Course Announcements

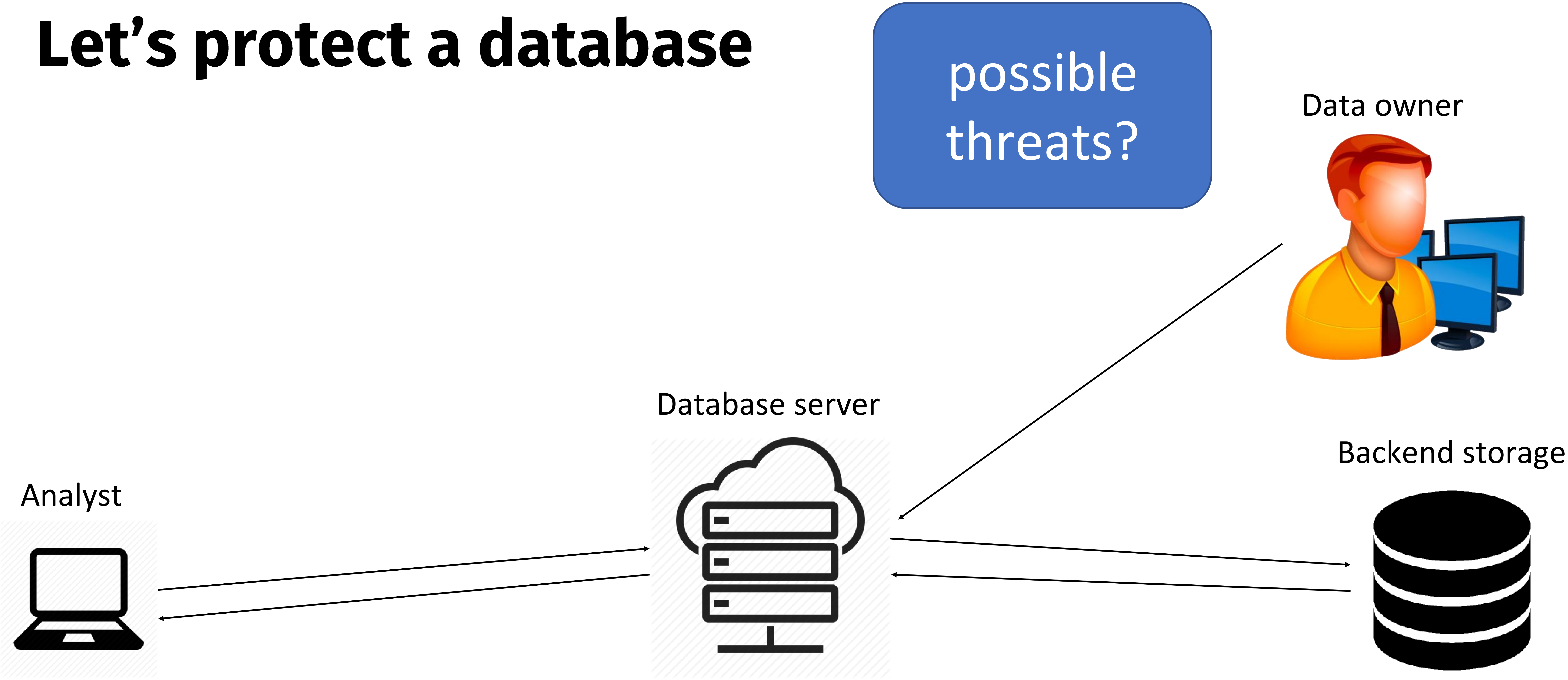
- Project
  - Project due Wednesday 4/22
  - Send a private Piazza post to the TA/grader overseeing your project
- Assignments
  - Reading: End-to-End Verifiability

