OpenAttestation SDK Overview

March 2012

OpenAttestation SDK (OAT)

A SDK for Remote Attestation

Table of Contents

1 Introduction

The Trusted Computing Group has defined a series of specifications that define how a commercial computing platform can support code measurement in a trusted manner. Intel has developed an OpenAttestation SDK built on NSA's National Information Assurance Research Laboratory (NIARL) developed Host Integrity at Startup to measure and report status for host platforms which contain a Trusted Platform Module (TPM). The implementation takes advantage of Infrastructure Work Group Integrity Report Schema Specification
http://www.trustedcomputinggroup.org/resources/infrastructure_work_group_integrity_report_schema_specification_version_10/

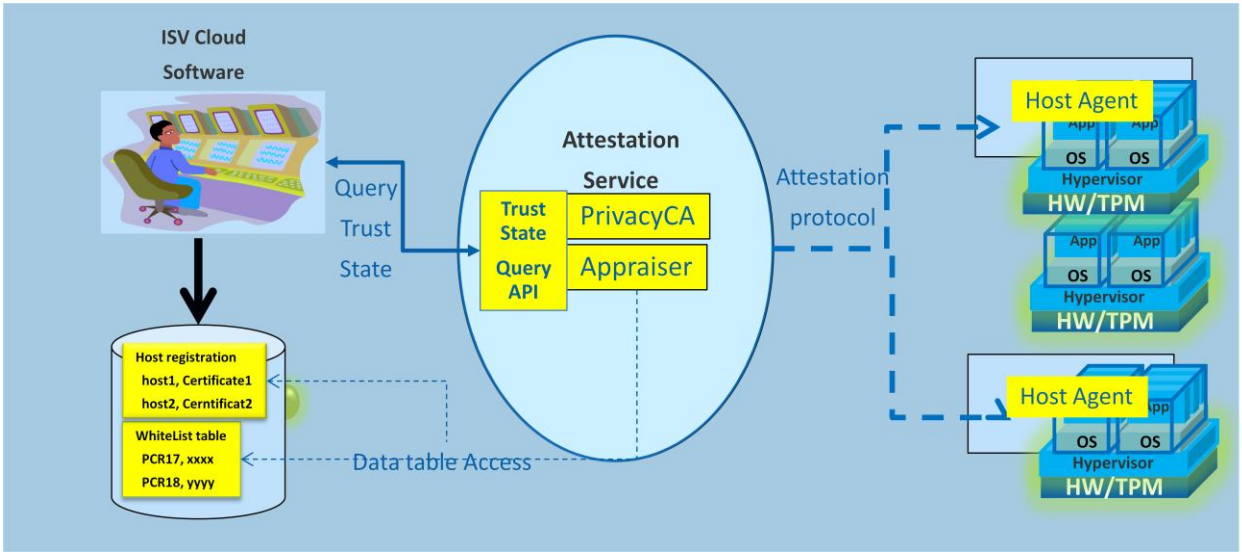
The OpenAttestation SDK supports web API for 3rd party software to integrate and access web-based attestation appraisals in order to support cloud usage model.

The OpenAttestation SDK is intended to be merged, modified and distributed as 3rd party ISV's cloud management stacks. ISV should enhance OAT for its distribution or enhance security implementation into their products

1.1 Scope

This guide describes OpenAttestation architecture and exported APIs. The intended audients can be ISV developers or administers of cloud providers.

