CHAPTER 5

Boundary Control in the Cloud: Geo-Tagging and Asset Tagging

Chapters 3 and 4 focused on platform boot integrity, trusted compute pools, and the attestation architecture. They covered the reference architecture for how organizations and service providers can deploy trusted pools as the enabler for trusted clouds. Data and workload locality and data sovereignty are top-line issues for organizations considering migrating their workloads and data into the cloud. A fundamental capability that is needed is to reliably identify the location of physical servers on which the data and workloads reside. Additionally, organizations would need to produce audit trails of data and workload movement, as well as carry out effective forensics when the occasion demands it. In particular, the asset location identification and attestation capability needs to be verifiable, auditable, and preferably anchored in hardware. These capabilities enable workload and data boundary control in the cloud, effectively conferring users control over where workloads and data are created, where they are run, and where they migrate to for performance, optimization, reliability, and high-availability purposes.

Geolocation and geo-fencing, and the higher level concept of asset tagging, are technology components and associated usages that enable monitoring and control of data processing and workload movement, and they are the subject of this chapter. Geolocation and geo-fencing constitute fitting adjacencies to trusted compute pools usages, and provide a critical security control point to assess and enforce in a data center. Asset tagging is still an emergent industry practice. So, we'll start with some definitions to provide the context, followed by a discussion of enabling the logical control points. The next step is to link asset tagging with the trusted compute pools usages discussed in the earlier chapters. Asset tagging is highly synergistic with trusted compute pools, and the capability adds significant value to any trusted data center operations and compute pools deployment. We will elaborate on this idea as we describe a reference implementation in the last part of this chapter.

Geolocation

As the NIST Interagency Report 7904 clearly delineates, shared cloud computing technologies, designed to be agile and flexible, transparently use geographically distributed resources to process workloads for their customers.[1] However, there are security and privacy considerations in allowing workloads—namely data and applications—to run in geographically dispersed locations with unrestricted workload migration. Even with controls governing the location of the launch of a workload, without additional controls and restrictions in place that workload could move from cloud servers located in one geographic location to servers located in another geographic location. Each country has laws protecting data security, privacy, and other aspects of information technology (IT). An organization may decide that it needs to restrict which cloud service providers and servers it uses based on their locations so as to ensure compliance. An example of such a requirement is to use only cloud servers physically located within the same country as the organization.

Determining the physical location of an object, such as a cloud computing server, is generally known as *geolocation.* It can be a logical description of geographic information, such as country or city, or it can be GPS-based latitude and longitude information. Geolocation can be accomplished in many ways, with varying degrees of accuracy, but traditional geolocation methods are not secure and they are presently enforced through management and operational controls not easily automated and scaled; therefore, traditional geolocation methods cannot be trusted to meet cloud security needs. NIST IR 7904 describes geolocation as follows:

> *Geolocation enables identification of a cloud server's approximate location by adding that information to the server's root of trust. The hardware root of trust is seeded by the organization with the host's unique identifier and platform metadata stored in tamperproof hardware. This information is accessed using secure protocols to assert the integrity of the platform and confirm the location of the host.*[2]

Geo-tagging constitutes the process of defining, creating, and provisioning a set of geolocation objects to a computing device securely. An interesting and very relevant application of the geo-tag is the enforcement of boundary control based on geo-tags; the concept is called *geo-fencing.*

Geo-fencing

The concept of geo-fencing is not new. It has been applied successfully in industries such as mobile computing, supply chain management, and transportation logistics. Geo-fencing is about defining geographical or virtual boundaries using a variety of GPS,

[1] Erin K. Banks et al., "Trusted Geolocation in the Cloud: Proof of Concept Implementation" (draft), NIST Interagency Report 7904, U.S. Dept. of Commerce, December 2012.

[2] http://nist.gov\publications\drafts\ir7904\draft_nistir_7904.pdf

RFID technologies, and geolocation attributes. Geo-fencing is also about ensuring that the boundaries are not violated; but if they are violated, that appropriate remediations are enforced. Applications supporting geo-fencing allow an administrator to set rules and apply triggers so that when a device, or workload, or data attempts to cross a boundary so defined by the administrator, the action is blocked and appropriate alerts are sent out for further investigation. Many geo-fencing applications employ mashup concepts, such as incorporating Google Earth, thus allowing administrators to define their boundaries using a satellite view of a specific geographic area. Other applications define the boundaries by longitude and latitude or through user-created and web-based maps.

In traditional data centers, workloads and data are pretty static and have a hard binding to the physical information systems on which they reside and execute. However, with virtualization and cloud computing, this is clearly no longer the case. Geolocation can be an attribute for a virtual machine. The ease with which a virtual machine can move has created intense interest in instituting mechanisms to track and control these movements, however. The power and appeal of cloud computing for IT is its agility, efficiency, and mobility of workloads in order to meet the service-level agreements for customers, and also to improve total cost of ownership for service operators. The mobility and agility are possible because of the abstraction and decoupling of the physical hardware from the virtual machines running on top. However, the mobility that allows workloads and data to move in an unrestricted fashion also brings concerns about violating security and privacy policies. Geo-fencing thus becomes an extremely useful capability in cloud computing environments. Geo-fencing usages in cloud computing environments take advantage of the geolocation attribute as described above. (We define and describe geolocation in exhaustive detail in the later sections.) This expanded usage involves attaching geolocation attributes to workloads or data. With the attributes in place, it is possible to create desired geo-fencing policies and set up the associated monitoring and control mechanisms at multiple levels in the cloud infrastructure.

Here are some potential use cases for geo-fencing, in virtualization and cloud computing:

- *Government security requirements.* Many countries and their governments require that data and workloads stay within designated country and geographic boundaries. For instance, certain data may not be allowed to leave the sovereign territory, with exceptions being made for embassies and safe-harbor countries.

- *E-commerce.* Retailers may want to optimize their business processes to improve taxation outcomes—for instance, in the United States, for interstate commerce where tax rates vary by state or to gain special tax benefits, such as hosting sites in export only zone. Geo-fencing allows restrictions where workloads and data are stored in the cloud and provides audit trails detailing where those workloads and data have been. Retail applications go beyond the brick-and-mortar stores when the consumables are digital, such as video, audio, images, software, books, and more. Banking is another regulated industry, and customer data sometimes enjoys greater privileges owing to international agreements.

- *Research.* Companies may restrict what categories of research are carried out in particular geographic locations, so as to be compliant with local regulations or for intellectual property management purposes. For example, stem cell research and pharmacological research fall into this category.

There are many other examples of situations in which geo-fencing is applicable, such as in finance, health care, and other regulated industries. An expansion of the geo-tagging concept is that of asset tagging, whereby the attribute associated with the device or a server is a functional asset descriptor.

Asset Tagging

Geo-tagging can be generalized to be any arbitrary datum about a server. Given a trusted source of information about a server, trusted compute pools with asset tagging enable organizations to enforce running workloads only on trusted servers tagged with specific attributes. For example, an organization might be willing to pay a premium for dedicated trusted servers with bonus points for a capability to segregate workloads by department, each of which may have different policies regarding trusted platforms. The organization can provision an asset tag to each server, indicating the department to which that server is assigned. The organization can then extend its overall trusted computing policy to restrict workload execution to servers carrying a specific asset tag. There are many such potential usage models for asset tagging:

- *SLA-based zoning of data center assets.* This would include tagging compute, storage, and network devices serving specific SLA zones, as in "bronze," "platinum," and "gold." The partitioning can be linked to security, performance, availability, or reliability goals, in any combination.
- *Sarbanes-Oxley audits.* The visibility and verifiability of asset tags augmented by the assurance from hardware-based roots of trust for any Sarbanes-Oxley–related audits can save IT operations a significant amount of time and resources.
- *Workload segregation.* This is useful where tenants request segregation of workloads from other tenants or workloads or workload types.

■ **Note** An asset tag is a geo-tag when the attributes of the tag represent geolocation. For the rest of this chapter, we will use *geo-tag* to represent an asset tag with geolocation attributes. *Asset tagging* and *geo-tagging* are terms used interchangeably, from an architecture and provisioning process perspective.

