STORAGE DEVELOPER CONFERENCE

SDC 22

Fremont, CA
September 12-15, 2022

BY Developers FOR Developers

A SNIA Event

Key Per I/O – Fine Grain Encryption For Storage

With TCG's Key Per I/O Encryption Key Selection

Festus Hategekimana, Solidigm Technology
Frederick Knight, NetApp

Agenda

- Data At Rest Protection – Background
- Key Per I/O Overview and Goals
- Key Per I/O Key Flow
- Key Per I/O SSC and I/O Architectures Interactions

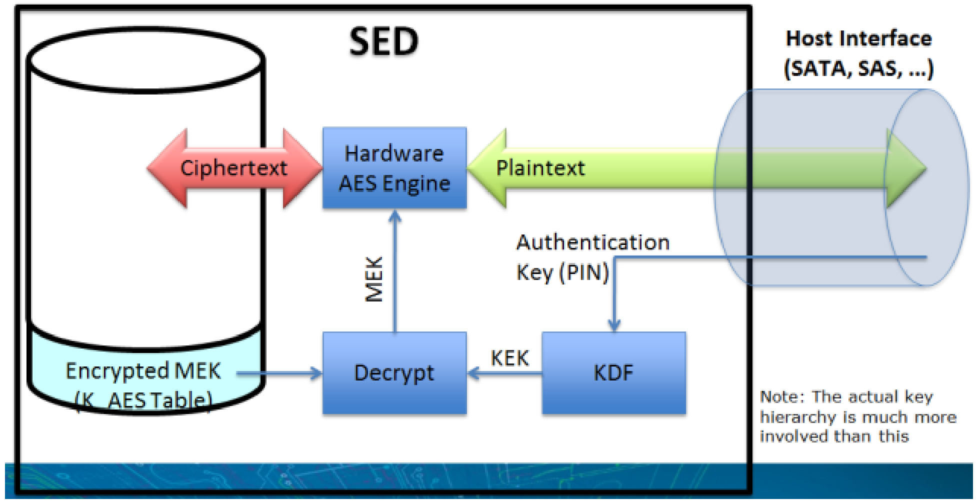
Key Per I/O and Evolution of Data At Rest Protection

Festus Hategekimana, Solidigm Technology

Background on Data At Rest Protection

Data At Rest Protection

Very High-Level Example



Note: The actual key hierarchy is much more involved than this

Properties

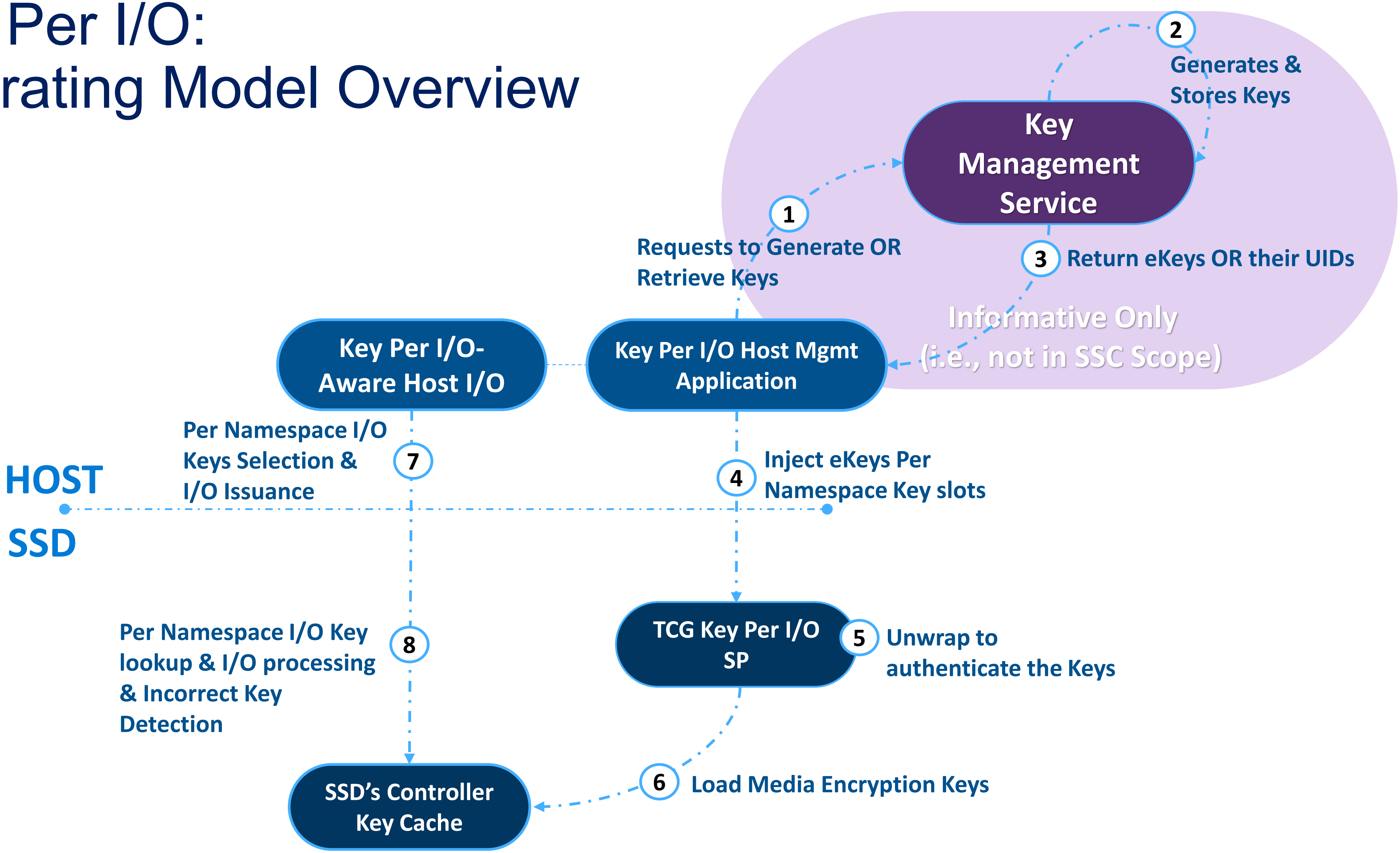
- Encrypt all user accessible data all the time, at interface speeds
- Keys generated & stored in NVM by the storage device
- Media Encryption Key (MEK) associated with contiguous LBA ranges or Namespaces
- Opal/Enterprise SSC* deliver passwords to drive in the clear (when not using Trusted Computing Group (TCG)* - Secure Messaging)

*Other names and brands may be claimed as the property of others.

