Security Automation Developer Days

OPEN VULNERABILITY AND ASSESSMENT LANGUAGE

OVAL®

June 9-10, 2009

MITRE

OVAL Session Overview

- Tuesday
  - Deprecation Policy Review
  - Schematron Usage in OVAL
  - Element name reconciliation
  - xsd:choice structure on objects
- Wednesday
  - Supporting N-Tuples in OVAL
  - Pattern match on enumerations
  - Tests reference multiple states
  - Introduce PCRE based pattern matches
  - Emerging Use Case: “OVAL for System Inventory?”

Deprecation Policy Review

Deprecation Policy Review

- Prior to April 2009, OVAL lacked a formal deprecation policy
  - Removal or modification of OVAL constructs were executed without an adequately defined workflow
  - This was seen as a problem for the maturing language
  - After looking at other well-known, open source projects it was realized that a deprecation policy had to be developed

Deprecation Policy Review

- Version 1.0 of the OVAL Deprecation Policy was developed in April of 2009
- It stated the following:
  - OVAL constructs will be deprecated for security issues, language consistency issues, or if a construct becomes obsolete due to new technologies or methodologies.

Deprecation Policy Defined

- All existing constructs must go through a deprecation phase prior to being removed.
- The duration of deprecation phases will be in terms of releases.
- Language constructs will remain in a deprecated state for at least one release. During this time deprecated constructs will be flagged using a machine-readable flag.

MITRE

Deprecation Policy Defined (2)

- When using a deprecated feature, Schematron validation will report a warning.
- Prior to a release, deprecated and removed constructs will be announced via email and posted on the OVAL Web site.

Deprecation Process

- Construct is nominated for deprecation via email to the OVAL Developer List.

- Discussion then deprecation (maybe).

- Deprecation for at least one minor release, then removal.

Deprecation Implementation

```
<xsd:element name="fileauditedpermissions_test" substitutionGroup="oval-def:test">
    <xsd:annotation> <!-- annotations --> </xsd:annotation>
    <xsd:appinfo>
        <oval:deprecated_info>
            <oval:version>5.5</oval:version>
            <oval:reason>Replaced by filesaudtiedpermissions_better_test</oval:reason>
            <oval:comment>Did not align with Win32 API</oval:comment>
        </oval:deprecated_info>
        <sch:pattern id="foo_pattern">
            <sch:rule context="win-def:fileauditedpermissions_test">
                <sch:report test=“.”>DEPRECATED ELEMENT: <sch:value-of select="name()"/></sch:report>
            </sch:rule>
        </sch:pattern>
    </xsd:appinfo>
    <!-- element definition -->
</xsd:element>
```

Schematron Usage In OVAL

Schematron Usage In OVAL

- Schematron has been utilized within OVAL since the release of 5.0
- Schematron validation is optional within OVAL
- This discussion will:
  - Briefly review Schematron
  - Explain its usage in OVAL
  - Discuss its future in OVAL

MITRE

Schematron Usage In OVAL

- Overview
  - Schematron is a complimentary validation mechanism to XML Schema validation
  - Uses XPath expressions to define constraints and relationships within an XML Schema
  - Can express both warnings as well as errors during validation

Schematron Usages In OVAL

- Why do we use it?
  - Express constraints that cannot be described in XML Schema
    - For example: limiting an attributes value to a subset of an enumeration
  - Co-constraints
    - For example: if a test has a check_existence value of ‘none_exist’ then a state cannot be referenced
  - Reporting warnings for deprecated schema constructs

Schematron Usages In OVAL

- Problems
  - Validation can be **very slow**
    - Documents > 2 MB in size can take minutes to validate
  - Schematron supports XSLT2 and XPath 2.0
    - Could be problematic for non-Java Developers

Schematron Usages In OVAL

- Where do we go from here with Schematron?
  - Required validation for OVAL content?
    - Only certain classes of OVAL content?
  - Keep it optional?

