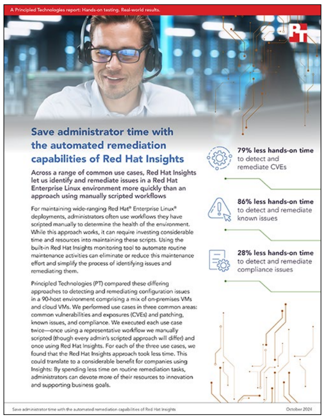

A Principled Technologies report: Hands-on testing. Real-world results.

Save administrator time with the automated remediation capabilities of Red Hat Insights

Across a range of common use cases, Red Hat Insights let us identify and remediate issues in a Red Hat Enterprise Linux environment more quickly than an approach using manually scripted workflows

For maintaining wide-ranging Red Hat® Enterprise Linux® deployments, administrators often use workflows they have scripted manually to determine the health of the environment. While this approach works, it can require investing considerable time and resources into maintaining these scripts. Using the built-in Red Hat Insights monitoring tool to automate routine maintenance activities can eliminate or reduce this maintenance effort and simplify the process of identifying issues and remediating them.

Principled Technologies (PT) compared these differing approaches to detecting and remediating configuration issues in a 90-host environment comprising a mix of on-premises VMs and cloud VMs. We performed use cases in three common areas: common vulnerabilities and exposures (CVEs) and patching, known issues, and compliance. We executed each use case twice—once using a representative workflow we manually scripted (though every admin's scripted approach will differ) and once using Red Hat Insights. For each of the three use cases, we found that the Red Hat Insights approach took less time. This could translate to a considerable benefit for companies using Insights: By spending less time on routine remediation tasks, administrators can devote more of their resources to innovation and supporting business goals.

79% less hands-on time to detect and remediate CVEs

86% less hands-on time to detect and remediate known issues

28% less hands-on time to detect and remediate compliance issues

Save administrator time with the automated remediation capabilities of Red Hat Insights October 2024

The science behind the report:

Save administrator time with the automated remediation capabilities of Red Hat Insights

This document describes what we tested, how we tested, and what we found. To learn how these facts translate into real-world benefits, read the report Save administrator time with the automated remediation capabilities of Red Hat Insights.

We concluded our hands-on testing on September 20, 2024. During testing, we determined the appropriate hardware and software configurations and applied updates as they became available. The results in this report reflect configurations that we finalized on June 20, 2024 or earlier. Unavoidably, these configurations may not represent the latest versions available when this report appears.

Our results

To learn more about how we have calculated the wins in this report, go to http://facts.pt/calculating-and-highlighting-wins. Unless we state otherwise, we have followed the rules and principles we outline in that document.

Table 1: Results of our testing.

	Red Hat Insights	Manually scripted workflow
Detect and remediate common vulnerabilities and exposures (CVEs) for 90 hosts		
Total hands-on time (hh:mm:ss)	0:01:05	0:05:14
Overall total time (hh:mm:ss)	0:02:01	1:50:37
Total number of steps	15	22
Identify and remediate known issues for 90 hosts		
Total hands-on time (hh:mm:ss)	0:00:36	0:04:28
Overall total time (hh:mm:ss)	0:02:49	0:06:36
Total number of steps	10	19
Evaluate and report on system for regulatory compliance for 85 hosts		
Total hands-on time (hh:mm:ss)	0:01:12	0:01:41
Overall total time (hh:mm:ss)	0:11:24	2:49:08
Total number of steps	21	14

System configuration information

Table 2: Detailed information on the on-premises systems we tested.

Server configuration information	**4x Dell™ VxRail™ V470F**
BIOS name and version	Dell 2.10.5
Non-default BIOS settings	N/A
Operating system name and version/build number	VMware® ESXi™ 8.0.1/21495797
Date of last OS updates/patches applied	05/07/2024
Power management policy	No Power Cap Policy Set, Balanced
Processor	
Number of processors	2
Vendor and model	Intel® Xeon® E5-2698 v4
Core count (per processor)	20
Core frequency (GHz)	2.20
Stepping	B0
Memory module(s)	
Total memory in system (GB)	512
Number of memory modules	16
Vendor and model	Hynix® Semiconductor HMA84GR7MFR4N-UH
Size (GB)	32
Type	DDR-4
Speed (MHz)	2,400
Speed running in the server (MHz)	2,400
Storage controller	
Vendor and model	Dell HBA330 Mini
Cache size (GB)	0 MB
Firmware version	16.17.00.05
Driver version	17.00.10.00-1vmw.700.1.0.15843807
Local storage	
Number of drives	9
Drive vendor and model	3x HGST HUSMM1680ASS204 6x HGST HUSMR1619ASS204
Drive size (GB)	3x 745.21 6x 1788.50
Drive information (speed, interface, type)	12Gb, SAS, SSD
Network adapter	
Vendor and model	Intel 82599, Intel Corporation Gigabit 4P X20/I350 rNDC
Number and type of ports	2x 10GbE, 4x1GbE
Driver version	1.7.1.26, 0.1.1.0

Server configuration information	4x Dell™ VxRail™ V470F
Cooling fans	
Vendor and model	Dell Embedded
Number of cooling fans	6
Power supplies	
Vendor and model	Dell PWR SPLY, 1100W, RDNT, LTON
Number of power supplies	2
Wattage of each (W)	1,100

Table 3: Detailed information on the cloud VMs we tested.

