Order-Revealing Encryption:

How to Search on Encrypted Data

David Wu

Stanford University

based on joint works with Nathan Chenette, Kevin Lewi, and Stephen A. Weis

Searching on Encrypted Data

ars TECHNICA BIZ & IT TECH SCIENCE POLICY CARS GAMING & CULTURE FORUMS

EVENT VERIZON —

Yahoo admits it’s been hacked again, and 1 billion accounts were exposed

That's a billion with a b—and is separate from the breach "cleared" in September.

SEAN GALLAGHER - 12/14/2016, 3:26 PM

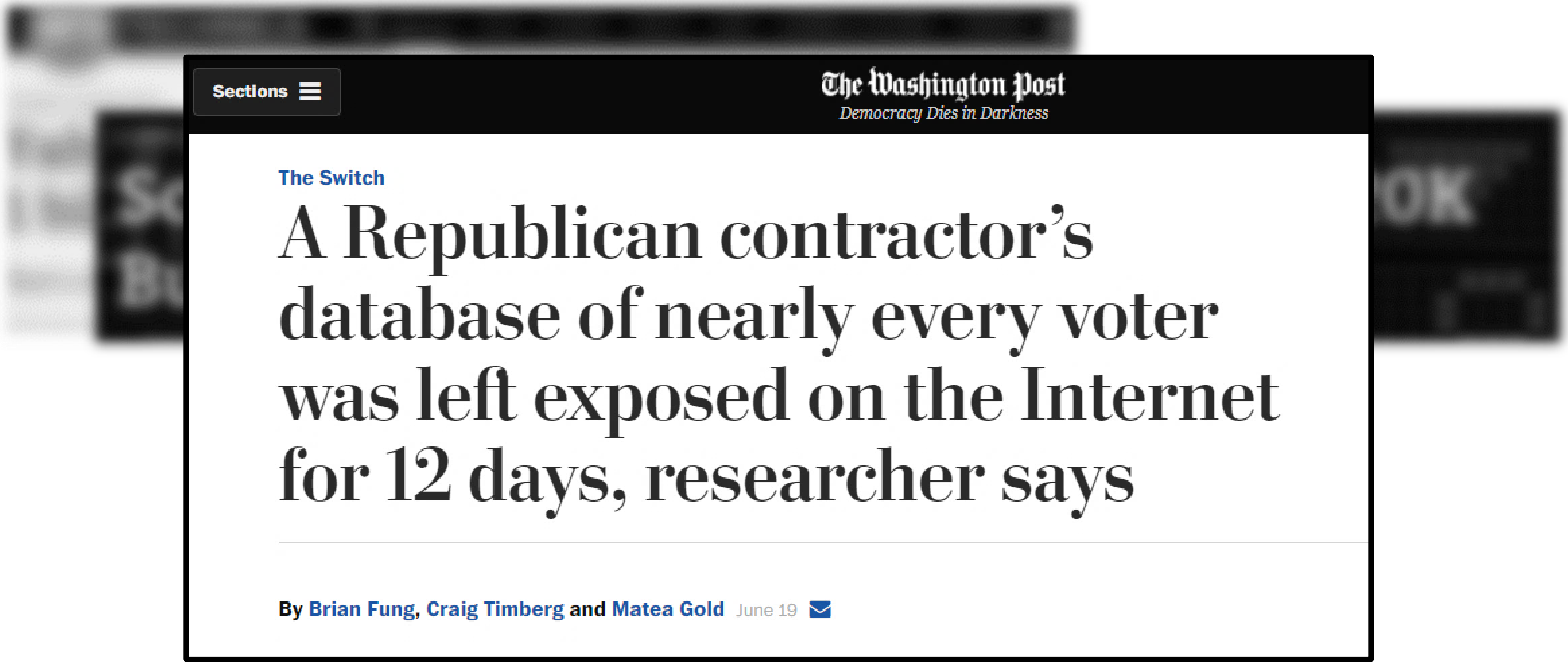
The information accessed from potentially exposed accounts “may have included names, email addresses, telephone numbers, dates of birth, hashed passwords (using MD5) and, in some cases, encrypted or unencrypted security questions and answers…”

Searching on Encrypted Data

The database was discovered by MacKeeper researcher Chris Vickery on March 31, in the course of searching for random phrases on the domain s3.amazonaws.com.

“It's as bad as I expected,” he tweeted. “Bank-related. Plaintext passwords. Big name company. I've reached out to them.”

Searching on Encrypted Data



Sections

The Washington Post
Democracy Dies in Darkness

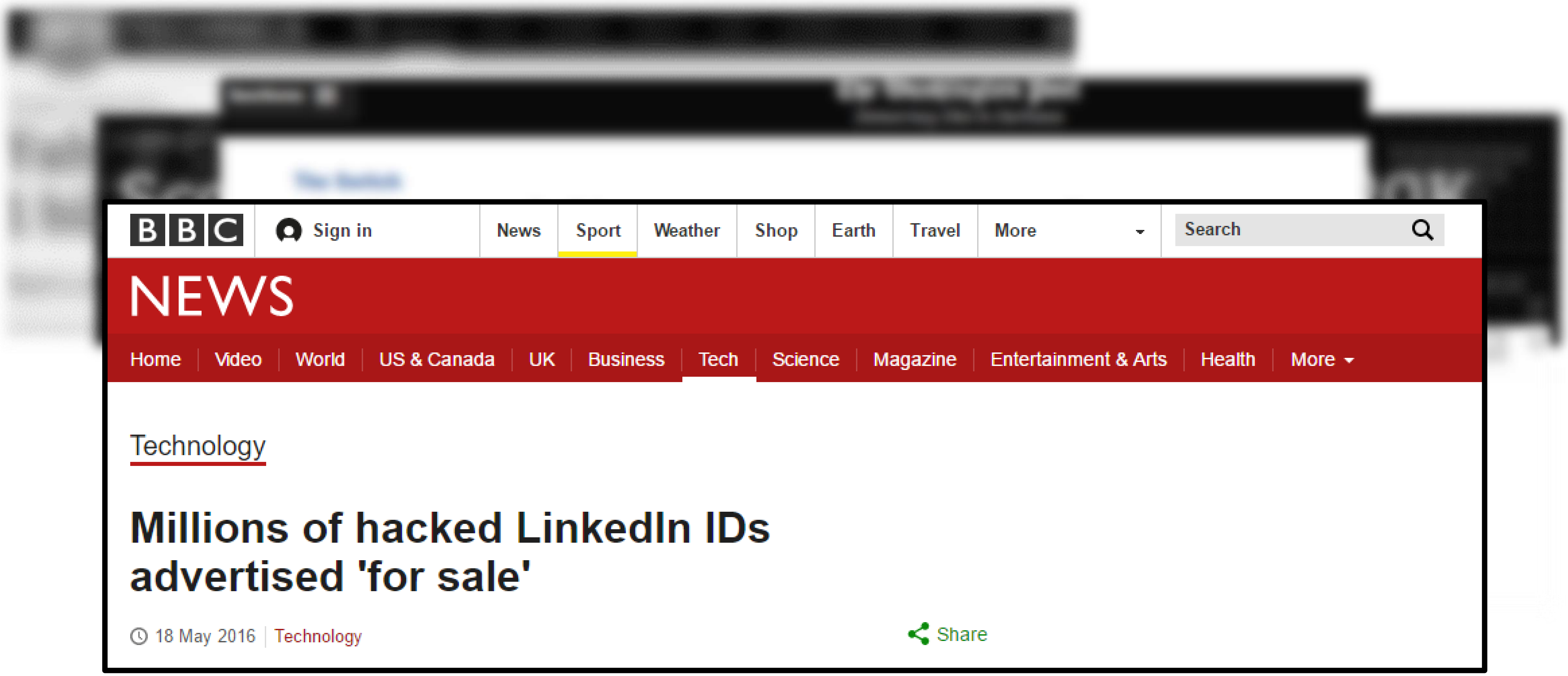
The Switch

A Republican contractor's database of nearly every voter was left exposed on the Internet for 12 days, researcher says

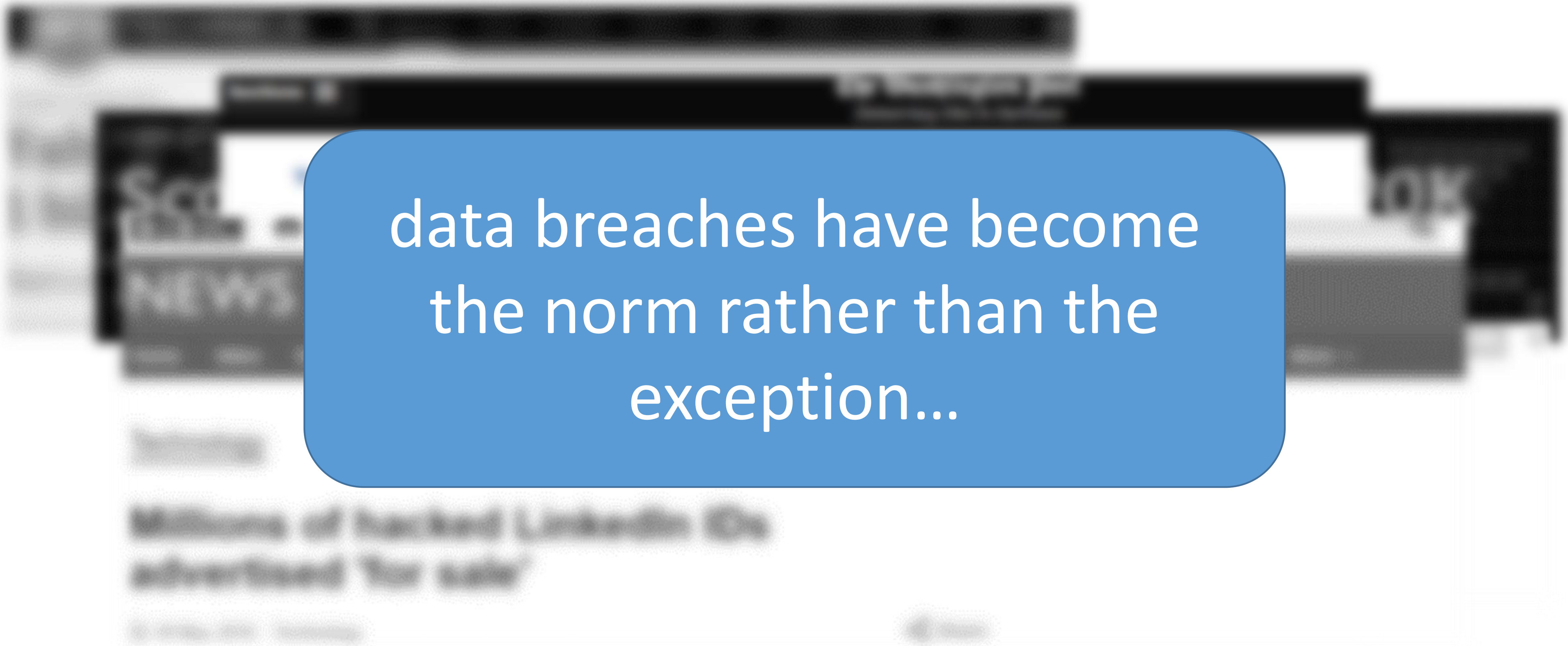
By Brian Fung, Craig Timberg and Matea Gold June 19

