SACM
Internet-Draft
Intended status: Standards Track
Expires: March 13, 2017

D. Waltermire, Ed.
NIST
K. Watson
DHS
C. Kahn
L. Lorenzin
Pulse Secure, LLC
M. Cokus
D. Haynes
The MITRE Corporation
H. Birkholz
Fraunhofer SIT
September 9, 2016

SACM Information Model
draft-ietf-sacm-information-model-07

Abstract

This document defines the Information Elements that are transported between SACM components and their interconnected relationships. The primary purpose of the Secure Automation and Continuous Monitoring (SACM) Information Model is to ensure the interoperability of corresponding SACM data models and addresses the use cases defined by SACM. The Information Elements and corresponding types are maintained as the IANA "SACM Information Elements" registry.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The SACM Information Model (IM) serves multiple purposes:

- o to ensure interoperability between SACM data models that are used as transport encodings,

- o to provide a standardized set of Information Elements - the SACM Vocabulary - to enable the exchange of content vital to automated security posture assessment, and

- o to enable secure information sharing in a scalable and extensible fashion in order to support the tasks conducted by SACM components.

A complete set of requirements imposed on the IM can be found in [I-D.ietf-sacm-requirements]. The SACM IM is intended to be used for standardized data exchange between SACM components (data in motion). Nevertheless, the Information Elements (IE) and their relationships defined in this document can be leveraged to create and align corresponding data models for data at rest.

The information model expresses, for example, target endpoint (TE) attributes, guidance, and evaluation results. The corresponding Information Elements are consumed and produced by SACM components as they carry out tasks.

The primary tasks that this information model supports (on data, control, and management plane) are:

- o TE Discovery

- o TE Characterization

- o TE Classification

- o Collection

- o Evaluation

- o Information Sharing

- o SACM Component Discovery

- o SACM Component Authentication

- o SACM Component Authorization

- SACM Component Registration

These tasks are defined in [I-D.ietf-sacm-terminology].

2. Conventions used in this document

2.1. Requirements Language

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

2.2. Information Element Examples

The notation used to define the SACM Information Elements (IEs) is based on a customized version of the IPFIX information model syntax [RFC7012] which is described in Figure 2. However, there are several examples presented throughout the document that use a simplified pseudo-code to illustrate the basic structure. It should be noted that while they include actual names of subjects and attributes as well as values, they are not intended to influence how corresponding SACM IEs should be defined in Section 7. The examples are provided for demonstration purposes only.

3. Information Elements

The IEs defined in this document comprise the building blocks by which all SACM content is composed. They are consumed and provided by SACM components on the data plane. Every Information Element has a unique label: its name. Every type of IE defined by the SACM IM is registered as a type at the IANA registry. The Integer Index of the IANA SMI number tables can be used by SACM data models.

3.1. Context of Information Elements

The IEs in this information model represent information related to assets in the following areas (based on the use cases described in [RFC7632]):

- Endpoint Management
- Software Inventory Management
- Hardware Inventory Management
- Configuration Management
- Vulnerability Management

3.2. Extensibility of Information Elements

A SACM data model based on this information model MAY include additional information elements that are not defined here. The labels of additional Information Elements included in different SACM data models MUST NOT conflict with the labels of the Information Elements defined by this information model, and the names of additional Information Elements MUST NOT conflict with each other or across multiple data models. In order to avoid naming conflicts, the labels of additional IEs SHOULD be prefixed to avoid collisions across extensions. The prefix MUST include an organizational identifier and therefore, for example, MAY be an IANA enterprise number, a (partial) name space URI, or an organization name abbreviation.

4. Structure of Information Elements

There are two basic types of IEs:

- o Attributes: an instance of an attribute type is the simplest IE structure comprised of a unique attribute name and an attribute value.

- o Subjects: a subject is a richer structure that has a unique subject name and one or more attributes or subjects. In essence, instances of a subject type are defined (and differentiated) by the attribute values and subjects associated with it.

```
hostname = "arbutus"

coordinates = (
latitude = N27.99619,
longitude = E86.92761
)
```

Figure 1: Example instance of an attribute and subject.

In general, every piece of information that enables security posture assessment or further enriches the quality of the assessment process can be associated with metadata. In the SACM IM, metadata is represented by specific subjects and is bundled with other attributes or subjects to provide additional information about them. The IM explicitly defines two kinds of metadata:

- o Metadata focusing on the data origin (the SACM component that provides the information to the SACM domain)

- o Metadata focusing on the data source (the target endpoint that is assessed)

Metadata can also include relationships that refer to other associated IEs (or SACM content in general) by using referencing labels that have to be included in the metadata of the associated IE.

Subjects can be nested and the SACM IM allows for circular or recursive nesting. The association of IEs via nesting results in a tree-like structure wherein subjects compose the root and intermediary nodes and attributes the leaves of the tree. This semantic structure does not impose a specific structure on SACM data models regarding data in motion or data repository schemata for data at rest.

The SACM IM provides two conceptual top-level subjects that are used to ensure a homogeneous structure for SACM content and its associated metadata: SACM statements and SACM content-elements. Every set of IEs that is provided by a SACM component must provide the information contained in these two subjects although it is up to the implementer whether or not the subjects are explicitly defined in a data model.

The notation the SACM IM is defined in is based on a modified version of the IP Information Flow Export (IPFIX) Information Model syntax described in Section 2.1 of [RFC7012]. The customized syntax used by the SACM IM is defined below in Figure 2.

```
elementId (required):    The numeric identifier of the
                         Information Element. It is used
                         for the compact identification
                         of an Information Element. If
                         this identifier is used without
                         an enterpriseID, then the
                         elementId must be unique, and
                         the description of allowed values
                         is administrated by IANA. The
                         value "TBD" may be used during
                         development of the information
                         model until an elementId is
                         assigned by IANA and filled
                         in at publication time.

enterpriseId (optional): Enterprises may wish to define
                         Information Elements without
                         registering them with IANA, for
                         example, for enterprise-internal
                         purposes.  For such Information
                         Elements, the elementId is
```

not sufficient when used outside the enterprise. If specifications of enterprise-specific Information Elements are made public and/or if enterprise-specific identifiers are used by SACM components outside the enterprise, then the enterprise-specific identifier MUST be made globally unique by combining it with an enterprise identifier. Valid values for the enterpriseId are defined by IANA as Structure of Management Information (SMI) network management private enterprise numbers.

name (required): A unique and meaningful name for the Information Element.

dataType (required): There are two kinds of datatypes: simple and structured. Attributes are defined using simple datatypes and subjects are defined using structured datatypes. The contents of the datatype field will be either a reference to one of the simple datatypes listed in Section 5.1, or the specification of structured datatype as defined in Section 5.2.

status (required): The status of the specification of the Information Element. Allowed values are "current" and "deprecated". All newly defined Information Elements have "current" status. The process for moving Information Elements to the "deprecated" status is TBD.

description (required): Describes the meaning of the Information Element, how it is derived, conditions for its use, etc.

structure (optional): A parsable property that provides details about the definition of

```
                                structured Information Elements as
                                described in Section 5.2.

       references (optional):  Identifies other RFCs or documents
                               outside the IETF which provide
                               additional information or context
                               about the Information Element.
```

Figure 2: Information Element Specification Template

4.1. Information Element Naming Convention

SACM Information Elements must adhere to the following naming conventions.

- o Names SHOULD be descriptive
- o Names MUST be unique within the SACM registry. Enterprise-specific names SHOULD be prefixed with a Private Enterprise Number [PEN].
- o Names MUST start with lowercase letters unless it begins with a Private Enterprise Number
- o Composed names MUST use capital letters for the first letter of each part

4.2. SACM Content Elements

Every piece of information that is provided by a SACM component is always associated with a set of metadata, for example, the timestamp at which this set of information was produced (e.g. by a collection task) or what target endpoint this set of information is about (e.g. the data-source or a target endpoint identifier, respectively). The subject that associates content IE with content-metadata IE is called a content-element. Content metadata can also include relationships that express associations with other content-elements.

```
content-element = (
  content-metadata = (
    collection-timestamp = 146193322,
    data-source = fb02e551-7101-4e68-8dec-1fde6bd10981
  ),
  hostname = "arbutus",
  coordinates = (
  latitude = N27.99619,
  longitude = E86.92761
  )
)
```

Figure 3: Example set of IEs associated with a timestamp and a target endpoint label.

4.3. SACM Statements

One or more SACM content elements are bundled in a SACM statement. In contrast to content-metadata, statement-metadata focuses on the providing SACM component instead of the target endpoint that the content is about. The only content-specific metadata included in the SACM statement is the content-type IE. Therefore, multiple content-elements that share the same statement metadata and are of the same content-type can be included in a single SACM statement. A SACM statement functions similar to an envelope or a header. Its purpose is to enable the tracking of the origin of data inside a SACM domain and more importantly to enable the mitigation of conflicting information that may originate from different SACM components. How a consuming SACM component actually deals with conflicting information is out-of-scope of the SACM IM. Semantically, the term statement implies that the SACM content provided by a SACM component might not be correct in every context, but rather is the result of a best-effort to produce correct information.

```
sacm-statement = (
  statement-metadata = (
    publish-timestamp = 1461934031,
    data-origin = 24e67957-3d31-4878-8892-da2b35e121c2,
    content-type = observation
  ),
  content-element = (
    content-metadata = (
      collection-timestamp = 146193322,
      data-source = fb02e551-7101-4e68-8dec-1fde6bd10981
    ),
    hostname = "arbutus"
  )
)
```

Figure 4: Example of a simple SACM statement including a single content-element.

```
sacm-statement = (
  statement-metadata = (
    publish-timestamp = 1461934031,
    data-origin = 24e67957-3d31-4878-8892-da2b35e121c2
    content-type = observation
  ),
  content-element = (
    content-metadata = (
      collection-timestamp = 146193322,
      data-source = fb02e551-7101-4e68-8dec-1fde6bd10981
    ),
    coordinates = (
      latitude = N27.99619,
      longitude = E86.92761
    )
  )
)

sacm-statement = (
  statement-metadata = (
    publish-timestamp = 1461934744,
    data-origin = e42885a1-0270-44e9-bb5c-865cf6bd4800,
    content-type = observation
  ),
  content-element = (
    content-metadata = (
      collection-timestamp = 146193821,
      te-label = fb02e551-7101-4e68-8dec-1fde6bd10981
    ),
    coordinates = (
      latitude = N16.67622,
      longitude = E141.55321
    )
  )
)
```

Figure 5: Example of conflicting information originating from different SACM components.

4.4. Relationships

An IE can be associated with another IE, e.g. a user-name attribute can be associated with a content-authorization subject. These references are expressed via the relationships subject, which can be included in a corresponding content-metadata subject. The relationships subject includes a list of one or more references. The SACM IM does not enforce a SACM domain to use unique identifiers as

references. Therefore, there are at least two ways to reference another

- o The value of a reference represents a specific content-label that is unique in a SACM domain (and has to be included in the corresponding content-element metadata in order to be referenced), or

- o The reference is a subject that includes an appropriate number of IEs in order to identify the referenced content-element by its actual content.

It is recommended to provide unique identifiers in a SACM domain and the SACM IM provides a corresponding naming-convention as a reference in Section 4.1. The alternative highlighted above summarizes a valid approach that does not require unique identifiers and is similar to the approach of referencing target endpoints via identifying attributes included in a characterization record.

```
content-element = (
  content-metadata = (
    collection-timestamp = 1461934031,
    te-label =
    fb02e551-7101-4e68-8dec-1fde6bd10981
    relationships = (
      associated-with-user-account =
      f3d70ef4-7e18-42af-a894-8955ba87c95d
    )
  ),
  hostname = "arbutus"
)

content-element = (
  content-metadata = (
    content-label = f3d70ef4-7e18-42af-a894-8955ba87c95d
  ),
  user-account = (
    username = romeo
    authentication = local
  )
)
```

Figure 6: Example instance of a content-element subject associated with another subject via its content metadata.

4.5. Event

Event subjects provide a structure to represent the change of IE values that was detected by a collection task at a specific point of time. It is mandatory to include the new values and the collection timestamp in an event subject and it is recommended to include the past values and a collection timestamp that were replaced by the new IE values. Every event can also be associated with a subject-specific event-timestamp and a lastseen-timestamp that might differ from the corresponding collection-timestamps. If these are omitted the collection-timestamp that is included in the content-metadata subject is used instead.

```
sacm-statement = (
  statement-metadata = (
    publish-timestamp = 1461934031,
    data-origin = 24e67957-3d31-4878-8892-da2b35e121c2,
    content-type = event
  ),
  event = (
    event-attributes = (
      event-name = "host-name change",
      content-element = (
        content-metadata = (
        collection-timestamp = 146193322,
        data-source =
          fb02e551-7101-4e68-8dec-1fde6bd10981,
          event-component = past-state
       ),
       hostname = "arbutus"
      ),
      content-element = (
        content-metadata = (
          collection-timestamp = 146195723,
          data-source =
          fb02e551-7101-4e68-8dec-1fde6bd10981,
          event-component = current-state
        ),
        hostname = "lilac"
      )
    )
  )
```

Figure 7: Example of a SACM statement containing an event.

4.6. Categories

Categories are special IEs that enable to refer to multiple types of IE via just one name. Therefore, they are similar to a type-choice. A prominent example of a category is network-address. Network-address is a category that every kind of network address is associated with, e.g. mac-address, ipv4-address, ipv6-address, or typed-network-address. If a subject includes network-address as one of its components, any of the category members are valid to be used in its place.

Another prominent example is EndpointIdentifier. Some IEs can be used to identify (and over time re-recognize) target endpoints - those are associated with the category endpoint-identifier.

5. Abstract Data Types

This section describes the set of valid abstract data types that can be used for the specification of the SACM Information Elements in Section 7. SACM currently supports two classes of datatypes that can be used to define Information Elements.

- o Simple: Datatypes that are atomic and are used to define the type of data represented by an attribute Information Element.

- o Structured: Datatypes that can be used to define the type of data represented by a subject Information Element.

Note that further abstract data types may be specified by future extensions of the SACM information model.

5.1. Simple Datatypes

5.1.1. IPFIX Datatypes

To facilitate the use of existing work, SACM supports the following abstract data types defined in Section 3 of [RFC7012].

- o unsigned8, unsigned16, unsigned32, unsigned64

- o signed8, signed16, signed32, signed64

- o float32, float64

- o boolean

- o macAddress

- octetArray
- string
- dateTimeSeconds, dateTimeMilliseconds, dateTimeMicroseconds, dateTimeNanoSeconds
- ipv4Address, ipv6Address

5.2. Structured Datatypes

5.2.1. List Datatypes

SACM defines the following abstract list data types that are used to represent the structured data associated with subjects.

- list: indicates that the Information Element order is not significant but MAY be preserved.
- orderedList: indicates that Information Element order is significant and MUST be preserved.

The notation for defining a SACM structured datatype is based on regular expressions, which are composed of the keywords "list" or "orderedList" and an Information Element expression. IE expressions use some of the regular expression syntax and operators, but the terms in the expression are the names of defined Information Elements instead of character classes. The syntax for defining list and orderedList datatypes is described below, using BNF:

```
<list-def> -> ("list"|"orderedList") "(" <ie-expression> ")"

<ie-expression> -> <ie-name> <cardinality>?
                   ( "," | "|") <ie-name> <cardinality>?)*

<cardinality> -> "*" | "+" | "?" |
                 ( "(" <non-neg-int> ("," <non-neg-int>)? ")" )
```

Figure 8: Syntax for Defining List Datatypes

As seen above, multiple occurences of an Information Element may be present in a structured datatype. The cardinality of an Information Element within a structured Information Element definition is defined by the following operators:

```
      * - zero or more occurrences

      + - one or more occurrences

      ? - zero or one occurrence

     (m,n) - between m and n occurrences
```

Figure 9: Specifying Cardinality for Structured Datatypes

The absence of a cardinality operator implies one mandatory occurrence of the Information Element.

Below is an example of a structured Information Element definition.

```
personInfo = list(firstName, middleNames?, lastName)
firstName = string
middleNames = orderedList(middleName+)
middleName = string
lastName = string
```

As an example, consider the name "John Ronald Reuel Tolkien". Below are instances of this name, structured according to the personInfo definition.

```
personInfo = (firstName="John", middleNames(middleName="Ronald",
              middleName="Reuel"), lastName="Tolkien")

personInfo = (middleNames(middleName="Ronald", middleName=" Reuel"),
              lastName="Tolkien", firstName="John")
```

The instance below is not legal with respect to the definition of personInfo because the order in middleNames is not preserved.

```
personInfo = (firstName="John", middleNames(middleName=" Reuel",
              middleName="Ronald"), lastName="Tolkien")
```

Figure 10: Example of Defining a Structured List Datatype

5.2.2. Enumeration Datatype

SACM defines the following abstract enumeration datatype that is used to represent the restriction of an attribute value to a set of values.

```
name, hex-value, description
<enumeration-def> -> -> <name> ";" <hex-value> ";" <description>
<name> -> [0-9a-zA-Z]+
<hex-value> -> 0x[0-9a-fA-F]+
<description> -> [0-9a-zA-Z\.\,]+
```

Figure 11: Syntax for Defining an Enumeration Datatype

Below is an example of a structured Information Element definition for an enumeration.

```
Red    ; 0x1  ; The color is red.
Orange ; 0x2  ; The color is orange.
Yellow ; 0x3  ; The color is yellow.
Green  ; 0x4  ; The color is green.
...
```

Figure 12: Example of Defining a Structured Enumeration Datatype

6. Information Model Assets

In order to represent the Information Elements related to the areas listed in Section 3.1, the information model defines the information needs (or metadata about those information needs) related to following types of assets which are defined in [I-D.ietf-sacm-terminology] (and included below for convenience) which are of interest to SACM. Specifically:

- Endpoint
- Software Component
- Hardware Component
- Identity
- Guidance
- Evaluation Results

The following figure shows the make up of an Endpoint asset which contains zero or more hardware components and zero or more software components each of which may have zero or more instances running an endpoint at any given time as well as zero or more identities that act on behalf of the endpoint when interfacing with other endpoints, tools, or services. An endpoint may also contain other endpoints in the case of a virtualized environment.

```
+---------+*______in>_______*+-----+
|Hardware |                  |!   !|
|Component|   +---------+    |!   !|
+---------+   |Software |in> |!   !|
              |Component|____|!   !|
              +---------+*  *|!   !|
                  1|         |!   !|
                  *|         |     |       +----------+
              +---------+    |End- |*_____*| Identity |
              |Software |in> |point| acts  +----------+
              |Instance |____|     | for>
              +---------+*  1|!   !|
                               |!   !|
                               |!   !|
                               |!   !|
                               |!   !|____
                               |!   !|0..1|
                               +-----+    |
                                  |*      |
                                  |_______|
                                     in>
```

Figure 13: Model of an Endpoint

6.1. Asset

As defined in [RFC4949], an asset is a system resource that is (a) required to be protected by an information system's security policy, (b) intended to be protected by a countermeasure, or (c) required for a system's mission.

