RED HAT®
CONSULTING

Encryption and Security

Travis Michette

Version 1.3

Table of Contents

1. System Security Fundamentals

There are multiple steps to system security. At the most basic levels, the following items need to be protected:

- Access to the System
- Access to the Data

The above items are the starting points for system security. This guide will focus mainly on data security and providing some hands-on labs which will demonstrate encrypting and decrypting of disks using LUKS. The guide will also provide a recap and overview of system security with OpenSCAP and have a hands-on lab for generating Ansible remediation playbooks based on SCAP compliance scans.

1.1. Protecting System Integrity

Protecting system integrity includes protecting the following:

- Access to the system (physical)
  - Locked room
  - On person
  - Network (network-based firewalls)
  - etc.
- Access to the system (logical)
  - Password
  - Network (host-based firewalls)
  - Encryption
  - etc.
- Valid Operating System and Configurations
  - Installed from trusted source
  - Updated and patched
  - Configured with recommended lockdown settings

Most of the system integrity and security concepts will not be discussed as part of this demonstration. We will mainly be focusing on the data security portion of encryption and a brief review of OpenSCAP as well as showing a hands-on demo of extending the SCAP presentation from a previous MeetUp.

1.2. Protecting System Data

One of the most valuable portions of a system is the actual data it contains. There are many things that can be done to protect both the system and the data. In addition to a solid backup procedure, there should be steps taken for protecting data should someone get unauthorized access to a computer. Most newer computer systems allow for encrypting. Windows has Bitlocker, MacOS allows for filesystem security, and Linux provides LUKS for disk encryption. There are also several other multi-platform,

third-party utilities for disk and file encryption. An older open-source encryption utility, TrueCrypt has been forked and is now released under several names, one of which is VeraCrypt.

VeraCrypt can be obtained from: https://www.veracrypt.fr/en/Home.html

With the portability of thumb drives, laptops, and other devices, it is critical to have personal and private information protected should the device become comprimised, stolen, or lost.

2. Data Encryption

2.1. Using LUKS to Encrypt Disks on Linux Systems

LUKS uses dmcrypt in Linux as part of the kernel to interact with block storage devices and allow encryption. The application/utility that will be used as part of the hands-on exercise is **cryptsetup**. The diagram in **Figure 1** shows the basic layers involved with LUKS. Fortunately, Linux has all the pieces needed to complete end-to-end encryption of block devices using LUKS.

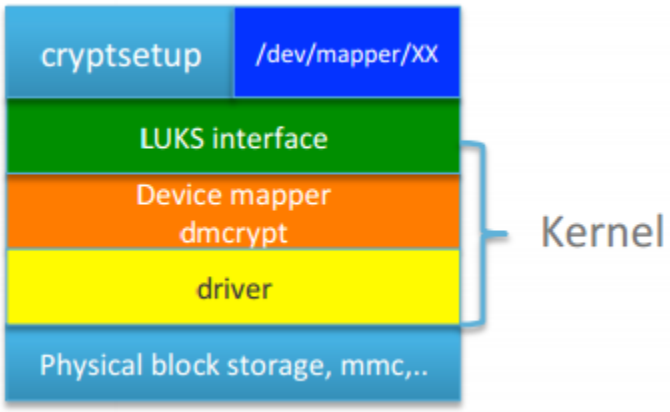

Figure 1. LUKS Layers and Interface Components

In the first portion of the encryption lab, we will be creating a partition and utilizing **cryptsetup** to prepare our disk partition to be encryped with LUKS.

The next diagram gives shows what is contained in the LUKS header on an encrypted volume/disk. It is important to note that in addition to the header, there are **key slots** which allow for multiple encryption/decryption keys to be stored.

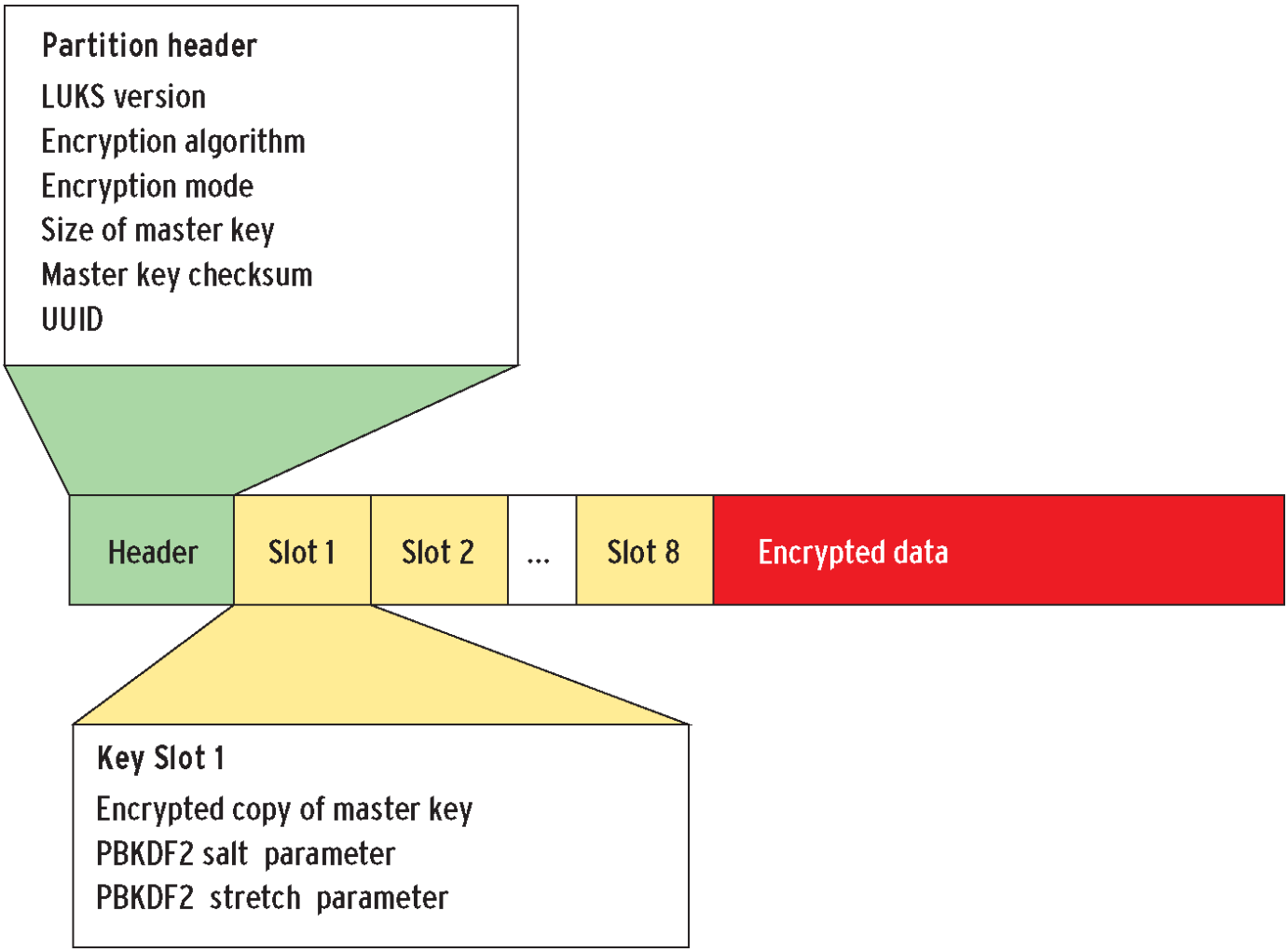

Figure 2. LUKS Partition Header

After the creation of the LUKS volume, we will use the **cryptsetup** command to dump the LUKS information for the encrypted volume to the screen.

It is a good practice to have a backup of LUKS header information in the event a volume gets damaged or something happens to the original header. Without LUKS header backup and metadata, if something goes wrong, the data becomes lost.

The diagram below shows a typical LUKS encryption setup in which the password and parameters can be provided all as part of the **/etc/crypttab** file or have human intervention in which the end-user is prompted for a password.

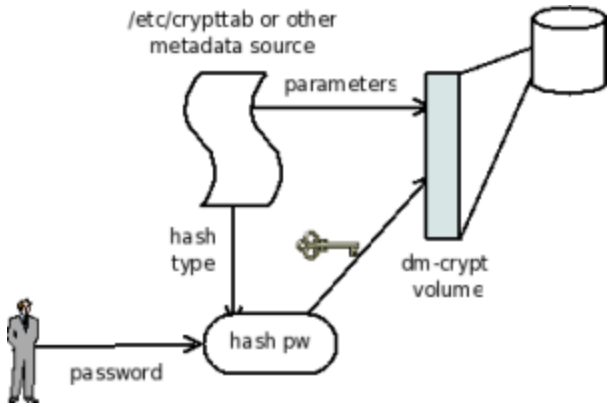

Figure 3. Traditional LUKS Password-Based Decryption Framework

For laptops or personal devices, it makes sense to prompt end-users for passwords as the device is typically easily accessible and there are typically few devices in use. For servers, the use of LUKS prompting for passwords can present challenges as typically servers reside in DataCenters with no keyboard/monitor. Security professionals generally frown on keeping the decryption key on the local drive which can be accessed by anyone, but allows for the system to be booted without prompting for a decryption password. Because of these scenarios, Network Bound Disk Encryption (NBDE) is often used as a solution for servers in a DataCenter environment.

2.1.1. Creating an Encrypted Disk with LUKS

The first portion of the lab will be to create the encrypted volume. For the sake of time, the virtual HDDs have been already applied to the systems and these will be used as part of the partition creation process and LUKS creation process.

