OptRand: Optmistically Responsive Reconfigurable Distributed Randomness


Adithya Bhat*


Nibesh Shrestha*


Aniket Kate


Kartik Nayak

PURDUE UNIVERSITY

RIT

PURDUE UNIVERSITY

Duke UNIVERSITY

*Equal contribution

Motivation

Random Beacon

Generates random numbers at regular intervals

- 1ffa108e7cfcd9fe125c
- 06485727a9a47b37401a
- afd090a44b761903d1fe

Applications

- Random selection: lotteries, shuffled decks
- Randomized consensus protocols: VABA[AMS'19] , HoneyBadger[MXCSS'16]
- Blockchain-sharding[ASBHD'17]
- Anonymous communication[GRPS'03]
- E-voting and many more…

Random Beacon: Key Properties

Bias Resistance

No entity can influence a future random beacon away from uniform

Unpredictable

No entity can distinguish the beacon output from a random value

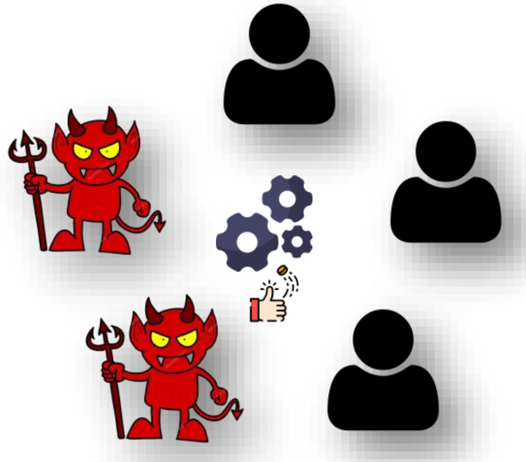
Byzantine Fault-tolerant Randomness Beacon

Generate *bias-resistant* and *unpredictable* random beacons

- despite t **Byzantine failures** out of n nodes

Additional Properties:

- **Optimal resilience:** tolerates $t < n/2$ Byzantine faults assuming synchrony
- **Low communication complexity**
- **Low computational overhead**
- **Low latency**
- **Reconfiguration-friendly:** Replace participating nodes without additional communication overhead



Prior Work

	Resilience	Communication		Unpredictability	Reusable setup	Assumption	Latency
		Best	Worst				
Drand	$t < \frac{n}{2}$	$O(n^2)$		1	✗	DKG	Low
Dfinity[HMW'18]	$t < \frac{n}{2}$	$O(n^2)$	$O(n^3)$	1	✗	DKG	Low
RandRunner[SJHSW'21]	$t < \frac{n}{2}$	$O(n^2)$		$t+1$	✓	VDF	High
BRandPiper[BSLKN'21]	$t < \frac{n}{2}$	$O(n^2)$	$O(n^3)$	1	✓	q-SDH	High

Can we design random beacon protocols with all desired properties?

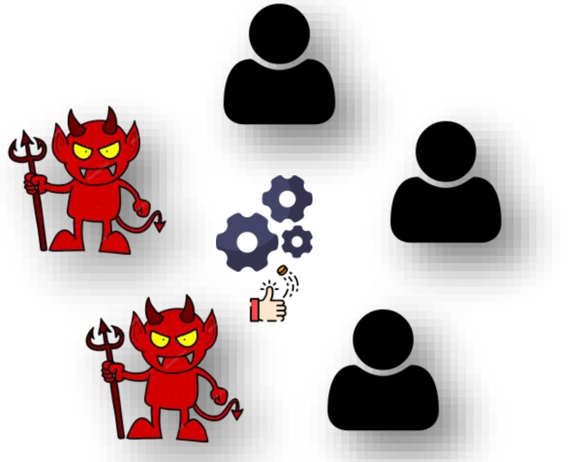
Prior Work

	Resilience	Communication		Unpredictability	Reusable setup	Assumption	Latency
		Best	Worst				
Drand	$t < \frac{n}{2}$	$O(n^2)$		1	✗	DKG	Low
Dfinity[HMW'18]	$t < \frac{n}{2}$	$O(n^2)$	$O(n^3)$	1	✗	DKG	Low
RandRunner[SJHSW'21]	$t < \frac{n}{2}$	$O(n^2)$		$t+1$	✓	VDF	High
BRandPiper[BSLKN'21]	$t < \frac{n}{2}$	$O(n^2)$	$O(n^3)$	1	✓	q-SDH	High
OptRand	$t < \frac{n}{2}$	$O(n^2)$		1	✓	**q-SDH**	Low

Our protocol - OptRand

Our random beacon protocol guarantees:

