ORAMs in a Quantum World

Tommaso Gagliardoni\textsuperscript{1} \quad Nikolaos P. Karvelas\textsuperscript{2}
Stefan Katzenbeisser\textsuperscript{2}

\textsuperscript{1} IBM Zurich Research Labs, Switzerland
\textsuperscript{2} TU Darmstadt, Germany

June 26th, 2017
Utrecht, The Netherlands

PQCRYPTO’17
**Scenario:** a computationally constrained client $C$ wants to perform read/write operations on a large database.
Scenario: a computationally constrained client $C$ wants to perform read/write operations on a large database.

Approach: outsource database to a more powerful server $S$ and access data remotely.
Scenario: a computationally constrained client $C$ wants to perform read/write operations on a large database.

Approach: outsource database to a more powerful server $S$ and access data remotely.

Problem: $S$ could be (controlled by) an honest-but-curious adversary $A$, and the database might store private information.
Scenario: a computationally constrained client $C$ wants to perform read/write operations on a large database.

Approach: outsource database to a more powerful server $S$ and access data remotely.

Problem: $S$ could be (controlled by) an honest-but-curious adversary $A$, and the database might store private information.

Possible solution: encrypt database with an encryption scheme $E$. 
Scenario: a computationally constrained client $C$ wants to perform read/write operations on a large database.

Approach: outsource database to a more powerful server $S$ and access data remotely.

Problem: $S$ could be (controlled by) an honest-but-curious adversary $A$, and the database might store private information.

Possible solution: encrypt database with an encryption scheme $E$.

Issue: still, $C$’s access patterns (positions of element read/written) can be leaked.
Introduction

Scenario: a computationally constrained client $C$ wants to perform read/write operations on a large database.

Approach: outsource database to a more powerful server $S$ and access data remotely.

Problem: $S$ could be (controlled by) an honest-but-curious adversary $A$, and the database might store private information.

Possible solution: encrypt database with an encryption scheme $E$.

Issue: still, $C$’s access patterns (positions of element read/written) can be leaked.

Countless applications: secure CPU, software protection, cloud computing, genome analysis, ...
ORAMs (Oblivious Random Access Machines): think of oblivious commitments, but random access (read/write).
ORAMs (Oblivious Random Access Machines): think of oblivious commitments, but random access (read/write).

ORAMs basically work with a double mechanism:
ORAMs (Oblivious Random Access Machines): think of oblivious commitments, but random access (read/write).

ORAMs basically work with a double mechanism:

- encrypting the database with an encryption scheme $\mathcal{E}$; and
- adopting further measures to hide $C$’s access patterns.

Basic example: ‘naif’ ORAM scheme
- encrypt the database with an IND-CPA (randomized) secret-key encryption scheme $\mathcal{E}$ (the key is stored by $C$)
- whenever performing an operation on any element of the database, do the following:
  1. download and decrypt the whole database
  2. perform operation
  3. re-encrypt (re-randomizing) the whole database
  4. upload the database again to $S$.

This works, but very inefficiently.
ORAMs (Oblivious Random Access Machines): think of oblivious commitments, but random access (read/write).

ORAMs basically work with a double mechanism:

- encrypting the database with an encryption scheme $E$; and
- adopting further measures to hide $C$’s access patterns.

Basic example: ‘naif’ ORAM scheme
ORAMs (Oblivious Random Access Machines): think of oblivious commitments, but random access (read/write).

ORAMs basically work with a double mechanism:

- encrypting the database with an encryption scheme $\mathcal{E}$; and
- adopting further measures to hide $\mathcal{C}$’s access patterns.

Basic example: ‘naïf’ ORAM scheme

- encrypt the database with an IND-CPA (randomized) secret-key encryption scheme $\mathcal{E}$ (the key is stored by $\mathcal{C}$)
ORAMs (Oblivious Random Access Machines): think of oblivious commitments, but random access (read/write).

ORAMs basically work with a double mechanism:

- encrypting the database with an encryption scheme $\mathcal{E}$; and
- adopting further measures to hide $C$’s access patterns.

Basic example: ‘naif’ ORAM scheme

- encrypt the database with an IND-CPA (randomized) secret-key encryption scheme $\mathcal{E}$ (the key is stored by $C$)
- whenever performing an operation on any element of the database, do the following:
  1. download and decrypt the whole database
  2. perform operation
  3. re-encrypt (re-randomizing) the whole database
  4. upload the database again to $S$. 
ORAMs (Oblivious Random Access Machines): think of oblivious commitments, but random access (read/write).

ORAMs basically work with a double mechanism:

- encrypting the database with an encryption scheme $E$; and
- adopting further measures to hide $C$’s access patterns.

Basic example: ‘naïf’ ORAM scheme

- encrypt the database with an IND-CPA (randomized) secret-key encryption scheme $E$ (the key is stored by $C$)
- whenever performing an operation on any element of the database, do the following:
  1. download and decrypt the whole database
  2. perform operation
  3. re-encrypt (re-randomizing) the whole database
  4. upload the database again to $S$.

This works, but very inefficiently.
PathORAM: ground-breaking ORAM construction by Stefanov et al. (CCS’13), basis for many recent, efficient ORAM constructions.
PathORAM: ground-breaking ORAM construction by Stefanov et al. (CCS’13), basis for many recent, efficient ORAM constructions.

