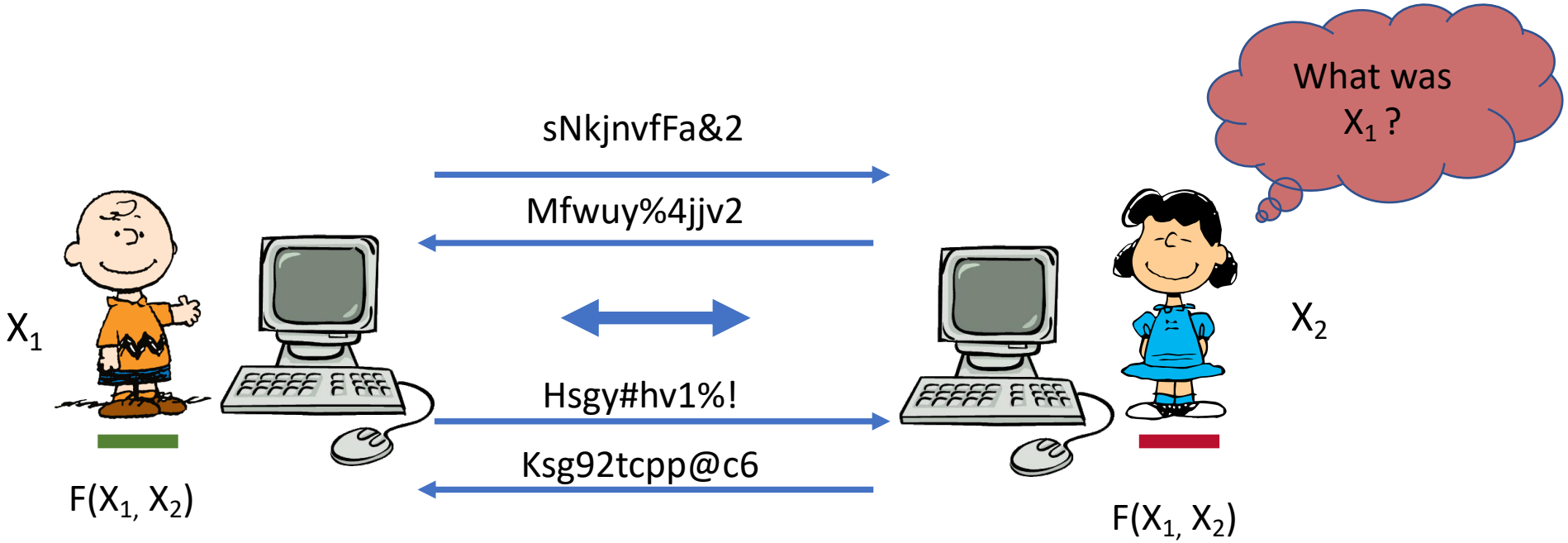
Reverse Firewalls for Adaptively Secure MPC without Setup

Suvradip Chakraborty, Chaya Ganesh, ***Mahak Pancholi****, Pratik Sarkar*

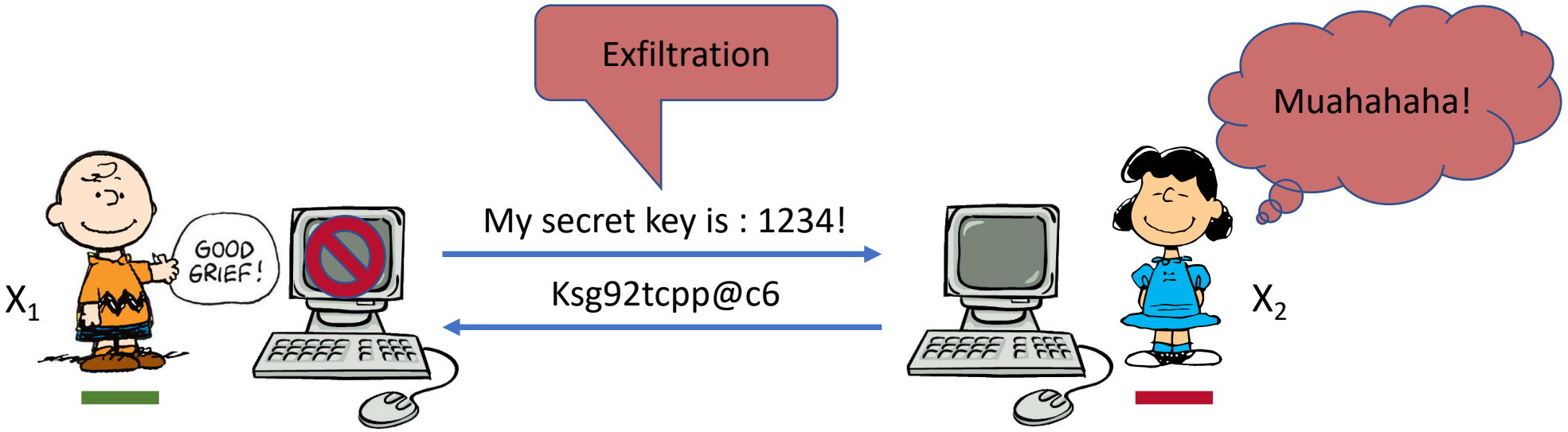
IST Austria, IISC Bangalore, **Aarhus University**, Boston University

