



RED HAT ENTERPRISE LINUX 7 STIG ANSIBLE DOCUMENTATION

Version 3, Release 1

October 2020

Developed by DISA for the DoD

TABLE OF CONTENTS

Page

1. BACKGROUND 1
2. INSTALLATION 2
2.1 Installing Ansible 2
2.2 Extracting 2
3. CONFIGURATION 3
3.1 Simple 3
3.2 Custom 3
4. COMPLIANCE EXTRACTION 4
5. OTHER CONSIDERATIONS 5
5.1 The libvirtd Service 5
5.2 FIPS Mode 5
5.3 Reboot for SELINUX Configuration Changes 5

1. BACKGROUND

Ansible is an open source, cross-platform configuration management solution used to define and enforce system and application configurations. This package provides Ansible configurations that implement most of the Red Hat Enterprise Linux STIG. While the content has been tested during development, all possible system and environmental factors could not be tested. Before using this content in a production environment, please perform testing with the intended settings in your own test environment. There is no mandate to use this content; it is published as a resource to assist in the application of security guidance to your systems. Use it in the manner and to the extent that it assists with this goal.

2. INSTALLATION

The following instructions are for standalone installation using ansible-playbook for testing purposes. A production environment may additionally use Ansible Tower. See here for details.

2.1 Installing Ansible

Newer versions of Ansible are in the Red Hat Enterprise Linux 7 EPEL repository. To install it, run the following:

```
sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
sudo yum install ansible
```

For other installation methods, see here.

2.2 Extracting

Unzip the **rhel7STIG-ansible.zip**.

3. CONFIGURATION

3.1 Simple

To apply the default STIG Ansible configuration to the local machine only, run the **`enforce.sh`** script to enforce the STIG. Please note that Ansible will refuse to reboot the local machine automatically (refer to section 4.3). To tailor the configuration, follow the steps in the next section.

3.2 Custom

To customize, create a YAML (.yml) file containing just the variables to customize from the variables named in the **`roles/rhel7STIG/defaults/main.yml`** file. This file contains configuration data to define which configuration settings to manage and the values for these settings. Edit the newly created configuration file in a text editor to best suit each system's requirements as needed. For example, if you wanted to turn off STIG rule ID 204417, you would set the "Manage" attribute equal to **`False`**. If you wanted to set STIG rule ID 204423's minimum password length to 20, you would set the "**`_etc_security_pwquality_conf_Line`**" attribute to **`'minlen = 20'`**.

```
rhel7STIG_stigrule_204417_Manage: False
rhel7STIG_stigrule_204417__etc_libuser_conf_Value: 'sha512'

rhel7STIG_stigrule_204423_Manage: True
rhel7STIG_stigrule_204423__etc_security_pwquality_conf_Line: 'minlen = 20'
```

To use the newly created, custom variables file, edit **`site.yml`** to include it. See the highlighted lines to add below:

```
- hosts: localhost
  gather_facts: no
  vars_files:
    - /path/to/custom/vars.yml
  roles:
  - rhel7STIG
```

For more information on variables, see here. For more information on YAML, see here.

4. COMPLIANCE EXTRACTION

This compliance extraction methodology returns results based on a system's compliance with the enforcement content. This may be different from STIG compliance. For example, multiple values may be allowed by the STIG but will be marked as "fail" if the value does not match the single exact value in the enforcement content. Additionally, if a value is customized in such a way to violate a STIG rule it will be marked as "pass" since it matches the enforcement content's expected value.

At the completion of a successful Ansible playbook play content extraction of the configuration results into XCCDF results can be performed via an Ansible callback plugin. Use of this plugin can be controlled via modification of the follow variable in the ansible.cfg file to include the name of the plugin to use:

```
[defaults]
callback_whitelist = stig_xml
```

Configuration of the plugin is controlled via creation/modification of the following environment variables:

- **`export STIG_PATH=/path/to/stig/U_Red_Hat_Enterprise_Linux_7_V2R3_Manual_STIG-xccdf.xml`**
- **`export XML_PATH=/path/where/to/write/results.xml`**

The above environmental variables control the plugin writing the XCCDF results to the file **`XML_PATH`** using the STIG at path **`STIG_PATH`**. The XCCDF results file is output by default to **`/tmp/xccdf-results.xml`**
Note: the STIG provided above should match the STIG release and version number that the Ansible content is built for.

Ansible provides means of checking compliance without enforcement called **`--check`** (aka "dry run"). To use ths mode, run the following:

```
ansible-playbook -v -b -i /dev/null --check site.yml
```

5. OTHER CONSIDERATIONS

5.1 The libvirtd Service

The libvirtd service modifies some sysctl values upon startup that are relevant to STIG compliance. To prevent this from occurring, disable the libvirtd service if not required by running **`systemctl disable libvirtd.service`** before rebooting.

5.2 FIPS Mode

This content does not configure the system for FIPS mode. We recommend configuring the system for FIPS mode during initial installation. Additional information for enabling FIPS mode is available here.

5.3 Reboot for SELINUX Configuration Changes

If this content makes changes to the SELINUX configuration, it will restart the system to allow the new settings to take effect. If an automatic restart is not desired, disable management of these rules.

When run against the local system and a reboot is required for a setting change, Ansible will refuse to reboot the system with the following message:

```
fatal: [localhost]: FAILED! => {"changed": false, "elapsed": 0, "msg":
"Running reboot with local connection would reboot the control node.",
"rebooted": false}
```

The system may be rebooted manually to apply the changes requiring it. For remote systems, Ansible will perform the remote system reboot.