Post-Quantum Signatures from Secure Multiparty Computation

Thibauld Feneuil

Quantum PEPR PQ-TLS project days

June 29, 2023, Paris

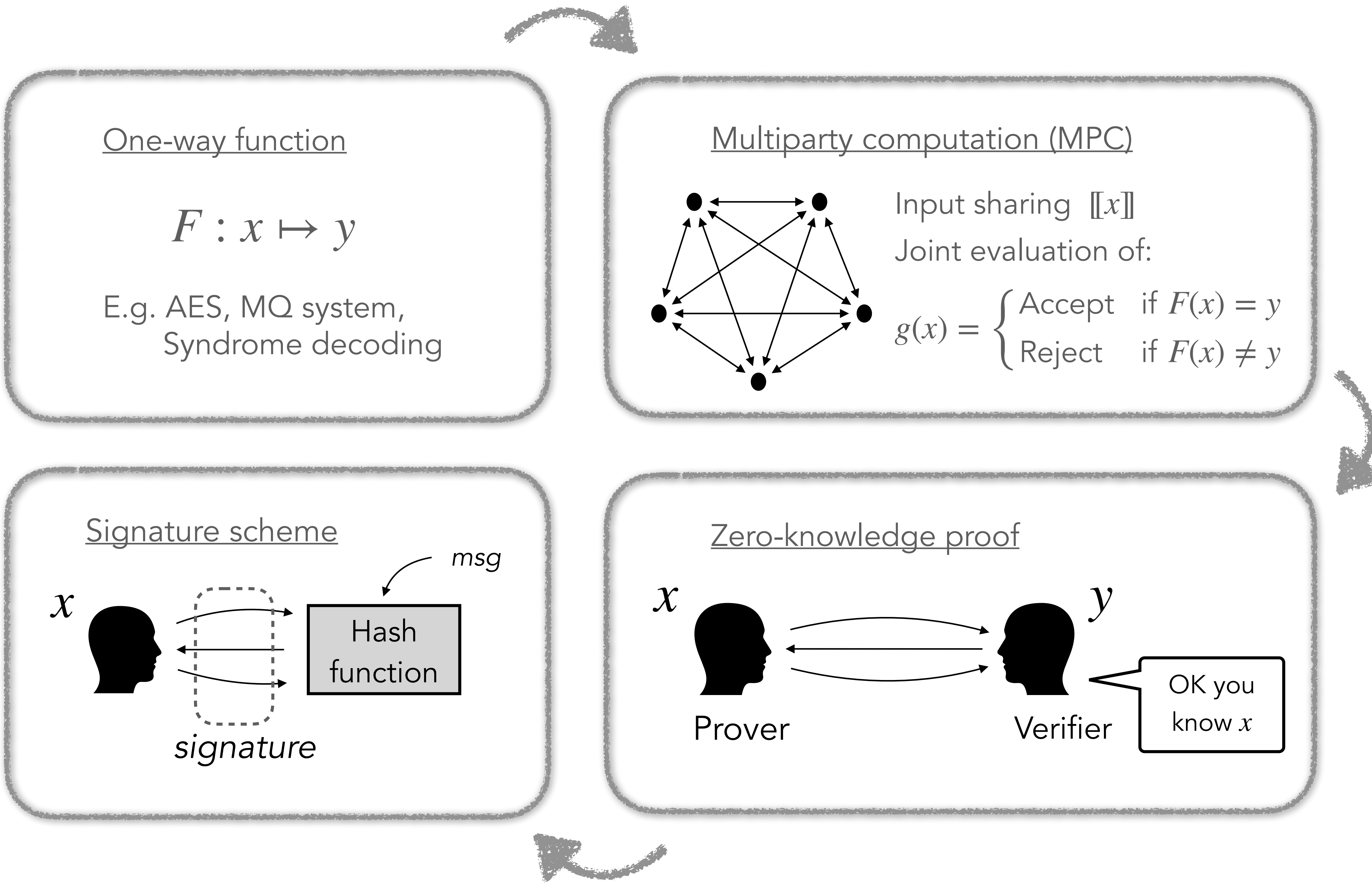
CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

Sorbonne Université

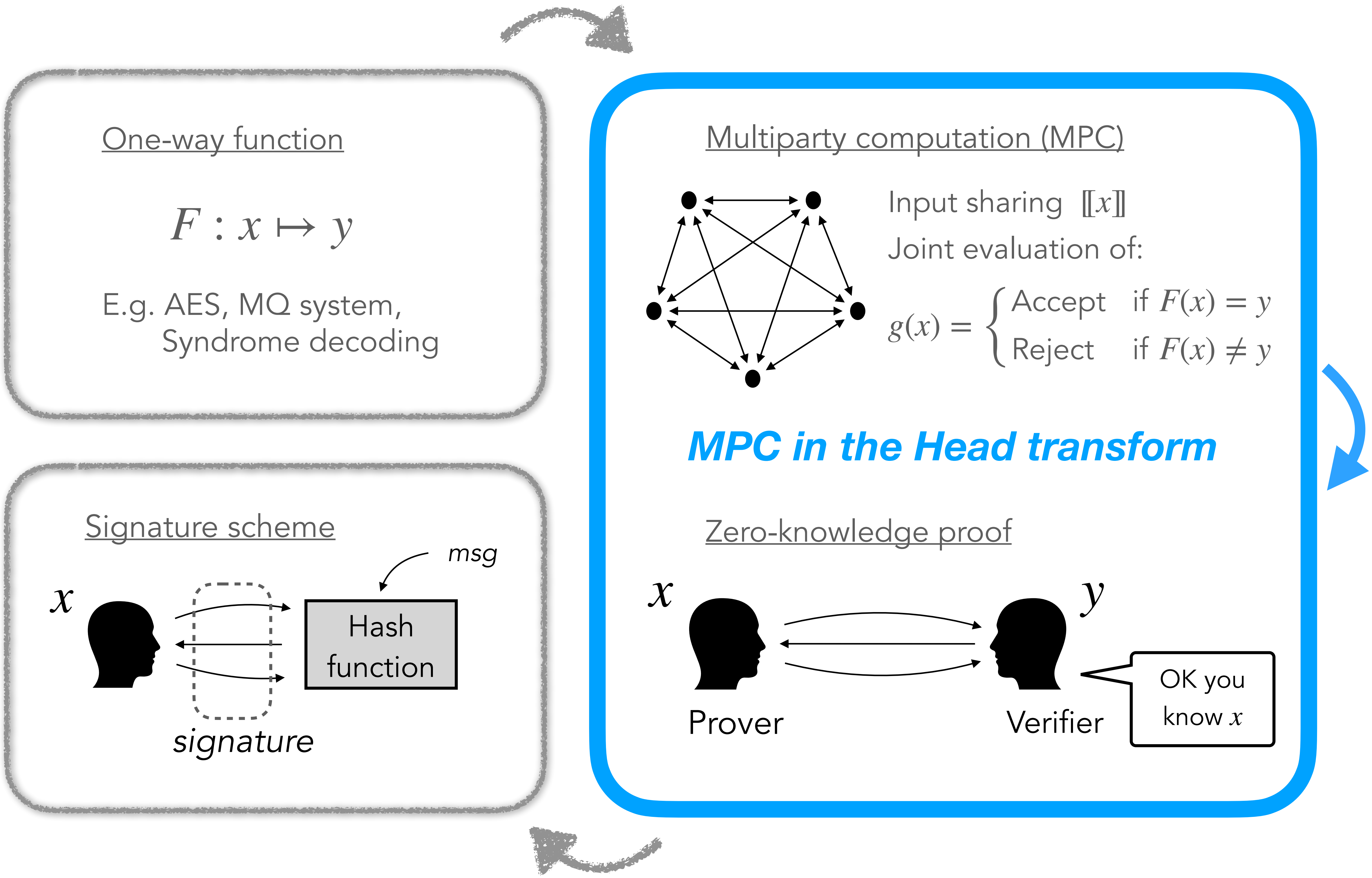
Introduction

MPC in the Head

- **[IKOS07]** Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, Amit Sahai: "Zero-knowledge from secure multiparty computation" (STOC 2007)
- Turn an MPC protocol into a zero knowledge proof of knowledge
- **Generic**: can be apply to any cryptographic problem
- Convenient to build (candidate) **post-quantum signature** schemes
- **Picnic**: submission to NIST (2017)
- Recent NIST call (01/06/2023): 7 MPCitH schemes / 50 submissions



One-way function
$F : x \mapsto y$
E.g. AES, MQ system,
Syndrome decoding
Multiparty computation (MPC)
Input sharing $[\![x]\!]$
Joint evaluation of:
$g(x) = \begin{cases} \text{Accept} & \text{if } F(x) = y \\ \text{Reject} & \text{if } F(x) \neq y \end{cases}$
Signature scheme
x
msg
Hash function
signature
Zero-knowledge proof
x
y
Prover
Verifier
OK you know x

