# Black Box Crypto is Useless for Doubly Efficient PIR

Wei-Kai Lin
University of Virginia

**Ethan Mook**
Northeastern

Daniel Wichs
Northeastern
& NTT Research

# Black Box Crypto is Useless for Doubly Efficient PIR

Wei-Kai Lin
University of Virginia

**Ethan Mook**
Northeastern

Daniel Wichs
Northeastern
& NTT Research

Eurocrypt 2025

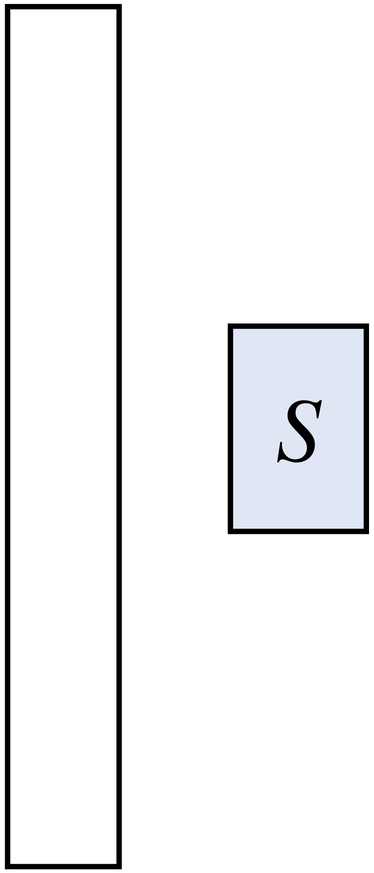
# Private Information Retrieval (PIR)

Interactive protocol between a server $S$ and a client $C$

# Private Information Retrieval (PIR)

Interactive protocol between a server $S$ and a client $C$

$\mathrm{DB} \in \{0,1\}^N$



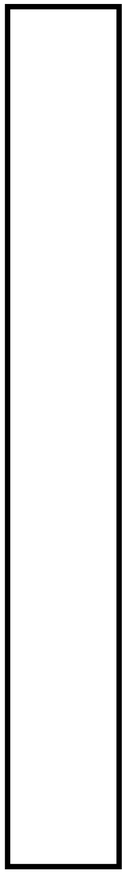
$S$

$C$

# Private Information Retrieval (PIR)

Interactive protocol between a server $S$ and a client $C$

$\mathrm{DB} \in \{0,1\}^N$



$S$

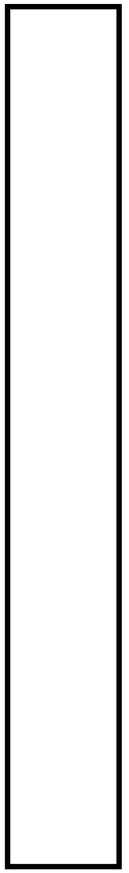
$C$

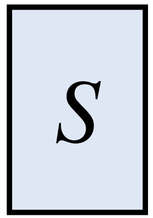
input: $i \in [N]$

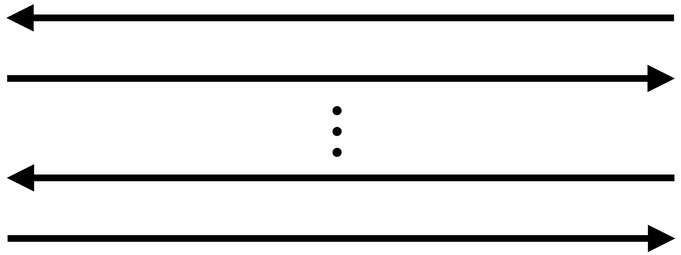
# Private Information Retrieval (PIR)

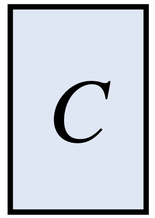
Interactive protocol between a server $S$ and a client $C$

$\mathrm{DB} \in \{0,1\}^N$





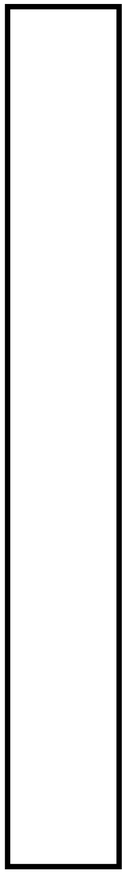


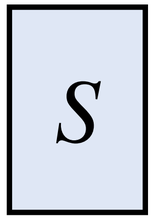

input: $i \in [N]$

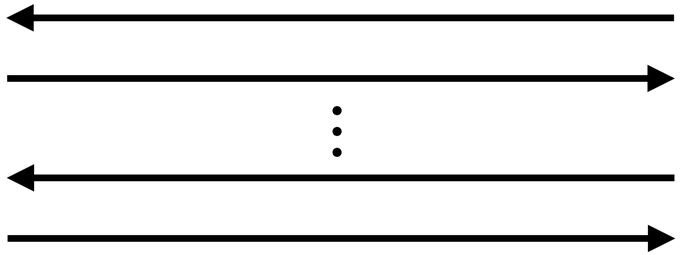
# Private Information Retrieval (PIR)

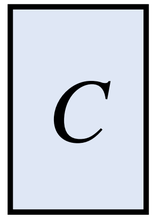
Interactive protocol between a server $S$ and a client $C$

$\mathrm{DB} \in \{0,1\}^N$







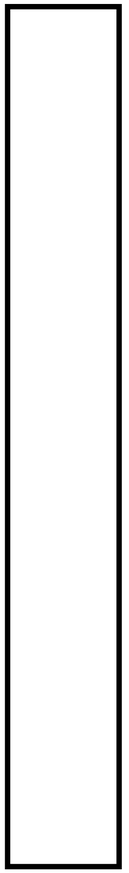

input: $i \in [N]$

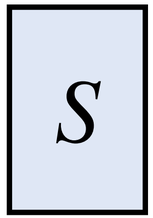
output: $\mathrm{DB}[i]$

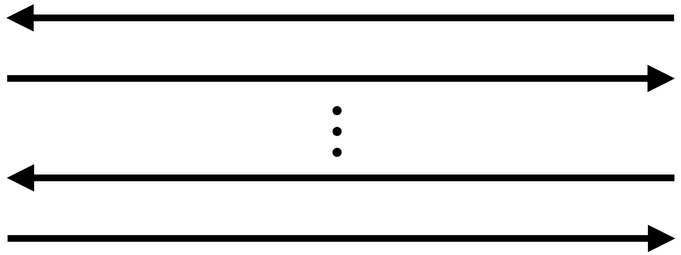
# Private Information Retrieval (PIR)

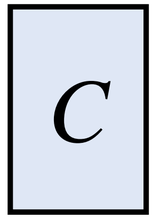
Interactive protocol between a server $S$ and a client $C$

$\mathrm{DB} \in \{0,1\}^N$








input: $i \in [N]$

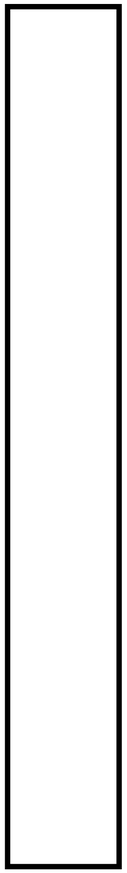
output: $\mathrm{DB}[i]$

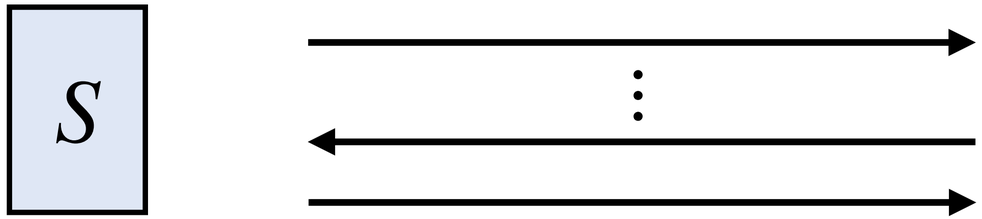
Security: Server learns nothing about $i$

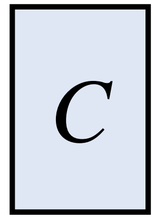
# Private Information Retrieval (PIR)

Interactive protocol between a server $S$ and a client $C$

$\mathrm{DB} \in \{0,1\}^N$






input: $i \in [N]$

output: $\mathrm{DB}[i]$

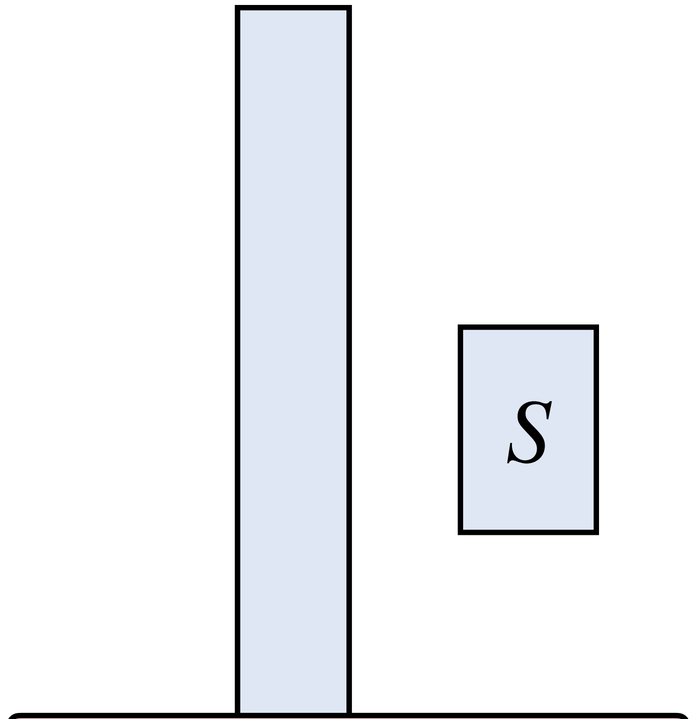
Communication: $o(N)$

Security: Server learns nothing about $i$

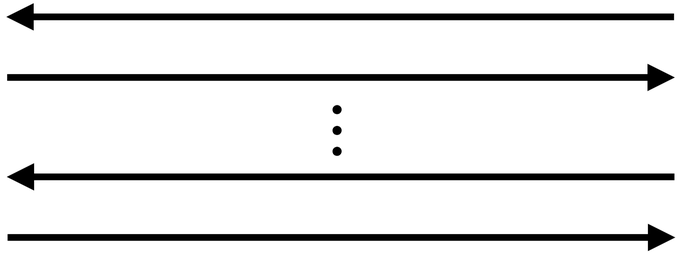
# Private Information Retrieval (PIR)

Interactive protocol between a server $S$ and a client $C$

$\mathrm{DB} \in \{0,1\}^N$



$S$



$C$

input: $i \in [N]$

output: $\mathrm{DB}[i]$

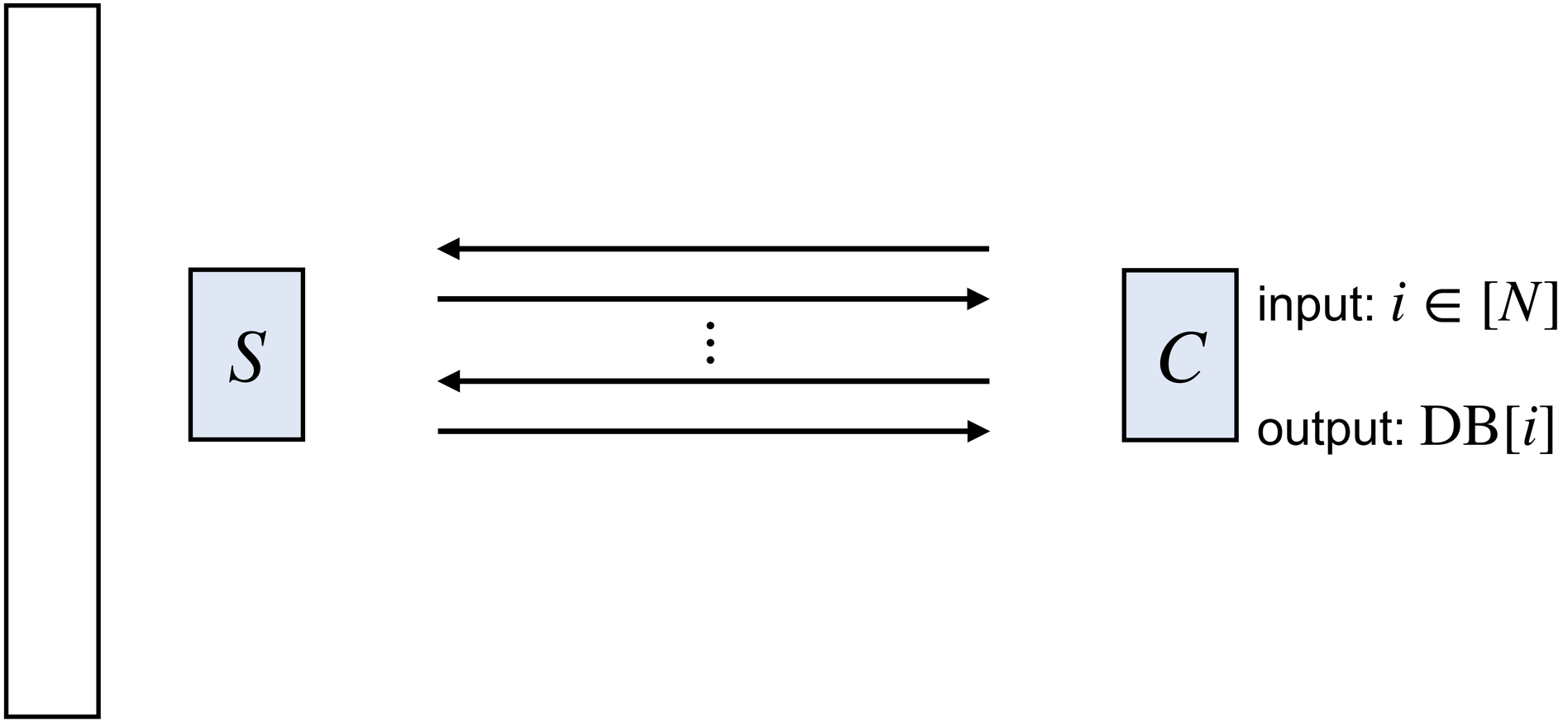
Lower bound: Server must touch ***all*** locations in $\mathrm{DB} \Rightarrow$ time $\Omega(N)$

