



Intel® Data Center Attestation Primitives (DCAP): Appraisal Engine Developer Guide for Linux* OS

April, 2024

Table of Contents

| | |
|---|----|
| 1. Introduction | 3 |
| 2. Quick Start steps..... | 4 |
| 3. Detail Steps to Use the Appraisal Engine | 6 |
| 3.1 Install the Necessary Packages..... | 6 |
| 3.2 Generate appraisal policy..... | 7 |
| Platform policy..... | 8 |
| SGX Enclave Identity Policy..... | 25 |
| TD Identity Policy..... | 30 |
| 3.3 Calling the Quote Appraisal API | 32 |
| 3.4 Calling the Appraisal Audit API..... | 33 |
| 4. Quote Verification and Appraisal APIs | 34 |
| 5. Appendix..... | 42 |
| 5.1 Appraisal Result Token..... | 42 |
| 5.2 Data Structures..... | 45 |
| 6. Disclaimer and Legal Information..... | 46 |

1. Introduction

Appraisal Engine is one new feature of Intel® DCAP, this document guides you on how to use the Intel® DCAP Appraisal Engine for Software Guard Extensions (SGX) and Trust Domain Extensions (TDX) remote attestation. The target users are Linux developers and cloud DevOps.

Features:

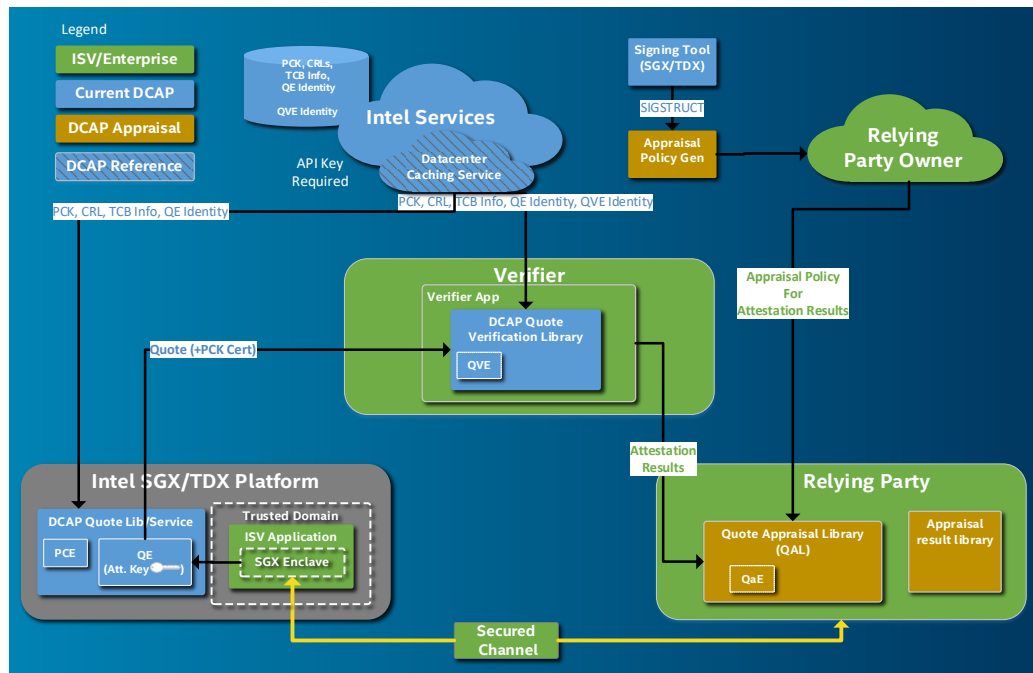
- Appraise SGX-based Quote.
- Appraise TDX-based Quote.
- Appraise SGX Enclave Identity.
- Appraise TD Identity.

Notes:

The current release of the Appraisal Engine does not execute in a SGX Enclave and does not require SGX. The platform should be protected against adversaries, or the Appraisal Engine can be run in a Confidential VM (e.g. TDX). Otherwise, a malicious user could modify the signing key, the verification result, or the appraisal result. Support for running the Appraisal Engine inside an SGX enclave as an option will be added in a future release.

2. Quick Start Steps

The following picture shows the set of DCAP components which are used in the attestation of a TDVMM running in the Trust Domain Extensions (TDX) TEE or an SGX enclave to a remote Relying Party.



This document assumes the reader is familiar with DCAP quote generation and quote verification process. The reader can refer to the DCAP documents at (download.01.org/intel-sgx/latest/dcap-latest/linux/docs/) for more information.

In the bare metal environment to use the Appraisal Engine, follow these steps:

1. Install the relative packages, including “Quote generation packages”, “Quoting verification library package”, “tee appraisal tool package”, “cache server (PCCS) package”, “multi-package agent package or PCKIDRetrieval tool package”, “PccsAdmintool in DCAP source code”
 - a. Install all the packages needed for end-to-end remote attestation.
 - b. Install the “tee appraisal tool package”.
2. Prepare the appraisal policies: including platform policy, user SGX enclave policy or user TD policy.
 - a. Intel has provided a tool (tee_appraisal_tool) to help user generate the policies. For user SGX enclave or TDVMM policy, this tool just generates the reference policy, user could modify the policy based on his/her workload’s requirement.
 - b. Then, upload the platform appraisal policies to your environment’s caching server with the PccsAdmintool. Uploading operation is one optional step, you can provide the policies directly when call the appraisal APIs.

3. Call the new appraisal APIs to enable the attestation appraisal functionality.
 - a. Refer to the Sample code:
QuoteAppraisalSample(<https://github.com/intel/SGXDataCenterAttestationPrimitives/tree/master/SampleCode/QuoteAppraisalSample>).
4. After finishing the appraisal of the attestation result, you can also audit the appraisal process.
 - a. In this release, from security perspective, audit the appraisal policy is a security requirement. You can refer to the Sample code:
(<https://github.com/intel/SGXDataCenterAttestationPrimitives/tree/master/SampleCode/QuoteAppraisalSample>)

In the cloud environment to use the Appraisal Engine, please refer to the Cloud Service Provider's documentation, typically need the following steps:

1. Install the relative packages, including "Quote generation packages", "Quoting verification library package", "tee appraisal tool package".
 - a. Install all the packages needed for end-to-end remote attestation.
 - b. Install the "tee appraisal tool package".
2. Prepare the appraisal policies: user SGX enclave policy or user TD policy.
 - a. Intel has provided a tool (tee_appraisal_tool) to help user generate the policies. For user SGX enclave or TDVM policy, this tool just generates the reference policy, user could modify the policy based on his/her workload's requirement.
 - b. In general, a platform owner (like a Cloud Service Provider) should own the platform policy's generation. As end users, you don't need to generate the platform policy.
3. Call the new appraisal APIs to enable the attestation appraisal functionality.
 - a. Refer to the Sample code:
QuoteAppraisalSample(<https://github.com/intel/SGXDataCenterAttestationPrimitives/tree/master/SampleCode/QuoteAppraisalSample>).
4. After finishing the appraisal of the attestation result, you can also audit the appraisal process.
 - a. In this release, from security perspective, audit the appraisal policy is a security requirement. You can refer to the Sample code:
(<https://github.com/intel/SGXDataCenterAttestationPrimitives/tree/master/SampleCode/QuoteAppraisalSample>)

3. Detail Steps to Use the Appraisal Engine

The Attestation Appraisal Engine is part of DCAP Attestation. Except the original DCAP software stack, a tool (`tee_appraisal_tool`) is provided for the appraisal functionality.

3.1 Install the Necessary Packages

First, install the necessary DCAP software stack into your platform including (you can refer to the doc: https://download.01.org/intel-sgx/latest/dcap-latest/linux/docs/Intel_SGX_SW_Installation_Guide_for_Linux.pdf):

1. Quote generation libraries
2. Quote Verification libraries
3. QCNL/QPL libraries
4. Cache server (PCCS)
5. Multi-package Agent or PCKIDRetrieval Tool
6. PccsAdminTool
7. If you are working on TDX remote attestation, you need to install TDX related packages:
 - a. Quote Generation Service
For rpm packages, you can install it as:

```
sudo yum install -y --setopt=install_weak_deps=False --nogpgcheck tdx-qgs
```


For Debian packages, you can install it as:

```
sudo apt install -y --no-install-recommends tdx-qgs
```
 - b. Ring 3 Attestation Abstraction Library
For rpm packages, you can install it as:

```
sudo yum install -y --setopt=install_weak_deps=False --nogpgcheck libtdx-attest  
sudo yum install -y --setopt=install_weak_deps=False --nogpgcheck libtdx-attest-devel
```


For Debian packages, you can install it as:

```
sudo apt install -y --no-install-recommends libtdx-attest  
sudo apt install -y --no-install-recommends libtdx-attest-dev
```

Notes:

If you use Appraisal Engine in cloud, please refer to Cloud Service Provider's documentation. Typically, Cloud Service Provider will take care item 4,5 and 6.