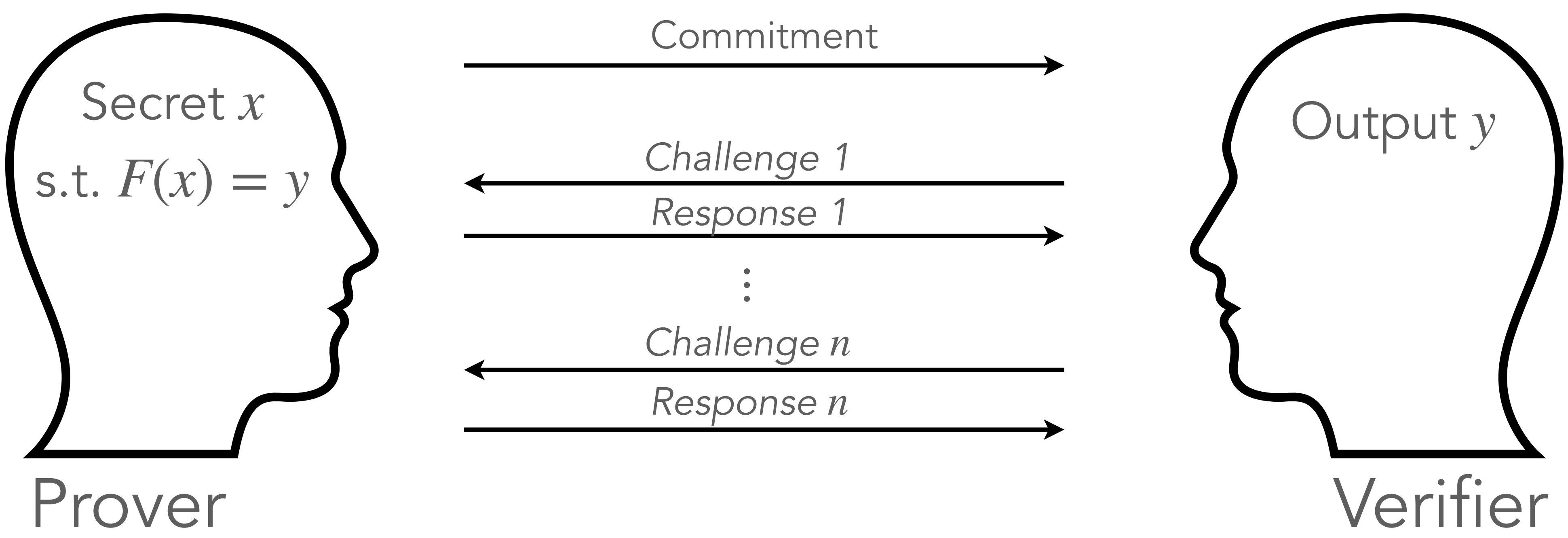

One-way function
F : x ↦ y
E.g. AES, MQ system,
Syndrome decoding
Signature scheme
msg
x
Hash
function
signature
Multiparty computation (MPC)
Input sharing ⟦x⟧
Joint evaluation of:
g(x) = Accept if F(x) = y
Reject if F(x) ≠ y
MPC in the Head transform
Zero-knowledge proof
x
y
Prover
Verifier
OK you
know x

Background: Additive secret sharing

$$[\![x]\!] = ([\![x]\!]_1, \ldots, [\![x]\!]_N) \quad \text{s.t.} \quad x = \sum_{i=1}^{N} [\![x]\!]_i$$

Any set of $N - 1$ *shares is random & independent of* x

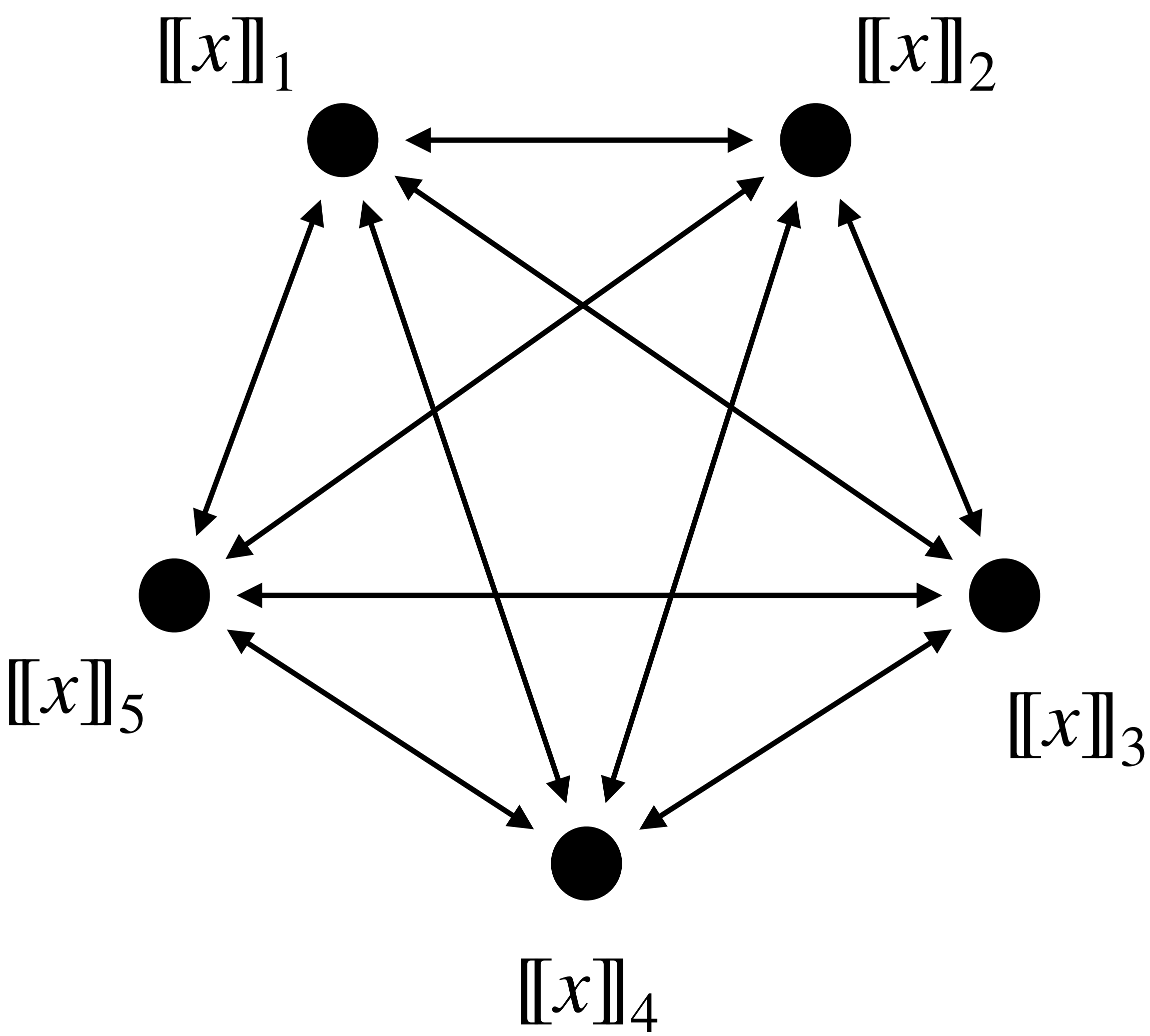
Background: Proof of knowledge



- **Completeness:** Pr[verif ✓ | honest prover] = 1
- **Soundness:** Pr[verif ✓ | malicious prover] $\leq \varepsilon$ (e.g. 2^{-128})
- **Zero-knowledge:** verifier learns nothing on x

