- Open Vulnerability and Assessment Language - Element Dictionary

- Schema: Core System Characteristics
- Version: 4.2
- Release Date: 2 December 2005

The following is a description of the elements, types, and attributes that compose the core schema for encoding Open Vulnerability and Assessment Language (OVAL) System Characteristics. The Core System Characteristics Schema defines all operating system independent objects. These objects are extended and enhanced by individual family schemas, which are described in separate documents. Each of the elements, types, and attributes that make up the Core System Characteristics Schema are described in detail and should provide the information necessary to understand what each object represents. This document is intended for developers and assumes some familiarity with XML. A high level description of the interaction between these objects is not outlined here.

The OVAL Schema is maintained by The Mitre Corporation and developed by the public OVAL Community. For more information, including how to get involved in the project and how to submit change requests, please visit the OVAL website at http://oval.mitre.org.

Elements

This section describes all the elements that are found within the schema, starting with the root element. Note that in the tables outlining possible attributes and child elements, square brackets [] means that the item is optional. All complex and simple types, along with attribute groups are described later in this document.

<system_characteristics>

The system_characteristics element is the root of an OVAL System Characteristics Document, and must occur exactly once. Its purpose is to bind together the three major sections of a system characteristics file - generator, system_info, and system_data - which are the children of the system_characteristics element. The generator section must be present and provides information about when the system characteristics file was compiled and under what version. The require system_info element is used to record information about the system being described. The optional system_data section defines the specific characteristics that have been collected from the system.

The optional Signature element allows an XML Signature as defined by the W3C to be attached to the document. This allows authentication and data integrety to be provided to the user. Enveloped signatures are supported.

Cardinality:	1
Attributes:	family

Content:	none
Parent Elements:	none
Child Elements:	generator, system_info, [system_data], [Signature]

<generator>

The generator element is used to format information about what application generated a particular file. This type is used to specify the source of the OVAL Definitions, the application used for the analysis, and optionally the application used for data collection.

Cardinality:	1
Attributes:	none
Content:	none
Parent Elements:	system_characteristics
Child Elements:	product_name, product_version, schema_version, timestamp

<product_name>

The name of the application used to generate this file of OVAL definitions.

Cardinality:	0-1
Attributes:	none
Content:	string
Parent Elements:	generator
Child Elements:	none

<product_version>

The version of the product used to generate this file of OVAL definitions

Cardinality:	0-1
Attributes:	none
Content:	string
Parent Elements:	generator
Child Elements:	none

<schema_version>

This element defines the version of the OVAL schema that the document has been validated against.

Cardinality:	1
Attributes:	none
Content:	decimal
Parent Elements:	generator
Child Elements:	none

<timestamp>

This element specifies the date/time at which the document was created. This timestamp can be used to differentiate between multiple OVAL files and to determine which document is the most up-to-date. The timestamp is of the form yyyymmddhhmmss.

Cardinality:	1
Attributes:	none
Content:	string
Parent Elements:	generator
Child Elements:	none

<system_info>

The system_info element specifies general information about the system that data was collected from including information that can be used to identify the system.

Cardinality:	1
Attributes:	none
Content:	string
Parent Elements:	system_characteristics
Child Elements:	os_name, os_version, architecture, primary_host_name, interfaces

<os_name>

The operating system of the machine the data was collected from.

Cardinality:	1
Attributes:	none
Content:	string
Parent Elements:	system_info
Child Elements:	none

<os_version>

The version of the operating system of the machine the data was collected from.

Cardinality:	1
Attributes:	none
Content:	string
Parent Elements:	system_info
Child Elements:	none

<architecture>

The architecture of the machine the data was collected from.

Cardinality:	1
Attributes:	none
Content:	string
Parent Elements:	system_info
Child Elements:	none

<primary_host_name>

The primary host name of the machine the data was collected from.

Cardinality:	1
Attributes:	none
Content:	string
Parent Elements:	system_info
Child Elements:	none

<interfaces>

The interfaces element holds a collection of interface elements that describe each interface on the machine that data was collected from.

Cardinality:	1
Attributes:	none
Content:	string
Parent Elements:	system_info
Child Elements:	interface

<interface>

The interface element is used to describe an interface on a system.

Cardinality:	1-n
Attributes:	none
Content:	string
Parent Elements:	interfaces
Child Elements:	interface_name, ip_address, mac_address

<interface_name>

The name of the interface

Cardinality:	1
Attributes:	none
Content:	string
Parent Elements:	interface
Child Elements:	none

<ip_address>

The ip address of the interface

Cardinality:	1
Attributes:	none
Content:	string
Parent Elements:	interface
Child Elements:	none

<mac_address>

The mac address of the interface

Cardinality:	1
Attributes:	none
Content:	string
Parent Elements:	interface
Child Elements:	none

<system_data>

The system_data element is a container for one or more item_container elements.

Cardinality:	0-1
Attributes:	none
Content:	none
Parent Elements:	system_characteristics
Child Elements:	item_container

<item_container>

The item_container abstract element is extended (via substitution groups) by the different containers found in the family schemas. Each item_container should contain a number of abstract item elements.

