TrustFS: An SGX-enabled Stackable File System Framework

1st Workshop on Distributed and Reliable Storage Systems (DRSS'19)
Lyon, 1st October 2019

Tânia Esteves[1], Ricardo Macedo[1], Alberto Faria[1], Bernardo Portela[2],
João Paulo[1], José Pereira[1] and Danny Harnik[3]

[1] INESC TEC and University of Minho, Portugal. [2] INESC TEC and University of Porto, Portugal.
[3] IBM Research – Haifa, Israel.

INESCTEC

Universidade do Minho

CONTEXTUALIZATION

- **Exponential growth** of digital information
- Need for ensuring **data confidentiality**
- Need for applying **content-aware functionalities** (for space reduction and query optimizations)
  - *E.g.*, deduplication, compression, indexing, *etc.*

CHALLENGES AND PROBLEMS

For ensuring data confidentiality, we can:

CHALLENGES AND PROBLEMS

For ensuring data confidentiality, we can:

- Encrypt the data before storing on a third-party storage, but…

CHALLENGES AND PROBLEMS

For ensuring data confidentiality, we can:

- Encrypt the data before storing on a third-party storage, but…

Encryption **limits** the use of **content-aware** functionalities

CHALLENGES AND PROBLEMS

For ensuring data confidentiality, we can:

- Encrypt the data before storing on a third-party storage, but…

Encryption **limits** the use of **content-aware** functionalities

How to ensure data confidentiality and privacy while allowing content-aware computations?

CURRENT SOLUTIONS

- Use of **property-preserving** schemes
  - *e.g., convergent encryption for deduplication*

CURRENT SOLUTIONS

- Use of **property-preserving** schemes
  - *e.g., convergent encryption for deduplication*

Weaker security guarantees.
Still limited.

CURRENT SOLUTIONS

- Use of **property-preserving** schemes
  - *e.g., convergent encryption for deduplication*

- Use **trusted hardware**
  - *e.g., Intel SGX*

Weaker security guarantees. Still limited.

CURRENT SOLUTIONS

- Use of **property-preserving** schemes
  - *e.g., convergent encryption for deduplication*

Weaker security guarantees.
Still limited.

- Use **trusted hardware**
  - *e.g., Intel SGX*

Hardware dependent.
No unified framework.

CURRENT SOLUTIONS

- Use of **property-preserving** schemes
  - *e.g., convergent encryption for deduplication*

Weaker security guarantees.
Still limited.

- Use **trusted hardware**
  - *e.g., Intel SGX*

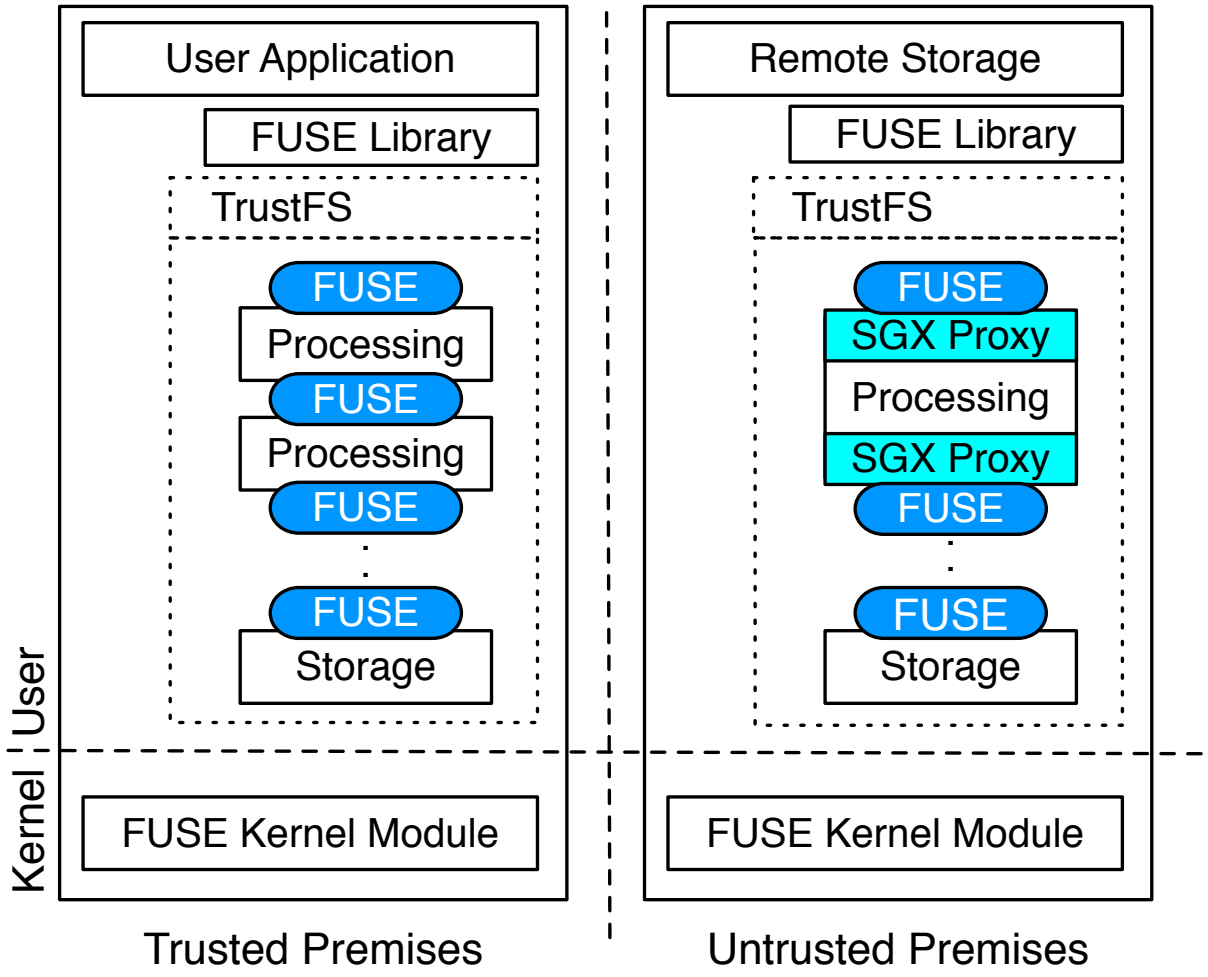
Hardware dependent.
No unified framework.

How can this be done without requiring a deep reimplementation of existing storage solutions?

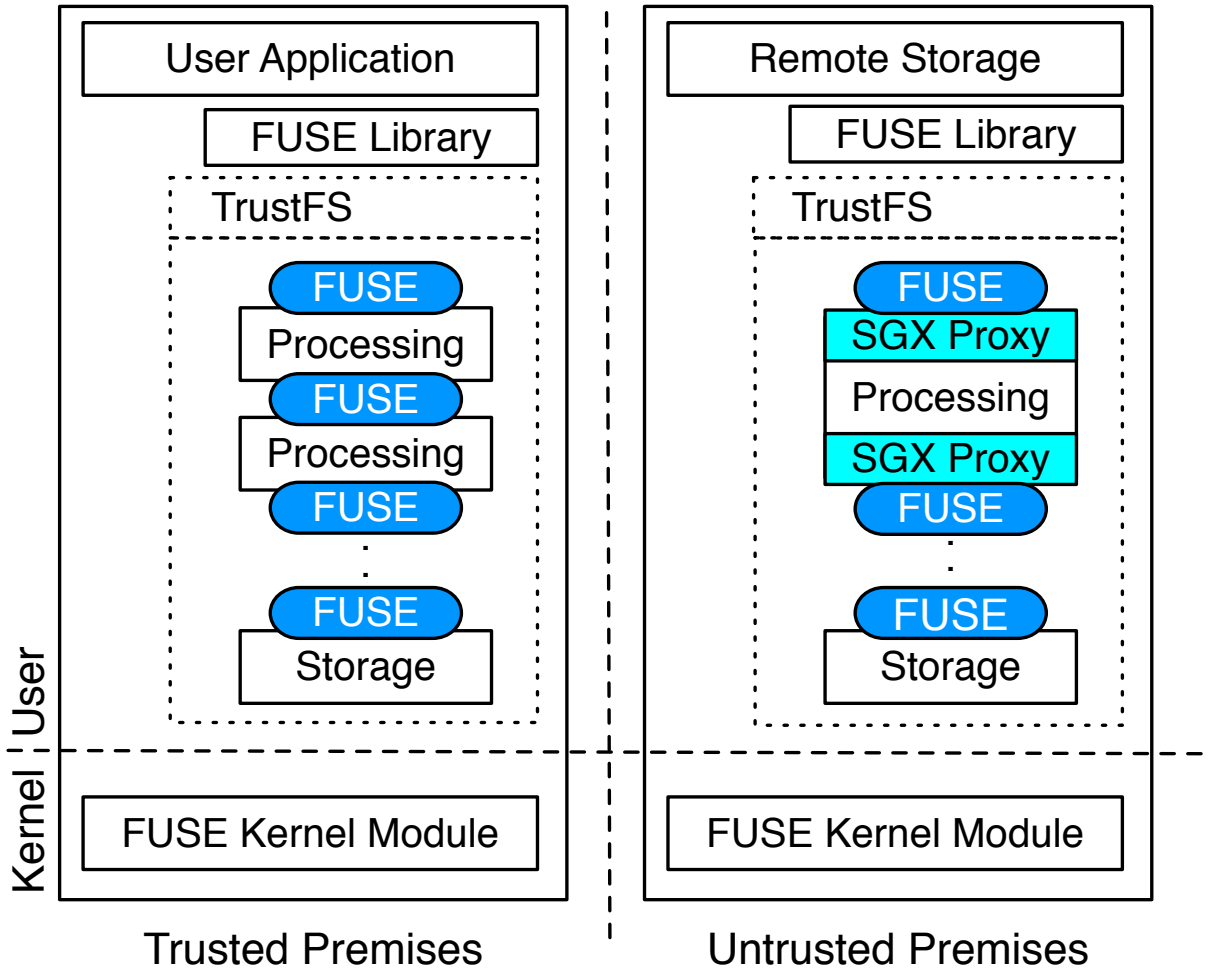
CONTRIBUTIONS

- **TRUSTFS**
  - An SGX-enabled stackable file system framework
  - Initial prototype and preliminary evaluation
  - Discussion of open issues and future directions

