Hochschule Harz
Harz University of Applied Sciences

Technische Universität Braunschweig

Institute of Operating Systems and Computer Networks

STANlite – a database engine for secure data processing at rack-scale level

IEEE International Conference on Cloud Engineering (IC2E'18)

V. A. Sartakov, N. Weichbrodt, S. Krieter, T. Leich, R. Kapitza, April 20, 2018

This work was partly supported by the DFG under priority program SPP2037

Intro

Data processing in cloud databases – commonly used practice

- Leakage of security sensitive information
- Compromising of data processing

Mechanisms of prevention:

- Own trusted infrastructure
- Secure processors
- Homomorphic encryption

Intro

Data processing in cloud databases – commonly used practice

- Leakage of security sensitive information
- Compromising of data processing

Mechanisms of prevention:

- Own trusted infrastructure
- Secure processors
- Homomorphic encryption

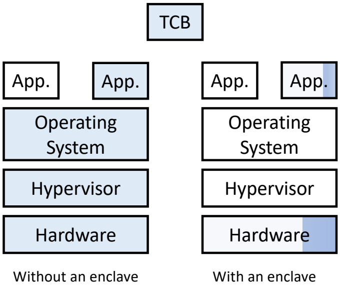
Intel Software Guard eXtensions (SGX)

- Trusted execution in untrusted environments

Trusted execution in untrusted environments

SGX Enclaves – new system entities:

- Located in User space
- Physical pages are encrypted
- Cannot be accessed by devices or software

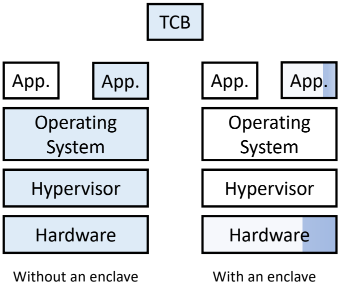


Trusted Computing Base

Trusted execution in untrusted environments

SGX Enclaves – new system entities:

- Located in User space
- Physical pages are encrypted
- Cannot be accessed by devices or software



Trusted Computing Base

⇒ Trusted execution on commodity hardware

Programming of enclaves

Challenges:

- Software should be self-contained and fully located inside an enclave
  - No dependencies

Programming of enclaves

Challenges:

- Software should be self-contained and fully located inside an enclave
  - No dependencies
- Some instructions are forbidden
  - No System calls

Programming of enclaves

Challenges:

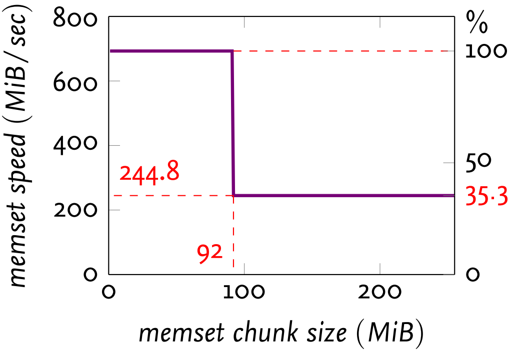
- Software should be self-contained and fully located inside an enclave
  - No dependencies
- Some instructions are forbidden
  - No System calls
- ECalls and OCalls – expensive switching mechanisms between trusted and untrusted modes
  - At least in 50 times slower than a system call [1, 2]

Programming of enclaves

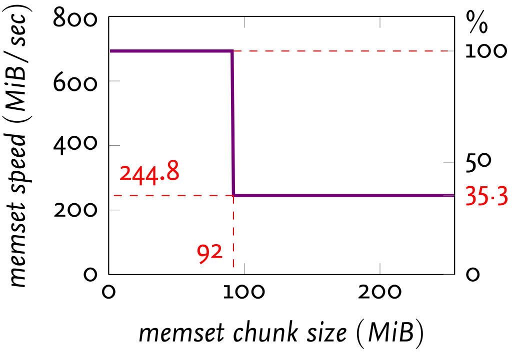
Challenges:

- Software should be self-contained and fully located inside an enclave
  - No dependencies
- Some instructions are forbidden
  - No System calls
- ECalls and OCalls – expensive switching mechanisms between trusted and untrusted modes
  - At least in 50 times slower than a system call [1, 2]
- Paging

Enclave Page Cache (EPC) limit



EPC limit



- ~92 MiB are available
- Heavyweight paging
  - Involves a kernel
  - Threads should exit
  - Encryption/decryption
  - Integrity protection

