RED HAT®
CONSULTING

Creating and Publishing Ansible Roles

Travis Michette

Version 1.0

Table of Contents

1. Ansible Roles 1
1.1. Creating a Github repository for an Ansible Role 1
1.1.1. Creating an Ansible Role for Publishing 5
1.2. Publishing Ansible Roles to Ansible Galaxy 10
2. System Security Policy and Compliance 19
2.1. Customizing SCAP Content 19
2.2. Running a SCAP Scan with Custom Content 31
2.3. Creating an Ansible Remediation Playbook Based on SCAP Scan Results 40

1. Ansible Roles

Ansible roles … blah blah blah

1.1. Creating a Github repository for an Ansible Role

The first step in the creation of an Ansible role for publishing is to create a Github repository to house the role and use the **ansible-galaxy init** command to initialize the proper directory structure.

1. Create and Define the Ansible ROLE and Github Repository

Listing 1. Initializing an Ansible Role

```
[student@workstation github]$ ansible-galaxy init Demo_Role
- Demo_Role was created successfully
```

Listing 2. Initializing a Git Repository

```
[student@workstation github]$ cd Demo_Role/

[student@workstation Demo_Role]$ git init
Initialized empty Git repository in /home/student/github/Demo_Role/.git/
```

Listing 3. Add Files to Local Repository

```
[student@workstation Demo_Role]$ git add .

[student@workstation Demo_Role]$ git commit -m "Added Role Structure"
[master (root-commit) 5c12d1a] Added Role Structure
 8 files changed, 106 insertions(+)
 create mode 100644 README.md
 create mode 100644 defaults/main.yml
 create mode 100644 handlers/main.yml
 create mode 100644 meta/main.yml
 create mode 100644 tasks/main.yml
 create mode 100644 tests/inventory
 create mode 100644 tests/test.yml
 create mode 100644 vars/main.yml
```

Local GIT Repository vs. Github Repository

At this point, the repository that has been created is a local **git** repository. It is necessary to create a **New Repository** on Github and connect the local repository to Github

2. Login to Github and click "New" to create a new repository

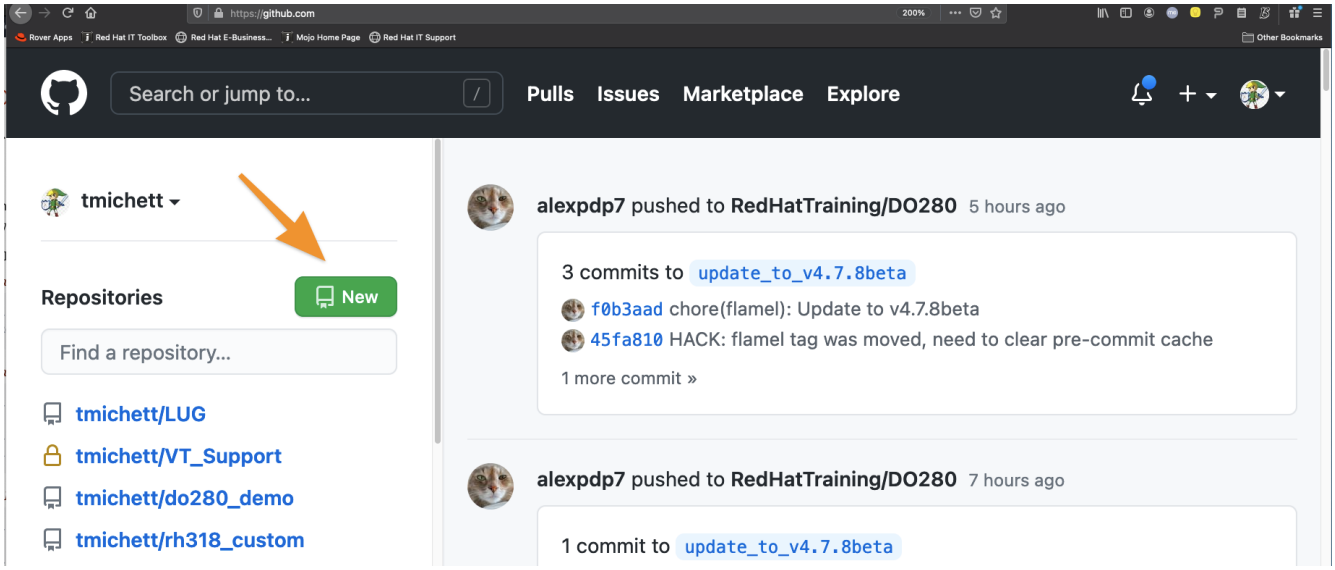

Figure 1. Github New Repository

3. Provide a **Name** for the repository **ONLY** but do not initialize as you are going to be linking a local repository. It is also OK to provide **Description** for this repository. Then click **Create Repository**

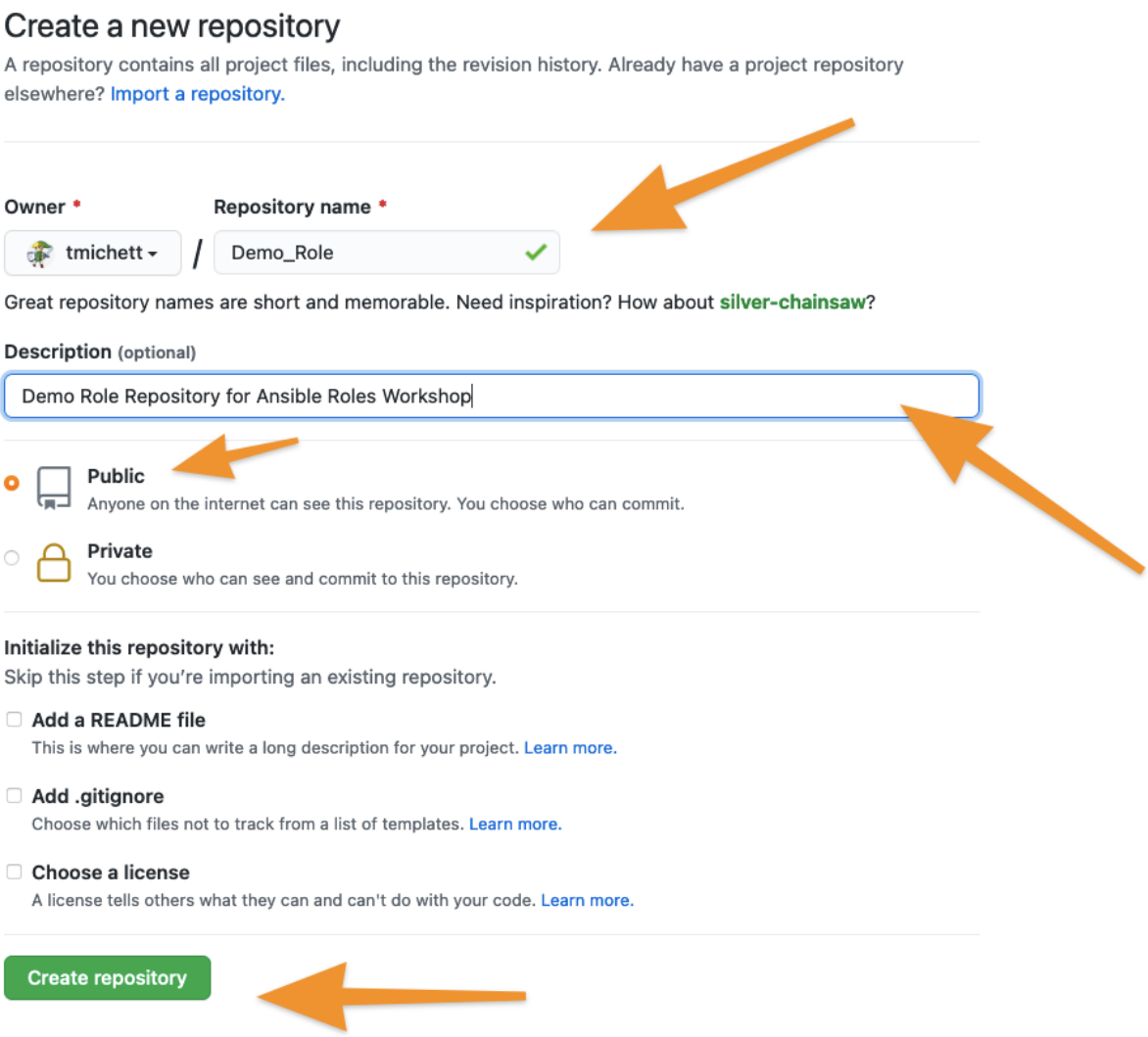

Figure 2. Github Repository Creation

Github Repository Quick Setup

Github will provide you with a **Quick Setup** set of instructions. These are basic instructions that can be used to import/connect an existing local **git** repository.

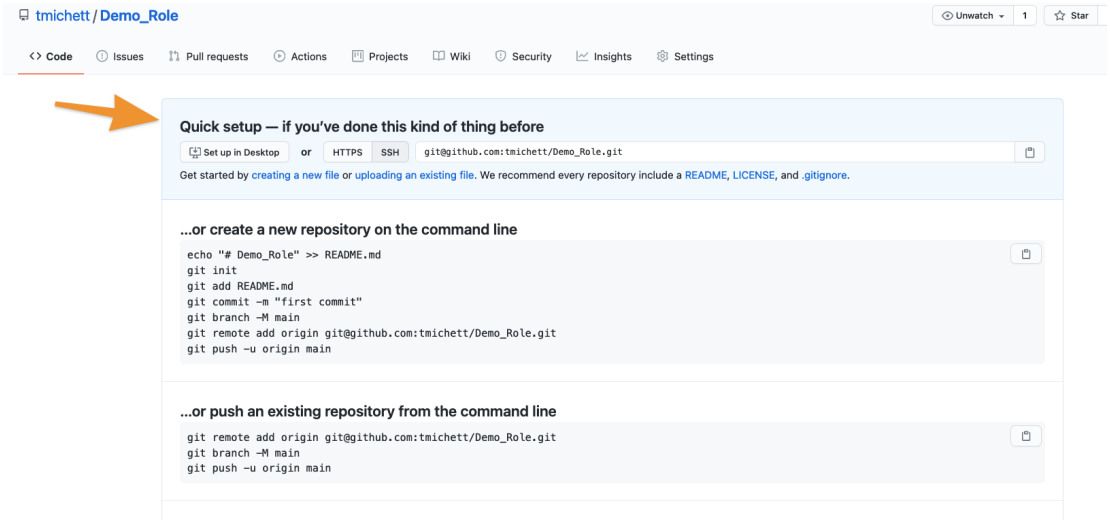

Figure 3. Github Quick Setup

4. Link your local GIT repository to Github by using **git add origin** and **git push origin master** commands.

```
[student@workstation Demo_Role]$ git remote add origin git@github.com:tmichett/Demo_Role.git

[student@workstation Demo_Role]$  git push -u origin master
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (16/16), 2.30 KiB | 471.00 KiB/s, done.
Total 16 (delta 0), reused 0 (delta 0)
To github.com:tmichett/Demo_Role.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

5. Verify that the initial contents were loaded to Github by refreshing the web browser

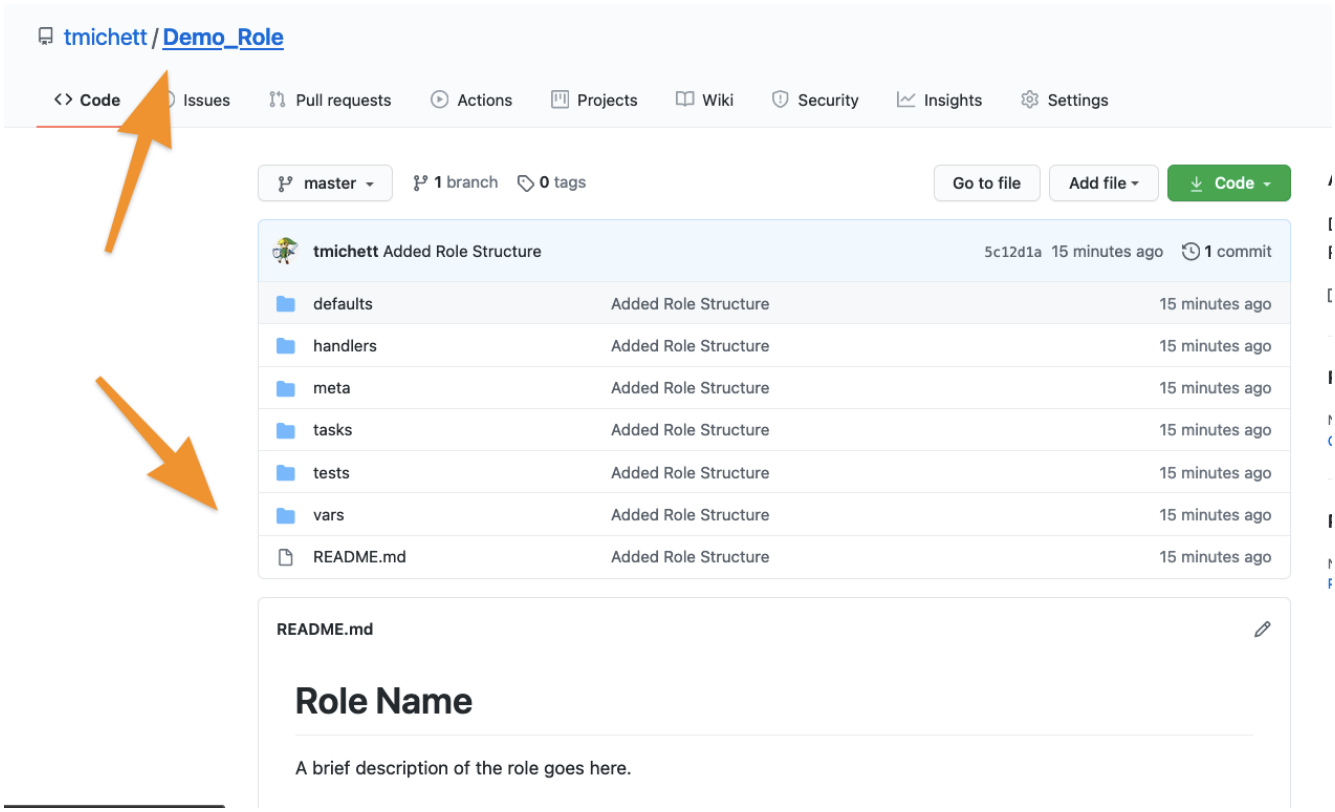

Figure 4. Github Repository Contents

1.1.1. Creating an Ansible Role for Publishing

After the creating a Github repository has been created along with the initial Ansible role structure created by the **ansible-galaxy** command, it is now time to take existing playbooks to break down into tasks and other sections to create the role or create the role from scratch.

Role Creation

I find that it is easier to create a role from an existing playbook and to break the playbook down into modules. Remember that Ansible roles and playbooks are supposed to be simple and easy to follow. An Ansible Role especially should be a simple role to accomplish a given outcome and the role should be broken down and easy to use. There are always trade-offs with this process as well as the **Ansiblize Role** we are looking at could be broken down into smaller roles:

- Create and Setup the User (user creation, password file, SSH keys, SUDOERS file)
- Configure SSHD

However, since I wanted a single role to perform a single **main** task, I decided not to break the larger **Ansiblize Role** into two smaller roles.

Creating the Ansible Role

The preferred method to create an Ansible role would be to start from a working playbook and to create the various role components. At a minimum, there should be a tasks section. After the various role components have been created with their corresponding **main.yml** files, any unused directories should be removed from the role. It is also very important for **Ansible Galaxy** to have the role created with a proper **README** file and **meta** files so that it will be scored a **5/5** on https://galaxy.ansible.com.

Ansiblize.YML Playbook

For this demonstration, we will be taking the Ansiblize Playbook and turning it into an Ansible Role. This workshop has the complete playbook as well as the individual components broken down for ease of use and implementation. The individual components are located at Playbook Components. The steps below demonstrate how to create the various files required for the Ansible role.

1. Create and copy the Ansible tasks into the **tasks/main.yml**

Listing 4. **tasks/main.yml** *File*

