Ph.D. Thesis Defense

On Large-Scale Multiparty Computation with sub-linear Communication using Ephemeral Servers

Anders Konring, IT University of Copenhagen

November 19, 2023

This Presentation

Encryption to the Future [CDK+22]

A Paradigm for Sending Secret Messages to Future (Anonymous) Committees

Matteo Campanelli, Bernardo David, Hamidreza Khoshakhlagh, Anders Konring, Jesper Buus Nielsen

YOLO YOSO [CDGK22]

Fast and Simple Encryption and Secret Sharing in the YOSO Model

Ignacio Cascudo, Bernardo David, Lydia Garms, Anders Konring

Layered MPC [DDG+23]

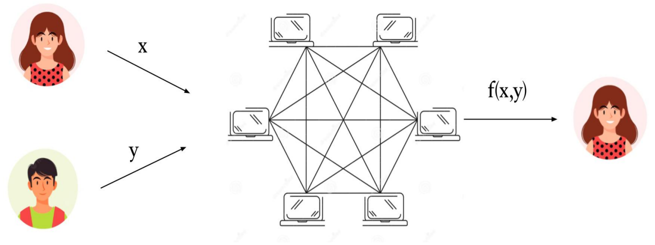
Perfect MPC over Layered Graphs

Bernardo David, Yuval Ishai, Anders Konring, Eyal Kushilevitz, Varun Narayanan
(Aarushi Goel, Chen-Da Liu-Zhang, Giovanni Deligios)

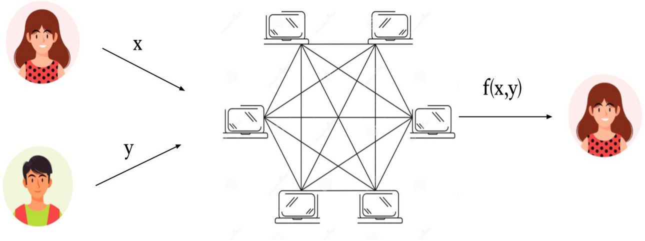
IT UNIVERSITY OF COPENHAGEN

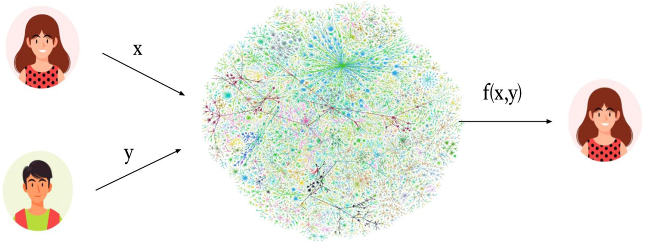
TECHNION
Israel Institute
of Technology

Multiparty Computation (MPC)



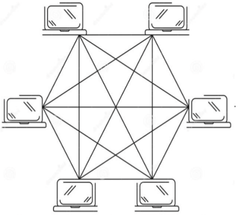
Multiparty Computation (MPC)



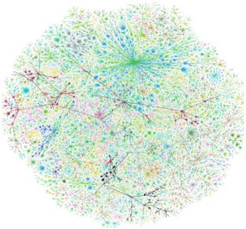


MPC Environments

MPC (classic environment):



MPC (dynamic environment):

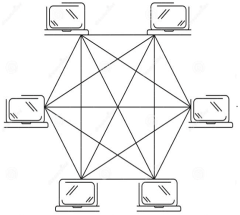


MPC Environments

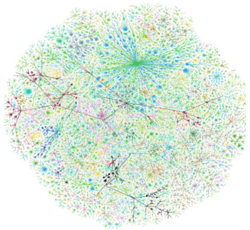
MPC (classic environment):

Computing parties are:

- small in number.
- well-connected.
- known (to each other) in advance.
- guaranteed to be online.



MPC (dynamic environment):

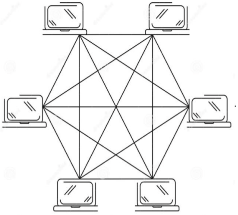


MPC Environments

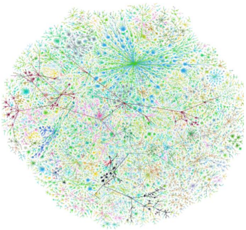
MPC (classic environment):

Computing parties are:

- small in number.
- well-connected.
- known (to each other) in advance.
- guaranteed to be online.
- easy target for an adaptive adversary.
- not resilient to machine failures.



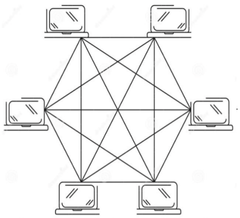
MPC (dynamic environment):



MPC (classic environment):

Computing parties are:

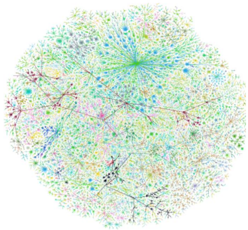
- small in number.
- well-connected.
- known (to each other) in advance.
- guaranteed to be online.
- easy target for an adaptive adversary.
- not resilient to machine failures.



MPC (dynamic environment):

Computing parties are:

- part of a large-scale (P2P) network.
- not necessarily connected or known.
- joining and leaving the network at any time.



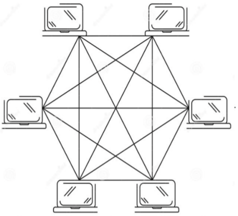
MPC Environments

MPC (classic environment):

Computing parties are:

- small in number.
- well-connected.
- known (to each other) in advance.
- guaranteed to be online.

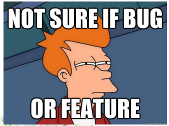
- easy target for an adaptive adversary.
- not resilient to machine failures.



MPC (dynamic environment):

Computing parties are:

- part of a large-scale (P2P) network.
- not necessarily connected or known.
- joining and leaving the network at any time.



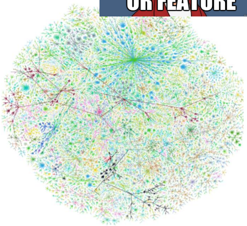


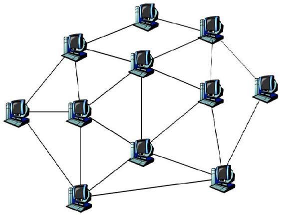
Table of Contents

Large-Scale MPC on Blockchains

Permissionless Blockchains

Blockchains are large public P2P networks.

- Incentivized coordination platform for miners/stakeholders.
- Secure if more than half of compute/stake is owned by honest parties.
- Even sophisticated smart-contract platforms do not provide privacy of input.

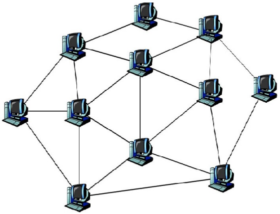


Permissionless Blockchains

Blockchains are large public P2P networks.

- Incentivized coordination platform for miners/stakeholders.
- Secure if more than half of compute/stake is owned by honest parties.
- Even sophisticated smart-contract platforms do not provide privacy of input.

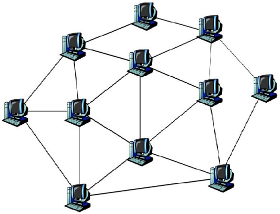
Built-in consensus layer:



Permissionless Blockchains

Blockchains are large public P2P networks.

- Incentivized coordination platform for miners/stakeholders.
- Secure if more than half of compute/stake is owned by honest parties.
- Even sophisticated smart-contract platforms do not provide privacy of input.



Built-in consensus layer:

- implements a PKI

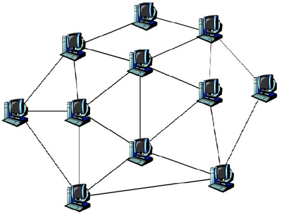
Permissionless Blockchains

Blockchains are large public P2P networks.

- Incentivized coordination platform for miners/stakeholders.
- Secure if more than half of compute/stake is owned by honest parties.
- Even sophisticated smart-contract platforms do not provide privacy of input.

Built-in consensus layer:

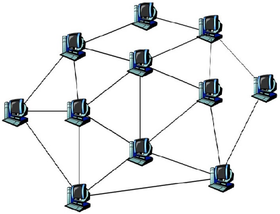
- implements a PKI
- implements total-ordered broadcast



Permissionless Blockchains

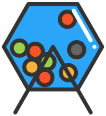
Blockchains are large public P2P networks.

- Incentivized coordination platform for miners/stakeholders.
- Secure if more than half of compute/stake is owned by honest parties.
- Even sophisticated smart-contract platforms do not provide privacy of input.



Built-in consensus layer:

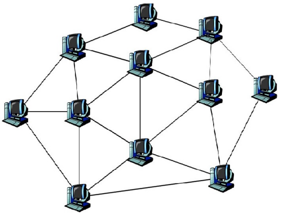
- implements a PKI
- implements total-ordered broadcast
- implements a ”lottery” mechanism (e.g. VRF-based)



Permissionless Blockchains

Blockchains are large public P2P networks.

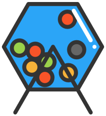
- Incentivized coordination platform for miners/stakeholders.
- Secure if more than half of compute/stake is owned by honest parties.
- Even sophisticated smart-contract platforms do not provide privacy of input.



Built-in consensus layer:

- implements a PKI
- implements total-ordered broadcast
- implements a ”lottery” mechanism (e.g. VRF-based)

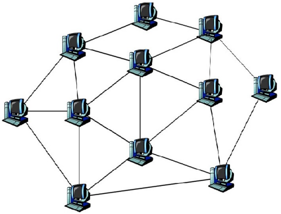
Can we use the blockchain infrastructure as a private computing platform?



Permissionless Blockchains

Blockchains are large public P2P networks.

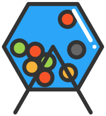
- Incentivized coordination platform for miners/stakeholders.
- Secure if more than half of compute/stake is owned by honest parties.
- Even sophisticated smart-contract platforms do not provide privacy of input.



Built-in consensus layer:

- implements a PKI
- implements total-ordered broadcast
- implements a ”lottery” mechanism (e.g. VRF-based)

Can we use the blockchain infrastructure as a private computing platform?

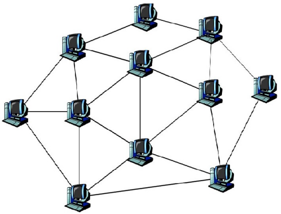


- Blockchain signatures (oracles, bridges, interop).
- Secret auctions and elections.
- Private smart contracts (selective decryption/re-encryption).

Permissionless Blockchains

Blockchains are large public P2P networks.

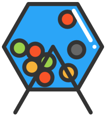
- Incentivized coordination platform for miners/stakeholders.
- Secure if more than half of compute/stake is owned by honest parties.
- Even sophisticated smart-contract platforms do not provide privacy of input.



Built-in consensus layer:

- implements a PKI
- implements total-ordered broadcast
- implements a ”lottery” mechanism (e.g. VRF-based)

Can we use the blockchain infrastructure as a private computing platform?



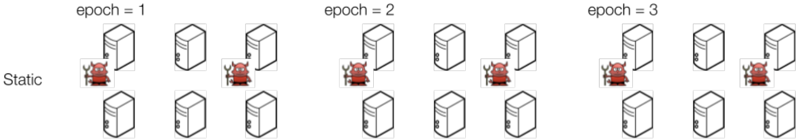
- Blockchain signatures (oracles, bridges, interop).
- Secret auctions and elections.
- Private smart contracts (selective decryption/re-encryption).

YES! [BGG+20, GHK+21, CGG+21]

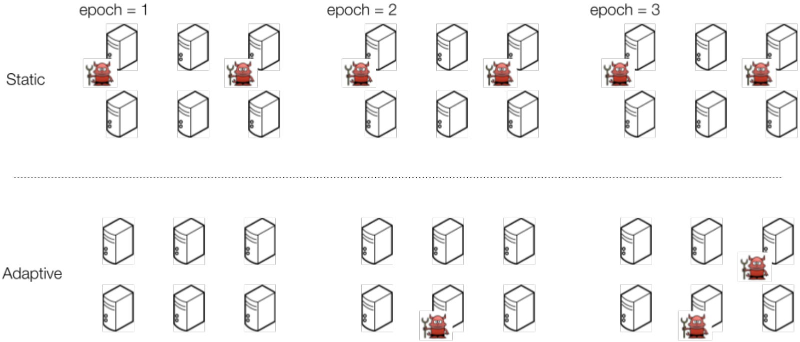
YOSO MPC [GHK+21]

Mobile Adversary

Mobile Adversary



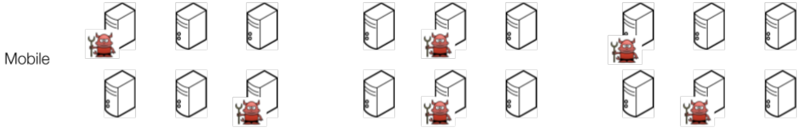
Mobile Adversary



Mobile Adversary

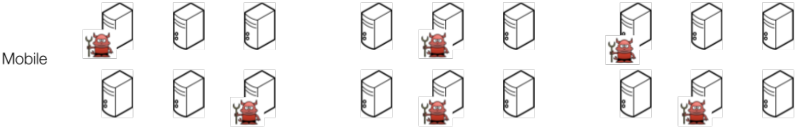
Mobile Adversary [OY91]

- The mobile adversary can move to a new set of parties between *epochs*.



Mobile Adversary [OY91]

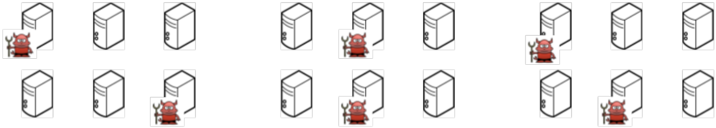
- The mobile adversary can move to a new set of parties between *epochs*.
- Assume that parties can do *secure erasures*.



Mobile Adversary [OY91]

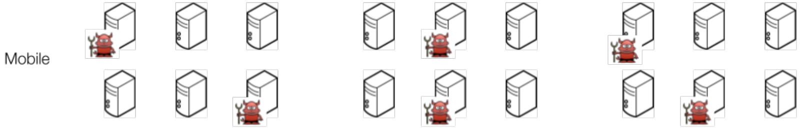
- The mobile adversary can move to a new set of parties between *epochs*.
- Assume that parties can do *secure erasures*.
- Make sure that the ”state” is re-randomized between epochs.

Mobile



Mobile Adversary [OY91]

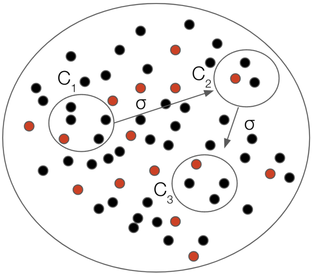
- The mobile adversary can move to a new set of parties between *epochs*.
- Assume that parties can do *secure erasures*.
- Make sure that the ”state” is re-randomized between epochs.
- Decades of research in Proactive Secret Sharing and MPC.
- But existing work either settles for
  - multi-round epochs, |epochs| > 1 [HJKY95, ADN06, BELO15, ELL20]
  - corruption threshold n/c [OY91]
  - generally incompatible with large networks (thousands/millions) of nodes.



YOSO MPC [GHK+21]

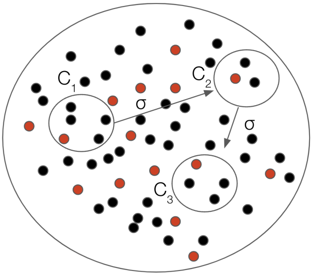
Key Ideas:

1. Use the scale of the network to provide resilience towards a powerful mobile adversary.



Key Ideas:

1. Use the scale of the network to provide resilience towards a powerful mobile adversary.
2. Randomly sample small committees C_i of size $n \ll N$ that do the computation ”on behalf of” of the larger network.



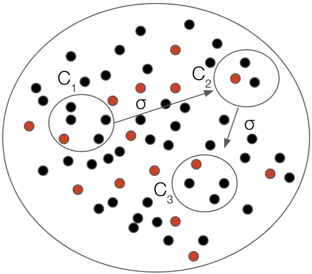
YOSO MPC [GHK+21]

Key Ideas:

1. Use the scale of the network to provide resilience towards a powerful mobile adversary.
2. Randomly sample small committees C_i of size $n \ll N$ that do the computation "on behalf of" of the larger network.

Committee members are

- anonymous until they speak.



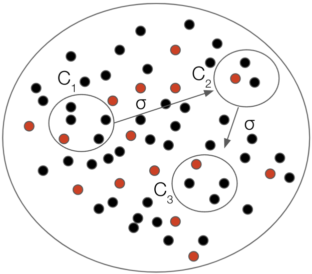
YOSO MPC [GHK+21]

Key Ideas:

1. Use the scale of the network to provide resilience towards a powerful mobile adversary.
2. Randomly sample small committees C_i of size $n \ll N$ that do the computation ”on behalf of” of the larger network.

Committee members are

- anonymous until they speak.
- limited to ”Only Speak Once”.



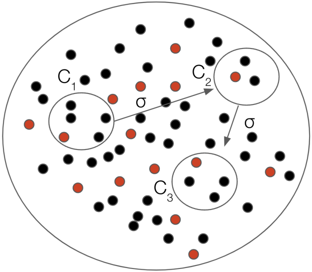
YOSO MPC [GHK+21]

Key Ideas:

1. Use the scale of the network to provide resilience towards a powerful mobile adversary.
2. Randomly sample small committees C_i of size $n \ll N$ that do the computation "on behalf of" of the larger network.

Committee members are

- anonymous until they speak.
- limited to "Only Speak Once".
- a moving target for the mobile adversary.



YOSO MPC [GHK+21]

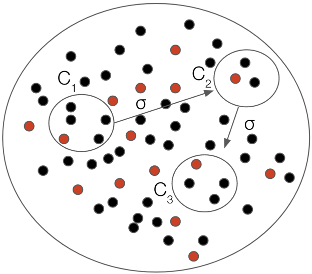
Key Ideas:

1. Use the scale of the network to provide resilience towards a powerful mobile adversary.
2. Randomly sample small committees C_i of size $n \ll N$ that do the computation ”on behalf of” of the larger network.

Committee members are

- anonymous until they speak.
- limited to ”Only Speak Once”.
- a moving target for the mobile adversary.

Attractive Side-effects



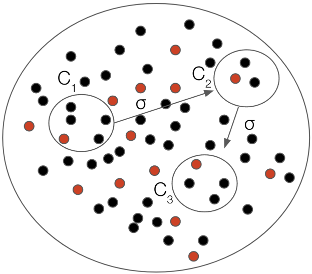
YOSO MPC [GHK+21]

Key Ideas:

1. Use the scale of the network to provide resilience towards a powerful mobile adversary.
2. Randomly sample small committees C_i of size $n \ll N$ that do the computation "on behalf of" of the larger network.

Committee members are

- anonymous until they speak.
- limited to "Only Speak Once".
- a moving target for the mobile adversary.



Attractive Side-effects

- Built-in support for node churn.

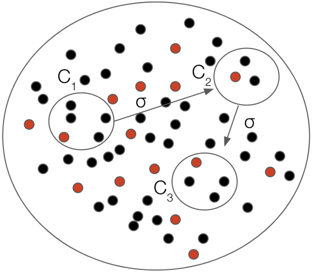
YOSO MPC [GHK+21]

Key Ideas:

1. Use the scale of the network to provide resilience towards a powerful mobile adversary.
2. Randomly sample small committees C_i of size $n \ll N$ that do the computation "on behalf of" of the larger network.

Committee members are

- anonymous until they speak.
- limited to "Only Speak Once".
- a moving target for the mobile adversary.



Attractive Side-effects

- Built-in support for node churn.
- Easy to sample committees with tolerated corruption threshold (whp.)

YOSO MPC [GHK+21]

Key Ideas:

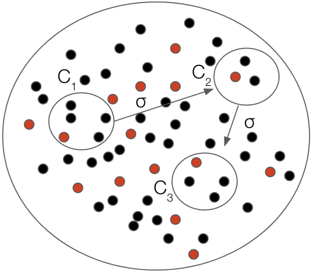
1. Use the scale of the network to provide resilience towards a powerful mobile adversary.
2. Randomly sample small committees C_i of size $n \ll N$ that do the computation ”on behalf of” of the larger network.

Committee members are

- anonymous until they speak.
- limited to ”Only Speak Once”.
- a moving target for the mobile adversary.

Attractive Side-effects

- Built-in support for node churn.
- Easy to sample committees with tolerated corruption threshold (whp.)
- Communication depends on n instead of N.



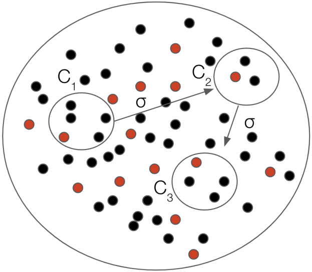
YOSO MPC [GHK+21]

Key Ideas:

1. Use the scale of the network to provide resilience towards a powerful mobile adversary.
2. Randomly sample small committees C_i of size $n \ll N$ that do the computation "on behalf of" of the larger network.

Committee members are

- anonymous until they speak.
- limited to "Only Speak Once".
- a moving target for the mobile adversary.



Attractive Side-effects

- Built-in support for node churn.
- Easy to sample committees with tolerated corruption threshold (whp.)
- Communication depends on n instead of N.

How do we design protocols where parties "speak only once"?

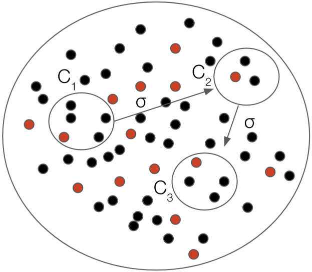
YOSO MPC [GHK+21]

Key Ideas:

1. Use the scale of the network to provide resilience towards a powerful mobile adversary.
2. Randomly sample small committees C_i of size $n \ll N$ that do the computation ”on behalf of” of the larger network.