1.2 Architecture



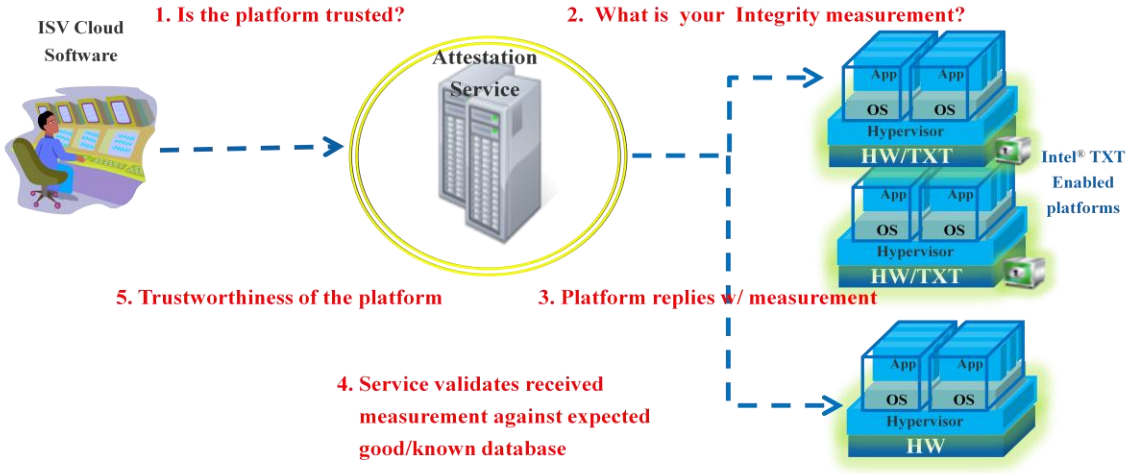
This above diagram highlights OpenAttestation architecture. OpenAttestation, as part of 3rd party ISV's software stack, enables ISV software remotely to retrieve and verify target

hosts' TPM PCRs and verify hosts' integrity through exported Query API. Accordingly, administer of cloud providers can pre-setup WhilteList tables, e.g., good/known PCR values, a.k.a, measurements, through SDK exported data table access APIs.

OpenAttestation service runs with Privacy Certificate Authority (PrivacyCA) and Appraiser to communicate with HostAgenst runs on target hosts. PrivacyCA provisions and certifies hosts' Attestation Identify key, where Appraiser communicates with HostAgent, access through TrouserS TSS stack, to retrieve and verify a host's PCRs.