Trusted Compute Pools Usage with Geo-Tagging

Cloud service providers who implement trusted compute pools (TCP) and their customers are requiring additional boundaries beyond platform trust to assure control of their workloads. A high-priority boundary condition to enforce is one based on the specific physical location of a host, such that workload placement can be:

- Monitored and enforced based on customer policies for boundary controls
- Verified and provided in audit and compliance reports to tenants to meet their internal and regulatory needs for data security reporting

There are a few ways of attaching geolocation attributes to a platform. For instance, geolocation can be arranged through a trusted platform module (TPM) security chip based on a Trusted Computing Group standard. This approach aligns naturally with trusted compute pools as the foundation for use case capabilities requiring established platform trust status and physical location with verification and reporting. That is exactly what trusted compute pools provide. Cloud service providers are expected to extend their current trusted compute pools solutions with trusted location controls to provide additional granularity of control above platform trust.

Trusted compute pools with geo-tagging enable organizations to ensure their workloads are executed only on trusted servers located in authorized geographical areas. For example, as depicted in Figure 5-1, an organization like U.S. government with multiple geographically distributed data centers, might require that certain virtual servers be located in U.S. data centers. Such controls are specified or supported by a growing body of customer requests and regulatory mandates, such as the ability to separate customers or workload types to address region-specific data protection requirements, as defined in FISMA SP800-53 and NIST IR 7409. The controls also support expected needs for eased auditability and verifiability pursuant to compliance mandates.

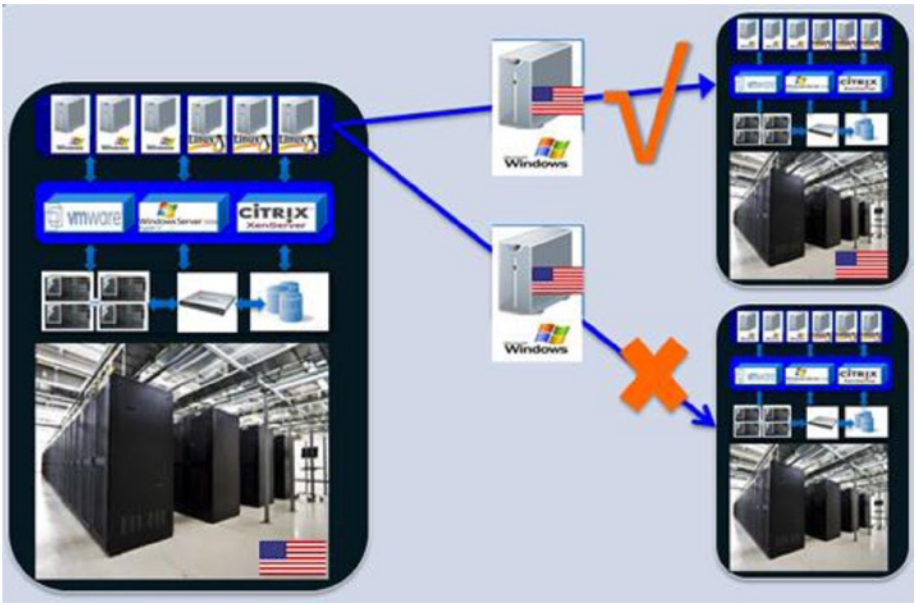

Figure 5-1. *Geolocation and geo-fencing*

NIST, in partnership with industry participants, published an interagency report, NIST IR 7904, documenting trusted compute pool usages with geolocation descriptors, as well as the geo-fencing policy enforcement in multi-tenant cloud computing environments. Figure 5-2 illustrates the IR 7904.

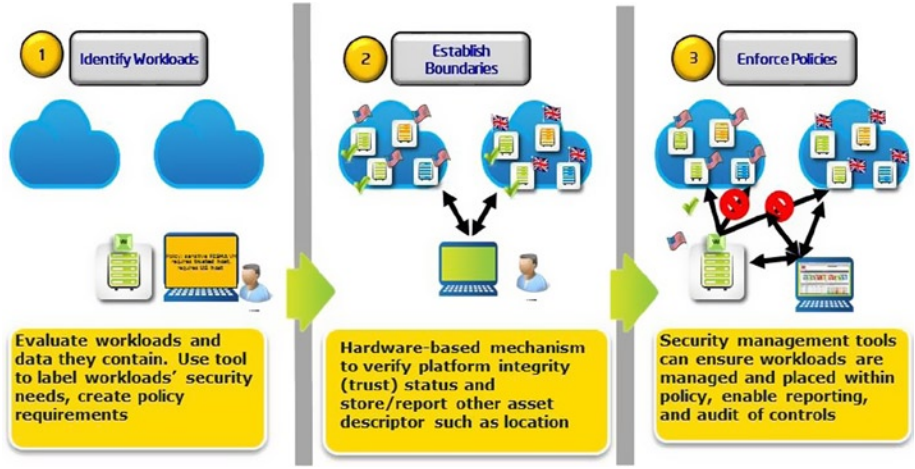

Figure 5-2. *NIST IR 7904 – trusted geolocation in the cloud*

Establishing a trusted compute pool with a trusted geolocation in a cloud comprises three main stages, as shown in Figure 5-3. First, each compute platform must be attested as trustworthy, enabling a safe hypervisor. Second, the cloud system must ensure that workload migration occurs only between trusted resources. And third, trusted geolocation is ensured with continuous monitoring and enforcement of geolocation restrictions. Let's look closer at each of these stages.

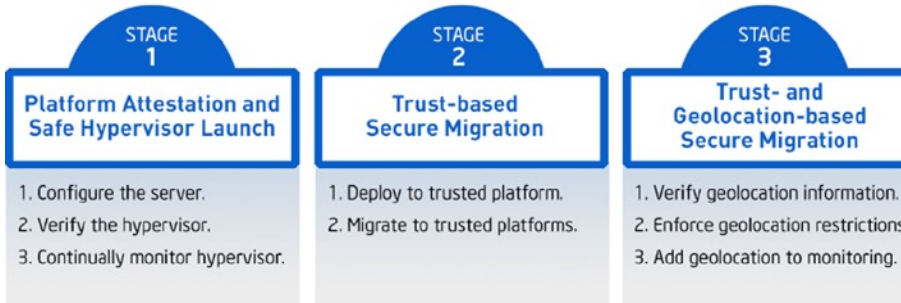

Figure 5-3. *The three stages for establishing a trusted compute pool with trusted geolocation*

Stage 1: Platform Attestation and Safe Hypervisor Launch

This initial stage provides a basic assurance of platform trustworthiness and enables faster detection of security issues. There are three steps to this stage:

1. *Configure the server.* Set up the cloud server platform as being trusted, including configuring the hardware, BIOS, and hypervisor.
2. *Verify the hypervisor.* Before each hypervisor launch, verify the trustworthiness of the cloud server platform set up in the previous step. Remote attestation is the way the integrity of the launch of the platform is verified.
3. *Continually monitor the hypervisor*. During execution, frequently repeat the measurements done in step 2 to continually ensure trustworthiness. These measurements should then become an ongoing part of a continuous monitoring process.

Stage 2: Trust-Based Secure Migration

Ensure that workloads are deployed and then are migrated only among trusted server platforms within the cloud. There are two steps to this stage:

1. *Deploy to trusted platforms.* Apply the verification tests established in stage 1, step 3 and only deploy a workload to those platforms deemed trustworthy.
2. *Migrate to trusted platforms.* Once a workload is deployed, ensure that it migrates only to hosts with comparable trust levels. This is determined by applying the verification tests from stage 1, step 3 on both the workload's current server and the server to migrate the workload to. Migration is allowed only if both servers pass their audits.

Stage 3: Trust- and Geolocation-Based Secure Migration

Build on previous stage by ensuring that workloads migrate only to trusted server platforms while also taking geolocation restrictions into consideration. There are three steps to this stage:

1. *Verify geolocation information.* Ensure that any platform to be included in the trusted geolocation pool has its geolocation set as part of its initial configuration in stage 1, step 1. This is a cryptographic hash within the hardware cryptographic module in BIOS. Ensure that the geolocation information can be verified and audited readily.
2. *Enforce geolocation restrictions.* Add a geolocation check to the pre-deployment and pre-migration verification in stage 2, steps 2 and 3 before deploying or migrating a workload.
3. *Add geolocation to monitoring.* Add geolocation checks to the continuous monitoring put in place in stage 1, step 3 to ensure trustworthiness of the platforms. This process should audit the geolocation of the cloud server platform against geolocation policy restrictions.

Adding Geo-Tagging to the Trusted Compute Pools Solution

As we discussed in the introduction to this chapter, geo-tagging and asset tagging will deliver increased value to trusted compute pools usages in data center operations and for customers. Geo-tagging and asset tagging bring valuable additional security controls to the data center, as well. Supporting geo-tagging and asset tagging, and implementing geo-fencing require some functional changes to the original trusted compute pools architecture that was introduced in Chapter 3. Figure 5-4 provides a summary of these changes, and in the next sections we explain the changes at each layer of the architecture.