TRUSTFS ARCHITECTURE

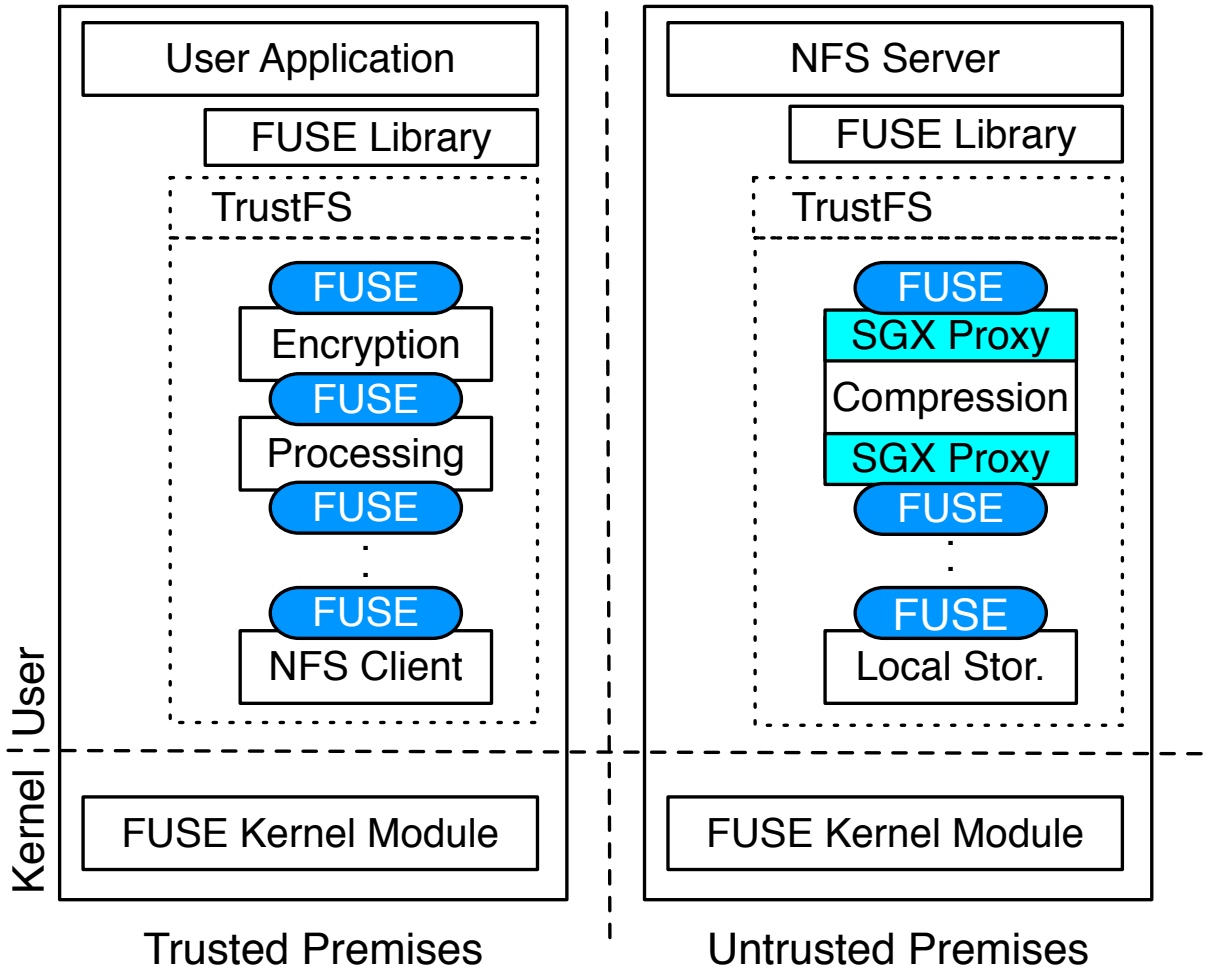


TRUSTFS ARCHITECTURE

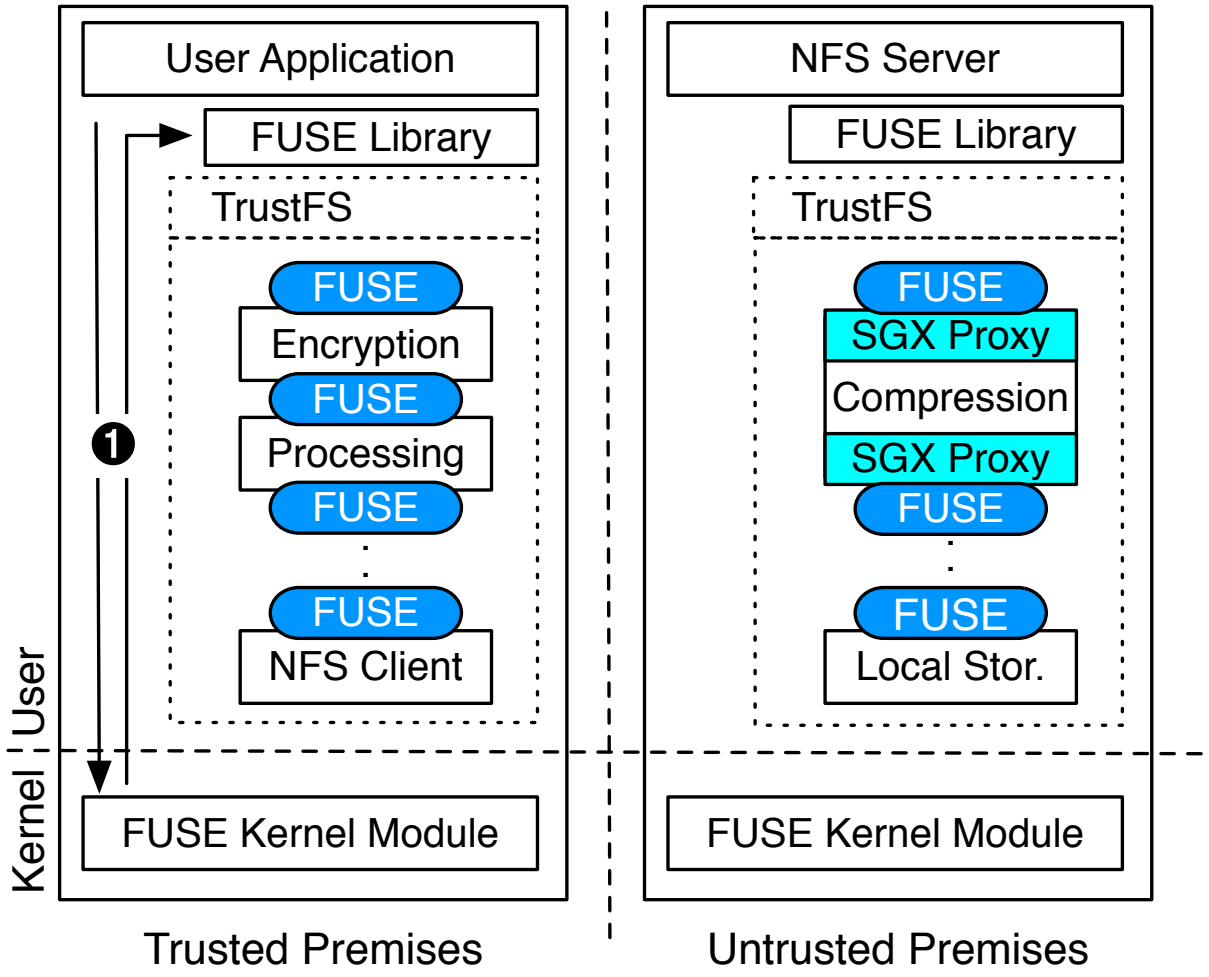


- Processing and storage layers
- Drivers with different algorithms
