RHELevant Security Practices

Joshua Loscar
Senior Technical Account Manager

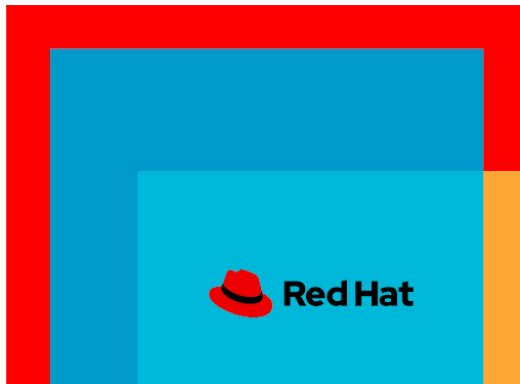
Nate Lager
Senior Technical Marketing Manager

Agenda

What we'll discuss today

- whoami
- Lab Setup
- KVM for Local Testing
- Ansible 101
- Creating a system baseline
- Theory, Threats, and Tools
- Common Misconfigurations
- Green Field Implementation
- Web Console

whoami



Joshua Loscar
Jscar

- 12+ Years In IT
  - Built 200+ Pc's
  - Homelab Enthusiast
  - 48 Raspberry Pi & counting
- 6 Years With The DoD
  - Hardening Via STIGs
  - Risk Management (RMF)
- 11 Years Attending & Contributing To Cyber Security Conferences
  - DEFCON
  - BSides
- 10 Years Running A Hackerspace
  - Tutor Security+
  - Online Security & Privacy
- Content Creator
  - Host 4 Year Podcast on Cybersecurity
  - Guest Host on Infosec Podcast
  - Publishing Videos for online training

jscar@redhat.com

Twitter @jscar_hawk

https://jscar-hawk.com

Red Hat

Nate Lager

- 26 Years In IT
  - Front-line ISP support
  - Level 2 network/connectivity support
  - Systems/Network admin in web hosting
  - Sr. Systems admin in Higher Ed
  - Red Hat Platform TAM
  - General passion for Tech, Gadgets, and Home-labbing
  - Learned the alphabet on my Dad's TI-99/4a
- Unhealthy passion for security
  - 10 year DerbyCon attendee
  - BSides Delaware speaker
  - DEFCON 610 group admin since 2017
- Content Creation, Videos, blogs, you name it
  - The Iron Sysadmin
  - Red Hat Technical Marketing
- Jeep Guy!
  - Driving and Modfying Jeeps for almost as long as I've been in IT
  - SWBCrawler

nlager@redhat.com

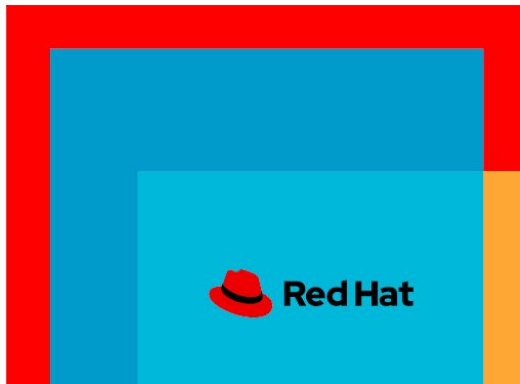
Fedi: @gangrif@social.undrground.org

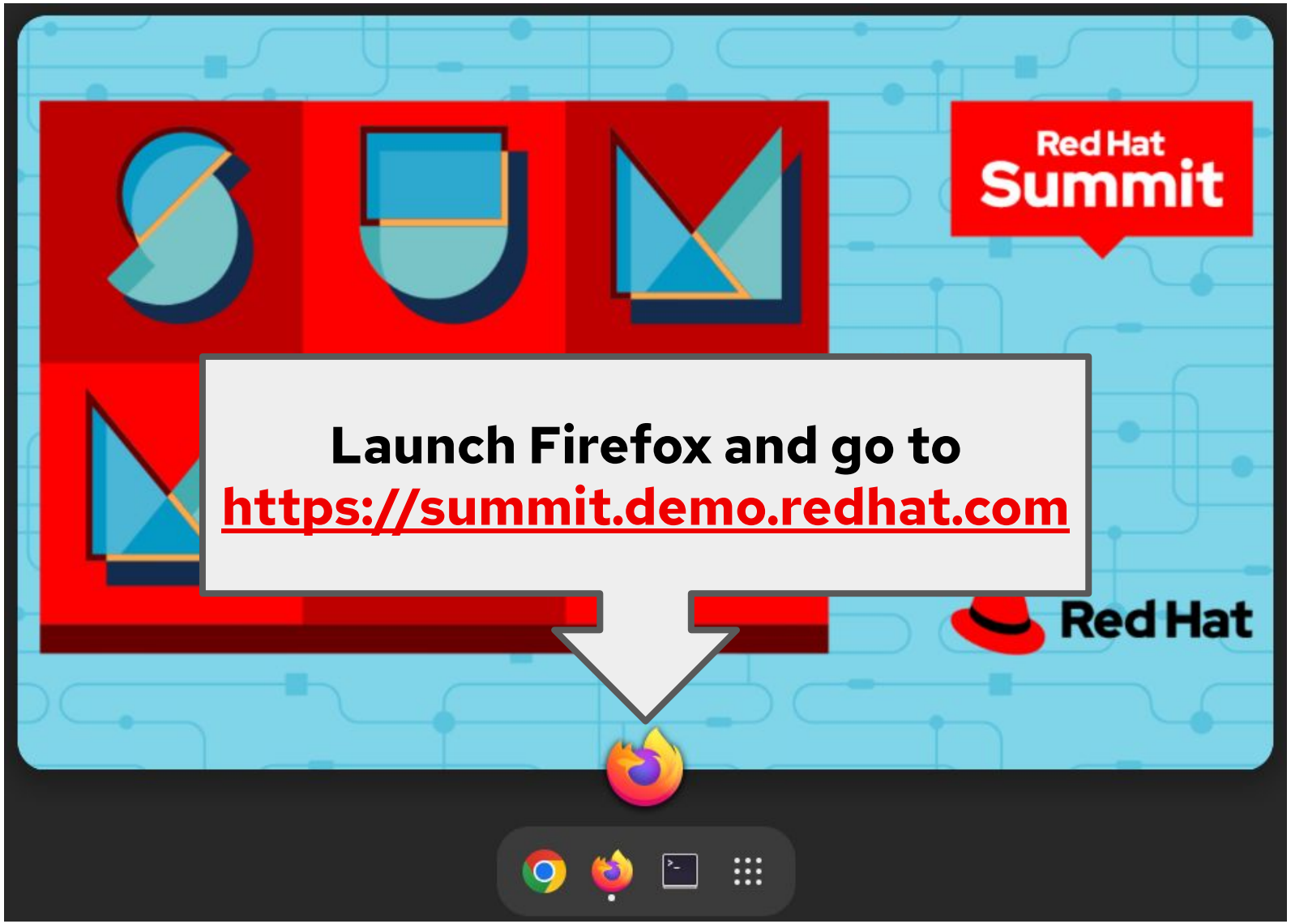
https://www.undrground.org

https://www.ironsysadmin.com

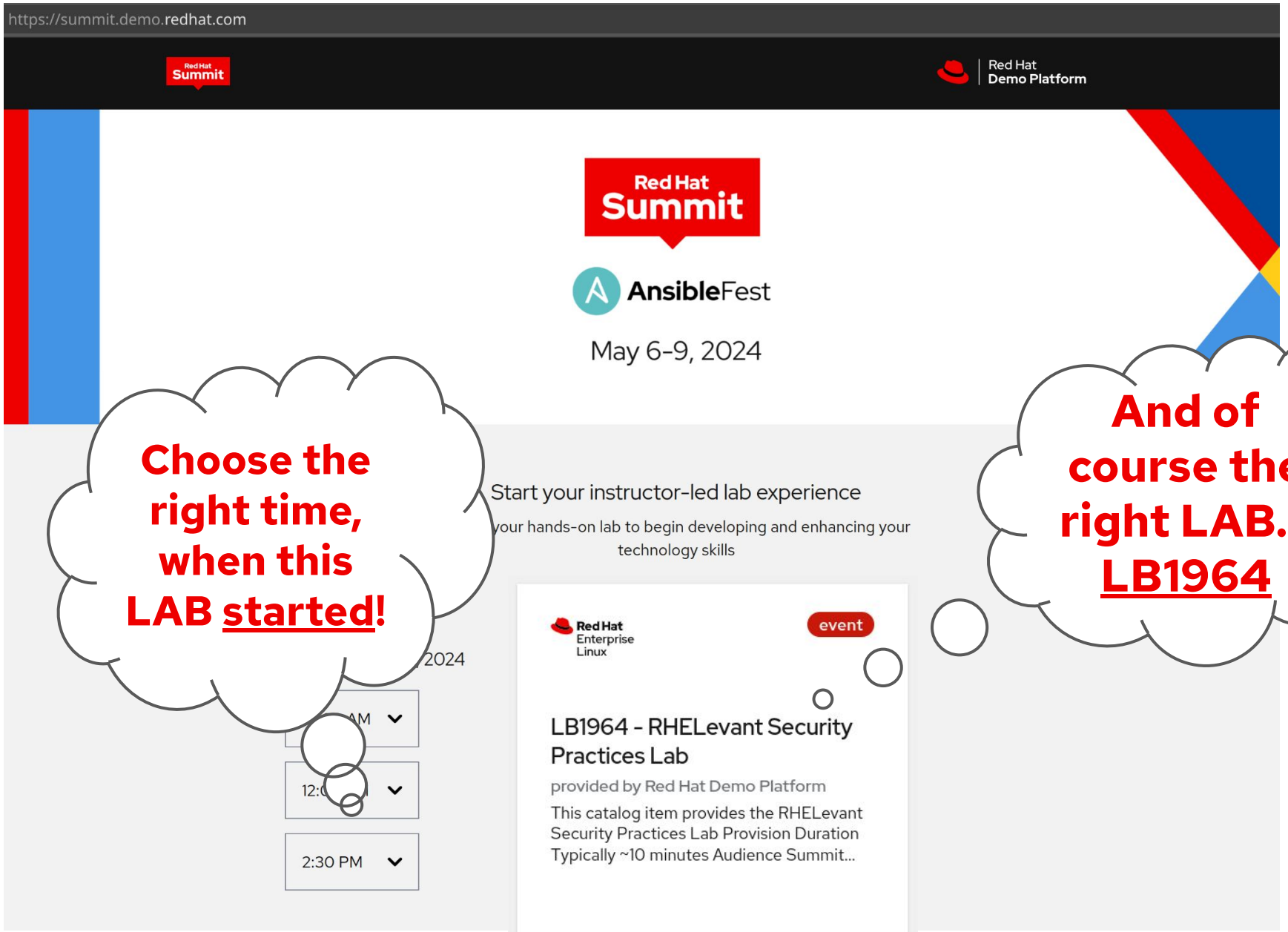
Red Hat

Lab Setup

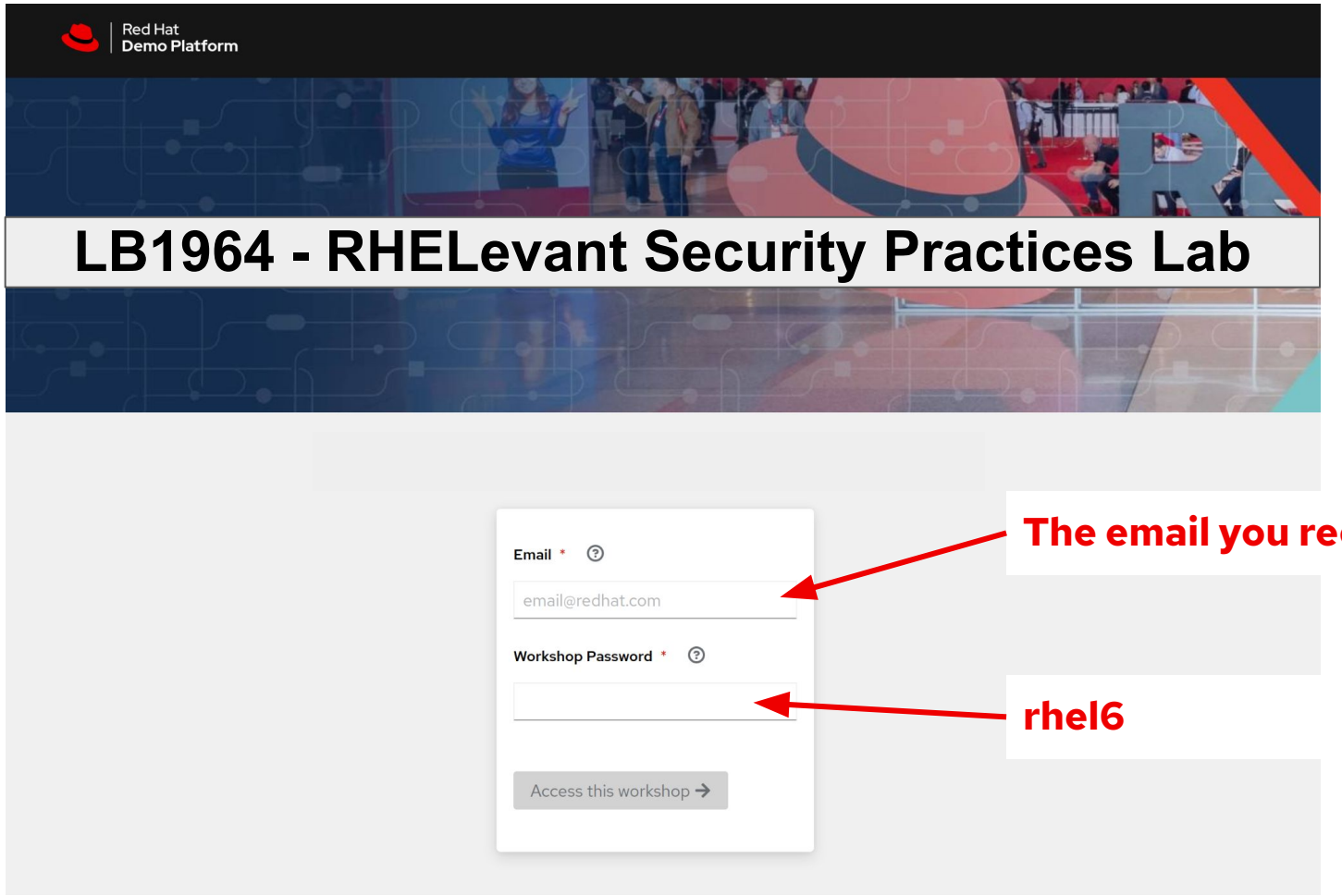



Red Hat
Summit
Launch Firefox and go to
https://summit.demo.redhat.com
Red Hat

Red Hat

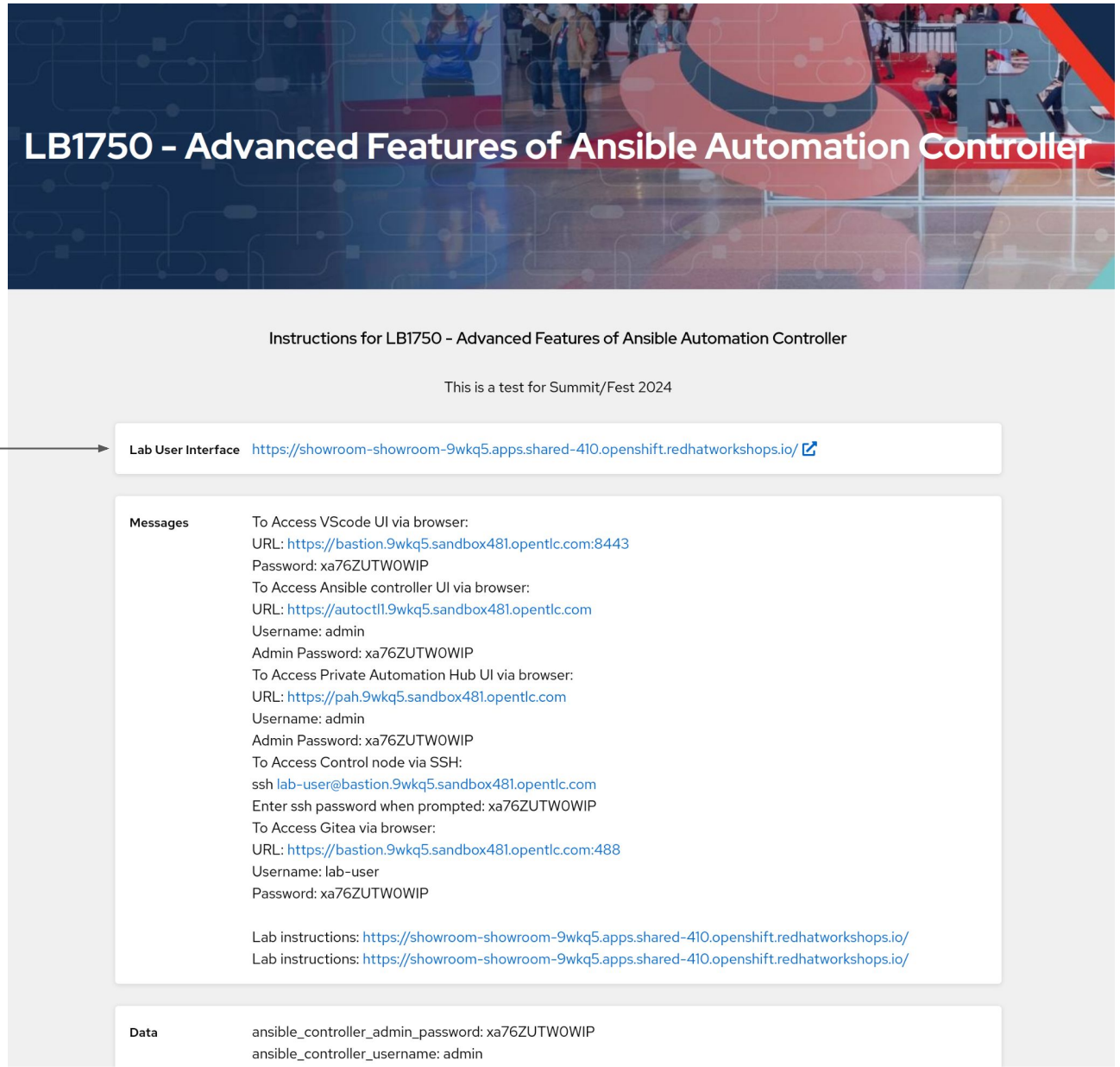

https://summit.demo.redhat.com
Red Hat Summit
Red Hat Demo Platform
Red Hat Summit
AnsibleFest
May 6-9, 2024
Choose the right time, when this LAB started!
And of course the right LAB... LB1964
Start your instructor-led lab experience
your hands-on lab to begin developing and enhancing your technology skills
2024
2:30 PM
Red Hat Enterprise Linux
event
LB1964 - RHELevant Security Practices Lab
provided by Red Hat Demo Platform
This catalog item provides the RHELevant Security Practices Lab Provision Duration Typically ~10 minutes Audience Summit...