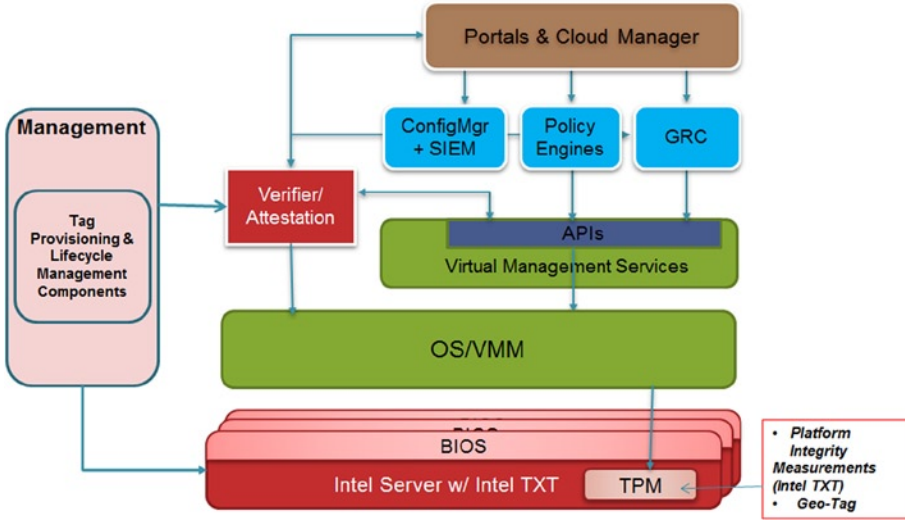

Figure 5-4. *Trusted compute pools solution architecture with geo-tagging*

Hardware Layer (Servers)

There are no changes required at this layer; the trusted platform module (TPM) takes care of secure storage for the geo-tags. Through a secure provisioning process, the geo-tag is provisioned into a nonvolatile index (NVRAM index) in the TPM, and the trusted boot process extends the contents of the specific index into a PCR in the TPM. PCR22 has been selected to capture the geo-tag attributes. As per the TCG client specifications, PCR22 is allocated for OS/VMM use, and in the case of VMWare ESX, Citrix XenServer, open-source Xen, and KVM implementations, it is not used for any other function, hence it was a logical choice to extend the geo-tag attributes. (Geo-tag provisioning and management will be covered in the following sections.) Entities above the stack use the TPMQuote process to fetch this PCR value for attestation and decision making, and this was covered in Chapter 4.

■ **Note** Re the NVRAM index for geo-tagging: For TPM 1.2 compliant devices, the NVRAM index is 20 bytes to accommodate a SHA-1 hash value. The current index used for storing the geo-tag is index 0x40000010, and is created with AUTHWRITE permissions. As TPM 2.0 begins to deploy, the geo-tag index will need to accommodate a SHA-256 hash value of 32 bytes in length. The same NVRAM index cannot be used for the SHA-256 value and hence the solution will require a different index. The trusted boot process (tboot) might require modification for TPM2.0 implementation to extend PCR22 from the new 32-byte index location.

Hypervisor and Operating System Layer

As we discussed in Chapter 3, operating systems and hypervisors participating in a trusted compute pool require servers provisioned with Intel TXT. Tboot is by far the most widely used mechanism to serve as a foundation for software vendors enabling their operating system or hypervisor. The tboot code extends PCR22 from the NVRAM index during the measured boot process. VMware ESX has been supporting tboot extensions to read the NVRAM index and extend PCR22 since ESX 5.1. As of this writing, the open-source tboot code has also been extended to extend PCR22 from the NVRAM index. This is the only incremental change at this layer to support these usages.

Virtualization, Cloud Management, and the Verification and Attestation Layer

To recap, the key functions of this layer are:

- Providing a *secure interface* to the measured launch measurements on each of the servers.
- Providing an *attestation mechanism* to evaluate platform trust and assert its integrity.
- Consuming the *trust information*, essentially helping to identify which platforms are trusted and which ones are not.
- Making use of this information to establish an *enhanced security capability* through policy definition and enforcement linked to platform trust.

The main functional change needed to extend TCP with geo-tagging support involves the attestation capability. The attestation server verifies the platform geo-tag and geolocation by comparing the attributes and the geo-tag certificate against a known-good geo-tag fingerprint for that server or device in addition to evaluating platform trust and verifying the integrity measurements of the launch in the original TCP. The attestation subsystem comprises additional APIs for geo-tag attestation, and the capture and storage of known-good geo-tags for the host. The SAML assertion for the attestation subsystem provided to the requester now includes geolocation assertion. We will dig deeper into this and also explain the additional APIs in Mt. Wilson to accommodate geo-tagging.

The resource scheduler in this layer makes decisions on the placement and migration of virtual machines and workloads. The location policy for data and virtual machines is evaluated and enforced at the security management layer, and the results are provided to the resource scheduler to make security decisions.

■ **Note** The functionality of trusted compute pools (TCP) as described in Chapter 3 has been implemented in OpenStack as scheduler filters. These extensions, and the Horizon dashboard and API extensions to tag Flavors with "Trust" policies, have been part of OpenStack since the Folsom release. As of this writing, a reference implementation demonstrating OpenStack TCP filter extensions to use the geo-tag or asset tag attributes is available. Extensions to Horizon and Flavor attributes are also provided as reference implementations. The expectation is that these will become part of core OpenStack distribution in the near future.

Security Management Layer

Policy managers, security monitoring tools, and compliance and risk management tools make their security decisions based on platform trust and geolocation assertions from the layers below. Policy tools use the geolocation assertions to control the creation, launching, and migration of the workloads and data to carry out geo-fencing policies. Policy management tools need to implement mechanisms to tag virtual machines and data with specific geolocation policies. For instance, the tags identify a virtual machine as *run only on data centers within the continental United States* or as *belongs to the Finance Department*.

The actual mechanisms for policy enforcement depend on how the orchestrator and scheduler software are architected. In OpenStack, policy management is integrated into the orchestrator as pluggable filters. These filters consume the attestation assertion from the attestation service and make decisions to identify and select the appropriate target platforms to instantiate virtual machines. With VMware, a HyTrust Appliance functions as a gateway between VMware VCenter and VMware ESXi hosts. The HyTrust Appliance evaluates the policy against the attestation information, including the geo-tag descriptor for a potential target ESXi host.

The outcome of a policy evaluation is either to proceed with the launch or migration of the virtual machine on the target host, or to deny the request to launch owing to a geolocation policy violation. Policy enforcement and control information is passed on to a security information and event management (SIEM) or governance and to risk compliance (GRC) solutions for reporting and audit compliance. If the solutions used already support trusted compute pool controls, simple extensions will suffice to read, understand, and display the compliance with geo-tagging security controls.

Provisioning and Lifecycle Management for Geo-Tags

The main capabilities needed to support geo-tagging in trusted compute pools are tag provisioning and lifecycle management. The capabilities allow securely creating, selecting, provisioning, and lifecycle management of geo-tags that enables the layers above to make decisions, carry out reporting, and evaluate tags against security controls. The associated process defines the geo-tag workflow lifecycle, covered in the next two sections, including architectural considerations.

Intel Corporation provides reference implementation for tag provisioning and lifecycle management. The reference implementation doesn't dictate what the contents for geo-tags or asset-tags should be. Cloud service providers or enterprise end users have the option of determining the appropriate tag taxonomy for their customers. The lifecycle of geo-tag provisioning and management is covered in the next section.

Geo-Tag Workflow and Lifecycle

The geo-tagging lifecycle consists of seven discrete steps, as depicted in Figure 5-5: tag creation, whitelisting, re-provisioning and deployment, in-validation, validation, attestation, and re-provisioning. Let's go over each.