Element Name Reconciliation

Element Name Reconciliation (1)

- In naming tests we have attempted to:
  - make element names as intuitive as possible
  - reduce schema bloat as much as possible
    - introduce new elements only when absolutely necessary
  - utilize consistent naming patterns
    - test, object, state, and item names align

Element Name Reconciliation (2)

- As the language evolves our guidelines for naming elements begin to contradict each other
  - Element name typos
    - Fixing the typo adds to bloat but improves readability (<inetlisteningservers_test/> uses a <inetlisteningserver_item/>)
  - New test versions
    - Utilizing existing state or item reduces bloat but reduces readability (<patch54_test/> uses a <patch_state/>)

Element Name Reconciliation (3)

- Over time the element names for tests, objects, states, and items have diverged
  - typos, new versions of tests using old items

- Proposal
  - Bring all the test, object, state, and item names into alignment
    - Deprecating old items.
  - Establish the convention that all names will align
    - ensure the names do not diverge again
    - automate name alignment checking

Element Name Reconciliation (4)

- Does this change fit into version 5.6? Is this change worthwhile?
  - Impact of change:
    - Introduces several new tests/objects/states/items
    - Deprecates all tests/objects/states/items that are not in alignment
    - Does not invalidate existing content
    - Adds schema bloat
  - Benefit of change:
    - Ensures that a constant naming pattern will be followed for all future changes
    - Simplifies some implementations (no need for a mapping)
    - Begins the process of removing inconsistent element names

Choice Structure on Objects

Choice Structure on Objects (1)

- Add a xsd:choice structure to objects to allow for more flexibility when declaring an object.
  - filepath vs. path + filename
  - SID vs. trustee name
- Discussed at 2008 OVAL Developer Days for version 6.
  - Consensus was that this flexibility was desirable
  - Further refined on the oval-developer-list:

    http://oval.mitre.org/community/archives.html#nabble-td1485589

Choice Structure on Objects (2)

Current windows file_object instance:

```
<file_object id="oval:sample:obj:1" version="1" xmlns=“…">
   <path>c:\windows</path>
   <filename>foo.exe</filename>
</file_object>
```

Proposed windows file_object instances:

```
<file_object id="oval:sample:obj:1" version="1" xmlns=“…">
   <path>c:\windows</path>
   <filename>foo.exe</filename>
</file_object>
```

OR

```
<file_object id="oval:sample:obj:2" version="1" xmlns=“…">
   <filepath>c:\windows\foo.exe</filepath>
</file_object>
```

Choice Structure on Objects (3)

Current windows file_object schema declaration:

```
<xsd:sequence>
   <xsd:element name="behaviors" type="win-def:FileBehaviors" minOccurs="0"/>
   <xsd:element name="path" type="oval-def:EntityObjectStringType“/>
   <xsd:element name="filename" type="oval-def:EntityObjectStringType" nillable="true"/>
</xsd:sequence>
```

Proposed windows file_object schema declaration:

```
<xsd:sequence>
   <xsd:element name="behaviors" type="win-def:FileBehaviors" minOccurs="0"/>
   <xsd:choice>
      <xsd:sequence>
         <xsd:element name="path" type="oval-def:EntityObjectStringType"/>
         <xsd:element name="filename" type="oval-def:EntityObjectStringType" nillable="true"/>
      </xsd:sequence>
      <xsd:sequence>
         <xsd:element name="filepath" type="oval-def:EntityObjectStringType"/>
      </xsd:sequence>
   </xsd:choice>
</xsd:sequence>
```

Choice Structure on Objects (4)

- Does this change fit into version 5.6? Is this change too big for a minor version?
  - Impact of change:
    - Introduces a new structure to several objects
      - New concept to learn
      - New concept to implement
    - Does not invalidate existing content
  - Benefit of change:
    - Enables file checking that currently cannot be done
    - Ensures standard meaning for all content

Supporting N-Tuples in OVAL

Supporting N-Tuples in OVAL (1)

- Several data repositories (WMI, XML, SQL) that OVAL supports querying can return results sets as n-tuples.
  - WQL – SELECT Name, ScreenSaverTimeOut FROM Win32_Desktop;
- OVAL currently only supports result sets with single values.
  - WQL – SELECT Name FROM Win32_Desktop;
- This discussion will:
  - review the deficiency in OVAL 5.5
  - review a proposal for addressing the issue
  - discuss the priority of addressing the issue

Supporting N-Tuples in OVAL (2)

- WQL – SELECT Name FROM Win32_Desktop;

- Current win-def:wmi_state

```
<wmi_state id="oval:sample:ste:1" version="1" xmlns="…">
   <result datatype="string" operation="equals" >user2</result>
</wmi_state>
```

- Current win-sc:wmi_item

```
<wmi_item id="1" xmlns="…">
   <namespace>root\CIMV2</namespace>
   <wql>SELECT Name FROM Win32_Desktop</wql>
   <result>user2</result>
   <result>user1</result>
</wmi_item>
```

MITRE

Supporting N-Tuples in OVAL (3)

WQL - SELECT Name, ScreenSaverTimeOut FROM Win32_Desktop;

```
<wmi_state id="oval:sample:ste:2" operator="AND" version="1" xmlns="…">
   <result datatype="record" operation="equals" >
      <field name= "Name" datatype="string" operation="equals">user2</field>
      <field name= "ScreenSaverTimeOut" datatype="int" operation="less than">600</field>
   </result>
</wmi_state>
```

