VMware SCAP Edit

User Guide

Revision 1.0 – 11-Jul-2016

Table of Contents

Introduction

VMware SCAP Edit is an updated version of the Enhanced SCAP Content Editor tool by G2, Inc. The modified tool builds on similar capabilities but few subtle differences required to build OVAL content based on latest OVAL schema.

The major changes done by VMware are as below:

- Added support for OVAL 5.11 for Independent, Unix, Linux and Windows schemas
- Added support for XCCDF 1.2 creation from OVAL
- Refreshed the tool with modern UI
- Dropped broken capabilities from previous versions of the tool
- Dropped obsolete schemas and all other seldom used features
- Updated libraries to latest versions
- Updated CPE version to 2.3
- Restructured the code
- Removed obsolete and unneeded libraries

These changes make the tool simple and intuitive to build OVAL content specially aligned with the latest 5.11 OVAL schema. Also, it can now create XCCDF 1.2 content directly from OVAL file.

Note: This tool and the documentation below assumes that you are fairly aware of SCAP, OVAL, XCCDF and the surrounding scheme of things. If you are not, please take a moment and familiarize yourself with these before proceeding further.

Installation and getting started

System Requirements

- A Java Runtime Environment (JRE) preferably 1.8 or later.
- 1 GB of memory.
- Supported OS – Microsoft Windows and Linux (with support for JRE).

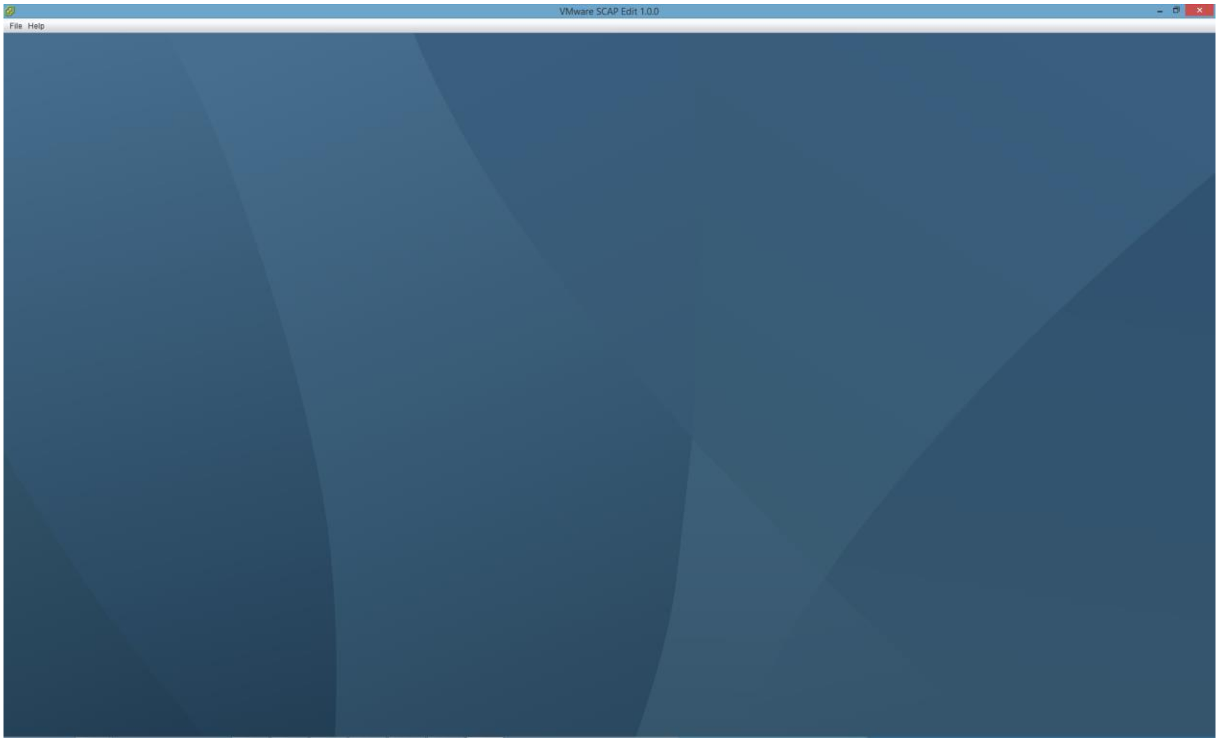
Installation

1) Extract the **VMware SCAP Edit 1.0.0.zip** file.
2) Navigate to the extracted directory.
3) If you are running a Microsoft Windows system, execute **startEditor.bat** file. This will launch the tool. If you are running a Linux system, execute **startEditor.sh** file to launch the tool.

These scripts will call Java with the appropriate arguments. These can be edited if you need to give Java more memory or set some other system property. Editing these files is only recommended for advanced users.

Using the tool

Launch the tool. You would see the tool interface as below:



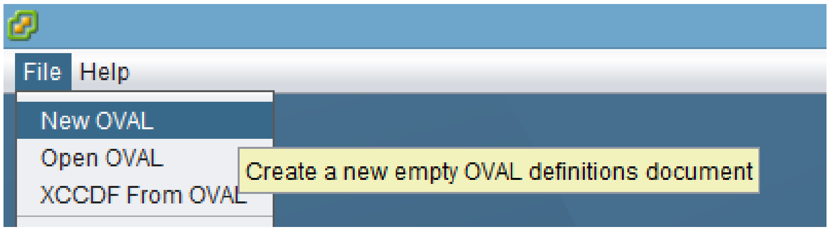
As you immediately notice, the tool has been simplified from its parent version. It now just has a file menu that offers you below core capabilities:

1) Creating a new OVAL file
2) Opening an existing OVAL file
3) Creating a XCCDF file from an existing OVAL file

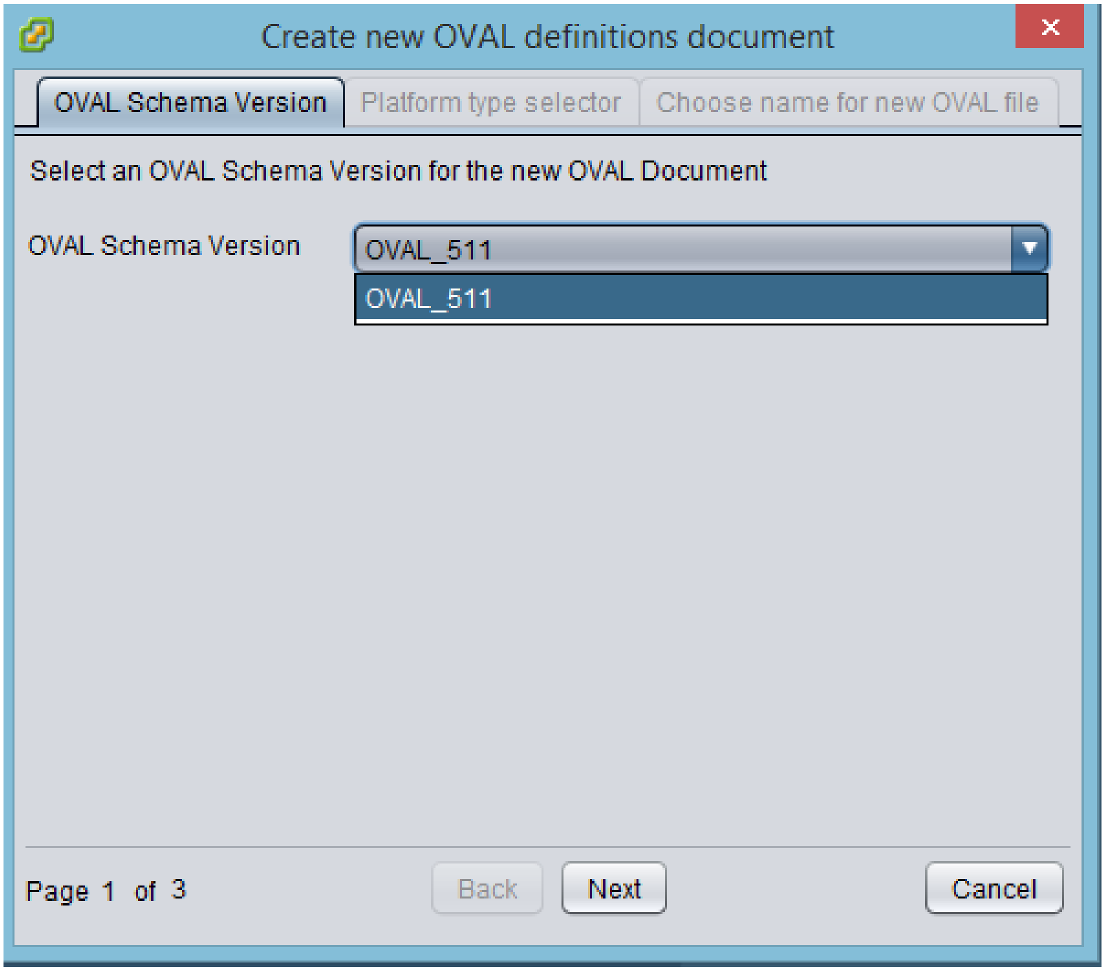
All other functionalities from the parent version have been taken out. Now, let us see each of the three functionalities in depth.

Creating a new OVAL file

Launch the tool. Navigate to **File → New OVAL**.



This opens a new dialog box with default OVAL 5.11 schema chosen. We have removed all other obsolete schema.



Create new OVAL definitions document

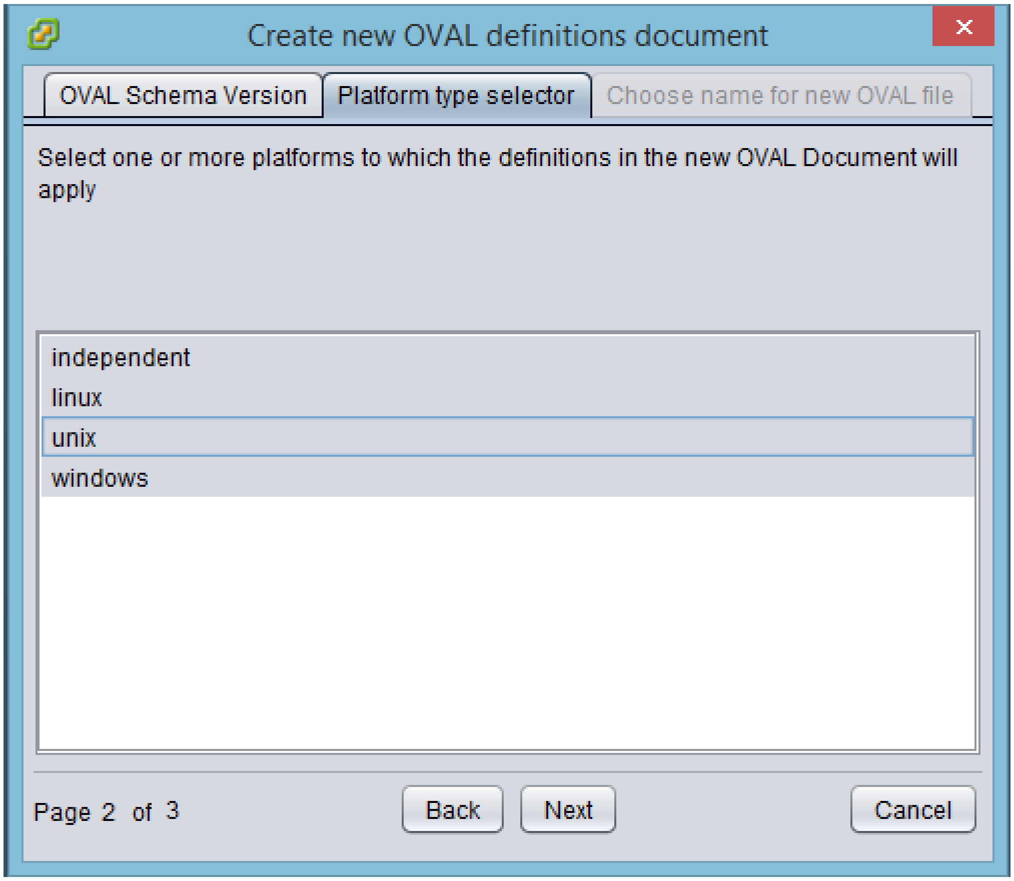
Select an OVAL Schema Version for the new OVAL Document

OVAL Schema Version: OVAL_511

Click **Next**.

Now, select the target platform. You may now only choose between Independent, Linux, Unix and Windows. We have dropped the support for any other platform.

For example, let us choose **Unix**.



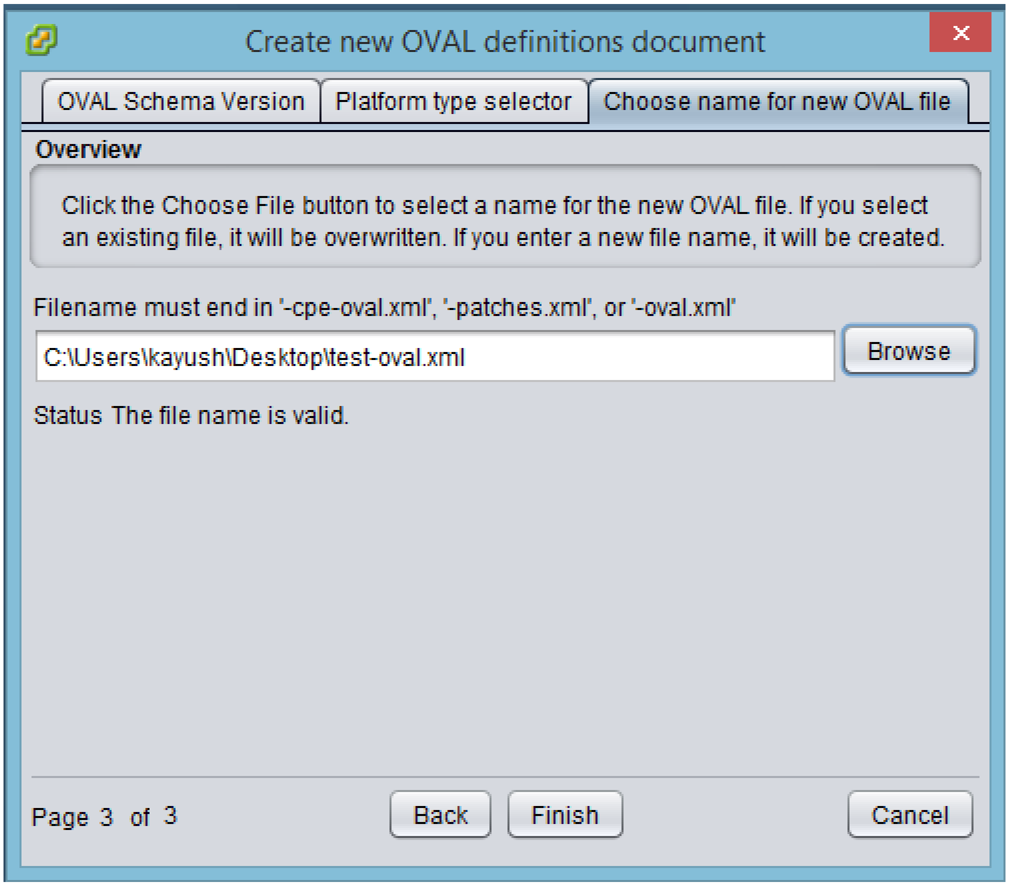
Create new OVAL definitions document
OVAL Schema Version
Platform type selector
Choose name for new OVAL file

Select one or more platforms to which the definitions in the new OVAL Document will apply

independent
linux
unix
windows

Click **Next**.

Finally, choose the file name.



Create new OVAL definitions document
OVAL Schema Version
Platform type selector
Choose name for new OVAL file

Overview

Click the Choose File button to select a name for the new OVAL file. If you select an existing file, it will be overwritten. If you enter a new file name, it will be created.

Filename must end in '-cpe-oval.xml', '-patches.xml', or '-oval.xml'

C:\Users\kayush\Desktop\test-oval.xml

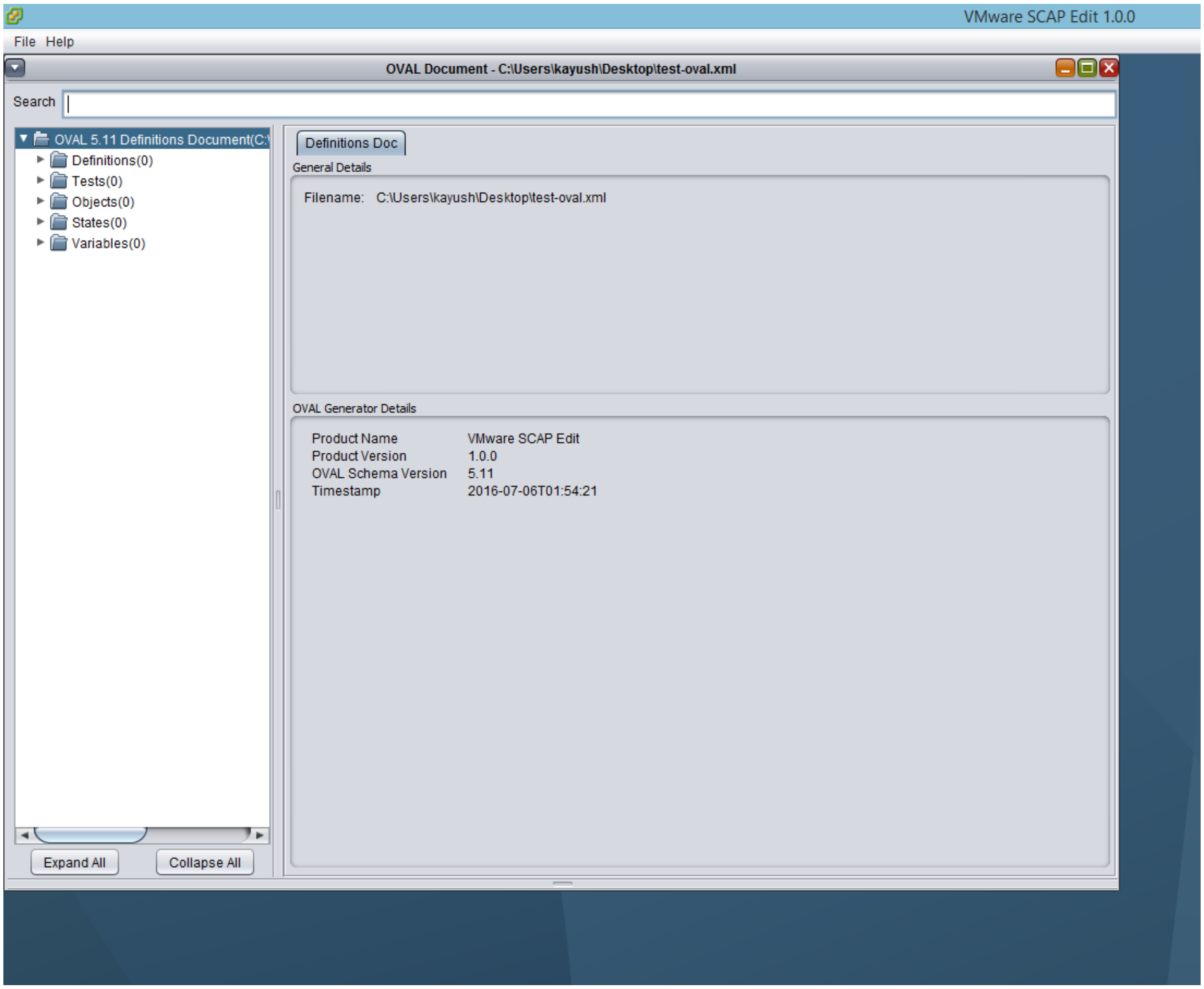
Status The file name is valid.

Page 3 of 3

Browse
Back
Finish
Cancel

Click **Finish**.

This creates a new and blank OVAL file.



