TRUSTED EXECUTION

Prof. Dr. Christof Fetzer
Systems Engineering Chair
TU Dresden
MOTIVATION

➤ We help stakeholders to protect

➤ data (e.g. training data), and/or

➤ code (e.g., Python code)
MOTIVATION

➤ We help stakeholders to protect
  ➤ data (e.g. training data), and/or
  ➤ code (e.g., Python code)

➤ Systems engineering:
  ➤ we build stuff to see if that really works
  ➤ publish in top systems conferences (OSDI, EuroSys, …)
MOTIVATION

➤ We help stakeholders to protect
  ➤ data (e.g. training data), and/or
  ➤ code (e.g., Python code)

➤ Systems engineering:
  ➤ we build stuff to see if that works in real life
  ➤ publish in top systems conferences (OSDI, EuroSys, ...)

➤ Try it out in practice:
  ➤ spin off: Cloud&Heat, SIListra Systems, SCONTAIN
PROTECT GOALS

➤ Protecting

➤ Confidentiality - keeping data / code secret

➤ Integrity - prevent unauthorized data & code modifications

➤ Freshness - prevent rollback to old versions of data and code
USE CASE: MODEL GENERATION

data provider

training data

application provider

machine learning

model owner/provider

model

python code
CONTAINER-BASED APPS

**Protection Objectives**

- Need to protect the integrity and freshness of the code (CIF) and C limits access to the code.
- Need to protect the confidentiality, integrity and freshness of application code (CIF).
- Protect data (CIF), i.e., protect training data as well as generated model.
USE CASE: NEED TO SUPPORT MULTIPLE DEVELOPERS

- **image provider**
  - Python Runtime Image
  - stored in public repository

- **developer 3**
  - docker build

- **developer 2**
  - application (image)

- **developer 1**
  - application (image)
USE CASE: NEED TO SUPPORT MULTIPLE INSTANCES

- **Image provider**
- **Python Runtime Image**
- **docker build**
- **docker run**
- **application (image)**
- **application A (instance)**
- **developer 1**
- **developer 2**
- **developer 3**
- **client 1**
- **client N**
- **training data**
- **model**
- **lives in public repository**

Flow:
- Image provider
- Python Runtime Image
- Docker build
- Docker run
- Application (image)
- Application A (instance)
- Client 1
- Client N
- Training data
- Model
THREAT MODEL: BYZANNTINE STAKEHOLDERS

➤ We do not trust any individual, i.e., no trusted person
We do not trust any individual, i.e., no trusted person.

We believe, however, one can define $N$ and $F \ (< N)$ and a group of persons $PB$ such that:

- $|PB| = N$
- at least $N-F$ in $PB$ can be trusted.

We typically do not know who to trust!
THREAT MODEL

client → Application ← attacker

https://sconedocs.github.io
THREAT MODEL

- **client**
- **Application**
- **attacker**
- **system administrator** (root, hardware access)
- **service provider administrator** (root, application rights)

[https://sconedocs.github.io](https://sconedocs.github.io)
IMPLICATION: OS-BASED ACCESS CONTROL INSUFFICIENT

Application

client

secret

dump memory

system administrator
(root, hardware access)

service provider administrator
(root, application rights)

attacker

https://sconedocs.github.io
WE NEED A CRYPTOGRAPHIC APPROACH!

application service provider administrator
(root, application rights)

client

TLS

Application

attacker

system administrator
(root, hardware access)

service provider administrator
(root, application rights)

https://sconedocs.github.io
HOW TO PROTECT THE KEYS?

Application

key

attacker

client

TLS

system administrator
(root, hardware access)

service provider administrator
(root, application rights)

crypto

https://sconedocs.github.io
RESEARCH PROBLEMS ADDRESSED

➤ How can we provide applications with secrets running in an untrusted environment?

➤ How can we delegate the management of these secrets to untrusted entities?

➤ How to manage the secrets despite malicious stakeholders?

➤ How to support secure application updates?

➤ How can we ensure that no rollbacks happen?

➤ How to protect against malicious developers, cloud providers and system admins?

➤ ...

