black hat USA 2017

JULY 22-27, 2017
MANDALAY BAY / LAS VEGAS

SGX Remote Attestation is not sufficient

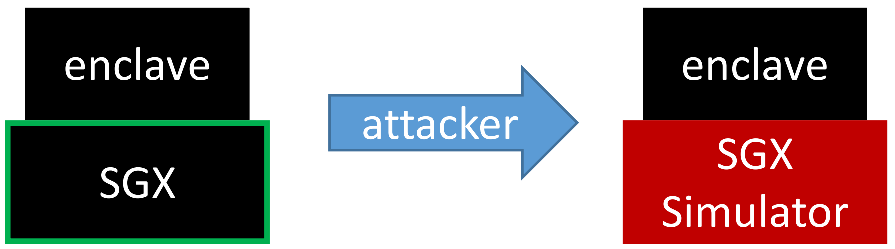
Yogesh Prem Swami

Roadmap

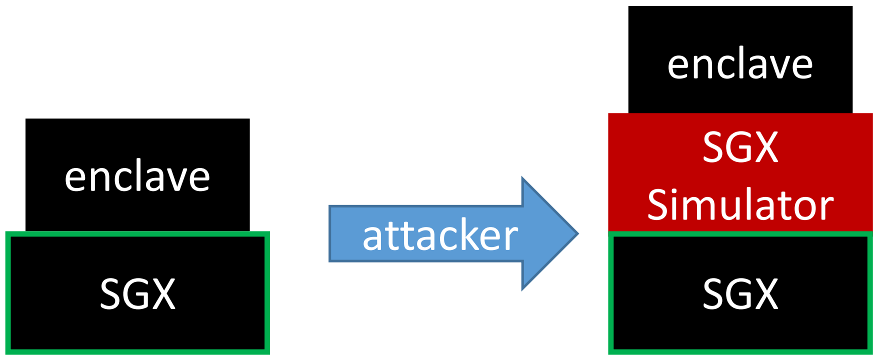
- Pitfalls in Protocol Composition
  - Sequential Composition
  - Concurrent Composition
  - Enclave State Malleability
- EPID Provisioning and Remote Attestation
  - EPID Signatures
  - SGX EPID Key Provisioning
  - SGX Quoting Enclaves

HW/SW crypto co-design

- How to ensure that remote endpoint is a real hardware
- Class-1 Simulation
  - Run simulator on general hardware
  - Countermeasure: Oracle access to a hardware resident key
- Cass-2 Simulation
  - Run simulator inside real hardware to man-in-the-middle
  - Countermeasure: More difficult
- SGX protects against both Class-1 and Class-2 Simulator



Class-1 Simulator

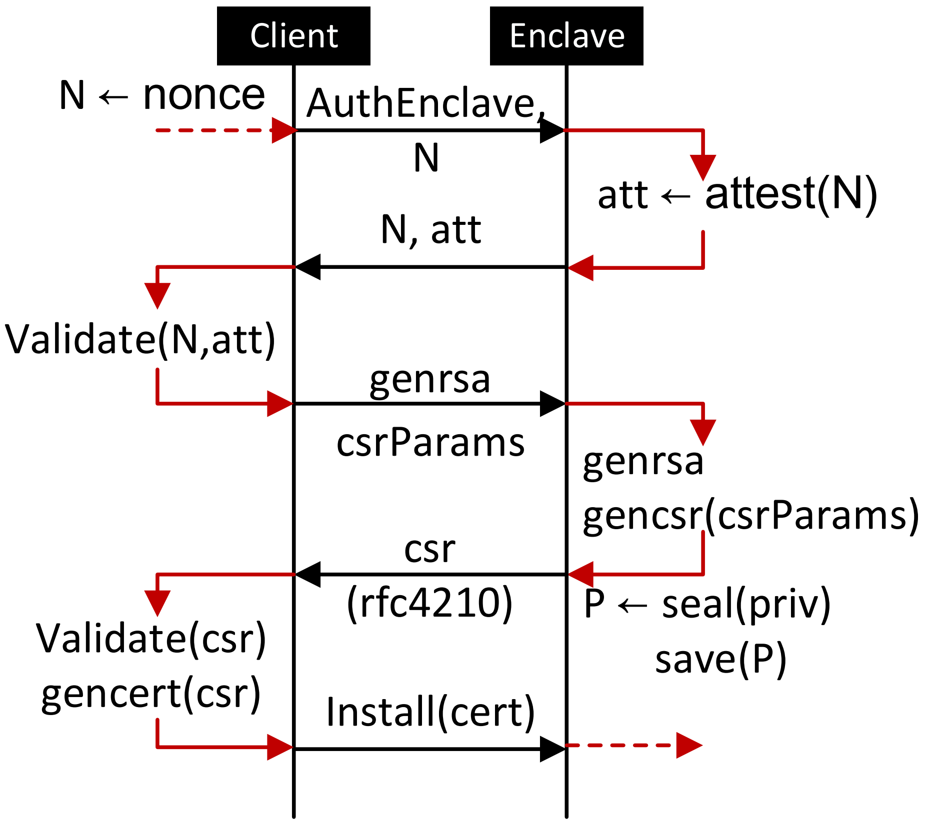


Class-2 Simulator

Common SGX Enclave Design

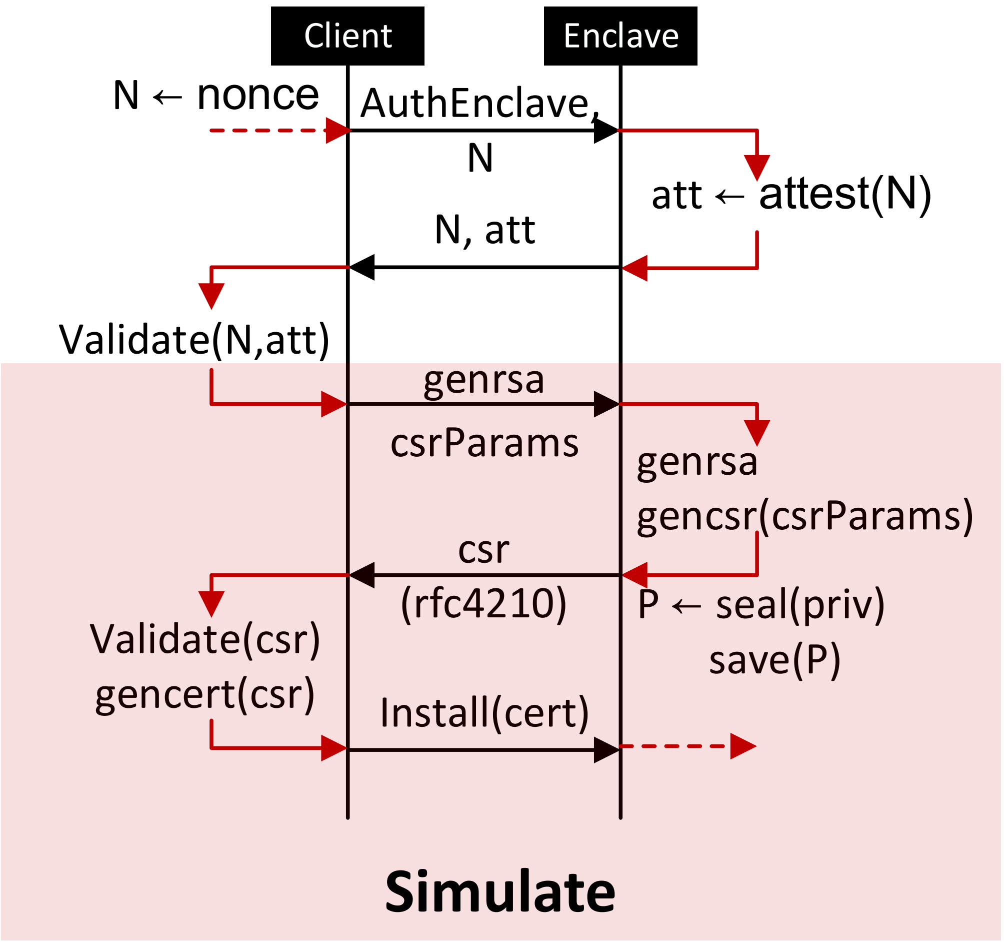
- **STEP-1**: Define a generic Remote Attestation Scheme
- **STEP-2**: Arbitrarily compose different cryptographic schemes
  - Generate keys, save them to disk
  - Generate CSR requests
  - Create Audit Log etc.
- **STEP-3**: Define a workflow that combines STEP-1 and STEP-2 to achieve the goal
- Several examples (both published as well as propriety)

Is this design paradigm secure?

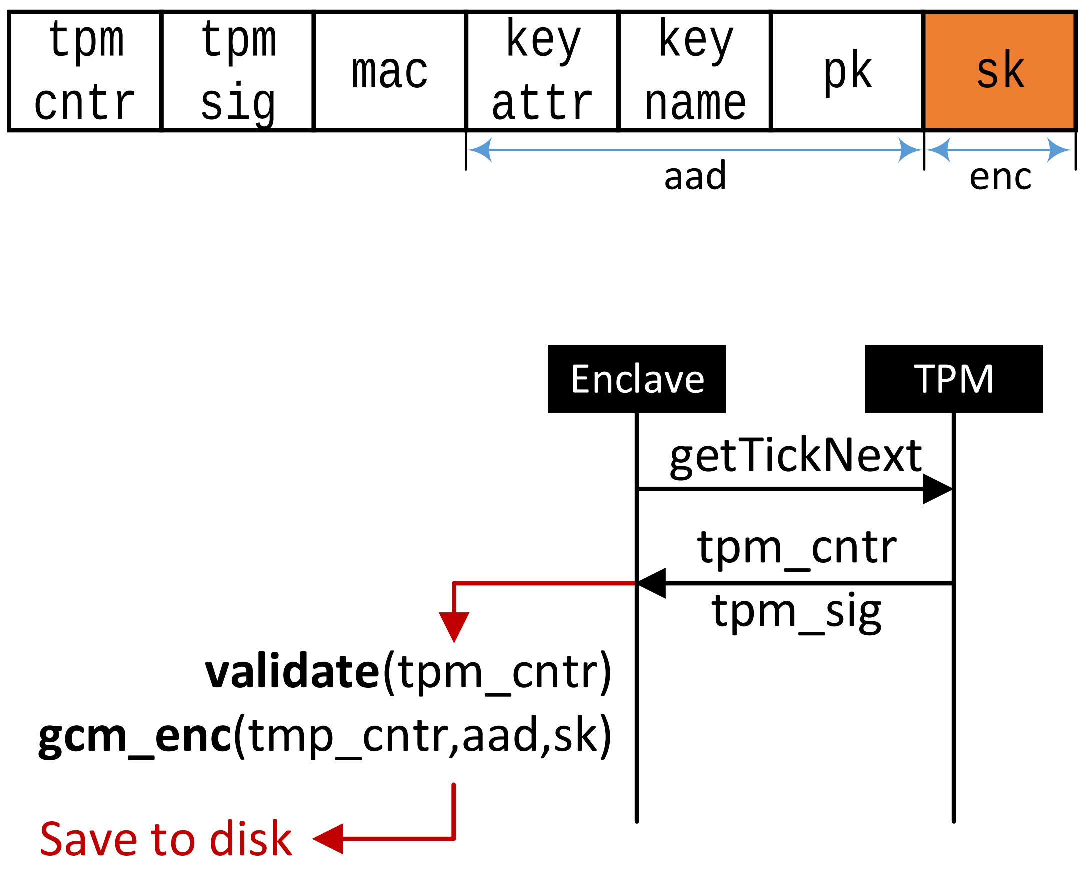


Sequential Composition

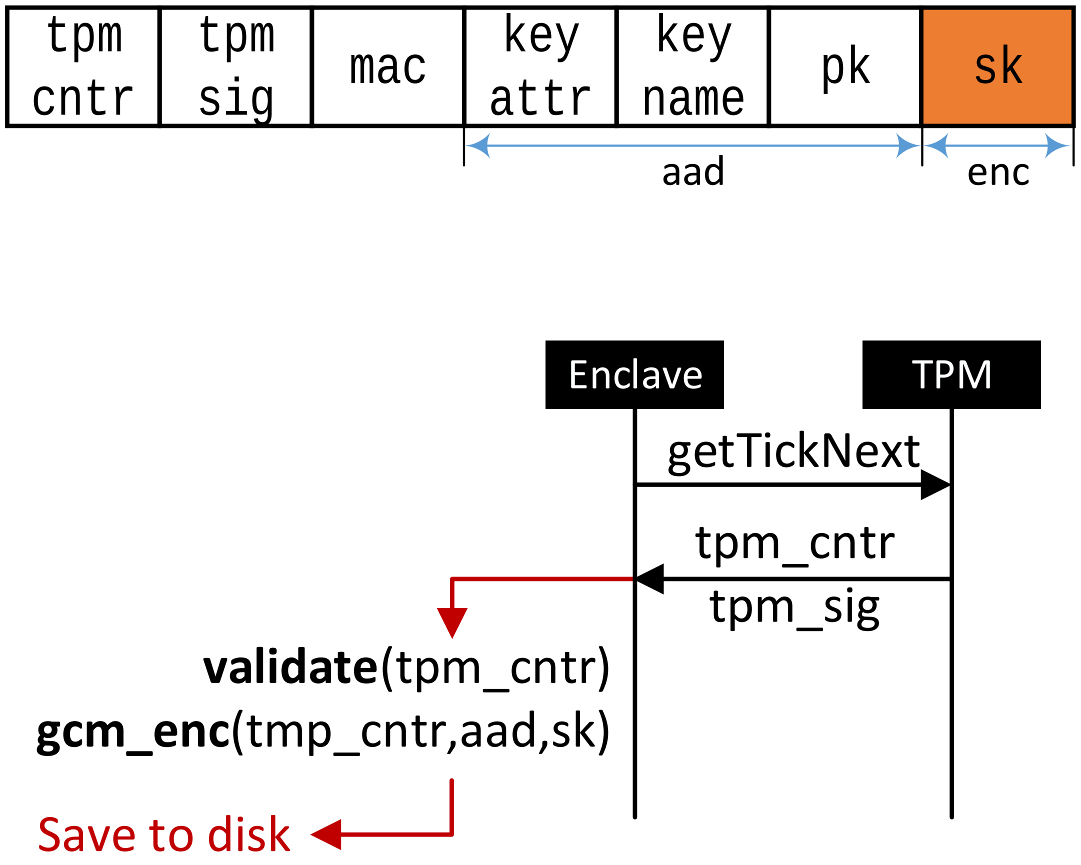
- Example vacuously broken
  - Attacker runs attestation correctly
  - Runs rest of the protocol outside the enclave
- Allows attacker to simulate some sub-computations
  - commitment log of confidential data (e.g., sha256 of someone's birthdate)
  - Send confidential data (birthdate) encrypted over TLS
  - Simulate states related to birthdate
- Enclave is a single protocol sequentially composed of sub-protocols



Concurrent Composition

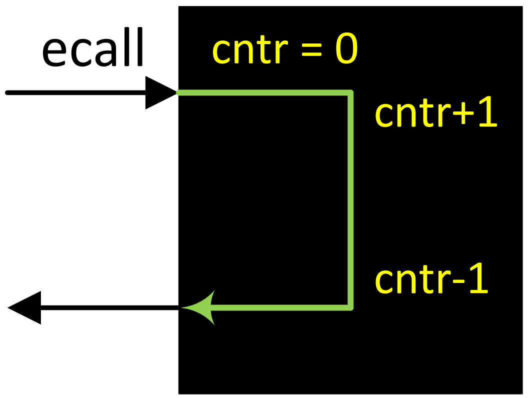


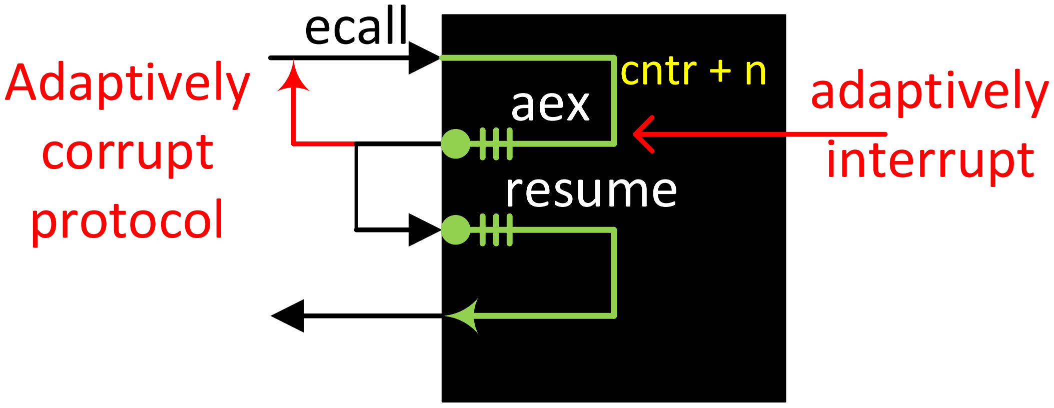
Concurrent Composition



- Attacker runs the same enclave concurrently
  - Feed the same <`tpm_cntr`,`tpm_sig`> to both instances
- Concurrent composition not limited to running same operation
- SGX Has no in-built replay protection
  - Adding TPM to TCB non-trivial
- Launch Enclave cannot limit concurrency
  - EINITTOKEN is a long-term credential
  - Whitelist ineffective

State Malleability and Knowledge Extractors





- SGX enclave <u>is not</u> a black-box
  - Adversary can force the enclave to exit at arbitrary execution point via AEX
  - Adversary controls what happens after AEX
- Global enclave state is malleable
- Partial rewinding effect possible
  - SGX allows multiple threads within the same enclave
  - Interrupt one-thread at appropriate point
  - ecall other threads
- Careful with interactive Proof-of-Knowledge (PoK) protocols
  - Σ-protocols require just 2 response per commitment to reveal the secret

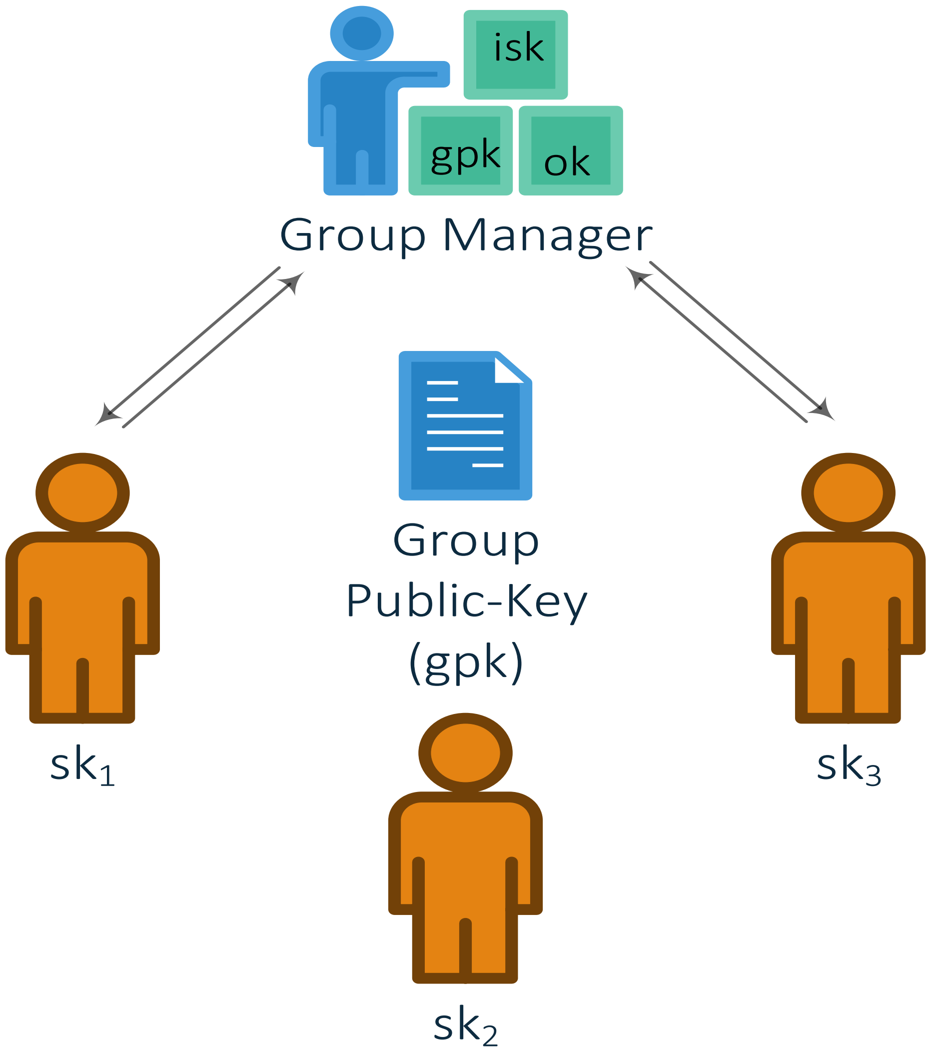
SGX EPID Provisioning and Remote Attestation

Motivation

- How to prove that Hardware knows a secret
  - Use signature-of-knowledge scheme (e.g., Schnorr signature)
  - Concerns about user privacy
- SGX Solution
  - Use group signatures

Group Signature Overview

- Members can anonymously sign messages on behalf of the group
  - Single group public-key
  - Unique private-key per-member
- Group manager decides who joins the group
  - Grants membership credentials to each member
- Several security goals
  - Fully Anonymity [BMW03]
  - Member revocation [BL09]
- EPID [BL09]:
  - Anonymity
  - Blinded join
  - Member revocation



Enhanced Privacy ID (EPID)

- Two distinct components
  - Basic Signature
  - Non-revoked signature proof
- Basic Signature based on BBS+ [BBS03, ASM06]
  - q-SDH assumption
  - CCA2 Secure in standard model
- EPID construction
  - **Join**: BBS+ signature on member's private-key (γ) in ZK
  - **Sign**: Proves knowledge of BBS+ signature in ZK
- Blinded join process
  - No concurrent join

System Param

$|G_1| = |G_2| = p;\ p \mid q^{12} - 1$

$G_1 := \langle g_1 \rangle = E(F_q)$

$G_2 := \langle g_2 \rangle = E(F_{q^2})$

$G_T := \mu_p \subset F^{\times}_{q^{12}}$

$e\ :\ G_1 \times G_2 \rightarrow G_T$

$e(g_1^x, g_2^y) = e(g_1^y, g_2^x)$

$e(g_1, g_2) \neq 1$

Setup

$\gamma \leftarrow F_p^{\times}$ :: Private Key

$h_1, h_2 \leftarrow G_1$

$w \leftarrow g_2^{\gamma}$

$\langle h_1, h_2, w \rangle$ :: Public Key

Sign (m :: F_p, γ)

$x, y \leftarrow F_p^{\times}$

$A \leftarrow (g_1 h_1^m h_2^y)^{1/(x+\gamma)}$

$\langle A, x, y \rangle$ :: Signature on m

Verify (m:: F_p, $\langle A, x, y \rangle$, $\langle h_1, h_2, w \rangle$)

$e(A, g_2^x\, w) == e(g_1 h_1^m h_2^y, g_2)$

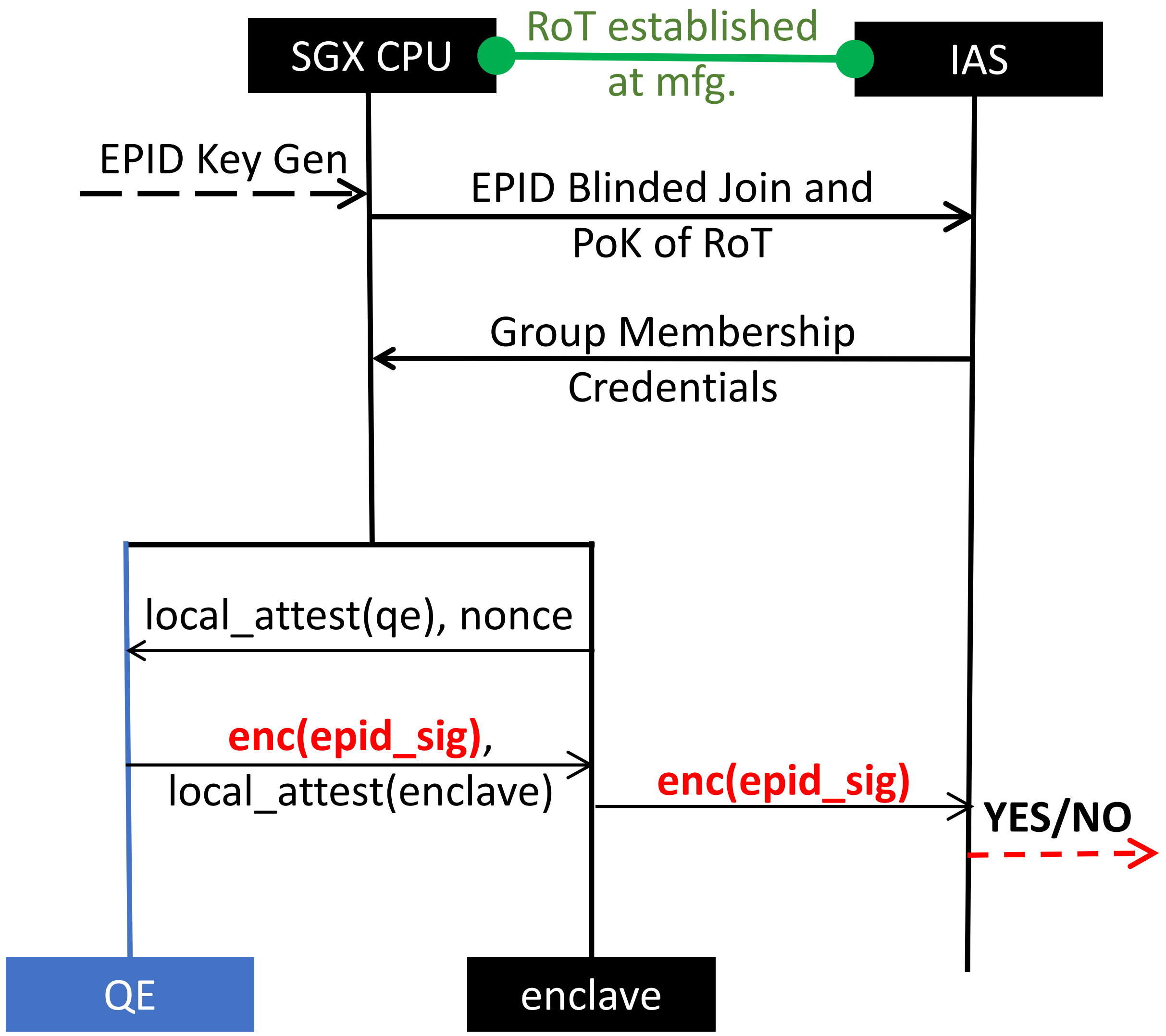
BBS+ Signature Scheme

EPID Revocation

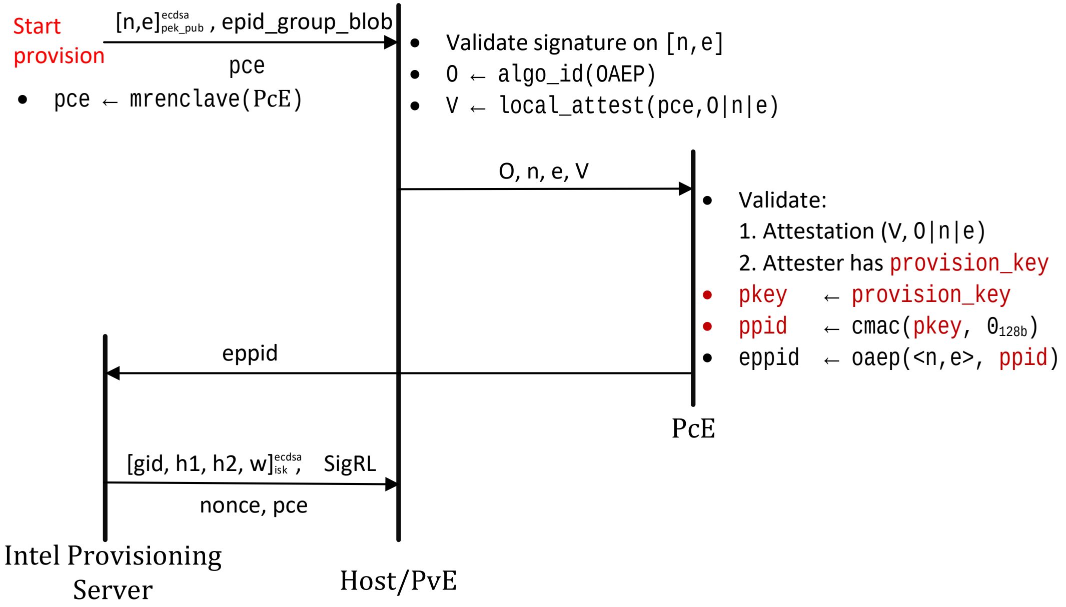
- In addition to basic signature, each signature also contains:
  - $B \leftarrow G_3$ (B can be fixed if user wants link-ability)
  - $K = B^f$ (f :: Private Key)
  - <B,K> added to signature in addition to Basic Signature
  - In SGX B = EcHash(SPID || <randomdata>)
- Signature Based Revocation (Sig-RL)
  - Sig-RL consists of $[\langle B_1,K_1\rangle, \langle B_2,K_2\rangle, \ldots, \langle B_n,K_n\rangle]$
  - Signing a message requires proving in ZK that $(K = B^f \wedge K_j \neq B_j^f)$ [CS03]
- Private-Key Based Revocation (Priv-RL)
  - Member's private-key directly placed in the revocation list
  - Retroactively destroys member's anonymity (no full anonymity [BMW04])

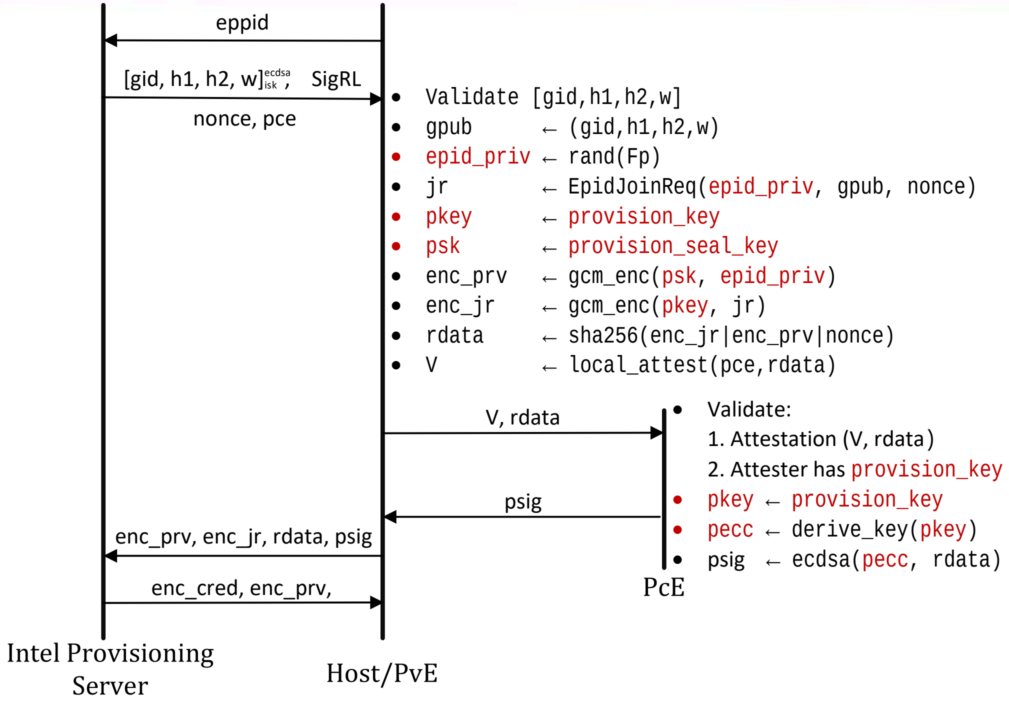
SGX Remote Attestation Big Picture

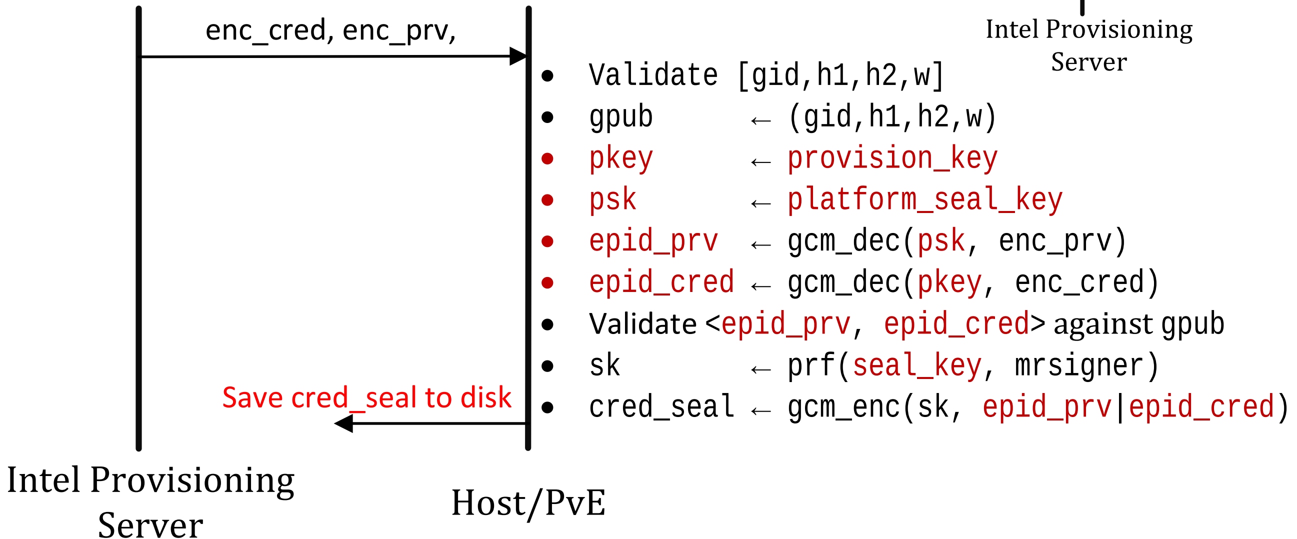
- EPID Provisioning
  - How to join an SGX EPID Group
  - Handled by PvE and PcE Intel Signed enclaves
  - Uses Root Provisioning Key as Root of Trust for joining the Group
- Remote Attestation
  - Generate EPID Signatures on enclave's identity (mrenclave, mrsigner, attr)
  - Handled by QE (Quoting Enclave)



SGX EPID Provisioning





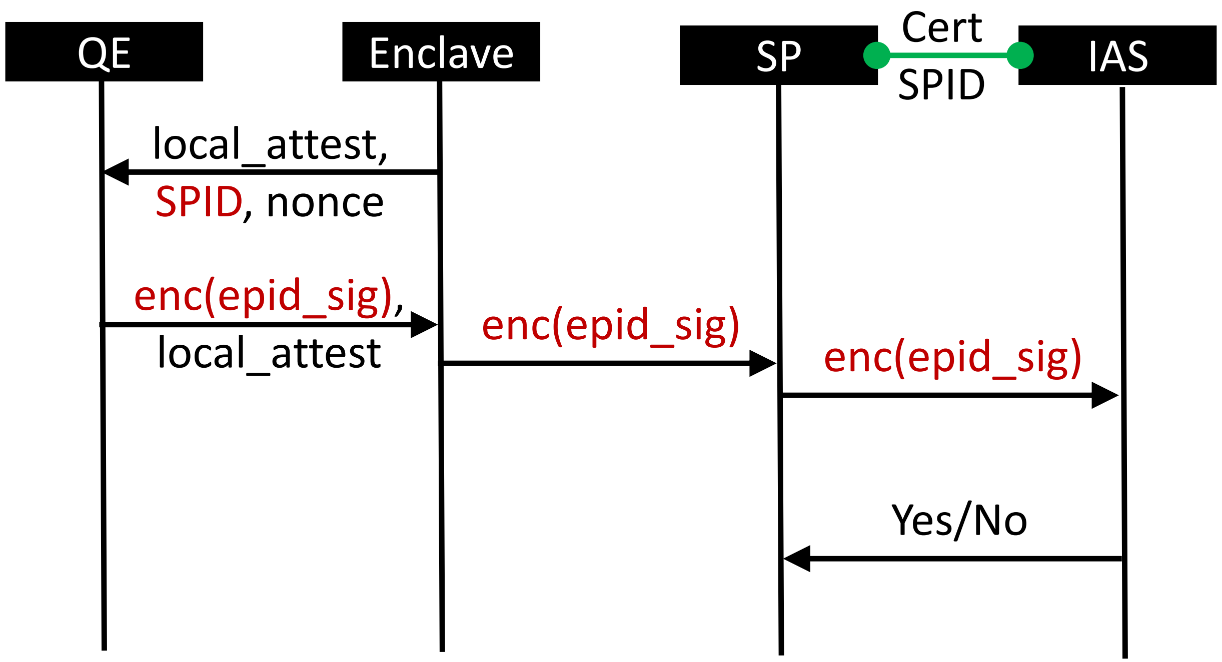


SGX EPID Provisioning

- Each platform given a Provisioning ID (PPID)
  - $\mathtt{ppid} \leftarrow \mathbf{cmac}(\mathtt{provision\_key}, \mathtt{0}_{128})$
  - Platform provisioning is not anonymous
- EpidJoin (PvE/PcE action)
  - ECDSA signature to convince IAS about hardware resident key
  - Encrypts EpidJoinRequest with Provisioning Key
  - Encrypts and uploads member’s private-key using Provisioning Seal Key
- End result
  - Group Membership credentials
  - SGX EPID Group ID

Remote Attestation

- Service Provider (SP) and IAS establish
  - SP Certificate + Service Provider ID (SPID)
- Enclave creates local attestation for QE
  - Optionally request QE to generate local attestation on Quote for Enclave
- QE Creates encrypted EPID Signature
  - Enclave validates QE's local attestation on encrypted Quote
- Enclave sends Encrypted Quote to SP
  - SP cannot validate the quote itself even if it has access to Group Public Key
- SP get Quote Validity YES/NO from IAS



