

SACM Working Group
Internet-Draft
Intended status: Standards Track
Expires: 10 January 2022

A. Montville
B. Munyan
CIS
9 July 2021

Security Automation and Continuous Monitoring (SACM) Architecture
draft-ietf-sacm-arch-13

Abstract

This document defines an architecture enabling a cooperative Security Automation and Continuous Monitoring (SACM) ecosystem. This work is predicated upon information gleaned from SACM Use Cases and Requirements ([RFC7632] and [RFC8248] respectively), and terminology as found in [I-D.ietf-sacm-terminology].

WORKING GROUP: The source for this draft is maintained in GitHub. Suggested changes should be submitted as pull requests at <https://github.com/sacmwg/ietf-mandm-sacm-arch/>. Instructions are on that page as well.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 10 January 2022.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights

and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
1.1. Requirements notation	4
2. Terms and Definitions	4
3. Architectural Overview	8
3.1. Producer	9
3.2. Consumer	9
3.3. Integration Service	9
3.4. Payload/Message	10
3.5. Payload Categorization	10
3.5.1. Topic-centric	10
3.5.2. Payload-centric	11
3.6. Capabilities	11
3.7. Interaction Categories	12
3.7.1. Broadcast	12
3.7.2. Directed	12
4. SACM Role-based Architecture	13
4.1. Architectural Roles/Components	14
4.1.1. Manager	14
4.1.2. Orchestrator(s)	14
4.1.3. Repositories	15
4.1.4. Integration Service	15
4.2. Downstream Uses	16
4.2.1. Reporting	16
4.2.2. Analytics	16
4.3. Sub-Architectures	16
4.3.1. Collection Sub-Architecture	16
4.3.2. Evaluation Sub-Architecture	19
5. Ecosystem Interactions	21
5.1. Manager	21
5.2. Component Registration	22
5.3. Administrative Interface	23
5.3.1. Capability Advertisement Handshake	23
5.3.2. Health Check	23
5.3.3. Heartbeat	23
5.3.4. Capability-specific Requests	24
5.4. Status Notifications	24
5.5. Component Interactions	24
5.5.1. Initiate Ad-Hoc Collection	24
5.5.2. Coordinate Periodic Collection	24
5.5.3. Coordinate Observational/Event-based Collection	25
5.5.4. Persist Collected Posture Attributes	26

5.5.5.	Initiate Ad-Hoc Evaluation	26
5.5.6.	Coordinate Periodic Evaluation	26
5.5.7.	Coordinate Change-based Evaluation	27
5.5.8.	Queries	27
6.	Operations	27
6.1.	Component Registration	27
6.1.1.	Request Payload	28
6.1.2.	Request Processing	28
6.1.3.	Response Payload	29
6.1.4.	Response Processing	29
6.2.	Administrative Interface	29
6.2.1.	Capability Advertisement Handshake	29
6.2.2.	Health Check	31
6.2.3.	Heartbeat	32
6.3.	Status Notification	33
6.3.1.	Request Payload	34
6.3.2.	Request Processing	34
6.3.3.	Response Payload	34
6.3.4.	Response Processing	34
6.4.	Initiate Ad-Hoc Collection	34
6.4.1.	SACM Producer to Orchestrator	36
6.4.2.	Orchestrator to Posture Collection Service	36
6.4.3.	Posture Collection Service to Posture Attribute Repository	38
6.5.	Initiate Ad-Hoc Evaluation	39
7.	Privacy Considerations	39
8.	Security Considerations	39
9.	IANA Considerations	39
9.1.	Component Types	39
9.2.	Component Capabilities	40
9.2.1.	Heartbeat	40
9.2.2.	Status Notification (Publish)	40
9.2.3.	Status Notification (Subscribe)	40
10.	References	40
10.1.	Normative References	40
10.2.	Informative References	41
Appendix A.	Security Domain Workflows	43
A.1.	IT Asset Management	43
A.1.1.	Components, Capabilities and Workflow(s)	44
A.2.	Vulnerability Management	44
A.2.1.	Components, Capabilities and Workflow(s)	45
A.3.	Configuration Management	46
A.3.1.	Components, Capabilities and Workflow(s)	47
Authors' Addresses	49

1. Introduction

The purpose of this draft is to define an architectural approach for a SACM Domain, based on the spirit of use cases found in [RFC7632] and requirements found in [RFC8248]. This approach gains the most advantage by supporting a variety of collection systems, and intends to enable a cooperative ecosystem of tools from disparate sources with minimal operator configuration.

1.1. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119, BCP 14 [RFC2119].

2. Terms and Definitions

Assessment: Defined in [RFC5209] as "the process of collecting posture for a set of capabilities on the endpoint (e.g., host-based firewall) such that the appropriate validators may evaluate the posture against compliance policy."

Asset: Is a system resource, as defined in [RFC4949], that may be composed of other assets.

Examples of Assets include: Endpoints, Software, Guidance, or X.509 public key certificates. An asset is not necessarily owned by an organization.

Asset Management: The IT process by which assets are provisioned, updated, maintained and deprecated.

Attribute: Is a data element, as defined in [RFC5209], that is atomic.

In the context of SACM, attributes are "atomic" information elements and an equivalent to attribute-value-pairs. Attributes can be components of Subjects.

Capability: A set of features that are available from a SACM Component.

See also "capability" in [I-D.ietf-i2nsf-terminology].

Collector: A piece of software that acquires information about one or more target endpoints by conducting collection tasks.

A collector can be distributed across multiple endpoints, e.g. across a target endpoint and a SACM component. The separate parts of the collector can communicate with a specialized protocol, such as PA-TNC [RFC5792]. At least one part of a distributed collector has to take on the role of a provider of information by providing SACM interfaces to propagate capabilities and to provide SACM content in the form of collection results.

Configuration: A non-volatile subset of the endpoint attributes of a endpoint that is intended to be unaffected by a normal reboot-cycle.

Configuration is a type of imperative guidance that is stored in files (files dedicated to contain configuration and/ or files that are software components), directly on block devices, or on specific hardware components that can be accessed via corresponding software components. Modification of configuration can be conducted manually or automatically via management (plane) interfaces that support management protocols, such as SNMP or WMI. A change of configuration can occur during both run-time and down-time of an endpoint. It is common practice to scheduled a change of configuration during or directly after the completion of a boot-cycle via corresponding software components located on the target endpoint itself.

Consumer: A SACM Role that requires a SACM Component to include SACM Functions enabling it to receive information from other SACM Components.

Endpoint: Defined in [RFC5209] as "any computing device that can be connected to a network."

Additional Information - The [RFC5209] definition continues, "Such devices normally are associated with a particular link layer address before joining the network and potentially an IP address once on the network. This includes: laptops, desktops, servers, cell phones, or any device that may have an IP address."

To further clarify the [RFC5209] definition, an endpoint is any physical or virtual device that may have a network address. Note that, network infrastructure devices (e.g. switches, routers, firewalls), which fit the definition, are also considered to be endpoints within this document.

Physical endpoints are always composites that are composed of

hardware components and software components. Virtual endpoints are composed entirely of software components and rely on software components that provide functions equivalent to hardware components.

The SACM architecture differentiates two essential categories of endpoints: Endpoints whose security posture is intended to be assessed (target endpoints) and endpoints that are specifically excluded from endpoint posture assessment (excluded endpoints).

Based on the definition of an asset, an endpoint is a type of asset.

Endpoint Attribute: Is a discreet endpoint characteristic that is computably observable.

Endpoint Attributes typically constitute Attributes that can be bundled into Subject (e.g. information about a specific network interface can be represented via a set of multiple AVP).

Endpoint Characteristics: The state, configuration and composition of the software components and (virtual) hardware components a target endpoint is composed of, including observable behavior, e.g. sys-calls, log-files, or PDU emission on a network.

In SACM work-flows, (Target) Endpoint Characteristics are represented via Information Elements.

Posture: Defined in [RFC5209] as "configuration and/or status of hardware or software on an endpoint as it pertains to an organization's security policy."

This term is used within the scope of SACM to represent the configuration and state information that is collected from a target endpoint in the form of endpoint attributes (e.g. software/hardware inventory, configuration settings, dynamically assigned addresses). This information may constitute one or more posture attributes.

Posture Attributes: Defined in [RFC5209] as "attributes describing the configuration or status (posture) of a feature of the endpoint. A Posture Attribute represents a single property of an observed state. For example, a Posture Attribute might describe the version of the operating system installed on the system."

Within this document this term represents a specific assertion

about endpoint configuration or state (e.g. configuration setting, installed software, hardware) represented via endpoint attributes. The phrase "features of the endpoint" highlighted above refers to installed software or software components.

Provider: A provider is a SACM role assigned to a SACM component that provides role-specific functions to provide information to other SACM components.

Repository: A repository is a controller that contains functions to consume, store and provide information of a particular kind.

Such information is typically data transported on the data plane, but potentially also data and metadata from the control and management plane. A single repository may provide the functions of more than one specific repository type (i.e. configuration baseline repository, assessment results repository, etc.)

Security Automation: The process of which security alerts can be automated through the use of different components to monitor, analyze and assess endpoints and network traffic for the purposes of detecting misconfigurations, misbehaviors or threats.

Security Automation is intended to identify target endpoints that cannot be trusted (see "trusted" in [RFC4949]). This goal is achieved by creating and processing evidence (assessment statements) that a target endpoint is not a trusted system [RFC4949].

SIEM: TBD

SOAR: TBD

State: A volatile set of endpoint attributes of a (target) endpoint that is affected by a reboot-cycle.

Local state is created by the interaction of components with other components via the control plane, via processing data plane payload, or via the functional properties of local hardware and software components. Dynamic configuration (e.g. IP address distributed dynamically via an address distribution and management services, such as DHCP) is considered state that is the result of the interaction with another component (e.g. provided by a DHCP server with a specific configuration).

Target Endpoint: Is an endpoint that is under assessment at some point in, or region of, time.

Every endpoint that is not specifically designated as an excluded endpoint is a target endpoint. A target endpoint is not part of a SACM domain unless it contains a SACM component (e.g. a SACM component that publishes collection results coming from an internal collector).

A target endpoint is similar to a device that is a Target of Evaluation (TOE) as defined in Common Criteria and as referenced by [RFC4949].

Vulnerability Assessment: An assessment specifically tailored to determining whether a set of endpoints is vulnerable according to the information contained in the vulnerability description information.

Workflow: A workflow is a modular composition of tasks that can contain loops, conditionals, multiple starting points and multiple endpoints.

The most prominent workflow in SACM is the assessment workflow.

-->

3. Architectural Overview

The generic approach proposed herein recognizes the need to obtain information from existing and future state collection systems, and makes every attempt to respect [RFC7632] and [RFC8248]. At the foundation of any architecture are entities, or components, that need to communicate. They communicate by sharing information, where, in a given flow, one or more components are consumers of information and one or more components are providers of information. Different roles within a cooperative ecosystem may act as both Producers and Consumers of SACM-relevant information.

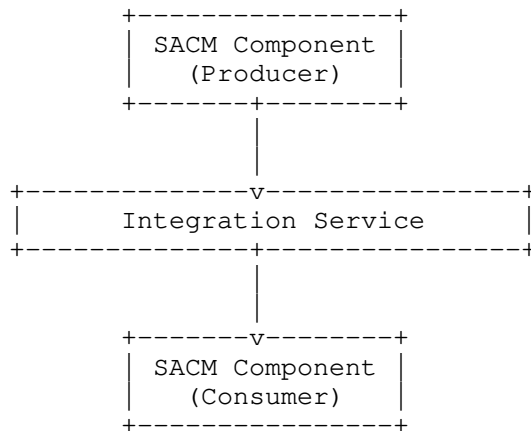


Figure 1: Basic Architectural Structure

3.1. Producer

A Producer can be described as an abstraction that refers to an entity capable of sending SACM-relevant information to one or many Consumers. In general, information (a "payload") is produced to a particular topic, subscribed to by one or more Consumers. Producers need not be concerned about any specifics of the payload it is providing to a given topic. A Producer may, for example, publish posture collection instructions to collector topics.

3.2. Consumer

A Consumer can be described as an abstraction that refers to an entity capable of receiving SACM-relevant information from one or many Producers. A Consumer acts as a subscriber to a given topic (or set of topics), enabling it to receive event notifications when a Producer provides a payload to that topic or topics. Consumers receive payloads and act upon them according to their capabilities. A Consumer may, for example, subscribe to a posture collection topic to receive and act upon, collection instructions.

3.3. Integration Service

The Integration Service acts as the broker between Producers and Consumers; acting as the destination for Producers to publish payloads, and as the source for Consumers subscribing to those payloads.

SACM Components are intended to interact with other SACM Components. These interactions can be thought of, at the architectural level, as the combination of interfaces with their supported operations. Each interaction will convey a classified payload of information. This classification of payload information allows Consumers to subscribe to only the classifications to which they are capable of handling. The payload information should contain subdomain-specific characteristics and/or instructions.

3.4. Payload/Message

The payload (sometimes referred to as a "message" or "message payload") is the unit of data involved in any given interaction between two SACM components. The payload MAY be used to convey the semantic meaning of the operation to be performed. Protocols such as [RFC6120] achieves this meaning through XML namespace identification within a "<message/>" or "<iq/>" stanza. Topic-centric protocols such as [MQTT] convey the meaning of payloads through topic naming techniques. Both methods require connected components to verify message payloads according to their respective capabilities.

With respect to the Integration Service, the payload is simply an array of bytes, so the data contained within it is not required to convey a specific format or meaning to the Integration Service. The serialization of the payload combined with the payload categorization provides meaning within the SACM context.

3.5. Payload Categorization

Within the SACM ecosystem, categorization of payloads and their transport provide the context through which various capabilities are achieved. Two types of payload categorization can be described.

3.5.1. Topic-centric

Topic-centric payload categorization allows for a broad spectrum of payloads by characterizing those payloads through the Integration Service topic. In this categorization, the topic name becomes a label attached to the payload to which the Integration Service matches against known subscriptions. The topic becomes the operational context for the payload. Topic-centric categorization allows for any payload to be sent to any topic, but requires that SACM consumers parse the payloads to determine whether or not they have the capability to act on those payloads.

When interacting using a topic-centric payload categorization, topic naming conventions SHOULD provide an adequate amount of information to be deterministic regarding the purpose of the interaction. For

example, a topic named `"/notification/collection/oval"` would indicate that (a) the topic is a broadcast/notification (publish/subscribe) topic, (b) subscribers to this topic are performing a "collection" action, and (c) the payloads published to the topic are represented using the OVAL serialization format.

3.5.2. Payload-centric

Payload-centric categorization encapsulates the intent of an interaction within the message payload itself, using an identifying token, tag, or namespace identifier. This method allows for the limitation of message types, and therefore increases the extensibility of message payloads.

Payload-centric categorization allows for modularization and specification of extensions, and for plugin-based support of capabilities based the categorization. XMPP is an example of utilization of payload-centric categorization, allowing only three distinct "stanzas" ("`<message/>`", "`<presence/>`", and "`<iq/>`"), using payloads defined by the various extension protocols maintained by the XMPP standards foundation.

3.6. Capabilities

SACM components interact with each other based on their capacity to perform specific actions. In advertising its capabilities, a SACM component indicates its competence to understand message payloads, perform any payload translation or normalization, and act upon that message. For example, an Orchestration component receives a message to initiate posture attribute collection. The Orchestrator may then normalize those instructions to a particular collection system's serialization. The normalized instructions are then published to the Integration Service, notifying the appropriate subscribers.

Capabilities are described using Uniform Resource Names (URNs), which will be maintained and enhanced via IANA tables (IANA Considerations). Using topic-centric categorization of message payloads, capability URNs SHOULD be associated with Integration Service topics to which publishers, subscribers, and service handlers, will interact. Topic naming conventions are considered implementation details and are not considered for standardization. Given a payload-centric categorization of message payloads, capability URNs SHOULD be used as the identifying token, tag, or namespace in order to distinguish specific payloads.

3.7. Interaction Categories

Two categories of interactions SHOULD be supported by the Integration Service: broadcast and directed. Broadcast interactions are asynchronous by default, and directed interactions may be invoked either synchronously or asynchronously.

3.7.1. Broadcast

A broadcast interaction, commonly referred to as publish/subscribe, allows for a wider distribution of a message payload. When a payload is published to the Integration Service, all subscribers to that payload are alerted and may consume the message payload. This category of interaction can also be described as a "unicast" interaction when only a single subscriber exists. An example of a broadcast interaction could be to publish Linux OVAL objects to a posture collection topic. Subscribing consumers receive the notification, and proceed to collect endpoint configuration posture based on the supplied message payload.

3.7.2. Directed

The intent of a directed interaction is to enable point-to-point communications between a producer and consumer, through the standard interfaces provided by the Integration Service. The provider component indicates which consumer is intended to receive the payload, and the Integration Service routes the payload directly to that consumer. Two "styles" of directed interaction exist, differing only by the response from the consumer.

3.7.2.1. Synchronous

Synchronous, request/response style interaction requires that the requesting component block and wait for the receiving component to respond, or to time out when that response is delayed past a given time threshold. A synchronous interaction example may be querying a CMDB for posture attribute information in order to perform an evaluation.

3.7.2.2. Asynchronous

An asynchronous interaction involves the payload producer directing the message to a consumer, but not blocking or waiting for an immediate response. This style of interaction allows the producer to continue on to other activities without the need to wait for responses. This style is particularly useful when the interaction payload invokes a potentially long-running task, such as data collection, report generation, or policy evaluation. The receiving

component may reply later via callbacks or further interactions, but it is not mandatory.

4. SACM Role-based Architecture

Within the cooperative SACM ecosystem, a number of roles act in coordination to provide relevant policy/guidance, perform data collection, storage, evaluation, and support downstream analytics and reporting.

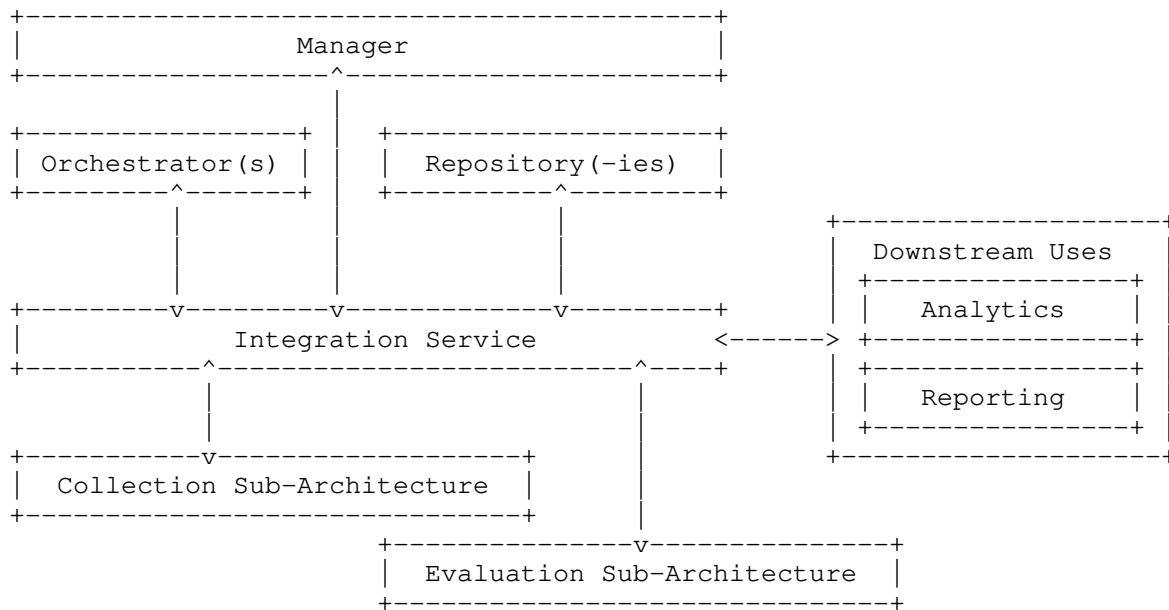


Figure 2: Notional Role-based Architecture

As shown in Figure 2, the SACM role-based architecture consists of some basic SACM Components communicating using an integration service. The integration service is expected to maximally align with the requirements described in [RFC8248], which means that the integration service will support brokered (i.e. point-to-point) and proxied data exchange.

4.1. Architectural Roles/Components

This document suggests a variety of players in a cooperative ecosystem; known as SACM Components. SACM Components may be composed of other SACM Components, and each SACM Component plays one, or more, of several roles relevant to the ecosystem. Roles may act as providers of information, consumers of information, or both provider and consumer. Figure 2 depicts a number of SACM components which are architecturally significant and therefore warrant discussion and clarification. Each role depicted in Figure 2 represents the interface to the component(s) fulfilling that role, not necessarily any specific implementation. For example, the "Repository" figure represents the interface to persistent storage, and not any particular persistent storage mechanism.

4.1.1. Manager

The Manager acts as the control plane for the SACM ecosystem; a sort of high level component capable of coordinating the actions, notifications, and events between components. The manager controls the administrative interfaces with the various components of the ecosystem, acting as the central point to which all other components will register and advertise their capabilities. It is the responsibility of the manager to control a component's access to the ecosystem, maintain an inventory of components attached to the ecosystem, and to initiate the various workflows involved in the collection and/or evaluation of posture attributes.

The manager should maintain the master set of capabilities that can be supported within the ecosystem. These are the various collection, evaluation, and persistence capabilities with which components may register. The manager MAY be responsible for assigning topics for each of the capabilities that are supported, as registering components subsequently subscribe to, or configure service handlers for, those topics.

The manager may act as the user interface to the ecosystem, providing user dashboards, inventories, component management, or operational controls within the boundary of responsibility.

4.1.2. Orchestrator(s)

Orchestration components provide the manager with resources for delegating work across the SACM ecosystem. Orchestrators are responsible for receiving messages from the manager, e.g. posture attribute collection instructions, and routing those messages to the appropriate "actions". For example, an orchestrator may support the capability of translating posture collection instructions using the

Open Vulnerability and Assessment Language (OVAL) and providing those instructions to OVAL collectors. An orchestrator may support the capability of initiating policy evaluation. Where the Manager is configured to ask a particular set of questions, those questions are delegated to Orchestrators, who are then capable of asking those questions using specific dialects.

4.1.3. Repositories

Figure 2 only includes a single reference to "Repository(-ies)", but in practice, a number of separate data repositories may exist, including posture attribute repositories, policy repositories, local vulnerability definition data repositories, and state assessment results repositories. The diagrammed notion of a repository within the SACM context represents an interface in which payloads are provided (based on the capabilities of the producer), normalized, and persisted.

These data repositories may exist separately or together in a single representation, and the design of these repositories may be as distinct as their intended purpose, such as the use of relational database management systems (RDBMS), filesystem-based storage, or graph/map implementations. Each implementation of a SACM repository should focus on the relationships between data elements and implement the SACM information and data model(s).

4.1.4. Integration Service

If each SACM component represents a set of capabilities, then the Integration Service represents the "fabric" by which SACM components are woven together. The Integration Service acts as a message broker, combining a set of common message categories and infrastructure to allow SACM components to communicate using a shared set of interfaces. The Integration Service's brokering capabilities enable the exchange of various information payloads, orchestration of component capabilities, message routing and reliable delivery. The Integration Service minimizes the dependencies from one system to another through the loose coupling of applications through messaging. SACM components will "attach" to the Integration Service either through native support for the integration implementation, or through the use of "adapters" which provide a proxied attachment.

The Integration Service should provide mechanisms for both synchronous and asynchronous request/response-style messaging, and a publish/subscribe mechanism to implement an event-based architecture. It is the responsibility of the Integration Service to coordinate and manage the sending and receiving of messages. The Integration Service should allow components to directly connect and produce or

consume messages, or connect via message translators which can act as a proxy, transforming messages from a component format to one implementing a SACM data model.

The Integration Service MUST provide routing capabilities for payloads between producers and consumers. The Integration Service MAY provide further capabilities within the payload delivery pipeline. Examples of these capabilities include, but are not limited to, intermediate processing, message transformation, type conversion, validation, or other enterprise integration patterns.

4.2. Downstream Uses

As depicted by Figure 2, a number of downstream uses exist in the cooperative ecosystem. Each notional SACM component represents distinct sub-architectures which will exchange information via the integration services, using interactions described in this draft.

