Lifecycle Management

NetApp Solutions SAP

NetApp
September 17, 2024

Table of Contents

Lifecycle Management

NetApp SAP Landscape Management Integration using Ansible

TR-4953: NetApp SAP Landscape Management Integration using Ansible

Michael Schlosser, Nils Bauer, NetApp

SAP Landscape Management (LaMa) enables SAP system administrators to automate SAP system operations, including end-to-end SAP system clone, copy, and refresh operations.

NetApp offers a rich set of Ansible modules that allows SAP LaMa to access technologies such as NetApp Snapshot and FlexClone through SAP LaMa Automation Studio. These technologies help to simplify and accelerate SAP system clone, copy, and refresh operations.

The integration can be used by customers who run NetApp storage solutions on-premises or by customers using NetApp storage services at public cloud providers such as Amazon Web Services, Microsoft Azure, or Google Cloud Platform.

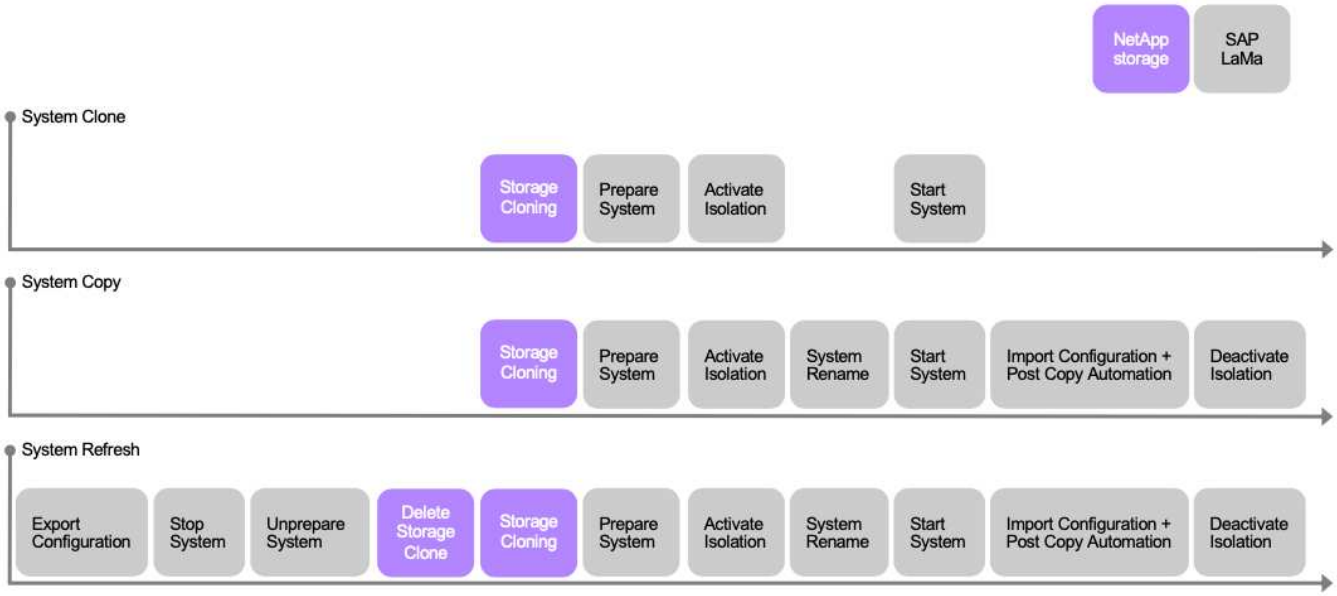
This document describes the configuration of SAP LaMa with NetApp storage features for SAP system copy, clone, and refresh operations using Ansible automation.

SAP system clone, copy, and refresh scenarios

The term SAP system copy is often used as a synonym for three different processes: SAP system clone, SAP system copy, or SAP system refresh. It is important to distinguish between the different operations because the workflows and use cases differ for each one.

- **SAP system clone.** An SAP system clone is an identical clone of a source SAP system. SAP system clones are typically used to address logical corruption or to test disaster recovery scenarios. With a system clone operation, the hostname, instance number, and SID remain the same. It is therefore important to establish proper network fencing for the target system to make sure that there is no communication with the production environment.
- **SAP system copy.** An SAP system copy is a setup of a new target SAP system with data from a source SAP system. The new target system could be, for example, an additional test system with data from the production system. The hostname, instance number, and SID are different for the source and target systems.
- **SAP system refresh.** An SAP system refresh is a refresh of an existing target SAP system with data from a source SAP system. The target system is typically part of an SAP transport landscape, for example a quality assurance system, that is refreshed with data from the production system. The hostname, instance number, and SID are different for the source and target systems.

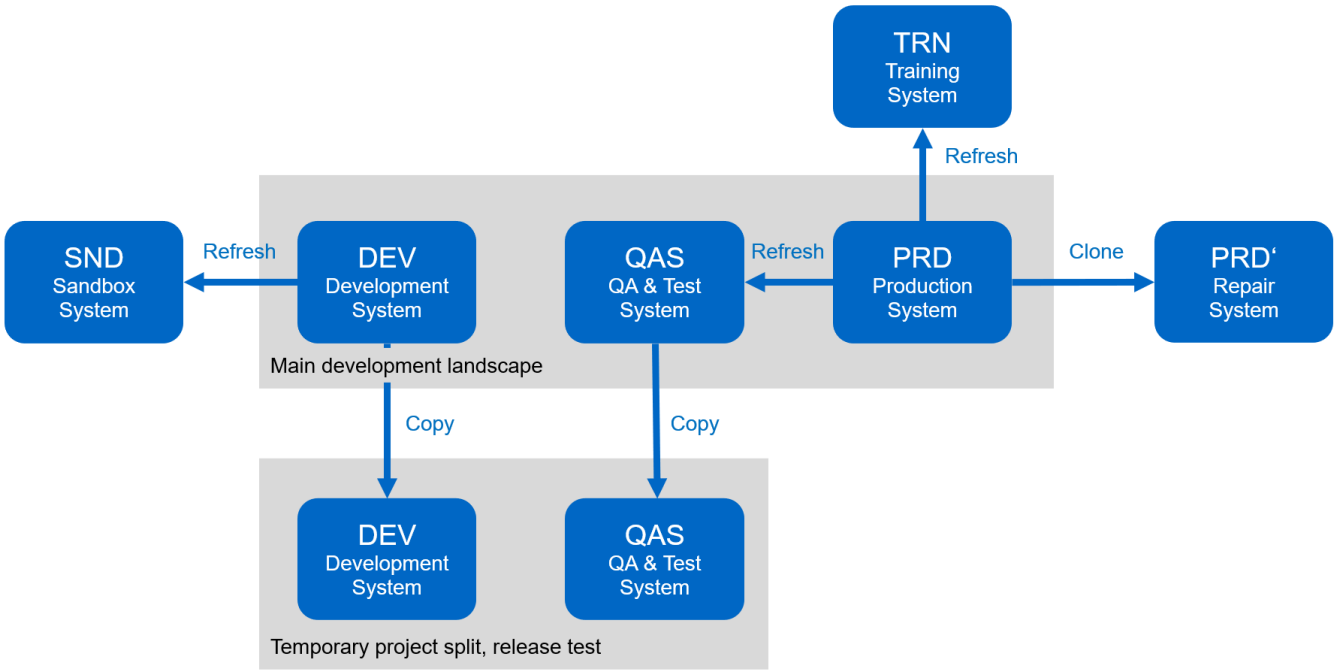
The following figure illustrates the main steps that must be performed during a system clone, system copy, or system refresh operation. The purple boxes indicate steps where NetApp storage features can be integrated. All three operations can be fully automated by using SAP LaMa.



Use cases for system refresh, copy, and cloning

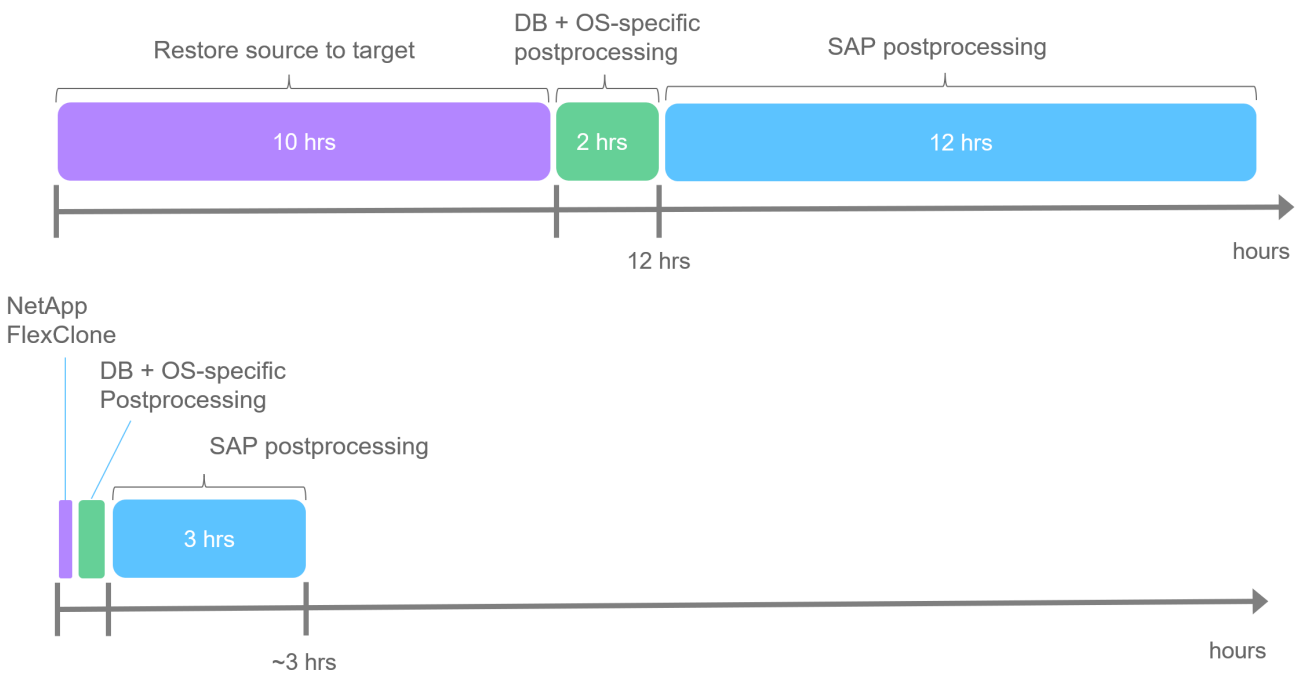
There are multiple scenarios in which data from a source system must be made available to a target system for testing or training purposes. These test and training systems must be updated with data from the source system on a regular basis to make sure that testing and training is performed with the current data set.

These system refresh operations consist of multiple tasks on the infrastructure, database, and application layers, and they can take multiple days depending on the level of automation.



SAP LaMa and NetApp cloning workflows can be used to accelerate and automate the required tasks at the infrastructure and database layers. Instead of restoring a backup from the source system to the target system,

SAP LaMa uses NetApp Snapshot copy and NetApp FlexClone technology so that required tasks up to a started HANA database can be performed in minutes instead of hours as shown in the following figure. The time needed for the cloning process is independent from the size of the database; therefore even very large systems can be created in a couple of minutes. Further reduction of the runtime is accomplished by automating tasks on the operating system and database layer as well as on the SAP post processing side.



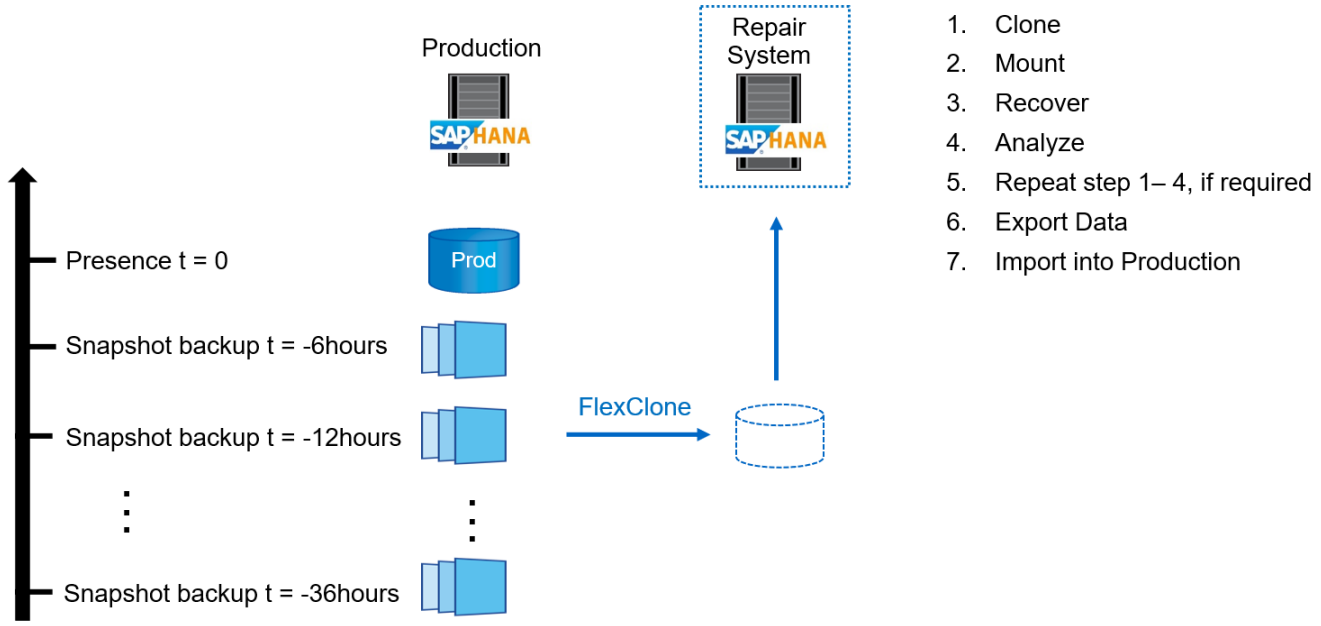
Address logical corruption

Logical corruption can be caused by software errors, human errors, or sabotage. Unfortunately, logical corruption often cannot be addressed with standard high-availability and disaster recovery solutions. As a result, depending on the layer, application, file system, or storage where the logical corruption occurred, minimal downtime and acceptable data loss requirements can sometimes not be fulfilled.

The worst case is logical corruption in an SAP application. SAP applications often operate in a landscape in which different applications communicate with each other and exchange data. Therefore, restoring and recovering an SAP system in which a logical corruption has occurred is not the recommended approach. Restoring the system to a point in time before the corruption occurred results in data loss. Also, the SAP landscape would no longer be in sync and would require additional postprocessing.

Instead of restoring the SAP system, the better approach is to try to fix the logical error within the system by analyzing the problem in a separate repair system. Root cause analysis requires the involvement of the business process and application owner. For this scenario, you create a repair system (a clone of the production system) based on data stored before the logical corruption occurred. Within the repair system, the required data can be exported and imported into the production system. With this approach, the production system does not need to be stopped, and, in the best-case scenario, no data or only a small fraction of data is lost.

When setting up the repair system, flexibility and speed are crucial. With NetApp storage-based Snapshot backups, multiple consistent database images are available to create a clone of the production system by using NetApp FlexClone technology. FlexClone volumes can be created in a matter of seconds rather than multiple hours if a redirected restore from a file-based backup is used to set up the repair system.

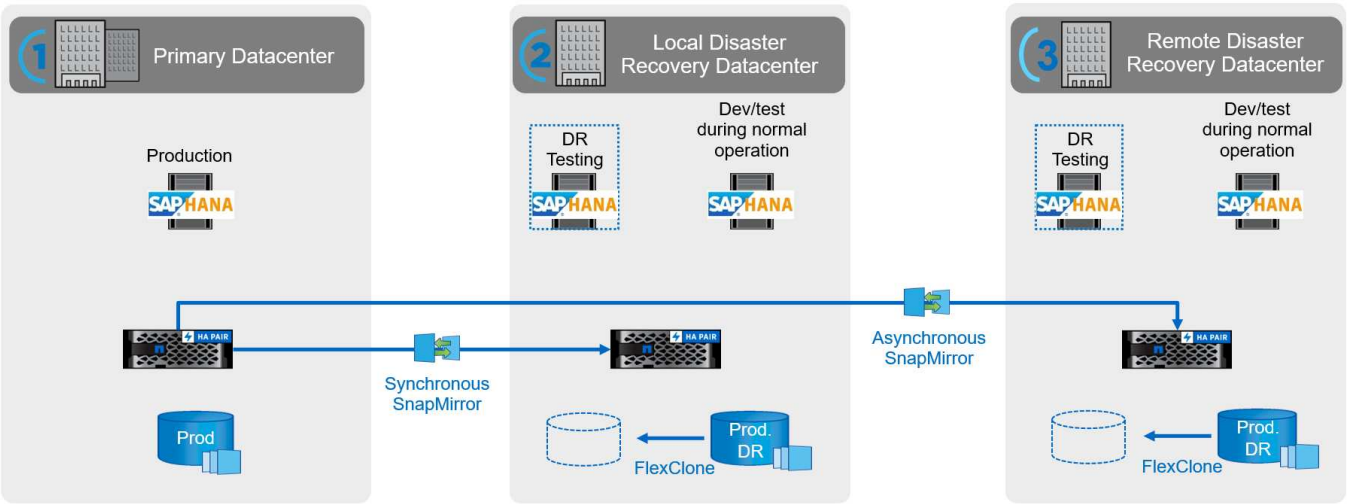


Disaster recovery testing

An effective disaster recovery strategy requires testing the required workflow. Testing demonstrates whether the strategy works and whether the internal documentation is sufficient. It also allows administrators to train on the required procedures.

Storage replication with SnapMirror makes it possible to execute disaster recovery testing without putting RTO and RPO at risk. Disaster recovery testing can be performed without interrupting data replication. Disaster recovery testing for both asynchronous and synchronous SnapMirror uses Snapshot backups and FlexClone volumes at the disaster recovery target.

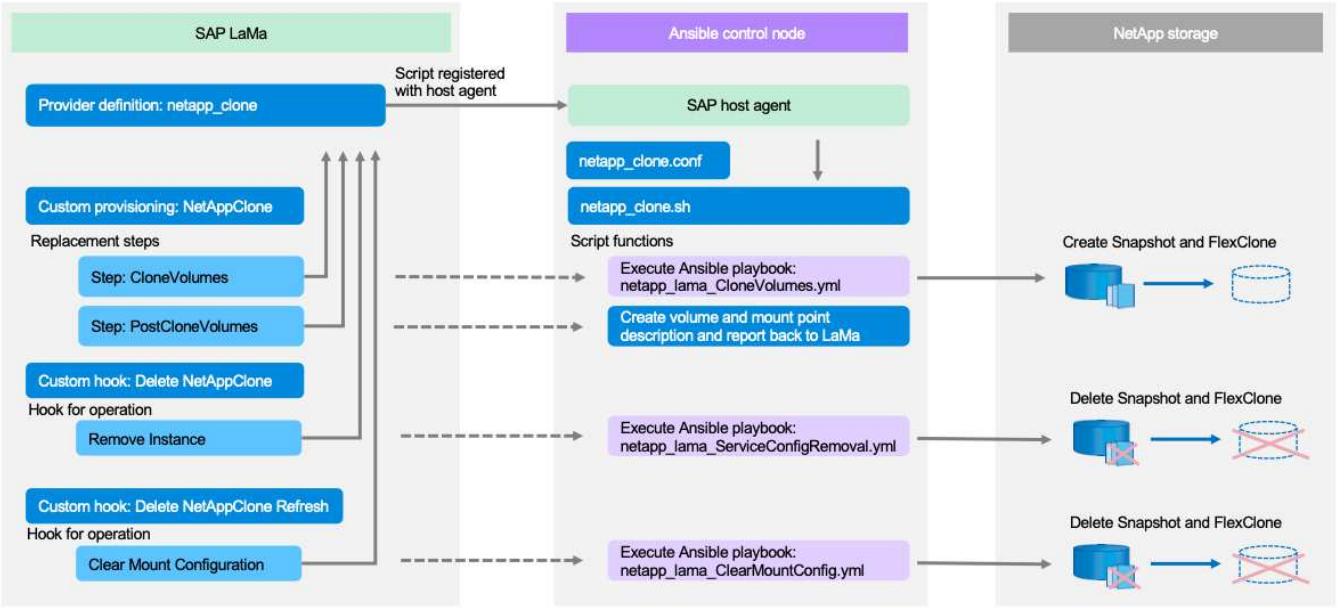
SAP LaMa can be used to orchestrate the entire testing procedure, and it also takes care of network fencing, target host maintenance, and so on.



NetApp SAP LaMa integration using Ansible

The integration approach uses SAP LaMa custom provisioning and operation hooks

combined with Ansible playbooks for NetApp storage management. The following figure shows a high-level overview of the configuration on the LaMa side as well as the corresponding components of the example implementation.



A central host acting as an Ansible control node is used to execute the requests from SAP LaMa and to trigger the NetApp storage operations using Ansible playbooks. The SAP host agent components must be installed on this host so that the host can be used as a communication gateway to SAP LaMa.

Within LaMa Automation Studio, a provider is defined that is registered at the Ansible host's SAP host agent. A host agent configuration file points to a shell script that is called by SAP LaMa with a set of command line parameters, depending on the requested operation.

Within LaMa Automation Studio, custom provisioning and a custom hook is defined to execute storage cloning operations during provisioning and also during clean-up operations when the system is deprovisioned. The shell script on the Ansible control node then executes the corresponding Ansible playbooks, which trigger the Snapshot and FlexClone operations as well as the deletion of the clones with the deprovisioning workflow.

More information on NetApp Ansible modules and the LaMa provider definitions can be found at:

- NetApp Ansible modules
- SAP LaMa documentation – provider definitions

Example implementation

Due to the large number of options available for system and storage setups, the example implementation should be used as a template your individual system setup and configuration requirements.

> The example scripts are provided as is and are not supported by NetApp. You can request the current version of the scripts via email to ng-sapcc@netapp.com.

Validated configurations and limitations

The following principles were applied to the example implementation and might need to be adapted to meet

customer needs:

- Managed SAP systems used NFS to access NetApp storage volumes and were set up based on the adaptive design principle.
- You can use all ONTAP releases supported by NetApp Ansible modules (ZAPI and REST API).
- Credentials for a single NetApp cluster and SVM were hard coded as variables in the provider script.
- Storage cloning was performed on the same storage system that was used by the source SAP system.
- Storage volumes for the target SAP system had the same names as the source with an appendix.
- No cloning at secondary storage (SV/SM) was implemented.
- FlexClone split was not implemented.
- Instance numbers were identical for the source and target SAP systems.

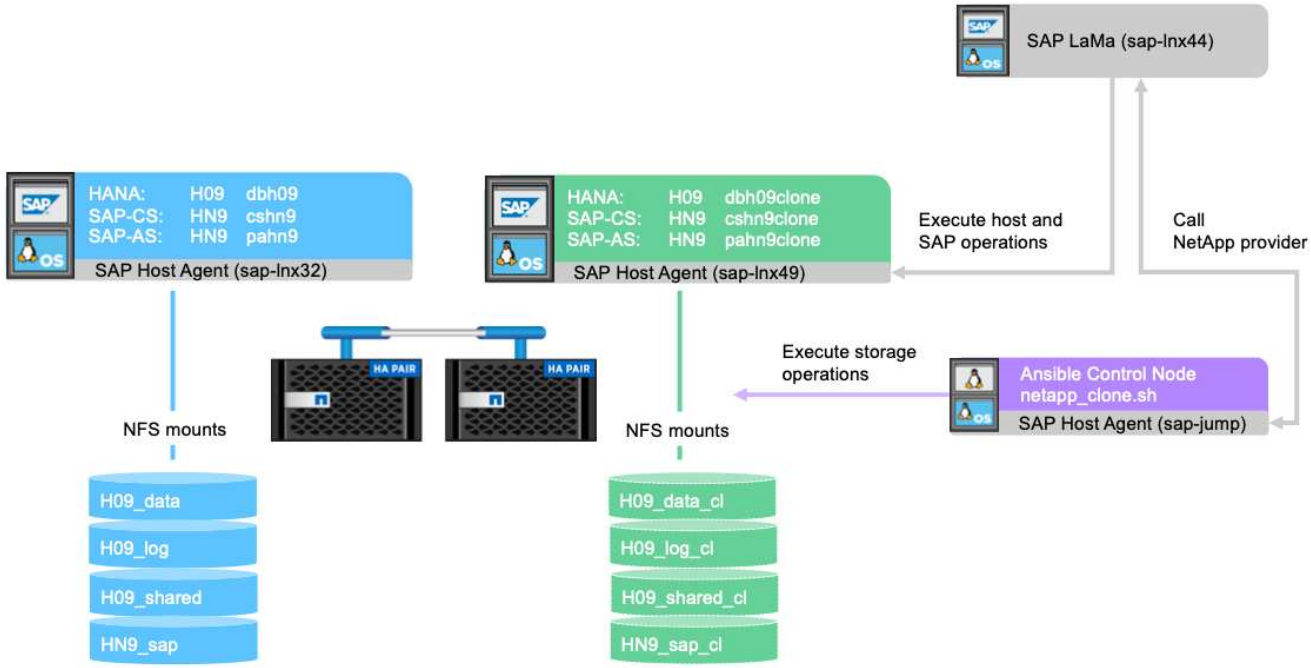
Lab setup

The following figure shows the lab setup we used. The source SAP system HN9 used for the system clone operation consisted of the database H09, the SAP CS, and the SAP AS services running on the same host (sap-lnx32) with installed adaptive design enabled. An Ansible control node was prepared according to the Ansible Playbooks for NetApp ONTAP documentation.

The SAP host agent was installed on this host as well. The NetApp provider script as well as the Ansible playbooks were configured on the Ansible control node as described in the "Appendix: Provider Script Configuration."

The host `sap-lnx49` was used as the target for the SAP LaMa cloning operations, and the isolation-ready feature was configured there.

Different SAP systems (HNA as source and HN2 as target) were used for system copy and refresh operations, because Post Copy Automation (PCA) was enabled there.



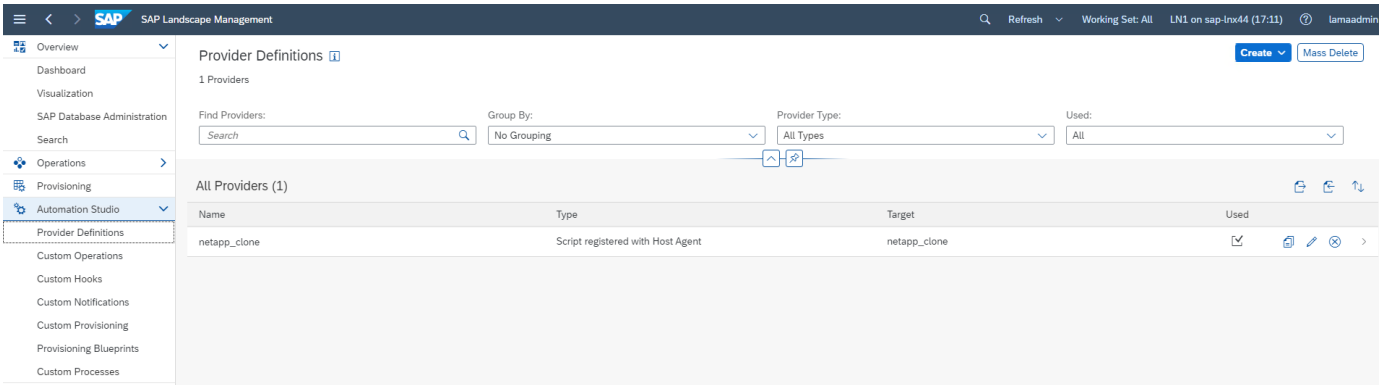
The following software releases were used in the lab setup:

- SAP LaMa Enterprise Edition 3.00 SP23_2
- SAP HANA 2.00.052.00.1599235305
- SAP 7.77 Patch 27 (S/4 HANA 1909)
- SAP Host Agent 7.22 Patch 56
- SAPACEXT 7.22 Patch 69
- Linux SLES 15 SP2
- Ansible 2. 13.7
- NetApp ONTAP 9.8P8

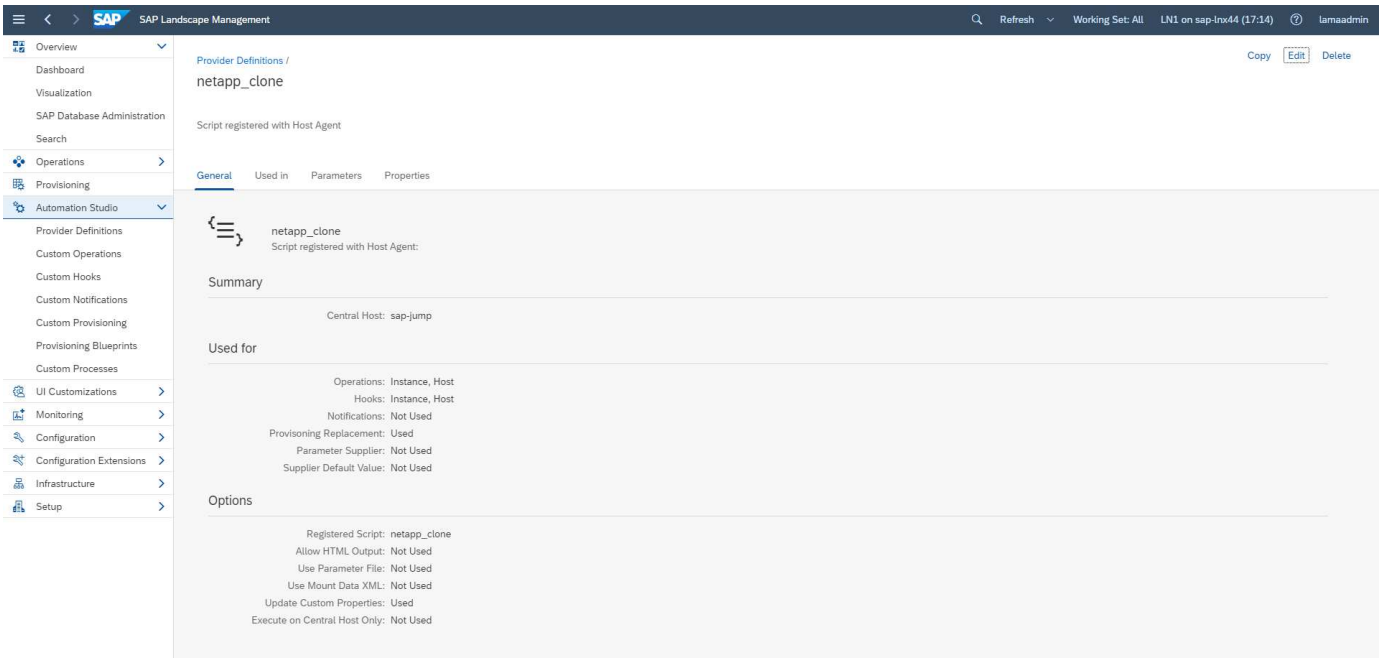
SAP LaMa configuration

SAP LaMa provider definition

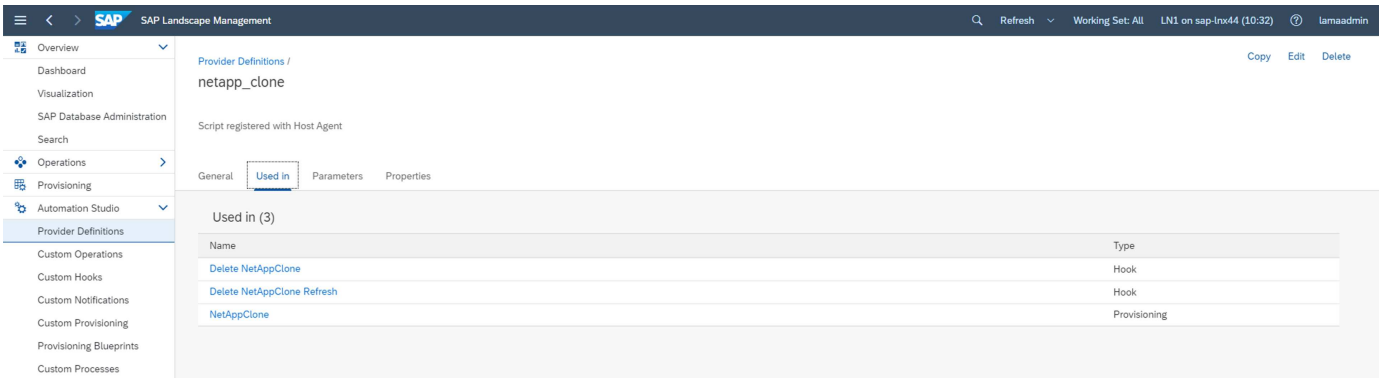
The provider definition is performed within Automation Studio of SAP LaMa as shown in the following screenshot. The example implementation uses a single provider definition that is used for different custom provisioning steps and operation hooks as explained before.