Red Hat



Make sure you remember the (unique) e-mail address you use!
It will allow you to come back to your lab if you lose connection.

Red Hat

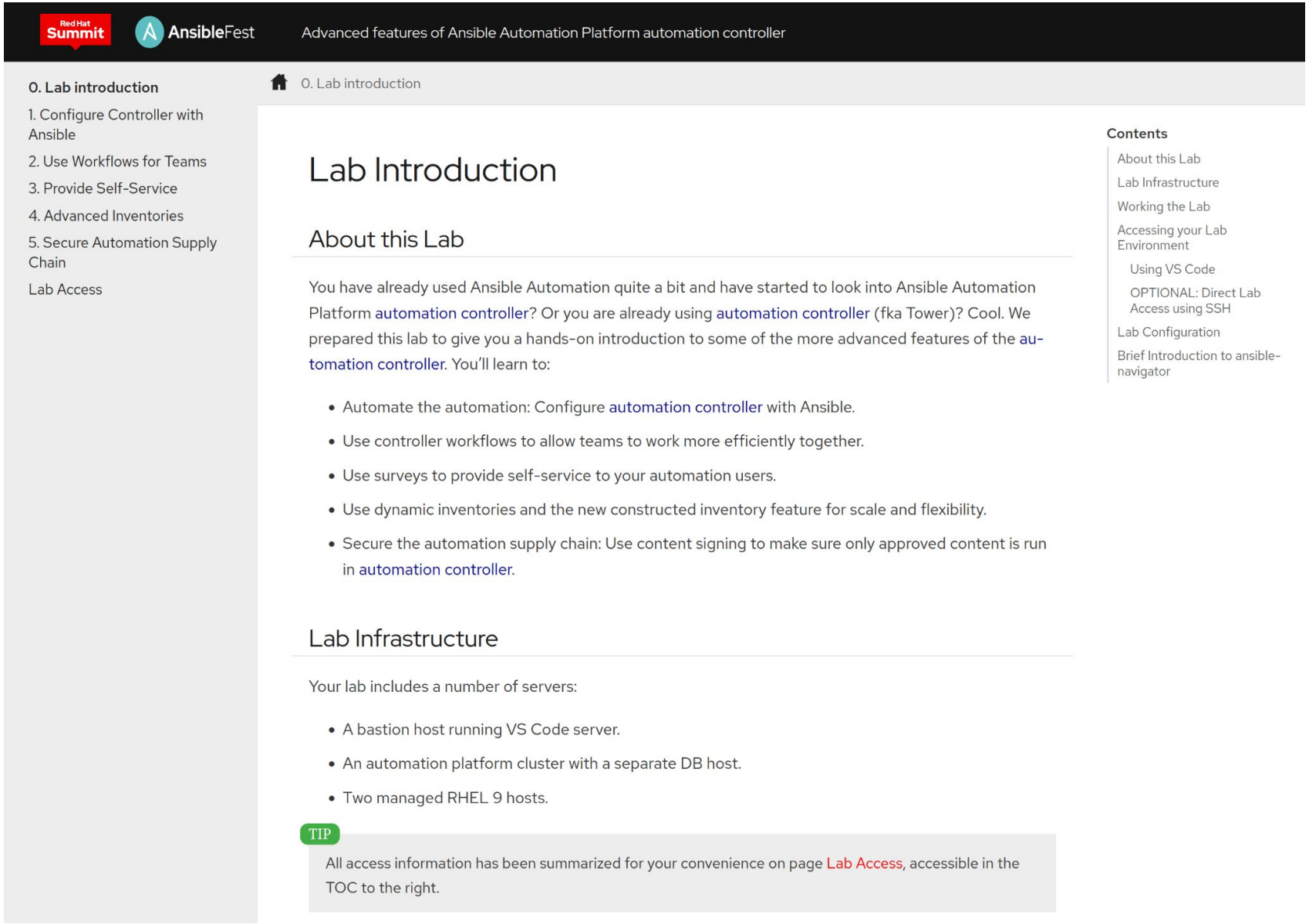


Click on the Lab User Interface link
and
follow the instructions

Red Hat

Lab Guide

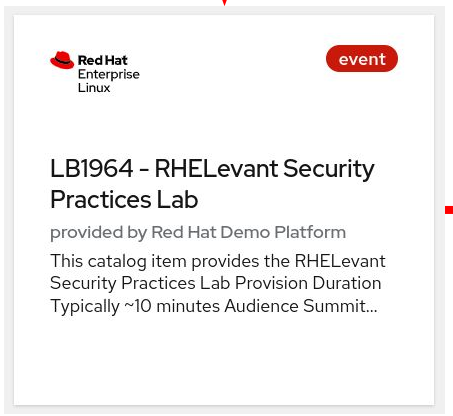
self-paced



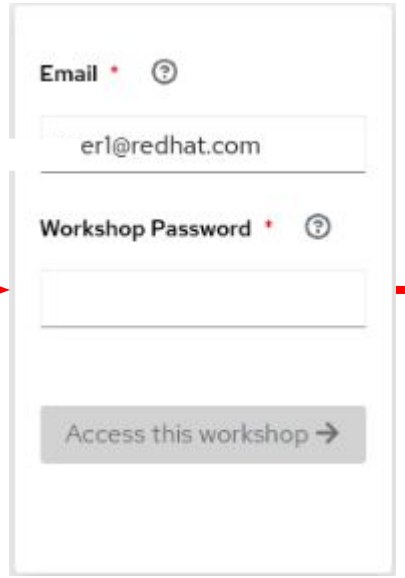
Red Hat

https://summit.demo.redhat.com

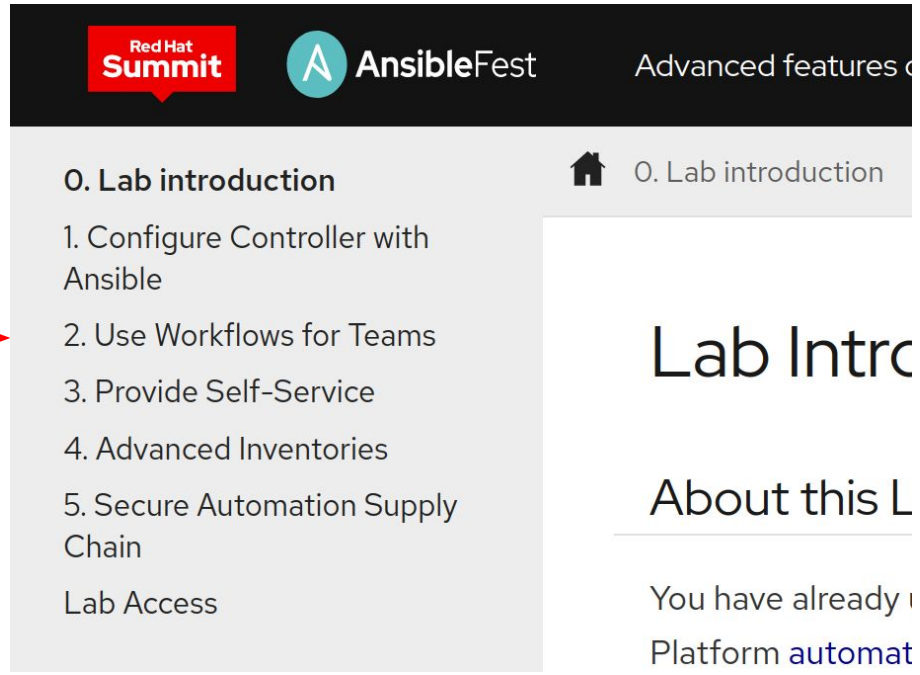
SELECT TIME



Select Lab



Login with your email and the password "gamepower?"



Follow the lab guide in your personalized "Showroom"

We're here to help, unstuck you and answer your questions

Red Hat

RHELevant Security Practices

Joshua Loscar
Senior Technical Account Manager

Nate Lager
Senior Technical Marketing Manager

Question for the audience

- Where from around the world are you coming from?
- What is your role in your organization?
- Who has used Ansible?
- Who has to harden their linux machines?

Lab Overview

How we will work on the Lab:

- We will do a 5-10 minute Presentation
- Then we will do a 10-15 minute Hands on Demo

Question or Lab Issues

At any point If you have any questions,

raise your Right hand and a Lab Volunteer will bring you a Mic.

If you are having trouble with the lab,

raise your Left hand and we will come to assist

Question

Lab Issue

TL;DR

You Can Do This At Home Or On Your Own Laptop

github.com/rhpds/summit_2024_RHELevant_Security_Practices_Lab_LB1964

Red Hat

Lab Setup

RHELevant Security Practices Lab - LB1964

Contents

Welcome to RHELevant Security Practices Lab! We hope you're ready for a dive into security practices with a couple of techies who love to talk about Red Hat Enterprise Linux (RHEL)!

Why is this lab important?

We're glad you asked.

RHEL is packed with security features that dont just make you or your administrators sleep better at night, they help you hit those compliance check-lists that we know matter so much to your CISO.

RHEL is a great platform right out of the box, but we can't make all the security decisions for you. So the base RHEL installation is only hardened to the point where we think it's a great base for you to customise to meet your needs. In this lab we're going to walk you through some practical examples of many of these features, so you can get your hands dirty and see how they work and what benefits they bring.

What will we cover?

In today's lab, we are going to take you down the path to better understand your lab, the status of the machines, and how then to make them more secure then before.

- Lab 1 RHPDS Lab Setup
  - Lab 1 alt KVM for Local testing

Terminal

Red Hat Demo Platform

```
Warning: Permanently added 'bastion.ppm4n.sandbox576.opentlc.com' (ED25519) to the list of known hos
[lab-user@bastion ~]$
```

to walk you through some practical examples of many of these features, so you can get your hands dirty and see how they work and what benefits they bring.

What will we cover?

In today's lab, we are going to take you down the path to better understand your lab, the status of the machines, and how then to make them more secure then before.

- Lab 1 RHPDS Lab Setup
  - Lab 1 alt KVM for Local testing
- Lab 2 Ansible 101
- Lab 3 Creating a system baseline
- Lab 4 Theory, Threats, and Tools
- Lab 5 Common Misconfigurations
- Lab 6 Green Field Implementation
- Lab 7 Web Console Administration

Lab Access

The terminal window to your right is **already** logged into the lab environment as the `lab-user` user via `ssh`. All steps of this lab are to be completed as the `lab-user` user on the bastion server.

You should be able to click the clipboard icon in the code box to copy the command, and you will eather use the middle mouse button or right click and paste. The key-board shortcuts for copy and paste (ctrl+c and ctrl+v) wont't work in this.

Next
Lab 1 RHPDS Lab Setup >

Terminal

Red Hat Demo Platform

```
Warning: Permanently added 'bastion.ppm4n.sandbox576.opentlc.com' (ED25519) to the list of known hos
[lab-user@bastion ~]$
```

RHPDS Lab Setup

Lab Length Medium/Average (~10-20 mins)

Goal Become familiar with the Summit lab environment

1.1: Introduction

If you are in a live classroom or in a phisical workshop, follow these instructions.

If you want to do this at home or on your own machine, you can follow the lab_1_alt_kvm_lab_for_local_testing.adoc which is after this page.

In this scenario you are a system administrator given a task to harden a Red Hat Enterprise Linux system. You have no idea what is on the system, what it is running, the hardware, or software installed. You are working on a black box.

You are given credentials that should give you access to the machine. You are given a Jump Box also known as a bastion host or as a control node, and a username to log in with.

Instead of having to type in the username and password, we are going to set up ssh keys for authentication. In this lab, ssh keys have been made and put on the end machine.

So first thing we need to do is to copy the SSH key from our bastion host to connect to the machines we will be working on.

There are 2 machines we will be connecting to.

- bastion
- node1

1.2: Create a SSH key pair

From your bastion host, use the open terminal on the right side of the screen

Terminal

Red Hat Demo Platform

```
Warning: Permanently added 'bastion.ppm4n.sandbox576.opentlc.com' (ED25519) to the list of known hos
[lab-user@bastion ~]$
```

1.2: Create a SSH key pair

From your bastion host, use the open terminal on the right side of the screen.

You should see a prompt with your current username and the hostname of the machine you are logged into.

```
[lab-user@bastion ~]$
```

Explanation of what you see in the terminal

`lab-user` is the current logged in user `@` is the at symbol which shows that you are connecting to a remote machine `bastion` is the current machine hostname

We want to find out if the current machine has any ssh keys under the users '.ssh' directory.

You should be able to click the clipboard icon in the code box to copy the command, and you will eather use the middle mouse button or right click and paste. The keyboard shortcuts for copy and paste (ctrl+c and ctrl+v) wont't work in this.

```
ls ~/.ssh/
```

Explanation of this command:

`ls` lists the files in the working directory.

`~` aka tilda key, is a shortcut to reffer to your users home directory.

`~/.ssh/` means it's going to /home/username/.ssh/

Next we see there are files in the directory.

You should now see 2 randomly named files called `xxxxxxx.pem` and `xxxxxxx.pub`

```
[lab-user@bastion ~]$ ls ~/.ssh/
authorized_keys  config  xxxxxxx.pem  xxxxxxx.pub
```

Terminal

Red Hat Demo Platform

```
Warning: Permanently added 'bastion.ppm4n.sandbox576.opentlc.com' (ED25519) to the list of known hos
[lab-user@bastion ~]$ ls ~/.ssh/
authorized_keys  config  ppm4nkey.pem  ppm4nkey.pub
[lab-user@bastion ~]$
```

1.3 Logging onto node1

Up to this point we have been using the lab-user account on our bastion host. On other machine 'node1' we have a different user we need to log in with.,

We want to test our credentials by logging into the node machine by typing the following into the terminal.

```
ssh ec2-user@node1
```

Explanation of this command:

`ssh` Secure Shell, is a cryptographic network protocol that enables secure remote connections and data sharing between two computers over an unsecured network

`ec2-user` the account username of the machine we want to log into

`@` the address we want to connect to

`node1` is the name of the device we want to log into. This could also be an Ip address or a FQDN

If you where successful in your connection, you should see a prompt with your current username and the hostname of the machine you are logged into.

```
[ec2-user@node1 ~]$
```

For one of our upcoming steps, we need to find out what our ip address for node 1 is. Your IP WILL BE DIFFERENT from what you see in the documentaion, and it will be be different from the other people in the workshop.

In your terminal type

```
ip addr
```

Explanation of this command:

`ip addr` lists each of your networking devices and the infomation about them

Terminal

Red Hat Demo Platform

```
Warning: Permanently added 'bastion.ppm4n.sandbox576.opentlc.com' (ED25519) to the list of known hos
[lab-user@bastion ~]$ ls ~/.ssh/
authorized_keys  config  ppm4nkey.pem  ppm4nkey.pub
[lab-user@bastion ~]$ ssh ec2-user@node1
```

1.3 Logging onto node1

Up to this point we have been using the lab-user account on our bastion host. On other machine 'node1' we have a different user we need to log in with.,

We want to test our credentials by logging into the node machine by typing the following into the terminal.

```
ssh ec2-user@node1
```

Explanation of this command:

`ssh` Secure Shell, is a cryptographic network protocol that enables secure remote connections and data sharing between two computers over an unsecured network

`ec2-user` the account username of the machine we want to log into

`@` the address we want to connect to

`node1` is the name of the device we want to log into. This could also be an Ip address or a FQDN

If you where successful in your connection, you should see a prompt with your current username and the hostname of the machine you are logged into.

```
[ec2-user@node1 ~]$
```

For one of our upcoming steps, we need to find out what our ip address for node 1 is. Your IP WILL BE DIFFERENT from what you see in the documentaion, and it will be be different from the other people in the workshop.

In your terminal type

```
ip addr
```

Explanation of this command:

`ip addr` lists each of your networking devices and the infomation about them

Terminal

Red Hat Demo Platform

```
Warning: Permanently added 'bastion.ppm4n.sandbox576.opentlc.com' (ED25519) to the list of known hosts.
[lab-user@bastion ~]$ ls ~/.ssh/
authorized_keys  config  ppm4nkey.pem  ppm4nkey.pub
[lab-user@bastion ~]$ ssh ec2-user@node1
Warning: Permanently added 'node1' (ED25519) to the list of known hosts.
Last login: Wed May  8 14:14:53 2024 from 192.168.0.128
[ec2-user@node1 ~]$
```

In your terminal type

```
ip addr
```

Explanation of this command:

`ip addr` lists each of your networking devices and the infomation about them.

You want to find the ip address for the ethernet interface.