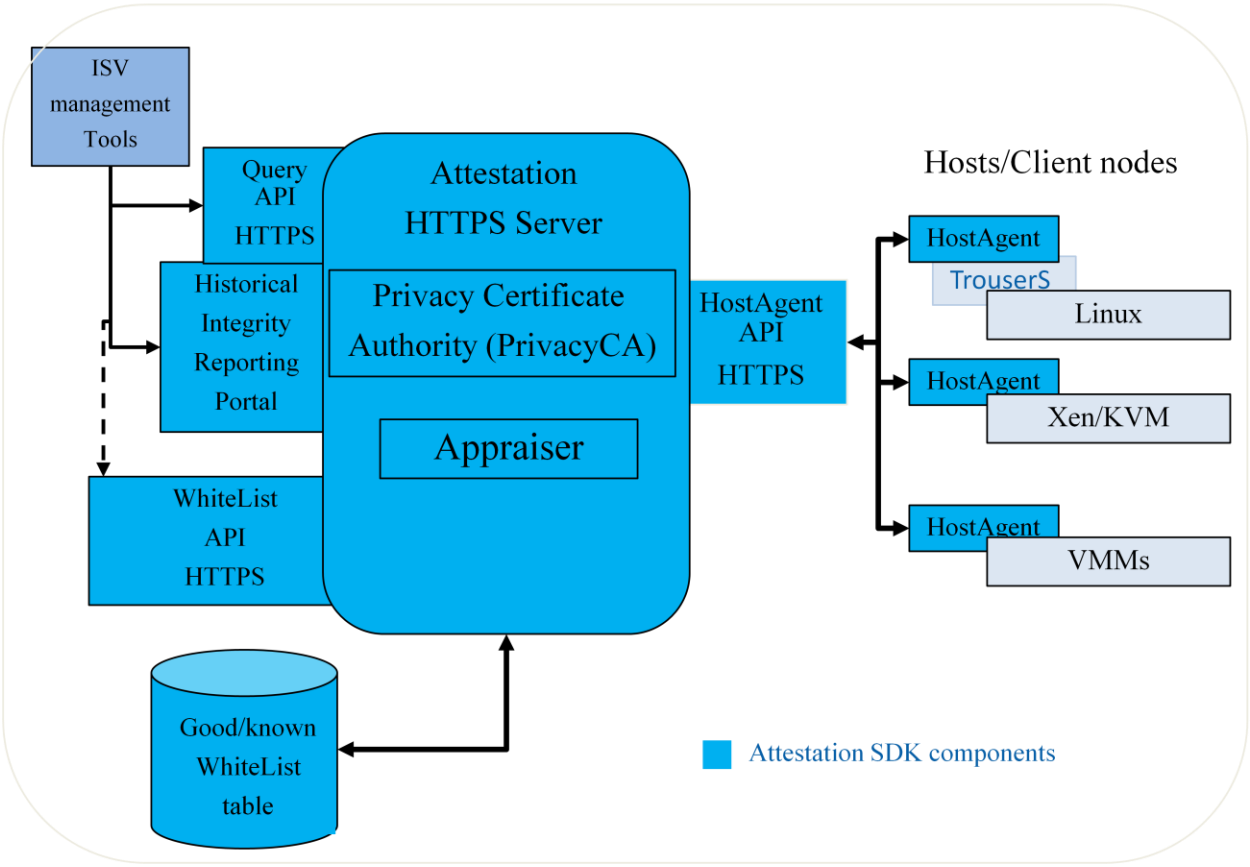
2 A Sample Usage model

Following diagram exemplifies possible usage model in cloud computing environment. Where in a cloud computing environment, a subscriber may require a service must be run on a trusted platform. That is, hosts running with good/known software. Attestation service provides ISV software with capability to verify target hosts' integrity before dispatch task(s) onto the host.



3 OpenAttestation Components

Following diagram describes exported components of Attestation service



Query API HTTPS HTTPS service exporting Restful API to enable ISV software to query hosts' integrity status

Whitelist API HTTPS HTTPS service exporting Restful API to enable administrator to create/delete good/known PCRs register data in/from database

Historical Integrity Reporting Portal Appraiser, as a reference implementation, internally tracks each host's historical integrity reports. The Portal provides an interface to view historical data as well as details of integrity data reported by hosts

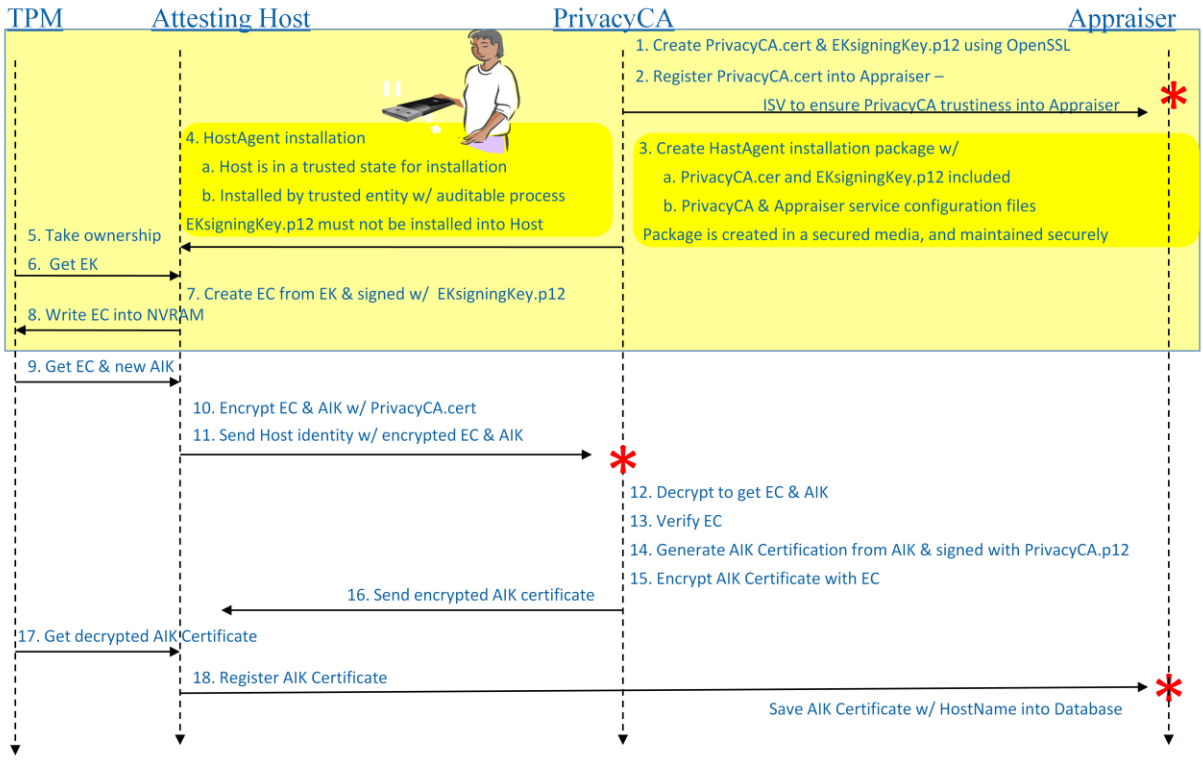
HostAgent API HTTPS Connection point for HostAgent to send in integrity report with TCG Integrity Report Schema

ProvacyCA verifies Hosts' TPM Endorsement Key (EK) & issue EK Certificate (EKC). EKC used by Host to request Attestation Identification Key (AIK) Certificate (AIC)

Appraiser verifies Host measurement which signed by Host AIC

4 OpenAttestation control flow

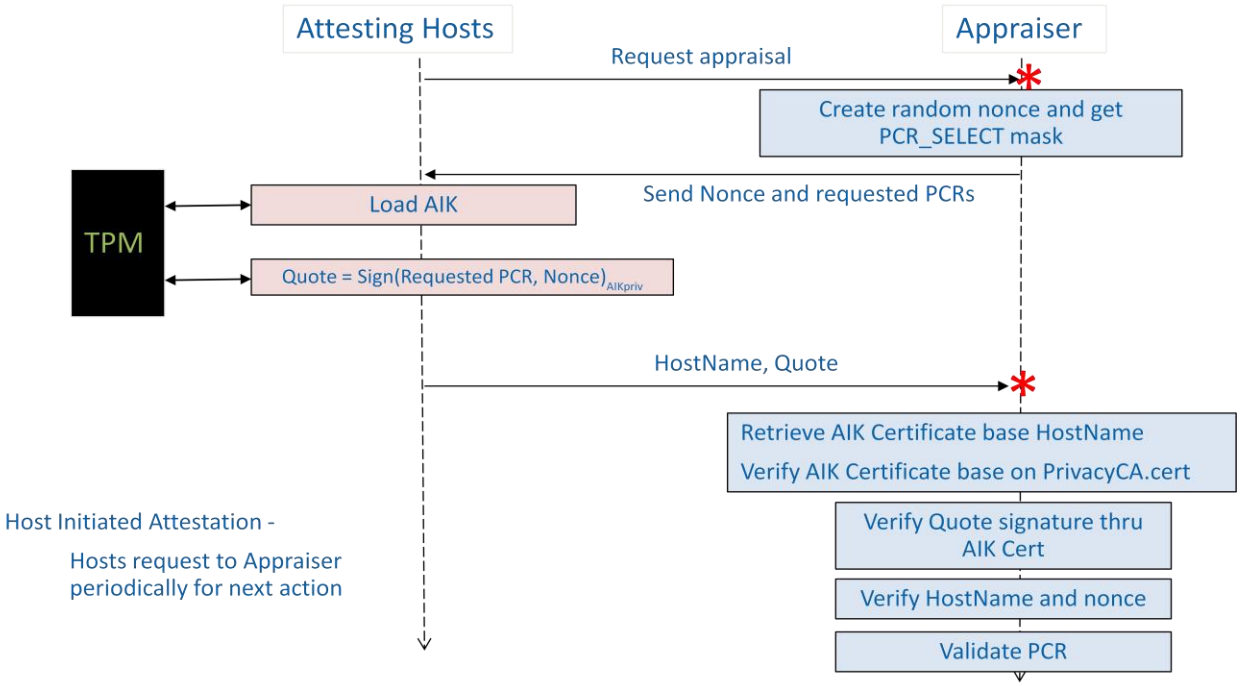
4.1 HostAgent Installation/Provisioning – EK and AIK certificates



The above flow describes HostAgent provisioning to get Endorsement Key (EK) certificate and how it gets Attestation Identity Key (AIK) certificate from ProvicyCA, and then register its Hostname associated with AIC into Appraiser

5 Attestation Control flow

5.1 Host initiated attestation



Attestation flow is run as *Host initiated Attestation*, that is, Host system will poll Appraiser periodically to query if any active task it should perform, Appraiser can then set active task to request integrity report.

6 OpenAttestation API

We will describe OpenAttestation exported Restful APIs, which is Integrity Query API enables querying a platform's integrity state, and WhiteList API which enables administrator to setup good/known measurement PCRs for appraiser.

