Developing a secure SQL/key-value translation service

WPCCG 2017

Davi Boberg, Luiz Gomes-Jr, Marcelo Rosa e Keiko Fonseca
UTFPR - Curitiba

Overview

- Data in the Cloud
- SecureCloud and Intel SGX
- Objectives
- Related Work
- System Architecture
- SQL translation
- Experiments
- Conclusion

Data in the Cloud

- Cloud services simplify deployment and maintenance of information systems
- Economical and technological advantages for the customers
- Data security and privacy becomes a central issue
  - Customers must trust providers
  - Even best security practices are vulnerable to internal attacks

SecureCloud
Secure Big Data Processing
in Untrusted Clouds

SecureCloud Project

- Goal: provide secure cloud services
- International and interdisciplinary collaboration
- Provides: service containers, communication protocols, data processing and **storage**

European Commission
Horizon 2020
European Union funding
for Research & Innovation
RNP
BRASIL
GOVERNO FEDERAL

Intel SGX

- New CPU instruction set debuted with 2015’s 6th generation Intel Core processors
- Trusted Platform Module (TPM): remote attestation, binding, sealing
- Processing done in encrypted memory regions (enclaves)
- Current memory limit: 128 M

Objectives

- Implement a SQL-compatible secure DBMS
- Use Intel’s SGX technology

Related Work

- Homomorphic encryption
  - Computationally expensive
- Securing entire DBMS
  - Not practical with current technology
- Our proposal:
  - Simplify data and query models (NoSQL/key-value)
  - Modularize the system to fit the SGX enclave

Key-value stores

- One of the first NoSQL/BigData approaches
- Most simple data and query models
- Easy to distribute and feed MapReduce jobs
- Data represented as uninterpreted values (like BLOBs); unique string keys identify the entries
- No querying standards, but usually patterns on keys' strings