4.2.1. Reporting

The Reporting component represents capabilities outside of the SACM architecture scope dealing with the query and retrieval of collected posture attribute information, evaluation results, etc. in various display formats that are useful to a wide range of stakeholders.

4.2.2. Analytics

The Analytics component represents capabilities outside of the SACM architecture scope dealing with the discovery, interpretation, and communication of any meaningful patterns of data in order to inform effective decision making within the organization.

4.3. Sub-Architectures

Figure 2 shows two components representing sub-architectural roles involved in a cooperative ecosystem of SACM components for the purpose of posture assessment: Collection and Evaluation.

4.3.1. Collection Sub-Architecture

The Collection sub-architecture is, in a SACM context, the mechanism by which posture attributes are collected from applicable endpoints and persisted to a repository, such as a configuration management database (CMDB). Control plane functions initiated by the Manager will coordinate the necessary orchestration components, who will choreograph endpoint data collection via defined interactions, using the Integration Service as a message broker. Instructions to perform endpoint data collection are directed to a Posture Collection Service

capable of performing collection activities utilizing any number of protocols, such as SNMP, NETCONF/RESTCONF, SCAP, SSH, WinRM, packet capture, or host-based. Instructions are orchestrated with the appropriate Posture Collection Services using serializations supported according to the collector's capabilities.

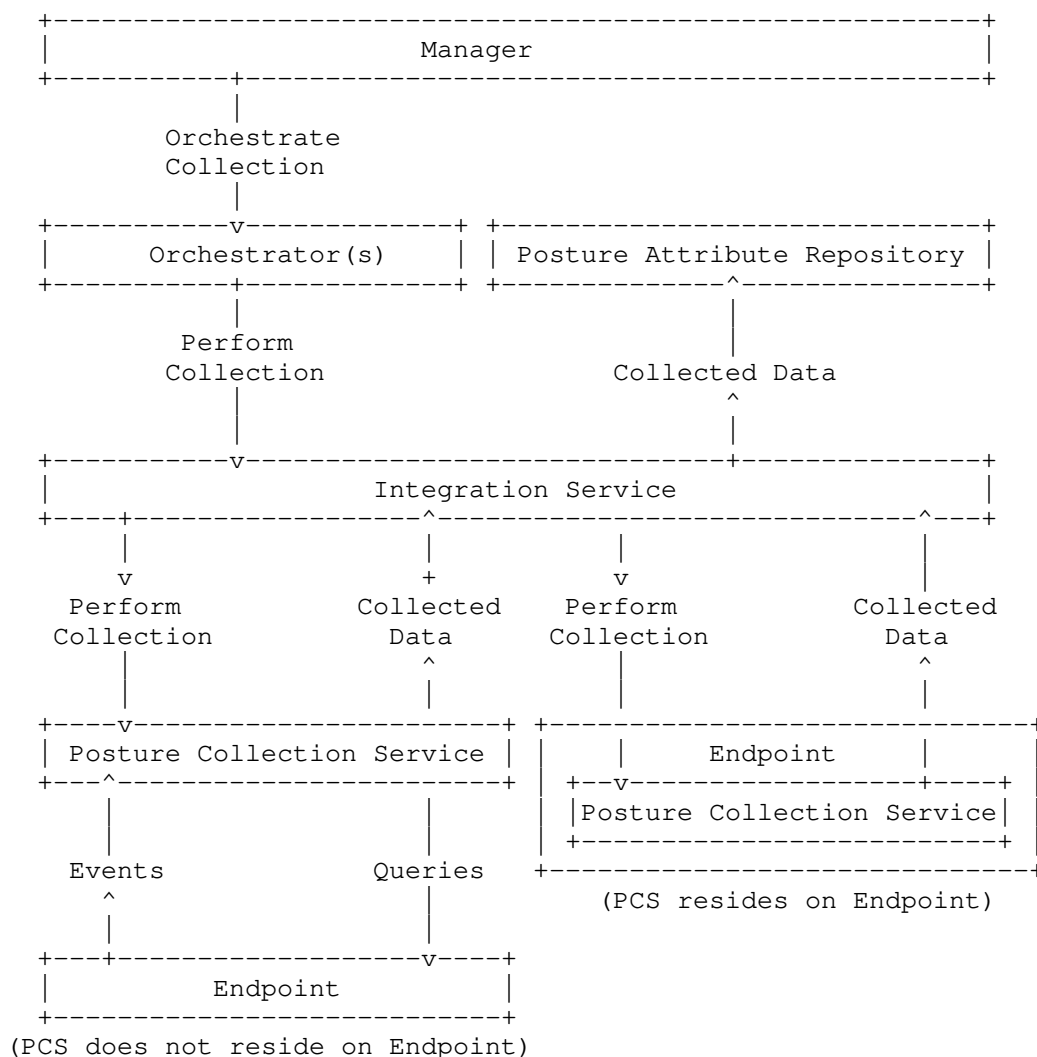


Figure 3: Decomposed Collection Sub-Architecture

4.3.1.1. Posture Collection Service

The Posture Collection Service (PCS) is a SACM component responsible for the collection of posture attributes from an endpoint or set of endpoints. A single PCS MAY be responsible for management of posture attribute collection from many endpoints. The PCS will interact with the Integration Service to receive collection instructions, and to provide collected posture attributes for persistence to one or more Posture Attribute Repositories. Collection instructions may be supplied in a variety of forms, including subscription to a publish/subscribe topic to which the Integration Service has published instructions, or via request/response-style messaging (either synchronous or asynchronous).

Four classifications of posture collections MAY be supported.

4.3.1.1.1. Ad-Hoc

Ad-Hoc collection is defined as a single collection of posture attributes, collected at a particular time. An example of ad-hoc collection is the single collection of a specific registry key.

4.3.1.1.2. Continuous/Scheduled

Continuous/Scheduled collection is defined as the ongoing, periodic collection of posture attributes. An example of scheduled collection is the collection of a specific registry key value every day at a given time.

4.3.1.1.3. Observational

This classification of collection is triggered by the observation, external to an endpoint, of information asserting posture attribute values for that endpoint. An example of observational collection is examination of netflow data for particular packet captures and/or specific information within those captures.

4.3.1.1.4. Event-based

Event-based collection may be triggered either internally or externally to the endpoint. Internal event-based collection is triggered when a posture attribute of interest is added, removed, or modified on an endpoint. This modification indicates a change in the current state of the endpoint, potentially affecting its adherence to some defined policy. Modification of the endpoint's minimum password length is an example of an attribute change which could trigger collection.

External event-based collection can be described as a collector being subscribed to an external source of information, receiving events from that external source on a periodic or continuous basis. An example of event-based collection is subscription to YANG Push notifications.

4.3.1.2. Endpoint

Building upon [I-D.ietf-sacm-terminology], the SACM Collection Sub-Architecture augments the definition of an Endpoint as a component within an organization's management domain from which a Posture Collection Service will collect relevant posture attributes.

4.3.1.3. Posture Attribute Repository

The Posture Attribute Repository is a SACM component responsible for the persistent storage of posture attributes collected via interactions between the Posture Collection Service and Endpoints.

4.3.1.4. Posture Collection Workflow

Posture collection may be triggered from a number of components, but commonly begin either via event-based triggering on an endpoint or through manual orchestration, both illustrated in Figure 3 above. Once orchestration has provided the directive to perform collection, posture collection services consume the directives. Posture collection is invoked for those endpoints overseen by the respective posture collection services. Collected data is then provided to the Integration Service, with a directive to store that information in an appropriate repository.

4.3.2. Evaluation Sub-Architecture

The Evaluation Sub-Architecture, in the SACM context, is the mechanism by which policy, expressed in the form of expected state, is compared with collected posture attributes to yield an evaluation result, that result being contextually dependent on the policy being evaluated.

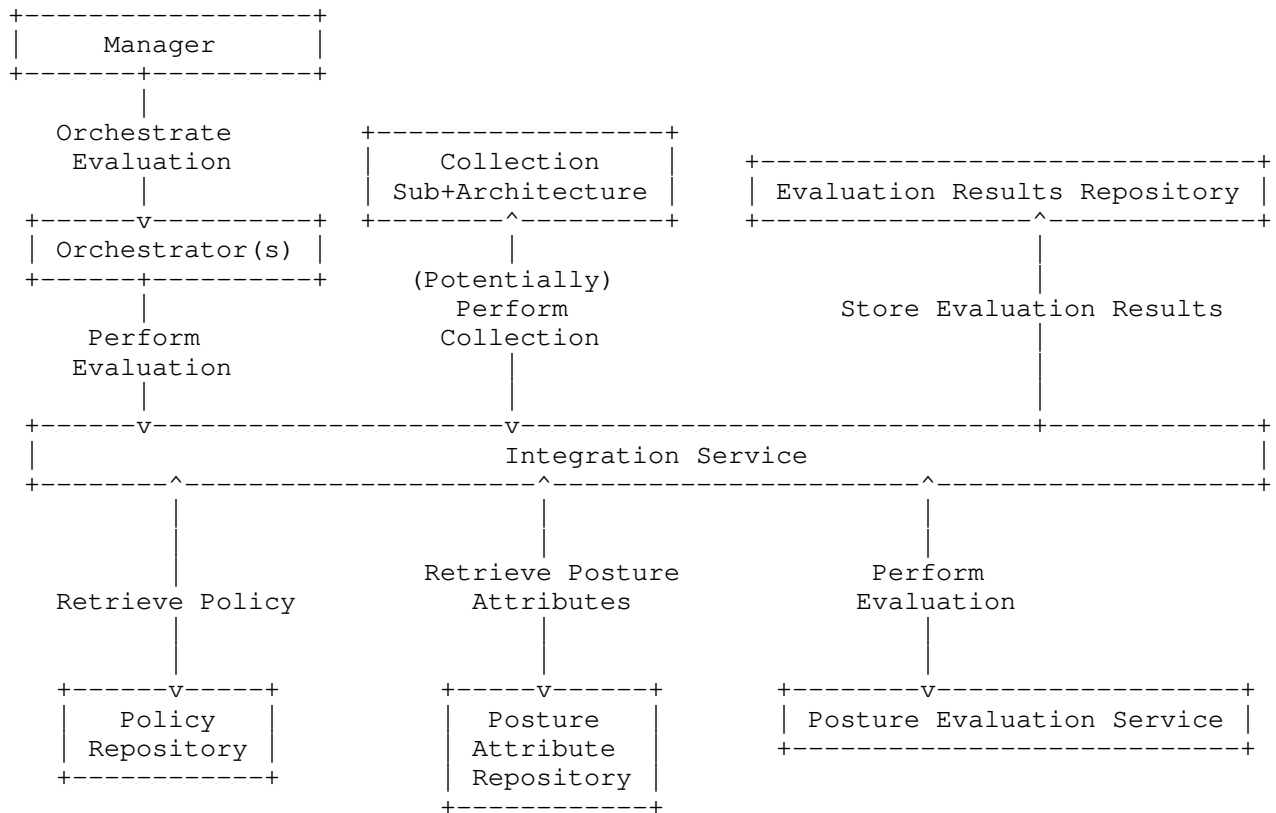


Figure 4: Decomposed Evaluation Sub-Architecture

4.3.2.1. Posture Evaluation Service

The Posture Evaluation Service (PES) represents the SACM component responsible for coordinating the policy to be evaluated and the collected posture attributes relevant to that policy, as well as the comparison engine responsible for correctly determining compliance with the expected state.

4.3.2.2. Policy Repository

The Policy Repository represents a persistent storage mechanism for the policy to be assessed against collected posture attributes to determine if an endpoint meets the desired expected state. Examples of information contained in a Policy Repository would be Vulnerability Definition Data or configuration recommendations as part of a CIS Benchmark or DISA STIG.

4.3.2.3. Evaluation Results Repository

The Evaluation Results Repository persists the information representing the results of a particular posture assessment, indicating those posture attributes collected from various endpoints which either meet or do not meet the expected state defined by the assessed policy. Consideration should be made for the context of individual results. For example, meeting the expected state for a configuration attribute indicates a correct configuration of the endpoint, whereas meeting an expected state for a vulnerable software version indicates an incorrect configuration.

4.3.2.4. Posture Evaluation Workflow

Posture evaluation is orchestrated through the Integration Service to the appropriate Posture Evaluation Service (PES). The PES will, using interactions defined by the applicable taxonomy, query both the Posture Attribute Repository and the Policy Repository to obtain relevant state data for comparison. If necessary, the PES may be required to invoke further posture collection. Once all relevant posture information has been collected, it is compared to expected state based on applicable policy. Comparison results are then persisted to an evaluation results repository for further downstream use and analysis.

5. Ecosystem Interactions

Ecosystem interactions describe the various functions between SACM components, including manager requirements, the onboarding of components, capability advertisement, administrative actions, and status updates, among others. The Manager component acts as the administrative "lead" for the SACM ecosystem, and must maintain records of registered components, manage capabilities, and more.

5.1. Manager

The Manager, being a specialized role in the architecture, enables the onboarding and capability management of the various SACM component roles. The Manager must support the set of capabilities needed to operate the SACM ecosystem.

With this in mind, the Manager must first authenticate to the Integration Service. Once authentication has succeeded, the Manager MUST establish a service handler capable of performing SACM component registration/onboarding activities (Component Registration Operation). The Manager MUST also establish a subscription to an ecosystem-wide status notification mechanism, in order to receive published status updates from other SACM components.

The following requirements exist for the Manager to establish service handlers supporting the component registration taxonomy (Component Registration Operation):

- * The Manager MUST enable the capability to receive onboarding requests,
- * The Manager MUST have the capability to generate, manage, and persist unique identifiers for all registered components,
- * The Manager MUST maintain the relationships between capabilities and payload categorizations (such as topic names or specific payload identifiers),
- * The Manager MUST have the capability to inventory and manage its "roster" (the list of registered components),
- * The Manager MUST have the capability to manage its roster's advertised capabilities, including those endpoints to which those capabilities apply.
- * In addition to supporting component registration, the Manager is responsible for many of the operational functions of the architecture, including initiating collection or evaluation, queries for repository data, or the assembly of information for downstream use.
- * The Manager MUST support making directed requests to registered components over the component's administrative interface. Administrative interface functions are described by their taxonomy, below.
- * The Manager MUST support each of the interaction categories as described above.

5.2. Component Registration

Component registration describes how an individual component becomes part of the SACM ecosystem; authenticating to the Integration Service, registering and establishing its administrative interface with, the Manager.

The component onboarding workflow involves multiple steps:

- * The component first authenticates to the Integration Service.

- * The component initiates registration with the Manager, per the component registration operation (Component Registration Operation).
- * The component handles the response from the Manager to configure a service handler allowing the component to receive directed messages over the administrative interface with the Manager.

5.3. Administrative Interface

The administrative interface represents a direct communication channel between the Manager and any registered Component. This interface allows the Manager to make directed requests to a component in order to perform specific actions.

5.3.1. Capability Advertisement Handshake

Capability Advertisement is the mechanism by which components initially indicate their capabilities to the Manager. This handshake is completed using the administrative interface with the Manager. It becomes the Manager's responsibility to persist component/capability relationships, and to provide the component the information necessary to receive and process message payloads specific to the supported capabilities.

5.3.2. Health Check

The administrative "health check" is a mechanism by which the Manager queries for the "liveness" of its roster of components, and to possibly alert users or other systems when components are no longer present. The Manager MAY enable a periodic message to each component to determine if that component is still listening to the Administrative Interface. The Health Check interaction MAY include a request for "Capability Refresh", to reinitiate the Capability Advertisement Handshake. This interaction is similar to the "Heartbeat" interaction, but is initiated by the Manager.

5.3.3. Heartbeat

The administrative "heartbeat" is a mechanism by which a Component indicates to the Manager that the Component remains connected to the ecosystem. The Heartbeat differs from the Health Check interaction in that the Component initiates the interaction, and that no response from the Manager is required.

5.3.4. Capability-specific Requests

Any number of capability-specific requests can be enabled through the administrative interface that allow the Manager to direct actions to be performed by a specific component. Utilizing the interface from a component to the Manager, this interface can be used to indicate a component has come back online, or to provide an updated capability advertisement, potentially resulting in updates to subscriptions or service handlers.

5.4. Status Notifications

A generic status notifications mechanism SHOULD be configured to which (a) the Manager is subscribed, and (b) all onboarded components can publish. Status notifications may be used by the Manager to update user interfaces, to provide notification of the start, finish, success or failure of ecosystem operations, or as events to trigger subsequent activities.

5.5. Component Interactions

Component interactions describe functionality between components relating to collection, evaluation, or other downstream processes. The following component interactions begin with the Manager providing a set of instructions to an Orchestrator or set of Orchestrators that have registered with the SACM ecosystem indicating the appropriate capabilities, such as collection or evaluation. Subscribing Orchestrator(s) MAY translate, manipulate, filter, augment, or otherwise transform the Manager's instructions into content supported through the Orchestrator's capabilities.

5.5.1. Initiate Ad-Hoc Collection

The Orchestrator supplies a payload of collection instructions to a Posture Collection Service either through direct or broadcast mechanisms. The receiving PCS components perform the required collection based on their capabilities. Each PCS then forms a payload of collected posture attributes (including endpoint identifying information) and provides that payload to the Posture Attribute Repository interface, for persistence.

5.5.2. Coordinate Periodic Collection

Similar to ad-hoc collection, the Orchestrator supplies a payload of collection instructions similar to those of ad-hoc collection. Additional information elements containing collection identification and periodicity are included.

5.5.2.1. Schedule Periodic Collection

To enable operations on periodic collection, the scheduling payload MUST include both a unique identifier for the set of collection instructions, as well as a periodicity expression to enable the collection schedule. An optional "immediate collection" flag will indicate to the collection component that, upon receipt of the collection instructions, a collection will automatically be initiated prior to engagement of the scheduled collection.

5.5.2.2. Cancel Periodic Collection

The Orchestrator disables the periodic collection of posture attributes by supplying collector(s) the unique identifier of previously scheduled collection instructions. An optional "final collection" flag will indicate to the collection component that, upon receipt of the cancellation instructions, a final ad-hoc collection is to take place.

5.5.3. Coordinate Observational/Event-based Collection

In these scenarios, the Posture Collection Service acts as the "observer". Interactions with the observer could specify a time period of observation and potentially information intended to filter observed posture attributes to aid the PCS in determining those attributes that are applicable for collection and persistence to the Posture Attribute Repository.

5.5.3.1. Initiate Observational/Event-based Collection

The Orchestrator supplies a payload of instructions to a topic or set of topics to which Posture Collection Services (observers) are subscribed. This payload could include specific instructions based on the observer's capabilities to determine specific posture attributes to observe and collect.

5.5.3.2. Cancel Observational/Event-based Collection

The Orchestrator supplies a payload of instructions to a topic or set of topics to which Posture Collection Services are subscribed. The receiving PCS components cancel the identified observational/event-based collection executing on those PCS components.

5.5.4. Persist Collected Posture Attributes

Following successful collection, Posture Collection Services (PCS) will supply the payload of collected posture attributes to the interface(s) supporting the persistent storage of those attributes to the Posture Attribute Repository. Information in this payload should include identifying information of the computing resource(s) for which attributes were collected.

5.5.5. Initiate Ad-Hoc Evaluation

The Orchestrator supplies a payload of evaluation instructions to a Posture Evaluation Services (PES) either through direct or broadcast mechanisms. The receiving PES components perform the required evaluation based on their capabilities. The PES generates a payload of posture evaluation results and publishes that payload to the Evaluation Results Repository interface, for persistence.

5.5.6. Coordinate Periodic Evaluation

Similar to ad-hoc evaluation, the Orchestrator supplies a payload of evaluation instructions similar to those of ad-hoc evaluation. Additional information elements containing evaluation identification and periodicity are included.

5.5.6.1. Schedule Periodic Evaluation

To enable operations on periodic evaluation, the scheduling payload MUST include both a unique identifier for the set of evaluation instructions, as well as a periodicity expression to enable the evaluation schedule. An optional "immediate evaluation" flag will indicate to the Posture Evaluation Service (PES) that, upon receipt of the evaluation instructions, an evaluation will automatically be initiated prior to engagement of the scheduled evaluation.

5.5.6.2. Cancel Periodic Evaluation

The Orchestrator disables the periodic evaluation of posture attributes by supplying Posture Evaluation Service(s) the unique identifier of previously scheduled evaluation instructions. An optional "final evaluation" flag will indicate to the PES that, upon receipt of the cancellation instructions, a final ad-hoc evaluation is to take place.

5.5.7. Coordinate Change-based Evaluation

A more fine-grained approach to periodic evaluation may be enabled through the triggering of Posture Evaluation based on changes to posture attribute values at the time of their collection and persistence to the Posture Attribute Repository.

5.5.7.1. Identify Attributes

The Orchestrator enables change-based evaluation through a payload published to Posture Attribute Repository component(s). This payload includes appropriate information elements describing the posture attributes on which changes in value will trigger posture evaluation.

5.5.7.2. Cancel Change-based Evaluation

An Orchestrator may disable change-based evaluation through a payload published to Posture Attribute Repository component(s), including those information elements necessary to identify those posture attributes for which change-based evaluation no longer applies.

5.5.8. Queries

Queries should allow for a "freshness" time period, allowing the requesting entity to determine if/when posture attributes must be re-collected prior to performing evaluation. This freshness time period can be "zeroed out" for the purpose of automatically triggering re-collection regardless of the most recent collection.

6. Operations

The following sections describe a number of operations required to enable a cooperative ecosystem of posture attribute collection and evaluation functions.

6.1. Component Registration

Component registration describes how an individual component becomes part of the SACM ecosystem; registering with the Manager, and establishing the administrative interface.

- * Interaction Type: Directed (Request/Response)
- * Source Component: Any component wishing to join the ecosystem, such as Posture Collection Services, Repository Interfaces, Posture Evaluation Services and more.
- * Target Component(s): Manager

6.1.1. Request Payload

When a component onboards with the ecosystem, it must identify itself to the Manager, using either descriptive information or an already-existing component unique identifier.

```
component-registration-request:
  {:component-identification:}
```

```
component-identification:
  component-unique-identifier (if re-establishing communication)
  #-OR-#
  component-type {:component-type:}
  component-name
  component-description (optional)
```

```
component-type:
  enumeration:
    - posture-collection-service
    - posture-evaluation-service
    - repository-interface
    - orchestrator
    - others?
```

When registering for the first time, the component will send identifying information including the component type and a name. If the component is re-establishing communications, for example after a restart of the component or deployment of a new version, the component only needs to supply its previously generated (and persisted) [component-unique-identifier].

6.1.2. Request Processing

When the Manager receives the component's request for onboarding, it will:

- * Generate a unique identifier, "[component-unique-identifier]", for the onboarding component,
- * Persist identifying information, including the "[component-unique-identifier]" to its component inventory, enabling an up-to-date roster of components being managed,
- * Establish the administrative interface to the onboarded component by enabling a service handler to listen for directed messages from the component.

6.1.3. Response Payload

The Manager will respond to the component with a payload including the component's unique identifier. At this point, the Manager is aware of the component's existence in the ecosystem, and the component can self-identify by virtue of receiving its unique identifier.

```
component-registration-response:  
  component-unique-identifier: [component-unique-identifier]
```

6.1.4. Response Processing

Successful receipt of the Manager's response, including the "[component-unique-identifier]", indicates the component is onboarded to the ecosystem. Using the response payload, the component can then establish its end of the administrative interface with the Manager. The component must then persist its unique identifier for use when re-establishing communication with the Manager after failure recovery or restart.

6.2. Administrative Interface

A number of functions may take place which, instead of being published to multiple subscribers, may require direct interaction between the Manager and a registered component (and vice-versa). During component onboarding, this direct channel, known as the Administrative Interface, is established first by the Manager and subsequently complemented by the component onboarding the SACM ecosystem. Three operations are defined for the administrative interface, but any number of application or capability-specific operations MAY be enabled using the directed messaging provided by this interface.

6.2.1. Capability Advertisement Handshake

Capability advertisement represents the ability of any registered component to inform the Manager of that component's capacity for performing certain operations. For example, a Posture Collection Service component may advertise its capability to perform collection using a particular collection system/serialization. This capability advertisement is important for the Manager to persist in order for the Manager to correctly classify components registered within the SACM ecosystem, and therefore provide the ability to publish messages to components in accordance with their capabilities.

