Authenticated private information retrieval

USENIX Security Symposium 2023

Simone Colombo
EPFL

Kirill Nikitin
NYGC & Columbia University

Henry Corrigan-Gibbs
MIT

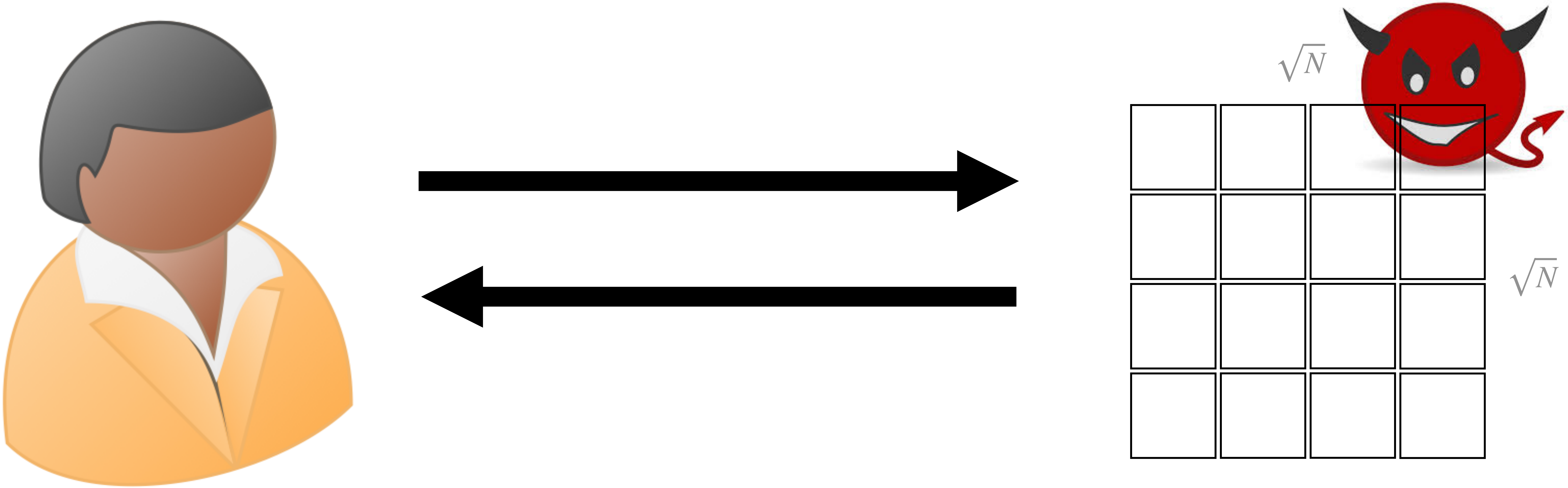
David J. Wu
UT Austin

Bryan Ford
EPFL

Private information retrieval (PIR) [CGKS95]

holds index $i \in \{1,\ldots,N\}$

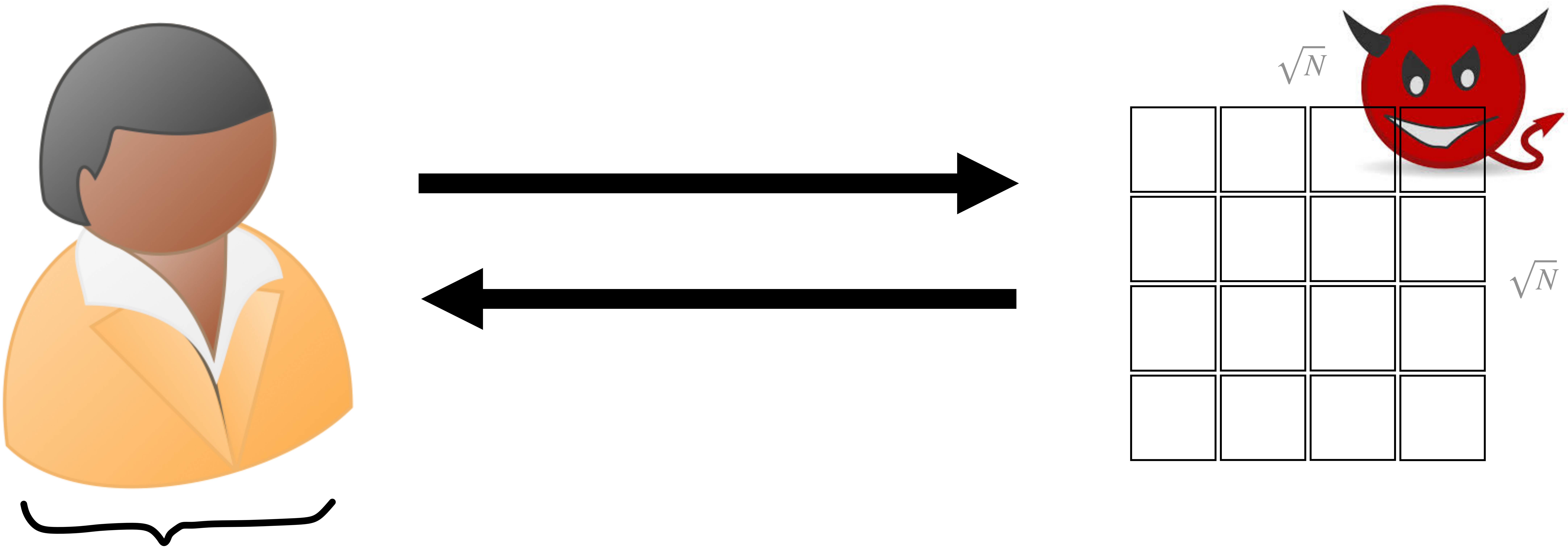
holds database $d \in \mathbb{F}^N$



Private information retrieval (PIR) [CGKS95]

holds index $i \in \{1, \ldots, N\}$

holds database $d \in \mathbb{F}^N$

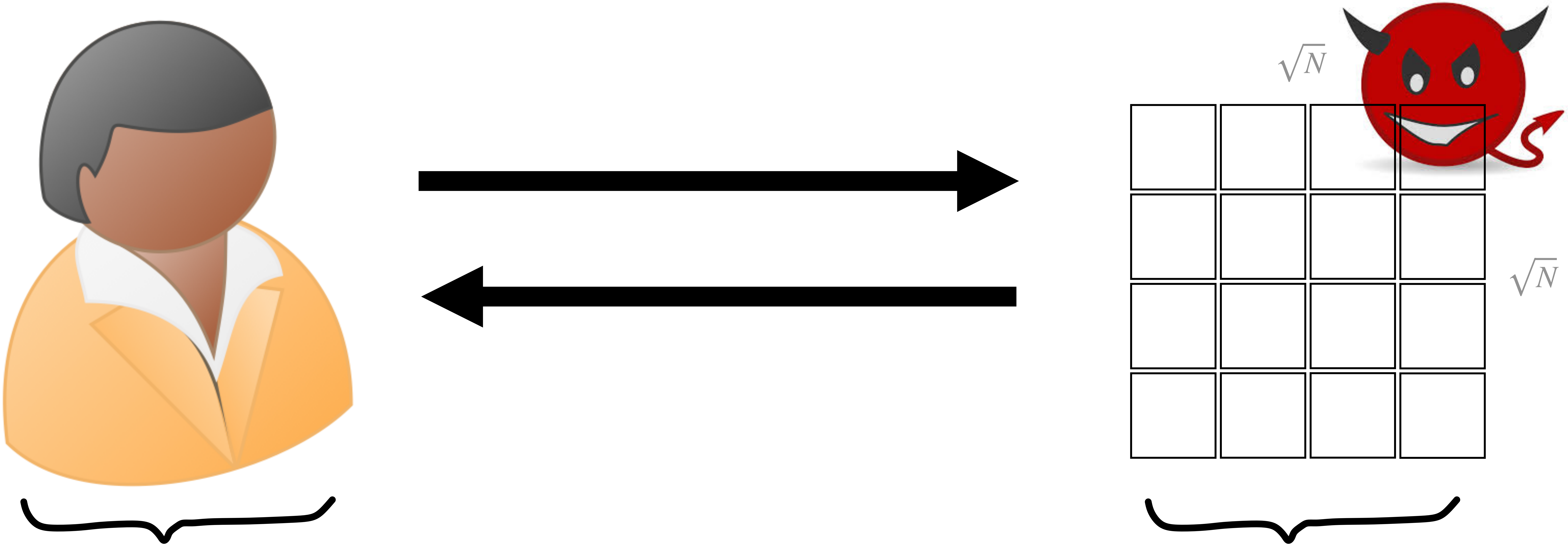


learns d_i

Private information retrieval (PIR) [CGKS95]

holds index $i \in \{1,\ldots,N\}$

holds database $d \in \mathbb{F}^N$



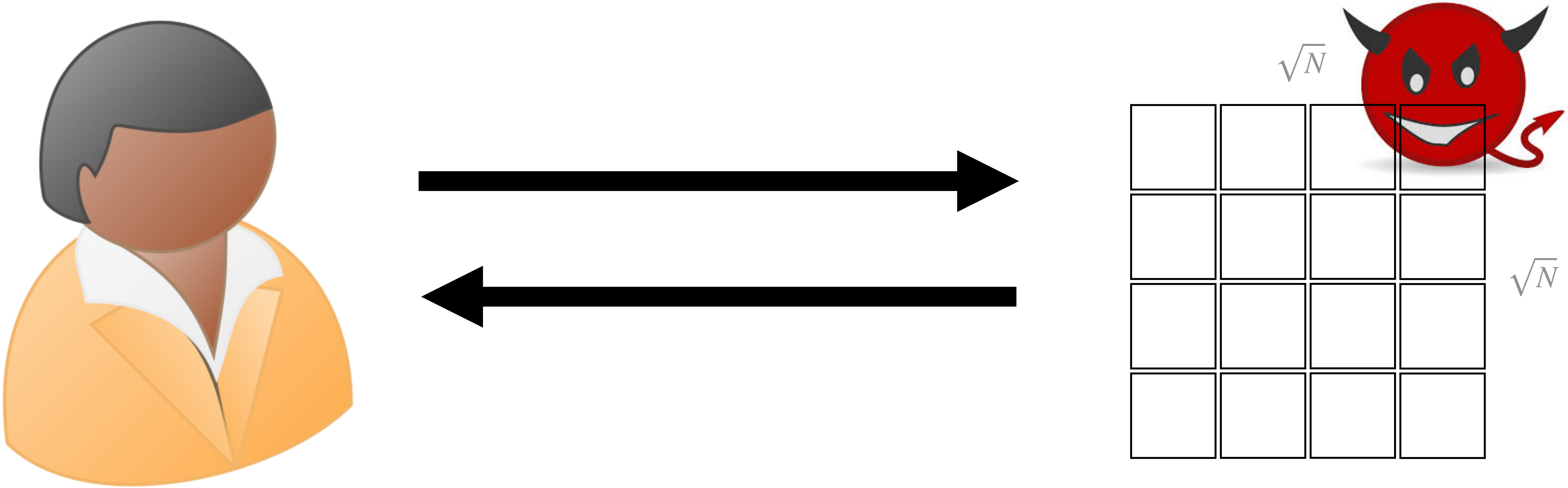
learns d_i

learns nothing

Private information retrieval (PIR) [CGKS95,WYGVZ17]

holds function $f\colon \mathbb{F}^N \to \mathbb{F}$

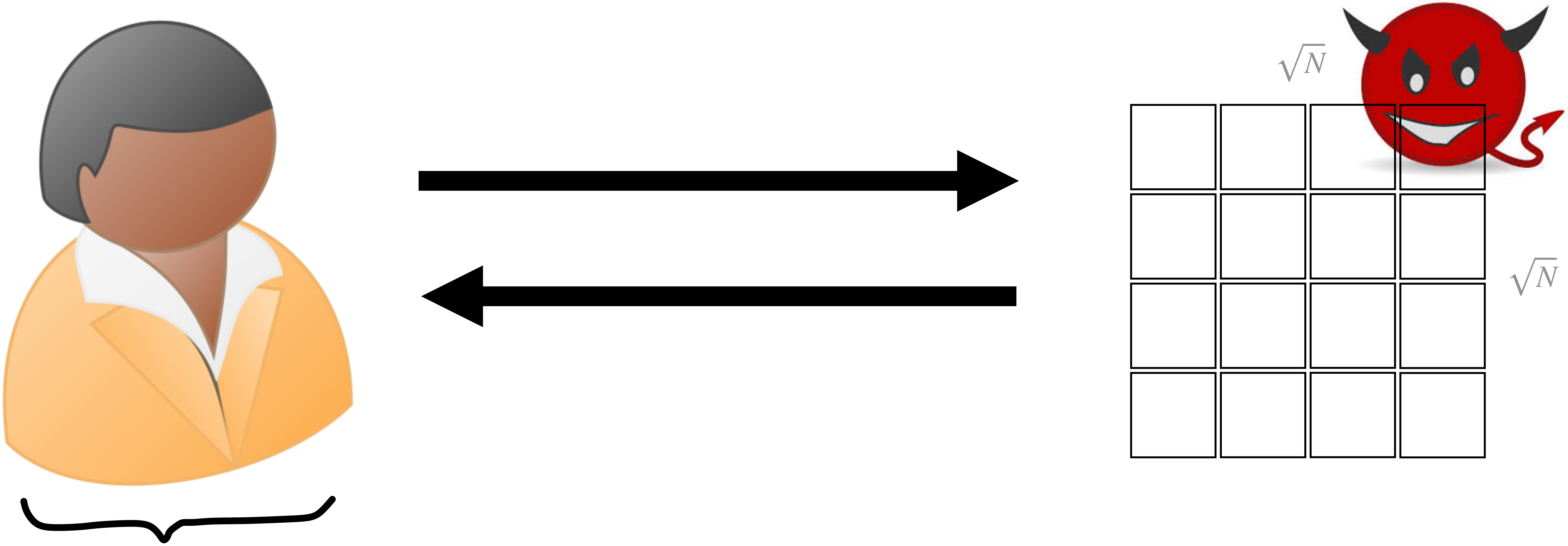
holds database $d \in \mathbb{F}^N$



Private information retrieval (PIR) [CGKS95,WYGVZ17]

holds function $f\colon \mathbb{F}^N \to \mathbb{F}$

holds database $d \in \mathbb{F}^N$

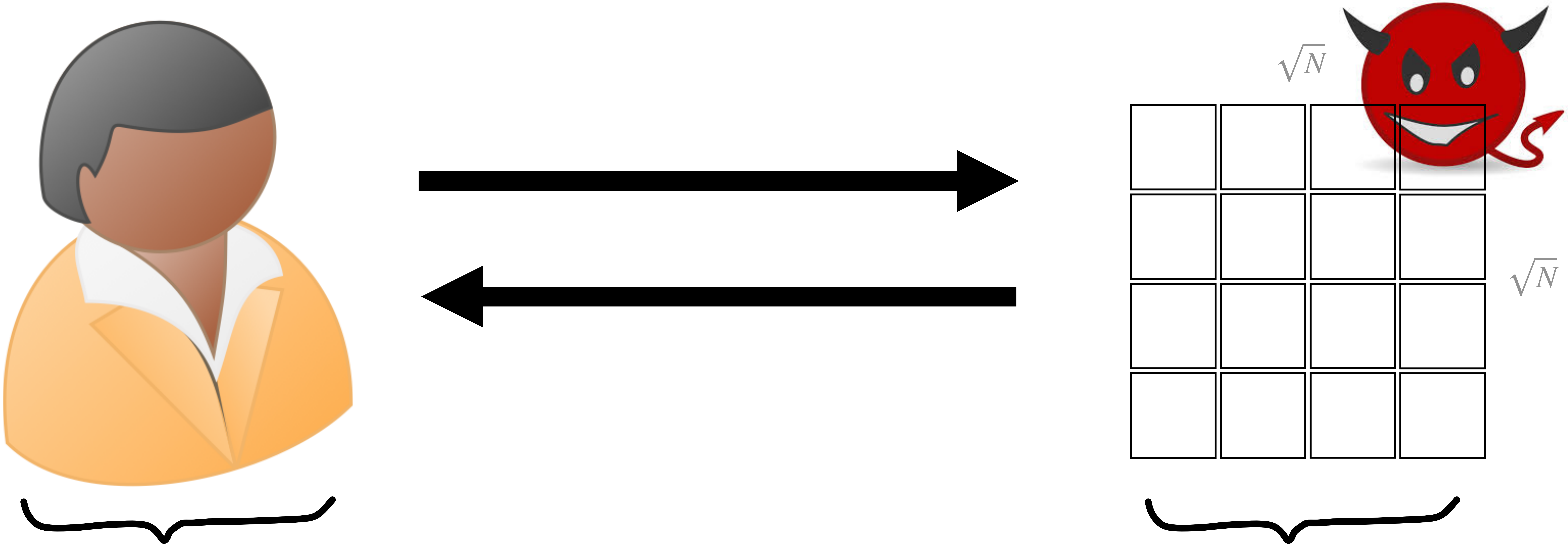


learns $f(d)$

Private information retrieval (PIR) [CGKS95,WYGVZ17]

holds function $f\colon \mathbb{F}^N \to \mathbb{F}$

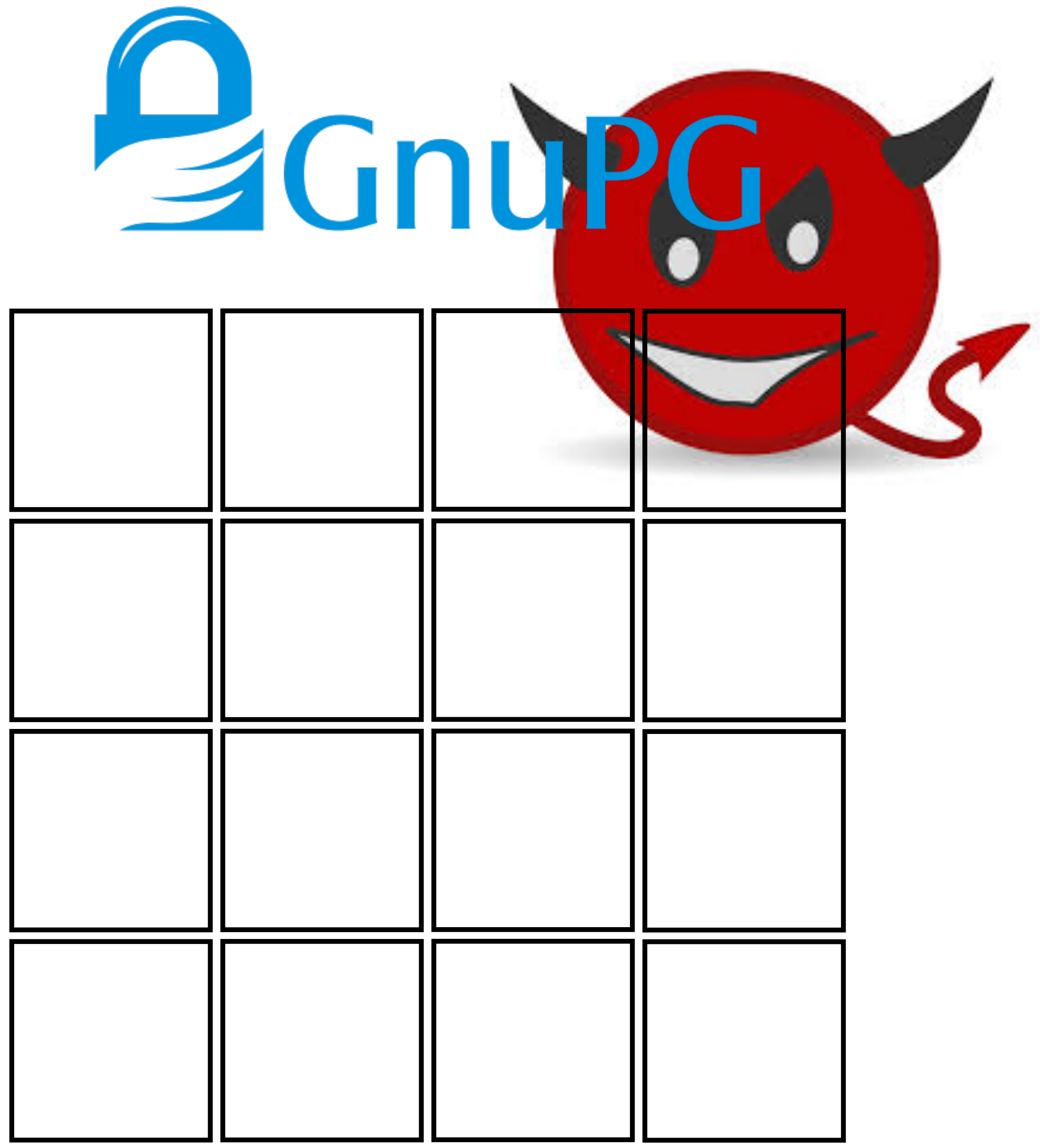
holds database $d \in \mathbb{F}^N$



learns $f(d)$

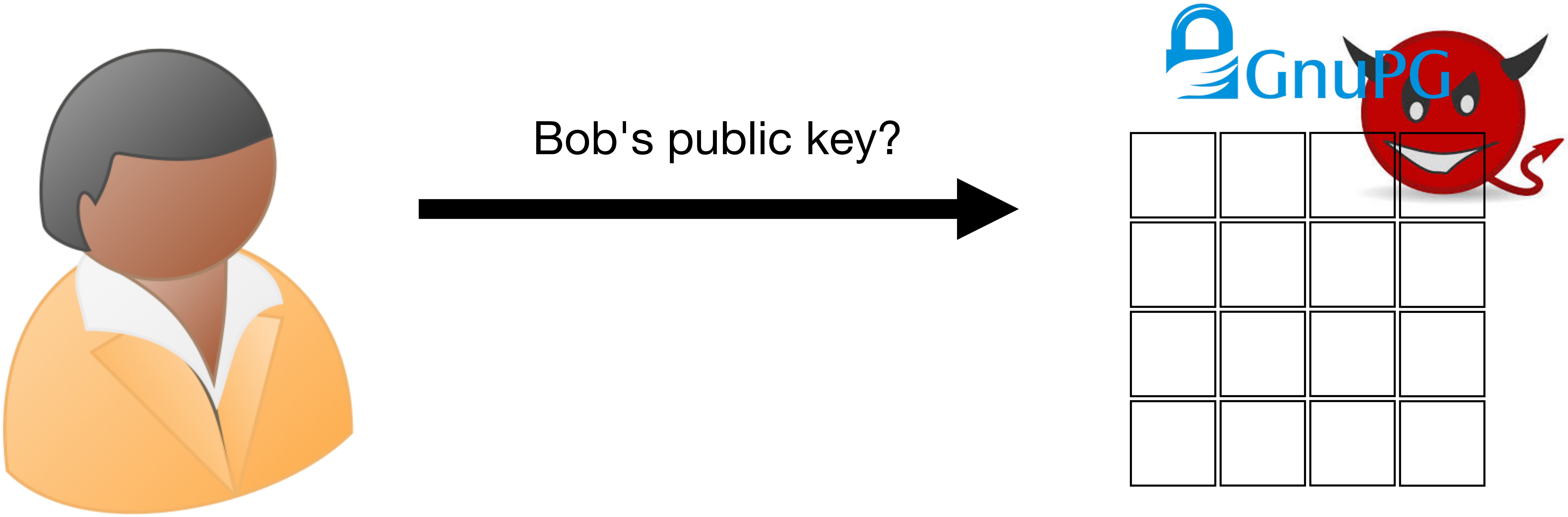
learns nothing

An example application: PGP key server



PGP key server

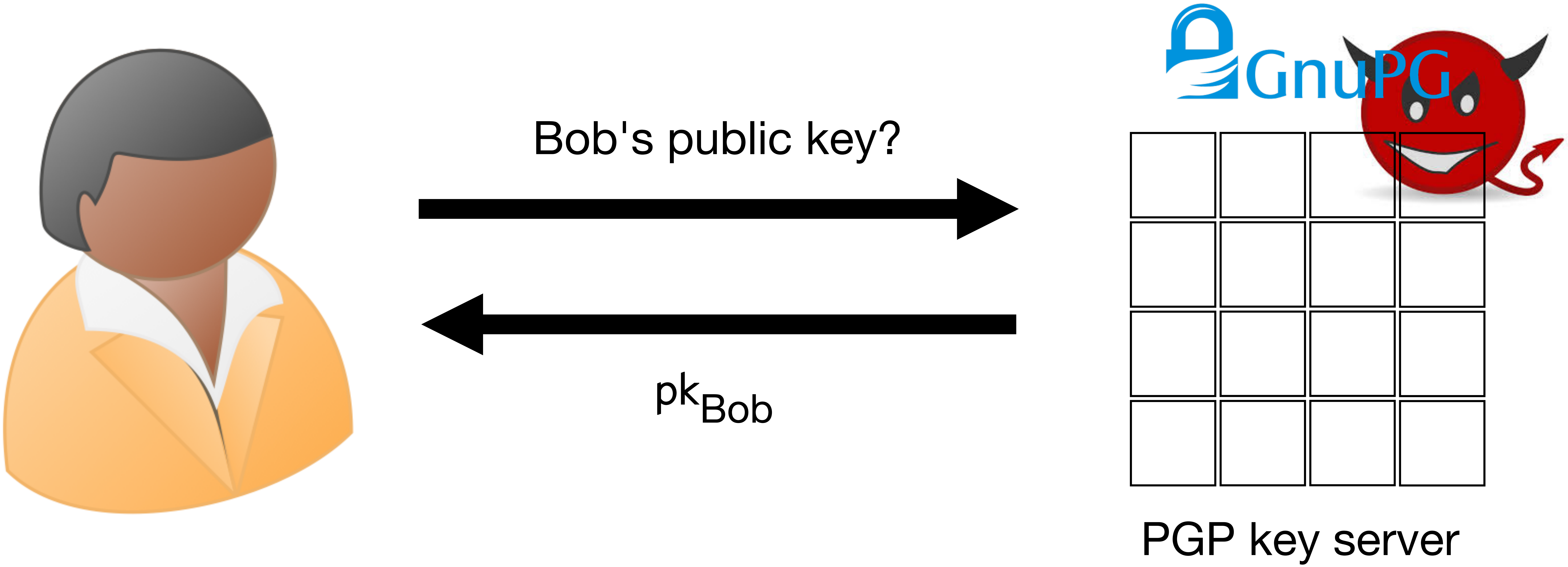
An example application: PGP key server



Bob's public key?

PGP key server

An example application: PGP key server



Bob's public key?

