Key Management as the Foundation of Data Security

The security of cryptographic protocols used to protect data at rest, in motion, and platform trust relies critically on the security of cryptographic keys. Growing sophistication in software and hardware attacks, cloud usage, and regulatory requirements mandate that private keys should not be exposed in the clear during their entire life cycle: generation, storage, and run time.

Regulated industry verticals such as banking, healthcare, and e-commerce require customer passwords, keys, and Personal Identification Numbers (PINs) are secured through industry certified key management devices called Hardware security modules (HSM). HSMs are deployed by enterprises either in standalone configurations or by cloud service providers as a service offering. While offering ironclad security for key management and crypto services, traditional HSM appliances have the drawback of a high price to performance ratio and require purpose-built hardware, making them difficult to be scaled out as a virtual network function (VNF).

Current uses of HSMs are restricted to a few key IT niches, due to their very high price/performance ratio. Cloud migration of enterprise applications, together with regulatory compliance, demand that tenants' cryptographic keys are invisible to the cloud operator, besides being immune to SW attacks. While cloud operators provide HSMs as an additional service, increasing HSM bandwidths to cloud scale can be prohibitively expensive. In many cases, cryptographic keys determine the last pieces of infrastructure that could not be migrated to the cloud for security and compliance reasons. In this backdrop, HSM vendors and Cloud Service Providers (CSPs) are also looking for ways to virtualize the HSM functions without compromising security.

Key Management Usage Examples

Web servers are identified by a signing private key, such as an RSA or ECDSA keys, whose public key is registered with a certificate authority. Web clients verify the identity (certificate) of the server by first verifying the public key (along with other public parameters such as the IP Address, and validity) with the certificate authority, before trusting signatures that are produced by the web server. By stealing the private key, an attacker can impersonate the original web server, resulting in loss of business, reputation, and trust of the initial web server and its owning business.

Databases support a well-known security feature called Transparent Data Encryption (TDE) for encrypting database content as a whole. TDE is supported by a 3-level key hierarchy comprising of: a service master key (SMK), which is a symmetric key. Asymmetric certificate key (CK), and Symmetric data encryption key (DEK). In a typical operational sequence, the SMK is stored in a remote key server or an HSM, while the DEK is attached to the database in an encrypted format. The DEK is recovered when starting up the database and is co-located with the data. Storing the DEK on the server can make it vulnerable to software attacks.

Storage encrypting file systems such as eCryptfs, feature an ability to encrypt individual files using symmetric keys. Each file is associated with a different symmetric key, with the symmetric key wrapped with yet another use-specific asymmetric private key. This master key, or key encryption key (KEK), is stored either an HSM, a Secure Token or a TPM. When a file is first accessed, the wrapped symmetric key attached to the file is first retrieved by requesting the hardware key server to unwrap the key.
Intel® Key Protection Technology

The new Intel® Xeon® processor Scalable family-based server platforms, (formerly code-named Purley), have a new platform security feature called Intel® Key Protection Technology (Intel® KPT) for helping to secure keys in hardware. KPT augments Intel's® QuickAssist Technology (Intel® QAT) hardware crypto accelerator, integrated with the Intel® C627-C628 Chipset, with the run-time protection of cryptographic keys using Intel® Platform Trust Technology (Intel® PTT) for persistent key storage at rest. Intel KPT, when complemented with popular crypto API frameworks like OpenSSL® and PKCS#11, lends itself to a high performance, scalable, security-enabled crypto, and key management framework on the server platform. Intel KPT enables the capability of a high throughput hardware security module (HSM) on the standard IA server. Existing applications integrated with traditional key management solutions like KMIPS, Barbican, and network HSMs can seamlessly take advantage of Intel KPT through suitable software adaptation layer. Intel® QuickAssist Technology supports virtualization in all standard device virtualization models, thus making Intel KPT available to VMs.

The Intel® C627 Chipset, the Peripheral Control Hub (PCH) in Intel® Xeon® processor Scalable Family servers supports the integrated Intel QAT HW crypto accelerator with a performance of 100 Gbps, symmetric crypto, and 100K RSA2K ops/sec. Intel QAT, together with the Intel® PTT, presents a unique opportunity to help protect keys from all SW attacks and simple hardware (probe) attacks. This capability lends itself to new solution frameworks for enabling high throughput platform HSMs on volume Intel servers, making hardware key management pervasive and available to a large class of applications that currently do not have access to HSMs. Furthermore, the Intel KPT also optimizes the round trip latency between applications and HSM appliances, this removing performance barriers to HSM adoption.