In the scope of SACM, an asset can be composed of other assets. Examples of Assets include: Endpoints, Software, Guidance, or Identity. Furthermore, an asset is not necessarily owned by an organization.

6.2. Endpoint

From [RFC5209], an endpoint is any computing device that can be connected to a network. 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, 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).

6.3. Hardware Component

Hardware components are the distinguishable physical components that 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. mainboard is a hardware type that provides local busses as an interconnect). In general, a hardware component can be distinguished by its serial number.

Examples of a hardware components include: motherboards, network interfaces, graphics cards, hard drives, etc.

6.4. Software Component

A software package installed on an endpoint (including the operating system) as well as a unique serial number if present (e.g. a text editor associated with a unique license key).

It should be noted that this includes both benign and harmful software packages. Examples of benign software components include: applications, patches, operating system kernel, boot loader, firmware, code embedded on a webpage, etc. Examples of malicious software components include: malware, trojans, viruses, etc.

6.4.1. Software Instance

A running instance of the software component (e.g. 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).

6.5. Identity

Any mechanism that can be used to identify an asset during an authentication process. Examples include usernames, user and device certificates, etc. Note, that this is different than the identity of assets in the context of designation as described in Section 11.1.

6.6. Guidance

Guidance is input instructions to processes and tasks, such as collection or evaluation. Guidance influences the behavior of a SACM component and is considered content of the management plane. Guidance 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. A prominent example of guidance are target endpoint profiles,but guidance can have many forms, including:

Configuration, e.g. a SACM component's name, or a CMDB's IPv6 address.

Profiles, e.g. a set of expected states for network behavior associated with target endpoints employed by specific users.

Policies, e.g. an interval to refresh the registration of a SACM component, or a list of required capabilities for SACM components in a specific location.

6.6.1. Collection Guidance

A collector may need guidance to govern what it collects and when. Collection Guidance provides instructions for a Collector that specifies which endpoint attributes to collect, when to collect them, and how to collect them. Collection Guidance is composed of Target Endpoint Attribute Guidance, Frequency Guidance, and Method Guidance.

- Target Endpoint Attribute Guidance: Set of endpoint attributes that are supposed to be collected from a target endpoint. The definition of the set of endpoint attributes is typically based on an endpoint characterization record.

- Frequency Guidance: Specifies when endpoint attributes are to be collected.

- Method Guidance: Indicates how endpoint attributes are to be collected.

6.6.2. Evaluation Guidance

An evaluator typically needs guidance to govern what it considers to be a good or bad security posture. Evaluation Guidance provides instructions for an Evaluator that specifies which endpoint attributes to evaluate, the desired state of those endpoint attributes, and any special requirements that enable an Evaluator to determine if the endpoint attributes can be used in the evaluation (e.g. freshness of data, how it was collected, etc.). Evaluation Guidance is composed of Target Endpoint Attribute Guidance, Expected Endpoint Attribute Value Guidance, and Frequency Guidance.

- Target Endpoint Attribute Guidance: Set of target endpoint attributes that are supposed to be used in an evaluation as well as any requirements on the endpoint attributes. The definition of the set of endpoint attributes is typically based on an endpoint characterization record.

- Expected Endpoint Attribute Value Guidance: The expected values of the endpoint attributes described in the Target Endpoint Attribute Guidance.

- Frequency Guidance: Specifies when endpoint attributes are to be evaluated.

- Method Guidance: Indicates how endpoint attributes are to be collected.

6.6.3. Classification Guidance

A SACM Component carrying out the Target Endpoint Classification Task may need guidance on how to classify an endpoint. Specifically, how to associate endpoint classes with a specific target endpoint characterization record. Target Endpoint Classes function as guidance for collection, evaluation, remediation and security posture assessment in general. Classification Guidance is composed of Target Endpoint Attribute Guidance and Class Guidance.

- Target Endpoint Attribute Guidance: Set of target endpoint attributes that are supposed to be used to identify the endpoint characterization record.

- Class Guidance: A list of target endpoint classes that are to be associated with the identified target endpoint characterization record.

6.6.4. Storage Guidance

An SACM Component typically needs guidance to govern what information it should store and where. Storage Guidance provides instructions for a SACM Component that specifies which security automation information should be stored, for how long, and on which endpoint. Storage Guidance is composed of Target Endpoint Attribute Guidance, Expected Security Automation Information Guidance, and Retention Guidance.

- Target Endpoint Attribute Guidance: Set of target endpoint attributes that are supposed to be used to identify the endpoint where the security automation information is to be stored.

- Expected Security Automation Information Guidance: The security automation information that is expected to be stored (guidance, collected posture attributes, results, etc.).

- Retention Guidance: Specifies how long the security automation information should be stored.

6.6.5. Evaluation Results

Evaluation Results are the output of comparing the actual state of an endpoint against the expected state of an endpoint. In addition to the actual results of the comparison, Evaluation Results should include the Evaluation Guidance and actual target endpoint attributes values used to perform the evaluation.

7. Information Model Elements

This section defines the specific Information Elements and relationships that will be implemented by data models and transported between SACM Components.

7.1. accessPrivilegeType

```
elementId: TBD
name: accessPrivilegeType
dataType: string
status: current
description: A set of types that represent access
privileges (read, write, none, etc.).
```

7.2. accountName

```
elementId: TBD
name: accountName
dataType: string
status: current
description: A label that uniquely identifies an account
that can require some form of (user) authentication to
access.
```

7.3. administrativeDomainType

```
elementId: TBD
name: accessPrivilegeType
dataType: string
status: current
description: A label the is supposed to uniquely
identify an administrative domain.
```

7.4. addressAssociationType

```
elementId: TBD
name: accessPrivilegeType
dataType: string
status: current
description: A label the is supposed to uniquely
identify an administrative domain.
```

7.5. addressMaskValue

```
elementId: TBD
name: addressMaskValue
dataType: string
status: current
description: A value that expresses a generic address
subnetting bitmask.
```

7.6. addressType

```
elementId: TBD
name: addressType
dataType: string
status: current
description: A set of types that specifies the type
of address that is expressed in an address subject
(e.g. ethernet, modbus, zigbee).
```

7.7. addressValue

```
elementId: TBD
name: addressValue
dataType: string
status: current
description: A value that expresses a generic network
             address.
```

7.8. applicationComponent

```
elementId: TBD
name: applicationComponent
dataType: string
status: current
description: A label that references a "sub"-application
that is part of the application (e.g. an add-on, a
cipher-suite, a library).
```

7.9. applicationLabel

```
elementId: TBD
name: applicationLabel
dataType: string
status: current
description: A label that is supposed to uniquely
reference an application.
```

7.10. applicationType

```
elementId: TBD
name: applicationType
dataType: string
status: current
description: A set of types (FIXME maybe a finite set
is not realistic here - value not enumerator?) that
identifies the type of (user-space) application
(e.g. text-editor, policy-editor, service-client,
service-server, calender, rouge-like RPG).
```

7.11. applicationManufacturer

```
elementId: TBD
name: applicationManufacturer
dataType: string
status: current
description: The name of the vendor that created the
application.
```

7.12. authenticator

```
elementId: TBD
name: authenticator
dataType: string
status: current
description: A label that references a SACM component
that can authenticate target endpoints (can be used in
a target-endpoint subject to express that the target
endpoint was authenticated by that SACM component.
```

7.13. authenticationType

```
elementId: TBD
name: authenticationType
dataType: string
status: current
description: A set of types that expresses which type
of authentication was used to enable a network
interaction/connection.
```

7.14. birthdate

```
elementId: TBD
name: birthdate
dataType: string
status: current
description: A label for the registered day of birth
of a natural person (e.g. the date of birth of a person
as an ISO date string).
references: http://rs.tdwg.org/ontology/voc/Person#birthdate
```

7.15. bytesReceived

```
elementId: TBD
name: bytesReceived
dataType: string
status: current
description: A value that represents a number of octets
received on a network interface.
```

7.16. bytesSent

```
elementId: TBD
name: bytesSent
dataType: string
status: current
description: A value that represents the number of
octets received on a network interface.
```

7.17. bytesSent

```
elementId: TBD
name: bytesSent
dataType: string
status: current
description: A value that represents the number of
octets sent on a network interface.
```

7.18. certificate

```
elementId: TBD
name: certificate
dataType: string
status: current
description: A value that expresses a certificate that
can be collected from a target endpoint.
```

7.19. collectionTaskType

```
elementId: TBD
name: collectionTaskType
dataType: string
status: current
description: A set of types that defines how collected
SACM content was acquired (e.g. network-observation,
remote-acquisition, self-reported).
```

7.20. confidence

```
elementId: TBD
name: confidence
dataType: string
status: current
description: A representation of the subjective probability
that the assessed value is correct. If no confidence value
is given, it is assumed that the confidence is 1. Acceptable
values are between 0 and 1.
```

7.21. contentAction

```
elementId: TBD
name: contentAction
dataType: string
status: current
description: A set of types that express a type of
action (e.g. add, delete, update). It can be associated,
for instance, with an event subject or with a network
observation.
```

7.22. countryCode

```
elementId: TBD
name: countryCode
dataType: string
status: current
description: A set of types according to ISO 3166-1.
```

7.23. dataOrigin

```
elementId: TBD
name: dataOrigin
dataType: string
status: current
description: A label that uniquely identifies a SACM
component in and across SACM domains.
```

7.24. dataSource

```
elementId: TBD
name: dataSource
dataType: string
status: current
description: A label that is supposed to uniquely
identify the data source (e.g. a target endpoint or
sensor) that provided an initial endpoint attribute
record.
```

7.25. default-depth

```
elementId: TBD
name: default-depth
dataType: string
status: current
description: A value that expresses how often a circular
reference of subject is allowed to repeat, or how deep
a recursive nesting may occur, respectively.
```

7.26. discoverer

```
elementId: TBD
name: contentAction
dataType: string
status: current
description: A label that refers to the SACM component
that discovered a target endpoint (can be used in a
target-endpoint subject to express, for example, that
the target endpoint was authenticated by that SACM
component).
```

7.27. emailAddress

```
elementId: TBD
name: countryCode
dataType: string
status: current
description: A value that expresses an email-address.
```

7.28. eventType

```
elementId: TBD
name: eventType
dataType: string
status: current
description: a set of types that define the categories
of an event (e.g. access-level-change,
change-of-priviledge, change-of-authorization,
environmental-event, or provisioning-event).
```

7.29. eventThreshold

```
elementId: TBD
name: eventThreshold
dataType: string
status: current
description: if applicable, a value that can be
included in an event subject to indicate what numeric
threshold value was crossed to trigger that event.
```

7.30. eventThresholdName

```
elementId: TBD
name: eventThresholdName
dataType: string
status: current
description: If an event is created due to a crossed
threshold, the threshold might have a name associated
with it that can be expressed via this value.
```

7.31. eventTrigger

```
elementId: TBD
name: eventTrigger
dataType: string
status: current
description: This value is used to express more
complex trigger conditions that may cause the creation
of an event.
```

7.32. eventTrigger

```
elementId: TBD
name: eventTrigger
dataType: string
status: current
description: This value is used to express more
complex trigger conditions that may cause the creation
of an event.
```

7.33. firmwareId

```
elementId: TBD
name: firmwareId
dataType: string
status: current
description: A label that represents the BIOS or
firmware ID of a specific target endpoint.
```

7.34. hostName

```
elementId: TBD
name: hostName
dataType: string
status: current
description: A label typically associated with an
endpoint, but, not always intended to be unique given
scope.
```

7.35. interfaceLabel

```
elementId: TBD
name: interfaceLabel
dataType: string
status: current
description: A unique label that can be used to
             reference a network interface.
```

7.36. ipv6AddressSubnetMask

```
elementId: TBD
name: ipv6AddressSubnetMask
dataType: string
status: current
description: An IPv6 subnet bitmask.
```

7.37. ipv6AddressSubnetMaskCidrNotation

```
elementId: TBD
name: ipv6AddressSubnetMaskCidrNotation
dataType: string
status: current
description: An IPv6 subnet bitmask in CIDR notation.
```

7.38. ipv6AddressValue

```
elementId: TBD
name: ipv6AddressValue
dataType: ipv6Address
status: current
description: An IPv6 subnet bitmask in CIDR notation.
a network interface.
```

7.39. ipv4AddressSubnetMask

```
elementId: TBD
name: ipv4AddressSubnetMask
dataType: string
status: current
description: An IPv4 subnet bitmask.
```

7.40. ipv4AddressSubnetMaskCidrNotation

```
elementId: TBD
name: ipv4AddressSubnetMaskCidrNotation
dataType: string
status: current
description: An IPv4 subnet bitmask in CIDR notation.
```

7.41. ipv4AddressValue

```
elementId: TBD
name: ipv4AddressValue
dataType: ipv4Address
status: current
description: An IPv4 address value.
```

7.42. layer2InterfaceType

```
elementId: TBD
name: layer2InterfaceType
dataType: string
status: current
description: A set of types referenced by IANA ifType.
```

7.43. layer4PortAddress

```
elementId: TBD
name: layer4PortAddress
dataType: unsigned32
status: current
description: A layer 4 port address
typically associated with TCP and UDP
protocols.
```

7.44. layer4Protocol

```
elementId: TBD
name: layer4Protocol
dataType: string
status: current
description: A set of types that express a layer 4
protocol (e.g. UDP or TCP).
```

7.45. locationName

```
elementId: TBD
name: locationName
dataType: string
status: current
description: A value that represents a named region of
             physical space.
```

7.46. macAddressValue

```
elementId: TBD
name: macAddressValue
dataType: string
status: current
description: A value that expresses an Ethernet address.
```

7.47. methodLabel

```
elementId: TBD
name: methodLabel
dataType: string
status: current
description: A label that references a specific method
registered and used in a SACM domain (e.g. method to
match and re-identify target endpoints via identifying
attributes).
```

7.48. methodRepository

```
elementId: TBD
name: methodRepository
dataType: string
status: current
description: A label that references a SACM component
methods can be registered at and that can provide
guidance in the form of registered methods to other
SACM components.
```

7.49. networkAccessLevelType

```
elementId: TBD
name: networkAccessLevelType
dataType: string
status: current
description: A set of types that expresses categories
of network access-levels (e.g. block, quarantine, etc.).
```

7.50. networkId

```
elementId: TBD
name: networkId
dataType: string
status: current
description: Most networks such as AS, OSBF domains,
or VLANs can have an ID.
```

7.51. networkInterfaceName

```
elementId: TBD
name: networkInterfaceName
dataType: string
status: current
description: A label that uniquely identifies an interface
associated with a distinguishable endpoint.
```

7.52. networkLayer

```
elementId: TBD
name: networkLayer
dataType: string
status: current
description: A set of layers that expresses the specific
network layer an interface operates on.
```

7.53. networkName

```
elementId: TBD
name: networkName
dataType: string
status: current
description: A label that is associated with a network.
Some networks, for example, effetive layer2-broadcast-domains
are difficult to "grasp" and therefore quite difficult to name.
```

7.54. organizationId

```
elementId: TBD
name: organizationId
dataType: string
status: current
description: A label that uniquely identifies an
             organization via a PEN.
```

7.55. osComponent

```
elementId: TBD
name: osComponent
dataType: string
status: current
description: A label that references a "sub-component"
that is part of the operating system (e.g. a kernel
module, microcode, or ACPI table).
```

7.56. osLabel

```
elementId: TBD
name: osLabel
dataType: string
status: current
description: A label that references a specific version
of an operating system, including patches and hotfixes.
```

7.57. osName

```
elementId: TBD
name: osName
dataType: string
status: current
description: The name of an operating system.
```

7.58. osType

```
elementId: TBD
name: osType
dataType: string
status: current
description: A set of types that identifies the type
of an operating system (e.g. real-time,
security-enhanced, consumer, server).
```

7.59. osVersion

```
elementId: TBD
name: osVersion
dataType: string
status: current
description: A value that represents the version of
an operating-system.
```

7.60. patchId

```
elementId: TBD
name: patchId
dataType: string
status: current
description: A label the uniquely identifies a specific
             software patch.
```

7.61. patchName

```
elementId: TBD
name: osVersion
dataType: string
status: current
description: The vendor's name of a software patch.
```

7.62. personFirstName

```
elementId: TBD
name: patchId
dataType: string
status: current
description: The first name of a natural person.
```

7.63. personLastName

```
elementId: TBD
name: personLastName
dataType: string
status: current
description: The last name of a natural person.
```

7.64. personMiddleName

```
elementId: TBD
name: personMiddleName
dataType: string
status: current
description: The middle name of a natural person.
```

7.65. phoneNumber

```
elementId: TBD
name: phoneNumber
dataType: string
status: current
description: A label that expresses the U.S. national
phone number (e.g. pattern value="((\d{3}) )?\d{3}-\d{4}").
```

7.66. phoneNumberType

```
elementId: TBD
name: phoneNumberType
dataType: string
status: current
description: A set of types that express the type of
a phone number (e.g. DSN, Fax, Home, Mobile, Pager,
Secure, Unsecure, Work, Other).
```

7.67. privilegeName

```
elementId: TBD
name: privilegeName
dataType: string
status: current
description: The attribute name of the privilege
represented as an AVP.
```

7.68. privilegeValue

```
elementId: TBD
name: privilegeValue
dataType: string
status: current
description: The value content of the privilege
represented as an AVP.
```

