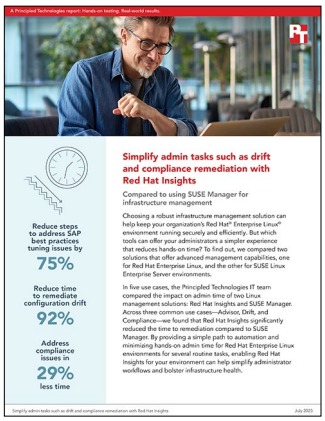


The science behind the report:

Simplify admin tasks such as drift and compliance remediation with 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 Simplify admin tasks such as drift and compliance remediation with Red Hat Insights.

We concluded our hands-on testing on March 16, 2023. 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 July 20, 2022 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		SUSE		Percentage win	
Use case	Time (h:mm:ss)	Steps	Time (h:mm:ss)	Steps	Time	Steps
Vulnerability (average of 3 trials)	0:01:29	18.3	0:01:12	19.0	-22.5%	3.5%
Bugfix (average of 3 trials)	0:01:08	15.7	0:01:07	16.0	-0.4%	2.0%
SAP (average of 2 trials)	0:01:15	15.3	0:01:11	16.0	-7.0%	4.1%
SAP (Best practices)	0:01:35	16.0	0:07:43	64.0	79.4%	75.0%
Drift	0:04:46	123	1:01:46	292	92.2%	57.8%
Compliance	0:06:39	83	0:09:28	99	29.7%	16.1%

System configuration information

Table 2: Detailed information on the systems we tested.

System configuration information	2x Dell PowerEdge R740
BIOS name and version	Dell Inc. 2.12.2
Operating system name and version/build number	ESXi-7.0U2a-17867351
Date of last OS updates/patches applied	12/13/21
Power management policy	Balanced
Processor	
Number of processors	2
Vendor and model	Intel® Xeon® Silver 4210R
Core count (per processor)	10
Core frequency (GHz)	2.40
Stepping	R1
Memory module(s)	
Total memory in system (GB)	384
Number of memory modules	12
Vendor and model	Samsung M393A4K40BB1-CRC
Size (GB)	32
Type	DDR4
Speed (MHz)	2,400
Speed running in the server (MHz)	2,400
Storage controller	
Vendor and model	Dell PERC H330
Cache size (GB)	0
Firmware version	25.5.9.0001
Driver version	7.716.03.00
Local storage	
Number of drives	6
Drive vendor and model	2x Pliant LB206M, 2x HGST HUSMM3280ASS205, 2x ATA VK003840GWTTD
Drive size (GB)	2x 185.75GB, 2x 744.63, 2x 3576.38
Drive information (speed, interface, type)	2x SSD SAS, 2x SSD SAS, 2x SSD SATA

System configuration information	2x Dell PowerEdge R740
Network adapter	
Vendor and model	Broadcom® Gigabit Ethernet BCM5720
Number and type of ports	4 x 1GbE
Driver version	ntg 14e4:165f 1028:1f5b
Cooling fans	
Vendor and model	Dell Standard
Number of cooling fans	6
Power supplies	
Vendor and model	Dell 0061XTA01
Number of power supplies	2
Wattage of each (W)	750

How we tested

According to Red Hat SAP install recommendations, we installed Red Hat Enterprise Linux® 8.0 on three VMs each with 4vCPUs, 24 GB of memory, and 16GB,100Gb,and 2x 250GB virtual hard disks connected to the VM network, which had a 1000Mbps physical connection to the network and internet.

According to SUSE SAP install recommendations, we installed SUSE Linux Enterprise Server 15 SP4 on three VMs each with 4 vCPUs, 24 GB of memory, and 100GB and 4x 50GB virtual hard disks connected to the VM network, which haf a 1000Mbps physical connection to the network and internet.

We spread the VMs evenly between the two ESXi™ hosts and used cloning to expedite VM creation.

SAP installation

Installing SAP S/4HANA with highly available HANA databases on Red Hat Enterprise Linux

The variables you need to complete testing are in the following files:

inventory/group_vars: hana.yml s4hana.yml sap_hosts.yml
inventory/host_vars: toad1.yml toad2.yml toad3.yml

File: inventory/hosts

```
[sap_hosts:children]
hana
s4hana
[hana]
toad2
toad3
[s4hana]
toad1
File: inventory/group_vars/hana/yml
storage_volumes:
  - name: shared
    type: disk
    disks: ["sdb"]
    mount_point: "/hana/shared"
    fs_label: shared
  - name: data
    type: disk
    disks: ["sdc"]
    mount_point: "/hana/data/RHE"
    fs_label: data
  - name: log
    type: disk
    disks: ["sdd"]
    mount_point: "/hana/log/RHE"
    fs_label: log
sap_hana_preconfigure_reboot_ok: no
sap_hana_preconfigure_fail_if_reboot_required: no
sap_hana_deployment_bundle_path: /opt/SAPHANADEPLOYMENT
sap_hana_deployment_bundle_sar_file_name: IMDB_SERVER20_059_4-80002031.SAR
sap_hana_deployment_sapcar_path: /opt/SAPHANADEPLOYMENT
sap_hana_deployment_sapcar_file_name: SAPCAR_1115-70006178.EXE
sap_hana_deployment_hana_sid: RHE
sap_hana_deployment_hana_instance_number: "00"
sap_hana_deployment_hana_env_type: development
sap_hana_deployment_hana_mem_restrict: "n"
sap_hana_deployment_apply_license: false
sap_hana_deployment_root_password: "Password123"
sap_hana_deployment_ase_user_password: "Password123"
sap_hana_deployment_use_master_password: "y"
sap_hana_deployment_common_master_password: "Password123"
sap_hana_hsr_hana_sid: RHE
sap_hana_hsr_hana_instance_number: "00"
sap_hana_hsr_hana_primary_hostname: toad2
sap_hana_hsr_hana_db_system_password: "Password123"
sap_hana_ha_pacemaker_hana_sid: RHE
sap_hana_ha_pacemaker_hana_instance_number: "00"
sap_hana_ha_pacemaker_vip: 10.215.1.74
sap_hana_ha_pacemaker_hacluster_password: "Password123"
sap_hana_ha_pacemaker_node1_fqdn: toad2.binion2.lab
sap_hana_ha_pacemaker_node2_fqdn: toad3.binion2.lab
sap_hana_ha_pacemaker_node1_ip: 10.215.215.12
sap_hana_ha_pacemaker_node2_ip: 10.215.215.13
```

File: inventory/group_vars/s4hana.yml

```
#sap_s4hana_deployment_product_id: NW_ABAP_OneHost:S4HANA2020.CORE.HDB.ABAP
sap_s4hana_deployment_product_id: NW_ABAP_OneHost:S4HANA1809.CORE.HDB.ABAP
sap_s4hana_deployment_sapcar_path: /opt/SAPHANADEPLOYMENT
sap_s4hana_deployment_sapcar_file_name: SAPCAR_1115-70006178.EXE
sap_s4hana_deployment_swpm_path: /opt/SAPS4HANADEPLOYMENT
#sap_s4hana_deployment_swpm_sar_file_name: SWPM20SP10_5-80003424.SAR
sap_s4hana_deployment_swpm_sar_file_name: SWPM20SP12_4-80003424.SAR
sap_s4hana_deployment_sid: RHE
sap_s4hana_deployment_ascs_instance_nr: "00"
sap_s4hana_deployment_db_host: 10.215.1.74
sap_s4hana_deployment_db_sid: RHE
sap_s4hana_deployment_hana_instance_nr: "00"
sap_s4hana_deployment_db_sidadm_password: "Password123"
sap_s4hana_deployment_db_schema_abap_password: "Password123"
sap_s4hana_deployment_master_password: "Password123"
sap_s4hana_deployment_hana_systemdb_password: "Password123"
sap_s4hana_deployment_hana_system_password: "Password123"
sap_s4hana_deployment_parallel_jobs_nr: 30
sap_s4hana_deployment_igs_path: /opt/SAPS4HANADEPLOYMENT
sap_s4hana_deployment_igs_file_name: igsexe_1-70005417.sar
sap_s4hana_deployment_igs_helper_path: /opt/SAPS4HANADEPLOYMENT
sap_s4hana_deployment_igs_helper_file_name: igshelper_17-10010245.sar
sap_s4hana_deployment_kernel_dependent_path: /opt/SAPS4HANADEPLOYMENT
#sap_s4hana_deployment_kernel_dependent_file_name: SAPEXEDB_200-70005282.SAR
sap_s4hana_deployment_kernel_dependent_file_name: SAPEXE_500-80004393.SAR
sap_s4hana_deployment_kernel_independent_path: /opt/SAPS4HANADEPLOYMENT
#sap_s4hana_deployment_kernel_independent_file_name: SAPEXE_200-70005283.SAR
sap_s4hana_deployment_kernel_independent_file_name: SAPEXEDB_500-80004392.SAR
sap_s4hana_deployment_sapadm_password: "Password123"
sap_s4hana_deployment_sap_sidadm_password: "Password123"
sap_s4hana_deployment_ascs_instance_hostname: toad1
sap_s4hana_deployment_db_schema_java_password: "Password123"
sap_s4hana_deployment_set_fqdn: "false"
sap_s4hana_deployment_software_path: /opt/SAPS4HANADEPLOYMENT

# sap_swpm
# Product ID for New Installation
sap_swpm_product_catalog_id: "NW_ABAP_OneHost:S4HANA2021.FNDN.HDB.ABAP"
# Do not touch /etc/hosts - routine is buggy
sap_swpm_update_etchosts: false
# Software
sap_swpm_software_path: "/opt/f"
sap_swpm_sapcar_path: "{{ sap_swpm_software_path }}"
sap_swpm_swpm_path: "{{ sap_swpm_software_path }}"
# NW Passwords
sap_swpm_master_password: "Password123"
sap_swpm_ddic_000_password: "{{ sap_swpm_master_password }}"
# HDB Passwords
sap_swpm_db_system_password: "{{ sap_swpm_master_password }}"
sap_swpm_db_systemdb_password: "{{ sap_swpm_master_password }}"
sap_swpm_db_schema_abap_password: "{{ sap_swpm_master_password }}"
sap_swpm_db_sidadm_password: "{{ sap_swpm_master_password }}"
# Defined as Default Value
# sap_swpm_db_schema_abap: "SAPHANADB"
# NW Instance Parameters
sap_swpm_sid: RHE
sap_swpm_pas_instance_nr: "01"
sap_swpm_ascs_instance_nr: "02"
sap_swpm_ascs_instance_hostname: "{{ ansible_hostname }}"
sap_swpm_fqdn: "{{ ansible_domain }}"
# HDB Instance Parameters
# For dual host installation, change the db_host to appropriate value
sap_swpm_db_host: "10.215.1.74"
sap_swpm_db_sid: RHE
sap_swpm_db_instance_nr: "00"
```