To prepare appraisal policy, you may need to install:

1. [tee_appraisal_tool](#)
For rpm packages, you can install it as:

```
sudo yum install -y --setopt=install_weak_deps=False --nogpgcheck tee-appraisal-tool
```


For Debian packages, you can install it as:

```
sudo apt install -y --no-install-recommends tee-appraisal-tool
```

Notes:

Repositories are supported to distribute packages for the following OSs(For how to setup the repo or local repo, you can refer to the documentation: https://download.01.org/intel-sgx/latest/dcap-latest/linux/docs/Intel_SGX_SW_Installation_Guide_for_Linux.pdf, in section “Installation Instructions-QuickStart” section) :

- Ubuntu and Debian: DCAP packages are provided in a Debian* repository at https://download.01.org/intel-sgx/sgx_repo/ and via a tar file located at <https://download.01.org/intel-sgx/latest/linux-latest/distro/>
- Red Hat, CentOS, Anolis, and SUSE: DCAP packages are provided as RPM Packages via a tar file located in the corresponding folder in <https://download.01.org/intel-sgx/latest/linux-latest/distro/>
 - a. Find RPM packages for SGX libraries and services, which are currently provided in a single TAR archive at <https://download.01.org/intel-sgx/latest/linux-latest/distro/>
 - b. Download the file `sgx_rpm_local_repo.tgz` to a selected folder, for example `/opt/intel cd /opt/intel sudo wget https://download.01.org/intel-sgx/latest/linux-latest/distro/<distro>/sgx_rpm_local_repo.tgz`
 - c. Verify the downloaded repo file with the SHA value in this file: `https://download.01.org/intel-sgx/latest/dcaplatest/linux/SHA256SUM_dcap_<version>.cfg`
 - d. Expand the archive: `sudo tar xvf sgx_rpm_local_repo.tgz` e. Add the RPM local repository to your local repository list `sudo yum-config-manager --add-repo file://PATH_TO_LOCAL_REPO`

3.2 Generate Appraisal Policy

A Relying Party needs appraisal policies to process the results of the quote verification. These include a policy to assess the platform and the trusted platform software generating the quote as well as a policy to assess the SGX Enclave or TD the Relying Party is evaluating.

DCAP now provides a tool to help user generate the appraisal policy: `tee_appraisal_tool`.

After installing this tool, its usage is described below :

`tee_appraisal_tool <Command> <options> <files>`

Command:

`gen_payload:` *Generate the policy payload file with JSON format from an SGX Enclave Report or a TDX Report.*

`sign_policy:` *Sign the input policy payload with the input EC key and generate the final policy file with JWT format.*

`verify_policy:` *Verify the JWT policy file.*

Options:

`-in` *Specify the input file path*

`-key` *Specify the key file. The key file must be a EC private key with 384 bytes
It is a required option for "sign_policy"*

`-out` *Speicify the output file path*

`-v` *Enable showing the extra dump message for each command*

Example:

```
tee_appraisal_tool gen_payload -in {enclave/TDReport} -out payload.json [-v]
tee_appraisal_tool sign_policy -in payload.json -key ec_private.pem -out policy.jwt [-v]
tee_appraisal_tool verify_policy -in policy.jwt [-v]
```

Notes:

1. “gen_payload” subcommand could only be used to generate SGX enclave or TDVM payload as SGX enclave or TDVM policy.
2. Subcommand “verify_policy” could be used to verify the policy that is generated by subcommand “sign_policy”.

Platform Policy

To generate platform policy, you can refer to the following detail information (sample policies for [SGX Platform policy](#) and [TDX platform policy](#) or Sample policies in Sample code) to write your own policy based on your workload requirements. Then you can the following command to sign it:

```
tee_appraisal_tool sign_policy -in platform_policy.json -key ec_private.pem -out platform_policy.jwt [-v]
```

After you generated the platform policy, you can use the PccsAdmintool to upload the policy to your cache server:

```
pccsamdin put -u https://localhost:8081/sgx/certification/v4/appraisalpolicy [-d] -f fmSPC -i policy_file [-d] means default flag for the platform policy, each FMSPC should only has one default policy
```

Notes:

1. In general, a platform owner (like a CSP) should own the platform policy’s generation. As an end-user, if you don’t own the platform, you don’t need to generate the platform policy. But, you need to understand and accept the platform owner’s platform policy. Typically, the platform owner will publish their platform policy and the platform user can audit the platform policy.

SGX Platform Policy

To appraise the platform’s TCB, the Relying Party must generate a JSON Web Token (JWT) which contains a platform TCB appraisal policy with expected or accepted platform TCB claims. The platform TCB policy is in the JWT payload.

Definition

The Platform TCB Policy is a JSON object, containing a “environment” JSON object and a “reference” JSON object composed of fields detailed in the following table:

Environment fields and values:

- “class_id”: “3123ec35-8d38-4ea5-87a5-d6c48b567570

| Field name | Type | R/O | Field Description |
|--------------------------------|------------------|-----|---|
| accepted_tcb_status | Array of Strings | R | <p>List of TCB Status that are accepted.</p> <p>The Policy fails if the Platform TCB Report includes any values that are not provided in <code>accepted_tcb_status</code>.</p> <p>Note: a policy should never include “Revoked” in this list. This would be overriding the revoking of the TCB. In addition, a policy should include “UpToDate” which indicates that the platform TCB is current.</p> <p>When the policy accepts “OutOfDate”, indicating that the policy is allowing a platform with an out-of-date TCB level, then it should qualify or limit how the platform may be out-of date. There are three methods to do this:</p> <ol style="list-style-type: none"> 1. Provide a <code>platform_grace_period</code> to indicates the amount of time that platform is allowed to be out-of-date. 2. Provide a <code>min_tcb_date</code> which asserts that TCB updates after the <code>min_tcb_date</code> are not required. 3. Provide a list of <code>allowed_advisory_ids</code> which indicates that a platform with only the <code>allowed_advisory_ids</code> is accepted. <p>The following is a mapping between <code>quote_verification_result</code> and <code>tcb_status</code> array:</p> <ul style="list-style-type: none"> • <code>SGX_QL_QV_RESULT_OK</code>: “UpToDate” • <code>SGX_QL_QV_RESULT_SW_HARDENING_NEEDED</code>: “UpToDate”, “SWHardeningNeeded” • <code>SGX_QL_QV_RESULT_CONFIG_NEEDED</code>: “UpToDate”, “ConfigurationNeeded” • <code>SGX_QL_QV_RESULT_CONFIG_AND_SW_HARDENING_NEEDED</code>: “UpToDate”, “SWHardeningNeeded”, “ConfigurationNeeded” • <code>SGX_QL_QV_RESULT_OUT_OF_DATE</code>: “OutOfDate” • <code>SGX_QL_QV_RESULT_OUT_OF_DATE_CONFIG_NEEDED</code>: “OutOfDate”, “ConfigurationNeeded” • <code>SGX_QL_QV_RESULT_INVALID_SIGNATURE</code>: No Platform TCB Report Generated • <code>SGX_QL_QV_RESULT_REVOKED</code>: “Revoked” • <code>SGX_QL_QV_RESULT_UNSPECIFIED</code>: No Platform TCB Report Generated |
| collateral_grace_period | Number | R* | <p>Policy provides a collateral grace period which is a measurement in seconds that the collateral is allowed to be expired. The policy must provide this field. A value of</p> |

| | | | |
|------------------------------|--------|---|--|
| | | | <p>zero will indicate that the policy requires collateral which is up-to-date.</p> <p>If the Platform TCB Report.earliest_expiration_date + Policy.collateral_grace_period < current time, then appraisal fails.</p> <p>Either min_eval_num or collateral_grace_period is required as both of these parameters define whether the policy allows collateral used in quote verification to be expired:</p> <ol style="list-style-type: none"> 1. collateral_grace_period indicates how long the collateral has been expired. This should be 0 when only valid non-expired collateral is permitted. 2. Min_eval_num provides a monotonically increasing number tied to collateral released with each TCB update. |
| platform_grace_period | Number | 0 | <p>Policy provides a platform grace period which is a measurement in seconds.</p> <p>The Policy fails if the Platform TCB Report tcb_date + platform_grace_period < current time.</p> <p>Providing a non-zero platform_grace_period implies that the policy is willing to accept a tcb_status which reports "OutOfDate"; thus, "OutOfDate" should be provided in the accepted_tcb_status array.</p> |
| min_tcb_date | String | 0 | <p>Policy may provide a time value asserting that TCB updates after this date/time are not required. The time value is UTC with the encoding compliant to ISO 8601 standard (YYYY-MM-DDThh:mm:ssZ).</p> <p>The Policy fails if the Platform TCB Report tcb_date value is before the Policy min_tcb_date, even if TCB status value "OutOfDate" is accepted.</p> <p>Essentially, this means that, if the platform tcb_date is equal to or newer than the policy min_tcb_date, the policy can accept the OutOfDate TCB status.</p> <p>A policy using this field will accept any patching level enforced by Intel on or after this date. This differs from the platform_grace_period in that the grace period is an allowance for a relative time (based on current time) related to the patching update, where min_tcb_date is an</p> |