7.69. protocol

```
elementId: TBD
name: protocol
dataType: string
status: current
description: A set of types that defines specific
protocols above layer 4 (e.g. http, https, dns, ipp,
or unknown).
```

7.70. publicKey

```
elementId: TBD
name: publicKey
dataType: string
status: current
description: The value of a public key (regardless of its
method of creation, crypto-system, or signature scheme)
that can be collected from a target endpoint.
```

7.71. relationshipContentElementGuid

```
elementId: TBD
name: relationshipContentElementGuid
dataType: string
status: current
description: A reference to a specific content element
used in a relationship subject.
```

7.72. relationshipStatementElementGuid

```
elementId: TBD
name: relationshipStatementElementGuid
dataType: string
status: current
description: A reference to a specific SACM statement
used in a relationship subject.
```

7.73. relationshipObjectLabel

```
elementId: TBD
name: relationshipObjectLabel
dataType: string
status: current
description: A reference to a specific label used in
content (e.g. a te-label or a user-id). This
reference is typically used if matching content
attribute can be done efficiantly and can also be
included in addition to a relationship-content-element-guid
reference.
```

7.74. relationshipType

```
elementId: TBD
name: relationshipType
dataType: string
status: current
description: A set of types that is in every instance
of a relationship subject to highlight what kind of
relationship exists between the subject the relationship
is included in (e.g. associated_with_user,
applies_to_session, seen_on_interface, associated_with_flow,
contains_virtual_device).
```

7.75. roleName

```
elementId: TBD
name: roleName
dataType: string
status: current
description: A label that references a collection of
privileges assigned to a specific entity (identity?
FIXME).
```

7.76. sessionStateType

```
elementId: TBD
name: sessionStateType
dataType: string
status: current
description: A set of types a discernible session (an
ongoing network interaction) can be in (e.g.
Authenticating, Authenticated, Postured, Started,
Disconnected).
```

7.77. statementGuid

```
elementId: TBD
name: statementGuid
dataType: string
status: current
description: A label that expresses a global unique
ID referencing a specific SACM statement that was
produced by a SACM component.
```

7.78. statementType

```
elementId: TBD
name: statementType
dataType: string
status: current
description: A set of types that define the type of
content that is included in a SACM statement (e.g.
Observation, DirectoryContent, Correlation, Assessment,
Guidance).
```

7.79. status

```
elementId: TBD
name: status
dataType: string
status: current
description: A set of types that defines possible
result values for a finding in general (e.g. true,
false, error, unknown, not applicable, not evaluated).
```

7.80. subAdministrativeDomain

```
elementId: TBD
name: subAdministrativeDomain
dataType: string
status: current
description: A label for related child domains an
administrative domain can be composed of (used in the
subject administrative-domain)
```

7.81. subInterfaceLabel

```
elementId: TBD
name: subInterfaceLabel
dataType: string
status: current
description: A unique label a sub network interface
(e.g. a tagged vlan on a trunk) can be referenced
with.
```

7.82. superAdministrativeDomain

```
elementId: TBD
name: superAdministrativeDomain
dataType: string
status: current
description: a label for related parent domains an
             administrative domain is part of (used
             in the subject s.administrative-domain).
```

7.83. superInterfaceLabel

```
elementId: TBD
name: superInterfaceLabel
dataType: string
status: current
description: a unique label a super network interface
             (e.g. a physical interface a tunnel
             interface terminates on) can be referenced
             with.
```

7.84. teAssessmentState

```
elementId: TBD
name: teAssessmentState
dataType: string
status: current
description: a set of types that defines the state of
             assessment of a target-endpoint (e.g.
             in-discovery, discovered, in-classification,
             classified, in-assessment, assessed).
```

7.85. teLabel

```
elementId: TBD
name: teLabel
dataType: string
status: current
description: an identifying label created from a set
             of identifying attributes used to reference
             a specific target endpoint.
```

7.86. teId

```
elementId: TBD
name: teId
dataType: string
status: current
description: an identifying label that is created
             randomly, is supposed to be unique, and
             used to reference a specific target
             endpoint.
```

7.87. timestampType

```
elementId: TBD
name: timestampType
dataType: string
status: current
description: a set of types that express what type of
             action or event happened at that point
             of time (e.g. discovered, classified,
             collected, published). Can be included in
             a generic s.timestamp subject.
```

7.88. unitsReceived

```
elementId: TBD
name: unitsReceived
dataType: string
status: current
description: a value that represents a number of units
             (e.g. frames, packets, cells or segments)
             received on a network interface.
```

7.89. unitsSent

```
elementId: TBD
name: unitsSent
dataType: string
status: current
description: a value that represents a number of units
             (e.g. frames, packets, cells or segments)
             sent on a network interface.
```

7.90. username

```
elementId: TBD
name: username
dataType: string
status: current
description: a part of the credentials required to
access an account that can be collected from a target
endpoint.
```

7.91. userDirectory

```
elementId: TBD
name: userDirectory
dataType: string
status: current
description: a label that identifies a specific type
of user-directory (e.g. ldap, active-directory,
local-user).
```

7.92. userId

```
elementId: TBD
name: userId
dataType: string
status: current
description: a label that references a specific user
known in a SACM domain.
```

7.93. webSite

```
elementId: TBD
name: webSite
dataType: string
status: current
description: a URI that references a web-site.
```

7.94. WGS84Longitude

```
elementId: TBD
name: WGS84Longitude
dataType: float
status: current
description: a label that represents WGS 84 rev 2004
longitude.
```

7.95. WGS84Latitude

```
elementId: TBD
name: WGS84Latitude
dataType: float
status: current
description: a label that represents WGS 84 rev 2004
latitude.
```

7.96. WGS84Altitude

```
               elementId: TBD
               name: WGS84Altitude
               dataType: float
               status: current
               description: a label that represents WGS 84 rev 2004
               altitude.
```

7.97. hardwareSerialNumber

```
elementId: TBD
name: hardwareSerialNumber
dataType: string
status: current
description: A globally unique identifier for a particular
             piece of hardware assigned by the vendor.
```

7.98. interfaceName

```
elementId: TBD
name: interfaceName
dataType: string
status: current
description: A short name uniquely describing an interface,
             eg "Eth1/0". See [RFC2863] for the definition
             of the ifName object.
```

7.99. interfaceIndex

```
elementId: TBD
name: interfaceIndex
dataType: unsigned32
status: current
description: The index of an interface installed on an endpoint.
             The value matches the value of managed object
             'ifIndex' as defined in [RFC2863]. Note that ifIndex
             values are not assigned statically to an interface
             and that the interfaces may be renumbered every time
             the device's management system is re-initialized,
             as specified in [RFC2863].
```

7.100. interfaceMacAddress

```
elementId: TBD
name: interfaceMacAddress
dataType: macAddress
status: current
description: The IEEE 802 MAC address associated with a network
             interface on an endpoint.
```

7.101. interfaceType

```
elementId: TBD
name: interfaceType
dataType: unsigned32
status: current
description: The type of a network interface. The value matches
             the value of managed object 'ifType' as defined in
             [IANA registry ianaiftype-mib].
```

7.102. interfaceFlags

```
elementId: TBD
name: interfaceFlags
dataType: unsigned16
status: current
description: This information element specifies the flags
             associated with a network interface. Possible
             values include:
structure: Up                  ; 0x1   ; Interface is up.
           Broadcast           ; 0x2   ; Broadcast address valid.
           Debug               ; 0x4   ; Turn on debugging.
           Loopback            ; 0x8   ; Is a loopback net.
           Point-to-point      ; 0x10  ; Interface is point-to-point link.
           No trailers         ; 0x20  ; Avoid use of trailers.
           Resources allocated ; 0x40  ; Resources allocated.
           No ARP              ; 0x80  ; No address resolution protocol.
           Receive all         ; 0x100 ; Receive all packets.
```

7.103. networkInterface

```
elementId: TBD
name: networkInterface
dataType: orderedList
status: current
description: Information about a network interface
             installed on an endpoint. The
             following high-level digram
             describes the structure of
             networkInterface information
             element.
structure: orderedList(interfaceName, interfaceIndex, macAddress,
                       ifType, flags)
```

7.104. softwareIdentifier

```
elementId: TBD
name: softwareIdentifier
dataType: string
status: current
description: A globally unique identifier for a particular
             software application.
```

7.105. softwareTitle

```
elementId: TBD
name: softwareTitle
dataType: string
status: current
description: The title of the software application.
```

7.106. softwareCreator

```
elementId: TBD
name: softwareCreator
dataType: string
status: current
description: The software developer (e.g., vendor or author).
```

7.107. simpleSoftwareVersion

```
elementId: TBD
name: simpleSoftwareVersion
dataType: string
status: current
description: The version string for a software application that
             conforms to the format of a list of hierarchical
             non-negative integers separated by a single character
             delimiter format.
```

7.108. rpmSoftwareVersion

```
elementId: TBD
name: rpmSoftwareVersion
dataType: string
status: current
description: The version string for a software application that
             conforms to the EPOCH:VERSION-RELEASE format.
```

7.109. ciscoTrainSoftwareVersion

```
elementId: TBD
name: ciscoTrainSoftwareVersion
dataType: string
status: current
description: The version string for a software application that
             conforms to the Cisco IOS Train string format.
```

7.110. softwareVersion

```
elementId: TBD
name: softwareVerison
dataType: list
status: current
description: The version of the software application. Software
             applications may be versioned using a number of
             schemas. The following high-level digram describes
             the structure of the softwareVersion information
             element.
structure: list(simpleSoftwareVersion | rpmSoftwareVersion |
                ciscoTrainSoftwareVersion)
```

7.111. lastUpdated

```
elementId: TBD
name: lastUpdated
dataType: dateTimeSeconds
status: current
description: The date and time when the software instance
             was last updated on the system (e.g., new
             version instlalled or patch applied)
```

7.112. softwareInstance

```
elementId: TBD
name: softwareInstance
dataType: orderedList
status: current
description: Information about an instance of software
             installed on an endpoint. The following
             high-level digram describes the structure of
             softwareInstance information element.
structure: orderedList(softwareIdentifier, title, creator,
                       softwareVersion, lastUpdated)
```

7.113. globallyUniqueIdentifier

```
elementId: TBD
name: globallyUniqueIdentifier
dataType: unsigned8
status: current
metadata: true
description: TODO.
```

7.114. dataOrigin

```
elementId: TBD
name: dataOrigin
dataType: string
status: current
metadata: true
description: The origin of the data.
```

7.115. dataSource

```
elementId: TBD
name: dataSource
dataType: string
status: current
metadata: true
description: The source of the data.
```

7.116. creationTimestamp

```
elementId: TBD
name: creationTimestamp
dataType: dateTimeSeconds
status: current
metadata: true
description: The date and time when the posture
             information was created by a SACM Component.
```

7.117. collectionTimestamp

```
elementId: TBD
name: collectionTimestamp
dataType: dateTimeSeconds
status: current
metadata: true
description: The date and time when the posture
             information was collected or observed by a SACM
             Component.
```

7.118. publicationTimestamp

```
elementId: TBD
name: publicationTimestamp
dataType: dateTimeSeconds
status: current
metadata: true
description: The date and time when the posture
             information was published.
```

7.119. relayTimestamp

```
elementId: TBD
name: relayTimestamp
dataType: dateTimeSeconds
status: current
metadata: true
description: The date and time when the posture
             information was relayed to another SACM Component.
```

7.120. storageTimestamp

```
elementId: TBD
name: storageTimestamp
dataType: dateTimeSeconds
status: current
metadata: true
description: The date and time when the posture
             information was stored in a Repository.
```

7.121. type

```
elementId: TBD
name: type
dataType: enumeration
status: current
metadata: true
description: The type of data model use to represent
             some set of endpoint information. The following
             table lists the set of data models supported by SACM.
structure: TBD
```

7.122. protocolIdentifier

```
elementId: TBD
name: protocolIdentifier
dataType: unsigned8
status: current
description: The value of the protocol number in the IP packet
             header. The protocol number identifies the IP packet
             payload type. Protocol numbers are defined in the
             IANA Protocol Numbers registry.

             In Internet Protocol version 4 (IPv4), this is
             carried in the Protocol field.  In Internet Protocol
             version 6 (IPv6), this is carried in the Next Header
             field in the last extension header of the packet.
```

7.123. sourceTransportPort

```
elementId: TBD
name: sourceTransportPort
dataType: unsigned16
status: current
description: The source port identifier in the transport header.
             For the transport protocols UDP, TCP, and SCTP, this
             is the source port number given in the respective
             header.  This field MAY also be used for future
             transport protocols that have 16-bit source port
             identifiers.
```

7.124. sourceIPv4PrefixLength

```
elementId: TBD
name: sourceIPv4PrefixLength
dataType: unsigned8
status: current
description: The number of contiguous bits that are relevant in
             the sourceIPv4Prefix Information Element.
```

7.125. ingressInterface

```
elementId: TBD
name: ingressInterface
dataType: unsigned32
status: current
description: The index of the IP interface where packets of this
             Flow are being received.  The value matches the
             value of managed object 'ifIndex' as defined in
             [RFC2863]. Note that ifIndex values are not assigned
             statically to an interface and that the interfaces
             may be renumbered every time the device's management
             system is re-initialized, as specified in [RFC2863].
```

7.126. destinationTransportPort

```
elementId: TBD
name: destinationTransportPort
dataType: unsigned16
status: current
description: The destination port identifier in the transport
             header. For the transport protocols UDP, TCP, and
             SCTP, this is the destination port number given in
             the respective header. This field MAY also be used
             for future transport protocols that have 16-bit
             destination port identifiers.
```

7.127. sourceIPv6PrefixLength

```
elementId: TBD
name: sourceIPv6PrefixLength
dataType: unsigned8
status: current
description: The number of contiguous bits that are relevant in
             the sourceIPv6Prefix Information Element.
```

7.128. sourceIPv4Prefix

```
elementId: TBD
name: sourceIPv4Prefix
dataType: ipv4Address
status: current
description: IPv4 source address prefix.
```

7.129. destinationIPv4Prefix

```
elementId: TBD
name: destinationIPv4Prefix
dataType: ipv4Address
status: current
description: IPv4 destination address prefix.
```

7.130. sourceMacAddress

```
elementId: TBD
name: sourceMacAddress
dataType: macAddress
status: current
description: The IEEE 802 source MAC address field.
```

7.131. ipVersion

```
elementId: TBD
name: ipVersion
dataType: unsigned8
status: current
description: The IP version field in the IP packet header.
```

7.132. interfaceDescription

```
elementId: TBD
name: interfaceDescription
dataType: string
status: current
description: The description of an interface, eg "FastEthernet
             1/0" or "ISP
connection".
```

7.133. applicationDescription

```
elementId: TBD
name: applicationDescription
dataType: string
status: current
description: Specifies the description of an application.
```

7.134. applicationId

```
elementId: TBD
name: applicationId
dataType: octetArray
status: current
description: Specifies an Application ID per [RFC6759].
```

7.135. applicationName

```
elementId: TBD
name: applicationName
dataType: string
status: current
description: Specifies the name of an application.
```

7.136. exporterIPv4Address

```
elementId: TBD
name: exporterIPv4Address
dataType: ipv4Address
status: current
description: The IPv4 address used by the Exporting Process.
             This is used by the Collector to identify the
             Exporter in cases where the identity of the Exporter
             may have been obscured by the use of a proxy.
```

7.137. exporterIPv6Address

```
elementId: TBD
name: exporterIPv6Address
dataType: ipv6Address
status: current
description: The IPv6 address used by the Exporting Process.
             This is used by the Collector to identify the
             Exporter in cases where the identity of the
             Exporter may have been obscured by the use of a
             proxy.
```

7.138. portId

```
elementId: TBD
name: portId
dataType: unsigned32
status: current
description: An identifier of a line port that is unique per
             IPFIX Device hosting an Observation Point.
             Typically, this Information Element is used for
             limiting the scope of other Information Elements.
```

7.139. templateId

```
elementId: TBD
name: templateId
dataType: unsigned16
status: current
description: An identifier of a Template that is locally unique
             within a combination of a Transport session and an
             Observation Domain.

             Template IDs 0-255 are reserved for Template Sets,
             Options Template Sets, and other reserved Sets yet
             to be created. Template IDs of Data Sets are
             numbered from 256 to 65535.

             Typically, this Information Element is used for
             limiting the scope of other Information Elements.
             Note that after a re-start of the Exporting Process
             Template identifiers may be re-assigned.
```

7.140. collectorIPv4Address

```
elementId: TBD
name: collectorIPv4Address
dataType: ipv4Address
status: current
description: An IPv4 address to which the Exporting Process sends
             Flow information.
```

7.141. collectorIPv6Address

```
elementId: TBD
name: collectorIPv6Address
dataType: ipv6Address
status: current
description: An IPv6 address to which the Exporting Process sends
             Flow information.
```

7.142. informationElementIndex

```
elementId: TBD
name: informationElementIndex
dataType: unsigned16
status: current
description: A zero-based index of an Information Element
             referenced by informationElementId within a Template
             referenced by templateId; used to disambiguate
             scope for templates containing multiple identical
             Information Elements.
```

7.143. informationElementId

```
elementId: TBD
name: informationElementId
dataType: unsigned16
status: current
description: This Information Element contains the ID of another
             Information Element.
```

7.144. informationElementDataType

```
elementId: TBD
name: informationElementDataType
dataType: unsigned8
status: current
description: A description of the abstract data type of an IPFIX
             information element.These are taken from the
             abstract data types defined in section 3.1 of the
             IPFIX Information Model [RFC5102]; see that section
             for more information on the types described in the
             informationElementDataType sub-registry.

             These types are registered in the IANA IPFIX
             Information Element Data Type subregistry.  This
             subregistry is intended to assign numbers for type
             names, not to provide a mechanism for adding data
             types to the IPFIX Protocol, and as such requires a
             Standards Action [RFC5226] to modify.
```

