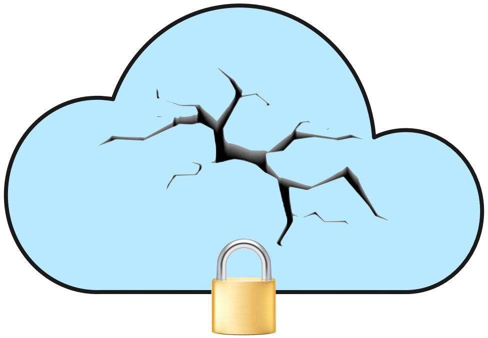


Securing data in compromised clouds

Raluca Ada Popa

UC Berkeley

riselab UC Berkeley

PREVEIL

raluca.popa@berkeley.edu

@ralucaadapopa

Massive cloud attacks are relentless

Yahoo 2014: Equifax 2017: Capital One 2019:

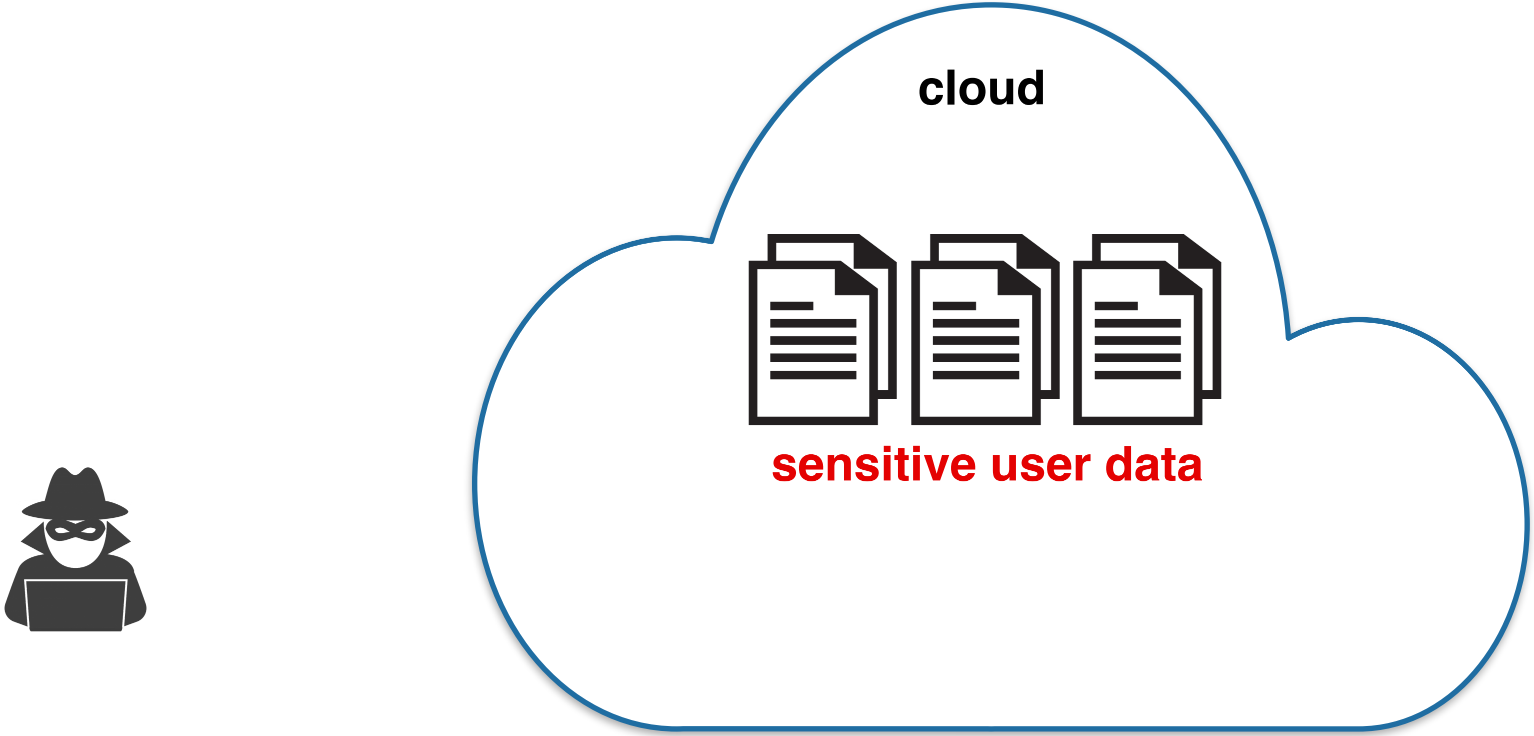
user records breached

Massive cloud attacks are relentless

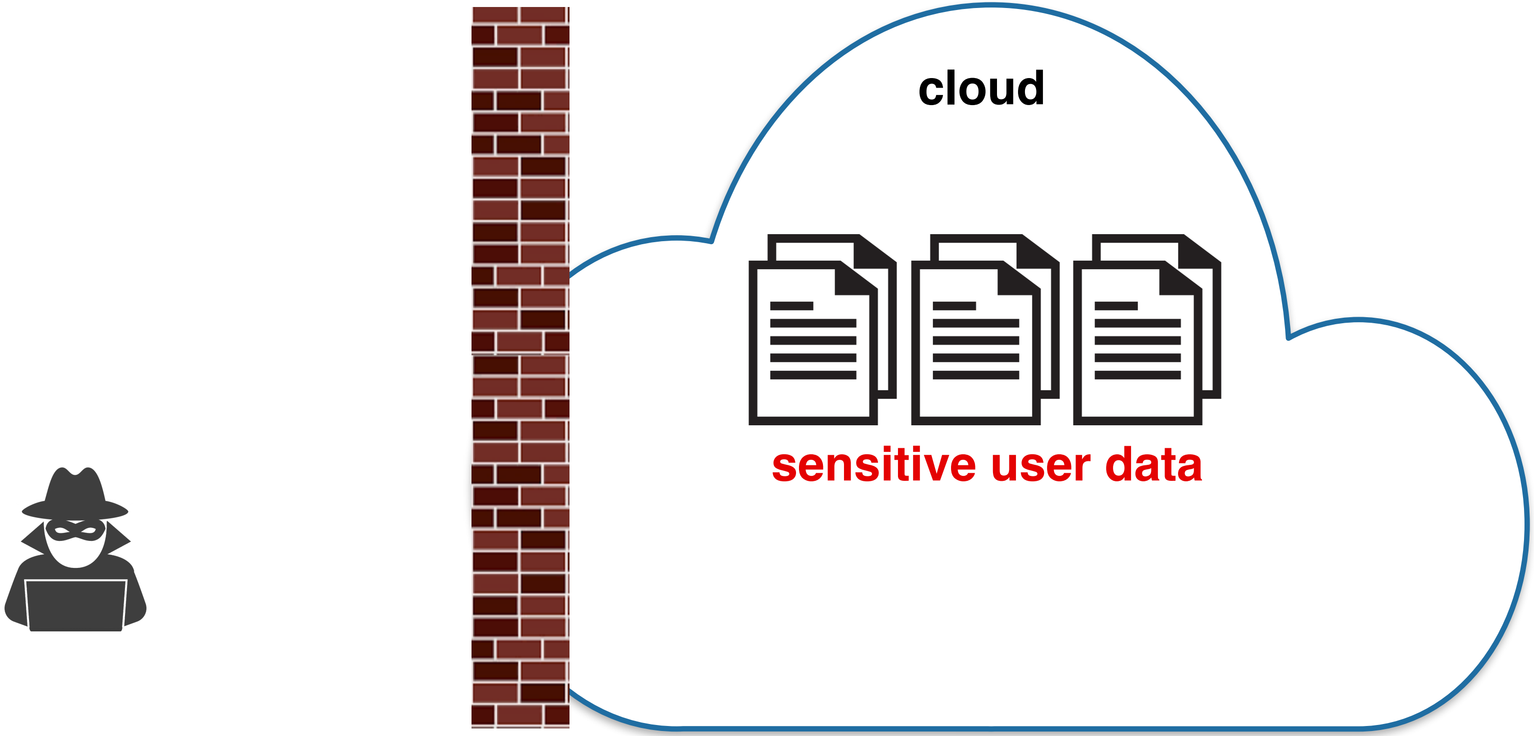
Yahoo 2014:	Equifax 2017:	Capital One 2019:
500,000,000	147,000,000	100,000,000

user records breached

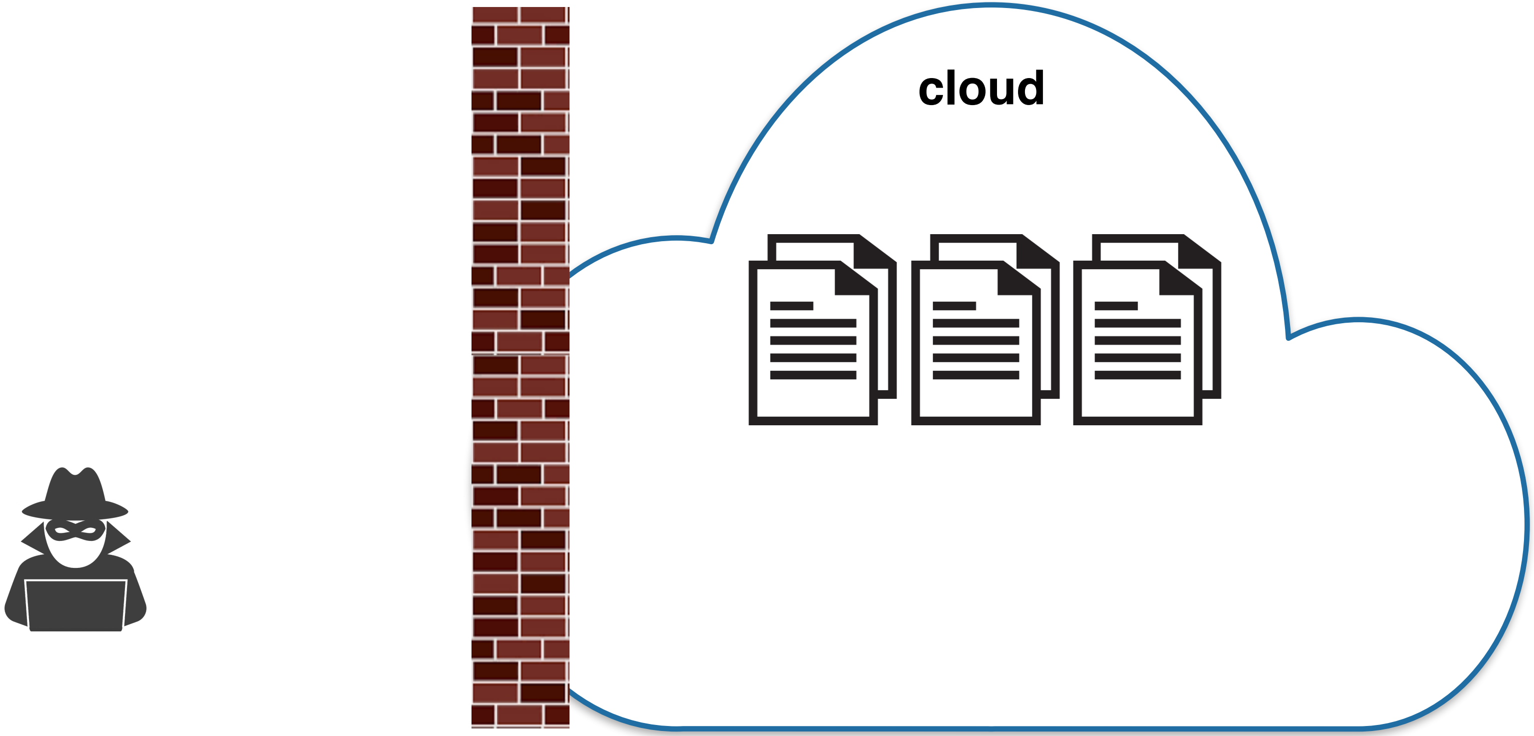
Traditional security has a fundamental weakness



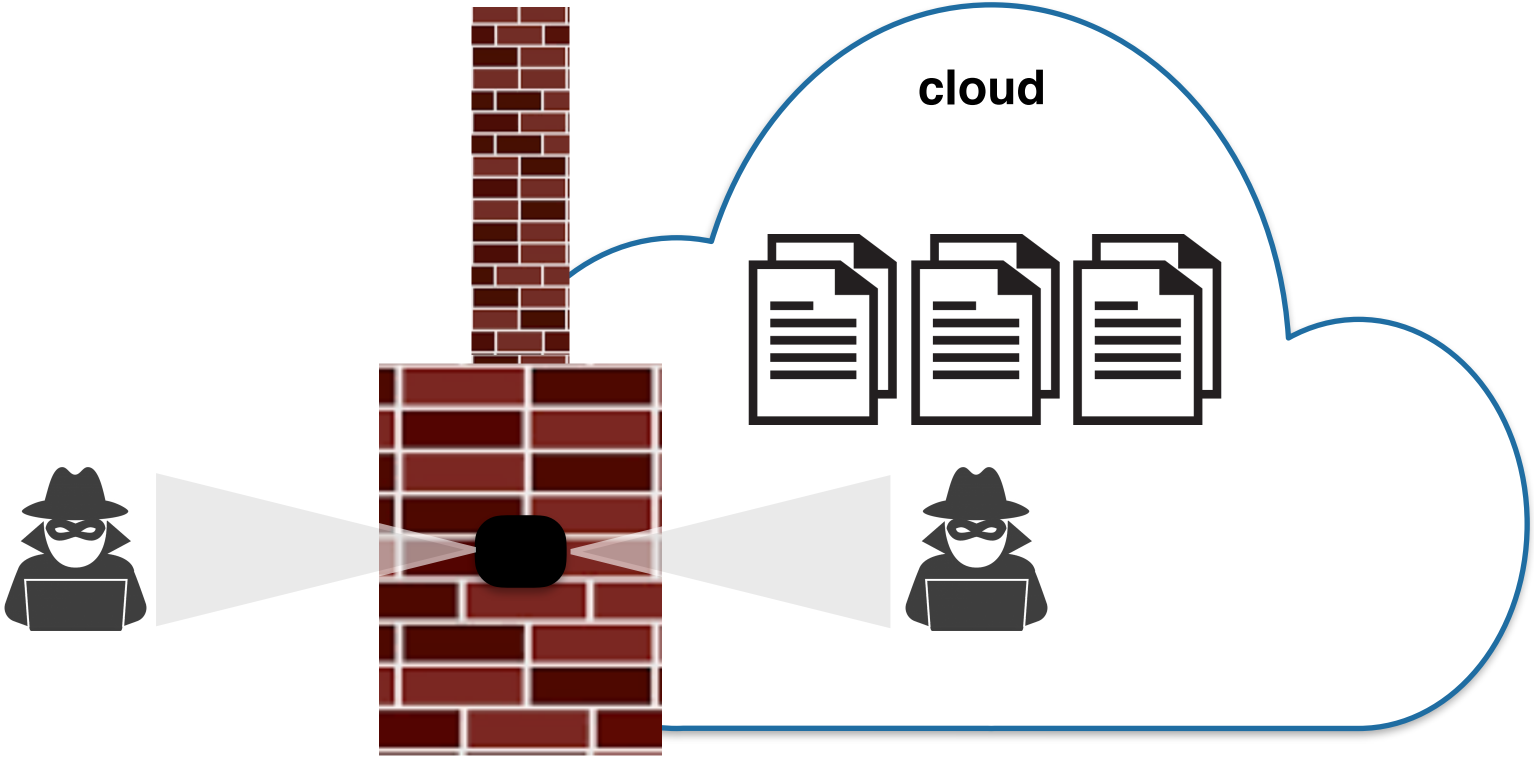
Traditional security has a fundamental weakness



Attackers eventually break in



Attackers eventually break in

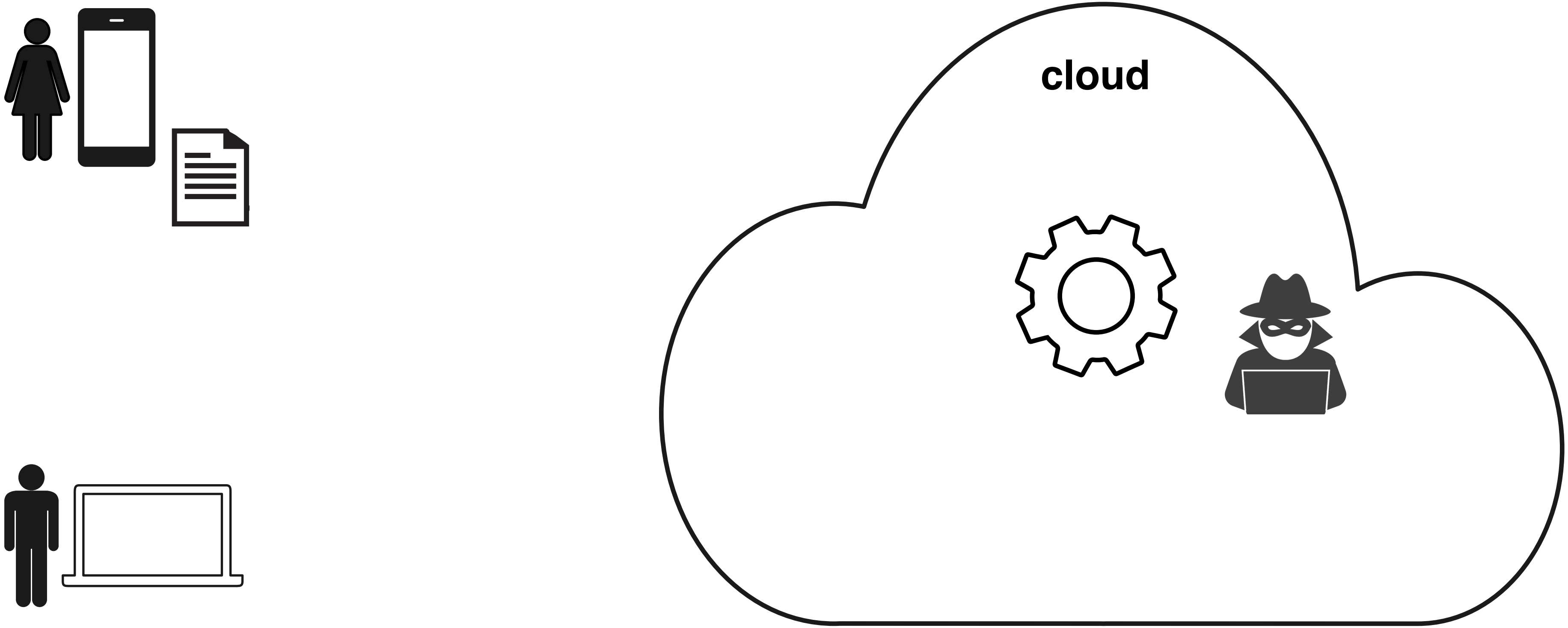


Assume the attacker will break in

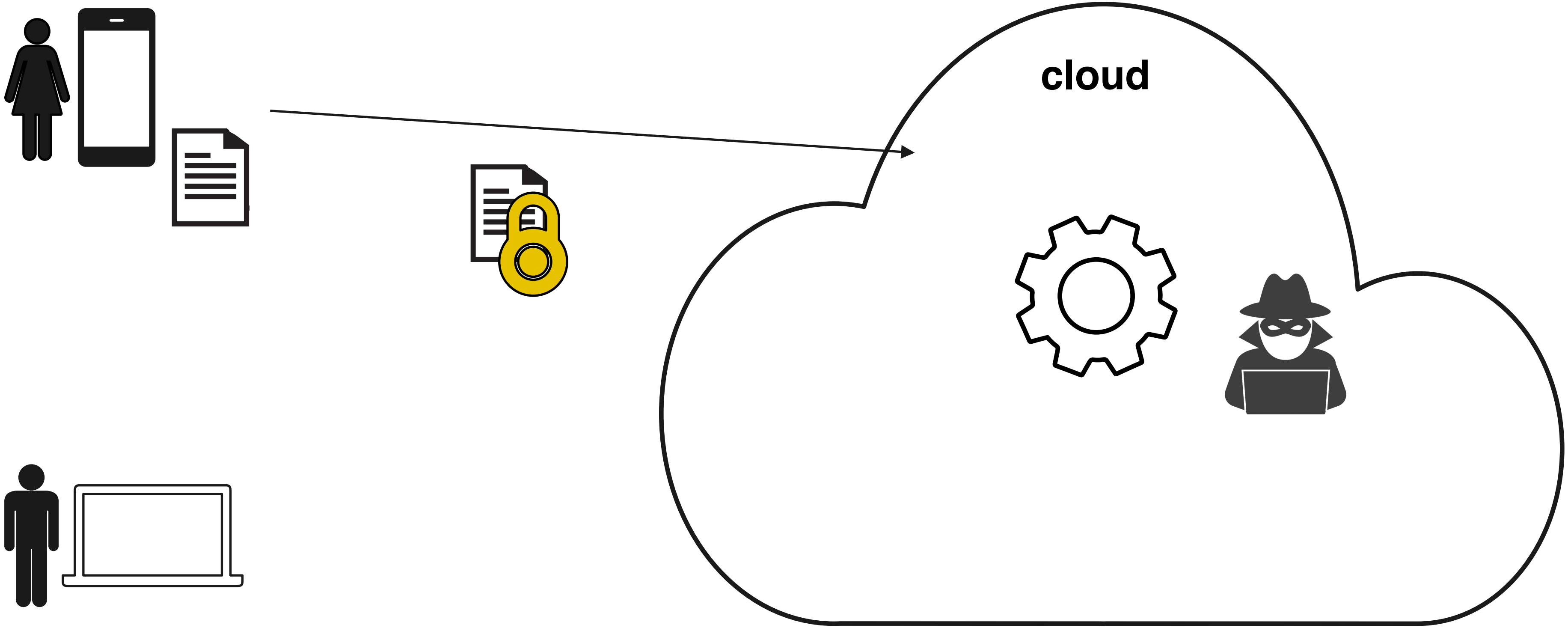
“in the cloud […] applications need to protect themselves instead of relying on firewall-like techniques”