| | | | |
|---------------------------------------|------------------|----|---|
| | | | absolute time related to the patch update which is not related to current time. MWS add an example for this |
| min_pck_crl_num | Number | O | Policy fails if this field is present, and the report.pck_crl_num has a value lower than the policy value. |
| min_root_ca_crl_num | Number | O | Policy fails if this field is present, and the report.root_ca_crl_num has a value lower than the policy value |
| min_eval_num | Number | R* | Policy may provide a min_eval_num indicating that Platform TCB Reports with higher tcb_eval_num are not required. The Policy fails if the Platform TCB Report includes an tcb_eval_num which is lower than the Policy min_eval_num. Either min_eval_num or collateral_grace_period is required. See note in collateral_grace_period. |
| accepted_platform_provider_ids | String | O | Policy may provide an Array of accepted platform_provider_ids. The Policy fails if the Policy provides a list of platformProvIDs and the Platform TCB Report platformProvID does not match one of the Policy acceptedPlatformProvIDs |
| accepted_sgx_types | Array of Numbers | O | Policy may provide a listed of accepted SGX type values as defined in sgx_qi_qv_supplemental_t.sgx_type. If this field is present, the policy fails if the report.sgx_type value is not one of the values listed in this field. |
| allow_dynamic_platform | bool | O | Indicates whether a platform can be extended with additional packages. This value is provided in sgx_qi_qv_supplemental_t.dynamic_platform. The policy fails if is_dynamic_platform is true and the policy allow_dynamic_platform is false. Note: this field is only evaluated if sgx_type is set to either 1 for “scalable” or 2 for “Scalable with Integrity”. This field should not be provided in the policy if accepted_sgx_types does not contain either a 1 or a 2. |
| allow_cached_keys | bool | O | Indicates whether platform root keys are cached by SGX Registration Backend. This value is provided in sgx_qi_qv_supplemental_t.cached_keys. |

| | | | |
|------------------------------|------------------|---|---|
| | | | The policy fails if <code>is_cached_keys</code> is true and the <code>policy.allow_cached_keys</code> is false. Note: this field is only evaluated if <code>sgx_type</code> is set to either 1 for “scalable” or 2 for “Scalable with Integrity”. This field should not be provided in the policy if <code>accepted_sgx_types</code> does not contain either a 1 or a 2. |
| allow_smt_enabled | bool | O | Indicates whether a platform has SMT (simultaneous multithreading) enabled. This value is provided in <code>sgx_qi_qv_supplemental_t.smt_enabled</code> . The policy fails if <code>is_smt_enabled</code> is true and the policy <code>allow_smt_enabled</code> is false. Note: this field is only evaluated if <code>sgx_type</code> is set to either 1 for “scalable” or 2 for “Scalable with Integrity”. This field should not be provided in the policy if <code>accepted_sgx_types</code> does not contain either a 1 or a 2. |
| rejected_advisory_ids | Array of Strings | O | List of advisories against the platform which are not allowed. If the Platform TCB Policy provides a list of <code>rejected_advisory_ids</code> and any of the <code>rejected_advisory_ids</code> match an <code>advisory_id</code> provided in the Platform TCB Report, then the appraisal fails |
| allowed_root_key_ids | Array of Strings | O | Policy may provide an array of <code>allowed_root_key_ids</code> . The Policy fails if the Policy provides a list of <code>allowed_root_key_ids</code> and the Platform TCB Report <code>root_key_id</code> does not match one of the Policy entries. |

Table 1: Platform TCB Policy Fields

Notes:

In this table, “R” means this field is required field; “O” means this field is optional field.

Example Policies

Different relying parties could require different level of platform TCB policies. Below are some of the levels that are commonly used:

- **Strict**
 - a) `accepted_tcb_status` includes only UpToDate
 - b) `collateral_grace_period` = 0

- **Platform grace period:**
Intel has published new TCB, and the collaterals have been upgraded in caching service, but the user still needs a grace period to update platform
 - a) `accepted_tcb_status` must includes `UpToDate` and `OutOfDate`
 - b) `collateral_grace_period` = 0
 - c) `platform_grace_period` indicates the time in seconds that the platform may be out-of-date based on
 - `report_tcb_date + platform_grace_period >= current_time`
 - `tcb_date = min(TCBInfo.TCBLevel.tcbDate, QEIdentity.TCBLevel.tcbDate)`
- **Collateral grace period:**
Intel has published new TCB, but collaterals cannot be upgraded in caching service due to limitations.
 - a) `accepted_tcb_status` must includes `UpToDate`
 - b) `collateral_grace_period` = time in seconds that the collateral is allowed beyond the expiration time where
 - `report_earliest_expiration_date + collateral_grace_period >= current_time`
 - c) `platform_grace_perokd` must not present in current policy
- **Platform with required patching to a specific TCB level:** Platform has patching applied according to a particular TCB Recovery Event
 - a) `accepted_tcb_status` includes 'UpToDate'
 - b) `tcb_eval_num >= min_eval_num`
 - It will allow expired collateral if this is true.
 - c) `platform_grace_period` must not present in current policy
 - d) `collateral_grace_period` must not present in current policy
- **Platform without rejected security advisories.**
Allow 'out-of-date' if the TCB level's Advisory-ID List does not contain any SA(s) listed in the policy AND no collateral is expired.

For “**platform grace period**”, “**collateral grace period**” and “**platform with required patching to a specific TCB level(min_eval_num)**”, You can refer to the following table to write your policy.

| platform_grace_period | collateral_grace_period | min_eval_num |
|---|---|---|
| accepted_tcb_status must include UpToDate and OutOfDate | accepted_tcb_status must include UpToDate | accepted_tcb_status must include UpToDate |
| collateral_grace_period==0 | platform_grace_period must not be present | platform_grace_period must not be present |
| min_eval_num must not be present | min_eval_num must not be present | collateral_grace_period must not be present |

Example “Strict” SGX Platform policy:

```
{
  "policy_array": [
    {
      "environment": {
        "class_id": "3123ec35-8d38-4ea5-87a5-d6c48b567570",
        "description": "Strict SGX Platform Policy"
      },
      "reference": {
        "#NOTE": "Replace the following configurations based on your own requirements",
        "allow_dynamic_plaform": true,
        "accepted_tcb_status": [
          "UpToDate"
        ],
        "collateral_grace_period": 0
      }
    }
  ]
}
```

Example “platform grace period” policy, with grace period of 120 days (10,368,000 seconds). This policy will also appraise on a platform which requires enclaves to be SW Hardened (i.e., the platform is allowed to report a tcb_status of “SWHardeningNeeded”).

platform_grace_period indicates the time in seconds that the platform may be out-of-date based on

- tcb_date = min(TCBInfo.TCBLevel.tcbDate, QEIdentity.TCBLevel.tcbDate)
- tcb_date + platform_grace_period >= current_time

```
{
  "policy_array": [
    {
      "environment": {
        "class_id": "3123ec35-8d38-4ea5-87a5-d6c48b567570",
        "description": "Grace Period Policy for SGX Platform"
      },
      "reference": {
        "#NOTE": "Replace the following configurations based on your own requirements",
        "allow_dynamic_platform": true,
        "#NOTE": "'accepted_tcb_status' must include 'UpToDate' and 'OutOfDate' if 'platform_grace_period' is defined",
        "accepted_tcb_status": [

```

```

        "UpToDate",
        "SWHardeningNeeded",
        "ConfigurationNeeded",
        "OutOfDate"
    ],
    "#NOTE": "'platform_grace_period' allows you to pass appraisal
verification even if the platform has expired within 120 days (10368000
seconds).",
    "platform_grace_period": 10368000,
    "#NOTE": "'collateral_grace_period' must be defined as 0 if
'platform_grace_period' is defined",
    "collateral_grace_period": 0
}
}
]
}

```

Example “collateral grace period” policy, with grace period of 90 days (7,776,000 seconds).
`collateral_grace_period` = time in seconds that the collateral is allowed beyond the expiration time
where “`report.earliest_expiration_date + collateral_grace_period >= current_time`”

```

{
  "policy_array": [
    {
      "environment": {
        "class_id": "3123ec35-8d38-4ea5-87a5-d6c48b567570",
        "description": "Collateral Grace Period Policy for SGX Platform"
      },
      "reference": {
        "#NOTE": "Replace the following configurations based on your own
requirements",
        "allow_dynamic_platform": true,
        "#NOTE": "'accepted_tcb_status' must include 'UpToDate' if
'collateral_grace_period' is defined.",
        "accepted_tcb_status": [
          "UpToDate",
          "SWHardeningNeeded",
          "ConfigurationNeeded",
          "OutOfDate"
        ],
        "#NOTE": "'collateral_grace_period' allows you to pass appraisal
verification even if the collateral has expired within 90 days (7776000
seconds).",

```

```

        "#NOTE": "If 'collateral_grace_period' is greater than 0,
'platform_grace_period' must not be defined",
        "collateral_grace_period": 7776000
    }
}
]
}

```

Example “platform with required patching” policy, requiring minimum eval number to be 5: It will allow expired collateral if “tcb_eval_num >= min_eval_num” is true.

```

{
  "policy_array":[
    {
      "environment": {
        "class_id": "3123ec35-8d38-4ea5-87a5-d6c48b567570",
        "description": "Mininum Evaluation Num Policy for SGX Platform"
      },
      "reference": {
        "#NOTE": "Replace the following configurations based on your own
requirements",
        "allow_dynamic_platform": true,
        "#NOTE": "'accepted_tcb_status' must include 'UpToDate' if
'min_eval_num' is defined.",
        "accepted_tcb_status": [
          "UpToDate",
          "SWHardeningNeeded",
          "ConfigurationNeeded",
          "OutOfDate"
        ],
        "#NOTE": "'min_eval_num' indicates that Platform TCB Reports with
higher tcb_eval_num are not required.",
        "#NOTE": "The Policy fails if the Platform TCB Report includes an
tcb_eval_num which is lower than the value defined in 'min_eval_num'.",
        "#NOTE": "If 'min_eval_num' is defined, 'platform_grace_period'
and 'collateral_grace_period' must not be defined",
        "min_eval_num": 5
      }
    }
  ]
}