You should have a few different devices.

```
<---------Output_Abbreviated------------------>
1: lo: <LOOPBACK,UP,LOWER_UP
   link/loopback
   inet 127.0.0.1
2: enp9s0u2u1u2 <NO-CARRIER,BROADCAST,MULTICAST,UP>
   link/ether
   inet 192.168.0.112
<---------Output_Abbreviated------------------>
```

It will be the interface with the `link/ether`, and make note of that IP Address for a future step.

You can now exit the node machine by typing,

```
exit
```

Which should take you back to your bastion host.

Prev RHELevant Security Practices

Next Lab 1 alt KVM for Local testing

Terminal

Red Hat Demo Platform

```
Warning: Permanently added 'bastion.ppm4n.sandbox576.opentlc.com' (ED25519) to the list of known hosts.
[lab-user@bastion ~]$ ls ~/.ssh/
authorized_keys  config  ppm4nkey.pem  ppm4nkey.pub
[lab-user@bastion ~]$ ssh ec2-user@node1
Warning: Permanently added 'node1' (ED25519) to the list of known hosts.
Last login: Wed May  8 14:14:53 2024 from 192.168.0.128
[ec2-user@node1 ~]$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc mq state UP group default qlen 1000
    link/ether 06:65:63:ce:70:d9 brd ff:ff:ff:ff:ff:ff
    altname enp0s5
    altname ens5
    inet 192.168.0.141/24 brd 192.168.0.255 scope global dynamic noprefixroute eth0
       valid_lft 3381sec preferred_lft 3381sec
    inet6 fe80::465:63ff:fece:70d9/64 scope link
       valid_lft forever preferred_lft forever
[ec2-user@node1 ~]$
```

In your terminal type

```
ip addr
```

Explanation of this command:

`ip addr` lists each of your networking devices and the infomation about them.

You want to find the ip address for the ethernet interface.

You should have a few different devices.

```
<---------Output_Abbreviated------------------->
1: lo: <LOOPBACK,UP,LOWER_UP
   link/loopback
   inet 127.0.0.1
2: enp9s0u2u1u2 <NO-CARRIER,BROADCAST,MULTICAST,UP>
   link/ether
   inet 192.168.0.112
<---------Output_Abbreviated------------------->
```

It will be the interface with the `link/ether`, and make note of that IP Address for a future step.

You can now exit the node machine by typing,

```
exit
```

Which should take you back to your bastion host.

Prev
RHELevant Security Practices

Next
Lab 1 alt KVM for Local testing

Terminal

Red Hat Demo Platform

```
Warning: Permanently added 'bastion.ppm4n.sandbox576.opentlc.com' (ED25519) to the list of known hosts.
[lab-user@bastion ~]$ ls ~/.ssh/
authorized_keys  config  ppm4nkey.pem  ppm4nkey.pub
[lab-user@bastion ~]$ ssh ec2-user@node1
Warning: Permanently added 'node1' (ED25519) to the list of known hosts.
Last login: Wed May  8 14:14:53 2024 from 192.168.0.128
[ec2-user@node1 ~]$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc mq state UP group default qlen 1000
    link/ether 06:65:63:ce:70:d9 brd ff:ff:ff:ff:ff:ff
    altname enp0s5
    altname ens5
    inet 192.168.0.141/24 brd 192.168.0.255 scope global dynamic noprefixroute eth0
       valid_lft 3381sec preferred_lft 3381sec
    inet6 fe80::465:63ff:fece:70d9/64 scope link
       valid_lft forever preferred_lft forever
[ec2-user@node1 ~]$
```

In your terminal type

```
ip addr
```

Explanation of this command:

`ip addr` lists each of your networking devices and the infomation about them.

You want to find the ip address for the ethernet interface.

You should have a few different devices.

```
<---------Output_Abbreviated------------------->
1: lo: <LOOPBACK,UP,LOWER_UP
   link/loopback
   inet 127.0.0.1
2: enp9s0u2u1u2 <NO-CARRIER,BROADCAST,MULTICAST,UP>
   link/ether
   inet 192.168.0.112
<---------Output_Abbreviated------------------->
```

It will be the interface with the `link/ether`, and make note of that IP Address for a future step.

You can now exit the node machine by typing,

```
exit
```

Which should take you back to your bastion host.

Prev
RHELevant Security Practices

Next
Lab 1 alt KVM for Local testing

Terminal — Red Hat Demo Platform

```
Warning: Permanently added 'bastion.ppm4n.sandbox576.opentlc.com' (ED25519) to the list of known hosts.
[lab-user@bastion ~]$ ls ~/.ssh/
authorized_keys  config  ppm4nkey.pem  ppm4nkey.pub
[lab-user@bastion ~]$ ssh ec2-user@node1
Warning: Permanently added 'node1' (ED25519) to the list of known hosts.
Last login: Wed May  8 14:14:53 2024 from 192.168.0.128
[ec2-user@node1 ~]$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc mq state UP group default qlen 1000
    link/ether 06:65:63:ce:70:d9 brd ff:ff:ff:ff:ff:ff
    altname enp0s5
    altname ens5
    inet 192.168.0.141/24 brd 192.168.0.255 scope global dynamic noprefixroute eth0
       valid_lft 3381sec preferred_lft 3381sec
    inet6 fe80::465:63ff:fece:70d9/64 scope link
       valid_lft forever preferred_lft forever
[ec2-user@node1 ~]$
```

Take Note Of Your IP, You'll need it shortly

My IP is 192.168.0.141

Yours will be different.

In your terminal type

```
ip addr
```

Explanation of this command:

`ip addr` lists each of your networking devices and the infomation about them.

You want to find the ip address for the ethernet interface.

You should have a few different devices.

```
<---------Output_Abbreviated------------------->
1: lo: <LOOPBACK,UP,LOWER_UP
   link/loopback
   inet 127.0.0.1
2: enp9s0u2u1u2 <NO-CARRIER,BROADCAST,MULTICAST,UP>
   link/ether
   inet 192.168.0.112
<---------Output_Abbreviated------------------->
```

It will be the interface with the `link/ether`, and make note of that IP Address for a future step.

You can now exit the node machine by typing,

```
exit
```

Which should take you back to your bastion host.

Prev RHELevant Security Practices

Next Lab 1 alt KVM for Local testing

Terminal

Red Hat Demo Platform

```
Warning: Permanently added 'bastion.ppm4n.sandbox576.opentlc.com' (ED25519) to the list of known hosts.
[lab-user@bastion ~]$ ls ~/.ssh/
authorized_keys  config  ppm4nkey.pem  ppm4nkey.pub
[lab-user@bastion ~]$ ssh ec2-user@node1
Warning: Permanently added 'node1' (ED25519) to the list of known hosts.
Last login: Wed May  8 14:14:53 2024 from 192.168.0.128
[ec2-user@node1 ~]$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc mq state UP group default qlen 1000
    link/ether 06:65:63:ce:70:d9 brd ff:ff:ff:ff:ff:ff
    altname enp0s5
    altname ens5
    inet 192.168.0.141/24 brd 192.168.0.255 scope global dynamic noprefixroute eth0
       valid_lft 3381sec preferred_lft 3381sec
    inet6 fe80::465:63ff:fece:70d9/64 scope link
       valid_lft forever preferred_lft forever
[ec2-user@node1 ~]$ exit
logout
Shared connection to node1 closed.
[lab-user@bastion ~]$
```

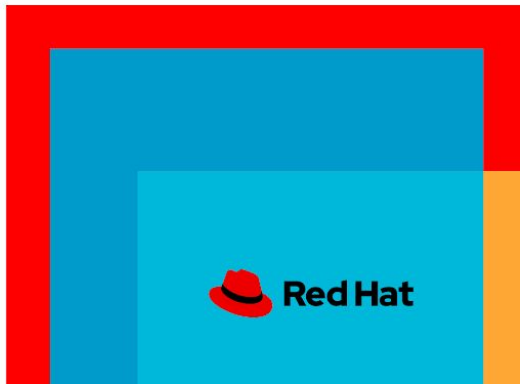
Lab Setup

- Introduction
- Create a SSH key pair
- Logging onto node1
- Install Web Console

Question

Lab Issue

KVM for Local Testing



KVM for Local Testing

How to set up a Local Linux Machine with KVM to do this lab

- Introduction
- Expectations of your Host OS
- Log into Red Hat Account
- RHEL KVM Downloads
- Installing the Cockpit Project aka Web Console

KVM for Local Testing

How to set up a Local Linux Machine with KVM to do this lab

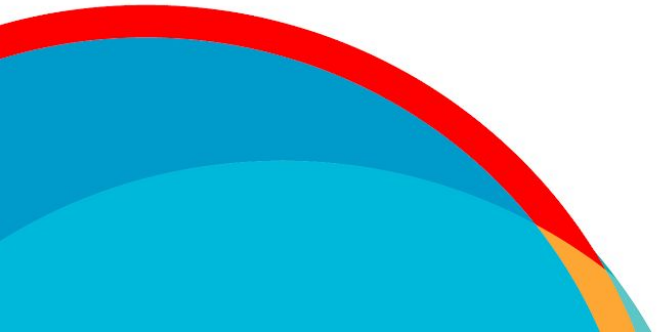
- Install Cockpit Project with Virtual Machine Management
- KVM Image Prep
- Configure Cockpit with storage
- Configure Cockpit for Virtual Machines
- Create ec2-user account on vm

KVM for Local Testing

How to set up a Local Linux Machine with KVM to do this lab

- Create a SSH key pair
- Push your public ssh Key to node1
- Verifying Key Pair Creation
- Verifying ssh keys work
- Copy your ssh key to your local machine

Ansible 101



Red Hat

Ansible 101

Lab Length Medium/Average (~10-20 mins)

Goal Become familiar with the basics of ansible

2.1 Introduction to Ansible

Ansible provides open-source automation that reduces complexity and runs everywhere.

Using Ansible lets you automate virtually any task.

Here are some common use cases for Ansible:

- Eliminate repetition and simplify workflows
- Manage and maintain system configuration
- Continuously deploy complex software
- Perform zero-downtime rolling updates

Ansible uses simple, human-readable scripts called playbooks to automate your tasks.

You declare the desired state of a local or remote system in your playbook.

Ansible ensures that the system remains in that state.

As automation technology, Ansible is designed around the following principles:

- **Agent-less architecture**
  - Low maintenance overhead by avoiding the installation of additional software across IT infrastructure.
- **Simplicity**
  - Automation playbooks use straightforward YAML syntax for code that reads like documentation. Ansible is also decentralized, using SSH existing OS credentials to access to remote machines


```
[lab-user@bastion ~]$
```

2.2 Install Ansible for Red Hat Enterprise Linux 9

If you use Ansible content within Red Hat Enterprise Linux (RHEL), you should know that important changes where made with RHEL 8.6 and 9.0.

RHEL 7 and RHEL 8.0-8.5 customers have access to a separate **Ansible Engine** repository.

In RHEL 8.6 and 9.0, customers will have access to **Ansible Core**, which will be included in the corresponding AppStream repository.

The move to Ansible Core in RHEL is being made to adapt to changes in the Ansible project.

For more info on the difference between Ansible core and engine

You'll be using Ansible core with RHEL 9, so lets get it installed.

```
sudo dnf install ansible-core -y
```

2.3 Basic Ansible setup for Red Hat Enterprise Linux 9

I like to create a working directory to work on my ansible projects in my home folder

```
mkdir ~/ansible
```

I also then like bring in a deafult ansible config file for later use.

```
ansible-config init --disabled > ~/ansible/ansible.cfg
```

We also need to create an inventory file to reference.

The inventory file is where we list our machines we will connect to.

In this lab we will connect to

- bastion
- node1

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ~]$
```

2.2 Install Ansible for Red Hat Enterprise Linux 9

If you use Ansible content within Red Hat Enterprise Linux (RHEL), you should know that important changes where made with RHEL 8.6 and 9.0.

RHEL 7 and RHEL 8.0-8.5 customers have access to a separate **Ansible Engine** repository.

In RHEL 8.6 and 9.0, customers will have access to **Ansible Core**, which will be included in the corresponding AppStream repository.

The move to Ansible Core in RHEL is being made to adapt to changes in the Ansible project.

For more info on the difference between Ansible core and engine

You'll be using Ansible core with RHEL 9, so lets get it installed.

```
sudo dnf install ansible-core -y
```

2.3 Basic Ansible setup for Red Hat Enterprise Linux 9

I like to create a working directory to work on my ansible projects in my home folder

```
mkdir ~/ansible
```

I also then like bring in a deafult ansible config file for later use.

```
ansible-config init --disabled > ~/ansible/ansible.cfg
```

We also need to create an inventory file to reference.

The inventory file is where we list our machines we will connect to.

In this lab we will connect to

- bastion
- node1

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ~]$ sudo dnf install ansible-core -y
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)     99 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)           119 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)        137 kB/s | 4.5 kB     00:00
Package ansible-core-1:2.14.9-1.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[lab-user@bastion ~]$
```

2.2 Install Ansible for Red Hat Enterprise Linux 9

If you use Ansible content within Red Hat Enterprise Linux (RHEL), you should know that important changes where made with RHEL 8.6 and 9.0.

RHEL 7 and RHEL 8.0-8.5 customers have access to a separate **Ansible Engine** repository.

In RHEL 8.6 and 9.0, customers will have access to **Ansible Core**, which will be included in the corresponding AppStream repository.

The move to Ansible Core in RHEL is being made to adapt to changes in the Ansible project.

For more info on the difference between Ansible core and engine

You'll be using Ansible core with RHEL 9, so lets get it installed.

```
sudo dnf install ansible-core -y
```

2.3 Basic Ansible setup for Red Hat Enterprise Linux 9

I like to create a working directory to work on my ansible projects in my home folder

```
mkdir ~/ansible
```

I also then like bring in a deafult ansible config file for later use.

```
ansible-config init --disabled > ~/ansible/ansible.cfg
```

We also need to create an inventory file to reference.

The inventory file is where we list our machines we will connect to.

In this lab we will connect to

- bastion
- node1

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ~]$ sudo dnf install ansible-core -y
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)     99 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)           119 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)        137 kB/s | 4.5 kB     00:00
Package ansible-core-1:2.14.9-1.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[lab-user@bastion ~]$ mkdir ~/ansible
[lab-user@bastion ~]$ ansible-config init --disabled > ~/ansible/ansible.cfg
[lab-user@bastion ~]$
```

We also need to create an inventory file to reference.

The inventory file is where we list our machines we will connect to.

In this lab we will connect to

- bastion
- node1

We will next create a simple inventory file for ansible to use to connect to the machines.

Check out our documentation on Ansible intro inventory.

We will be using the vi file editor.

If you have never worked with vi before, here is a very basic How to get started with the Vi editor

```
vi ~/ansible/inventory
```

This will open a new file named inventory.

To edit a file in vi, you will need to change to edit mode.

You will need to type `i` in your terminal to go into 'insert' mode.

You should now be able to to type, edit and paste into the ternimal to change the file.

You will want to copy the following and paste it into the terminal.

```
[bastion]
localhost


[nodes]
node1 ansible_host=YOURNODE1IPADDRESS
```

Under the nodes section is where you will put your node1 ip addesss you made note of

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ~]$ sudo dnf install ansible-core -y
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)            99 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)                  119 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)               137 kB/s | 4.5 kB     00:00
Package ansible-core-1:2.14.9-1.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[lab-user@bastion ~]$ mkdir ~/ansible
[lab-user@bastion ~]$ ansible-config init --disabled > ~/ansible/ansible.cfg
[lab-user@bastion ~]$
```

We also need to create an inventory file to reference.

The inventory file is where we list our machines we will connect to.

In this lab we will connect to

- bastion
- node1

We will next create a simple inventory file for ansible to use to connect to the machines.

Check out our documentation on Ansible intro inventory.

We will be using the vi file editor.

If you have never worked with vi before, here is a very basic How to get started with the Vi editor

```
vi ~/ansible/inventory
```

This will open a new file named inventory.

To edit a file in vi, you will need to change to edit mode.

You will need to type `i` in your terminal to go into 'insert' mode.

You should now be able to to type, edit and paste into the ternimal to change the file.

You will want to copy the following and paste it into the terminal.

```
[bastion]
localhost


[nodes]
node1 ansible_host=YOURNODE1IPADDRESS
```

Under the nodes section is where you will put your node1 ip addesss you made note of

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ~]$ sudo dnf install ansible-core -y
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)     99 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)           119 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)        137 kB/s | 4.5 kB     00:00
Package ansible-core-1:2.14.9-1.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[lab-user@bastion ~]$ mkdir ~/ansible
[lab-user@bastion ~]$ ansible-config init --disabled > ~/ansible/ansible.cfg
[lab-user@bastion ~]$ vi ~/ansible/inventory
```

We also need to create an inventory file to reference.

The inventory file is where we list our machines we will connect to.

In this lab we will connect to

- bastion
- node1

We will next create a simple inventory file for ansible to use to connect to the machines.

Check out our documentation on Ansible intro inventory.

We will be using the vi file editor.

If you have never worked with vi before, here is a very basic How to get started with the Vi editor

```
vi ~/ansible/inventory
```

This will open a new file named inventory.

To edit a file in vi, you will need to change to edit mode.

You will need to type `i` in your terminal to go into 'insert' mode.

You should now be able to to type, edit and paste into the ternimal to change the file.

You will want to copy the following and paste it into the terminal.

```
[bastion]
localhost


[nodes]
node1 ansible_host=YOURNODE1IPADDRESS
```

Under the nodes section is where you will put your node1 ip addesss you made note of

Terminal

Red Hat Demo Platform

If you have never worked with VI,
You will need to type 'i' to go into Insert mode

-- INSERT -- 0,1 All

We also need to create an inventory file to reference.

The inventory file is where we list our machines we will connect to.

In this lab we will connect to

- bastion
- node1

We will next create a simple inventory file for ansible to use to connect to the machines.

Check out our documentation on Ansible intro inventory.

We will be using the vi file editor.

If you have never worked with vi before, here is a very basic How to get started with the Vi editor

```
vi ~/ansible/inventory
```

This will open a new file named inventory.

To edit a file in vi, you will need to change to edit mode.

You will need to type `i` in your terminal to go into 'insert' mode.

You should now be able to to type, edit and paste into the ternimal to change the file.

You will want to copy the following and paste it into the terminal.