MPCitH: general principle

MPC model

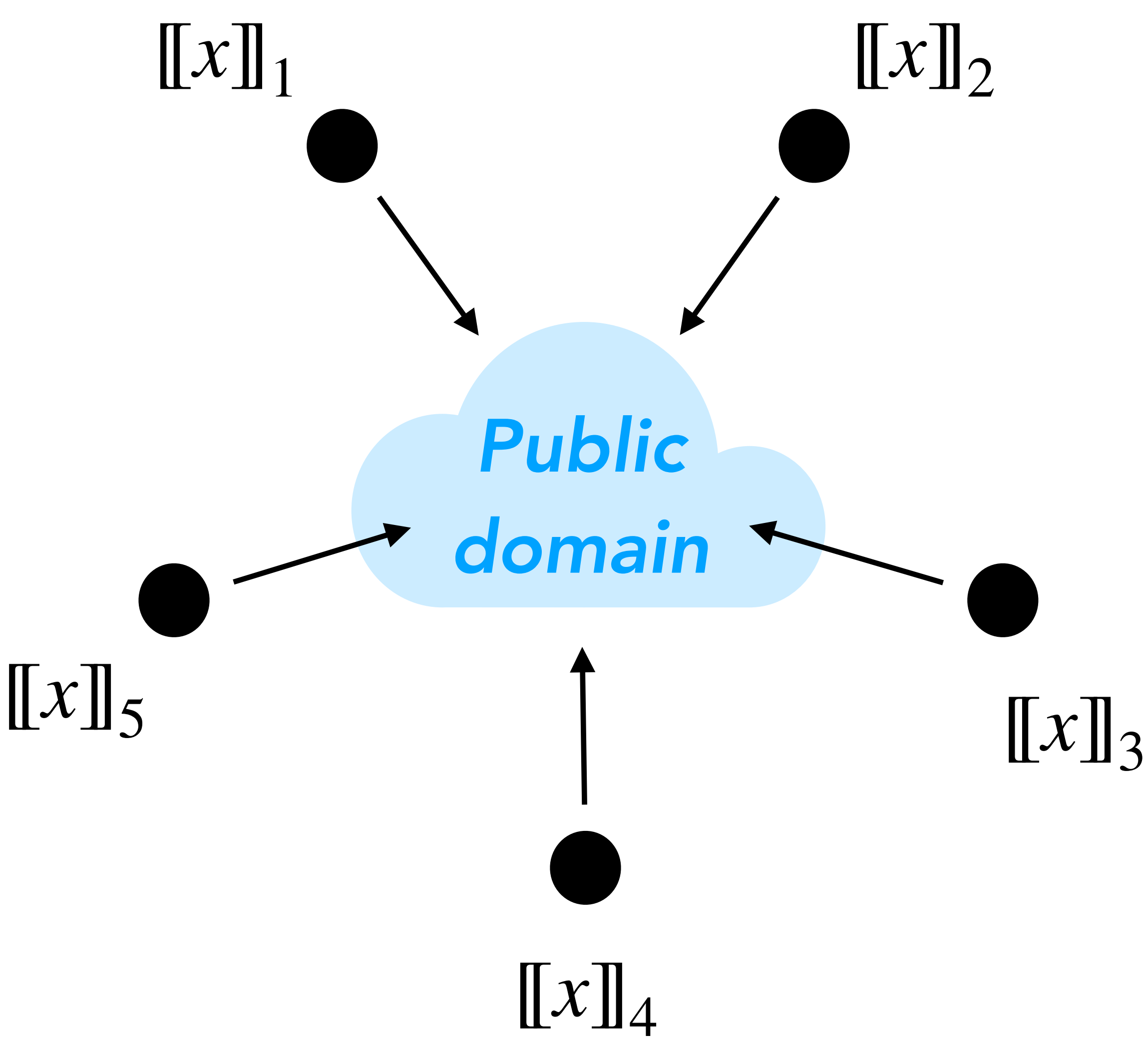


- **Jointly compute**

$$g(x) = \begin{cases} \text{Accept} & \text{if } F(x) = y \\ \text{Reject} & \text{if } F(x) \neq y \end{cases}$$

- $(N-1)$ **private:** the views of any $N-1$ parties provide no information on x
- **Semi-honest model:** assuming that the parties follow the steps of the protocol

MPC model



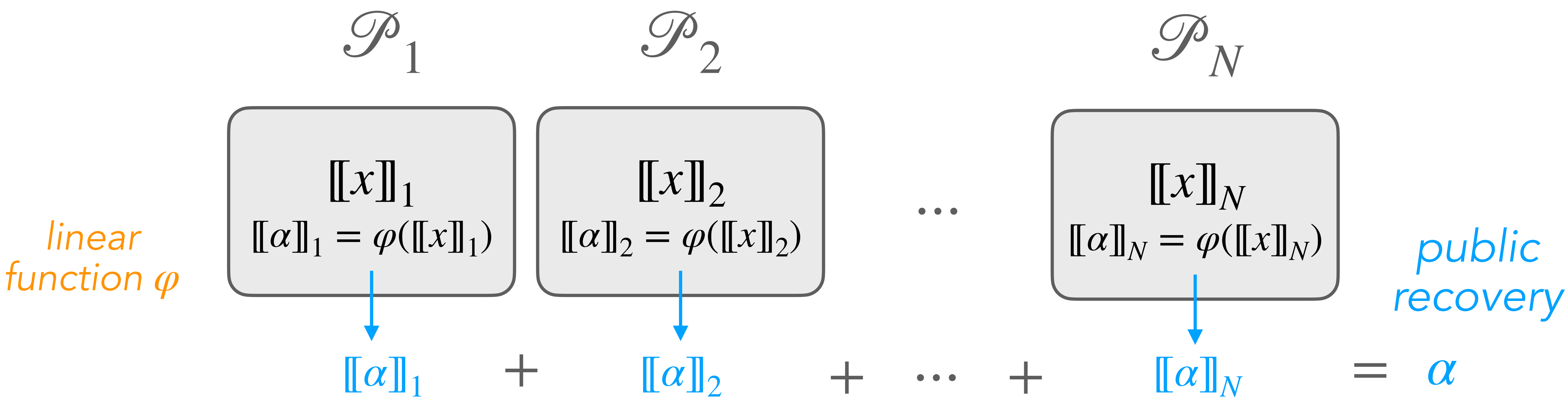
- **Jointly compute**

$$g(x) = \begin{cases} \text{Accept} & \text{if } F(x) = y \\ \text{Reject} & \text{if } F(x) \neq y \end{cases}$$

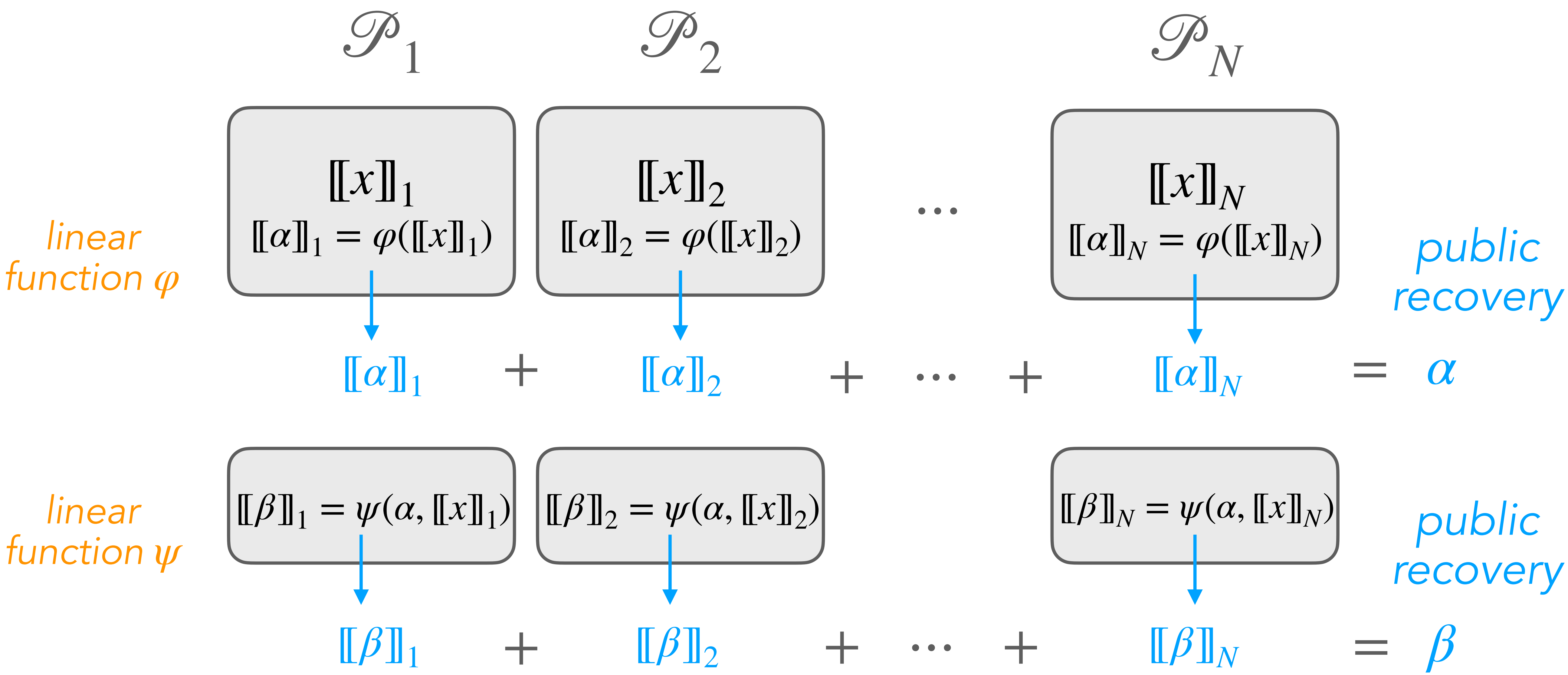
- $(N-1)$ **private:** the views of any $N-1$ parties provide no information on x

- **Semi-honest model:** assuming that the parties follow the steps of the protocol