Werner Vogels,
Amazon CTO

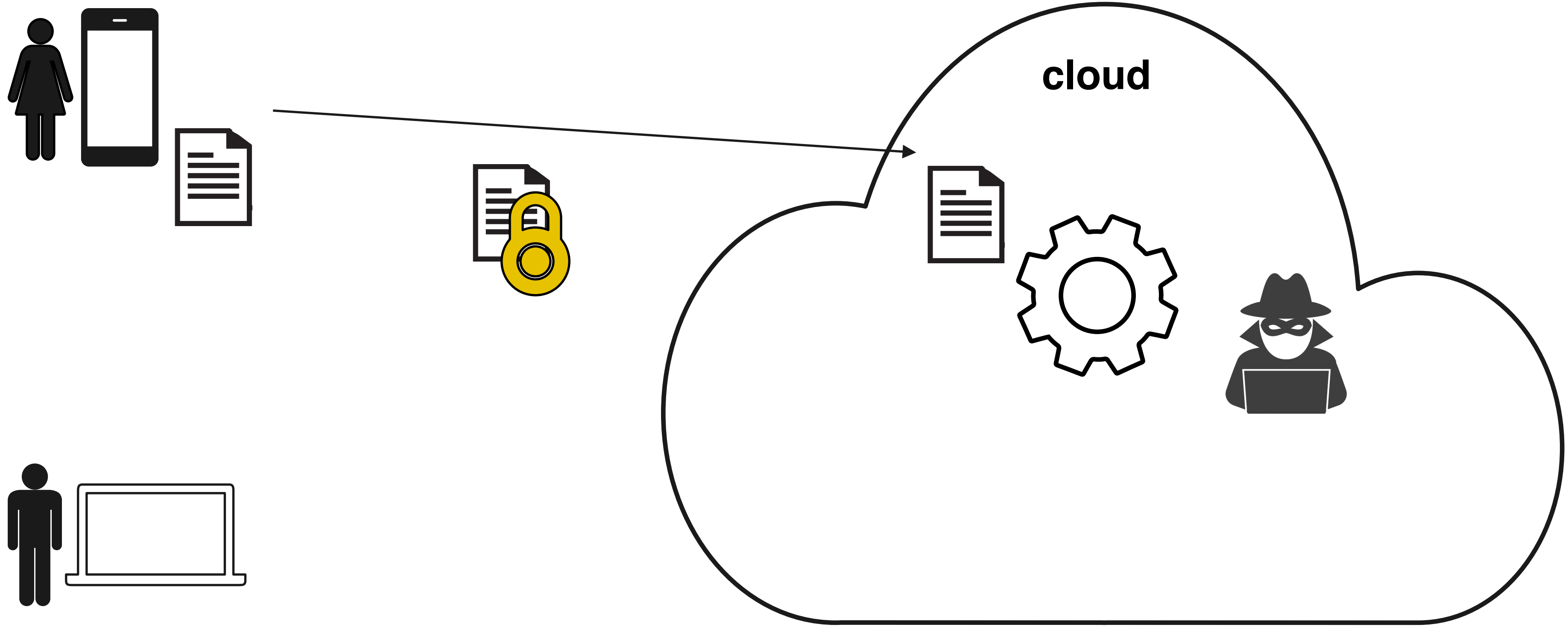
Standard use of encryption



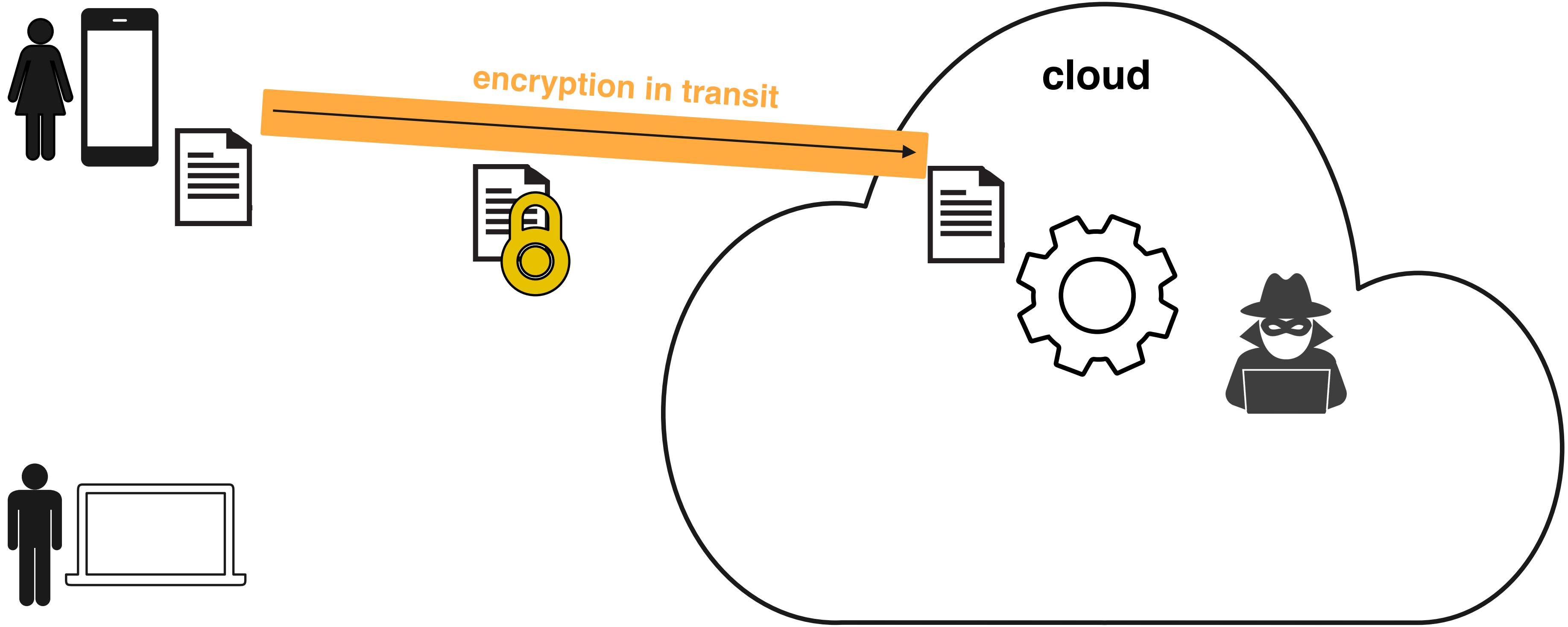
Standard use of encryption



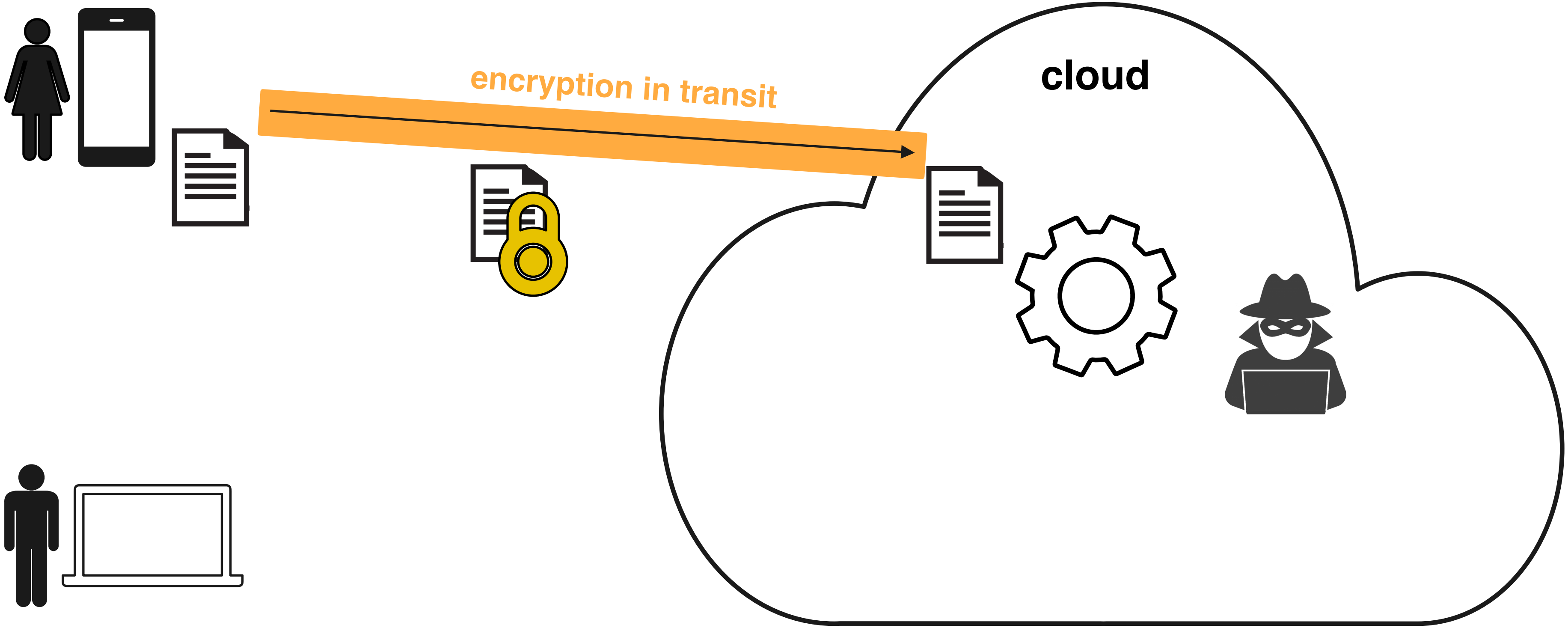
Standard use of encryption



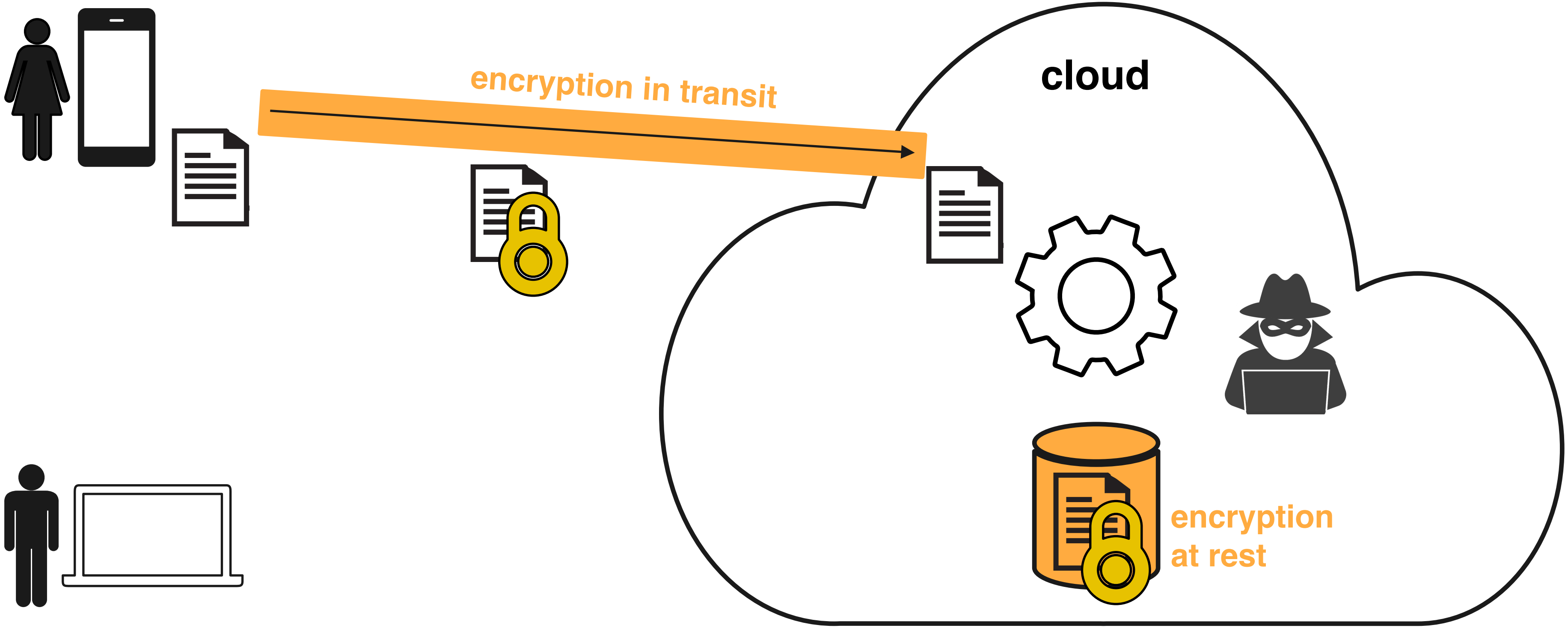
Standard use of encryption



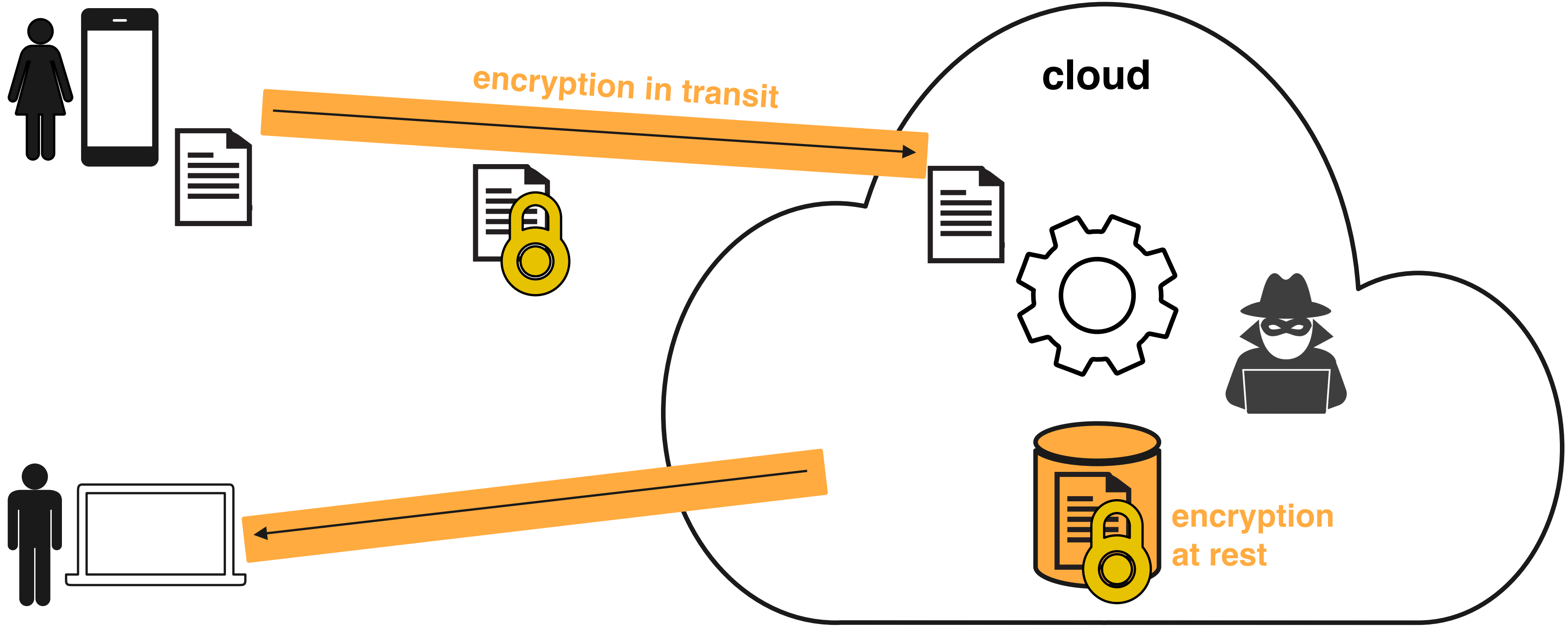
Standard use of encryption



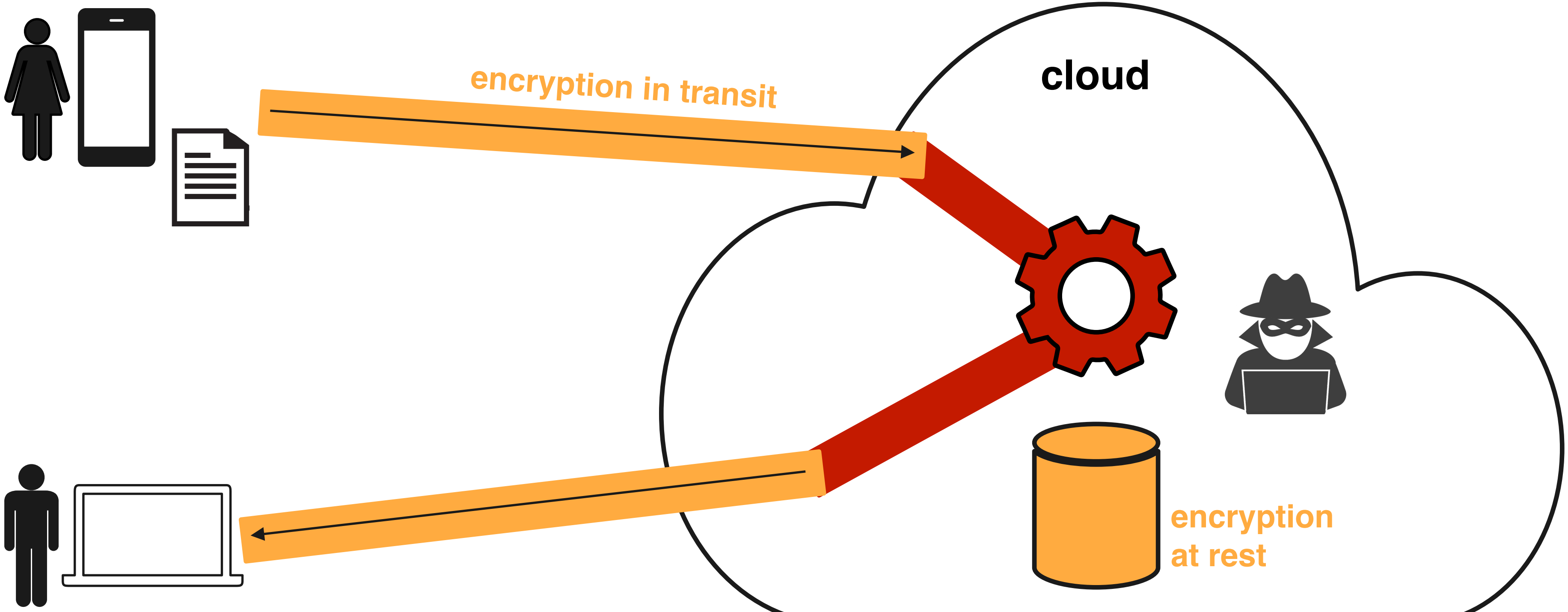
Standard use of encryption



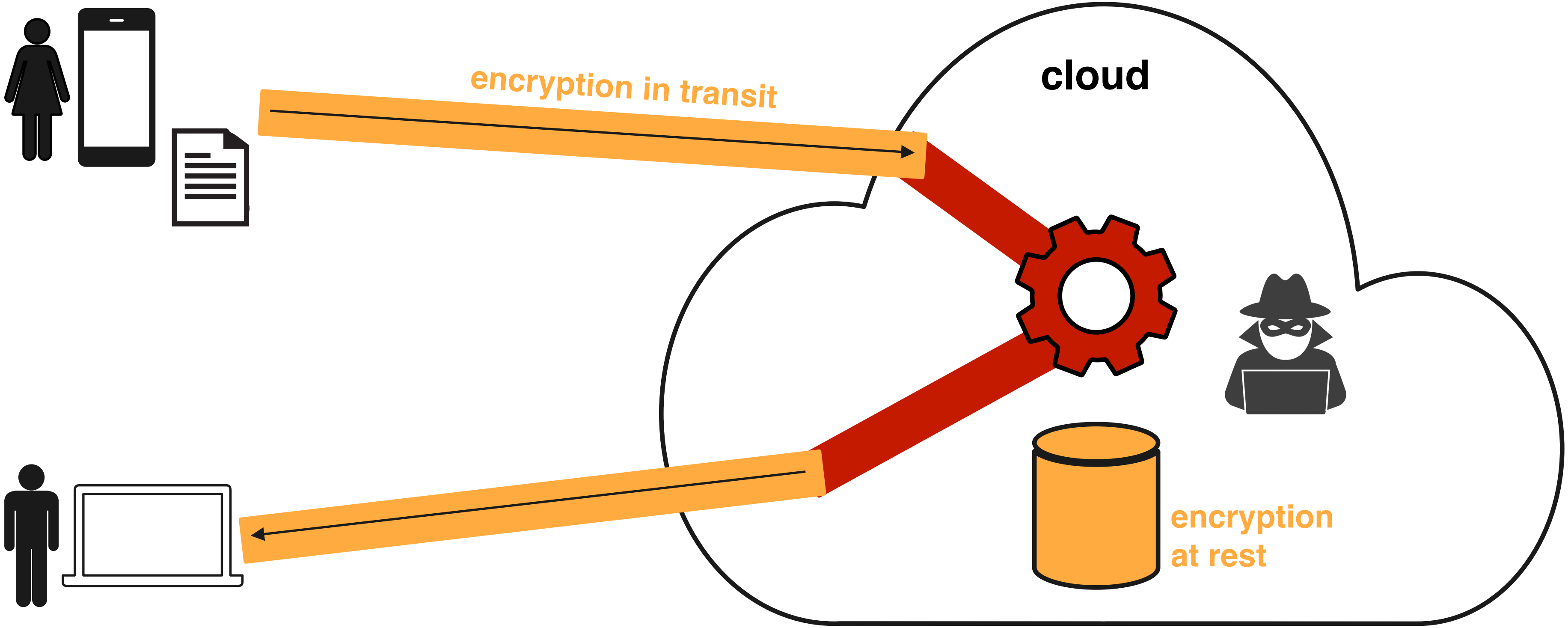
Standard use of encryption



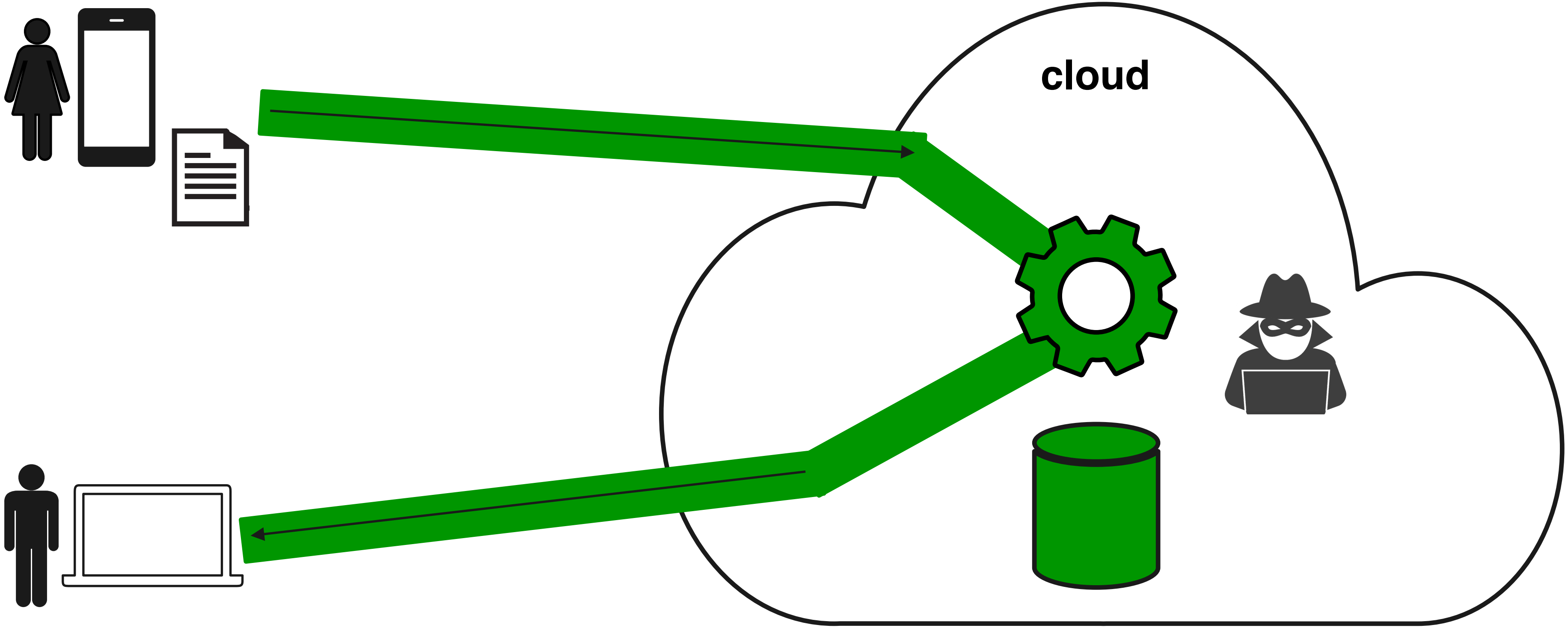
Use encryption



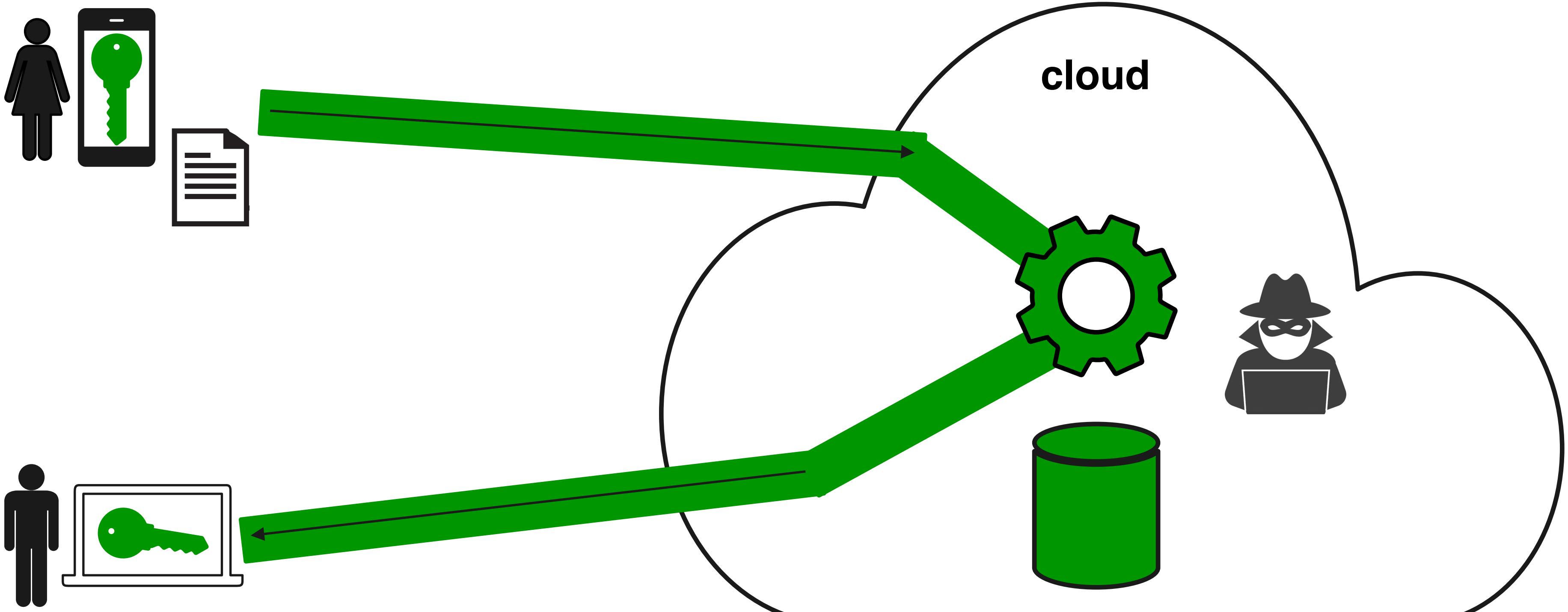
Use encryption



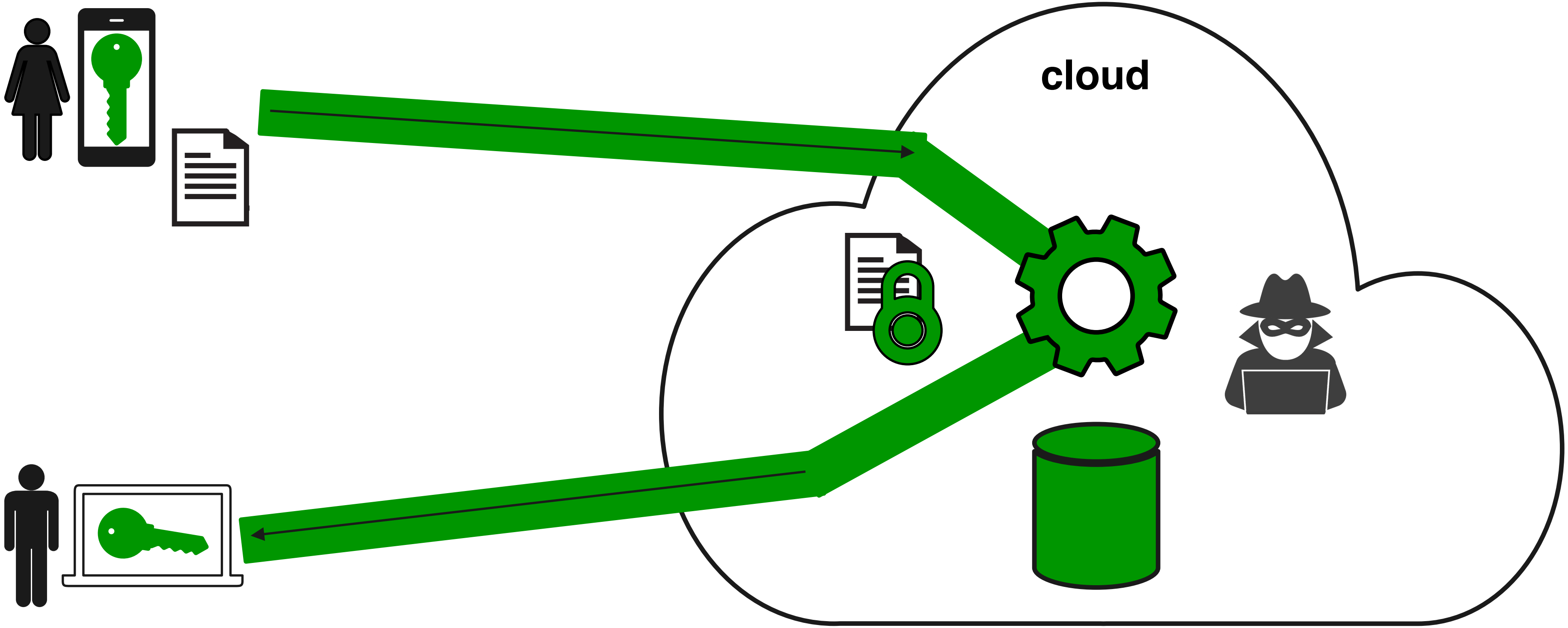
Use **end-to-end** encryption



Use **end-to-end** encryption

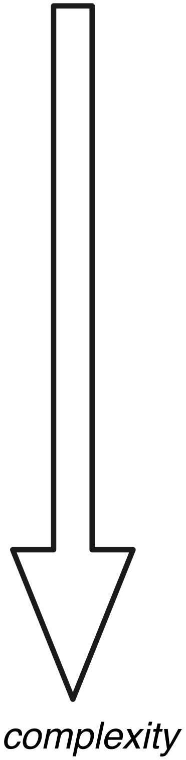


Use **end-to-end** encryption

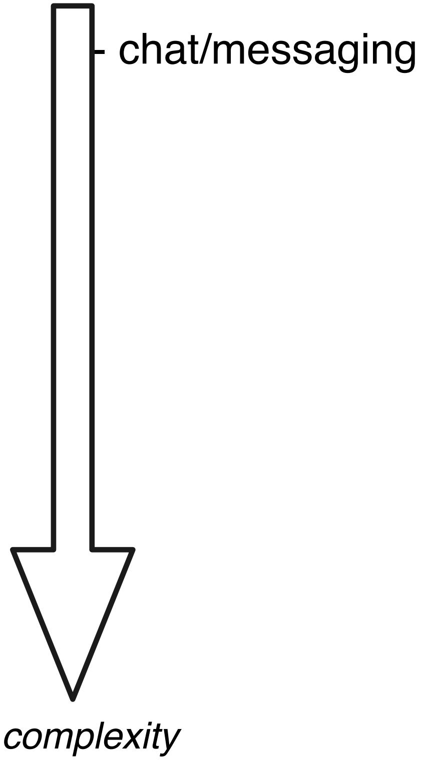


Systems in the cloud

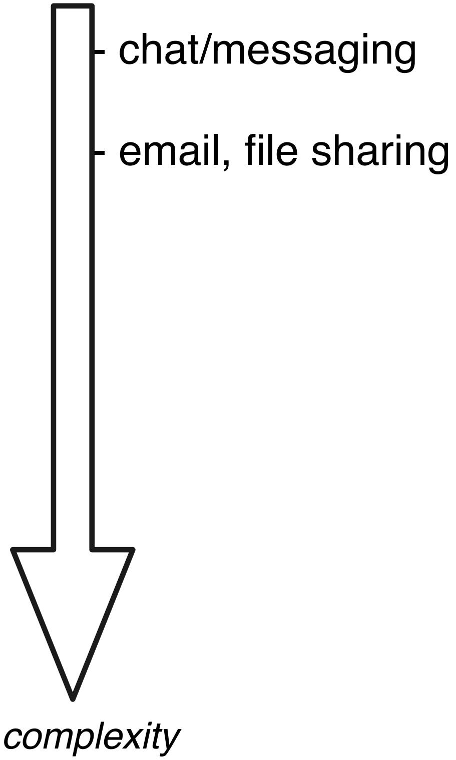
Systems in the cloud



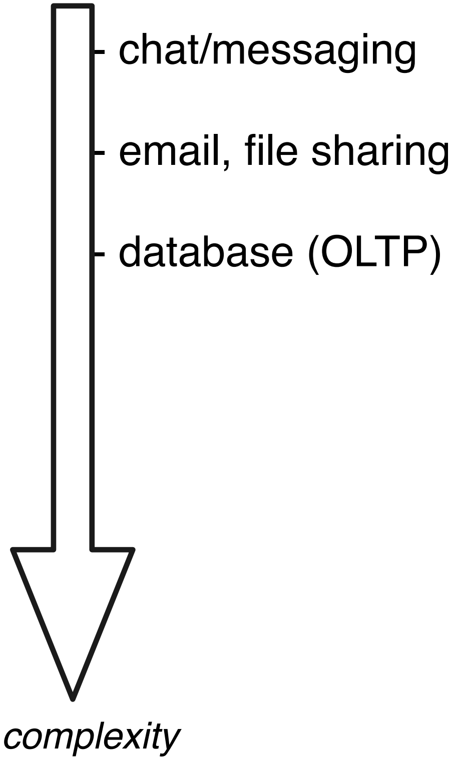
Systems in the cloud



Systems in the cloud



Systems in the cloud

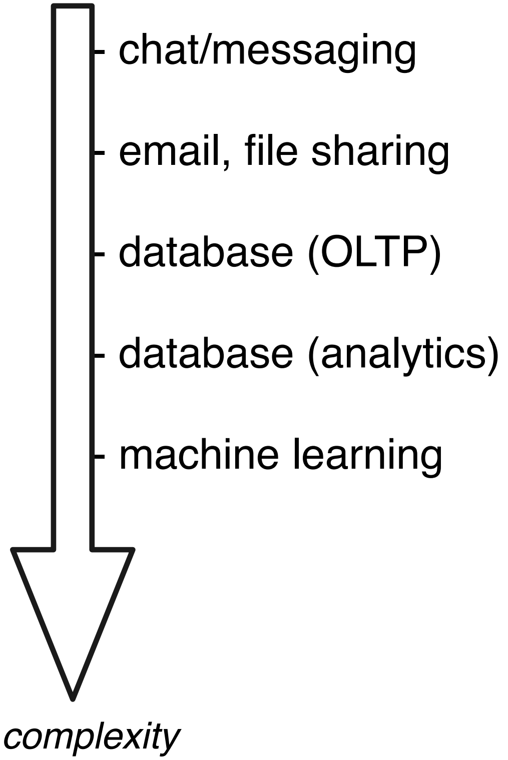


Systems in the cloud

- chat/messaging
- email, file sharing
- database (OLTP)
- database (analytics)

complexity

Systems in the cloud



My work

- chat/messaging — **JEDI**[USEC19] **WAVE**[USEC19]
- email, file sharing — **Verena**[IEEESP16] **Mylar** [NSDI14] **CloudProof** [Usenix11]

PREVEIL

- database (OLTP) — **Arx**[VLDB19], **Oblix**[IEEESP18], **CryptDB**[SOSP11],**mOPE**[IEEESP13], **BlindBox**[SIGCOMM15],**Embark**[NSDI16]
- database (analytics) — **Opaque**[NSDI17]
- machine learning — **Helen**[IEEESP19], **Delphi**[USEC20], **Bost et al.**[NDSS15]

complexity

My work

- chat/messaging: **JEDI**[USEC19] **WAVE**[USEC19]
- email, file sharing: **Verena**[IEEESP16] **Mylar** [NSDI14] **CloudProof** [Usenix11]
- database (OLTP): **Arx**[VLDB19], **Oblix**[IEEESP18], **CryptDB**[SOSP11],**mOPE**[IEEESP13], **BlindBox**[SIGCOMM15],**Embark**[NSDI16]
- database (analytics): **Opaque**[NSDI17]
- machine learning: **Helen**[IEEESP19], **Delphi**[USEC20], **Bost et al.**[NDSS15]

PREVEIL

complexity

End-to-end (E2E) encrypted chat/messaging

End-to-end (E2E) encrypted chat/messaging

Widely adopted industry solutions

End-to-end (E2E) encrypted chat/messaging

Widely adopted industry solutions

WhatsApp

Research on many-to-many (JEDI[USEC19]), constrained devices (e.g. IoT WAVE[USEC19]), usability

E2E encrypted email and file sharing

E2E encrypted email and file sharing

- More complex than chat: add access, revoke access, edit documents

E2E encrypted email and file sharing

- More complex than chat: add access, revoke access, edit documents
- Challenge: key distribution without affecting usability

E2E encrypted email and file sharing

- More complex than chat: add access, revoke access, edit documents
- Challenge: key distribution without affecting usability

ProtonMail

tresorit

PREVEIL

E2E encrypted email and file sharing

- More complex than chat: add access, revoke access, edit documents
- Challenge: key distribution without affecting usability

ProtonMail

tresorit

PREVEIL

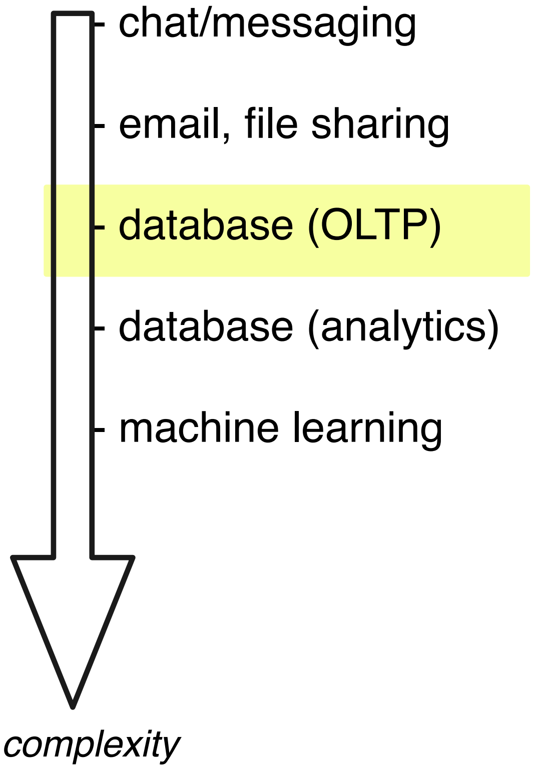
- Research focusing on malicious cloud attackers (Verena[IEEESP16]), usability, search

Systems in the cloud

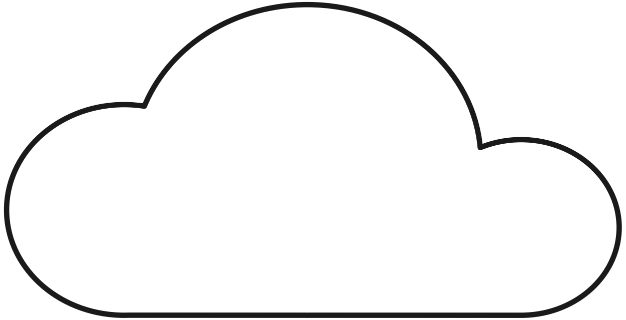
- chat/messaging
- email, file sharing
- database (OLTP)
- database (analytics)
- machine learning

complexity

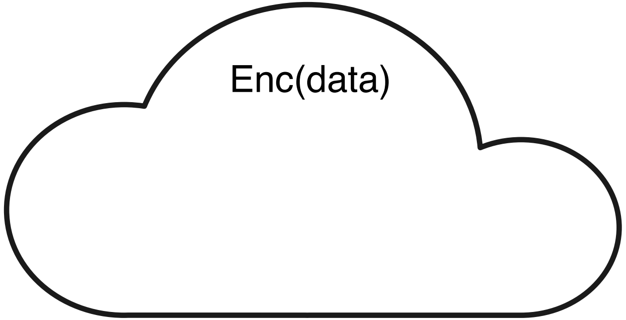
Systems in the cloud



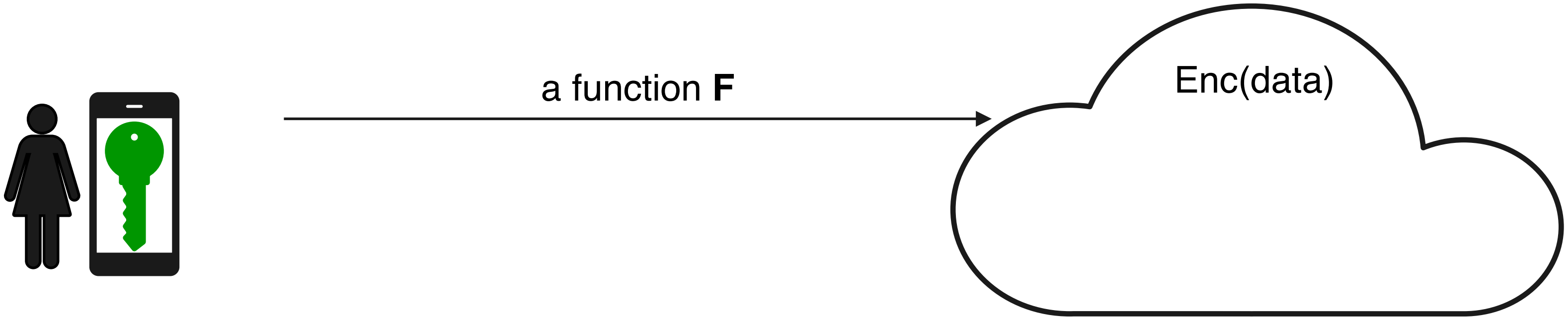
Computation on encrypted data [RAD78, Gentry09]