File: inventory/group_vars/sap_hosts.yml

```
sap_rhsm_username: "principled"
sap_rhsm_password: "ptredhat06"
sap_rhsm_force_register: "no"
sap_rhsm_register_insights: false
sap_preconfigure_modify_etc_hosts: false
sap_preconfigure_fail_if_reboot_required: no
sap_hostagent_installation_type: "sar"
sap_hostagent_sar_local_path: "/opt/SAPHOSTAGENT"
sap_hostagent_sar_file_name: "SAPHOSTAGENT56_56-80004822.SAR"
sap_hostagent_sapcar_local_path: "/opt/SAPHOSTAGENT"
sap_hostagent_sapcar_file_name: "SAPCAR_1115-70006178.EXE"
sap_hostagent_agent_tmp_directory: /tmp/a
sap_hostagent_clean_tmp_directory: true
```

File: inventory/host_vars/toad2.yml

```
sap_hana_hsr_role: "primary"
sap_hana_hsr_alias: "DC1"
```

File: inventory/host_vars/toad3.yml

```
sap_hana_hsr_role: "secondary"
sap_hana_hsr_alias: "DC2"
```

1. Grab roles:

```
cd /root/sap-deployment
ansible-galaxy install -r playbooks/requirements.yml -p roles
# Add the working role for pacemaker from https://github.com/redhat-sap/sap-hana-ha-pacemaker/
archive/refs/tags/v1.0.0.tar.gz
tar -xf v1.0.0.tar.gz
mv sap-hana-ha-pacemaker-1.0.0 roles/
```

2. Complete prerequisites (OS, SAP packages, storage, SAP host installation—these are the steps 1 through 4 from the email):

```
ansible-playbook playbooks/sap_pre.yml |& tee logs/sap_pre01.txt
```

3. Complete HANA playbook:

```
ansible-playbook playbooks/hana.yml |& tee logs/hana01.txt
```

4. Complete the cluster:

```
ansible-playbook playbooks/hsr.yml |& tee logs/hsr01.txt
ansible-playbook playbooks/ha.yml |& tee logs/ha01.txtssh toad2
pcs stonith create vmfence fence_vmware_soap \
  pcmk_host_map="toad2.binion2.lab:Toad2;toad3.binion2.lab:Toad3" ip=10.215.1.68
  username=root password='Password1!' ssl_insecure=1
```

5. Check fencing status:

```
pcs stonith config vmfence
```

6. Complete S/4Hana:

```
ansible-playbook playbooks/swpm_s4hana.yml |& tee logs/swpm_s4hana01.txt
```

Installing SUSE Manager

1. Download SUSE Manager 4.3 from https://www.suse.com/download/suse-manager/.
2. Right-click Virtual Machines. From the menu, select Create/Register VM. Select Create new virtual machine, and click Next.
3. At the Select a name and guest OS screen, under Name type `SUSE Manager`. To the right of Guest OS family, from the drop-down menu, select Linux. To the right of Guest OS version, select SUSE Linux Enterprise 15 (64-bit).
4. At the Select storage screen, make sure the appropriate storage is highlighted, and click Next.
5. At the Customize settings screen, use the drop-down menus to specify the following, and click Next.
    - CPU: 4
    - Memory: 32 GB
    - Hard disk: 200
6. Expand the drop-down menu to the left of CD/DVD Drive 1. From the drop-down, select Datastore ISO file. Click Browse, and select the first downloaded ISO.
7. Click Add other device→CD/DVD Drive. From the CD/DVD drive drop-down, select Datastore ISO file. Click Browse.
8. Click Finish.
9. Power on the VM.
10. At the Language, Keyboard and Product Selection screen, select SUSE Manager Server 4.3.
11. Click Next.
12. At the SUSE Manager Server 4.3 License Agreement, check theI Agree to the License Terms checkbox, and click Next.
13. On the Registration screen, enter your email address. Under Registration Code, enter your registration code from https://scc.suse.com/organizations/773871/subscriptions. Click Next.
14. At the The registration server offers update repositories dialogue box, click Yes.
15. At the Extension and Module Select Screen, leave the defaults selected, and click Next. (It should have Base\system Module 15 SP4 x86_64, SUSE Manager Server Module 4.3 x86_64, Server Applications Module 15 SP4 x86_64, and Web and Scripting Module 15 SP4 x86_64.)
16. On the Add On Product screen, leave I would like to install an additional Add on Product unchecked, and click Next.
17. At the System Role Screen, click SUSE Manager Server (single disk), and click Next.
18. At the Suggested Partitioning screen, leave the defaults selected, and click Next.
19. At the Clock and Time Zone screen, select the appropriate region and time zone from the drop-down menu, and click Next.
20. At the Create New User screen, enter the Fullname, Username, Password, and Confirm Password. Click Use this password for system administrator, and click Next.
21. At the Installation Settings Screen, click Network Configuration, and edit any necessary details. GIve system the hostname SUSEManager, and click Next.
22. At the Installation Setting screen, click Install.
23. At the Confirm Installation dialogue box, click Install.
24. Reboot the server:

```
shutdown -r now
```

25. Add an entry for your SUSEManager system and each of your client systems to your DNS.
26. Point your SUSEManager to your DNS server:

```
vi /etc/sysconfig/network/config
NETCONFIG_DNS_STATIC_SERVERS="your IP"
netconfig update -f
```

27. Reboot the server:

```
shutdown -r now
```

28. Log into the first node using SSH. In this document, our first node is tide0, the second tide1, and the third tide2.
    a. Generate a key:

```
ssh-keygen
```

    b. Press Enter three times.
    c. Copy the key:

```
ssh-copy-id tide0
```

    d. Enter the password.

29. Repeat step 28 for tide1 and tide2.
30. Log into tide0 via ssh:

```
vi /etc/hosts
```

31. Add the entry for SUSEManager.
32. Repeat steps 30 and 31 for tide1 and tide2.
33. To complete SUSEManager setup, start YAST2:

```
yast2 susemanager_setup
```

    a. At the Setup menu screen, check Set up SUSE Manager from scratch, and press Enter.
    b. On the screen, accept the default email address for SUSE Manager Administrator. Press tab four times to highlight [Next], and press Enter.
    c. At the Certificate Setup screen, enter your company information, add a password, tab to highlight [Next], and press Enter.
    d. At the Database Settings screen, leave the default for Database User, and enter the Database Password. Tab to highlight [Next], and press Enter.
    e. At the Write Settings screen, make sure[Yes] is highlighted, and press Enter.
    f. At the Setup is completed screen, press tab until [Next] is highlighted, and press Enter.
    g. At the Setup Completed screen, press Enter.

34. Use a web browser to open https://<your machine> to create the SUSE Manager administrator account.

    a. At the Create SUSE Manager Administrator screen, fill out your organization information, and click Create Organization.
    b. Click Admin.
    c. At the Setup Wizard, click Organization Credentials.
    d. Above Add a new Credential, click the +.
    e. On another web browser tab, open https://scc.suse.com, and log into the SUSE customer center.
    f. Click Users→Organization Credentials.
    g. Highlight and copy your username, and paste into the Username section of the Edit credentials dialogue box in SUSE Manager.
    h. In the SUSE Customer Center, click the eyeball, and highlight and copy the password, pasting it into the Password section of the Edit credentials dialogue box.
    i. Click Save.

35. Click Products, and if nothing is listed, right-click Refresh.
36. Next to Filter by, click and type `15 SP4`
37. Select the following products for x86_64 architecture:

    a. SUSE Linux Enterprise Server 15 SP4 x86_64
    b. SUSE Linux Enterprise Server for SAP Applications 15 SP4
    c. SUSE Manager Server 4.3

38. To start product synchronization, click Add Products. Wait for the circles next to each selected channel to clear, which may require refreshing.
39. Click Systems →Activation Keys.

    a. Click + Create Key.
    b. Next to Description, type `Key1 SAP`
    c. Next to Key, type `SLES15-SP4`
    d. From the Base Channel drop-down menu, select SLE-Product-SLES_SAP15-SP4-Pool for x86_64.
    e. Under child channels, click the Include Recommended slider to select all child channels. SLE-Manager-Tools15-Pool for x86_6 SAP SP4 recommended:

```
SLE-Manager-Tools15-Updates for x86_64 SAP SP4  recommended
SLE-Module-Basesystem15-SP4-Pool for x86_64 SAP  recommended
SLE-Module-Basesystem15-SP4-Updates for x86_64 SAP  recommended
SLE-Module-Desktop-Applications15-SP4-Pool for x86_64 SAP  recommended
SLE-Module-Desktop-Applications15-SP4-Updates for x86_64 SAP  recommended
SLE-Module-SAP-Applications15-SP4-Pool for x86_64  recommended
SLE-Module-SAP-Applications15-SP4-Updates for x86_64  recommended
SLE-Module-Server-Applications15-SP4-Pool for x86_64 SAP  recommended
SLE-Module-Server-Applications15-SP4-Updates for x86_64 SAP  recommended
SLE-Product-HA15-SP4-Pool for x86_64 SAP  recommended
SLE-Product-HA15-SP4-Updates for x86_64 SAP  recommended
SLE-Product-SLES_SAP15-SP4-Updates for x86_64  mandatory)
```

    f. Click Create Activation Key.

40. At the SUSE Manager terminal, verify that the important channels have been synced:

```
grep -E 'Importing packages (started|finished)|Linking packages to channel|Sync completed' \
  /var/log/rhn/reposync/sle-module-{suse-manager-server-4.3,{sap,server}-applications15-
sp4}-pool-x86_64.log
/var/log/rhn/reposync/sle-module-suse-manager-server-4.3-pool-x86_64.log:2022/11/18 16:06:54 -04:00
Importing packages started.
/var/log/rhn/reposync/sle-module-suse-manager-server-4.3-pool-x86_64.log:2022/11/18 16:06:57 -04:00
Importing packages finished.
/var/log/rhn/reposync/sle-module-suse-manager-server-4.3-pool-x86_64.log:2022/11/18 16:06:58 -04:00
Sync completed.
/var/log/rhn/reposync/sle-module-sap-applications15-sp4-pool-x86_64.log:2022/11/18 15:59:32 -04:00
Importing packages started.
/var/log/rhn/reposync/sle-module-sap-applications15-sp4-pool-x86_64.log:2022/11/18 15:59:33 -04:00
Importing packages finished.
/var/log/rhn/reposync/sle-module-sap-applications15-sp4-pool-x86_64.log:2022/11/18 15:59:34 -04:00
Sync completed.
/var/log/rhn/reposync/sle-module-server-applications15-sp4-pool-x86_64.log:2022/11/18 15:40:04
-04:00 Importing packages started.
/var/log/rhn/reposync/sle-module-server-applications15-sp4-pool-x86_64.log:2022/11/18 15:40:15
-04:00 Importing packages finished.
/var/log/rhn/reposync/sle-module-server-applications15-sp4-pool-x86_64.log:2022/11/18 15:40:18
-04:00 Sync completed.
mgr-create-bootstrap-repo -l SLE-module-sap-applications15-sp4-pool-x86_64 \
  SLE-module-server-applications15-sp4-pool sle-module-suse-manager-server-4.3-pool
```

41. Run SLE-15-SP4-x86_64:

```
mgr-create-bootstrap-repo -c SLE-15-SP4-x86_64
...
Pool started (with 5 workers)
Pool finished
```

42. Log into the UI by navigating a web browser to the SUSE Manager IP.
43. Click Admin→Manager Configuration→Bootstrap Script.
44. Check Enable Remote Configuration and Enable remote commands. Keep other selections the defaults (Bootstrap using Salt and Enable Client GPG checking).
45. Click Update.
46. Add the activation key to the bootstrap file:

```
cd /srv/www/htdocs/pub/bootstrap
sed 's/^ACTIVATION_KEYS.*/ACTIVATION_KEYS=1-SLES15-SP4' < bootstrap.sh  >bootstrap-sles15.sh
```

47. Apply the bootstrap commands to the three servers:

```
for i in {0..2}; do echo $i; cat bootstrap-sles15.sh | ssh tide${i} /bin/bash; done
for i in {0..2}; do echo $i; ssh tide${i} reboot; done
```

48. Fix machineID duplication for cloned VM tide1 by SSHing into it and running the following command:

```
rm -f /etc/machine-id
rm -f /var/lib/dbus-machine-id
dbus-uuidgen --ensure
systemd-machine-id-setup
```

49. Repeat step 41 for the cloned VM tide2.
50. Prepare the VMs for SALT.
51. SSH into VM tide0, and run the following commands:

```
zypper ar http://susemanager.binion2.lab/pub/repositories/sle/15/4/bootstrap/sles15-sp4
zypper in salt-minion
```

52. Set the SALT master server to susemanager.binion2.lab, and restart the service for the change to take effect:

```
sed -i 's/^#master:.*/master: susemanager.binion2.lab' /etc/salt/minion
systemctl restart salt-minion
```

53. Repeat steps 44 and 45 for the VMs tide1 tide2.
54. From the WebUI, click Salt→ Keys. Next to each pending hostname, right-click the button with a checkmark on it.