**Theorem [Stefanov+13]**

PathORAM is an ORAM scheme with client storage/bandwidth requirements scaling only logarithmically with the size of the database.
PathORAM

Data request
(read/write a certain block "id")
Indexing Table ("position map")

\[ \text{id} \leftrightarrow \text{leaf} \]

data request (read/write a certain block "id")
PathORAM

Indexing Table ("position map")

\[ i \overset{\leftrightarrow}{\longrightarrow} \text{leaf} \]

Data request (read/write a certain block "id")
PathORAM

Indexing Table ("position map")
\[ \text{id} \leftrightarrow \text{leaf} \]

data request (read/write a certain block "\text{id}")

\[
\begin{array}{c}
\text{decrypt} \\
\end{array}
\]
PathORAM

\[ S \]

\[ C \]

Indexing Table ("position map")
\[ id \leftrightarrow \text{leaf} \]

data request (read/write a certain block "\( id \))

find requested block and perform operation
PathORAM

Indexing Table ("position map")

\[ id \leftrightarrow \text{leaf} \]

data request (read/write a certain block "id")

scramble block positions (using a PRNG)
PathORAM

Indexing Table ("position map")

data request (read/write a certain block "id")

re-encrypt (re-randomizing)
PathORAM

Indexing Table ("position map")

\[ \text{id} \leftrightarrow \text{leaf} \]

data request (read/write a certain block "id")
Open Problems

- What is a formal definition of ORAM?
- What is the right security model?
- Is it possible to have post-quantum ORAMs?
- What about quantum ORAMs, i.e., if you want to store a ‘quantum database’?
Open Problems

• What is a **formal definition** of ORAM?
• What is the right **security model**?
• Is it possible to have **post-quantum** ORAMs?
• What about **quantum** ORAMs, i.e., if you want to store a ‘quantum database’?

**Motivation for the last point:** first quantum computers expensive, likely to appear “quantum cloud” models (server/client)
Existing ORAM model: Goldreich, Ostrovski (ACM’96).
Existing ORAM model: Goldreich, Ostrovski (ACM’96). This model is formally sound, but very complicated. Consequence: people are not using it. No formal/correct security proofs for existing ORAM schemes.
Existing ORAM model: Goldreich, Ostrovski (ACM’96). This model is formally sound, but very complicated. Consequence: people are not using it. No formal/correct security proofs for existing ORAM schemes.

Our contribution: new, simplified definition and security model (game-based indistinguishability of access patterns under chosen query attack, or AP-IND-CQA)
ORAM Database DB: memory area of server $S$
encoding blocks 1, . . . , $N$. 
The New Model

**ORAM Database DB:** memory area of server $S$ encoding blocks $1, \ldots, N$.

**Data Request:** tuple $dr = (op, id, data)$

- $op \in \{\text{'read'}, \text{'write'}\}$
- $id \in \{1, \ldots, N\}$
The New Model

**ORAM Database DB**: memory area of server $S$ encoding blocks $1, \ldots, N$.

**Data Request**: tuple $dr = (\text{op}, \text{id}, \text{data})$

- $\text{op} \in \{ 'read', 'write' \}$
- $\text{id} \in \{ 1, \ldots, N \}$

**ORAM** as couple of PPT algorithms:

1. $\text{Init}(n, N) \rightarrow (C.\text{mem}, S.\text{mem})$
2. $\text{Access}(C.\text{mem}, S.\text{mem}, dr) \rightarrow (C.\text{mem}', S.\text{mem}', \text{com})$

where $\text{com}$ is the transcript of the channel (communication register) between $C$ and $S$ during execution of Access.
Adversary: PPT circuit $\mathcal{A}$ with complete control of $S$ (as long as it keeps soundness of the ORAM)
Adversary: PPT circuit \( \mathcal{A} \) with complete control of \( S \) (as long as it keeps soundness of the ORAM)

Access pattern (adversarial view) of \( \mathcal{A} \) upon data request \( \text{dr} \): tuple
\[
ap(\text{dr}) = (S.\text{mem}, S.\text{mem}', \text{com})
\]
where \( \text{Access}(C.\text{mem}, S.\text{mem}, \text{dr}) \rightarrow (C.\text{mem}', S.\text{mem}', \text{com}) \)
Access Patterns in PathORAM

access pattern (adversarial view) generated by client's data request
**PathORAM Security**

**AP-IND-CQA** (Access Pattern Indistinguishability under Adaptive Chosen Query Attack)

- the ORAM is initialized
- $A$ performs first CQA phase
- $A$ performs AP-IND (challenge) phase
- $A$ performs second CQA phase
- $A$ outputs a bit

$$\forall A \implies \left| \Pr [A \text{ wins }] - \frac{1}{2} \right| \leq \text{negl}(n)$$

Our contribution:

Theorem PathORAM is an AP-IND-CQA secure ORAM.
AP-IND-CQA (Access Pattern Indistinguishability under Adaptive Chosen Query Attack)

- the ORAM is initialized
- $A$ performs first CQA phase
- $A$ performs AP-IND (challenge) phase
- $A$ performs second CQA phase
- $A$ outputs a bit

$$\forall A \implies \left| \Pr[A \text{ wins }] - \frac{1}{2} \right| \leq \text{negl}(n)$$

Our contribution:

**Theorem**

PathORAM is an AP-IND-CQA secure ORAM.
If $\mathcal{E}$ is a post-quantum IND-CPA encryption scheme, then it is sufficient to instantiate any ORAM with $\mathcal{E}$. 
If $\mathcal{E}$ is a post-quantum IND-CPA encryption scheme, then it is sufficient to instantiate any ORAM with $\mathcal{E}$. ... Isn’t it?
If $\mathcal{E}$ is a post-quantum IND-CPA encryption scheme, then it is sufficient to instantiate any ORAM with $\mathcal{E}$. ... Isn’t it?

