Efficient 3-Party Distributed ORAM

PAUL BUNN
JONATHAN KATZ
EYAL KUSHILEVITZ
RAFAIL OSTROVSKY

Overview

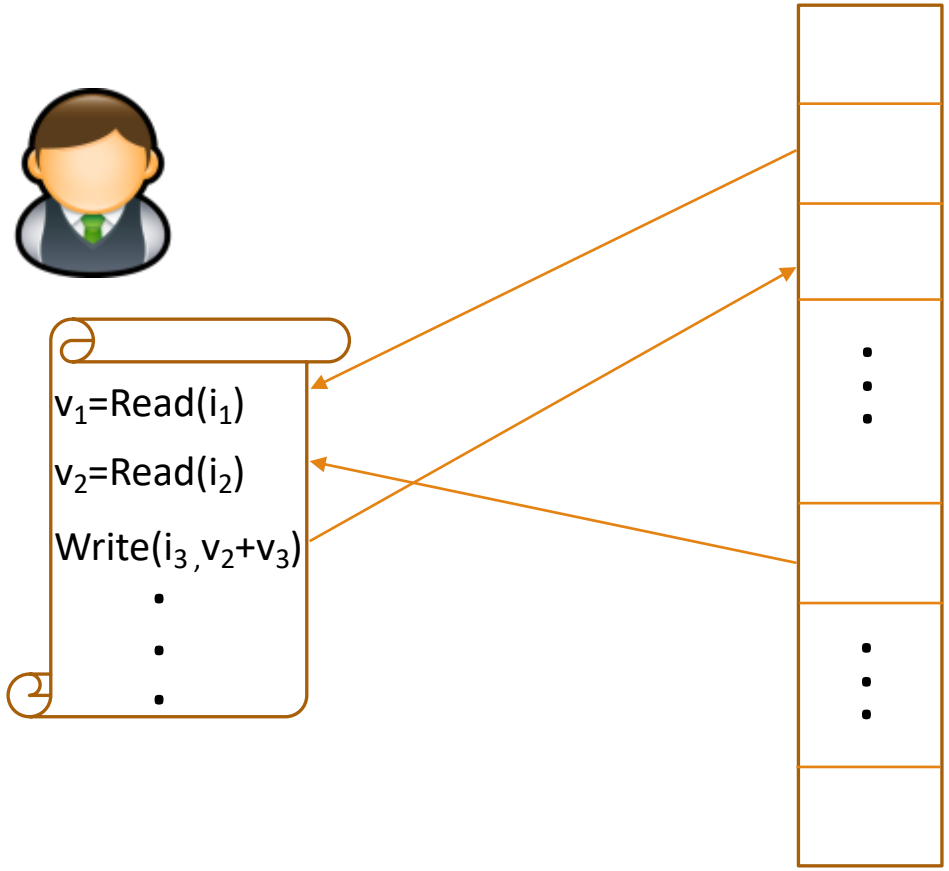
- Background
  - ORAM
  - Distributed ORAM (DORAM)
  - FSS/DPF
  - Our Results
- 3-Party DPF Construction
- 3-Party DORAM Construction
- Future Directions

Our Results

1. **3-Party DPF protocol**
   - Secure against two semi-honest corruptions
   - O($\sqrt{N}$) communication/computation
   - Matches asymptotic complexity of [GI14,BGI15,BGI16]:
     - Concrete efficiency gains
     - Simpler protocol (easier to use, extend/modify)
2. **3-Party DORAM protocol**
   - Secure against one semi-honest corruption
   - O($\sqrt{N}$) communication, O(N) computation per operation
   - Concrete efficiency [DS17]
     - Black-box use of crypto primitives
     - Small constants
     - O(N) operations local and fast (XOR)