* Interaction Type: Directed (Request/Response)

- * Source Component: Any registered component, such as Posture Collection Services, Repository Interfaces, Posture Evaluation Services and more.
- * Target Component(s): Manager

6.2.1.1. Request Payload

The component's capability advertisement request payload will include a list of "Capability URNs" (TBD IANA SECTION) that represent it's supported operational capabilities.

```
capability-advertisement:
  capabilities:
    capability-urn: [urn]
    capability-urn: [urn]
    capability-urn: [urn]
    ...
```

6.2.1.2. Request Processing

Upon receipt of the component's capability advertisement, the Manager SHOULD:

- * Persist the component's capabilities to the Manager's inventory
- * Coordinate, based on the supplied capabilities, the service handlers (for directed messages) and/or event listeners (for broadcast messages) to which the component should support.

6.2.1.3. Response Payload

The response payload delivered to the component should include the appropriate service handling/event listening information required for the component to handle further interactions based on each advertised capability. If a capability was not registered successfully, appropriate error messages SHOULD be supplied to inform the component of the failure(s).

```
capability-advertisement-response:
  capabilities:
    capability:
      capability-urn: [urn]
      registration-status: (success | failure)
      service-handler-or-event-listener: [info]
      messages: [messages]
    capability:
      capability-urn: [urn]
      registration-status: (success | failure)
      service-handler-or-event-listener: [info]
      messages: [messages]
```

6.2.1.4. Response Processing

Once the component has received the response to its capability advertisement, it should configure the capability-specific service handler(s) or event listener(s). Once these handlers/listeners have been configured, the component is considered fully onboarded to the SACM ecosystem.

6.2.2. Health Check

As time passes, it is important that the Manager maintains knowledge of all registered component's current operational status. The health check operation describes the efforts taken by the Manager to maintain the most up-to-date inventory of it's component roster, and to potentially trigger events to users or outside systems (e.g. a SIEM or SOAR) indicating unavailable components.

- * Interaction Type: Directed (Request/Response)
- * Source Component: Manager
- * Target Component(s): Any registered component, such as Posture Collection Services, Repository Interfaces, Posture Evaluation Services and more.

6.2.2.1. Request Payload

The request for the health check is a simple "ping".

```
health-check-request:
  action: ping
```

6.2.2.2. Request Processing

When the target component receives the health check request, the target component need only respond that it is operational and connected to the integration service. This is a simple "Hello component, are you listening? Yes, I am" interaction. The health check request from the Manager should be made with an appropriately small timeout indicator; only an operational component will be able to respond to the request, so if that component is offline and cannot respond, the Manager should not be kept waiting for an extended amount of time.

6.2.2.3. Response Payload

When responding to the health check request, the response payload will simply indicate success: ~~~~~ health-check-response: response: success ~~~~~

6.2.2.4. Response Processing

Upon receipt of the "health-check-response" payload, the Manager will update its inventory of currently operational components with the timestamp of the receipt. Manager implementations may raise alerts, inform users, or take other actions when health checks are unsuccessful, at their discretion.

6.2.3. Heartbeat

As time passes and SACM ecosystem components which have previously registered are brought offline (perhaps for maintenance or redeployment) and back online, it is important that registered components maintain administrative contact with the Manager. The heartbeat operation describes the efforts taken by a registered component to determine the status of contact with the Manager, and to take appropriate action if such contact cannot be made.

- * Interaction Type: Directed (Request/Response)
- * Source Component: Any registered component, such as Posture Collection Services, Repository Interfaces, Posture Evaluation Services and more.
- * Target Component(s): Manager

6.2.3.1. Request Payload

The request payload simply defines the heartbeat action:


```
heartbeat-request:
  action: pulse
```

6.2.3.2. Request Processing

When the Manager receives the heartbeat request, it need only respond that it is operational and connected to the integration service. This is a simple "Hello Manager, are you listening? Yes, I am" interaction. The heartbeat request from the component should be made with an appropriately small timeout indicator; only an operational Manager will be able to respond to the request, so if it is offline and cannot respond, the component should not be kept waiting for an extended amount of time.

6.2.3.3. Response Payload

When responding to the heartbeat request, the response payload will simply indicate success: ~~~~~ heartbeat-response: response: success ~~~~~

6.2.3.4. Response Processing

Upon receipt of the "heartbeat-response" payload, the component may reset its heartbeat timer and continue normal operations, awaiting incoming message payloads. Component implementations may raise alerts, inform users, or take other actions when heartbeat requests are unsuccessful (potentially indicating a downed Manager), at their discretion.

6.3. Status Notification

From time to time during the performance of any given operation, a component may need to supply status information to the Manager (or any other concerned component), for use in display to users, or to trigger other events within the SACM ecosystem. The status notification operation is designed to allow any component to broadcast status message payloads to any subscribers with the need to know. For example, a collection component could broadcast to the Manager that it has initiated collection, subsequent collection progress updates, and finally completion or error conditions.

- * Interaction Type: Broadcast (Publish/Subscribe)
- * Source Component: Any registered component, such as Posture Collection Services, Repository Interfaces, Posture Evaluation Services and more.

- * Target Component(s): Typically the Manager, but any registered component may subscribe to status notifications.

6.3.1. Request Payload

At a minimum, the payload broadcast for a status notification MUST include the status message and the publishing component's "component-unique-identifier". Further identifying information, such as status codes, operation indicators, etc., MAY be provided by implementing components.

```
status-notification:
  publisher: [component-unique-identifier]
  message: [message]
  [additional information]
```

6.3.2. Request Processing

When subscribers are notified of the status message, respective components may act upon them in component/application-specific ways, including persisting those messages to repositories, forwarding to log aggregation tools, displaying on user interfaces, and so on. Potential for use of component status notifications is only limited by application implementations.

6.3.3. Response Payload

N/A

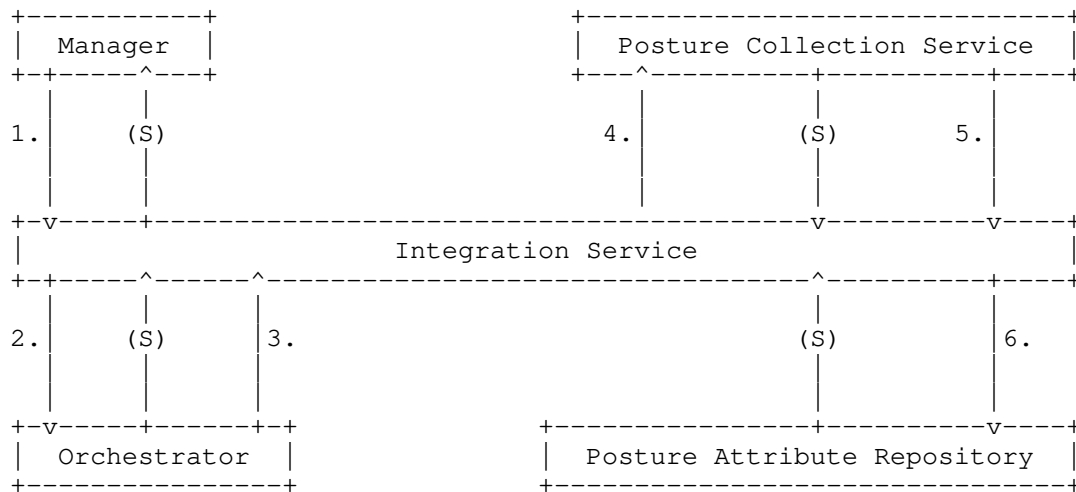
6.3.4. Response Processing

N/A

6.4. Initiate Ad-Hoc Collection

The Ad-hoc collection workflow MAY be initiated by the Manager, via user interaction, or through a Posture Evaluation Service, and represents a single, point-in-time operation to collect posture attributes from applicable endpoints. The SACM Producer initiates a message payload, either through directed channels (such as the administrative interface) or through broadcast notifications to multiple subscribers, to Orchestrator components. Orchestrators MAY manipulate the Manager's collection instructions according to various collection capabilities, prior to providing those instructions to Posture Collection Service (PCS) components. Once collection instructions are received by the PCS, it will collect the requested posture attributes from the designated endpoints, using its advertised collection capabilities. The following diagram

illustrates this workflow with the Manager as the initiating SACM Producer:



1. The Manager initiates a request to one or more Orchestrators to perform collection,
2. The Orchestrator receives collection instructions and potentially manipulates them according to one or more collection capabilities,
3. The Orchestrator publishes a notification to subscribed Posture Collection Service components, indicating the posture attributes to be collected,
4. The Posture Collection Service receives the collection instructions and performs the actual collection of posture attributes from an endpoint or endpoints.
5. The Posture Collection Service publishes a notification(s) containing the collected posture attributes to be persisted to the Posture Attribute Repository,
6. The Posture Attribute Repository persists the collected posture attributes, potentially performing normalization of the data as part of its process.

Interactions labeled (S) indicate the capability of each component to publish status notifications, subscribed to by the Manager.

6.4.1. SACM Producer to Orchestrator

The Ad-hoc collection workflow MAY be initiated by a number of SACM components, such as the Manager, a Posture Evaluation Service, or other events outside the scope of this document.

- * Interaction Type: Directed (Request/Response) or Broadcast (Publish/Subscribe)
- * Source Component: Various
- * Target Component(s): Orchestrator

6.4.1.1. Request Payload

A request to orchestrate posture attribute collection MUST include enough information to describe those attributes being collected, and MAY include endpoint targeting information.

collection-instructions:
TBD

6.4.1.2. Request Processing

When the Orchestrator receives the collection instructions, it may be required to manipulate them according to the capabilities it's collector(s) support. For example, generic collection instructions could be transformed to the appropriate OVAL serialization for collection via OVAL-compliant collectors.

6.4.1.3. Response Payload

Orchestrators have the option to provide broadcast status update messages to indicate success, failure, or other error messages when receiving posture collection orchestration payloads.

6.4.1.4. Response Processing

N/A

6.4.2. Orchestrator to Posture Collection Service

Once the Orchestrator has received collection instructions from the initiating SACM component, and has performed any manipulation of the instructions to conform to it's capabilities, it will provide those instructions to relevant Posture Collection Services.

- * Interaction Type: Directed (Request/Response) or Broadcast (Publish/Subscribe)
- * Source Component: Orchestrator
- * Target Component(s): Posture Collection Service

6.4.2.1. Request Payload

The payload exchanged between the Orchestrator and its associated Posture Collection Services will be collection instructions adhering to a data model supported by the PCS based on its advertised capabilities.

collection-instructions:
TBD

6.4.2.2. Request Processing

Upon receipt of the payload containing collection instructions, the Posture Collection Service should parse and validate them, indicating any errors in the process. If the payload does not conform to any serialization or data model to which the PCS' capabilities correspond, status messages indicating such nonconformance SHOULD be provided to both the Orchestrator and the initiating SACM producer.

Once successfully parsed and validated, the PCS MUST perform collection of posture attributes according to the collection instructions, from any endpoint to which the PCS has access, or from the list of endpoints described in any targeting information included in the collection instructions.

6.4.2.3. Response Payload

Posture Collection Service components will respond using the generic status update mechanisms to indicate success, failure, or any errors that occur. Errors may occur parsing collection instructions, verifying them, targeting indicated endpoints, or from the act of collecting the indicated posture attributes.

6.4.2.4. Response Processing

Any messages received by components regarding the success, failure, or errors involved in the collection of posture attributes MAY be processed according to the receiving components' capabilities.

6.4.3. Posture Collection Service to Posture Attribute Repository

Upon completion of posture attribute collection, the PCS constructs the payload of collected attributes based on its advertised capabilities, e.g. OVAL system characteristics. This payload is provided to either a specific posture attribute repository via directed messages or to subscribed repository interfaces via broadcast messages.

- * Interaction Type: Directed (Request/Response) or Broadcast (Publish/Subscribe)
- * Source Component: Posture Collection Service
- * Target Component(s): Posture Attribute Repository

6.4.3.1. Request Payload

The payload supplied by the Posture Collection Service SHOULD conform to information and data models supported by its advertised capabilities. These data models, at a minimum, SHOULD include name/value pairs for each collected attribute.

```
collection-results:
  [
    attribute-name,
    attribute-value
  ]
```

6.4.3.2. Request Processing

As the Posture Attribute Repository interface receives the payload of collected posture attributes, some data normalization MAY occur in order to persist the information most efficiently based on the persistence technology. This normalization is dependent on the implementation of the repository interface as well as the persistence technology. For example, OVAL system characteristics, an XML payload, could be normalized to a property graph representation when persisted to a Neo4j database.

6.4.3.3. Response Payload

Once the Posture Attribute Repository has received, it MAY respond to the Posture Collection Service that it has successfully received the collected posture attributes. This response would only be applicable when receiving payloads via directed requests. If payloads are received via broadcast interactions, there may not be a PCS to which a response can be sent. The Posture Attribute Repository MAY utilize

the generic status update interactions to provide response messages to appropriate subscribers.

6.4.3.4. Response Processing

Any messages received by components regarding the success, failure, or errors involved in the persistence of collected posture attributes MAY be processed according to the receiving components' capabilities. For example, a generic status update message could be processed by a Manager component, correlated with the initial posture collection instructions in order to "close the loop" on the posture attribute collection workflow.

6.5. Initiate Ad-Hoc Evaluation

Manager to Orchestrator ### Orchestrator to Evaluator ###
Evaluator to Posture Evaluation Repository

7. Privacy Considerations

[TBD]

8. Security Considerations

[TBD]

9. IANA Considerations

[TBD] Some boilerplate code...

9.1. Component Types

URI: "urn:ietf:sacm:component-type" Description: The allowed enumeration of the various component types permitted to utilize the SACM ecosystem.

- * Manager
- * Orchestrator
- * Collector
- * Evaluator
- * Repository Interface
- * [MORE]

9.2. Component Capabilities

Health Check A URN representing a component's capability to initiate Health Check operations and to process any provided response payloads.

URN: "urn:ietf:sacm:capability:action:health-check"

9.2.1. Heartbeat

A URN representing a component's capability to initiate Heartbeat operations and to process any provided response payloads.

URN: "urn:ietf:sacm:capability:action:heartbeat"

9.2.2. Status Notification (Publish)

A URN representing a component's capability to publish status notifications.

URN: "urn:ietf:sacm:capability:publish:status-notification"

9.2.3. Status Notification (Subscribe)

A URN representing a component's capability to subscribe to status notification events.

URN: "urn:ietf:sacm:capability:subscribe:status-notification"

10. References

10.1. Normative References

- [I-D.ietf-sacm-ecp] Haynes, D., Fitzgerald-McKay, J., and L. Lorenzin, "Endpoint Posture Collection Profile", Work in Progress, Internet-Draft, draft-ietf-sacm-ecp-05, 21 June 2019, <<https://www.ietf.org/archive/id/draft-ietf-sacm-ecp-05.txt>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC6120] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, DOI 10.17487/RFC6120, March 2011, <<https://www.rfc-editor.org/info/rfc6120>>.

- [RFC8412] Schmidt, C., Haynes, D., Coffin, C., Waltermire, D., and J. Fitzgerald-McKay, "Software Inventory Message and Attributes (SWIMA) for PA-TNC", RFC 8412, DOI 10.17487/RFC8412, July 2018, <<https://www.rfc-editor.org/info/rfc8412>>.
- [RFC8600] Cam-Winget, N., Ed., Appala, S., Pope, S., and P. Saint-Andre, "Using Extensible Messaging and Presence Protocol (XMPP) for Security Information Exchange", RFC 8600, DOI 10.17487/RFC8600, June 2019, <<https://www.rfc-editor.org/info/rfc8600>>.

10.2. Informative References

- [CISCONTROLS] "CIS Controls v7.1", n.d., <<https://www.cisecurity.org/controls>>.
- [draft-birkholz-sacm-yang-content] Birkholz, H. and N. Cam-Winget, "YANG subscribed notifications via SACM Statements", n.d., <<https://tools.ietf.org/html/draft-birkholz-sacm-yang-content-01>>.
- [HACK100] "IETF 100 Hackathon - Vulnerability Scenario EPCP+XMPP", n.d., <<https://www.github.com/sacmwg/vulnerability-scenario/ietf-hackathon>>.
- [HACK101] "IETF 101 Hackathon - Configuration Assessment XMPP", n.d., <<https://www.github.com/CISecurity/Integration>>.
- [HACK102] "IETF 102 Hackathon - YANG Collection on Traditional Endpoints", n.d., <<https://www.github.com/CISecurity/YANG>>.
- [HACK103] "IETF 103 Hackathon - N/A", n.d., <<https://www.ietf.org/how/meetings/103/>>.
- [HACK104] "IETF 104 Hackathon - A simple XMPP client", n.d., <<https://github.com/CISecurity/SACM-Architecture>>.
- [HACK105] "IETF 105 Hackathon - A more robust XMPP client including collection extensions", n.d., <<https://github.com/CISecurity/SACM-Architecture>>.
- [HACK99] "IETF 99 Hackathon - Vulnerability Scenario EPCP", n.d., <<https://www.github.com/sacmwg/vulnerability-scenario/ietf-hackathon>>.

[I-D.ietf-i2nsf-terminology]

Hares, S., Strassner, J., Lopez, D. R., Xia, L., and H. Birkholz, "Interface to Network Security Functions (I2NSF) Terminology", Work in Progress, Internet-Draft, draft-ietf-i2nsf-terminology-08, 5 July 2019, <<https://www.ietf.org/archive/id/draft-ietf-i2nsf-terminology-08.txt>>.

[I-D.ietf-sacm-terminology]

Birkholz, H., Lu, J., Strassner, J., Cam-Winget, N., and A. Montville, "Security Automation and Continuous Monitoring (SACM) Terminology", Work in Progress, Internet-Draft, draft-ietf-sacm-terminology-16, 14 December 2018, <<https://www.ietf.org/archive/id/draft-ietf-sacm-terminology-16.txt>>.

[MQTT] "MQTT", n.d., <<https://mqtt.org/mqtt-specification/>>.

[NIST800126]

Waltermire, D., Quinn, S., Booth, H., Scarfone, K., and D. Prisaca, "SP 800-126 Rev. 3 - The Technical Specification for the Security Content Automation Protocol (SCAP) - SCAP Version 1.3", February 2018, <<https://csrc.nist.gov/publications/detail/sp/800-126/rev-3/final>>.

[NISTIR7694]

Halbardier, A., Waltermire, D., and M. Johnson, "NISTIR 7694 Specification for Asset Reporting Format 1.1", n.d., <<https://csrc.nist.gov/publications/detail/nistir/7694/final>>.

[RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between Information Models and Data Models", RFC 3444, DOI 10.17487/RFC3444, January 2003, <<https://www.rfc-editor.org/info/rfc3444>>.

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2", FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007, <<https://www.rfc-editor.org/info/rfc4949>>.

[RFC5023] Gregorio, J., Ed. and B. de hOra, Ed., "The Atom Publishing Protocol", RFC 5023, DOI 10.17487/RFC5023, October 2007, <<https://www.rfc-editor.org/info/rfc5023>>.

- [RFC5209] Sangster, P., Khosravi, H., Mani, M., Narayan, K., and J. Tardo, "Network Endpoint Assessment (NEA): Overview and Requirements", RFC 5209, DOI 10.17487/RFC5209, June 2008, <<https://www.rfc-editor.org/info/rfc5209>>.
- [RFC6192] Dugal, D., Pignataro, C., and R. Dunn, "Protecting the Router Control Plane", RFC 6192, DOI 10.17487/RFC6192, March 2011, <<https://www.rfc-editor.org/info/rfc6192>>.
- [RFC7632] Waltermire, D. and D. Harrington, "Endpoint Security Posture Assessment: Enterprise Use Cases", RFC 7632, DOI 10.17487/RFC7632, September 2015, <<https://www.rfc-editor.org/info/rfc7632>>.
- [RFC8248] Cam-Winget, N. and L. Lorenzin, "Security Automation and Continuous Monitoring (SACM) Requirements", RFC 8248, DOI 10.17487/RFC8248, September 2017, <<https://www.rfc-editor.org/info/rfc8248>>.
- [RFC8322] Field, J., Banghart, S., and D. Waltermire, "Resource-Oriented Lightweight Information Exchange (ROLIE)", RFC 8322, DOI 10.17487/RFC8322, February 2018, <<https://www.rfc-editor.org/info/rfc8322>>.
- [XMPPEXT] "XMPP Extensions", n.d., <<https://xmpp.org/extensions/>>.

Appendix A. Security Domain Workflows

This section describes three primary information security domains from which workflows may be derived: IT Asset Management, Vulnerability Management, and Configuration Management.

A.1. IT Asset Management

Information Technology asset management is easier said than done. The [CISCONTROLS] have two controls dealing with IT asset management. Control 1, Inventory and Control of Hardware Assets, states, "Actively manage (inventory, track, and correct) all hardware devices on the network so that only authorized devices are given access, and unauthorized and unmanaged devices are found and prevented from gaining access." Control 2, Inventory and Control of Software Assets, states, "Actively manage (inventory, track, and correct) all software on the network so that only authorized software is installed and can execute, and that unauthorized and unmanaged software is found and prevented from installation or execution."

In spirit, this covers all of the processing entities on your network (as opposed to things like network cables, dongles, adapters, etc.), whether physical or virtual, on-premises or in the cloud.

A.1.1. Components, Capabilities and Workflow(s)

TBD

A.1.1.1. Components

TBD

A.1.1.2. Capabilities

An IT asset management capability needs to be able to:

- * Identify and catalog new assets by executing Target Endpoint Discovery Tasks
- * Provide information about its managed assets, including uniquely identifying information (for that enterprise)
- * Handle software and/or hardware (including virtual assets)
- * Represent cloud hybrid environments

A.1.1.3. Workflow(s)

TBD

A.2. Vulnerability Management

Vulnerability management is a relatively established process. To paraphrase the [CISCONTROLS], continuous vulnerability management is the act of continuously acquiring, assessing, and taking subsequent action on new information in order to identify and remediate vulnerabilities, therefore minimizing the window of opportunity for attackers.

A vulnerability assessment (i.e. vulnerability detection) is performed in two steps:

- * Endpoint information collected by the endpoint management capabilities is examined by the vulnerability management capabilities through Evaluation Tasks.

- * If the data possessed by the endpoint management capabilities is insufficient, a Collection Task is triggered and the necessary data is collected from the target endpoint.

Vulnerability detection relies on the examination of different endpoint information depending on the nature of a specific vulnerability. Common endpoint information used to detect a vulnerability includes:

- * A specific software version is installed on the endpoint
- * File system attributes
- * Specific state attributes

In some cases, the endpoint information needed to determine an endpoint's vulnerability status will have been previously collected by the endpoint management capabilities and available in a Repository. However, in other cases, the necessary endpoint information will not be readily available in a Repository and a Collection Task will be triggered to perform collection from the target endpoint. Of course, some implementations of endpoint management capabilities may prefer to enable operators to perform this collection even when sufficient information can be provided by the endpoint management capabilities (e.g. there may be freshness requirements for information).

A.2.1. Components, Capabilities and Workflow(s)

TBD

A.2.1.1. Components

TBD

A.2.1.2. Capabilities

TBD

A.2.1.3. Workflow(s)

TBD

A.3. Configuration Management

Configuration management involves configuration assessment, which requires state assessment. The [CISCONTROLS] specify two high-level controls concerning configuration management (Control 5 for non-network devices and Control 11 for network devices). As an aside, these controls are listed separately because many enterprises have different organizations for managing network infrastructure and workload endpoints. Merging the two controls results in the following paraphrasing: Establish, implement, and actively manage (track, report on, correct) the security configuration of systems using a rigorous configuration management and change control process in order to prevent attackers from exploiting vulnerable services and settings.

Typically, an enterprise will use configuration guidance from a reputable source, and from time to time they may tailor the guidance from that source prior to adopting it as part of their enterprise standard. The enterprise standard is then provided to the appropriate configuration assessment tools and they assess endpoints and/or appropriate endpoint information.

A preferred flow follows:

- * Reputable source publishes new or updated configuration guidance
- * Enterprise configuration assessment capability retrieves configuration guidance from reputable source
- * Optional: Configuration guidance is tailored for enterprise-specific needs
- * Configuration assessment tool queries asset inventory repository to retrieve a list of affected endpoints
- * Configuration assessment tool queries configuration state repository to evaluate compliance
- * If information is stale or unavailable, configuration assessment tool triggers an ad hoc assessment

The SACM architecture needs to support varying deployment models to accommodate the current state of the industry, but should strongly encourage event-driven approaches to monitoring configuration.

A.3.1. Components, Capabilities and Workflow(s)

This section provides more detail about the components and capabilities required when considering the aforementioned configuration management workflow.

A.3.1.1. Components

The following is a minimal list of SACM Components required to implement the aforementioned configuration assessment workflow.

- * Configuration Policy Feed: An external source of authoritative configuration recommendations.
- * Configuration Policy Repository: An internal repository of enterprise standard configurations.
- * Configuration Assessment Orchestrator: A component responsible for orchestrating assessments.
- * Posture Attribute Collection Subsystem: A component responsible for collection of posture attributes from systems.
- * Posture Attribute Repository: A component used for storing system posture attribute values.
- * Configuration Assessment Evaluator: A component responsible for evaluating system posture attribute values against expected posture attribute values.
- * Configuration Assessment Results Repository: A component used for storing evaluation results.

A.3.1.2. Capabilities

Per [RFC8248], solutions MUST support capability negotiation. Components implementing specific interfaces and operations (i.e. interactions) will need a method of describing their capabilities to other components participating in the ecosystem; for example, "As a component in the ecosystem, I can assess the configuration of Windows, MacOS, and AWS using OVAL".

A.3.1.3. Configuration Assessment Workflow

This section describes the components and interactions in a basic configuration assessment workflow. For simplicity, error conditions are recognized as being necessary and are not depicted. When one component messages another component, the message is expected to be handled appropriately unless there is an error condition, or other notification, messaged in return.

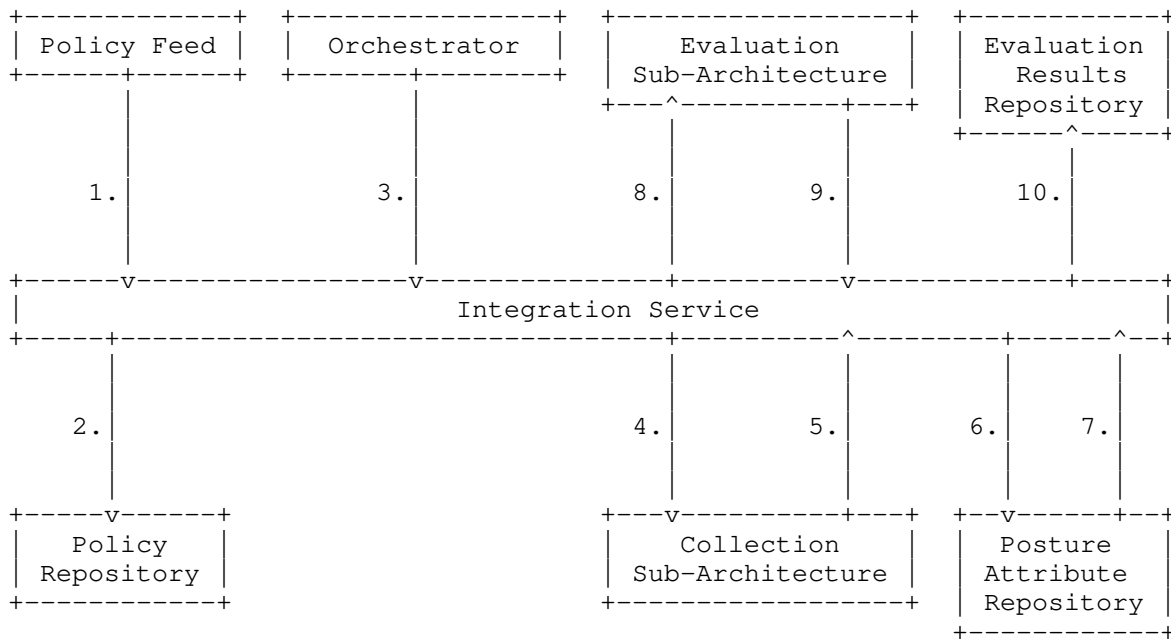


Figure 5: Configuration Assessment Component Interactions

Figure 5 depicts configuration assessment components and their interactions, which are further described below.

1. A policy feed provides a configuration assessment policy payload to the Integration Service.
2. The Policy Repository, a consumer of Policy Feed information, receives and persists the Policy Feed's payload.
3. Orchestration component(s), either manually invoked, scheduled, or event-based, publish a payload to begin the configuration assessment process.

4. If necessary, Collection Sub-Architecture components may be invoked to collect needed posture attribute information.
5. If necessary, the Collection Sub-Architecture will provide collected posture attributes to the Integration Service for persistence to the Posture Attribute Repository.
6. The Posture Attribute Repository will consume a payload querying for relevant posture attribute information.
7. The Posture Attribute Repository will provide the requested information to the Integration Service, allowing further orchestration payloads requesting the Evaluation Sub-Architecture perform evaluation tasks.
8. The Evaluation Sub-Architecture consumes the evaluation payload and performs component-specific state comparison operations to produce evaluation results.
9. A payload containing evaluation results are provided by the Evaluation Sub-Architecture to the Integration Service
10. Evaluation results are consumed by/persisted to the Evaluation Results Repository

In the above flow, the payload information is expected to convey the context required by the receiving component for the action being taken under different circumstances. For example, a directed message sent from an Orchestrator to a Collection sub-architecture might be telling that Collector to watch a specific posture attribute and report only specific detected changes to the Posture Attribute Repository, or it might be telling the Collector to gather that posture attribute immediately. Such details are expected to be handled as part of that payload, not as part of the architecture described herein.

Authors' Addresses

Adam W. Montville
Center for Internet Security
31 Tech Valley Drive
East Greenbush, NY 12061
United States of America

Email: adam.montville.sdo@gmail.com

Bill Munyan
Center for Internet Security
31 Tech Valley Drive
East Greenbush, NY 12061
United States of America

Email: bill.munyan.ietf@gmail.com

SACM
Internet-Draft
Intended status: Standards Track
Expires: August 28, 2020

D. Haynes
The MITRE Corporation
J. Fitzgerald-McKay
Department of Defense
February 25, 2020

Endpoint Posture Collection Profile
draft-ietf-sacm-epcp-01

Abstract

This document specifies the Endpoint Posture Collection Profile, which describes the requirements for the application of IETF, TNC, and ISO/IEC data models, protocols, and interfaces to support the on-going collection and communication of endpoint posture to a centralized server where it can be stored and made available to other tools.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 28, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
1.1. Conventions Used in This Document	5
1.2. Terminology	5
2. Endpoint Posture Collection Profile	5
2.1. Components	6
2.1.1. Endpoint	7
2.1.1.1. Posture Collection Engine	8
2.1.2. Posture Manager	8
2.1.2.1. Posture Collection Manager	8
2.1.3. Repository	9
2.1.4. Evaluator	9
2.1.5. Orchestrator	9
2.1.6. Application Programming Interface	9
2.2. Transactions	10
2.2.1. Provisioning	10
2.2.2. Discovery and Validation	10
2.2.3. Event-Driven Collection	10
2.2.4. Querying the Endpoint	10
2.2.5. Data Storage	11
2.2.6. Data Sharing	11
3. IETF NEA EPCP Implementation for Traditional Endpoints	11
3.1. Endpoint Provisioning	13
3.2. Endpoint	13
3.2.1. Posture Collector	14
3.2.2. Posture Broker Client	14
3.2.3. Posture Transport Client	14
3.3. Posture Manager	14
3.3.1. Posture Validator	14
3.3.2. Posture Broker Server	14
3.3.3. Posture Transport Server	14
3.4. Repository	15
3.5. IETF SACM Software Asset Management Extension to the IETF NEA EPCP Implementation	15
3.5.1. Endpoint Pre-Provisioning	15
3.5.2. SWID Tags	15
3.5.3. SWID Posture Collectors and Posture Validators	16
3.5.3.1. The SWID Posture Collector	16
3.5.3.2. The SWID Posture Validator	16
3.5.4. Repository	17
4. IETF NETCONF EPCP Implementation for Network Device Endpoints	17
4.1. Endpoint Provisioning	18
4.2. Posture Manager Provisioning	18
4.3. Endpoint	18

4.3.1. Datastore	18
4.4. Posture Manager	19
4.5. Repository	19
5. Future Work	19
6. Contributors	20
7. IANA Considerations	20
8. Security Considerations	21
8.1. Threat Model	21
8.1.1. Endpoint Attacks	21
8.1.2. Network Attacks	22
8.1.3. Posture Manager Attacks	22
8.1.4. Repository Attacks	22
8.2. Countermeasures	23
8.2.1. Countermeasures for Endpoint Attacks	23
8.2.2. Countermeasures for Network Attacks	23
8.2.3. Countermeasures for Posture Manager Attacks	24
8.2.4. Countermeasures for Repository Attacks	25
9. Privacy Considerations	25
10. References	26
10.1. Informative References	26
10.2. Normative References	26
Appendix A. Rationale for an EPCP Solution	29
A.1. Preventative Posture Assessments	29
A.2. All Network-Connected Endpoints are Endpoints	29
A.3. All Endpoints on the Network Must be Uniquely Identified	30
A.4. Standardized Data Models	30
A.5. Posture Information Must Be Stored	30
A.6. Posture Information Can Be Shared	31
A.7. Enterprise Asset Posture Information Belongs to the Enterprise	31
Appendix B. EPCP Supported Use Cases and Non-Supported Use Cases	31
B.1. Supported Use Cases	31
B.1.1. Hardware Asset Management	31
B.1.2. Software Asset Management	32
B.1.3. Vulnerability Management	32
B.1.4. Threat Detection and Analysis	32
B.2. Non-Supported Use Cases	33
Appendix C. Endpoint Posture Collection Profile Examples	33
C.1. Continuous Posture Assessment of an Endpoint	33
C.1.1. Change on Endpoint Triggers Posture Assessment	34
C.2. Administrator Searches for Vulnerable Endpoints	37
Appendix D. Change Log	38
D.1. -00 to -01	38
D.2. -05 to -00	38
D.3. -04 to -05	39
D.4. -03 to -04	39
D.5. -02 to -03	40
D.6. -01 to -02	40

D.7.	-00 to -01	41
D.8.	-01 to -02	41
D.9.	-02 to -00	41
D.10.	-00 to -01	41
Authors' Addresses		41

1. Introduction

The Endpoint Posture Collection Profile (EPCP) describes the requirements for the collection and communication of posture information from network-connected endpoints to a centralized server leveraging prior work from the IETF NEA WG, the IETF NETCONF WG, IETF NETMOD WG, the Trusted Computing Group (TCG) Trusted Network Communications [TNC] Work Group, and the International Organization for Standardization/International Electrotechnical Commission Joint Technical Committee (JTC) 1, Subcommittee (SC) 7, WG 21 (ISO/IEC JTC 1, SC7, WG21).

This document focuses on reducing the security exposure of a network by enabling:

- o event-driven posture collection;
- o standardized querying of additional posture information as needed;
- o and the communication of that data to a centralized server where it can be made available to other components.

Thus, eliminating the need for multiple collection tools on an endpoint collecting the same data for different purposes. Future revisions of this document may include support for the collection of posture information from other endpoint types as well as a standardized interface for storing and querying data in repositories among other capabilities. Additional information about this future work can be found in Section 5 of this document.

To support the collection of posture information from new endpoint types, this document is organized such that it first provides a high-level overview of EPCP as well as the abstract components and transactions that will be realized by implementations (Section 2). This is followed by individual sections that discuss the requirements for specific implementations of the EPCP for a given endpoint type (e.g., traditional workstations and servers, network devices, mobile devices, etc.) along with any extensions for supported use cases (software asset management, vulnerability management, etc.). Over time, the requirements may be expanded to address issues that arise, support new capabilities, or support new implementations beyond IETF NEA and IETF NETCONF.

1.1. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP14 [RFC2119][RFC8174] when, and only when, they appear in all capitals, as shown here.

This specification does not distinguish blocks of informative comments and normative requirements. Therefore, for the sake of clarity, note that lower case instances of must, should, etc. do not indicate normative requirements.

1.2. Terminology

This document uses terms as defined in [I-D.ietf-sacm-terminology] unless otherwise specified.

2. Endpoint Posture Collection Profile

The EPCP describes how IETF, TCG, and ISO/IEC data models, protocols, and interfaces can be used to support the posture assessment of endpoints on a network. This profile does not generate new data models, protocols, or interfaces; rather, it offers requirements for a full end-to-end solution for posture assessment, as well as a fresh perspective on how existing standards can be leveraged against vulnerabilities. Rationale for the EPCP solution as well as the supported and non-supported use cases is available in Appendix A and Appendix B respectively.

The EPCP makes it possible to perform posture assessments against all network-connected endpoints by:

1. uniquely identifying the endpoint;
2. collecting and evaluating posture based on data from the endpoint (asset management, software asset management, vulnerability management, and configuration management);
3. creating a secure, authenticated, confidential channel between the endpoint and the posture manager;
4. enabling the endpoint to notify the posture manager about changes to its configuration;
5. enabling the posture manager to request information about the configuration of the endpoint; and

6. storing the posture information in a repository linked to the identifier for the endpoint.

Furthermore, the EPCP aims to support data storage and data sharing capabilities to make the collected posture information available to authorized parties and components in support of other post-processes (analytic, access control, remediation, reporting, etc.).

2.1. Components

To support posture assessment, data storage, and data sharing capabilities, the EPCP defines several components. Some of these components reside on the target endpoint. Others reside on the posture manager that manages communications with the target endpoint and stores the target endpoint's posture information in a repository.

The primary focus of this document is on the communication between the posture manager and endpoints through the posture collection manager and posture collection engine components. While the orchestrator, evaluator, repository, and API will be discussed in the context of the EPCP, these components are for illustrative purposes only and are not strictly defined nor are requirements provided for them. As a result, vendors are free to implement these components and interfaces in a way that makes the most sense for their products.

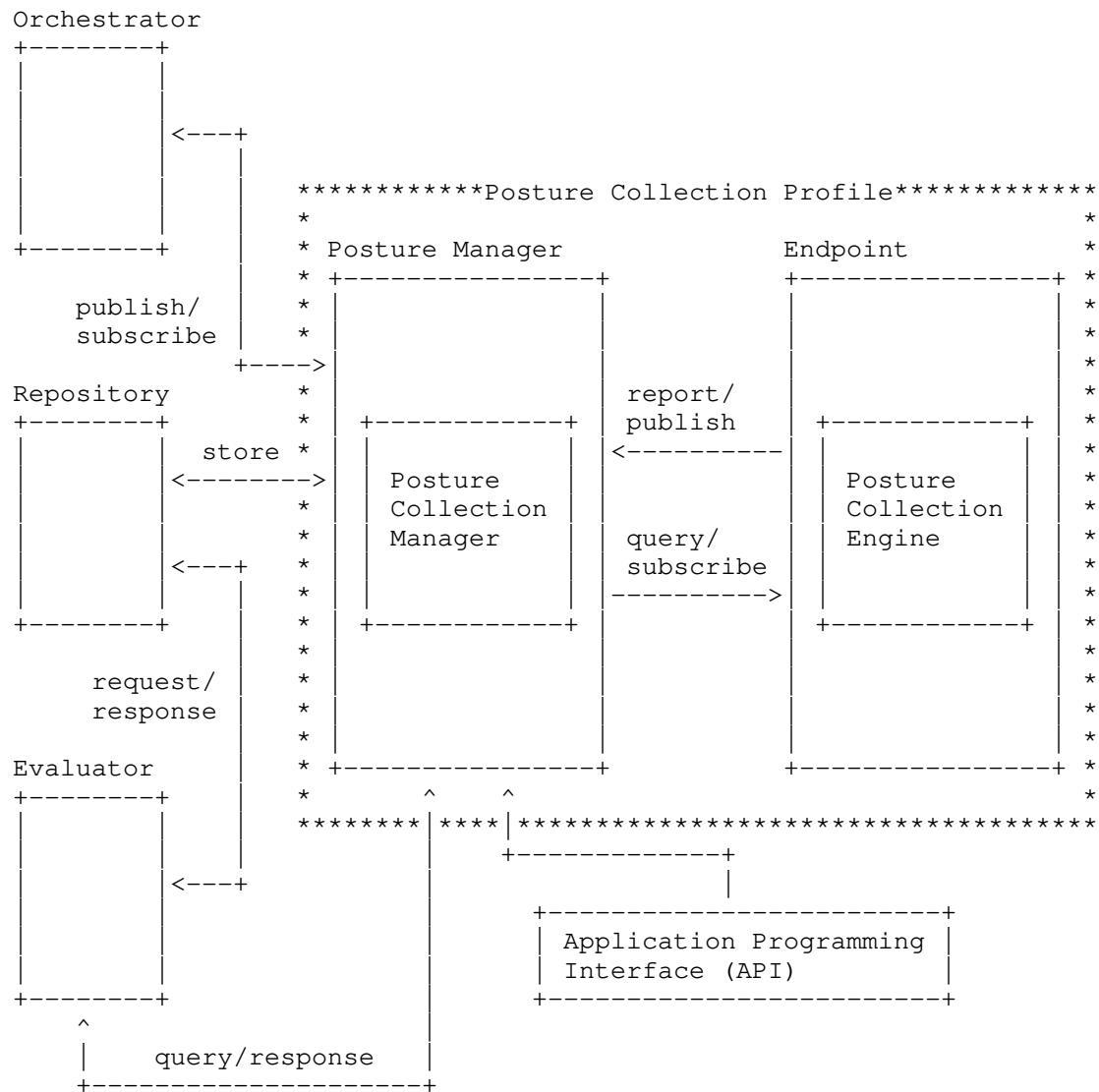


Figure 1: EPCP Components

2.1.1. Endpoint

An endpoint is defined in [RFC6876]. In the EPCP, the endpoint is monitored by the enterprise and is the target of posture assessments. To support these posture assessments, posture information is collected via a posture collection engine.

2.1.1.1. Posture Collection Engine

The posture collection engine is located on the target endpoint and can either push data to the posture collection manager (see Section 2.2.3) or receive queries for data from the posture collection manager (see Section 2.2.4). The posture collection engine sends collected posture information to the posture manager where it can be sanity checked and stored in the repository. The posture collection engine also contains a capability that sets up exchanges between the target endpoint and posture manager. This capability makes the posture collection engine responsible for performing the client-side portion of encryption handshakes, and for locating authorized posture managers with which to communicate.

2.1.2. Posture Manager

The posture manager is an endpoint that collects and validates posture information received about a target endpoint. It also stores the posture information it receives in the repository where it can be retrieved and used in evaluations. The posture manager does not evaluate the posture information.

2.1.2.1. Posture Collection Manager

The posture collection manager is a lightweight and extensible component that facilitates the coordination and execution of posture collection requests using collection mechanisms deployed across the enterprise. The posture collection manager may query and retrieve guidance from the repository to guide the collection of posture information from the target endpoint.

The posture collection manager also contains a capability that sets up exchanges between the target endpoint and the posture manager, and manages data sent to and from the posture collection engine. It is also responsible for performing the server-side portion of encryption handshakes.

If the posture manager wants to register for the continuous collection of endpoint posture changes with the endpoint, then it must do so in a secure and scalable way. Specifically, it will need to create subscriptions with endpoints in a way which allows the posture data to be pushed. Effectively, this means that the target endpoint must be able to establish secure transport connectivity to the posture collection manager as needed, and the posture collection manager must be able to periodically collect the current state of the endpoint and assess its posture.

2.1.3. Repository

The repository hosts guidance, endpoint identification information, and posture information reported by target endpoints where it is made available to authorized components and persisted over a period of time set by the administrator. Information stored in the repository will be accessible to authorized parties via a standardized API. The repository may be a standalone component or may be located on the posture manager. Furthermore, an implementation is not restricted to a single repository and may leverage several repositories to provide this functionality.

2.1.4. Evaluator

The evaluator assesses the posture status of a target endpoint by comparing collected posture information against the desired state of the target endpoint specified in guidance. The evaluator queries and retrieves the appropriate guidance from the repository as well as queries and retrieves the posture information required for the assessment from the repository. If the required posture information is not available in the repository, the evaluator may request the posture information from the posture collection manager, which will result in the collection of additional posture information from the target endpoint. This information is subsequently stored in the repository where it is made available to the evaluator and other components. The results of the assessment are stored in the repository where they are available to tools and administrators for post-processes including follow-up actions, further evaluation, and historical purposes. The evaluator may also be triggered by events on an endpoint or the network.

2.1.5. Orchestrator

The orchestrator provides a publish/subscribe interface for the repository so that infrastructure endpoints can subscribe to and receive published posture assessment results from the repository regarding endpoint posture changes.

2.1.6. Application Programming Interface

The API allows authorized users, infrastructure endpoints, and software to query the repository as well as manage endpoints and other components used in EPCP via the posture manager.

2.2. Transactions

The following sections describe the transactions associated with EPCP components and may be provided in an implementation. The transactions span the deployment of an endpoint, integration into the EPCP, data collection, and the storage and dissemination of that information for different use cases.

2.2.1. Provisioning

An endpoint is provisioned with one or more attributes that will serve as its unique identifier on the network as well as the components (e.g., posture collection engine, etc.) and data models (e.g., SWID) necessary to interact with the posture manager. Examples of such attributes include serial numbers, hardware certificates compliant with [IEEE-802-1ar], and the identities of hardware cryptographic modules among others. An endpoint should also have a MAC address which should change over time. Once provisioning is complete, the endpoint is deployed on the network. Over time, components and data models may need to be added to the endpoint or updated to support the collection needs of an enterprise.

2.2.2. Discovery and Validation

If necessary, the target endpoint finds and validates the posture manager. The posture collection engine on the target endpoint and posture collection manager on the posture manager complete an encryption handshake, during which endpoint identity information is exchanged.

2.2.3. Event-Driven Collection

The posture assessment is initiated when the posture collector engine on the target endpoint notices that relevant posture information on the endpoint has changed. Then, the posture collection engine initiates a posture assessment information exchange with the posture collection manager.

2.2.4. Querying the Endpoint

The posture assessment is initiated by the posture collection manager. This can occur because:

1. policy states that a previous assessment has become invalid, or
2. the posture collection manager is triggered by a sensor or an administrator (via the posture manager's API) that an assessment must be completed.

2.2.5. Data Storage

Once posture information is received by the posture manager, it is forwarded to the repository. The repository could be co-located with the posture manager, or standalone where the repository and posture manager directly communicate with each other or the communication is brokered through the orchestrator. The posture information is stored in the repository along with past posture information collected about the target endpoint.

2.2.6. Data Sharing

Because the target endpoint posture information was sent in standards-based data models over secure, standardized protocols, and then stored in a centralized repository linked to unique endpoint identifiers, authorized parties are able to access the posture information. Such authorized parties may include, but are not limited to, administrators or endpoint owners (via the posture manager's API), evaluators that access the repository directly, and orchestrators that rely on publish/subscribe communications with the repository.

3. IETF NEA EPCP Implementation for Traditional Endpoints

When EPCP is used, posture collectors running on the target endpoint gather posture information as changes occur on the endpoint. The posture information is aggregated by the posture broker client and forwarded to a posture manager, over a secure channel, via the posture transport client. Once received by the posture transport server on the posture manager, the posture information is directed by the posture broker server to the appropriate posture validators where it can be processed and stored in a repository. There the posture information can be used to carry out assessments or other post-processing tasks. Posture collectors can also be queried by posture validators to refresh posture information about the target endpoint or to ask a specific question about posture information. This is shown in Figure 2.

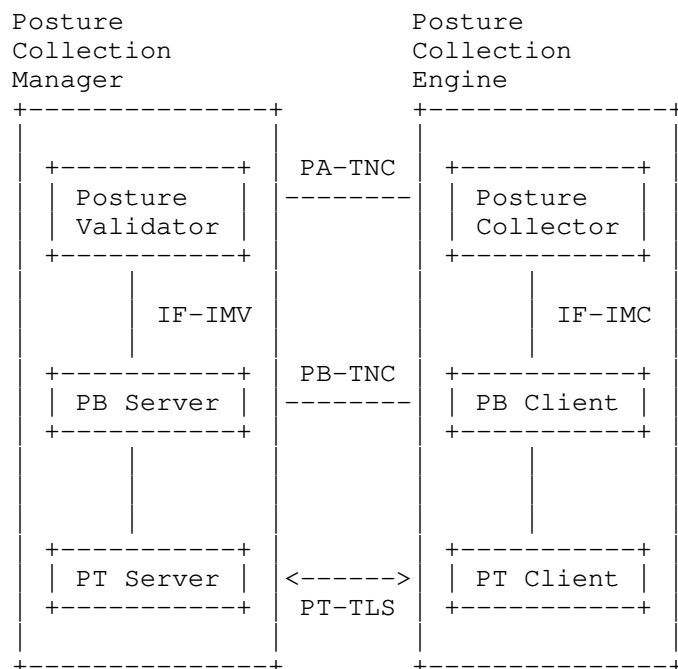


Figure 2: NEA Components

These requirements are written with a view to performing a posture assessment on an endpoint and refer to defined components of the NEA architecture [RFC5209] as well as the IF-IMV [IF-IMV] and IF-IMC [IF-IMC] interfaces defined in the Trusted Computing Group's TNC Work Group. As with the NEA architecture, vendors have discretion as to how these NEA components map to separate pieces of software or endpoints.