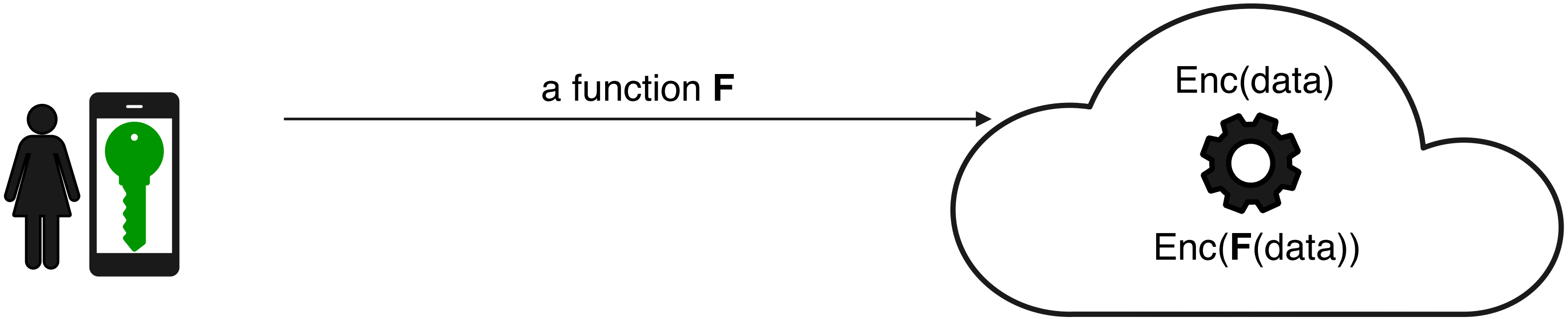
Computation on encrypted data [RAD78, Gentry09]



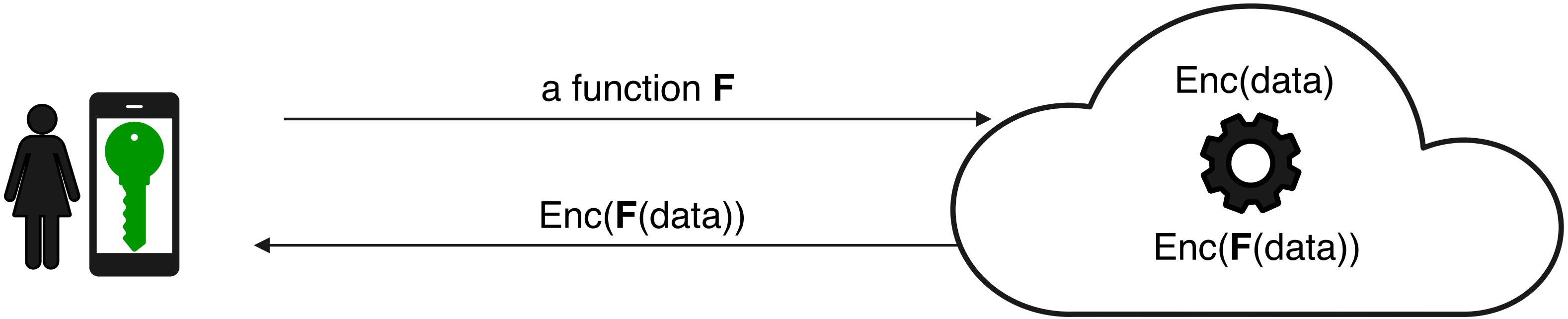
Computation on encrypted data [RAD78, Gentry09]



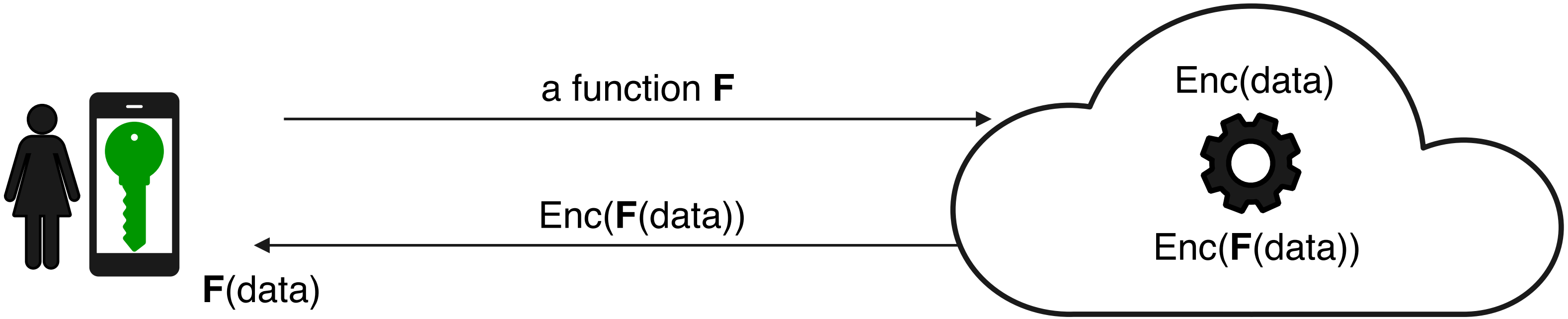
Computation on encrypted data [RAD78, Gentry09]



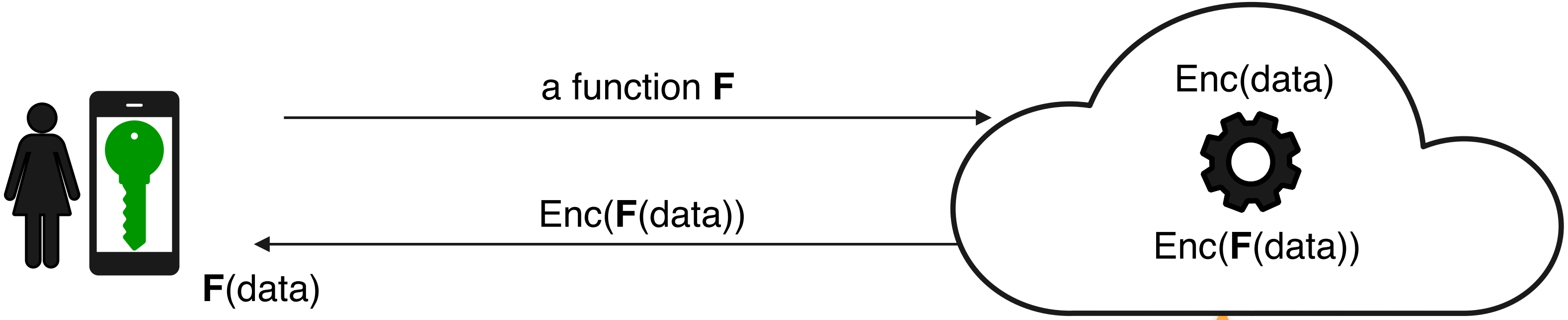
Computation on encrypted data [RAD78, Gentry09]



Computation on encrypted data [RAD78, Gentry09]



Computation on encrypted data [RAD78, Gentry09]

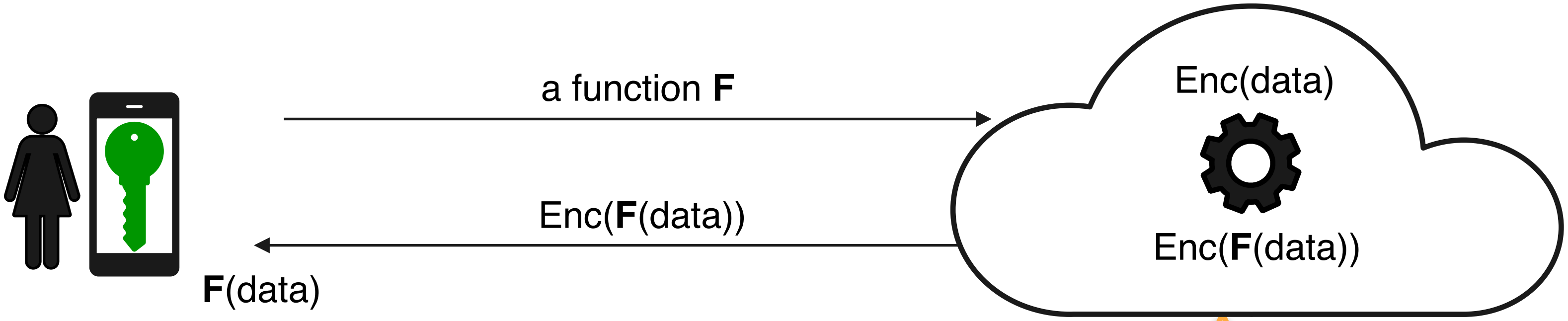


Example: Paillier cryptosystem, **F** = +

Enc(x) = $g^x r^n \bmod n^2$

Enc(y) = $g^y r^n \bmod n^2$

Computation on encrypted data [RAD78, Gentry09]



Example: Paillier cryptosystem, **F** = +

Enc(x) = $g^x r^n \bmod n^2$

Enc(y) = $g^y r^n \bmod n^2$

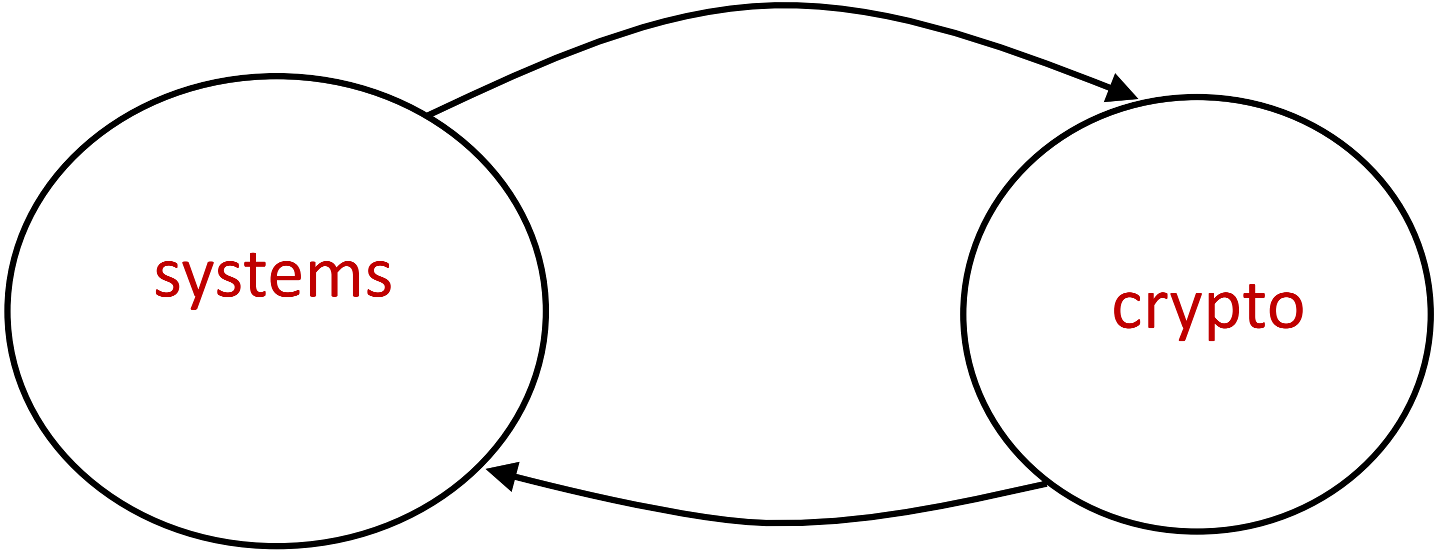
(multiply)

Enc(x) * Enc(y) = $g^{x+y}(rr')^n \bmod n^2$ = Enc(x+y)

Fully homomorphic encryption [Gentry09]

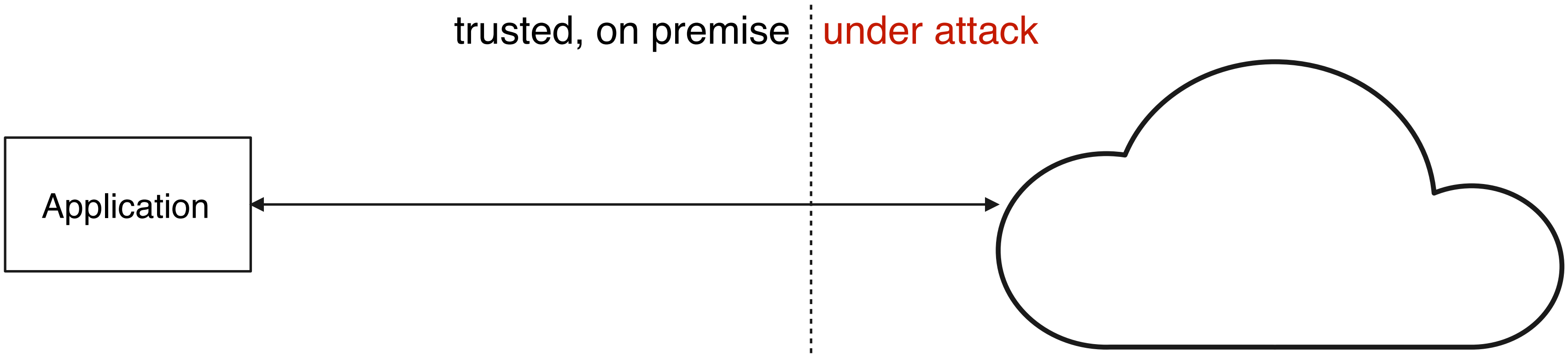
- enables general functions on encrypted data
- despite much progress, remains orders of magnitude too slow

Approach to build practical systems: co-design systems and cryptography



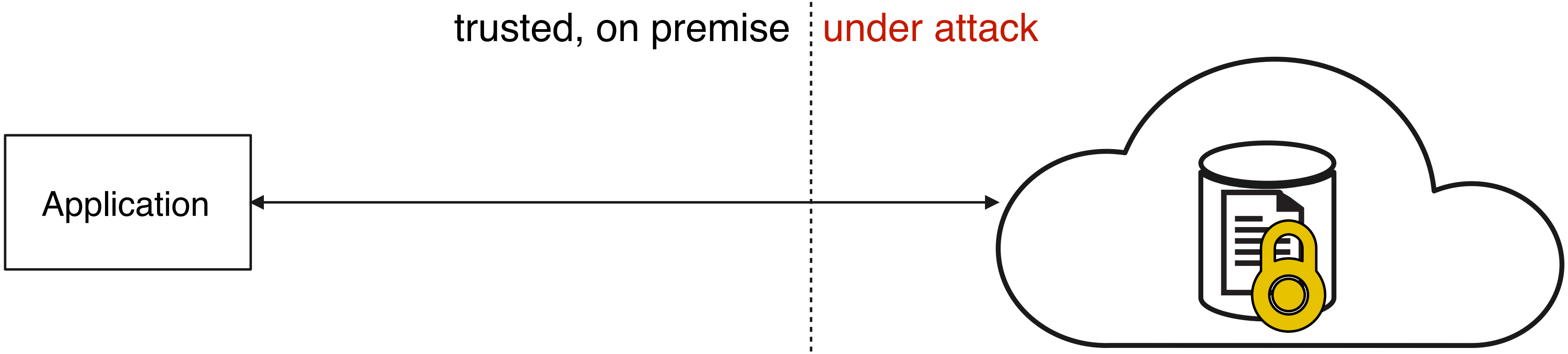
Encrypted databases: CryptDB [SOSP11]

CryptDB was the first DBMS to process SQL queries on encrypted data



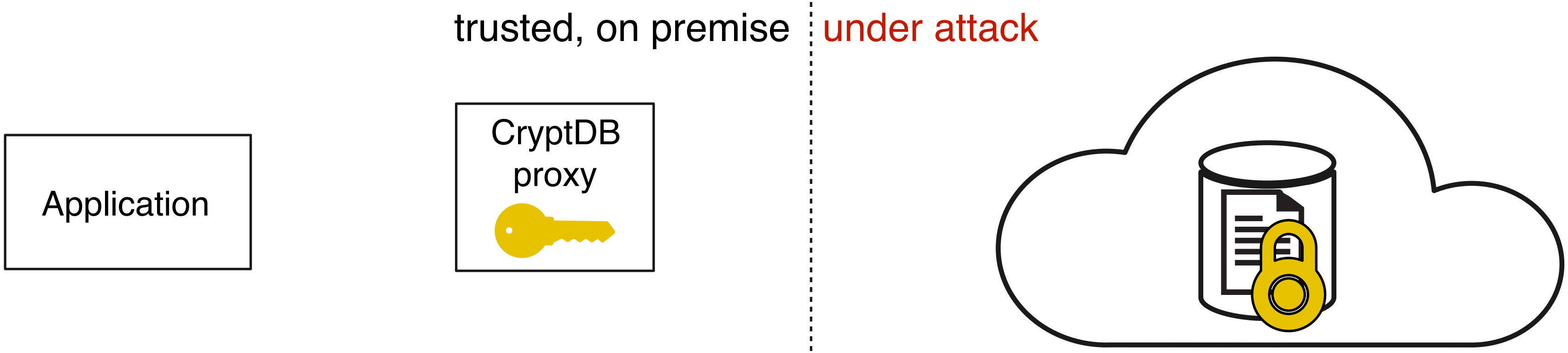
Encrypted databases: CryptDB [SOSP11]

CryptDB was the first DBMS to process SQL queries on encrypted data



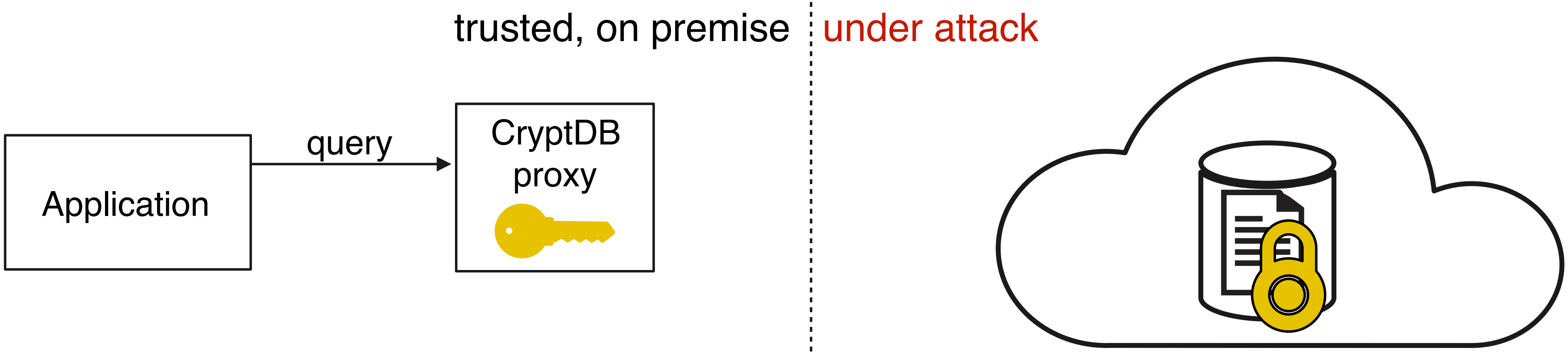
Encrypted databases: CryptDB [SOSP11]

CryptDB was the first DBMS to process SQL queries on encrypted data



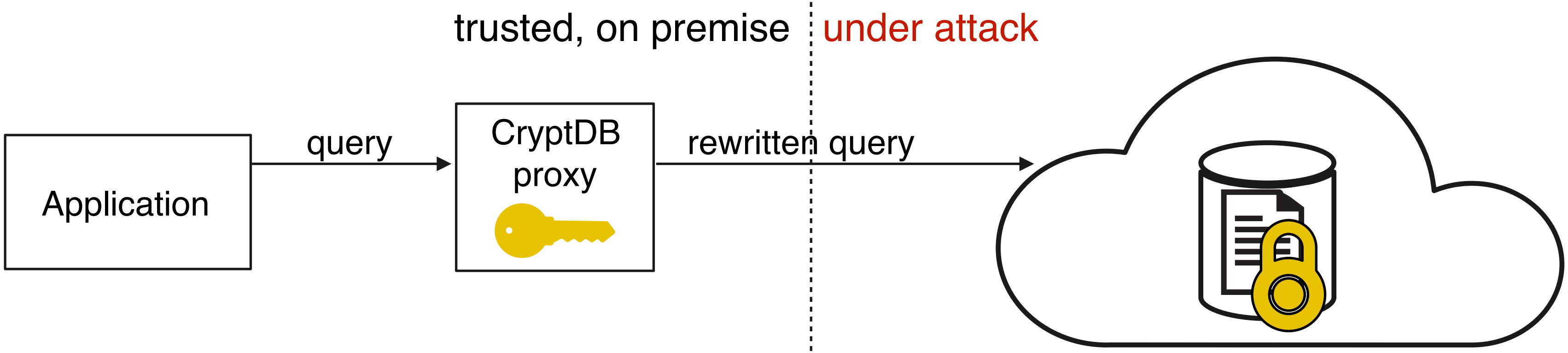
Encrypted databases: CryptDB [SOSP11]

CryptDB was the first DBMS to process SQL queries on encrypted data



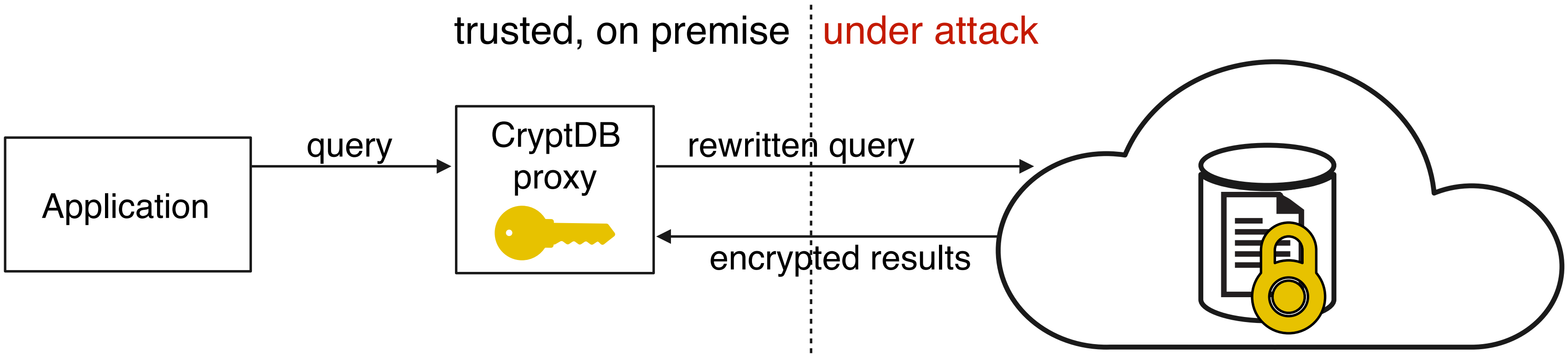
Encrypted databases: CryptDB [SOSP11]

CryptDB was the first DBMS to process SQL queries on encrypted data



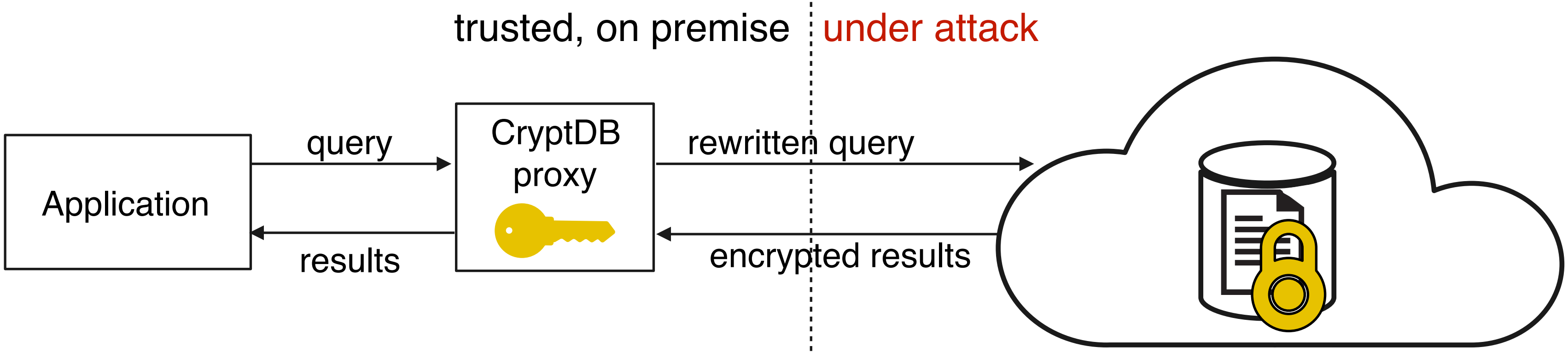
Encrypted databases: CryptDB [SOSP11]

CryptDB was the first DBMS to process SQL queries on encrypted data



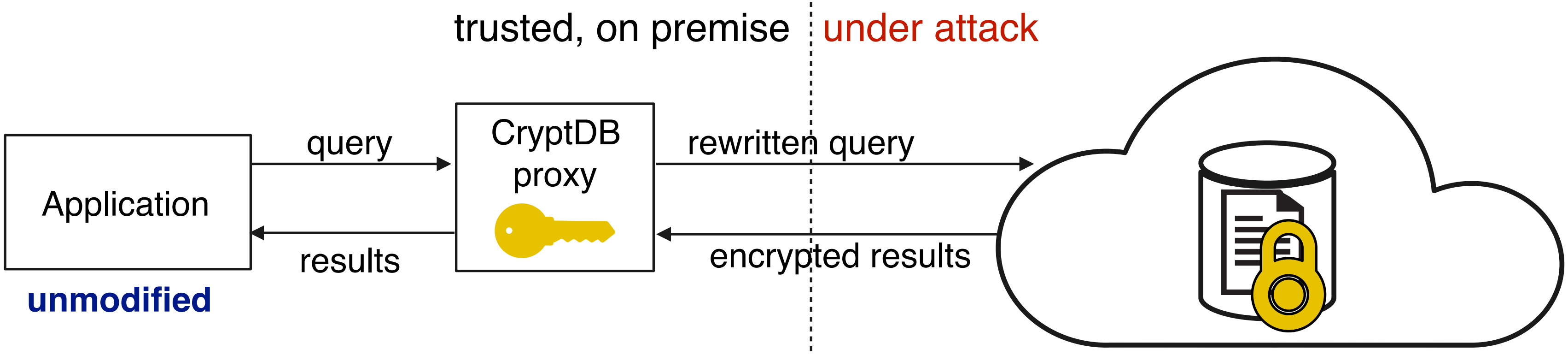
Encrypted databases: CryptDB [SOSP11]

CryptDB was the first DBMS to process SQL queries on encrypted data



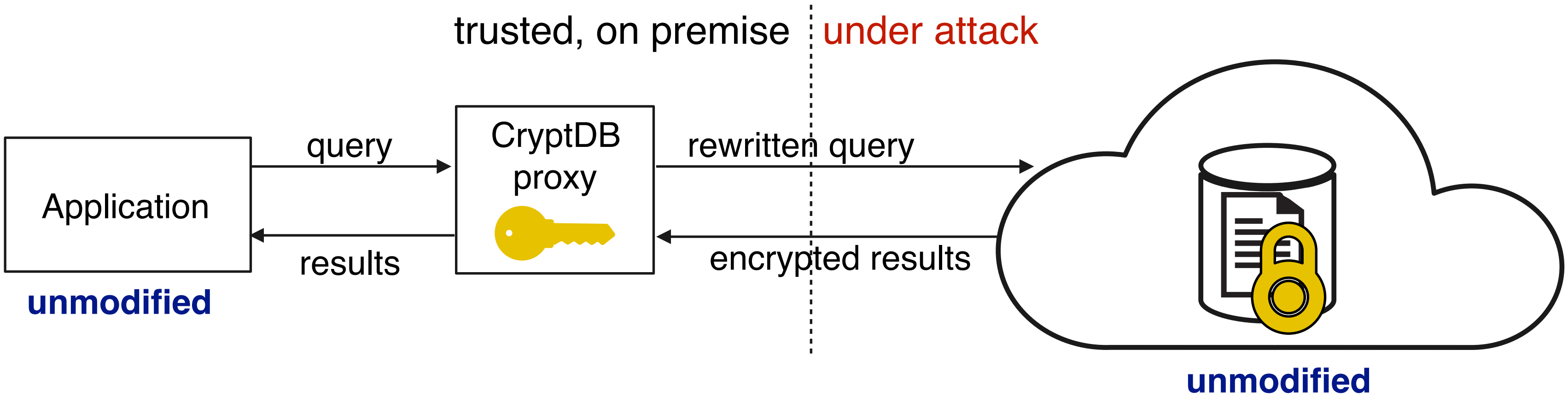
Encrypted databases: CryptDB [SOSP11]

CryptDB was the first DBMS to process SQL queries on encrypted data



Encrypted databases: CryptDB [SOSP11]

CryptDB was the first DBMS to process SQL queries on encrypted data



CryptDB in a nutshell

CryptDB in a nutshell

Observation: Most SQL can be implemented with a few core operations (e.g.,+,=,>)

CryptDB in a nutshell

Observation: Most SQL can be implemented with a few core operations (e.g.,+,=,>)

Tech.#1: Employ an efficient encryption scheme for each operation

CryptDB in a nutshell

Observation: Most SQL can be implemented with a few core operations (e.g.,+,=,>)

Tech.#1: Employ an efficient encryption scheme for each operation

- design your own if needed

CryptDB in a nutshell

Observation: Most SQL can be implemented with a few core operations (e.g.,+,=,>)

Tech.#1: Employ an efficient encryption scheme for each operation

- design your own if needed

Tech.#2, Onion Encryption: combine encryptions based on security vs. functionality

CryptDB in a nutshell

Observation: Most SQL can be implemented with a few core operations (e.g.,+,=,>)

Tech.#1: Employ an efficient encryption scheme for each operation

- design your own if needed

Tech.#2, Onion Encryption: combine encryptions based on security vs. functionality

- e.g., Paillier for +, DET for =

CryptDB in a nutshell

Observation: Most SQL can be implemented with a few core operations (e.g.,+,=,>)

Tech.#1: Employ an efficient encryption scheme for each operation

- design your own if needed

Tech.#2, Onion Encryption: combine encryptions based on security vs. functionality

- e.g., Paillier for +, DET for =

Tech.#3: Redesign the query planner to produce encrypted and transformed query plans

CryptDB in a nutshell

Observation: Most SQL can be implemented with a few core operations (e.g.,+,=,>)

Tech.#1: Employ an efficient encryption scheme for each operation

- design your own if needed

Tech.#2, Onion Encryption: combine encryptions based on security vs. functionality

- e.g., Paillier for +, DET for =

Tech.#3: Redesign the query planner to produce encrypted and transformed query plans

- resulting queries did not change the DBMS

CryptDB in a nutshell

Observation: Most SQL can be implemented with a few core operations (e.g.,+,=,>)

Tech.#1: Employ an efficient encryption scheme for each operation

- design your own if needed

Tech.#2, Onion Encryption: combine encryptions based on security vs. functionality

- e.g., Paillier for +, DET for =

Tech.#3: Redesign the query planner to produce encrypted and transformed query plans

- resulting queries did not change the DBMS

CryptDB in a nutshell

Observation: Most SQL can be implemented with a few core operations (e.g.,+,=,>)

Tech.#1: Employ an efficient encryption scheme for each operation

- design your own if needed

Tech.#2, Onion Encryption: combine encryptions based on security vs. functionality

- e.g., Paillier for +, DET for =

Tech.#3: Redesign the query planner to produce encrypted and transformed query plans

- resulting queries did not change the DBMS

Supported all of TPC-C, 27% throughput loss

A rich line of work followed

- Academic work:

  Cipherbase, CMD, Cryptsis, Autocrypt, Clome, SensorCloud, [ABE+13], [TKM+13], Seabed [PBC+16], BlindSeer[PKV+14], [CJJ+14], [FJK+15], [K15], Arx, MrCrypt, Monomi, [NKW15],[DDC16],[GSB17],KKN+16], [DCF+20],… > 1000 citations.