```
---
### Create the Ansible User
- name: Create Ansible User
  user:
    name: "{{ ansible_user_name }}"
    state: present
    shell: /bin/bash
    comment: Ansible User for System


### Create the Ansible User
- name: Create Ansible User Password (if required)
  shell: echo {{ ansible_user_password }} | passwd {{ ansible_user_name }} --stdin

### Copy Ansible User Public SSH Keys (if defined)
### This will take the public key and set as the authorized_key from the current user
### Only works if the current user/remote user is the same as user being created
- name: Copy Ansible User SSH Keys
  authorized_key:
    user: "{{ ansible_user_name }}"
    state: present
    key: "{{ lookup('file', lookup('env','HOME') + '/.ssh/id_rsa.pub') }}"
  when: ssh_key_file_data is not defined and ssh_key_answer

### Copy Ansible User Public SSH Keys from file (if defined)
### This will take the public key specified and set as and authorized_key for the user specified
- name: Copy Ansible User SSH Keys
  authorized_key:
    user: "{{ ansible_user_name }}"
    state: present
    key: "{{ ssh_key_file_data }}"
  when: ssh_key_file_data is defined

### Modify Sudoers to Allow Passwordless SUDO
- name: Create Ansible User SUDOERS Entry
  copy:
    content: "{{ ansible_user_name }} ALL=(ALL) NOPASSWD:ALL\n"
    dest: /etc/sudoers.d/{{ ansible_user_name }}

### Modify SSH_CONFIG to prevent Root Login
- name: Prevent Root Login via SSH (if required)
  lineinfile:
```

```
    path: /etc/ssh/sshd_config
    regexp: '^PasswordAuthentication'
    line: PasswordAuthentication {{ ssh_passwords_allowed }}
    backup: yes
  notify:
    - Restart SSHD Test

### Modify SSH_CONFIG to permit SSH Key ONLY Access (no passwords)
- name: Prevent Root Login via SSH (if required)
  lineinfile:
    path: /etc/ssh/sshd_config
    regexp: '^PermitRootLogin'
    line: PermitRootLogin {{ ssh_root_allowed }}
    backup: yes
  notify:
    - Restart SSHD

### Test the SSHD Config File
### Technically this will exit out and break the Ansible playbook if it fails
- name: Test SSHD Config File
  shell: sshd -t
  register: sshd_results

### Test the SSHD Config File
- name: Test SSHD Config File Debug Output
  debug:
    msg: The SSHD_CONFIG file is broken. Please fix the file as there is an issue with either the line with "PermitRootLogin" or the line
with "PasswordAuthentication". The specific issue is {{ sshd_results.stderr_lines }}
  when: sshd_results.rc != 0
```

2. Create and copy the Handler into the **handlers/main.yml** file

Listing 5. **handlers/main.yml**

```
---
- name: Restart SSHD
  systemd:
    name: sshd
    state: restarted
  when: sshd_results.rc==0
```

3. Create and copy the Variables into the **defaults/main.yml** file

Listing 6. **defaults/main.yml**

```
---
ansible_user_name: ansible-user
ansible_user_password: redhat
```

4. Modify the **meta/main.yml** file

Listing 7. **defaults/main.yml**

```
---
galaxy_info:
  author: Travis Michette
  description: Simple setup systems to become Ansible managed hosts.
  company: Michette Technologies
```

```
  # If the issue tracker for your role is not on github, uncomment the
  # next line and provide a value
  # issue_tracker_url: http://example.com/issue/tracker
  issue_tracker_url: https://github.com/tmichett/Demo_Role/issues
  # Some suggested licenses:
  # - BSD (default)
  # - MIT
  # - GPLv2
  # - GPLv3
  # - Apache
  # - CC-BY
  license: BSD

  min_ansible_version: 2.7

  # If this a Container Enabled role, provide the minimum Ansible Container version.
  # min_ansible_container_version:

  # Optionally specify the branch Galaxy will use when accessing the GitHub
  # repo for this role. During role install, if no tags are available,
  # Galaxy will use this branch. During import Galaxy will access files on
  # this branch. If Travis integration is configured, only notifications for this
  # branch will be accepted. Otherwise, in all cases, the repo's default branch
  # (usually master) will be used.
  #github_branch:

  #
  # Provide a list of supported platforms, and for each platform a list of versions.
  # If you don't wish to enumerate all versions for a particular platform, use 'all'.
  # To view available platforms and versions (or releases), visit:
  # https://galaxy.ansible.com/api/v1/platforms/
  #
  platforms:
    - name: Fedora
      versions:
        - all
  #   - 25
  # - name: SomePlatform
  #   versions:
  #   - all
  #   - 1.0
  #   - 7
  #   - 99.99
    - name: EL
      versions:
        - all
  galaxy_tags:
    - sudoers
    - sshd
    - users
    - ansible
    # List tags for your role here, one per line. A tag is a keyword that describes
    # and categorizes the role. Users find roles by searching for tags. Be sure to
    # remove the '[]' above, if you add tags to this list.
    #
    # NOTE: A tag is limited to a single word comprised of alphanumeric characters.
    #       Maximum 20 tags per role.

dependencies: []
  # List your role dependencies here, one per line. Be sure to remove the '[]' above,
  # if you add dependencies to this list.
```

5. Modify the **README.md** file to contain the correct items and README information for your role.

Listing 8. **README.md** *File*

```
Ansiblize Systems
=========

This role is meant to setup and create an Ansible user with a username, password, SSH key, and to add the user to the SUDOERS file with
password-less sudo access.

Requirements
------------

This role assumes that you are operating on an EL-based Linux distribution utilizing SystemD.

Role Variables
--------------

**pkg_name** - This variable is the name of the package or a list of package names that can be installed on the system. This is the "ONLY"
required variable to be supplied.

**pkg_state** - This variable is a default variable and set to "latest". The allowed values for this variable are "latest" and "present"
to install the package(s) or "absent" to ensure that the package has been removed.

Dependencies
------------

There are no dependencies for this playbook, but there is another related role published to work with Linux services.

Example Playbook
----------------

Including an example of how to use your role (for instance, with variables passed in as parameters) is always nice for users too:



    ---
    - name: Install Software Packages
      hosts: serverc
      vars:
        pkg_name:
          - vim
          - tree
          - httpd
      roles:
        - tmichett.deploy_packages