Setting up Red Hat Satellite 6.8

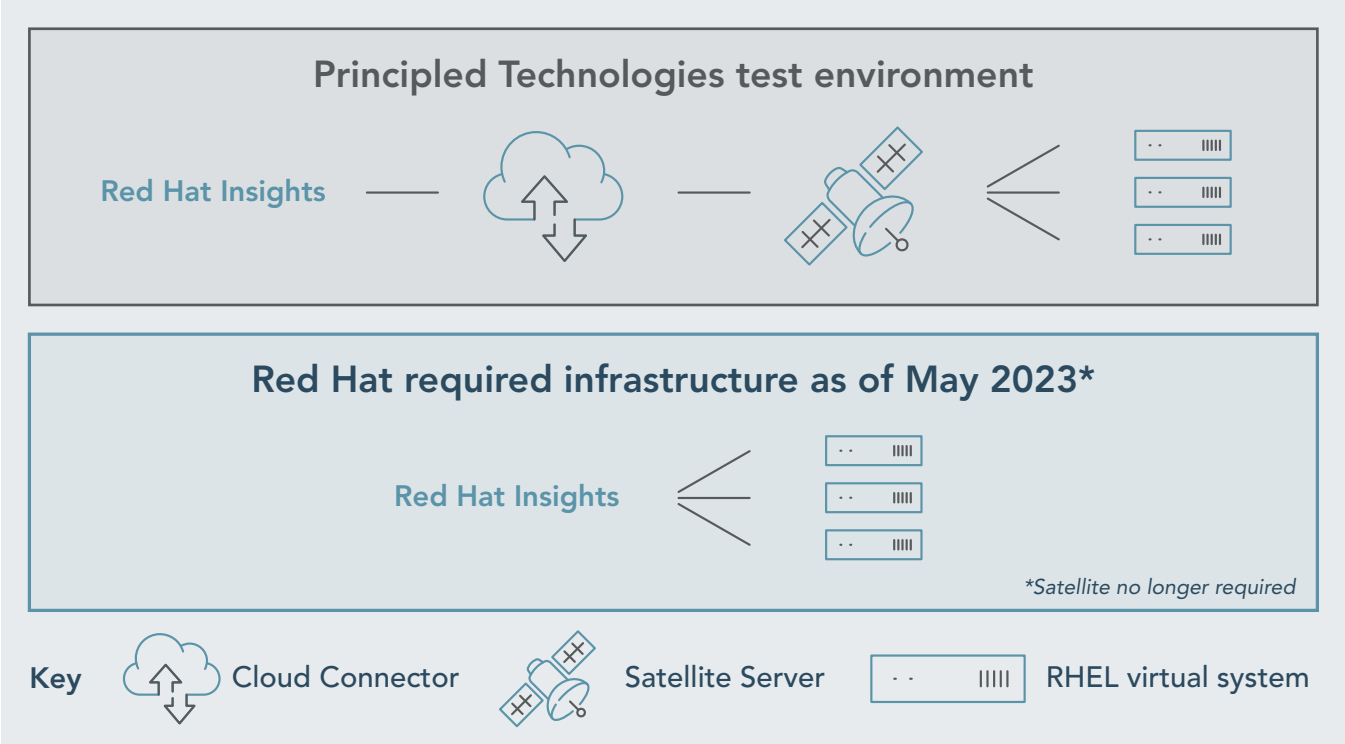
This section uses Red Hat's "Installing Red Hat Satellite Server from a Connected Network" product documentation, available at https://access.redhat.com/documentation/en-us/red_hat_satellite/6.8/html/installing_satellite_server_from_a_connected_network/index and Red Hat's "Remediating issues across your Red Hat Satellite infrastructure using Red Hat Insights" product documentation, available at https://access.redhat.com/documentation/en-us/red_hat_insights/2020-10/html/remediating_issues_across_your_red_hat_satellite_infrastructure_using_red_hat_insights/index.

At the time of testing in early 2023, Red Hat Satellite was required to execute remediations from Insights. The upper portion of the diagram below illustrates the test environment. As of May 2023, Red Hat Satellite is no longer required for these specific test scenarios, so the environment can be simplified. The lower portion of the diagram below illustrates the current relationship of the components.

Red Hat Satellite is included with Red Hat Enterprise Linux for SAP Solutions, but is an add-on for most Red Hat Enterprise Linux subscriptions. While no longer required for remediation from Red Hat Insights, Red Hat Satellite is still recommended for some use cases unrelated to these test scenarios such as:

- Automated provisioning to non-cloud environments,
- Scheduling remediations from Red Hat Insights recommendations,
- Managing granular control over software and package distribution,
- Managing large, complex environments spanning multiple locations and clouds, or
- Managing disconnected (air-gapped) environments.

Figure 1: The difference between the test environment in this study and current requirements.



All CLI commands in this section are meant to be executed on the Satellite VM. You can execute browser instructions from any system on the same network as the Satellite VM.

1. In your local vSphere, set up a system named satellite with 4CPU, 20,000MB of memory, 1 32GB HDD, and one network adapter connected. Perform a minimal install of RHEL 7.6, using custom storage to reduce the storage for / to 23.13 GB, and add a 4.67 GB LVM partition for /var. Set up manual IPv4 settings, and configure the full hostname before install.
2. Create a DNS entry for the satellite machine in your DNS server.
3. Use ssh to log into the admin VM.
4. Log into the Satellite system, and install Satellite:

    a. Log into the system:

    ```
    ssh-agent bash
    ```

    b. Change the hostname as necessary, and verify DNS resolution:

    ```
    sudo hostnamectl --sethostname satellite
    ping -c1 localhost
    ping -c1 `hostname -f`
    ```

    c. Register to Red Hat subscription management:

    ```
    sudo subscription-manager register
    ```

    d. Attach the Satellite Infrastructure subscription and Red Hat basic subscription:

    ```
    sudo subscription-manager list --all --available --matches 'Red Hat Satellite
    Infrastructure Subscription'
    sudo subscription-manager attach --pool=<pool_id from previous command>
    sudo subscription-manager list --all --available --matches 'Red Hat Enterprise Linux, Self-
    Support (128 Sockets, NFR, Partner Only)'
    sudo subscription-manager attach --pool=<pool_id from previous command>
    ```

    e. Verify subscriptions:

    ```
    sudo subscription-manager list --consumed
    ```

    f. Configure the Satellite Server repositories:

    ```
    sudo subscription-manager repos --disable "*"
    sudo subscription-manager repos --enable=rhel-7-server-rpms \
    --enable=rhel-7-server-satellite-6.8-rpms \
    --enable=rhel-7-server-satellite-maintenance-6-rpms \
    --enable=rhel-server-rhscl-7-rpms \
    --enable=rhel-7-server-ansible-2.9-rpms
    sudo yum clean all
    sudo yum repolist enabled
    ```

    g. Open up the required network ports:

    ```
    sudo firewall-cmd --add-port="443/tcp" --add-port="8140/tcp" --add-port="9090/tcp" --add-
    port="5000/tcp" --add-port="80/tcp" \
    --add-port="8000/tcp" --add-port="5646/tcp" --add-port="5647/tcp" --add-port="7/tcp" --add-
    port="7/udp" --add-port="53/tcp" \
    --add-port="53/udp" --add-port="67/udp" --add-port="69/udp" --add-port="22/tcp"
    sudo firewall-cmd --runtime-to-permanent
    ```

h. Install Satellite Server packages, adding a recent hack introduced here, https://community.theforeman.org/t/katello-installation-broken/21374, needed to install python2:

```
sudo yum -y update
sudo yum -y install wget
wget https://download-ib01.fedoraproject.org/pub/epel/7/aarch64/Packages/p/python2-
qpid-1.37.0-4.el7.noarch.rpm
sudo yum -y localinstall python2-qpid-1.37.0-4.el7.noarch.rpm
sudo yum -y install satellite
```

i. Double-check the clock synchronization (it was already active in our Satellite):

```
systemctl status chronyd
```

j. Install the SOS package:

```
sudo yum -y install sos
```

k. Run the installer:

```
sudo chmod a+w /etc/foreman-installer/scenarios.d/satellite.yaml
sudo satellite-installer --scenario satellite \
--foreman-initial-admin-username admin \
--foreman-initial-admin-password redhat \
--foreman-proxy-puppetca true \
--foreman-proxy-tftp true \
--enable-foreman-plugin-discovery
```

5. Export and import a subscription manifest named Satellite1. (Export uses instructions from https://access.redhat.com/documentation/en-us/red_hat_subscription_management/1/html/using_red_hat_subscription_management/using_manifests_con.)

   a. In a browser, open https://access.redhat.com.
   b. Log in.
   c. Click Subscriptions.
   d. Click Subscription Allocations.
   e. Enable Simple Content Access.
   f. Create a new Subscription Allocation:

      i. Click New Subscription Allocation.
      ii. For name, type `Satellite1`
      iii. Next to Type, click the down arrow and click Satellite 6.8.
      iv. Click Create.
      v. Under subscriptions, next to Simple content access, click the slider so it says Enabled.
      vi. Click Export Manifest, and save the file.

   g. Import the Satellite subscription:

      i. In a browser, navigate to http://<satellite ip>.
      ii. Log in as admin redhat.
      iii. Click Content.
      iv. Click Subscriptions.
      v. Click Import a Manifest.
      vi. Click Choose File.
      vii. Navigate to and select the appropriate manifest file.
      viii. Click Open.
      ix. An Import Manifest progress bar appears; when it completes the subscriptions will appear.

