Imperial College London

LSDS Large-Scale Distributed Systems Group

Maru: Hardware-Assisted Secure Cloud Computing

Peter Pietzuch
prp@imperial.ac.uk

Large-Scale Distributed Systems Group
Department of Computing, Imperial College London
http://lsds.doc.ic.ac.uk

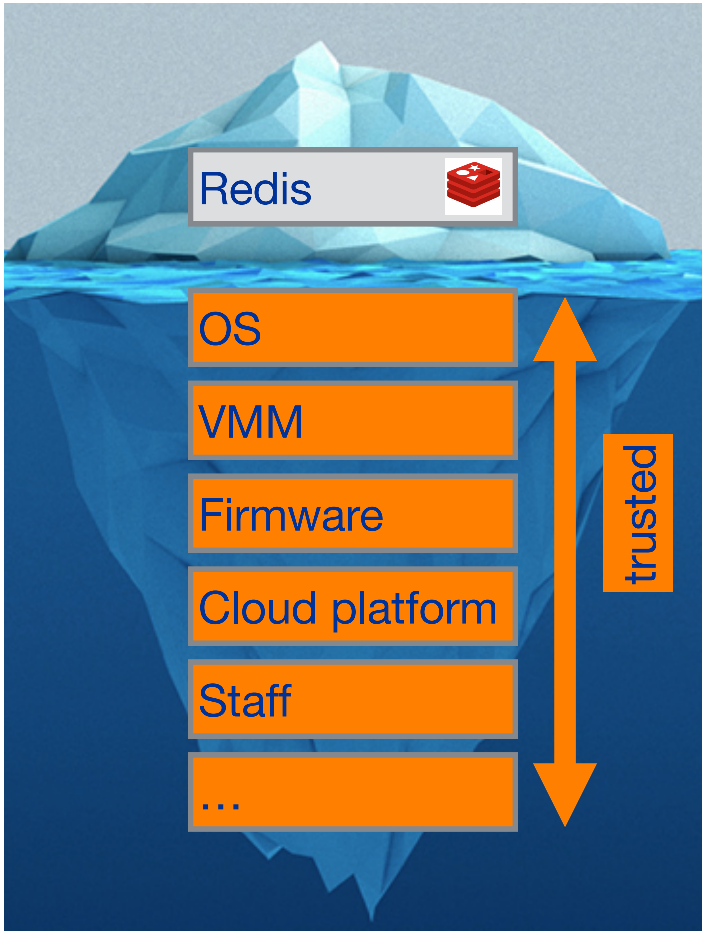
ATI – February 2017

Trust Issues: Provider Perspective

Cloud provider does not trust users

Use virtual machines to isolate users from each other and the host

VMs only provide one way protection

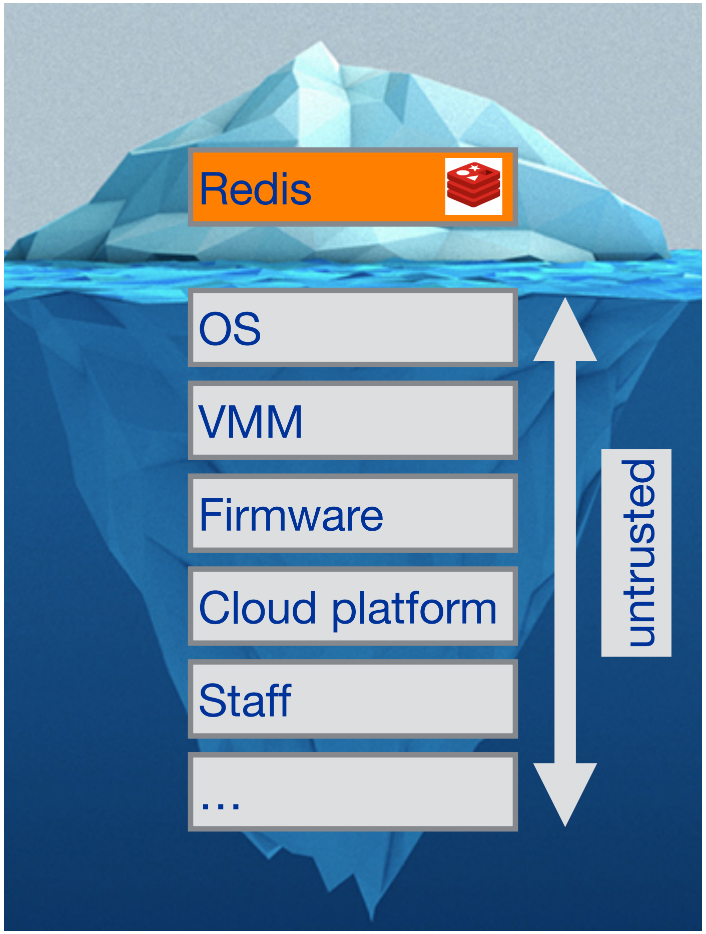


Trust Issues: User Perspective