License
-------

BSD

Author Information
------------------

Travis Michette
tmichett@redhat.com
```

1.2. Publishing Ansible Roles to Ansible Galaxy

One the role has been created and tested thoroughly, it can be shared through Ansible Galaxy. Again, this assumes that you've already setup an account with Github and linked that account to your Ansible Galaxy account.

1. Login to Ansible Galaxy

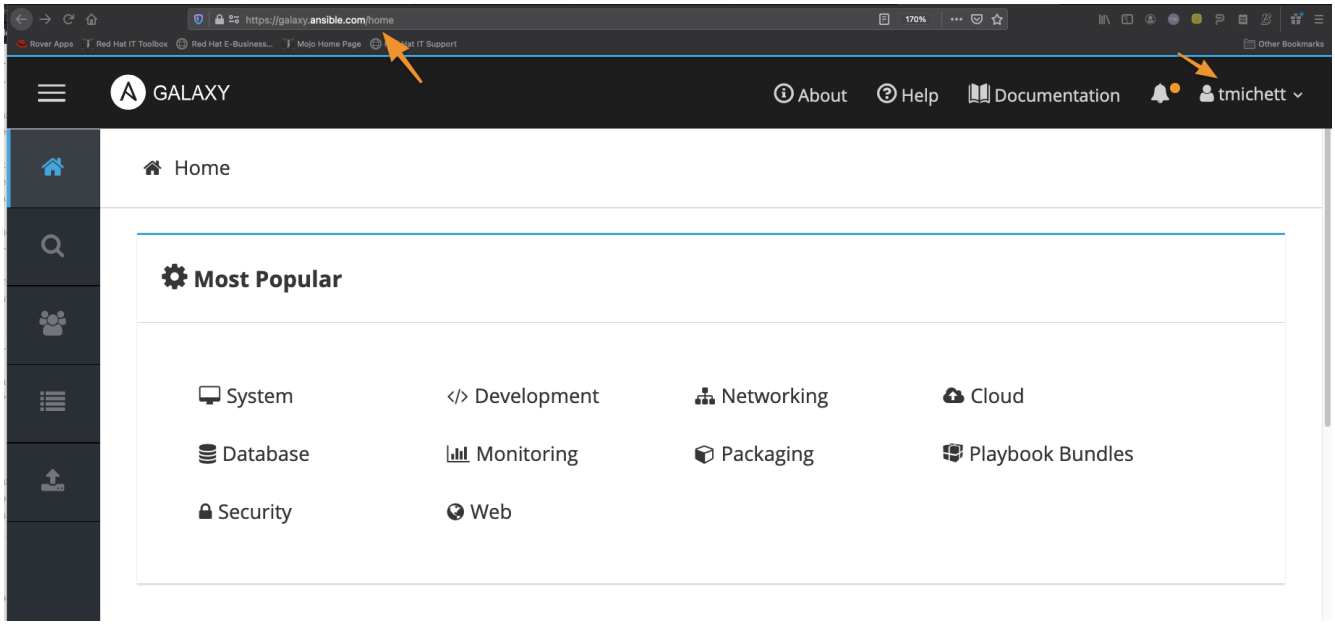

Figure 5. Ansible Galaxy Home

2. Select your **My Content** (**Namespaces**) selection from the side navigation menu and click "**Add Content**"

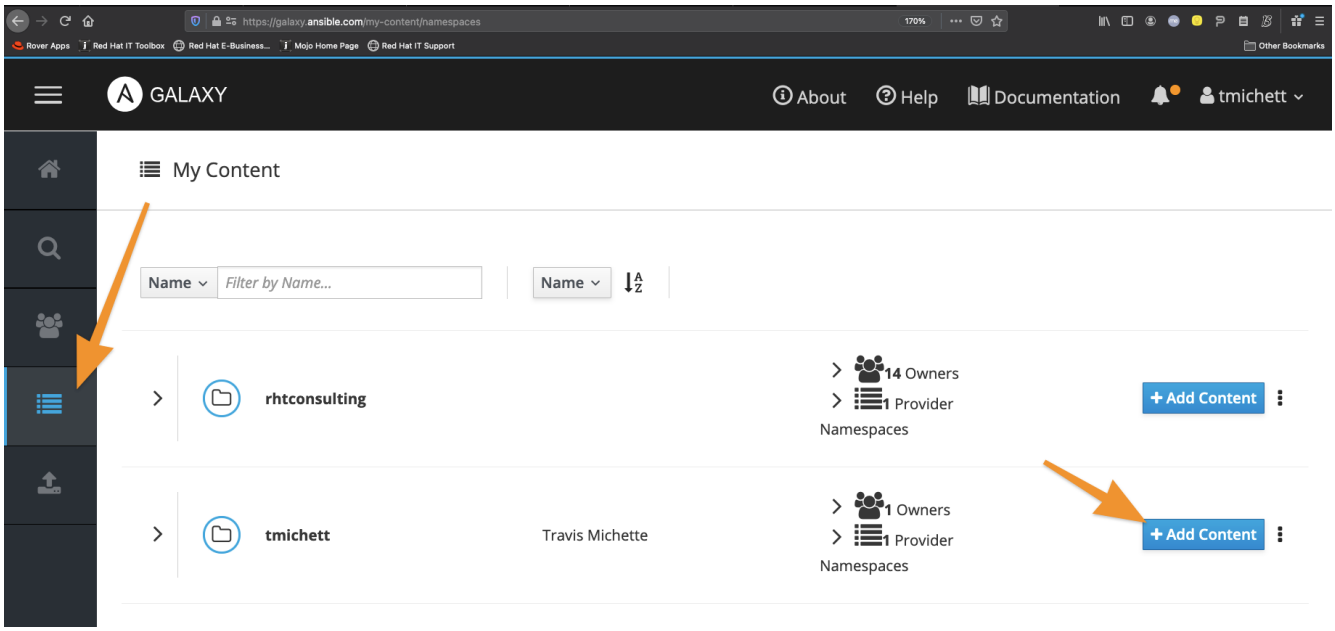

Figure 6. Ansible Galaxy Content

3. Click **Import Role from Github**

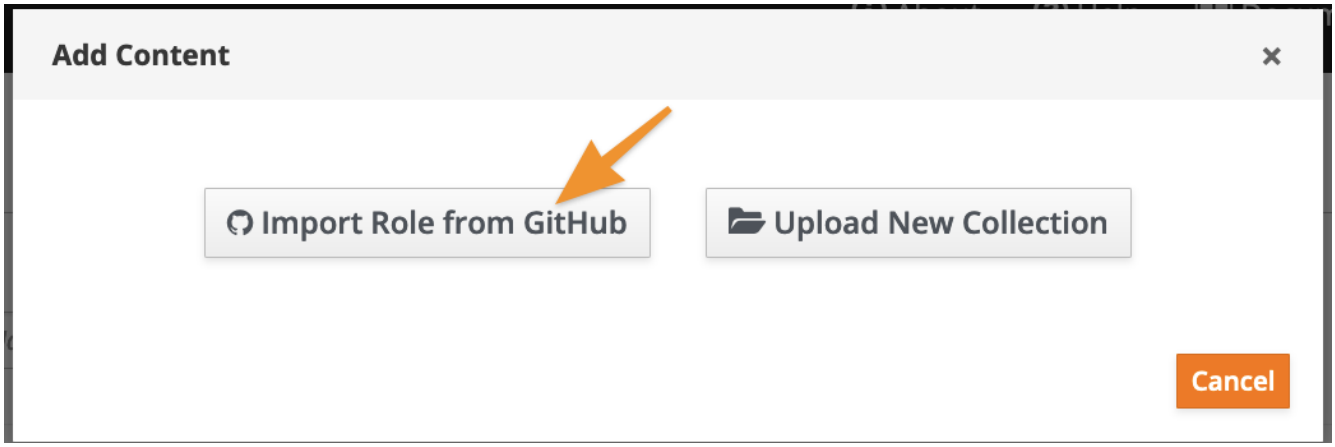

Figure 7. Ansible Galaxy Content Import

4. Locate and select the Github repository containing the role you wish to import. *(Make use of the filter to assist in locating the correct repository)*. Once you've located the correct repository, place a **checkmark** in the box and then click "**OK**".

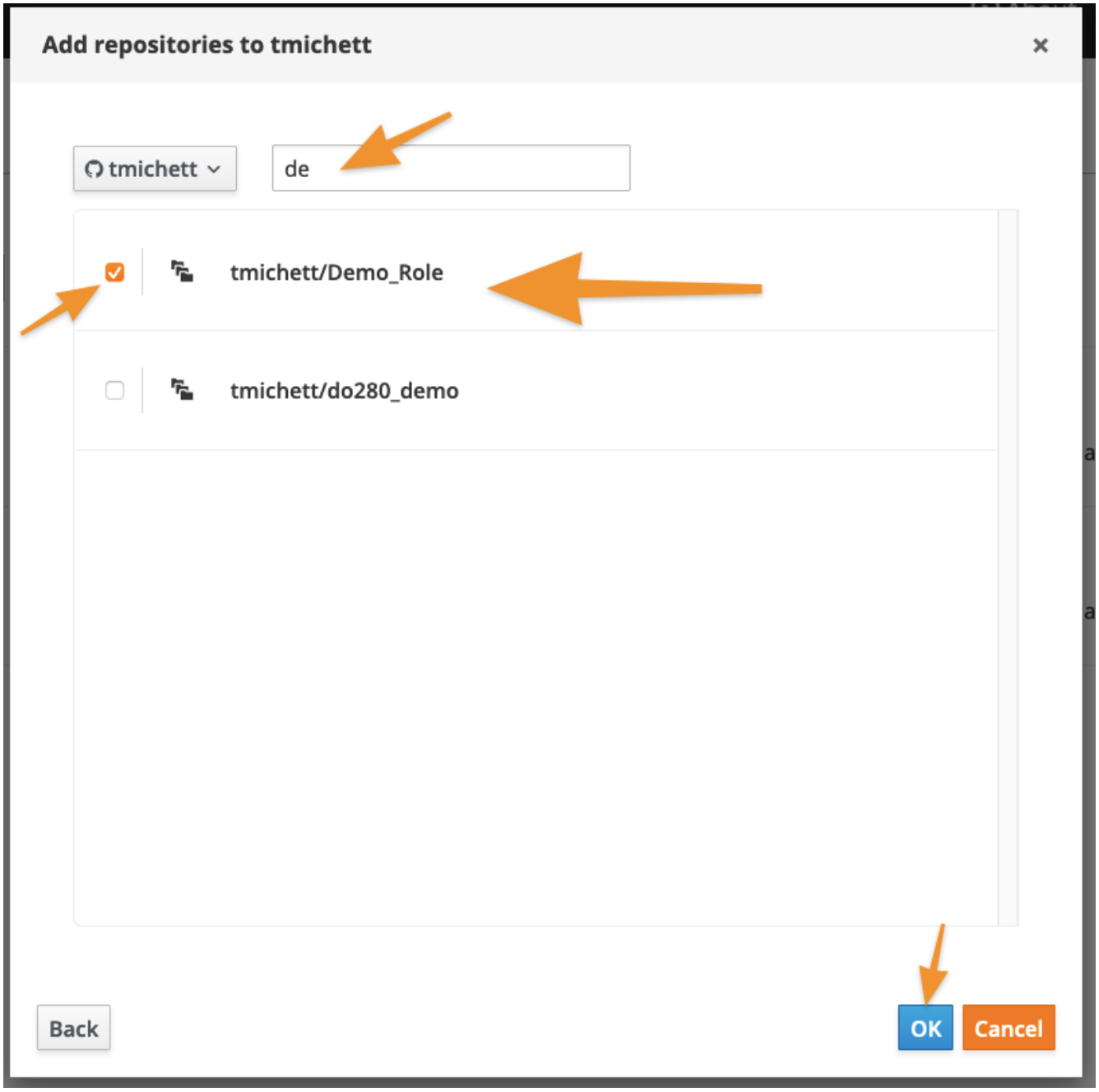

Figure 8. Ansible Galaxy Import Role from Github

5. The role import process will begin and Ansible Galaxy will automatically perform validation and linting on the imported role and components. You can click the **>** beside your Ansible Galaxy ID or the Ansible Galaxy Team ID (*depending on where you imported the roles*) to expand the list of roles.

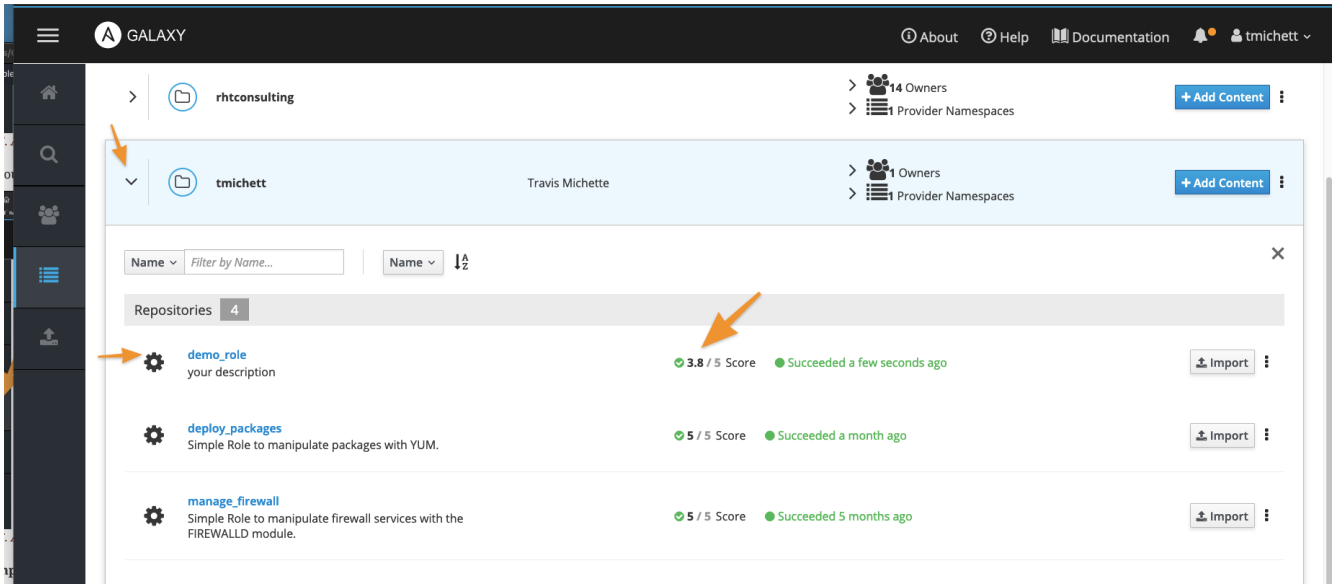

Figure 9. Ansible Galaxy Content Listing

Ansible Galaxy Content Provider Namespaces

In this demo, I have two (2) **Ansible Galaxy Content Provider Namespaces**. One of these is my personal namespace **tmichett** and the other is where I'm a member of Red Hat Consulting. Most likely you will have a single Namespace.

Warning Header

You will want to check the status of your imported Role and see what type of score it received. Ideally, you should be receiving a score of 5/5 meaning that at least all role components were imported successfully and passed the **linting** process and there were no errors that could be picked up from an automated scan process.

6. After importing the role, check the **score** and look for anything that can be fixed easily and re-published/re-processed to Ansible Galaxy. Select the role to receive more information. Click the **Show Details** for the **Quality Score** to get a breakdown of what may need to be fixed.

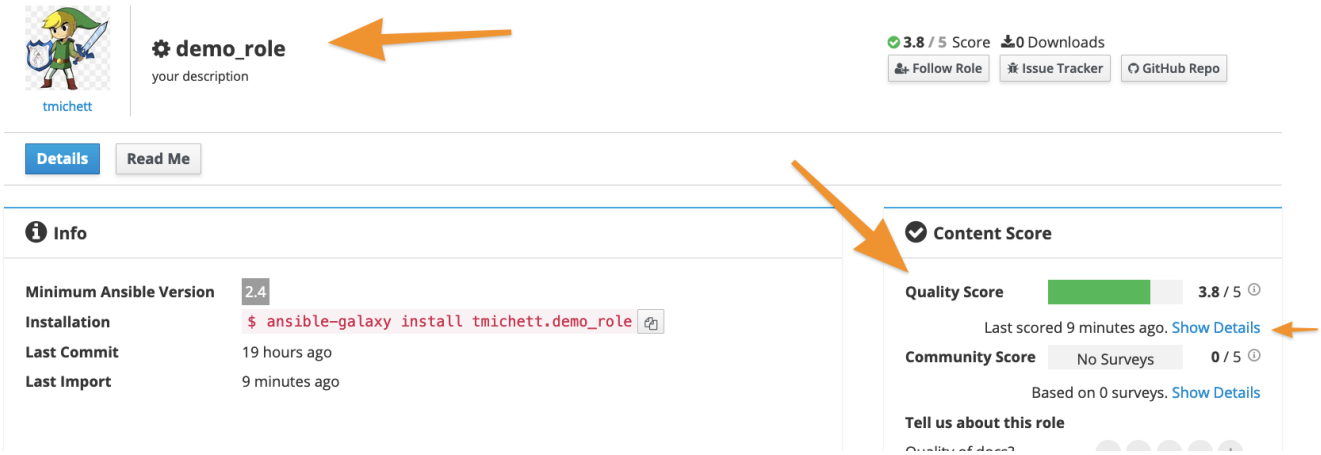

Figure 10. Ansible Galaxy - Role Details

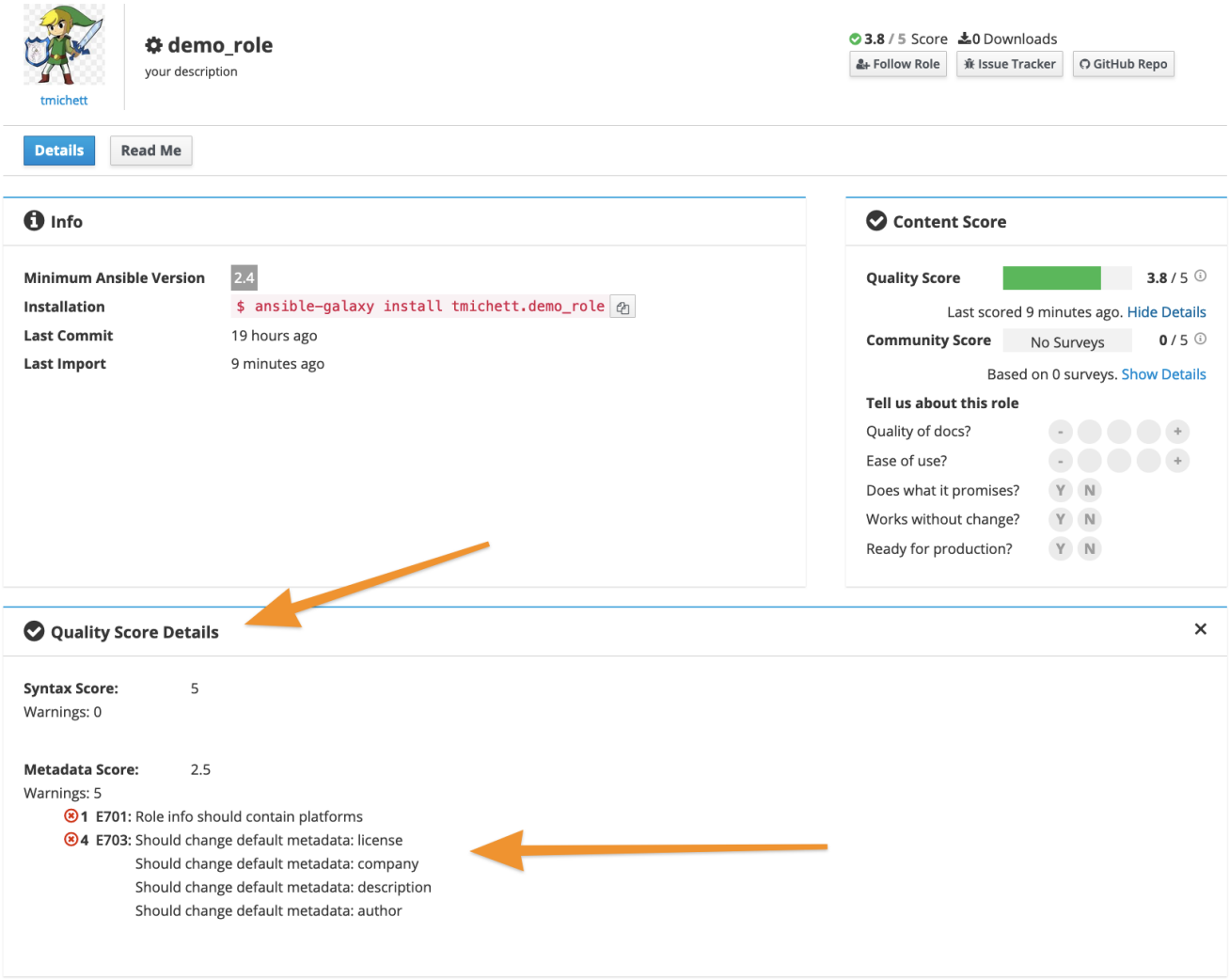

Figure 11. Ansible Galaxy - Role Details (Quality Score Details - Shown)

7. Resolve issues with role and commit changes to Github. Then return to Ansible Galaxy and re-process the role.

Listing 9. Performing **git** *Checks*

```
[student@workstation Demo_Role]$ git status ①
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   README.md
    modified:   defaults/main.yml
    modified:   handlers/main.yml
    modified:   meta/main.yml
    modified:   tasks/main.yml
    deleted:    vars/main.yml

no changes added to commit (use "git add" and/or "git commit -a")


[student@workstation Demo_Role]$ git add . ②


[student@workstation Demo_Role]$ git commit -m "Updated Git for the role contents" ③
[master 9406a9c] Updated Git for the role contents
 6 files changed, 162 insertions(+), 63 deletions(-)
 rewrite README.md (74%)
 rewrite tasks/main.yml (86%)
 delete mode 100644 vars/main.yml