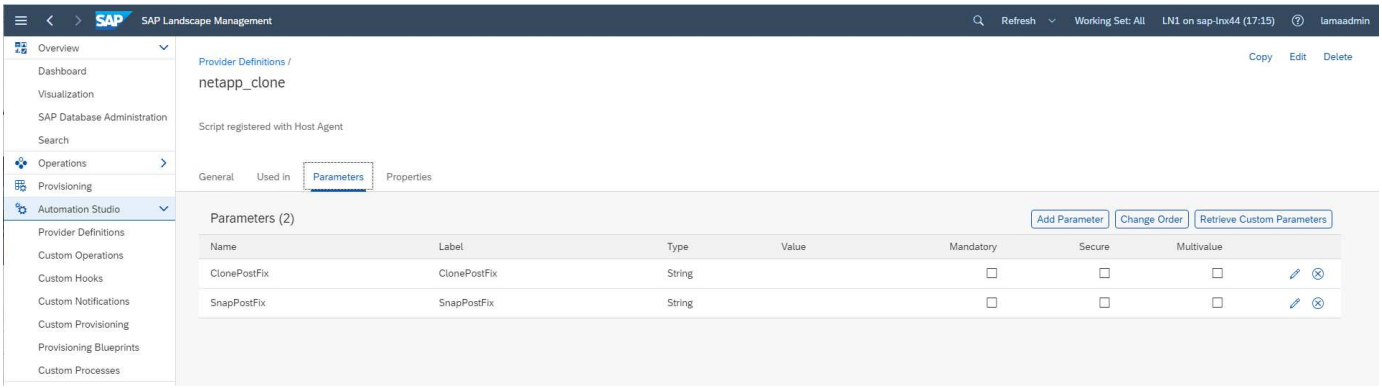
The provider `netapp_clone` is defined as the script `netapp_clone.sh` registered at the SAP host agent. The SAP host agent runs on the central host `sap-jump`, which also acts as the Ansible control node.



The **Used in** tab shows which custom operations the provider is used for. The configuration for the custom provisioning **NetAppClone** and the custom hooks **Delete NetAppClone** and **Delete NetAppClone Refresh** are shown in the next chapters.

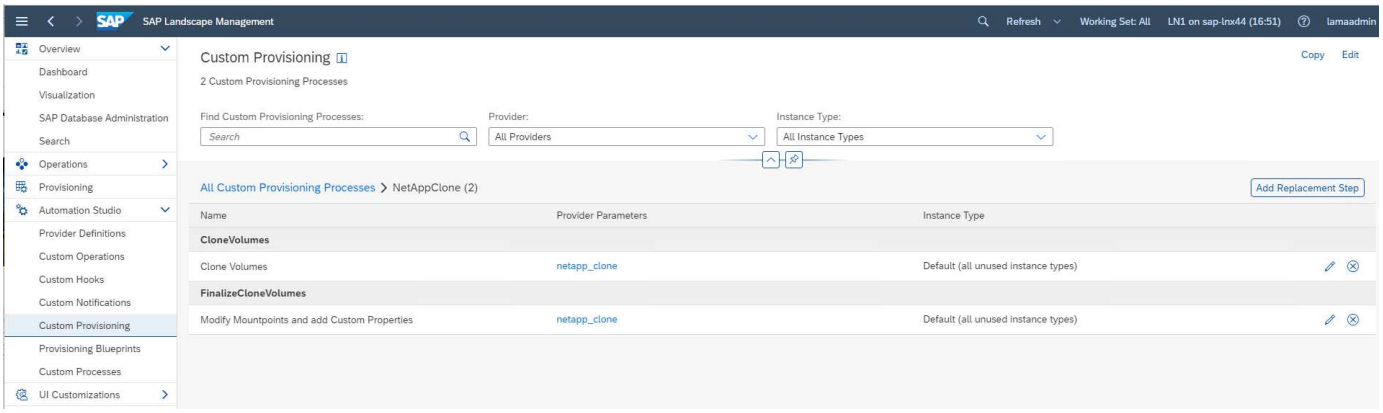


The parameters **ClonePostFix** and **SnapPostFix** are requested during the execution of the provisioning workflow and are used for the Snapshot and FlexClone volume names.



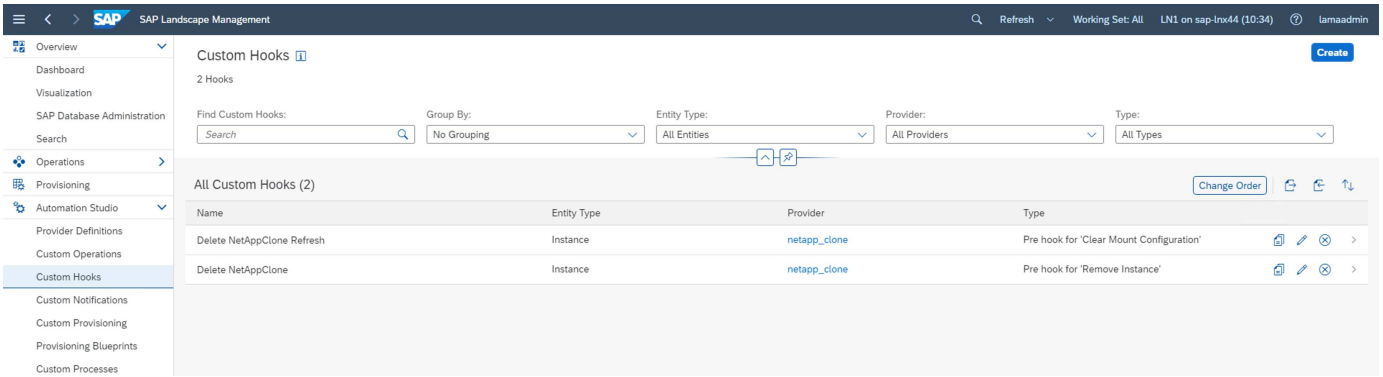
SAP LaMa custom provisioning

In the SAP LaMa custom provisioning configuration, the customer provider described before is used to replace the provisioning workflow steps **Clone Volumes** and **PostCloneVolumes**.



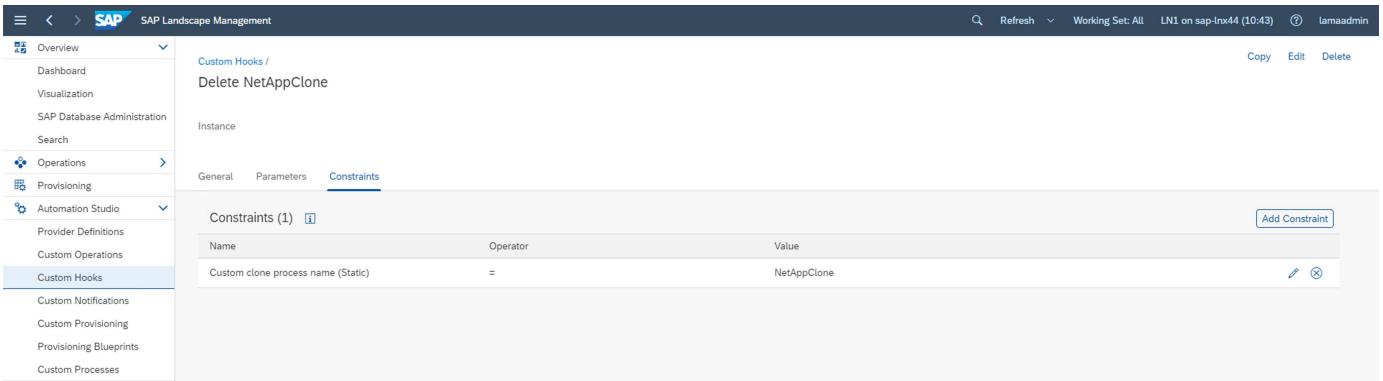
SAP LaMa custom hook

If a system is deleted with the system destroy workflow, the hook **Delete NetAppClone** is used to call the provider definition `netapp_clone`. The **Delete NetApp Clone Refresh** hook is used during the system refresh workflow because the instance is preserved during the execution.



It is important to configure **Use Mount Data XML** for the custom hook, so that SAP LaMa provides the information of the mount point configuration to the provider.

To ensure that the custom hook is only used and executed when the system was created with a custom provisioning workflow, the following constraint is added to it.

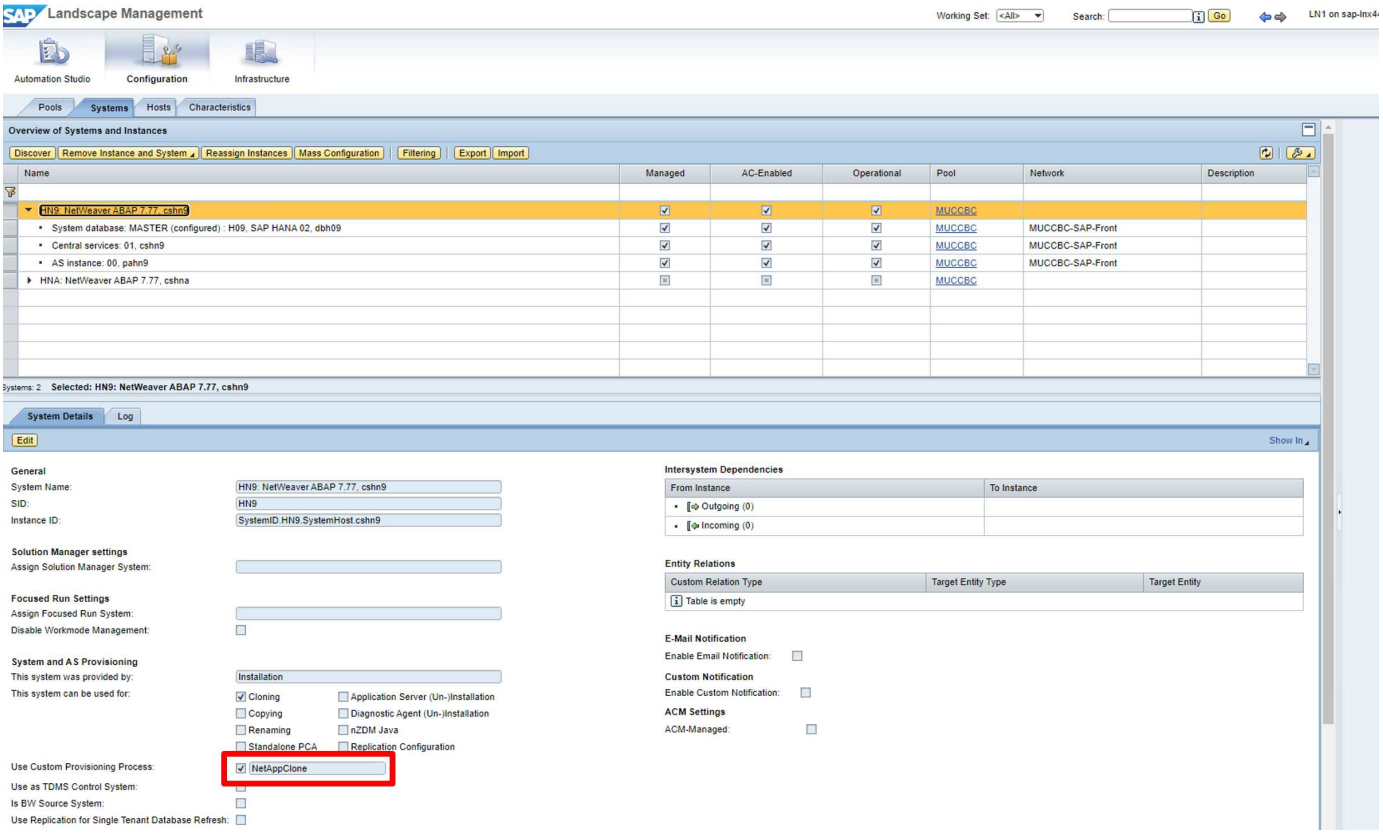


More information about the use of custom hooks can be found in the SAP LaMa Documentation.

Enable custom provisioning workflow for SAP source system

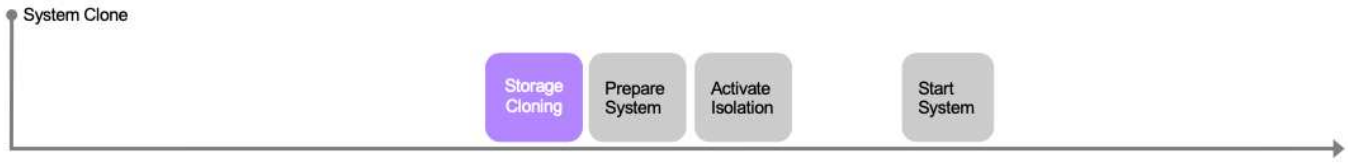
To enable the custom provisioning workflow for the source system, it must be adapted in the configuration. The **Use Custom Provisioning Process** checkbox with the corresponding custom provisioning definition must be

selected.

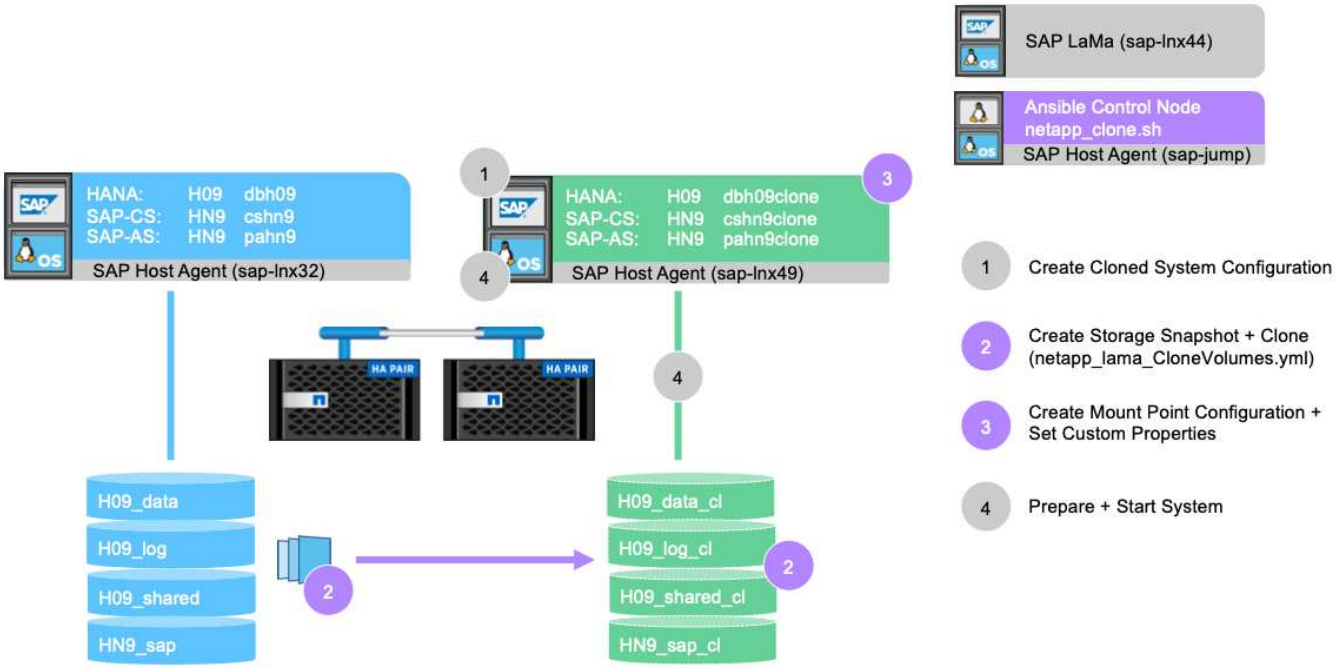


SAP LaMa provisioning workflow - clone system

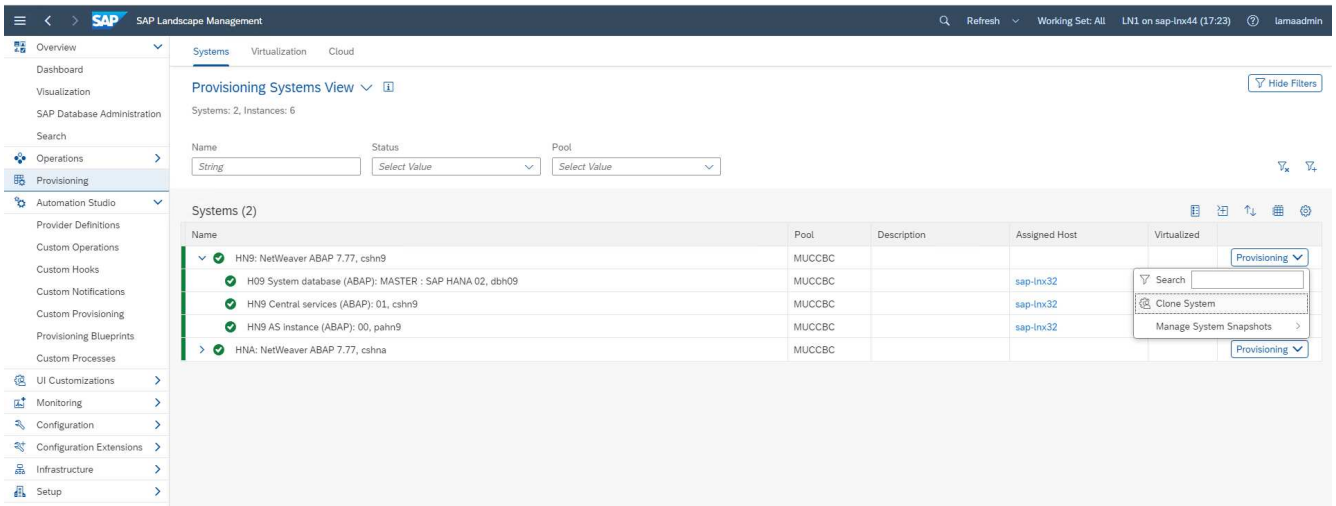
The following figure highlights the main steps executed with the system clone workflow.



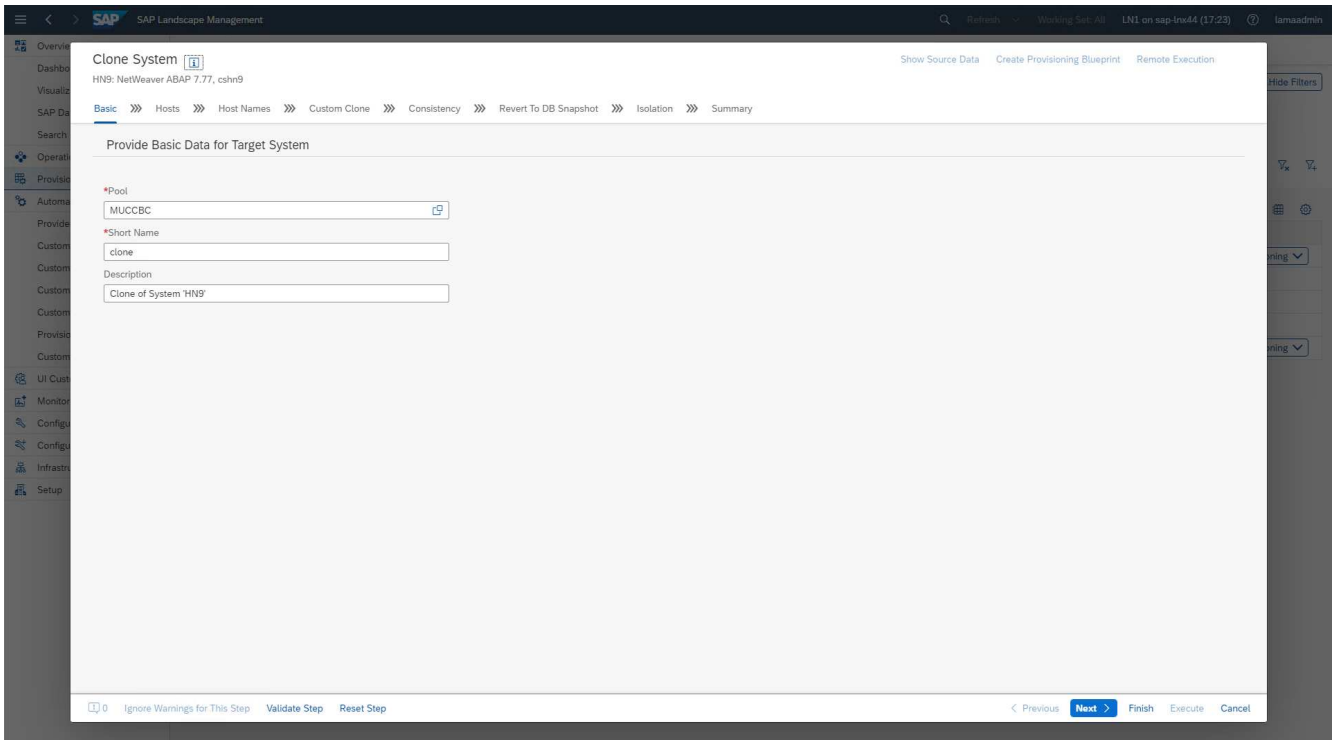
In this section, we go through the complete SAP LaMa system cloning workflow based on the source SAP system HN9 with HANA database H09. The following picture gives an overview of the steps executed during the workflow.



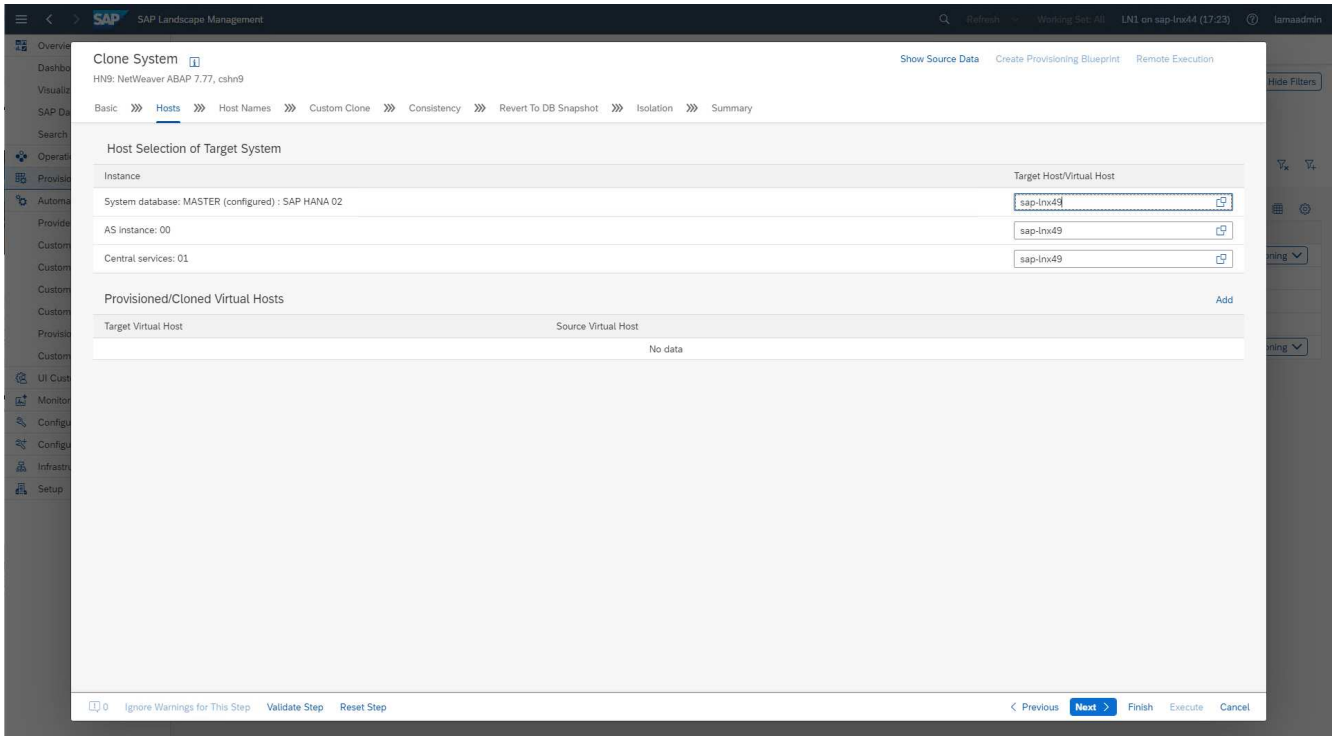
1. To start the cloning workflow, open **Provisioning** in the menu tree and select the source system (in our example HN9). Then start the **Clone System** wizard.



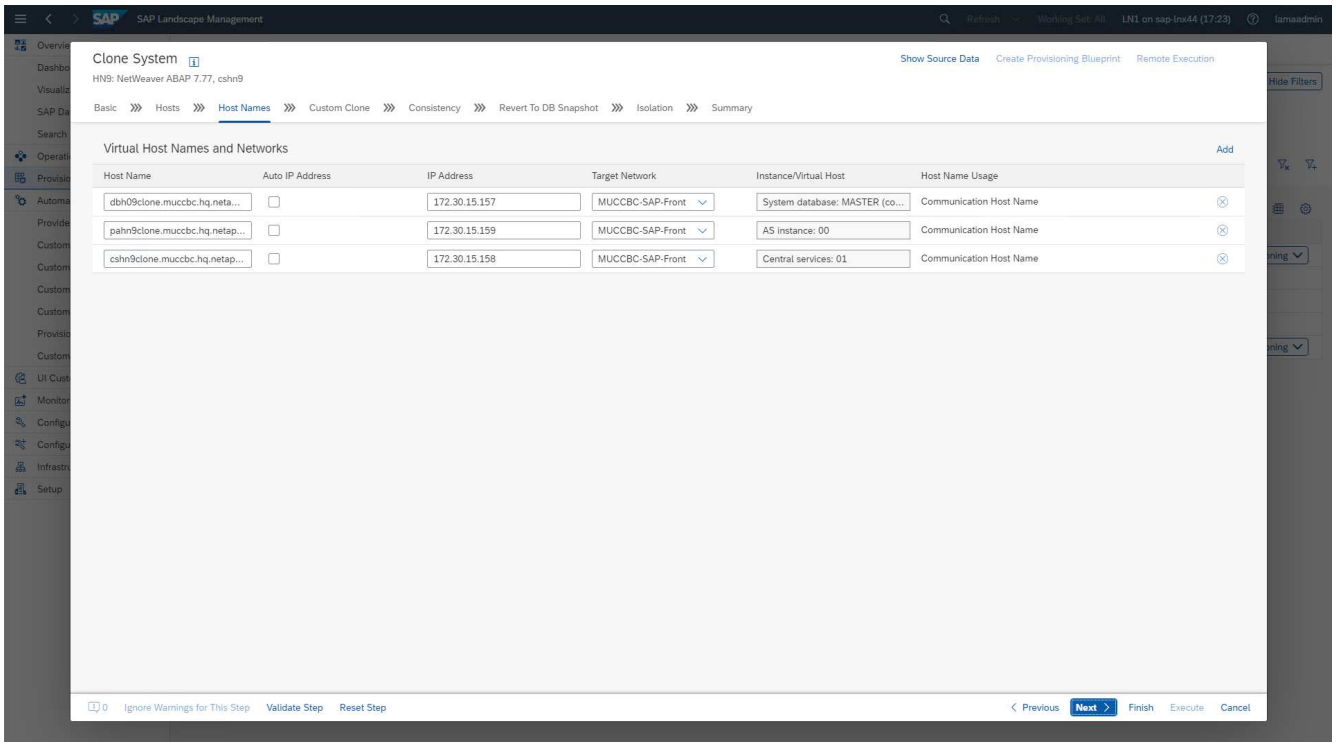
2. Enter the requested values. Screen 1 of the wizard asks for the pool name for the cloned system. This step specifies the instances (virtual or physical) on which the cloned system will be started. The default is to clone the system into the same pool as the target system.



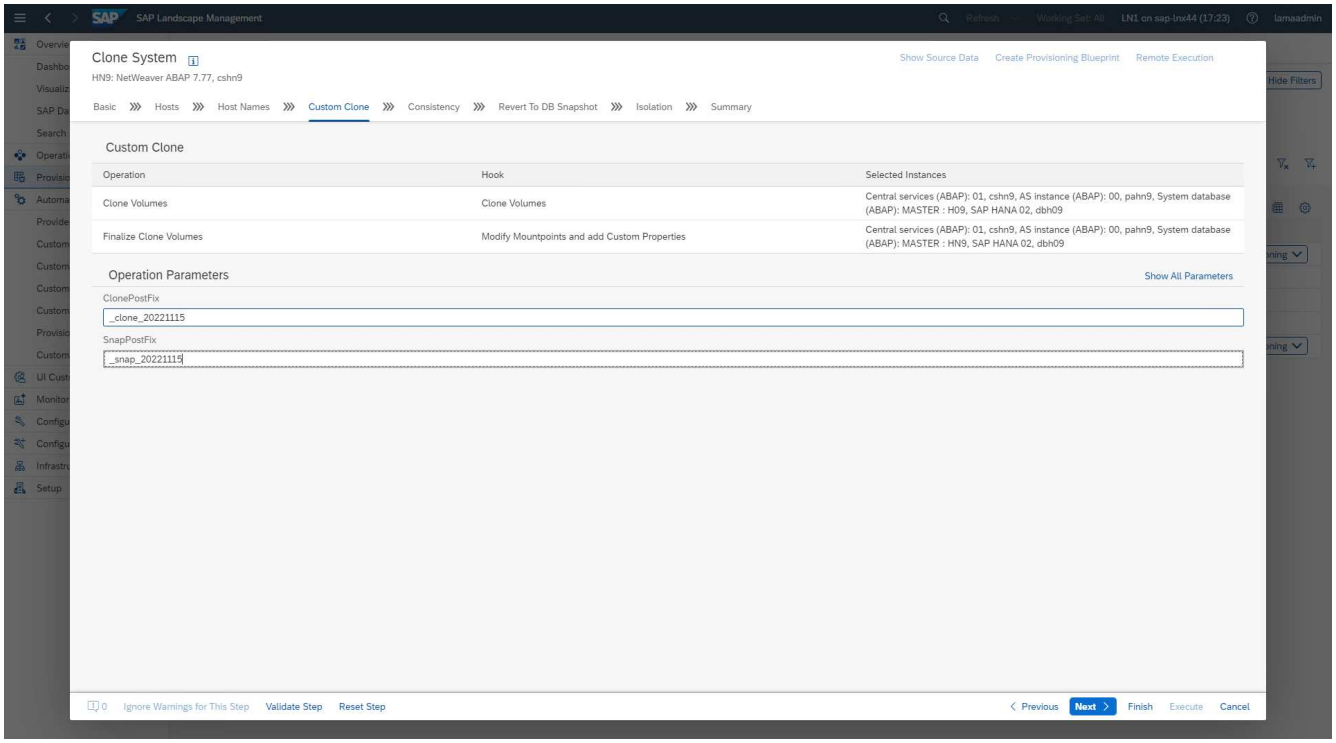
3. Screen 2 of the wizard asks for the target hosts that the new SAP instances are started on. The target hosts for this instance(s) can be selected out of the host pool specified in the previous screen. Each instance or service can be started on a different host. In our example, all three services run on the same host.



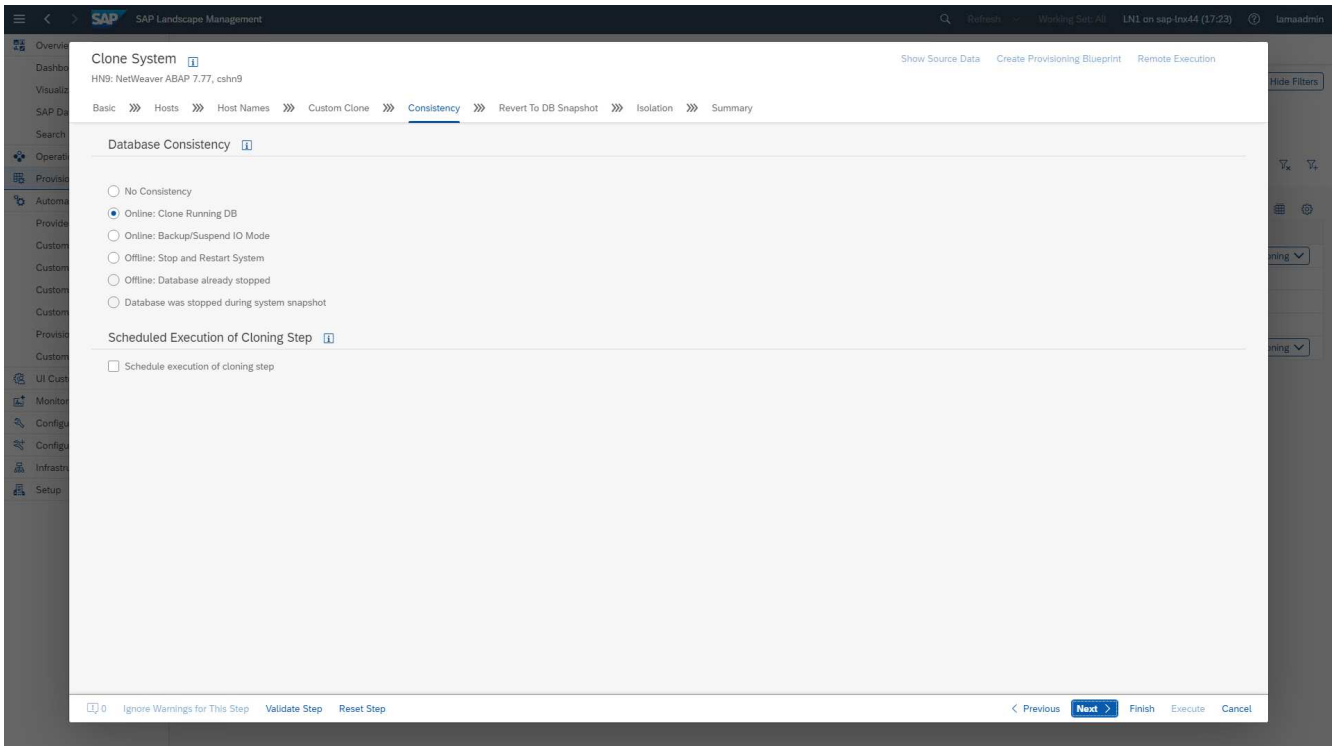
4. Provide the information requested in screen 3, which asks for virtual host names and networks. Typically, the host names are maintained in DNS, so the IP addresses are prepopulated accordingly.