Servers to Use

- servera

Step 1 - Looking at Disks and Partitions

*Example 1. Use **parted** to find partition details*

Listing 1. parted to list disks

```
[root@servera ~]# parted -l
Model: Virtio Block Device (virtblk)
Disk /dev/vda: 10.7GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type     File system  Flags
 1      1049kB  10.7GB  10.7GB  primary  xfs          boot


Error: /dev/vdb: unrecognised disk label
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 1074MB
Sector size (logical/physical): 512B/512B
Partition Table: unknown
Disk Flags:

[root@servera ~]#
```

Step 2 - Creating a Disk Partition

*Example 2. Use **parted** to Create a Partition*

Listing 2. Create Partition

```
# parted /dev/vdb \
 mklabel msdos \
 mkpart primary xfs 1M 1G
```

Listing 3. Verify Partition

```
# parted /dev/vdb print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 1074MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type     File system  Flags
 1      1049kB  1074MB  1073MB  primary
```

Step 3 - Preparing a Disk Partition for LUKS

*Example 3. Use **cryptsetup** to Prepare Partition with LUKS*

Listing 4. LUKS Formatting

```
# cryptsetup luksFormat /dev/vdb1

WARNING!
========
This will overwrite data on /dev/vdb1 irrevocably.

Are you sure? (Type uppercase yes): YES
Enter passphrase: meetup2018
Verify passphrase: meetup2018
```

Step 4 - Opening an Encrypted LUKS Partition

*Example 4. Use **cryptsetup** to Open a LUKS Partition*

Listing 5. LUKS Opening

```
# cryptsetup luksOpen /dev/vdb1 Encrypted_Disk
Enter passphrase for /dev/vdb1: meetup2018
```

Step 5 - Creating a Filesystem on an Un-Encrypted (*Open*) LUKS Partition

Example 5. Creating a Filesystem

Listing 6. Creating XFS Filesystem on LUKS Partition

```
# mkfs.xfs /dev/mapper/Encrypted_Disk
meta-data=/dev/mapper/Encrypted_Disk isize=512    agcount=4, agsize=65344 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=0, sparse=0
data     =                       bsize=4096   blocks=261376, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=1
log      =internal log           bsize=4096   blocks=855, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

Step 6 - Creating a Mount Point

Example 6. Creating a Mountpoint

Listing 7. Creating a Filesystem Mount Point

```
# mkdir /EncryptedDrive
```

Step 7 - Mounting the Partition

Example 7. Mounting Filesystem and Creating a Test File

Listing 8. Creating a Filesystem Mount Point

```
# mount /dev/mapper/Encrypted_Disk /EncryptedDrive
```

Listing 9. Creating a Test File

```
# echo "This is a test on encrypted filesystem" > /EncryptedDrive/TestFile.txt
```

Listing 10. Verifying Test File Contents

```
# cat /EncryptedDrive/TestFile.txt
This is a test on encrypted filesystem
```

Step 8 - Unmounting LUKS Partition

Example 8. Unmount the Filesystem

Listing 11. Unmounting the Filesystem

```
# umount /EncryptedDrive
```

Step 9 - Encrypting (*Close*) the LUKS Partition

Example 9. Closing the LUKS Device

Listing 12. **cryptsetup** *to Close Partition*

```
# cryptsetup luksClose Encrypted_Disk
```

Obtaining LUKS Information

```
# cryptsetup luksDump /dev/vdb1
LUKS header information for /dev/vdb1

Version:        1
Cipher name:    aes
Cipher mode:    xts-plain64
Hash spec:      sha256
Payload offset: 4096
MK bits:        256
MK digest:      5d f5 d6 77 40 25 6e d8 b8 d7 b8 34 9a d7 37 48 c4 4f a0 41
MK salt:        7e ac e3 68 3c 52 12 f5 ca cd 3d ef 42 96 cd 7e
                61 52 17 c0 2e 4a ba 46 5d 4d ed c7 0a 1e 1e 72
MK iterations:  75250
UUID:           21e34e9f-0142-44ad-a628-b9934d8c2225

Key Slot 0: ENABLED
    Iterations:             598830
    Salt:                   ba 06 6b 66 d9 28 33 75 65 33 08 61 91 bc f4 a1
                            05 30 20 44 0c d1 2c d0 53 79 a7 24 cf 98 5d cf
    Key material offset:    8
    AF stripes:             4000
Key Slot 1: DISABLED
Key Slot 2: DISABLED
Key Slot 3: DISABLED
Key Slot 4: DISABLED
Key Slot 5: DISABLED
Key Slot 6: DISABLED
Key Slot 7: DISABLED
```

Backup of LUKS Header

It is a good idea to backup LUKS in the event the encrypted volume would need to be repaired.

Listing 13. Backup of LUKS Header

```
# cryptsetup luksHeaderBackup /path/to/encrypted/device --header-backup-file
/path/to/backup/file
```

Listing 14. Testing a Backup of LUKS Header

```
# cryptsetup luksOpen /path/to/encrypted/device --header /path/to/backup/file
```

Listing 15. Restoring a LUKS Header Backup

```
# cryptsetup luksHeaderRestore path/to/encrypted/device --header-backupfile
/path/to/backup/file
```

2.2. Using Clevis and Tang to have Network Bound Disk Encryption (NBDE)

Clevis and Tang are two complimentary services that are provided to allow Network Bound Disk Encryption (NBDE). Clevis is considered the **client** while Tang is considered the **server**.

Terms

Clevis: Clevis is a plugable framework for automated decryption. It can be used to provide automated decryption of data or even automated unlocking of LUKS volumes.

Tang: Server side service that Clevis connects to in order to receive a decryption key and allow the NBDE service connection.

A really good reference from Red Hat is available here: https://rhelblog.redhat.com/2018/04/13/an-easier-way-to-manage-disk-decryption-at-boot-with-red-hat-enterprise-linux-7-5-using-nbde/#more-4351

The diagram below depicts the Clevis and Tang framework on a high-level. As seen the cryptographic interface used between the two applications is the JOSE Crypto Library.

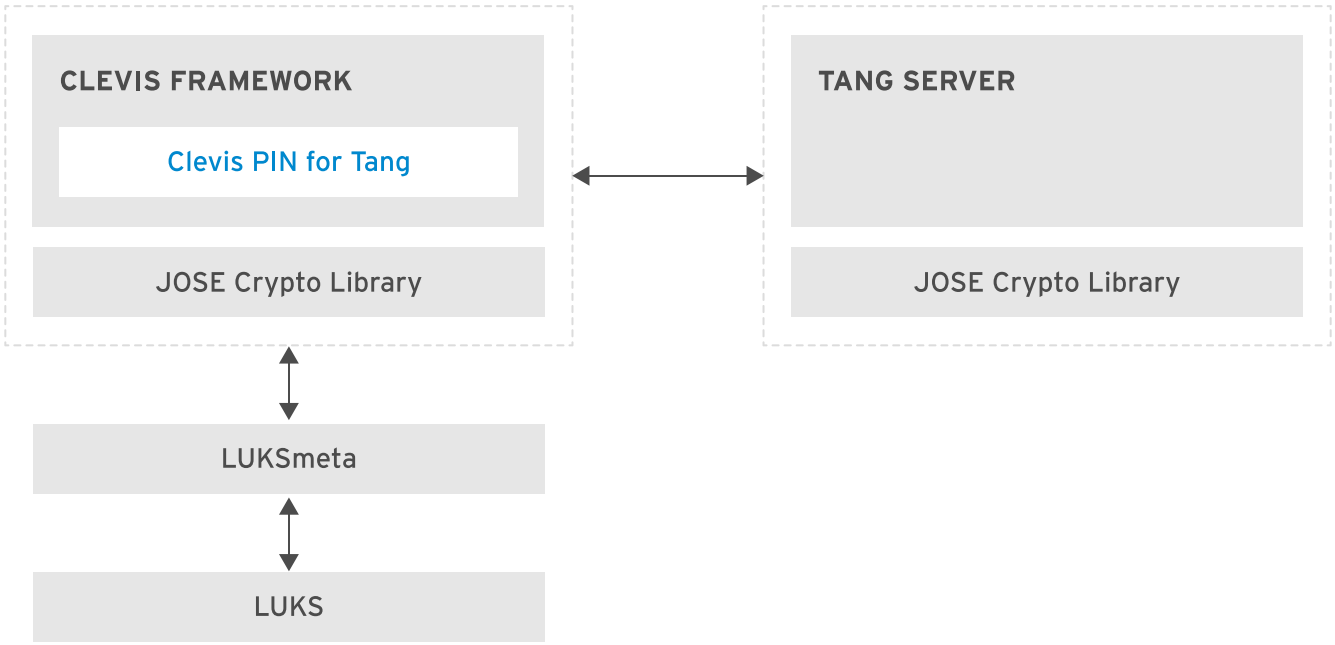

Figure 4. NBDE Decryption Framework

As part of the second portion of disk encryption labs, we will be setting up three (3) Tang servers and configuring the Clevis client to decrypt the volume created in the first portion of the encryption lab. We will be using Shamir's Secret Sharing (SSS) with Clevis to have a threshold of two (2) across the three (3) Tang servers.

The **threshold** is how many Tang servers must supply the LUKS key before the disk can be decrypted.

2.2.1. Setting up a Tang Sever

The Tang packages and services must be setup and configured. As part of the demonstration and hands-on lab, you will be installing the Tang packages and configuring servers for the service along with the Linux firewall service. In the hands-on lab below, you will be configuring three (3) Tang servers so that Clevis can be setup to require responses from any two (2) Tang servers.

Servers to Configure

- serverb
- serverc
- serverd

Packages to Install

- tang

Step 1 - Install Tang Packages on Each Server

Example 10. Install Tang on the Keyservers

```
#  yum install tang
```

Step 2 - Configure Tang Service on Each Server

Example 11. Configure the Tang Service

```
systemctl enable tangd.socket --now
```

Step 3 - Configure Tang Firewall Ports on Each Server

Example 12. Configure Firewalld on the Tang Servers

```
# firewall-cmd --zone=public --add-port=80/tcp \
--permanent

# firewall-cmd --reload
```

Before proceeding, please repeat steps 1-3 on the remaining servers (**serverc** and **serverd**).

2.2.2. Setting up Clevis Client

The Clevis client packages must be installed and setup on the server containing the encrypted LUKS device. This service allows the server to connect to the Tang server for obtaining the network-based decryption keys.

Servers to Configure

- servera

Packages to Install

- clevis
- clevis-luks
- clevis-dracut

Step 1 - Install the Packages

Example 13. Install Clevis packages on the Server

```
#  yum install clevis clevis-luks clevis-dracut
```

Step 2 - Create the SSS Config for LUKS Binding

For this step, we are wanting to use a TANG configuration of 3 servers noting that at least two (2) of the TANG servers need to be contacted. You could just as easily have a threshold of 1 or 3.

The Clevis client

Use the **# man clevis** to see the usage and syntax information for Clevis.

Listing 16. Excerpts from man page

```
For example, let's create a high-availability setup using Tang:

   cfg='{"t":1,"pins":{"tang":[{"url":...},{"url":...}]}}'

... output omitted ...

