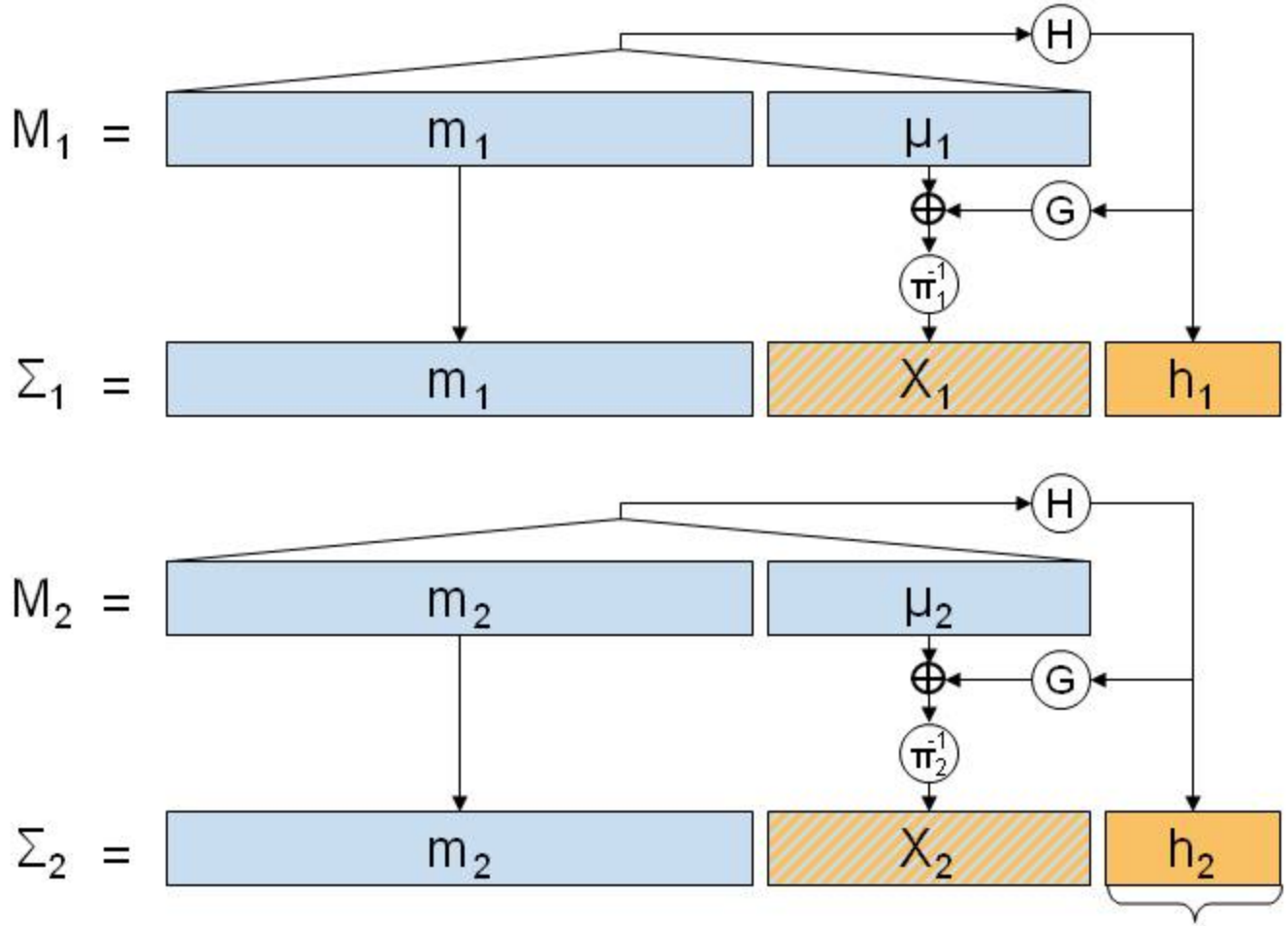
Step 1. Full-domain hash with message recovery