7.145. informationElementDescription

```
elementId: TBD
name: informationElementDescription
dataType: string
status: current
description: A UTF-8 [RFC3629] encoded Unicode string containing
             a human-readable description of an Information
             Element.  The content of the
             informationElementDescription MAY be annotated with
             one or more language tags [RFC4646], encoded
             in-line [RFC2482] within the UTF-8 string, in order
             to specify the language in which the description is
             written.  Description text in multiple languages MAY
             tag each section with its own language tag; in this
             case, the description information in each language
             SHOULD have equivalent meaning.  In the absence of
             any language tag, the "i-default" [RFC2277] language
             SHOULD be assumed.  See the Security Considerations
             section for notes on string handling for Information
             Element type records.
```

7.146. informationElementName

```
elementId: TBD
name: informationElementName
dataType: string
status: current
description: A UTF-8 [RFC3629] encoded Unicode string containing
             the name of an Information Element, intended as a
             simple identifier.  See the Security Considerations
             section for notes on string handling for Information
             Element type records.
```

7.147. informationElementRangeBegin

```
elementId: TBD
name: informationElementRangeBegin
dataType: unsigned64
status: current
description: Contains the inclusive low end of the range of
             acceptable values for an Information Element.
```

7.148. informationElementRangeEnd

```
elementId: TBD
name: informationElementRangeEnd
dataType: unsigned64
status: current
description: Contains the inclusive high end of the range of
             acceptable values for an Information Element.
```

7.149. informationElementSemantics

```
elementId: TBD
name: informationElementSemantics
dataType: unsigned8
status: current
description: A description of the semantics of an IPFIX
             Information Element.  These are taken from the data
             type semantics defined in section 3.2 of the IPFIX
             Information Model [RFC5102]; see that section for
             more information on the types defined in the
             informationElementSemantics sub-registry.  This
             field may take the values in Table ; the special
             value 0x00 (default) is used to note that no
             semantics apply to the field; it cannot be
             manipulated by a Collecting Process or File Reader
             that does not understand it a priori.

             These semantics are registered in the IANA IPFIX
             Information Element Semantics subregistry.  This
             subregistry is intended to assign numbers for
             semantics names, not to provide a mechanism for
             adding semantics to the IPFIX Protocol, and as such
             requires a Standards Action [RFC5226] to modify.
```

7.150. informationElementUnits

```
elementId: TBD
name: informationElementUnits
dataType: unsigned16
status: current
description: A description of the units of an IPFIX Information
             Element.  These correspond to the units implicitly
             defined in the Information Element definitions in
             section 5 of the IPFIX Information Model [RFC5102];
             see that section for more information on the types
             described in the informationElementsUnits
             sub-registry. This field may take the values in
             Table 3 below; the special value 0x00 (none) is
             used to note that the field is unitless.

             These types are registered in the IANA IPFIX
             Information Element Units subregistry; new types
             may be added on a First Come First Served [RFC5226]
             basis.
```

7.151. userName

```
elementId: TBD
name: userName
dataType: string
status: current
description: User name associated with the flow.
```

7.152. applicationCategoryName

```
elementId: TBD
name: applicationCategoryName
dataType: string
status: current
description: An attribute that provides a first level
             categorization for each Application ID.
```

7.153. mibObjectValueInteger

```
elementId: TBD
name: mibObjectValueInteger
dataType: signed64
status: current
description: An IPFIX Information Element which denotes that the
             integer value of a MIB object will be exported.
             The MIB Object Identifier ("mibObjectIdentifier")
             for this field MUST be exported in a MIB Field
             Option or via another means.  This Information
             Element is used for MIB objects with the Base
             Syntax of Integer32 and INTEGER with IPFIX Reduced
             Size Encoding used as required. The value is
             encoded as per the standard IPFIX Abstract Data Type
             of signed64.
```

7.154. mibObjectValueOctetString

```
elementId: TBD
name: mibObjectValueOctetString
dataType: octetArray
status: current
description: An IPFIX Information Element which denotes that an
             Octet String or Opaque value of a MIB object will
             be exported. The MIB Object Identifier
             ("mibObjectIdentifier") for this field MUST be
             exported in a MIB Field Option or via another means.
             This Information Element is used for MIB objects
             with the Base Syntax of OCTET STRING and Opaque. The
             value is encoded as per the standard IPFIX Abstract
             Data Type of octetArray.
```

7.155. mibObjectValueOID

```
elementId: TBD
name: mibObjectValueOID
dataType: octetArray
status: current
description: An IPFIX Information Element which denotes that an
             Object Identifier or OID value of a MIB object will
             be exported. The MIB Object Identifier
             ("mibObjectIdentifier") for this field MUST be
             exported in a MIB Field Option or via another means.
             This Information Element is used for MIB objects
             with the Base Syntax of OBJECT IDENTIFIER.  Note -
             In this case the "mibObjectIdentifier" will define
             which MIB object is being exported while the value
             contained in this Information Element will be an
             OID as a value.  The mibObjectValueOID Information
             Element is encoded as ASN.1/BER [BER] in an
             octetArray.
```

7.156. mibObjectValueBits

```
elementId: TBD
name: mibObjectValueBits
dataType: octetArray
status: current
description: An IPFIX Information Element which denotes that a
             set of Enumerated flags or bits from a MIB object
             will be exported. The MIB Object Identifier
             ("mibObjectIdentifier") for this field MUST be
             exported in a MIB Field Option or via another means.
             This Information Element is used for MIB objects
             with the Base Syntax of BITS.  The flags or bits are
             encoded as per the standard IPFIX Abstract Data Type
             of octetArray, with sufficient length to accommodate
             the required number of bits.  If the number of bits
             is not an integer multiple of octets then the most
             significant bits at end of the octetArray MUST be
             set to zero.
```

7.157. mibObjectValueIPAddress

```
elementId: TBD
name: mibObjectValueIPAddress
dataType: ipv4Address
status: current
description: An IPFIX Information Element which denotes that the
             IPv4 Address of a MIB object will be exported.  The
             MIB Object Identifier ("mibObjectIdentifier") for
             this field MUST be exported in a MIB Field Option
             or via another means.  This Information Element is
             used for MIB objects with the Base Syntax of
             IPaddress. The value is encoded as per the standard
             IPFIX Abstract Data Type of ipv4Address.
```

7.158. mibObjectValueCounter

```
elementId: TBD
name: mibObjectValueCounter
dataType: unsigned64
status: current
description: An IPFIX Information Element which denotes that the
             counter value of a MIB object will be exported.
             The MIB Object Identifier ("mibObjectIdentifier")
             for this field MUST be exported in a MIB Field
             Option or via another means.  This Information
             Element is used for MIB objects with the Base
             Syntax of Counter32 or Counter64 with IPFIX Reduced
             Size Encoding used as required. The value is encoded
             as per the standard IPFIX Abstract Data Type
             of unsigned64.
```

7.159. mibObjectValueGauge

```
elementId: TBD
name: mibObjectValueGauge
dataType: unsigned32
status: current
description: An IPFIX Information Element which denotes that the
             Gauge value of a MIB object will be exported.  The
             MIB Object Identifier ("mibObjectIdentifier") for
             this field MUST be exported in a MIB Field Option
             or via another means.  This Information Element is
             used for MIB objects with the Base Syntax of Gauge32.
             The value is encoded as per the standard IPFIX
             Abstract Data Type of unsigned64.  This value will
             represent a non-negative integer, which may increase
             or decrease, but shall never exceed a maximum
             value, nor fall below a minimum value.
```

7.160. mibObjectValueTimeTicks

```
elementId: TBD
name: mibObjectValueTimeTicks
dataType: unsigned32
status: current
description: An IPFIX Information Element which denotes that the
             TimeTicks value of a MIB object will be exported.
             The MIB Object Identifier ("mibObjectIdentifier")
             for this field MUST be exported in a MIB Field
             Option or via another means.  This Information
             Element is used for MIB objects with the Base
             Syntax of TimeTicks. The value is encoded as per
             the standard IPFIX Abstract Data Type of unsigned32.
```

7.161. mibObjectValueUnsigned

```
elementId: TBD
name: mibObjectValueUnsigned
dataType: unsigned64
status: current
description: An IPFIX Information Element which denotes that an
             unsigned integer value of a MIB object will be
             exported.  The MIB Object Identifier
             ("mibObjectIdentifier") for this field MUST be
             exported in a MIB Field Option or via another means.
             This Information Element is used for MIB objects
             with the Base Syntax of unsigned64 with IPFIX
             Reduced Size Encoding used as required. The value is
             encoded as per the standard IPFIX Abstract Data Type
             of unsigned64.
```

7.162. mibObjectValueTable

```
elementId: TBD
name: mibObjectValueTable
dataType: orderedList
status: current
description: An IPFIX Information Element which denotes that a
             complete or partial conceptual table will be
             exported.  The MIB Object Identifier
             ("mibObjectIdentifier") for this field MUST be
             exported in a MIB Field Option or via another means.
             This Information Element is used for MIB objects
             with a SYNTAX of SEQUENCE.  This is encoded as a
             subTemplateList of mibObjectValue Information
             Elements.  The template specified in the
             subTemplateList MUST be an Options Template and
             MUST include all the Objects listed in the INDEX
             clause as Scope Fields.
structure:   orderedList(mibObjectValueRow+)
```

7.163. mibObjectValueRow

```
elementId: TBD
name: mibObjectValueRow
dataType: orderedList
status: current
description: An IPFIX Information Element which denotes that a
             single row of a conceptual table will be exported.
             The MIB Object Identifier ("mibObjectIdentifier")
             for this field MUST be exported in a MIB Field
             Option or via another means.  This Information
             Element is used for MIB objects with a SYNTAX of
             SEQUENCE.  This is encoded as a subTemplateList of
             mibObjectValue Information Elements.  The
             subTemplateList exported MUST contain exactly one
             row (i.e., one instance of the subtemplate).  The
             template specified in the subTemplateList MUST be
             an Options Template and MUST include all the
             Objects listed in the INDEX clause as Scope Fields.
structure:   orderedList(mibObjectValue+)
```

7.164. mibObjectIdentifier

```
elementId: TBD
name: mibObjectIdentifier
dataType: octetArray
status: current
description: An IPFIX Information Element which denotes that a
             MIB Object Identifier (MIB OID) is exported in the
             (Options) Template Record.  The mibObjectIdentifier
             Information Element contains the OID assigned to
             the MIB Object Type Definition encoded as
             ASN.1/BER [BER].
```

7.165. mibSubIdentifier

```
elementId: TBD
name: mibSubIdentifier
dataType: unsigned32
status: current
description: A non-negative sub-identifier of an Object
             Identifier (OID).
```

7.166. mibIndexIndicator

```
elementId: TBD
name: mibIndexIndicator
dataType: unsigned64
status: current
description: This set of bit fields is used for marking the
             Information Elements of a Data Record that serve as
             INDEX MIB objects for an indexed Columnar MIB
             object.  Each bit represents an Information Element
             in the Data Record with the n-th bit representing
             the n-th Information Element.  A bit set to value 1
             indicates that the corresponding Information Element
             is an index of the Columnar Object represented by
             the mibFieldValue.  A bit set to value 0 indicates
             that this is not the case.

             If the Data Record contains more than 64
             Information Elements, the corresponding Template
             SHOULD be designed such that all INDEX
             Fields are among the first 64 Information Elements,
             because the mibIndexIndicator only contains 64 bits.
             If the Data Record contains less than 64
             Information Elements, then the extra bits in the
             mibIndexIndicator for which no corresponding
             Information Element exists MUST have the value 0,
             and must be disregarded by the Collector.  This
             Information Element may be exported with
             IPFIX Reduced Size Encoding.
```

7.167. mibCaptureTimeSemantics

```
elementId: TBD
name: mibCaptureTimeSemantics
dataType: unsigned8
status: current
description: Indicates when in the lifetime of the flow the MIB
             value was retrieved from the MIB for a
             mibObjectIdentifier.  This is used to indicate if
             the value exported was collected from the MIB
             closer to flow creation or flow export time and
             will refer to the Timestamp fields included in the
             same record.  This field SHOULD be used when
             exporting a mibObjectValue that specifies counters
             or statistics.

             If the MIB value was sampled by SNMP prior to the
             IPFIX Metering Process or Exporting Process
             retrieving the value (i.e., the data is already
             stale) and it's important to know the exact sampling
             time, then an additional observationTime* element
             should be paired with the OID using structured data.
             Similarly, if different mibCaptureTimeSemantics
             apply to different mibObject elements within the
             Data Record, then individual mibCaptureTimeSemantics
             should be paired with each OID using structured data.

             Values:
             0.  undefined
             1.  begin - The value for the MIB object is captured
             from the MIB when the Flow is first observed
             2.  end - The value for the MIB object is captured
             from the MIB when the Flow ends
             3.  export - The value for the MIB object is
             captured from the MIB at export time
             4.  average - The value for the MIB object is an
             average of multiple captures from the MIB over the
             observed life of the Flow
```

7.168. mibContextEngineID

```
elementId: TBD
name: mibContextEngineID
dataType: octetArray
status: current
description: A mibContextEngineID that specifies the SNMP engine
             ID for a MIB field being exported over IPFIX.
             Definition as per [RFC3411] section 3.3.
```

7.169. mibContextName

```
elementId: TBD
name: mibContextName
dataType: string
status: current
description: This Information Element denotes that a MIB Context
             Name is specified for a MIB field being exported
             over IPFIX. Reference [RFC3411] section 3.3.
```

7.170. mibObjectName

```
elementId: TBD
name: mibObjectName
dataType: string
status: current
description: The name (called a descriptor in [RFC2578]
             of an object type definition.
```

7.171. mibObjectDescription

```
elementId: TBD
name: mibObjectDescription
dataType: string
status: current
description: The value of the DESCRIPTION clause of an MIB object
             type definition.
```

7.172. mibObjectSyntax

```
elementId: TBD
name: mibObjectSyntax
dataType: string
status: current
description: The value of the SYNTAX clause of an MIB object type
             definition, which may include a Textual Convention
             or Subtyping. See [RFC2578].
```

7.173. mibModuleName

```
elementId: TBD
name: mibModuleName
dataType: string
status: current
description: The textual name of the MIB module that defines a MIB
             Object.
```

7.174. interface

```
elementId: TBD
name: interface
dataType: list
structure: list (InterfaceName, hwAddress, inetAddr, netmask)
status: current
description: Represents an interface and its configuration
options.
```

7.175. interfaceName

```
elementId: TBD
name: interfaceName
dataType: string
status: current
description: The interface
      name.
```

7.176. iflisteners

```
elementId: TBD
name: iflisteners
dataType: list
structure: list (interfaceName, physicalProtocol, hwAddress,
      programName, pid, userId)
status: current
description: Stores the results of checking for applications that
are bound to an ethernet interface on the system.
```

7.177. physicalProtocol

```
elementId: TBD
name: physicalProtocol
dataType: enumeration
structure:
ETH_P_LOOP ; 0x1 ; Ethernet loopback packet.
ETH_P_PUP ; 0x2 ; Xerox PUP packet.
ETH_P_PUPAT ; 0x3 ; Xerox PUP Address Transport packet.
ETH_P_IP ; 0x4 ; Internet protocol packet.
ETH_P_X25 ; 0x5 ; CCITT X.25 packet.
ETH_P_ARP ; 0x6 ; Address resolution packet.
ETH_P_BPQ ; 0x7 ; G8BPQ AX.25 ethernet packet.
ETH_P_IEEEPUP ; 0x8 ; Xerox IEEE802.3 PUP packet.
ETH_P_IEEEPUPAT ; 0x9 ; Xerox IEEE802.3 PUP address transport
                        packet.
ETH_P_DEC ; 0xA ; DEC assigned protocol.
ETH_P_DNA_DL ; 0xB ; DEC DNA Dump/Load.
```

```
ETH_P_DNA_RC ; 0xC ; DEC DNA Remote Console.
ETH_P_DNA_RT ; 0xD ; DEC DNA Routing.
ETH_P_LAT ; 0xE ; DEC LAT.
ETH_P_DIAG ; 0xF ; DEC Diagnostics.
ETH_P_CUST ; 0x10 ; DEC Customer use.
ETH_P_SCA ; 0x11 ; DEC Systems Comms Arch.
ETH_P_RARP ; 0x12 ; Reverse address resolution packet.
ETH_P_ATALK ; 0x13 ; Appletalk DDP.
ETH_P_AARP ; 0x14 ; Appletalk AARP.
ETH_P_8021Q ; 0x15 ; 802.1Q VLAN Extended Header.
ETH_P_IPX ; 0x16 ; IPX over DIX.
ETH_P_IPV6 ; 0x17 ; IPv6 over bluebook.
ETH_P_SLOW ; 0x18 ; Slow Protocol. See 802.3ad 43B.
ETH_P_WCCP ; 0x19 ; Web-cache coordination protocol.
ETH_P_PPP_DISC ; 0x1A ; PPPoE discovery messages.
ETH_P_PPP_SES ; 0x1B ; PPPoE session messages.
ETH_P_MPLS_UC ; 0x1C ; MPLS Unicast traffic.
ETH_P_MPLS_MC ; 0x1D ; MPLS Multicast traffic.
ETH_P_ATMMPOA ; 0x1E ; MultiProtocol Over ATM.
ETH_P_ATMFATE ; 0x1F ; Frame-based ATM Transport over Ethernet.
ETH_P_AOE ; 0x20 ; ATA over Ethernet.
ETH_P_TIPC ; 0x21 ; TIPC.
ETH_P_802_3 ; 0x22 ; Dummy type for 802.3 frames.
ETH_P_AX25 ; 0x23 ; Dummy protocol id for AX.25.
ETH_P_ALL ; 0x24 ; Every packet.
ETH_P_802_2 ; 0x25 ; 802.2 frames.
ETH_P_SNAP ; 0x26 ; Internal only.
ETH_P_DDCMP ; 0x27 ; DEC DDCMP: Internal only
ETH_P_WAN_PPP ; 0x28 ; Dummy type for WAN PPP frames.
ETH_P_PPP_MP ; 0x29 ; Dummy type for PPP MP frames.
ETH_P_PPPTALK ; 0x2A ; Dummy type for Atalk over PPP.
ETH_P_LOCALTALK ; 0x2B ; Localtalk pseudo type.
ETH_P_TR_802_2 ; 0x2C ; 802.2 frames.
ETH_P_MOBITEX ; 0x2D ; Mobitex.
ETH_P_CONTROL ; 0x2E ; Card specific control frames.
ETH_P_IRDA ; 0x2F ; Linux-IrDA.
ETH_P_ECONET ; 0x30 ; Acorn Econet.
ETH_P_HDLC ; 0x31 ; HDLC frames.
ETH_P_ARCNET ; 0x32 ; 1A for ArcNet.
             ; 0x33 ; The empty string value is permitted here
             to allow for detailed error reporting.
status: current
description: The physical layer protocol used by the AF_PACKET
socket.
```

