Avocado
A Secure In-Memory Distributed Storage System

Maurice Bailleu, Dimitra Giantsidi,
Vasilis Gavrielatos, Vijay Nagarajan

Do Le Quoc

Pramod Bhatotia
Distributed in-memory KVS

- Provides a high-performance, scalable, & fault-tolerant storage system
- Extensively used as a fundamental building block in modern online services
Trust in cloud storage

User → Storage operations → Distributed system → Untrusted infrastructure
How to design a **secure distributed in-memory KV store** for untrusted cloud environments?
Avocado

A secure distributed in-memory KVS for untrusted computing infrastructure

Properties:

• Security: confidentiality + integrity + freshness

• Fault tolerance

• Performance
Design
Basic design

Untrusted host

KV store

Replication layer

OS

NIC

Untrusted host

KV store

Replication layer

OS

NIC

...
Trusted computing

Can we use trusted computing for distributed in-memory KV stores?

**Trusted Execution Environment (TEEs):**
Hardware extensions for trusted computing, e.g., Intel SGX and ARM TrustZone

**Limitations:**
- Untrusted network
- Not well-suited for distributed systems
- Architectural limitations: memory, I/O, and attestation
Design challenges

#1: Networking
How to design a secure network stack?

#2: Fault tolerance
How to tolerate faults in Byzantine settings?

#3: Hardware limitations
How to overcome the architectural limitations of TEEs?
#1: Networking

- Frequent network operations are expensive
- NIC, network and OS are not trusted
- NIC cannot access TEE memory

We designed a network stack for trusted computing based on eRPC and DPDK for fast networking without exiting enclave
Splits the network stack into:
- Logic in the enclave and buffers in the host

Package format guarantees freshness
#2: Fault tolerance

- Crash-stop failure
  - Replication

- Network provider can manipulate traffic
  - BFT protocol

We can employ a non-byzantine protocol, due to the trust provided by TEEs and our network layer.
#2: Trusted replication protocol

Avocado based on non-Byzantine protocol (ABD):
- It runs inside the enclave to prevent equivocation
- Majority voting guarantees liveness and forking protection
#3: Hardware limitations

- EPC is limited (94 MiB)
- Secure paging for bigger memory area
- EPC paging incurs high overheads

We designed a fast EPC conserving in-memory KV data structure to overcome the enclave physical memory limitation.
#3: In-memory KV store

Split in Memory KV store into two parts:
- Keys -> stored in enclave for fast lookup
- Meta data -> split from key for atomic updates
- Values -> stored in untrusted memory, reducing EPC pressure
Overall system design

Enclave
- CAS
- Replication layer
- Networking layer

Key & metadata
- Encrypted values

NIC memory
- OS

Host

Untrusted network
Evaluation
Evaluation

• Questions
  1. What is the overall performance compared to BFT?
  2. How well does Avocado scale?

• Experimental setup:
  • 5x Intel i9-9900K (@3.60GHz, 8 cores, 16 HT)
  • Intel NIC XL710 (40Gb/s, QSFP+)

See the paper for more results
Avocado performs similar in read and write heavy workloads and outperforms BFT.
Scalability

Throughput [kOp/s]

Avocado scales with the number of increasing nodes

Avocado
Native

Higher is better
Avocado: A secure in-memory distributed storage system

**Security properties:** confidentiality + integrity + freshness

**Challenge:** How to leverage TEEs to design a high-performance secure and fault-tolerant in-memory KV store?

**Contributions**

<table>
<thead>
<tr>
<th>Contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trusted network stack</td>
</tr>
<tr>
<td>Trusted replication protocol</td>
</tr>
<tr>
<td>Trusted in-memory KV store</td>
</tr>
<tr>
<td>Configuration and attestation service (see the paper for details)</td>
</tr>
</tbody>
</table>
Thank you!

If you have follow up questions, please contact us:

Maurice Bailleu
M.Bailleu@ed.ac.uk

Dimitra Giantsidi
D.Giantsidi@sms.ed.ac.uk