Aalto University

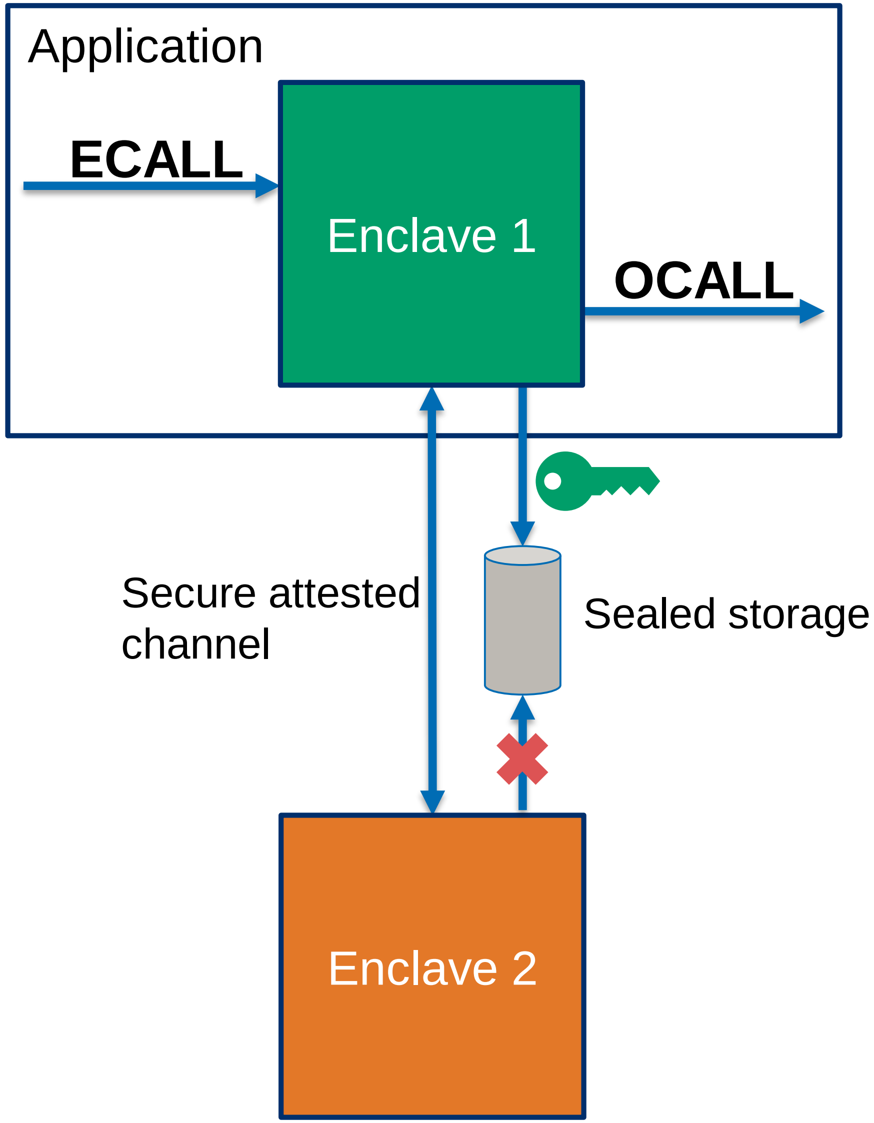
Migrating SGX enclaves with persistent state

Fritz Alder, Arseny Kurnikov, Andrew Paverd, N. Asokan

Intel SGX – a Trusted Execution Environment (TEE)

Enclave **(trusted code) – protected from OS and app**

- Isolated execution
- Data confidentiality and integrity (memory encryption)
- Encryption keys derived from CPU
- Attestation – proves what code is executed
  - Using group signatures



From the enclave‘s perspective, everything else is untrusted.

➢ Hypervisor, operating system, applications

Motivation

Intel SGX provides features that can be useful in a cloud computing setting

This requires integration of SGX into the existing cloud ecosystem

But: Bindings to physical machine clashes with cloud computing practices

➢ Cloud migration requires independence from physical machines

Persistent data to be migrated:

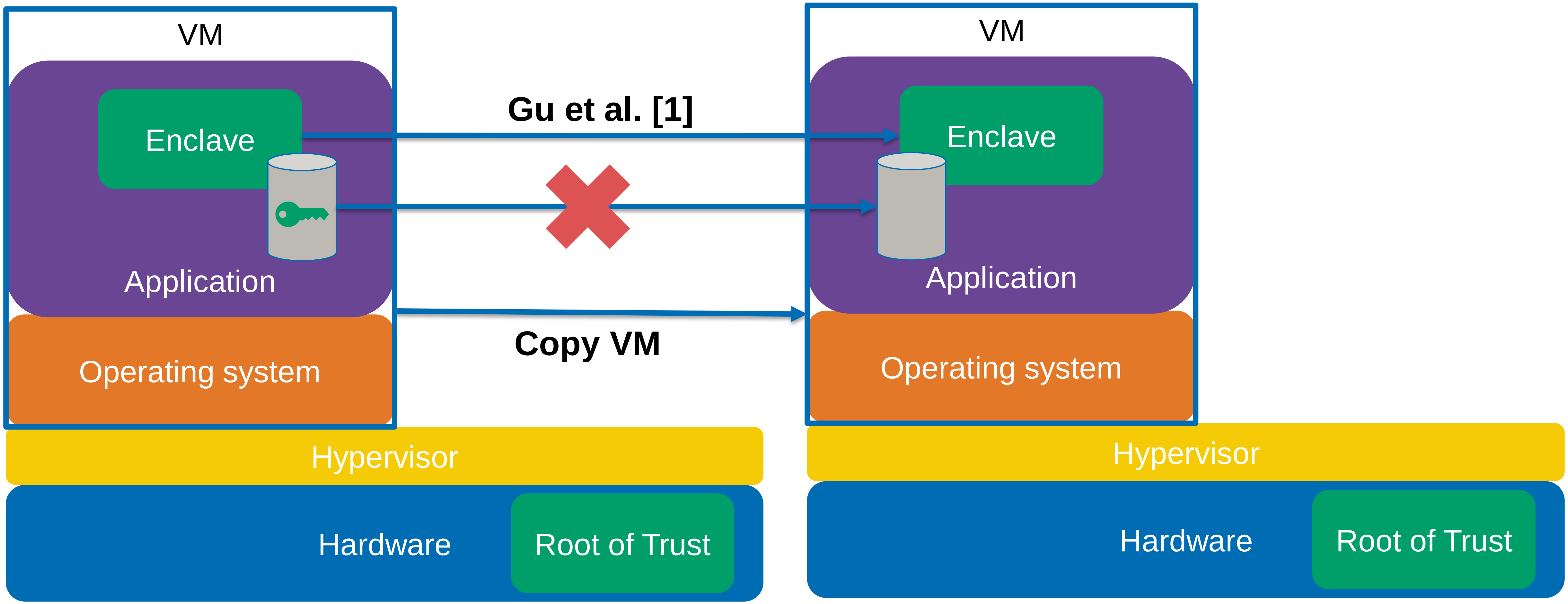
- Sealed data (Sealing key)
- Monotonic counter values