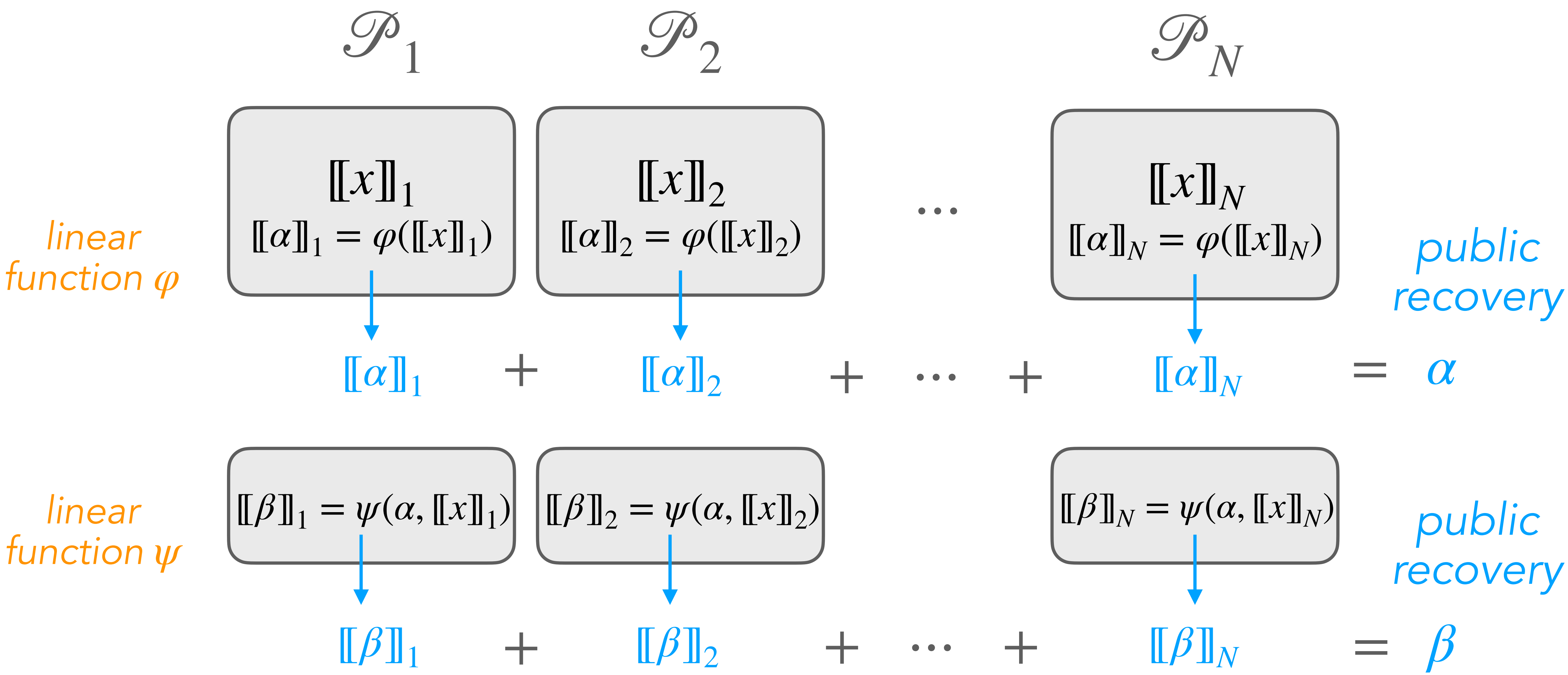
- ***Broadcast model***
  - *Parties locally compute on their shares* $[\![x]\!] \mapsto [\![\alpha]\!]$
  - *Parties broadcast* $[\![\alpha]\!]$ *and recompute* α
  - *Parties start again (now knowing* α*)*



$\mathcal{P}_1$
$\mathcal{P}_2$
$\mathcal{P}_N$
$[\![x]\!]_1$
$[\![\alpha]\!]_1 = \varphi([\![x]\!]_1)$
$[\![x]\!]_2$
$[\![\alpha]\!]_2 = \varphi([\![x]\!]_2)$
$\cdots$
$[\![x]\!]_N$
$[\![\alpha]\!]_N = \varphi([\![x]\!]_N)$
linear function φ
public recovery
$[\![\alpha]\!]_1 + [\![\alpha]\!]_2 + \cdots + [\![\alpha]\!]_N = \alpha$



$\mathscr{P}_1$
$\mathscr{P}_2$
$\mathscr{P}_N$
$[\![x]\!]_1$
$[\![\alpha]\!]_1 = \varphi([\![x]\!]_1)$
$[\![x]\!]_2$
$[\![\alpha]\!]_2 = \varphi([\![x]\!]_2)$
$\cdots$
$[\![x]\!]_N$
$[\![\alpha]\!]_N = \varphi([\![x]\!]_N)$
linear function φ
public recovery
$[\![\alpha]\!]_1 + [\![\alpha]\!]_2 + \cdots + [\![\alpha]\!]_N = \alpha$
linear function ψ
$[\![\beta]\!]_1 = \psi(\alpha, [\![x]\!]_1)$
$[\![\beta]\!]_2 = \psi(\alpha, [\![x]\!]_2)$
$[\![\beta]\!]_N = \psi(\alpha, [\![x]\!]_N)$
public recovery
$[\![\beta]\!]_1 + [\![\beta]\!]_2 + \cdots + [\![\beta]\!]_N = \beta$



and so on…

$$g : (y, \alpha, \beta, \ldots) \mapsto \begin{cases} \text{Accept} \\ \text{Reject} \end{cases}$$

Example: matrix multiplication $y = Hx$

$\mathscr{P}_1$ | $\mathscr{P}_2$ | $\cdots$ | $\mathscr{P}_N$

$[\![x]\!]_1$, $[\![\alpha]\!]_1 = H \cdot [\![x]\!]_1$

$[\![x]\!]_2$, $[\![\alpha]\!]_2 = H \cdot [\![x]\!]_2$

$[\![x]\!]_N$, $[\![\alpha]\!]_N = H \cdot [\![x]\!]_N$

mult. by H is linear

$$[\![\alpha]\!]_1 + [\![\alpha]\!]_2 + \cdots + [\![\alpha]\!]_N = \alpha$$

public recovery

$$g(y, \alpha) = \begin{cases} \text{Accept} & \text{if } y = \alpha \\ \text{Reject} & \text{if } y \neq \alpha \end{cases}$$

$$g(y, \alpha) = \text{Accept} \iff Hx = y$$

MPCitH transform

Prover

Verifier

MPCitH transform

① Generate and commit shares

$[\![x]\!] = ([\![x]\!]_1, \ldots, [\![x]\!]_N)$

$\mathrm{Com}^{\rho_1}([\![x]\!]_1)$

$\cdots$

$\mathrm{Com}^{\rho_N}([\![x]\!]_N)$

Prover

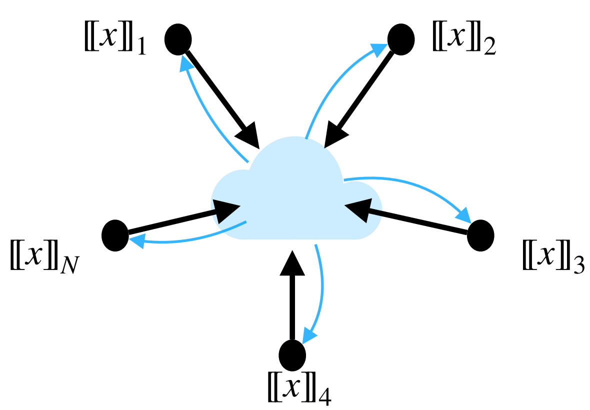
Verifier

MPCitH transform

① Generate and commit shares

$[\![x]\!] = ([\![x]\!]_1, \ldots, [\![x]\!]_N)$

② Run MPC in their head



$\mathrm{Com}^{\rho_1}([\![x]\!]_1)$

$\ldots$

$\mathrm{Com}^{\rho_N}([\![x]\!]_N)$

send broadcast

$[\![\alpha]\!]_1, \ldots, [\![\alpha]\!]_N$

Prover

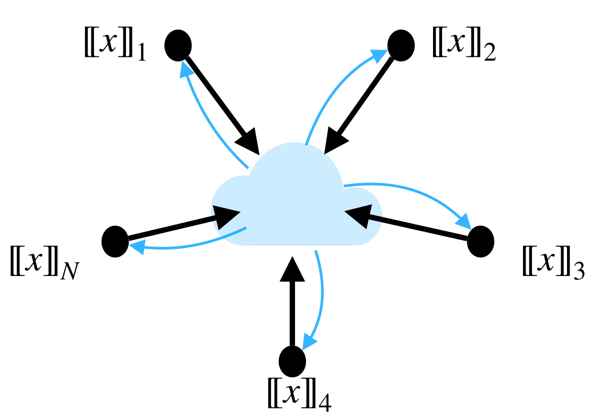
Verifier

MPCitH transform

① Generate and commit shares

$[\![x]\!] = ([\![x]\!]_1, \ldots, [\![x]\!]_N)$

② Run MPC in their head



$\mathrm{Com}^{\rho_1}([\![x]\!]_1)$

...