6. Install Insights, and register Satellite with Insights-client for system monitoring:

```
sudo satellite-maintain packages install -y insights-client
sudo insights-client --register
```

7. Enable repositories and synchronize.
8. In your browser, access http://<satellite ip>.
    a. Log in as admin redhat.
    b. Click Content.
    c. Click Red Hat Repositories.
    d. To the left of Available, type `Satellite Tools 6.8 (for RHEL 7 Server) (RPMs)`
    e. Click Search.
    f. Click Red Hat Satellite Tools 6.8 (for RHEL 7 Server) (RPMs).
    g. Next to x86_64, click the + icon to enable the repository.
    h. To the left of Available, type `Red Hat Enterprise Linux 7 Server (RPMs)`
    i. Click Search.
    j. Click Red Hat Satellite Tools 6.8 (for RHEL 7 Server) (RPMs).
    k. Next to x86_64 7Server, click the + icon to enable the repository.
    l. Click Content.
    m. Click Sync Status.
    n. Next to Red Hat Enterprise Linux Server, click the arrow.
    o. Next to Red Hat Satellite Tools 6.8 for RHEL 7 Server RPMs x86_64, click the checkbox.
    p. Click Synchronize Now.
    q. Wait for the Result column to say Synching Complete.
9. Create a content view and activation key:
    a. In your browser, access http://<satellite ip>.
    b. Log in as admin redhat.
    c. Click Content.
    d. Click Content Views.
    e. Click Create New View.
    f. Under Name, type `View1`
    g. Click Save.
    h. Click the boxes next to Red Hat Enterprise Linux 7 Server RPMs x86_64 7Server and Red Hat Satellite Tools 6.8 for RHEL 7 Server RPMs x86_64.
    i. Click Add Repositories.
    j. Click Publish New Version.
    k. Under Description, type `Initial View`
    l. Click Save.
    m. Click Content.
    n. Click Activation keys.
    o. Click Create Activation Key.
    p. Under Name, type `ActivationKey1`
    q. Click the Library checkbox.
    r. Under Content View, click the down arrow, and select View1.
    s. Click Save.
10. Register all local hosts to Satellite.
    a. Copy /etc/hosts from the Satellite server to the admin system ~/allserver.
    b. Add each local host to Satellite:
        i. Edit subscribe.sh:

```
subscription-manager remove --all
subscription-manager unregister
subscription-manager clean
rpm -Uvh http://<satellite IP>/pub/katello-ca-consumer-latest.noarch.rpm
subscription-manager register --org="Default_Organization" --activationkey=ActivationKey1
subscription-manager refresh
subscription-manager repos --enable=rhel-7-server-satellite-tools-6.8-rpms
yum repolist
yum install katello-host-tools
```

    c. On Satellite, edit the ~/servers file to contain all local servers:

```
grep VRedHat ~/allserver | awk '{ print $1 }'| head -90 > ~/servers
```

d. On Satellite, edit the copyssh.sh file as follows:

```
username=root
readarray -t SERVERS < ~/servers
for HOST in ${SERVERS[@]}; do
ssh-copy-id -i ~foreman-proxy/.ssh/id_rsa_foreman_proxy.pub ${username}@${HOST}
cat subscribe.sh | ssh -i ~foreman-proxy/.ssh/id_rsa_foreman_proxy ${username}@${HOST}
done
```

e. Make copyssh.sh executable:

```
chmod 0744 copyssh.sh
```

f. Execute the looping script, answering questions posed with yes or the system password, as appropriate:

```
ssh-agent bash
./copyssh.sh
```

11. Shut down every VRedHat*** system.
12. Create a Snapshot of each VRedHat*** system. (Before performing any use cases, revert each VM back to the snapshot.)
13. Turn on every VRedHat*** system.
14. Configure the VPN in Satellite.

Installing and setting up Cloud Connector

1. Upload your host inventory to Insights, and install Cloud Connector:

```
satellite-installer \
--enable-foreman-plugin-rh-cloud \
--foreman-proxy-plugin-remote-execution-ssh-install-key true
```

2. In a browser, open the Satellite web UI at https://<satellite IP>.
3. Click Configure.
4. Click Inventory Upload.
5. Click Default_Organization.
6. Click Restart, and wait until Exit Code: displays the pid and an exit code of 0.
7. Click Administer.
8. Click Users.
9. Click Create User.
10. Next to Username, enter `User1`
11. Under Authorized by, click the down arrow, and click INTERNAL.
12. Next to Password, type `Password`
13. Next to verify, type `Password`
14. Click Organization.
15. Ensure Default Organization is in the Selected items box, otherwise click on it to move it there.
16. Ensure Default on login is blank.
17. Click Roles.
18. Click the Administrator box.
19. Click Submit.
20. Click Hosts.
21. Click All Hosts.
22. Click the Satellite host.
23. Click Schedule Remote Job.
24. Next to Job Category list, click the down arrow, and select Ansible Playbook.
25. Next to Job Template list, click the down arrow, and select Configure Cloud Connector.
26. Next to satellite_user field, type `User1`

27. Next to satellite_password , type `Password`
28. Click Submit.
29. Wait for the Results to say 100% Success.
30. Make sure the following is at the end of the /etc/systemd/system/receptor@.service file. If it is not, append it:

```
[Install]
WantedBy=multi-user.target
```

31. Check https://cloud.redhat.com/settings/sources to make sure Satellite is there.
32. Use a new browser tab to access https://access.redhat.com/management.api.
33. Click GENERATE TOKEN.
34. Click Copy.
35. Return to the original browser tab.
36. Click Administer.
37. Click Settings.
38. Click RH Cloud.
39. In the Value column of the Red Hat Cloud token, click the pencil to edit.
40. Highlight the four stars.
41. Press Delete.
42. From the menu that appears, right-click, and click Paste.
43. Click the blue checkmark.
44. Click Configure.
45. Click Insights.
46. Click Sync now.
47. Next to Synchronize Automatically, click the blank box to turn the setting ON.
48. Open a new browser window,and navigate to cloud.redhat.com/settings/groups.
49. Click the Group you are a member of. In our case, it was Default access.
50. Use the box beside the down arrow to select all the roles.
51. Click Add role.
52. At the warning, click OK.
53. Roles that are already inherited will be removed; click the top box to select all the remaining roles.
54. Click Add to group.

Use case: SAP Advisory (Red Hat)

We used these steps for both SAP Advisory use cases: best practices and patches.

1. Navigate to console.redhat.com/insights.
2. Log in.
3. Click Advisor.
4. Click Topics.
5. Click SAP.
6. Click the advisory this Use Case is about.
7. Select all affected systems.
8. Click Remediate.
9. Type in a name for the remediation playbook.
10. Click Next.
11. Click Next.
12. Click Submit.
13. Click to Open Playbook.
14. Click Execute Playbook.
15. Click Execute playbook on two systems.
16. Wait for remediation.

Use case: SAP Advisory - package update (SUSE)

We used these steps for the SAP Advisory patches use case.

1. Navigate to SUSE Manager.
2. Log in.
3. Click Patches.
4. In the search bar, enter SAP.
5. Click the appropriate SAP Patch.
6. Scroll and click on the provided Vendor Advisory link.
7. Scan through the Vendor Advisory (not counting reading time).
8. Click Packages, and view.
9. Click Affected Systems, and view.
10. Click Select All.
11. Click Apply Patches.
12. Click Confirm.
13. Click Schedule.
14. Click the appropriate patch update.
15. Click In Progress Systems.
16. Refresh until all systems disappear from list of In Progress systems.
17. Click Completed Systems.

Use case: SAP Advisory - best practices (SUSE)

We used these steps for the SAP Advisory best practices use case.

1. Spend time researching an error with SAP or best practices and how to manually remediate. Find four errors, each requiring a Google search, clicking a link, and skimming an article. (Used pre-determined searches and sites.) Find known issues with tempfs Filesystem / dev/shm is getting full in HANA host at https://answers.sap.com/questions/13065054/filesystem-sevshm-is-getting-fll-in-haha-host-html Find clues on how to resize /dev/shm at https://stackoverflow.com/questions/58804022/how-to-resize-dev-shm and https://answers.sap.com/questions/10161697/dispwork-starting-but-stop-after-some-time.html. Decide to follow best practices and resize before it becomes an issue.
2. Go to SUSE Manager.
3. Log in.
4. Click Systems.
5. Click First System.
6. Click Remote Command.
7. Copy and paste a diagnostic command from your previous searches at https://answers.sap.com/questions/10161697/dispwork-starting-but-stop-after-some-time.html:

```
sappfpar -check PF=<instance profile>
```

8. Alter the command to suit your system.
9. Click Schedule.
10. Click Schedule, and find that there are no actions pending.
11. Click Systems.
12. Click First System.
13. Click Remote Command.
14. Copy and paste a diagnostic command from your previous searches:

```
sappfpar -check
```

15. Next to Command label, click, and name the remote command `sappfpar`
16. Click Schedule.
17. Click Event link, and scan to see that the diagnostic failed: command not found.
18. Click Back.

19. Type a different diagnostic command:

```
free -h
```

20. Click Schedule.
21. Click Event link, and scan to see diagnostic output from the command.
22. After finding the SAP commands aren't working, review previously found websites for remediation instructions and see that the recommended size of tempfs is 75% of RAM + swap.
23. Click Back.
24. Type more diagnostic/fix commands to diagnose swap size and tempfs:

```
cat /etc/fstab
cp /etc/fstab /etc/fstabbkup
ls /etc/fstab
```

25. Click Schedule.
26. Click Event link, and scan to see client has not completed this action.
27. Click Refresh, and get the same result.
28. Click Schedule.
29. Click the Remote command.
30. Click In Progress Systems, and see the first system listed.
31. Click Refresh until the system disappears from In Progress Systems.
32. Click Completed Systems, and see the first system listed.
33. To get the command output, click System.
34. Copy commands needed from earlier research (https://stackoverflow.com/questions/58804022/how-to-resize-dev-shm).
35. Click Salt.
36. Click Remote Commands.
37. Type and paste remediation commands into interface:

```
cp /etc/fstab /etc/fstabbkup; echo "none /dev/shm tempfs defaults,size=4G 0 0" >> /etc/fstab; mount
/dev/shm; lsblk
```

38. Keep * for the list of systems, and click Run Command.
39. Expand each of the three systems, and ensure the results are correct.
40. Enter command to double check results:

```
cat /etc/fstab
```

41. Click Run Command, and view already expanded results.
42. Enter another command to double check results:

```
lsblk
```

43. Click Run Command, and view already expanded results.
44. Enter another command to double check results:

```
df -h
```

45. Click Run Command, and view already expanded results.
46. Enter another command to double check results:

```
cat /etc/fstab
```

47. Click Run Command, and view already expanded results.

48. Enter a command to fix the size from the cut and pasted 4G to 31G:

```
sed -ie 's+4G+31G+g' /etc/fstab
```

49. Click Run Command.
50. Enter another command to double check results:

```
cat /etc/fstab
```

51. Click Run Command, and view already expanded results.
52. Enter the final remediation command:

```
mount /dev/shm
```

53. Click Run Command.
54. Enter a final diagnostic command to check results:

```
df -h
```

55. Click Run Command, and view already expanded results.

Use case: Drift (Red Hat)

Creating a baseline configuration

1. Navigate to console.redhat.com/insights.
2. Log in.
3. Click Drift.
4. Click Baselines.
5. Click Create baseline.
6. Click the Copy an existing system or historical profile radio button.
7. Enter the desired Baseline name.
8. Select the radio button for the first SAP system from the list of systems.
9. Click Create baseline.
10. Scroll through to last_boot_time.
11. Right-click the three dots, and from the drop-down menu, select Delete.
12. Click Delete facts.
13. Scroll through to network_interfaces.
14. Right-click the three dots, and from the drop-down menu, select Delete.
15. Click Delete facts.
16. At the top left, click the Baselines link.
17. Click Create baseline.
18. Click the Copy an existing system or historical profile radio button.
19. Enter the desired Baseline name.
20. Select the radio button for the second SAP system from the list of systems.
21. Click Create baseline.
22. Scroll through to fqdn.
23. Right-click the three dots, and from the drop-down menu, select Delete.
24. Click Delete facts.
25. Scroll through to last_boot_time.
26. Right-click the three dots, and from the drop-down menu, select Delete.
27. Click Delete facts.
28. Scroll through to network_interfaces.
29. Right-click the three dots, and from the drop-down menu, select Delete Category.
30. Click Delete facts.