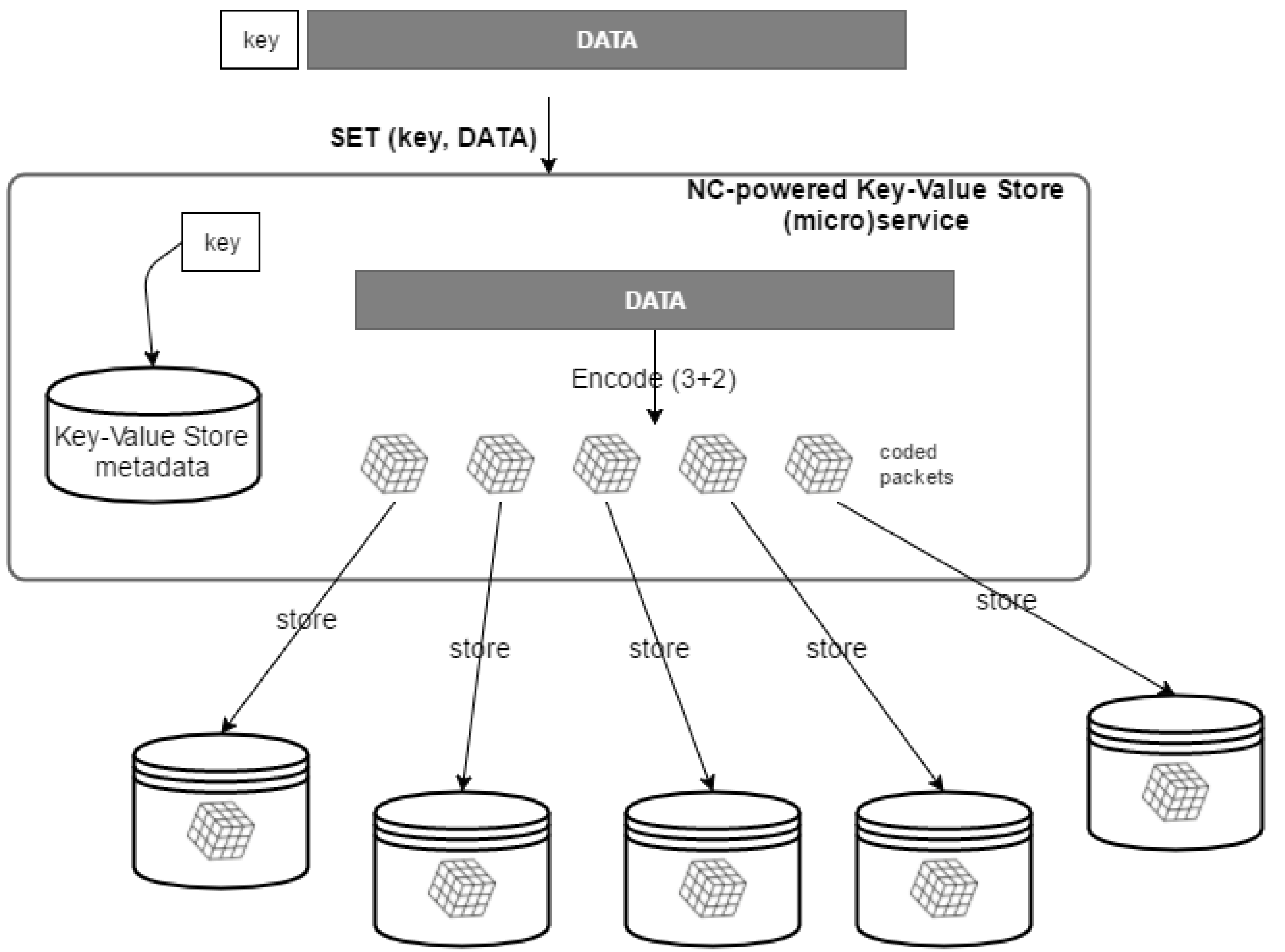
Key-value stores

Key	Value
customers/403624/name	John
customers/403624/purchases	21/10/2017-$50 || 19/10/2017-$80 || 1/10/2017-$10 ...
customers/403624/picture	LeNxAXyPDS24oom/l+GIELfj9bLE/TxDuJ6Q3WsgsQ+Zk50ohXo2IFvK42Nftk+gNL5HReMQOXHAzwrCMMG5N+5rhaICCreDVRAMu1oLufjoTR8IAQklbG3SDdXdxoXkDOr+Tq6Dzo6z2KuVSkD0+rCeMZHmp31qgQn4/L4RXJRXoV/9AHX