Trapdoor permutation π, message M = m||μ



SASD scheme intuition

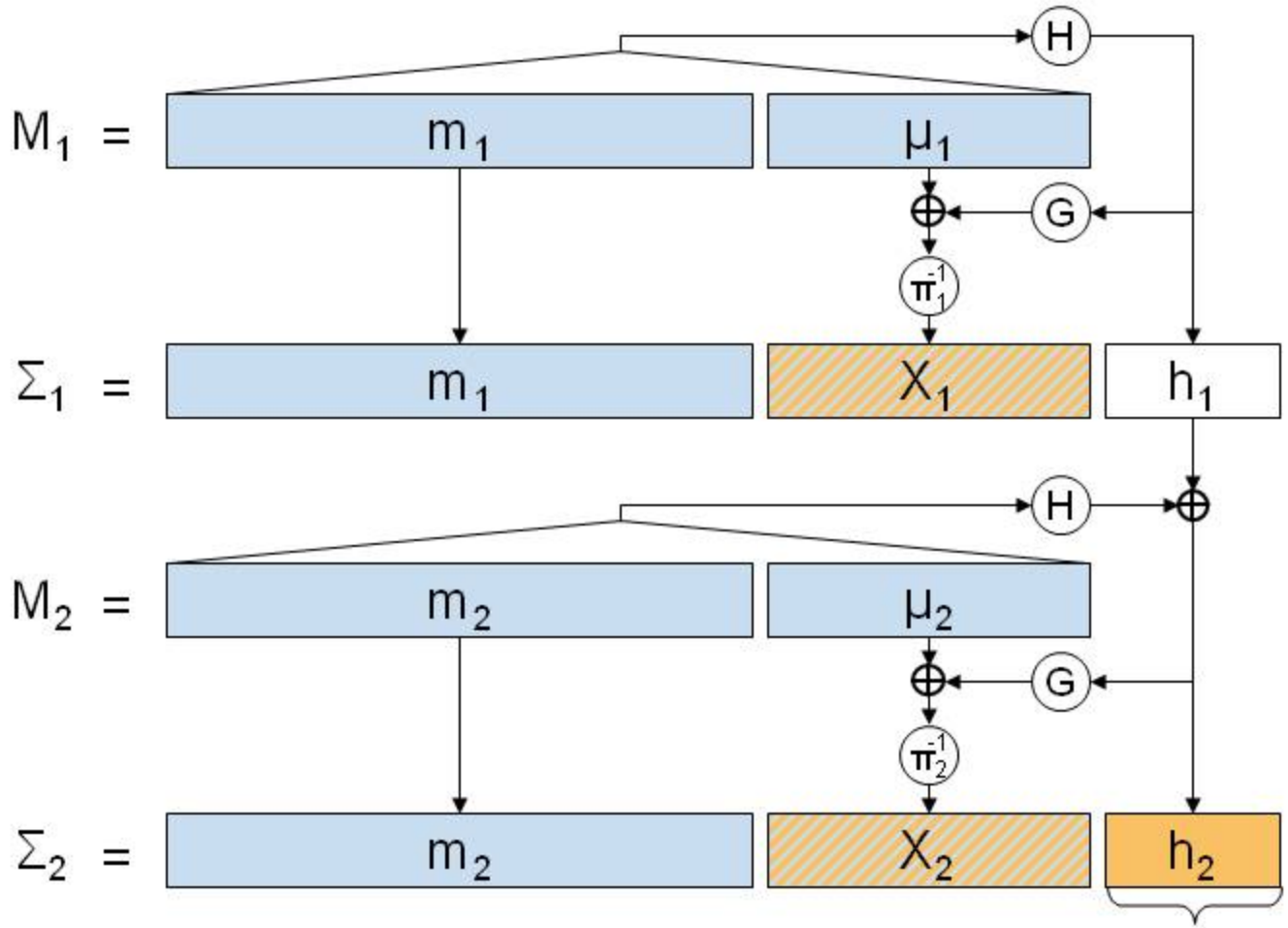
Step 2. Aggregating the hashes