```
[bastion]
localhost


[nodes]
node1 ansible_host=YOURNODE1IPADDRESS
```

Under the nodes section is where you will put your node1 ip addesss you made note of

Terminal

Red Hat Demo Platform

```
[bastion]
localhost


[nodes]
node1 ansible_host=

-- INSERT --                                                    6,20          All
```

Delete the "YOURNODE1IPADDRESS" and add Your IP address that you took note of earlier.

We also need to create an inventory file to reference.

The inventory file is where we list our machines we will connect to.

In this lab we will connect to

- bastion
- node1

We will next create a simple inventory file for ansible to use to connect to the machines.

Check out our documentation on Ansible intro inventory.

We will be using the vi file editor.

If you have never worked with vi before, here is a very basic How to get started with the Vi editor

```
vi ~/ansible/inventory
```

This will open a new file named inventory.

To edit a file in vi, you will need to change to edit mode.

You will need to type `i` in your terminal to go into 'insert' mode.

You should now be able to to type, edit and paste into the ternimal to change the file.

You will want to copy the following and paste it into the terminal.

```
[bastion]
localhost


[nodes]
node1 ansible_host=YOURNODE1IPADDRESS
```

Under the nodes section is where you will put your node1 ip addesss you made note of

Terminal

Red Hat Demo Platform

```
[bastion]
localhost


[nodes]
node1 ansible_host=192.168.0.141
```

To save and exit vim
On your keyboard Hit the 'Esc' Key
Then press hold 'Shift' + ':'
This puts you in command mode
The type 'wq'
And hit 'Enter'

6,32 All

2.4 Basic Ansible smoke ping test

Lets make sure we can connect to both our machines

first lets change directories to our working folder

```
cd ~/ansible
```

Then issue a command to check the localhost (bastion) machine

```
ansible -m ping -i inventory localhost
```

Then issue a command to check the localhost (bastion) machine

```
ansible -m ping -i inventory node1
```

If everything is working correctly you should see something like this in your terminal

```
machinehostname | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

2.5 More Ansible Reference Documentation

Updates to using Ansible in RHEL 8.6 and 9.0

Updates to using Ansible Core in Red Hat Enterprise Linux

Using Ansible in RHEL 9

Red Hat Enterprise Linux (RHEL) System Roles

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ~]$ sudo dnf install ansible-core -y
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)            99 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)                  119 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)               137 kB/s | 4.5 kB     00:00
Package ansible-core-1:2.14.9-1.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[lab-user@bastion ~]$ mkdir ~/ansible
[lab-user@bastion ~]$ ansible-config init --disabled > ~/ansible/ansible.cfg
[lab-user@bastion ~]$ vi ~/ansible/inventory
[lab-user@bastion ~]$ vi ~/ansible/inventory
[lab-user@bastion ~]$
```

2.4 Basic Ansible smoke ping test

Lets make sure we can connect to both our machines

first lets change directories to our working folder

```
cd ~/ansible
```

Then issue a command to check the localhost (bastion) machine

```
ansible -m ping -i inventory localhost
```

Then issue a command to check the localhost (bastion) machine

```
ansible -m ping -i inventory node1
```

If everything is working correctly you should see something like this in your terminal

```
machinehostname | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

2.5 More Ansible Reference Documentation

Updates to using Ansible in RHEL 8.6 and 9.0

Updates to using Ansible Core in Red Hat Enterprise Linux

Using Ansible in RHEL 9

Red Hat Enterprise Linux (RHEL) System Roles

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ~]$ sudo dnf install ansible-core -y
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)            99 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)                  119 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)               137 kB/s | 4.5 kB     00:00
Package ansible-core-1:2.14.9-1.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[lab-user@bastion ~]$ mkdir ~/ansible
[lab-user@bastion ~]$ ansible-config init --disabled > ~/ansible/ansible.cfg
[lab-user@bastion ~]$ vi ~/ansible/inventory
[lab-user@bastion ~]$ vi ~/ansible/inventory
[lab-user@bastion ~]$ cd ~/ansible
[lab-user@bastion ansible]$
```

2.4 Basic Ansible smoke ping test

Lets make sure we can connect to both our machines

first lets change directories to our working folder

```
cd ~/ansible
```

Then issue a command to check the localhost (bastion) machine

```
ansible -m ping -i inventory localhost
```

Then issue a command to check the localhost (bastion) machine

```
ansible -m ping -i inventory node1
```

If everything is working correctly you should see something like this in your terminal

```
machinehostname | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

2.5 More Ansible Reference Documentation

Updates to using Ansible in RHEL 8.6 and 9.0

Updates to using Ansible Core in Red Hat Enterprise Linux

Using Ansible in RHEL 9

Red Hat Enterprise Linux (RHEL) System Roles

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ~]$ sudo dnf install ansible-core -y
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)   99 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)         119 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)      137 kB/s | 4.5 kB     00:00
Package ansible-core-1:2.14.9-1.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[lab-user@bastion ~]$ mkdir ~/ansible
[lab-user@bastion ~]$ ansible-config init --disabled > ~/ansible/ansible.cfg
[lab-user@bastion ~]$ vi ~/ansible/inventory
[lab-user@bastion ~]$ vi ~/ansible/inventory
[lab-user@bastion ~]$ cd ~/ansible
[lab-user@bastion ansible]$ ansible -m ping -i inventory localhost
localhost | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
[lab-user@bastion ansible]$
```

2.4 Basic Ansible smoke ping test

Lets make sure we can connect to both our machines

first lets change directories to our working folder

```
cd ~/ansible
```

Then issue a command to check the localhost (bastion) machine

```
ansible -m ping -i inventory localhost
```

Then issue a command to check the localhost (bastion) machine

```
ansible -m ping -i inventory node1
```

If everything is working correctly you should see something like this in your terminal

```
machinehostname | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

2.5 More Ansible Reference Documentation

Updates to using Ansible in RHEL 8.6 and 9.0

Updates to using Ansible Core in Red Hat Enterprise Linux

Using Ansible in RHEL 9

Red Hat Enterprise Linux (RHEL) System Roles

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ~]$ sudo dnf install ansible-core -y
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)            99 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)                  119 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)               137 kB/s | 4.5 kB     00:00
Package ansible-core-1:2.14.9-1.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[lab-user@bastion ~]$ mkdir ~/ansible
[lab-user@bastion ~]$ ansible-config init --disabled > ~/ansible/ansible.cfg
[lab-user@bastion ~]$ vi ~/ansible/inventory
[lab-user@bastion ~]$ vi ~/ansible/inventory
[lab-user@bastion ~]$ cd ~/ansible
[lab-user@bastion ansible]$ ansible -m ping -i inventory localhost
localhost | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
[lab-user@bastion ansible]$ ansible -m ping -i inventory node1
node1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
[lab-user@bastion ansible]$
```

Ansible 101

- Introduction to Ansible
- Install Ansible for RHEL 9
- Set Up Ansible User
- Create & use ssh key
- Inventory setup
- Ping Test

Question

Lab Issue

Creating a system baseline

Creating a system baseline

Contents

Lab Length Medium/Average (~10-20 mins)

Goal Capture a consistant machine state with SOSReport and Ansible

3.0 Capture machine baseline

We need to capture starting point for our systems.

There are a few different ways:

- sosreport
- Ansible facts
- Insights (out of scope for this lab)
- Satellite (out of scope for this lab)

In this lab scenario we are working on an air-gapped system (Not connected to the internet).

The first best way to do this is to use a built-in utility.

We are go use the built in sos utility, aka sosreport.

3.1 Sos utility

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$
```

3.1 Installing the sos package from the command line

To use the sos utility, install the sos package.

Prerequisites

You need `root` privileges.

Procedure

Install the `sos` package.

```
sudo dnf install sos -y
```

Verification steps

Use the rpm utility to verify that the sos package is installed.

```
sudo rpm -q sos
```

3.2 Generating an sos report from the command line

Use the `sos report` command to gather an sos report from a RHEL server.

Prerequisites

- You have installed the sos package.
- You need root privileges.

Procedure

Run the sos report command and follow the on-screen instructions.

You can add the `--upload option` to transfer the sos report to Red Hat immediately after generating it.

```
sudo sos report
```

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$
```

3.1 Installing the sos package from the command line

To use the sos utility, install the sos package.

Prerequisites

You need `root` privileges.

Procedure

Install the `sos` package.

```
sudo dnf install sos -y
```

Verification steps

Use the rpm utility to verify that the sos package is installed.

```
sudo rpm -q sos
```

3.2 Generating an sos report from the command line

Use the `sos report` command to gather an sos report from a RHEL server.

Prerequisites

- You have installed the sos package.
- You need root privileges.

Procedure

Run the sos report command and follow the on-screen instructions.

You can add the `--upload option` to transfer the sos report to Red Hat immediately after generating it.

```
sudo sos report
```

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$ sudo dnf install sos
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)          106 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)                  93 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)              113 kB/s | 4.5 kB     00:00
Dependencies resolved.
================================================================================================
 Package                  Architecture  Version            Repository                      Size
================================================================================================
Installing:
 sos                      noarch        4.7.0-1.el9        rhel-9-for-x86_64-baseos-rpms  1.1 M
Installing dependencies:
 python3-pexpect          noarch        4.8.0-7.el9        rhel-9-for-x86_64-baseos-rpms  155 k
 python3-ptyprocess       noarch        0.6.0-12.el9       rhel-9-for-x86_64-baseos-rpms   35 k

Transaction Summary
================================================================================================
Install  3 Packages

Total download size: 1.3 M
Installed size: 3.6 M
Is this ok [y/N]:
```

3.1 Installing the sos package from the command line

To use the sos utility, install the sos package.

Prerequisites

You need `root` privileges.

Procedure

Install the `sos` package.

```
sudo dnf install sos -y
```

Verification steps

Use the rpm utility to verify that the sos package is installed.

```
sudo rpm -q sos
```

3.2 Generating an sos report from the command line

Use the `sos report` command to gather an sos report from a RHEL server.

Prerequisites

- You have installed the sos package.
- You need root privileges.

Procedure

Run the sos report command and follow the on-screen instructions.

You can add the `--upload option` to transfer the sos report to Red Hat immediately after generating it.

```
sudo sos report
```

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$ sudo dnf install sos
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)          106 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)                  93 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)              113 kB/s | 4.5 kB     00:00
Dependencies resolved.
================================================================================================
 Package                 Architecture  Version            Repository                       Size
================================================================================================
Installing:
 sos                     noarch        4.7.0-1.el9        rhel-9-for-x86_64-baseos-rpms   1.1 M
Installing dependencies:
 python3-pexpect         noarch        4.8.0-7.el9        rhel-9-for-x86_64-baseos-rpms   155 k
 python3-ptyprocess      noarch        0.6.0-12.el9       rhel-9-for-x86_64-baseos-rpms    35 k

Transaction Summary
================================================================================================
Install  3 Packages

Total download size: 1.3 M
Installed size: 3.6 M
Is this ok [y/N]: y
Downloading Packages:
(1/3): python3-ptyprocess-0.6.0-12.el9.noarch.rpm                612 kB/s |  35 kB     00:00
(2/3): sos-4.7.0-1.el9.noarch.rpm                                 14 MB/s | 1.1 MB     00:00
(3/3): python3-pexpect-4.8.0-7.el9.noarch.rpm                    1.7 MB/s | 155 kB     00:00
------------------------------------------------------------------------------------------------
Total                                                             14 MB/s | 1.3 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                        1/1
  Installing       : python3-ptyprocess-0.6.0-12.el9.noarch                                 1/3
  Installing       : python3-pexpect-4.8.0-7.el9.noarch                                     2/3
  Installing       : sos-4.7.0-1.el9.noarch                                                 3/3
  Running scriptlet: sos-4.7.0-1.el9.noarch                                                 3/3
  Verifying        : python3-pexpect-4.8.0-7.el9.noarch                                     1/3
  Verifying        : python3-ptyprocess-0.6.0-12.el9.noarch                                 2/3
  Verifying        : sos-4.7.0-1.el9.noarch                                                 3/3
Installed products updated.

Installed:
  python3-pexpect-4.8.0-7.el9.noarch   python3-ptyprocess-0.6.0-12.el9.noarch   sos-4.7.0-1.el9.noarch

Complete!
[lab-user@bastion ansible]$ sudo rpm -q sos -y
rpm: -y: unknown option
[lab-user@bastion ansible]$ sudo rpm -q sos
sos-4.7.0-1.el9.noarch
[lab-user@bastion ansible]$
```

3.2 Generating an sos report from the command line

Use the `sos report` command to gather an sos report from a RHEL server.

Prerequisites

- You have installed the sos package.
- You need root privileges.

Procedure

Run the sos report command and follow the on-screen instructions.

You can add the `--upload option` to transfer the sos report to Red Hat immediately after generating it.

```
sudo sos report
```

3.2.1 (Optional) If you have already opened a Technical Support case with Red Hat,

enter the case number to embed it in the sos report file name, and it will be uploaded to that case if you specified the `--upload` option.

If you do not have a case number, leave this field blank. Entering a case number is optional and does not affect the operation of the sos utility.

3.3 Take note of the sos report file name displayed at the end of the console output.

i.e. /var/tmp/sosreport-bastion-2024-04-15-outurcc.tar.xz

3.4 Cleaning an sos report

If you work in an airgapped environment, you can still collect and send data to Red Hat or to your Own IT group

The sos utility offers a routine to obfuscate potentially sensitive data, such as

- user names

Terminal — Red Hat Demo Platform

```
[lab-user@bastion ansible]$ sudo dnf install sos
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)          106 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)                  93 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)              113 kB/s | 4.5 kB     00:00
Dependencies resolved.
================================================================================================
 Package                 Architecture  Version            Repository                       Size
================================================================================================
Installing:
 sos                     noarch        4.7.0-1.el9        rhel-9-for-x86_64-baseos-rpms   1.1 M
Installing dependencies:
 python3-pexpect         noarch        4.8.0-7.el9        rhel-9-for-x86_64-baseos-rpms   155 k
 python3-ptyprocess      noarch        0.6.0-12.el9       rhel-9-for-x86_64-baseos-rpms    35 k

Transaction Summary
================================================================================================
Install  3 Packages

Total download size: 1.3 M
Installed size: 3.6 M
Is this ok [y/N]: y
Downloading Packages:
(1/3): python3-ptyprocess-0.6.0-12.el9.noarch.rpm                612 kB/s |  35 kB     00:00
(2/3): sos-4.7.0-1.el9.noarch.rpm                                 14 MB/s | 1.1 MB     00:00
(3/3): python3-pexpect-4.8.0-7.el9.noarch.rpm                    1.7 MB/s | 155 kB     00:00
------------------------------------------------------------------------------------------------
Total                                                             14 MB/s | 1.3 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                        1/1
  Installing       : python3-ptyprocess-0.6.0-12.el9.noarch                                 1/3
  Installing       : python3-pexpect-4.8.0-7.el9.noarch                                     2/3
  Installing       : sos-4.7.0-1.el9.noarch                                                 3/3
  Running scriptlet: sos-4.7.0-1.el9.noarch                                                 3/3
  Verifying        : python3-pexpect-4.8.0-7.el9.noarch                                     1/3
  Verifying        : python3-ptyprocess-0.6.0-12.el9.noarch                                 2/3
  Verifying        : sos-4.7.0-1.el9.noarch                                                 3/3
Installed products updated.

Installed:
  python3-pexpect-4.8.0-7.el9.noarch   python3-ptyprocess-0.6.0-12.el9.noarch   sos-4.7.0-1.el9.noarch

Complete!
[lab-user@bastion ansible]$ sudo rpm -q sos -y
rpm: -y: unknown option
[lab-user@bastion ansible]$ sudo rpm -q sos
sos-4.7.0-1.el9.noarch
[lab-user@bastion ansible]$
```

3.2 Generating an sos report from the command line

Use the `sos report` command to gather an sos report from a RHEL server.

Prerequisites

- You have installed the sos package.
- You need root privileges.

Procedure

Run the sos report command and follow the on-screen instructions.

You can add the `--upload option` to transfer the sos report to Red Hat immediately after generating it.

```
sudo sos report
```

3.2.1 (Optional) If you have already opened a Technical Support case with Red Hat,

enter the case number to embed it in the sos report file name, and it will be uploaded to that case if you specified the `--upload` option.

If you do not have a case number, leave this field blank. Entering a case number is optional and does not affect the operation of the sos utility.

3.3 Take note of the sos report file name displayed at the end of the console output.

i.e. /var/tmp/sosreport-bastion-2024-04-15-outurcc.tar.xz

3.4 Cleaning an sos report

If you work in an airgapped environment, you can still collect and send data to Red Hat or to your Own IT group

The sos utility offers a routine to obfuscate potentially sensitive data, such as

- user names

Terminal — Red Hat Demo Platform

```
[lab-user@bastion ansible]$ sudo dnf install sos
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)          106 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)                  93 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)              113 kB/s | 4.5 kB     00:00
Dependencies resolved.
================================================================================================
 Package                 Architecture  Version            Repository                       Size
================================================================================================
Installing:
 sos                     noarch        4.7.0-1.el9        rhel-9-for-x86_64-baseos-rpms   1.1 M
Installing dependencies:
 python3-pexpect         noarch        4.8.0-7.el9        rhel-9-for-x86_64-baseos-rpms   155 k
 python3-ptyprocess      noarch        0.6.0-12.el9       rhel-9-for-x86_64-baseos-rpms    35 k

Transaction Summary
================================================================================================
Install  3 Packages

Total download size: 1.3 M
Installed size: 3.6 M
Is this ok [y/N]: y
Downloading Packages:
(1/3): python3-ptyprocess-0.6.0-12.el9.noarch.rpm                612 kB/s |  35 kB     00:00
(2/3): sos-4.7.0-1.el9.noarch.rpm                                 14 MB/s | 1.1 MB     00:00
(3/3): python3-pexpect-4.8.0-7.el9.noarch.rpm                    1.7 MB/s | 155 kB     00:00
------------------------------------------------------------------------------------------------
Total                                                             14 MB/s | 1.3 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                        1/1
  Installing       : python3-ptyprocess-0.6.0-12.el9.noarch                                 1/3
  Installing       : python3-pexpect-4.8.0-7.el9.noarch                                     2/3
  Installing       : sos-4.7.0-1.el9.noarch                                                 3/3
  Running scriptlet: sos-4.7.0-1.el9.noarch                                                 3/3
  Verifying        : python3-pexpect-4.8.0-7.el9.noarch                                     1/3
  Verifying        : python3-ptyprocess-0.6.0-12.el9.noarch                                 2/3
  Verifying        : sos-4.7.0-1.el9.noarch                                                 3/3
Installed products updated.

