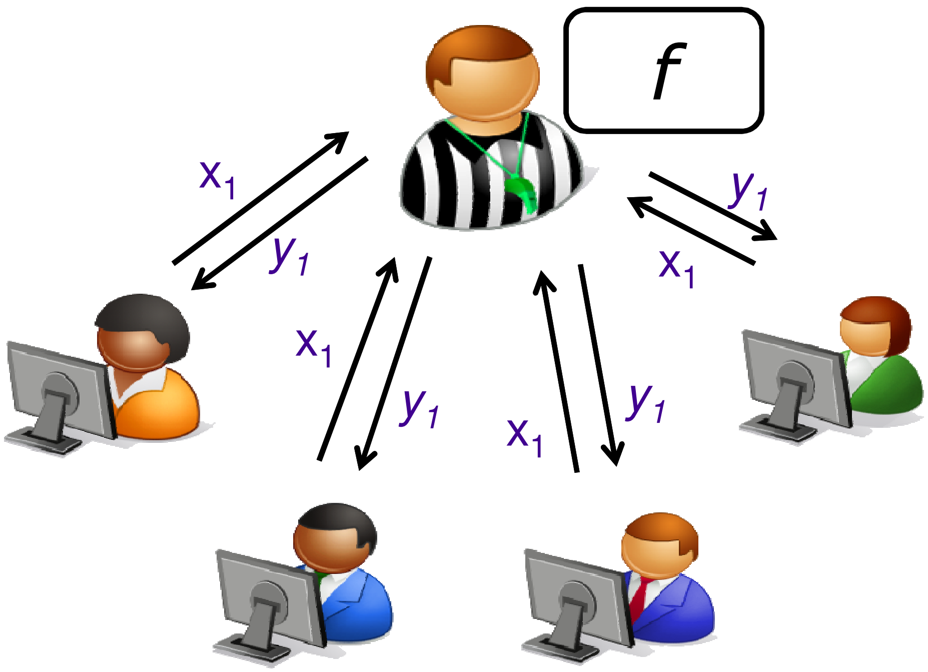
# Broadcast (and Round) Efficient Secure Multiparty Computation

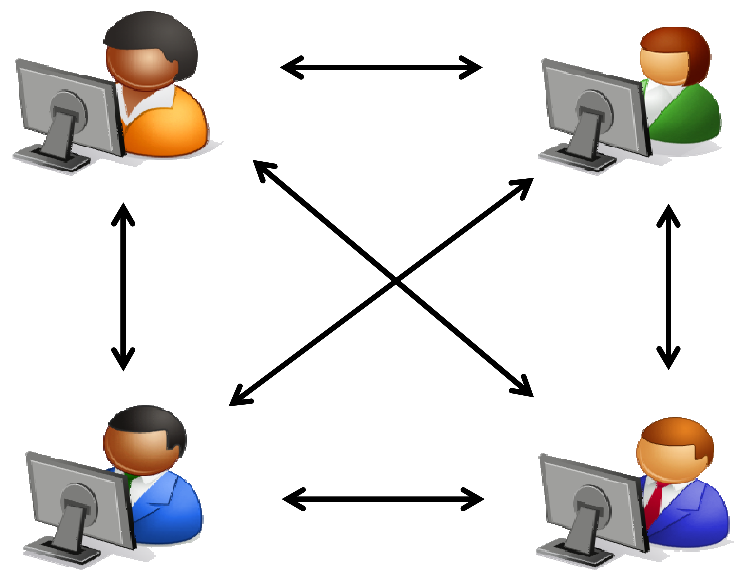
Juan Garay (Yahoo Labs)
Clint Givens (Maine School of Science and Mathematics)
Rafail Ostrovsky (UCLA)
Pavel Raykov (ETH)

# Secure Multiparty Computation (MPC)

**Ideal World**
**(trusted party)**

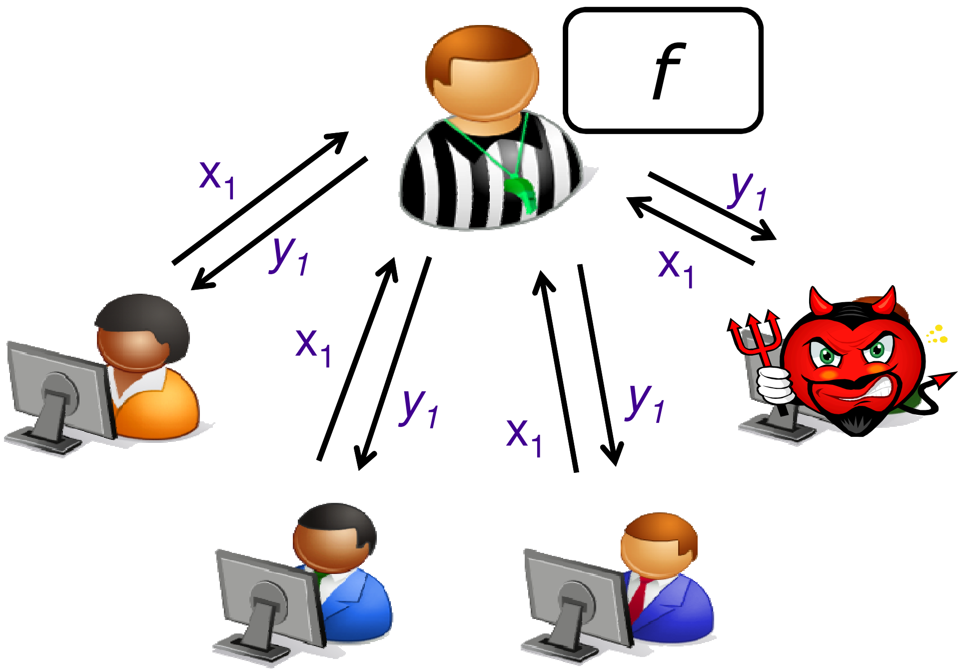



**Real World**
**(just the players)**

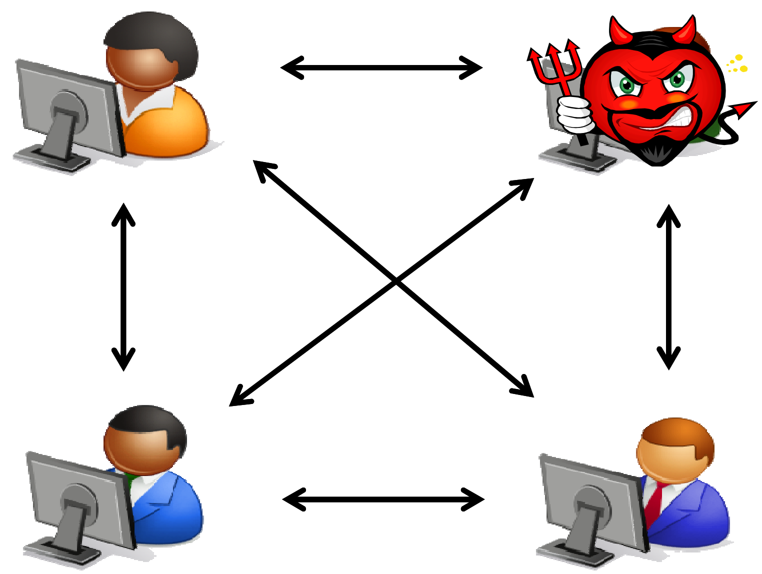


# Secure Multiparty Computation (MPC)

**Ideal World**
**(trusted party)**




**Real World**
**(just the players)**



# Secure Multiparty Computation (MPC)

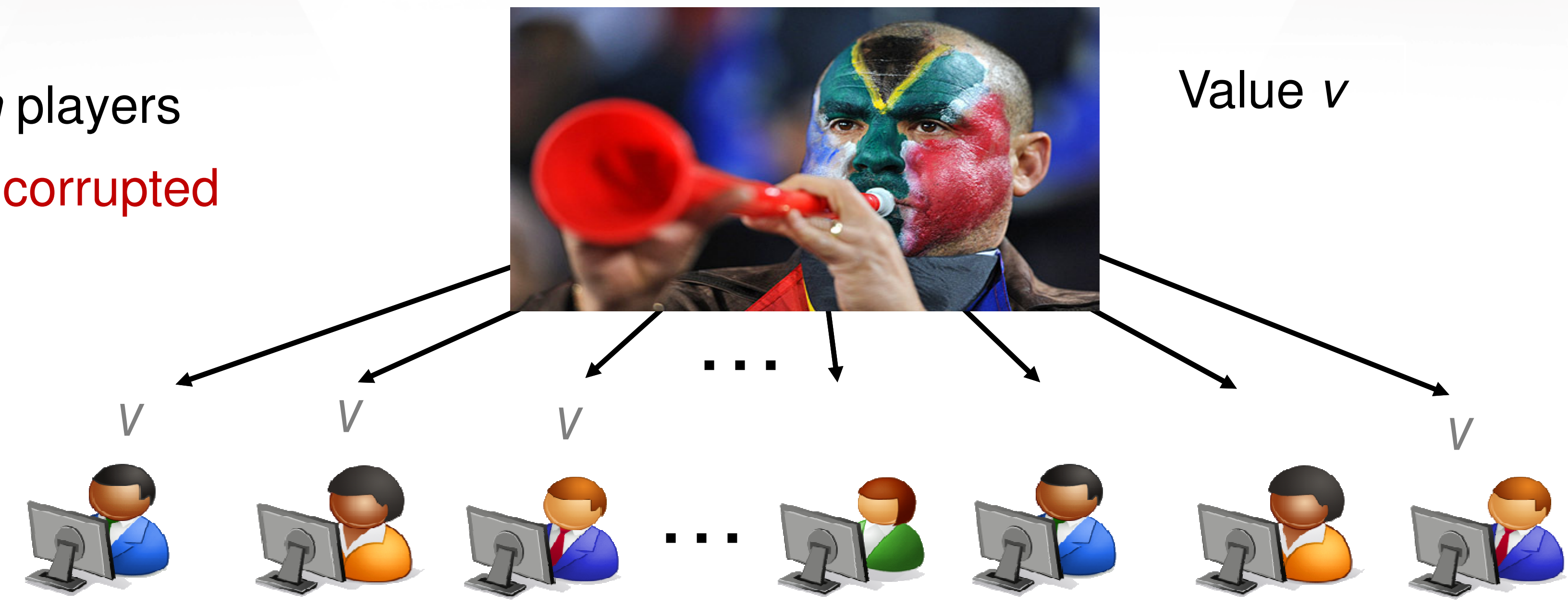
- **Secure multi-party computation** (MPC) [GMW'87] :
  - $n$ parties $\{P_1, P_2, \ldots, P_n\}$, $t$ *corrupted*; each $P_i$ holds a private input $x_i$
  - One public function $f(x_1,x_2,\ldots,x_n)$
  - All want to learn $y = f(x_1,x_2,\ldots,x_n)$ (*Correctness*)
  - Nobody wants to disclose his private input (*Privacy*)
- **Secure 2-party computation** (2PC) [Yao'82] : $n=2$
- *Computationally secure* MPC (2PC)