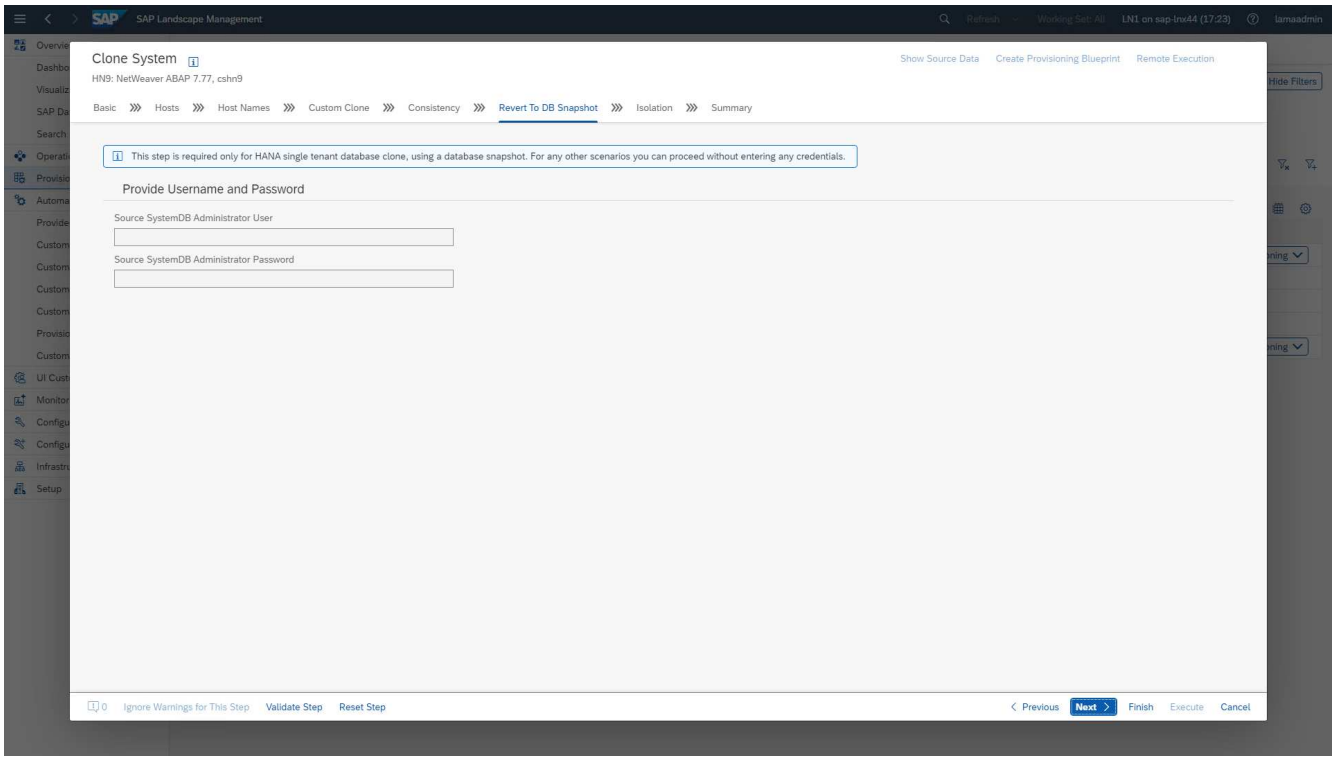
5. In screen 4, the custom clone operations are listed. A clone and a **SnapPostfix** name are provided, which are used during the storage clone operation for the FlexClone volume and Snapshot name, respectively. If you leave these fields empty, the default value configured in the variable section of the provider script `netapp_clone.sh` is used.



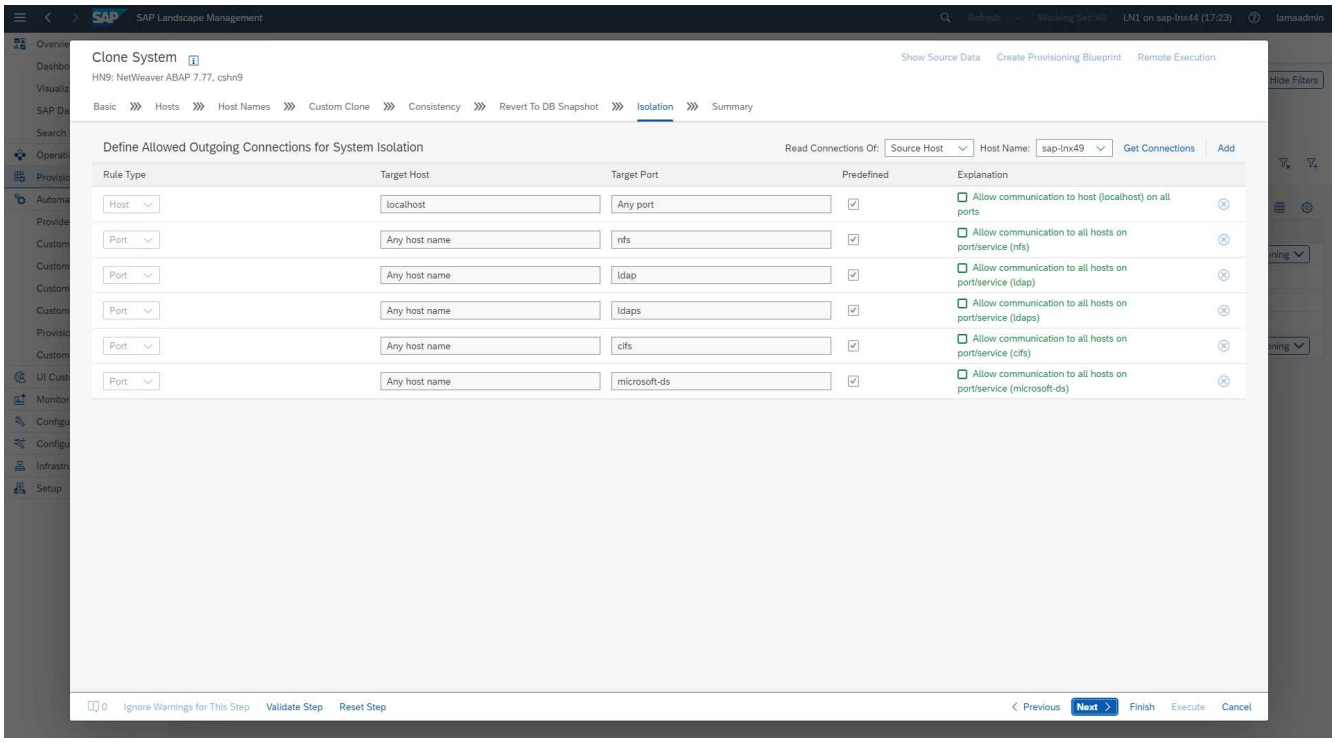
6. In screen 5, the database consistency option is selected. In our example, we selected **Online: Clone running DB**.



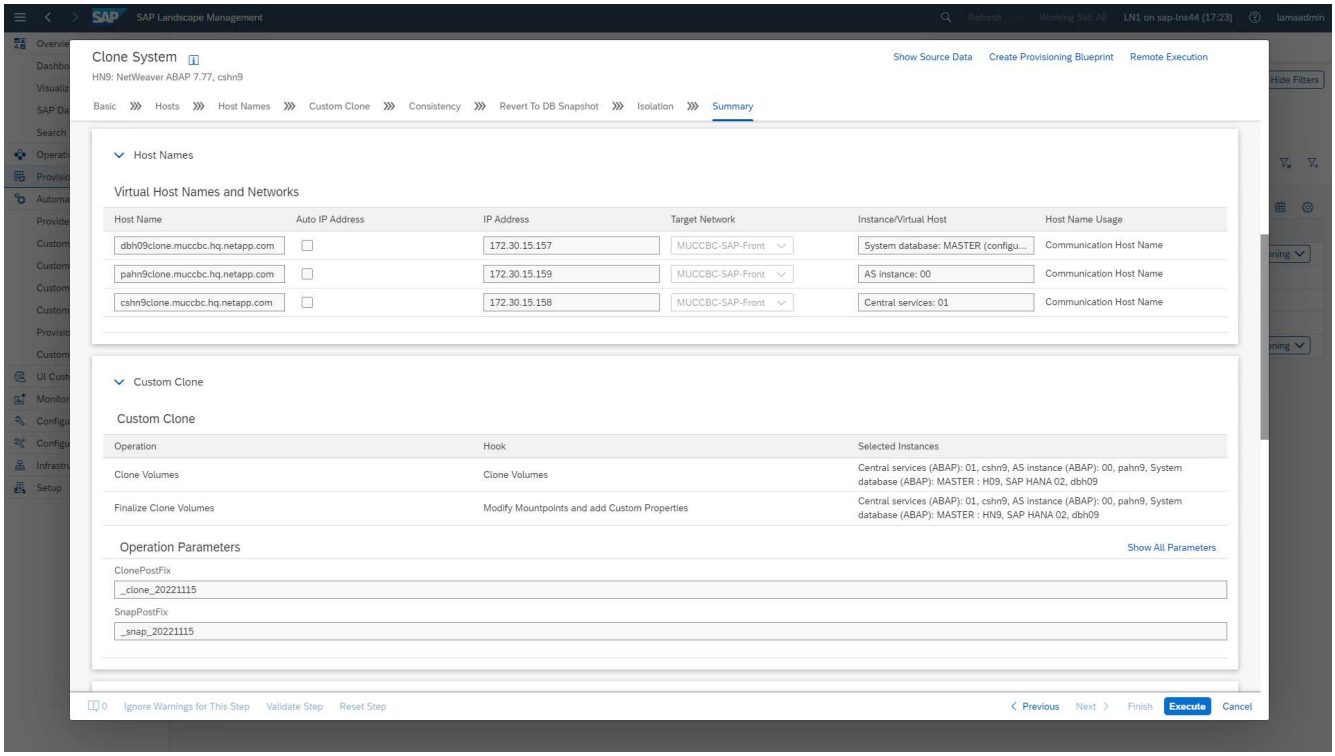
7. In screen 6, input is only required if you perform a tenant clone.



8. In screen 7, system isolation can be configured.

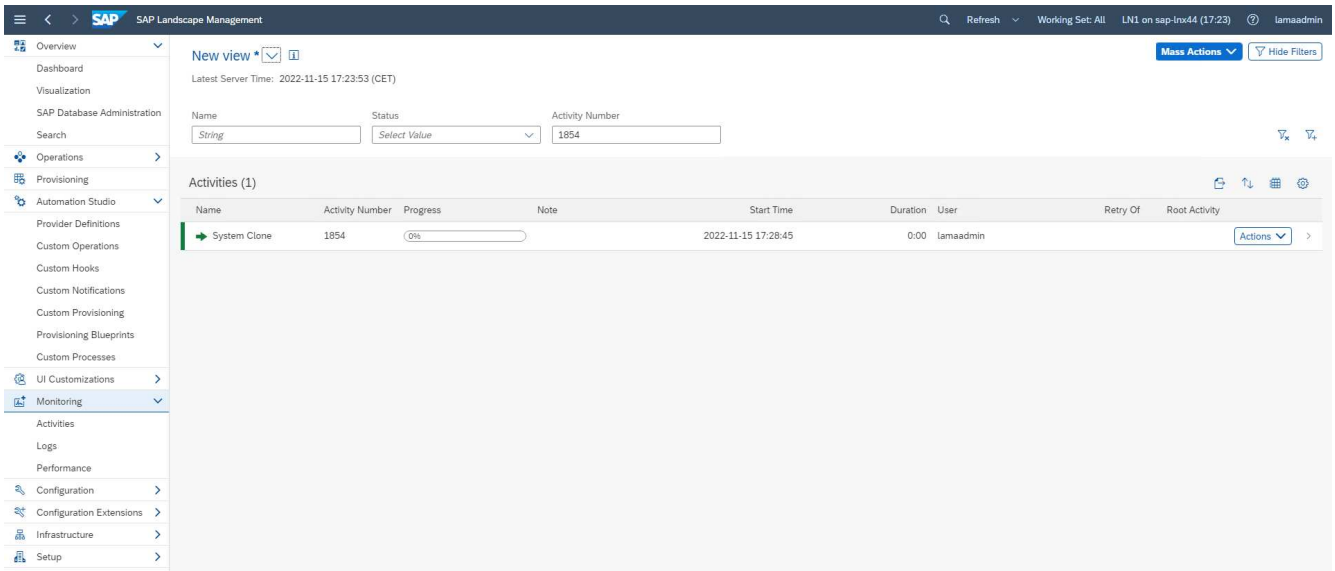


9. In screen 8, a summary page contains all the settings for final confirmation before the workflow is started. Click **Execute** to start the workflow.

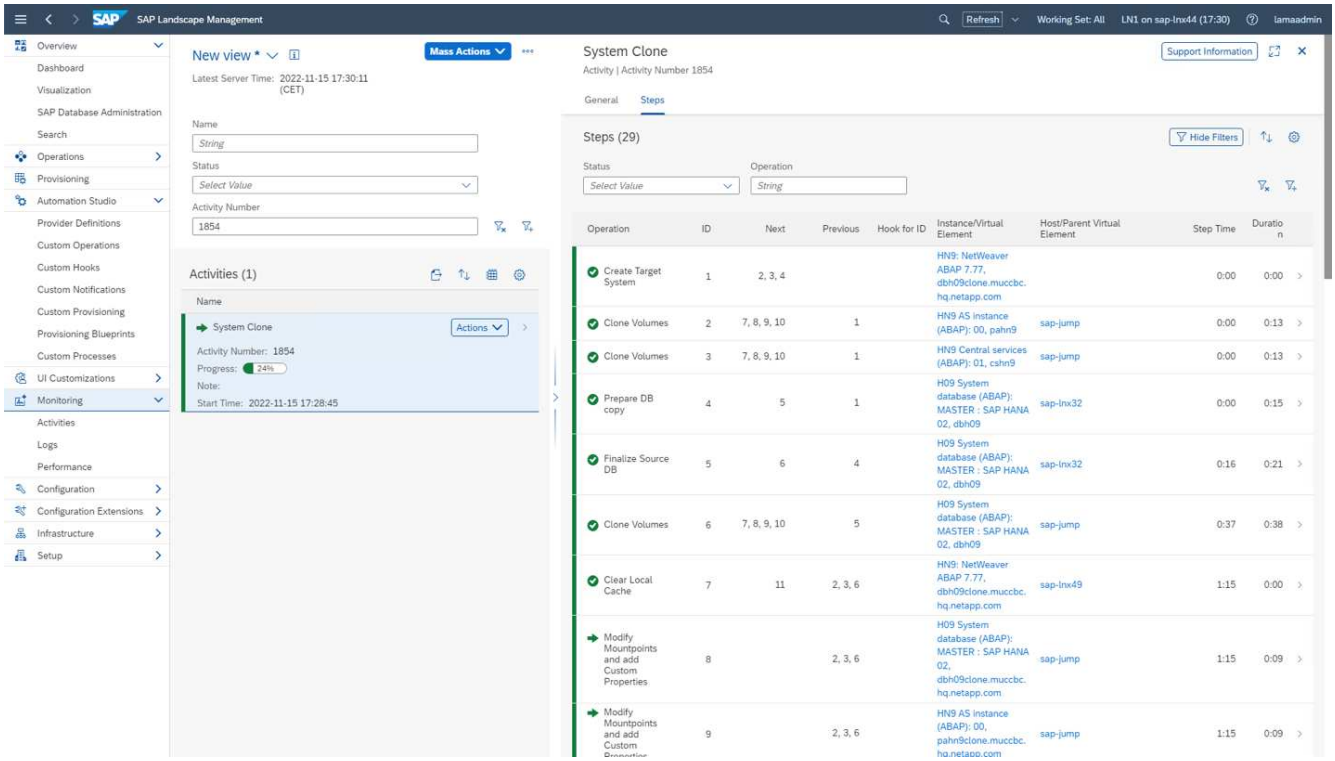


SAP LaMa now performs all the actions indicated in the configuration. These actions include creating the storage volume clones and exports, mounting them to the target host, adding the firewall rules for isolation, and starting the HANA database and SAP services.

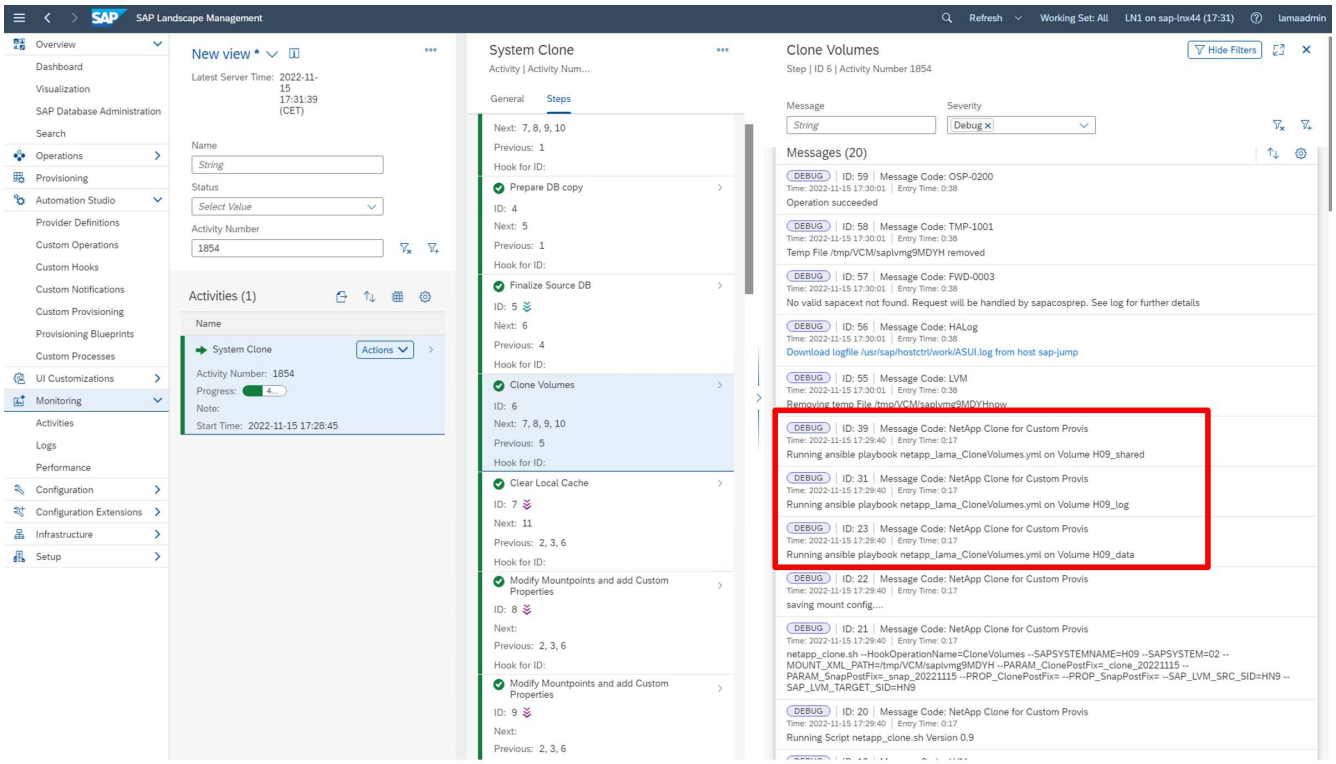
10. You can monitor the progress of the clone workflow under the **Monitoring** menu.



Within the detailed log, the operations **Clone Volume** and **Modify Mountpoints and add Custom Properties** are executed at the Ansible node, the `sap-jump` host. These steps are executed for each service, the HANA database, the SAP central services, and the SAP AS service.



11. By selecting the **Clone Volumes** task the detailed log for that step is displayed and the execution of the Ansible Playbook is shown here. You can see, that the Ansible playbook `netapp_lama_CloneVolumes.yml` is executed for each HANA database volume, data, log, and shared.



Latest Server Time: 2022-11-15 17:31:39 (CET)

Activity Number: 1854

Start Time: 2022-11-15 17:28:45

Clone Volumes

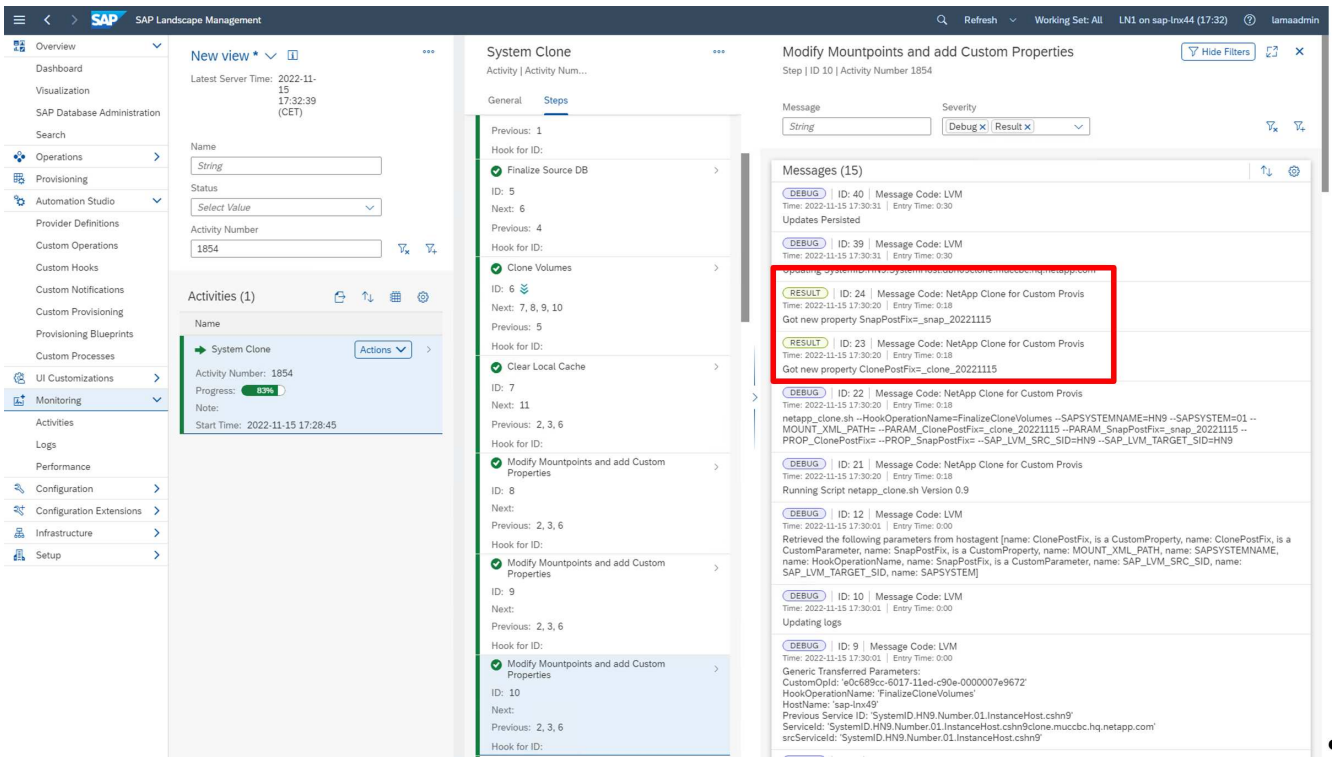
Step | ID 6 | Activity Number 1854

Severity: Debug

Messages (20)

```
DEBUG | ID: 59 | Message Code: OSP-0200
Time: 2022-11-15 17:30:01 | Entry Time: 0:38
Operation succeeded

DEBUG | ID: 58 | Message Code: TMP-1001
Time: 2022-11-15 17:30:01 | Entry Time: 0:38
Temp File /tmp/VCM/saplvmg9MDYH removed

DEBUG | ID: 57 | Message Code: FWD-0003
Time: 2022-11-15 17:30:01 | Entry Time: 0:38
No valid sapacext not found. Request will be handled by sapacosprep. See log for further details

DEBUG | ID: 56 | Message Code: HALog
Time: 2022-11-15 17:30:01 | Entry Time: 0:38
Download logfile /usr/sap/hostctrl/work/ASUI.log from host sap-jump

DEBUG | ID: 55 | Message Code: LVM
Time: 2022-11-15 17:30:01 | Entry Time: 0:38
Removing temp File /tmp/VCM/saplvmg9MDYHnow

DEBUG | ID: 39 | Message Code: NetApp Clone for Custom Provis
Time: 2022-11-15 17:29:40 | Entry Time: 0:17
Running ansible playbook netapp_lama_CloneVolumes.yml on Volume H09_shared

DEBUG | ID: 31 | Message Code: NetApp Clone for Custom Provis
Time: 2022-11-15 17:29:40 | Entry Time: 0:17
Running ansible playbook netapp_lama_CloneVolumes.yml on Volume H09_log

DEBUG | ID: 23 | Message Code: NetApp Clone for Custom Provis
Time: 2022-11-15 17:29:40 | Entry Time: 0:17
Running ansible playbook netapp_lama_CloneVolumes.yml on Volume H09_data

DEBUG | ID: 22 | Message Code: NetApp Clone for Custom Provis
Time: 2022-11-15 17:29:40 | Entry Time: 0:17
saving mount config....

DEBUG | ID: 21 | Message Code: NetApp Clone for Custom Provis
Time: 2022-11-15 17:29:40 | Entry Time: 0:17
netapp_clone.sh --HookOperationName=CloneVolumes --SAPSYSTEMNAME=H09 --SAPSYSTEM=02 --
MOUNT_XML_PATH=/tmp/VCM/saplvmg9MDYH --PARAM_ClonePostFix=_clone_20221115 --
PARAM_SnapPostFix=_snap_20221115 --PROP_ClonePostFix= --PROP_SnapPostFix= --SAP_LVM_SRC_SID=HN9 --
SAP_LVM_TARGET_SID=HN9

DEBUG | ID: 20 | Message Code: NetApp Clone for Custom Provis
Time: 2022-11-15 17:29:40 | Entry Time: 0:17
Running Script netapp_clone.sh Version 0.9
```

12. In the details view of the step **Modify Mountpoints and add Custom Properties**, you can find information about the mount points and the custom properties handed over by the execution script.



Latest Server Time: 2022-11-15 17:32:39 (CET)

Activity Number: 1854

Start Time: 2022-11-15 17:28:45

Modify Mountpoints and add Custom Properties

Step | ID 10 | Activity Number 1854

Severity: Debug, Result

Messages (15)

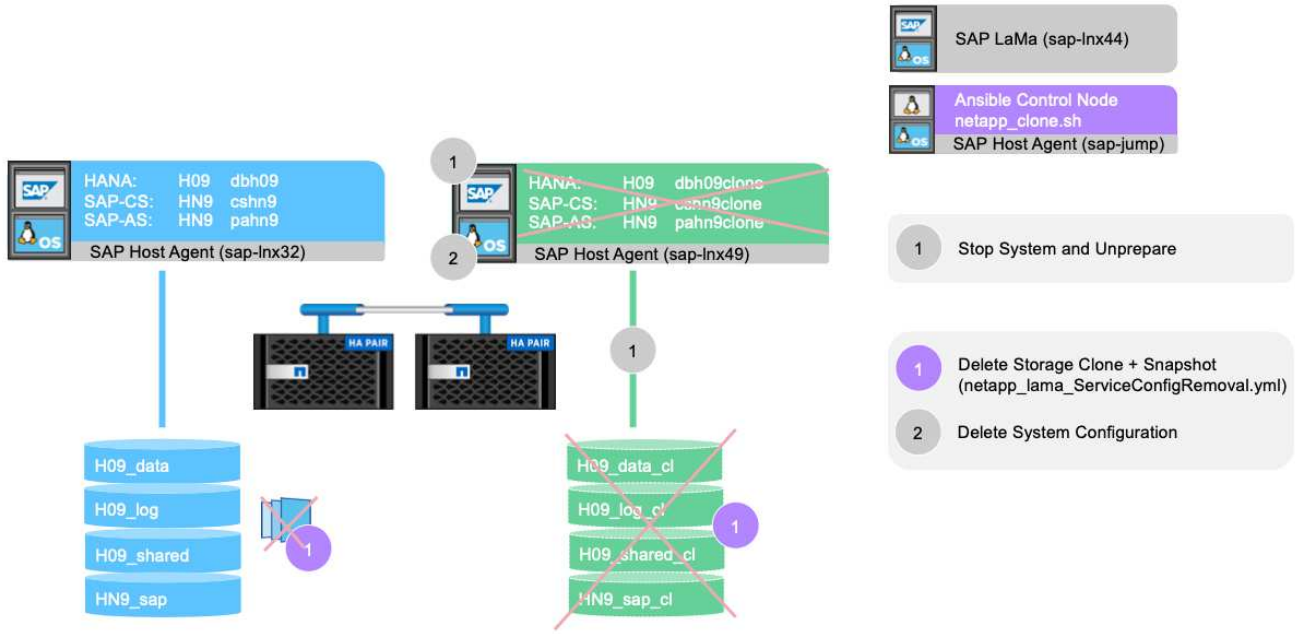
```
DEBUG | ID: 40 | Message Code: LVM
Time: 2022-11-15 17:30:31 | Entry Time: 0:30
Updates Persisted

DEBUG | ID: 39 | Message Code: LVM
Time: 2022-11-15 17:30:31 | Entry Time: 0:30

RESULT | ID: 24 | Message Code: NetApp Clone for Custom Provis
Time: 2022-11-15 17:30:20 | Entry Time: 0:18
Got new property SnapPostFix=_snap_20221115

RESULT | ID: 23 | Message Code: NetApp Clone for Custom Provis
Time: 2022-11-15 17:30:20 | Entry Time: 0:18
Got new property ClonePostFix=_clone_20221115

DEBUG | ID: 22 | Message Code: NetApp Clone for Custom Provis
Time: 2022-11-15 17:30:20 | Entry Time: 0:18
netapp_clone.sh --HookOperationName=FinalizeCloneVolumes --SAPSYSTEMNAME=HN9 --SAPSYSTEM=01 --
MOUNT_XML_PATH= --PARAM_ClonePostFix=_clone_20221115 --PARAM_SnapPostFix=_snap_20221115 --
PROP_ClonePostFix= --PROP_SnapPostFix= --SAP_LVM_SRC_SID=HN9 --SAP_LVM_TARGET_SID=HN9

DEBUG | ID: 21 | Message Code: NetApp Clone for Custom Provis
Time: 2022-11-15 17:30:20 | Entry Time: 0:18
Running Script netapp_clone.sh Version 0.9

DEBUG | ID: 12 | Message Code: LVM
Time: 2022-11-15 17:30:01 | Entry Time: 0:00
Retrieved the following parameters from hostagent [name: ClonePostFix, is a CustomProperty, name: ClonePostFix, is a
CustomParameter, name: SnapPostFix, is a CustomProperty, name: MOUNT_XML_PATH, name: SAPSYSTEMNAME,
name: HookOperationName, name: SnapPostFix, is a CustomParameter, name: SAP_LVM_SRC_SID, name:
SAP_LVM_TARGET_SID, name: SAPSYSTEM]

DEBUG | ID: 10 | Message Code: LVM
Time: 2022-11-15 17:30:01 | Entry Time: 0:00
Updating logs

DEBUG | ID: 9 | Message Code: LVM
Time: 2022-11-15 17:30:01 | Entry Time: 0:00
Generic Transferred Parameters:
CustomOpId: 'e0c689cc-6017-11ed-c90e-0000007e9672'
HookOperationName: 'FinalizeCloneVolumes'
HostName: 'sap-lnx49'
Previous Service ID: 'SystemID.HN9.Number.01.InstanceHost.cshn9'
ServiceId: 'SystemID.HN9.Number.01.InstanceHost.cshn9clone.muccbc.hq.netapp.com'
srcServiceId: 'SystemID.HN9.Number.01.InstanceHost.cshn9'
```

After the workflow has been completed, the cloned SAP system is prepared, started, and ready for use.

