Runtime Secure Keys in OP-TEE

Sahil Malhotra
Arun Pathak

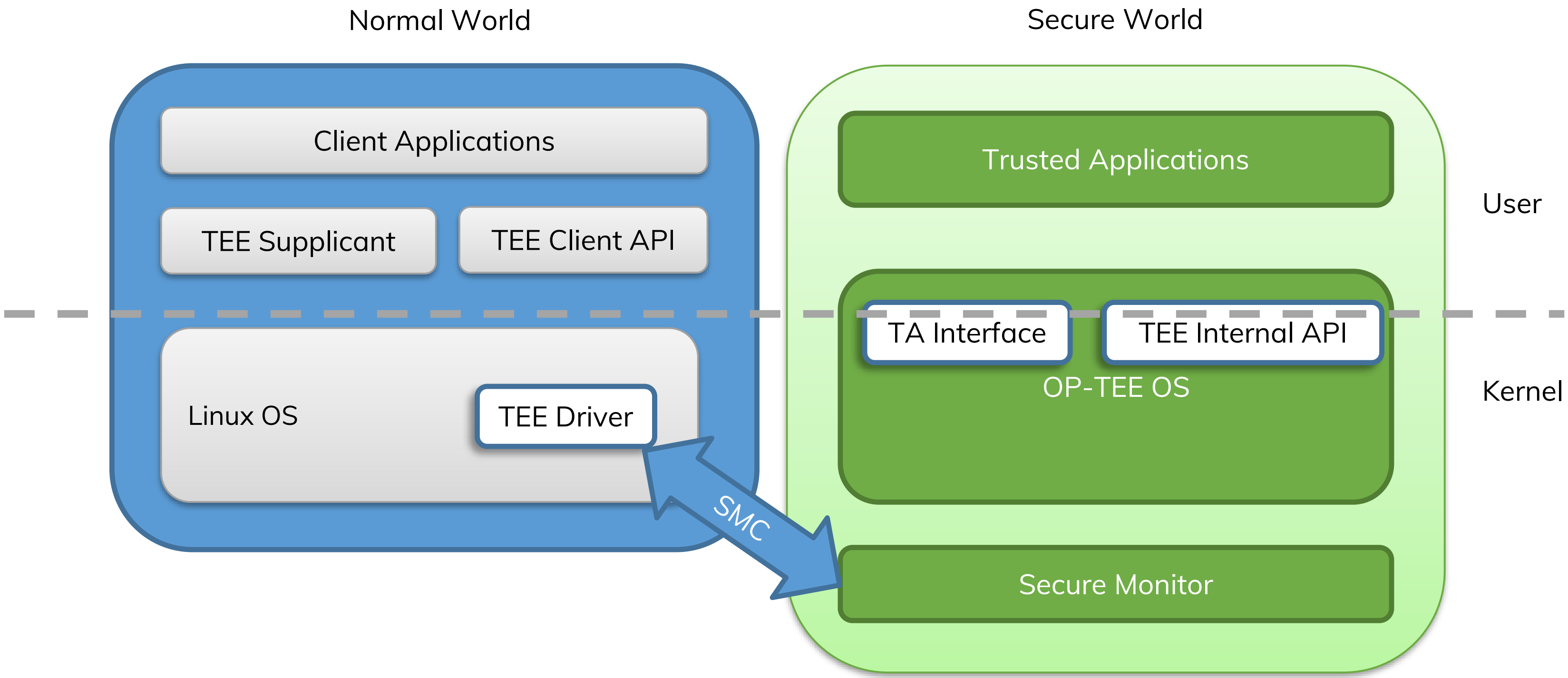
NXP Platform Security Team

Linaro connect
San Diego 2019

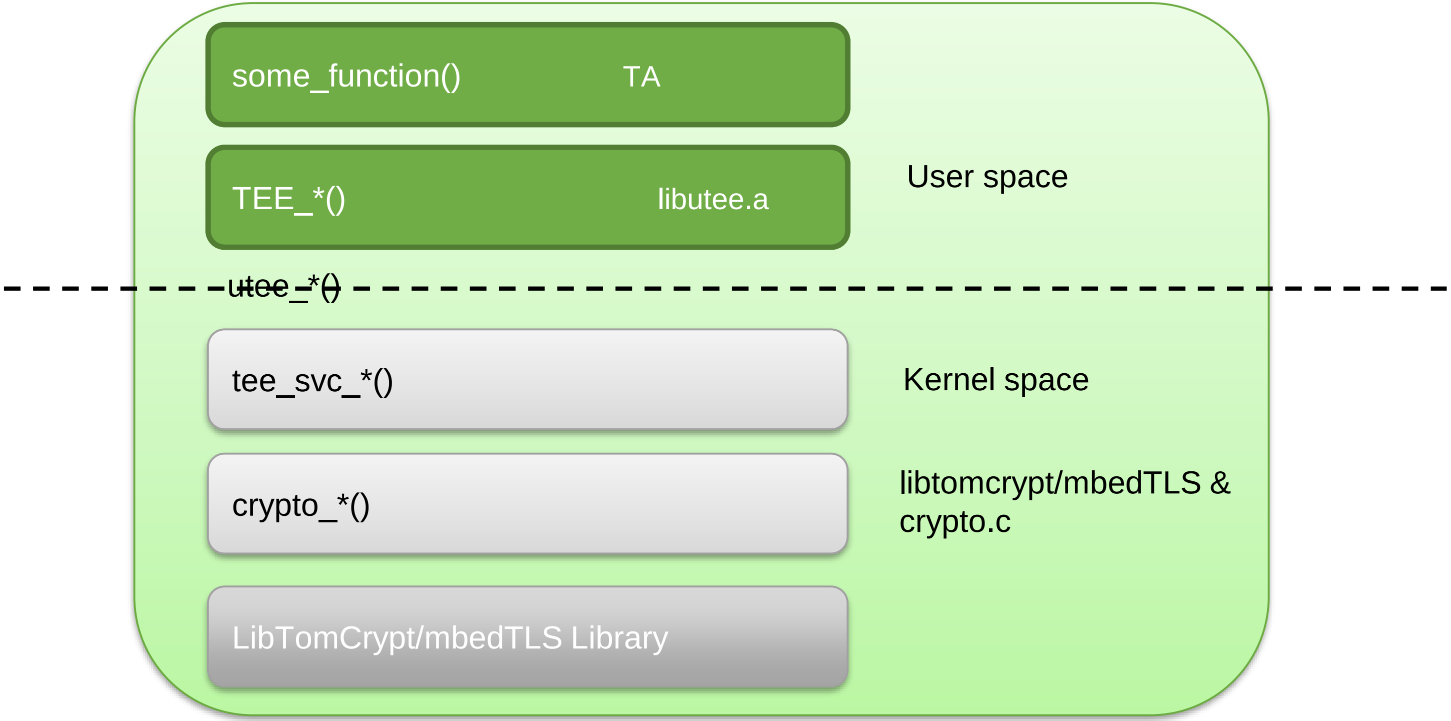
Agenda

- OP-TEE Overview
- OP-TEE Crypto Layers Overview
- Crypto Operation
    - Key Generation & storage
    - Key Usage
- Security view in current implementation
- Well known security vulnerabilities
- Prevention – Hardware Backed Runtime Secure Keys
- NXP Proposal

