cherenkov@riseup.net

a.davidson@fct.unl.pt

Deploying Private Information Retrieval for Real Databases

Cryptographic Applications Workshop @ Eurocrypt 2024

Sofía Celi, Alex Davidson

Brave Software

NOVALINCS
LABORATORY FOR COMPUTER SCIENCE AND INFORMATICS

Universidade NOVA de Lisboa, Portugal

Private Information Retrieval (PIR)

Considers the cryptographic problem of retrieving data from **untrusted**, remote databases.

- ❏ Parties:
    - ❏ Client
    - ❏ Server (one or multiple)
- ❏ Steps:
    - ❏ Query
    - ❏ Response

Current outlook

❑ Very active research area

❑ Promising efficiency

❑ Variety of applications

Issues to discuss today

- ❑ Which performance criteria / applications matter?

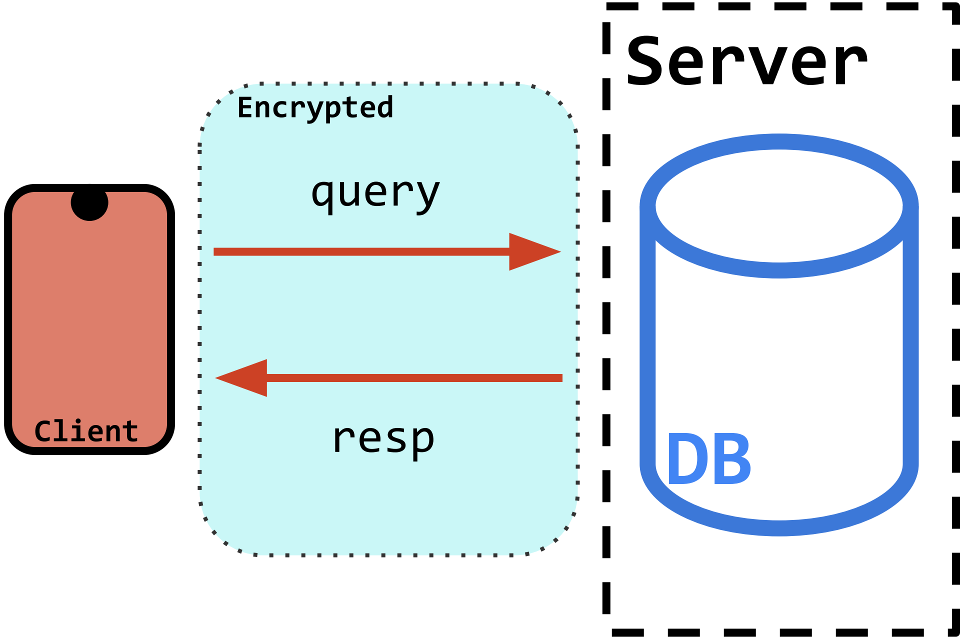
- ❑ What databases should be supported?

- ❑ How to unify PIR design for real-world?

Private Information Retrieval

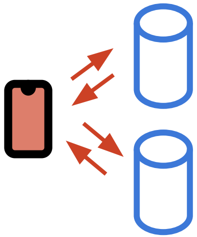
Base-case for this talk:

- ❑ Single (semi-honest) server
- ❑ query / resp =
  - ❑ DB.get(i)
  - ❑ DB.get(kw)
  - ❑ SELECT * FROM DB WHERE <condition>
  - ❑ …
- ❑ DB is assumed public
- ❑ May involve offline preprocessing



Why single-server?