- Current result element remains
- Introduce new ‘record’ datatype
  - allows mixed content
  - defines a field element
- Field elements have:
  - @name must be unique
  - support @datatype and @operation
  - include @var_ref, @var_check, and @entity_check

Considerations

- keeps the result entity closely aligned with others
- leaves several unneeded/unused attributes
- attribute, not element name, distinguishing contents
- Changes nature of current element

MITRE

Supporting N-Tuples in OVAL (4)

WQL - SELECT Name, ScreenSaverTimeOut FROM Win32_Desktop;

```
<wmi_state id="oval:sample:ste:2" operator="AND" version="1" xmlns=“…“>
   <result datatype="string" operation=“equals" >user2</result>
   <resultset entity_check="all">
      <field name= "Name" datatype="string" operation="equals">user2</field>
      <field name= "ScreenSaverTimeOut" datatype="int" operation="less than">600</field>
   </resultset>
</wmi_state>
```

- Current result element remains optional
- Introduce new resultset element
  - has child field elements
  - @entity_check
- Field elements have:
  - @name must be unique
  - support @datatype and @operation
  - include @var_ref, @var_check, and @entity_check

Considerations

- resultset is not like any other entity
- this structure would be used elsewhere
- handling of unnamed fields

MITRE

Supporting N-Tuples in OVAL (5)

WQL - SELECT Name, ScreenSaverTimeOut FROM Win32_Desktop;

```
<wmi_state id="oval:sample:ste:2" operator="AND" version="1" xmlns=“…“>
   <result datatype="string" operation=“equals" >user2</result>
   <result_1 datatype="string" operation="equals" >user2</result>
   <result_2 datatype="int" operation="equals" >333</result>
</wmi_state>
```

- Current result element remains optional and unchanged
- Introduce several new sequentially named result elements

Considerations

- addresses some cases
- leaves a lot to be desired

MITRE

Supporting N-Tuples in OVAL (6)

- Are there other options that should be considered?

Supporting N-Tuples in OVAL (7)

- When can this change be made?
  - Impact of change:
    - Introduces a new structure (diverges for a consistent pattern)
    - Does not invalidate existing content
    - Isolated to WMI, SQL, XML, Active Directory related tests
  - Benefit of change:
    - Allows for increased adoption of OVAL by configuration guidance authors
    - Ensures OVAL will support WMI and XML as we increasingly need to query them
    - Improves support in OVAL for databases
- Is this a minor or major revision?

Pattern Match on Enumerations

Pattern Match on Enumerations (1)

- OVAL uses xsd:enumerations to define allowed values for many system constructs.
- Without these enumerations content naturally diverges.
  - HKEY_LOCAL_MACHINE vs. HKLM
  - AUDIT_FAILURE vs. FAILURE
- Need consistency to ensure tool interoperability and increase content readability.

Pattern Match on Enumerations (2)

- xsd:enumerations prevent using pattern matches on enumerated values

```
<auditeventpolicy_state id="oval:sample:ste:1" version="1" xmlns="…">
   <account_logon datatype="string"
                  operation="pattern match">AUDIT_(SUCCESS|SUCCESS_FAILURE)</account_logon>
</auditeventpolicy_state>
```

- Only allowed values are:
  - "AUDIT_FAILURE“, "AUDIT_NONE", "AUDIT_SUCCESS", "AUDIT_SUCCESS_FAILURE“

- Lack of support for pattern matches is considered a deficiency
  - intent is to support pattern matches, but restricting possible values has been considered more important

Pattern Match on Enumerations (3)

- Is there a workaround?

- Refer to variable for the value:

```
<auditeventpolicy_state id="oval:sample:ste:1" version="1" xmlns="...">
    <account_logon datatype="string" operation="pattern match" var_ref="oval:sample:var:1"/>
</auditeventpolicy_state>
```

- Declare the regular expression in a variable:

```
<constant_variable id="oval:sample:var:1" version="1" comment="..." datatype="string">
    <value>AUDIT_(SUCCESS|SUCCESS_FAILURE)</value>
</constant_variable>
```

Pattern Match on Enumerations (4)

- Schematron rules were developed to restrict allowed operations to just ‘equals’ and ‘not equal’
  - a pattern match on a restricted set of strings does not make sense
  - Version 5.3 has Schematron rules to prevent using the pattern match operation on most enumerations.
- Schematron rules were refactored in version 5.4
  - inadvertently dropped the rules for restricting the use of the pattern match operation
  - opened the door to a workaround???