Lecture 21: Protecting Databases and Elections

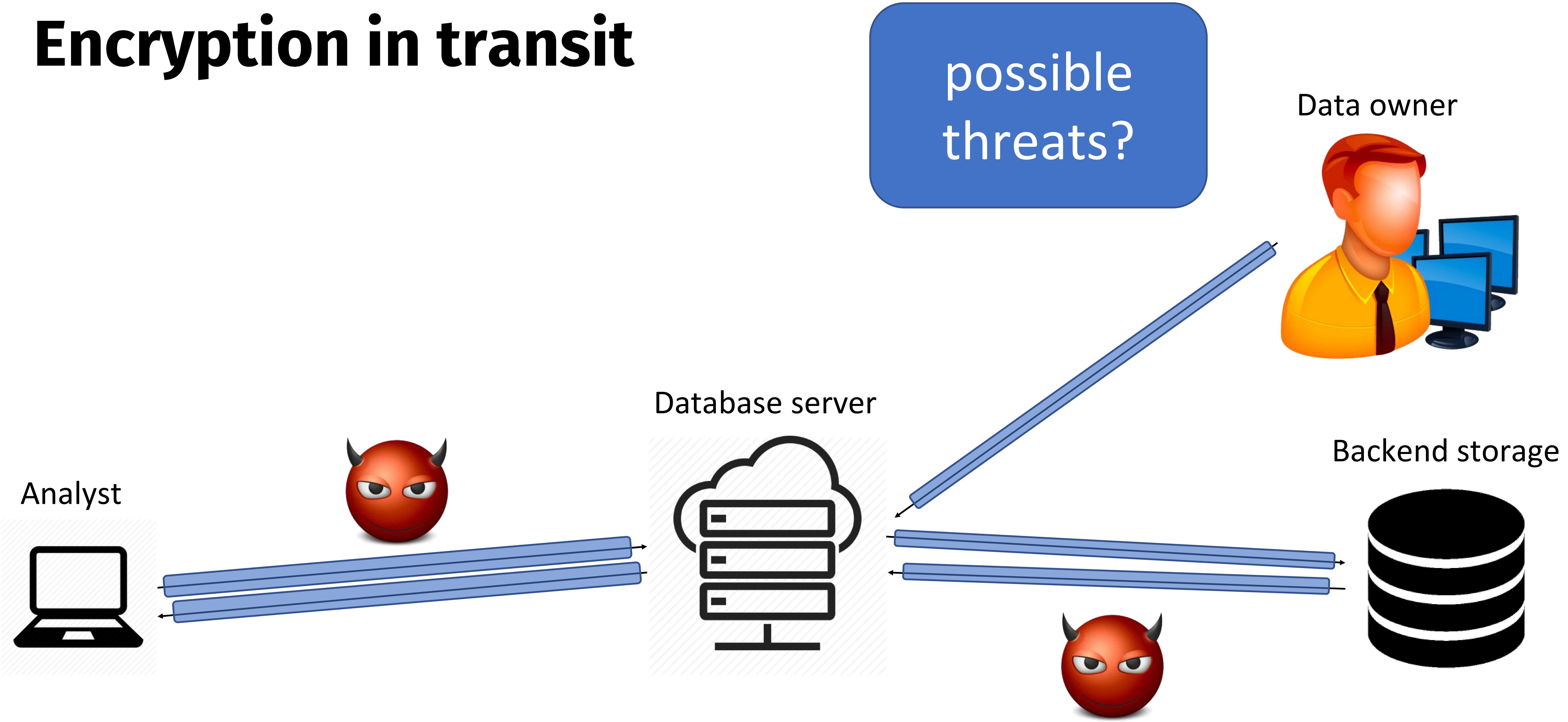
1. Protected database search
2. End-to-end verifiable elections