- Industry deployments:

  Microsoft AlwaysEncrypted

  Google's EncryptedBigQuery

  SAP SEEED

  skyhigh

  …

Lesson: co-design of systems and cryptography

Lesson: co-design of systems and cryptography

A recipe:

1. Focus on a workload. Identify a set of core operations the system needs
2. Identify a suitable encryption building block efficient for each operation
3. Design a planner/compiler that can combine the encryption building blocks based on their constraints and cost model

Lesson: co-design of systems and cryptography

A recipe:

1. Focus on a workload. Identify a set of core operations the system needs
2. Identify a suitable encryption building block efficient for each operation
3. Design a planner/compiler that can combine the encryption building blocks based on their constraints and cost model

For the architecture:

- avoid changing existing applications and cloud systems

Lesson: co-design of systems and cryptography

A recipe:

1. Focus on a workload. Identify a set of core operations the system needs
2. **Identify a suitable encryption building block efficient for each operation**
3. Design a planner/compiler that can combine the encryption building blocks based on their constraints and cost model

For the architecture:

- avoid changing existing applications and cloud systems

Research challenge: functionality vs security vs performance

Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

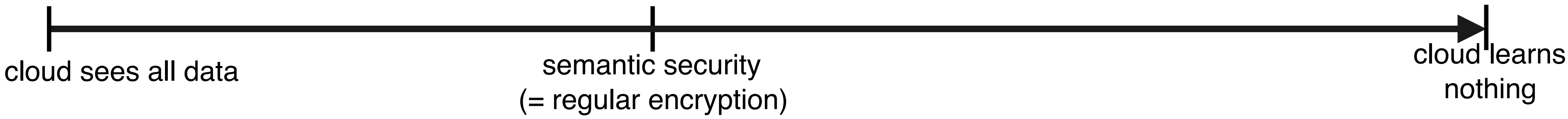
2. Sharp security/performance tradeoff. A “rough” sketch:

Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

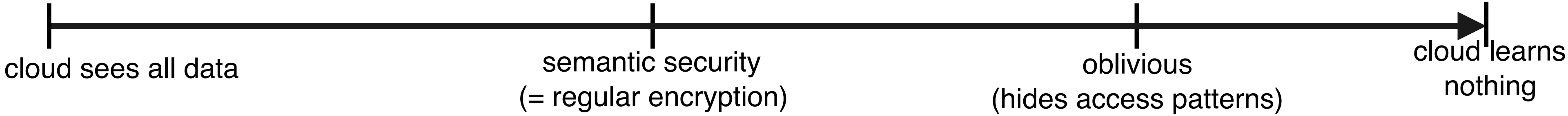


Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

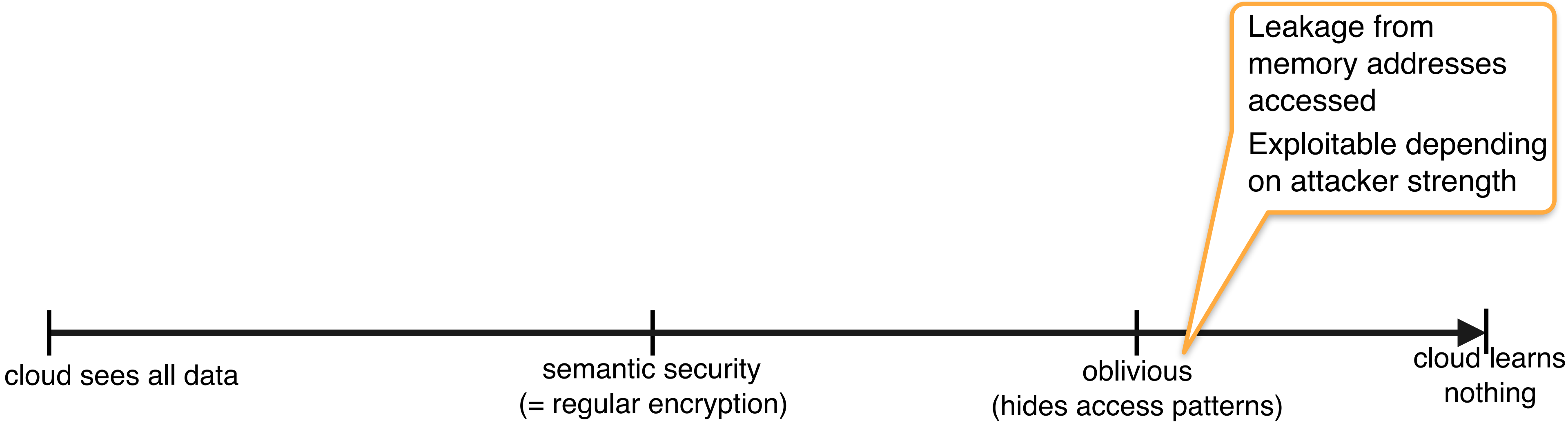


Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

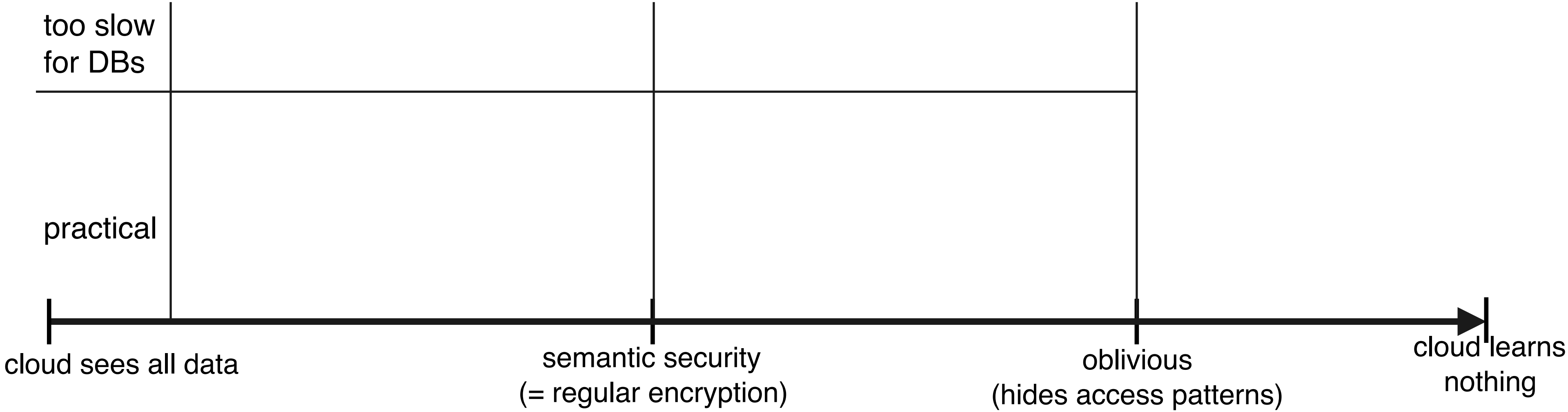


Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

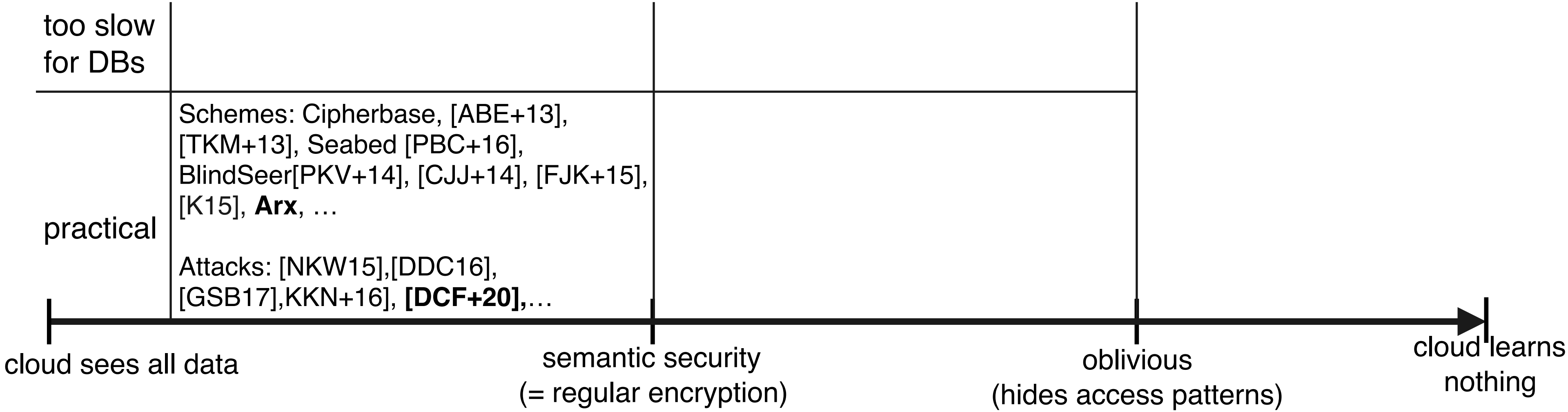


Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

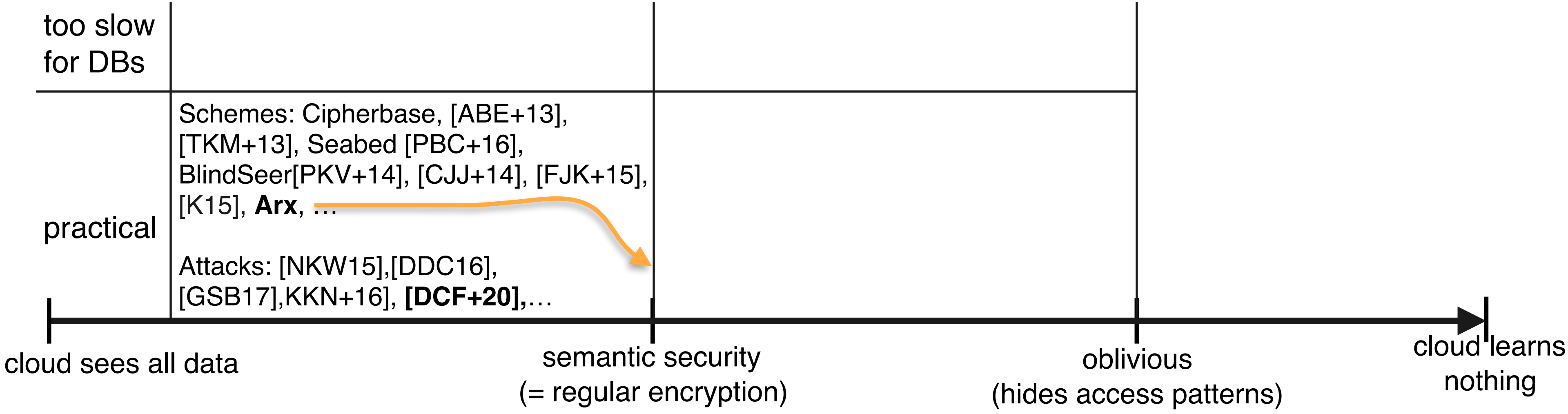


Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

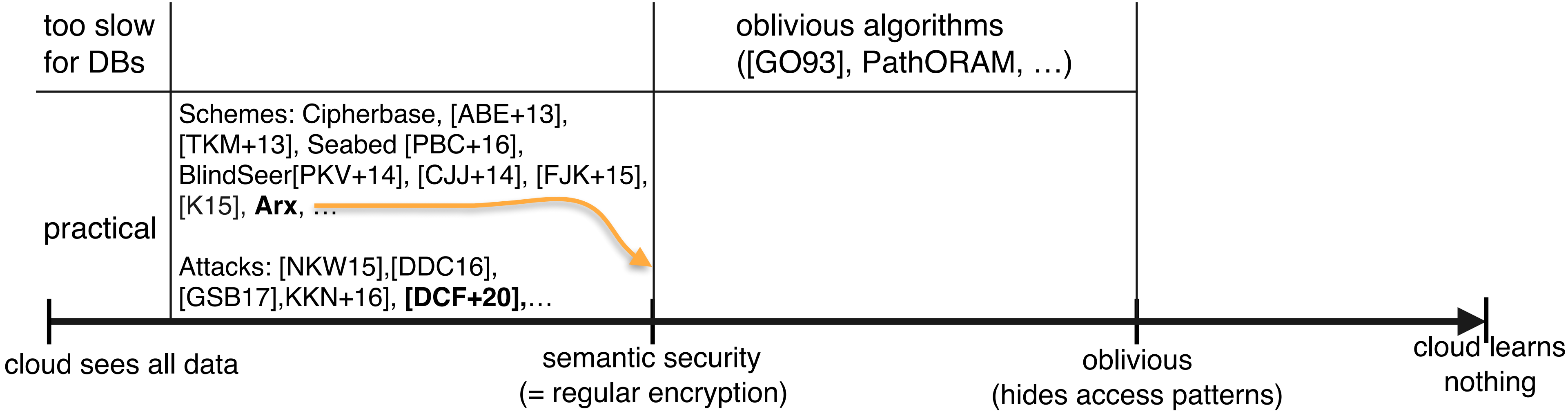


Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

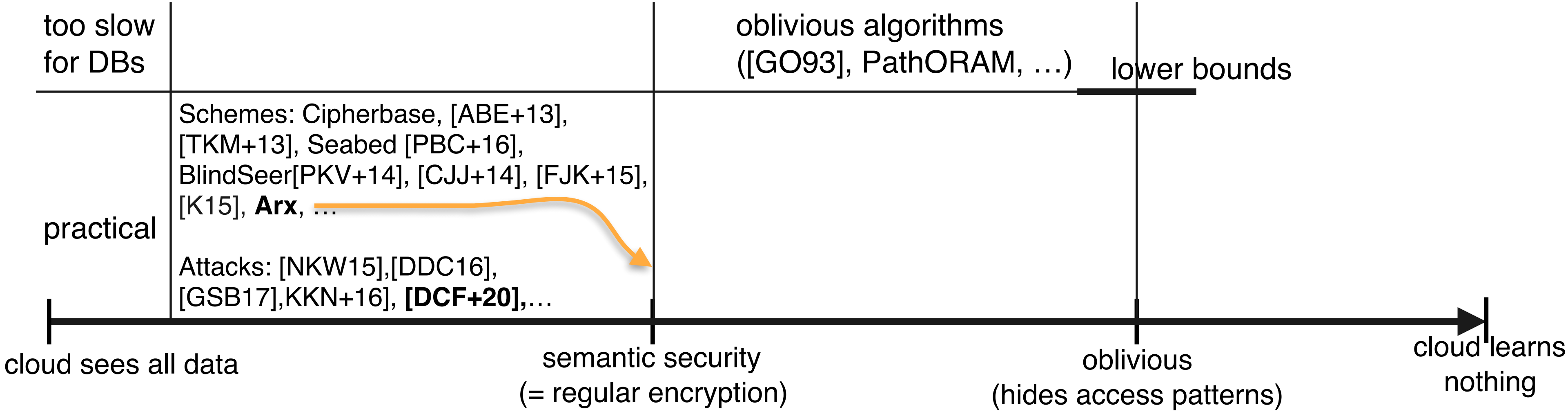


Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

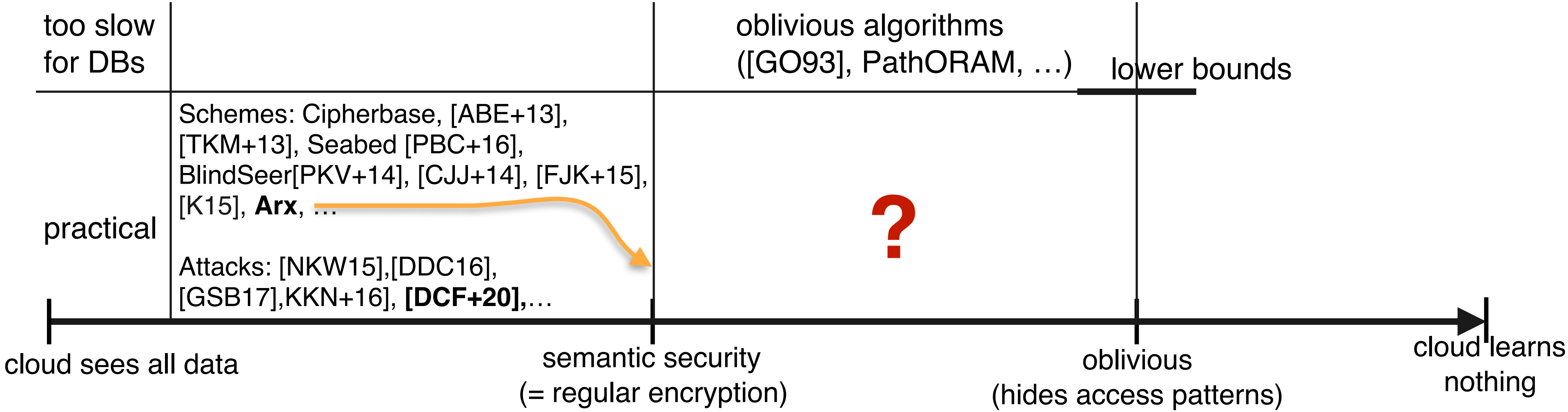


Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

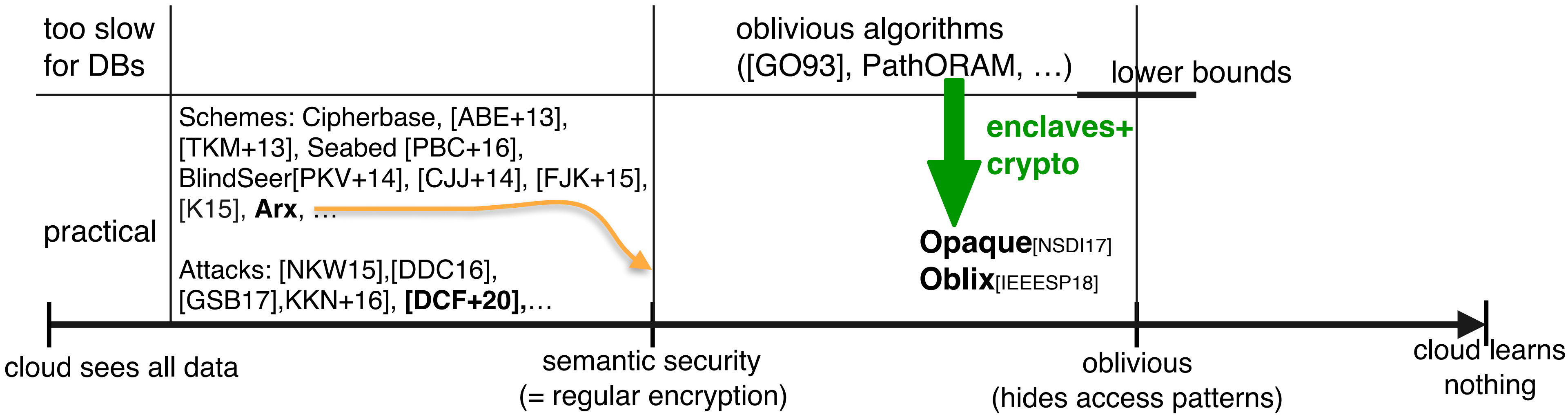


Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

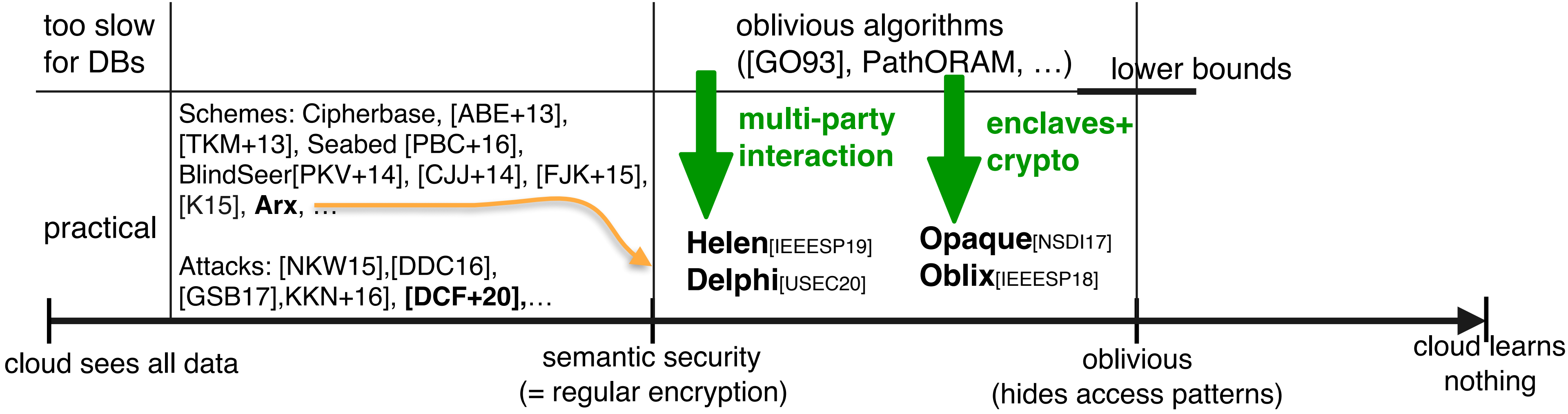


Research challenge: functionality vs security vs performance

1. Existing building blocks had limited functionality

? complex analytics or ML

2. Sharp security/performance tradeoff. A “rough” sketch:

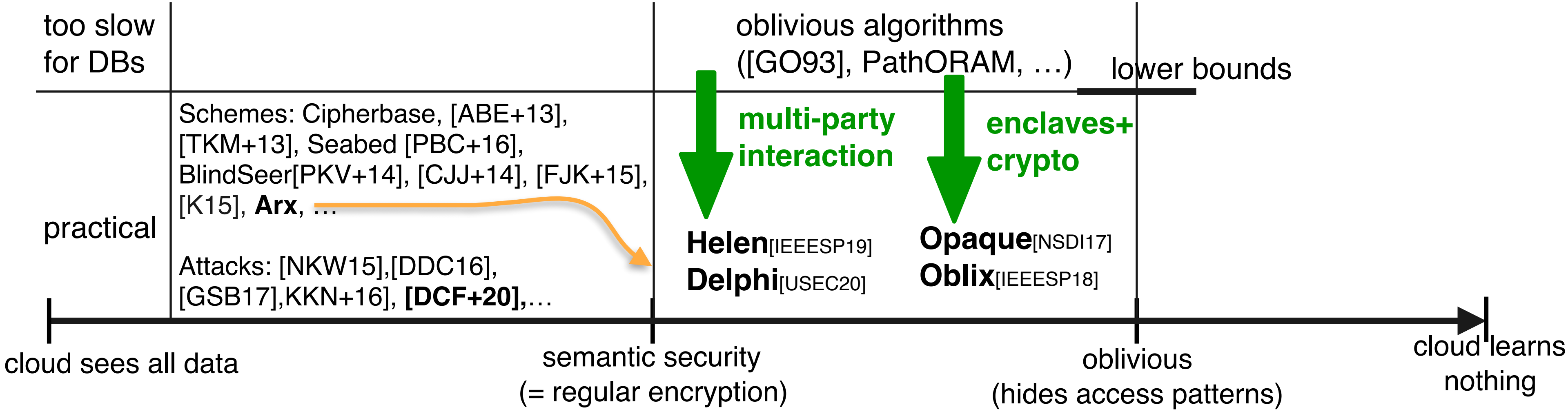


Research challenge: functionality vs security vs performance

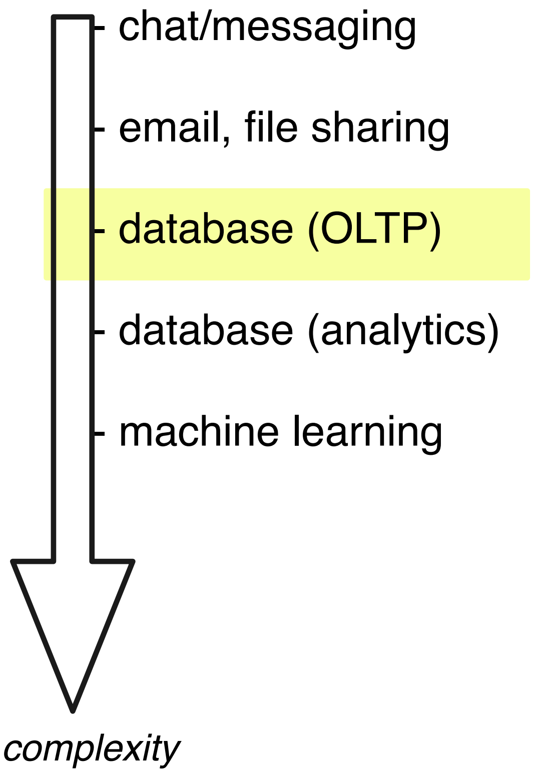
1. Existing building blocks had limited functionality

   ? complex analytics or ML

2. Sharp security/performance tradeoff. A "rough" sketch:



Systems in the cloud



Systems in the cloud

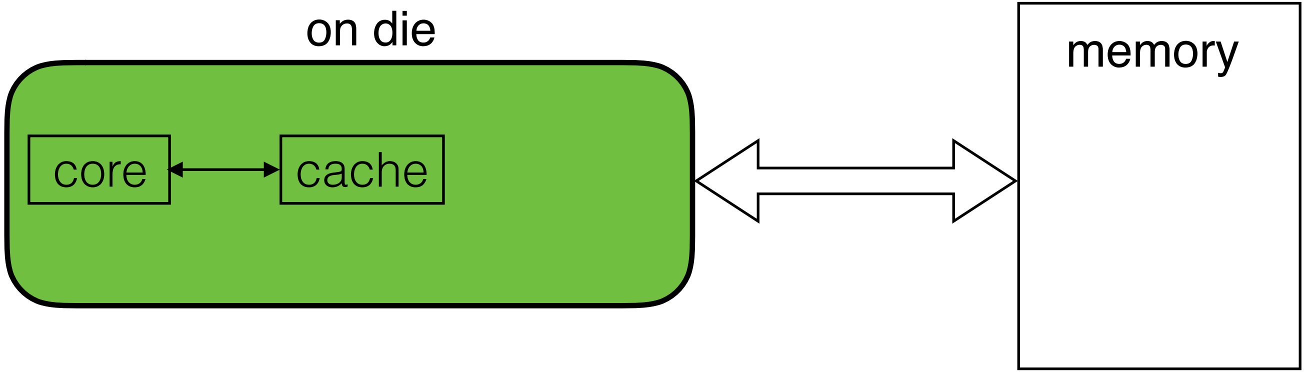
- chat/messaging
- email, file sharing
- database (OLTP)
- database (analytics)
- machine learning

complexity

Hardware enclaves 101

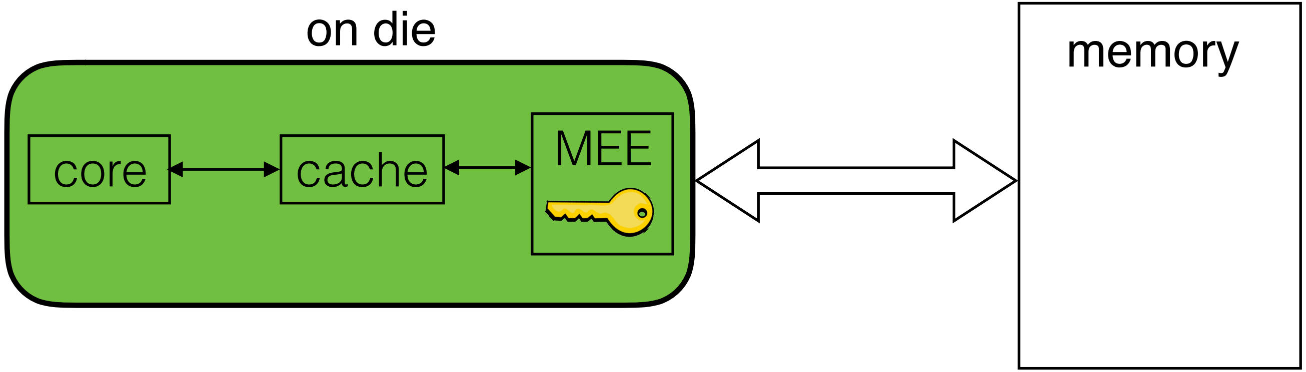
Hardware enclaves (Intel SGX)

- Hardware-enforced isolated execution environment



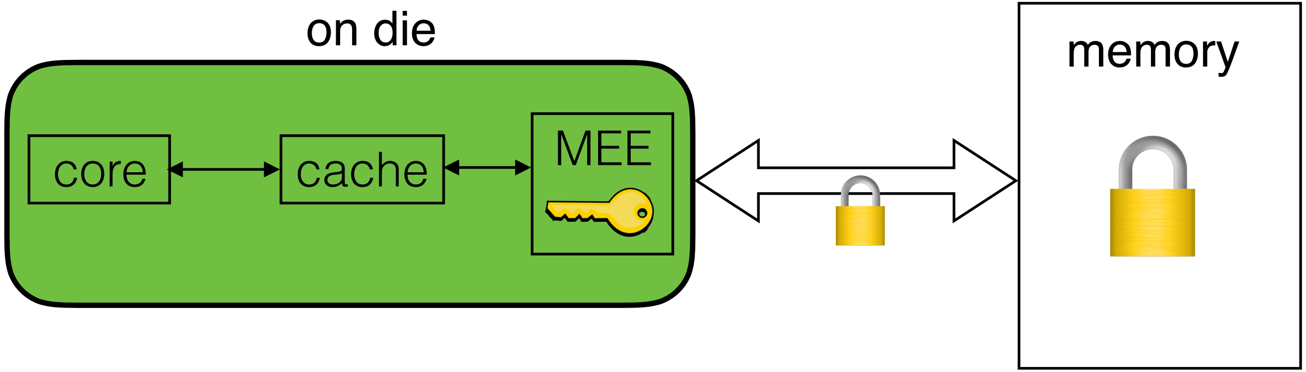
Hardware enclaves (Intel SGX)

- Hardware-enforced isolated execution environment



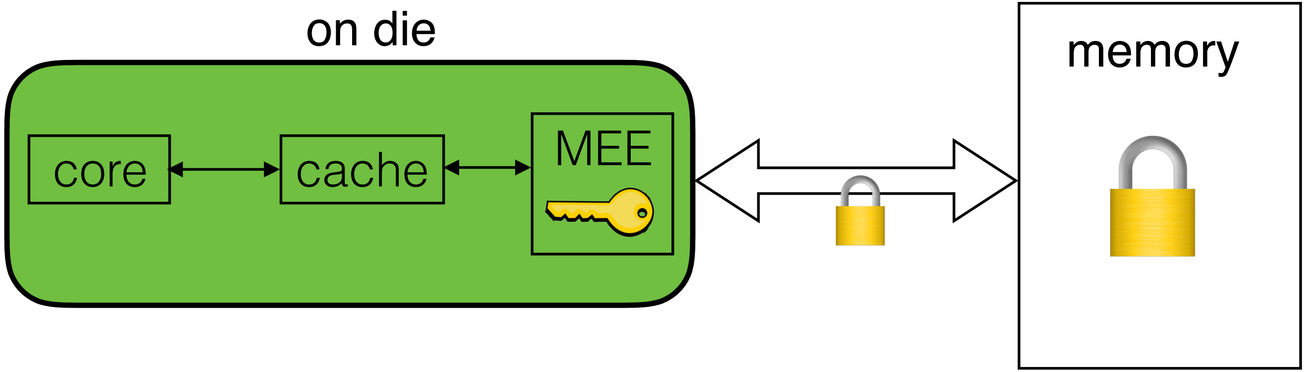
Hardware enclaves (Intel SGX)

- Hardware-enforced isolated execution environment



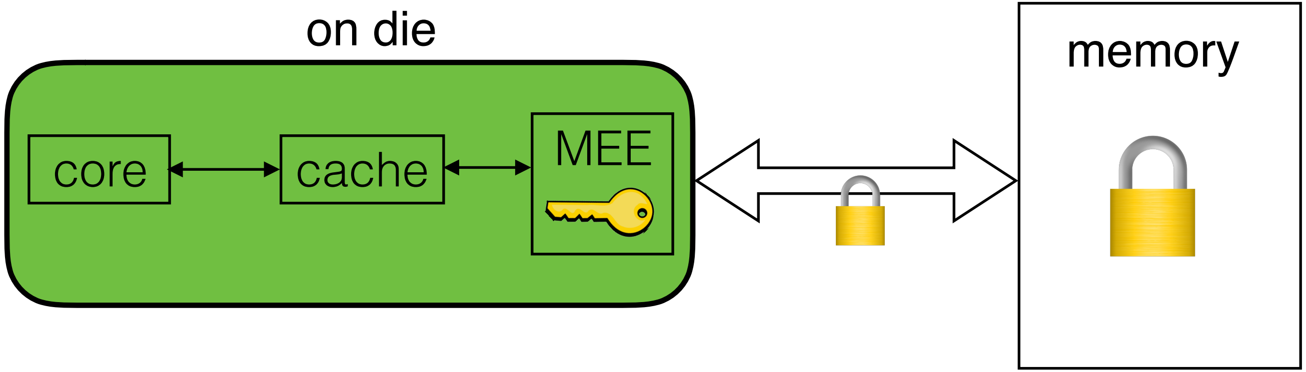
Hardware enclaves (Intel SGX)

- Hardware-enforced isolated execution environment
- Data decrypted only on the processor



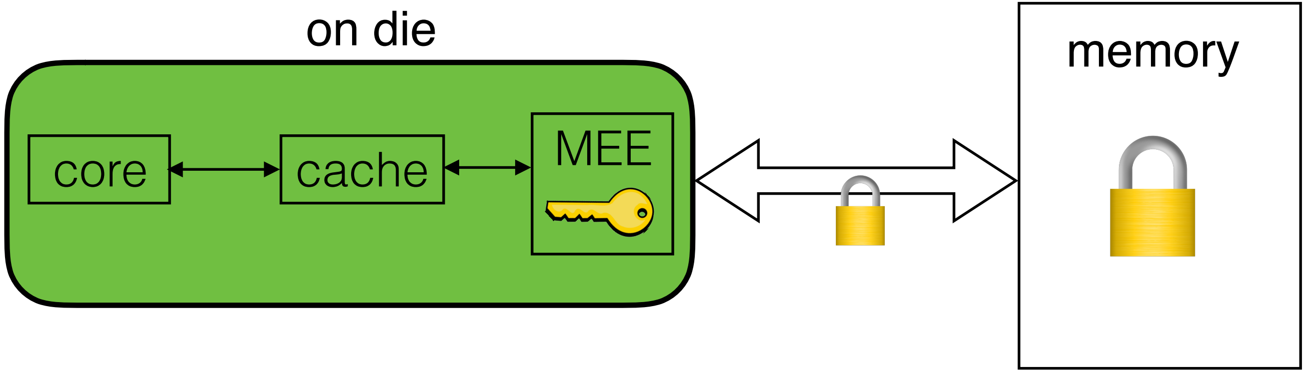
Hardware enclaves (Intel SGX)

- Hardware-enforced isolated execution environment
- Data decrypted only on the processor
- Protect against an attacker who has root access or compromised OS