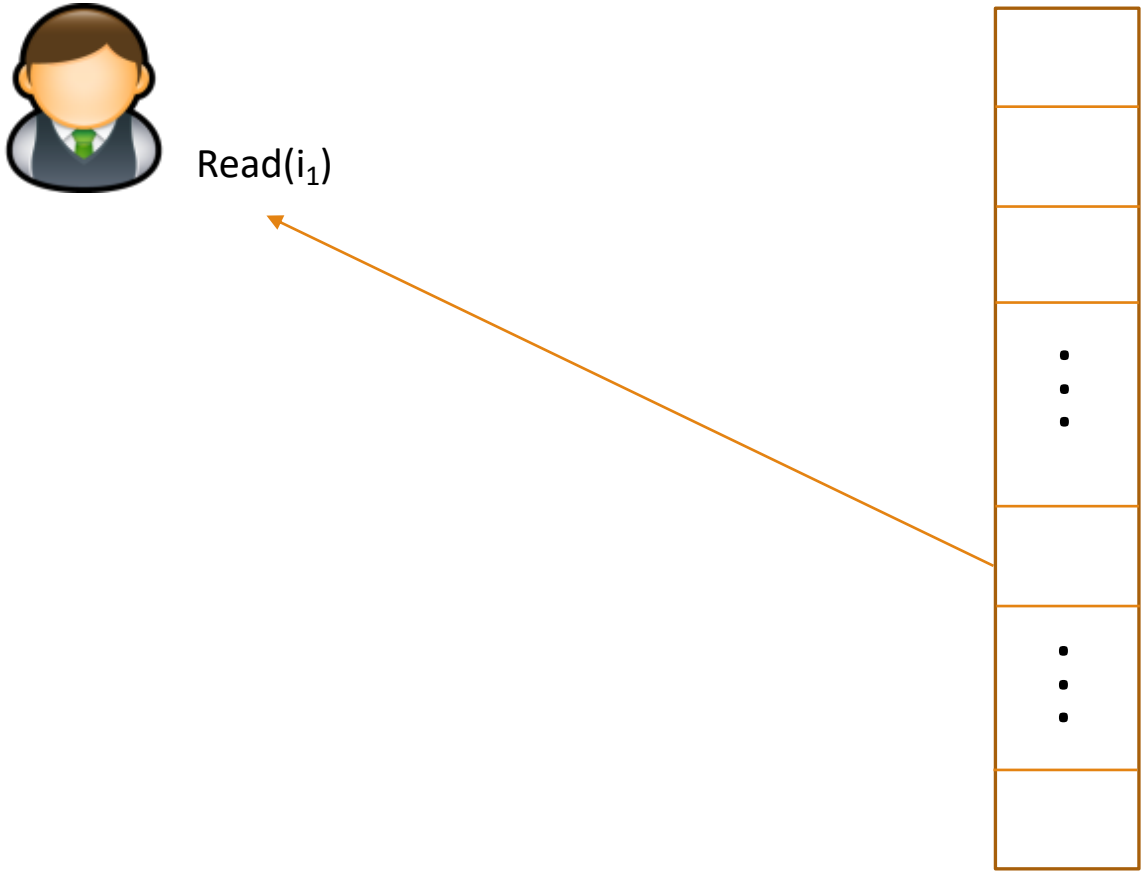
Background: Oblivious RAM (ORAM) [GO96]

- Client
- Memory
- List of Instructions
- Security (Obliviousness):
  - Read: Server does not learn position *i*
  - Write: Server does not learn position *i* nor write value *v*
    - Repetition
    - Timing Attacks
  - Op: Server Cannot distinguish between Read vs. Write
    - Server learns an Op was performed
  - No security required for Client



Background: Oblivious RAM (ORAM)