[student@workstation Demo_Role]$ git push ④
Enumerating objects: 21, done.
Counting objects: 100% (21/21), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (11/11), 2.80 KiB | 718.00 KiB/s, done.
Total 11 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:tmichett/Demo_Role.git
   5c12d1a..9406a9c  master -> master
```

① Checking local repository status

② Adding changed content to local git repository

③ Committing content to local git repository

④ Pushing content to Github Repository

DEMO Role Note

In this example, I created the initial Ansible Role and pushed to Github, but I neglected to commit and push any of the changes and the actual role into the Github repository. The import into **Ansible Galaxy** imported just the role skeleton which was created using the **ansible-galaxy init** command.

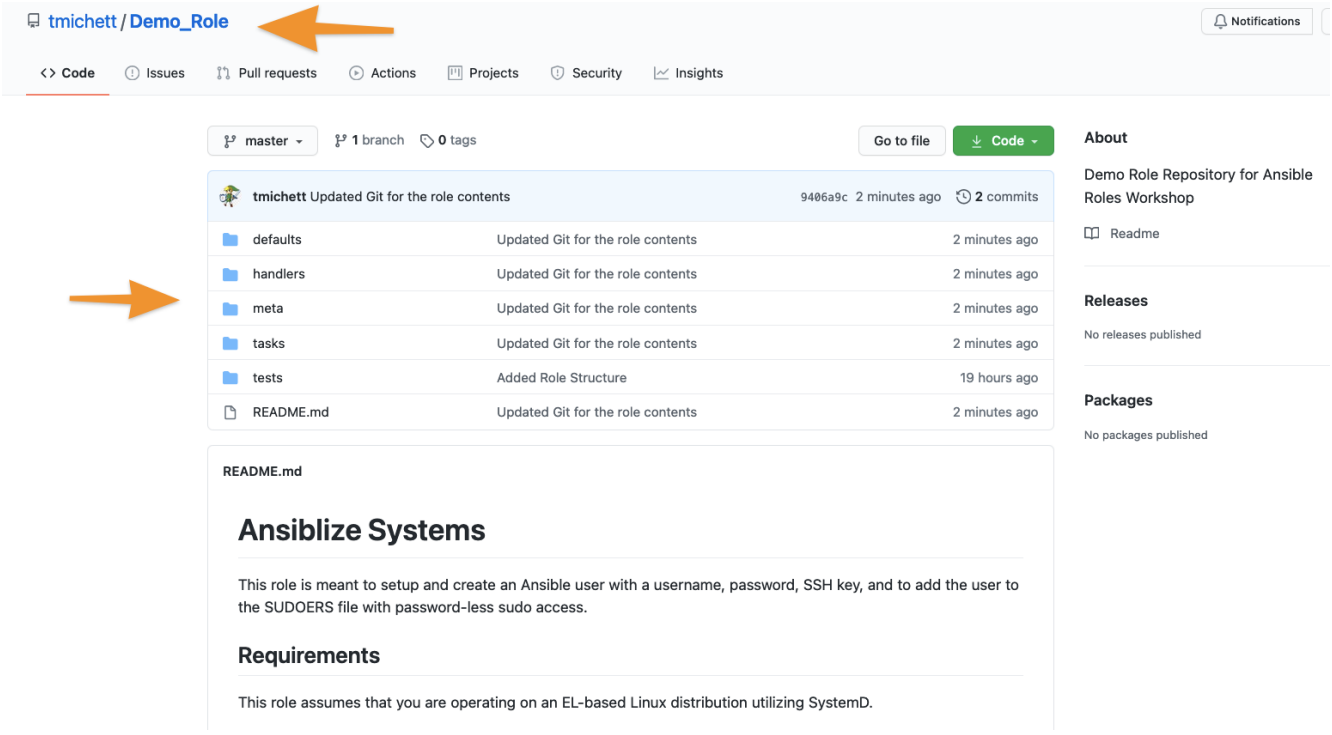

Figure 12. Github Repository Verification

8. Import/Replace the role in Ansible Galaxy using the **Import** button listed by the role.

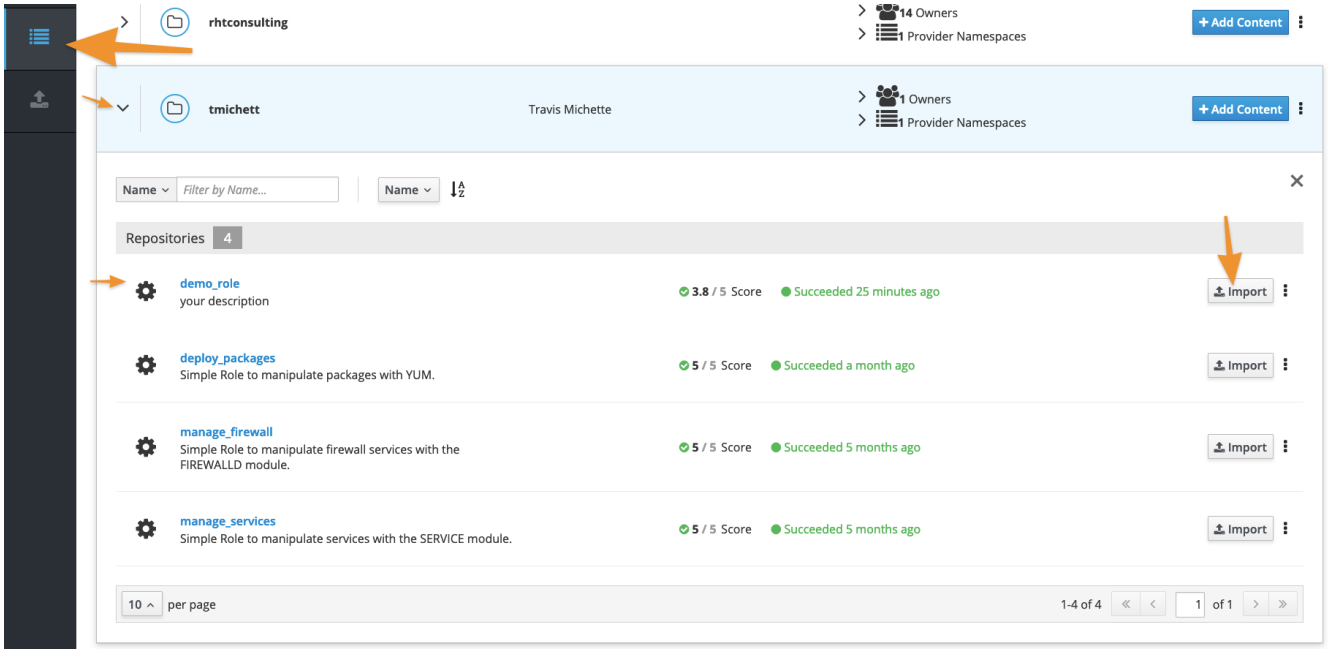

Figure 13. Refreshing Role from Github Repository

Refreshing the Role

The refresh process will perform the linting and verification process again as the role is imported/updated in Ansible Galaxy. This will create a new score which will hopefully achieve the 5/5 score on quality. Keep in mind, this score doesn't track how good/useful the role is, but more that it conforms to 100% Ansible Galaxy standards and that no required options are missing.

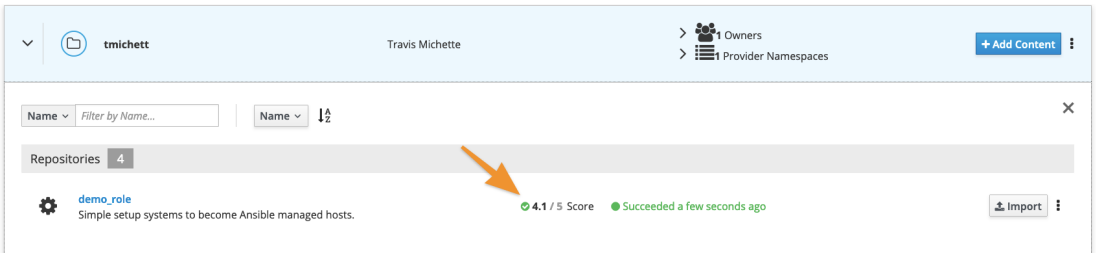

Figure 14. Ansible Galaxy - Quality Score Updated

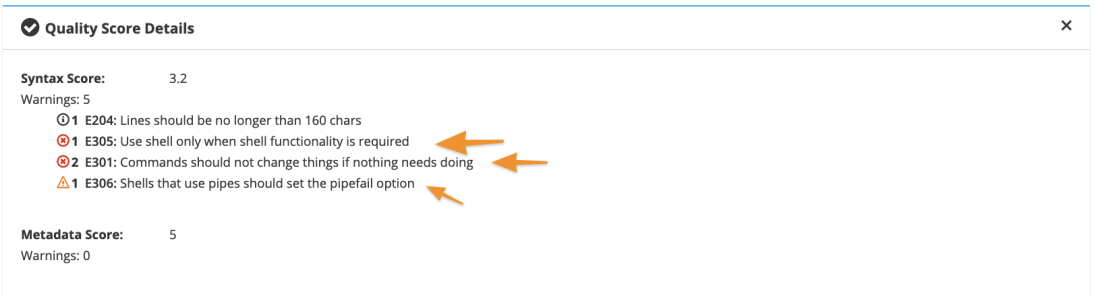

Figure 15. Ansible Galaxy - Quality Score Updated (Details)

In this instance, some of the items found are around choices for implementation of tasks using modules such as **shell** and **command** that are not idempotent. The modules were used to achieve specific functions and are considered fine for the implementation of this role. There are other modules that could be used for the password functionality, but this method is the simplest illustration of Ansible and how to create roles.

2. System Security Policy and Compliance

System security and compliance is a primary concern for people when thinking about the protection of systems and integrity of data. This set of hands-on procedures will focus on obtaining the content and necessary packages to perform a basic scan and remediate the system based on scan results. As part of the lab, you will be customizing your own SCAP content for a scan, view the results, and generate an Ansible playbook based on the failed results.

2.1. Customizing SCAP Content

Red Hat includes the SCAP Workbench application as a GUI application which allows scanning, customizing, and saving SCAP scans and results. The SCAP Workbench can be used to perform scans of remote systems over an SSH connection or it can be utilized to scan the local system. Since the SCAP Workbench is a GUI application, it must run on a system with X-Windows installed.

Servers to Configure

- workstation

Packages to Install

- scap-workbench

Since we are using an application on a VM that requires a GUI, we can use X11 forwarding with the SSH connection by specifying a **-X** on the command line.

Step 1 - SSH to Workstation

The fist step is to connect to the workstation and forward X11 traffic back to the local system.

Example 1. Connecting to the Workstation VM

Listing 10. Connecting to Workstation Using SSH

```
# ssh root@workstation -X
```

Step 2 - Install Packages

After connecting to the Workstation VM, you will need to install the SCAP Workbench and its dependencies.

Example 2. Installing SCAP Workbench

Listing 11. Using Yum to Install SCAP Workbench

```
# yum install scap-workbench

Loaded plugins: langpacks, search-disabled-repos
Resolving Dependencies
--> Running transaction check
---> Package scap-workbench.x86_64 0:1.1.6-1.el7 will be installed
--> Processing Dependency: openscap-utils >= 1.2.0 for package: scap-workbench-1.1.6-1.el7.x86_64
--> Processing Dependency: scap-security-guide for package: scap-workbench-1.1.6-1.el7.x86_64
--> Running transaction check
---> Package openscap-utils.x86_64 0:1.2.16-8.el7_5 will be installed
--> Processing Dependency: openscap-containers = 1.2.16-8.el7_5 for package: openscap-utils-1.2.16-8.el7_5.x86_64
---> Package scap-security-guide.noarch 0:0.1.36-9.el7_5 will be installed
--> Processing Dependency: openscap-scanner >= 1.2.5 for package: scap-security-guide-0.1.36-9.el7_5.noarch
--> Running transaction check
---> Package openscap-containers.noarch 0:1.2.16-8.el7_5 will be installed
---> Package openscap-scanner.x86_64 0:1.2.16-8.el7_5 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                Arch      Version             Repository           Size
================================================================================
Installing:
 scap-workbench         x86_64    1.1.6-1.el7         rhel--server-dvd    1.8 M
Installing for dependencies:
 openscap-containers    noarch    1.2.16-8.el7_5      rhel_updates         27 k
 openscap-scanner       x86_64    1.2.16-8.el7_5      rhel_updates         61 k
 openscap-utils         x86_64    1.2.16-8.el7_5      rhel_updates         27 k
 scap-security-guide    noarch    0.1.36-9.el7_5      rhel_updates        2.6 M

Transaction Summary
================================================================================
Install  1 Package (+4 Dependent packages)

Total download size: 4.5 M
Installed size: 64 M
Is this ok [y/d/N]:
```

Some things might already be installed for you, if SCAP Workbench is already installed, please move on to the next step. Also note the dependencies for SCAP Workbench as they are automatically installed.

Step 3 - Launching SCAP Workbench

In order to run a scan or customize SCAP content, you will need to launch the SCAP Workbench application.

You **must** use SCAP Workbench from a GUI, so it will need to run either locally or through an SSH connection with X11 forwarded.

Example 3. Launching SCAP Workbench

Listing 12. Using SCAP Workbench