net overhead ≈ 2×160 = 320 bits

SASD scheme intuition

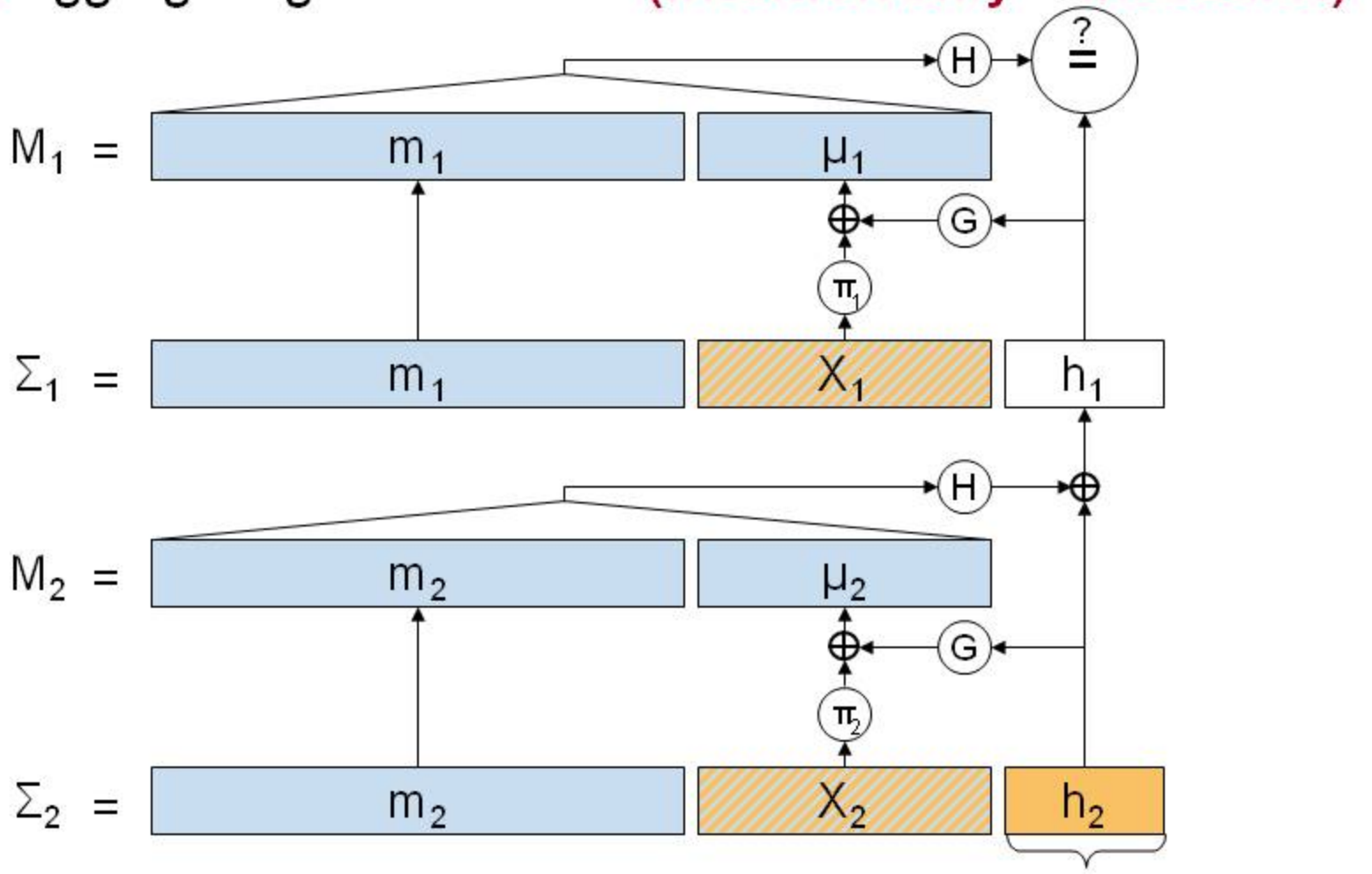
Step 2. Aggregating the hashes **(intuition only – insecure!)**