Comparison Between HSM and Intel® KPT Usage Models

Figure 1 contrasts the usage sequence of a network HSM appliance with that of an Intel KPT.

With a traditional network HSM, an application running on a computer server first starts a session with a remote (network) HSM appliance and presents its identity and credentials to access its keys more securely. (Application keys never leave the HW boundary of the HSM). At run time, the application sends ‘cryptography work’ requests to the HSM appliance, the applications’ keys are used to compute the requested crypto operations within the security-enabled hardware boundary of the HSM. Results of the computation (e.g., signatures or hashes) are returned to the calling application over the network.

With Intel KPT, the application’s first step consists of logging into the remote HSM, or key server, and requesting a secure transfer of the key from the key server to the local (compute) server. In the second step, the key is transferred more securely over the network to the computing platform’s hardware for local run time storage. With Intel KPT, this local storage is in the Intel PTT inside the PCH. Runtime usage of the key comprises of transferring the key to the Intel

Some benefits of Intel KPT over the traditional HSM model are:

• Reduced latency for runtime crypto operations performed locally on the compute server
• High crypto throughput (up to 100 Gbps of symmetric crypto throughput and 100,000 RSA2K ops/sec) available locally on the platform.
• Hardware features compatible with support for a platform HSM as a virtual network function.
• Lower price/performance for HSM functionality.

Intel® KPT Architecture Overview

Intel KPT is a new feature in the Intel® Xeon® processor Scalable Family server based platform for helping to protect sensitive private keys in hardware at runtime. Intel KPT leverages a combination of Intel server platform technologies: Intel QAT Cryptographic Accelerator, Intel PTT and Converged Security Manageability Engine (CSME) – all are integrated into the Intel® C625-C628 Chipsets.

Intel KPT leverages the proximity between Intel QuickAssist and CSME in the Intel® C625-C628 Chipsets. Intel QuickAssist and CSME are integrated into the PCH and connected through a private hardware link that is invisible to x86 SW and cannot be reached with simple hardware probes. Furthermore, CSME implements Intel PTT, which is Intel’s implementation of TPM 2.0 in the PCH. TPM 2.0 features within a platform root of trust, remote attestation and PCR capabilities and security-enabled storage hierarchy. Intel PTT supports platform identification and attestation, along with security-enabled import of customer keys into the key hierarchy rooted in Intel PTT. A SW application that owns a key in Intel PTT storage hierarchy can request Intel PTT to transfer the key to the Intel QAT hardware over the internal sideband link.
1. **Key Generation and Storage:** A user’s Symmetric Wrapping Key (SWK) is first generated and stored in an external key server, such as an HSM appliance or KMIP server. The key server is also responsible for related key management functions such as backup, restoration, revocation, audits, and logs. The SWK is used to wrap user’s clear private key (K) to generate a wrapped private key (SWK(K), or WPK). The WPK is stored with the application’s image.

2. **Secure Key Provisioning:** This is the process by which a key is transferred more securely from the external key server to the platform Intel PTT’s key hierarchy using TPM 2.0’s key import protocol.

3. **Launch Time Key Sharing:** At the request of the owning application, as symmetric wrapping key (SWK, or a Key Encryption Key (KEK)) is transferred from Intel PTT to Intel QAT over an internal hardware link. In response, the application is provided with a dynamic handle value, called ‘KptHandle.’

4. **Runtime Usage:** The application performs Intel QAT private key requests through an API call that uses KptHandle and WPK as the parameters. (There will not be a clear private key provided in the API). QAT device and firmware, when processing the API, use KptHandle to locate the corresponding SWK from internal memory, unwrap the WPK to retrieve the CPK and perform the requested private key operation. The clear private key is discarded at the conclusion of the crypto request.

---

**At Aspen Systems our mission is to provide leading best-in-class solutions to High Performance Computing (HPC) users and administrators. We are proud to service all key market sectors including government, universities, corporations and anywhere research and science can be found. ISO 9001:2008 Certified. GSA Contract GS-35F-0192K.**
Applications can leverage Intel® KPT capability in two different ways – either by interfacing directly with the Intel® QAT API, or through a popular crypto framework like OpenSSL* or PKCS#11 together with an adaptation layer ("shim") between the framework and Intel QAT API.

**Software Architecture Overview**

Applications can leverage Intel® KPT capability in two different ways – either by interfacing directly with the Intel® QAT API, or through a popular crypto framework like OpenSSL*, or PKCS#11 together with an adaptation layer (“shim”) between the framework and Intel QAT API.

