

REMOTE ATTESTATION

Overall View of Intel SGX Infrastructure Services

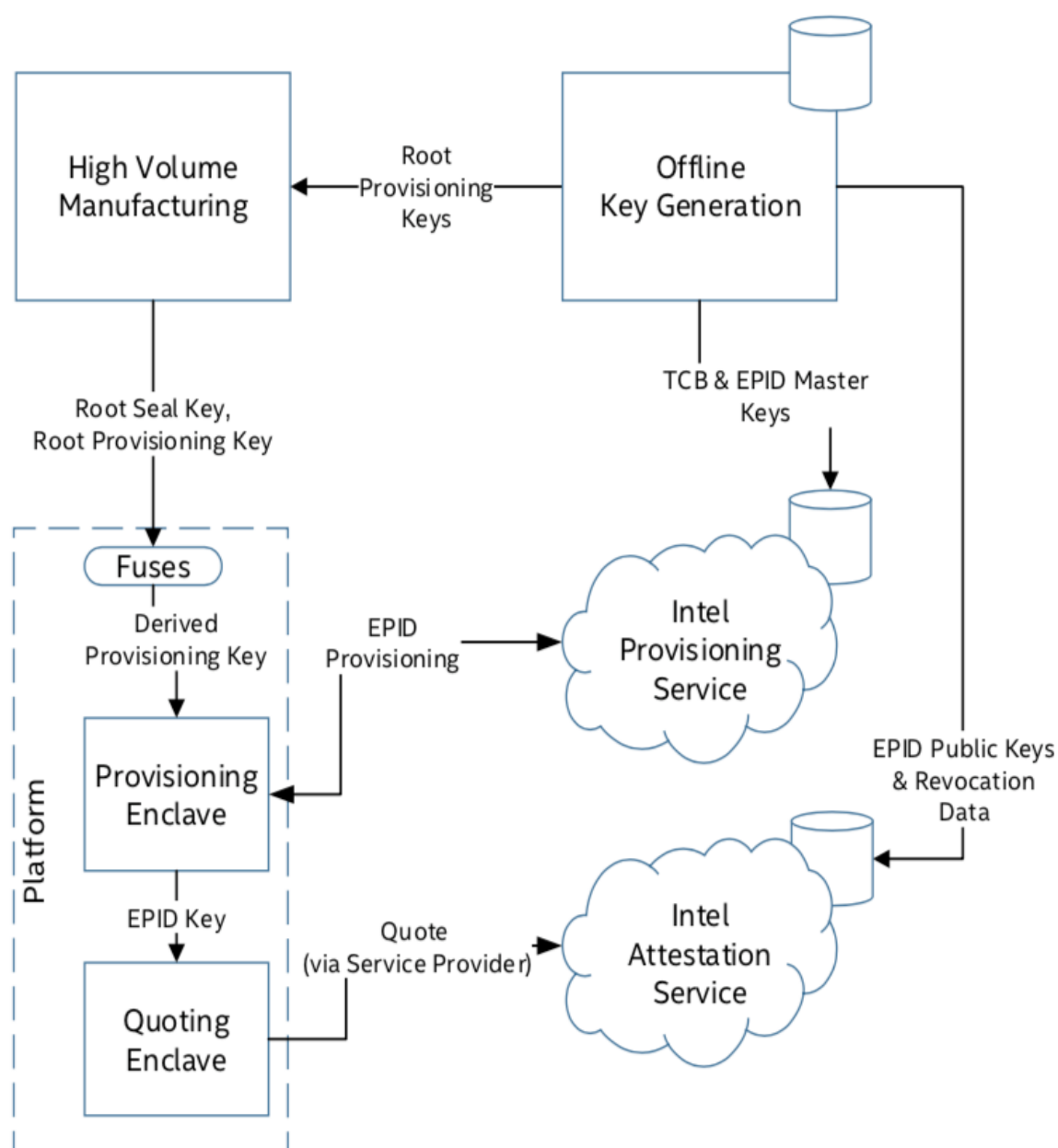


Figure 6: SGX infrastructure Services

(<http://www.sgx101.com/wp-content/uploads/2017/09/Screen-Shot-2018-06-30-at-5.51.39-PM.png>)

Platform Provisioning

In order to transform a local REPORT into a remotely verifiable QUOTE, Quoting Enclave uses a platform unique asymmetric attestation key. The QUOTE can then be verified by a remote party using the corresponding public key.