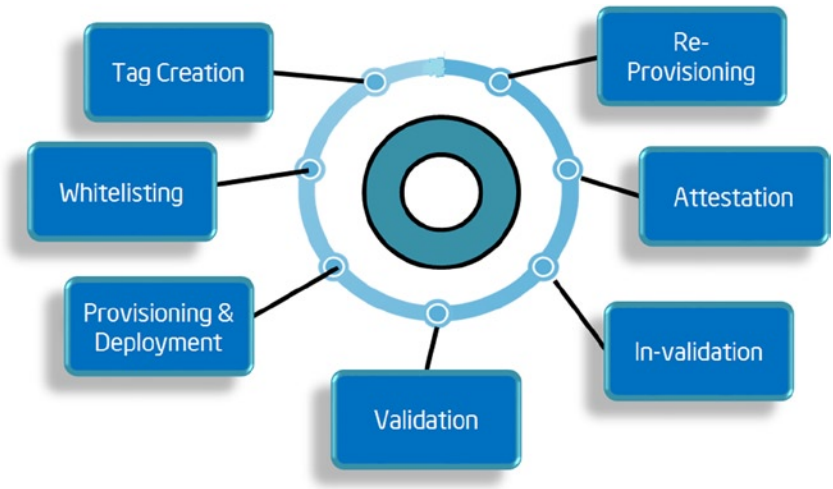

Figure 5-5. *The geo-tagging management lifecycle*

Tag Creation

A tag, as shown in Figure 5-6, is an attribute that has a name and one or more values. The values can be "user-defined," like *united states*, or *san jose* or *Finance*. Values can be "pre-defined," like country or state or postal codes from USG/NIST databases. Values can be dynamic, like latitude/longitude/altitude from a GPS system. The dynamic values would be fetched during the actual provisioning of the tag onto an asset. The tags can be geolocation objects or asset descriptors as well. In this context, an asset is a compute node like a server, end-user device, storage, or network device. The tag creation step involves creating a taxonomy of tags—a set of acceptable name-value pairs applicable to an organization.

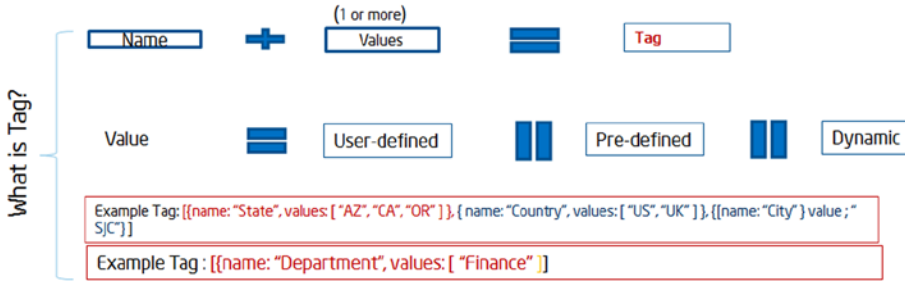

Figure 5-6. *Tags defined*

Tag Whitelisting

Typically, a business analyst at an organization or a suborganization creates this taxonomy of acceptable tags at the corporate level. A subset of tags is then selected for a particular business function. The subset defines the *whitelist* of the tags for that business function, and compliance is evaluated and enforced against that whitelist. A policy creation and definition tool uses this whitelist to associate the geo-tags with the VMs or workloads, and also to enforce the policy.

Tag Provisioning

There are two distinct steps in tag provisioning:

Tag selection

This is the process of selecting one or more tags from the whitelist that would be bound to an asset. In most cases, a selection is applied to many assets. The selection has a name that is a unique descriptor of the purpose of those tags and the list of associated tags. This construct becomes the unit of deployment of these tags onto various computing assets. The binding of this selection to a specific asset (a computing device) is an *asset tag*. To ensure that the tags in the selection are associated with an unique physical asset, the selection is bound to a unique hardware attribute of the asset that is usable as a *universally unique identifier* (UUID), such as a motherboard identifier. As dicussed in the earlier sections, an asset tag that has geolocation attributes is a geo-tag.

To ensure cryptographically secure binding associated to the intended asset, we define the concept of an *asset certificate*. An asset certificate is a document containing a digital signature of the tags in the selection, with the binding to the asset with the UUID. The certificate is digitally signed by a trusted authority and maintained for verification and attestation as X.509 attribute certificate or SAML certificate. A SHA-1 Hash (SHA-2 in the future with TPM2.0) of the asset certificate is what that gets provisioned into a secure location on the asset as the asset tag or a geo-tag (the latter, if the attributes are geolocation attributes). Figure 5-7 illustrates how the asset tag is created from an asset certificate, which in turn is created with the tag selection and the UUID of the asset.

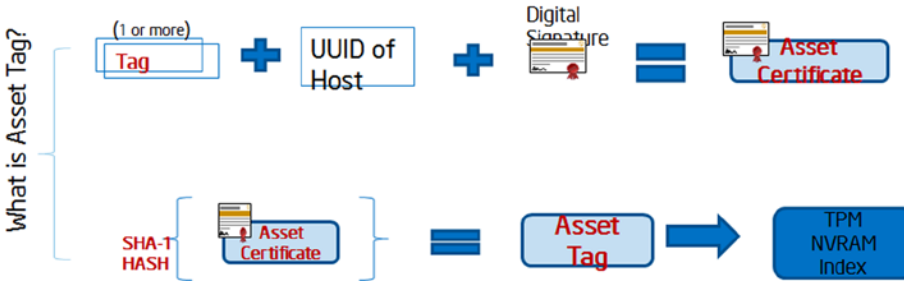

Figure 5-7. *Asset tags*

Tag deployment

This encompasses the secure deployment of that asset tag onto the asset. We recommend using the trusted platform module (TPM) for securely storing the geo-tags and asset tags on the platform, taking advantage of the hardware roots of trust with attestation. Figure 5-8 shows the template of what an asset certificate looks like. A SH1-hash of this is written in the TPM NVRAM index during the provisioning process. At the end of a successful provisioning process, the asset certificate and the geo-tag (the fingerprint) are securely imported into the attestation authority (like a Mt. Wilson) for attestation during policy execution and enforcement.

UUID: Motherboard ID..
Validity:
Owner:
Issue Date:
Tags
- Country: USA
- State: CA, NV
- City: SJC, Fremont

Digital Signature of all of this
Signing Algorithm
Public Key

Figure 5-8. *Asset tag certificate fields*

Figure 5-9 illustrates the tag creation and provisioning steps. It shows the two actors and the functions they perform to define, select, and provision the asset tag and/or geo-tag to the TPM. Tag re-provisioning essentially follows the same process as provisioning. It is triggered by an invalidation event, where the asset tag on the asset is invalidated. (Invalidation is covered in the next section.)

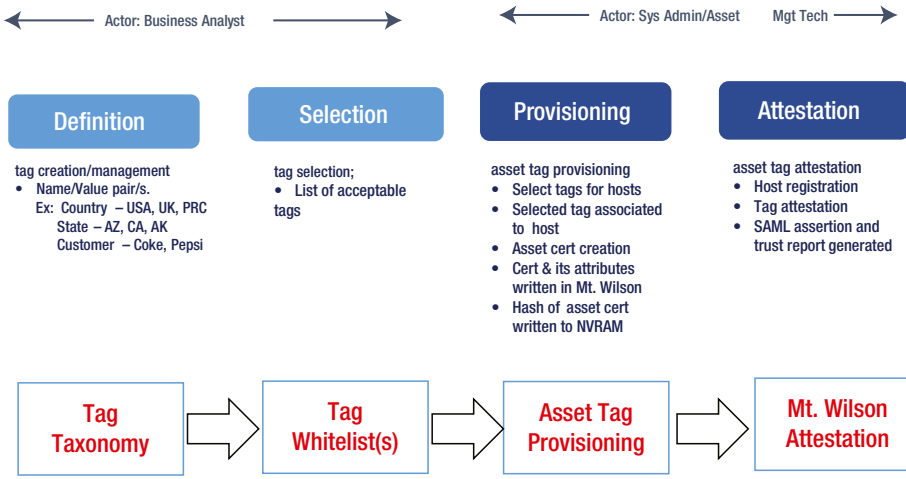

Figure 5-9. *Steps for tag creation and provisioning*

Validation and Invalidation of Asset Tags and Geo-Tags

This is a mandatory step in the geo-tagging lifecycle to prevent misuse and spoofing of the geo-tags, either accidentally or maliciously. Validation can be carried as a manual process, but ideally it should be intelligent, proactive, and automated. Automated processes enable deployment scaling and security automation, offering an extra backstop against provisioning and deployment errors or even malfeasance. Local and remote methods allow automated and auditable validation and invalidation, as well as modification of tags, on individual and groups of assets. Here are some automation mechanisms that have been considered in the development of the reference architecture:

- Heuristic analysis models using external comparison, such as near-neighbor tag analysis, GPS inputs
- Marking geo-tag certificates signed by an unknown authority as untrusted
- Marking expired geo-tags as untrusted and expired
- Marking geo-tags with UUID mismatches as untrusted
- Automated hardware-based mechanisms to monitor power cable connections to the device, or network heartbeat or deadman mechanisms to assess the validity of the geo-tags

Validation and invalidation capabilities would be pretty rudimentary in the initial implementations of the geo-tagging solutions, and they can support one or all of the first four mechanisms listed above. The expectation is that over time the automated hardware-based mechanisms would be broadly supported so the geo-tags become highly tamper resistant and can enable automated compliance with policy controls.