Clevis can be used to bind an existing LUKS volume  to  its  automation policy.  This is accomplished with a simple
command:

   $ clevis luks bind -d /dev/sda tang '{"url":...}'
```

Other Clevis Client References

man clevis-encrypt-sss

man clevis-luks-bind

The threshold specifies how many TANG servers must provide a key to decrypt a given LUKS volume. If one (1) is specified as the threshold value, then as long as a single TANG server can be reached and provides a valid key, the volume can be decrypted.

Example 14. Associate the LUKS partition and Configure Clevis to use Tang Servers

Listing 17. Configuring SSS Encryption

```
#  cfg=$'{"t":2,"pins":{"tang":[\
{"url":"http://serverb.lab.example.com"},\
{"url":"http://serverc.lab.example.com"},\
{"url":"http://serverd.lab.example.com"}]}}'
```

Step 3 - Use the SSS Config for Clevis LUKS Binding

Example 15. Clevis to Bind a LUKS Volume

Listing 18. Binding to LUKS with SSS Encryption to Tang Servers

```
[root@servera ~]#  clevis luks bind -d /dev/vdb1 sss "$cfg"
The advertisement contains the following signing keys:

9B65OW9ly08QrYjGZ5CgMv2H7XM

Do you wish to trust these keys? [ynYN] y
The advertisement contains the following signing keys:

-IGeEyKKphOqu3-1DPtP7CN_ESw

Do you wish to trust these keys? [ynYN] y
The advertisement contains the following signing keys:

LDGB1B4_Va1skLi7npgv-uQPryg

Do you wish to trust these keys? [ynYN] y
You are about to initialize a LUKS device for metadata storage.
Attempting to initialize it may result in data loss if data was
already written into the LUKS header gap in a different format.
A backup is advised before initialization is performed.

Do you wish to initialize /dev/vdb1? [yn] y
Enter existing LUKS password:

[root@servera ~]#
```

Step 4 - Enable Clevis Client Services

Example 16. Enabling Clevis Services

Listing 19. Enabling Clevis Service

```
[root@servera ~]# systemctl enable clevis-luks-askpass.path
Created symlink from /etc/systemd/system/remote-fs.target.wants/clevis-luks-askpass.path to /usr/lib/systemd/system/clevis-luks-
askpass.path.
[root@servera ~]#
```

Verifying TANG Keys Added to LUKS

It is possible to verify that new keys have been added to the keyslots in LUKS for the Clevis/Tang setup by using **crypsetup luksDump <volume>**.

Listing 20. Verifying New Key Exists

```
[root@servera ~]#  cryptsetup luksDump /dev/vdb1
LUKS header information for /dev/vdb1

Version:        1
Cipher name:    aes
Cipher mode:    xts-plain64
Hash spec:      sha256
Payload offset: 4096
MK bits:        256
MK digest:      12 5b 60 00 1a 90 d6 69 10 75 4d df a5 e8 f9 85 bd 22 72 1a
MK salt:        1e fd 23 69 3f a3 6d 45 82 ba 93 bc 37 dd db 33
                a3 93 63 43 29 49 60 a7 ab 10 8c fd 29 a3 4b f2
MK iterations:  73500
UUID:           c2ef1b5a-7c1d-4a22-be9d-4728fe9a9975

Key Slot 0: ENABLED
    Iterations:             618357
    Salt:                   62 cb 43 7c 59 20 2f 9e 62 7f e9 a7 e7 9f ff 3c
                            38 b9 92 35 b6 2e 88 74 84 6c a8 2b ec f7 28 b5
    Key material offset:    8
    AF stripes:             4000
Key Slot 1: ENABLED
    Iterations:             593277
    Salt:                   c0 6c 8b 70 85 c5 2e 4d c6 12 c6 cc e9 a1 46 5a
                            39 43 94 fb 9c c1 cf b5 d1 f6 4d ec 67 1d ad 40
    Key material offset:    264
    AF stripes:             4000
Key Slot 2: DISABLED
Key Slot 3: DISABLED
Key Slot 4: DISABLED
Key Slot 5: DISABLED
Key Slot 6: DISABLED
Key Slot 7: DISABLED
[root@servera ~]#
```

2.3. Using LUKS, Clevis, and Tang to have NBDE (*Putting it all Together*)

Once all the preliminary setup is done, you can setup the **/etc/crypttab** on the client machine as well as the **/etc/fstab** file to mount the filesystem upon boot using the Clevis client and the Tang server. This is the main purpose of setting up NBDE as it prevents the system from prompting for a password providing that the proper network requirements are met.

Servers to Configure

- servera

Step 1 - Create the Crypttab File

The **/etc/crypttab** file describes encrypted block devices that are set up during system boot.

*Example 17. Creating **/etc/crypttab***

Listing 21. Creating a Crypttab for LUKS Volumes

```
# vi /etc/crypttab
Encrypted_Disk /dev/vdb1   none  _netdev
```

Step 2 - Modify the fstab File

*Example 18. Modifying **/etc/fstab***

*Listing 22. Modifying **fstab** for LUKS Volumes*

```
# vi /etc/fstab
...output omitted...

/dev/mapper/Encrypted_Disk    /EncryptedDrive xfs  _netdev   1 2
```

Step 3 - Reboot the System

Example 19. Rebooting the System

Listing 23. Rebooting

```
# reboot
```

Step 4 - Verify Server Rebooted and Drive is Mapped with Data

Example 20. System Verification

Listing 24. Verifying System

```
# ssh root@servera
System is booting up. See pam_nologin(8)
Last login: Thu Sep  6 19:40:06 2018 from 172.25.250.250

[root@servera ~]# df
Filesystem                1K-blocks    Used Available Use% Mounted on
/dev/vda1                  10474476 1744372   8730104  17% /
devtmpfs                     486060       0    486060   0% /dev
tmpfs                        507768       0    507768   0% /dev/shm
tmpfs                        507768   13188    494580   3% /run
tmpfs                        507768       0    507768   0% /sys/fs/cgroup
/dev/mapper/Encrypted_Disk  1042084   32948   1009136   4% /EncryptedDrive
tmpfs                        101556       0    101556   0% /run/user/0

[root@servera ~]# cat /EncryptedDrive/TestFile.txt
This is a test on encrypted filesystem
[root@servera ~]#
```

3. 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.

3.1. 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 21. Connecting to the Workstation VM

Listing 25. 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 22. Installing SCAP Workbench

Listing 26. 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 23. Launching SCAP Workbench

Listing 27. Using SCAP Workbench