```


Example “platform without rejected security advisories” policy, with “INTEL-SA-00078” as a rejected security advisory. This policy allows a grace period for the platform to be out-of-date unless the platform reports a specific security advisory:

```
{
  "policy_array":[
    {
      "environment": {
        "class_id": "3123ec35-8d38-4ea5-87a5-d6c48b567570",
        "description": "SGX Platform Policy with rejected security
advisories"
      },
      "reference": {
        "#NOTE": "Replace the following configurations based on your own
requirements",
        "allow_dynamic_platform": true,
        "accepted_tcb_status": [
          "UpToDate",
          "OutOfDate"
        ],
        "platform_grace_period": 0,
        "collateral_grace_period": 0,
        "rejected_advisory_ids": [
          "INTEL-SA-00617"
        ]
      }
    }
  ]
}
```

To generate platform policy, you can refer to the above sample policy to write your own policy based on your workload requirements. Then you can use this tool’s following command to sign it:

```
tee_appraisal_tool sign_policy -in platform_policy.json -key ec_private.pem -out
platform_policy.jwt [-v]
```

Notes:

1. Requirement for signing key: EC key, the key length is 384 bits and the key format is PEM .
2. In a production environment, you need to use your production signing key with the proper protections of its confidentiality. In a development or validation environment, you can generate your own ECDSA signing key pair for test. Run the below command to utilize OpenSSL along with the tool, <https://github.com/danedmunds/pem-to-jwk>:

```
$ openssl ecparam -name secp384r1 -genkey --noout > ec_priv.pem
```

TDX Platform Policy

To appraise the platform TCB, the Relying Party must generate a JSON Web Token (JWT) which contains a platform TCB appraisal policy with the expected or accepted platform TCB claims. The platform TCB policy is in the JWT payload.

Definition

The TDX platform TCB policy is like the SGX platform TCB policy, containing a “environment” JSON object and a “reference” JSON object composed of fields detailed in [Table 1: Platform TCB Policy Fields](#). But the class_id has been changed in the “environment” section:

- “class_id”
 - a) For TDX version 1.0: “9eec018b-7481-4b1c-8e1a-9f7c0c8c777f”
 - b) For TDX version 1.5: “f708b97f-0fb2-4e6b-8b03-8a5bcd1221d3”

| Field name | Type | Version | R/O | Field Description |
|--|---------|---------------------|---------|--|
| All fields in the SGX platform policy | Various | Version 1.0 and 1.5 | Various | Every field defined in the SGX platform policy has the same name, datatype, required/optional, values / appraisal algorithm in TDX platform policy |

We have merged the TD QE policy into the platform policy. A verified TD QE identity policy is a JSON object containing an “environment” section and a “reference” section with the fields identified in the table below.

Environment fields:

- “class_id”: “3769258c-75e6-4bc7-8d72-d2b0e224cad2”

| Field name | Type | R/O | Field Description |
|----------------------------|------------------|-----|---|
| accepted_tcb_status | Array of Strings | R | <p>List of TCB Status that are accepted. See Table 1: Platform TCB Policy Fields accepted_tcb_status for a definition of acceptable values</p> <p>The Policy fails if the report “tcb_status” field includes any values that are not provided in this field.</p> <p>The following is additional specific mapping between quote_verification_result and tcb_status array for TDX:</p> <ul style="list-style-type: none"> • TEE_QV_RESULT_TD_RELAUNCH_ADVISED: “TDRelaunchAdvised” |

| | | | |
|--------------------------------|------------------|---|--|
| | | | <ul style="list-style-type: none"> TEE_QV_RESULT_TD_RELAUNCH_ADVISED_CONFIG_NEEDED: "TDRelaunchAdvised", "ConfigurationNeeded" |
| collateral_grace_period | Number | O | <p>Policy provides a collateral grace period which is a measurement in seconds</p> <p>If the report.expiration_date + policy.collateral_grace_period < current time, then appraisal fails.</p> |
| platform_grace_period | Number | O | <p>Policy provides a platform grace period which is a measurement in seconds</p> <p>The Policy fails if the report.tcb_date + policy.platform_grace_period < current time.</p> |
| min_tcb_date | String | O | <p>Policy may provide a time value asserting that TCB updates after this date/time are not required. The time value is UTC with the encoding compliant to ISO 8601 standard (YYYY-MM-DDThh:mm:ssZ).</p> <p>The Policy fails if the report.tcb_date value is before the policy.min_tcb_date, even if TCB status value "OutOfDate" is accepted.</p> <p>Note: a policy using this field will accept any patching level on or after this date. This differs from the platform_grace_period in that the grace period is an allowance for a relative time related to the patching update which will expire., where min_tcb_date is an absolute time related to the patch update which does not expire.</p> |
| min_eval_num | Number | O | <p>Policy may provide a min_eval_num indicating that TDX platform reports with higher tcb_eval_num are not required.</p> <p>The Policy fails if the report.tcb_eval_num is lower than policy.min_eval_num.</p> |
| allowed_root_key_ids | Array of Strings | O | <p>Policy may provide an array of allowed root_key_ids.</p> <p>The Policy fails if the Policy provides a list of allowed root_key_ids and the Verified QE Identity Report root_key_id does not match one of the Policy entries.</p> |

Example Policies

Example "Strict" TDX Platform policy:

```
{
  "policy_array": [
    {
      "environment": {
        "class_id": "f708b97f-0fb2-4e6b-8b03-8a5bcd1221d3",
```

```

        "description": "Strict TDX platform policy"
    },
    "reference": {
        "#NOTE": "Replace the following configurations based on your own
requirements",
        "allow_dynamic_plaform": true,
        "accepted_tcb_status": [
            "UpToDate"
        ],
        "collateral_grace_period": 0
    }
},
{
    "environment": {
        "class_id": "3769258c-75e6-4bc7-8d72-d2b0e224cad2",
        "description": "Strict Verified TDQE Identity policy"
    },
    "reference": {
        "accepted_tcb_status": [
            "UpToDate"
        ],
        "collateral_grace_period": 0
    }
}
]
}

```

Example “platform grace period” policy, with grace period of 120 days (10,368,000 seconds).
platform_grace_period indicates the time in seconds that the platform may be out-of-date based on

- tcb_date = min(TCBInfo.TCBLevel.tcbDate, QEIdentity.TCBLevel.tcbDate)
- tcb_date + platform_grace_period >= current_time

```

{
    "policy_array": [
        {
            "environment": {
                "class_id": "f708b97f-0fb2-4e6b-8b03-8a5bcd1221d3",
                "description": "Grace Period Policy for TDX Platform"
            },
            "reference": {
                "#NOTE": "Replace the following configurations based on your own
requirements",
                "allow_dynamic_plaform": true,

```

```

        "#NOTE": "'accepted_tcb_status' must include 'UpToDate' and
'OutOfDate' if 'platform_grace_period' is defined",
        "accepted_tcb_status": [
            "UpToDate",
            "TDRelaunchAdvised",
            "SWHardeningNeeded",
            "ConfigurationNeeded",
            "OutOfDate"
        ],
        "#NOTE": "'platform_grace_period' allows you to pass appraisal
verification even if the platform has expired within 120 days (10368000
seconds).",
        "platform_grace_period": 10368000,
        "#NOTE": "'collateral_grace_period' must be defined as 0 if
'platform_grace_period' is defined",
        "collateral_grace_period": 0
    }
},
{
    "environment": {
        "class_id": "3769258c-75e6-4bc7-8d72-d2b0e224cad2",
        "description": "Grace Period Policy for Verified TDQE"
    },
    "reference": {
        "#NOTE": "'accepted_tcb_status' must include 'UpToDate' and
'OutOfDate' if 'platform_grace_period' is defined",
        "accepted_tcb_status": [
            "UpToDate",
            "OutOfDate"
        ],
        "#NOTE": "'platform_grace_period' allows you to pass appraisal
verification even if the platform has expired within 60 days (5184000
seconds).",
        "platform_grace_period": 5184000,
        "#NOTE": "'collateral_grace_period' must be defined as 0 if
'platform_grace_period' is defined",
        "collateral_grace_period": 0
    }
}
]
}