$\mathrm{Com}^{\rho_N}([\![x]\!]_N)$ →

send broadcast

$[\![\alpha]\!]_1, \ldots, [\![\alpha]\!]_N$ →

← i^*

③ Chose a random party

$i^* \xleftarrow{\$} \{1, \ldots, N\}$

Prover

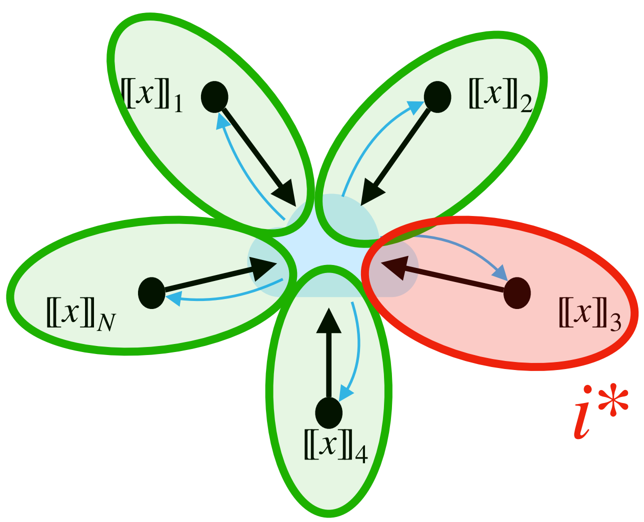
Verifier

MPCitH transform

① Generate and commit shares

$[\![x]\!] = ([\![x]\!]_1, \ldots, [\![x]\!]_N)$

② Run MPC in their head



④ Open parties $\{1,\ldots,N\}\backslash\{i^*\}$

$\mathrm{Com}^{\rho_1}([\![x]\!]_1)$

...

$\mathrm{Com}^{\rho_N}([\![x]\!]_N)$

send broadcast

$[\![\alpha]\!]_1, \ldots, [\![\alpha]\!]_N$

i^*

$([\![x]\!]_i, \rho_i)_{i \neq i^*}$

③ Chose a random party

$i^* \leftarrow^{\$} \{1,\ldots,N\}$

Prover

Verifier

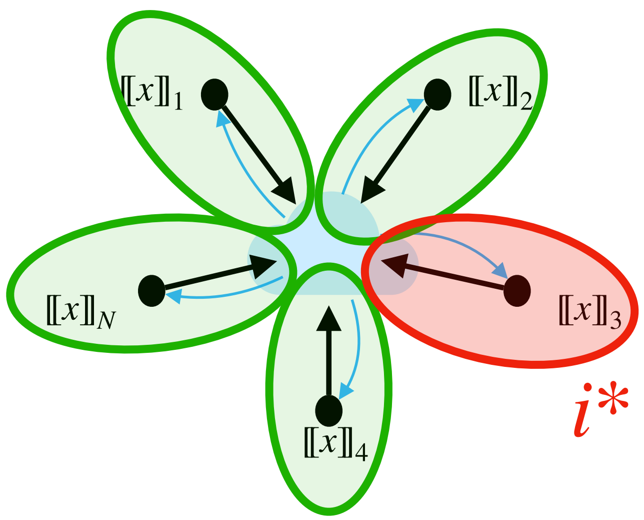
MPCitH transform

① Generate and commit shares

$[\![x]\!] = ([\![x]\!]_1, \ldots, [\![x]\!]_N)$

$\mathrm{Com}^{\rho_1}([\![x]\!]_1)$
$\cdots$
$\mathrm{Com}^{\rho_N}([\![x]\!]_N)$ →

② Run MPC in their head



send broadcast
$[\![\alpha]\!]_1, \ldots, [\![\alpha]\!]_N$ →

③ Chose a random party

$i^* \xleftarrow{\$} \{1,\ldots,N\}$

← i^*

④ Open parties $\{1,\ldots,N\}\backslash\{i^*\}$

$([\![x]\!]_i, \rho_i)_{i \neq i^*}$ →

⑤ Check $\forall i \neq i^*$

- Commitments $\mathrm{Com}^{\rho_i}([\![x]\!]_i)$
- MPC computation $[\![\alpha]\!]_i = \varphi([\![x]\!]_i)$

Check $g(y, \alpha) =$ Accept

Prover | Verifier

MPCitH transform

- **Zero-knowledge** $\iff$ MPC protocol is $(N-1)$-private

MPCitH transform

- **Zero-knowledge** $\iff$ MPC protocol is $(N-1)$-private
- **Soundness**
  - if $g(y, \alpha) \neq$ Accept → Verifier rejects
  - if $g(y, \alpha) =$ Accept, then
    - either $[\![x]\!]$ = sharing of correct witness $F(x) = y$ → Prover honest
    - or Prover has cheated for at least one party

→ Cheat undetected with proba $\frac{1}{N}$

MPCitH transform

- **Zero-knowledge** $\iff$ MPC protocol is $(N-1)$-private
- **Soundness**
  - if $g(y, \alpha) \neq$ Accept → Verifier rejects
  - if $g(y, \alpha) =$ Accept, then
    - either $[\![x]\!]$ = sharing of correct witness $F(x) = y$ → Prover honest
    - or Prover has cheated for at least one party

      → Cheat undetected with proba $\frac{1}{N}$
- **Parallel repetition**

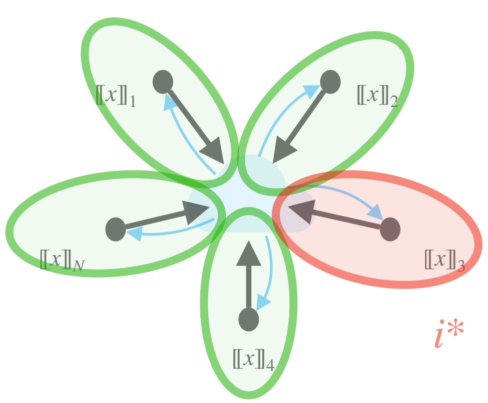
  Protocol repeated τ times in parallel → soundness error $\left(\frac{1}{N}\right)^{\tau}$

Example: matrix multiplication $y = Hx$

$\mathscr{P}_1$	$\mathscr{P}_2$		$\mathscr{P}_N$
$[\![x]\!]_1$	$[\![x]\!]_2$	$\cdots$	$[\![x]\!]_N$
$[\![\alpha]\!]_1 = H \cdot [\![x]\!]_1$	$[\![\alpha]\!]_2 = H \cdot [\![x]\!]_2$		$[\![\alpha]\!]_N = H \cdot [\![x]\!]_N$

$$[\![\alpha]\!]_1 + [\![\alpha]\!]_2 + \cdots + [\![\alpha]\!]_N = \alpha$$

Prover



$\{\mathrm{Com}^{\rho_i}([\![x]\!]_i)\}$ →

$\{[\![\alpha]\!]_i\}$ →

← i^*

$\{[\![x]\!]_i, \rho_i\}_{i \neq i^*}$ →

Verifier

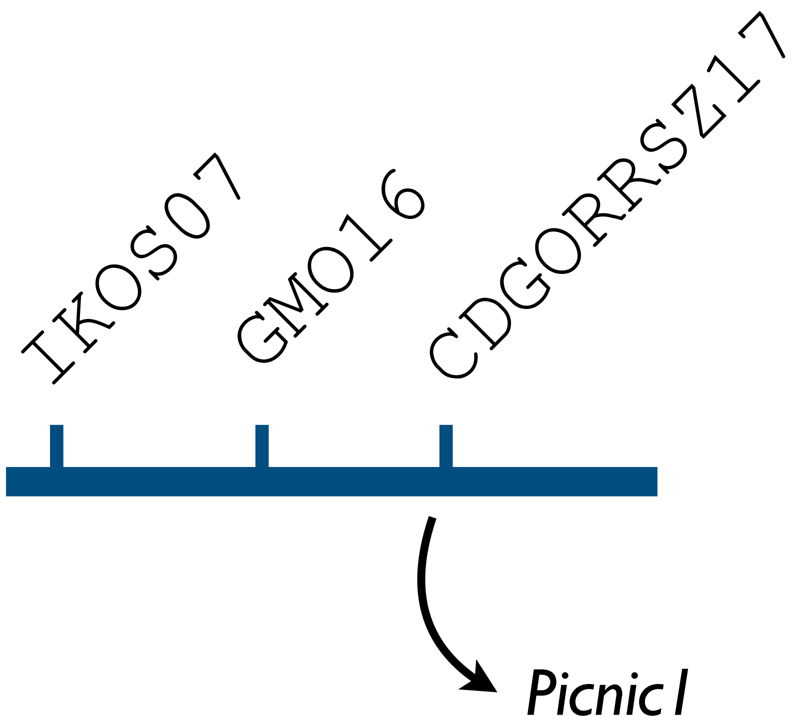
Check $\forall i \neq i^*$

- Commitments $\mathrm{Com}^{\rho_i}([\![x]\!]_i)$
- MPC computation $[\![\alpha]\!]_i = H \cdot [\![x]\!]_i$

