Probabilistic Termination and Composability of Cryptographic Protocols*

[Crypto '16]

Ran Cohen (BIU)
Sandro Coretti (ETH Zurich)
Juan Garay (Yahoo Research)
Vassilis Zikas (RPI)

* Slides by Ran Cohen

Motivation

Given: Protocol with *expected* O(1) running time

Motivation

Given: Protocol with *expected* O(1) running time
(e.g., geometric distribution)

Motivation

Given: Protocol with *expected* O(1) running time

What's the expected running time of *n* parallel instances?

Motivation

Given: Protocol with *expected* $O(1)$ running time

What's the expected running time of n parallel instances?

$\Theta(\log n)$ rounds

Motivation

Given: Protocol with *expected* O(1) running time

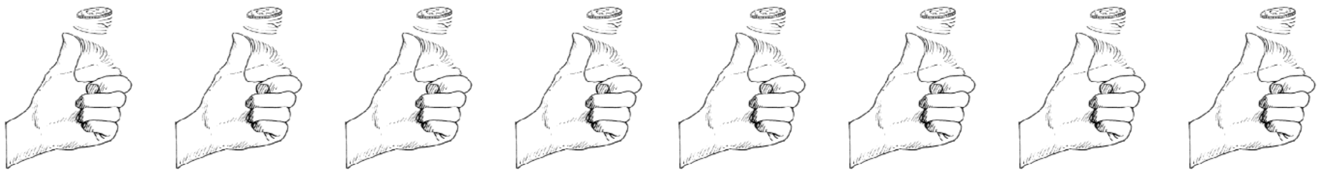
What's the expected running time of *n* parallel instances?

$\Theta(\log n)$ rounds

<u>Example:</u> Coin flipping

- Stand-alone coin flip: Pr(*heads*) = ½
  Output is *heads* in expected 2 rounds

- Flipping in parallel *n* coins, each coin until *heads*
  Expected log *n* rounds



Motivation (2)

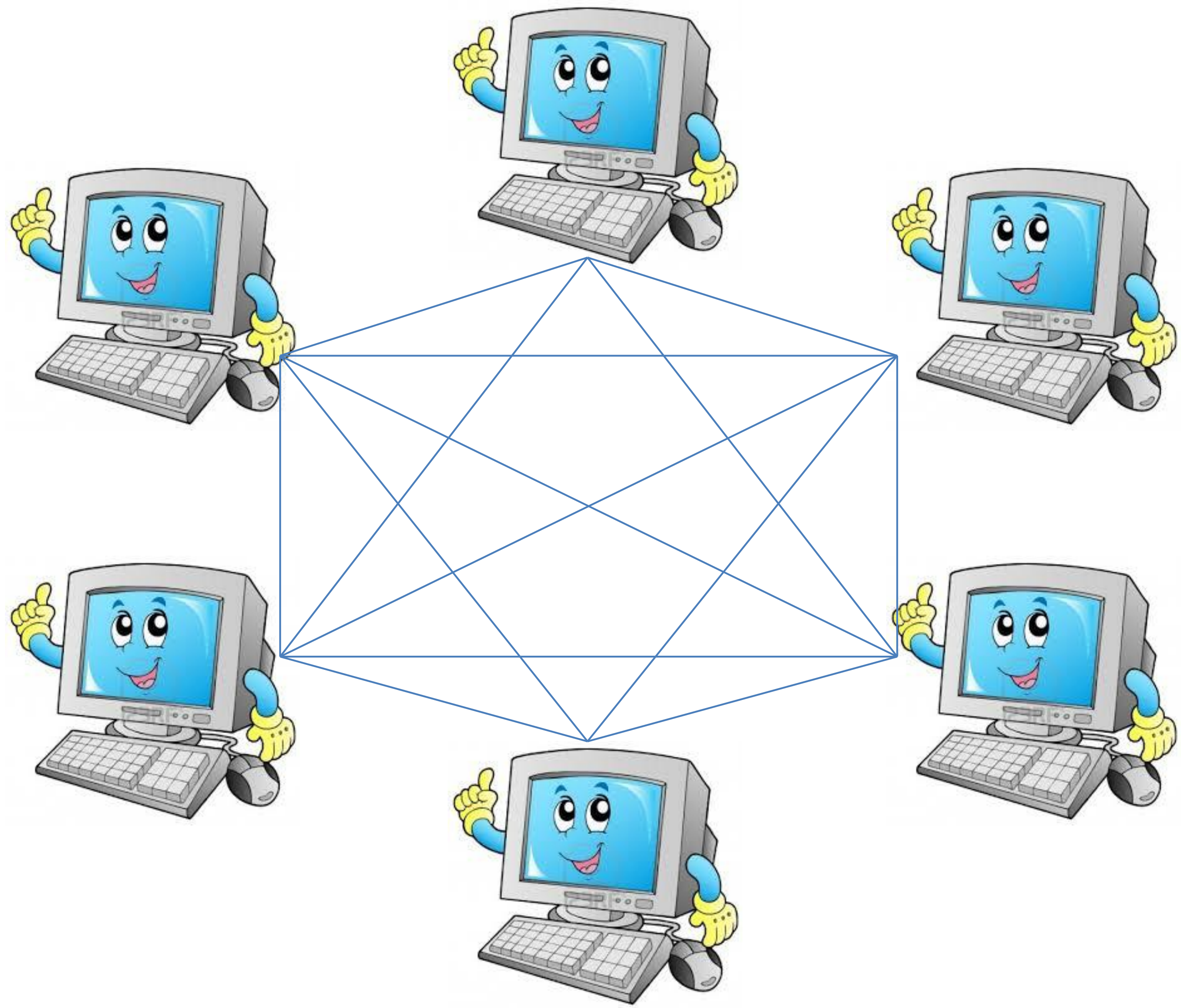
- The mathematical expectation of the maximum of *n* random variables does not necessarily equal the maximum of their expectations [BE’03,Eis’08]
- Fast implementations of broadcast protocols run in expected O(1) time
  → parallel executions no longer constant (nor fixed)
  → non-simultaneous termination
- Composition — how to simulate probabilistic termination?

This Work

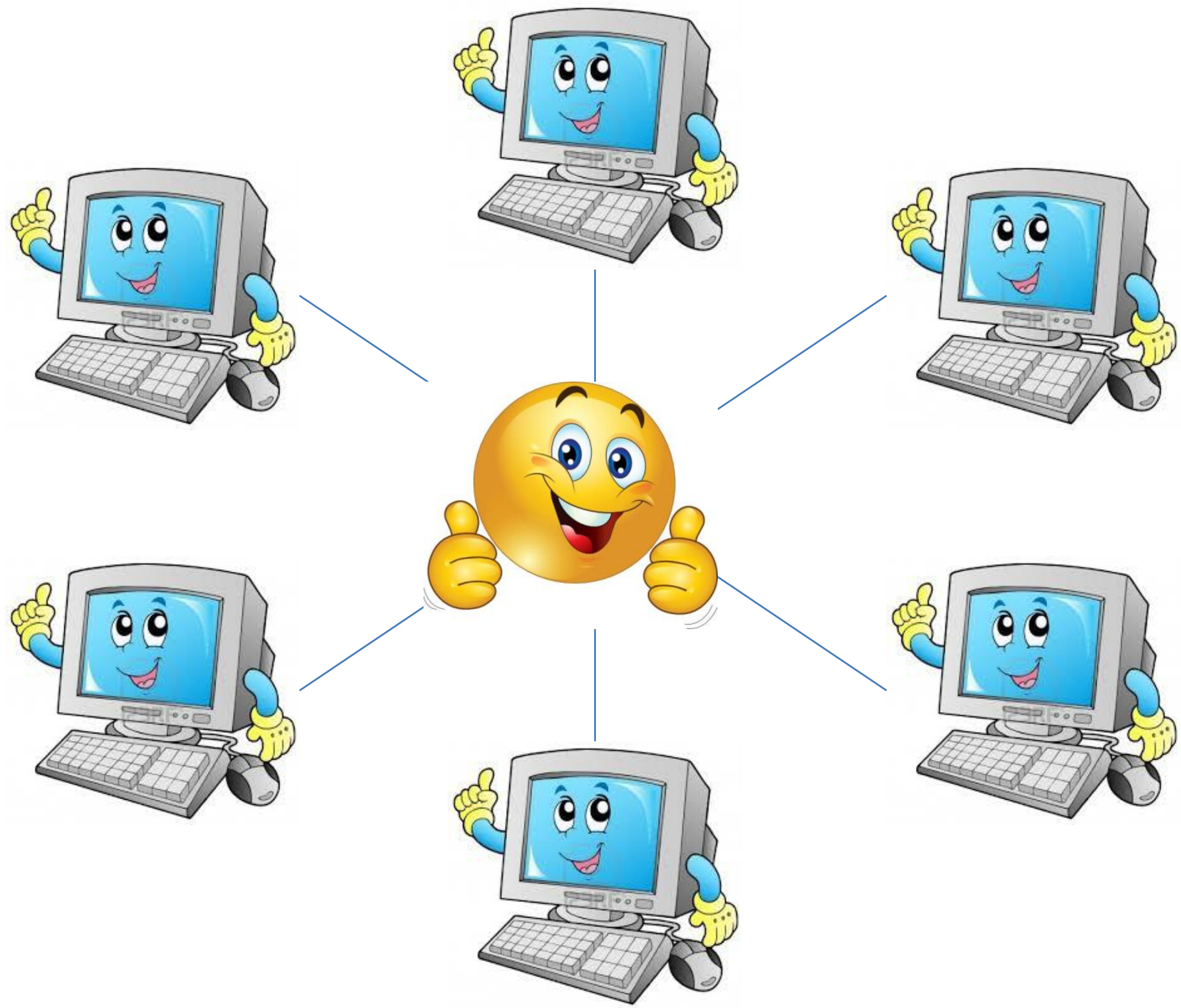
We consider composability of cryptographic protocols with *probabilistic termination*

- Framework for designing cryptographic protocols in stand-alone fashion and compiler to fast composition in the UC framework
- Perfect, adaptively secure protocols in the P2P model
  1) BA with expected $O(1)$ rounds
  2) Parallel broadcast with expected $O(1)$ rounds
  3) SFE with expected $O(d)$ rounds