```
# scap-workbench
```

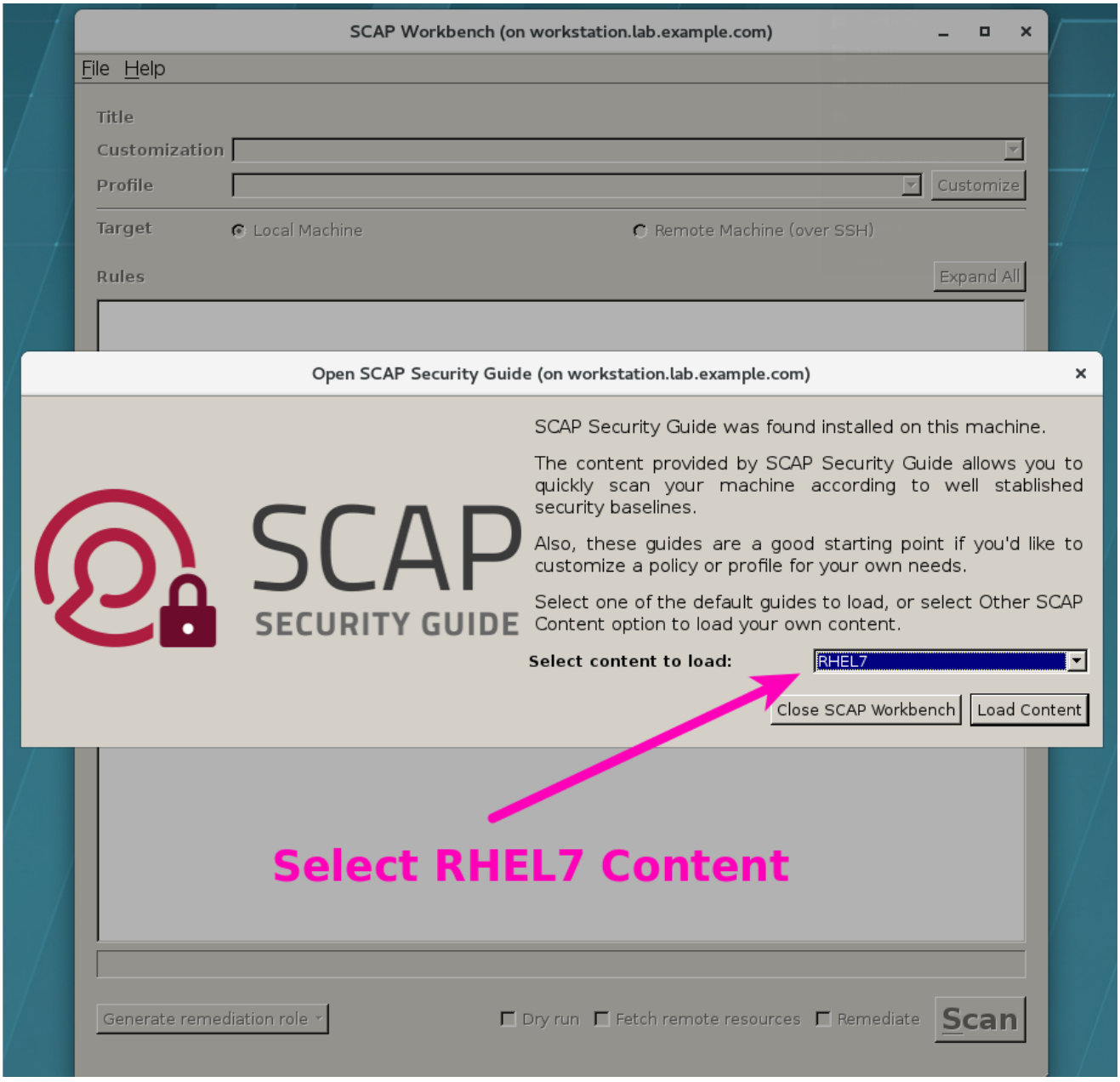

Figure 5. SCAP Workbench Startup

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 24. Creating Custom Content

1. For **Select content to load:** select "**RHEL7**", then click "**Load Content**"
2. Select the **Profile** you want to use to start customization

For this example, we will use the **USGCB/STIG** Baseline

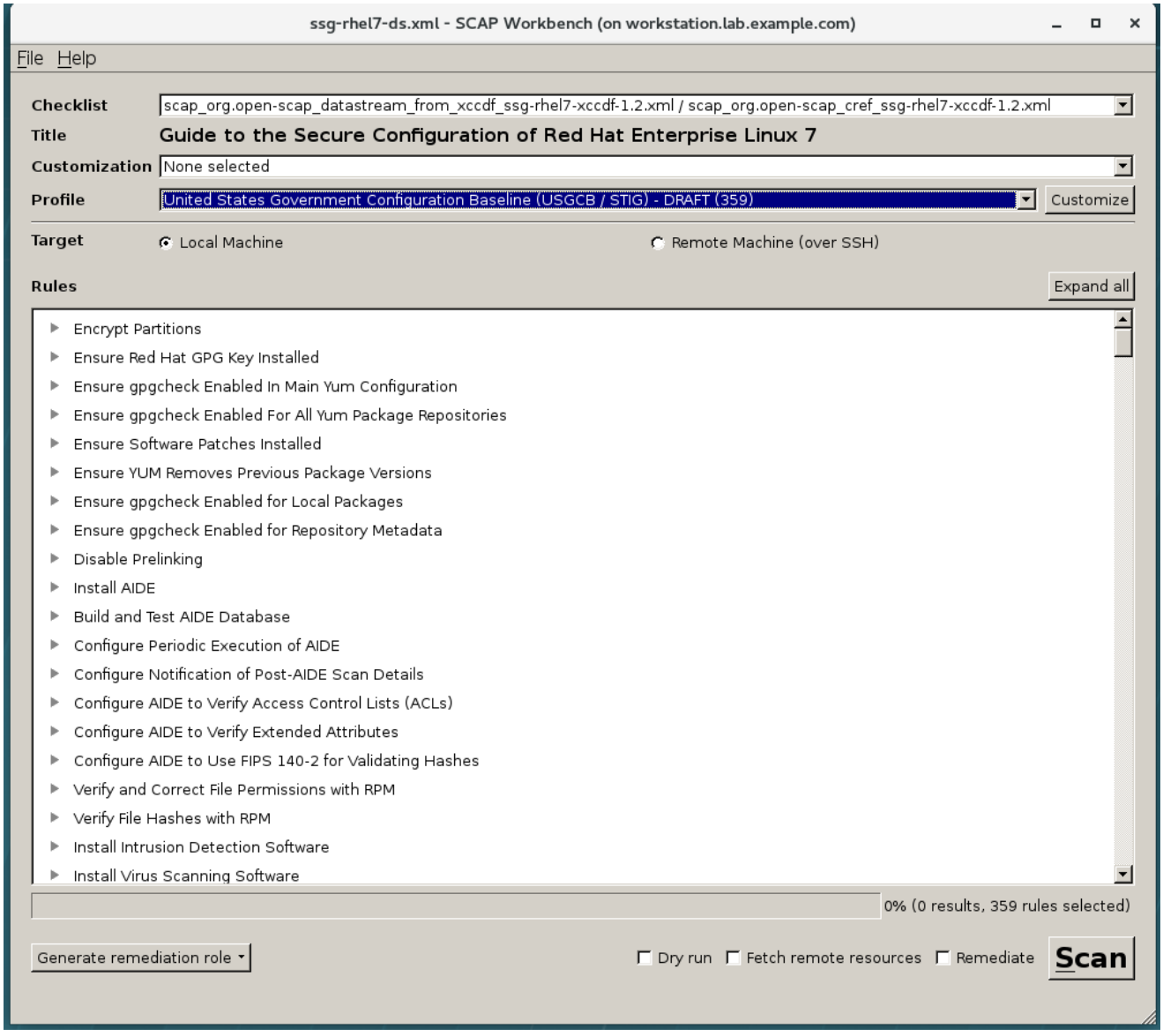

Figure 6. SCAP Workbench USGCB/STIG Profile

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

