# Cryptographically Protected Database Search

*Benjamin Fuller*, Mayank Varia, Arkady Yerukhimovich, Emily Shen,
Ariel Hamlin, Vijay Gadepally,
Richard Shay, Darby Mitchell, Robert Cunningham

benjamin.fuller@uconn.edu

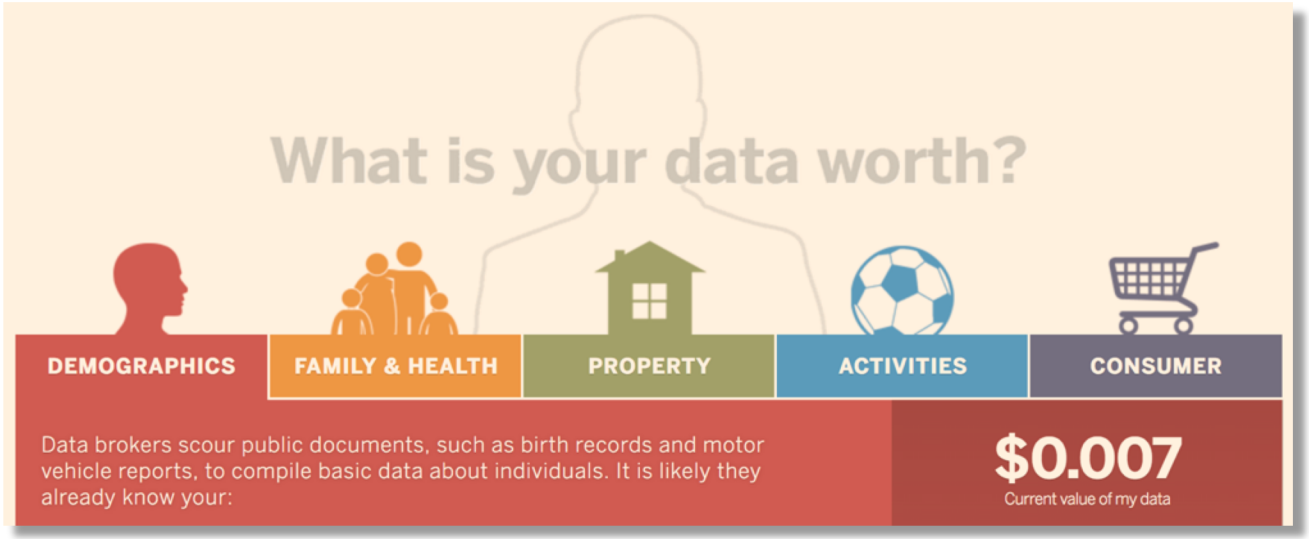


# The Data Economy

**The Rise of the Data Economy: Driving Value through Internet of Things Data Monetization**

A Perspective for Chief Digital Officers and Chief Technology Officers

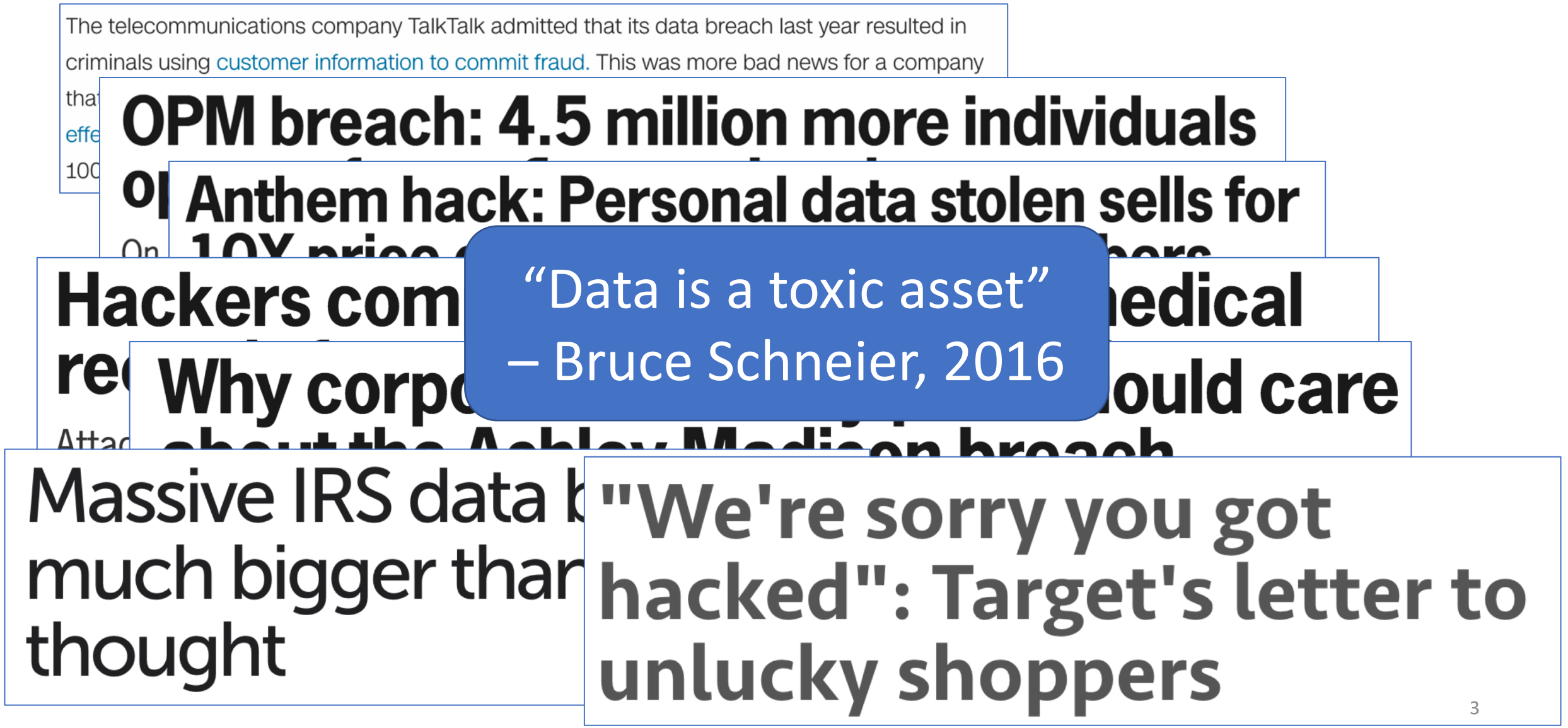
By Albert Opher, Alex Chou, Andrew Onda, and Krishna Sounderrajan




**Interesting takeaway No. 1:** 61% of respondents "acknowledge that big data is now a driver of revenues in its own right and is becoming as valuable to their businesses as their existing products and services."

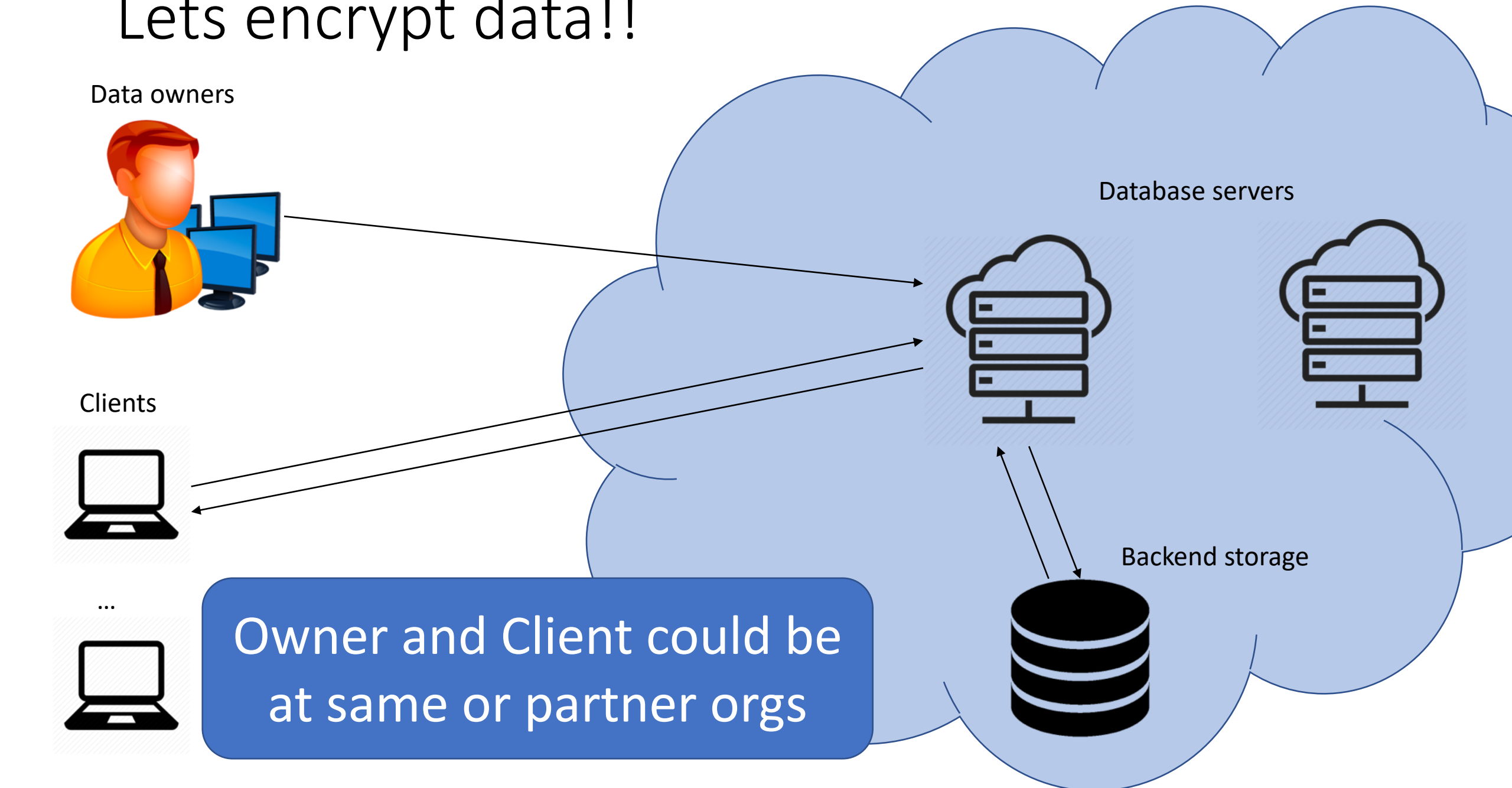
"Data is the new oil"
– Shivon Zilis, Bloomberg Beta
"Data will become a currency"
– David Kenny, IBM Watson
"… the fourth industrial revolution is connectivity and data"
– Mukesh Ambani, Reliance