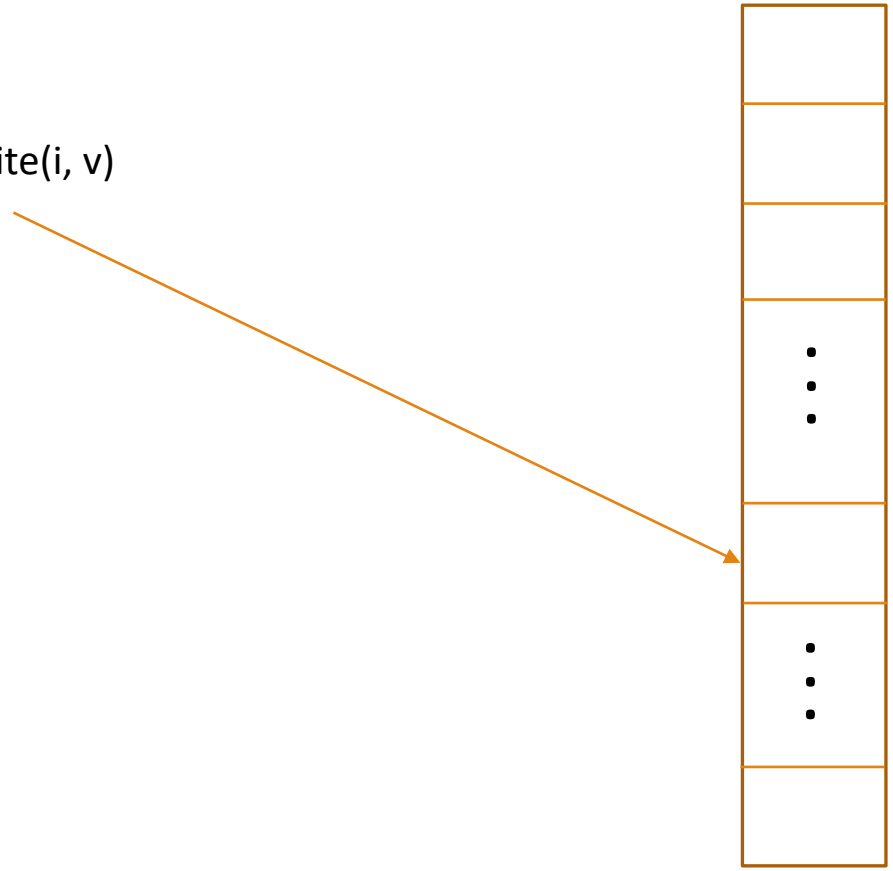
- ORAM via PIR/PIW



Background: Oblivious RAM (ORAM)

- ORAM via PIR/PIW
- Security
  - PIR: Server does not learn *i*
  - PIW: Server does not learn (*i*, *v*)

Write(i, v)



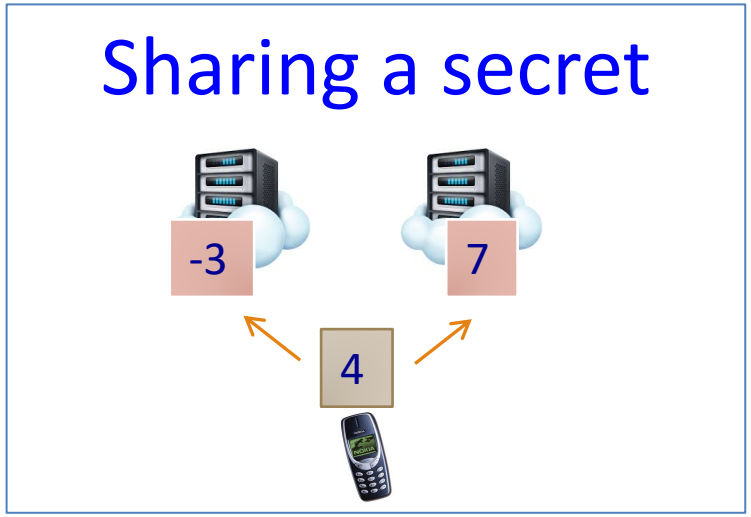
Background: Distributed ORAM (DORAM)