31. At the top left, click the Baselines link.
32. Click Create baseline.
33. Click the Copy an existing system or historical profile radio button.
34. Enter the desired Baseline name.
35. Select the radio button for the third SAP system from the list of systems.
36. Click Create baseline.
37. Scroll through to fqdn.
38. Right-click the three dots, and from the drop-down menu, select Delete.
39. Click Delete facts.
40. Scroll through to last_boot_time.
41. Right-click the three dots, and from the drop-down menu, select Delete.
42. Click Delete facts.
43. Scroll through to network_interfaces.
44. Right-click the three dots, and from the drop-down menu, select Delete Category.
45. Click Delete facts.

Creating drift

1. Uninstall tmux package and rmmod overlay kernel module from Node 0.
2. Uninstall compat-sap-c++-9 package from Node 1.
3. Do a system update, and uninstall the compat-sap-c++-9 package from Node 2.

Detecting drift

While completing the following steps, omit reading time from your recorded results.

1. Log in.
2. Click Drift.
3. Click Comparisons.
4. Select the radio button for the first SAP system from the list of systems.
5. At the top of the box, click Baselines.
6. Scroll to and click the checkbox to select the first baseline you previously created.
7. Click Submit.
8. Observe that Insights has marked differences in installed_packages, kernel_modules, network_interfaces and running_processes. Scroll through to check the other categories that don't have differences.
9. Click the arrow to expand the installed_packages category to reveal kernel, kernel-core, and kernel-modules all marked as having differences, as well as one individual package below the subcategories, tmux, is marked as having differences.
10. Click the arrow to expand the kernel category to reveal that the kernels are the same, but a kernel line of the baseline is blank while a the same line in the system is filled in. All other kernel lines are identical. There isn't a kernel difference, simply one of the kernel lines is missing from the baseline and isn't missing from the system so it got marked as a difference.
11. Observe that the tmux package has a version in the baseline and is blank in the system, therefore it has been uninstalled and needs to be reinstalled. Make a note of it.
12. Scroll down to check on the rest of the packages whose versions match.
13. Scroll back and click the arrow to unexpand the kernel subcategory.
14. Click the arrow to expand the kernel-core category to reveal that the kernels are the same and this category is falsely marked as changed just as before.
15. Click the arrow to expand the kernel-modules category to once again reveal the category is falsely marked, as before.
16. Click the arrow next to installed_packages to unexpand it.
17. Click the arrow next to kernel_modules to expand it.
18. Notice the one disabled module, and make a note of it.
19. Scroll through the rest of the modules that haven't been disabled.
20. Click the arrow next to kernel_modules to unexpand it.
21. Click the arrow next to network_interfaces to expand it and reveal that it is falsely marked as changed because the baseline doesn't have any data in the category, while the system does.
22. Click the arrow next to network_interfaces to unexpand it.
23. Click the arrow next to running processes to expand it.
24. Note the change in running processes (mostly kworker).
25. Click the X to remove the baseline from the comparison.
26. Click the X to remove the system from the comparison.
27. Click Add systems or baselines.

28. Scroll to and click the checkbox to select the second baseline you previously created.
29. At the top of the box, click Systems,
30. See error: Unable to load inventory component. Failed to load section. More info can be found in browser console output. (Omit from steps.)
31. While viewing the error, a list of systems comes up.
32. Select the second system.
33. Click Baselines.
34. Scroll to and click the checkbox to select the second baseline you previously created.
35. Click Submit.
36. Observe that Insights has marked differences in installed_packages, kernel_modules, network_interfaces and running_processes.
37. Click the arrow to expand the installed_packages category to reveal kernel, kernel-core, and kernel-modules all marked as having differences, as well as one individual package above the subcategories, compat-sap-c++-9, is marked as having differences. Make a note of the package.
38. Click the arrow to expand the kernel category to reveal that the kernels are the same, but a kernel line of the baseline is blank while a the same line in the system is filled in. All other kernel lines are identical. There isn't a kernel difference, simply one of the kernel lines is missing from the baseline and isn't missing from the system so it got marked as a difference.
39. Click the arrow to expand the kernel-core category to reveal that the kernels are the same and this category is falsely marked as changed just as before.
40. Click the arrow to expand the kernel-modules category to once again reveal the category is falsely marked, as before.
41. Click the arrow next to installed_packages to unexpand it.
42. Click the arrow next to kernel_modules to expand it.
43. Notice the one disabled module, and make a note of it.
44. Click the arrow next to kernel_modules to unexpand it.
45. Click the arrow next to network_interfaces to expand it and reveal that it is falsely marked as changed because the baseline doesn't have any data in the category while the system does.
46. Click the arrow next to network_interfaces to unexpand it.
47. Click the arrow to running processes to expand it.
48. Note the change in running processes.
49. Scroll through the rest of the processes and baseline comparison to double check for any other differences.
50. Click the X to remove the baseline from the comparison.
51. Click the X to remove the system from the comparison.
52. Click Add systems or baselines.
53. Scroll to and click the checkbox to select the third baseline you previously created.
54. At the top of the box, click Systems.
55. See error: Unable to load inventory component. Failed to load section. More info can be found in browser console output. (Omit from steps.)
56. Click Cancel.
57. Click the username.
58. Click Log out.
59. Log in again.
60. Add /insights to URL to navigate to the Insights from Home screen.
61. Click Drift.
62. Click Comparisons.
63. Click Add systems or baselines.
64. Scroll to and click the checkbox to select the third system.
65. At the top of the box, click Baselines.
66. Scroll to and click the checkbox to select the third baseline you previously created.
67. Click Submit.
68. Observe that Insights has marked differences in enabled_services, installed_packages, installed_services, kernel_modules, network_interfaces and running_processes.
69. Click the arrow to expand the enabled_services category to show that pacemaker and pcsd services are missing. Make a note.
70. Click the arrow next to enabled_services to unexpand it.
71. Click the arrow to expand the installed_packages category to reveal kernel, kernel-core, and kernel-modules all marked as having differences, as well as multiple individual packages having differences. Make a note of the packages.
72. Click the arrow to expand the kernel category to reveal that the kernels are the same, but a kernel line of the baseline is blank while a the same line in the system is filled in. All other kernel lines are identical. There isn't a kernel difference, simply one of the kernel lines is missing from the baseline and isn't missing from the system so it got marked as a difference.
73. Click the arrow to expand the kernel-core category to reveal that the kernels are the same and this category is falsely marked as changed just as before.

74. Click the arrow to expand the kernel-modules category to once again reveal the category is falsely marked, as before.
75. Click the arrow next to installed_packages to unexpand it.
76. Click the arrow next to installed_services to expand it.
77. Make a note that the pcsd service is missing.
78. Scroll through the rest of the services.
79. Click the arrow next to installed_services to unexpand it.
80. Notice that after reviewing some categories, the facts categories have changed order. Most likely the arrow at the top of the category was clicked to flip it on accident. Click installed_packages because it is underneath installed_services even though it has already been done. Unexpand. (Omit time and steps from user experience/error)
81. Click enabled services due to the same issue. Unexpand. (Omit time and steps from user experience/error.)
82. Click the arrow next to running_processes to expand it.
83. Make note of the marked missing running_processes.

Use case: Drift (SUSE)

Creating a baseline

1. Open a shell connection to SUSE Manager console.
2. Log in.
3. Change to the salt directory /srv/salt:

```
cd /srv/salt/
```

4. Do Google searches and searches of docs.saltproject.io to find the appropriate salt modules to use and experiment at the command line to find ways to alter the outputs from various commands to create lists of output that can be usable in the *.sls files. We would expect even a salt expert to have to do some research and scripting to get the correct commands and formatting from salt, dmidecode, ps aux, diff, head, tail, awk, sed, fgrep, history, eval, scp, ssh, suseconnect, free, zypper, and systemctl for uname/architecture grains, bios-release-date grains, system-manufacturer/biosvendor grains, bios vendor, cpu data, system-product-name/infrastructure grains, os-release grains, lists of running processes, subscription status, free memory, repos, running services, lists of installed and available packages, and lists of enabled kernel modules. While it took the tester significant time, we can't penalize SUSE for our tester being new to some of the necessary commands. If we assume the status of a master, which is a best-case scenario for SUSE, it still makes sense to include some research time into each item, if only to make sure that all lists are formatted correctly for the *.sls files to use. Assuming at least 3 minutes/3 steps (search, click, scroll and read) for each command listed and 3 minutes/3 steps (try out at command line, tweak output, repeat at least once with alterations) for each data grain or list to obtain, we have a total of (31 categories X 3 minutes or steps each) = 93 minutes, 93 steps. This is a minimum and a highly subjective estimate. It is not included in our grand total of steps/minutes for this reason.
5. Edit the top.sls file to redirect to a different *.sls baseline file for each of the three systems in the cluster (vi command, enter 7 lines, save = 9 steps):

```
base:
  'tide0.example.com':
    - tide0
  'tide1.example.com':
    - tide1
  'tide2.example.com':
    - tide2
```

6. Make a directory for baseline files for the first system on the SUSE Manager:

```
mkdir /home/username/tide0.example.comsalt
```

7. Make a directory for baseline files for the second system on the SUSE Manager:

```
mkdir /home/username/tide1.example.comsalt
```

8. Make a directory for baseline files for the third system on the SUSE Manager:

```
mkdir /home/username/tide2.example.comsalt
```