It should be noted that the posture broker client and posture transport client components of the posture collection engine and the posture broker server and posture transport server components of the posture collection manager would likely need to be implemented by a single vendor because there are no standardized interfaces between the respective components and would not be interoperable.

Examples of the EPCP as implemented using the components from the NEA architecture are provided in Appendix C.

3.1. Endpoint Provisioning

An endpoint SHOULD be provisioned with a machine certificate that will serve as its unique identifier on the network as well as the components necessary to interact with the posture manager. This includes a posture collection engine to manage requests from the posture manager and the posture collectors necessary to collect the posture information of importance to the enterprise. The endpoint is deployed on the network.

The target endpoint SHOULD authenticate to the posture manager using a machine certificate during the establishment of the outer tunnel achieved with the posture transport protocol defined in [RFC6876]. [IF-IMV] specifies how to pull an endpoint identifier out of a machine certificate. An endpoint identifier SHOULD be created in conformance with [IF-IMV] from a machine certificate sent via [RFC6876].

Other authenticators are possible. The target endpoint MAY authenticate to the posture manager using a combination of the machine account and password; however, this is less secure and not recommended. A more secure approach would leverage a hardware certificate compliant with [IEEE-802-lar]; this identifier SHOULD be associated with the identity of a hardware cryptographic module, in accordance with [IEEE-802-lar], if present on the endpoint. The enterprise SHOULD establish a certificate root authority; install its root certificate on endpoints and on the posture manager; and provision the endpoints and the posture manager with machine certificates.

3.2. Endpoint

The endpoint MUST conform to [RFC5793], which levies several requirements against the endpoint. An endpoint that complies with these requirements will be able to:

1. attempt to initiate a session with the posture manager if the posture makes a request to send an update to the posture manager;
2. notify the posture collector if no PT-TLS session with the posture manager can be created;
3. notify the posture collector when a PT-TLS session is established; and
4. receive information from the posture collectors, forward this information to the posture manager via the posture collection engine.

3.2.1. Posture Collector

Any posture collector used in an EPCP solution MUST be conformant with the TCG TNC Integrity Measurement Collector interface [IF-IMC].

3.2.2. Posture Broker Client

The posture broker client MUST conform to [IF-IMC] to enable communications between the posture broker client and the posture collectors on the endpoint.

3.2.3. Posture Transport Client

The posture transport client MUST implement PT-TLS.

The posture transport client MUST support the use of machine certificates for TLS at each endpoint consistent with the requirements stipulated in [RFC6876] and [Server-Discovery].

The posture transport client MUST be able to locate an authorized posture manager, and switch to a new posture manager when required by the network, in conformance with [Server-Discovery].

3.3. Posture Manager

The posture manager MUST conform to all requirements in [RFC5793].

3.3.1. Posture Validator

Any posture validator used in an EPCP solution MUST be conformant with the TCG TNC Integrity Measurement Verifier interface [IF-IMV].

3.3.2. Posture Broker Server

The posture broker server MUST conform to [IF-IMV]. Conformance to [IF-IMV] enables the posture broker server to obtain endpoint identity information from the posture transport server, and pass this information to any posture validators on the posture manager.

3.3.3. Posture Transport Server

The posture transport server MUST implement PT-TLS.

The posture transport server MUST support the use of machine certificates for TLS at each endpoint consistent with the requirements stipulated in [RFC6876] and [Server-Discovery].

3.4. Repository

EPCP requires a simple interface for the repository. Posture validators on the posture manager receive the target endpoint posture information via PA-TNC [RFC5792] messages sent from corresponding posture collectors on the target endpoint. The posture validators store this information in the repository linked to the identity of the target endpoint where the posture collectors are located.

3.5. IETF SACM Software Asset Management Extension to the IETF NEA EPCP Implementation

This section defines the requirements associated with the Software Inventory Message and Attributes (SWIMA) extension for PA-TNC [RFC8412] in support of the software asset management use case with the IETF NEA EPCP implementation.

3.5.1. Endpoint Pre-Provisioning

The following requirements assume that the platform or OS vendor supports the use of [SWID] and/or [I-D.ietf-sacm-coswid] tags and the standard directory locations for the SWID and CoSWID tags as specified by the [SWID] specification.

3.5.2. SWID Tags

The primary content for the EPCP is the information conveyed in the elements of a SWID or CoSWID tag. The SWID specification defines an XML-based software identification tag and the CoSWID specification defines a Concise Binary Object Representation (CBOR) that is compatible with the SWID specification. CoSWID tags require significantly less memory and bandwidth to store and transmit as compared to the traditional XML-based SWID tags.

For readability, since CoSWID is a concise representation of SWID, only SWID is used throughout the remainder of this document although CoSWID may be used in addition to, or in place of, SWID.

The endpoint MUST have SWID tags stored in a directory specified in [SWID]. The tags SHOULD be provided by the software vendor; they MAY also be generated by:

- o the software installer; or
- o third-party software that creates tags based on the applications it sees installed on the endpoint.

The elements in the SWID tag MUST be populated as specified in [SWID]. These tags, and the directory in which they are stored, MUST be updated as software is added, removed, or updated.

3.5.3. SWID Posture Collectors and Posture Validators

The following sections outline the requirements for SWID Posture Collectors and Posture Validators.

3.5.3.1. The SWID Posture Collector

For the EPCP, the SWID posture collector MUST be conformant with [RFC8412], which includes requirements for:

1. Collecting SWID tags from the SWID directory;
2. Monitoring the SWID directory for changes;
3. Initiating a session with the posture manager to report changes to the directory;
4. Maintaining a list of changes to the SWID directory when updates take place and no PT-TLS connection can be created with the posture manager;
5. Responding to a request for SWID tags from the SWID Posture Validator on the posture manager; and
6. Responding to a query from the SWID posture validator as to whether all updates have been sent.

The SWID posture collector is not responsible for detecting that the SWID directory was not updated when an application was either installed or uninstalled.

3.5.3.2. The SWID Posture Validator

Conformance to [RFC8412] enables the SWID posture validator to:

1. Send messages to the SWID posture collector (at the behest of the administrator at the posture manager console) requesting updates for SWID tags located on endpoint;
2. Ask the SWID posture collector whether all updates to the SWID directory located at the posture manager have been sent; and
3. Perform any validation and processing on the collected SWID posture information prior to storage.

In addition to these requirements, a SWID posture validator used in conformance with this profile MUST be capable of passing this SWID posture information as well as the associated endpoint identity to the repository for storage.

3.5.4. Repository

The interface SHOULD enable an administrator to:

1. Query which endpoints have reported SWID tags for a particular application
2. Query which SWID tags are installed on an endpoint; and
3. Query tags based on characteristics, such as vendor, publisher, etc.

4. IETF NETCONF EPCP Implementation for Network Device Endpoints

When EPCP is used, a NETCONF client that implements the posture collection manager sends a query to target network device endpoint requesting posture information over a secure channel. Once the NETCONF server on the endpoint receives the request, it queries one or more datastores for the posture information. The NETCONF server then reports the information back to the NETCONF client where it can be stored in a repository for use by other tools. This is shown in Figure 3.

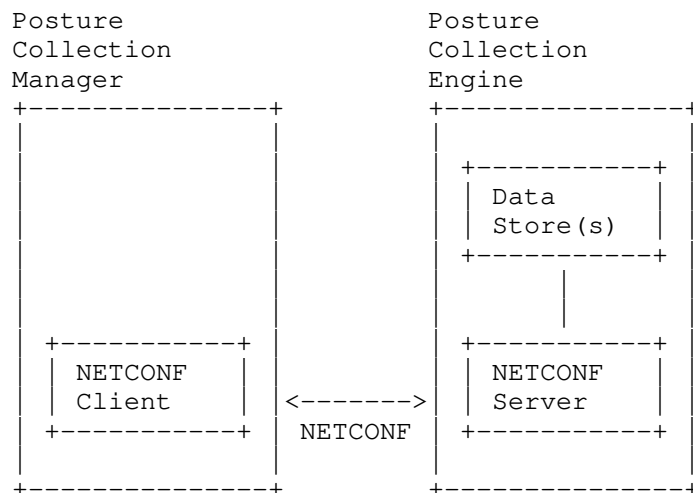


Figure 3: NETCONF Components

These requirements are written with a view to performing a posture assessment on network device endpoints (routers, switches, etc.) and refer to defined components of the NETCONF architecture and map back to EPCP. As with the NETCONF architecture, vendors have discretion as to how these NETCONF components map to separate pieces of software or endpoints.

4.1. Endpoint Provisioning

For the posture manager to be able to query the datastores on the endpoint, the endpoint **MUST** be configured to grant the posture manager access to its datastores as described in [RFC6241]. The posture manager is identified by its NETCONF username. The endpoint is deployed on the network.

4.2. Posture Manager Provisioning

For the posture manager to be able to query the datastores on the endpoint, the posture manager **MUST** be provisioned with a NETCONF username that will be used to authenticate the posture manager to the endpoint as described in [RFC6241]. The username generated will be determined by the selected transport protocol. The posture manager is deployed on the network.

4.3. Endpoint

An endpoint **MUST** conform to the requirements outlined for servers in the NETCONF protocol as defined in [RFC6241]. This requires the implementation of NETCONF over SSH [RFC6242]. An endpoint **MAY** support the NETCONF protocol over other transports such as TLS [RFC7589] as well as the RESTCONF protocol as defined in [RFC8040].

4.3.1. Datastore

A NETCONF datastore on an endpoint **MUST** support the operations outlined in [RFC6241], but, the actual implementation of the datastore is left to the endpoint vendor.

Datastores **MUST** support the YANG data modeling language [RFC7950] for expressing endpoint posture information in a structured format. In addition, datastores **MAY** support other data models such as XML (via YIN) for representing posture information.

Datastores **MUST** support the compliance posture information specified in [RFC7317]. Datastores **MAY** support other models standardized or proprietary as deemed appropriate by the endpoint vendor.

4.4. Posture Manager

A posture manager MUST conform to the requirements specified for clients in the NETCONF protocol as defined in [RFC6241]. This requires the implementation of NETCONF over SSH [RFC6242]. A posture manager MAY also support the NETCONF protocol over other transports such as TLS [RFC7589]. In addition, a posture manager MAY support the RESTCONF protocol as defined in [RFC8040].

4.5. Repository

EPCP requires a simple interface for the repository. The posture collection manager on the posture manager receives the target endpoint posture information via NETCONF [RFC6241] messages sent from posture collection engine on the target endpoint. The posture collection manager stores this information in the repository linked to the identity of the target endpoint from which it was collected.

5. Future Work

This section captures ideas for future work related to EPCP that might be of interest to the IETF SACM WG. These ideas are listed in no particular order.

- o [RFC8639], [RFC8640], and [RFC8641] could be leveraged for an HTTP-based subscription for EPCP. Specifically, it could be used for the posture collection manager to continuously receive posture changes as they happen from the posture collection engine. At this point, it seems like [I-D.ietf-netconf-restconf-notif] would be a good match to these requirements. However, further investigation into the applicability of supporting a RESTCONF server capability to handle subscription requests needs to be made. Specific questions which should be examined include:
 - * Number of endpoints which can be continuously tracked by a single posture collection manager. Scalability questions to be considered include elements from the number of transport connections maintained as well as the volume and churn of posture evidence which will be continuously pushed to the posture collection manager.
 - * Ability of the posture collection manager to establish and maintain a continuous state of endpoint posture during failures. This includes failures/reboots on either side of the interface.
 - * Ability to support the full set of functions described for NETCONF within Section 4.

- o Add support endpoint types beyond workstations, servers, and network infrastructure devices.
- o Examine the integration of [I-D.ietf-mile-xmpp-grid].
- o Define a standard interface and API for interacting with the repository. Requirements to consider include: creating a secure channel between a publisher and the repository, creating a secure channel between a subscriber and the repository, and the types of interactions that must be supported between publishers and subscribers to a repository.
- o Define a standard interface for communications between the posture broker client and posture transport client(s) as well as the posture broker server and posture transport server(s).
- o Retention of posture information on the target endpoint.
- o Define an orchestrator component as well as publish/subscribe interface for it.
- o Define an evaluator component as well as an interface for it.
- o Reassess the use of MAC addresses as a device identifier among network tools, based on technical research into current security best practices in IoT, automotive, mobile, and other privacy-sensitive market domains.

6. Contributors

The authors wish to thank all of those in the TCG TNC work group who contributed to development of the TNC ECP specification [ECP] upon which this document is based.

The authors also wish to give a special thanks to Henk Birkholz, Dan Ehrlich, Ira McDonald, Kathleen Moriarty, David Oliva, and Eric Voit for their thoughtful comments and edits to this document.

7. IANA Considerations

This document does not define any new IANA registries. However, this document does reference other documents that do define IANA registries. As a result, the IANA Considerations section of the referenced documents should be consulted.

8. Security Considerations

This Security Considerations section includes an analysis of the attacks that may be mounted against systems that implement the EPCP (Section 8.1) and the countermeasures that may be used to prevent or mitigate these attacks (Section 8.2). Overall, a substantial reduction in cyber risk can be achieved.

8.1. Threat Model

This section lists the attacks that can be mounted on a NEA implementation of an EPCP environment. The following section (Section 8.2) describes countermeasures.

Because the EPCP describes a specific use case for NEA components, many security considerations for these components are addressed in more detail in the technical specifications: [RFC8412], [IF-IMC], [RFC5793], [Server-Discovery], [RFC6876], [IF-IMV].

8.1.1. Endpoint Attacks

While the EPCP provides substantial improvements in endpoint security, endpoints can still be compromised. For this reason, all parties must regard data coming from endpoints as potentially unreliable or even malicious. An analogy can be drawn with human testimony in an investigation or trial. Human testimony is essential but must be regarded with suspicion.

- o Compromise of endpoint: A compromised endpoint may report false information to confuse or even provide maliciously crafted information with a goal of infecting others.
- o Putting bad information in SWID directory: Even if an endpoint is not completely compromised, some of the software running on it may be unreliable or even malicious. This software, potentially including the SWID generation or discovery tool, or malicious software pretending to be a SWID generation or discovery tool, can place incorrect or maliciously crafted information into the SWID directory. Endpoint users may even place such information in the directory, whether motivated by curiosity or confusion or a desire to bypass restrictions on their use of the endpoint.
- o Identity spoofing (impersonation): A compromised endpoint may attempt to impersonate another endpoint to gain its privileges or to besmirch the reputation of that other endpoint. This is of particular concern when using MAC addresses to identify endpoints, which while widely used in endpoint behavior monitoring and threat assessment tools, are easy to spoof.

8.1.2. Network Attacks

Generally, the network cannot be trusted. A variety of attacks can be mounted using the network, including:

- o Eavesdropping, modification, injection, replay, deletion;
- o Traffic analysis; and
- o Denial of service and blocking traffic.

8.1.3. Posture Manager Attacks

The posture manager is a critical security element and therefore merits considerable scrutiny. A variety of attacks can be leveraged against the Posture Manager.

- o Compromised trusted posture manager: A compromised posture manager or a malicious party that is able to impersonate a posture manager can incorrectly grant or deny access to endpoints, place incorrect information into the repository, or send malicious messages to endpoints.
- o Misconfiguration of posture manager: Accidental or purposeful misconfiguration of a trusted posture manager can cause effects that are similar to those listed for "Compromised trusted posture manager".
- o Malicious untrusted posture manager: An untrusted posture manager cannot mount any significant attacks because all properly implemented endpoints will refuse to engage in any meaningful dialog with such a posture manager.

8.1.4. Repository Attacks

The repository is also an important security element and therefore merits careful scrutiny.

- o Putting bad information into trusted repository: An authorized repository client such as a server may be able to put incorrect information into a trusted repository or delete or modify historical information, causing incorrect decisions about endpoint security. Placing maliciously crafted data in the repository could even lead to the compromise of repository clients, if they fail to carefully check such data.
- o Compromised trusted repository: A compromised trusted repository or a malicious untrusted repository that is able to impersonate a

trusted repository can lead to effects similar to those listed for "Putting bad information into trusted repository". Further, a compromised trusted repository can report different results to different repository clients or deny access to the repository for selected repository clients.

- o Misconfiguration of trusted repository: Accidental or purposeful misconfiguration of a trusted repository can deny access to the repository or result in loss of historical data.
- o Malicious untrusted repository: An untrusted repository cannot mount any significant attacks because all properly implemented repository clients will refuse to engage in any meaningful dialog with such a repository.

8.2. Countermeasures

This section lists the countermeasures that can be used in a NEA implementation of an EPCP environment.

8.2.1. Countermeasures for Endpoint Attacks

This profile is in and of itself a countermeasure for a compromised endpoint. A primary defense for an endpoint is to run up to date software configured to be run as safely as possible.

Ensuring that anti-virus signatures are up to date and that a firewall is configured are also protections for an endpoint that are supported by the current NEA specifications.

For secure device identification and to correlate device identifiers if the MAC address is randomized, MAC addresses should be collected along with other, more secure endpoint identifiers. Endpoints that have hardware cryptographic modules that are provisioned by the enterprise, in accordance with [IEEE-802-1ar], can protect the private keys used for authentication and help prevent adversaries from stealing credentials that can be used for impersonation. Future versions of the EPCP may want to discuss in greater detail how to use a hardware cryptographic module, in accordance with [IEEE-802-1ar], to protect credentials and to protect the integrity of the code that executes during the bootstrap process by hashing or recording indicators of compromise.

8.2.2. Countermeasures for Network Attacks

To address network attacks, [RFC6876] includes required encryption, authentication, integrity protection, and replay protection. [Server-Discovery] also includes authorization checks to ensure that

only authorized servers are trusted by endpoints. Any unspecified or not yet specified network protocols employed in the EPCP (e.g., the protocol used to interface with the repository) should include similar protections.

These protections reduce the scope of the network threat to traffic analysis and denial of service. Countermeasures for traffic analysis (e.g., masking) are usually impractical but may be employed. Countermeasures for denial of service (e.g., detecting and blocking particular sources) SHOULD be used when appropriate to detect and block denial of service attacks. These are routine practices in network security.

8.2.3. Countermeasures for Posture Manager Attacks

Because of the serious consequences of posture manager compromise, posture managers SHOULD be especially well-hardened against attack and minimized to reduce their attack surface. They SHOULD be monitored using the NEA protocols to ensure the integrity of the behavior and analysis data stored on the posture manager and SHOULD utilize an [IEEE-802-1ar]-compliant hardware cryptographic module for identity and/or integrity measurements of the posture manager. They should be well-managed to minimize vulnerabilities in the underlying platform and in systems upon which the posture manager depends. Network security measures such as firewalls or intrusion detection systems may be used to monitor and limit traffic to and from the posture manager. Personnel with administrative access to the posture manager should be carefully screened and monitored to detect problems as soon as possible. Posture manager administrators should not use password-based authentication but should instead use non-reusable credentials and multi-factor authentication (where available). Physical security measures should be employed to prevent physical attacks on posture managers.

To ease detection of posture manager compromise, should it occur, posture manager behavior should be monitored to detect unusual behavior (such as a server reboot, unusual traffic patterns, or other odd behavior). Endpoints should log and/or notify users and/or administrators when peculiar posture manager behavior is detected. To aid forensic investigation, permanent read-only audit logs of security-relevant information pertaining to posture manager (especially administrative actions) should be maintained. If posture manager compromise is detected, the posture manager's certificate should be revoked and careful analysis should be performed of the source and impact of this compromise. Any reusable credentials that may have been compromised should be reissued.

Endpoints can reduce the threat of server compromise by minimizing the number of trusted posture managers, using the mechanisms described in [Server-Discovery].

8.2.4. Countermeasures for Repository Attacks

If the host for the repository is located on its own endpoint, it should be protected with the same measures taken to protect the posture manager. In this circumstance, all messages between the posture manager and repository should be protected with a mature security protocol such as TLS or IPsec.

The repository can aid in the detection of compromised endpoints if an adversary cannot tamper with its contents. For instance, if an endpoint reports that it does not have an application with a known vulnerability installed, an administrator can check whether the endpoint might be lying by querying the repository for the history of what applications were installed on the endpoint.

To help prevent tampering with the information in the repository:

1. Only authorized parties should have privilege to run code on the endpoint and to change the repository.
2. If a separate endpoint hosts the repository, then the functionality of that endpoint should be limited to hosting the repository. The firewall on the repository should only allow access to the posture manager and to any endpoint authorized for administration.
3. The repository should ideally use "write-once" media to archive the history of what was placed in the repository, to include a snapshot of the current status of applications on endpoints.

9. Privacy Considerations

The EPCP specifically addresses the collection of posture data from enterprise endpoints by an enterprise network. As such, privacy is a fundamental concern for those deploying this EPCP solution, given EU GDPR, California CCPA, and many other privacy regulations. The enterprise SHOULD implement and enforce their duty of care.

A possible exception may be the concerns a user may have when attempting to connect a personal endpoint (such as a phone or mobile endpoint) to an enterprise network. The user may not want to share certain details, such as an endpoint identifier or SWID tags, with the enterprise. The user can configure their NEA client to reject requests for this information; however, it is possible that the

enterprise policy will not allow the user's endpoint to connect to the network without providing the requested data.

An enterprise network SHOULD limit access to endpoint posture and identification information to authorized users and SHOULD enforce policies that prevent the export of endpoint posture metadata to unauthorized third parties.

10. References

10.1. Informative References

- [CIS] <http://www.cisecurity.org/controls/>, "CIS Critical Security Controls".
- [DSD] http://www.dsd.gov.au/publications/csocprotect/top_4_mitigations.htm, "Top 4 Mitigation Strategies to Protect Your ICT System", November 2012.
- [ECP] Trusted Computing Group, "TCG Trusted Network Connect Endpoint Compliance Profile, Version 1.10", December 2014.
- [IEEE-802-1ar] Institute of Electrical and Electronics Engineers, "IEEE 802.1ar", December 2009.
- [RFC5209] Sangster, P., Khosravi, H., Mani, M., Narayan, K., and J. Tardo, "Network Endpoint Assessment (NEA): Overview and Requirements", RFC 5209, DOI 10.17487/RFC5209, June 2008, <<https://www.rfc-editor.org/info/rfc5209>>.
- [TNC] Trusted Computing Group, "TCG Trusted Network Connect TNC Architecture for Interoperability, Version 1.5", February 2012.

10.2. Normative References

- [I-D.ietf-mile-xmpp-grid] Cam-Winget, N., Appala, S., Pope, S., and P. Saint-Andre, "Using XMPP for Security Information Exchange", draft-ietf-mile-xmpp-grid-04 (work in progress), October 2017.
- [I-D.ietf-netconf-restconf-notif] Voit, E., Rahman, R., Nilsen-Nygaard, E., Clemm, A., and A. Bierman, "Dynamic subscription to YANG Events and Datastores over RESTCONF", draft-ietf-netconf-restconf-notif-15 (work in progress), June 2019.