STANlite

STANlite: a secure database for data processing in clouds

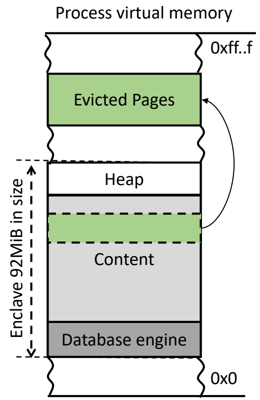
- Built on top of SGX Enclaves
- Processes large volumes of data without paging
- ECall-free high-performance communications over Remote Direct Memory Access (RDMA)

Agenda

- **Implementation of key components**
- **Evaluation**
- **Related works**
- **Conclusion**

STANlite

- Enclaved software can access untrusted memory
- The database can manage own pages
  - Swap in and swap out on request
  - Keep frequently used content inside
  - Evict rarely used content in encrypted form
- Fix memory layout to prevent the heavyweight paging



⇒ Special Virtual Memory Engine (VME)

Architecture

VME components:

- Warm Store
- Cold Store
- Least Recently Used list

Swapping:

- Encryption/Decryption
- Hash sums for rollback attacks prevention

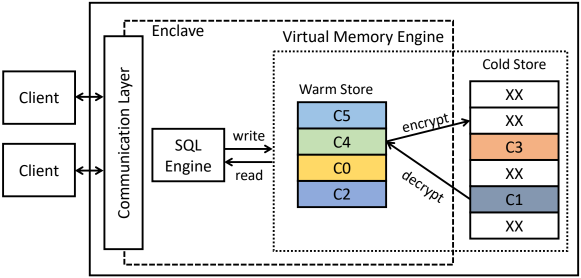


Table of Contents

The basis

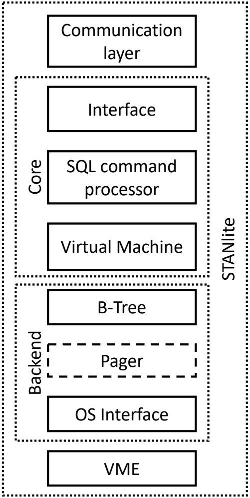
SQLite as SQL engine:

- Low footprint
- Read/Write semantic

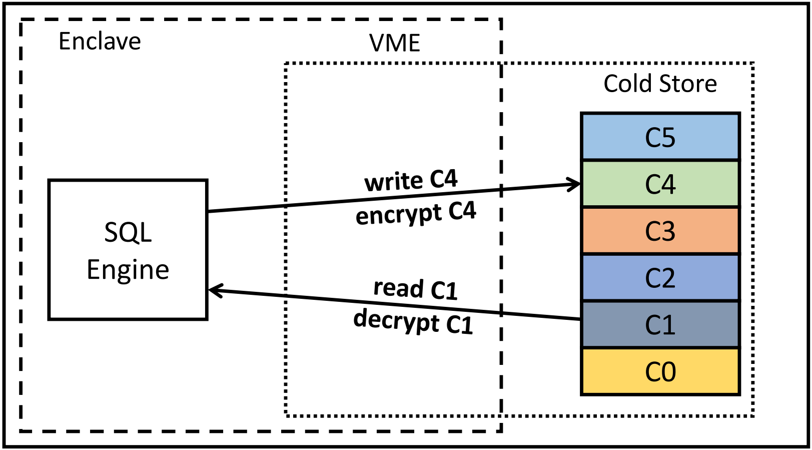
VME Integration:

- OS Interface
- Disabled Pager

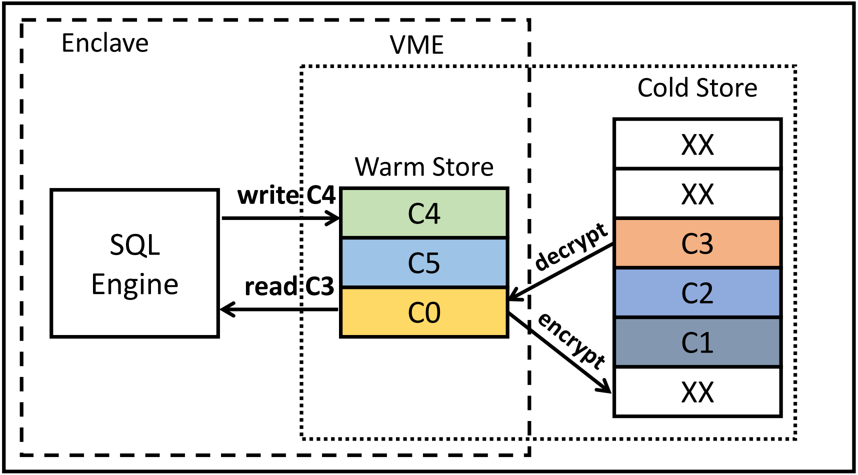
Three VME modes



VME mode: Integrity and Confidentiality (Integrity)