ChocolateCloud

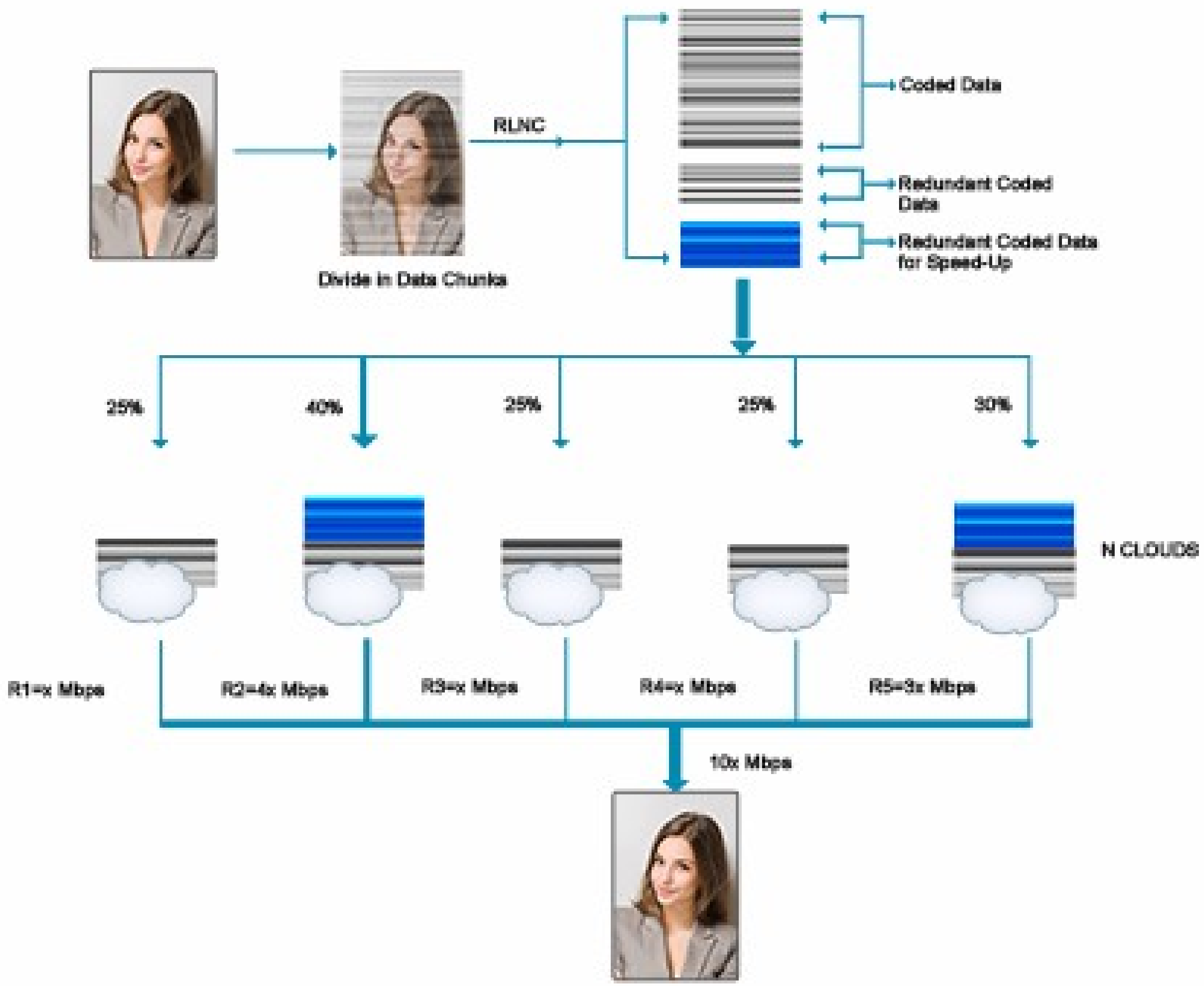
- Secure key-value store
- Network coding for data encryption and redundancy
  - Attacker needs to compromise multiple servers to recompose data
- SecureCloud partner
- SGX implementation