7.178. hwAddress

```
elementId: TBD
name: hwAddress
dataType: string
status: current
description: The hardware address associated
      with the interface.
```

7.179. programName

```
elementId: TBD
name: programName
dataType: string
status: current
description: The name of the communicating
      program.
```

7.180. userId

```
elementId: TBD
name: userId
dataType: integer
status: current
description: The numeric user id.
```

7.181. inetlisteningserver

```
elementId: TBD
name: inetlisteningserver
dataType: list
structure: list (transportProtocol, localAddress,
      localPort, localFullAddress, programName, foreignAddress,
      foreignPort, foreignFullAddress, pid, userId)
status:
      current
description: Stores the results of checking for network servers
currently active on a system. It holds information pertaining to
a specific protocol-address-port combination.
```

7.182. transportProtocol

```
elementId: TBD
name: transportProtocol
dataType: string
status: current
description: The transport-layer
      protocol (tcp or udp).
```

7.183. localAddress

```
elementId: TBD
name: localAddress
dataType: ipAddress
status: current
description: This is the IP address being listened to. Note that
the IP address can be IPv4 or IPv6.
```

7.184. localPort

```
elementId: TBD
name: localPort
dataType: integer
status: current
description: This is the TCP or UDP port
      being listened to.
```

7.185. localFullAddress

```
elementId: TBD
name: localFullAddress
dataType: string
status: current
description: The IP address and network port on which the program
listens, including the local address and the local port. Note
that the IP address can be IPv4 or IPv6.
```

7.186. foreignAddress

```
elementId: TBD
name: foreignAddress
dataType: ipAddresss
status: current
description: The IP address with which the program is
communicating, or with which it will communicate. Note that the
IP address can be IPv4 or IPv6.
```

7.187. foreignFullAddress

```
elementId: TBD
name: foreignFullAddress
dataType: ipAddresss
status: current
description: The IP address and network port to which the program
is communicating or will accept communications from, including
the foreign address and foreign port. Note that the IP address
can be IPv4 or IPv6.
```

7.188. selinuxboolean

```
elementId: TBD
name: selinuxboolean
dataType: list
structure: list (selinuxName, currentStatus,
      pendingStatus)
status: current
description: Describes the current and pending status of a
SELinux boolean.
```

7.189. selinuxName

```
elementId: TBD
name: selinuxName
dataType: string
status: current
description: The name of the SELinux
      boolean.
```

7.190. currentStatus

```
elementId: TBD
name: currentStatus
dataType: boolean
status: current
description: Indicates current state of
      the specified SELinux boolean.
```

7.191. pendingStatus

```
elementId: TBD
name: pendingStatus
dataType: boolean
status: current
description: Indicates the pending
      state of the specified SELinux boolean.
```

7.192. selinuxsecuritycontext

```
elementId: TBD
name: selinuxsecuritycontext
dataType: list
structure: list (filepath, path, filename, pid,
      username, role, domainType, lowSensitivity, lowCategory,
      highSensitivity, highCategory, rawlowSensitivity,
      rawlowCategory, rawhighSensitivity, rawhighCategory)
status: current
description: Describes the SELinux security
      context of a file or process on the local system.
```

7.193. filepath

```
elementId: TBD
name: filepath
dataType: string
status: current
description: Specifies the absolute path for a file on the
machine. A directory cannot be specified as a filepath.
```

7.194. path

```
elementId: TBD
name: path
dataType: string
status: current
description: Specifies the directory component of
      the absolute path to a file on the machine.
```

7.195. filename

```
elementId: TBD
name: filename
dataType: string
status: current
description: The name of the file.
```

7.196. pid

```
elementId: TBD
name: pid
dataType: integer
status: current
description: The process ID of the
      process.
```

7.197. role

```
elementId: TBD
name: role
dataType: string
status: current
description: Specifies the types that a process
      may transition to (domain transitions).
```

7.198. domainType

```
elementId: TBD
name: domainType
dataType: string
status: current
description: Specifies the domain in which the file is accessible
or the domain in which a process executes.
```

7.199. lowSensitivity

```
elementId: TBD
name: lowSensitivity
dataType: string
status: current
description: Specifies the current sensitivity of a file or
process.
```

7.200. lowCategory

```
elementId: TBD
name: lowCategory
dataType: string
status: current
description: Specifies the set of
      categories associated with the low sensitivity.
```

7.201. highSensitivity

```
elementId: TBD
name: highSensitivity
dataType: string
status: current
description: Specifies the maximum
      range for a file or the clearance for a process.
```

7.202. highCategory

```
elementId: TBD
name: highCategory
dataType: string
status: current
description: Specifies the set of
      categories associated with the high sensitivity.
```

7.203. rawlowSensitivity

```
elementId: TBD
name: rawlowSensitivity
dataType: string
status: current
description: Specifies the current sensitivity of a file or
process but in its raw context.
```

7.204. rawlowCategory

```
elementId: TBD
name: rawlowCategory
dataType: string
status: current
description: Specifies the set of categories associated with the
low sensitivity but in its raw context.
```

7.205. rawhighSensitivity

```
elementId: TBD
name: rawhighSensitivity
dataType: string
status: current
description: Specifies the maximum range for a file or the
clearance for a process but in its raw context.
```

7.206. rawhighCategory

```
elementId: TBD
name: rawhighCategory
dataType: string
status: current
description: Specifies the set of categories associated with the
high sensitivity but in its raw context.
```

7.207. systemdunitdependency

```
elementId: TBD
name: systemdunitdependency
dataType: list
structure: list (unit, dependency)
status: current

description: Stores the dependencies of the systemd
unit.
```

7.208. unit

```
elementId: TBD
name: unit
dataType: string
status: current
description: Refers to the full systemd unit name, which has a
form of "$name.$type". For example "cupsd.service". This name is
usually also the filename of the unit configuration file.
```

7.209. dependency

```
elementId: TBD
name: dependency
dataType: string
status: current
description: Refers to the name of a unit that was confirmed to
be a dependency of the given unit.
```

7.210. systemdunitproperty

```
elementId: TBD
name: systemdunitproperty
dataType: list
structure: list (unit, property, systemdunitValue)

status: current
description: Stores the properties and values of a systemd unit.
```

7.211. property

```
elementId: TBD
name: property
dataType: string
status: current
description: The property associated with a
      systemd unit.
```

7.212. systemdunitValue

```
elementId: TBD
name: systemdunitValue
dataType: string
status: current
description: The value of the property associated with a systemd
unit. Exactly one value shall be used for all property types
except dbus arrays - each array element shall be represented by
one value.
```

7.213. file

```
elementId: TBD
name: file
dataType: list
structure: list (filepath, path, filename, fileType, userId,
aTime, changeTime, mTime, size)
status: current
description: The metadata associated with a file on the endpoint.
```

7.214. fileType

```
elementId: TBD
name: fileType
dataType: string
status: current
description: The file's type (e.g., regular file (regular),
directory, named pipe (fifo), symbolic link, socket or block
special.)
```

7.215. groupId

```
elementId: TBD
name: groupId
dataType: integer
status: current
description: The group owner of the file, by
      group number.
```

7.216. aTime

```
elementId: TBD
name: aTime
dataType: timeStamp
status: current
description: The time that the file was last
      accessed.
```

7.217. changeTime

```
elementId: TBD
name: changeTime
dataType: timeStamp
status: current
description: The time of the last change
      to the file's inode.
```

7.218. mTime

```
elementId: TBD
name: mTime
dataType: timeStamp
status: current
description: The time of the last change to
      the file's contents.
```

7.219. size

```
elementId: TBD
name: size
dataType: integer
status: current
description: This is the size of the file in
      bytes.
```

7.220. suid

```
elementId: TBD
name: suid
dataType: boolean
status: current
description: Indicates whether the program runs with the uid
(thus privileges) of the file's owner, rather than the calling
user.
```

7.221. sgid

```
elementId: TBD
name: sgid
dataType: boolean
status: current
description: Indicates whether the program runs with the gid
(thus privileges) of the file's group owner, rather than the
calling user's group.
```

7.222. sticky

```
elementId: TBD
name: sticky
dataType: boolean
status: current
description: Indicates whether users can delete each other's
files in this directory, when said directory is writable by
those users.
```

7.223. hasExtendedAcl

```
elementId: TBD
name: hasExtendedAcl
dataType: boolean
status: current
description: Indicates whether the file or directory hasACL
permissions applied to it. If a system supports ACLs and the
file or directory doesn't have an ACL, or it matches the standard
UNIX permissions, the entity will have a status of 'exists' and
a value of 'false'. If the system supports ACLs and the file or
directory has an ACL, the entity will have a status of 'exists'
and a value of 'true'. Lastly, if a system doesn't support ACLs,
the entity will have a status of 'does not exist'.
```

7.224. inetd

```
elementId: TBD
name: inetd
dataType: list
structure: list (serviceProtocol, serviceName, serverProgram,
      serverArguments, endpointType, execAsUser, waitStatus)
status: current
description: Holds information associated
      with different Internet services.
```

7.225. serverProgram

```
elementId: TBD
name: serverProgram
dataType: string
status: current
description: Either the pathname of a server program to be
invoked by inetd to perform the requested service, or the value
internal if inetd itself provides the service.
```

7.226. endpointType

```
elementId: TBD
name: endpointType
dataType: enumeration
structure:
stream ; 0x1 ; The stream value is used to describe a stream
socket.
dgram ; 0x2 ; The dgram value is used to describe a datagram
socket.
raw ; 0x3 ; The raw value is used to describe a raw socket.
seqpacket ; 0x4 ; The seqpacket value is used to describe a
sequenced packet socket.
tli ; 0x5 ; The tli value is used to describe all TLI endpoints.
sunrpc_tcp ; 0x6 ; The sunrpc_tcp value is used to describe all
SUNRPC TCP endpoints.
sunrpc_udp ; 0x7 ; The sunrpc_udp value is used to describe all
SUNRPC UDP endpoints.
 ; 0x8 ; The empty string value is permitted here to allow for
detailed error reporting.
status: current
description: The endpoint type (aka, socket type) associated with
the service.
```

7.227. execAsUser

```
elementId: TBD
name: execAsUser
dataType: string
status: current
description: The user id of the user the
      server program should run under.
```

7.228. waitStatus

```
elementId: TBD
name: waitStatus
dataType: enumeration
structure: wait ; 0x1 ; The value of 'wait' specifies that the
server that is invoked by inetd will take over the listening
socket associated with the service, and once launched, inetd will
wait for that server to exit, if ever, before it resumes
listening for new service requests.

nowait ; 0x2 ; The value of 'nowait' specifies that the server
that is invoked by inetd will not wait for any existing server
to finish before taking over the listening socket associated with
the service.

; 0x3 ; The empty string value is permitted here to allow for
detailed error reporting.
status: current
description: Specifies whether the server that is invoked by
inetd will take over the listening socket associated with the
service, and whether once launched, inetd will wait for that
server to exit, if ever, before it resumes listening for new
service requests. The legal values are "wait" or "nowait".
```

7.229. inetAddr

```
elementId: TBD
name: inetAddr
dataType: ipAddress
status: current
description: The IP address of the specific interface. Note that
the IP address can be IPv4 or IPv6.
```

7.230. netmask

```
elementId: TBD
name: netmask
dataType: ipAddress
status: current
description: The bitmask used to calculate
      the interface's IP network.
```

7.231. passwordInfo

```
elementId: TBD
name: passwordInfo
dataType: list
structure: list (username, password, userId, groupId, gcos,
      homeDir, loginShell, lastLogin)
status: current
description: Describes user account information for a
      system.
```

7.232. username

```
elementId: TBD
name: username
dataType: string
status: current
description: The name of the user.
```

7.233. password

```
elementId: TBD
name: password
dataType: string
status: current
description: The encrypted version of the
      user's password.
```

7.234. gcos

```
elementId: TBD
name: gcos
dataType: string
status: current
description:
```

7.235. homeDir

```
elementId: TBD
name: homeDir
dataType: string
status: current
description: The user's home
      directory.
```

7.236. loginShell

```
elementId: TBD
name: loginShell
dataType: string
status: current
description: The user's shell
      program.
```

7.237. lastLogin

```
elementId: TBD
name: lastLogin
dataType: integer
status: current
description: The date and time when the
      last login occurred.
```

7.238. process

```
elementId: TBD
name: process
dataType: list
structure: list (commandLine, pid, ppid, priority, startTime)

status: current
description: Information about a process running on an endpoint.
```

7.239. commandLine

```
elementId: TBD
name: commandLine
dataType: string
status: current
description: The string used to start the
      process. This includes any parameters that are part of the
      command line.
```

7.240. ppid

```
elementId: TBD
name: ppid
dataType: integer
status: current
description: The process ID of the process's
      parent process.
```

7.241. priority

```
elementId: TBD
name: priority
dataType: integer
status: current
description: The scheduling priority with
      which the process runs.
```

7.242. startTime

```
elementId: TBD
name: startTime
dataType: string
status: current
description: The time of day the process
      started.
```

7.243. routingtable

```
elementId: TBD
name: routingtable
dataType: list
structure: list (destination, gateway, flags,
      interfaceName)
status: current
description: Holds information about an individual routing table
entry found in a system's primary routing table.
```

7.244. destination

```
elementId: TBD
name: destination
dataType: ipaddress
status: current
description: The destination IP address
      prefix of the routing table entry.
```

7.245. gateway

```
elementId: TBD
name: gateway
dataType: ipaddress
status: current
description: The gateway of the specified
      routing table entry.
```

7.246. runlevelInfo

```
elementId: TBD
name: runlevelInfo
dataType: list
structure: list (serviceName, runlevel, start, kill)

status: current
description: Information about the start or kill state of a
specified service at a given runlevel.
```

7.247. runlevel

```
elementId: TBD
name: runlevel
dataType: string
status: current
description: Specifies the system runlevel
      associated with a service.
```

7.248. start

```
elementId: TBD
name: start
dataType: boolean
status: current
description: Specifies whether the service is
      scheduled to start at the runlevel.
```

7.249. kill

```
elementId: TBD
name: kill
dataType: boolean
status: current
description: Specifies whether the service is
      scheduled to be killed at the runlevel.
```

7.250. shadowItem

```
elementId: TBD
name: shadowItem
dataType: list
structure: list (username, password, chgLst, chgAllow,
      chgReq, expWarn, expInact, expDate, flags, encryptMethod)
status: current
description:
```

7.251. chgLst

```
elementId: TBD
name: chgLst
dataType: timeStamp
status: current
description: The date of the last password
      change.
```

7.252. chgAllow

```
elementId: TBD
name: chgAllow
dataType: integer
status: current
description: Specifies how often in days a
      user may change their password. It can also be thought of
      as the minimum age of a password.
```

7.253. chgReq

```
elementId: TBD
name: chgReq
dataType: integer
status: current
description: Describes how long a user can
      keep a password before the system forces her to change it.
```

7.254. expWarn

```
elementId: TBD
name: expWarn
dataType: integer
status: current
description: Describes how long before
      password expiration the system begins warning the user.
```

7.255. expInact

```
elementId: TBD
name: expInact
dataType: integer
status: current
description: Describes how many days of
      account inactivity the system will wait after a password
      expires before locking the account.
```

7.256. expDate

```
elementId: TBD
name: expDate
dataType: timeStamp
status: current
description: Specifies when will the
      account's password expire.
```

7.257. encryptMethod

```
elementId: TBD
name: encryptMethod
dataType: enumeration
structure: DES ; 0x1 ; The DES method corresponds to the (none)
prefix.
      BSDi ; 0x2 ; The BSDi method corresponds to BSDi modified
      DES or the '_' prefix.
      MD5 ; 0x3 ; The MD5 method corresponds to MD5 for Linux/BSD
      or the $1$ prefix.
      Blowfish ; 0x4 ; The Blowfish method corresponds to Blowfish
      (OpenBSD) or the $2$ or $2a$ prefixes.
      Sun MD5 ; 0x5 ; The Sun MD5 method corresponds to the $md5$
      prefix.
      SHA-256 ; 0x6 ; The SHA-256 method corresponds to the $5$
      prefix.
      SHA-512 ; 0x7 ; The SHA-512 method corresponds to the $6$
      prefix. ; 0x8 ; The empty string value is permitted here to
      allow for empty elements associated with variable references.
status: current
description: Describes method that is used for hashing
      passwords.
```

7.258. symlink

```
elementId: TBD
name: symlink
dataType: list
structure: list (symlinkFilepath, canonicalPath)
status: current

description: Identifies the result generated for a symlink.
```

7.259. symlinkFilepath

```
elementId: TBD
name: symlinkFilepath
dataType: string
status: current
description: Specifies the filepath to
      the subject symbolic link file.
```

7.260. canonicalPath

```
elementId: TBD
name: canonicalPath
dataType: string
status: current
description: Specifies the canonical
      path for the target of the symbolic link file specified by
      the filepath.
```

7.261. sysctl

```
elementId: TBD
name: sysctl
dataType: list
structure: list (kernelParameterName, kernelParameterValue+,
      uname, machineClass, nodeName, osName, osRelease,
      osVersion, processorType)
status: current
description: Stores
      information retrieved from the local system about a kernel
      parameter and its respective value(s).
```

7.262. kernelParameterName

```
elementId: TBD
name: kernelParameterName
dataType: string
status: current
description: The name of a kernel
      parameter that was collected from the local system.
```

7.263. kernelParameterValue

```
elementId: TBD
name: kernelParameterValue
dataType: string
status: current
description: The current value(s)
      for the specified kernel parameter on the local system.
```

7.264. uname

```
elementId: TBD
name: uname
dataType: list
structure: list (machineClass, nodeName, osName, osRelease,
      osVersion, processorType)
status: current
description: Information about the hardware the machine is running
      on.
```