Committee members are

- anonymous until they speak.
- limited to ”Only Speak Once”.
- a moving target for the mobile adversary.



Attractive Side-effects

- Built-in support for node churn.
- Easy to sample committees with tolerated corruption threshold (whp.)
- Communication depends on n instead of N.

How do we design protocols where parties ”speak only once”?

How do we send a message to an anonymous party?

YOSO MPC [GHK+21]

Role Execution (RX)

- Divide the protocol into small ”units” called roles executed by a single round on single machine (player-replaceable).

Role Assignment (RA)

YOSO MPC [GHK+21]

Role Execution (RX)

- Divide the protocol into small ”units” called roles executed by a single round on single machine (player-replaceable).
- Each committee member (role) executes the protocol step according to the specification.

Role Assignment (RA)

YOSO MPC [GHK+21]

Role Execution (RX)

- Divide the protocol into small "units" called roles executed by a single round on single machine (player-replaceable).
- Each committee member (role) executes the protocol step according to the specification.
- Performs secure erasure.

Role Assignment (RA)

YOSO MPC [GHK+21]

Role Execution (RX)

- Divide the protocol into small ”units” called roles executed by a single round on single machine (player-replaceable).
- Each committee member (role) executes the protocol step according to the specification.
- Performs secure erasure.
- Send messages to future roles using RA (Only Speak Once).

Role Assignment (RA)

YOSO MPC [GHK+21]

Role Execution (RX)

- Divide the protocol into small ”units” called roles executed by a single round on single machine (player-replaceable).
- Each committee member (role) executes the protocol step according to the specification.
- Performs secure erasure.
- Send messages to future roles using RA (Only Speak Once).
- Existing solutions include:
  - YOSO IT (RB) [GHK+21]
  - YOSO Comp. (CDN) [GHK+21]
  - Fluid MPC [CGG+21]

Role Assignment (RA)

YOSO MPC [GHK+21]

Role Execution (RX)

- Divide the protocol into small ”units” called roles executed by a single round on single machine (player-replaceable).
- Each committee member (role) executes the protocol step according to the specification.
- Performs secure erasure.
- Send messages to future roles using RA (Only Speak Once).
- Existing solutions include:
  - YOSO IT (RB) [GHK+21]
  - YOSO Comp. (CDN) [GHK+21]
  - Fluid MPC [CGG+21]

Role Assignment (RA)

- (Randomly) associates a machine in the network with a role in the protocol.

YOSO MPC [GHK+21]

Role Execution (RX)

- Divide the protocol into small "units" called roles executed by a single round on single machine (player-replaceable).
- Each committee member (role) executes the protocol step according to the specification.
- Performs secure erasure.
- Send messages to future roles using RA (Only Speak Once).
- Existing solutions include:
    - YOSO IT (RB) [GHK+21]
    - YOSO Comp. (CDN) [GHK+21]
    - Fluid MPC [CGG+21]

Role Assignment (RA)

- (Randomly) associates a machine in the network with a role in the protocol.
- Establishes a receiver-anonymous channel to the machine.

YOSO MPC [GHK+21]

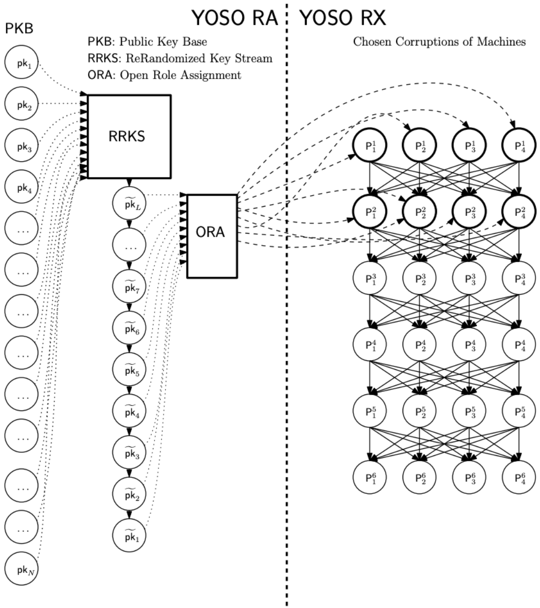
Role Execution (RX)

- Divide the protocol into small ”units” called roles executed by a single round on single machine (player-replaceable).
- Each committee member (role) executes the protocol step according to the specification.
- Performs secure erasure.
- Send messages to future roles using RA (Only Speak Once).
- Existing solutions include:
  - YOSO IT (RB) [GHK+21]
  - YOSO Comp. (CDN) [GHK+21]
  - Fluid MPC [CGG+21]

Role Assignment (RA)

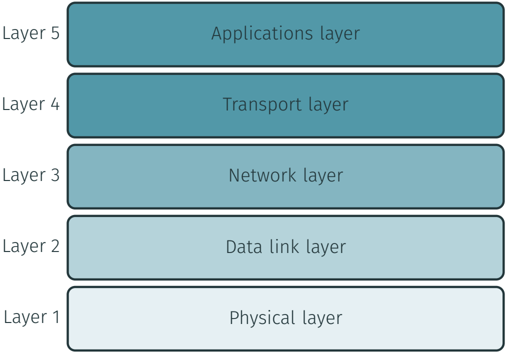
- (Randomly) associates a machine in the network with a role in the protocol.
- Establishes a receiver-anonymous channel to the machine.
- Existing solutions include:
  - YOSO Compiler [GHK+21]
  - Random-Index PIR [GHM+21]
  - General WE [GGSW13]

YOSO MPC [GHK+21]



Overview

Overview



OSI Model

Overview

Layer 1 PoS Blockchain (PKI + BC + lottery)

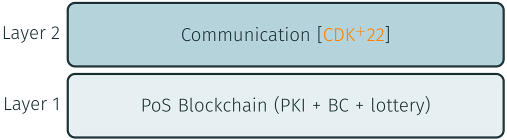
Ephemeral Committees Model

Layer 1:
Public PoS blockchain such as Ouroboros Praos [DGKR18].

Overview

Layer 2 (Encryption to the Future):
Communication towards unknown lottery winners.

Layer 1:
Public PoS blockchain such as Ouroboros Praos [DGKR18].



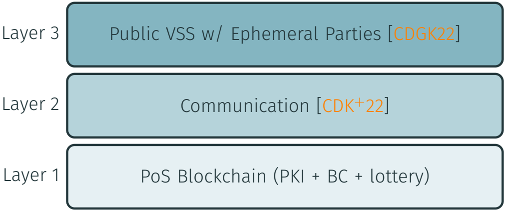
Ephemeral Committees Model

Overview

Layer 3 (YOLO-YOSO):
PVSS and resharing - basic building block for MPC and other applications.

Layer 2 (Encryption to the Future):
Communication towards unknown lottery winners.

Layer 1:
Public PoS blockchain such as Ouroboros Praos [DGKR18].



Ephemeral Committees Model

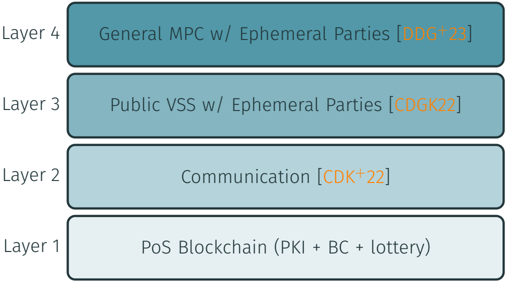
Overview

Layer 4 (Layered MPC):
Perfect General MPC over a layered graph using only ephemeral servers.

Layer 3 (YOLO-YOSO):
PVSS and resharing - basic building block for MPC and other applications.

Layer 2 (Encryption to the Future):
Communication towards unknown lottery winners.

Layer 1:
Public PoS blockchain such as Ouroboros Praos [DGKR18].



Ephemeral Committees Model

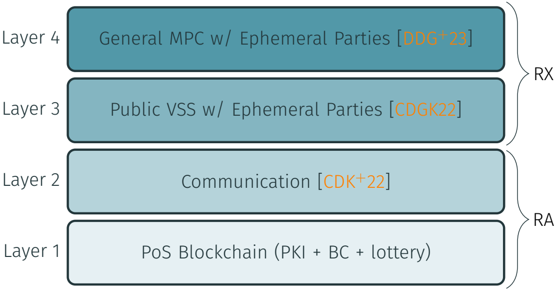
Overview

Layer 4 (Layered MPC):
Perfect General MPC over a layered graph using only ephemeral servers.

Layer 3 (YOLO-YOSO):
PVSS and resharing - basic building block for MPC and other applications.

Layer 2 (Encryption to the Future):
Communication towards unknown lottery winners.

Layer 1:
Public PoS blockchain such as Ouroboros Praos [DGKR18].



Ephemeral Committees Model

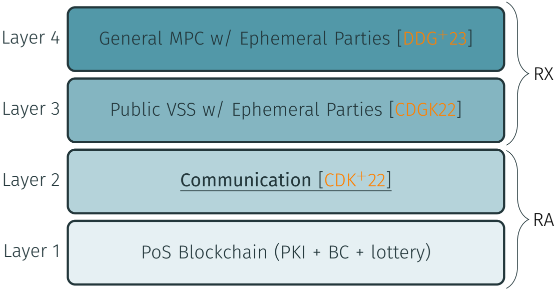
Overview

Layer 4 (Layered MPC):
Perfect General MPC over a layered graph using only ephemeral servers.

Layer 3 (YOLO-YOSO):
PVSS and resharing - basic building block for MPC and other applications.

Layer 2 (Encryption to the Future):
Communication towards unknown lottery winners.

Layer 1:
Public PoS blockchain such as Ouroboros Praos [DGKR18].



Ephemeral Committees Model

Overview: Encryption to the Future [CDK+22]

Existing work on (RA):

Overview: Encryption to the Future [CDK+22]

Existing work on (RA):

- YOSO RA (toy example) [GHK+21]
  A blueprint for implementing RA but too concrete to generalize the problem of transferring secret state to future committees.

Overview: Encryption to the Future [CDK+22]

Existing work on (RA):

- YOSO RA (toy example) [GHK+21]
  A blueprint for implementing RA but too concrete to generalize the problem of transferring secret state to future committees.
- Random-Index PIR [GHM+21]
  Implementation of RA using the YOSO RA blueprint but relies on strong assumptions such as Mixnets or FHE.

Overview: Encryption to the Future [CDK+22]

Existing work on (RA):

- YOSO RA (toy example) [GHK+21]
  A blueprint for implementing RA but too concrete to generalize the problem of transferring secret state to future committees.
- Random-Index PIR [GHM+21]
  Implementation of RA using the YOSO RA blueprint but relies on strong assumptions such as Mixnets or FHE.
- General Witness Encryption [GGSW13]
  Sufficient for implementing any RA mechanism but constructions rely on existence of efficient multi-linear maps or iO

Overview: Encryption to the Future [CDK+22]

Existing work on (RA):

- YOSO RA (toy example) [GHK+21]
  A blueprint for implementing RA but too concrete to generalize the problem of transferring secret state to future committees.
- Random-Index PIR [GHM+21]
  Implementation of RA using the YOSO RA blueprint but relies on strong assumptions such as Mixnets or FHE.
- General Witness Encryption [GGSW13]
  Sufficient for implementing any RA mechanism but constructions rely on existence of efficient multi-linear maps or iO

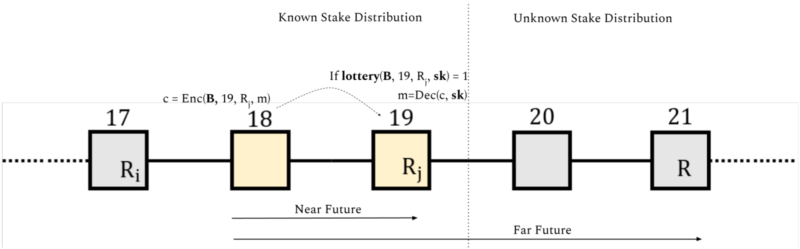
Motivation: Transferring secret state to future committees

- Consider secret state to the ”near” vs. ”far” future.
- Investigate the need for auxiliary committees for carrying state into the future.
- Consider the need for authenticated channels (Authentication-from-the-Past).
- Possibility of realizing RA using ”standard” assumptions.

Overview: Encryption to the Future [CDK+22]

Encryption to the near Future.

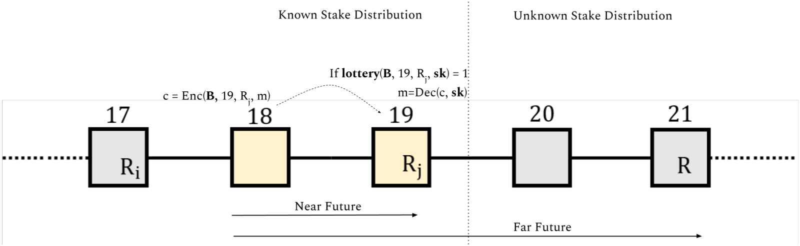
1. Instantiate YOSO using EtF with an anonymous lottery.



Overview: Encryption to the Future [CDK+22]

Encryption to the near Future.

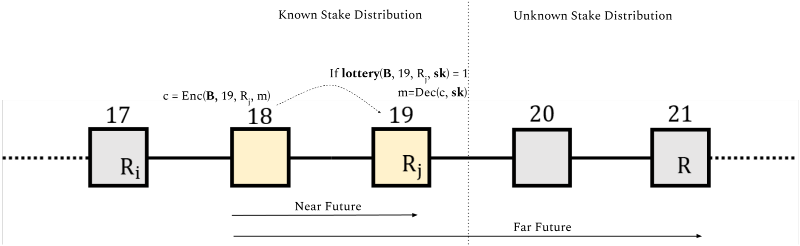
1. Instantiate YOSO using EtF with an anonymous lottery.
2. Introduce a relaxed version of WE called "WE over Commitments" (cWE).



Overview: Encryption to the Future [CDK+22]

Encryption to the near Future.

1. Instantiate YOSO using EtF with an anonymous lottery.
2. Introduce a relaxed version of WE called "WE over Commitments" (cWE).
3. Construction using cWE based on standard assumptions (OT + GC).



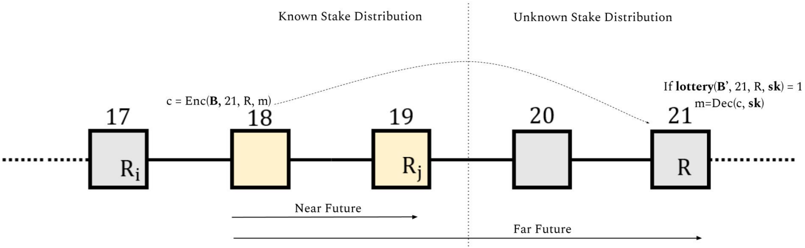
Overview: Encryption to the Future [CDK+22]

Encryption to the near Future.

1. Instantiate YOSO using EtF with an anonymous lottery.
2. Introduce a relaxed version of WE called "WE over Commitments" (cWE).
3. Construction using cWE based on standard assumptions (OT + GC).

Encryption to the far Future.

1. No auxiliary committees $\iff$ BWE (Blockchain Witness Encryption).



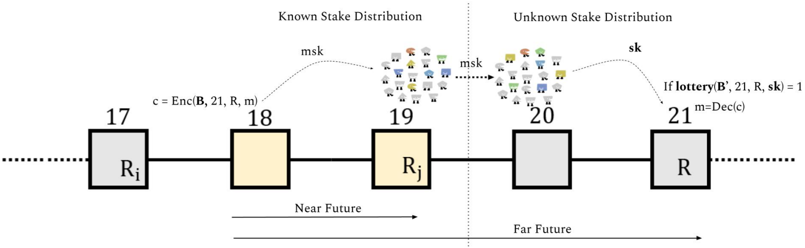
Overview: Encryption to the Future [CDK+22]

Encryption to the near Future.

1. Instantiate YOSO using EtF with an anonymous lottery.
2. Introduce a relaxed version of WE called "WE over Commitments" (cWE).
3. Construction using cWE based on standard assumptions (OT + GC).

Encryption to the far Future.

1. No auxiliary committees $\iff$ BWE (Blockchain Witness Encryption).
2. Construction using EtF (near) + TIBE. With minimal use of auxiliary committees (indep. of size/number of messages)



Overview: Encryption to the Future [CDK+22]

Type	Scheme	Communication	Committee?	Interaction?
EtF (near)	CaBKaS [BGG+20]	$O(1)$	yes	yes
	RPIR [GHK+21]	$O(1)$	yes	yes
	cWE(GC+OT) (Sec. 4.2)	**O(N)**	**no**	**no***
EtF (far)	IBE (Sec. 7)	$O(1)$	yes	yes
	WEB [GKM+20]	$O(M)$	yes	yes
	Full-fledged WE	$O(1)$	no	no

- "Committee?" indicates whether a committee is required.
- "Communication" refers to the communication complexity in the number of all parties N, or the number of plaintexts (called deposited secrets in [GKM+20]) M of a given fixed length.
- Asterisk* means non-interactive solutions that require sending a first reusable message

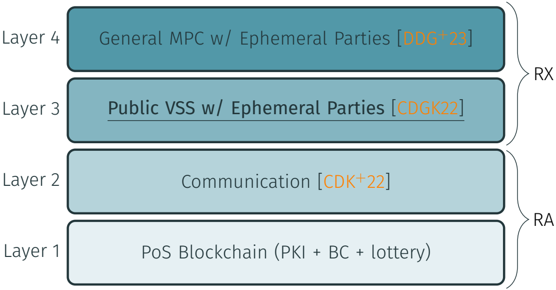
Overview

Layer 4 (Layered MPC):
Perfect General MPC over a layered graph using only ephemeral servers.

Layer 3 (YOLO-YOSO):
PVSS and resharing - basic building block for MPC and other applications.

Layer 2 (Encryption to the Future):
Communication towards unknown lottery winners.

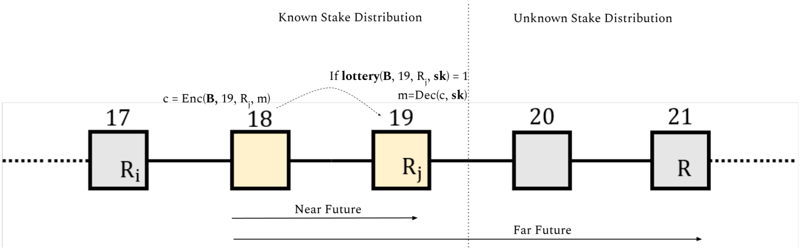
Layer 1:
Public PoS blockchain such as Ouroboros Praos [DGKR18].



Ephemeral Committees Model

Encryption to the near Future (Revisited).

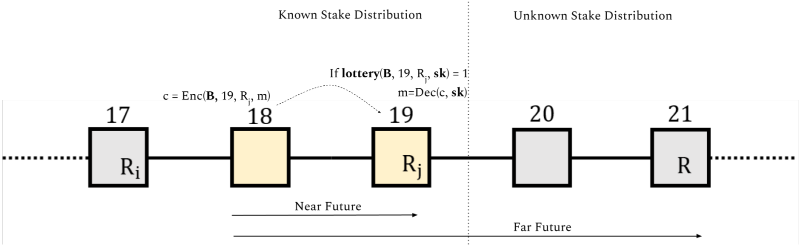
Existing solutions have several downsides:



Encryption to the near Future (Revisited).

Existing solutions have several downsides:

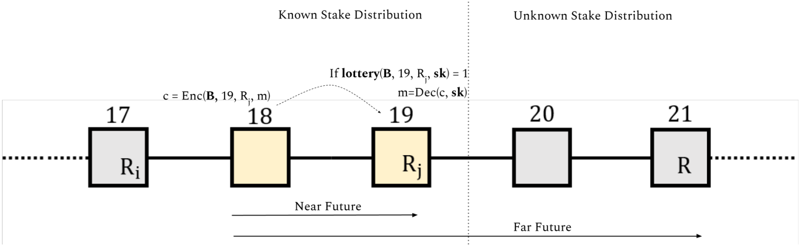
- Require an initial commitment phase when joining the network.



Encryption to the near Future (Revisited).

Existing solutions have several downsides:

- Require an initial commitment phase when joining the network.
- Have ciphertext length which is $O(N)$ or require interaction.

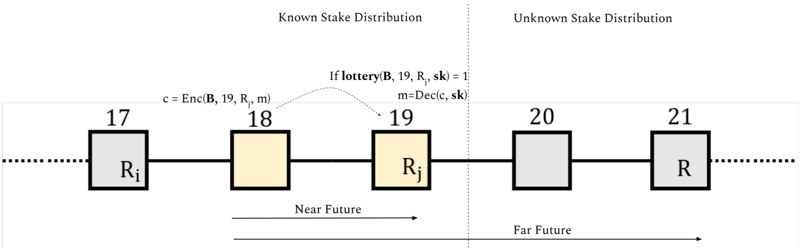


Overview: YOLO YOSO [CDGK22]

Encryption to the near Future (Revisited).

Existing solutions have several downsides:

- Require an initial commitment phase when joining the network.
- Have ciphertext length which is $O(N)$ or require interaction.
- Are not amendable to efficient techniques for proving correct resharing (relies on generic ZK proofs).



Overview: YOLO YOSO [CDGK22]

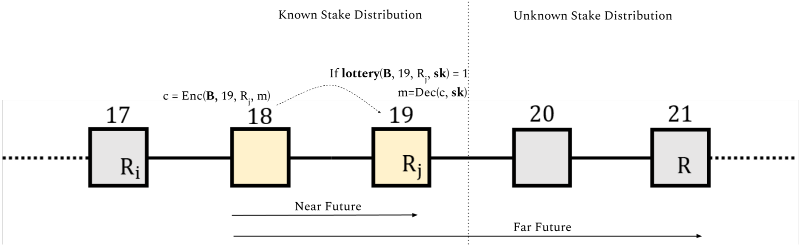
Encryption to the near Future (Revisited).