ClocolateCloud – Network Coding

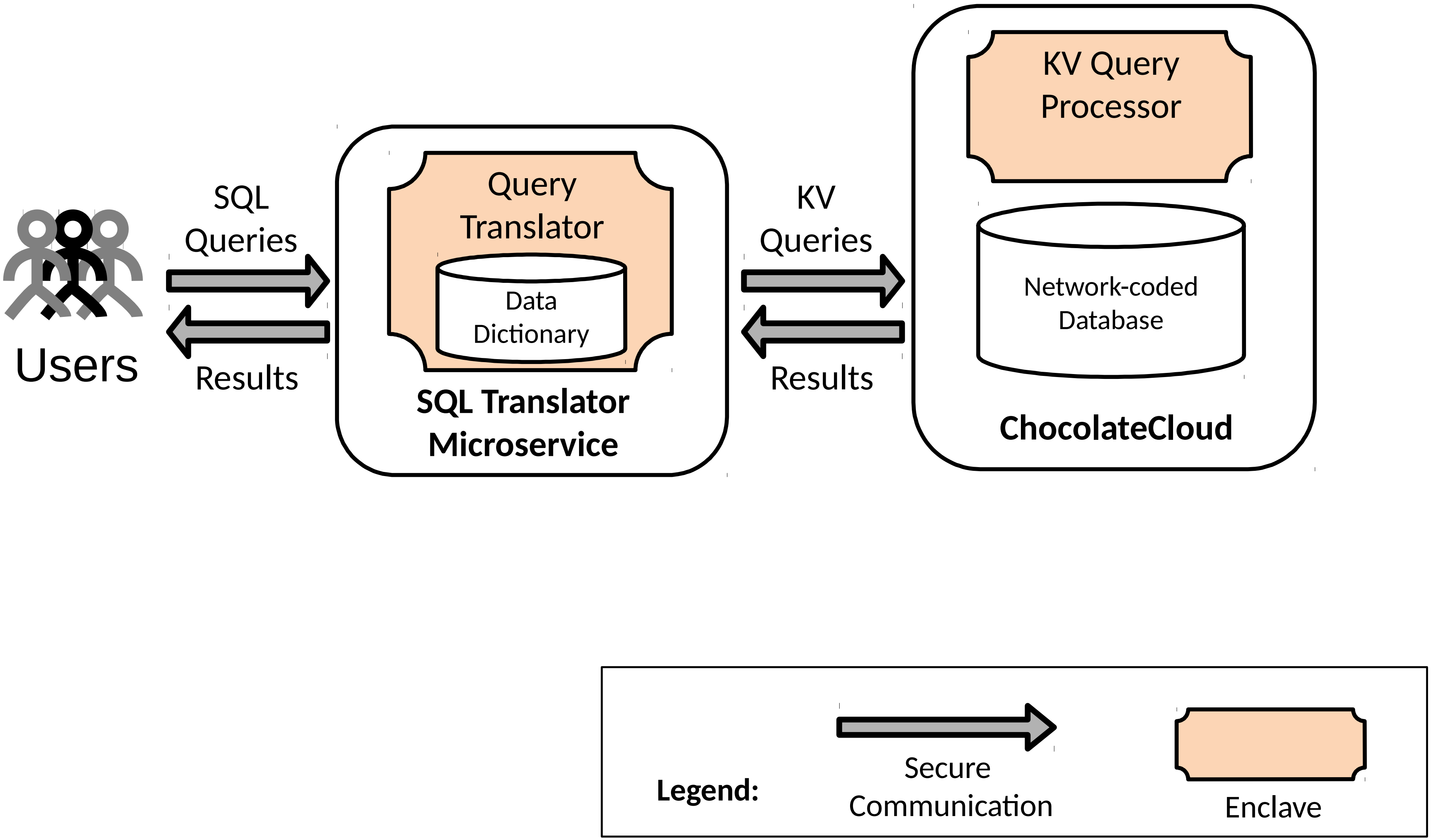


Total data stored: 167%
Protection against 2 node failures
Nodes store coded packets, not plaintext data

ClocolateCloud – Network Coding



SQL Translator



Data mapping

- Compose keys with table names and primary keys:

  $table\_name/pk_1/pk_2/.../pk_n$

- Concatenate attribute contents as values

Data mapping

ID	**timestamp**	**tensionA**	**tensionB**	**tensionC**
65678	20170907-10:30	234	21	148
12334	20170825-14:10	67	41	53