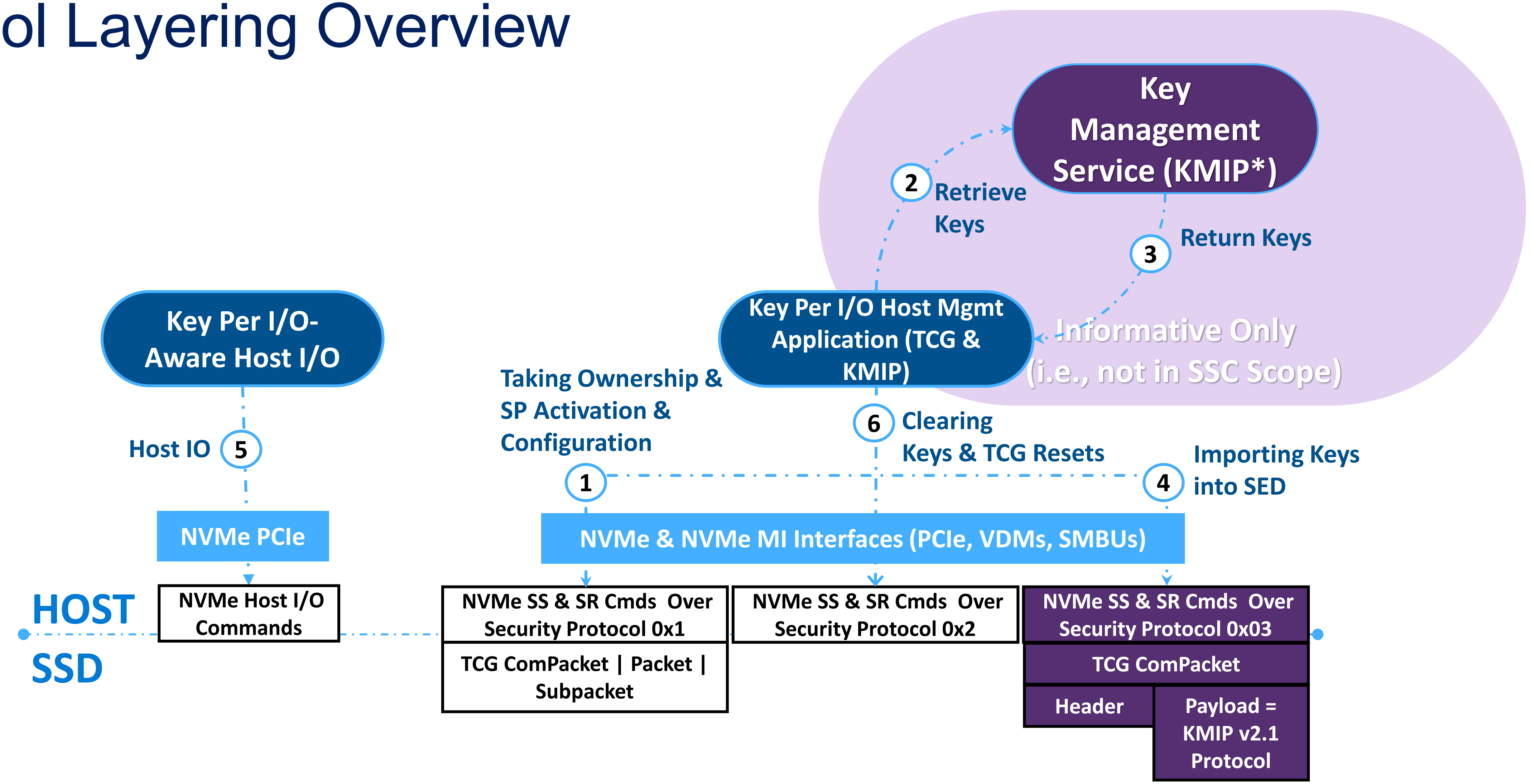
Can we do better?

- Desired Properties:
  - Select an encryption key for each I/O to a Storage Device?
    - Associate encryption domains with higher-level objects (abstractions) than drives or volumes or pre-defined LBA ranges.
    - Crypto erase individual higher-level objects
    - Easier to support European Union's General Data Protection Regulations' "Right to be forgotten"
  - Externally manage Media Encryption keys?
    - Centralized key management infrastructure, consistent key policies
    - High assurance key generation and control, e.g., master keys in HSM (Hardware Security Module)
  - Ensure that a Storage Device with no power has no encryption keys?
    - Shorter physical drive loss/theft discussion with security auditor
    - Easier decommissioning process

Key Per I/O: Operating Model Overview


Key Management Service
2
Generates & Stores Keys
1
Requests to Generate OR Retrieve Keys
3
Return eKeys OR their UIDs
Informative Only (i.e., not in SSC Scope)
Key Per I/O-Aware Host I/O
Key Per I/O Host Mgmt Application
Per Namespace I/O Keys Selection & I/O Issuance
7
4
Inject eKeys Per Namespace Key slots
HOST
SSD
Per Namespace I/O Key lookup & I/O processing & Incorrect Key Detection
8
TCG Key Per I/O SP
5
Unwrap to authenticate the Keys
6
Load Media Encryption Keys
SSD's Controller Key Cache