1. Protected database search

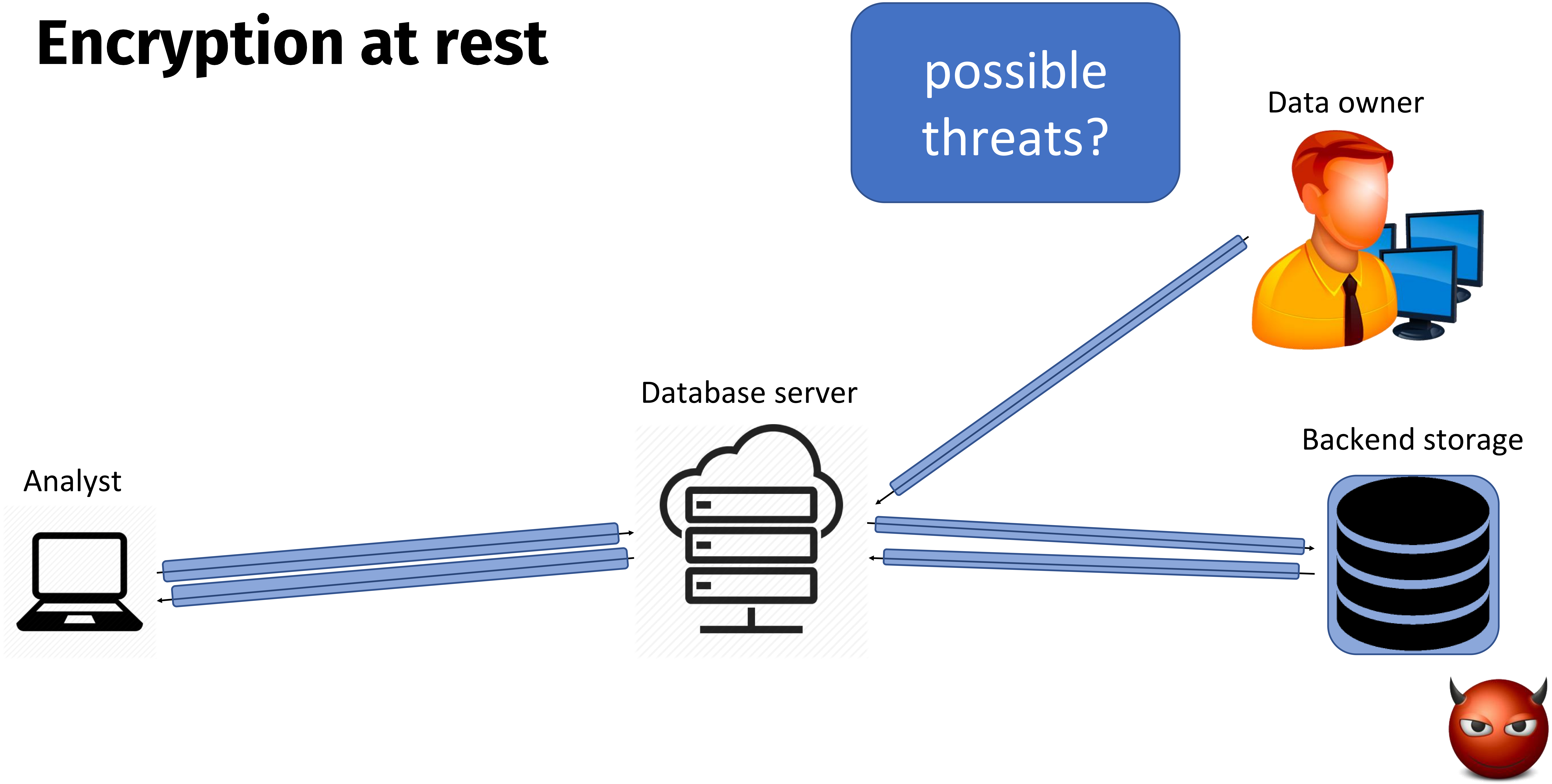
Let’s protect a database



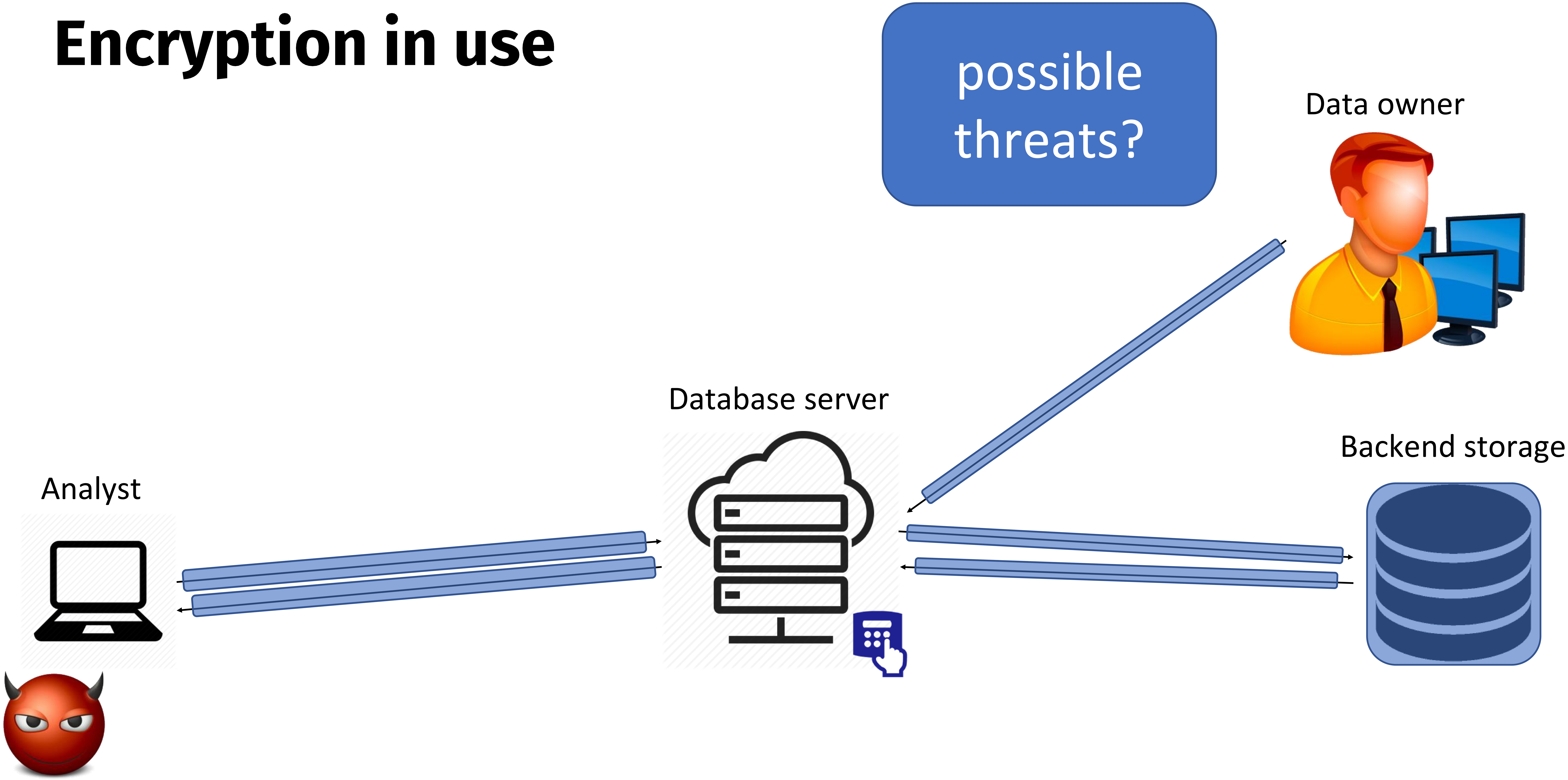
Encryption in transit



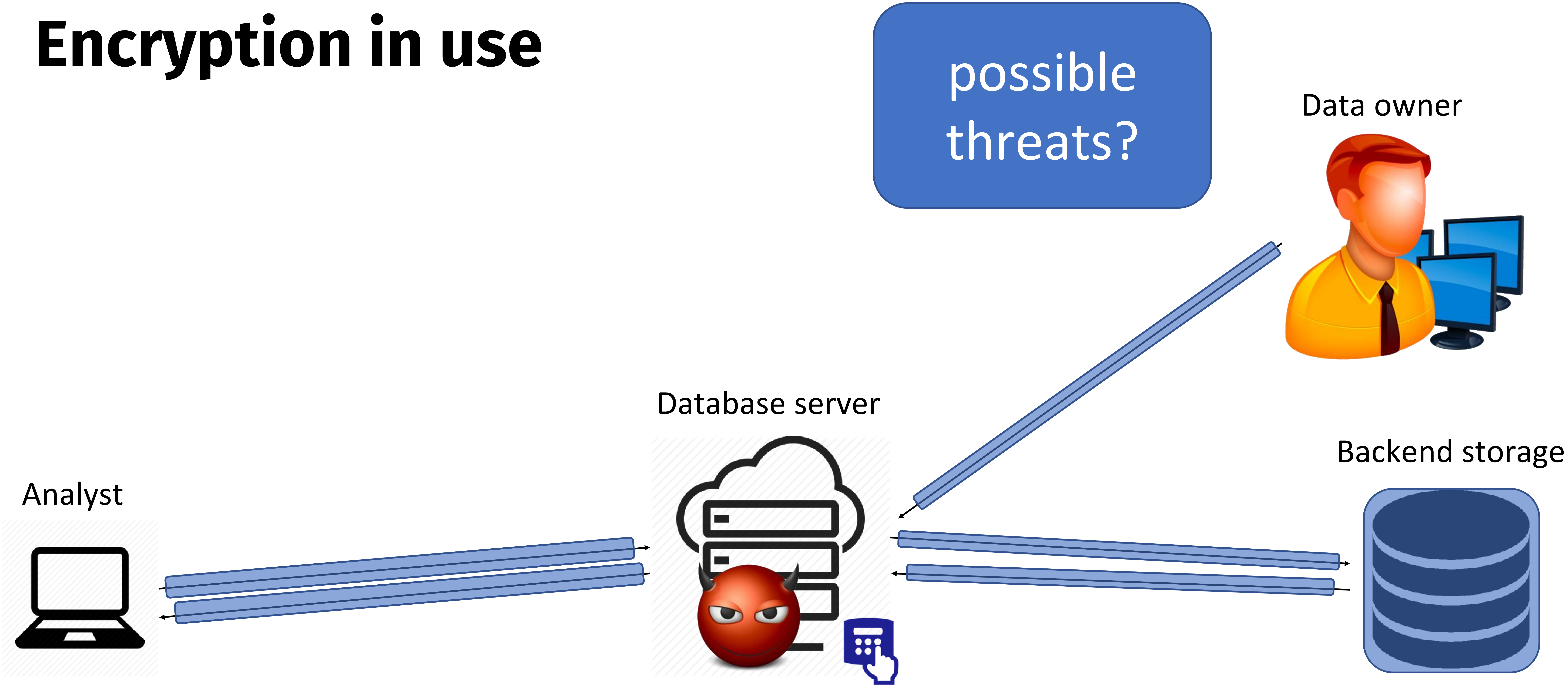
Encryption at rest