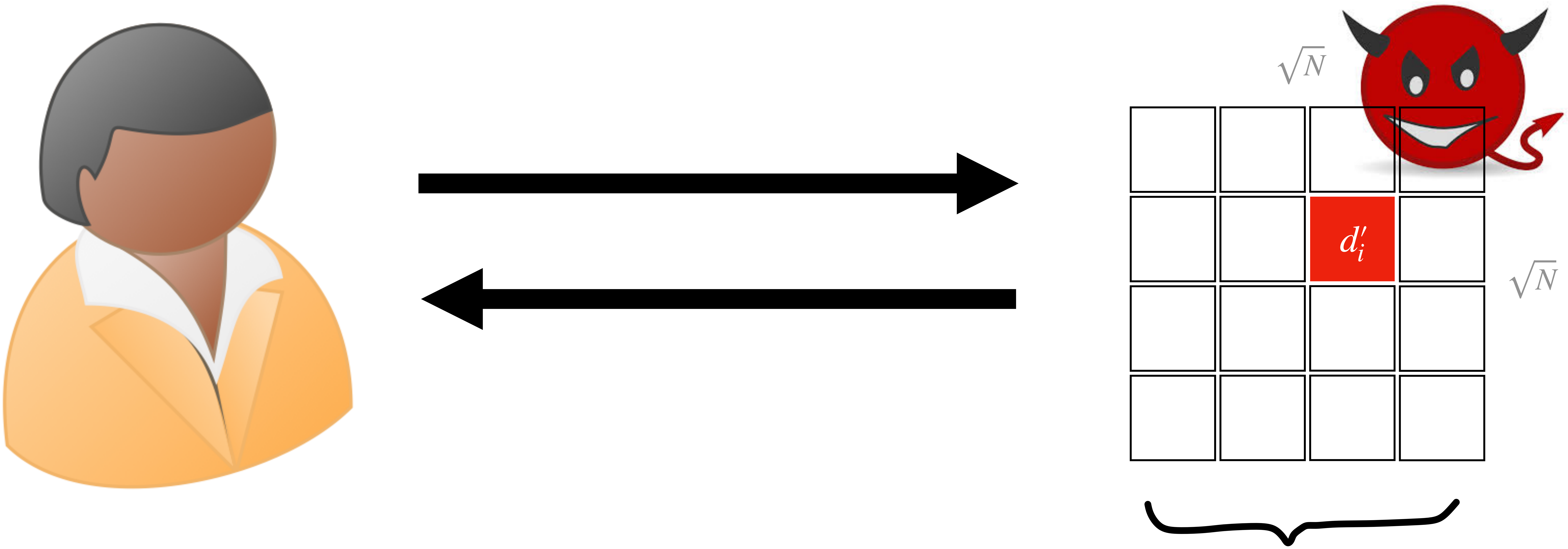
pk_{Bob}

PGP key server

PIR does not consider integrity

holds index $i \in \{1, \ldots, N\}$

holds database $d \in \mathbb{F}^N$

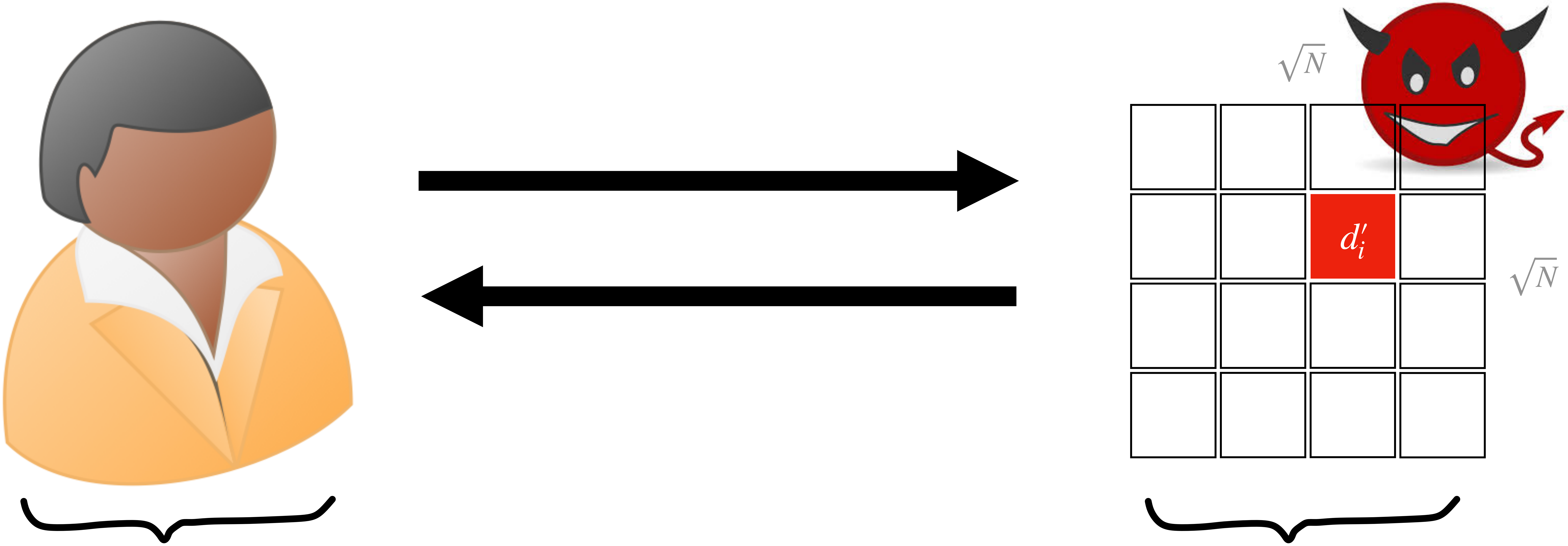


learns nothing

PIR does not consider integrity

holds index $i \in \{1, \ldots, N\}$

holds database $d \in \mathbb{F}^N$



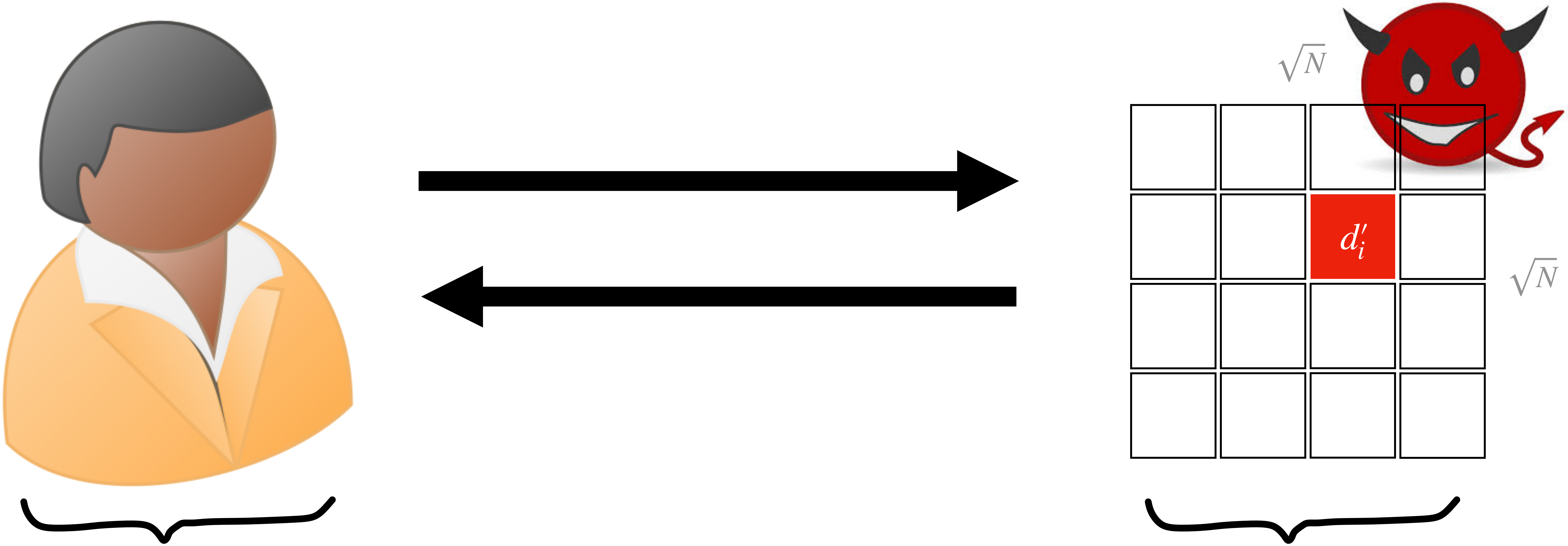
learns wrong d_i'

learns nothing

PIR does not consider integrity

holds index $i \in \{1,\ldots,N\}$

holds database $d \in \mathbb{F}^N$



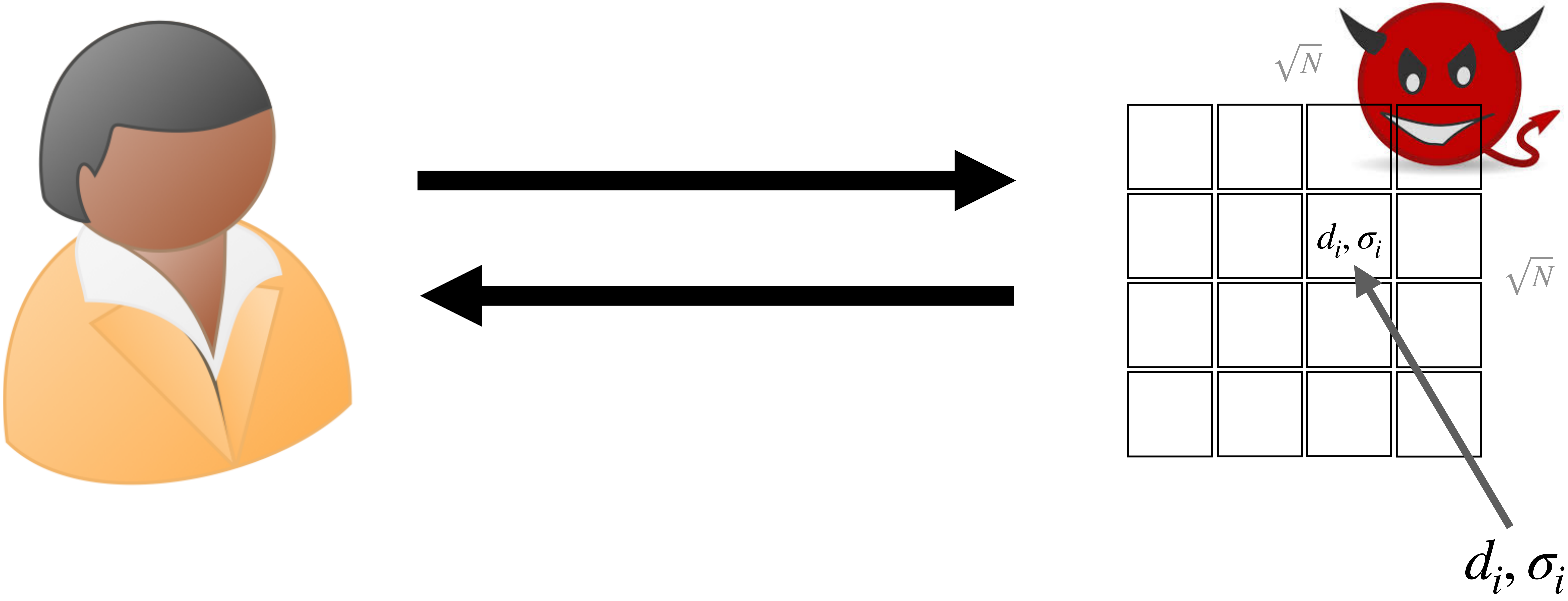
learns wrong $\text{pk}_{\text{adversary}}$

learns nothing

PIR and authentication are not enough

holds index $i \in \{1, \ldots, N\}$

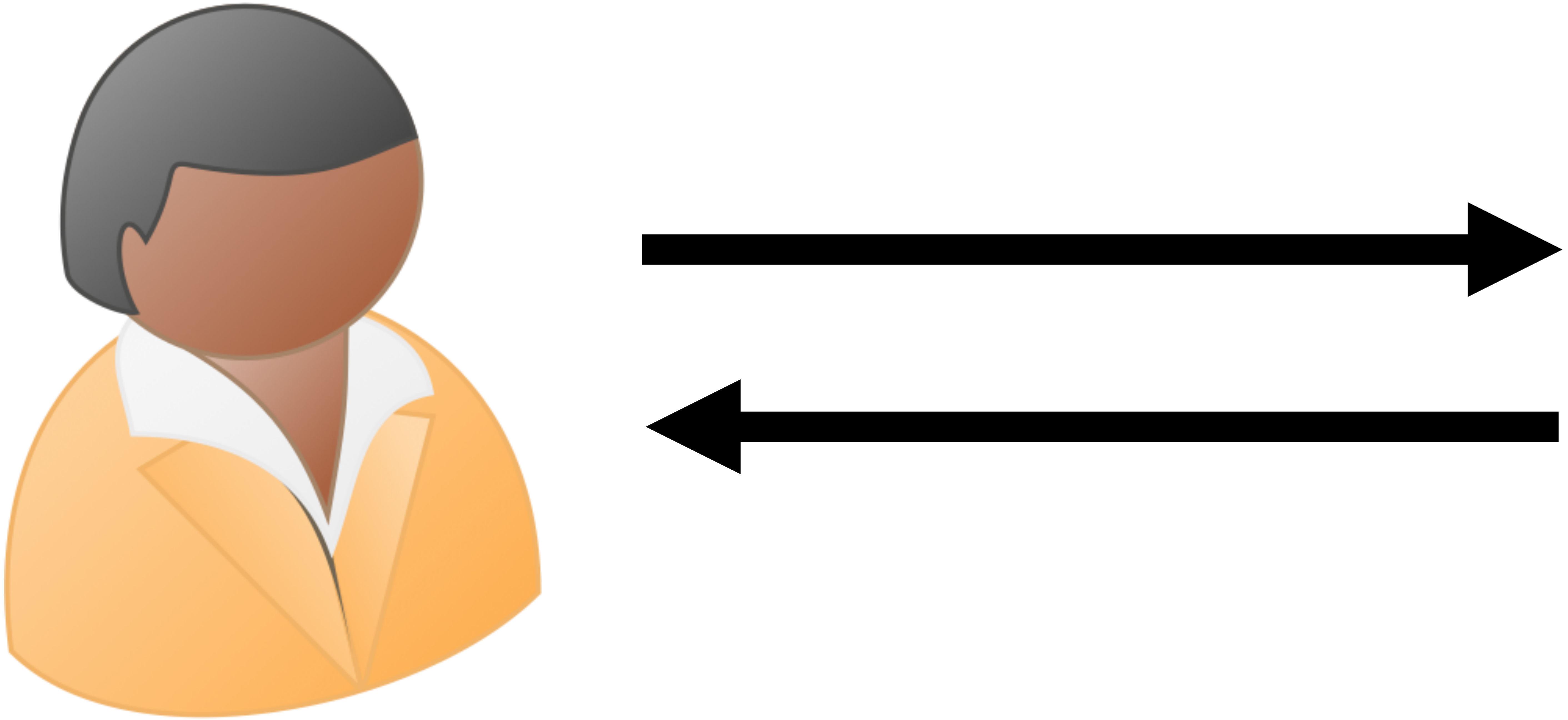
holds database $d \in \mathbb{F}^N$

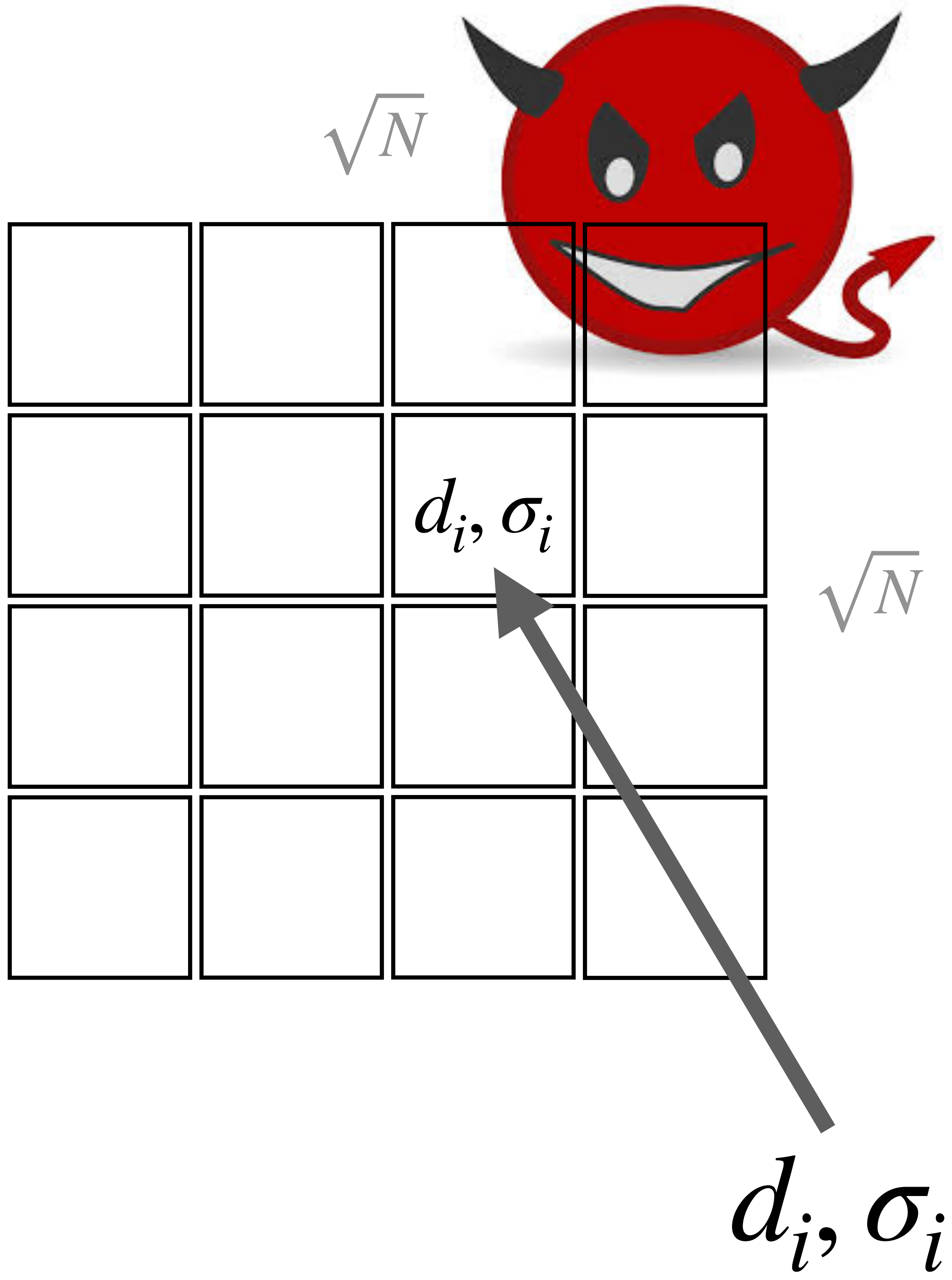


PIR and authentication are not enough

holds index $i \in \{1,\ldots,N\}$

holds database $d \in \mathbb{F}^N$





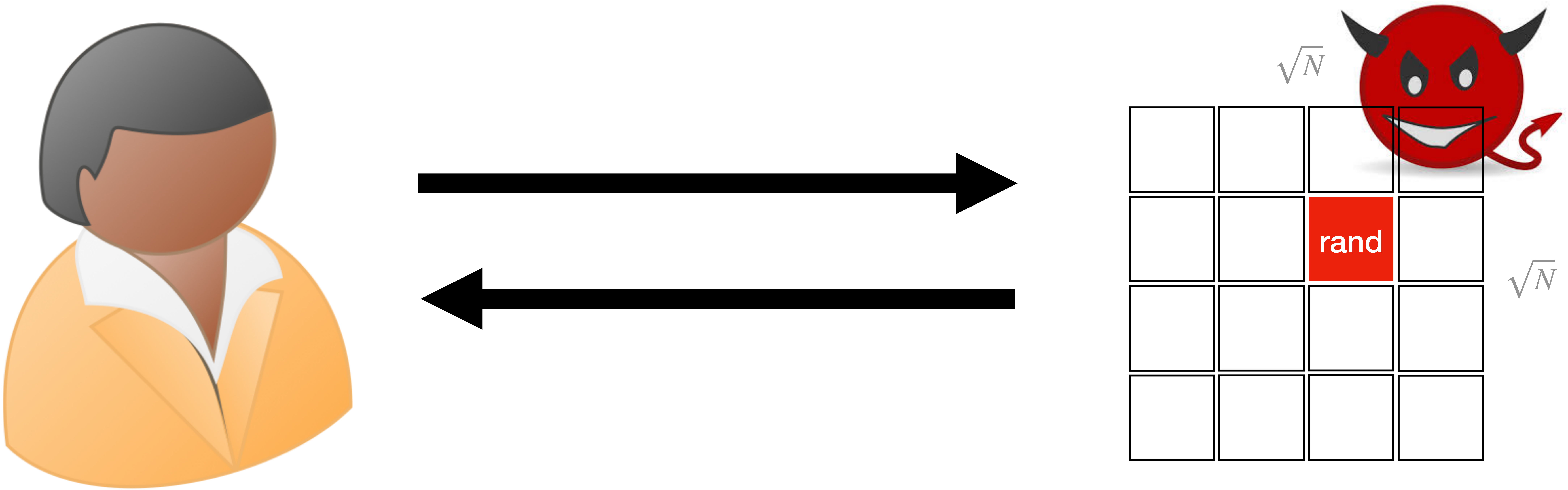
if $\mathsf{Verify}(\mathsf{pk}, d_i, \sigma_i) = \top$ return d_i

else abort

PIR and authentication are not enough

holds index $i \in \{1, \ldots, N\}$

holds database $d \in \mathbb{F}^N$



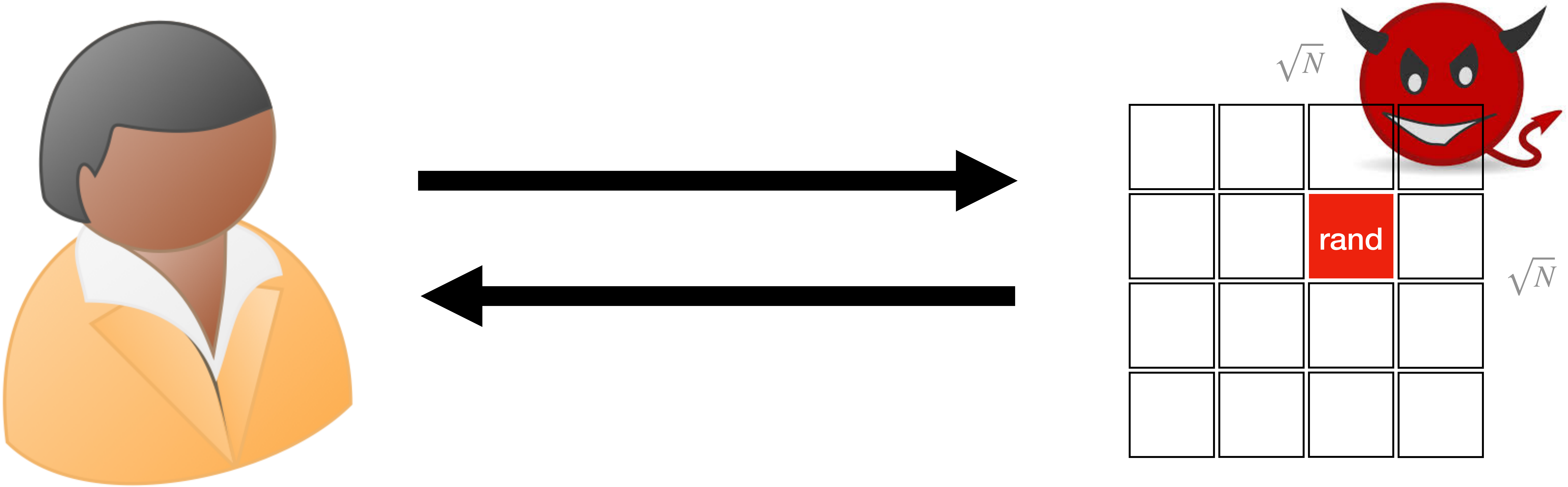
if Verify$(\mathsf{pk}, d_i, \sigma_i) = \top$ return d_i

else abort

PIR and authentication are not enough

holds index $i \in \{1, \ldots, N\}$

holds database $d \in \mathbb{F}^N$



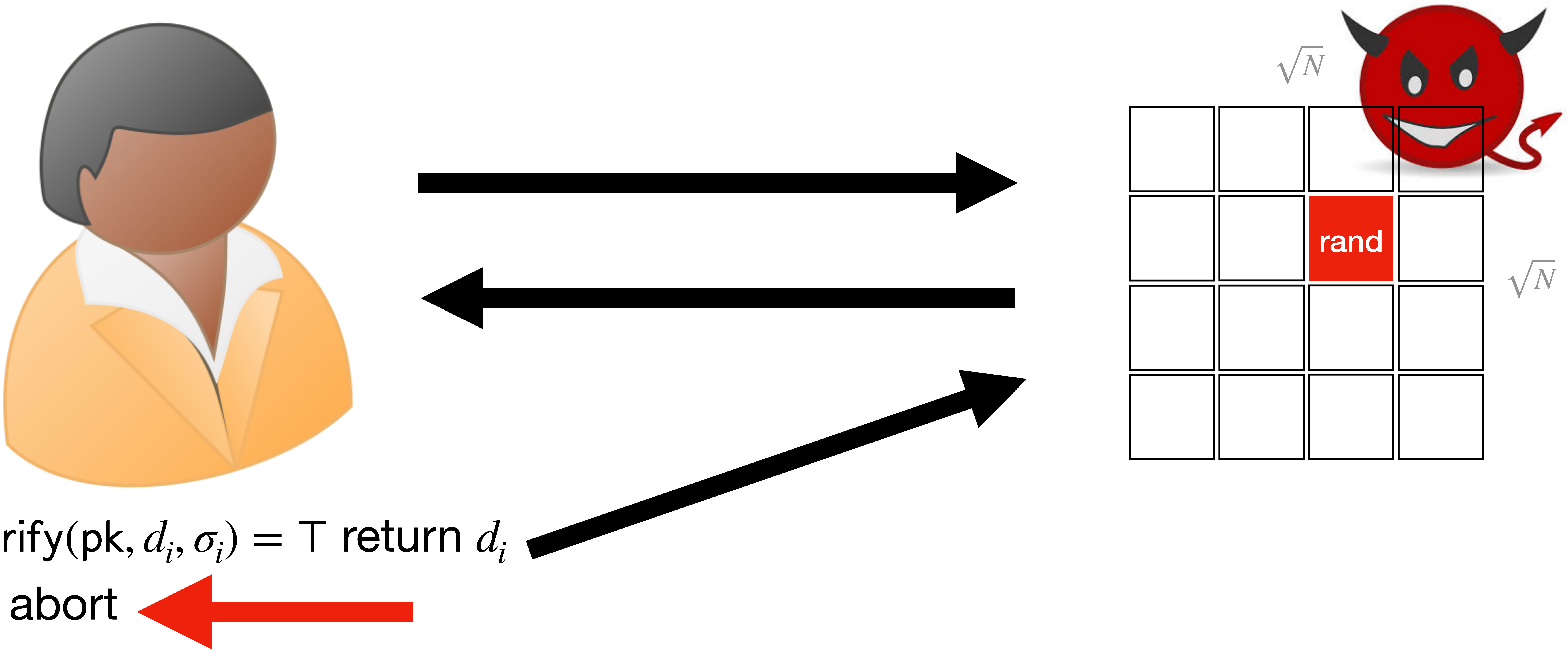
if Verify$(\mathsf{pk}, d_i, \sigma_i) = \top$ return d_i

else abort

PIR and authentication are not enough

holds index $i \in \{1, \ldots, N\}$

holds database $d \in \mathbb{F}^N$



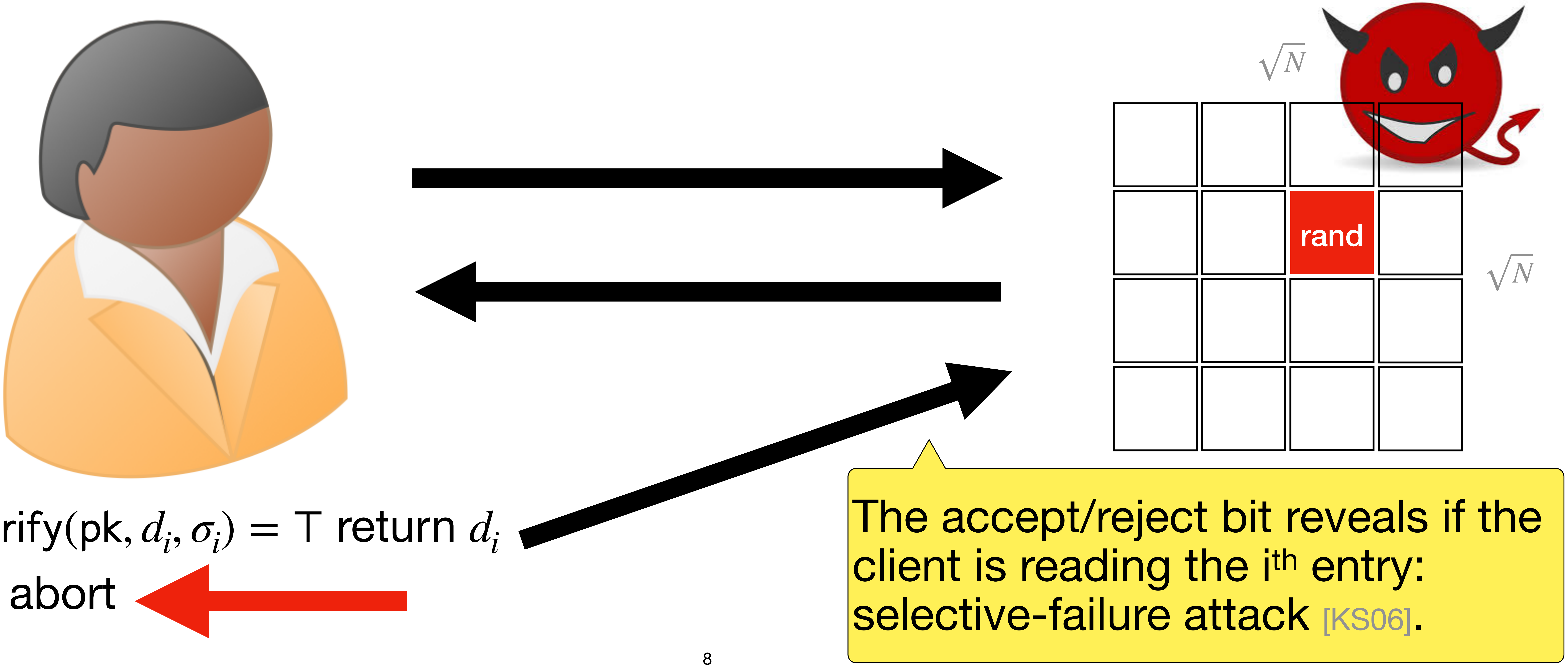
if $\mathsf{Verify}(\mathsf{pk}, d_i, \sigma_i) = \top$ return d_i