Server configuration information	5x m6i.large	5x Standard_D2s_v3 x5
Tested by	Principled Technologies	Principled Technologies
Date testing ended	06/17/2024	06/17/2024
Cloud service provider (CSP)	Amazon Web Services	Microsoft Azure
Region	us-east-1d	East US (Zone 1)
Server platform	m6i.large	Standard_D2s_v3
Operating system name and version/build number	Red Hat Enterprise Linux 9.3	Red Hat Enterprise Linux 9.0
Date of last OS updates/patches applied	06/07/2024	06/07/2024
Processor		
Number of vCPU	2	2
Vendor and model	Intel Xeon Platinum 8375C	Intel Xeon E5-2673 v4
Core count (per processor)	32	20
Core frequency (GHz)	2.90	2.30
Stepping	6	1
Hyper-threading	Yes	Yes
Turbo	Yes	Yes
Memory module(s)		
Total memory in system (GB)	8	8
OS		
Number of drives	1	1
Drive size (GB)	20	20
Drive information (speed, interface, type)	gp3, EBS, 125MBps/3000 IOPS	Standard SSD, 100 MBps/500 IOPS

How we tested

We installed and configured the latest available version of VMware vSphere® 8.0.1 build 21495797 on a cluster of four Dell VXRAIL V470F servers with Intel Xeon E5-2698 v4 processors. On each host, we created a 1.75TB VMware datastore using a locally attached NVMe® drive.

We deployed 80 VMs (20 per host) in the cluster. Each VM had one vCPU, 16 GB of memory fully reserved, and a 20GB OS drive. On each VM, we installed Red Hat Enterprise Linux 9.0. Additionally, we deployed five AWS m6i.large instances and five Azure Standard_D2s_v3 VMs, each with two vCPUs, 8 GB of memory, and a 20GB OS drive. The AWS instances used a newer version of RHEL, 9.3, due to a lack of officially supported images for older versions, while the Azure VMs used an official Red Hat Enterprise Linux (RHEL) 9.0 image.

To test the management and remediation capabilities of Red Hat Insights vs. a manual approach, we first connected our VMs to Red Hat Insights using Remote Host Configuration (rhc) and tested our use cases with the built-in features of Insights. Then, we disconnected our VMs from Insights, reverted them to their original state, and tested the same use cases with a manual approach. We recorded hands-on time, execution time, and number of steps for each scenario.

The following sections describe the steps we took to configure the test environment and run the tests. All shell scripts we mention in this section appear in full in the Scripts section following this one.

Installing vSphere version 8.0.1 build 21495797 on the VxRail hosts

1. From the VMware website, download ESXi Version 8.0.1 build 21495797.
2. Open a new browser tab, and connect to the IP address of the VxRail iDRAC.
3. Log in with the iDRAC credentials.
4. In the main screen, click Launch Virtual Console.
5. In the console menu bar, select Virtual Media, and select Virtual Storage.
6. In the Virtual Storage pop-up window, select ISO from the drop-down menu, click Open Image to browse local computers, and select the image you downloaded in step 1.
7. To mount the ISO image, click Plug In, and click OK.
8. On the console menu bar, click the Power Control, and select Power Reset.
9. After the system boots to the mounted image and the Loading ESXi installer screen appears, when prompted, press Enter to continue.
10. To Accept the EULA and Continue, press F11.
11. Select the storage device to target for installation. We selected the internal SD card. To continue, press Enter.
12. To confirm the storage target, press Enter.
13. Select the keyboard layout, and press Enter.
14. Provide a root password, and confirm the password. To continue, press Enter.
15. To install, press F11.
16. Upon completion, reboot the server by pressing Enter.
17. Complete steps 1 through 16 for each server under test.

Installing VMware vCenter Server Appliance version 8.0.1 build 22368047

Note: We deployed the VMware vCenter® appliance on one of the client hosts for management of the vSphere environment.

1. From the VMware support portal at https://my.vmware.com, download VMware vCenter version 8.0.1 build 22368047.
2. Mount the image on your local system, and browse to the vcsa-ui-installer folder. Expand the folder for your OS, and launch the installer if it doesn't automatically begin.
3. When the vCenter Server Installer wizard opens, click Install.
4. To begin installation of the new vCenter server appliance, click Next.
5. Check the box to accept the license agreement, and click Next.
6. Enter the IP address of the infrastructure server with VMware ESXi version 8.0.1 build 21495797. Provide the root password, and click Next.
7. To accept the SHA1 thumbprint of the server's certificate, click Yes.
8. Accept the VM name, and provide and confirm the root password for the VCSA. Click Next.
9. Set the size for environment you're planning to deploy. We selected Medium. Click Next.
10. Select the datastore on which to install vCenter. Accept the datastore defaults, and click Next.
11. Enter the FQDN, IP address information, and DNS servers you want to use for the vCenter server appliance. Click Next.
12. To begin deployment, click Finish.
13. When Stage 1 has completed, click Close. To confirm, click Yes.
14. Open a browser window, and connect to https://[vcenter.FQDN}:5480/.
15. On the Getting Started - vCenter Server page, click Set up.