Key Per I/O:
Protocol Layering Overview

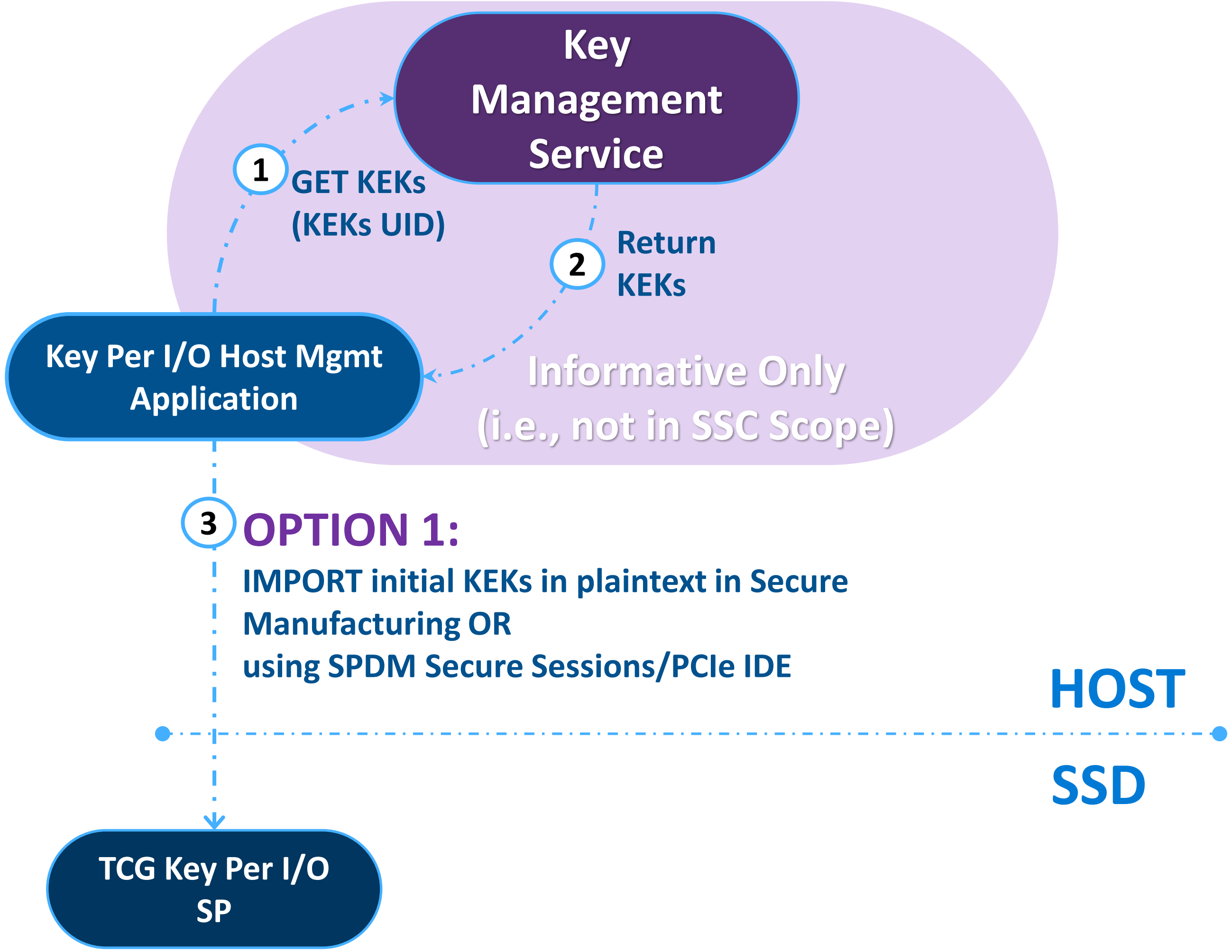

Key Management Service (KMIP*)
2 Retrieve Keys
3 Return Keys
Key Per I/O Host Mgmt Application (TCG & KMIP)
Informative Only (i.e., not in SSC Scope)
Key Per I/O-Aware Host I/O
Host IO 5
Taking Ownership & SP Activation & Configuration
1
6 Clearing Keys & TCG Resets
4 Importing Keys into SED
NVMe PCIe
NVMe & NVMe MI Interfaces (PCIe, VDMs, SMBUs)
HOST
SSD
NVMe Host I/O Commands
NVMe SS & SR Cmds Over Security Protocol 0x1
TCG ComPacket | Packet | Subpacket
NVMe SS & SR Cmds Over Security Protocol 0x2
NVMe SS & SR Cmds Over Security Protocol 0x03
TCG ComPacket
Header
Payload = KMIP v2.1 Protocol

*Other names and brands may be claimed as the property of others.

Key Per I/O: Injecting Keys

KEKs Transport Security:

- Objectives:
  - Confidentiality
    - **Option 1**: Initial KEKs are pre-shared in secure manufacturing or over secure transport link (e.g., SPDM* Secure session, PCIe IDE*)
  - Authentication & integrity
    - Subsequent KEKs are transported wrapped via AES-GCM or NIST AES-KeyWrap using initial KEKs
    - KEKs are stored persistently to authenticate other keys across resets



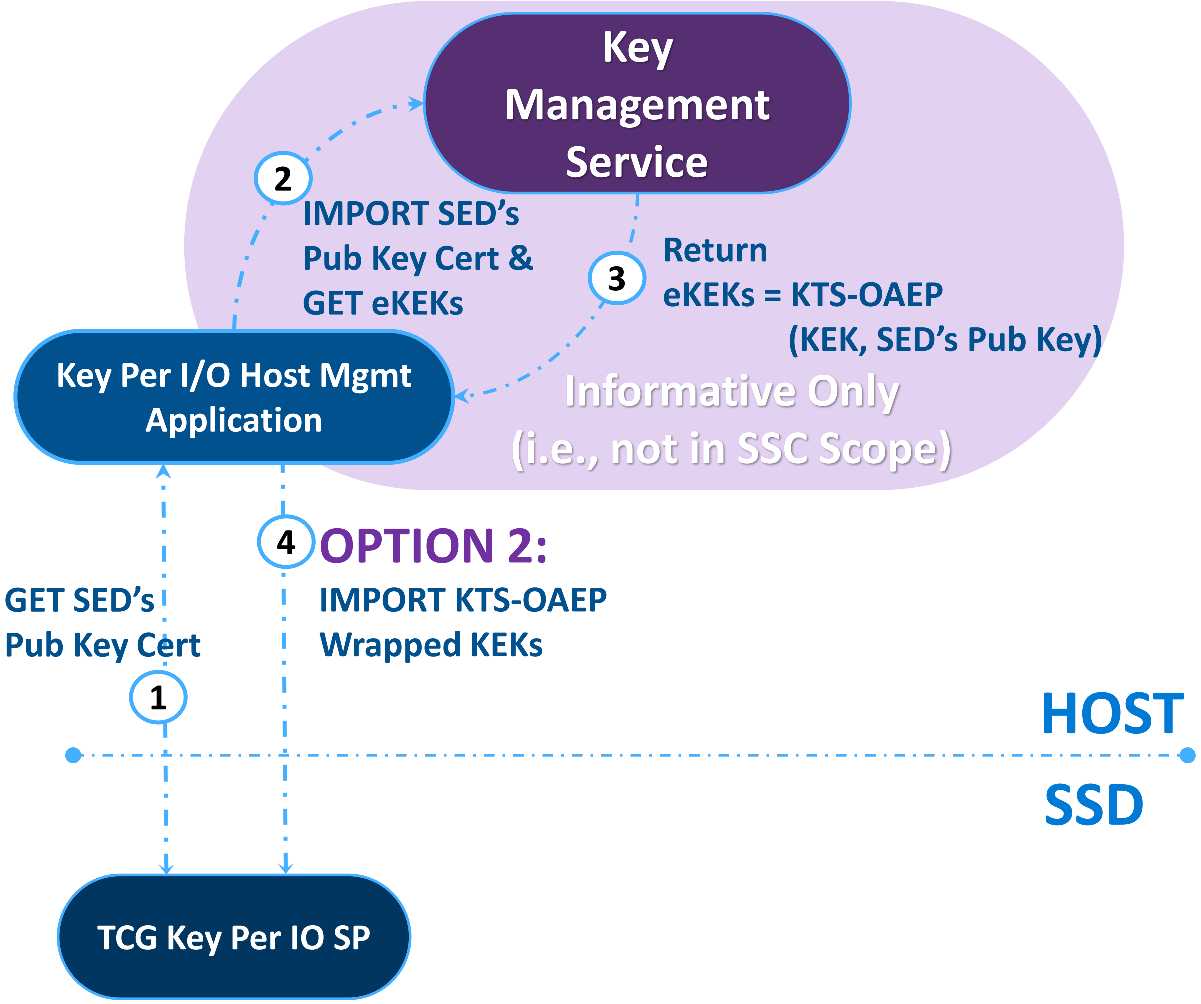
Key Management Service
1
GET KEKs (KEKs UID)
2
Return KEKs
Key Per I/O Host Mgmt Application
Informative Only (i.e., not in SSC Scope)
3
OPTION 1:
IMPORT initial KEKs in plaintext in Secure Manufacturing OR using SPDM Secure Sessions/PCIe IDE
HOST
SSD
TCG Key Per I/O SP

*Other names and brands may be claimed as the property of others.