Encryption in use



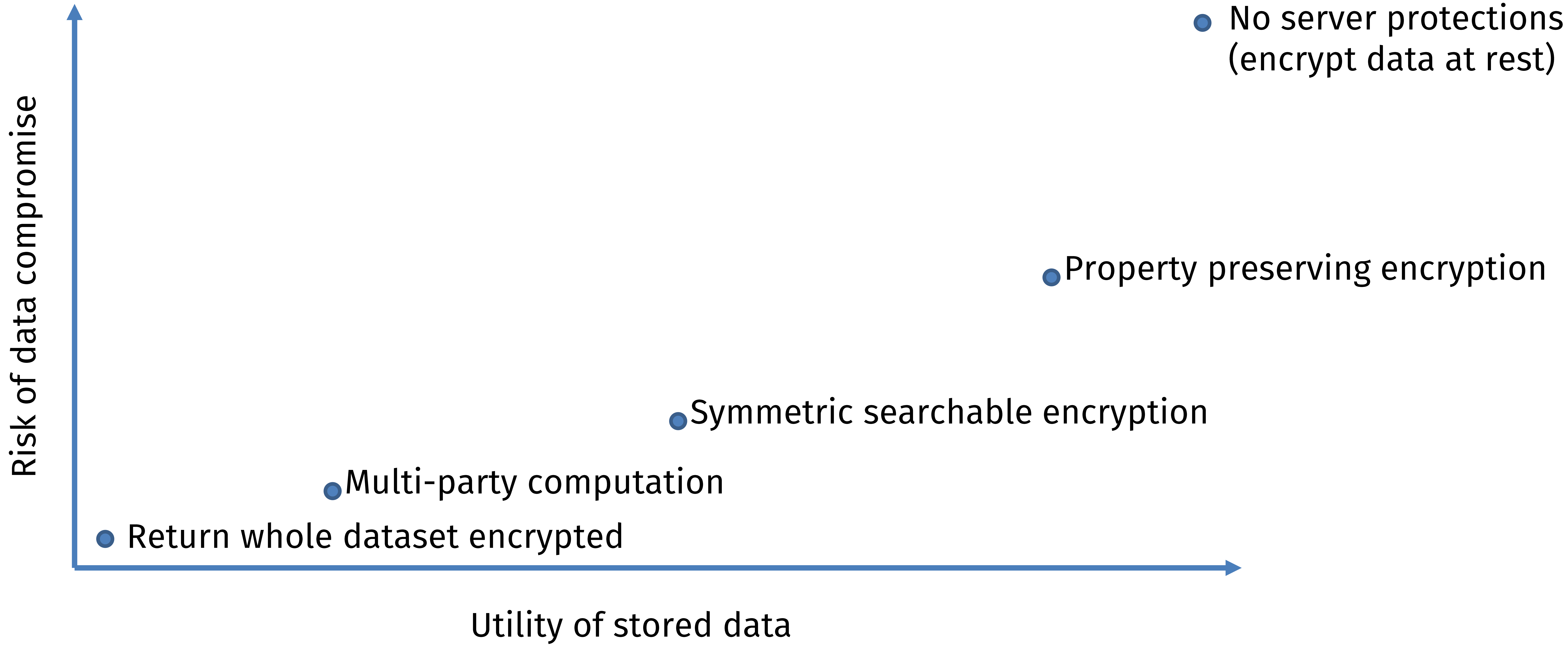
Encryption in use



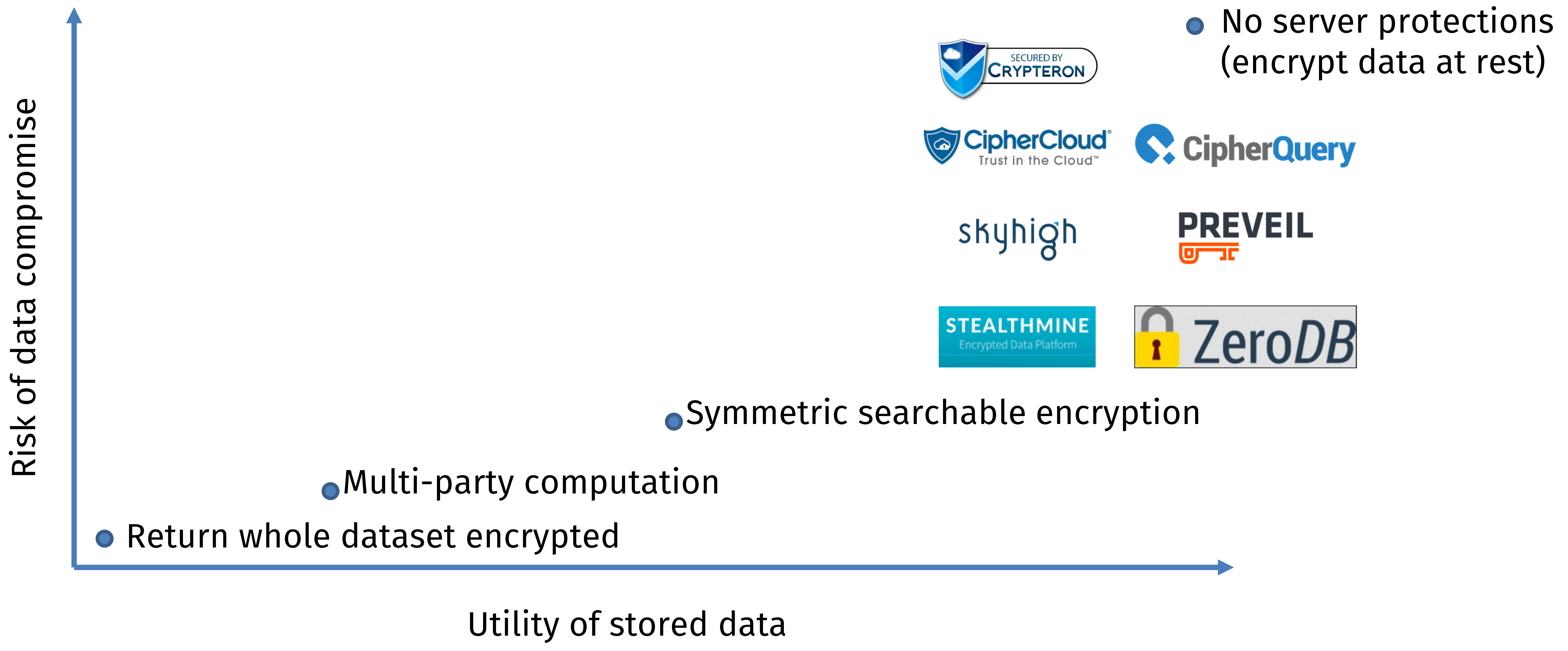
Desired goal: “encrypted indexes” that permit the server to search directly over encrypted records