16. Enter the root password, and click Log in.
17. Click Next.
18. Enable SSH access, and click Next.
19. To confirm the changes, click OK.
20. Type `vsphere.local` for the Single Sign-On domain name. Enter a password for the administrator account, confirm it, and click Next.
21. Click Next.
22. Click Finish.

Creating a vSphere cluster in vCenter

1. Open a browser, and enter the address of the vCenter server you deployed. For example: `https://[vcenter.FQDN]/ui`
2. In the left panel, select the vCenter server, right-click, and select New Datacenter.
3. Provide a name for the new data center, and click OK.
4. Select the data center you just created, right-click, and select New Cluster.
5. Give a name to the cluster, and click OK.
6. In the cluster configuration panel, under Add hosts, click Add.
7. Check the box for Use the same credentials for all hosts. Enter the IP address and root credentials for the first host, and enter the IP addresses of all remaining hosts. Click Next.
8. Check the box beside Hostname/IP Address to select all hosts. Click OK.
9. Click Next.
10. Click Finish.

Creating a VMware datastore

1. Open a browser, and enter the address of the first ESXi host.
2. Enter the host's credentials, and click Login.
3. Click the Storage tab.
4. Click New datastore.
5. Select Create new VMFS datastore, and click Next.
6. Name the datastore, select a device on which to create a VMFS partition, and click Next.
7. Click Next.
8. Click Finish.
9. Repeat steps 1 through 8 on the remaining hosts.

Creating the local gold test VM

1. In vCenter, select a host to create a VM.
2. Right-click, and select New Virtual Machine.
3. Click Create a new virtual machine, and click Next.
4. Enter an appropriate VM name, select Datacenter, and click Next.
5. Select an appropriate host, and click Next.
6. Choose an appropriate datastore, and click Next.
7. Select an appropriate compatibility mode (ESXi 8.0 and later), and click Next.
8. For the Guest OS Family, select Linux. For the Guest OS Version, select Red Hat Enterprise Linux (64 bit). Click Next.
9. In customize hardware, select the following:
    - 1 vCPU
    - 16 GB of memory fully reserved
    - 20GB OS hard drive
    - Choose the appropriate VM network
    - Point the DVD drive to the installation media ISO in your datastore
    - Click Next.
10. Review, and click finish.

Installing Red Hat Enterprise Linux 9.0 on the local gold test VM

1. Attach the Red Hat Enterprise Linux 9.0 ISO to the virtual machine.
2. Open the VM console, and start the VM.
3. When the system prompts you, select Install Red Hat Enterprise Linux 9.0.
4. Select your language of choice, and click Continue.
5. Click Software Selection, select Minimal Install, and click Done.
6. Click Installation Destination, ensure that the 20GB OS volume is selected, and click Done.
7. Click KDUMP, deselect Enable kdump, and click Done.
8. Click Root Password, enter your desired password twice, and click Done.
9. Click User Creation, and give the user a full name, username, and password.
10. Select Make this user administrator, and click Done.
11. Once the installation is complete, transfer the following scripts to the gold VM: hostname.sh, network.sh, and register.sh.
12. Make the scripts executable by running the following command on each:

    ```
    sudo chmod +x [scriptname].sh
    ```

13. Shut down the gold VM in preparation for cloning.

Cloning the local gold test VM

1. In vCenter, right-click the local gold test VM.
2. Click Clone → Clone to Virtual Machine.
3. Give the VM a name, select your Datacenter, and click Next.
4. Expand the Datacenter and Cluster drop-down menus, and select a host for the cloned VM. We used a round-robin system to distribute the VMs evenly across the four hosts.
5. Select the Datastore attached to the host, and click Next.
6. Click Next.
7. Click Finish.
8. Repeat steps 1 through 7 until you have 80 cloned test VMs and one cloned controller VM of the same parameters.
9. To configure the clones, log into each one and run the scripts with desired parameters:

    ```
    ./hostname.sh [number]
    ./network.sh [number]
    ./register.sh
    ```

Creating the AWS test VMs

1. In a web browser, navigate to the AWS website.
2. Click the EC2 service.
3. Click Instances.
4. Click Launch instances.
5. Name the VM, and add any additional tags as required.
6. In the Quick Start tab, under Application and OS Images (Amazon Machine Image), select Red Hat → Red Hat Enterprise Linux 9 (HVM), SSD Volume Type.
7. Under Instance type, select m6i.large.
8. Under key pair (login), click Create new key pair.
9. Give the key pair a name, and select your preferred format. We used RSA and .pem.
10. Click Create key pair.
11. Under Network settings, click Edit, and change the following:
    - Select a subnet to use across the set of VMs. We used us-east-1d.
    - Either create or select an existing security group. If you create a new group, make sure to add an Inbound Rule allowing access from your controller VM.
    - Under Configure storage, increase the volume size to 20GiB of gp3 storage.
12. Click Launch instance.
13. Repeat steps 3 through 12 four more times to create a total of five AWS VMs.