Communication: $o(N)$

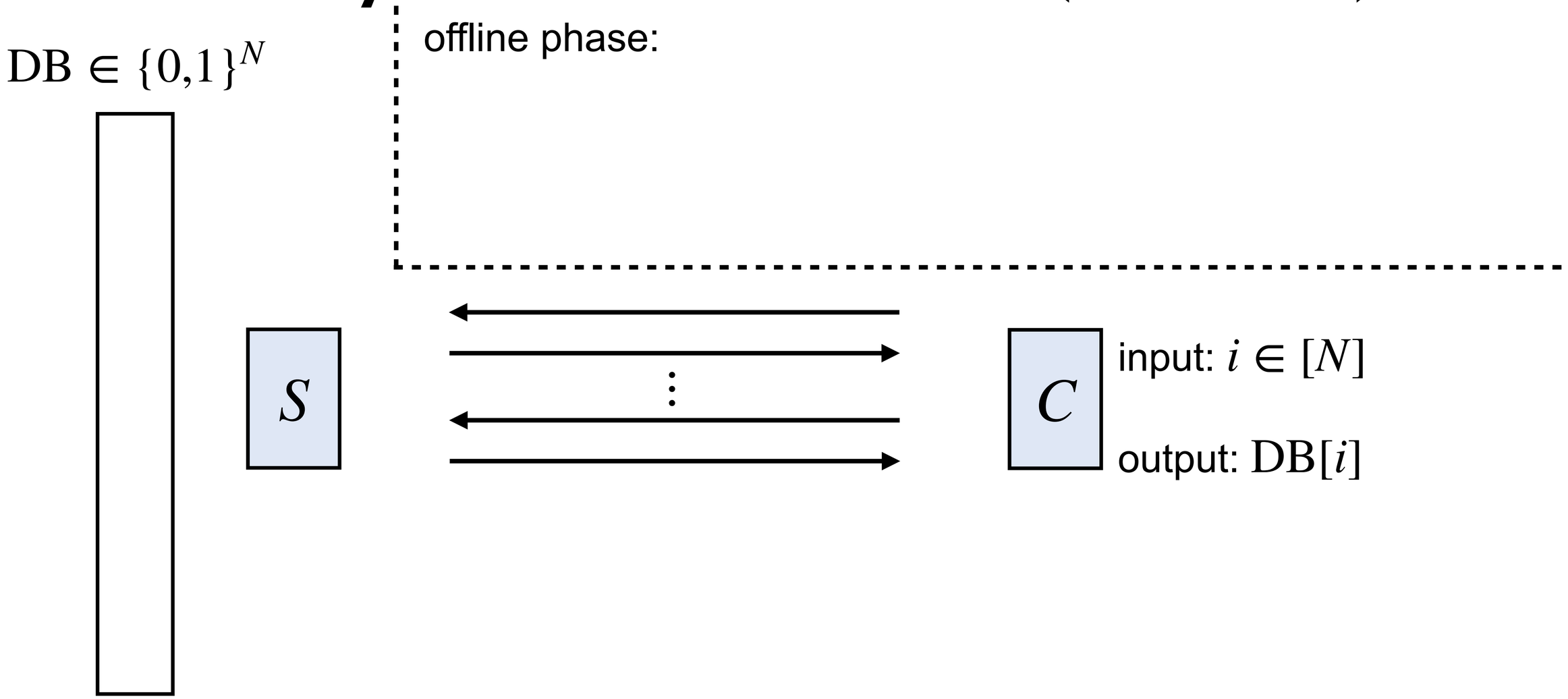
Security: Server learns nothing about $i$

# Doubly Efficient PIR (DEPIR)

$\mathrm{DB} \in \{0,1\}^N$



# Doubly Efficient PIR (DEPIR)

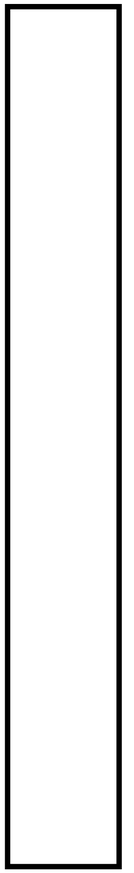


# Doubly Efficient PIR (DEPIR)

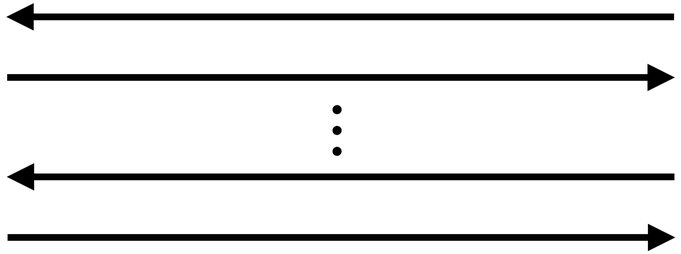
$\mathrm{DB} \in \{0,1\}^N$

offline phase: $k \leftarrow \mathsf{KeyGen}(1^\lambda)$

$k$ DB-ind. and static



$S$



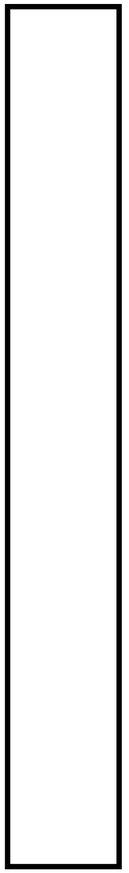
$C$

input: $i \in [N], k$

output: $\mathrm{DB}[i]$

# Doubly Efficient PIR (DEPIR)

$\mathrm{DB} \in \{0,1\}^N$



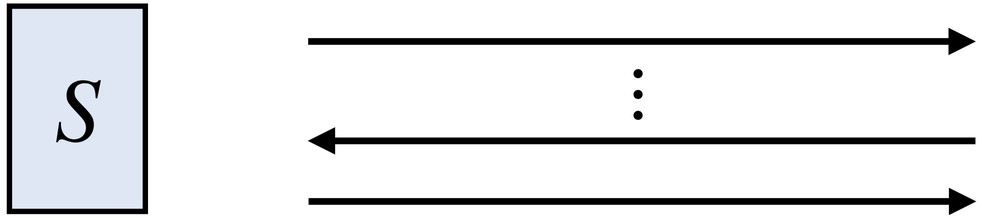
offline phase:

$k \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, k)$

$k$ DB-ind. and static

DB one-time preprocessed




input: $i \in [N], k$

output: $\mathrm{DB}[i]$

# Doubly Efficient PIR (DEPIR)

$\widetilde{\mathrm{DB}} \in \{0,1\}^{\mathrm{poly}(N)}$

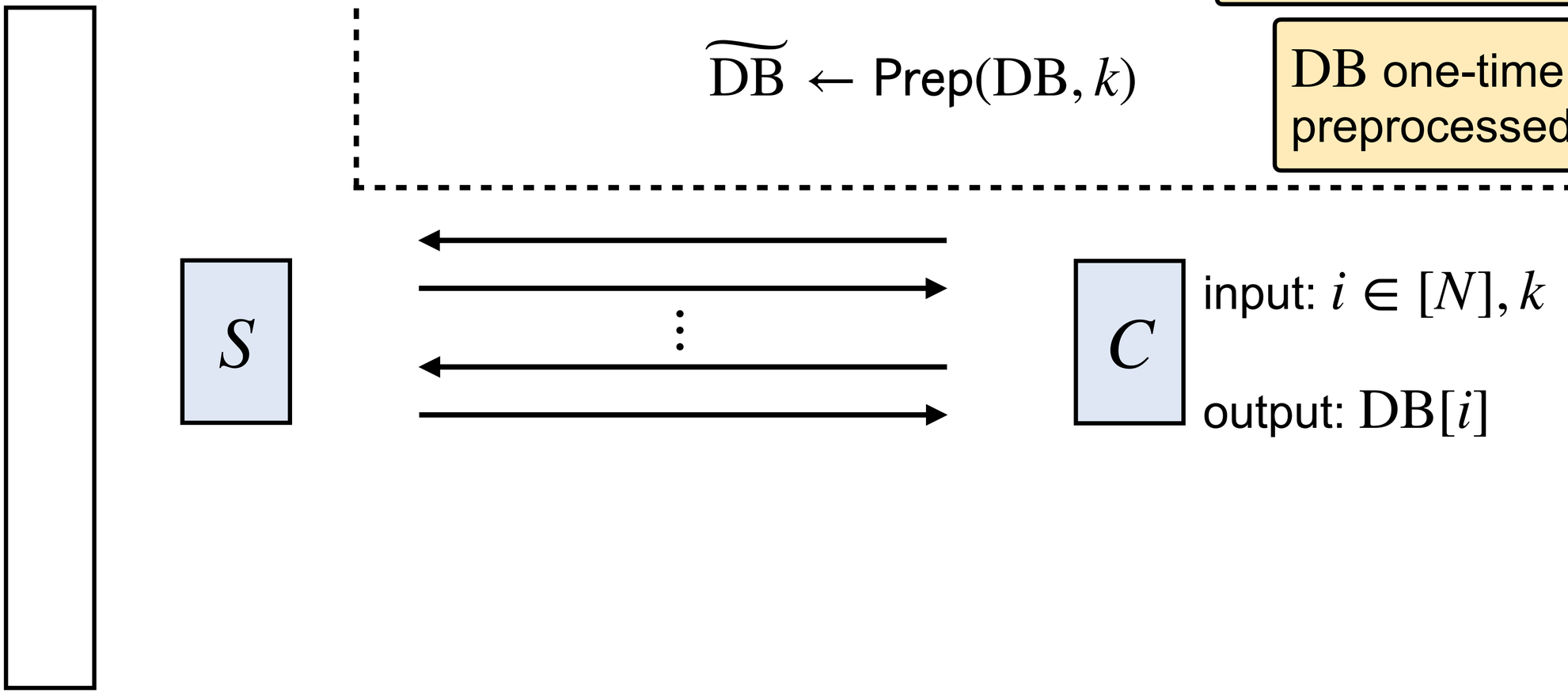
offline phase:

$k \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, k)$

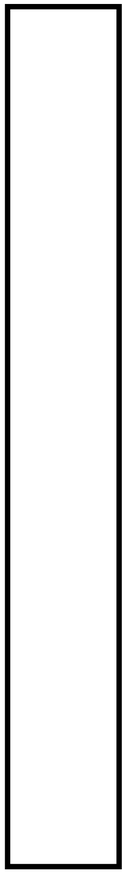
$k$ DB-ind. and static

DB one-time preprocessed



# Doubly Efficient PIR (DEPIR)

$\widetilde{\mathrm{DB}} \in \{0,1\}^{\mathrm{poly}(N)}$



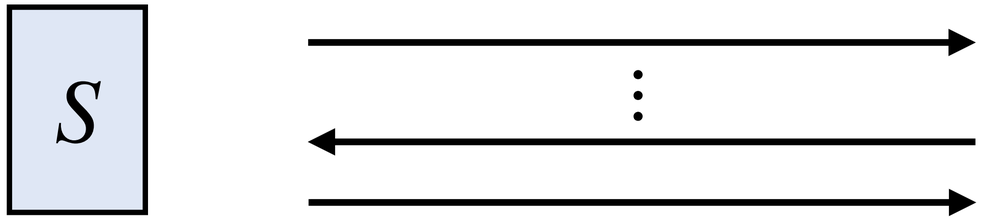
offline phase:

$k \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, k)$

$k$ DB-ind. and static

$\mathrm{DB}$ one-time preprocessed




input: $i \in [N], k$

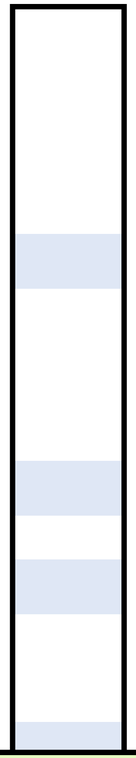
output: $\mathrm{DB}[i]$

Communication: $o(N)$

Security: Server learns nothing about $i$

# Doubly Efficient PIR (DEPIR)

$\widetilde{\mathrm{DB}} \in \{0,1\}^{\mathrm{poly}(N)}$



offline phase:

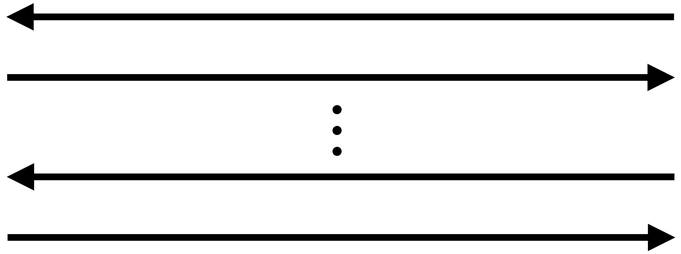
$k \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, k)$

$k$ DB-ind. and static

DB one-time preprocessed

$S$



$C$

input: $i \in [N], k$

output: $\mathrm{DB}[i]$

Efficiency: Server touches $o(N)$ locations

Communication: $o(N)$

Security: Server learns nothing about $i$

# Three Flavors of DEPIR

Prior constructions

Secret-Key
(SK-DEPIR)

---

Public-Key
(PK-DEPIR)

---

Unkeyed
(UK-DEPIR)

# Three Flavors of DEPIR

Prior constructions

Secret-Key
(SK-DEPIR)

$sk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, sk)$

$C$

---

Public-Key
(PK-DEPIR)

---

Unkeyed
(UK-DEPIR)

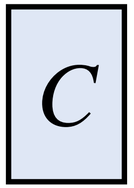
# Three Flavors of DEPIR

Prior constructions

Secret-Key
(SK-DEPIR)

$sk \leftarrow \mathrm{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathrm{Prep}(\mathrm{DB}, sk)$




---

Public-Key
(PK-DEPIR)

$pk \leftarrow \mathrm{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathrm{Prep}(\mathrm{DB}, pk)$

---

Unkeyed
(UK-DEPIR)

# Three Flavors of DEPIR

Prior constructions

Secret-Key
(SK-DEPIR)

$sk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, sk)$

$C$

---

Public-Key
(PK-DEPIR)

$pk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, pk)$

---

Unkeyed
(UK-DEPIR)

$\widetilde{\mathrm{DB}} := \mathsf{Prep}(\mathrm{DB})$

$S$

# Three Flavors of DEPIR

Prior constructions

Secret-Key
(SK-DEPIR)

$sk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, sk)$

$C$

[CHR'17,BIPW'17]: From "permuted codes with noise"

---

Public-Key
(PK-DEPIR)

$pk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, pk)$

---

Unkeyed
(UK-DEPIR)

$\widetilde{\mathrm{DB}} := \mathsf{Prep}(\mathrm{DB})$

$S$

# Three Flavors of DEPIR

Prior constructions

Secret-Key
(SK-DEPIR)

$sk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, sk)$

$C$

[CHR'17,BIPW'17]: From "permuted codes with noise"

---

Public-Key
(PK-DEPIR)

$pk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, pk)$

[CHR'17,BIPW'17]: From SK-DEPIR + heuristic obf.

---

Unkeyed
(UK-DEPIR)

$\widetilde{\mathrm{DB}} := \mathsf{Prep}(\mathrm{DB})$

$S$

# Three Flavors of DEPIR

Prior constructions

Secret-Key
(SK-DEPIR)

$sk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, sk)$

$C$

[CHR'17,BIPW'17]: From "permuted codes with noise"

---

Public-Key
(PK-DEPIR)

$pk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, pk)$

[CHR'17,BIPW'17]: From SK-DEPIR + heuristic obf.

---

Unkeyed
(UK-DEPIR)

$\widetilde{\mathrm{DB}} := \mathsf{Prep}(\mathrm{DB})$

$S$

[LMW'23]: From RingLWE

# Three Flavors of DEPIR

Prior constructions

Secret-Key
(SK-DEPIR)

$sk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, sk)$

$C$

[CHR'17,BIPW'17]: From "permuted codes with noise"

⇑

Public-Key
(PK-DEPIR)

$pk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, pk)$

[CHR'17,BIPW'17]: From SK-DEPIR + heuristic obf.

⇑

Unkeyed
(UK-DEPIR)

$\widetilde{\mathrm{DB}} := \mathsf{Prep}(\mathrm{DB})$

$S$

[L**M**W'23]: From RingLWE

# Three Flavors of DEPIR

Prior constructions

Secret-Key
(SK-DEPIR)

$sk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, sk)$

$C$

[CHR'17,BIPW'17]: From "permuted codes with noise"

⇑

Public-Key
(PK-DEPIR)

$pk \leftarrow \mathsf{KeyGen}(1^\lambda)$

$\widetilde{\mathrm{DB}} \leftarrow \mathsf{Prep}(\mathrm{DB}, pk)$

[CHR'17,BIPW'17]: From SK-DEPIR + heuristic obf.

⇑

Unkeyed
(UK-DEPIR)

$\widetilde{\mathrm{DB}} := \mathsf{Prep}(\mathrm{DB})$

$S$

[LMW'23]: From RingLWE

⇓

(Standard) PIR

# How does (SK,PK,UK)-DEPIR relate to other crypto primitives?

How does (**SK**,PK,UK)-DEPIR relate to other crypto primitives?