Filename: C:\Users\kayush\Desktop\test-oval.xml

```
Product Name          VMware SCAP Edit
Product Version       1.0.0
OVAL Schema Version   5.11
Timestamp             2016-07-06T01:54:21
```

This tool helps you define various OVAL components:

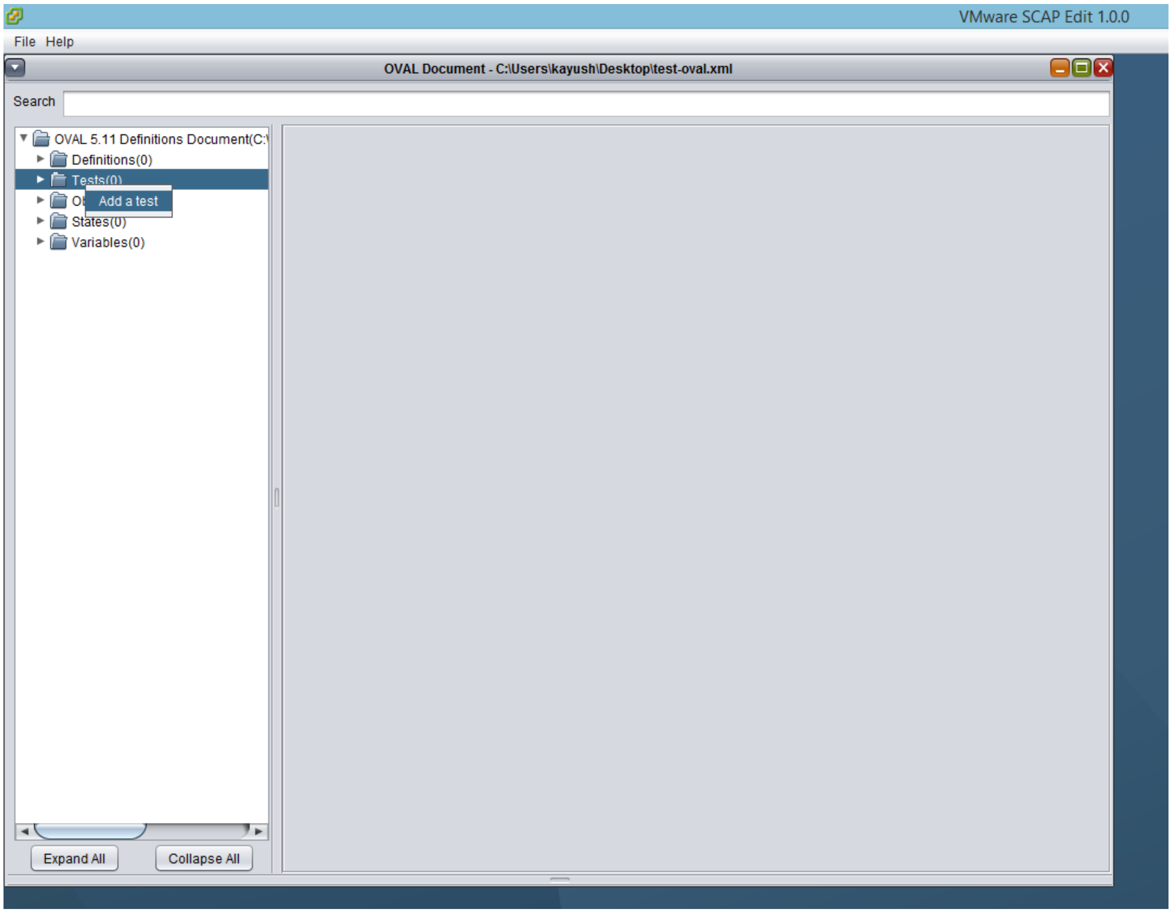
- Definitions
- Tests
- Objects
- States and
- Variables

You can define those components individually and then link them up together later. But, for this example, let us walk through a typical flow in creating an OVAL definition.

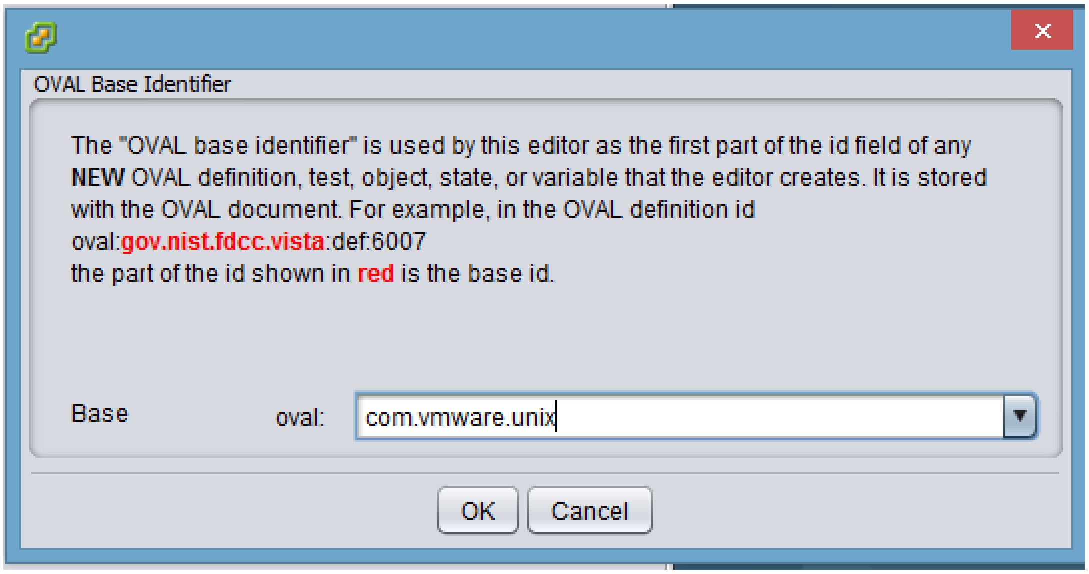
Adding a new OVAL definition

Let us take an example that we want to write an OVAL definition that verifies that the `/etc/passwd` file is owned and group-owned by `root` and has permissions of `644`.

Start with adding a Test. Right click on **Tests** and then click **Add Test**.



For your first test only, you will have to define OVAL base identifier.



OVAL Base Identifier

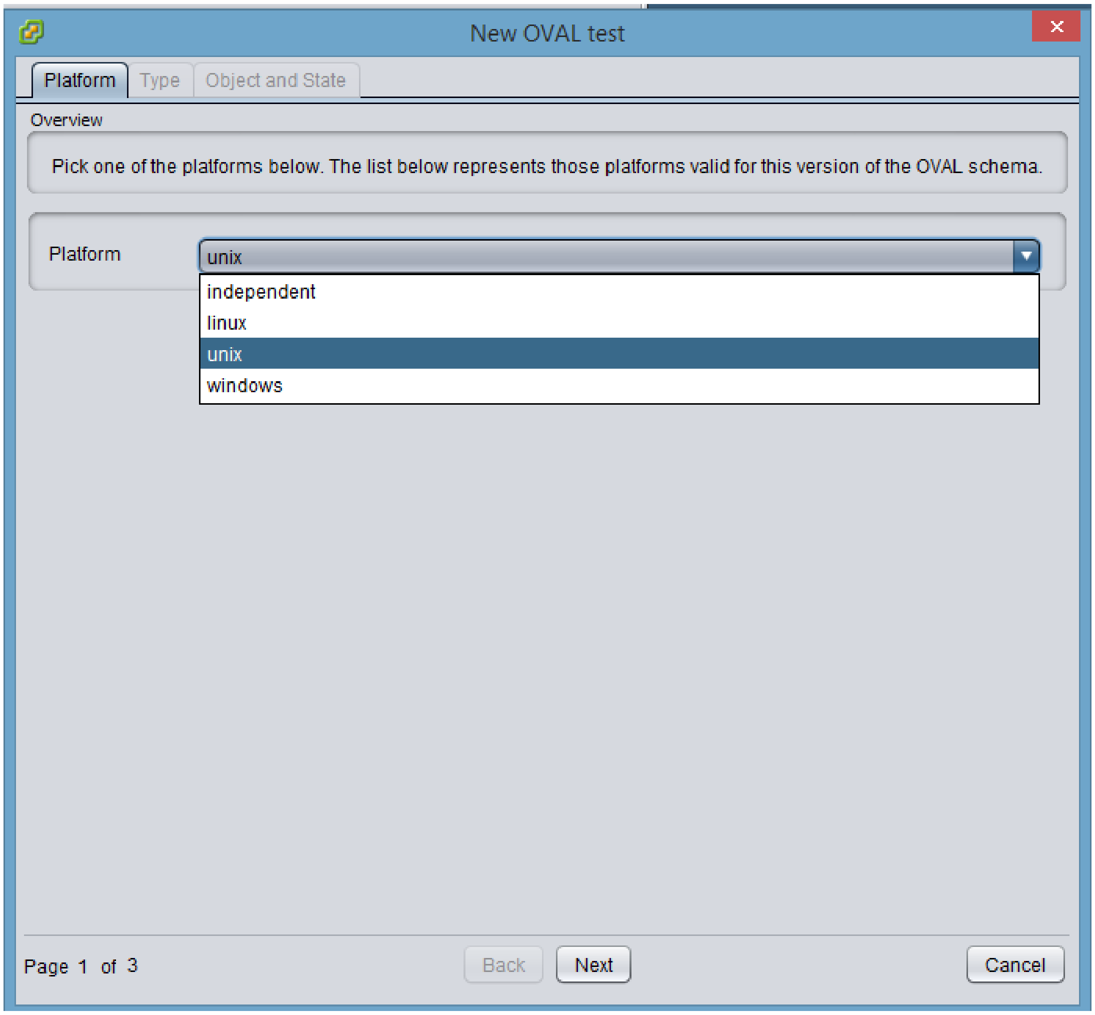
The "OVAL base identifier" is used by this editor as the first part of the id field of any **NEW** OVAL definition, test, object, state, or variable that the editor creates. It is stored with the OVAL document. For example, in the OVAL definition id
oval:gov.nist.fdcc.vista:def:6007
the part of the id shown in red is the base id.

Base oval: com.vmware.unix

For this example, let us set the base identifier to **com.vmware.unix**. You can set this up as you would require.

Click **OK**.

Now, choose the target platform for which you are writing this test. Let us choose **Unix** here.

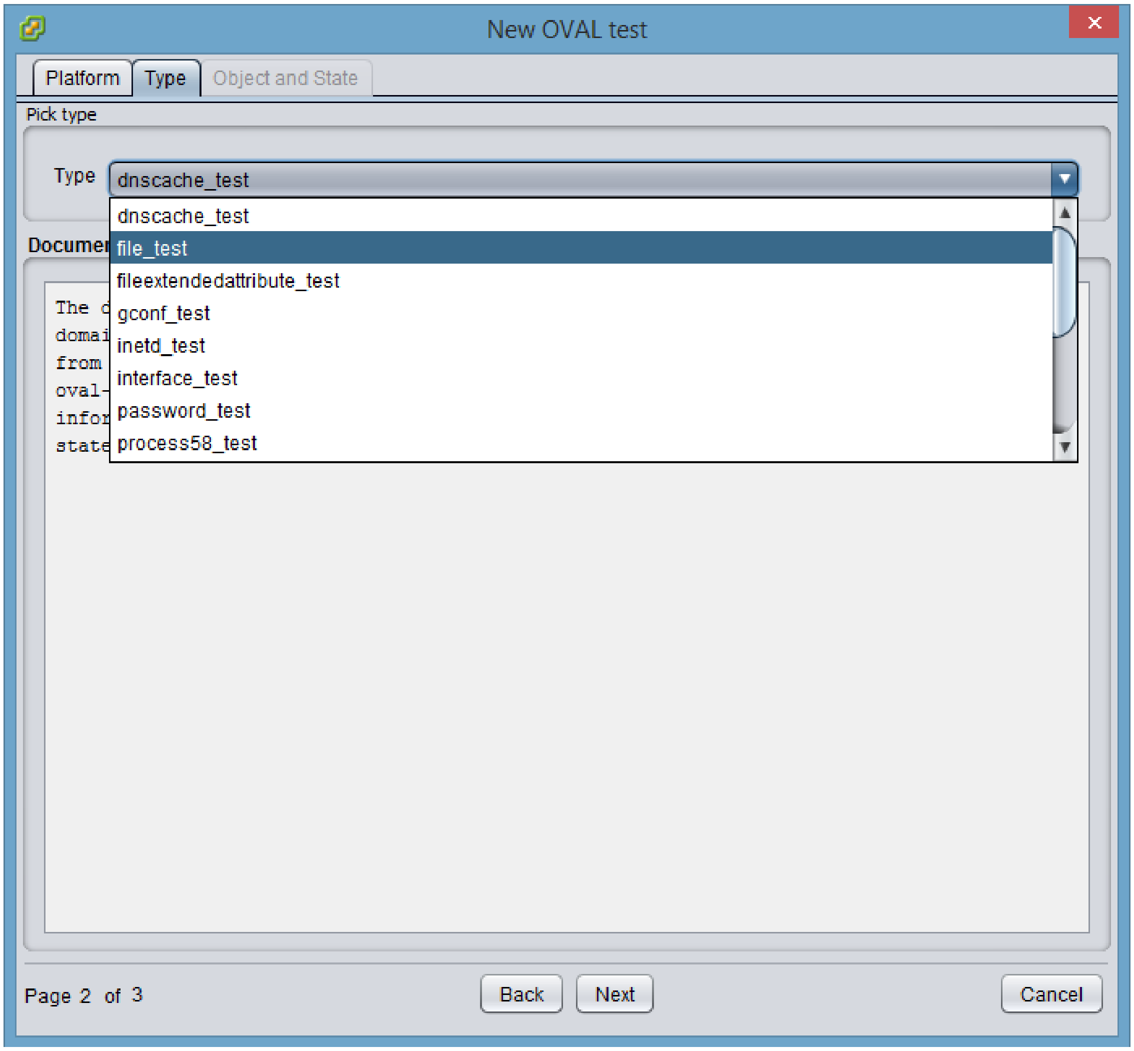


Pick one of the platforms below. The list below represents those platforms valid for this version of the OVAL schema.

Platform: unix

Click **Next**.

Now choose Test type or OVAL probe against which you want to define the test. Check out the references to read about these in detail.

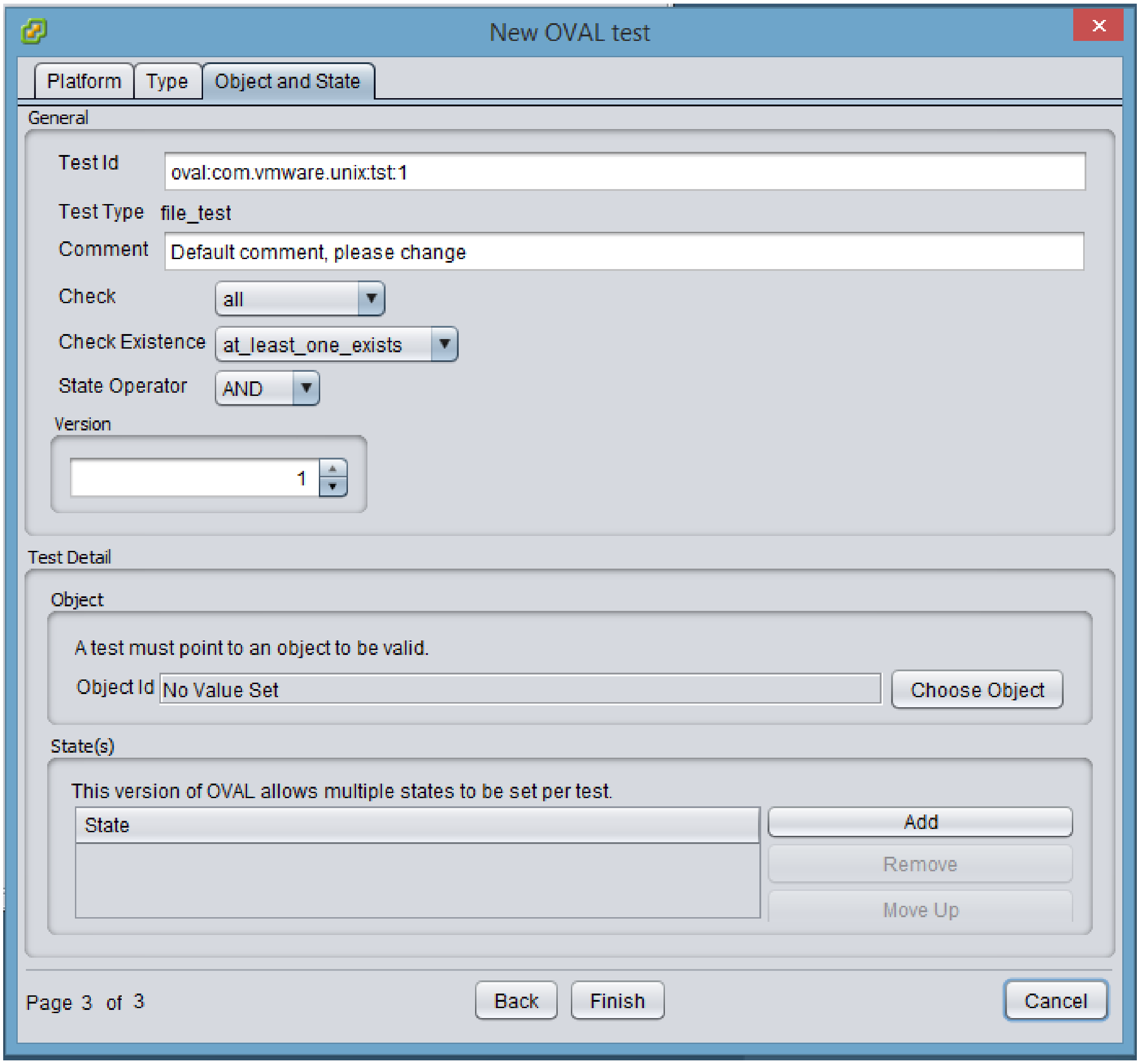


Type: dnscache_test

Page 2 of 3

For this example, let us choose **file_test**. Using this test type we can check various attributes on files such as ownership and permissions. Click **Next**.

As you know, each test is comprised of objects and states. On this screen, we will define objects and states to match our requirements (checking `/etc/passwd` file for ownership and group-ownership to be `root` and permissions of `644`).



General

Test Id: oval:com.vmware.unix:tst:1

Test Type: file_test

Comment: Default comment, please change

Check: all

Check Existence: at_least_one_exists

State Operator: AND

Version: 1

Test Detail

Object

A test must point to an object to be valid.

Object Id: No Value Set

State(s)

This version of OVAL allows multiple states to be set per test.

State

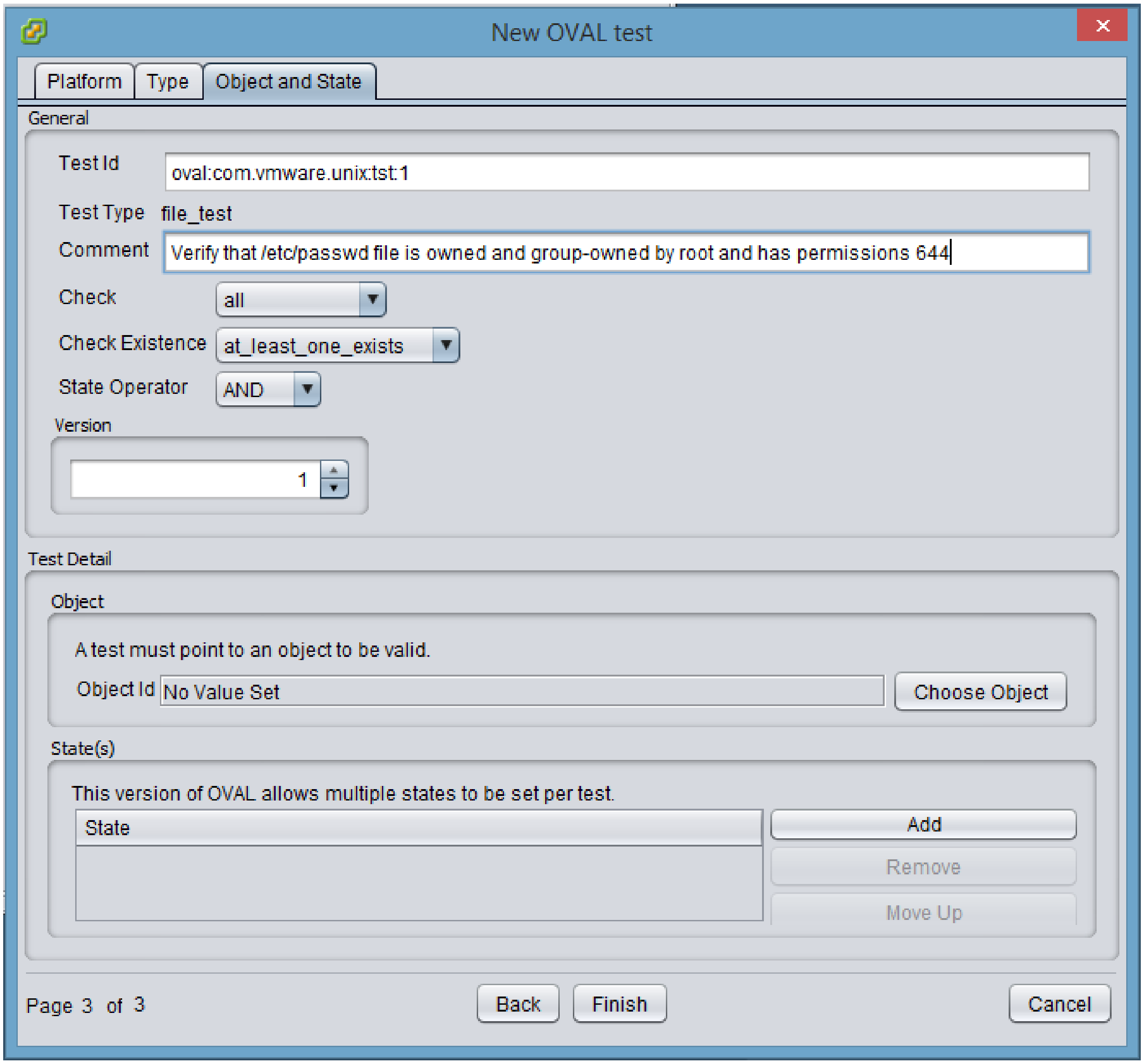
Page 3 of 3

As you notice above, the base identifier is preset. Also, you don’t have to manage the ids yourself once you define the base identifier. The tool keeps incrementing the ids as you add more tests. The same holds true for any other OVAL components such as definitions, objects, states and variables that you write. All the ids are taken care of by the tool.