Attestation of Geo-Tags

Attestation of geo-tags involves ensuring that the geo-tag fingerprint that is reported from the server or device is what is expected for that server or device. When a geo-tag is provisioned to the server, it is also stored in the attestation server as the golden fingerprint. During operation of the data center environment, the geo-tag fingerprint as reported by the server is verified against the golden one, and an assertion is generated about the trustability of the geo-tag. The orchestration, policy, and compliance tools use this assertion to make decisions in the cloud. The geo-tag attestation process piggybacks on the platform boot integrity attestation architecture that was covered in Chapter 4. Two new APIs have been added to the attestation authority to address the needs for geo-tagging and asset tagging. These attestation server changes and extensions are covered in the attestation service section later in the chapter.

Architecture for Geo-Tag Provisioning

Figure 5-10 shows an abstract architecture for defining, provisioning, monitoring, and enforcing geo-tags in a trusted compute pools host.

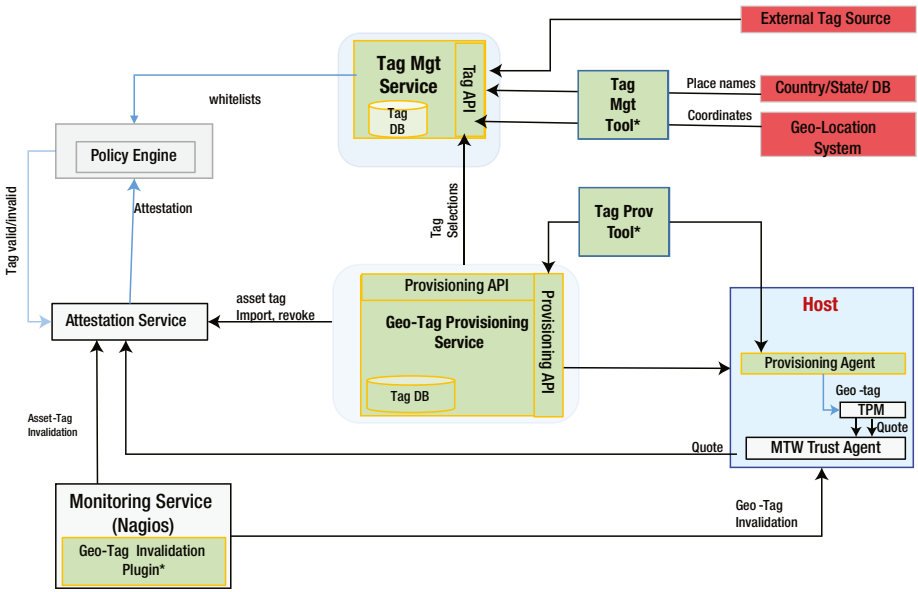

Figure 5-10. *Geo-tag solution architecture*

There are four key components of the solution architecture:

1. Tag provisioning service
2. Tag provisioning agent
3. Tag management service and management tool
4. Attestation service

Let's cover each in sequence.

Tag Provisioning Service

The tag provisioning service implements tag creation—creating asset tag certificates when tags are bound to the UUID of a host—and communicates with the tag provisioning agent on the host to securely deploy and write the geo-tag to the TPM. An asset tag authority (ATA) can be part of the tag provisioning service for automatic approval of certificate requests, or it may reside in external software, polling the tag provisioning service for pending requests and posting certificates for approved requests back to the tag provisioning service. There must be at least one asset tag authority in a working asset tag system. The public key certificates of external authorities must be imported to verify the certificates they create.

The tag provisioning service exposes a set of RESTful APIs for the various entities to interface and integrate with it. Callers are fully authenticated to ensure that legitimate entities are invoking these APIs.

There are two set of APIs for this service:

- *Tag provisioning APIs,* for the tag selection tool and provisioning agent to request and create an asset certificate, and to search existing certificate requests or provisioned certificates.
- *Invalidation APIs,* for monitoring and policy enforcement engines in the data center to invalidate existing asset certificates.

Table 5-1 shows the tag provisioning API. These APIs include functions to create, fetch, delete, search, and revoke asset certificates.

Table 5-1. *RESTful Tag Provisioning APIs*

API Name	Parameters	Description
POST /certificate-requests	{tags[{ uuid|url|name, value }+], authority? }	Create certificate
GET /certificate-requests/{id}	{id, url, tags[ ], status, certificate-url? + }	Read one or more certificate
DELETE /certificate-request/{id}	{id}	Delete certificate

(*continued*)

Table 5-1. (*continued*)

API Name	Parameters	Description
GET /certificate-requests?criteria	{id}	Search certificate id={id} id={id},{id},... tagNameEqualTo={name} tagNameContains={text} tagValueEqualTo={name} tagValueContains={text}
POST /certificate-revocations	{id}	Revoke certificate
GET /certificate-revocations/{id}	{id}	View revoked certificate

Tag Provisioning Agent

The tag provisioning agent provides an API for deploying asset tags to the TPM on the asset. This API is only available on systems where the provisioning agent can run to accept asset tags in push mode. For systems where that is not possible or desirable, the provisioning agent can be activated whenever the administrator needs to provision and deploy an asset tag and request the asset tag from the tag provisioning service in pull mode.

The tag provisioning agent needs authorization to interact with the TPM and write the geo-tag into the NVRAM index. This means it needs the ownership password to acquire ownership of the TPM and write the index. The security of the ownership password, the authentication of the provisioning agent to get access to the ownership password, and the authentication of the provisioning agent with the tag provisioning service is a critical design element of the solution. In the Intel reference implementation, the ownership password is in a configuration file on the host with root access, and the configuration file is encrypted with a symmetric password used by the system administration during provisioning.

Tag Management Service and Management Tool

The tag management service and management tool are primarily required to create the tags—the name-value pairs of the tag taxonomy selected and used to create the asset certificates and the geo-tags and asset tags. These components are an optional part of the geo-tagging architecture; the architecture and workflows do not depend on the existence of these two components. The architecture allows integration of third-party tag-creation tools, such as the HyTrust Appliance. The architecture also provides a well-defined

XML file to codify the tag selection to be used with the provisioning. Provisioning tools can take the file as input to complete the geo-tag provisioning. Alternative tag creation and management tools provide the selected tags in the XML configuration file for the provisioning tools to import and create the asset certificates and the geo-tags with binding to the individual hosts.

The reference tag management service provides the APIs and functionality to store the tag taxonomy and allow other software to access it to create and store the tags. The tag management service provides APIs for creating attribute definitions (the attribute name and possible values for the attribute); for searching the taxonomy for attributes having a specific name or possible value; for managing relationships between attributes; and for managing any local policies associated with the provisioning of attributes.

The relationship between attributes may be hierarchical, such as country-state-city or datacenter-room-aisle-rack, or flat, such as price and location. A policy associated with provisioning the attributes could be that an asset certificate containing the customer attribute *Coca-Cola* cannot also contain the customer attribute *Pepsi* at the same time; or that an asset certificate containing the department attribute *Finance Server* must also contain the country attribute *United States*. Table 5-2 shows the tag management service API in its reference implementation.

Table 5-2. *RESTful Tag Management API*

API Name	Parameters	Description
POST /tags	{ oid?, name, values[ ]? }	Create single or multiple tag definition
POST /tags/{id}/ values	[value+]	Add values to existing tag definition
PUT /tags/{id}/ values	[value+]	Update values for existing tag definition; [ ] empty array deletes all values for existing tag definition
GET /tags/	{id}	Read/load tag contents by ID
GET /tags?criteria	criteria	Search tag definitions Examples: Id = {id}; nameEqualTo{name};nameContains= {text};valueEqualTo={name};valueContains={text}
POST /rdf-triples	{subject, predicate, object}	Create relationship between tags Example: { subject: *Country*, predicate: *contains*, object: *State* }

Attestation Service

The attestation service is an extension of the trust attestation service code-named Mt. Wilson, covered in Chapter 4. These extensions effectively add another plank to the attestation platform providing the geo-tag and asset tag attestation capabilities. That is, the attestation service adds asset tag verification information to its security assertions. It keeps an audit log of asset tag certificates associated with specific compute nodes, and it maintains copies of the asset tag certificates. This allows the attestation service to log not just when an asset tag is updated in an asset but also any changes made to the set of attributes associated with that asset from one asset tag to the next. Thus, the attestation service must apply integrity protection to its repository of trusted asset tag authorities to prevent tampering.