else abort

PIR and authentication are not enough

holds index $i \in \{1,\ldots,N\}$

holds database $d \in \mathbb{F}^N$



if $\mathrm{Verify}(\mathrm{pk}, d_i, \sigma_i) = \top$ return d_i

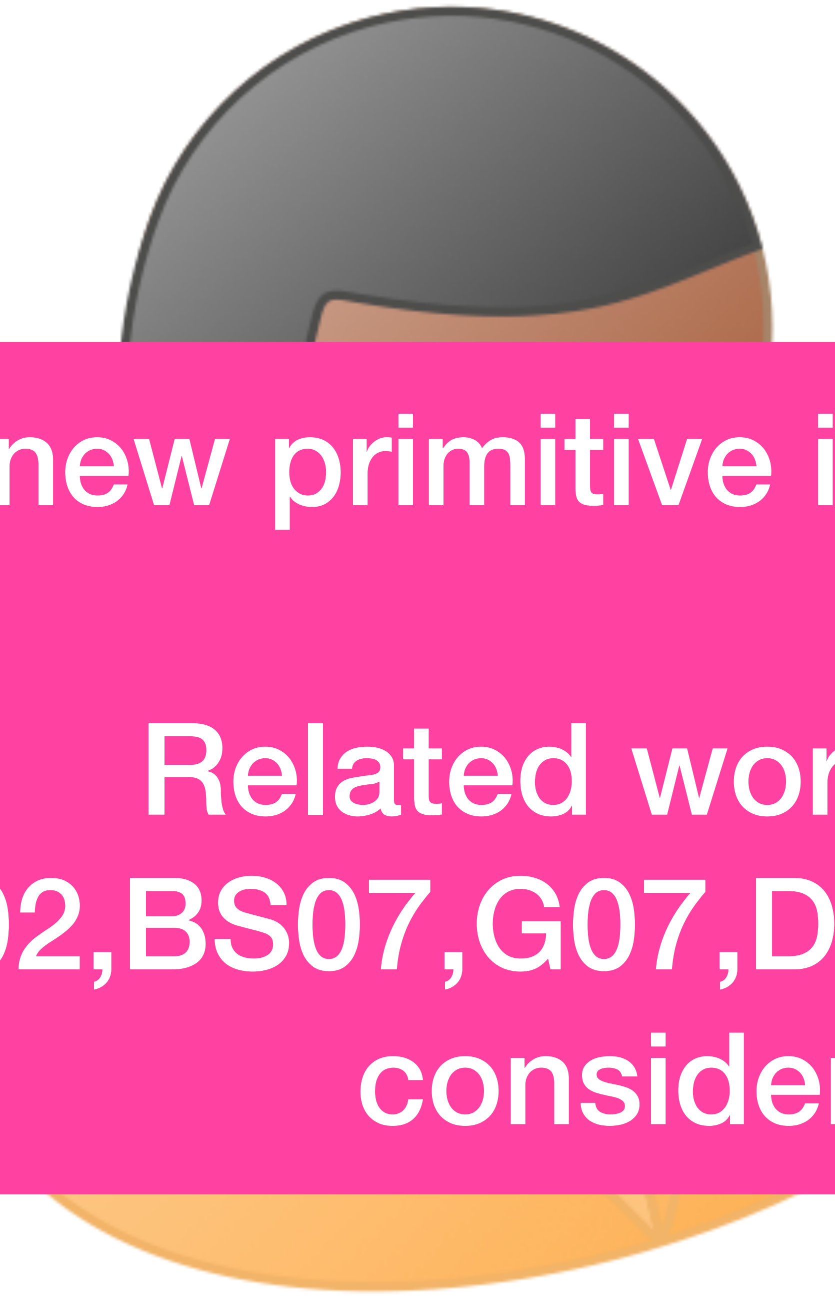
else abort

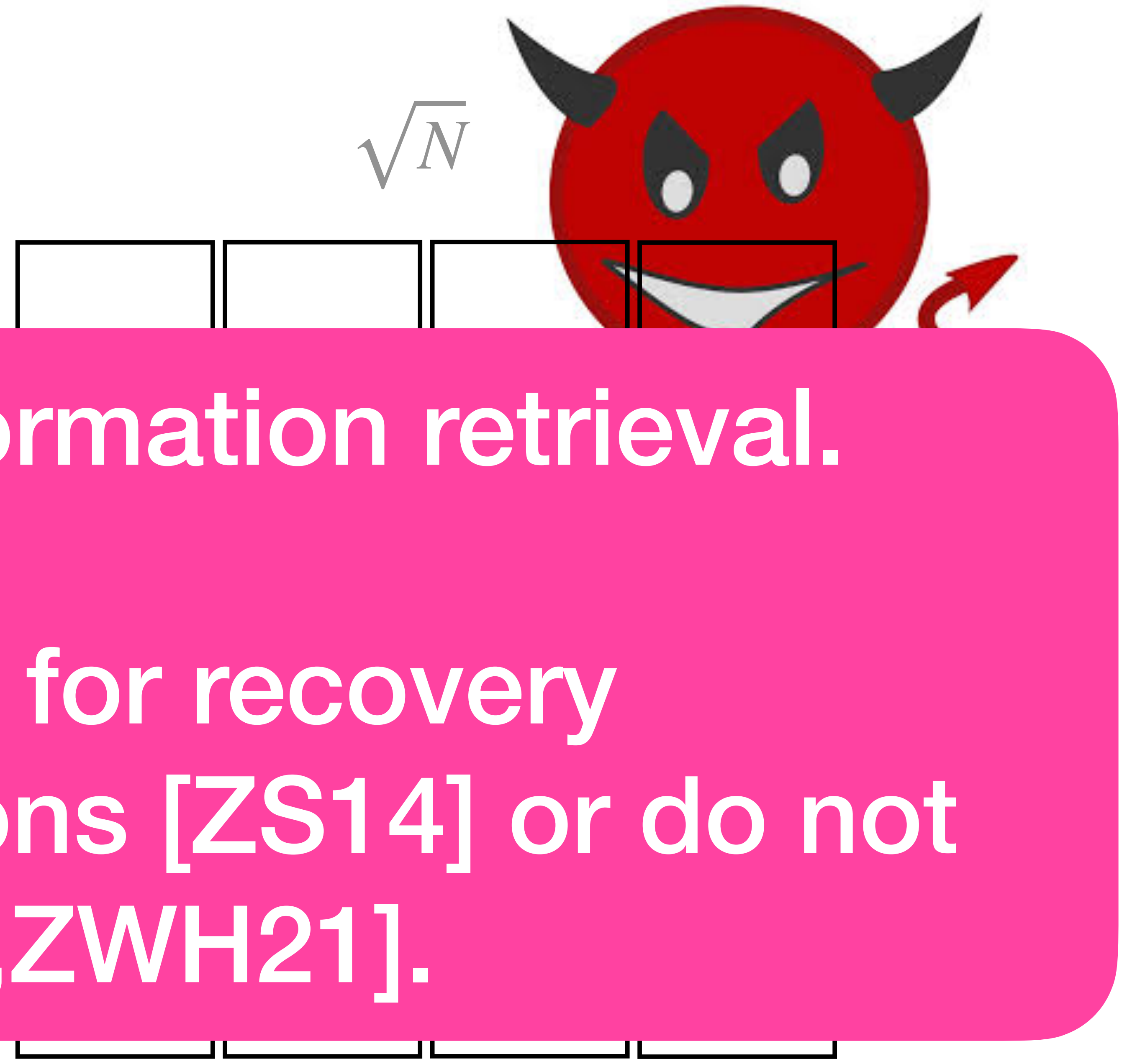
The accept/reject bit reveals if the client is reading the i^{th} entry: selective-failure attack [KS06].

PIR and authentication are not enough

holds index $i \in \{1,\ldots,N\}$

holds database $d \in \mathbb{F}^N$





A new primitive is necessary: authenticated private information retrieval.

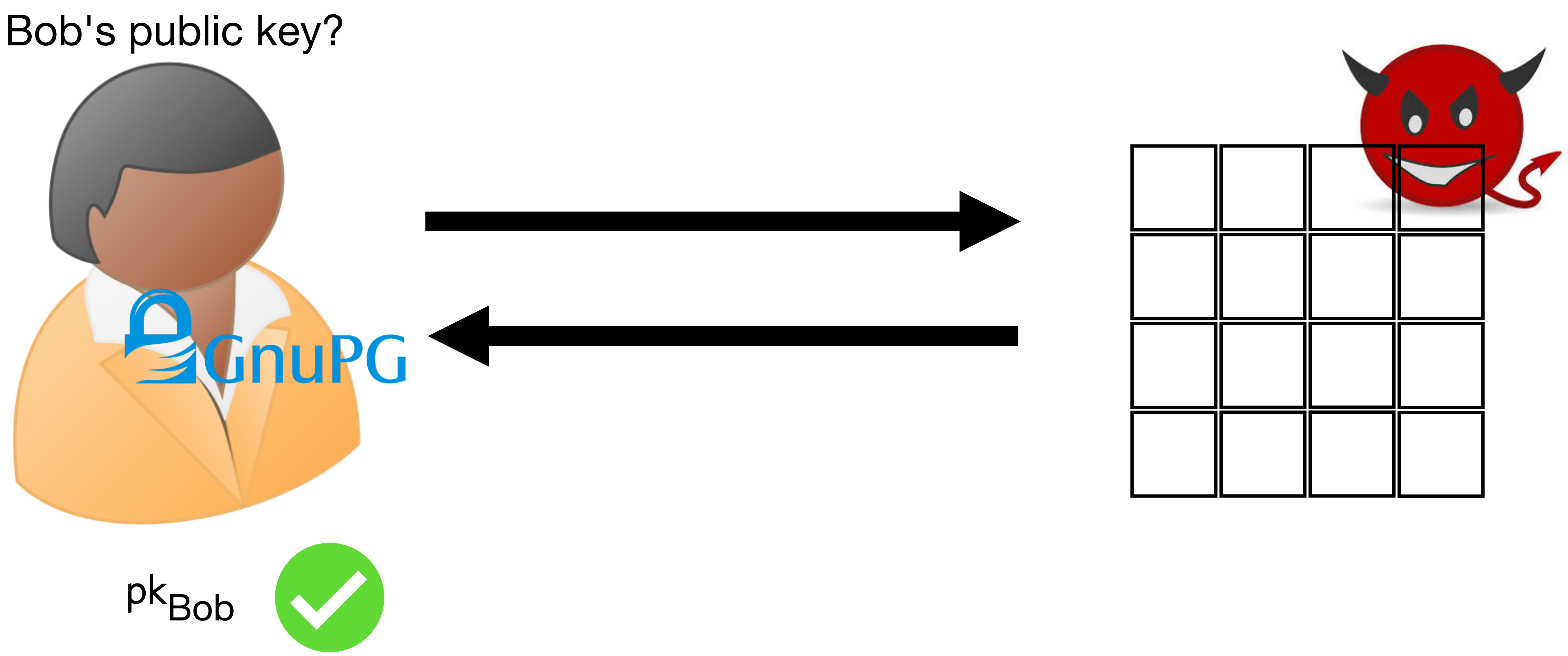
Related works require a majority of honest servers for recovery [BS02,BS07,G07,DGN12,K19,YXB02], stronger assumptions [ZS14] or do not consider selective-failure attacks [KO97,WZ18,ZWH21].

if Verify(pk, d_i, σ_i) $= \top$ return d_i

else abort

The accept/reject bit reveals if the client is reading the ith entry: selective-failure attack [KS06].

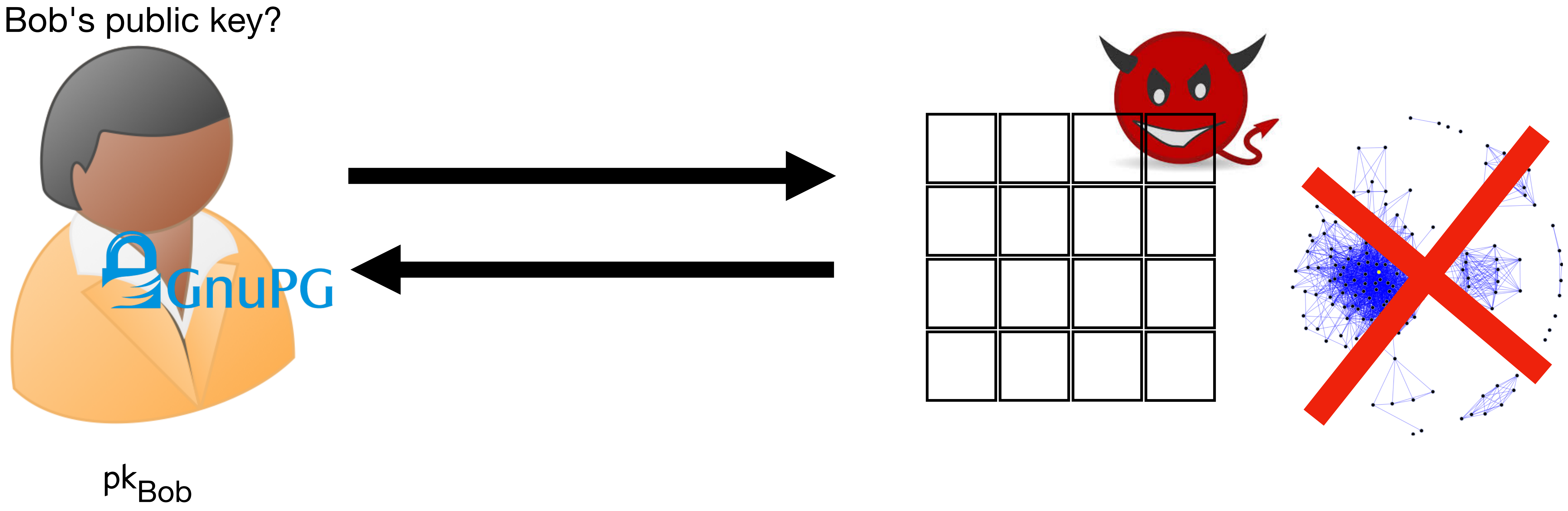
Authenticated PIR properties


Bob's public key?

pk_{Bob} ✓

- Correctness: If client and server are honest, the client recovers pk_{Bob}.

Authenticated PIR properties

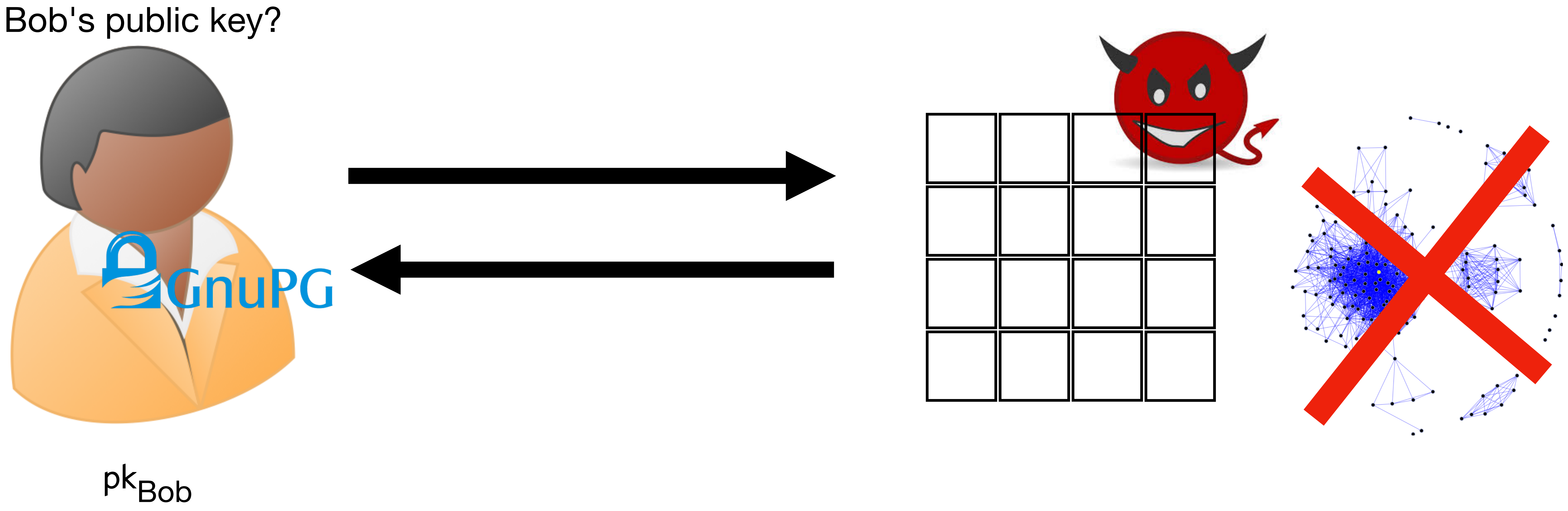


Bob's public key?

pk_{Bob}

- Correctness: If client and server are honest, the client recovers pk_{Bob}.
- Privacy: The server(s) learns nothing about the content of the client’s query, even if the server(s) learns whether the client aborted during reconstruction.

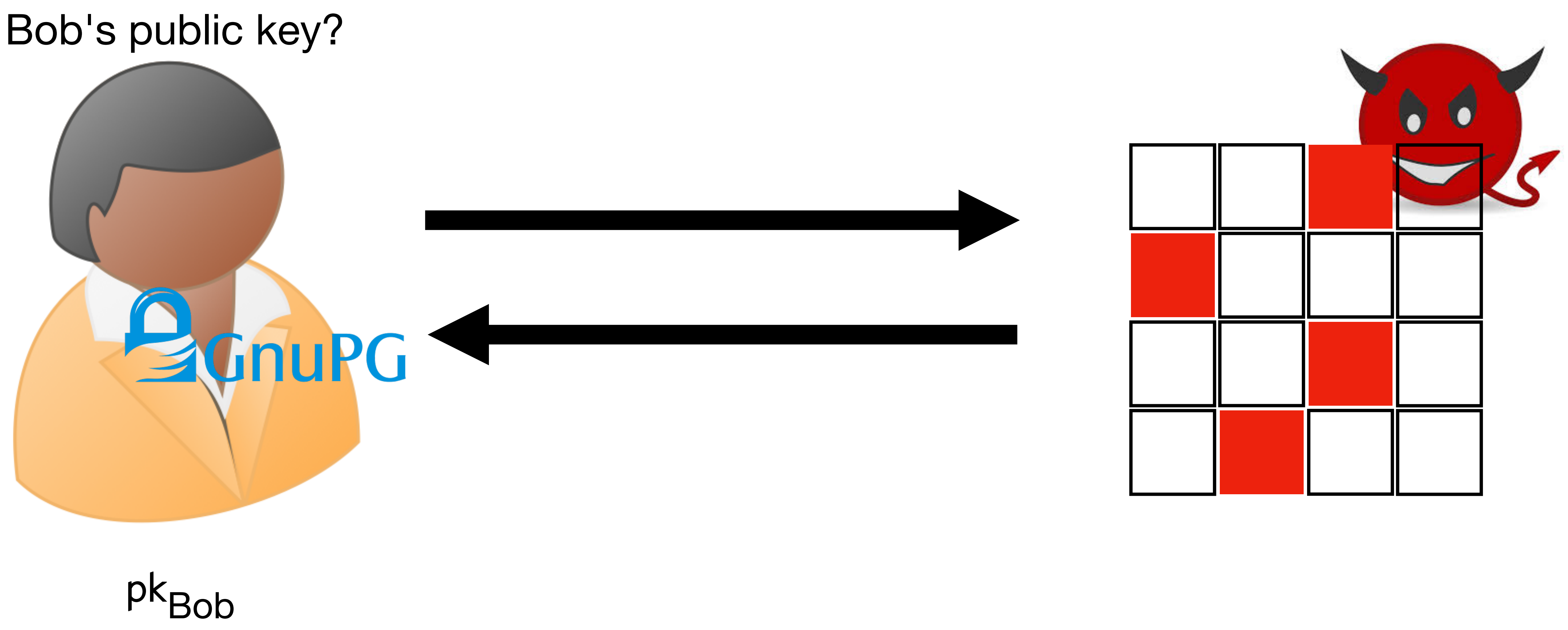
Authenticated PIR properties



- Correctness: If client and server are honest, the client recovers pk_{Bob}.
- Privacy: The server(s) learns nothing about the content of the client’s query, even if the server(s) learns whether the client aborted during reconstruction.

Selective-failure attacks.

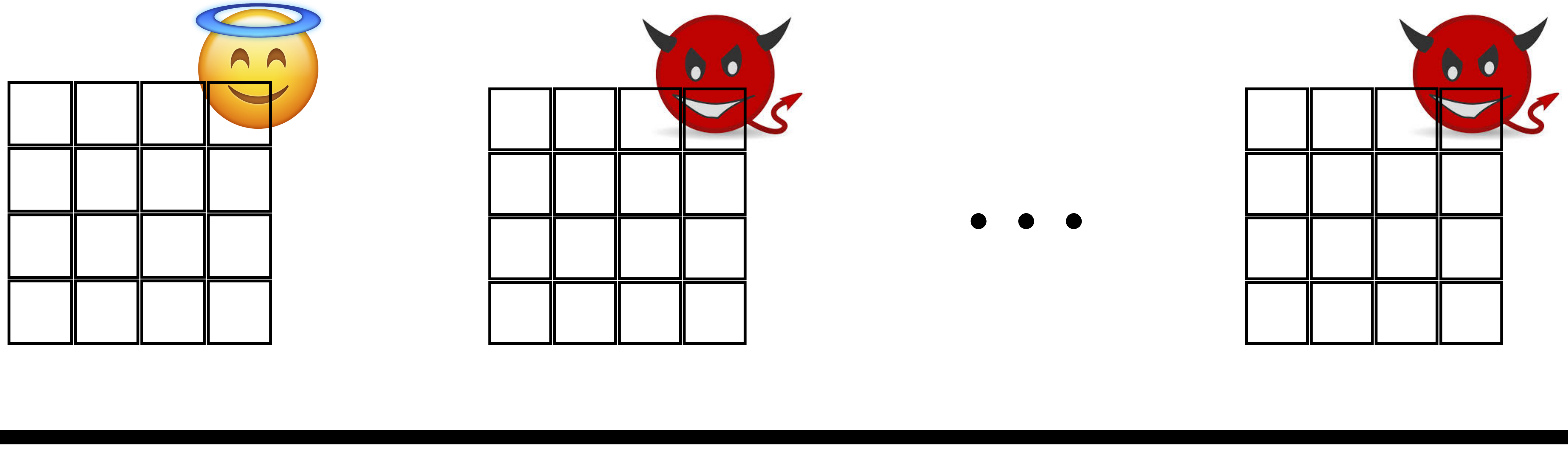
Authenticated PIR properties



- Correctness: If client and server are honest, the client recovers pk_{Bob}.
- Privacy: The server(s) learn nothing about the content of the client's query, even if the server(s) learn whether the client aborted during reconstruction.
- Integrity: The client either outputs the authentic pk_{Bob} or aborts, except with negligible probability.

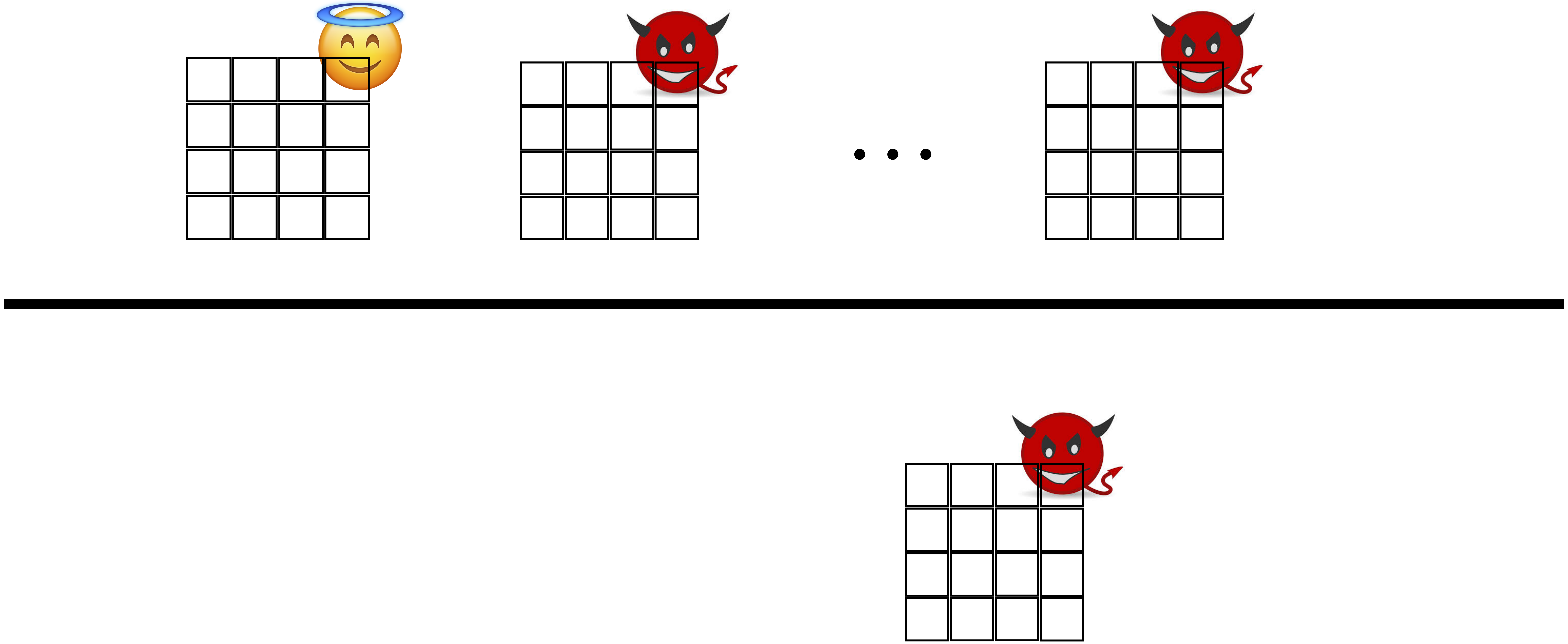
How to define authentic data?

Multi-server schemes: honest server's view of the database.



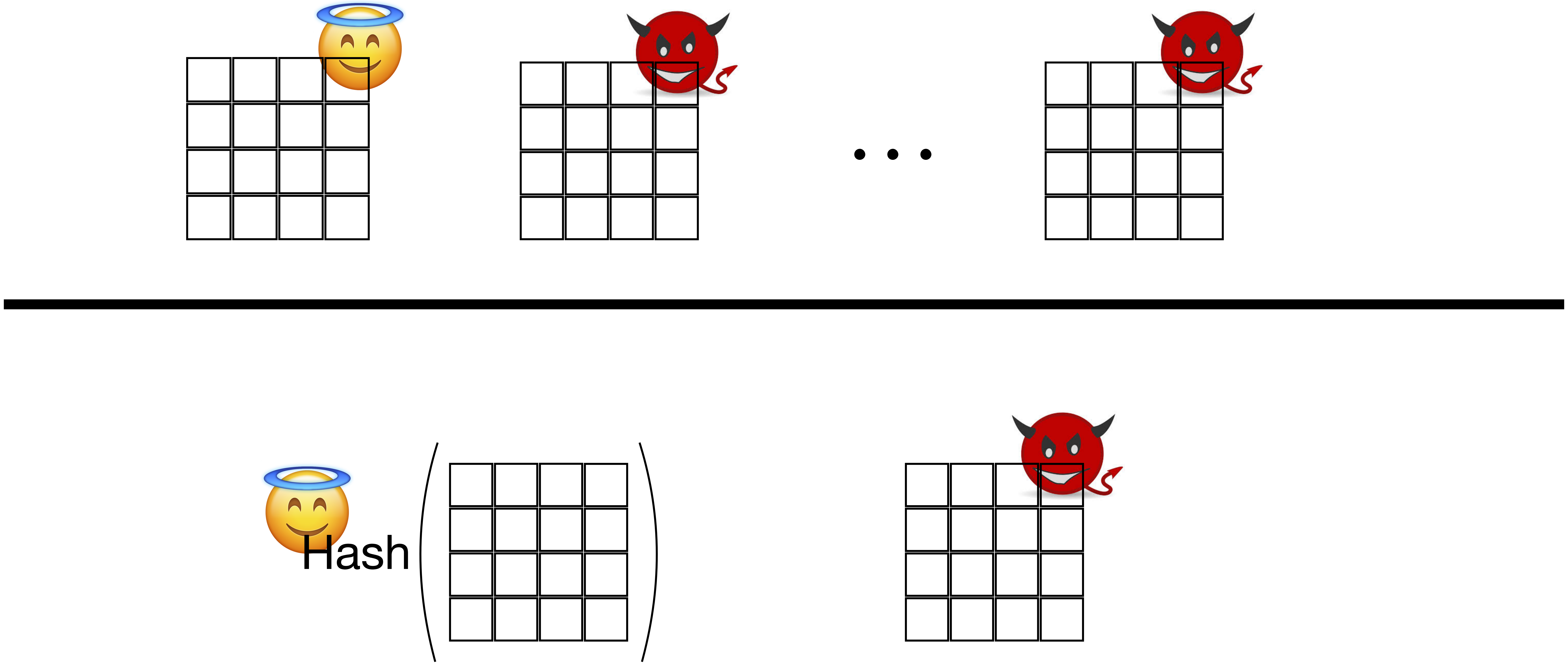
How to define authentic data?

Multi-server schemes: honest server’s view of the database.



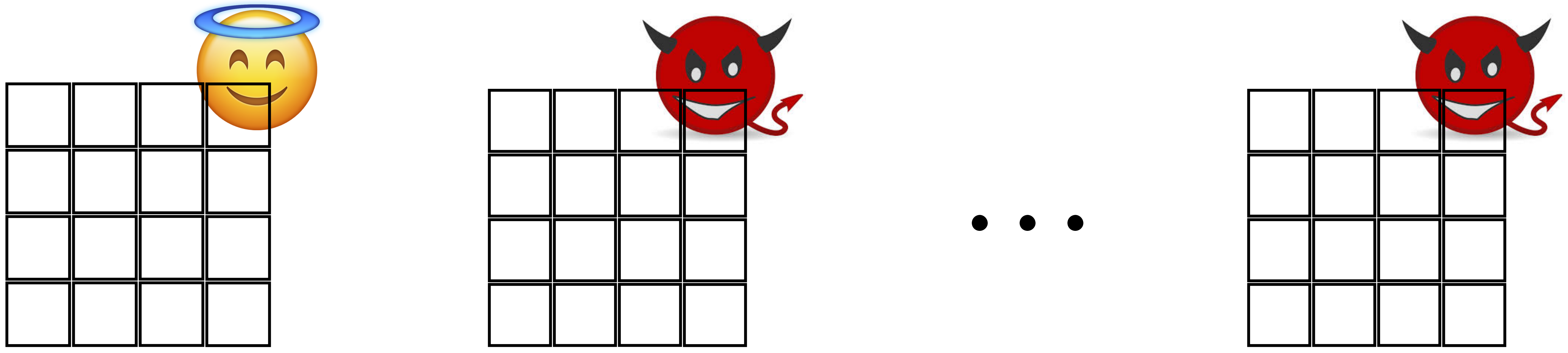
How to define authentic data?