Pattern Match on Enumerations (5)

- Moving forward is this a feature that should stay?
  - Do we add the rules back for version 5.6?
    - Prevent pattern matches until some other solution can be found
  - Do we utilize this as an opportunity to close a long standing feature request?

Tests Reference Multiple States

OPEN VULNERABILITY AND ASSESSMENT LANGUAGE

OVAL®

Tests Reference Multiple States (1)

- Need to allow a single item to be tested against multiple states.
  - specify acceptable ranges
  - specify multiple acceptable values
  - simplify test authoring

- Test that min password length is between 8 and 16
  - Items must satisfy state 1:

```
<min_passwd_len datatype="int" operation="greater than or equal">8</min_passwd_len>
```

  - AND state 2:

```
<min_passwd_len datatype="int" operation=“less than or equal">16</min_passwd_len>
```

Tests Reference Multiple States (2)

- What would change?
  - Change the maxOccurs on each test’s state element to unbounded.

```
<xsd:element name="state" type=“StateRefType" minOccurs="0" maxOccurs="unbounded"/>
```

  - Need to specify how to logically combine states.
    - Introduce the @state_operator on the oval-def:TestType
      - based on the oval:OperatorEnumeration (AND, OR, XOR, & ONE)

Tests Reference Multiple States (3)

- Does this change fit into version 5.6? Is this change too big for a minor version?
  - Impact of change:
    - Introduces a new state_operation on the oval-def:TestType
    - Changes the multiplicity of states in the oval-def:TestType
    - Does not invalidate existing content
  - Benefit of change:
    - Allows for the expression of ranges of acceptable values
    - Simplifies content authoring

Introduce PCRE Based Pattern Matches

Introduce PCRE Based Pattern Matches (1)

- Changing regular expression syntax was discussed at 2008 OVAL Developer Days for version 6.
  - PCRE based regular expressions are best fit for OVAL. See the “Regular Expression Syntax” section of the minutes: http://oval.mitre.org/oval/documents/docs-08/developerdays_minutes.pdf

Introduce PCRE Based Pattern Matches (2)

- Proposal to introduce PCRE and deprecate POSIX
  - deprecate the “pattern match” operation of the oval:OperationEnumeration

```
<value datatype="string" operation="pattern match">\d</value>
```

  - add “pcre pattern match” to the oval:OperationEnumeration

```
<value datatype="string" operation="pcre pattern match">\d</value>
```

Introduce PCRE Based Pattern Matches (3)

- Does this change fit into version 5.6? Is this change too big for a minor version?
  - Impact of change:
    - Introduces a new operation in the oval:OperationEnumeration
    - Deprecates the current POSIX based pattern match
    - Does not invalidate existing content
    - Must support two regex syntaxs
  - Benefit of change:
    - Nearly everyone is using PCRE anyway
    - Ensures standard meaning for all content

OVAL for System Querying?

MITRE

OVAL for System Inventory? (1)

- The OVAL Definitions Schema defines a framework for making assertions about machine state.
- OVAL Objects easily allow an author to express a request for all items on a system.
  - open ports, RPMs, files, registry keys, packages, …
- OVAL does not provide a framework for a performing a system inventory.

OVAL for System Querying? (2)

- Should OVAL consider System Querying as a new emerging use case?
  - Is there enough interest to justify the work?
  - Is there enough support to do the work?
  - Is this simply a distraction for OVAL?

OVAL Repository Considerations

OVAL Repository Considerations

- Should inventory definitions be required to have CPE Names?
  - Suggests that Windows XP SP2 or later is not an inventory definition. Changes its class to miscellaneous
- Should compliance definitions be required to have CCE IDs?
  - Suggests that definitions without CCE IDs are not compliance definitions. Changes its class to miscellaneous.

OVAL Repository Considerations

- We have introduced test and example content
  - Currently available under the ‘miscellaneous’ class
  - Should we further segregate them to their own namespace? ‘org,mitre.oval.test’
- Test content has system dependencies
  - How do we convey these dependencies?
    - Are notes sufficient at the definition level?

OVAL Repository Considerations

- Inconsistent usage of the affected platform and product on inventory definitions.
  - Should the Windows XP is installed inventory definition have an affected platform?
  - Should the Windows XP SP3 is installed inventory definition have an affected platform of Windows XP or Windows XP SP3?
  - Should the IE 7 inventory definition have an affected product at all?
  - Should the IE 7 SP 1 inventory definition have an affected product of IE 7 or IE 7 SP1?