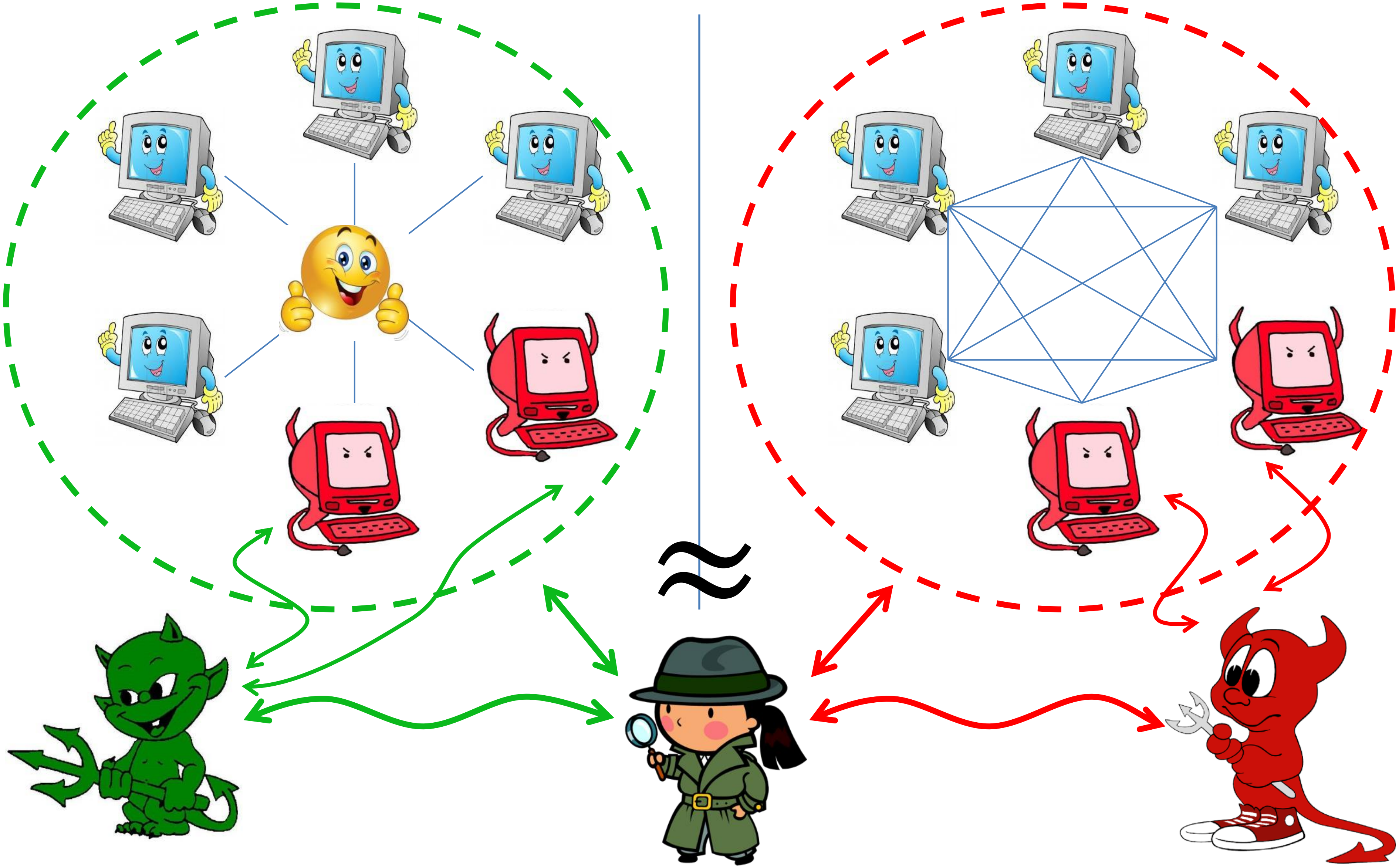
Secure Multiparty Computation (MPC)



Ideal World/"Functionality"

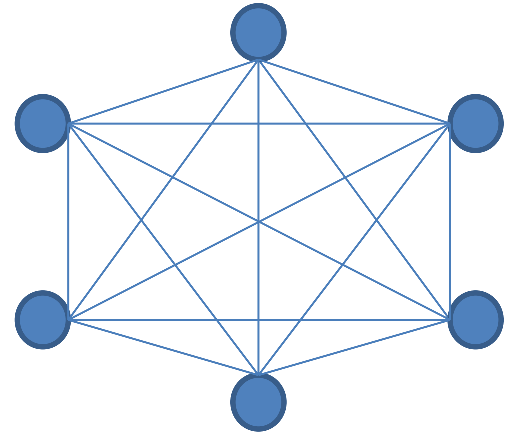


Simulation-based Security



Communication Model

- Point-to-point model
  - Secure (private) channels between the parties (*Secure Message Transmission*)



- Broadcast model
  - Additional *broadcast channel*

- Synchronous communication
  - Bounded delay
  - Global clock
  - Protocol proceeds in rounds
  - Guaranteed termination

Instantiating Broadcast Channel

Broadcast

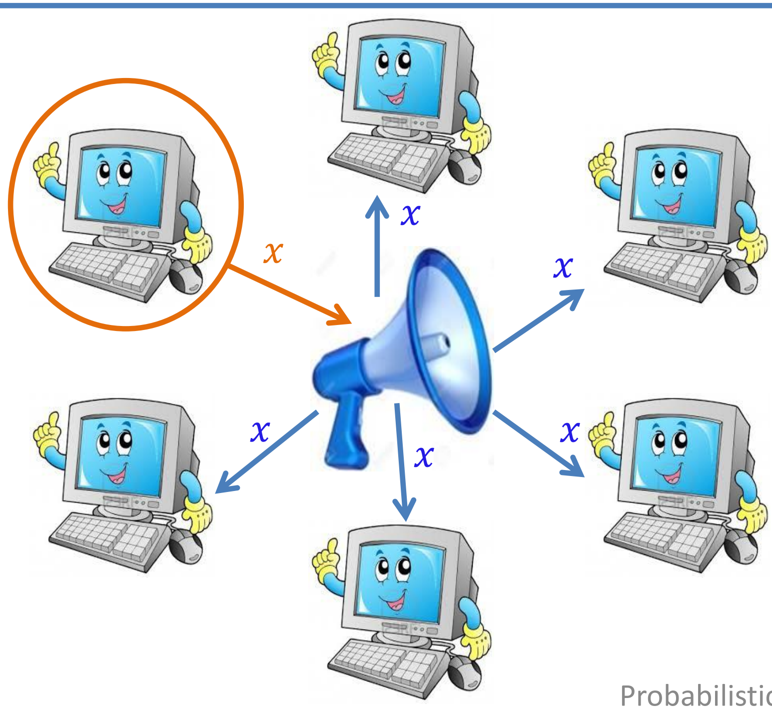
Sender with input x

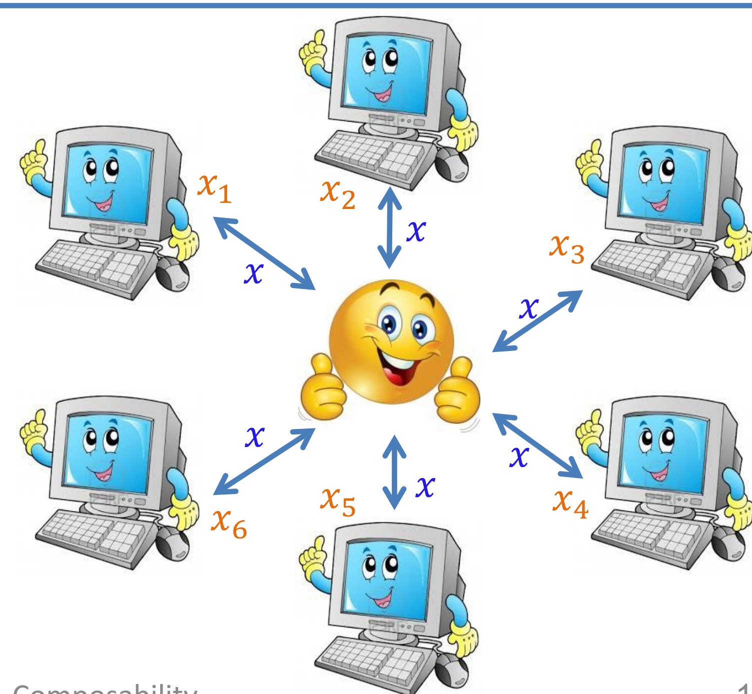
- **Agreement**: all honest parties output the same value
- **Validity**: if the sender is honest, the common output is x

Byzantine agreement

Each P_i has input x_i

- **Agreement**: all honest parties output the same value
- **Validity**: if all honest parties have the same input x, the common output is x





Instantiating Broadcast Channel

Broadcast

Sender with input x

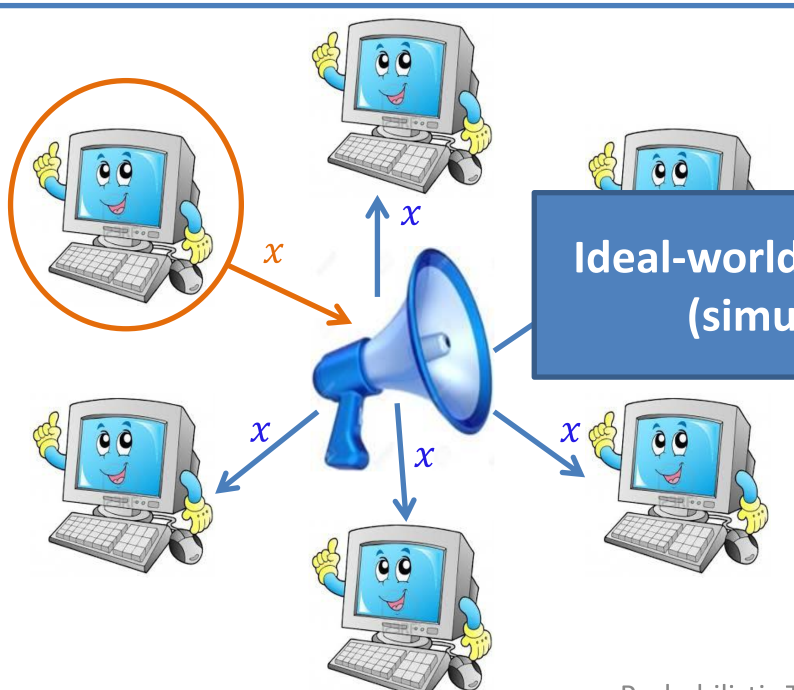
- **Agreement**: all ho[…] output the same va[…]
- **Validity**: if the sender is honest, the common output is x

Real-world security definition (property-based)

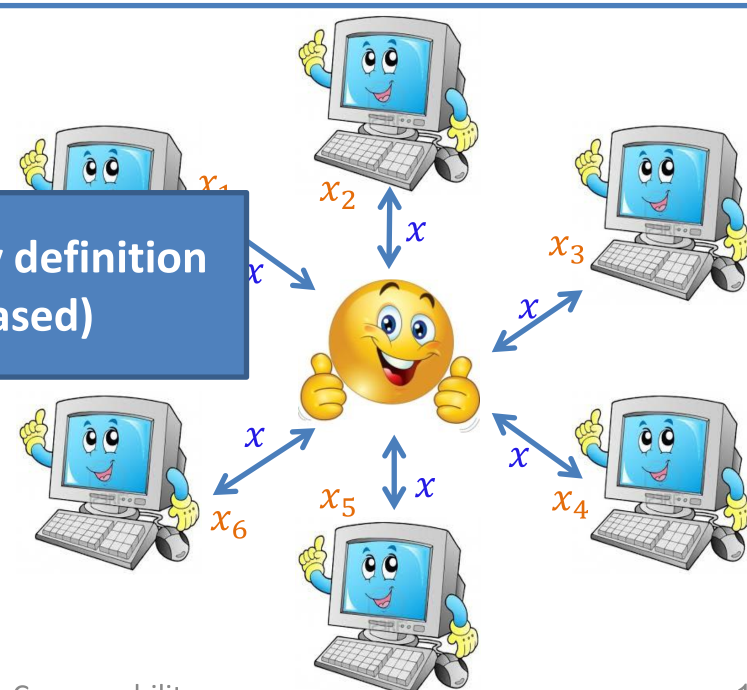
Byzantine agreement

[…] input x_i

- […]**nt**: all honest parties […]e same value
- **Validity**: if all honest parties have the same input x, the common output is x