Multi-server schemes: honest server's view of the database.

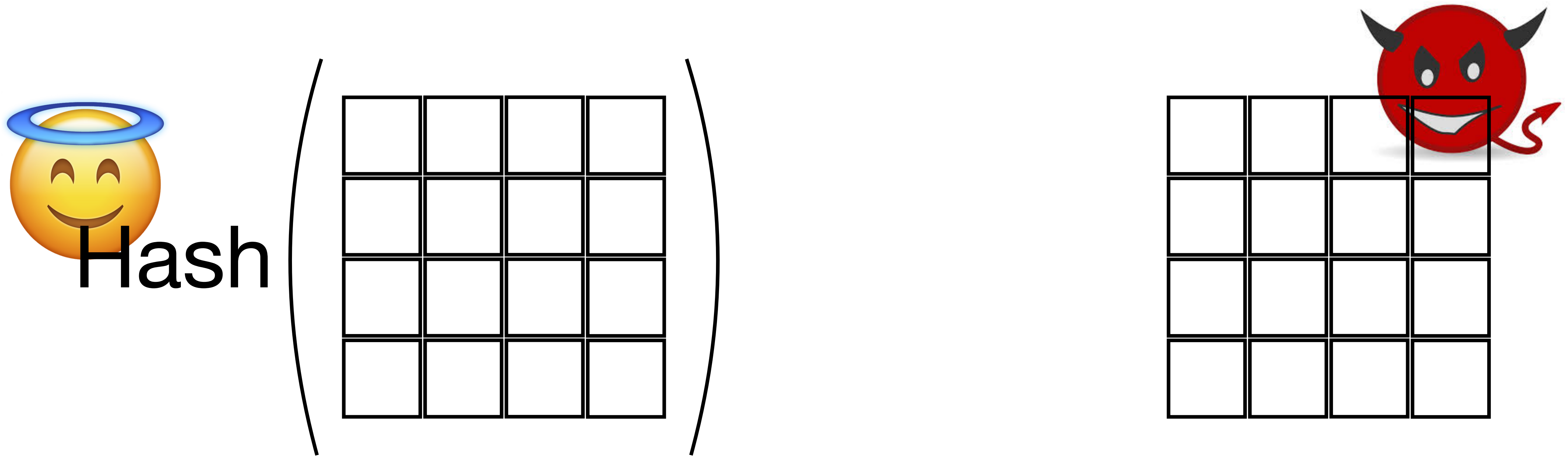


How to define authentic data?

Multi-server schemes: honest server’s view of the database.



Single-server schemes: digest of the true database.



Our results: multi-server schemes

Our results: multi-server schemes

(1) Multi-servers, single-record query

Given a Merkle-tree scheme, on a database of size N

- the per-query communication is $O(\log N)$, same as unauthenticated PIR,
- the integrity error is negligible.

See paper

Our results: multi-server schemes

(1) Multi-servers, single-record query

Given a Merkle-tree scheme, on a database of size N

- the per-query communication is $O(\log N)$, same as unauthenticated PIR,
- the integrity error is negligible.

See paper

(2) Two-servers, single-record and aggregate queries

Given PRG and a field $\mathbb{F}$, on a database of size N

- the per-query communication is $O(\log N)$, same as unauthenticated PIR,
- the integrity error is $1/|\mathbb{F}|$

This talk (roughly)

Our results: single-server schemes

Our results: single-server schemes

(3) Single-record query from LWE

Under the LWE secret dimension s and ciphertext modulus q, on a N-bit database

- the client downloads a one-time digest of size $n\sqrt{N}$ elements of $\mathbb{Z}_q$,
- the per-query communication cost is $2\sqrt{N}$ elements of $\mathbb{Z}_q$,
- the integrity error is roughly $\sqrt{N}/q$, can be amplified generically.

See paper

Our results: single-server schemes

(3) Single-record query from LWE

Under the LWE secret dimension s and ciphertext modulus q, on a N-bit database

- the client downloads a one-time digest of size $n\sqrt{N}$ elements of $\mathbb{Z}_q$,
- the per-query communication cost is $2\sqrt{N}$ elements of $\mathbb{Z}_q$,
- the integrity error is roughly $\sqrt{N}/q$, can be amplified generically.

See paper

(4) Single-record query from DDH

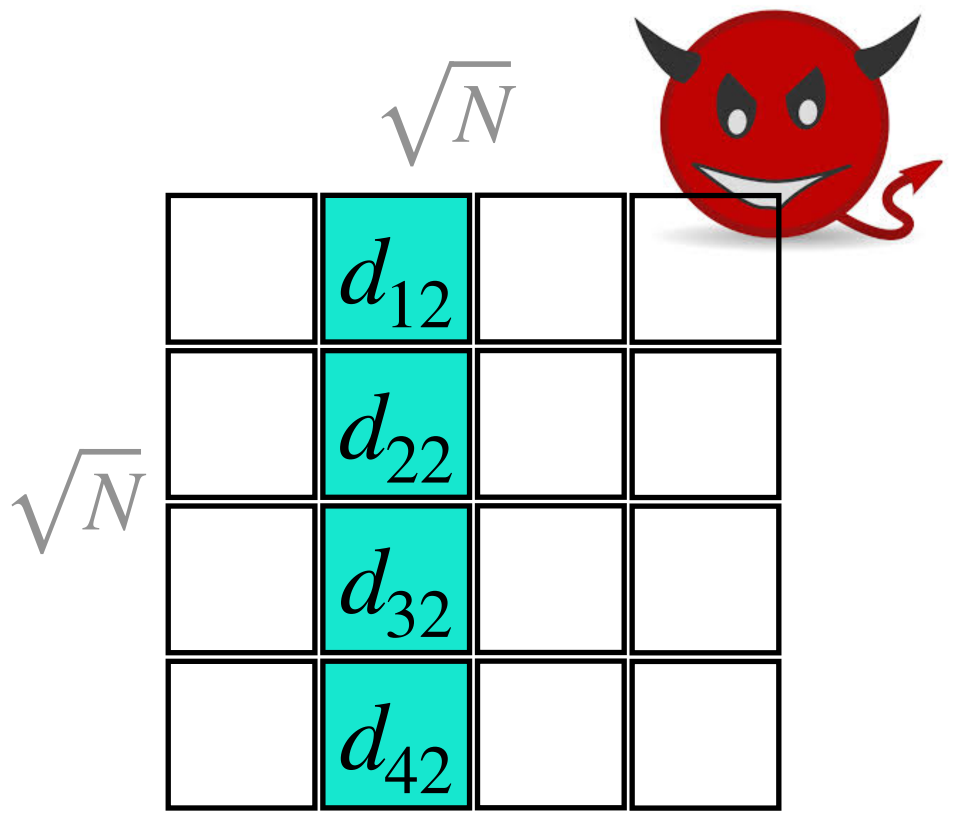
Under the DDH assumption in a group $\mathbb{G}$, on a N-bit database

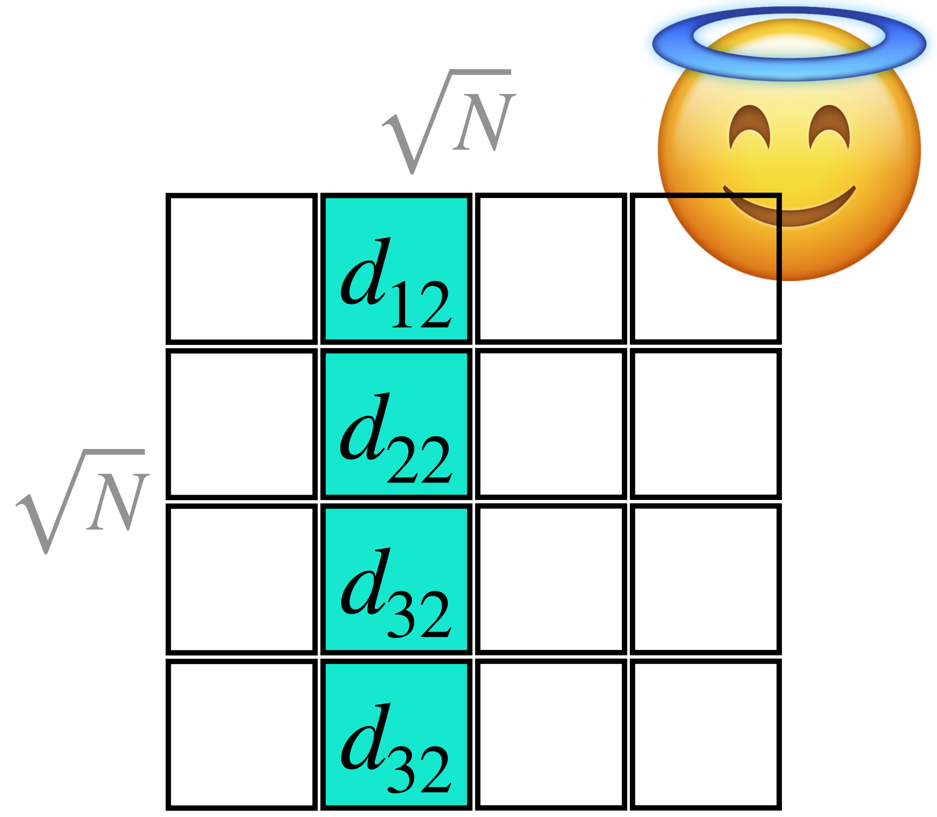
- the client downloads a one-time digest of size $\sqrt{N}$ elements of $\mathbb{G}$,
- the per-query communication cost is $2\sqrt{N}$ elements of $\mathbb{G}$,
- the integrity error is negligible.

See paper

Classic multi-server PIR [CGKS95]

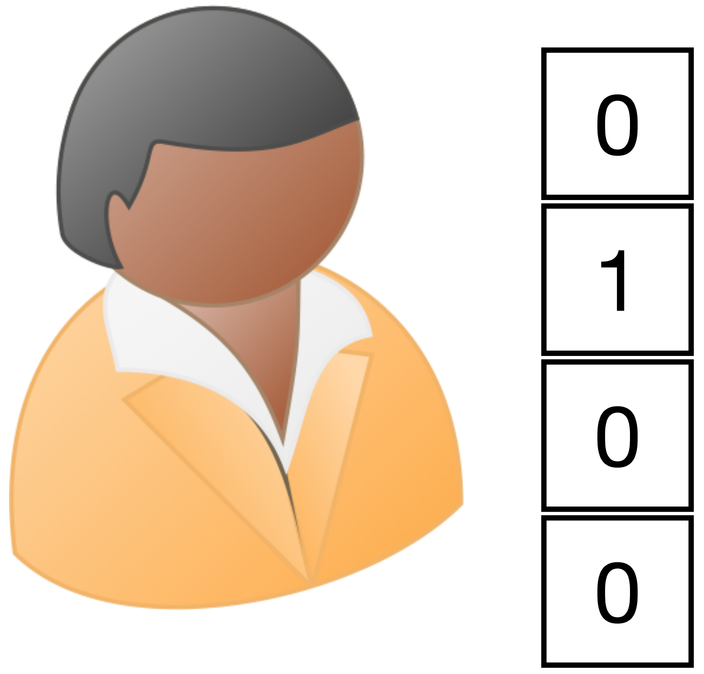
pk

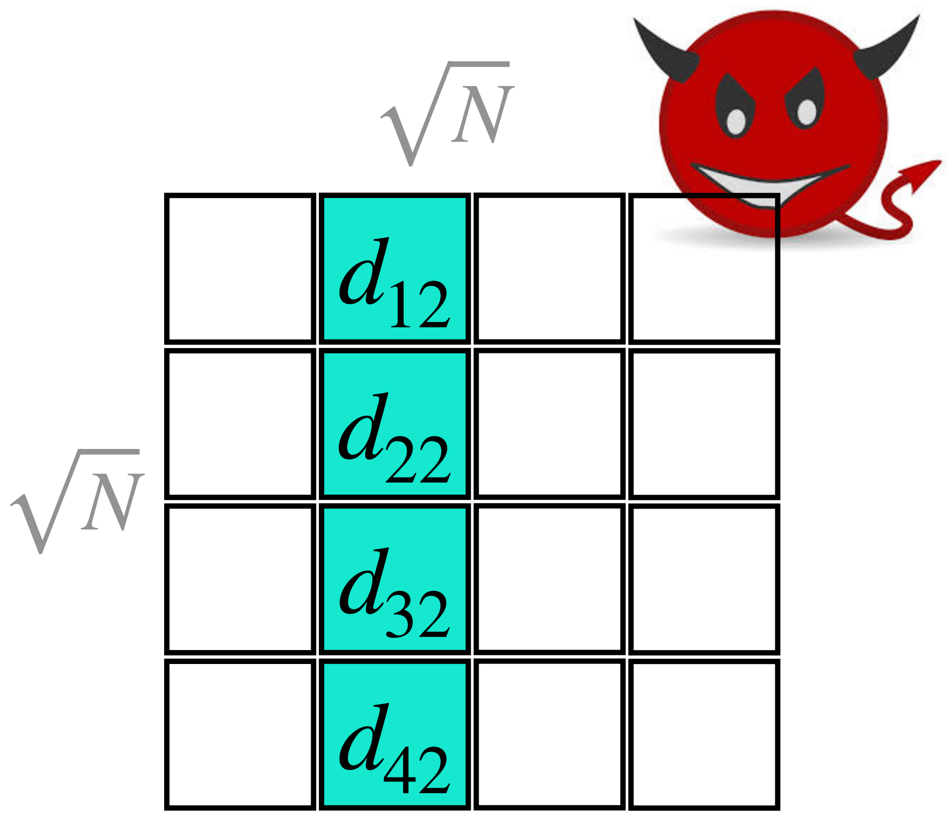


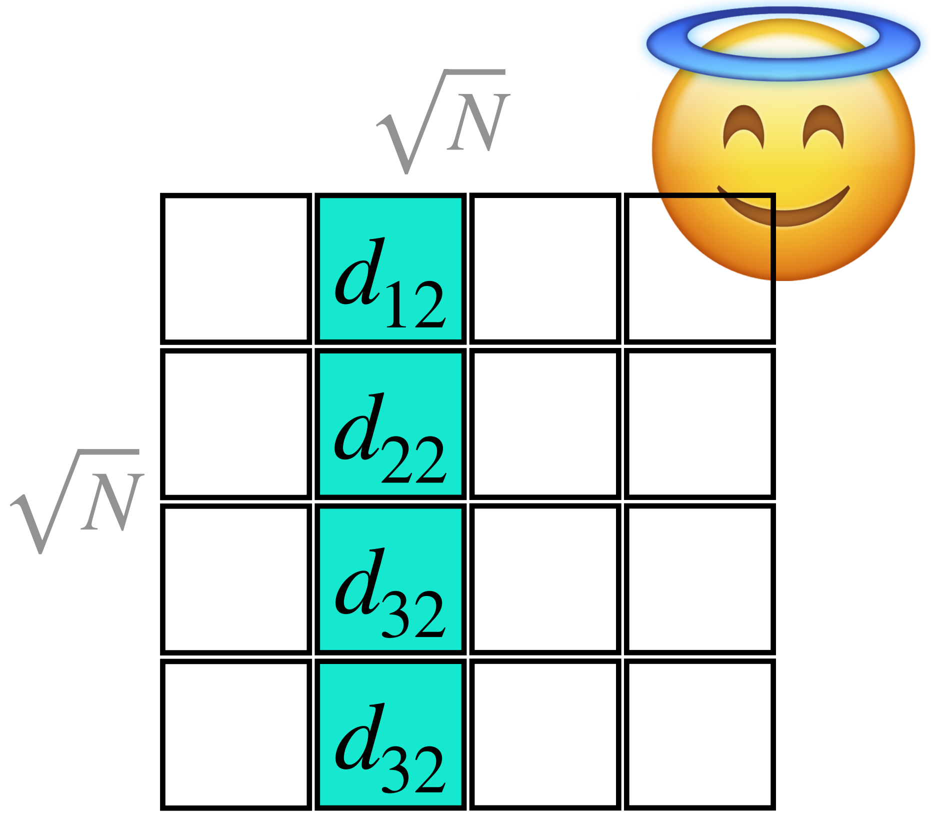


Classic multi-server PIR [CGKS95]

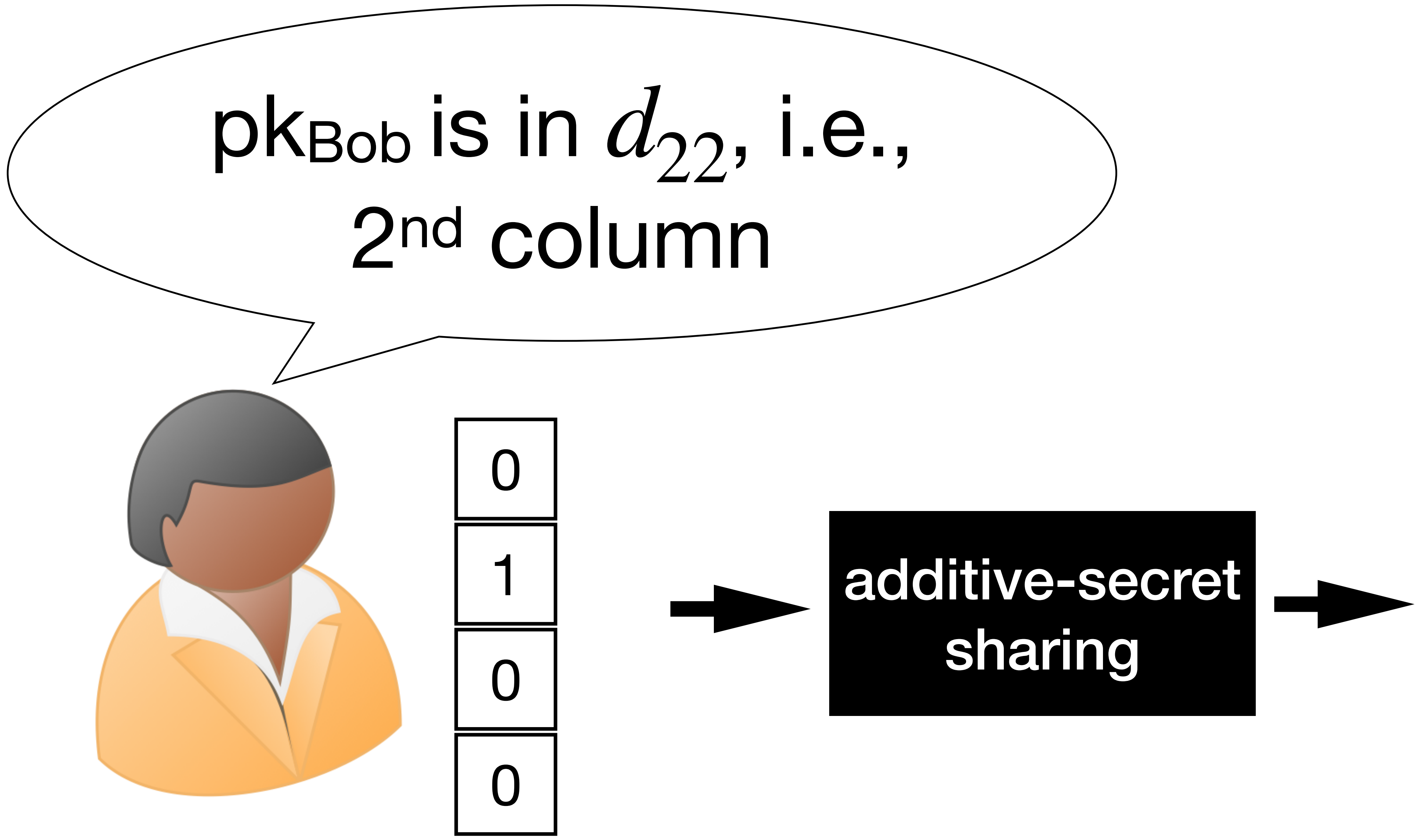
pk_{Bob} is in d_{22}, i.e.,
2nd column



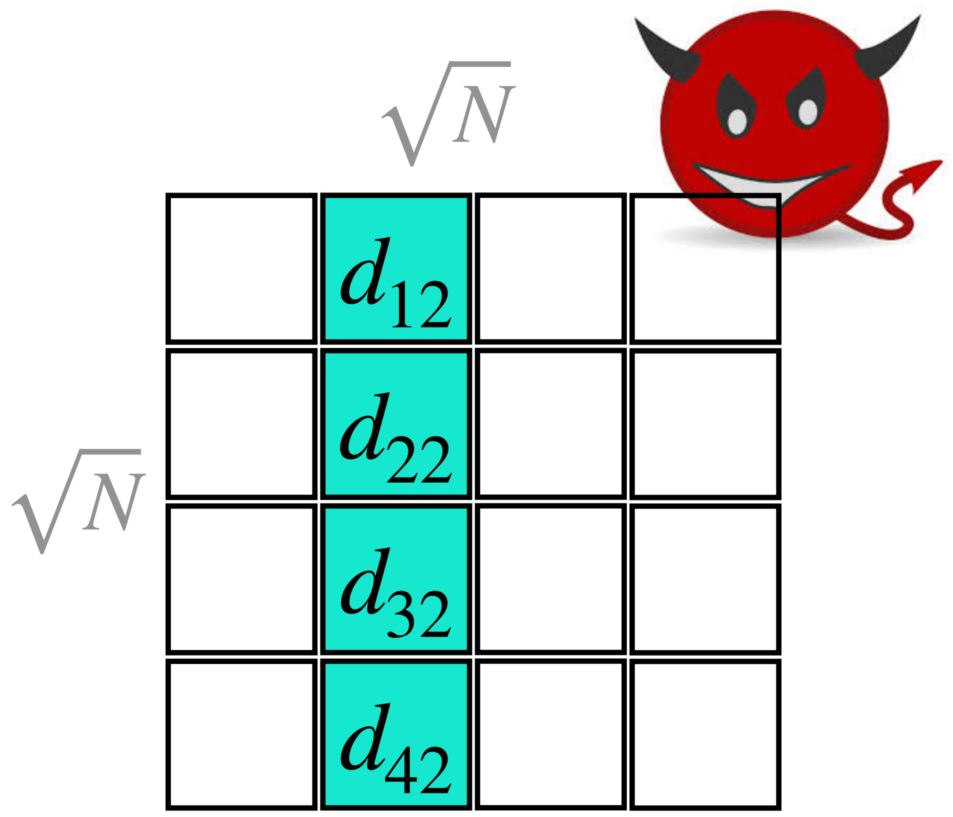


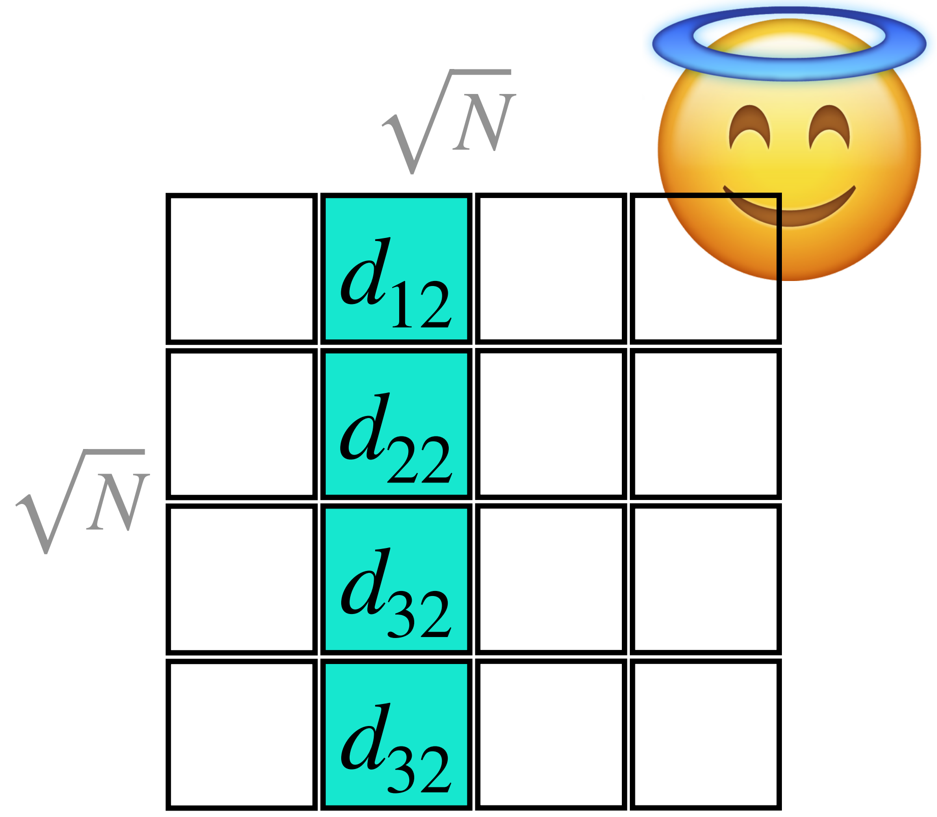


Classic multi-server PIR [CGKS95]



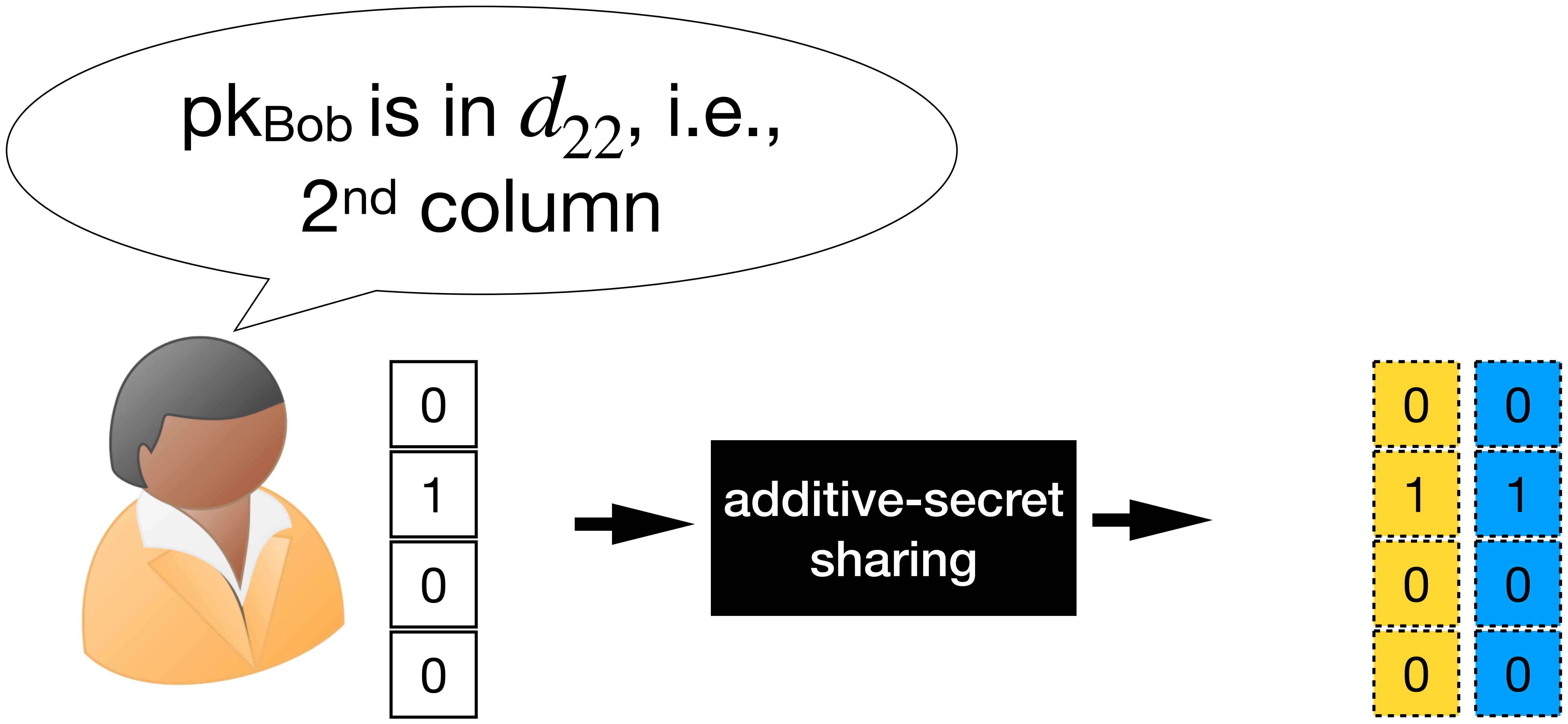
pk$_{\text{Bob}}$ is in d_{22}, i.e., 2nd column