VME mode: +Cache (Caching)



VME mode: +Fetch (Fetching)

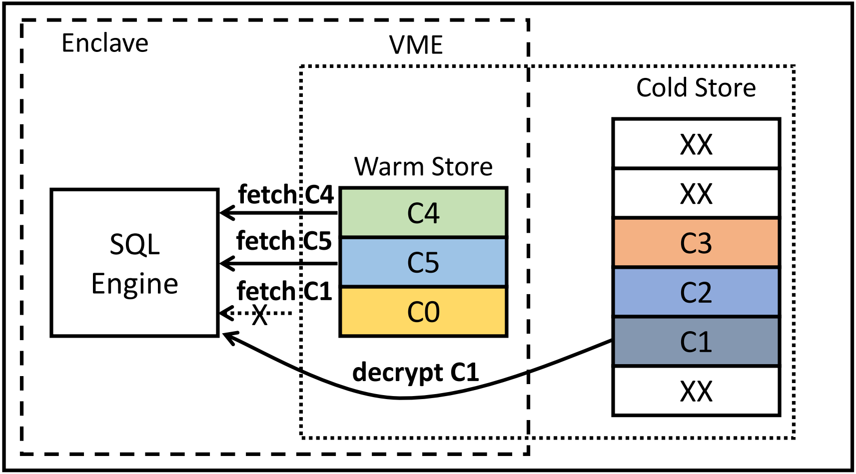


Table of Contents

Evaluation

Goals:

- Compare virtual memory engines in synthetic tests
- Real-life performance

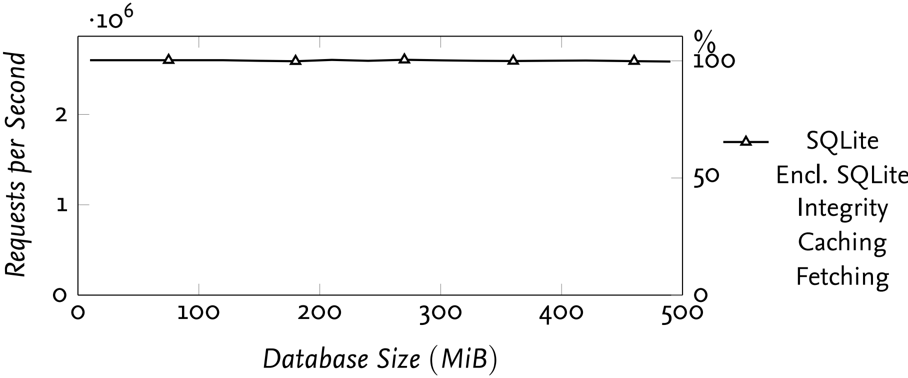
Benchmarks:

- Microbenchmark: a database with random access
- Speedtest1 benchmark: compares different request types
- TPC-C benchmark: real-life load

Setups:

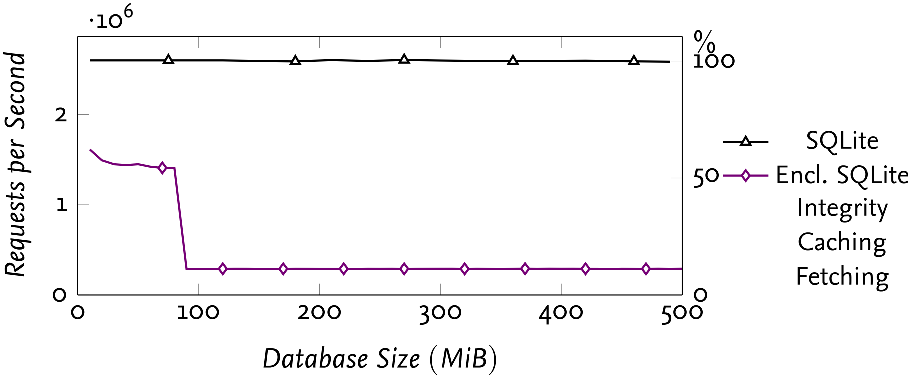
- VME modes: Integrity, Caching, Fetching
- Baselines: Enclaved vanilla SQLite, Non-enclaved vanilla SQLite