9. Edit the sls file for the first system, including a command for obtaining the current data and a command for comparing it to stored baseline data (two commands each) for the following:

   a. Architecture

```
'echo -n "    " > /tmp/archcurrent; uname -m >> /tmp/archcurrent':
  cmd.run
'diff /root/tide0.example.comsalt/archbaseline /tmp/archcurrent':
  cmd.run
```

   b. BIOS release date

```
'echo -n "    " > /tmp/biosreleasedatecurrent; dmidecode -s bios-release-date >> /tmp/
biosreleasedatecurrent':
  cmd.run
'diff /root/tide0.example.comsalt/biosreleasedatebaseline /tmp/biosreleasedatecurrent':
  cmd.run
```

   c. System manufacturer

```
'echo -n "    " > /tmp/biosvendorcurrent; dmidecode -s system-manufacturer >> /tmp/
biosvendorcurrent':
  cmd.run
'diff /root/tide0.example.comsalt/biosvendorbaseline /tmp/biosvendorcurrent':
  cmd.run
```

   d. BIOS version

```
'echo -n "    " > /tmp/biosversioncurrent; dmidecode -s bios-version >> /tmp/
biosversioncurrent':
  cmd.run
'diff /root/tide0.example.comsalt/biosversionbaseline /tmp/biosversioncurrent':
  cmd.run
```

   e. CPU data

```
'lscpu > /tmp/cpucurrent':
  cmd.run
'diff /root/tide0.example.comsalt/cpubaseline /tmp/cpucurrent':
  cmd.run
```

   f. Infrastructure

```
'echo -n "    " > /tmp/infrastructurecurrent; dmidecode -s system-product-name >> /tmp/
infrastructurecurrent':
  cmd.run
'diff /root/tide0.example.comsalt/infrastructurebaseline /tmp/infrastructurecurrent':
  cmd.run
```

   g. OS release

```
'echo -n "    " > /tmp/oscurrent; cat /etc/os-release | grep PRETTY_NAME | head -c -2 | tail -c
+14 >> /tmp/oscurrent; echo >> /tmp/oscurrent':
  cmd.run
'diff /root/tide0.example.comsalt/osbaseline /tmp/oscurrent':
  cmd.run
```

h. Terminated processes that were in the baseline

```
'ps aux | awk "{ print \$11 }" > /tmp/processes':
  cmd.run
'echo "terminated processes that are not running that were in the baseline:"':
  cmd.run
'fgrep -v -x -f /tmp/processes /root/tide0.example.comsalt/processes':
  cmd.run
```

i. New processes that were not in the baseline

```
'echo "new processes that are running that were not in the baseline:"':
  cmd.run
'fgrep -v -x -f /root/tide0.example.comsalt/processes /tmp/processes':
  cmd.run
```

j. Subscription status

```
'suseconnect --status-text > /tmp/subscriptioncurrent':
  cmd.run
'diff /root/tide0.example.comsalt/subscriptionbaseline /tmp/subscriptioncurrent':
  cmd.run
```

k. Total memory

```
'free -h | grep Mem | head -c -61 | tail -c +16 > /tmp/memory; echo >> /tmp/memory':
  cmd.run
'diff /root/tide0.example.comsalt/memory /tmp/memory':
  cmd.run
```

l. Software repositories.

```
'zypper lr -E -e - | grep name= > /tmp/repos':
  cmd.run
'diff /root/tide0.example.comsalt/repos /tmp/repos':
  cmd.run
```

10. Note: The vi command + 50 lines + check for and fix 7 errors + save = 59 steps.
11. Run salt state.apply to make sure file gets processed properly, and find another error:

```
salt tide0* state.apply
```

12. Fix the error (vi command + fix + save = 3 steps).
13. Run salt state.apply to make sure file gets processed properly. No parsing errors.

```
salt tide0* state.apply
```

14. Copy the first sls file to make a duplicate for the second system:

```
cp tide0.sls tide1.sls
```

15. Copy the first sls file to make a duplicate for the third system:

```
cp tide0.sls tide2.sls
```

16. Use sed to substitute the host name in the sls file for the second system:

```
sed -i -e 's+tide0+tide1+g' tide1.sls
```

17. To make sure the substitution happened, vi the second sls file.
18. Use sed to substitute the host name in the sls file for the third system:

```
sed -i -e 's+tide0+tide2+g' tide2.sls
```

19. At this point, you have finished the most easily manipulated portion of the sls file. Now it is time to manually create the baseline data that we will be comparing to as well as formatting and putting any lists into the individual sls files. We use environment variables instead of host names and do this for the first host, so that when we do this for the other two hosts we can refer back to and use the command history to execute more quickly.
20. Assign the environment variable to store the first host fqdn:

```
currentminion=tide0.example.com
```

21. Assign the environment variable to store the first host short hostname:

```
shortcurrentminion=tide0
```

22. Obtain a list of salt grains that are available to double check what is needed:

```
salt tide0* grains.ls
```

23. Obtain the architecture information for the baseline and store in the baseline files directory you previously created:

```
salt ${currentminion} grains.fetch cpuarch | tail -n 1 > /home/username/${currentminion}
salt/archbaseline
```

24. Obtain the BIOS release date for the baseline, and store it:

```
salt ${currentminion} grains.fetch biosreleasedate | tail -n 1 > /home/username/${currentminion}
salt/biosreleasedatebaseline
```

25. Fix a typo in the last command and re-execute it correctly (omit user experience time and steps).
26. Obtain the manufacturer for the baseline and store it:

```
salt ${currentminion} grains.fetch manufacturer | tail -n 1 > /home/username/${currentminion}salt/
biosvendorbaseline
```

27. Obtain the BIOS version, and store it:

```
salt ${currentminion} grains.fetch biosversion | tail -n 1 > /home/username/${currentminion}salt/
biosversionbaseline
```

28. Obtain the CPU information, and store it:

```
ssh ${currentminion} "lscpu" > /home/username/${currentminion}salt/cpubaseline
```

29. Obtain the infrastructure information, and store it:

```
salt ${currentminion} grains.fetch productname | tail -n 1 > /home/username/${currentminion}salt/
infrastructurebaseline
```

30. Obtain the OS release, and store it:

```
salt ${currentminion} grains.fetch oscodename | tail -n 1 > /home/username/${currentminion}
salt/osbaseline
```

31. Obtain a list of currently running processes, and store it:

```
ssh ${currentminion} "ps aux" | awk '{print $11}' | tail -n +2 > /home/
username/${currentminion}salt/processes
```

32. Obtain the subscription status and store

```
ssh ${currentminion}   suseconnect --status-text > /home/username/${currentminion}salt/
subscriptionbaseline
```

33. Obtain the total memory information, and store it:

```
ssh ${currentminion} "free -h" | grep Mem | awk '{print $2}' > /home/
username/${currentminion}salt/memory
```

34. Obtain the system repositories, and store it:

```
ssh ${currentminion} "zypper lr -E -e -" | grep name= > /home/username/${currentminion}salt/repos
```

35. Cat all the stored information files to make sure they are correct (omitting reading time):

```
cat /home/username/${currentminion}salt/repos/*
```

36. Obtain a list of all the enabled services and store temporarily:

```
ssh ${currentminion} "systemctl list-unit-files --state=enabled" | awk '{print $1}' | tail -n +2 |
head -n -2 > /tmp/servicesenabled
```

37. Cat the list of services to make sure it is correctly stored (omitting reading time):

```
cat /tmp/servicesenabled
```

38. Create and execute a bash loop to go through the list of services and append them to the *.sls file as specified service.enabled (five lines to type, when typed perfectly):

```
readarray -t ENABLED < /tmp/servicesenabled
for SERVICE in ${ENABLED[@]}; do
  echo "${SERVICE}:" >> ${shortcurrentminion}.sls
  echo "  service.enabled" >> ${shortcurrentminion}.sls
done
```

39. Cat *sls file to make sure the services appended to the end of the file, and find out they didn't (omit reading time):

```
cat tide0.sls
```

40. Find the typo from the command history, and re-execute the bash loop (omitting user experience/errors time and steps).
41. Cat the *.sls file again to make sure it is correct (omitting user experience /errors time and steps):

```
cat tide0.sls
```

42. Obtain a list of all the services that are disabled and store it:

```
ssh ${currentminion} "systemctl list-unit-files --state=disabled" | awk '{print $1}' | tail -n +2 |
head -n -2 > /tmp/servicesdisabled
```

43. Check the new file to make sure the services were stored correctly (omit reading time):

```
cat tide0.sls
```

44. Create and execute a bash loop to go through the list of services and append them to the *.sls file as specified service.disabled (five lines to type, when typed perfectly):

```
readarray -t DISABLED < /tmp/servicesdisabled
for SERVICE in ${DISABLED[@]}; do
  echo "${SERVICE}:" >> ${shortcurrentminion}.sls
  echo "  service.disabled" >> ${shortcurrentminion}.sls
done
```

45. Cat the *.sls file again to make sure it is correct (omit reading time):

```
cat tide0.sls
```

46. Edit the end of the sls file to include the setup for checking for installed packages: (`vi $shortcurrentminion).sls`)

```
mypkgs:
 pkg.installed:
  - pkgs:
```

47. Enter vi, shift+g to end of file, enter three lines, and save work (six steps).
48. Obtain a list of installed packages, format, and append to the *.sls file:

```
salt ${currentminion} pkg.list_pkgs | tail -n +3 | paste -d " " - - | awk '{print "      - "$1"
"$2}' >> ${shortcurrentminion}.sls
```

49. Edit the end of the sls file to include the setup for checking for available but not installed packages: (`vi $shortcurrentminion).sls`)

```
purgedpkgs:
 pkg.purged:
  - pkgs:
```

50. Enter vi, shift+g to end of file, enter three lines, and save work (six steps), plus double check the last append, for seven steps.
51. Obtain a list of available but not installed packages, format, and append to the *.sls file:

```
salt ${currentminion} pkg.search '*' not_installed_only=True | tail -n +3 | grep -B1 "\-\-\-\-\-" |
grep ":" | awk -F' |:' '{print "      - "$5}' >> ${shortcurrentminion}.sls
```

52. Edit the end of the sls file to include the setup for checking for kernel modules: (`vi $shortcurrentminion).sls`)

```
mods:
 kmod.present:
  - mods:
```

53. Enter vi, shift+g to end of file, enter three lines, and save work (six steps), plus double check the last append, for seven steps.
54. Obtain a list of loaded kernel modules, format, and append to the *.sls file:

```
salt ${currentminion} kmod.mod_list | tail -n +2 | awk '{ print "  "$0}' >>
${shortcurrentminion}.sls
```

55. Obtain a list of loaded kernel modules, format, and save it in a temporary file for later:

```
salt ${currentminion} kmod.mod_list | tail -n +2 > /tmp/mod_list
```

56. Edit the end of the sls file to include the setup for checking for kernel modules that should not be loaded: (`vi $shortcurrentminion).sls`)

```
absentmods:
 kmod.absent:
  - mods:
```

57. Enter vi, shift+g to end of file, enter three lines, and save work (six steps), plus double check the last append, for seven steps.
58. Obtain a list of available kernel modules and save it in a temporary file for later:

```
salt $currentminion kmod.available | tail -n +2 > /tmp/avail
```

59. Check the new files to make sure the modules were stored correctly (omit reading time):

```
cat /tmp/mod_list; cat /tmp/avail
```

60. Use grep to compare the temporary files and obtain a list of kernel modules that are available but not currently loaded, format the output, and append it to the *sls file scroll to double check the grep executed correctly:

```
grep -v -x -f /tmp/mod_list /tmp/avail | awk '{ print "  "$0}' >> ${shortcurrentminion}.sls
```