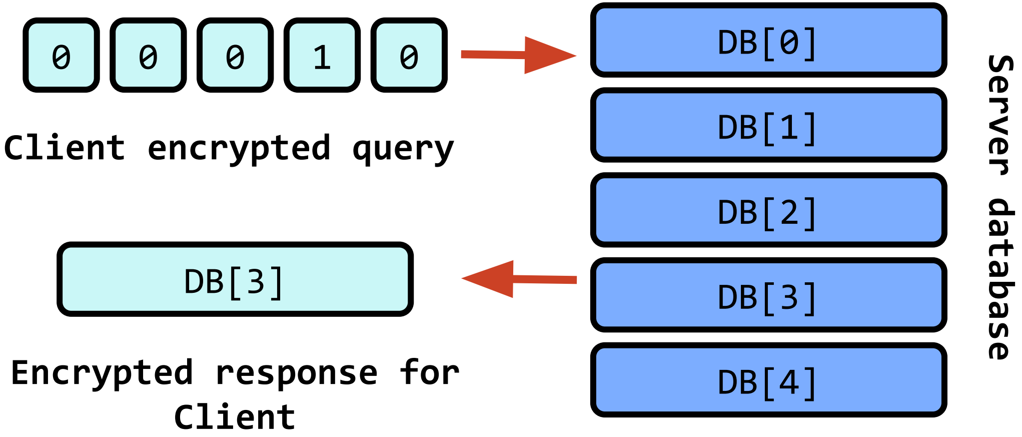
Multi-server PIR is more efficient and gives information-theoretic guarantees, but:



- No clear process (legal/practical) for finding independent, non-colluding partners
  - Co-deployment seems like a form of collusion
- Single-server efficiency is improving
- We already believe computational assumptions

Single-server constructions

Based on practical® constructions of homomorphic encryption from LWE or RLWE



- ❑ LWE-based (stateful) are simpler to implement, and process queries faster
- ❑ RLWE-based (stateless/stateful) include optimisations for amortisation, and are more flexible for higher-level applications

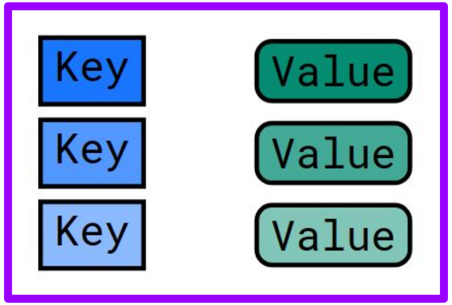
Current state of PIR

❏ High-throughput (<< 1 sec query, Θ(GBps))

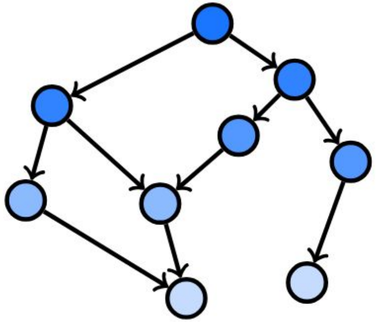
❏ High rate (|Enc(r)| ~ 3*|r|, Θ(KB))

❏ Practical queries for index or keywords

Real databases

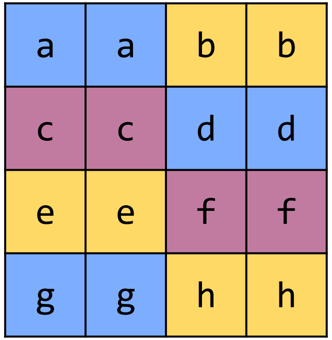


Key Value
Key Value
Key Value

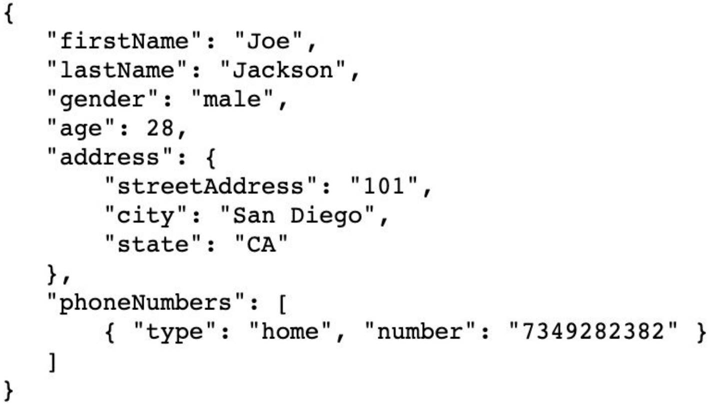


col_1	col_2	col_3
x_1	y_1	z_1
x_2	y_2	z_2
x_3	y_3	z_3

Issue #1: Non-uniform data

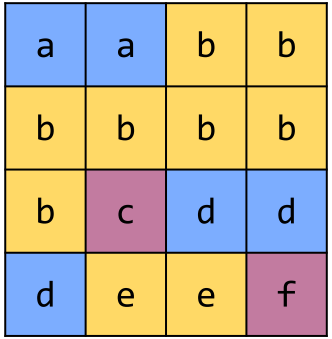


a	a	b	b
c	c	d	d
e	e	f	f
g	g	h	h



```
{
    "firstName": "Joe",
    "lastName": "Jackson",
    "gender": "male",
    "age": 28,
    "address": {
        "streetAddress": "101",
        "city": "San Diego",
        "state": "CA"
    },
    "phoneNumbers": [
        { "type": "home", "number": "7349282382" }
    ]
}
```