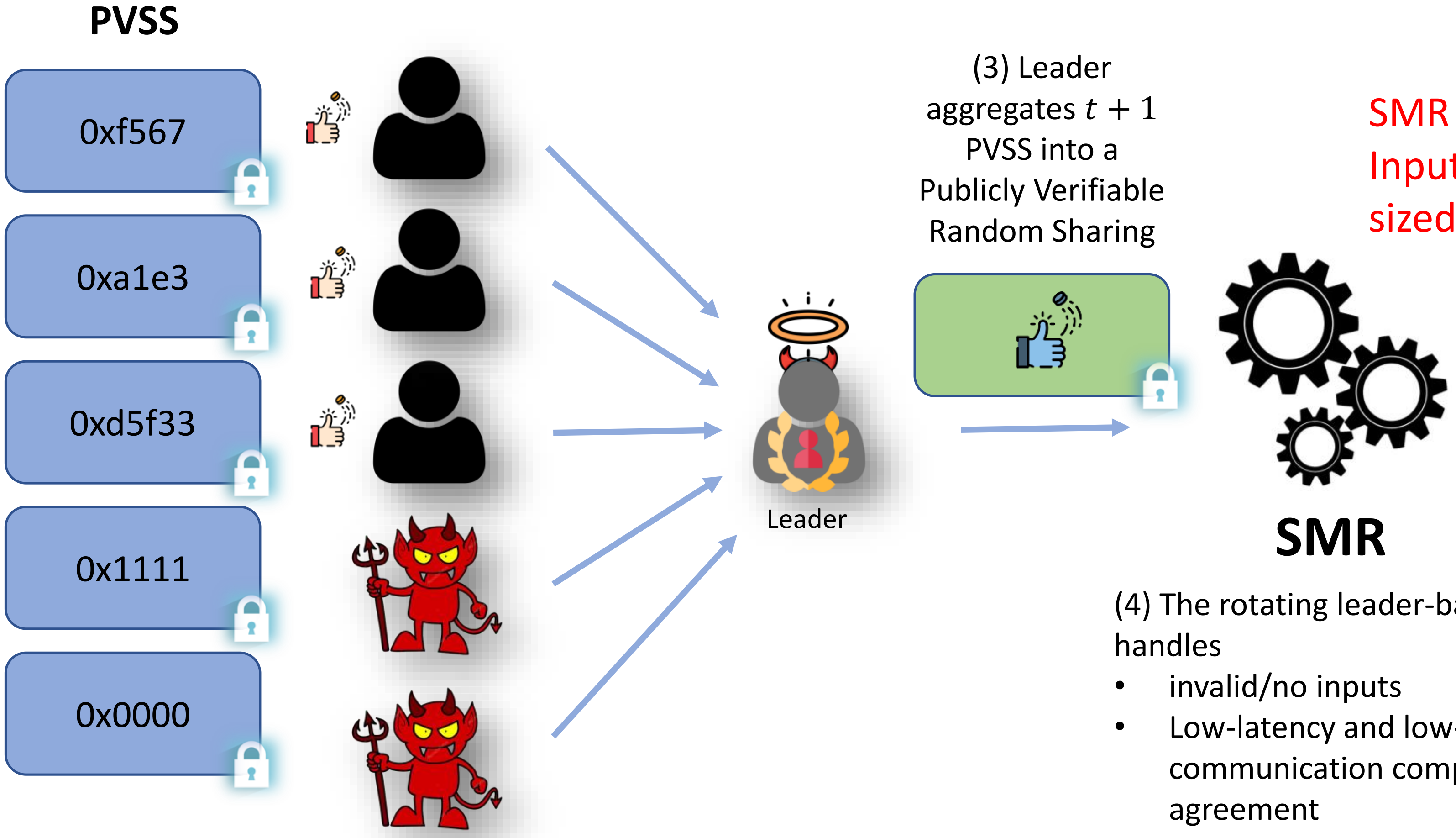
- **$1-$absolute unpredictability**
- **Bias-resistance**
- **Optimal resilience** of $t < n/2$
- **Always $O(n^2)$** communication complexity
- **Optimistic latency**
  - $O(\delta)$ latency during optimistic conditions
  - 11Δ latency otherwise
- **Reconfiguration-friendly** with reconfiguration in $t + 1$ rounds



Δ: known upper bound on n/w delay, δ: actual n/w delay

Technique Overview

(2) Nodes generate random values shared via a homomorphic Publicly Verifiable Secret Sharing (PVSS)