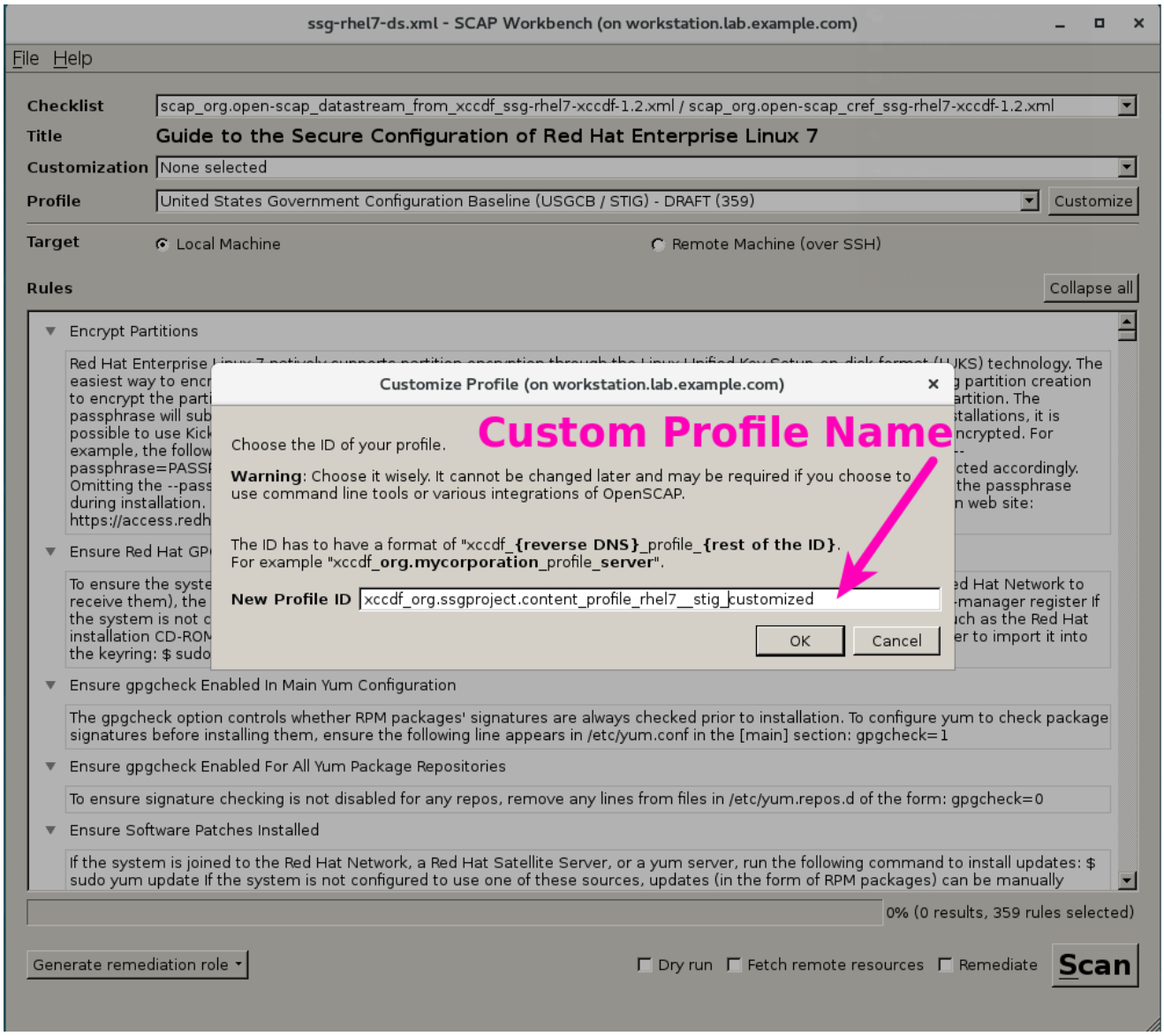

Figure 7. SCAP Custom STIG Profile Creation

The name for this is: **xccdf_org.ssgproject.content_profile_rhel7__stig_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.

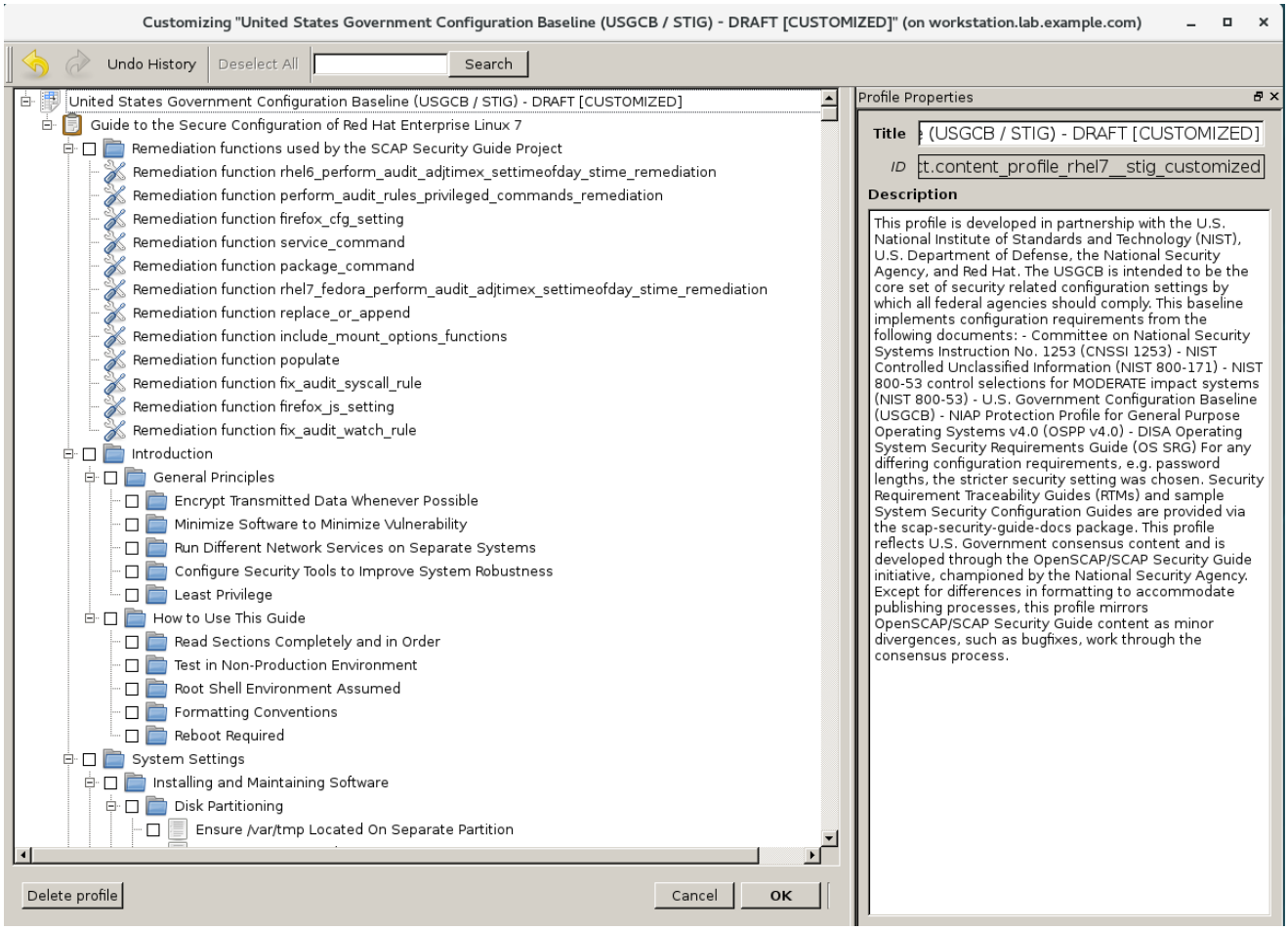

Figure 8. SCAP Custom Profile Selections

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**

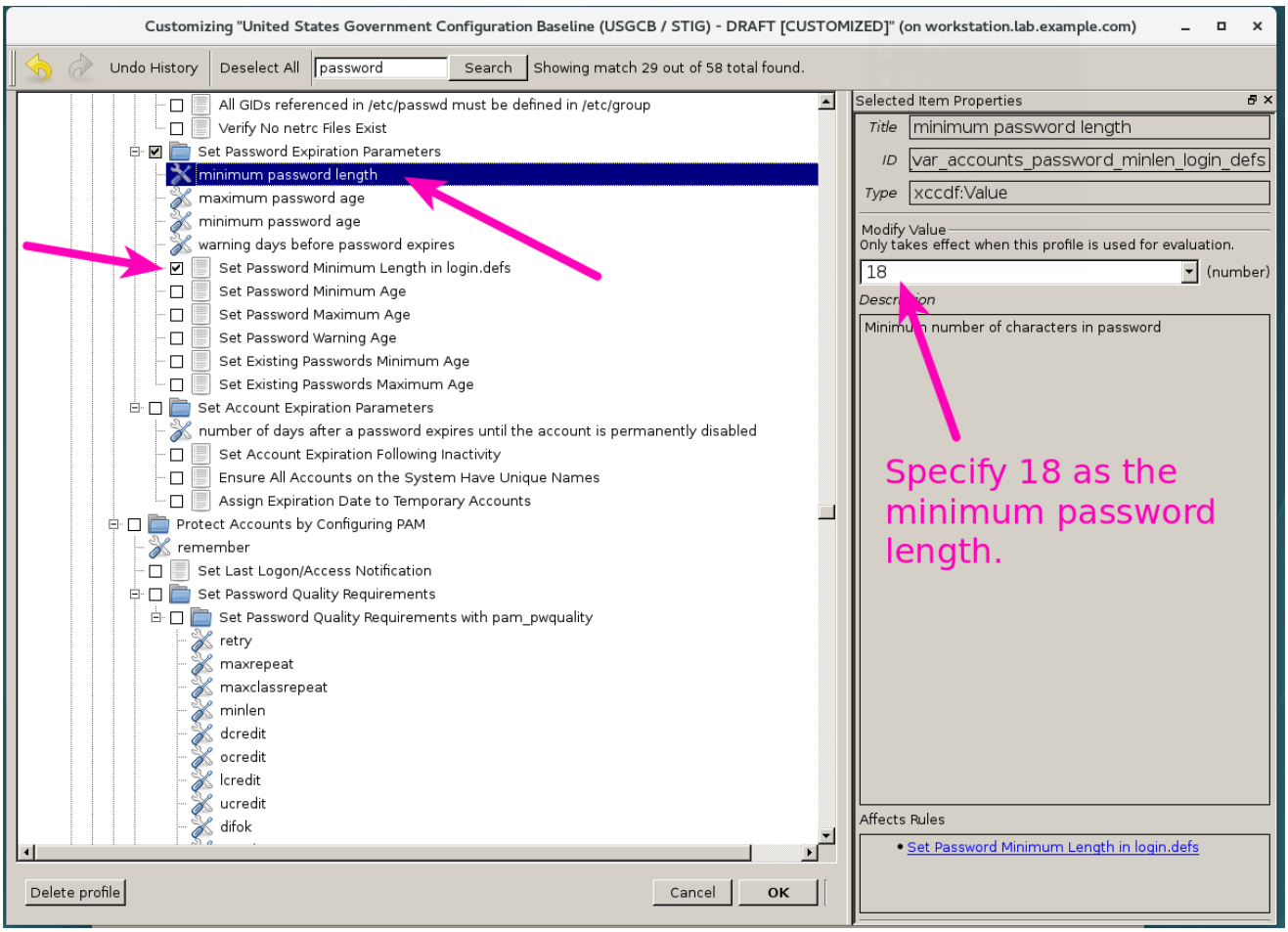

Figure 9. SCAP Custom Profile Password Settings for Login.Defs

6. Set password quality reqirements 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"