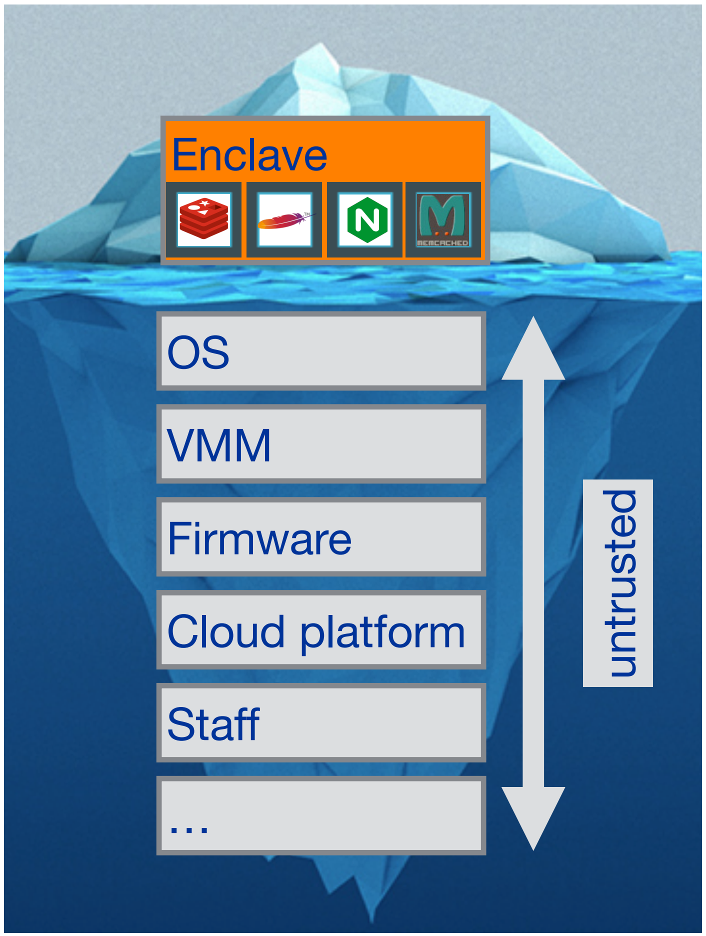
Users trust their applications

Users must implicitly trust cloud provider

Existing applications implicitly assume trusted operating system



Trusted Execution with Intel SGX



Users create HW-enforced trusted environment

Supports unprivileged user code

Protects against strong attacker model

Remote attestation

Available on commodity CPUs

Intel SGX: Hardware-Assisted Security

New **enclave** processor mode

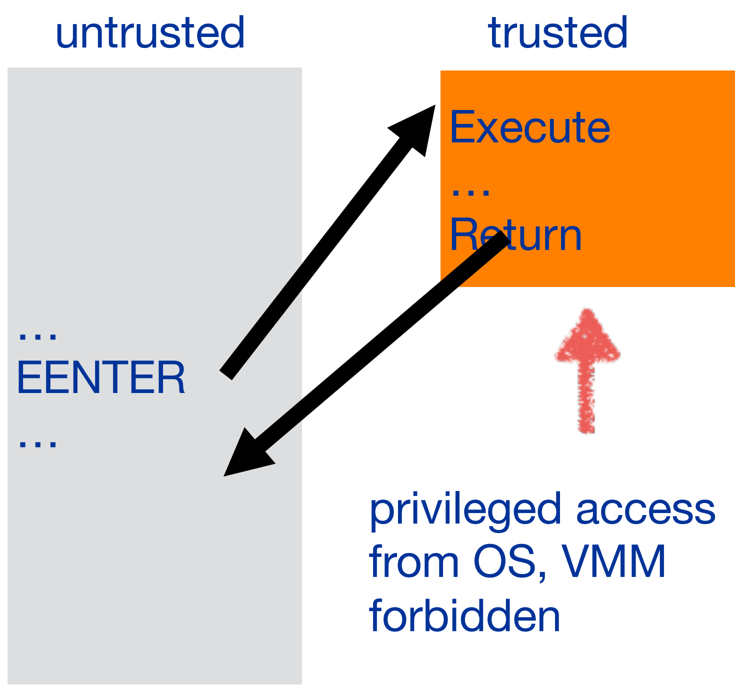
18 new instructions to manage enclave life cycle