61. Double check the append to the *.sls file is correct (omit reading time):

```
vi $shortcurrentminion).sls
```

62. Make a directory on the current host for the baseline files:

```
ssh $currentminion "mkdir /root/${currentminion}salt"
```

63. Copy a set of baseline files from SUSE Manager to the baseline host for salt to use. (Note: When salt processes an *.sls file, it runs on the minion file system, so any comparisons must be to files located on the minion instead of the manager.)

```
scp /home/username/${currentminion}salt/* ${currentminion}:/root/${currentminion}salt/
```

64. Double-check the files copied correctly:

```
ssh tide0 "ls -l /root/tide0.example.comsalt/*"
```

65. Double-check the new sls file execute successfully with salt, find an error with the kmodspresent due to some of the modules having numeric names and being treated as a int instead of a string. (Note: This would happen regardless of user experience/error and would need to be fixed. Keeping steps for this error, omitting reading time searching for error.)

```
salt tide0* state.apply
```

66. Edit the numeric modules names in the sls file so that they have quotes around them:

```
vi $shortcurrentminion).sls
```

67. Double-check the new sls file executes successfully with salt:

```
salt tide0* state.apply
```

68. Execute the history command, and note which series of commands we can repeat in large bunches using the eval command, adding in any text commands to be entered in editors. We ended up with 13 bunches of commands and four bits of text to enter. Adding setting environment variables at the beginning and testing the state file at the end results in 20 steps.

```
history | more
```

69. Assign the environment variable to store the second host fqdn, and assign the environment variable to store the short hostname of the second host all in one step using copy/paste from the notes:

```
currentminion=tide1.example.com
shortcurrentminion=tide1
```

70. Obtain the architecture information for the baseline, BIOS release date, manufacturer, BIOS version, CPU, infrastructure, OS release, currently running processes, subscription status, total memory, and system repos in stored files, and cat these files in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

71. Execute a bash loop to go through the list of services and append them to the *.sls file as specified service.enabled and cat the *.sls file to make sure the services appended in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

72. Obtain a list of all the services that are disabled and store, and check the file in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

73. Execute a bash loop to go through the list of services and append them to the *.sls file as specified service.disabled and cat the sls file again to double-check -in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

74. While already in vi, edit the end of the sls file using copy paste to include the setup for checking for installed packages:

```
mypkgs:
 pkg.installed:
  - pkgs:
```

75. Shift+g to end of file, set paste, enter three lines, and save work (six steps).
76. Obtain a list of installed packages, format, append to the *.sls file, and check the append in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

77. While already in vi, edit the end of the sls file using copy paste to include the setup for checking for available but not installed packages:

```
purgedpkgs:
 pkg.purged:
  - pkgs:
```

78. Shift+g to end of file, set paste, enter three lines, and save work (six steps).
79. Obtain a list of available but not installed packages, format, and append to the *.sls file, then check the append in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

80. While already in vi, edit the end of the sls file using copy paste to include the setup for checking for kernel modules:

```
mods:
 kmod.present:
  - mods:
```

81. Shift+g to end of file, set paste, enter three lines, and save work (six steps).
82. Obtain a list of loaded kernel modules, format, and append to the *.sls file and obtain a list of loaded kernel modules, format and save it in a temporary file for later, and open the sls file to double-check the append in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

83. While already in vi, edit the end of the sls file using copy paste to include the setup for checking for kernel modules that should not be loaded:

```
absentmods:
 kmod.absent:
  - mods:
```

84. Shift+g to end of file, set paste, enter three lines, and save work (six steps).
85. Obtain a list of available kernel modules, save it in a temporary file for later, and check the files to make sure the information was stored correctly in two steps using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

86. Use grep to compare the temporary files and obtain a list of kernel modules that are available but not currently loaded, format the output and append it to the *sls file scroll to double-check the grep executed correctly, and check the append in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

87. While in vi, fix the numeric kernel modules so they have quotes.
88. Make a directory on the current host for the baseline files and copy a set of baseline files from SUSE Manager to the baseline host for salt to use, and double-check the files copied properly in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

89. Double-check the new sls file executes successfully with salt, and find out that two kernel versions were listed, which messes with the sls file. (This would be an error/problem no matter what, keeping time and steps for fixing this error.)

```
salt tide1* state.apply
```

90. Edit the sls file to check for only the latest kernel version:

```
vi tide1.sls
```

91. Check that the new sls file executes successfully with salt:

```
salt tide1* state.apply
```

92. Assign the environment variable to store the third host fqdn, and assign the environment variable to store the short hostname of the third host in one step using copy/paste from the notes:

```
currentminion=tide2.example.com
shortcurrentminion=tide2
```

93. Obtain the architecture information for the baseline, BIOS release date, manufacturer, BIOS version, CPU, infrastructure, OS release, currently running processes, subscription status, total memory, and system repos in stored files and cat all these files in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

94. Execute a bash loop to go through the list of services and append them to the *.sls file specified as service.enabled, and cat the *.sls file to make sure the services appended in one step using history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

95. Obtain a list of all the services that are disabled, store it, and check the file in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

96. Execute a bash loop to go through the list of services and append them to the *.sls file specified service.disabled, and cat the sls file again to double-check in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

97. While already in vi, edit the end of the sls file using copy/paste to include the setup for checking for installed packages:

```
mypkgs:
 pkg.installed:
  - pkgs:
```

98. Shift+g to end of file, set paste, enter three lines, and save work (six steps).
99. Obtain a list of installed packages, format, and append to the *.sls file, and check the append in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

100. While already in vi, edit the end of the sls file using copy/paste to include the setup for checking for available but not installed packages:

```
purgedpkgs:
 pkg.purged:
  - pkgs:
```

101. Shift+g to end of file, set paste, enter three lines, and save work (six steps).
102. Obtain a list of available but not installed packages, format, and append to the *.sls file, then check the append in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

103. While already in vi, edit the end of the sls file using copy/paste to include the setup for checking for kernel modules:

```
mods:
 kmod.present:
  - mods:
```

104. Shift+g to end of file, set paste, enter three lines, and save work (six steps).
105. Obtain a list of loaded kernel modules, format, and append to the *.sls file and obtain a list of loaded kernel modules, format, and save it in a temporary file for later, and open the sls file to check the append in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

106. While already in vi, edit the end of the sls file using copy/paste to include the setup for checking for kernel modules that should not be loaded:

```
absentmods:
 kmod.absent:
  - mods:
```

107. Shift+g to end of file, set paste, enter three lines, and save work (six steps).
108. Obtain a list of available kernel modules, save it in a temporary file for later, and check the files to make sure the information was stored correctly in two steps using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

109. Use grep to compare the temporary files and obtain a list of kernel modules that are available but not currently loaded, format the output and append it to the *.sls file scroll to check that the grep executed correctly, and check the append to the sls file in one step using the history function, where <start> is the history line number to start at a,nd <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

110. While in vi, fix the numeric kernel modules so they have quotes.
111. While in vi, edit to check for only the latest kernel version.
112. Make a directory on the current host for the baseline files and copy a set of baseline files from SUSE Manager to the baseline host for salt to use, and check that the files copied properly in one step using the history function, where <start> is the history line number to start at and <stop> is the history line number to stop at.

```
eval "$(fc -ln <start> <stop>)"
```

113. Check that the new sls file executes successfully with salt:

```
salt tide2* state.apply
```

Creating drift

1. Uninstall alee-fonts package and rmmod ip6table kernel module from Node 0.
2. Uninstall arabic-bitmap-fonts package from Node 1.
3. Do a system update, and uninstall the bluez package from Node1.

Detecting drift

1. Navigate to SUSE Manager.
2. Log in.
3. Click Systems.
4. Click the first system.
5. Click States.
6. Click to turn on Test mode.
7. Click Apply Highstate.
8. Click Show full highstate output, and discover it doesn't give you the output that the salt state.appy command does. It tells what order the commands are submitted. Scroll through the output. (User experience error here. Omitting steps and time.)
9. Turn test mode off, and click Apply highstate again. (User experience error here. Omitting steps and times.)
10. When the output doesn't update, refresh the browser. (User experience error here. Omitting steps and times.)
11. Click Apply Highstate. (User experience error here. Omitting steps and times.)
12. Click Show full highstate output, and discover it yet again does not give the salt state.apply output as expected, instead it gives the PID of the process. (User experience error here. Omitting steps and times.)
13. Click Systems. (User experience error here. Omitting steps and times.)
14. Click the first system. (User experience error here. Omitting steps and times.)
15. Click States. (User experience error here. Omitting steps and times.)
16. Turn on Test mode. (User experience error here. Omitting steps and times.)
17. Click Apply Highstate. (User experience error here. Omitting steps and times.)
18. Click Applying the highstate has been scheduled.
19. Click Completed Systems, to which it says states No systems have completed this action.
20. Click In Progress Systems to see current system in progress.
21. Click Systems.
22. Click the second system.
23. Click States. (Note: We start the second system running in parallel).
24. Turn on test mode.
25. Click Apply Highstate.
26. Click Systems.
27. Click the third system.
28. Click States. (Note: We start third system running in parallel.)
29. Turn on test mode.
30. Click Apply Highstate.
31. Click Schedule.
32. Click the first in the list of three: Apply highstate in test-mode scheduled by SUSEManagerAdmin.
33. Click Completed Systems, to which it states No systems have completed this action.
34. Attempt to reload the page multiple times to see it update, until the first system appears under completed systems. (Omit machine time in totals.)
35. Click the first system.
36. Notice the results in the System History Event. Scroll to find cmd.run statements aren't run, but the other commands have test output indicating if anything needs to be fixed, that if these were executed without test mode, would be reverted to baseline state. Scroll through to find that if you executed a return to baseline, it would re-install a missing package alee-fonts and reload the ip6table_security module. Also notice that non-executed commands and/or commands that came back false.Commands that require changes are red, while commands requiring no changes are black. (Omit reading time.)
37. Click Systems.
38. Click the second system.
39. Click Events.
40. Click History, since there are no pending events.
41. Click Apply highstate in test-mode scheduled by SUSEManagerAdmin.
42. Scroll through the results, note that to return to baseline, SALT would reinstall the bluez package, which would bring back the Bluetooth service, all in red font indicating deviations from the baseline. (Omit reading time.)
43. Click Overview.
44. Click the third system.
45. Click Events.
46. Click History, since there are no pending events.
47. Click Apply highstate in test-mode scheduled by SUSEManagerAdmin.
48. Scroll through the results, note that to return to baseline after a system update, multiple packages would need to be reverted, including bluez package/Bluetooth service as indicated by red font. (Omit reading time.)