# Broadcast Functionality ("Channel") [PSL'80]

$n$ players

$t$ corrupted

Value $v$

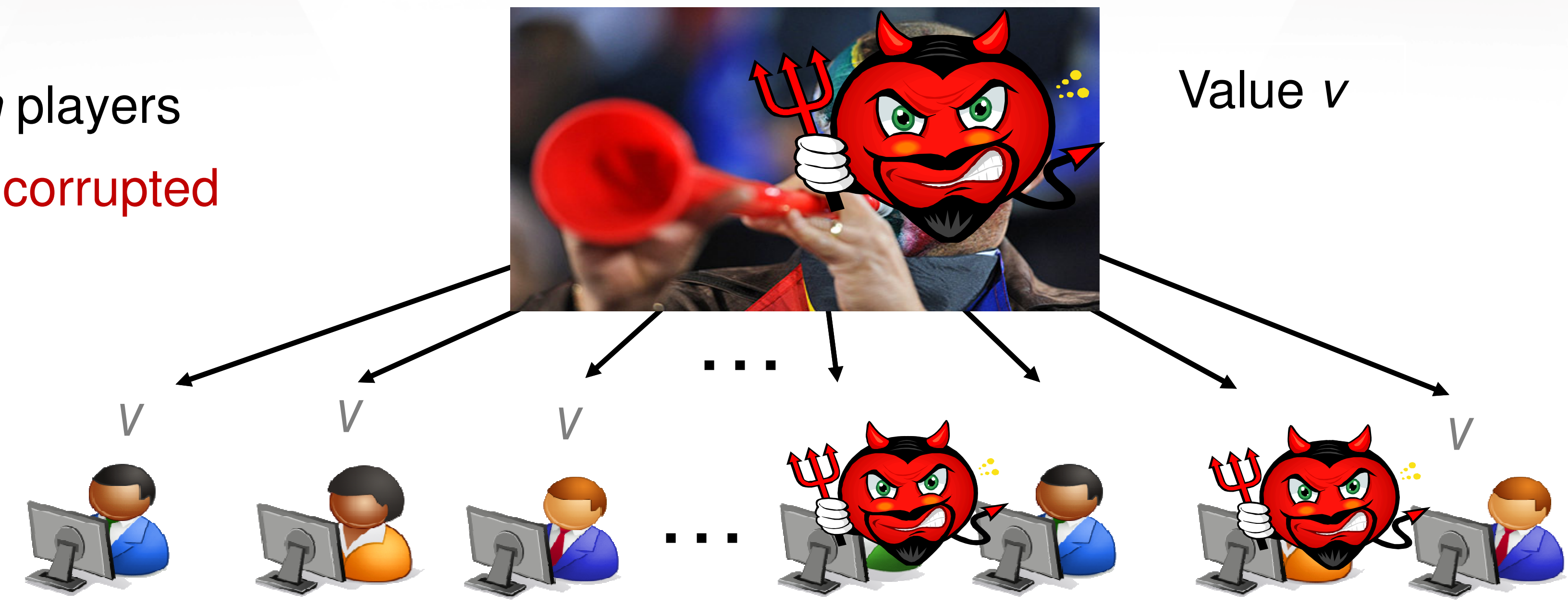


# Broadcast Functionality (“Channel”) [PSL’80]

*n* players

*t* corrupted

Value *v*




If source is honest, $v_i = v$ (Validity)

$v_i = v_j$ (Agreement)

# MPC: Model Assumptions

*Unconditionally secure* MPC typically assumes:

- for $t < n/3$ [BGW'88, CCD'88]:
  - secure (private and authentic) pairwise channels
  - broadcast channel—but it may be realized by Byzantine agreement protocol
- for $t < n/2$ [RB'89]:
  - secure (private and authentic) pairwise channels
  - physical ***broadcast channel*** (no protocol exists!)

# An MPC Protocol for *f*

The "*share-compute-reveal*" paradigm:

1. *Share phase*: Each $P_i$ "commits" to his input (using Verifiable Secret Sharing [VSS] — next slide).
2. *Compute phase*: Shared inputs are used to evaluate an arithmetic circuit *C* gate-by-gate. (Typically a *linear* VSS scheme is used.)
3. *Reveal phase*: At *C* 's output gate, parties possess a verifiable sharing of $f(x_1,x_2,\ldots,x_n)$; parties publicly reconstruct this value.

- **Multiplication gate:** Most expensive part of MPC protocol — typically requires broadcast channel



YAHOO!

# An MPC Protocol for *f* (cont'd)

- **Multiplication gate:** Most expensive part of MPC protocol ─ typically require broadcast channel
- [Beaver91]: Technique for evaluating multiplication gates efficiently based on (verifiably shared) *multiplication triples*, vectors $(a,b,c) \in \mathbf{F}^3$ s.t. $a,b$ random and $ab = c$,

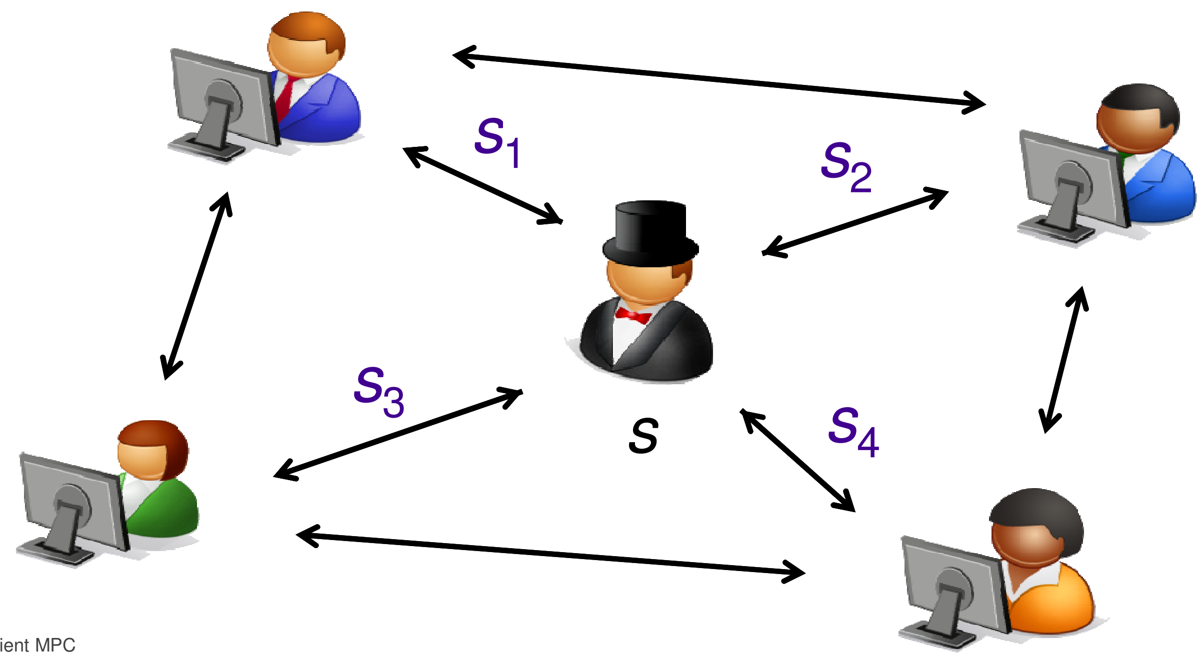
  → cost of mult. gate = cost of one VSS reconstruction phase
- The *pre-processing* phase of MPC



YAHOO!

