OVAL Definition Tutorial

MITRE

Agenda

- Common XML Concepts
- OVAL Definition Tutorial
  - The Basics
    - Definition structure
    - Hello World
  - Advanced Topics
    - OVAL Definitions document Extended Definitions
    - Variables
    - Complex objects
    - Behaviors
    - Nil
- Known Issues

XML Namespaces

- namespace vs prefix
  - `xmlns:win-def="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows"`
- default namespace
  - `xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5"`
- using namespace
  - `<oval:schema_version>5.0</oval:schema_version>`
  - `<file_test xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows">`
  - `<file_test xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#unix">`

schemaLocation

- used to identify schema file to validate content

```
<?xml version="1.0" encoding="UTF-8"?>
<oval_definitions xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5"
                  xmlns:oval="http://oval.mitre.org/XMLSchema/oval-common-5"
                  xmlns:oval-def="http://oval.mitre.org/XMLSchema/oval-definitions-5"
                  xmlns:win-def="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:schemaLocation="http://oval.mitre.org/XMLSchema/oval-common-5 oval-common-schema.xsd
                                      http://oval.mitre.org/XMLSchema/oval-definitions-5 oval-definitions-schema.xsd
                                      http://oval.mitre.org/XMLSchema/oval-definitions-5#windows windows-definitions-schema.xsd">


      <definitions> ... </definitions>


</oval_definitions>
```

Which schema file is used to validate the <definitions> element?

OVAL Language Namespaces

OVAL Common Schema

xmlns:**oval**="http://oval.mitre.org/XMLSchema/oval-common-5"

OVAL Definition Schema

xmlns:**oval-def**="http://oval.mitre.org/XMLSchema/oval-definitions-5"
xmlns:**apache-def**="http://oval.mitre.org/XMLSchema/oval-definitions-5#apache"
xmlns:**macos-def**="http://oval.mitre.org/XMLSchema/oval-definitions-5#macos"
xmlns:**win-def**="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows"

OVAL System Characteristics Schema

xmlns:**oval-sc**="http://oval.mitre.org/XMLSchema/oval-system-characteristics-5"
xmlns:**unix-sc**="http://oval.mitre.org/XMLSchema/oval-system-characteristics-5#unix"
xmlns:**ios-sc**="http://oval.mitre.org/XMLSchema/oval-system-characteristics-5#ios"