6.1 Security Notes

For access control and security purpose, all the APIs exported from Attestation service will be HTTPS based. However the SDK internal implementation doesn't impose any authentication or access control verification, rather it solely depends following 2 mechanisms for authentication –

- ISV specific authentication verification – An *Auth-blob* is included in all the headers as request to attestation service. Attestation service will pass this blob to ISV specific function for authentication verification.
- Tomcat Truststore client and server certificate authentication for platform accesses – Tomcat provided SSL authentication through Keystore/Truststore authentication for client to access server as well as client verifying server before connection

It is strongly recommended that ISV must implement its own software specific authentication logic, and Attestation service provider should also implement Tomcat 2-

way authentication to securely control platforms that are allowed to access to API services

6.2 API Request/Response Types

Current Attestation APIs support JSON data serialization format, where

- The request format is specified by using *Context-Type* header and is required for operations that havea request body
- The response format is specified in request header using *Accept*
- *Accept* type is default to JSON if not specified in request header

6.3 Authentication header

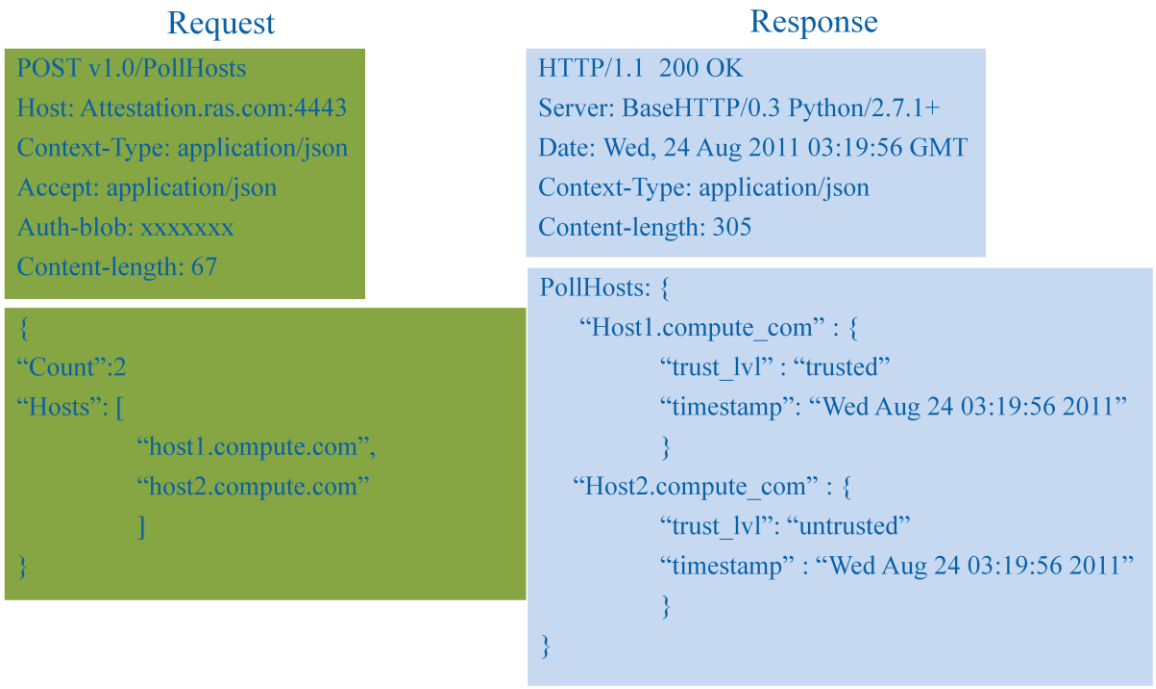
All requests to attestation server is required to include an authentication blob, Auth-blob, within request header. Where Auth-blob is ISV authentication specific and is opaque to OpenAttestation API.