Also, all the OVAL defaults for various fields such as Check, Check Existence and State Operator are preset. You can use the drop down to pick and choose the various options as matching your requirements.

Now, let us go ahead and define the test as needed by our requirement.

We start by giving a **Comment**.



Test Id: oval:com.vmware.unix:tst:1

Test Type: file_test

Comment: Verify that /etc/passwd file is owned and group-owned by root and has permissions 644

Check: all

Check Existence: at_least_one_exists

State Operator: AND

Version: 1

A test must point to an object to be valid.

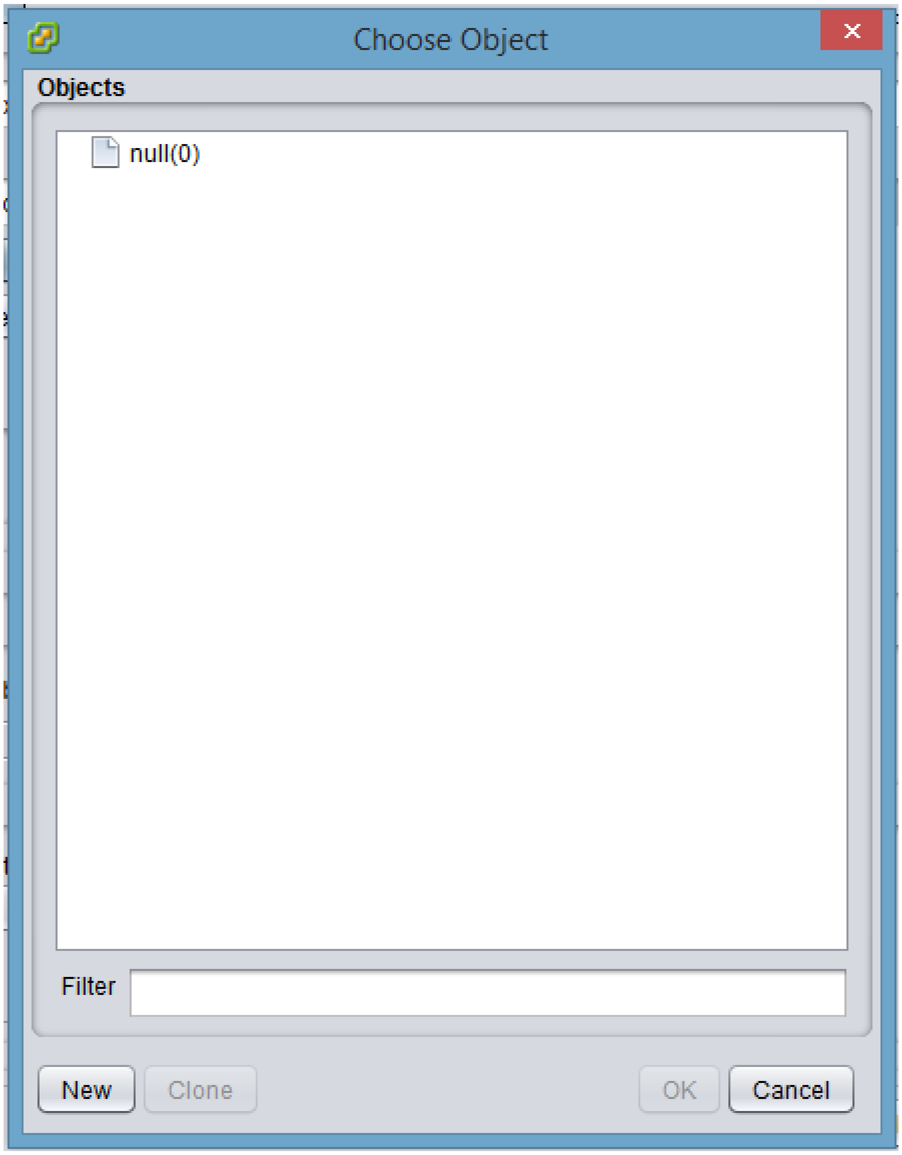
Object Id: No Value Set

This version of OVAL allows multiple states to be set per test.

Page 3 of 3

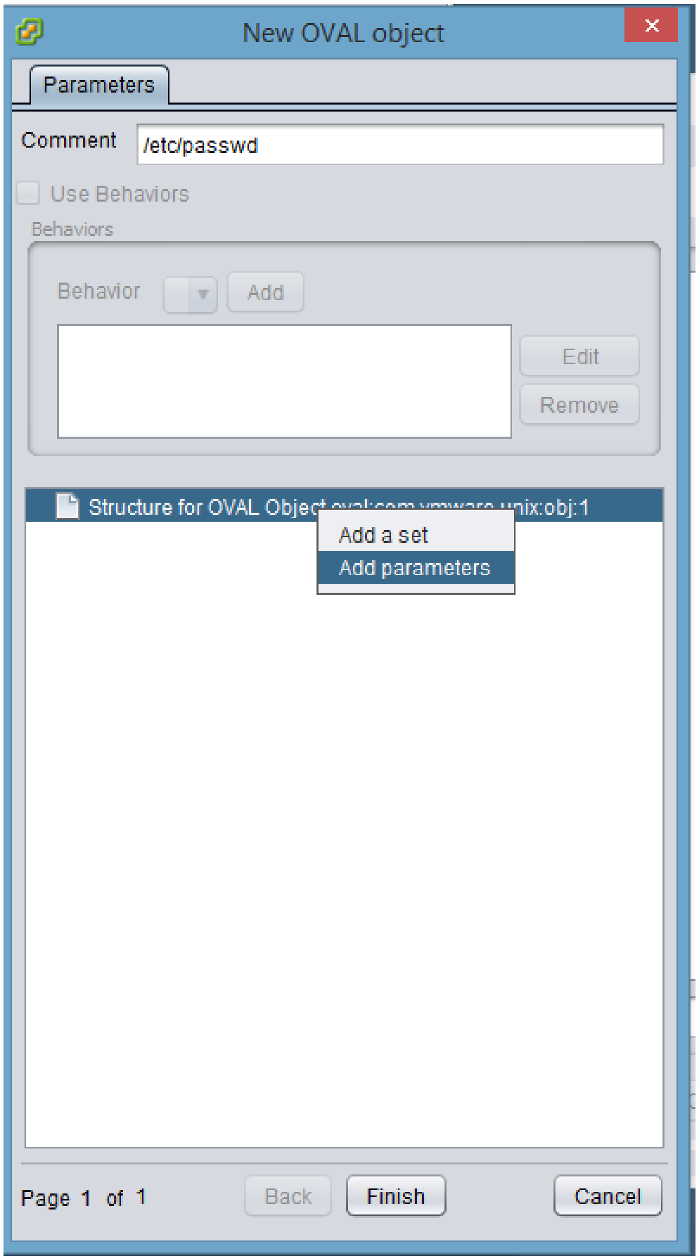
Now click on **Choose Object**.

This opens up Object wizard.



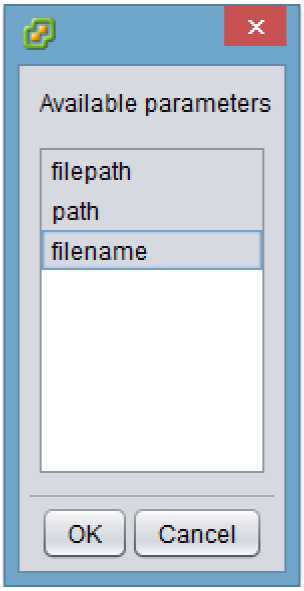
Click **New**.

This opens up New Object wizard. Give a comment. Right click on the below shown area to add parameters.



Comment /etc/passwd

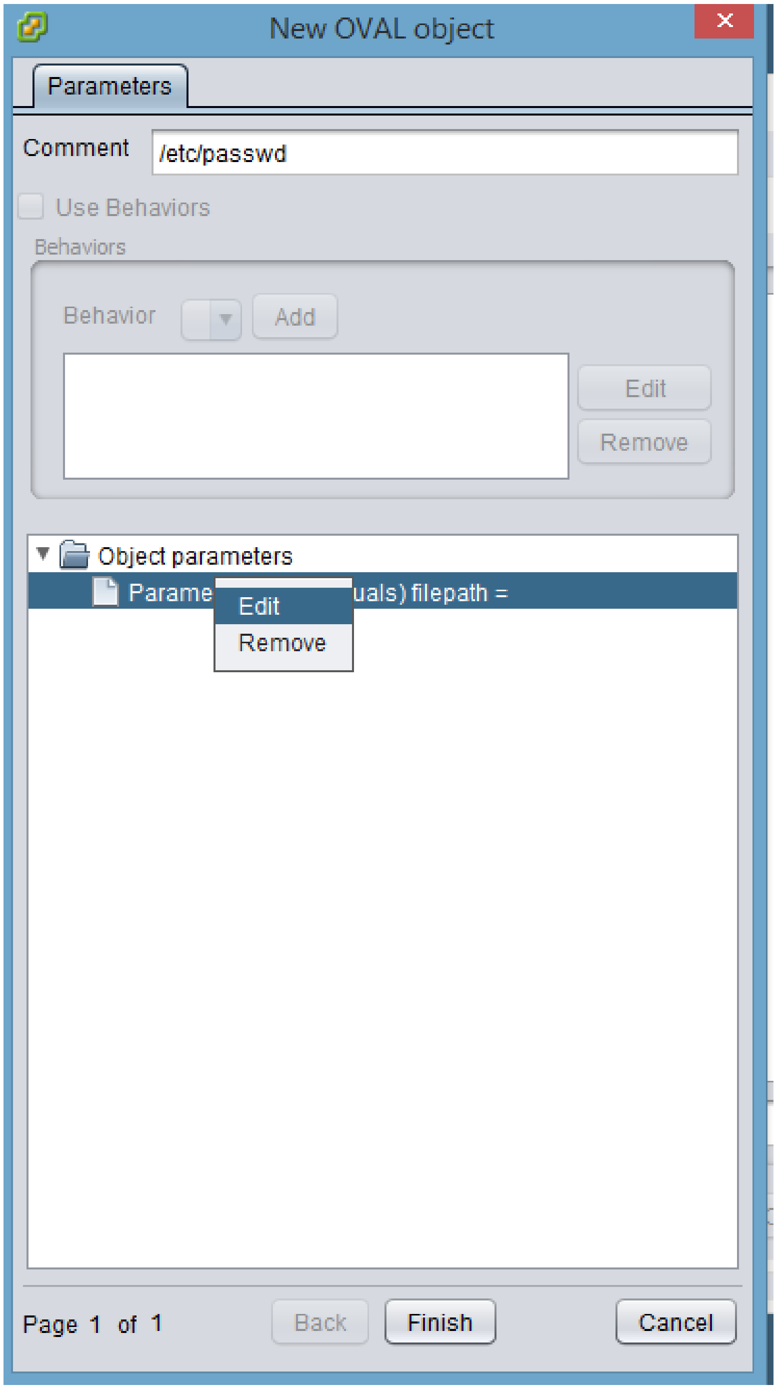
Since, we chose test type as file_test, only below corresponding object parameters are available.



Available parameters

filepath
path
filename

Here, let us choose to give absolute path of the file. In our example, it is `/etc/passwd`. Click **OK**.



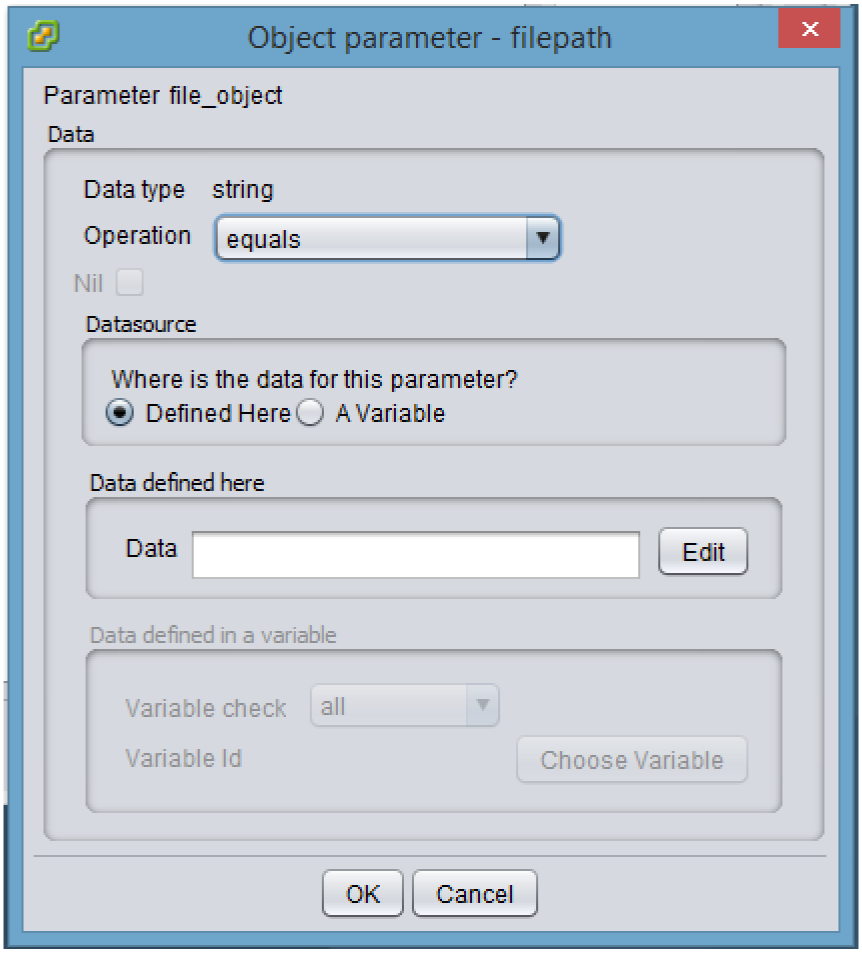
Comment: /etc/passwd

Object parameters

Parame… uals) filepath =

Page 1 of 1

Right click and select **Edit** to set the value for `filepath` object parameter.


Object parameter - filepath
Data
Datasource
Where is the data for this parameter?
Data defined here
Edit
Data defined in a variable
Variable check
Variable Id
Choose Variable
OK
Cancel

Parameter file_object

Data type string

Operation equals

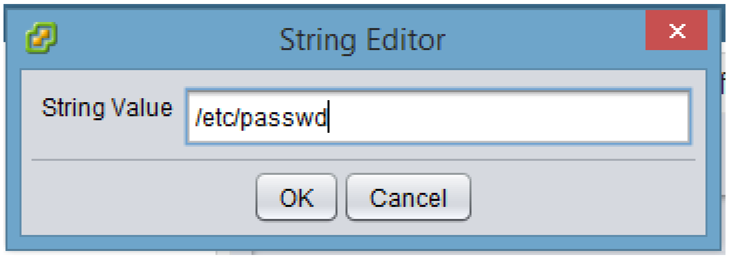
Nil □

● Defined Here ○ A Variable

Data

Variable check all

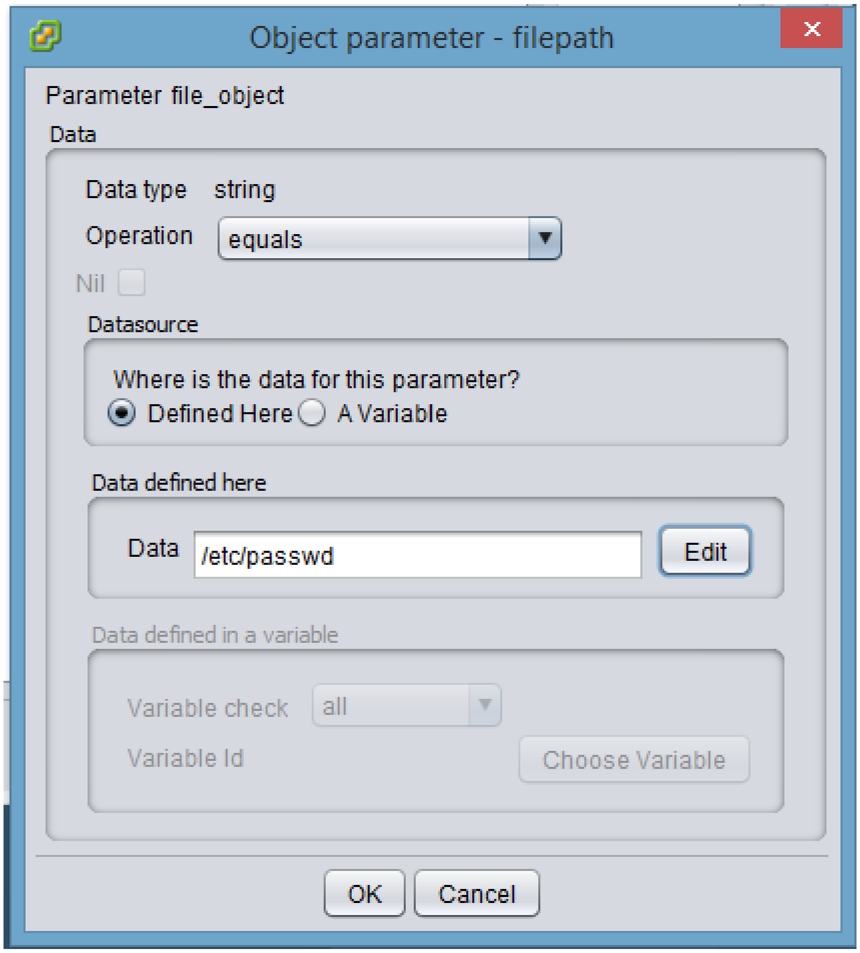
In this wizard, click on **Edit** and define the parameter value.



String Value /etc/passwd

Click **OK**.

The value gets set for `filepath`.



Object parameter - filepath

Parameter file_object

Data

Data type string

Operation equals

Nil □

Datasource

Where is the data for this parameter?

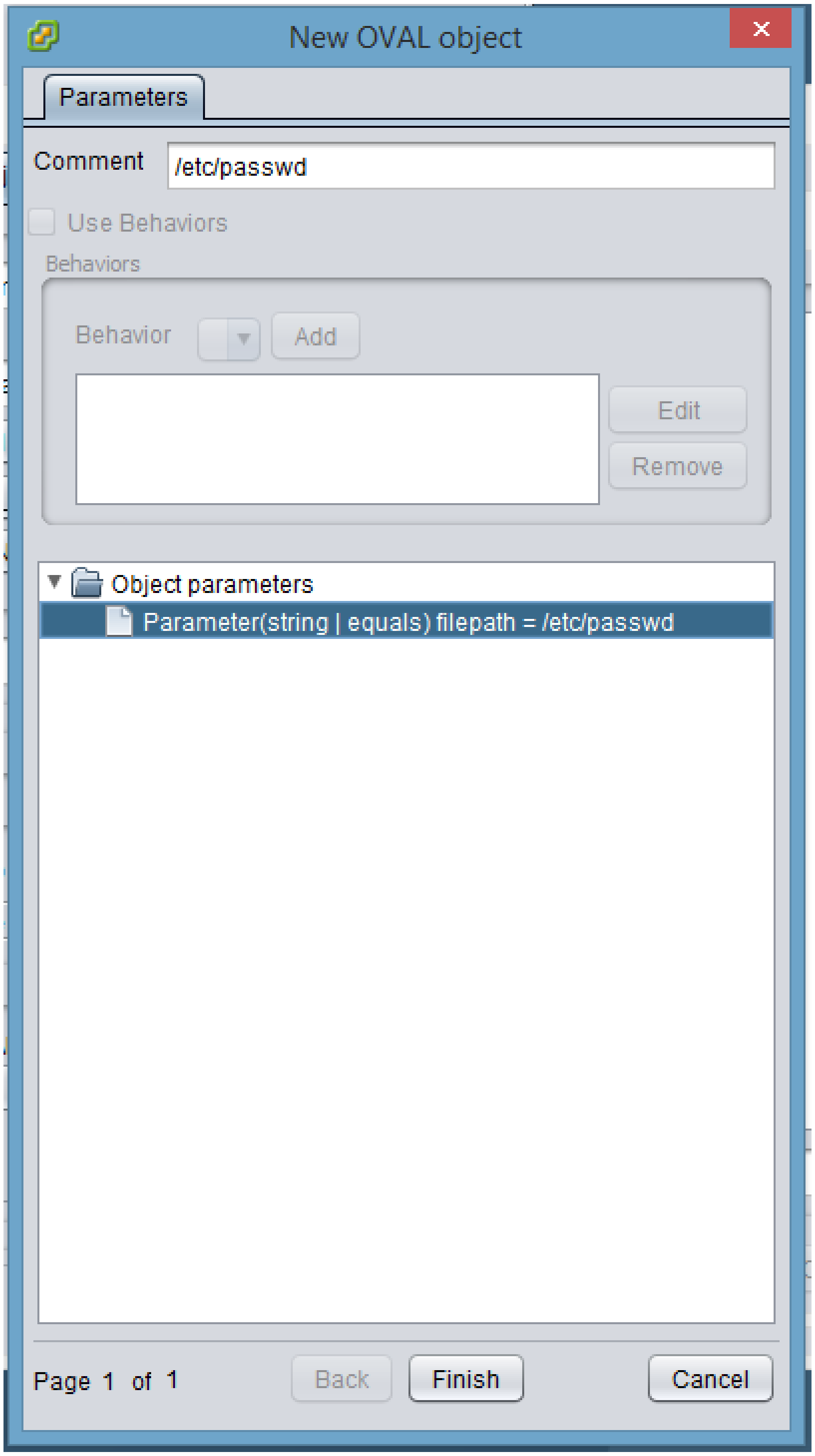
● Defined Here ○ A Variable

Data defined here

Data /etc/passwd

Click **OK**.