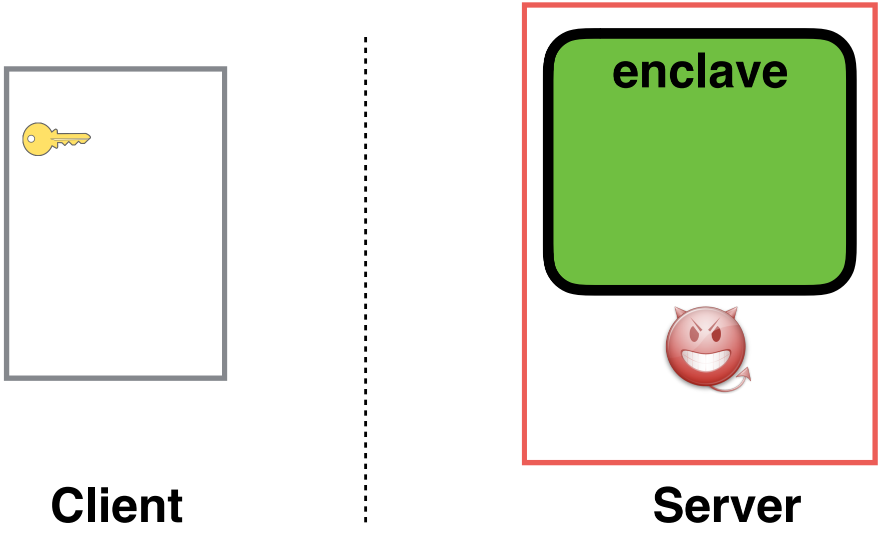
Hardware enclaves (Intel SGX)

- Hardware-enforced isolated execution environment
- Data decrypted only on the processor
- Protect against an attacker who has root access or compromised OS



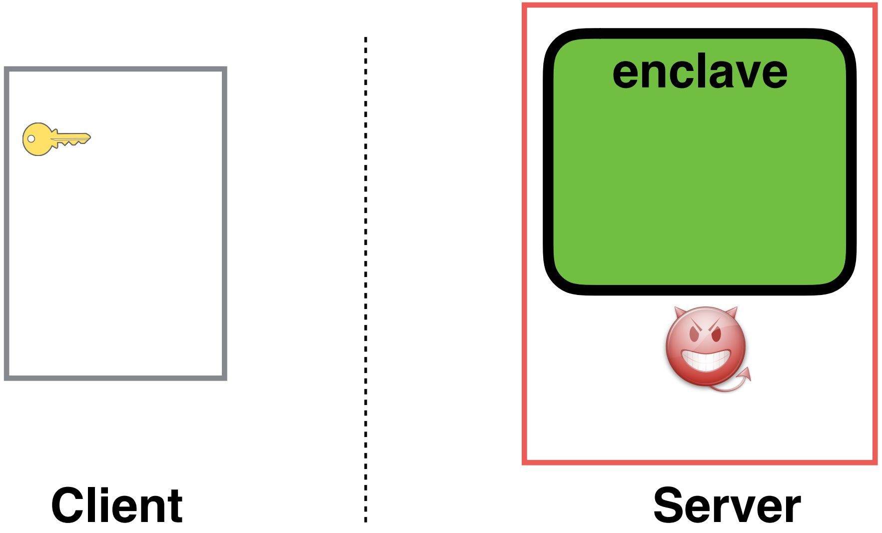
- Cloud offerings: Azure Confidential Computing, Alibaba Cloud

Remote attestation



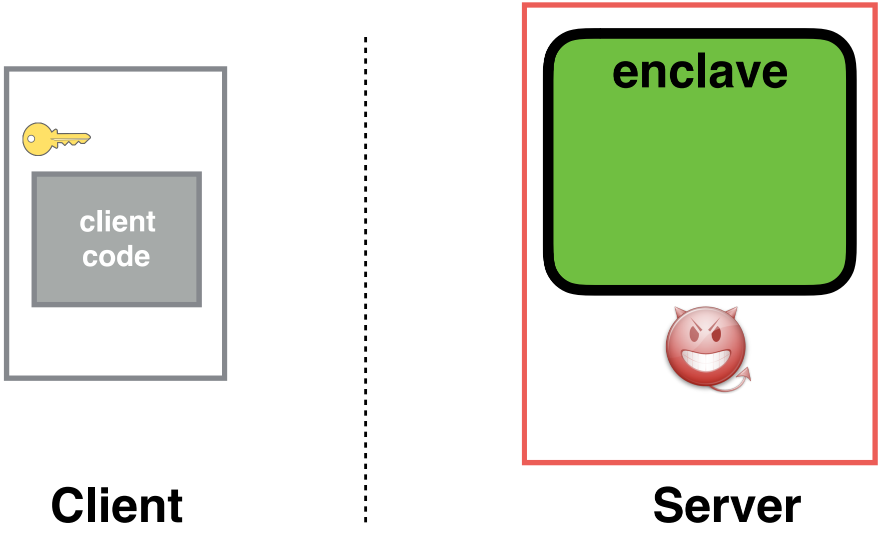
Remote attestation

Enables verifying which code runs in the enclave and performing key exchange



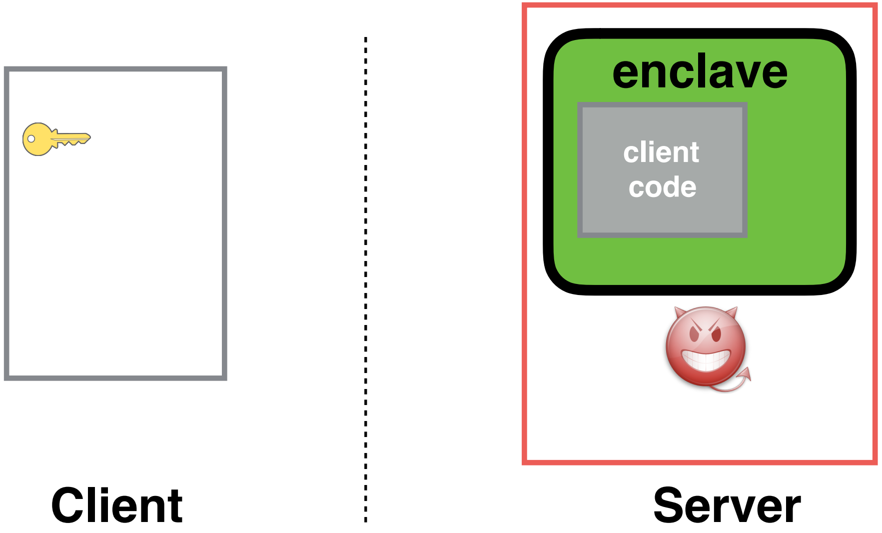
Remote attestation

Enables verifying which code runs in the enclave and performing key exchange



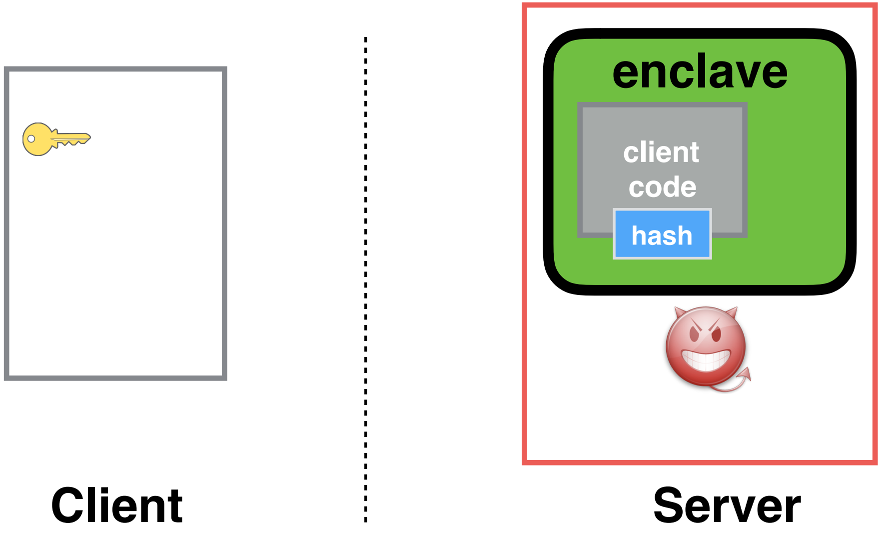
Remote attestation

Enables verifying which code runs in the enclave and performing key exchange



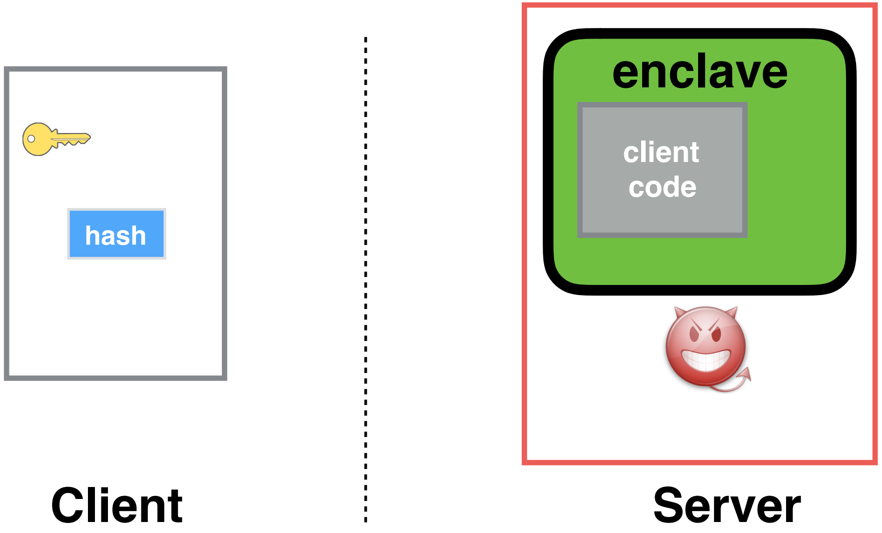
Remote attestation

Enables verifying which code runs in the enclave and performing key exchange



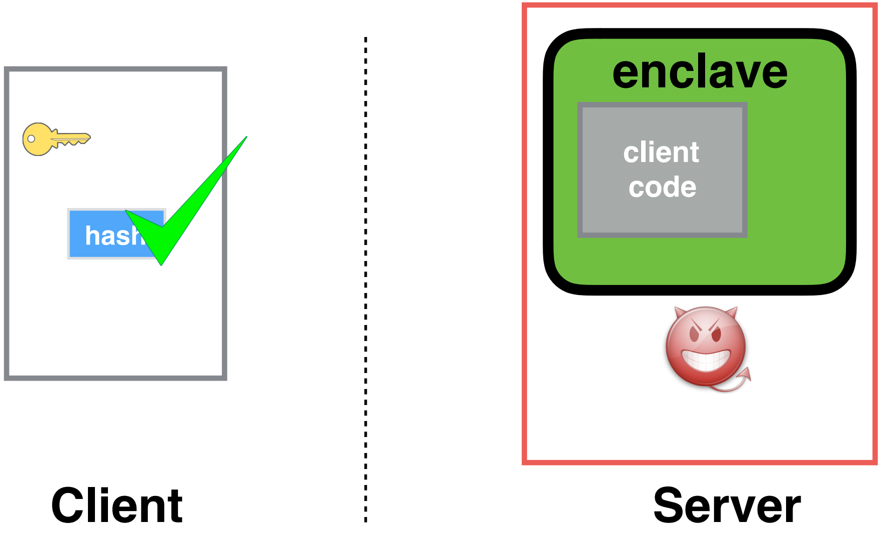
Remote attestation

Enables verifying which code runs in the enclave and performing key exchange



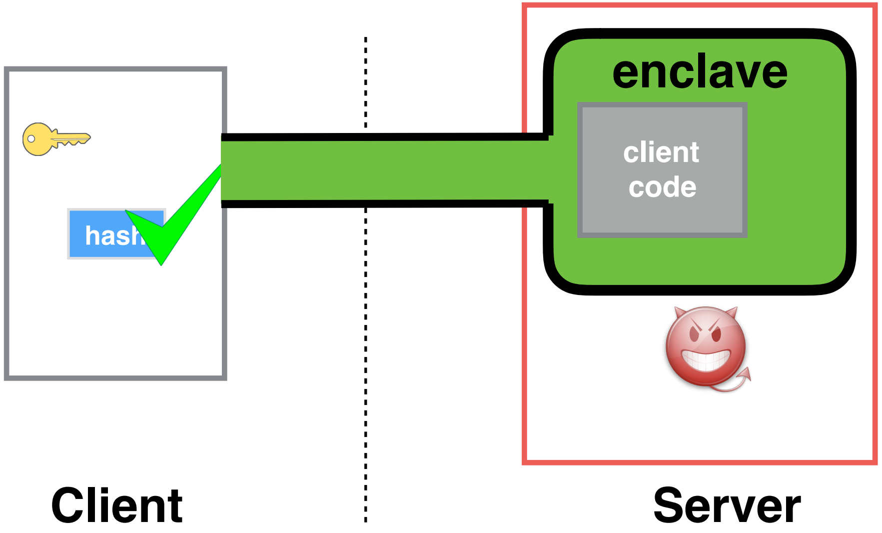
Remote attestation

Enables verifying which code runs in the enclave and performing key exchange



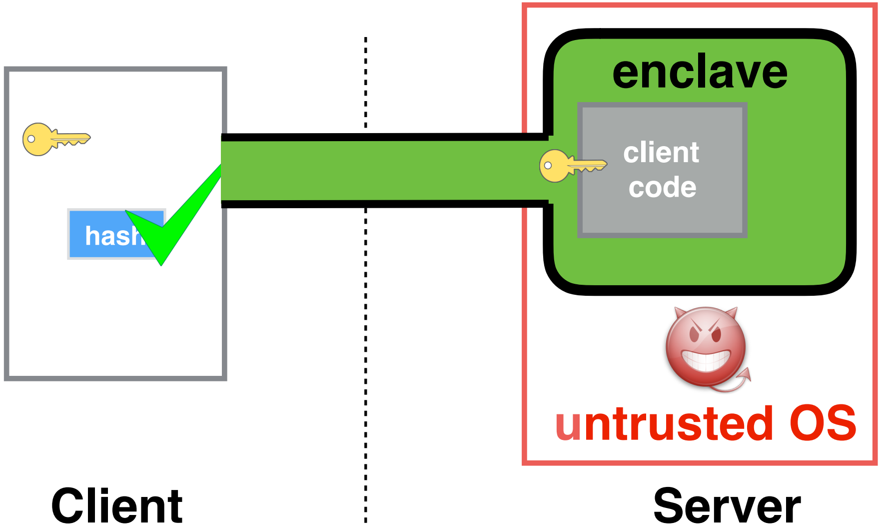
Remote attestation

Enables verifying which code runs in the enclave and performing key exchange



Remote attestation

Enables verifying which code runs in the enclave and performing key exchange



Side channels

Enclaves suffer from many side channels:

- cache-timing attacks ([Gotzfried et al17],[Brasser17,…])
- branch predictor based attacks ([Lee et al17],…)
- page fault based attacks ([Xu et al15], …)
- memory bus based attacks (Membuster[USEC20])
- dirty-bit based attacks

Side channels

Enclaves suffer from many side channels:

- cache-timing attacks ([Gotzfried et al17],[Brasser17,…])
- branch predictor based attacks ([Lee et al17],…)
- page fault based attacks ([Xu et al15], …)
- memory bus based attacks (Membuster[USEC20])
- dirty-bit based attacks

} reduce to exploit memory addresses

Side channels

Enclaves suffer from many side channels:

- cache-timing attacks ([Gotzfried et al17],[Brasser17,…])
- branch predictor based attacks ([Lee et al17],…)
- page fault based attacks ([Xu et al15], …)
- memory bus based attacks (Membuster[USEC20])
- dirty-bit based attacks

reduce to exploit memory addresses

prevented by oblivious computation

Side channels

Enclaves suffer from many side channels:

- cache-timing attacks ([Gotzfried et al17],[Brasser17,…])
- branch predictor based attacks ([Lee et al17],…)
- page fault based attacks ([Xu et al15], …)
- memory bus based attacks (Membuster[USEC20])
- dirty-bit based attacks

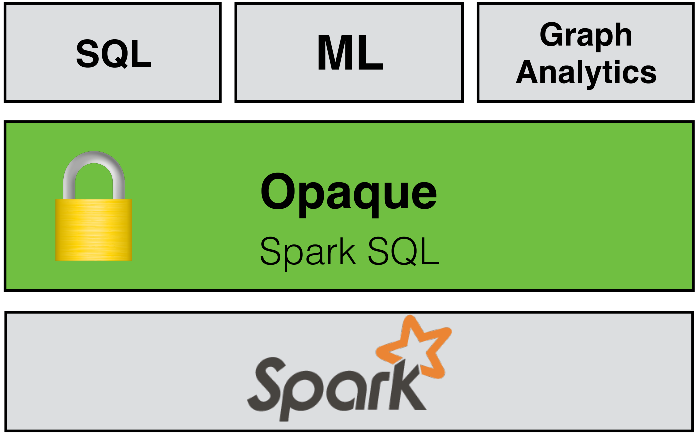
} reduce to exploit memory addresses

↓ prevented by oblivious computation

Synergy: enclaves remove expensive network communication of oblivious algorithms

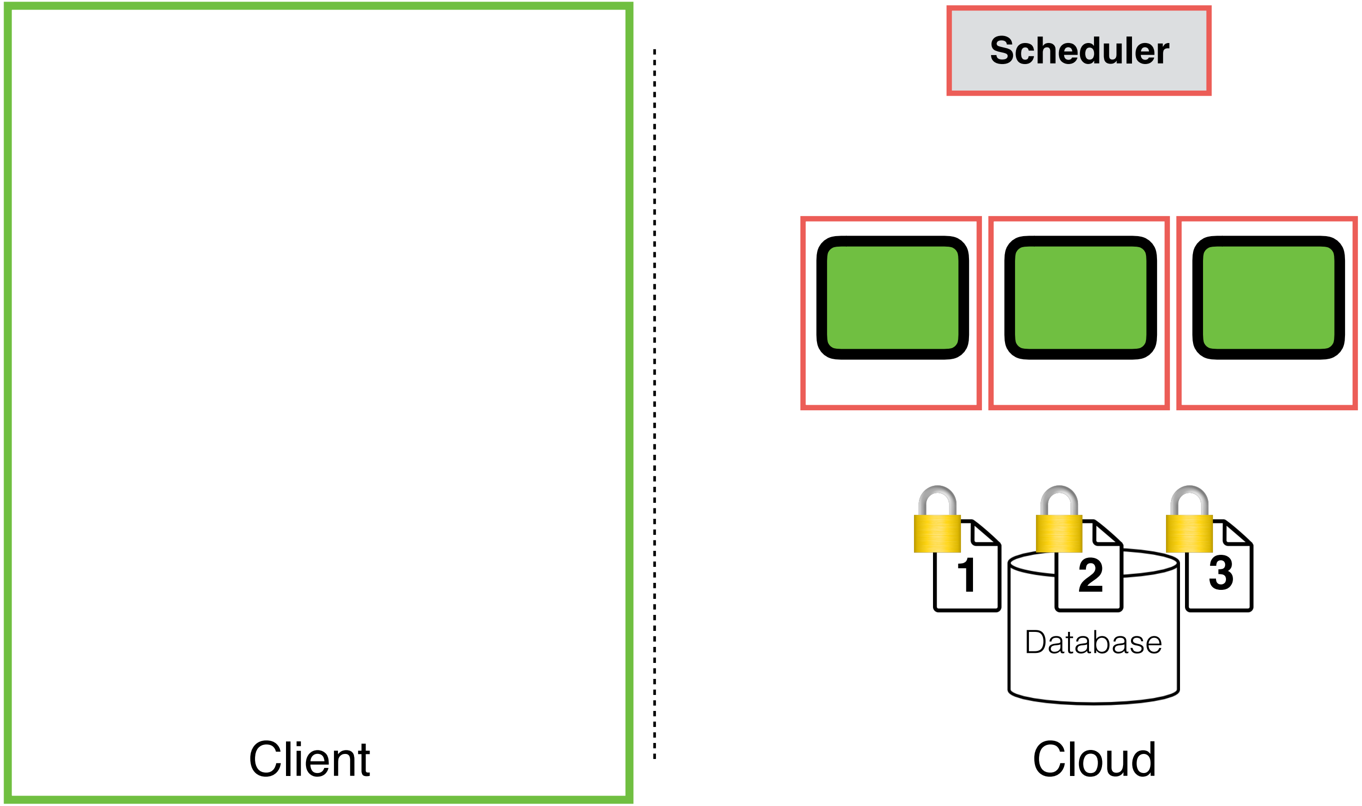
[NSDI17]