- Server shouldn’t see either data or queries
- Server might observe access patterns though

Cryptographically protected database search

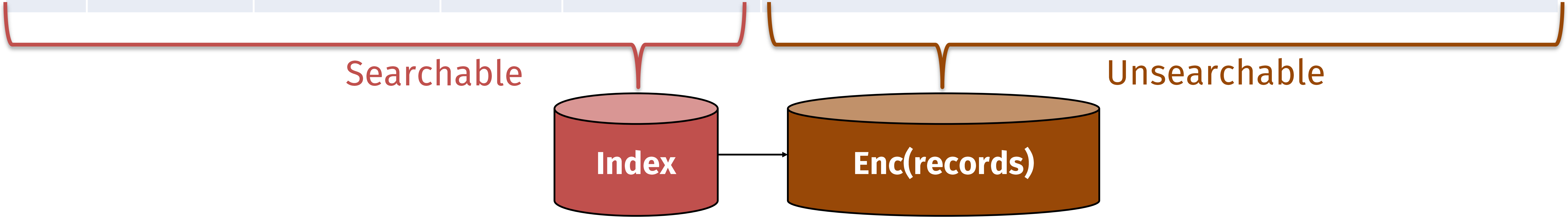


State of the art



Abstract view of a single-table database

id	fname	lname	Age	Income	Photo
1	Alice	Jones	20	71,000	<alice.jpg>
2	Bob	Jones	25	58,000	<bob.jpg>
3	Charlie	Smith	50	62,000	<charlie.jpg>
4	David	Williams	55	75,000	<david.jpg>