Enclave memory only accessible from enclave

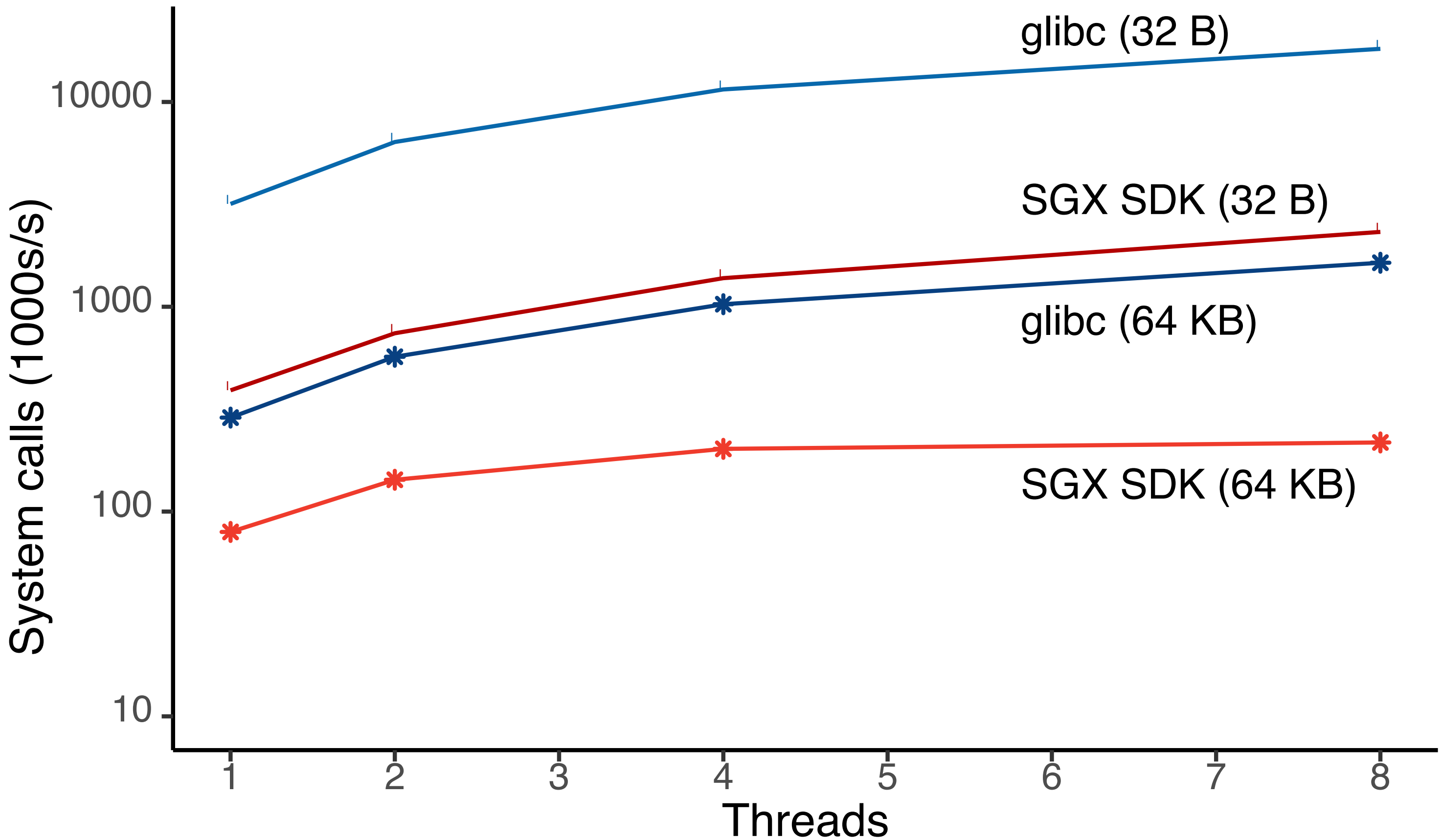
Certain instructions disallowed, e.g., **syscall**