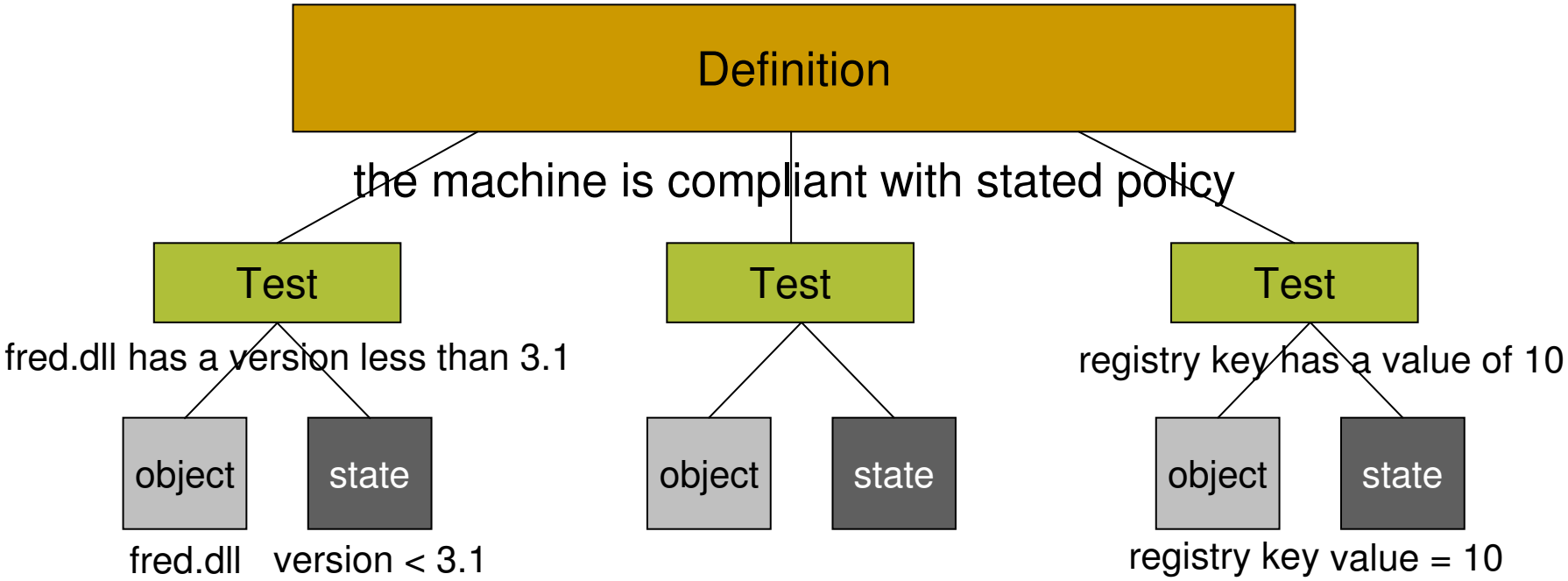
OVAL Results Schema

xmlns:**oval-res**="http://oval.mitre.org/XMLSchema/oval-results-5"

OVAL Definitions

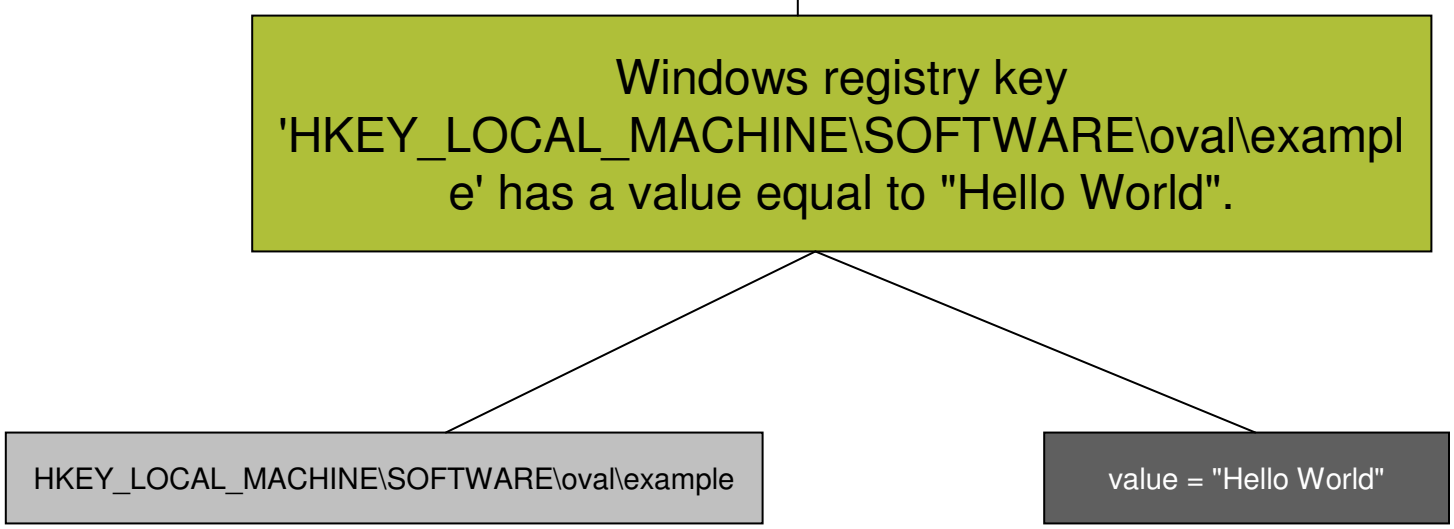
MITRE

Structure of an OVAL Definition

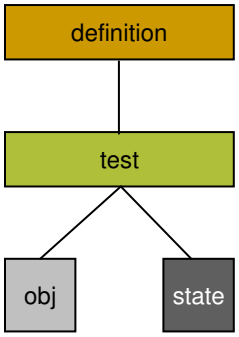


Hello World

write an OVAL Definition to test that the (hypothetical) Windows registry key 'HKEY_LOCAL_MACHINE\SOFTWARE\oval\example' has a value equal to "Hello World".