SMR Requirement: Input must be $O(n)$ sized

(1) We have $t < n/2$ Byzantine nodes in the system

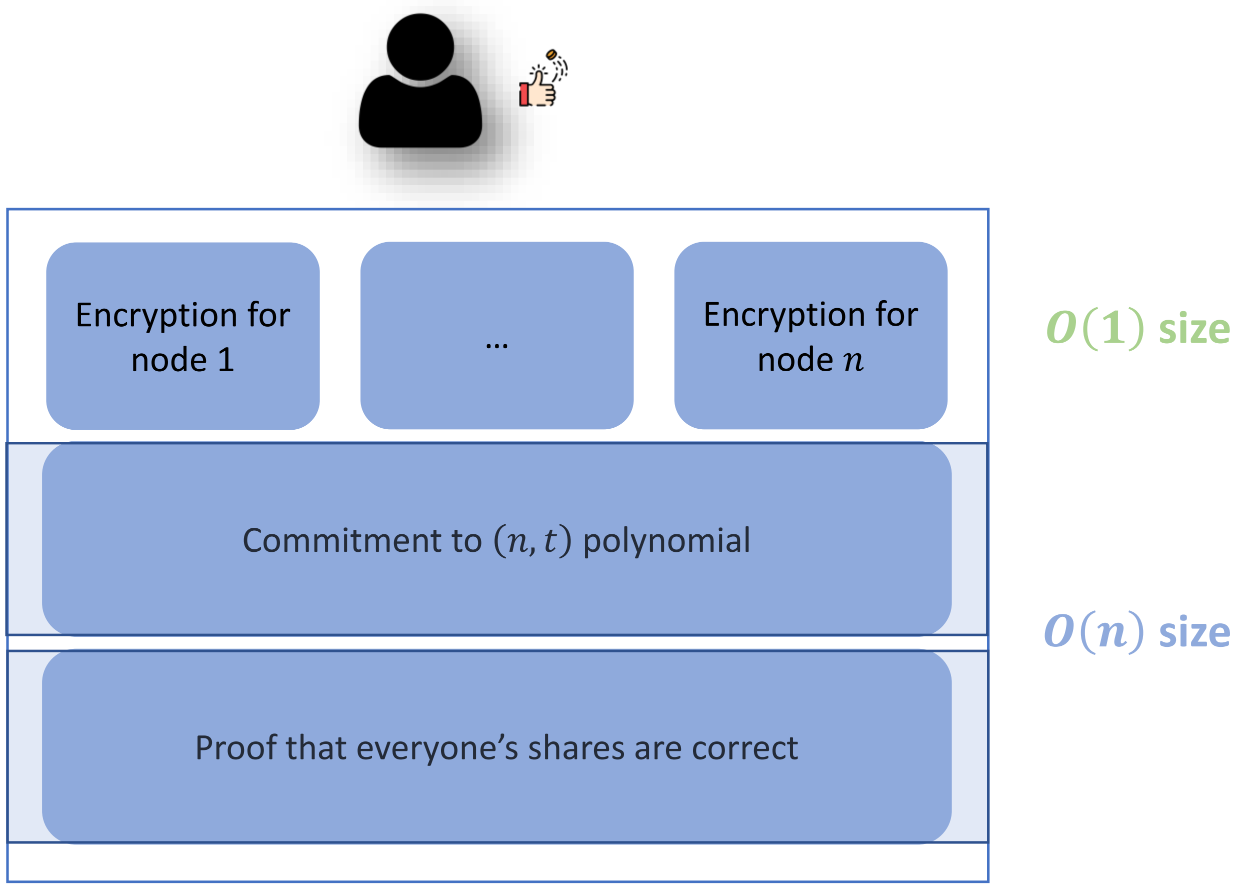
(4) The rotating leader-based SMR handles

- invalid/no inputs
- Low-latency and low-communication complexity agreement
- Beacon generation using reconstruction of PVRS

I will focus on (1), (2), and (3)

indicates secret sharing

Publicly Verifiable Secret Sharing (PVSS)



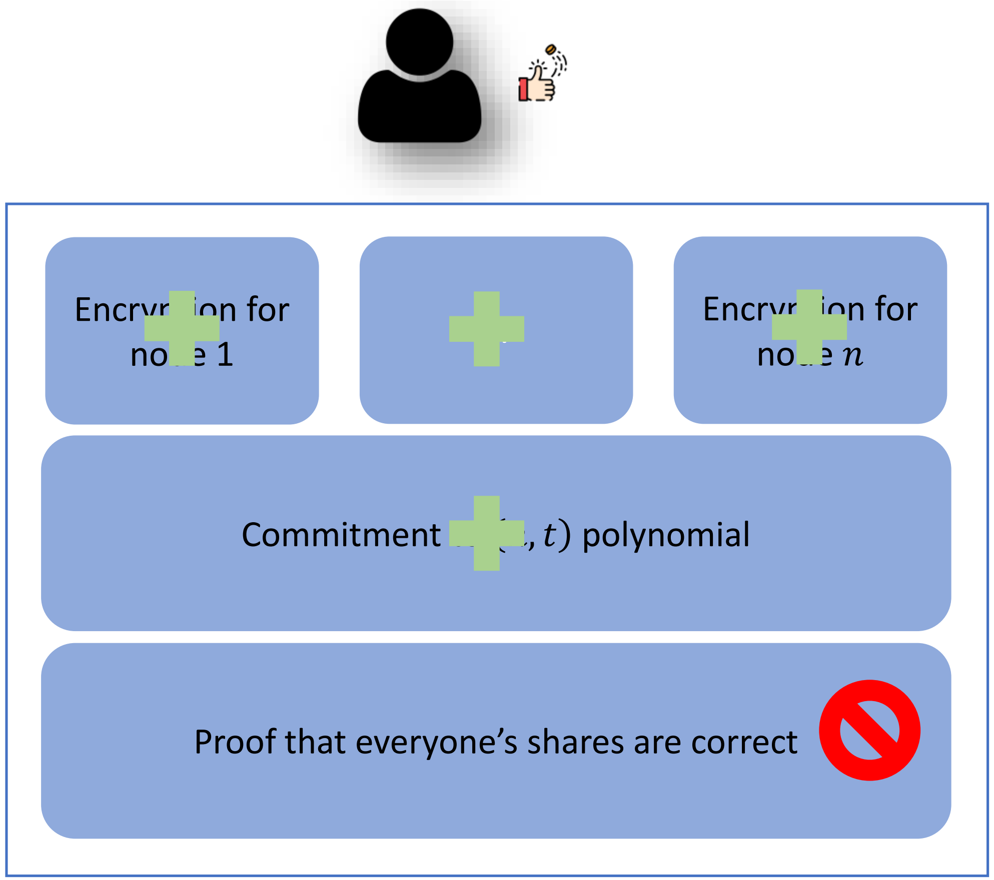
Output of PVSS Share generation

General PVSS Structure

The proof guarantees that

- ✓ **The degree of the polynomial in the commitment portion of the PVSS is t**
- ✓ **The encryptions correspond to the committed polynomial**