Installed:
  python3-pexpect-4.8.0-7.el9.noarch   python3-ptyprocess-0.6.0-12.el9.noarch   sos-4.7.0-1.el9.noarch

Complete!
[lab-user@bastion ansible]$ sudo rpm -q sos -y
rpm: -y: unknown option
[lab-user@bastion ansible]$ sudo rpm -q sos
sos-4.7.0-1.el9.noarch
[lab-user@bastion ansible]$
```

3.2 Generating an sos report from the command line

Use the `sos report` command to gather an sos report from a RHEL server.

Prerequisites

- You have installed the sos package.
- You need root privileges.

Procedure

Run the sos report command and follow the on-screen instructions.

You can add the `--upload option` to transfer the sos report to Red Hat immediately after generating it.

```
sudo sos report
```

3.2.1 (Optional) If you have already opened a Technical Support case with Red Hat,

enter the case number to embed it in the sos report file name, and it will be uploaded to that case if you specified the `--upload` option.

If you do not have a case number, leave this field blank. Entering a case number is optional and does not affect the operation of the sos utility.

3.3 Take note of the sos report file name displayed at the end of the console output.

i.e. /var/tmp/sosreport-bastion-2024-04-15-outurcc.tar.xz

3.4 Cleaning an sos report

If you work in an airgapped environment, you can still collect and send data to Red Hat or to your Own IT group

The sos utility offers a routine to obfuscate potentially sensitive data, such as

- user names

Terminal — Red Hat Demo Platform

```
[lab-user@bastion ansible]$ sudo dnf install sos
Updating Subscription Management repositories.
Red Hat Enterprise Linux 9 for x86_64 - Supplementary (RPMs)          106 kB/s | 3.7 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - BaseOS (RPMs)                  93 kB/s | 4.1 kB     00:00
Red Hat Enterprise Linux 9 for x86_64 - AppStream (RPMs)              113 kB/s | 4.5 kB     00:00
Dependencies resolved.
================================================================================================
 Package                 Architecture  Version            Repository                       Size
================================================================================================
Installing:
 sos                     noarch        4.7.0-1.el9        rhel-9-for-x86_64-baseos-rpms   1.1 M
Installing dependencies:
 python3-pexpect         noarch        4.8.0-7.el9        rhel-9-for-x86_64-baseos-rpms   155 k
 python3-ptyprocess      noarch        0.6.0-12.el9       rhel-9-for-x86_64-baseos-rpms    35 k

Transaction Summary
================================================================================================
Install  3 Packages

Total download size: 1.3 M
Installed size: 3.6 M
Is this ok [y/N]: y
Downloading Packages:
(1/3): python3-ptyprocess-0.6.0-12.el9.noarch.rpm                612 kB/s |  35 kB     00:00
(2/3): sos-4.7.0-1.el9.noarch.rpm                                 14 MB/s | 1.1 MB     00:00
(3/3): python3-pexpect-4.8.0-7.el9.noarch.rpm                    1.7 MB/s | 155 kB     00:00
------------------------------------------------------------------------------------------------
Total                                                             14 MB/s | 1.3 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                        1/1
  Installing       : python3-ptyprocess-0.6.0-12.el9.noarch                                 1/3
  Installing       : python3-pexpect-4.8.0-7.el9.noarch                                     2/3
  Installing       : sos-4.7.0-1.el9.noarch                                                 3/3
  Running scriptlet: sos-4.7.0-1.el9.noarch                                                 3/3
  Verifying        : python3-pexpect-4.8.0-7.el9.noarch                                     1/3
  Verifying        : python3-ptyprocess-0.6.0-12.el9.noarch                                 2/3
  Verifying        : sos-4.7.0-1.el9.noarch                                                 3/3
Installed products updated.

Installed:
  python3-pexpect-4.8.0-7.el9.noarch   python3-ptyprocess-0.6.0-12.el9.noarch   sos-4.7.0-1.el9.noarch

Complete!
[lab-user@bastion ansible]$ sudo rpm -q sos -y
rpm: -y: unknown option
[lab-user@bastion ansible]$ sudo rpm -q sos
sos-4.7.0-1.el9.noarch
[lab-user@bastion ansible]$ sudo sos report
```

3.2 Generating an sos report from the command line

Use the `sos report` command to gather an sos report from a RHEL server.

Prerequisites

- You have installed the sos package.
- You need root privileges.

Procedure

Run the sos report command and follow the on-screen instructions.

You can add the `--upload option` to transfer the sos report to Red Hat immediately after generating it.

```
sudo sos report
```

3.2.1 (Optional) If you have already opened a Technical Support case with Red Hat,

enter the case number to embed it in the sos report file name, and it will be uploaded to that case if you specified the `--upload` option.

If you do not have a case number, leave this field blank. Entering a case number is optional and does not affect the operation of the sos utility.

3.3 Take note of the sos report file name displayed at the end of the console output.

i.e. /var/tmp/sosreport-bastion-2024-04-15-outurcc.tar.xz

3.4 Cleaning an sos report

If you work in an airgapped environment, you can still collect and send data to Red Hat or to your Own IT group

The sos utility offers a routine to obfuscate potentially sensitive data, such as

- user names

Terminal — Red Hat Demo Platform

```
Transaction Summary
================================================================================
Install  3 Packages

Total download size: 1.3 M
Installed size: 3.6 M
Is this ok [y/N]: y
Downloading Packages:
(1/3): python3-ptyprocess-0.6.0-12.el9.noarch.rpm          612 kB/s |  35 kB     00:00
(2/3): sos-4.7.0-1.el9.noarch.rpm                           14 MB/s | 1.1 MB     00:00
(3/3): python3-pexpect-4.8.0-7.el9.noarch.rpm              1.7 MB/s | 155 kB     00:00
--------------------------------------------------------------------------------
Total                                                       14 MB/s | 1.3 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : python3-ptyprocess-0.6.0-12.el9.noarch                 1/3
  Installing       : python3-pexpect-4.8.0-7.el9.noarch                     2/3
  Installing       : sos-4.7.0-1.el9.noarch                                 3/3
  Running scriptlet: sos-4.7.0-1.el9.noarch                                 3/3
  Verifying        : python3-pexpect-4.8.0-7.el9.noarch                     1/3
  Verifying        : python3-ptyprocess-0.6.0-12.el9.noarch                 2/3
  Verifying        : sos-4.7.0-1.el9.noarch                                 3/3
Installed products updated.

Installed:
  python3-pexpect-4.8.0-7.el9.noarch   python3-ptyprocess-0.6.0-12.el9.noarch   sos-4.7.0-1.el9.noarch

Complete!
[lab-user@bastion ansible]$ sudo rpm -q sos -y
rpm: -y: unknown option
[lab-user@bastion ansible]$ sudo rpm -q sos
sos-4.7.0-1.el9.noarch
[lab-user@bastion ansible]$ sudo sos report

sosreport (version 4.7.0)

This command will collect diagnostic and configuration information from
this Red Hat Enterprise Linux system and installed applications.

An archive containing the collected information will be generated in
/var/tmp/sos.0motjvmo and may be provided to a Red Hat support
representative.

Any information provided to Red Hat will be treated in accordance with
the published support policies at:

        Distribution Website : https://www.redhat.com/
        Commercial Support   : https://access.redhat.com/

The generated archive may contain data considered sensitive and its
content should be reviewed by the originating organization before being
passed to any third party.

No changes will be made to system configuration.

Press ENTER to continue, or CTRL-C to quit.

Optionally, please enter the case id that you are generating this report for []:
```

3.2 Generating an sos report from the command line

Use the `sos report` command to gather an sos report from a RHEL server.

Prerequisites

- You have installed the sos package.
- You need root privileges.

Procedure

Run the sos report command and follow the on-screen instructions.

You can add the `--upload option` to transfer the sos report to Red Hat immediately after generating it.

```
sudo sos report
```

3.2.1 (Optional) If you have already opened a Technical Support case with Red Hat,

enter the case number to embed it in the sos report file name, and it will be uploaded to that case if you specified the `--upload` option.

If you do not have a case number, leave this field blank. Entering a case number is optional and does not affect the operation of the sos utility.

3.3 Take note of the sos report file name displayed at the end of the console output.

i.e. /var/tmp/sosreport-bastion-2024-04-15-outurcc.tar.xz

3.4 Cleaning an sos report

If you work in an airgapped environment, you can still collect and send data to Red Hat or to your Own IT group

The sos utility offers a routine to obfuscate potentially sensitive data, such as

- user names

Terminal — Red Hat Demo Platform

```
  Verifying        : python3-ptyprocess-0.6.0-12.el9.noarch                                  2/3
  Verifying        : sos-4.7.0-1.el9.noarch                                                  3/3
Installed products updated.

Installed:
  python3-pexpect-4.8.0-7.el9.noarch   python3-ptyprocess-0.6.0-12.el9.noarch   sos-4.7.0-1.el9.noarch

Complete!
[lab-user@bastion ansible]$ sudo rpm -q sos -y
rpm: -y: unknown option
[lab-user@bastion ansible]$ sudo rpm -q sos
sos-4.7.0-1.el9.noarch
[lab-user@bastion ansible]$ sudo sos report

sosreport (version 4.7.0)

This command will collect diagnostic and configuration information from
this Red Hat Enterprise Linux system and installed applications.

An archive containing the collected information will be generated in
/var/tmp/sos.0motjvmo and may be provided to a Red Hat support
representative.

Any information provided to Red Hat will be treated in accordance with
the published support policies at:

        Distribution Website : https://www.redhat.com/
        Commercial Support   : https://access.redhat.com/

The generated archive may contain data considered sensitive and its
content should be reviewed by the originating organization before being
passed to any third party.

No changes will be made to system configuration.

Press ENTER to continue, or CTRL-C to quit.

Optionally, please enter the case id that you are generating this report for []:

 Setting up archive ...
 Setting up plugins ...
[plugin:fwupd] skipped command 'fwupdmgr get-approved-firmware': required services missing: fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-devices --no-unreported-check': required services missing:
fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-history': required services missing: fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-remotes': required services missing: fwupd.
[plugin:fwupd] skipped command '/usr/libexec/fwupd/fwupdagent get-devices': required services missing: f
wupd.
[plugin:fwupd] skipped command '/usr/libexec/fwupd/fwupdagent get-updates': required services missing: f
wupd.
[plugin:networking] skipped command 'ip -s macsec show': required kmods missing: macsec.   Use '--allow-
system-changes' to enable collection.
[plugin:networking] skipped command 'ss -peaonmi': required kmods missing: xsk_diag.   Use '--allow-syst
em-changes' to enable collection.
[plugin:sssd] skipped command 'sssctl config-check': required services missing: sssd.
[plugin:sssd] skipped command 'sssctl domain-list': required services missing: sssd.
[plugin:systemd] skipped command 'systemd-resolve --status': required services missing: systemd-resolved
.
[plugin:systemd] skipped command 'systemd-resolve --statistics': required services missing: systemd-reso
lved.
 Running plugins. Please wait ...

  Starting 27/86 hardware        [Running: boot dnf grub2 hardware]
```

3.2 Generating an sos report from the command line

Use the `sos report` command to gather an sos report from a RHEL server.

Prerequisites

- You have installed the sos package.
- You need root privileges.

Procedure

Run the sos report command and follow the on-screen instructions.

You can add the `--upload option` to transfer the sos report to Red Hat immediately after generating it.

```
sudo sos report
```

3.2.1 (Optional) If you have already opened a Technical Support case with Red Hat,

enter the case number to embed it in the sos report file name, and it will be uploaded to that case if you specified the `--upload` option.

If you do not have a case number, leave this field blank. Entering a case number is optional and does not affect the operation of the sos utility.

3.3 Take note of the sos report file name displayed at the end of the console output.

i.e. /var/tmp/sosreport-bastion-2024-04-15-outurcc.tar.xz

3.4 Cleaning an sos report

If you work in an airgapped environment, you can still collect and send data to Red Hat or to your Own IT group

The sos utility offers a routine to obfuscate potentially sensitive data, such as

- user names

Terminal

Red Hat Demo Platform

```
sosreport (version 4.7.0)

This command will collect diagnostic and configuration information from
this Red Hat Enterprise Linux system and installed applications.

An archive containing the collected information will be generated in
/var/tmp/sos.0motjvmo and may be provided to a Red Hat support
representative.

Any information provided to Red Hat will be treated in accordance with
the published support policies at:

        Distribution Website : https://www.redhat.com/
        Commercial Support   : https://access.redhat.com/

The generated archive may contain data considered sensitive and its
content should be reviewed by the originating organization before being
passed to any third party.

No changes will be made to system configuration.

Press ENTER to continue, or CTRL-C to quit.

Optionally, please enter the case id that you are generating this report for []:

 Setting up archive ...
 Setting up plugins ...
[plugin:fwupd] skipped command 'fwupdmgr get-approved-firmware': required services missing: fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-devices --no-unreported-check': required services missing:
fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-history': required services missing: fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-remotes': required services missing: fwupd.
[plugin:fwupd] skipped command '/usr/libexec/fwupd/fwupdagent get-devices': required services missing: f
wupd.
[plugin:fwupd] skipped command '/usr/libexec/fwupd/fwupdagent get-updates': required services missing: f
wupd.
[plugin:networking] skipped command 'ip -s macsec show': required kmods missing: macsec.   Use '--allow-
system-changes' to enable collection.
[plugin:networking] skipped command 'ss -peaonmi': required kmods missing: xsk_diag.   Use '--allow-syst
em-changes' to enable collection.
[plugin:sssd] skipped command 'sssctl config-check': required services missing: sssd.
[plugin:sssd] skipped command 'sssctl domain-list': required services missing: sssd.
[plugin:systemd] skipped command 'systemd-resolve --status': required services missing: systemd-resolved
.
[plugin:systemd] skipped command 'systemd-resolve --statistics': required services missing: systemd-reso
lved.
 Running plugins. Please wait ...

  Finishing plugins             [Running: subscription_manager]                         ]
  Finished running plugins
Creating compressed archive...

Your sosreport has been generated and saved in:
        /var/tmp/sosreport-bastion-2024-05-08-prjizqb.tar.xz

 Size   9.96MiB
 Owner  root
 sha256 82a9f94174783c213ceecf419abcc6ad273499544cb6191c9d68383554d05761

Please send this file to your support representative.

[lab-user@bastion ansible]$
```

3.4 Cleaning an sos report

If you work in an airgapped environment, you can still collect and send data to Red Hat or to your Own IT group

The sos utility offers a routine to obfuscate potentially sensitive data, such as

- user names
- host names
- IP Addresses
- MAC addresses
- other user-specified keywords.

The original sos report or sos collect stays unchanged, and a new `*-obfuscated.tar.xz` file is generated and intended to be shared with a third party.

Prerequisites

- You have generated an sos report or an sos collect tarball.
- (Optional) You have a list of specific keywords beyond the user names, host names, and other data you want to obfuscate.

Procedure

- Run the sos clean command on either an sos report or sos collect tarball and follow the on-screen instructions.
  - You can add the `--keywords` option to additionally clean a given list of keywords.
  - You can add the `--usernames` option to obfuscate further sensitive user names.

```
sudo sos clean --archive-type tarball /var/tmp/sosreport*.tar.xz
```

The automatic user name cleaning will automatically run for users reported through the lastlog file for users with an UID of 1000 and above. This option is used for LDAP

Terminal

Red Hat Demo Platform

```
sosreport (version 4.7.0)

This command will collect diagnostic and configuration information from
this Red Hat Enterprise Linux system and installed applications.

An archive containing the collected information will be generated in
/var/tmp/sos.0motjvmo and may be provided to a Red Hat support
representative.

Any information provided to Red Hat will be treated in accordance with
the published support policies at:

        Distribution Website : https://www.redhat.com/
        Commercial Support   : https://access.redhat.com/

The generated archive may contain data considered sensitive and its
content should be reviewed by the originating organization before being
passed to any third party.

No changes will be made to system configuration.

Press ENTER to continue, or CTRL-C to quit.

Optionally, please enter the case id that you are generating this report for []:

 Setting up archive ...
 Setting up plugins ...
[plugin:fwupd] skipped command 'fwupdmgr get-approved-firmware': required services missing: fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-devices --no-unreported-check': required services missing:
fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-history': required services missing: fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-remotes': required services missing: fwupd.
[plugin:fwupd] skipped command '/usr/libexec/fwupd/fwupdagent get-devices': required services missing: f
wupd.
[plugin:fwupd] skipped command '/usr/libexec/fwupd/fwupdagent get-updates': required services missing: f
wupd.
[plugin:networking] skipped command 'ip -s macsec show': required kmods missing: macsec.   Use '--allow-
system-changes' to enable collection.
[plugin:networking] skipped command 'ss -peaonmi': required kmods missing: xsk_diag.   Use '--allow-syst
em-changes' to enable collection.
[plugin:sssd] skipped command 'sssctl config-check': required services missing: sssd.
[plugin:sssd] skipped command 'sssctl domain-list': required services missing: sssd.
[plugin:systemd] skipped command 'systemd-resolve --status': required services missing: systemd-resolved
.
[plugin:systemd] skipped command 'systemd-resolve --statistics': required services missing: systemd-reso
lved.
 Running plugins. Please wait ...