Note: ISV must implement its own ***ISV_Authentication_verification()*** function to complete the authentication logic. The default ***ISV_Authentication_Verification()*** logic is true.

```
Context-Type: application/json
Accept:  application/json
Auth-Blob: ISV_specific_string
```

6.4 Example of request with headers: JSON

Following example should a request to attestation server and its response in JSON format



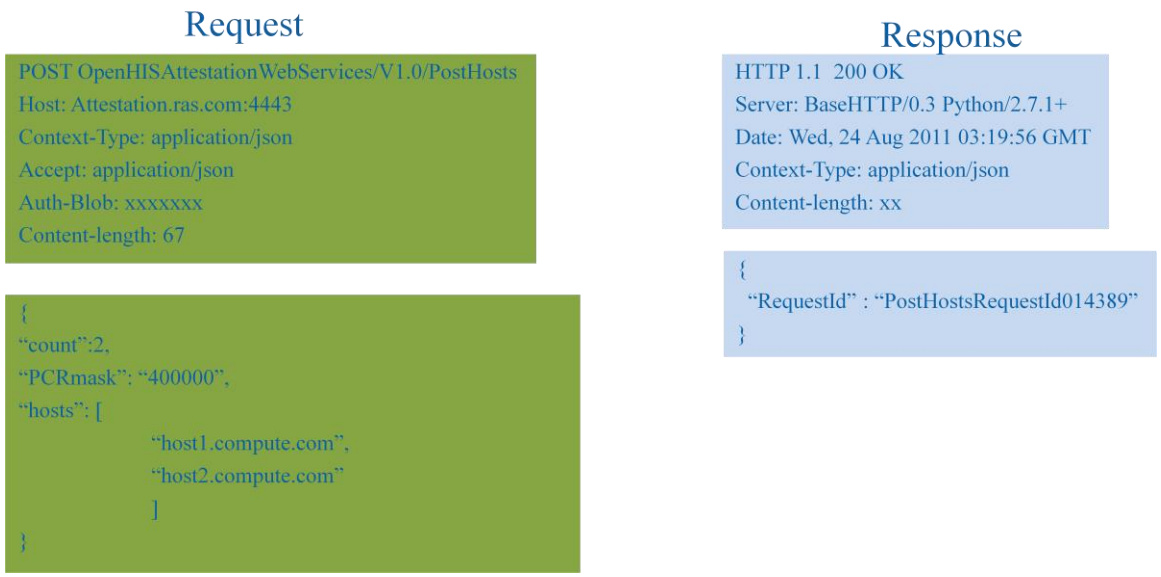
6.5 Integrity Query API

Following lists Integrity query APIs

Verb	Command	Body/URI	Result	Description
POST	/PostHosts	Body	Async	Query hosts trustworthiness
GET	/PostHosts	Body	Sync	Retrieve previously posted /PostHosts result
POST	/PollHosts	Body	Sync	Post and wait for hosts trustworthiness

6.5.1 POST /PostHosts

Post request to Attestation service for retrieving specified hosts' integrity reports. Attestation server returns *RequestId* immediately for application to poll the station result asynchronously



Where parameters

PCRmask – specify specific PCR index request to be return

- An optionally parameter
- 6 hex ascii can be specified to specify PCR index registers to be returned to application
    - "FFFFFF" – PCR0~23 from right-most to left-most bits

RequetId

- An unique string created by the server randomly, and is opaque to application

NOTE: Attestation server doesn't using PCRmask to selectively verify hosts PCRs, rather the server uniformly verify hosts' PCRs by using system-wide defined PCR_SELECTED

6.5.2 GET /PostHosts

Retrieve previously posted /PostHosts result base on *requested.* The application needs to poll or retry the operation to check if the server has completed the operation.

Server may return (0/1/2) for operation to indicate either (in-progress/done/failed)

Application performs checking –

ReturnStatus = (GET /PostHosts).body.jobstatus
If ReturnStatus == 0 : try later
If ReturnStatus == 2 : failed operation, do recovery
If ReturnStatus == 1 : Get result from response body

Response

Request

```
HTTP 1.1  200 OK
Server: BaseHTTP/0.3 Python/2.7.1+
Date: Wed, 24 Aug 2011 03:19:56 GMT
Context-Type: application/json
```

```
GET OpenHISAttestationWebServices/V1.0/PostedHosts
Host: Attestation.ras.com:4443
Context-Type: application/json
Accept: application/json
Auth-Blob:xxxxxx
Content-length: xx
```

```
{
 "requestId" : "PostHostsRequestId014389"
 "jobstatus" : "1",
 "jobresult : "done",
 "count" : "2",
 "hosts": [
    { "host_name":"host1.compute.com",
     "trust_lvl" : "trusted",
     "vtime": "Wed Aug 24 03:19:56 2011",
     "pcr_values":[{"numer":"17","value":"0123456789012345678901234567890123456789"},
                   {"numer":"22","value":"0123456789012345678901234567890123456789"}
                  ]
},
    { "host_name":"host2.compute.com"
     "trust_lvl" : "untrusted",
     "vtime: 'Wed Aug 24 03:19:56 2011",
     "pcr_values":[{"numer":"17","value":"0123456789012345678901234567890123456789"},
                   {"numer":"22","value":"0123456789012345678901234567890123456789"}
                  ]
    }
  ]
}
```

```
{
   "requestId" : "PostHostsRequestId014389"
}
```

6.5.3 POST /PollHosts

Post and get current integrity report from attestation server.