So how does QE obtain this attestation key in the first place? In this tutorial we explain the provisioning process in which an SGX platform receives its remote attestation key.

Provisioning is the process by which an SGX device demonstrates to Intel its authenticity as well as its CPU SVN and other system components attributes, in order to receive an appropriate attestation key reflecting its SGX genuinely and TCB version. Normally, provisioning is done during platform initial setup phase, but re-provisioning can also be performed after purchase due to update to crucial system components such as firmware, BIOS or microcodes due to vulnerabilities. In such cases, the attestation key may be replaced to reflect platform renewed TCB security level.

Attestation key is the core asset in the SGX ecosystem. Relying parties trust valid attestation signatures as an Intel signed certificate that guarantees the platform's authenticity. In order to facilitate SGX provisioning services, Intel operates a dedicated online provisioning infrastructure. SGX provisioning and remote attestation protocol follows a group signature scheme developed by Intel called Enhanced Privacy ID (EPID). To implement the EPID provisioning process Intel provides an architectural enclave called the Provisioning Enclave (PvE).

Provisioning Enclave (PvE)

The PvE is responsible for conducting the provisioning process on the platform against Intel's online provisioning servers. In this process PvE demonstrates that it has a key that Intel put in a real SGX processor and in return, is provisioned with a unique platform attestation for future remote attestations. Both sides implement the EPID scheme join protocol; the PvE functions as a new joining member and Intel as the group membership issuer that issues new group membership credentials.

PvE proves its authenticity by using several SGX privileged key types which are accessible through EGETKEY instruction only by SGX architectural enclaves. Two of those keys are Provisioning Key (PK) and Provisioning Seal Key (PSK). The uniqueness of PvE and QE is based on their SIGSTRUCT certificates signed by Intel (MRSIGNER). Those enclaves are thus authorized to launch with privileged attributes in order to later obtain special keys by executing EGETKEY instruction.

Two phases are involved in the derivation process of PK. First, bind Root Provisioning Key to HW TCB. TCB key occurs during processors boot time by looping over PRF with the current platform SVN patch level which reflects platform's firmware components. Second, add SW properties to the resulting PK. It occurs when EGETKEY is called and uses the TCB key as basis for derivation. PvE's software elements are reflected by EGETKEY input parameters. Root Signing Key and Owner Epoch value are ignored in this case to render the same platform-specific key regardless of its current owner. The resulting PK is then a unique key that reflects both HW and SW components of the SGX platform. This process also minimizes the exposure of Root Provisioning Key.

Provisioning Protocol

After getting the PK, the platform can start the provisioning process to get the attestation key.

Enclave Hello

Once we have TCB specific PK, PvE generates two values to initiate the provisioning protocol. The first is a hash of the PK called Platform Provisioning ID (PPID). The second reflects the claimed TCB level based on current SVN. Both encrypted using IPS's public key and sent to IPS.

Server Challenge

Intel uses PPID to determine whether the platform has been previously provisioned. If so, an encrypted version of a previously generated attestation key is added to the server's challenge. If not, the server determines the EPID group for that platform, and adds the EPID group parameters together with a liveness nonce and a pre-computed TCB challenge to the message sent back to the platform.

Since all RPKs are stored by the offline Intel Key Generation Facility (iKGF), it can perform the same hardware and software TCB specific derivation process as performed by the PvE using EGETKEY (how to get SW attributes?) on every SGX device to produce its own provisioning key (how to know which SGX is which?). This PK is used to encrypt a random value to generate a platform specific TCB challenge. All pre-computed challenges are sent to Intel's online servers to support the provisioning protocol.

Enclave Response

After PvE decrypts the TCB challenge with its PK, it uses it to generate a TCB proof by using the TCB challenge as a key to CMAC the nonce received from Intel. Next, PvE generates a random EPID membership key and hides it mathematically according to

EPID protocol so that IPS cannot learn the membership key.

To facilitate future attestation key retrieval service, the non-hidden membership key is encrypted by PvE using another special key, PSK. PSK derivation does not include the Owner Epoch and uses RSK as the root key for derivation. The PSK thus is not affected by the platform changing owners, and is exclusive to that specific platform.

If the platform has been formerly provisioned and the ongoing protocol is an attestation key retrieval or TCB update, the platform has to prove that it has never been revoked in the past. This is achieved by using PSK to decrypt the backed up attestation key copies obtained from the server, and using them to sign a selected message chosen by Intel.