➤ How can we do all this without changing application source code?
MORE USE CASES

➤ Electronic Patient Records
➤ Decentralized Apps (DApps)
➤ Blockchain related use cases
➤ Secure Data-as-a-Service
➤ Health Domain / DNA
➤ …

➤ Approach:
  ➤ do not start from scratch for each application!
SCONE PLATFORM  (HTTPS://SCONEDOCS.GITHUB.IO)

trusted client computer

input data

encrypted data

encrypted processing

remote untrusted computers

output data

SCONE:
- supports always encrypted code and data
always encrypted

encrypted processing

application (no source code changes)

remote untrusted computers

SCONE:
- attests that the correct application is running!
- manages keys & secrets for applications
- de/encrypts data and files - transparent to application

Palaemon
SCONE secrets management service
END-TO-END ENCRYPTION

S Cone: supports end-to-end encryption and trusted, remote and local computation.
ADVANTAGES OF USING SCONE

➤ Attests that the correct code is running

➤ Protects confidentiality, integrity and freshness of data and code even against attackers with root privileges

➤ Provides an integrated secret management

➤ Can be used for a more secure licensing management

➤ Even if attacker would have root access...
SCONE USE CASES

➤ Medical domain:
  ➤ electronic patient records

➤ AI / Machine Learning:
  ➤ supports TensorFlow

➤ Blockchain domain:
  ➤ decentralized applications
  ➤ Data-as-a-service
  ➤ Supports Parity Substrate inside of enclaves

➤ General:
  ➤ Vault, Barbican, PySpark, Blender, …
**EXTENDED THREAT MODEL**

- Attacker has *root access* on all machines
- Attacker has *hardware access* on all machines
- Attacker *controls* (credentials of) some but not all stakeholders
- Attacker knows sufficient *vulnerabilities in software*
  - note: *about one bug every 2000 lines of source code*
- Supply chain attacks on some chips and motherboard

https://sconedocs.github.io
ALL SECRETS ARE PROTECTED BY POLICIES

security policy

Palaemon

encrypted processing

encrypted data

encrypted

https://sconedocs.github.io
NO TRUST IN ANY INDIVIDUAL OPERATORS / USERS / …

SCONE:
- policies are protected by **policy boards**
- members can be humans and (attested) scripts
- changes requires approval from all/majority/.. members

https://sconedocs.github.io
Application Provider (provides container images) publishes Data Owner (wants to monetize data).

Data scientist (User) uses always encrypted remote untrusted computers.

Application Provider (provides container images) provides encrypted processing application. Palaemon operates cloud.

Infrastructure Provider (operates computers & services) operates cloud. Auditor (checks source code) checks application.

SCONE has Multi-Stakeholder support!
CURRENT IMPLEMENTATION

➤ Intel SGX protects application’s confidentiality and integrity by preventing accesses to application state in cache and encrypting main memory

➤ SGX is a TEE (Trusted Execution Environment)

SGX (Software Guard eXtensions) protects application from accesses by other software

https://sconedocs.github.io
DEFENDER’S DILEMMA

➤ Attackers:
  ➤ success by exploiting a single vulnerability

➤ Defender:
  ➤ must protect against every vulnerability
  ➤ system software & application
  ➤ millions of lines of source code

https://sconedocs.github.io
**TPC-C:** increasing buffer pool has little impact on performance
Overheads

Lower the better

< 22% overhead compared to native execution
PyPy SCONE: just in time Python inside enclave

Python Native: CPython in native mode
SCONE PLATFORM ADVANTAGES

➤ SCONE supports protection of multiple stakeholders.
➤ SCONE has an integrated secrets&configuration management
➤ SCONE scales better (high performance syscall interface)
➤ SCONE generates smaller executables.
➤ SCONE comes with a toolchain.
➤ SCONE protects the OS interface.
➤ SCONE ensures better Linux compatibility.
➤ SCONE transparently attests applications.
➤ SCONE’s design is hardware independent.
BA, MSc, Diplom Thesis

» Not much on website
» Customized to students
  » talk to me to find an interesting top
BA, MSC, DIPLOM THESIS

➤ Not much on website
➤ Customized to students
  ➤ talk to me to find an interesting top

➤ Current topics:
  ➤ Function as a service in DB (with Oracle)
  ➤ Secure GraalVM (with Oracle)
  ➤ Blockchain topics (with vmware)
  ➤ Encrypted binary code
  ➤ …
JOBS

- Always looking for students
  - SHK, WHK
  - PhD students
  - PostDocs

- Talk to me regarding external jobs