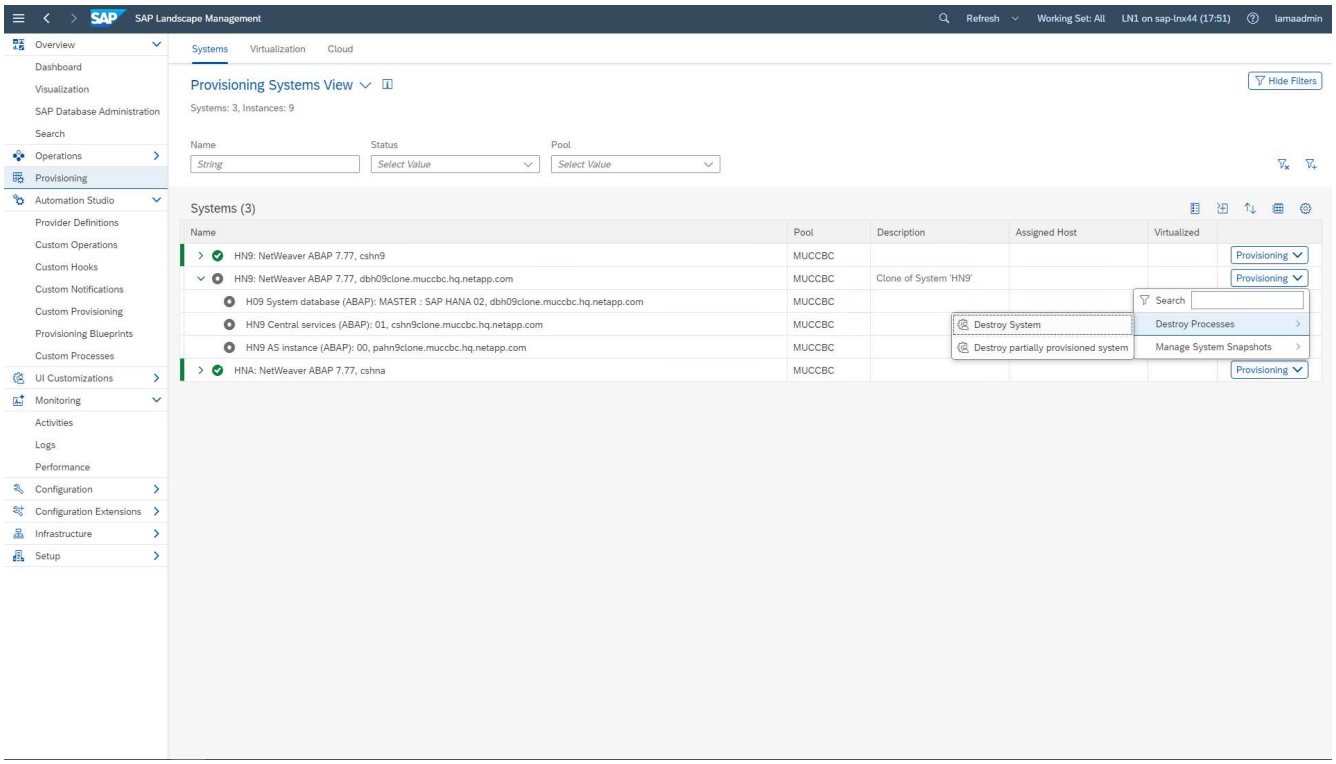
SAP LaMa deprovisioning workflow - system destroy

The following figure highlights the main steps executed with the system destroy workflow.

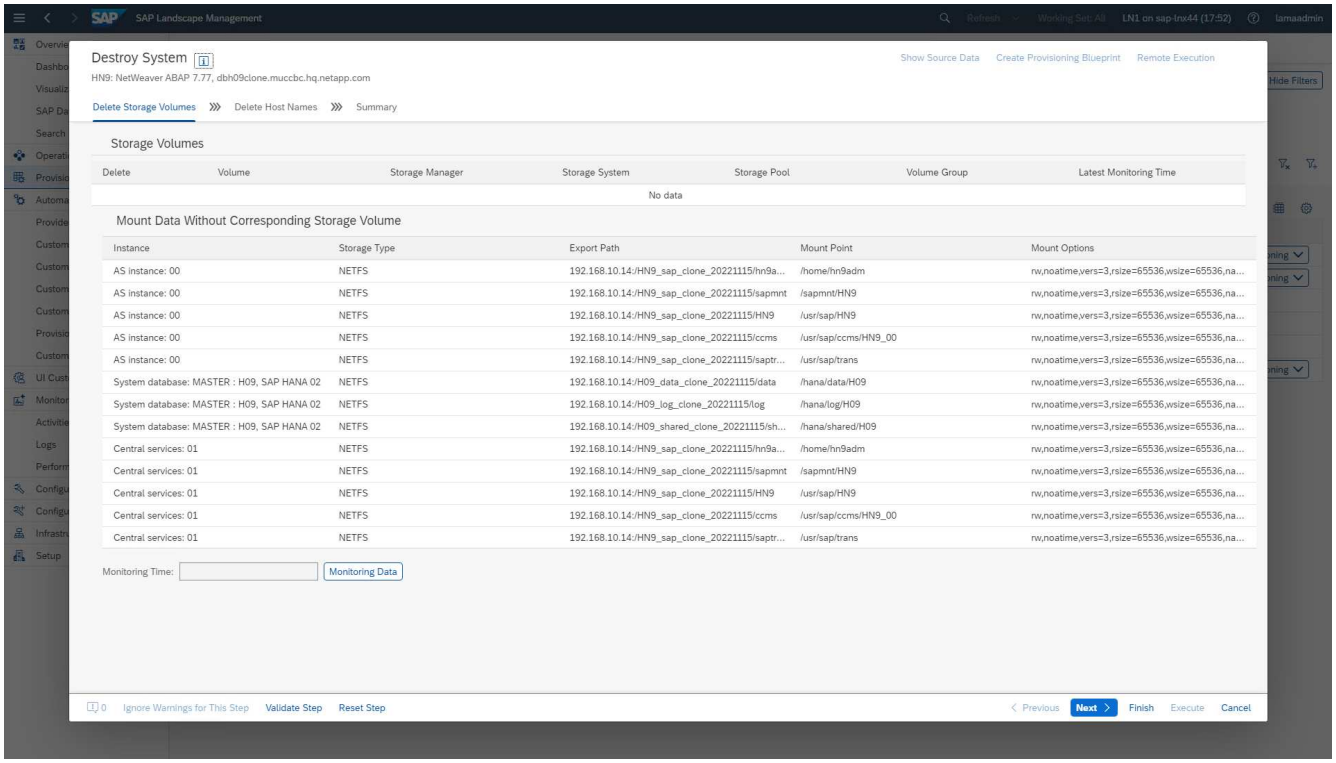
1. To decommission a cloned system, it must be stopped and prepared in advance. Afterwards the system destroy workflow can be started.



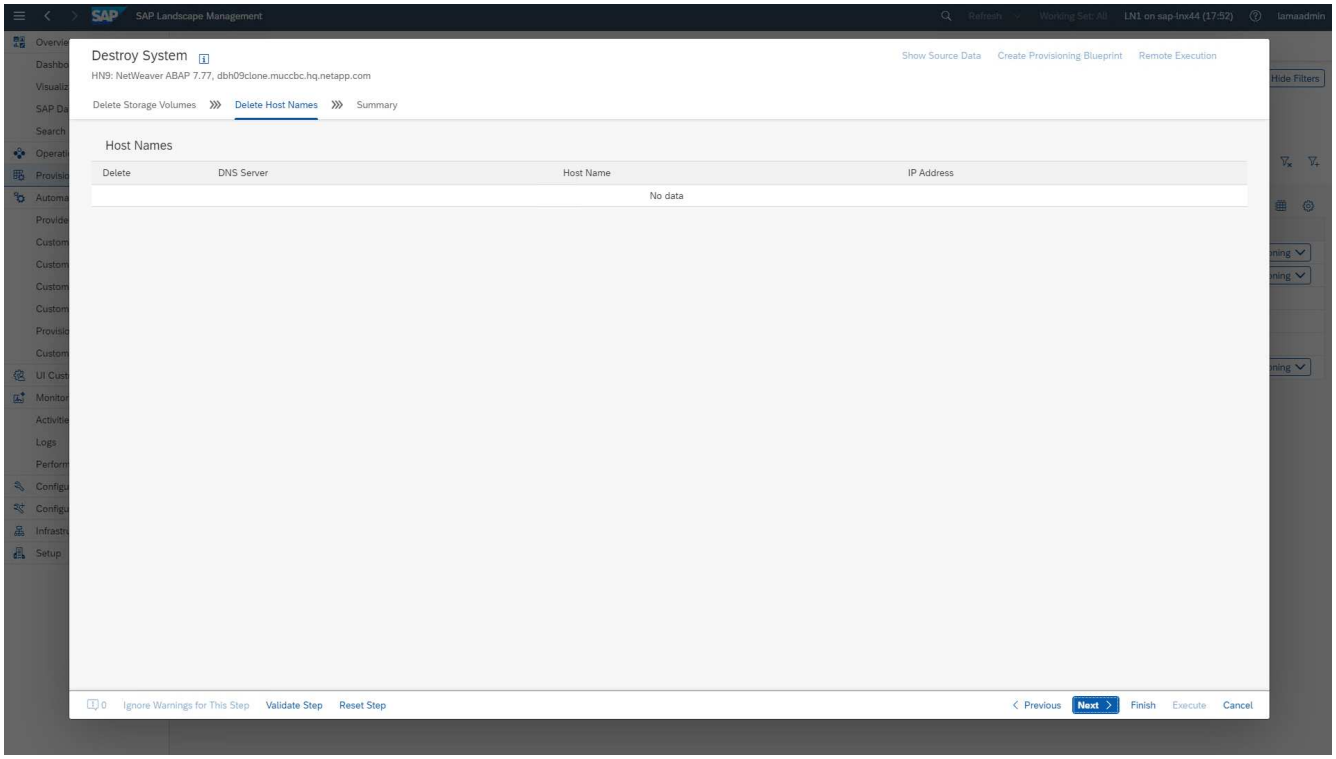
2. In this example, we run the system destroy workflow for the system created before. We select the system in the **System View** screen and start the system destroy workflow under **Destroy Processes**.



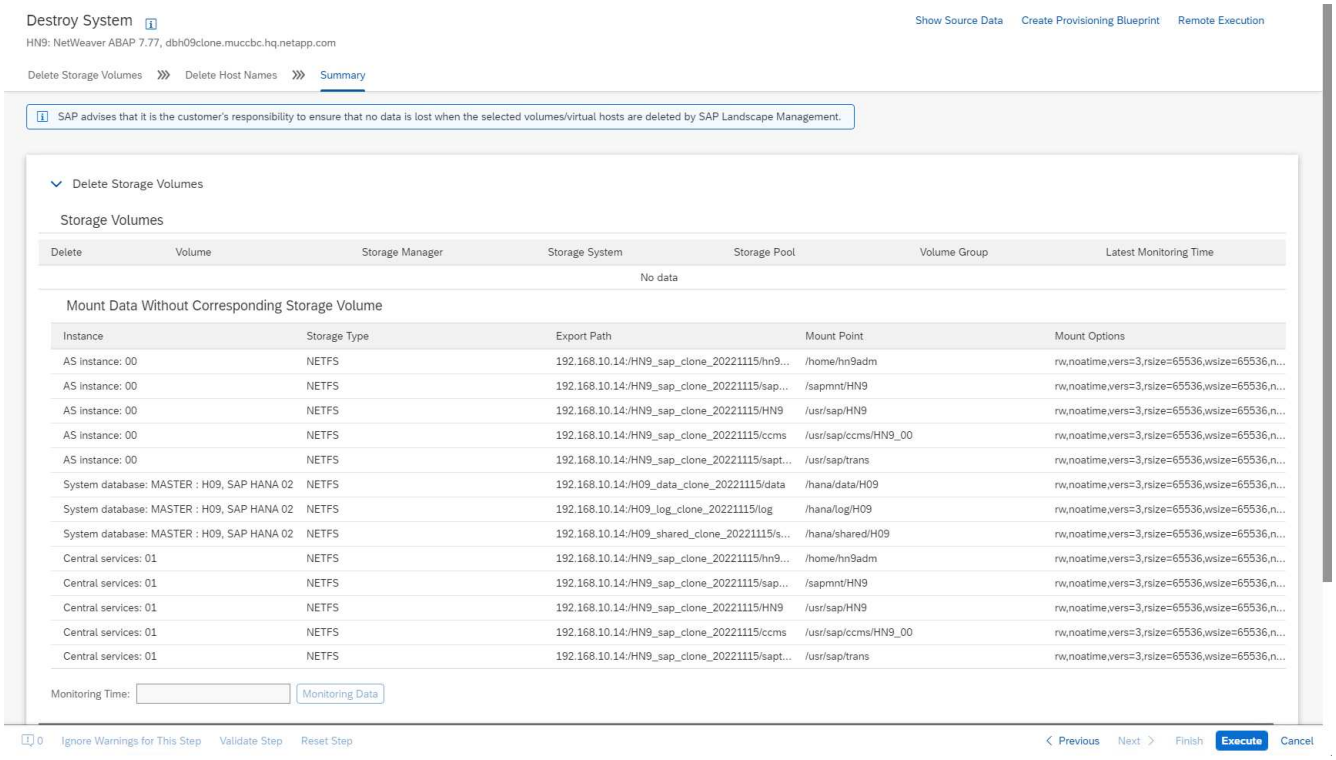
3. All the mount points maintained during the provisioning phase are shown here and are deleted during the system destroy workflow process.



No virtual hostnames are deleted because they are maintained through DNS and have been assigned automatically.

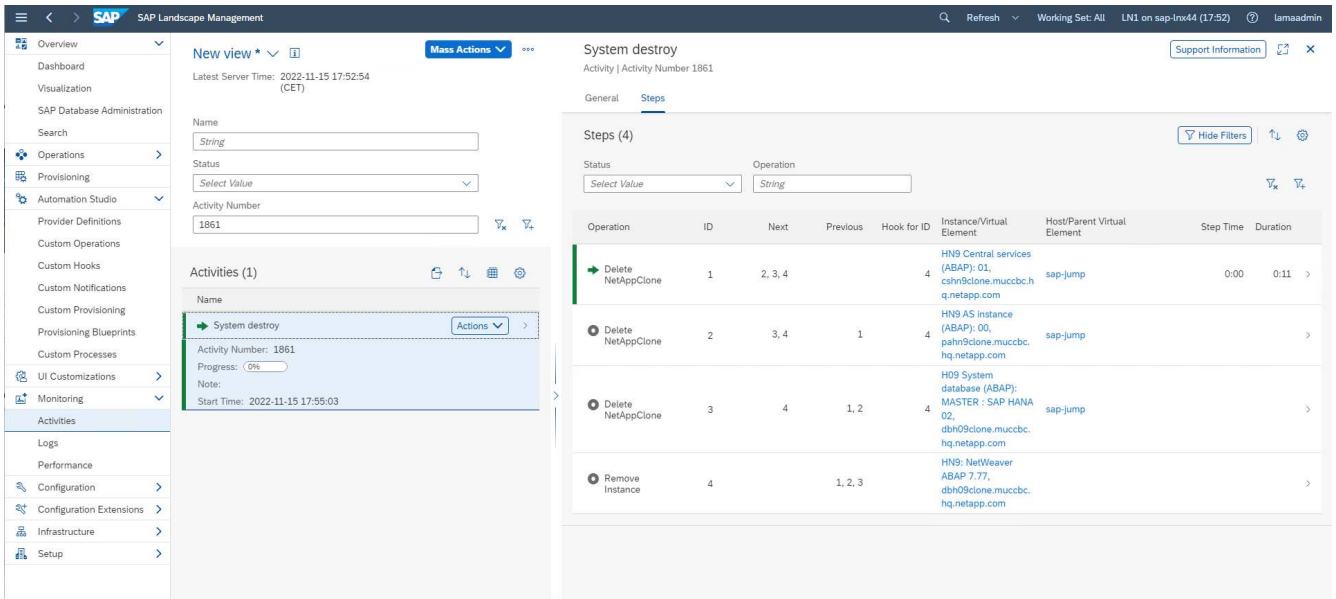


4. The operation is started by clicking the execute button.

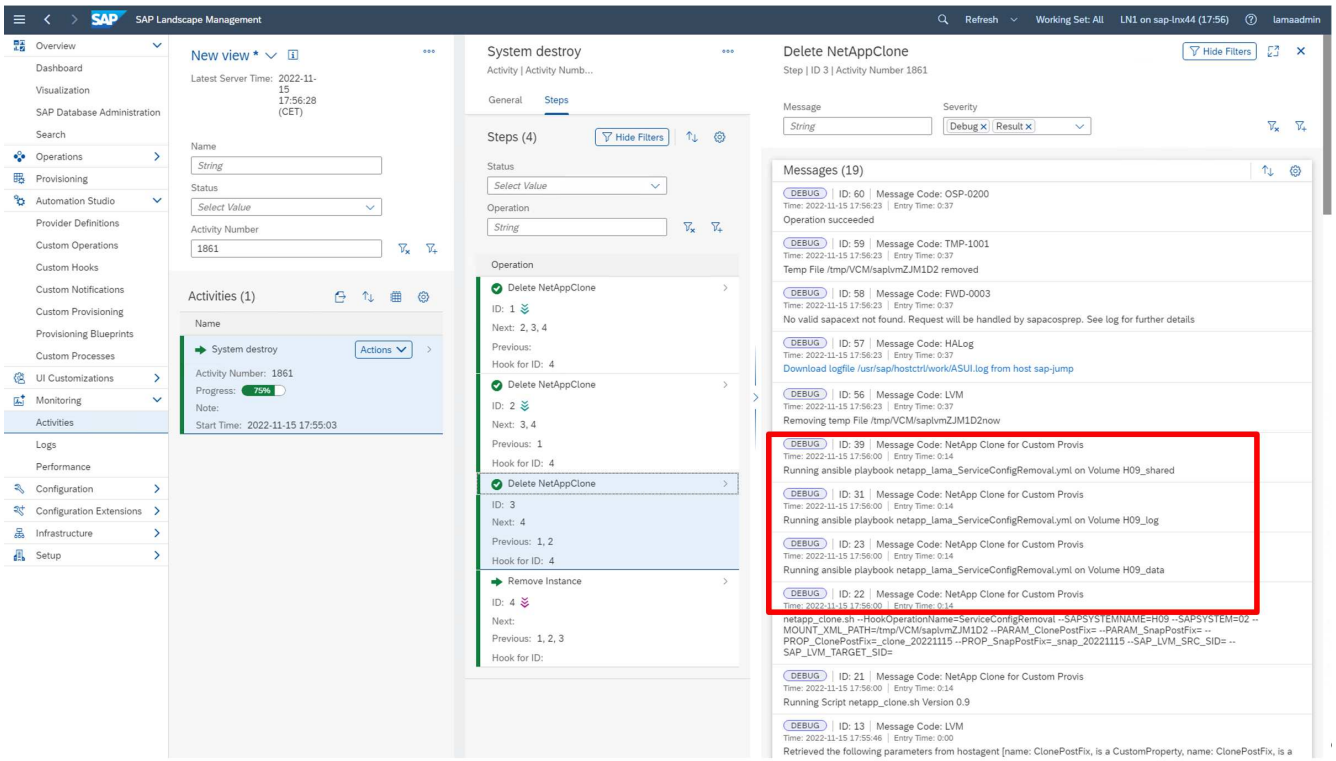


SAP LaMa now performs the deletion of the volume clones and deletes the configuration of the cloned system.

5. You can monitor the progress of the clone workflow under the **Monitoring** menu.

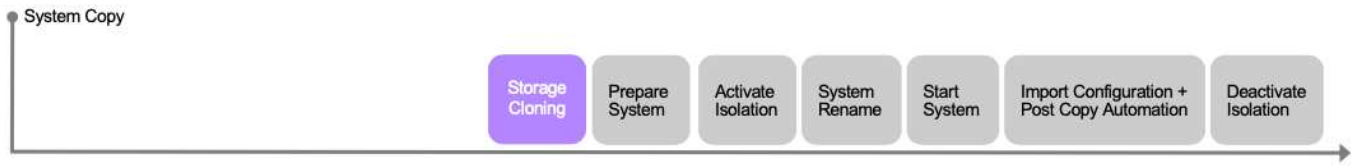


6. By selecting the **Delete NetAppClone** task, the detailed log for that step is displayed. The execution of the Ansible Playbook is shown here. As you can see, the Ansible playbook `netapp_lama_ServiceConfigRemoval.yml` is executed for each HANA database volume, data, log, and shared.

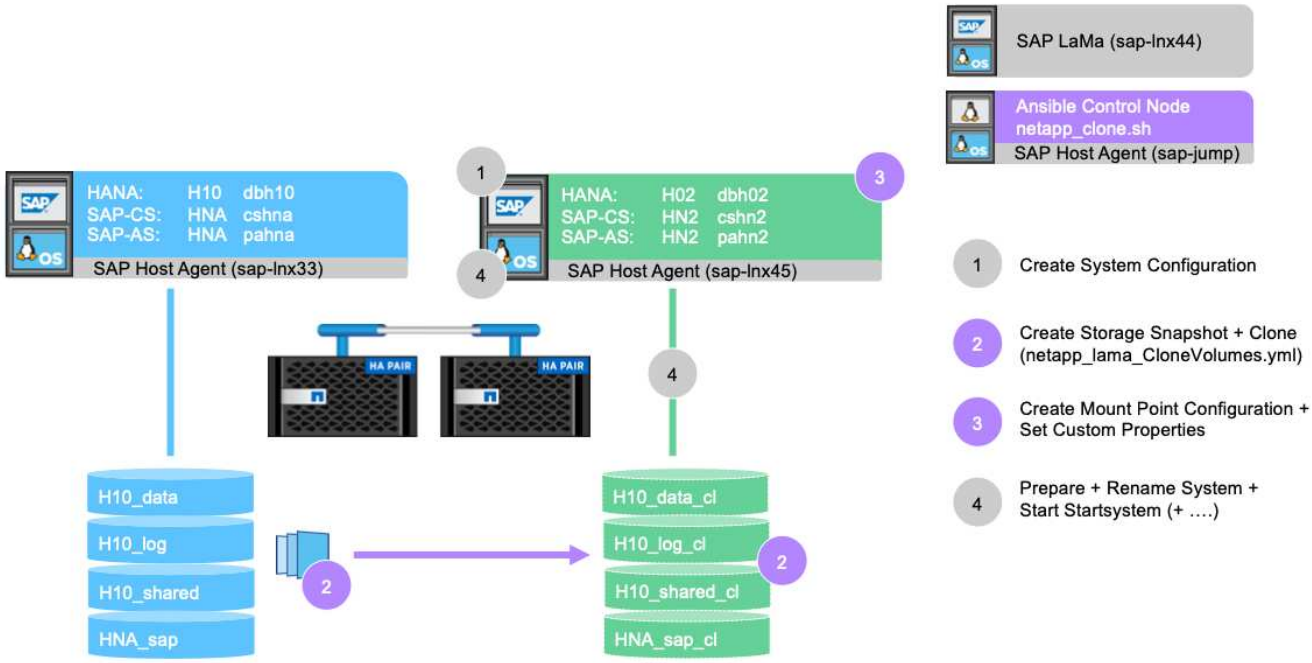


SAP LaMa provisioning workflow - copy system

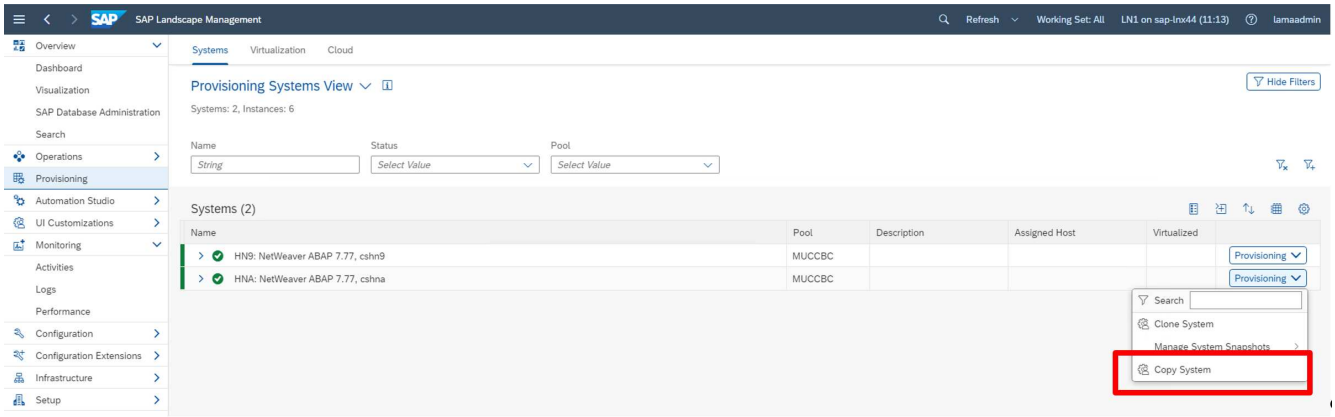
The following figure highlights the primary steps executed with the system copy workflow.



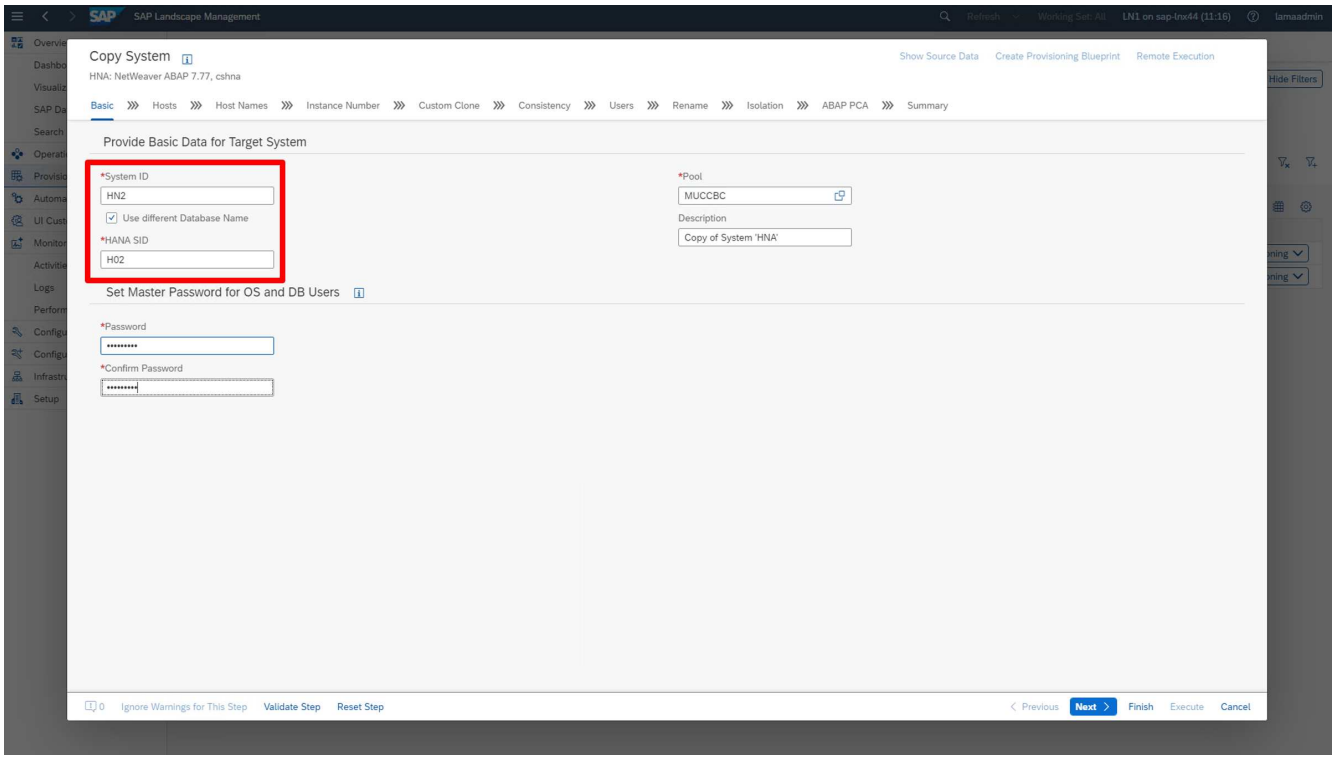
In this chapter, we briefly discuss the differences for the system clone workflow and input screens. As you can see in the following image, nothing changes in the storage workflow.



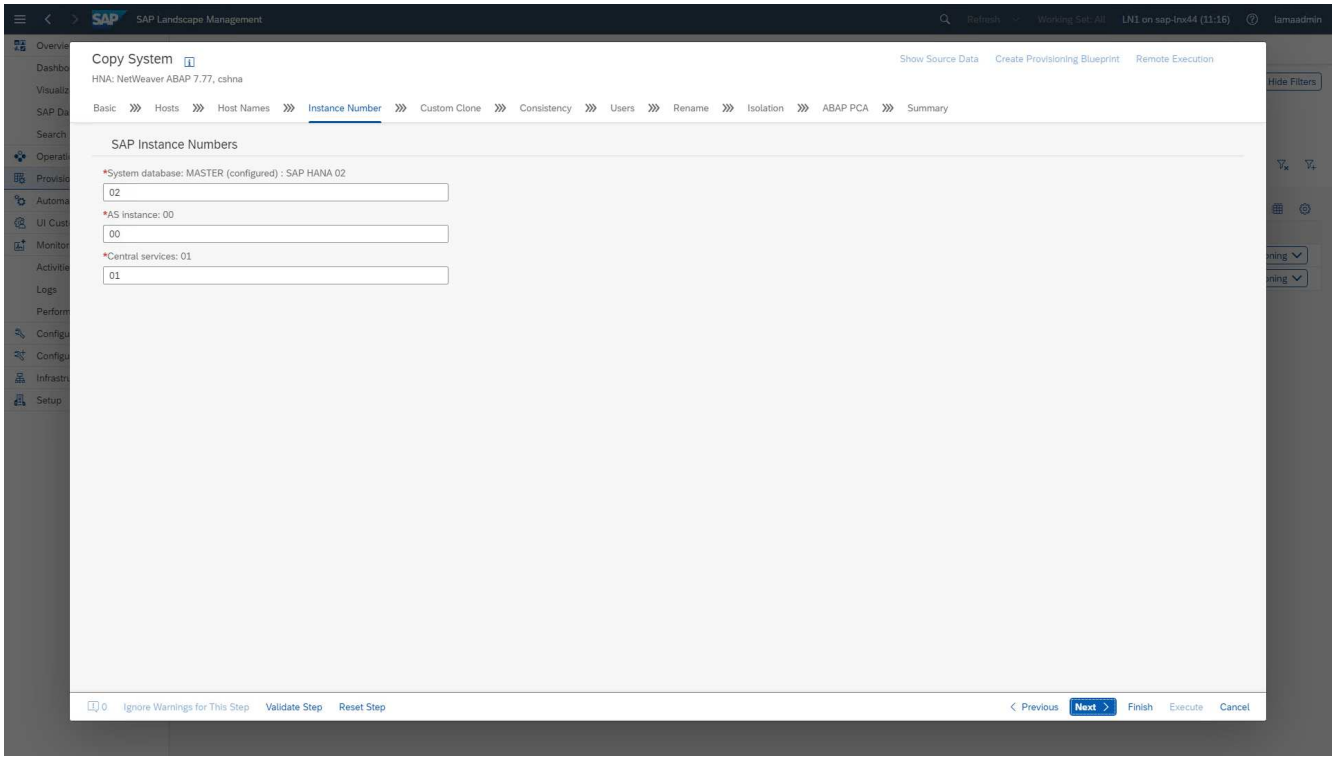
1. The system copy workflow can be started when the system is prepared accordingly. This is not a specific task for this configuration, and we do not explain it in detail. If you need further information, review the SAP LaMa documentation.