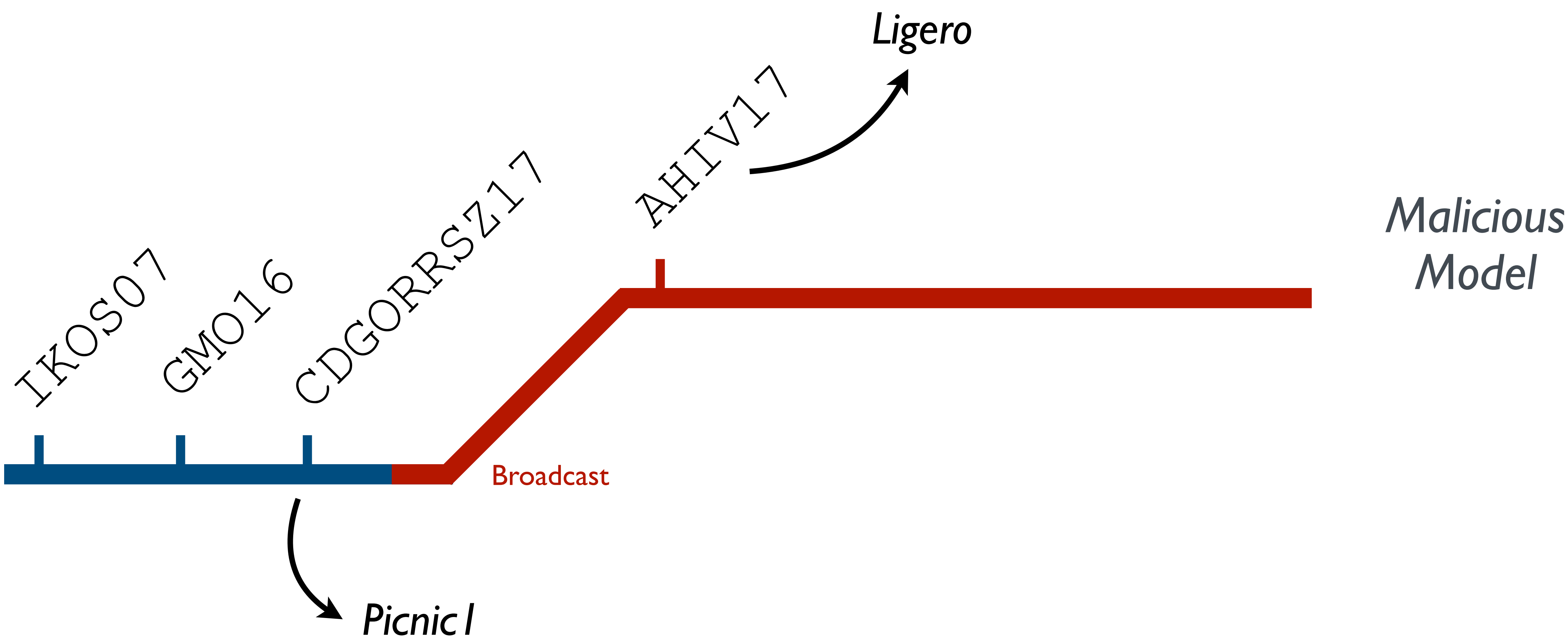
Check $\alpha := \Sigma_i [\![\alpha]\!]_i \; = \; y$