Small data structure: map searchable terms to associated record ids

Large file store: standard authenticated encryption applied to each record

1. Property Preserving Encryption (PPE)

- Apply transformation that preserves relevant features
- Insert into a legacy database for indexing & searching

id	fname	lname	Age	Income
1	Alice	Jones	20	71,000
2	Bob	Jones	25	58,000
3	Charlie	Smith	50	62,000
4	David	Williams	55	75,000

id	fname	lname	Age	Income
1	qlap1	Lf4Pz	cnr	g^{71} r^{90}
2	7fBwo	Lf4Pz	duo	g^{58} r^{84}
3	AKx0k	sw2AD	syv	g^{62} r^{22}
4	CK6ZD	6lVTH	tng	g^{75} r^{38}

Operation:	**DET (=)**	**OPE (<)**	**HOM (+, x)**
Method:	Choose Enc function at random	Choose random *monotonic* function	Public-key crypto
Drawback:	Cloud sees equality patterns	Cloud sees < and ~distances	Slow

1. Property Preserving Encryption (PPE)

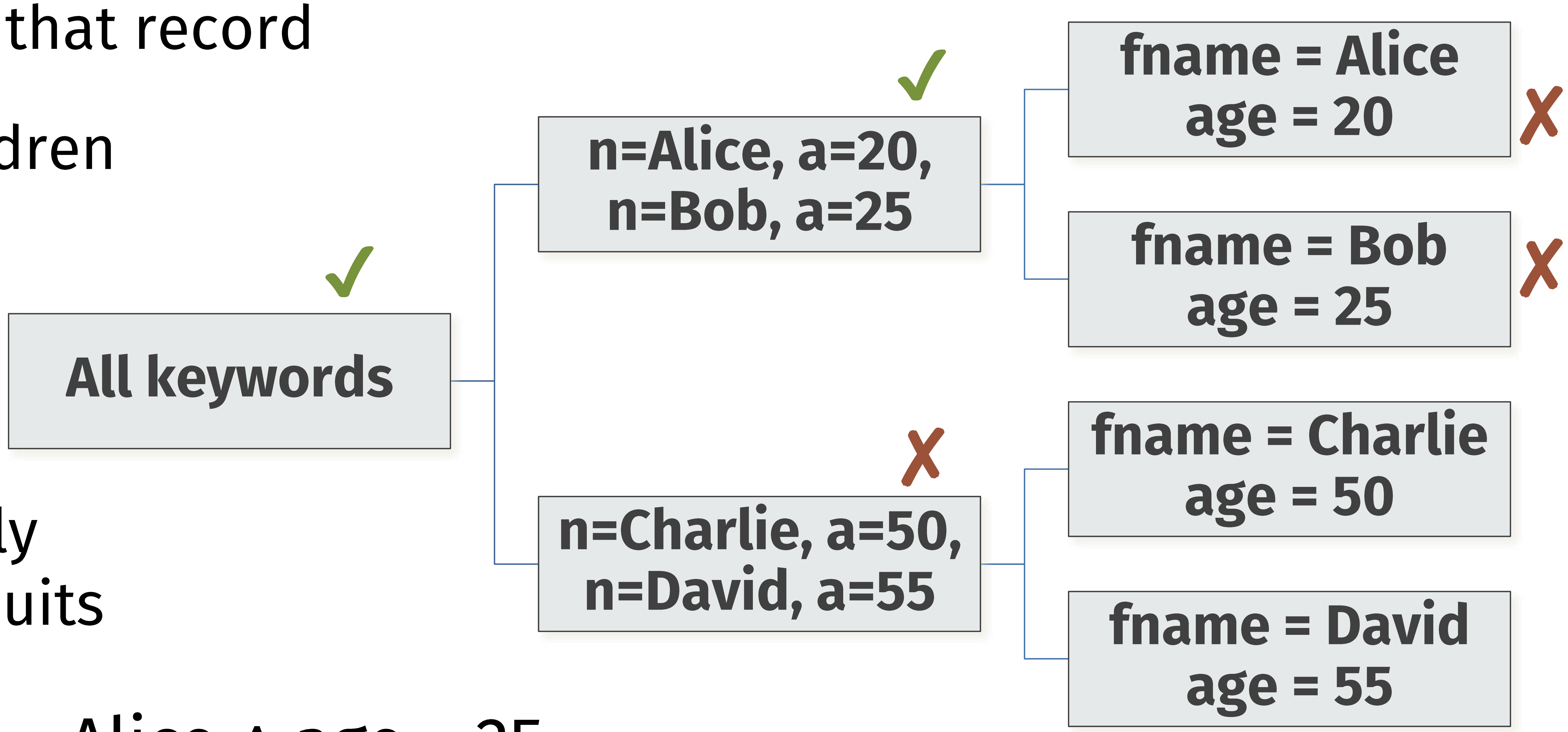
- Fast & legacy compliant
- Supported by a database near you!
  - Google: Encrypted BigQuery
  - Microsoft: SQL Server 2016, Azure SQL Database
  - Startups: Bitglass, Ciphercloud, CipherQuery, Crypteron, IQrypt, Kryptonostic, PreVeil, Skyhigh, ZeroDB, …
- Weakness: even though data isn’t stored in the clear, the revealed information is strong enough to reconstruct data and queries