Both the hidden and the encrypted EPID membership keys are sent, together with the TCB and non-revoked proofs.

Completion

After receiving the response, IPS first validates the TCB proof using the value received from iKGF and continues the EPID Join protocol on success. The hidden membership key is processed to create a unique certificate signed with the EPID group issuer key and stored together with the encrypted membership key for future re-provisioning events. The final message completing the protocol is then sent by the server containing the signed certificate.

Platform's membership key together with the matching signed certificate form a unique EPID private key. Since the attestation key is constructed collaboratively by both parties, no one can forge a valid membership signature produced by the platform.

Final

PvE encrypts the attestation key with PSK and stores on the platform for future use. Since EPID groups are categorized according to TCB levels, EPID signature can thus be used to represent both platform's SGX genuineness and its TCB level.

Remote Attestation

Generally speaking, the goal of Remote Attestation is for a Hardware entity or a combination of Hardware and Software to gain the trust of a remote service provider, such that the service provider can confidently provide the client with the secrets requested. With Intel SGX, Remote Attestation software includes the application's enclave, and the Intel-provided Quoting Enclave (QE) and Provisioning Enclave (PvE). The attestation Hardware is the Intel SGX enabled CPU. Remote attestation provides

verification for three things: the application's identity, its intactness (that it has not been tampered with), and that it is running securely within an enclave on an Intel SGX enabled platform.

Some Concepts

Sigma protocol

Sigma is a protocol that includes a Diffie-Hellman key exchange, but also addresses the weaknesses of DH. The protocol Intel SGX uses differs from the regular Sigma protocol in that the Intel SGX platform uses Intel EPID to authenticate while the service provider uses Public Key Infrastructure (in regular Sigma, both parties use PKI). Finally, the Key Exchange libraries require the service provider to use an ECDSA, not an RSA, key pair in the authentication portion of the protocol and the libraries use ECDH for the actual key exchange.

As a result of this exchange between the client and the service provider, a shared key between the enclave and the challenger is produced that can be used for encrypting secrets that are to be provisioned in the enclave. Once inside the enclave, these secrets could then be decrypted by the application.

Diffie-Hellman Key Exchange (DHKE)

A method for exchanging keys over a public channel without leaking the actual key to other listeners. The cryptographic algorithm is explained here (<https://security.stackexchange.com/questions/45963/diffie-hellman-key-exchange-in-plain-english>).

Intel Enhanced Privacy ID (EPID)

It is an extension to an existing Direct Anonymous Attestation (https://en.wikipedia.org/wiki/Direct_Anonymous_Attestation) (DAA) scheme with some additions, for example the use of SigRL (Signature Revocation List). EPID enables signing objects without leaving a trace that can be uniquely backtracked to the signer, making the signing process anonymous. This is done by dividing signers to groups (also known as EPID groups), based on their processor type. This way they create signatures with their own secret keys, but the signatures can be verified only with the public key of the group they belong to, making it possible to check that the signer belongs to the right group, but impossible to uniquely identify the signer.

Revocation Lists

SGX facilitates three types of Revocation Lists (RLs): Group-RL which holds all revoked EPID groups, Priv-RL listing all revoked private-keys of the same EPID group, and Sig-RL that lists tuples of a basename and its corresponding signature of all revoked

members in the same EPID group.

Trusted Computing Base (TCB)

An entity responsible for protecting the secret provisioned to the enclave (both software and hardware).

Quoting Enclave (QE)

A special enclave on every SGX processor and is tasked entirely with handling the remote attestation. It receives REPORTs from other enclaves, verifies them and signs them with the attestation key before returning the result, also known as a QUOTE, to the application.