```
# scap-workbench
```

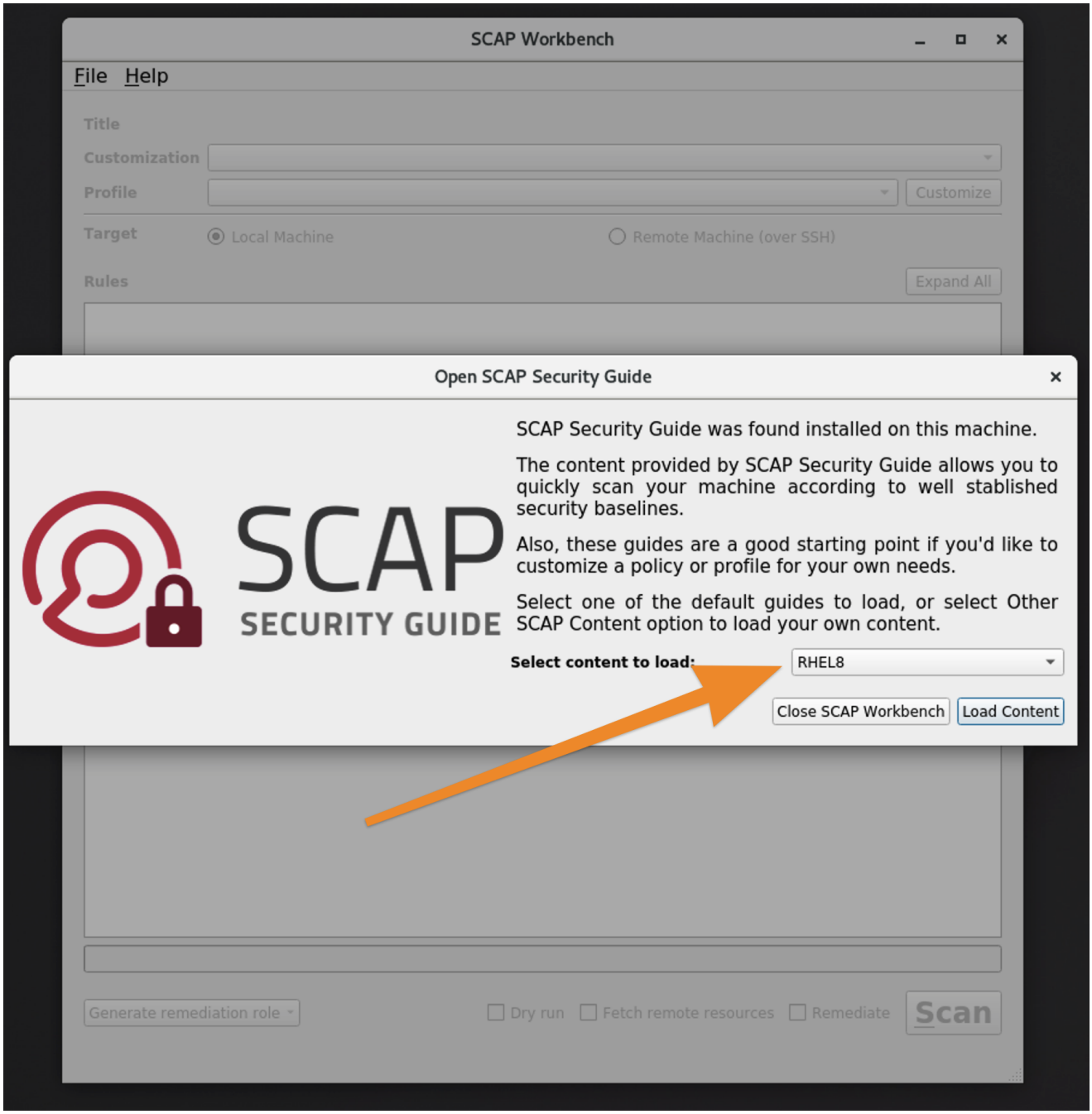

Figure 16. SCAP Workbench Startup

Step 4 - Creating Custom Content

Once SCAP Workbench has been launched, select the content to load. For this lab, we will be using the RHEL7 content.

Example 4. Creating Custom Content

1. For **Select content to load:** select "**RHEL8**", then click "**Load Content**"
2. Select the **Profile** you want to use to start customization

For this example, we will use the **OSPP - Protection Profile for General Purpose Operating Systems** Baseline

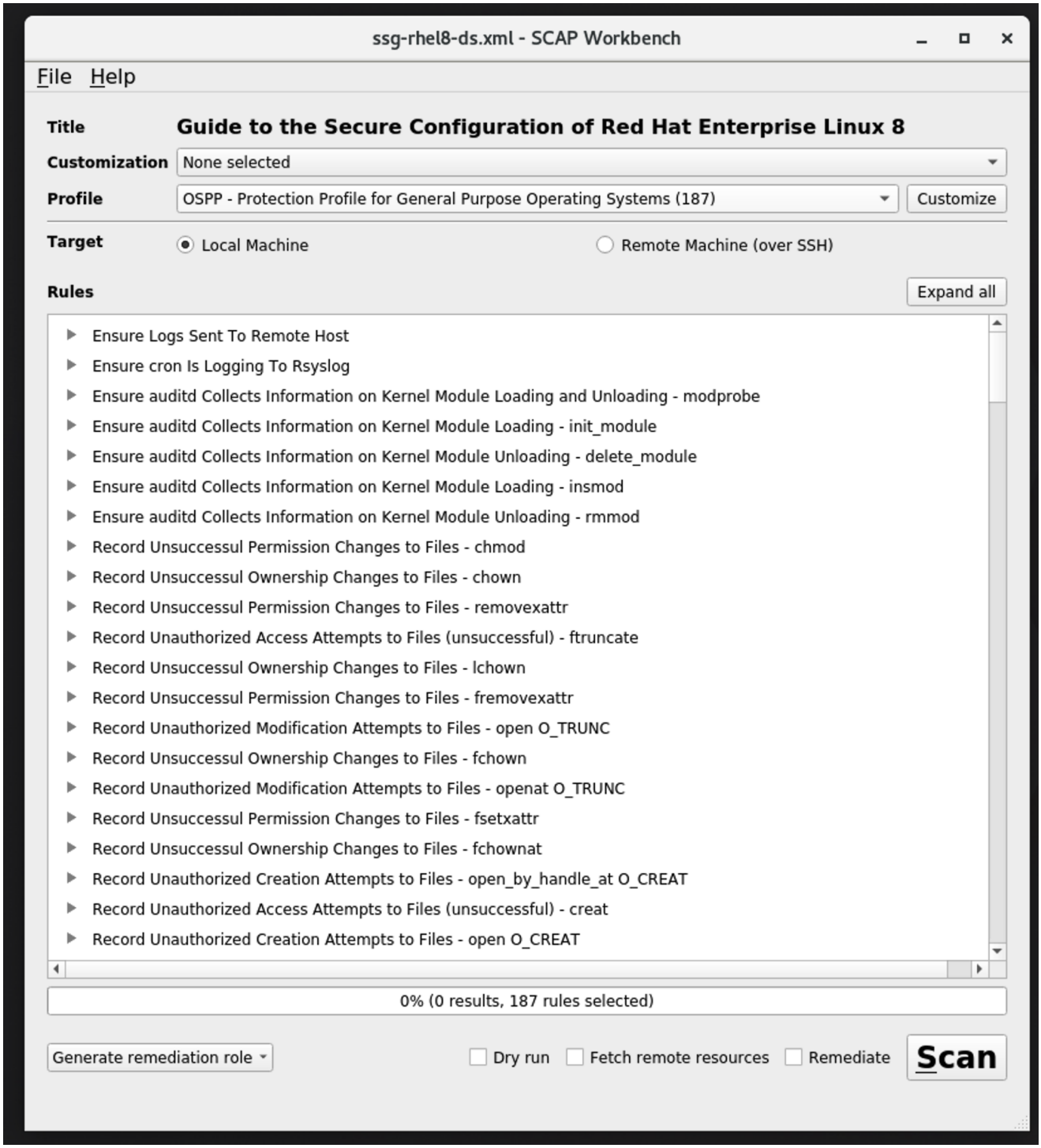

Figure 17. SCAP Workbench OSPP Profile

3. Click "**Customize**" to create custom SCAP content based on the chosen profile, and give it a name.

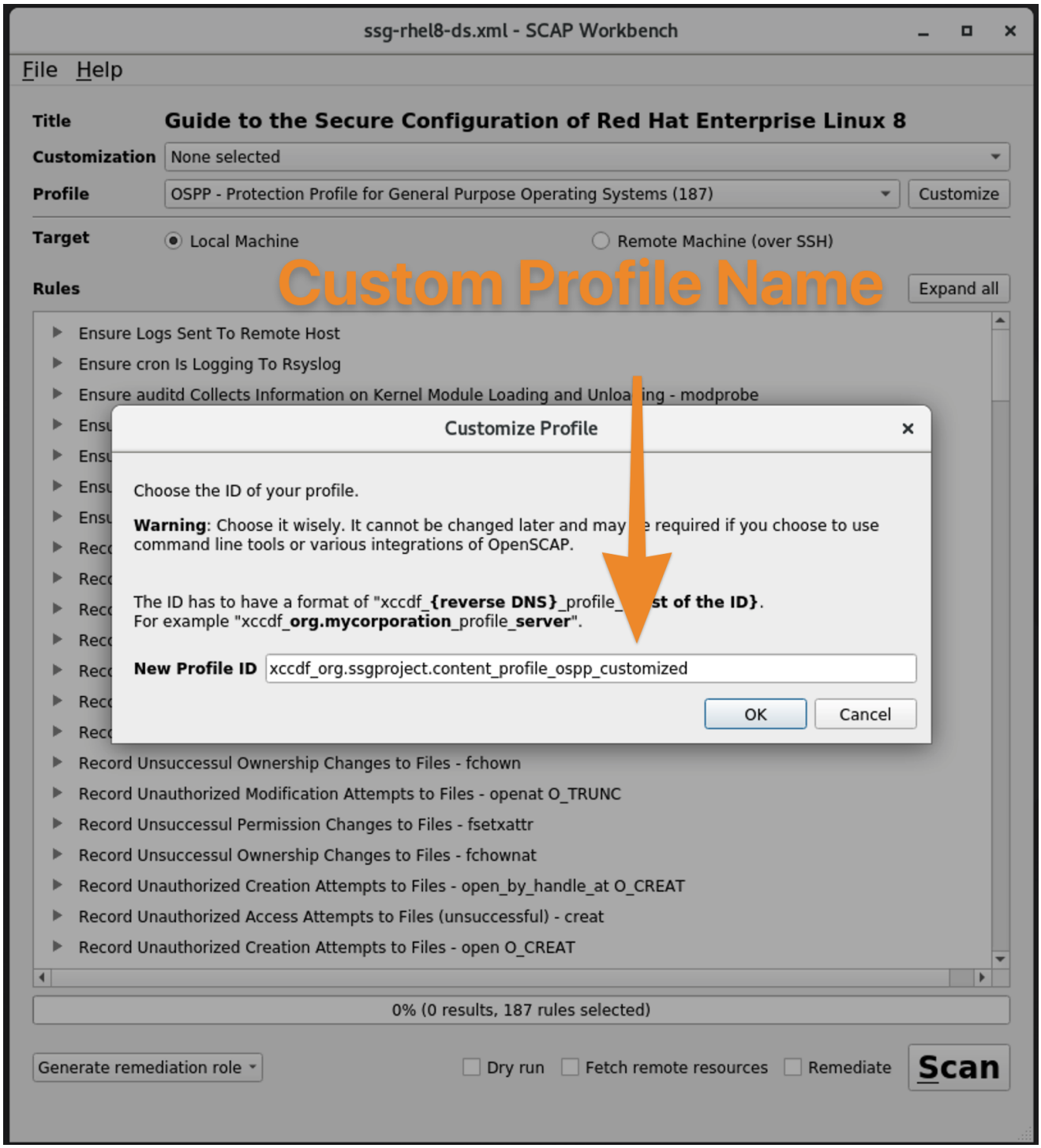

Figure 18. SCAP Custom OSPP Profile Creation

The name for this is: **xccdf_org.ssgproject.content_profile_ospp_customized**

4. Click "Deselect All" so that you can select the items you wish to include in your custom scan profile. **NOTE:** we are doing this to also limit it to a few checks for the example.

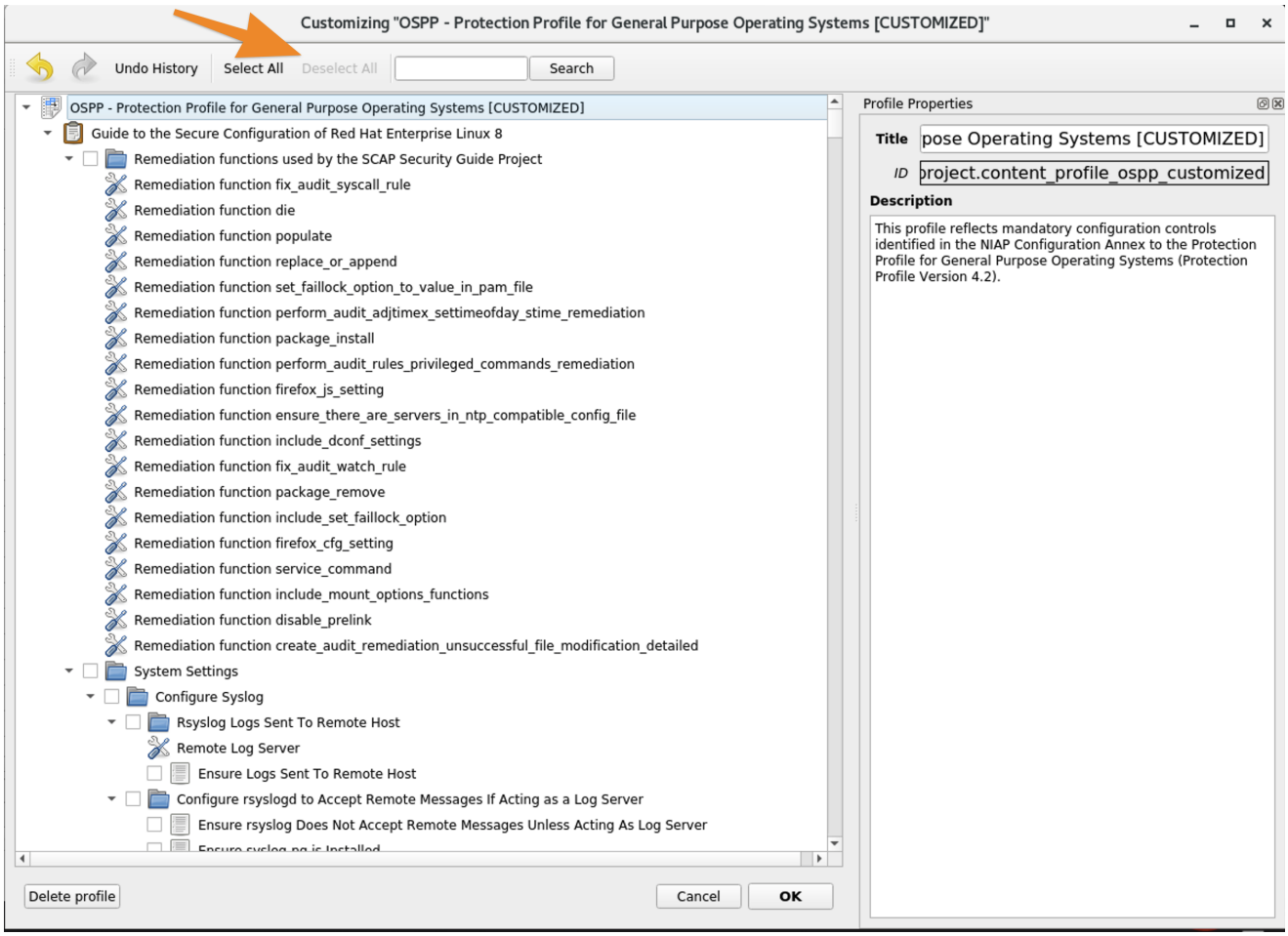

Figure 19. SCAP Custom Profile Selections

For this lab, we will be setting the minimum password length and PAM Password quality settings