Existing solutions have several downsides:

- Require an initial commitment phase when joining the network.
- Have ciphertext length which is $O(N)$ or require interaction.
- Are not amendable to efficient techniques for proving correct resharing (relies on generic ZK proofs).

Main Question

Can we design an EtF (near) scheme with $O(1)$ ciphertext length and allows for building practical PVSS (amenable to efficient techniques for proof of correct sharing)?



Overview: YOLO YOSO [CDGK22]

Contributions:

- EtF (near) scheme:

Overview: YOLO YOSO [CDGK22]

Contributions:

- EtF (near) scheme:
  - That is concretely efficient using <u>shuffling</u>.

Overview: YOLO YOSO [CDGK22]

Contributions:

- EtF (near) scheme:
  - That is concretely efficient using shuffling.
  - Assuming anonymous broadcast, is both non-interactive and with ciphertext length $O(1)$.

Overview: YOLO YOSO [CDGK22]

Contributions:

- EtF (near) scheme:
  - That is concretely efficient using shuffling.
  - Assuming anonymous broadcast, is both non-interactive and with ciphertext length $O(1)$.
- Propose PVSS schemes compatible with this EtF (EtF + PVSS):

Overview: YOLO YOSO [CDGK22]

Contributions:

- EtF (near) scheme:
  - That is concretely efficient using shuffling.
  - Assuming anonymous broadcast, is both non-interactive and with ciphertext length $O(1)$.
- Propose PVSS schemes compatible with this EtF (EtF + PVSS):
  - Generic PVSS from any $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.

Overview: YOLO YOSO [CDGK22]

Contributions:

- EtF (near) scheme:
  - That is concretely efficient using shuffling.
  - Assuming anonymous broadcast, is both non-interactive and with ciphertext length $O(1)$.
- Propose PVSS schemes compatible with this EtF (EtF + PVSS):
  - Generic PVSS from any $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.
  - DDH-based PVSS where sharing proofs are public and independent of number of parties—(2 $\mathbb{Z}_p$-elements).

Overview: YOLO YOSO [CDGK22]

Contributions:

- EtF (near) scheme:
  - That is concretely efficient using shuffling.
  - Assuming anonymous broadcast, is both non-interactive and with ciphertext length $O(1)$.
- Propose PVSS schemes compatible with this EtF (EtF + PVSS):
  - Generic PVSS from any $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.
  - DDH-based PVSS where sharing proofs are public and independent of number of parties—(2 $\mathbb{Z}_p$-elements).
  - Efficient PVSS resharing protocols.

Overview: YOLO YOSO [CDGK22]

Contributions:

- EtF (near) scheme:
  - That is concretely efficient using shuffling.
  - Assuming anonymous broadcast, is both non-interactive and with ciphertext length $O(1)$.
- Propose PVSS schemes compatible with this EtF (EtF + PVSS):
  - Generic PVSS from any $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.
  - DDH-based PVSS where sharing proofs are public and independent of number of parties—(2 $\mathbb{Z}_p$-elements).
  - Efficient PVSS resharing protocols.
  - Applications to efficient distributed randomness generation and keeping secrets on a blockchain.

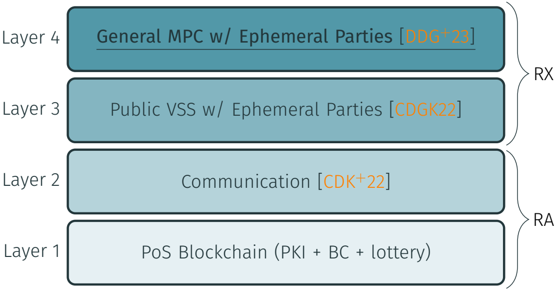
Overview

Layer 4 (Layered MPC):
Perfect General MPC over a layered graph using only ephemeral servers.

Layer 3 (YOLO-YOSO):
PVSS and resharing - basic building block for MPC and other applications.

Layer 2 (Encryption to the Future):
Communication towards unknown lottery winners.

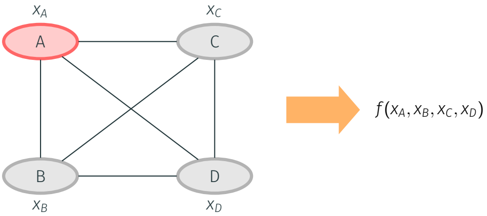
Layer 1:
Public PoS blockchain such as Ouroboros Praos [DGKR18].



Ephemeral Committees Model

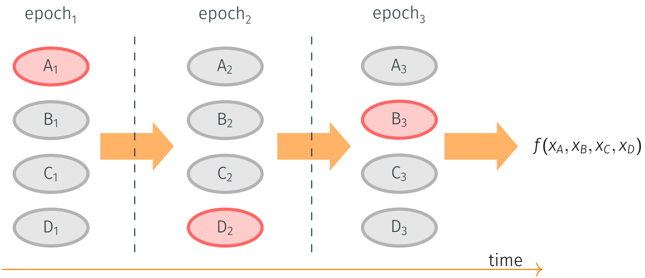
Overview: Layered MPC [DDG+23]

- [BGW88]: general MPC with perfect, full security and optimal corruption threshold ($t < n/3$).



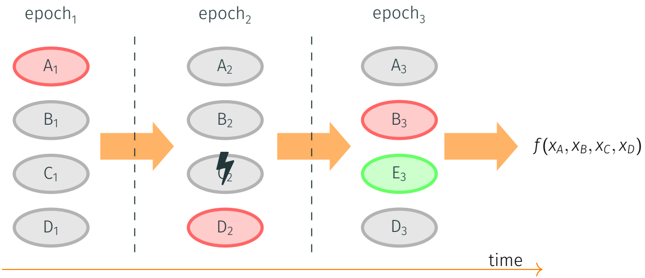
Overview: Layered MPC [DDG+23]

- [BGW88]: general MPC with perfect, full security and optimal corruption threshold ($t < n/3$).
- [OY91]: feasibility result of general MPC with **mobile adversary**
  - Show feasibility of general IT MPC [BGW88, RB89].
- [HJKY95, BELO14, CH01]: **Proactive** Secret Sharing & MPC.



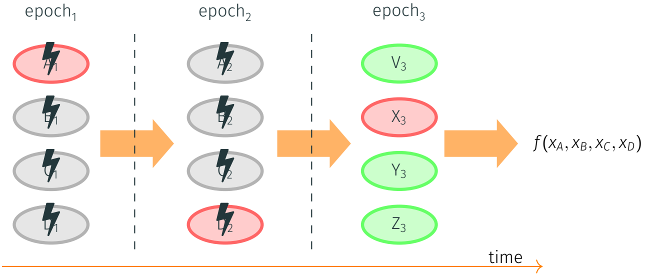
Overview: Layered MPC [DDG+23]

- [BGW88]: general MPC with perfect, full security and optimal corruption threshold ($t < n/3$).
- [OY91]: feasibility result of general MPC with **mobile adversary**
  - Show feasibility of general IT MPC [BGW88, RB89].
- [HJKY95, BELO14, CH01]: **Proactive** Secret Sharing & MPC.
- [DJ97, WWW02, MZW+19, ELL20]: **Dynamic Proactive** SS & MPC.

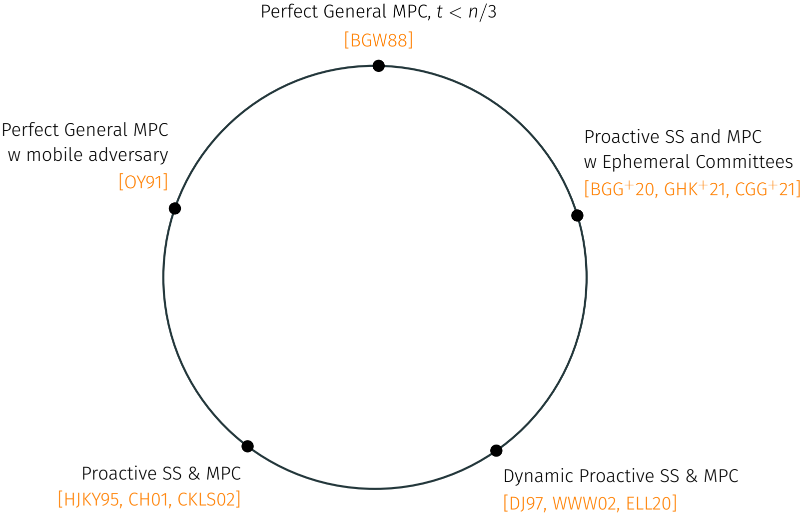


Overview: Layered MPC [DDG+23]

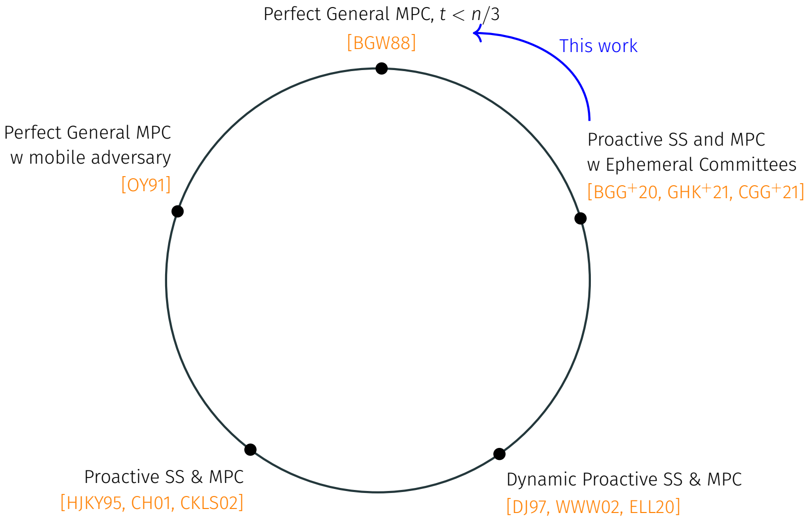
- [BGW88]: general MPC with perfect, full security and optimal corruption threshold ($t < n/3$).
- [OY91]: feasibility result of general MPC with **mobile adversary**
  - Show feasibility of general IT MPC [BGW88, RB89].
- [HJKY95, BELO14, CH01]: **Proactive** Secret Sharing & MPC.
- [DJ97, WWW02, MZW+19, ELL20]: **Dynamic Proactive** SS & MPC.
- [GHM+17, BGG+20, GHK+21, CGG+21]: Secret sharing and MPC using **ephemeral committees** (YOSO, Fluid).



Overview: Layered MPC [DDG+23]



Overview: Layered MPC [DDG+23]



Overview: Layered MPC [DDG+23]

Is it possible to construct MPC with **ephemeral committees** *achieving* **perfect full security** *against a* **maximally mobile adversary*** *while maintaining* **optimal corruption threshold**?

Overview: Layered MPC [DDG+23]

Area	Reference	|epoch|	Security	Corruption	Setup (BC+Chan.)
Proactive MPC	[HJKY95]	>1	Comp (full)	$t < n/2$	Next Round
	[OY91]	=1	Stat (full)	$t < n/c^{\dagger}$	Next Round
Ephemeral Committees	[GHK+21] (YOSO)	=1	Stat (full)	$\mathbb{E}[t] < n/2$	Any Future Round
	[CGG+21] (Fluid)	=1	Stat (abort)	$t < n/2$	Next Round

Overview: Layered MPC [DDG+23]

Area	Reference	|epoch|	Security	Corruption	Setup (BC+Chan.)
Proactive MPC	[HJKY95]	>1	Comp (full)	$t < n/2$	Next Round
	[OY91]	=1	Stat (full)	$t < n/c^{\dagger}$	Next Round
Ephemeral Committees	[GHK+21] (YOSO)	=1	Stat (full)	$\mathbb{E}[t] < n/2$	Any Future Round
	[CGG+21] (Fluid)	=1	Stat (abort)	$t < n/2$	Next Round
	This work	**=1**	**Perfect (full)**	$\mathbf{t < n/3}$	**Next Round**

Overview: Layered MPC [DDG+23]

Main Contribution:

- Formalize the model of **Layered MPC**—standard MPC with special interaction pattern and adversary structure.

Overview: Layered MPC [DDG+23]

Main Contribution:

- Formalize the model of **Layered MPC**—standard MPC with special interaction pattern and adversary structure.
- Present layered MPC protocols for general functionalities with **perfect, full security** and **optimal corruption threshold** $t < n/3$.
  - CNF (Replicated) Secret Sharing based protocols [GIKR01, Mau06].
  - Shamir Secret Sharing based protocols (efficient) [BGW88].

Overview: Layered MPC [DDG+23]

Main Contribution:

- Formalize the model of **Layered MPC**—standard MPC with special interaction pattern and adversary structure.
- Present layered MPC protocols for general functionalities with **perfect, full security** and **optimal corruption threshold** $t < n/3$.
  - CNF (Replicated) Secret Sharing based protocols [GIKR01, Mau06].
  - Shamir Secret Sharing based protocols (efficient) [BGW88].
- Improve on existing results on **maximally proactive** MPC protocols [OY91] and on new work on MPC with **ephemeral committees** [GHK+21, CGG+21].

Overview: Layered MPC [DDG+23]

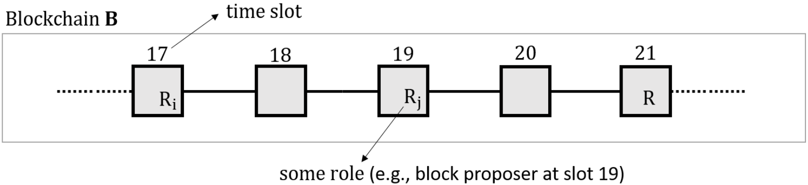
Main Contribution:

- Formalize the model of **Layered MPC**—standard MPC with special interaction pattern and adversary structure.
- Present layered MPC protocols for general functionalities with **perfect, full security** and **optimal corruption threshold** $t < n/3$.
  - CNF (Replicated) Secret Sharing based protocols [GIKR01, Mau06].
  - Shamir Secret Sharing based protocols (efficient) [BGW88].
- Improve on existing results on **maximally proactive** MPC protocols [OY91] and on new work on MPC with **ephemeral committees** [GHK+21, CGG+21].
- Present layered MPC protocols for general functionalities with **computational, full security** and $t < n/2$.

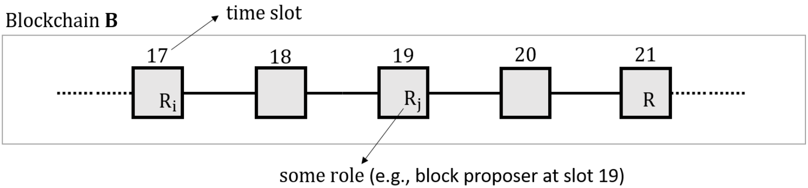
Contributions

Encryption to the Future [CDK+22]

Blockchain Lotteries

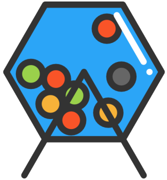


Blockchain Lotteries

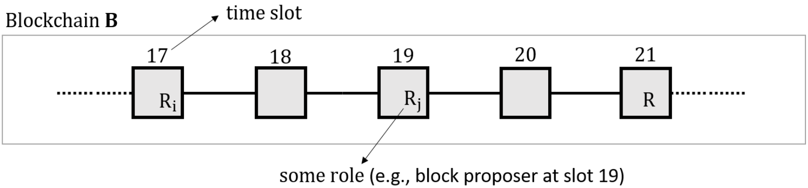


A self-selection mechanism that gives the winner the right to play a role R:

- propose a new block for the chain

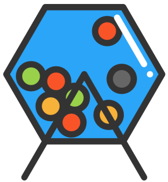


Blockchain Lotteries

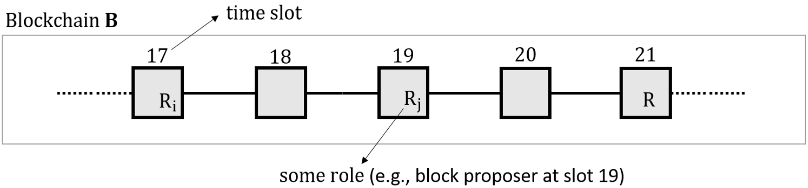


A self-selection mechanism that gives the winner the right to play a role R:

- propose a new block for the chain
- introduce new randomness

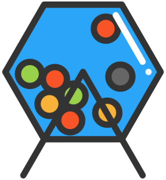


Blockchain Lotteries

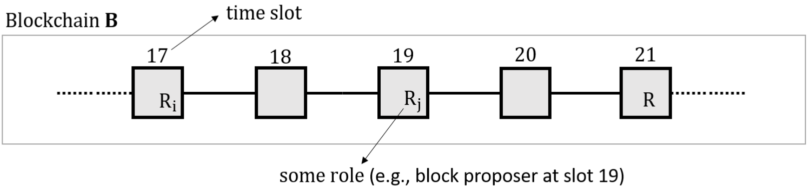


A self-selection mechanism that gives the winner the right to play a role R:

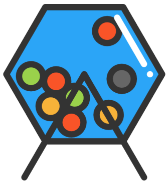
- propose a new block for the chain
- introduce new randomness
- become a member of a committee



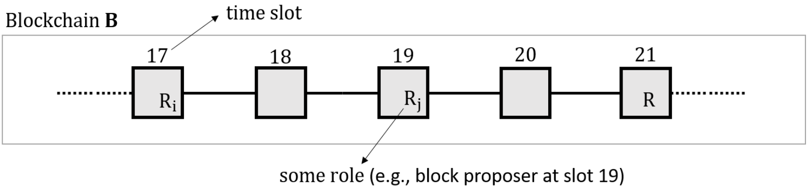
Blockchain Lotteries



Lottery Predicate. $\text{lottery}(\mathbf{B}, \text{slot}, \text{R}, \text{sk}_i) \in \{0, 1\}$

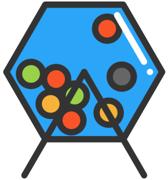


Blockchain Lotteries

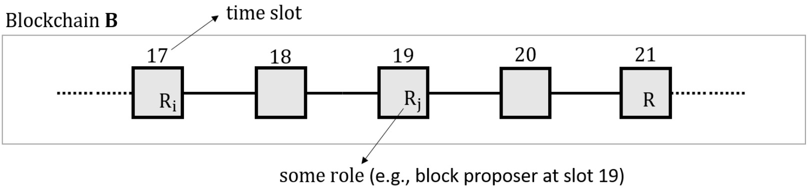


Lottery Predicate. $\text{lottery}(\mathbf{B}, \text{slot}, R, sk_i) \in \{0, 1\}$

- Anonymous Lotteries
  (e.g. VRF-based Cryptographic Sortition, Nakamoto PoW)

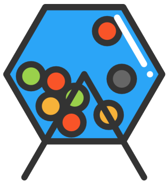


Blockchain Lotteries

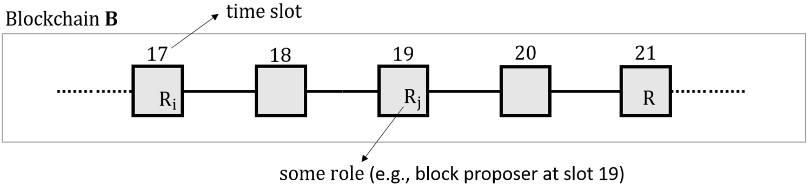


Lottery Predicate. $\text{lottery}(\mathbf{B}, \text{slot}, \text{R}, \text{sk}_i) \in \{0, 1\}$

- Anonymous Lotteries
  (e.g. VRF-based Cryptographic Sortition, Nakamoto PoW)
- Transparent Lotteries
  (e.g. "Round-Robin", "Follow-the-Satoshi")

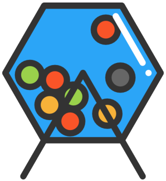


Blockchain Lotteries

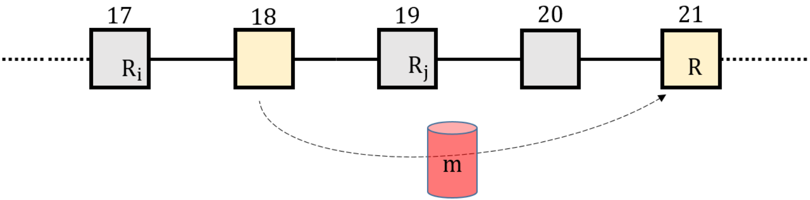


Lottery Predicate. $\mathrm{lottery}(\mathbf{B}, \mathrm{slot}, R, sk_i) \in \{0, 1\}$

- Anonymous Lotteries
  (e.g. VRF-based Cryptographic Sortition, Nakamoto PoW)
- Transparent Lotteries
  (e.g. "Round-Robin", "Follow-the-Satoshi")
  - No adaptive security!



Encryption to the Future

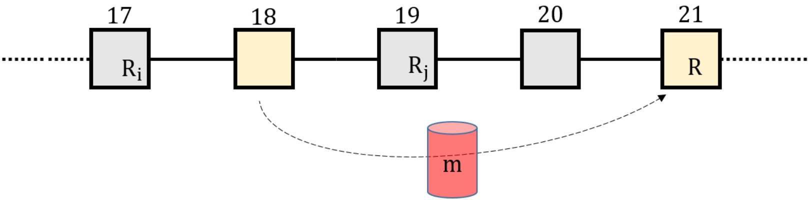


Encryption to the Future (EtF) w.r.t. lottery(**B**, slot, R, sk).