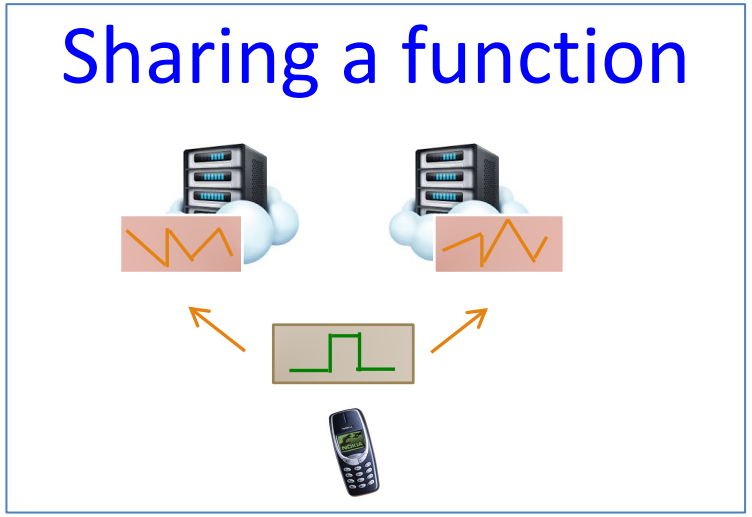
- In some settings, security against *Client* is required
  - Operation being performed (Read, Write)
  - Read/Write position and value
- Client is *distributed*: above information is secret shared amongst *n* parties
  - DORAM ≠ Multi-Server ORAM [LO14]
- Motivating Example: MPC (RAM model of computation)
- As with ORAM, distributed Client (and Server) learn number of Operations

Background: Distributed ORAM (DORAM)

- Using MPC to simulate Client, generic transformation ORAM → DORAM [OS97]
  - Generic MPC expensive (often the bottleneck):
    - Attempt to minimize Client complexity in the computation
    - Search for tailored DORAM solutions (black-box only): [GKK+12, FJK+15, DS17, JW18]
- Superior asymptotic complexity ⇏ faster
  - Hidden constants, system capabilities/bottlenecks (memory, power, bandwidth/latency)
  - Survey of [DS17]

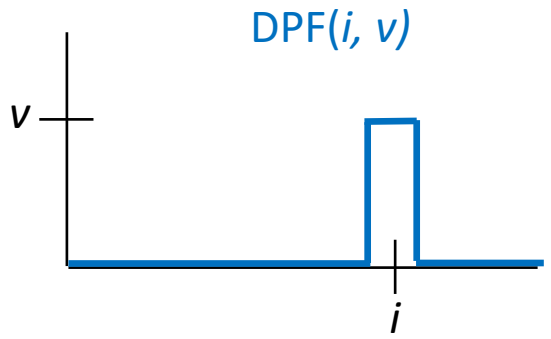
Background: Function Secret Sharing [GI15]