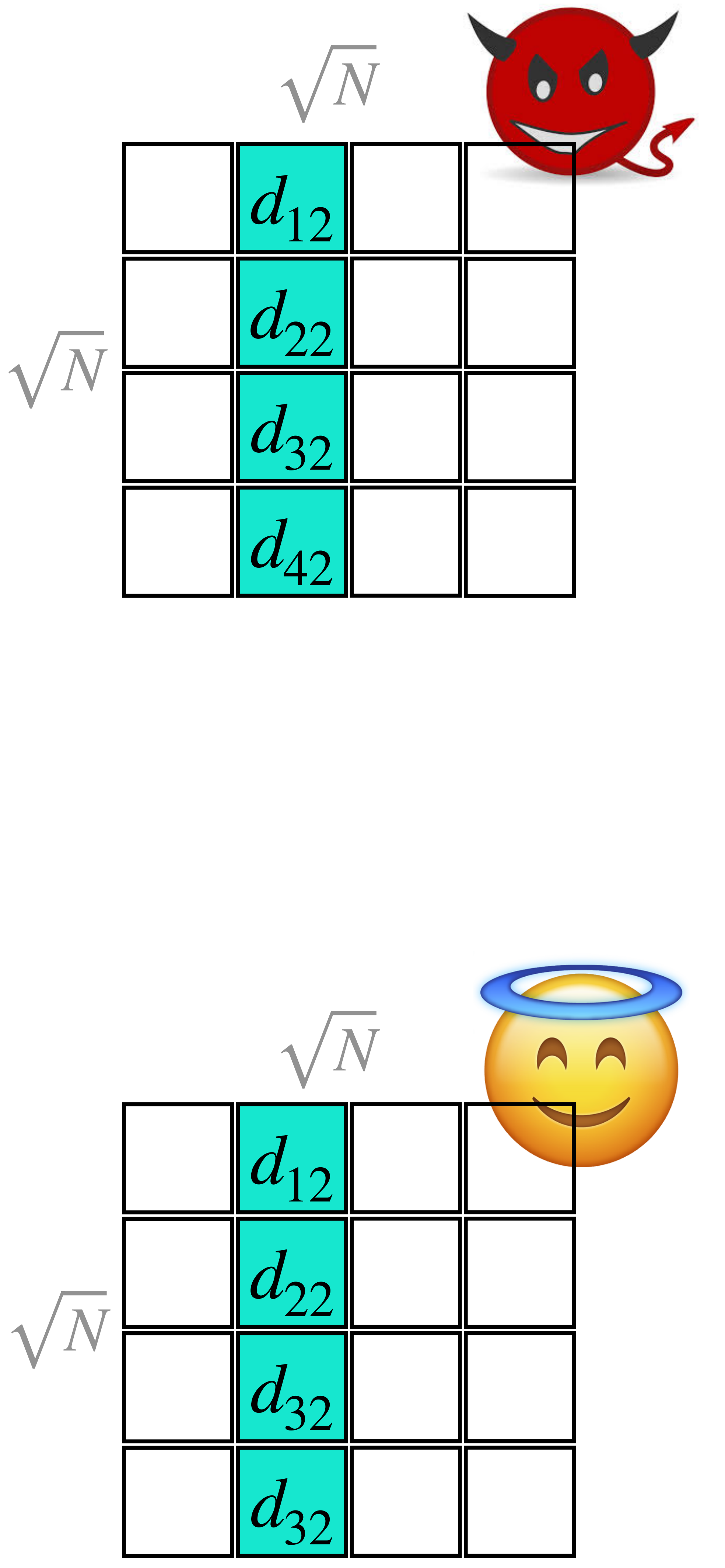




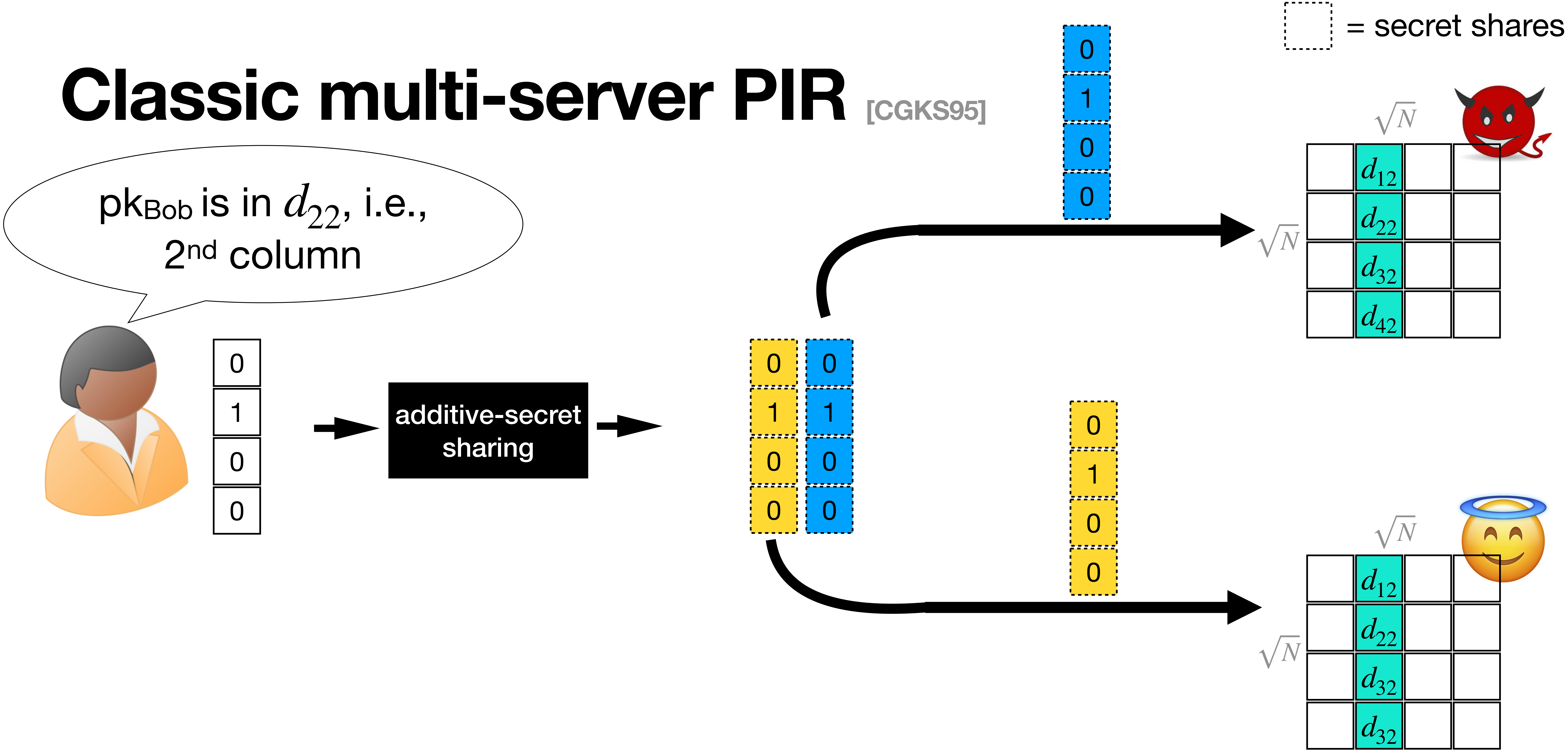
⬚ = secret shares

Classic multi-server PIR [CGKS95]



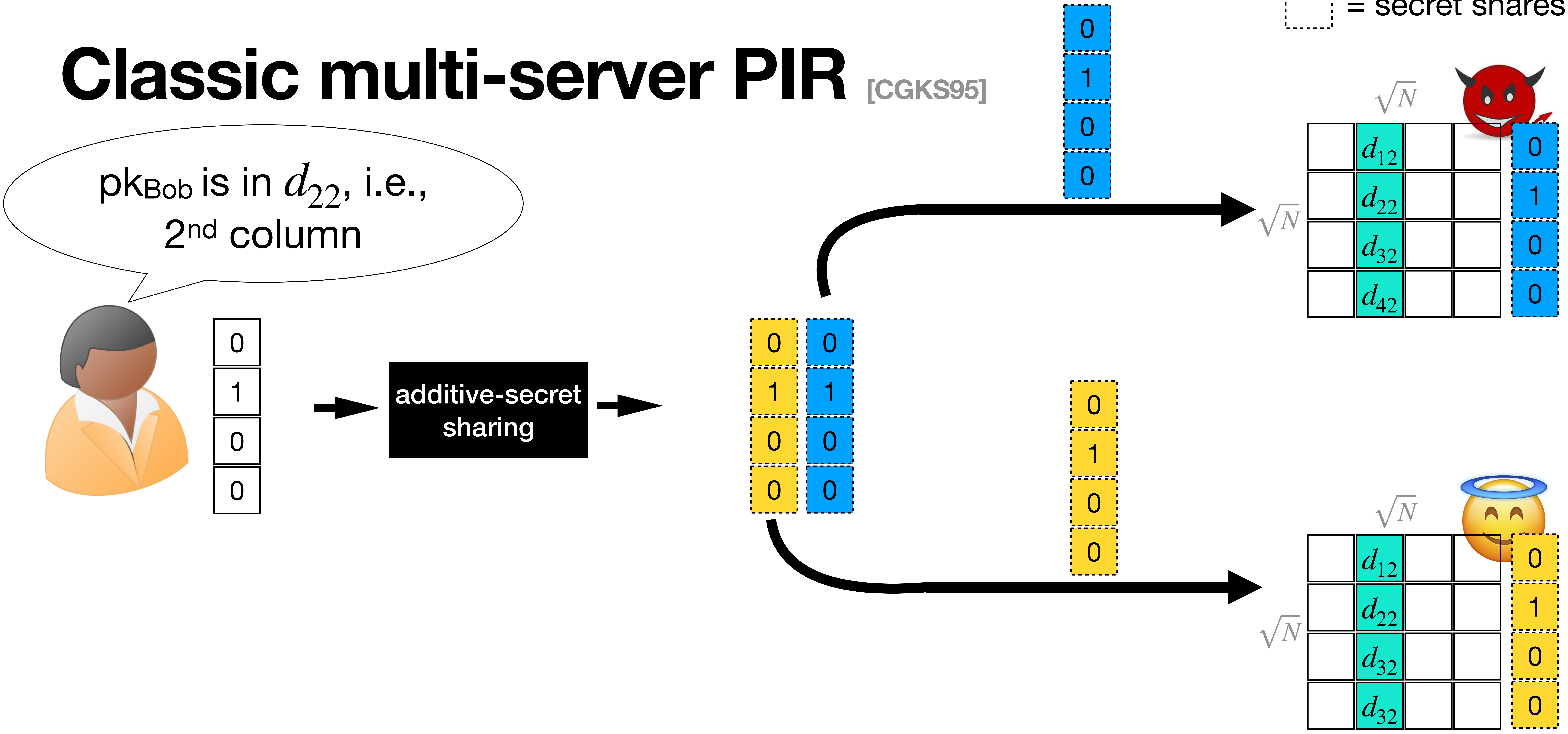


Classic multi-server PIR [CGKS95]



Classic multi-server PIR [CGKS95]

⬚ = secret shares

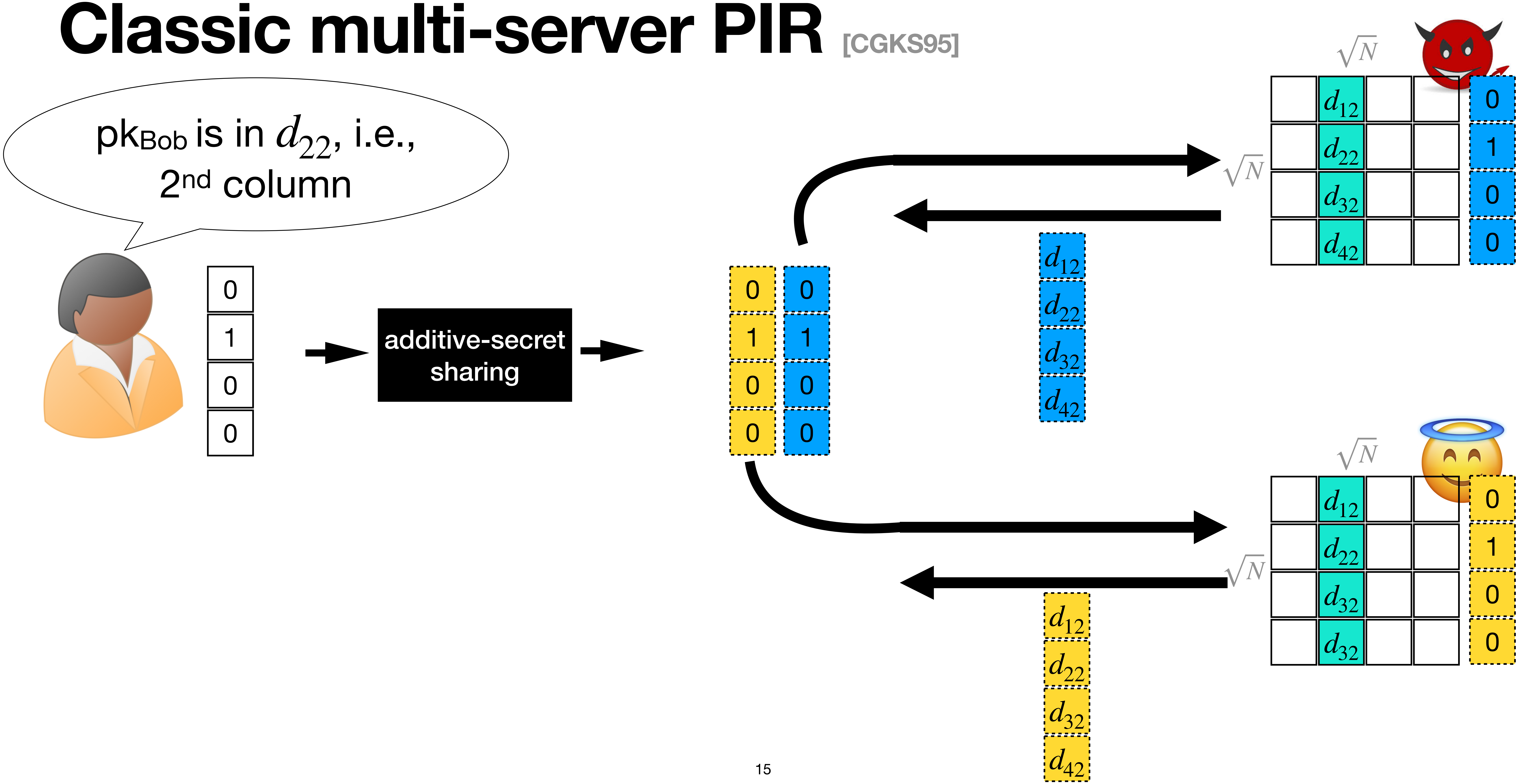


pkBob is in d_{22}, i.e.,
2nd column

additive-secret sharing

⬚ = secret shares

Classic multi-server PIR [CGKS95]



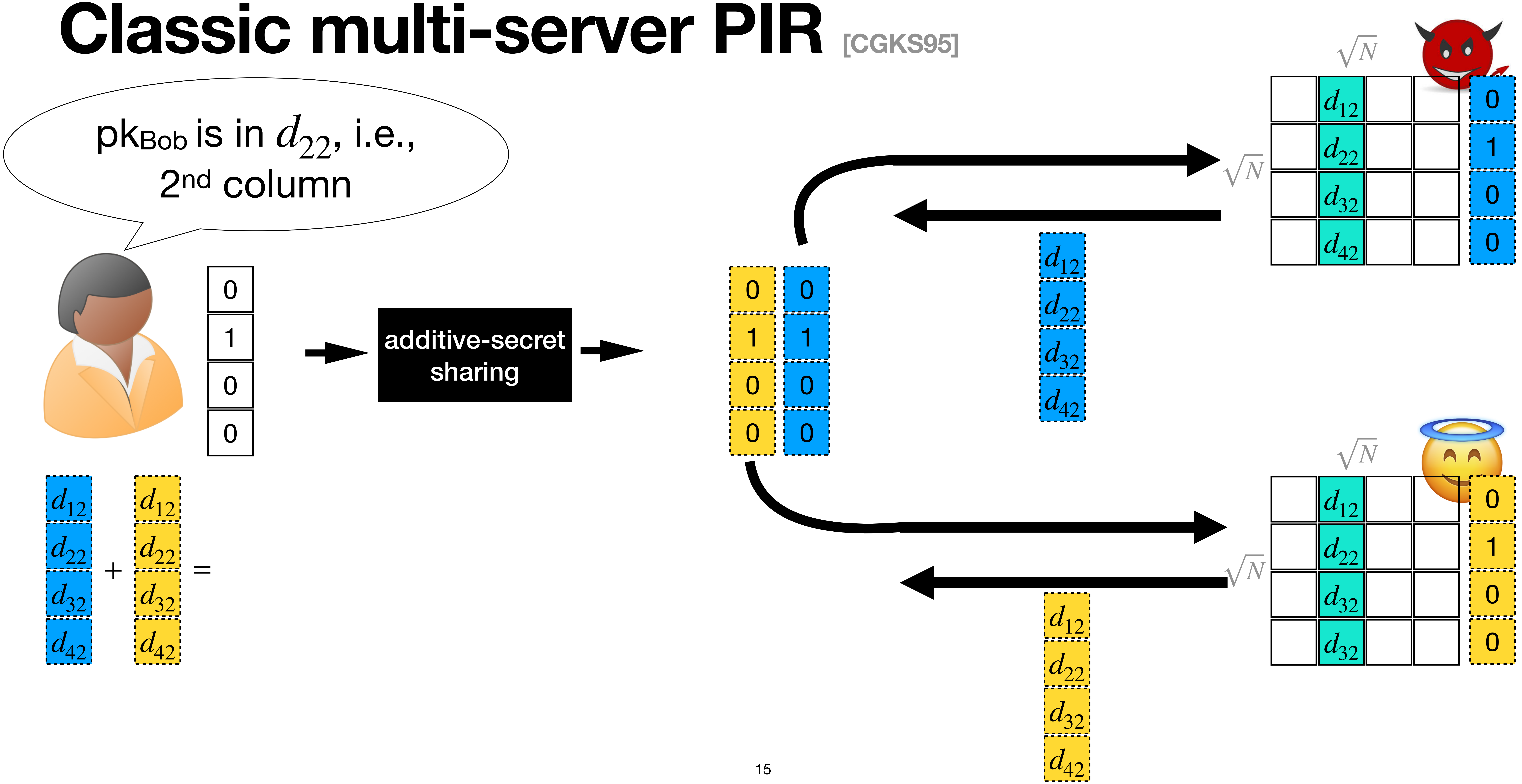
pkBob is in d_{22}, i.e., 2nd column

additive-secret sharing

⬚ = secret shares

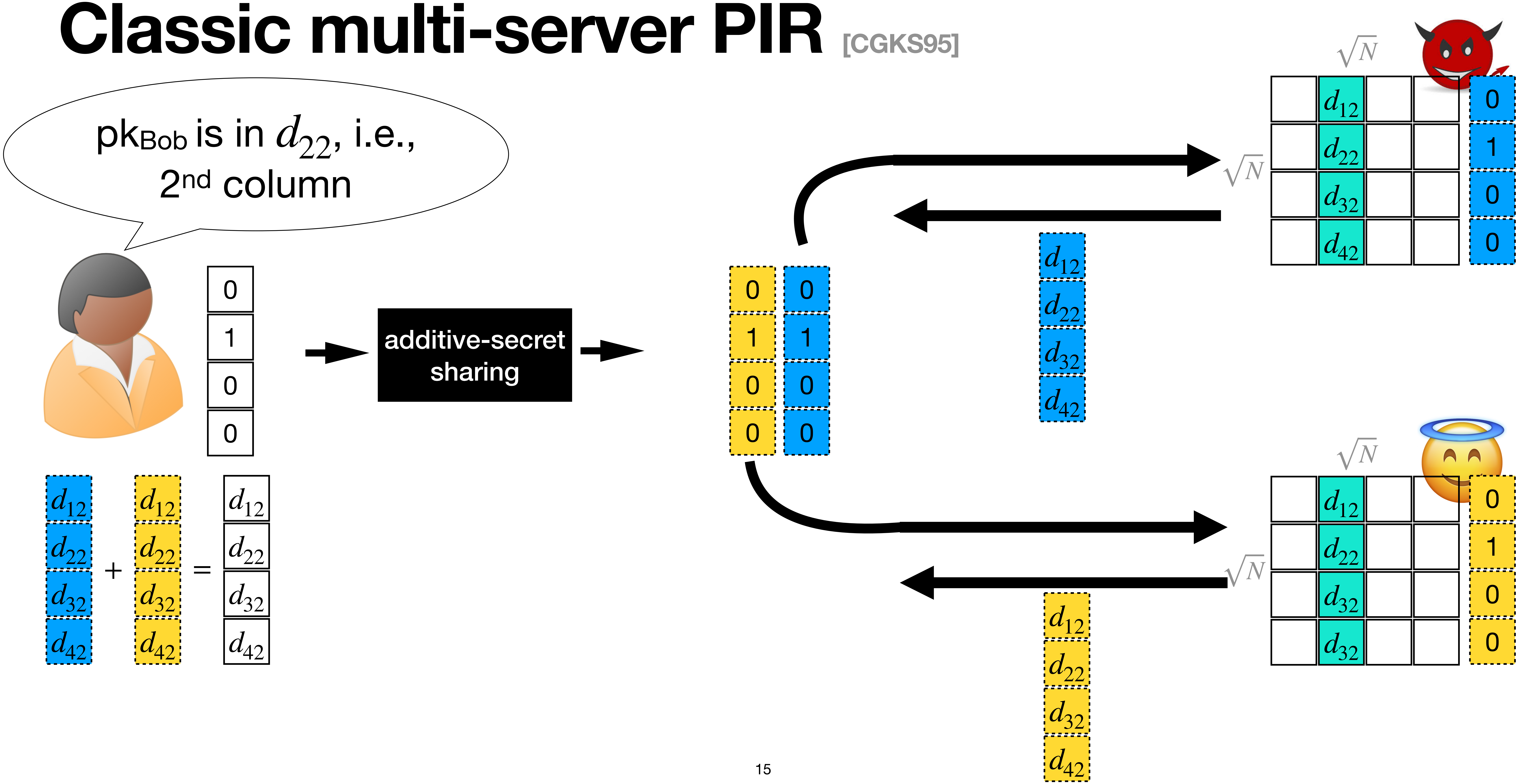
Classic multi-server PIR [CGKS95]

pk_{Bob} is in d_{22}, i.e., 2nd column



⬚ = secret shares

Classic multi-server PIR [CGKS95]



pk_{Bob} is in d_{22}, i.e.,
2nd column

0
1
0
0

additive-secret sharing

d_{12} d_{22} d_{32} d_{42} + d_{12} d_{22} d_{32} d_{42} = d_{12} d_{22} d_{32} d_{42}

0 1 0 0

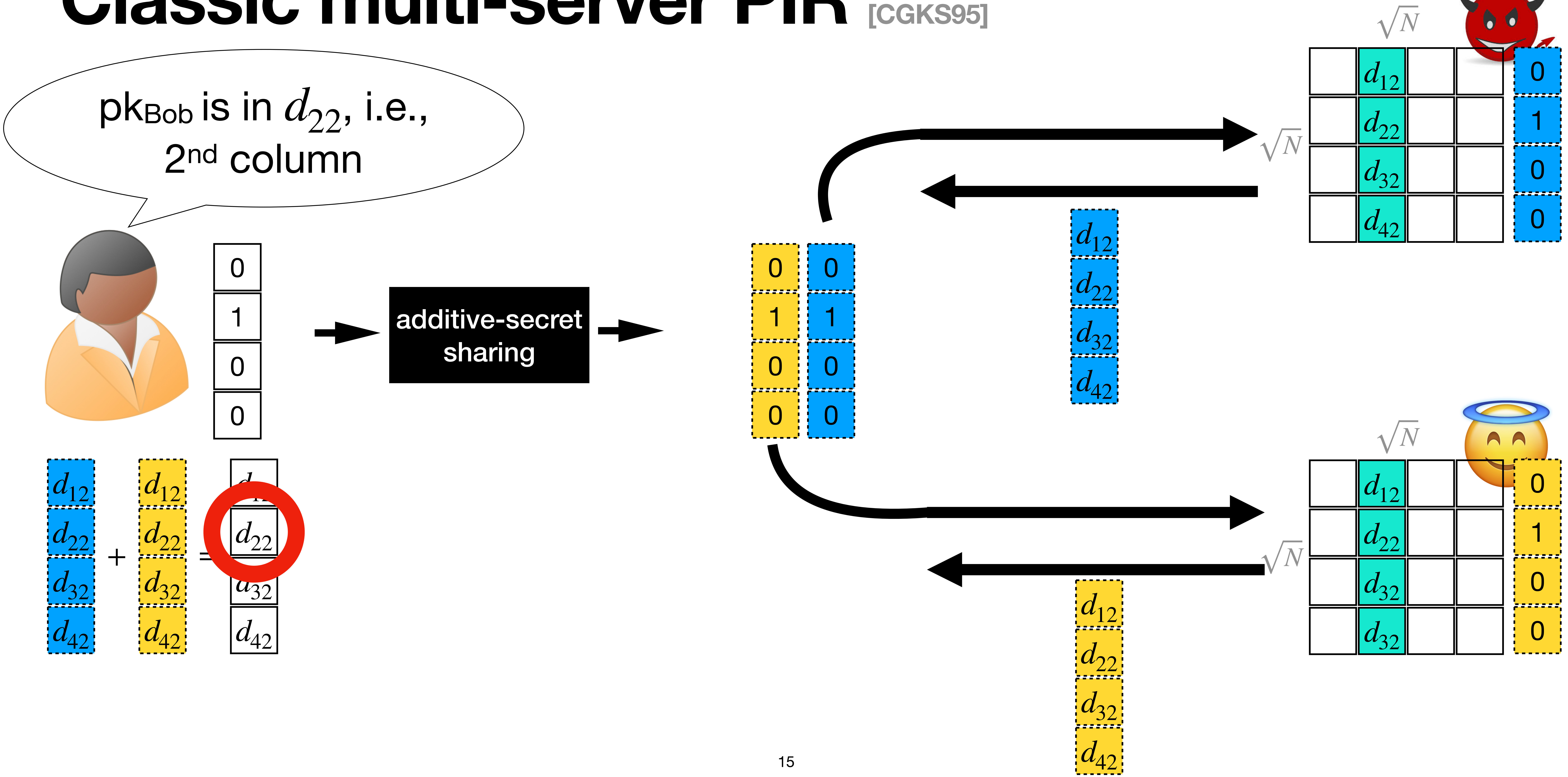
0 1 0 0

d_{12} d_{22} d_{32} d_{42}

d_{12} d_{22} d_{32} d_{42}

⬚ = secret shares

Classic multi-server PIR [CGKS95]

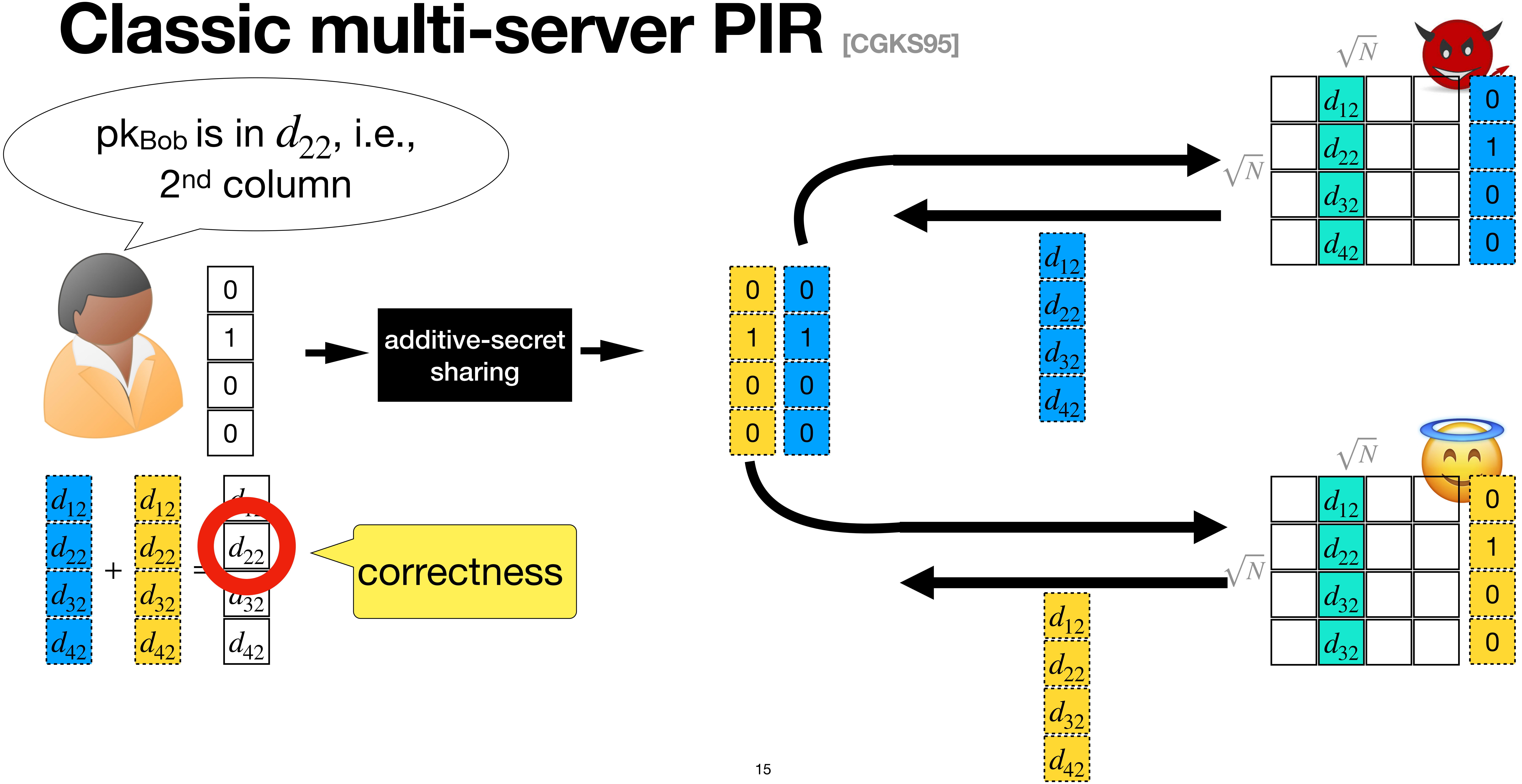


pkBob is in d_{22}, i.e., 2nd column

0
1
0
0
additive-secret sharing
0 0
1 1
0 0
0 0
d_{12} d_{22} d_{32} d_{42} + d_{12} d_{22} d_{32} d_{42} = d_{12} d_{22} d_{32} d_{42}
$\sqrt{N}$
$\sqrt{N}$
d_{12} d_{22} d_{32} d_{42}
0 1 0 0
d_{12} d_{22} d_{32} d_{42}
$\sqrt{N}$
$\sqrt{N}$
d_{12} d_{22} d_{32} d_{32}
0 1 0 0
d_{12} d_{22} d_{32} d_{42}