Key Per I/O: Injecting Keys

KEKs Transport Security:

- Objectives:
  - Confidentiality
    - **Option 2**: Initial KEKs are pre-shared using KTS-OAEP & SED's Pre-provisioned Public Key Certificate
  - Authentication & Integrity
    - Subsequent KEKs are transported wrapped via AES-GCM or NIST AES-KeyWrap using initial KEKs
    - KEKs are stored persistently to authenticate other keys across resets

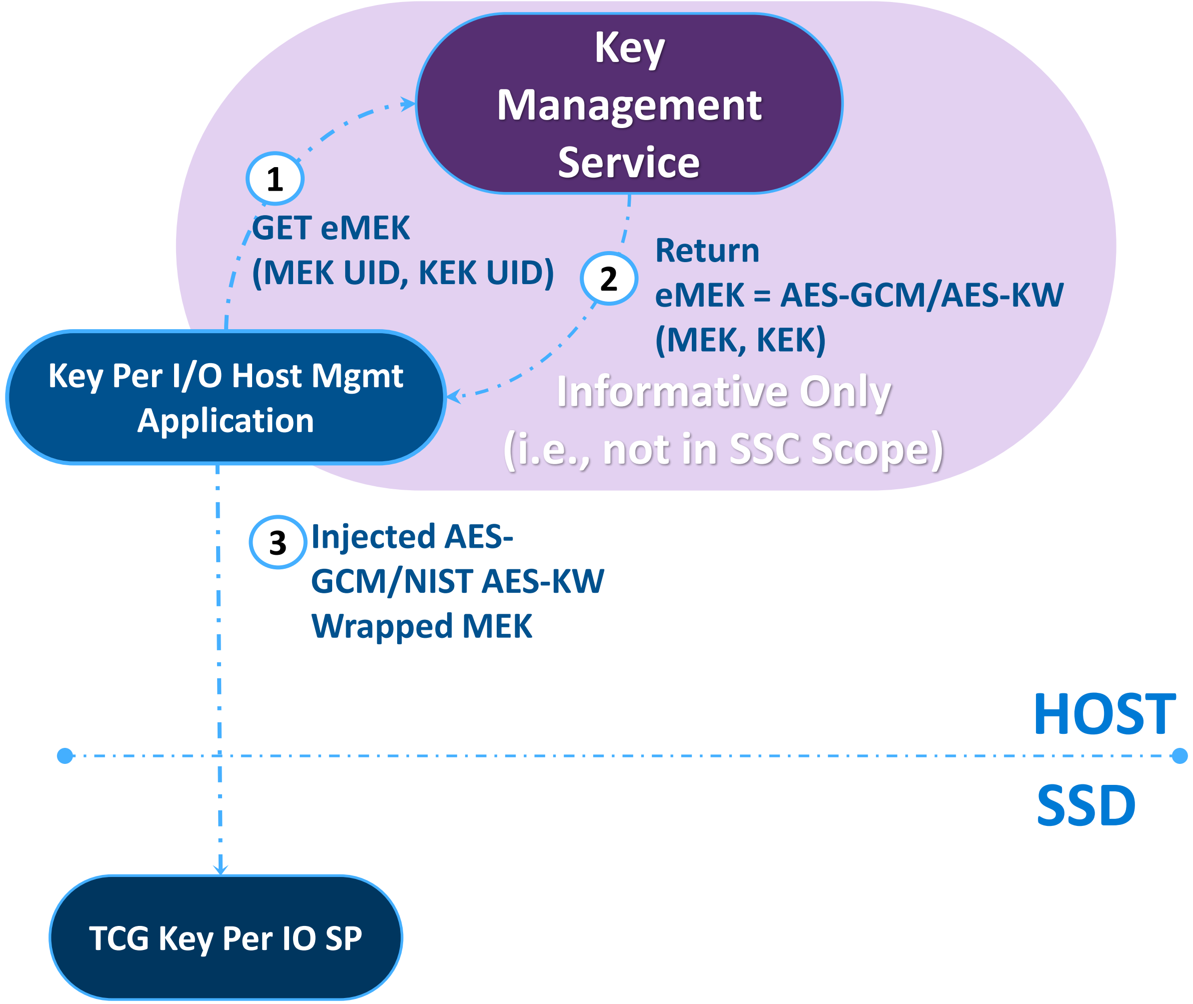


Key Management Service
2
IMPORT SED's Pub Key Cert & GET eKEKs
3
Return eKEKs = KTS-OAEP (KEK, SED's Pub Key)
Key Per I/O Host Mgmt Application
Informative Only (i.e., not in SSC Scope)
4
OPTION 2:
IMPORT KTS-OAEP Wrapped KEKs
GET SED's Pub Key Cert
1
HOST
SSD
TCG Key Per IO SP

Key Per I/O:
Injecting Keys

MEKs Transport Security:

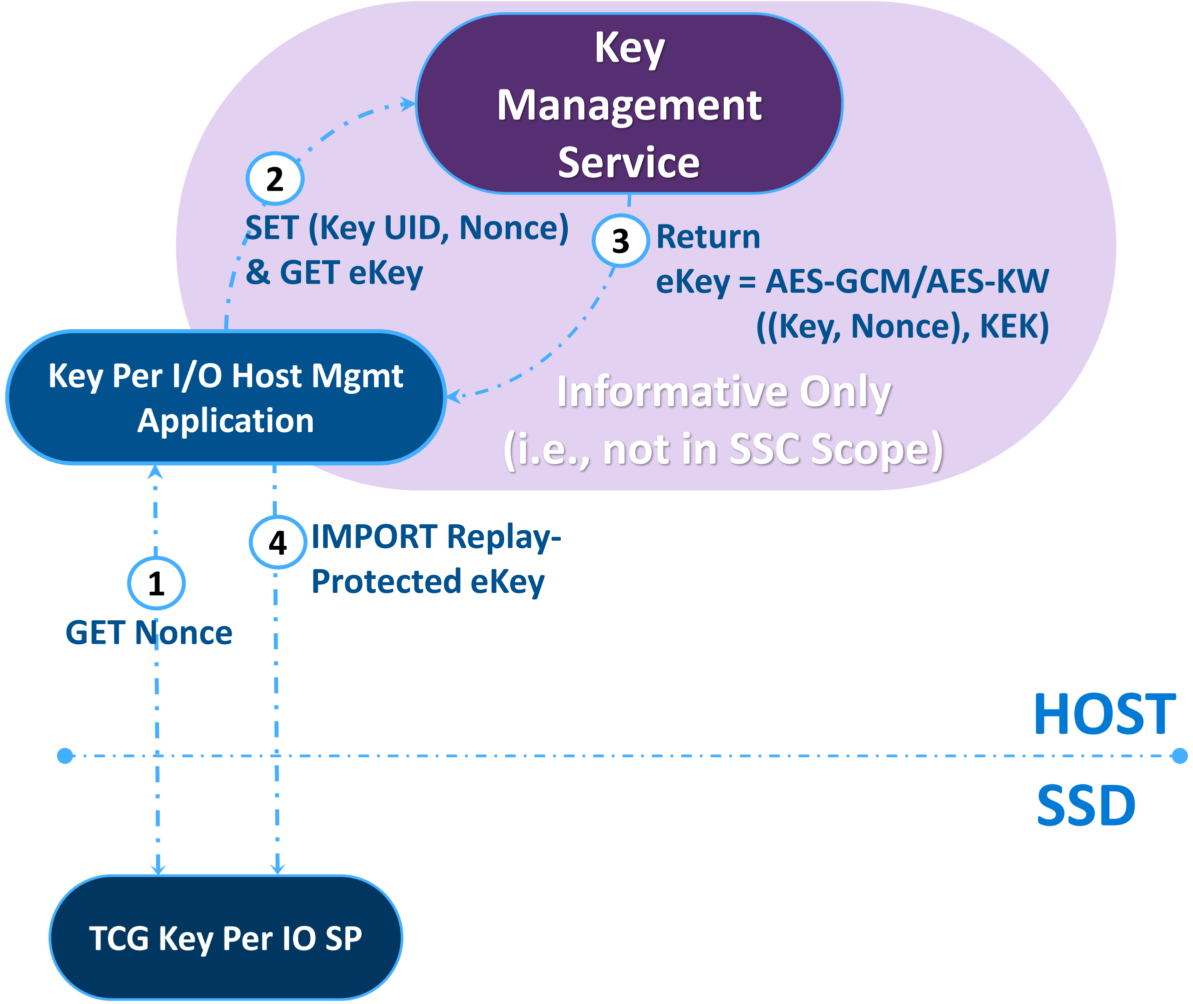
- Objectives:
  - Confidentiality, Authentication & Integrity
    - MEKs are transported wrapped via AES-GCM or NIST AES-KeyWrap using pre-shared KEKs
    - SSD loses MEKs on power cycle.
    - Host re-injects MEKs on subsequent boot to access data again.