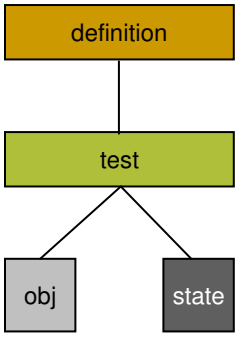


Hello World - Registry Object



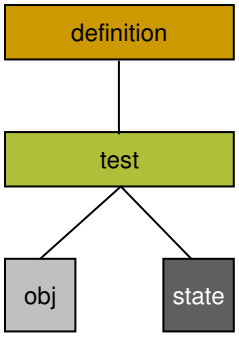

```
<registry_object id="oval:com.example:obj:1">
    <hive>HKEY_LOCAL_MACHINE</hive>
    <key>SOFTWARE\oval</key>
    <name>example</name>
</registry_object>
```

Hello World - Registry State



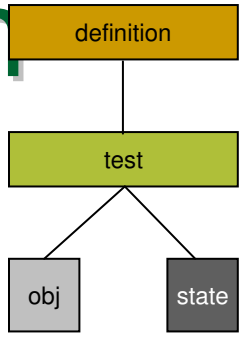

```
<registry_state id="oval:com.example:ste:1">
  <value operation="equals">Hello World</value>
</registry_state>
```

Hello World - Registry Test




```
<registry_test id="oval:com.example:tst:1" check="all">
    <object object_ref="oval:com.example:obj:1"/>
    <state state_ref="oval:com.example:ste:1"/>
</registry_test>
```

Hello World - OVAL Definition




```
<definition id="oval:com.example:def:1">
  <metadata>
    <title>Hello World Example</title>
    <description>
      This definition is used to introduce the
      OVAL Language to individuals interested
      in writing OVAL Content.
    </description>
  </metadata>
  <criteria>
    <criterion test_ref="oval:com.example:tst:1"
   comment="the value of the registry key equals
   Hello World"/>
  </criteria>
</definition>
```

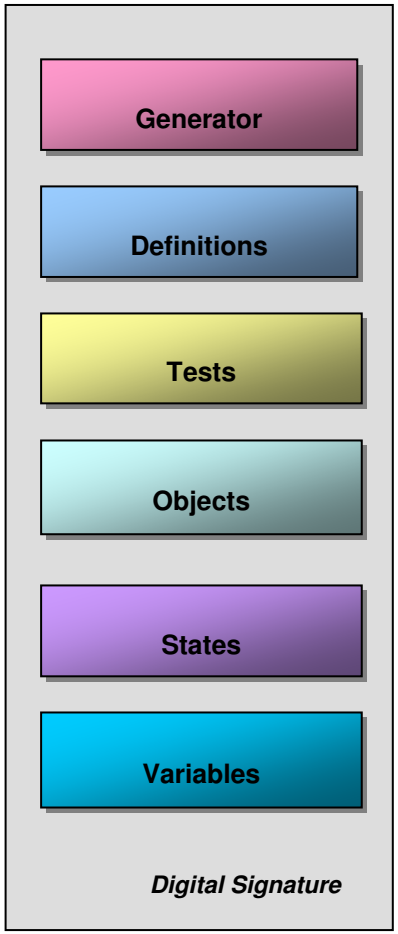
Full XML

Advanced Topics

An OVAL Definition File

XML Example

- Generator
- Definitions
- Tests
- Objects
- States
- Variables
- Digital Signature



Generator Section

- Information about how the OVAL Document was created
  - product name
  - product version
  - schema version
  - timestamp

- Not about the content, but about the document!

```
<generator>
      <oval:product_name>Guide Writer</oval:product_name>
      <oval:product_version>1.2</oval:product_version>
      <oval:schema_version>5.0</oval:schema_version>
      <oval:timestamp>2005-10-12T18:13:45</oval:timestamp>
</generator>
```

Definitions Section

- A container for individual OVAL Definitions
- Each definition has two parts
  - metadata
  - criteria
- Different classes of definitions
  - vulnerability
  - configuration
  - patch
  - inventory

Definition Metadata

- **data to help classify an OVAL Definition**
  - not part of the <criteria>
  - not used in evaluating the definition

- **<xsd:any>**
  - allow other information that a definition writer feels is important
  - tools can use if they want
  - can not count on an OVAL Compatible tool understanding this information
  - OVAL Repository metadata as an example

Definition Metadata

```
<definition id="" version="" class="">

   <metadata>

      <title></title>
      <affected family="windows">
          <platform>Microsoft Windows Server 2003</platform>
          <product>Adobe Reader</product>
      </affected>
      <reference source="CVE" ref_id="CVE-1234-5678"/>
      <description>A description of the definition.</description>

      <any-metadata/>

   </metadata>

   <criteria> ... </criteria>

</definition>
```