**Theorem**

$\mathcal{E}$ pq-IND-CPA $\nRightarrow$ instantiated ORAM pq-AP-IND-CPA
If $\mathcal{E}$ is a post-quantum IND-CPA encryption scheme, then it is sufficient to instantiate any ORAM with $\mathcal{E}$. ... Isn’t it?

**Theorem**

$\mathcal{E}$ pq-IND-CPA $\Leftrightarrow$ instantiated ORAM pq-AP-IND-CPA

Somewhat “dumb” counterexample, using a post-quantum $\mathcal{E}$ but a quantum-weak PRNG.
If $\mathcal{E}$ is a post-quantum IND-CPA encryption scheme, then it is sufficient to instantiate any ORAM with $\mathcal{E}$. ... Isn’t it?

**Theorem**

$\mathcal{E}$ pq-IND-CPA $\nRightarrow$ instantiated ORAM pq-AP-IND-CPA

Somewhat “dumb” counterexample, using a post-quantum $\mathcal{E}$ but a quantum-weak PRNG. **However:** the explicit quantum attack we provide (using previously known, Shor-like quantum routines) is interesting, check it out!
If $\mathcal{E}$ is a post-quantum IND-CPA encryption scheme, then it is sufficient to instantiate any ORAM with $\mathcal{E}$. … Isn’t it?

**Theorem**

$\mathcal{E}$ pq-IND-CPA $\not\Rightarrow$ instantiated ORAM pq-AP-IND-CPA

Somewhat “dumb” counterexample, using a post-quantum $\mathcal{E}$ but a quantum-weak PRNG. However: the explicit quantum attack we provide (using previously known, Shor-like quantum routines) is interesting, check it out!

**Theorem**

$\mathcal{E}$ pq-IND-CPA and PRG post-quantum secure $\implies$ instantiated PathORAM pq-AP-IND-CPA
Quantum database: stores qubits (states)
Quantum database: stores qubits (states), possibly entangled.
Quantum ORAMs

Quantum database: stores qubits (states), possibly entangled.

Quantum ORAM (QORAM): new primitive, with new security model adapted from our classical ORAM model:
Quantum database: stores qubits (states), possibly entangled.

Quantum ORAM (QORAM): new primitive, with new security model adapted from our classical ORAM model:

- $C$ and $S$ are now quantum machines
Quantum database: stores qubits (states), possibly entangled.

Quantum ORAM (QORAM): new primitive, with new security model adapted from our classical ORAM model:

- $C$ and $S$ are now quantum machines
- new model of honest-but-curious quantum adversary
Quantum ORAMs

Quantum database: stores qubits (states), possibly entangled.

Quantum ORAM (QORAM): new primitive, with new security model adapted from our classical ORAM model:

- $C$ and $S$ are now quantum machines
- new model of honest-but-curious quantum adversary
- game-based security definition analogous to the classical case: QAP-IND-CQA
Idea: the action of the adversary on a quantum register during honest execution is \textbf{computationally undetectable}.
Idea: the action of the adversary on a quantum register during honest execution is computationally undetectable.
Idea: the action of the adversary on a quantum register during honest execution is computationally undetectable
Idea: the action of the adversary on a quantum register during honest execution is computationally undetectable.
Safe Extractor

Idea: the action of the adversary on a quantum register during honest execution is **computationally undetectable**

This is a relaxation of other known quantum techniques, e.g., collapsing hash functions, upper bounding diamond norm, etc. (because it only applies undetectability to a certain subset of states in the target register $T$, although the side information $E$ can be arbitrary)
New construction: PathQORAM, similar to PathORAM (with extra tweaks), using a quantum encryption scheme $E_Q$. 
PathQORAM

New construction: **PathQORAM**, similar to PathORAM (with extra tweaks), using a quantum encryption scheme $\mathcal{E}_Q$.

**Theorem**

$\mathcal{E}_Q$ QIND-CPA and PRG post-quantum secure $\Rightarrow$ PathQORAM is QAP-IND-CPA
New construction: PathQORAM, similar to PathORAM (with extra tweaks), using a quantum encryption scheme $\mathcal{E}_Q$.

**Theorem**

$\mathcal{E}_Q$ QIND-CPA and PRG post-quantum secure $\Rightarrow$ PathQORAM is QAP-IND-CPA

- quantum data registers identified by a classical tag
- read and write operation act by swapping quantum data between server’s and client’s memory
- consequence: quantum read and write operations inherently equivalent!
Thanks for your attention!

tog@zurich.ibm.com
PathORAM in detail
PathORAM in detail

\[ S \]

\[ C \]

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>
PathORAM in detail

\[ \text{data request} \ (\text{op, } i, \text{data}) \]
(e.g.: \text{op} = "read")
PathORAM in detail

\[ S \]

\[ C \]