No system calls

Performance overhead

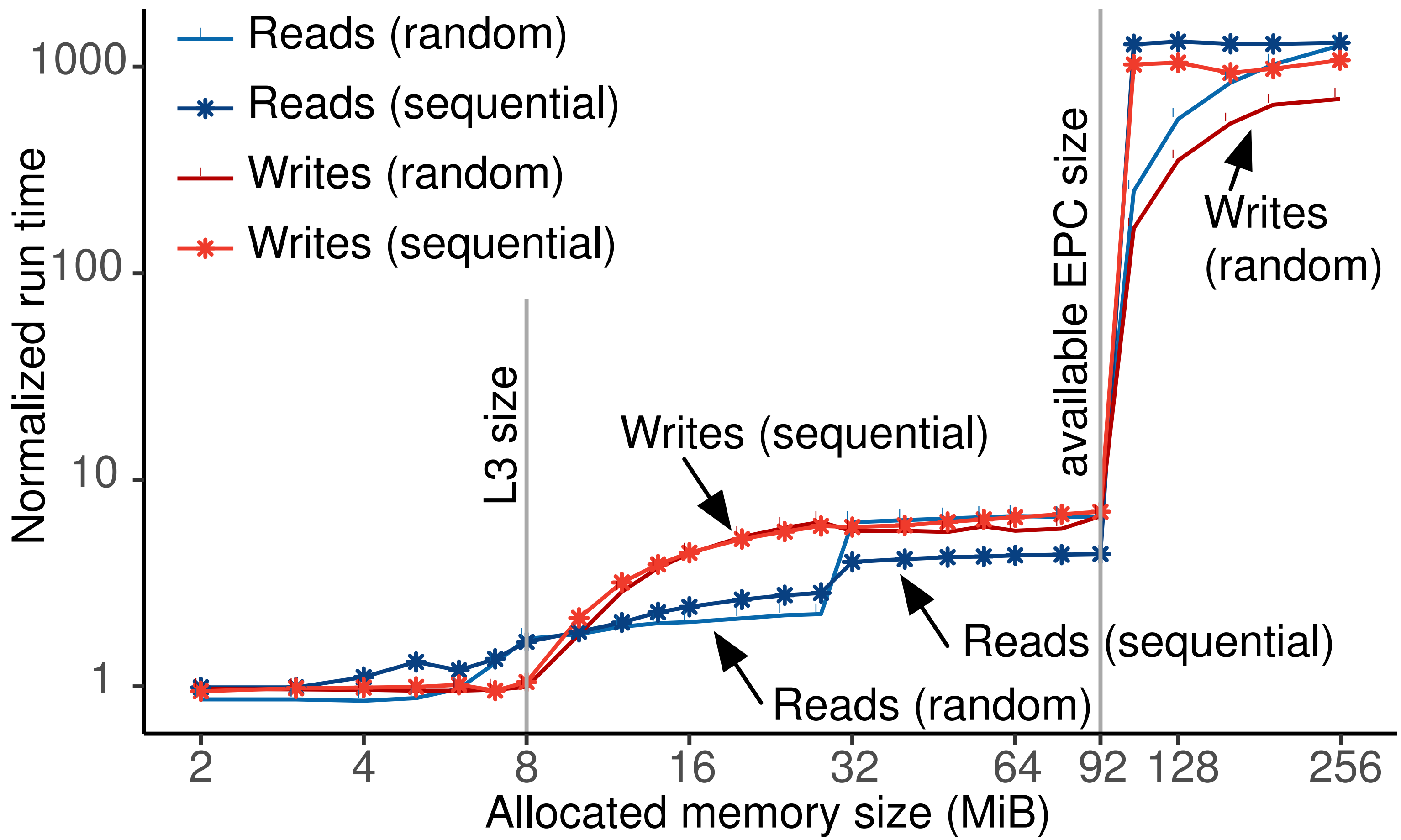


SGX: System Call Overhead (pwrite)