Creating the Azure test VMs

1. In a web browser, navigate to the Azure website.
2. From the home portal, click Virtual machines.
3. Click Create → Azure virtual machine.
4. Select your Subscription and Resource group, and give the VM a name.
5. Select your desired Region and Availability zone. We used East US – Zone 1.
6. Under Image, select Red Hat Enterprise Linux 9.0 (LVM) – x64 Gen2.
7. For Size, select Standard_D2s_v3.
8. Under Administrator account, select an appropriate Username.
9. For SSH public key source, select Generate a new key pair and give it a name.
10. Click Next: Disks.
11. For the OS disk type, select Standard SSD.
12. Click Next: Networking.
13. Leave the default options, and check Delete NIC when VM is deleted.
14. Click Next: Management.
15. Leave the default options, and click Next: Monitoring.
16. Leave the default options, and click Next: Advanced.
17. Leave the default options, and click Next: Tags.
18. Give the VM any desired tags, and click Review + create.
19. Review the VM settings, and click Create.
20. Click Home, and navigate to Resource groups.
21. Select your Resource group, and click your newly created Network security group.
22. Click Inbound security rules, and click Add.
23. Add a rule to allow access from your controller VM IP to systems in the security group.
24. Repeat steps 2 through 19 four more times to create a total of 5 Azure VMs, choosing "Advanced" in the Networking tab and selecting the previously created Network security group instead of creating a new group for each VM.

Configuring the cloud VMs (AWS and Azure)

1. Using WinSCP or another file transfer program of your choice, transfer the hostname.sh and register.sh scripts to the home directory of the cloud VM.
2. Use SSH to connect to the cloud VM using the appropriate RSA key and username.
3. Change the root user password:

```
sudo passwd
```

4. Make the scripts executable:

```
sudo chmod +x [scriptname].sh
```

5. Execute the hostname.sh script using appropriate parameters, and execute the register.sh script:

```
sudo ./hostname.sh [number]
sudo ./register.sh
```

6. Repeat steps 1 through 5 for all cloud VMs.

Backing up the local VMs

1. In a web browser, navigate to the vCenter appliance.
2. Log in with the appropriate credentials.
3. Shut down all cloned VMs.
4. In the left pane, select your first local VM.
5. At the top of the page, click the Snapshots tab.
6. Click Take Snapshot.
7. Give the Snapshot a name, and click Create.
8. Repeat steps 4 through 7 for the remaining cloned VMs.

Backing up the AWS VMs

1. In a web browser, navigate to the AWS website.
2. Click the EC2 service.
3. Click Instances.
4. Shut down all five instances.
5. Click the checkbox next to the first instance you wish to back up.
6. Click Actions → Image and templates → Create Image.
7. Give the image a name (and optionally, a description).
8. Tag the image if desired.
9. Click Create image.
10. Repeat steps 5 through 9 for the remaining instances.

Backing up the Azure VMs

1. In a web browser, navigate to the Azure website.
2. From the home portal, click Virtual Machines.
3. Shut down all five VMs.
4. Navigate to the Snapshots service.
5. Click Create.
6. Select the appropriate Subscription and Resource group, and name the snapshot.
7. For Snapshot type, select Full.
8. Leave Source type as Disk, and for Source disk, select your first VM's OS disk.
9. Click Next: Encryption.
10. Click the Tags tab at the top of the screen.
11. Enter any appropriate tags, and click Review + create.
12. Review the snapshot parameters, and click Create.
13. Repeat steps 4 through 12 for the remaining VMs.

Conducting the tests: Use case 1 – Common vulnerabilities and exposures patching

Testing the Red Hat Insights approach

Generating a complete list of all CVEs that affect a system

1. From the Insights dashboard, navigate to Security → Vulnerability → Reports.
2. Under Report by CVEs, click Create report.
3. Click Export report.
4. Click downloaded report to view.

Determining whether a RHBA applies to specific RHEL systems and applying appropriate patches

1. From the Insights dashboard, navigate to Security → Vulnerability → CVEs.
2. In the CVE browser, click CVE-2024-1086.
3. Click the checkbox drop-down, and choose Select all.
4. Click Remediate.
5. Name the playbook.
6. Ensure all affected systems are selected, and click Next.
7. To create the playbook, click Submit.
8. Click the newly created playbook.
9. Click Execute playbook.
10. Click Execute playbook on X systems.

Testing the manual approach

1. Open a web browser and, using your preferred security sources, hunt for potentially relevant CVEs.
2. Open a web browser, and search for the CVE in question.
3. Edit the server list to the servers the CVE may potentially affect.
4. Edit Vulnerability.sh by typing the following:

```
vi Vulnerability.sh
```

5. Change Vulnerability.sh to check for the appropriate CVE by modifying the CVE variable.
6. Save Vulnerability.sh.
7. Run ALL-Vulnerability.sh:

```
./ALL-Vulnerability.sh
```

8. Look in Vulnerability-summary.txt:

```
vi Vulnerability-summary.txt
```

9. Edit the FixVulnerability.sh script:

```
vi FixVulnerability.sh
```

10. Change FixVulnerability.sh to check for the appropriate CVE by modifying the CVE variable.
11. Save FixVulnerability.sh
12. Run the following three commands to filter to the servers that need fixing:

```
grep "Important Security notice(s)" ~/Vulnerability/Vulnerability_summary.txt | grep "local" | awk
-F: '{ print $1 }' > ~/VulnerabilityFix_servers
grep "Important Security notice(s)" ~/Vulnerability/Vulnerability_summary.txt | grep "aws" | awk -F:
'{ print $1 }' > ~/VulnerabilityFix_servers_aws
grep "Important Security notice(s)" ~/Vulnerability/Vulnerability_summary.txt | grep "azure" | awk
-F: '{ print $1 }' > ~/VulnerabilityFix_servers_azure
```

13. Run ALL-Vulnerability_fix.sh:

```
./ALL-Vulnerability_fix.sh
```

Conducting the tests: Use case 2 – Identifying and remediating known issues

Testing the Red Hat Insights approach

1. From the Insights dashboard, navigate to Operations → Advisor → Recommendations.
2. Under Recommendations, select Decreased performance occurs when KVM and VMware guests are not running with recommended tuned service configuration.
3. Click the checkbox drop-down menu, and choose Select all.
4. Click Remediate.
5. Name the playbook, and click Next.
6. Click Next.
7. Click Submit.
8. Click the newly created playbook.
9. Click Execute playbook.
10. Click Execute playbook on X systems.