<table>
<thead>
<tr>
<th>position map</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>( N )</td>
</tr>
</tbody>
</table>

data request
(op, \( i \), data)
PathORAM in detail

\[ S \]

\[ C \]

position map

\begin{align*}
\text{id} & \rightarrow \text{leaf} \\
1 & \rightarrow l_1 \\
2 & \rightarrow l_2 \\
\vdots & \\
i & \rightarrow l_i \\
\vdots & \\
N & \rightarrow l_N
\end{align*}

data request

\((\text{op}, i, \text{data})\)
PathORAM in detail

$S$

$C$

$\ell_i$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>i</td>
<td>$l_i$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>N</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

data request $(op, i, data)$
PathORAM in detail

$S$

$C$

$l_i$

data request $(op, i, data)$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>
PathORAM in detail

Data request: $(op, i, data)$

Position map:

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>
PathORAM in detail

\[ S \]

\[ C \]

Data request \((\text{op}, i, \text{data})\)

Position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(l_1)</td>
</tr>
<tr>
<td>2</td>
<td>(l_2)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>(N)</td>
<td>(l_N)</td>
</tr>
</tbody>
</table>
PathORAM in detail

\[ S \]

\[ C \]

Data request (op, i, data)

Position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( l_1 )</td>
</tr>
<tr>
<td>2</td>
<td>( l_2 )</td>
</tr>
<tr>
<td>( \ldots )</td>
<td>( \ldots )</td>
</tr>
<tr>
<td>( N )</td>
<td>( l_N )</td>
</tr>
</tbody>
</table>
PathORAM in detail

$S$

$C$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>N</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

data request

$(op, i, data)$

id data
PathORAM in detail

$S$

$C$

\[
\begin{array}{c|c}
\text{id} & \text{leaf} \\
1 & l_1 \\
2 & l_2 \\
\vdots & \vdots \\
N & l_N \\
\end{array}
\]

data request 
(op, i, data)

id | data
---|---
0  | data
PathORAM in detail

$S$

$C$

data request $(\text{op}, i, \text{data})$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

empty block
PathORAM in detail

\[ S \]

\[ C \]

data request
\[ (\text{op}, i, \text{data}) \]

position map
\[
\begin{array}{c|c}
\text{id} & \text{leaf} \\
1 & l_1 \\
2 & l_2 \\
\vdots & \vdots \\
N & l_N \\
\end{array}
\]

id  
data

empty block  
\[ = \]
do nothing
PathORAM in detail

$S$

$C$

$\ell_i$

data request
(op, $i$, data)

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\ell_1$</td>
</tr>
<tr>
<td>2</td>
<td>$\ell_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$\ell_N$</td>
</tr>
</tbody>
</table>

id  | data |
-----|------|
$j$  | dx05F2p1... |
PathORAM in detail

\[ S \]

\[ C \]

\[ \ell_i \]

data request
\((\text{op, } i, \text{data})\)

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( \ell_1 )</td>
</tr>
<tr>
<td>2</td>
<td>( \ell_2 )</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>( j )</td>
<td>( \ell_j )</td>
</tr>
<tr>
<td>( N )</td>
<td>( \ell_N )</td>
</tr>
</tbody>
</table>

id | data
---|------
\( j \) | dx05F2p1...
PathORAM in detail

\[ S \]

\[ C \]

\[ \ell_i \]

data request \((op, i, data)\)

\begin{array}{|c|c|}
\hline
id & leaf \\
\hline
1 & \ell_1 \\
2 & \ell_2 \\
\vdots & \vdots \\
N & \ell_N \\
\hline
\end{array}

\[ \ell_j \]

id \quad data

\[ j \quad dx05F2p1... \]
PathORAM in detail
PathORAM in detail

- $S$: Secure storage
- $C$: Client
- $\ell_i$: Position map
  - $\text{id}$, $\ell_1$
  - $2 \rightarrow \ell_2$
  - $\ldots$
  - $N \rightarrow \ell_N$
- Data request $(\text{op}, i, \text{data})$
- $j$: Data item
  - $dx05F2p1...$
PathORAM in detail

\[ S \]

\[ C \]

position map

\begin{align*}
\text{id} & \quad \text{leaf} \\
1 & \rightarrow l_1 \\
2 & \rightarrow l_2 \\
\vdots & \\
N & \rightarrow l_N
\end{align*}

data request

(op, i, data)

id data

\[ j \]

dx05F2p1...
PathORAM in detail

$S$

$C$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

data request

$(\text{op}, i, \text{data})$

id  
data

id  
data

$\text{id}_{dx05F2p1...}$

$\text{id}_{88r8ytKLi...}$

$\text{id}_{13}$
PathORAM in detail

\[ S \quad C \]

\[ \ell_i \quad \text{data request} \quad (\text{op}, i, \text{data}) \]

\[
\begin{array}{c|c}
\text{id} & \text{leaf} \\
1 & \ell_1 \\
2 & \ell_2 \\
& \ldots \\
N & \ell_N \\
\end{array}
\]

id data
\[ j \quad \text{dx05F2p1...} \]

id data
\[ 8 \quad \text{f2QpA10x...} \]
PathORAM in detail

\[ S \]

\[ C \]

Position map:

\[
\begin{array}{c|c}
\text{id} & \text{leaf} \\
1 & \ell_1 \\
2 & \ell_2 \\
\vdots & \vdots \\
N & \ell_N \\
\end{array}
\]

Data request \((\text{op}, i, \text{data})\)

\[
\begin{array}{c|c}
\text{id} & \text{data} \\
\hline
j & dx05F2p1... \\
0 & \emptyset \\
\end{array}
\]