Opaque*: oblivious and encrypted distributed analytics platform

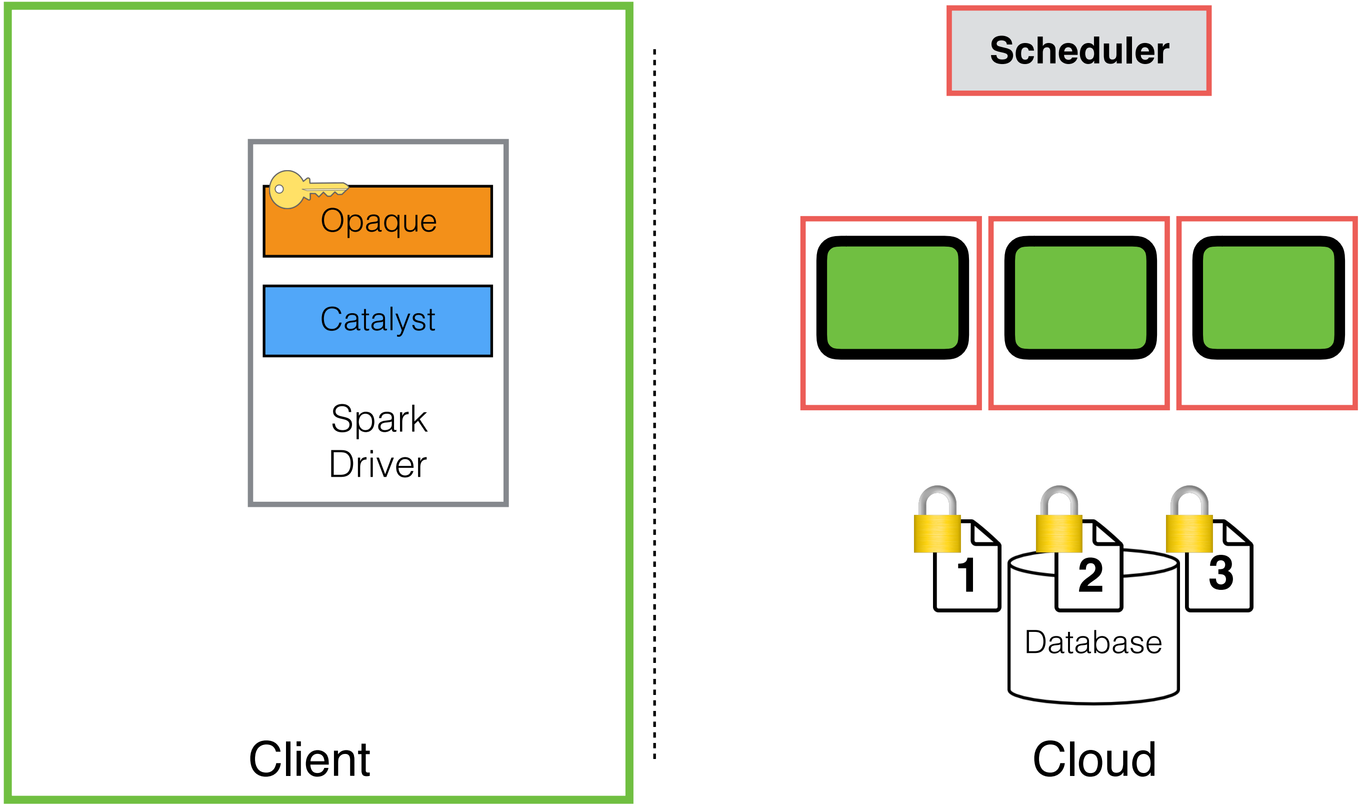


* Oblivious Platform for Analytic QUEries

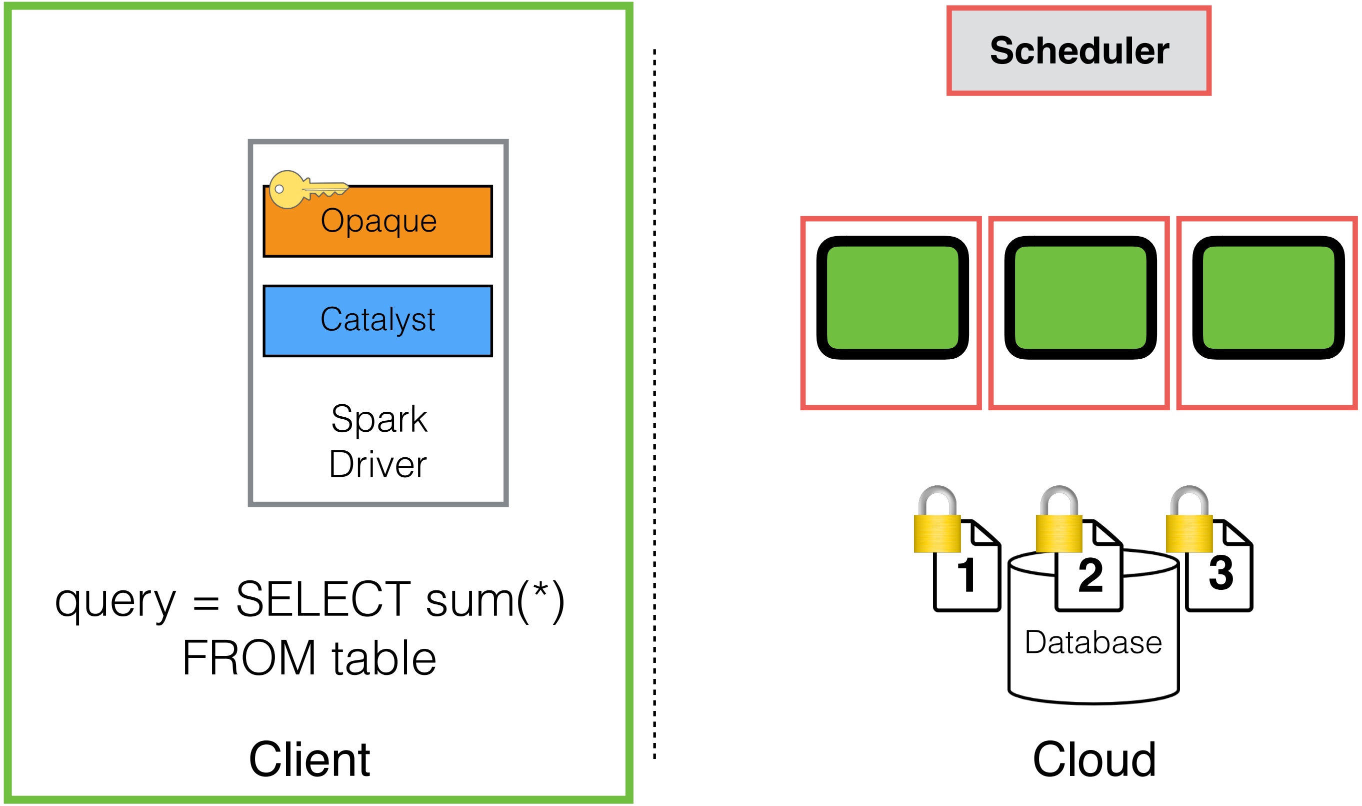
Query execution



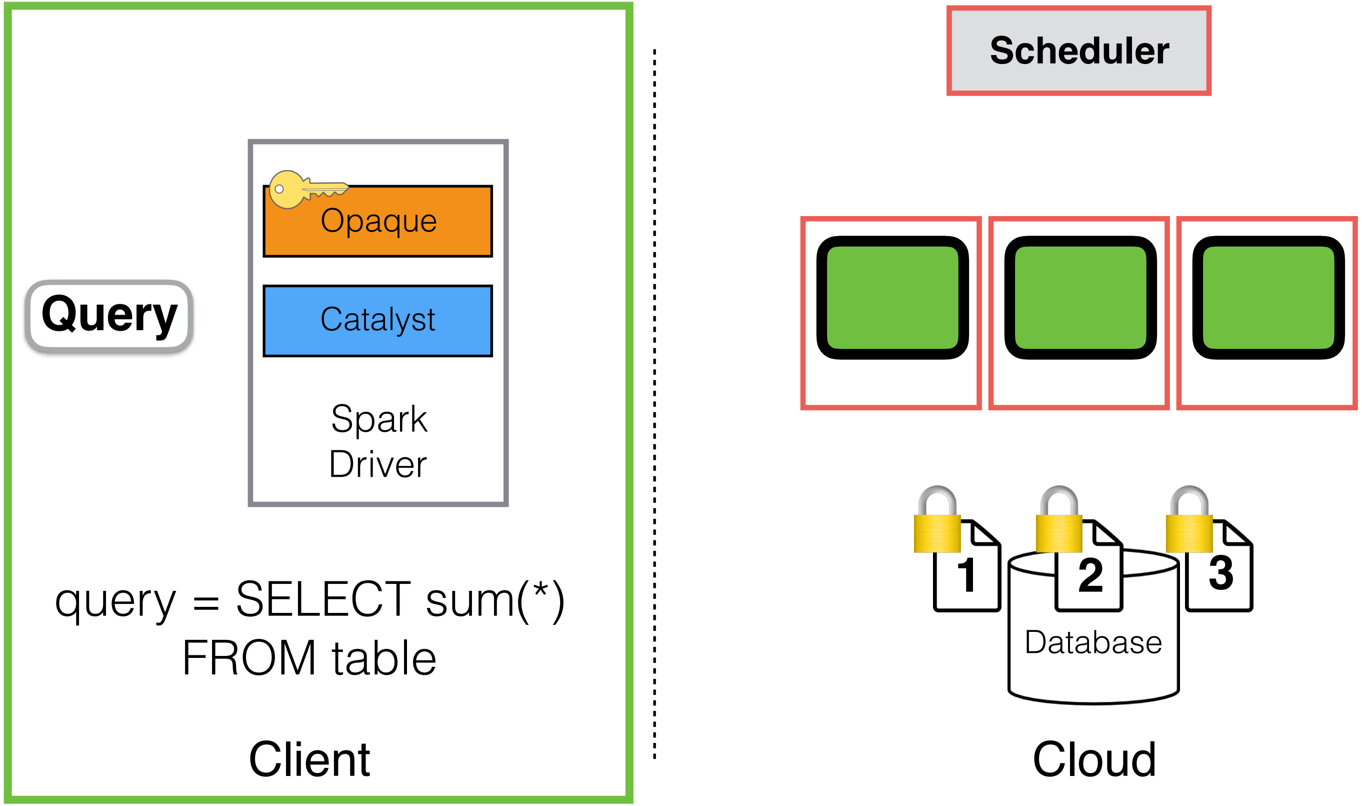
Query execution



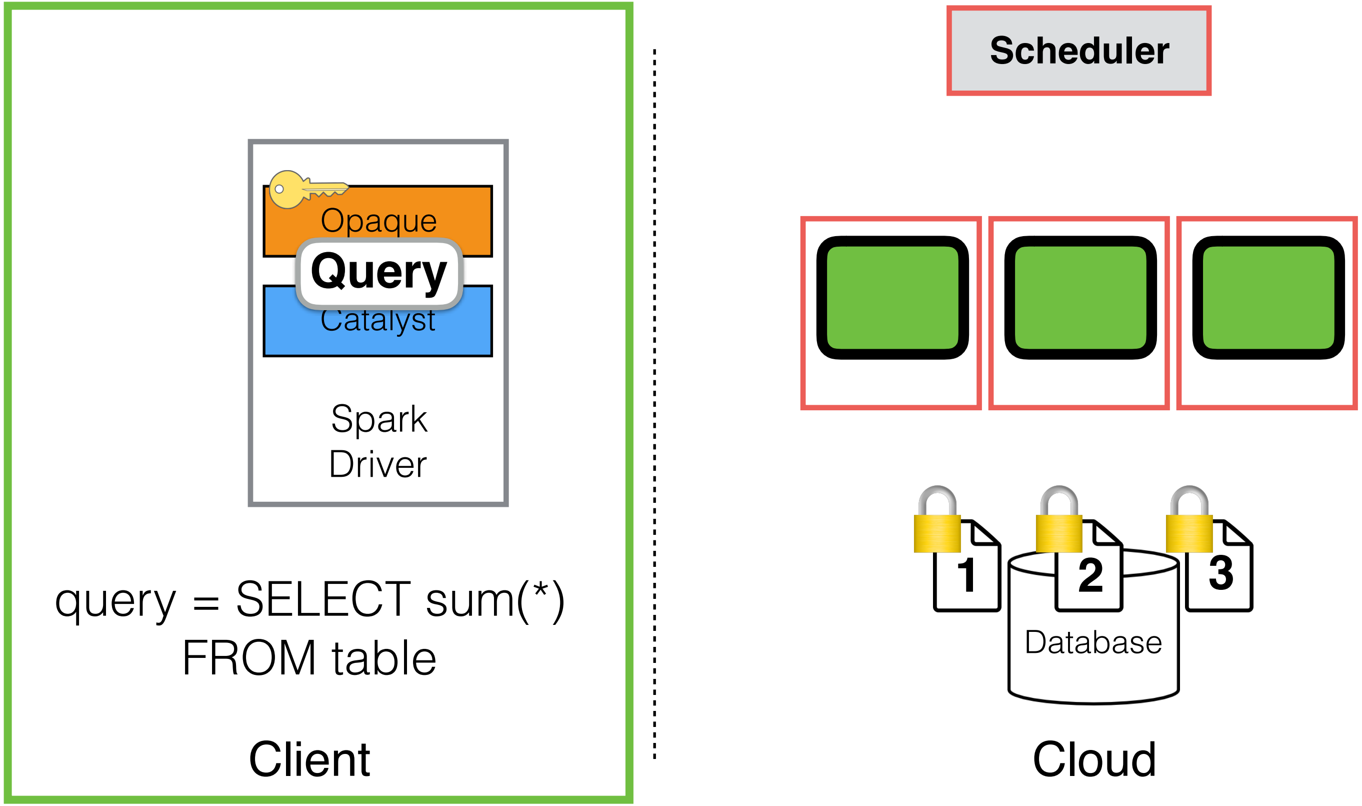
Query execution



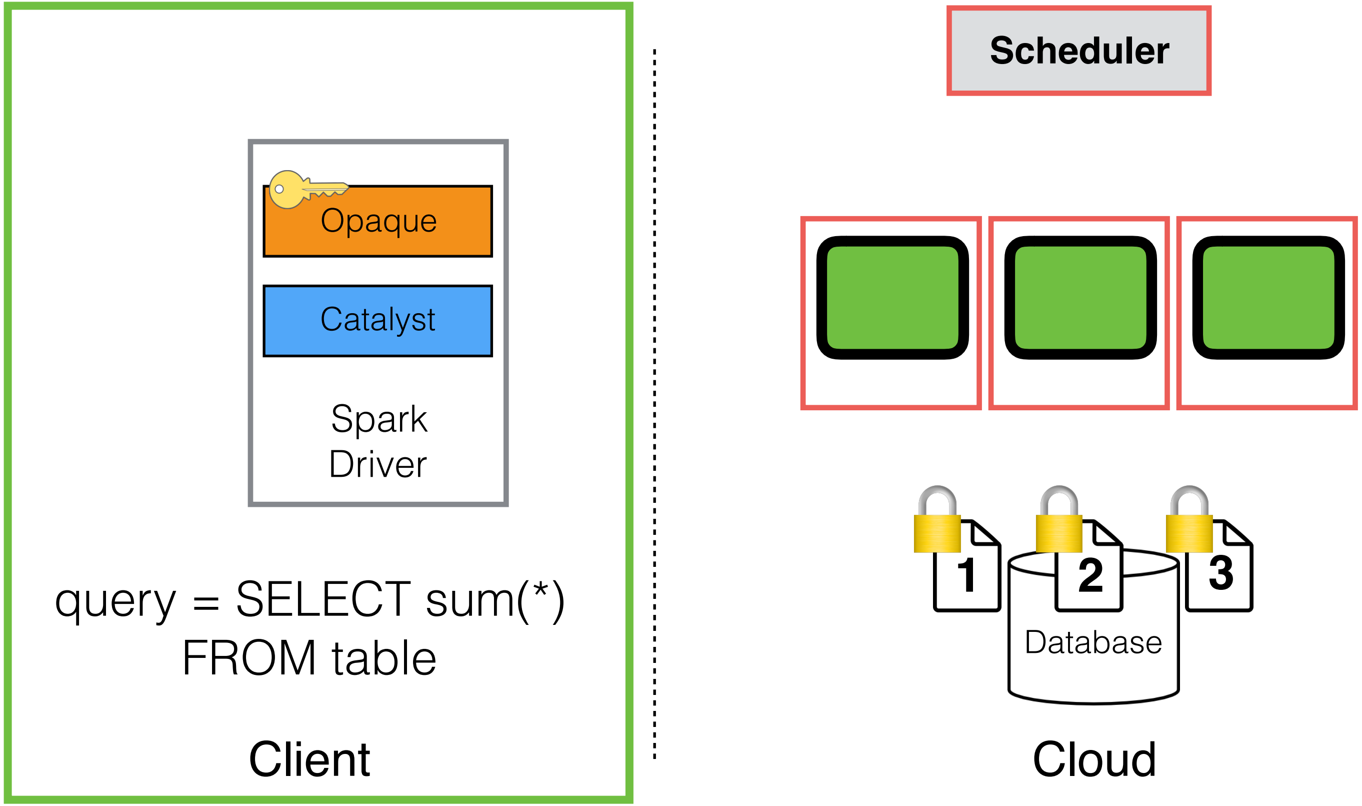
Query execution



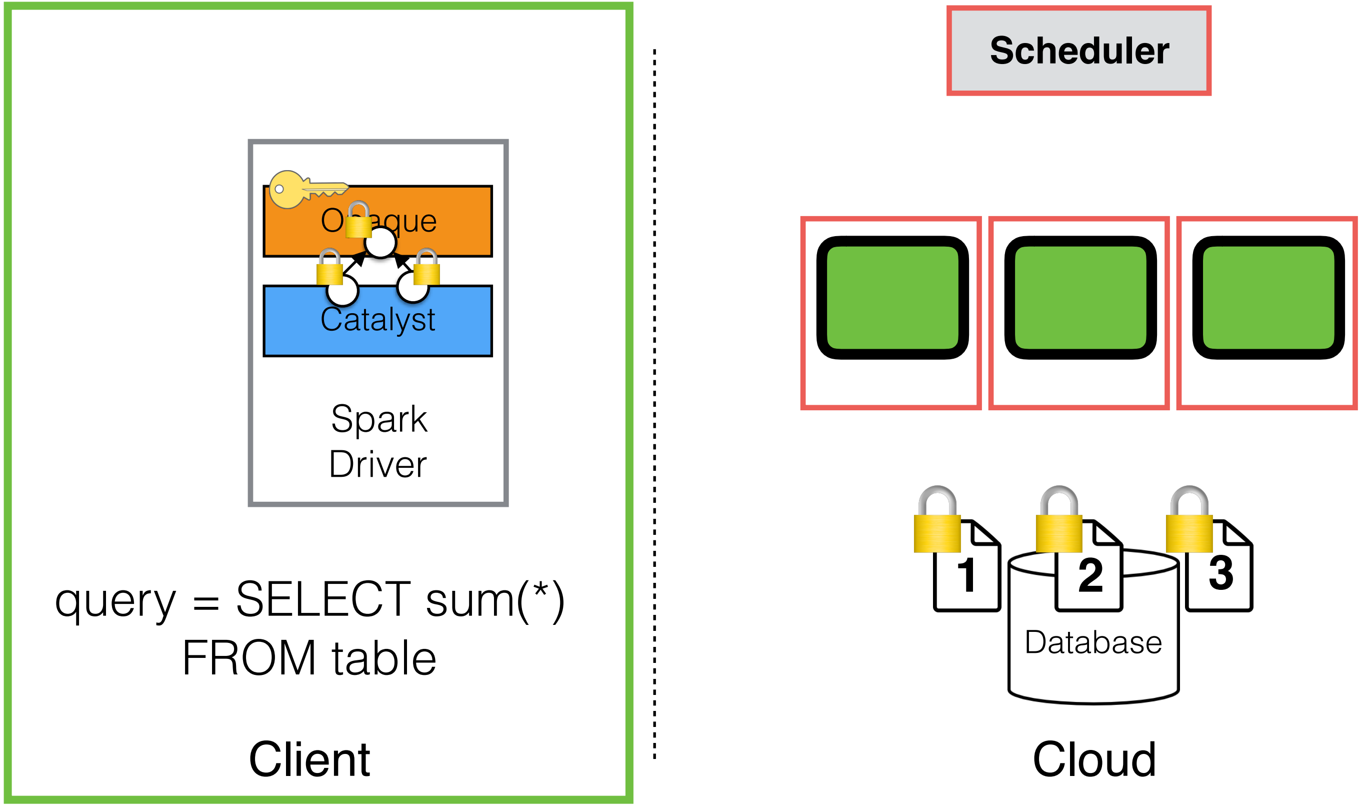
Query execution



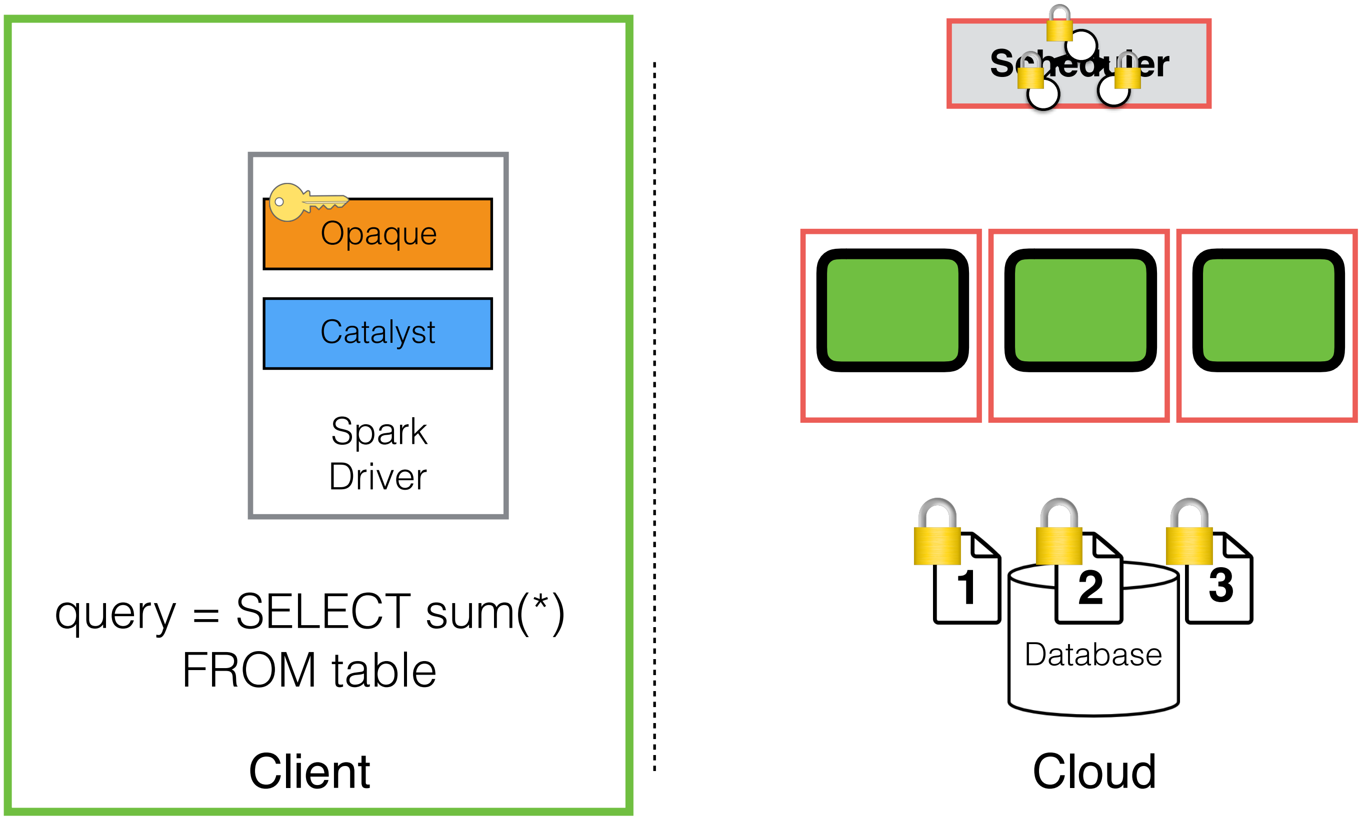
Query execution



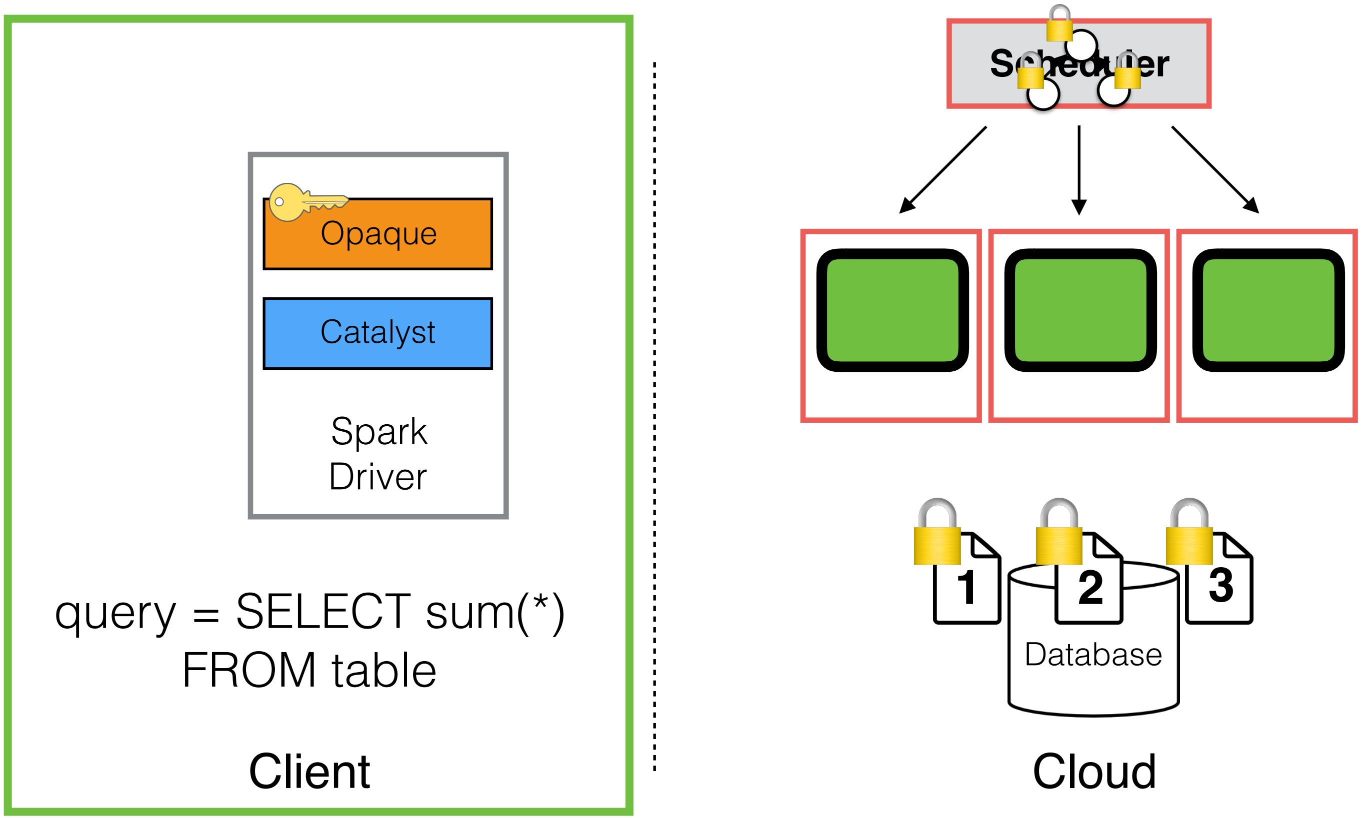
Query execution



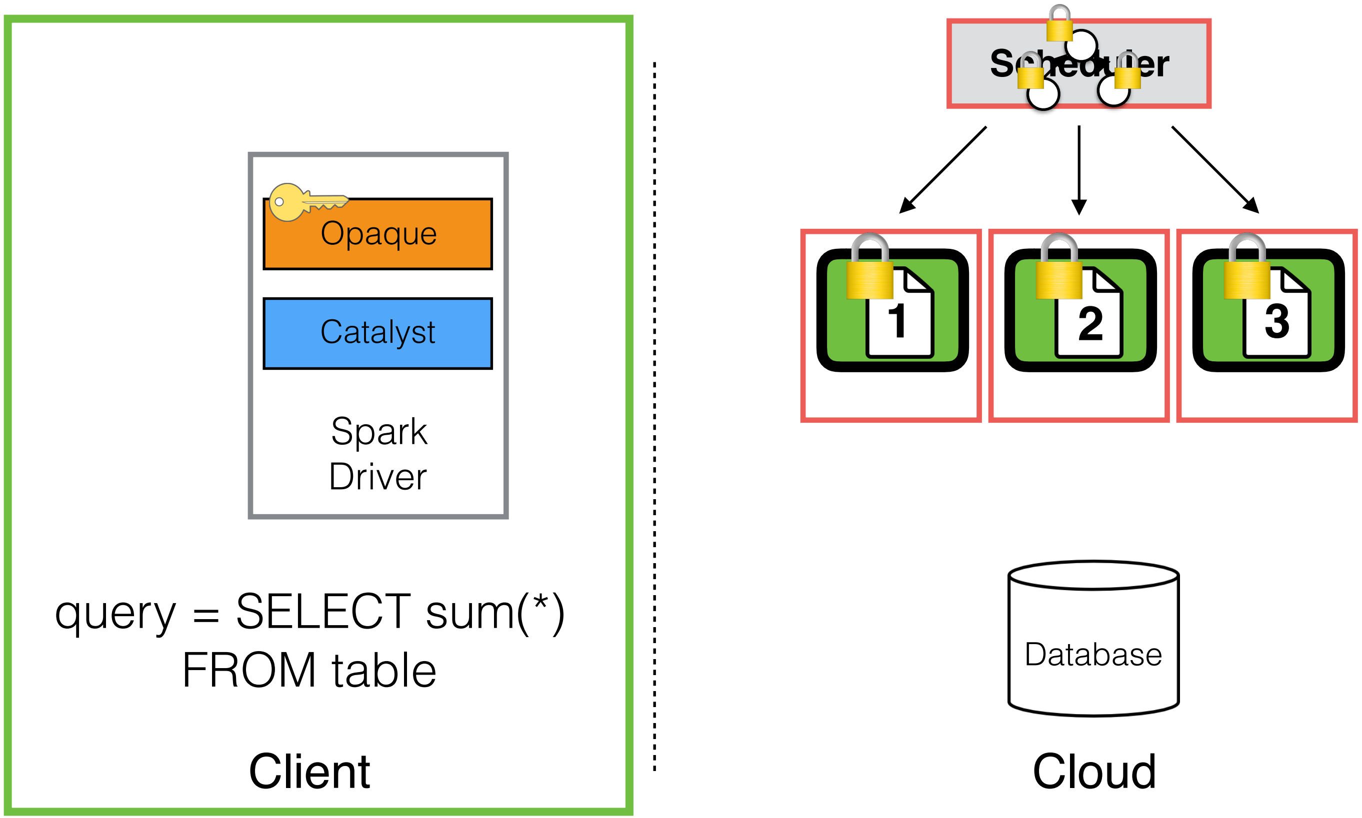
Query execution



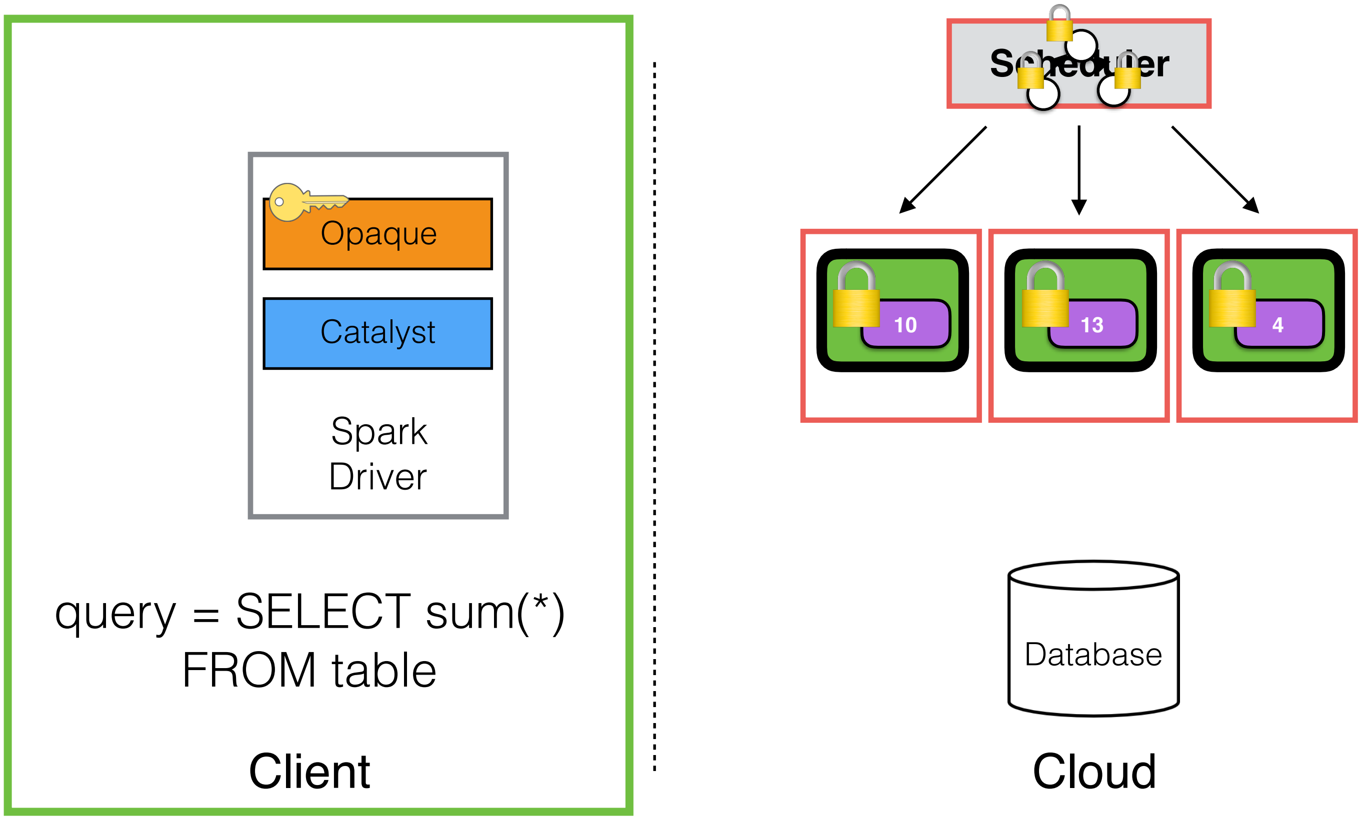
Query execution



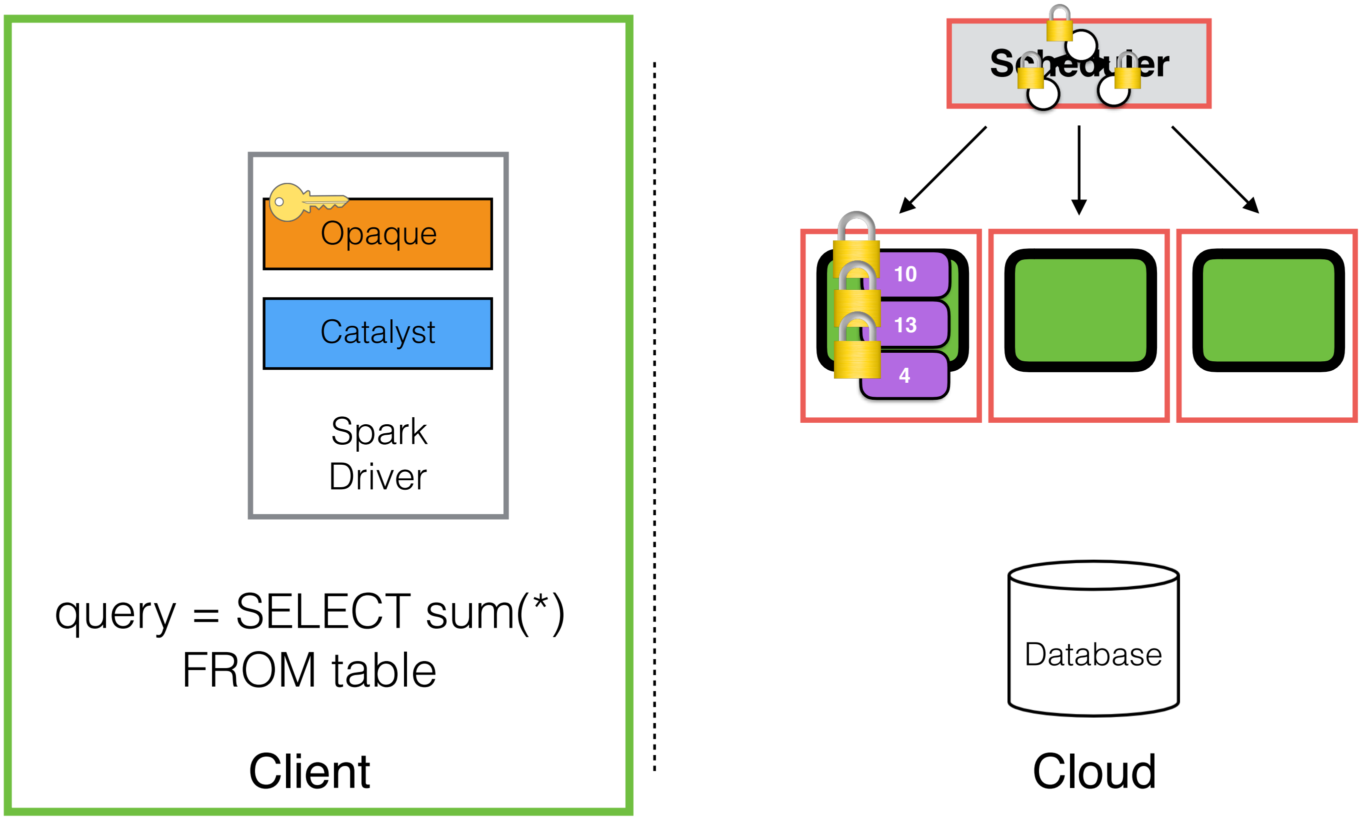
Query execution



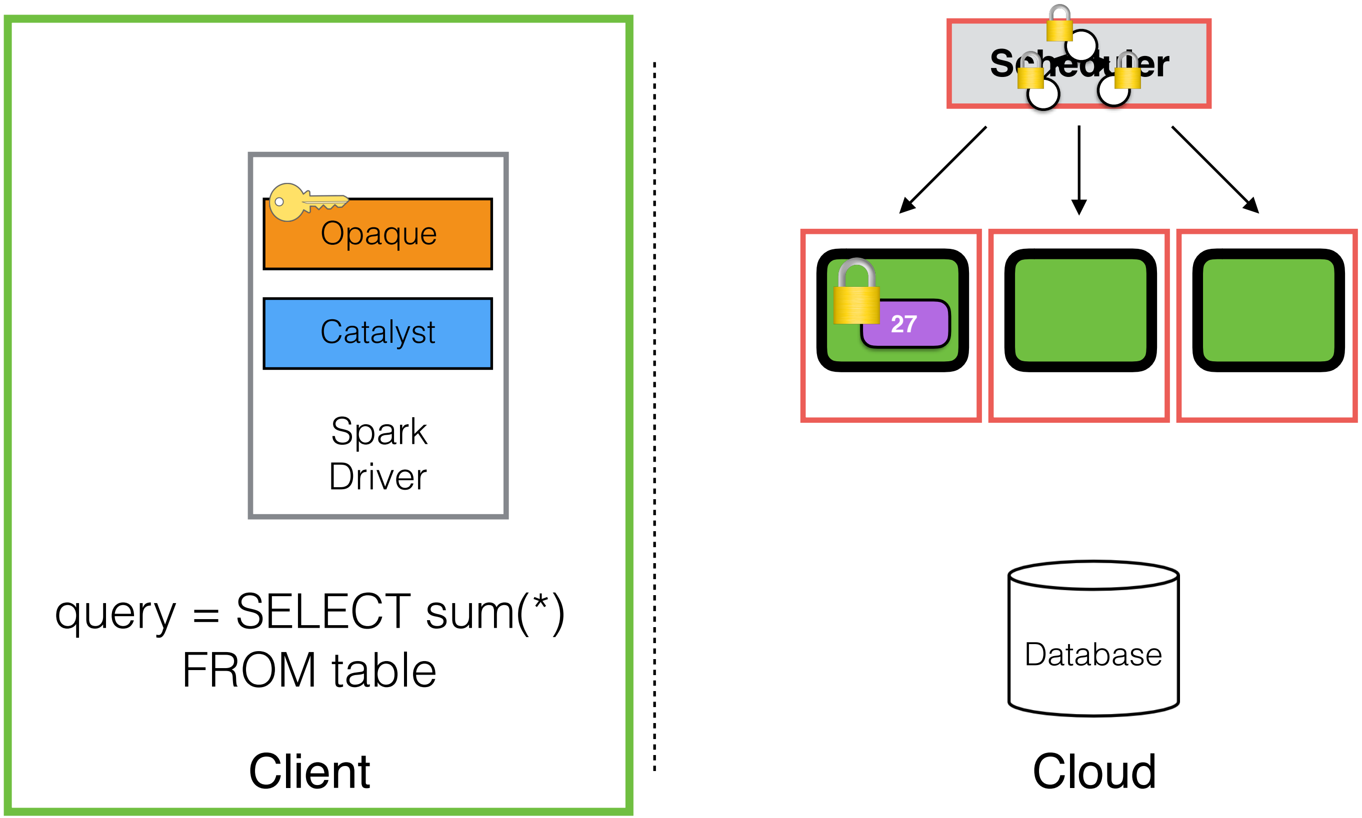
Query execution



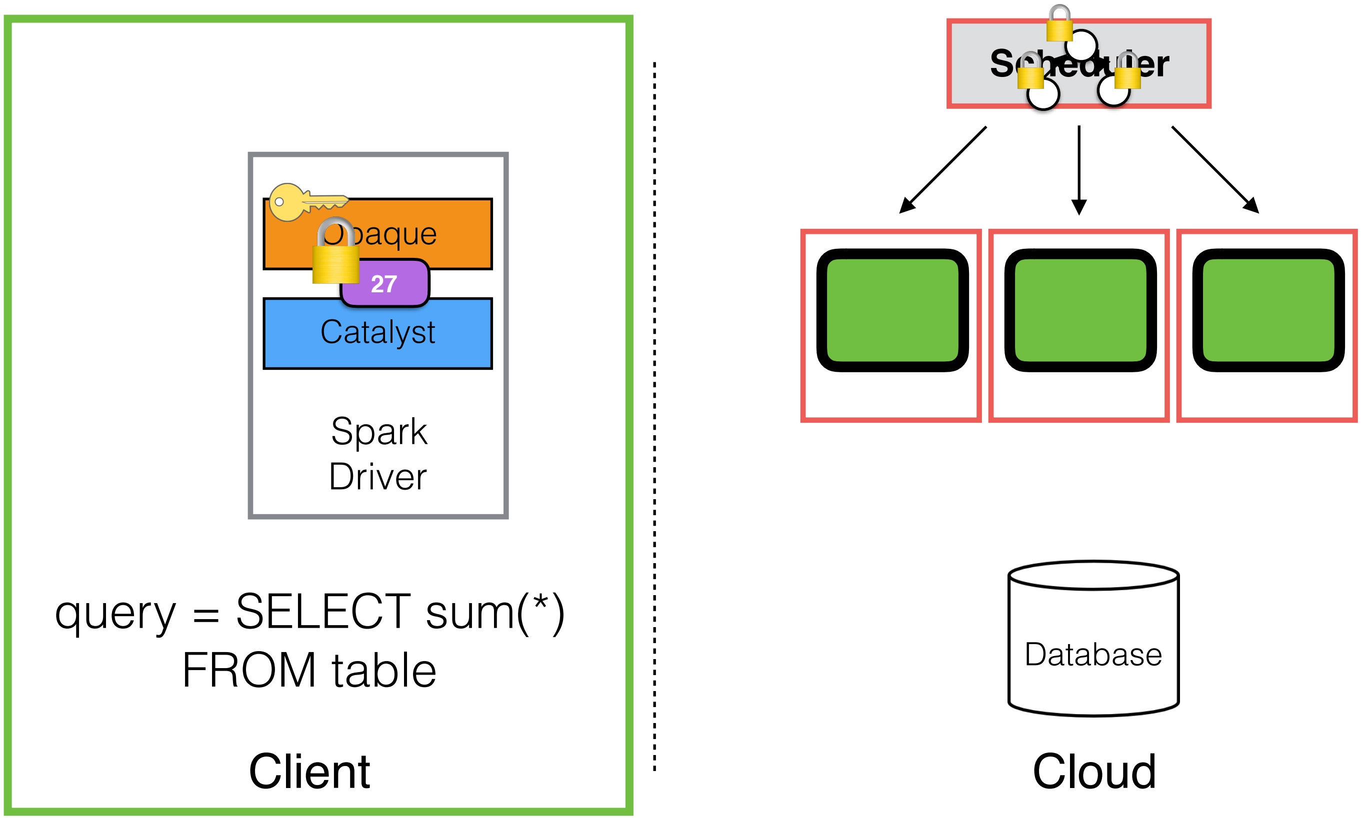
Query execution



Query execution



Query execution



Opaque components

Opaque components

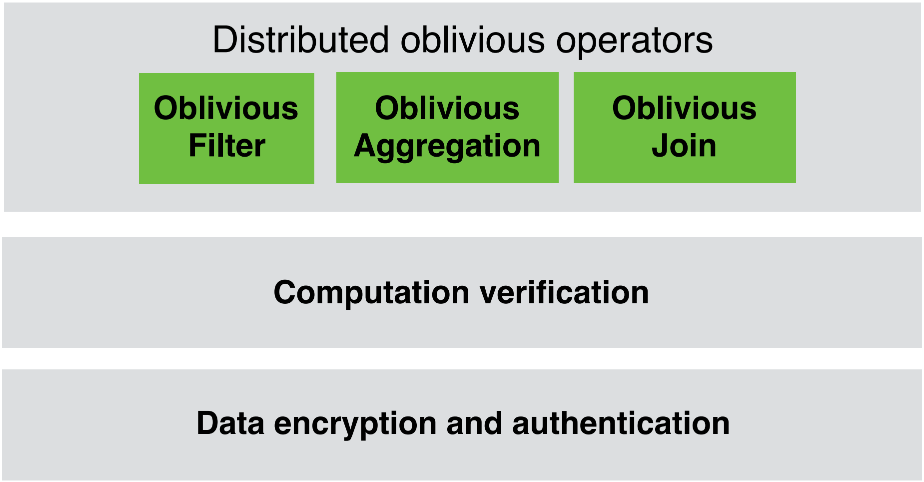
Data encryption and authentication

Opaque components

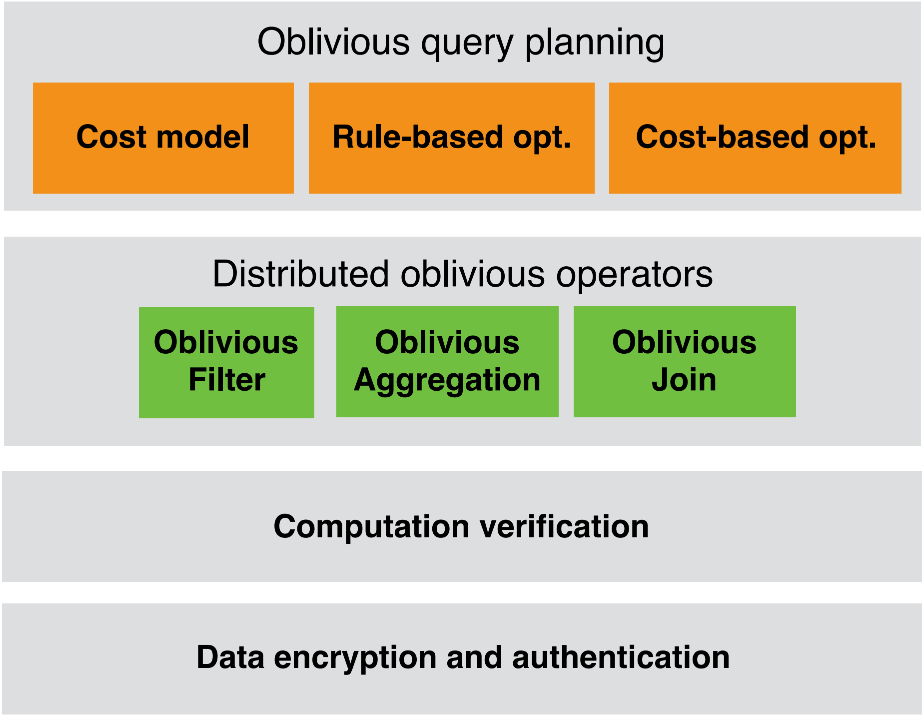
Computation verification

Data encryption and authentication

Opaque components



Opaque components

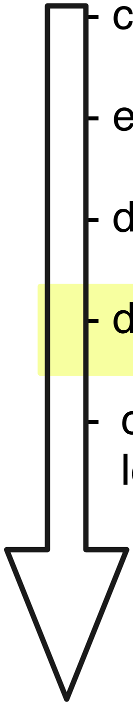


Open source

https://github.com/ucbrise/opaque

Adoption: IBM RestAssured, Ericsson, Alibaba

Systems in the cloud



- chat/messaging
- email, file sharing
- database (OLTP)
- database (analytics)
- collaborative machine learning