The Mt. Wilson attestation service adds two new APIs that support the geo-tag implementation.

API: importAssetTagCertificate

This API is invoked by the tag provisioning service when a new asset tag certificate is created and is provisioned into the TPM. The certificate is mapped to the host information in Mt. Wilson during the host registration step.

API: revokeAssetTagCertificate

This is also invoked by the tag provisioning services when a geo-tag or asset tag certificate is revoked (expired, invalidated, decommissioned). On the Mt. Wilson side, it is disassociated from the host and is also deprecated in the certificate store.

From the attestation side, the SAML security assertion from a trust attestation request adds one additional assertion section, as shown here. In this example, the security assertion is asserting that the geo-tag or asset tag has been verified for a specific server, host, or device as indicated by the UUID of the host, carrying highlighted attributes (name-value pairs). Note the multiple types of attributes from the tag definitions, geo-tags, and tenant descriptors. This SAML assertion is digitally signed by the Mt. Wilson attestation authority to guarantee the integrity of the assertion. (Chapter 4 covered the attestation components and the SAML assertion contents and its integrity.)

```
<saml2:Attribute Name="Asset_Tag">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
"xs:anyType">attested(true)</saml2:AttributeValue>
      </saml2:Attribute>
      <saml2:Attribute Name="ATAG :Country ">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">US</saml2:AttributeValue>
      </saml2:Attribute>
      <saml2:Attribute Name="ATAG :State">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:type="xs:string">CA</saml2:AttributeValue>
      </saml2:Attribute>
      <saml2:Attribute Name="ATAG :City">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">Folsom</saml2:AttributeValue>
      </saml2:Attribute>
      <saml2:Attribute Name="ATAG :Tenant">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">Coke</saml2:AttributeValue>
      </saml2:Attribute>
      <saml2:Attribute Name="ATAG :Tenant">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">Pepsi</saml2:AttributeValue>
      </saml2:Attribute>
      <saml2:Attribute Name="ATAG :UUID">
        <saml2:AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">e64d9248-59d9-e111-b527-001e67576c61</
saml2:AttributeValue>
      </saml2:Attribute>
```

The first attribute section of the example SAML code above asserts that the geo-tag fingerprint on the host has been verified against the expected/known-good fingerprint in the attestation authority. The next set of attribute sections of the SAML provides the various attributes and the descriptors that are asserted by this SAML certificate. These are the various geo-tags and/or asset tags presented by the host and verified against the attestation authority. The last section in the example is the assertion of the UUID of the host. This SAML certificate is provided to any entity or component that would make decisions about VM and data placement, migration, and access decisions.

Now that we have covered the various architectural components of the geo-tagging architecture, let's look at the tag provisioning models and process.

Geo-Tag Provisioning Process

We envision two models for geo-tag provisioning in virtualized data center environments. As indicated in Table 5-3, depending on the type of operating system or virtual machine monitor, one or both options are available.

Table 5-3. *Geo-Tag Provisioning Model*

Provisioning Mode	KVM	Xen	ESXi	Hyper-V
"push" tags to running host	Yes, requires provisioning agent	Yes, using XenAPI	No	?
PXE Boot	Yes	Yes	Yes	Yes

Push Model

Provisioning under the push model, shown in Figure 5-11, is initiated remotely by a provisioning tool. After mutual authentication between the provisioning agent and the provisioning tool, the geo-tag, which is the SHA-1 hash of the host's asset certificate, is pushed to the running host and the geo-tag is written (or updated) in the NVRAM index. A reboot of the host or server is needed to complete provisioning. This option is available for Xen, KVM, and Citrix XenServer hypervisor environments, but not for VMware. VMWare ESXi takes exclusive ownership of the TPM once it is installed and running, and no other entity can manipulate the TPM thereafter.

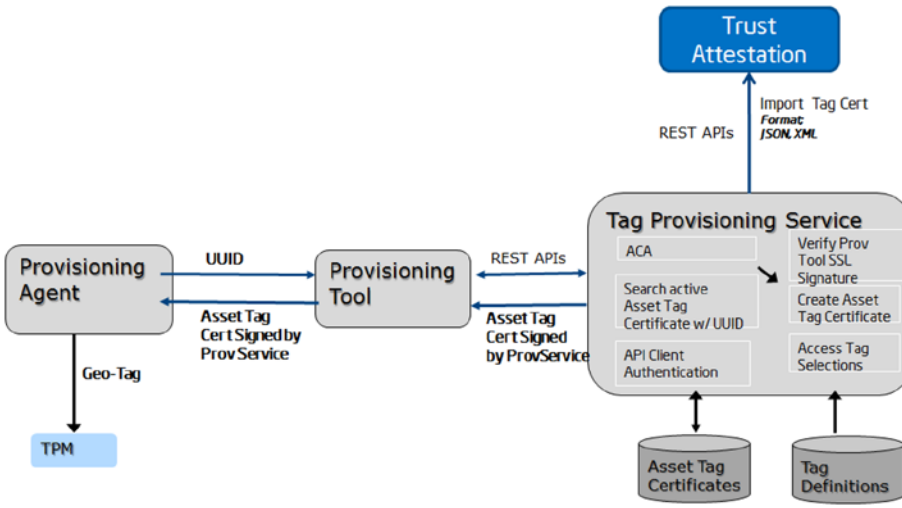

Figure 5-11. *Push mode for geo-tag provisioning*

Pull Model

Pull provisioning, shown in Figure 5-12, is initiated by modifying the boot order on the host and launching a custom PXE boot image to provision the geo-tag. For hosts with VMware ESX, the action needs to be carried out prior to installing or running ESX on the host. The PXE script is built to launch the provisioning agent to interact with the tag

provisioning service for creating the asset certificate and the geo-tag, and their storage to the TPM. The location of the tags is provided to the PXE script to allow the tag provisioning service to create certificates for the geo-tags. The PXE script can then initiate a reboot to start running the hypervisor on the host or start installing the operating system or hypervisor. Figure 5-12 shows the PXE-based pull model for provisioning geo-tags.

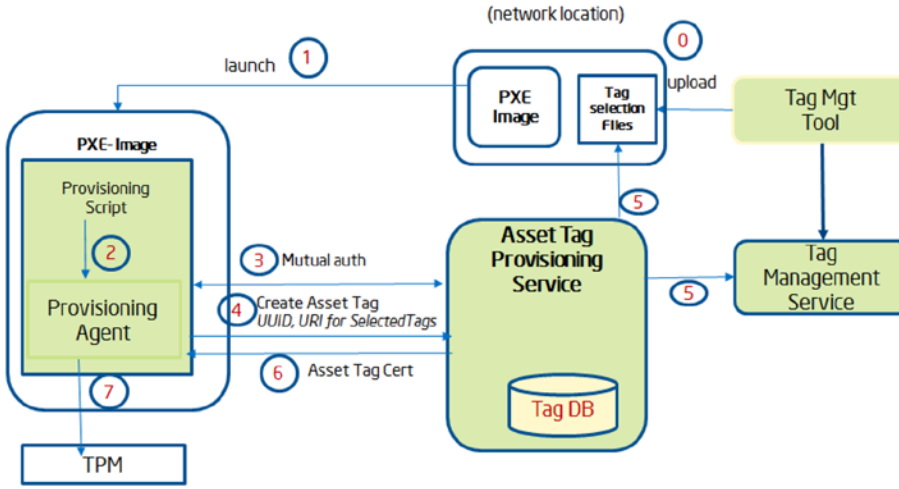

Figure 5-12. *Pull mode for geo-tag provisioning*

Table 5-4 summarizes the key steps of the pull model.

Table 5-4. *Steps for Geo-Tag Provisioning*

Step	Geo-Tag Provisioning with PXE
0	With the tag management tool, the business analyst selects tags to be associated with hosts and uploads them in the form of a pre-defined XML tag specification file format to the network location as the PXE image, or stores them in the repository of the tag management service. This is referred to as "tag selections." The XML is optional encrypted and the keys are provided to the tag provisioning service with appropriate authentication.
1	The system administrator launches the PXE image for provisioning the geo-tag on the targeted host.
2	The PXE image is launched and it then starts the provisioning script, which starts the provisioning agent.
3	The provisioning agent and the tag provisioning service mutually authenticate each other using SSL/TLS certificates.