Request

Response

```
POST OpenHISAttestationWebServices/V1.0/PollHosts
Host: Attestation.ras.com:4443
Context-Type: application/json
Accept: application/json
Auth-Blob:xxxxxxx
Content-length: 67
```

```
{
count:2
PCRmask: 400000
hosts: [
            host1.compute.com,
            host2.compute.com
            ]
}
```

```
HTTP/1.1  200 OK
Server: BaseHTTP/0.3 Python/2.7.1+
Date: Wed, 24 Aug 2011 03:19:56 GMT
Context-Type: application/json
Content-length: 305
```

```
{
 "hosts": [
    { "host_name":"host1.compute.com",
     "trust_lvl" : "trusted",
     "vtime": "Wed Aug 24 03:19:56 2011",
     "pcr_values":[{"numer":"17","value":"0123456789012345678901234567890123456789"},
                   {"numer":"22","value":"0123456789012345678901234567890123456789"}
                  ]
    },
    { "host_name":"host2.compute.com"
     "trust_lvl" : "untrusted",
     "vtime: 'Wed Aug 24 03:19:56 2011",
     "pcr_values":[{"numer":"17","value":"0123456789012345678901234567890123456789"},
                   {"numer":"22","value":"0123456789012345678901234567890123456789"}
                  ]
    }
  ]
}
```

The request is to retrieve hosts' integrity report from attestation server. The server may return following status for each of the requesting hosts

- Tursted – the host's integrity has been verified against WhiteList and passed
- Untrusted – the host's integrity report has failed verification
- Unknown – The attestation server has not or can't connect to the request host for verification
- Pending – the server is in process of verifying target host's integrity report

6.6 WhiteList API

WhiteList API enables programming interfaces to manipulate good/known measurements which administrator can create and for attestation server to validate integrity reports against.

Current PCR-based measurement Whitelist contains following entries –
(index, PCR_number, PCR_value, PCR_desc, create_time, create_user, last_update_time, last_update_user)

- Index: An automatically incremental sequential number
- PCR_number: PCR index register number 0~23
- PCR_value: expected good PCR value
- PCR_desc: ASCII string in describing content of the PCR index register
- Create_time: The time this entry got created
- Create_user: The hostname of the system who creates the entry
- Last_update_time: The time this entry last got updated
- Last_update_user: The hostname of the system who updated the entry

Requirements –

(PCR_number + PCR_value) must be unique across entries

6.6.1 Interface Support

Following interfaces are supported through API

- Create a new PCR entry
- Retrieve an existing PCR entry
- Delete an existing PCR entry
- Update an existing PCR entry

6.6.1.1 Create a new PCR entry

Create a new (index, PCR_number, PCR_value, PCR_desc) entry into WhiteList
Format: PUT OpenAttestationManifestWebServices/V1.0/PCR

6.6.1.2 Retrieve an existing PCR entry

Get all PCR entries
Format: GET OpenAttestationManifestWebServices/V1.0/PCR

Get selected PCR entry
Format: GET OpenAttestationManifestWebServices/V1.0/PCR?Index=index

Get all entries with PCR_number
Format: GET OpenAttetsationManifestWebServices/V1.0/PCR?PCRnumber=number

Get entries with PCRdescription
Format: GET OpenAttetsationManifestWebServices/V1.0/PCR?PCRdesc=description

Get entries with combination
Format: GET
OpenAttetstaionManifestWebService/V1.0/PCR?PCRnumber=number&PCRdesc=descri
ption

6.6.1.3 Delete an existing PCR entry

Delete a specific entry
Format: DELETE OpenAttestationManifestWebServices/V1.0/PCR?Index=index

6.6.1.4 Update an existing PCR entry

Update a PCR entry in request body
Format: UPDATE OpenAttestationManifestWebServices/V1.0/PCR