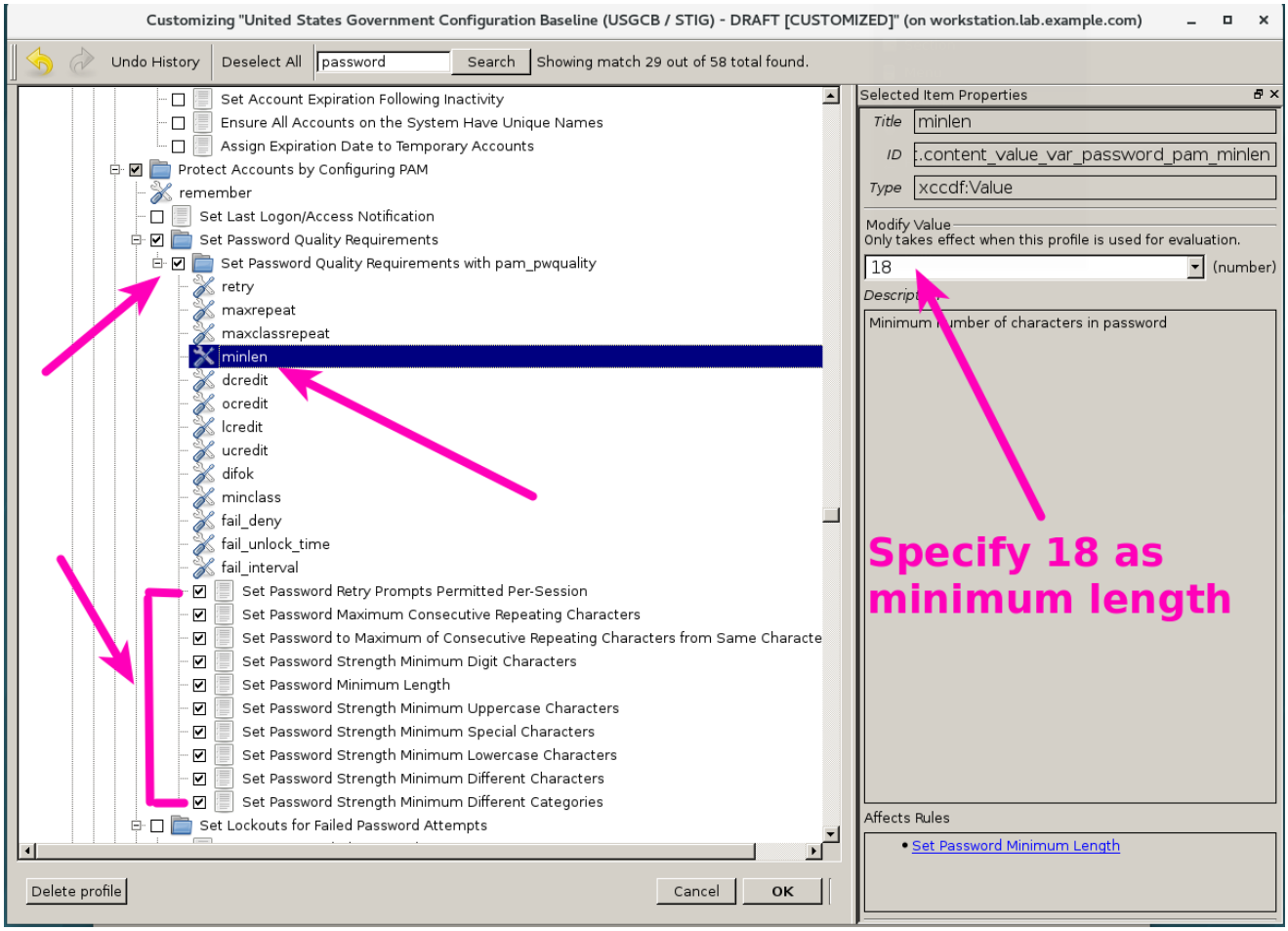

Figure 10. SCAP Custom Profile PAM Quality Requirements

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

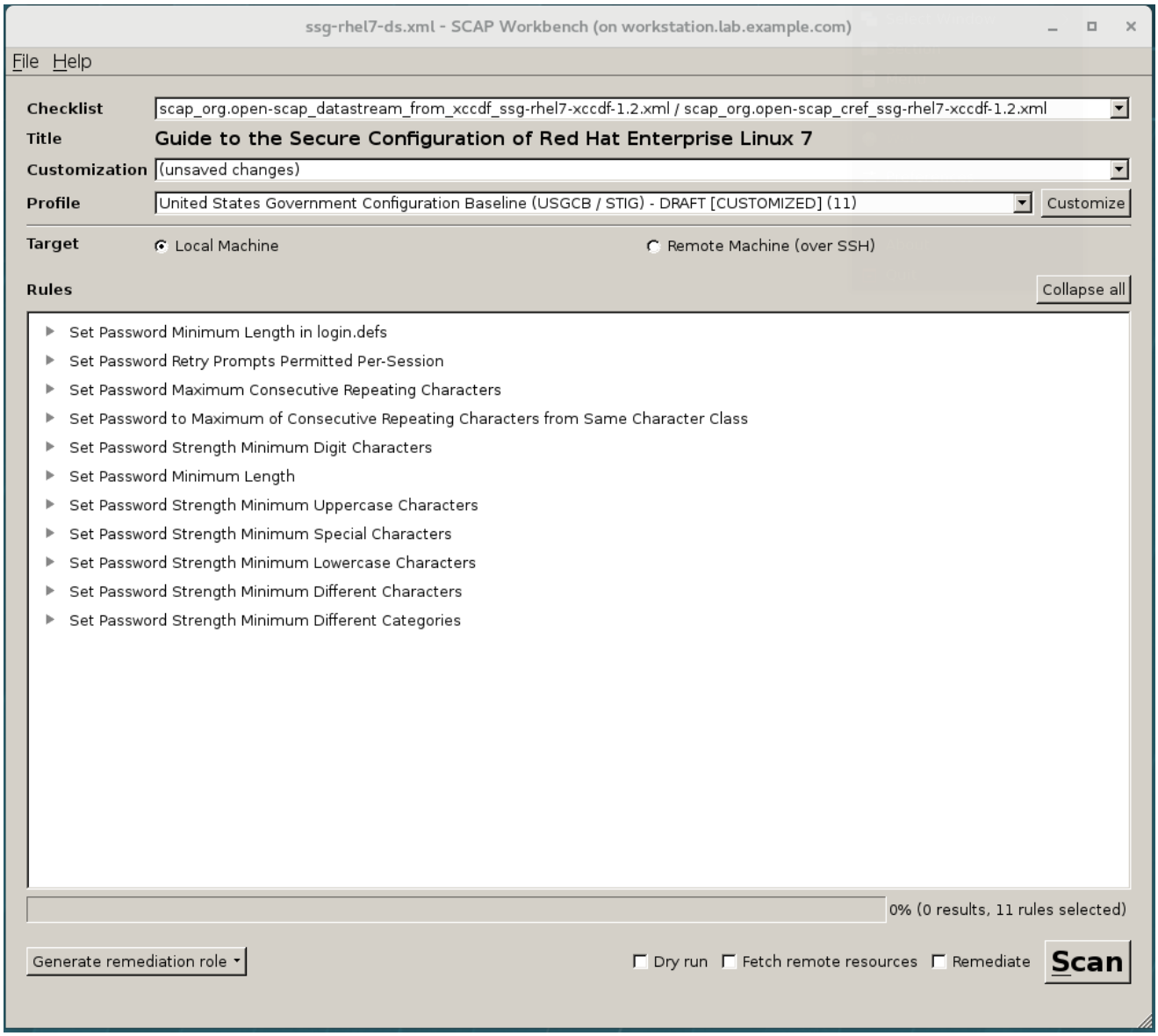

Figure 11. SCAP Custom Profile Selected Settings View

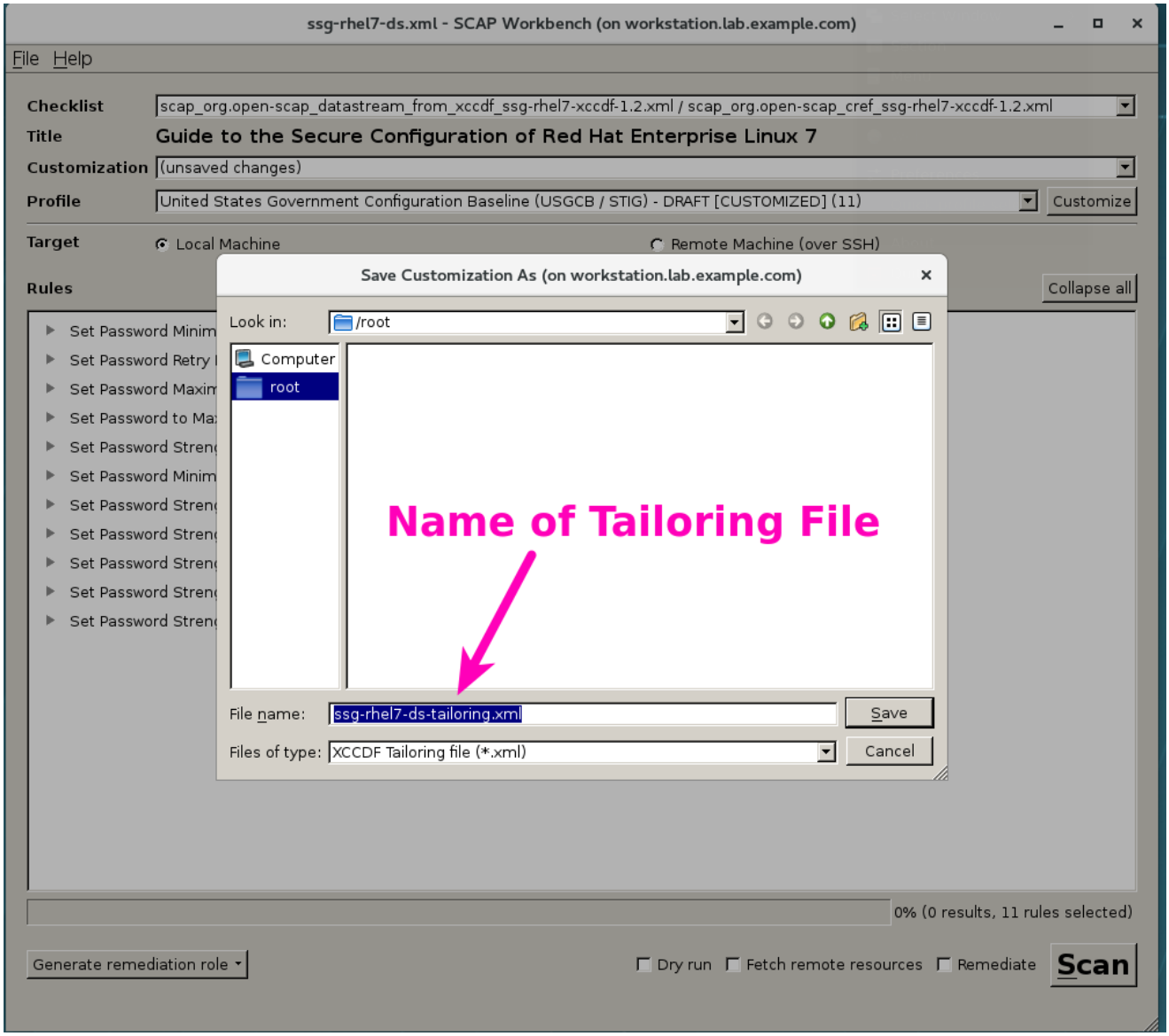

Figure 12. SCAP Custom Profile Creation Saving

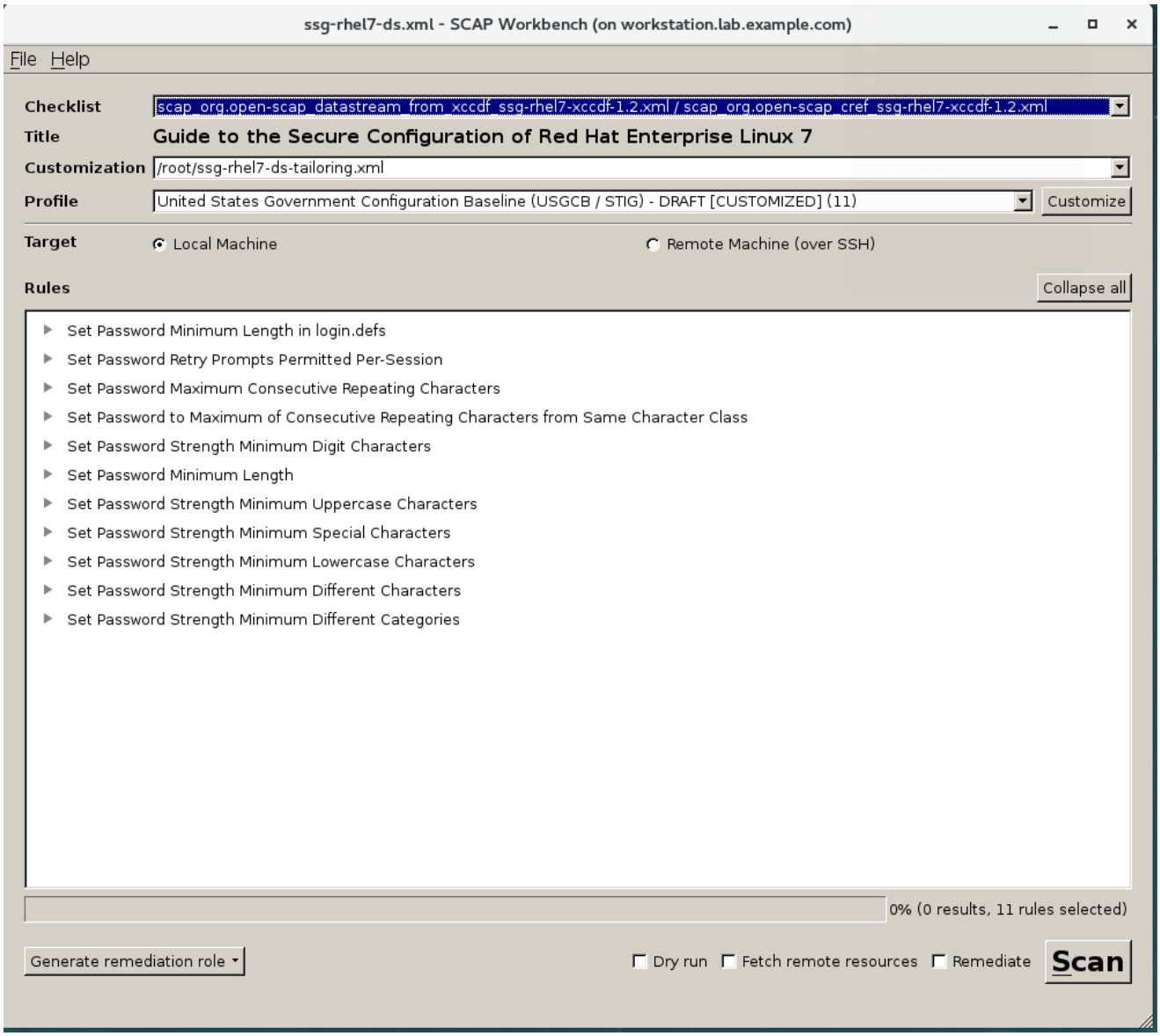

Figure 13. SCAP Custom Profile Final View

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

Listing 28. Copy custom content