Definition Criteria

- references to the actual tests that must be performed

```
<definition id="" version="" class="">

   <metadata> ... </metadata>

   <criteria operator="AND">

      <criterion test_ref="" comment=""/>
      <criteria operator="OR">
         <criterion test_ref="" comment=""/>
         <criterion test_ref="" comment=""/>
      </criteria>
      <extend_definition definition_ref="" comment=""/>

   </criteria>

</definition>
```

Extended Definitions

- Existing definitions may be extended.
  - Add workarounds to an existing vulnerability def
- Common units of logic can be broken out.
  - Microsoft Windows XP SP2 is installed
- Easier/Faster to create new definitions

Test Section

- A container for a set of tests
- A test checks a set of items on a system for an expected state.
- Each test calls out
  - an object set
  - a state used for comparison
  - a check attributes to guide the evaluation

Check Attributes

- check_existence attribute
  - Specifies the number of items that must be present for the test to evaluate to true
    - all_exist, any_exist, at_least_one_exists, none_exist, only_one_exists

- check attribute
  - Specifies the number of items that must satisfy the state.
    - all, at least one, none exist, none satisfy, only one

Unknown Tests

- a placeholder for tests whose implementation is unknown.
- Any information that is known about the test should be held in the notes
- The required check attribute is ignored during evaluation
- Always evaluates to “unknown”

Object Section

- A container for a set of objects
- An object defines a **set** of items on a system to examine
- Each object has
  - id
  - comment
  - deprecated flag
  - version

Complex Objects - intro

An Object identifies 0 or more items on a system.

Set consists of all registry keys that match the object

```
<registry_object …>

    <hive>HKEY_LOCAL_MACHINE</hive
    >
   <key>ExampleKey</key>
   <name>ExampleName</name>
</registry_object>
```

```
<registry_object …>
   <hive>HKEY_LOCAL_MACHINE</hive>
   <key>ExampleKey</key>
   <name operation="pattern match">.*</name>
</registry_object>
```

Complex Objects - set element

- ability to manipulate these sets.
  - set element
    - set_operator
    - object references
    - filters

Set consists of all registry keys that match the criteria

```
<registry_object …>
   <set set_operator="UNION">
      <object_reference>objId1</object_reference>
      <object_reference>objId2</object_reference>
      <filter>stateId1</filter>
      <filter>stateId2</filter>
   </set>
</registry_object>
```