complexity

Systems in the cloud

- chat/messaging
- email, file sharing
- database (OLTP)
- database (analytics)
- collaborative machine learning

complexity

Money laundering detection

Money laundering detection

- Bank wants to detect money laundering using machine learning

Money laundering detection

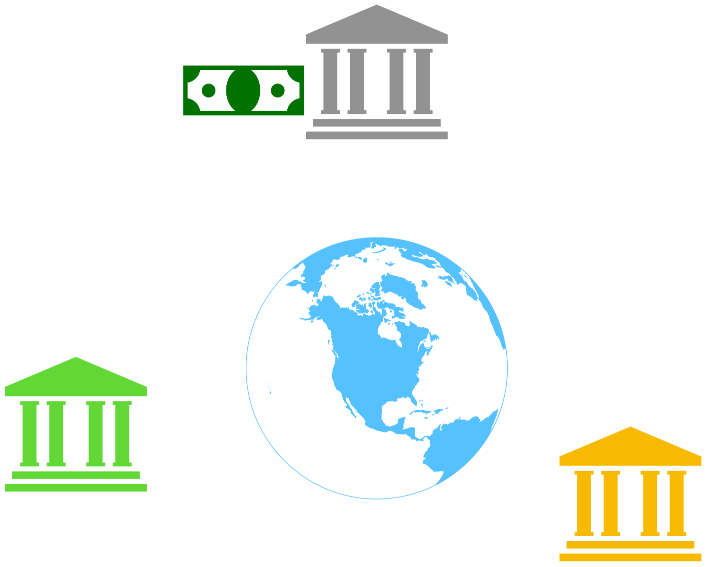
- Bank wants to detect money laundering using machine learning

Money laundering detection

- Bank wants to detect money laundering using machine learning
- Criminals conceal illegal activities across many banks

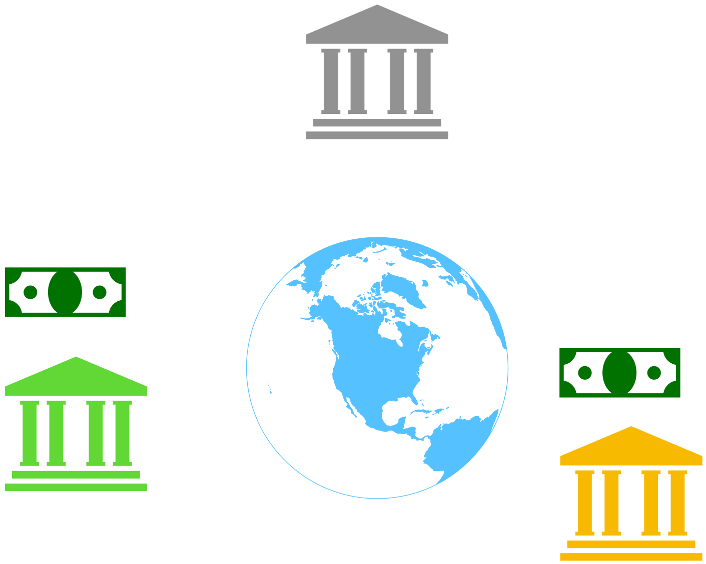
Money laundering detection

- Bank wants to detect money laundering using machine learning
- Criminals conceal illegal activities across many banks



Money laundering detection

- Bank wants to detect money laundering using machine learning
- Criminals conceal illegal activities across many banks



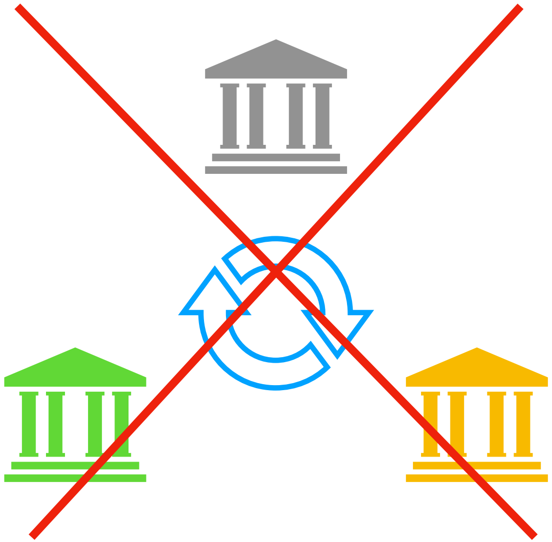
Money laundering detection

Money laundering detection

- Want to jointly compute a model on customer transaction data across many banks

Money laundering detection

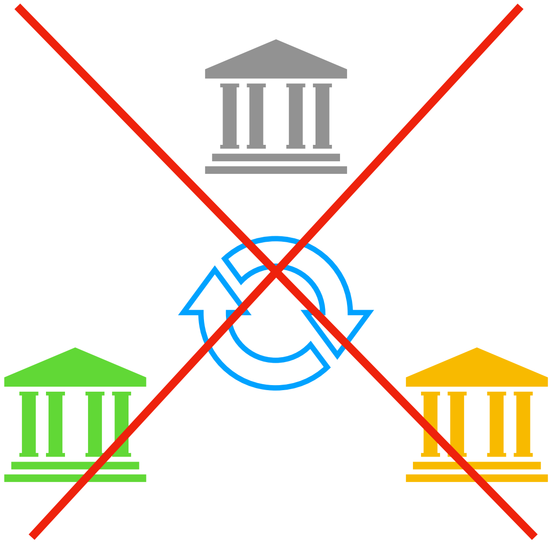
- Want to jointly compute a model on customer transaction data across many banks



Money laundering detection

- Want to jointly compute a model on customer transaction data across many banks

- **Cannot share data because these banks are competing with each other**



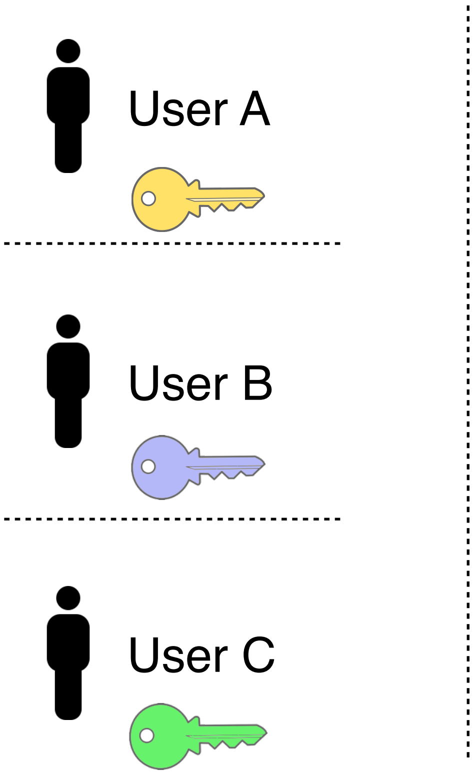
Two approaches

Two approaches

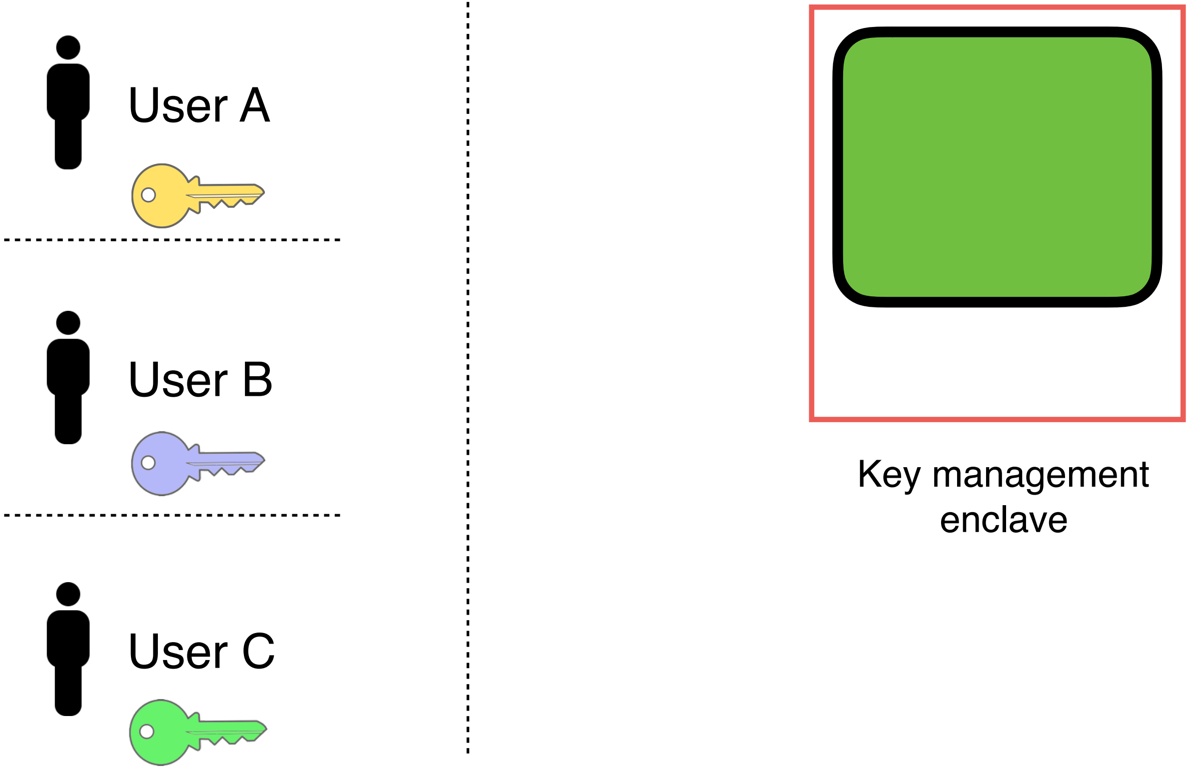
A different setup tradeoff:

- Hardware enclaves + oblivious algorithms
- Secure multi-party computation

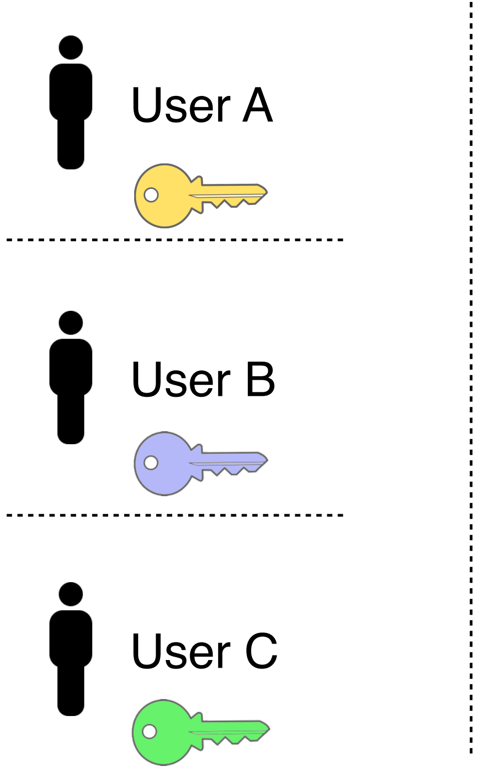
Secure collaborative ML via enclaves

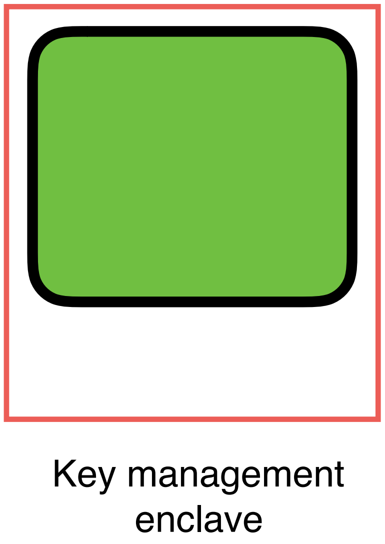


Secure collaborative ML via enclaves



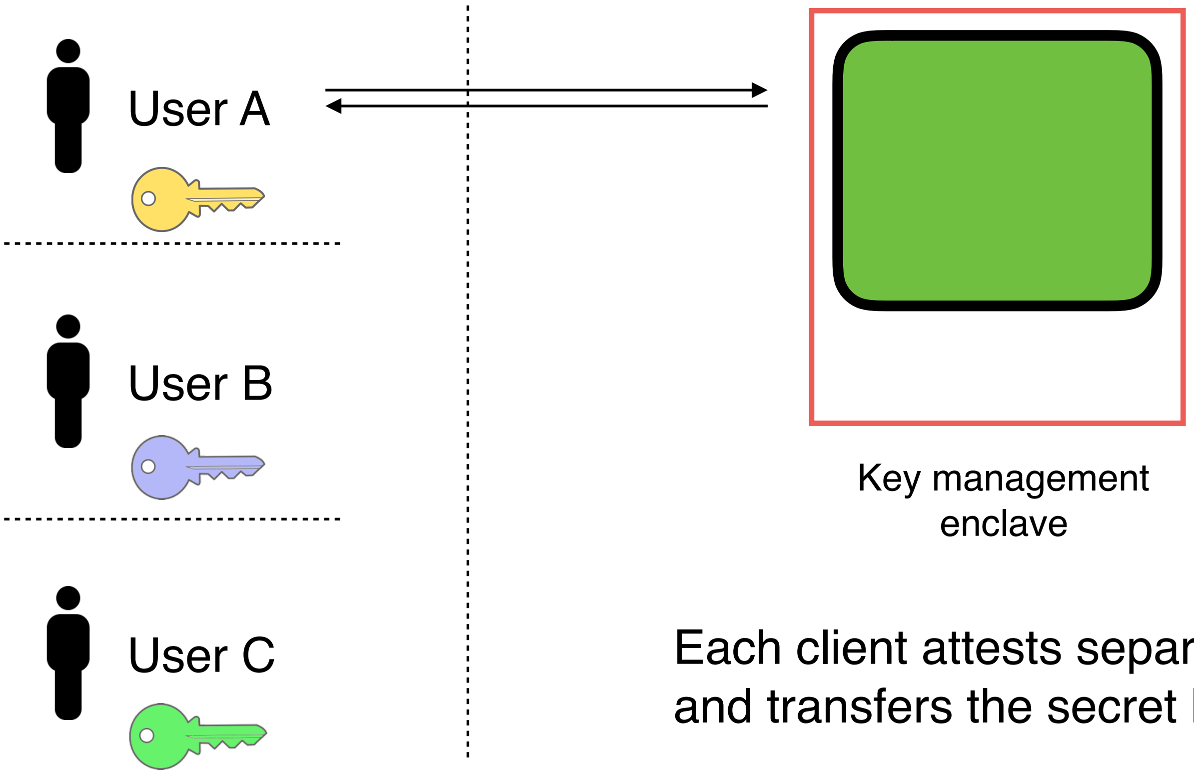
Secure collaborative ML via enclaves





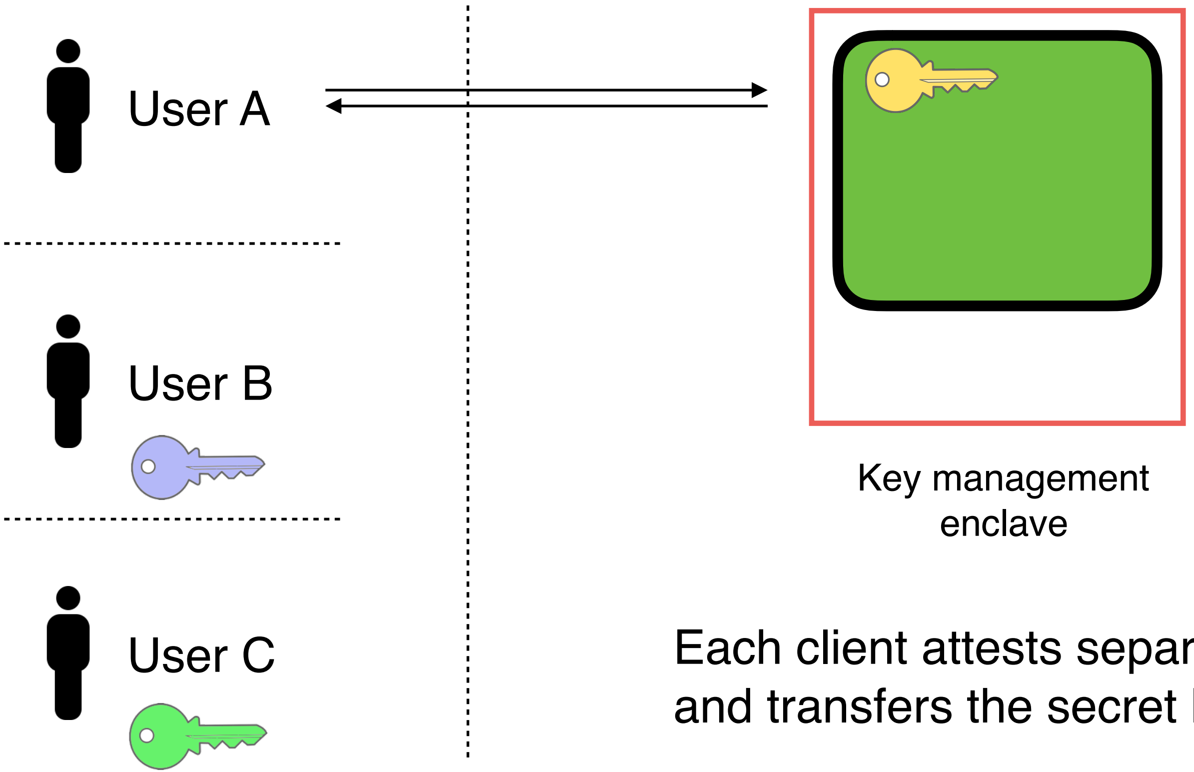
Each client attests separately
and transfers the secret key

Secure collaborative ML via enclaves



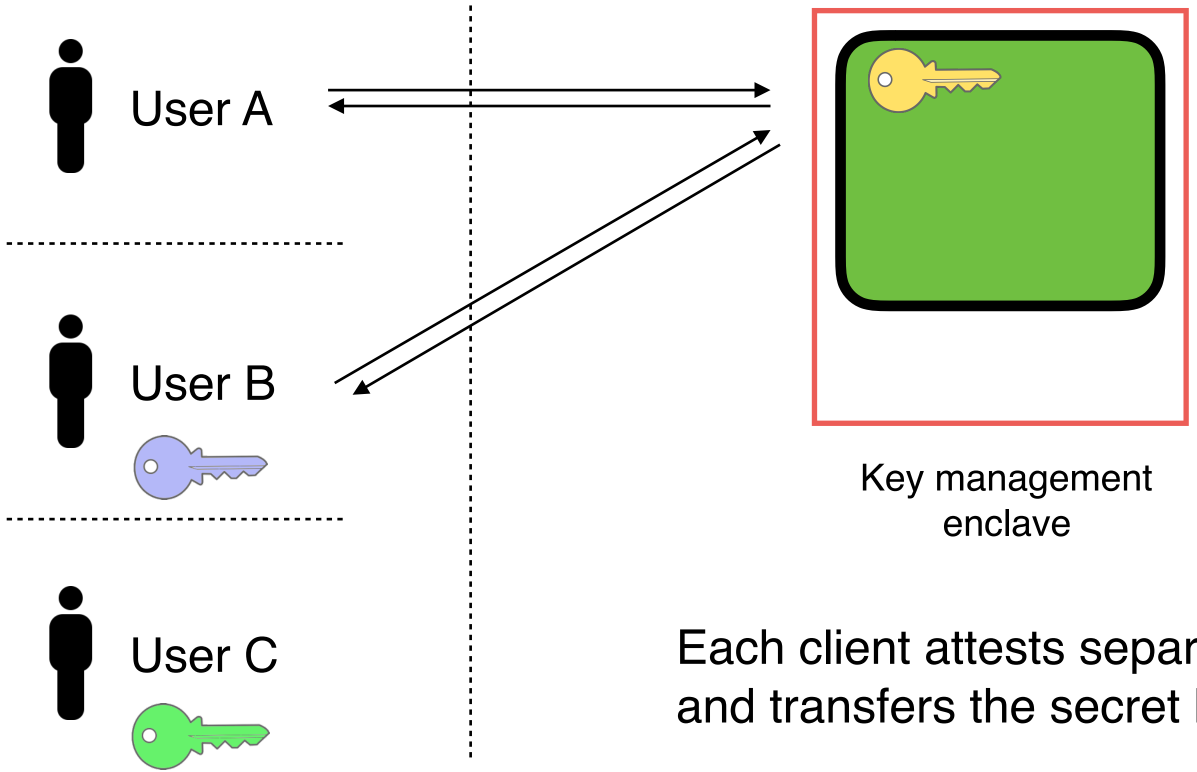
Each client attests separately
and transfers the secret key

Secure collaborative ML via enclaves



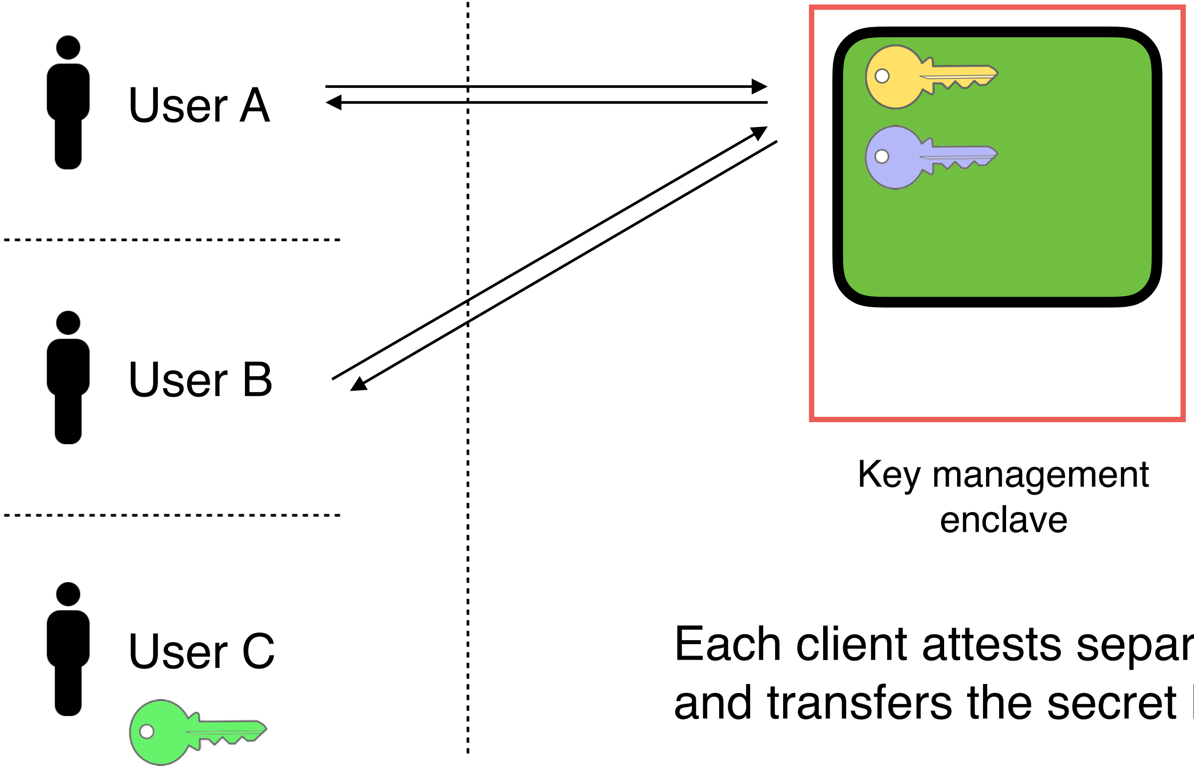
Each client attests separately
and transfers the secret key

Secure collaborative ML via enclaves



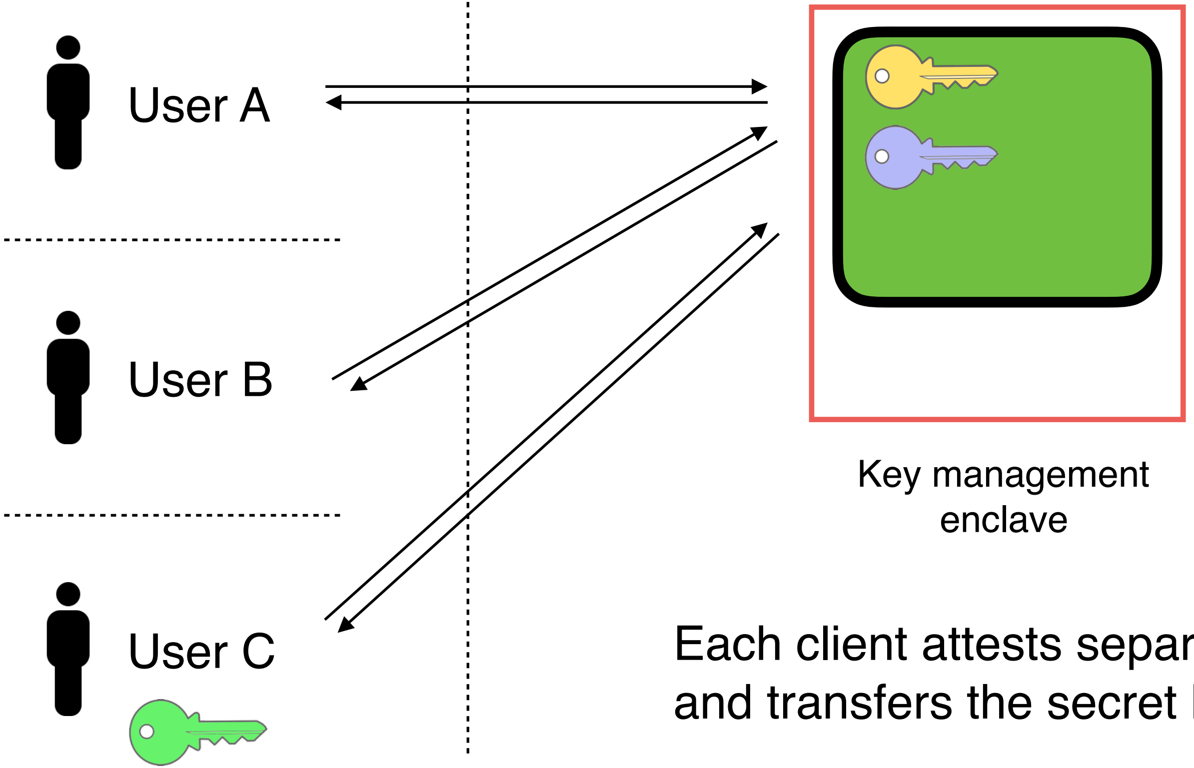
Each client attests separately
and transfers the secret key

Secure collaborative ML via enclaves



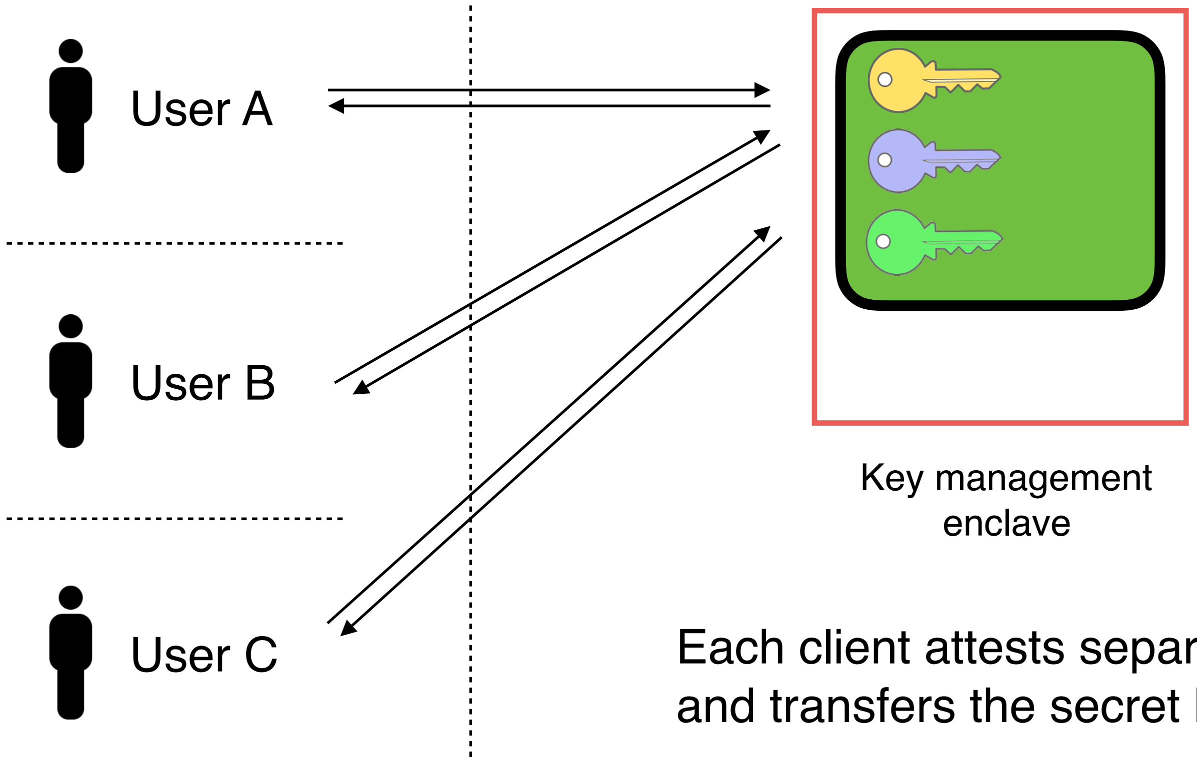
Each client attests separately and transfers the secret key