Publicly Verifiable Secret Sharing (PVSS)



Output of PVSS Share generation

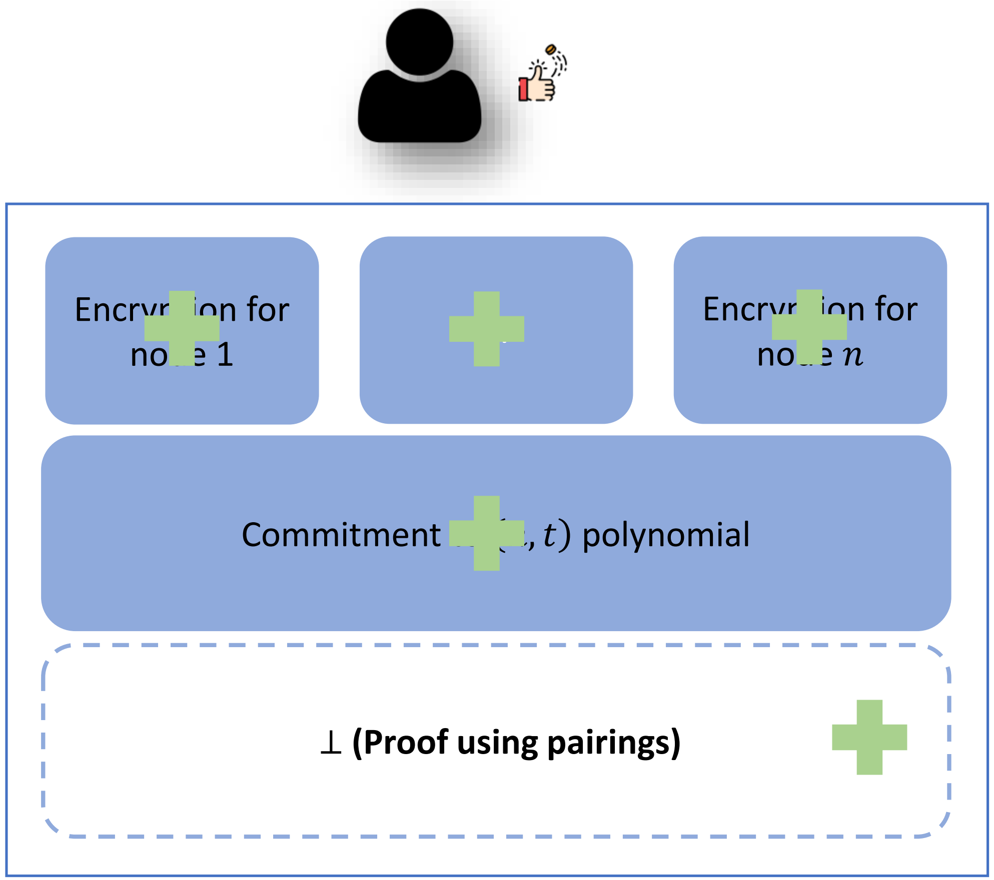
General PVSS Structure

is homomorphic

not homomorphic

Problem: If $O(t)$ sharings are combined, the resulting PVSS is $O(nt)$ sized

Publicly Verifiable Secret Sharing (PVSS)



Output of Pairing-based PVSS Share generation

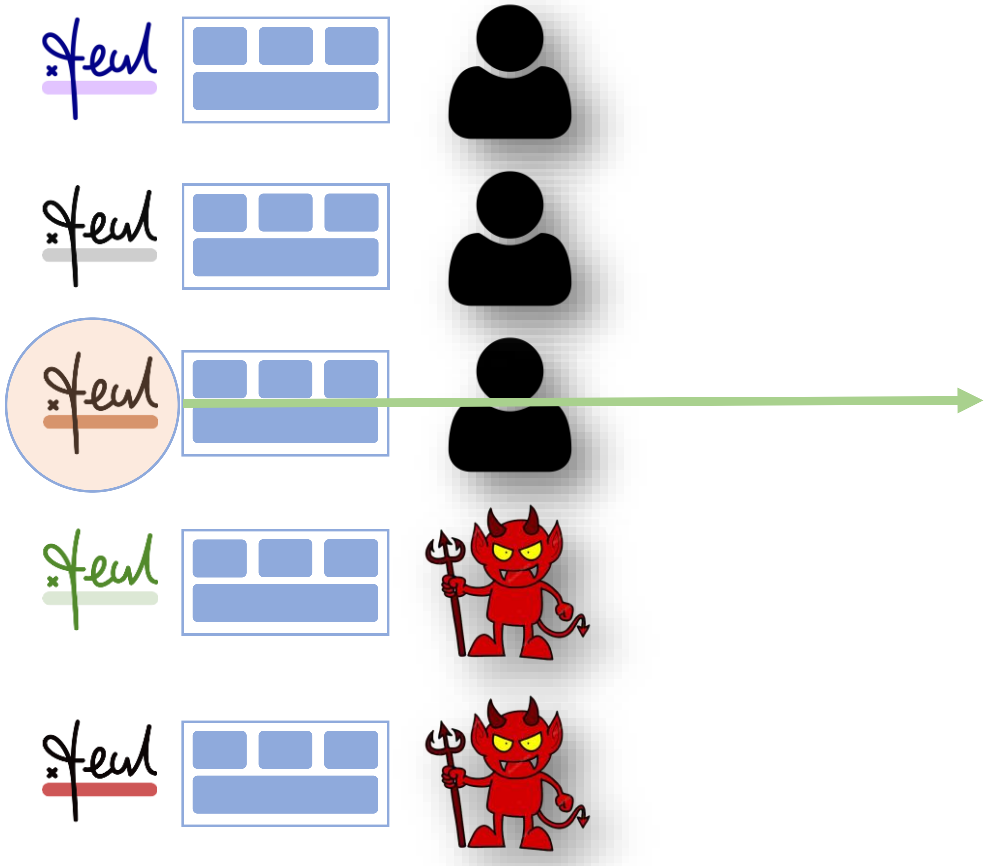
Using Pairing based PVSS from SCRAPE[CD19]