Microbenchmark



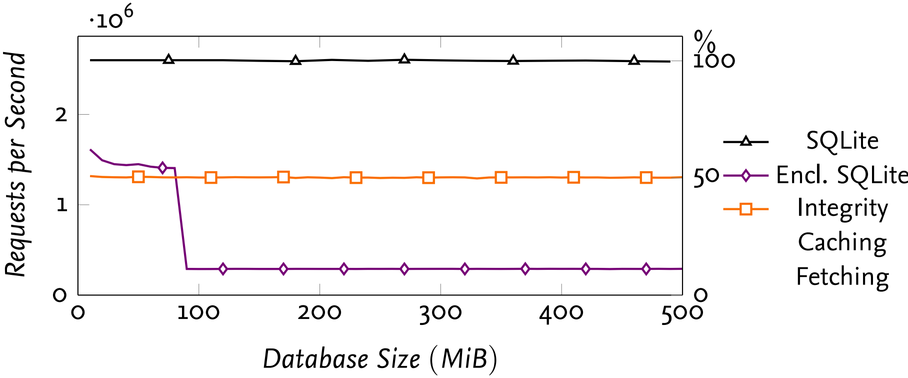

```
CREATE TABLE stest(ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, BODY CHAR;
INSERT INTO stest (BODY) VALUES('<...>'))
SELECT * FROM stest ORDER BY RANDOM() LIMIT 1
```

Microbenchmark



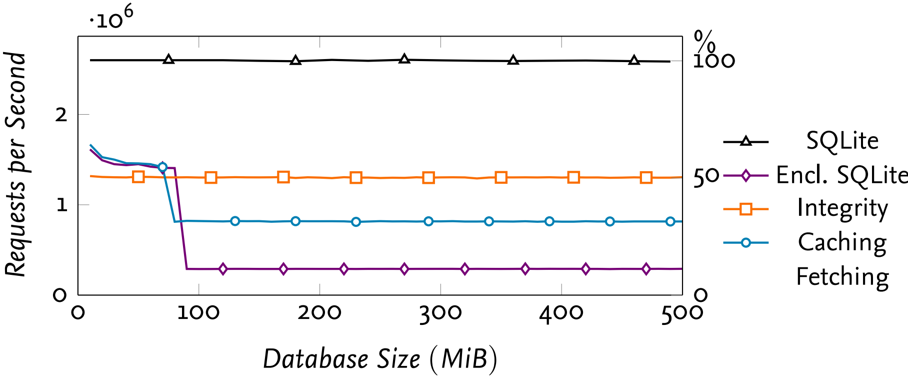

```
CREATE TABLE stest(ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, BODY CHAR;
INSERT INTO stest (BODY) VALUES('<...>'))
SELECT * FROM stest ORDER BY RANDOM() LIMIT 1
```

Microbenchmark



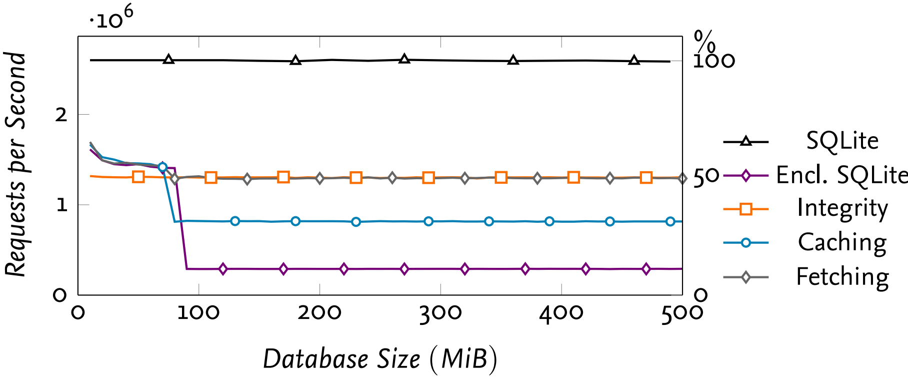

```
CREATE TABLE stest(ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, BODY CHAR;
INSERT INTO stest (BODY) VALUES('<...>'))
SELECT * FROM stest ORDER BY RANDOM() LIMIT 1
```