net overhead ≈ 160 bits

SASD scheme intuition

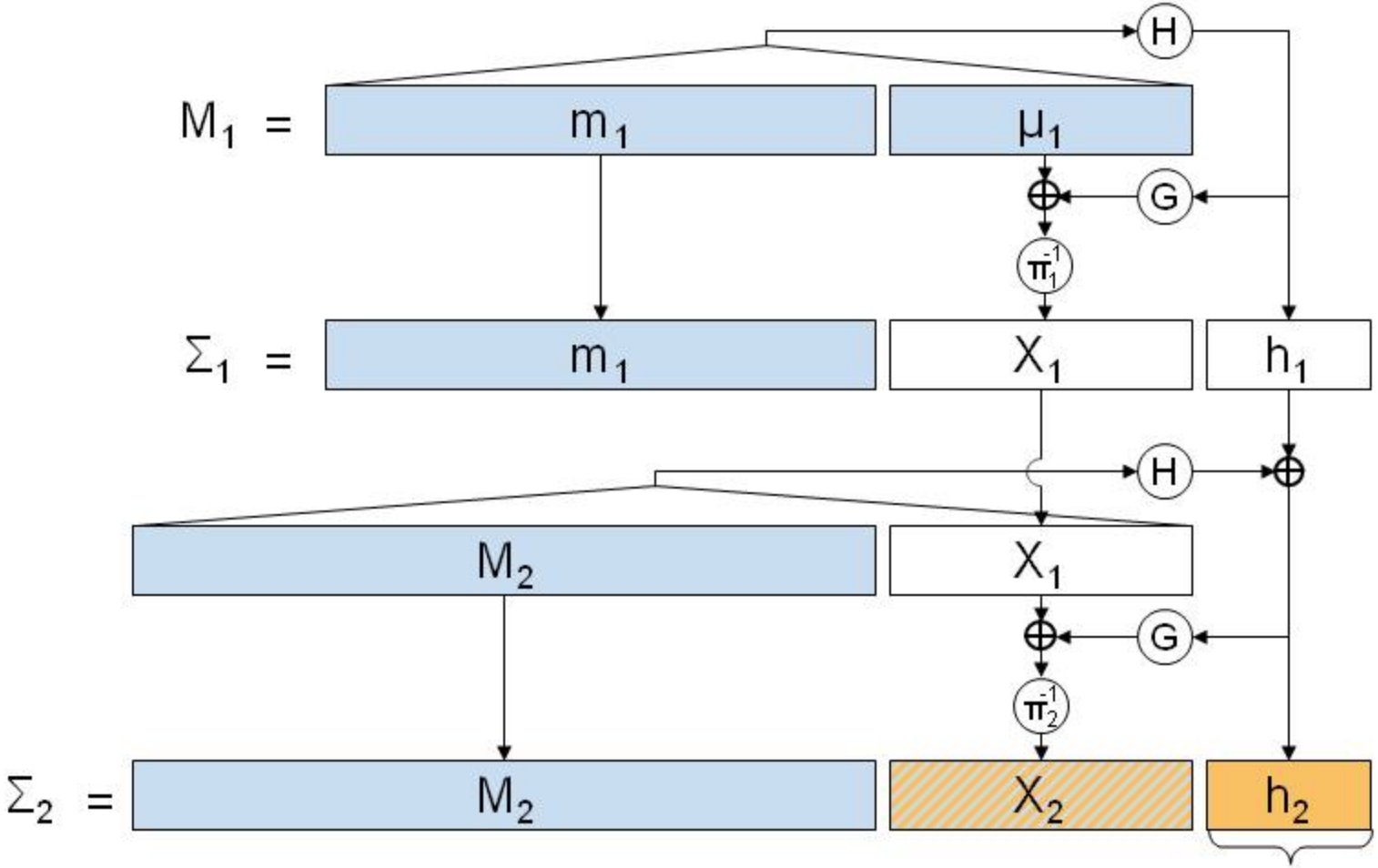
Step 2. Aggregating the hashes **(intuition only – insecure!)**



net overhead ≈ 160 bits

SASD scheme intuition

Step 3. Recovering any type of data **(intuition only – insecure!)**



net overhead ≈ 160 bits

The SASD scheme

Step 4. Getting the details right: see paper.

Theorem. If there exists a forger that $(t,q_S,q_H,q_G,n,\varepsilon)$-breaks SASD in the random oracle model, then there exists an algorithm that (t',ε')-finds a claw in Π, where

$$\varepsilon' \geq \frac{\varepsilon}{e(q_S+1)} - \frac{4(q_H+q_G+2n_{max}(q_S+1))^2}{2^L}$$