The object is now defined.



Comment: /etc/passwd

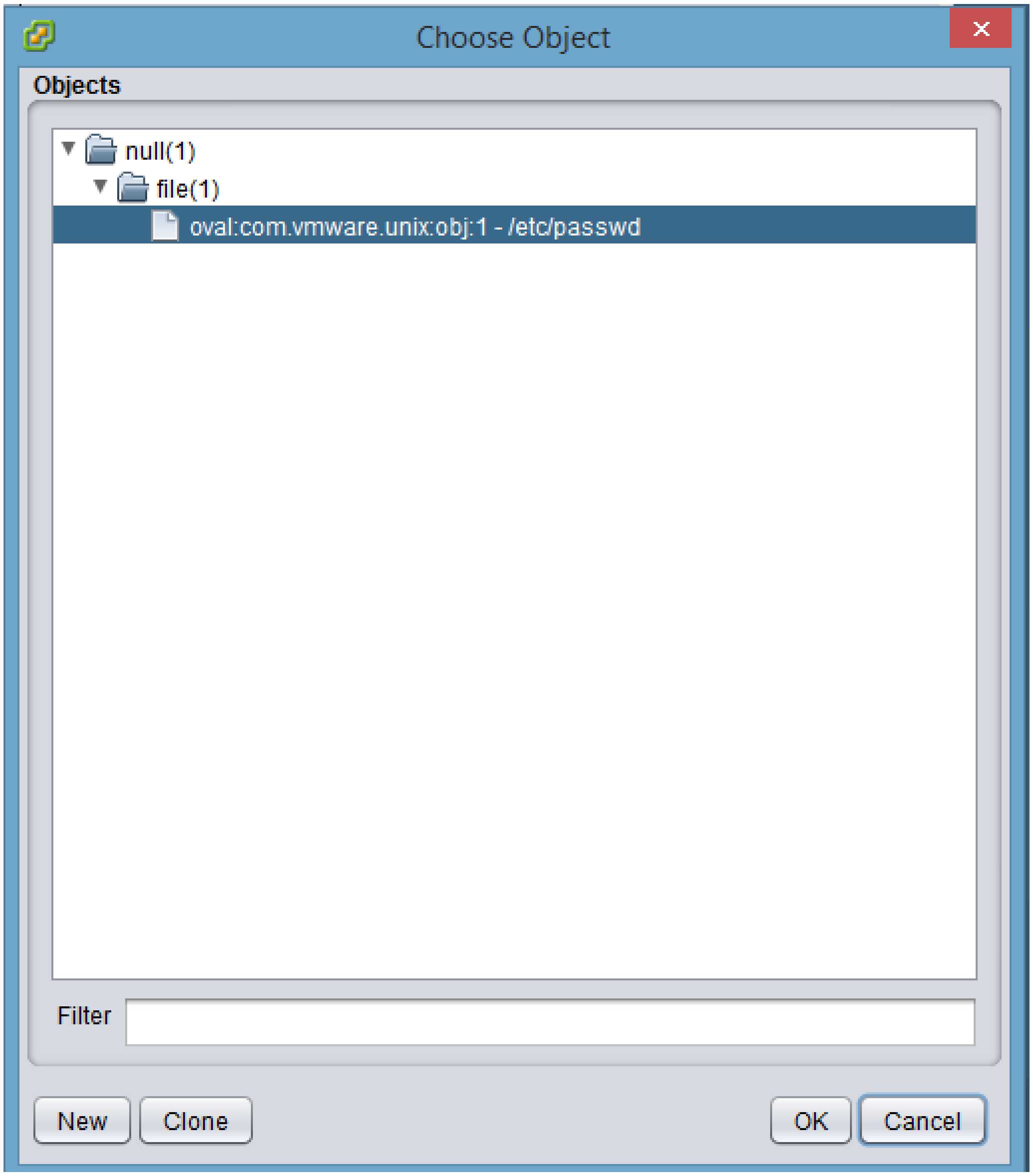
☐ Use Behaviors

Parameter(string | equals) filepath = /etc/passwd

Page 1 of 1

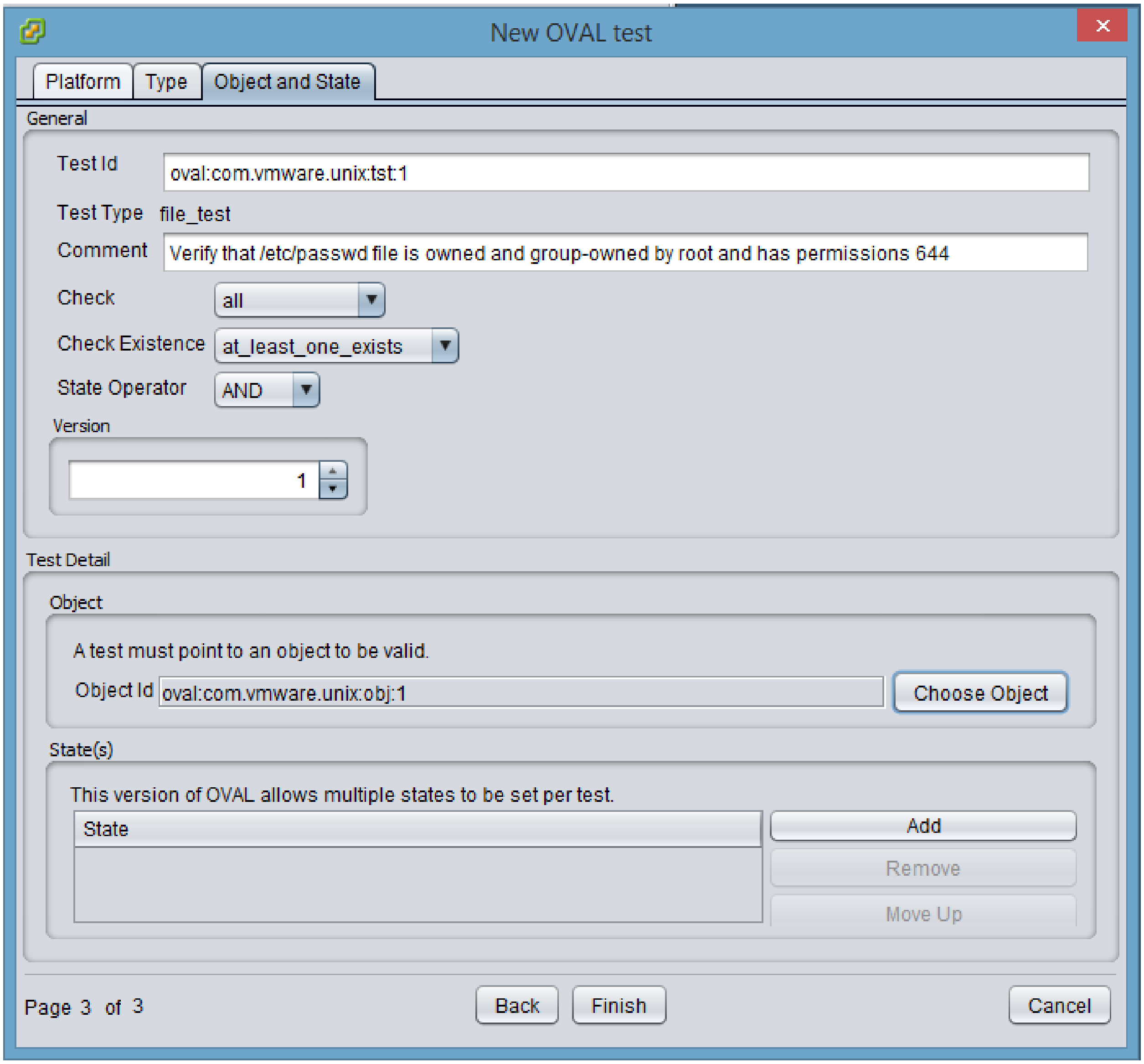
Click **Finish**.

Object is now chosen for the corresponding test.



Click **OK**.

You will now see that the object id is populated in the test.



General

Test Id: oval:com.vmware.unix:tst:1

Test Type: file_test

Comment: Verify that /etc/passwd file is owned and group-owned by root and has permissions 644

Check: all

Check Existence: at_least_one_exists

State Operator: AND

Version: 1

Test Detail

Object

A test must point to an object to be valid.

Object Id: oval:com.vmware.unix:obj:1

State(s)

This version of OVAL allows multiple states to be set per test.

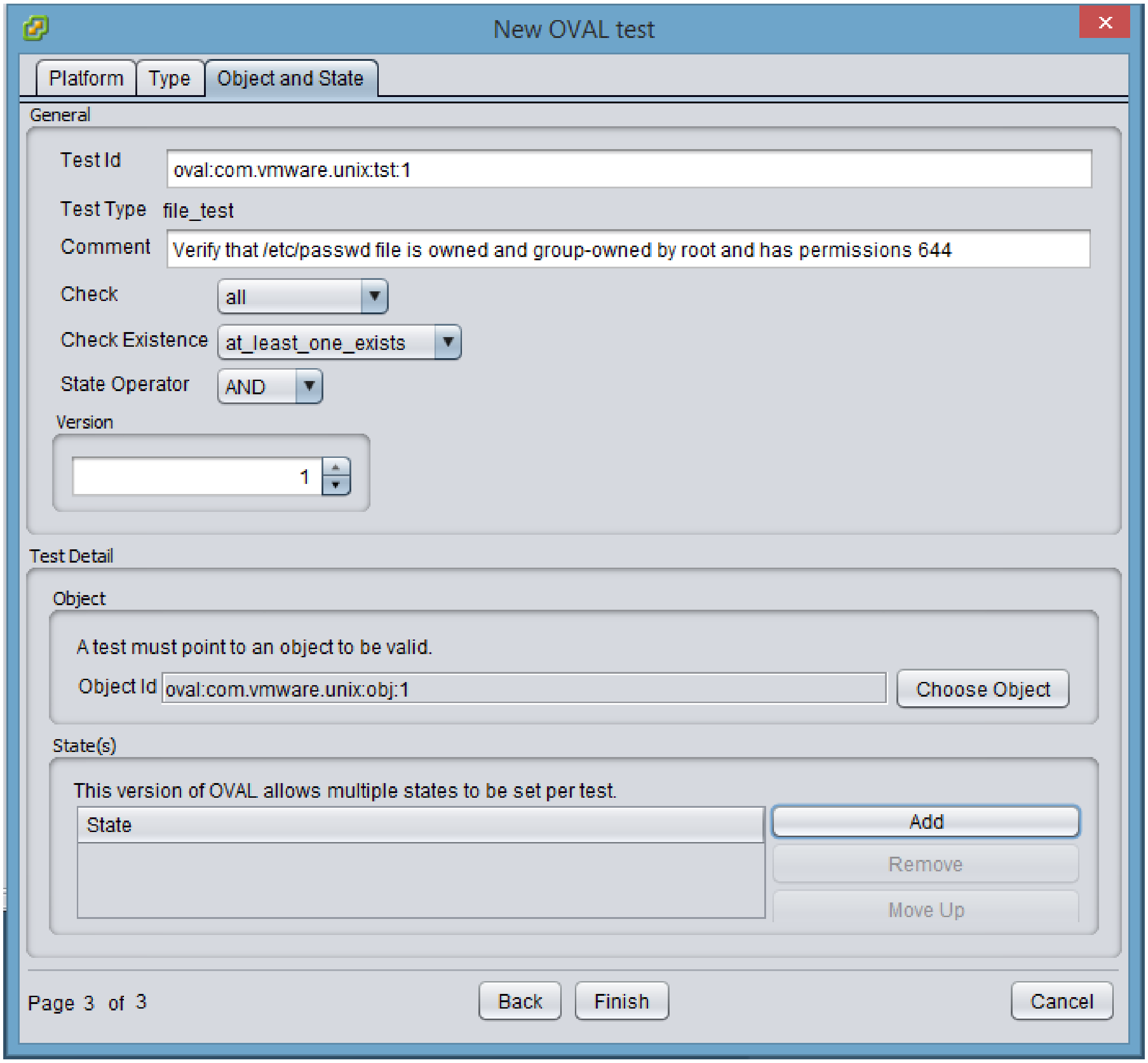
State

Now, let us follow similar steps and define states. You can define multiple states per test. In our example, the states to be defined are:

- group_id = 0
- user_id = 0
- Permissions 644

- uexec = 0
- gwrite = 0
- gexec = 0
- owrite = 0
- oexec = 0

In the state definition area, click on **Add**.



General

Test Id: oval:com.vmware.unix:tst:1

Test Type: file_test

Comment: Verify that /etc/passwd file is owned and group-owned by root and has permissions 644

Check: all

Check Existence: at_least_one_exists

State Operator: AND

Version: 1

Test Detail

Object

A test must point to an object to be valid.

Object Id: oval:com.vmware.unix:obj:1

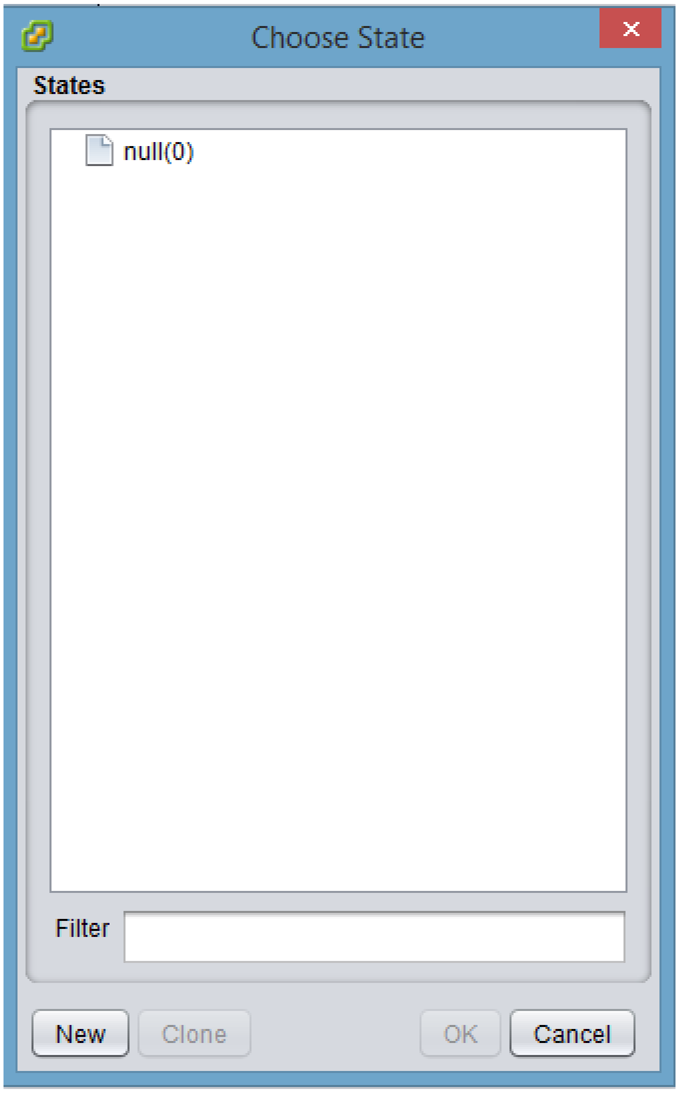
State(s)

This version of OVAL allows multiple states to be set per test.

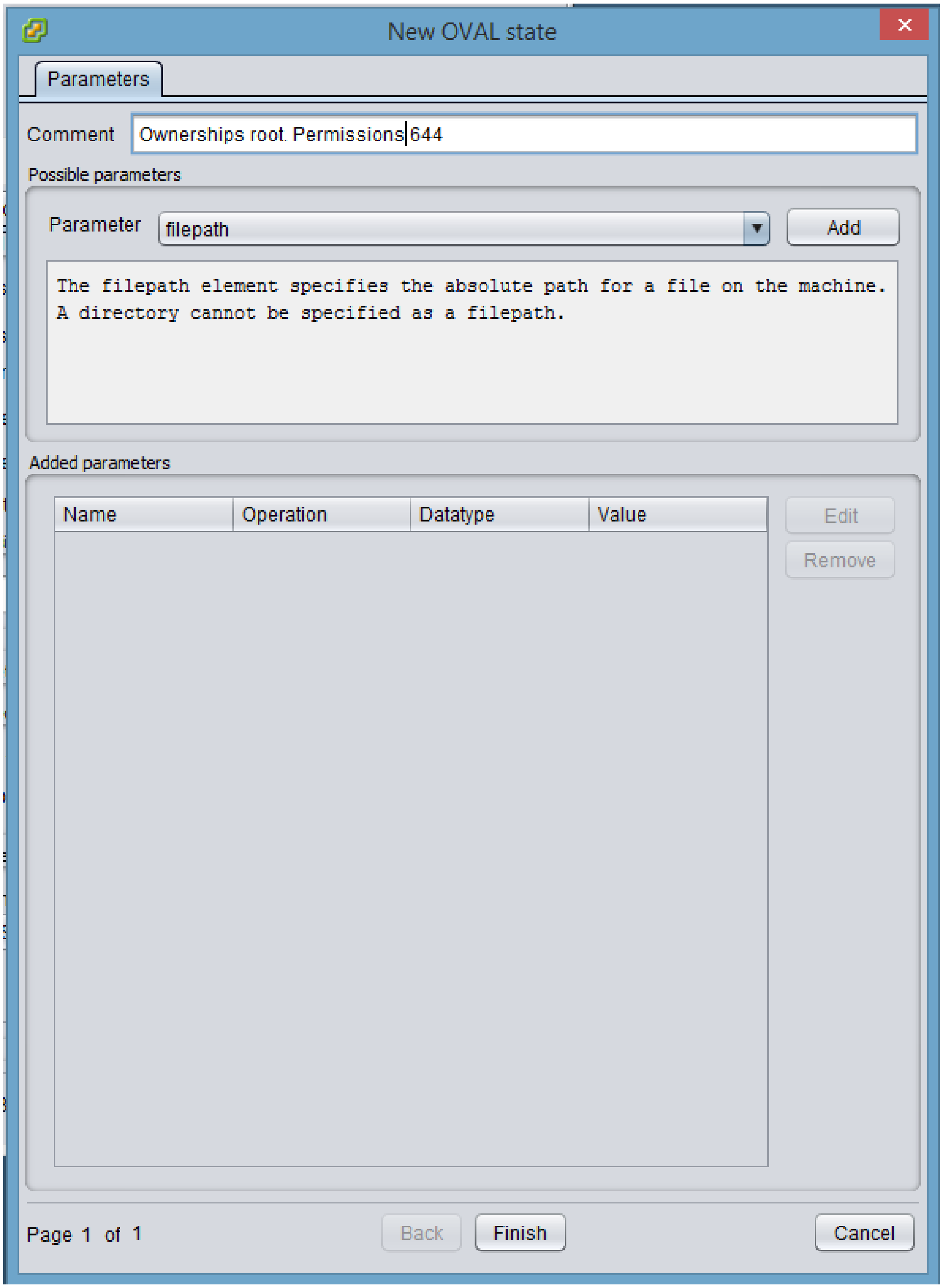
State

Page 3 of 3

The state wizard opens up. Click **New**.



Define a new state. First, give a **Comment**.