Microbenchmark




```
CREATE TABLE stest(ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, BODY CHAR;
INSERT INTO stest (BODY) VALUES('<...>'))
SELECT * FROM stest ORDER BY RANDOM() LIMIT 1
```

Microbenchmark




```
CREATE TABLE stest(ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, BODY CHAR;
INSERT INTO stest (BODY) VALUES('<...>'))
SELECT * FROM stest ORDER BY RANDOM() LIMIT 1
```

Speedtest1

VMEs show better performance (1.52×–2×) for:

- `SELECT`, `UPDATE`, `DELETE`, `4-way JOINs`, `subquery`, `ANALYZE`

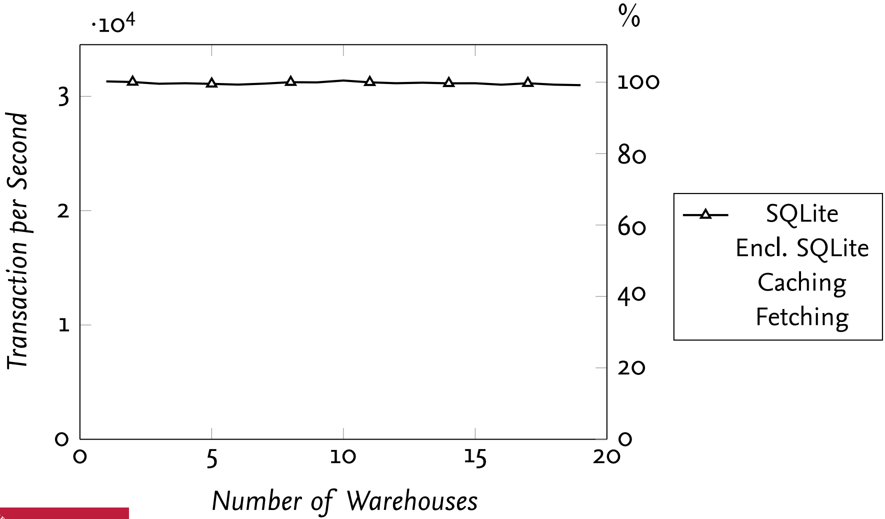
Enclaved SQLite shows better performance (1%–27%) for:

- `INDEX`, `DELETE with refill`, `refill`
- consumes heap memory

Warm Cache improves performance (up to 2×):