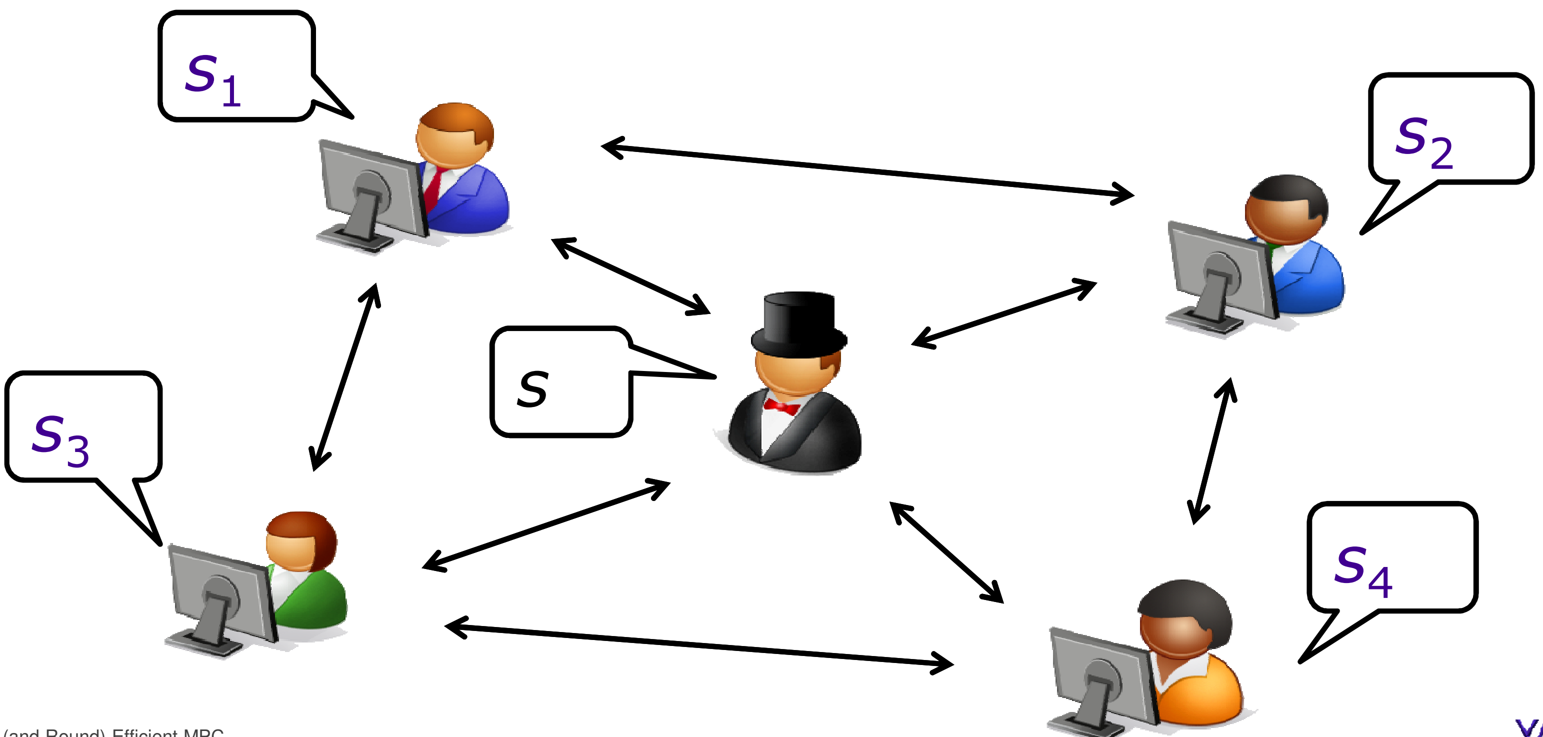
# Verifiable Secret Sharing [CGMA'85]

- **Phase 1: Share**
  - *Dealer* distributes shares of a secret *s*
  - Players interact to verify sharing is *valid*



# Verifiable Secret Sharing (cont'd)

- **Phase 2: Reconstruct**
  - Players reveal shares and use them to recover *s*






YAHOO!

# Verifiable Secret Sharing (cont'd)

## Security Requirements:

- Even a cheating Dealer is committed to *some* secret after **Share** phase (Commitment)
- Honest Dealer committed to correct secret (Correctness)
- Prior to **Reconstruct** phase, cheating players have *zero information* on value of $s$ (Privacy)
- If the parties verifiably share secrets $\{s^{(k)}\}$, then they also (w/o further interaction) verifiably share any (public) linear combination of the secrets (Linearity)

**Statistical security:** $t < n/2$: Protocols subject to some (negligibly small) error probability [CCD88, DDWY93]

# MPC: Model Assumptions

RECALL

*Unconditionally secure* MPC typically assumes:

- for $t < n/3$ [BGW'88, CCD'88]:
  - secure (private and authentic) pairwise channels
  - broadcast channel—but it may be realized by Byzantine agreement protocol
- for $t < n/2$ [RB'89]:
  - secure (private and authentic) pairwise channels
  - physical ***broadcast channel*** (no protocol exists!)



YAHOO!

# MPC: Model Assumptions

*Unconditionally secure* MPC typically assumes:

- for $t < n/3$ [BGW'88, CCD'88]:
  - secure (private and authentic) pairwise channels
  - broadcast channel—but it may be realized by Byzantine agreement protocol
- for $t < n/2$ [RB'89]:
  - secure (private and authentic) pairwise channels
  - physical ***broadcast channel*** (no protocol exists!)
  - What if the broadcast operation is **expensive**?

# Goooaaalll!!!:

To drastically reduce the number of broadcast rounds required in MPC for $n > 2t$ (while minimizing overall no. rounds)



YAHOO!

# Our Results

- VSS with **two** b'cast rounds, constant overall rounds
  - First linear VSS protocol enjoying these features
  - (2,0)-bcast (20,1)-round VSS protocol

- Constant-round *pseudosignatures* [PW96]
  - Unconditionally secure *anonymous channel* (aka DC-nets [Chaum'88])
  - Black-box use of VSS; same b'cast complexity



YAHOO!

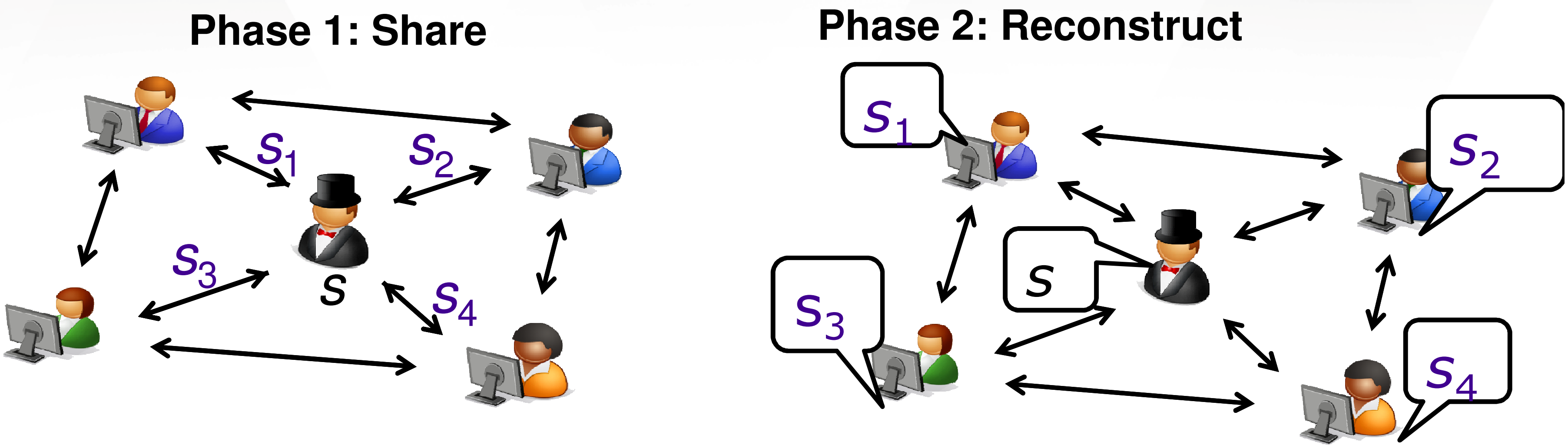
# Our Results

- VSS with **two** b'cast rounds, constant overall rounds
  - First linear VSS protocol enjoying these features
  - (2,0)-bcast  (20,1)-round VSS protocol

- Constant-round *pseudosignatures* [PW96]
  - Unconditionally secure *anonymous channel* (aka DC-nets [Chaum'88])
  - Black-box use of VSS; same b'cast complexity

# Building Blocks

# Weak Secret Sharing



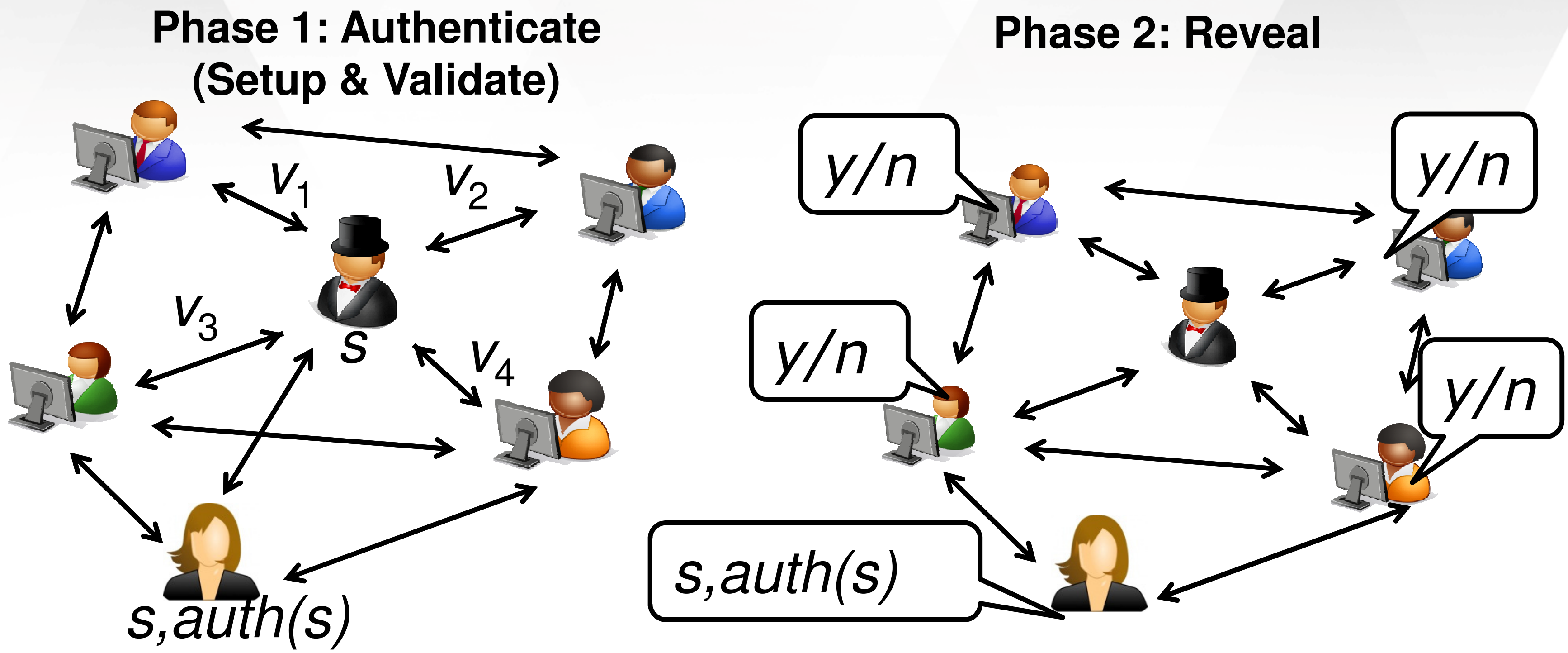

## Security Requirements:

- Even a cheating Dealer is committed to *some* secret $s^* \in \mathbf{F} \cup \{\perp\}$ after **Share** phase (Weak Commitment [without Agreement])
- (Correctness), (Privacy), (Linearity)



YAHOO!