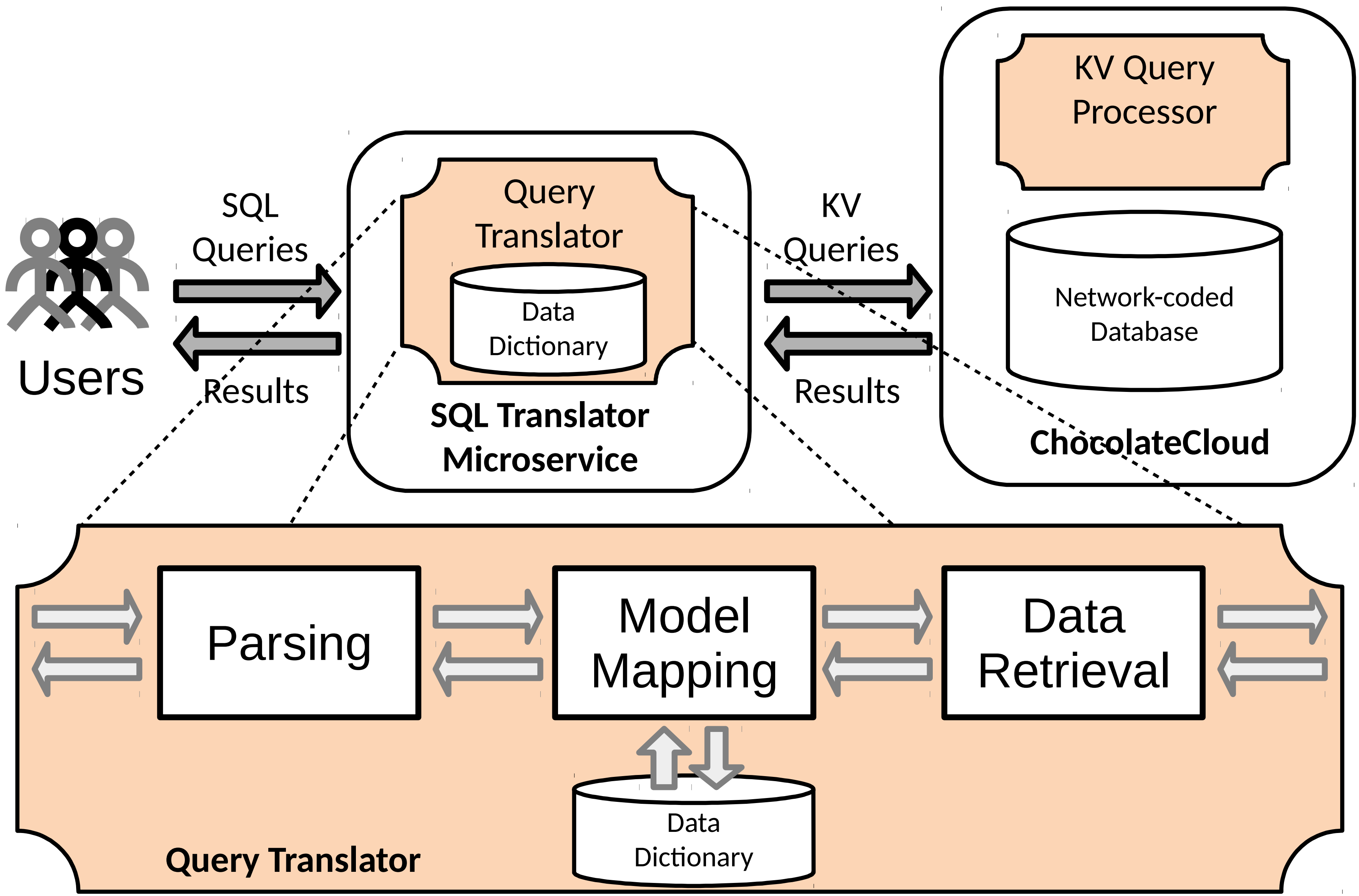
Key	**Value**
reading/65678/20170907-10:30	234|21|148
reading/12334/20170825-14:10	67|41|53

Query mapping

```
SELECT TensionA, Tension B
   FROM reading
   WHERE ID=32144
```

```
GET https://chocolate-cloud.cc/consumption/
      ?key_prefix=reading/32144
```

SQL Translator



Experiments

- Real scenario: electricity consumption data from a large power company
- Tests implemented: simple data insertions and selections
- Goal 1: show that the translation has a small impact over total processing
- Goal 2: show that the translation times grow linearly with data size

Running times

Table 1: Performance tests for select and insert queries (in seconds)

Test	Processing time	Total time	Total time/query
10 `Selects`	0.000171	3.0767	0.308
100 `Selects`	0.000189	28.5287	0.285
1000 `Selects`	0.000172	286.5795	0.286
10 `Inserts`	0.001626	4.1538	0.415
100 `Inserts`	0.013217	41.3288	0.413
1000 `Inserts`	0.13250	418.5262	0.418