5. Search for Password to set **minimum password length** and set the values in **login.defs**. Check **Set Password Minimum Length in login.defs** and click on the **minimum password legnth** and set the value to **18**

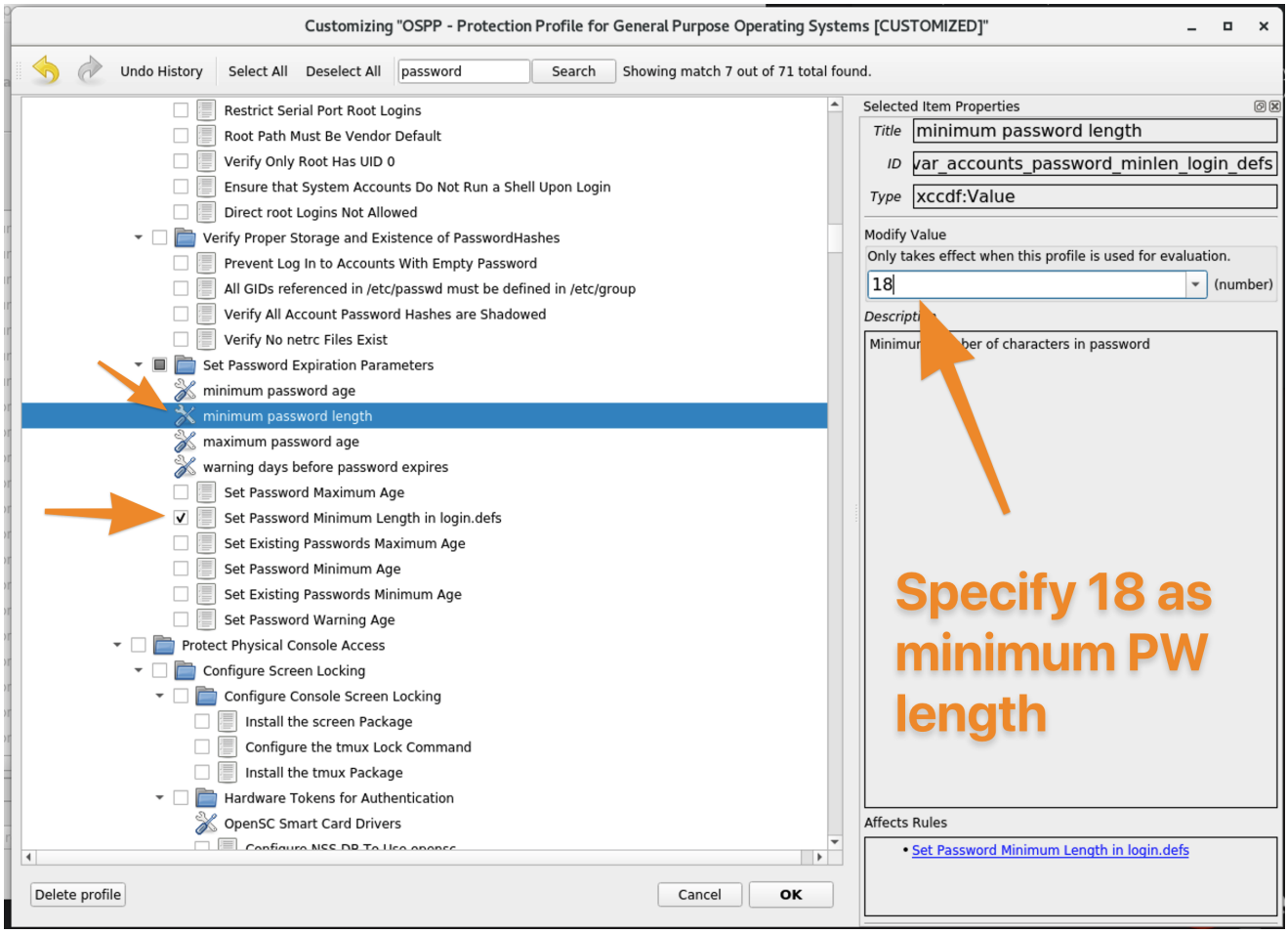

Figure 20. SCAP Custom Profile Password Settings for Login.Defs

6. Set password quality requirements with PAM. Search for the minlen and set it to **18**. Also, place a checkbox in **Set Password Quality Requirements with pam_quality**. Then click "OK"

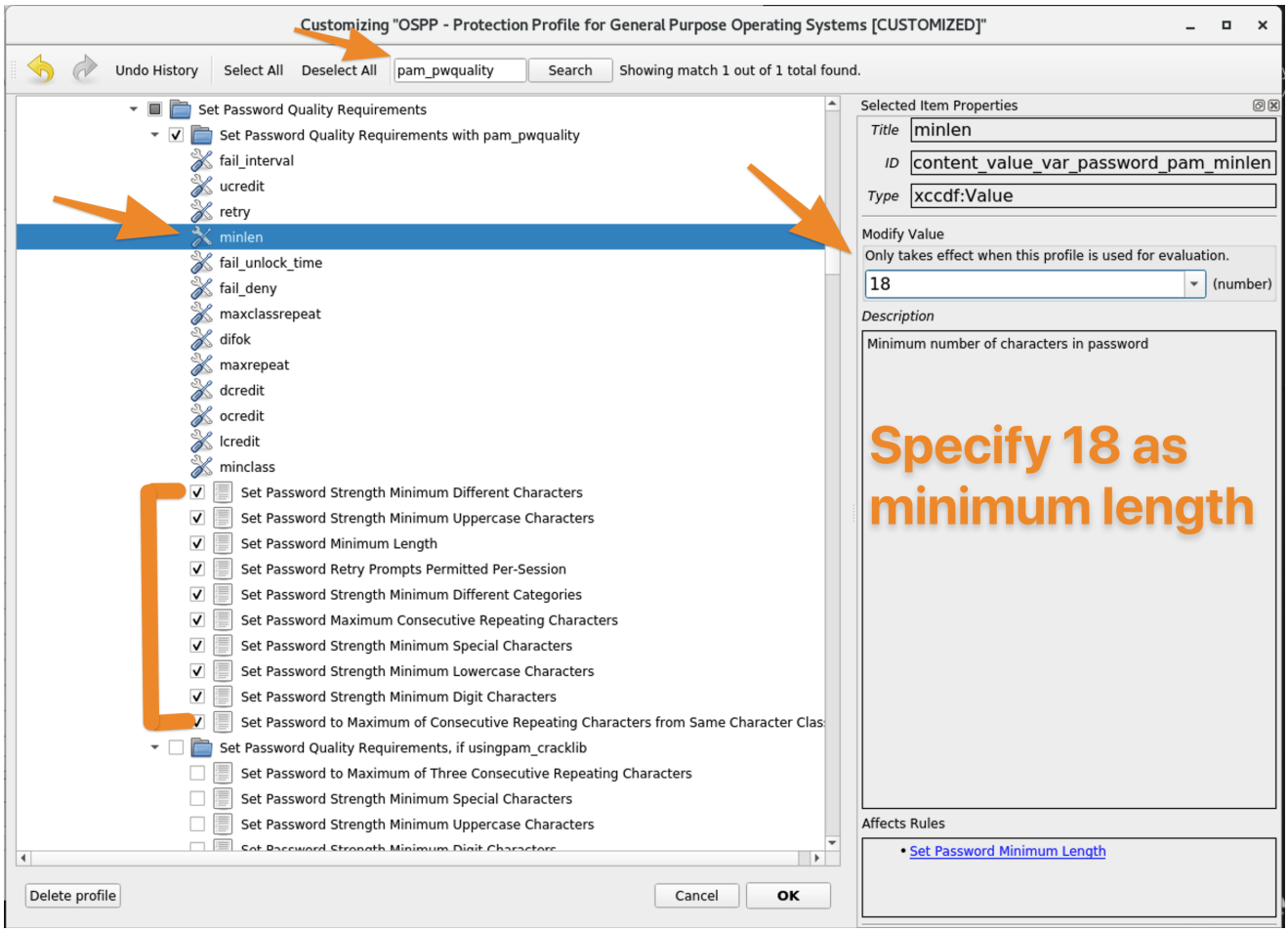

Figure 21. SCAP Custom Profile PAM Quality Requirements

7. At this point, we have taken the default settings from the OSPP profile with only the tailored pieces that we selected. The next step is to click "File ⇒ Save Customization Only" to save the custom content

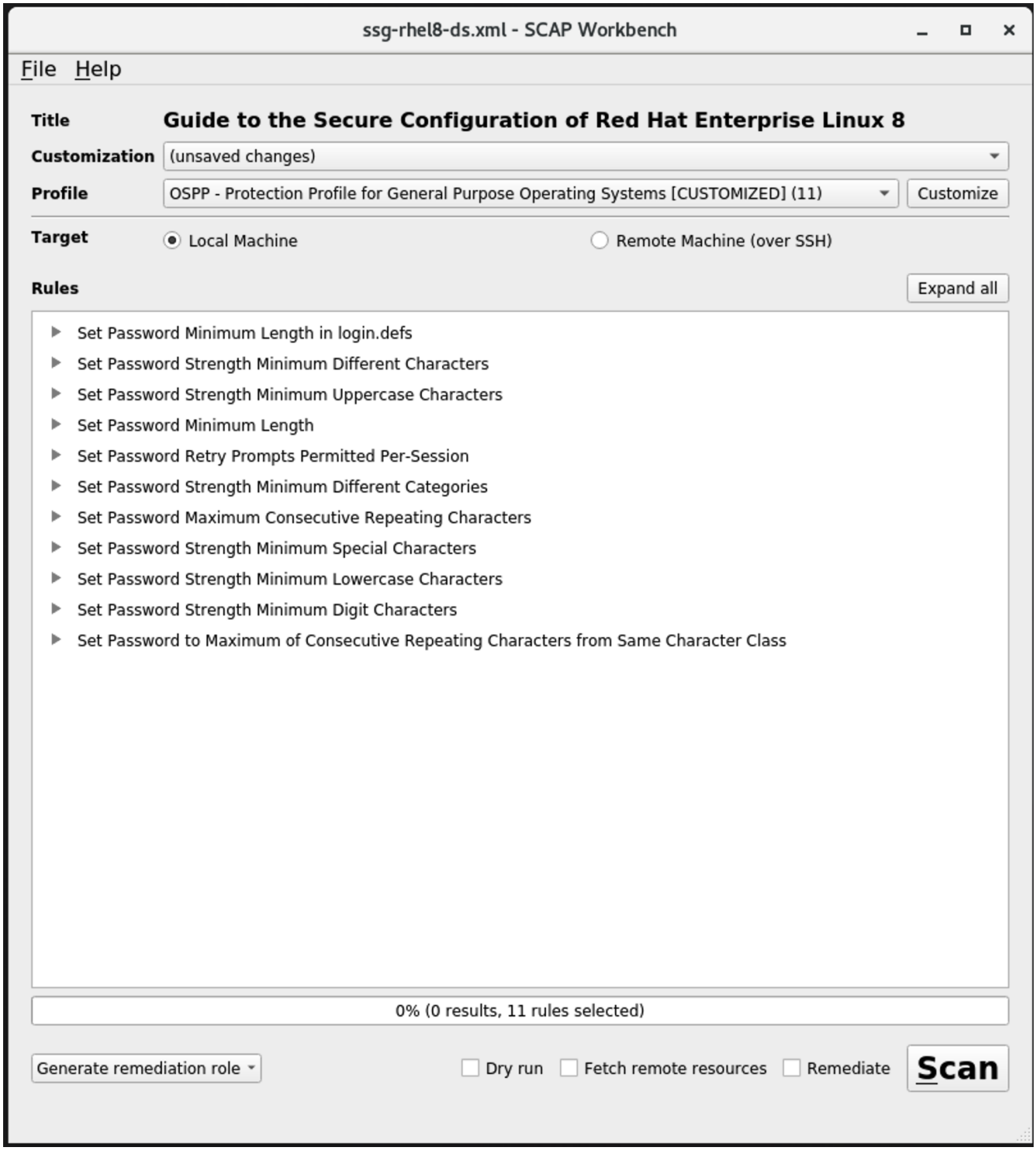

Figure 22. SCAP Custom Profile Selected Settings View

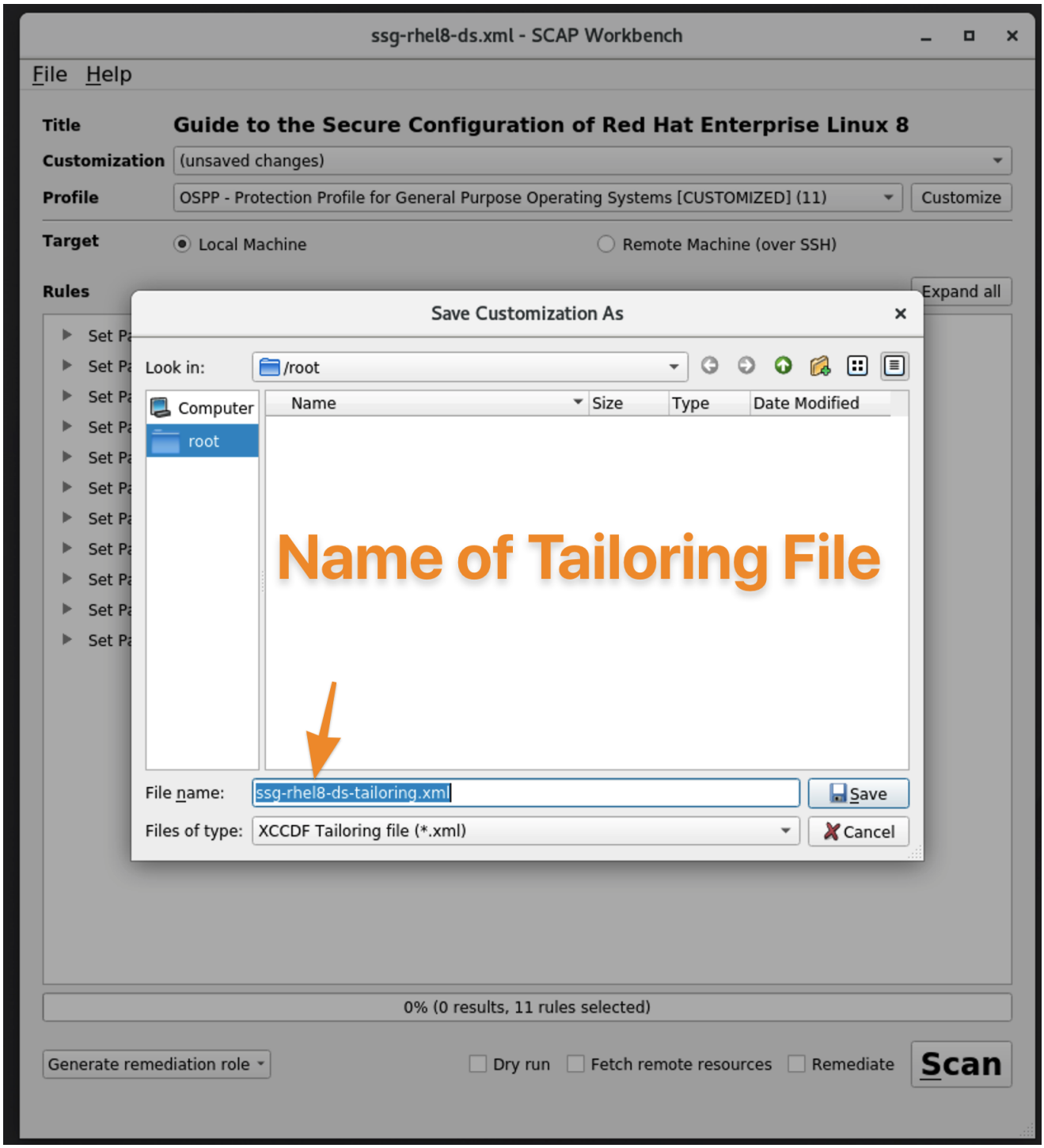

Figure 23. SCAP Custom Profile Creation Saving

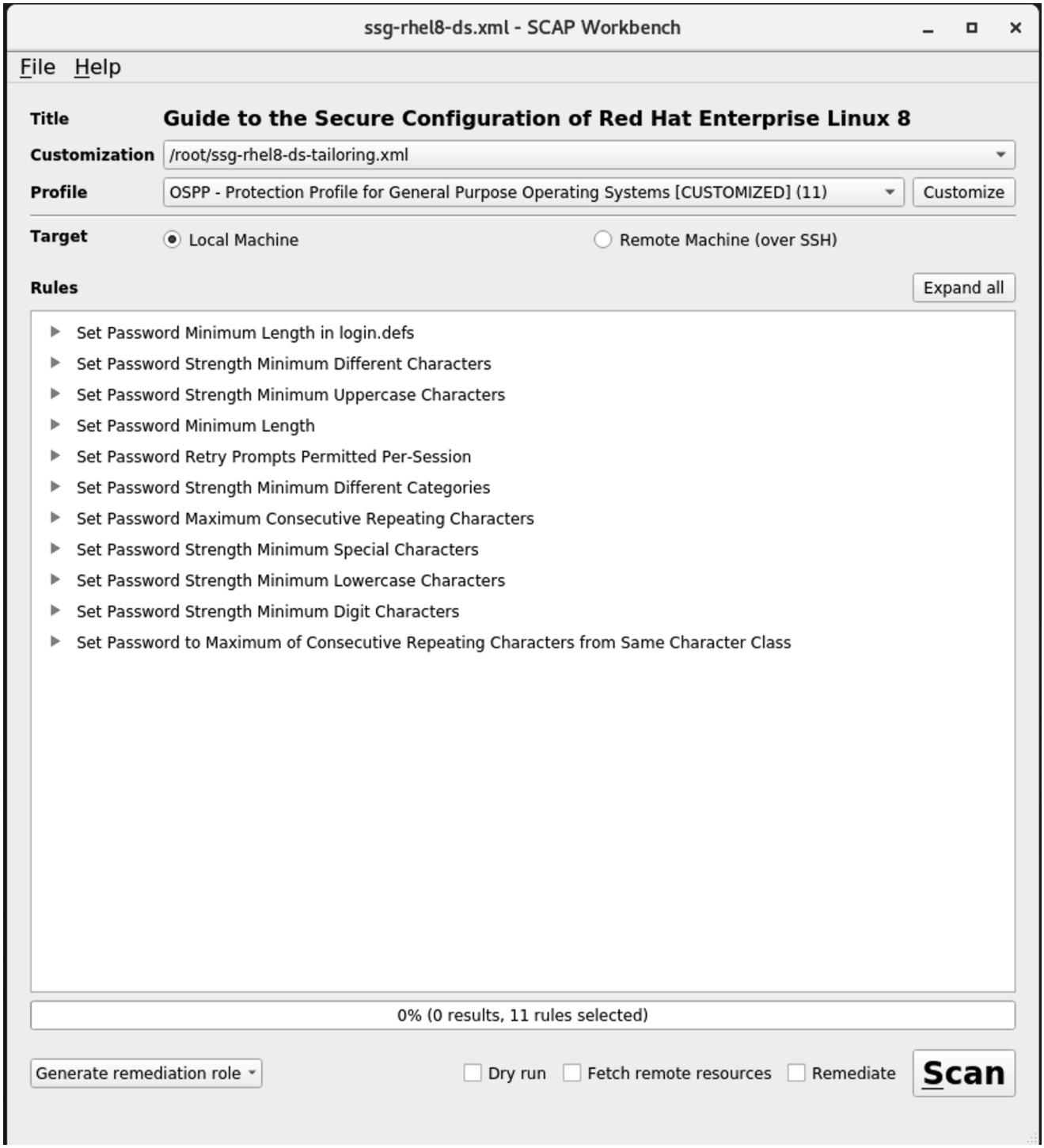

Figure 24. SCAP Custom Profile Final View

8. Copy the custom tailoring file to the server(s) being scanned. In this case, we will want to copy the file to **servera**

Listing 13. Copy custom content