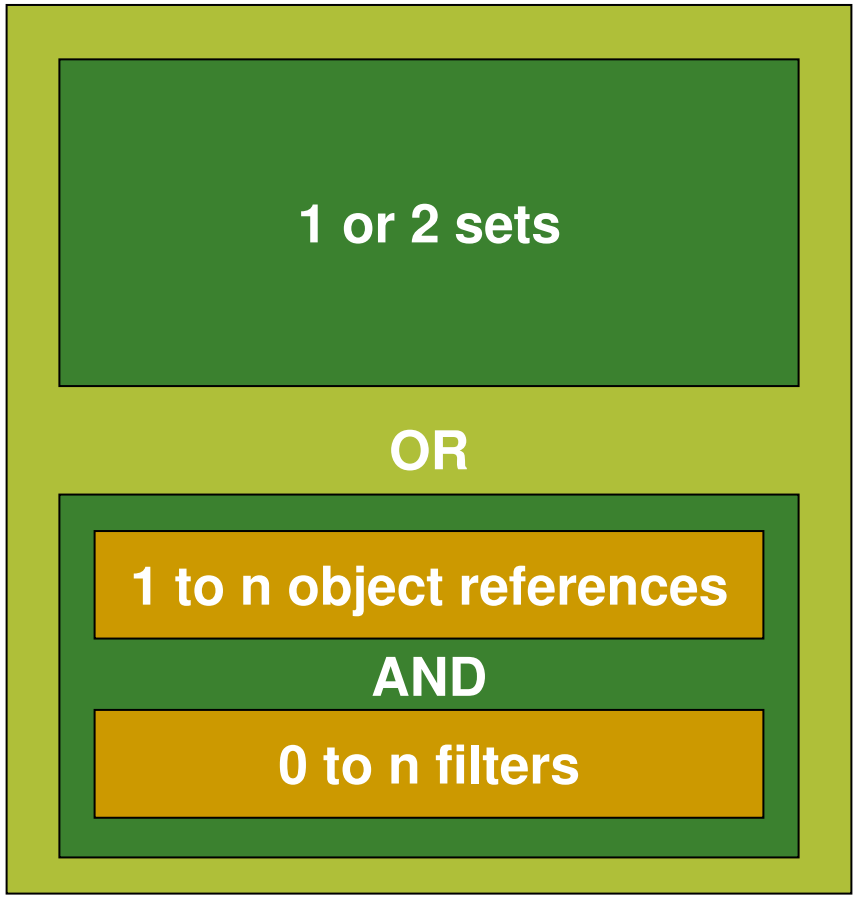
Complex Objects - set element - details

- Element contents
  - 1 or 2 child set elements

  OR
  - 1 to n object_reference elements

    AND
  - 0 to n filters

- set_operator attribute
  - UNION
  - COMPLEMENT
  - INTERSECTION



Complex Objects - Filters

- A filter is a state that is used to “filter out” items from a set.
- Any number of filters can be applied.
- Filters are applied before the set_operator is applied.

Complex Objects

Trustees not part of the ADMINISTRATORS group or the user SYSTEM do not have access to the specified file.

- Identify the file.
- Identify the trustees that should not have access.
  - Identify all trustees on the system
  - and remove the trustees in the admin group
  - and remove the System user
- Check permissions on the file for each trustee that should not have access.

Behaviors

- Allow more detailed definition of an Object
- Implemented on a per object basis
- Guides data collectors

```
<file_object ...>
  <behaviors max_depth="2" recurse_direction="down"/>
  <path>c:\windows</path>
  <filename>fred.dll</filename>
</file_object>
```

State Section

- Container for a set of states
- A state defines the expected “state” for a set of items on a system
- Each state has
  - id
  - comment
  - deprecated flag
  - version

Variables Section

- A container for a set of variables
- Variables define values to be obtained at run time
- Variables represent an array of values
- Three types of variables
  - local_variable
  - external_variable
  - constant_variable

Variables - constant_variable

- Value is set by definition author.

- Helpful when
  - creating complex variables.
  - easy reuse of common constant values

```
<constant_variable comment="…" datatype="string" version="1" id="…">
   <value>system32</value>
</constant_variable>
```

Variables - local_variable

- Value determined during evaluation
- Manipulate values fetched from objects, other variables, or literals.
- Functions (concat, substring, split, …)

```
<constant_variable datatype="string" id="var1">
   <value>\system32</value>
</constant_variable>
<local_variable id="var2" datatype="string">
   <concat>
      <object_component item_field="value" object_ref="obj1"/>
      <variable_component var_ref="var1"/>
   </concat>
</local_variable>
```

Variables - external_variable

- Defines a variable with an external source.

- Gives suggestion about type of data and reasonable values.

```
<external_variable id=“var1" comment=“the range 8-16, or 32"
datatype="int">
   <possible_restriction hint="min is 8">
      <restriction operation="greater than">7</restriction>
      <restriction operation="less than">17 </restriction>
   </possible_restriction>
   <possible_value>32</possible_value>
</external_variable>
```

Nil vs. pattern match .*

Confirm that the specified directory exists…

- Nil allows authors to specify higher level objects.
- Nil is only allowed on select entities.
- Implemented with xsi:nil="true“
- file_object example:
  - xsi:nil="true" on filename entity
    - Don’t collect file information.
  - Pattern match .* on filename entity
    - Collect file information about all files.

Signing OVAL Documents

- Defined by the XML-Signature Syntax and Processing W3C Recommendation

- Enveloped Signature - The signature is over the XML content that contains the signature as an element.

Known Issues

- patch definitions
- remediation
- <xsd:any>
- pattern match on enumerations
- xmlcontents, wmi and sql test
- multi-line text file contents
- splitting file paths and file names

Backup

Hello World - Full XML

Return

```
<oval_definitions …>
  <generator>…</generator>
  <definitions>
    <definition id="oval:org.mitre.oval.tutorial:def:1" version="1" class="miscellaneous">
      <metadata>
        <title>Hello World Example</title>
        <affected family="windows"/>
        <description>This definition is used to introduce the OVAL Language to individuals interested in writing OVAL Content.</description>
      </metadata>
      <criteria comment="Software section" operator="AND">
        <criterion comment="The oval example registry key has a value of &amp;quot;Hello World&amp;quot;" test_ref="oval:org.mitre.oval.tutorial:tst:1"/>
      </criteria>
    </definition>
  </definitions>
  <tests>
    <registry_test id="oval:org.mitre.oval.tutorial:tst:1" version="1" check="at least one" comment="The oval example registry key has a value of &amp;quot;Hello
        World&amp;quot;" xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows">
      <object object_ref="oval:org.mitre.oval.tutorial:obj:1"/>
      <state state_ref="oval:org.mitre.oval.tutorial:ste:1"/>
    </registry_test>
  </tests>
  <objects>
    <registry_object id="oval:org.mitre.oval.tutorial:obj:1" version="1" xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows">
      <hive>HKEY_LOCAL_MACHINE</hive>
      <key operation="equals">SOFTWARE\oval</key>
      <name operation="equals">example</name>
    </registry_object>
  </objects>
  <states>
    <registry_state id="oval:org.mitre.oval.tutorial:ste:1" version="1" xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows">
      <value operation="equals">Hello World</value>
    </registry_state>
  </states>
</oval_definitions>
```