2. Searchable Symmetric Encryption (SSE)

- Privacy: reveals or “leaks” less information to the database server
- Query expressivity: large subset of SQL
- Scale: tested on databases with 100m records
- Performance: ~3-5x of MySQL

SSE example (Blind Seer)

- Consider a tree in which each node stores a set
  - Leaves: set of keywords in that record
  - Other nodes: union of children
- Roles
  - Data owner makes tree
  - Cloud server & client jointly traverse using garbled circuits



- Consider the query name = Alice ∧ age = 25
- Imperfect security: tree search pattern reveals info about data

SSE example (Blind Seer)

- Main cryptographic innovation: represent set as *encrypted* Bloom filter
- Evaluate each node of the tree using secure two-party computation

n=Alice, a=20,
n=Bob, a=25

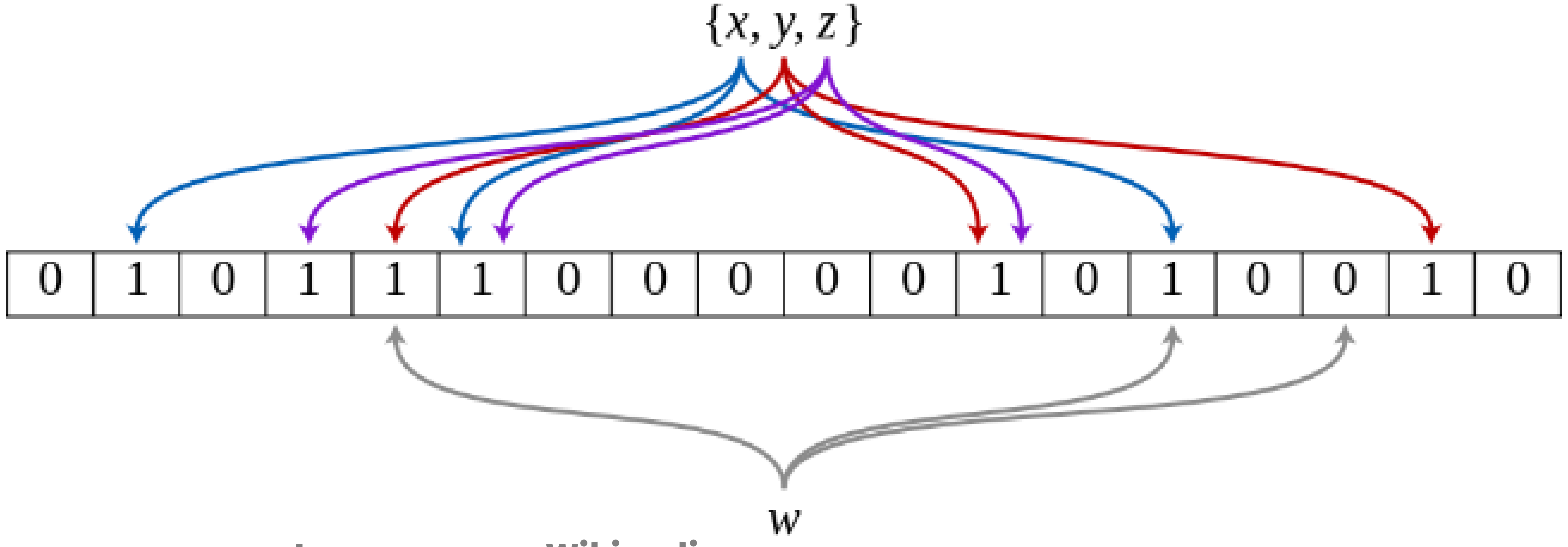


Image source: Wikipedia

Information revealed by SSE

- Protected search schemes reveal or leak some information about the query, data set, and result set to each party.
    1. Structure: size of an object, e.g. length of a string or cardinality of a set
    2. Identifiers: pointers to objects that persist across multiple accesses
    3. Equality or Order of values
- Some schemes leak:
    1. At Initialization on entire DB
    2. At Query on relevant records
- (More details in last week's reading assignment)

2. End-to-end verifiable elections

Slides produced by Ben Adida, and available at
http://assets.adida.net/presentations/ucl-voting-2009-02-03.pdf
http://assets.adida.net/presentations/2010-03-19-truly-verifiable-voting.pdf