Key Per I/O: Injecting Keys

Replay Protection:

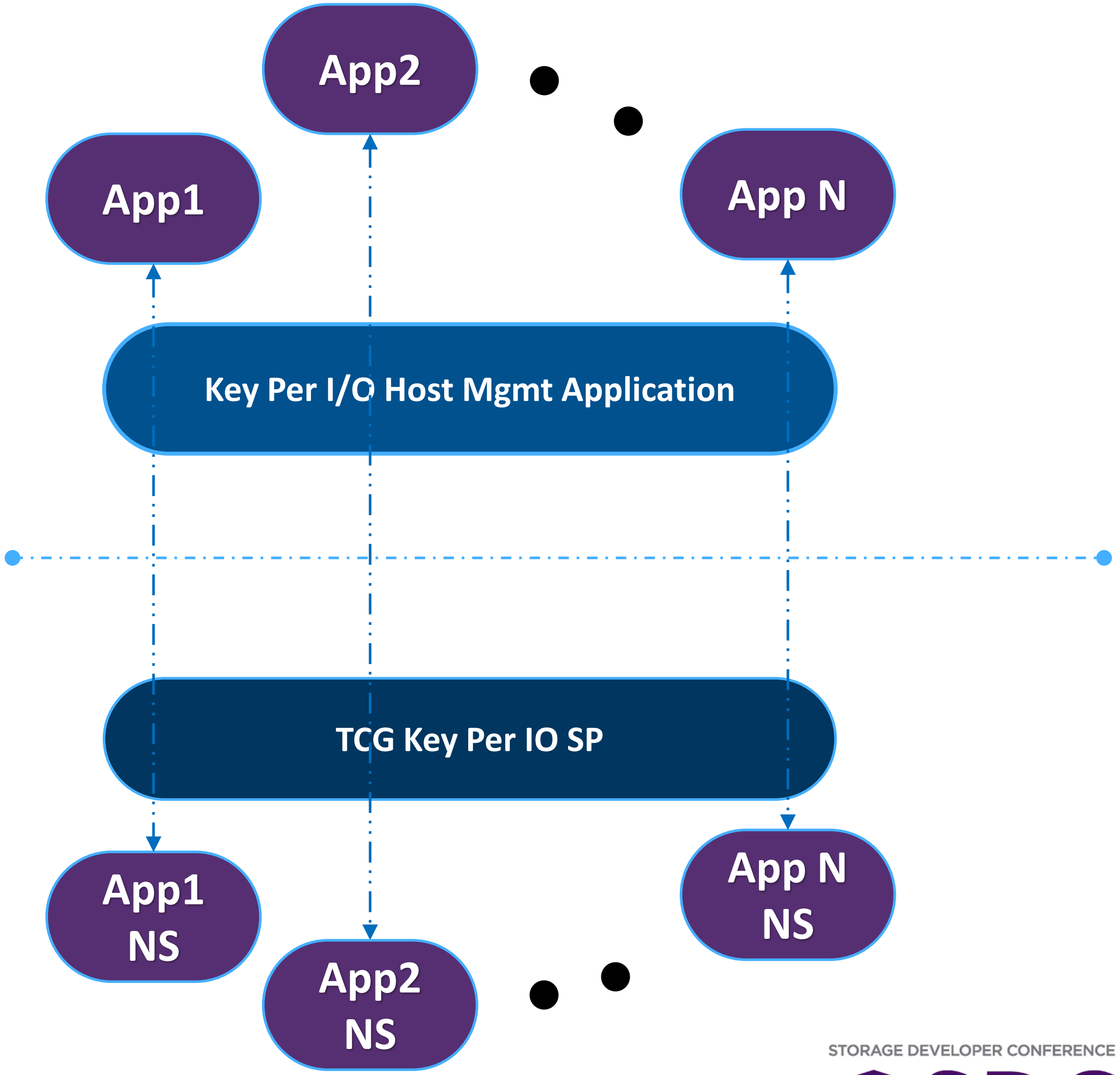
- Objective:
  - Preventing replay of old copies of keys by unauthorized entity
    - Nonce is cryptographically tied to key during injection.
    - Encryption authentication tag must encompass both key & Nonce


Key Management Service
2
SET (Key UID, Nonce) & GET eKey
3
Return eKey = AES-GCM/AES-KW ((Key, Nonce), KEK)
Key Per I/O Host Mgmt Application
Informative Only (i.e., not in SSC Scope)
1
GET Nonce
4
IMPORT Replay-Protected eKey
HOST
SSD
TCG Key Per IO SP

Key Per I/O:
Injecting Keys

Future Work:

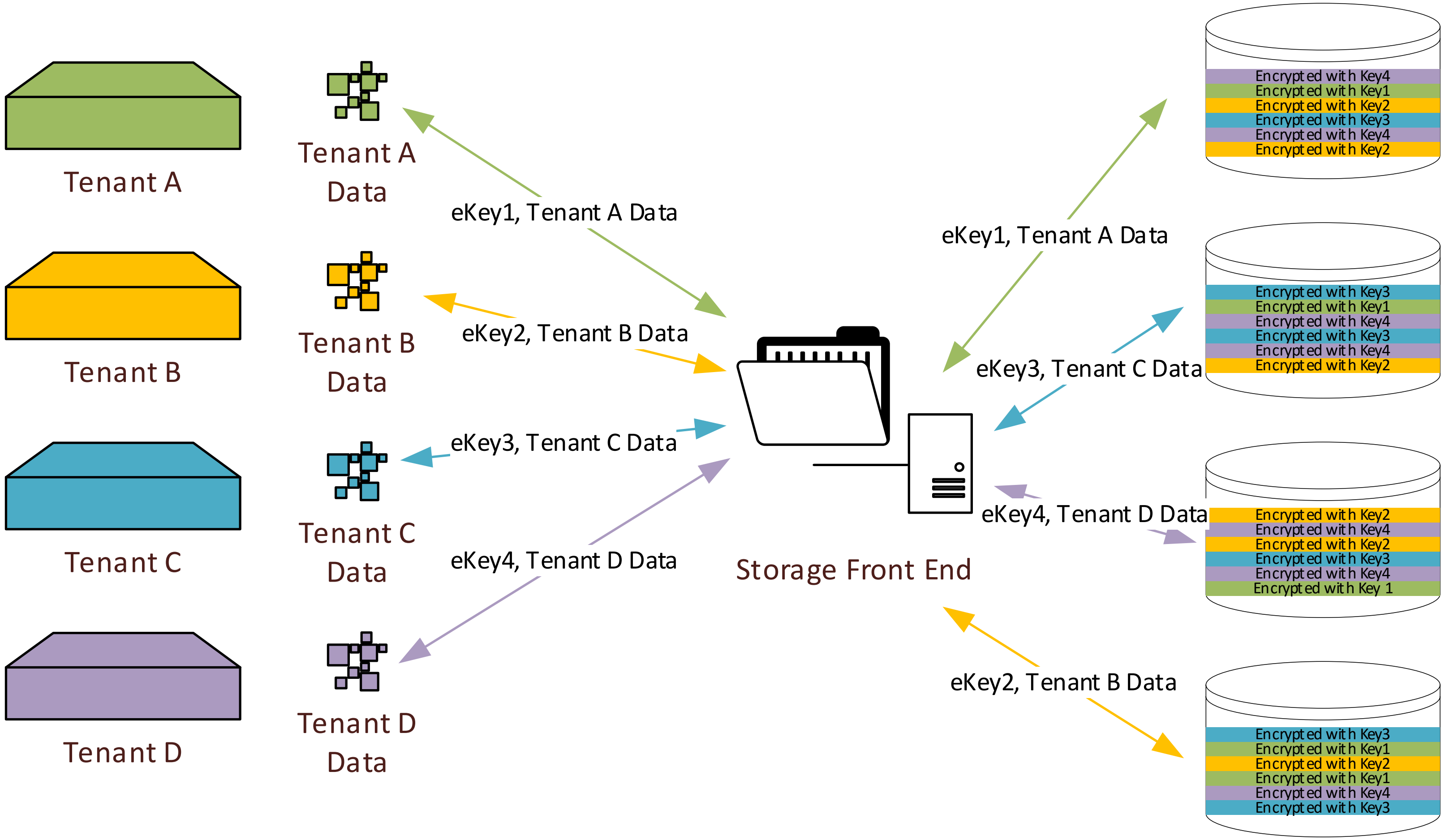
- Objective:
  - Per NS confidentiality, authentication & integrity establishment using ephemeral KEKs ?
    - SP800-56A C(1e, 2s) to derive Key Derivation Keys (KDKs) between the SSD & Key Per IO Host Mgmt App.
    - Per NS KEKs = KDF(KDKs, Per App Derivation Data)
    - Ephemeral KEKs are used to encrypt MEKs
    - SSD loses both KEKs & MEKs on power cycle
    - Across power cycles, SSD-Host authentication done via static public key certificates



Key Per I/O SSC and I/O Architecture Interactions

Frederick Knight, NetApp

KPIO Use Cases



Tenant A

Tenant A Data

Tenant B

Tenant B Data

Tenant C

Tenant C Data

Tenant D

Tenant D Data

eKey1, Tenant A Data

eKey2, Tenant B Data

eKey3, Tenant C Data

eKey4, Tenant D Data

Storage Front End

eKey1, Tenant A Data

eKey3, Tenant C Data

eKey4, Tenant D Data

eKey2, Tenant B Data

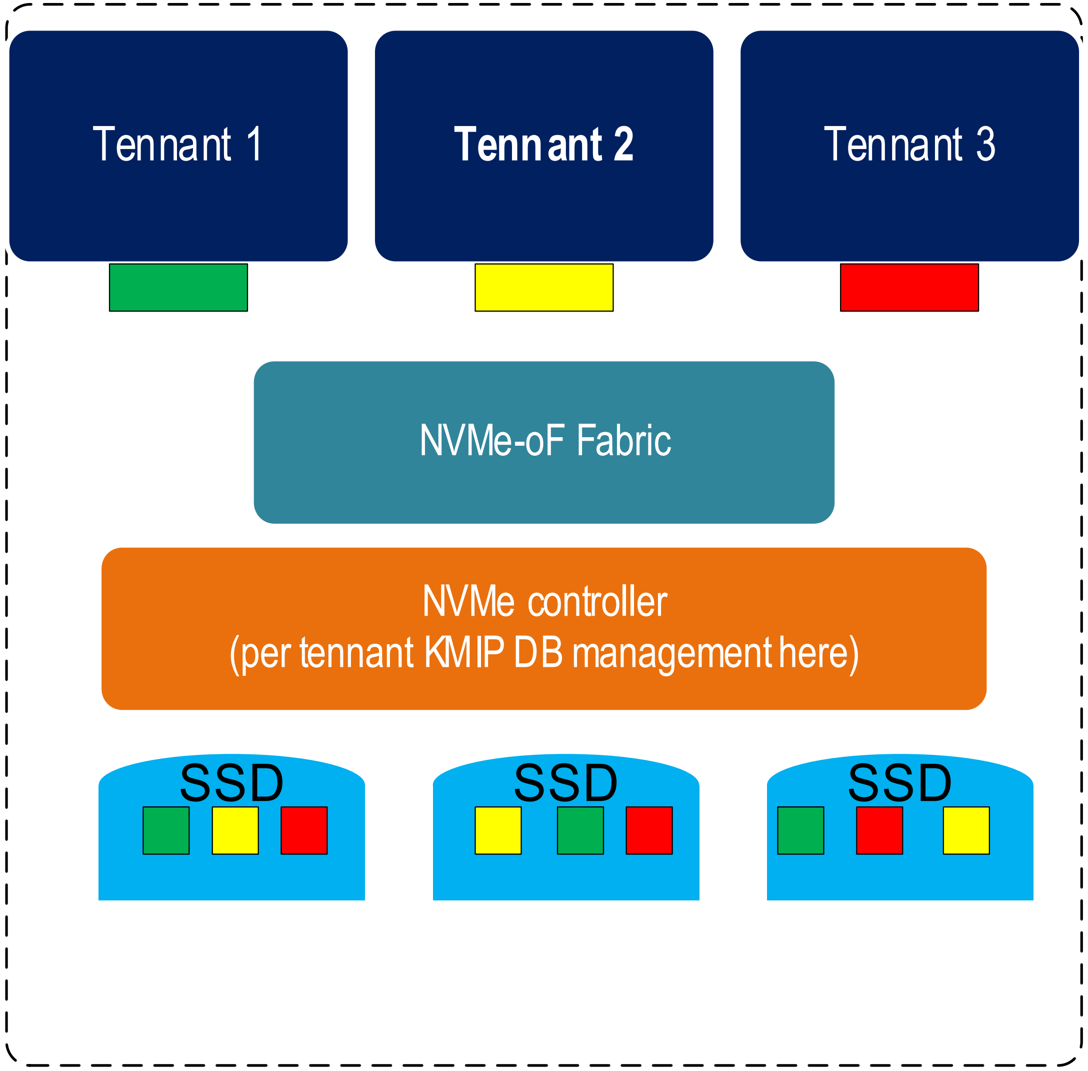
Encrypted with Key4
Encrypted with Key1
Encrypted with Key2
Encrypted with Key3
Encrypted with Key4
Encrypted with Key2