Testing the manual approach

1. Open preferred web browser and look for the recommended tuned profile for your systems.
2. Edit the list of systems list to the systems you want to modify to the target tuned profile.
3. Edit tuning-check.sh:

```
vi tuning-check.sh
```

4. Change tuning-check.sh to check for the appropriate tuning.
5. Save tuning-check.sh.
6. Run ALL-tuning-check.sh:

```
./ALL-tuning-check.sh
```

7. Look in tuning-check-summary.txt:

```
vi tuning-check-summary.txt
```

8. Edit the tuning-fix.sh script:

```
vi tuning-fix.sh
```

9. Change tuning-fix.sh to change to the appropriate tuning.
10. Save tuning-fix.sh.
11. Run the following three commands to filter to the servers that need fixing:

```
grep -v "virtual-guest" ~/tuned-status/tuned_summary.txt | grep "local" | awk -F: '{ print $1 }' >
~/tuned-fix_servers_local
grep -v "virtual-guest" ~/tuned-status/tuned_summary.txt | grep "aws" | awk -F: '{ print $1 }' > ~/
tuned-fix_servers_ec2
grep -v "virtual-guest" ~/tuned-status/tuned_summary.txt | grep "azure" | awk -F: '{ print $1 }' >
~/tuned-fix_servers_azure
```

12. Run ALL-tuning-fix.sh:

```
./ALL-tuning-fix.sh
```

Conducting the tests: Use case 3 – Compliance with regulatory policies

Testing the Red Hat Insights approach

Evaluating and reporting on systems for regulatory compliance

1. From the Insights dashboard, navigate to Security → Compliance → SCAP Policies.
2. Click Create new policy.
3. Choose the operating system of the VMs under test (RHEL 9).
4. Look for and select CIS Red Hat Enterprise Linux 9 Benchmark for Level 1 – Server.
5. Click Next.
6. Type in a Compliance threshold percentage. We used 95.
7. Click Next.
8. Click the checkbox drop-down menu, and choose Select all.
9. Click Next.
10. Review the Policy Rules, and click Next.
11. Click Finish.

Remediating systems to bring them into compliance

1. From the Insights dashboard, navigate to Security → Compliance → Reports.
2. Select the report for the policy you created (CIS Red Hat Enterprise Linux 9 Benchmark for Level 1 – Server).
3. Click the checkbox drop-down menu, and choose Select all.
4. Click Remediate.
5. Name the playbook, and click Next.
6. Review systems, and click Next.
7. Click Submit.
8. Click into the newly created playbook.
9. Click Execute playbook.
10. Click Execute.

Testing the manual approach

1. Edit `cis-evaluation.sh`:

```
vi cis-evaluation.sh
```

2. Change cis-evaluation.sh to check for the CIS Level 1 benchmark.
3. Save cis-evaluation.sh.
4. Run ALL- cis-evaluation.sh:

```
./ALL-cis-evaluation.sh
```

5. Look in cis-eval-summary.txt:

```
vi cis-eval-summary.txt
```

6. Edit the cis-remediation.sh script:

```
vi cis-remediation.sh
```

7. Change cis-remediation.sh to the appropriate benchmark.
8. Save cis-remediation.sh.
9. Run the following three commands to filter to the servers that need fixing (servers that need fixing are servers that failed 10 or more checks in the previous benchmark):

```
grep "local" ~/CIS_eval/CIS_summary.txt | awk -F: '$3>9' | awk -F: '{ print $1 }' > ~/
CIS-target_local
grep "aws" ~/CIS_eval/CIS_summary.txt | awk -F: '$3>9' | awk -F: '{ print $1 }' > ~/CIS-target_aws
grep "azure" ~/CIS_eval/CIS_summary.txt | awk -F: '$3>9' | awk -F: '{ print $1 }' > ~/
CIS-target_azure
```

10. Run ALL-cis-remediation.sh:

```
./ALL-cis-remediation.sh
```

Scripts

hostname.sh

```
#!/bin/bash
SUFFIX=${1}

echo "Your new hostname is: redhat${SUFFIX}"
hostnamectl set-hostname redhat${SUFFIX}
```

network.sh

```
#!/bin/bash
SUFFIX=${1}
sudo nmcli con modify 'ens33' ifname ens33 ipv4.method manual ipv4.addresses
"10.218.100.${SUFFIX}/16" gw4 10.218.0.1
sudo nmcli con modify 'ens33' ipv4.dns 10.41.0.10
sudo nmcli con down 'ens33'
sudo nmcli con up 'ens33'
```

register.sh

```
#!/bin/bash
sed -i 's/SELINUX=.*/SELINUX=Permissive/' /etc/selinux/config

subscription-manager register --username principled --password ptredhat06 --auto-attach

dnf install openscap openscap-scanner scap-security-guide-0.1.72-1.el9_3 -y
dnf install rhc rhc-worker-playbook ansible-core -y
dnf install firewalld tuned -y

rhc connect

authselect select sssd --force

dnf install insights-client -y

insights-client --register
mkdir Vulnerability
mkdir Vulnerability_fix
mkdir tuned-status
mkdir new-tuned-status
mkdir CIS-eval
mkdir CIS-remed
```