Empty block

Path ORAM diagram showing the interaction between nodes in the structure and the underlying key and data operations.
PathORAM in detail

$S$

$C$

$\ell_i$

Data request $(op, i, data)$

Position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

Empty block

$0 \quad \emptyset$

$\quad dx05F2p1...$

Swap
PathORAM in detail

$S$

$C$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

data request

$(\text{op}, i, \text{data})$

id | data  
- |------
0  | $\emptyset$

id | data  
- |------
$j$ | dx05F2p1...
PathORAM in detail

\[ S \]

\[ C \]

**position map**

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( l_1 )</td>
</tr>
<tr>
<td>2</td>
<td>( l_2 )</td>
</tr>
<tr>
<td>\ldots</td>
<td>( \ldots )</td>
</tr>
<tr>
<td>( N )</td>
<td>( l_N )</td>
</tr>
</tbody>
</table>

**data request**

\((\text{op}, i, \text{data})\)

<table>
<thead>
<tr>
<th>id</th>
<th>data</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>791xx12d...</td>
</tr>
</tbody>
</table>
PathORAM in detail

\[ S \]

\[ C \]

data request
\((\text{op}, i, \text{data})\)

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(l_1)</td>
</tr>
<tr>
<td>2</td>
<td>(l_2)</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>(N)</td>
<td>(l_N)</td>
</tr>
</tbody>
</table>

id | data
---|-----
13 | 88r8ytKL...
PathORAM in detail

$S$

$C$

$\ell_i$

Data request $(op, i, data)$

**Position map**

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\ell_1$</td>
</tr>
<tr>
<td>2</td>
<td>$\ell_2$</td>
</tr>
<tr>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$N$</td>
<td>$\ell_N$</td>
</tr>
</tbody>
</table>

id | data
---|---------
8  | f2QpA10x...
PathORAM in detail

\[ S \]

\[ C \]

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( l_1 )</td>
</tr>
<tr>
<td>2</td>
<td>( l_2 )</td>
</tr>
<tr>
<td>( \ldots )</td>
<td>( \ldots )</td>
</tr>
<tr>
<td>( N )</td>
<td>( l_N )</td>
</tr>
</tbody>
</table>

data request

\((\text{op}, i, \text{data})\)

id  data

4  dx05F2p1...
PathORAM in detail

\[ S \]

\[ C \]

\[ \ell_i \]

Data request \((\text{op}, i, \text{data})\)

\[
\begin{array}{c|c}
\text{id} & \text{leaf} \\
1 & \ell_1 \\
2 & \ell_2 \\
\vdots & \vdots \\
N & \ell_N \\
\end{array}
\]

\[
\begin{array}{c|c}
\text{id} & \text{data} \\
i & \text{NiceData...} \\
\end{array}
\]

Block requested
PathORAM in detail

\[ S \]

\[ C \]

*position map*

\[
\begin{array}{c|c}
\text{id} & \text{leaf} \\
1 & l_1 \\
2 & l_2 \\
\vdots & \vdots \\
N & l_N \\
\end{array}
\]

*data request*

\[(\text{op}, i, \text{data})\]

*block requested*

\[=\]

*perform operation*

\[(\text{e.g., read data})\]

\[\text{id} \]

\[\text{data} \]

\[
\begin{array}{c|c}
\text{id} & \text{data} \\
\hline
i & \text{NiceData...} \\
\end{array}
\]
PathORAM in detail

$$S$$

$$C$$

Data request $$(op, i, data)$$

<table>
<thead>
<tr>
<th>position map</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>$i$</td>
</tr>
<tr>
<td>$N$</td>
</tr>
</tbody>
</table>

id | data
---|---
$i$ | NiceData...
PathORAM in detail

$S$

$C$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>$i$</td>
<td>$l'_i$</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

data request

(op, $i$, data)

id   data

$i$   NiceData...

$\$ new random value

$l'_i$
PathORAM in detail

$S$

$C$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$i$</td>
<td>$l'_i$</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

data request

$\text{(op, } i, \text{data)}$

$id$

$data$

$N$

NiceData...
PathORAM in detail

\[ S \]

\[ C \]

\[ \ell_i \]

Data request \((\text{op}, i, \text{data})\)

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(\ell_1)</td>
</tr>
<tr>
<td>2</td>
<td>(\ell_2)</td>
</tr>
<tr>
<td>(\cdots)</td>
<td>(\cdots)</td>
</tr>
<tr>
<td>(i)</td>
<td>(\ell'_i)</td>
</tr>
<tr>
<td>(N)</td>
<td>(\ell_N)</td>
</tr>
</tbody>
</table>

\[ \text{id} \quad \text{data} \]

\[ i \quad \text{NiceData...} \]
PathORAM in detail

$S$

$C$

data request
$(op, i, data)$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

id | data
---|---
$i$ | NiceData...
PathORAM in detail

- $S$: Source node
- $C$: Cloud node
- $\ell_i$: Position map
  - $1 \rightarrow \ell_1$
  - $2 \rightarrow \ell_2$
  - $\ldots$
  - $N \rightarrow \ell_N$