7.265. machineClass

```
elementId: TBD
name: machineClass
dataType: string
status: current
description: Specifies the machine
      hardware name.
```

7.266. nodeName

```
elementId: TBD
name: nodeName
dataType: string
status: current
description: Specifies the host
      name.
```

7.267. osName

```
elementId: TBD
name: osName
dataType: string
status: current
description: Specifies the operating system
      name.
```

7.268. osRelease

```
elementId: TBD
name: osRelease
dataType: string
status: current
description: Specifies the build
      version.
```

7.269. osVersion

```
elementId: TBD
name: osVersion
dataType: string
status: current
description: Specifies the operating system
      version.
```

7.270. processorType

```
elementId: TBD
name: processorType
dataType: string
status: current
description: Specifies the processor
      type.
```

7.271. internetService

```
elementId: TBD
name: internetService
dataType: list
structure: list (serviceProtocol, serviceName, flags,
      noAccess, onlyFrom, port, server, serverArguments,
      socketType, registeredServiceType, user, wait, disabled)

status: current
description: Holds information associated with Internet services.
```

7.272. serviceProtocol

```
elementId: TBD
name: serviceProtocol
dataType: string
status: current
description: Specifies the protocol
      that is used by the service.
```

7.273. serviceName

```
elementId: TBD
name: serviceName
dataType: string
status: current
description: Specifies the name of the
      service.
```

7.274. flags

```
elementId: TBD
name: flags
dataType: string
status: current
description: Specifies miscellaneous settings
      associated with the service with executing a program.
```

7.275. noAccess

```
elementId: TBD
name: noAccess
dataType: string
status: current
description: Specifies the remote hosts to
      which the service is unavailable.
```

7.276. onlyFrom

```
elementId: TBD
name: onlyFrom
dataType: ipAddress
status: current
description: Specifies the remote hosts to
      which the service is available.
```

7.277. port

```
elementId: TBD
name: port
dataType: integer
status: current
description: The port entity specifies the port
      used by the service.
```

7.278. server

```
elementId: TBD
name: server
dataType: string
status: current
description: Specifies the executable that is
      used to launch the service.
```

7.279. serverArguments

```
elementId: TBD
name: serverArguments
dataType: string
status: current
description: Specifies the arguments
      that are passed to the executable when launching the service.
```

7.280. socketType

```
elementId: TBD
name: socketType
dataType: string
status: current
description: Specifies the type of socket
      that is used by the service. Possible values include: stream,
      dgram, raw, or seqpacket.
```

7.281. registeredServiceType

```
elementId: TBD
name: registeredServiceType
dataType: enumeration
structure: INTERNAL ; 0x1 ; The INTERNAL type is used to describe
services like echo, chargen, and others whose functionality is
supplied by xinetd itself.
      RPC ; 0x2 ; The RPC type is used to describe services that
      use remote procedure call ala NFS.
      UNLISTED ; 0x3 ; The UNLISTED type is used to describe
      services that aren't listed in /etc/protocols or /etc/rpc.
      TCPMUX ; 0x4 ; The TCPMUX type is used to describe services
      that conform to RFC 1078. This type indiciates that the service
      is responsible for handling the protocol handshake.
      TCPMUXPLUS ; 0x5 ; The TCPMUXPLUS type is used to describe
      services that conform to RFC 1078. This type indicates that
      xinetd is responsible for handling the protocol
      handshake.
      ; 0x6 ; The empty string value is permitted here to allow
      for detailed error reporting.
status: current

description: Specifies the type of internet service.
```

7.282. wait

```
elementId: TBD
name: wait
dataType: boolean
status: current
description: Specifies whether or not the service is single-threaded
or multi-threaded and whether or not xinetd accepts the connection
or the service accepts the connection. A value of 'true' indicates
that the service is single-threaded and the service will accept the
connection. A value of 'false' indicates that the service is multi-
threaded and xinetd will accept the connection.
```

7.283. disabled

```
elementId: TBD
name: disabled
dataType: boolean
status: current
description: Specifies whether or not the
      service is disabled. A value of 'true' indicates that the
      service is disabled and will not start. A value of
      'false' indicates that the service is not disabled.
```

7.284. windowsView

```
elementId: TBD
name: windowsView
dataType: enumeration
structure: 32_bit ; 0x1 ; Indicates the 32_bit windows view.
64_bit ; 0x2 ; Indicates the 64_bit windows view.
; 0x3 ; The empty string value is permitted here to allow for
empty elements associated with error conditions.
status: current
description: Indicates from which
      view (32-bit or 64-bit), the information was collected.
      A value of '32_bit' indicates the Item was collected from
      the 32-bit view. A value of '64-bit' indicates the Item
      was collected from the 64-bit view.
```

7.285. fileauditedpermissions

```
elementId: TBD
name: fileauditedpermissions
dataType: list
structure: list (filepath, path, filename,
      trusteeSid, trusteeName, auditStandardDelete,
      auditStandardReadControl, auditStandardWriteDac,
      auditStandardWriteOwner, auditStandardSynchronize,
      auditAccessSystemSecurity, auditGenericRead, auditGenericWrite,
      auditGenericExecute, auditGenericAll, auditFileReadData,
      auditFileWriteData, auditFileAppendData, auditFileReadEa,
      auditFileWriteEa, auditFileExecute, auditFileDeleteChild,
      auditFileReadAttributes, auditFileWriteAttributes,
      windowsView)
status: current
description: Stores the audited access rights of a file that a
system access control list (SACL) structure grants to a specified
trustee. The trustee's audited access rights are determined checking
all access control entries (ACEs) in the SACL.
```

7.286. trusteeName

```
elementId: TBD
name: trusteeName
dataType: string
status: current
description: Specifies the trustee name. A
      trustee can be a user, group, or program (such as a Windows
      service).
```

7.287. auditStandardDelete

elementId: TBD
name: auditStandardDelete
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: The right to delete the object.

7.288. auditStandardReadControl

elementId: TBD
name: auditStandardReadControl
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: The right to read the information in the object's security descriptor, not including the information in the SACL.

7.289. auditStandardWriteDac

elementId: TBD
name: auditStandardWriteDac
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: The right to modify the DACL in the object's security descriptor.

7.290. auditStandardWriteOwner

elementId: TBD
name: auditStandardWriteOwner
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: The right to change the owner in the object's security descriptor.

7.291. auditStandardSynchronize

elementId: TBD
name: auditStandardSynchronize
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: The right to use the object for synchronization. This enables a thread to wait until the object is in the signaled state. Some object types do not support this access right.

7.292. auditAccessSystemSecurity

elementId: TBD
name: auditAccessSystemSecurity
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Indicates access to a system access control list (SACL).

7.293. auditGenericRead

elementId: TBD
name: auditGenericRead
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Read access.

7.294. auditGenericWrite

elementId: TBD
name: auditGenericWrite
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Write access.

7.295. auditGenericExecute

elementId: TBD
name: auditGenericExecute
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Execute access.

7.296. auditGenericAll

elementId: TBD
name: auditGenericAll
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Read, write, and execute access.

7.297. auditFileReadData

elementId: TBD
name: auditFileReadData
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Grants the right to read data from the file.

7.298. auditFileWriteData

elementId: TBD
name: auditFileWriteData
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Grants the right to write data to the file.

7.299. auditFileAppendData

elementId: TBD
name: auditFileAppendData
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Grants the right to append data to the file.

7.300. auditFileReadEa

elementId: TBD
name: auditFileReadEa
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Grants the right to read extended attributes.

7.301. auditFileWriteEa

elementId: TBD
name: auditFileWriteEa
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Grants the right to write extended attributes.

7.302. auditFileExecute

elementId: TBD
name: auditFileExecute
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Grants the right to execute a file.

7.303. auditFileDeleteChild

elementId: TBD
name: auditFileDeleteChild
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Right to delete a directory and all the files it contains (its children), even if the files are read-only.

7.304. auditFileReadAttributes

elementId: TBD
name: auditFileReadAttributes
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description: Grants the right to read file attributes.

7.305. auditFileWriteAttributes

```
elementId: TBD
name: auditFileWriteAttributes
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is
used to perform audits on all unsuccessful occurrences of
specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel
all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to
perform audits on all successful occurrences of the specified
events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE
is used to perform audits on all successful and unsuccessful
occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for
detailed error reporting.
status: current
description: Grants the right to change file attributes.
```

7.306. fileeffectiverights

```
elementId: TBD
name: fileeffectiverights
dataType: list
structure: list (filepath, path, filename,
      trusteeSid, trusteeName, standardDelete, standardReadControl,
      standardWriteDac, standardWriteOwner,
      standardSynchronize, accessSystemSecurity, genericRead,
      genericWrite, genericExecute, genericAll, fileReadData,
      fileWriteData, fileAppendData, fileReadEa, fileWriteEa,
      fileExecute, fileDeleteChild, fileReadAttributes,
      fileWriteAttributes, windowsView)
status: current
description: Stores the effective rights of a file that a
      discretionary access control list (DACL) structure grants
      to a specified trustee. The trustee's effective rights
      are determined checking all access-allowed and access-denied
      access control entries (ACEs) in the DACL.
```

7.307. standardDelete

```
elementId: TBD
name: standardDelete
dataType: boolean
status: current
description: The right to delete the
      object.
```

7.308. standardReadControl

```
elementId: TBD
name: standardReadControl
dataType: boolean
status: current
description: The right to read
      the information in the object's security descriptor, not
      including the information in the SACL.
```

7.309. standardWriteDac

```
elementId: TBD
name: standardWriteDac
dataType: boolean
status: current
description: The right to modify the
      DACL in the object's security descriptor.
```

7.310. standardWriteOwner

```
elementId: TBD
name: standardWriteOwner
dataType: boolean
status: current
description: The right to change
      the owner in the object's security descriptor.
```

7.311. standardSynchronize

```
elementId: TBD
name: standardSynchronize
dataType: boolean
status: current
description: The right to use the
      object for synchronization. This enables a thread to wait
      until the object is in the signaled state. Some object
      types do not support this access right.
```

7.312. accessSystemSecurity

```
elementId: TBD
name: accessSystemSecurity
dataType: boolean
status: current
description: Indicates access to
      a system access control list (SACL).
```

7.313. genericRead

```
elementId: TBD
name: genericRead
dataType: boolean
status: current
description: Read access.
```

7.314. genericWrite

```
elementId: TBD
name: genericWrite
dataType: boolean
status: current
description: Write access.
```

7.315. genericExecute

```
elementId: TBD
name: genericExecute
dataType: boolean
status: current
description: Execute access.
```

7.316. genericAll

```
elementId: TBD
name: genericAll
dataType: boolean
status: current
description: Read, write, and execute
      access.
```

7.317. fileReadData

```
elementId: TBD
name: fileReadData
dataType: boolean
status: current
description: Grants the right to read
      data from the file
```

7.318. fileWriteData

```
elementId: TBD
name: fileWriteData
dataType: boolean
status: current
description: Grants the right to write
      data to the file.
```

7.319. fileAppendData

```
elementId: TBD
name: fileAppendData
dataType: boolean
status: current
description: Grants the right to
      append data to the file.
```

7.320. fileReadEa

```
elementId: TBD
name: fileReadEa
dataType: boolean
status: current
description: Grants the right to read
      extended attributes.
```

7.321. fileWriteEa

```
elementId: TBD
name: fileWriteEa
dataType: boolean
status: current
description: Grants the right to write
      extended attributes.
```

7.322. fileExecute

```
elementId: TBD
name: fileExecute
dataType: boolean
status: current
description: Grants the right to execute
      a file.
```

7.323. fileDeleteChild

```
elementId: TBD
name: fileDeleteChild
dataType: boolean
status: current
description: Right to delete a
      directory and all the files it contains (its children),
      even if the files are read-only.
```

7.324. fileReadAttributes

```
elementId: TBD
name: fileReadAttributes
dataType: boolean
status: current
description: Grants the right to
      read file attributes.
```

7.325. fileWriteAttributes

```
elementId: TBD
name: fileWriteAttributes
dataType: boolean
status: current
description: Grants the right to
      change file attributes.
```

7.326. groupInfo

```
elementId: TBD
name: groupInfo
dataType: list
structure: list (group, username, subgroup)
status: current
description: Specifies the different users and subgroups, that
directly belong to specific groups.
```

7.327. group

```
elementId: TBD
name: group
dataType: string
status: current
description: Represents the name of a particular
      group.
```

7.328. user

```
elementId: TBD
name: user
dataType: string
status: current
description: Represents the name of a particular
      user.
```

7.329. subgroup

```
elementId: TBD
name: subgroup
dataType: string
status: current
description: Represents the name of a
      particular subgroup in the specified group.
```

7.330. groupSidInfo

```
elementId: TBD
name: groupSidInfo
dataType: list
structure: list (groupSid, userSid, subgroupSid)
status:
      current
description: Specifies the different users and subgroups, that
directly belong to specific groups
      (identified by SID).
```

7.331. userSidInfo

```
elementId: TBD
name: userSidInfo
dataType: list
structure: list (userSid, enabled, groupSid, lastLogon)

status: current
description: Specifies the different groups (identified by SID)
that a user belongs to.
```

7.332. userSid

```
elementId: TBD
name: userSid
dataType: string
status: current
description: Represents the SID of a
      particular user.
```

7.333. subgroupSid

```
elementId: TBD
name: subgroupSid
dataType: string
status: current
description: Represents the SID of a
      particular subgroup.
```

7.334. lockoutpolicy

```
elementId: TBD
name: lockoutpolicy
dataType: list
structure: list (forceLogoff, lockoutDuration,
      lockoutObservationWindow, lockoutThreshold)
status: current
description: Specifies various attributes associated
      with lockout information for users and global groups in the
      security database.
```

7.335. forceLogoff

```
elementId: TBD
name: forceLogoff
dataType: integer
status: current
description: Specifies, in seconds, the
      amount of time between the end of the valid logon time and
      the time when the user is forced to log off the
      network.
```

7.336. lockoutDuration

```
elementId: TBD
name: lockoutDuration
dataType: integer
status: current
description: Specifies, in seconds,
      how long a locked account remains locked before it is
      automatically unlocked.
```

7.337. lockoutObservationWindow

```
elementId: TBD
name: lockoutObservationWindow
dataType: integer
status: current
description: Specifies the
      maximum time, in seconds, that can elapse between any two
      failed logon attempts before lockout occurs.
```

7.338. lockoutThreshold

```
elementId: TBD
name: lockoutThreshold
dataType: integer
status: current
description: Specifies the number of
      invalid password authentications that can occur before an
      account is marked "locked out."
```

7.339. passwordpolicy

```
elementId: TBD
name: passwordpolicy
dataType: list
structure: list (maxPasswdAge, minPasswdAge,
      minPasswdLen, passwordHistLen, passwordComplexity,
      reversibleEncryption)
status: current
description: Specifies
      policy information associated with passwords.
```

7.340. maxPasswdAge

```
elementId: TBD
name: maxPasswdAge
dataType: integer
status: current
description: Specifies, in seconds (from
      a DWORD), the maximum allowable password age. A value of
      TIMEQ_FOREVER (max DWORD value, 4294967295) indicates
      that the password never expires. The minimum valid value
      for this element is ONE_DAY (86400). See the
      USER_MODALS_INFO_0 structure returned by a call to
      NetUserModalsGet().
```

7.341. minPasswdAge

```
elementId: TBD
name: minPasswdAge
dataType: integer
status: current
description: Specifies the minimum
      number of seconds that can elapse between the time a password
      changes and when it can be changed again. A value of
      zero indicates that no delay is required between password
      updates.
```

7.342. minPasswdLen

```
elementId: TBD
name: minPasswdLen
dataType: integer
status: current
description: Specifies the minimum
      allowable password length. Valid values for this element are
      zero through PWLEN.
```

7.343. passwordHistLen

```
elementId: TBD
name: passwordHistLen
dataType: integer
status: current
description: Specifies the length of
      password history maintained. A new password cannot match any
      of the previous usrmod0_password_hist_len passwords.
      Valid values for this element are zero through DEF_MAX_PWHIST.
```

7.344. passwordComplexity

```
elementId: TBD
name: passwordComplexity
dataType: boolean
status: current
description: Indicates whether
      passwords must meet the complexity requirements put forth
      by the operating system.
```

7.345. reversibleEncryption

```
elementId: TBD
name: reversibleEncryption
dataType: boolean
status: current
description: Indicates whether
      or not passwords are stored using reversible encryption.
```

7.346. portInfo

```
elementId: TBD
name: portInfo
dataType: list
structure: list (localAddress, localPort, transportProtocol,
      pid, foreignAddress, foreignPort)
status: current
description: Information about open listening ports.
```

7.347. foreignPort

```
elementId: TBD
name: foreignPort
dataType: string
status: current
description: The TCP or UDP port to which
      the program communicates.
```

7.348. printereffectiverights

```
elementId: TBD
name: printereffectiverights
dataType: list
structure: list (printerName, trusteeSid,
      standardDelete, standardReadControl, standardWriteDac,
      standardWriteOwner, standardSynchronize,
      accessSystemSecurity, genericRead, genericWrite,
      genericExecute, genericAll, printerAccessAdminister,
      printerAccessUse, jobAccessAdminister, jobAccessRead)
status: current
description: Stores the effective rights of a printer that a
discretionary access control list (DACL) structure grants to a
specified trustee. The trustee's effective rights are determined
checking all access-allowed and access-denied access control
entries (ACEs) in the DACL.
```

7.349. printerName

```
elementId: TBD
name: printerName
dataType: string
status: current
description: Specifies the name of the
      printer.
```

7.350. printerAccessAdminister

```
elementId: TBD
name: printerAccessAdminister
dataType: boolean
status: current
description:
```

7.351. printerAccessUse

```
elementId: TBD
name: printerAccessUse
dataType: boolean
status: current
description:
```

7.352. jobAccessAdminister

```
elementId: TBD
name: jobAccessAdminister
dataType: boolean
status: current
description:
```

7.353. jobAccessRead

```
elementId: TBD
name: jobAccessRead
dataType: boolean
status: current
description:
```

7.354. registry

```
elementId: TBD
name: registry
dataType: list
structure: list (hive, key, registryKeyName, lastWriteTime,
      registryKeyType, registryKeyValue, windowsView)
status: current
description: Specifies information that can be
      collected about a particular registry key.
```