```
[root@workstation ~]# scp ssg-rhel8-ds-tailoring.xml root@servera:
ssg-rhel8-ds-tailoring.xml                    100%   28KB  10.7MB/s   00:00
[root@workstation ~]#
```

2.2. Running a SCAP Scan with Custom Content

Servers to Configure

- servera

Packages to Install

- openscap-scanner
- scap-security-guide

Step 1 - SSH to serverc

The fist step is to connect to the server.

Example 5. Connecting to the **servera** *VM*

Listing 14. Connecting to **servera** *Using SSH*

```
# ssh root@servera
```

Step 2 - Install packages on servera

The second step is to install software on the server.

Example 6. Install software on **servera**

Listing 15. Installing Software on **servera**

```
[root@servera ~]# yum install scap-security-guide
Last metadata expiration check: 0:47:44 ago on Thu 16 Apr 2020 08:26:15 AM EDT.
Dependencies resolved.
================================================================================
 Package          Arch   Version       Repository                          Size
================================================================================
Installing:
 scap-security-guide
                  noarch 0.1.42-11.el8 rhel-8.0-for-x86_64-appstream-rpms 3.4 M
Installing dependencies:
 openscap         x86_64 1.3.0-7.el8   rhel-8.0-for-x86_64-appstream-rpms 3.3 M
 openscap-scanner x86_64 1.3.0-7.el8   rhel-8.0-for-x86_64-appstream-rpms  66 k
 xml-common       noarch 0.6.3-50.el8  rhel-8.0-for-x86_64-baseos-rpms     39 k

Transaction Summary
================================================================================
Install  4 Packages

Total download size: 6.9 M
Installed size: 132 M
Is this ok [y/N]: y


... output omitted ...

  Verifying        : openscap-1.3.0-7.el8.x86_64                              1/4
  Verifying        : openscap-scanner-1.3.0-7.el8.x86_64                      2/4
  Verifying        : scap-security-guide-0.1.42-11.el8.noarch                 3/4
  Verifying        : xml-common-0.6.3-50.el8.noarch                           4/4

Installed:
  scap-security-guide-0.1.42-11.el8.noarch    openscap-1.3.0-7.el8.x86_64
  openscap-scanner-1.3.0-7.el8.x86_64         xml-common-0.6.3-50.el8.noarch

Complete!
```

Learning about SCAP Commands

The SSG man page is a very good source of information for usage of the **oscap** tool as well as provides examples of how to use the SCAP SSG Guide profiles itself.

Listing 16. Looking at **SCAP Security Guide (SSG)** *Man Page*

```
# man scap-security-guide

scap-security-guide(8)      System Manager's Manual     scap-security-guide(8)

NAME
       SCAP  Security Guide - Delivers security guidance, baselines, and asso-
       ciated validation mechanisms utilizing the Security Content  Automation
       Protocol (SCAP).


... output omitted ...

EXAMPLES
       To  scan  your  system utilizing the OpenSCAP utility against the ospp-
       rhel7 profile:

       oscap xccdf eval --profile  ospp-rhel7  --results  /tmp/`hostname`-ssg-
       results.xml  --report  /tmp/`hostname`-ssg-results.html  --oval-results
       /usr/share/xml/scap/ssg/content/ssg-rhel7-xccdf.xml
```

Listing 17. Looking at ***oscap*** *Man Page*

```
# man oscap

OSCAP(8)                System Administration Utilities               OSCAP(8)

NAME
       oscap - OpenSCAP command line tool

SYNOPSIS
       oscap  [general-options]  module operation [operation-options-and-argu-
       ments]

DESCRIPTION
       oscap is Security Content Automation Protocol (SCAP) toolkit  based  on
       OpenSCAP  library.  It  provides  various  functions for different SCAP
       specifications (modules).

       OpenSCAP tool claims to provide capabilities of Authenticated  Configu-
       ration  Scanner  and  Authenticated Vulnerability Scanner as defined by
       The National Institute of Standards and Technology.

... output omitted ...

EXAMPLES
       Evaluate XCCDF content using CPE dictionary and produce html report. In
       this  case  we  use  United  States  Government  Configuration Baseline
       (USGCB) for Red Hat Enterprise Linux 5 Desktop.

               oscap xccdf eval --fetch-remote-resources --oval-results \
                       --profile united_states_government_configuration_baseline
 \
                       --report usgcb-rhel5desktop.report.html \
                       --results usgcb-rhel5desktop-xccdf.xml.result.xml \
                       --cpe usgcb-rhel5desktop-cpe-dictionary.xml \
                       usgcb-rhel5desktop-xccdf.xml
```

Step 3 - Running *oscap* scan

We will run the **oscap** utility to generate a report and a results file that can be sent back to the **workstation** system so that we can create an Ansible playbook for remediation and view the results of the report.

Be very careful about the name of the profile as this was selected during the creation of the custom profile/tailoring file portion when doing SCAP Workbench customizations.

Example 7. Scanning **servera**

Listing 18. Using **oscap** *and the tailoring profile to scan* **servera**

```
# [root@servera ~]# oscap xccdf eval \
--profile xccdf_org.ssgproject.content_profile_ospp_customized \
--tailoring-file ssg-rhel8-ds-tailoring.xml \
--results custom_scan_results.xml \
/usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml

Title   Set Password Minimum Length in login.defs
Rule    xccdf_org.ssgproject.content_rule_accounts_password_minlen_login_defs
Ident   CCE-80652-1
Result  fail

Title   Set Password Strength Minimum Different Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_difok
Ident   CCE-80654-7
Result  fail

Title   Set Password Strength Minimum Uppercase Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_ucredit
Ident   CCE-80665-3
Result  fail

Title   Set Password Minimum Length
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_minlen
Ident   CCE-80656-2
Result  fail

Title   Set Password Retry Prompts Permitted Per-Session
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_retry
Ident   CCE-80664-6
Result  fail

Title   Set Password Strength Minimum Different Categories
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_minclass
Result  fail

Title   Set Password Maximum Consecutive Repeating Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_maxrepeat
Result  fail

Title   Set Password Strength Minimum Special Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_ocredit
Ident   CCE-80663-8
Result  fail

Title   Set Password Strength Minimum Lowercase Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_lcredit
Ident   CCE-80655-4
Result  fail

Title   Set Password Strength Minimum Digit Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_dcredit
Ident   CCE-80653-9
Result  fail

Title   Set Password to Maximum of Consecutive Repeating Characters from Same Character Class
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_maxclassrepeat
Result  fail
```

Getting Custom Profile Name from Tailoring File

If you need to locate the profile used for the custom scanning content from the tailoring file, you can search for it with **grep**.

```
[root@servera ~]#  grep "Profile id" ssg-rhel8-ds-tailoring.xml
  <xccdf:Profile id="xccdf_org.ssgproject.content_profile_ospp_customized" extends
="xccdf_org.ssgproject.content_profile_ospp">
```

Step 4 - Creating a Results Report

You can create a results report file from the results file so you have a nice HTML file that is easy to ready with the results from the SCAP scan.

Example 8. Creating a SCAP Report from a Results File

Listing 19. Generating a Report

```
[root@servera ~]# oscap xccdf generate report \
custom_scan_results.xml > Custom_Scan_Report.html
```

Combining Steps 3 & 4

It is possible to perform a custom content scan which will generate the results file and the report for transfer back to the workstation for review.

Need to Specify

- **--results**
- **--report**

Listing 20. Creating a Results File and Report During Custom Content Scan

```
[root@servera ~]# oscap xccdf eval \
--profile xccdf_org.ssgproject.content_profile_ospp_customized \
--tailoring-file ssg-rhel8-ds-tailoring.xml \
--results custom_scan_results_2.xml \
--report Custom_Scan_Report_2.html \
/usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml

Title   Set Password Minimum Length in login.defs
Rule    xccdf_org.ssgproject.content_rule_accounts_password_minlen_login_defs
Ident   CCE-80652-1
Result  fail

Title   Set Password Strength Minimum Different Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_difok
Ident   CCE-80654-7
Result  fail

Title   Set Password Strength Minimum Uppercase Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_ucredit
Ident   CCE-80665-3
Result  fail

Title   Set Password Minimum Length
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_minlen
Ident   CCE-80656-2
Result  fail

Title   Set Password Retry Prompts Permitted Per-Session
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_retry
Ident   CCE-80664-6
Result  fail

Title   Set Password Strength Minimum Different Categories
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_minclass
Result  fail

Title   Set Password Maximum Consecutive Repeating Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_maxrepeat
Result  fail

Title   Set Password Strength Minimum Special Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_ocredit
Ident   CCE-80663-8
Result  fail

Title   Set Password Strength Minimum Lowercase Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_lcredit
Ident   CCE-80655-4
Result  fail

Title   Set Password Strength Minimum Digit Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_dcredit
Ident   CCE-80653-9
Result  fail

Title   Set Password to Maximum of Consecutive Repeating Characters from Same Character Class
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_maxclassrepeat
Result  fail
```

Step 5 - Transferring Results File and Report to Workstation

After you have the results files and the report, you should transfer it to your graphical workstation (**workstation**) for further analysis.

Example 9. Transferring Results

Listing 21. Transferring the Results and Report Files

```
[root@servera ~]# scp *.xml *.html root@workstation:
The authenticity of host 'workstation (<no hostip for proxy command>)' can't be established.
ECDSA key fingerprint is SHA256:pOQ1OJmyF2PFI+jxyFoOSCfi+1oWNsUruy2DZNjg+N0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'workstation' (ECDSA) to the list of known hosts.
root@workstation's password:
custom_scan_results_2.xml                     100% 4086KB  32.4MB/s   00:00
custom_scan_results.xml                       100% 4086KB  60.0MB/s   00:00
ssg-rhel8-ds-tailoring.xml                    100%   28KB  16.2MB/s   00:00
Custom_Scan_Report_2.html                     100%  332KB  44.3MB/s   00:00
Custom_Scan_Report.html                       100%  332KB  37.6MB/s   00:00
[root@servera ~]#
```

Step 6 - Viewing the SCAP scan report

After you have transferred the results file to **workstation** you can open the HTML report in a web browser. In this case we will use ***firefox*** to open the file.

Example 10. Viewing the SCAP Report

Listing 22. Opening the SCAP HTML Report with Firefox