- [I-D.ietf-sacm-coswid]
Birkholz, H., Fitzgerald-McKay, J., Schmidt, C., and D. Waltermire, "Concise Software Identification Tags", draft-ietf-sacm-coswid-12 (work in progress), July 2019.
- [I-D.ietf-sacm-terminology]
Waltermire, D., Montville, A., Harrington, D., and N. Cam-Winget, "Terminology for Security Assessment", draft-ietf-sacm-terminology-05 (work in progress), August 2014.
- [IF-IMC] Trusted Computing Group, "TCG Trusted Network Connect TNC IF-IMC, Version 1.3", February 2013.
- [IF-IMV] Trusted Computing Group, "TCG Trusted Network Connect TNC IF-IMV, Version 1.4", December 2014.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5792] Sangster, P. and K. Narayan, "PA-TNC: A Posture Attribute (PA) Protocol Compatible with Trusted Network Connect (TNC)", RFC 5792, DOI 10.17487/RFC5792, March 2010, <<https://www.rfc-editor.org/info/rfc5792>>.
- [RFC5793] Sahita, R., Hanna, S., Hurst, R., and K. Narayan, "PB-TNC: A Posture Broker (PB) Protocol Compatible with Trusted Network Connect (TNC)", RFC 5793, DOI 10.17487/RFC5793, March 2010, <<https://www.rfc-editor.org/info/rfc5793>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC6876] Sangster, P., Cam-Winget, N., and J. Salowey, "A Posture Transport Protocol over TLS (PT-TLS)", RFC 6876, DOI 10.17487/RFC6876, February 2013, <<https://www.rfc-editor.org/info/rfc6876>>.
- [RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <<https://www.rfc-editor.org/info/rfc7317>>.

- [RFC7589] Badra, M., Luchuk, A., and J. Schoenwaelder, "Using the NETCONF Protocol over Transport Layer Security (TLS) with Mutual X.509 Authentication", RFC 7589, DOI 10.17487/RFC7589, June 2015, <<https://www.rfc-editor.org/info/rfc7589>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8412] Schmidt, C., Haynes, D., Coffin, C., Waltermire, D., and J. Fitzgerald-McKay, "Software Inventory Message and Attributes (SWIMA) for PA-TNC", RFC 8412, DOI 10.17487/RFC8412, July 2018, <<https://www.rfc-editor.org/info/rfc8412>>.
- [RFC8639] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Subscription to YANG Notifications", RFC 8639, DOI 10.17487/RFC8639, September 2019, <<https://www.rfc-editor.org/info/rfc8639>>.
- [RFC8640] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Dynamic Subscription to YANG Events and Datastores over NETCONF", RFC 8640, DOI 10.17487/RFC8640, September 2019, <<https://www.rfc-editor.org/info/rfc8640>>.
- [RFC8641] Clemm, A. and E. Voit, "Subscription to YANG Notifications for Datastore Updates", RFC 8641, DOI 10.17487/RFC8641, September 2019, <<https://www.rfc-editor.org/info/rfc8641>>.
- [Server-Discovery]
Trusted Computing Group, "DRAFT: TCG Trusted Network Connect PDP Discovery and Validation, Version 1.0", October 2015.
- [SWID] "Information technology--Software asset management--Part 2: Software identification tag", ISO/IEC 9899:1999, 2009.

Appendix A. Rationale for an EPCP Solution

A.1. Preventative Posture Assessments

The value of continuous endpoint posture assessment is well established. Security experts have identified asset management and vulnerability remediation as a critical step for preventing intrusions. Application whitelisting, patching applications and operating systems, and using the latest versions of applications top the Defense Signals Directorate's "Top 4 Mitigations to Protect Your ICT System". [DSD] "Inventory of Authorized and Unauthorized Endpoints", "Inventory of Authorized and Unauthorized Software", and "Continuous Vulnerability Assessment and Remediation" are Controls 1, 2, and 3, respectively, of the CIS Controls [CIS]. While there are commercially available solutions that attempt to address these security controls, these solutions do not:

- o run on all types of endpoints;
- o consistently interoperate with other tools that could make use of the data collected;
- o collect posture information from all types of endpoints in a consistent, standardized schema;
- o require vetted, standardized protocols that have been evaluated by the international community for cryptographic soundness.

As is true of most solutions offered today, the solution found in the EPCP does not attempt to solve the lying endpoint problem, or detect infected endpoints; rather, it focuses on ensuring that healthy endpoints remain healthy by keeping software up-to-date and patched.

A.2. All Network-Connected Endpoints are Endpoints

As defined by [I-D.ietf-sacm-terminology], an endpoint is any physical or virtual computing endpoint that can be connected to a network. Posture assessment against policy is equally, if not more, important for continuously-connected endpoints, such as enterprise workstations and infrastructure endpoints, as it is for sporadically connected endpoints. Continuously-connected endpoints are just as likely to fall out of compliance with policy, and a standardized posture assessment method is necessary to ensure they can be properly handled.

A.3. All Endpoints on the Network Must be Uniquely Identified

Many administrators struggle to identify what endpoints are connected to the network at any given time. By requiring a standardized method of endpoint identity, the EPCP will enable administrators to answer the basic question, "What is on my network?" In [I-D.ietf-sacm-terminology], SACM defines this set of endpoints on the network as the SACM domain. Unique endpoint identification also enables the comparison of current and past endpoint posture assessments, by allowing administrators to correlate assessments from the same endpoint. This makes it easier to flag suspicious changes in endpoint posture for manual or automatic review, and helps to swiftly identify malicious changes to endpoint applications.

A.4. Standardized Data Models

EPCP requirements prescribe the use of standardized data models for the exchange of posture information. This helps to ensure that the posture information sent from endpoints to the repository can be easily stored, due to their known format, and shared with authorized endpoints and users.

Posture information must be sent over standardized protocols to ensure the confidentiality and authenticity of this data while in transit. Implementations of the EPCP include [RFC6876] and [RFC6241] for communication between the target endpoint and the posture manager. These protocols allow networks that implement this solution to collect large amounts of posture information from an endpoint to make decisions about that endpoint's compliance with some policy. The EPCP offers a solution for all endpoints already connected to the network. Periodic assessments and automated reporting of changes to endpoint posture allow for instantaneous identification of connected endpoints that are no longer compliant with some policy.

A.5. Posture Information Must Be Stored

Posture information must be stored by the repository and must be exposed to an interface at the posture manager. Standardized data models enable standardized queries from an interface exposed to an administrator at the posture manager. A repository must retain any current posture information retrieved from the target endpoint and store it indexed by the unique identifier for the endpoint. Any posture collection manager specified by this profile must be able to ascertain from its corresponding posture collection engine whether the posture information is up to date. An interface on the posture manager must support a request to obtain up-to-date information when an endpoint is connected. This interface must also support the ability to make a standard set of queries about the posture

information stored by the repository. In the future, some forms of posture information might be retained at the endpoint. The interface on the posture manager must accommodate the ability to make a request to the corresponding posture collection engine about the posture of the target endpoint. Standardized data models and protocols also enable the security of posture assessment results. By storing these results indexed under the endpoint's unique identifier, secure storage itself enables endpoint posture information correlation, and ensures that the enterprise's repositories always offer the freshest, most up-to-date view of the enterprise's endpoint posture information possible.

A.6. Posture Information Can Be Shared

By exposing posture information using a standardized interface and API, other security and operational components have a high level of insight into the enterprise's endpoints and the software installed on them. This will support innovation in the areas of asset management, vulnerability scanning, and interfaces, as any authorized infrastructure endpoint can interact with the posture information.

A.7. Enterprise Asset Posture Information Belongs to the Enterprise

Owners and administrators must have complete control of posture information, policy, and endpoint mitigation. Standardized data models, protocols and interfaces help to ensure that this posture information is not locked in proprietary databases, but is made available to its owners. This enables administrators to develop as nuanced a policy as necessary to keep their networks secure. Of course, there may be exceptions to this such as the case with privacy-related information (e.g., personally identifiable information).

Appendix B. EPCP Supported Use Cases and Non-Supported Use Cases

B.1. Supported Use Cases

The following sections describe the different use cases supported by the EPCP.

B.1.1. Hardware Asset Management

Using the API on the posture manager, an authorized user can learn:

- o what endpoints are connected to the network at any given time; and
- o what SWID tags were reported for the endpoints.

The ability to answer these questions offers a standards-based approach to asset management, which is a vital part of enterprise processes such as compliance report generation for the Federal Information Security Modernization Act (FISMA), Payment Card Industry Data Security Standard (PCI DSS), Health Insurance Portability and Accountability Act (HIPAA), etc.

B.1.2. Software Asset Management

The API on the posture manager provides the ability for authorized users and infrastructure to know which software is installed on which endpoints on the enterprise's network. This allows the enterprise to answer questions about what software is installed to determine if it is licensed or prohibited. This information can also drive other use cases such as:

- o vulnerability management: knowing what software is installed supports the ability to determine which endpoints contain vulnerable software and need to be patched.
- o configuration management: knowing which security controls need to be applied to harden installed software and better protect endpoints.

B.1.3. Vulnerability Management

The API also provides the ability for authorized users or infrastructure to locate endpoints running software for which vulnerabilities have been announced. Because of

1. the unique IDs assigned to each endpoint; and
2. the rich application data provided in the endpoints' posture information,

the repository can be queried to find all endpoints running a vulnerable application. Endpoints suspected of being vulnerable can be addressed by the administrator or flagged for further scrutiny.

B.1.4. Threat Detection and Analysis

The repository's standardized API allows authorized infrastructure endpoints and software to search endpoint posture assessment information for evidence that an endpoint's software inventory has changed, and can make endpoint software inventory data available to other endpoints. This automates security data sharing in a way that expedites the correlation of relevant network data, allowing administrators and infrastructure endpoints to identify odd endpoint

behavior and configuration using secure, standardized data models and protocols.

B.2. Non-Supported Use Cases

Several use cases, including but not limited to these, are not covered by the EPCP:

- o Gathering non-standardized types of posture information: The EPCP does not prevent administrators from collecting posture information in proprietary formats from the endpoint; however, it does not set requirements for doing so.
- o Solving the lying endpoint problem: The EPCP does not address the lying endpoint problem; the profile makes no assertions that it can catch an endpoint that is, either maliciously or accidentally, reporting false posture information to the posture manager. However, other solutions may be able to use the posture information collected using the capabilities described in this profile to catch an endpoint in a lie. For example, a sensor may be able to compare the posture information it has collected on an endpoint's activity on the network to what the endpoint reported to the posture manager and flag discrepancies. However, these capabilities are not described in this profile.

Appendix C. Endpoint Posture Collection Profile Examples

The following subsections provide examples of the EPCP as implemented using components from the NEA architecture.

C.1. Continuous Posture Assessment of an Endpoint

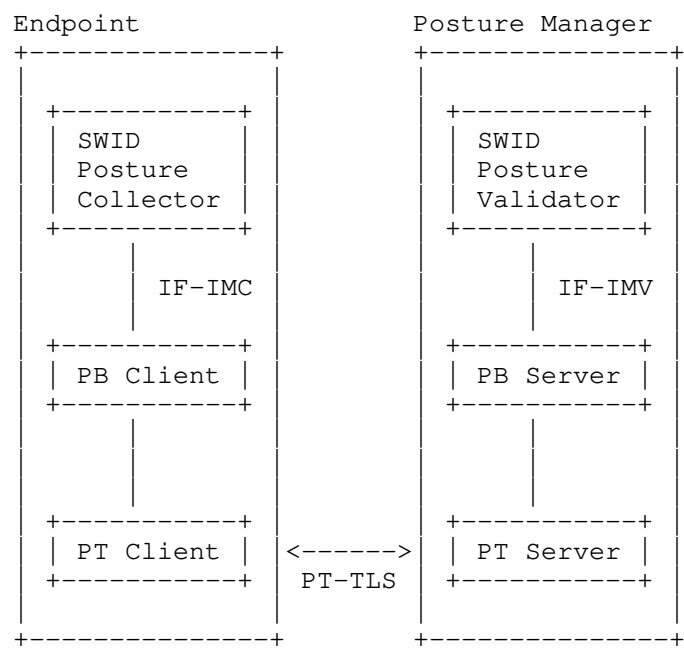


Figure 4: Continuous Posture Assessment of an Endpoint

C.1.1. Change on Endpoint Triggers Posture Assessment

A new application is installed on the endpoint, and the SWID directory is updated. This triggers an update from the SWID posture collector to the SWID posture validator. The message is sent down the NEA stack, encapsulated by NEA protocols until it is sent by the posture transport client to the posture transport server. The posture transport server then forwards it up through the stack, where the layers of encapsulation are removed until the SWID message arrives at the SWID posture validator.

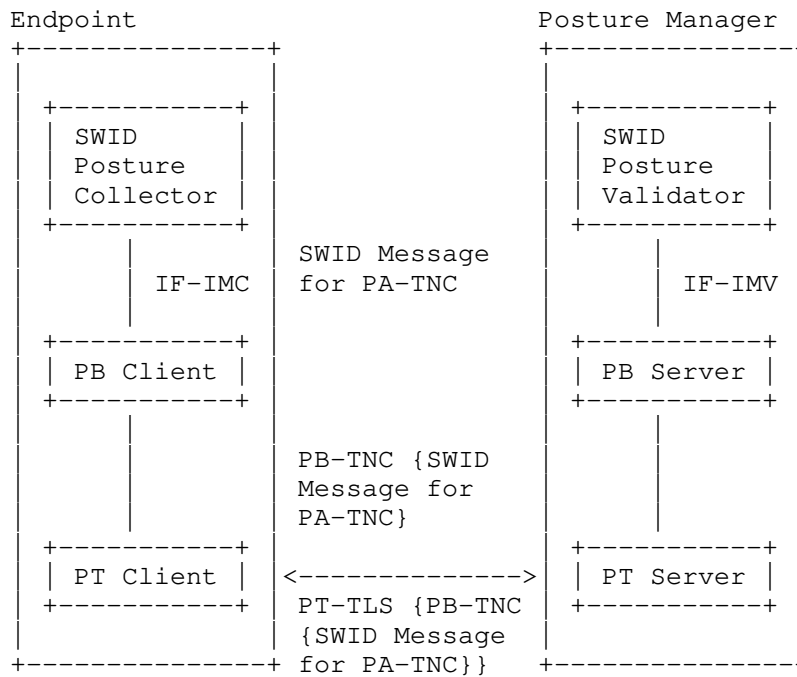


Figure 5: Compliance Protocol Encapsulation

The SWID posture validator stores the new tag information in the repository. If the tag indicates that the endpoint is compliant with the policy, then the process is complete until the next time an update is needed (either because policy states that the endpoint must submit posture assessment results periodically or because an install/uninstall/update event on the endpoint triggers a posture assessment).

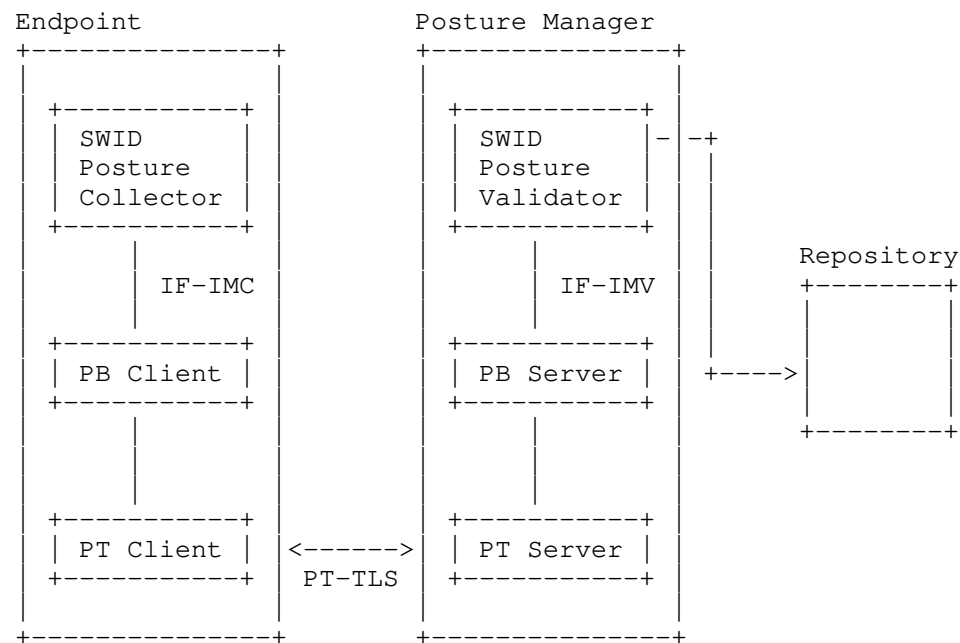


Figure 6: Storing SWIDs in the Repository

If the endpoint has fallen out of compliance with a policy, the posture manager can alert the administrator via the posture manager’s API. The administrator can then take steps to address the problem. If the administrator has already established a policy for automatically addressing this problem, that policy will be followed.

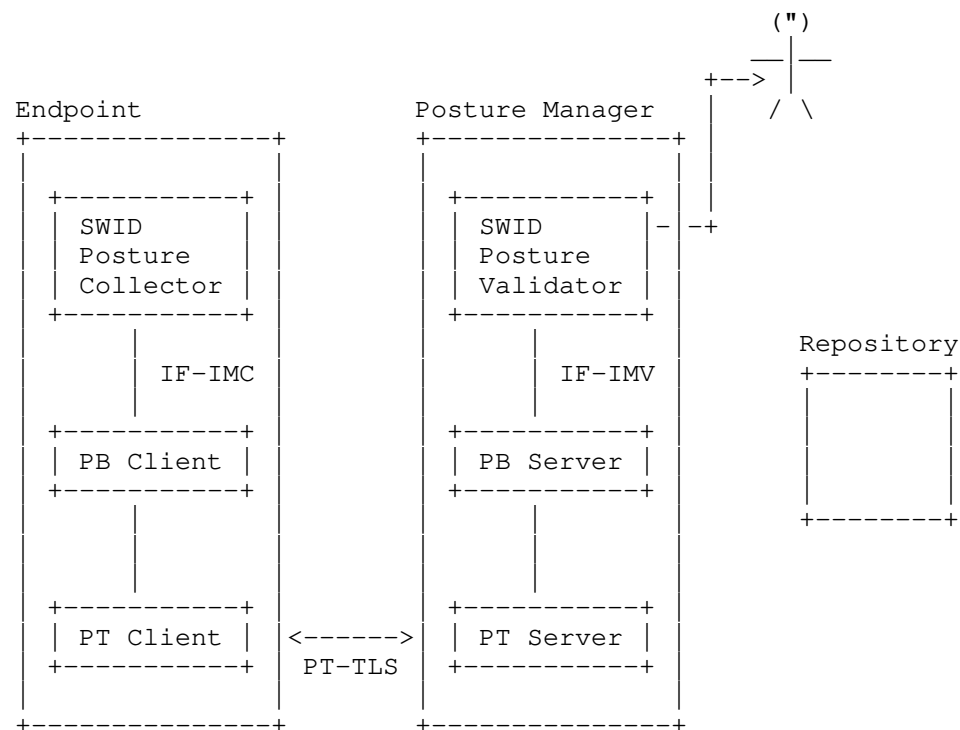


Figure 7: Server Alerts Network Admin

C.2. Administrator Searches for Vulnerable Endpoints

An announcement is made that a particular version of a piece of software has a vulnerability. The administrator uses the API on the posture manager to search the repository for endpoints that reported the SWID tag for the vulnerable software.

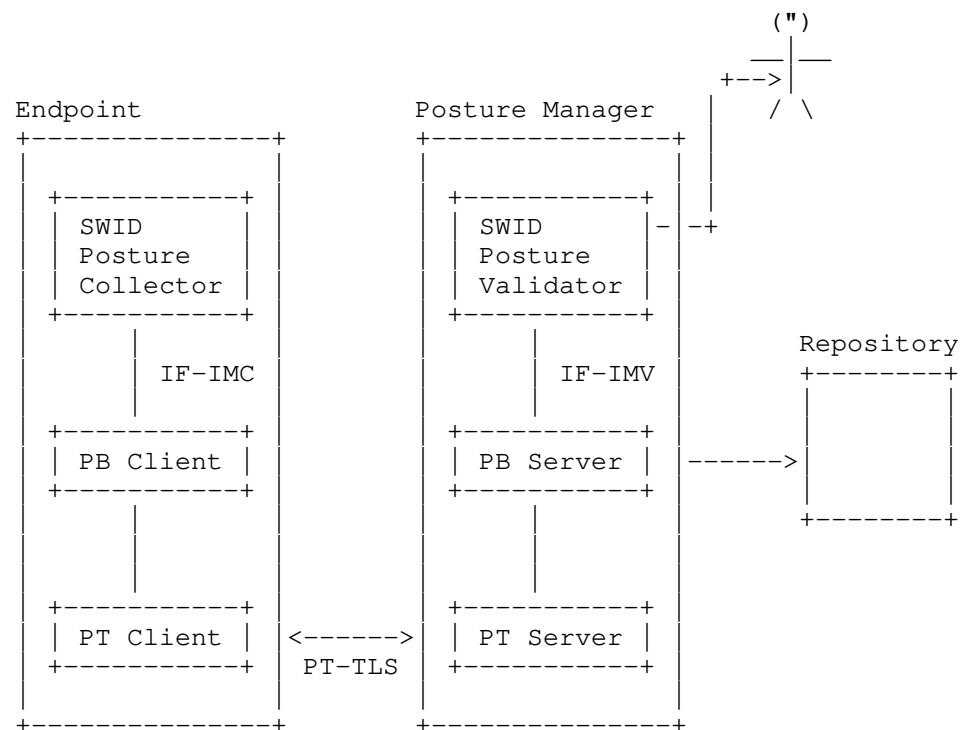


Figure 8: Admin Searches for Vulnerable Endpoints

The repository returns a list of entries matching the administrator’s search. The administrator can then address the vulnerable endpoints by taking some follow-up action such as removing it from the network, quarantining it, or updating the vulnerable software.

Appendix D. Change Log

D.1. -00 to -01

Changed the status of the draft from "Best Current Practices" to "Standards Track".

D.2. -05 to -00

Changed the title of the draft to draft-ietf-sacm-epcp.

Updated the diagram so the Endpoint and Posture Manager are the primary focus of EPCP.

Added a reference to CoSWID in the Software Asset Management extension of the IETF NEA EPCP implementation.

Further clarified the use of MAC addresses in EPCP.

Included a requirement in the Privacy Considerations that the enterprise should exercise due diligence with respect to the privacy of certain data given privacy regulations.

Added a requirement around an endpoint being provisioned with a machine certificate.

Clarified that other protocols and interfaces may be supported beyond IETF NEA and NETCONF.

Made various typographical and editorial changes.

D.3. -04 to -05

Updated the diagram so the Evaluator and Repository are "current work".

Clarified how the Posture Collection Engine can push data, respond to queries, and establish secure transport connectivity for fulfilling subscriptions.

Expanded on the future work around leveraging NETCONF, RESTCONF, and YANG Push for network devices.

Documented the need to reassess MAC addresses as a device identifier.

Made various typographical and editorial changes.

D.4. -03 to -04

Addressed various comments from the SACM WG.

Refactored the document to better focus it on the communications between endpoints and the posture manager and the best practices for EPCP implementations.

Made other editorial changes and improved consistency throughout the document.

D.5. -02 to -03

Addressed various comments from the SACM WG.

Added a reference to TCG ECP 1.0.

Removed text in the "SWID Posture Validator" section that states it performs evaluation. This was removed because it contradicts the posture manager not performing any evaluations.

Expanded the "Provisioning" section of the "EPCP Transactions" section to include examples of endpoint identifiers and the need to provision endpoints with components and data models.

Combined text for the capabilities of the Administrative Interface and API.

Removed superfluous and introductory text from the "Security Considerations" section.

Renamed section "Vulnerability Searches" to Vulnerability Management".

Changed I-D category to BCP.

Changed references to the NETMOD architecture to the NETCONF architecture because NETCONF represents the management protocol whereas NETMOD is focused on the definition of data models.

Addressed various editorial suggestions.

D.6. -01 to -02

Addressed various comments from the SACM WG.

Added a section for the collection of posture information from network devices using standards from the NETMOD WG.

Updated EPCP component diagrams so they were not specific to a NEA-based implementation.

Updated EPCP NEA example diagrams to reflect all the components in the NEA architecture.

D.7. -00 to -01

There are no textual changes associated with this revision. This revision simply reflects a resubmission of the document so that it remains in active status.

D.8. -01 to -02

Added references to the Software Inventory Message and Attributes (SWIMA) for PA-TNC I-D.

Replaced references to PC-TNC with IF-IMC.

Removed erroneous hyphens from a couple of section titles.

Made a few minor editorial changes.

D.9. -02 to -00

Draft adopted by IETF SACM WG.

D.10. -00 to -01

Significant edits to up-level the draft to describe SACM collection over multiple different protocols.

Replaced references to SANS with CIS.

Made other minor editorial changes.