# Information Checking: Signature-like Functionality [RB'89]




## Security Requirements:

- Correctness, Non-Forgery, Commitment, Privacy, Linearity



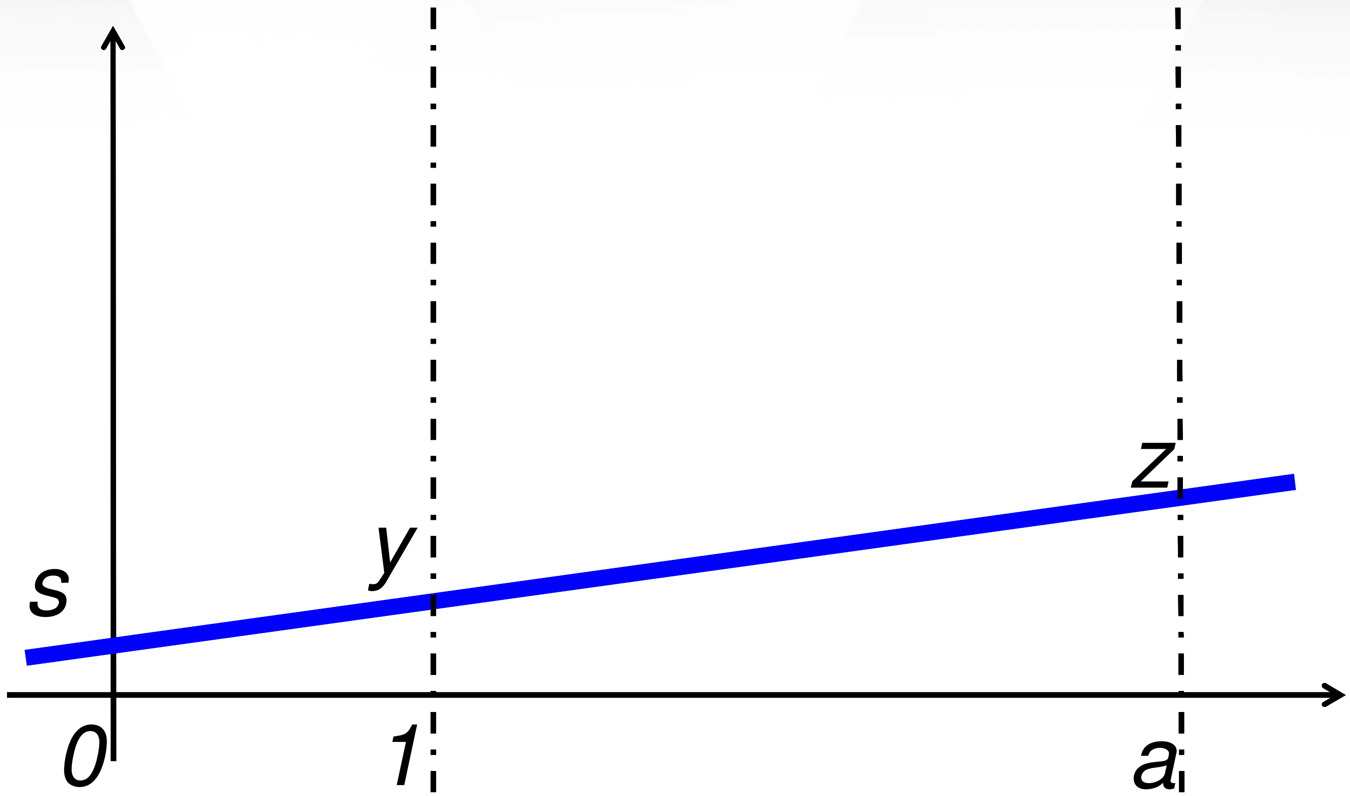
YAHOO!

# Information Checking (cont'd)

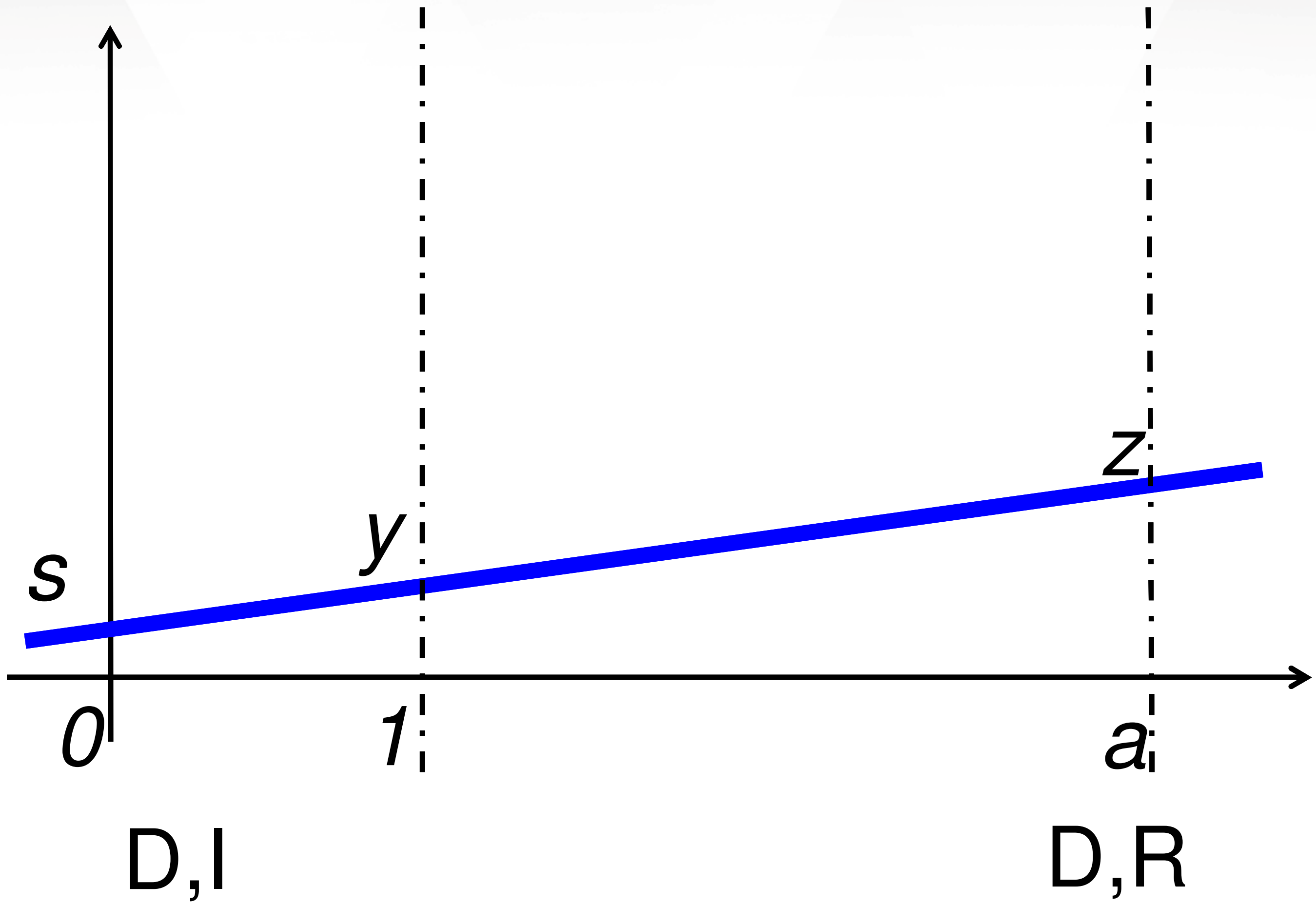
Triple of protocols (ICSetup, ICValidate, ICReveal) which achieves a signature-like functionality for three players: D, I and R

- D holds as input a secret $s \in \mathbf{F}$, which he passes to I in ICSetup
- ICValidate ensures that even if D cheats, I knows a value which R will accept
- In ICReveal, I sends $s$ to R plus some authentication data, based on which R accepts or rejects $s$ as having originated from D