- SGX Proxy

TRUSTFS FLOW OF REQUESTS

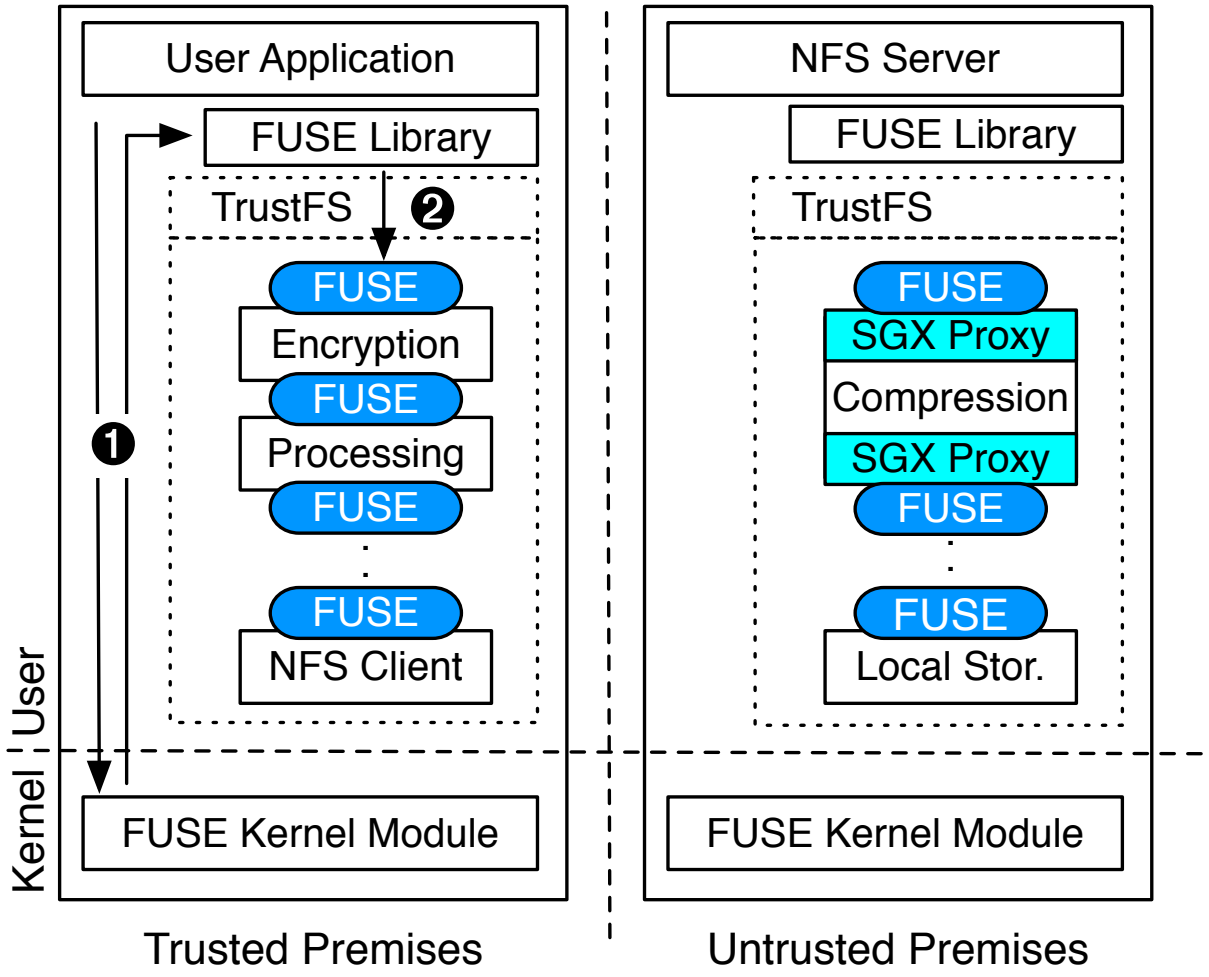


TRUSTFS FLOW OF REQUESTS



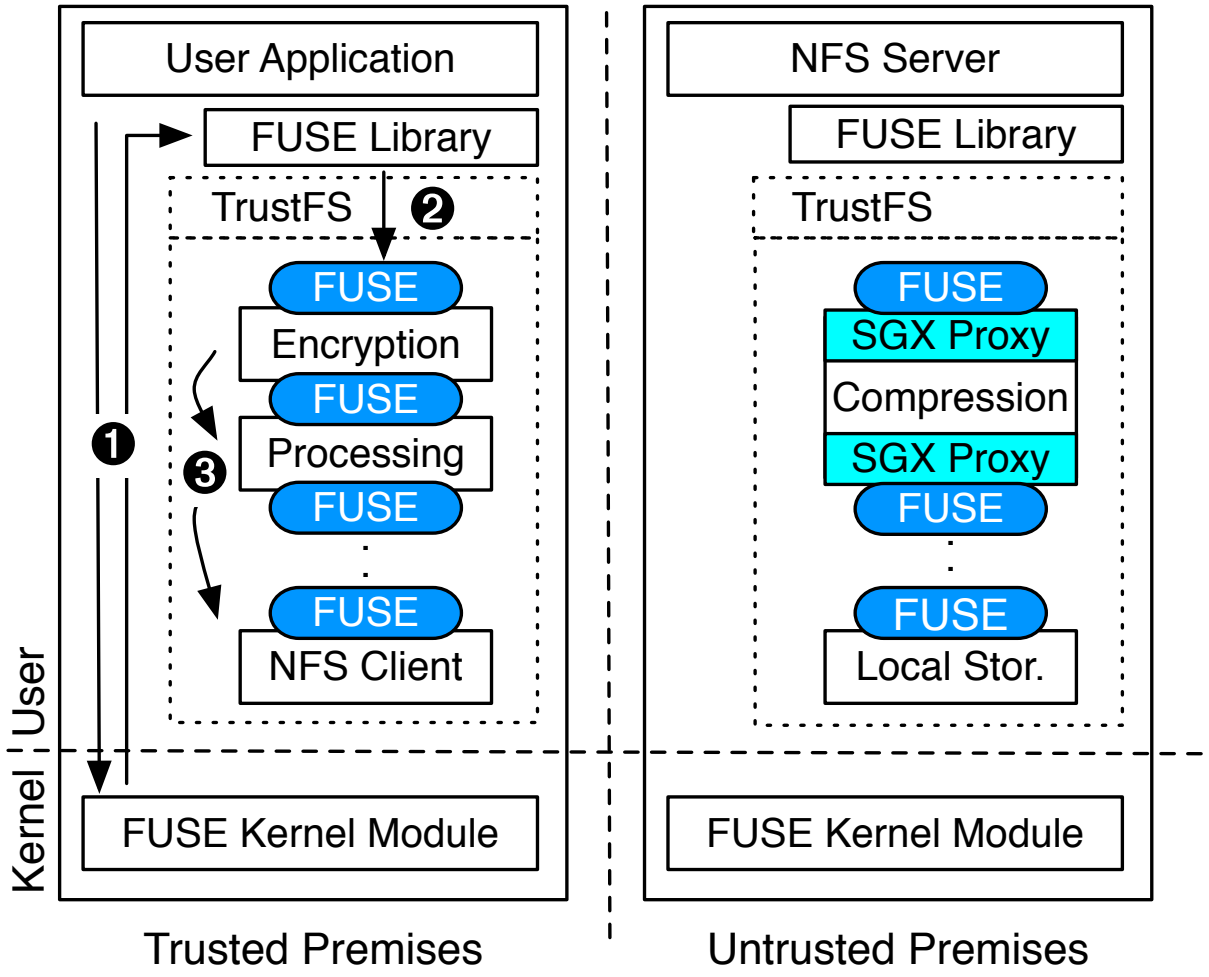
Operations are intercepted by the FUSE kernel module