Vulnerability.sh

```
CVE="CVE-2024-1086"
hostname=$(hostname -f)
Vulnerability_filename=Sys-Vulnerability_"${hostname}"_$(date +"%Y-%m-%d")
touch "/tmp/${Vulnerability_filename}"
sudo dnf updateinfo --cve "${CVE}" | grep "Important Security notice" >> "/
tmp/${Vulnerability_filename}"
```

ALL-Vulnerability.sh

```
CVE="CVE-2024-1086"
hostname=$(hostname -f)
Vulnerability_filename=Sys-Vulnerability_"${hostname}"_$(date +"%Y-%m-%d")
touch "/tmp/${Vulnerability_filename}"
sudo dnf updateinfo --cve "${CVE}" | grep "Important Security notice" >> "/
tmp/${Vulnerability_filename}"
[root@controller-rhel9 ~]# cat ALL-Vulnerability.sh
date
username=root
readarray -t SERVERS < ~/local_list
for HOST in ${SERVERS[@]}; do
  cat Vulnerability.sh | ssh ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp ${username}@${HOST}:/tmp/Sys-Vulnerability*$mydate ~/Vulnerability/Vulnerability_${HOST}_${mydate}
  cat ~/Vulnerability/Vulnerability_${HOST}_${mydate} | awk -v h=$HOST '{print h": local: "$0} ' >> ~/
Vulnerability/Vulnerability_summary.txt
done
username=ec2-user
readarray -t SERVERS < ~/ec2_list
for HOST in ${SERVERS[@]}; do
  cat Vulnerability.sh | ssh ${username}@${HOST} -i bannister2.pem
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2.pem ${username}@${HOST}:/tmp/Sys-Vulnerability*$mydate ~/Vulnerability/
Vulnerability_${HOST}_${mydate}
  cat ~/Vulnerability/Vulnerability_${HOST}_${mydate} | awk -v h=$HOST '{print h": aws: "$0} ' >> ~/
Vulnerability/Vulnerability_summary.txt
done
username=ptadmin
readarray -t SERVERS < ~/azure_list
for HOST in ${SERVERS[@]}; do
  cat Vulnerability.sh | ssh ${username}@${HOST} -i bannister2_azure.pem
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2_azure.pem ${username}@${HOST}:/tmp/Sys-Vulnerability*$mydate ~/Vulnerability/
Vulnerability_${HOST}_${mydate}
  cat ~/Vulnerability/Vulnerability_${HOST}_${mydate} | awk -v h=$HOST '{print h": azure: "$0} ' >> ~/
Vulnerability/Vulnerability_summary.txt
done
```

FixVulnerability.sh

```
CVE="CVE-2024-1086"
hostname=$(hostname -f)
Vulnerability_filename=Sys-Vulnerability_fix_"$(hostname)"_$(date +"%Y-%m-%d")
touch "/tmp/${Vulnerability_filename}"
sudo yum update -y --cve "${CVE}" >> "/tmp/${Vulnerability_filename}"
```

ALL-Vulnerability_fix.sh

```
username=root
readarray -t SERVERS < ~/VulnerabilityFix_servers
for HOST in ${SERVERS[@]}; do
  cat FixVulnerability.sh | ssh ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp ${username}@${HOST}:/tmp/Sys-Vulnerability_fix*$mydate ~/Vulnerability_fix/Vulnerability_
fix_${HOST}_${mydate}
  cat ~/Vulnerability_fix/Vulnerability_fix_${HOST}_${mydate} | awk -v h=$HOST '{print h": local: "$0} '
>> ~/Vulnerability_fix/Vulnerability_fix_summary.txt
done
username=ec2-user
readarray -t SERVERS < ~/VulnerabilityFix_servers_aws
for HOST in ${SERVERS[@]}; do
  cat FixVulnerability.sh | ssh -i bannister2.pem ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2.pem ${username}@${HOST}:/tmp/Sys-Vulnerability_fix*$mydate ~/Vulnerability_fix/
Vulnerability_fix_${HOST}_${mydate}
  cat ~/Vulnerability_fix/Vulnerability_fix_${HOST}_${mydate} | awk -v h=$HOST '{print h": aws: "$0} ' >>
~/Vulnerability_fix/Vulnerability_fix_summary.txt
done
username=ptadmin
readarray -t SERVERS < ~/VulnerabilityFix_servers_azure
for HOST in ${SERVERS[@]}; do
  cat FixVulnerability.sh | ssh -i bannister2_azure.pem ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2_azure.pem ${username}@${HOST}:/tmp/Sys-Vulnerability_fix*$mydate ~/Vulnerability_fix/
Vulnerability_fix_${HOST}_${mydate}
  cat ~/Vulnerability_fix/Vulnerability_fix_${HOST}_${mydate} | awk -v h=$HOST '{print h": azure: "$0} '
>> ~/Vulnerability_fix/Vulnerability_fix_summary.txt
done
```

tuning-check.sh

```
hostname=$(hostname -f)
Tuned_status=Sys-tuned_"${hostname}"_$(date +"%Y-%m-%d")
touch "/tmp/${Tuned_status}"
sudo tuned-adm active >> "/tmp/${Tuned_status}"
```