```

Example “collateral grace period” policy, with grace period of 90 days (7,776,000 seconds)
 collateral_grace_period = time in seconds that the collateral is allowed beyond the expiration time
 where “report.earliest_expiration_date + collateral_grace_period >= current_time”

```
{
  "policy_array": [
    {
      "environment": {
        "class_id": "f708b97f-0fb2-4e6b-8b03-8a5bcd1221d3",
        "description": "Collateral Grace Period Policy for TDX Platform"
      },
      "reference": {
        "#NOTE": "Replace the following configurations based on your own requirements",
        "allow_dynamic_plaform": true,
        "#NOTE": "'accepted_tcb_status' must include 'UpToDate' if 'collateral_grace_period' is defined.",
        "accepted_tcb_status": [
          "UpToDate",
          "TDRelaunchAdvised",
          "SWHardeningNeeded",
          "ConfigurationNeeded",
          "OutOfDate"
        ],
        "#NOTE": "'collateral_grace_period' allows you to pass appraisal verification even if the collateral has expired within 90 days (7776000 seconds).",
        "#NOTE": "If 'collateral_grace_period' is greater than 0, 'platform_grace_period' must not be defined",
        "collateral_grace_period": 7776000
      }
    },
    {
      "environment": {
        "class_id": "3769258c-75e6-4bc7-8d72-d2b0e224cad2",
        "description": "Collateral Grace Period Policy for Verified TDQE"
      },
      "reference": {
        "#NOTE": "'accepted_tcb_status' must include 'UpToDate' if 'collateral_grace_period' is defined.",
        "accepted_tcb_status": [
          "UpToDate",
          "OutOfDate"
        ],
      }
    }
  ]
}
```

```

        "#NOTE": "'collateral_grace_period' allows you to pass appraisal
verification even if the collateral has expired within 60 days (5184000
seconds).",
        "#NOTE": "If 'collateral_grace_period' is greater than 0,
'platform_grace_period' must not be defined",
        "collateral_grace_period": 5184000
    }
}
]
}

```

Example “platform without rejected security advisories” policy, with “INTEL-SA-00617” as a rejected security advisory. This policy allows a grace period for the platform to be out-of-date unless the platform reports a specific security advisory:

```

{
  "policy_array": [
    {
      "environment": {
        "class_id": "f708b97f-0fb2-4e6b-8b03-8a5bcd1221d3",
        "description": "TDX Platform Policy with rejected security
advisories"
      },
      "reference": {
        "#NOTE": "Replace the following configurations based on your own
requirements",
        "allow_dynamic_plaform": true,
        "accepted_tcb_status": [
          "UpToDate",
          "OutOfDate"
        ],
        "rejected_advisory_ids": [
          "INTEL-SA-00617"
        ]
      }
    },
    {
      "environment": {
        "class_id": "3769258c-75e6-4bc7-8d72-d2b0e224cad2",
        "description": "Verified TDQE Identity policy with rejected
security advisories"
      },
      "allow_dynamic_plaform": true,
      "reference": {

```

```

        "accepted_tcb_status": [
            "UpToDate",
            "OutOfDate"
        ],
        "rejected_advisory_ids": [
            "INTEL-SA-00617"
        ]
    }
}
]
}

```

Example “platform with required patching” policy, requiring minimum eval number to be 5: It will allow expired collateral if “tcb_eval_num >= min_eval_num” is true.

```

{
  "policy_array": [
    {
      "environment": {
        "class_id": "f708b97f-0fb2-4e6b-8b03-8a5bcd1221d3",
        "description": "Mininum Evaluation Num Policy for TDX Platform"
      },
      "reference": {
        "#NOTE": "Replace the following configurations based on your own requirements",
        "allow_dynamic_plaform": true,
        "#NOTE": "'accepted_tcb_status' must include 'UpToDate' if 'min_eval_num' is defined.",
        "accepted_tcb_status": [
          "UpToDate",
          "TDRelaunchAdvised",
          "SWHardeningNeeded",
          "ConfigurationNeeded",
          "OutOfDate"
        ],
        "#NOTE": "'min_eval_num' indicates that Platform TCB Reports with higher tcb_eval_num are not required.",
        "#NOTE": "The Policy fails if the Platform TCB Report includes a tcb_eval_num which is lower than the value defined in 'min_eval_num'.",
        "#NOTE": "If 'min_eval_num' is defined, 'platform_grace_period' and 'collateral_grace_period' must not be defined",
        "min_eval_num": 5
      }
    },
  ],
}

```



```

    {
      "environment": {
        "class_id": "3769258c-75e6-4bc7-8d72-d2b0e224cad2",
        "description": "Minimum Evaluation Num Policy for Verified TDQE"
      },
      "reference": {
        "#NOTE": "'accepted_tcb_status' must include 'UpToDate' if
'min_eval_num' is defined.",
        "accepted_tcb_status": [
          "UpToDate",
          "OutOfDate"
        ],
        "#NOTE": "'min_eval_num' indicates that Platform TCB Reports with
higher tcb_eval_num are not required.",
        "#NOTE": "The Policy fails if the Platform TCB Report includes an
tcb_eval_num which is lower than the value defined in 'min_eval_num'.",
        "#NOTE": "If 'min_eval_num' is defined, 'platform_grace_period'
and 'collateral_grace_period' must not be defined",
        "min_eval_num": 5
      }
    }
  ]
}

```

SGX Enclave Identity Policy

To appraise the enclave, the Relying Party must generate and sign a JSON Web Token appraisal policy that contains reference measurements to compare against the enclave's Report to evaluate expected or accepted values.

To generate an enclave identity policy, you can use this tool's following command to generate the policy template:

```
tee_appraisal_tool gen_payload -in {enclave } -out payload.json [-v]
```

then you can edit the generated policy template based on your requirements, then you can sign the policy:

```
tee_appraisal_tool sign_policy -in payload.json -key ec_private.pem -out policy.jwt [-v]
```

This section details the Enclave Identity Policy and provides a few example policies.

Definition

The Enclave Identity Policy is a JSON object containing an “environment” JSON object and “reference” JSON object composed of fields detailed in Table 1: Platform TCB Policy Fields

. For each field, the definition includes how the measurements in the Enclave Identity report are evaluated by the field.

Environment fields and values:

- “class_id”: “bef7cb8c-31aa-42c1-854c-10db005d5c41”
- “description”: optional field that can be set with a descriptive text.

| Field name | Type | R/O | Field value and Appraisal Policy |
|--------------------------------|--------|-----|---|
| sgx_miscselect | String | O | Hex-encoded byte array (4 bytes) representing sgx_miscselect expected value (upon applying sgx_miscselect_mask). |
| sgx_miscselect_mask | String | O | Hex-encoded byte array (4 bytes) representing mask to be applied to miscselect value retrieved from the platform. <ul style="list-style-type: none"> REPORT.sgx_miscselect & POLICY.sgx_miscselect_mask == POLICY.sgx_miscselect |
| sgx_attributes | String | R | Hex-encoded byte array (16 bytes) representing attributes expected value (upon applying mask). This is required to confirm ATTRIBUTES.DEBUG Setting Note: the KSS bit in sgx_attributes and sgx_attributes_mask field must be set for the fields sgx_configid / sgx_configsvn / sgx_isvextprodid / sgx_isvfamilyid to be valid. |
| sgx_attributes_mask | String | R | Hex-encoded byte array (16 bytes) representing mask to be applied to attributes value retrieved from the platform. This is evaluated in the following way: <ul style="list-style-type: none"> REPORT.sgx_attributes & POLICY.sgx_attributes_mask == POLICY.sgx_attributes (Note: POLICY.sgx_attributes & POLICY.sgx_attributes_mask == POLICY.sgx_attributes) |
| sgx_cet_attributes | String | O | Hex-encoded byte (1 byte) representing CET Attributes expected value (upon applying mask). |
| sgx_cet_attributes_mask | String | O | Hex-encoded byte (1 byte) representing mask to be applied to CET Attributes value retrieved from the platform. <ul style="list-style-type: none"> REPORT.sgx_cet_attributes & POLICY.sgx_cet_attributes_mask == POLICY.sgx_cet_attributes |
| sgx_mrsigner | String | R* | Hex-encoded byte array (32 bytes) representing sgx_mrsigner hash. |
| sgx_mrenclave | String | R* | Hex-encoded byte array (32 bytes) representing a hash of the enclave build process (ENCLAVEHASH value from SIGSTRUCT). |
| sgx_isvprodid | Number | R* | ISV Product ID. |

| | | | |
|--------------------------|--------|----|--|
| sgx_isvsvn_min | Number | R* | Minimum ISV SVN. A valid policy should provide. The policy creator tool should only allow the following: <ul style="list-style-type: none"> • sgx_mrsigner, sgx_isvprodId, sgx_isvsvn_min : Identifies a specific enclave and a minimum SVN, but not a specific build of the enclave • sgx_mrenclave, sgx_mrsigner : identifies a specific build of a specific enclave, and the signer (this impacts sealing to mrsigner) • *sgx_mrenclave : we can allow this, but must warn that this impacts sealing to mrsigner and configuration of other features (miscselect, attributes, etc.) |
| sgx_configid | String | O | If provided, then REPORT.sgx_attributes.KSS must be 1 and then REPORT.sgx_configid == POLICY.sgx_configid (note: the appraisal service should select the Appraisal POLICY based on REPORT.sgx_configid) |
| sgx_configsvn_min | Number | O | If provided, then REPORT.sgx_attributes.KSS must be 1 and then REPORT.sgx_configsvn >= POLICY.sgx_configsvn_min (may need this if sgx_configid is used) |
| sgx_isvextprodid | String | O | If provided, then REPORT.attributes.KSS must be 1 and then REPORT.sgx_isvextprodid == POLICY.sgx_isvextprodid |
| sgx_isvfamilyid | String | O | If provided, then REPORT.sgx_attributes.KSS must be 1 and then REPORT.sgx_isvfamilyid == POLICY.sgx_isvfamilyid |