```
[root@workstation ~]# firefox Custom_Scan_Report.html
```

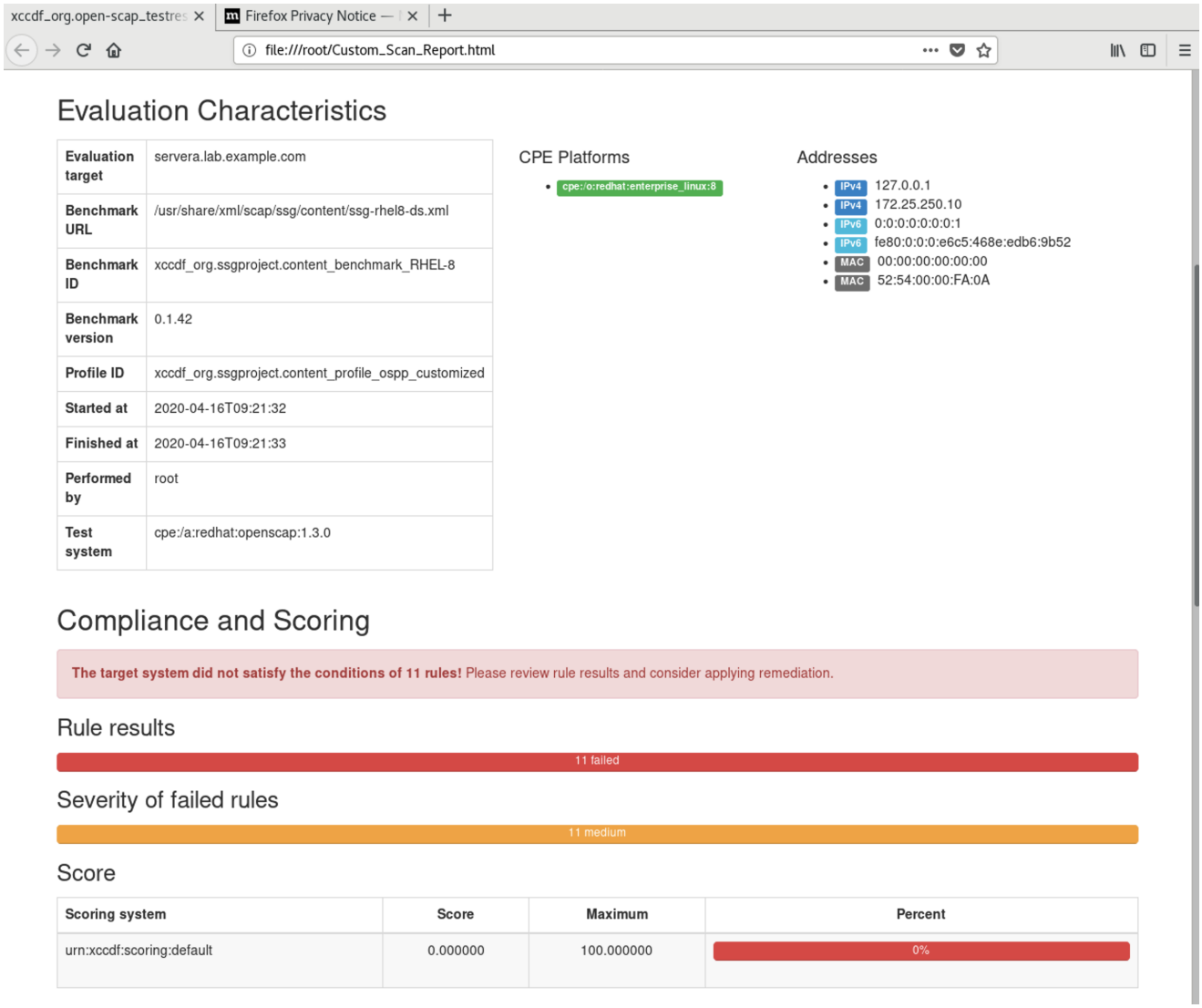

Figure 25. SCAP Scan Results Report in Firefox

Firefox may not open the file based on SELinux context triggers. In order to get around this you can use the command prompt and do **setenforce 0** to allow you to open the report.

2.3. Creating an Ansible Remediation Playbook Based on SCAP Scan Results

The OpenSCAP project and content created by Red Hat can automatically remediate findings from OpenSCAP scans. The findings can be remediated in many ways (**BASH**, **Ansible**, etc.). While things are mostly complete, there are some automated remediations that have not yet been developed.

There are multiple automatic remediation methods developed, but at this time, there isn't a script to fix everything.

Servers to Configure

- servera

We will continue to use **workstation** as our master SCAP system as it should have Ansible and SCAP Workbench installed.

Step 1 - Creating an Ansible Playbook from Results

The first step will be to generate an Ansible playbook from the SCAP scan results for system remediation.

Example 11. Generating Ansible Playbook

Listing 23. Ansible Playbook Generation

```
[root@workstation ~]# oscap xccdf generate fix \
--profile xccdf_org.ssgproject.content_profile_ospp_customized \
--tailoring-file ssg-rhel8-ds-tailoring.xml \
--fix-type ansible \
--result-id "" \
custom_scan_results.xml > Custom_Scan_Fix.yml
```

Viewing Remediation Playbook

It is also a good idea to view the created playbook for the system prior to running it.

```
[root@workstation ~]# cat Custom_Scan_Fix.yml
---
#################################################################################
#
# Ansible remediation role for the results of evaluation of profile
xccdf_org.ssgproject.content_profile_ospp_customized
# XCCDF Version:  unknown
#
# Evaluation Start Time:  2020-04-16T09:21:32
# Evaluation End Time:  2020-04-16T09:21:33
#
# This file was generated by OpenSCAP 1.3.0 using:
#   $ oscap xccdf generate fix --result-id xccdf_org.open-
scap_testresult_xccdf_org.ssgproject.content_profile_ospp_customized --template urn:xccdf:fix:script:ansible
xccdf-results.xml
#
# This script is generated from the results of a profile evaluation.
```

```
# It attempts to remediate all issues from the selected rules that failed the test.
#
# How to apply this remediation role:
# $ ansible-playbook -i "localhost," -c local playbook.yml
# $ ansible-playbook -i "192.168.1.155," playbook.yml
# $ ansible-playbook -i inventory.ini playbook.yml
#
################################################################################


 - hosts: all
   vars:
      var_accounts_password_minlen_login_defs: !!str 18
      var_password_pam_difok: !!str 8
      var_password_pam_ucredit: !!str -1
      var_password_pam_minlen: !!str 18
      var_password_pam_retry: !!str 3
      var_password_pam_minclass: !!str 3
      var_password_pam_maxrepeat: !!str 3
      var_password_pam_ocredit: !!str -1
      var_password_pam_lcredit: !!str -1
      var_password_pam_dcredit: !!str -1
      var_password_pam_maxclassrepeat: !!str 4
   tasks:


    - name: "Set Password Minimum Length in login.defs"
      lineinfile:
        dest: /etc/login.defs
        regexp: "^PASS_MIN_LEN *[0-9]*"
        state: present
        line: "PASS_MIN_LEN        {{ var_accounts_password_minlen_login_defs }}"
      tags:
        - accounts_password_minlen_login_defs
        - medium_severity
        - restrict_strategy
        - low_complexity
        - low_disruption
        - CCE-80652-1
        - NIST-800-53-IA-5(f)
        - NIST-800-53-IA-5(1)(a)
        - NIST-800-171-3.5.7
        - CJIS-5.6.2.1


... Output Omitted ...

    - name: Ensure PAM variable maxclassrepeat is set accordingly

      lineinfile:
        create: yes
        dest: "/etc/security/pwquality.conf"
        regexp: '^#?\s*maxclassrepeat'
        line: "maxclassrepeat = {{ var_password_pam_maxclassrepeat }}"

      tags:
        - accounts_password_pam_maxclassrepeat
        - medium_severity
        - restrict_strategy
        - low_complexity
        - low_disruption
        - NIST-800-53-IA-5
        - NIST-800-53-IA-5(c)
```

Ansible is not setup for the lab

Before we can do the next steps, we will download an Ansible config file and an inventory file so we can properly run the playbook.

Listing 24. Error Output Message

```
[root@workstation ~]# ansible-playbook Custom_Scan_Fix.yml
 [WARNING]: provided hosts list is empty, only localhost is available. Note
that the implicit localhost does not match 'all'


PLAY [all] *********************************************************************
skipping: no hosts matched

PLAY RECAP *********************************************************************
[root@workstation ~]#
```

Step 2 - Downloading Ansible Config and Ansible Inventory Files

This step is needed so that our Ansible system can be configured with various configuration options and the inventory files so we can run the given playbook.

Example 12. Downloading Ansible Files

Listing 25. Downloading Ansible Files

```
[root@workstation ~]# wget http://people.redhat.com/~tmichett/rh354/inventory
--2020-04-16 09:38:50--  http://people.redhat.com/~tmichett/rh354/inventory
Resolving people.redhat.com (people.redhat.com)... 209.132.183.19
Connecting to people.redhat.com (people.redhat.com)|209.132.183.19|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 24
Saving to: 'inventory'

inventory           100%[==================>]      24  --.-KB/s    in 0s

2020-04-16 09:38:50 (2.69 MB/s) - 'inventory' saved [24/24]


[root@workstation ~]# wget http://people.redhat.com/~tmichett/rh354/ansible.cfg
--2020-04-16 09:39:34--  http://people.redhat.com/~tmichett/rh354/ansible.cfg
Resolving people.redhat.com (people.redhat.com)... 209.132.183.19
Connecting to people.redhat.com (people.redhat.com)|209.132.183.19|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 159
Saving to: 'ansible.cfg'

ansible.cfg         100%[==================>]     159  --.-KB/s    in 0s

2020-04-16 09:39:35 (13.8 MB/s) - 'ansible.cfg' saved [159/159]
```

Reviewing Ansible Configurations

The **inventory** file provided only has a single host **servera** in there. On real systems, you must be very cautious of running remediation playbooks against an inventory file as it could apply to unintended systems. Additionally the **ansible.cfg** file provided was created for use in this lab environment. Both of these items should be taken into account when doing going through the process on production systems.

```
[root@workstation ~]# cat inventory
servera.lab.example.com

[root@workstation ~]# cat ansible.cfg
[defaults]
roles_path = /etc/ansible/roles:/usr/share/ansible/roles
log_path   = /tmp/ansible.log
inventory  = ./inventory

[privilege_escalation]
become=True
[root@workstation ~]#
```

Step 3 - Run the Ansible Playbook

This step will utilize the **workstation** system which is configured as your Ansible management node and will run the playbook to remediate the results on the **servera** system.

*Example 13. Remediation of **serverc** with Ansible Playbook*

Listing 26. Running the Ansible Playbook

```
[root@workstation ~]# ansible-playbook Custom_Scan_Fix.yml

PLAY [all] *********************************************************************

TASK [Gathering Facts] *********************************************************
ok: [servera.lab.example.com]

TASK [Set Password Minimum Length in login.defs] *******************************
changed: [servera.lab.example.com]

TASK [Ensure PAM variable difok is set accordingly] ****************************
changed: [servera.lab.example.com]

TASK [Ensure PAM variable ucredit is set accordingly] **************************
changed: [servera.lab.example.com]

TASK [Ensure PAM variable minlen is set accordingly] ***************************
changed: [servera.lab.example.com]

TASK [Set Password Retry Prompts Permitted Per-Session - system-auth (change)] ***
ok: [servera.lab.example.com]

TASK [Set Password Retry Prompts Permitted Per-Session - system-auth (add)] ****
changed: [servera.lab.example.com]

TASK [Ensure PAM variable minclass is set accordingly] *************************
changed: [servera.lab.example.com]

TASK [Ensure PAM variable maxrepeat is set accordingly] ************************
changed: [servera.lab.example.com]

TASK [Ensure PAM variable ocredit is set accordingly] **************************
changed: [servera.lab.example.com]

TASK [Ensure PAM variable lcredit is set accordingly] **************************
changed: [servera.lab.example.com]

TASK [Ensure PAM variable dcredit is set accordingly] **************************
changed: [servera.lab.example.com]

TASK [Ensure PAM variable maxclassrepeat is set accordingly] *******************
changed: [servera.lab.example.com]

PLAY RECAP *********************************************************************
servera.lab.example.com    : ok=13   changed=11   unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

After running the playbook, you can see that there were 10 changes that were made to the system and exactly which parameters were changed. The next thing to do is perform another scan of the system to ensure that it is now fully compliant.

Step 4 - Rescan System and Review Results

Example 14. Scanning System after Fixes and Verifying Results

Listing 27. Performing SCAP Verification Scan

```
[root@servera ~]# oscap xccdf eval \
--profile xccdf_org.ssgproject.content_profile_ospp_customized \
--tailoring-file ssg-rhel8-ds-tailoring.xml \
--results custom_scan_results_fixed.xml \
--report Custom_Scan_Report_Fixed.html \
/usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

Listing 28. Copying Results to Workstation

```
[root@servera ~]# scp custom_scan_results_fixed.xml Custom_Scan_Report_Fixed.html workstation:
root@workstation's password:
custom_scan_results_fixed.xml                 100% 4086KB  60.3MB/s   00:00
Custom_Scan_Report_Fixed.html                 100%  282KB  48.4MB/s   00:00
```

Listing 29. Viewing Results on Workstation

```
[root@workstation ~]# firefox Custom_Scan_Report_Fixed.html
```

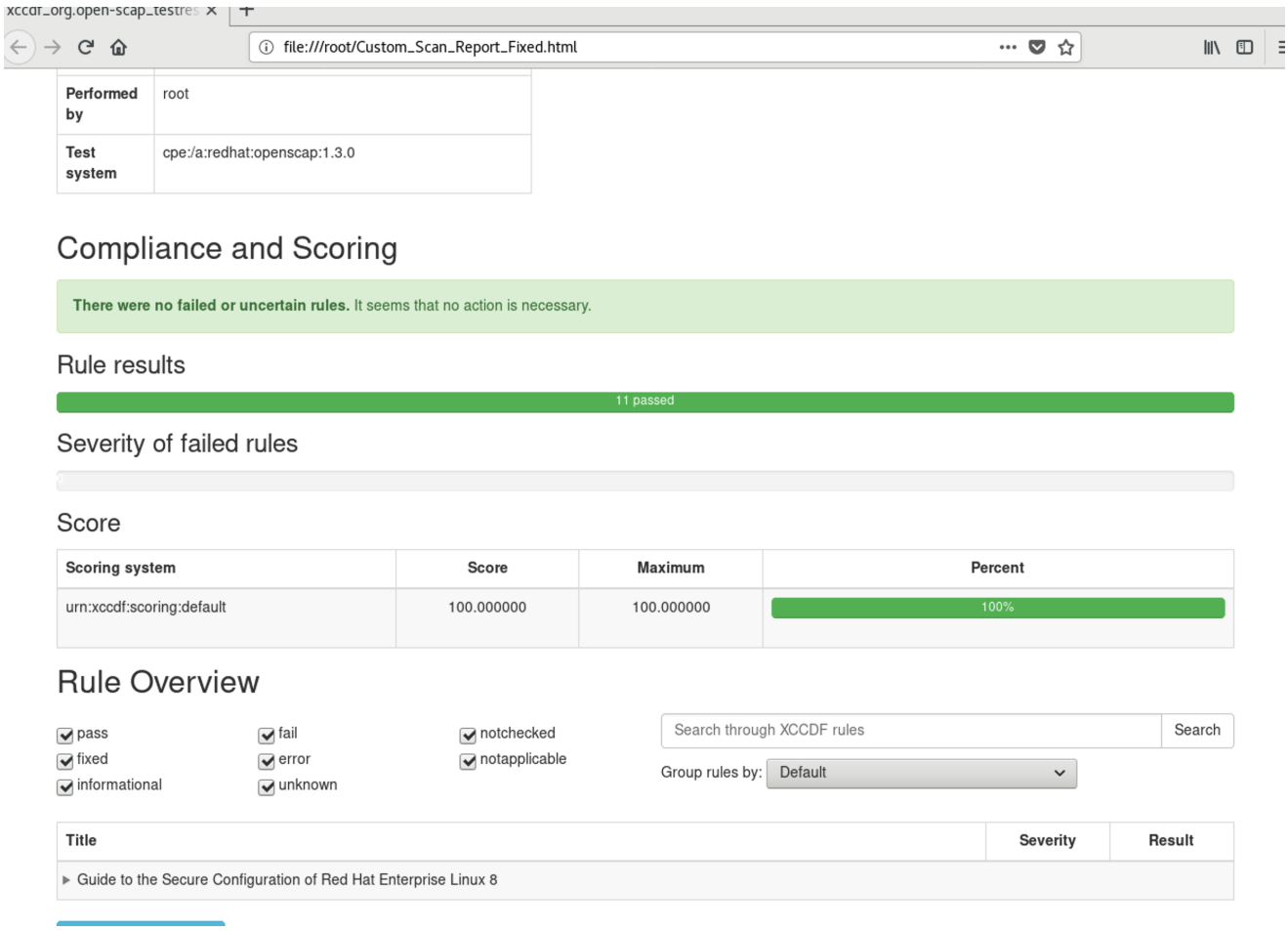

Figure 26. Fixed SCAP Scan Results Report in Firefox