Hardware protection

- Session keys

In run-time enclave memory

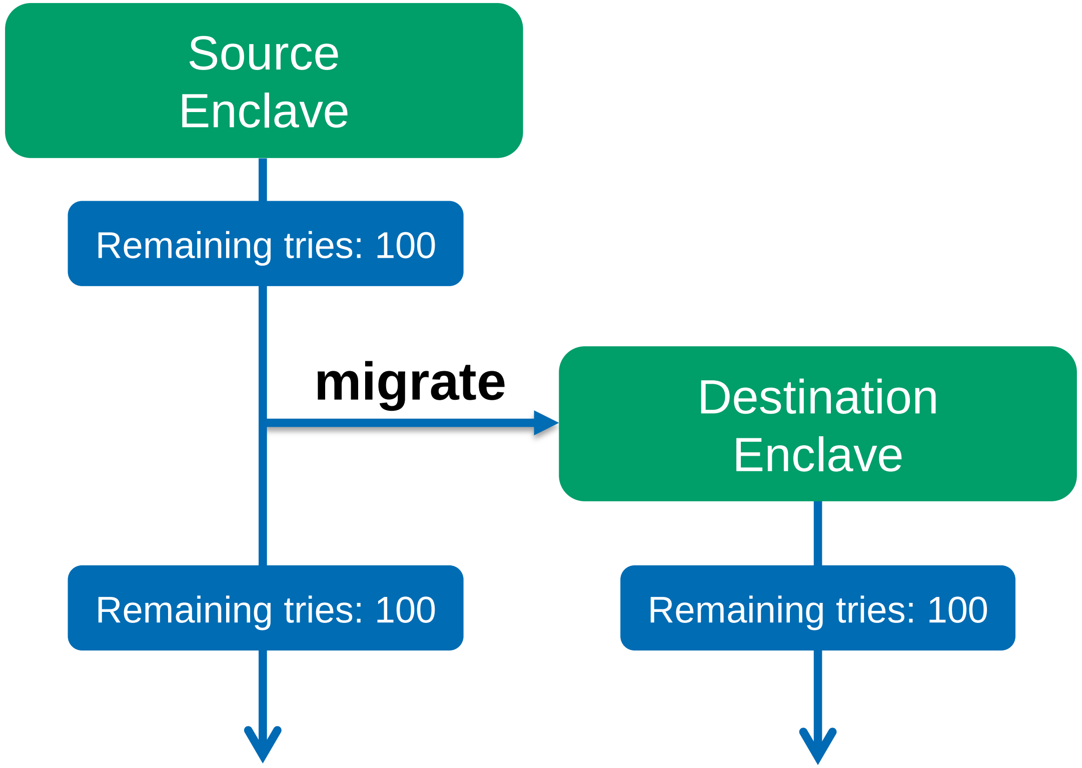
VM migration with Intel SGX



[1] Gu, Z. Hua, Y. Xia, H. Chen, B. Zang, H. Guan, and J. Li, “Secure live migration of SGX enclaves on untrusted cloud”, DSN 2017

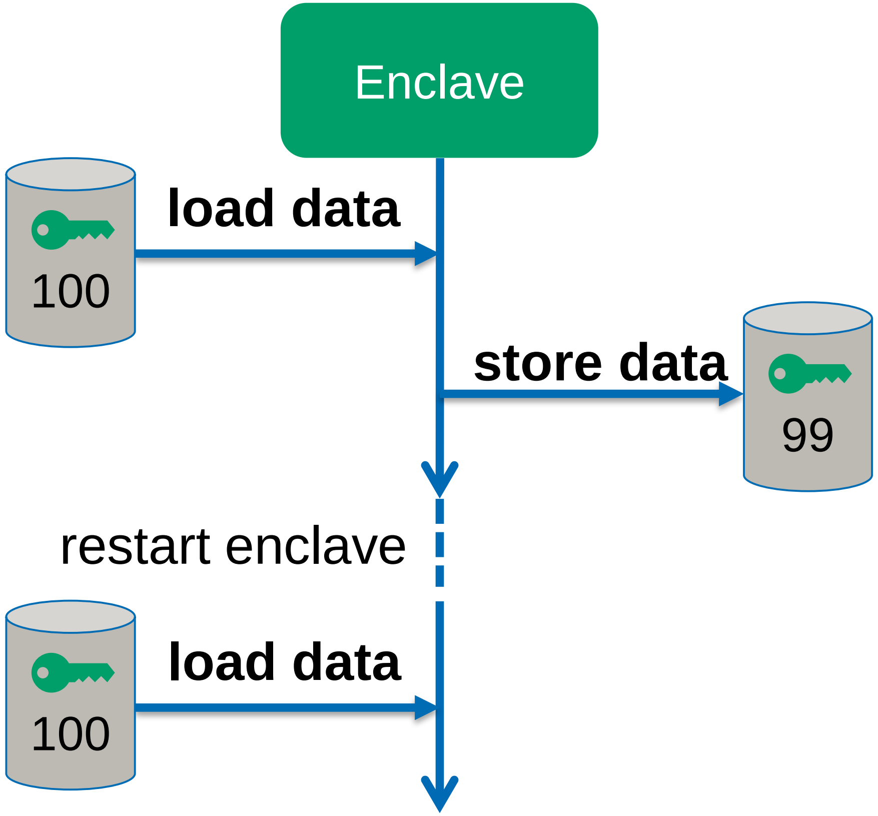
Fork attack

Example: Password guessing limitation



Do not shutdown source enclave after migration
→ Forked execution of enclaves
(with current state)

Rollback attack



Provide old persistent data on restart
Roll back the enclave state

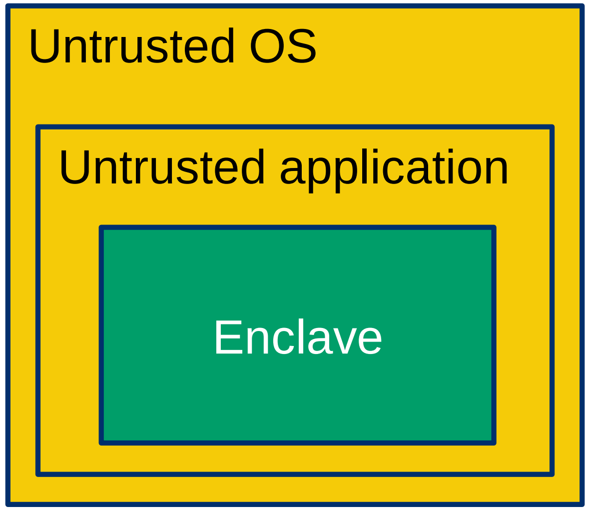
Requirements and adversary model

Requirements:

1. Maintain all SGX guarantees (Integrity, Confidentiality, Isolation)
2. Only migrate to correct and authorized machines
3. Prevent Fork and Rollback attacks
4. Low performance overhead (runtime <u>and</u> migration)
5. No hardware changes

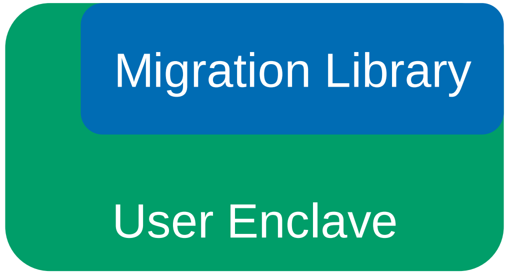
Adversary Model:

- **Same as SGX model:** Everything is untrusted
- Adversary can DoS (out of scope)
- Enclave developer is benign (wants to migrate)



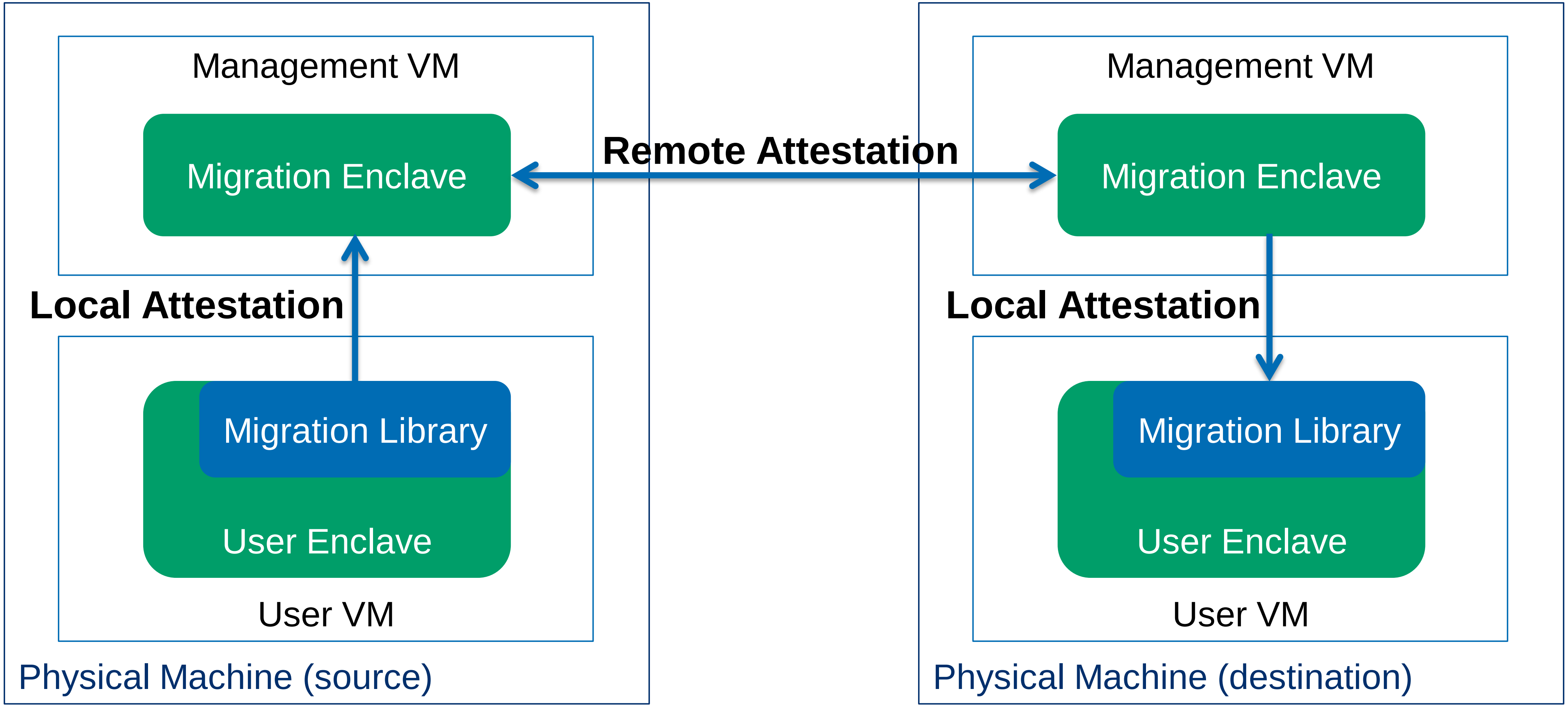
Design

1. **Migration Library – part of the user enclave (C++, 940 LOC)**
    - Sealing
        - Migratable sealing key
    - Monotonic counters
        - Migratable alternatives for SGX counters