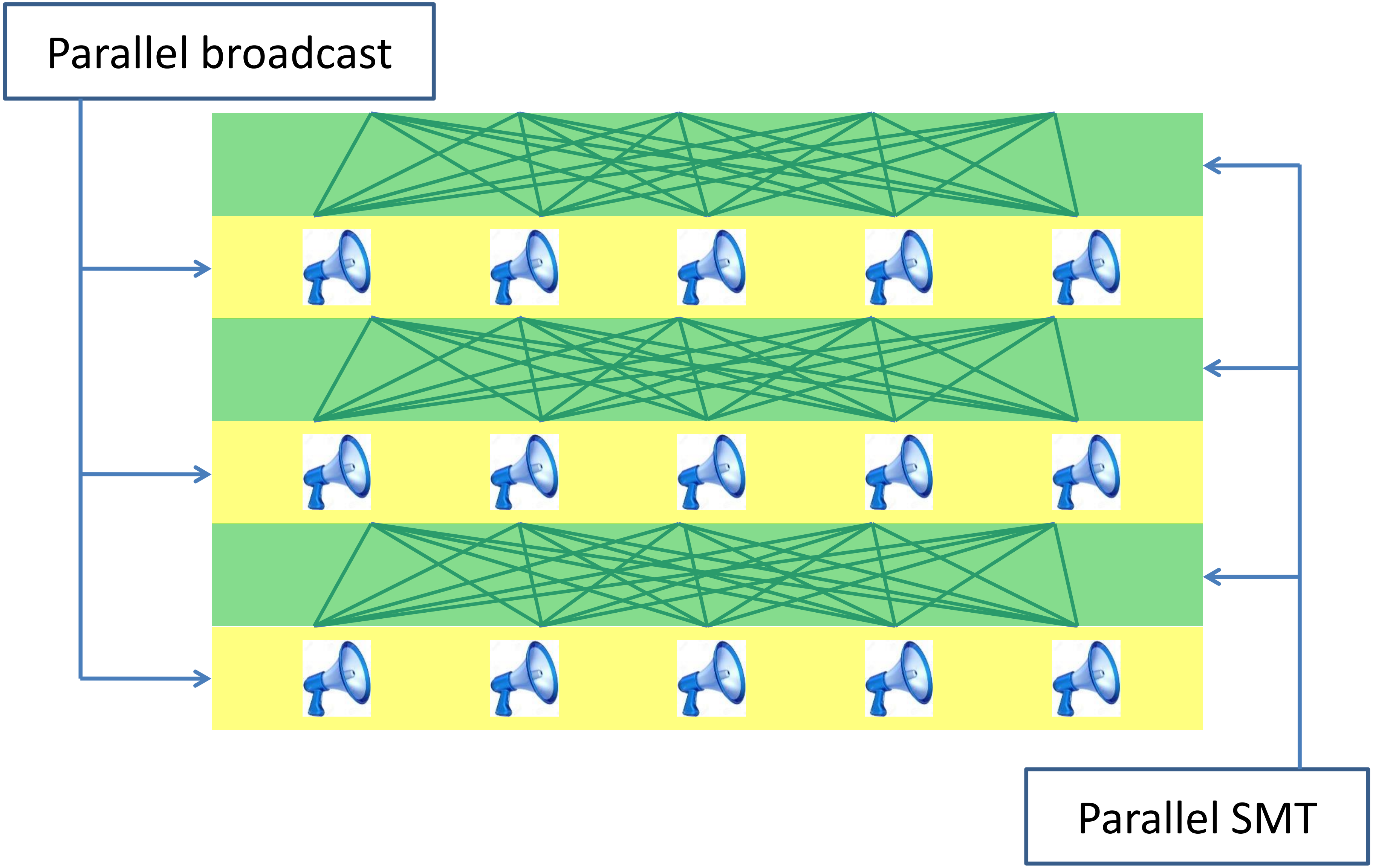
Ideal-world security definition (simulation-based)



Feasibility of MPC with Broadcast

- Classical result [BGW'88]
  - Share-Compute-Reveal paradigm
  - Perfect, adaptively secure for $t < n/3$
  - Concurrently composable
  - $O(d)$ rounds, $O(d)$ broadcasts
- Improving communication complexity
  - E.g., player-elimination framework [HMP'00] [HM'01] [BH'06] [HN'06] [DN'07] [BH'08] [BFO'12]
  - $O(d+n)$ rounds
- Improving round complexity
  - $O(d)$ rounds, 1 broadcast [KK'07]

Protocols with Broadcast

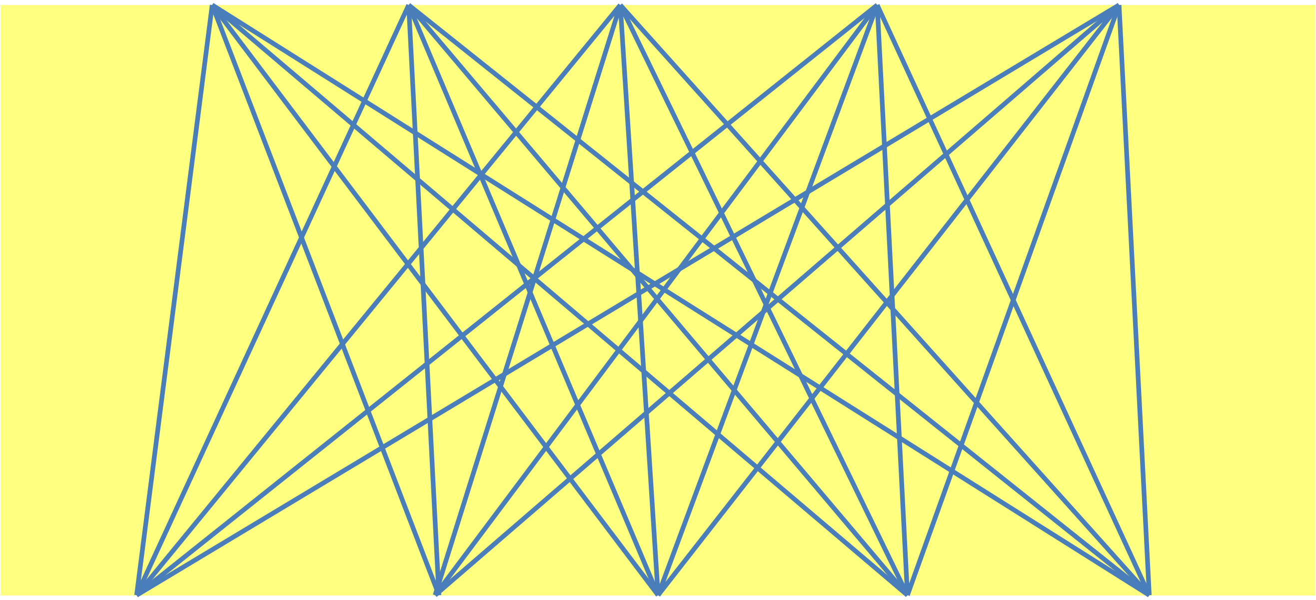


Deterministic BA/Broadcast Protocols

- Perfect and adaptive security for $t < n/3$ [BGP'89] [GM'93] [HZ'10]
- Deterministic Termination (DT) – single output round
- Compose nicely
- Require $O(n)$ rounds – this is inherent [FL'82]

Deterministic BA/Broadcast Protocols

- Perfect and adaptive security for $t < n/3$ [BGP'89] [GM'93] [HZ'10]
- Deterministic Termination (DT) – single output round
- Compose nicely
- Require $O(n)$ rounds – this is inherent [FL'82]



Probabilistic BA/Broadcast Protocols

Randomization can help [Ben-Or'83] [Rabin'83]

Binary BA protocol [Feldman, Micali'88]

- Proceeds in phases until termination
- In each phase each party has an input bit
  - If all honest parties start the phase with the same bit, they terminate at the end of the phase
  - Otherwise, with probability $p > 0$ all honest parties agree on the same bit at the end of the phase (and terminate in the next phase)
  - With probability $1 - p$
    - No agreement at the end of the phase, or
    - the adversary makes some of the honest parties terminate; the remaining parties will terminate in the next phase

Probabilistic BA/Broadcast Protocols (2)

- [FM'88] has *Probabilistic Termination* (PT):
  - Expected $O(1)$ rounds
  - No guaranteed termination: statistical security (for PPT parties)
  - No simultaneous termination: honest parties might terminate at different rounds [DRS'90]
  - All honest parties terminate in a constant window
- Extends to multi-valued BA [Turpin, Coan'84]
  - Two additional rounds
- Perfect security [Goldreich, Petrank'90]
  - Best of both worlds
- Variant for parallel broadcast [Ben-Or, El-Yaniv'03]

What's Missing?

- All PT broadcast protocols are proven secure using a property-based definition
- Composition theorems require simulation-based proofs
- [KMTZ'13] defined a UC-based framework for synchronous DT protocols
- PT protocols are very delicate
  Many subtle issues not captured by [KMTZ'13]
- We introduce a framework for designing and analyzing PT protocols

Rest of the Talk

1. The Framework, Part I: Probabilistic Termination
   - Two-round canonical synchronous functionalities
   - Round-extension *wrappers*
   - Construct PT protocols when parties start at the same time

2. The Framework, Part II: Dealing with “Slack”
   - Adjust wrappers to deal with non-simultaneous start
   - Composition theorem
   - Construct PT protocols, when parties start during time window

3. Applications

The Framework

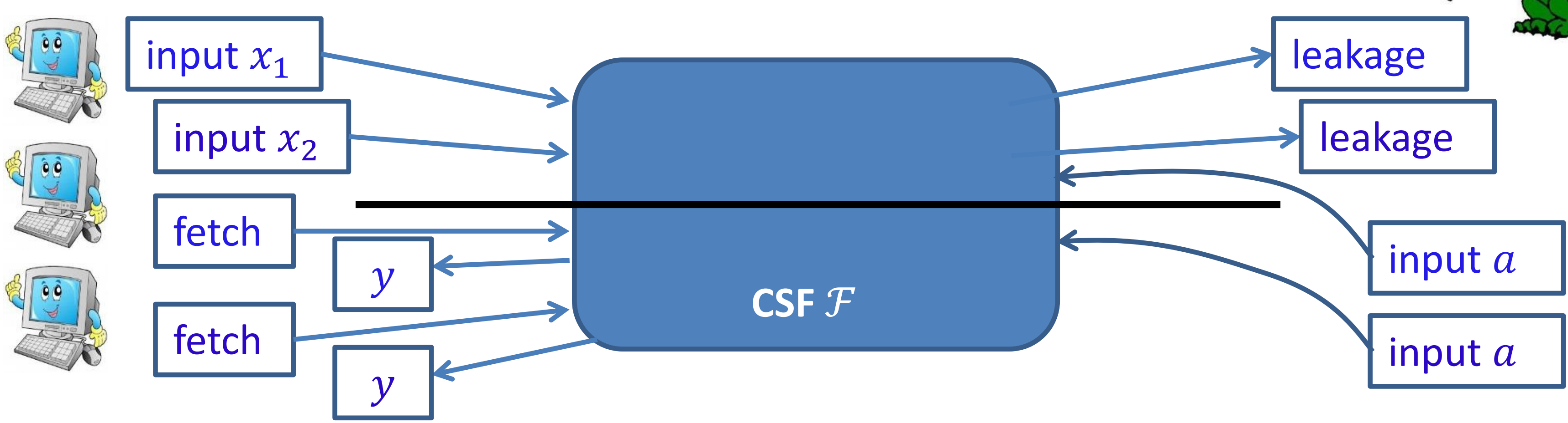
Part I: Probabilistic Termination

Synchronous Protocols in UC

- The environment can observe in which round parties terminate [KTMZ'13]
- Cannot hide the round complexity of hybrids
- In [KTMZ'13] each ideal functionality is parameterized by number of rounds
- Parties continuously request output and receive at the last round
- $\Rightarrow$ Parties in ideal world receive output at same round as in protocol execution in the real world