# Our Results

Black-box use of advanced crypto is ***useless*** for constructing DEPIR

# Our Results

Black-box use of advanced crypto is ***useless*** for constructing DEPIR

**Theorem 1:** Let $P$ be a crypto primitive that *large class (to be specified)*

# Our Results

Black-box use of advanced crypto is ***useless*** for constructing DEPIR

**<u>Theorem 1</u>:** Let $P$ be a crypto primitive that *large class (to be specified)*

- ***If:*** there is a BB construction of SK-DEPIR from $P$

# Our Results

Black-box use of advanced crypto is ***useless*** for constructing DEPIR

**Theorem 1**: Let $P$ be a crypto primitive that *large class (to be specified)*

- ***If:*** there is a BB construction of SK-DEPIR from $P$
- ***Then:*** there is a BB construction of SK-DEPIR from OWFs

# Our Results

Black-box use of advanced crypto is ***useless*** for constructing DEPIR

**Theorem 1**: Let $P$ be a crypto primitive that *large class (to be specified)*

- ***If:*** there is a BB construction of SK-DEPIR from $P$
- ***Then:*** there is a BB construction of SK-DEPIR from OWFs

Partial progress ruling out SK-DEPIR from OWFs

# Our Results

Black-box use of advanced crypto is ***useless*** for constructing DEPIR

**Theorem 1**: Let $P$ be a crypto primitive that *large class (to be specified)*

- ***If:*** there is a BB construction of SK-DEPIR from $P$
- ***Then:*** there is a BB construction of SK-DEPIR from OWFs

Partial progress ruling out SK-DEPIR from OWFs

**Theorem 2**: There is no BB construction of ***2-round, passive server*** SK-DEPIR from OWFs

# Techniques

**Starting point:** An SK-DEPIR that exists in an idealized world

$$\text{DEPIR} = (\text{Prep}^{\mathcal{O}_P}, C^{\mathcal{O}_P}, S^{\mathcal{O}_P})$$

# Techniques

$\mathcal{O}_P$ = oracle implementing $P$

**Starting point:** An SK-DEPIR that exists in an idealized world

$$\text{DEPIR} = (\text{Prep}^{\mathcal{O}_P}, C^{\mathcal{O}_P}, S^{\mathcal{O}_P})$$

# Techniques

$\mathcal{O}_P$ = oracle implementing $P$

**Starting point:** An SK-DEPIR that exists in an idealized world

$$\text{DEPIR} = (\text{Prep}^{\mathcal{O}_P}, C^{\mathcal{O}_P}, S^{\mathcal{O}_P})$$

**Key insight:** No hiding property from the client
⇒ client can make the server's oracle calls for it

# Techniques

$\mathcal{O}_P$ = oracle implementing $P$

**Starting point:** An SK-DEPIR that exists in an idealized world

$$\text{DEPIR} = (\text{Prep}^{\mathcal{O}_P}, C^{\mathcal{O}_P}, S^{\mathcal{O}_P})$$

**Key insight:** No hiding property from the client
⇒ client can make the server's oracle calls for it

Inspired by techniques of [Dujmovic-Hajiabadi'24]

# Techniques

$\mathcal{O}_P$ = oracle implementing $P$

**Starting point:** An SK-DEPIR that exists in an idealized world

$$\text{DEPIR} = (\text{Prep}^{\mathcal{O}_P}, C^{\mathcal{O}_P}, S^{\mathcal{O}_P})$$

**Key insight:** No hiding property from the client
⇒ client can make the server's oracle calls for it

Inspired by techniques of [Dujmovic-Hajiabadi'24]

Compile into an SK-DEPIR where the server doesn't use the oracle

$$\widetilde{\text{DEPIR}} = (\text{Prep}^{\mathcal{O}_P}, \tilde{C}^{\mathcal{O}_P}, \tilde{S})$$

# Techniques

$\mathcal{O}_P$ = oracle implementing $P$

**Starting point:** An SK-DEPIR that exists in an idealized world

$$\text{DEPIR} = (\text{Prep}^{\mathcal{O}_P}, C^{\mathcal{O}_P}, S^{\mathcal{O}_P})$$

**Key insight:** No hiding property from the client
⇒ client can make the server's oracle calls for it

Inspired by techniques of [Dujmovic-Hajiabadi'24]

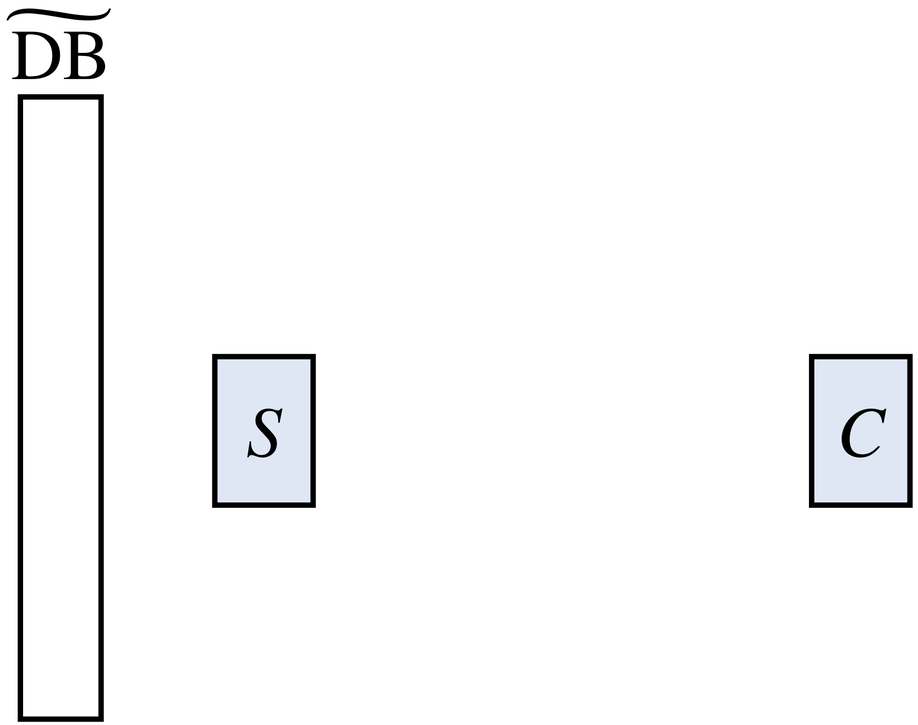
Compile into an SK-DEPIR where the server doesn't use the oracle

$$\widetilde{\text{DEPIR}} = (\text{Prep}^{\mathcal{O}_P}, \tilde{C}^{\mathcal{O}_P}, \tilde{S})$$

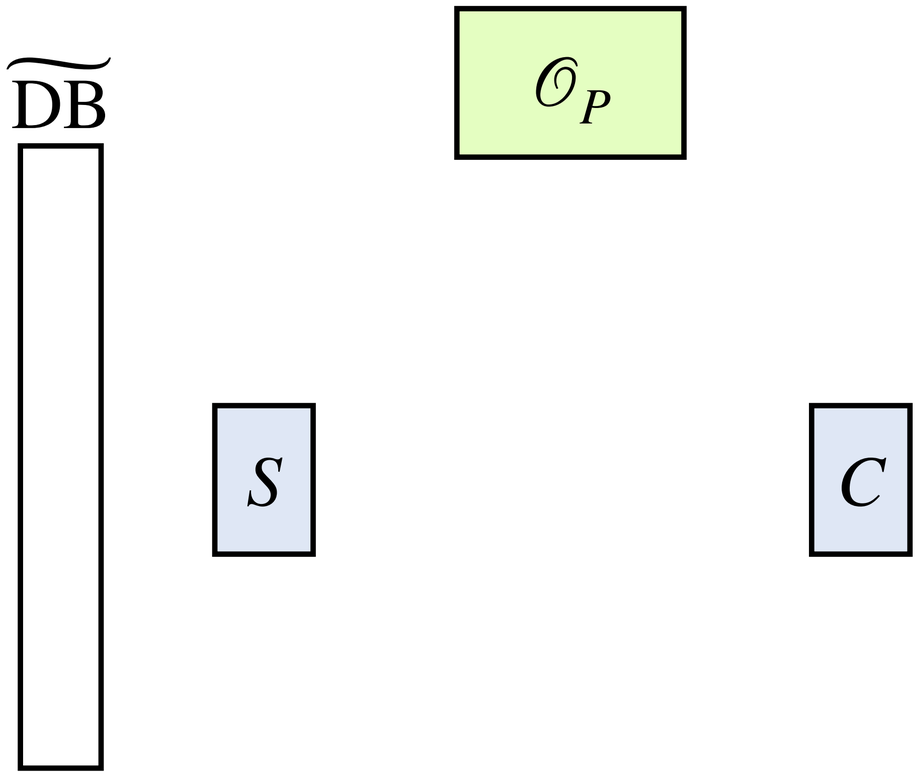
If $\mathcal{O}_P$ is *"nice"*, then the client can emulate $\mathcal{O}_P$ with just a PRF

$$\overline{\text{DEPIR}} = (\overline{\text{Prep}}^{\text{PRF}}, \overline{C}^{\text{PRF}}, \tilde{S})$$