OP-TEE Overview

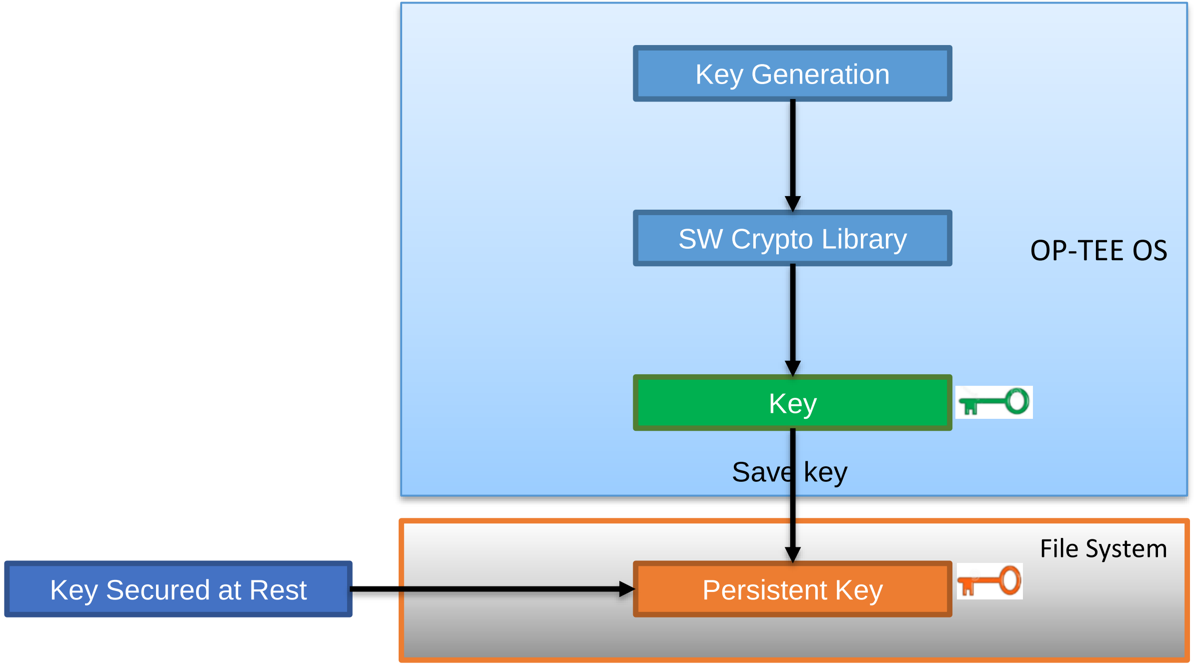


OP-TEE Crypto Layers

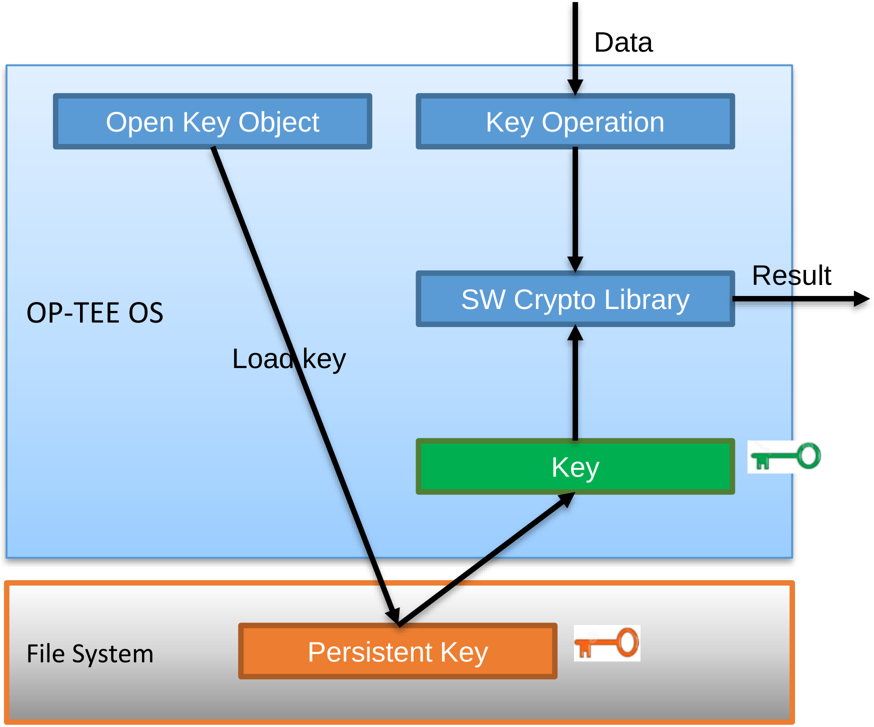