$$t' \leq t + (1/d+2)(q_H+2n_{max}(q_S+1)+n_{max})\cdot t_\pi$$

Comparison of SAS(D) schemes

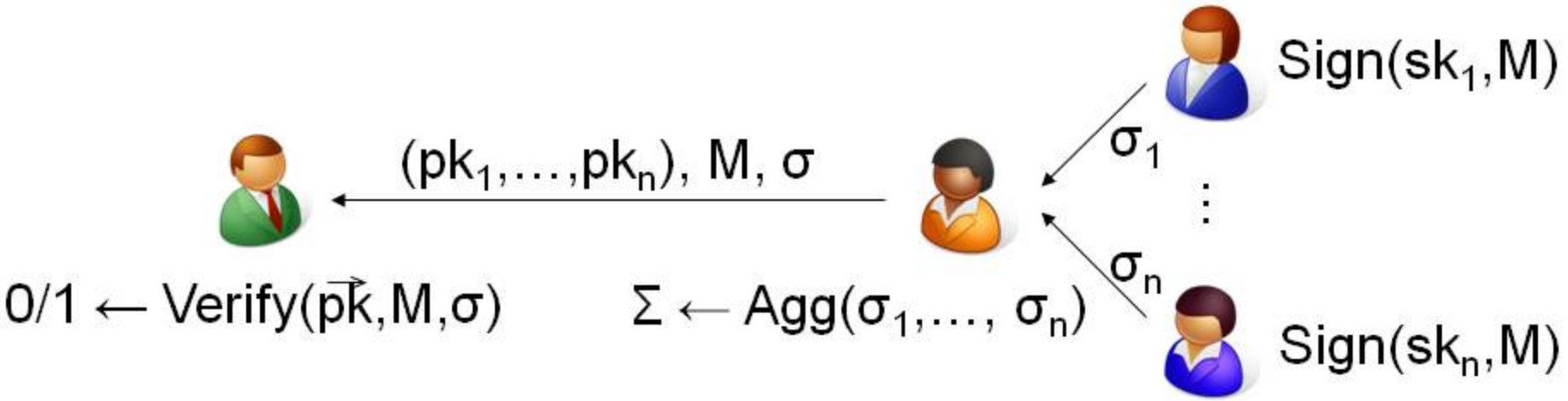
Scheme	Based on	Overhead (– $\lvert\vec{pk}\rvert$)	Sign	Verify
BGLS	pairings	160	1 E	n P
LOSSW	pairings	320	2 P + 160n M	2 P + 160n M
LMRS	RSA	1024	n E	n E
SASD	RSA, factoring	160…1184	1 E + 2n M	2n M
SAS	RSA, factoring	1184	1 E + 2n M	2n M

P = pairing E = exponentiation M = multiplication

n = #signatures in aggregate

Non-interactive multi-signatures (MS)

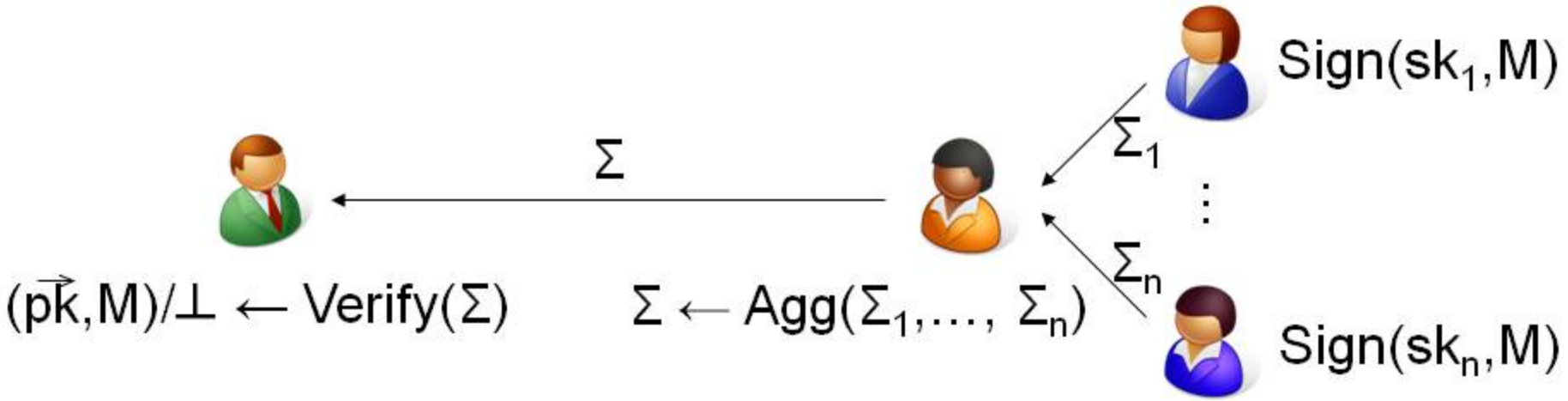
n signatures on same message M



Goal: $|\sigma| < |\sigma_1| + \ldots + |\sigma_n|$

Non-interactive multi-signed data (MSD)

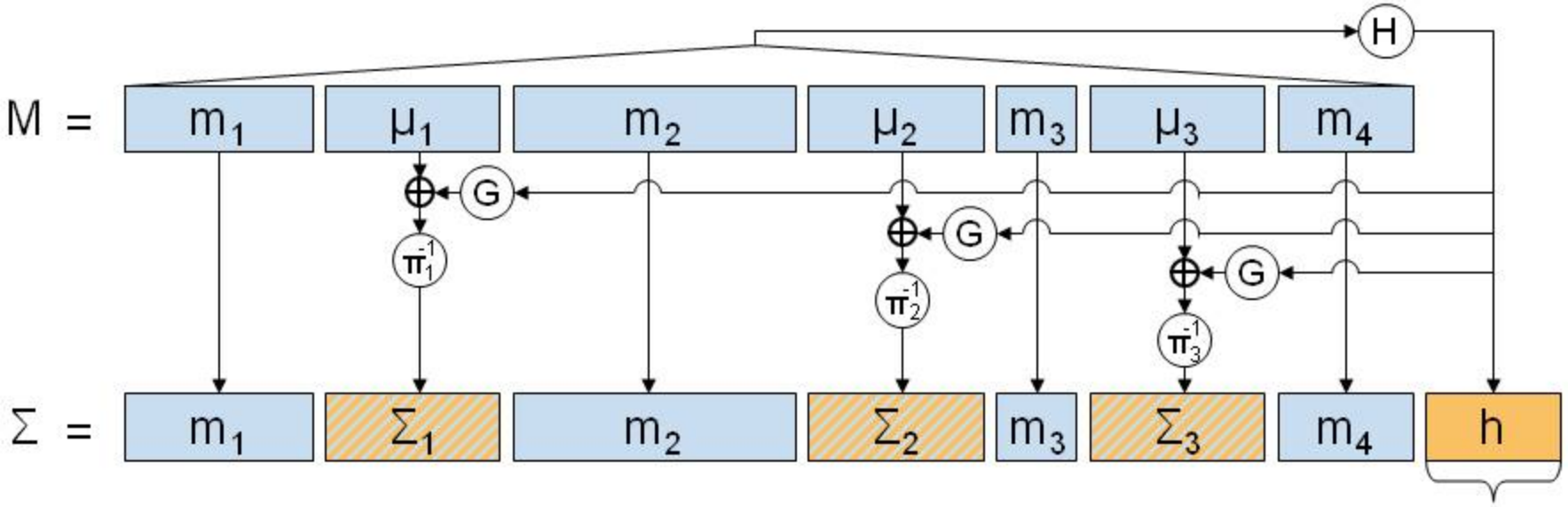
n signatures on same message M



Goal: minimize "net overhead" $|\Sigma| - |M|$

MSD scheme

Each partial signature contains part of M



net overhead ≈ 160 bits

Who takes which part of M?

- Fully non-interactive: pos = hash(π_i,M)
- Known co-signers: fixed (e.g. lexicographic) order

Comparison of MS(D) schemes

Scheme	Based on	Overhead (– $\vec{pk}$)	Sign	Verify
Bol	pairings	160	1 E	2 P + n M
LOSSW	pairings	320	2 E + 160 M	2 P + (160+n) M
MSD	RSA, factoring	160 … 1024n + 160	1 E + 2n M	2n M

P = pairing E = exponentiation M = multiplication

n = #signatures in aggregate

Closing remarks

- In summary: propose SAS, SASD, MSD schemes
  - first based on low-exponent RSA and factoring
  - outperform existing schemes in many respects
  - free choice of modulus size
  - work with existing RSA/Rabin keys
- Tight reduction using Katz-Wang, or next talk
- Full version: ePrint Report 2008/063