ALL-tuning-check.sh

```
username=root
readarray -t SERVERS < ~/local_list
for HOST in ${SERVERS[@]}; do
  cat tuning-check.sh | ssh ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp ${username}@${HOST}:/tmp/Sys-tuned*$mydate ~/tuned-status/tuned_${HOST}_${mydate}
  cat ~/tuned-status/tuned_${HOST}_${mydate} | awk -v h=$HOST '{print h": local: "$0} ' >> ~/tuned-
status/tuned_summary.txt
done
username=ec2-user
readarray -t SERVERS < ~/ec2_list
for HOST in ${SERVERS[@]}; do
  cat tuning-check.sh | ssh -i bannister2.pem ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2.pem ${username}@${HOST}:/tmp/Sys-tuned*$mydate ~/tuned-status/
tuned_${HOST}_${mydate}
  cat ~/tuned-status/tuned_${HOST}_${mydate} | awk -v h=$HOST '{print h": aws: "$0} ' >> ~/tuned-status/
tuned_summary.txt
done
username=ptadmin
readarray -t SERVERS < ~/azure_list
for HOST in ${SERVERS[@]}; do
  cat tuning-check.sh | ssh -i bannister2_azure.pem ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2_azure.pem ${username}@${HOST}:/tmp/Sys-tuned*$mydate ~/tuned-status/
tuned_${HOST}_${mydate}
  cat ~/tuned-status/tuned_${HOST}_${mydate} | awk -v h=$HOST '{print h": azure: "$0} ' >> ~/tuned-
status/tuned_summary.txt
done
```

tuning-set.sh

```
hostname=$(hostname -f)
Tuned_status=Sys-new-tuned_"${hostname}"_$(date +"%Y-%m-%d")
touch "/tmp/${Tuned_status}"
sudo tuned-adm profile virtual-guest
sudo tuned-adm active >> "/tmp/${Tuned_status}"
```

ALL-tuning-set.sh

```
username=root
readarray -t SERVERS < ~/ tuned-fix_servers_local
for HOST in ${SERVERS[@]}; do
  cat tuning-set.sh | ssh ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp ${username}@${HOST}:/tmp/Sys-new-tuned*$mydate ~/new-tuned-status/new-tuned_${HOST}_${mydate}
  cat ~/new-tuned-status/new-tuned_${HOST}_${mydate} | awk -v h=$HOST '{print h": local: "$0} ' >> ~/
new-tuned-status/new-tuned_summary.txt
done
username=ec2-user
readarray -t SERVERS < ~/ tuned-fix_servers_aws
for HOST in ${SERVERS[@]}; do
  cat tuning-set.sh | ssh -i bannister2.pem ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2.pem ${username}@${HOST}:/tmp/Sys-new-tuned*$mydate ~/new-tuned-status/new-
tuned_${HOST}_${mydate}
  cat ~/new-tuned-status/new-tuned_${HOST}_${mydate} | awk -v h=$HOST '{print h": aws: "$0} ' >> ~/new-
tuned-status/new-tuned_summary.txt
done
username=ptadmin
readarray -t SERVERS < ~/ tuned-fix_servers_azure
for HOST in ${SERVERS[@]}; do
  cat tuning-set.sh | ssh -i bannister2_azure.pem ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2_azure.pem ${username}@${HOST}:/tmp/Sys-new-tuned*$mydate ~/new-tuned-status/new-
tuned_${HOST}_${mydate}
  cat ~/new-tuned-status/new-tuned_${HOST}_${mydate} | awk -v h=$HOST '{print h": azure: "$0} ' >> ~/
new-tuned-status/new-tuned_summary.txt
done
```

cis-evaluation.sh

```
profile="cis_server_l1"
hostname=$(hostname -f)
CIS_filename=Sys-CIS_"${hostname}"_$(date +"%Y-%m-%d")
touch "/tmp/${CIS_filename}"
sudo oscap xccdf eval --report /tmp/report_"${hostname}"_$(date +"%Y-%m-%d").html --profile $profile /usr/
share/xml/scap/ssg/content/ssg-rhel9-ds.xml | grep "fail" | wc -l >> "/tmp/${CIS_filename}"
```