MPC in Classical Setting



Caveat:
Only if Charlie's machine **behaves** honestly!

MPC in Classical Setting

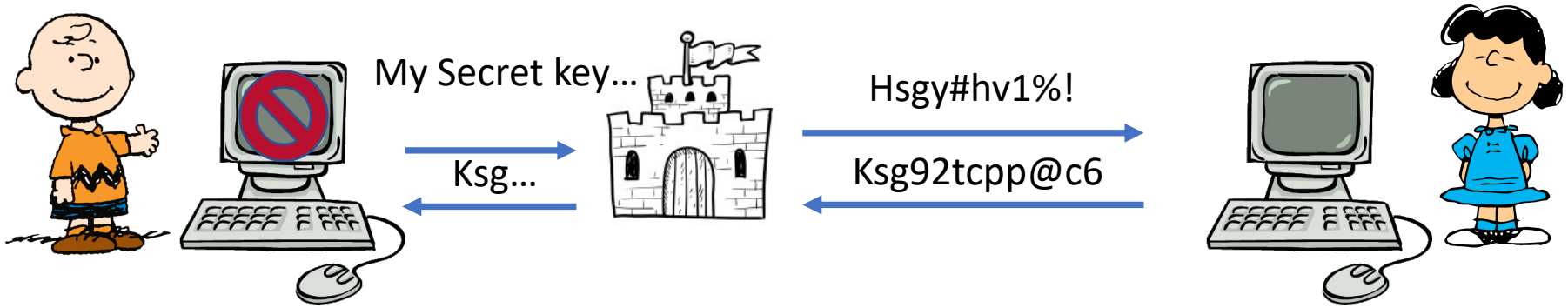


Can we design an MPC protocol such that we obtain some meaningful notion of security even when machines of the honest parties are tampered by the adversary?

- In general?
  - **No**
  - Tampering can simply send the honest party's input in clear instead of sending a valid first round message.
- But for a weaker class of tampering and more assumptions?
  - **Yes!**

Cryptographic Reverse Firewalls (RF)

(Mironov and Stephens-Davidowitz EC'15)



- **Intercepts all outgoing and incoming messages**
- **Has very simple operations**
- **Does not hold Charlie's secrets**

Properties of an RF

- Functionality Preserving: must preserve original protocol's functionality.
- Exfiltration Resistant (ER): prevents tampering from leaking any secrets.
- Security Preserving (SP): preserve the security properties of the underlying protocol for an honest party even when its implementation is tampered with.
- Transparent: Transcript messages look indistinguishable from the ones generated by honest implementation of the original protocol.

Some previous results

- Mironov and Stephens-Davidowitz (EC'15):
  - RF for 2PC for **passive** and **static** corruptions
- Chakraborty et al. (Crypto'20):
  - RF for general MPC with **active** and **static** corruptions
  - But **assuming CRS**
- This work:
  - RF for general MPC with **active** and **adaptive** corruptions
  - In **plain model**!

Active: Corrupted parties behave arbitrarily and may not follow the protocol.