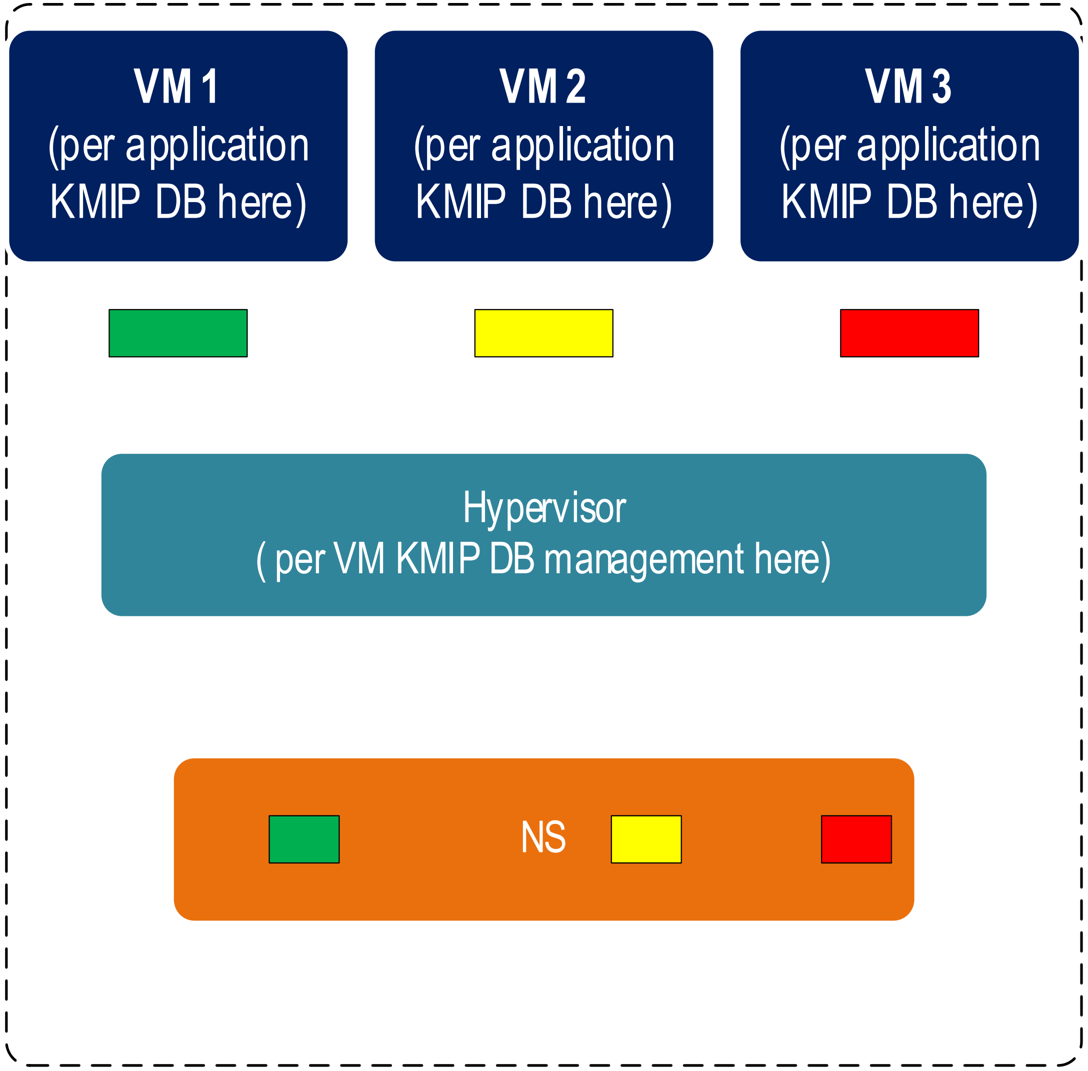
Encrypted with Key3
Encrypted with Key1
Encrypted with Key4
Encrypted with Key3
Encrypted with Key4
Encrypted with Key2

Encrypted with Key2
Encrypted with Key4
Encrypted with Key2
Encrypted with Key3
Encrypted with Key4
Encrypted with Key 1

Encrypted with Key3
Encrypted with Key1
Encrypted with Key2
Encrypted with Key1
Encrypted with Key4
Encrypted with Key3

KPIO Use Cases





KPIO Discovery

Host Detection of KPIO

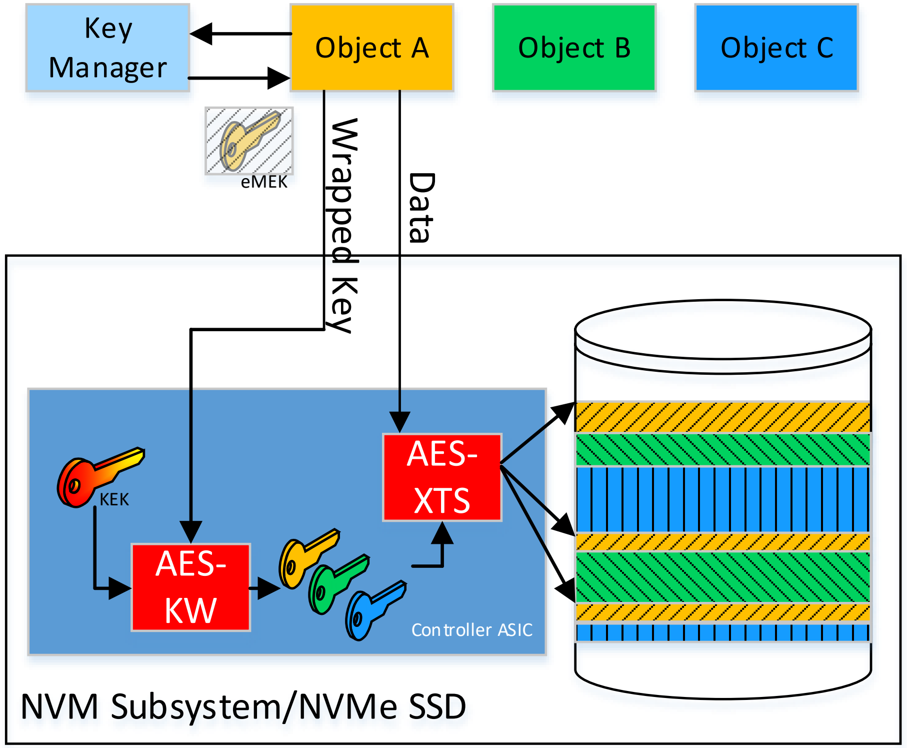
- Number of Key Tags supported
- Granularity and alignment of operations
- NVMe Identify command
  - Per namespace
- TCG Discovery (Security Send and Security Receive)
  - Authenticate
  - Security Receive (Level 0 Discovery)
  - Discovery security characteristics

KPIO Configuration

Establish KEKs (Key Encryption Keys)