*Figure 3* illustrates how native Intel® QuickAssist Technology applications can incorporate Intel® KPT in their usage.

At launch time, the application accesses its keys from Intel® PTT’s storage hierarchy, by presenting its authorization credentials (such as a password, for example). The application requests Intel® PTT to transfer its symmetric wrapping key, (previously stored in Intel® PTT), to Intel® QuickAssist Technology hardware device. When the transfer is complete, an application receives a session-specific ‘KptHandle’ as a proxy to its SWK. At run time, when requesting private key operations from Intel® QuickAssist Technology device, it passes KptHandle and the wrapped private key instead of the original private key. Intel® QAT device, when processing the request for a private key operation, first uses the KptHandle to lookup the corresponding SWK, unwraps the WPK to derive the CPK and then performs the requested private key operation.

*Figure 4* illustrates how applications that natively interface with an existing cryptographic API framework can “seamlessly” integrate with Intel® QuickAssist Technology and Intel® KPT Technology. With the help of a remote key server, such as a KMIP server or an HSM, this solution can scale well to an enterprise server farm, as for example a web server.

An Intel® KPT adaptation layer (or “shim”) intercepts key management and crypto service APIs from the application. With key management requests, such as loading of a private key, the adaptation layer connects to the remote key server and requests it to provision a Symmetric Wrapping Key and a Wrapped Private Key owned by the application. The Intel® KPT adaptation layer transparently manages the activities of:
- Accessing CPK from the key server
- Requesting an SWK from the key server
- Requesting key server to provide a WPK
- More securely transferring the SWK to the platform Intel® PTT
- Caching the WPK within
- Transferring SWK to Intel® QAT and obtaining a KptHandle in lieu
- Maintaining an internal map between the KptHandle, the WPK and the application’s original reference to its private key in the key server.

At run time, the application requests private key operations using references to its clear private key in the key server. The adaptation layers intercept this request, maps this reference to the corresponding KptHandle and WPK, and translates this request into a corresponding Intel® QuickAssist Technology request that uses KptHandle and a WPK. Responses from Intel® QuickAssist Technology device at the end of the crypto operations routed back to the application through the native API (e.g., OpenSSL*).
There are significant advantages to using an Intel KPT adaptation layer as shown in Figure 3. Enabling Intel KPT with Intel® QuickAssist Technology API on page 4:

- Applications can take advantage of Intel QAT and Intel KPT without modifying their sources, nor their relationship with the key server.
- Applications use Intel QuickAssist Technology with Key Protection Technology without dealing with the specific of Intel QAT, Intel PTT, Intel QAT API, TSS API, provisioning of SWK into Intel PTT and the launch time transfer of SWK from Intel PTT to Intel QAT.
- Using a remote key server, the solution scales well to a large number of servers and applications.
Conclusion

As data encryption becomes pervasive in the cloud, data center, Internet of Things and mobile devices, protecting keys in hardware is necessary. Intel KPT complements Intel QAT high throughput crypto accelerator with hardware-based key management. Together, Intel QAT and Intel KPT lower the cost and effort barriers to the ubiquitous adoption of encryption and key management – a usage model that has become central to cloud services and CDNs.

Complexity in deployment and usage tend to slow down the adoption of new security technologies. Adoption of Intel KPT will be straightforward and seamless when integrated with known crypto and key management use models centered around HSMs. By presenting Intel KPT through the model of a “platform HSM” accessible through legacy crypto API frameworks such as PKCS#11 and OpenSSL®, the user benefits from the security and performance of Intel QAT with Intel KPT, while being transparent to the details of underlying platform technologies, such as Intel QAT and Intel PTT.
<table>
<thead>
<tr>
<th><strong>Table 1. Terminology</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Intel® QAT</strong></td>
</tr>
<tr>
<td><strong>Intel® KPT</strong></td>
</tr>
<tr>
<td><strong>Lookaside Acceleration</strong></td>
</tr>
<tr>
<td><strong>PCH</strong></td>
</tr>
<tr>
<td><strong>TPM</strong></td>
</tr>
<tr>
<td><strong>Storage Hierarchy</strong></td>
</tr>
<tr>
<td><strong>Intel® PTT</strong></td>
</tr>
<tr>
<td><strong>Clear Private Key</strong></td>
</tr>
<tr>
<td><strong>Symmetric Wrapping Key</strong></td>
</tr>
</tbody>
</table>