Canonical Synchronous Functionality

- Separate the function from the round structure
- A CSF consists of input round and output round
- Parameterized by
  - (Randomized) function $f(x_1, \ldots, x_n, a)$
  - Leakage function $l(x_1, \ldots, x_n)$



CSF Examples

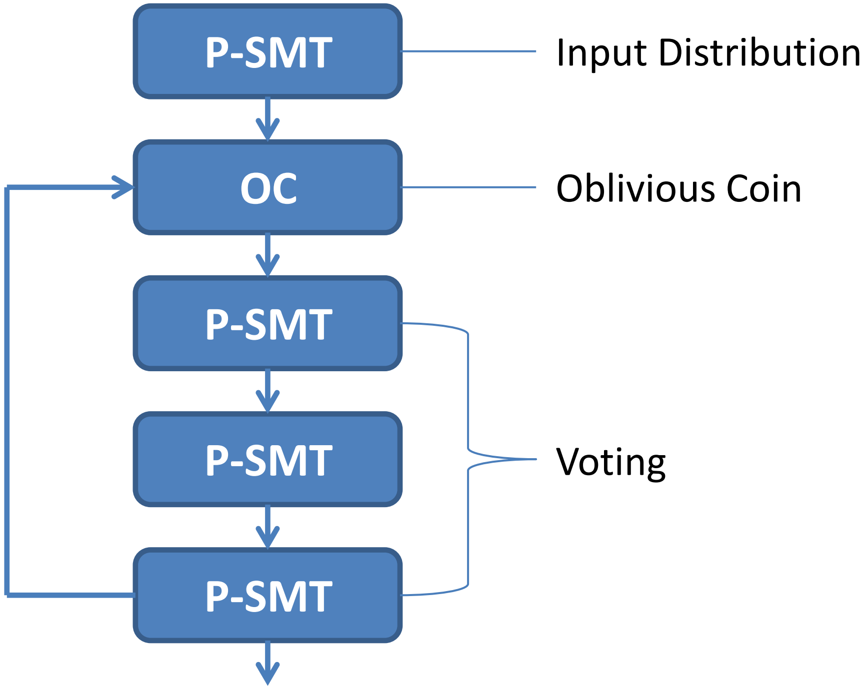
- **SMT**: P_i sends x_i to P_j
  - $f(x_1, \dots, x_n, a) = (y_1, \dots, y_n)$, s.t. $y_j = x_i$ and $y_k = \lambda$ $(k \neq j)$
  - $l(x_1, \dots, x_n) = \begin{cases} |x_i| \text{ if } P_j \text{ honest} \\ x_i \quad \text{if } P_j \text{ corrupted} \end{cases}$
- **Broadcast**: P_i broadcasts x_i
  - $f(x_1, \dots, x_n, a) = (x_i, \dots, x_i)$
  - $l(x_1, \dots, x_n) = |x_i|$

Parallel version

- **SFE**: parties compute a function g
  - $f(x_1, \dots, x_n, a) = g(x_1, \dots, x_n)$
  - $l(x_1, \dots, x_n) = (|x_1|, \dots, |x_n|)$
- **BA**:
  - $f(x_1, \dots, x_n, a) = \begin{cases} y \text{ if at least } n - t \text{ inputs are } y \\ a \text{ otherwise} \end{cases}$
  - $l(x_1, \dots, x_n) = (x_1, \dots, x_n)$

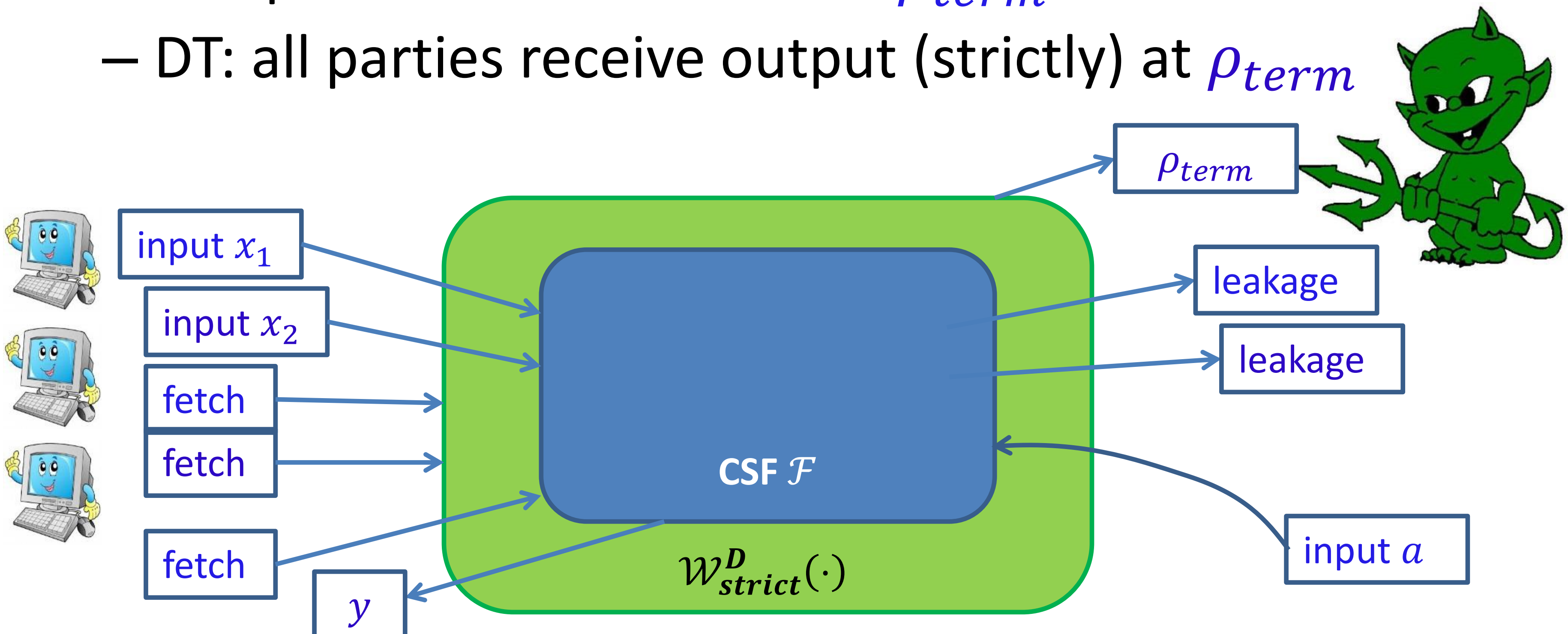
Synchronous Normal Form (SNF)

- SNF protocol:
  - In each round exactly one ideal functionality is called (as in stand-alone [Canetti'00])
  - All hybrids are (2-round) CSFs
- Example: Protocol π_{RBA} (based on [FM'87])



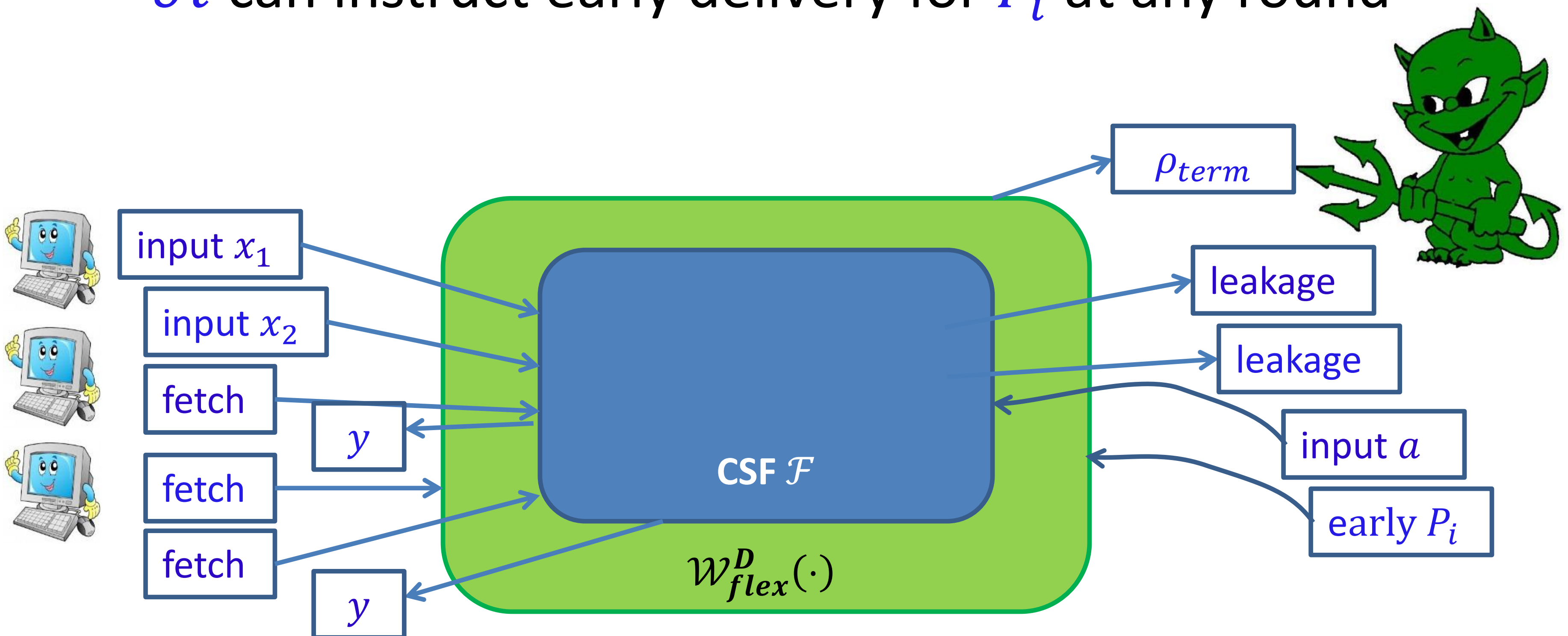
Extending Rounds (DT)

- Most functionalities cannot be implemented by two-round protocols
- Wrap the CSFs with *round-extension* wrappers
  - Sample a termination round $\rho_{term} \leftarrow D$
  - DT: all parties receive output (strictly) at ρ_{term}