Table 2: Enclave Identity Policy Fields

Example Policies

Below is an example JSON object holding an example application enclave policy.

```
{
  "policy_array": [
    {
      "environment": {
        "class_id": "bef7cb8c-31aa-42c1-854c-10db005d5c41",
        "description": "Application SGX Enclave Policy"
      },
      "reference": {
        "#NOTE": "Replace the following configurations based on your own
enclave",
        "sgx_miscselect": "00000000",
        "sgx_miscselect_mask": "FFFFFFFF",

```

```

        "sgx_attributes": "00000000000000300000000000000005",
        "sgx_attributes_mask": "FFFFFFFFFFFFFFFF1BFF00FFFFFFFFFFFFD",
        "sgx_isvprodid": 0,
        "sgx_isvsvn_min": 0,
        "sgx_isvfamilyid": "00000000000000000000000000000000",
        "sgx_isvextprodid": "00000000000000000000000000000000",
        "#NOTE": "To make the following configurations take effect,
modify the value in the following lines, and uncomment the lines by removing '#'
from the key",
        "#sgx_mrenclave": "8845C3F4D658FF5C12DD2868F4D01DFE73471642C7141EF
5D80EFF551FFD299B",
        "#sgx_mrsigner": "258C20C86869C8AE080CD78D3543DD1E6A419DAB1E4A8332
5D4EEFC018FFE6E1"
    }
}
]
}

```

An enclave can either be identified by its mrenclave, or a mrsigner/isvprodid/ISVSVN tuple. It's not necessary to use all fields. In a production environment, an enclave shall always run in release mode (e.g. not in debug mode).

Below is an example of a policy which identifies an enclave by its signer, ISV Product ID and a minimum SVN. This policy also requires that the attributes debug bit, bit 1, is not set. This information can be obtained from the SGX Sign Tool when an enclave is signed.

```

{
  "policy_array": [
    {
      "environment": {
        "class_id": "bef7cb8c-31aa-42c1-854c-10db005d5c41",
        "description": "Application SGX Enclave Policy"
      },
      "reference": {
        "#NOTE": "Replace the following configurations based on your own
enclave",
        "sgx_mrsigner": "8845C3F4D658FF5C12DD2868F4D01DFE73471642C7141EF5D
80EFF551FFD299B",
        "sgx_isvprodid": 123,
        "sgx_isvsvn_min": 5,
        "sgx_attributes": "00000000000000300000000000000005",
        "sgx_attributes_mask": "FFFFFFFFFFFFFFFF1BFF00FFFFFFFFFFFFD"
      }
    }
  ]
}

```

```

    }
  }
]
}

```

Below is an example of a policy which just identifies an enclave by its mrenclave. This information can be obtained from the SGX Sign Tool when an enclave is signed.

```

{
  "policy_array": [
    {
      "environment": {
        "class_id": "bef7cb8c-31aa-42c1-854c-10db005d5c41",
        "description": "Application SGX Enclave Policy"
      },
      "reference": {
        "#NOTE": "Replace the following configurations based on your own
enclave",
        "sgx_mrenclave": "8845C3F4D658FF5C12DD2868F4D01DFE73471642C7141EF5
D80EFF551FFD299B",
        "sgx_attributes": "00000000000000300000000000000005",
        "sgx_attributes_mask": "FFFFFFFFFFFFFFFF1BFF00FFFFFFFFFFFFD"
      }
    }
  ]
}

```

TD Identity Policy

To appraise the TD, the Relying Party must generate and sign a JSON Web Token that contains reference measurements to compare against the enclave's Report measurements in order to evaluate expected or accepted values.

To generate TD identity policy, you can use this tool's following command to generate the policy template:

```
tee_appraisal_tool gen_payload -in {TDReport} -out payload.json [-v]
```

then you can edit the generated policy template based on your requirement, then you can sign the policy:

```
tee_appraisal_tool sign_policy -in payload.json -key ec_private.pem -out policy.jwt [-v]
```

Definition

The TD identity policy is a JSON object containing an “environment” JSON object and “reference” JSON object.

Environment fields and values:

- Required “class_id”
 - o An UUID string that identifies the type of the TCB the report is for TD.
 - For TDX version 1.0, UUID is “a1e4ee9c-a12e-48ac-bed0-e3f89297f687”
 - For TDX version 1.5, UUID is “45b734fc-aa4e-4c3d-ad28-e43d08880e68”
- Optional “description”

The “reference” JSON object is composed of fields listed in the table below. For each field, the definition includes how TD identity report fields are appraised against it.

| Field name | Type | Version | R/O | Field value |
|--------------------------|--------|-------------|-----|--|
| tdx_attributes | String | 1.0 and 1.5 | O | If present, (report.tdx_attributes & policy.tdx_attributes_mask) == (policy.tdx_attributes & policy.tdx_attributes_mask) |
| tdx_xfam | String | 1.0 and 1.5 | O | If present, the masked report.tdx_xfam must be equal to this policy value |
| tdx_mrconfigid | String | 1.0 and 1.5 | O | If present, the report.tdx_mrconfigid must be equal to this policy value |
| tdx_mrowner | String | 1.0 and 1.5 | O | If present, the report.tdx_mrowner must be equal to this policy value |
| tdx_mrownerconfig | String | 1.0 and 1.5 | O | If present, the report.tdx_mrownerconfig must be equal to this policy value |
| tdx_mrtd | String | 1.0 and 1.5 | O | If present, the report.tdx_mrtd must be equal to this policy value |
| tdx_rtmr0 | String | 1.0 and 1.5 | O | If present, the report.tdx_rtmr0 must be equal to this policy value |
| tdx_rtmr1 | String | 1.0 and 1.5 | O | If present, the report.tdx_rtmr1 must be equal to this policy value |
| tdx_rtmr2 | String | 1.0 and 1.5 | O | If present, the report.tdx_rtmr2 must be equal to this policy value |
| tdx_rtmr3 | String | 1.0 and 1.5 | O | If present, the report.tdx_rtmr3 must be equal to this policy value |
| tdx_mrservicetd | String | 1.5 only | O | If present, report.tdx_mrservicetd must be equal to this policy value |

Example Policy

```
{
  "policy_array":[
    {
      "environment":{
        "class_id":"45b734fc-aa4e-4c3d-ad28-e43d08880e68",
        "description":"Application TD TCB 1.5"
      },
      "reference":{
        "tdx_attributes":"0000000010000000",
        "#NOTE": "To make the following configurations take effect,
modify the value in the following lines, and uncomment the lines by removing '#'
from the key",
        "#tdx_xfam":"000000000061AE7",
        "#tdx_mrtd":"B3A00908F83729DF749648AD1769EA8F5C1205CEF83A13E23D3F
6C33C6F34769B97D5E7FD8780DC36234C8F5823A08AA",
        "#tdx_mrconfigid":"000000000000000000000000000000000000000000000000",
        "#tdx_mrowner":"000000000000000000000000000000000000000000000000",
        "#tdx_mrownerconfig":"000000000000000000000000000000000000000000000000",
        "#tdx_rtmr0":"253BD8107069789723F4FD400A229A68E1DE4D4B02F2DF7F5E5
A07BE50B8C267170C9133FA371C24FC48AD2A6823EA5A",
        "#tdx_rtmr1":"000000000000000000000000000000000000000000000000",
        "#tdx_rtmr2":"03A0D035817AFED078819D64EBD683ADCDD84B8B8C06E37A3A3
963DB4AB9B82E8ACB5A1A6E9D092B77C5FDC315FACDB2",
        "#tdx_rtmr3":"03A0D035817AFED078819D64EBD683ADCDD84B8B8C06E37A3A3
963DB4AB9B82E8ACB5A1A6E9D092B77C5FDC315FACDB2"
      }
    }
  ]
}
```

3.3 Calling the Quote Appraisal API

After the platform generates the quote, you can use the new API([tee_verify_quote_qvt](#)) to verify it and get the verification result token; then you need call the new API `tee_appraise_verification_token ()` to appraise the verification result token.

The Quote Appraisal Sample code has been provided to demo how to use these APIs. You can refer to it when generating your code. See `QuoteAppraisalSample`.

Appraisal functional APIs:

[tee_verify_quote_qvt](#)

[tee_free_verify_quote_qvt](#)

[tee_appraise_verification_token](#)

[tee_free_appraisal_token](#)

3.4 Calling the Appraisal Audit API

A tenant or relying party must audit the policies that were used in the appraisal process to make sure the final appraisal result is trusted. An API has been provided to help the user to perform the audit:

[tee_authenticate_appraisal_result](#)

4. Quote Verification and Appraisal APIs

tee_verify_quote_qvt()

The `tee_verify_quote_qvt()` function verifies an SGX/TDX Quote. It takes the same quote and collateral inputs as the existing `sgx_qv_verify_quote()`, but provides the result as a Verification Result (VR) JSON Web Token (JWT). It also has an optional input of User Data which is presented as serialized JSON string and verified using the REPORTDATA of the SGX/TDX Quote.