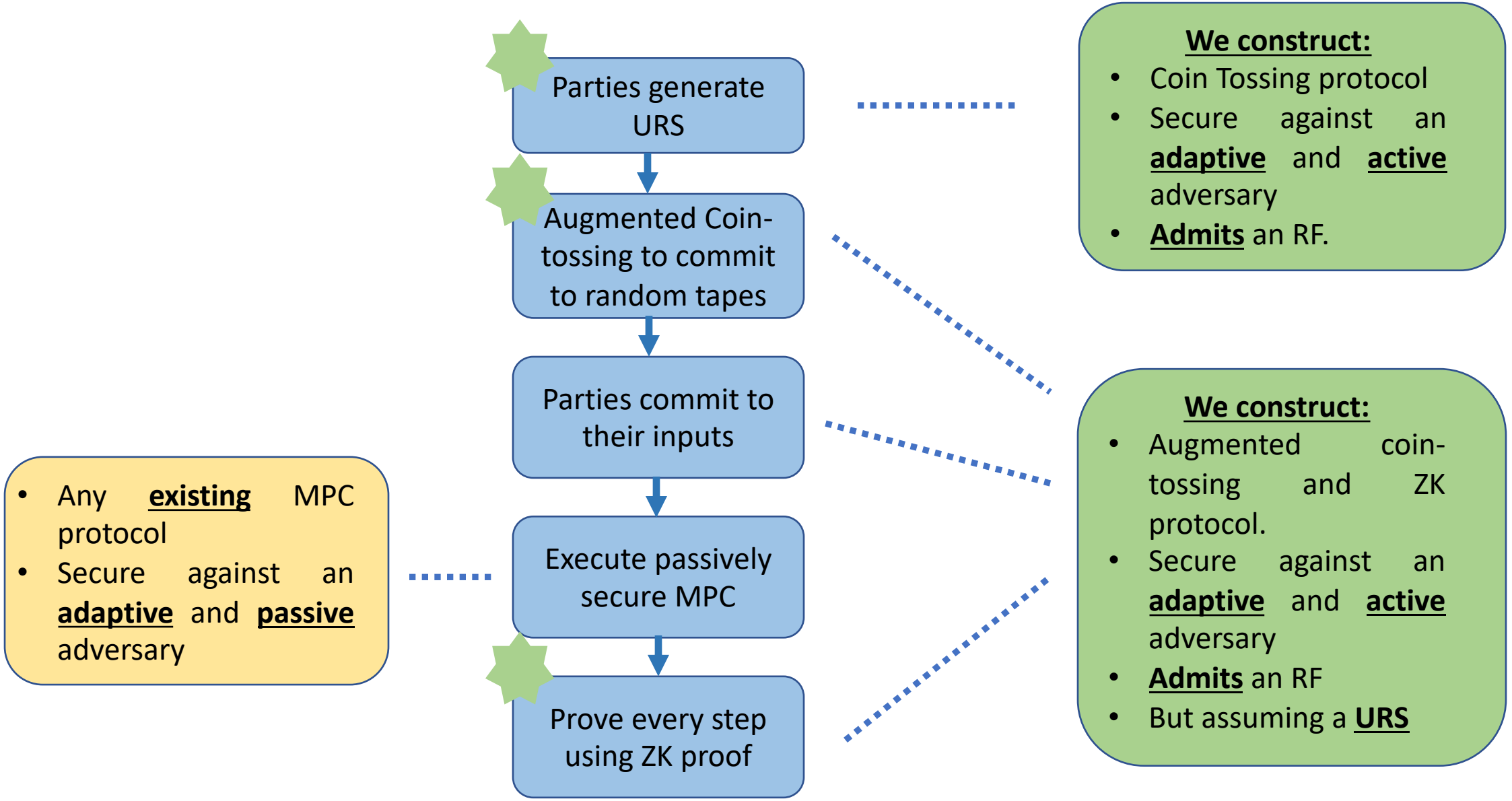
Adaptive: Adv can corrupt some honest party during protocol execution.

Our Results

- New definitions for the adaptive case.
- Implication: ER implies SP for protocols with simulation-based security.
- New construction secure against adaptive, active adversary, and secure even when honest parties' machines are tampered.

Approach and Challenges

- We follow the GMW compiler approach
  - Generate and commit to random tapes
  - Commit to inputs
  - Execute a passively secure protocol and at each step prove correctness by using ZK proofs.
- Problem: No security guaranteed if coins are not random
- Problem: Round messages might exfiltrate some secrets
- Solution: RF must ensure that random tapes are indeed random
- Solution: RF must randomize every outgoing and incoming message


Parties generate URS
Augmented Coin-tossing to commit to random tapes
Parties commit to their inputs
Execute passively secure MPC
Prove every step using ZK proof
We construct:
• Coin Tossing protocol
• Secure against an adaptive and active adversary
• Admits an RF.
We construct:
• Augmented coin-tossing and ZK protocol.
• Secure against an adaptive and active adversary
• Admits an RF
• But assuming a URS
• Any existing MPC protocol
• Secure against an adaptive and passive adversary