 **is homomorphic**

Problem: An adversarial combiner can **cancel** honest node's shares of r by generating shares of $-r$

We need a mechanism to prevent adversary from forging honest node's shares

Publicly Verifiable Secret Sharing (PVSS)

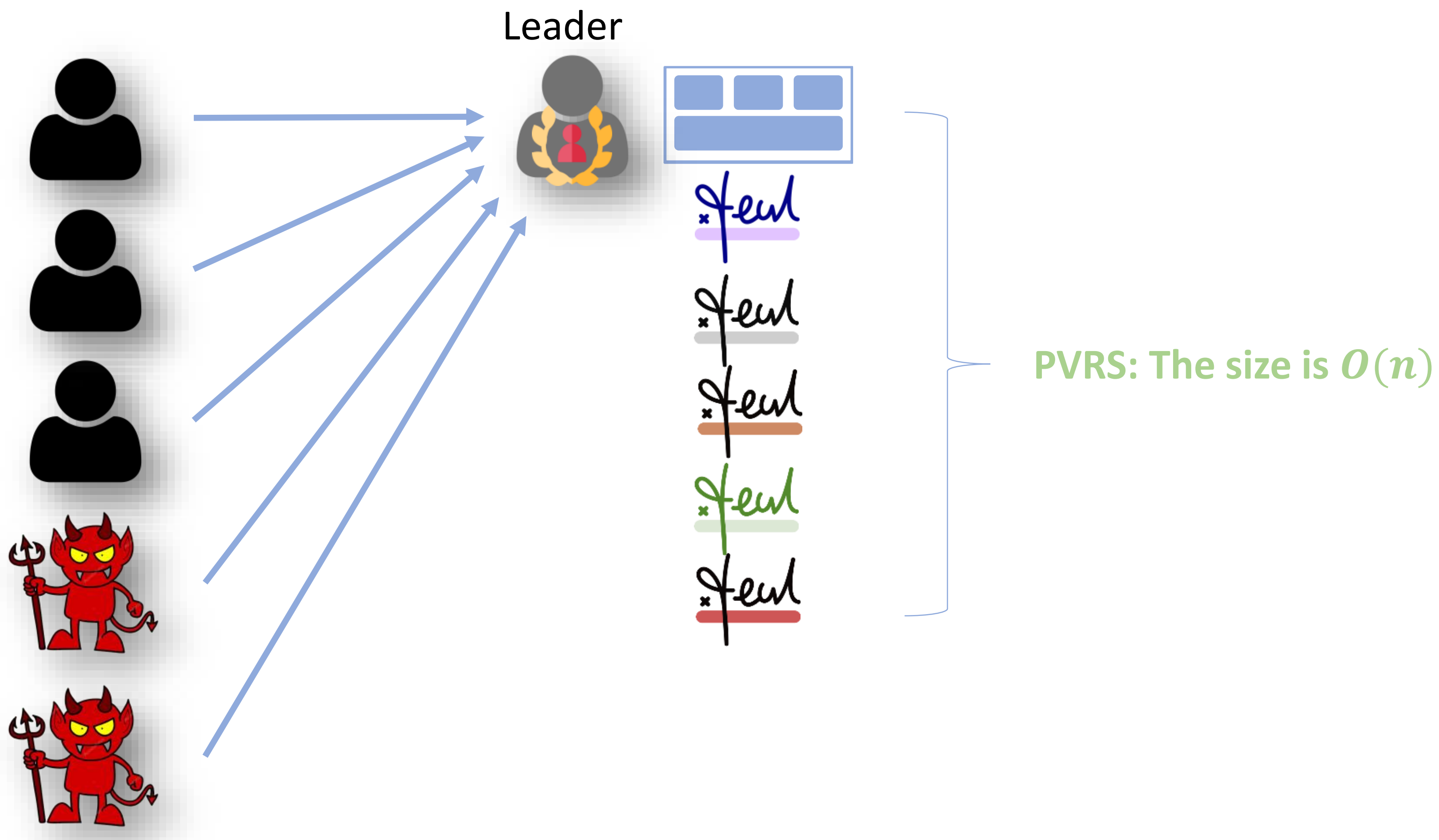


Solution: Add **decomposition proofs** that contain

- A NIZK proof that the node creating the sharing knows the secret in the PVSS
- Authentication information (e.g., digital signature)