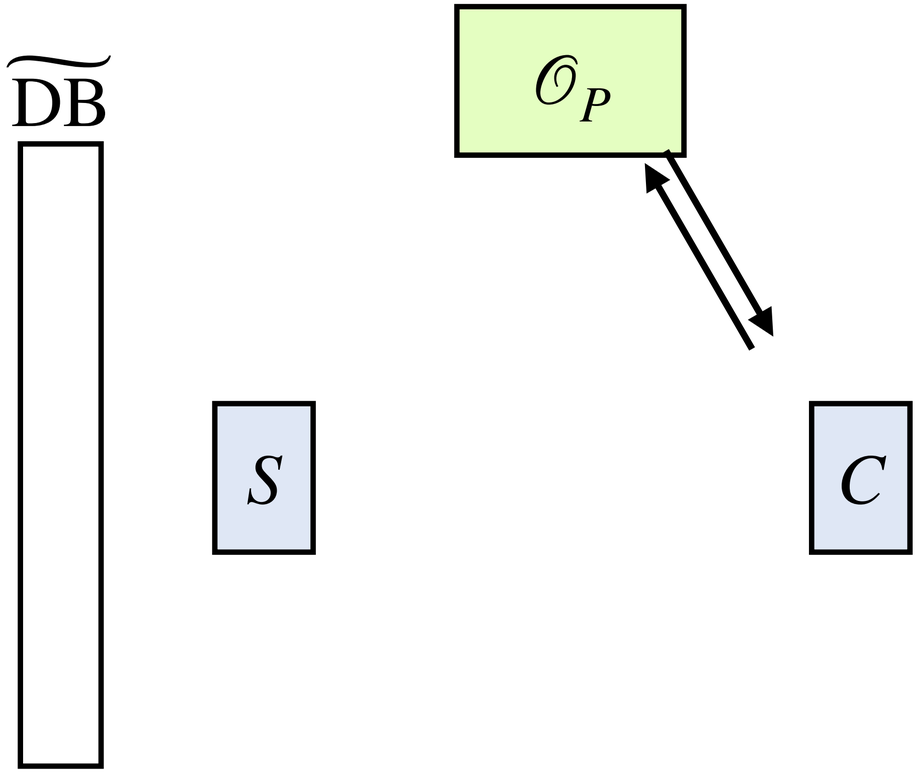
# Removing the Server's Oracle



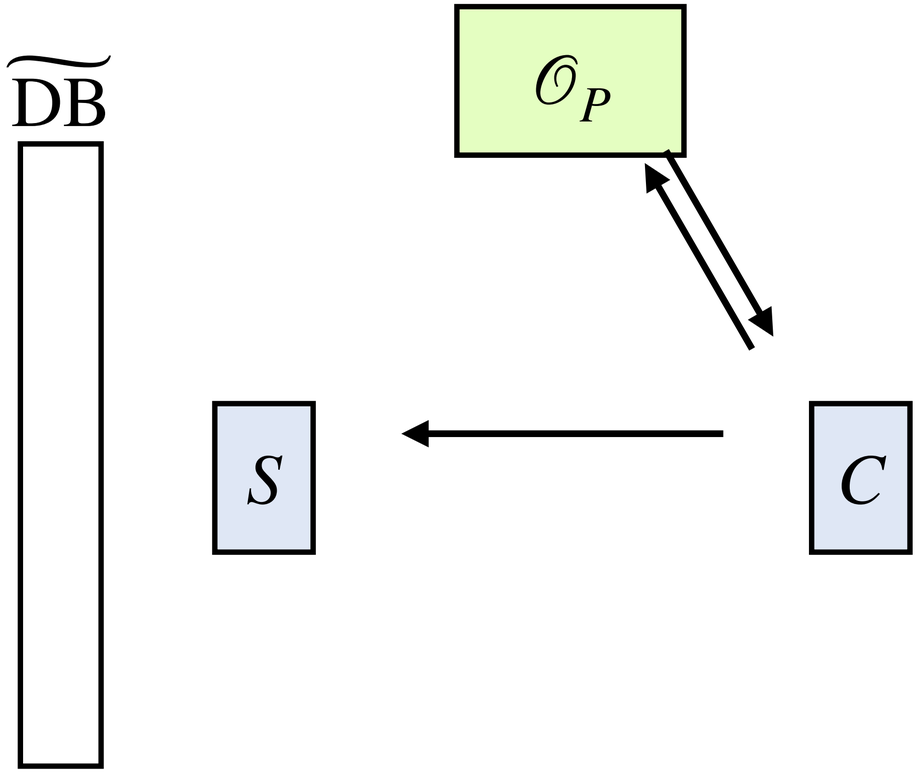
# Removing the Server's Oracle



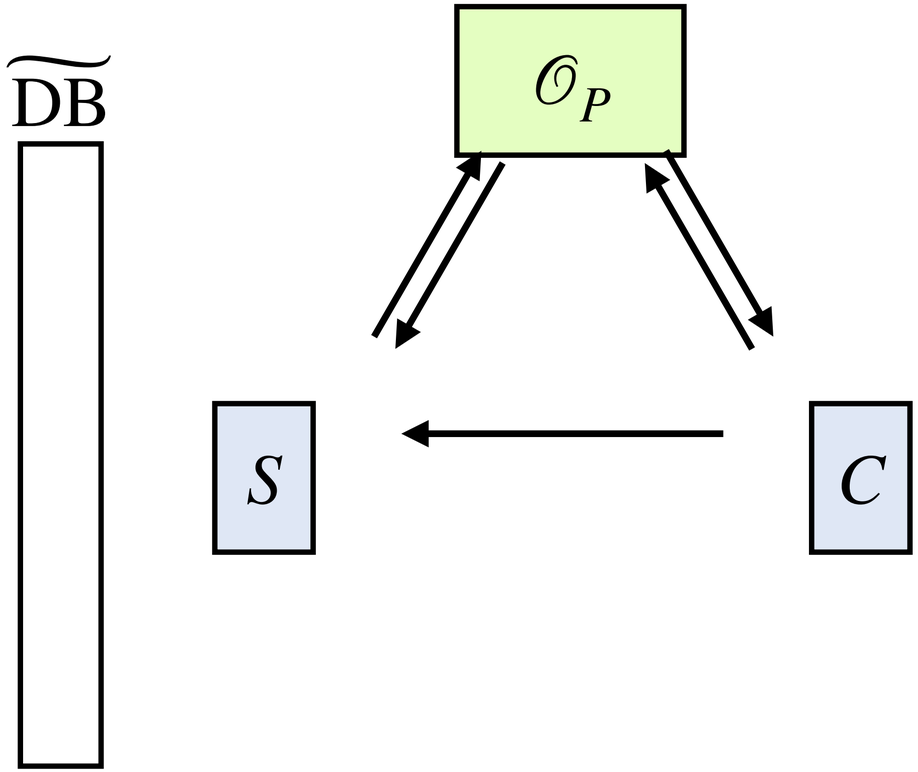
# Removing the Server's Oracle



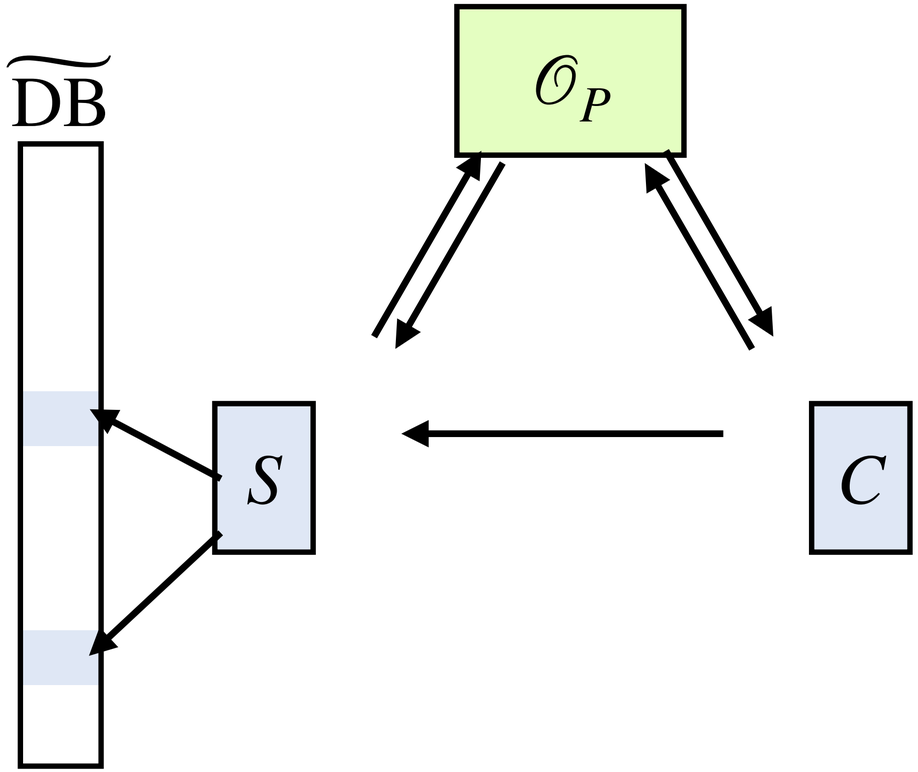
# Removing the Server's Oracle



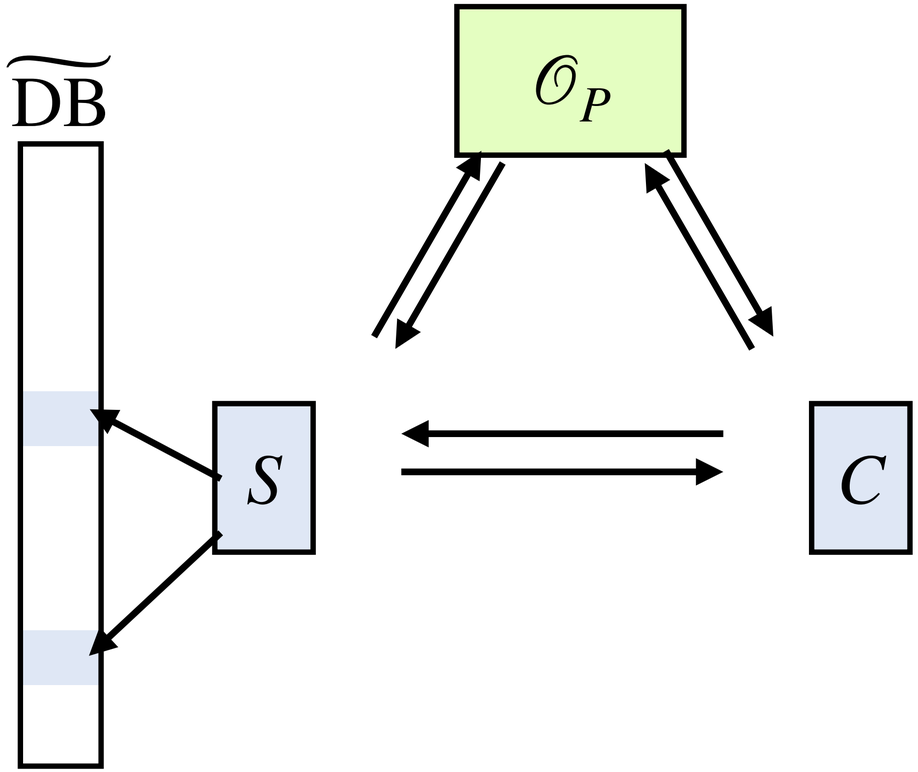
# Removing the Server's Oracle



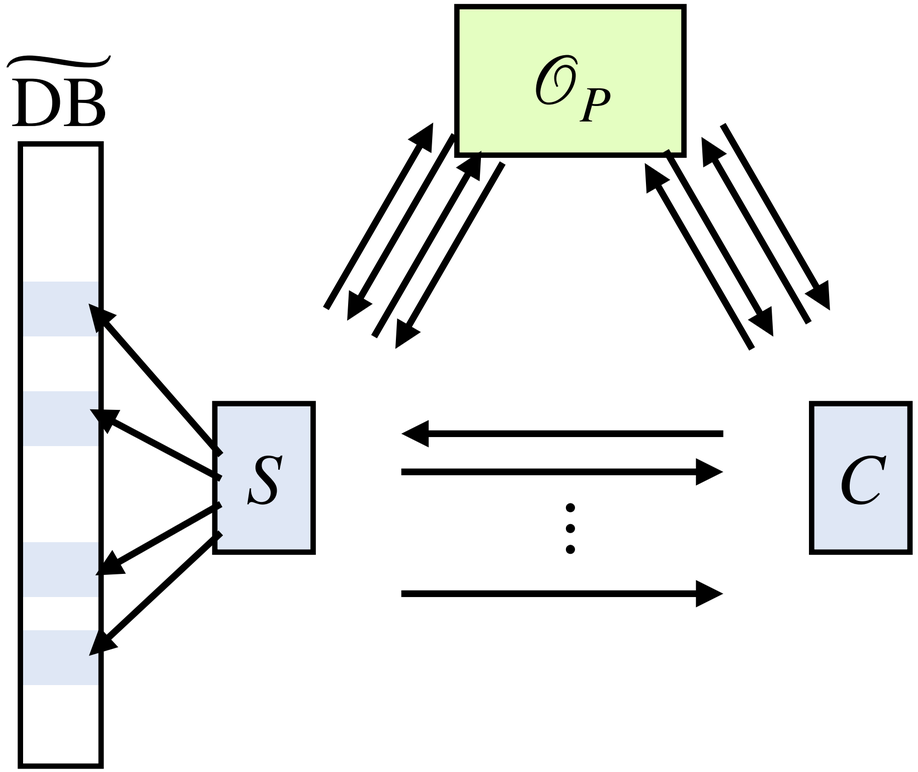
# Removing the Server's Oracle



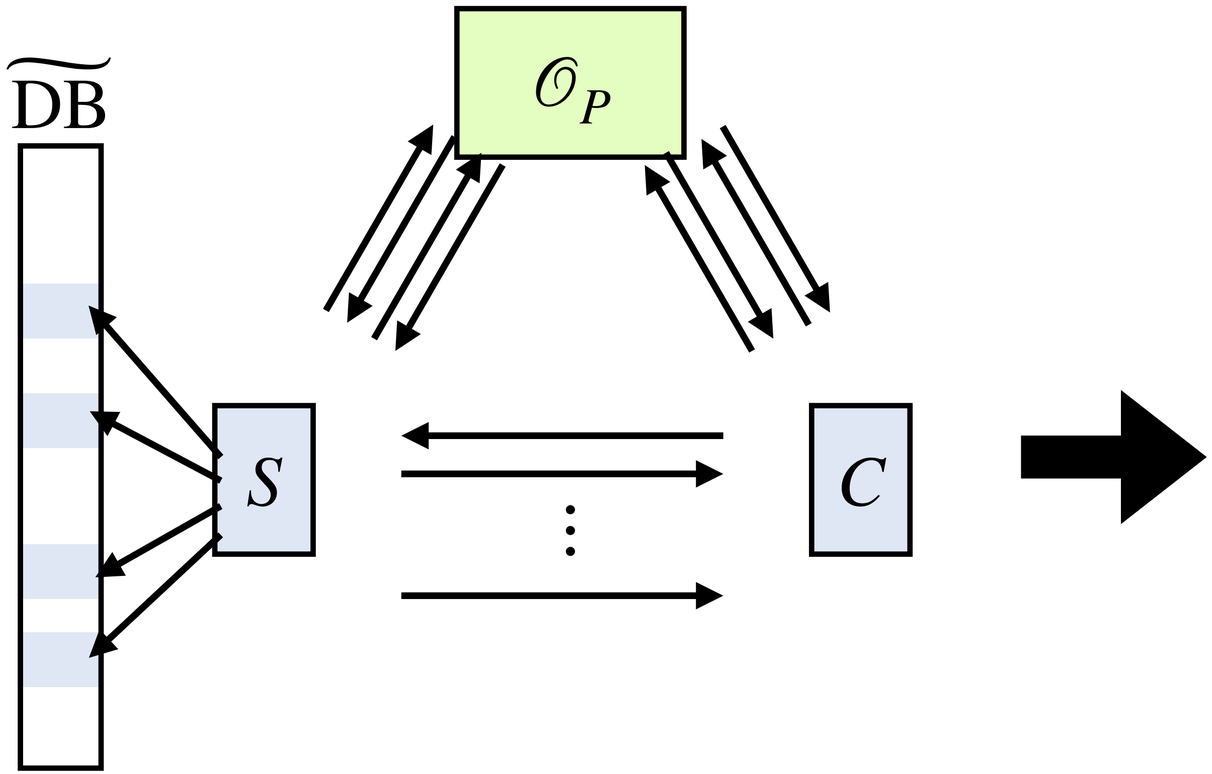
# Removing the Server's Oracle



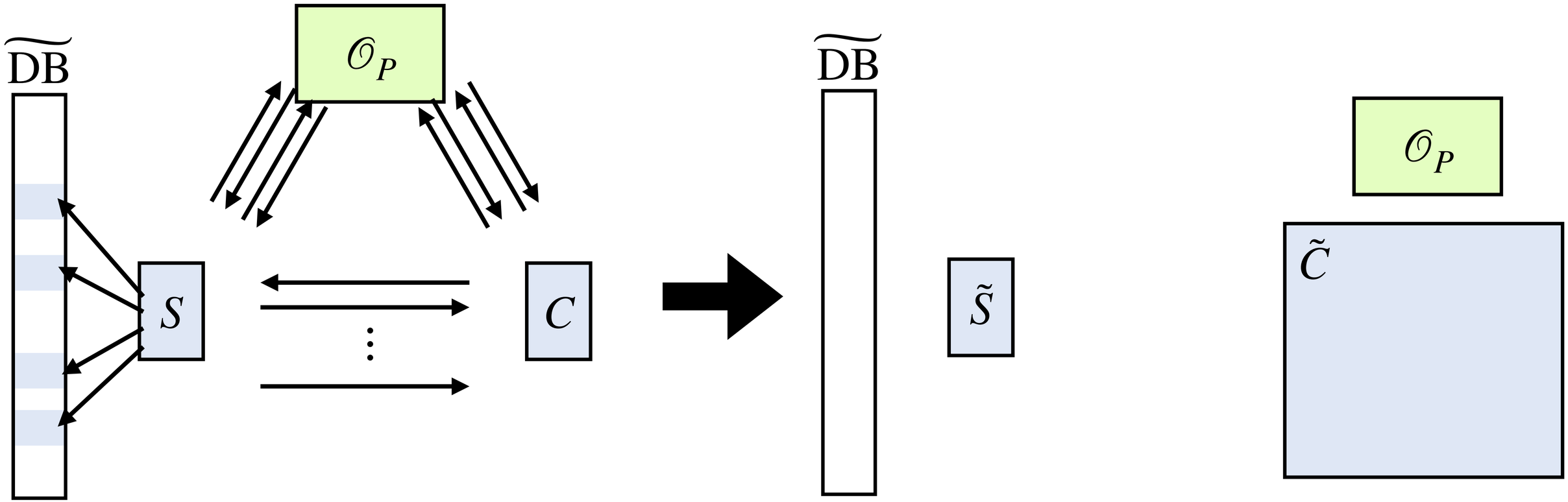
# Removing the Server's Oracle



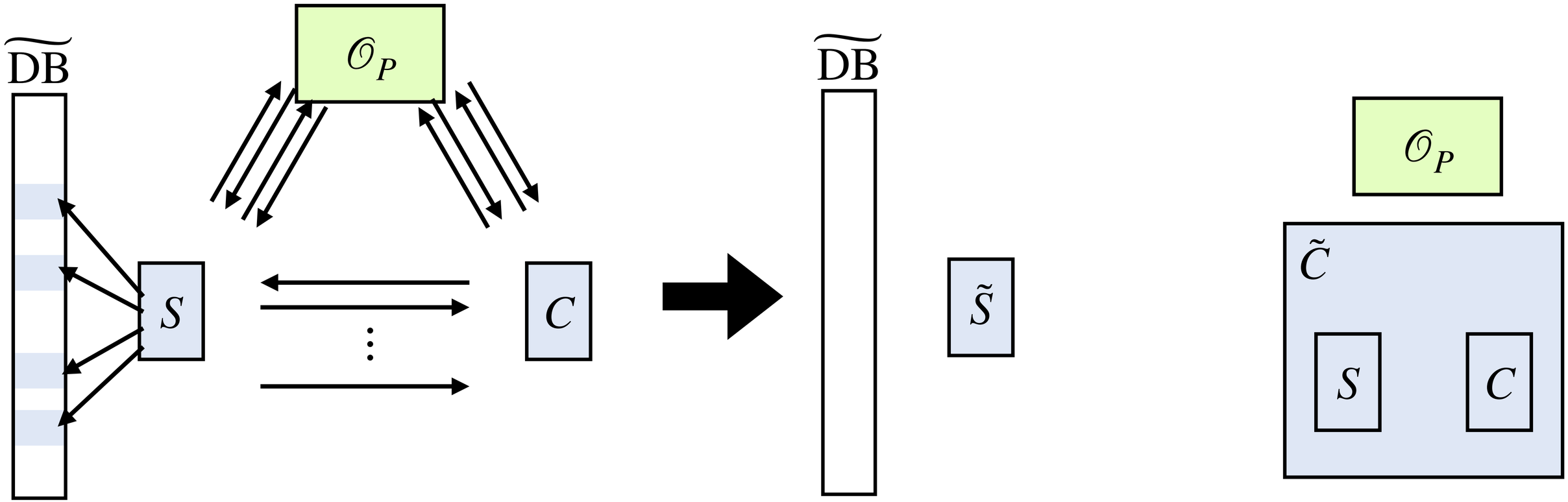
# Removing the Server's Oracle



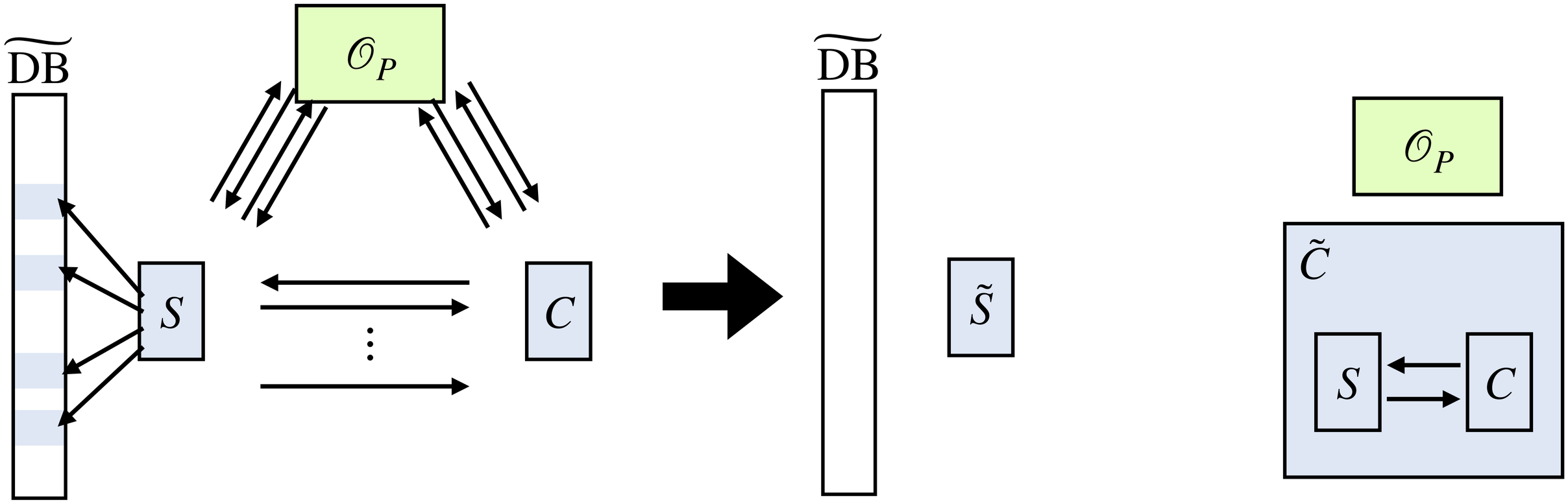
# Removing the Server's Oracle



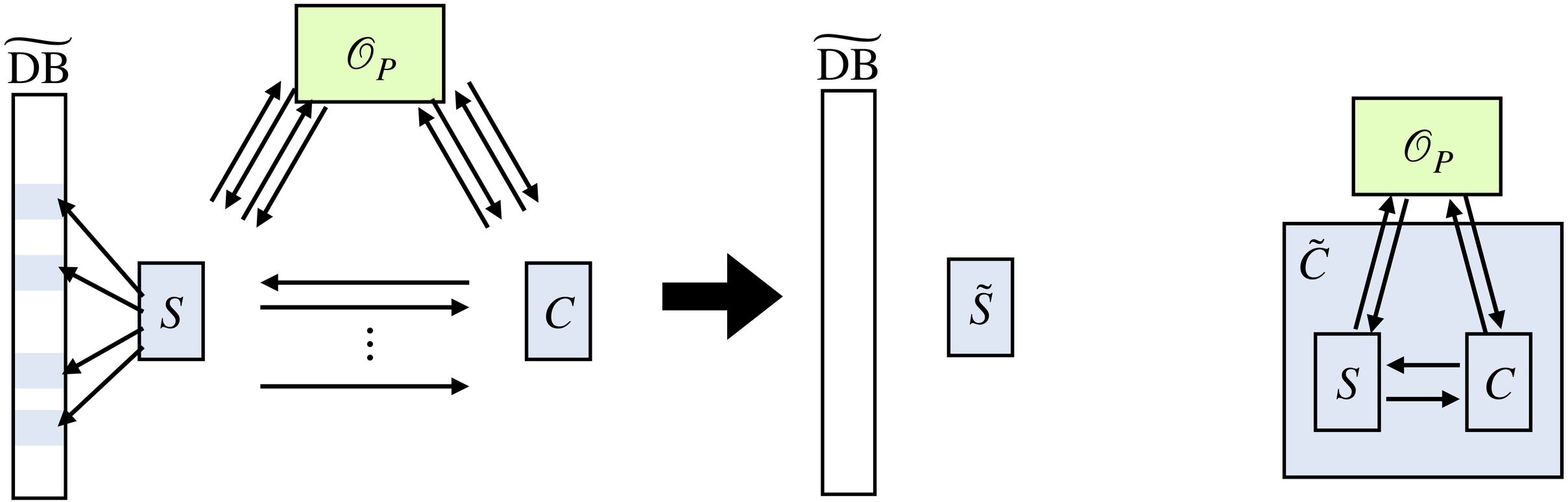
# Removing the Server's Oracle



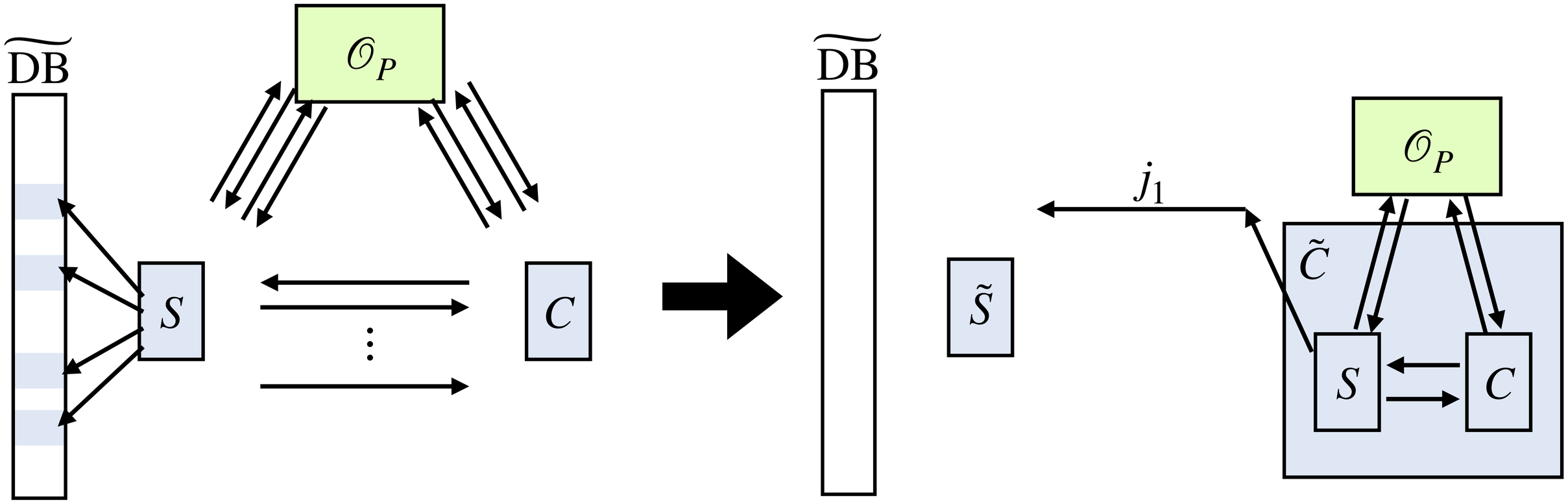
# Removing the Server's Oracle



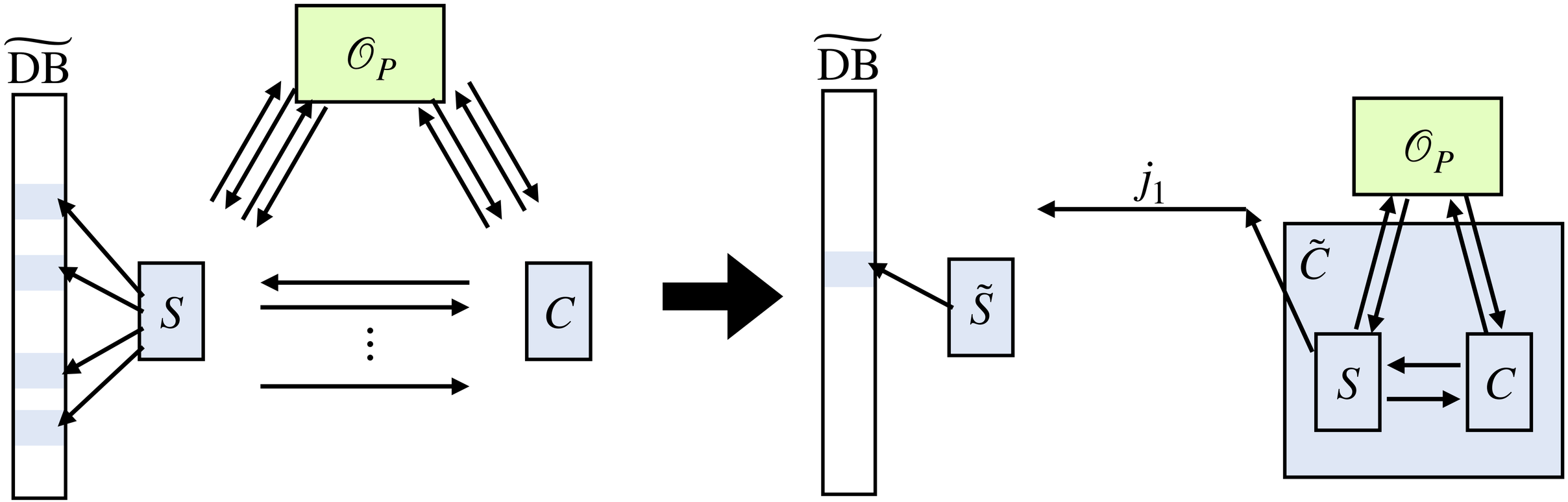
# Removing the Server's Oracle



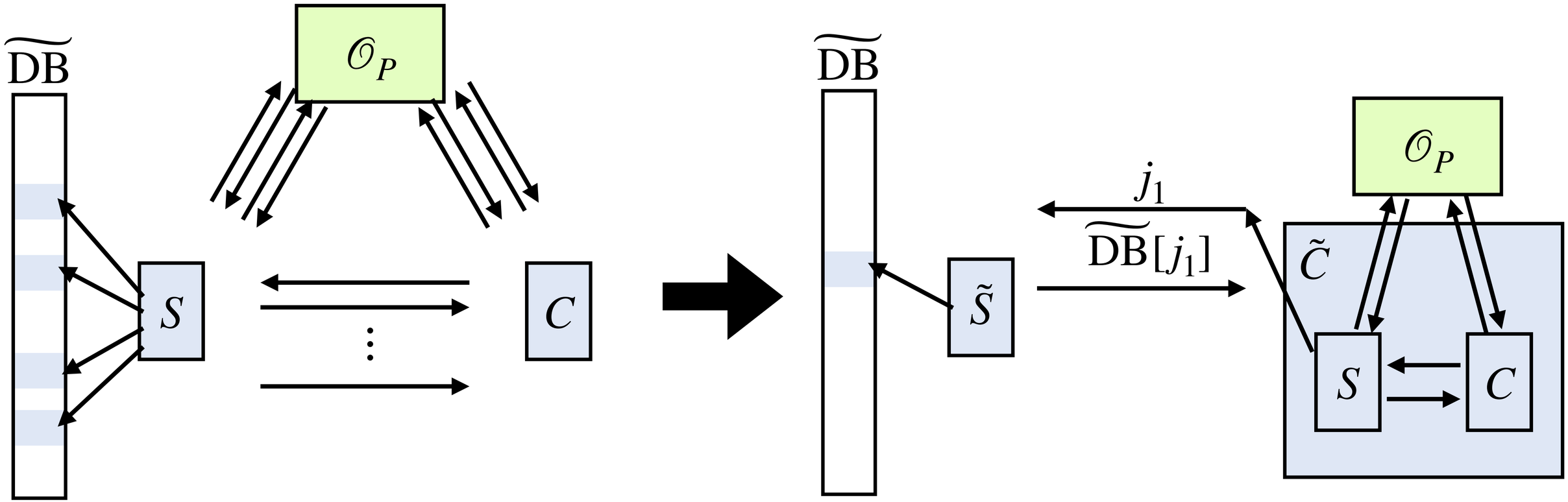
# Removing the Server's Oracle



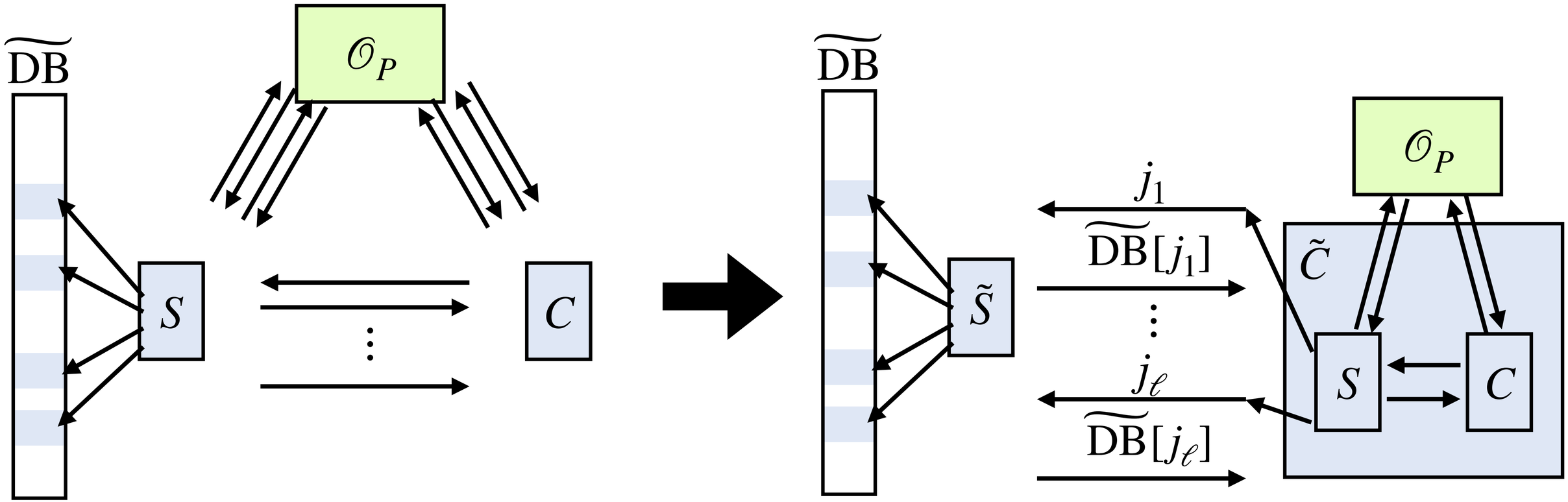
# Removing the Server's Oracle



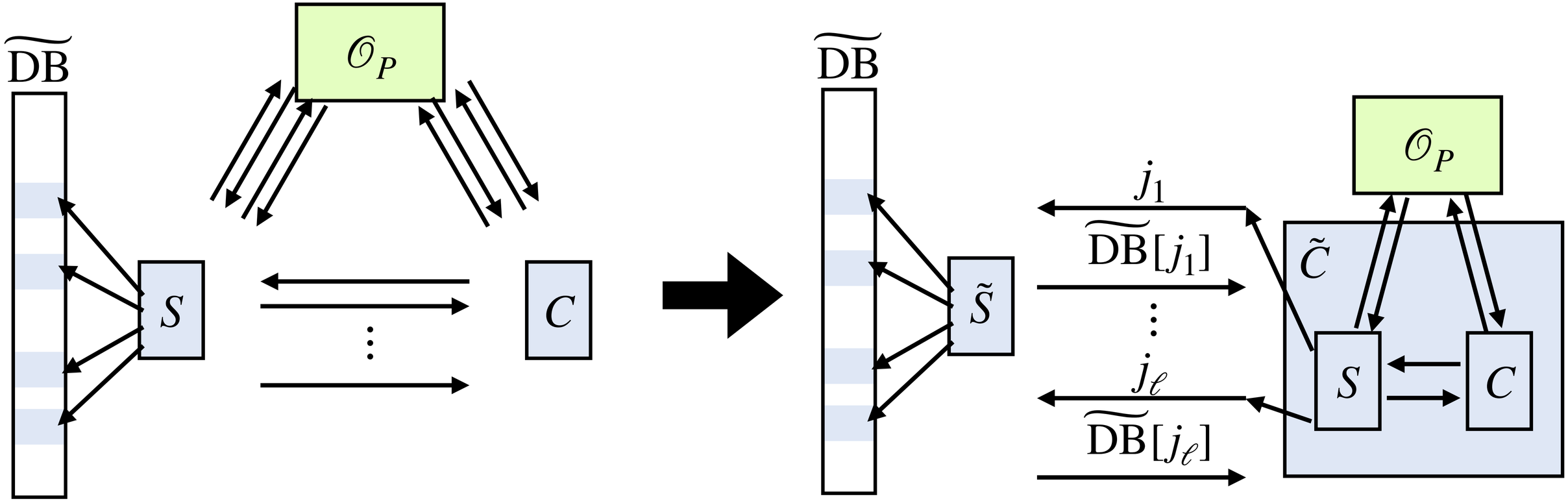
# Removing the Server's Oracle



# Removing the Server's Oracle

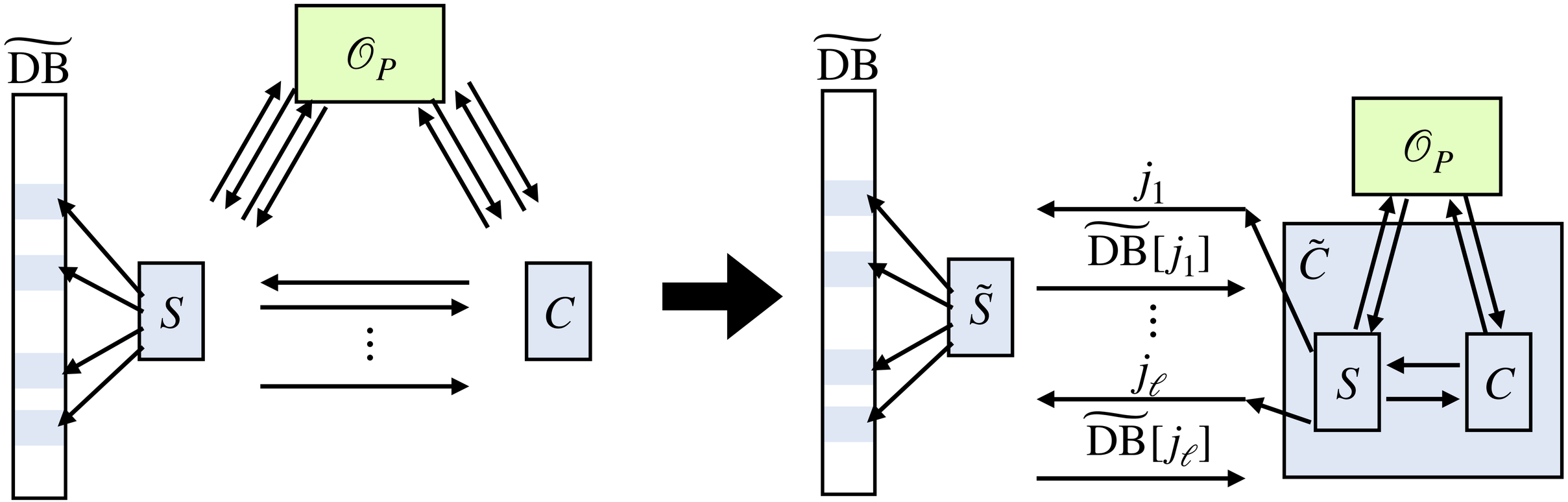


# Removing the Server's Oracle




Communication & $\tilde{S}$ efficiency:
$O(\#\text{ locations read}) = o(N)$

# Removing the Server's Oracle




$\tilde{S}$ is purely passive

Communication & $\tilde{S}$ efficiency:
$O(\#\text{ locations read}) = o(N)$

# Removing the Oracle Altogether

# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

- $B$ is a poly time (stateless, deterministic) Turing machine

# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

- $B$ is a poly time (stateless, deterministic) Turing machine
- $R$ is a private random function available only to $B$

# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

- $B$ is a poly time (stateless, deterministic) Turing machine
- $R$ is a private random function available only to $B$

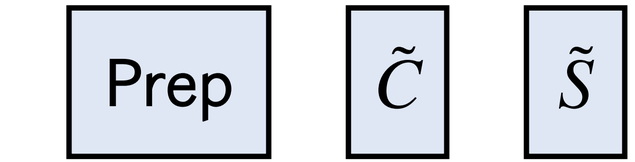
Suppose $\mathcal{O}_P = B^R$ is a crypto oracle

# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

- $B$ is a poly time (stateless, deterministic) Turing machine
- $R$ is a private random function available only to $B$

Suppose $\mathcal{O}_P = B^R$ is a crypto oracle



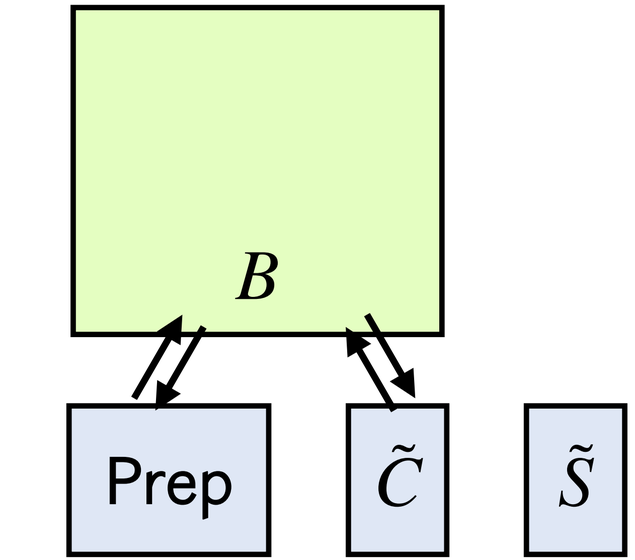

Passive server SK-DEPIR

# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

- $B$ is a poly time (stateless, deterministic) Turing machine
- $R$ is a private random function available only to $B$

Suppose $\mathcal{O}_P = B^R$ is a crypto oracle