Augmented Coin-tossing (assuming URS)

A remark about previous attempt

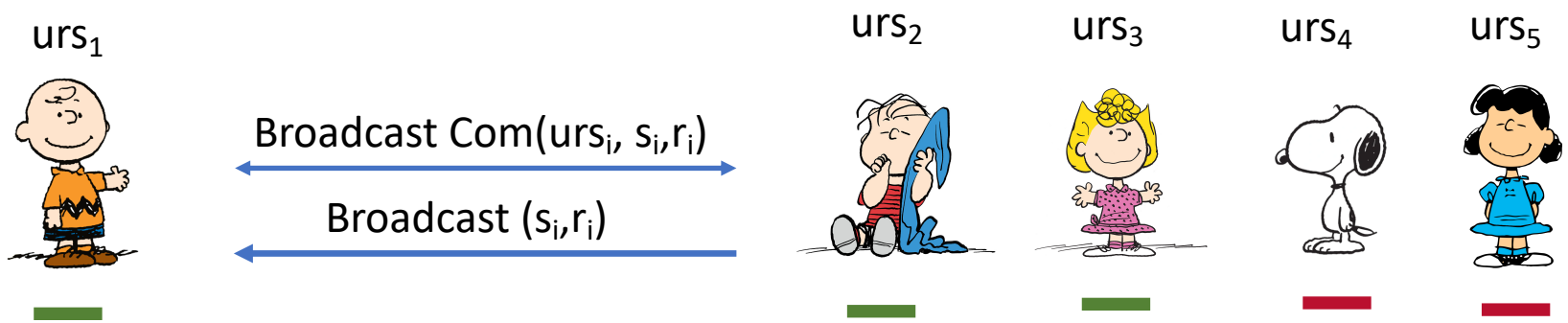
[CDN20]

- RFs with Augmented coin-tossing results in different views for each party.
- As a result, RF has to maul every proof during protocol execution.
- This requires controlled malleable nizks.
- No known adaptively secure construction.
- Crucial Observation: Make each party have a consistent view!

Augmented Coin-tossing (assuming URS)

Commitment Scheme that:

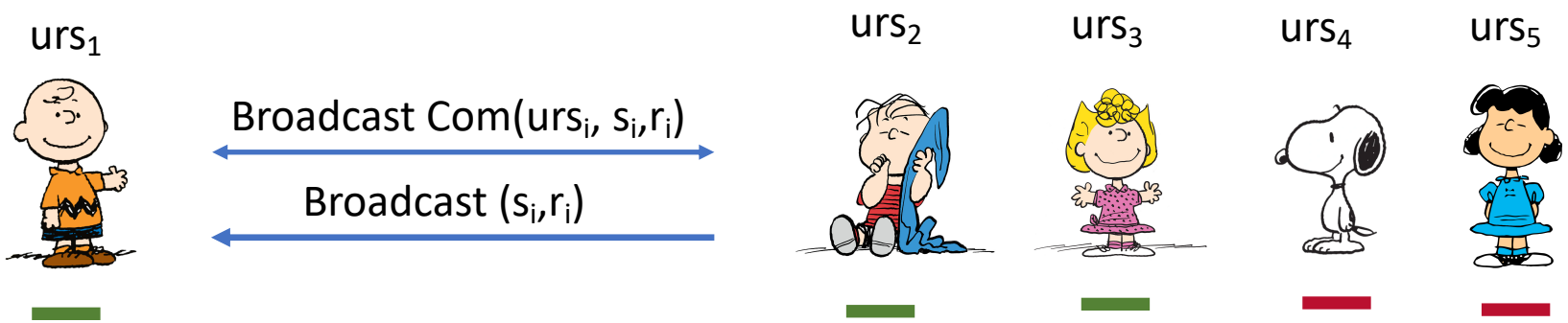
- Has adaptive and active security
- Is additively homomorphic under urs



Output: random string s

Output: com to random string s

Augmented Coin-tossing (assuming URS)