  Finishing plugins             [Running: subscription_manager]                        ]
  Finished running plugins
Creating compressed archive...

Your sosreport has been generated and saved in:
        /var/tmp/sosreport-bastion-2024-05-08-prjizqb.tar.xz

 Size   9.96MiB
 Owner  root
 sha256 82a9f94174783c213ceecf419abcc6ad273499544cb6191c9d68383554d05761

Please send this file to your support representative.

[lab-user@bastion ansible]$
```

3.4 Cleaning an sos report

If you work in an airgapped environment, you can still collect and send data to Red Hat or to your Own IT group

The sos utility offers a routine to obfuscate potentially sensitive data, such as

- user names
- host names
- IP Addresses
- MAC addresses
- other user-specified keywords.

The original sos report or sos collect stays unchanged, and a new `*-obfuscated.tar.xz` file is generated and intended to be shared with a third party.

Prerequisites

- You have generated an sos report or an sos collect tarball.
- (Optional) You have a list of specific keywords beyond the user names, host names, and other data you want to obfuscate.

Procedure

- Run the sos clean command on either an sos report or sos collect tarball and follow the on-screen instructions.
  - You can add the `--keywords` option to additionally clean a given list of keywords.
  - You can add the `--usernames` option to obfuscate further sensitive user names.

```
sudo sos clean --archive-type tarball /var/tmp/sosreport*.tar.xz
```

The automatic user name cleaning will automatically run for users reported through the lastlog file for users with an UID of 1000 and above. This option is used for LDAP

Terminal

Red Hat Demo Platform

```
No changes will be made to system configuration.

Press ENTER to continue, or CTRL-C to quit.

Optionally, please enter the case id that you are generating this report for []:

 Setting up archive ...
 Setting up plugins ...
[plugin:fwupd] skipped command 'fwupdmgr get-approved-firmware': required services missing: fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-devices --no-unreported-check': required services missing:
fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-history': required services missing: fwupd.
[plugin:fwupd] skipped command 'fwupdmgr get-remotes': required services missing: fwupd.
[plugin:fwupd] skipped command '/usr/libexec/fwupd/fwupdagent get-devices': required services missing: f
wupd.
[plugin:fwupd] skipped command '/usr/libexec/fwupd/fwupdagent get-updates': required services missing: f
wupd.
[plugin:networking] skipped command 'ip -s macsec show': required kmods missing: macsec.   Use '--allow-
system-changes' to enable collection.
[plugin:networking] skipped command 'ss -peaonmi': required kmods missing: xsk_diag.   Use '--allow-syst
em-changes' to enable collection.
[plugin:sssd] skipped command 'sssctl config-check': required services missing: sssd.
[plugin:sssd] skipped command 'sssctl domain-list': required services missing: sssd.
[plugin:systemd] skipped command 'systemd-resolve --status': required services missing: systemd-resolved
.
[plugin:systemd] skipped command 'systemd-resolve --statistics': required services missing: systemd-reso
lved.
 Running plugins. Please wait ...

  Finishing plugins             [Running: subscription_manager]                          ]
  Finished running plugins
Creating compressed archive...

Your sosreport has been generated and saved in:
        /var/tmp/sosreport-bastion-2024-05-08-prjizqb.tar.xz

 Size   9.96MiB
 Owner  root
 sha256 82a9f94174783c213ceecf419abcc6ad273499544cb6191c9d68383554d05761

Please send this file to your support representative.

[lab-user@bastion ansible]$ sudo sos clean --archive-type tarball /var/tmp/sosreport*.tar.xz

sos clean (version 4.7.0)

This command will attempt to obfuscate information that is generally considered
to be potentially sensitive. Such information includes IP addresses, MAC
addresses, domain names, and any user-provided keywords.

Note that this utility provides a best-effort approach to data obfuscation, but
it does not guarantee that such obfuscation provides complete coverage of all
such data in the archive, or that any obfuscation is provided to data that does
not fit the description above.

Users should review any resulting data and/or archives generated or processed by
this utility for remaining sensitive content before being passed to a third
party.


Press ENTER to continue, or CTRL-C to quit.
```

3.4 Cleaning an sos report

If you work in an airgapped environment, you can still collect and send data to Red Hat or to your Own IT group

The sos utility offers a routine to obfuscate potentially sensitive data, such as

- user names
- host names
- IP Addresses
- MAC addresses
- other user-specified keywords.

The original sos report or sos collect stays unchanged, and a new `*-obfuscated.tar.xz` file is generated and intended to be shared with a third party.

Prerequisites

- You have generated an sos report or an sos collect tarball.
- (Optional) You have a list of specific keywords beyond the user names, host names, and other data you want to obfuscate.

Procedure

- Run the sos clean command on either an sos report or sos collect tarball and follow the on-screen instructions.
  - You can add the `--keywords` option to additionally clean a given list of keywords.
  - You can add the `--usernames` option to obfuscate further sensitive user names.

```
sudo sos clean --archive-type tarball /var/tmp/sosreport*.tar.xz
```

The automatic user name cleaning will automatically run for users reported through the lastlog file for users with an UID of 1000 and above. This option is used for LDAP

Terminal

Red Hat Demo Platform

```
em-changes' to enable collection.
[plugin:sssd] skipped command 'sssctl config-check': required services missing: sssd.
[plugin:sssd] skipped command 'sssctl domain-list': required services missing: sssd.
[plugin:systemd] skipped command 'systemd-resolve --status': required services missing: systemd-resolved
.
[plugin:systemd] skipped command 'systemd-resolve --statistics': required services missing: systemd-reso
lved.
 Running plugins. Please wait ...

  Finishing plugins             [Running: subscription_manager]                          ]
  Finished running plugins
Creating compressed archive...

Your sosreport has been generated and saved in:
        /var/tmp/sosreport-bastion-2024-05-08-prjizqb.tar.xz

 Size   9.96MiB
 Owner  root
 sha256 82a9f94174783c213ceecf419abcc6ad273499544cb6191c9d68383554d05761

Please send this file to your support representative.

[lab-user@bastion ansible]$ sudo sos clean --archive-type tarball /var/tmp/sosreport*.tar.xz

sos clean (version 4.7.0)

This command will attempt to obfuscate information that is generally considered
to be potentially sensitive. Such information includes IP addresses, MAC
addresses, domain names, and any user-provided keywords.

Note that this utility provides a best-effort approach to data obfuscation, but
it does not guarantee that such obfuscation provides complete coverage of all
such data in the archive, or that any obfuscation is provided to data that does
not fit the description above.

Users should review any resulting data and/or archives generated or processed by
this utility for remaining sensitive content before being passed to a third
party.


Press ENTER to continue, or CTRL-C to quit.

Found 1 total reports to obfuscate, processing up to 4 concurrently

sosreport-bastion-2024-05-08-prjizqb :            Extracting...
sosreport-bastion-2024-05-08-prjizqb :            Beginning obfuscation...
sosreport-bastion-2024-05-08-prjizqb :            Re-compressing...
sosreport-bastion-2024-05-08-prjizqb :            Obfuscation completed [removed 886 unprocessable file
s]

Successfully obfuscated 1 report(s)

A mapping of obfuscated elements is available at
        /var/tmp/sosreport-bastion-2024-05-08-prjizqb-private_map

The obfuscated archive is available at
        /var/tmp/sosreport-bastion-2024-05-08-prjizqb-obfuscated.tar.xz

        Size    3.53MiB
        Owner   root

Please send the obfuscated archive to your support representative and keep the mapping file private
[lab-user@bastion ansible]$
```

Prerequisites

- You have generated an sos report or an sos collect tarball.
- (Optional) You have a list of specific keywords beyond the user names, host names, and other data you want to obfuscate.

Procedure

- Run the sos clean command on either an sos report or sos collect tarball and follow the on-screen instructions.
  - You can add the `--keywords` option to additionally clean a given list of keywords.
  - You can add the `--usernames` option to obfuscate further sensitive user names.

```
sudo sos clean --archive-type tarball /var/tmp/sosreport*.tar.xz
```

The automatic user name cleaning will automatically run for users reported through the lastlog file for users with an UID of 1000 and above. This option is used for LDAP users that may not appear as an actual login, but may occur in certain log files.

Verification steps

- Verify that the sos clean command created an obfuscated archive and an obfuscation mapping in the /var/tmp/ directory matching the description from the command output.
- Check the `*-private_map` file for the obfuscation mapping

It should give you a path to both files created, as an example

A mapping of obfuscated elements is available at `/var/tmp/sosreport-*-unczult-private_map`

The obfuscated archive is available at `/var/tmp/sosreport-*-obfuscated.tar.xz`

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$ ls /var/tmp/sosreport-bastion-*
/var/tmp/sosreport-bastion-2024-05-08-prjizqb-obfuscated.tar.xz
/var/tmp/sosreport-bastion-2024-05-08-prjizqb-obfuscated.tar.xz.sha256
/var/tmp/sosreport-bastion-2024-05-08-prjizqb-obfuscation.log
/var/tmp/sosreport-bastion-2024-05-08-prjizqb-private_map
/var/tmp/sosreport-bastion-2024-05-08-prjizqb.tar.xz
/var/tmp/sosreport-bastion-2024-05-08-prjizqb.tar.xz.sha256
[lab-user@bastion ansible]$
```

Creating a system baseline

SOSReport

- Capture machine baseline
- Sos utility
- Generating an SOSReport
- Cleaning an sos report

Question

Lab Issue

3.5 Capturing Ports, Protocalls, and Services with Anisble

Here's an example of an Ansible playbook that gathers facts from a RHEL 9 machine and creates a report with hardware information, software, services, and system details along with the current date, open your favorite text editor, and create a file named 'PPS_Gather_And_Report.yml'

In this example we will use Vi text editor.

Building on the past directories we have set up our ansible working directory we are going to create a new file using vi.

```
vi ~/ansible/PPS_Gather_And_Report.yml
```

Then we are going into insert mode in the file by pressing `i` then you will paste the playbook below.

```
---
- name: Gather Facts and Generate Report
  hosts: localhost
  gather_facts: true
  tasks:
    - name: Capture current date
      set_fact:
        current_date: "{{ ansible_date_time.iso8601 }}"

    - name: Gather hostname
      ansible.builtin.debug:
        msg: "Hostname: {{ ansible_hostname }}"
      register: hostname_output

    - name: Gather OS version
      ansible.builtin.debug:
        msg: "OS Version: {{ ansible_distribution }} {{ ansible_distri

    - name: Gather network information
      ansible.builtin.debug:
        msg: "Network Info: {{ ansible_all_ipv4_addresses }}"
```

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$
```

3.5 Capturing Ports, Protocalls, and Services with Anisble

Here's an example of an Ansible playbook that gathers facts from a RHEL 9 machine and creates a report with hardware information, software, services, and system details along with the current date, open your favorite text editor, and create a file named 'PPS_Gather_And_Report.yml'

In this example we will use Vi text editor.

Building on the past directories we have set up our ansible working directory we are going to create a new file using vi.

```
vi ~/ansible/PPS_Gather_And_Report.yml
```

Then we are going into insert mode in the file by pressing `i` then you will paste the playbook below.

```
---
- name: Gather Facts and Generate Report
  hosts: localhost
  gather_facts: true
  tasks:
    - name: Capture current date
      set_fact:
        current_date: "{{ ansible_date_time.iso8601 }}"

    - name: Gather hostname
      ansible.builtin.debug:
        msg: "Hostname: {{ ansible_hostname }}"
      register: hostname_output

    - name: Gather OS version
      ansible.builtin.debug:
        msg: "OS Version: {{ ansible_distribution }} {{ ansible_distri

    - name: Gather network information
      ansible.builtin.debug:
        msg: "Network Info: {{ ansible_all_ipv4_addresses }}"
```

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$ vi ~/ansible/PPS_Gather_And_Report.yml
```

3.5 Capturing Ports, Protocalls, and Services with Anisble

Here's an example of an Ansible playbook that gathers facts from a RHEL 9 machine and creates a report with hardware information, software, services, and system details along with the current date, open your favorite text editor, and create a file named 'PPS_Gather_And_Report.yml'

In this example we will use Vi text editor.

Building on the past directories we have set up our ansible working directory we are going to create a new file using vi.

```
vi ~/ansible/PPS_Gather_And_Report.yml
```

Then we are going into insert mode in the file by pressing `i` then you will paste the playbook below.

```
---
- name: Gather Facts and Generate Report
  hosts: localhost
  gather_facts: true
  tasks:
    - name: Capture current date
      set_fact:
        current_date: "{{ ansible_date_time.iso8601 }}"

    - name: Gather hostname
      ansible.builtin.debug:
        msg: "Hostname: {{ ansible_hostname }}"
      register: hostname_output

    - name: Gather OS version
      ansible.builtin.debug:
        msg: "OS Version: {{ ansible_distribution }} {{ ansible_distri

    - name: Gather network information
      ansible.builtin.debug:
        msg: "Network Info: {{ ansible_all_ipv4_addresses }}"
```

Terminal

Red Hat Demo Platform

If you have never worked with VI,
You will need to type 'i' to go into Insert mode

"~/ansible/PPS_Gather_And_Report.yml" [New] 0,0-1 All

3.5 Capturing Ports, Protocalls, and Services with Anisble

Here's an example of an Ansible playbook that gathers facts from a RHEL 9 machine and creates a report with hardware information, software, services, and system details along with the current date, open your favorite text editor, and create a file named 'PPS_Gather_And_Report.yml'

In this example we will use Vi text editor.

Building on the past directories we have set up our ansible working directory we are going to create a new file using vi.

```
vi ~/ansible/PPS_Gather_And_Report.yml
```

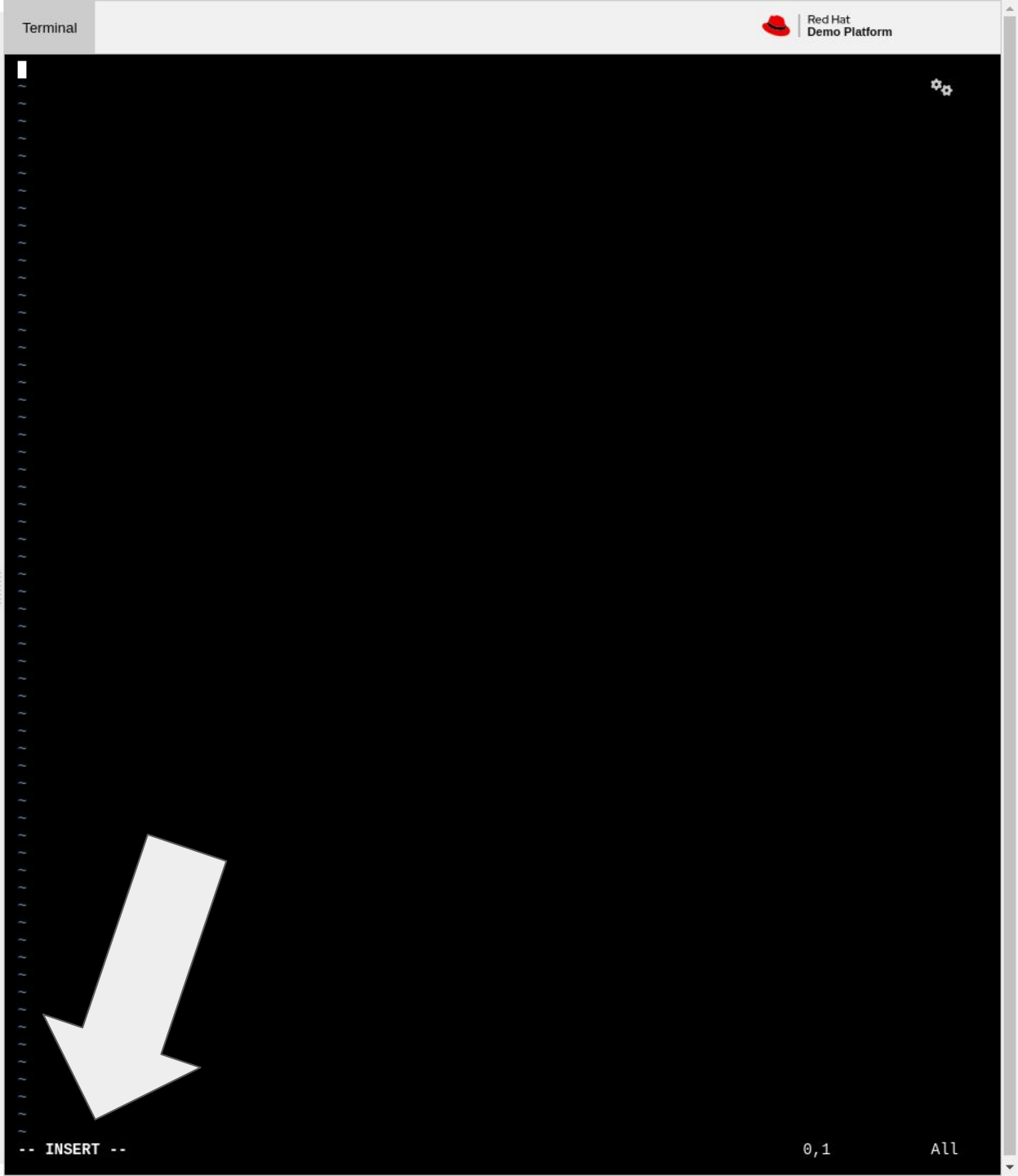
Then we are going into insert mode in the file by pressing `i` then you will paste the playbook below.

```
---
- name: Gather Facts and Generate Report
  hosts: localhost
  gather_facts: true
  tasks:
    - name: Capture current date
      set_fact:
        current_date: "{{ ansible_date_time.iso8601 }}"

    - name: Gather hostname
      ansible.builtin.debug:
        msg: "Hostname: {{ ansible_hostname }}"
      register: hostname_output

    - name: Gather OS version
      ansible.builtin.debug:
        msg: "OS Version: {{ ansible_distribution }} {{ ansible_distri

    - name: Gather network information
      ansible.builtin.debug:
        msg: "Network Info: {{ ansible_all_ipv4_addresses }}"
```



```
- name: Gather Facts and Generate Report
  hosts: localhost
  gather_facts: true
  tasks:
    - name: Capture current date
      set_fact:
        current_date: "{{ ansible_date_time.iso8601 }}"

    - name: Gather hostname
      ansible.builtin.debug:
        msg: "Hostname: {{ ansible_hostname }}"
      register: hostname_output

    - name: Gather OS version
      ansible.builtin.debug:
        msg: "OS Version: {{ ansible_distribution }} {{ ansible_distri

    - name: Gather network information
      ansible.builtin.debug:
        msg: "Network Info: {{ ansible_all_ipv4_addresses }}"

    - name: Gather open ports, protocols, and running services
      ansible.builtin.shell: "ss -tulnp"
      register: open_ports_output

    - name: Generate report
      ansible.builtin.template:
        src: report_template.j2
        dest: "{{ ansible_hostname }}_{{ current_date }}_system_report
      vars:
        hostname: "{{ ansible_hostname }}"
        os_version: "{{ ansible_distribution }} {{ ansible_distributio
        network_info: "{{ ansible_all_ipv4_addresses }}"
        open_ports_info: "{{ open_ports_output.stdout_lines }}"
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Next we are going to create a jinja2 template named `report_template.j2` also using vi.

Terminal

Red Hat Demo Platform