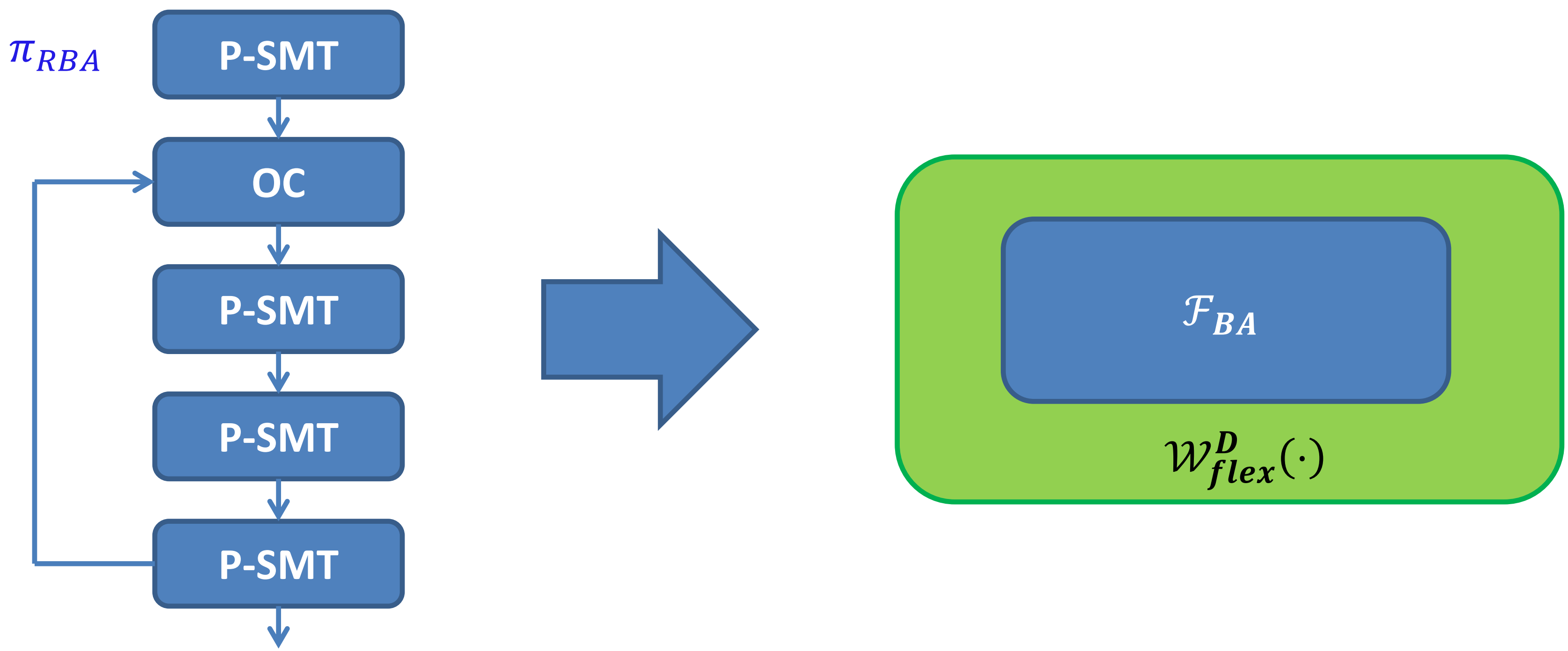
Extending Rounds (PT)

- PT: ρ_{term} is an upper bound
  - Sample a termination round $\rho_{term} \leftarrow D$
  - All parties receive output *by* ρ_{term} (flexible)
  - $\mathcal{A}$ can instruct early delivery for P_i at any round



Where Do We Stand?

Thm: Protocol π_{RBA} implements $\mathcal{W}^{D}_{flex}(\mathcal{F}_{BA})$ in the $(\mathcal{F}_{PSMT}, \mathcal{F}_{OC})$-hybrid model, for $t < n/3$,
assuming all parties start at the same round



The Framework

Part II: Dealing with “Slack”

Problem: Sequential Composition

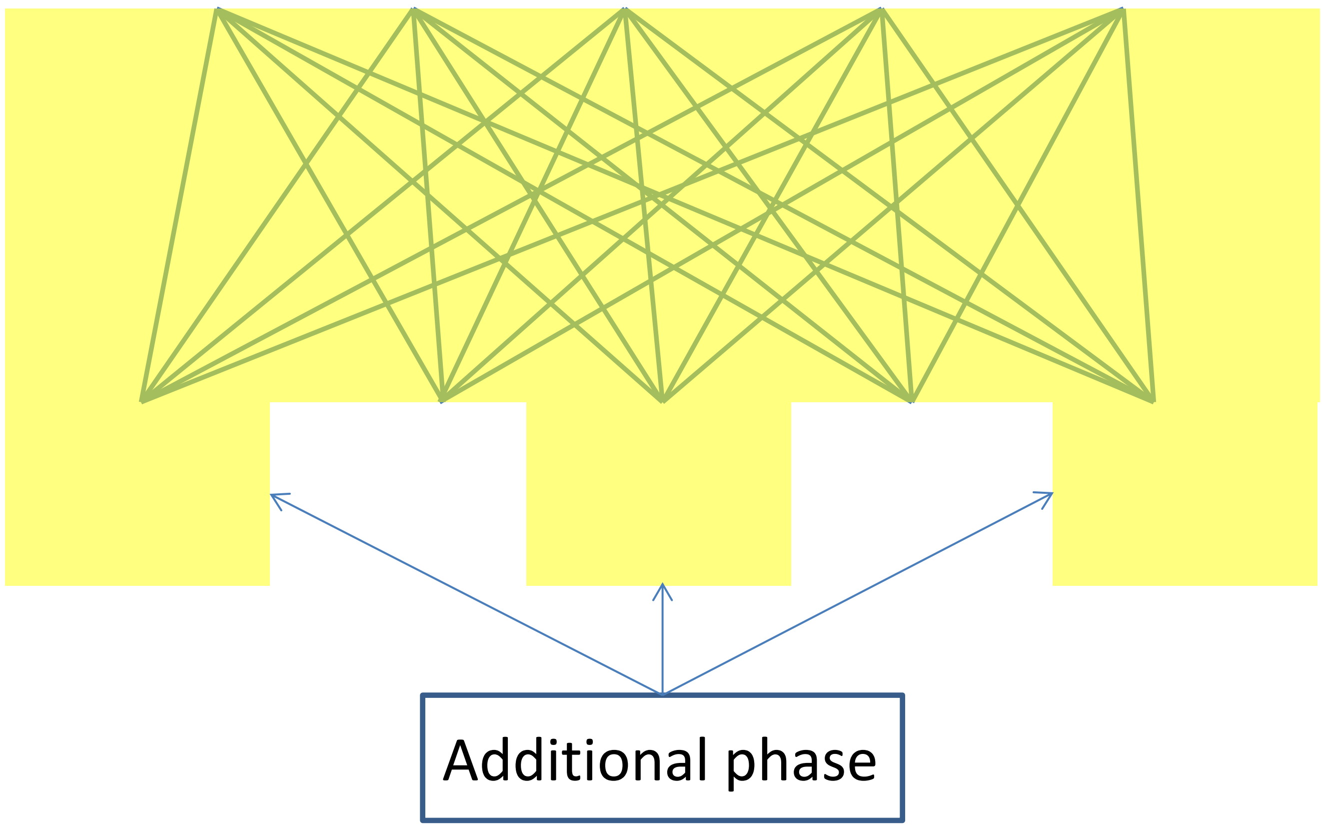
New execution starts **after all** parties finished previous one

With PT protocols, fast parties start new execution **before** slow parties finished previous execution

Problem: Sequential Composition

New execution starts **after all** parties finished previous one

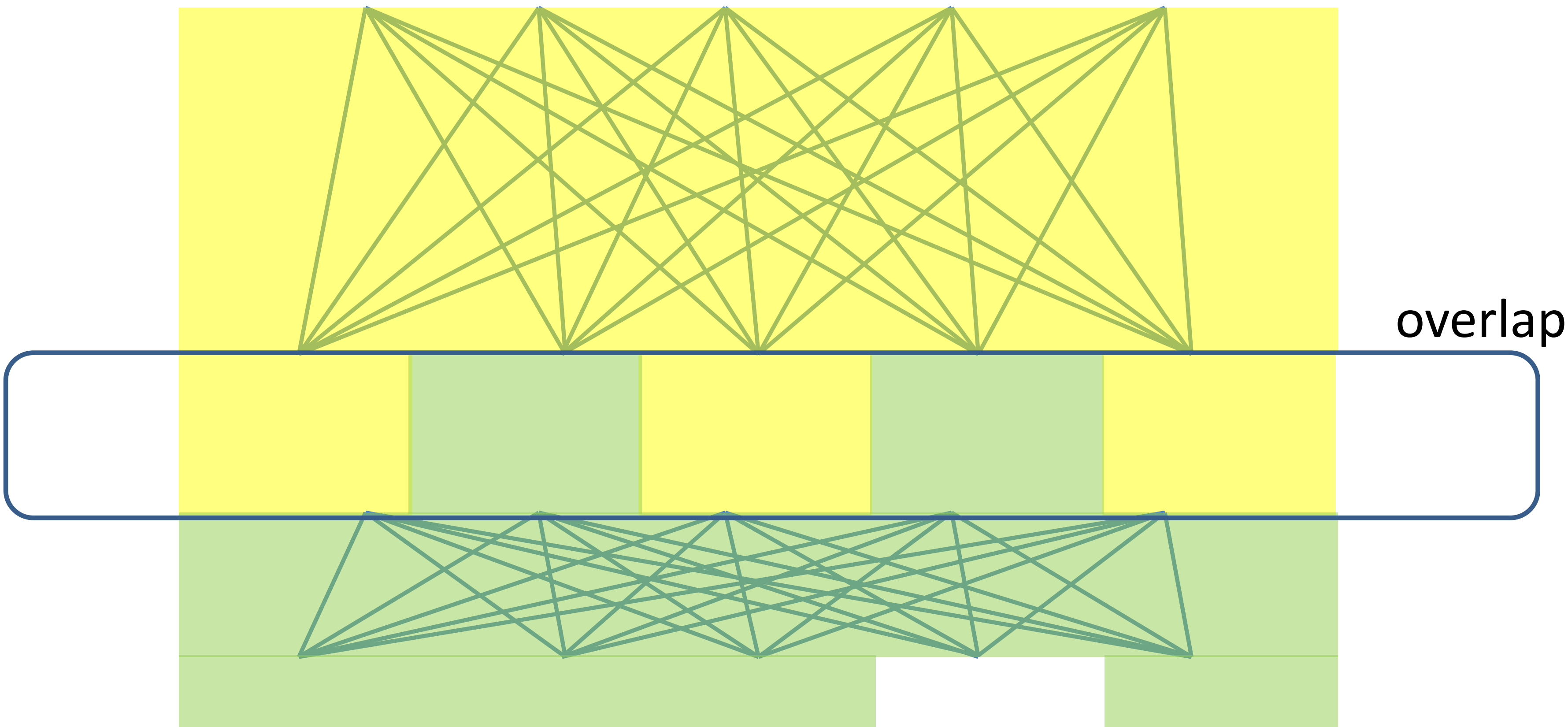
With PT protocols, fast parties start new execution **before** slow parties finished previous execution