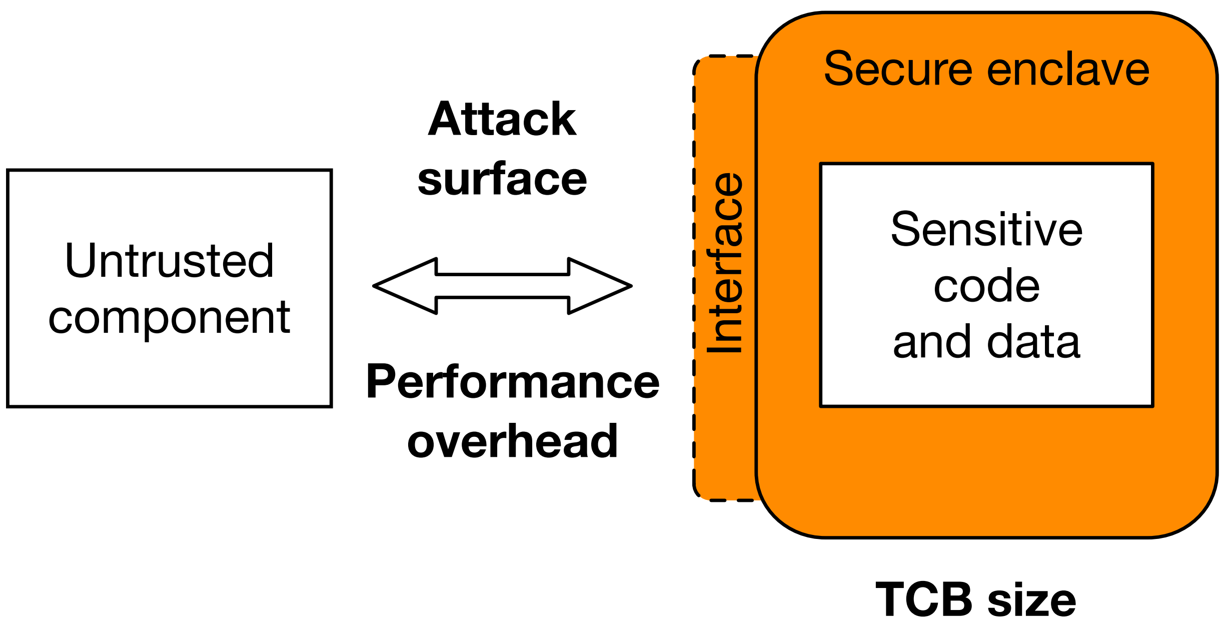
System calls outside of enclave are expensive