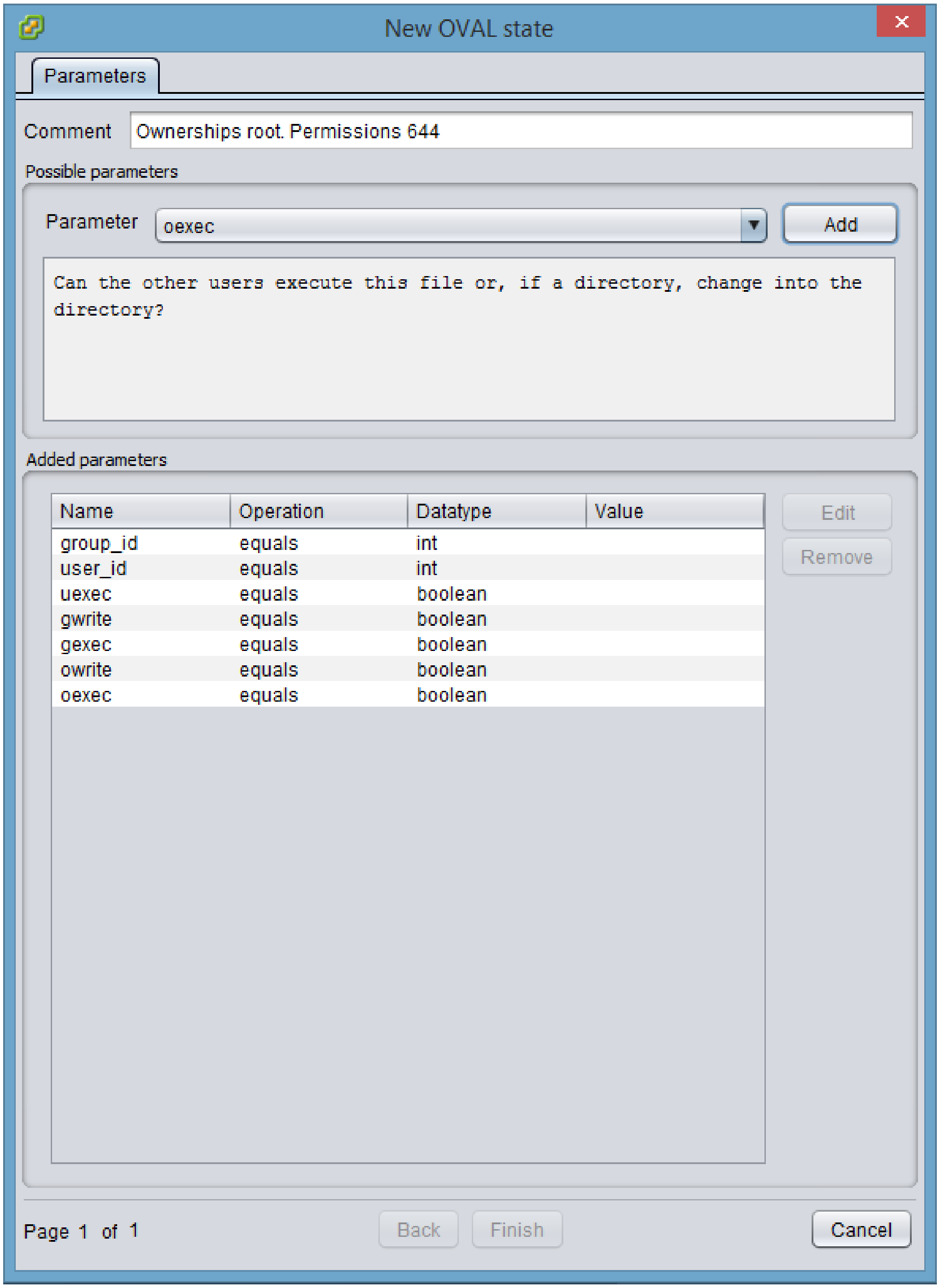
Comment: Ownerships root. Permissions 644

Parameter: filepath

The filepath element specifies the absolute path for a file on the machine. A directory cannot be specified as a filepath.

Name	Operation	Datatype	Value

Then, add all the state parameters needed from the parameter drop down list, one at a time.



Comment: Ownerships root. Permissions 644

Possible parameters

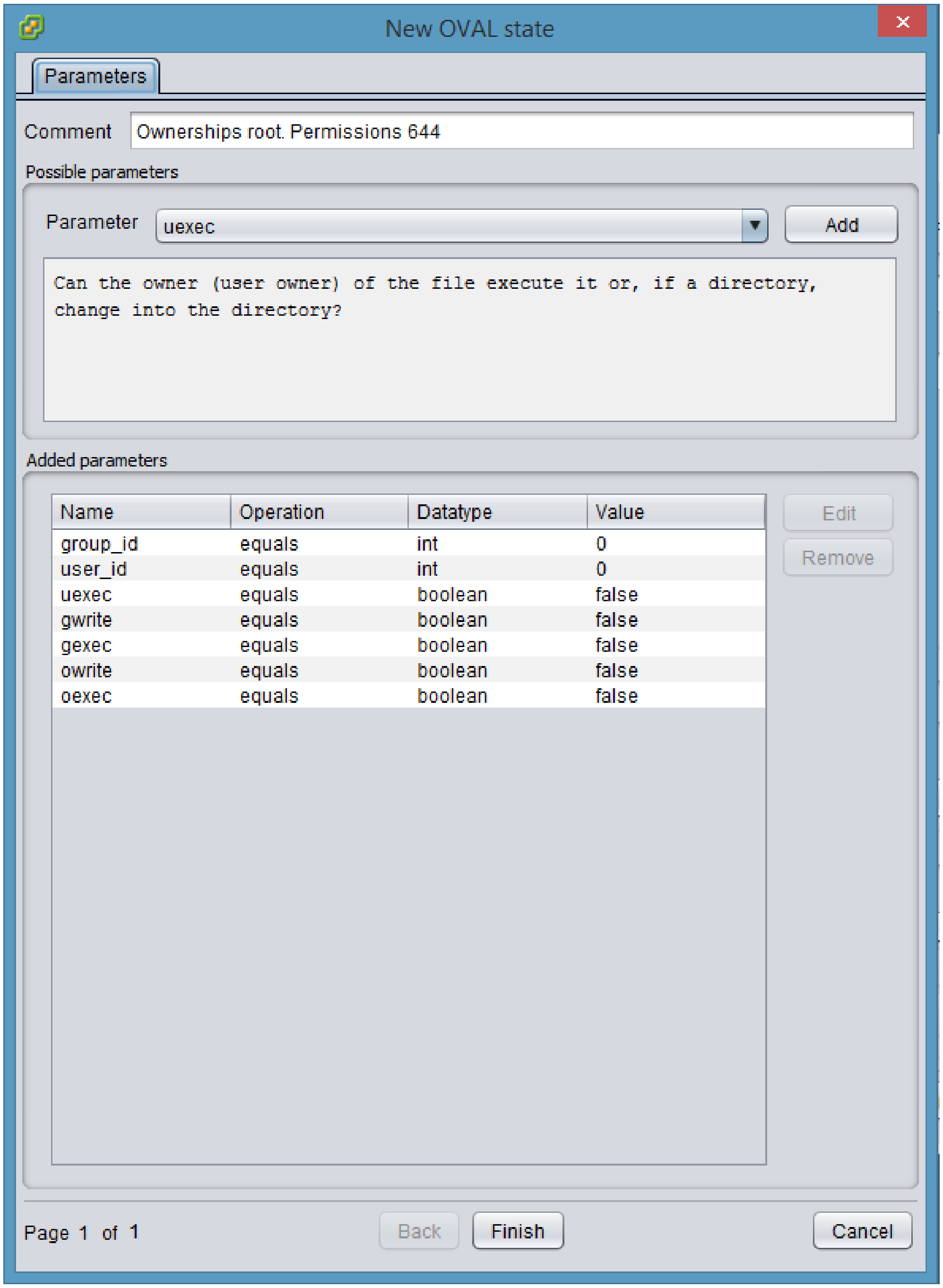
Parameter: oexec

Can the other users execute this file or, if a directory, change into the directory?

Added parameters

Name	Operation	Datatype	Value
group_id	equals	int	
user_id	equals	int	
uexec	equals	boolean	
gwrite	equals	boolean	
gexec	equals	boolean	
owrite	equals	boolean	
oexec	equals	boolean	

Then select each parameter and click **Edit** to set the desired value.



Comment: Ownerships root. Permissions 644

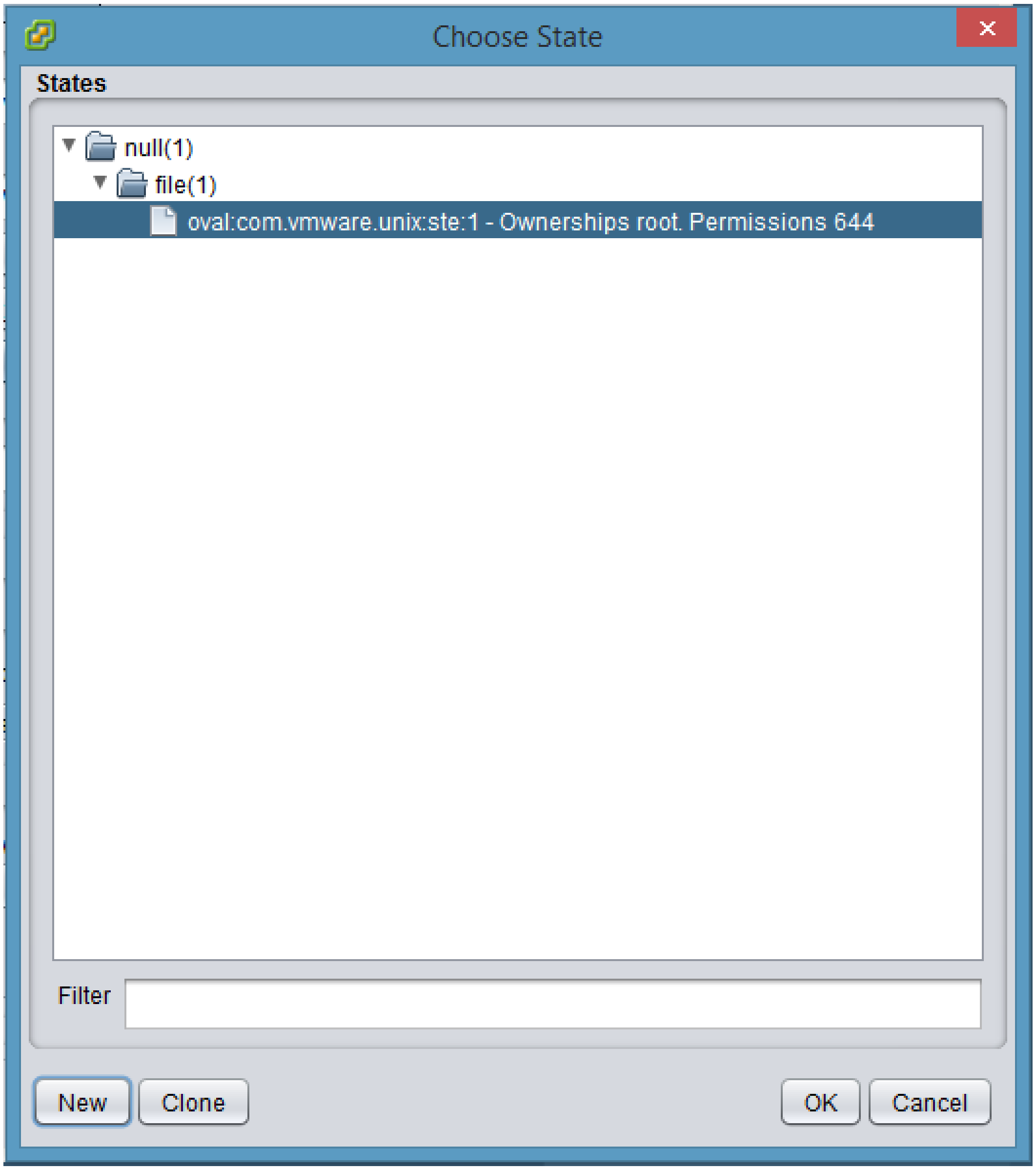
Parameter: uexec

Can the owner (user owner) of the file execute it or, if a directory, change into the directory?

Name	Operation	Datatype	Value
group_id	equals	int	0
user_id	equals	int	0
uexec	equals	boolean	false
gwrite	equals	boolean	false
gexec	equals	boolean	false
owrite	equals	boolean	false
oexec	equals	boolean	false

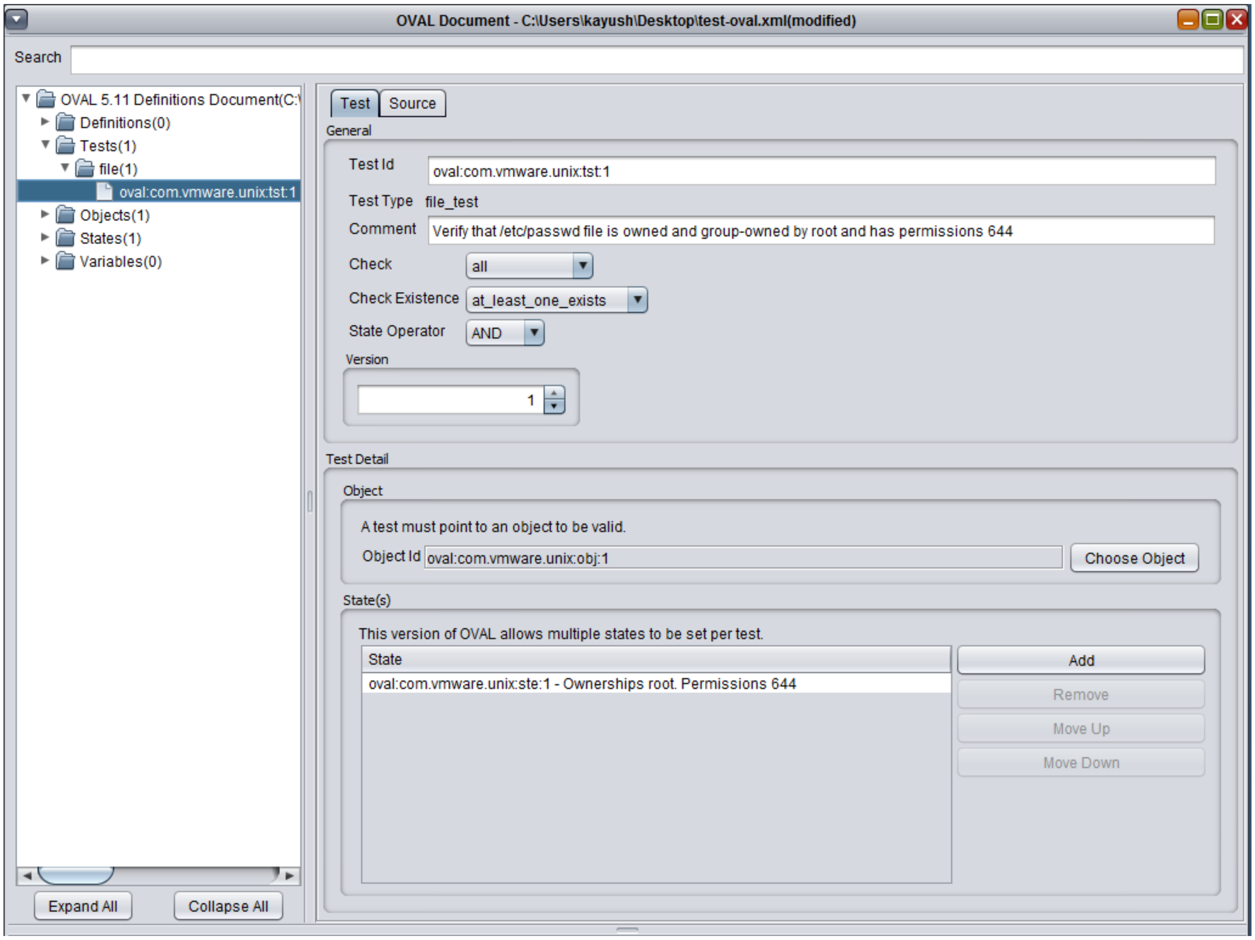
Click **Finish**.

Choose the state you just created.


Choose State
States
null(1)
file(1)
oval:com.vmware.unix:ste:1 - Ownerships root. Permissions 644
Filter
New
Clone
OK
Cancel

Click **OK**.

Now, the test is complete. We defined all the test elements.



OVAL Document - C:\Users\kayush\Desktop\test-oval.xml(modified)

Search

- OVAL 5.11 Definitions Document(C:\
  - Definitions(0)
  - Tests(1)
    - file(1)
      - oval:com.vmware.unix:tst:1
  - Objects(1)
  - States(1)
  - Variables(0)

Test | Source

General

Test Id: oval:com.vmware.unix:tst:1

Test Type: file_test

Comment: Verify that /etc/passwd file is owned and group-owned by root and has permissions 644

Check: all

Check Existence: at_least_one_exists

State Operator: AND

Version: 1

Test Detail

Object

A test must point to an object to be valid.

Object Id: oval:com.vmware.unix:obj:1

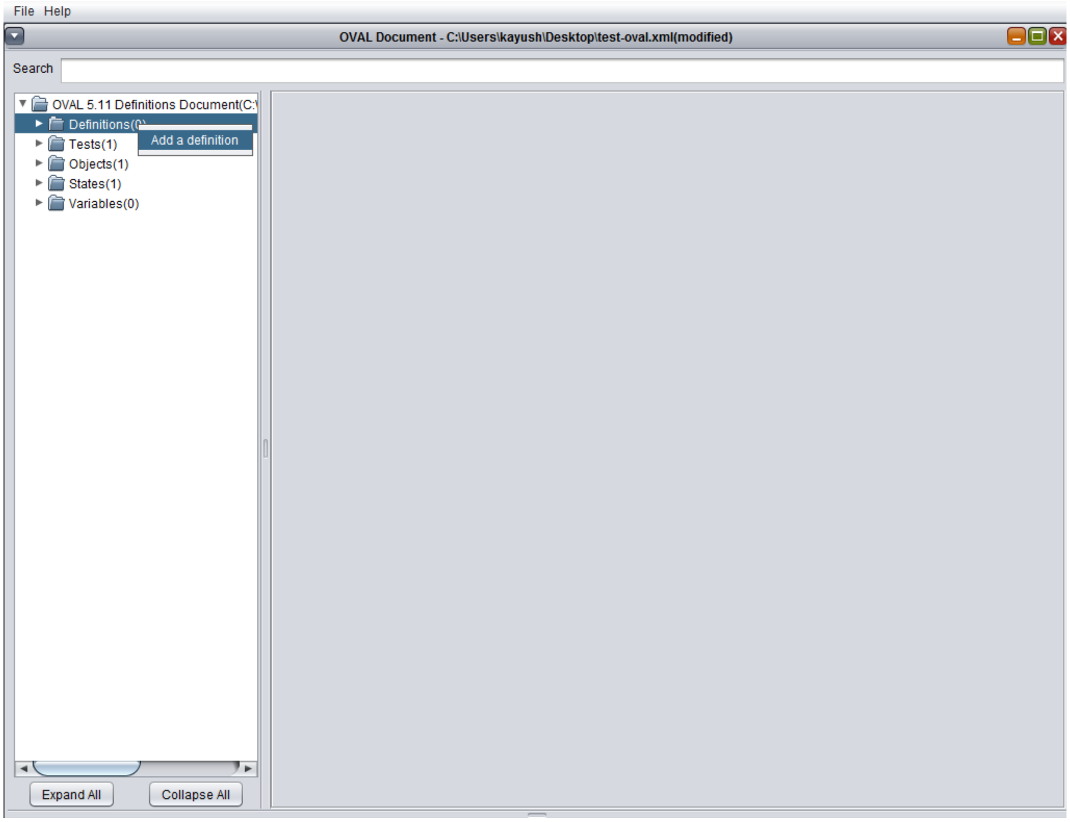
State(s)

This version of OVAL allows multiple states to be set per test.

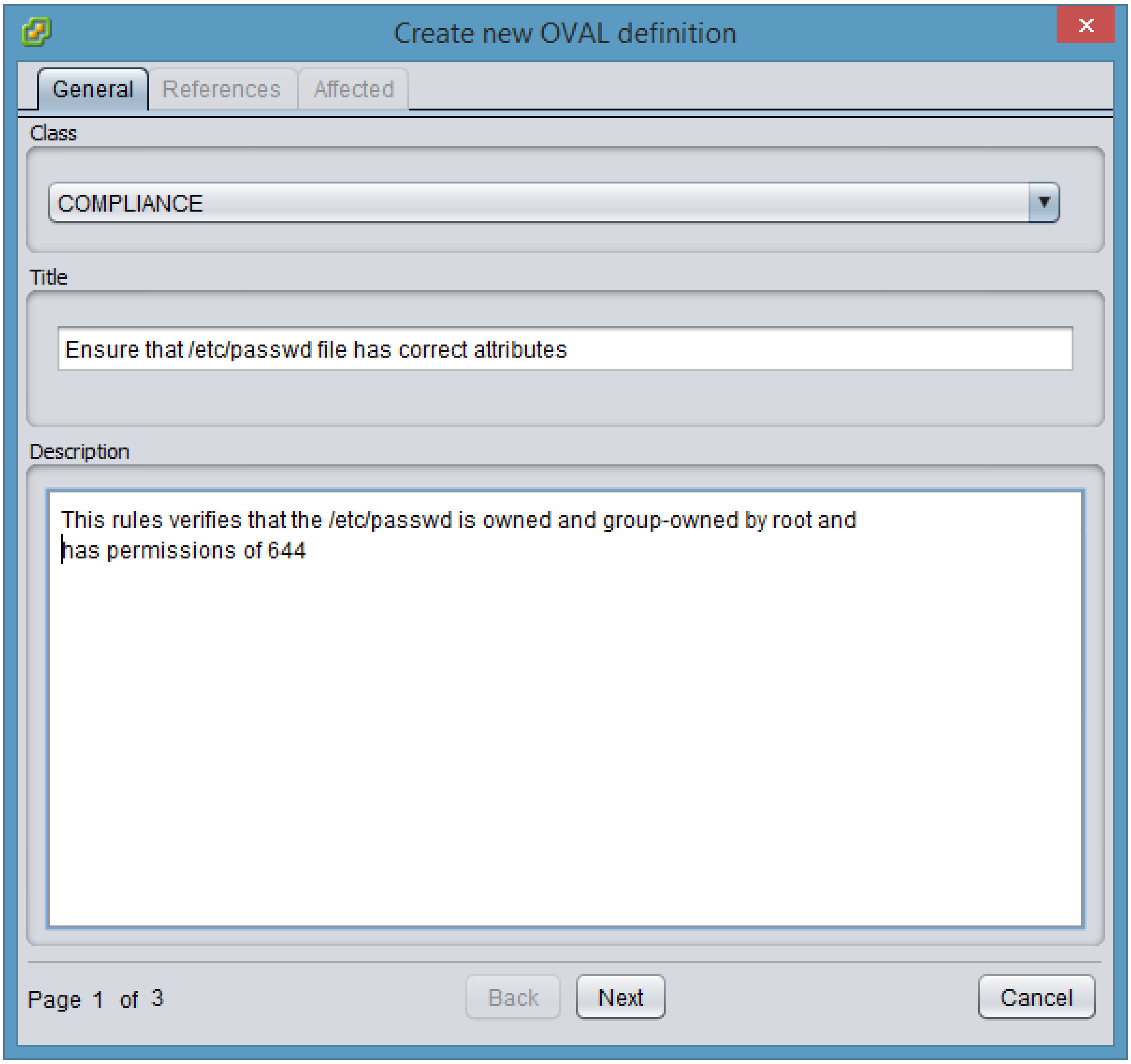
State
oval:com.vmware.unix:ste:1 - Ownerships root. Permissions 644

The next step is to link the test to an OVAL definition.