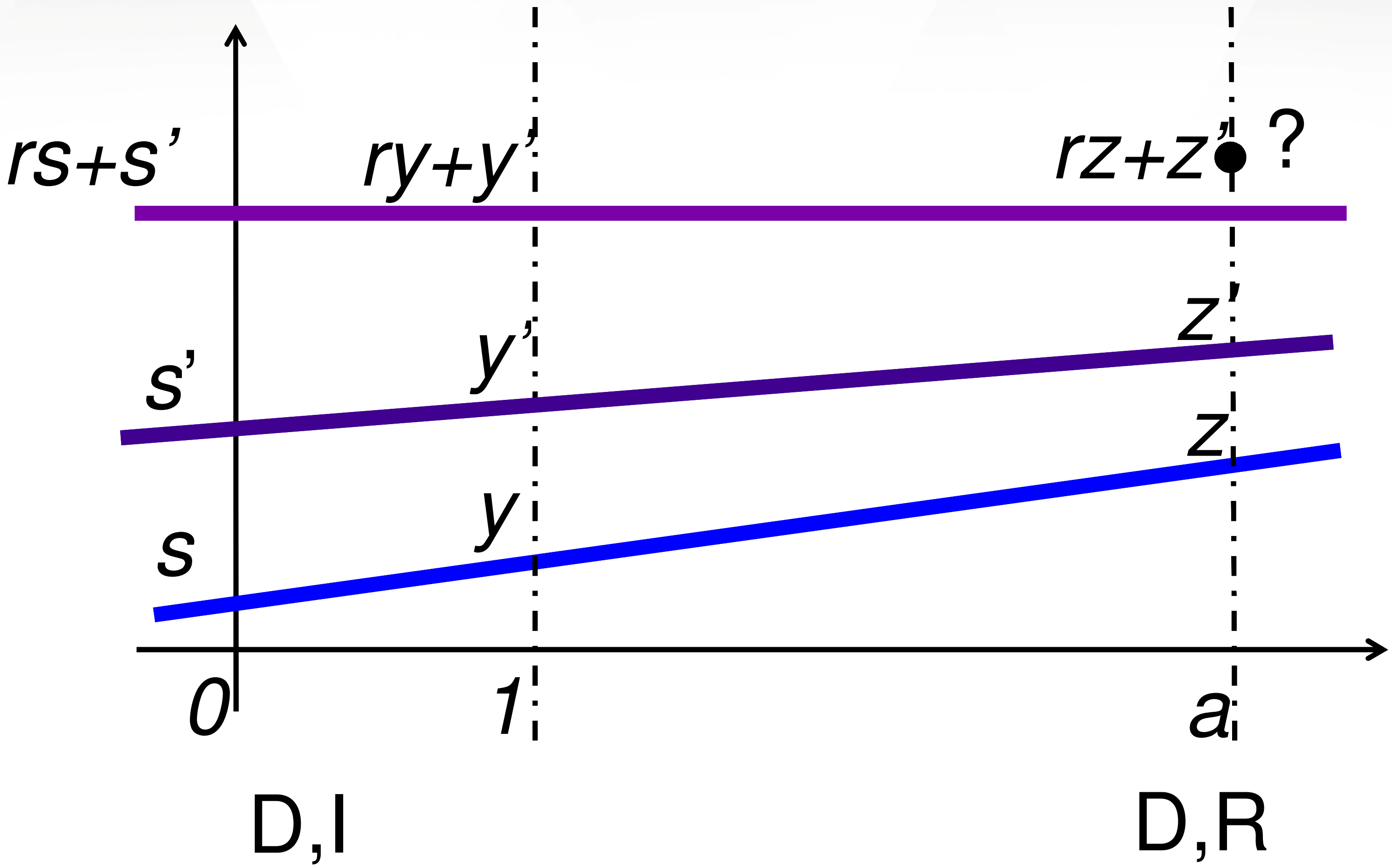
# Implementing IC (n=3)



# Implementing IC (n=3)



# Implementing IC (n=3)

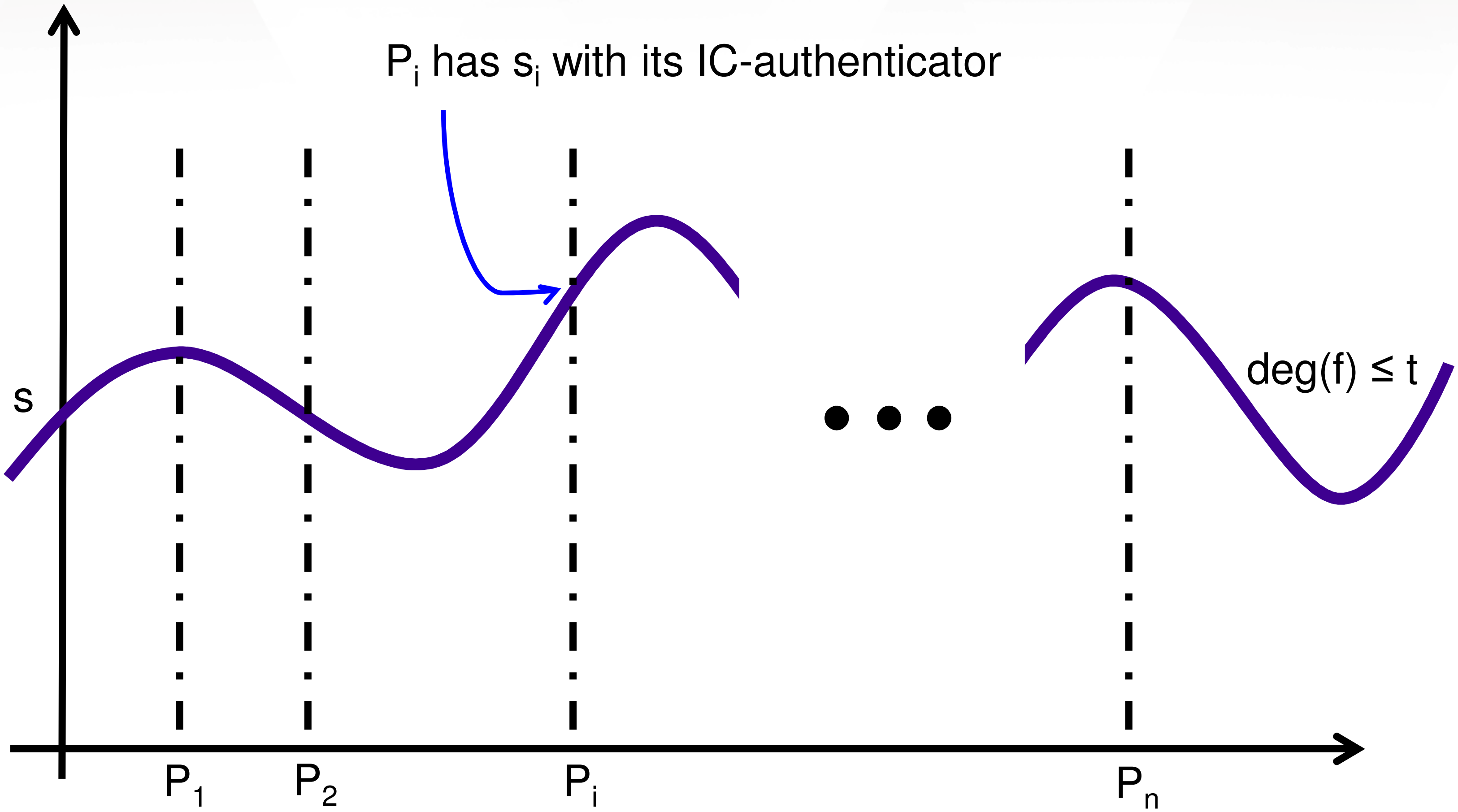




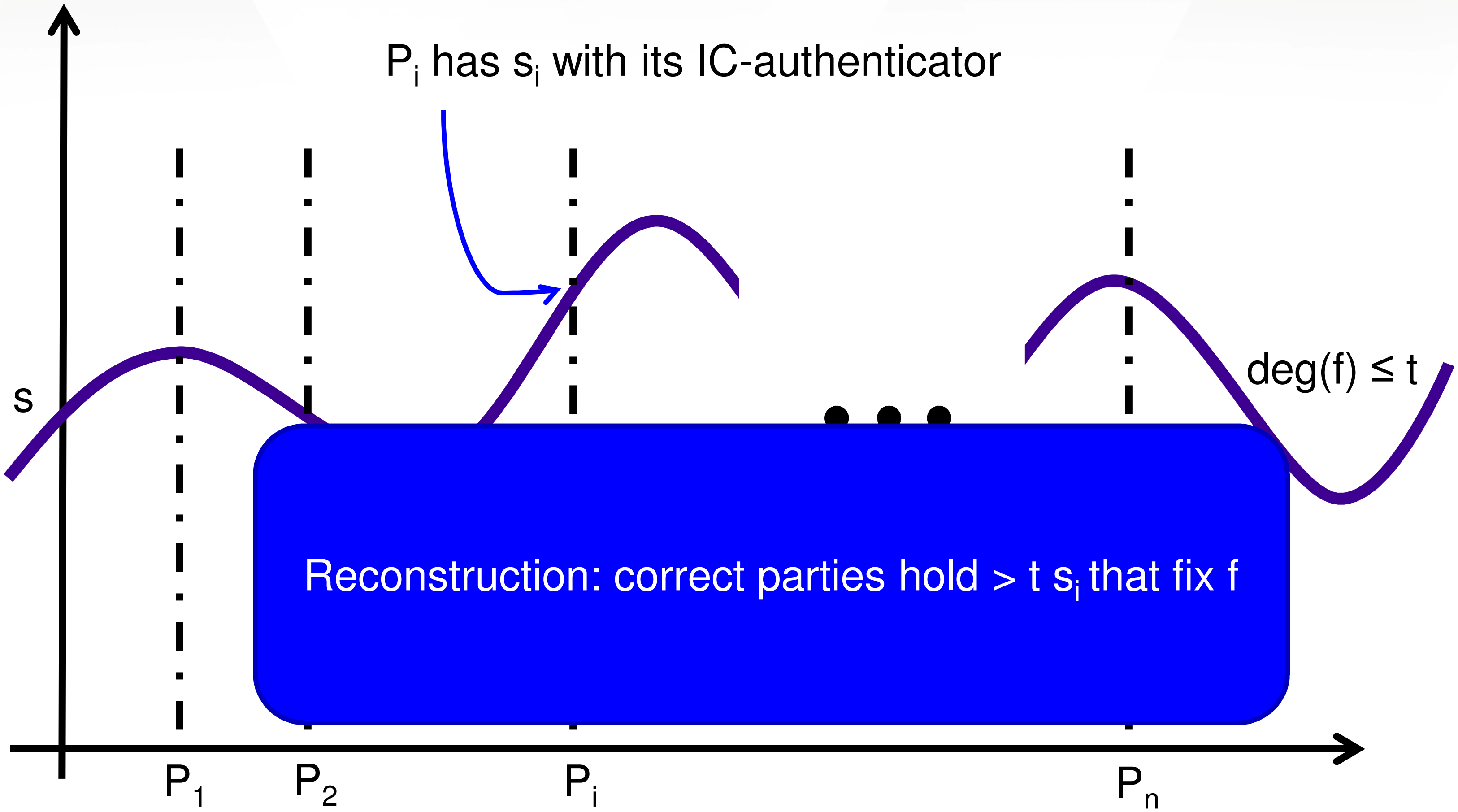

YAHOO!

# The (2,0)-bcast VSS Protocol

# WSS Protocol



# WSS Protocol






YAHOO!