2. During the copy workflow, the system is renamed, as must be specified in the first screen.

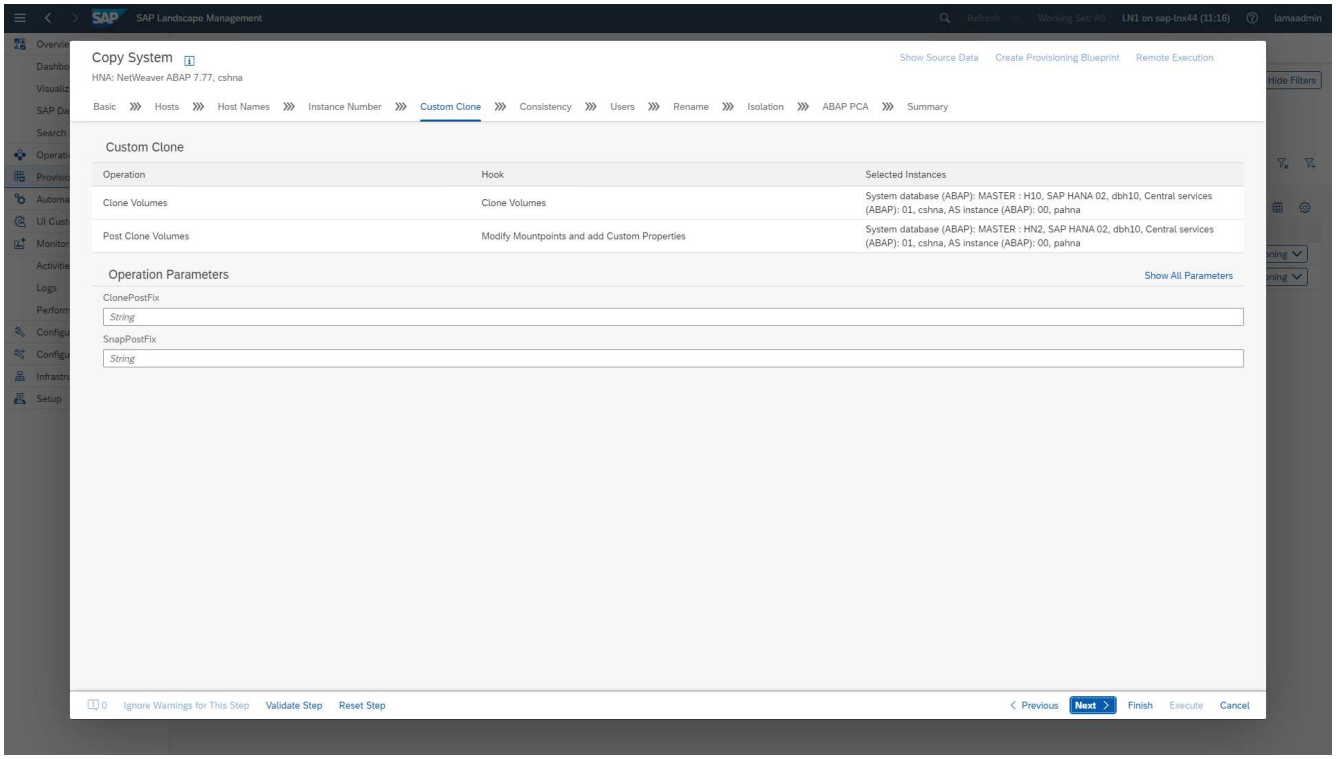


3. During the workflow, you can change the instance numbers.

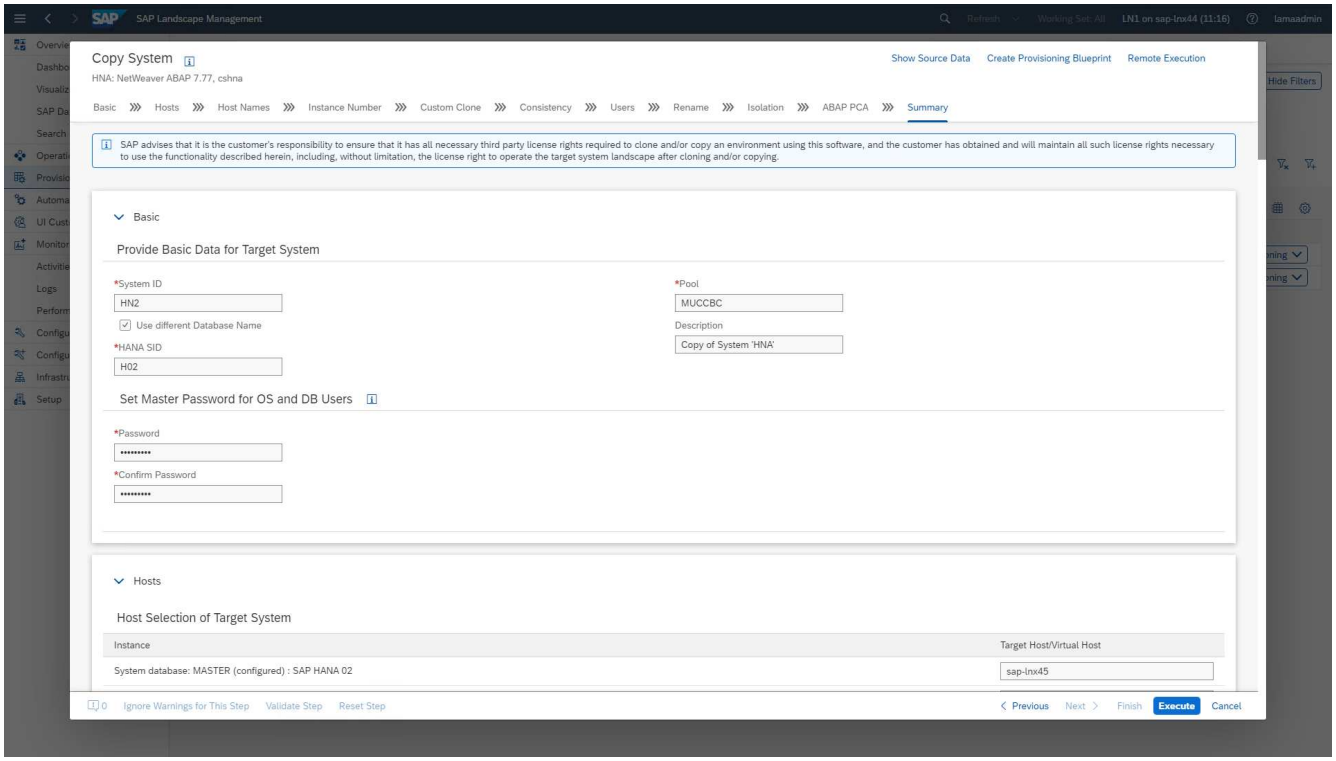


> Changing instance numbers has not been tested and might require changes in the provider script.

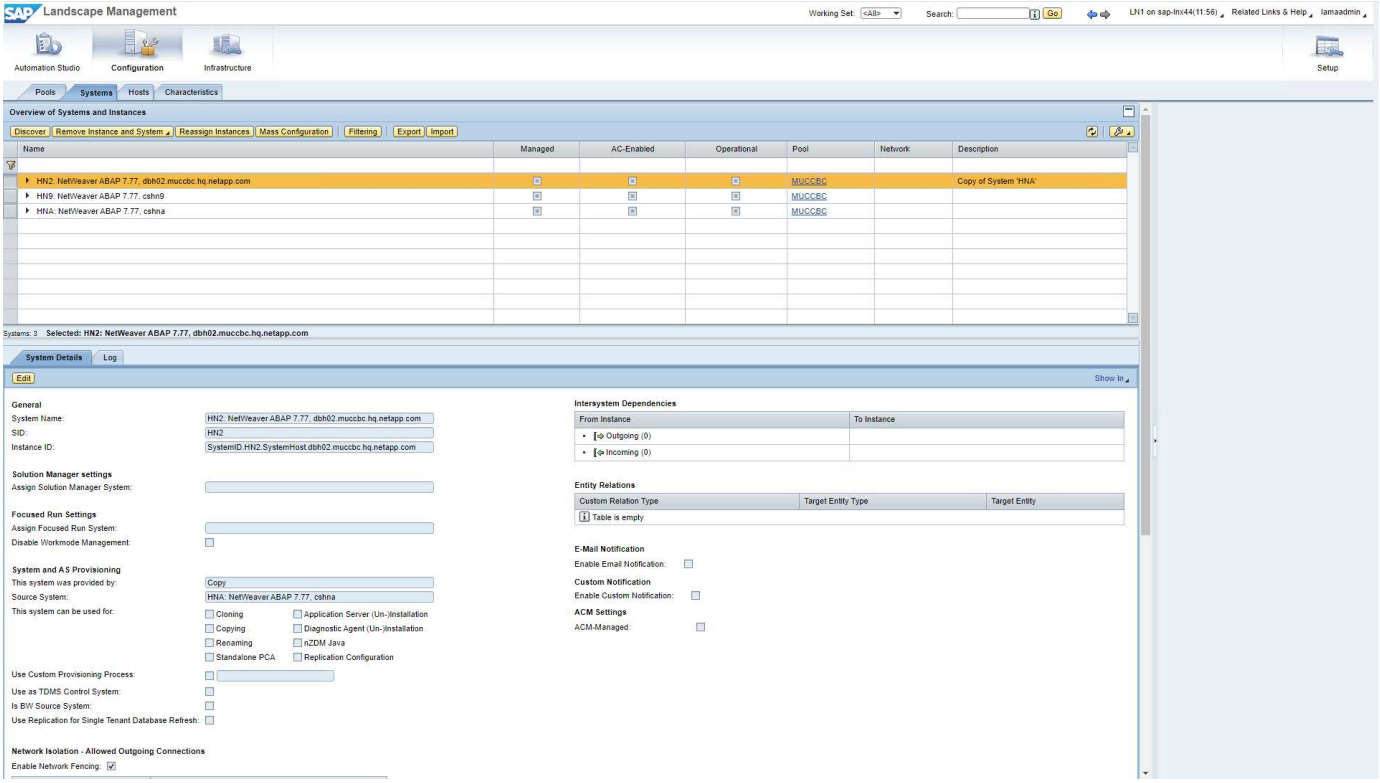
4. As described, the **Custom Clone** screen does not differ from the cloning workflow, as is shown here.



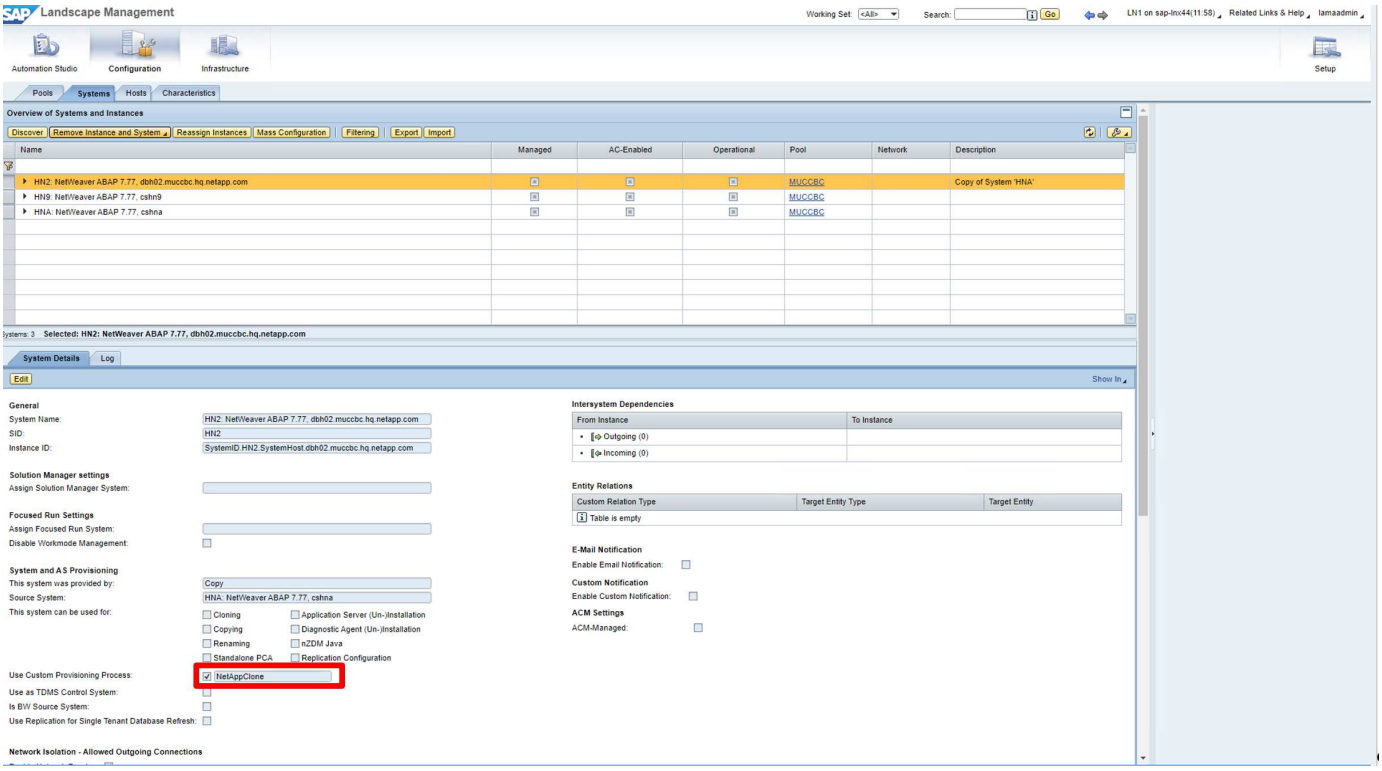
5. As we already described, the remaining input masks do not deviate from the standard, and we do not go into them any further here. The final screen shows a summary, and execution can now be started.



After the copy process, the target instance is not enabled for the custom cloning process.

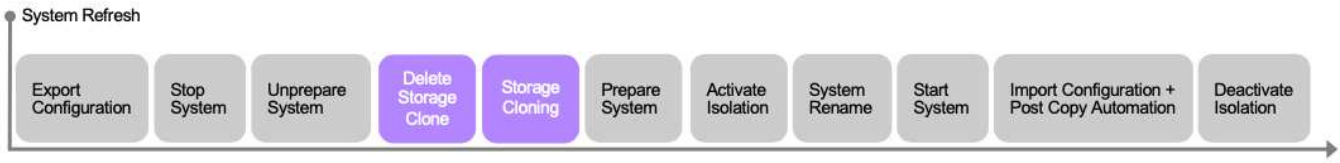


It must be adopted manually to run the pre-hook step during the system destroy process because a constraint is set and would prevent execution.

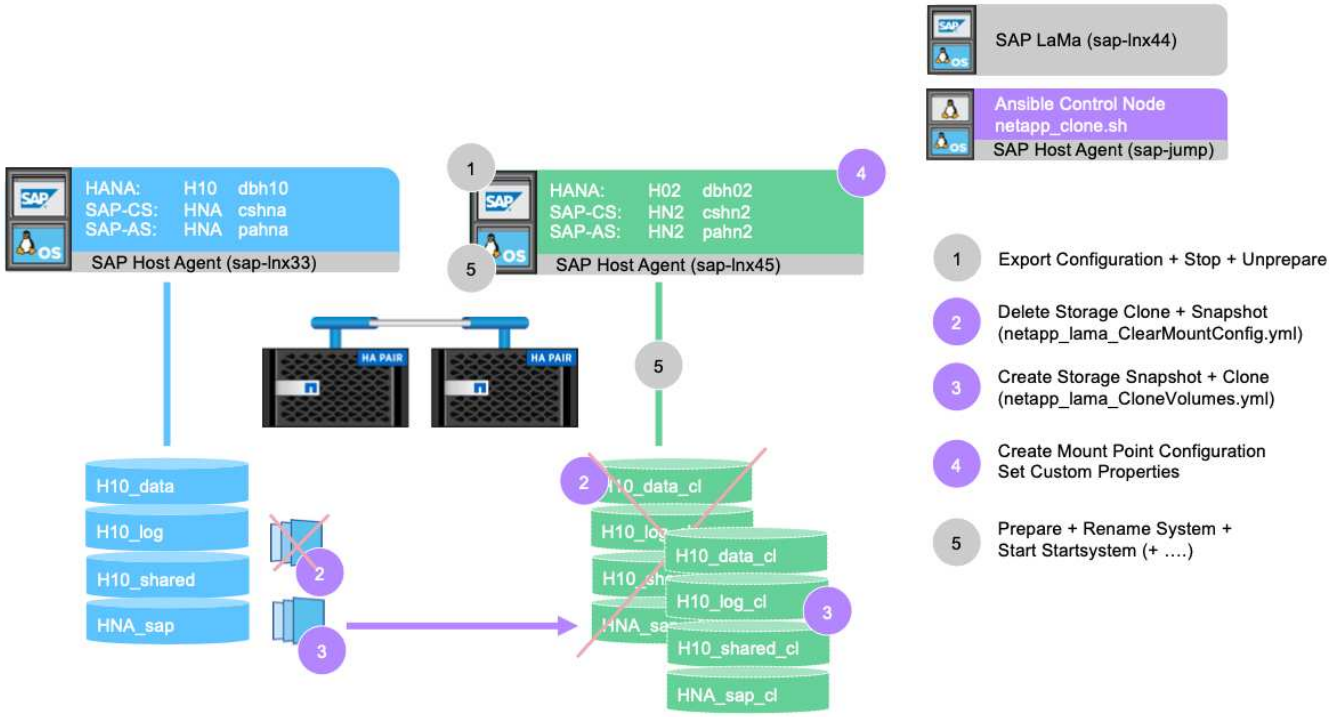


SAP LaMa provisioning workflow - system refresh

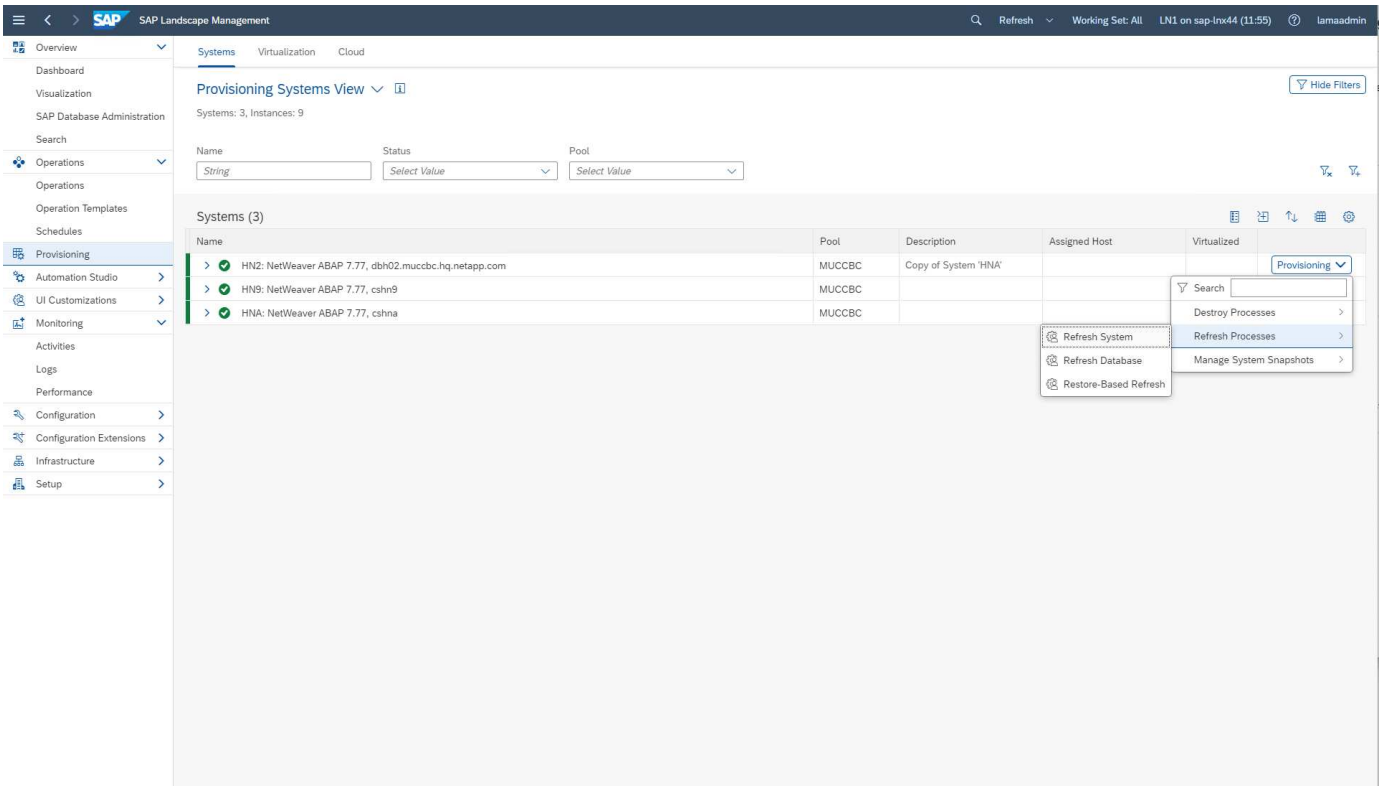
The following figure highlights the main steps executed with the system refresh workflow.



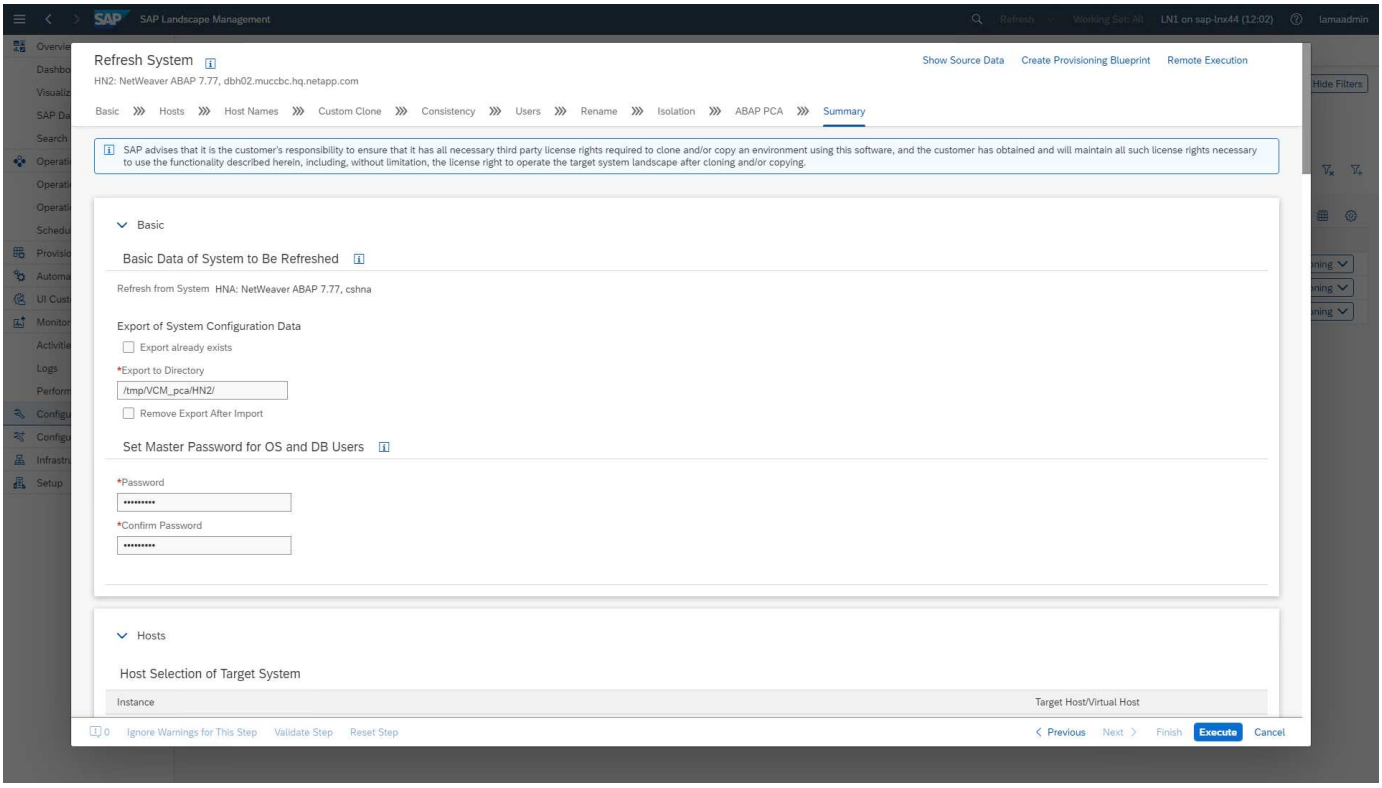
During the refresh workflow, the storage clone must be deleted. You can use the same Ansible playbook as for the system destroy workflow. However, the custom hook is defined to a different step, so the playbook is named accordingly. The process step for the clone doesn´t differ.



The refresh workflow can be triggered through the provisioning screen for a copied system.



Again, nothing differs in the input screens from the standard, and the workflow execution can be started from the summary screen.



Provider script configuration and Ansible playbooks

The following provider configuration file, execution script, and Ansible playbooks are used during the sample deployment and workflow execution in this documentation.

> The example scripts are provided as is and are not supported by NetApp. You can request the current version of the scripts via email to ng-sapcc@netapp.com.

Provider configuration file netapp_clone.conf

The configuration file is created as described in the SAP LaMa Documentation - Configuring SAP Host Agent Registered Scripts. This configuration file must be located on the Ansible control node where the SAP host agent is installed.

The configured os-user `sapuser` must have the appropriate permissions to execute the script and the called Ansible playbooks. You can place the script in a common script directory. SAP LaMa can provide multiple parameters when calling the script.

In addition to the custom parameters, `PARAM_ClonePostFix`, `PROP_ClonePostFix`, `PARAM_ClonePostFix`, and `PROP_ClonePostFix`, many others can be handed over, as is shown in the SAP LaMa Documentation.

```
root@sap-jump:~# cat /usr/sap/hostctrl/exe/operations.d/netapp_clone.conf
Name: netapp_clone
Username: sapuser
Description: NetApp Clone for Custom Provisioning
Command: /usr/sap/scripts/netapp_clone.sh
--HookOperationName=$[HookOperationName] --SAPSYSTEMNAME=$[SAPSYSTEMNAME]
--SAPSYSTEM=$[SAPSYSTEM] --MOUNT_XML_PATH=$[MOUNT_XML_PATH]
--PARAM_ClonePostFix=$[PARAM-ClonePostFix] --PARAM_SnapPostFix=$[PARAM
-SnapPostFix] --PROP_ClonePostFix=$[PROP-ClonePostFix]
--PROP_SnapPostFix=$[PROP-SnapPostFix]
--SAP_LVM_SRC_SID=$[SAP_LVM_SRC_SID]
--SAP_LVM_TARGET_SID=$[SAP_LVM_TARGET_SID]
ResulConverter: hook
Platform: Unix
```

Provider script netapp_clone.sh

The provider script must be stored in `/usr/sap/scripts` as configured in the provider configuration file.

Variables

The following variables are hard coded in the script and must be adapted accordingly.

- `PRIMARY_CLUSTER=`<hostname of netapp cluster>
- `PRIMARY_SVM=`<SVM name where source system volumes are stored>

The certificate files `PRIMARY_KEYFILE=/usr/sap/scripts/ansible/certs/ontap.key` and `PRIMARY_CERTFILE=/usr/sap/scripts/ansible/certs/ontap.pem` must be provided as described in NetApp Ansible modules - Prepare ONTAP.

> If different clusters or SVMs are required for different SAP systems, these variables can be added as parameters in the SAP LaMa provider definition.

Function: create inventory file

To make Ansible playbook execution more dynamic, an `inventory. yml` file is created on the fly. Some static values are configured in the variable section and some are dynamically created during execution.

Function: run Ansible playbook

This function is used to execute the Ansible playbook together with the dynamically created `inventory.yml` file. The naming convention for the playbooks is `netapp_lama_${HookOperationName}.yml`. The values for `${HookOperationName}` is dependent on the LaMa operation and handed over by LaMa as a command line parameter.

Section Main

This section contains the main execution plan. The variable `${HookOperationName}` contains the name of the LaMa replacement step and is provided by LaMa when the script is called.

- Values with the system clone and system copy provisioning workflow:
  - CloneVolumes
  - PostCloneVolumes
- Value with the system destroy workflow:
  - ServiceConfigRemoval
- Value with the system refresh workflow:
  - ClearMountConfig

HookOperationName = CloneVolumes

With this step, the Ansible playbook is executed, which triggers the Snapshot copy and cloning operation. The volume names and mount configuration are handed over by SAP LaMa through an XML file defined in the variable `$MOUNT_XML_PATH`. This file is saved because it is used later in the step `FinalizeCloneVolumes` to create the new mount-point configuration. The volume names are extracted from the XML file and the Ansible cloning playbook is executed for each volume.

> In this example, the AS instance and the central services share the same volume. Therefore, volume cloning is only executed when the SAP instance number (`$SAPSYSTEM`) is not `01`. This might differ in other environments and must be changed accordingly.

HookOperationName = PostCloneVolumes

During this step, the custom properties `ClonePostFix` and `SnapPostFix` and the mount point configuration for the target system are maintained.

The custom properties are used later as input when the system is decommissioned during the `ServiceConfigRemoval` or `ClearMountConfig` phase. The system is designed to preserve the settings of the custom parameters that were specified during the system provisioning workflow.

The values used in this example are `ClonePostFix=_clone_20221115` and

`SnapPostFix=_snap_20221115`.

For the volume `HN9_sap`, the dynamically created Ansible file includes the following values: `datavolumename: HN9_sap`, `snapshotpostfix: _snap_20221115`, and `clonepostfix: _clone_20221115`.

Which leads into the snapshot name on the volume HN9_sap `HN9_sap_snap_20221115` and the created volume clone name `HN9_sap_clone_20221115`.

> Custom properties could be used in any way to preserve parameters used during the provisioning process.

The mount point configuration is extracted from the XML file that has been handed over by LaMa in the `CloneVolume` step. The `ClonePostFix` is added to the volume names and send back to LaMa through the default script output. The functionality is described in SAP Note 1889590.

> In this example, qtrees on the storage system are used as a common way to place different data on a single volume. For example, `HN9_sap` holds the mount points for `/usr/sap/HN9`, `/sapmnt/HN9`, and `/home/hn9adm`. Subdirectories work in the same way. This might differ in other environments and must be changed accordingly.

HookOperationName = ServiceConfigRemoval

In this step, the Ansible playbook that is responsible for the deletion of the volume clones is running.

The volume names are handed over by SAP LaMa through the mount configuration file, and the custom properties `ClonePostFix` and `SnapPostFix` are used to hand over the values of the parameters originally specified during the system provisioning workflow (see the note at `HookOperationName = PostCloneVolumes`).

The volume names are extracted from the xml file, and the Ansible cloning playbook is executed for each volume.

> In this example, the AS instance and the central services share the same volume. Therefore, the volume deletion is only executed when the SAP instance number (`$SAPSYSTEM`) is not `01`. This might differ in other environments and must be changed accordingly.

HookOperationName = ClearMountConfig

In this step, the Ansible playbook that is responsible for the deletion of the volume clones during a system refresh workflow is running.

The volume names are handed over by SAP LaMa through the mount configuration file, and the custom properties `ClonePostFix` and `SnapPostFix` are used to hand over the values of the parameters originally specified during the system provisioning workflow.

The volume names are extracted from the XML file and the Ansible cloning playbook is executed for each volume.