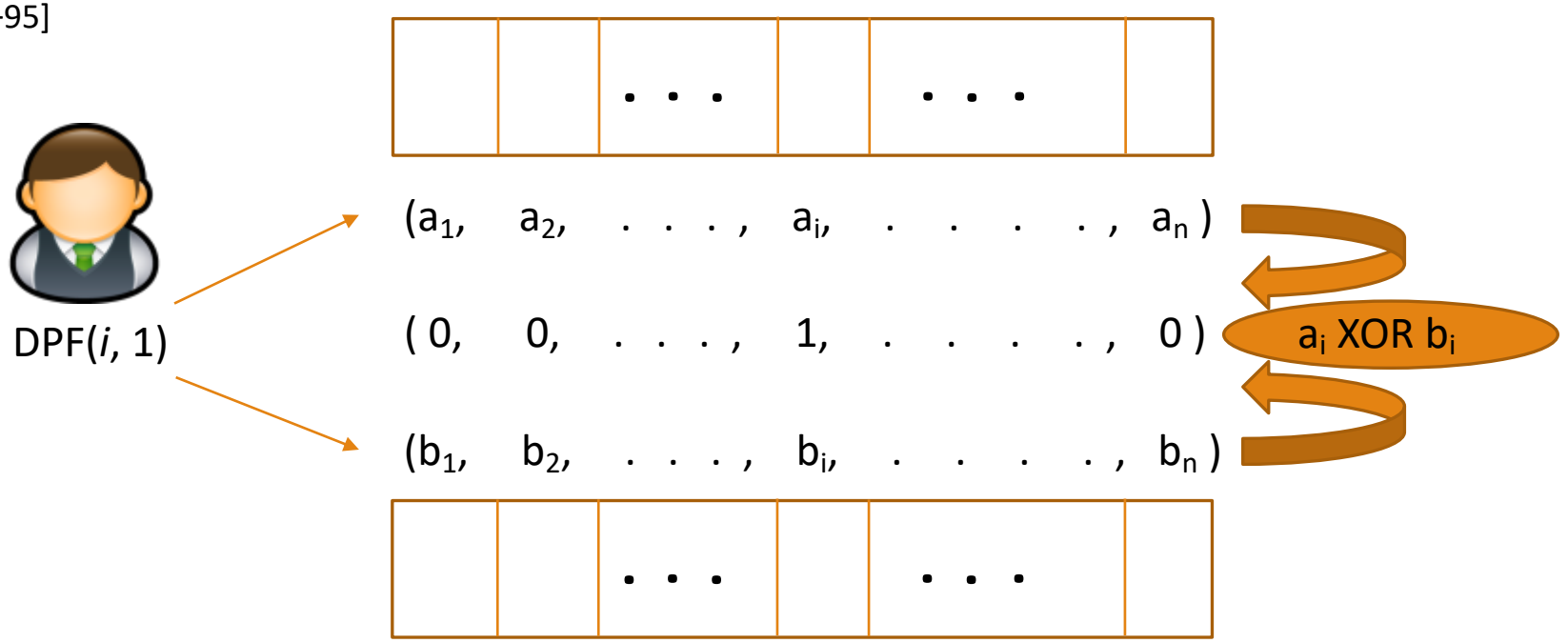
Background: Function Secret Sharing

- Consists of Dealer and N-Parties, and three protocols:
  - Gen(λ, f): Dealer gives "keys" to each party
  - `Eval`($k_{\mathcal{P}}$, x): Party $\mathcal{P}$ uses Gen key $k_{\mathcal{P}}$ to compute their share of f(x)
  - `Reconstruct({Eval}`($k_{\mathcal{P}}$)`}`): Combines all parties' output shares to form f(x)
- Known FSS constructions for certain classes of functions
  - Distributed Point Function (DPF) [GI15,BGI16,BGI17]
  - Step-Functions, Intervals



Background: Function Secret Sharing

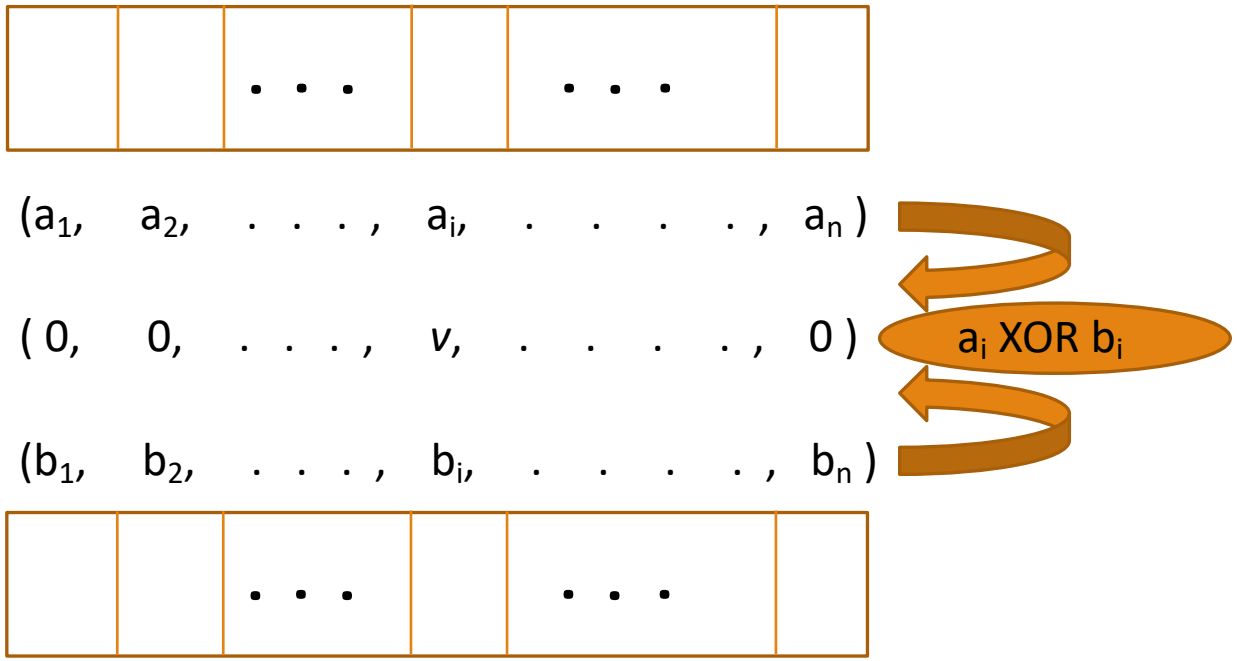
- FSS (DPF) ⟹ PIR [CGK+95]