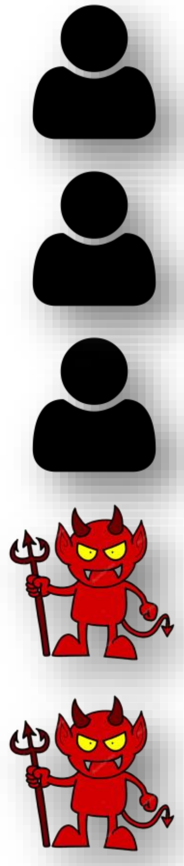
NIZK – Non-Interactive Zero Knowledge

Publicly Verifiable Random Sharing



Publicly Verifiable Random Sharing

Leader



BONUS: If the nodes reconstruct the secret S, then anyone can verify that S is the correct reconstruction using $O(1)$ information

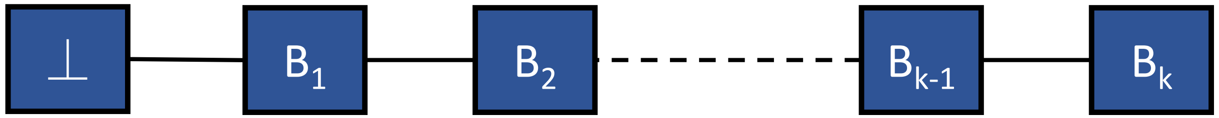
In this example, anyone can verify that

- ✓ All nodes (1, 2, …, 5) have contributed to this PVRS
- ✓ It is an (n, t) sharing
- ✓ The shares for all the nodes are correct

Broadcast Channel or SMR

BFT SMR

All honest nodes output a common set of blocks
- ➢ Despite *t* Byzantine failures out of *n* nodes



Prior BFT SMR protocols with $t < n/2$ resilience:
- $O(n^2)$ communication with threshold setup
  - ➢ Not-reconfiguration friendly
- $O(n^3)$ communication w/o threshold setup
  - ➢ Size of certificate is $O(n)$ bits

BFT SMR of RandPiper[BSLKN'21]
- tolerates $t < n/2$ Byzantine failures
- $O(n^2)$ communication w/o threshold setup
  - ➢ Reconfiguration-friendly
- Each epoch lasts 11Δ

Our approach: Reduce latency during optimistic conditions

certificate: a quorum of signatures, Δ: known upper bound on n/w delay

Optimistic Responsiveness [PS'17]

Allows synchronous protocols to commit responsively in O(δ) time under optimistic conditions

Optimistic conditions:

- Leader is honest
- > 3n/4 nodes in the system follow the protocol

Primary concern:

- Not easy to decide if optimistic conditions are met
  - ➢ Should the protocol progress responsively or synchronously ?

δ: actual n/w delay, Δ: known upper bound on n/w delay, $\delta \ll \Delta$

Responsive commit: commit at δ

Our BFT-SMR Protocol

Fast Path

Makes progress at n/w speed during optimistic conditions

Slow Path

1. Makes progress synchronously under normal conditions
2. Identical to RandPiper BFT SMR

Execute both paths simultaneously

δ: actual n/w delay, Δ: known upper bound on n/w delay, δ << Δ Responsive commit: commit at δ

Key Challenges of the Fast Path Protocol

- **Responsive propagation of linear-sized message**
  - E.g. block proposal, certificates
  - A Byzantine leader could send the message to only some honest nodes
    - All-to-all multicast incurs cubic communication

- **Responsively changing epochs**
  - Traditionally**,** performed using all-to-all multicast of certificates
    - Incurs cubic communication