Problem: Sequential Composition

New execution starts **after all** parties finished previous one

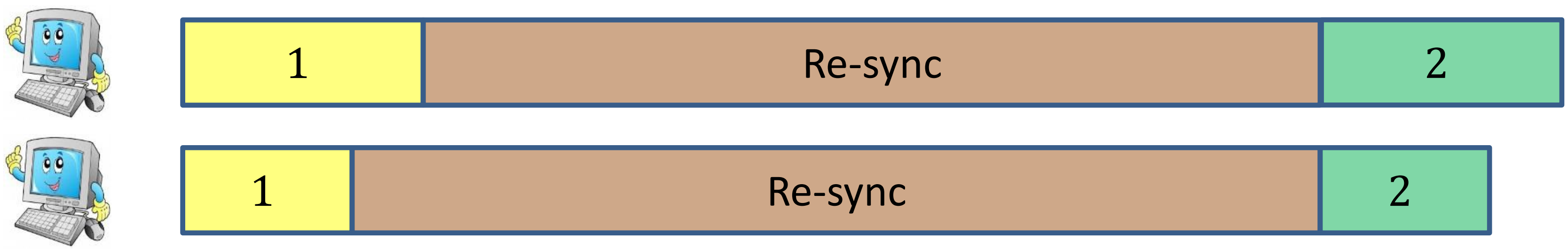
With PT protocols, fast parties start new execution **before** slow parties finished previous execution



Sequential Composition

Goal: ℓ sequential executions of expected $O(1)$ rounds protocols in expected $O(\ell)$ rounds

Naïve solution: wait until re-synchronized



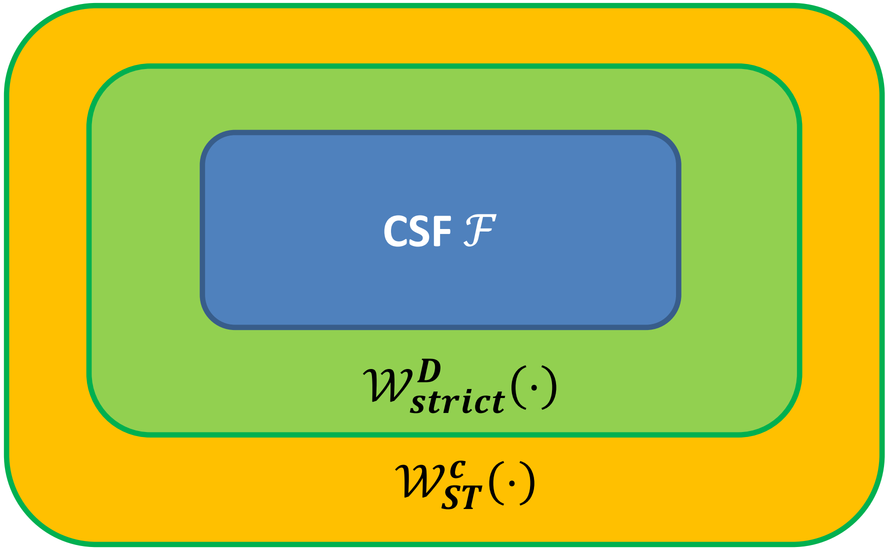
Sequential Composition: Solutions

Goal: ℓ sequential executions of expected $O(1)$ rounds protocols in expected $O(\ell)$ rounds

- [LLR'02] – adding re-synchronization points
  - Statistical security (inherent)
  - Static corruptions
  - Property-based security
- [BE'03] [KK'06]
  - Simpler solutions, partial proofs (no simulation)
- We introduce a generic compiler for PT protocols
  - Supports non-simultaneous start of the protocol
  - Reduces the slackness to 1
  - Simulation-based security – a composition theorem

"Slack" Tolerance

- **Main idea:** Make the overlap meaningless by adding "dummy" rounds
  - Assume slack of c rounds
  - Extend each round to $3c + 1$ rounds
  - Messages of P_i are queued and forwarded in cycles of $3c + 1$
- DT functionalities: wrap $\mathcal{W}^{D}_{strict}(\mathcal{F})$ with $\mathcal{W}^{c}_{ST}(\cdot)$
  - Each party runs the same number of rounds
  - The slack remains the same

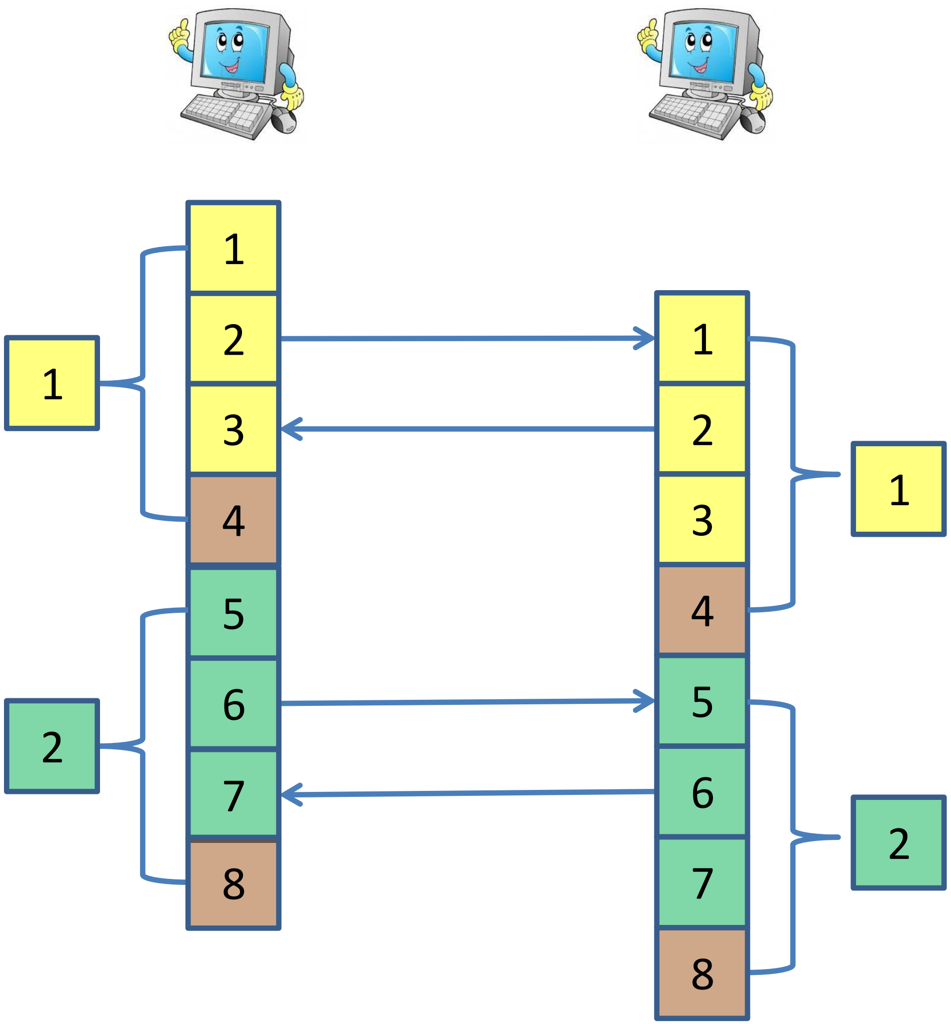


Non-Simultaneous Start

Each round extends to $3c + 1$ rounds:

- Listen for $2c + 1$ rounds
- Send in round $c + 1$
- Wait (without listening) for c rounds

Example: PSMT (c = 1)



Concurrent Composition

Round r messages

after round $r - 1$

before round $r + 1$

Each party proceeds in a locally sequential manner

Slack Tolerance and Reduction (PT)

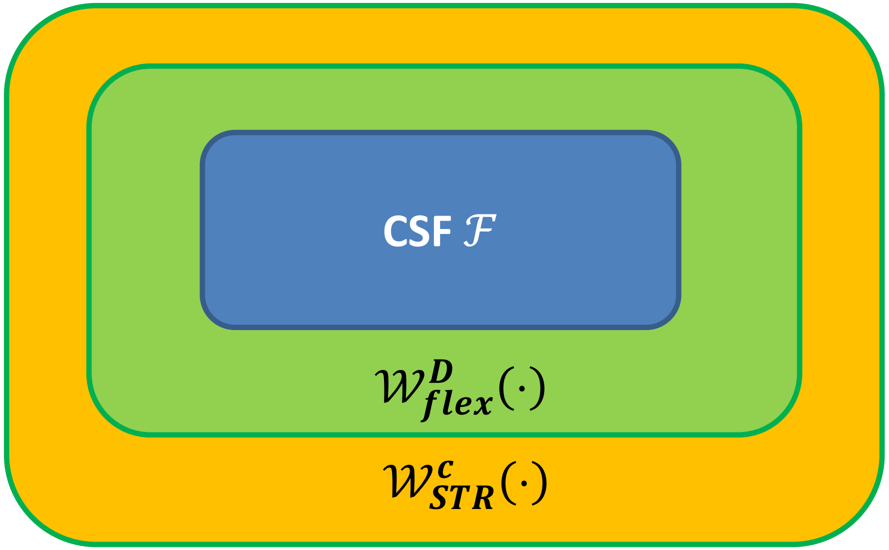
Hybrids introduce additional slack, rounds might blow-up

Use slack-reduction techniques [Bracha'84]

- Upon receiving output v, send (ok, v) to all the parties
- Upon receiving $t + 1$ messages (ok, v), accepts v
- Upon receiving $n - t$ messages (ok, v), terminates

Wrap $\mathcal{W}_{flex}^{D}(\mathcal{F})$ with $\mathcal{W}_{STR}^{c}(\cdot)$

Applies to <u>public-output</u> functionalities



Composition Theorem (Informal)

Denote $$\mathcal{W}_{DT}^{c,D}(\mathcal{F}) = \mathcal{W}_{ST}^{c}\left(\mathcal{W}_{strict}^{D}(\mathcal{F})\right)$$

$$\mathcal{W}_{PT}^{c,D}(\mathcal{F}) = \mathcal{W}_{STR}^{c}\left(\mathcal{W}_{flex}^{D}(\mathcal{F})\right)$$

Thm: Let $c \geq 0$ and $t < n/3$ (adaptive & perfect security)