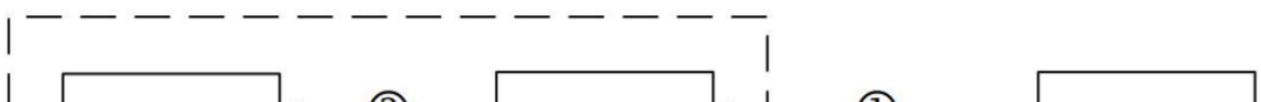
SGX Service Providers

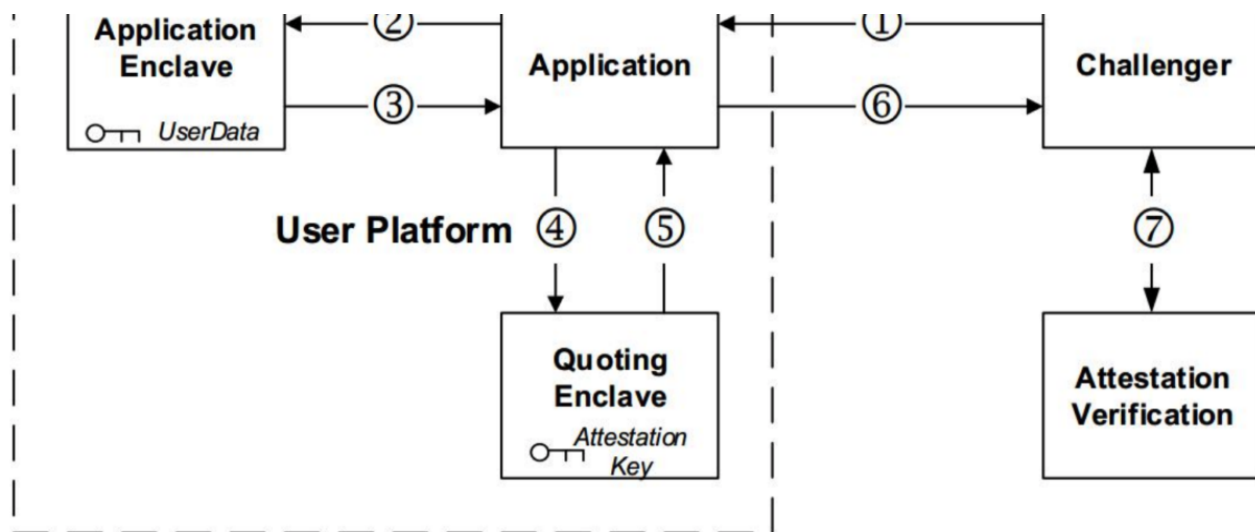
Relying parties are referred to as service providers and do not have to hold SGX enabled hardware. Service providers are expected to register to the IAS and meet a set of Intel defined requirements in order to submit attestation evidence for IAS verification. This registration binds service providers' Transport Layer Security (TLS) certificate to a unique Service Provider ID (SPID), and permits access to the IAS services. Some of these main IAS services are: verifying ISV enclave Quotes, requesting updated attestation revocation lists and retrieving the assertion information history associated with a Quote.

Remote Attestation Modes

The QE supports two Quote signature modes with different link-ability properties, Fully-anonymous and Pseudonymous Quotes. The link-ability property of a Quote is determined by a *basename* parameter signed using platform's unique attestation key. Using the same attestation key to sign the same *basename* parameter multiple times yields pseudonymous Quotes that are easily linkable. This mode is used by service providers to keep track of revisiting users and protect against sybil attacks, while preserving user's privacy. When a pseudonymous Quote is used, the IAS first validates that the *basename* used is associated to that specific service provider. This role of the IAS enforces user's pseudonymous separation between different service providers. In contrast, by signing multiple signatures on different *basenames*, it is computationally infeasible to determine whether the Quotes were produced using the same attestation key or not, thus preserving platform's anonymity. Therefore random *basenames* are used by the QE to sign Fully-anonymous Quotes.

Remote Attestation Abstract





(<http://www.sgx101.com/wp-content/uploads/2017/09/Screen-Shot-2018-06-28-at-2.15.08-PM.png>)

For remote attestation, both symmetric and asymmetric key systems are used. The symmetric key system is used in local attestation with only the quoting enclave and the EREPORT instruction having access to the authentication key. Asymmetric key system is used for creating an attestation that can be verified from other platforms. The attestation key itself is asymmetric (EPID keys).

There are mainly three platforms involved in Remote Attestation:

- The service provide (challenger)
- The application with its enclave and its QE
- Intel Attestation Service (IAS) that verifies the enclave

Stages

1. At first, the ISV enclave sends out an initial request to the remote service provider, which includes the EPID group the platform claims to currently be a member of.
2. If the service provider wishes to serve members of the claimed group, it may proceed by requesting an updated SigRL from the IAS.
3. The service provider then constructs a challenge message that consists its SPID, a liveness random nonce, the updated SigRL and an optional *basename* parameter (if a pseudonym signature is required).
4. If the enclave supports the requested signature mode, it invokes the EREPORT instruction to create a locally verifiable report addressed to platform's QE. In order to establish an authenticated secure channel between the enclave and the service provider, a freshly generated ephemeral public key may be added to the report's user data field. The report and SP's challenge are sent to QE.
5. (*Local Attestation happens here*) The QE calls EGETKEY to obtain the REPORT KEY and verifies the report. If successful, QE calls EGETKEY again to receive platform's Provisioning Seal Key to decrypt platform's remote attestation key (EPID private key).