> In this example, the AS instance and the central services share the same volume. Therefore, volume deletion is only executed when the SAP instance number (`$SAPSYSTEM`) is not `01`. This might differ in other environments and must be changed accordingly.

```
root@sap-jump:~# cat /usr/sap/scripts/netapp_clone.sh
#!/bin/bash
#Section - Variables
#########################################
VERSION="Version 0.9"
#Path for ansible play-books
ANSIBLE_PATH=/usr/sap/scripts/ansible
#Values for Ansible Inventory File
PRIMARY_CLUSTER=grenada
PRIMARY_SVM=svm-sap01
PRIMARY_KEYFILE=/usr/sap/scripts/ansible/certs/ontap.key
PRIMARY_CERTFILE=/usr/sap/scripts/ansible/certs/ontap.pem
#Default Variable if PARAM ClonePostFix / SnapPostFix is not maintained in
LaMa
DefaultPostFix=_clone_1
#TMP Files - used during execution
YAML_TMP=/tmp/inventory_ansible_clone_tmp_$$.yml
TMPFILE=/tmp/tmpfile.$$
MY_NAME="`basename $0`"
BASE_SCRIPT_DIR="`dirname $0`"
#Sendig Script Version and run options to LaMa Log
echo "[DEBUG]: Running Script $MY_NAME $VERSION"
echo "[DEBUG]: $MY_NAME $@"
#Command declared in the netapp_clone.conf Provider definition
#Command: /usr/sap/scripts/netapp_clone.sh
--HookOperationName=$[HookOperationName] --SAPSYSTEMNAME=$[SAPSYSTEMNAME]
--SAPSYSTEM=$[SAPSYSTEM] --MOUNT_XML_PATH=$[MOUNT_XML_PATH]
--PARAM_ClonePostFix=$[PARAM-ClonePostFix] --PARAM_SnapPostFix=$[PARAM
-SnapPostFix] --PROP_ClonePostFix=$[PROP-ClonePostFix]
--PROP_SnapPostFix=$[PROP-SnapPostFix]
--SAP_LVM_SRC_SID=$[SAP_LVM_SRC_SID]
--SAP_LVM_TARGET_SID=$[SAP_LVM_TARGET_SID]
#Reading Input Variables hand over by LaMa
for i in "$@"
do
case $i in
--HookOperationName=*)
HookOperationName="${i#*=}";shift;;
--SAPSYSTEMNAME=*)
SAPSYSTEMNAME="${i#*=}";shift;;
--SAPSYSTEM=*)
SAPSYSTEM="${i#*=}";shift;;
--MOUNT_XML_PATH=*)
MOUNT_XML_PATH="${i#*=}";shift;;
--PARAM_ClonePostFix=*)
```

```
PARAM_ClonePostFix="${i#*=}";shift;;
--PARAM_SnapPostFix=*)
PARAM_SnapPostFix="${i#*=}";shift;;
--PROP_ClonePostFix=*)
PROP_ClonePostFix="${i#*=}";shift;;
--PROP_SnapPostFix=*)
PROP_SnapPostFix="${i#*=}";shift;;
--SAP_LVM_SRC_SID=*)
SAP_LVM_SRC_SID="${i#*=}";shift;;
--SAP_LVM_TARGET_SID=*)
SAP_LVM_TARGET_SID="${i#*=}";shift;;
*)
# unknown option
;;
esac
done
#If Parameters not provided by the User - defaulting to DefaultPostFix
if [ -z $PARAM_ClonePostFix ]; then PARAM_ClonePostFix=$DefaultPostFix;fi
if [ -z $PARAM_SnapPostFix ]; then PARAM_SnapPostFix=$DefaultPostFix;fi
#Section - Functions
#########################################
#Function Create (Inventory) YML File
#########################################
create_yml_file()
{
echo "ontapservers:">$YAML_TMP
echo " hosts:">>$YAML_TMP
echo "  ${PRIMARY_CLUSTER}:">>$YAML_TMP
echo "   ansible_host: "'"'$PRIMARY_CLUSTER'"'">>$YAML_TMP
echo "   keyfile: "'"'$PRIMARY_KEYFILE'"'">>$YAML_TMP
echo "   certfile: "'"'$PRIMARY_CERTFILE'"'">>$YAML_TMP
echo "   svmname: "'"'$PRIMARY_SVM'"'">>$YAML_TMP
echo "   datavolumename: "'"'$datavolumename'"'">>$YAML_TMP
echo "   snapshotpostfix: "'"'$snapshotpostfix'"'">>$YAML_TMP
echo "   clonepostfix: "'"'$clonepostfix'"'">>$YAML_TMP
}
#Function run ansible-playbook
#########################################
run_ansible_playbook()
{
echo "[DEBUG]: Running ansible playbook
netapp_lama_${HookOperationName}.yml on Volume $datavolumename"
ansible-playbook -i $YAML_TMP
$ANSIBLE_PATH/netapp_lama_${HookOperationName}.yml
}
#Section - Main
```

```
##########################################
#HookOperationName - CloneVolumes
##########################################
if [ $HookOperationName = CloneVolumes ] ;then
#save mount xml for later usage - used in Section FinalizeCloneVolues to
generate the mountpoints
echo "[DEBUG]: saving mount config...."
cp $MOUNT_XML_PATH /tmp/mount_config_${SAPSYSTEMNAME}_${SAPSYSTEM}.xml
#Instance 00 + 01 share the same volumes - clone needs to be done once
if [ $SAPSYSTEM != 01 ]; then
#generating Volume List - assuming usage of qtrees - "IP-
Adress:/VolumeName/qtree"
xmlFile=/tmp/mount_config_${SAPSYSTEMNAME}_${SAPSYSTEM}.xml
if [ -e $TMPFILE ];then rm $TMPFILE;fi
numMounts=`xml_grep --count "/mountconfig/mount" $xmlFile | grep "total: "
| awk '{ print $2 }'`
i=1
while [ $i -le $numMounts ]; do
     xmllint --xpath "/mountconfig/mount[$i]/exportpath/text()" $xmlFile
|awk -F"/" '{print $2}' >>$TMPFILE
i=$((i + 1))
done
DATAVOLUMES=`cat  $TMPFILE |sort -u`
#Create yml file and rund playbook for each volume
for I in $DATAVOLUMES; do
datavolumename="$I"
snapshotpostfix="$PARAM_SnapPostFix"
clonepostfix="$PARAM_ClonePostFix"
create_yml_file
run_ansible_playbook
done
else
echo "[DEBUG]: Doing nothing .... Volume cloned in different Task"
fi
fi
#HookOperationName - PostCloneVolumes
##########################################
if [ $HookOperationName = PostCloneVolumes] ;then
#Reporting Properties back to LaMa Config for Cloned System
echo "[RESULT]:Property:ClonePostFix=$PARAM_ClonePostFix"
echo "[RESULT]:Property:SnapPostFix=$PARAM_SnapPostFix"
#Create MountPoint Config for Cloned Instances and report back to LaMa
according to SAP Note: https://launchpad.support.sap.com/#/notes/1889590
echo "MountDataBegin"
echo '<?xml version="1.0" encoding="UTF-8"?>'
echo "<mountconfig>"
```

```
xmlFile=/tmp/mount_config_${SAPSYSTEMNAME}_${SAPSYSTEM}.xml
numMounts=`xml_grep --count "/mountconfig/mount" $xmlFile | grep "total: "
| awk '{ print $2 }'`
i=1
while [ $i -le $numMounts ]; do
MOUNTPOINT=`xmllint --xpath "/mountconfig/mount[$i]/mountpoint/text()"
$xmlFile`;
        EXPORTPATH=`xmllint --xpath
"/mountconfig/mount[$i]/exportpath/text()" $xmlFile`;
        OPTIONS=`xmllint --xpath "/mountconfig/mount[$i]/options/text()"
$xmlFile`;
#Adopt Exportpath and add Clonepostfix - assuming usage of qtrees - "IP-
Adress:/VolumeName/qtree"
TMPFIELD1=`echo $EXPORTPATH|awk -F":/" '{print $1}'`
TMPFIELD2=`echo $EXPORTPATH|awk -F"/" '{print $2}'`
TMPFIELD3=`echo $EXPORTPATH|awk -F"/" '{print $3}'`
EXPORTPATH=$TMPFIELD1":/"${TMPFIELD2}$PARAM_ClonePostFix"/"$TMPFIELD3
echo -e '\t<mount fstype="nfs" storagetype="NETFS">'
echo -e "\t\t<mountpoint>${MOUNTPOINT}</mountpoint>"
echo -e "\t\t<exportpath>${EXPORTPATH}</exportpath>"
echo -e "\t\t<options>${OPTIONS}</options>"
echo -e "\t</mount>"
i=$((i + 1))
done
echo "</mountconfig>"
echo "MountDataEnd"
#Finished MountPoint Config
#Cleanup Temporary Files
rm $xmlFile
fi
#HookOperationName - ServiceConfigRemoval
##########################################
if [ $HookOperationName = ServiceConfigRemoval ] ;then
#Assure that Properties ClonePostFix and SnapPostfix has been configured
through the provisioning process
if [ -z $PROP_ClonePostFix ]; then echo "[ERROR]: Propertiy ClonePostFix
is not handed over - please investigate";exit 5;fi
if [ -z $PROP_SnapPostFix ]; then echo "[ERROR]: Propertiy SnapPostFix is
not handed over - please investigate";exit 5;fi
#Instance 00 + 01 share the same volumes - clone delete needs to be done
once
if [ $SAPSYSTEM != 01 ]; then
#generating Volume List - assuming usage of qtrees - "IP-
Adress:/VolumeName/qtree"
xmlFile=$MOUNT_XML_PATH
if [ -e $TMPFILE ];then rm $TMPFILE;fi
```

```
numMounts=`xml_grep --count "/mountconfig/mount" $xmlFile | grep "total: "
| awk '{ print $2 }'`
i=1
while [ $i -le $numMounts ]; do
     xmllint --xpath "/mountconfig/mount[$i]/exportpath/text()" $xmlFile
|awk -F"/" '{print $2}' >>$TMPFILE
i=$((i + 1))
done
DATAVOLUMES=`cat  $TMPFILE |sort -u| awk -F $PROP_ClonePostFix '{ print $1
}'`
#Create yml file and rund playbook for each volume
for I in $DATAVOLUMES; do
datavolumename="$I"
snapshotpostfix="$PROP_SnapPostFix"
clonepostfix="$PROP_ClonePostFix"
create_yml_file
run_ansible_playbook
done
else
echo "[DEBUG]: Doing nothing .... Volume deleted in different Task"
fi
#Cleanup Temporary Files
rm $xmlFile
fi
#HookOperationName - ClearMountConfig
#########################################
if [ $HookOperationName = ClearMountConfig ] ;then
        #Assure that Properties ClonePostFix and SnapPostfix has been
configured through the provisioning process
        if [ -z $PROP_ClonePostFix ]; then echo "[ERROR]: Propertiy
ClonePostFix is not handed over - please investigate";exit 5;fi
        if [ -z $PROP_SnapPostFix ]; then echo "[ERROR]: Propertiy
SnapPostFix is not handed over - please investigate";exit 5;fi
        #Instance 00 + 01 share the same volumes - clone delete needs to
be done once
        if [ $SAPSYSTEM != 01 ]; then
                #generating Volume List - assuming usage of qtrees - "IP-
Adress:/VolumeName/qtree"
                xmlFile=$MOUNT_XML_PATH
                if [ -e $TMPFILE ];then rm $TMPFILE;fi
                numMounts=`xml_grep --count "/mountconfig/mount" $xmlFile
| grep "total: " | awk '{ print $2 }'`
                i=1
                while [ $i -le $numMounts ]; do
                        xmllint --xpath
"/mountconfig/mount[$i]/exportpath/text()" $xmlFile |awk -F"/" '{print
```

```
$2}' >>$TMPFILE
                        i=$((i + 1))
                done
                DATAVOLUMES=`cat  $TMPFILE |sort -u| awk -F
$PROP_ClonePostFix '{ print $1 }'`
                #Create yml file and rund playbook for each volume
                for I in $DATAVOLUMES; do
                        datavolumename="$I"
                        snapshotpostfix="$PROP_SnapPostFix"
                        clonepostfix="$PROP_ClonePostFix"
                        create_yml_file
                        run_ansible_playbook
                done
        else
                echo "[DEBUG]: Doing nothing .... Volume deleted in
different Task"
        fi
        #Cleanup Temporary Files
        rm $xmlFile
fi
#Cleanup
#########################################
#Cleanup Temporary Files
if [ -e $TMPFILE ];then rm $TMPFILE;fi
if [ -e $YAML_TMP ];then rm $YAML_TMP;fi
exit 0
```

Ansible Playbook netapp_lama_CloneVolumes.yml

The playbook that is executed during the CloneVolumes step of the LaMa system clone workflow is a combination of `create_snapshot.yml` and `create_clone.yml` (see NetApp Ansible modules - YAML files). This playbook can be easily extended to cover additional use cases like cloning from secondary and clone split operations.

```
root@sap-jump:~# cat /usr/sap/scripts/ansible/netapp_lama_CloneVolumes.yml
---
- hosts: ontapservers
  connection: local
  collections:
    - netapp.ontap
  gather_facts: false
  name: netapp_lama_CloneVolumes
  tasks:
  - name: Create SnapShot
    na_ontap_snapshot:
      state: present
      snapshot: "{{ datavolumename }}{{ snapshotpostfix }}"
      use_rest: always
      volume: "{{ datavolumename }}"
      vserver: "{{ svmname }}"
      hostname: "{{ inventory_hostname }}"
      cert_filepath: "{{ certfile }}"
      key_filepath: "{{ keyfile }}"
      https: true
      validate_certs: false
  - name: Clone Volume
    na_ontap_volume_clone:
      state: present
      name: "{{ datavolumename }}{{ clonepostfix }}"
      use_rest: always
      vserver: "{{ svmname }}"
      junction_path: '/{{ datavolumename }}{{ clonepostfix }}'
      parent_volume: "{{ datavolumename }}"
      parent_snapshot: "{{ datavolumename }}{{ snapshotpostfix }}"
      hostname: "{{ inventory_hostname }}"
      cert_filepath: "{{ certfile }}"
      key_filepath: "{{ keyfile }}"
      https: true
      validate_certs: false
```

Ansible Playbook netapp_lama_ServiceConfigRemoval.yml

The playbook that is executed during the `ServiceConfigRemoval` phase of the LaMa system destroy workflow is combination of `delete_clone.yml` and `delete_snapshot.yml` (see NetApp Ansible modules - YAML files). It must be aligned to the execution steps of the `netapp_lama_CloneVolumes` playbook.

```
root@sap-jump:~# cat
/usr/sap/scripts/ansible/netapp_lama_ServiceConfigRemoval.yml
---
- hosts: ontapservers
  connection: local
  collections:
    - netapp.ontap
  gather_facts: false
  name: netapp_lama_ServiceConfigRemoval
  tasks:
  - name: Delete Clone
    na_ontap_volume:
      state: absent
      name: "{{ datavolumename }}{{ clonepostfix }}"
      use_rest: always
      vserver: "{{ svmname }}"
      wait_for_completion: True
      hostname: "{{ inventory_hostname }}"
      cert_filepath: "{{ certfile }}"
      key_filepath: "{{ keyfile }}"
      https: true
      validate_certs: false
  - name: Delete SnapShot
    na_ontap_snapshot:
      state: absent
      snapshot: "{{ datavolumename }}{{ snapshotpostfix }}"
      use_rest: always
      volume: "{{ datavolumename }}"
      vserver: "{{ svmname }}"
      hostname: "{{ inventory_hostname }}"
      cert_filepath: "{{ certfile }}"
      key_filepath: "{{ keyfile }}"
      https: true
      validate_certs: false
root@sap-jump:~#
```

Ansible Playbook netapp_lama_ClearMountConfig.yml

The playbook, which is executed during the `netapp_lama_ClearMountConfig` phase of the LaMa system refresh workflow is combination of `delete_clone.yml` and `delete_snapshot.yml` (see NetApp Ansible modules - YAML files). It must be aligned to the execution steps of the `netapp_lama_CloneVolumes` playbook.

```
root@sap-jump:~# cat
/usr/sap/scripts/ansible/netapp_lama_ServiceConfigRemoval.yml
---
- hosts: ontapservers
  connection: local
  collections:
    - netapp.ontap
  gather_facts: false
  name: netapp_lama_ServiceConfigRemoval
  tasks:
  - name: Delete Clone
    na_ontap_volume:
      state: absent
      name: "{{ datavolumename }}{{ clonepostfix }}"
      use_rest: always
      vserver: "{{ svmname }}"
      wait_for_completion: True
      hostname: "{{ inventory_hostname }}"
      cert_filepath: "{{ certfile }}"
      key_filepath: "{{ keyfile }}"
      https: true
      validate_certs: false
  - name: Delete SnapShot
    na_ontap_snapshot:
      state: absent
      snapshot: "{{ datavolumename }}{{ snapshotpostfix }}"
      use_rest: always
      volume: "{{ datavolumename }}"
      vserver: "{{ svmname }}"
      hostname: "{{ inventory_hostname }}"
      cert_filepath: "{{ certfile }}"
      key_filepath: "{{ keyfile }}"
      https: true
      validate_certs: false
root@sap-jump:~#
```

Sample Ansible inventory.yml

This inventory file is dynamically built during workflow execution, and it is only shown here for illustration.

```
ontapservers:
 hosts:
  grenada:
   ansible_host: "grenada"
   keyfile: "/usr/sap/scripts/ansible/certs/ontap.key"
   certfile: "/usr/sap/scripts/ansible/certs/ontap.pem"
   svmname: "svm-sap01"
   datavolumename: "HN9_sap"
   snapshotpostfix: " _snap_20221115"
   clonepostfix: "_clone_20221115"
```

Conclusion

The integration of a modern automation framework like Ansible into SAP LaMa provisioning workflows gives customers a flexible solution to address standard or more complex infrastructure requirements.

Where to find additional information

To learn more about the information that is described in this document, review the following documents and/or websites:

- Collections in the NetApp Namespace

  https://docs.ansible.com/ansible/latest/collections/netapp/index.html

- Documentation about Ansible Integration and Sample Ansible Playbooks

  https://github.com/sap-linuxlab/demo.netapp_ontap

- General Ansible and NetApp Integration

  https://www.ansible.com/integrations/infrastructure/netapp

- Blog on integrating SAP LaMa with Ansible

  https://blogs.sap.com/2020/06/08/outgoing-api-calls-from-sap-landscape-management-lama-with-automation-studio/

- SAP Landscape Management 3.0, Enterprise Edition Documentation

  https://help.sap.com/doc/700f9a7e52c7497cad37f7c46023b7ff/3.0.11.0/en-US/4df88a8f418c5059e10000000a42189c.html#loio4df88a8f418c5059e10000000a42189c

- SAP LaMa Documentation – Provider Definitions

  https://help.sap.com/doc/700f9a7e52c7497cad37f7c46023b7ff/3.0.11.0/en-US/bf6b3e43340a4cbcb0c0f3089715c068.html

- SAP LaMa Documentation - Custom Hooks

https://help.sap.com/doc/700f9a7e52c7497cad37f7c46023b7ff/3.0.11.0/en-US/139eca2f925e48738a20dbf0b56674c5.html

- SAP LaMa Documentation - Configuring SAP Host Agent Registered Scripts

  https://help.sap.com/doc/700f9a7e52c7497cad37f7c46023b7ff/3.0.11.0/en-US/250dfc5eef4047a38bab466c295d3a49.html

- SAP LaMa Documentation - Parameters for Custom Operations and Custom Hooks

  https://help.sap.com/doc/700f9a7e52c7497cad37f7c46023b7ff/3.0.11.0/en-US/0148e495174943de8c1c3ee1b7c9cc65.html

- SAP LaMa Documentation - Adaptive Design

  https://help.sap.com/doc/700f9a7e52c7497cad37f7c46023b7ff/3.0.11.0/en-US/737a99e86f8743bdb8d1f6cf4b862c79.html

- NetApp Product Documentation

  https://www.netapp.com/support-and-training/documentation/

Version history

Version	Date	Document version history
Version 1.0	January 2023	Initial release

Automating SAP HANA System Copy and Clone Operations with SnapCenter

TR-4667: Automating SAP HANA System Copy and Clone Operations with SnapCenter

Nils Bauer, NetApp

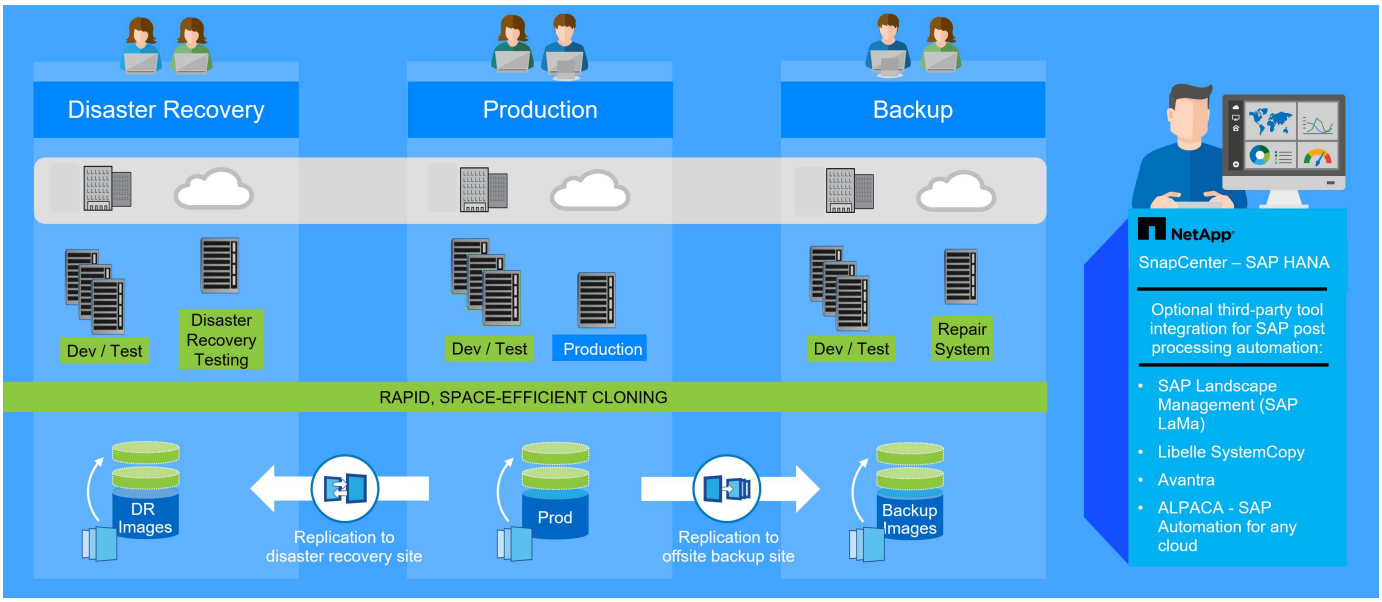
Introduction

In today's dynamic business environment, companies must provide ongoing innovation and react quickly to changing markets. Under these competitive circumstances, companies that implement greater flexibility in their work processes can adapt to market demands more effectively.

Changing market demands also affect a company's SAP environments such that they require regular integrations, changes, and updates. IT departments must implement these changes with fewer resources and over shorter time periods. Minimizing risk when deploying those changes requires thorough testing and training which require additional SAP systems with actual data from production.

Traditional SAP lifecycle-management approaches to provision these systems are primarily based on manual processes. These manual processes are often error-prone and time-consuming, delaying innovation and the response to business requirements.

NetApp solutions for optimizing SAP lifecycle management are integrated into SAP HANA database and lifecycle management tools, combining efficient application-integrated data protection with the flexible provisioning of SAP test systems, as is shown in the following figure. These solutions are available for SAP HANA running on-premises or running in the cloud on Azure NetApp Files (ANF) or Amazon FSx for NetApp ONTAP (FSx for ONTAP).



Application-integrated Snapshot backup operations

The ability to create application-consistent Snapshot backups on the storage layer is the foundation for the system copy and system clone operations described in this document. Storage-based Snapshot backups are created by using the NetApp SnapCenter Plug-In for SAP HANA and interfaces provided by the SAP HANA database. SnapCenter registers Snapshot backups in the SAP HANA backup catalog so that the backups can be used for restore and recovery as well as for cloning operations.

Off-site backup and/or disaster recovery data replication

Application-consistent Snapshot backups can be replicated on the storage layer to an off-site backup site or a disaster recovery site controlled by SnapCenter. Replication is based on changed and new blocks and is therefore space and bandwidth efficient.

Use any Snapshot backup for SAP system copy or clone operations

NetApp technology and software integration allows you to use any Snapshot backup of a source system for an SAP system copy or clone operation. This Snapshot backup can be either selected from the same storage that is used for the SAP production systems, the storage that is used for off-site backups, or the storage at the disaster recovery site. This flexibility allows you to separate development and test systems from production if required and covers other scenarios, such as the testing of disaster recovery at the disaster recovery site.

> Cloning from the off-site backup or disaster recovery storage is supported for on-premises NetApp systems and for Amazon FSx for NetApp ONTAP. With Azure NetApp Files clones can only be created at the source volume.

Automation with integration

There are various scenarios and use cases for the provisioning of SAP test systems, and you might also have different requirements for the level of automation. NetApp software products for SAP integrate into database

and lifecycle management products from SAP to support different scenarios and levels of automation.

NetApp SnapCenter with the plug-in for SAP HANA is used to provision the required storage volumes based on an application-consistent Snapshot backup and to execute all required host and database operations up to a started SAP HANA database. Depending on the use case, SAP system copy, system clone, system refresh, or additional manual steps such as SAP postprocessing might be required. More details are covered in the next section.

A fully automated, end-to-end provision of SAP test systems can be performed by using third-party tools and integration of NetApp features. More details are available at:

TR-4953: NetApp SAP Landscape Management Integration using Ansible

TR-4929: Automating SAP system copy operations with Libelle SystemCopy (netapp.com)

Automating SAP system copy, refresh, and clone workflows with ALPACA and NetApp SnapCenter

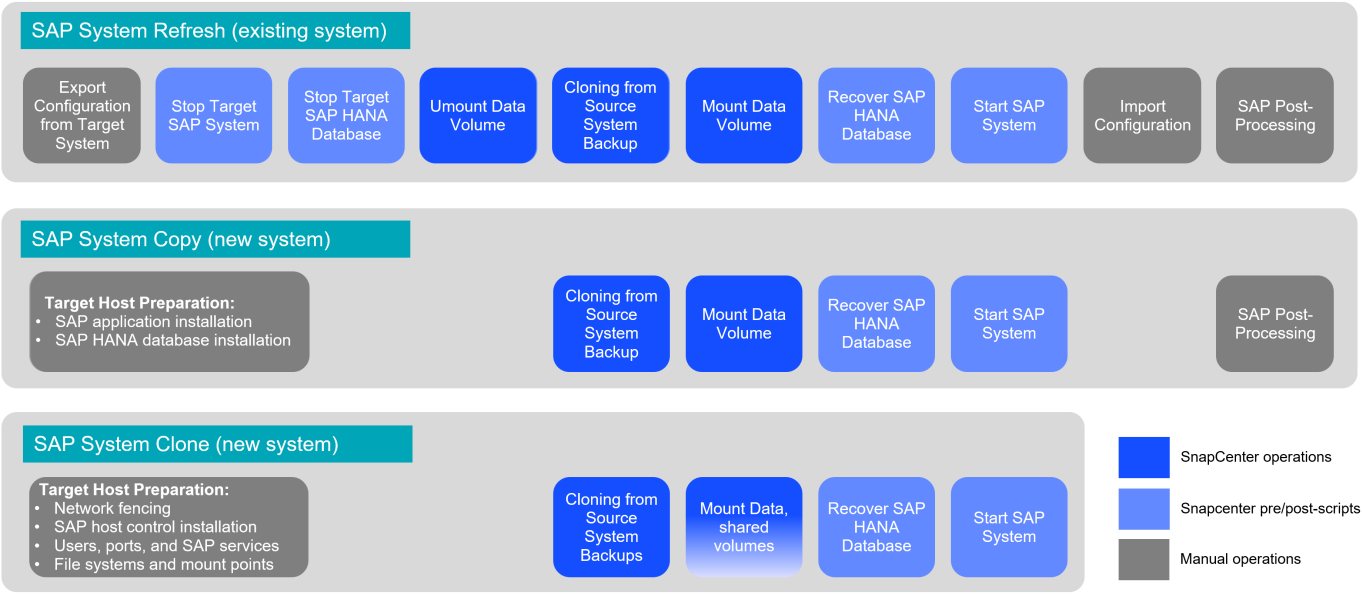
Automating SAP system copy, refresh, and clone workflows with Avantra and NetApp SnapCenter

SAP system copy, refresh, and clone scenarios

The term SAP system copy is often used as a synonym for three different processes: SAP system refresh, SAP system copy, or SAP system clone operations. It is important to distinguish between the different operations because the workflows and use cases differ for each one.

- **SAP system refresh.** An SAP system refresh is a refresh of an existing target SAP system with data from a source SAP system. The target system is typically part of an SAP transport landscape, for example a quality assurance system, that is refreshed with data from the production system. The hostname, instance number, and SID are different for the source and target systems.
- **SAP system copy.** An SAP system copy is a setup of a new target SAP system with data from a source SAP system. The new target system could be, for example, an additional test system with data from the production system. The hostname, instance number, and SID are different for the source and target systems.
- **SAP system clone.** An SAP system clone is an identical clone of a source SAP system. SAP system clones are typically used to address logical corruption or to test disaster recovery scenarios. With a system clone operation, the hostname, instance number, and SID remain the same. It is therefore important to establish proper network fencing for the target system to make sure that there is no communication with the production environment.

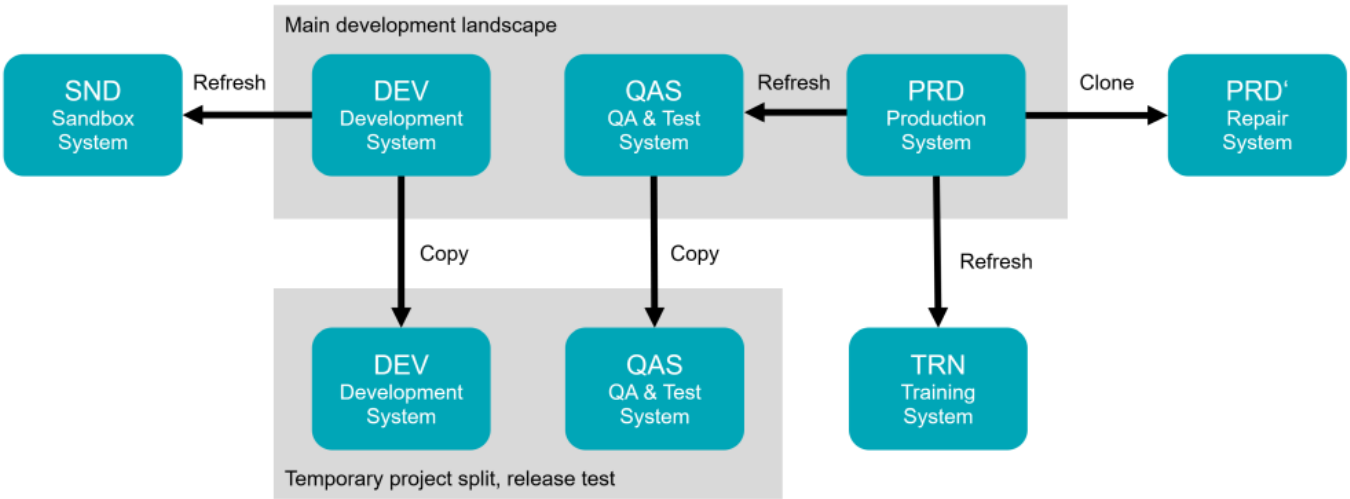
The figure below illustrates the main steps that must be performed during a system refresh, system copy, or system clone operation. The blue boxes indicate steps that can be automated with SnapCenter, while the gray boxes indicate steps that must be performed outside of SnapCenter, either manually or by using third-party tools.



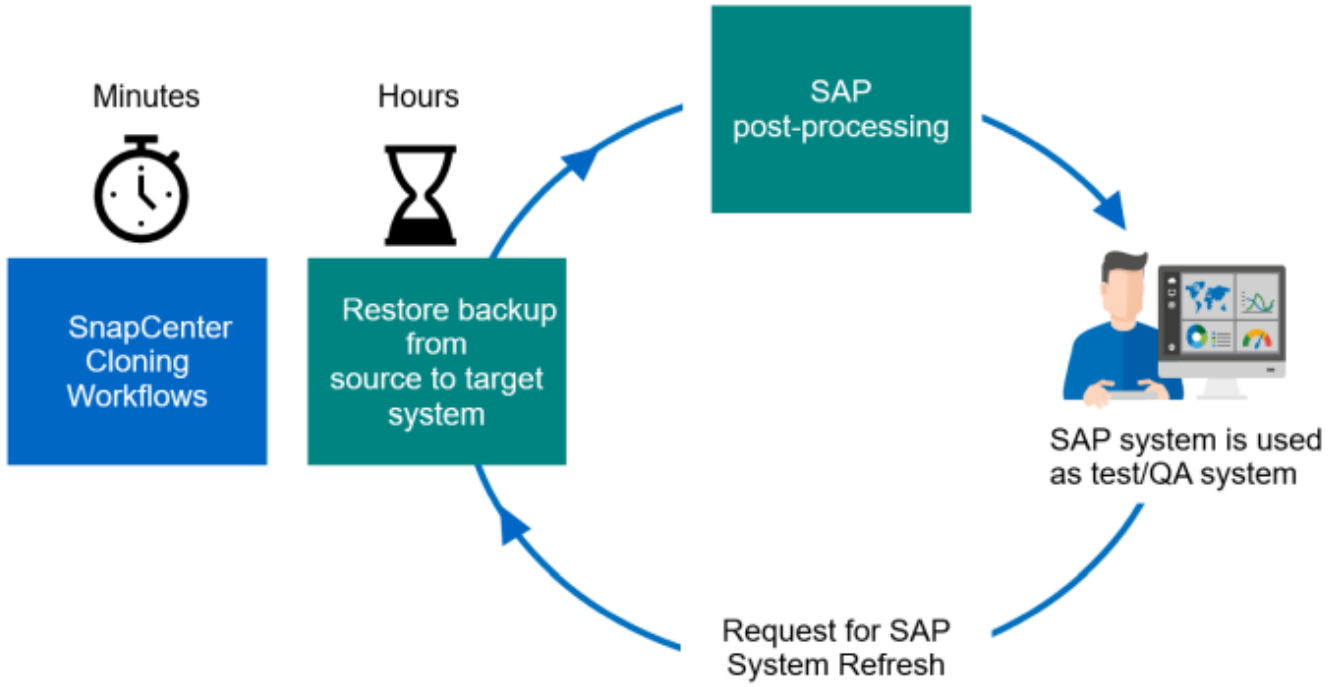
Use cases for system refresh and cloning

Data refresh of QA, test, sandbox, or training systems

There are multiple scenarios in which data from a source system must be made available to a target system for testing or training purposes. These test and training systems must be updated with data from the source system on a regular basis to make sure that testing and training is performed with the current data set. These system refresh operations consist of multiple tasks on the infrastructure, database, and application layers, and they can take multiple days depending on the level of automation.



SnapCenter cloning workflows can be used to accelerate and automate the required tasks at the infrastructure and database layers. Instead of restoring a backup from the source system to the target system, SnapCenter uses NetApp Snapshot copy and NetApp FlexClone technology, so that required tasks up to a started SAP HANA database can be performed in minutes instead of hours. The time needed for the cloning process is independent from the size of the database, therefore even very large systems can be created in a couple of minutes. The startup time just depends on the database size and the connectivity between the database server and the storage system.