SGX: Memory Access Overhead

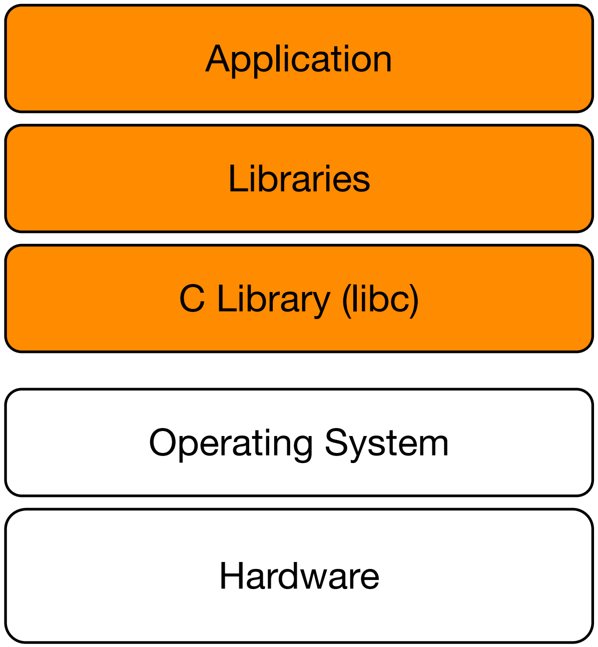


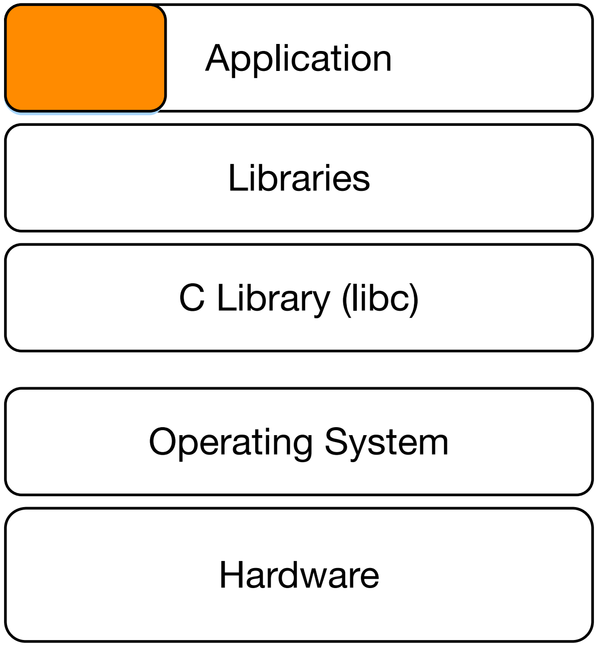
Large amount of enclave memory leads to poor performance

SGX Research Challenges



Systems Support for SGX?

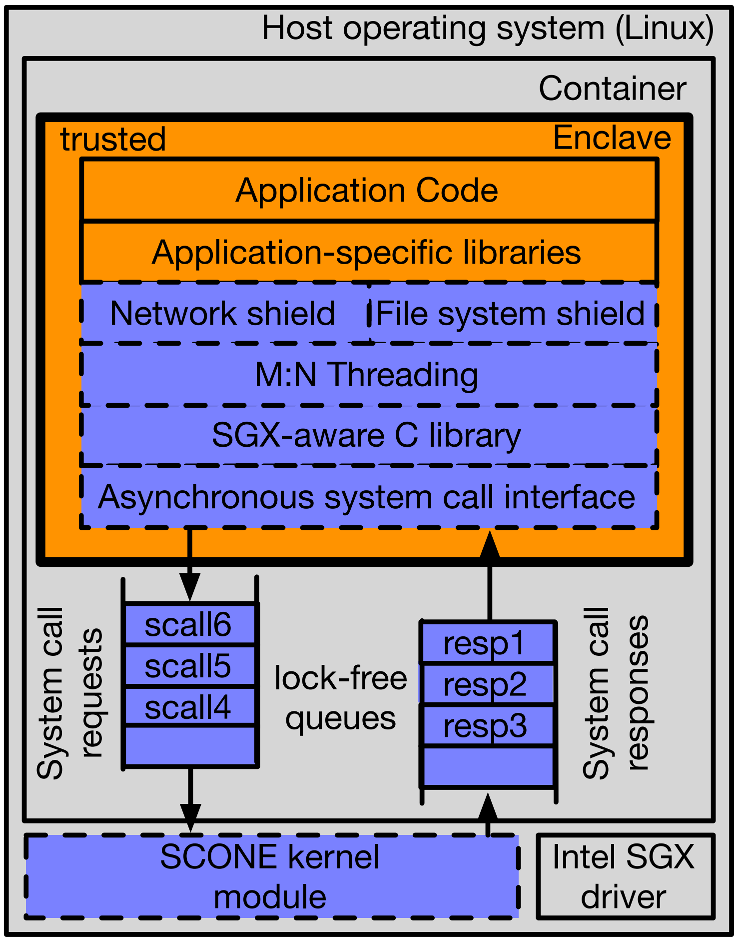




I. Complete unmodified applications in enclaves
(Systems support?)

II. Privilege Separation
(Minimal TCB?)

1. SCONE: Secure CONtainer Environment



1. Good performance/security trade-off

- Small TCB (0.8×–2.1× of native size)
- Low overhead (0.3×–1.1× of native throughput)