□ = secret shares

Classic multi-server PIR [CGKS95]


pkBob is in d_{22}, i.e.,
2nd column

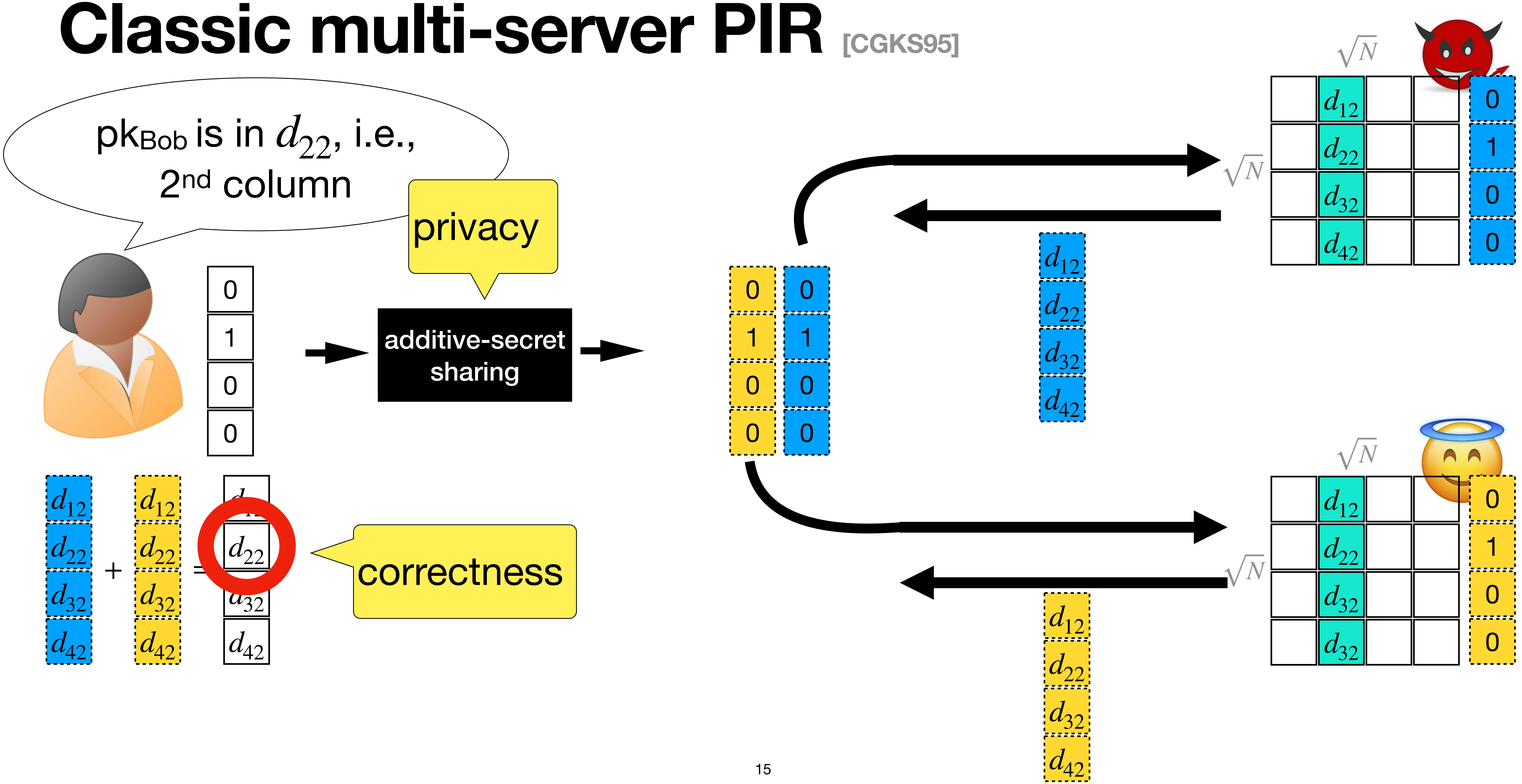
additive-secret sharing

correctness

0
1
0
0
d_{12} d_{22} d_{32} d_{42} + d_{12} d_{22} d_{32} d_{42} = d_{12} d_{22} d_{32} d_{42}
0 1 0 0
0 1 0 0
$\sqrt{N}$
$\sqrt{N}$
d_{12} d_{22} d_{32} d_{42}
0 1 0 0
d_{12} d_{22} d_{32} d_{42}
$\sqrt{N}$
$\sqrt{N}$
d_{12} d_{22} d_{32} d_{32}
0 1 0 0
d_{12} d_{22} d_{32} d_{42}

⬚ = secret shares

Classic multi-server PIR [CGKS95]



pkBob is in d_{22}, i.e., 2nd column

privacy

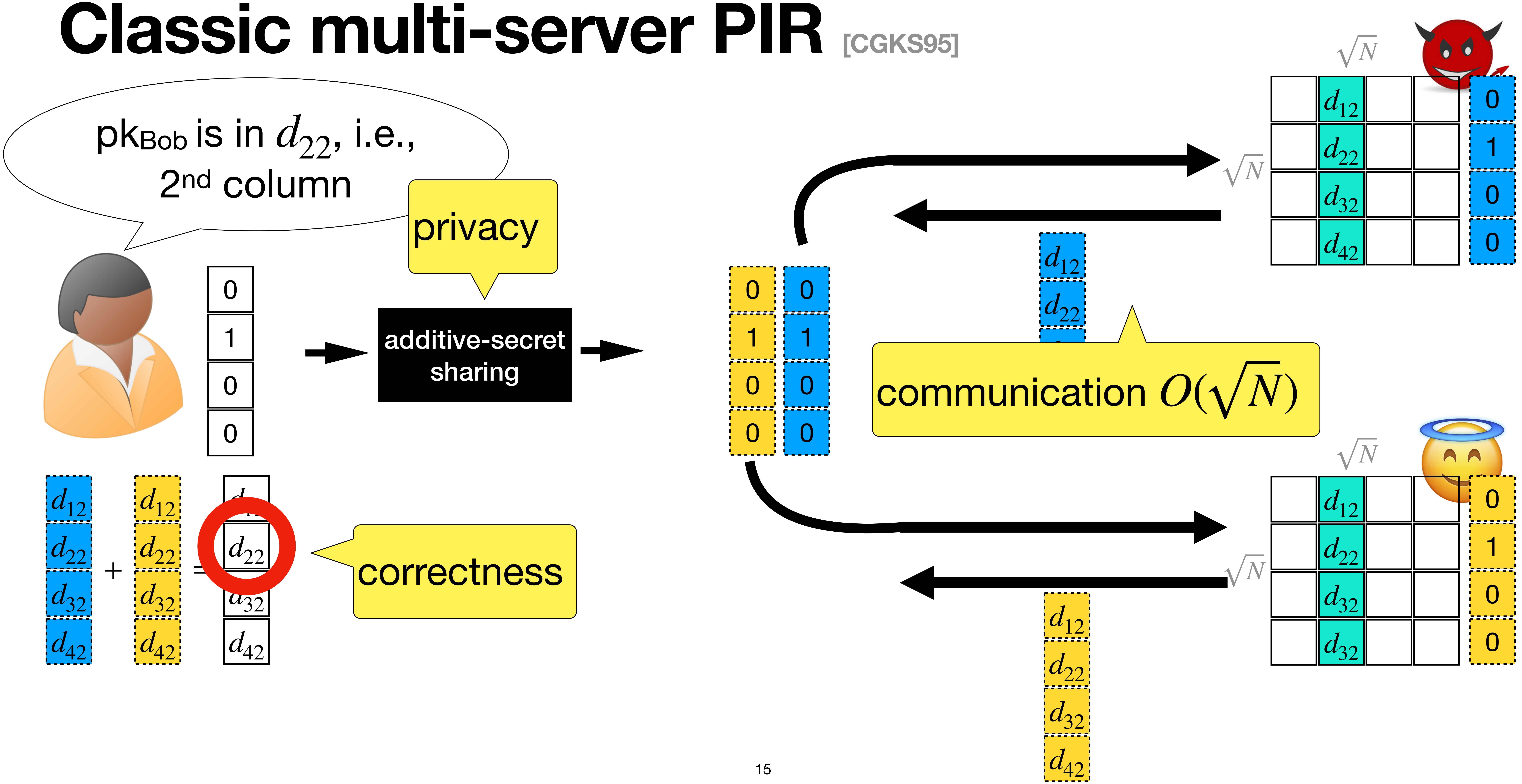
correctness

additive-secret sharing

0
1
0
0
d_{12} d_{22} d_{32} d_{42} + d_{12} d_{22} d_{32} d_{42} = d_{12} d_{22} d_{32} d_{42}
0 1 0 0
0 1 0 0
$\sqrt{N}$
$\sqrt{N}$
d_{12} d_{22} d_{32} d_{42}
0 1 0 0
d_{12} d_{22} d_{32} d_{42}
$\sqrt{N}$
$\sqrt{N}$
d_{12} d_{22} d_{32} d_{32}
0 1 0 0
d_{12} d_{22} d_{32} d_{42}

□ = secret shares

Classic multi-server PIR [CGKS95]



pkBob is in d_{22}, i.e., 2nd column

privacy

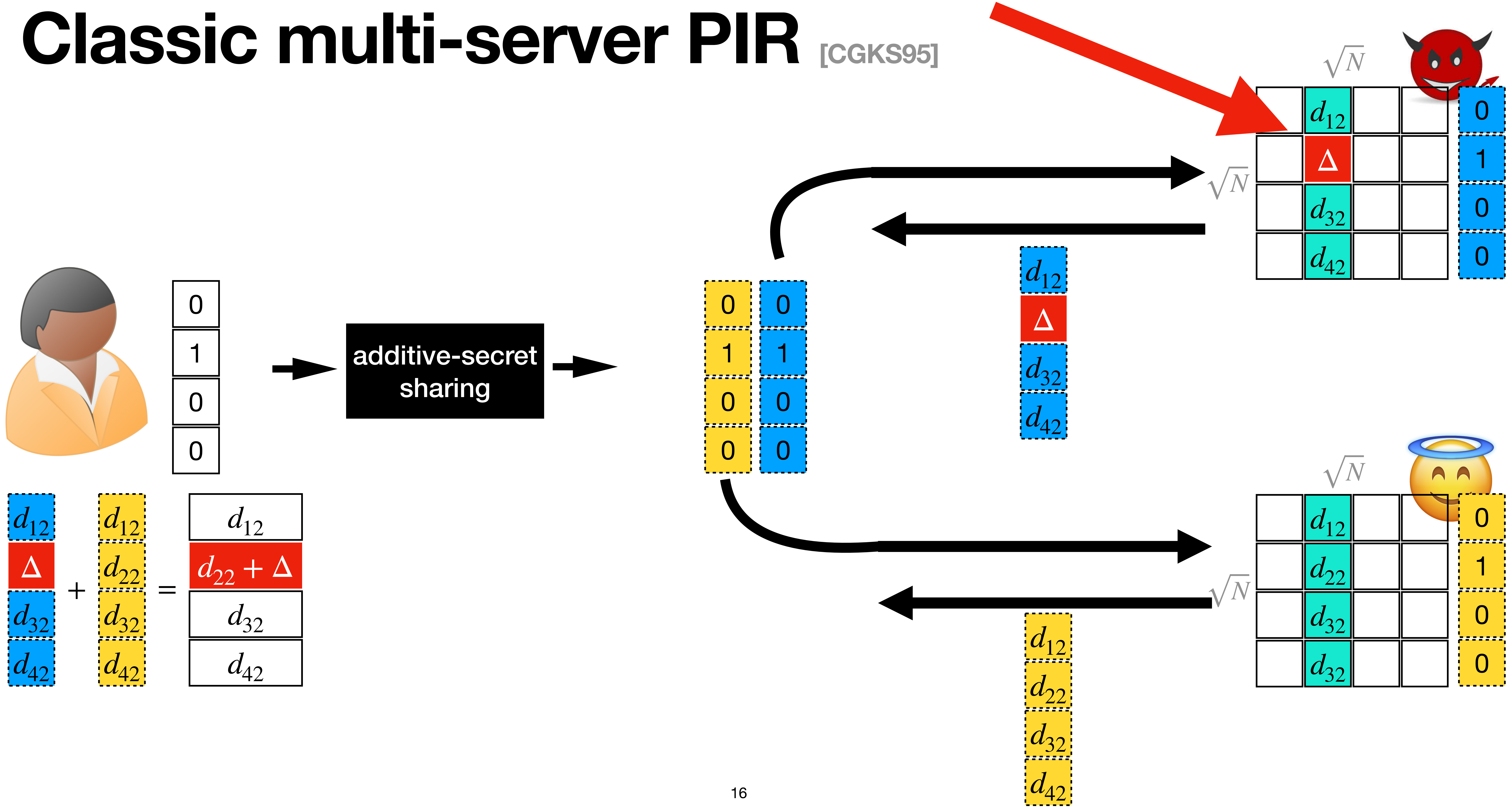
additive-secret sharing

correctness

communication $O(\sqrt{N})$

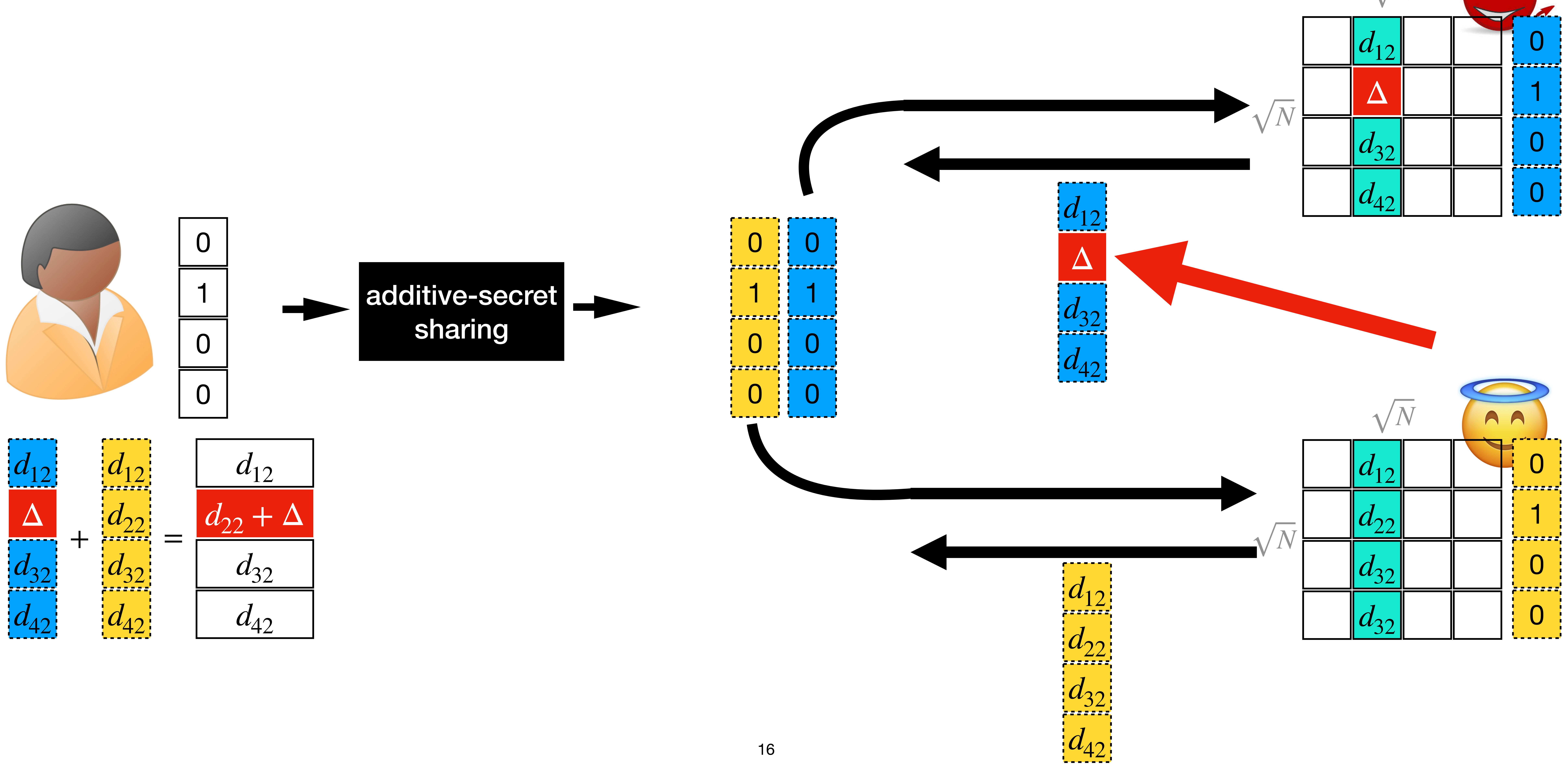
⬚ = secret shares

Classic multi-server PIR [CGKS95]



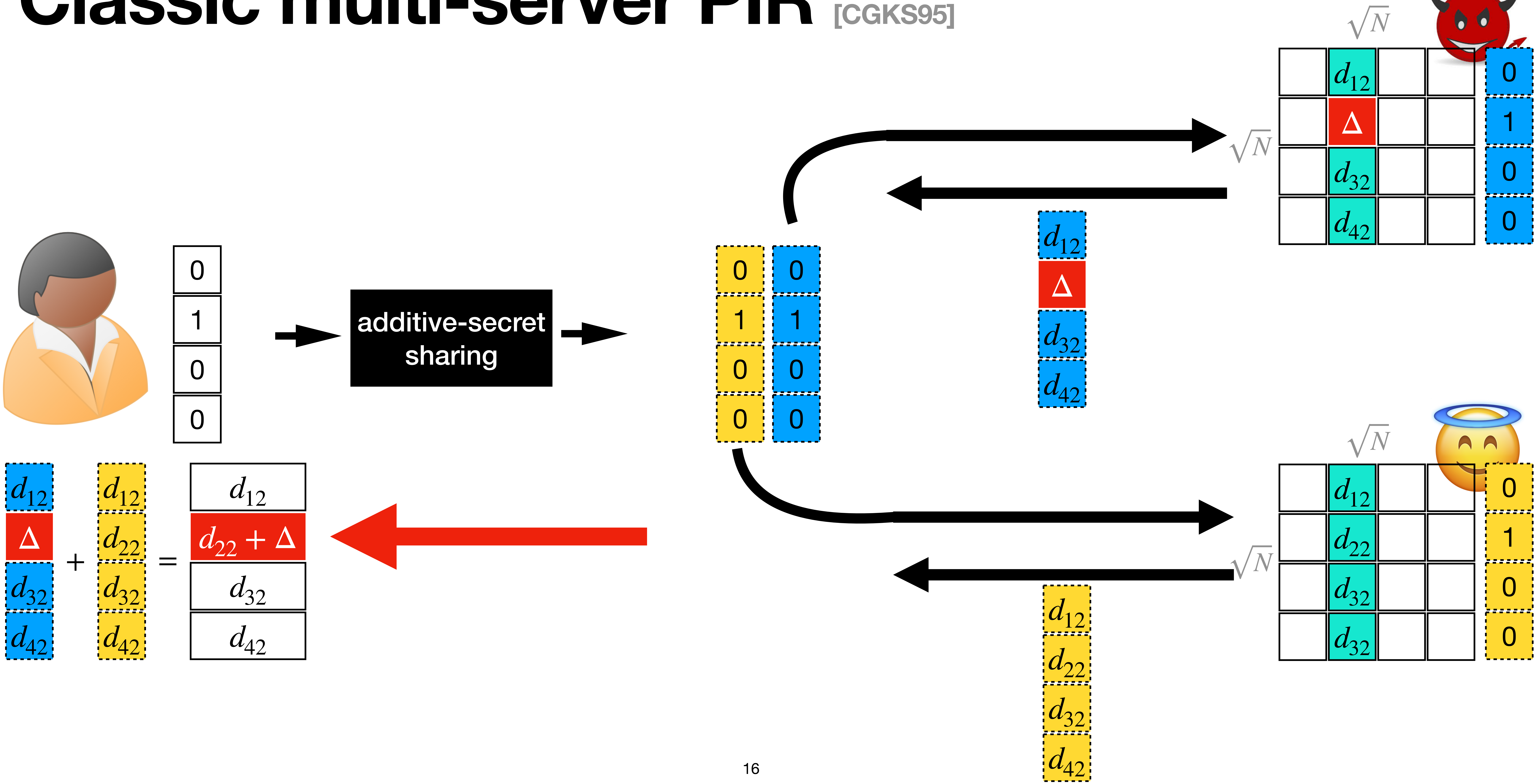
⬚ = secret shares

Classic multi-server PIR [CGKS95]

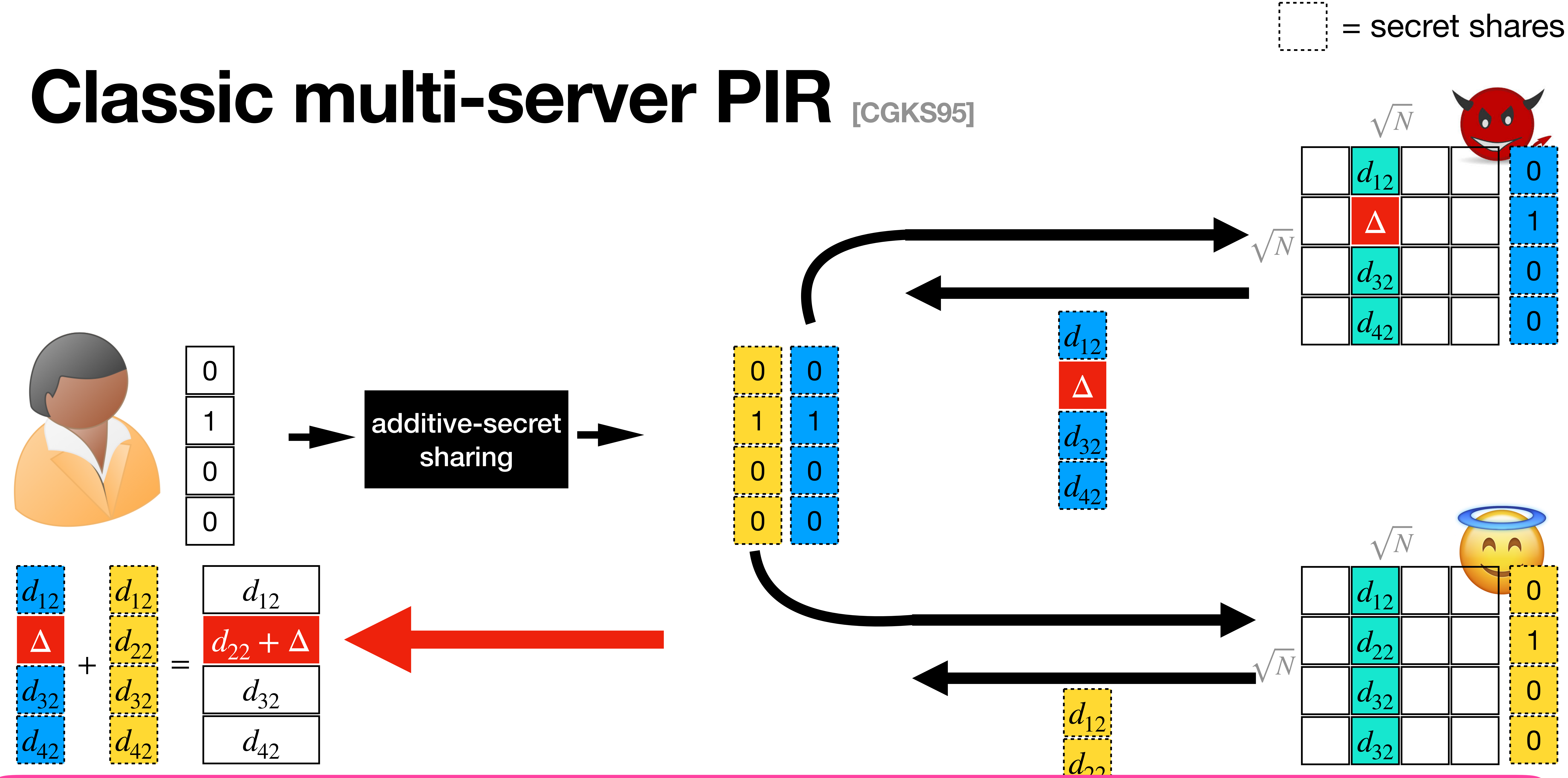


Classic multi-server PIR [CGKS95]

⬚ = secret shares

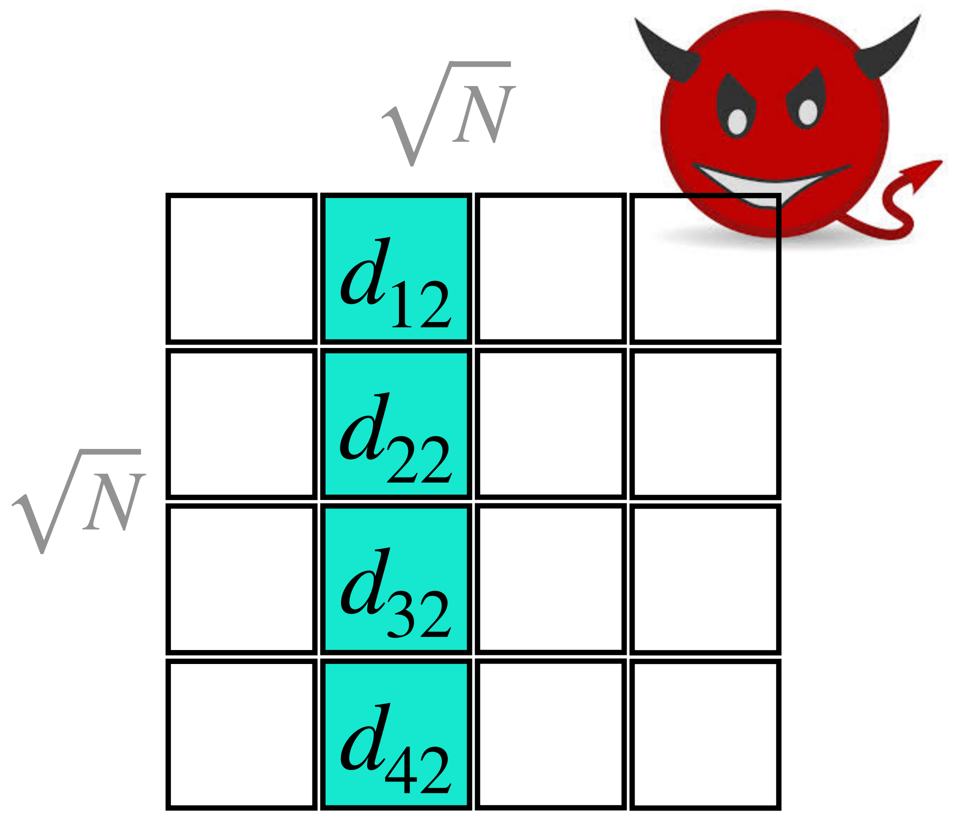


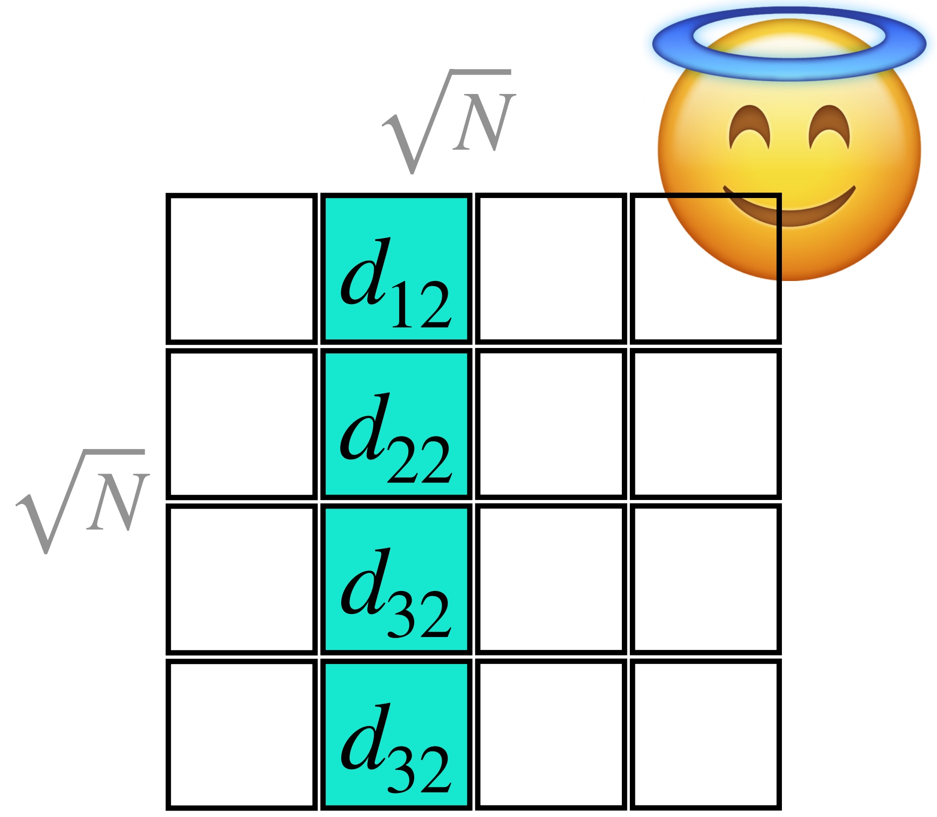
Classic multi-server PIR [CGKS95]