certificate: a quorum of vote messages

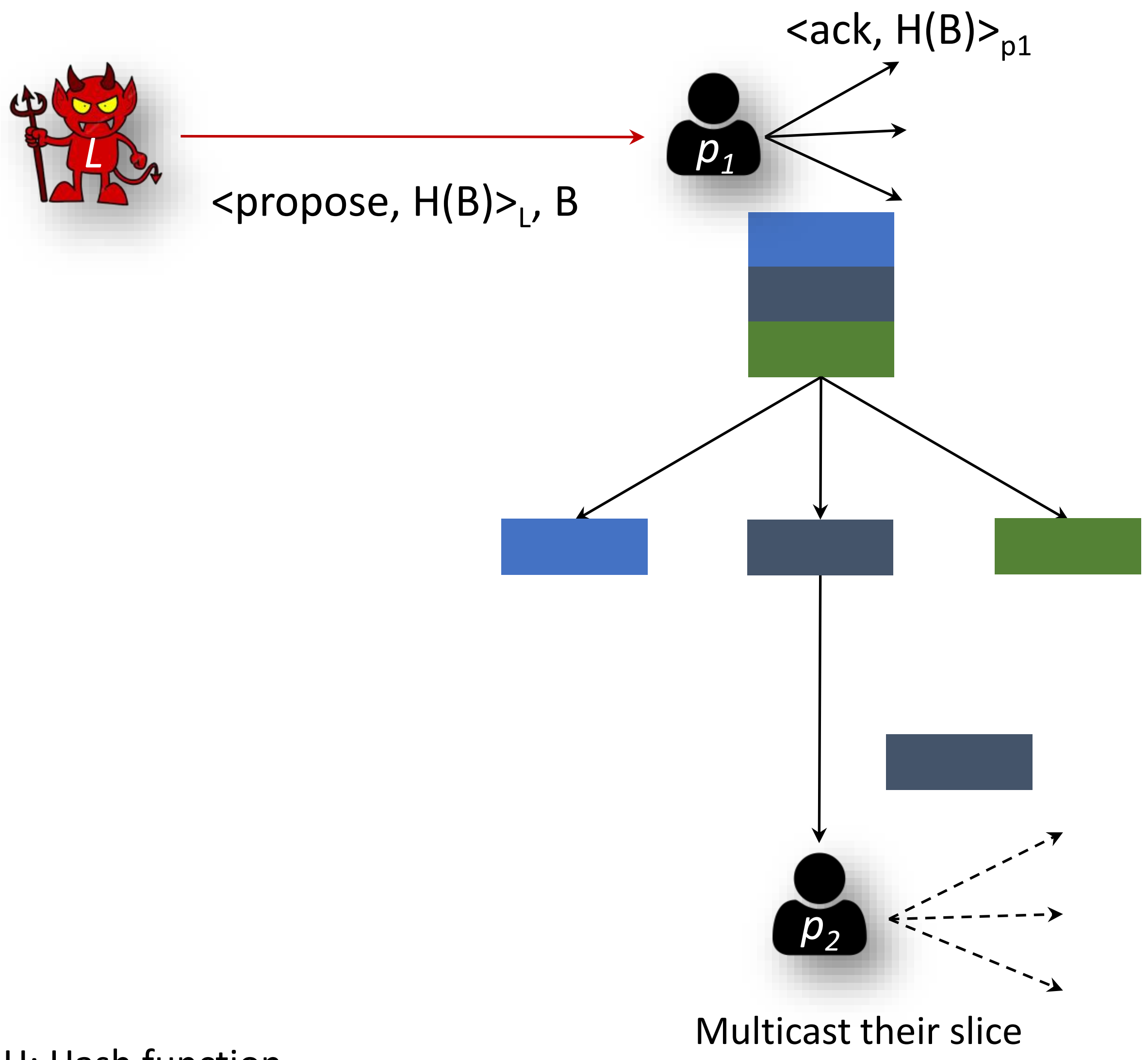
Primitives

- Linear erasure and error correcting code (Reed-Solomon codes)
  - (n, b) RS code
    - Encode: $m_1, \ldots, m_b \longrightarrow s_1, \ldots, s_n$
    - Decode: $s_1, \ldots., s_n \longrightarrow m_1, \ldots, m_b$ tolerates *n* - *b* erasures

In our protocol, we set b = n/4 + 1

- Cryptographic accumulator
  - To prove membership of slices
  - Bilinear accumulator

Responsive Propagation of Linear-sized Message



1. Encode proposal with (n, n/4+1) RS code
2. Send slice s_i to node p_i , multicast ack for B
3. Multicast its slice

1. Consider block B propagated when 3n/4 + 1 nodes ack for block B
2. Decode block B from n/4 + 1 slices

H: Hash function

Responsive Propagation of Linear-sized Message

$3n/4 + 1$ nodes have sent acks for B

At least $n/4 + 1$ of the nodes are honest

$n/4 + 1$ honest nodes will send their slices to all other nodes

- All honest nodes will receive at least $n/4 + 1$ valid slices sufficient to decode the original block proposal

H: Hash function

Responsively Changing Epochs

A synchronization primitive is required to signal all honest nodes to move to higher epoch.

Reconstructed secret opened in an epoch as a synchronization primitive

- Reconstructed secret is constant sized
- All-to-all broadcast of the reconstructed secret incurs O(n^2) communication