2. Efficient system call support

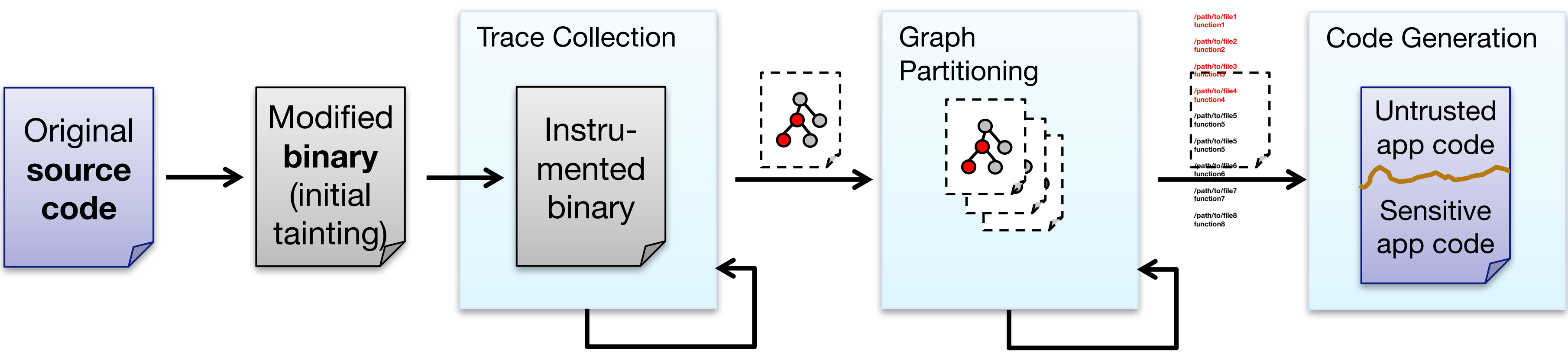
- M:N user-level threading
- Asynchronous syscall execution

3. Transparent interface shielding

- Encryption of file descriptors
- TLS support for network sockets
- Encrypted data stored outside enclave

2. Glamdring: Application Partitioning

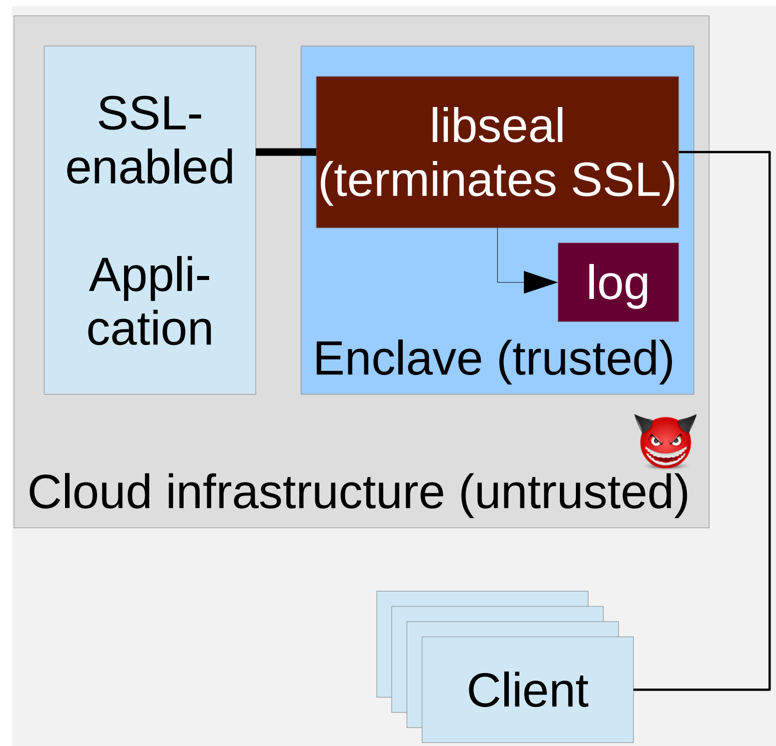
1. Static / Dynamic Analysis	2. Graph partitioning	3. Automated source-to-source code transform
Collect information to obtain valid partitioning	Find partitioning of application	Implement partitioning using Intel SGX SDK



3. LibSEAL: Secure Auditing Library

LibSEAL: Secure TLS Auditing Library

- Provide accountability to TLS-enabled application
- Help link integrity violations to origin