TRUSTFS FLOW OF REQUESTS



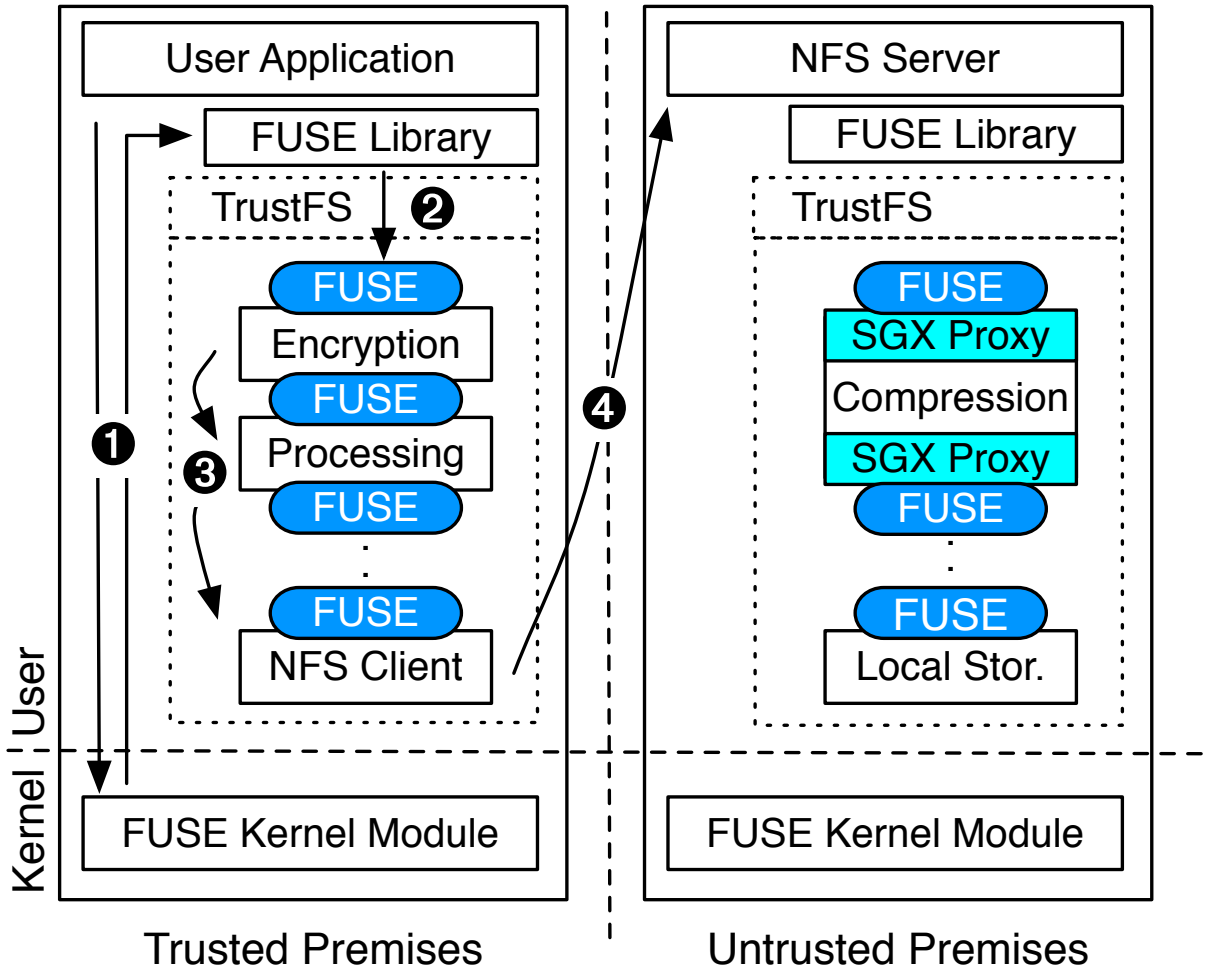
and redirected to the corresponding TRUSTFS user-space daemon.

TRUSTFS FLOW OF REQUESTS



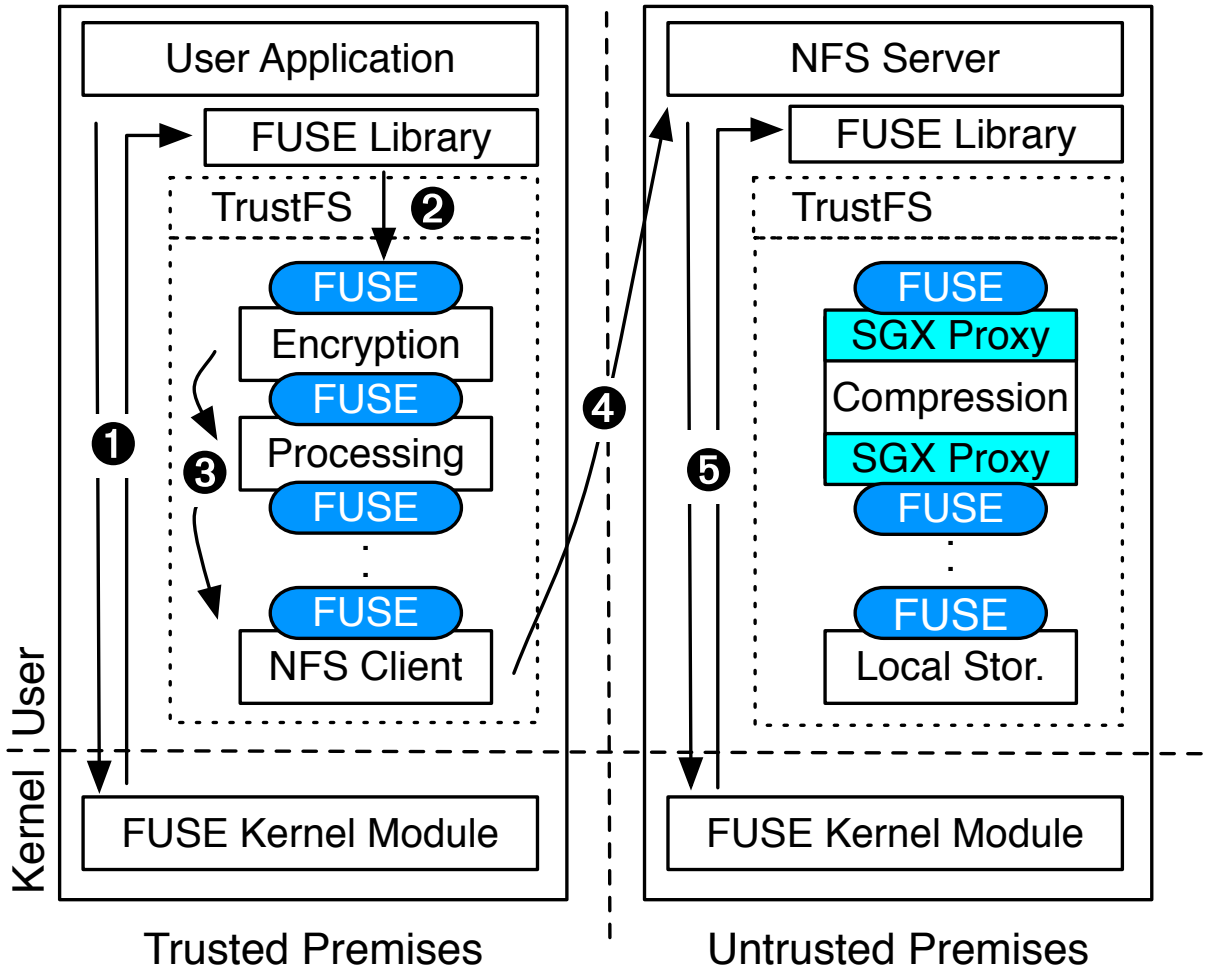
Then, requests are encrypted by a privacy-preserving layer,

TRUSTFS FLOW OF REQUESTS



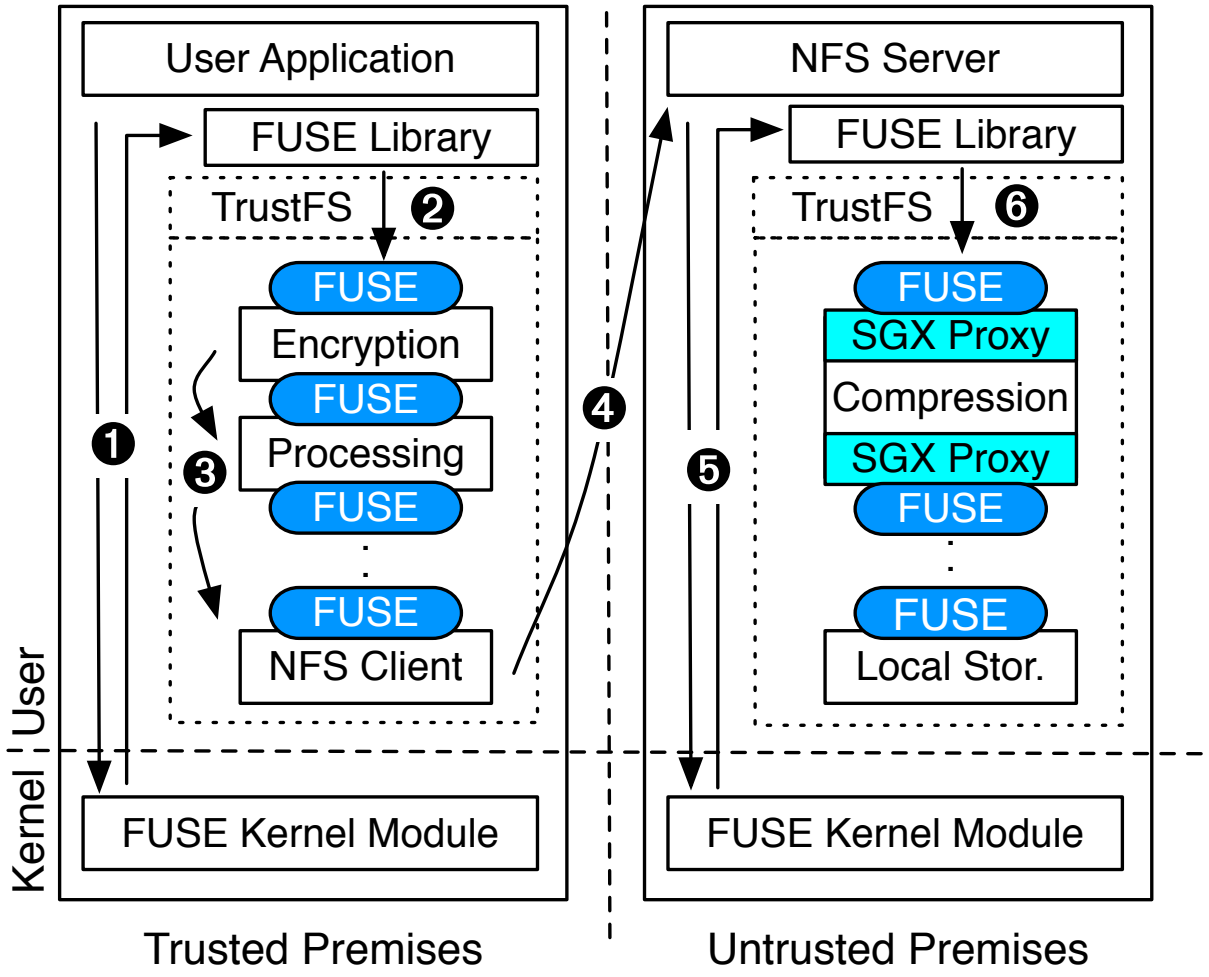
forwarded to a terminal layer, and sent to the server via a remote storage protocol.