Encryption. $\mathsf{ct} \leftarrow \mathsf{Enc}(\hat{\mathbf{B}}, \mathsf{slot}, \mathsf{R}, m)$

Decryption. $m/\perp \leftarrow \mathsf{Dec}(\tilde{\mathbf{B}}, \mathsf{ct}, \mathsf{sk})$
Outputs m iff $\mathsf{lottery}(\tilde{\mathbf{B}}, \mathsf{slot}, \mathsf{R}, \mathsf{sk}) = 1$

Encryption to the Future



Encryption to the Future (EtF) w.r.t. lottery(**B**, slot, R, sk).

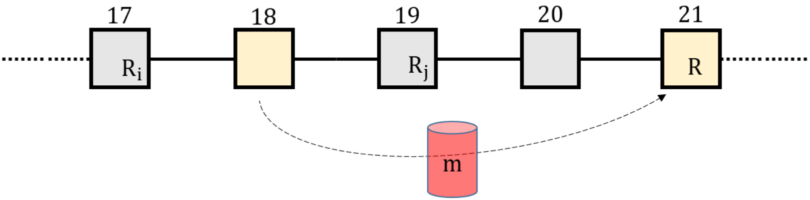
Encryption. ct $\leftarrow$ Enc($\hat{\mathbf{B}}$, slot, R, m)

Decryption. $m/\perp \leftarrow$ Dec($\tilde{\mathbf{B}}$, ct, sk)
Outputs m iff lottery($\tilde{\mathbf{B}}$, slot, R, sk) $= 1$

$\hat{\mathbf{B}} = \tilde{\mathbf{B}}$ (near future)

blockchain state is unchanged (known stake distribution)

Encryption to the Future



Encryption to the Future (EtF) w.r.t. lottery($\mathbf{B}$, slot, R, sk).

Encryption. $\mathsf{ct} \leftarrow \mathsf{Enc}(\hat{\mathbf{B}}, \mathsf{slot}, \mathsf{R}, m)$

Decryption. $m/\perp \leftarrow \mathsf{Dec}(\tilde{\mathbf{B}}, \mathsf{ct}, \mathsf{sk})$
Outputs m iff $\mathsf{lottery}(\tilde{\mathbf{B}}, \mathsf{slot}, \mathsf{R}, \mathsf{sk}) = 1$

$\hat{\mathbf{B}} = \tilde{\mathbf{B}}$ (near future)

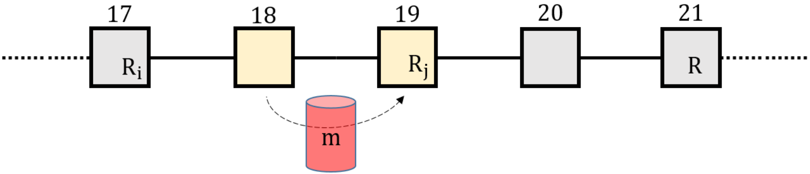
blockchain state is unchanged (known stake distribution)

$\hat{\mathbf{B}} \neq \tilde{\mathbf{B}}$ such that $\hat{\mathbf{B}}^{\lceil\kappa} \preceq \tilde{\mathbf{B}}$ (far future)

stake distribution is unknown at encryption time.

"Harder" to realize, similar to [GKM+20] and implies Blockchain WE.

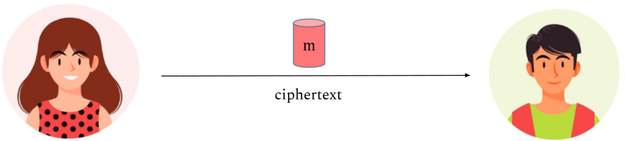
Encryption to the Future



Weaker Notion: Encryption to the Near Future

- Encryption w.r.t. lottery($\tilde{B}$, slot, R_j, sk)
- The state of blockchain when the lottery winner is decided is known at the time of encryption: $\hat{B} = \tilde{B}$
- Can be constructed from ”Witness Encryption over Commitments” (cWE)

Witness Encryption [?]



m: plaintext

Encryption under NP statement x

Decrypts ciphertext

Using witness w s.t. $(x, w) \in R$

Witness Encryption [?]

A Witness Encryption scheme for **NP** language $\mathcal{L}$ (and witness relation $\mathsf{R}_{\mathcal{L}}$).

Encrypt. $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{x}, m)$,

Decrypt. $m/\perp \leftarrow \mathsf{Dec}(\mathsf{ct}, \mathsf{w})$

Properties:

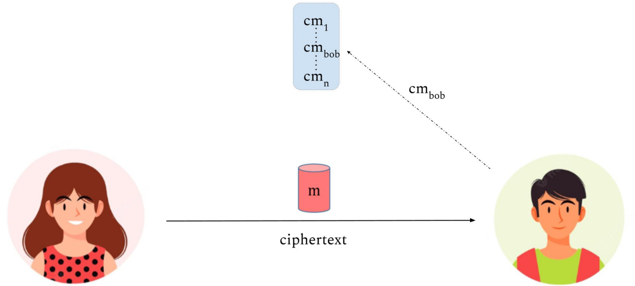
- Correctness: For any $\mathsf{x} \in \mathcal{L}$ such that $(\mathsf{x}, \mathsf{w}) \in \mathsf{R}_{\mathcal{L}}$

$$\Pr\left[\mathsf{Dec}(\mathsf{Enc}(x, m), \mathsf{w}) = m\right] = 1$$

- Security: For any PPT A, if $x \notin \mathcal{L}$ then

$$\Pr\left[\mathsf{A}(Enc(x, 0)) = 1\right] - \Pr\left[\mathsf{A}(Enc(x, 1)) = 1\right] \leq \mathsf{negl}(\lambda)$$

Witness Encryption over Commitments (cWE)



m: plaintext

Encryption under statement $x = (cm_{bob}, C, y)$

Decrypts ciphertext

Using witness $w = (s, \varrho)$ s.t. $C(s) = y$ AND $cm_{bob} = \mathrm{Commit}(s, \varrho)$

Witness Encryption over Commitments (cWE)

Witness Encryption over Commitments (cWE)

Setup Phase. Bob publishes a re-usable commitment $\mathsf{cm}_{bob} \leftarrow \mathsf{Commit}(\mathsf{ck}, \mathsf{s}; \rho)$

Witness Encryption over Commitments (cWE)

Setup Phase. Bob publishes a re-usable commitment
$\mathsf{cm}_{bob} \leftarrow \mathsf{Commit}(\mathsf{ck}, \mathsf{s}; \rho)$

Encrypt Phase. Define a language of statements $\mathsf{x} = (\mathsf{com}, C, y)$ and witnesses $\mathsf{w} = (\mathsf{s}, \rho)$.

Let $(\mathsf{x}, \mathsf{w}) \in \mathsf{R}$ iff
"com commits to s using randomness ρ such that $C(\mathsf{s}) = y$"

Witness Encryption over Commitments (cWE)

Setup Phase. Bob publishes a re-usable commitment $\mathsf{cm}_{bob} \leftarrow \mathsf{Commit}(\mathsf{ck}, \mathsf{s}; \rho)$

Encrypt Phase. Define a language of statements $\mathsf{x} = (\mathsf{com}, C, y)$ and witnesses $\mathsf{w} = (\mathsf{s}, \rho)$.

Let $(\mathsf{x}, \mathsf{w}) \in \mathsf{R}$ iff
"com commits to s using randomness ρ such that $C(\mathsf{s}) = y$"

Properties

- <u>Correctness</u>: For any $\mathsf{x} \in \mathcal{L}$ such that $(\mathsf{x}, \mathsf{w}) \in \mathsf{R}$

$$\Pr\left[\mathsf{Dec}(\mathsf{Enc}(x, m), \mathsf{w}) = m\right] = 1$$

Witness Encryption over Commitments (cWE)

Setup Phase. Bob publishes a re-usable commitment $\mathsf{cm}_{bob} \leftarrow \mathsf{Commit}(\mathsf{ck}, \mathsf{s}; \rho)$

Encrypt Phase. Define a language of statements $\mathsf{x} = (\mathsf{com}, C, y)$ and witnesses $\mathsf{w} = (\mathsf{s}, \rho)$.

Let $(\mathsf{x}, \mathsf{w}) \in \mathsf{R}$ iff
"com commits to s using randomness ρ such that $C(\mathsf{s}) = y$"

Properties

- Correctness: For any $\mathsf{x} \in \mathcal{L}$ such that $(\mathsf{x}, \mathsf{w}) \in \mathsf{R}$
$$\Pr\left[\mathsf{Dec}(\mathsf{Enc}(x, m), \mathsf{w}) = m\right] = 1$$
- Strong Semantic Security:

Witness Encryption over Commitments (cWE)

Setup Phase. Bob publishes a re-usable commitment
$\mathsf{cm}_{bob} \leftarrow \mathsf{Commit}(\mathsf{ck}, \mathsf{s}; \rho)$

Encrypt Phase. Define a language of statements $\mathsf{x} = (\mathsf{com}, C, y)$ and witnesses $\mathsf{w} = (\mathsf{s}, \rho)$.

Let $(\mathsf{x}, \mathsf{w}) \in \mathsf{R}$ iff
"com commits to s using randomness ρ such that $C(\mathsf{s}) = y$"

Properties

- Correctness: For any $\mathsf{x} \in \mathcal{L}$ such that $(\mathsf{x}, \mathsf{w}) \in \mathsf{R}$

$$\Pr\left[\mathsf{Dec}(\mathsf{Enc}(x, m), \mathsf{w}) = m\right] = 1$$

- Strong Semantic Security:
  1. Adversary receives $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{ck}, (\mathsf{com}, C, y), m)$ but does not know satisfying witness
  2. Adversary sees other $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{ck}, (\mathsf{com}_i, C, y), m)$ but without knowing the opening to com_i

Witness Encryption over Commitments (cWE)

Setup Phase. Bob publishes a re-usable commitment $\mathsf{cm}_{bob} \leftarrow \mathsf{Commit}(\mathsf{ck}, \mathsf{s}; \rho)$

Encrypt Phase. Define a language of statements $\mathsf{x} = (\mathsf{com}, C, y)$ and witnesses $\mathsf{w} = (\mathsf{s}, \rho)$.

Let $(\mathsf{x}, \mathsf{w}) \in \mathsf{R}$ iff
"com commits to s using randomness ρ such that $C(\mathsf{s}) = y$"

Properties

- <u>Correctness</u>: For any $\mathsf{x} \in \mathcal{L}$ such that $(\mathsf{x}, \mathsf{w}) \in \mathsf{R}$

$$\Pr\left[\mathsf{Dec}(\mathsf{Enc}(x, m), \mathsf{w}) = m\right] = 1$$

- <u>Strong Semantic Security</u>:
  1. Adversary receives $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{ck}, (\mathsf{com}, C, y), m)$ but does not know satisfying witness
  2. Adversary sees other $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{ck}, (\mathsf{com}_i, C, y), m)$ but without knowing the opening to com_i
  3. "The adversary has no advantage in guessing m if it cannot point to a commitment of a satisfying s where it knows the opening."

Encryption to the (near) Future

Obtain EtF (near) from Witness Encryption over Commitments (cWE)

Setup. Let each party publish a commitment $\mathsf{cm}_i \leftarrow \mathsf{Commit}(\mathsf{sk}_i; \rho)$ of the their lottery key

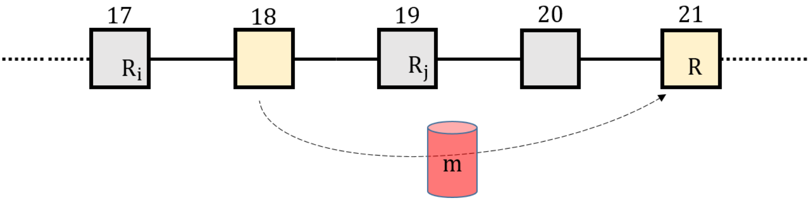
Encrypt. Let the circuit C encode the predicate $\mathsf{lottery}(\mathsf{B}, \mathsf{slot}, \mathsf{R}, \cdot)$.
Use the statement $\mathsf{x}_i = (\mathsf{com}_i, C, 1)$ for encryption.

Decrypt. The lottery-winning party with sk_i successfully decrypts since $C(\mathsf{sk}_i) = 1$.

Result:

- The first non-interactive (using no auxiliary committees) Role Assignment protocol.
- Encryption has to be done ”towards” every potential lottery winner—ciphertext length $O(N)$.
- For additional candidate constructions - read the paper.

Encryption to the Future



Encryption to the Future (EtF) w.r.t. $\mathsf{lottery}(\mathbf{B}, \mathsf{slot}, \mathsf{R}, \mathsf{sk})$.

Encryption. $\mathsf{ct} \leftarrow \mathsf{Enc}(\hat{\mathbf{B}}, \mathsf{slot}, \mathsf{R}, m)$

Decryption. $m/\perp \leftarrow \mathsf{Dec}(\tilde{\mathbf{B}}, \mathsf{ct}, \mathsf{sk})$
Outputs m iff $\mathsf{lottery}(\tilde{\mathbf{B}}, \mathsf{slot}, \mathsf{R}, \mathsf{sk}) = 1$

$\hat{\mathbf{B}} = \tilde{\mathbf{B}}$ (near future)

blockchain state is unchanged (known stake distribution)

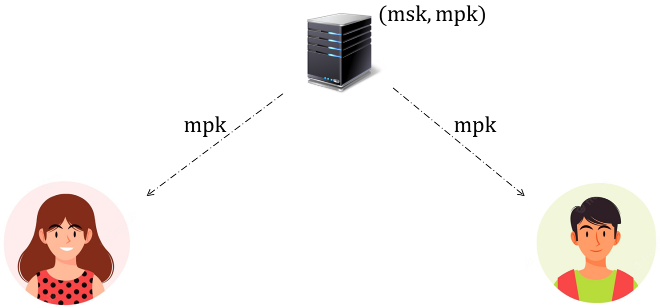
$\hat{\mathbf{B}} \neq \tilde{\mathbf{B}}$ such that $\hat{\mathbf{B}}^{\lceil \kappa} \preceq \tilde{\mathbf{B}}$ (far future)

stake distribution is unknown at encryption time.

~~"Harder" to realize, similar to [GKM+20] and implies Blockchain WE.~~

Easy to realize using EtF (near future) + TIBE scheme and use of auxiliary committees

Identity Based Encryption (IBE)



Identity Based Encryption (IBE)

(msk, mpk)

m: plaintext

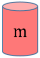
Encryption under mpk + ID_{Bob}

Identity Based Encryption (IBE)

$(\mathrm{msk}, \mathrm{mpk})$

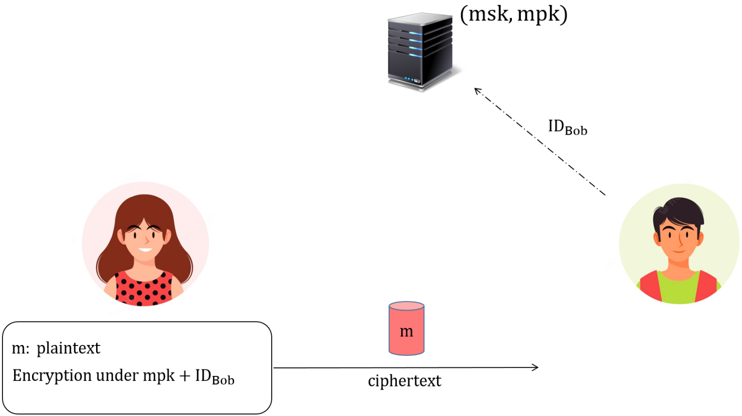
m: plaintext

Encryption under mpk + $\mathrm{ID}_{\mathrm{Bob}}$

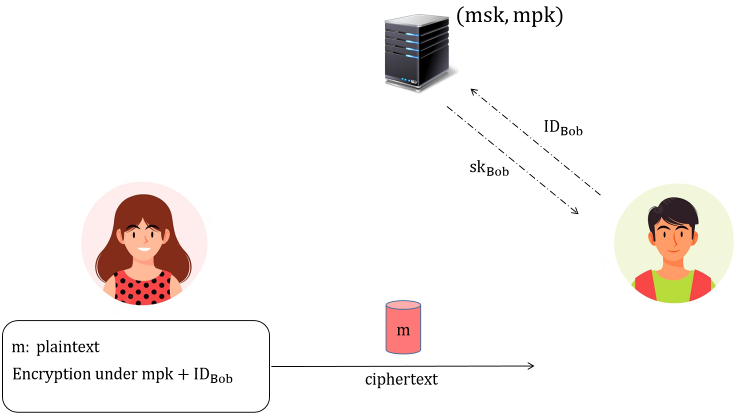


ciphertext

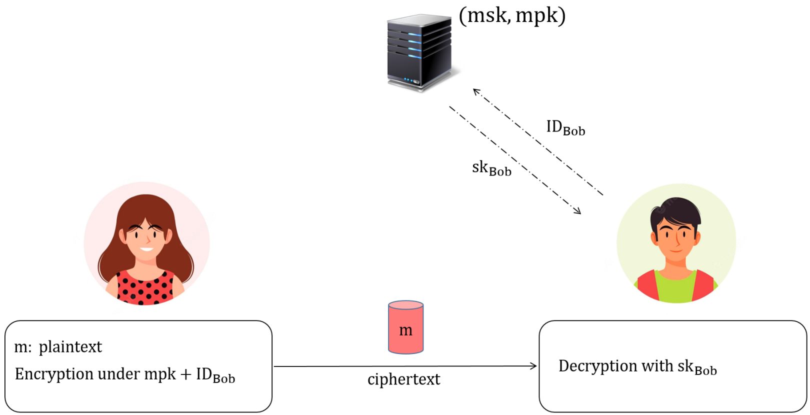
Identity Based Encryption (IBE)



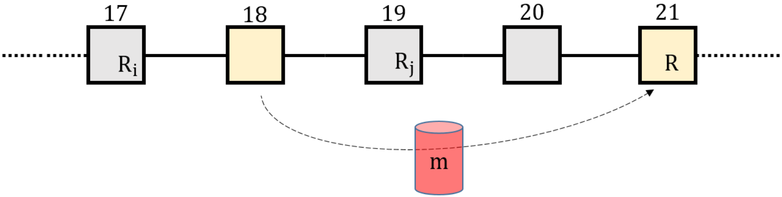
Identity Based Encryption (IBE)



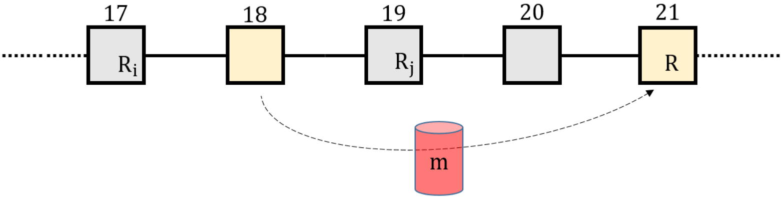
Identity Based Encryption (IBE)



Encryption to the Future with Committee

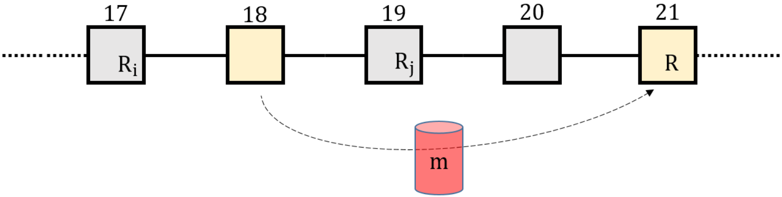


Encryption to the Future with Committee



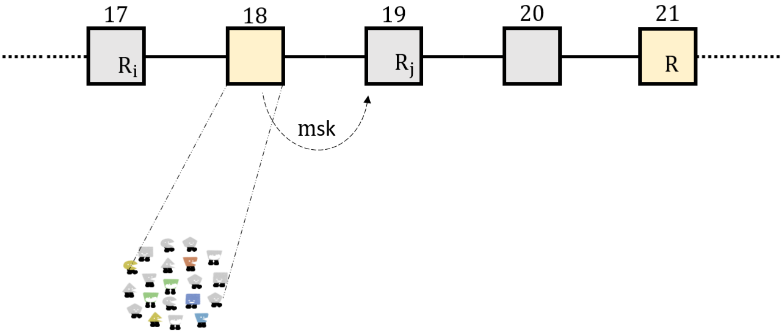
- **Encrypt.** Party publishes $ct \leftarrow \Pi_{TIBE}.\mathrm{Enc}(mpk, ID = (slot, R), m)$.

Encryption to the Future with Committee



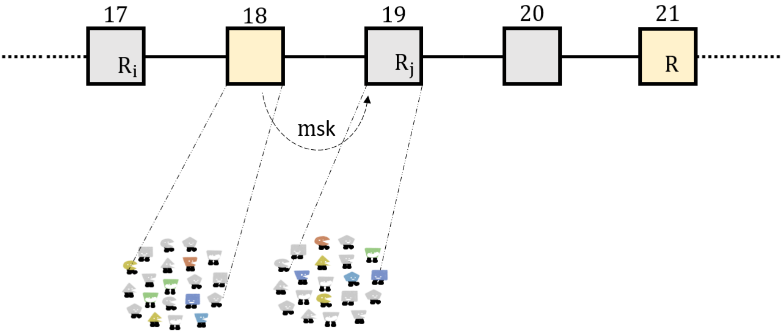
- **Setup.** (YOSO MPC) constructs the TIBE setup $(\mathsf{mpk}, \mathsf{msk} = (\mathsf{msk}_1, \ldots, \mathsf{msk}_n))$.
    1. $\mathsf{msk} = (\mathsf{msk}_1, \ldots, \mathsf{msk}_n)$ is proactively reshared through the slots in blockchain execution.