(*continued*)

Table 5-4. (*continued*)

Step	Geo-Tag Provisioning with PXE
4	The provisioning agent requests the asset tag from the tag provisioning service. The UUID of the host and URI for tag selections is passed to it.
5	Depending on the policy at the tag provisioning service, if a valid and latest asset certificate is available for that host, it is returned to the provisioning agent , or else the provisioning service creates an asset certificate for the host using the URI for the "selected tag" and the UUID of the host.
6	The asset certificate is downloaded to the tag provisioning agent, and the SHA-1 hash of the certificate, which is the asset tag, is created by the provisioning agent. Alternatively, the asset tag is downloaded to the provisioning agent. This depends on implementation of the provisioning service.
7	The provisioning agent writes (or over-writes) the geo-tag to NVRAM index of the TPM, after the appropriate ownership of the TPM has been acquired.

As we have seen in this section, there are two models supported for provisioning geo-tags to assets. The two provisioning models have very different deployment considerations, however. The pull model requires changes to the boot options on the hosts, with modified PXE configuration options to launch the tag PXE boot image. This PXE image is used with iPXE (or equivalent) on a provisioning network to boot to the provisioning image remotely. The model requires the hosts to be on a provisioning network prior to installation, configuration, and launch of the OS/VMM, and they are moved later to the production management network. On the other hand, the push model can happen on the production management network with appropriate authentication of the provisioning tools. Both of these models have a place in a virtual environment and in cloud data centers. The pull model is applicable to all the OS/VMM platforms, but the push model is not available for VMWare ESXi hosts, owing to the way ESXi handles TPMs on the compute platforms.

In the next section, we will look at reference implementation of a complete geo-tag solution, including the definition of tags, selection, and attestation.

Reference Implementation

This section describes a reference implementation highlighting the tag provisioning, management, and attestation steps. The purpose of this implementation is to facilitate knowledge sharing and also to demonstrate the possible visualization of the functionality to partners. The expectation is that ISVs and CSPs will provide their specific implementation for tag provisioning and management in a way that seamlessly integrates with their respective solution environments and interfaces. Key screenshots from the reference implementation are included to illustrate the various steps in the geo-tag solution.

Step 1

This is the tag definition step, where organizations create the tag taxonomy and a tag whitelist to be used for geo-tagging or asset tagging purposes.

Tag creation is the core function of the asset tag service. A tag is an arbitrary name for a classification, which has one or more potential values. For example, a tag named *State* might have values like *California* or *New York*, while a tag named *Department* might have values like *Accounting, Sales,* and so on. As shown in Figure 5-13, a set of tags forms a tag taxonomy. The whitelist for a given domain or function is drawn from this taxonomy, to be provisioned to a host or an asset (generically). For example, you might have a server tagged with a selection like *State: California; Department: Accounting*.

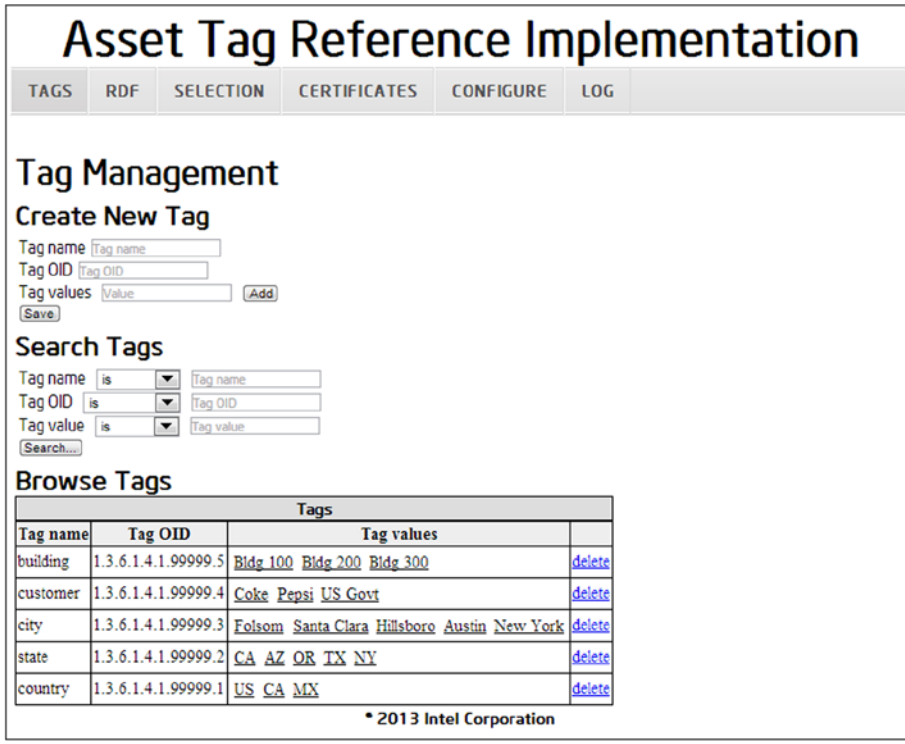

Figure 5-13. *Tag taxonomy*

Step 2

This is the selection step, whereby a specific set of tags for a business function are picked from the whitelist, as shown in Figure 5-14. In this example, the selection is named "default" and has six tags selected that would be provisioned to one or more hosts. As part of the tag provisioning service and API design, automation and scalability have been given deliberate attention. There are well-documented configuration options provided for the tag provisioning service that fully automate the asset certificate creation, geo-tag and/or asset tag generation, provision the tag to the TPM, and register it with Mt. Wilson.

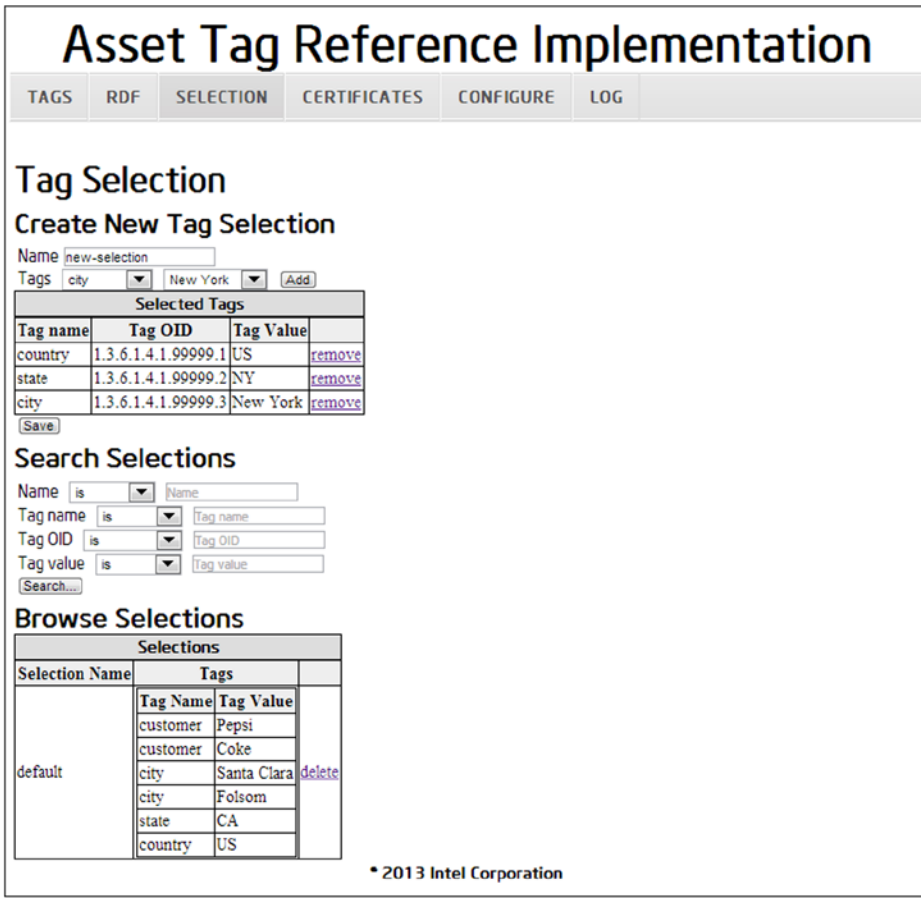

Figure 5-14. *Tag whitelist selection*

Step 3

This is the provisioning step, whereby an asset tag or geo-tag is created by associating one or more of the selection attributes with the asset's UUID, as shown in Figure 5-15; this could be either the push or the pull model for provisioning. As shown in Figure 5-15, the tag provisioning service creates the asset certificate, and the provisioning agent in either of the two models writes the tag to the specific TPM NVRAM index.