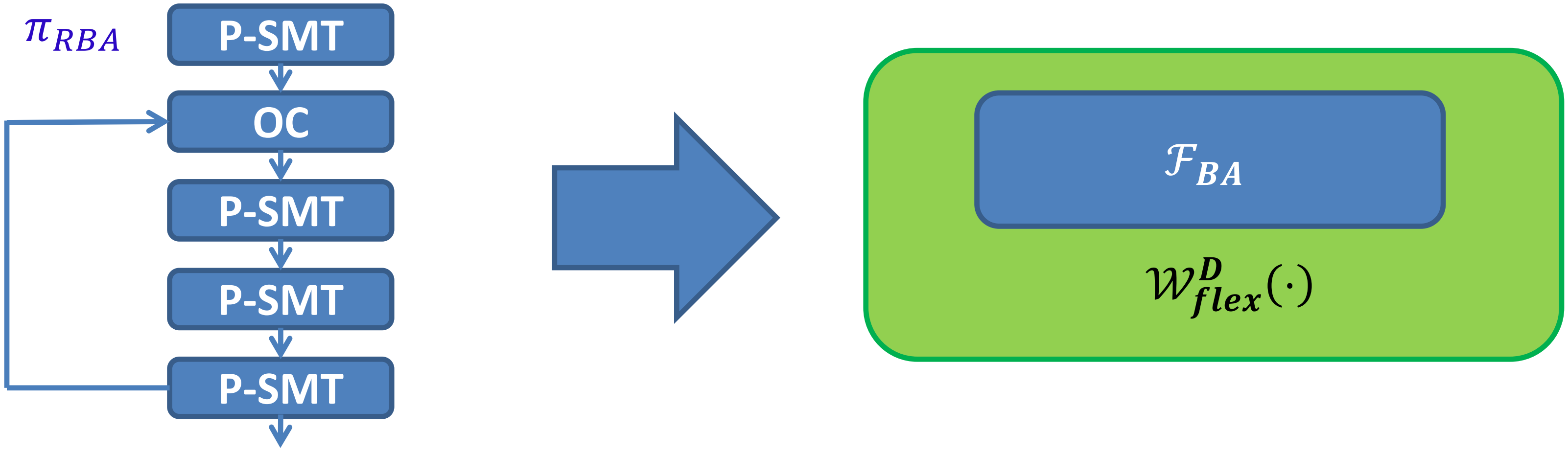
Let π be an SNF protocol implementing a wrapped CSF $\mathcal{W}_{flex}^{D}(\mathcal{F})$ in the $(\mathcal{F}_1, \ldots, \mathcal{F}_\ell, \mathcal{F}'_1, \ldots, \mathcal{F}'_m)$-hybrid model, assuming all parties start at the same round

Then, $\mathrm{Comp}^{c}(\pi)$ implements $\mathcal{W}_{PT}^{c,\tilde{D}}(\mathcal{F})$ in the $\left(\mathcal{W}_{PT}^{c,D_1}(\mathcal{F}_1), \ldots, \mathcal{W}_{PT}^{c,D_\ell}(\mathcal{F}_\ell), \mathcal{W}_{DT}^{c,D'_1}(\mathcal{F}'_1), \ldots, \mathcal{W}_{DT}^{c,D'_m}(\mathcal{F}'_m)\right)$-hybrid model, assuming all parties start within $c+1$ rounds

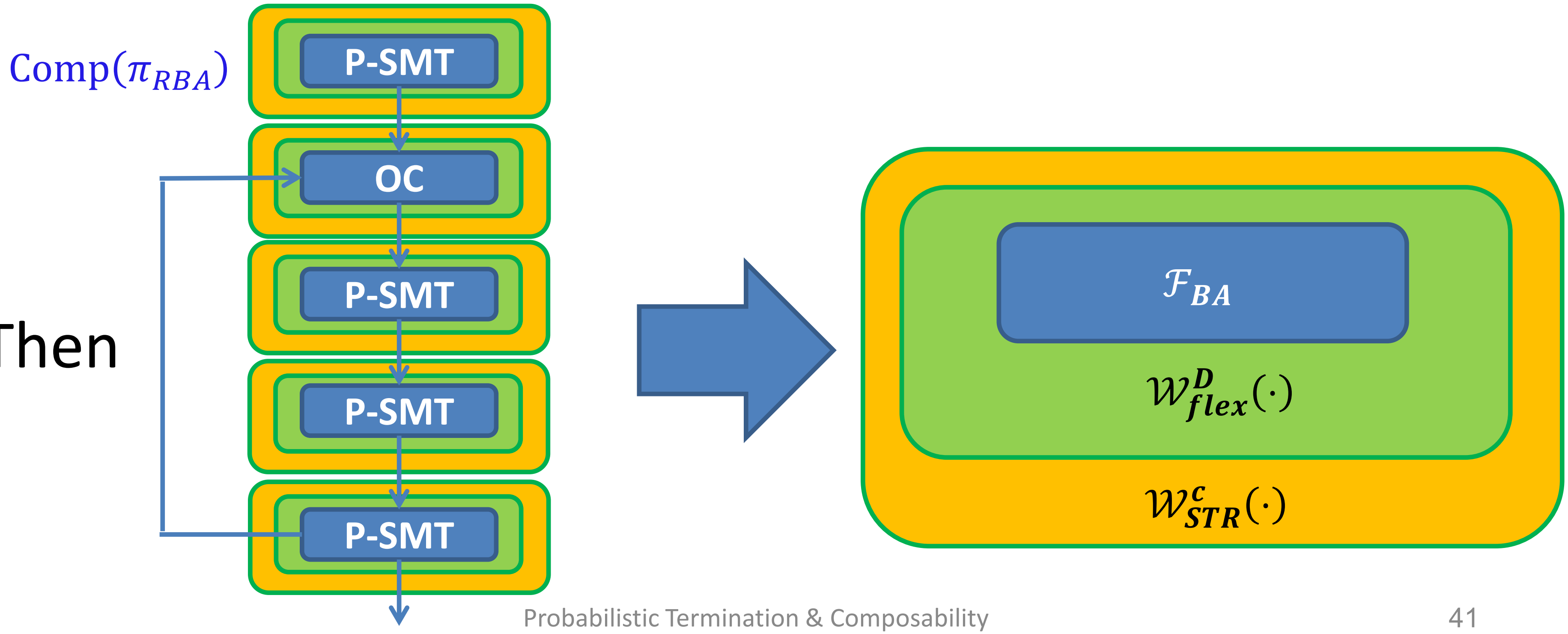
If each D_i (D'_i) has constant expectation then $\mathrm{Comp}^{c}(\pi)$ has (asymptotically) same round complexity as π, in expectation

Corollary

If



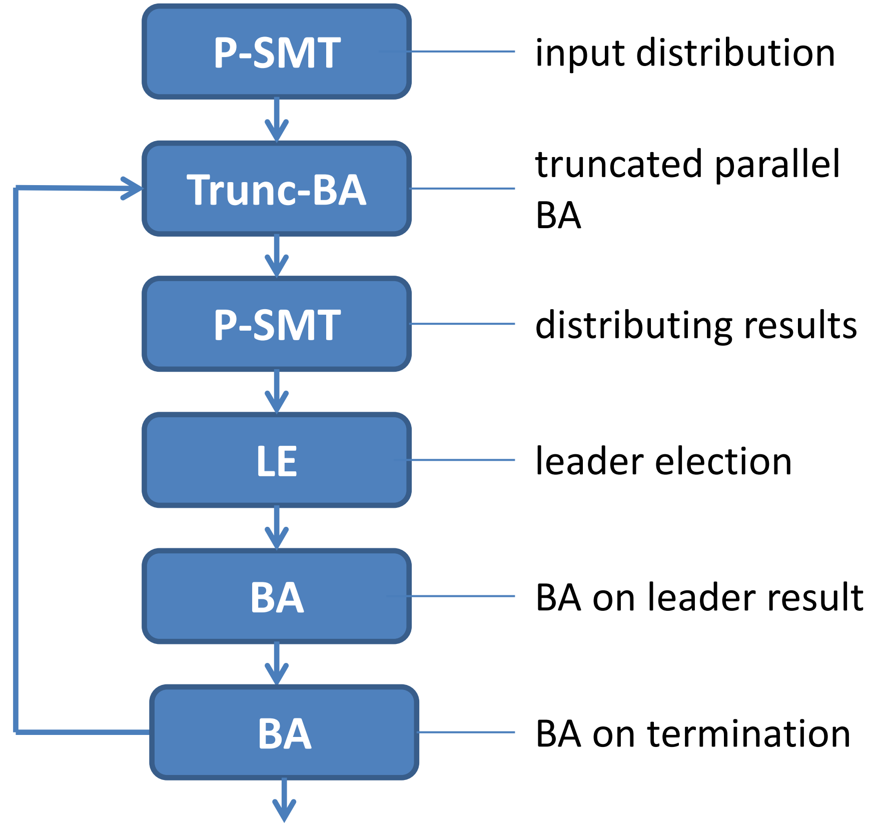
Then



Applications

Parallel Broadcast

- Running [FM'88] n times in parallel requires expected $\Theta(\log n)$ rounds
- Parallel broadcast in expected $O(1)$ [BE'03]
  - First round: each P_i distributes its input x_i
  - Proceeds in phases until termination



Parallel Broadcast (2)

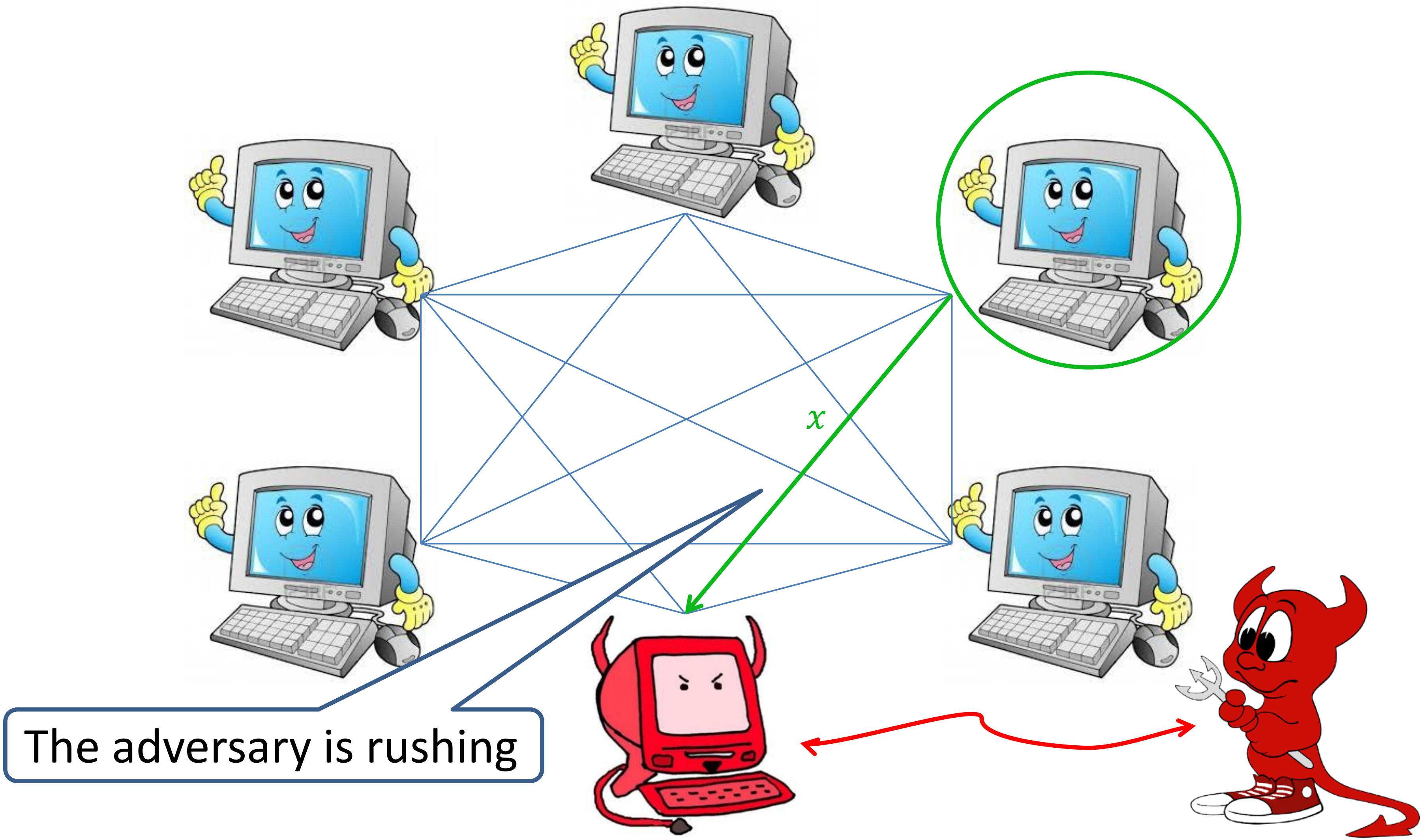
Thm [BE'03]: For appropriate parameters the protocol computes parallel broadcast in expected $O(1)$ rounds