Authors' Addresses

Danny Haynes
The MITRE Corporation
202 Burlington Road
Bedford, MA 01730
USA

Email: dhaynes@mitre.org

Jessica Fitzgerald-McKay
Department of Defense
9800 Savage Road
Ft. Meade, Maryland
USA

Email: jmfitz2@nsa.gov

SACM Working Group
Internet-Draft
Intended status: Informational
Expires: June 17, 2019

H. Birkholz
Fraunhofer SIT
J. Lu
Oracle Corporation
J. Strassner
Huawei Technologies
N. Cam-Winget
Cisco Systems
A. Montville
CIS
December 14, 2018

Security Automation and Continuous Monitoring (SACM) Terminology
draft-ietf-sacm-terminology-16

Abstract

This memo documents terminology used in the documents produced by SACM (Security Automation and Continuous Monitoring).

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on June 17, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terms and Definitions	2
3. IANA Considerations	21
4. Security Considerations	21
5. Acknowledgements	22
6. Change Log	22
7. Contributors	26
8. References	27
8.1. Normative References	28
8.2. Informative References	28
Appendix A. The Attic	29
Authors' Addresses	29

1. Introduction

Our goal with this document is to improve our agreement on the terminology used in documents produced by the IETF Working Group for Security Automation and Continuous Monitoring. Agreeing on terminology should help reach consensus on which problems we're trying to solve, and propose solutions and decide which ones to use.

2. Terms and Definitions

This section describes terms that have been defined by other RFC's and defines new ones. The predefined terms will reference the RFC and where appropriate will be annotated with the specific context by which the term is used in SACM. Note that explanatory or informational augmentation to definitions are segregated from the definitions themselves. The definition for the term immediately follows the term on the same line, whereas expository text is contained in subsequent paragraphs immediately following the definition.

Assertion: Defined by the ITU in [X.1252] as "a statement made by an entity without accompanying evidence of its validity".

In the context of SACM, an assertion is the output of a SACM Component in the form of a SACM Statement (including metadata about the data source and data origin, e.g. timestamps). While the validity of an assertion about Content and Content Metadata cannot be verified without, for example, Integrity Proofing of the

Data Source, an assertion (and therefore a SACM statement, respectively) of the validity of Statement Metadata can be enabled by including corresponding Integrity Evidence created by the Data Origin.

Assessment: Defined in [RFC5209] as "the process of collecting posture for a set of capabilities on the endpoint (e.g., host-based firewall) such that the appropriate validators may evaluate the posture against compliance policy."

Attribute: Is a data element, as defined in [RFC5209], that is atomic.

In the context of SACM, attributes are "atomic" information elements and an equivalent to attribute-value-pairs. Attributes can be components of Subjects, the basic composite definitions that are defined in the SACM Information Model.

Capability: A set of features that are available from a SACM Component.

See also "capability" in [I-D.ietf-i2nsf-terminology].

In the context of SACM, the extent of a SACM component's ability is enabled by the functions it is composed of. Capabilities are registered at a SACM broker (potentially also at a proxy or a repository component if it includes broker functions) by a SACM component via the SACM component registration task and can be discovered by or negotiated with other SACM components via the corresponding tasks. For example, the capability of a SACM provider may be to provide target endpoint records (declarative guidance about well-known or potential target endpoints), or only a subset of that data.

A capability's description is in itself imperative guidance on what functions are exposed to other SACM components in a SACM domain and how to use them in workflows.

The SACM Vulnerability Assessment Scenario [I-D.ietf-sacm-vuln-scenario] defines the terms Endpoint Management Capabilities, Vulnerability Management Capabilities, and Vulnerability Assessment Capabilities, which illustrate specific sets of SACM capabilities on an enterprise IT department's point of view and therefore compose sets of declarative guidance.

Collection Result: Is a composition of one or more content elements carrying information about a target endpoint, that is produced by a collector when conducting a collection task.

Collection Task: A targeted task that collects attributes and/or corresponding attribute values from target endpoint.

There are four types of frequency collection tasks can be conducted with:

ad-hoc, e.g. triggered by a unsolicited query

conditional, e.g. triggered in accordance with policies included in the compositions of workflows

scheduled, e.g. in regular intervals, such as every minute or weekly

continuously, e.g. a network behavior observation

There are three types of collection methods, each requiring an appropriate set of functions to be included in the SACM component conducting the collection task:

Self-Reporting: A SACM component located on the target endpoint itself conducts the collection task.

Remote-Acquisition: A SACM component located on an Endpoint different from the target endpoint conducts the collection task via interfaces available on the target endpoint, e.g. SNMP/NETCONF or WMI.

Behavior-Observation: A SACM component located on an Endpoint different from the target endpoint observes network traffic related to the target endpoint and conducts the collection task via interpretation of that network traffic.

Collector: A piece of software that acquires information about one or more target endpoints by conducting collection tasks.

A collector can be distributed across multiple endpoints, e.g. across a target endpoint and a SACM component. The separate parts of the collector can communicate with a specialized protocol, such as PA-TNC [RFC5792]. At least one part of a distributed collector has to take on the role of a provider of information by providing SACM interfaces to propagate capabilities and to provide SACM content in the form of collection results.

Configuration: A non-volatile subset of the endpoint attributes of a endpoint that is intended to be unaffected by a normal reboot-cycle.

Configuration is a type of imperative guidance that is stored in files (files dedicated to contain configuration and/ or files that are software components), directly on block devices, or on specific hardware components that can be accessed via corresponding software components. Modification of configuration can be conducted manually or automatically via management (plane) interfaces that support management protocols, such as SNMP or WMI. A change of configuration can occur during both run-time and down-time of an endpoint. It is common practice to schedule a change of configuration during or directly after the completion of a boot-cycle via corresponding software components located on the target endpoint itself.

Examples: The static association of an IP address and a MAC address in a DHCP server configuration, a directory-path that identifies a log-file directory, a registry entry.

Configuration Drift: The disposition of endpoint characteristics to change over time.

Configuration drift exists for both hardware components and software components. Typically, the frequency and scale of configuration drift of software components is significantly higher than the configuration drift of hardware components.

Consumer: A SACM Role that requires a SACM Component to include SACM Functions enabling it to receive information from other SACM Components.

Content Element: Content elements constitute the payload data (SACM content) transferred via statement Subjects emitted by providers of information. Every content element Subject includes a specific content Subject and a corresponding content metadata Subject.

Content Metadata: Data about content Subjects. Every content-element includes a content metadata Subject. The Subject can include any information element that can annotate the content transferred. Examples include time stamps or data provenance Subjects.

Control Plane: An architectural component that provides common control functions to all SACM components.

Typically used as a term in the context of routing, e.g. [RFC6192]. SACM components may include authentication, authorization, (capability) discovery or negotiation, registration and subscription. The control plane orchestrates the flow on the data plane according to imperative guidance (i.e. configuration) received via the management plane. SACM components with interfaces to the control plane have knowledge of the capabilities of other SACM components within a SACM domain.

Controller: A controller is a SACM Role that is assigned to a SACM component containing control plane functions managing and facilitating information sharing or execute on security functions.

There are three types of SACM controllers: Broker, Proxy, and Repository. Depending on its type, a controller can also contain functions that have interfaces on the data plane.

Data Confidentiality: Defined in [RFC4949] as "the property that data is not disclosed to system entities unless they have been authorized to know the data."

Data In Motion: Data that is being transported via a network; also referred to as "Data in Transit" or "Data in Flight".

Data in motion requires a data model to transfer the data using a specific encoding. Typically, data in motion is serialized (marshalling) into a transport encoding by a provider of information and deserialized (unmarshalling) by a consumer of information. The termination points of provider of information and consumer of information data is transferred between are interfaces. In regard to data in motion, the interpretation of the roles consumer of information and provider of information depends on the corresponding OSI layer (e.g. on layer2: between interfaces connected to a broadcast domain, on layer4: between interfaces that maintain a TCP connection). In the context of SACM, consumer of information and provider of information are SACM components.

Data At Rest: Data that is stored.

Data at rest requires a data model to encode the data to be stored. In the context of SACM, data at rest located on a SACM component can be provided to other SACM components via discoverable capabilities.

Data Integrity: Defined in [RFC4949] as "the property that data has not been changed, destroyed, or lost in an unauthorized or accidental manner."

Data Origin: The SACM Component that initially acquired or produced data about an endpoint.

Data Origin enables a SACM component to identify the SACM component that initially acquired or produced data about a (target) endpoint (e.g. via collection from a data source) and made it available to a SACM domain via a SACM statement. Data Origin can be expressed by an endpoint label information element (e.g. to be used as metadata in statement).

Data Plane: Is an architectural component providing operational functions enabling information exchange that is not command and control or management related.

Typically used as a term in the context of routing (and used as a synonym for forwarding plane, e.g. [RFC6192]). In the context of SACM, the data plane is an architectural component providing operational functions to enable a SACM component to provide and consume SACM statements and therefore SACM content, which composes the actual SACM content. The data plane in a SACM domain is used to conduct distributed SACM tasks by transporting SACM content via specific transport encodings and corresponding operations defined by SACM data models.

Data Provenance: An historical record of the sources, origins and evolution, as it pertains to data, that is influenced by inputs, entities, functions and processes.

Additional Information - In the context of SACM, data provenance is expressed as metadata that identifies SACM statements and corresponding content elements a new statement is created from. In a downstream process, this references can cascade, creating a data provenance tree that enables SACM components to trace back the original data sources involved in the creation of SACM statements and take into account their characteristics and trustworthiness.

Data Source: Is an endpoint from which a particular set of attributes and/or attribute values have been collected.

Data Source enables a SACM component to identify - and potentially characterize - a (target) endpoint that is claimed to be the original source of endpoint attributes in a SACM statement. Data Source can be expressed as metadata by an endpoint label information element or a corresponding subject of identifying endpoint attributes.

Endpoint: Defined in [RFC5209] as "any computing device that can be connected to a network."

Additional Information - The [RFC5209] definition continues, "Such devices normally are associated with a particular link layer address before joining the network and potentially an IP address once on the network. This includes: laptops, desktops, servers, cell phones, or any device that may have an IP address."

To further clarify the [RFC5209] definition, an endpoint is any physical or virtual device that may have a network address. Note that, network infrastructure devices (e.g. switches, routers, firewalls), which fit the definition, are also considered to be endpoints within this document.

Physical endpoints are always composites that are composed of hardware components and software components. Virtual endpoints are composed entirely of software components and rely on software components that provide functions equivalent to hardware components.

The SACM architecture differentiates two essential categories of endpoints: Endpoints whose security posture is intended to be assessed (target endpoints) and endpoints that are specifically excluded from endpoint posture assessment (excluded endpoints).

Based on the definition of an asset, an endpoint is a type of asset.

Endpoint Attribute: Is a discreet endpoint characteristic that is computably observable.

Endpoint Attributes typically constitute Attributes that can be bundled into Subject (e.g. information about a specific network interface can be represented via a set of multiple AVP).

Endpoint Characteristics: The state, configuration and composition of the software components and (virtual) hardware components a target endpoint is composed of, including observable behavior, e.g. sys-calls, log-files, or PDU emission on a network.

In SACM work-flows, (Target) Endpoint Characteristics are represented via Information Elements.

Endpoint Characterization Task: The task of endpoint characterization that uses endpoint attributes that represent distinct endpoint characteristics.

Endpoint Classification: The categorization of of the endpoint into one or more taxonomic structures.

Endpoint classification requires declarative guidance in the form of an endpoint profile, discovery results and potentially collection results. Types, classes or the characteristics of an individual target endpoint are defined via endpoint profiles.

Endpoint Classification Task: The task of endpoint classification that uses an endpoint's characteristics to determine how to categorize the given endpoint into one or more taxonomic structures.

Endpoint Label: A unique label associated with a unique endpoint.

Endpoint specializations have corresponding endpoint label specializations. For example, an endpoint label used on a SACM Component is a SACM Component Label.

Endpoint Management Capabilities: Enterprise IT management capabilities that are tailored to manage endpoint identity, endpoint information, and associated metadata.

Evaluation Task: A task by which an endpoint's asserted attribute value is evaluated against a policy-compliant attribute value.

Evaluation Result: The resulting value from having evaluated a set of posture attributes.

Expected Endpoint Attribute State: The policy-compliant state of an endpoint attribute that is to be compared against.

Sets of expected endpoint attribute states are transported as declarative guidance in target endpoint profiles via the management plane. This, for example, can be a policy, but also a recorded past state. An expected state is represented by an Attribute or a Subject that represents a set of multiple attribute value pairs.

Guidance: Machine-processable input directing SACM processes or tasks.

Examples of such processes/tasks include automated device management, remediation, collection, evaluation. Guidance influences the behavior of a SACM Component and is considered content of the management plane. In the context of SACM, guidance is machine-readable and can be manually or automatically generated

or provided. Typically, the tasks that provide guidance to SACM components have a low-frequency and tend to be sporadic.

There are two types of guidance:

Declarative Guidance: Guidance that defines the configuration or state an endpoint is supposed to be in, without providing specific actions or methods to produce that desired state. Examples include Target Endpoint Profiles or network topology based requirements.

Imperative Guidance: Guidance that prescribes specific actions to be conducted or methods to be used in order to achieve an outcome. Examples include a targeted Collection Task or the IP-Address of a SACM Component that provides a registration function.

Prominent examples include: modification of the configuration of a SACM component or updating a target endpoint profile that resides on an evaluator. In essence, guidance is transported via the management plane.

Endpoint Hardware Inventory: The set of hardware components that compose a specific endpoint representing its hardware configuration.

Hardware Component: A distinguishable physical component used to compose an endpoint.

The composition of an endpoint can be changed over time by adding or removing hardware components. In essence, every physical endpoint is potentially a composite of multiple hardware components, typically resulting in a hierarchical composition of hardware components. The composition of hardware components is based on interconnects provided by specific hardware types (e.g. FRU in a chassis are connected via redundant busses). In general, a hardware component can be distinguished by its serial number. Occasionally, hardware components are referred to as power sucking aliens.

Information Element: A representation of information about physical and virtual "objects of interest".

Information elements are the building blocks that constitute the SACM information model. In the context of SACM, an information element that expresses a single value with a specific name is referred to as an Attribute (analogous to an attribute-value-pair). A set of attributes that is bundled into a more complex composite information element is referred to as a Subject. Every

information element in the SACM information model has a unique name. Endpoint attributes or time stamps, for example, are represented as information elements in the SACM information model.

Information Model: An abstract representation of data, their properties, relationships between data and the operations that can be performed on the data.

While there is some overlap with a data model, [RFC3444] distinguishes an information model as being protocol and implementation neutral whereas a data model would provide such details. The purpose of the SACM information model is to ensure interoperability between SACM data models (that are used as transport encoding) and to provide a standardized set of information elements for communication between SACM components.

Interaction Model: The definition of specific sequences regarding the exchange of messages (data in motion), including, for example, conditional branching, thresholds and timers.

An interaction model, for example, can be used to define operations, such as registration or discovery, on the control plane. A composition of data models for data in motion and a corresponding interaction model is a protocol.

Internal Collector: A collector that runs on a target endpoint to acquire information from that target endpoint.

Management Plane: An architectural component providing common functions to steer the behavior of SACM components, e.g. their behavior on the control plane.

Typically, a SACM component can fulfill its purpose without continuous input from the management plane. In contrast, without continuous availability of control plane functions a typical SACM component could not function properly. In general, interaction on the management plane is less frequent and less regular than on the control plane. Input via the management plane can be manual (e.g. via a CLI), or can be automated via management plane functions that are part of other SACM components.

Network Address: A layer-specific address that follows a layer-specific address scheme.

The following characteristics are a summary derived from the Common Information Model and ITU-T X.213. Each Network Interface of a specific layer can be associated with one or more addresses appropriate for that layer. There is no guarantee that a network

address is globally unique. A dedicated authority entity can provide a level of assurance that a network address is unique in its given scope. In essence, there is always a scope to a network address, in which it is intended to be unique.

Examples include: physical Ethernet port with a MAC address, layer 2 VLAN interface with a MAC address, layer 3 interface with multiple IPv6 addresses, layer 3 tunnel ingress or egress with an IPv4 address.

Network Interface: An Endpoint is connected to a network via one or more Network Interfaces. Network Interfaces can be physical (Hardware Component) or logical (virtual Hardware component, i.e. a dedicated Software Component). Network Interfaces of an Endpoint can operate on different layers, most prominently what is now commonly called layer 2 and 3. Within a layer, interfaces can be nested.

In SACM, the association of Endpoints and Network Addresses via Network Interfaces is vital to maintain interdependent autonomous processes that can be targeted at Target Endpoints, unambiguously.

Examples include: physical Ethernet port, layer 2 VLAN interface, a MC-LAG setup, layer 3 Point-to-Point tunnel ingress or egress.

Metadata: Data about data.

In the SACM information model, data is referred to as Content. Metadata about the content is referred to as Content-Metadata, respectively. Content and Content-Metadata are combined into Subjects called Content-Elements in the SACM information model. Some information elements defined by the SACM information model can be part of the Content or the Content-Metadata. Therefore, if an information element is considered data or data about data depends on which kind of Subject it is associated with. The SACM information model also defines metadata about the data origin via the Subject Statement-Metadata. Typical examples of metadata are time stamps, data origin or data source.

Posture: Defined in [RFC5209] as "configuration and/or status of hardware or software on an endpoint as it pertains to an organization's security policy."

This term is used within the scope of SACM to represent the configuration and state information that is collected from a target endpoint in the form of endpoint attributes (e.g. software/hardware inventory, configuration settings, dynamically assigned

addresses). This information may constitute one or more posture attributes.

Posture Attributes: Defined in [RFC5209] as "attributes describing the configuration or status (posture) of a feature of the endpoint. A Posture Attribute represents a single property of an observed state. For example, a Posture Attribute might describe the version of the operating system installed on the system."

Within this document this term represents a specific assertion about endpoint configuration or state (e.g. configuration setting, installed software, hardware) represented via endpoint attributes. The phrase "features of the endpoint" highlighted above refers to installed software or software components.

Provider: A provider is a SACM role assigned to a SACM component that provides role-specific functions to provide information to other SACM components.

Repository: A repository is a controller that contains functions to consume, store and provide information of a particular kind.

Such information is typically data transported on the data plane, but potentially also data and metadata from the control and management plane. A single repository may provide the functions of more than one specific repository type (i.e. configuration baseline repository, assessment results repository, etc.)

SACM Broker Controller: A SACM Broker Controller is a controller that contains control plane functions to provide and/or connect services on behalf of other SACM components via interfaces on the control plane.

A broker may provide, for example, authorization services and find, upon request, SACM components providing requested services.

SACM Component: Is a component, as defined in [I-D.ietf-i2nsf-terminology], that is composed of SACM capabilities.

In the context of SACM, a set of SACM functions composes a SACM component. A SACM component conducts SACM tasks, acting on control plane, data plane and/or management plane via corresponding SACM interfaces. SACM defines a set of standard components (e.g. a collector, a broker, or a data store). A SACM component contains at least a basic set of control plane functions and can contain data plane and management plane functions. A SACM component residing on an endpoint assigns one or more SACM roles

to the corresponding endpoint due to the SACM functions it is composed of. A SACM component "resides on" an endpoint and an endpoint "contains" a SACM component, correspondingly. For example, a SACM component that is composed solely of functions that provide information would only take on the role of a provider.

SACM Component Discovery: The task of discovering the capabilities provided by SACM components within a SACM domain.

This is likely to be performed via an appropriate set of control plane functions.

SACM Component Label: A specific endpoint label that is used to identify a SACM component.

In content-metadata, this label is called data origin.

SACM Content: The payload provided by SACM components to the SACM domain on the data plane.

SACM content includes the SACM data models.

SACM Domain: Endpoints that include a SACM component compose a SACM domain.

(To be revised, additional definition content TBD, possible dependencies to SACM architecture)

SACM Function: A behavioral aspect of a SACM component that provides external SACM Interfaces or internal interfaces to other SACM Functionse.

For example, a SACM Function with SACM Interfaces on the Control Plane can provide a brokering function to other SACM Components. Via Data Plane interfaces, a SACM Function can act as a provider and/or as a consumer of information. SACM Functions can be propagated as the Capabilities of a SACM Component and can be discovered by or negotiated with other SACM Components.

SACM Interface: An interface, as defined in [I-D.ietf-i2nsf-terminology], that provides SACM-specific operations.

[I-D.ietf-i2nsf-terminology] defines interface as a "set of operations one object knows it can invoke on, and expose to, another object," and further defines interface by stating that an interface "decouples the implementation of the operation from its

specification. An interface is a subset of all operations that a given object implements. The same object may have multiple types of interfaces to serve different purposes."

In the context of SACM, SACM Functions provide SACM Interfaces on the management, control, or data plane. Operations a SACM Interface provides are based on corresponding data model defined by SACM. SACM Interfaces are used for communication between SACM components.

SACM Proxy Controller: A SACM Proxy Controller is a controller that provides data plane and control plane functions, information, or services on behalf of another component, which is not directly participating in the SACM architecture.

SACM Role: Is a role, as defined in [I-D.ietf-i2nsf-terminology], that requires the SACM Component assuming the role to bear a set of SACM functions or interfaces.

SACM Roles provide three important benefits. First, it enables different behavior to be supported by the same Component for different contexts. Second, it enables the behavior of a Component to be adjusted dynamically (i.e., at runtime, in response) to changes in context, by using one or more Roles to define the behavior desired for each context. Third, it decouples the Roles of a Component from the Applications that use that Component."

In the context of SACM, SACM roles are associated with SACM components and are defined by the set of functions and interfaces a SACM component includes. There are three SACM roles: provider, consumer, and controller. The roles associated with a SACM component are determined by the purpose of the SACM functions and corresponding SACM interfaces the SACM component is composed of.

SACM Statement: Is an assertion that is made by a SACM Component.

Security Automation: The process of which security alerts can be automated through the use of different components to monitor, analyze and assess endpoints and network traffic for the purposes of detecting misconfigurations, misbehaviors or threats.

Security Automation is intended to identify target endpoints that cannot be trusted (see "trusted" in [RFC4949]). This goal is achieved by creating and processing evidence (assessment statements) that a target endpoint is not a trusted system [RFC4949].

Software Package: A generic software package (e.g. a text editor).

Software Component: A software package installed on an endpoint.

The software component may include a unique serial number (e.g. a text editor associated with a unique license key).

Software Instance: A running instance of a software component.

For example, on a multi-user system, one logged-in user has one instance of a text editor running and another logged-in user has another instance of the same text editor running, or on a single-user system, a user could have multiple independent instances of the same text editor running.

State: A volatile set of endpoint attributes of a (target) endpoint that is affected by a reboot-cycle.

Local state is created by the interaction of components with other components via the control plane, via processing data plane payload, or via the functional properties of local hardware and software components. Dynamic configuration (e.g. IP address distributed dynamically via an address distribution and management services, such as DHCP) is considered state that is the result of the interaction with another component (e.g. provided by a DHCP server with a specific configuration).

Examples: The static association of an IP address and a MAC address in a DHCP server configuration, a directory-path that identifies a log-file directory, a registry entry.

Statement: A statement is the root/top-level subject defined in the SACM information model.

A statement is used to bundle Content Elements into one subject and includes metadata about the data origin.

Subject: A semantic composite information element pertaining to a system entity that is a target endpoint.

Like Attributes, subjects have a name and are composed of attributes and/or other subjects. Every IE that is part of a subject can have a quantity associated with it (e.g. zero-one, none-unbounded). The content IE of a subject can be an unordered or an ordered list.

In contrast to the definitions of subject provided by [RFC4949], a subject in the scope of SACM is neither "a system entity that

causes information to flow among objects or changes the system state" nor "a name of a system entity that is bound to the data items in a digital certificate".

In the context of SACM, a subject is a semantic composite of information elements about a system entity that is a target endpoint. Every acquirable subject-as defined in the scope of SACM-about a target endpoint represents and therefore identifies every subject-as defined by [RFC4949]-that is a component of that target endpoint. The semantic difference between both definitions can be subtle in practice and is in consequence important to highlight.

Supplicant: A component seeking to be authenticated via the control plane for the purpose of participating in a SACM domain.

System Resource: Defined in [RFC4949] as "data contained in an information system; or a service provided by a system; or a system capacity, such as processing power or communication bandwidth; or an item of system equipment (i.e., hardware, firmware, software, or documentation); or a facility that houses system operations and equipment."

Target Endpoint: Is an endpoint that is under assessment at some point in, or region of, time.