Data request $(op, i, data)$

- $id$ $data$
  - $i$: NiceData...
  - $13$: 88r8ytKLi...
PathORAM in detail

\[ S \]

\[ C \]

data request \((\text{op}, i, \text{data})\)

position map

\begin{align*}
\text{id} & \quad \text{leaf} \\
1 & \quad \ell_1 \\
2 & \quad \ell_2 \\
& \quad \vdots \\
N & \quad \ell_N
\end{align*}

\begin{align*}
\text{id} & \quad \text{data} \\
i & \quad \text{NiceData...} \\
8 & \quad \text{f2QpA10x...}
\end{align*}
PathORAM in detail

\[ S \]

\[ C \]

\[ \ell_i \]

data request
\([\text{op}, i, \text{data}]\)

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>\ell_1</td>
</tr>
<tr>
<td>2</td>
<td>\ell_2</td>
</tr>
<tr>
<td></td>
<td>\vdots</td>
</tr>
<tr>
<td>N</td>
<td>\ell_N</td>
</tr>
</tbody>
</table>

position map

id | data
---|------
i  | NiceData...
4  | dx05F2p1...

all blocks occupied!
PathORAM in detail

$S$  

$C$

Data request $(op, i, data)$

<table>
<thead>
<tr>
<th>position</th>
<th>map</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>leaf</td>
</tr>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

id | data
---|-----
$i$ | NiceData...

All blocks occupied!
Go up one node
PathORAM in detail

$S$

$C$

$\ell_i$

Data request $(\text{op}, i, \text{data})$

<table>
<thead>
<tr>
<th>position map</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>\vdots</td>
</tr>
<tr>
<td>$N$</td>
</tr>
</tbody>
</table>

id | data
---|---
$i$ | NiceData...

id | data
---|---
$0$ | $\emptyset$

Empty block found!
PathORAM in detail

$S$

$C$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

data request

$(\text{op}, i, \text{data})$

id  data

<table>
<thead>
<tr>
<th>0</th>
<th>$\emptyset$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i$</td>
<td>NiceData...</td>
</tr>
</tbody>
</table>
PathORAM in detail

\[ S \]

\[ C \]

\[ \ell_i \]

Data request: \((\text{op}, i, \text{data})\)

Position map:

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(\ell_1)</td>
</tr>
<tr>
<td>2</td>
<td>(\ell_2)</td>
</tr>
<tr>
<td>(\ldots)</td>
<td>(\ldots)</td>
</tr>
<tr>
<td>(N)</td>
<td>(\ell_N)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>id</th>
<th>data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(\emptyset)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>id</th>
<th>data</th>
</tr>
</thead>
<tbody>
<tr>
<td>(i)</td>
<td>NiceData...</td>
</tr>
</tbody>
</table>
PathORAM in detail

\[ S \]

\[ C \]

\[ \ell_i \]

Data request:
\[(\text{op}, i, \text{data})\]

<table>
<thead>
<tr>
<th>position map</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>( N )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>id</th>
<th>data</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>664REiDi...</td>
</tr>
</tbody>
</table>
PathORAM in detail

\[ S \]

\[ C \]

\[ \ell_i \]

data request \((\text{op},i,\text{data})\)

position map

\[
\begin{array}{c|c}
\text{id} & \text{leaf} \\
\hline
1 & \ell_1 \\
2 & \ell_2 \\
\vdots & \vdots \\
N & \ell_N \\
\end{array}
\]

id \hspace{1cm} data

10 \hspace{1cm} 664REiDi...

no empty blocks here
PathORAM in detail

$S$

$C$

data request
$(op, i, data)$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
</tr>
</tbody>
</table>

id | data
---|------
10 | 664REiDi...

no empty blocks here as well!!!
PathORAM in detail

S

C

\[ \ell_i \]

(data request \((op, i, data)\))

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>\ell_1</td>
</tr>
<tr>
<td>2</td>
<td>\ell_2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>N</td>
<td>\ell_N</td>
</tr>
</tbody>
</table>

id | data
---|---
10 | 664REiDi...

stash

22 | bluH201j...
7  | c4R4m9x...
...
PathORAM in detail

- $S$: Storage nodes
- $C$: Computation nodes

Data request $(op, i, data)$

Position map:

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
<th>$l_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$N$</td>
<td>$l_N$</td>
<td></td>
</tr>
</tbody>
</table>

Stash:

<table>
<thead>
<tr>
<th>id</th>
<th>data</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>664REiDi...</td>
</tr>
<tr>
<td>22</td>
<td>bluH201j...</td>
</tr>
<tr>
<td>7</td>
<td>c4R4m9x...</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

Key:
PathORAM in detail

\[ S \]

\[ C \]

data request
\((op, i, data)\)

position map
\[
\begin{array}{c|c}
\text{id} & \text{leaf} \\
\hline
1 & l_1 \\
2 & l_2 \\
& \vdots \\
N & l_N \\
\end{array}
\]

id | data
---|---
0 | \emptyset

stash

<table>
<thead>
<tr>
<th>id</th>
<th>data</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>664REiDi...</td>
</tr>
<tr>
<td>22</td>
<td>bluH201j...</td>
</tr>
<tr>
<td>7</td>
<td>c4R4m9x...</td>
</tr>
<tr>
<td></td>
<td>\ldots</td>
</tr>
</tbody>
</table>
PathORAM in detail

$S$ and $C$ with a data request $(\text{op}, i, \text{data})$.
PathORAM in detail