ALL-cis-evaluation.sh

```
username=root
readarray -t SERVERS < ~/local_list
for HOST in ${SERVERS[@]}; do
  cat cis-evaluation.sh | ssh ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp ${username}@${HOST}:/tmp/Sys-CIS_*$mydate ~/CIS-eval/CIS_${HOST}_${mydate}
  scp ${username}@${HOST}:/tmp/report_*${mydate}.html ~/CIS-eval/report_${HOST}_${mydate}.html
  cat ~/CIS-eval/CIS_${HOST}_${mydate} | awk -v h=$HOST '{print h": local: "$0} ' >> ~/CIS-eval/
CIS_summary.txt
done
username=ec2-user
readarray -t SERVERS < ~/ec2_list
for HOST in ${SERVERS[@]}; do
  cat cis-evaluation.sh | ssh -i bannister2.pem ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2.pem ${username}@${HOST}:/tmp/Sys-CIS_*$mydate ~/CIS-eval/CIS_${HOST}_${mydate}
  scp -i bannister2.pem ${username}@${HOST}:/tmp/report_*$mydate ~/CIS-eval/report_${HOST}_${mydate}
  cat ~/CIS-eval/CIS_${HOST}_${mydate} | awk -v h=$HOST '{print h": aws: "$0} ' >> ~/CIS-eval/
CIS_summary.txt
done
username=ptadmin
readarray -t SERVERS < ~/azure_list
```

```
for HOST in ${SERVERS[@]}; do
  cat cis-evaluation.sh | ssh -i bannister2_azure.pem ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2_azure.pem ${username}@${HOST}:/tmp/Sys-CIS_*$mydate ~/CIS-eval/CIS_${HOST}_${mydate}
  scp -i bannister2_azure.pem ${username}@${HOST}:/tmp/report_*$mydate ~/CIS-eval/
report_${HOST}_${mydate}
  cat ~/CIS-eval/CIS_${HOST}_${mydate} | awk -v h=$HOST '{print h": azure: "$0} ' >> ~/CIS-eval/
CIS_summary.txt
done
```

cis-remediation.sh

```
profile="cis_server_l1"
hostname=$(hostname -f)
CIS_filename=Sys-CIS_fix_"${hostname}"_$(date +"%Y-%m-%d")
touch "/tmp/${CIS_filename}"
sudo oscap xccdf eval --profile $profile --remediate /usr/share/xml/scap/ssg/content/ssg-rhel9-ds.xml
sudo oscap xccdf eval --report /tmp/report_"${hostname}"_$(date +"%Y-%m-%d").html --profile $profile /usr/
share/xml/scap/ssg/content/ssg-rhel9-ds.xml | grep "fail" | wc -l >> "/tmp/${CIS_filename}"
```

ALL-cis-remediation.sh

```
username=root
readarray -t SERVERS < ~/CIS-target_local
for HOST in ${SERVERS[@]}; do
  cat cis-remediation.sh | ssh ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp ${username}@${HOST}:/tmp/Sys-CIS_fix_*$mydate ~/CIS-remed/CIS_fix_${HOST}_${mydate}
  scp ${username}@${HOST}:/tmp/report_*${mydate}.html ~/CIS-remed/report_${HOST}_${mydate}.html
  cat ~/CIS-remed/CIS_fix_${HOST}_${mydate} | awk -v h=$HOST '{print h": "$0} ' >> ~/CIS-remed/CIS_
fix_summary.txt
done
username=ec2-user
readarray -t SERVERS < ~/CIS-target_aws
for HOST in ${SERVERS[@]}; do
  cat cis-remediation.sh | ssh -i bannister2.pem ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2.pem ${username}@${HOST}:/tmp/Sys-CIS_fix_*$mydate ~/CIS-remed/CIS_
fix_${HOST}_${mydate}
  scp -i bannister2.pem ${username}@${HOST}:/tmp/report_*${mydate}.html ~/CIS-remed/
report_${HOST}_${mydate}.html
  cat ~/CIS-remed/CIS_fix_${HOST}_${mydate} | awk -v h=$HOST '{print h": "$0} ' >> ~/CIS-remed/CIS_
fix_summary.txt
done
username=ptadmin
readarray -t SERVERS < ~/CIS-target_azure
for HOST in ${SERVERS[@]}; do
  cat cis-remediation.sh | ssh -i bannister2_azure.pem ${username}@${HOST}
  mydate=$(date +%Y-%m-%d)
  scp -i bannister2_azure.pem ${username}@${HOST}:/tmp/Sys-CIS_fix_*$mydate ~/CIS-remed/CIS_
fix_${HOST}_${mydate}
  scp -i bannister2_azure.pem ${username}@${HOST}:/tmp/report_*${mydate}.html ~/CIS-remed/
report_${HOST}_${mydate}.html
  cat ~/CIS-remed/CIS_fix_${HOST}_${mydate} | awk -v h=$HOST '{print h": "$0} ' >> ~/CIS-remed/CIS_
fix_summary.txt
done
```

Read the report at https://facts.pt/HG7bfPw ▶

This project was commissioned by Red Hat.

Principled Technologies®

Facts matter.®

DISCLAIMER OF WARRANTIES; LIMITATION OF LIABILITY:
Principled Technologies, Inc. has made reasonable efforts to ensure the accuracy and validity of its testing, however, Principled Technologies, Inc. specifically disclaims any warranty, expressed or implied, relating to the test results and analysis, their accuracy, completeness or quality, including any implied warranty of fitness for any particular purpose. All persons or entities relying on the results of any testing do so at their own risk, and agree that Principled Technologies, Inc., its employees and its subcontractors shall have no liability whatsoever from any claim of loss or damage on account of any alleged error or defect in any testing procedure or result.

In no event shall Principled Technologies, Inc. be liable for indirect, special, incidental, or consequential damages in connection with its testing, even if advised of the possibility of such damages. In no event shall Principled Technologies, Inc.'s liability, including for direct damages, exceed the amounts paid in connection with Principled Technologies, Inc.'s testing. Customer's sole and exclusive remedies are as set forth herein.