The workflow for system-refresh operations is described in the section "SAP HANA system refresh with SnapCenter."

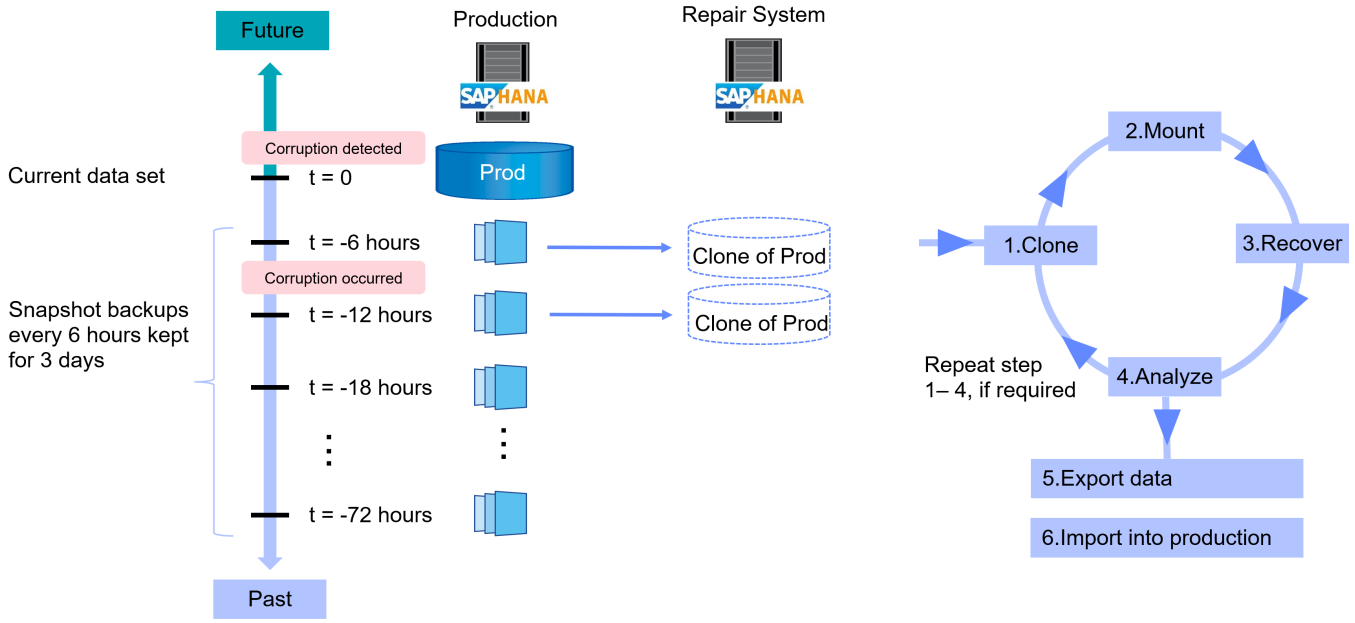
Address logical corruption

Logical corruption can be caused by software errors, human errors, or sabotage. Unfortunately, logical corruption often cannot be addressed with standard high-availability and disaster recovery solutions. As a result, depending on the layer, application, file system, or storage where the logical corruption occurred, minimal downtime and maximum data loss requirements can sometimes not be fulfilled.

The worst case is logical corruption in an SAP application. SAP applications often operate in a landscape in which different applications communicate with each other and exchange data. Therefore, restoring and recovering an SAP system in which a logical corruption has occurred is not the recommended approach. Restoring the system to a point in time before the corruption occurred results in data loss. Also, the SAP landscape would no longer be in sync and would require additional postprocessing.

Instead of restoring the SAP system, the better approach is to try to fix the logical error within the system by analyzing the problem in a separate repair system. Root cause analysis requires the involvement of the business process and application owner. For this scenario, you create a repair system (a clone of the production system) based on data stored before the logical corruption occurred. Within the repair system, the required data can be exported and imported to the production system. With this approach, the production system does not need to be stopped, and, in the best-case scenario, no data or only a small fraction of data is lost.

When setting up the repair system, flexibility and agility is crucial. When using NetApp storage-based Snapshot backups, multiple consistent database images are available to create a clone of the production system by using NetApp FlexClone technology. FlexClone volumes can be created in a matter of seconds rather than multiple hours if a redirected restore from a file-based backup is used to set up the repair system.



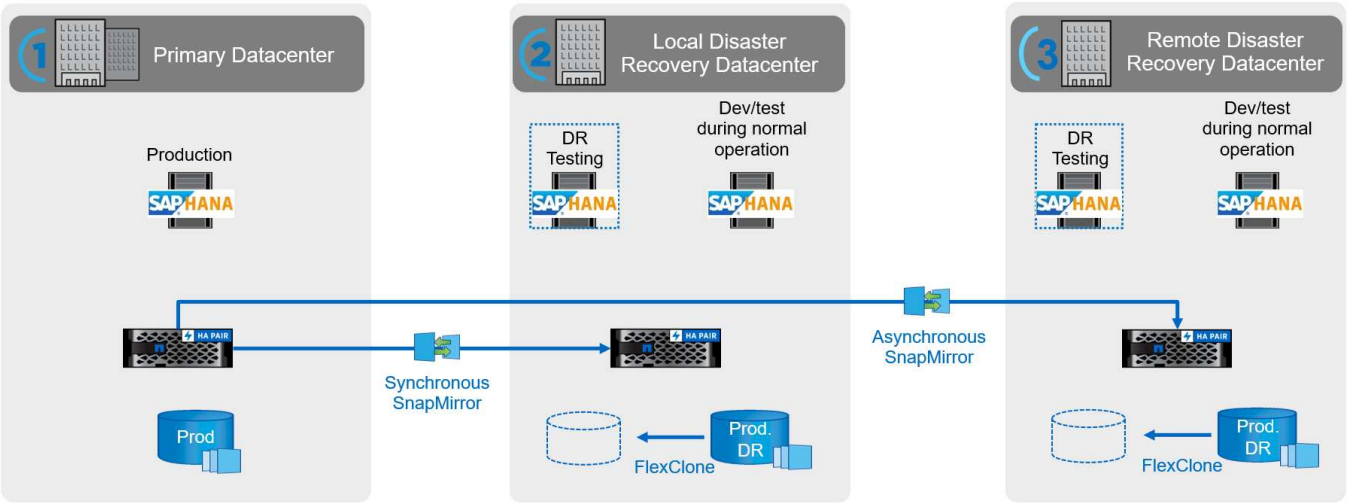
The workflow of the repair system creation is described in the section "SAP system clone with SnapCenter."

Disaster recovery testing

An effective disaster recovery strategy needs testing the required workflow. Testing demonstrates whether the strategy works and whether the internal documentation is sufficient. It also allows administrators to train the required procedures.

Storage replication with SnapMirror makes it possible to execute disaster recovery testing without putting RTO and RPO at risk. Disaster recovery testing can be performed without interrupting data replication.

Disaster recovery testing for both asynchronous and synchronous SnapMirror uses Snapshot backups and FlexClone volumes at the disaster recovery target.



A detailed step-by-step description can be found in the technical reports

TR-4646: SAP HANA Disaster Recovery with Storage Replication (netapp.com)

TR-4891: SAP HANA disaster recovery with Azure NetApp Files

Supported infrastructure and scenarios

This document covers SAP system refresh and cloning scenarios for SAP HANA systems running on on-premises NetApp systems, on Amazon FSx for NetApp ONTAP systems and on Azure NetApp Files. However not all features and scenarios are available on every storage platform. The table below summarizes the supported configurations.

Within the document, we are using an SAP HANA landscape running on on-premises NetApp systems with NFS as the storage protocol. Most workflow steps are identical across the different platforms, and if there are differences, they are highlighted in this document.

	On-premises NetApp systems	**AWS FSx for NetApp ONTAP**	**Azure NetApp Files**
Storage protocol	NFS, Fibre Channel	NFS	NFS
Thin clone (FlexClone)	Yes	Yes	No, with the current ANF version, cloned volume is always split
Clone split operation	Yes	Yes	N/A
Cloning from primary	Yes	Yes	Yes
Cloning from off-site backup	Yes	Yes	No
Cloning at DR site	Yes	Yes	Yes, but not integrated into SnapCenter

Overview of SAP system refresh workflow with SnapCenter

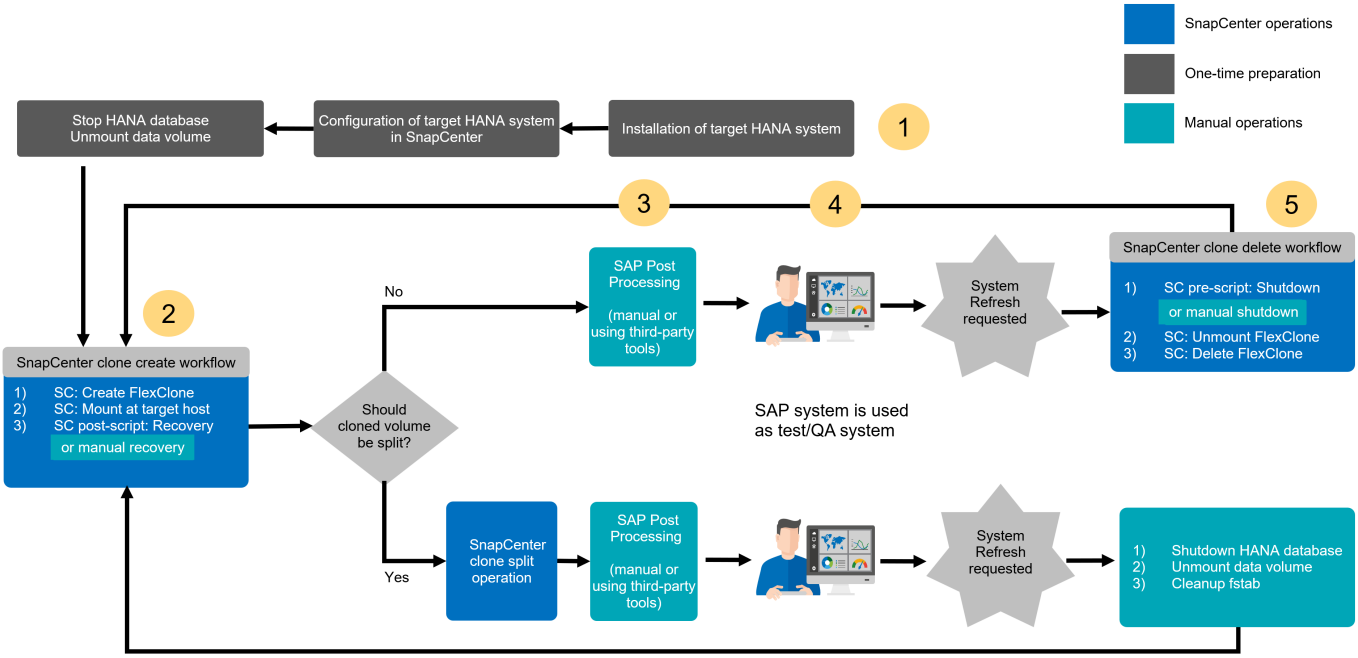
SnapCenter provides workflows that allow you to manage clones of data sets from any existing Snapshot backup. This cloned data set, a FlexClone volume, can be used to rapidly provision a HANA data volume from a source system and attach it to a target system. It is therefore a perfect fit for executing system refresh operations for QA, test, sandbox, or training systems.

The SnapCenter cloning workflows handle all required operations on the storage layer and can be extended using scripts to execute host-specific and HANA database-specific operations. In this document, we use a script to perform HANA database recovery and shutdown operations. SnapCenter workflows with further automation using the script handle all required HANA database operations but do not cover any required SAP post-processing steps. SAP post processing must be performed manually or with third-party tools.

The SAP system refresh workflow with SnapCenter consists of five main steps as shown in the below figure.

1. A one-time, initial installation and preparation of the target system
    a. The SnapCenter HANA plugin must be installed on the new target system and the HANA system must be configured in SnapCenter
    b. The target system must be stopped, and the HANA data volume must be unmounted
2. The SnapCenter clone create workflow
    a. SnapCenter creates a FlexClone volume of the selected Snapshot of the source system
    b. SnapCenter mounts the FlexClone volume at the target system
    c. Recovery of the target HANA database can be automated using the `sc-system-refresh` script as a post-script or can be executed manually

3. SAP post processing (manual or with a third-party tool)
4. The system can now be used as test/QA system.
5. When a new system refresh is requested, the SnapCenter clone delete workflow is used to remove the FlexClone volume
    a. If the target HANA system has been protected in SnapCenter, the protection must be removed before the clone delete workflow is started.
    b. The HANA system must be stopped manually or stopped automatically using the `sc-system-refresh` script as a SnapCenter pre-script
    c. SnapCenter unmounts the HANA data volume
    d. SnapCenter deletes the FlexClone volume
    e. A refresh is restarted with step 2.



In most cases, target test/QA systems are used for at least a couple of weeks. Since the FlexClone volume is blocking the Snapshot of the source system volume, this Snapshot will require additional capacity based on the block change rate at the source system volume. For production source systems and an average change rate of 20% per day, the blocked Snapshot will reach 100% after 5 days. Therefore, NetApp recommends splitting the FlexClone volume either immediately or after a couple of days, if the clone is based on a production source system. The clone split operation does not block use of the cloned volume and can therefore be performed at any time while the HANA database is in use.

> When splitting the FlexClone volume, SnapCenter deletes all backups that were created at the target system.

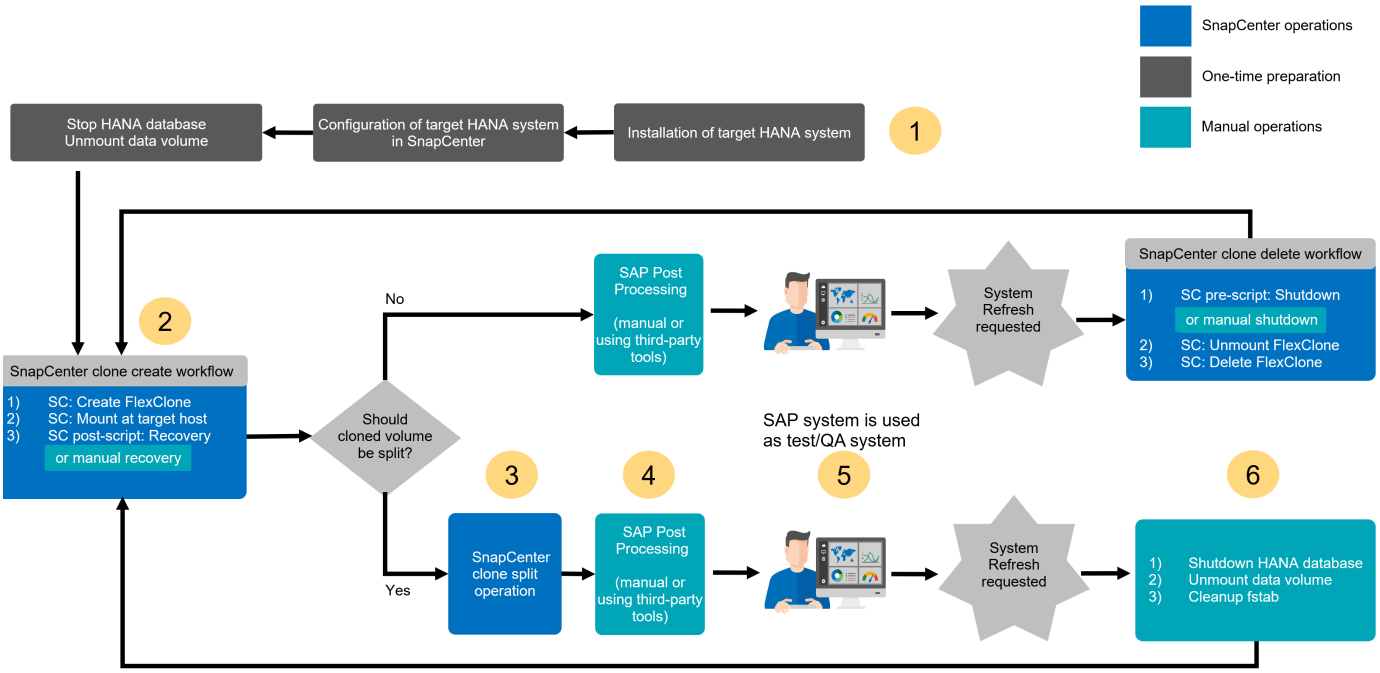
> With SnapCenter and Azure NetApp Files, the clone split operation is not available, since Azure NetApp Files always splits the clone after creation.

The refresh operation including the clone split consists of the following steps.

1. A one-time, initial installation and preparation of the target system

   a. The SnapCenter HANA plugin must be installed on the new target system and the HANA system must be configured in SnapCenter
   b. The target system must be stopped, and the HANA data volume must be unmounted
2. The SnapCenter clone create workflow
   a. SnapCenter creates a FlexClone volume of the selected Snapshot of the source system
   b. SnapCenter mounts the FlexClone volume at the target system
   c. Recovery of the target HANA database can be automated using the `sc-system-refresh` script as a post-script or can be executed manually
3. The FlexClone volume is split using the SnapCenter clone split workflow.
4. SAP post processing (manual or with a third-party tool)
5. The system can now be used as test/QA system.
6. When a new system refresh is requested, the cleanup is done with the following manual steps
   a. If the target HANA system has been protected in SnapCenter, the protection must be removed.
   b. The HANA system must be stopped manually
   c. The HANA data volume must be unmounted and the fstab entry from SnapCenter must be removed (manual task)
   d. A refresh is restarted with step 2.

> The old data volume, which was split previously, must be deleted manually on the storage system.



The section "SAP HANA system refresh with SnapCenter" provides a detailed step-by-step description of both system-refresh workflows.

Overview of SAP system clone workflow with SnapCenter

As discussed in the previous section, SnapCenter can manage clones of data sets from any existing Snapshot backup and can rapidly provision these data sets to any target system. The flexible and agile provisioning of production data to a repair system to address logical corruption is critical, since it is often necessary to reset the repair system and to choose a different production data set. FlexClone technology enables a rapid provisioning process, and provides significant capacity savings, since the repair system is typically only used for a short time.

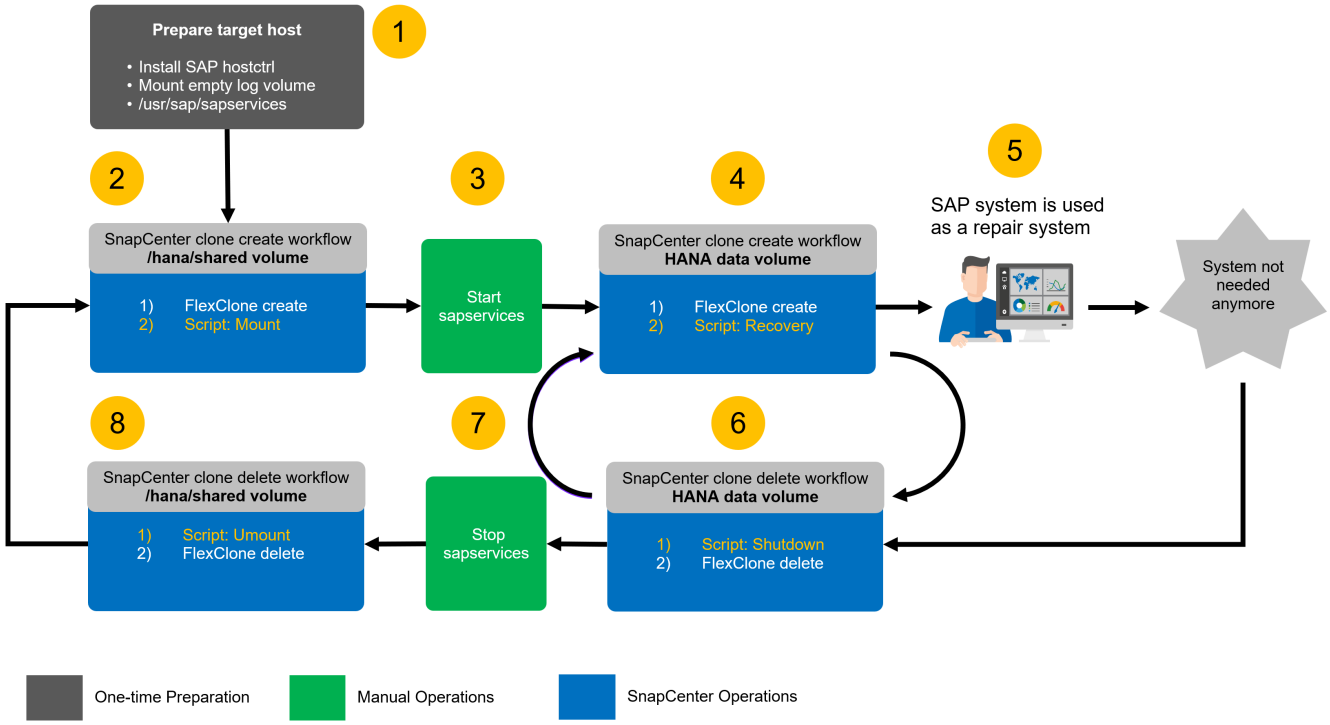
The figure below summarizes the required steps for an SAP system clone operation using SnapCenter.

1. Prepare the target host.
2. SnapCenter clone create workflow for the SAP HANA shared volume.
3. Start SAP HANA services.
4. SnapCenter clone create workflow for the SAP HANA data volume including database recovery.
5. The SAP HANA system can now be used as a repair system.

If the system is not needed anymore, the clean-up process is performed with the following steps.

6. SnapCenter clone delete workflow for the SAP HANA data volume including database shutdown (when using the automation script).
7. Stop SAP HANA services.
8. SnapCenter clone delete workflow for the SAP HANA shared volume.

> If you must reset the system to a different Snapshot backup, then step 6 and step 4 are sufficient. A refresh of the SAP HANA shared volume is not required.



The section "SAP system clone with SnapCenter" provides a detailed step-by-step description of the system clone workflow.

Considerations for SAP HANA system refresh operations using storage snapshot backups

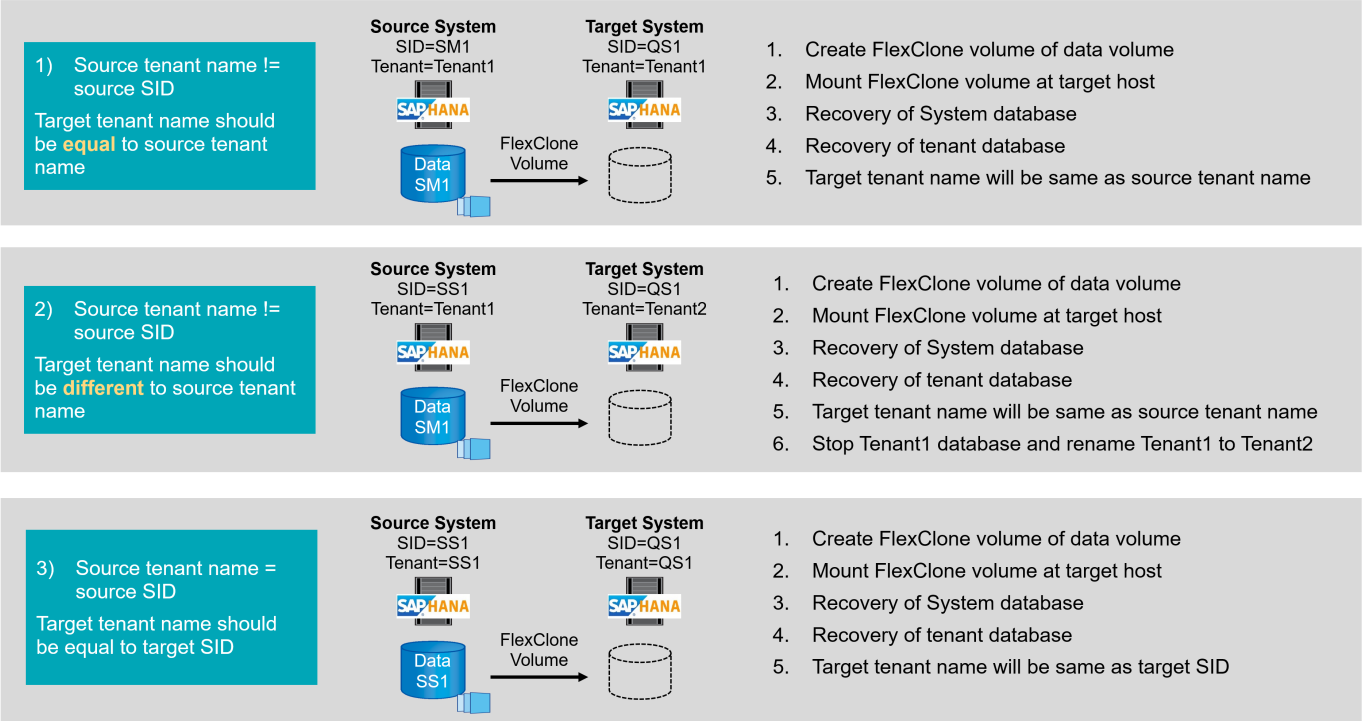
Tenant name(s) at target system

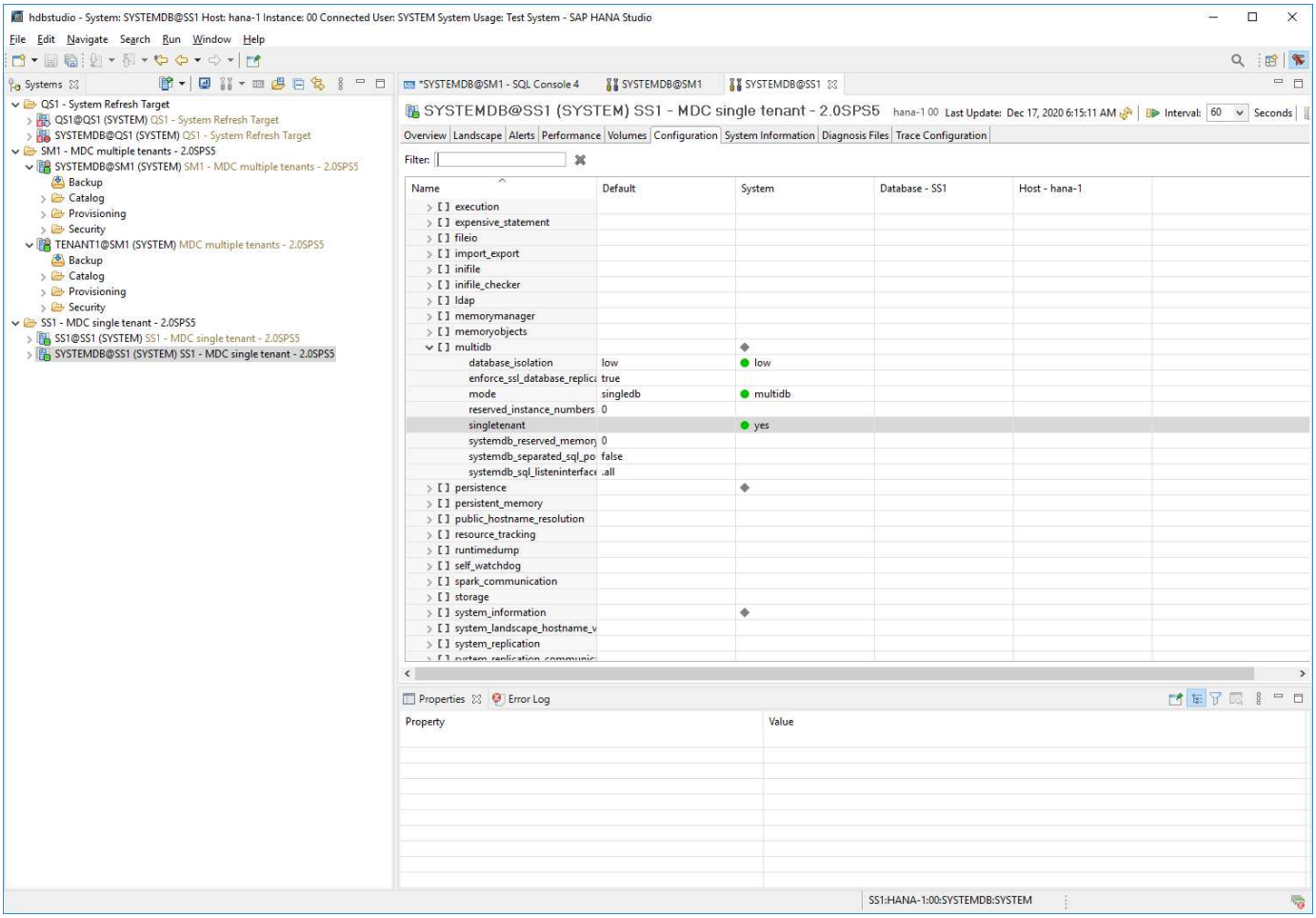
The steps required to perform an SAP HANA system refresh depend on the source system tenant configuration and the required tenant name at the target system, as shown in the figure below.

Since the tenant name is configured in the system database, the tenant name of the source system is also available at the target system after the recovery of the system database. Therefore, the tenant at the target system can only be recovered with the same name as the source tenant as shown in option 1. If the tenant name at the target system must be different, the tenant must first be recovered with the same name as the source tenant and then renamed to the required target tenant name. This is option 2.

An exception of this rule is an SAP HANA system with a single tenant, where the tenant name is identical to the system SID. This configuration is the default after an initial SAP HANA installation. This specific configuration is flagged by the SAP HANA database. In this case, tenant recovery at the target system can be executed with the tenant name of the target system, which must be also identical to the system SID of the target system. This workflow is shown in option 3.

> As soon as any tenant create, rename, or drop operation is executed at the source system, this configuration flag is deleted by the SAP HANA database. Therefore, even if the configuration has been brought back to tenant = SID, the flag is no longer available and the exception regarding tenant recovery with workflow 3 is no longer possible. In this case, option 2 is the required workflow.





System refresh workflow with enabled SAP HANA encryption

When SAP HANA persistence encryption is enabled, additional steps are required before you can recover the SAP HANA database at the target system.

At the source system you need to create a backup of the encryption root keys for the system database, as well as for all tenant databases. The backup files must be copied to the target system and the root keys must be imported from the backup before the recovery operation is executed.

See also SAP HANA Administration Guide.

Backup of root keys

A backup of the root keys is always required, if any changes to the root keys have been made.
The backup command requires the dbid as a CLI parameter. The dbid's can be identified using the below SQL statement.

SYSTEMDB@SS1 (SYSTEM) hana-1 00

SQL Result

```
SELECT DATABASE_NAME,
    CASE WHEN (DBID = '' AND
    DATABASE_NAME = 'SYSTEMDB')
    THEN 1
    WHEN (DBID = '' AND
    DATABASE_NAME <> 'SYSTEMDB')
    THEN 3
    ELSE TO_INT(DBID)
    END DATABASE_ID
FROM (SELECT DISTINCT DATABASE_NAME, SUBSTR_AFTER (SUBPATH,'.') AS DBID FROM SYS_DATABASES.M_VOLUMES)
```

	DATABASE_NAME	DATABASE_ID
1	SYSTEMDB	1
2	SS1	3

The SQL statement and further documentation is available in the SAP HANA Admin Guide at Back Up Root Keys | SAP Help Portal

The following steps are illustrating the required operations for a HANA system with a single tenant SS1 and are executed at the source system.

1. Set backup password for system and tenant (SS1) databases (if not done yet).

```
hdbsql SYSTEMDB=> ALTER SYSTEM SET ENCRYPTION ROOT KEYS BACKUP PASSWORD
Netapp123;
0 rows affected (overall time 3658.128 msec; server time 3657.967 msec)
hdbsql SYSTEMDB=>
hdbsql SS1=> ALTER SYSTEM SET ENCRYPTION ROOT KEYS BACKUP PASSWORD
Netapp123;
0 rows affected (overall time 2424.236 msec; server time 2424.010 msec)
hdbsql SS1=>
```

2. Create backup of root keys for system and tenant (SS1) databases.

```
ss1adm@hana-1:/usr/sap/SS1/home> /usr/sap/SS1/HDB00/exe/hdbnsutil
-backupRootKeys root-key-backup-SS1-SYSTEMDB.rkb --dbid=1 --type='ALL'
Exporting root key backup for database SYSTEMDB (DBID: 1) to
/usr/sap/SS1/home/root-key-backup-SS1-SYSTEMDB.rkb
done.
ss1adm@hana-1:/usr/sap/SS1/home> /usr/sap/SS1/HDB00/exe/hdbnsutil
-backupRootKeys root-key-backup-SS1-SS1.rkb --dbid=3 --type='ALL'
Exporting root key backup for database SS1 (DBID: 3) to
/usr/sap/SS1/home/root-key-backup-SS1-SS1.rkb
done.
```

3. Validate root key backups (optional)

```
ss1adm@hana-1:/usr/sap/SS1/home> ls -al root*
-rw-r----- 1 ss1adm sapsys 1440 Apr 24 07:00 root-key-backup-SS1-SS1.rkb
-rw-r----- 1 ss1adm sapsys 1440 Apr 24 06:54 root-key-backup-SS1-
SYSTEMDB.rkb
ss1adm@hana-1:/usr/sap/SS1/home>

ss1adm@hana-1:/usr/sap/SS1/home> /usr/sap/SS1/HDB00/exe/hdbnsutil
-validateRootKeysBackup root-key-backup-SS1-SS1.rkb
Please Enter the password:
Successfully validated SSFS backup file /usr/sap/SS1/home/root-key-backup-
SS1-SS1.rkb
done.

ss1adm@hana-1:/usr/sap/SS1/home> /usr/sap/SS1/HDB00/exe/hdbnsutil
-validateRootKeysBackup root-key-backup-SS1-SYSTEMDB.rkb
Please Enter the password:
Successfully validated SSFS backup file /usr/sap/SS1/home/root-key-backup-
SS1-SYSTEMDB.rkb
done.
```

Import of root keys at the target system

The import of the root keys is required initially for the first system refresh operation. If the root keys are not changed at the source system, no additional import is required.
The import command requires the dbid as a CLI parameter. The dbid's can be identified in the same way as described for the root key backup.