$S$

$C$

$\ell_i$

data request
$(\text{op, } i, \text{data})$

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\ell_1$</td>
</tr>
<tr>
<td>2</td>
<td>$\ell_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$N$</td>
<td>$\ell_N$</td>
</tr>
</tbody>
</table>

stash
PathORAM in detail

\[ S \]

\[ C \]

position map

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( l_1 )</td>
</tr>
<tr>
<td>2</td>
<td>( l_2 )</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>( N )</td>
<td>( l_N )</td>
</tr>
</tbody>
</table>

data request

\((\text{op}, i, \text{data})\)
PathORAM in detail

\[ S \]

\[ C \]

**position map**

<table>
<thead>
<tr>
<th>id</th>
<th>leaf</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( l_1 )</td>
</tr>
<tr>
<td>2</td>
<td>( l_2 )</td>
</tr>
<tr>
<td>( \ldots )</td>
<td>( \ldots )</td>
</tr>
<tr>
<td>( N )</td>
<td>( l_N )</td>
</tr>
</tbody>
</table>

data request

\((\text{op}, i, \text{data})\)

stash
PathORAM in detail

access pattern (adversarial view) generated by client's data request
PathORAM Improvements

Improvements:

- Storing the position map recursively into nested PathORAMs
- Round complexity can be reduced to just two rounds by using garbled circuits (but with higher computation cost) - see ‘TWORAM’ paper (CRYPTO’16) by Garg et al.
**PathORAM Improvements**

**Improvements:**

Storing the position map **recursively** into nested PathORAMs

<table>
<thead>
<tr>
<th>O-RAM Capacity</th>
<th># Blocks</th>
<th>Block Size</th>
<th>Client Storage</th>
<th>Server Storage</th>
<th>Client Storage O-RAM Capacity</th>
<th>Practical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 GB</td>
<td>$2^{20}$</td>
<td>64 KB</td>
<td>204 MB</td>
<td>205 GB</td>
<td>0.297%</td>
<td>22.5X</td>
</tr>
<tr>
<td>256 GB</td>
<td>$2^{22}$</td>
<td>64 KB</td>
<td>415 MB</td>
<td>819 GB</td>
<td>0.151%</td>
<td>24.1X</td>
</tr>
<tr>
<td>1 TB</td>
<td>$2^{24}$</td>
<td>64 KB</td>
<td>858 MB</td>
<td>3.2 TB</td>
<td>0.078%</td>
<td>25.9X</td>
</tr>
<tr>
<td>16 TB</td>
<td>$2^{28}$</td>
<td>64 KB</td>
<td>4.2 GB</td>
<td>51 TB</td>
<td>0.024%</td>
<td>29.5X</td>
</tr>
<tr>
<td>256 TB</td>
<td>$2^{32}$</td>
<td>64 KB</td>
<td>31 GB</td>
<td>819 TB</td>
<td>0.011%</td>
<td>32.7X</td>
</tr>
<tr>
<td>1024 TB</td>
<td>$2^{34}$</td>
<td>64 KB</td>
<td>101 GB</td>
<td>3072 TB</td>
<td>0.009%</td>
<td>34.4X</td>
</tr>
</tbody>
</table>
PathORAM Improvements

Improvements:

Storing the position map **recursively** into nested PathORAMs

<table>
<thead>
<tr>
<th>O-RAM Capacity</th>
<th># Blocks</th>
<th>Block Size</th>
<th>Client Storage</th>
<th>Server Storage</th>
<th>Client Storage O-RAM Capacity</th>
<th>Practical Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 GB</td>
<td>$2^{20}$</td>
<td>64 KB</td>
<td>204 MB</td>
<td>205 GB</td>
<td>0.297%</td>
<td>22.5X</td>
</tr>
<tr>
<td>256 GB</td>
<td>$2^{22}$</td>
<td>64 KB</td>
<td>415 MB</td>
<td>819 GB</td>
<td>0.151%</td>
<td>24.1X</td>
</tr>
<tr>
<td>1 TB</td>
<td>$2^{24}$</td>
<td>64 KB</td>
<td>858 MB</td>
<td>3.2 TB</td>
<td>0.078%</td>
<td>25.9X</td>
</tr>
<tr>
<td>16 TB</td>
<td>$2^{28}$</td>
<td>64 KB</td>
<td>4.2 GB</td>
<td>51 TB</td>
<td>0.024%</td>
<td>29.5X</td>
</tr>
<tr>
<td>256 TB</td>
<td>$2^{32}$</td>
<td>64 KB</td>
<td>31 GB</td>
<td>819 TB</td>
<td>0.011%</td>
<td>32.7X</td>
</tr>
<tr>
<td>1024 TB</td>
<td>$2^{34}$</td>
<td>64 KB</td>
<td>101 GB</td>
<td>3072 TB</td>
<td>0.009%</td>
<td>34.4X</td>
</tr>
</tbody>
</table>

Round complexity can be reduced to just **two** rounds by using garbled circuits (but with higher computation cost) - see ‘TWORAM’ paper (CRYPTO’16) by Garg et al.
AP-IND-CQA Game in detail

- $A$ chooses $n, N$ and Init$(n, N)$ is executed
• $A$ chooses $n, N$ and $\text{Init}(n, N)$ is executed
• (first CQA phase) repeat adaptively up to $\text{poly}$ times:
The AP-IND-CQA Game in detail:

- $A$ chooses $n, N$ and $\text{Init}(n, N)$ is executed.
- (first CQA phase) repeat adaptively up to poly times:
  - $A$ chooses data request $dr$
  - $C$ executes $\text{Access}(C, \text{mem}, S, \text{mem}, dr)$
  - $A$ receives back $ap(dr)$
- (second CQA phase) as before
- $A$ outputs a bit $b'$ and wins iff $b = b'$. 