Key idea: two correlated queries, one for data and one to authenticate

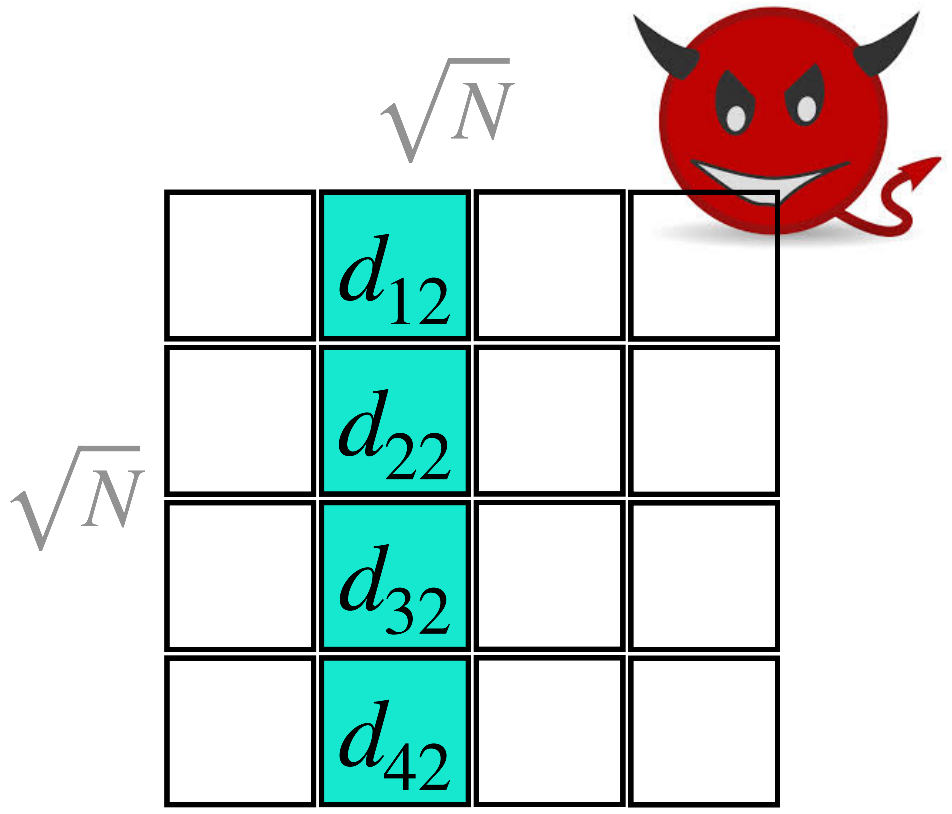
Authenticated multi-server PIR





Authenticated multi-server PIR

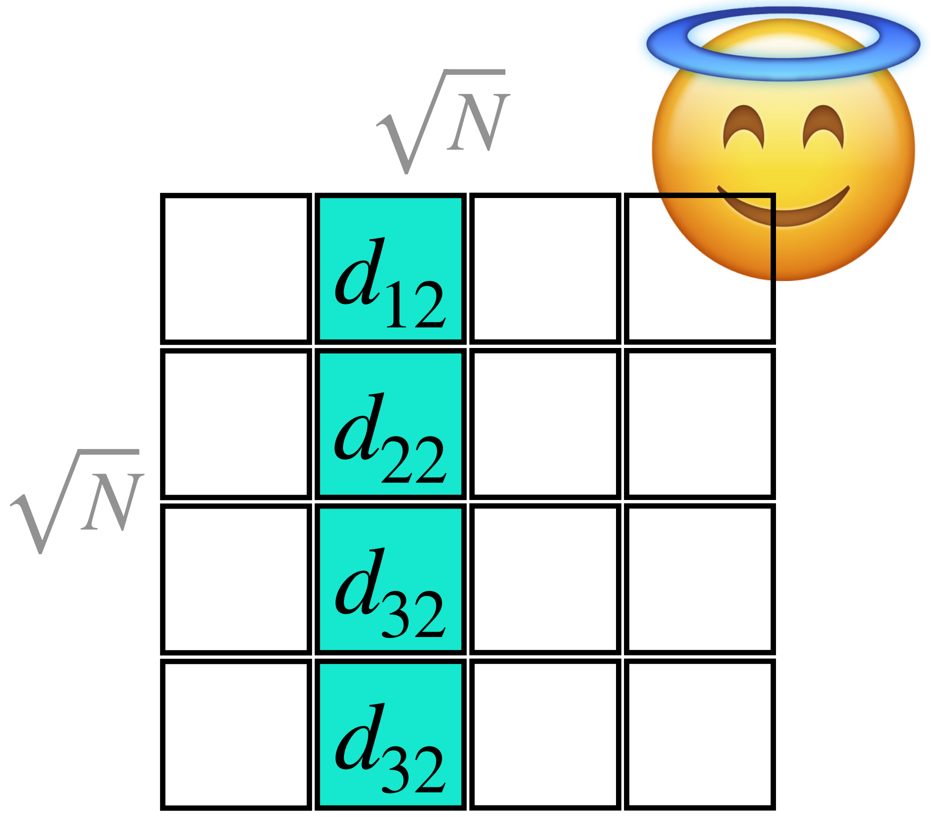
samples random $\alpha \in_R \mathbb{F}$



$\sqrt{N}$

$\sqrt{N}$

	d_{12}		
	d_{22}		
	d_{32}		
	d_{42}		



$\sqrt{N}$

$\sqrt{N}$

	d_{12}		
	d_{22}		
	d_{32}		
	d_{32}		

Authenticated multi-server PIR

samples random $\alpha \in_R \mathbb{F}$

0	0
1	α
0	0
0	0

$\sqrt{N}$ 😈

$\sqrt{N}$

	d_{12}		
	d_{22}		
	d_{32}		
	d_{42}		

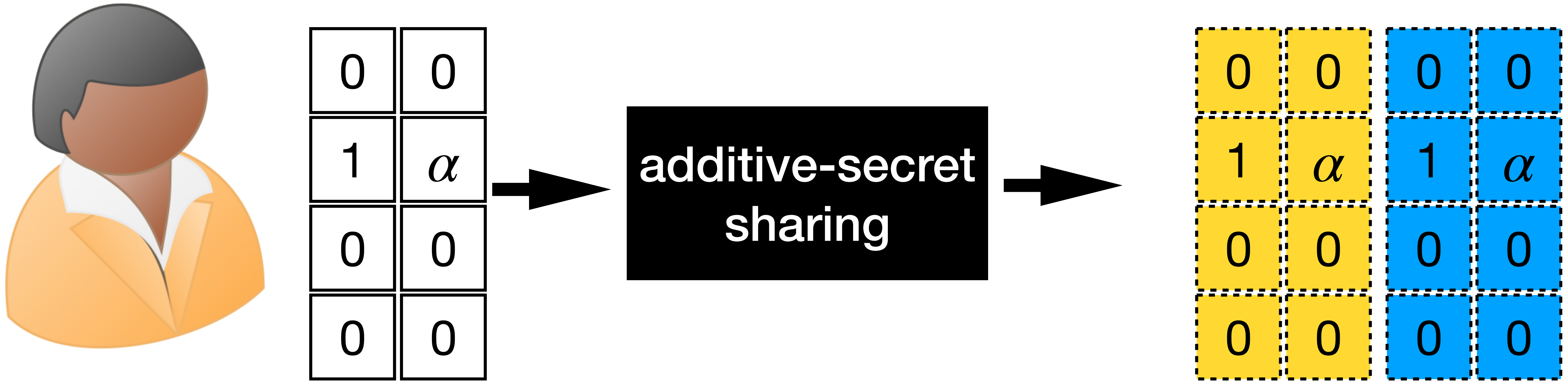
$\sqrt{N}$ 😇

$\sqrt{N}$

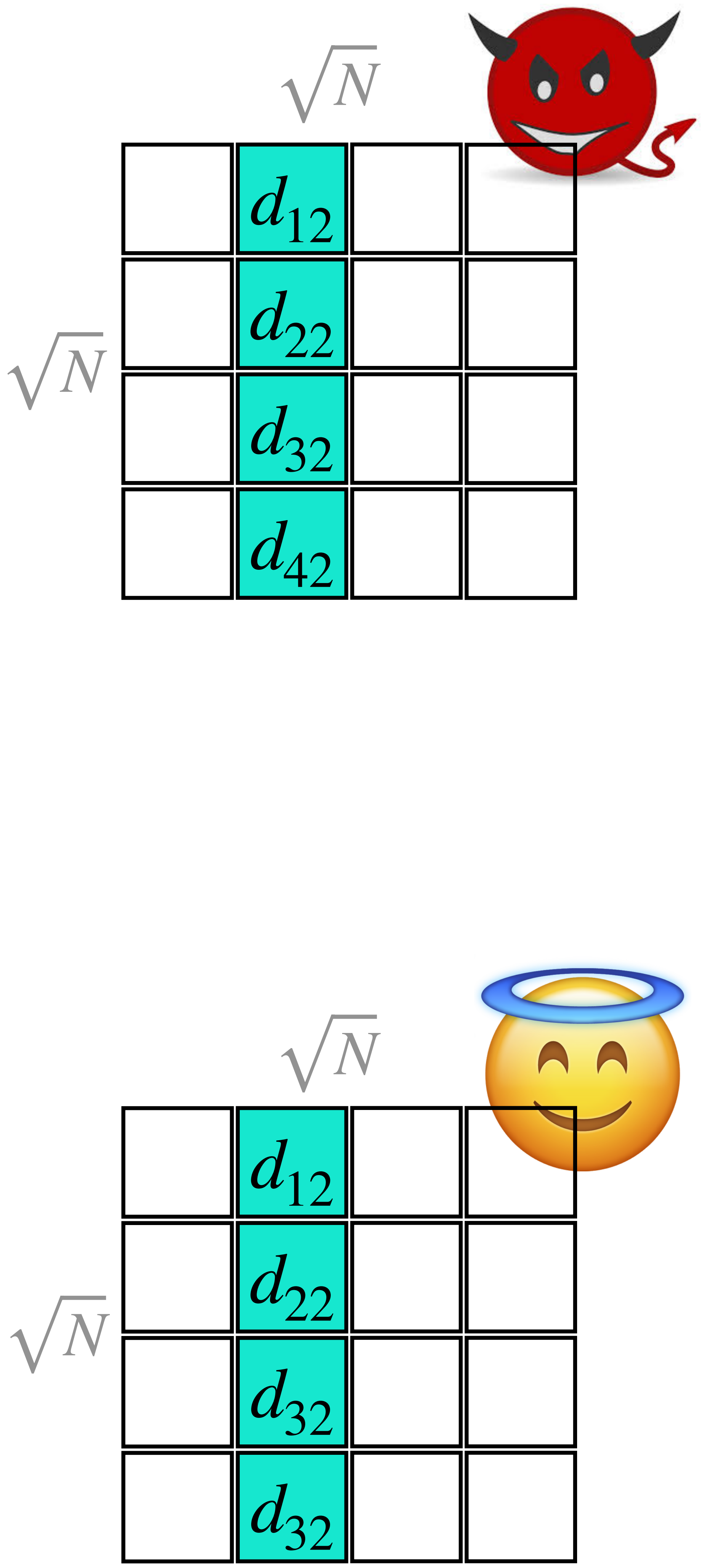
	d_{12}		
	d_{22}		
	d_{32}		
	d_{32}		

Authenticated multi-server PIR

samples random $\alpha \in_R \mathbb{F}$



$$\begin{bmatrix} 0 & 0 \\ 1 & \alpha \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \rightarrow \boxed{\text{additive-secret sharing}} \rightarrow \begin{bmatrix} 0 & 0 \\ 1 & \alpha \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 1 & \alpha \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

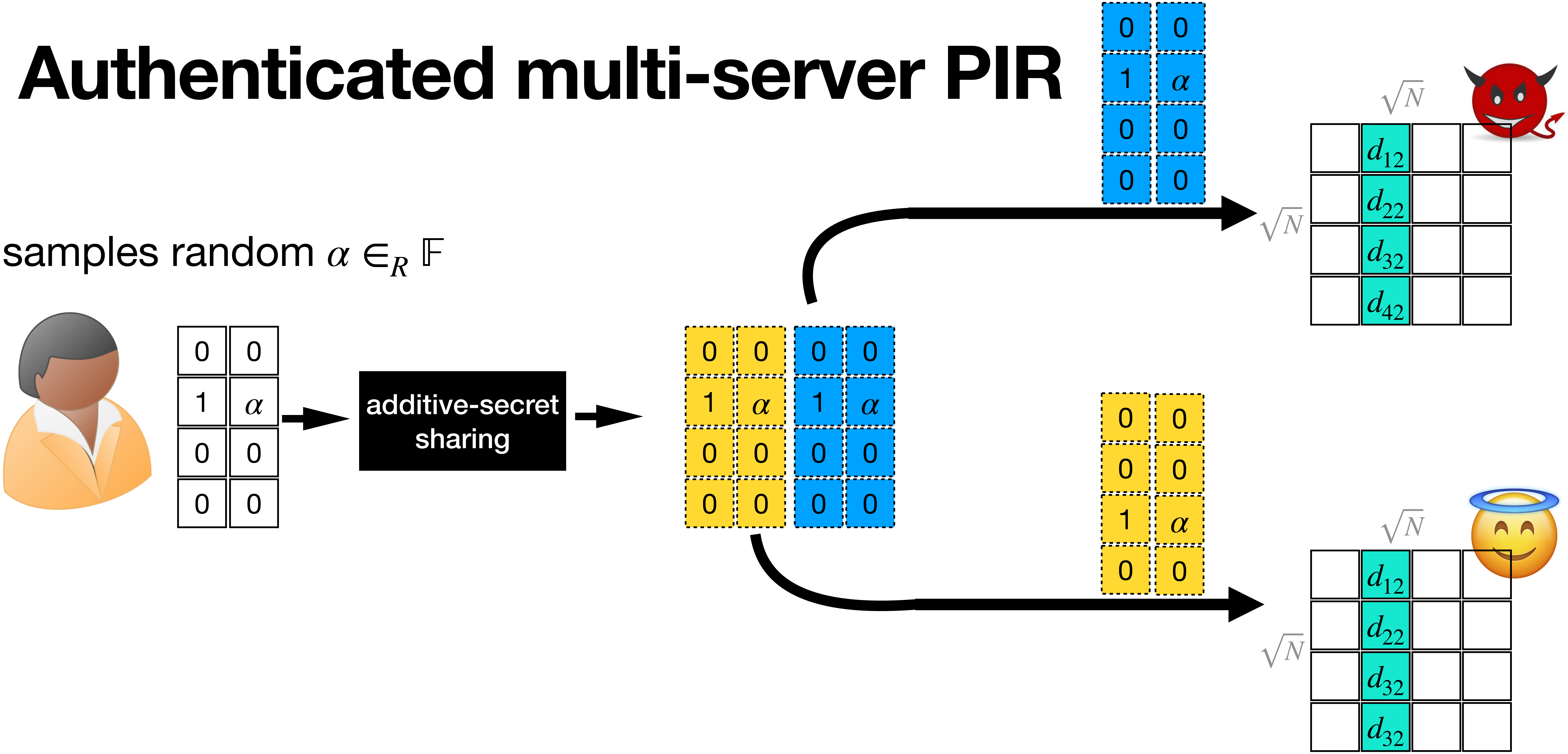


$$\begin{array}{c|cccc} & & \sqrt{N} & & \\ \hline & & d_{12} & & \\ & & d_{22} & & \\ \sqrt{N} & & d_{32} & & \\ & & d_{42} & & \end{array}$$

$$\begin{array}{c|cccc} & & \sqrt{N} & & \\ \hline & & d_{12} & & \\ & & d_{22} & & \\ \sqrt{N} & & d_{32} & & \\ & & d_{32} & & \end{array}$$

Authenticated multi-server PIR

samples random $\alpha \in_R \mathbb{F}$



$$\begin{array}{|c|c|}\hline 0 & 0 \\ \hline 1 & \alpha \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline \end{array}$$

additive-secret sharing

$$\begin{array}{|c|c|}\hline 0 & 0 \\ \hline 1 & \alpha \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline \end{array} \quad \begin{array}{|c|c|}\hline 0 & 0 \\ \hline 1 & \alpha \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline \end{array}$$

$$\begin{array}{|c|c|}\hline 0 & 0 \\ \hline 1 & \alpha \\ \hline 0 & 0 \\ \hline 0 & 0 \\ \hline \end{array}$$

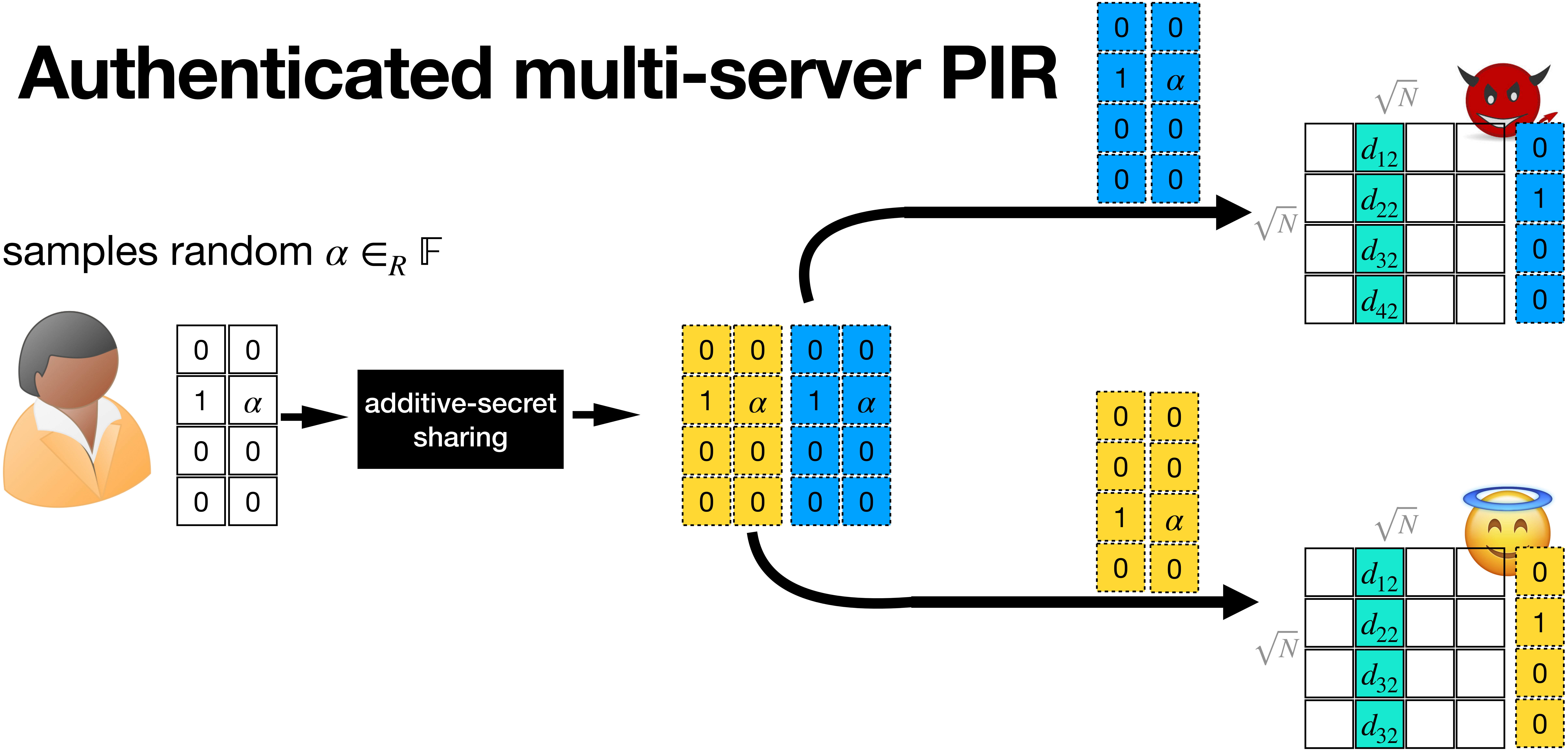
$$\begin{array}{c|c|c|c|c|} & \multicolumn{4}{c}{\sqrt{N}} \\ \cline{2-5} & & d_{12} & & \\ \cline{2-5} \sqrt{N} & & d_{22} & & \\ \cline{2-5} & & d_{32} & & \\ \cline{2-5} & & d_{42} & & \\ \cline{2-5} \end{array}$$

$$\begin{array}{|c|c|}\hline 0 & 0 \\ \hline 0 & 0 \\ \hline 1 & \alpha \\ \hline 0 & 0 \\ \hline \end{array}$$

$$\begin{array}{c|c|c|c|c|} & \multicolumn{4}{c}{\sqrt{N}} \\ \cline{2-5} & & d_{12} & & \\ \cline{2-5} \sqrt{N} & & d_{22} & & \\ \cline{2-5} & & d_{32} & & \\ \cline{2-5} & & d_{32} & & \\ \cline{2-5} \end{array}$$

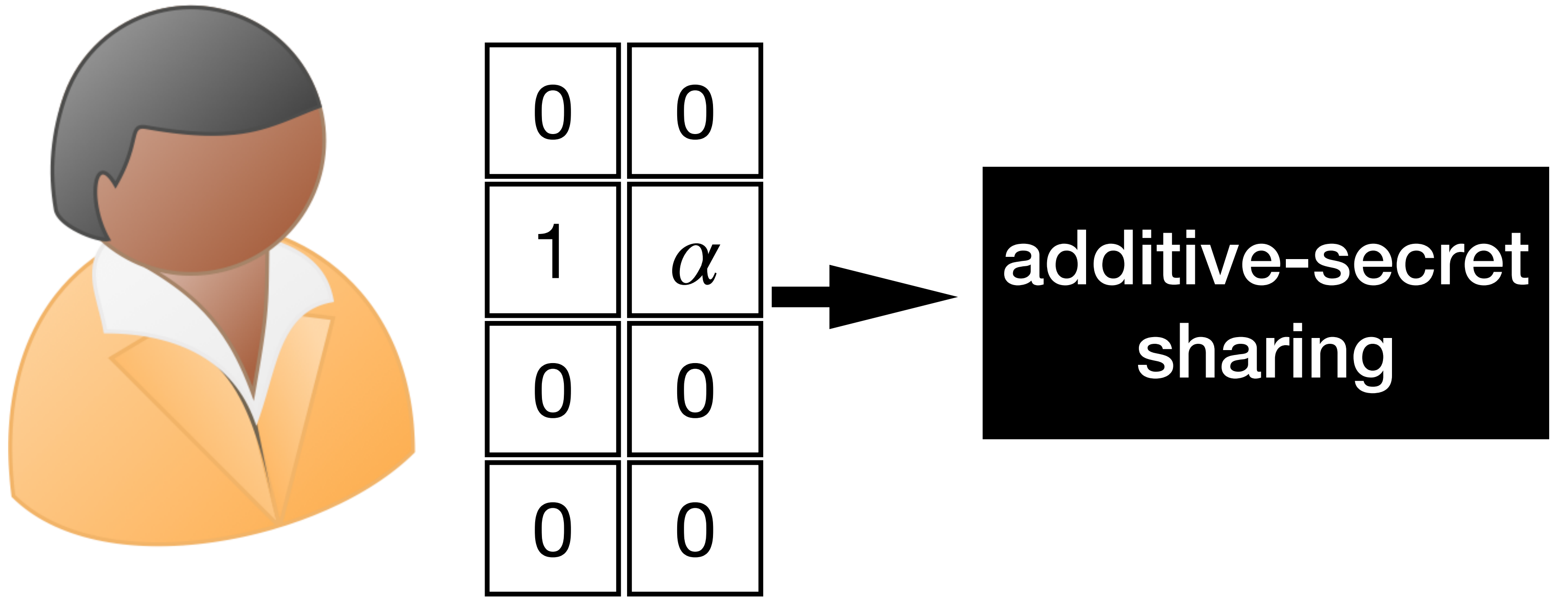
Authenticated multi-server PIR

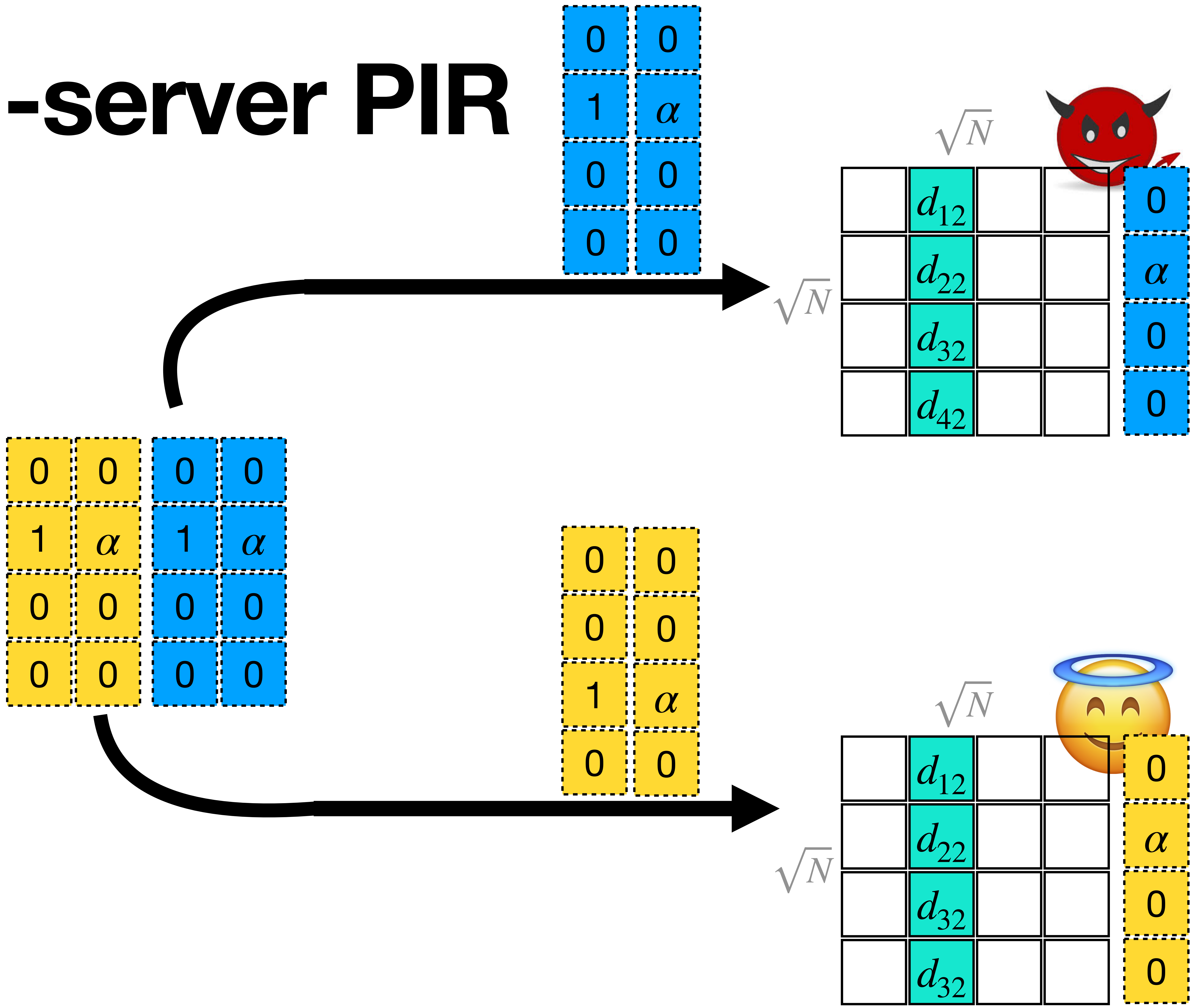
samples random $\alpha \in_R \mathbb{F}$