2. **Migration Enclave – handles migration to destination (C++, 217 LOC)**
    - Receives data from local user enclave
    - Performs remote attestation to other Migration Enclave
    - Transfers data
    - Restores user enclave at destination

Architecture



- **Migratable versions of SGX primitives**
- **Migration Enclaves authorize themselves to each other**
  - **belong to the same provider**

SGX monotonic counters

- **Hardware supported counters**
- **Referenced by UUID**
- **Guaranteed to increase only**

UUID	Value
fa68e33b-…	5
864d2906-...	2

- **Naive solution to migrate monotonic counters:**
  1. Reference counters by static ID
  2. Map IDs to UUIDs
  3. Migrate Ids and Values, increment on destination

ID	UUID	Value
0	fa68e33b-…	5
1	864d2906-...	2

migrate

ID	UUID	Value
0	e20f15c6b-…	5
1	a0dd231a-...	2

- **This is very slow! 22 seconds to increment to 100**
  - Not feasible for large values

Migrating counters with offsets

Source machine

0 read

ID	UUID	Value	Offset
0	fa68e33b-…	0	0

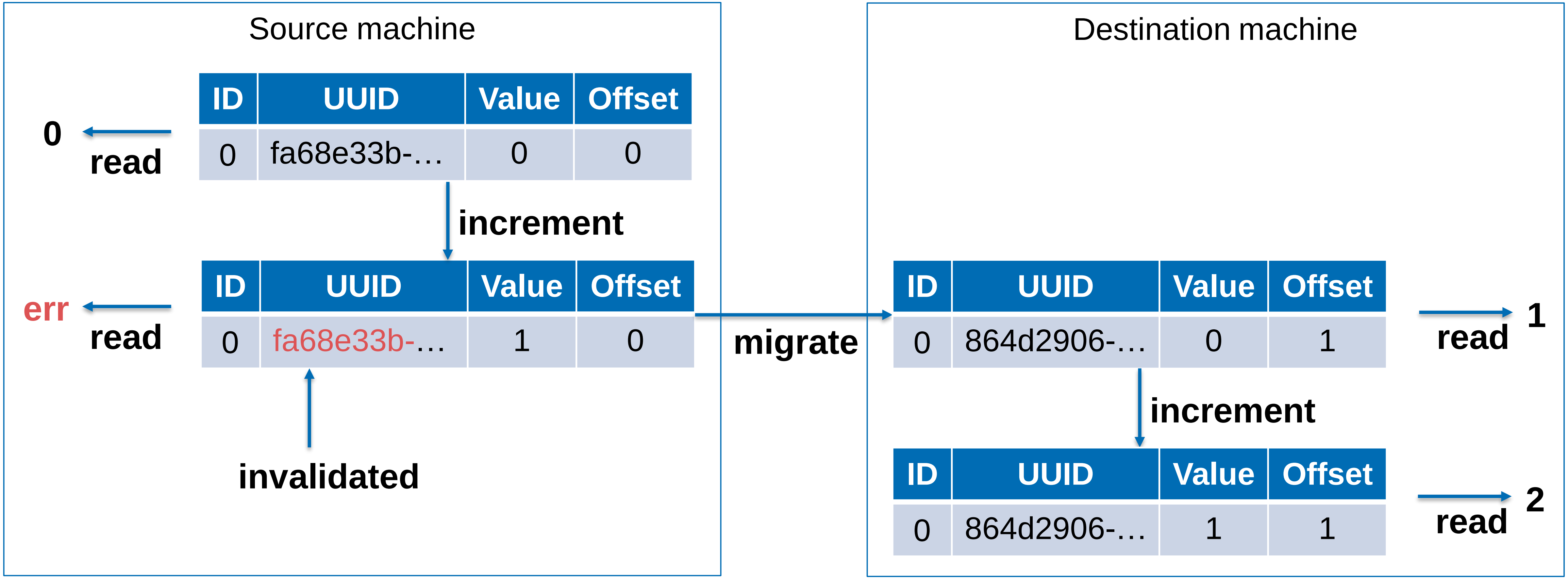
increment

1 read

ID	UUID	Value	Offset
0	fa68e33b-…	1	0

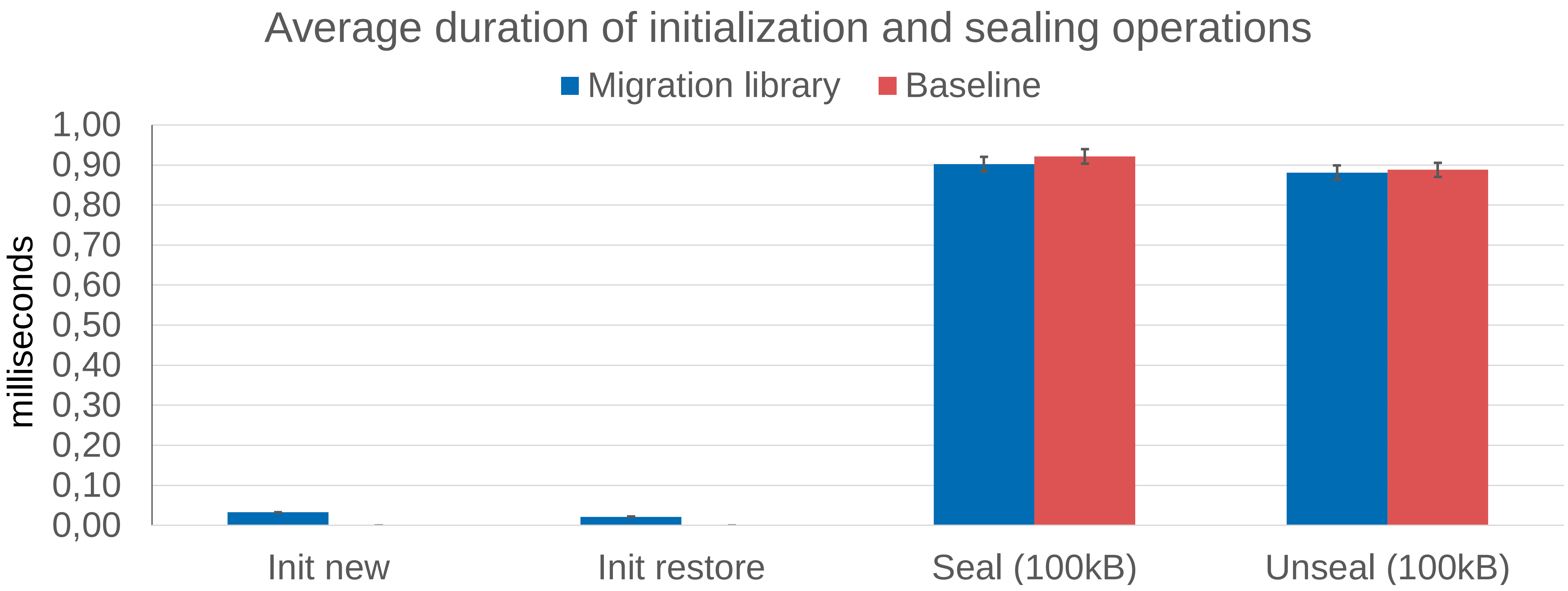
Destination machine

Migrating counters with offsets



- **No fork attacks possible (Counters get invalidated)**
- **No rollback possible (Offset static during lifetime on one machine)**