```
[root@workstation ~]# scp ssg-rhel7-ds-tailoring.xml root@serverc:
Warning: Permanently added 'serverc,172.25.250.12' (ECDSA) to the list of known hosts.
ssg-rhel7-ds-tailoring.xml                    100%   51KB  14.9MB/s   00:00
[root@workstation ~]#
```

3.2. Running a SCAP Scan with Custom Content

Servers to Configure

- serverc

Packages to Install

- openscap-scanner
- scap-security-guide

Step 1 - SSH to serverc

The fist step is to connect to the server.

Example 25. Connecting to the **serverc** *VM*

Listing 29. Connecting to **serverc** *Using SSH*

```
# ssh root@serverc
```

Step 2 - Install packages on serverc

The second step is to install software on the server.

Example 26. Install software on **serverc**

Listing 30. Installing Software on **serverc**

```
# yum install scap-security-guide
Loaded plugins: langpacks, search-disabled-repos
rhel--server-dvd                                         | 4.3 kB     00:00
(1/2): rhel--server-dvd/group_gz                           | 145 kB   00:00
(2/2): rhel--server-dvd/primary_db                         | 4.1 MB   00:00
Resolving Dependencies
--> Running transaction check
---> Package scap-security-guide.noarch 0:0.1.36-7.el7 will be installed
--> Processing Dependency: openscap-scanner >= 1.2.5 for package: scap-security-guide-0.1.36-7.el7.noarch
--> Processing Dependency: xml-common for package: scap-security-guide-0.1.36-7.el7.noarch
--> Running transaction check
---> Package openscap-scanner.x86_64 0:1.2.16-6.el7 will be installed
--> Processing Dependency: openscap(x86-64) = 1.2.16-6.el7 for package: openscap-scanner-1.2.16-6.el7.x86_64
--> Processing Dependency: libopenscap.so.8()(64bit) for package: openscap-scanner-1.2.16-6.el7.x86_64
---> Package xml-common.noarch 0:0.6.3-39.el7 will be installed
--> Running transaction check
---> Package openscap.x86_64 0:1.2.16-6.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                Arch       Version          Repository            Size
================================================================================
Installing:
 scap-security-guide    noarch     0.1.36-7.el7     rhel--server-dvd     2.6 M
Installing for dependencies:
 openscap               x86_64     1.2.16-6.el7     rhel--server-dvd     3.8 M
 openscap-scanner       x86_64     1.2.16-6.el7     rhel--server-dvd      61 k
 xml-common             noarch     0.6.3-39.el7     rhel--server-dvd      26 k

Transaction Summary
================================================================================
Install  1 Package (+3 Dependent packages)

Total download size: 6.5 M
Installed size: 122 M
Is this ok [y/d/N]: y

... output omitted ...

Installed:
  scap-security-guide.noarch 0:0.1.36-7.el7

Dependency Installed:
  openscap.x86_64 0:1.2.16-6.el7      openscap-scanner.x86_64 0:1.2.16-6.el7
  xml-common.noarch 0:0.6.3-39.el7
```

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 31. 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 32. 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 27. Scanning **serverc**

Listing 33. Using **oscap** *and the tailoring profile to scan* **serverc**

```
# [root@serverc ~]# oscap xccdf eval \
--profile xccdf_org.ssgproject.content_profile_rhel7__stig_customized \
--tailoring-file ssg-rhel7-ds-tailoring.xml \
--results custom_scan_results.xml \
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml

WARNING: This content points out to the remote resources. Use `--fetch-remote-resources' option to download them.

WARNING: Skipping https://www.redhat.com/security/data/oval/com.redhat.rhsa-RHEL7.xml.bz2 file which is referenced from XCCDF content

Title   Set Password Minimum Length in login.defs
Rule    xccdf_org.ssgproject.content_rule_accounts_password_minlen_login_defs
Ident   CCE-27123-9
Result  fail

Title   Set Password Retry Prompts Permitted Per-Session
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_retry
Ident   CCE-27160-1
Result  pass

Title   Set Password Maximum Consecutive Repeating Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_maxrepeat
Ident   CCE-27333-4
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
Ident   CCE-27512-3
Result  fail

Title   Set Password Strength Minimum Digit Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_dcredit
Ident   CCE-27214-6
Result  fail

Title   Set Password Minimum Length
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_minlen
Ident   CCE-27293-0
Result  fail

Title   Set Password Strength Minimum Uppercase Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_ucredit
Ident   CCE-27200-5
Result  fail

Title   Set Password Strength Minimum Special Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_ocredit
Ident   CCE-27360-7
Result  fail

Title   Set Password Strength Minimum Lowercase Characters
```

```
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_lcredit
Ident   CCE-27345-8
Result  fail

Title   Set Password Strength Minimum Different Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_difok
Ident   CCE-26631-2
Result  fail

Title   Set Password Strength Minimum Different Categories
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_minclass
Ident   CCE-27115-5
Result  fail

[root@serverc ~]#
```

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@serverc ~]# grep "Profile id" ssg-rhel7-ds-tailoring.xml
  <xccdf:Profile id="xccdf_org.ssgproject.content_profile_rhel7__stig_customized" extends
="xccdf_org.ssgproject.content_profile_ospp-rhel7">
```

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 28. Creating a SCAP Report from a Results File

Listing 34. Generating a Report

```
[root@serverc ~]# 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 35. Creating a Results File and Report During Custom Content Scan

```
[root@serverc ~]# oscap xccdf eval \
```

```
--profile xccdf_org.ssgproject.content_profile_rhel7__stig_customized \
--tailoring-file ssg-rhel7-ds-tailoring.xml \
--results custom_scan_results_2.xml \
--report Custom_Scan_Report_2.html \
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml

WARNING: This content points out to the remote resources. Use '--fetch-remote-resources' option to download them.

WARNING: Skipping https://www.redhat.com/security/data/oval/com.redhat.rhsa-RHEL7.xml.bz2 file which is referenced from
XCCDF content

Title   Set Password Minimum Length in login.defs
Rule    xccdf_org.ssgproject.content_rule_accounts_password_minlen_login_defs
Ident   CCE-27123-9
Result  fail

Title   Set Password Retry Prompts Permitted Per-Session
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_retry
Ident   CCE-27160-1
Result  pass

Title   Set Password Maximum Consecutive Repeating Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_maxrepeat
Ident   CCE-27333-4
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
Ident   CCE-27512-3
Result  fail

Title   Set Password Strength Minimum Digit Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_dcredit
Ident   CCE-27214-6
Result  fail

Title   Set Password Minimum Length
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_minlen
Ident   CCE-27293-0
Result  fail

Title   Set Password Strength Minimum Uppercase Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_ucredit
Ident   CCE-27200-5
Result  fail

Title   Set Password Strength Minimum Special Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_ocredit
Ident   CCE-27360-7
Result  fail

Title   Set Password Strength Minimum Lowercase Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_lcredit
Ident   CCE-27345-8
Result  fail

Title   Set Password Strength Minimum Different Characters
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_difok
Ident   CCE-26631-2
Result  fail

Title   Set Password Strength Minimum Different Categories
Rule    xccdf_org.ssgproject.content_rule_accounts_password_pam_minclass
Ident   CCE-27115-5
Result  fail
```

```
[root@serverc ~]#
```

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 29. Transferring Results

Listing 36. Transferring the Results and Report Files