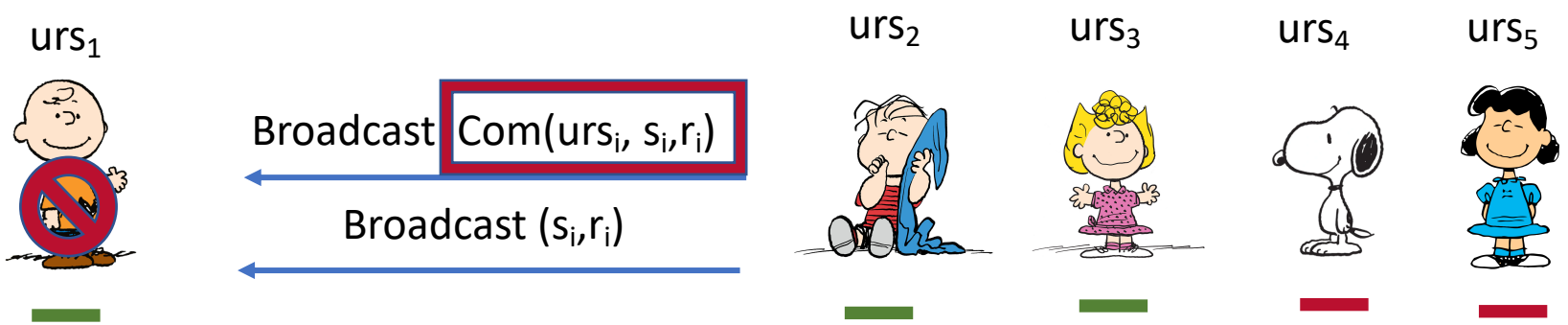


Local Computation:

1. Check if the opening is correct
2. Compute: $c_i = Com(urs_1, s_i, r_i)$
3. $C = \sum c_i$

Augmented Coin-tossing (assuming URS)

1. s_i is not random; the final coin is not random.
2. r_i is not random; com exfiltrates



RF: Use homomorphism

$Com(urs,a_1,r_1) + Com(urs,a_2,r_2) = Com(urs, a_1+a_2, r_1+r_2)$

Augmented Coin-tossing (assuming URS)

Theorem: Assuming Com is an adaptively secure homomorphic commitment in the URS model, there exists a protocol that securely implements the augmented coin-tossing functionality against adaptive corruption of parties in the URS model.

Zero-Knowledge Protocol (assuming URS)

[CSW20]

Additively homomorphic

Oblivious ciphertext sampling

G, w, urs = (urs_{com}, pk)

G, urs = (urs_{com}, pk)

com(H) = com(1/0,r), enc_0(pk,r,r'), enc_1(r,r')

Challenge e

Response

Response:
If e = 0: Open com to H
If e = 1:
Open non edges in π(H)
and send π.

RF:
Randomly permute H, and of the first-round messages accordingly. Randomize com and enc.
Adjust **Response** according to round 1.

Zero-Knowledge Protocol (assuming URS)

Canetti et al [CSW20]

G, w, urs = (urs_{com}, pk)

G, urs = (urs_{com}, pk)

H, com(1/0,r), enc(r,r'), **enc(e_p,r_p)**

Challenge **e_v**

Response

Response:

Set e = $e_p \oplus e_v$

If e = 0: Open H

If e = 1:

Open non edges in π(H) and send π.

RF:

Additionally randomize ep and enc(ep,rp)

Zero-Knowledge Protocol(assuming URS)

Theorem: If Com is a non-interactive equivocal commitment scheme in the urs model and PKE is an IND-CPA public key encryption scheme with oblivious ciphertext sampleability, then there exists a protocol that realizes ZK functionality for all NP relations against adaptive corruptions in the urs model.

Coin-tossing in the plain model

Construction

Proof Idea

Parameter generation phase

Generate pairwise public key (pk) (with oblivious ciphertext and key sampleability) and Pedersen commitment parameters (g_i,h_i)

Extract commitment trapdoor

Set pk such that Sim knows the sk

Commitment generation phase

Commit to a coin: $Com(g_i,h_i,s_i,r_i)$ and encrypt: $Enc(pk,r_i)$

Commit to random coins

Extract by decrypting $Enc(pk,r_i)$

Commitment opening phase

Open Commitments

Equivocate using the trapdoor

Output Phase

Output Random Coin

Coin-tossing in the plain model

Theorem: Assuming Discrete Log and Knowledge Assumption and a Public key encryption scheme with oblivious ciphertext sampling, oblivious public key sampling, satisfying additive homomorphism there exists a protocol that securely implements coin-tossing functionality against adaptive corruptions in the plain model.

In Conclusion

- In this talk:
  - Cryptographic reverse firewall model
  - Construction secure against adaptive, active adversary, and secure even when honest parties' machines are tampered.
- Additional Results:
  - New definitions for the adaptive case
  - Implications between above definitions

Thank you!

eprint.iacr.org/2021/1262