# Value implies Risk

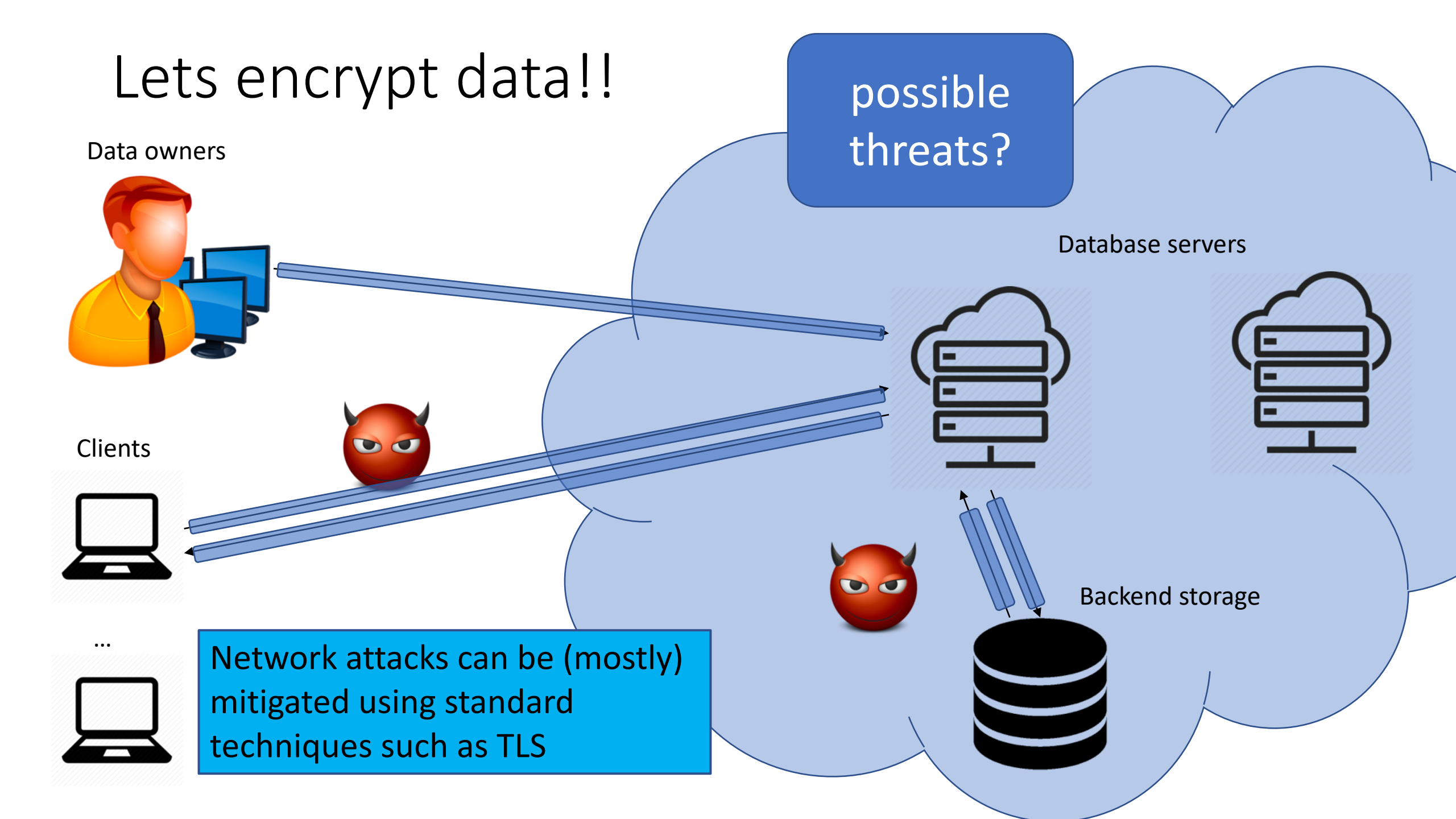



"Data is a toxic asset"
– Bruce Schneier, 2016

# Lets encrypt data!!



# Lets encrypt data!!



# Lets encrypt data!!

Data owners

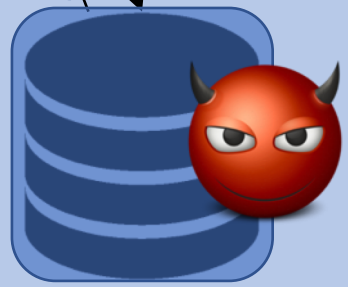
Clients

...

possible threats?

Database servers

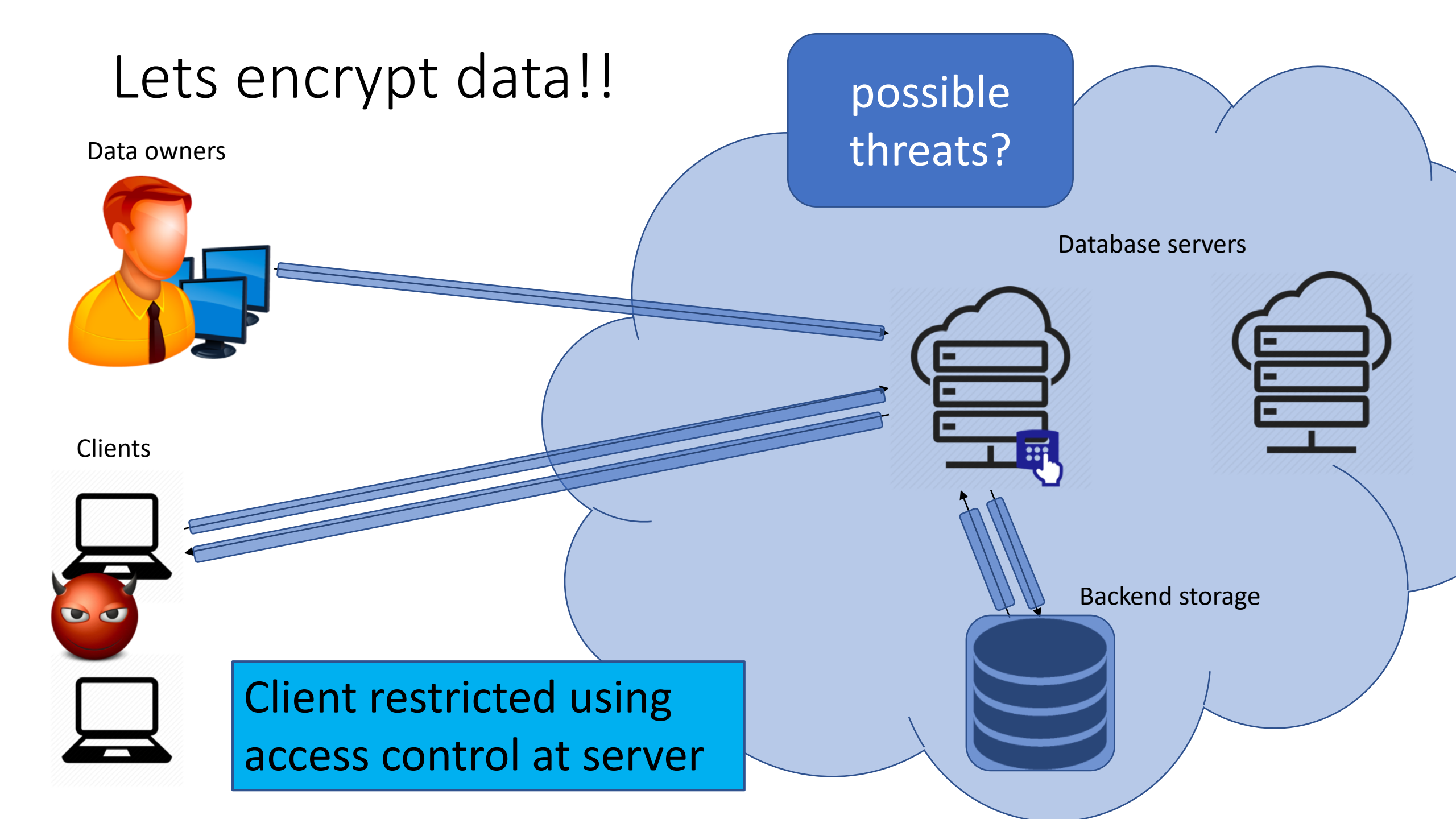
Backend storage



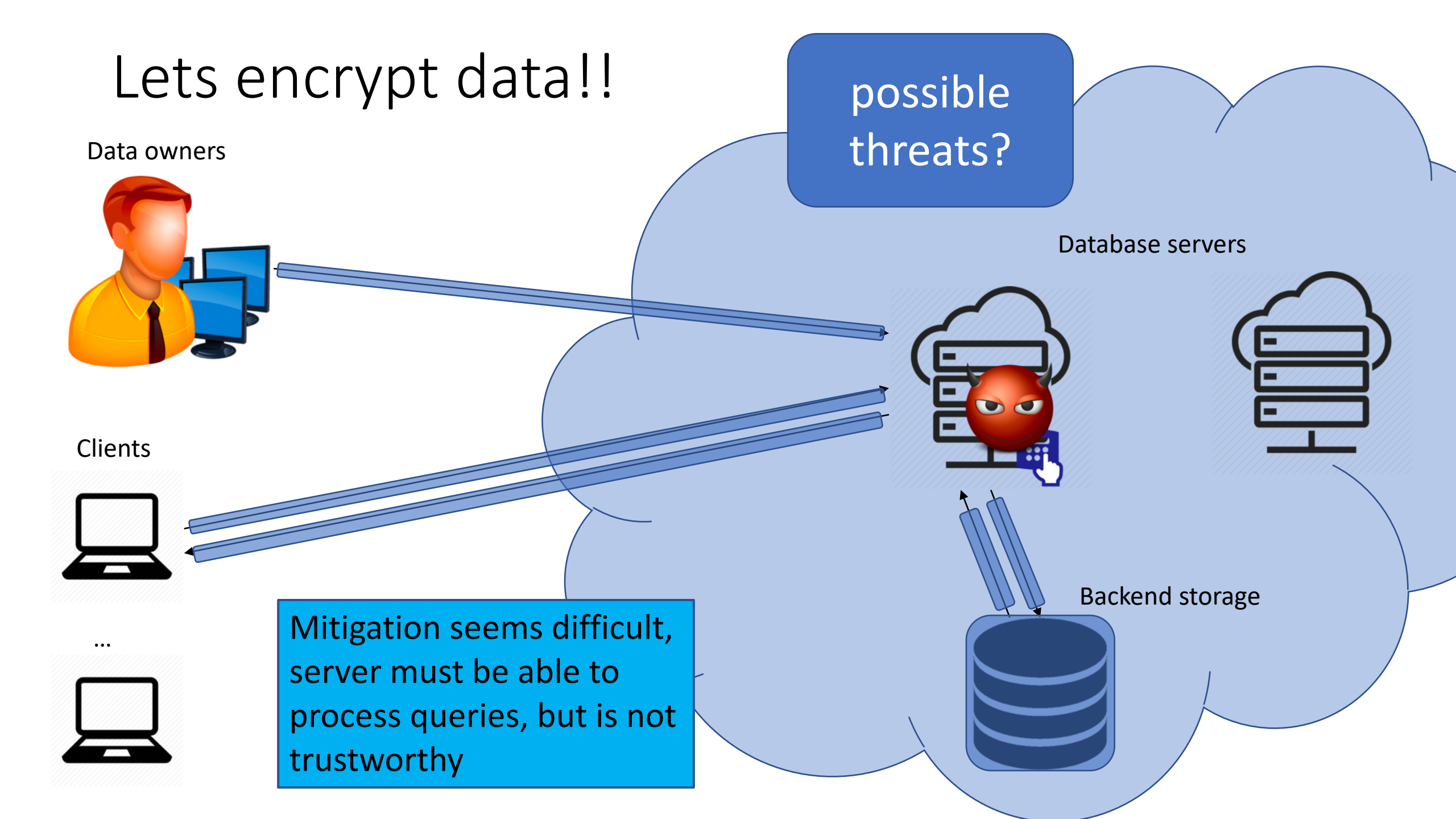
Encrypt data at rest, required for some data types due to regulation (HIPAA)

- Capability exists in modern databases (e.g. Accumulo)

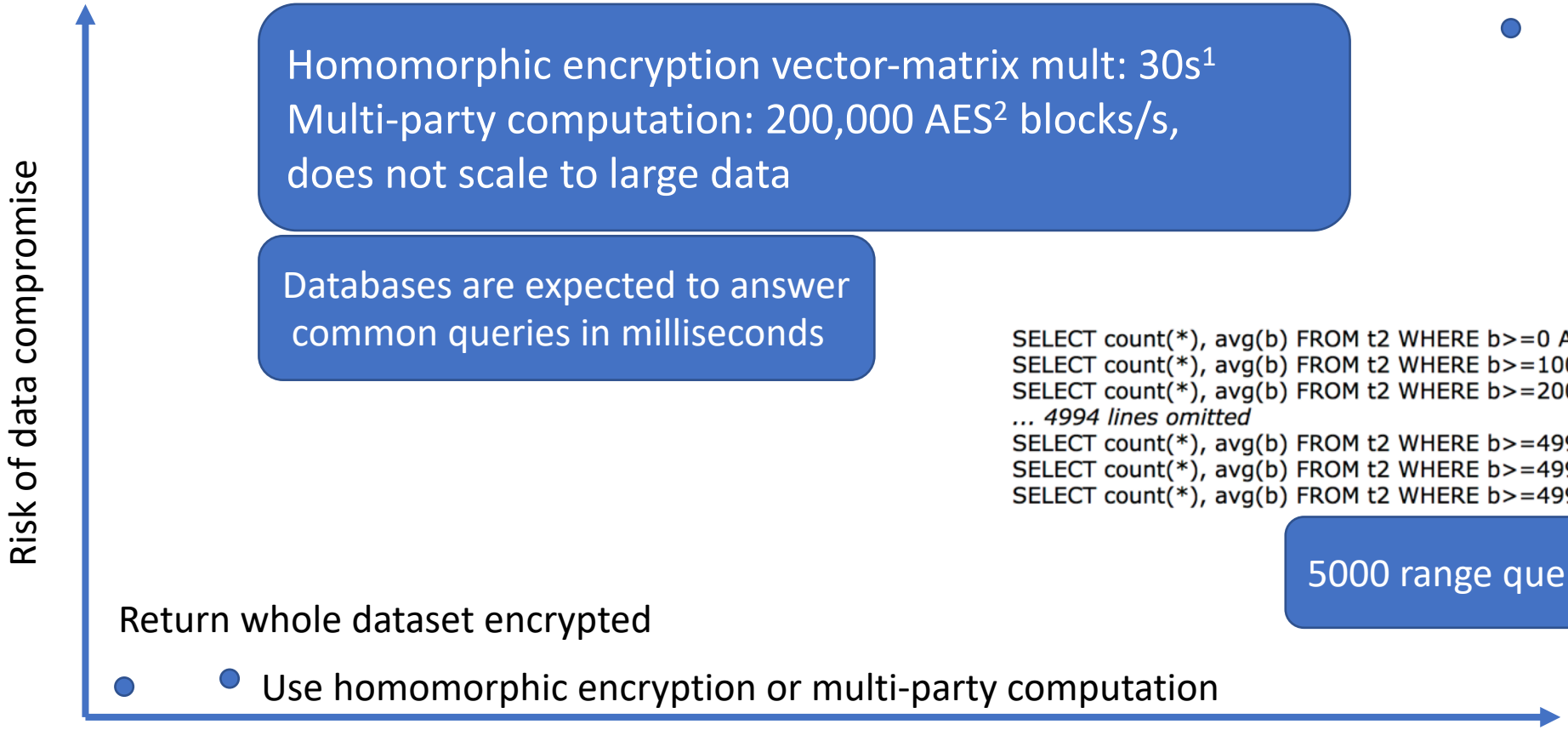
# Lets encrypt data!!



# Lets encrypt data!!



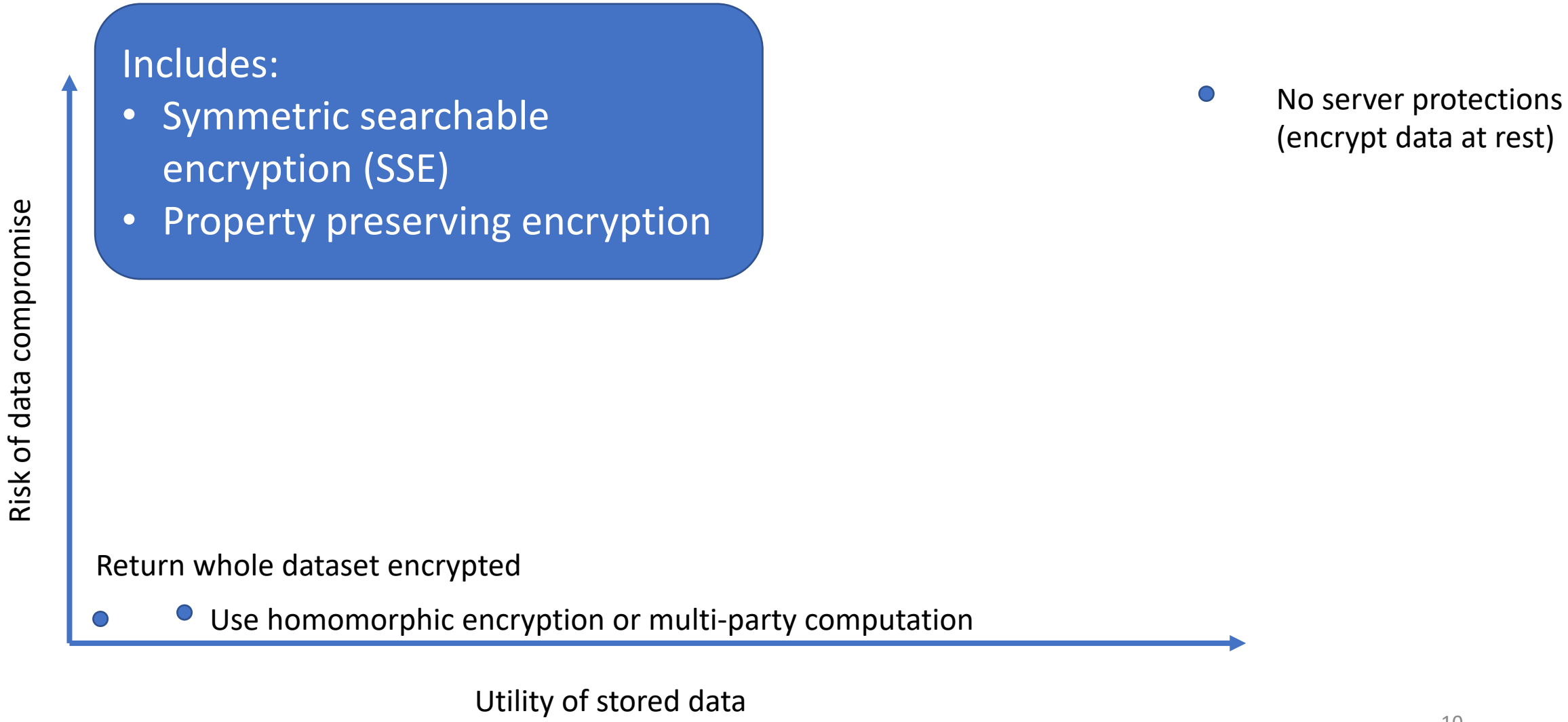
# Cryptographically Protected Search



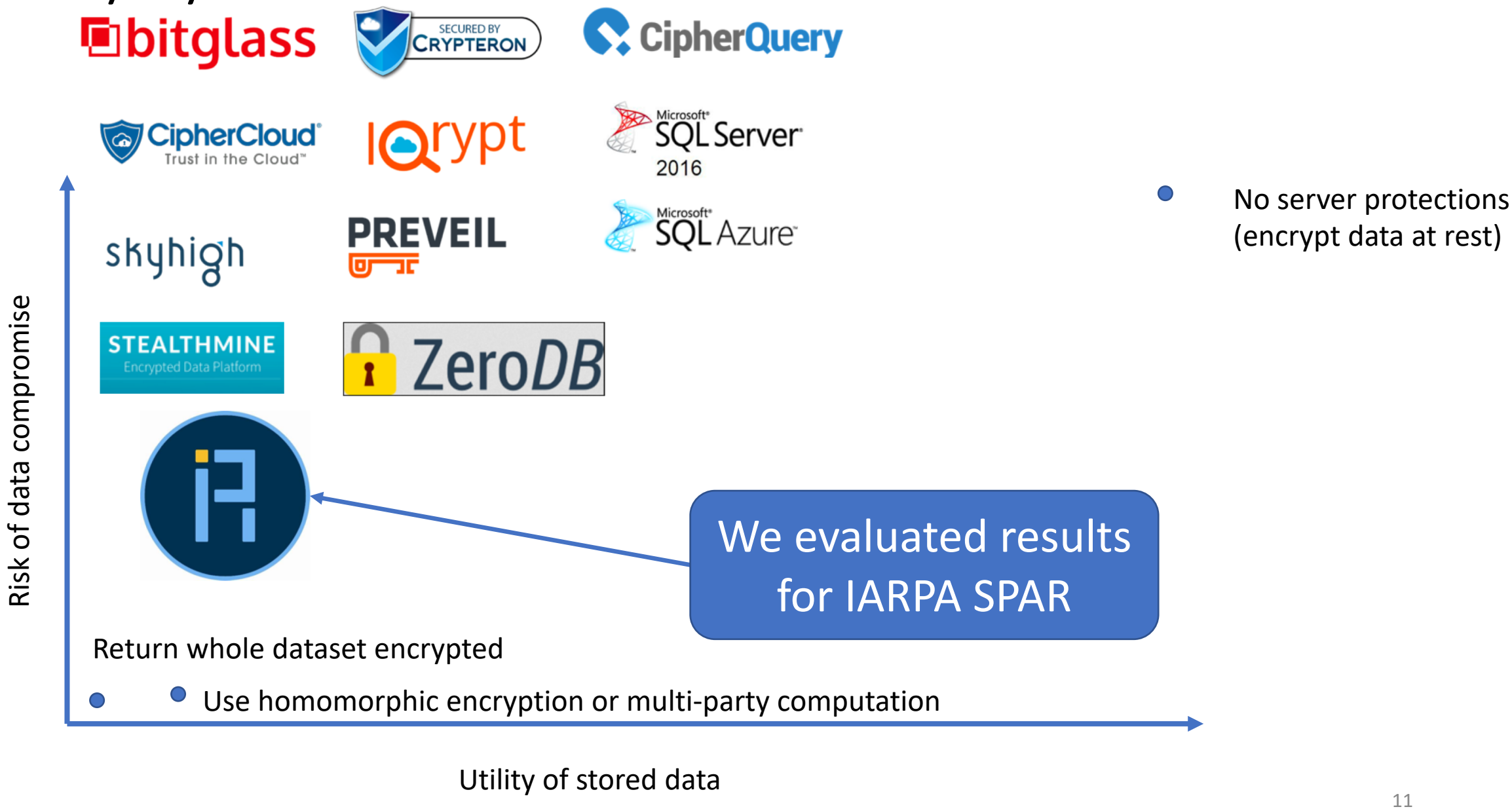

[1]S. Halevi and V. Shoup. (2014) HElib - an implementation of homomorphic encryption. [Online]. Available: https://github.com/shaih/ Helib

[2]M. Keller, E. Orsini, D. Rotaru, P. Scholl, E. Soria-Vazquez, S. Vivek, "Faster Secure Multi-Party Computation of AES and DES Using Lookup Tables," in ACNS 2017

# Cryptographically Protected Search



# Why systematize?



# Outline

- Overview of Protected Search
- Leakage Impacts
- Finding a basis for search results
  - Range queries
    - Compatible approach: Order-Preserving Encryption / CryptDB
    - Custom approach: Partial Order-Preserving Encryption
    - Obliv approach: SisoSPIR
  - Combining queries
- Extending to new database paradigms

# Common Language for Leakage

**Protected search schemes reveal some information about the query, data set, and result set to *each* party.**

**Called *leakage.***

**Difficult to compare,**
**phrased to make proofs work,**
**not to compare schemes**

**Define five types of leakage of increasing impact[1]:**

1. **Structure**
2. **Identifiers**
3. **Predicates**
4. **Equality**
5. **Order**

**Some schemes leak:**

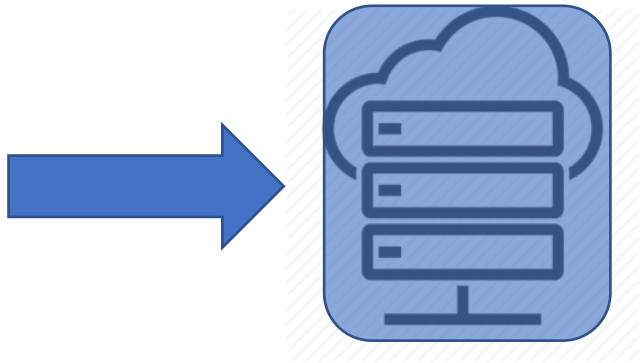
**1. At *Initialization* on entire DB**

***2.* At *Query* on relevant records**

[1]Partially based on S.Kamara, "Structured encryption and leakage suppression," presented at Encryption for Secure Search and Other Algorithms, Bertinoro, Italy, June 2015.

# Hospital Data Set

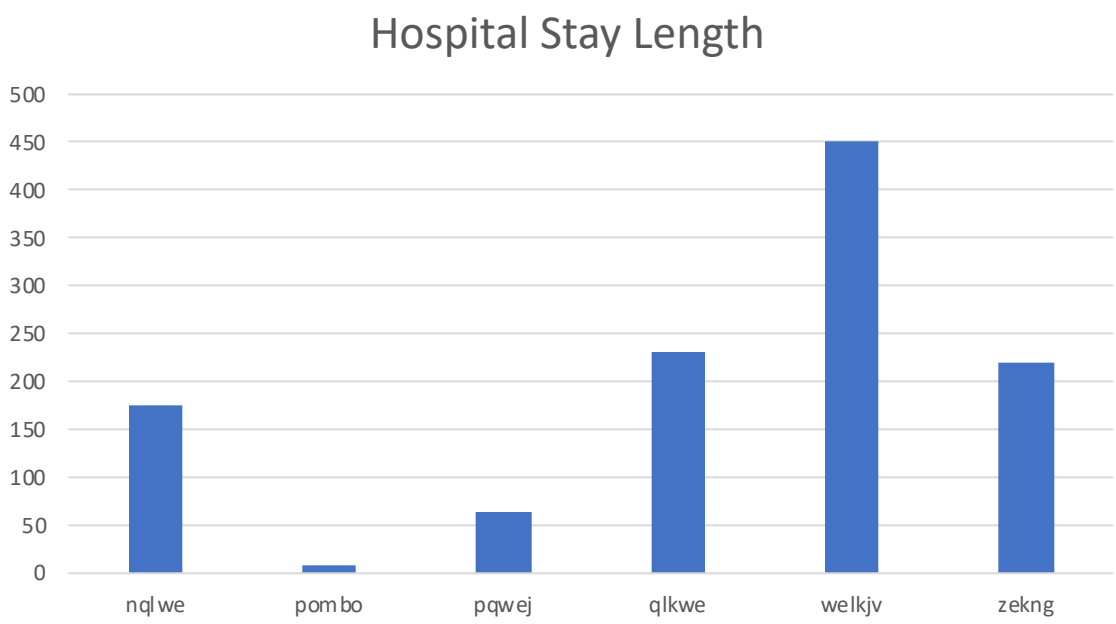
| Birth Month | Length of Stay | Gender | Diagnosis | SSN |
|---|---|---|---|---|
| February | 1 | M | Flu | 000-00-001 |
| April | 30 | M | Cancer | 000-00-002 |
| June | 3 | F | Pneumonia | 000-00-003 |



- Assume:
  - Server sees which field queried
  - Records are identifiable between queries

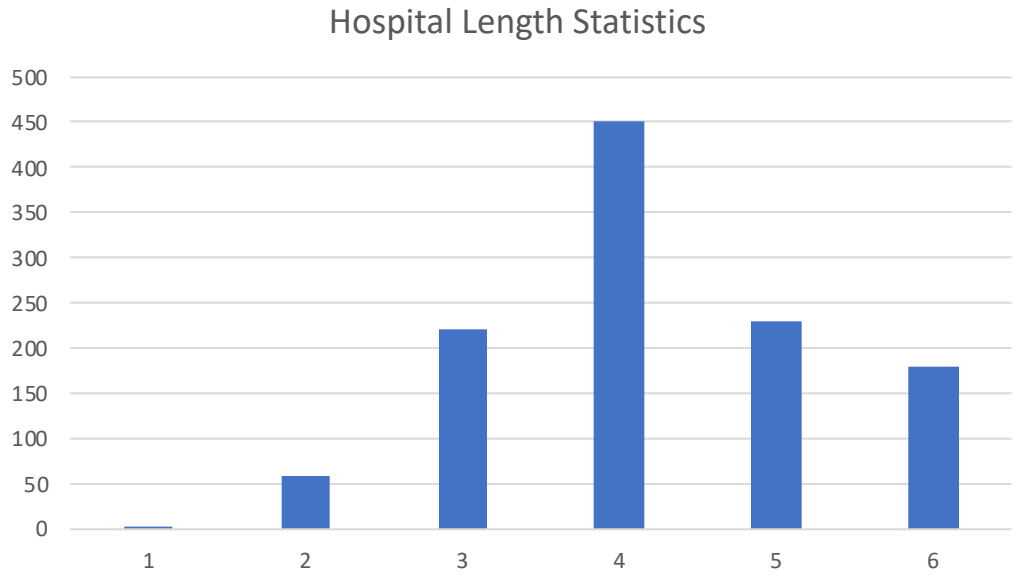
# Statistical Attack Against Hospital Length of Stay

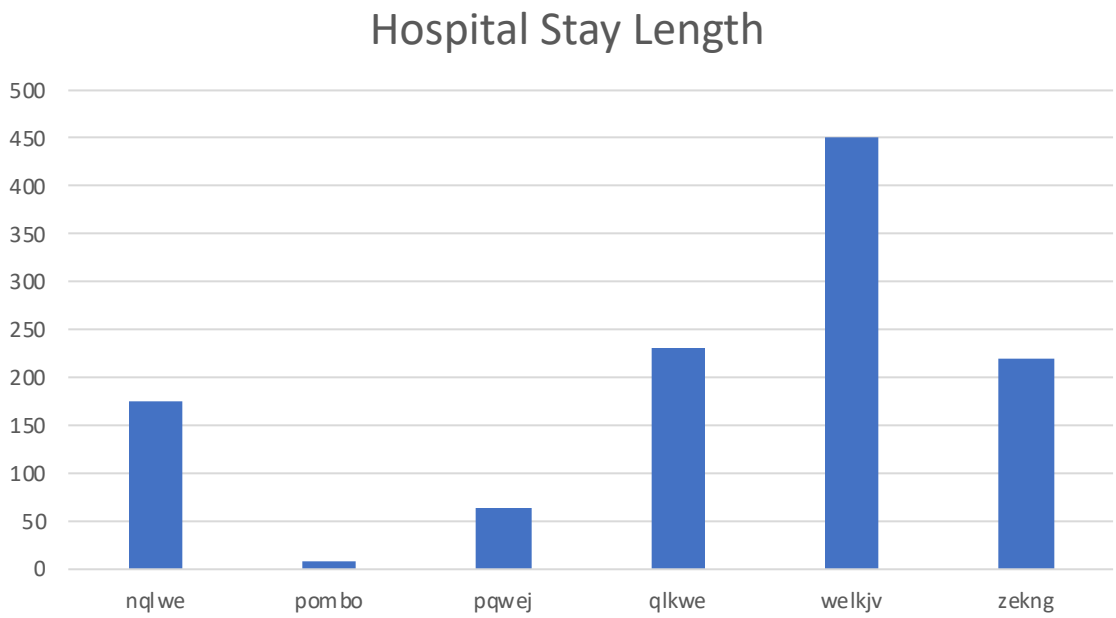
- Suppose:
  - Queries of form:
    SELECT * FROM table
    WHERE
      length_stay=XXXXX;
  - Observe |records|
  - Create unique id for query