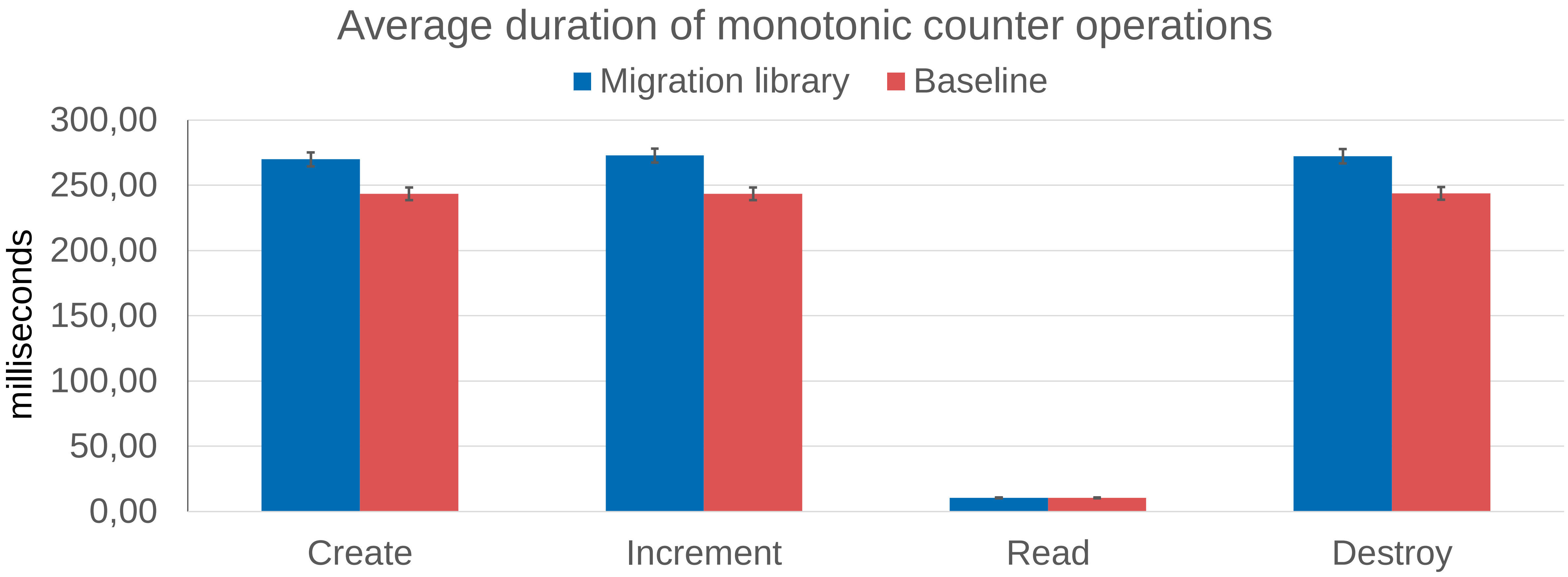
Evaluation – initialization and sealing



Full migration overhead: Fixed at ~0.5 seconds

Test setup: Intel i5-6500, 8GB RAM, 500GB HDD ; 1000 runs

Evaluation – migratable counters



Test setup: Intel i5-6500, 8GB RAM, 500GB HDD ; 1000 runs

Future work

Drawback: Migration is not transparent

- Migration Library needs to be notified to start migration process

➢ **Hypervisor could notify the library directly**

- Would make migration transparent for application and OS

Possible extension: Migrate data memory

- We focus on persistent state
- Earlier work exists on migrating data memory of an enclave
- Combining these approaches would allow to migrate enclaves without restarting them

Possible extension: Migration policies

- Developers could define policies that are to be enforced by Migration Enclave

➢ ”Only migrate to european data centers”

Summary

https://github.com/SGX-Cloud/migration

VM migration does not work directly for Intel SGX

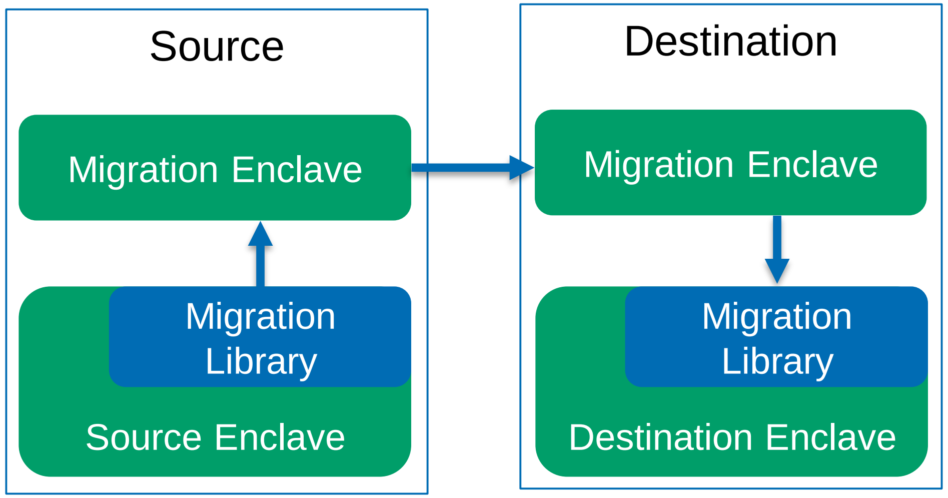
- Secrets are bound to physical hosts

Solution: Migration Library and Migration Enclave

- Migration Library: migratable versions of sealing and monotonic counter functions
- Migration Enclave: attestation and migration to destination

Pros and cons:

- Small trusted computing base (940 and 217 LOC)
- Small overhead for migratable functions
- Fixed overhead of ~0.5 seconds per migration
- But: Migration not transparent