Right click on Definitions and select **Add a definition**.



This would open OVAL definition wizard. In this wizard, choose the definition **Class** from the drop down and provide a suitable **Title** and **Description** for the definition.



Class: COMPLIANCE

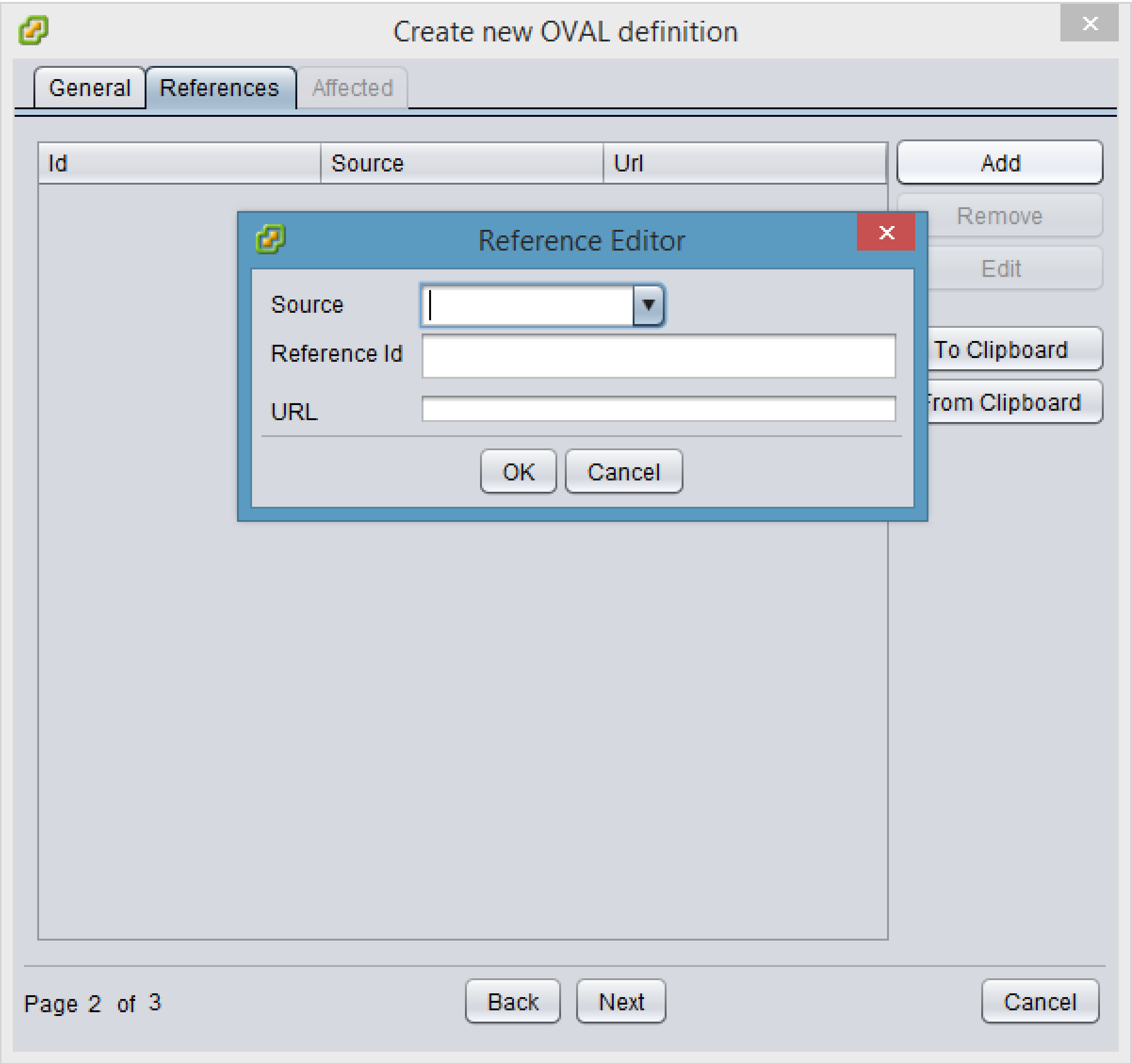
Title: Ensure that /etc/passwd file has correct attributes

Description: This rules verifies that the /etc/passwd is owned and group-owned by root and has permissions of 644

Page 1 of 3

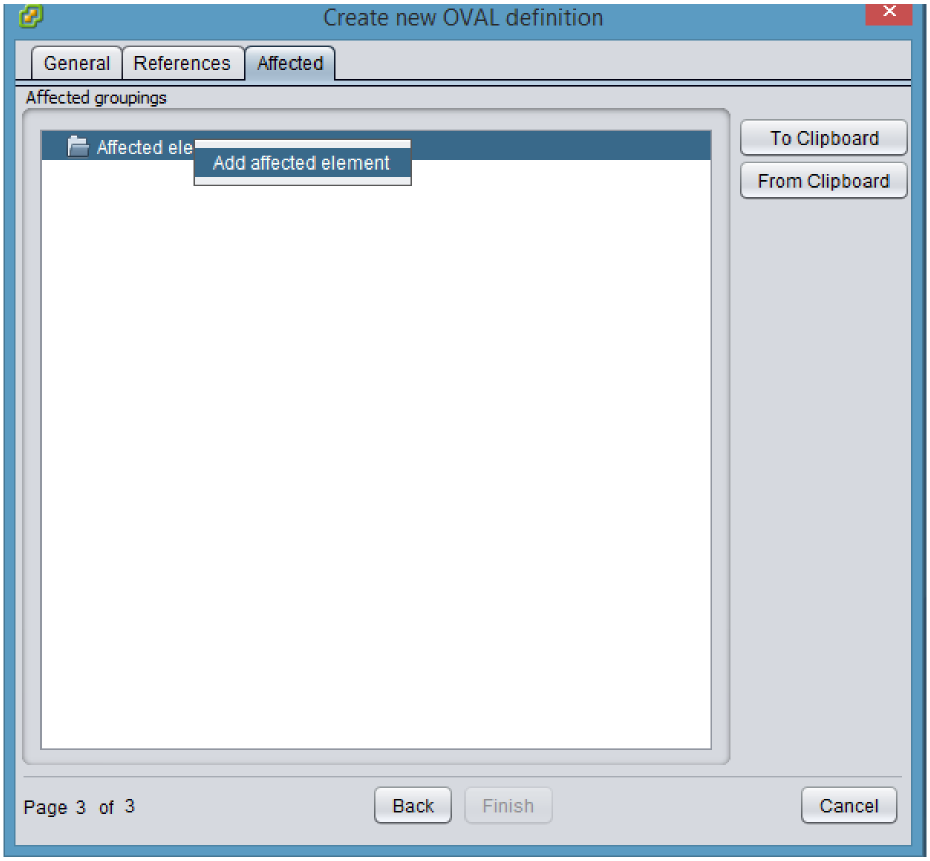
Click **Next**.

Add references if you want to. This is optional.

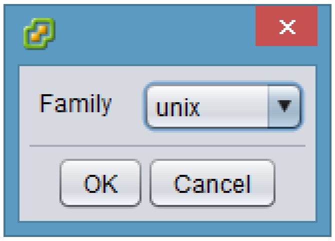


Click **Next**.

Add affected platforms and products. Right click on **Affected elements** and select **Add affected element**.

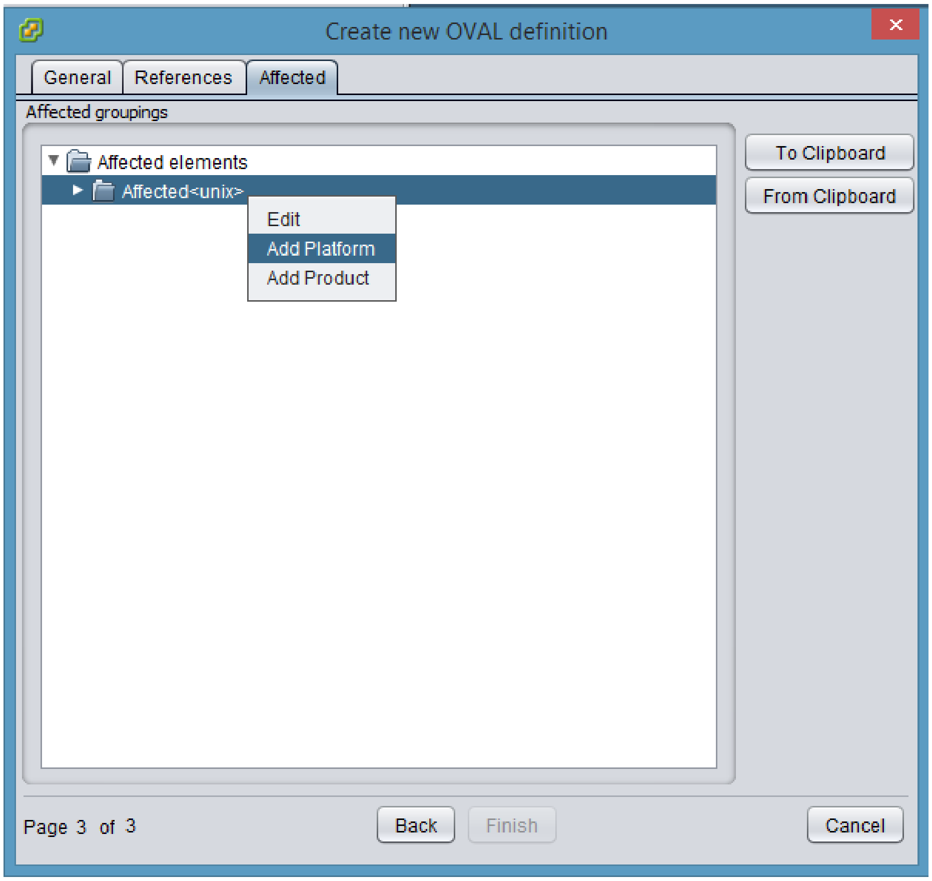


In our example, we will choose Unix.

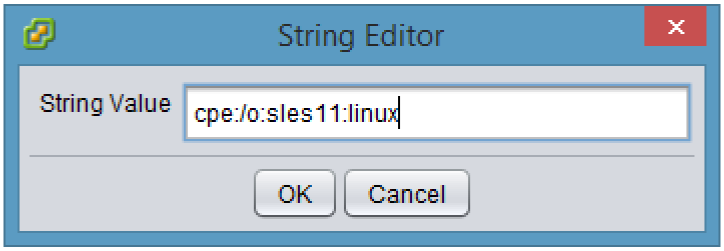


Click **OK**.

Then, define a specific platform or product. Right click on **Affected<unix>** and choose the target as needed.

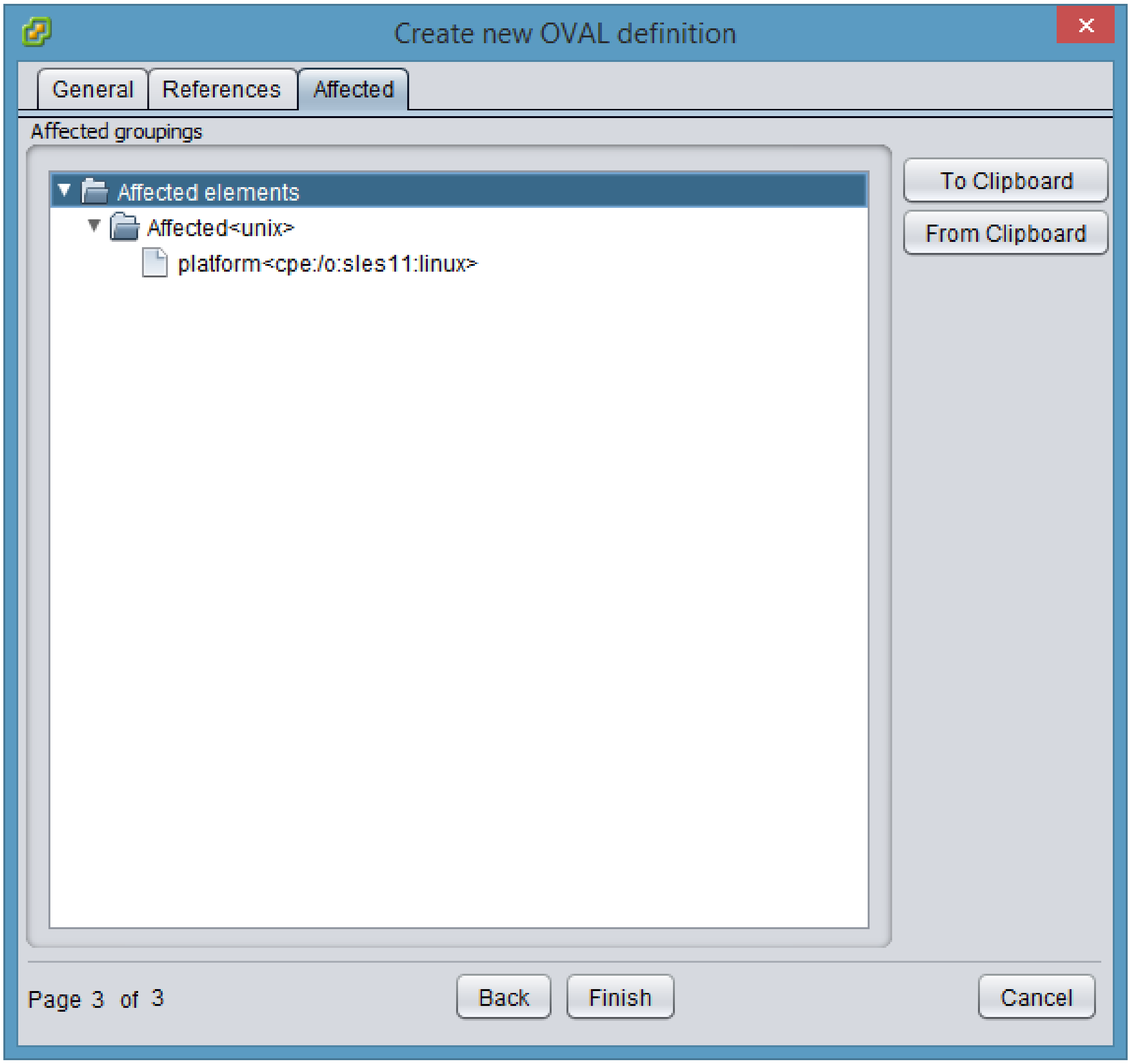


For our example, let us give SLES 11 as the target platform. An entry for it will look like below.



String Value: cpe:/o:sles11:linux

Click **OK**.



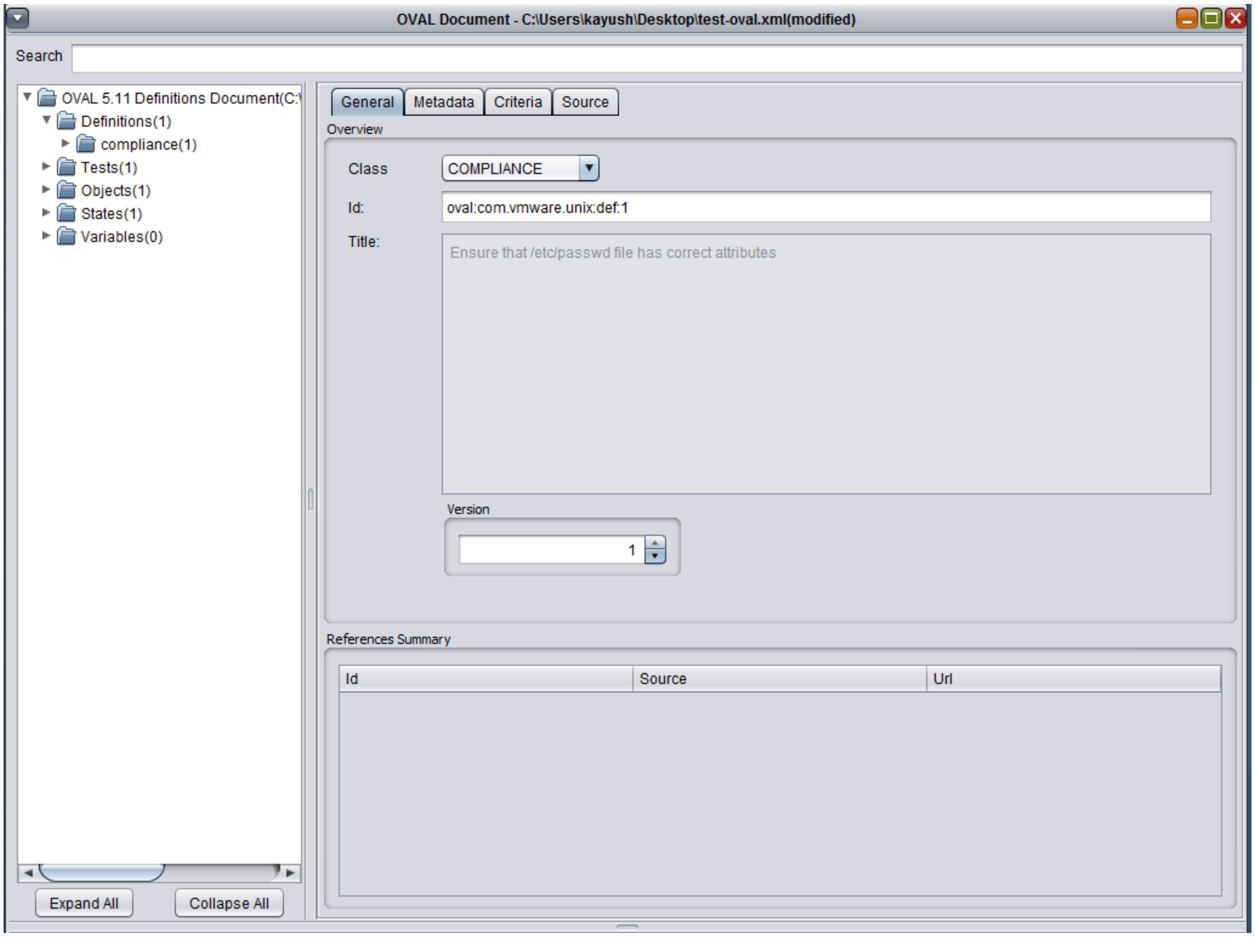
Affected groupings

- Affected elements
  - Affected<unix>
    - platform<cpe:/o:sles11:linux>

Page 3 of 3

Click **Finish**.

This adds a new OVAL definition. Now, we will link this definition with a test. Definitions can have one or more tests. In our example, we will associate this definition with the test we created in earlier steps.