Encryption to the Future with Committee



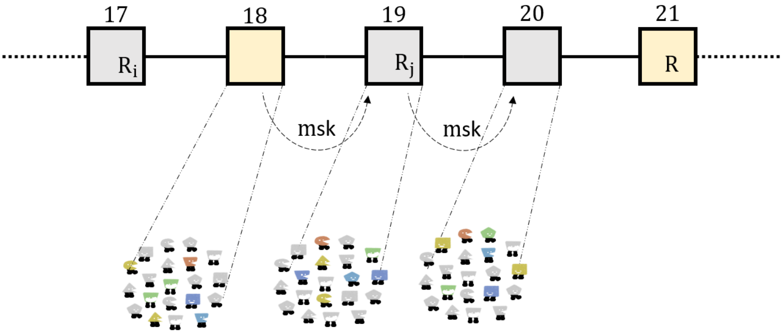
- **Setup.** (YOSO MPC) constructs the TIBE setup $(\mathsf{mpk}, \mathsf{msk} = (\mathsf{msk}_1, \ldots, \mathsf{msk}_n))$.
  1. $\mathsf{msk} = (\mathsf{msk}_1, \ldots, \mathsf{msk}_n)$ is proactively reshared through the slots in blockchain execution.

Encryption to the Future with Committee



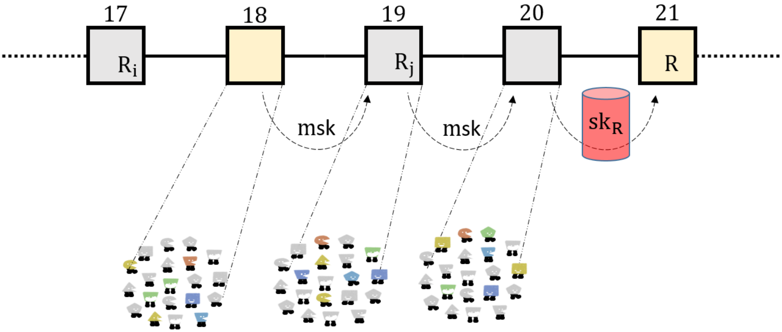
- **Setup.** (YOSO MPC) constructs the TIBE setup $(\mathrm{mpk}, \mathrm{msk} = (\mathrm{msk}_1, \ldots, \mathrm{msk}_n))$.
  1. $\mathrm{msk} = (\mathrm{msk}_1, \ldots, \mathrm{msk}_n)$ is proactively reshared through the slots in blockchain execution.
  2. Check if any EtF ciphertexts have a receiving $(\mathrm{slot}, \mathrm{R})$ that has been decided. If true, then:

Encryption to the Future with Committee



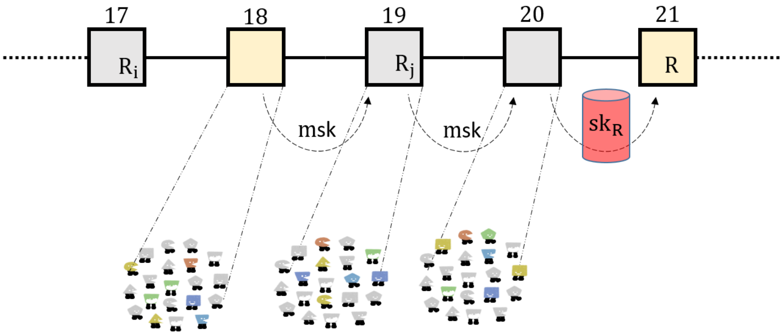
- **Setup.** (YOSO MPC) constructs the TIBE setup $(\mathsf{mpk}, \mathsf{msk} = (\mathsf{msk}_1, \ldots, \mathsf{msk}_n))$.
  1. $\mathsf{msk} = (\mathsf{msk}_1, \ldots, \mathsf{msk}_n)$ is proactively reshared through the slots in blockchain execution.
  2. Check if any EtF ciphertexts have a receiving (slot, R) that has been decided. If true, then:
     - Sample share of the IBE key for (slot, R) $\mathsf{sk}^i_{(\mathsf{slot},\mathsf{R})} \leftarrow \Pi_{\mathsf{TIBE}}.\mathsf{IDKeygen}(\mathsf{msk}_i, (\mathsf{slot}, \mathsf{R}))$

Encryption to the Future with Committee



- **Setup.** (YOSO MPC) constructs the TIBE setup $(\mathsf{mpk}, \mathsf{msk} = (\mathsf{msk}_1, \ldots, \mathsf{msk}_n))$.
  1. $\mathsf{msk} = (\mathsf{msk}_1, \ldots, \mathsf{msk}_n)$ is proactively reshared through the slots in blockchain execution.
  2. Check if any EtF ciphertexts have a receiving $(\mathsf{slot}, \mathsf{R})$ that has been decided. If true, then:
     - Sample share of the IBE key for $(\mathsf{slot}, \mathsf{R})$ $\mathsf{sk}^i_{(\mathsf{slot},\mathsf{R})} \leftarrow \Pi_{\mathsf{TIBE}}.\mathsf{IDKeygen}(\mathsf{msk}_i, (\mathsf{slot}, \mathsf{R}))$
     - Send shares of ID-key by EtF (near) $\mathsf{ct}^{\mathsf{sk},i}_{(\mathsf{slot},\mathsf{R})} \leftarrow \Pi_{\mathsf{EtF}}.\mathsf{Enc}(\mathsf{B}, \mathsf{slot}, \mathsf{R}, \mathsf{sk}^i_{(\mathsf{slot},\mathsf{R})})$

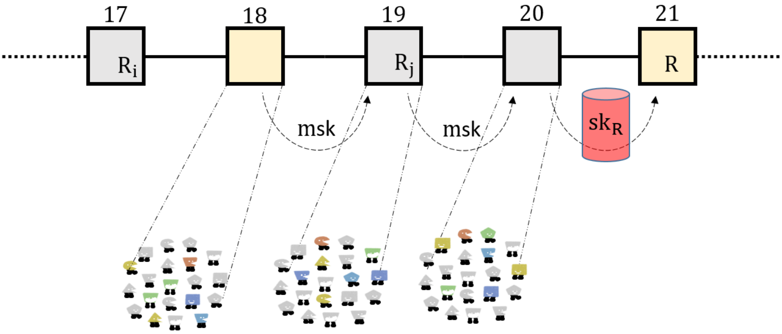
Encryption to the Future with Committee



Setup. […]

Encrypt. Party publishes $\mathsf{ct} \leftarrow \Pi_{\mathsf{TIBE}}.\mathsf{Enc}(\mathsf{mpk}, \mathsf{ID} = (\mathsf{slot}, \mathsf{R}), m)$.

Encryption to the Future with Committee



Setup. […]

Encrypt. Party publishes $\mathsf{ct} \leftarrow \Pi_{\mathsf{TIBE}}.\mathsf{Enc}(\mathsf{mpk}, \mathsf{ID} = (\mathsf{slot}, \mathsf{R}), m)$.

Decrypt. The lottery-winner for $(\mathsf{slot}, \mathsf{R})$ decrypts EtF (near) ciphertexts and combine shares $\{\mathsf{sk}^i_{(\mathsf{slot},\mathsf{R})}\}$ to obtain $\mathsf{sk}_{(\mathsf{slot},\mathsf{R})}$. Finally outputs $m \leftarrow \Pi_{\mathsf{TIBE}}.\mathsf{Dec}(\mathsf{sk}_{(\mathsf{slot},\mathsf{R})}, \mathsf{ct})$.

YOLO YOSO [CDGK22]

YOLO YOSO: EtF (near) using Shuffling

- We propose a simple EtF approach:
    - Each party will be associated with an anonymous key pair (PKE).

YOLO YOSO: EtF (near) using Shuffling

- We propose a simple EtF approach:
  - Each party will be associated with an anonymous key pair (PKE).
  - Concretely, each party inputs their PK $\mathsf{pk}_{\mathcal{E},i}$ into a mixnet. The resulting list of PKs is published on the blockchain with $\mathsf{pk}_{\mathsf{Anon},\psi(i)} = \mathsf{pk}_{\mathcal{E},i}$ for random permutation ψ.

YOLO YOSO: EtF (near) using Shuffling

- We propose a simple EtF approach:
  - Each party will be associated with an anonymous key pair (PKE).
  - Concretely, each party inputs their PK $\mathsf{pk}_{\mathcal{E},i}$ into a mixnet. The resulting list of PKs is published on the blockchain with $\mathsf{pk}_{\mathsf{Anon},\psi(i)} = \mathsf{pk}_{\mathcal{E},i}$ for random permutation ψ.
  - Lottery will select an unused key in the list.

Algorithm 3 $\mathsf{lottery}(\mathsf{B}, \mathsf{slot}, \mathsf{R}, \mathsf{sk}_{L,i})$

$(\{(j, \mathsf{pk}_{\mathsf{Anon},j})\}_{j\in[n]}, \eta) \leftarrow \mathsf{param}(\mathsf{B}, \mathsf{slot})$

$(\mathsf{pk}_{\mathcal{E},i}, \mathsf{sk}_{\mathcal{E},i}) \leftarrow \mathsf{sk}_{L,i}$

$k \leftarrow \mathcal{H}(\mathsf{slot}||\mathsf{R}||\eta)$

return 1 iff $\mathsf{pk}_{\mathcal{E},i} = \mathsf{pk}_{\mathsf{Anon},k}$

YOLO YOSO: EtF (near) using Shuffling

- We propose a simple EtF approach:
  - Each party will be associated with an anonymous key pair (PKE).
  - Concretely, each party inputs their PK $\mathsf{pk}_{\mathcal{E},i}$ into a mixnet. The resulting list of PKs is published on the blockchain with $\mathsf{pk}_{\mathsf{Anon},\psi(i)} = \mathsf{pk}_{\mathcal{E},i}$ for random permutation ψ.
  - Lottery will select an unused key in the list.
- More elaborate strategy using Camenisch-Lysyanskaya signatures: preserves anonymity (among committee members) even after speaking.

Algorithm 4 lottery(B, slot, R, $\mathsf{sk}_{L,i}$)

1: $(\{(j, \mathsf{pk}_{\mathsf{Anon},j})\}_{j\in[n]}, \eta) \leftarrow \mathsf{param}(\mathsf{B}, \mathsf{slot})$
2: $(\mathsf{pk}_{\mathcal{E},i}, \mathsf{sk}_{\mathcal{E},i}) \leftarrow \mathsf{sk}_{L,i}$
3: $k \leftarrow \mathcal{H}(\mathsf{slot}||\mathsf{R}||\eta)$
4: **return** 1 iff $\mathsf{pk}_{\mathcal{E},i} = \mathsf{pk}_{\mathsf{Anon},k}$

YOLO YOSO: EtF (near) using Shuffling

- We propose a simple EtF approach:
  - Each party will be associated with an anonymous key pair (PKE).
  - Concretely, each party inputs their PK $\mathsf{pk}_{\mathcal{E},i}$ into a mixnet. The resulting list of PKs is published on the blockchain with $\mathsf{pk}_{\mathsf{Anon},\psi(i)} = \mathsf{pk}_{\mathcal{E},i}$ for random permutation ψ.
  - Lottery will select an unused key in the list.
- More elaborate strategy using Camenisch-Lysyanskaya signatures: preserves anonymity (among committee members) even after speaking.
- This is good enough for EtF, but how about secret sharing to a committee? How do we prove consistency between shares?

Algorithm 5 lottery$(\mathsf{B}, \mathsf{slot}, \mathsf{R}, \mathsf{sk}_{L,i})$

$(\{(j, \mathsf{pk}_{\mathsf{Anon},j})\}_{j\in[n]}, \eta) \leftarrow \mathsf{param}(\mathsf{B}, \mathsf{slot})$
$(\mathsf{pk}_{\mathcal{E},i}, \mathsf{sk}_{\mathcal{E},i}) \leftarrow \mathsf{sk}_{L,i}$
$k \leftarrow \mathcal{H}(\mathsf{slot}||\mathsf{R}||\eta)$
return 1 iff $\mathsf{pk}_{\mathcal{E},i} = \mathsf{pk}_{\mathsf{Anon},k}$

Publicly Verifiable Secret Sharing (PVSS) [Sch99]

Setup …

Publicly Verifiable Secret Sharing (PVSS) [Sch99]

Setup …

Distribution

- $\underline{\mathsf{Dist}(pp, \mathsf{pk}_D, \mathsf{sk}_D, \{\mathsf{pk}_i : i \in [n]\}, S)}$ performed by the dealer, and where $S \in \mathbb{G}$ is a secret, outputs encrypted shares $C_i : i \in [n]$ and a proof $\mathsf{Pf}_{\mathsf{Sh}}$ of sharing correctness.

Publicly Verifiable Secret Sharing (PVSS) [Sch99]

Setup ...

Distribution

- $\underline{\mathsf{Dist}(pp, \mathsf{pk}_D, \mathsf{sk}_D, \{\mathsf{pk}_i : i \in [n]\}, S)}$ performed by the dealer, and where $S \in \mathbb{G}$ is a secret, outputs encrypted shares $C_i : i \in [n]$ and a proof $\mathsf{Pf}_{\mathsf{Sh}}$ of sharing correctness.

Verification

- $\underline{\mathsf{Verify}(pp, \mathsf{pk}_D, \{(\mathsf{pk}_i, C_i) : i \in [n]\}, \mathsf{Pf}_{\mathsf{Sh}})}$ performed by the public verifier outputs 0/1 (as a verdict on whether the sharing is valid)

Publicly Verifiable Secret Sharing (PVSS) [Sch99]

Setup ...

Distribution

- $\underline{\mathsf{Dist}(pp, \mathsf{pk}_D, \mathsf{sk}_D, \{\mathsf{pk}_i : i \in [n]\}, S)}$ performed by the dealer, and where $S \in \mathbb{G}$ is a secret, outputs encrypted shares $C_i : i \in [n]$ and a proof $\mathsf{Pf}_{\mathsf{Sh}}$ of sharing correctness.

Verification

- $\underline{\mathsf{Verify}(pp, \mathsf{pk}_D, \{(\mathsf{pk}_i, C_i) : i \in [n]\}, \mathsf{Pf}_{\mathsf{Sh}})}$ performed by the public verifier outputs 0/1 (as a verdict on whether the sharing is valid)

Reconstruction

- $\underline{\mathsf{DecShare}(pp, \mathsf{pk}_D, \mathsf{pk}_i, \mathsf{sk}_i, C_i)}$, performed by a share receiver, outputs decrypted share A_i and proof $\mathsf{Pf}_{\mathsf{Dec}\,i}$ of correct decryption.

Publicly Verifiable Secret Sharing (PVSS) [Sch99]

Setup …

Distribution

- $\underline{\mathsf{Dist}(pp, \mathsf{pk}_D, \mathsf{sk}_D, \{\mathsf{pk}_i : i \in [n]\}, S)}$ performed by the dealer, and where $S \in \mathbb{G}$ is a secret, outputs encrypted shares $C_i : i \in [n]$ and a proof $\mathsf{Pf}_{\mathsf{Sh}}$ of sharing correctness.

Verification

- $\underline{\mathsf{Verify}(pp, \mathsf{pk}_D, \{(\mathsf{pk}_i, C_i) : i \in [n]\}, \mathsf{Pf}_{\mathsf{Sh}})}$ performed by the public verifier outputs 0/1 (as a verdict on whether the sharing is valid)

Reconstruction

- $\underline{\mathsf{DecShare}(pp, \mathsf{pk}_D, \mathsf{pk}_i, \mathsf{sk}_i, C_i)}$, performed by a share receiver, outputs decrypted share A_i and proof $\mathsf{Pf}_{\mathsf{Dec}\,i}$ of correct decryption.
- $\underline{\mathsf{VerifyDec}(pp, \mathsf{pk}_D, C_i, A_i, \mathsf{Pf}_{\mathsf{Dec}\,i})}$ outputs 0/1 (as a verdict on whether A_i is a valid decryption of C_i)

Publicly Verifiable Secret Sharing (PVSS) [Sch99]

Setup ...

Distribution

- Dist($pp, \mathrm{pk}_D, \mathrm{sk}_D, \{\mathrm{pk}_i : i \in [n]\}, S$) performed by the dealer, and where $S \in \mathbb{G}$ is a secret, outputs encrypted shares $C_i : i \in [n]$ and a proof $\mathrm{Pf}_{\mathrm{Sh}}$ of sharing correctness.

Verification

- Verify($pp, \mathrm{pk}_D, \{(\mathrm{pk}_i, C_i) : i \in [n]\}, \mathrm{Pf}_{\mathrm{Sh}}$) performed by the public verifier outputs 0/1 (as a verdict on whether the sharing is valid)

Reconstruction

- DecShare($pp, \mathrm{pk}_D, \mathrm{pk}_i, \mathrm{sk}_i, C_i$), performed by a share receiver, outputs decrypted share A_i and proof $\mathrm{Pf}_{\mathrm{Dec}\,i}$ of correct decryption.
- VerifyDec($pp, \mathrm{pk}_D, C_i, A_i, \mathrm{Pf}_{\mathrm{Dec}\,i}$) outputs 0/1 (as a verdict on whether A_i is a valid decryption of C_i)
- Rec($pp, \{A_i : i \in \mathcal{T}\}$) for some $\mathcal{T} \subseteq [n]$ of size $t+1$ outputs a secret S.

Publicly Verifiable Secret Sharing (PVSS) [Sch99]

Setup ...

Distribution

- Dist(pp, pk_D, sk_D, $\{\mathsf{pk}_i : i \in [n]\}$, S) performed by the dealer, and where $S \in \mathbb{G}$ is a secret, outputs encrypted shares $C_i : i \in [n]$ and a proof $\mathsf{Pf}_{\mathsf{Sh}}$ of sharing correctness.

Verification

- Verify(pp, pk_D, $\{(\mathsf{pk}_i, C_i) : i \in [n]\}$, $\mathsf{Pf}_{\mathsf{Sh}}$) performed by the public verifier outputs 0/1 (as a verdict on whether the sharing is valid)

Reconstruction

- DecShare(pp, pk_D, pk_i, sk_i, C_i), performed by a share receiver, outputs decrypted share A_i and proof $\mathsf{Pf}_{\mathsf{Dec}\,i}$ of correct decryption.
- VerifyDec(pp, pk_D, C_i, A_i, $\mathsf{Pf}_{\mathsf{Dec}\,i}$) outputs 0/1 (as a verdict on whether A_i is a valid decryption of C_i)
- Rec(pp, $\{A_i : i \in \mathcal{T}\}$) for some $\mathcal{T} \subseteq [n]$ of size $t + 1$ outputs a secret S.

Our constructions satisfy Correctness, Verifiability and Indistinguishability of Secrets.

Publicly Verifiable Secret Sharing (PVSS) [Sch99]

Setup ...

Distribution

- Dist($pp, \mathsf{pk}_D, \mathsf{sk}_D, \{\mathsf{pk}_i : i \in [n]\}, S$) performed by the dealer, and where $S \in \mathbb{G}$ is a secret, outputs encrypted shares $C_i : i \in [n]$ and a **proof** $\mathsf{Pf}_{\mathsf{Sh}}$ **of sharing correctness**.

Verification

- Verify($pp, \mathsf{pk}_D, \{(\mathsf{pk}_i, C_i) : i \in [n]\}, \mathsf{Pf}_{\mathsf{Sh}}$) performed by the public verifier outputs 0/1 (as a verdict on whether the sharing is valid)

Reconstruction

- DecShare($pp, \mathsf{pk}_D, \mathsf{pk}_i, \mathsf{sk}_i, C_i$), performed by a share receiver, outputs decrypted share A_i and **proof** $\mathsf{Pf}_{\mathsf{Dec}\,i}$ **of correct decryption**.
- VerifyDec($pp, \mathsf{pk}_D, C_i, A_i, \mathsf{Pf}_{\mathsf{Dec}\,i}$) outputs 0/1 (as a verdict on whether A_i is a valid decryption of C_i)
- Rec($pp, \{A_i : i \in \mathcal{T}\}$) for some $\mathcal{T} \subseteq [n]$ of size $t+1$ outputs a secret S.

Our constructions satisfy Correctness, Verifiability and Indistinguishability of Secrets.

PVSS Constructions

We present two constructions of PVSS:

- HE-PVSS:

PVSS Constructions

We present two constructions of PVSS:

- HE-PVSS:
  - Generic PVSS from a $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.

PVSS Constructions

We present two constructions of PVSS:

- HE-PVSS:
  - Generic PVSS from a $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.
  - Plaintext, randomness, ciphertext each have a $\mathbb{Z}_p$-vector space structure (e.g. groups of order p).

PVSS Constructions

We present two constructions of PVSS:

- HE-PVSS:
  - Generic PVSS from a $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.
  - Plaintext, randomness, ciphertext each have a $\mathbb{Z}_p$-vector space structure (e.g. groups of order p).
  - $\mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_1; \rho_1) \boxplus_{\mathfrak{C}} \mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_2; \rho_2) = \mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_1 \boxplus_{\mathfrak{P}} m_2; \rho_1 \boxplus_{\mathfrak{R}} \rho_2)$

PVSS Constructions

We present two constructions of PVSS:

- HE-PVSS:
  - Generic PVSS from a $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.
  - Plaintext, randomness, ciphertext each have a $\mathbb{Z}_p$-vector space structure (e.g. groups of order p).
  - $\mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_1; \rho_1) \boxplus_{\mathfrak{C}} \mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_2; \rho_2) = \mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_1 \boxplus_{\mathfrak{P}} m_2; \rho_1 \boxplus_{\mathfrak{R}} \rho_2)$
  - Allows for simple "Schnorr-like" PoK of plaintext.
  - And simple proof of correct $\mathbb{Z}_p$-Linear decryption (e.g. ElGamal)

PVSS Constructions

We present two constructions of PVSS:

- **HE-PVSS**:
  - Generic PVSS from a $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.
  - Plaintext, randomness, ciphertext each have a $\mathbb{Z}_p$-vector space structure (e.g. groups of order p).
  - $\mathcal{E}.\mathrm{Enc}_{\mathrm{pk}}(m_1; \rho_1) \boxplus_{\mathfrak{C}} \mathcal{E}.\mathrm{Enc}_{\mathrm{pk}}(m_2; \rho_2) = \mathcal{E}.\mathrm{Enc}_{\mathrm{pk}}(m_1 \boxplus_{\mathfrak{P}} m_2; \rho_1 \boxplus_{\mathfrak{R}} \rho_2)$
  - Allows for simple ”Schnorr-like” PoK of plaintext.
  - And simple proof of correct $\mathbb{Z}_p$-Linear decryption (e.g. ElGamal)
- **DH-PVSS**:
  - We present the first PVSS with **constant size overhead**.