TRUSTFS FLOW OF REQUESTS



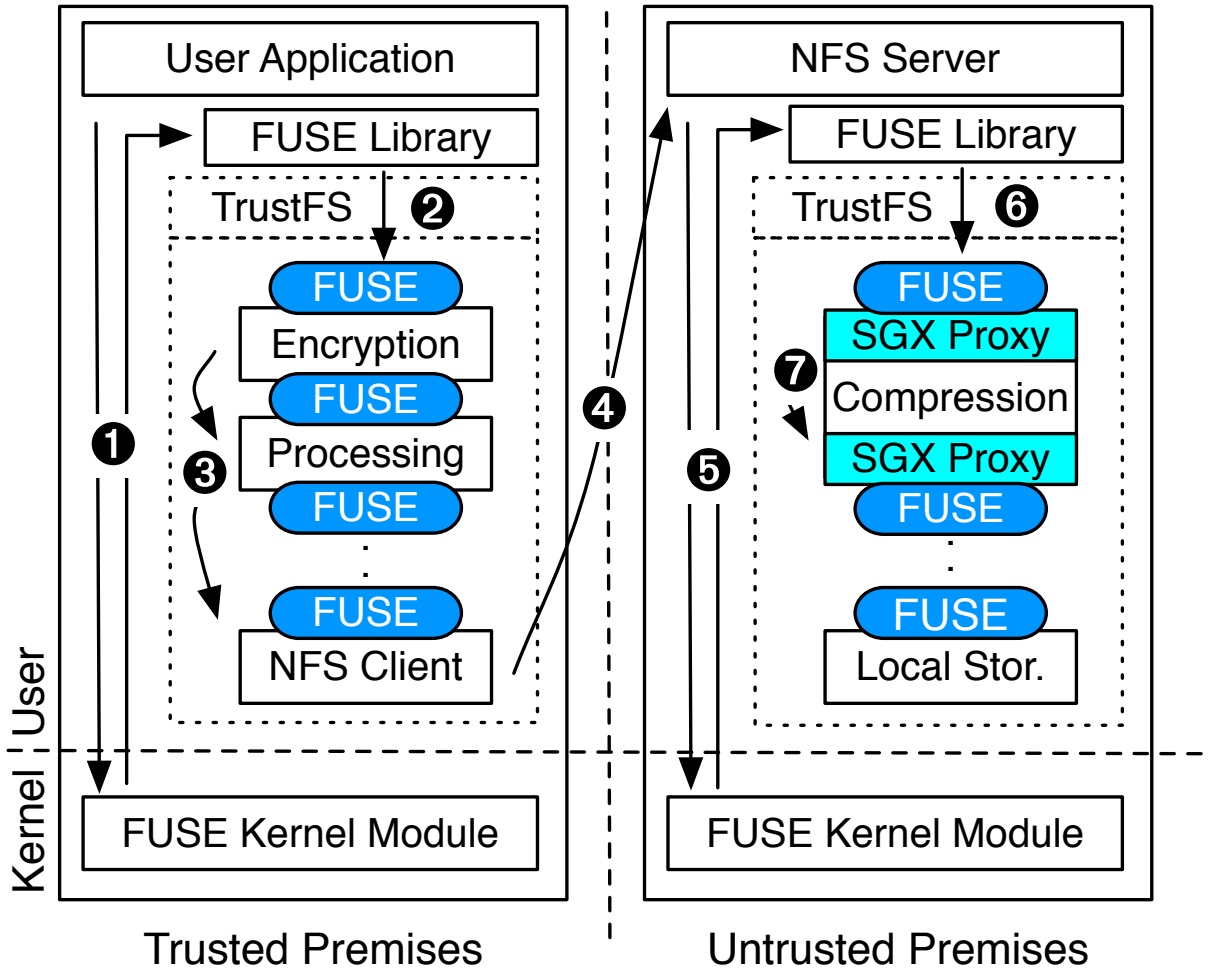
At the server-side, data is stored and retrieved from another TRUSTFS stack.

TRUSTFS FLOW OF REQUESTS



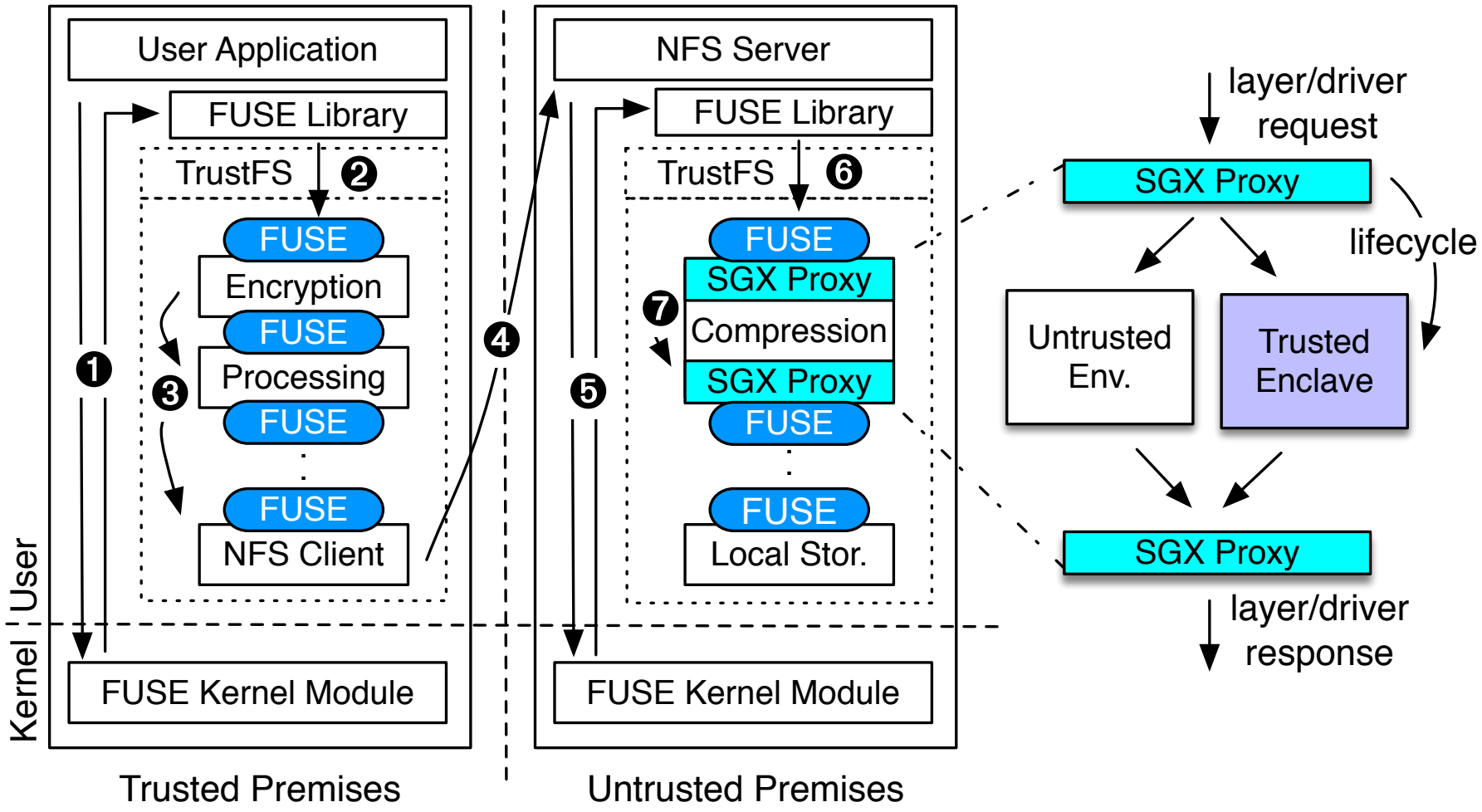
Requests reach the topmost layer of the stack,