Syntax:

```
quote3_error_t tee_verify_quote_qvt(
    const uint8_t *p_quote,
    uint32_t quote_size,
    const sgx_qi_qve_collateral_t *p_quote_collateral,
    sgx_qi_qe_report_info_t *p_qve_report_info,
    const uint8_t *p_user_data,
    uint32_t *p_verification_result_token_buffer_size,
    uint8_t **p_verification_result_token);
```

Parameters:

p_quote [In]

Pointer to an SGX/TDX Quote. The QVE only supports version 3 of the SGX ECDSA Quote. Currently, the QVE only supports Quotes with CertType = 5. The Intel signed QVE will only verify Quotes generated by an Intel Signed QE. This type of certification data contains the PCK Certificate Chain in the Quote.

quote_size [In]

Size of the buffer pointed to by p_quote (in bytes).

p_quote_collateral [In]

This parameter is optional. If not NULL, this is a pointer to the Quote Certification Collateral provided by the caller. The quote collateral structure contains a version number. This is the data that is required to fully verify the quote. Such as the TCInfo, QEIdentity and CRL structures, etc. If it is NULL, the DCAP library will attempt to retrieve the collateral from the platform quote provider library if available. If the platform quote provider library is not available or the collateral cannot be retrieved, this API will return `SGX_QL_PLATFORM_LIB_UNAVAILABLE` or `SGX_QL_UNABLE_TO_GET_COLLATERAL` respectively.

p_qae_report_info [In, Out]

This parameter is optional. If not NULL, then this parameter should provide a `target_info` structure for an enclave which will verify the generated report. If this parameter is provided, then the QVE will be used to perform the verification and it will generate a report using the `target_info`

provided in the `sgx_qi_qe_report_info_t` structure . The `QVE.REPORT.REPORT_DATA` is set to `SHA256_HASH(VR JWT)`.

`const uint8_t *p_user_data [IN]`

If not `NULL`, this points to a buffer holding a null-terminated serialized JSON string for user data. The hash of this string will be verified to match the `SGX/TDX reportData` held in the input quote. Upon successful verification, the user data will be converted back into JSON format and included in the output VR JWT.

`uint32_t *p_verification_result_token_buffer_size[IN, OUT]`

Points to a variable holding the size of the input `quote_verification_token` buffer. If the size is not big enough, the function will return an error code with the needed size in this variable.

`uint8_t **p_verification_result_token[OUT]`

If a sufficiently sized buffer is provided in `p_verification_result_token_size / p_verification_result_token`, then a VR(Verification Result) JWT as a null-terminated string is output in this buffer.

The output VR JWT can be unsigned or signed.

Return Values

`SGX_QL_SUCCESS:`

Successfully evaluated the Quote.

`SGX_QL_INVALID_PARAMETER:`

One of the input parameters is invalid.

`SGX_QL_QUOTE_FORMAT_UNSUPPORTED:`

The format of the inputted Quote is not supported. Either because the header information is not supported or the Quote is malformed in some way.

`SGX_QL_QUOTE_CERTIFICATION_DATA_UNSUPPORTED:`

The QVL doesn't support the certification data in the Quote. Currently, the QVL only support QE Report Certification Data.CertType = 5.

`SGX_QL_QE_REPORT_UNSUPPORTED_FORMAT:`

The QVL doesn't support the format of the QE3/TDQE Report in the Quote.

`SGX_QL_QE_REPORT_INVALID_SIGNATURE:`

The signature over the QE3/TDQE Report is invalid.

SGX_QL_QE_REPORT_ATT_KEY_MISMATCH:

The Attestation Key contained in the Quote was not generated by the QE3/TDQE described in the Quote.

SGX_QL_PCK_CERT_UNSUPPORTED_FORMAT: The format of the PCK Cert is unsupported.

SGX_QL_PCK_CERT_CHAIN_ERROR:

There was an error verifying the certificate chain contained in the PCK Cert. This error can also be returned while validating the PCK Cert revocation.

SGX_QL_TCBINFO_UNSUPPORTED_FORMAT:

The format of the TCBInfo structure is unsupported.

SGX_QL_TCBINFO_CHAIN_ERROR:

There was an error verifying the signature chain of the TCBInfo. This error can also be returned while validating the TCBInfo revocation.

SGX_QL_TCBINFO_MISMATCH:

PCK Cert FMSPc does not match the TCBInfo FMSPc.

SGX_QL_QEIDENTITY_UNSUPPORTED_FORMAT:

The format of the QEIdentity structure is unsupported.

SGX_QL_QEIDENTITY_MISMATCH:

The identity of the TDQE contained in the Quote does not match the provided QEIdentity.

SGX_QL_QEIDENTITY_CHAIN_ERROR:

There was an error verifying the signature chain of the QEIdentity. This error can also be returned while validating QEIdentity revocation.

SGX_QL_OUT_OF_MEMORY:

Heap memory allocation error in the QVL.

SGX_QL_INVALID_REPORT:

Report MAC check failed on QE3/TDQE Report.

SGX_QL_PLATFORM_LIB_UNAVAILABLE:

The Quote Verification Library could not locate the Platform Quote Provider Library or one of Platform Quote Provider Library's required function.

SGX_QL_NO_QUOTE_COLLATERAL_DATA:

The Platform Quote Provider Library available, but it could not retrieve the data.

SGX_QL_ERROR_UNEXPECTED: An unexpected internal error occurred.

SGX_QL_UNKNOWN_MESSAGE_RESPONSE:

QVL received unexpected message while retrieving verification collateral.

SGX_QL_ERROR_MESSAGE_PARSING_ERROR:

Generic message parsing error from the attestation infrastructure while retrieving the platform data.

SGX_QL_PLATFORM_UNKNOWN:

The Platform Quote Provider Library was not able to collect the TDTCBInfo for this platform.

SGX_QL_TDX_MODULE_MISMATCH:

The security level of the TDX Module installed on the platform cannot be evaluated because it is not represented in the TDTCBInfo. This can happen when an old TDTCBInfo is used in the verification.

tee_free_verify_quote_qvt()

The `tee_free_verify_quote_qvt()` function frees “quote verification result token” buffer that is returned by `tee_verify_quote_qvt` function. So this function need work with `tee_verify_quote_qvt` function together.

Syntax:

```
quote3_error_t tee_free_verify_quote_qvt(
    uint8_t * p_verification_result_token,
    uint32_t *p_verification_result_token_buffer_size);
```

Parameters:

`uint8_t *p_verification_result_token[IN]`
pointer to the verification result token buffer.

`uint32_t *p_verification_result_token_buffer_size[IN]`
Points to the verification result token buffer size.

The Quote Appraisal Library (QAL) implements a set of functions to receive input Verification Result JWT and Quote Appraisal Policies (QAP), appraise the Verification Result JWT against the QAP to generate Appraisal Result JWT.

Return Values

SGX_QL_SUCCESS:

Successfully evaluated the Quote.

SGX_QL_INVALID_PARAMETER:

One of the input parameters is invalid.

tee_appraise_verification_token()

tee_appraise_verification_token() appraises a Verification Result JWT against one or more Quote Appraisal Policies. If the Verification Result meets a policy, then the policy ID is inserted into the Appraisal Result JWT.

Syntax:

```
quote3_error_t tee_appraise_verification_token(  
    const uint8_t *p_verification_result_token,  
    uint8_t **p_qaps,  
    uint8_t qaps_count,  
    const time_t appraisal_check_date,  
    sgx_ql_qe_report_info_t *p_qae_report_info,  
    uint32_t *p_appraisal_result_token_buffer_size,  
    uint8_t **p_appraisal_result_token);
```

Parameters:

uint8_t* p_verification_result_token[IN]

- A null-terminated string containing the input Verification Result JWT generated by tee_verify_quote(). The VR JWT can be unsigned or signed

uint8_t** p_qaps[IN]

- Points to an array of pointers, with each pointer pointing to a buffer holding a quote appraisal policy JWT token. Each token is a null-terminated string holding a JWT. Each token can hold one or more policies.
- Each policy token is signed JWT, the signing cert or public key needs to be included in the QAP JWT header.

uint8_t qaps_count[IN]

Holds the number of pointers in the p_qaps array.

time_t appraisal_check_date [IN]

User input, used by the appraisal engine as its “current time” for expiration dates check.

`sgx_ql_qe_report_info_t* p_enclave_report_info[IN,OUT],`

This parameter is optional. If not NULL, then this parameter should provide a `target_info` structure for an enclave which will verify the generated report. If this parameter is provided, then the QAE will be used to perform the appraisal and it will be generate a report using the `target_info` provided in the `sgx_ql_qe_report_info_t` structure . The QAE.REPORT.REPORT_DATA = SHA256_HASH(AR JWT).

`uint32_t* p_appraisal_result_token_buffer_size[IN,OUT]`

Points to a variable holding the size of the `p_appraisal_result_token` buffer

If the size is not big enough, the function will return an error code with the needed size in this variable.

`uint8_t** p_appraisal_result_token [OUT]`

If a sufficiently sized buffer is provided in `p_appraisal_result_token_buffer_size / p_appraisal_result_token`, then an Appraisal Result (AR) JWT as a null-terminated string is output in this buffer.

The output AR JWT can be unsigned or signed.