# Weak Secret Sharing Protocol

WSS protocol uses **2 b'casts** in its sharing phase, and admits 2 different reconstruction phases (one w/o b'cast but achieves only WSS-w/o-agreement)

- Protocol WSS-Share($\mathbf{P}$,D,s)

1. D chooses random polynomial $f(x)$ of degree $\leq t$ s.t. $f(0) = s$, and sets $s_i = f(i)$. For each pair $P_i,P_j \in \mathbf{P} - \{D\}$, run **ICSetup**($D,P_i,P_j,s_i$)

2-5. **2 x BROADCAST in 4,5:** For each $P_i,P_j \in \mathbf{P} - \{D\}$, run **ICValidate**($D,P_i,P_j,s_i$)

# Weak Secret Sharing Protocol (cont'd)

- Protocol WSS-Rec-NoBC(**P**,D,s)
  1. For each pair $P_i,P_j \in \mathbf{P} - \{D\}$, run **ICReveal**$(P_i,P_j,s_i)$. If $P_i$ accepts at least $n-t$ pieces, and all accepted pieces lie on a polynomial $f(x)$ of degree $\leq t$, then takes $s = f(0)$ to be the secret, otherwise $\perp$

(WSS-Share, WSS-Rec-NoBC) is a linear WSS-without-agreement scheme, secure against a static, unbounded adversary who corrupts $t < n/2$ players.

**Note:** (2,0)-bcast, (6,1)-round protocol (**same no. of b'casts** as IC protocol)

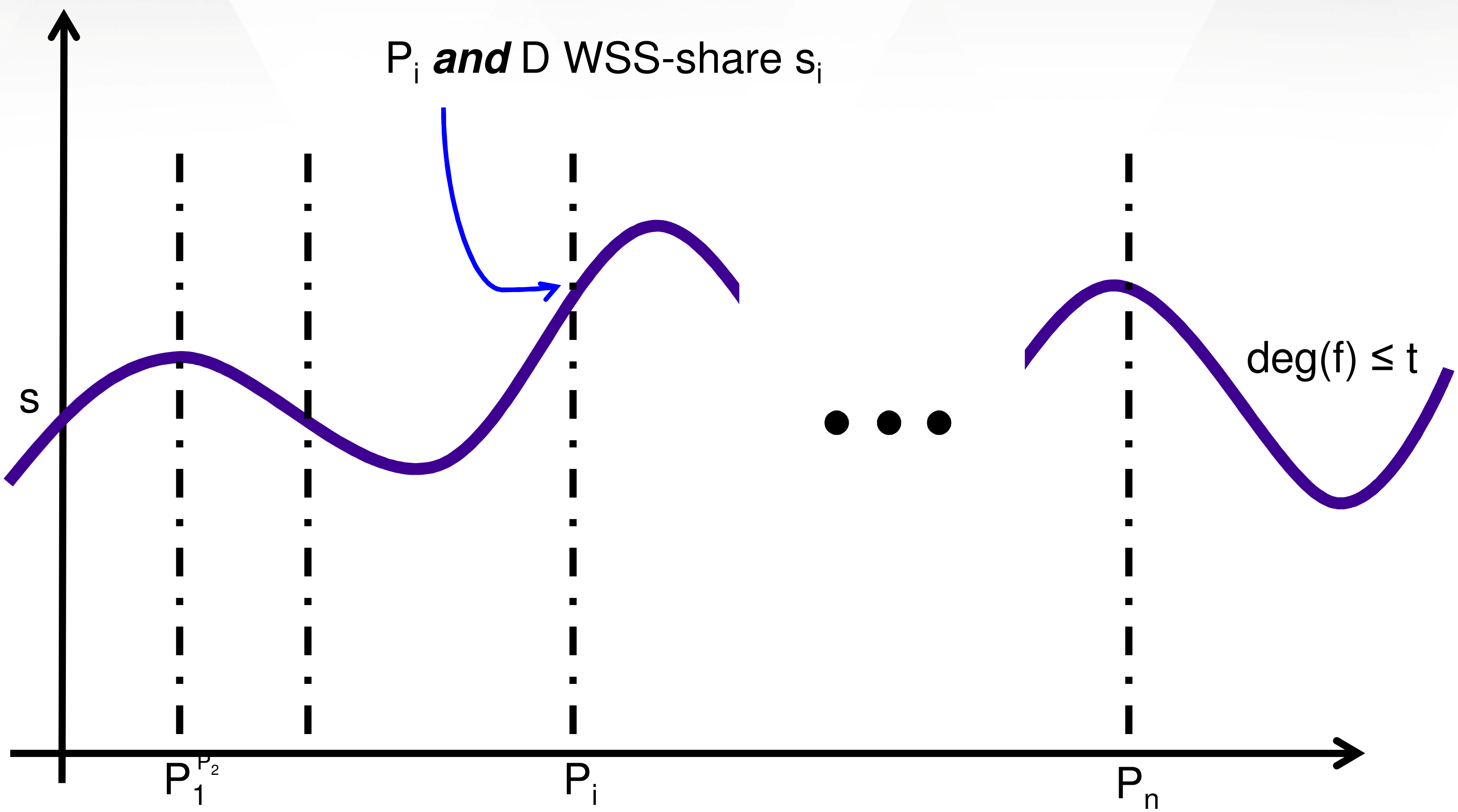
# (**3**,0)-bcast VSS Protocol (high level)

- Inspired by [RB89].
  - First, D distributes shares of $t$-degree polynomial $f(x)$ s.t. $f(0) = s$, and of additional random $t$-degree polynomials $g_k(x)$ $(1 \leq k \leq n\lambda)$
  - *Each* player commits to all shares via **WSS** protocol
  - All players jointly carry out *cut-and-choose*, in which players are challenged to reconstruct $g_k$'s or $f + g_k$'s
  - Players who complain of incompatible shares, or fail to participate, have their shares **broadcast** (and hence fixed) by D



YAHOO!

# VSS Protocol: Distributing Shares






YAHOO!

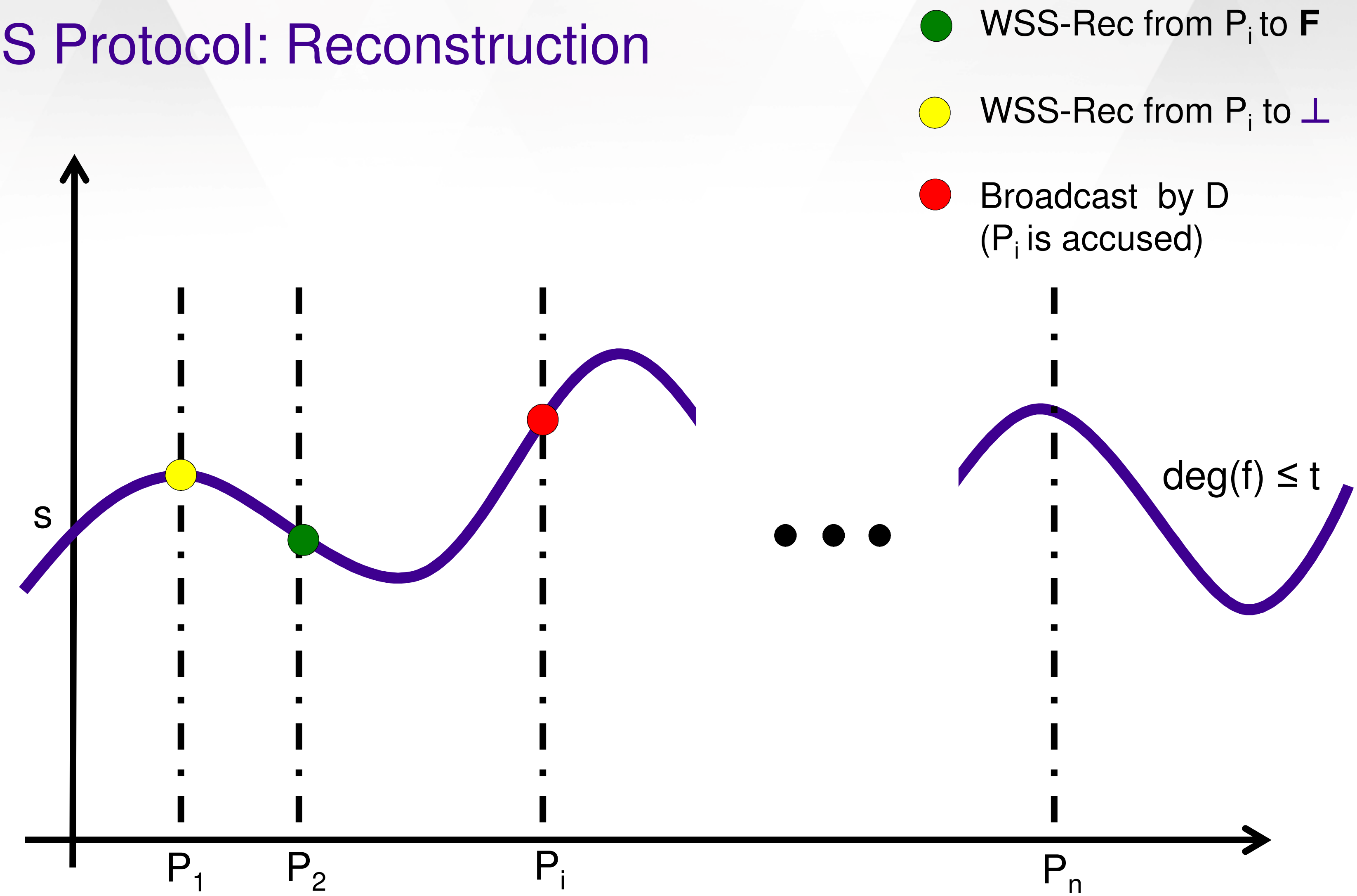
# (**3**,0)-bcast VSS Protocol (high level – cont'd)

- [RB89]'s VSS requires 7 b'cast rounds in sharing phase. Our novelties:
  - We require *the dealer* D *as well as the players* to commit via WSS to the shares he distributed
    - After all commitments are in place, players **broadcast** a round of cut-and-choose challenges
  - Additional trick: *pre-broadcast* in WSS share phase
    - Parties "semi-commit" to their intended WSS final-round broadcast, by first sending on *point-to-point* channels, which is then echoed in the next round. This allows cut-and-choose challenges in the *same round*



YAHOO!