# Statistical Attack Against Hospital Length of Stay

**Query with highest number of returned records likely represents 4 days**

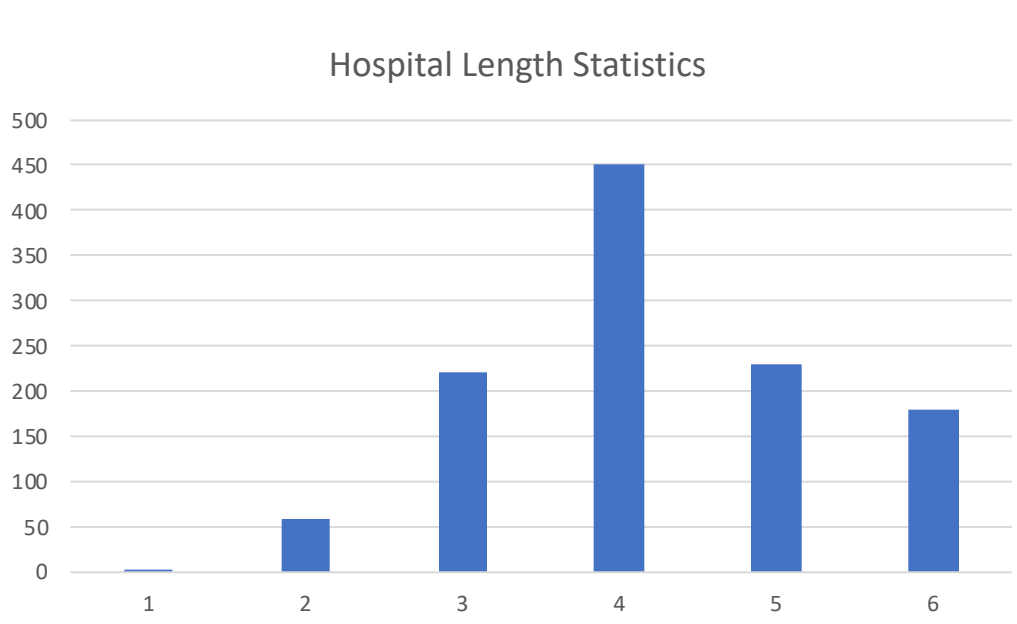


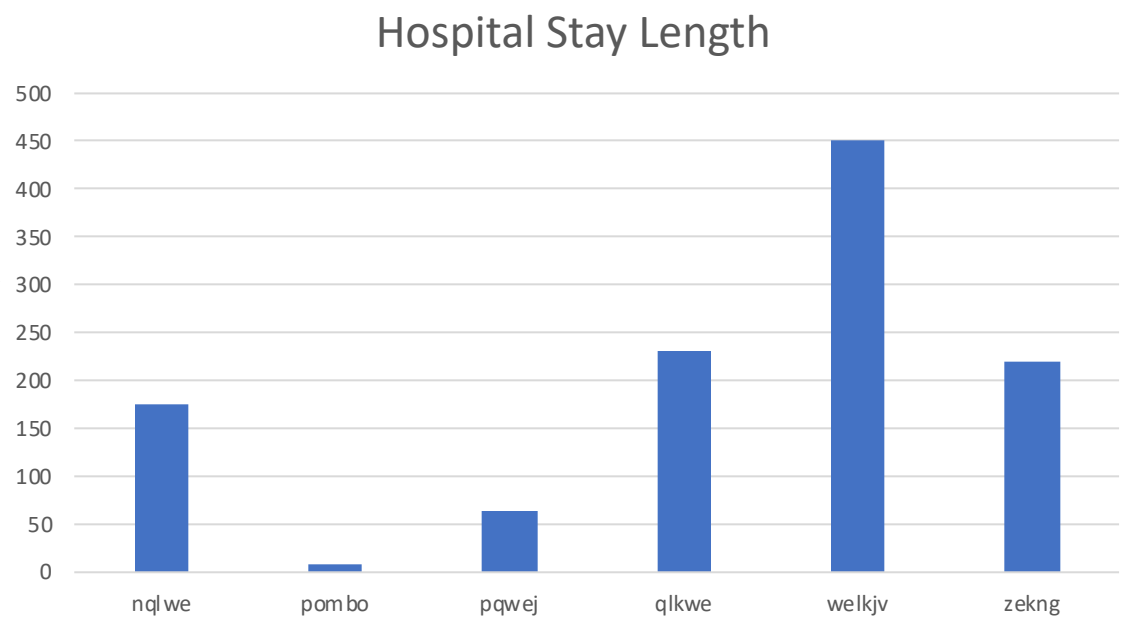




**Distribution of length of stay is known, attacker can use prior statistical information**

# Statistical Attack Against Hospital Length of Stay

**Query with highest number of returned records likely represents 4 days**

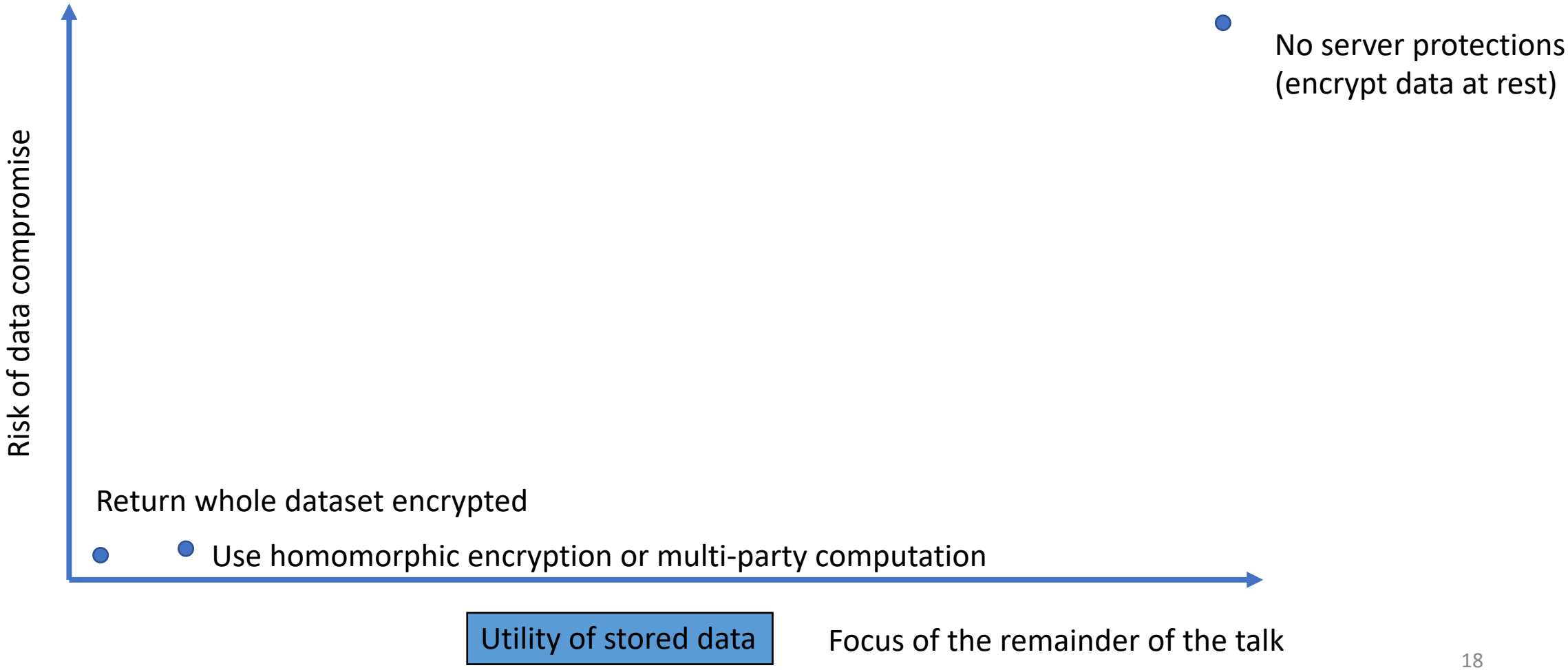



**What to do if number of records is not identifying enough?**
**Or statistical prior is inaccurate?**

**Attacks exploit correlation between fields, use techniques from optimization**

# Why systematize?



# Approaches to Protected Databases

Define five types of leakage of increasing impact[4]:

1. Structure
2. Identifiers
3. Predicates
4. Equality
5. Order

Distinguish between schemes that leak this information at *Initialization* and at *Query*

Find three approaches to protected databases:

*1. Legacy*:

- Leak at *Initialization*
- Inherit DB advances

2. Custom:

- Leak during *Query*

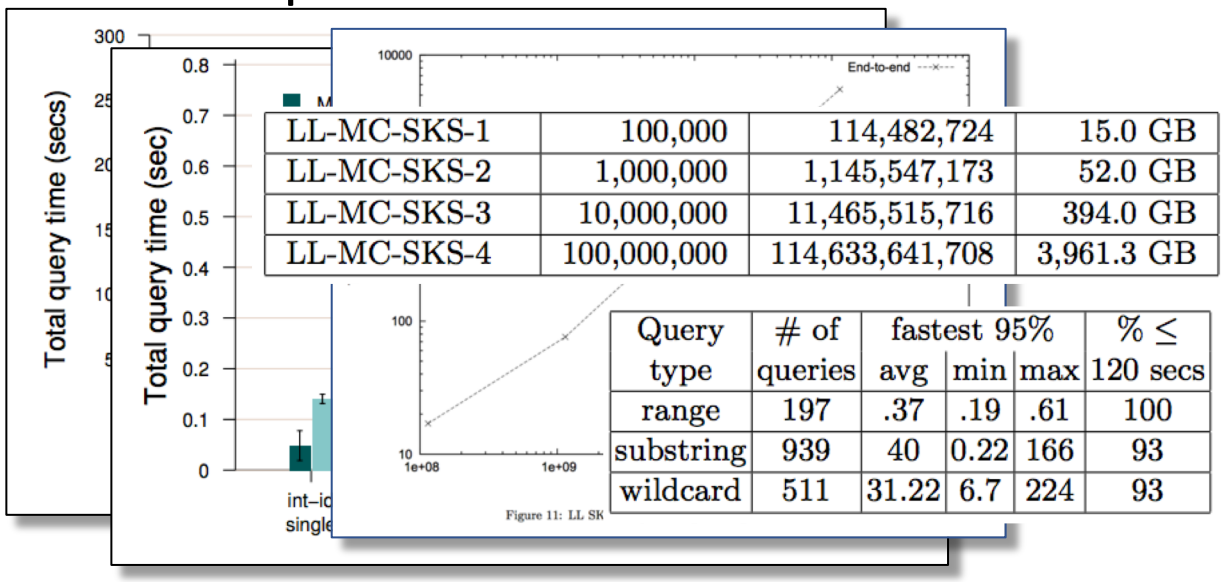
3. Obliv:

- Leak only structure
- Require multiple servers to be efficient

[4]Partially based on S.Kamara, "Structured encryption and leakage suppression," presented at Encryption for Secure Search and Other Algorithms, Bertinoro, Italy, June 2015.

# Approaches to Protected Databases

- Developed[9]:
  - a database instrumentation platform
  - data and query generator
- Used in prior work[10, 11]




| | | | |
|---|---|---|---|
| LL-MC-SKS-1 | 100,000 | 114,482,724 | 15.0 GB |
| LL-MC-SKS-2 | 1,000,000 | 1,145,547,173 | 52.0 GB |
| LL-MC-SKS-3 | 10,000,000 | 11,465,515,716 | 394.0 GB |
| LL-MC-SKS-4 | 100,000,000 | 114,633,641,708 | 3,961.3 GB |

| Query type | # of queries | fastest 95% avg | min | max | % ≤ 120 secs |
|---|---|---|---|---|---|
| range | 197 | .37 | .19 | .61 | 100 |
| substring | 939 | 40 | 0.22 | 166 | 93 |
| wildcard | 511 | 31.22 | 6.7 | 224 | 93 |

Figure 11: LL SK

[9]https://github.com/mit-ll/sparta
[10]V. Pappas et al. "Blind Seer: A Private Scalable DBMS," S&P 2014
[11]D. Cash et al. "Dynamic Searchable Encryption in Very-Large Databases: Data Structures and Implementation," NDSS 2014

Find three approaches to protected databases:

*1. Legacy*:
- Leak at *Initialization*
- Inherit DB advances

2. Custom:
- Leak during *Query*

3. Obliv:
- Leak only structure
- Require multiple servers to be efficient

# How to compare functionality?

- Natural approach: what fraction of a unprotected database language is supported?