Backup

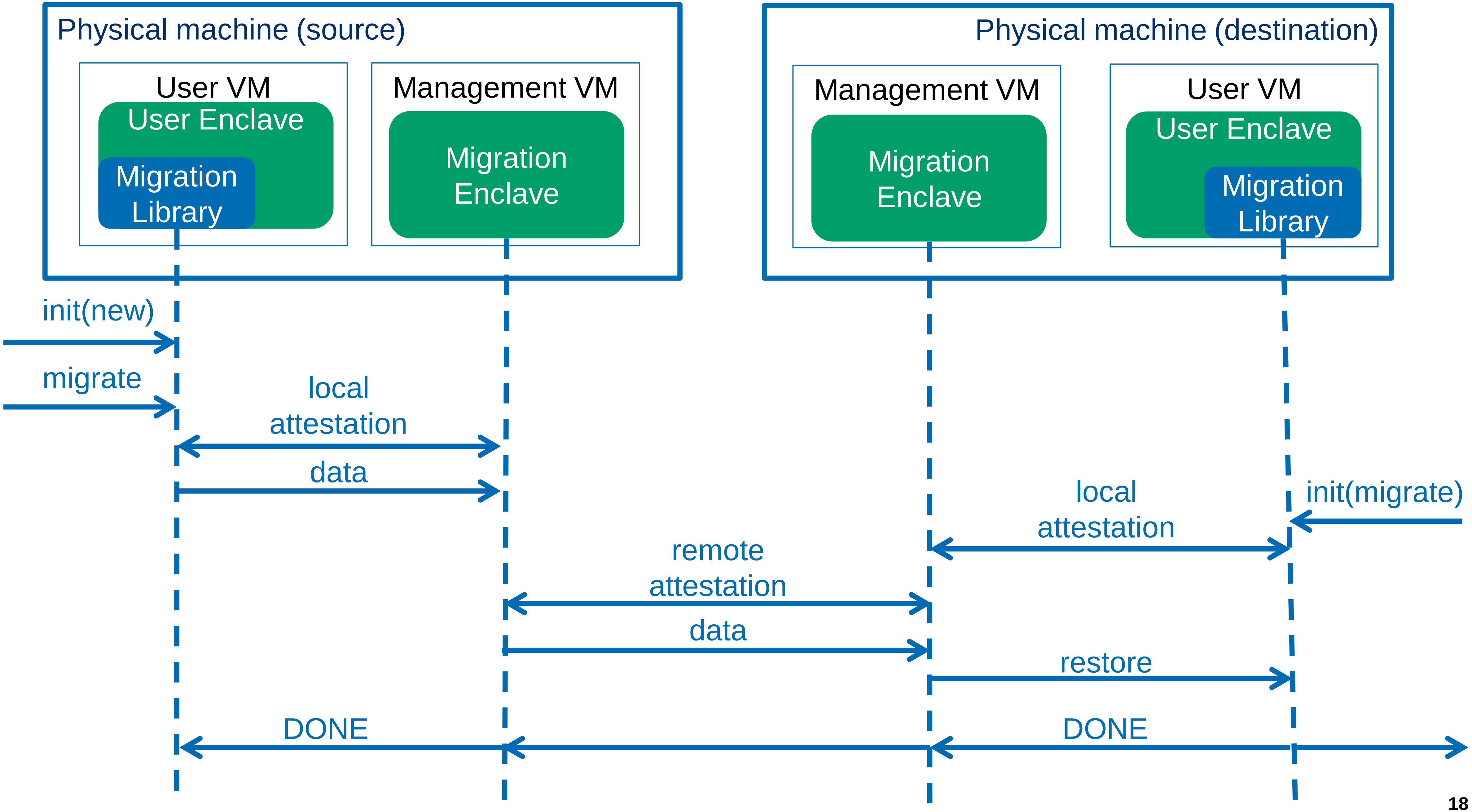
Why persistent storage?

- External storage services can store persistent data
- Local devices can poll storage services on demand

- **However:**
  1. What keys are used to authenticate and communicate with the service?
  2. How is freshness of external data guaranteed (Rollback attacks)

➢ **Small amount of persistent data is needed locally (some keys, some counters)**

➢ **Migrating this data across machines might be necessary**



Physical machine (source)
User VM
User Enclave
Migration Library
Management VM
Migration Enclave
Physical machine (destination)
Management VM
Migration Enclave
User VM
User Enclave
Migration Library
init(new)
migrate
local attestation
data
local attestation
init(migrate)
remote attestation
data
restore
DONE
DONE

Remote attestation of Migration Enclaves

Remote attestation checks whether the destination target is also a Migration Enclave

- SGX MRENCLAVE value contains hash of source code
- Accept only connections to valid versions of a Migration Enclave (hardcoded)

Attacker could run his own instance of a Migration Enclave

- Standard remote attestation only checks if the destination instance is valid (authentication)
- This would allow attackers to migrate user enclaves to machines they control

Additional authorization check needed

- Maintain a shared secret inside the Migration Enclave
- E.g. certificates provisioned by the cloud provider
- Only accept connections to Migration Enclaves that can present a valid certificate
- ➢ Requires a secure setup phase (i.e. during installation of the server)