Secure collaborative ML via enclaves



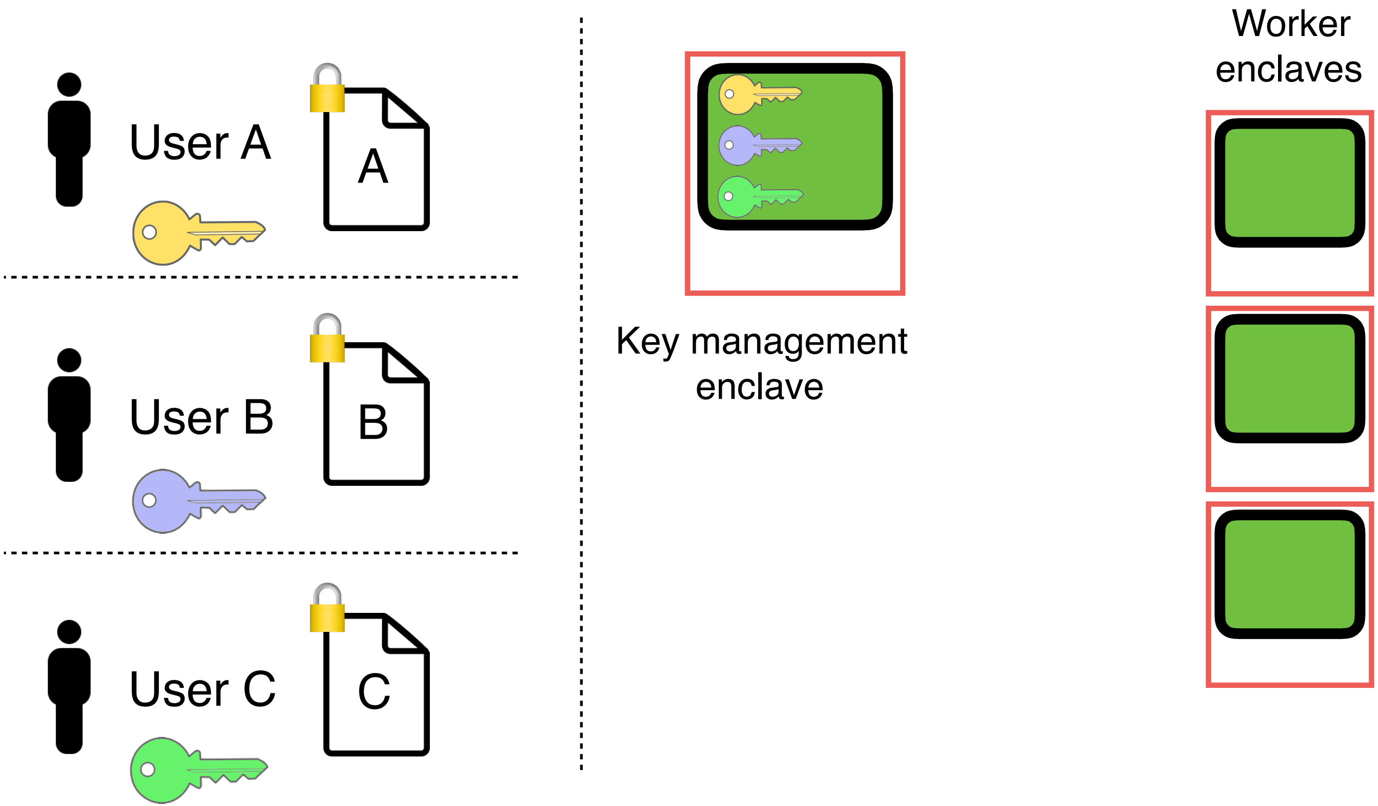
Each client attests separately and transfers the secret key

Secure collaborative ML via enclaves

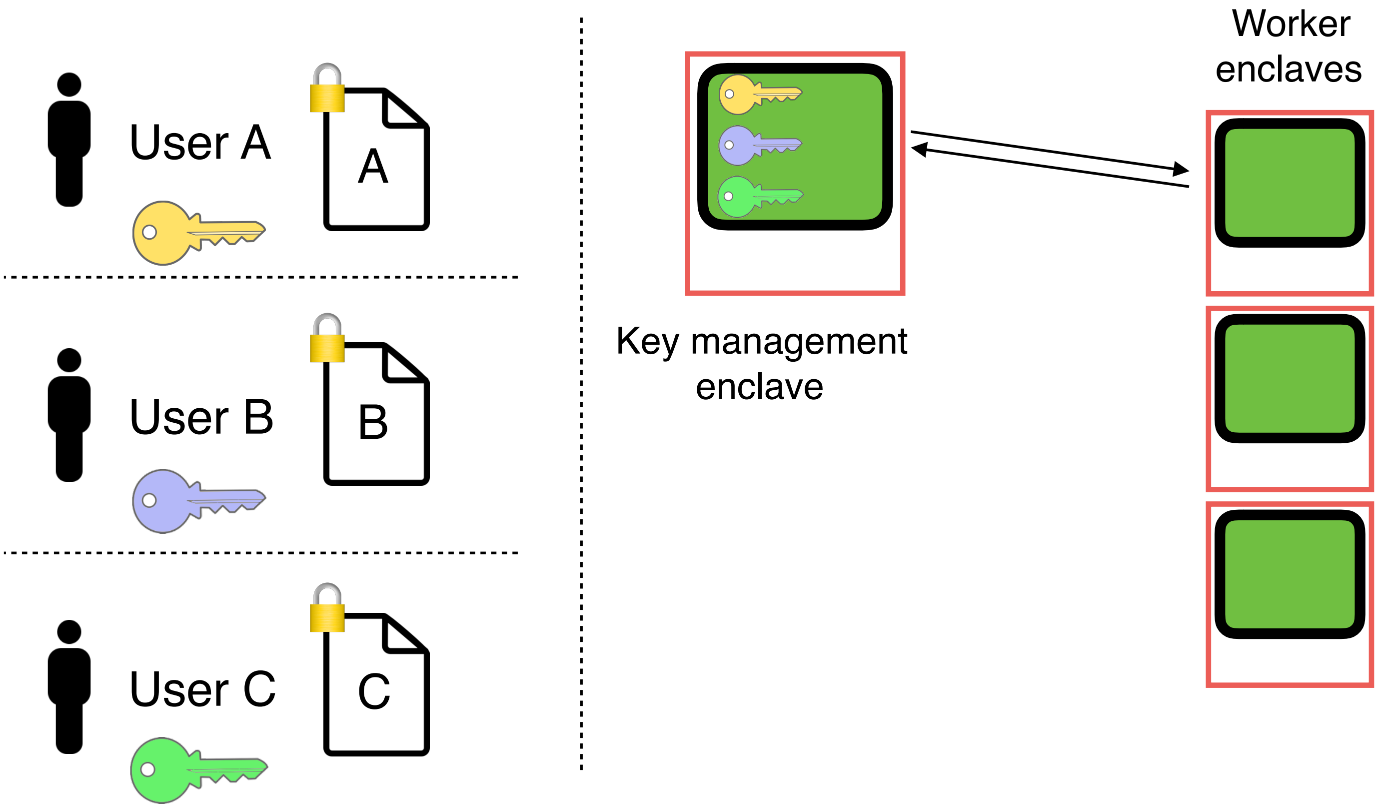


Each client attests separately
and transfers the secret key

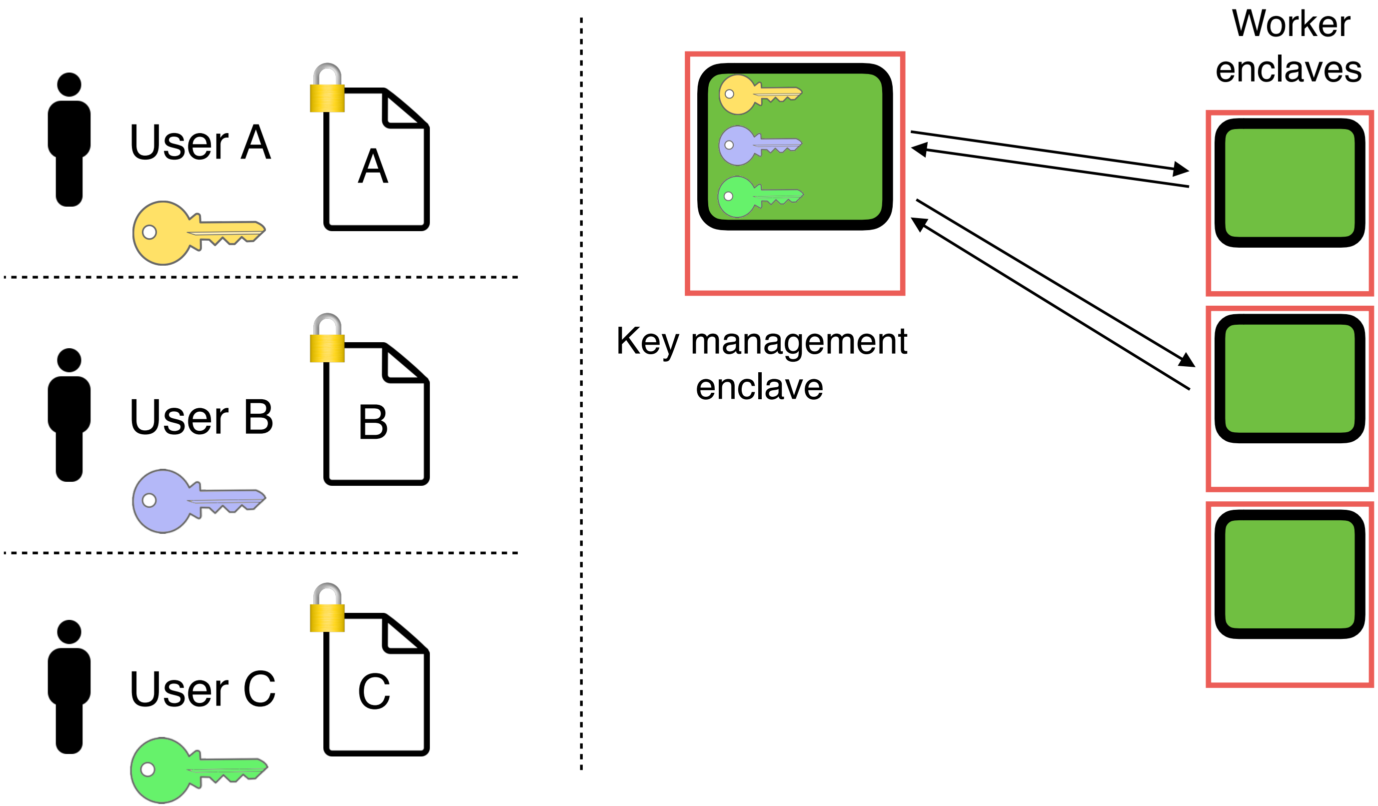
Secure collaborative ML via enclaves



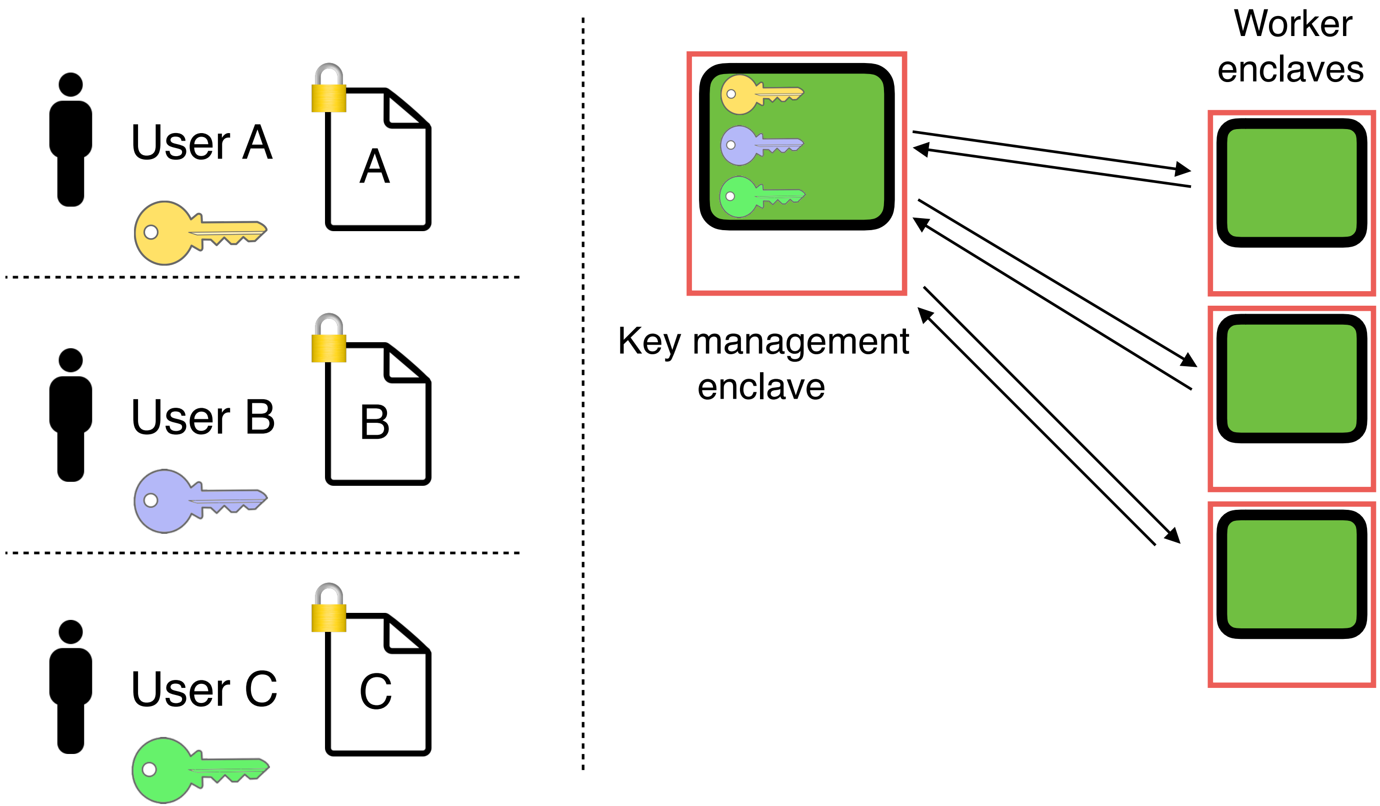
Secure collaborative ML via enclaves



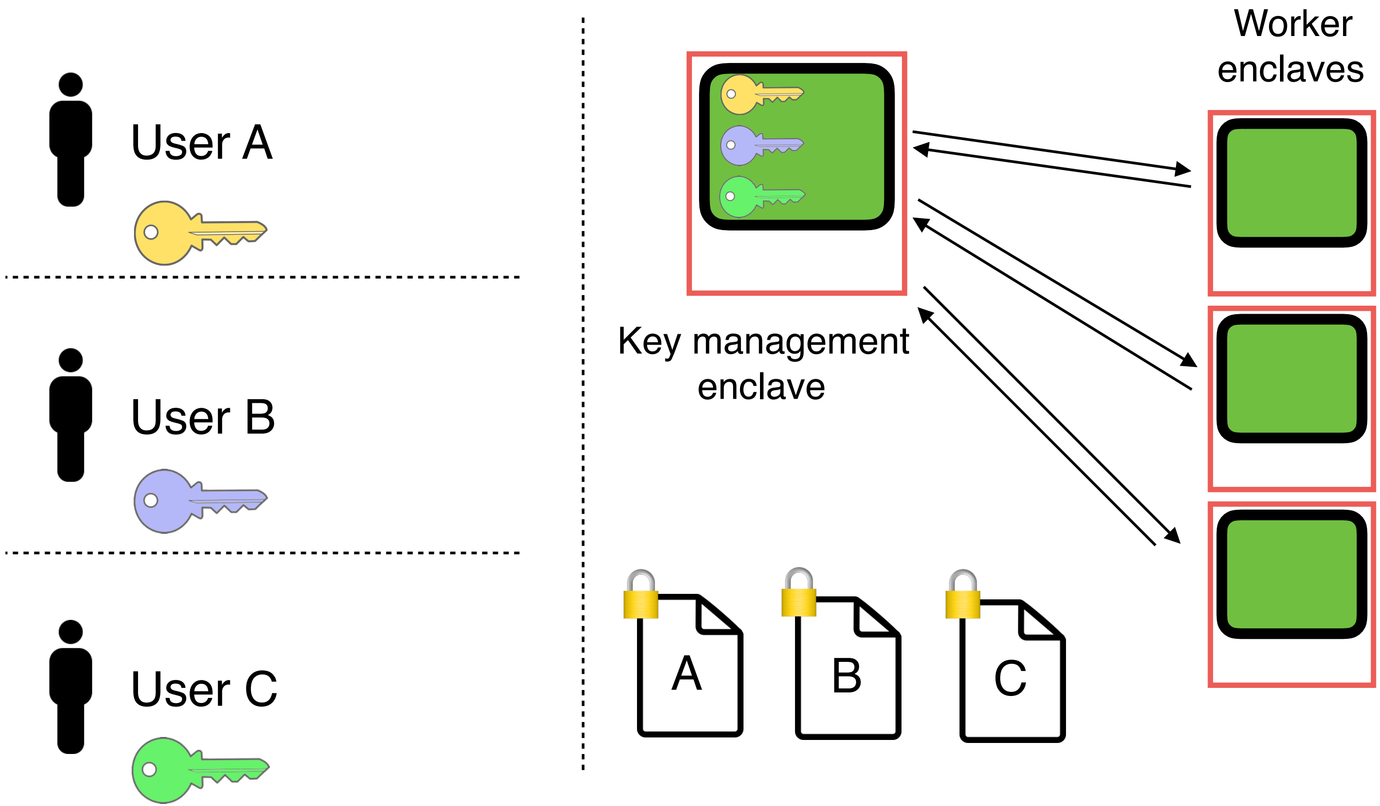
Secure collaborative ML via enclaves



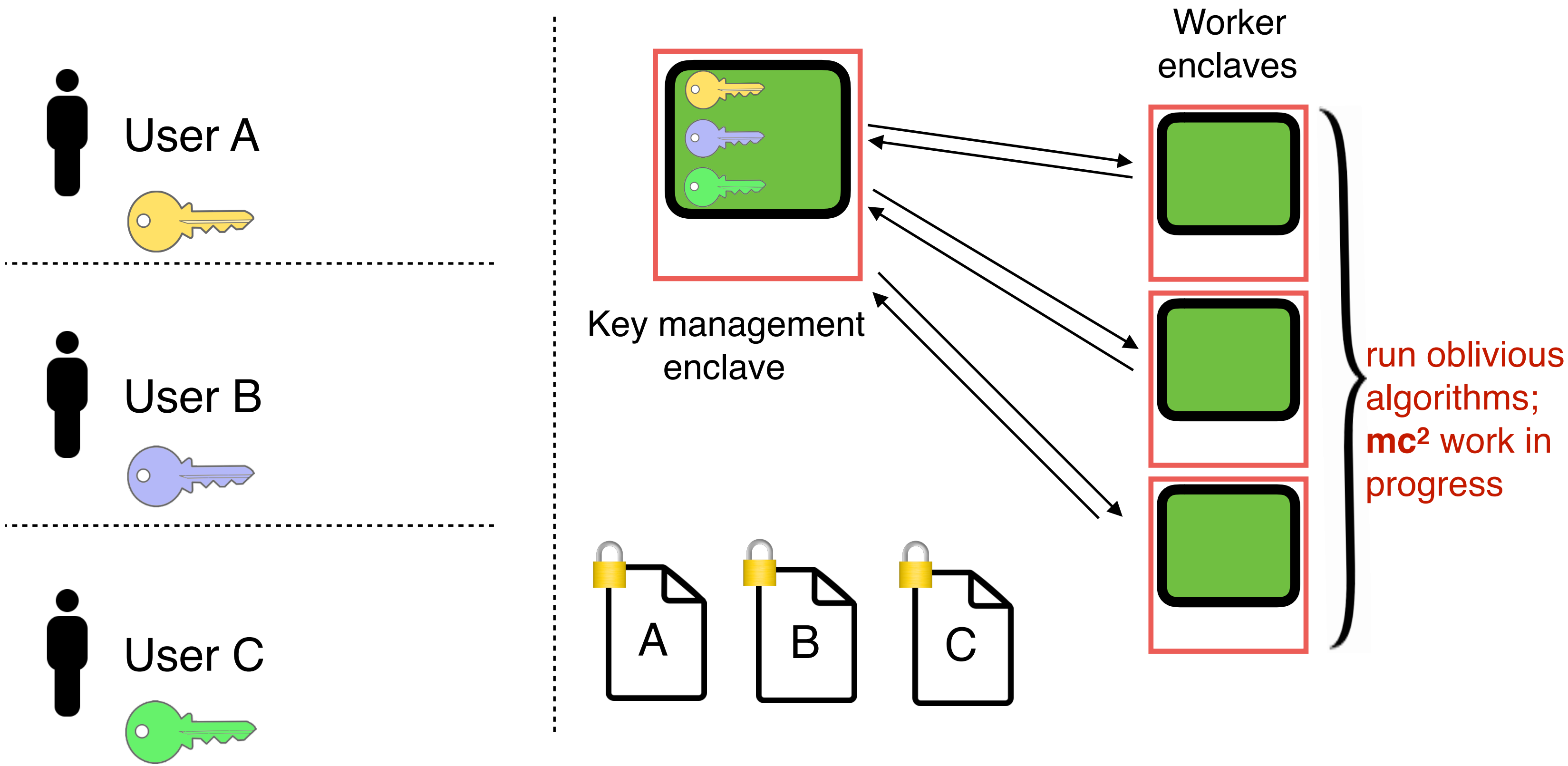
Secure collaborative ML via enclaves



Secure collaborative ML via enclaves

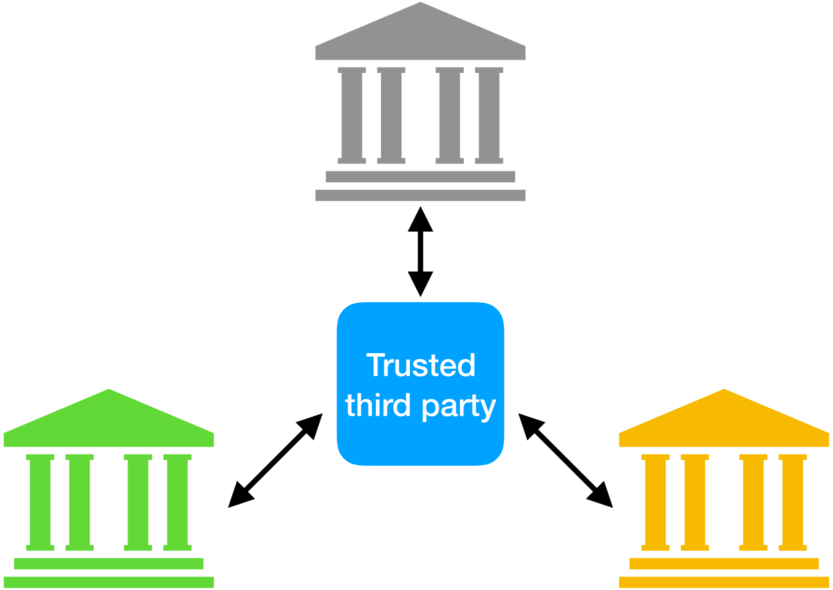


Secure collaborative ML via enclaves

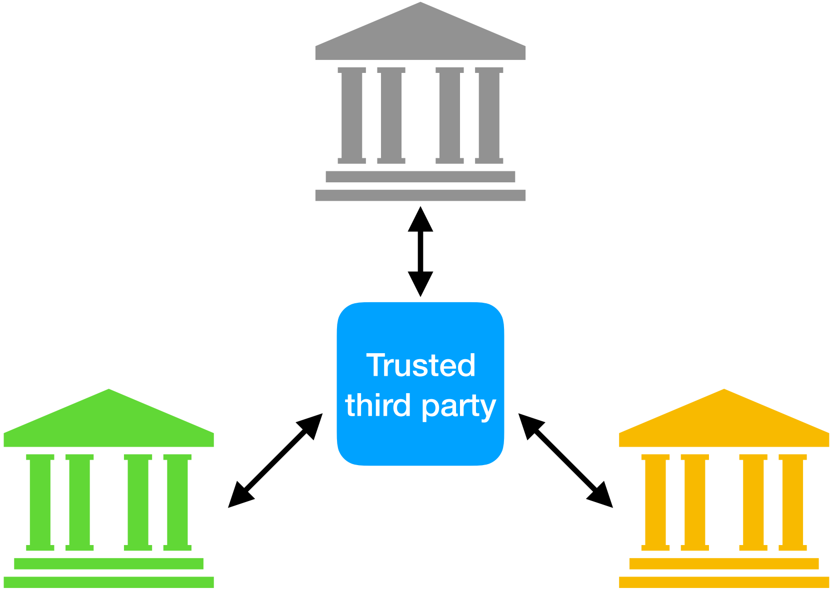


Secure multiparty computation

(MPC [Yao82,GMW87,BGW88])



Secure multiparty computation (MPC [Yao82,GMW87,BGW88])



- Parties emulate a trusted third party via cryptography

Secure multiparty computation (MPC[Yao82,GMW87,BGW88])

- Parties emulate a trusted third party via cryptography

Secure multiparty computation (MPC[Yao82,GMW87,BGW88])

- Parties emulate a trusted third party via cryptography

Secure multiparty computation (MPC[Yao82,GMW87,BGW88])

- Parties emulate a trusted third party via cryptography
- No party learns any party's input beyond the final result

Main challenge: Performance

Generic secure multi-

party computation

[SPDZ]

Main challenge: Performance

Generic secure multi-
party computation
[SPDZ]

Example: train linear models

Main challenge: Performance

Generic secure multi-party computation [SPDZ]

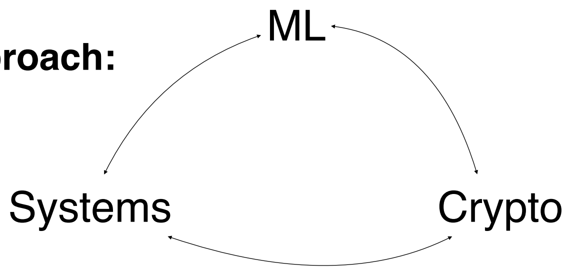
Example: train linear models

3 months

Main challenge: Performance

Generic secure multi-party computation [SPDZ]

Our approach:



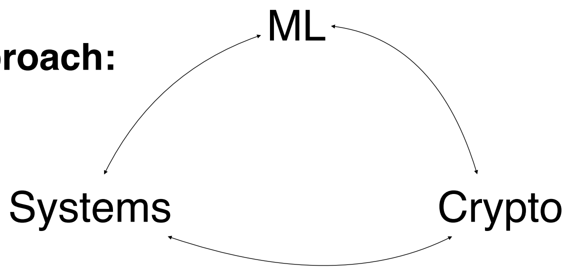
Example: train linear models

3 months

Main challenge: Performance

Generic secure multi-party computation [SPDZ]

Our approach:



Example: train linear models

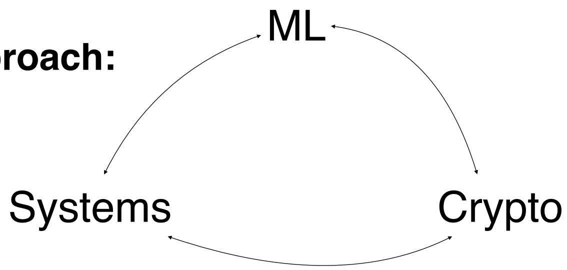
Helen [IEEESP'19]

3 months

Main challenge: Performance

Generic secure multi-party computation [SPDZ]

Our approach:



Example: train linear models

Helen [IEEESP'19]

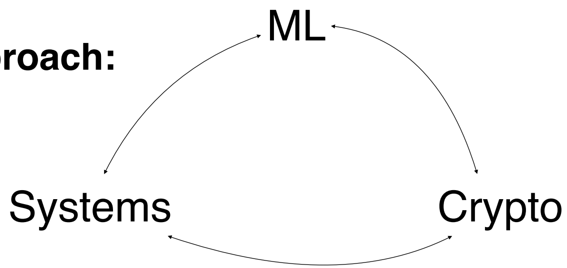
3 months

< 3 hours

Main challenge: Performance

Generic secure multi-party computation [SPDZ]

Our approach:



Example: train linear models

Helen [IEEESP'19]

3 months → **< 3 hours**

Delphi [USEC20]: secure inference for neural networks

mc^2: work in progress

multi-party **c**ryptographic
collaboration

mc^2: work in progress

multi-party **c**ryptographic
collaboration

An easy-to-use secure collaborative learning platform for the non-expert

mc^2: work in progress

multi-party **c**ryptographic **c**ollaboration

An easy-to-use secure collaborative learning platform for the non-expert

mc^2: work in progress

multi-party **c**ryptographic **c**ollaboration

An easy-to-use secure collaborative learning platform for the non-expert

User specifies Python DSL for learning task which automatically compiles to oblivious collaborative computation in enclaves or in MPC

mc^2: work in progress

multi-party **c**ryptographic
collaboration

An easy-to-use secure collaborative learning platform for the non-expert

User specifies Python DSL for learning task which automatically compiles to oblivious collaborative computation in enclaves or in MPC

Open source: https://github.com/mc2-project

- Secure collaborative XGBoost

mc^2: work in progress

multi-party **c**ryptographic **c**ollaboration

An easy-to-use secure collaborative learning platform for the non-expert

User specifies Python DSL for learning task which automatically compiles to oblivious collaborative computation in enclaves or in MPC

Open source: https://github.com/mc2-project

- Secure collaborative XGBoost
- Collaboration with ScotiaBank, Azure Confidential, Ericsson, and Ant Financial

mc^2: work in progress

multi-party **c**ryptographic **c**ollaboration

An easy-to-use secure collaborative learning platform for the non-expert

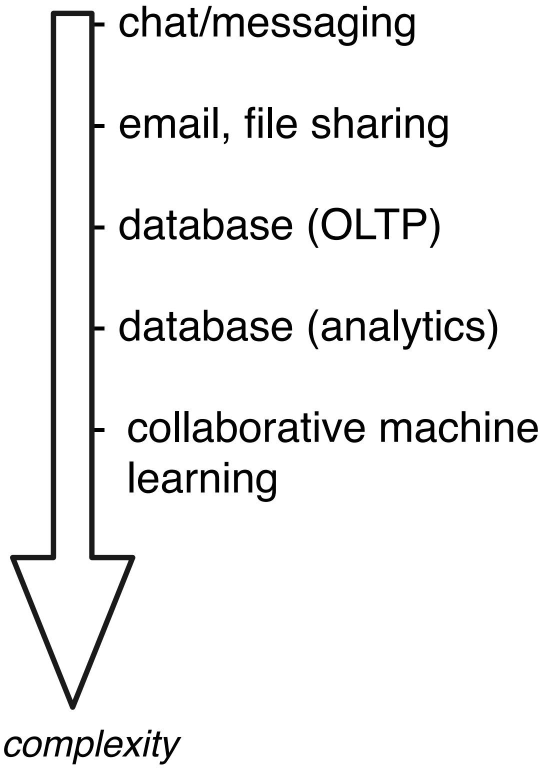
User specifies Python DSL for learning task which automatically compiles to oblivious collaborative computation in enclaves or in MPC

Open source: https://github.com/mc2-project

- Secure collaborative XGBoost
- Collaboration with ScotiaBank, Azure Confidential, Ericsson, and Ant Financial

Potential societal impact is exciting

Systems in the cloud



Principles

- Assume attackers will eventually break into the cloud
- Be prepared by processing data in encrypted form

- Co-design systems and cryptography for performance

Principles

- Assume attackers will eventually break into the cloud
- Be prepared by processing data in encrypted form

- Co-design systems and cryptography for performance

1. Focus on a workload. Identify a set of core operations the system needs
2. Identify an efficient secure protocol for each operation
3. Design a planner to combine the building blocks based on their constraints and cost model

Principles

- Assume attackers will eventually break into the cloud
- Be prepared by processing data in encrypted form

- Co-design systems and cryptography for performance

1. Focus on a workload. Identify a set of core operations the system needs
2. Identify an efficient secure protocol for each operation
3. Design a planner to combine the building blocks based on their constraints and cost model

Thank you!

raluca.popa@berkeley.edu @ralucaadapopa