- Current systems implement *base* queries using cryptography, extend from these base queries:
  - Keyword Equality
  - Range
  - Boolean Combination
  - Other (graph alg and substring)

Find three approaches to protected databases:

*1. Legacy*:
  - Leak at *Initialization*
  - Inherit DB advances

2. Custom:
  - Leak during *Query*

3. Obliv:
  - Leak only structure
  - Require multiple servers to be efficient

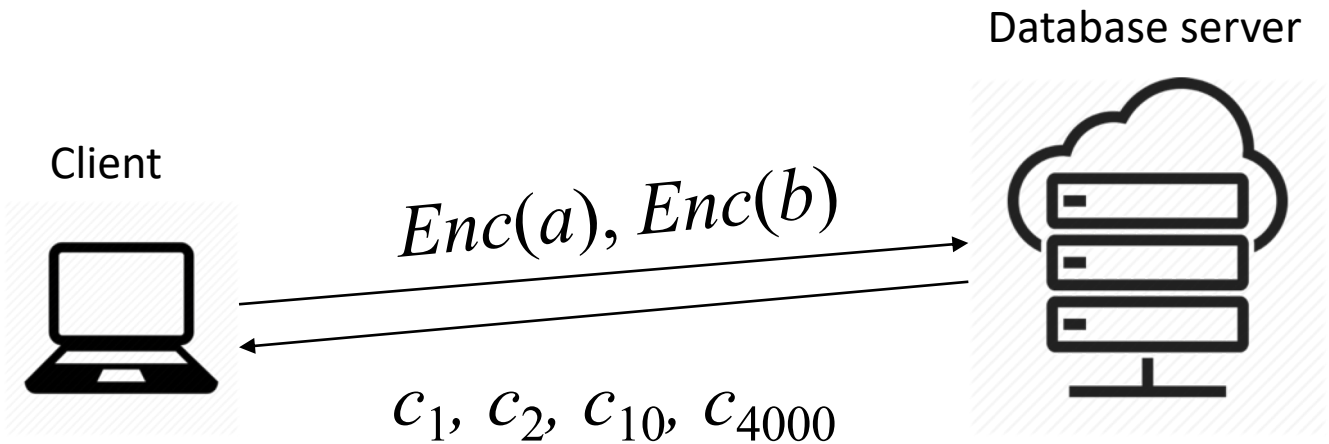
# Outline

- Overview of Protected Search
- Leakage Impacts
- Finding a basis for search results
  - Range queries
    - Order-Preserving Encryption
    - Partial Order-Preserving Encryption
    - SisoSPIR
  - Combining queries
- Extending to new database paradigms

# Order-Preserving Encryption

- Enc that preserves plaintext order:
  - If $m_1 < m_2$ then $Enc(m_1) < Enc(m_2)$

1. Encrypt query $Enc(a), Enc(b)$
2. Let server use standard search mechanism
3. Return encrypted records



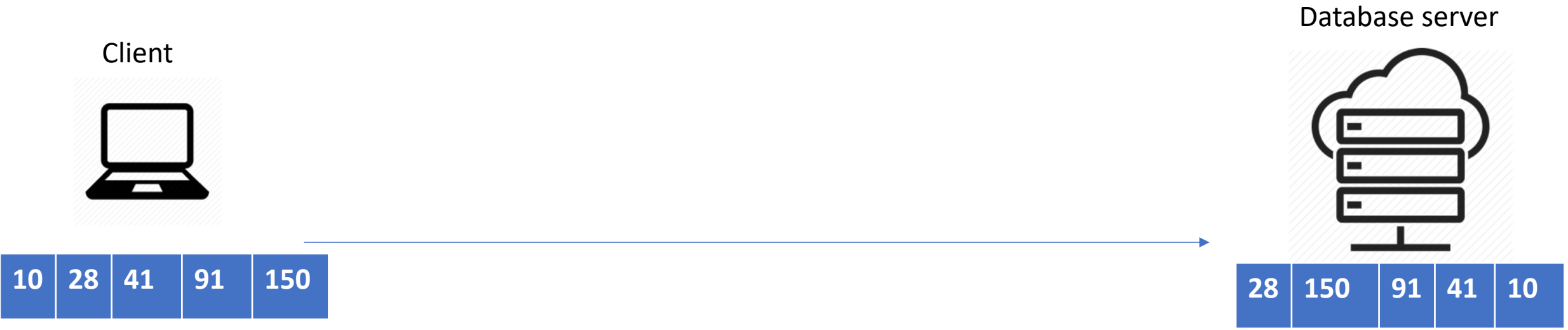

# Leakage Attacks of OPE

- Data is sorted, does not protect dense data
- Strongest leakage attack applies to OPE
- Technique used in many commercial product

| Attacker goal | Required $S$ leakage | | Required attack conditions | | Attack efficacy | | |
|---|---|---|---|---|---|---|---|
| | Init | Query | Ability to inject data | Prior knowledge | Runtime | Sensitivity to prior knowledge | Keyword universe tested |
| Query Recovery | ○ | ○ | — | ◔ | ● | ? | ○ |
| | ○ | ◔ | ✔ | ○ | ○ | ○ | ○ |
| | ○ | ◔ | — | ◔ | ● | ? | ○ |
| | ○ | ◔ | — | ◕ | ◐ | ● | ● |
| | ○ | ◔ | ✔ | ◕ | ◐ | ○ | ● |
| | ○ | ◔ | — | ● | ◐ | ● | ● |
| Data Recovery | ○ | ◔ | — | ◐ | ● | ● | ◐ |
| | ◕ | — | — | ◐ | ○ | ? | ○ |
| | ◕ | — | ✔ | ◐ | ○ | ? | ● |
| | ◕ | — | — | ◕ | ○ | ? | ● |
| | ● | — | — | ◐ | ○ | ○ | ● |

Row corresponding to OPE

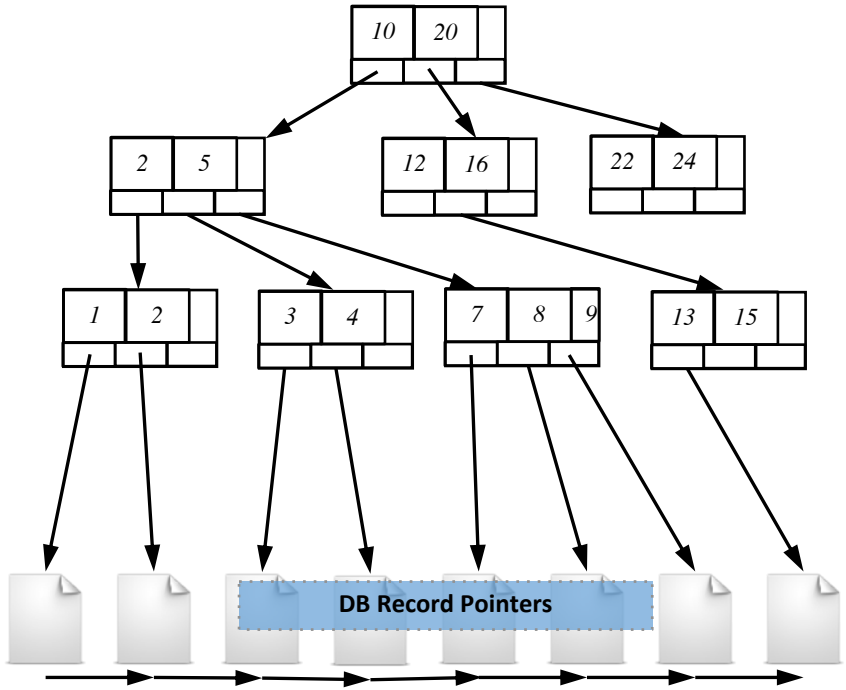
# Partial Order Preserving Encoding[13]




- Client sends data to server encrypted and unsorted
- Client and Server work together to create partially sorted tree
  - Client performs all comparisons
  - Server is able to build tree based on client comparisons
- Stronger security than Order-Preserving Encryption if tree is only partially built

[13]D. Roche, D. Apon, S. Choi, A. Yerukhimovich “POPE: Partial Order Preserving Encoding” CCS 2016