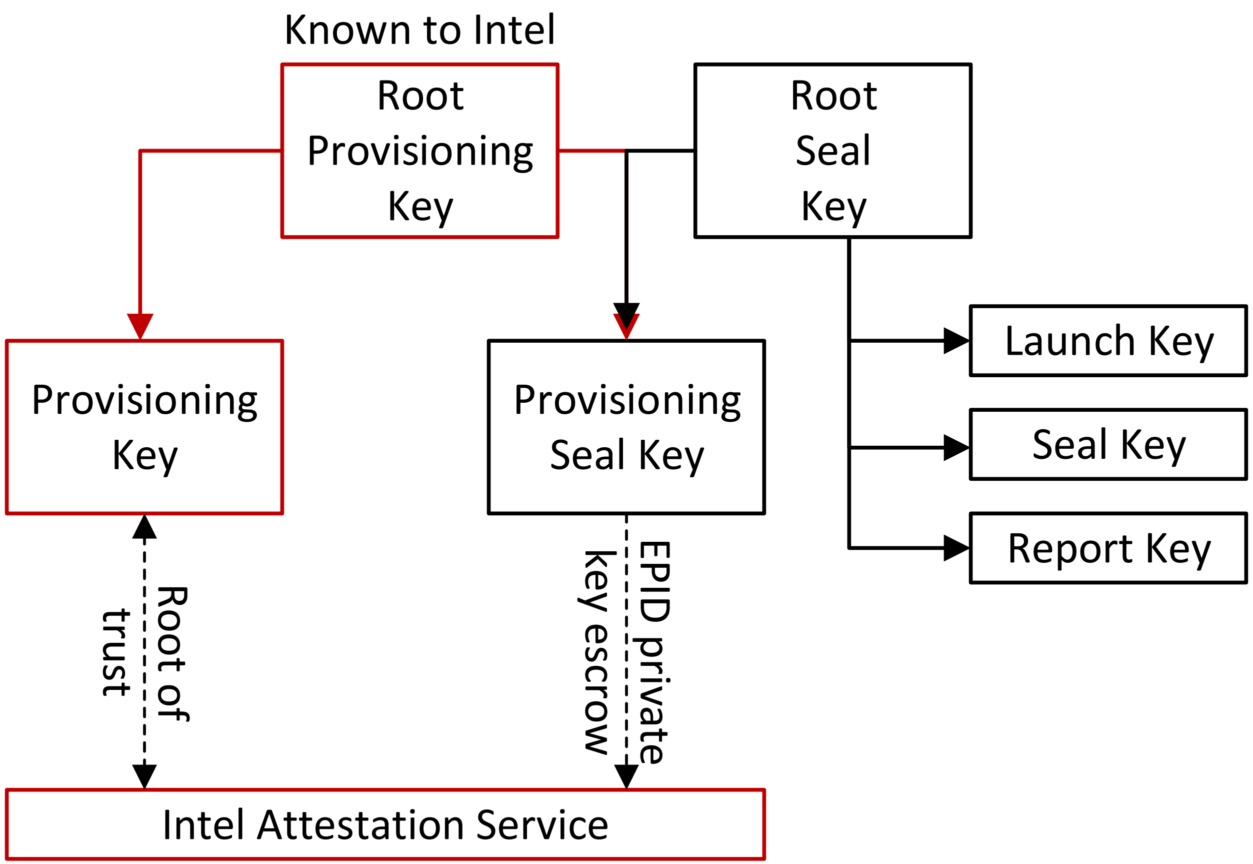
Conclusion

- Provisioning Enclave and Quoting Enclaves are securely implemented
  - But lots of bike shedding crypto
  - Secure against sequential, concurrent, and state malleability
- No privacy in-spite of group signatures
  - Provisioning uses PPID to identify platforms
  - Remote attestation quotes are encrypted and can only be validated by Intel
    - Destroys privacy
    - Could be abused for MitM

Questions?

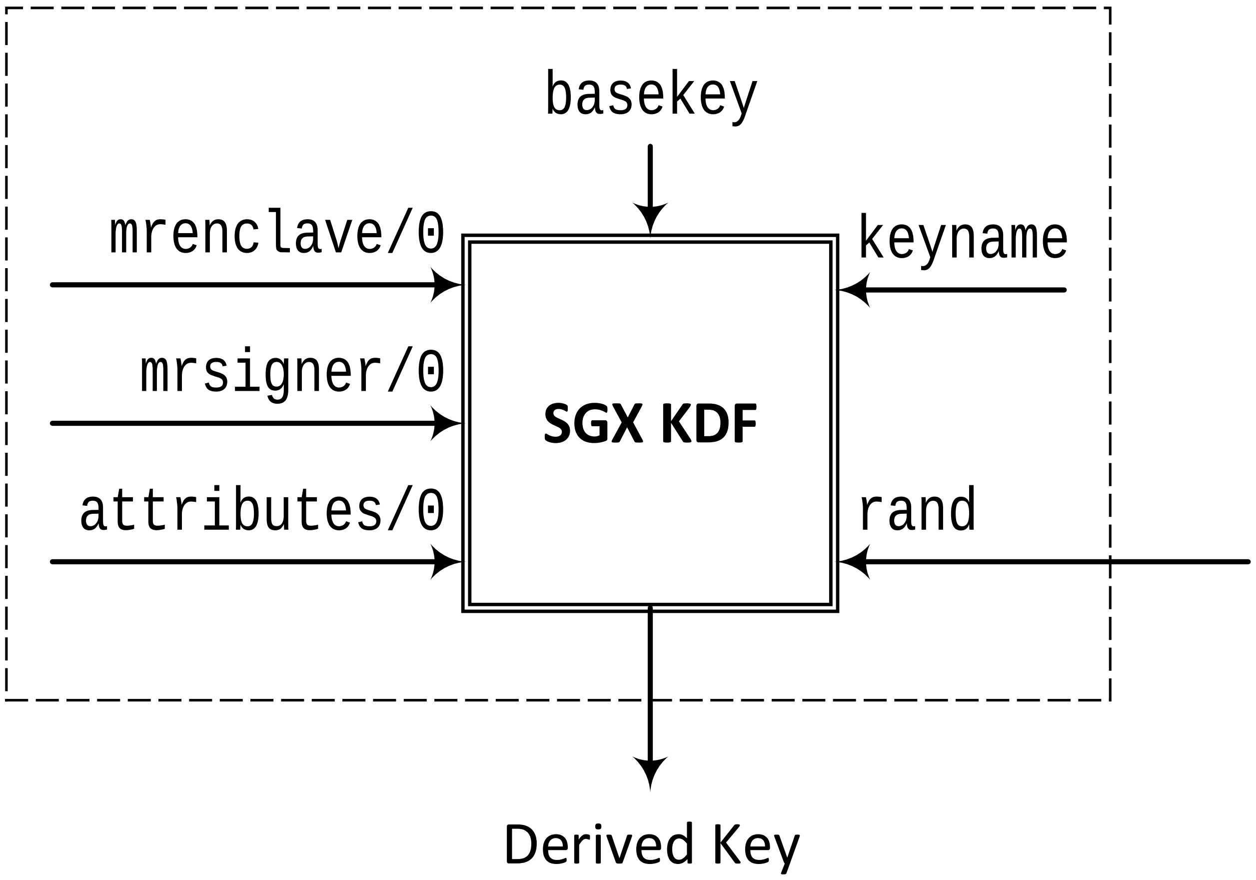
Backup Slides

SGX Key Hierarchy



- Two statistically independent base keys
- Root Provisioning Key
  - Known to Intel
  - Used during EPID Join
- Root Seal Key
  - Intel claims not to know this
  - Unclear if key is generation via oracle access or injected from outside
- Several named-keys derived from base key
  - Coarse Access Control by Launch Enclave

SGX Key Derivation



- Each named key can further be diversified
  - Enclave's own identity (mrenclave, mrsigner, attributes)
  - Potentially adversarial selected 128-bit random number
- Identity used directly from CPU's private data structures
  - Absence ➔ Use zeros
- Keys cannot be diversified for other enclave's identity

SGX Local Attestation

- Allows source enclave to prove that it's identity is valid to any target enclave
  - Allows 512-bits additional report data
- EREPORT provides oracle access to target enclave's ReportKey
  - `cmac` over source enclave's identity present in CPU (cannot be forged)
  - Randomized by CR_REPORT_KEYID set at bootup time
- Target enclave manually computes the CMAC using it's own ReportKey