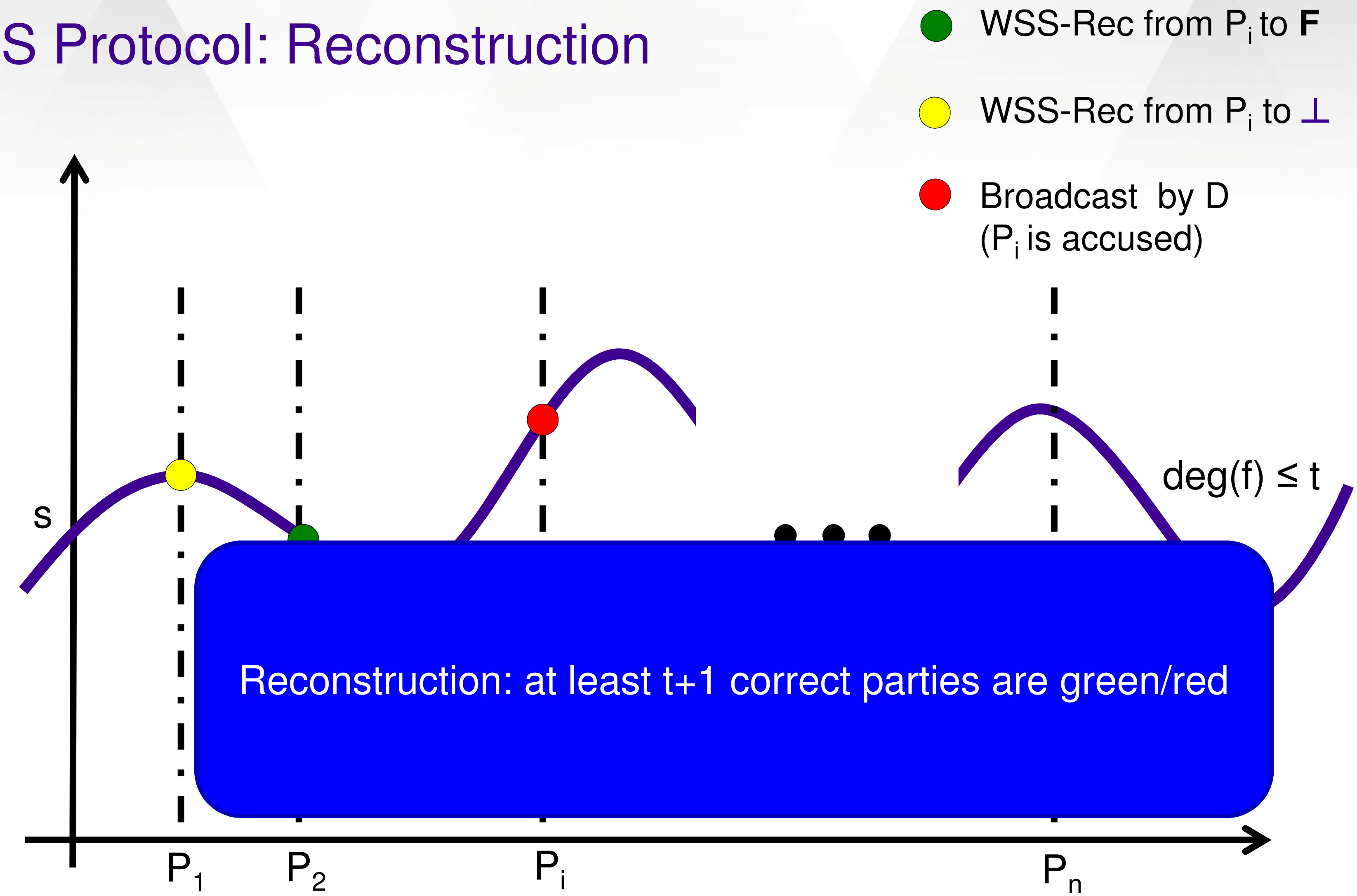
# VSS Protocol: Reconstruction





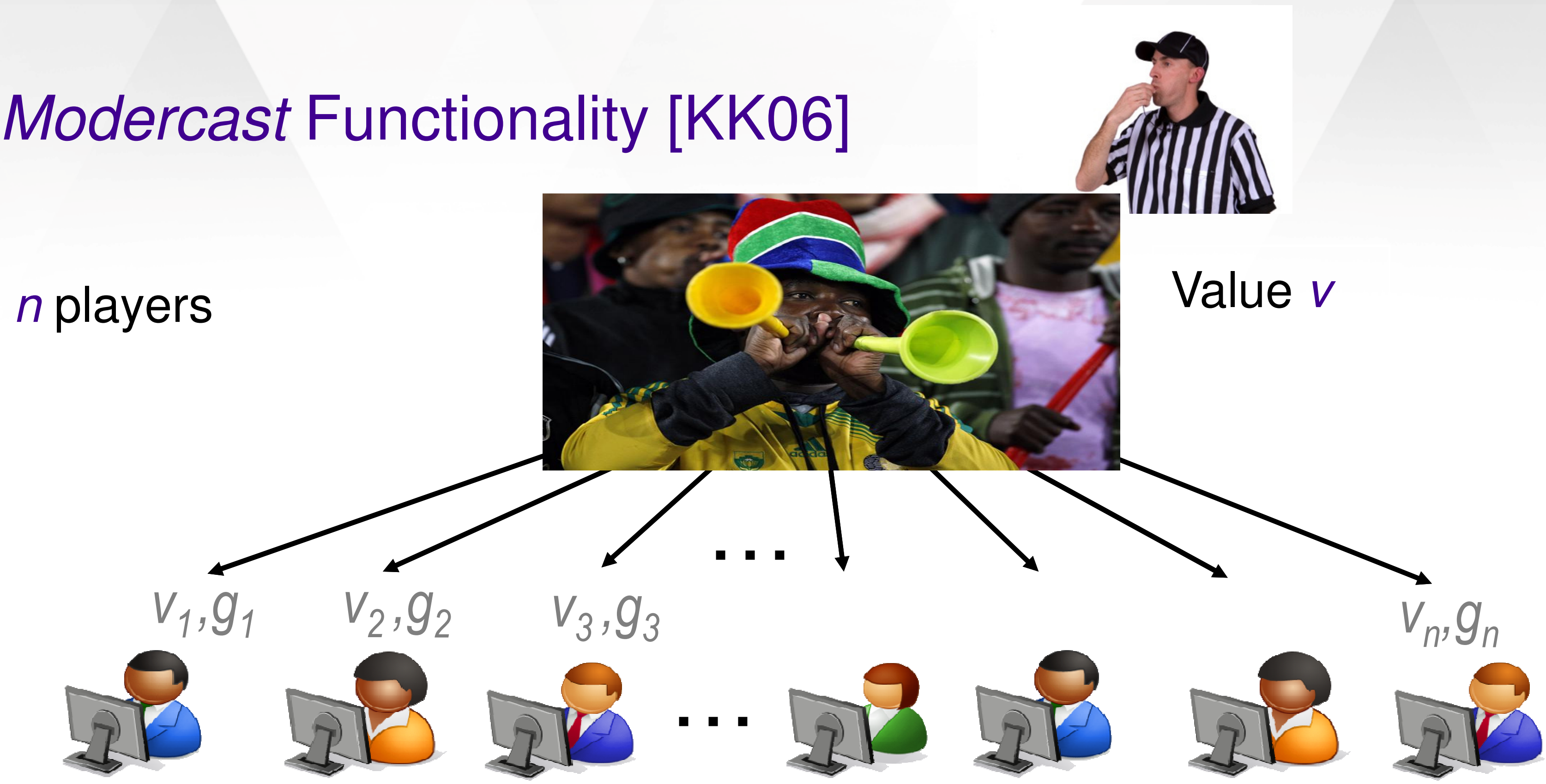

YAHOO!

# VSS Protocol: Reconstruction



# From (3,0)-bcast to (2,0)-bcast VSS

# *Modercast* Functionality [KK06]




If the moderator is honest, $(v_i,g_i) = (v',1)$ (*Moderated* Agreement)

If some honest party has $g=1$, $(v_i,g_i) = (v',0/1)$ (*Graded* Agreement)

If some honest party has $g=1$ and the source is honest, $(v_i,g_i) = (v,0/1)$ (Validity)



YAHOO!

# Moderated Protocols

**Original protocol**

1. Send/Receive/Compute
2. Broadcast
3. Send/Receive/Compute
4. Send/Receive/Compute
5. Broadcast

**Moderated protocol**

Each $P_i$ starts with $f_i=1$

1. Send/Receive/Compute
2. Modercast, update $f_i$
3. Send/Receive/Compute
4. Send/Receive/Compute
5. Modercast, update $f_i$

# Moderated Protocols

**Original protocol**

1. Send/Receive/Compute
2. Broadcast
3. Send/Receive/Compute
4. Send/Receive/Compute
5. Broadcast

**Moderated protocol**

Each $P_i$ starts with $f_i=1$

1. Send/Receive/Compute
2. Modercast, update $f_i$
3. Send/Receive/Compute
4. Send/Receive/Compute

As secure as original protocol if at least one honest player has $f_i=1$



YAHOO!