```
[root@serverc ~]# scp *.xml *.html root@workstation:
The authenticity of host 'workstation (172.25.250.254)' can't be established.
ECDSA key fingerprint is SHA256:GCpIQxItJSWgZDzlmpnZINbwsjf9axrs+o6170OyOuk.
ECDSA key fingerprint is MD5:2b:98:e1:85:8b:c7:ea:31:72:08:4d:39:15:ec:5d:da.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'workstation,172.25.250.254' (ECDSA) to the list of known hosts.
root@workstation's password:
custom_scan_results_2.xml                                     100% 4307KB  62.5MB/s   00:00
custom_scan_results.xml                                       100% 4307KB  56.7MB/s   00:00
ssg-rhel7-ds-tailoring.xml                                    100%   51KB  23.6MB/s   00:00
Custom2_Report.html                                           100%    0     0.0KB/s   00:00
Custom_Scan_Report_2.html                                     100%  331KB  29.1MB/s   00:00
Custom_Scan_Report.html                                       100%  331KB  35.8MB/s   00:00
[root@serverc ~]#
```

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 30. Viewing the SCAP Report

Listing 37. Opening the SCAP HTML Report with Firefox

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

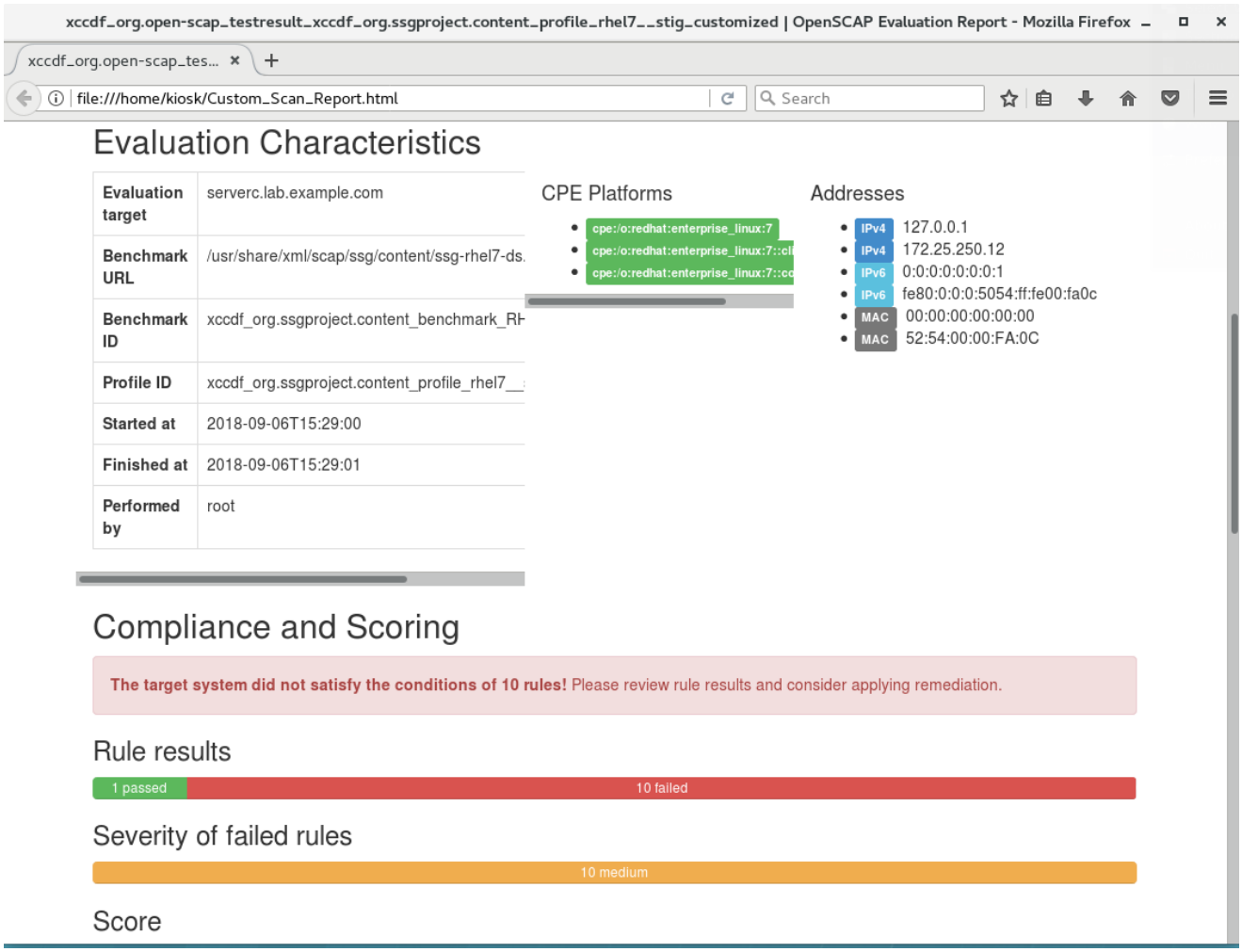

Figure 14. SCAP Scan Results Report in Firefox

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.

3.3. 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

- serverc

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 31. Generating Ansible Playbook

Listing 38. Ansible Playbook Generation

```
[root@workstation ~]# oscap xccdf generate fix \
--profile xccdf_org.ssgproject.content_profile_rhel7__stig_customized \
--tailoring-file ssg-rhel7-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_rhel7__stig_customized
# XCCDF Version:  unknown
#
# Evaluation Start Time:  2018-09-06T15:42:54
# Evaluation End Time:  2018-09-06T15:42:54
#
# This file was generated by OpenSCAP 1.2.16 using:
#   $ oscap xccdf generate fix --result-id xccdf_org.open-
scap_testresult_xccdf_org.ssgproject.content_profile_rhel7__stig_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 "192.168.1.155," playbook.yml
# $ ansible-playbook -i inventory.ini playbook.yml
#
#################################################################################
```

```
- hosts: all
  vars:
     var_accounts_password_minlen_login_defs: 18
     var_password_pam_maxrepeat: 2
     var_password_pam_maxclassrepeat: 4
     var_password_pam_dcredit: -1
     var_password_pam_minlen: 18
     var_password_pam_ucredit: -1
     var_password_pam_ocredit: -1
     var_password_pam_lcredit: -1
     var_password_pam_difok: 8
     var_password_pam_minclass: 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-27123-9
       - 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 minclass is set accordingly
      lineinfile:
        create=yes
        dest="/etc/security/pwquality.conf"
        regexp="^minclass"
        line="minclass = {{ var_password_pam_minclass }}"
      tags:
        - accounts_password_pam_minclass
        - medium_severity
        - CCE-27115-5
        - NIST-800-53-IA-5
        - DISA-STIG-RHEL-07-010170
```

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 39. 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 32. Downloading Ansible Files

Listing 40. Downloading Ansible Files

```
[root@workstation ~]# wget http://content/meetup/inventory
--2018-09-06 17:49:04--  http://content/meetup/inventory
Resolving content (content)... 172.25.254.254
Connecting to content (content)|172.25.254.254|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 24
Saving to: 'inventory'

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

2018-09-06 17:49:04 (2.74 MB/s) - 'inventory' saved [24/24]

[root@workstation ~]# wget http://content/meetup/ansible.cfg
--2018-09-06 17:49:19--  http://content/meetup/ansible.cfg
Resolving content (content)... 172.25.254.254
Connecting to content (content)|172.25.254.254|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 159
Saving to: 'ansible.cfg'

100%[======================================>] 159         --.-K/s   in 0s

2018-09-06 17:49:19 (12.0 MB/s) - 'ansible.cfg' saved [159/159]

[root@workstation ~]#
```

Reviewing Ansible Configurations

The **inventory** file provided only has a single host **serverc** 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
serverc.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 **serverc** system.

*Example 33. Remediation of **serverc** with Ansible Playbook*

Listing 41. Running the Ansible Playbook

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

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

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

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

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

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

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

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

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

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

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

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

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

PLAY RECAP *********************************************************************
serverc.lab.example.com    : ok=11   changed=10   unreachable=0    failed=0

[root@workstation ~]#
```

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 34. Scanning System after Fixes and Verifying Results

Listing 42. Performing SCAP Verification Scan

```
[root@serverc ~]# oscap xccdf eval \
--profile xccdf_org.ssgproject.content_profile_rhel7__stig_customized \
--tailoring-file ssg-rhel7-ds-tailoring.xml \
--results custom_scan_results_fixed.xml \
--report Custom_Scan_Report_Fixed.html \
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

Listing 43. Copying Results to Workstation

```
[root@serverc ~]# scp custom_scan_results_fixed.xml Custom_Scan_Report_Fixed.html workstation:
root@workstation's password:
custom_scan_results_fixed.xml                                 100% 4330KB  70.2MB/s   00:00
Custom_Scan_Report_Fixed.html                                 100%  291KB  33.1MB/s   00:00
[root@serverc ~]#
```

Listing 44. Viewing Results on Workstation

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

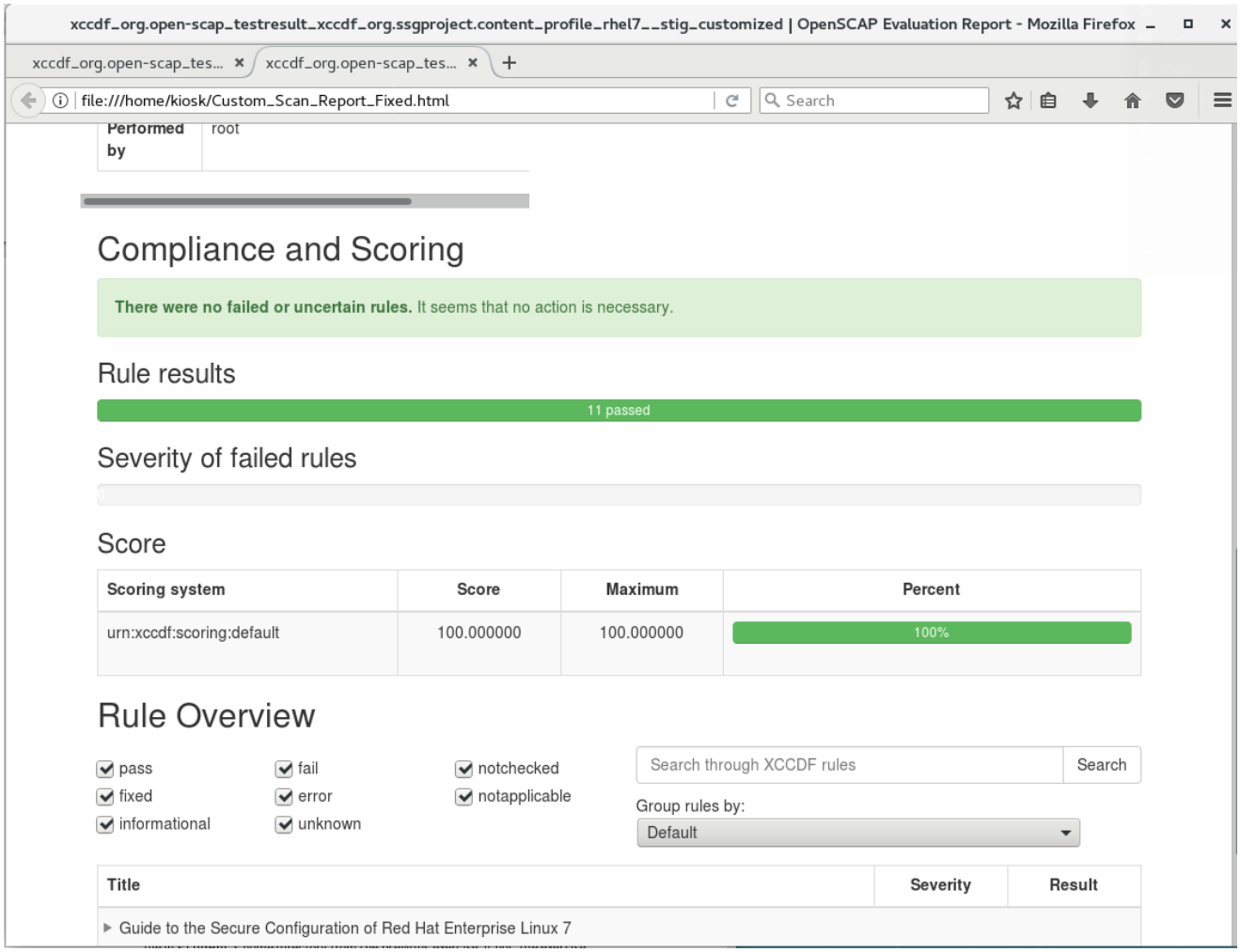

Figure 15. Fixed SCAP Scan Results Report in Firefox