Searching on Encrypted Data

Searching on Encrypted Data

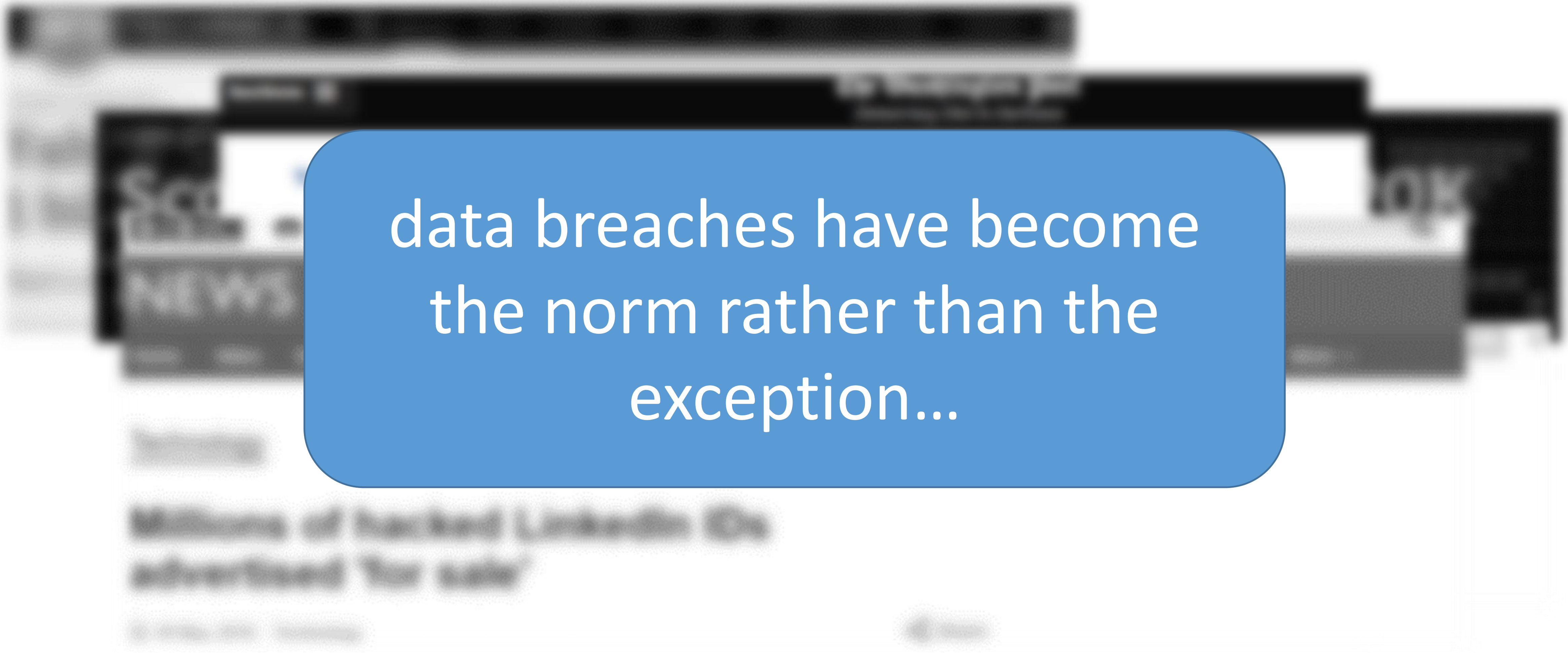


Searching on Encrypted Data



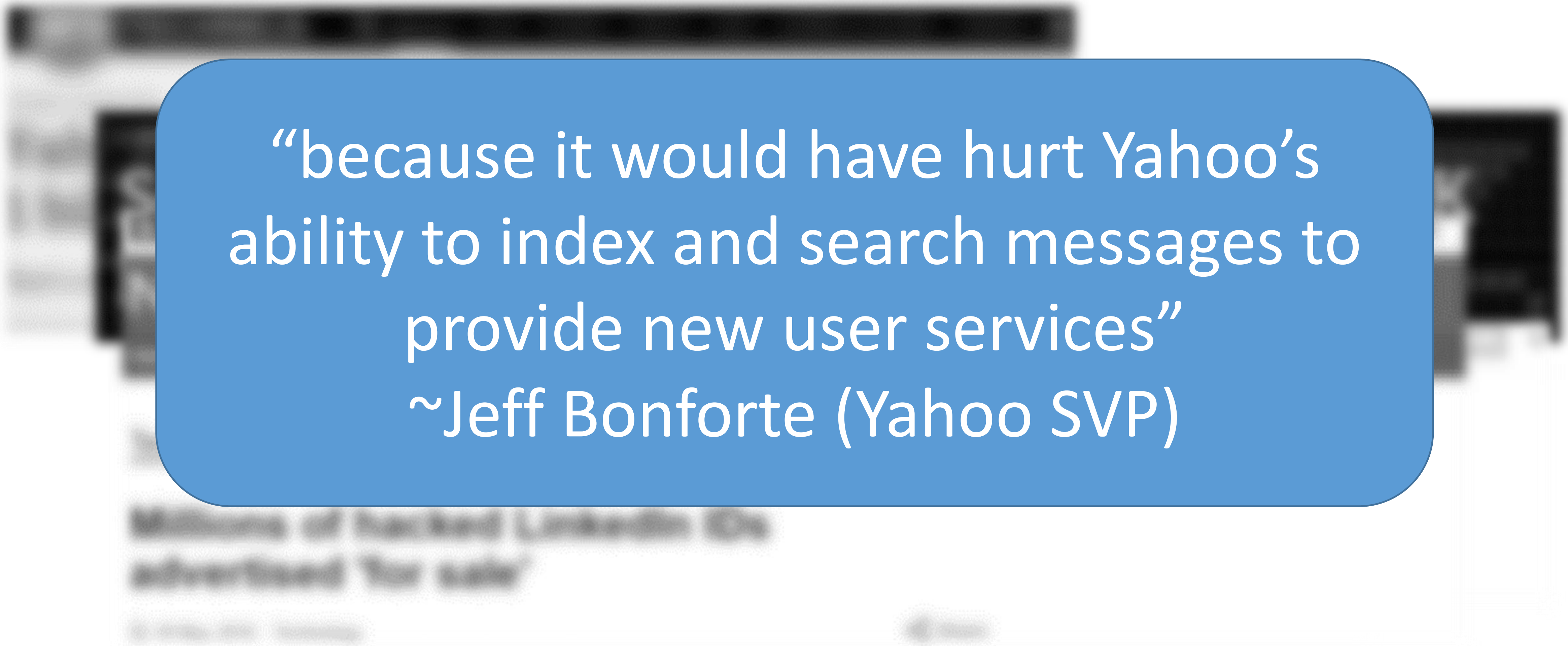
data breaches have become the norm rather than the exception…

Why Not Encrypt?



data breaches have become the norm rather than the exception…

Why Not Encrypt?

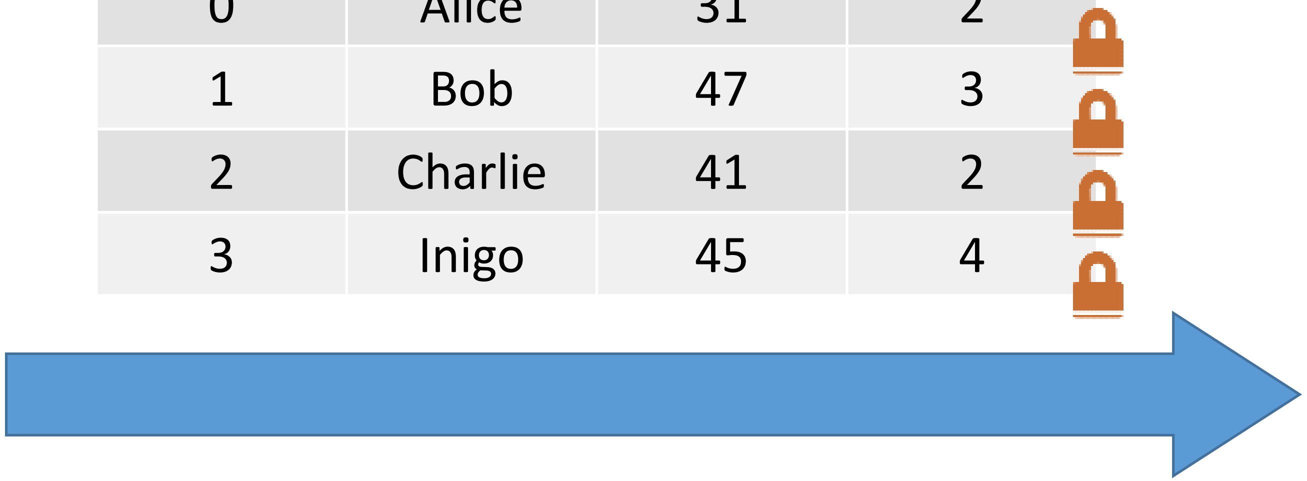


"because it would have hurt Yahoo's ability to index and search messages to provide new user services"
~Jeff Bonforte (Yahoo SVP)