# From 3 to 2 Broadcast Rounds

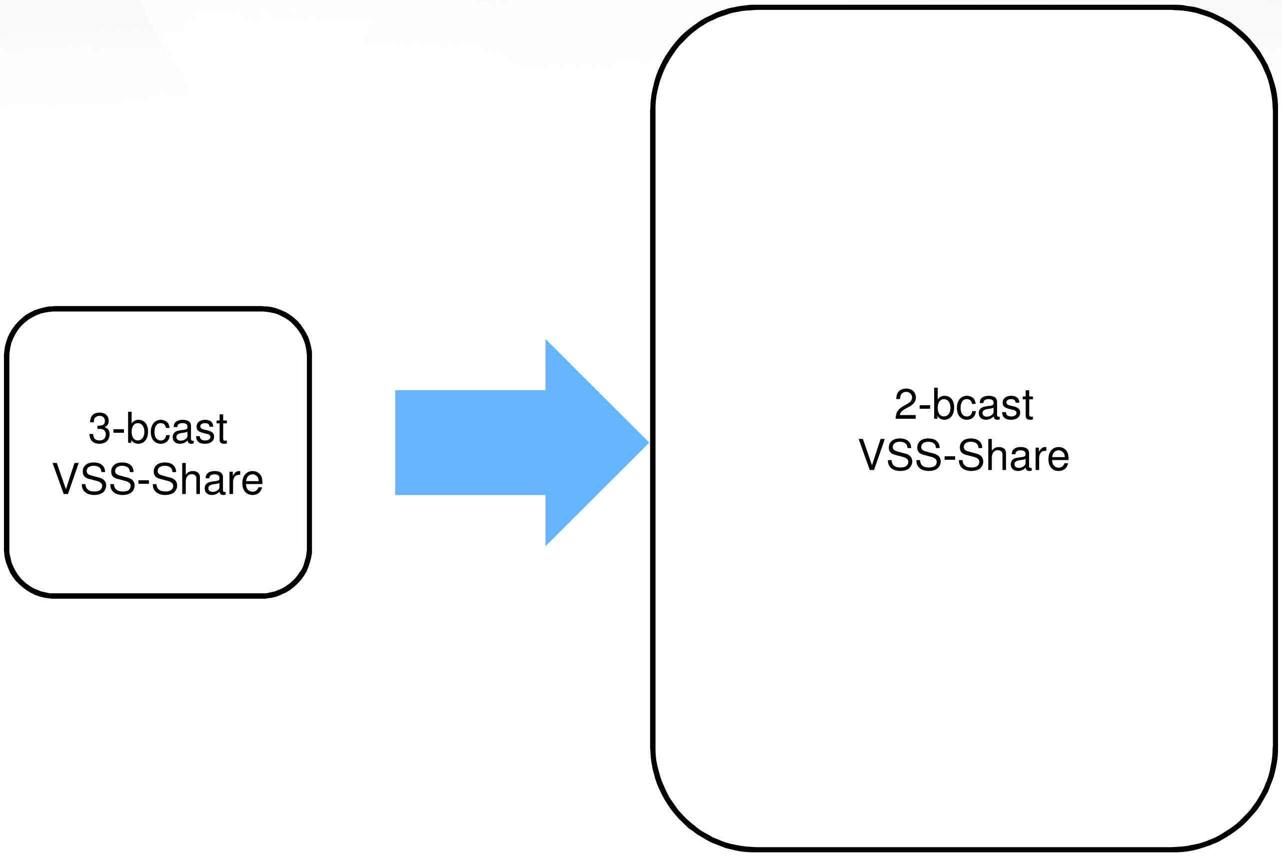
3-bcast
VSS-Share

Prepare setups for
Modercast with
1 round of **broadcast**
(based on [HR13])

**Moderated**
**by the Dealer**
3-bcast
VSS-Share

**Broadcast** $f_i$. If > n/2
$f_i=1$ ⟶ success.

# From 3 to 2 Broadcast Rounds



# Related Work (VSS)

- [RB89,Rab94]: (7,0)-bcast O(1)-round VSS
- [KPC10]: (2,2)-bcast (3,2)-round VSS
  - Exponential time; not (apparently) linear
  - (3,2)-bcast (4,2)-round poly-time VSS protocol; linear?
- [HR13]: (1,0)-bcast **O(n)**-round VSS
  - Linear no. of rounds → not ideal for natural VSS app's
- [KK07,Koo07,KKK08]: Role of b'cast in VSS and MPC
  - Reduce overall no. of rounds when b'cast is simulated over p2p channels

# Our Results

RECALL

- VSS with **two** b’cast rounds, constant overall rounds
  - First linear VSS protocol enjoying these features
  - (2,0)-bcast (20,1)-round VSS protocol

- Constant-round *pseudosignatures* [PW96]
  - Unconditionally secure *anonymous channel* (aka DC-nets [Chaum’88])
  - Black-box use of VSS; same b’cast complexity



YAHOO!

# Our Results

RECALL

- VSS with **two** b'cast rounds, constant overall rounds
  - First linear VSS protocol enjoying these features
  - (2,0)-bcast (20,1)-round VSS protocol

- Constant-round *pseudosignatures* [PW96]
  - Unconditionally secure *anonymous channel* (aka DC-nets [Chaum'88])
  - Black-box use of VSS; same b'cast complexity



YAHOO!

# Constant-Round Pseudosignatures

# *Pseudosignatures* [PW96]

- *Information-theoretic* signature scheme…
  - For a fixed-in-advance set of players
  - Verification **keys** are kept secret
  - Needs physical broadcast setup
  - Only *bounded transferability* of signatures

- Once we have them, we can implement *authenticated* broadcast protocol (e.g., [DS83, KK06]; tolerate $n > t$)

→ No more physical broadcasts required!



YAHOO!

# *Pseudosignatures* [PW96]

- Based on **sender-anonymous channel** [Chaum88]:

$msg_1$ →
$msg_2$ →
$msg_3$ →
$msg_4$ →

Anonymize messages (e.g., sort or shuffle)

→ { $msg_2$, $msg_4$, $msg_1$, $msg_3$ }

# Anonymous Channel: Security Requirements

- Even a cheating Receiver learns no more about honest senders' inputs than the multiset of them (Anonymity)
- Honest Receiver correctly gets all honest messages (Correctness)
- Cheating players have *zero information* on value of honest players' messages, for honest Receiver (Privacy)
- Cheating players' messages are independent of honest players' messages, for honest Receiver (Non-Malleability)

# Our Anonymous Channel

- The Idea: *Throwing Darts* [Hag91]

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **$P_1$:** | 0 | 0 | 0 | 0 | $k_1$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **$P_2$:** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $k_2$ | 0 | 0 |
| **$P_3$:** | 0 | $k_3$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |



YAHOO!

# Our Anonymous Channel


- The Idea: *Throwing Darts* [Hag91]

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **$P_1$:** | 0 | 0 | 0 | 0 | $k_1$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **$P_2$:** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $k_2$ | 0 | 0 |
| **$P_3$:** | 0 | $k_3$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Each player commits to vector, and gives *ZK proof* that committed vector is mostly zeroes
- Accepted (committed) vectors are added coordinate-wise:

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | $k_3$ | 0 | 0 | $k_1$ | 0 | 0 | 0 | 0 | 0 | $k_2$ | 0 | 0 |



YAHOO!

# From Anonymous Channel to Pseudosignatures

- Every party sends random keys, anonymously, to Signer. Repeat the process "several" (say, $p$) times.
- Signer receives $p$ ***signature blocks*** of keys $B_1 = ((a_{11},b_{11}),\ldots,(a_{1n},b_{1n})), \ldots, B_p = ((a_{p1},b_{p1}),\ldots,(a_{pn},b_{pn}))$

$$\begin{aligned}\text{Signature}(M) = &(a_{11}M \oplus b_{11}, \ldots, a_{1n}M \oplus b_{1n}),\\ &(a_{21}M \oplus b_{21}, \ldots, a_{2n}M \oplus b_{2n}),\\ &\ldots\\ &(a_{p1}M \oplus b_{p1}, \ldots, a_{pn}M \oplus b_{pn}).\end{aligned}$$

- 1st verifier: Given $(M, \sigma)$, verify *all* blocks have correct $aM \oplus b$
- 2nd verifier: Verify *most* blocks have correct $aM \oplus b$

# Our Anonymous Channel (cont'd)

- AnonChan implements an anonymous channel for $t < n/2$, using only black-box access to a linear VSS protocol
- Protocol is constant-round, and uses no additional broadcast rounds beyond those required by VSS
- Broadcast complexity: $B_{share} + B_{rec}$
- Our VSS protocol: $B_{share} = 2$, $B_{rec} = 0$


YAHOO!

# Related Work (Anonymous Channels, Pseudosig's)

- [Chaum88]: Introduced DC-nets; passive adversary
- [CR91]: Unconditionally secure signatures; only one transfer
- [PW96]: Introduced pseudosig's; $\Omega(n^2)$ rounds ($t < n$)
- Cryptographic constructions of anonymous channels:
  - [vABH03]: "k-anonymity;" also "dart-throwing;" not *reliable*
  - [GJ06]: "collisions" not considered; *malleable*
- [SHZI02, BTHR07] Alternative pseudosig. construction, based on random low-degree multi-variate polynomials
  - Not constant-round; not domain independent – only sign msgs from underlying field
  - More communication efficient



YAHOO!

# Putting It All Together: $\pi_f$

- **Preprocessing phase:**
  - Parties invoke AnonChan with each $P_i$ acting as receiver for many sessions in parallel, to generate a pseudosig' setup for future broadcasts
    - Parties use VSS protocol → 2 b'cast rounds
  - Parties now leverage information-theoretic PKI to generate sufficiently many random multiplication triples (using, e.g., constant-round authenticated protocol [Koo07])
- **Computation phase:**
  - To compute circuit, parties first share their inputs using VSS, replacing calls to b'cast with p2p authenticated b'cast protocol
  - Parties use the multiplication triples to evaluate any arithmetic circuit w/o further use of b'cast, in $O(D)$ rounds

# Summary

- VSS with **two** b’cast rounds, constant overall rounds
  - First linear VSS protocol enjoying these features
  - (2,0)-bcast (20,1)-round VSS protocol
- Constant-round *pseudosignatures*
  - Unconditionally secure *anonymous channel* (aka DC-nets)
  - Black-box use of VSS; same b’cast complexity

# References


- J. Garay, C. Givens, R. Ostrovsky and P. Raykov, “Broadcast (and Round) Efficient Verifiable Secret Sharing.” *Proc ICITS 2013.* Cryptology ePrint Archive: http://eprint.iacr.org/2012/130.

- J. Garay, C. Givens, R. Ostrovsky and P. Raykov, “Fast and Unconditionally Secure Anonymous Channel.” In submission.

YAHOO! LABS

Science-Driven Innovation