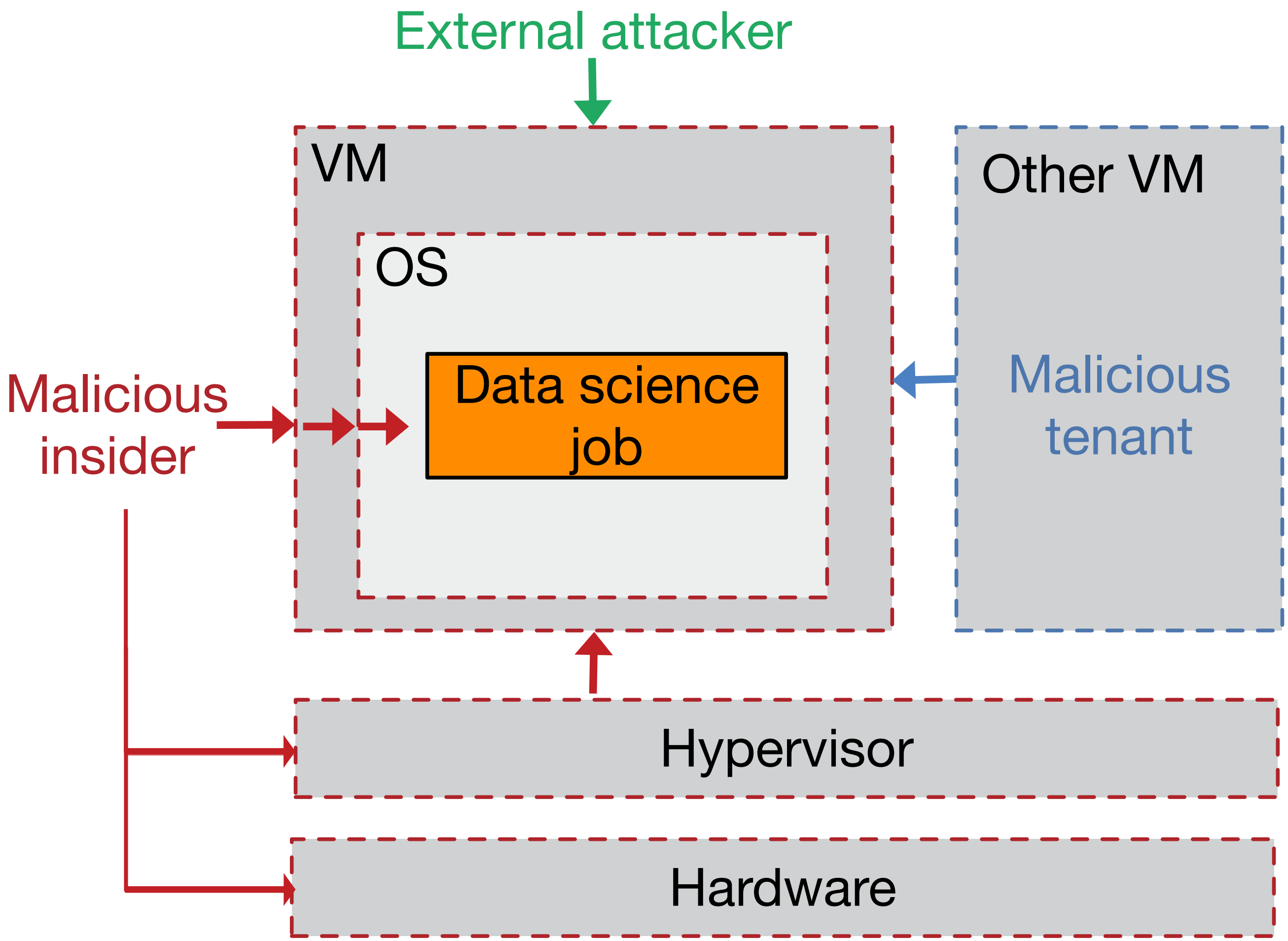
Workflow:

1. Securely log communication between client and service
2. Audit against application-specific invariants

Use cases:

- Dropbox: Have files been lost?
- Git: Is the the server hiding commits?
- Owncloud: Were there illegitimate modifications to content or layout?

Maru: Security Threats in Data Science



Maru Research Directions

1. **Security model for shielded data science jobs**
   - How to harden shielded jobs? How to deal with vulnerabilities, bugs?
   - What about external dependencies/libraries?
2. **Integration of language runtimes with secure enclaves**
   - How to integrate SGX support for the JVM?
   - What is the right programming model for SGX enclaves?
3. **Unikernel support for secure enclaves**
   - How to support existing legacy binaries?
   - How to build type-safe minimal secure enclaves for data science jobs?
4. **Prototype platform implementation and evaluation**
   - Integration with Apache Flink or other dataflow frameworks
5. **Dataflow attacks and mitigations strategies**
   - What attacks are possible by observing encrypted dataflows?
   - Can we apply techniques for unobservable communication?