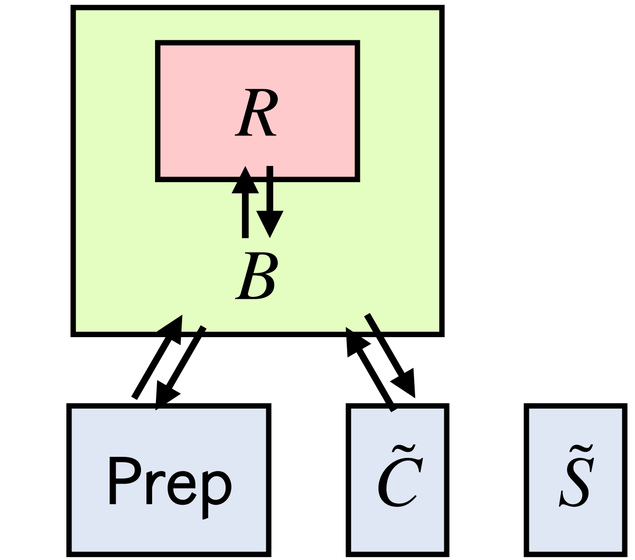

Passive server SK-DEPIR

# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

- $B$ is a poly time (stateless, deterministic) Turing machine
- $R$ is a private random function available only to $B$

Suppose $\mathcal{O}_P = B^R$ is a crypto oracle



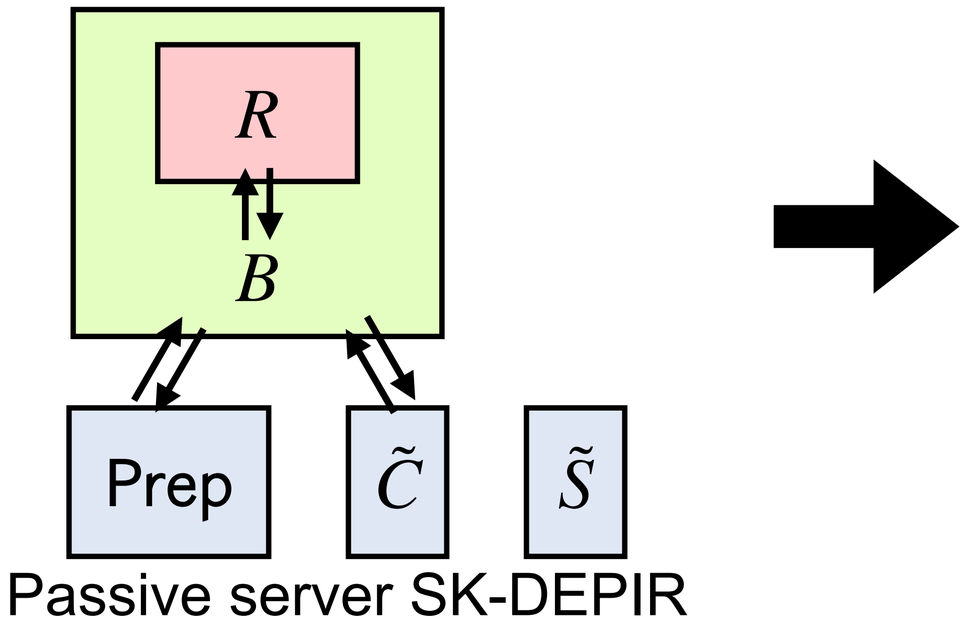

Passive server SK-DEPIR

# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

- $B$ is a poly time (stateless, deterministic) Turing machine
- $R$ is a private random function available only to $B$

Suppose $\mathcal{O}_P = B^R$ is a crypto oracle

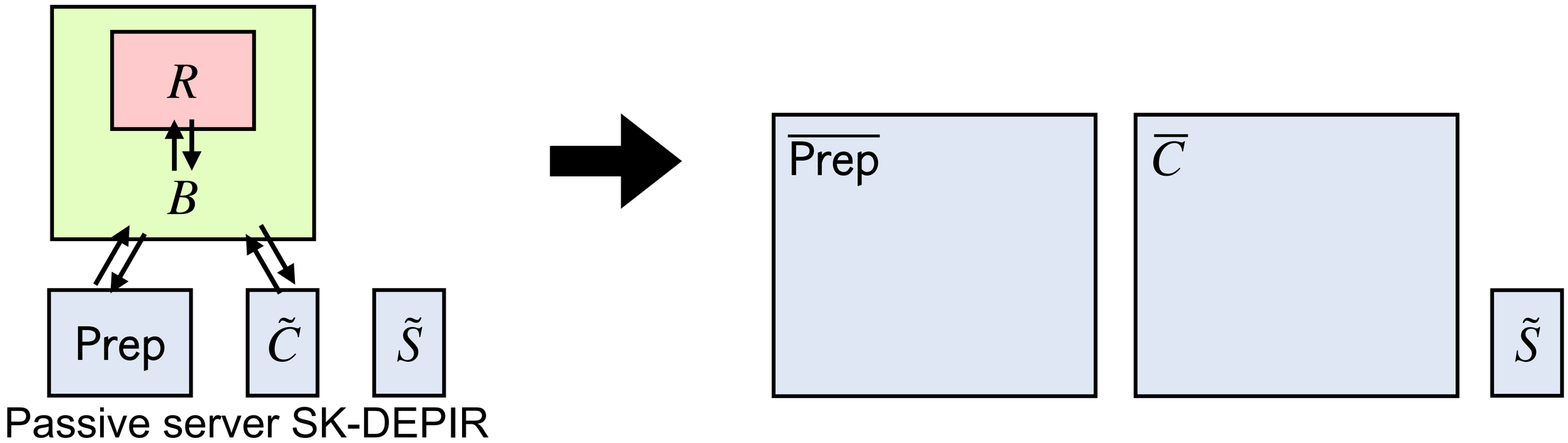


# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

- $B$ is a poly time (stateless, deterministic) Turing machine
- $R$ is a private random function available only to $B$

Suppose $\mathcal{O}_P = B^R$ is a crypto oracle

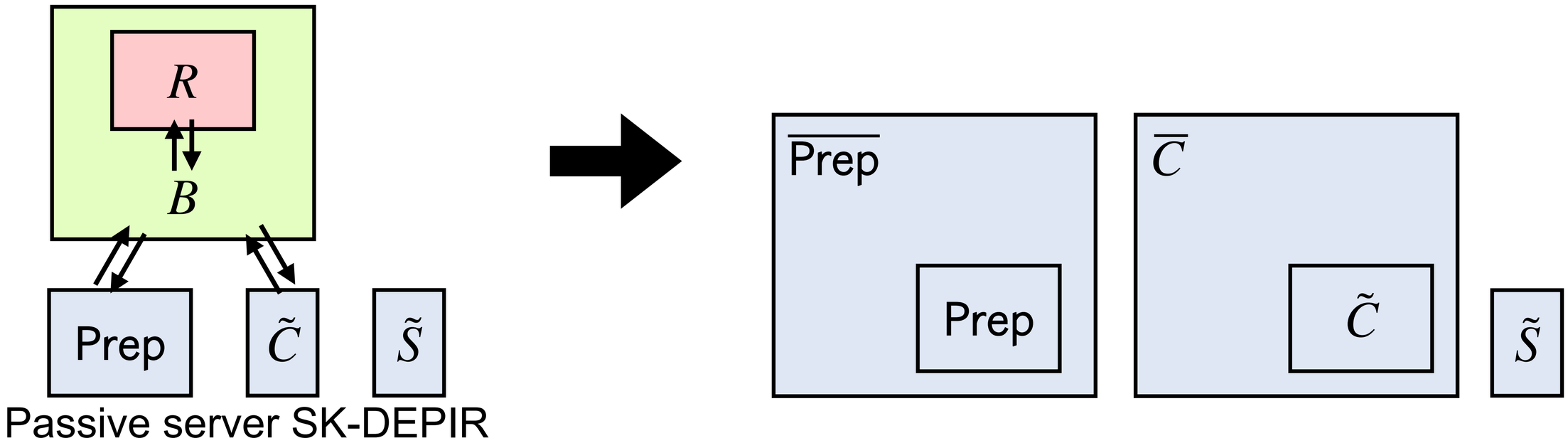


# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

- $B$ is a poly time (stateless, deterministic) Turing machine
- $R$ is a private random function available only to $B$

Suppose $\mathcal{O}_P = B^R$ is a crypto oracle

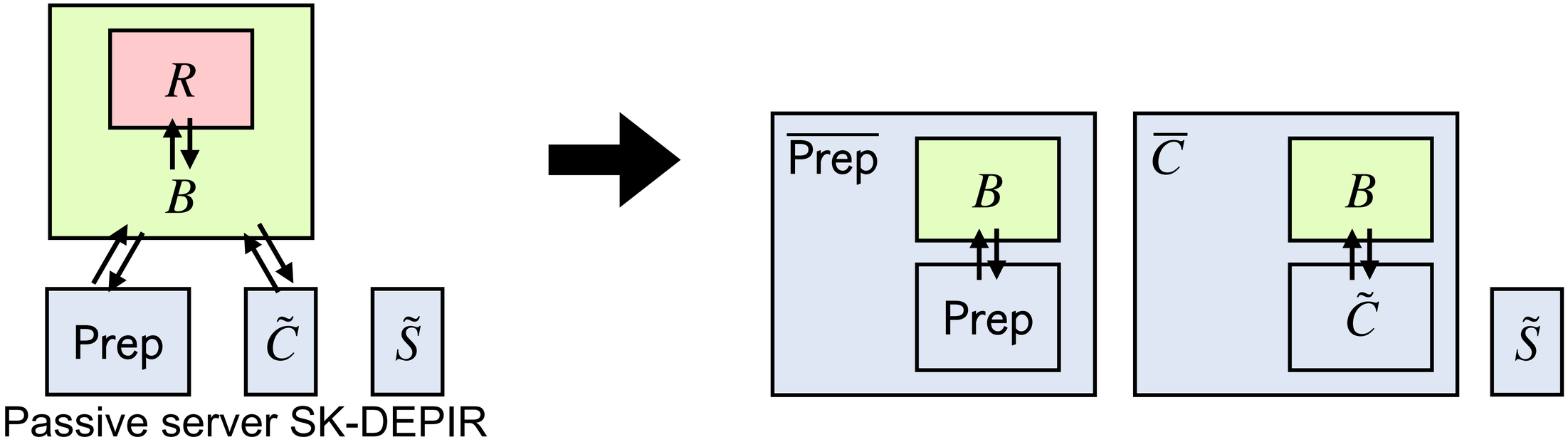


# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

- $B$ is a poly time (stateless, deterministic) Turing machine
- $R$ is a private random function available only to $B$