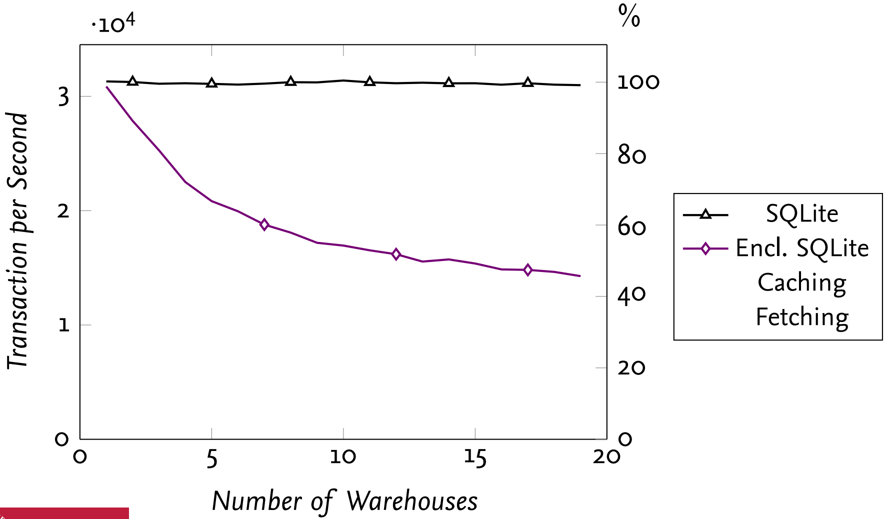
- Integrity check, refill

TPC-C



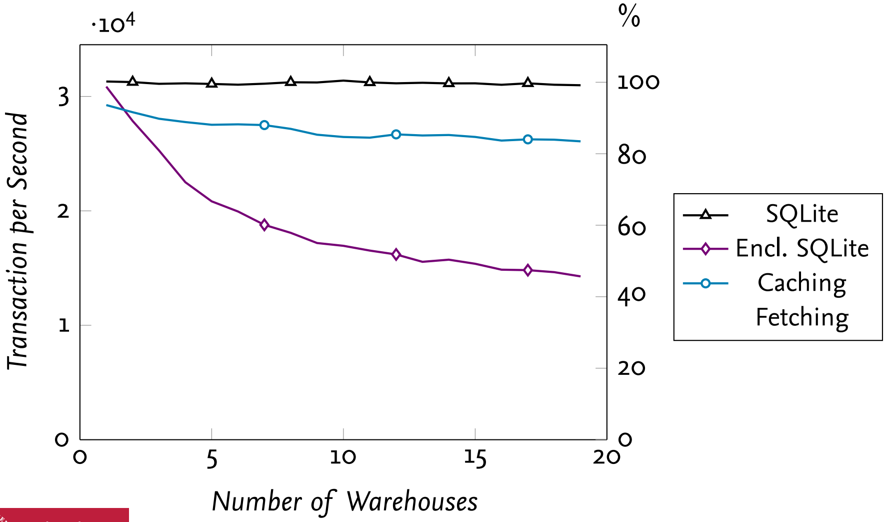
·10^4
%
Transaction per Second
3
2
1
0
100
80
60
40
20
0
0
5
10
15
20
Number of Warehouses
SQLite
Encl. SQLite
Caching
Fetching

TPC-C

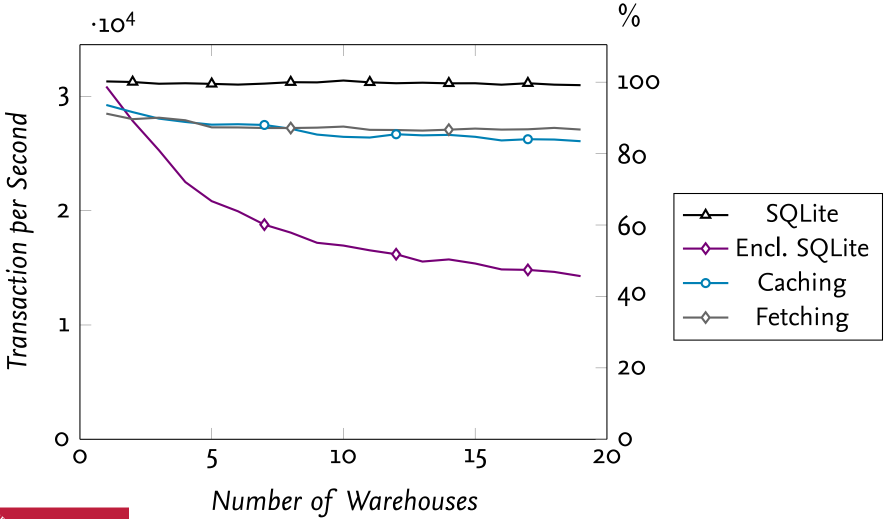


·10^4
%
Transaction per Second
3
2
1
0
100
80
60
40
20
0
0
5
10
15
20
Number of Warehouses
SQLite
Encl. SQLite
Caching
Fetching

TPC-C


·10^4
%
Transaction per Second
Number of Warehouses
0
1
2
3
0
5
10
15
20
0
20
40
60
80
100
SQLite
Encl. SQLite
Caching
Fetching

TPC-C



·10^4
%
Transaction per Second
Number of Warehouses
0
1
2
3
0
5
10
15
20
0
20
40
60
80
100
SQLite
Encl. SQLite
Caching
Fetching

Table of Contents

Related work

- Eleos – Paging for C++-based programs
  - Different memory types
  - Multiple VME modes
- Panoply, Graphene-SGX, SCONE – environments for legacy applications
  - Increase Trusted Computing Base and memory consumption
- Glamdring – code partitioning
  - We virtualise storage memory

Table of Contents

Conclusion

Intel SGX

- Trusted execution in untrusted environment
- Challenges: EPC limit, ECalls

STANlite

- Enclaved in-memory database
- Virtual memory engine
- RDMA-based communication layer

Evaluation

- Microbenchmark: 4.44×
- *Speedtest1*: 1.79×
- TPC-C (2GiB): 2.44×

References

- M. Orenbach, P. Lifshits, M. Minkin, and M. Silberstein, "Eleos: ExitLess OS Services for SGX Enclaves," in *EuroSys*, 2017, pp. 238–253.
- O. Weisse, V. Bertacco, and T. Austin, "Regaining Lost Cycles with HotCalls: A Fast Interface for SGX Secure Enclaves," in *Proceedings of the 44th Annual International Symposium on Computer Architecture*, 2017, pp. 81–93.