Two issues:

1) No guaranteed termination: statistical security. We achieve perfect security (cf. [GP'90])
   - Run at most T phases
   - If not terminated, run a deterministic protocol
2) Adaptive security according to property-based definition (not simulation)

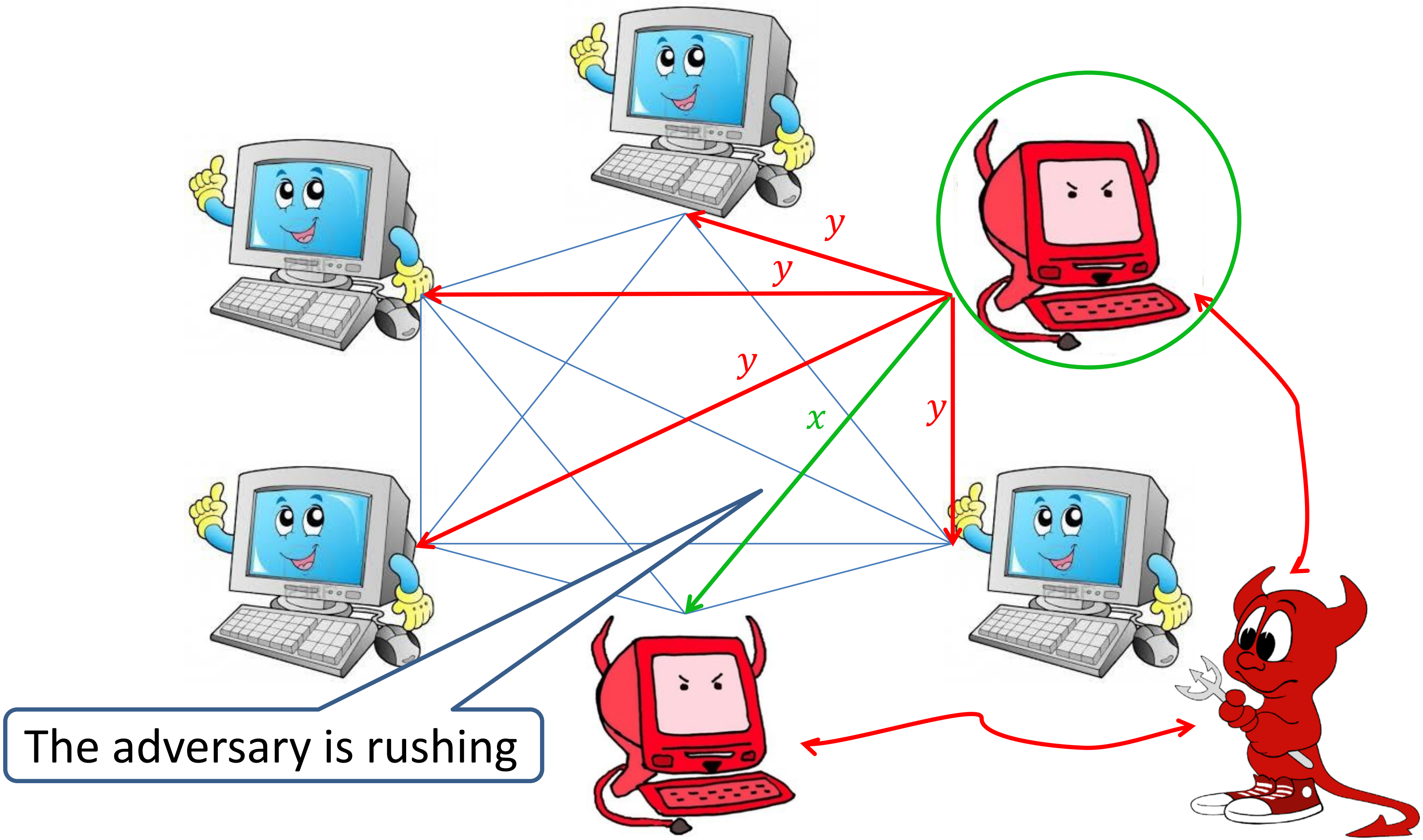
Attack on [BE'03]

Round 1: each party P_i distributes its input x_i



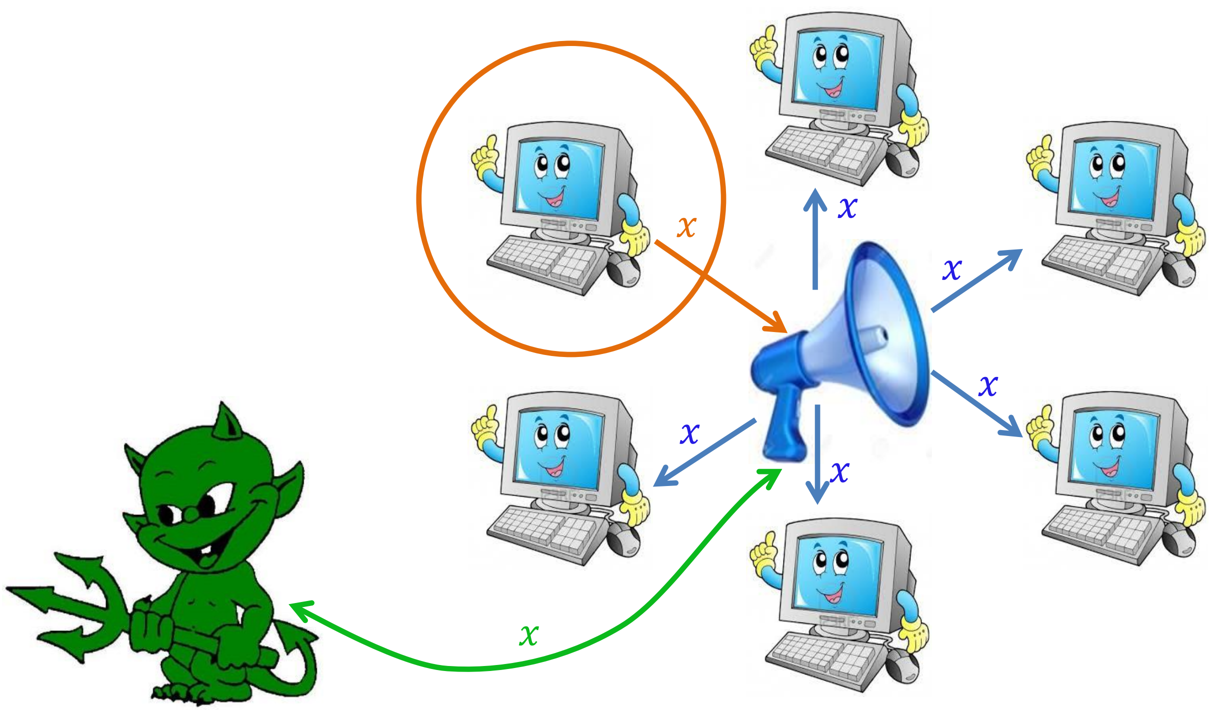
Attack on [BE'03]

Round 1: each party P_i distributes its input x_i



Attack on [BE'03]

- The adversary can corrupt an honest party and change its input x <u>after</u> the protocol started
- This behavior cannot be simulated in the ideal world (as in [HZ'10])



Unfair Broadcast

The ideal adversary is allowed to corrupt the sender and change its input – before any party received it

Def: Unfair broadcast for sender P_i is CSF with

- $f(x_1, \ldots, x_n, a) = (x_i, \ldots, x_i)$
- $l(x_1, \ldots, x_n) = x_i$

The difference from broadcast is the leakage function

Thm: Protocol [BE'03] implements $\mathcal{W}_{flex}^{D}(\mathcal{F}_{U-PBC})$ in the $(\mathcal{F}_{PSMT}, \mathcal{F}_{LE}, \mathcal{F}_{BA}, \mathcal{F}_{Trunc-BA})$-hybrid model, for $t < n/3$, assuming all parties start at the same round

Unfair Parallel Bcast $\Rightarrow$ Parallel Bcast

Before P_i distributes x_i, it commits to its input

1) Each party secret shares its input using $(t+1)$-out-of-n secret sharing
2) Each party broadcasts all the shares it received using an unfair parallel broadcast channel
3) Reconstruct and output the values

Intuition: In round 1 $\mathcal{A}$ only learns random shares

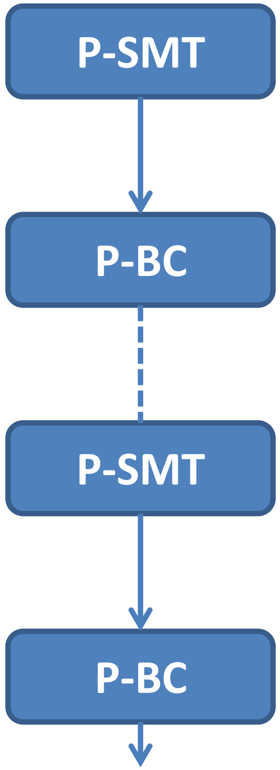
In round 2 $\mathcal{A}$ can change only $t < n/3$ shares

$\Rightarrow$ Inputs of parties that are honest in round 1 (before $\mathcal{A}$ learns anything) are reconstructed properly

Thm: $\mathcal{W}^{D}_{flex}(\mathcal{F}_{PBC})$ can be implemented in the $(\mathcal{F}_{PSMT}, \mathcal{F}_{U-PBC})$-hybrid model, for $t < n/3$, assuming all parties start at the same round

SFE with Expected $O(d)$ Rounds

Thm: Protocol [BGW'88] implements $\mathcal{W}^{D}_{flex}(\mathcal{F}_{SFE})$ in the $(\mathcal{F}_{PSMT}, \mathcal{F}_{PBC})$-hybrid model in $O(d)$ rounds,
assuming all parties start at same round



Thm: Let $c \geq 0$.
$\mathcal{W}_{PT}(\mathcal{F}_{SFE})$ can be implemented in the $\left(\mathcal{W}_{DT}(\mathcal{F}_{PSMT}), \mathcal{W}_{PT}(\mathcal{F}_{PBC})\right)$-hybrid model in expected $O(d)$ rounds,
assuming all parties start within $c+1$ rounds

Summary

We consider composability of cryptographic protocols with probabilistic termination

- Framework for designing cryptographic protocols in stand-alone fashion and compiler to fast composition in the UC framework
- Perfect, adaptively secure protocols in the P2P model
  1) BA with expected $O(1)$ rounds
  2) Parallel broadcast with expected $O(1)$ rounds
  3) SFE with expected $O(d)$ rounds

Thank You