Every endpoint that is not specifically designated as an excluded endpoint is a target endpoint. A target endpoint is not part of a SACM domain unless it contains a SACM component (e.g. a SACM component that publishes collection results coming from an internal collector).

A target endpoint is similar to a device that is a Target of Evaluation (TOE) as defined in Common Criteria and as referenced by {{RFC4949}}.

Target Endpoint Address: An address that is layer specific and which follows layer specific address schemes.

Each interface of a specific layer can be associated with one or more addresses appropriate for that layer. There is no guarantee that an address is globally unique. In general, there is a scope to an address in which it is intended to be unique.

Examples include: physical Ethernet port with a MAC address, layer 2 VLAN interface with a MAC address, layer 3 interface with multiple IPv6 addresses, layer 3 tunnel ingress or egress with an IPv4 address.

Target Endpoint Characterization: The description of the distinctive nature of a target endpoint, that is based on its characteristics.

Target Endpoint Characterization Record: A set of endpoint attributes about a target endpoint that was encountered in a SACM domain, which are associated with that target endpoint as a result of a Target Endpoint Characterization Task.

A characterization record is intended to be a representation of an endpoint. It cannot be assured that a record distinctly represents a single target endpoint unless a set of one or more endpoint attributes that compose a unique set of identifying endpoint attributes are included in the record. Otherwise, the set of identifying attributes included in a record can match more than one target endpoints, which are - in consequence - indistinguishable to a SACM domain until more qualifying endpoint attributes can be acquired and added to the record. A characterization record is maintained over time in order to assert that acquired endpoint attributes are either about an endpoint that was encountered before or an endpoint that has not been encountered before in a SACM domain. A characterization record can include, for example, acquired configuration, state or observed behavior of a specific target endpoint. Multiple and even conflicting instances of this information can be included in a characterization record by using timestamps and/or data origins to differentiate them. The endpoint attributes included in a characterization record can be used to re-identify a distinct target endpoint over time. Classes or profiles can be associated with a characterization record via the Classification Task in order to guide collection, evaluation or remediation tasks.

Target Endpoint Characterization Task: An ongoing task of continuously adding acquired endpoint attributes to a corresponding record. The TE characterization task manages the representation of encountered target endpoints in the SACM domain in the form of characterization records. For example, the output of a target endpoint discovery task or a collection task can be processed by the characterization task and added to the record. The TE characterization Task also manages these representations of target endpoints encountered in the SACM domain by splitting or merging the corresponding records as new or more refined endpoint attributes become available.

Target Endpoint Classification Task: The task of associating a class from an extensible list of classes with an endpoint characterization record. TE classes function as imperative and declarative guidance for collection, evaluation, remediation and security posture assessment in general.

Target Endpoint Discovery Task: The ongoing task of detecting previously unknown interaction of a potential target endpoint in the SACM domain. TE Discovery is not directly targeted at a specific target endpoint and therefore an un-targeted task. SACM Components conducting the discovery task as a part of their function are typically distributed and located, for example, on infrastructure components or collect from those remotely via appropriate interfaces. Examples of infrastructure components that are of interest to the discovery task include routers, switches, VM hosting or VM managing components, AAA servers, or servers handling dynamic address distribution.

Target Endpoint Identifier: The target endpoint discovery task and the collection tasks can result in a set of identifying endpoint attributes added to a corresponding Characterization Record. This subset of the endpoint attributes included in the record is used as a target endpoint identifier, by which a specific target endpoint can be referenced. Depending on the available identifying attributes, this reference can be ambiguous and is a "best-effort" mechanism. Every distinct set of identifying endpoint attributes can be associated with a target endpoint label that is unique in a SACM domain.

Target Endpoint Label: An endpoint label that identifies a specific target endpoint.

Target Endpoint Profile: A bundle of expected or desired component composition, configurations and states that is associated with a target endpoint.

The corresponding task by which the association with a target endpoint takes places is the endpoint classification task. The task by which an endpoint profile is created is the endpoint characterization task. A type or class of target endpoints can be defined via a target endpoint profile. Examples include: printers, smartphones, or an office PC.

In respect to [RFC4949], a target endpoint profile is a protection profile as defined by Common Criteria (analogous to the target endpoint being the target of evaluation).

SACM Task: Is a task conducted within the scope of a SACM domain by one or more SACM functions that achieves a SACM-defined outcome.

A SACM task can be triggered by other operations or functions (e.g. a query from another SACM component or an unsolicited push on the data plane due to an ongoing subscription). A task is part of a SACM process chain. A task starts at a given point in time

and ends in a deterministic state. With the exception of a collection task, a SACM task consumes SACM statements provided by other SACM components. The output of a task is a result that can be provided (e.g. published) on the data plane.

The following tasks are defined by SACM:

Target Endpoint Discovery

Target Endpoint Characterization

Target Endpoint Classification

Collection

Evaluation [TBD]

Information Sharing [TBD]

SACM Component Discovery

SACM Component Authentication [TBD]

SACM Component Authorization [TBD]

SACM Component Registration [TBD]

Timestamps : Defined in [RFC4949] as "with respect to a data object, a label or marking in which is recorded the time (time of day or other instant of elapsed time) at which the label or marking was affixed to the data object".

A timestamp always requires context, i.e. additional information elements that are associated with it. Therefore, all timestamps wrt information elements are always metadata. Timestamps in SACM Content Elements may be generated outside a SACM Domain and may be encoded in an unknown representation. Inside a SACM domain the representation of timestamps is well-defined and unambiguous.

Virtual Endpoint: An endpoint composed entirely of logical system components (see [RFC4949]).

The most common example is a virtual machine/host running on a target endpoint. Effectively, target endpoints can be nested and at the time of this writing the most common example of target endpoint characteristics about virtual components is the EntLogicalEntry in [RFC6933].

Vulnerability Assessment: An assessment specifically tailored to determining whether a set of endpoints is vulnerable according to the information contained in the vulnerability description information.

Vulnerability Description Information: Information pertaining to the existence of a flaw or flaws in software, hardware, and/or firmware, which could potentially have an adverse impact on enterprise IT functionality and/or security.

Vulnerability description information should contain enough information to support vulnerability detection.

Vulnerability Detection Data: A type of imperative guidance extracted or derived from vulnerability description information that describes the specific mechanisms of vulnerability detection that is used by an enterprise's vulnerability management capabilities to determine if a vulnerability is present on an endpoint.

Vulnerability Management Capabilities: An IT management capability tailored toward managing endpoint vulnerabilities and associated metadata on an ongoing basis by ingesting vulnerability description information and vulnerability detection data, and performing vulnerability assessments.

Vulnerability assessment capabilities: An assessment capability that is tailored toward determining whether a set of endpoints is vulnerable according to vulnerability description information.

Workflow: A workflow is a modular composition of tasks that can contain loops, conditionals, multiple starting points and multiple endpoints.

The most prominent workflow in SACM is the assessment workflow.

3. IANA Considerations

This memo includes no request to IANA.

4. Security Considerations

This memo documents terminology for security automation. While it is about security, it does not affect security.

5. Acknowledgements

6. Change Log

Changes from version 00 to version 01:

- o Added simple list of terms extracted from UC draft -05. It is expected that comments will be received on this list of terms as to whether they should be kept in this document. Those that are kept will be appropriately defined or cited.

Changes from version 01 to version 02:

- o Added Vulnerability, Vulnerability Management, xposure, Misconfiguration, and Software flaw.

Changes from version 02 to version 03:

- o Removed Section 2.1. Cleaned up some editing nits; broke terms into 2 sections (predefined and newly defined terms). Added some of the relevant terms per the proposed list discussed in the IETF 89 meeting.

Changes from version 03 to version 04:

- o TODO

Changes from version 04 to version 05:

- o TODO

Changes from version 05 to version 06:

- o Updated author information.
- o Combined "Pre-defined Terms" with "New Terms and Definitions".
- o Removed "Requirements language".
- o Removed unused reference to use case draft; resulted in removal of normative references.
- o Removed introductory text from Section 1 indicating that this document is intended to be temporary.
- o Added placeholders for missing change log entries.

Changes from version 06 to version 07:

- o Added Contributors section.
- o Updated author list.
- o Changed title from "Terminology for Security Assessment" to "Secure Automation and Continuous Monitoring (SACM) Terminology".
- o Changed abbrev from "SACM-Terms" to "SACM Terminology".
- o Added appendix The Attic to stash terms for future updates.
- o Added Authentication, Authorization, Data Confidentiality, Data Integrity, Data Origin, Data Provenance, SACM Component, SACM Component Discovery, Target Endpoint Discovery.
- o Major updates to Building Block, Function, SACM Role, Target Endpoint.
- o Minor updates to Broker, Capability, Collection Task, Evaluation Task, Posture.
- o Relabeled Role to SACM Role, Endpoint Target to Target Endpoint, Endpoint Discovery to Endpoint Identification.
- o Moved Asset Targeting, Client, Endpoint Identification to The Attic.
- o Endpoint Attributes added as a TODO.
- o Changed the structure of the Change Log.

Changes from version 07 to version 08:

- o Added Assertion, Collection Result, Collector, Excluded Endpoint, Internal Collector, Network Address, Network Interface, SACM Domain, Statement, Target Endpoint Identifier, Target Endpoint Label, Timestamp.
- o Major updates to Attributes, Broker, Collection Task, Consumer, Controller, Control Plane, Endpoint Attributes, Expected Endpoint State, SACM Function, Provider, Proxy, Repository, SACM Role, Target Endpoint.
- o Minor updates to Asset, Building Block, Data Origin, Data Source, Data Provenance, Endpoint, Management Plane, Posture, Posture Attribute, SACM Component, SACM Component Discovery, Target Endpoint Discovery.

- o Relabeled Function to SACM Function.

Changes from version 08 to version 09:

- o Updated author list.
- o Added Data Plane, Endpoint Characterization, Endpoint Classification, Guidance, Interaction Model, Software Component, Software Instance, Software Package, Statement, Target Endpoint Profile, SACM Task.
- o Removed Building Block.
- o Major updates to Control Plane, Endpoint Attribute, Expected Endpoint State, Information Model, Management Plane.
- o Minor updates to Attribute, Capabilities, SACM Function, SACM Component, Collection Task.
- o Moved Asset Characterization to The Attic.

Changes from version 09 to version 10:

- o Added Configuration Drift, Data in Motion, Data at Rest, Endpoint Management Capability, Hardware Component, Hardware Inventory, Hardware Type, SACM Interface, Target Endpoint Characterization Record, Target Endpoint Characterization Task, Target Endpoint Classification Task, Target Endpoint Discovery Task, Vulnerability Description Information, Vulnerability Detection Data, Vulnerability Management Capability, Vulnerability Assessment
- o Added references to i2nsf definitions in Capability, SACM Component, SACM Interface, SACM Role.
- o Added i2nsf Terminology I-D Reference.
- o Major Updates to Endpoint, SACM Task, Target Endpoint Identifier.
- o Minor Updates to Guidance, SACM Component Discovery, Target Endpoint Label, Target Endpoint Profile.
- o Relabeled SACM Task
- o Removed Target Endpoint Discovery

Changes from version 10 to version 11:

- o Added Content Element, Content Metadata, Endpoint Label, Information Element, Metadata, SACM Component Label, Workflow.
- o Major Updates to Assessment, Capability, Collector, Endpoint Management Capabilities, Guidance, Vulnerability Assessment Capabilities, Vulnerability Detection Data, Vulnerability Assessment Capabilities.
- o Minor updates to Collection Result, Control Plane, Data in Motion, Data at Rest, Data Origin, Network Interface, Statement, Target Endpoint Label.
- o Relabeled Endpoint Management Capability, Vulnerability Management Capability, Vulnerability Assessment.

Changes from version 11 to version 12:

- o Added Configuration, Endpoint Characteristic, Event, SACM Content, State, Subject.
- o Major Updates to Assertion, Data in Motion, Data Provenance, Data Source, Interaction Model.
- o Minor Updates to Attribute, Control Plane, Data Origin, Data Provenance, Expected Endpoint State, Guidance, Target Endpoint Classification Task, Vulnerability Detection Data.

Changes from version 12 to version 13:

- o Added Virtual Component.
- o Major Updates to Capability, Collection Task, Hardware Component, Hardware Type, Security Automation, Subject, Target Endpoint, Target Endpoint Profile.
- o Minor Updates to Assertion, Data Plane, Endpoint Characteristics.

Changes from version 13 to version 14:

- o Handled a plethora of issues listed in GitHub.
- o Pruned some commonly understood terms.
- o Narrowing term labels per their definitions.
- o In some cases, excised expositional text.

- o Where expositional text was left intact, it has been separated from the actual definition of a term.

Changes from version 14 to version 16:

- o moved obsolete definitions into the Appendix (attic).

7. Contributors

David Waltermire
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, MD 20877
USA

Email: david.waltermire@nist.gov

Adam W. Montville
Center for Internet Security
31 Tech Valley Drive
East Greenbush, NY 12061
USA

Email: adam.w.montville@gmail.com

David Harrington
Effective Software
50 Harding Rd
Portsmouth, NH 03801
USA

Email: ietfdbh@comcast.net

Brian Ford
Lancope
3650 Brookside Parkway, Suite 500
Alpharetta, GA 30022
USA

Email: bford@lancope.com

Merike Kaeo
Double Shot Security
3518 Fremont Avenue North, Suite 363
Seattle, WA 98103
USA

Email: merike@doubleshotsecurity.com

8. References

8.1. Normative References

- [RFC5792] Sangster, P. and K. Narayan, "PA-TNC: A Posture Attribute (PA) Protocol Compatible with Trusted Network Connect (TNC)", RFC 5792, DOI 10.17487/RFC5792, March 2010, <<https://www.rfc-editor.org/info/rfc5792>>.
- [RFC6933] Bierman, A., Romascanu, D., Quittek, J., and M. Chandramouli, "Entity MIB (Version 4)", RFC 6933, DOI 10.17487/RFC6933, May 2013, <<https://www.rfc-editor.org/info/rfc6933>>.

8.2. Informative References

- [I-D.ietf-i2nsf-terminology]
Hares, S., Strassner, J., Lopez, D., Xia, L., and H. Birkholz, "Interface to Network Security Functions (I2NSF) Terminology", draft-ietf-i2nsf-terminology-06 (work in progress), July 2018.
- [I-D.ietf-netmod-entity]
Bierman, A., Bjorklund, M., Dong, J., and D. Romascanu, "A YANG Data Model for Hardware Management", draft-ietf-netmod-entity-08 (work in progress), January 2018.
- [I-D.ietf-sacm-vuln-scenario]
Coffin, C., Cheikes, B., Schmidt, C., Haynes, D., Fitzgerald-McKay, J., and D. Waltermire, "SACM Vulnerability Assessment Scenario", draft-ietf-sacm-vuln-scenario-02 (work in progress), September 2016.
- [RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between Information Models and Data Models", RFC 3444, DOI 10.17487/RFC3444, January 2003, <<https://www.rfc-editor.org/info/rfc3444>>.
- [RFC4949] Shirey, R., "Internet Security Glossary, Version 2", FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007, <<https://www.rfc-editor.org/info/rfc4949>>.
- [RFC5209] Sangster, P., Khosravi, H., Mani, M., Narayan, K., and J. Tardo, "Network Endpoint Assessment (NEA): Overview and Requirements", RFC 5209, DOI 10.17487/RFC5209, June 2008, <<https://www.rfc-editor.org/info/rfc5209>>.
- [RFC6192] Dugal, D., Pignataro, C., and R. Dunn, "Protecting the Router Control Plane", RFC 6192, DOI 10.17487/RFC6192, March 2011, <<https://www.rfc-editor.org/info/rfc6192>>.

[X.1252] "ITU-T X.1252 (04/2010)", n.d..

Appendix A. The Attic

The following terms are stashed for now and will be updated later:

Asset: Is a system resource, as defined in [RFC4949], that may be composed of other assets.

Examples of Assets include: Endpoints, Software, Guidance, or X.509 public key certificates. An asset is not necessarily owned by an organization.

Asset Management: The IT process by which assets are provisioned, updated, maintained and deprecated.

Asset Characterization: Asset characterization is the process of defining attributes that describe properties of an identified asset.

Asset Targeting: Asset targeting is the use of asset identification and categorization information to drive human-directed, automated decision making for data collection and analysis in support of endpoint posture assessment.

Client: An architectural component receiving services from another architectural component.

Endpoint Identification (TBD per list; was "Endpoint Discovery"):
The process by which an endpoint can be identified.

Authors' Addresses

Henk Birkholz
Fraunhofer SIT
Rheinstrasse 75
Darmstadt 64295
Germany

Email: henk.birkholz@sit.fraunhofer.de

Jarrett Lu
Oracle Corporation
4180 Network Circle
Santa Clara, CA 95054
USA

Email: jarrett.lu@oracle.com

John Strassner
Huawei Technologies
2330 Central Expressway
Santa Clara, CA 95138
USA

Email: john.sc.strassner@huawei.com

Nancy Cam-Winget
Cisco Systems
3550 Cisco Way
San Jose, CA 95134
USA

Email: ncamwing@cisco.com

Adam Montville
Center for Internet Security
31 Tech Valley Drive
East Greenbush, NY 12061
USA

Email: adam.w.montville@gmail.com

sacm
Internet-Draft
Intended status: Standards Track
Expires: January 26, 2020

C. Inacio
CMU
July 25, 2019

SACM Information Model
draft-inacio-sacm-infomodel-00

Abstract

This defines the information model for the Security Automation and Continuous Monitoring (SACM) standards. The working group faces a set of complex issues when trying to define an information model that complicates this effort:

- o There are many standards in the SACM space which are not interoperable
- o There exists an extremely large and diverse set of data types which are desirable to exchange
- o Many data types depend on the operating systems from which they are collected; making a universal typing harder
- o A goal of SACM is to cover a diverse set of system types

These complex needs create a information model which is difficult to unify within the environment. Instead, this information model design is focused on minimum needed functionality with the desire to include a type system design into the information model allowing for easy expandability. It is envisioned that this information model will serve the following purposes:

- o Enough well specified elements in order to exchange key data fields between systems
- o Sufficient typing system to expand key fields over time and use of a registry to standardize common expansions
- o Meta information such that complete information exchange using various other formats understood by all parties can be used as needed to exchange complete records on demand
- o Sufficient action verbs defined to allow orchestration between various systems to allow unified control of federated components

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 26, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Conventions and Terminology	3
2. Minimal Needed Information Elements	4
3. Information Element Metadata	4
3.1. Information Elements	4
3.1.1. IPv4 Address	4
3.1.2. IPv6 Address	5
3.1.3. Hostname	5
3.1.4. AssetID	6
3.1.5. MACAddress	6
3.1.6. Timestamp	6
3.1.7. Action	7
3.1.8. Action Parameters	7
3.1.9. AdditionalDataType	7

3.1.10. AdditionalData	8
3.1.11. Extra	8
4. Updates	9
5. IANA Considerations	9
6. Security Considerations	9
7. Normative References	9
Appendix A. Acknowledgements	9
Author's Address	10

1. Introduction

The set of elements which are desired to standarize are the subset of data elements used within the SACM standards and related standards. To this end, the core capability to reasonably identify a network end point and minimally describe an event along with enough information that two parties involved in the communication may determine a way forward for further information exchange. The minimal set of activity and endpoint identifiers will allow parties participating in SACM communications to effectively search their respective data stores for relevent and related information and respond to queries or accept events in kind.

This information model is intended to describe a minimal number of elements which enable this functionality, but also sufficiently describe the attributes which can define those elements. This combination of information intends to provide enough meta information about information elements to allow both in protocol definition of types in possible data models as well as clear construction of future standardized element definitions. Conversely, this information model is not attempting to define all possible information elements that need to be exchanged. Many information elements, especially those related to host monitoring, are heavily related to the operating system and related software for proper context - beyond the initial scope of this standard.

1.1. Conventions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119][RFC8174] when, and only when, they appear in all capitals, as shown here.

Additionally, the key words "*MIGHT*", "*COULD*", "*MAY WISH TO*", "*WOULD PROBABLY*", "*SHOULD CONSIDER*", and "*MUST (BUT WE KNOW YOU WON'T)*" in this document are to be interpreted as described in RFC 6919 [RFC6919].

2. Minimal Needed Information Elements

IP Address, hostname, time/date, SWID/CoSWID ID's, firmware versions, serial number, MAC address, certificate ID

3. Information Element Metadata

name, basic_data_type, octet_length, data_use_type (label, counter, gauge), description, std/vendor type, structure/composite

The following fields are defined in the set of metadata about each information element

name:

A descriptive but concise name to be used for human understanding

basic data type:

A fundamental data type supported by the this information model. The predefined types include unsigned integers, signed integers, octet array, string, IP addresses, MAC addresses

octet length:

The number of octets maximally used for this information

data use type:

This refines the basic data type expressing the usage of the value. For example, some integers represent mathematical values and may be added together (counts for example) while some things may be expressed as an integer, but are really a type of label (e.g. IP address)

description:

A longer textual description of this data type

registration domain:

The domain in which this information element is defined.

composite structure:

The definition of the composite structure of following elements, e.g. list, set, map

3.1. Information Elements

3.1.1. IPv4 Address

Field	Value
Name	IPv4
Basic data type	32-bit unsigned integer
Octet length	4
Data use type	Label
Description	An Internet Protocol version 4 address
Registration domain	standard
Composite structure	N/A
Comments	

3.1.2. IPv6 Address

Field	Value
Name	IPv6
Basic data type	octet array
Octet length	16
Data use type	Label
Description	An Internet Protocol version 6 address
Registration domain	standard
Composite structure	N/A
Comments	

3.1.3. Hostname

Field	Value
Name	Hostname
Basic data type	string
Octet length	up to 256
Data use type	Label
Description	Fully qualified domain name of endpoint system
Registration domain	standard
Composite structure	N/A
Comments	

3.1.4. AssettID

Field	Value
Name	AssettID
Basic data type	string
Octet length	up to 256
Data use type	Label
Description	AssettID of topic assett
Registration domain	standard
Composite structure	N/A
Comments	

3.1.5. MACAddress

Field	Value
Name	MACAddress
Basic data type	string
Octet length	6
Data use type	Label
Description	IEEE 802 Hardware Address
Registration domain	standard
Composite structure	N/A
Comments	

3.1.6. Timestamp

Field	Value
Name	timestamp
Basic data type	ISO time formatted string
Octet length	variable
Data use type	time/date
Description	time date string
Registration domain	standard
Composite structure	N/A
Comments	

3.1.7. Action

Field	Value
Name	Action
Basic data type	enumeration
Octet length	2
Data use type	label
Description	
Registration domain	standard
Composite structure	
Comments	RunAssessment, AssessmentResult, Subscribe, PubEvent,

3.1.8. Action Parameters

Field	Value
Name	Action Parameters
Basic data type	list
Octet length	variable
Data use type	variable
Description	parameters for the action command, defined per action command
Registration domain	standard
Composite structure	list
Comments	

3.1.9. AdditionalDataType

Field	Value
Name	AdditionalDataType
Basic data type	16-bit integer
Octet length	2
Data use type	label
Description	An enumeration of registered additional data types that can be contained in the AdditionalData field
Registration domain	standard
Composite structure	N/A
Comments	

3.1.10. AdditionalData

Field	Value
Name	AdditionalData
Basic data type	octet-array
Octet length	variable
Data use type	opaque
Description	This is an envelope to contain other standardized data exchange formats
Registration domain	standard
Composite structure	N/A
Comments	formats like OVAL or IF-MAP may be contained in here

3.1.11. Extra

[ed: remove before publication]

Field	Value
Name	
Basic data type	
Octet length	
Data use type	
Description	
Registration domain	standard
Composite structure	
Comments	

4. Updates

- o 25-July-2019 - initial document

5. IANA Considerations

This will create a IANA registry of elements, eventually. IANA language to be added

6. Security Considerations

To be completed.

7. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC6919] Barnes, R., Kent, S., and E. Rescorla, "Further Key Words for Use in RFCs to Indicate Requirement Levels", RFC 6919, DOI 10.17487/RFC6919, April 2013, <<https://www.rfc-editor.org/info/rfc6919>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

Appendix A. Acknowledgements

The contributions of the SACM working group have greatly impacted the thinking presented here. In particular, we wish to thank Bill Munyan, Adam Monteville, and Henk Birkholz.

Author's Address

Christopher Inacio
Carnegie Mellon University
4500 5th Ave.
Pittsburgh PA 15213
United States

Email: inacio@cert.org