MPCitH: signature schemes

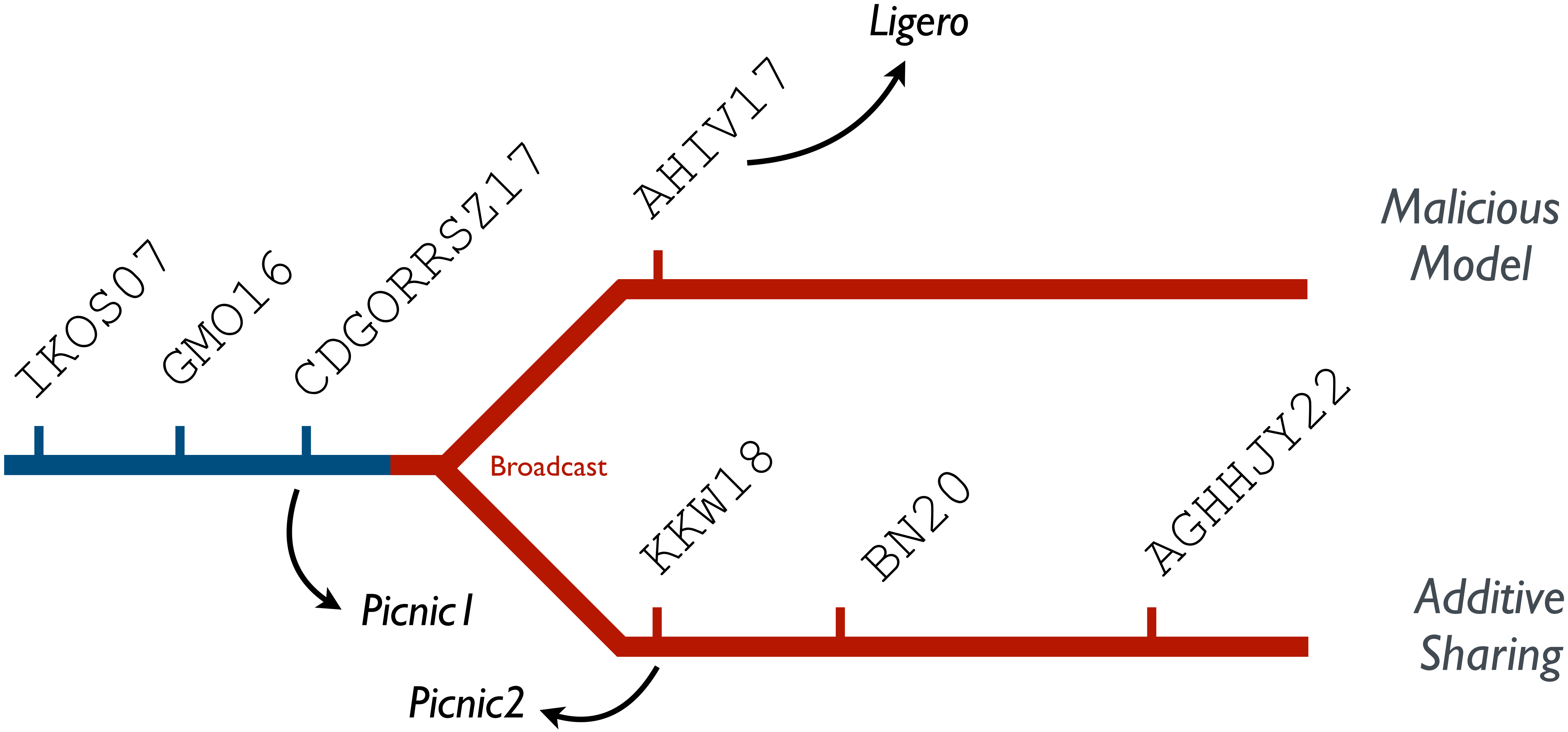
MPC-in-the-Head Paradigm



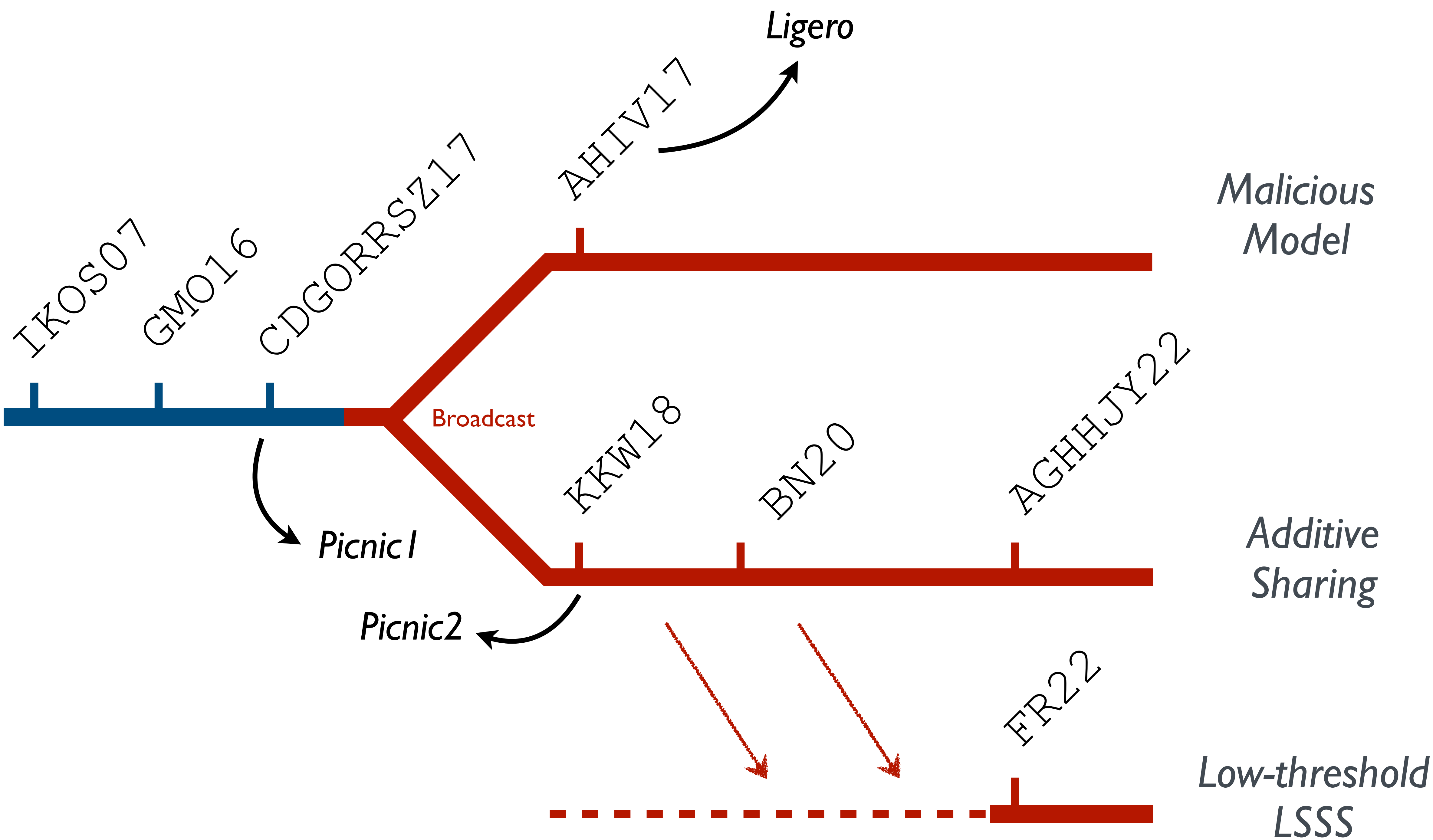
MPC-in-the-Head Paradigm



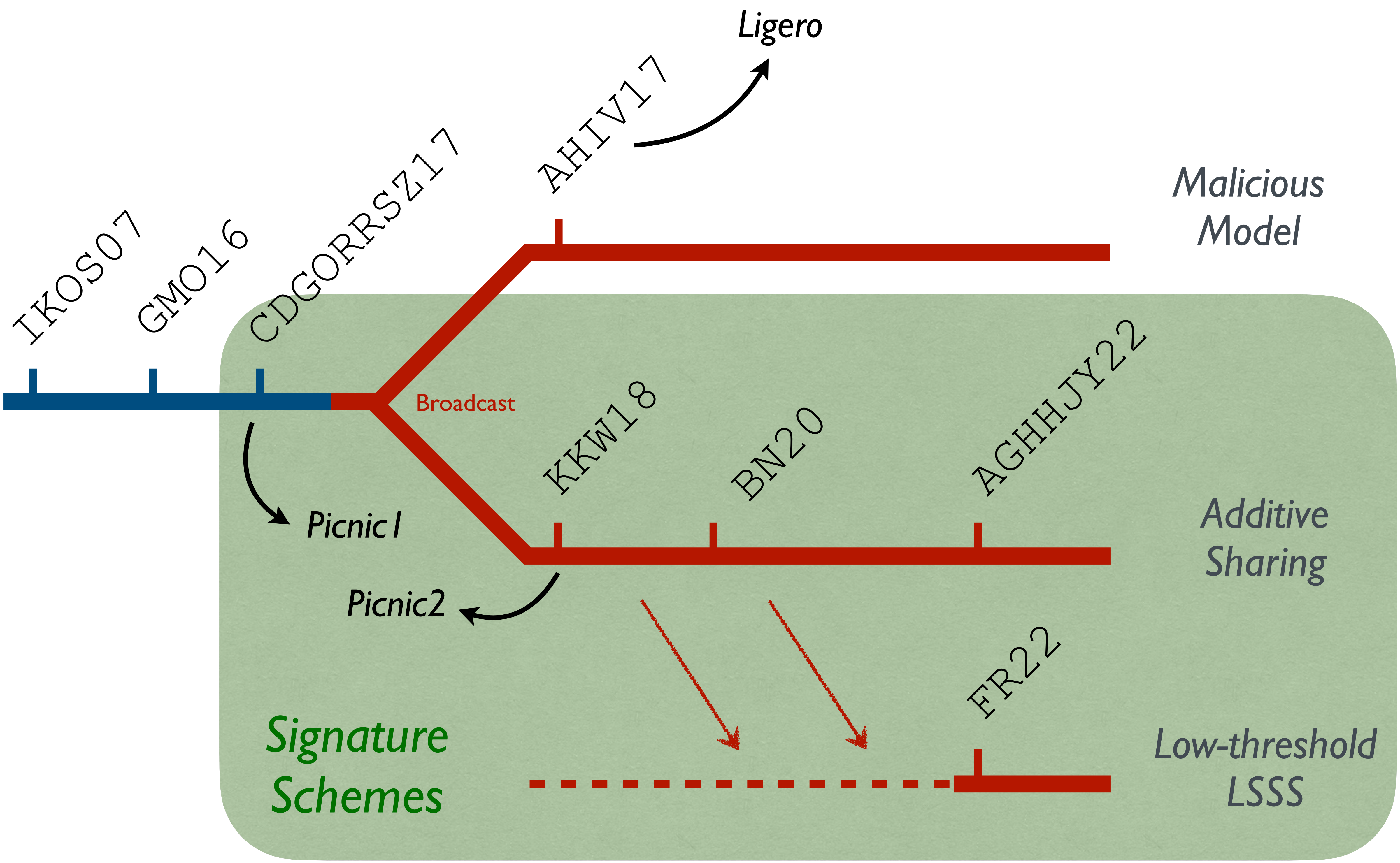
MPC-in-the-Head Paradigm



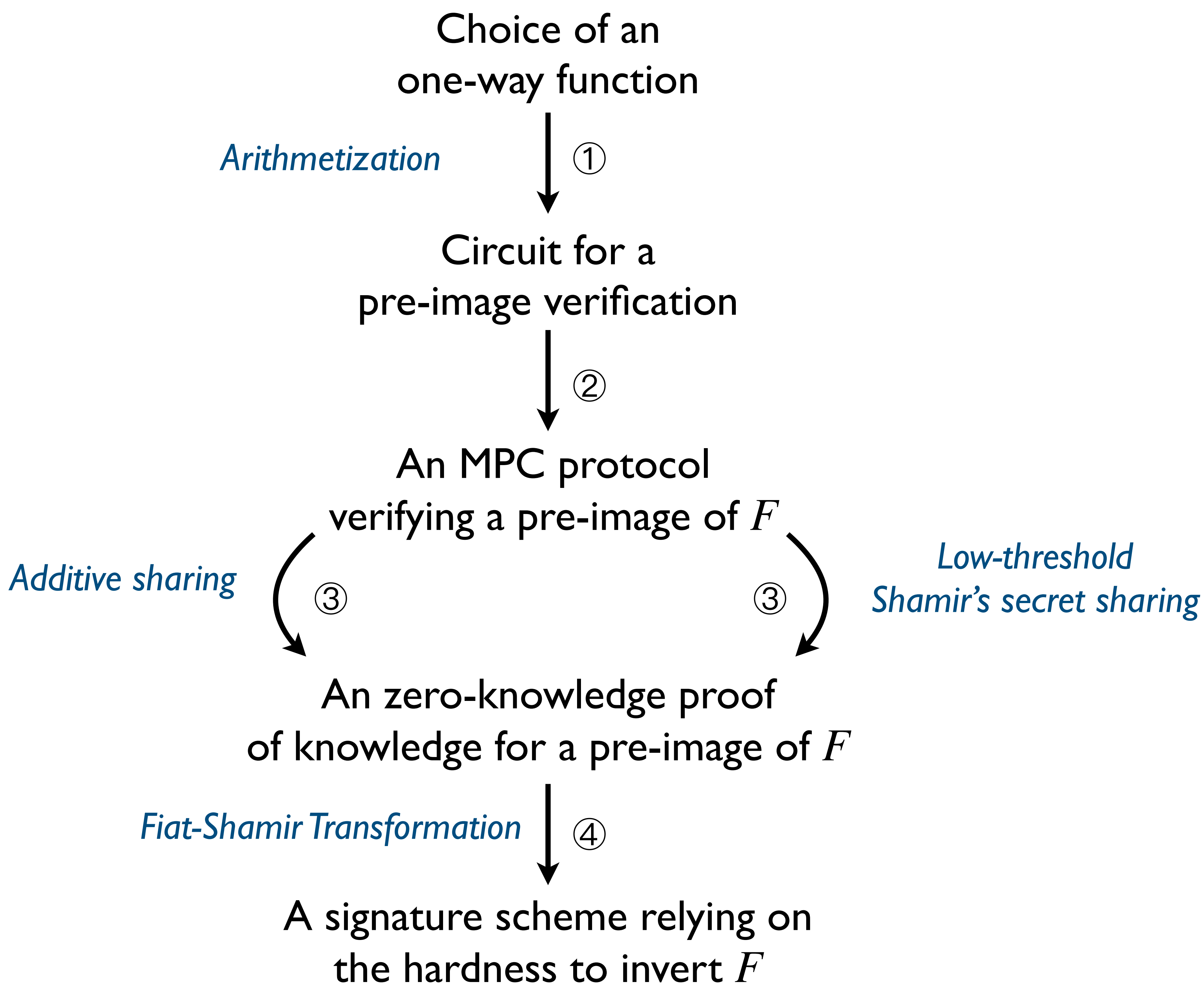
MPC-in-the-Head Paradigm



MPC-in-the-Head Paradigm



Build a signature scheme from MPC



Build a signature scheme from MPC

- Syndrome Decoding (Hamming metric)
- Syndrome Decoding (rank metric)
- MinRank
- Multivariate Quadratic