Background: Function Secret Sharing

- FSS (DPF) ⟹ PIR
- FSS (DPF) ⟹ PIW [OS97]

DPF(i, v)



Background: Function Secret Sharing

- FSS (DPF) $\Longrightarrow$ PIR
- FSS (DPF) $\Longrightarrow$ PIW
- Notes:
  - Server/Databases are different (replicated vs. distinct)
  - Low communication
  - High (linear) server computation, but…
    - All computations are cheap (XOR)
    - Small constants

Our Results

1. **3-Party DPF protocol**
   - Secure against two semi-honest corruptions
   - O($\sqrt{N}$) communication/computation
   - Matches asymptotic complexity of [GI14,BGI15,BGI16]:
     - Concrete efficiency gains
     - Simpler protocol (easier to use, extend/modify)
2. **3-Party DORAM protocol**
   - Secure against one semi-honest corruption
   - O($\sqrt{N}$) communication, O(N) computation per operation
   - Concrete efficiency [DS17]
     - Black-box use of crypto primitives
     - Small constants
     - O(N) operations local and fast (XOR)

Overview

- Background
- 3-Party DPF Construction
  - Why N = 3?
  - Previous Work
  - Our Results
- 3-Party DORAM Construction
- Future Directions

3-Party DPF Construction: Why N=3?

- In some scenarios, number of parties is tunable (so pick optimal)

#Parties	**PROS**	**CONS**
2	2-Party FSS has minimal communication	For ORAM: How to reconcile PIR vs. PIW? For DORAM: How to *distribute* Client?
3	For ORAM: Easy to reconcile PIR vs PIW For DORAM: Black-box Client *distribution*	Best known communication is $\Omega(\sqrt{N})$
4	For ORAM: Easy to reconcile PIR vs PIW	For DORAM: How to *distribute* Client?

3-Party DPF Construction: Why N=3?

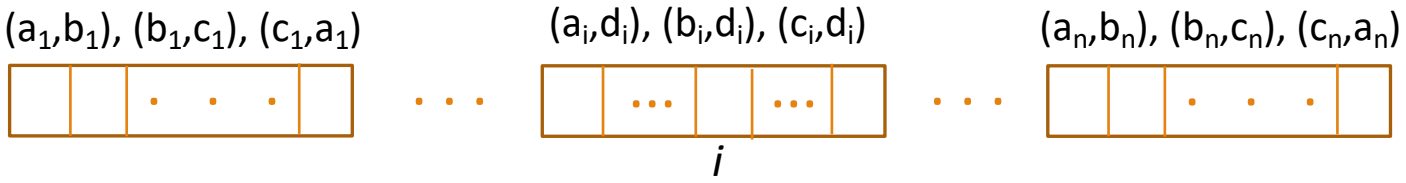
- In some scenarios, number of parties is tunable (so pick optimal)

#Parties	**PROS**	**CONS**
2	2-Party FSS has minimal communication	For ORAM: How to For DORAM: How to *d*
3	For ORAM: Easy to reconcile PIR vs PIW For DORAM: Black-box Client *distribution*	Best known communication: $\Omega(\sqrt{N})$
4	For ORAM: Easy to reconcile PIR vs PIW	For DORAM: How to *distribute* Client?