Searching on Encrypted Data

database

ID	Name	Age	Diagnosis
0	Alice	31	2
1	Bob	47	3
2	Charlie	41	2
3	Inigo	45	4



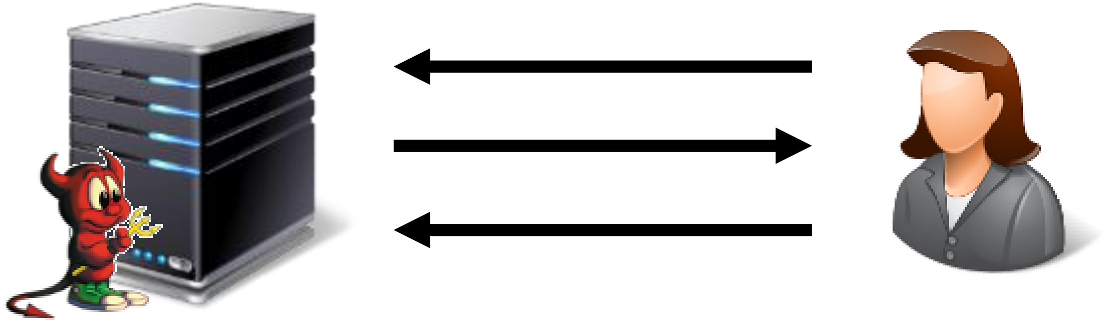
client

server

client holds a secret key (needed to encrypt + query the server)

server stores encrypted database

Security for Encrypted Search



adversary sees encrypted database + queries and can interact with the database

active adversary

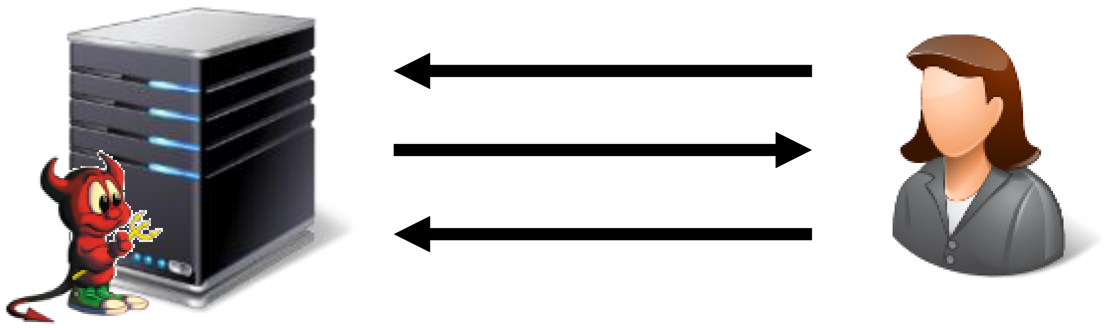
online attacks (e.g., active corruption)

offline attacks (e.g., passive snapshots)

adversary only sees contents of encrypted database

snapshot adversary

Security for Encrypted Search



adversary sees encrypted database + queries and can interact with the database

online attacks (e.g., active corruption)

offline attacks (e.g., passive snapshots)

adversary only sees contents of encrypted database

typical database breach: contents of database are stolen and dumped onto the web

Order-Revealing Encryption [BLRSZZ'15]

secret-key encryption scheme

Which is greater: the value encrypted by ct_1 or the value encrypted by ct_2?

sk

client

$$\mathrm{ct}_1 = \mathrm{Enc}(\mathrm{sk}, 123)$$
$$\mathrm{ct}_2 = \mathrm{Enc}(\mathrm{sk}, 512)$$
$$\mathrm{ct}_3 = \mathrm{Enc}(\mathrm{sk}, 273)$$

server

(legacy-friendly) range queries on encrypted data

Order-Revealing Encryption [BLRSZZ'15]

given any two ciphertexts

$\mathrm{ct}_1 = \mathrm{Enc}(\mathrm{sk}, x)$

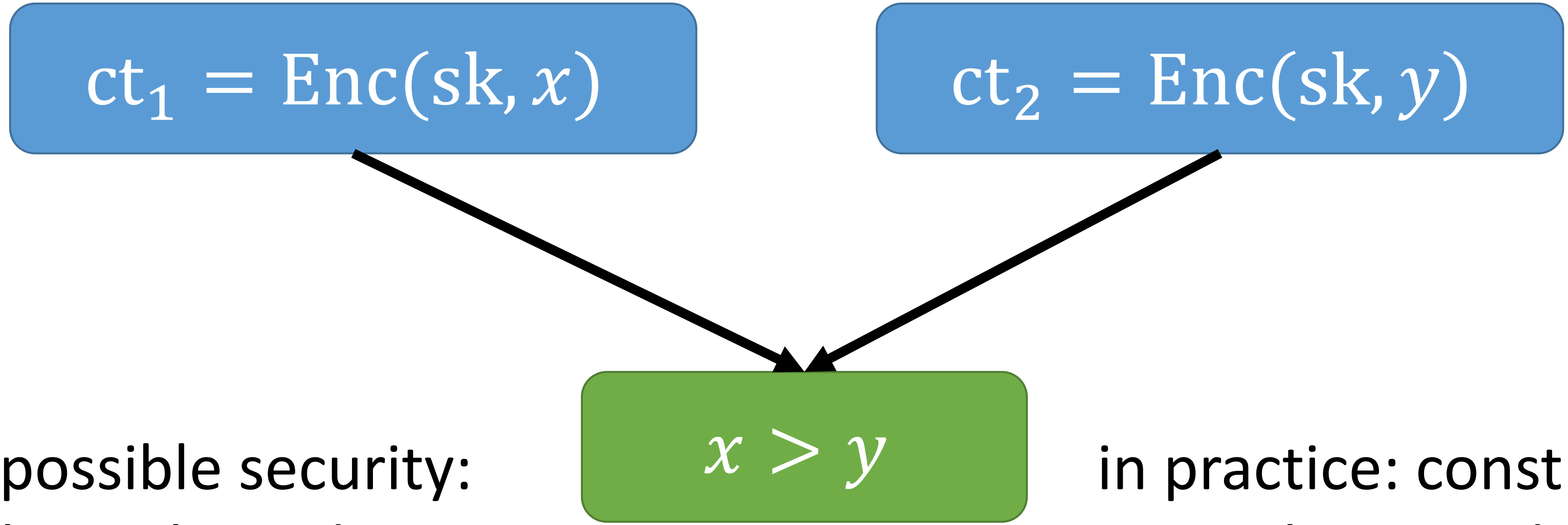
$\mathrm{ct}_2 = \mathrm{Enc}(\mathrm{sk}, y)$

$x > y$

there is a <u>public</u> function for performing comparisons

Order-Revealing Encryption [BLRSZZ'15]

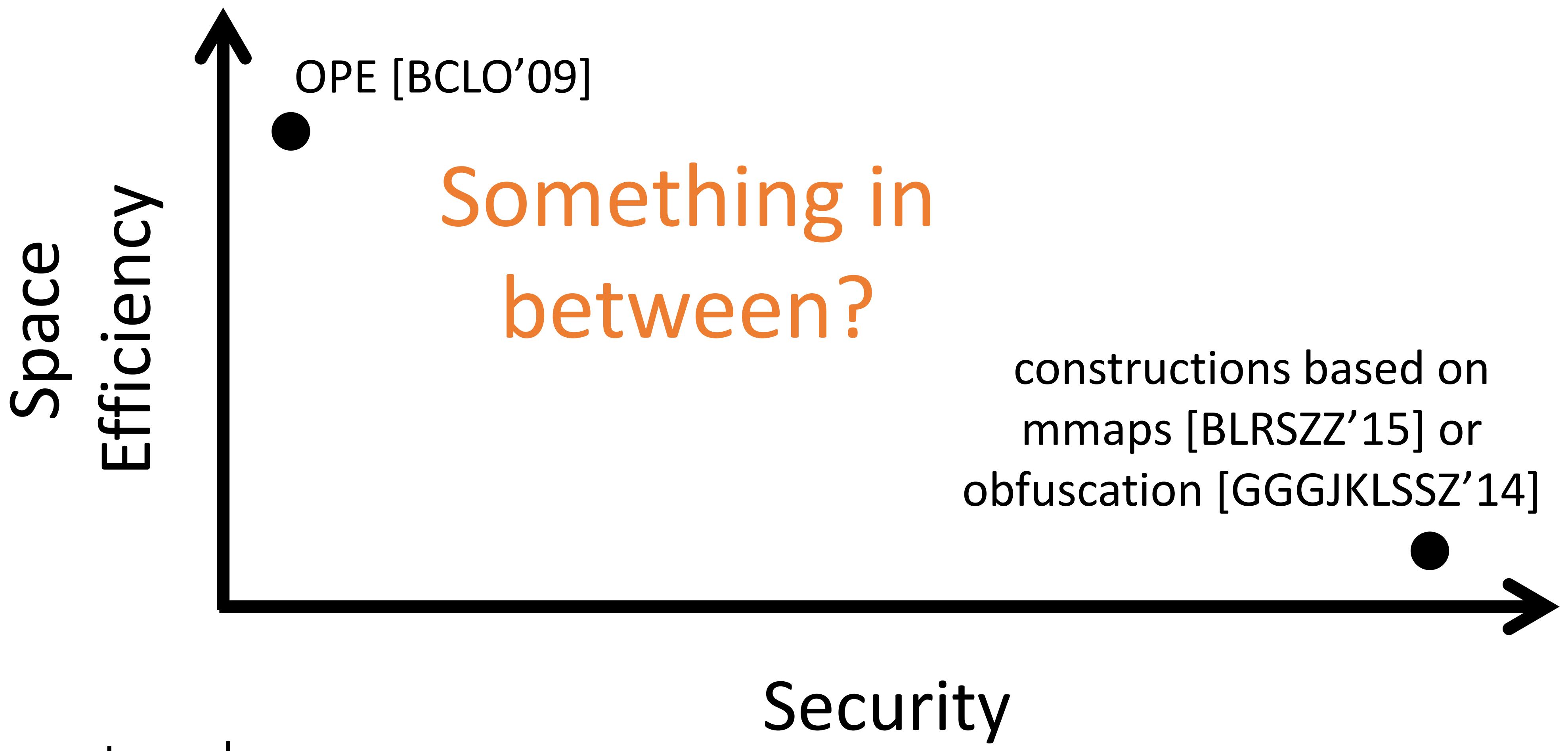
given any two ciphertexts



best-possible security: reveal just the ordering and nothing more

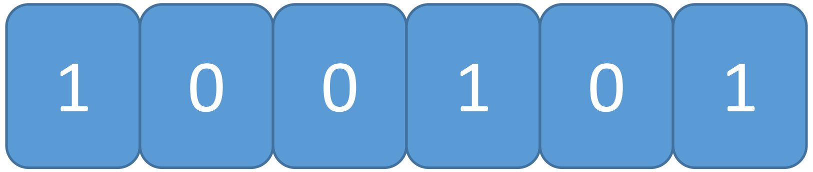
in practice: constructions reveal some additional information