PVSS Constructions

We present two constructions of PVSS:

- HE-PVSS:
  - Generic PVSS from a $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.
  - Plaintext, randomness, ciphertext each have a $\mathbb{Z}_p$-vector space structure (e.g. groups of order p).
  - $\mathcal{E}.\mathrm{Enc}_{\mathrm{pk}}(m_1; \rho_1) \boxplus_{\mathfrak{C}} \mathcal{E}.\mathrm{Enc}_{\mathrm{pk}}(m_2; \rho_2) = \mathcal{E}.\mathrm{Enc}_{\mathrm{pk}}(m_1 \boxplus_{\mathfrak{P}} m_2; \rho_1 \boxplus_{\mathfrak{R}} \rho_2)$
  - Allows for simple "Schnorr-like" PoK of plaintext.
  - And simple proof of correct $\mathbb{Z}_p$-Linear decryption (e.g. ElGamal)
- DH-PVSS:
  - We present the first PVSS with **constant size overhead**.
  - The dealer has an initial key-pair (pk_D, sk_D) to enable the "SCRAPE check".

PVSS Constructions

We present two constructions of PVSS:

- **HE-PVSS**:
  - Generic PVSS from a $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.
  - Plaintext, randomness, ciphertext each have a $\mathbb{Z}_p$-vector space structure (e.g. groups of order p).
  - $\mathcal{E}.\mathrm{Enc}_{\mathrm{pk}}(m_1; \rho_1) \boxplus_{\mathfrak{C}} \mathcal{E}.\mathrm{Enc}_{\mathrm{pk}}(m_2; \rho_2) = \mathcal{E}.\mathrm{Enc}_{\mathrm{pk}}(m_1 \boxplus_{\mathfrak{P}} m_2; \rho_1 \boxplus_{\mathfrak{R}} \rho_2)$
  - Allows for simple "Schnorr-like" PoK of plaintext.
  - And simple proof of correct $\mathbb{Z}_p$-Linear decryption (e.g. ElGamal)
- **DH-PVSS**:
  - We present the first PVSS with **constant size overhead**.
  - The dealer has an initial key-pair (pk_D, sk_D) to enable the "SCRAPE check".
  - Secret $S = s \cdot G$ is in group $\mathbb{G} = \langle G \rangle$ of order p, where DDH is hard.

PVSS Constructions

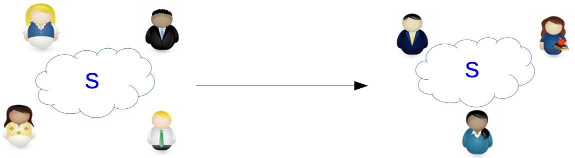
We present two constructions of PVSS:

- **HE-PVSS**:
  - Generic PVSS from a $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.
  - Plaintext, randomness, ciphertext each have a $\mathbb{Z}_p$-vector space structure (e.g. groups of order p).
  - $\mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_1; \rho_1) \boxplus_{\mathfrak{C}} \mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_2; \rho_2) = \mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_1 \boxplus_{\mathfrak{P}} m_2; \rho_1 \boxplus_{\mathfrak{R}} \rho_2)$
  - Allows for simple "Schnorr-like" PoK of plaintext.
  - And simple proof of correct $\mathbb{Z}_p$-Linear decryption (e.g. ElGamal)
- **DH-PVSS**:
  - We present the first PVSS with **constant size overhead**.
  - The dealer has an initial key-pair (pk_D, sk_D) to enable the "SCRAPE check".
  - Secret $S = s \cdot G$ is in group $\mathbb{G} = \langle G \rangle$ of order p, where DDH is hard.
  - Dealer publishes:
    - n encrypted shares: each 1 element in $\mathbb{G}$.
    - Correctness proof: $\underline{2\ \mathbb{Z}_p \text{ elements.}}$

Resharing and Reconstruction

Resharing:

1. Transfer S from $\mathcal{C}_{\mathrm{r}}$ to $\mathcal{C}_{\mathrm{r}+1}$ where $|\mathcal{C}_k| = n_k$ with threshold t_k.

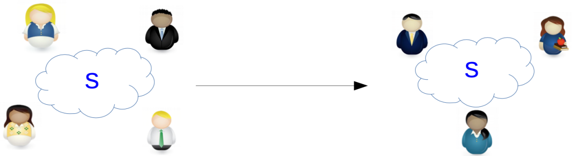


Resharing and Reconstruction

Resharing:

1. Transfer S from $\mathcal{C}_\mathrm{r}$ to $\mathcal{C}_{\mathrm{r}+1}$ where $|\mathcal{C}_k| = n_k$ with threshold t_k.
2. Each party $\mathrm{R}_{\mathrm{r},i}$ in committee $\mathcal{C}_\mathrm{r}$ has $A_{\mathrm{r},i}$ as share with public encryption

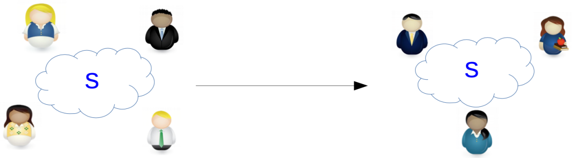
$$C_{\mathrm{r},i} = \mathcal{E}.\mathrm{Enc}_{\mathrm{pk}_{\mathrm{r},i}}(A_{\mathrm{r},i})$$



Resharing and Reconstruction

Resharing:

1. Transfer S from $\mathcal{C}_\mathrm{r}$ to $\mathcal{C}_{\mathrm{r}+1}$ where $|\mathcal{C}_k| = n_k$ with threshold t_k.
2. Each party $\mathrm{R}_{\mathrm{r},i}$ in committee $\mathcal{C}_\mathrm{r}$ has $A_{\mathrm{r},i}$ as share with public encryption
$$C_{\mathrm{r},i} = \mathcal{E}.\mathrm{Enc}_{\mathrm{pk}_{\mathrm{r},i}}(A_{\mathrm{r},i})$$
3. Let $A_{i\to j}$ be share of $A_{\mathrm{r},i}$ that will be sent from $\mathrm{R}_{\mathrm{r},i}$ to $\mathrm{R}_{\mathrm{r}+1,j}$ encrypted as
$$C_{i\to j} = \mathcal{E}.\mathrm{Enc}_{\mathrm{pk}_{\mathrm{r}+1,j}}(A_{i\to j})$$

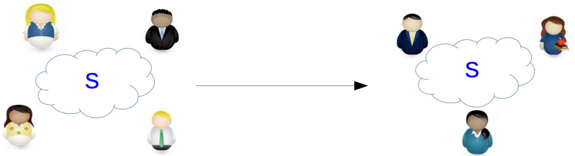


Resharing and Reconstruction

Resharing:

1. Transfer S from $\mathcal{C}_\mathsf{r}$ to $\mathcal{C}_{\mathsf{r}+1}$ where $|\mathcal{C}_k| = n_k$ with threshold t_k.
2. Each party $\mathsf{R}_{\mathsf{r},i}$ in committee $\mathcal{C}_\mathsf{r}$ has $A_{\mathsf{r},i}$ as share with public encryption
$$C_{\mathsf{r},i} = \mathcal{E}.\mathsf{Enc}_{\mathsf{pk}_{\mathsf{r},i}}(A_{\mathsf{r},i})$$
3. Let $A_{i\to j}$ be share of $A_{\mathsf{r},i}$ that will be sent from $\mathsf{R}_{\mathsf{r},i}$ to $\mathsf{R}_{\mathsf{r}+1,j}$ encrypted as
$$C_{i\to j} = \mathcal{E}.\mathsf{Enc}_{\mathsf{pk}_{\mathsf{r}+1,j}}(A_{i\to j})$$

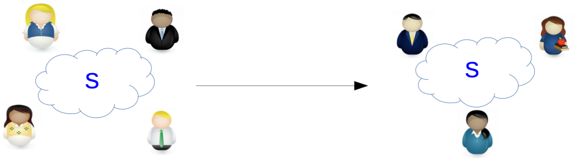
5. When a subset of $t_\mathsf{r} + 1$ parties have correctly reshared, each $\mathsf{R}_{\mathsf{r}+1,j}$ sets
$$A_{\mathsf{r}+1,j} = \sum_{\ell\in L_\mathsf{r}} \lambda_{\ell,L_\mathsf{r}} A_{\ell\to j} \quad C_{\mathsf{r}+1,j} = \sum_{\ell\in L_\mathsf{r}} \lambda_{\ell,L_\mathsf{r}} C_{\ell\to j}$$
.



Resharing and Reconstruction

Resharing:

1. Transfer S from $\mathcal{C}_\mathrm{r}$ to $\mathcal{C}_{\mathrm{r}+1}$ where $|\mathcal{C}_k| = n_k$ with threshold t_k.
2. Each party $\mathrm{R}_{\mathrm{r},i}$ in committee $\mathcal{C}_\mathrm{r}$ has $A_{\mathrm{r},i}$ as share with public encryption
$$C_{\mathrm{r},i} = \mathcal{E}.\mathrm{Enc}_{\mathrm{pk}_{\mathrm{r},i}}(A_{\mathrm{r},i})$$
3. Let $A_{i\to j}$ be share of $A_{\mathrm{r},i}$ that will be sent from $\mathrm{R}_{\mathrm{r},i}$ to $\mathrm{R}_{\mathrm{r}+1,j}$ encrypted as
$$C_{i\to j} = \mathcal{E}.\mathrm{Enc}_{\mathrm{pk}_{\mathrm{r}+1,j}}(A_{i\to j})$$
4. $\mathrm{R}_{\mathrm{r},i}$ **that** $C_{i\to j}$ **are encryptions of a correct sharing whose secret is plaintext of** $C_{\mathrm{r},i}$**.**
5. When a subset of $t_\mathrm{r}+1$ parties have correctly reshared, each $\mathrm{R}_{\mathrm{r}+1,j}$ sets
$$A_{\mathrm{r}+1,j} = \sum_{\ell\in L_\mathrm{r}} \lambda_{\ell,L_\mathrm{r}} A_{\ell\to j} \quad C_{\mathrm{r}+1,j} = \sum_{\ell\in L_\mathrm{r}} \lambda_{\ell,L_\mathrm{r}} C_{\ell\to j}$$
.



Resharing and Reconstruction

Reconstruction:

- Number of parties $n_1, \ldots n_{\text{last}}$ and thresholds $t_1, \ldots, t_{\text{last}}$ may differ from round to round.

Resharing and Reconstruction

Reconstruction:

- Number of parties $n_1, \ldots n_{\text{last}}$ and thresholds $t_1, \ldots, t_{\text{last}}$ may differ from round to round.
- Assuming proof of correct resharing, this implies proofs of correct reconstruction if $(n_{\text{last}} = 1, t_{\text{last}} = 0)$.

Resharing and Reconstruction

Reconstruction:

- Number of parties $n_1, \ldots n_{\text{last}}$ and thresholds $t_1, \ldots, t_{\text{last}}$ may differ from round to round.
- Assuming proof of correct resharing, this implies proofs of correct reconstruction if $(n_{\text{last}} = 1, t_{\text{last}} = 0)$.
- Applications include:
    - "Keeping secrets on a blockchain" [BGG+20].
    - EtF (far) future (carrying the TIBE key).
    - Distributed Randomness Generation (Beacons).

PVSS Constructions

HE-PVSS:

- Generic PVSS from a $\mathbb{Z}_p$-Linearly Homomorphic Encryption (LHE) scheme.
- Plaintext, randomness, ciphertext each have a $\mathbb{Z}_p$-vector space structure (e.g. groups of order p).
- $\mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_1; \rho_1) \boxplus_{\mathfrak{C}} \mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_2; \rho_2) = \mathcal{E}.\mathsf{Enc}_{\mathsf{pk}}(m_1 \boxplus_{\mathfrak{P}} m_2; \rho_1 \boxplus_{\mathfrak{R}} \rho_2)$
- Allows for simple "Schnorr-like" PoK of plaintext.
- And simple proof of correct $\mathbb{Z}_p$-Linear decryption (e.g. ElGamal)

DH-PVSS:

- We present a DL-based PVSS. First one (as far as we know) with **constant size overhead**.
- The dealer has an initial key-pair (pk_D, sk_D) to enable the "SCRAPE check".
- Secret $S = s \cdot G$ is in group $\mathbb{G} = \langle G \rangle$ of order p, where DDH is hard.
- Dealer publishes:
  - n encrypted shares: each 1 element in $\mathbb{G}$.
  - Correctness proof: $\underline{2\ \mathbb{Z}_p \text{ elements.}}$

DH-PVSS - Ideas 1

- Previously [CD20]: DL-based PVSS share-receivers have key pairs $(sk_i, PK_i = sk_i \cdot G)$.

DH-PVSS - Ideas 1

- Previously [CD20]: DL-based PVSS share-receivers have key pairs $(sk_i, PK_i = sk_i \cdot G)$.
- **New**: Dealer will also have a key pair $(sk_D, PK_D = sk_D \cdot G)$.

DH-PVSS - Ideas 1

- Previously [CD20]: DL-based PVSS share-receivers have key pairs $(sk_i, PK_i = sk_i \cdot G)$.
- **New**: Dealer will also have a key pair $(sk_D, PK_D = sk_D \cdot G)$.
- Shamir shares of $S = s \cdot G$ (dealer does not need to know s)

$$(A_i)_{i \in [n]} \leftarrow \text{GShamir.Share}(pp_{\text{Sh}}, S)$$

- A_i encrypted as $C_i = A_i + sk_D \cdot PK_i$ ($sk_D \cdot PK_i$ is shared DH key).

Algorithm 8 GShamir.Share(pp, S)

1: **Input:** $S \in \mathbb{G}$
2: $m(X) \xleftarrow{\$} \{m(X) \in \mathbb{Z}_p[X]_{\leq t} : m(\alpha_0) = 0\}$
3: $A_i = S + m(\alpha_i) \cdot G,\ i \in [n]$
4: **return** $(A_1, \ldots, A_n)$

DH-PVSS - Ideas 2

"SCRAPE Test" (Cascudo, David - ACNS17 [CD17]):

Theorem (SCRAPE dual-code test)

Let $1 \leq t < n$ be integers. Let p be a prime number with $p \geq n$. Let $\alpha_1, \ldots, \alpha_n$ be pairwise different points in $\mathbb{Z}_p$. Define the coefficients $v_i = \prod_{j \in [n] \setminus \{i\}} (\alpha_i - \alpha_j)^{-1}$. Let

$$C = \{(m(\alpha_1), \ldots, m(\alpha_n)) : \; m(X) \in \mathbb{Z}_p[X]_{\leq t}\}.$$

Then, for every vector $(\sigma_1, \ldots, \sigma_n)$ in $\mathbb{Z}_p^n$:

$$(\sigma_1, \ldots, \sigma_n) \in C \quad \Leftrightarrow \quad \sum_{i=1}^{n} v_i \cdot m^*(\alpha_i) \cdot \sigma_i = 0, \;\; \forall m^* \in \mathbb{Z}_p[X]_{\leq n-t-1}.$$

DH-PVSS - Ideas 2

"SCRAPE Test" (Cascudo, David - ACNS17 [CD17]):

- $\mathcal{C} = \{(m(1), \ldots, m(n)) : \deg(m) \leq t\}$ is a linear (Reed-Solomon) code space.

DH-PVSS - Ideas 2

"SCRAPE Test" (Cascudo, David - ACNS17 [CD17]):

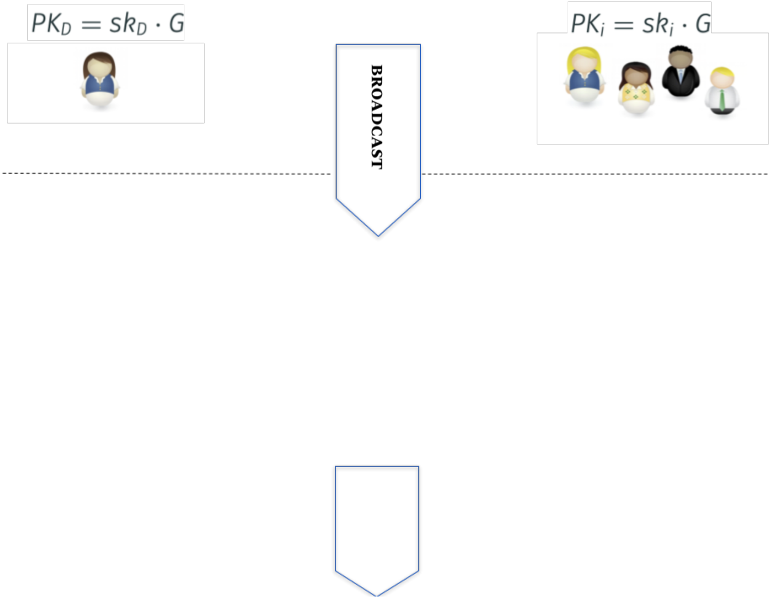
- $\mathcal{C} = \{(m(1), \ldots, m(n)) : \deg(m) \leq t\}$ is a linear (Reed-Solomon) code space.
- It has a dual code space: $\mathcal{D} = \{(m^*(1), \ldots, m^*(n)) : \deg(m^*) \leq n - t - 1\}$.
- Let $\mathbf{a} = (a_1, \ldots, a_n)$ in $(\mathbb{Z}_p)^n$. Sample $\mathbf{d} = (d_1, \ldots, d_n)$ from $\mathcal{D}$
  - If $\mathbf{a} \in \mathcal{C}$, then $\sum_{i=1}^n v_i \cdot d_i \cdot a_i = 0$.
  - If $\mathbf{a} \notin \mathcal{C}$, then $\sum_{i=1}^n v_i \cdot d_i \cdot a_i = 0$, with probability $1/p$.

Extends to group $\mathbb{G} = \langle G \rangle$ where $A_i = a_i \cdot G$:

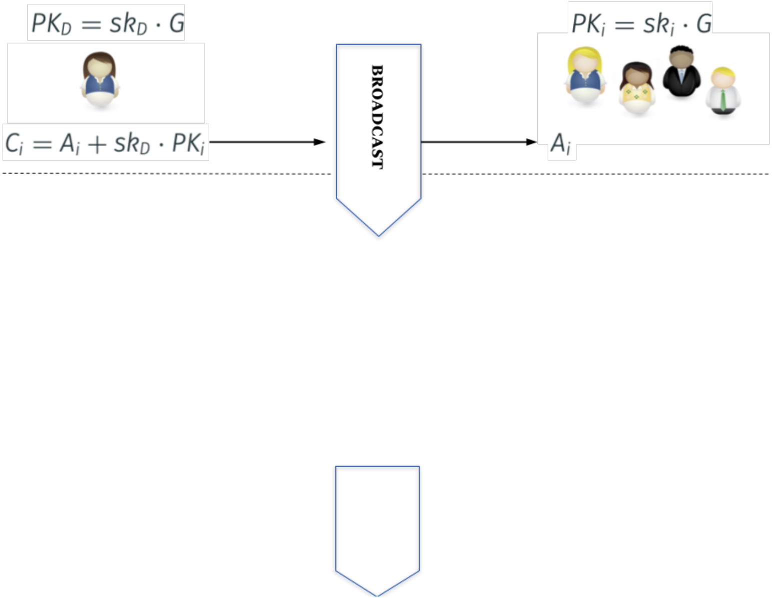
- Given $(A_1, \ldots, A_n)$ in $\mathbb{G}^n$. Sample $\mathbf{d} = (d_1, \ldots, d_n)$ from $\mathcal{D}$.

$$\sum_{i=1}^{n} v_i \cdot d_i \cdot A_i \stackrel{?}{=} O,$$

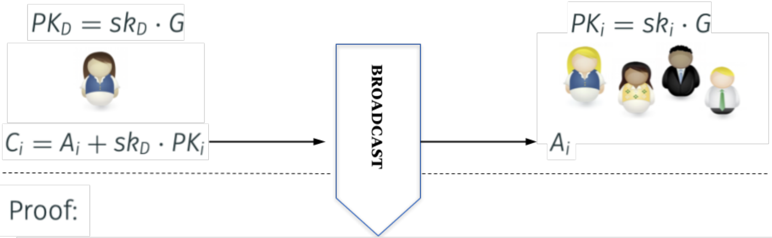
DH-PVSS - Ideas 2



DH-PVSS - Ideas 2



DH-PVSS - Ideas 2



Proof:

$(d_1, \ldots, d_n) \leftarrow H((C_i), (PK_i))$ for random oracle $H : \mathbb{G}^n \times \mathbb{G}^n \rightarrow \mathcal{D}$

$$\sum_{i=1}^{n} d_i \cdot C_i = \sum_{i=1}^{n} d_i \cdot A_i + \sum_{i=1}^{n} d_i \cdot sk_D \cdot PK_i = sk_D \sum_{i=1}^{n} d_i \cdot PK_i$$

DH-PVSS - Ideas 2

$PK_D = sk_D \cdot G$

$C_i = A_i + sk_D \cdot PK_i$ → BROADCAST → A_i

$PK_i = sk_i \cdot G$

Proof:

$(d_1, \ldots, d_n) \leftarrow H((C_i), (PK_i))$ for random oracle $H : \mathbb{G}^n \times \mathbb{G}^n \rightarrow \mathcal{D}$

$$\overbrace{\sum_{i=1}^{n} d_i \cdot C_i}^{V\,=} = \cancel{\sum_{i=1}^{n} d_i \cdot A_i} + \sum_{i=1}^{n} d_i \cdot sk_D \cdot PK_i = \overbrace{sk_D \sum_{i=1}^{n} d_i \cdot PK_i}^{sk_D \cdot U}$$

DH-PVSS - Ideas 2

$PK_D = sk_D \cdot G$

$C_i = A_i + sk_D \cdot PK_i \longrightarrow$ BROADCAST $\longrightarrow A_i$

$PK_i = sk_i \cdot G$

Proof:

$(d_1, \ldots, d_n) \leftarrow H((C_i), (PK_i))$ for random oracle $H : \mathbb{G}^n \times \mathbb{G}^n \rightarrow \mathcal{D}$

$$\underbrace{\sum_{i=1}^{n} d_i \cdot C_i}_{V =} = \cancel{\sum_{i=1}^{n} d_i \cdot A_i} + \sum_{i=1}^{n} d_i \cdot sk_D \cdot PK_i = sk_D \underbrace{\sum_{i=1}^{n} d_i \cdot PK_i}_{sk_D \cdot U}$$

$sk_D = \mathrm{DLOG}_G(PK_D) = \mathrm{DLOG}_U(V)$

$\pi_{\mathrm{PoK}} = \mathrm{DLEQ}(sk_D, G, PK_D, U, V)$

DH-PVSS - Ideas 2

$PK_D = sk_D \cdot G$

$C_i = A_i + sk_D \cdot PK_i$

BROADCAST

$PK_i = sk_i \cdot G$

A_i

Proof:

$(d_1, \dots, d_n) \leftarrow H((C_i), (PK_i))$ for random oracle $H : \mathbb{G}^n \times \mathbb{G}^n \to \mathcal{D}$

$$\overbrace{\sum_{i=1}^{n} d_i \cdot C_i}^{V\,=} = \cancel{\sum_{i=1}^{n} d_i \cdot A_i} + \sum_{i=1}^{n} d_i \cdot sk_D \cdot PK_i = sk_D \overbrace{\sum_{i=1}^{n} d_i \cdot PK_i}^{sk_D \cdot U}$$

$sk_D = \mathrm{DLOG}_G(PK_D) = \mathrm{DLOG}_U(V)$

$\pi_{\mathrm{PoK}} = \mathrm{DLEQ}(sk_D, G, PK_D, U, V)$

$PK_D = sk_D \cdot G$

$C_i = A_i + sk_D \cdot PK_i$

BROADCAST

$PK_i = sk_i \cdot G$

A_i

Proof:

$(d_1, \ldots, d_n) \leftarrow H((C_i), (PK_i))$ for random oracle $H : \mathbb{G}^n \times \mathbb{G}^n \to \mathcal{D}$

$$\overbrace{\sum_{i=1}^{n} d_i \cdot C_i}^{V\,=} = \cancel{\sum_{i=1}^{n} d_i \cdot A_i} + \sum_{i=1}^{n} d_i \cdot sk_D \cdot PK_i = \overbrace{sk_D \sum_{i=1}^{n} d_i \cdot PK_i}^{sk_D \cdot U}$$

$sk_D = \mathrm{DLOG}_G(PK_D) = \mathrm{DLOG}_U(V)$

$\pi_{\mathrm{PoK}} = \mathrm{DLEQ}(sk_D, G, PK_D, U, V)$

$\mathrm{Verify}(pp, \mathrm{pk}_D, \{(\mathrm{pk}_i, C_i) : i \in [n]\}, \pi_{\mathrm{PoK}})$

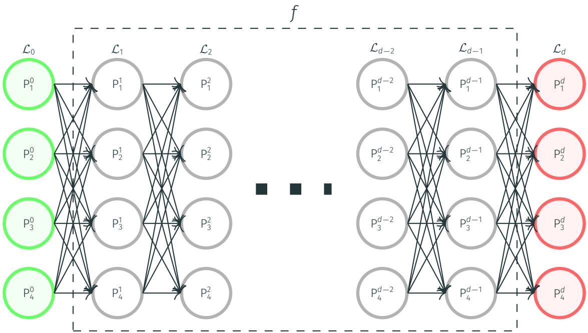
Layered MPC [DDG+23]

Layered MPC

An (n, t, d)-**layered protocol** has the following properties:

Parties. $N = n(d+1)$ parties partitioned into $d+1$ layers $\mathcal{L}_i$, $0 \leq i \leq d$, where $|\mathcal{L}_i| = n$.

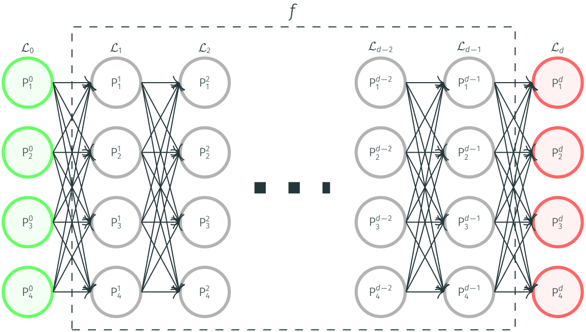
Interaction. d synchronous rounds where parties in $\mathcal{L}_{i-1}$ may send messages to parties in $\mathcal{L}_i$ over secure channels and broadcast.



Layered MPC

Functionalities. We consider functionalities f that take inputs from input clients and deliver outputs to output clients.

Adversaries. We consider active, rushing, adaptive adversaries who may corrupt any number of input/output clients, and t parties in layers $\mathcal{L}_i$, $0 < i < d$.



Layered MPC

A note on Layered Broadcast

- The model of layered MPC assumes layer-to-layer broadcast.
- Deterministic Broadcast is impossible in the layered setting.
- Derived from the result of [Gar94] on reaching agreement in the mobile setting.

Lemma 2

Deterministic Broadcast is possible iff $t = 0$.

Basic Primitives

Future Messaging

Future Messaging functionality f_{FM}

PUBLIC PARAMETERS: Sender $\mathsf{S} \in \mathcal{L}_0$, receiver $\mathsf{R} \in \mathcal{L}_d$ for $d > 0$ and message domain M.

SECRET INPUTS: S has input $m \in M$.

f_{FM} receives m from S, and delivers m to R.

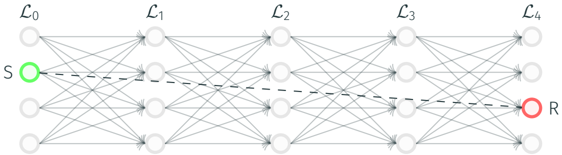
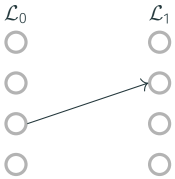


Figure 1: Π_{FM} from S of m to R

Future Messaging

Π_{FM} from $\mathcal{L}_0$ to $\mathcal{L}_1$:

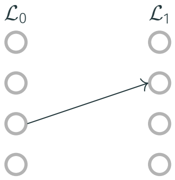
Use the secure point-to-point channels from layer to the next layer.



Future Messaging

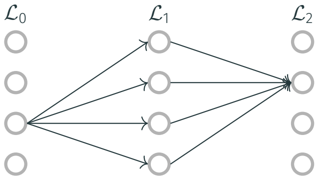
Π_{FM} from $\mathcal{L}_0$ to $\mathcal{L}_1$:

Use the secure point-to-point channels from layer to the next layer.



Π_{FM} from $\mathcal{L}_0$ to $\mathcal{L}_2$:

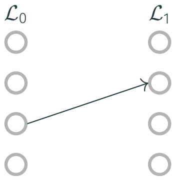
1. S does $\mathsf{Sh}(m) = (s_1, \ldots, s_n)$ and sends s_j to P_j^1.
2. P_j^1 forwards s_j to R and R obtains $\hat{m} = \mathsf{Rec}(\hat{s}_1, \ldots, \hat{s}_n)$



Future Messaging

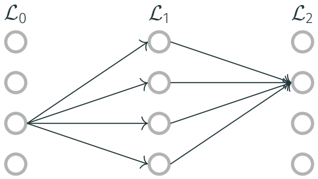
Π_{FM} from $\mathcal{L}_0$ to $\mathcal{L}_1$:

Use the secure point-to-point channels from layer to the next layer.



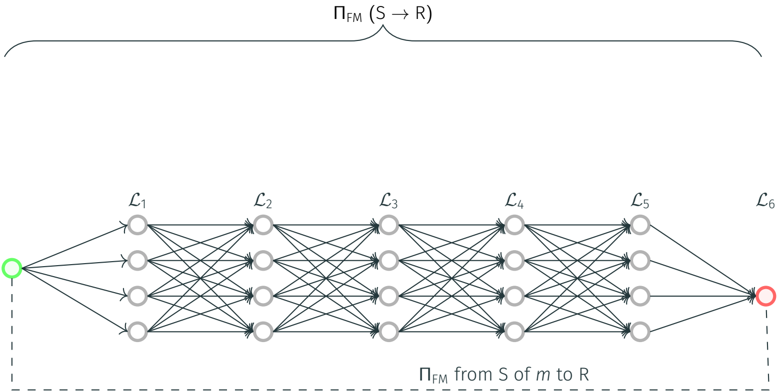
Π_{FM} from $\mathcal{L}_0$ to $\mathcal{L}_2$:

1. S does $\mathrm{Sh}(m) = (s_1, \ldots, s_n)$ and sends s_j to P_j^1.
2. P_j^1 forwards s_j to R and R obtains $\hat{m} = \mathrm{Rec}(\hat{s}_1, \ldots, \hat{s}_n)$

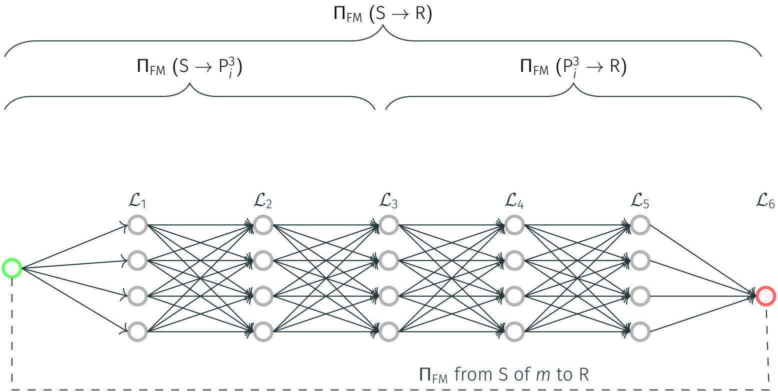


(Equivalent to perfect malicious 1-way SMT [DDWY93])

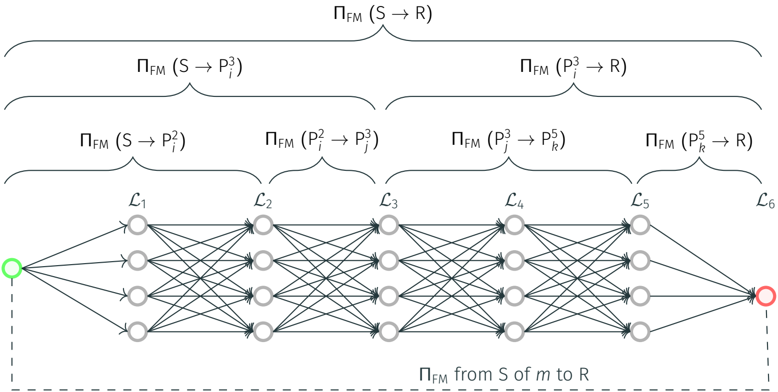
Future Messaging



Future Messaging



Future Messaging

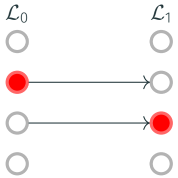


Future Messaging (and Rushing)

Dishonest Sender and problems with rushing

PARALLEL INVOCATIONS f^n_{FM}:

- When invoking multiple f_{FM} in parallel, the adversary can cause a **correlation attack**.
- Model the parallel functionality as **corruption-aware**.

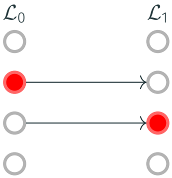


Future Messaging (and Rushing)

Dishonest Sender and problems with rushing

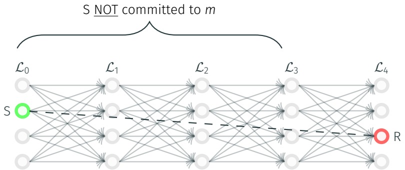
PARALLEL INVOCATIONS f^n_{FM}:

- When invoking multiple f_{FM} in parallel, the adversary can cause a **correlation attack**.
- Model the parallel functionality as **corruption-aware**.



NON-COMMITTING PRIMITIVE:

- The adversary can change the message m to a message of its choosing m' in f_{FM} until the last round.
- Where YOSO assumes ideal **committing** communication to future rounds.



(Conditional) Future Broadcast

- FUTURE BROADCAST:
  Invoke f_{FM} where parties in $\mathcal{L}_{d-1}$ are instructed to broadcast their shares instead of sending to a recipient R.
- CONDITIONAL DISCLOSURE:
  Conditioned on some event E, honest parties in $\mathcal{L}_{d-1}$ reveal their shares.

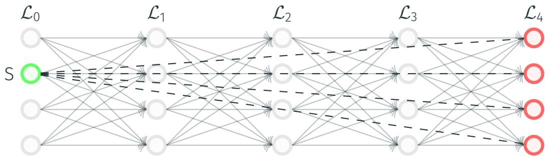


Figure 2: Future Broadcast from S of m to $\mathcal{L}_4$

Future Messaging

Summary of Future Messaging:

Complexity Assuming a linear secret sharing scheme, Π_{FM} is a recursive protocol realizing f_{FM} with communication complexity $O(n^{\lceil \log d \rceil} \log |M|)$.

Security Honest sender reduces to an instance of SMT Dishonest sender is challenging with rushing.
Especially, when composing protocols.

Extension Future Messaging can be extended to (Conditional) Future Broadcast.

Towards Layered MPC

Layered CNF-VSS Protocol

Design Approach

CNF-VSS of [GIKR01]

↓

Weak Future Multicast $\Pi_{\text{weak-FMcast}}$

↓

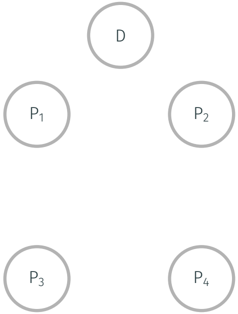
Future Multicast Π_{FMcast}

↓

Verifiable Secret Sharing Π_{VSS}

4-round perfect CNF VSS

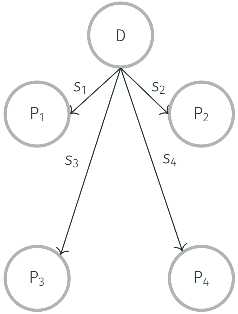
D (dealer) holds a secret $s \in \mathbb{F}$ and obtains $\mathsf{Sh}_{\mathsf{CNF}}(s) = (s_1, \ldots, s_n)$.



4-round perfect CNF VSS

D (dealer) holds a secret $s \in \mathbb{F}$ and obtains $\mathsf{Sh}_{\mathsf{CNF}}(s) = (s_1, \ldots, s_n)$.

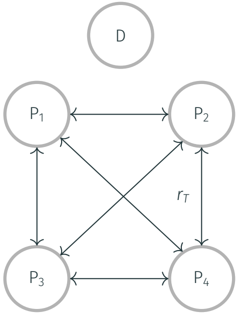
1. D sends $s_j = (r_T)_{T \ni j}$ to P_j.



4-round perfect CNF VSS

D (dealer) holds a secret $s \in \mathbb{F}$ and obtains $\mathrm{Sh}_{\mathrm{CNF}}(s) = (s_1, \ldots, s_n)$.

1. D sends $s_j = (r_T)_{T \ni j}$ to P_j.
2. Each pair $(\mathrm{P}_j, \mathrm{P}_{j'})$ exchange share r_T (if $j, j' \in T$).

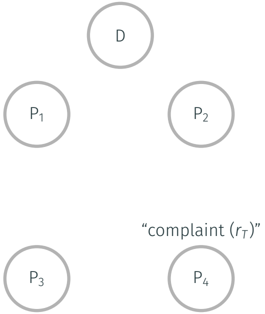


Techniques from [GIKR01]

4-round perfect CNF VSS

D (dealer) holds a secret $s \in \mathbb{F}$ and obtains $\mathsf{Sh}_{\mathsf{CNF}}(s) = (s_1, \ldots, s_n)$.

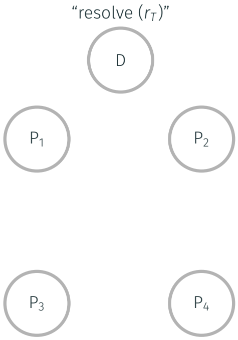
1. D sends $s_j = (r_T)_{T \ni j}$ to P_j.
2. Each pair $(\mathsf{P}_j, \mathsf{P}_{j'})$ exchange share r_T (if $j, j' \in T$).
3. If disagreement, involved parties broadcast "complaint (r_T)".



4-round perfect CNF VSS

D (dealer) holds a secret $s \in \mathbb{F}$ and obtains $\mathsf{Sh}_{\mathsf{CNF}}(s) = (s_1, \ldots, s_n)$.

1. D sends $s_j = (r_T)_{T \ni j}$ to P_j.
2. Each pair $(\mathsf{P}_j, \mathsf{P}_{j'})$ exchange share r_T (if $j, j' \in T$).
3. If disagreement, involved parties broadcast "complaint (r_T)".
4. D then broadcasts "resolve (r_T)", if any complaints received from P_j or $\mathsf{P}_{j'}$.



Techniques from [GIKR01]

Challenges with layered [GIKR01]

- Dealer speaks more than once (round 1 and round 4).

Techniques from [GIKR01]

Challenges with layered [GIKR01]

- Dealer speaks more than once (round 1 and round 4).

  Solution:
  Emulate the dealer using Conditional Future Broadcast.

Techniques from [GIKR01]

Challenges with layered [GIKR01]

- Dealer speaks more than once (round 1 and round 4).

  Solution:
  Emulate the dealer using Conditional Future Broadcast.

- P_j and $P_{j'}$ exchange additive shares.

Techniques from [GIKR01]

Challenges with layered [GIKR01]

- Dealer speaks more than once (round 1 and round 4).

  Solution:
  Emulate the dealer using Conditional Future Broadcast.

- P_j and $P_{j'}$ exchange additive shares.

  Solution:
  Invoke a Distributed Equality Check with Π_{add} for each pair (j, j').

Future Multicast

Future Multicast functionality f_{FMcast}

PUBLIC PARAMETERS: Sender $\mathsf{S} \in \mathcal{L}_0$, receiving set of parties $R \subseteq \mathcal{L}_d, d \geq 5$, message domain M.

SECRET INPUTS: S has input $m \in M$.

f_{FMcast} receives m from S, and delivers m to all parties in R.

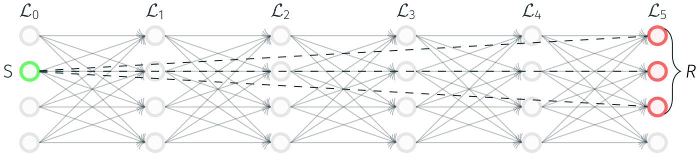
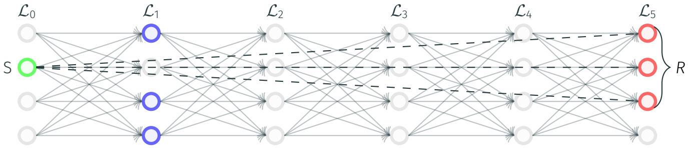


Figure 3: Π_{FMcast} from $\mathsf{S} \in \mathcal{L}_0$ of m to $R \subseteq \mathcal{L}_5$

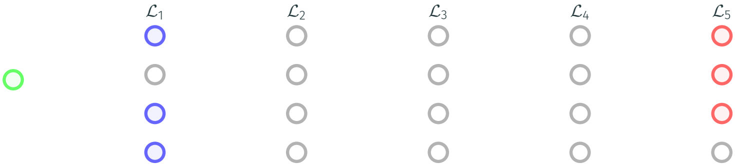
Sketch of Π_{FMcast}

1. S samples additive shares $\{r_T\}_{T\in\mathcal{T}}$ of m.
2. S sends each r_T to $R \subseteq \mathcal{L}_5$ using $\Pi_{\text{weak-FMcast}}$.
   Using a different set of intermediaries $U_T \subset \mathcal{L}_1$ where $|U_T| = n - t$.
3. Parties in $R \subseteq \mathcal{L}_5$ do $\hat{m} = \sum_{T\in\mathcal{T}} \hat{r}_T$.



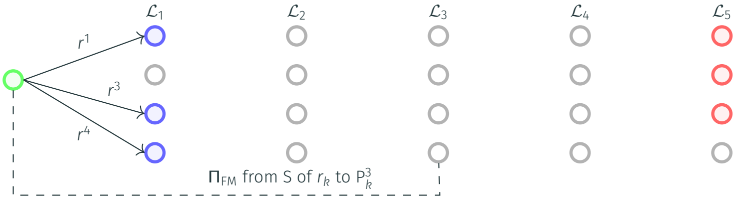
Weak Future Multicast

$\Pi_{\text{weak-FMcast}}$ of $r = r_T$ from $\mathsf{S} \in \mathcal{L}_0$ to R using U_T as intermediaries.