The attestation key is first used to produce an identity signature by either signing the challenged *basename* or a random value, according to the attestation mode requested.

6. The attestation key is then used to compute two signatures of knowledge over the platform's identity signature (MRENCLAVE). The first proves the identity signature was signed with a key certified by Intel. The second is a non-revoked proof that proves the key used for the identity signature does not create any of the identity signatures listed in the challenged SigRL. A final QUOTE is then generated and encrypted using IAS's public key, which is hardcoded in QE, and the result is sent back to the attesting enclave. The QUOTE holds the identity of the attesting enclave, execution mode details (e.g. SVN level) and additional data.

7. The enclave then forwards the QUOTE to the SP for verification.

8. Since the QUOTE is encrypted, it is verifiable exclusively by Intel. Hence, the service provider simply forwards the QUOTE to the IAS for verification.

9. The IAS examines the QUOTE by first validating its EPID proofs against its identity signature. It then verifies the platform is not listed on the group Priv-RL by computing an identity signature on the QUOTE *basename* for each private key in the list, and verifying that none of them equals the QUOTE's identity signature. This finalizes the validity check of the platform, and the IAS then creates a new attestation verification report as a response to the SP. The Attestation Verification Report includes the QUOTE structure generated by the platform for the attesting enclave.

10. A positive Attestation Verification Report confirms the enclave as running a particular piece of code on a genuine intel SGX processor. It is then the responsibility of the SP to validate the ISV enclave identity and serve an appropriate response back to the platform.


References:

1. https://courses.cs.ut.ee/MTAT.07.022/2017_spring/uploads/Main/hiie-report-s16-17.pdf (https://courses.cs.ut.ee/MTAT.07.022/2017_spring/uploads/Main/hiie-report-s16-17.pdf)
2. <https://software.intel.com/en-us/articles/code-sample-intel-software-guard-extensions-remote-attestation-end-to-end-example> (<https://software.intel.com/en-us/articles/code-sample-intel-software-guard-extensions-remote-attestation-end-to-end-example>)
3. <https://arxiv.org/pdf/1801.05863.pdf> (<https://arxiv.org/pdf/1801.05863.pdf>)
4. <https://www.idc.ac.il/en/schools/cs/research/Documents/jackson-msc-thesis.pdf> (<https://www.idc.ac.il/en/schools/cs/research/Documents/jackson-msc-thesis.pdf>)
5. <https://software.intel.com/en-us/articles/innovative-technology-for-cpu-based-attestation-and-sealing> (<https://software.intel.com/en-us/articles/innovative-technology-for-cpu-based-attestation-and-sealing>)

based-attestation-and-sealing)

6. <https://software.intel.com/en-us/forums/intel-software-guard-extensions-intel-sgx/topic/747718> (<https://software.intel.com/en-us/forums/intel-software-guard-extensions-intel-sgx/topic/747718>)
7. <https://software.intel.com/en-us/node/702983> (<https://software.intel.com/en-us/node/702983>)
8. <http://tce.webee.eedev.technion.ac.il/wp-content/uploads/sites/8/2015/10/SGX-for-Technion-TCE.pdf> (<http://tce.webee.eedev.technion.ac.il/wp-content/uploads/sites/8/2015/10/SGX-for-Technion-TCE.pdf>)
9. <https://eprint.iacr.org/2007/194.pdf> (<https://eprint.iacr.org/2007/194.pdf>)
10. <https://software.intel.com/en-us/articles/code-sample-intel-software-guard-extensions-remote-attestation-end-to-end-example> (<https://software.intel.com/en-us/articles/code-sample-intel-software-guard-extensions-remote-attestation-end-to-end-example>)

NEXT PROJECT  ([HTTP://WWW.SGX101.COM/PORTFOLIO/REMOTE_ATTESTATION_EXAMPLE/](http://www.sgx101.com/portfolio/remote-attestation-example/))

 **PREV PROJECT** ([HTTP://WWW.SGX101.COM/PORTFOLIO/LOCAL_ATTESTATION/](http://www.sgx101.com/portfolio/local-attestation/))

TWITTER **FACEBOOK**

Copyright © 2018 by SSLab (<https://gts3.org/>). All rights reserved.