Existing Approaches



Not drawn to scale

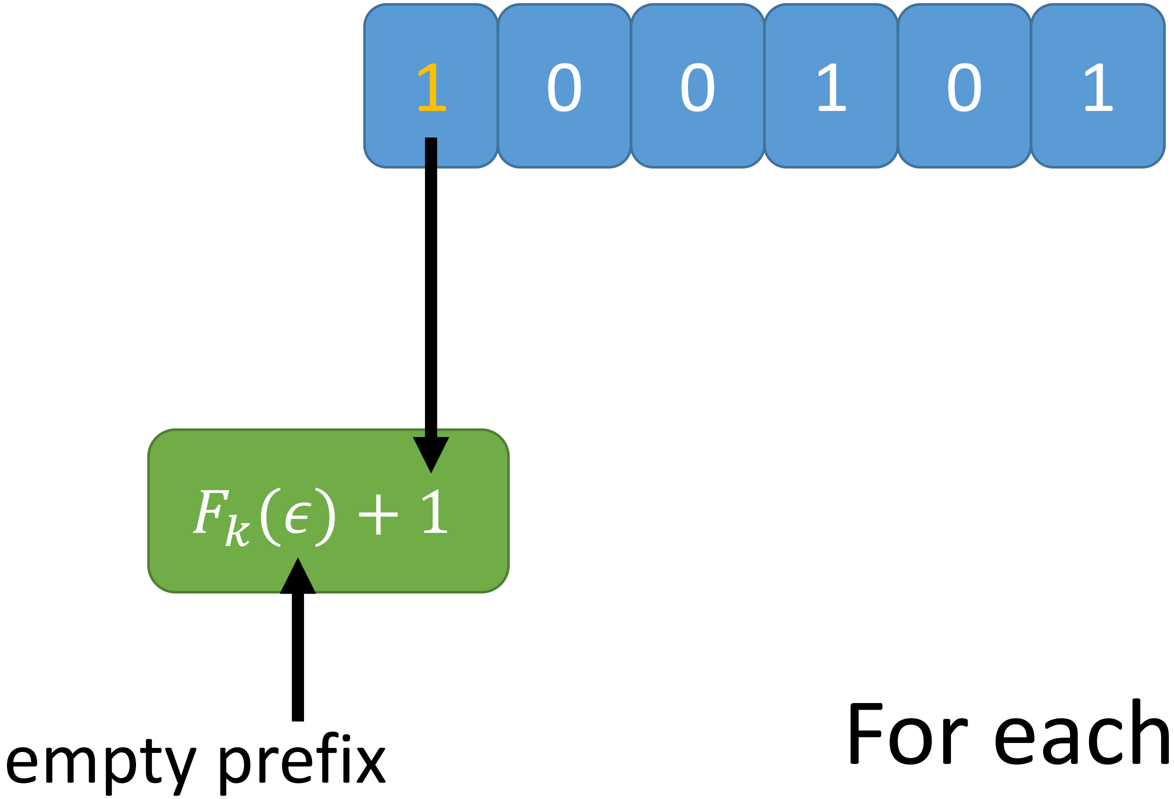
A Simple ORE Construction [CLWW'16]



For each index i, apply a PRF (e.g., AES) to the first $i - 1$ bits, then add $b_i \pmod 3$

$$F: \mathcal{K} \times \{0,1\}^* \to \{0,1,2\}$$

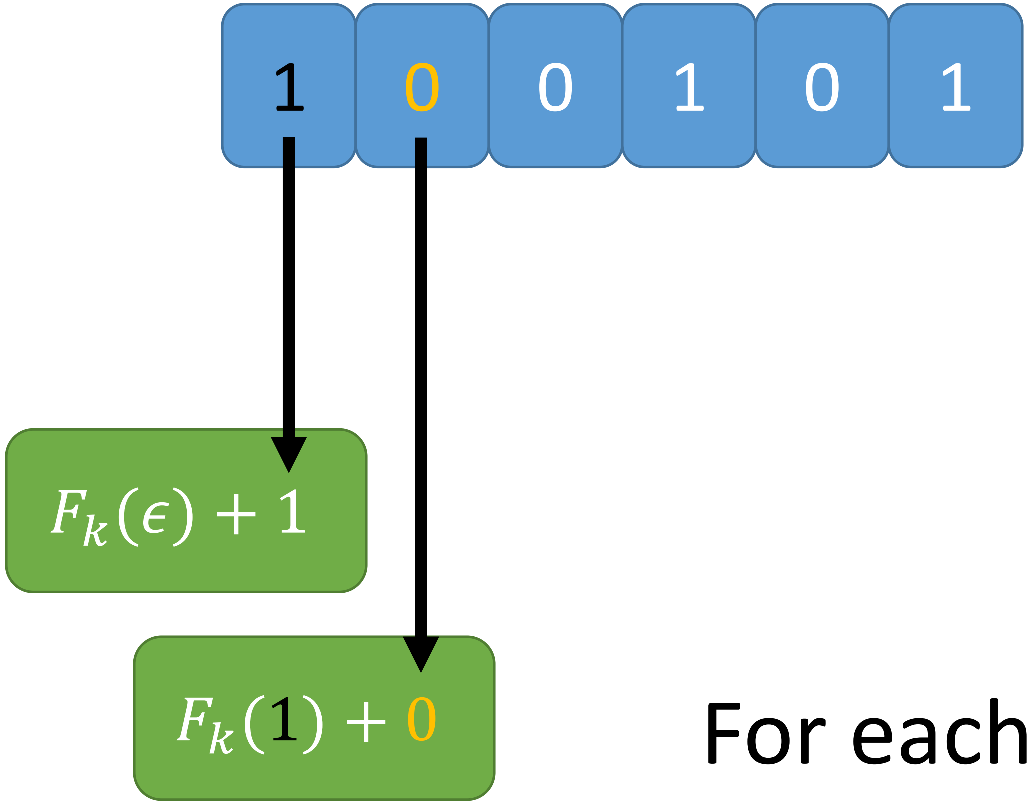
A Simple ORE Construction [CLWW'16]



$F: \mathcal{K} \times \{0,1\}^* \to \{0,1,2\}$

For each index i, apply a PRF (e.g., AES) to the first $i - 1$ bits, then add $b_i \pmod 3$

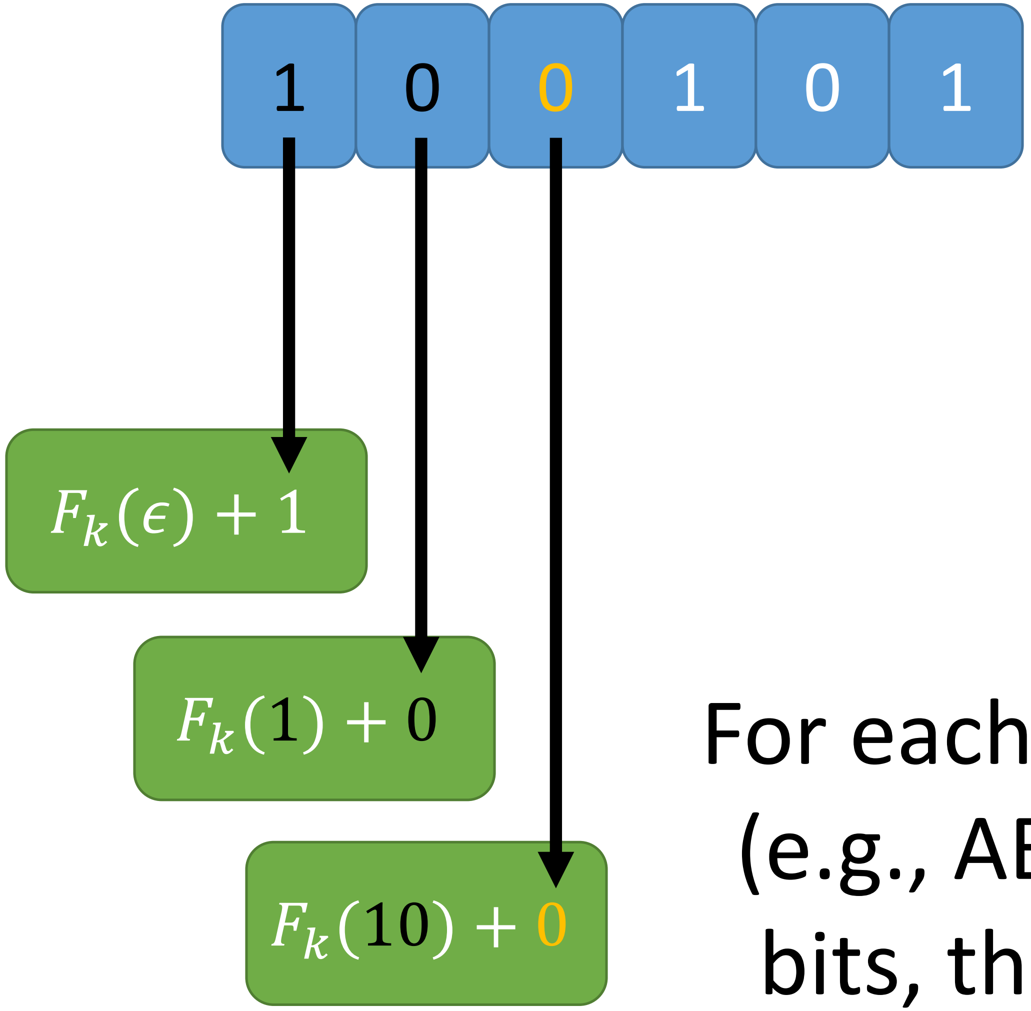
A Simple ORE Construction [CLWW'16]



For each index i, apply a PRF (e.g., AES) to the first $i - 1$ bits, then add $b_i \pmod 3$

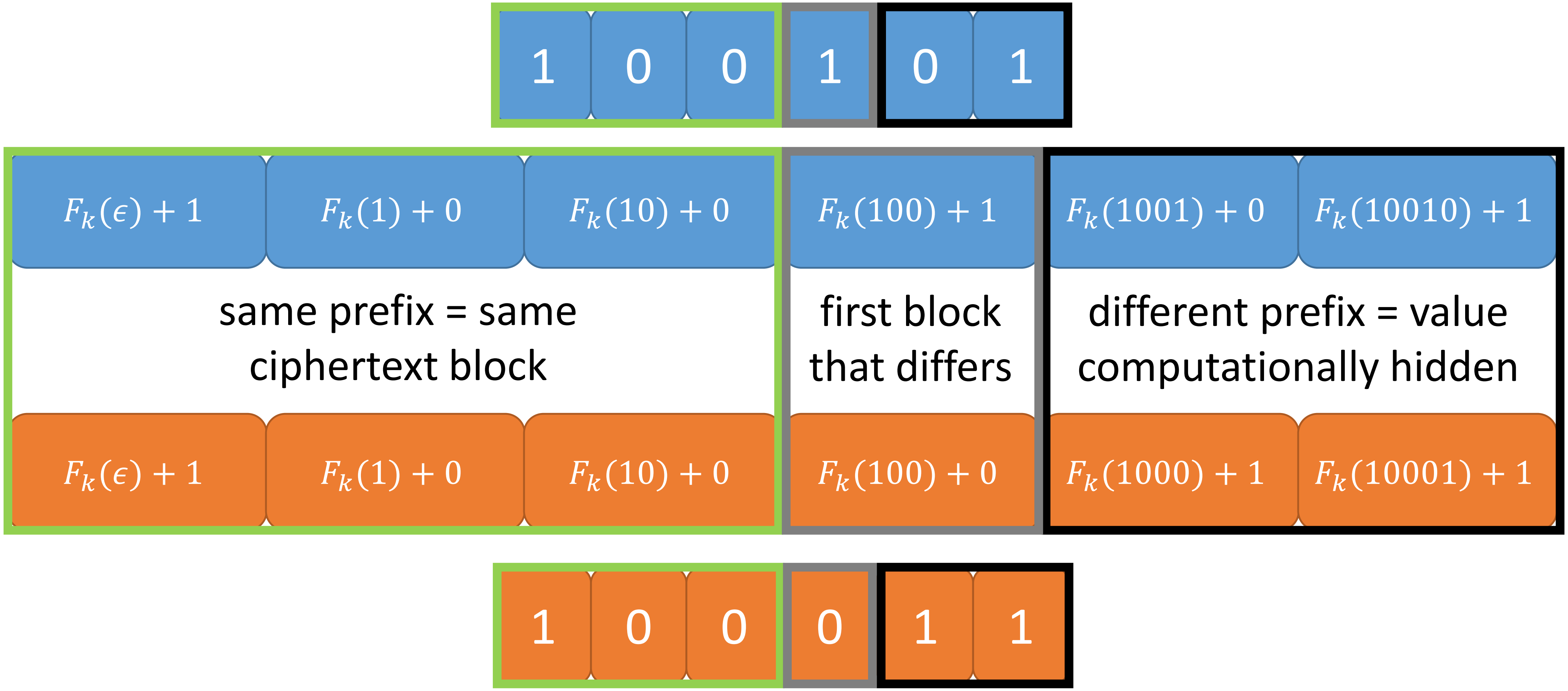
$F: \mathcal{K} \times \{0,1\}^* \to \{0,1,2\}$