Authenticated multi-server PIR

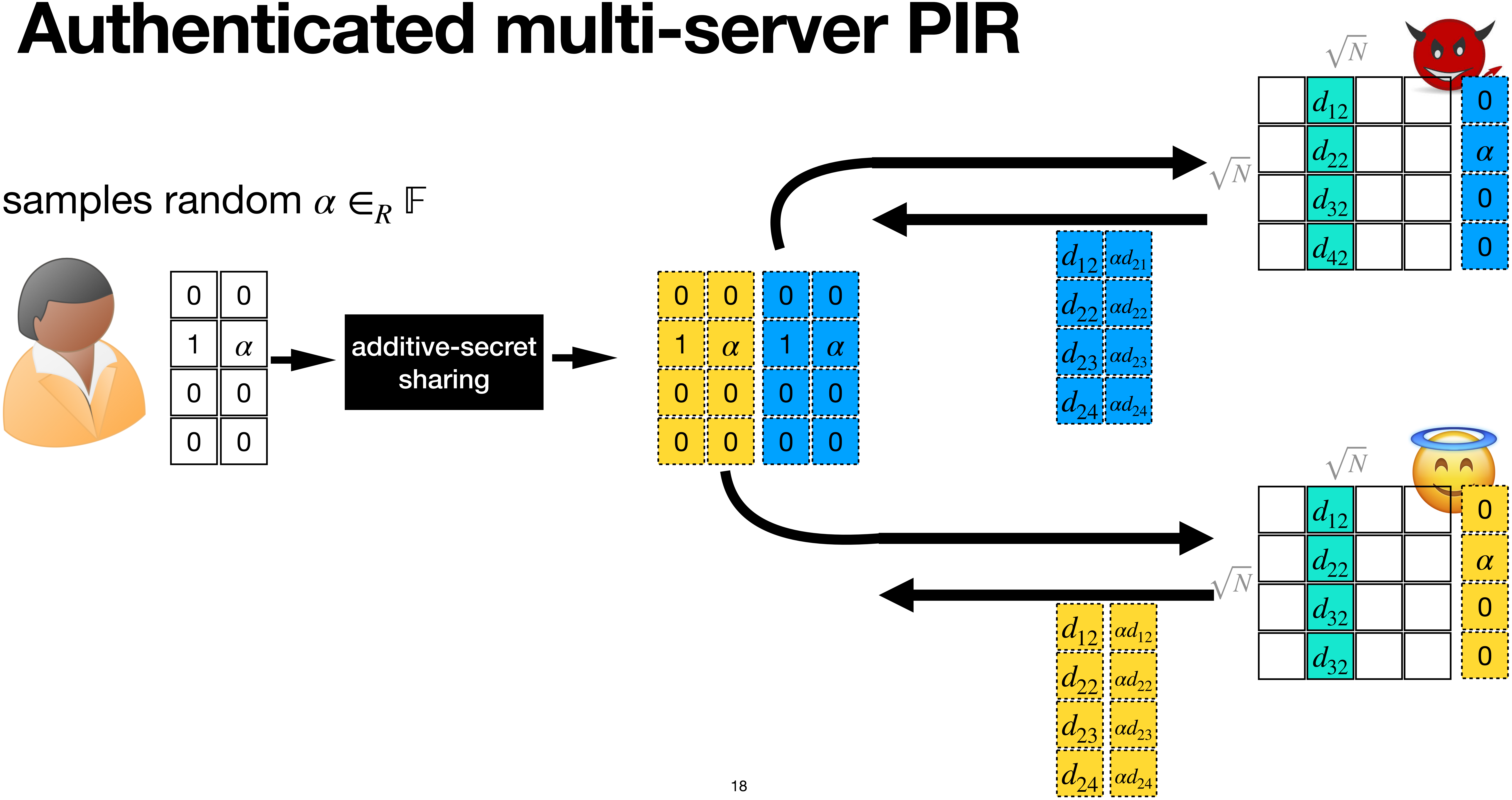
samples random $\alpha \in_R \mathbb{F}$





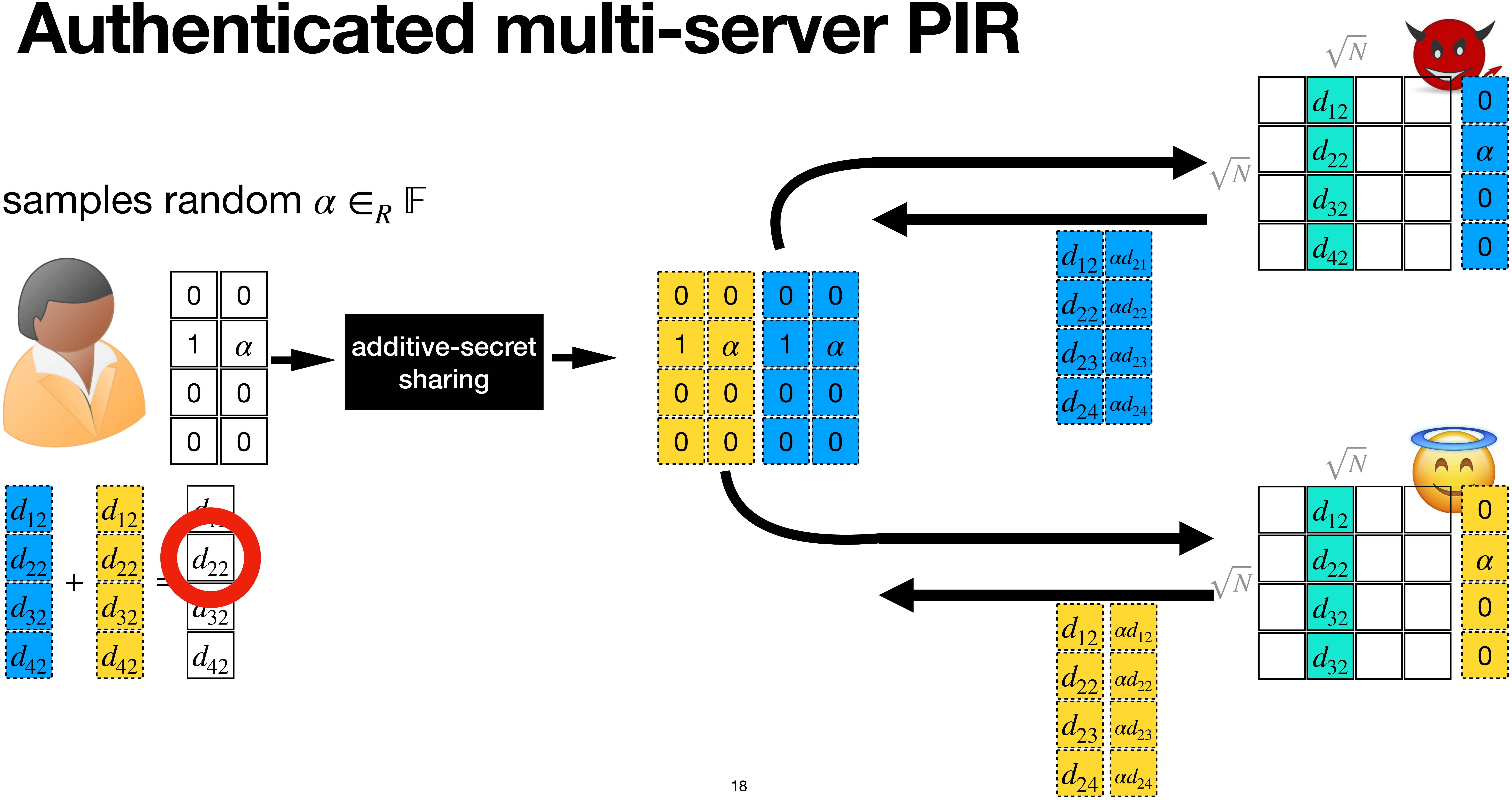
Authenticated multi-server PIR

samples random $\alpha \in_R \mathbb{F}$



Authenticated multi-server PIR

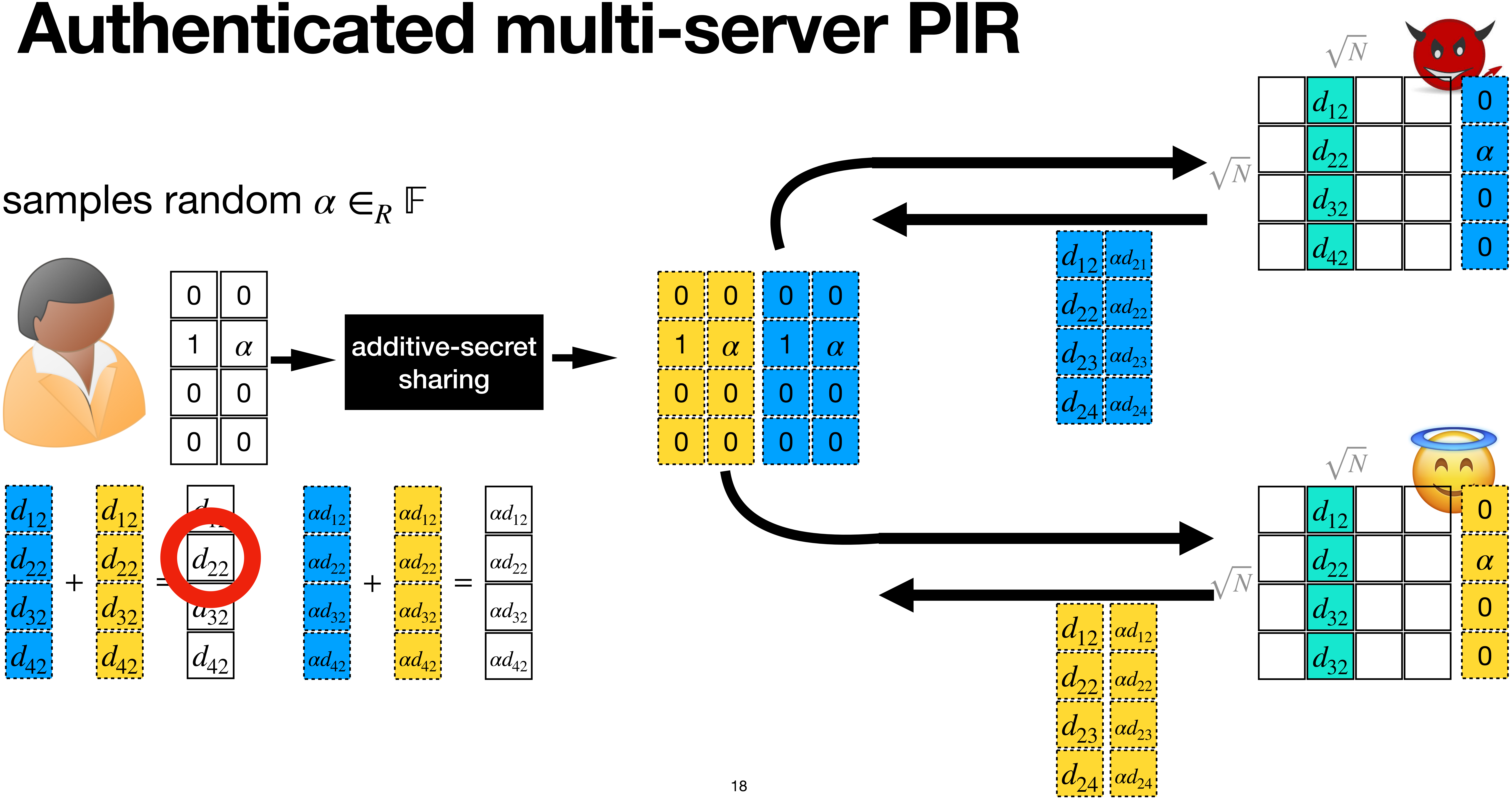
samples random $\alpha \in_R \mathbb{F}$



additive-secret sharing

Authenticated multi-server PIR

samples random $\alpha \in_R \mathbb{F}$



$\begin{bmatrix} 0 & 0 \\ 1 & \alpha \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
additive-secret sharing
$\begin{bmatrix} 0 & 0 \\ 1 & \alpha \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$ $\begin{bmatrix} 0 & 0 \\ 1 & \alpha \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
$\sqrt{N}$
$\sqrt{N}$
d_{12} d_{22} d_{32} d_{42}
0 α 0 0
d_{12} αd_{21} d_{22} αd_{22} d_{23} αd_{23} d_{24} αd_{24}
$\sqrt{N}$
$\sqrt{N}$
d_{12} d_{22} d_{32} d_{32}
0 α 0 0
d_{12} αd_{12} d_{22} αd_{22} d_{23} αd_{23} d_{24} αd_{24}
$\begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{bmatrix} + \begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{bmatrix} = \begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{bmatrix}$
$\begin{bmatrix} \alpha d_{12} \\ \alpha d_{22} \\ \alpha d_{32} \\ \alpha d_{42} \end{bmatrix} + \begin{bmatrix} \alpha d_{12} \\ \alpha d_{22} \\ \alpha d_{32} \\ \alpha d_{42} \end{bmatrix} = \begin{bmatrix} \alpha d_{12} \\ \alpha d_{22} \\ \alpha d_{32} \\ \alpha d_{42} \end{bmatrix}$

Authenticated multi-server PIR integrity

Authenticated multi-server PIR integrity

$$\text{if } \alpha \cdot \left(\begin{matrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{matrix} + \begin{matrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{matrix} \right) = \begin{matrix} \alpha d_{12} \\ \alpha d_{22} \\ \alpha d_{32} \\ \alpha d_{42} \end{matrix} + \begin{matrix} \alpha d_{12} \\ \alpha d_{22} \\ \alpha d_{32} \\ \alpha d_{42} \end{matrix}$$

Authenticated multi-server PIR integrity

$$\text{if } \alpha \cdot \left(\begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{bmatrix} + \begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{bmatrix} \right) = \begin{bmatrix} \alpha d_{12} \\ \alpha d_{22} \\ \alpha d_{32} \\ \alpha d_{42} \end{bmatrix} + \begin{bmatrix} \alpha d_{12} \\ \alpha d_{22} \\ \alpha d_{32} \\ \alpha d_{42} \end{bmatrix}$$

$$\text{return second element of } \begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{bmatrix} + \begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{32} \end{bmatrix} = \begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{bmatrix}$$

Authenticated multi-server PIR integrity

if $\alpha \cdot \left(\begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{bmatrix} + \begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{bmatrix} \right) = \begin{bmatrix} \alpha d_{12} \\ \alpha d_{22} \\ \alpha d_{32} \\ \alpha d_{42} \end{bmatrix} + \begin{bmatrix} \alpha d_{12} \\ \alpha d_{22} \\ \alpha d_{32} \\ \alpha d_{42} \end{bmatrix}$

return second element of $\begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{bmatrix} + \begin{bmatrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{32} \end{bmatrix} = \begin{bmatrix} d_{12} \\ \enclose{circle}{d_{22}} \\ d_{32} \\ d_{42} \end{bmatrix}$

else abort

Authenticated multi-server PIR integrity

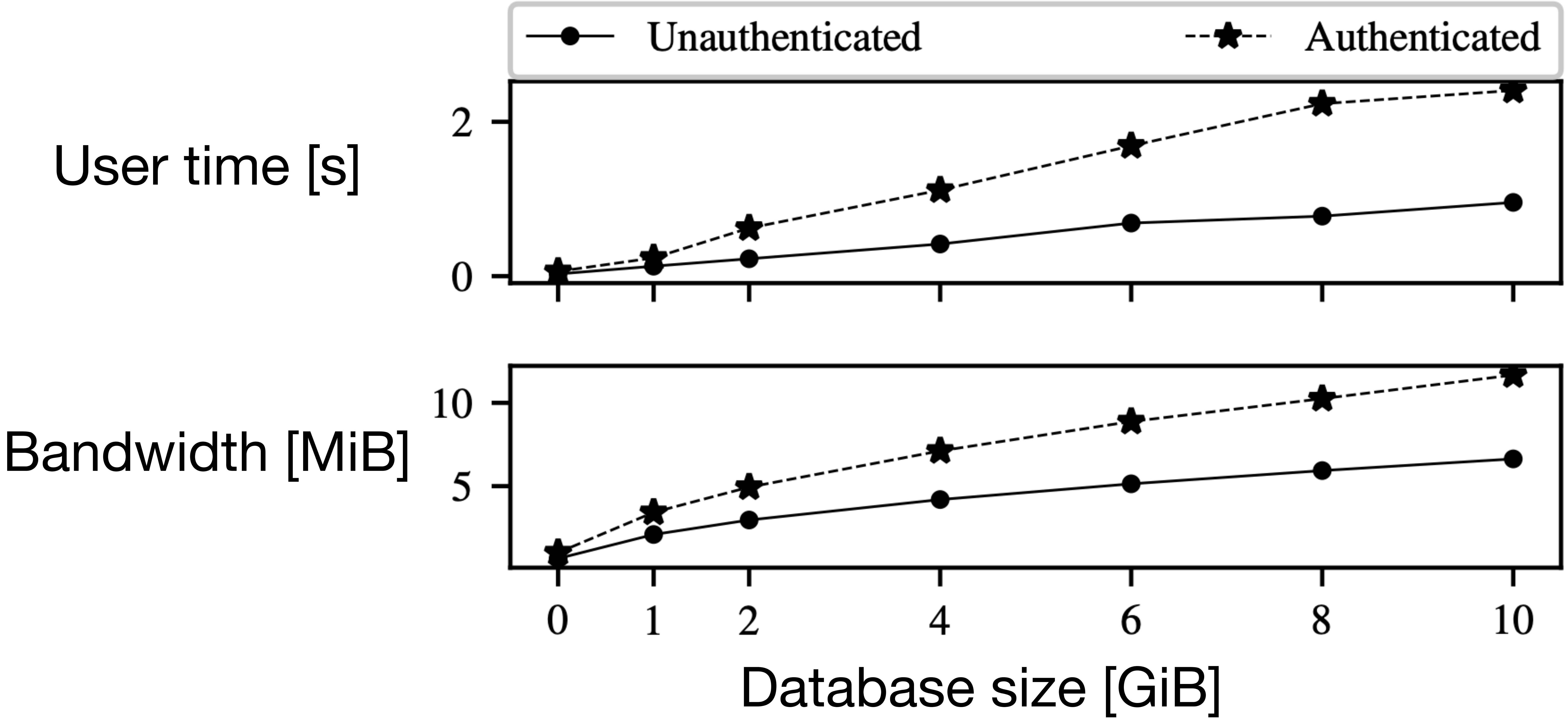
if $\alpha \cdot \left(\begin{matrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{matrix} + \begin{matrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{matrix} \right) = \begin{matrix} \alpha d_{12} \\ \alpha d_{22} \\ \alpha d_{32} \\ \alpha d_{42} \end{matrix} + \begin{matrix} \alpha d_{12} \\ \alpha d_{22} \\ \alpha d_{32} \\ \alpha d_{42} \end{matrix}$

return second element of $\begin{matrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{matrix} + \begin{matrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{32} \end{matrix} = \begin{matrix} d_{12} \\ d_{22} \\ d_{32} \\ d_{42} \end{matrix}$

else abort

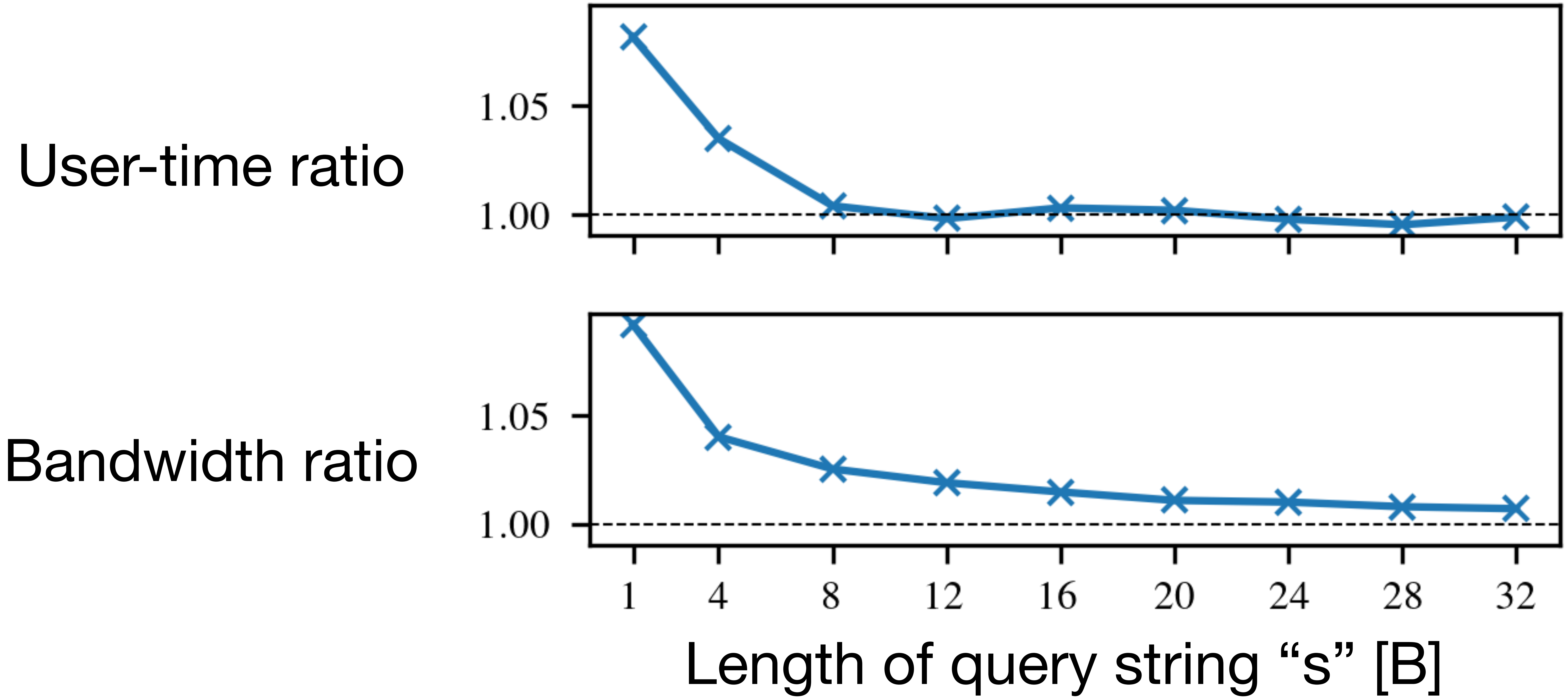
communication $O(\sqrt{N})$, see paper for $O(\log N)$ with function secret sharing [BGI16]

Evaluation: single-record queries



Cost of retrieving a 1KiB record

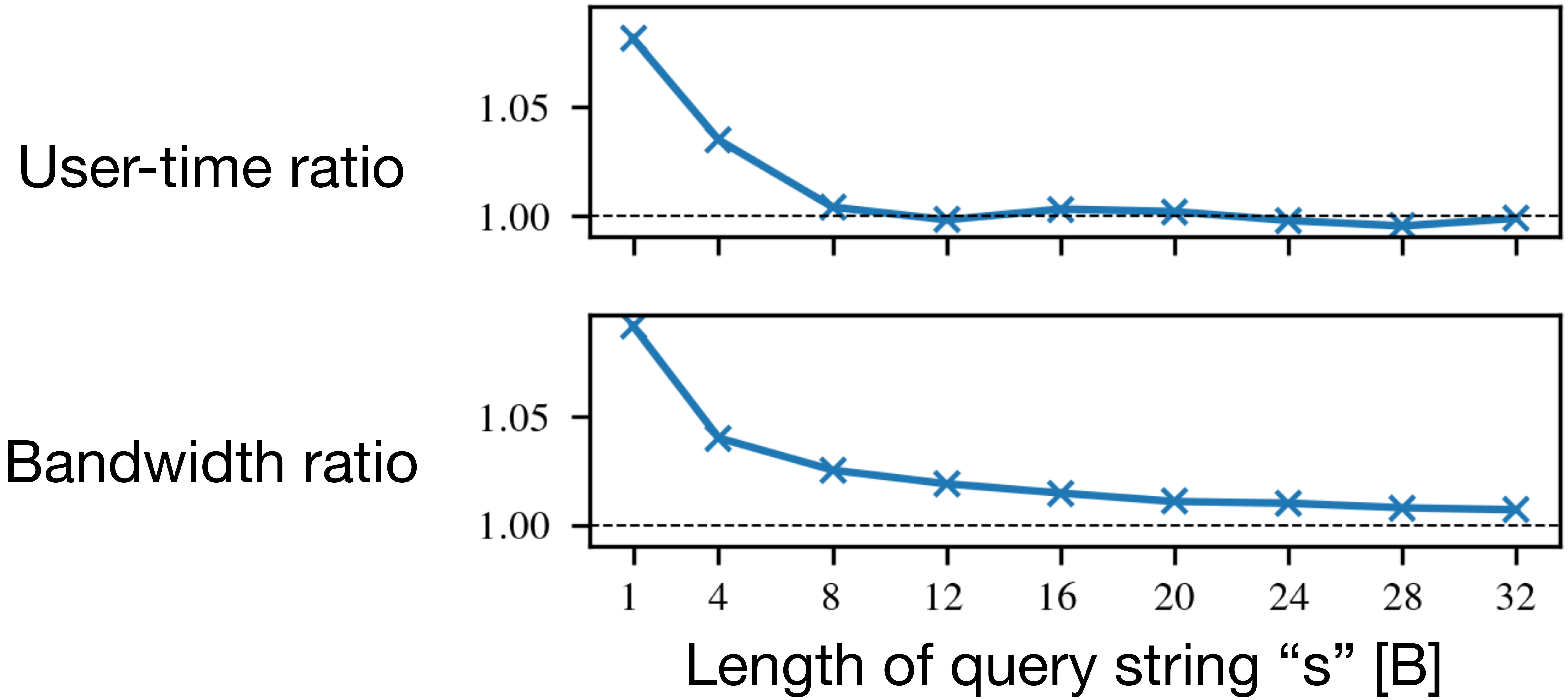
Evaluation: aggregate queries



SELECT COUNT(*) FROM keys WHERE email LIKE “%s”

Evaluation: aggregate queries

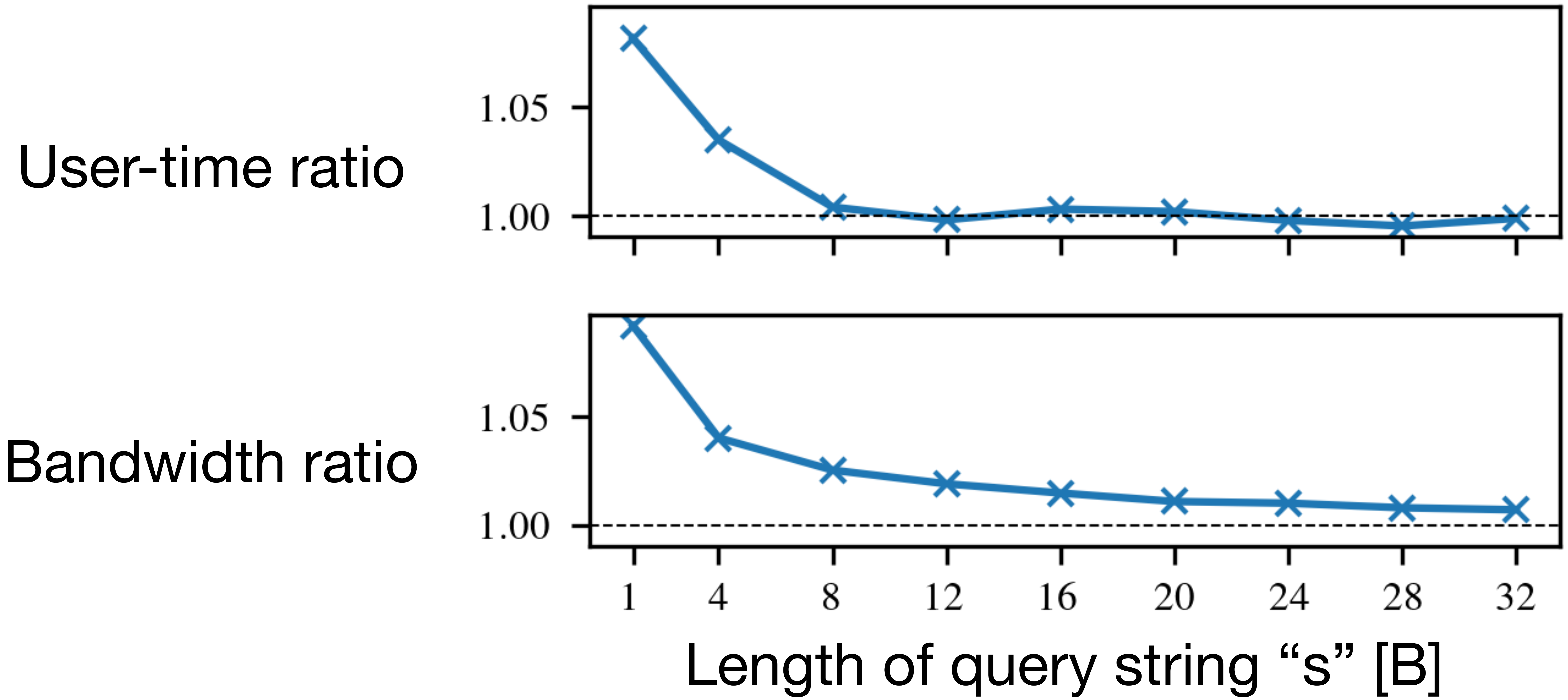
ratio of authenticated and classic unauthenticated PIR



SELECT COUNT(*) FROM keys WHERE email LIKE “%s”

Evaluation: aggregate queries

ratio of authenticated and classic unauthenticated PIR



Count emails that end with string “s”

SELECT COUNT(*) FROM keys WHERE email LIKE “%s”

DEMO TIME

Conclusion

ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

- New integrity definition for PIR schemes: either authentic record or abort.
    - In multi-server setting comes almost for free.
    - In single-server setting imposes 30-100× overhead: can we do better?
- Key directory service: PoC, but not deployed yet.
- Full paper: https://ia.cr/2023/297, code: https://github.com/dedis/apir-code.
- Keyd: https://keyd.org/.