Source: https://optee.readthedocs.io/architecture/crypto.html

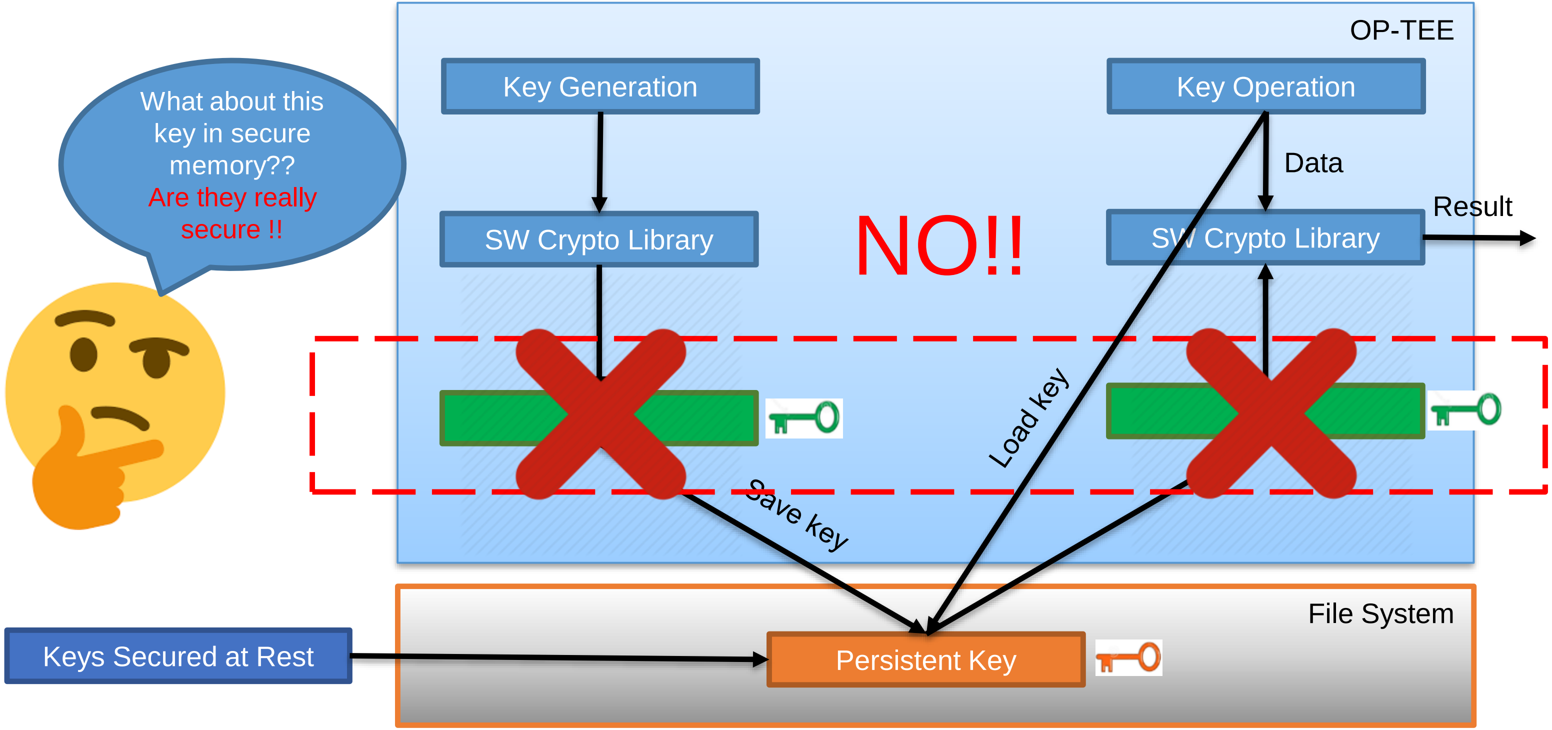
Crypto Operation: Key Generation & Storage



Crypto Operation: Key Usage

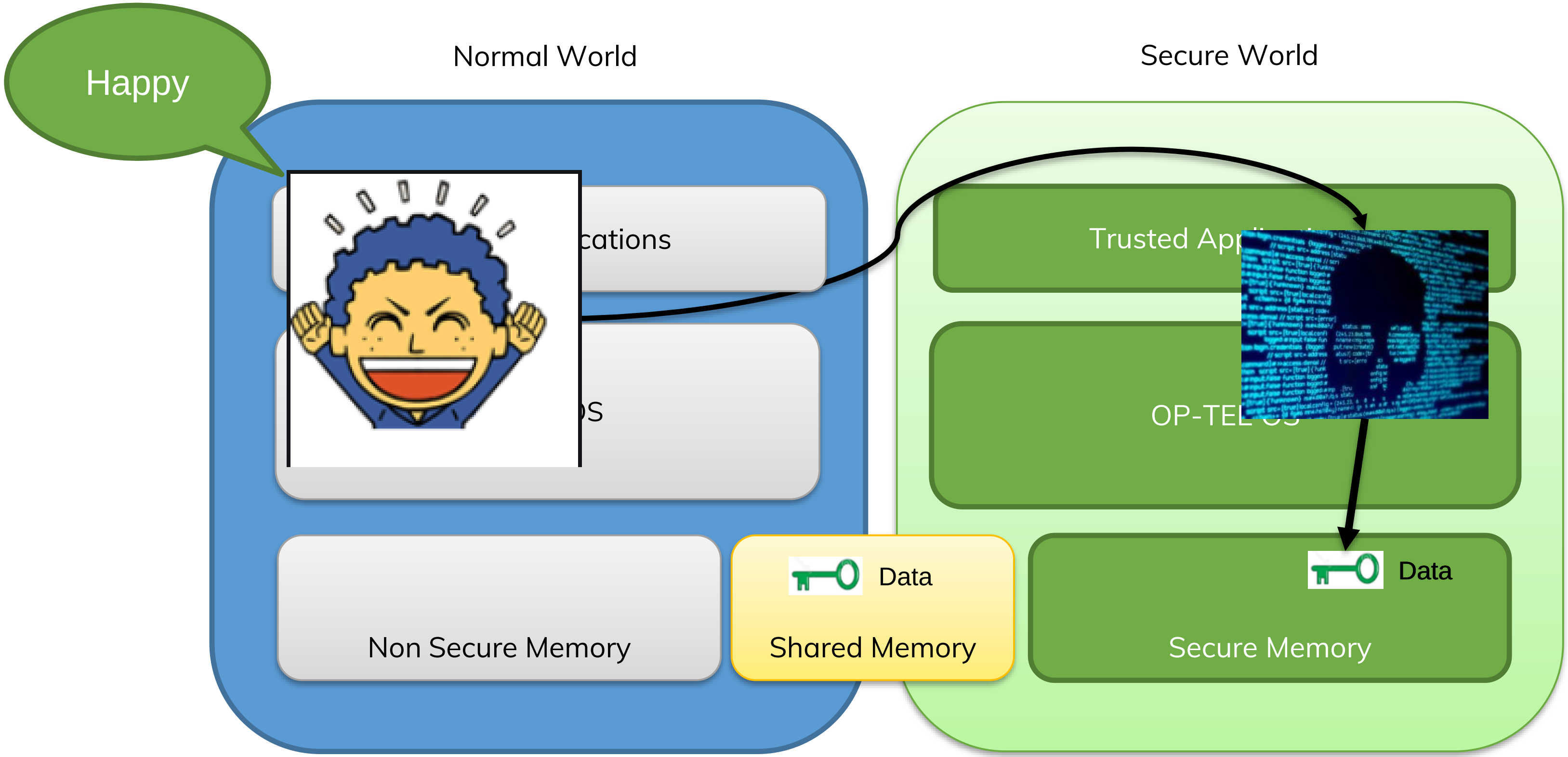


Security view in current implementation



Well Known Security Vulnerabilities

- OP-TEE: Integer Overflow in crypto system call - syscall_obj_generate_key
  - It takes the length of key to be generated, type, number of attributes(param_count) it should have.
  - Allocates a buffer of size sizeof(TEE_Attribute) * param_count, without checking for the integer overflow.
  - This can result in lesser heap buffer than required.
  - Then user supplied params is then copied into this buffer, that may result in heap based buffer overflow with attacker data written outside buffer boundaries.
  - Such corruption might allow code execution in context of Secure EL1
- CVE-2018-14491 - Vulnerability in Third-Party Application
  - Qualcomm based device
  - Allows arbitrary execution of code in Secure EL0



Happy

Normal World

Secure World

cations

S

Trusted App

OP-TEE OS

Data

Non Secure Memory

Shared Memory

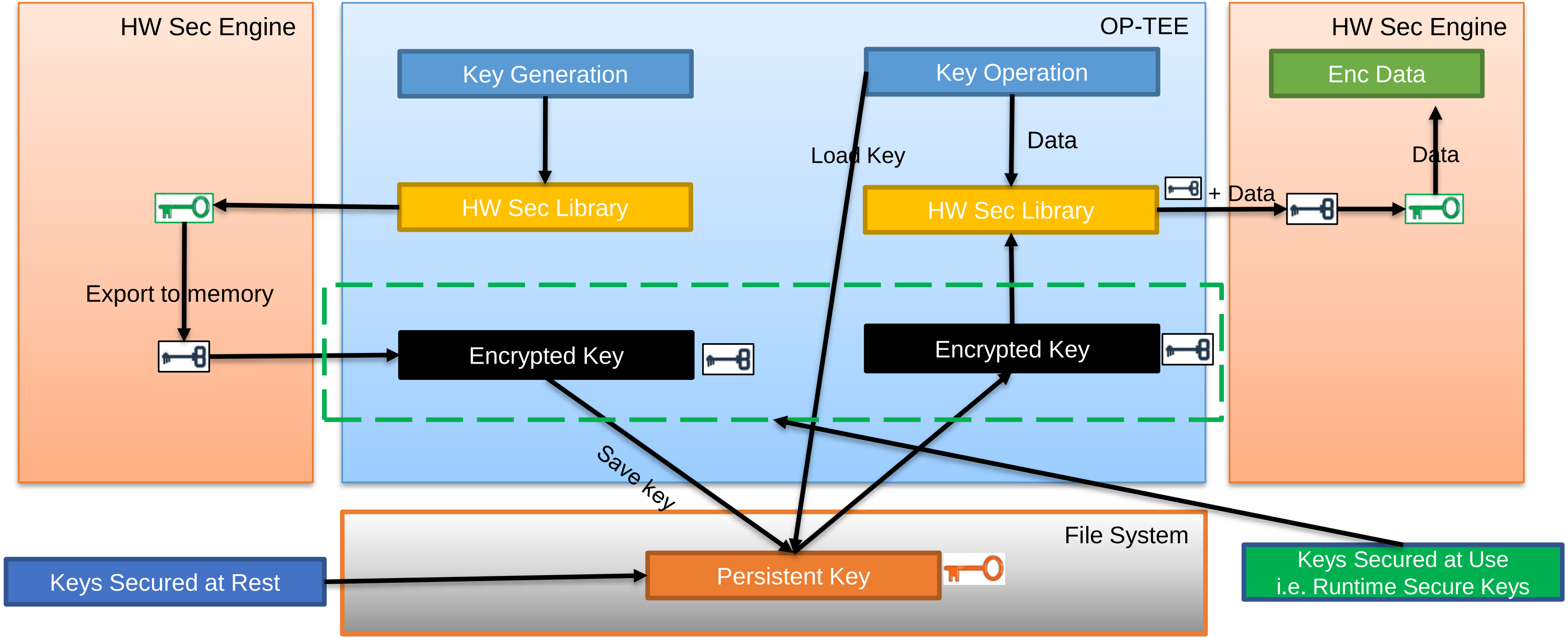
Data

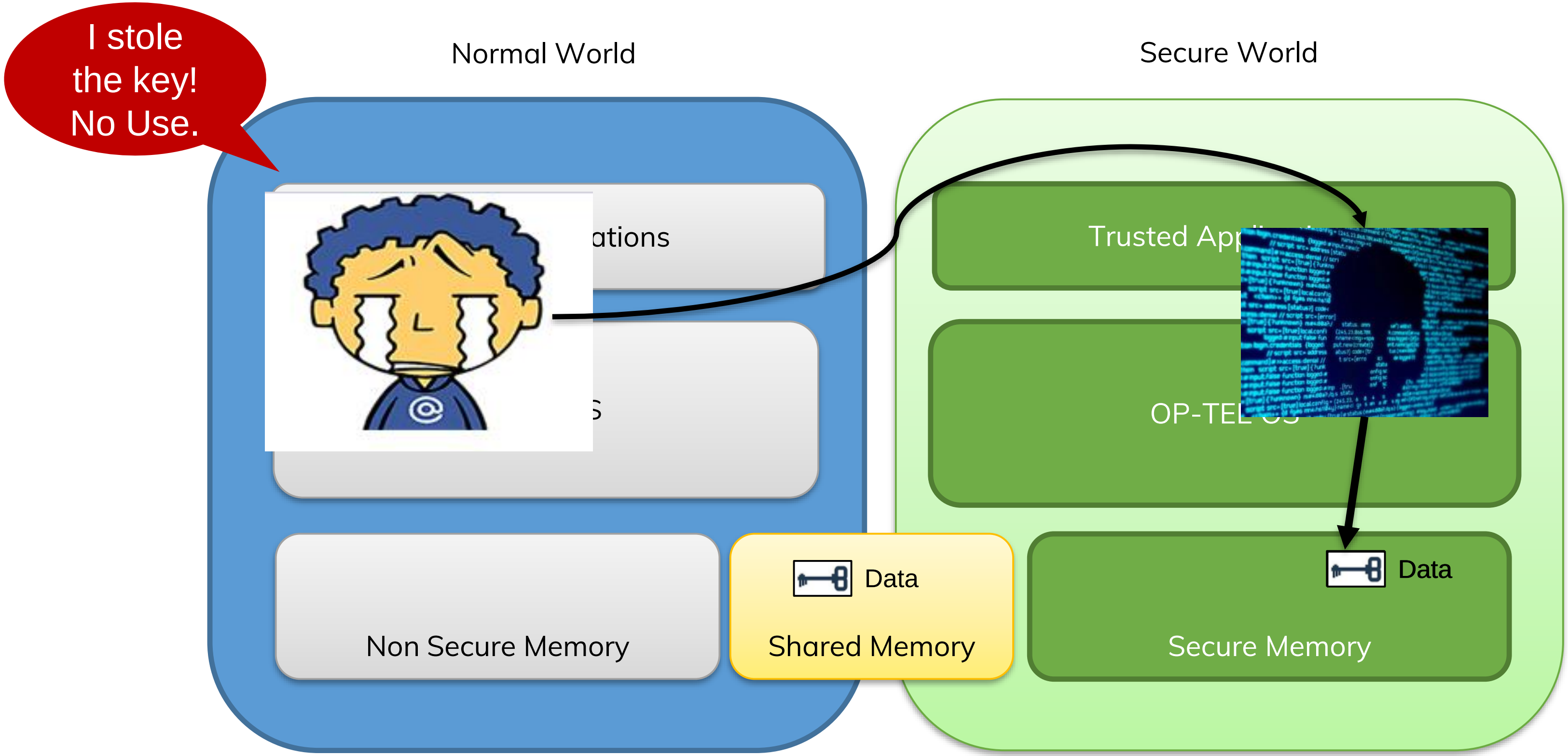
Secure Memory

Prevention – Hardware Backed Runtime Secure Keys

- Hardware Backed Runtime Secure Keys
  - Cryptographic operations are offloaded to Hardware Security Engine.
  - Hardware Security Engine gives and takes keys only in encrypted form.
  - Encryption of these keys are done with some hardware key.

Prevention – Hardware Backed Runtime Secure Keys

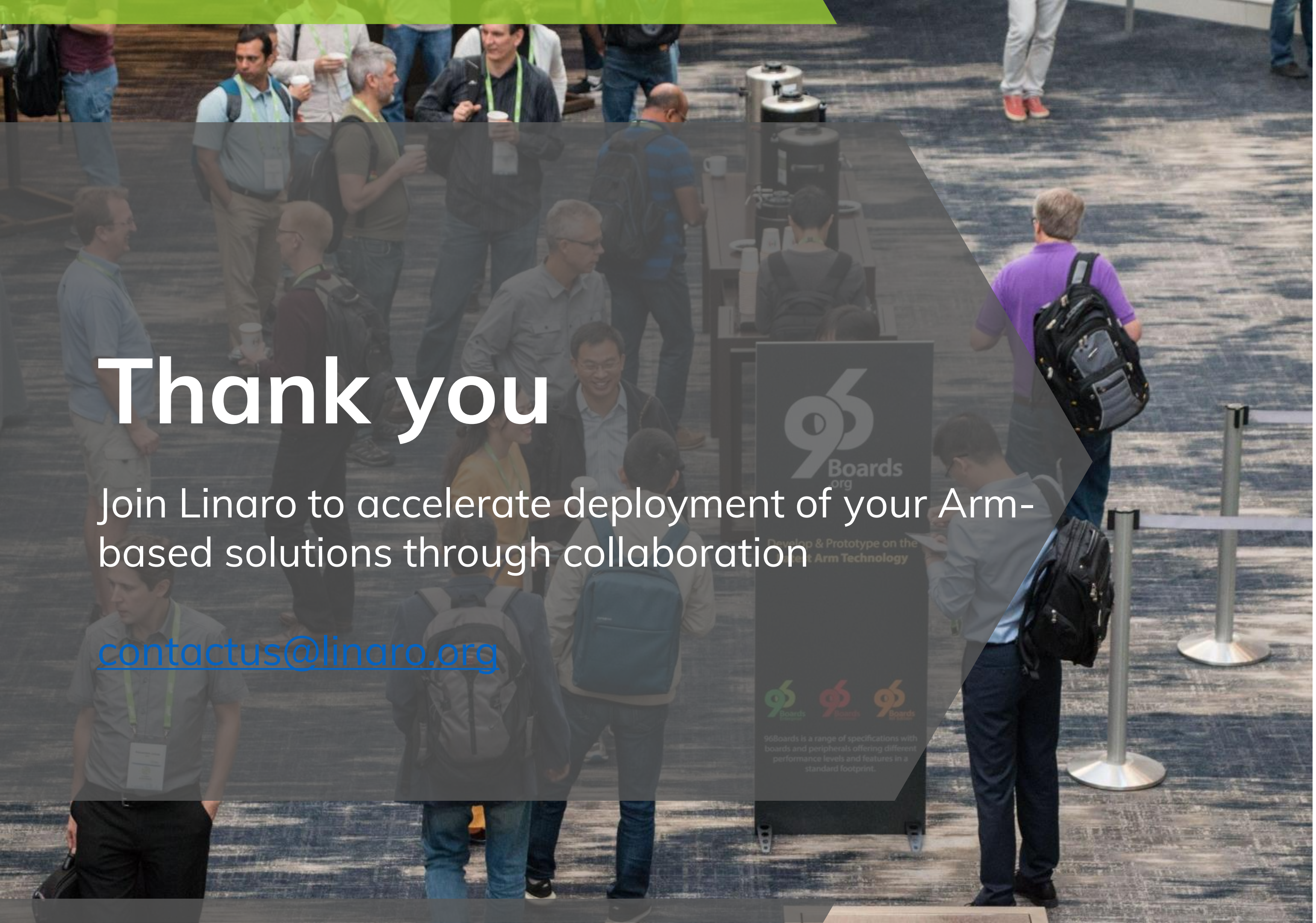




I stole the key! No Use.
Normal World
Secure World
ations
S
Trusted App
OP-TE
Data
Data
Non Secure Memory
Shared Memory
Secure Memory

NXP Proposal

- **Encrypted Key that we just discussed is NXP CAAM Black key mechanism**.
- Using the Hardware Security Engines we can protect the confidentiality and integrity of the keys while we are using them, i.e. Making them secure at runtime also.
- So we are proposing a generic framework in OP-TEE for seamless implementation of Hardware Backed Runtime Secure Keys, so that other vendors can also implement this feature on their SoCs.
- Already did PoC for implementing the Hardware Backed Runtime Secure Keys for RSA & ECDSA on top of NXP CAAM driver which is in process of upstreaming in OP-TEE.
- For Technical discussion raised an issue on OP-TEE github portal
  - https://github.com/OP-TEE/optee_os/issues/3287



Thank you

Join Linaro to accelerate deployment of your Arm-based solutions through collaboration

contactus@linaro.org

Linaro connect
San Diego 2019

References

- https://blog.quarkslab.com/attacking-the-arms-trustzone.html
- https://www.op-tee.org/security-advisories/
- https://s3.amazonaws.com/connect.linaro.org/bkk19/presentations/bkk19-419.pdf
- https://migrationobservatory.ox.ac.uk/resources/reports/thinking-behind-the-numbers-understanding-public-opinion-on-immigration-in-britain/blue-binary-code/