A Simple ORE Construction [CLWW'16]

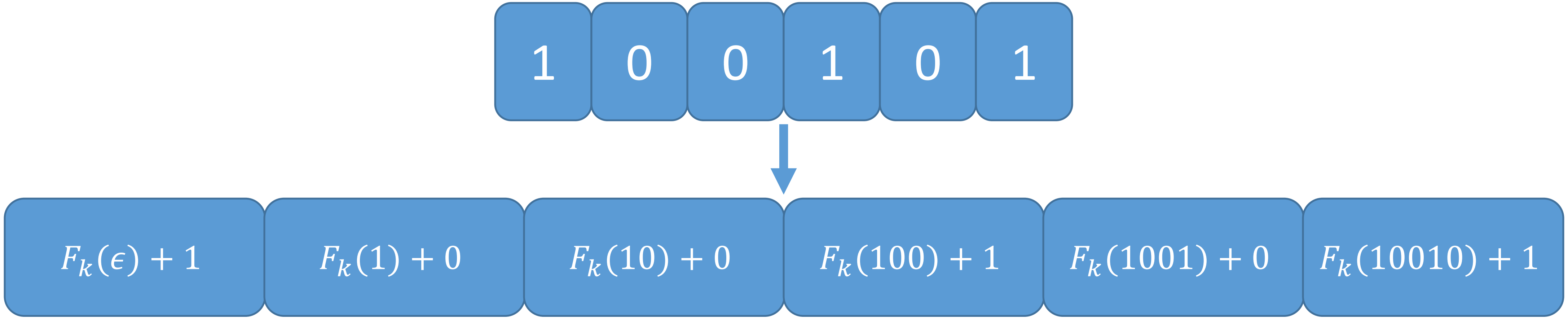


For each index i, apply a PRF (e.g., AES) to the first $i - 1$ bits, then add $b_i \pmod 3$

A Simple ORE Construction [CLWW'16]



Efficiency

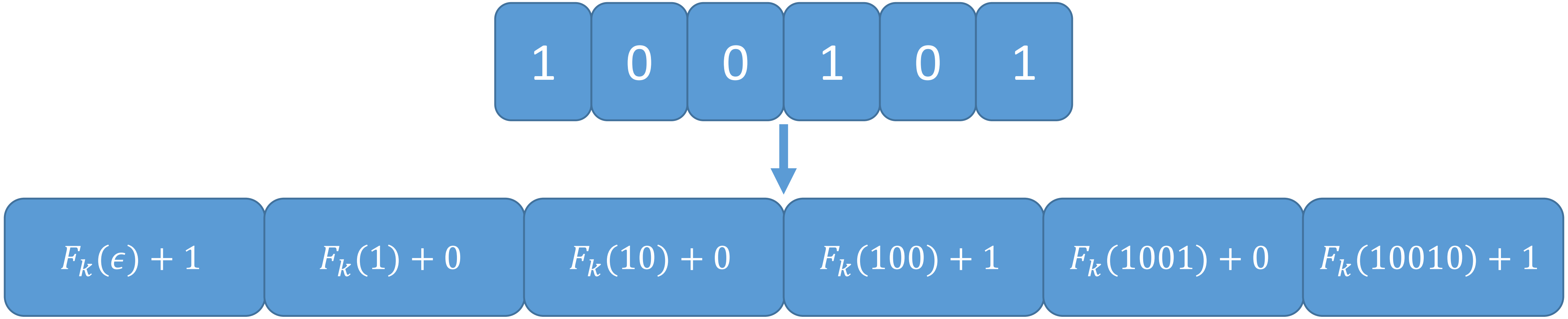


Each ciphertext block is element in $\{0,1,2\}$

For n-bit messages, can obtain ciphertexts of length $\approx 1.6n$

Encryption only requires PRF evaluations while decryption just requires bitwise comparisons

Security



Security follows directly from security of the PRF

Construction reveals the first bit on which two message differ (in addition to the ordering)

Inference Attacks [NKW'15, DDC'16, GSBNR'17]

ID	Name	Age	Diagnosis
wpjOos	2wzXW8	SqX9l9	KqLUXE
XdXdg8	y9GFpS	gwilE3	MJ23b7
P6vKhW	EgN0Jn	S0pRJe	aTaeJk
orJRe6	KQWy9U	tPWF3M	4FBEO0

encrypted database

+

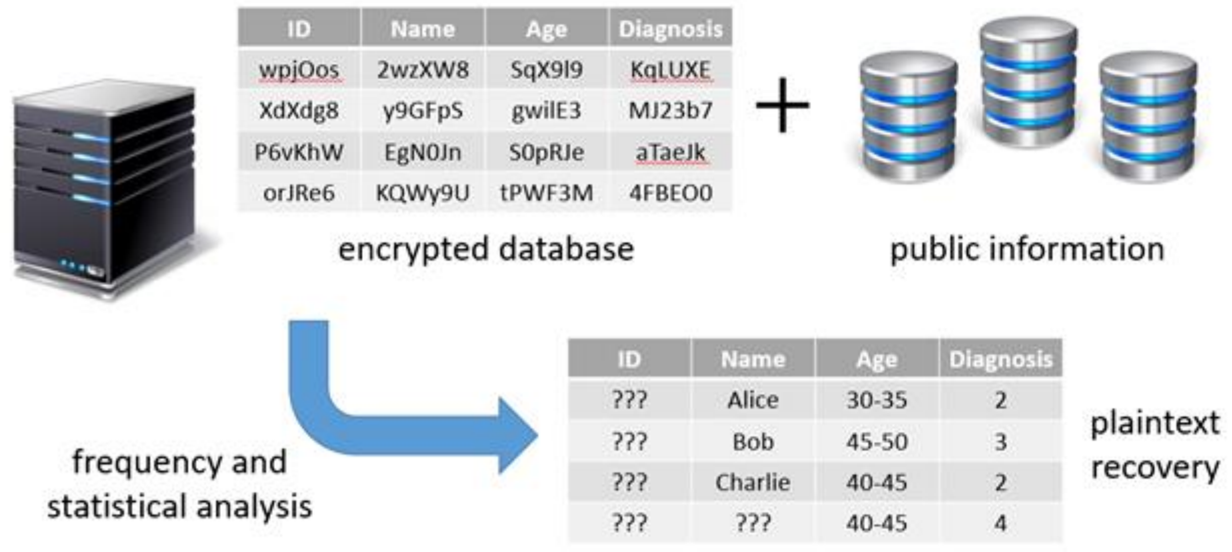
public information

frequency and statistical analysis

ID	Name	Age	Diagnosis
???	Alice	30-35	2
???	Bob	45-50	3
???	Charlie	40-45	2
???	???	40-45	4

plaintext recovery

Inference Attacks [NKW'15, DDC'16, GSBNR'17]



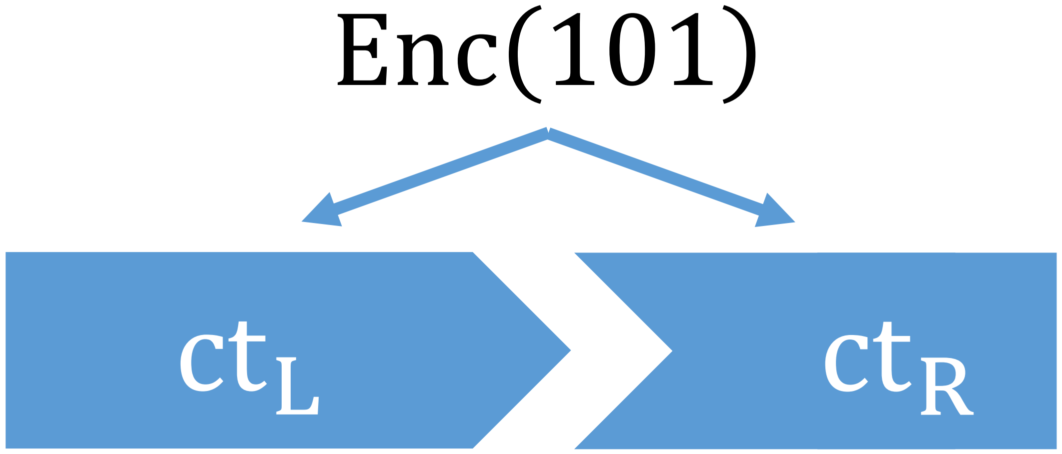
ORE schemes <u>always</u> reveal order of ciphertexts and thus, are vulnerable to <u>offline inference attacks</u>

Can we <u>fully</u> defend against offline inference attacks while remaining legacy-friendly?