Use case: Compliance (Red Hat)

1. Navigate to console.redhat.com/insights.
2. Log in.
3. Click Compliance.
4. Click SCAP Policies.
5. Click Create New Policy.
6. Click RHEL 8.
7. Scroll down the list of standards looking for PCI-DSS, and press the > for the next page.
8. Click the PCI-DSS radio button.
9. Click Next.
10. In the textbox, enter the Business objective.
11. In the textbox, enter the Compliance threshold percent.
12. Click Next.
13. Above the Name column, click the checkbox to select all three systems.
14. Click Next.
15. Click Next.
16. Click Finish.
17. Click Return to application.
18. Open an SSH connection to the first host.
19. Open an SSH connection to the second host.
20. Open an SSH connection to the third host.
21. Start an Insights compliance run on the first host:

```
insights-client --compliance
```

22. Start an Insights compliance run on the second host:

```
insights-client --compliance
```

23. Start an Insights compliance run on the third host:

```
insights-client --compliance
```

24. Wait for completion.
25. Click Reports.
26. Click PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 8.
27. See that Insights isn't done getting all the results. Click refresh until all the results are in.
28. Click the first host.
29. Click the CCE-80862-6 "Ensure System Log Files Have Correct Permissions checkbox.
30. Click Remediate.
31. In the Create new playbook textbox, enter a playbook name.
32. Click Next.
33. Click Next.
34. Click Submit.
35. Click on the new playbook.
36. Click Execute playbook.
37. Click Execute playbook on 1 system, and wait for execution to complete.
38. Click Compliance.
39. Click Reports.
40. Click PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 8.
41. Click the next host.
42. Click the CCE-80862-6 "Ensure System Log Files Have Correct Permissions checkbox.
43. Click Remediate.
44. In the Create new playbook textbox, enter a playbook name.

45. Click Next.
46. Click Next.
47. Click Submit.
48. Click on the new playbook.
49. Click Execute playbook.
50. Click Execute playbook on 1 system, and wait for execution to complete.
51. Click Compliance.
52. Click Reports.
53. Click PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 8.
54. Click on the next host.
55. Click theCCE-80862-6 "Ensure System Log Files Have Correct Permissions checkbox.
56. Click Remediate.
57. In the Create new playbook textbox, enter a playbook name.
58. Click Next.
59. Click Next.
60. Click Submit.
61. Click on the new playbook.
62. Click Execute playbook.
63. Click Execute playbook on 1 system, and wait for execution to complete.
64. Click Compliance.
65. Click Reports.
66. Click PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 8.
67. Click on the first host.
68. Click the arrow next to CCE-80674-5 Ensure All Accounts on the System Have Unique Names, and notice that it does not have an Ansible playbook.
69. Open an SSH connection to the first host.
70. Open /etc/passwd and find duplicate lines:

```
cat /etc/passwd
```

71. Attempt userdel on the extra user name. The error message will say to use pwck to fix the multiple entries. (Omit user experience/error time and steps.)

```
userdel user2
```

72. Use pwck, tell it to delete the duplicate lines:

```
pwck
```

73. Click Compliance.
74. Click Reports.
75. Click PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 8.
76. Click on the next host.
77. Notice this host also has CCE-80674-5 Ensure All Accounts on the System Have Unique Names unsatisfied.
78. Click Compliance.
79. Click Reports.
80. Click PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 8.
81. Click on the next host.
82. Notice this host also has CCE-80674-5 Ensure All Accounts on the System Have Unique Names unsatisfied.
83. Open an SSH connection to the second host.
84. Use pwck, and tell it to delete the duplicate lines:

```
pwck
```

85. Open an SSH connection to the third host.
86. Use pwck, and tell it to delete the duplicate lines:

```
pwck
```

Use case: Compliance (SUSE)

1. Navigate to SUSE Manager.
2. Log in.
3. Click Overview.
4. Click the first host.
5. Click Audit.
6. Click Schedule.
7. In the textbox, enter the command line arguments for oscap:

```
--profile xcdf_org.ssgproject.content_profile_pci-dss
```

8. In the textbox, enter the path for XCCDF for PCI-DSS:

```
/usr/share/xml/scap/ssg/content/ssg-sle15-ds.xml
```

9. Click Schedule.
10. Click Systems.
11. Click the second host.
12. Click Audit.
13. Click Schedule.
14. In the textbox, enter the command line arguments for oscap:

```
--profile xcdf_org.ssgproject.content_profile_pci-dss
```

15. In the textbox, enter the path for XCCDF for PCI-DSS:

```
/usr/share/xml/scap/ssg/content/ssg-sle15-ds.xml
```

16. Click Schedule.
17. Click Systems.
18. Click the third host.
19. Click Audit.
20. Click Schedule. In the textbox, enter the command line arguments for oscap:

```
--profile xcdf_org.ssgproject.content_profile_pci-dss
```

21. In the textbox, enter the path for XCCDF for PCI-DSS:

```
/usr/share/xml/scap/ssg/content/ssg-sle15-ds.xml
```

22. Click Schedule.
23. Click Events.
24. Click History.
25. Click Refresh until it indicates the OpenScap scan is done.
26. Click Audit.
27. Click on the latest xccdf_org_testresult_xccdf_org.ssgproject.content_profile_pci-dss scan report.
28. Note the CCE-85845-6 xccdf_org.ssgproject.content_rule_acount_unique_name fail result, and click on it.
29. Note that no details are given, only a link to https://nvd.nist.gov/cce/index.cfm, which is not a working hyperlink.
30. Copy the text and paste it in a browser window, and find that the link is broken.
31. Click on the NCP menu that the broken link provided.
32. Click CCE, and get a list of CCE downloads.
33. Click CCE v5 - SUSE Linux Enterprise Server 15.
34. Open the downloaded file in Excel.

35. Fix the column width to be able to read properly.
36. Search for 85845.
37. Read through the information for the CCE Ensure All Accounts on the System Have Unique Names (omit reading time).
38. To enable copy/paste, click Enable Editing.
39. Copy the diagnostic command from the text.
40. Return to SUSEManager.
41. Click Salt.
42. Click Remote Commands.
43. Paste the diagnostic command.
44. Click Find targets.
45. Click Run command.
46. Click Show response for each host (three steps), and see that each has a duplicate user2 detected.
47. In the command textbox, enter `userdel user2` (omit this user experience error time and steps).
48. Click Run command. Notice that the error message says to fix this with pwck. (Omit this user experience error time and steps.)
49. In the command textbox, enter `pwck` (omit this user experience error time and steps).
50. Click Run command. Notice that the command needed user input and did not run properly without it. (Omit this user experience error time and steps.)
51. Open an SSH connection to the first host.
52. Use `pwck`, and tell it to delete the duplicate lines.
53. Click Systems.
54. Click the second host.
55. Click Audit.
56. Click the latest xccdf_org_testresult_xccdf_org.ssgproject.content_profile_pci-dss scan report.
57. Note the CCE-85845-6 xccdf_org.ssgproject.content_rule_acount_unique_name failure.
58. Click Systems.
59. Click the last host.
60. Click Audit.
61. Click the latest xccdf_org_testresult_xccdf_org.ssgproject.content_profile_pci-dss scan report.
62. Note the CCE-85845-6 xccdf_org.ssgproject.content_rule_acount_unique_name failure.
63. Open an SSH connection to the second host.
64. Use `pwck`, and tell it to delete the duplicate lines.
65. Open an SSH connection to the third host.
66. Use `pwck`, and tell it to delete the duplicate lines.
67. Click Systems.
68. Click the first host.
69. Click Audit.
70. Click the latest xccdf_org_testresult_xccdf_org.ssgproject.content_profile_pci-dss scan report.
71. Note the CCE-85811-8 xccdf_org.ssgproject.content_rule_file_permissions_var_log_audit failure result and click on it, noticing that it gives the same links as before.
72. Click Systems.
73. Click the second host.
74. Click Audit.
75. Click the latest xccdf_org_testresult_xccdf_org.ssgproject.content_profile_pci-dss scan report
76. Note the CCE-85811-8 xccdf_org.ssgproject.content_rule_file_permissions_var_log_audit failure result again.
77. Click Systems.
78. Click the last host.
79. Click Audit.
80. Click the latest xccdf_org_testresult_xccdf_org.ssgproject.content_profile_pci-dss scan report.
81. Note the CCE-85811-8 xccdf_org.ssgproject.content_rule_file_permissions_var_log_audit failure result again.
82. Go back to the Excel file.
83. Search for 85811-8, and read the instructions.
84. Copy /etc/audit/auditd.conf from file to check permissions.
85. Go back to SUSEManager.
86. Click Salt.
87. Click Remote Commands.
88. In the textbox, enter `ls -l /etc/audit/auditd.conf`
89. Click Find targets.

90. Click Run command.
91. Click Show response for each host (three steps), and see the permissions on each system.
92. Go back to the Excel file, and read the instructions.
93. Go back to SUSEManager, and enter `ls -l on the /var/log/audit` files.
94. Click Run command.
95. See that the permissions are wrong on all systems. In the textbox, enter `chmod 600 /var/log/audit`
96. Click Run command.
97. In the textbox,enter `chmod 600 /var/log/audit/*`
98. Click Run command.

Use case: Vulnerability (Red Hat)

1. Navigate to console.redhat.com/insights.
2. Log in.
3. Click Vulnerability.
4. Click CVEs.
5. Click the CVE to remediate.
6. Click the provided link for information.
7. Read through the 1.75 pages of text.
8. Select all affected systems.
9. Click Remediate.
10. Type in a name for the remediation playbook.
11. Click Next.
12. Click Next.
13. Click Submit.
14. Click Open Playbook.
15. Click Execute Playbook.
16. Click Execute playbook on 3 systems.
17. Wait for remediation.

Use case: Vulnerability (SUSE)

1. Navigate to SUSE Manager.
2. Log in.
3. Click Patches.
4. Click Security Patches.
5. Click the Security Advisory that this use case is about.
6. Read and click the provided links for information (This Security Patch has 1 associated CVE).
7. Click Packages, and click View.
8. Click Affected Systems, and click View.
9. Read through the short text in cve.mitre.org provided for the CVE.
10. Click Select All.
11. Click Apply Patches.
12. Click Confirm.
13. Click Schedule.
14. Click the appropriate patch update.
15. Click In Progress Systems.
16. Refresh until all systems disappear from list of In Progress systems.
17. Click Completed Systems, and ensure all systems are on the list.

Use case: Bugfix (Red Hat)

1. Navigate to console.redhat.com/insights.
2. Log in.
3. Click Patch.
4. Click Advisories.
5. Click the RHBA that this use case is about.
6. Read through the half page of text.
7. Select all affected systems.
8. Click Remediate.
9. Enter a name for the remediation playbook.
10. Click Next.
11. Click Next.
12. Click Submit.
13. Click Open Playbook.
14. Click Execute Playbook.
15. Click Execute playbook on 3 systems.
16. Wait for remediation.

Use case: Bugfix (SUSE)

1. Navigate to SUSE Manager.
2. Log in.
3. Click Patches.
4. Click Bugfix Patches.
5. Click the patch that this use case is about.
6. Scroll, and click the provided Vendor Advisory link.
7. Scan through the Vendor Advisory.
8. Click Packages, and click View.
9. Click Affected Systems, and click View.
10. Click Select All.
11. Click Apply Patches.
12. Click Confirm.
13. Click Schedule.
14. Click the appropriate patch update.
15. Click In Progress Systems.
16. Refresh until all systems disappear from list of In Progress systems.
17. Click Completed Systems, and ensure that all systems are listed.

Read the report at https://facts.pt/emX9abL ▶

This project was commissioned by Red Hat.

Principled Technologies®

Facts matter.®