7.355. hive

```
elementId: TBD
name: hive
dataType: enumeration
structure: HKEY_CLASSES_ROOT ; 0x1 ; This registry subtree
      contains information that associates file types with programs
      and configuration data for automation (e.g. COM
      objects and Visual Basic Programs).
      HKEY_CURRENT_CONFIG ; 0x2 ; This registry subtree contains
      configuration data for the current hardware profile.
      HKEY_CURRENT_USER ; 0x3 ; This registry subtree contains the
      user profile of the user that is currently logged into the
      system.
      HKEY_LOCAL_MACHINE ; 0x4 ; This registry subtree contains
      information about the local system.
      HKEY_USERS ; 0x5 ; This registry subtree contains user-specific
      data.
      ; 0x6 ; The empty string value is permitted here to allow
      for detailed error reporting.
status: current
description: The
      hive that the registry key belongs to.
```

7.356. registryKey

```
elementId: TBD
name: registryKey
dataType: string
status: current
description: Describes the registry key.
      Note that the hive portion of the string should not be
      included, as this data can be found under the hive
      element.
```

7.357. registryKeyName

```
elementId: TBD
name: registryKeyName
dataType: string
status: current
description: Describes the name of a
      registry key.
```

7.358. lastWriteTime

```
elementId: TBD
name: lastWriteTime
dataType: integer
status: current
description: The last time that the key or any of its value entries
      were modified. The value of this entity represents the
      FILETIME structure which is a 64-bit value representing the
      number of 100-nanosecond intervals since January 1, 1601
      (UTC). Last write time can be queried on any key, with hives
      being classified as a type of key. When collecting only
      information about a registry hive or key the last write time
      will be the time the key or any of its entries were modified.
      When collecting only information about a registry name the
      last write time will be the time the containing key was
      modified. Thus when collecting information about a registry
      name, the last write time does not correlate directly
      to the specified name. See the RegQueryInfoKey function
      lpftLastWriteTime.
```

7.359. registryKeyType

```
elementId: TBD
name: registryKeyType
dataType: enumeration
structure: reg_binary ; 0x1 ; The reg_binary type
      is used by registry keys that specify binary data in any
      form.
      reg_dword ; 0x2 ; The reg_dword type is used by
      registry keys that specify an unsigned 32-bit integer.
      reg_dword_little_endian ; 0x3 ; The reg_dword_little_endian
      type is used by registry keys that specify an unsigned 32-bit
      little-endian integer. It is designed to run on
      little-endian computer architectures.
      reg_dword_big_endian ; 0x4 ; The reg_dword_big_endian type
      is used by registry keys that specify an unsigned 32-bit
      big-endian integer. It is designed to run on big-endian
      computer architectures.
```

```
        reg_expand_sz ; 0x5 ; The reg_expand_sz type is used by
        registry keys to specify a null-terminated
        string that contains unexpanded references to environment
        variables (for example, "%PATH%").
        reg_link ; 0x6 ; The reg_link type is used by the registry
        keys for null-terminated unicode strings. It is related to
        target path of a symbolic link created by the
        RegCreateKeyEx function.
        reg_multi_sz ; 0x7 ; The reg_multi_sz type is used by
        registry keys that specify an array of null-terminated
        strings, terminated by two null characters.
        reg_none; 0x8 ;
        The reg_none type is used by registry keys that have no
        defined value type.
        reg_qword; 0x9 ; The reg_qword type is used by registry keys
        that specify an unsigned 64-bit integer.
        reg_qword_little_endian; 0xA ; The reg_qword_little_endian
        type is used by registry keys that specify an unsigned
        64-bit integer in little-endian computer architectures.
        reg_sz; 0xB ; The reg_sz type is used by registry keys that
        specify a single null-terminated string.
        reg_resource_list; 0xC ; The reg_resource_list type is used
        by registry keys that specify a resource list.
        reg_full_resource_descriptor; 0xD ; The
        reg_full_resource_descriptor type is used by registry
        keys that specify a full resource descriptor.
        reg_resource_requirements_list; 0xE ; The
        reg_resource_requirements_list type is used by registry keys
        that specify a resource requirements list.
        ; 0xF ; The empty string value is permitted here to allow
        for detailed error reporting.
   status: current
   description:
        Specifies the type of data stored by the registry key.
```

7.360. registryKeyValue

```
elementId: TBD
name: registryKeyValue
dataType: string
status: current
description: Holds the actual value
      of the specified registry key. The representation of the
      value as well as the associated datatype attribute
      depends on type of data stored in the registry key. If the
      value being tested is of type REG_BINARY, then the
      datatype attribute should be set to 'binary' and the data
      represented by the value entity should follow the
      xsd:hexBinary form. (each binary octet is encoded as two hex
      digits) If the value being tested is of type
      REG_DWORD, REG_QWORD, REG_DWORD_LITTLE_ENDIAN,
      REG_DWORD_BIG_ENDIAN, or REG_QWORD_LITTLE_ENDIAN then the
      datatype attribute should be set to 'int' and the value
      entity should represent the data as an unsigned integer.
      DWORD and QWORD values represnt unsigned 32-bit and 64-bit
      integers, respectively. If the value being tested is of type
      REG_EXPAND_SZ, then the datatype attribute should be set to
      'string' and the pre-expanded string should be
      represented by the value entity. If the value being tested
      is of type REG_MULTI_SZ, then only a single string (one
      of the multiple strings) should be tested using the value
      entity with the datatype attribute set to 'string'. In
      order to test multiple values, multiple OVAL registry tests
      should be used. If the specified registry key is of
      type REG_SZ, then the datatype should be 'string' and the
      value entity should be a copy of the string. If the
      value being tested is of type REG_LINK, then the datatype
      attribute should be set to 'string' and the
      null-terminated Unicode string should be represented by the
      value entity.
```

7.361. regkeyauditedpermissions

```
elementId: TBD
name: regkeyauditedpermissions
dataType: list
structure: list (key, trusteeSid, trusteeName,
      standardDelete, standardReadControl, standardWriteDac,
      standardWriteOwners, tandardSynchronize,
      accessSystemSecurity, genericRead, genericWrite,
      genericExecute, genericAll, keyQueryValue, keySetValue,
      keyCreateSubKey, keyEnumerateSubKeys, keyNotify,
      keyCreateLink, keyWow6464Key, keyWow6432Key, keyWow64Res,
      windowsView)
status: current
description: Stores the audited access rights of a registry key
that a system access control list (SACL) structure grants to a
specified trustee. The trustee's audited access rights are
determined checking all access control entries (ACEs) in the SACL.
```

7.362. auditKeyQueryValue

```
elementId: TBD
name: auditKeyQueryValue
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is
used to perform audits on all unsuccessful occurrences of
specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel
all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to
perform audits on all successful occurrences of the specified
events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE
is used to perform audits on all successful and unsuccessful
occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for
detailed error reporting.
status: current
description:
```

7.363. auditKeySetValue

elementId: TBD
name: auditKeySetValue
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description:

7.364. auditKeyCreateSubKey

elementId: TBD
name: auditKeyCreateSubKey
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description:

7.365. auditKeyEnumerateSubKeys

elementId: TBD
name: auditKeyEnumerateSubKeys
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description:

7.366. auditKeyNotify

elementId: TBD
name: auditKeyNotify
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description:

7.367. auditKeyCreateLink

elementId: TBD
name: auditKeyCreateLink
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description:

7.368. auditKeyWow6464Key

elementId: TBD
name: auditKeyWow6464Key
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description:

7.369. auditKeyWow6432Key

elementId: TBD
name: auditKeyWow6432Key
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description:

7.370. auditKeyWow64Res

elementId: TBD
name: auditKeyWow64Res
dataType: enumeration
structure: AUDIT_FAILURE ; 0x1 ; The audit type AUDIT_FAILURE is used to perform audits on all unsuccessful occurrences of specified events when auditing is enabled.
AUDIT_NONE ; 0x2 ; The audit type AUDIT_NONE is used to cancel all auditing options for the specified events.
AUDIT_SUCCESS ; 0x3 ; The audit type AUDIT_SUCCESS is used to perform audits on all successful occurrences of the specified events when auditing is enabled.
AUDIT_SUCCESS_FAILURE ; 0x4 ; The audit type AUDIT_SUCCESS_FAILURE is used to perform audits on all successful and unsuccessful occurrences of the specified events when auditing is enabled.
; 0x5 ; The empty string value is permitted here to allow for detailed error reporting.
status: current
description:

7.371. regkeyeffectiverights

```
elementId: TBD
name: regkeyeffectiverights
dataType: list
structure: list (hive, key, trusteeSid,
      trusteeName, standardDelete, standardReadControl,
      standardWriteDac, standardWriteOwner, standardSynchronize,
      accessSystemSecurity, genericRead, genericWrite,
      genericExecute, genericAll, keyQueryValue, keySetValue,
      keyCreateSubKey, keyEnumerateSubKeys, keyNotify,
      keyCreateLink, keyWow6464Key, keyWow6432Key, keyWow64Res,
      windowsView)
status: current
description: Stores the effective rights of a registry key that a
discretionary access control list (DACL) structure grants to a
specified trustee. The trustee's effective rights are determined
checking all access-allowed and access-denied access control
entries (ACEs) in the DACL.
```

7.372. keyQueryValue

```
elementId: TBD
name: keyQueryValue
dataType: boolean
status: current
description: Specifies whether or not
      permission is granted to query the key's value.
```

7.373. keySetValue

```
elementId: TBD
name: keySetValue
dataType: boolean
status: current
description: Specifies whether or not
      permission is granted to set the key's value.
```

7.374. keyCreateSubKey

```
elementId: TBD
name: keyCreateSubKey
dataType: boolean
status: current
description: Specifies whether or not
      permission is granted to create a subkey.
```

7.375. keyEnumerateSubKeys

```
elementId: TBD
name: keyEnumerateSubKeys
dataType: boolean
status: current
description: Specifies whether or
      not permission is granted to list the subkeys associated
      with key.
```

7.376. keyNotify

```
elementId: TBD
name: keyNotify
dataType: boolean
status: current
description:
```

7.377. keyCreateLink

```
elementId: TBD
name: keyCreateLink
dataType: boolean
status: current
description:
```

7.378. keyWow6464Key

```
elementId: TBD
name: keyWow6464Key
dataType: boolean
status: current
description:
```

7.379. keyWow6432Key

```
elementId: TBD
name: keyWow6432Key
dataType: boolean
status: current
description:
```

7.380. keyWow64Res

```
elementId: TBD
name: keyWow64Res
dataType: boolean
status: current
description:
```

7.381. service

```
elementId: TBD
name: service
dataType: list
structure: list (serviceName, displayName, description,
      serviceType, tartType, currentState, controlsAccepted,
      startName, path, pid, serviceFlag, dependencies)
status: current
description: Stores information about Windows services that are
present on the system.
```

7.382. displayName

```
elementId: TBD
name: displayName
dataType: string
status: current
description: Specifies the name of the
      service as specified in administrative tools.
```

7.383. description

```
elementId: TBD
name: description
dataType: string
status: current
description: Specifies the description of
      the service.
```

7.384. serviceType

```
elementId: TBD
name: serviceType
dataType: enumeration
structure: SERVICE_FILE_SYSTEM_DRIVER ; 0x1 ; The
      SERVICE_FILE_SYSTEM_DRIVER type means that the service is
      a file system driver. The DWORD value that this
      corresponds to is 0x00000002.
      SERVICE_KERNEL_DRIVER ; 0x2 ; The SERVICE_KERNEL_DRIVER type
      means that the service is a driver. The DWORD value that
      this corresponds to is 0x00000001.
      SERVICE_WIN32_OWN_PROCESS ; 0x3 ; The SERVICE_WIN32_OWN_PROCESS
      type means that the service runs in its own process. The DWORD
      value that this corresponds to is 0x00000010.
      SERVICE_WIN32_SHARE_PROCESS ; 0x4 ; The
      SERVICE_WIN32_SHARE_PROCESS type means that the service runs
      in a process with other services. The DWORD value that this
      corresponds to is 0x00000020.
      SERVICE_INTERACTIVE_PROCESS ; 0x5 ; The
      SERVICE_WIN32_SHARE_PROCESS type means that the service runs
      in a process with other services. The DWORD value that this
      corresponds to is 0x00000100.
      ; 0x6 ; The empty string value is permitted here to allow for
      empty elements associated with error conditions.
status: current
description:
      Specifies the type of the service.
```

7.385. startType

```
elementId: TBD
name: startType
dataType: enumeration
structure: SERVICE_AUTO_START ; 0x1 ; The SERVICE_AUTO_START type
      means that the service is started automatically by the Service
      Control Manager (SCM) during startup. The DWORD value that
      this corresponds to is 0x00000002.
      SERVICE_BOOT_START ; 0x2 ; The SERVICE_BOOT_START type means
      that the driver service is started by the system loader. The
      DWORD value that this corresponds to is 0x00000000.
      SERVICE_DEMAND_START ; 0x3 ; The SERVICE_DEMAND_START type
      means that the service is started by the Service Control
      Manager (SCM) when StartService() is called. The DWORD value
      that this corresponds to is 0x00000003.
      SERVICE_DISABLED ; 0x4 ; The SERVICE_DISABLED type means
      that the service cannot be started. The DWORD value that
      this corresponds to is 0x00000004.
      SERVICE_SYSTEM_START ; 0x5 ; The SERVICE_SYSTEM_START type
      means that the service is a device driver started by
      IoInitSystem(). The DWORD value that this corresponds to is
      0x00000001.
      ; 0x6 ; The empty string value is permitted here to allow
      for empty elements associated with error conditions.
status: current
description: Specifies when the service should be started.
```

7.386. currentState

```
   elementId: TBD
   name: currentState
   dataType: enumeration
   structure: SERVICE_CONTINUE_PENDING ; 0x1 ; The
         SERVICE_CONTINUE_PENDING type means that the service has been
         sent a command to continue, however, the command has
         not yet been executed. The DWORD value that this corresponds
         to is 0x00000005. SERVICE_PAUSE_PENDING ; 0x2 ; The
         SERVICE_PAUSE_PENDING type means that the service has been
         sent a command to pause, however, the command has not
         yet been executed. The DWORD value that this corresponds to
         is 0x00000006.
         SERVICE_PAUSED ; 0x3 ; The SERVICE_PAUSED type means that
         the service is paused. The DWORD value that this corresponds
         to is 0x00000007.
         SERVICE_RUNNING ; 0x4 ; The SERVICE_RUNNING type means that
         the service is running. The DWORD value that this
         corresponds to is 0x00000004.
         SERVICE_START_PENDING ; 0x5 ; The SERVICE_START_PENDING type
         means that the service has been sent a command to start,
         however, the command has not yet been executed. The DWORD
         value that this corresponds to is 0x00000002.
         SERVICE_STOP_PENDING ; 0x6 ; The SERVICE_STOP_PENDING type
         means that the service
         has been sent a command to stop, however, the command has
         not yet been executed. The DWORD value that this
         corresponds to is 0x00000003.
         SERVICE_STOPPED ; 0x7 ; The SERVICE_STOPPED type means that
         the service is stopped. The DWORD value that this corresponds
         to is 0x00000001.
         ; 0x8 ; The empty string value is permitted here to allow
         for empty elements associated with error conditions.
   status: current
   description: Specifies the current state of
         the service.
```

7.387. controlsAccepted

```
 elementId: TBD
 name: controlsAccepted
 dataType: enumeration strucutre: SERVICE_ACCEPT_NETBINDCHANGE ; 0x1 ;
       The SERVICE_ACCEPT_NETBINDCHANGE type means that the
       service is a network component and can accept changes in its
       binding without being stopped or restarted. The DWORD value
       that this corresponds to is 0x00000010.
       SERVICE_ACCEPT_PARAMCHANGE ; 0x2 ; The SERVICE_ACCEPT_PARAMCHANGE
       type means that the service can re-read its
       startup parameters without being stopped or restarted. The
```

```
      DWORD value that this corresponds to is 0x00000008.
      SERVICE_ACCEPT_PAUSE_CONTINUE ; 0x3 ; The
      SERVICE_ACCEPT_PAUSE_CONTINUE type means that the service
      can be paused or continued. The DWORD value that this
      corresponds to is 0x00000002.
      SERVICE_ACCEPT_PRESHUTDOWN ; 0x4 ; The
      SERVICE_ACCEPT_PRESHUTDOWN type means that the service can
      receive pre-shutdown notifications. The DWORD value
      that this corresponds to is 0x00000100.
      SERVICE_ACCEPT_SHUTDOWN ; 0x5 ; The SERVICE_ACCEPT_SHUTDOWN
      type means that the service can receive shutdown notifications.
      The DWORD value that this corresponds to is 0x00000004.
      SERVICE_ACCEPT_STOP ; 0x6 ; The SERVICE_ACCEPT_STOP type
      means that the service can be stopped. The DWORD value
      that this corresponds to is 0x00000001.
      SERVICE_ACCEPT_HARDWAREPROFILECHANGE ; 0x7 ; The
      SERVICE_ACCEPT_HARDWAREPROFILECHANGE type means that the
      service can receive notifications when the system's
      hardware profile changes. The DWORD value that this
      corresponds to is 0x00000020.
      SERVICE_ACCEPT_POWEREVENT ; 0x8 ; The SERVICE_ACCEPT_POWEREVENT
      type means that the service can receive notifications when the
      system's power status has changed. The DWORD value that this
      corresponds to is 0x00000040.
      SERVICE_ACCEPT_SESSIONCHANGE ; 0x9 ; The
      SERVICE_ACCEPT_SESSIONCHANGE type means that the service can
      receive notifications when the system's session
      status has changed. The DWORD value that this corresponds
      to is 0x00000080.
      SERVICE_ACCEPT_TIMECHANGE ; 0xA ; The SERVICE_ACCEPT_TIMECHANGE
      type means that the service can receive notifications when
      the system time changes. The DWORD value that this corresponds
      to is 0x00000200.
      SERVICE_ACCEPT_TRIGGEREVENT ; 0xB ; The
      SERVICE_ACCEPT_TRIGGEREVENT type means that the service can
      receive notifications when an event that the service
      has registered for occurs on the system. The DWORD value that
      this corresponds to is 0x00000400.
      ; 0xC ; The empty string value is permitted here to allow
      for empty elements associated with error conditions.
status: current

description: Specifies the control codes that a service will
             accept and process.
```