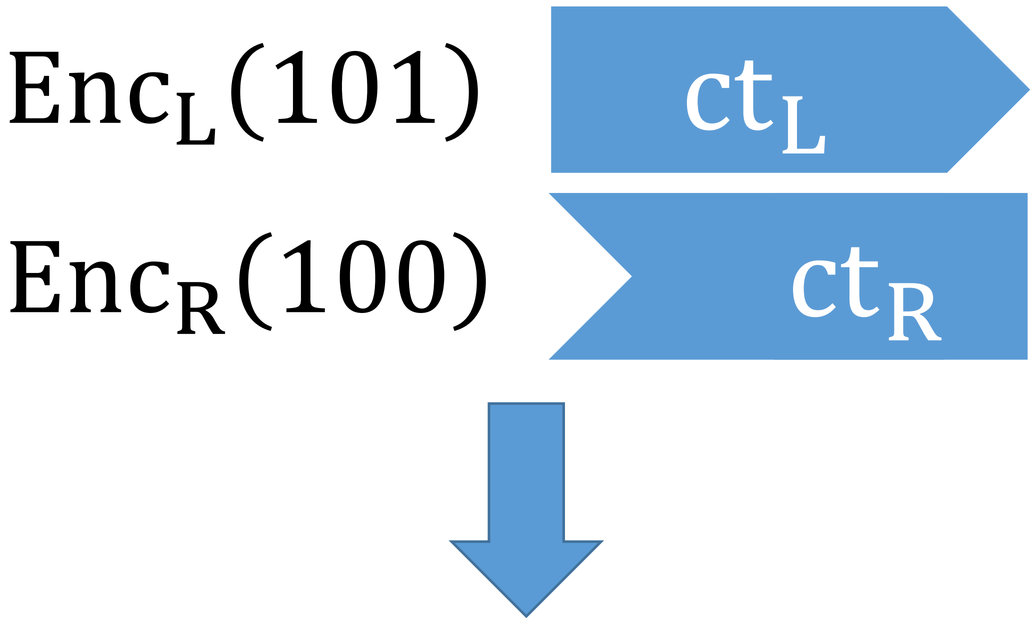
ORE with Additional Structure

Desired functionality: range queries on encrypted data

Key primitive: order-revealing encryption scheme where ciphertexts have a “decomposable” structure



ciphertexts naturally split into two components

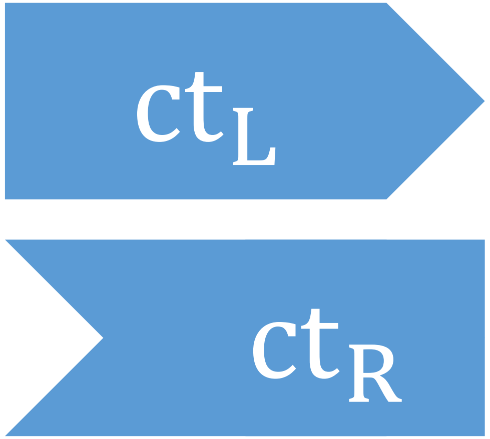


greater than

ORE with Additional Structure

$\mathrm{Enc_L}(101)$

$\mathrm{Enc_R}(100)$



comparison can be performed between left ciphertext and right ciphertext

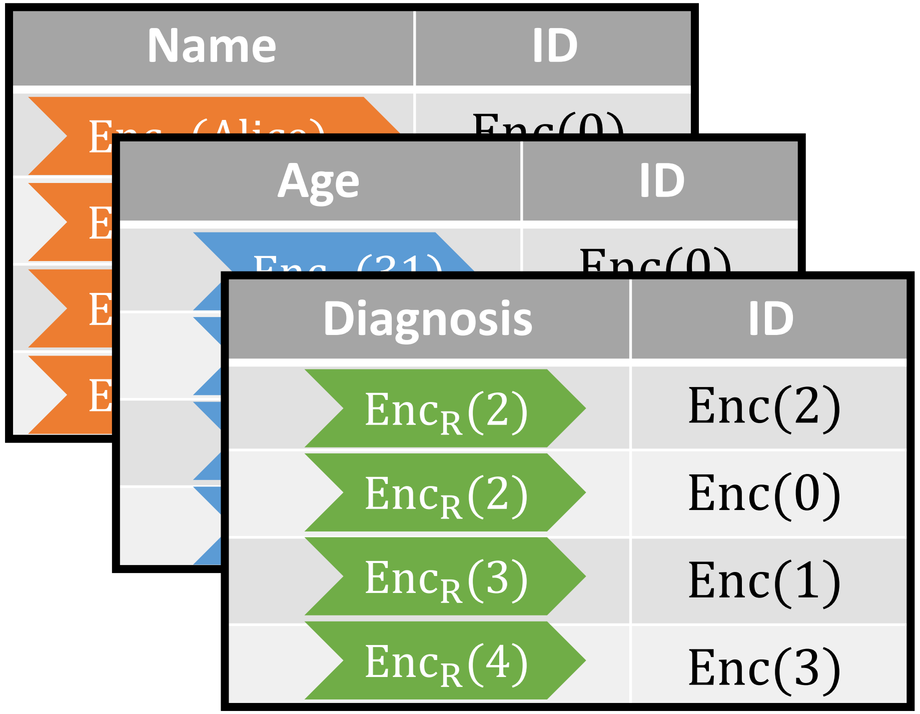
right ciphertexts provide **semantic security**!

robustness against offline inference attacks!

Encrypted Range Queries

ID	Name	Age	Diagnosis
0	Alice	31	2
1	Bob	47	3
2	Charlie	41	2
3	Inigo	45	4

build encrypted index



separate index for each searchable column, and using independent ORE keys

store right ciphertexts in sorted order

Age	ID
$Enc_R(31)$	Enc(0)
$Enc_R(41)$	Enc(2)
$Enc_R(45)$	Enc(3)
$Enc_R(47)$	Enc(1)

record IDs encrypted under independent key

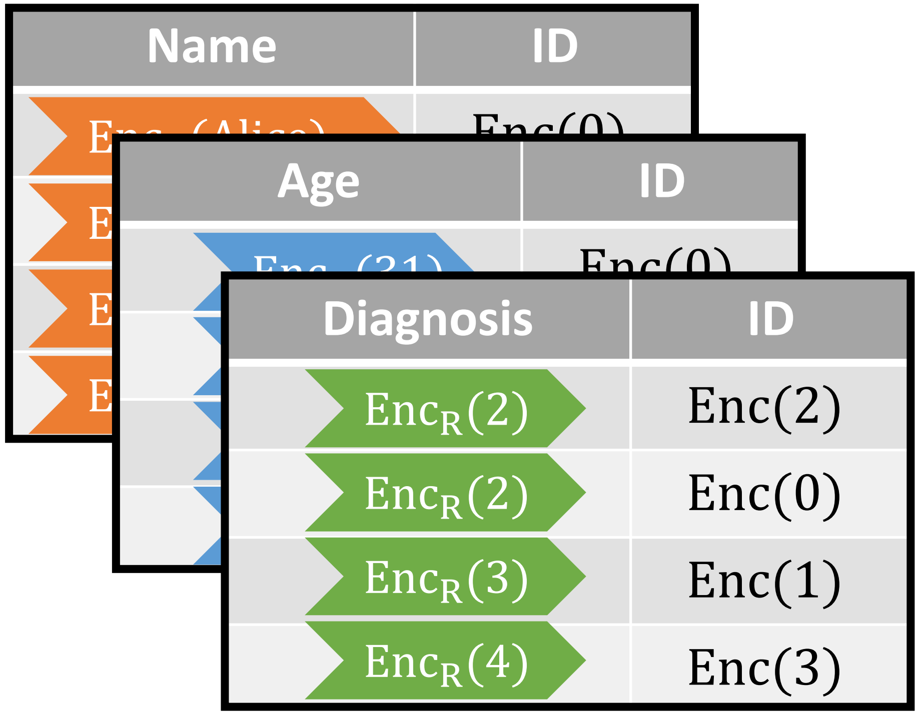
Encrypted Range Queries

Encrypted database:

ID	Name	Age	Diagnosis
0	Alice	31	2
1	Bob	47	3
2	Charlie	41	2
3	Inigo	45	4

columns (other than ID) are encrypted using a semantically-secure encryption scheme

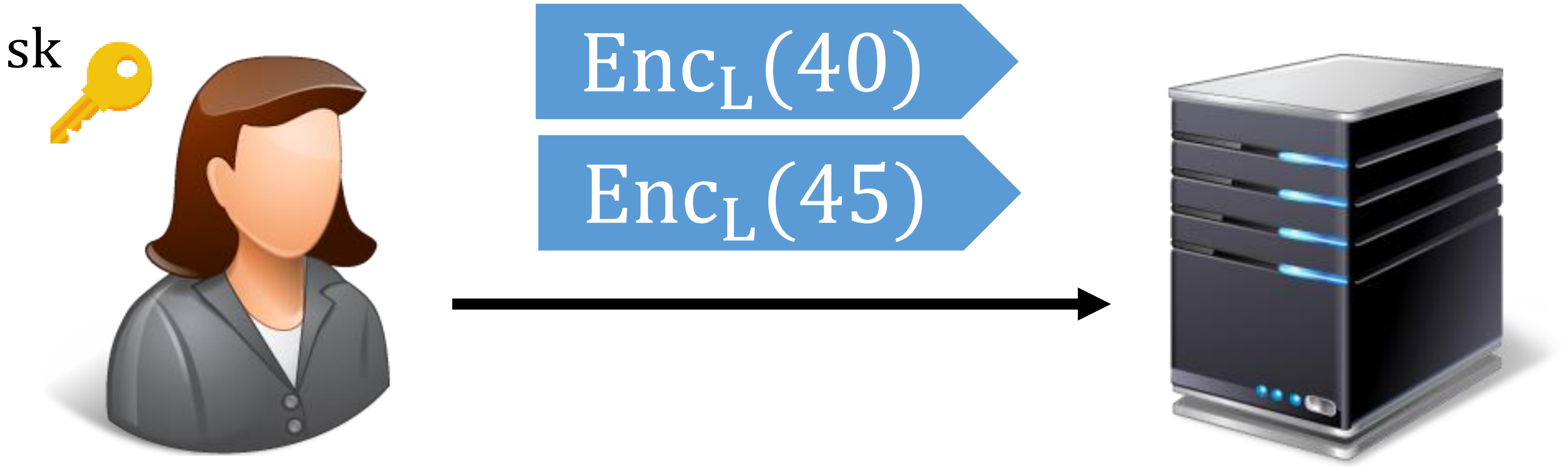
clients hold (secret) keys needed to decrypt and query database



encrypted search indices

Encrypted Range Queries

Query for all records where $40 \geq \mathrm{age} \geq 45$:



Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

$\text{Enc}_\text{L}(40)$

$\text{Enc}_\text{L}(45)$

Age	ID
$\text{Enc}_\text{R}(31)$	$\text{Enc}(0)$
$\text{Enc}_\text{R}(41)$	$\text{Enc}(2)$
$\text{Enc}_\text{R}(45)$	$\text{Enc}(3)$
$\text{Enc}_\text{R}(47)$	$\text{Enc}(1)$

Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

$\text{Enc}_L(40)$

$\text{Enc}_L(45)$

Age	ID
$\text{Enc}_R(31)$	$\text{Enc}(0)$
$\text{Enc}_R(41)$	$\text{Enc}(2)$
$\text{Enc}_R(45)$	$\text{Enc}(3)$
$\text{Enc}_R(47)$	$\text{Enc}(1)$

use binary search to determine endpoints (comparison via ORE)

Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

$\text{Enc}_\text{L}(40)$

$\text{Enc}_\text{L}(45)$

Age	ID
$\text{Enc}_\text{R}(31)$	$\text{Enc}(0)$
$\text{Enc}_\text{R}(41)$	$\text{Enc}(2)$
$\text{Enc}_\text{R}(45)$	$\text{Enc}(3)$
$\text{Enc}_\text{R}(47)$	$\text{Enc}(1)$

use binary search to determine endpoints (comparison via ORE)

Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

$\text{Enc}_L(40)$

$\text{Enc}_L(45)$

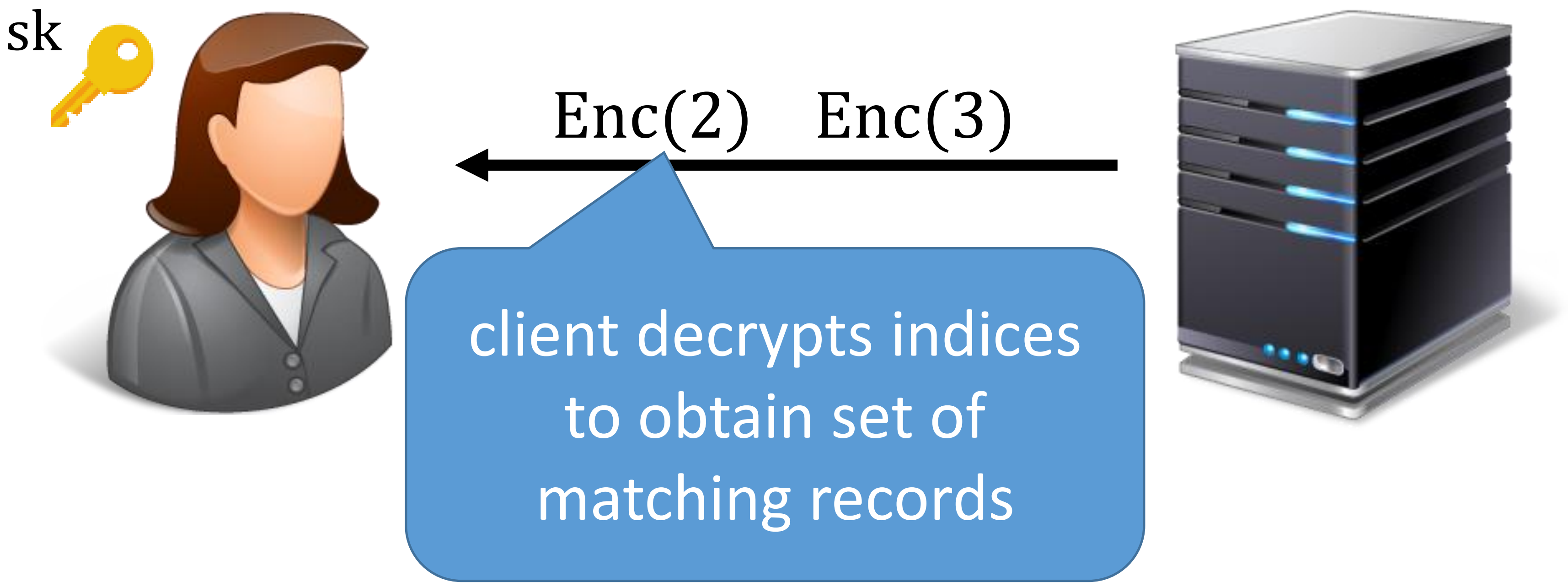
Age	ID
$\text{Enc}_R(31)$	Enc(0)
$\text{Enc}_R(41)$	Enc(2)
$\text{Enc}_R(45)$	Enc(3)
$\text{Enc}_R(47)$	Enc(1)

return encrypted indices that match query

use binary search to determine endpoints (comparison via ORE)

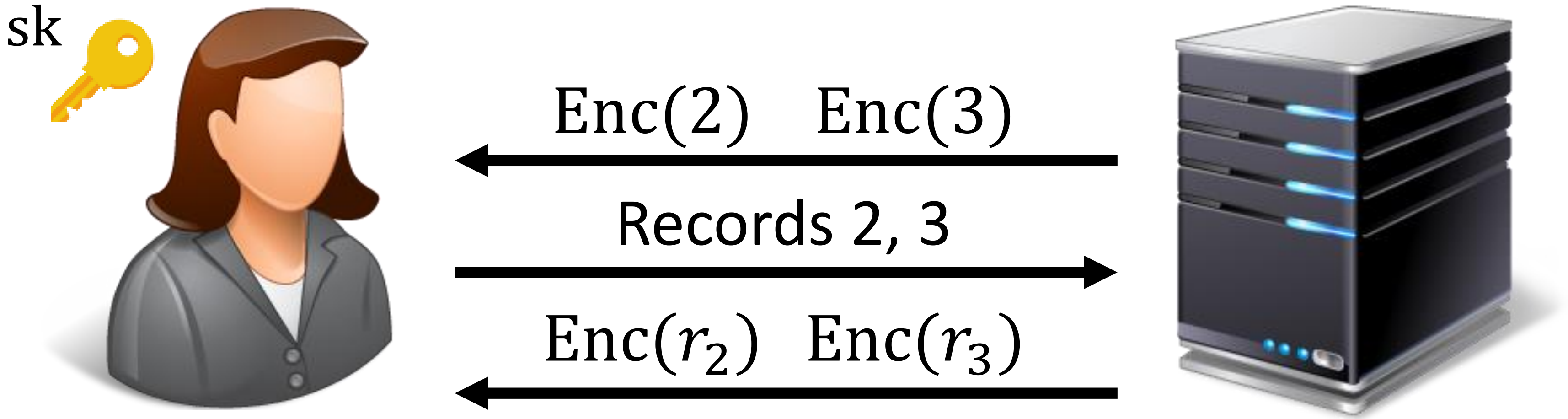
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:



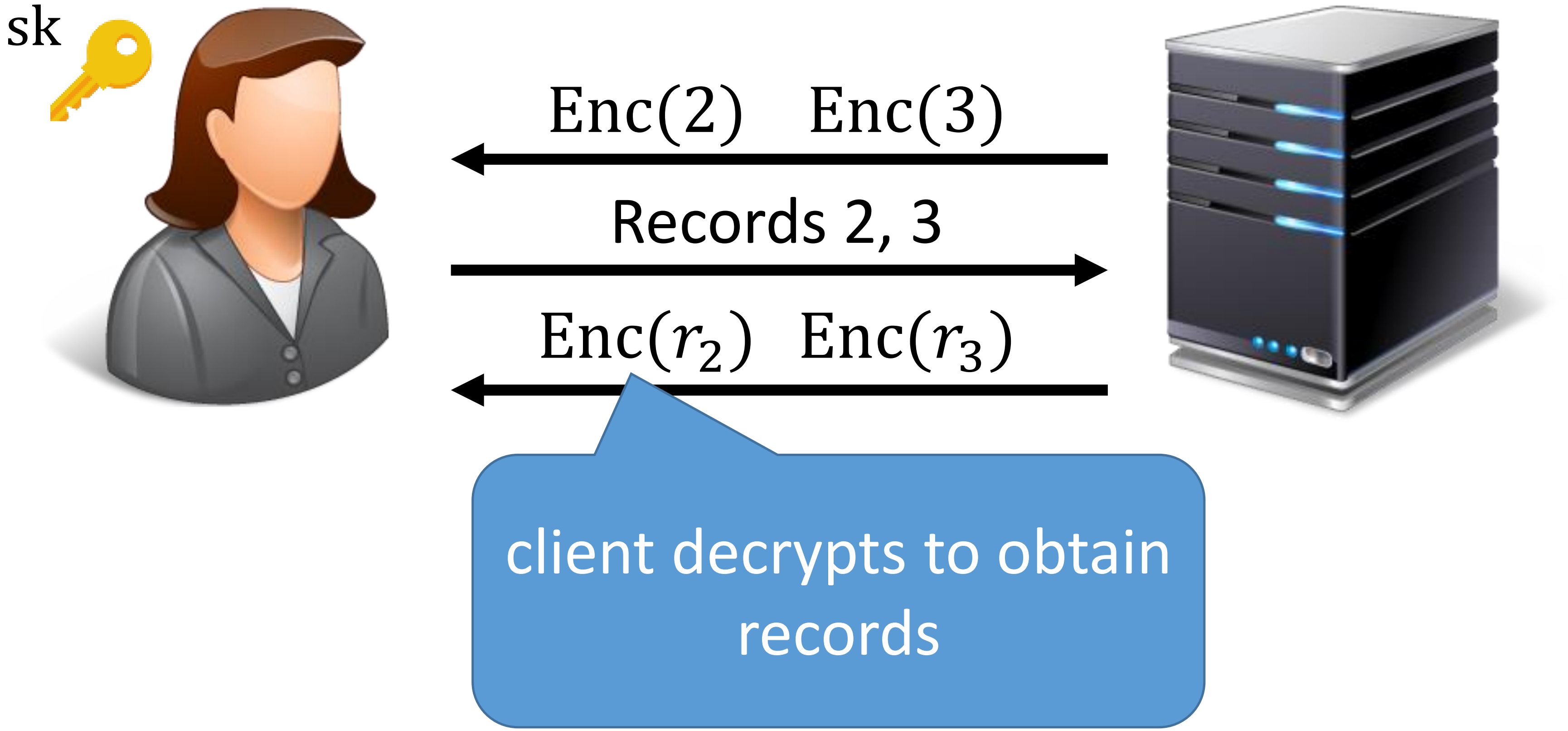
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:



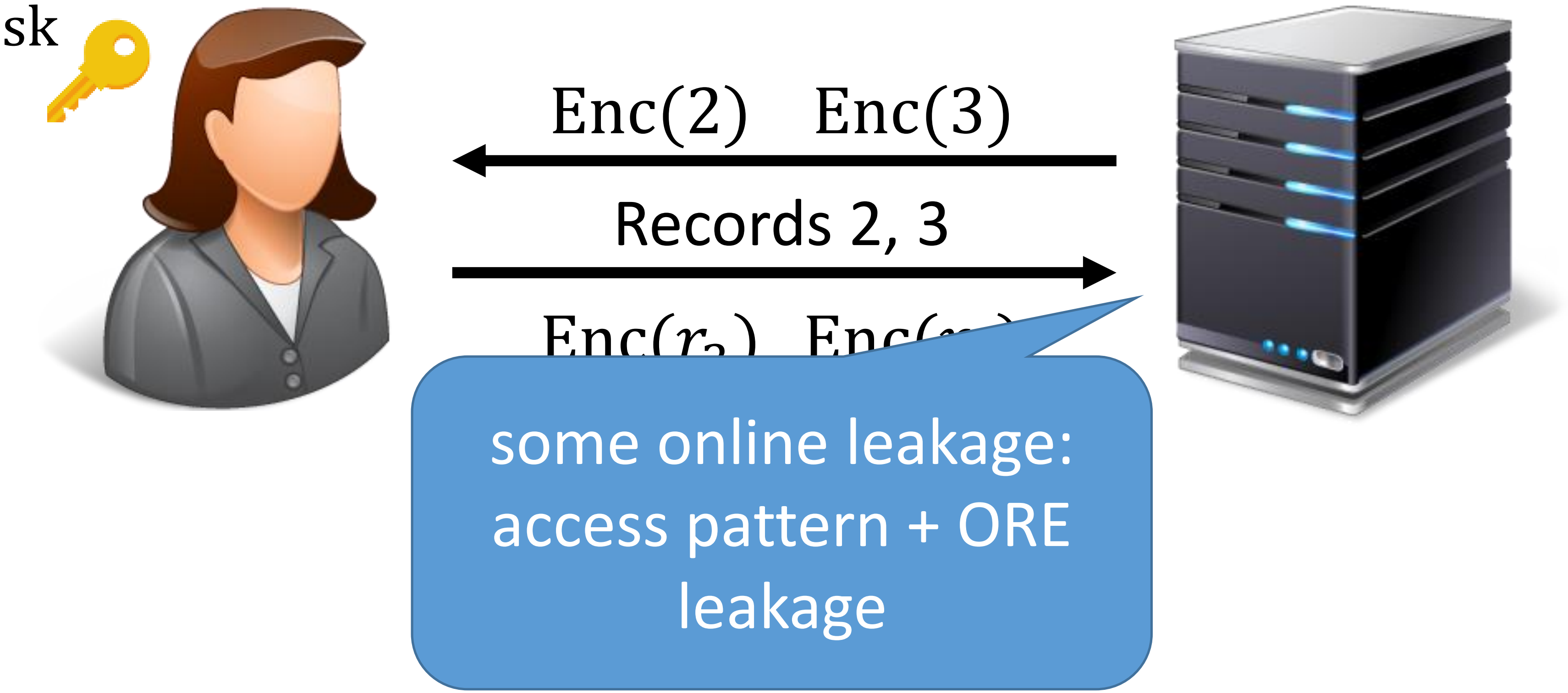
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:



Encrypted Range Queries

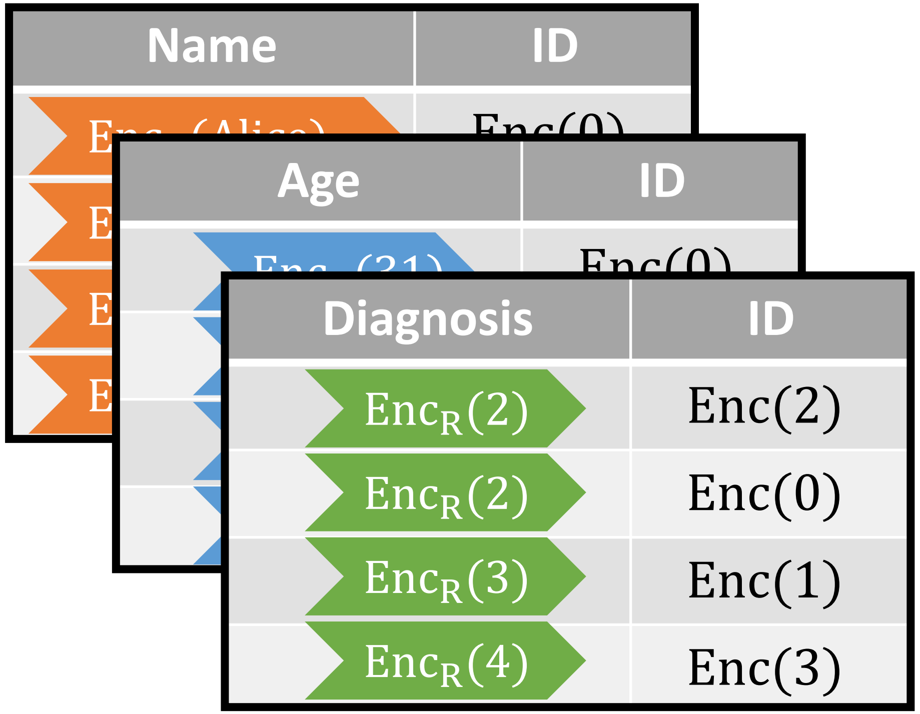
Query for all records where $40 \geq \text{age} \geq 45$:



Encrypted Range Queries

Encrypted database (view of the snapshot adversary):

ID	Name	Age	Diagnosis
0	Alice	31	2
1	Bob	47	3
2	Charlie	41	2
3	Inigo	45	4



encrypted search indices

encrypted database is semantically secure!

Perfect offline security

A New ORE Scheme [LW'16]

"small-domain" ORE with best-possible security

+

domain extension technique from [CLWW'16]

→

ORE with some leakage

first practical ORE construction that can provide best-possible offline security!

Performance Evaluation

Scheme	Encrypt (μs)	Compare (μs)	|ct| (bytes)
OPE [BCLO'09]	3601.82	0.36	8
[CLWW'16] ORE	2.06	0.48	8
[LW'16] ORE (8-bit blocks)	54.87	0.63	224

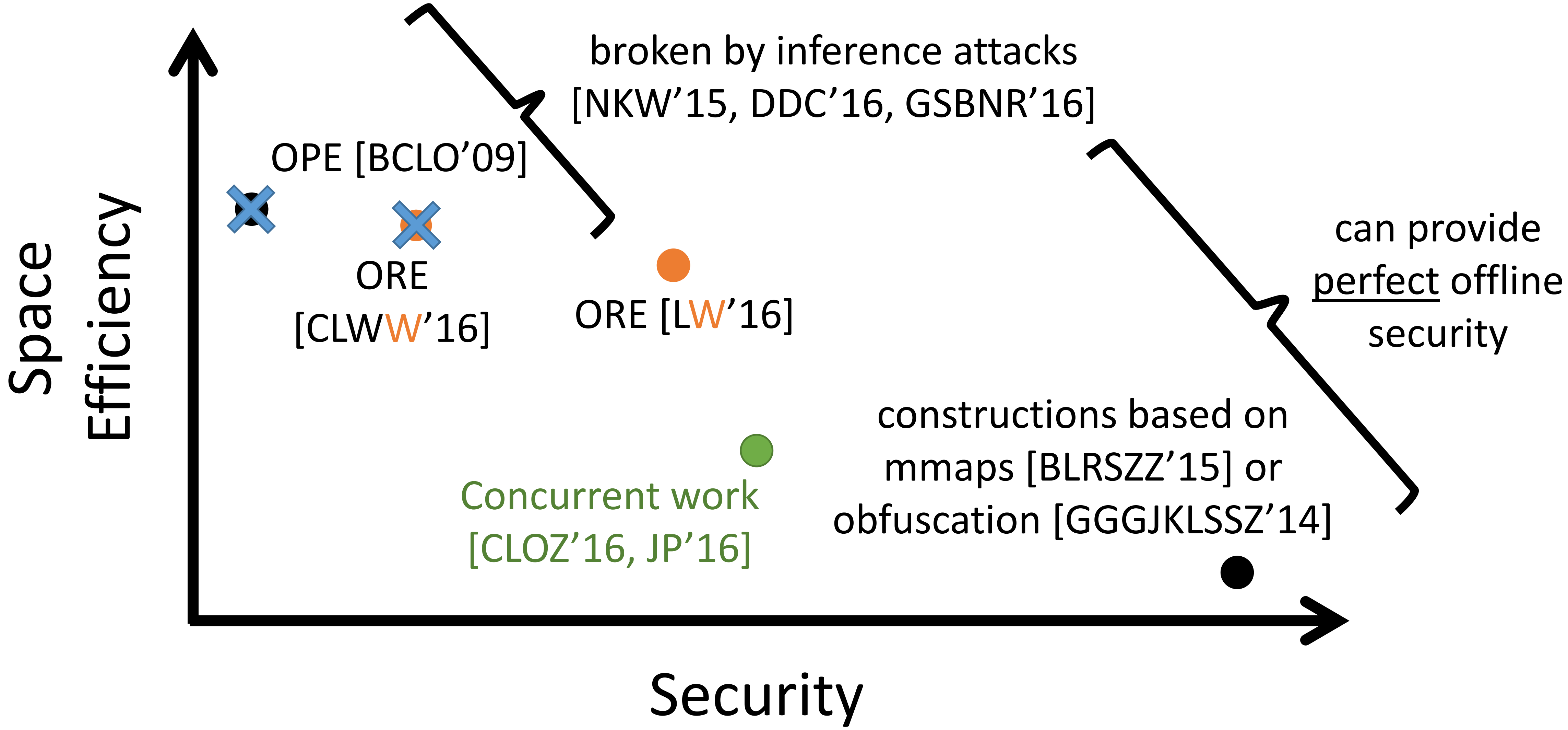
Benchmarks taken for C implementation of different schemes (with AES-NI). Measurements for encrypting 32-bit integers.

Performance Evaluation

Scheme	Encrypt (μs)	Compare (μs)	|ct| (bytes)
OPE [BCLO'09]	3601.82	0.36	8
[CLWW'16] ORE	2.06	0.48	8
[LW'16] ORE (8-bit blocks)	54.87	0.63	224

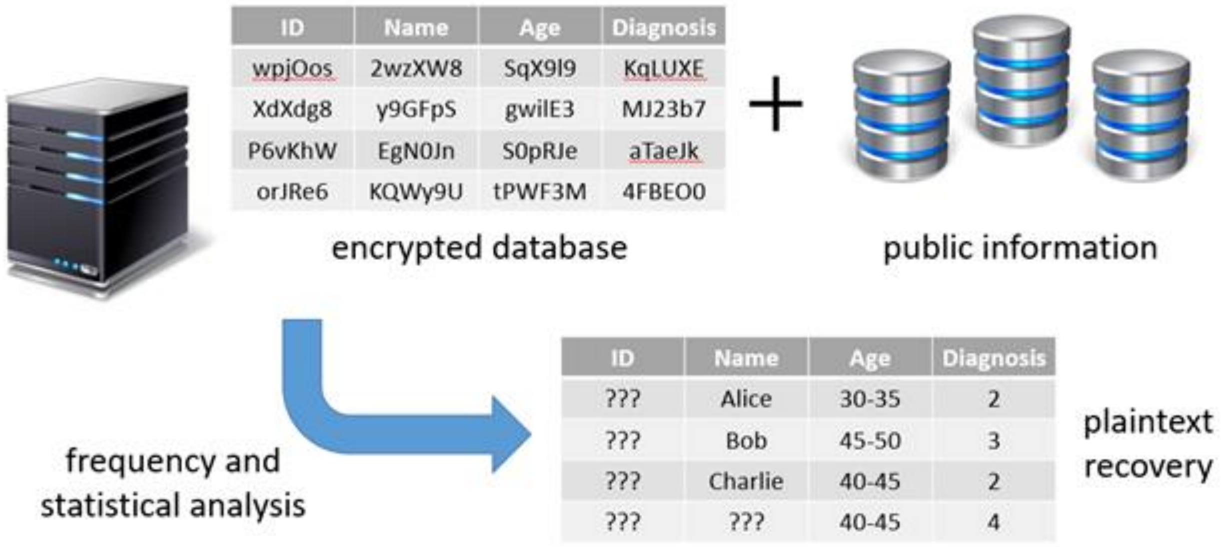
Encrypting byte-size blocks is 65x faster than OPE, but ciphertexts are 30x longer. Security is substantially better.

The Landscape of ORE



Not drawn to scale

Conclusions



- Inference attacks render direct usage of ORE insecure
- However, ORE is still a useful building block for encrypted databases

- Introduced new paradigm for constructing ORE that enables range queries in a way that is mostly <u>legacy-compatible</u> and provides <u>offline semantic security</u>
- New ORE construction that is concretely efficient with strong security

Questions?

Website: `https://crypto.stanford.edu/ore/`

Code: `https://github.com/kevinlewi/fastore`