All FSS-based constructions have linear computation, so $\sqrt{N}$ communication not bottleneck

3-Party DPF Construction: Our Results

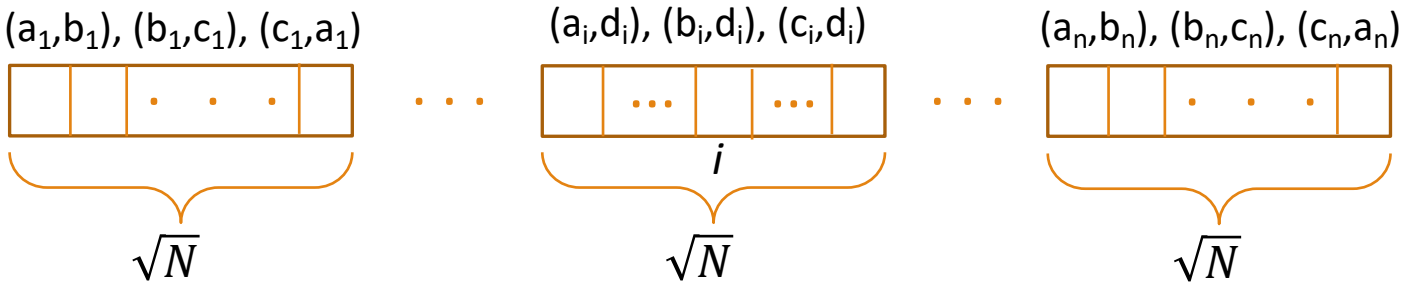
- 3-Party DPF(i, v)
- Warm-Up: Linear Solution
  Pick (N/ λ) keys of length λ:
  - $j \neq i$: keys overlap:
    $0 = (a_j + b_j) + (b_j + c_j) + (c_j + a_j)$
  - $j = i$: keys add to v:
    $v = (a_i + d_i) + (b_i + d_i) + (c_i + d_i)$



- 2-out-of-3 Security:
  For any two players, in all positions j (including $j = i$ and $j \neq i$), exactly one key matches

3-Party DPF Construction: Our Results

- 3-Party DPF(i, v)
- Actual Solution:
  Pick $\sqrt{N}$ PRG seeds of length λ:
  - $j \neq i$: keys overlap:
    (a_j, b_j), (b_j, c_j), (c_j, a_j)
  - $j = i$: keys don't overlap:
    (a_i, d_i), (b_i, d_i), (c_i, d_i)



- Hiccup: Cannot choose seeds:
  $v = G(a_i) + G(b_i) + G(c_i) + G(d_i)$
  Solution: Deal CW:
  $v + G(a_i) + G(b_i) + G(c_i) + G(d_i)$
- Why $\sqrt{N}$?

Overview

- Background
- 3-Party DPF Construction
- 3-Party DORAM Construction
- Future Directions

3-Party DORAM Construction

- DPF $\Longrightarrow$ ORAM: Straightforward (via PIR+PIW)
- DPF $\Longrightarrow$ DORAM: Expensive if using generic MPC reduction
- Recall our 3-Party DPF Construction:
  Generate seeds:
  Off- Block: $\{a_i, b_i\}$, $\{b_i, c_i\}$, $\{c_i, a_i\}$
  On- Block: $\{a_i, d_i\}$, $\{b_i, d_i\}$, $\{c_i, d_i\}$
  Because seeds are random (just need to satisfy overlapping property), they are easy to generate with black-box crypto (OT)

Overview

- Background
- 3-Party DPF Construction
- 3-Party DORAM Construction
- Future Directions

Future Directions

- 3-Party FSS (DPF): Can beat $\sqrt{N}$ communication?
- 3-Party DORAM: Implement and Benchmark
  - Extend feasible parameter regime?
  - Parameter regime where our protocol is superior?
- 3-Party DORAM: Feasibility/Benchmarking for all parameter regimes
  - Mix-and-match
  - Generic DORAM backend for MPC (RAM model)

Thank You!