Running times

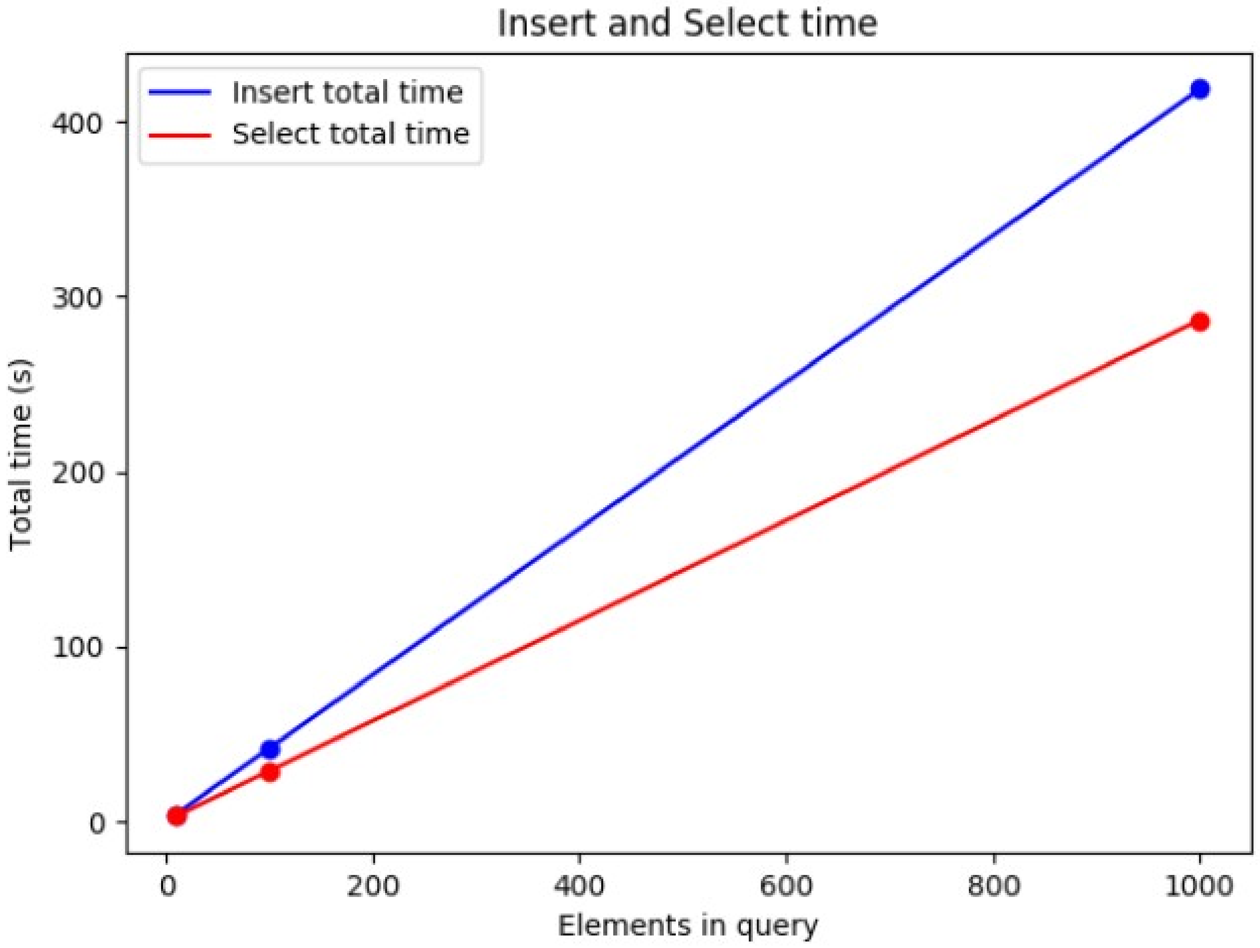


Figure 3: Evaluation of performance (total time) for `INSERT` and `SELECT` statements