Key Features of Our BFT SMR

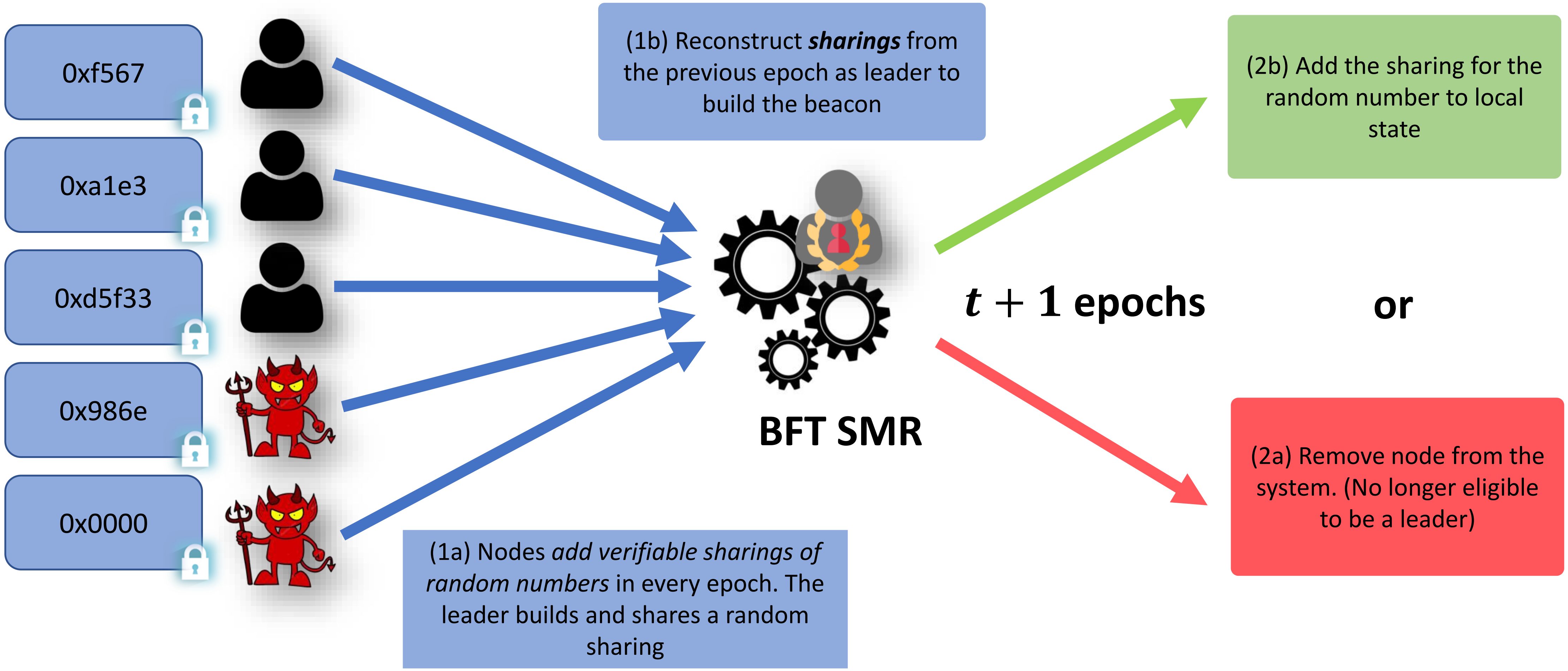
- Rotating leader protocol
  - Leaders rotated every epoch
  - Each epoch lasts for $O(\delta)$ time during optimistic conditions
    - Otherwise, lasts 11Δ time

- $O(n^2)$ communication for $O(n)$-sized input

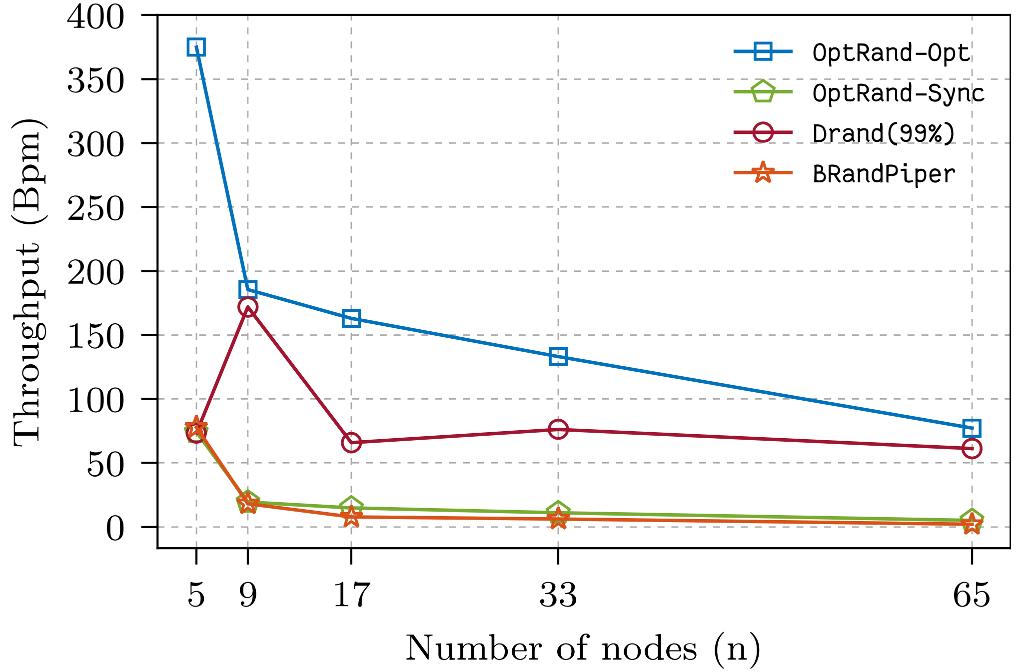
- Commits a decision in $t+1$ epochs in the worst case

δ: actual n/w delay, Δ: known upper bound on n/w delay, $\delta << \Delta$

Putting Things Together - OptRand



Evaluation



AWS

t3.medium

Code: https://github.com/nibeshrestha/optrand

Conclusion

Protocols

1. Optimistically Responsive Distributed Random beacons with $O(n^2)$ communication per beacon

2. Efficient Reconfiguration with $O(n^2)$ communication per epoch and optimistically responsive latency


Adithya Bhat*
PURDUE UNIVERSITY


Nibesh Shrestha*
RIT


Aniket Kate
PURDUE UNIVERSITY


Kartik Nayak
Duke UNIVERSITY

*Equal contribution

Thank You!