```
---
- name: Gather Facts and Generate Report
  hosts: localhost
  gather_facts: true
  tasks:
    - name: Capture current date
      set_fact:
        current_date: "{{ ansible_date_time.iso8601 }}"

    - name: Gather hostname
      ansible.builtin.debug:
        msg: "Hostname: {{ ansible_hostname }}"
      register: hostname_output

    - name: Gather OS version
      ansible.builtin.debug:
        msg: "OS Version: {{ ansible_distribution }} {{ ansible_distribution_version }}"

    - name: Gather network information
      ansible.builtin.debug:
        msg: "Network Info: {{ ansible_all_ipv4_addresses }}"

    - name: Gather open ports, protocols, and running services
      ansible.builtin.shell: "ss -tulnp"
      register: open_ports_output

    - name: Generate report
      ansible.builtin.template:
        src: report_template.j2
        dest: "{{ ansible_hostname }}_{{ current_date }}_system_report.txt"
      vars:
        hostname: "{{ ansible_hostname }}"
        os_version: "{{ ansible_distribution }} {{ ansible_distribution_version }}"
        network_info: "{{ ansible_all_ipv4_addresses }}"
        open_ports_info: "{{ open_ports_output.stdout_lines }}"
~
                                                            35,63         All
```

```
- name: Gather Facts and Generate Report
  hosts: localhost
  gather_facts: true
  tasks:
    - name: Capture current date
      set_fact:
        current_date: "{{ ansible_date_time.iso8601 }}"

    - name: Gather hostname
      ansible.builtin.debug:
        msg: "Hostname: {{ ansible_hostname }}"
      register: hostname_output

    - name: Gather OS version
      ansible.builtin.debug:
        msg: "OS Version: {{ ansible_distribution }} {{ ansible_distri

    - name: Gather network information
      ansible.builtin.debug:
        msg: "Network Info: {{ ansible_all_ipv4_addresses }}"

    - name: Gather open ports, protocols, and running services
      ansible.builtin.shell: "ss -tulnp"
      register: open_ports_output

    - name: Generate report
      ansible.builtin.template:
        src: report_template.j2
        dest: "{{ ansible_hostname }}_{{ current_date }}_system_report
      vars:
        hostname: "{{ ansible_hostname }}"
        os_version: "{{ ansible_distribution }} {{ ansible_distributio
        network_info: "{{ ansible_all_ipv4_addresses }}"
        open_ports_info: "{{ open_ports_output.stdout_lines }}"
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Next we are going to create a jinja2 template named `report_template.j2` also using vi.

Terminal

Red Hat Demo Platform

```
---
- name: Gather Facts and Generate Report
  hosts: localhost
  gather_facts: true
  tasks:
    - name: Capture current date
      set_fact:
        current_date: "{{ ansible_date_time.iso8601 }}"

    - name: Gather hostname
      ansible.builtin.debug:
        msg: "Hostname: {{ ansible_hostname }}"
      register: hostname_output

    - name: Gather OS version
      ansible.builtin.debug:
        msg: "OS Version: {{ ansible_distribution }} {{ ansible_distribution_version }}"

    - name: Gather network information
      ansible.builtin.debug:
        msg: "Network Info: {{ ansible_all_ipv4_addresses }}"

    - name: Gather open ports, protocols, and running services
      ansible.builtin.shell: "ss -tulnp"
      register: open_ports_output

    - name: Generate report
      ansible.builtin.template:
        src: report_template.j2
        dest: "{{ ansible_hostname }}_{{ current_date }}_system_report.txt"
      vars:
        hostname: "{{ ansible_hostname }}"
        os_version: "{{ ansible_distribution }} {{ ansible_distribution_version }}"
        network_info: "{{ ansible_all_ipv4_addresses }}"
        open_ports_info: "{{ open_ports_output.stdout_lines }}"
~
~
                                                        35,63          All
```

To save and exit vim
On your keyboard Hit the 'Esc' Key
Then press hold 'Shift' + ':'
This puts you in command mode
The type 'wq'
And hit 'Enter'

```
        hostname: "{{ ansible_hostname }}"
        os_version: "{{ ansible_distribution }} {{ ansible_distributio
        network_info: "{{ ansible_all_ipv4_addresses }}"
        open_ports_info: "{{ open_ports_output.stdout_lines }}"
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Next we are going to create a jinja2 template named `report_template.j2` also using vi.

```
vi ~/ansible/report_template.j2
```

Then we are going into insert mode in the file by pressing `i` then you will paste the jinja2 template below.

```
System Report - Generated on {{ current_date }}

Hostname: {{ hostname }}

OS Version: {{ os_version }}

Network Info: {{ network_info | join(', ') }}

Open Ports, Protocols, and Running Services:
{% for line in open_ports_info %}
{{ line }}
{% endfor %}
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Now It's time for you to run your ansible playbook, in your terminal type:

```
ansible-playbook PPS_Gather_And_Report.yml
```

This will run it on the localhost and write the report with the hostname and time dates stamp it was run for the file.

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$ vi ~/ansible/report_template.j2
```

```
        hostname: "{{ ansible_hostname }}"
        os_version: "{{ ansible_distribution }} {{ ansible_distributio
        network_info: "{{ ansible_all_ipv4_addresses }}"
        open_ports_info: "{{ open_ports_output.stdout_lines }}"
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Next we are going to create a jinja2 template named `report_template.j2` also using vi.

```
vi ~/ansible/report_template.j2
```

Then we are going into insert mode in the file by pressing `i` then you will paste the jinja2 template below.

```
System Report - Generated on {{ current_date }}

Hostname: {{ hostname }}

OS Version: {{ os_version }}

Network Info: {{ network_info | join(', ') }}

Open Ports, Protocols, and Running Services:
{% for line in open_ports_info %}
{{ line }}
{% endfor %}
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Now It's time for you to run your ansible playbook, in your terminal type:

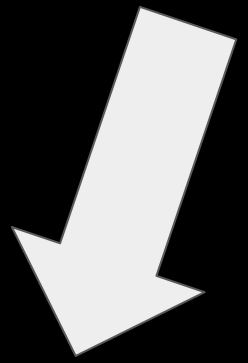
```
ansible-playbook PPS_Gather_And_Report.yml
```

This will run it on the localhost and write the report with the hostname and time dates stamp it was run for the file.

Terminal

Red Hat Demo Platform

If you have never worked with VI,
You will need to type 'i' to go into Insert mode



-- INSERT -- 0,1 All

```
        hostname: "{{ ansible_hostname }}"
        os_version: "{{ ansible_distribution }} {{ ansible_distributio
        network_info: "{{ ansible_all_ipv4_addresses }}"
        open_ports_info: "{{ open_ports_output.stdout_lines }}"
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Next we are going to create a jinja2 template named `report_template.j2` also using vi.

```
vi ~/ansible/report_template.j2
```

Then we are going into insert mode in the file by pressing `i` then you will paste the jinja2 template below.

```
System Report - Generated on {{ current_date }}

Hostname: {{ hostname }}

OS Version: {{ os_version }}

Network Info: {{ network_info | join(', ') }}

Open Ports, Protocols, and Running Services:
{% for line in open_ports_info %}
{{ line }}
{% endfor %}
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Now It's time for you to run your ansible playbook, in your terminal type:

```
ansible-playbook PPS_Gather_And_Report.yml
```

This will run it on the localhost and write the report with the hostname and time dates stamp it was run for the file.

Terminal

Red Hat Demo Platform


```
System Report - Generated on {{ current_date }}

Hostname: {{ hostname }}

OS Version: {{ os_version }}

Network Info: {{ network_info | join(', ') }}

Open Ports, Protocols, and Running Services:
{% for line in open_ports_info %}
{{ line }}
{% endfor %}
~
-- INSERT --                                    12,13         All
```

```
        hostname: "{{ ansible_hostname }}"
        os_version: "{{ ansible_distribution }} {{ ansible_distributio
        network_info: "{{ ansible_all_ipv4_addresses }}"
        open_ports_info: "{{ open_ports_output.stdout_lines }}"
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Next we are going to create a jinja2 template named `report_template.j2` also using vi.

```
vi ~/ansible/report_template.j2
```

Then we are going into insert mode in the file by pressing `i` then you will paste the jinja2 template below.

```
System Report - Generated on {{ current_date }}

Hostname: {{ hostname }}

OS Version: {{ os_version }}

Network Info: {{ network_info | join(', ') }}

Open Ports, Protocols, and Running Services:
{% for line in open_ports_info %}
{{ line }}
{% endfor %}
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Now It's time for you to run your ansible playbook, in your terminal type:

```
ansible-playbook PPS_Gather_And_Report.yml
```

This will run it on the localhost and write the report with the hostname and time dates stamp it was run for the file.

Terminal

Red Hat Demo Platform

```
System Report - Generated on {{ current_date }}

Hostname: {{ hostname }}

OS Version: {{ os_version }}

Network Info: {{ network_info | join(', ') }}

Open Ports, Protocols, and Running Services:
{% for line in open_ports_info %}
{{ line }}
{% endfor %}
```

To save and exit vim
On your keyboard Hit the 'Esc' Key
Then press hold 'Shift' + ':'
This puts you in command mode
The type 'wq'
And hit 'Enter'

:wq!

```
        hostname: "{{ ansible_hostname }}"
        os_version: "{{ ansible_distribution }} {{ ansible_distributio
        network_info: "{{ ansible_all_ipv4_addresses }}"
        open_ports_info: "{{ open_ports_output.stdout_lines }}"
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Next we are going to create a jinja2 template named `report_template.j2` also using vi.

`vi ~/ansible/report_template.j2`

Then we are going into insert mode in the file by pressing `i` then you will paste the jinja2 template below.

```
System Report - Generated on {{ current_date }}

Hostname: {{ hostname }}

OS Version: {{ os_version }}

Network Info: {{ network_info | join(', ') }}

Open Ports, Protocols, and Running Services:
{% for line in open_ports_info %}
{{ line }}
{% endfor %}
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Now It's time for you to run your ansible playbook, in your terminal type:

```
ansible-playbook PPS_Gather_And_Report.yml
```

This will run it on the localhost and write the report with the hostname and time dates stamp it was run for the file.

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$ vi ~/ansible/report_template.j2
[lab-user@bastion ansible]$
```

```
{% endfor %}
```

Next we will switch to command mode in vi by pressing the `Esc` button on your keyboard, then type `:wq` to write the file, and exit vi.

Now It's time for you to run your ansible playbook, in your terminal type:

```
ansible-playbook PPS_Gather_And_Report.yml
```

This will run it on the localhost and write the report with the hostname and time dates stamp it was run for the file.

You can look at the content of your report by running.

```
cat ~/ansible/bastion*system_report.txt
```

You'll have a report that looks something like this.

```
System Report - Generated on 2024-04-18T12:38:20Z

Hostname: bastion

OS Version: RedHat 9.3

Network Info: 192.168.0.11

Open Ports, Protocols, and Running Services:
Netid State  Recv-Q Send-Q Local Address:Port Peer Address:PortProcess
udp   UNCONN 0      0          127.0.0.1:323       0.0.0.0:*
udp   UNCONN 0      0              [::1]:323          [::]:*
tcp   LISTEN 0      128          0.0.0.0:22        0.0.0.0:*
tcp   LISTEN 0      128             [::]:22           [::]:*
tcp   LISTEN 0      4096               *:443             *:*
tcp   LISTEN 0      4096               *:9090            *:*
```


Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$ vi ~/ansible/report_template.j2
[lab-user@bastion ansible]$ ansible-playbook PPS_Gather_And_Report.yml
```

```
{% endfor %}
```

Next we will switch to command mode in vi by pressing the `Esc` button on your key-board, then type `:wq` to write the file, and exit vi.

Now It's time for you to run your ansible playbook, in your terminal type:

```
ansible-playbook PPS_Gather_And_Report.yml
```

This will run it on the localhost and write the report with the hostname and time dates stamp it was run for the file.

You can look at the content of your report by running.

```
cat ~/ansible/bastion*system_report.txt
```

You'll have a report that looks something like this.

```
System Report - Generated on 2024-04-18T12:38:20Z

Hostname: bastion

OS Version: RedHat 9.3

Network Info: 192.168.0.11

Open Ports, Protocols, and Running Services:
Netid State  Recv-Q Send-Q Local Address:Port Peer Address:PortProcess
udp   UNCONN 0      0          127.0.0.1:323       0.0.0.0:*
udp   UNCONN 0      0              [::1]:323          [::]:*
tcp   LISTEN 0      128          0.0.0.0:22        0.0.0.0:*
tcp   LISTEN 0      128             [::]:22           [::]:*
tcp   LISTEN 0      4096               *:443             *:*
tcp   LISTEN 0      4096               *:9090            *:*
```


Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$ vi ~/ansible/report_template.j2
[lab-user@bastion ansible]$ ansible-playbook PPS_Gather_And_Report.yml

PLAY [Gather Facts and Generate Report] ****************************************************

TASK [Gathering Facts] *********************************************************************
```

```
{% endfor %}
```

Next we will switch to command mode in vi by pressing the `Esc` button on your key-board, then type `:wq` to write the file, and exit vi.

Now It's time for you to run your ansible playbook, in your terminal type:

```
ansible-playbook PPS_Gather_And_Report.yml
```

This will run it on the localhost and write the report with the hostname and time dates stamp it was run for the file.

You can look at the content of your report by running.

```
cat ~/ansible/bastion*system_report.txt
```

You'll have a report that looks something like this.

```
System Report - Generated on 2024-04-18T12:38:20Z

Hostname: bastion

OS Version: RedHat 9.3

Network Info: 192.168.0.11

Open Ports, Protocols, and Running Services:
Netid State  Recv-Q Send-Q Local Address:Port Peer Address:PortProcess
udp   UNCONN 0      0          127.0.0.1:323       0.0.0.0:*
udp   UNCONN 0      0              [::1]:323          [::]:*
tcp   LISTEN 0      128          0.0.0.0:22        0.0.0.0:*
tcp   LISTEN 0      128             [::]:22           [::]:*
tcp   LISTEN 0      4096               *:443             *:*
tcp   LISTEN 0      4096               *:9090            *:*
```


Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$ vi ~/ansible/report_template.j2

[lab-user@bastion ansible]$ ansible-playbook PPS_Gather_And_Report.yml

PLAY [Gather Facts and Generate Report] ****************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [localhost]

TASK [Capture current date] ****************************************************************************
ok: [localhost]

TASK [Gather hostname] *********************************************************************************
ok: [localhost] => {
    "msg": "Hostname: bastion"
}

TASK [Gather OS version] *******************************************************************************
ok: [localhost] => {
    "msg": "OS Version: RedHat 9.3"
}

TASK [Gather network information] **********************************************************************
ok: [localhost] => {
    "msg": "Network Info: ['192.168.0.128']"
}

TASK [Gather open ports, protocols, and running services] **********************************************
changed: [localhost]

TASK [Generate report] *********************************************************************************
changed: [localhost]

PLAY RECAP *********************************************************************************************
localhost                  : ok=7    changed=2    unreachable=0    failed=0    skipped=0    rescued=0
 ignored=0

[lab-user@bastion ansible]$
```

```
{% endfor %}
```

Next we will switch to command mode in vi by pressing the `Esc` button on your key-board, then type `:wq` to write the file, and exit vi.

Now It's time for you to run your ansible playbook, in your terminal type:

```
ansible-playbook PPS_Gather_And_Report.yml
```

This will run it on the localhost and write the report with the hostname and time dates stamp it was run for the file.

You can look at the content of your report by running.

```
cat ~/ansible/bastion*system_report.txt
```

You'll have a report that looks something like this.

