DSN 2021
The 51st Annual IEEE/IFIP International Conference
on Dependable Systems and Networks
Taipei, Taiwan, June 21-24, 2021

Plinius: Secure and Persistent Machine Learning Model Training

Peterson Yuhala[1] **Pascal Felber**[1] **Valerio Schiavoni**[1] **Alain Tchana**[2]

[1] University of Neuchâtel, Switzerland

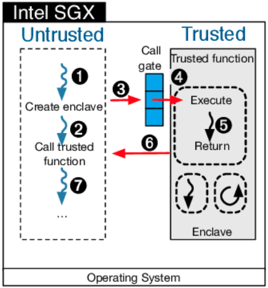
[2] ENS Lyon, France

Context and Problem

- Increasing popularity of cloud-based ML services (*e.g.*, Amazon ML, MS Azure AI).
- Security and privacy issues, *i.e.*, sensitive training data and models.
- DRAM scalability issues and high-access times of secondary storage = bottlenecks for ML.
- We need practical solutions to both problems.

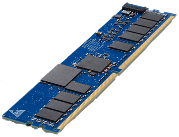
Intel SGX

- We solve security issues with TEEs (*e.g.*, Intel SGX).
- Secure *enclaves*: no system functionality, *i.e.*, system calls
- Legacy applications must be re-written/partitioned.



Persistent Memory

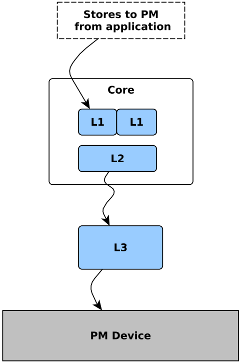
Context
Intel SGX
PM
Architecture
Evaluation
Conclusion



- We use persistent memory (PM) to solve DRAM/storage related issues. PM is:
  - Byte-addressable (like DRAM), and accessed via Load/Store.
  - Fast (low-latency, faster than SSD)
  - Persistent (like SSD)
  - Higher capacity than DRAM

How to use PM

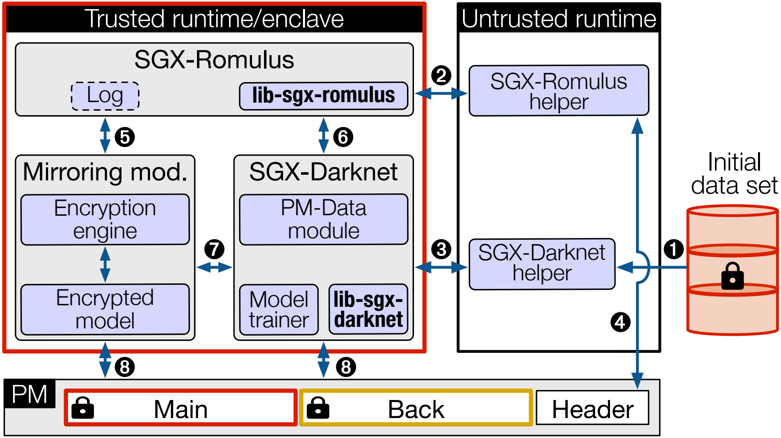
- Like secondary storage: no program changes but smaller performance improvements.
- Leverage byte-addressability: requires program changes but better performance.



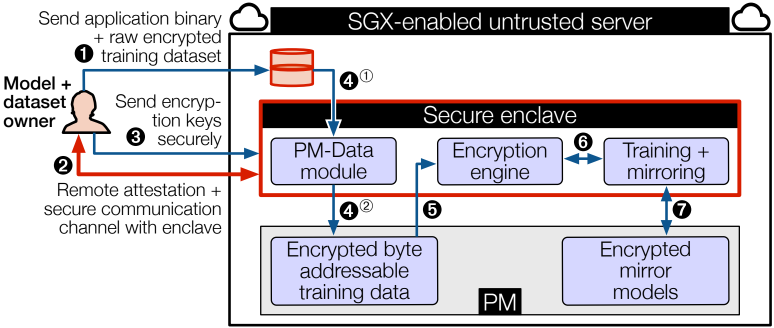
Plinius in a nutshell

- Plinius ports a PM and ML library into SGX.
- It leverages the byte-addressability of PM for fast access to data in PM.
- Models trained in the enclave are mirrored to/from PM.

Plinius Architecture



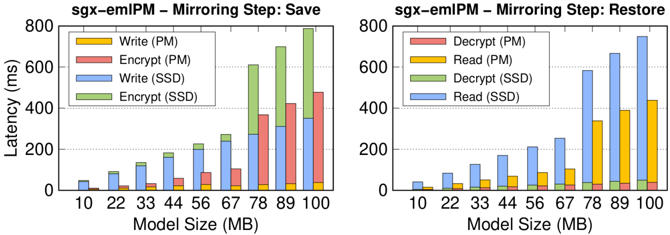
Plinius Workflow


Send application binary + raw encrypted ❶ training dataset
Model + dataset owner
Send encryption keys ❸ securely
❷ Remote attestation + secure communication channel with enclave
SGX-enabled untrusted server
❹①
Secure enclave
PM-Data module
Encryption engine
❻
Training + mirroring
❹②
❺
❼
Encrypted byte addressable training data
PM
Encrypted mirror models

Main Evaluation Goals

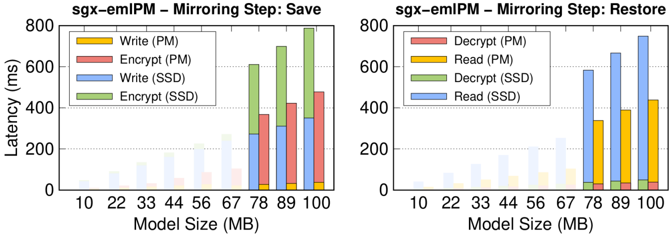
- How does Plinius improve save/restore performance ?
- How scalable is Plinius with varying model sizes ?

Evaluation



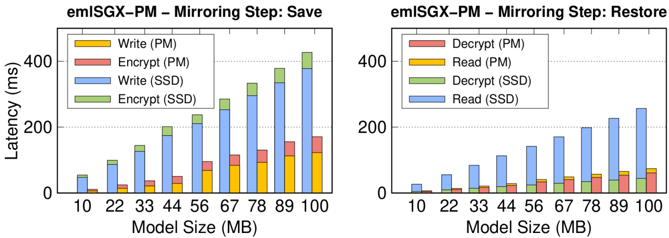
- Emulated PM + real SGX server: saves 3.5× and restores 2.5× faster vs SSD.

Evaluation



- Performance drops at the EPC limit due to page swapping operations.

Evaluation



- Real PM + sim SGX server: saves 3.2× and restores 3.7× faster vs SSD.

Conclusion

- Plinius is the first framework to leverage SGX for security and PM for fault tolerance.
- We leverage a mirroring mechanism for fault tolerance.
- Model and training data in memory → near instantaneous recovery after crashes.
- Test Plinius on github: