# Quick guide for using the consolidated data streams

For applying the required configuration, users may use the individual scripts or commands provided in the ..\validationTestSuites\Consolidated \<Platform>\ configurationTools folder.

## Windows systems

1. Ensure that the following pre-requisites are met:

   1.1 The test systems shall be named TESTMACHINE, TESTMACHINE01, TESTMACHINE02, up to TESTMACHINE99.

   1.2 Python 3.2 is installed

   - On Windows x86 platforms, download and install: http://www.python.org/ftp/python/3.2.5/python-3.2.5.msi
   - On Windows x64 platforms, download and install: http://www.python.org/ftp/python/3.2.5/python-3.2.5.amd64.msi

   1.3 The Python extensions for Windows (PyWin32 Build 220 is recommended) are installed

   - On Windows x86 platforms, download and install: http://sourceforge.net/projects/pywin32/files/pywin32/Build%20220/pywin32-220.win32-py3.2.exe/download
   - On Windows x64 platforms, download and install: http://sourceforge.net/projects/pywin32/files/pywin32/Build%20220/pywin32-220.win-amd64-py3.2.exe/download

   1.4 The Microsoft Visual C++ 2010 and 2012 runtimes are installed

   - On Windows x86 platforms, download and install: https://www.microsoft.com/en-us/download/details.aspx?id=8328 and https://www.microsoft.com/en-us/download/details.aspx?id=30679
   - On Windows x64 platforms, download and install: https://www.microsoft.com/en-us/download/details.aspx?id=13523 and https://www.microsoft.com/en-us/download/details.aspx?id=30679

1.5 The Java Runtime Environment must be installed if you are running the SCAP reference implementation.

The latest version of the Java Runtime Environment can be downloaded from:http://www.oracle.com/technetwork/java/javase/downloads/index.html.

1.6 Windows PowerShell 3.0[1] or later shall be installed on Windows 7, 8.1, 10 and Server 2012 R2.

1.7 Please ensure the system is fully patched.

1.8 The test machine must be member of a domain and it is recommended to block the GPO inheritance:

- On the AD server: Start Group Policy Management -> Right Click on your domain -> New Organizational Unit: "OU-NO-GPOs" -> right click on "OU-NO-GPOs" -> Block Inheritance

- Start Active Directory Users and computers -> Drag and drop your test system to "OU-NO-GPOs"

- On the test system: Restart the test system

2. Create c:\validationTestSuites\ folder. Copy the combinedDataStreams_<version>.zip file to the test system and extract it to c:\validationTestSuites\. For the next steps we will assume that the zip file has been extracted to “c:\validationTestSuites\”. If you extract the files to a different location substitute that location in the following instructions.

The content of the c:\validationTestSuites\ folder should be as follows:

| Parent folder | Folder or File name | Description |
|---|---|---|
| ..\Windows\ | ValidationFiles | SCAP datastream validation results |
| | Documents | OVAL Test Data and configuration spreadsheet |
| | configurationTools | Tools for configuring the system |
| | Windows-datastream.xml | SCAP datastream |
| | catalog.xml | Expected results |
| | | |
| ..\tools\ | compare.py | Python program used to compare the actual scan results with the expected results |

[1] Requirements for Windows PowerShell system: https://technet.microsoft.com/en-us/library/hh847769.aspx

3. Configure the target system(s), either using the provided tools and instructions below or using an alternate method of your choice:

   3.1 From a command prompt, change to the folder to "c:\validationTestSuites\Windows\configurationTools"

   3.2 Execute the **config#.bat** file from the folder, where # is the configuration number that you wish to execute. This script shall be run with administrator privileges. There are 4 different configurations for Windows XP (config 1 to 4) and 8 configurations for Windows Vista, 7, 8.1, 10, and Server 2012 (config 1 to 8).

   For example, to apply the Configuration 1 on Windows 7 64bit, run as Administrator: "config1.bat Win764bit" or just run config1.bat and select the option for Windows 7. The following options are available for the config#.bat scripts:
   config[1-8].bat {WinXP32bit | WinVista32bit | Win7-8-32bit | Win7-8-10-2012-64bit}

4. Import the validation test content data stream for Windows (c:\validationTestSuites\Windows\Windows-datastream.xml) into the product/module following the vendor's instructions.

5. Perform configuration scans and export the evaluation results following the vendor's instructions. The result file format may look like:

   [arf | xccdf | oval | ocil | sc ]-[OS]-[platform]-config[1-8].xml

   Where:
   arf = ARF results
   xccdf = XCCDF results
   oval = OVAL results
   ocil = OCIL results
   sc = systems characteristics
   OS = w7, wvista, wxp, w8.1, or 2012
   Platform = 32 or 64

   For instance, the results file for consolidated data stream on Windows 7 64, configuration 1 may look like:
   arf-w7-64-config1.xml

   If you're exporting results from an USGCB data stream, replace "config[1-8]" with "exact", "reduced" or "enhanced":

   xccdf-win7-64-exact.xml

6. Compare the results produced by the product/module with the expected results from the spreadsheets or catalog.xml file. You may use the provide **compare.py** tool or an alternate method. The instructions below apply to the use of compare.py:

- From a command prompt, change the directory to c:\validationTestSuites\tools or the directory where the compare.py tool is located.

- Run: **python compare.py {RESULTSFILE} {CATALOGFILE} -i {SuiteID_from_catalog.xml} -o {Output_file}**

    - The compare.py script expects an ARF file. If you want to use it to parse an XCCDF file you must specify the **-x** switch. If the version of XCCDF is something other than 1.2 use the **-xv** {XccdfVersion} switch specify the version. For additional help or to see all available option run compare.py with the **-h** switch.

For instance, using the catalog.xml file available for the Windows datastream (c:\validationTestSuites\Windows\catalog.xml), run:

**python compare.py arf-w7-64-config1.xml catalog.xml -i G2.w7_configuration1 -o arf-w7-64-config1.output**

The script will flag all the differences between the results obtained by the product/module and the expected results from the catalog.xml and save it to a file (arf-w7-64-config1.output). If there are any differences, please verify all the configuration settings have been successfully applied to the system.

7. If you plan to use the same system for testing another configuration you should create a snapshot and clean up the configuration changes made in step 4. You can do this using the provided **cleanup.bat** tool or an alternate method. Run the cleanup.bat from the "c:\validationTestSuites\Windows\configurationTools" to restore a default configuration to the test system (as much as possible). Please note that the cleanup.bat file attempts to restore the system to an out-of-the-box state, not the state that was present before running the configuration scripts. As such, if you are testing a product/module that requires a specific configuration on the system you may need to manually clean up the settings and/or modify the cleanup process to work with the vendor's product/module.

8. Repeat steps 3 - 7 for each of the 8 configurations.

## RHEL systems

1. Ensure that the following pre-requisites are met:

    1.1 The /tmp must be created as separate partition and not as a folder on the root partition.

    1.2 The test systems shall be named localhost.localdomain

    1.3 The dos2unix command shall be installed (i.e. yum install dos2unix)

    1.4 Python 3.2 is installed

    Instruction for installing Python 3.2.6 on RHEL:

    ```
    # yum groupinstall "Development Tools"
    # wget https://www.python.org/ftp/python/3.2.6/Python-3.2.6.tgz
    # tar xvf Python-3.2.6.tgz
    # cd Python-3.2.6
    # ./configure
    # make
    # make install
    ```

    Verify if Python was installed successful:
    # python3 -V
    (it should print out: Python 3.2.6)

    1.5 If using the reference implementations the Java Runtime environment shall be installed

    1.6 Set SELINUX to permissive mode: edit /etc/selinux/config, replace “SELINUX=enforcing” with “SELINUX=permissive” and reboot the system.

    1.7 The Red Hat Network Daemon (rhnsd) should be installed as default. Please verify that this service is present on your test system.

    1.8 Ensure the xinetd is installed:

    ```
    # yum install xinetd
    ```

2. Create /validationTestSuites folder directly on the / partition. Copy the combinedDataStreams_<version>.zip file to the /validationTestSuites folder and extract it there. For the next steps we will assume that the zip file has been extracted to “/validationTestSuites”. If you extract the files to a different location substitute that location in the following instructions.

The content of the /validationTestSuites folder should be as follows:

| Parent folder | Folder or File name | Description |
|---|---|---|
| ../RHEL/ | ValidationFiles | SCAP datastream validation results |
| | Documents | OVAL Test Data and configuration spreadsheet |
| | configurationTools | Tools for configuring the system |
| | RHEL-datastream.xml | SCAP datastream |
| | catalog.xml | Expected results |
| | README.txt | Readme file |
| ../tools/ | compare.py | Python program used to compare the actual scan results with the expected results |

3. Configure the target system(s), either using the provided tools and instructions below or using an alternate method of your choice:

3.1 Run the following commands as root:

```
#  cd /validationTestSuites/RHEL/configurationTools/

# dos2unix *.sh

# dos2unix *.py

# chmod +x *.sh
```

3.2 Apply Configuration 1 to System 1

Depending on the RHEL version, run as root:

RHEL5: # source config1_rhel5.sh

or

RHEL6: # source config1_rhel6.sh

or

RHEL7: # source config1_rhel7.sh

3.3 Apply Configuration 2 to System 2

Depending on the RHEL version, run as root:

RHEL5: # source config2_rhel5.sh

or

RHEL6: # source config2_rhel6.sh

or

RHEL7: # source config2_rhel7.sh

4. Import the consolidated data stream for RHEL (/validationTestSuites/RHEL/RHEL-datastream.xml) into the product/module

5. Perform configuration scans and export the evaluation results following the vendor's instructions. The result file format may look like:

[arf | xccdf | oval | ocil | sc ]-[OS]-[platform]-config[1-2].xml

Where:

arf = ARF results

xccdf = XCCDF results

oval = OVAL results

ocil = OCIL results

sc = systems characteristics

OS = rhel5

Platform = 32 or 64

For instance, the results file for consolidated data stream on RHEL 5 64 bit, configuration 1 may look like: arf-rhel5-64-config1.xml

6. Compare the results with the expected results from the catalog.xml using the compare.py tool:

- From a command prompt, change the directory to /validationTestSuites/tools or the directory where the compare.py tool is located.
- Run: **python3 compare.py {RESULTSFILE} {CATALOGFILE} -i {SuiteID_from_catalogfile} -o {Output_file}**
- The compare.py script expects an ARF results file by default, if you want to use it to parse and XCCDF results file then you must also specify the -x switch to indicate that the file is XCCDF. If the XCCDF version is not version 1.2 you must use the -xv VersionNumber switch to specify the XCCDF version. For additional help or to see all available option run compare.py with the -h switch.

For instance, using the catalog.xml file available for the RHEL datastream (/validationTestSuites/RHEL/catalog.xml), run:

python3 compare.py arf-rhel5-64-config1.xml catalog.xml  -i G2.configuration1 -o arf-rhel5-64-config1.output

The script will flag all the differences between the results obtained by the product/module and the expected results from the catalog.xml and save it to output file (arf-rhel5-64-config1.output). If there are any differences, please verify all the configuration settings have been successfully applies to the system.

## MAC OS X systems

1. Ensure that the following pre-requisites are met:

   1.1 Mac OS 10.11 (OS X El Capitan) is installed with default settings

   1.2 The test systems shall be named TESTMACHINE, TESTMACHINE01, TESTMACHINE02, up to TESTMACHINE99.

   1.3 Python 3.5.1 is installed

   Download and install https://www.python.org/ftp/python/3.5.1/python-3.5.1-macosx10.6.pkg

   Verify if Python was installed successful:
   $ python3 -V
   (it should print out: Python 3.5.1)

2. Create /validationTestSuites folder directly on the / partition. Copy the combinedDataStreams_<version>.zip file to the /validationTestSuites folder and extract it there. For the next steps we will assume that the zip file has been extracted to "/validationTestSuites". If you extract the files to a different location substitute that location in the following instructions.

The content of the /validationTestSuites folder should be as follows:

| Parent folder | Folder or File name | Description |
|---|---|---|
| ../MacOS/ | ValidationFiles | SCAP datastream validation results |
| | Documents | OVAL Test Data and configuration spreadsheet |
| | configurationTools | Tools for configuring the system |
| | MacOS-datastream.xml | SCAP datastream |
| | catalog.xml | Expected results |
| | README.txt | Readme file |
| ../tools/ | compare.py | Python program used to compare the actual scan results with the expected results |

3. Configure the target system(s), either using the provided tools and instructions below or using an alternate method of your choice:

   3.1 Run the following commands:

   $ cd /validationTestSuites/MacOS/configurationTools/

$ sudo chmod +x *.sh

1.1 Apply Configuration 1 to System 1

$ sudo ./config1_macosx.sh

1.2 Apply Configuration 2 to System 2

$ sudo ./config2_macosx.sh

4. Import the consolidated data stream for MacOS (/validationTestSuites/MacOS/MacOS-datastream.xml) into the product/module

5. Perform configuration scans and export the evaluation results following the vendor's instructions. The result file format may look like:

[arf | xccdf | oval | ocil | sc ]-[OS]-[platform]-config[1-2].xml

Where:

arf = ARF results

xccdf = XCCDF results

oval = OVAL results

ocil = OCIL results

sc = systems characteristics

OS = rhel5

Platform = 32 or 64

For instance, the results file for consolidated data stream on MacOS 5 64 bit, configuration 1 may look like: arf-macosx-64-config1.xml

6. Compare the results with the expected results from the catalog.xml using the compare.py tool:

- From a command prompt, change the directory to /validationTestSuites/tools or the directory where the compare.py tool is located.
- Run: **python3 compare.py {RESULTSFILE} {CATALOGFILE} -i {SuiteID_from_catalogfile} -o {Output_file}**
- The compare.py script expects an ARF results file by default, if you want to use it to parse and XCCDF results file then you must also specify the -x switch to indicate that the file is XCCDF. If the XCCDF version is not version 1.2 you must

use the -xv VersionNumber switch to specify the XCCDF version. For additional help or to see all available option run compare.py with the -h switch.

For instance, using the catalog.xml file available for the MacOS datastream (/validationTestSuites/MacOS/catalog.xml), run:

$ python3 compare.py arf-macosx-64-config1.xml catalog.xml -i G2.macos_configuration1 -o arf-macosx-64-config1.output

The script will flag all the differences between the results obtained by the product/module and the expected results from the catalog.xml and save it to output file (arf-rhel5-64-config1.output). If there are any differences, please verify all the configuration settings have been successfully applies to the system.