Note: The content provided is a simplified representation of the AP-IND-CQA game. For a more detailed understanding, refer to the original document or relevant academic resources.
AP-IND-CQA Game in detail

- $\mathcal{A}$ chooses $n$, $N$ and $\text{Init}(n, N)$ is executed
- (first CQA phase) repeat adaptively up to poly times:
  - $\mathcal{A}$ chooses data request $\text{dr}$
  - $\mathcal{C}$ executes $\text{Access}(\mathcal{C}.\text{mem}, S.\text{mem}, \text{dr})$
• $A$ chooses $n, N$ and $\text{Init}(n, N)$ is executed
• (first CQA phase) repeat adaptively up to poly times:
  • $A$ chooses data request $dr$
  • $C$ executes $\text{Access}(C \cdot \text{mem}, S \cdot \text{mem}, dr)$
  • $A$ receives back $\text{ap}(dr)$
AP-IND-CQA Game in detail

- $A$ chooses $n, N$ and $\text{Init}(n, N)$ is executed
- (first CQA phase) repeat adaptively up to $\text{poly}$ times:
  - $A$ chooses data request $dr$
  - $C$ executes $\text{Access}(C.\text{mem}, S.\text{mem}, dr)$
  - $A$ receives back $\text{ap}(dr)$
- (challenge phase) $A$ chooses two data requests $dr_0, dr_1$ and sends them to $C$
• $\mathcal{A}$ chooses $n, N$ and $\text{Init}(n, N)$ is executed
• (first CQA phase) repeat adaptively up to poly times:
  • $\mathcal{A}$ chooses data request $\text{dr}$
  • $\mathcal{C}$ executes $\text{Access}(\mathcal{C}.\text{mem}, S.\text{mem}, \text{dr})$
  • $\mathcal{A}$ receives back $\text{ap}(\text{dr})$
• (challenge phase) $\mathcal{A}$ chooses two data requests $\text{dr}_0, \text{dr}_1$ and sends them to $\mathcal{C}$
  • $\mathcal{C}$ flips a random bit $b$
• \( A \) chooses \( n, N \) and \( \text{Init}(n, N) \) is executed

• (first CQA phase) repeat adaptively up to \( \text{poly} \) times:
  • \( A \) chooses data request \( dr \)
  • \( C \) executes \( \text{Access}(C.\text{mem}, S.\text{mem}, dr) \)
  • \( A \) receives back \( \text{ap}(dr) \)

• (challenge phase) \( A \) chooses two data requests \( dr_0, dr_1 \) and sends them to \( C \)
  • \( C \) flips a random bit \( b \)
  • \( C \) executes \( \text{Access}(C.\text{mem}, S.\text{mem}, dr_b) \)
A chooses \( n, N \) and \( \text{Init}(n, N) \) is executed

(first CQA phase) repeat adaptively up to poly times:

- A chooses data request \( \text{dr} \)
- \( C \) executes \( \text{Access}(C.m\text{em}, S.m\text{em}, \text{dr}) \)
- A receives back \( \text{ap}(	ext{dr}) \)

(challenge phase) A chooses two data requests \( \text{dr}_0, \text{dr}_1 \) and sends them to \( C \)

- \( C \) flips a random bit \( b \)
- \( C \) executes \( \text{Access}(C.m\text{em}, S.m\text{em}, \text{dr}_b) \)
- A receives back \( \text{ap}(	ext{dr}_b) \)
• $\mathcal{A}$ chooses $n, N$ and $\text{Init}(n, N)$ is executed
• (first CQA phase) repeat adaptively up to poly times:
  • $\mathcal{A}$ chooses data request $\text{dr}$
  • $\mathcal{C}$ executes $\text{Access}(\mathcal{C}.\text{mem}, S.\text{mem}, \text{dr})$
  • $\mathcal{A}$ receives back $\text{ap}(\text{dr})$
• (challenge phase) $\mathcal{A}$ chooses two data requests $\text{dr}_0, \text{dr}_1$ and sends them to $\mathcal{C}$
  • $\mathcal{C}$ flips a random bit $b$
  • $\mathcal{C}$ executes $\text{Access}(\mathcal{C}.\text{mem}, S.\text{mem}, \text{dr}_b)$
  • $\mathcal{A}$ receives back $\text{ap}(\text{dr}_b)$
• (second CQA phase) as before
AP-IND-CQA Game in detail

- $A$ chooses $n, N$ and $\text{Init}(n, N)$ is executed
- (first CQA phase) repeat adaptively up to $\text{poly}$ times:
  - $A$ chooses data request $\text{dr}$
  - $C$ executes $\text{Access}(C.\text{mem}, S.\text{mem}, \text{dr})$
  - $A$ receives back $\text{ap}(\text{dr})$
- (challenge phase) $A$ chooses two data requests $\text{dr}_0, \text{dr}_1$ and sends them to $C$
  - $C$ flips a random bit $b$
  - $C$ executes $\text{Access}(C.\text{mem}, S.\text{mem}, \text{dr}_b)$
  - $A$ receives back $\text{ap}(\text{dr}_b)$
- (second CQA phase) as before
- $A$ outputs a bit $b'$ and wins iff $b = b'$. 