7.388. startName

```
elementId: TBD
name: startName
dataType: string
status: current
description: Specifies the account under
      which the process should run.
```

7.389. serviceFlag

```
elementId: TBD
name: serviceFlag
dataType: boolean
status: current
description: Specifies whether the
      service is in a system process that must always run (true)
      or if the service is in a non-system process or is not
      running (false).
```

7.390. dependencies

```
elementId: TBD
name: dependencies
dataType: string
status: current
description: Specifies the dependencies
      of this service on other services.
```

7.391. serviceeffectiverights

```
elementId: TBD
name: serviceeffectiverights
dataType: list
structure: list (serviceName, trusteeSid,
      standardDelete, standardReadControl, standardWriteDac,
      standardWriteOwner, genericRead, genericWrite,
      genericExecute, serviceQueryConfs, erviceChangeConf,
      serviceQueryStat, serviceEnumDependents, serviceStart,
      serviceStop, servicePause, serviceInterrogate,
      serviceUserDefined)
status: current
description: Stores the
      effective rights of a service that a discretionary access
      control list (DACL) structure grants to a specified
      trustee. The trustee's effective rights are determined by
      checking all access-allowed and access-denied access
      control entries (ACEs) in the DACL.
```

7.392. trusteeSid

```
elementId: TBD
name: trusteeSid
dataType: string
status: current
description: Specifies the SID that is
      associated with a user, group, system, or program (such as a
      Windows service).
```

7.393. serviceQueryConf

```
elementId: TBD
name: serviceQueryConf
dataType: boolean
status: current
description: Specifies whether or
      not permission is granted to query the service configuration.
```

7.394. serviceChangeConf

```
elementId: TBD
name: serviceChangeConf
dataType: boolean
status: current
description: Specifies whether or
      not permission is granted to change service configuration.
```

7.395. serviceQueryStat

```
elementId: TBD
name: serviceQueryStat
dataType: boolean
status: current
description: Specifies whether or
      not permission is granted to query the service control
      manager about the status of the service.
```

7.396. serviceEnumDependents

```
elementId: TBD
name: serviceEnumDependents
dataType: boolean
status: current
description: Specifies whether
      or not permission is granted to query for an enumeration of
      all the services dependent on the service.
```

7.397. serviceStart

```
elementId: TBD
name: serviceStart
dataType: boolean
status: current
description: Specifies whether or not
      permission is granted to start the service.
```

7.398. serviceStop

```
elementId: TBD
name: serviceStop
dataType: boolean
status: current
description: Specifies whether or not
      permission is granted to stop the service.
```

7.399. servicePause

```
elementId: TBD
name: servicePause
dataType: boolean
status: current
description: Specifies whether or not
      permission is granted to pause or continue the service.
```

7.400. serviceInterrogate

```
elementId: TBD
name: serviceInterrogate
dataType: boolean
status: current
description: Specifies whether or not permission is granted to
             request the service to report its status immediately.
```

7.401. serviceUserDefined

```
elementId: TBD
name: serviceUserDefined
dataType: boolean
status: current
description: Specifies whether or
      not permission is granted to specify a user-defined
      control code.
```

7.402. sharedresourceauditedpermissions

```
elementId: TBD
name: sharedresourceauditedpermissions
dataType: list
structure: list (netname, trusteeSid,
      standardDelete, standardReadControl, standardWriteDac,
      standardWriteOwner, standardSynchronize,
      accessSystemSecurity, genericRead, genericWrite,
      genericExecute, genericAll)
status: current
description: Stores
      the audited access rights of a shared resource that a system
      access control list (SACL) structure grants to a
      specified trustee. The trustee's audited access rights are
      determined checking all access control entries (ACEs)
      in the SACL.
```

7.403. netname

```
elementId: TBD
name: netname
dataType: string
status: current
description: Specifies the name associated
      with a particular shared resource.
```

7.404. sharedresourceeffectiverights

```
elementId: TBD
name: sharedresourceeffectiverights
dataType: list
structure: list (netname, trusteeSid,
      standardDelete, standardReadControl, standardWriteDac,
      standardWriteOwner, standardSynchronize,
      accessSystemSecurity, genericRead, genericWrite,
      genericExecute, genericAll)
status: current
description: Stores
      the effective rights of a shared resource that a
      discretionary access control list (DACL) structure grants
      to a specified trustee. The trustee's effective rights are
      determined checking all access-allowed and access-denied
      access control entries (ACEs) in the DACL.
```

7.405. user

```
elementId: TBD
name: user
dataType: list
structure: list (username, enabled, group, lastLogon)
status: current
description: Specifies the groups to which a user belongs.
```

7.406. enabled

```
elementId: TBD
name: enabled
dataType: boolean
status: current
description: Represents whether the
      particular user is enabled or not.
```

7.407. lastLogon

```
elementId: TBD
name: lastLogon
dataType: integer
status: current
description: The date and time when the
      last logon occurred.
```

7.408. groupSid

```
elementId: TBD
name: groupSid
dataType: string
status: current
description: Represents the SID of a
      particular group. If the specified user belongs to more than
      one group, then multiple groupSid elements are
      applicable. If the specified user is not a member of a single
      group, then a single groupSid element should be
      incldued with a status of 'does not exist'. If there is an
      error determining the groups that the user belongs to,
      then a single groupSid element should be included with a
      status of 'error'.
```

8. Acknowledgements

Many of the specifications in this document have been developed in a public-private partnership with vendors and end-users. The hard work

of the SCAP community is appreciated in advancing these efforts to their current level of adoption.

Over the course of developing the initial draft, Brant Cheikes, Matt Hansbury, Daniel Haynes, Scott Pope, Charles Schmidt, and Steve Venema have contributed text to many sections of this document.

9. IANA Considerations

This document specifies an initial set of Information Elements for SACM in Section 7. An Internet Assigned Numbers Authority (IANA) registry will be created and populated with the Information Elements in Section 7. New assignments for SACM Information Elements will be administered by IANA through Expert Review [RFC2434]. The designated experts MUST check the requested Information Elements for completeness and accuracy of the submission with respect to the template and requirements expressed in Section 4 and Section 4.1. Requests for Information Elements that duplicate the functionality of existing Information Elements SHOULD be declined. The smallest available Information Element identifier SHOULD be assigned to a new Information Element. The definition of new Information Elements MUST be published using a well-established and persistent publication medium.

10. Security Considerations

Posture Assessments need to be performed in a safe and secure manner. In that regard, there are multiple aspects of security that apply to the communications between components as well as the capabilities themselves. This information model only contains an initial listing of items that need to be considered with respect to security and will need to be augmented as the model continues to be developed.

Security considerations include:

Authentication: Every SACM Component and asset needs to be able to identify itself and verify the identity of other SACM Components and assets.

Confidentiality: Communications between SACM Components need to be protected from eavesdropping or unauthorized collection. Some communications between SACM Components and assets may need to be protected as well.

Integrity: The information exchanged between SACM Components needs to be protected from modification. Some exchanges between assets and SACM Components will also have this requirement.

Restricted Access: Access to the information collected, evaluated, reported, and stored should only be viewable and consumable to authenticated and authorized entities.

Considerations with respect to the operational aspects of collection, evaluation, and storage security automation information can be found in Section 11.

Considerations concerning the privacy of security automation information can be found in Section 12.

11. Operational Considerations

The following sections outline a series of operational considerations for SACM deployments within an organization. This section may be expanded to include other considerations as the WG gains additional operational experience with SACM deployments and extending the information model.

11.1. Endpoint Designation

In order to successfully carry out endpoint posture assessment, it is necessary to be able to identify the endpoints on a network and track the changes to them over time. Specifically, enabling SACM Components to:

- Tell whether two endpoint attribute assertions concern the same endpoint

- Respond to compliance measurements, for example by reporting, remediating, and quarantining (SACM does not specify these responses, but SACM exists to enable them).

Ideally, every endpoint would be identified by a unique identifier present on the endpoint, but, this is complicated due to different factors such as the variety of endpoints on a network, the ability of tools to reliably access such an identifer, and the ability of tools to correlate disparate identifiers. As a result, it is necessary for an endpoint to be identified by a set of attributes that uniquely identify it on a network. The set of attributes that uniquely identify an endpoint on a network will likely vary by organization; however, there are a number of properties to consider when selecting identifying attributes as some are better suited for identification purposes than others.

Multiplicity: Is the attribute typically associated with a single endpoint or with multiple endpoints? If the attribute is

associated with a single endpoint, it is better for identifying an endpoint on a network.

Persistence: How likely is the attribute to change? Does it never change? Does it only change when the endpoint is reprovisioned? Does it only change due to an event? Does it change on an ad-hoc and often unpredictable basis? Does it constantly change? The less likely it is for an attribute to change over time, the better it is for identifying an endpoint on a network.

Immutability: How difficult is it to change the attribute? Is the attribute hardware rooted and never changes? Can the attribute be changed by a user/process with the appropriate access? Can the attribute be changed without controlled access. The less likely an attribute is to change over time, the better chance it will be usable to identify an endpoint over time.

Verifiable: Can the attribute be corroborated? Can the attribute be externally verified with source authentication? Can the attribute be externally verified without source authentication? Is it impossible to externally verify the attribute. Attributes that can be externally verified are more likely to be accurate and are better for identifying endpoints on a network.

With that said, requiring SACM Components and end users to constantly refer to a set of attributes to identify an endpoint, is particularly burdensome. As a result, SACM supports the concept of a target endpoint label which associates an identifier (unique to a SACM domain) with the set of attributes used by an organization to identify endpoints on a network. Once defined for an endpoint, the target endpoint label can be used in place of the set of identifying attributes.

11.2. Timestamp Accuracy

An organization will likely have different collectors deployed across the network that will be configured to collect posture attributes on varying frequencies (periodic, ad-hoc, event-driven, on endpoint, off endpoint, etc.). Some collectors will detect changes as soon as they occur whereas others will detect them at a later point during a periodic scan or when an event has triggered the collection of posture attributes. Furthermore, some changes will be detected on the endpoint and others will be observed off of the endpoint. As a result of these differences, the accuracy of the timestamp associated with the collected information will vary. For example, if a

collector is only running once every 12 hours, the change probably happened at some point in time prior to the scan and the timestamp is likely not accurate. Due to this, it is important for system administrators to determine if the accuracy of a timestamp is good enough for their intended purposes.

12. Privacy Considerations

In the IETF, there are privacy concerns with respect to endpoint identity and monitoring. This is especially true when the activity on an endpoint can be linked to a particular person. For example, by correlating endpoint attributes such as usernames, certificates, etc. with browser activity, it may be possible to gain insight in to user behavior and trends beyond what is required to carry out endpoint posture assessments. In the hands of the wrong person, this information could be used to negatively influence a user's behavior or to plan attacks against the organization's infrastructure.

As a result, SACM data models should incorporate a mechanism by which an organization can designate which endpoint attributes are considered sensitive with respect to privacy. This will allow SACM Components to handle endpoint attributes in a manner consistent with the organization's privacy policies. Furthermore, organization's should put the proper mechanism in place to ensure endpoint attributes are protected when transmitted, stored, and accessed to ensure only authorized parties are granted access.

It should also be noted that some of this is often mitigated by organizational policies that require a user of an organization's network to consent to some level of monitoring in return for access to the network and other resources. The information that is monitored and collected will vary by organization and further highlights the need for a mechanism by which an organization can specify what constitutes privacy sensitive information for them.

13. References

13.1. Normative References

[PEN] Internet Assigned Numbers Authority, "Private Enterprise Numbers", July 2016, <https://www.iana.org/assignments/enterprise-numbers/enterprise-numbers>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

13.2. Informative References

[I-D.ietf-sacm-requirements]
Cam-Winget, N. and L. Lorenzin, "Secure Automation and Continuous Monitoring (SACM) Requirements", draft-ietf-sacm-requirements-01 (work in progress), October 2014.

[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.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 2434, DOI 10.17487/RFC2434, October 1998, <http://www.rfc-editor.org/info/rfc2434>.

[RFC3580] Congdon, P., Aboba, B., Smith, A., Zorn, G., and J. Roese, "IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines", RFC 3580, DOI 10.17487/RFC3580, September 2003, <http://www.rfc-editor.org/info/rfc3580>.

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2", FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007, <http://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, <http://www.rfc-editor.org/info/rfc5209>.

[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, <http://www.rfc-editor.org/info/rfc5793>.

[RFC7012] Claise, B., Ed. and B. Trammell, Ed., "Information Model for IP Flow Information Export (IPFIX)", RFC 7012, DOI 10.17487/RFC7012, September 2013, <http://www.rfc-editor.org/info/rfc7012>.

[RFC7632] Waltermire, D. and D. Harrington, "Endpoint Security Posture Assessment: Enterprise Use Cases", RFC 7632, DOI 10.17487/RFC7632, September 2015, <http://www.rfc-editor.org/info/rfc7632>.

Appendix A. Change Log

A.1. Changes in Revision 01

Added some proposed normative text.

For provenance:

Added a class "Method"

Added the produced-using relationship between an AVP and a method

Added the produced-by relationship between a Guidance and a SACM Component

Added the hosted-by relationship between a SACM Component and an Endpoint

asserted-by and summarized-by have been renamed to produced-by.

"User" is now "Account". If a user has different credentials, SACM cannot know that they belong to the same user. But, per Kim W, many organizations do have accounts that associate credentials.

The multiplicity of the based-on relationships has been corrected.

More relationships now have labels, per UML convention.

The diagram no longer has causal arrow. They had become redundant and were nonstandard and clutter.

Renamed "credential" to "identity", following industry usage. A credential includes proof, such as a key or password. A username or a distinguished name is called an "identity".

Removed Session, because an endpoint's network activity is not SACM's initial focus

Removed Authorization, for the same reason

Added many-to-many relationship between Hardware Component and Endpoint, for clarity

Added many-to-many relationship between Software Component and Endpoint, for clarity

Added "contains" relationship between Network Interface and Network Interface

Removed relationship between Network Interface and Account. The endpoint knows the identity it used to gain network access. The PDP also knows that. But they probably do not know the account.

Added relationship between Network Interface and Identity. The endpoint and the PDP will typically know the identity.

Made identity-to-account a many-to-one relationship.

A.2. Changes in Revision 02

Added Section Identifying Attributes.

Split the figure into Figure Model of Endpoint and Figure Information Elements.

Added Figure Information Elements Take 2, proposing a triple-store model.

Some editorial cleanup

A.3. Changes in Revision 03

Moved Appendix A.1, Appendix A.2, and Mapping to SACM Use Cases into the Appendix. Added a reference to it in Section 1

Added the Section 4 section. Provided notes for the type of information we need to add in this section.

Added the Section 6 section. Moved sections on Endpoint, Hardware Component, Software Component, Hardware Instance, and Software Instance there. Provided notes for the type of information we need to add in this section.

Removed the Provenance of Information Section. SACM is not going to solve provenance rather give organizations enough information to figure it out.

Updated references to the Endpoint Security Posture Assessment: Enterprise Use Cases document to reflect that it was published as an RFC.

Fixed the formatting of a few figures.

Included references to [RFC3580] where RADIUS is mentioned.

A.4. Changes in Revision 04

Integrated the IPFIX [RFC7012] syntax into Section 4.

Converted many of the existing SACM Information Elements to the IPFIX syntax.

Included existing IPFIX Information Elements and datatypes that could likely be reused for SACM in Section 7 and Section 4 respectively.

Removed the sections related to reports as described in https://github.com/sacmwg/draft-ietf-sacm-information-model/issues/30.

Cleaned up other text throughout the document.

A.5. Changes in Revision 05

Merged proposed changes from the I-D IM into the WG IM (https://github.com/sacmwg/draft-ietf-sacm-information-model/issues/41).

Fixed some formatting warnings.

Removed a duplicate IE and added a few IE datatypes that were missing.

A.6. Changes in Revision 06

Clarified that the SACM statement and content-element subjects are conceptual and that they do not need to be explicitly defined in a data model as long as the necessary information is provided.

Updated the IPFIX syntax used to define Information Elements. There are still a couple of open issues that need to be resolved.

Updated some of the Information Elements contained in Section 7 to use the revised IPFIX syntax. The rest of the Information Elements will be converted in a later revision.

Performed various clean-up and refactoring in Sections 6 and 7. Still need to go through Section 8.

Removed appendices that were not referenced in the body of the draft. The text from them is still available in previous revisions of this document if needed.

A.7. Changes in Revision 07

Made various changes to the IPFIX syntax based on discussions at the IETF 96 Meeting. Changes included the addition of a structure property to the IE specification template, the creation of an enumeration datatype, and the specification of an IE naming convention.

Provided text to define Collection Guidance, Evaluation Guidance, Classification Guidance, Storage Guidance, and Evaluation Results.

Included additional IEs related to software, configuration, and the vulnerability assessment scenario.

Added text for the IANA considerations, security considerations, operational considerations, and privacy considerations sections.

Performed various other editorial changes and clean-up.

Authors' Addresses

David Waltermire (editor)
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, Maryland 20877
USA

Email: david.waltermire@nist.gov

Kim Watson
United States Department of Homeland Security
DHS/CS&C/FNR
245 Murray Ln. SW, Bldg 410
MS0613
Washington, DC 20528
USA

Email: kimberly.watson@hq.dhs.gov

Clifford Kahn
Pulse Secure, LLC
2700 Zanker Road, Suite 200
San Jose, CA 95134
USA

Email: cliffordk@pulsesecure.net

Lisa Lorenzin
Pulse Secure, LLC
2700 Zanker Road, Suite 200
San Jose, CA 95134
USA

Email: llorenzin@pulsesecure.net

Michael Cokus
The MITRE Corporation
903 Enterprise Parkway, Suite 200
Hampton, VA 23666
USA

Email: msc@mitre.org

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

Email: dhaynes@mitre.org

Henk Birkholz
Fraunhofer SIT
Rheinstrasse 75
Darmstadt 64295
Germany

Email: henk.birkholz@sit.fraunhofer.de