TRUSTFS FLOW OF REQUESTS

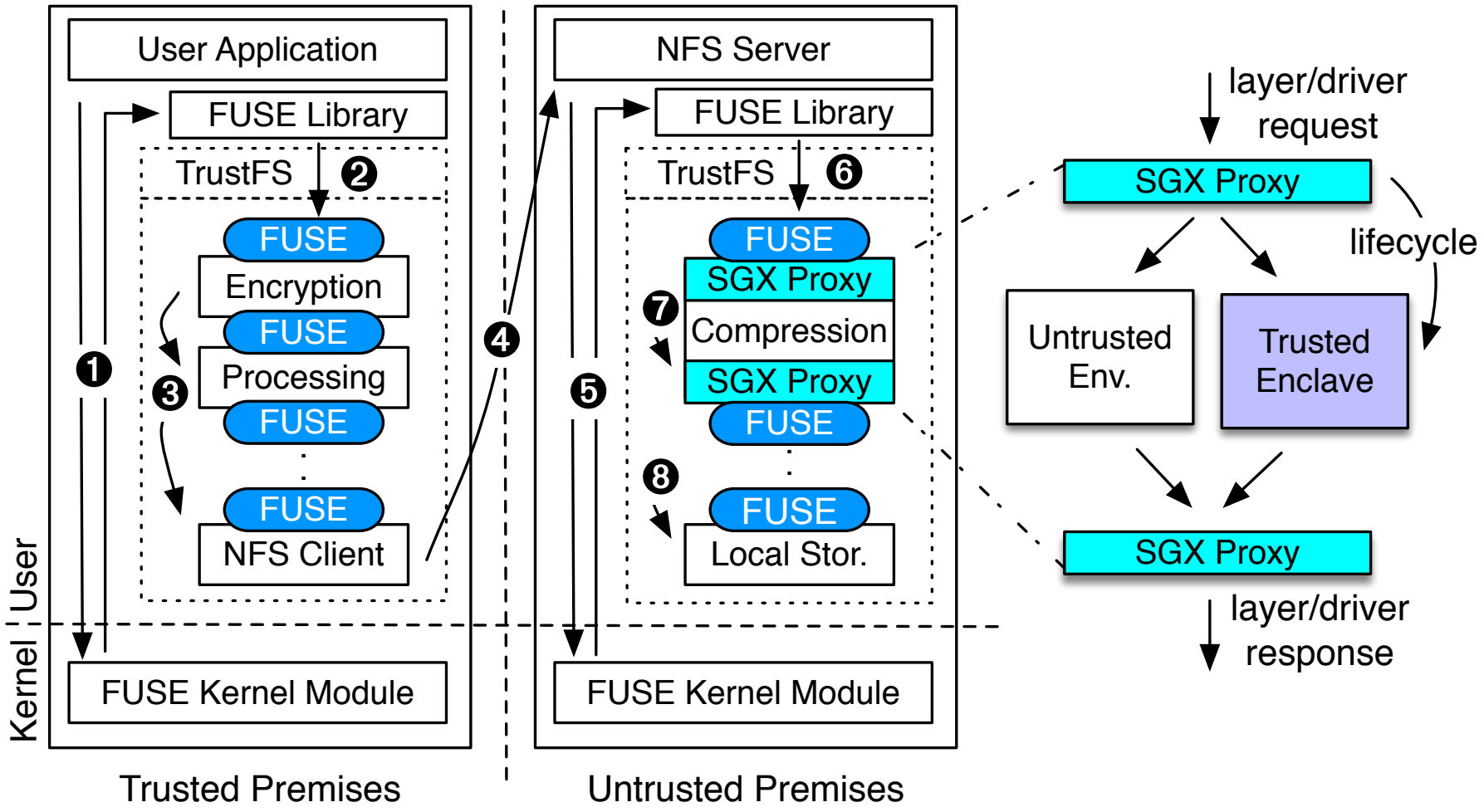


And are handled by the SGX proxy.

TRUSTFS FLOW OF REQUESTS



TRUSTFS FLOW OF REQUESTS



Finally, data is persisted in a storage medium.

TRUSTFS IMPLEMENTATION

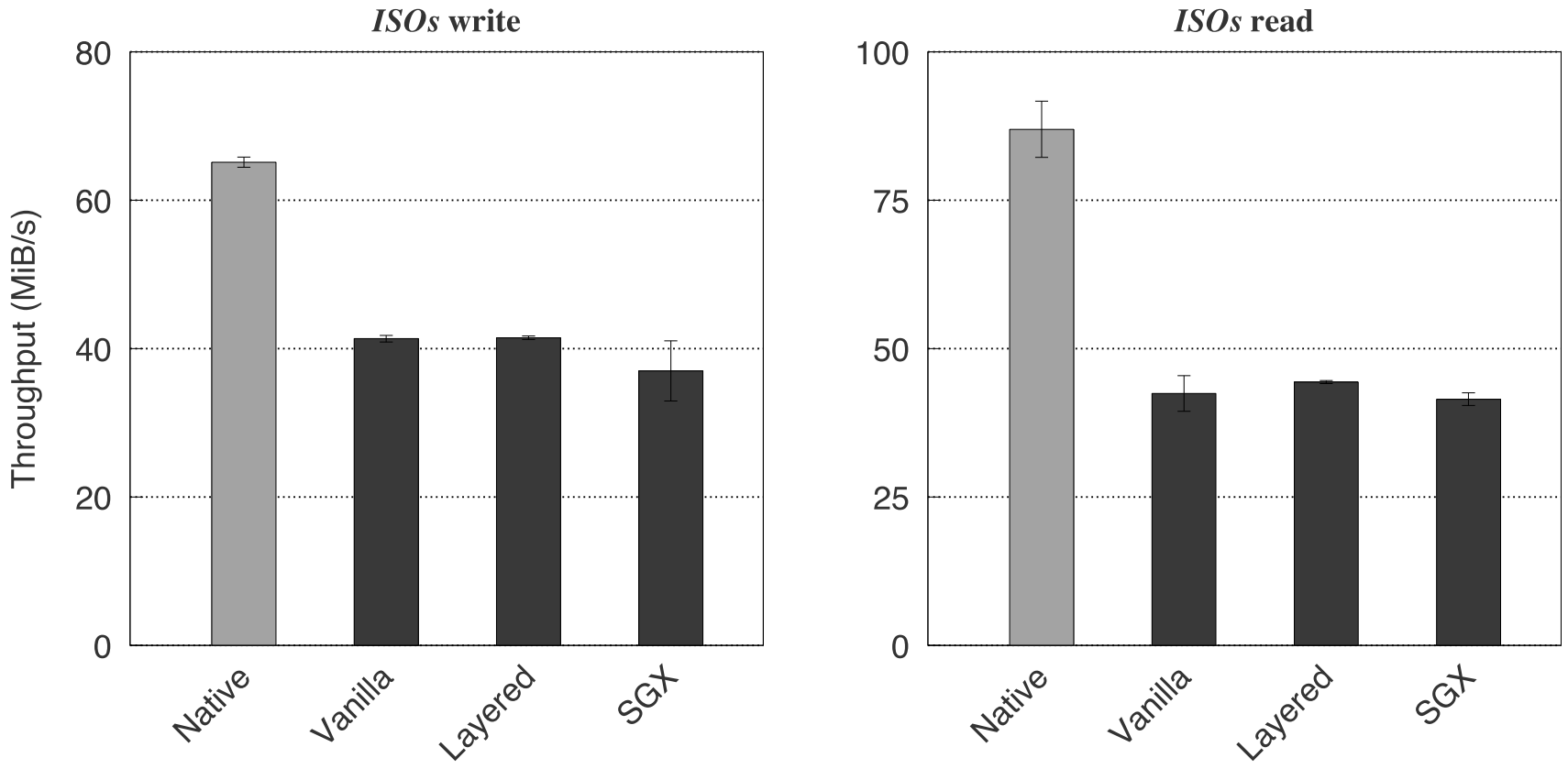
- Integration of the FUSECOMPRESS file system as a novel TRUSTFS layer
  - Less than 230 of 5276 LoC modified
- Development of a SGX-enabled driver for LZO algorithm
  - Less than 200 LoC added

PRELIMINARY EVALUATION

- Four setups:
  - *Native*, *Vanilla*, *Layered* and *SGX*
- Two dumps:
  - 21 *ISO* images (22.3GiB) and 20 Linux *Kernel* source code releases (4.5GiB)
- Four workloads:
  - *ISOs* write, *ISOs* read, *Kernels* write and *Kernels* read
- 3 runs for each experiment