Class: COMPLIANCE

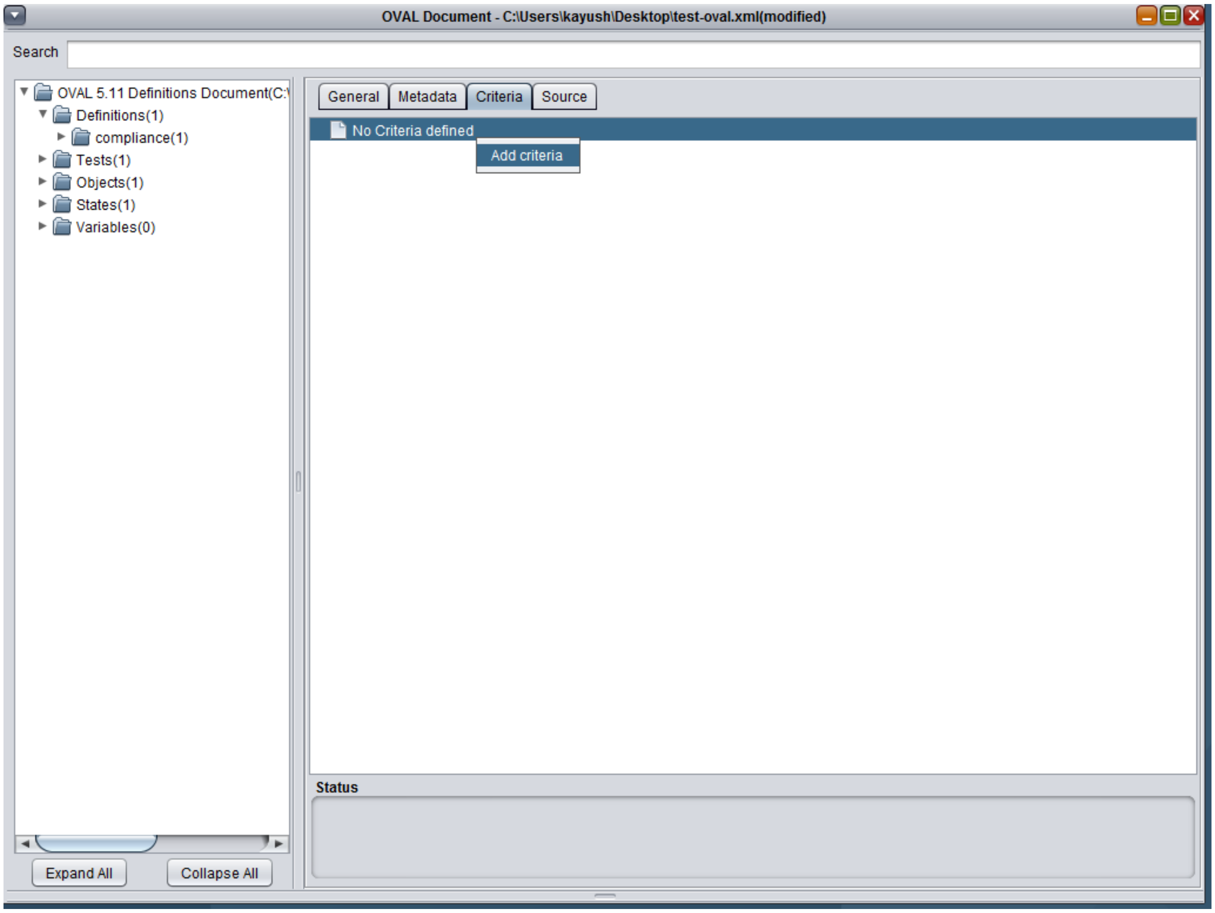
Id: oval:com.vmware.unix:def:1

Title: Ensure that /etc/passwd file has correct attributes

Version: 1

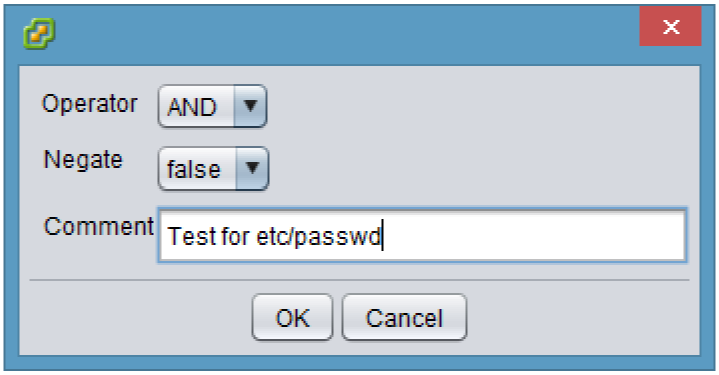
Id	Source	Url

Click on **Criteria** tab.

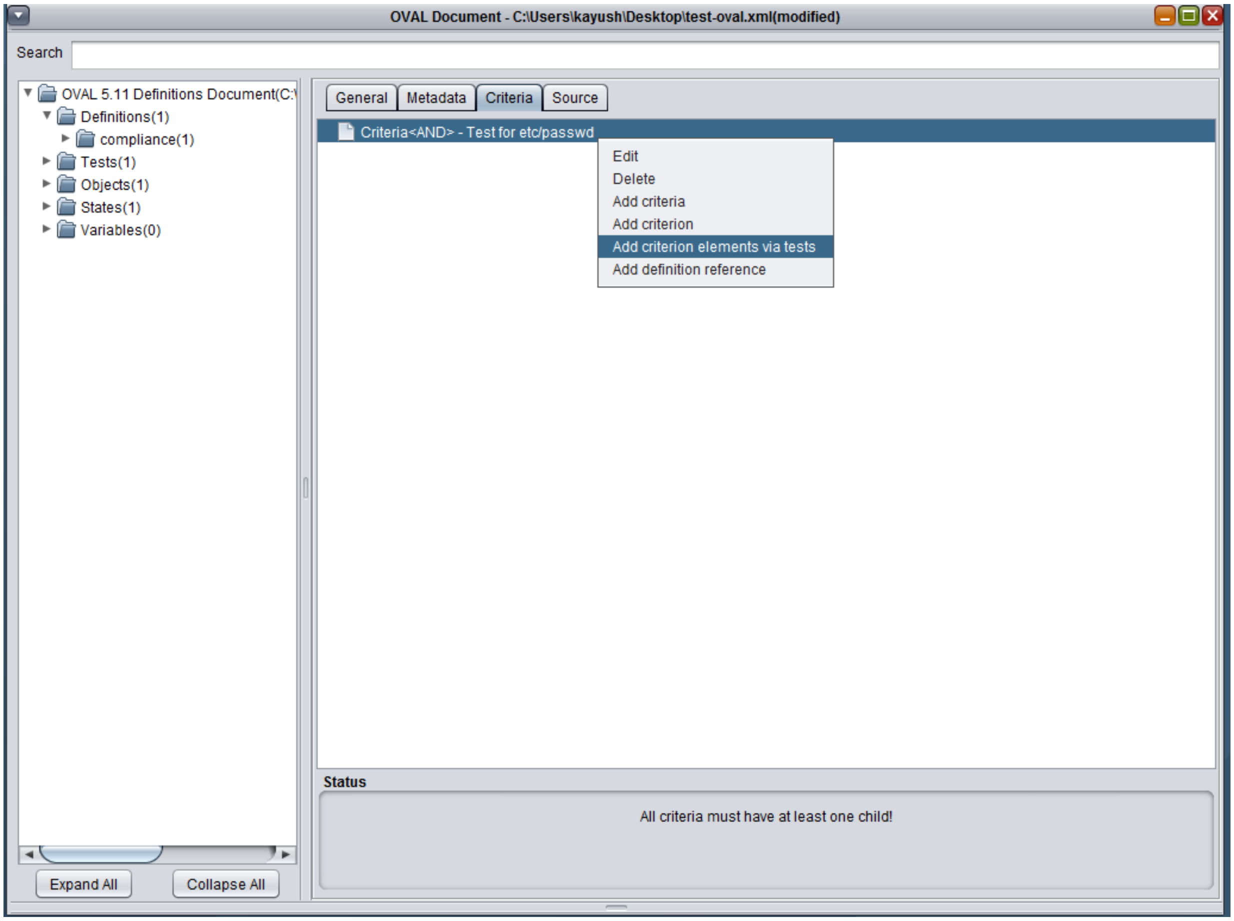


Right click on **No Criteria defined** and then choose **Add criteria**.

Choose criteria operators (defaults are preset) and provide a **Comment** and then click **OK**.

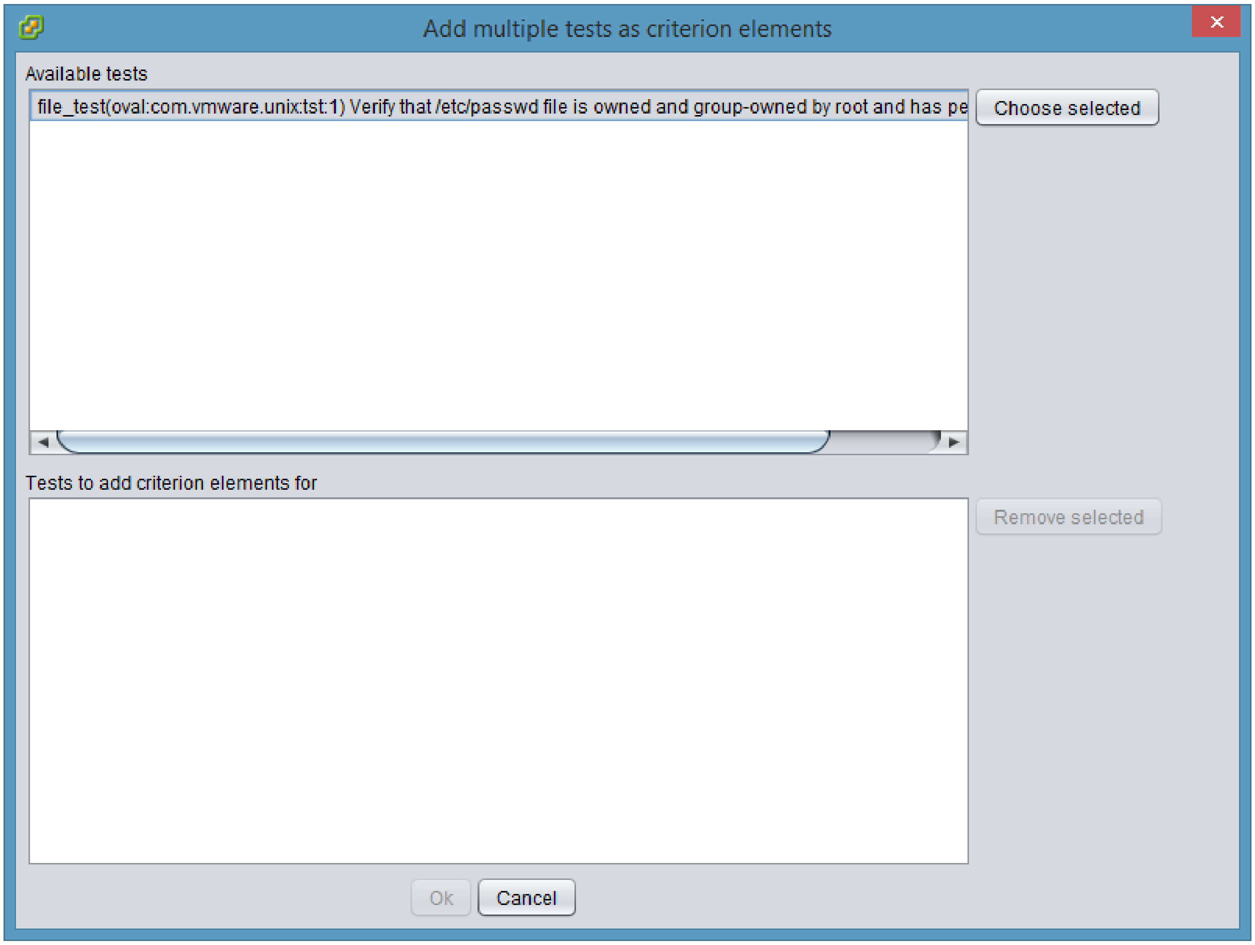


Operator AND
Negate false
Comment Test for etc/passwd



All criteria must have at least one child!

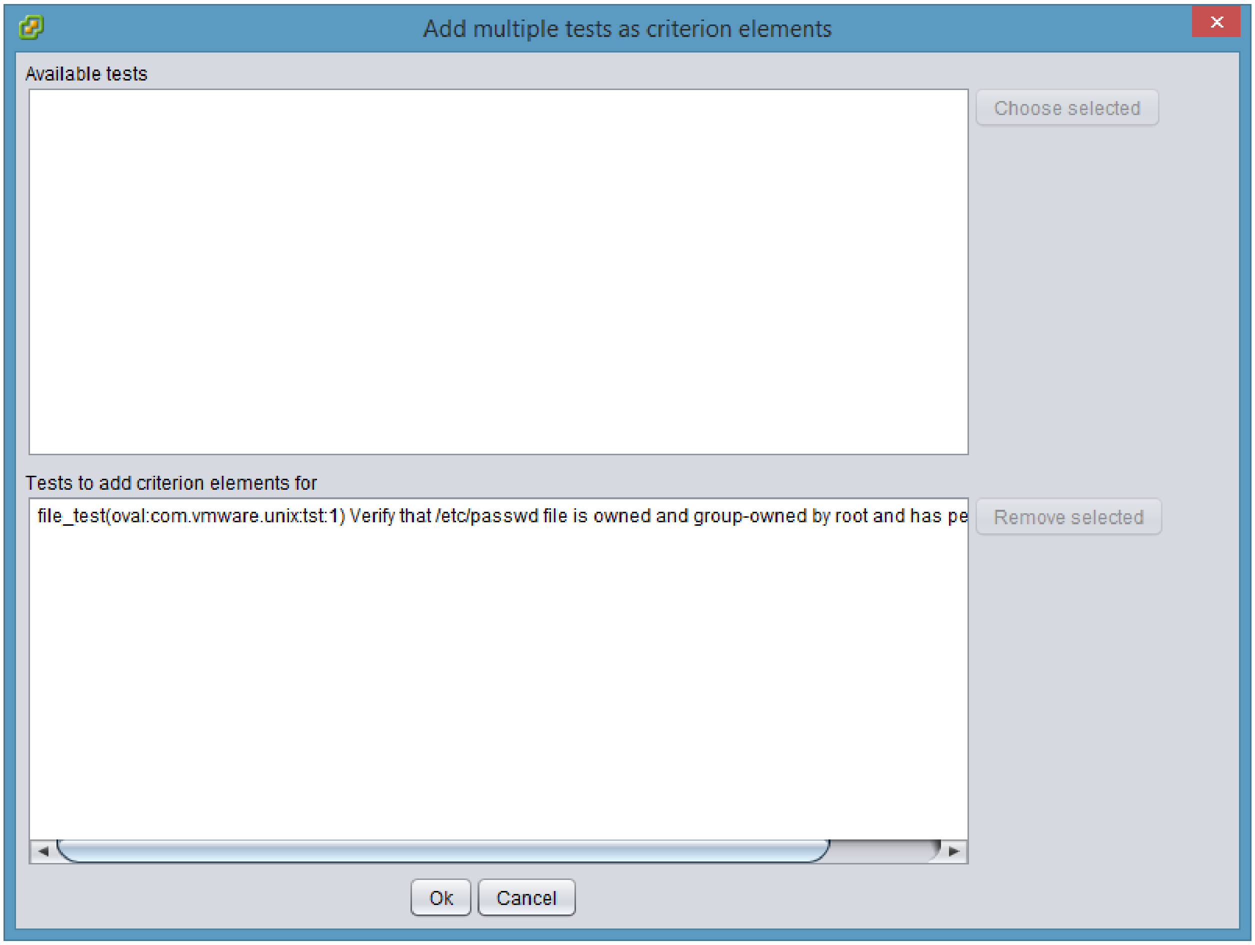
Now right click on **Criteria** to choose **Add criterion elements via tests**.



Add multiple tests as criterion elements
Available tests
Choose selected
Tests to add criterion elements for
Remove selected
Ok
Cancel

file_test(oval:com.vmware.unix:tst:1) Verify that /etc/passwd file is owned and group-owned by root and has pe

Select the desired **Test** and click on **Choose selected**.



Add multiple tests as criterion elements

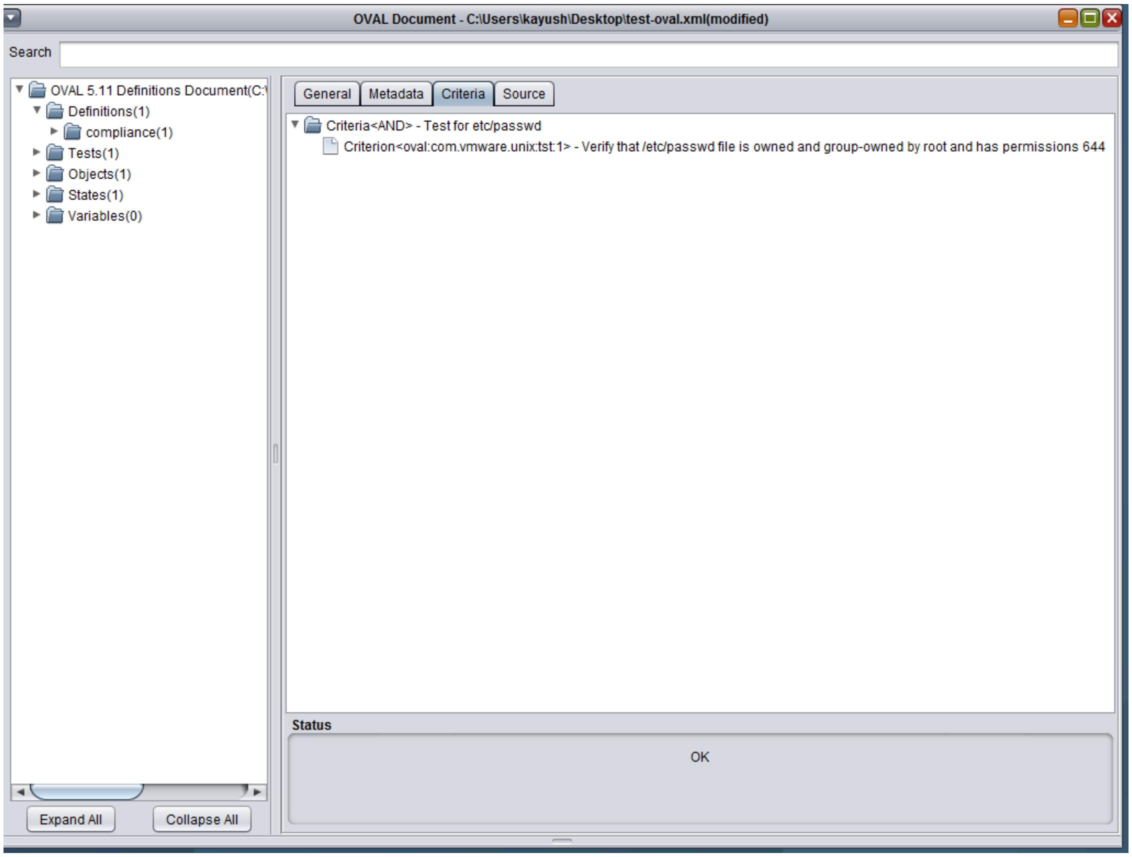
Available tests

Tests to add criterion elements for

file_test(oval:com.vmware.unix:tst:1) Verify that /etc/passwd file is owned and group-owned by root and has pe

Click **Ok**.

This will add the test to the definition.



OVAL Document - C:\Users\kayush\Desktop\test-oval.xml(modified)

Criteria<AND> - Test for etc/passwd

Criterion<oval:com.vmware.unix:tst:1> - Verify that /etc/passwd file is owned and group-owned by root and has permissions 644

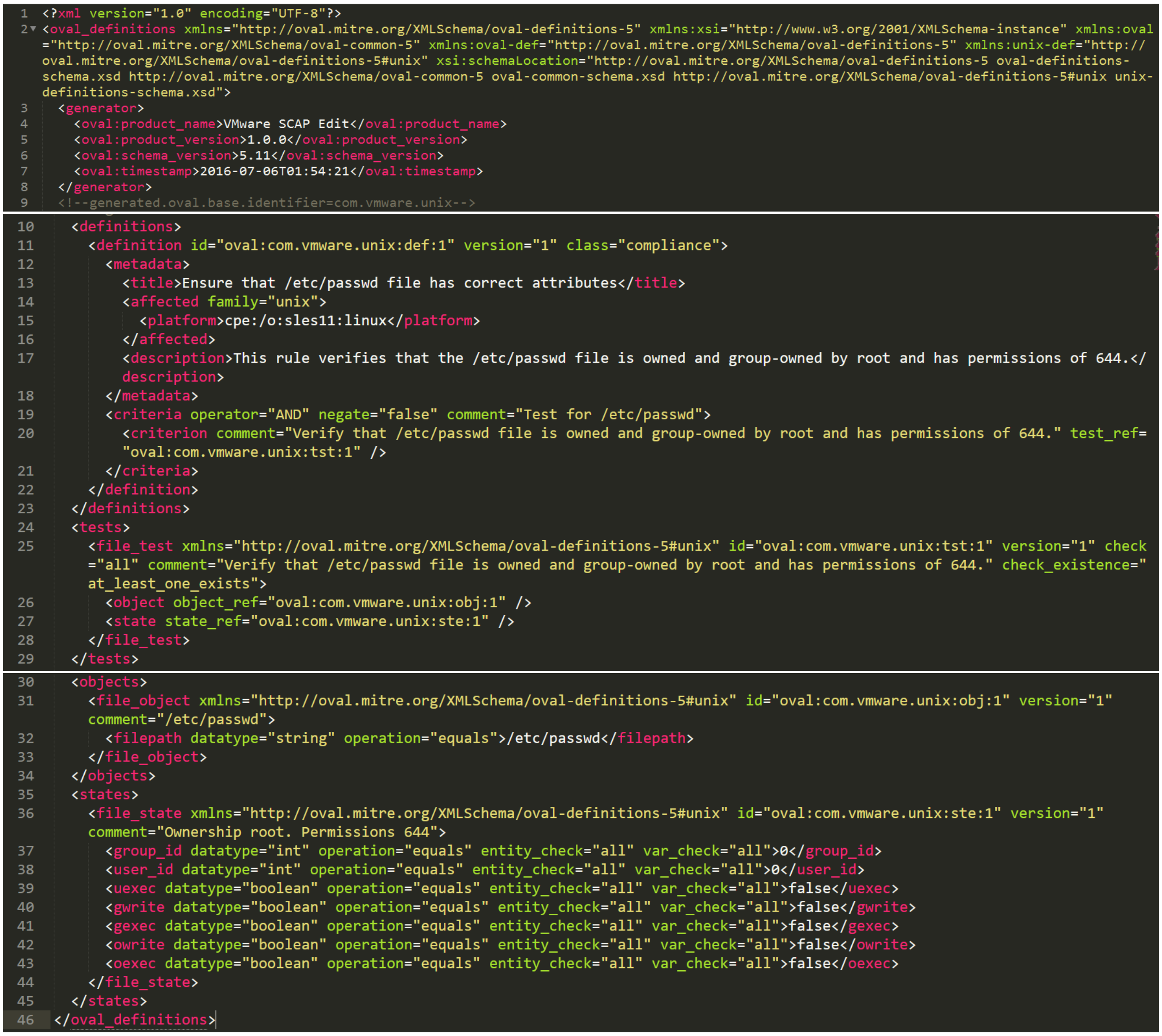
Status

OK

Go to **File** menu and choose **Save**. Done!