Running times

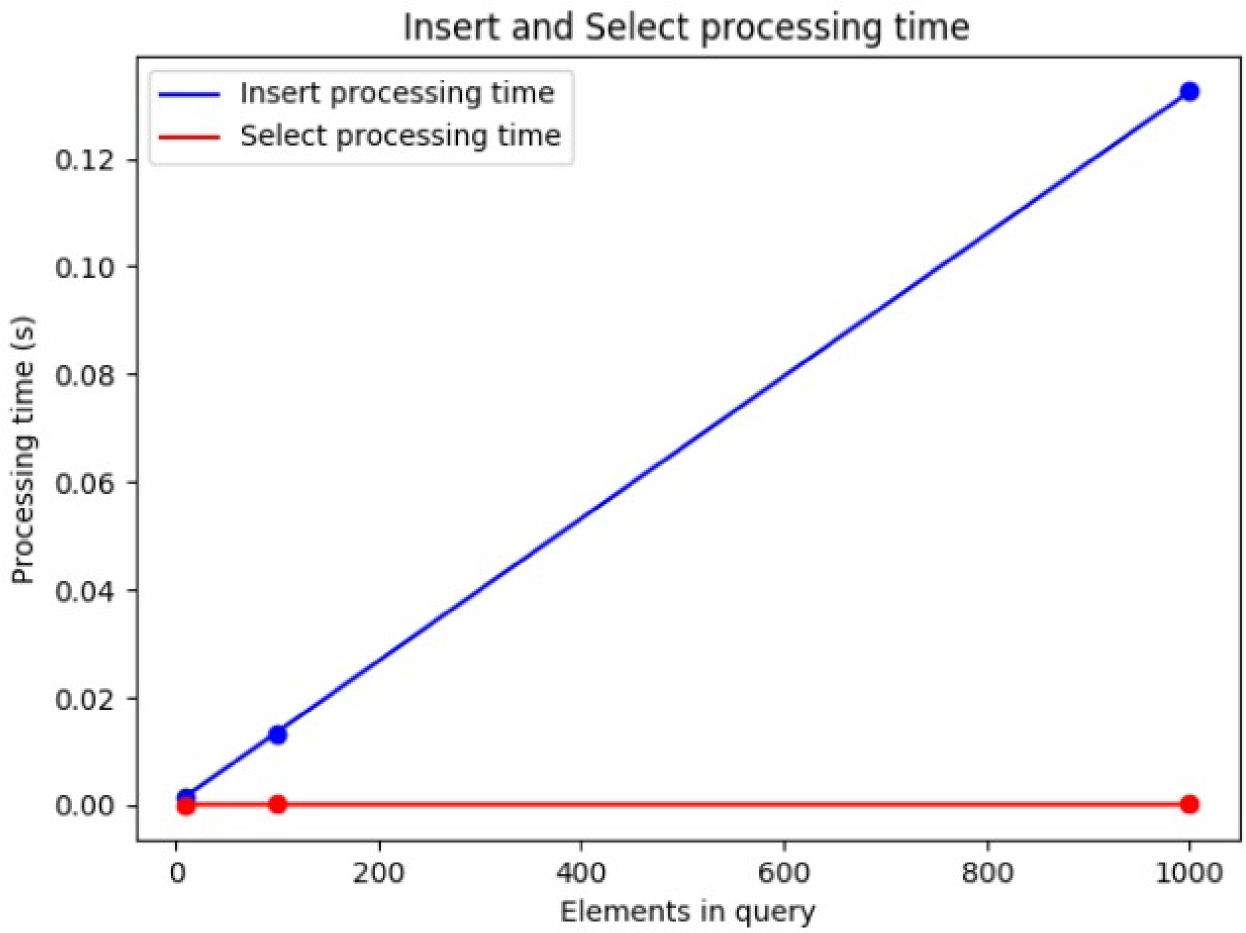
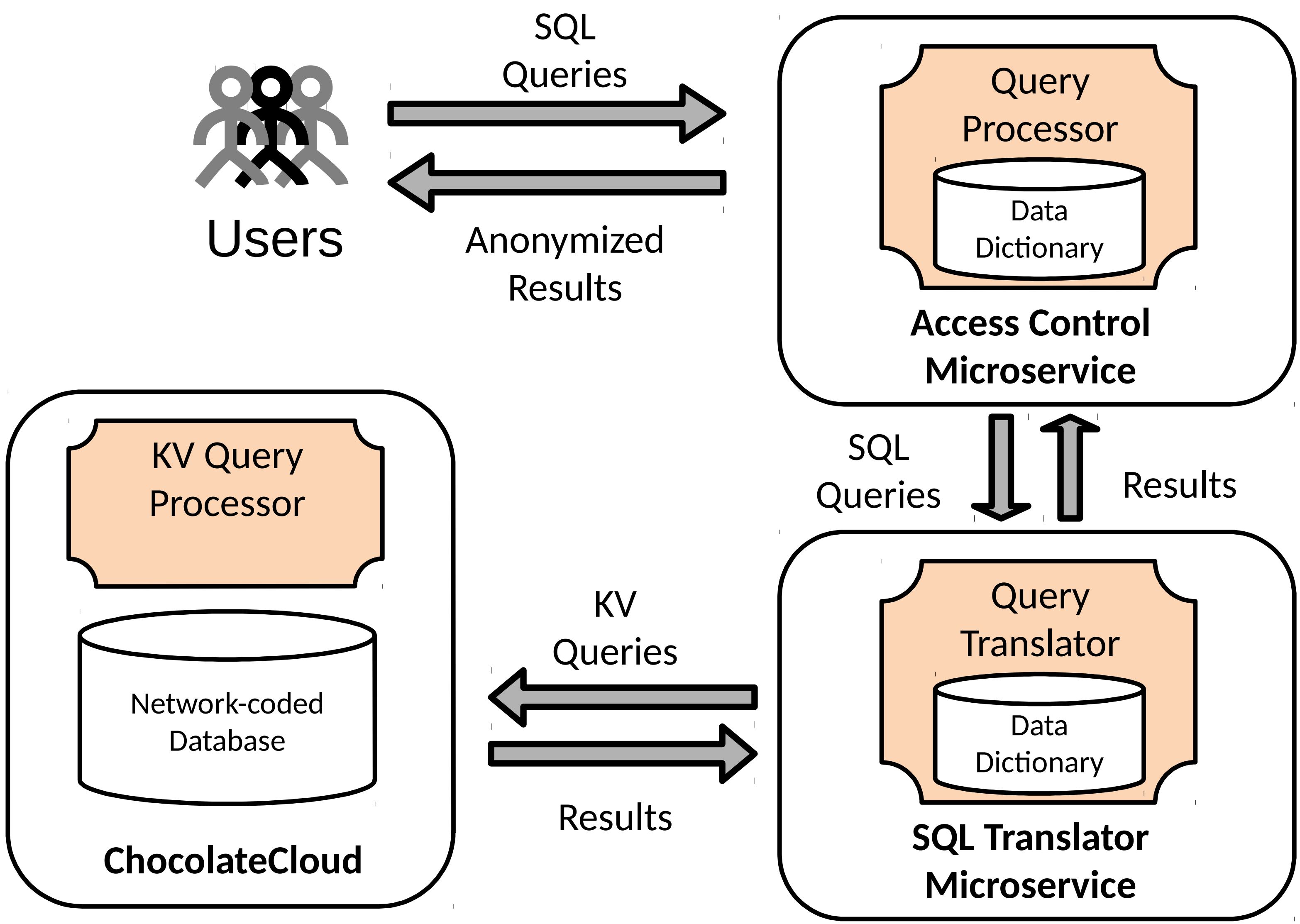


Figure 4: Evaluation of performance (processing time) for `INSERT` and `SELECT` statements

Conclusion

- Work in progress, many cool things to deal
- Current focus: SGX and network services integration
- Future work: more complex queries and optimizations
- Complete architecture with access control/anonymization service

Full architecture



Acknowledgments

This research is being performed in the context of the SecureCloud project. The SecureCloud project has received funding from the European Union's Horizon 2020 research and innovation programme and was supported by the Swiss State Secretariat for Education, Research and Innovation~(SERI) under grant agreement number 690111. This work was partially funded by the EU-BR SecureCloud project (MCTI/RNP 3rd Coordinated Call).

Thank you

Contact: gomesjr@dainf.ct.utfpr.edu.br