# SisoSPIR[14] – *Obliv* Approach to Range




B+ trees are used in many unprotected databases

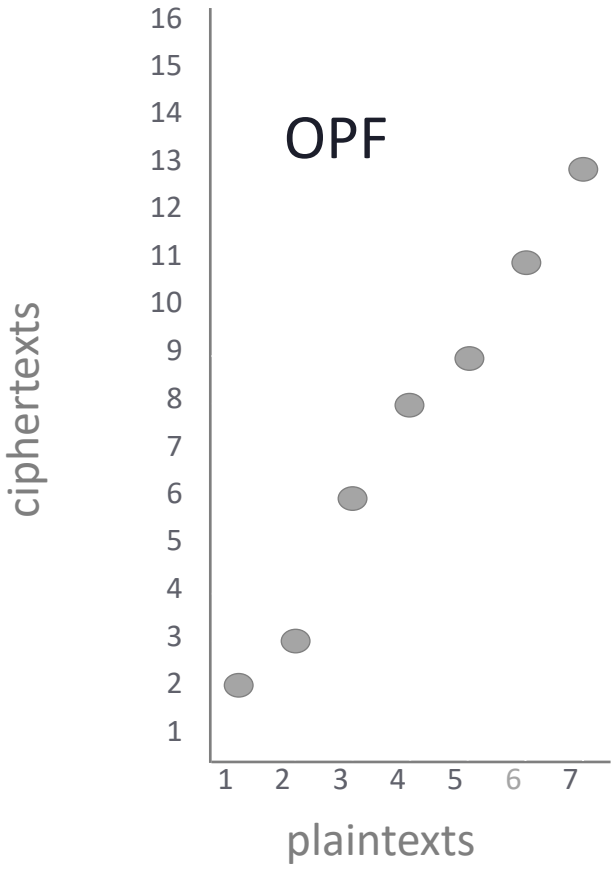
Variable number of children per node

Idea of approach: use crypto to hide all information in traversing B+ tree

**Requires multiple servers for practical efficiency**

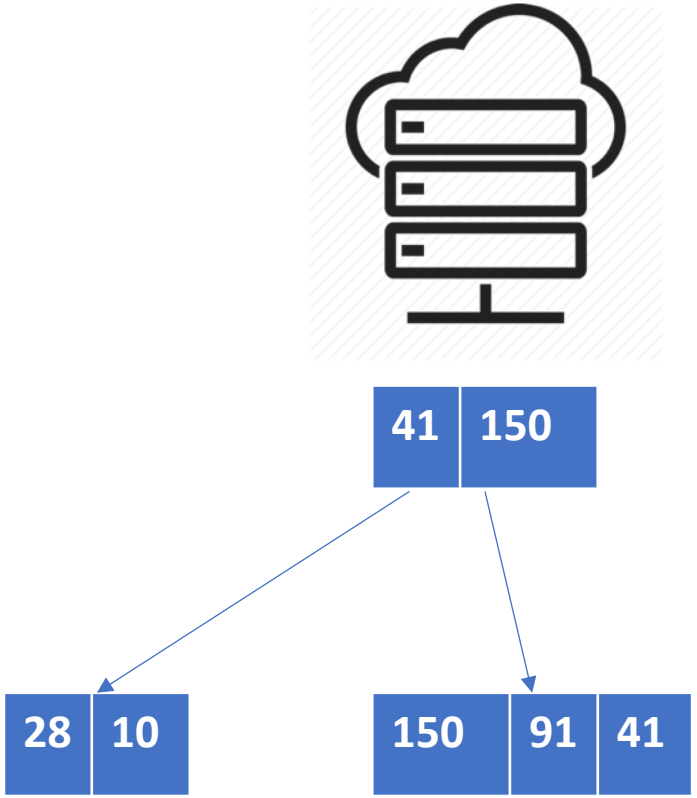
[14]Y. Ishai, E. Kushilevitz, S. Lu and R. Ostrovsky, "Private Large-Scale Databases with Distributed Searchable Symmetric Encryption," CT-RSA 2015

# Legacy



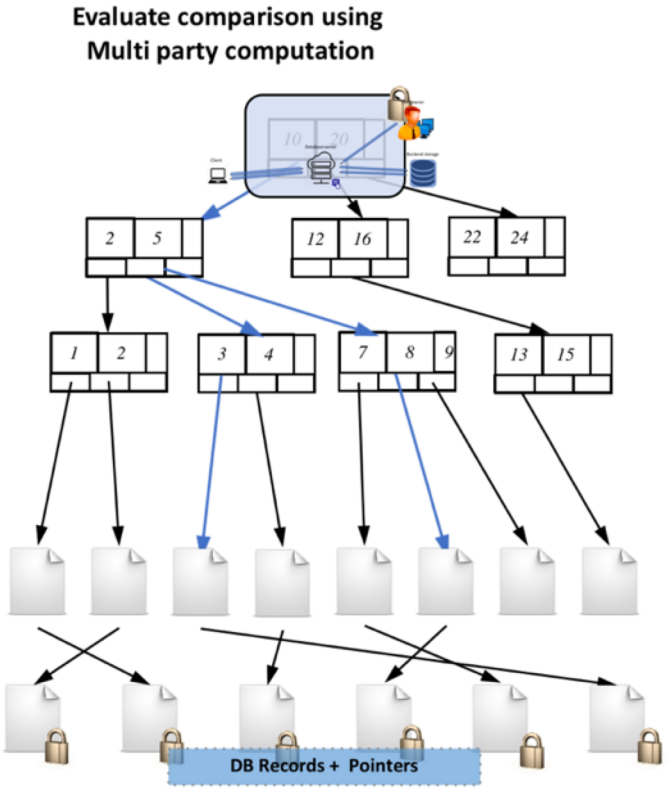

**Data is high entropy**

# Custom




**Small fraction of DB is returned by queries**

# Obliv




**Otherwise**

# Query Combination

- Techniques to *combine base* queries:
  - Range → Equality, search for [a, a]
  - Boolean → Range, using set covers
  - Range → Substring, by inserting each prefix

- Most combination techniques are less efficient and have more leakage than equivalent base query

- Allow for rapid expansion of query functionality

# Approaches to Protected Databases

- Natural approach: what fraction of a unprotected database language is supported?

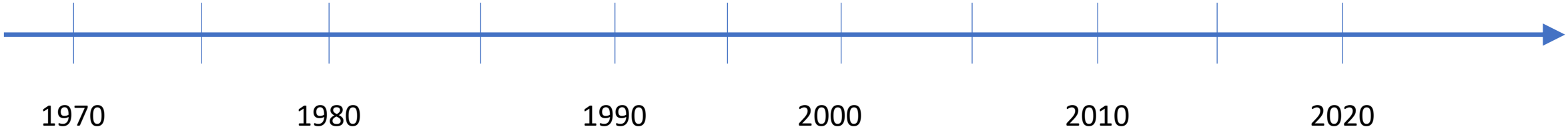
- Systems implement *base* queries w/ crypto, extend from these base queries:
  - Keyword Equality
  - Range
  - Boolean Combination
  - Other (graph alg and substring)

SQL has a well defined mathematical set-theory basis of operations[14]:

- Union: AUB
- Difference: A \ B
- Join: A x B
- Projection: Take some dimensions of results
- Selection: Take rows satisfying some condition

[14]E. Codd, "A relational model of data for large shared data banks," *Communications of the ACM*, 1970

# Unprotected DB Development




Introduction of relational model by Codd

MySQL, PostGRES, Oracle

Crypto community starts working on protected search

Google BigTable NoSQL Key-Value

**It took 20 years to secure SQL**

**How can we catch up?**

NoSQL Graph DBs

SciDB NewSQL

Polystore

# Keeping up with database diversification

Common unprotected databases have a mathematical basis of operations:

- For SQL: Union, Difference, Join, Projection, Selection
- For Array-Store:
  Construct,
  Find,
  Array (+, x),
  Element-wise x
- For Graph:
  Linear algebra over matrices

Cryptographers and DB designers should work together to:

1. Identify base queries that are likely to be useful across DB paradigms
2. Understand critical functions of emerging databases
3. Quickly fill gaps using *combiners*

Questions?
https://arxiv.org/abs/1703.02014

# Backups

| DB Paradigm | Basis Operation | Crypto Base Operation? |
|---|---|---|
| NoSQL – Key Value Store | Construct | Yes |
| | Find | Yes – Mature range search with variety of techniques |
| | Array (+, x) | Some – Addition possible using partially homomorphic techniques |
| | Element Wise x | Some – Using partially homomorphic techniques |

**Main gap is support for very high insert rates above 1M records per second**

| DB Paradigm | Basis Operation | Crypto Base Operation? |
|---|---|---|
| Graph Databases– Linear Algebra | Construct | Yes |
| | Find | Yes – Mature range search with variety of techniques |
| | Matrix (+, x) | Some – Have private algorithms for matrix mult./add. |
| | Element Wise x | Some – Using homomorphic operations |

**Current matrix operations operate on full structure, need algorithms for sparse matrices (most graph algorithms)**

# Current systems

Questions?
https://arxiv.org/abs/1703.02014

- Currently mature systems with peer-reviewed descriptions
- All systems use the basis and combination approach to get rich functionality

| | Supported Operations | | | | | | | | | Properties | | | Features | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| System | Equality | Boolean | Keyword | Range | Substring | Wildcard | Sum | Join | Update | Approach | # of parties | Code available | Multi-client | User auth. | Access control | Query policy | Leakage | Performance |
| CryptDB [15] | ● | ● | ○ | ● | ○ | ○ | ● | ● | ● | Legacy | 2 | ● | ● | ● | ● | ○ | ◕ | ● |
| Arx [14] | ● | ○ | ○ | ● | ○ | ○ | ● | ● | ● | Custom | 2 | ○ | ○ | ○ | ○ | ○ | ◐ | ◐ |
| BLIND SEER [16], [17] | ● | ● | ● | ● | ○ | ○ | ○ | ○ | ● | Custom | 3 | ○ | ● | ○ | ○ | ● | ◐ | ◐ |
| OSPIR-OXT [18]–[21], [103], [104] | ● | ● | ● | ● | ● | ● | ○ | ○ | ● | Custom | 3 | ○ | ● | ○ | ○ | ● | ◐ | ◕ |
| SisoSPIR [22] | ● | ○ | ● | ● | ● | ○ | ○ | ○ | ○ | Obliv | 3 | ○ | ○ | ○ | ○ | ● | ○ | ◐ |