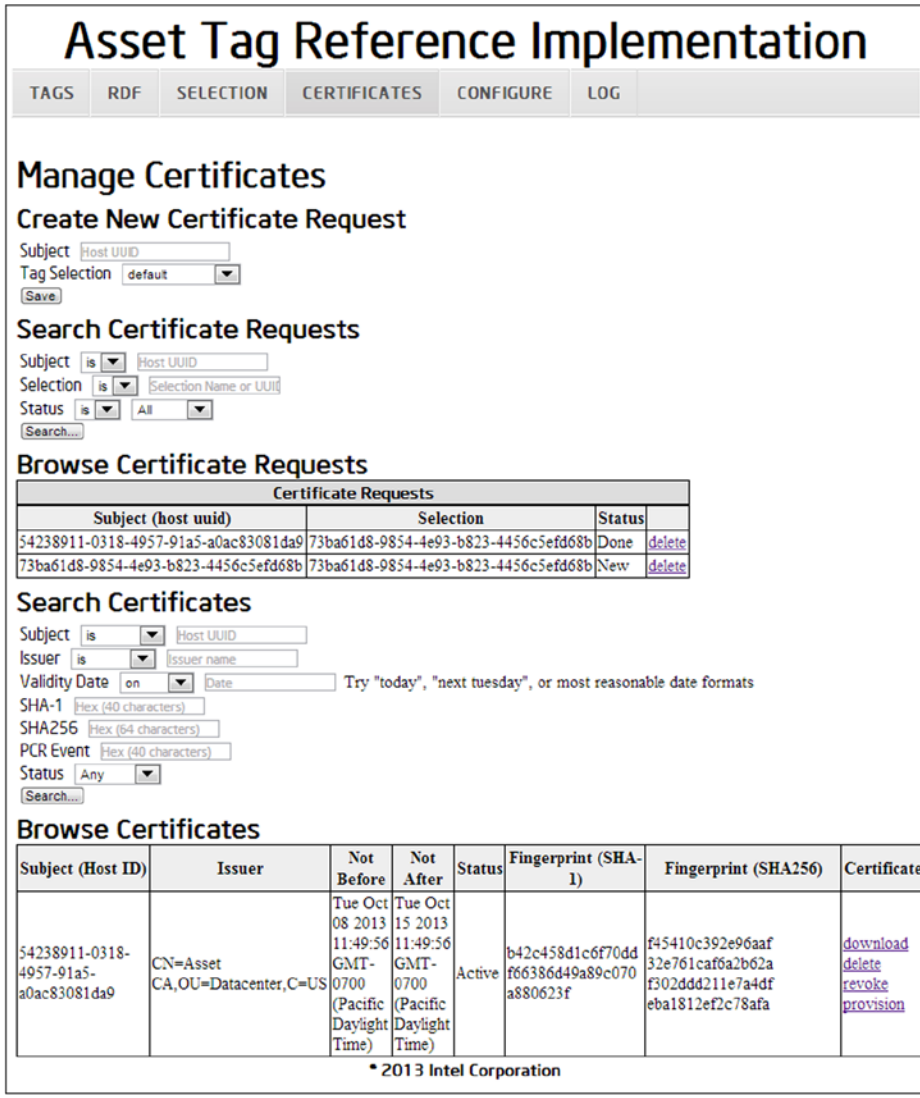

Figure 5-15. *Asset certificate, asset tag and geo-tag creation and provisioning*

Step 4

The last step is to provide visibility and attestation for the tags and certificates, as shown in Figure 5-16. Once the host is registered with Mt. Wilson (after the host has been provisioned with asset tags), the Mt. Wilson trust dashboard displays the tags provisioned to the host and allows Mt. Wilson to attest to the validity of the asset certificate, as well as assert the geo-tag. Essentially, the geo-tag and/or asset tag fingerprint reported by the host is compared and verified to be the same as the expected fingerprint stored in the Mt. Wilson environment. If they are the same, the location attestation is affirmed; if not, it is marked as untrusted. As described in the attestation section, there are multiple reasons for failing the attestation: bad certificate, different fingerprint compared to the expected, and so on. Figure 5-16 also shows the current PCR22 value (where the tag is extended) and the expected value of the PCR22, as well as the SAML assertion that indicates the results of the verification.

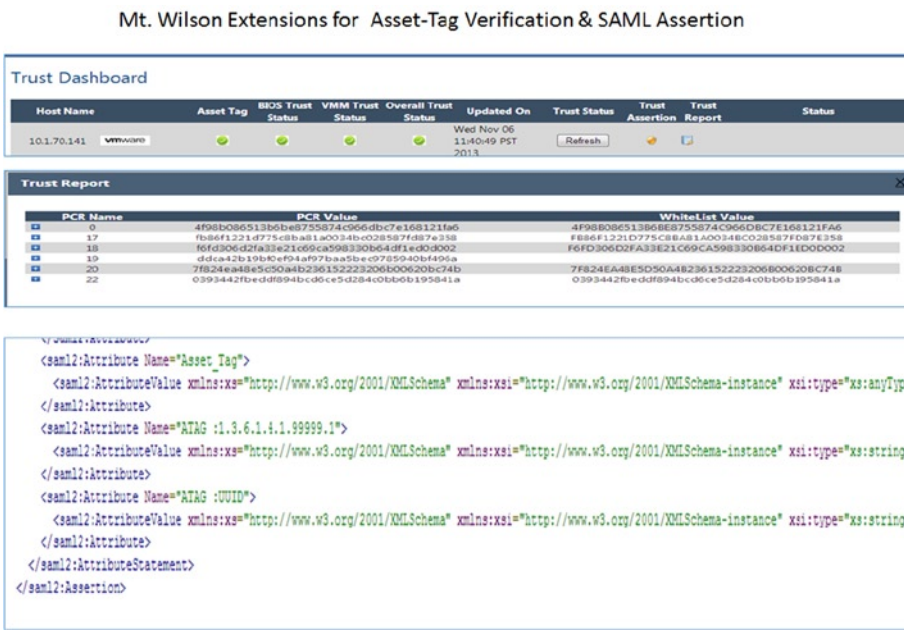

Figure 5-16. *Asset tag verification and example of SAML assertion for asset tag—Mt. Wilson extensions*

As of this writing, the geo-tag provisioning and management solution, as well as the reference implementation, have been provided to many Intel ISVs and CSP partners to enable geo-fencing, workload segregation, and other interesting solutions for cloud computing usage models. Given the significant interest in these uses, the expectation is that many ISVs and CSPs will complete the eventual enablement and integration of these capabilities into their services and product offerings, and they begin to offer them as core services to their customers.

Summary

Boundary control of workloads and data in the cloud through asset tagging and geo-tagging constitutes a critical requirement for organizations as they consider moving their business-critical applications and data to the cloud. Capabilities with trusted compute pools usage models take organizations a long way toward attaining the visibility and transparency they need for confirming the integrity of their cloud infrastructure through a hardware roots of trust. Organizations also gain control of the placement and migration of their workloads. Asset tagging and geo-tagging as described in this chapter are highly complementary to the trusted pool usages, because they enable organizations to securely provision an asset and geolocation descriptors to platforms with desired location properties. Cloud service providers and IT organizations building private clouds can provide the boundary control for workloads and data in their clouds with extensions to the trusted compute pools solution architecture, as described in this chapter. The controls are rooted in hardware, and are auditable and enforceable. The trusted compute pools solution architecture, with tag provisioning and lifecycle management of the constituent services, provides significant additional capabilities to address customer needs. In this chapter we presented a reference architecture and an implementation for these asset tag provisioning and lifecycle management components, with details on tag definition and specification, APIs for tag management and provisioning, and extensions to the Mt. Wilson attestation service to attest the geo-tags.

Geo-fencing is just one and the most obvious many possible usages that can be enabled with a hardware roots of trust-based asset tag or geo-tag information. Usages like SLA-based zoning of data center assets, Sarbanes-Oxley audits, and workload segregation can be enabled by thistagging mechanism, resulting in better compliance and higher quality of service that is rooted in hardware. As the solution stack becomes pervasive in the data center, the expectation is that many such usages of this tagging could be explored to provide proof of locality, of both physical and virtual data center assets.

In the next chapter, we shift gears a bit and focus attention on network security, the synergy of trusted infrastructure, and how it is essential to have hardware-assisted security in network devices to provide network security in the cloud.