Suppose $\mathcal{O}_P = B^R$ is a crypto oracle

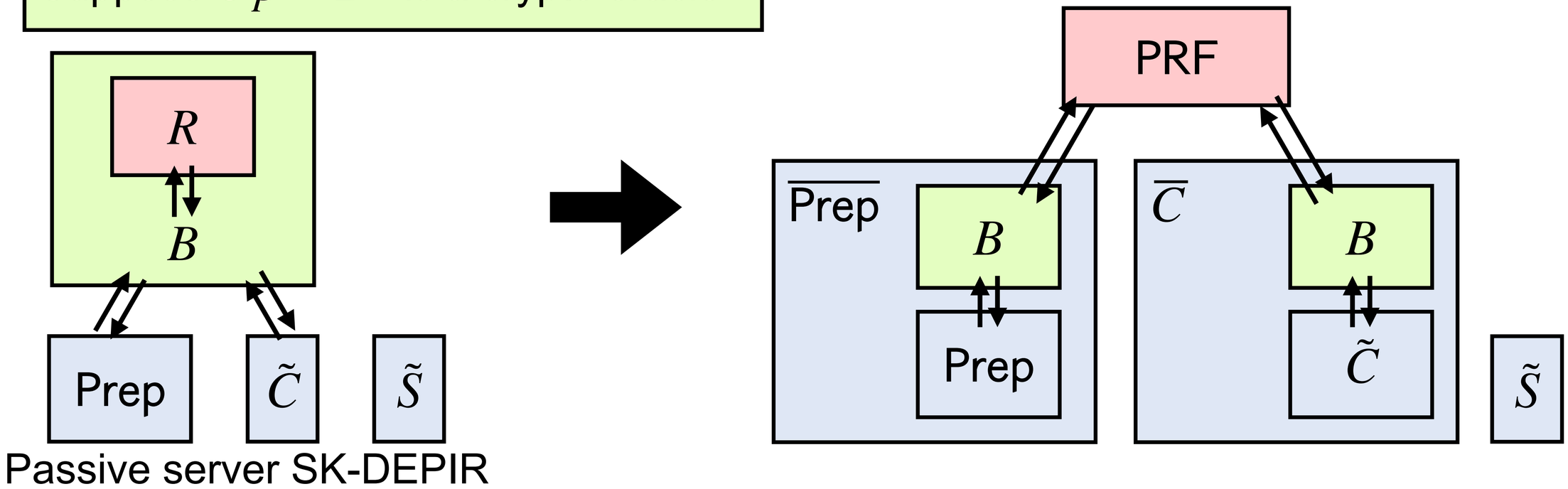


# Removing the Oracle Altogether

**Definition:** A *crypto oracle* is an oracle of the form $B^R$ where

- $B$ is a poly time (stateless, deterministic) Turing machine
- $R$ is a private random function available only to $B$

Suppose $\mathcal{O}_P = B^R$ is a crypto oracle




Passive server SK-DEPIR

# Our Results

Black-box use of advanced crypto is ***useless*** for constructing DEPIR

**Theorem 1**: Let $P$ be a crypto primitive that *large class (to be specified)*

- ***If:*** there is a BB construction of SK-DEPIR from $P$
- ***Then:*** there is a BB construction of SK-DEPIR from OWFs

Partial progress ruling out SK-DEPIR from OWFs

**Theorem 2**: There is no BB construction of ***2-round, passive server*** SK-DEPIR from OWFs

# Our Results

Black-box use of advanced crypto is ***useless*** for constructing DEPIR

**Theorem 1:** Let $P$ be a crypto primitive that exists wrt a crypto oracle

- ***If:*** there is a BB construction of SK-DEPIR from $P$
- ***Then:*** there is a BB construction of SK-DEPIR from OWFs

Partial progress ruling out SK-DEPIR from OWFs

**Theorem 2:** There is no BB construction of ***2-round, passive server*** SK-DEPIR from OWFs

# Our Results

Black-box use of advanced crypto is ***useless*** for constructing DEPIR

**Theorem 1**: Let $P$ be a crypto primitive that exists wrt a crypto oracle

- ***If:*** there is a BB construction of SK-DEPIR from $P$
- ***Then:*** there is a BB construction of SK-DEPIR from OWFs