This completes adding your first OVAL definition and the required elements.

The generated XML file looks like below:

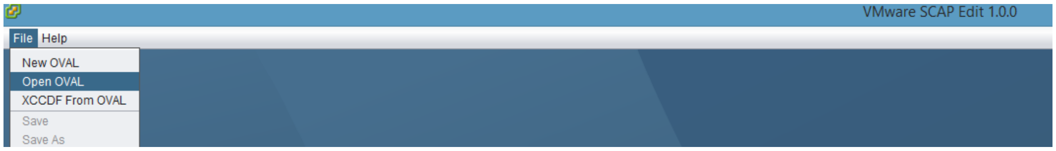


```xml
<?xml version="1.0" encoding="UTF-8"?>
<oval_definitions xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:oval="http://oval.mitre.org/XMLSchema/oval-common-5" xmlns:oval-def="http://oval.mitre.org/XMLSchema/oval-definitions-5" xmlns:unix-def="http://oval.mitre.org/XMLSchema/oval-definitions-5#unix" xsi:schemaLocation="http://oval.mitre.org/XMLSchema/oval-definitions-5 oval-definitions-schema.xsd http://oval.mitre.org/XMLSchema/oval-common-5 oval-common-schema.xsd http://oval.mitre.org/XMLSchema/oval-definitions-5#unix unix-definitions-schema.xsd">
  <generator>
    <oval:product_name>VMware SCAP Edit</oval:product_name>
    <oval:product_version>1.0.0</oval:product_version>
    <oval:schema_version>5.11</oval:schema_version>
    <oval:timestamp>2016-07-06T01:54:21</oval:timestamp>
  </generator>
  <!--generated.oval.base.identifier=com.vmware.unix-->
  <definitions>
    <definition id="oval:com.vmware.unix:def:1" version="1" class="compliance">
      <metadata>
        <title>Ensure that /etc/passwd file has correct attributes</title>
        <affected family="unix">
          <platform>cpe:/o:sles11:linux</platform>
        </affected>
        <description>This rule verifies that the /etc/passwd file is owned and group-owned by root and has permissions of 644.</description>
      </metadata>
      <criteria operator="AND" negate="false" comment="Test for /etc/passwd">
        <criterion comment="Verify that /etc/passwd file is owned and group-owned by root and has permissions of 644." test_ref="oval:com.vmware.unix:tst:1" />
      </criteria>
    </definition>
  </definitions>
  <tests>
    <file_test xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#unix" id="oval:com.vmware.unix:tst:1" version="1" check="all" comment="Verify that /etc/passwd file is owned and group-owned by root and has permissions of 644." check_existence="at_least_one_exists">
      <object object_ref="oval:com.vmware.unix:obj:1" />
      <state state_ref="oval:com.vmware.unix:ste:1" />
    </file_test>
  </tests>
  <objects>
    <file_object xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#unix" id="oval:com.vmware.unix:obj:1" version="1" comment="/etc/passwd">
      <filepath datatype="string" operation="equals">/etc/passwd</filepath>
    </file_object>
  </objects>
  <states>
    <file_state xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#unix" id="oval:com.vmware.unix:ste:1" version="1" comment="Ownership root. Permissions 644">
      <group_id datatype="int" operation="equals" entity_check="all" var_check="all">0</group_id>
      <user_id datatype="int" operation="equals" entity_check="all" var_check="all">0</user_id>
      <uexec datatype="boolean" operation="equals" entity_check="all" var_check="all">false</uexec>
      <gwrite datatype="boolean" operation="equals" entity_check="all" var_check="all">false</gwrite>
      <gexec datatype="boolean" operation="equals" entity_check="all" var_check="all">false</gexec>
      <owrite datatype="boolean" operation="equals" entity_check="all" var_check="all">false</owrite>
      <oexec datatype="boolean" operation="equals" entity_check="all" var_check="all">false</oexec>
    </file_state>
  </states>
</oval_definitions>
```

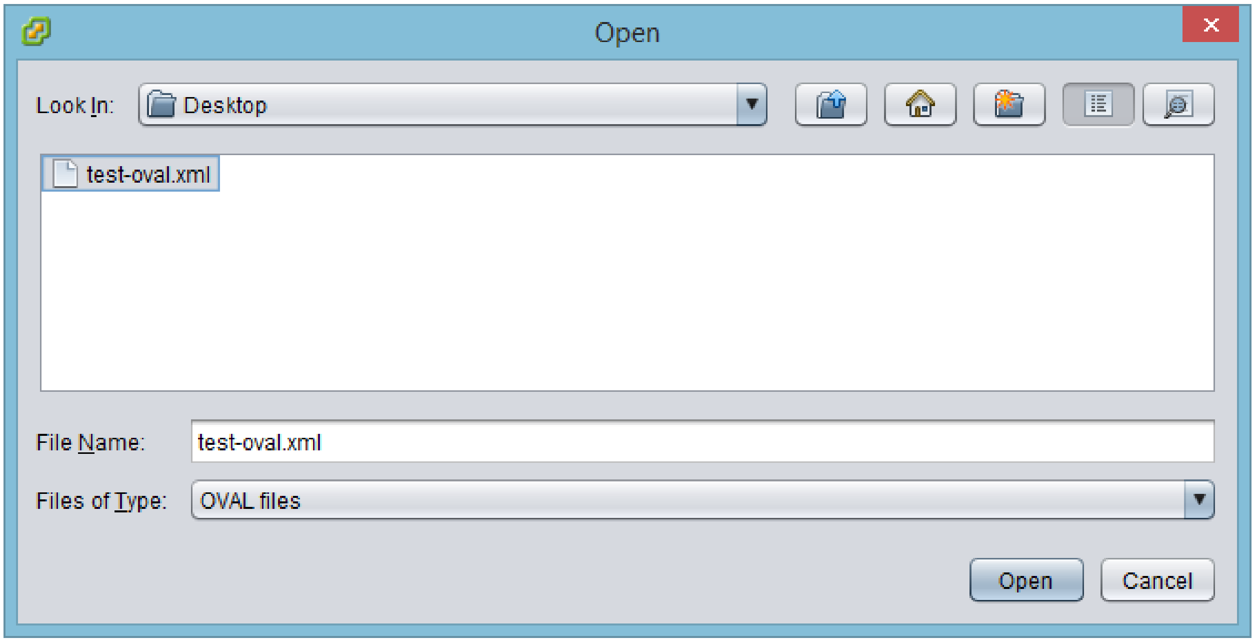
Now, you can run this OVAL definition using a tool such as VMware STIG compliance tool on the target platform.

Opening an existing OVAL file

To open an existing OVAL file, go to **File → Open OVAL**.



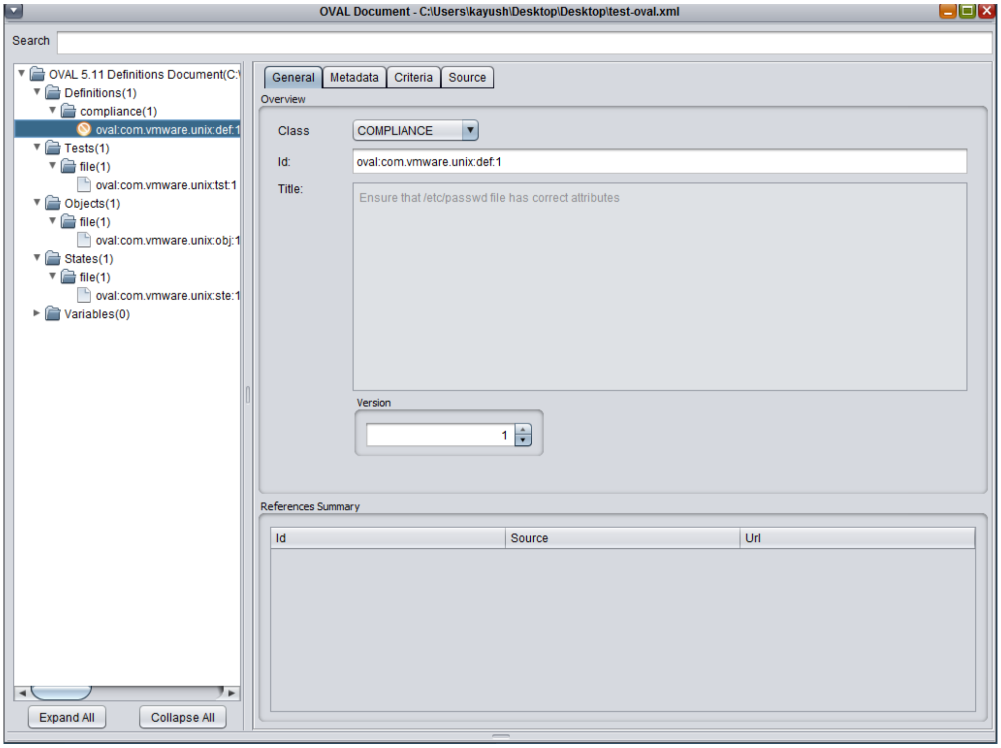
Browse the file, select it and click **Open**.



File Name: test-oval.xml

Files of Type: OVAL files

This would open up the OVAL file that we just created.



OVAL Document - C:\Users\kayush\Desktop\Desktop\test-oval.xml

Search

- OVAL 5.11 Definitions Document(C:\
  - Definitions(1)
    - compliance(1)
      - oval:com.vmware.unix:def:1
  - Tests(1)
    - file(1)
      - oval:com.vmware.unix:tst:1
  - Objects(1)
    - file(1)
      - oval:com.vmware.unix:obj:1
  - States(1)
    - file(1)
      - oval:com.vmware.unix:ste:1
  - Variables(0)

Overview

Class: COMPLIANCE

Id: oval:com.vmware.unix:def:1

Title: Ensure that /etc/passwd file has correct attributes

Version: 1

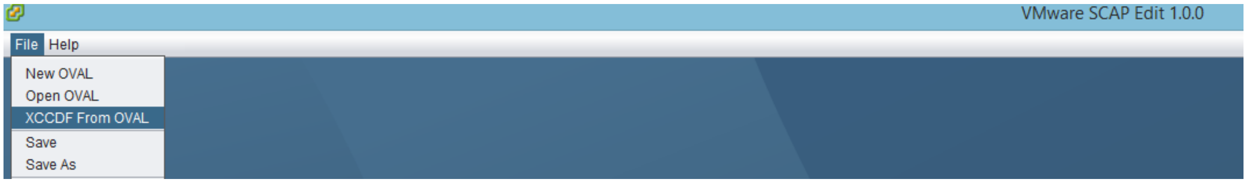
References Summary

Id	Source	Url

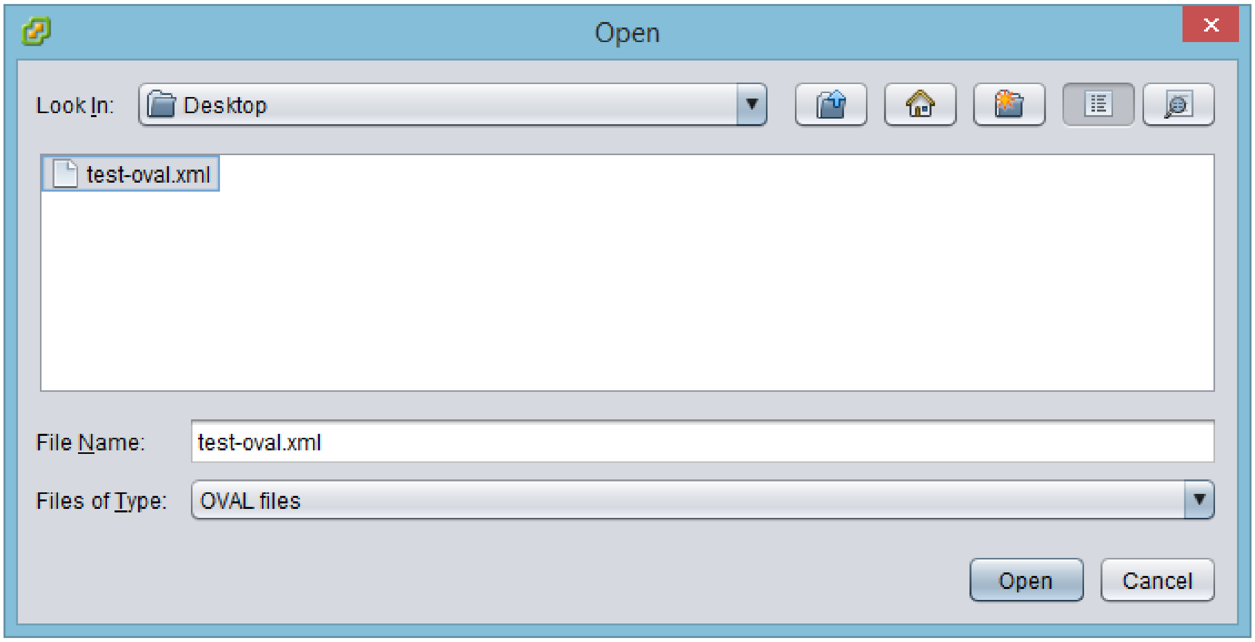
You may then add, edit or remove any OVAL elements to this file as desired. Do not forget to **Save** the file each time you have made changes to it that should be persisted.

Creating XCCDF file from an existing OVAL file

Using the tool, you can directly create a minimal XCCDF 1.2 file from an existing OVAL file. To do this, just launch the tool and click on **File → XCCDF from OVAL**.



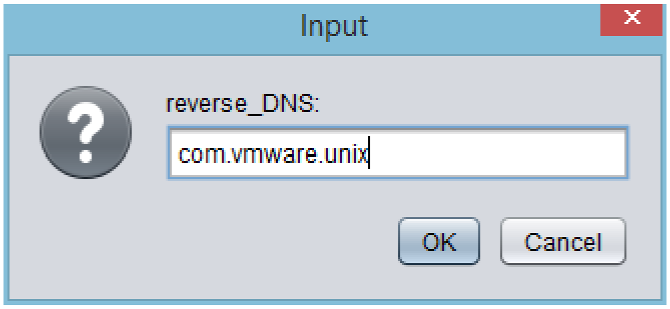
Browse the file, select it and click **Open**.



File Name: test-oval.xml

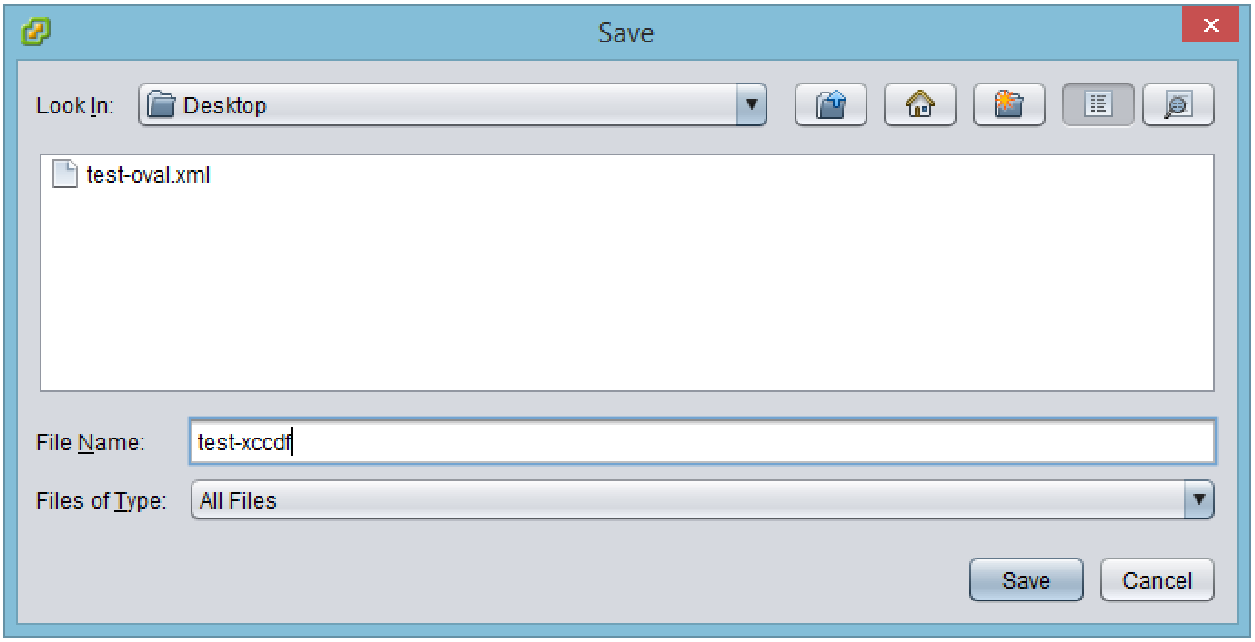
Files of Type: OVAL files

Provide the desired reverse DNS string and click **OK**.



reverse_DNS: com.vmware.unix

Provide desired XCCDF file name and click **Save**.



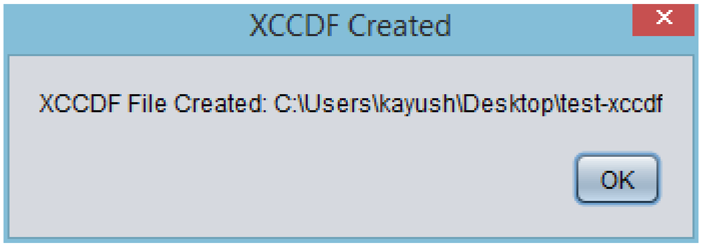
Look In: Desktop

test-oval.xml

File Name: test-xccdf

Files of Type: All Files

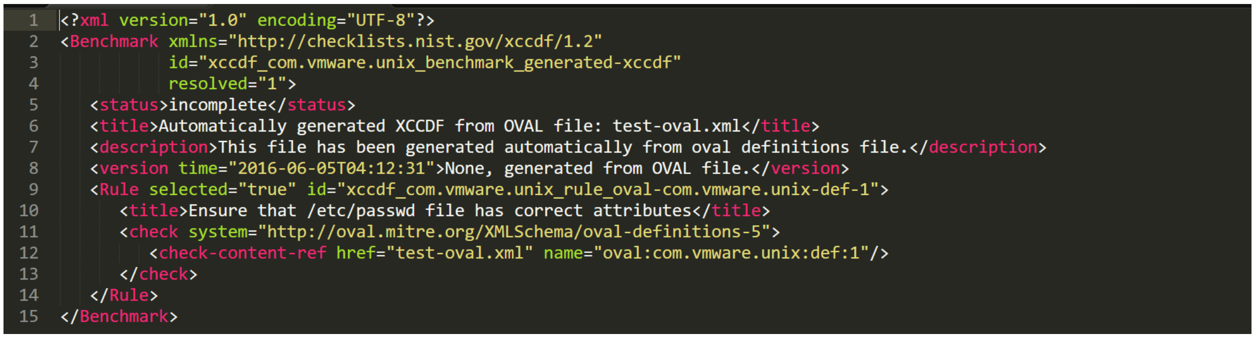
This would create a XCCDF 1.2 file from chosen OVAL file.



XCCDF File Created: C:\Users\kayush\Desktop\test-xccdf

Click **OK**.

The generated XCCDF XML file looks like below:



```
<?xml version="1.0" encoding="UTF-8"?>
<Benchmark xmlns="http://checklists.nist.gov/xccdf/1.2"
           id="xccdf_com.vmware.unix_benchmark_generated-xccdf"
           resolved="1">
   <status>incomplete</status>
   <title>Automatically generated XCCDF from OVAL file: test-oval.xml</title>
   <description>This file has been generated automatically from oval definitions file.</description>
   <version time="2016-06-05T04:12:31">None, generated from OVAL file.</version>
   <Rule selected="true" id="xccdf_com.vmware.unix_rule_oval-com.vmware.unix-def-1">
      <title>Ensure that /etc/passwd file has correct attributes</title>
      <check system="http://oval.mitre.org/XMLSchema/oval-definitions-5">
         <check-content-ref href="test-oval.xml" name="oval:com.vmware.unix:def:1"/>
      </check>
   </Rule>
</Benchmark>
```

Modify the XML tags such as status, title, description and version and then your XCCDF file is good to go. Optionally, you can manually create XCCDF profiles and groups, if required.

You can then do XCCDF assessments or use the XCCDF to create SCAP source data stream, if required.

References

Below are some useful references.

OVAL core definition schema – https://oval.mitre.org/language/version5.11/ovaldefinition/documentation/oval-definitions-schema.html

OVAL independent definition schema – http://oval.mitre.org/language/version5.11/ovaldefinition/documentation/independent-definitions-schema.html

OVAL Linux definition schema – http://oval.mitre.org/language/version5.11/ovaldefinition/documentation/linux-definitions-schema.html

OVAL Unix definition schema – http://oval.mitre.org/language/version5.11/ovaldefinition/documentation/unix-definitions-schema.html

OVAL Windows definition schema – http://oval.mitre.org/language/version5.11/ovaldefinition/documentation/windows-definitions-schema.html

VMware STIG compliance tool –

https://blogs.vmware.com/security/2016/05/vmware-releases-stig-compliance-app-for-free.html