Cardinality:	1-n
Attributes:	none
Content:	none
Parent Elements:	system_data
Child Elements:	item

<item>

The abstract item element is a container for information about an item on a system. An item might be a file, a rpm, a process, etc. This element is extended by the different families through substitution groups. Each item represents a unique object as specified by the object section of the item. There are cases where the object declaration results in many different instances, all described by the unique object section. An example of this would be in the wmi_item where a wql query my return multiple instances. Each instance

would be represented by a different item in the system characterics file, yet each item would have the same object section.

Cardinality:	1-n
Attributes:	id
Content:	none
Parent Elements:	item_container
Child Elements:	(specified through extension)

<message>

The message element is used to store helpful messages reported during data collection on a system.

Cardinality:	0-1
Attributes:	none
Content:	string
Parent Elements:	item
Child Elements:	none

Complex Types

This section describes any global complex types defined in the schema. These types can be instantiated by elements in this schema as well as elements in other schemas. Note that in the tables outlining possible attributes and child elements, square brackets [] means that the item is optional.

-- itemType --

The itemType specifies an optional message element as well as the id attribute.

Attributes:	id
Content:	none
Child Elements:	[message]

-- objectType --

The object type specifies an element that shall be used as a container for an object on a system. Examples of objects are: a file path, a registry key, a user name. If the specified object is not found on a system the

optional status attribute is set to 'does not exist'.

Attributes:	[status]
Content:	none
Child Elements:	none

-- dataType --

The data type specifies an element that shall be used as a container for a data related to an object on a system. Examples of data are: a file version, a registry key value, a user's rights.

Attributes:	none
Content:	none
Child Elements:	none

-- objectIntType --

Describes an integer element in an object section of an item and specified its attributes.

Attributes:	(includes objectAttributes)
Content:	integer
Child Elements:	none

-- objectStringType --

Describes a string element in an object section of an item and specified its attributes.

Attributes:	(includes objectAttributes)
Content:	string
Child Elements:	none

-- objectBaseType --

Describes complex data along with the standard object element attributes.

Attributes:	(includes objectAttributes)
Content:	(anyType)
Child Elements:	(anyType)

-- dataBoolType --

Describes simple boolean data along with the standard data element attributes. The empty string is listed to allow and element of this type to be empty. This might occur when an error is reported on that element.

Attributes:	(includes dataAttributes)
Content:	boolean
Child Elements:	none

-- dataIntType --

Describes simple integer data along with the standard data element attributes.

Attributes:	(includes dataAttributes)
Content:	integer
Child Elements:	none

-- dataStringType --

Describes simple string data along with the standard data element attributes.

Attributes:	(includes dataAttributes)
Content:	string
Child Elements:	none

-- dataBaseType --

Describes complex data along with the standard data element attributes.

Attributes:	(includes dataAttributes)
Content:	(anyType)
Child Elements:	(anyType)

Attribute Groups

This section describes any global attribute groups defined in the schema. An attribute group can be included by various types providing a standard set of attributes across each of the types. Note that in the tables outlining possible attributes, square brackets [] means that the item is optional.

-- objectAttributes --

The possible attributes for an element in the object section of an item. The optional datatype attribute can be either 'literal' or 'pattern match'. A value of pattern match is for reporting information about the collection of objects associated with a specific pattern. Helps answers the question 'How do I know that everything that matches this pattern is in the SC file?' The pattern would be presented and the status would reflect the data collection as a whole. Then each matching object would exists on their own with a datatype of literal.

Attributes:	[datatype]

-- dataAttributes --

The possible attributes for an element in the data section of an item. The optional datatype determines the type of data expected. (the default datatype is 'string') The datatype attribute is optional in order to keep the XML clean and readable. The default value is used most of the time and putting datatype="string" for each element would muddy up the XML. The optional status attribute holds information regarding the success of the data collection. For example, if there was an error collecting a particular piece of data, then the status would be 'error'.

Attributes:	[datatype], [status]

Simple Types

This section describes any global simple type defined in the schema. A simple type is a restriction of one of the base types (string, int, etc.) and allows a valid entry to be limited to a specific subset of values.

---

dataDatatypes values

This simple type defines the legal data types that are used to describe data sub elements in items.

The component datatype represents a value that is built from one or more component strings. Each component can be a literal string or can reference a value held elsewhere, say a registry key.

The version datatype represents a value that is a hierarchical list of versions. For example '#.#.#' or '#-#-#-#' where the numbers to the left are 'more significant' than the numbers to the right.

-- binary
-- boolean
-- component
-- float
-- int
-- string
-- version

families values

The families simple type is a listing of families that OVAL supports at this time.

```
-- aix
-- apache
-- debian
-- freebsd
-- hp-ux
-- ios
-- macos
-- openbsd
-- oracle
-- os400
-- pix
-- redhat
-- solaris
-- suse
-- windows
```

itemId values

Define acceptable item ids as an integer.

objectDatatypes values

Define acceptable data types for an element in an object section.

```
-- literal
-- pattern match
```

statusType values

```
-- error
-- exists
-- does not exist
-- not collected
```

timeStamp values

Define acceptable timestamps as a string with the form yyyymmddhhmmss.

```
-- a value satisfying the pattern '\d{14}'
```