Partial progress ruling out SK-DEPIR from OWFs

**Theorem 2**: There is no BB construction of ***2-round, passive server*** SK-DEPIR from OWFs

Adapted from [CHR'17] info. theoretic lower bound

↳see paper for details

# How Strong are Crypto Oracles?

# How Strong are Crypto Oracles?

**Lemma:** Virtual black-box obfuscation exists relative to a crypto oracle

# How Strong are Crypto Oracles?

**Lemma:** Virtual black-box obfuscation exists relative to a crypto oracle

**Proof sketch:**

# How Strong are Crypto Oracles?

**Lemma:** Virtual black-box obfuscation exists relative to a crypto oracle

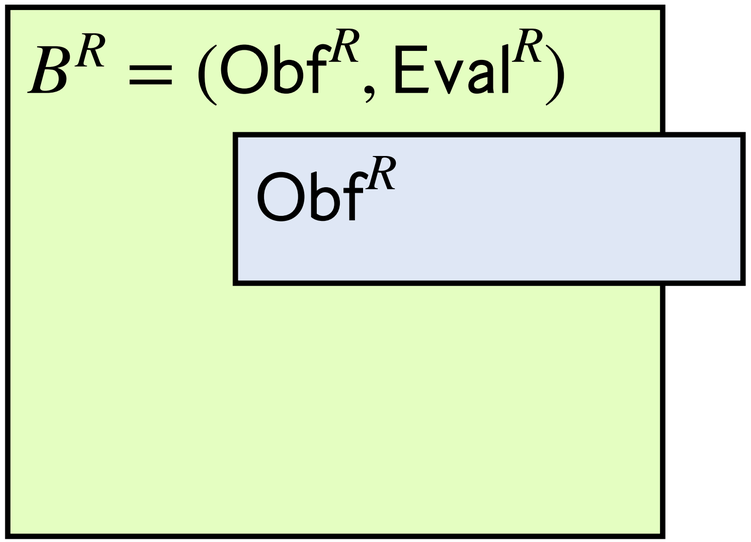
**Proof sketch:**

$B^R = (\mathsf{Obf}^R, \mathsf{Eval}^R)$

# How Strong are Crypto Oracles?

**Lemma:** Virtual black-box obfuscation exists relative to a crypto oracle

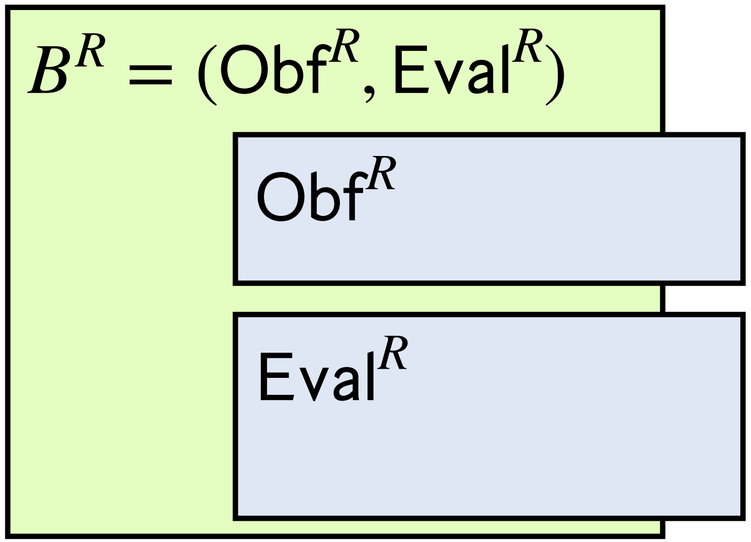
**Proof sketch:**



# How Strong are Crypto Oracles?

**Lemma:** Virtual black-box obfuscation exists relative to a crypto oracle

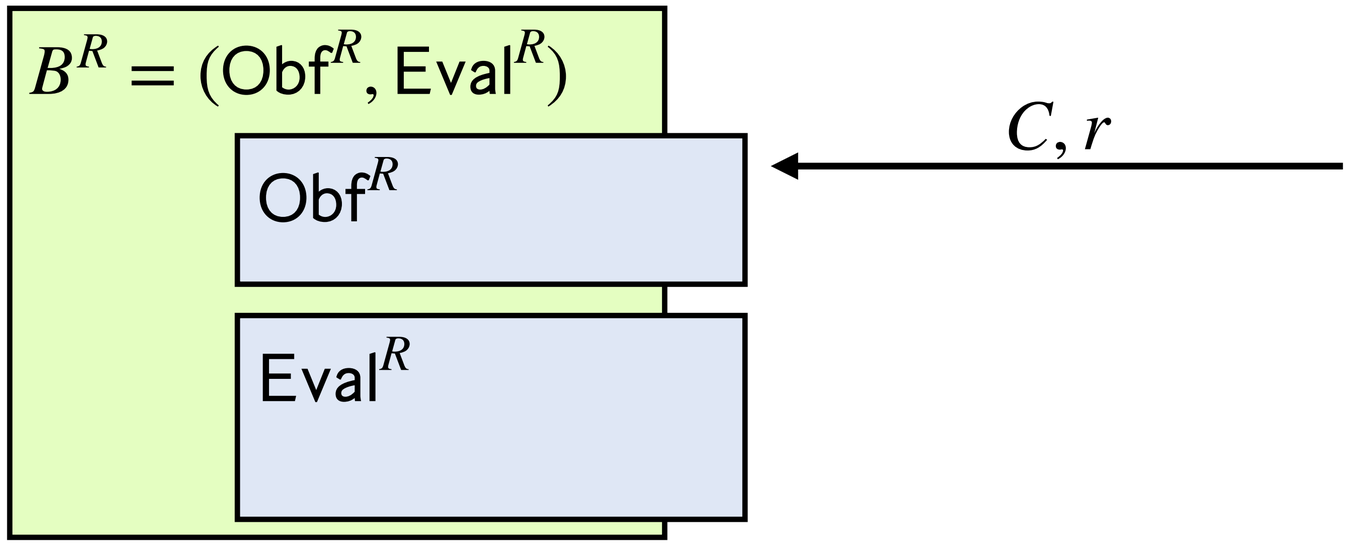
**Proof sketch:**



# How Strong are Crypto Oracles?

**Lemma:** Virtual black-box obfuscation exists relative to a crypto oracle

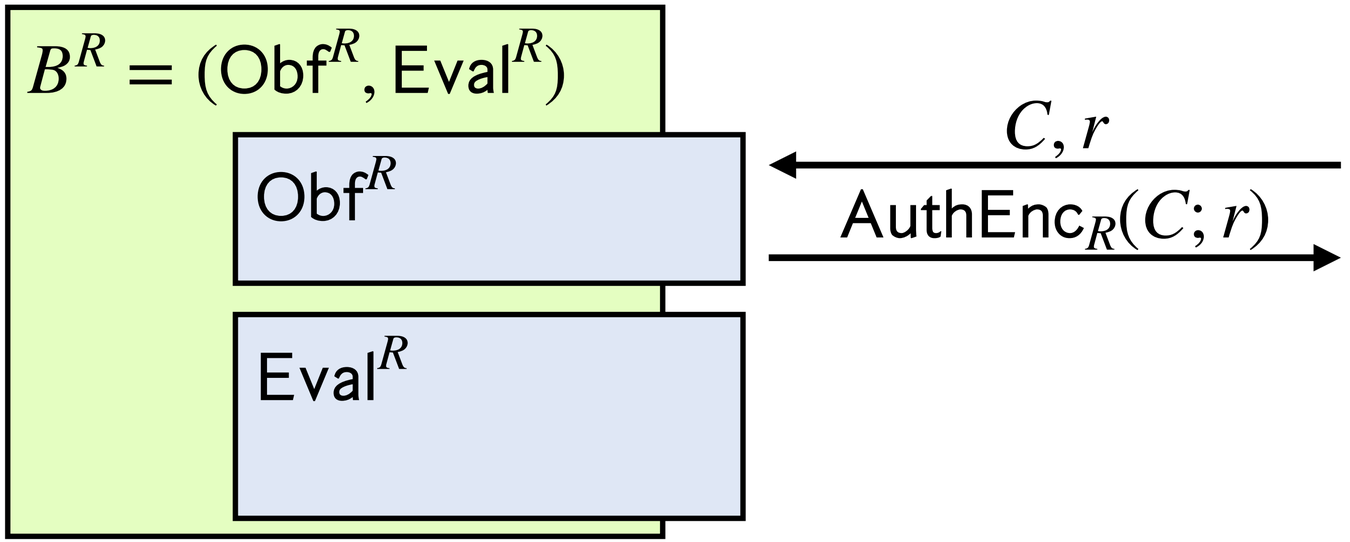
**Proof sketch:**



# How Strong are Crypto Oracles?

**Lemma:** Virtual black-box obfuscation exists relative to a crypto oracle

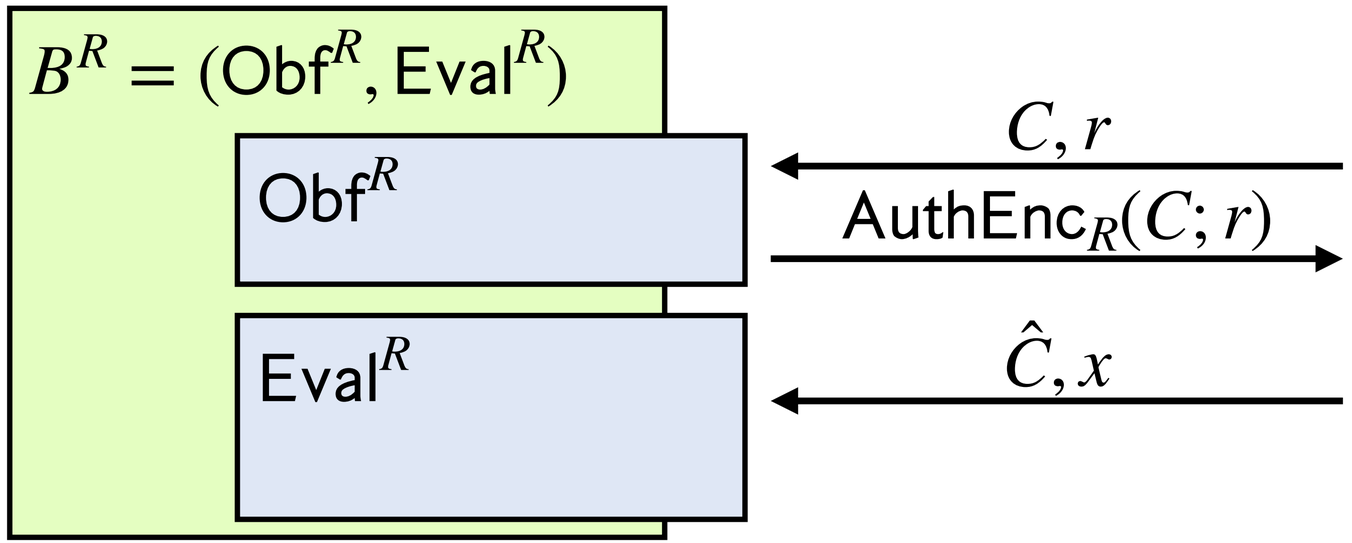
**Proof sketch:**



# How Strong are Crypto Oracles?

**Lemma:** Virtual black-box obfuscation exists relative to a crypto oracle

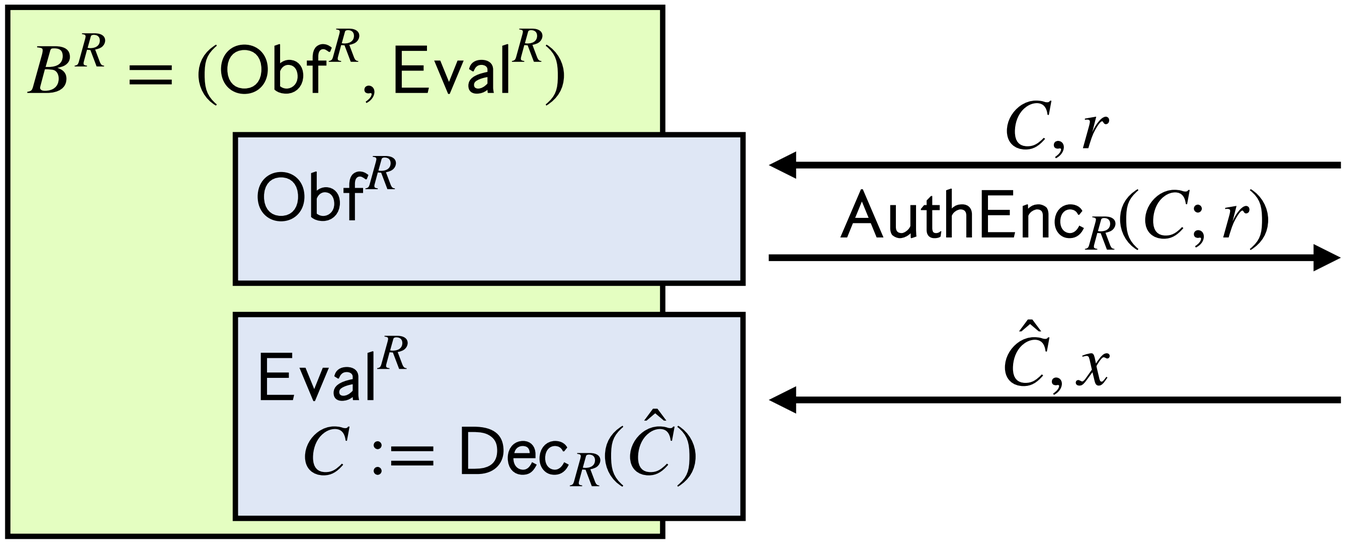
**Proof sketch:**



# How Strong are Crypto Oracles?

**Lemma:** Virtual black-box obfuscation exists relative to a crypto oracle

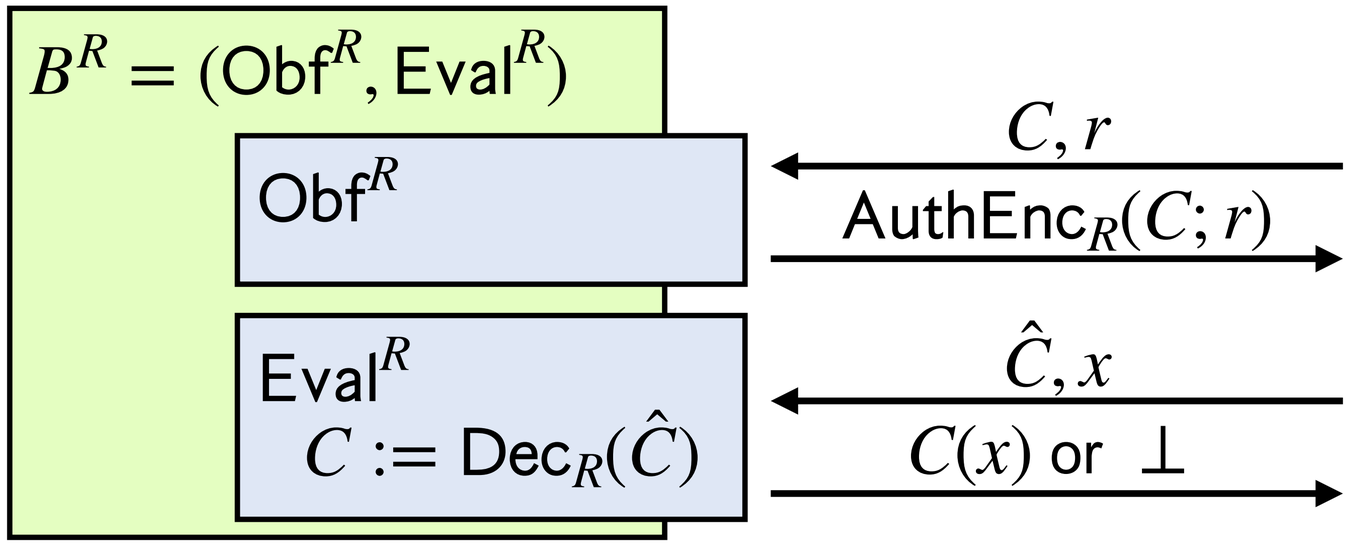
**Proof sketch:**



# How Strong are Crypto Oracles?

**Lemma:** Virtual black-box obfuscation exists relative to a crypto oracle

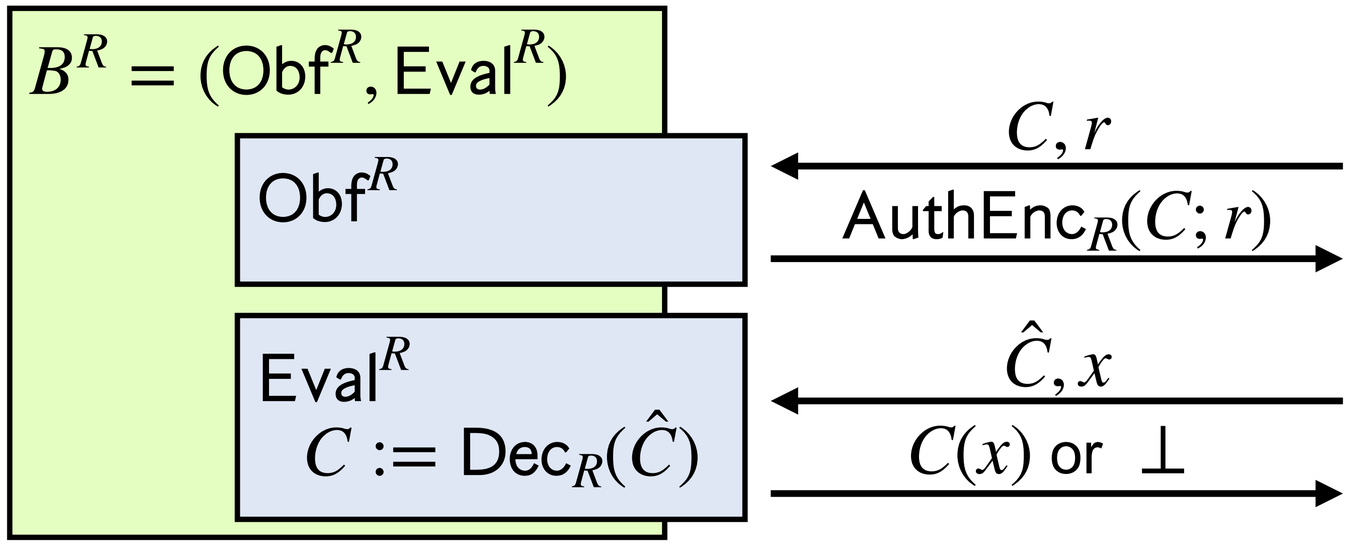
**Proof sketch:**



# How Strong are Crypto Oracles?

**Lemma:** Virtual black-box obfuscation exists relative to a crypto oracle

**Proof sketch:**




**Lemma:** The generic multi-linear group model can be realized as a crypto oracle ⇒ OT, FHE, … can as well

# Corollaries & Interpretation

Crypto oracles capture a wide array of primitives, even combinations

**Corollary**:

- ***If:*** there is a BB construction of SK-DEPIR from $P$
- ***Then:*** there is a BB construction of SK-DEPIR from OWFs

**Corollary**: There is no BB construction of ***2-round, passive server*** SK-DEPIR from OWFs

# Corollaries & Interpretation

Crypto oracles capture a wide array of primitives, even combinations

**Corollary**:

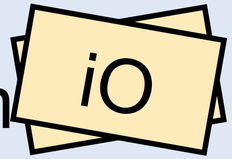
- ***If:*** there is a BB construction of SK-DEPIR from FHE
- ***Then:*** there is a BB construction of SK-DEPIR from OWFs

**Corollary**: There is no BB construction of ***2-round, passive server*** SK-DEPIR from FHE

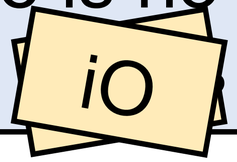
# Corollaries & Interpretation

Crypto oracles capture a wide array of primitives, even combinations

**Corollary**:

- ***If:*** there is a BB construction of SK-DEPIR from 

- ***Then:*** there is a BB construction of SK-DEPIR from OWFs

**Corollary**: There is no BB construction of ***2-round, passive server*** SK-DEPIR from 

# Corollaries & Interpretation

Crypto oracles capture a wide array of primitives, even combinations

**Corollary**:

- ***If:*** there is a BB construction of SK-DEPIR from 

- ***Then:*** there is a BB construction of SK-DEPIR from OWFs

**Corollary**: There is no BB construction of ***2-round, passive server*** SK-DEPIR from 

# Corollaries & Interpretation

Crypto oracles capture a wide array of primitives, even combinations

**Corollary**:

- ***If:*** there is a BB construction of SK-DEPIR from 

- ***Then:*** there is a BB construction of SK-DEPIR from OWFs

**Corollary**: There is no BB construction of ***2-round, passive server*** SK-DEPIR from 


Prior lemma gives VBB for oracle-aided circuits
⇒ capture non-BB techniques like “eval crypto under obfuscation”

# Corollaries & Interpretation

Our compiler is agnostic to the presence of other assumptions

# Corollaries & Interpretation

Our compiler is agnostic to the presence of other assumptions

**Corollary (e.g.):** Let $P$ be a primitive that exists wrt a crypto oracle

# Corollaries & Interpretation

Our compiler is agnostic to the presence of other assumptions

**Corollary (e.g.):** Let $P$ be a primitive that exists wrt a crypto oracle

- ***If:*** there is a construction of SK-DEPIR making BB use of $P$ and assuming DDH

# Corollaries & Interpretation

Our compiler is agnostic to the presence of other assumptions

**Corollary (e.g.):** Let $P$ be a primitive that exists wrt a crypto oracle

- ***If:*** there is a construction of SK-DEPIR making BB use of $P$ and assuming DDH
- ***Then:*** there is a construction of SK-DEPIR assuming just DDH

# Corollaries & Interpretation

Our compiler is agnostic to the presence of other assumptions

**Corollary (e.g.):** Let $P$ be a primitive that exists wrt a crypto oracle

- ***If:*** there is a construction of SK-DEPIR making BB use of $P$ and assuming DDH
- ***Then:*** there is a construction of SK-DEPIR assuming just DDH

**Crucial:** need DDH over a ***concrete group*** (e.g. $\mathbb{Z}_p^*$), the generic group can be realized as crypto oracle

# Corollaries & Interpretation

# Corollaries & Interpretation

**Recall:** Prior positive results

- SK-DEPIR from “permuted codes with noise” [CHR’17,BIPW’17]
- UK-DEPIR from RingLWE [L**M**W’23]

# Corollaries & Interpretation

**Recall:** Prior positive results

- SK-DEPIR from “permuted codes with noise” [CHR’17,BIPW’17]
- UK-DEPIR from RingLWE [L**M**W’23]

Both make use of concrete algebraic structure of the underlying assumption

# Corollaries & Interpretation

**Recall:** Prior positive results

- SK-DEPIR from “permuted codes with noise” [CHR’17,BIPW’17]
- UK-DEPIR from RingLWE [L**M**W’23]

Both make use of concrete algebraic structure of the underlying assumption

- [CHR’17,BIPW’17]: Hardness over a locally decodable code

# Corollaries & Interpretation

**Recall:** Prior positive results

- SK-DEPIR from “permuted codes with noise” [CHR’17,BIPW’17]
- UK-DEPIR from RingLWE [L**M**W’23]

Both make use of concrete algebraic structure of the underlying assumption

- [CHR’17,BIPW’17]: Hardness over a locally decodable code
- [L**M**W’23]: RingLWE ring is conducive to some preprocessing

# Corollaries & Interpretation

**Recall:** Prior positive results

- SK-DEPIR from “permuted codes with noise” [CHR’17,BIPW’17]
- UK-DEPIR from RingLWE [L**M**W’23]

Both make use of concrete algebraic structure of the underlying assumption

- [CHR’17,BIPW’17]: Hardness over a locally decodable code
- [L**M**W’23]: RingLWE ring is conducive to some preprocessing

**Our result:** This concreteness is ***inherent***

# Summary

**Theorem 1**: For a large class of primitives $P$

- ***If:*** BB construction of SK-DEPIR from $P$
- ***Then:*** BB construction of SK-DEPIR from OWFs

The class: constructible relative to crypto oracle

**Theorem 2**: There is no BB construction of
***2-round, passive server*** SK-DEPIR from OWFs

# Summary

**Theorem 1**: For a large class of primitives $P$

- ***If:*** BB construction of SK-DEPIR from $P$
- ***Then:*** BB construction of SK-DEPIR from OWFs

The class: constructible relative to crypto oracle

**Theorem 2**: There is no BB construction of
***2-round, passive server*** SK-DEPIR from OWFs

**Open Questions:**

# Summary

**Theorem 1**: For a large class of primitives $P$

- ***If:*** BB construction of SK-DEPIR from $P$
- ***Then:*** BB construction of SK-DEPIR from OWFs

The class: constructible relative to crypto oracle

**Theorem 2**: There is no BB construction of ***2-round, passive server*** SK-DEPIR from OWFs

**Open Questions:**

- Fully rule out SK-DEPIR from OWFs

# Summary

**Theorem 1**: For a large class of primitives $P$

- ***If:*** BB construction of SK-DEPIR from $P$
- ***Then:*** BB construction of SK-DEPIR from OWFs

The class: constructible relative to crypto oracle

**Theorem 2**: There is no BB construction of ***2-round, passive server*** SK-DEPIR from OWFs

**Open Questions:**

- Fully rule out SK-DEPIR from OWFs
- DEPIR from new (concrete) assumptions
  - Weaker than RingLWE, more standard than permuted codes with noise

# Summary

**Theorem 1**: For a large class of primitives $P$

- ***If:*** BB construction of SK-DEPIR from $P$
- ***Then:*** BB construction of SK-DEPIR from OWFs

The class: constructible relative to crypto oracle

**Theorem 2**: There is no BB construction of ***2-round, passive server*** SK-DEPIR from OWFs

**Open Questions:**

- Fully rule out SK-DEPIR from OWFs
- DEPIR from new (concrete) assumptions
  - Weaker than RingLWE, more standard than permuted codes with noise

Thank you!

eprint: 2025/552