```
System Report - Generated on 2024-04-18T12:38:20Z

Hostname: bastion

OS Version: RedHat 9.3

Network Info: 192.168.0.11

Open Ports, Protocols, and Running Services:
Netid State  Recv-Q Send-Q Local Address:Port Peer Address:PortProcess
udp   UNCONN 0      0          127.0.0.1:323       0.0.0.0:*
udp   UNCONN 0      0              [::1]:323          [::]:*
tcp   LISTEN 0      128          0.0.0.0:22        0.0.0.0:*
tcp   LISTEN 0      128             [::]:22           [::]:*
tcp   LISTEN 0      4096               *:443             *:*
tcp   LISTEN 0      4096               *:9090            *:*
```


Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$ vi ~/ansible/report_template.j2

[lab-user@bastion ansible]$ ansible-playbook PPS_Gather_And_Report.yml

PLAY [Gather Facts and Generate Report] ****************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [localhost]

TASK [Capture current date] ****************************************************************************
ok: [localhost]

TASK [Gather hostname] *********************************************************************************
ok: [localhost] => {
    "msg": "Hostname: bastion"
}

TASK [Gather OS version] *******************************************************************************
ok: [localhost] => {
    "msg": "OS Version: RedHat 9.3"
}

TASK [Gather network information] **********************************************************************
ok: [localhost] => {
    "msg": "Network Info: ['192.168.0.128']"
}

TASK [Gather open ports, protocols, and running services] **********************************************
changed: [localhost]

TASK [Generate report] *********************************************************************************
changed: [localhost]

PLAY RECAP *********************************************************************************************
localhost                  : ok=7    changed=2    unreachable=0    failed=0    skipped=0    rescued=0
 ignored=0

[lab-user@bastion ansible]$ cat ~/ansible/bastion*system_report.txt
```

```
{% endfor %}
```

Next we will switch to command mode in vi by pressing the `Esc` button on your key-board, then type `:wq` to write the file, and exit vi.

Now It's time for you to run your ansible playbook, in your terminal type:

```
ansible-playbook PPS_Gather_And_Report.yml
```

This will run it on the localhost and write the report with the hostname and time dates stamp it was run for the file.

You can look at the content of your report by running.

```
cat ~/ansible/bastion*system_report.txt
```

You'll have a report that looks something like this.

```
System Report - Generated on 2024-04-18T12:38:20Z

Hostname: bastion

OS Version: RedHat 9.3

Network Info: 192.168.0.11

Open Ports, Protocols, and Running Services:
Netid State  Recv-Q Send-Q Local Address:Port Peer Address:PortProcess
udp   UNCONN 0      0          127.0.0.1:323       0.0.0.0:*
udp   UNCONN 0      0              [::1]:323          [::]:*
tcp   LISTEN 0      128          0.0.0.0:22        0.0.0.0:*
tcp   LISTEN 0      128             [::]:22           [::]:*
tcp   LISTEN 0      4096               *:443             *:*
tcp   LISTEN 0      4096               *:9090            *:*
```


Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$ vi ~/ansible/report_template.j2

[lab-user@bastion ansible]$ ansible-playbook PPS_Gather_And_Report.yml

PLAY [Gather Facts and Generate Report] ****************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [localhost]

TASK [Capture current date] ****************************************************************************
ok: [localhost]

TASK [Gather hostname] *********************************************************************************
ok: [localhost] => {
    "msg": "Hostname: bastion"
}

TASK [Gather OS version] *******************************************************************************
ok: [localhost] => {
    "msg": "OS Version: RedHat 9.3"
}

TASK [Gather network information] **********************************************************************
ok: [localhost] => {
    "msg": "Network Info: ['192.168.0.128']"
}

TASK [Gather open ports, protocols, and running services] **********************************************
changed: [localhost]

TASK [Generate report] *********************************************************************************
changed: [localhost]

PLAY RECAP *********************************************************************************************
localhost                  : ok=7    changed=2    unreachable=0    failed=0    skipped=0    rescued=0
 ignored=0

[lab-user@bastion ansible]$ cat ~/ansible/bastion*system_report.txt
System Report - Generated on 2024-05-08T15:59:54Z

Hostname: bastion

OS Version: RedHat 9.3

Network Info: 192.168.0.128

Open Ports, Protocols, and Running Services:
Netid State  Recv-Q Send-Q Local Address:Port Peer Address:PortProcess
udp   UNCONN 0      0          127.0.0.1:323       0.0.0.0:*
udp   UNCONN 0      0              [::1]:323          [::]:*
tcp   LISTEN 0      128          0.0.0.0:22        0.0.0.0:*
tcp   LISTEN 0      4096               *:443             *:*
tcp   LISTEN 0      128             [::]:22           [::]:*
[lab-user@bastion ansible]$
```

Creating a system baseline

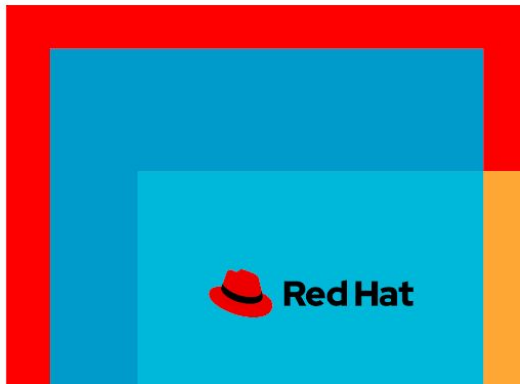
Ansible Playbook

- Capturing Ports, Protocols, and Services
- Create jinja2 Report Template

Question

Lab Issue

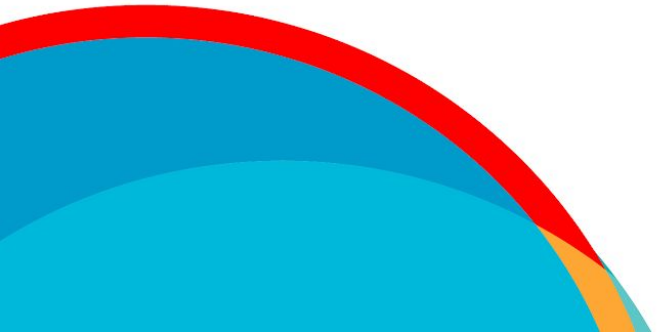
Theory, Threats, and Tools



Question for the audience

- Who Has Heard of NIST CFS?
- Who Is Working with it?
- What Version are you running?
- How many controls are you tracking?

Governance, Risk, and Compliance (aka GRC)



Red Hat

Governance

The G in GRC

Refers to the structure of rules, processes, and decision-making within an organization to ensure that objectives are achieved, risks are managed appropriately, and resources are used responsibly.

Risk Management

The R in GRC

Involves identifying, assessing, and mitigating risks that could potentially hinder the achievement of organizational goals. This includes financial risks, operational risks, legal risks, and more.

Compliance

The C in GRC

Ensures that organizations adhere to relevant laws, regulations, standards, and internal policies that are applicable to their industry and operations. Compliance efforts aim to prevent legal issues, regulatory fines, and reputational damage.

There is one more thing about GRC

What about your organization culture?

It takes a culture shift to make these changes.

Not only the adoption, but the implementation and standardization.

Risk management steps

Risk management

- Risk identification
- Risk analysis
- Risk assessment and evaluation
- Risk mitigation
- Risk monitoring

Compliance Frameworks

CYBERSECURITY FRAMEWORK

Helping organizations to better understand and improve their management of cybersecurity risk

CONNECT WITH US

BIG NEWS | The NIST CSF 2.0 has been released, along with other supplementary resources!

CSF 2.0

For industry, government, and organizations to reduce cybersecurity risks

Read the Document

Quick Start Guides

For users with specific common goals

View the Quick Start Guides

CSF 2.0 Profiles

Templates and useful resources for creating and using both CSF profiles

See the Profiles

Informative References (Mappings)

See how NIST's resources overlap and share themes

See the Mappings

LATEST UPDATES

- NIST awarded 'Ecosystem Champion' Cyber Policy Award for CSF 2.0 efforts on April 24, 2024.
- A CSF 2.0 Community Profiles NCCoE Webinar took place on April 23, 2024 and focused on opportunities to help organizations develop community profiles based on the CSF 2.0.
- On March 20, 2024, NIST hosted a webinar titled "Overview of the NIST Cybersecurity Framework 2.0 Small Business Quick Start Guide." The video recording and slides are available here.
- Aspen Institute hosted a discussion on CSF 2.0, including the Under Secretary for Standards and Technology and NIST Director

CSF 2.0 Resource Center

Download (PDF)

Quick Start Guides

Profiles

Informative References

Resources

FAQs

Translations

CSF 2.0 Tool

News and Events

Related Programs

Ways to Engage

Cybersecurity @ NIST

CSF 1.1 Archive

CONNECT WITH US

Share

Navigating NIST's CSF 2.0 Quick Start Guides

Resource and Overview Guide

Understand the basics and learn about the many available helpful CSF 2.0 resources

Download

The below targeted guides will help you with specific topics.

CSF 2.0 Organizational Profiles

Guidance for organizations, with considerations for creating and using spreadsheets called *Profiles*, to implement the CSF 2.0.

Download

CSF 2.0 Community Profiles

This guide provides considerations for creating and using Community Profiles to implement the CSF 2.0 and support the needs of organizations in communities that share common priorities.

Download

CSF 1.1 Archive +

CONNECT WITH US

Share

CSF 2.0 Organizational Profiles
Guidance for organizations, with considerations for creating and using spreadsheets called *Profiles,* to implement the CSF 2.0.

Download

CSF 2.0 Community Profiles
This guide provides considerations for creating and using Community Profiles to implement the CSF 2.0 and support the needs of organizations in communities that share common priorities.

Download

Small Business
Resources specifically tailored to small businesses with modest or no cybersecurity plans currently in place.

Download

C-SCRM
Helps organizations become smarter acquirers and suppliers of technology products and services.

Download

Tiers
Organizations can use these to apply the CSF 2.0 Tiers to Profiles to characterize the rigor of their cybersecurity risk governance and management outcomes.

Download

Enterprise Risk Management
How ERM practitioners can utilize the outcomes provided in the CSF 2.0 to improve organizational cybersecurity risk management.

Download

NIST NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
U.S. DEPARTMENT OF COMMERCE

NIST Cybersecurity Framework 2.0:
Enterprise Risk Management Quick-Start Guide

U.S. Department of Commerce
Gina M. Raimondo, Secretary
National Institute of Standards and Technology
Laurie E. Locascio, NIST Director and Under Secretary of Commerce for Standards and Technology

NIST Special Publication
NIST SP 1303 ipd (Initial Public Draft)
https://doi.org/10.6028/NIST.SP.1303.ipd
The public comment period for this draft ends May 3, 2024.
Please send your comments to cyberframework@nist.gov.
February 2024

Managing Risk & Security Framework

Contents

Lab Length Medium/Average (~10-20 mins)

Goal Scoping our lab needs for with Risk and Security Frameworks

Lab 4. Why address risk & security frameworks?

Before we get hands-on, we need to talk about Risk, Frameworks, and what your are trying to protect yourself from.

First lets go for definitions.

4.1 Governance, Risk, and Compliance (aka GRC)

It's a framework or approach that organizations use to manage and align their business activities with their objectives, manage risk effectively, and comply with regulations and standards.

Terminal

Red Hat Demo Platform

```
[lab-user@bastion ansible]$
```

Theory, Threats, and Tools

- Why address risk & security frameworks?
- Governance, Risk, and Compliance (aka GRC)
- Risk management steps
- Risk Management Examples

Question

Lab Issue

Theory, Threats, and Tools

- Compliance Frameworks
- How do you pick the right framework?
- NIST CSF Connected with RHEL Hardening
- NIST Cybersecurity Framework (CSF)

Question

Lab Issue

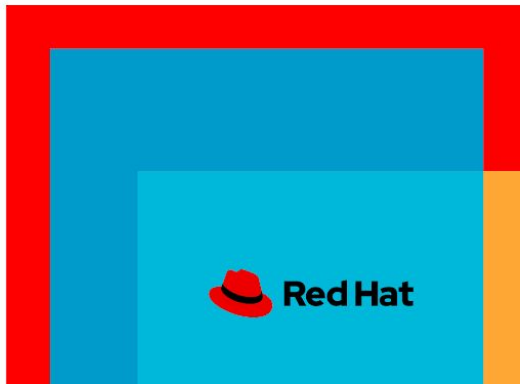
Theory, Threats, and Tools

- RHEL9 STIG
- Wrap up definition
- Configuration compliance in RHEL
- Install OpenSCAP on RHEL 9

Question

Lab Issue

Common Misconfigurations



Common Misconfigurations

- Threats seen in Red Hat Security Reports
- Weak Passwords
- Permission Issue
- Service Misconfiguration
- Neglecting Updates
- Major CVE's

Question

Lab Issue

Green Field Implementation

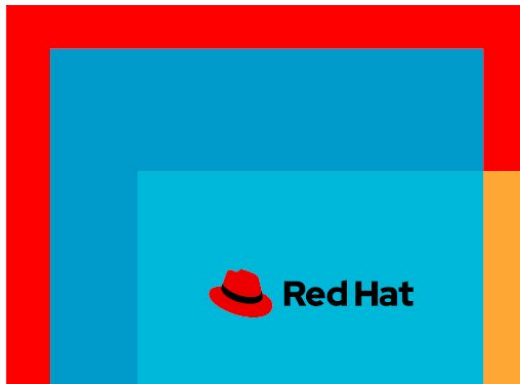
Green Field Implementation

- Where do I start?
- Download RHEL ISO
- Create a bootable USB Drive
- Advanced RHEL installs
- Customize your kickstart file
- Harden your machines
- Using Ansible for universal, stable and consistent configuration
- Additional Labs you can try later
- Security Based Labs

Question

Lab Issue

Web Console



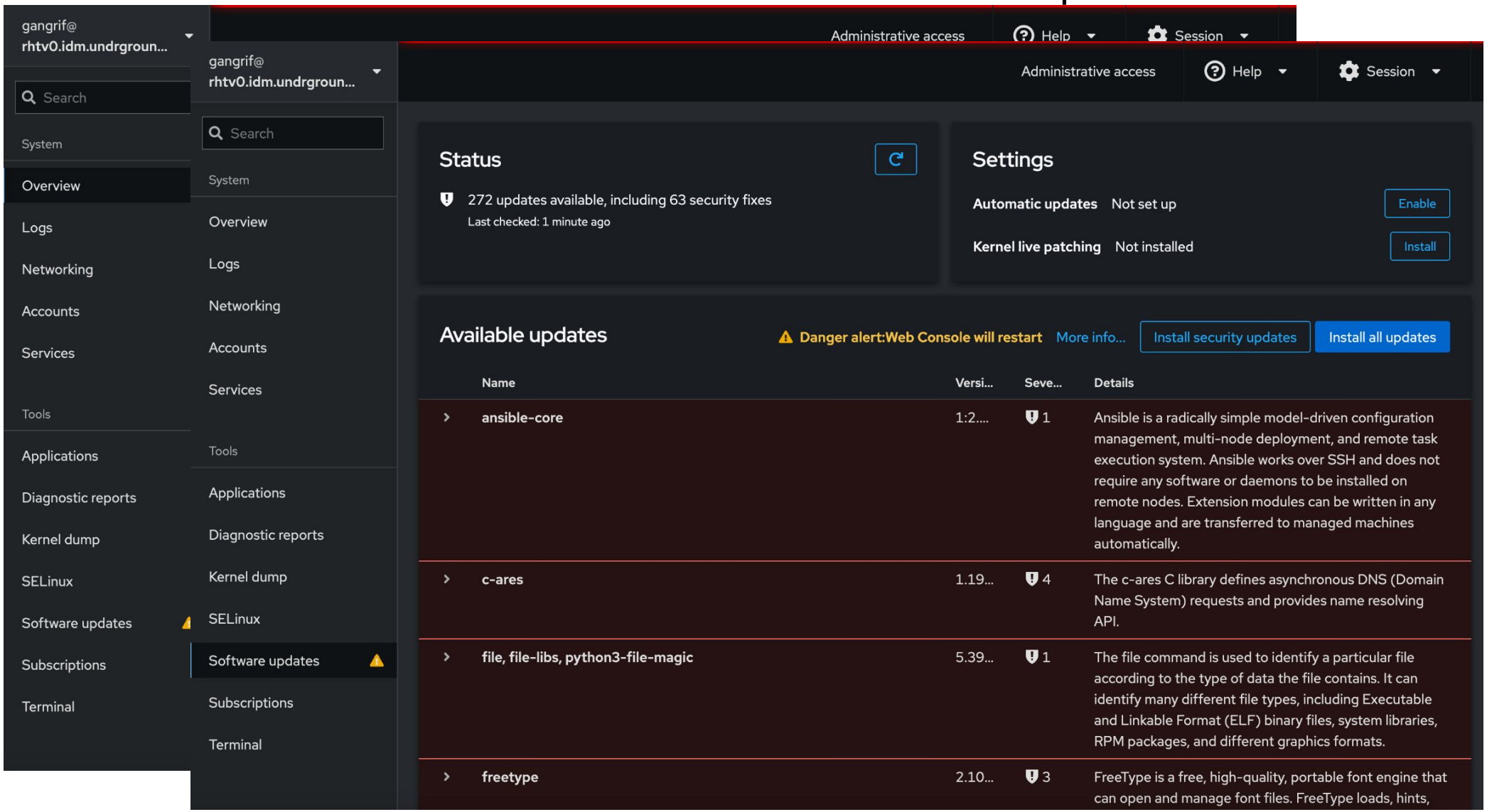
Web Console AKA Cockpit

- How to install for your OS
- Start the service
- Look at Add-ons

Question

Lab Issue

Web Console AKA Cockpit



Red Hat

And before you go,

Give us feedback using the survey in the "Red Hat Events Guide" app

Apple

⇐ if you haven't yet installed the app ⇒

Android

Red Hat

But after you're gone,

the lab guide stays available at:

Long URL: github.com/rhpds/summit_2024_RHELevant_Security_Practices_Lab_LB1964

Red Hat

Red Hat Summit

AnsibleFest

Thank you

 linkedin.com/company/red-hat

 facebook.com/redhatinc

 youtube.com/user/RedHatVideos

 twitter.com/RedHat

Red Hat