- Agreement between host and device for secure transmission of the KEKs (secure manufacturing (pre-shared keys), public/private key pairs (PKI), certificate, etc)
- Host obtains MEK (Media Encryption Keys) from a key management database (e.g., KMIP)
- MEKs are “wrapped” with KEKs and sent to the device

Wrapped MEKs sent from the host to the device



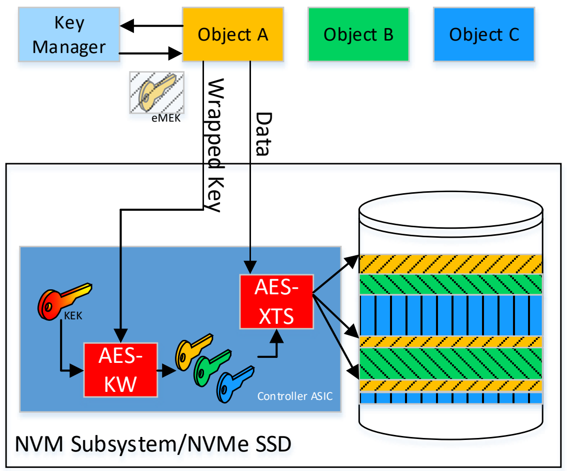
KPIO Configuration

MEKs are Loaded the Device Key Cache

- Associate each MEK injected with a Key Tag
  - Per namespace – loaded using Security Send command

Key Tag	MEK example (256 bit)
1	0x1234567890ABC…90ABCDEF
2	0x1234567890ABC…90ABCDE0
100	0x1234567890ABC…90ABCDE1
101	0x1234567890ABC…90ABCDE2
103	0x1234567890ABC…90ABCDE3
200	0x1234567890ABC…90ABCDE4
217	0x1234567890ABC…90ABCDE5

KPIO Configuration



Load New Keys

- Associate a Key Tag with a different MEK
  - Per namespace – loaded using Security Send command

Key Tag	MEK example (256 bit)
1	0x1234567890ABC…90ABCDEF
2	0x1234567890ABC…90ABCDE0
100	0x1234567890ABC…90ABCDE6
109	0x1234567890ABC…90ABCDE7
110	0x1234567890ABC…90ABCDE8
111	0x1234567890ABC…90ABCDE9
220	0x1234567890ABC…90ABCDEA

KPIO Usage

NVMe I/O Command Usage

- Compare
- Copy
- Verify
- Read
- Write
- Write Zeroes
- Zone Append

- A field in each command to specify the Key Tag value to use for that individual I/O
- An indicator that a Key Tag is present

Key Tag	MEK example (256 bit)
1	0x1234567890ABC…90ABCDEF
2	0x1234567890ABC…90ABCDE0
100	0x1234567890ABC…90ABCDE6
109	0x1234567890ABC…90ABCDE7
110	0x1234567890ABC…90ABCDE8
111	0x1234567890ABC…90ABCDE9
220	0x1234567890ABC…90ABCDEA

KPIO NVMe Example Commands

- WRITE (LBA=100, LEN=8, flag=1, keytag=1)

MEK = 0x1234567890ABC…90ABCDEF

- WRITE (LBA=200, LEN=16, flag=1, keytag=100)

MEK = 0x1234567890ABC…90ABCDE6

- READ (LBA=100, LEN=8, flag=1, keytag=1)
  - Gets your data back
- READ (LBA=200, LEN=16, flag=1, keytag=1)
  - Gets error or bad data
- READ (LBA=200, LEN=16, flag=1, keytag=100)
  - Gets your data back

Flag = 1 means the keytag is present

Key Tag	MEK example (256 bit)
1	0x1234567890ABC…90ABCDEF
2	0x1234567890ABC…90ABCDE0
100	0x1234567890ABC…90ABCDE6
109	0x1234567890ABC…90ABCDE7
110	0x1234567890ABC…90ABCDE8
111	0x1234567890ABC…90ABCDE9
220	0x1234567890ABC…90ABCDEA

KPIO Impact in TCG

Security Send / Security Receive Commands

- Uses new TCG protocol ID (0x03)
- Authentication
- Discovery
- Key Injection method (Establish Key Tag to MEK association)
- Key Clear method (Remove Key Tag to MEK association)
- Key Replacement method (Replace MEK for a Key Tag)
- Securely Purge Key Cache
- Define encryption / decryption algorithms that can be supported (e.g., XTS-AES-256)

KPIO Impact On Hosts

Host Responsibilities to use KPIO

- Hosts must manage the full life cycle of the Keys
  - Including secure purging of the keys
- Host is responsible for the correctness of the MEK injection / key tag association and use of the correct key tag for each I/O command
- Host is responsible for preventing incorrect key tag use
  - Key tag associations may change during operation – such as key tag cache size smaller than key tag needed usage
  - Using the key tag associated with the correct MEK
- Host must handle errors for improper use of key tags
  - Invalid key tag value (out of range), or a key tag with no associated MEK
  - Trying to use a key tag before injection is complete or after removal

KPIO Project Status

Current Key Topics in Progress

- Testing use cases
- Overall, still a work in progress in some respects; but almost completed
- Next generation features …

- NVMe work in final review!

  https://nvmexpress.org/

- TCG work in final review!

  https://trustedcomputinggroup.org

- Come join us at TCG Storage Work Group to continue the discussions!
  - Email: admin@trustedcomputtinggroup.org

KPIO For Other IO Architectures

What about SCSI and/or SATA

- The same TCG architecture is used by SCSI and SATA
  - Security Send / Security Protocol Out / Security Receive / Security Protocol In
- But completely new I/O commands would be required
  - Such as 32-byte CDBs for SCSI (to carry the Key Tag value)

- Limited interest being shown to undertake such an effort at this time

KPIO Key Takeaways

- The KPIO SSC is being defined such that an SD that claims TCG Opal SSC compatibility could be a KPIO SSC.
- Intended to protect confidentiality of data at rest from unauthorized access once it leaves the owner’s control.
- Creating a fine-grained approach to enhance SED technology to better support multi-tenancy usage models.
- Standards based designs for multi-vendor interoperability.

Who are TCG and NVMe?

Trusted Computing Group (TCG) is a not-for-profit organization formed to enable secure computing through open standards and specifications. Benefits of TCG technologies include protection of business-critical data and systems, secure authentication and strong protection of user identities, and the establishment of strong machine identity and network integrity. Trusted hardware and applications reduce enterprise total cost of ownership and support regulatory compliance. Through its member-driven work groups, TCG enables the benefits of trust in computing devices from mobile to embedded systems, as well as networks, storage, infrastructure, and cloud security. Almost all enterprise PCs, many servers, and embedded systems include the TPM; while networking equipment, drives, and other devices and systems deploy other TCG specifications, including self-encrypting drives and network security specifications.

The original NVM Express Work Group was incorporated as NVM Express in 2014 and is the consortium responsible for the development of the NVM Express specification. The organization currently has over 100 member companies.

NVM Express is an open collection of standards and information to fully expose the benefits of non-volatile memory in all types of computing environments from mobile to data center.

NVMe is designed from the ground up to deliver high bandwidth and low latency storage access for current and future NVM technologies.

Please take a moment to rate this session.

Your feedback is important to us.