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<oval_definitions xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5"
                  xmlns:oval="http://oval.mitre.org/XMLSchema/oval-common-5"
                  xmlns:oval-def="http://oval.mitre.org/XMLSchema/oval-definitions-5"
                  xmlns:win-def="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:schemaLocation="http://oval.mitre.org/XMLSchema/oval-common-5 oval-common-schema.xsd
                                      http://oval.mitre.org/XMLSchema/oval-definitions-5 oval-definitions-schema.xsd
                                      http://oval.mitre.org/XMLSchema/oval-definitions-5#windows windows-definitions-schema.xsd">

      <generator>
            <oval:schema_version>5.0</oval:schema_version>
            <oval:timestamp>2005-10-12T18:13:45</oval:timestamp>
      </generator>

      <definitions>
            <definition id="oval:org.mitre.oval:def:999" version="1" class="inventory">
                  <metadata>
                        <title>Microsoft Windows Server 2003 32-Bit Edition is installed</title>
                        <affected family="windows">
                              <platform>Microsoft Windows Server 2003</platform>
                        </affected>
                        <description>A version of Microsoft Windows Server 2003 32-Bit Edition is installed.</description>
                  </metadata>
                  <criteria operator="AND">
                        <criterion test_ref="oval:org.mitre.oval:tst:61" comment="Windows Server 2003 is installed"/>
                        <criterion test_ref="oval:org.mitre.oval:tst:72" comment="32-Bit version of Windows is installed"/>
                  </criteria>
            </definition>
      </definitions>

      ...
```

Example

```
...

<tests>
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~   windows registry tests   ~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <registry_test id="oval:org.mitre.oval:tst:61"
                     version="1"
                     check="at least one"
                     comment="Windows Server 2003 is installed"
                     xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows">
            <object object_ref="oval:org.mitre.oval:obj:3"/>
            <state state_ref="oval:org.mitre.oval:ste:3"/>
      </registry_test>
      <registry_test id="oval:org.mitre.oval:tst:72"
                     version="1"
                     check="at least one"
                     comment="32-Bit version of Windows is installed"
                     xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows">
            <object object_ref="oval:org.mitre.oval:obj:4"/>
            <state state_ref="oval:org.mitre.oval:ste:4"/>
      </registry_test>
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
</tests>

...
```

Example

```
...
<objects>
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~   windows registry objects   ~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <registry_object id="oval:org.mitre.oval:obj:3" version="1" xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-
5#windows">
            <hive>HKEY_LOCAL_MACHINE</hive>
            <key>SOFTWARE\Microsoft\Windows NT\CurrentVersion</key>
            <name>CurrentVersion</name>
      </registry_object>
      <registry_object id="oval:org.mitre.oval:obj:4" version="1" xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-
5#windows">
            <hive>HKEY_LOCAL_MACHINE</hive>
            <key>SYSTEM\CurrentControlSet\Control\Session Manager\Environment</key>
            <name>PROCESSOR_ARCHITECTURE</name>
      </registry_object>
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
</objects>
<states>
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~   windows registry states   ~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <registry_state id="oval:org.mitre.oval:ste:3" version="1" xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-
5#windows">
            <value>5.2</value>
      </registry_state>
      <registry_state id="oval:org.mitre.oval:ste:4" version="1" xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-
5#windows">
            <value>x86</value>
      </registry_state>
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
      <!-- ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ -->
</states>
...
```

Example

```
...

<variables>
      <local_variable id="oval:org.mitre.oval:var:1" version="1" datatype="string" comment="Windows system32 directory">
            <concat>
                  <object_component object_ref="oval:org.mitre.oval:obj:123" item_field="value"/>
                  <literal_component>\system32\</literal_component>
            </concat>
      </local_variable>
</variables>

</oval_definitions>
```