- ❑ Data-specific privacy?
- ❑ Efficiency for multi-layer keys?
- ❑ Client storage?



a	a	b	b
b	b	b	b
b	c	d	d
d	e	e	f

Goals:

Design PIR with real databases in mind.

*Security and performance modelling should take **database format** into account.*

Issue #2: Necessary Applications?

Some deployments / related technologies exist:

❑ Brave (compromised credential-checking, TBD)
❑ Blyss (https://github.com/blyssprivacy/sdk)
❑ Google (Device Enrollment)
❑ Microsoft (Password Monitor)

More complex use-cases (not deployed):

❑ Approximate nearest-neighbor: Brave News
❑ Private search: TipToe
❑ Oblivious document ranking: Coeus

***Open questions*:**

1. Build complex functions embedded directly into queries

2. Basic PIR used as part of higher-level application

Issue #3: Rapidly-updating databases

Differing update-cycles depending on application

- ❑ Slower cadence: contact discovery, compromised credentials
- ❑ Faster cadence: safe browsing, recommendation systems (*)

Stateful PIR: require state regeneration with every update

<u>Goals:</u>

1. *More benchmarking of stateful PIR with support for incremental updates*
2. *More efficient (and simpler®) stateless PIR*

Issue #4: Configurability

Different performance metrics matter to different people

- ❑ **Financial costs** may be more important than bandwidth for those without hardware
- ❑ **Server load** may be more important for CDNs, Google, etc.
- ❑ **Device load / bandwidth** for mobile devices

***Question**: Separate approaches for each criteria? Or support for simple re-parametrisation?*

Issue #5: Important security properties

- ❏ Does a semi-honest, public DB satisfy all applications?
  - ❏ **Probably not**: compromised credentials, contact-checking…

- ❏ Private DB + semi-honest seems important
  - ❏ Privacy measures are *ad-hoc* (OPRF, masking).
  - ❏ Implications: sub-optimal rounds, not post-quantum…

- ❏ Authenticated/verifiable/malicious PIR exists, is this what we should be using everywhere?

Issue #6: Simplicity®

- ❑ FHE-based PIR is very complex
  - ❑ Libraries are hard to audit/verify
  - ❑ Non-standard security parameters
  - ❑ Low-level optimisations required for PIR

- ❑ AHE-based is simpler and configurable
  - ❑ Restricted applications
  - ❑ Real-time databases require more complex RLWE

<u>Question</u>: *Do we want* ***widespread****, or* ***centralised*** *deployments?*

Conclusions

- PIR is a ***central*** cryptographic functionality
- However, not much evidence of real-world usage
- One-size-fits-all scheme seems **unlikely**
- New approaches need to consider and prioritise:
  - **Real** DB representations
  - **Real-time** updates
  - Enhanced **functionality** and **security** properties
  - More consideration of **higher-level** applications

Thank you!

@claucece @alxdavids

cherenkov@riseup.net

a.davidson@fct.unl.pt

Our opinionated reading list

- Keyword-based PIR:
  - “Call Me By My Name: Simple, Practical Private Information Retrieval for Keyword Queries”: https://eprint.iacr.org/2024/092
  - “Don't be Dense: Efficient Keyword PIR for Sparse Databases”: https://eprint.iacr.org/2023/466
- Security properties:
  - “Fully Malicious Authenticated PIR”: https://eprint.iacr.org/2023/1804
  - “VeriSimplePIR: Verifiability in SimplePIR at No Online Cost for Honest Servers”: https://eprint.iacr.org/2024/341
- Complex queries:
  - “Private Web Search with Tiptoe”: https://eprint.iacr.org/2023/1438
  - “Coeus: A System for Oblivious Document Ranking and Retrieval”: https://eprint.iacr.org/2022/154