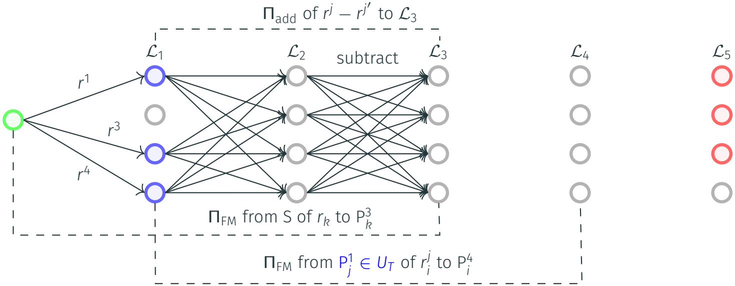
Weak Future Multicast

$\Pi_{\text{weak-FMcast}}$ of $r = r_T$ from $\mathsf{S} \in \mathcal{L}_0$ to R using U_T as intermediaries.



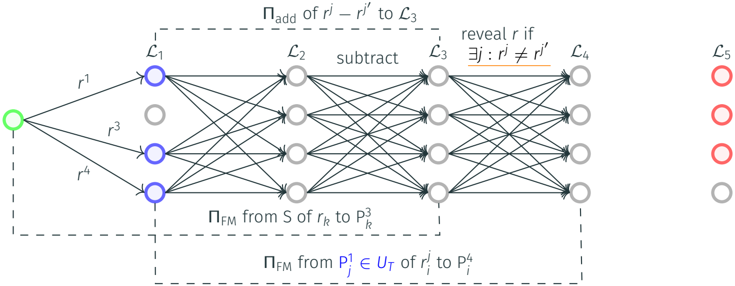
Weak Future Multicast

$\Pi_{\text{weak-FMcast}}$ of $r = r_T$ from $S \in \mathcal{L}_0$ to R using U_T as intermediaries.



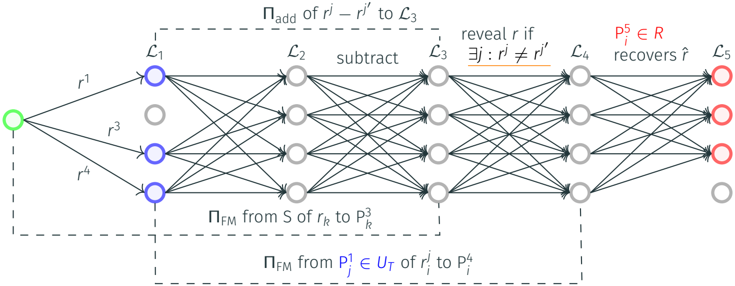
Weak Future Multicast

$\Pi_{\text{weak-FMcast}}$ of $r = r_T$ from $\mathsf{S} \in \mathcal{L}_0$ to R using U_T as intermediaries.



Weak Future Multicast

$\Pi_{\text{weak-FMcast}}$ of $r = r_T$ from $\mathsf{S} \in \mathcal{L}_0$ to R using U_T as intermediaries.



Layered VSS

An $(n, t, 5)$-layered protocol Π_{VSS} realizing f_{VSS} where $t < n/3$.

From $\Pi_{\text{weak-FMcast}}$ to Π_{FMcast}:

From Π_{FMcast} to Π_{VSS}:

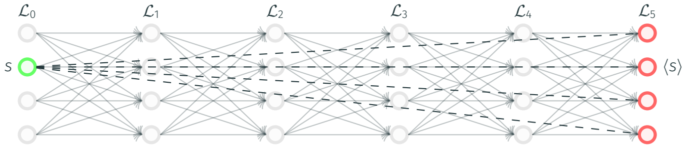


Figure 4: Π_{VSS} from $\mathsf{S} \in \mathcal{L}_0$ of m to $\mathcal{L}_5$

Layered VSS

An $(n, t, 5)$-layered protocol Π_{VSS} realizing f_{VSS} where $t < n/3$.

From $\Pi_{\text{weak-FMcast}}$ to Π_{FMcast}:

- Each additive share r_T is transferred to R using U_T.

From Π_{FMcast} to Π_{VSS}:

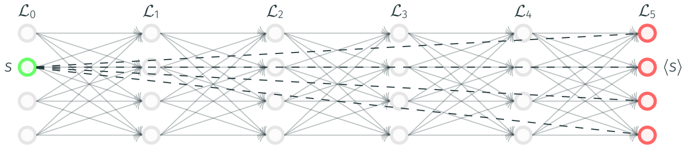


Figure 4: Π_{VSS} from $\mathsf{S} \in \mathcal{L}_0$ of m to $\mathcal{L}_5$

Layered VSS

An $(n, t, 5)$-layered protocol Π_{VSS} realizing f_{VSS} where $t < n/3$.

From $\Pi_{\text{weak-FMcast}}$ to Π_{FMcast}:

- Each additive share r_T is transferred to R using U_T.
- Since at least one set (U_T) is comprised of only honest parties the message $m = \sum_{T \in \mathcal{T}} r_T$ remains secure if S and R are honest.

From Π_{FMcast} to Π_{VSS}:

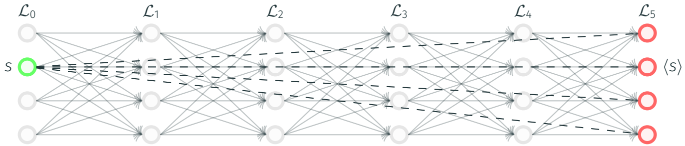


Figure 4: Π_{VSS} from $\mathsf{S} \in \mathcal{L}_0$ of m to $\mathcal{L}_5$

Layered VSS

An $(n, t, 5)$-layered protocol Π_{VSS} realizing f_{VSS} where $t < n/3$.

From $\Pi_{\text{weak-FMcast}}$ to Π_{FMcast}:

- Each additive share r_T is transferred to R using U_T.
- Since at least one set (U_T) is comprised of only honest parties the message $m = \sum_{T \in \mathcal{T}} r_T$ remains secure if S and R are honest.

From Π_{FMcast} to Π_{VSS}:

- S samples $\{r_T\}_{T \in \mathcal{T}}$ as additive secret sharing of secret s.

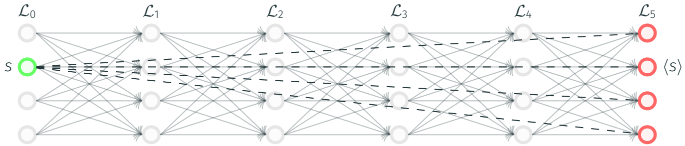


Figure 4: Π_{VSS} from $\mathsf{S} \in \mathcal{L}_0$ of m to $\mathcal{L}_5$

Layered VSS

An $(n, t, 5)$-layered protocol Π_{VSS} realizing f_{VSS} where $t < n/3$.

From $\Pi_{\text{weak-FMcast}}$ to Π_{FMcast}:

- Each additive share r_T is transferred to R using U_T.
- Since at least one set (U_T) is comprised of only honest parties the message $m = \sum_{T \in \mathcal{T}} r_T$ remains secure if S and R are honest.

From Π_{FMcast} to Π_{VSS}:

- S samples $\{r_T\}_{T \in \mathcal{T}}$ as additive secret sharing of secret s.
- For each $T \in \mathcal{T}$, execute Π_{FMcast} with S as sender with input r_T and $\{\text{P}_i^5 : i \in T\}$ as receivers.

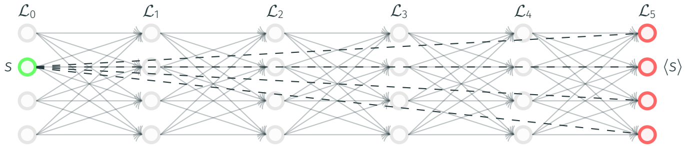


Figure 4: Π_{VSS} from $\text{S} \in \mathcal{L}_0$ of m to $\mathcal{L}_5$

Results

CNF-based Layered MPC [GIKR01, Mau06]

Theorem 1: CNF-Based Layered MPC

Let f be an n-party functionality computed by a **layered arithmetic circuit** C over a finite ring, with D **layers** and M gates. Then, for any $t < n/3$, there is an $(n, t, O(D))$**-layered MPC protocol** for f. The communication consists of $2^{O(n)} \cdot M$ ring elements.

CNF-based Layered MPC [GIKR01, Mau06]

Theorem 1: CNF-Based Layered MPC

Let f be an n-party functionality computed by a **layered arithmetic circuit** C over a finite ring, with D **layers** and M gates. Then, for any $t < n/3$, there is an $(n, t, O(D))$**-layered MPC protocol** for f. The communication consists of $2^{O(n)} \cdot M$ ring elements.

Corollary 1: Secure Maximally Proactive MPC

Let f be an n-party functionality computed by a layered arithmetic circuit C over a finite ring, **with** D **layers**. Then, for $t < n/3$, there is a **maximally proactive MPC protocol** computing f in $r = O(D)$ **rounds**.

CNF-based Layered MPC [GIKR01, Mau06]

Theorem 1: CNF-Based Layered MPC

Let f be an n-party functionality computed by a **layered arithmetic circuit** C over a finite ring, with D **layers** and M gates. Then, for any $t < n/3$, there is an $(n, t, O(D))$**-layered MPC protocol** for f. The communication consists of $2^{O(n)} \cdot M$ ring elements.

Corollary 1: Secure Maximally Proactive MPC

Let f be an n-party functionality computed by a layered arithmetic circuit C over a finite ring, **with** D **layers**. Then, for $t < n/3$, there is a **maximally proactive MPC protocol** computing f in $r = O(D)$ **rounds**.

- May be concretely efficient for small n.
- Use techniques from [CDI05] to amortize the communication overhead by sending k-bit seeds and let the receivers generate most shares locally.
- This technique makes use of black-box access to PRG (computational security).

Efficient Layered MPC [BGW88]

Theorem 2: Efficient Layered MPC

Let f be an n-party functionality computed by a **layered arithmetic circuit** C over a finite field, with D **layers** and M gates. Then, for any $t < n/3$, there is an $(n, t, O(D))$**-layered MPC protocol** for f. The communication consists of $O(n^9) \cdot M$ field elements.

Efficient Layered MPC [BGW88]

Theorem 2: Efficient Layered MPC

Let f be an n-party functionality computed by a **layered arithmetic circuit** C over a finite field, with D **layers** and M gates. Then, for any $t < n/3$, there is an $(n, t, O(D))$**-layered MPC protocol** for f. The communication consists of $O(n^9) \cdot M$ field elements.

Corollary 2: (Efficient) Secure Maximally Proactive MPC

Let f be an n-party functionality computed by a layered arithmetic circuit C over a finite field, **with** D **layers**. Then, for $t < n/3$, there is an efficient **maximally proactive MPC protocol** computing f in $r = O(D)$ **rounds**.

- Extending the techniques for Distributed Equality Check and Conditional Future Broadcast to the [BGW88]-setting.

Results

f	Reference	Level	Security	Comm.	Threshold
FM	This work	perfect	full	poly(n)	$t < n/3$
VSS	[BGG+20]	comp.	full	poly(n)	$t < n/4^*$
	This work	perfect	full	$2^{O(n)}$	$t < n/3$
	This work (Sec. 5)	perfect	full	poly(n)	$t < n/3$
MPC	[GHK+21] (YOSO)	statistical	full +setup†	poly(n)	$t < n/2^*$
	[CGG+21] (Fluid)	statistical	abort	poly(n)	$t < n/2$
	[OY91]	perfect	full	poly(n)	$t < n/d$
	This work	perfect	full	$2^{O(n)}$	$t < n/3$
	This work (Sec. 5)	perfect	full	poly(n)	$t < n/3$
	This work (Sec. 6)	comp.	full	poly(n)	$t < n/2$

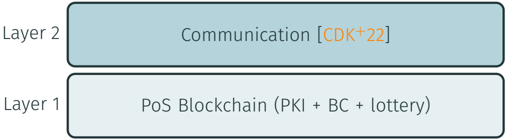
Table 1: Protocols realizing primitives in the most extreme proactive settings. (*protocol security relies on the adversary only doing probabilistic corruption, †assumes access to ideal target-anonymous channels for future messaging)

Conclusion

Conclusion

Layer 1 | PoS Blockchain (PKI + BC + lottery)

Ephemeral Committees Model



Ephemeral Committees Model

Layer 2 (Encryption to the Future):

- Formalize and define Encryption to the Future
- Construct EtF (near) using only OT+GC (w/o auxiliary commitees)
- Construct EtF (far) using TIBE w/ comm. independent of *M*.

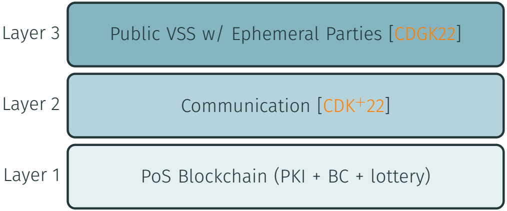
Conclusion

Layer 3 (YOLO-YOSO):

- Mixnet-based EtF (near) using standard PKE
- Propose a generic HE-PVSS and a extremely efficient DH-PVSS.
- Both HE-PVSS and DH-PVSS extendable to proactive resharing (YOSO).

Layer 2 (Encryption to the Future):

- Formalize and define Encryption to the Future
- Construct EtF (near) using only OT+GC (w/o auxiliary commitees)
- Construct EtF (far) using TIBE w/ comm. independent of M.



Ephemeral Committees Model

Conclusion

Layer 4 (Layered MPC):

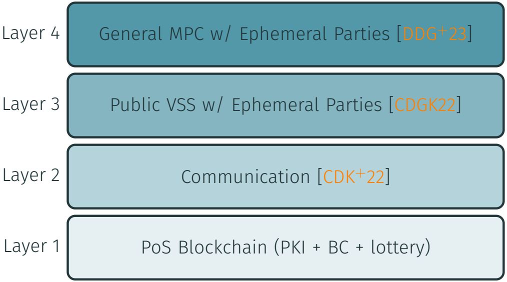
- MPC w/ restricted interaction.
- Prove feasibility of general perfect MPC with $n/3$.
- Show implications to classic proactive MPC and newer YOSO.

Layer 3 (YOLO-YOSO):

- Mixnet-based EtF (near) using standard PKE
- Propose a generic HE-PVSS and a extremely efficient DH-PVSS.
- Both HE-PVSS and DH-PVSS extendable to proactive resharing (YOSO).

Layer 2 (Encryption to the Future):

- Formalize and define Encryption to the Future
- Construct EtF (near) using only OT+GC (w/o auxiliary commitees)
- Construct EtF (far) using TIBE w/ comm. independent of M.



Ephemeral Committees Model

Thank You!

Link to Eprints:
https://ia.cr/2021/1423
https://ia.cr/2022/242
https://ia.cr/2023/330

Link to Thesis:
https://akonring.github.io/thesis.pdf

J. F. Almansa, I. Damgård, and J. B. Nielsen.
Simplified threshold RSA with adaptive and proactive security.
In EUROCRYPT 2006, LNCS 4004, pages 593–611. Springer, Heidelberg, May / June 2006.

J. Baron, K. El Defrawy, J. Lampkins, and R. Ostrovsky.
How to withstand mobile virus attacks, revisited.
In 33rd ACM PODC, pages 293–302. ACM, July 2014.

J. Baron, K. El Defrawy, J. Lampkins, and R. Ostrovsky.
Communication-optimal proactive secret sharing for dynamic groups.
In ACNS 15, LNCS 9092, pages 23–41. Springer, Heidelberg, June 2015.

F. Benhamouda, C. Gentry, S. Gorbunov, S. Halevi, H. Krawczyk, C. Lin, T. Rabin, and L. Reyzin.
Can a public blockchain keep a secret?
In TCC 2020, Part I, LNCS 12550, pages 260–290. Springer, Heidelberg, November 2020.

M. BenOr, S. Goldwasser, and A. Wigderson.
Completeness theorems for non-cryptographic fault-tolerant distributed computation.
In Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali, pages 351–371. 1988.

I. Cascudo and B. David.
SCRAPE: Scalable randomness attested by public entities.
In ACNS 17, LNCS 10355, pages 537–556. Springer, Heidelberg, July 2017.

I. Cascudo and B. David.
Albatross: publicly attestable batched randomness based on secret sharing.
In Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part III 26, pages 311–341. Springer, 2020.

I. Cascudo, B. David, L. Garms, and A. Konring.
YOLO YOSO: Fast and simple encryption and secret sharing in the YOSO model.
In ASIACRYPT 2022, Part I, LNCS 13791, pages 651–680. Springer, Heidelberg, December 2022.

R. Cramer, I. Damgård, and Y. Ishai.
Share conversion, pseudorandom secret-sharing and applications to secure computation.
In Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings, Lecture Notes in Computer Science 3378, pages 342–362. Springer, 2005.

M. Campanelli, B. David, H. Khoshakhlagh, A. Konring, and J. B. Nielsen.
Encryption to the future - A paradigm for sending secret messages to future (anonymous) committees.
In ASIACRYPT 2022, Part III, LNCS 13793, pages 151–180. Springer, Heidelberg, December 2022.

A. R. Choudhuri, A. Goel, M. Green, A. Jain, and G. Kaptchuk.
Fluid MPC: Secure multiparty computation with dynamic participants.
In CRYPTO 2021, Part II, LNCS 12826, pages 94–123, Virtual Event, August 2021. Springer, Heidelberg.

R. Canetti and A. Herzberg.
Maintaining security in the presence of transient faults.
In Advances in Cryptology—CRYPTO'94: 14th Annual International Cryptology Conference Santa Barbara, California, USA August 21–25, 1994 Proceedings, pages 425–438. Springer, 2001.

C. Cachin, K. Kursawe, A. Lysyanskaya, and R. Strobl.
Asynchronous verifiable secret sharing and proactive cryptosystems.
In Proceedings of the 9th ACM Conference on Computer and Communications Security, pages 88–97, 2002.

B. David, G. Deligios, A. Goel, Y. Ishai, A. Konring, E. Kushilevitz, C.-D. Liu-Zhang, and V. Narayanan.
Perfect MPC over layered graphs.
In CRYPTO 2023, Part I, LNCS 14081, pages 360–392. Springer, Heidelberg, August 2023.

D. Dolev, C. Dwork, O. Waarts, and M. Yung.
Perfectly secure message transmission.
Journal of the ACM (JACM), 40(1):17–47, 1993.

B. David, P. Gazi, A. Kiayias, and A. Russell.
Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain.
In EUROCRYPT 2018, Part II, LNCS 10821, pages 66–98. Springer, Heidelberg, April / May 2018.

Y. Desmedt and S. Jajodia.
Redistributing secret shares to new access structures and its applications.
Technical report, Citeseer, 1997.

K. Eldefrawy, T. Lepoint, and A. Leroux.
Communication-efficient proactive secret sharing for dynamic groups with dishonest majorities.
In ACNS 20, Part I, LNCS 12146, pages 3–23. Springer, Heidelberg, October 2020.

J. A. Garay.
Reaching (and maintaining) agreement in the presence of mobile faults.
In International Workshop on Distributed Algorithms, pages 253–264. Springer, 1994.

S. Garg, C. Gentry, A. Sahai, and B. Waters.
Witness encryption and its applications.
In 45th ACM STOC, pages 467–476. ACM Press, June 2013.

C. Gentry, S. Halevi, H. Krawczyk, B. Magri, J. B. Nielsen, T. Rabin, and S. Yakoubov.
YOSO: You only speak once - secure MPC with stateless ephemeral roles.
In CRYPTO 2021, Part II, LNCS 12826, pages 64–93, Virtual Event, August 2021. Springer, Heidelberg.

Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich.
Algorand: Scaling byzantine agreements for cryptocurrencies.
In Proceedings of the 26th Symposium on Operating Systems Principles, pages 51–68, 2017.

C. Gentry, S. Halevi, B. Magri, J. B. Nielsen, and S. Yakoubov.
Random-index PIR and applications.
In TCC 2021, Part III, LNCS 13044, pages 32–61. Springer, Heidelberg, November 2021.

R. Gennaro, Y. Ishai, E. Kushilevitz, and T. Rabin.
The round complexity of verifiable secret sharing and secure multicast.
In Proceedings of the thirty-third annual ACM symposium on Theory of computing, pages 580–589, 2001.

V. Goyal, A. Kothapalli, E. Masserova, B. Parno, and Y. Song.
Storing and retrieving secrets on a blockchain.
Cryptology ePrint Archive, Report 2020/504, 2020.
https://eprint.iacr.org/2020/504.

A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung.
Proactive secret sharing or: How to cope with perpetual leakage.
In CRYPTO'95, LNCS 963, pages 339–352. Springer, Heidelberg, August 1995.

U. Maurer.
Secure multi-party computation made simple.
Discrete Applied Mathematics, 154(2):370–381, 2006.

S. K. D. Maram, F. Zhang, L. Wang, A. Low, Y. Zhang, A. Juels, and D. Song.
CHURP: Dynamic-committee proactive secret sharing.
In ACM CCS 2019, pages 2369–2386. ACM Press, November 2019.

R. Ostrovsky and M. Yung.
How to withstand mobile virus attacks (extended abstract).
In 10th ACM PODC, pages 51–59. ACM, August 1991.

T. Rabin and M. BenOr.
Verifiable secret sharing and multiparty protocols with honest majority.
In Proceedings of the twenty-first annual ACM symposium on Theory of computing, pages 73–85, 1989.

B. Schoenmakers.
A simple publicly verifiable secret sharing scheme and its application to electronic.
In CRYPTO'99, LNCS 1666, pages 148–164. Springer, Heidelberg, August 1999.

T. M. Wong, C. Wang, and J. M. Wing.
Verifiable secret redistribution for archive systems.
In First International IEEE Security in Storage Workshop, 2002. Proceedings., pages 94–105. IEEE, 2002.