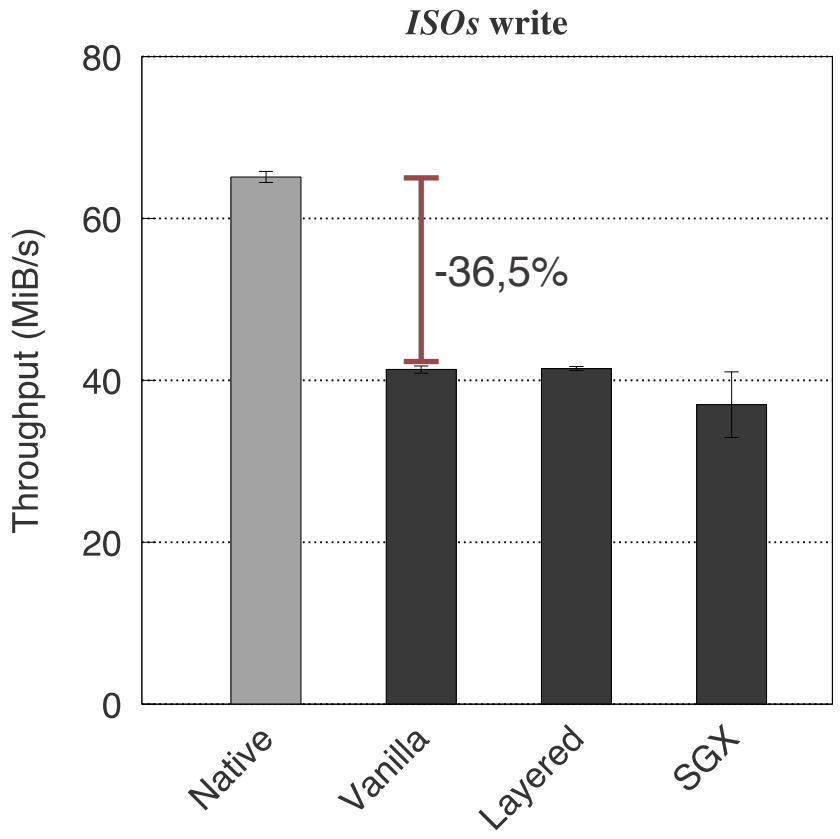
PRELIMINARY EVALUATION

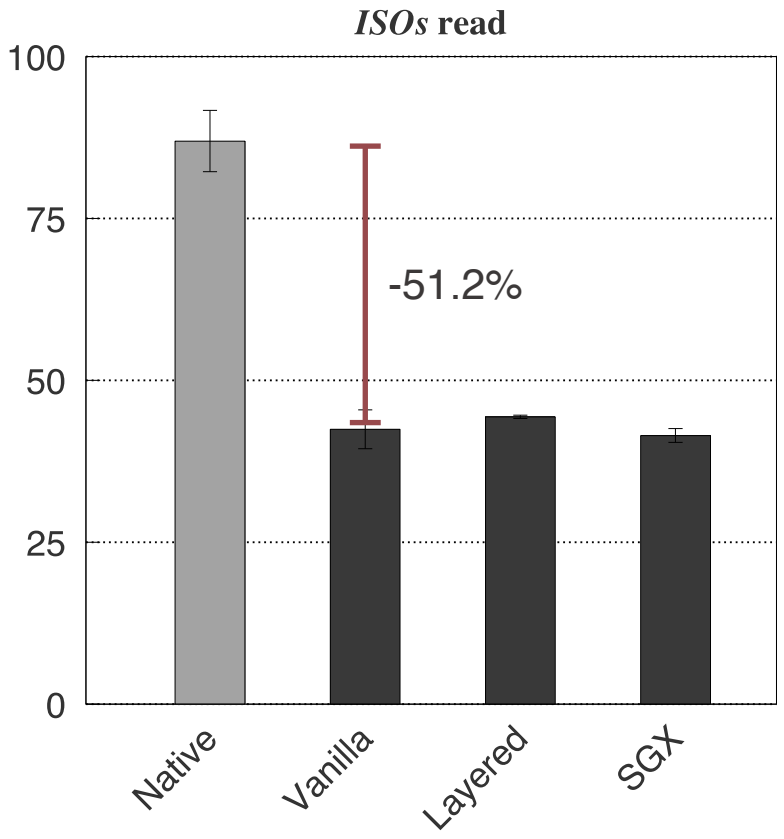
ISO DATA DUMPS



PRELIMINARY EVALUATION

ISO DATA DUMPS

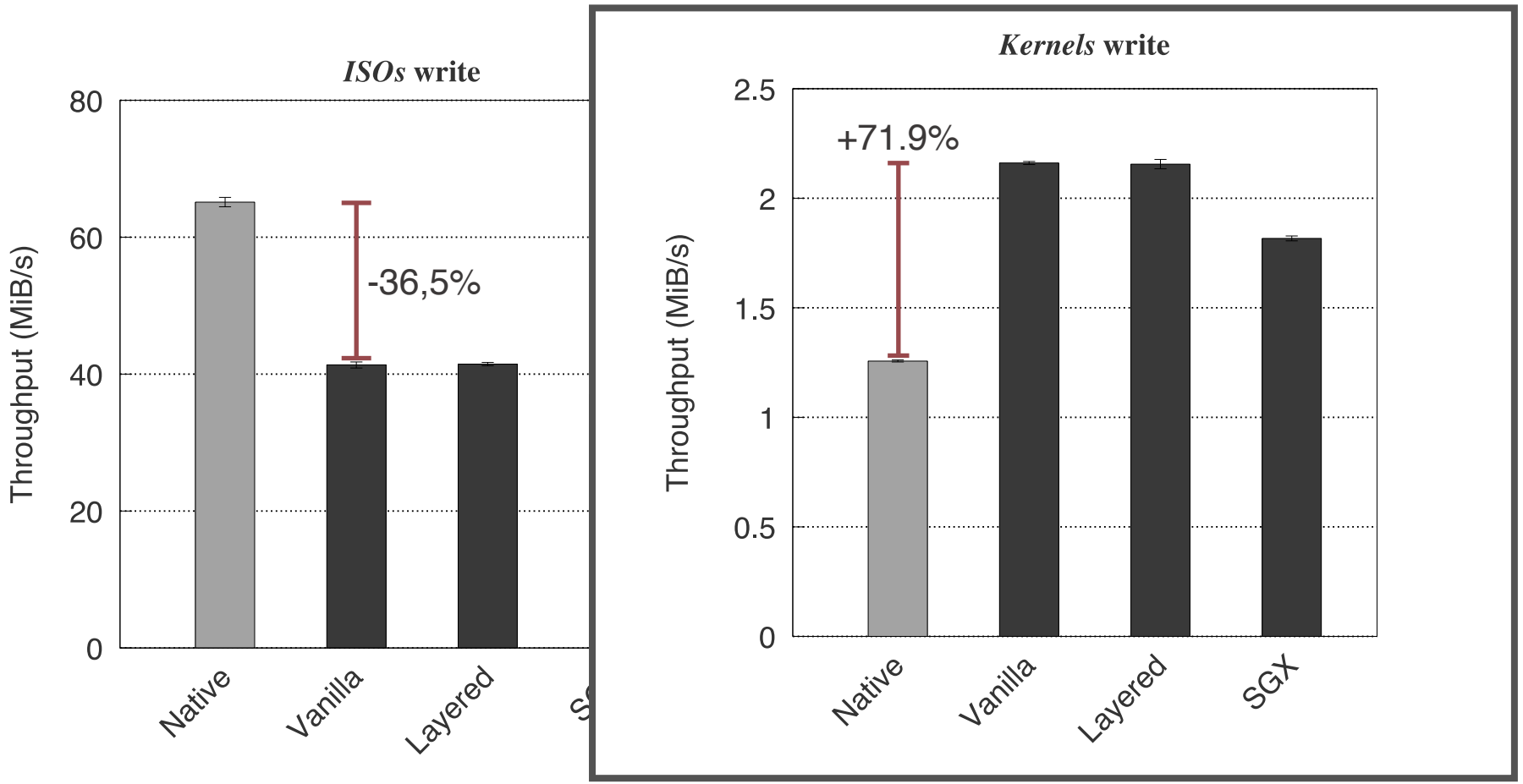




- Throughput degradation from Native setup to Vanilla setup

PRELIMINARY EVALUATION

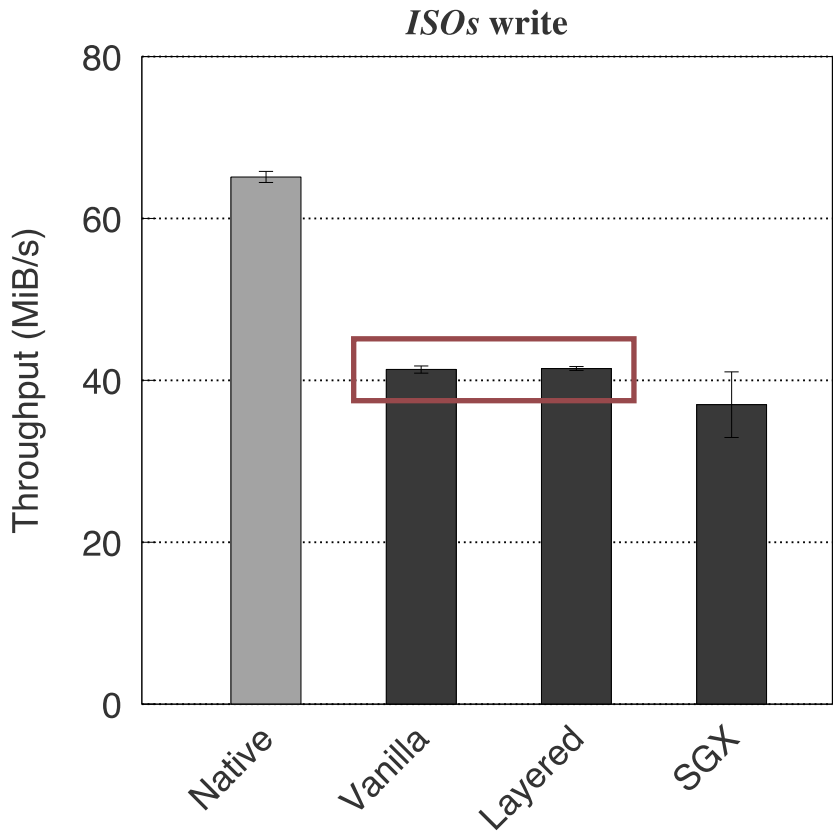
ISO DATA DUMPS

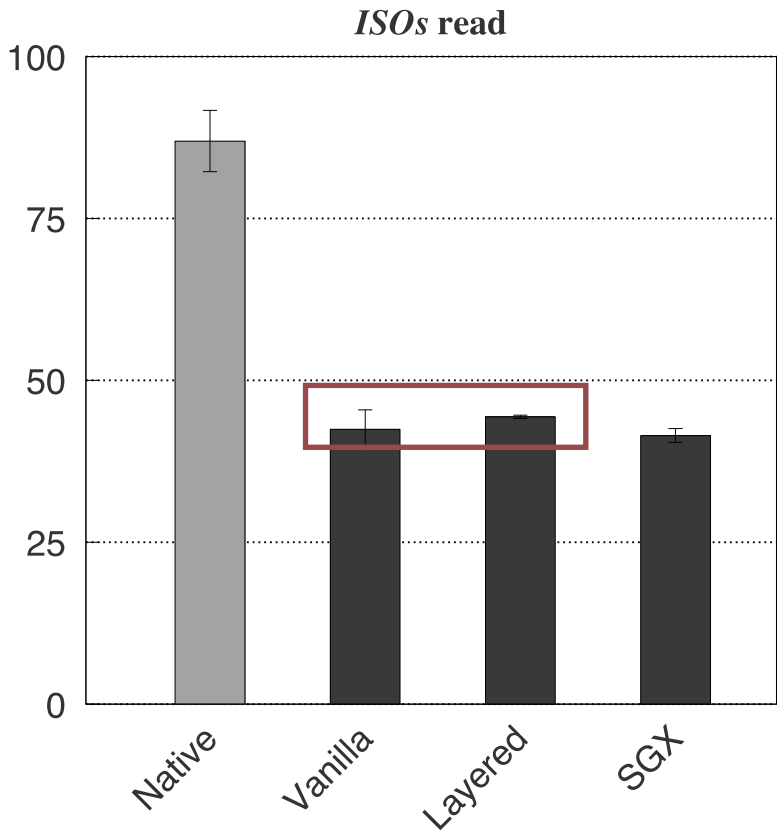


Throughput degradation from Native setup to Vanilla setup

PRELIMINARY EVALUATION

ISO DATA DUMPS

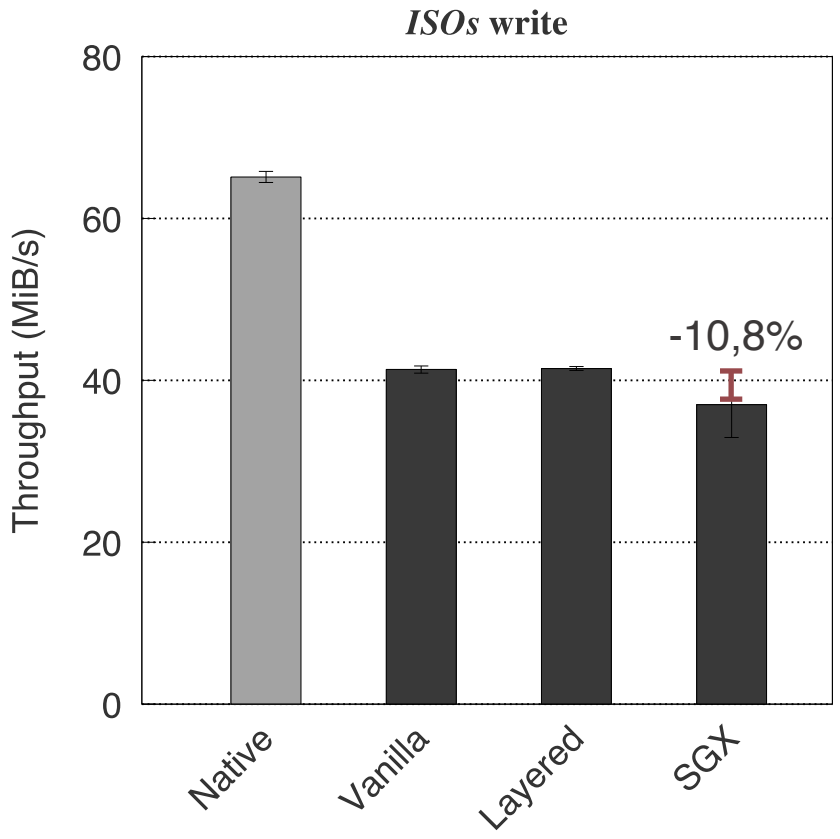


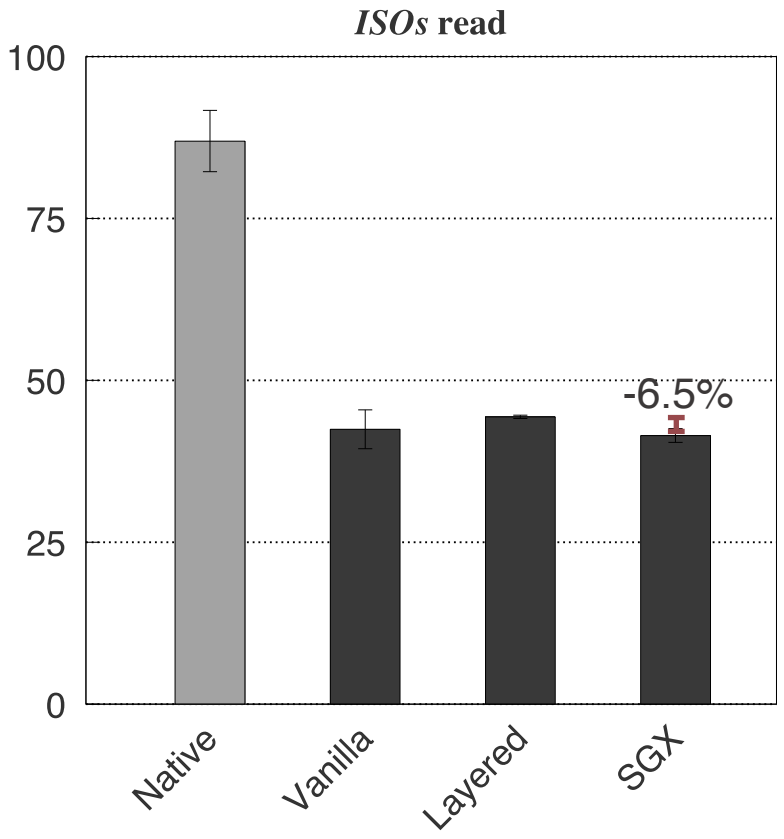


- Throughput degradation from Native setup to Vanilla setup
- Similar performance for Vanilla and Layered setups

PRELIMINARY EVALUATION

ISO DATA DUMPS





- Throughput degradation from Native setup to Vanilla setup
- Similar performance for Vanilla and Layered setups
- Throughput degradation of 10.8% (writes) and 6.5% (reads) from Layered setup to SGX setup

CONCLUSION

- **TrustFS**, an SGX-enabled stackable file system framework for building secure content-aware storage systems
  - Modular and programmable architecture with support for Intel SGX
- Preliminary evaluation of a compression prototype shows a reasonable performance overhead under most workloads
  - Throughput degradation from 6.5% up to 31.3%

OPEN ISSUES AND FUTURE DIRECTIONS

- Storage layout changes across layers
- Chunk splitting across layers
- Integration of existing storage solutions
- Key exchange and management

TrustFS: An SGX-enabled Stackable File System Framework

1st Workshop on Distributed and Reliable Storage Systems (DRSS'19)
Lyon, 1st October 2019

Tânia Esteves[1], Ricardo Macedo[1], Alberto Faria[1], Bernardo Portela[2],
João Paulo[1], José Pereira[1] and Danny Harnik[3]

[1] INESC TEC and University of Minho, Portugal. [2] INESC TEC and University of Porto, Portugal.
[3] IBM Research – Haifa, Israel.

INESCTEC

Universidade do Minho