Choice of an
one-way function

Arithmetization ↓ ①

Circuit for a
pre-image verification

↓ ②

An MPC protocol
verifying a pre-image of F

Additive sharing ③ ↓ ③ *Low-threshold Shamir's secret sharing*

An zero-knowledge proof
of knowledge for a pre-image of F

Fiat-Shamir Transformation ↓ ④

A signature scheme relying on
the hardness to invert F

Build a signature scheme from MPC

Choice of an
one-way function

Arithmetization ↓ ①

Circuit for a
pre-image verification

↓ ②

An MPC protocol
verifying a pre-image of F

Additive sharing ③ ↙ ↘ ③ *Low-threshold Shamir's secret sharing*

An zero-knowledge proof
of knowledge for a pre-image of F

Fiat-Shamir Transformation ↓ ④

A signature scheme relying on
the hardness to invert F

$x \in \mathbb{F}_{q^m}^n$ satisfies $\mathrm{wt}_R(x) \leq r$

$\iff$

There exists $P(X) := X^{q^r} + \sum_{i=1}^{n} p_i X^{q^i}$

such that $P(x_i) = 0$ for all x_i

Build a signature scheme from MPC

An MPC protocol
verifying a pre-image of F

Use only a broadcast channel
Perform only linear operations

Additive sharing
[KKW18,AGHHJY22]

Shorter signatures
Highly parallelizable
Slower signing time
Signing time ≈ verification time
Computational cost is mainly due to symmetric primitive

Low-threshold Shamir's secret sharing
[FR22]

Faster signing time
Highly parallelizable
Very fast verification
A bit simpler implementation
Larger signatures
Restriction # of parties
Computational cost is mainly due to arithmetics

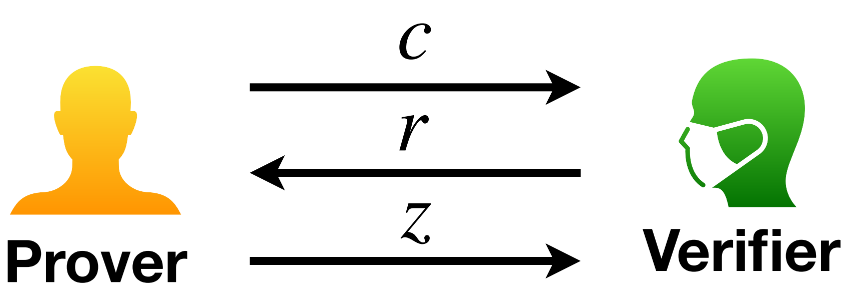
An zero-knowledge proof
of knowledge for a pre-image of F

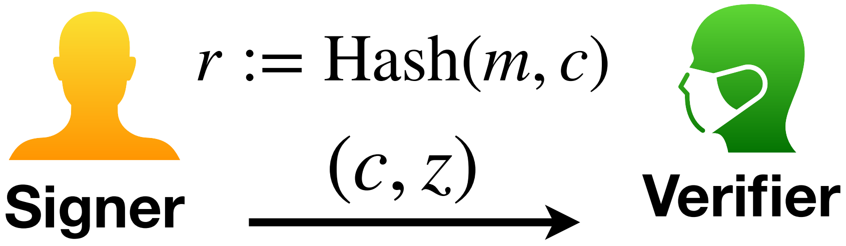
Build a signature scheme from MPC

An zero-knowledge proof
of knowledge for a pre-image of F

Fiat-Shamir Transformation

A signature scheme relying on
the hardness to invert F





Submitted candidates at NIST call

- *Syndrome Decoding Problem:*

SD-in-the-Head

C. Aguilar Melchor, T. Feneuil, N. Gama, S. Gueron, J. Howe, D. Joseph, A. Joux, E. Persichetti, T. Randrianarisoa, M. Rivain, D. Yue.

- *Rank Syndrome Decoding Problem:*

RYDE

N. Aragon, M. Bardet, L. Bidoux, J.-J. Chi-Domínguez, V. Dyseryn, T. Feneuil, P. Gaborit, A. Joux, M. Rivain, J.-P. Tillich, A. Vinçotte.

- *Min Rank Problem:*

MIRA

N. Aragon, M. Bardet, L. Bidoux, J.-J. Chi-Domínguez, V. Dyseryn, T. Feneuil, P. Gaborit, R. Neveu, M. Rivain, J.-P. Tillich.

- *Multivariate Quadratic Problem:*

MQOM: MQ on my Mind

T. Feneuil, M. Rivain

Performances

Shamir's sharing

	Short Instance			Fast Instance		
	$\|\mathrm{sig}\|$	t_{sign}	t_{verify}	$\|\mathrm{sig}\|$	t_{sign}	t_{verify}
SDitH-256	8.3	13.4	12.5	10.1	5.1	1.6
SDitH-251	8.3	22.1	21.2	10.1	4.4	0.6
MQOM-251	6.6	28.5	27.3	7.9	11.5	10.2
MQOM-31	6.4	44.4	41.7	7.7	17.7	15.5
RYDE	6.0	23.4	20.1	7.4	5.4	4.4
MIRA	5.6	46.8	43.9	7.3	37.4	36.7

Additive sharing

128-bit security
Isochronous implementations
Size in kilobytes, timing in Mcycles
@2.60GHz: 1 millisecond ≈ 2.6 Mcycles

Performances

- *How it scales for high security level?*

	Short Instance	Fast Instance
Category I	5.6 KB → 8.3 KB	7.3 KB → 10.1 KB
Category III	11.8 KB → 19.2 KB	15.5 KB → 25.6 KB
Category V	20.8 KB → 33.4 KB	27.8 KB → 43.9 KB

- *What about the public key?*
  - Between 47 and 120 bytes for category I
  - Between 99 and 234 bytes for category V

Advantages and limitations

- Limitations
  - Relatively ***slow***
    - Greedy use of symmetric cryptography
  - Relatively ***large*** signatures
  - ***Quadratic*** growth in the security level

- Advantages
  - ***Conservative*** hardness assumption
  - ***Small*** (public) keys
  - Highly ***parallelizable***
  - ***Good*** public key + signature size
  - Adaptive and ***tunable*** parameters

Conclusion

- MPC-in-the-Head
  - Very versatile and tunable
  - Can be applied on any one-way function
  - A practical tool to build *conservative* signature schemes
    - *No structure* in the security assumption

- Perspectives
  - *Additive-based MPCitH:* stable
  - *Low-threshold-based MPCitH*: new approach, could lead to follow-up works

References

- [AGHHJY22] C. Aguilar-Melchor, N. Gama, J. Howe, A. Hülsing, D. Joseph, D. Yue. *The Return of the SDitH*. Eurocrypt 2023.
- [AHIV17] S. Ames, C. Hazay, Y. Ishai, M. Venkitasubramaniam. *Ligero: Lightweight sublinear arguments without a trusted setup.* CCS 2017.
- [BN20] C. Baum, A. Nof. *Concretely-efficient zero-knowledge arguments for arithmetic circuits and their application to lattice-based cryptography*. PKC 2020.
- [CDGORRSZ17] M. Chase, D. Derler, S. Goldfeder, C. Orlandi, S. Ramacher, C. Rechberger, D. Slamanig, G. Zaverucha. *Post-quantum zero-knowledge and signatures from symmetric-key primitives*. CCS 2017.
- [FR22] T. Feneuil, M. Rivain. *Threshold Linear Secret Sharing to the Rescue of MPC-in-the-Head*. Cryptology ePrint Archive, paper 2022/1407.
- [GMO16] I. Giacomelli, J. Madsen, C. Orlandi. *ZKBoo: Faster zero-knowledge for Boolean circuits*. USENIX Security 2016
- [IKOS07] Y. Ishai, E. Kushilevitz, R. Ostrovsky, A. Sahai. Zero-knowledge from secure multiparty computation. STOC 2007.
- [KKW18] J. Katz, V. Kolesnikov, X. Wang. *Improved non-interactive zero knowledge with applications to post-quantum signatures*. CCS 2018.