Notes:

For parameters `qaps_count` and `p_qaps`, user needs to provide a “TD Identity” policy or an “Application Enclave Identity” policy. For platform policy, user may provide the platform policy when he/she call this API, but it is not a must requirement. If a platform policy is not provided, this API will internally try to retrieve the platform policy from the cache server (like PCCS), if a platform policy could NOT be retrieved from the cache server, this API will use the hard-coded ‘[strict](#)’ platform policy. The priority for platform policy selection is as follows:

1. User input platform policy provided to this API.
2. If user does not provide platform policy to this API, it will internally try to get platform policy from the cache server.
3. If the cache server doesn’t contain a platform policy, this API will use the hard-coded ‘[strict](#)’ platform policy.

Return Values

`SGX_QL_SUCCESS:`

Successfully evaluated the Quote.

`SGX_QL_INVALID_PARAMETER:`

One of the input parameters is invalid.

`SGX_QL_OUT_OF_MEMORY:`

Heap memory allocation error in the QVL.

SGX_QL_ERROR_UNEXPECTED: An unexpected internal error occurred.

SGX_QL_UNKNOWN_MESSAGE_RESPONSE:

QVL received unexpected message while retrieving verification collateral.

SGX_QL_ERROR_MESSAGE_PARSING_ERROR:

Generic message parsing error from the attestation infrastructure while retrieving the platform data.

tee_free_appraisal_token ()

The tee_free_appraisal_token() function frees “appraisal result token” buffer that is returned by tee_appraise_verification_token function. So this function need work with tee_appraise_verification_token function together.

Syntax:

```
quote3_error_t tee_free_appraisal_token ( uint8_t * p_appraisal_result_token);
```

Parameters:

uint8_t *p_appraisal_result_token[IN]
pointer to the appraisal result token buffer.

Return Values

SGX_QL_SUCCESS:

Successfully evaluated the Quote.

SGX_QL_INVALID_PARAMETER:

One of the input parameters is invalid.

tee_authenticate_appraisal_result ()

The tee_authenticate_appraisal_result() function will check:

Whether the input policies are used in the appraisal process by comparing the policies with the appraisal result.

Syntax:

```
quote3_error_t tee_authenticate_appraisal_result (  
    const uint8_t * p_appraisal_result_token,  
    const tee_policy_bundle_t *p_policies,  
    tee_policy_auth_result_t *result);
```


Parameters:

const uint8_t *p_appraisal_result_token[IN]

pointer to the appraisal result token JWT buffer that is generated by the “tee_appraise_verification_token” function.

const tee_policy_bundle_t *p_policies[IN]

Policies that will be used to authenticate the appraisal result. To make sure the appraisal process use the same policies during functional call “tee_appraise_verification_token”. For this parameter’s structure, you can refer to the tee_policy_bundle_t definition in Appendix’s [Data Structure](#) part.

tee_policy_auth_result_t* result[OUT]

Output the authentication result. For this parameter’s detail information, you can refer to the tee_policy_auth_result_t definition in Appendix’s [Data Structure](#) part.

Return Values

SGX_QL_SUCCESS:

Successfully evaluated the Quote.

SGX_QL_INVALID_PARAMETER:

One of the input parameters is invalid.

SGX_QL_OUT_OF_MEMORY:

Heap memory allocation error in the QVL.

SGX_QL_ERROR_UNEXPECTED: An unexpected internal error occurred.

5. Appendix

5.1 Appraisal Result Token

The Appraisal Result Token (ART) is a JSON Web Token (JWT). Every JWT has a JOSE header with the fields in the table below. Some field names are defined in [RFC7515](#) or [RFC7519](#)

| JOSE header field | Req./Opt. | Notes |
|-------------------|-----------|--|
| alg | O | Registered JWT claim for Algorithm, defined in RFC 7518 Only required for signed JWT |
| jwk | O | Registered JWS claim for JSON Web Key, defined in RFC 7517 Only required for signed JWT |
| iat | R | “Issued At” time. Its value MUST be a number containing a NumericDate value |
| exp | R | Expiration Time. Its value MUST be a number containing a NumericDate value, which is “Seconds since the Epoch”. |
| version | O | A string identifying the version for the report data structure format. This string is composed of two decimal numbers separated by a dot in between, e.g., “1.2”. The number on the left is major version, and the number on the right is minor version. |

Example JWT JOSE header:

```
{
  "alg": "ES384",
  "typ": "JWT",
  "jwk": {
    "kty": "EC",
    "crv": "P-384",
    "x": "<x-value>",
    "y": "<y-value>",
    "kid": "<key-ID-string>"
  }
}
```

For every input report JSON object, an appraisal result JSON object is produced with these fields:

- Field “appraisal_result”, a number representing one of the results.
 - 1 – a policy is identified and appraisal is successful
 - 0 – a policy is identified but appraisal failed
 - -1 – no policy is identified for this report. In this case, the policy field is not present.
- Report information

- Report environment
- Report measurement including endorsed values upon successful appraisal
- Policy information, if a policy is identified for the report. Otherwise, this field is not present
 - a) The policy environment copied directly from the policy
 - b) Policy signing public key and signature
 - Public key in JSON Web Key ([RFC7517](#)) format, with algorithms defined in [RFC7518](#).
 - Signature is the signature section in the signed JWT string.
 - c) “appraisal_check_date”, which is the reference time that the appraisal engine uses to check expiration dates. This is typically the current time.

The appraisal result includes an overall appraisal result claim, “overall_appraisal_result”, with the possible values:

- 1 – all reports have been successfully appraised
- 0 – at least one report failed appraisal
 - There could be zero or more reports without identified policies
- -1 – There is at least one report without identified policy
 - All other reports (if any) are appraised successfully

The result JWT payload is a JSON object as defined below:

```
{
  "overall_appraisal_result" : 1,
  "nonce" : xxxx,
  "appraisal_check_date": 17023703,
  "appraised_reports": [
    {
      "appraisal_result": 1,
      "report":{
        "environment": {
          // Environment fields
        },
        "measurement": {
          // Measurement values, including endorsed values
        }
      }
    },
    "policy": {
      "environment" : {
        "class_id": "3123ec35-8d38-4ea5-87a5-d6c48b567570",
        "description": "SGX Platform Policy for CSP"
      },
      "signing_key" : {
        // Signing public key in JSON Web Key (RFC7517) format
      },
      "signature" : {
```

```

        // JWT signature, the signature section in the signed JWT
string
    }
}
},
{
    "appraisal_result": 1,
    "report":{
        "environment": {
            // Environment fields
        },
        "measurement": {
            // Measurement values, including endorsed values
        }
    }
    "policy": {
        "environment" : {
            "class_id": "bef7cb8c-31aa-42c1-854c-10db005d5c41",
            "description": "SGX Enclave Policy"
        },
        "signing_key" : {
            // Signing public key in JSON Web Key (RFC7517) format
        },
        "signature" : {
            // JWT signature, the signature section in the signed JWT
string
        }
    }
},
    ],
    "certification_data" : [
        {
            "certification_data": {
                "qe_identity_issue_chain": "xxx",
                "root_ca_crl": "xxx",
                "pck_crl": "xxx",
                "pck_crl_issuer_chain": "xxx",
                "tcb_info": "xxx",
                "qe_identify": "xxx",
                "tcb_info_issue_chain": "xxx"
            }
        }
    ]
}

```

5.2 Data Structures

```

typedef enum _tee_platform_policy_type_t
{
    DEFAULT_STRICT = 0,
    CUSTOMIZED
} tee_platform_policy_type_t;

typedef struct _tee_platform_policy_t
{
    tee_platform_policy_type_t pt;
    const uint8_t* p_policy;
} tee_platform_policy_t;

typedef struct _tee_policy_bundle_t
{
    const uint8_t *p_tenant_identity_policy;
    tee_platform_policy_t platform_policy;

    tee_platform_policy_t tdqe_policy; /* For tdqe. Only for TDX and only need
to be set when user uses a seperate tdqe_policy
                                     * instead of an integrated
platform_policy including both TDX platform policy and TDQE. */

    tee_platform_policy_t reserved[2]; /* Reserved for future usage */
} tee_policy_bundle_t;

typedef enum _tee_policy_auth_result_t
{
    TEE_AUTH_INCOMPLET = -1, /* Only part of the policies are provided and
authenticated successfully. For example, you only input
                             * SGX platform policy for an SGX appraisal
token, and the platform policy is authenticated successfully */
    TEE_AUTH_SUCCESS = 0, /* All the policies are authenticated
successfully. For SGX, both SGX platform policies are provided and successfully
*/
    TEE_AUTH_FAILURE = 1, /* At least one of the input policies are
authenticated failed */
} tee_policy_auth_result_t;

```

6. Disclaimer and Legal Information

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request.

Intel technologies features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at Intel.com, or from the OEM or retailer.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.

Intel, the Intel logo, Xeon, and Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

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

© Intel Corporation.

This software and the related documents are Intel copyrighted materials, and your use of them is governed by the express license under which they were provided to you (**License**). Unless the License

provides otherwise, you may not use, modify, copy, publish, distribute, disclose or transmit this software or the related documents without Intel's prior written permission.

This software and the related documents are provided as is, with no express or implied warranties, other than those that are expressly stated in the License.