1. In our setup the root keys are copied from the source system to an NFS share

```
hana-1:~ # cp /usr/sap/SS1/home/root-key-backup-SS1-SS1.rkb /mnt/sapcc-
share/SAP-System-Refresh/
hana-1:~ # cp /usr/sap/SS1/home/root-key-backup-SS1-SYSTEMDB.rkb
/mnt/sapcc-share/SAP-System-Refresh/
```

2. The root keys can now be imported using hdbnsutil. The dbid for the system and tenant database must be provided with the command. The backup password is also required.

```
qs1adm@hana-7:/usr/sap/QS1/HDB11> ./exe/hdbnsutil -recoverRootKeys
/mnt/sapcc-share/SAP-System-Refresh/root-key-backup-SS1-SYSTEMDB.rkb
--dbid=1 --type=ALL
Please Enter the password:
Importing root keys for DBID: 1 from /mnt/sapcc-share/SAP-System-
Refresh/root-key-backup-SS1-SYSTEMDB.rkb
Successfully imported root keys from /mnt/sapcc-share/SAP-System-
Refresh/root-key-backup-SS1-SYSTEMDB.rkb
done.

qs1adm@hana-7:/usr/sap/QS1/HDB11> ./exe/hdbnsutil -recoverRootKeys
/mnt/sapcc-share/SAP-System-Refresh/root-key-backup-SS1-SS1.rkb --dbid=3
--type=ALL Please Enter the password:
Importing root keys for DBID: 3 from /mnt/sapcc-share/SAP-System-
Refresh/root-key-backup-SS1-SS1.rkb
Successfully imported root keys from /mnt/sapcc-share/SAP-System-
Refresh/root-key-backup-SS1-SS1.rkb
done.
qs1adm@hana-7:/usr/sap/QS1/HDB11>
```

Root key import, if dbid does not exist at target

As described in the chapter before, the dbid is required to import the root key for the system and all tenant databases. While the system database has always dbid=0, the tenant databases can have different dbid's.

```
SQL  Result
SELECT DATABASE_NAME,
   CASE WHEN (DBID = '' AND
   DATABASE_NAME = 'SYSTEMDB')
   THEN 1
   WHEN (DBID = '' AND
   DATABASE_NAME <> 'SYSTEMDB')
   THEN 3
   ELSE TO_INT(DBID)
   END DATABASE_ID
FROM (SELECT DISTINCT DATABASE_NAME, SUBSTR_AFTER (SUBPATH,'.') AS DBID FROM SYS_DATABASES.M_VOLUMES)
```

	DATABASE_NAME	DATABASE_ID
1	TENANT1	4
2	SYSTEMDB	1
3	TENANT2	3

The output above shows two tenants with dbid=3 and dbid=4. If the target system has not yet hosted a tenant with dbid=4, the import of the root key will fail. In that case you need to recover the system database first and then import the key for the tenant with dbid=4.

Automation example scripts

In this document, two scripts are used to further automate SnapCenter clone create and clone delete operations.

- The script `sc-system-refresh`.sh is used for the system refresh and the system clone workflow to

execute recovery and shutdown operations of the SAP HANA database.

- The script `sc-mount-volume.sh` is used for the system clone workflow to execute mount and unmount operations for the SAP HANA shared volume.

> The example scripts are provided as is and are not supported by NetApp. You can request the scripts via email to ng-sapcc@netapp.com.

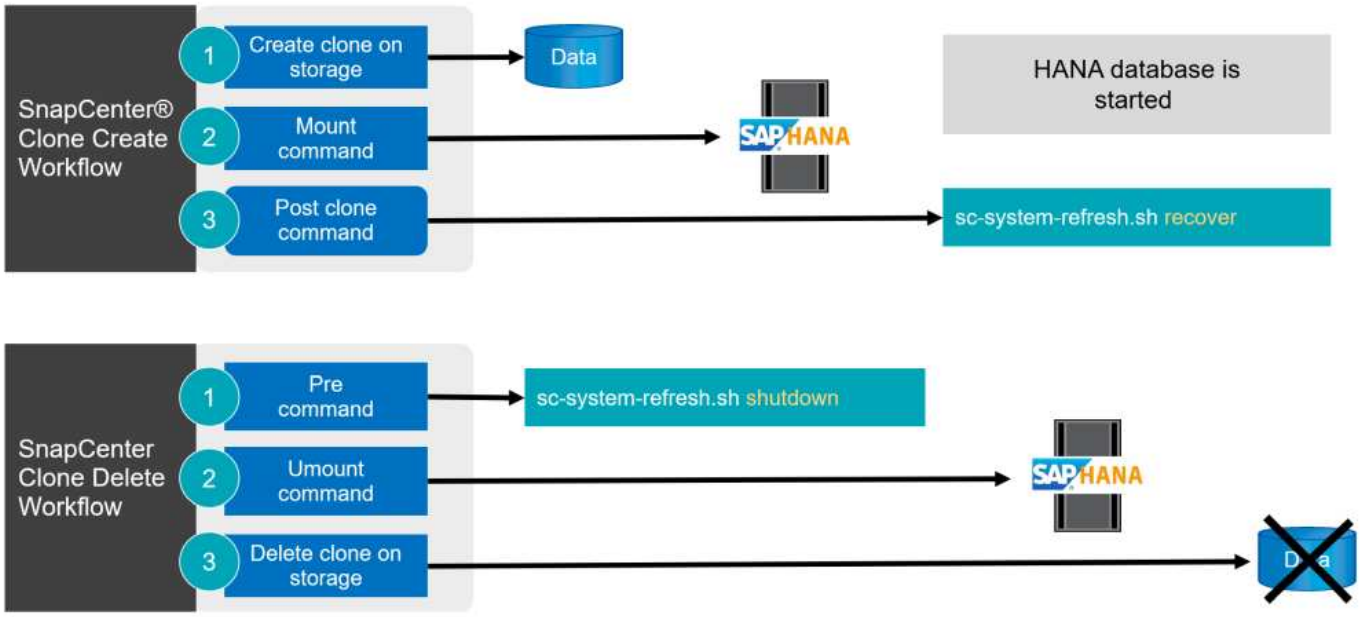
Script sc-system-refresh.sh

The example script `sc-system-refresh.sh` is used to execute recovery and shutdown operations. The script is called with specific command-line options within the SnapCenter workflows clone create and clone delete, as shown in the figure below.

The script is generic and reads all required parameters, like the SID from the target system. The script must be available at the target host of the system refresh operation. An hdb user store key must be configured for the user <SID>adm at the target system. The key must allow access to the SAP HANA system database and provide privileges for recovery operations. The key must have the name <TARGET-SID>KEY.

The script writes a log file `sc-system-refresh-SID.log`, to the same directory, where it gets executed.

> The current version of the script supports single host systems MDC single tenant, or MDC multiple tenant configurations. It does not support SAP HANA multiple-host systems.



Supported tenant recovery operations

As described in the section “SAP HANA system refresh operation workflows using storage snapshot” the possible tenant recovery operations at the target system depend on the tenant configuration of the source system. The script `sc-system-refresh.sh` supports all tenant recovery operations, which are possible dependent on the source system configuration, as shown in the table below.

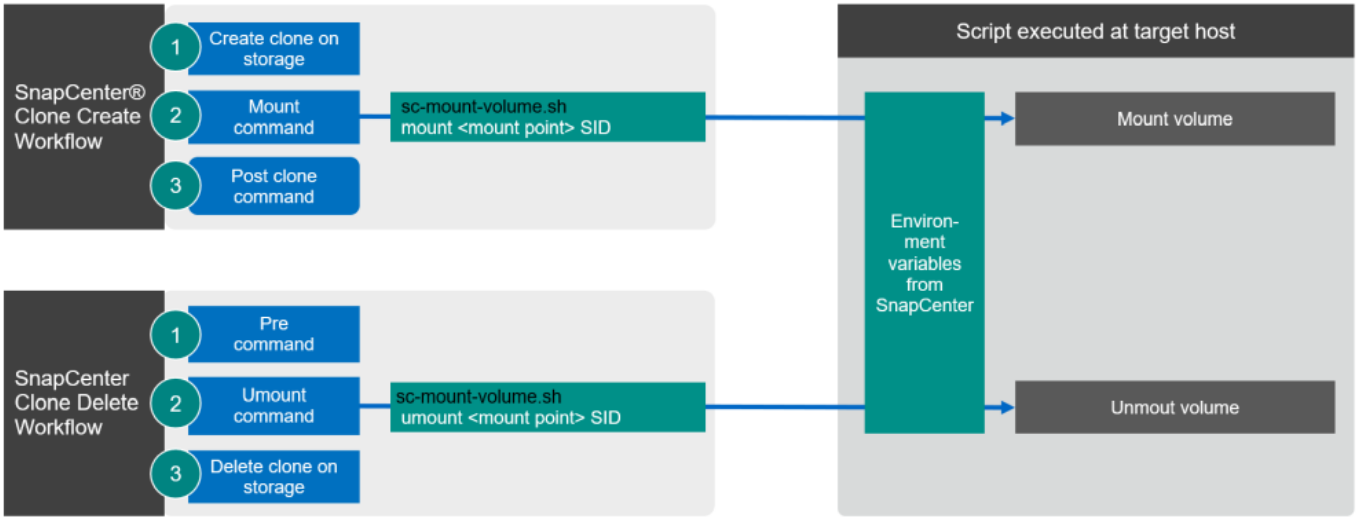
If a different tenant name is required at the target system, the tenant must be renamed manually after the recovery operation.

SAP HANA system	Tenant configuration at source system	Resulting tenant configuration at target system
MDC single tenant	Source tenant name equal to source SID	Target tenant name is equal to target SID
MDC single tenant	Source tenant name not equal to source SID	Target tenant name is equal to source tenant name
MDC multiple tenants	Any tenant names	All tenants are recovered and will have the same name as the source tenants.

Script sc-mount-volume.sh

The example script `sc-mount-volume.sh` is used to execute mount and unmount for any volume. The script is used to mount the SAP HANA shared volume with the SAP HANA system clone operation. The script is called with specific command-line options within the SnapCenter workflows clone create and clone delete, as shown in the figure below.

> The script supports SAP HANA systems using NFS as a storage protocol.



SnapCenter environment variables

SnapCenter provides a set of environment variables that are available within the script that is executed at the target host. The script uses these variables to determine relevant configuration settings.

- The script variables `STORAGE, JUNCTION_PATH` are used for the mount operation.
- Derived from `CLONED_VOLUMES_MOUNT_PATH` environment variable.
- `CLONED_VOLUMES_MOUNT_PATH=${STORAGE}:/${JUNCTION_PATH}`
- For example: `CLONED_VOLUMES_MOUNT_PATH=192.168.175.117:/SS1_shared_Clone_05112206115489411`

Script to get SnapCenter environment variables

If the automation scripts should not be used and the steps should be executed manually, you need to know the storage system junction path of the FlexClone volume. The junction path is not visible within SnapCenter, so

you need to either look up the junction path directly at the storage system, or you could use a simple script that provides the SnapCenter environment variables at the target host. This script needs to be added as a mount operation script within the SnapCenter clone create operation.

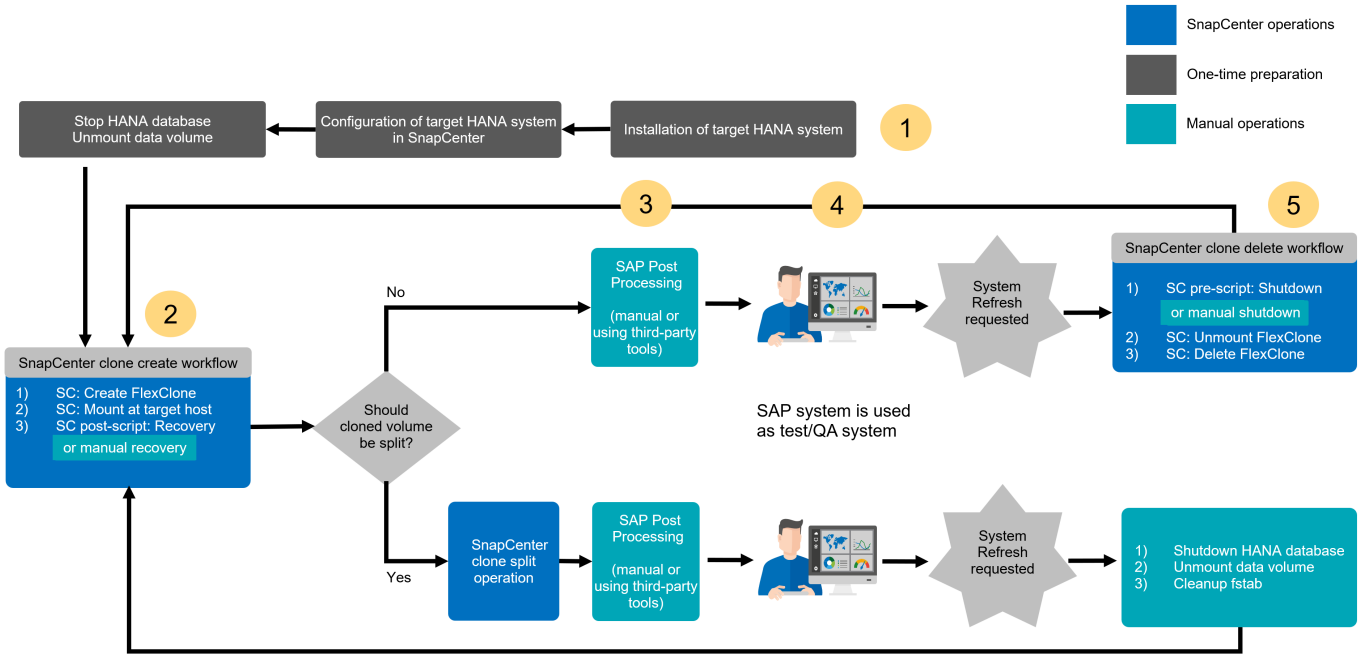
```
ss1adm@hana-1:/mnt/sapcc-share/SAP-System-Refresh> cat get-env.sh
#!/bin/bash
env > /tmp/env-from-sc.txt
ss1adm@hana-1:/mnt/sapcc-share/SAP-System-Refresh>
```

Within the `env-from-sc.txt` file, look for the variable `CLONED_VOLUMES_MOUNT_PATH` to get the storage system IP address and junction path of the FlexClone volume.

For example:
`CLONED_VOLUMES_MOUNT_PATH=192.168.175.117:/SS1_data_mnt00001_Clone_05112206115489411`

SAP HANA system refresh with SnapCenter

The following section provides a step-by-step description for the different system refresh operation options of an SAP HANA database.



Depending on the SAP HANA database configuration additional steps are executed or need to be prepared. The table below provides a summary.

Source system	Source system configuration	SnapCenter and SAP HANA operations
MDC single tenant SID = tenant name	Standard configuration	SnapCenter clone operation and optional recovery script execution.

Source system	Source system configuration	SnapCenter and SAP HANA operations
	SAP HANA persistence encryption	Initially, or if root keys have been changed at the source system, root key backup(s) must be imported at the target system before recovery can be executed.
	SAP HANA system replication source	No additional steps required. If target system has no HSR configured it will stay a standalone system.
	SAP HANA multiple partitions	No additional steps required, but mount points for SAP HANA volume partitions must be available at the target system with same naming convention (only SID is different).
MDC multiple tenants or MDC single tenant with SID <> tenant name	Standard configuration	SnapCenter clone operation and optional recovery script execution. Script recovers all tenants. If tenants or tenant names does not exist at the target system names, required directories will be automatically created during the SAP HANA recovery operation. Tenant names will be same as source and need to be renamed after recovery, if required.
	SAP HANA persistence encryption	If a DBID of the source system does not exist before at the target system, the system database must be recovered first, before the root key backup of this tenant can be imported.
	HANA system replication source	No additional steps required. If target system has no HSR configured it will stay a standalone system.
	HANA multiple partitions	No additional steps required, but mount points for SAP HANA volume partitions must be available at the target system with same naming convention (only SID is different).

Within this section, the following scenarios are covered.

- SAP HANA system refresh without a clone split operation.
- Cloning from primary storage with tenant name equal to the SID
- Cloning from off-site backup storage
- Cloning from primary storage with multiple tenants
- Clone delete operation
- SAP HANA system refresh with a clone split operation
- Cloning from primary storage with tenant name equal to the SID
- Clone split operation

Prerequisites and limitations

The workflows described in the following sections have a few prerequisites and limitations regarding the SAP HANA system architecture and the SnapCenter configuration.

- The described workflows are only valid for the SnapCenter 5.0 release or higher.
- The described workflows are valid for single host SAP HANA MDC systems with single or multiple tenants. SAP HANA multiple host systems are not covered.
- The SnapCenter SAP HANA plug-in must be deployed on the target host to enable SnapCenter auto discovery and the execution of automation scripts.
- The workflows are valid for SAP HANA systems using NFS or FCP on physical hosts, or for virtual hosts using in-guest NFS mounts.

Lab setup

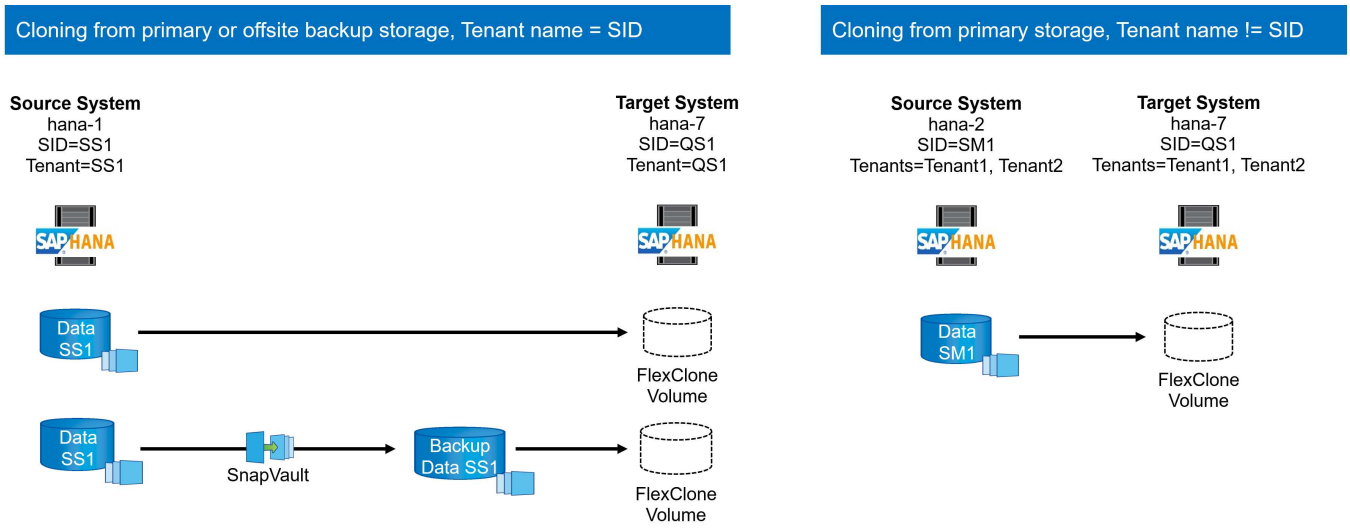
The figure below shows the lab setup that was used for the different system refresh operation options.

- Cloning from primary storage or off-site backup storage; tenant name is equal to the SID.
  - Source SAP HANA system: SS1 with Tenant SS1
  - Target SAP HANA system: QS1 with Tenant QS1
- Cloning from primary storage; multiple tenants.
  - Source SAP HANA system: SM1 with Tenant1 and Tenant2
  - Target SAP HANA system: QS1 with Tenant1 and Tenant2

The following software versions were used:

- SnapCenter 5.0
- SAP HANA systems: HANA 2.0 SPS7 rev.73
- SLES 15
- ONTAP 9.14P1

All SAP HANA systems must be configured based on the configuration guide SAP HANA on NetApp AFF systems with NFS. SnapCenter and the SAP HANA resources were configured based on the best practice guide SAP HANA Backup and Recovery with SnapCenter.



Initial one-time preparation steps

As an initial step, the target SAP HANA system must be configured within SnapCenter.

1. Installation of SAP HANA target system
2. Configuration of SAP HANA system in SnapCenter
   as described in TR-4614: SAP HANA Backup and Recovery with SnapCenter
   a. Configuration of SAP HANA database user for SnapCenter backup operations
      This user must be identical at the source and the target system.
   b. Configuration of hdbuserstore key for the <sid>adm with above backup user. If the automation script is used for recovery the key name must be <SID>KEY
   c. Deployment of SnapCenter SAP HANA plug-in at target host. SAP HANA system is auto discovered by SnapCenter.
   d. Configuration of SAP HANA resource protection (optional)

The first SAP system refresh operation after the initial installation is prepared with the following steps:

3. Shutdown target SAP HANA system
4. Unmount SAP HANA data volume.

You must add the scripts that should be executed at the target system to the SnapCenter allowed commands config file.

```
hana-7:/opt/NetApp/snapcenter/scc/etc # cat
/opt/NetApp/snapcenter/scc/etc/allowed_commands.config
command: mount
command: umount
command: /mnt/sapcc-share/SAP-System-Refresh/sc-system-refresh.sh
hana-7:/opt/NetApp/snapcenter/scc/etc #
```

Cloning from primary storage with tenant name equal to SID

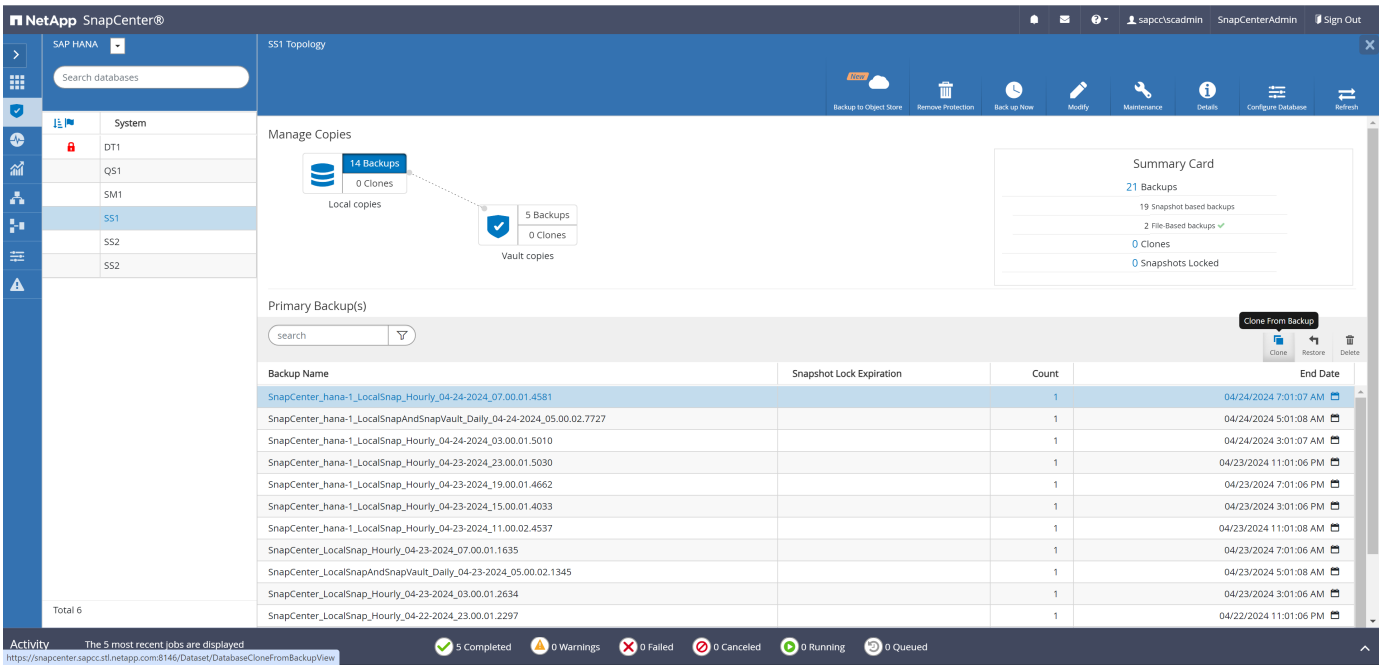
This section describes the SAP HANA system refresh workflow where the tenant name at the source and the target system is identical to the SID. The storage cloning is executed at the primary storage and the recovery is automated with the script `sc-system-refresh.sh`.

The workflow consists of the following steps:

1. If SAP HANA persistence encryption is enabled at the source system, the encryption root keys must be imported once. An import is also required if the keys have been changed at the source system. See chapter "Considerations for SAP HANA system refresh operations using storage snapshot backups"
2. If the target SAP HANA system has been protected in SnapCenter, the protection must be removed first.
3. SnapCenter clone create workflow.
   a. Select Snapshot backup from the source SAP HANA system SS1.
   b. Select target host and provide storage network interface of target host.
   c. Provide SID of the target system, in our example QS1
   d. Optionally, provide script for recovery as a post-clone operation.
4. SnapCenter cloning operation.
   a. Creates FlexClone volume based on selected Snapshot backup of source SAP HANA system.
   b. Exports FlexClone volume to target host storage network interface or igroup.
   c. Executes mount operation of Mounts FlexClone volume at target host.
   d. Executes post-clone operation recovery script, if configured before. Otherwise, recovery needs to be done manually when SnapCenter workflow is finished.
      - Recovery of system database.
      - Recovery of tenant database with tenant name = QS1.
5. Optionally, protect the target SAP HANA resource in SnapCenter.

The following screenshots show the required steps.

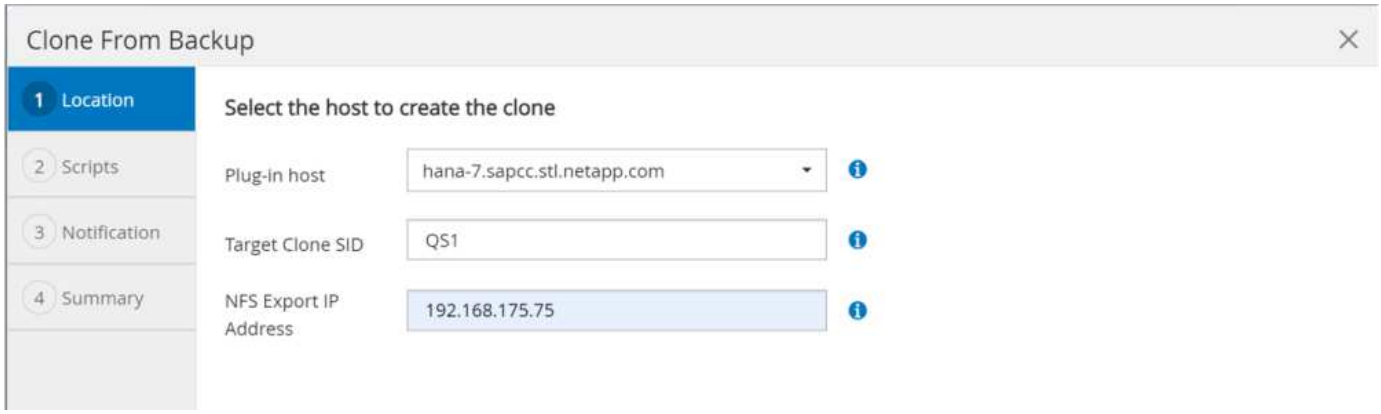
1. Select a Snapshot backup from the source system SS1 and click Clone.



2. Select the host where the target system QS1 is installed. Enter QS1 as the target SID. The NFS export IP address must be the storage network interface of the target host.

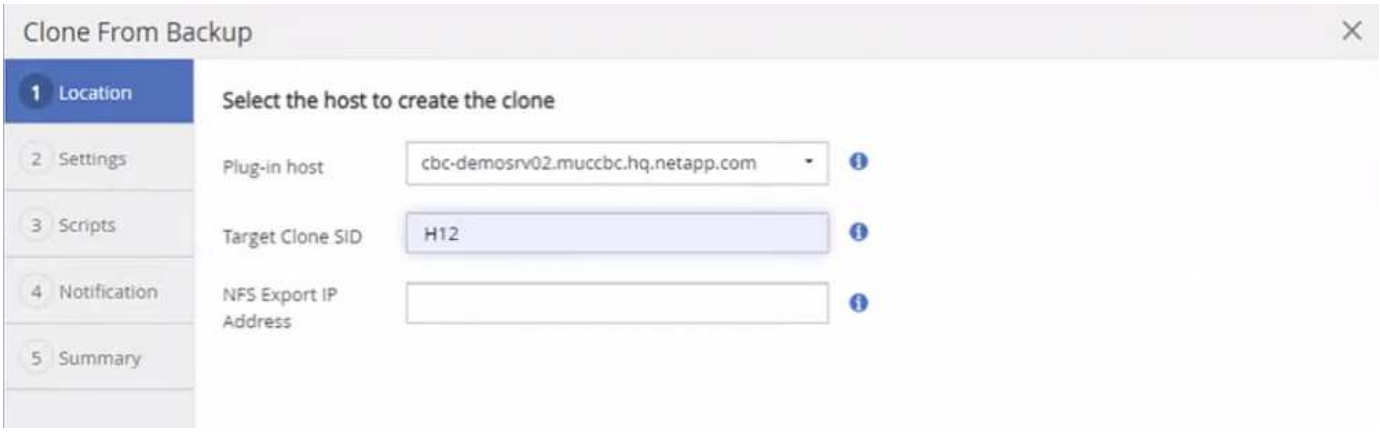
> The target SID which is entered controls how SnapCenter manages the cloned resource. If a resource with the target SID is already configured in SnapCenter and matches the plug-in host, SnapCenter just assigns the clone to this resource. If the SID is not configured on the target host, SnapCenter creates a new resource.

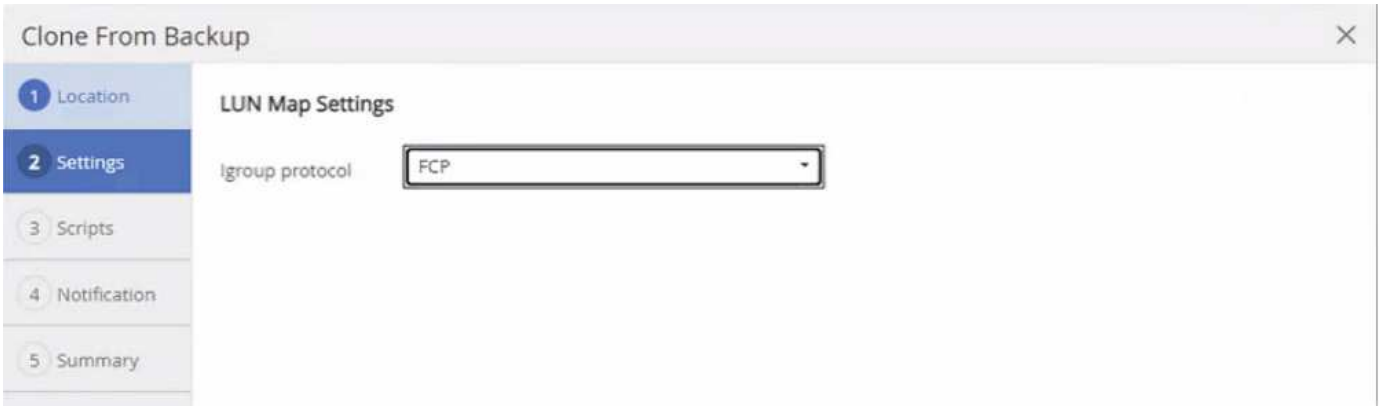
> It is crucial that the target system resource and host has been configured in SnapCenter before you start the cloning workflow. Otherwise, the new resource created by SnapCenter will not support auto discovery and the described workflows won't work.



In a Fibre Channel SAN setup, no export IP address is required, but you need to provide the used protocol in the next screen.

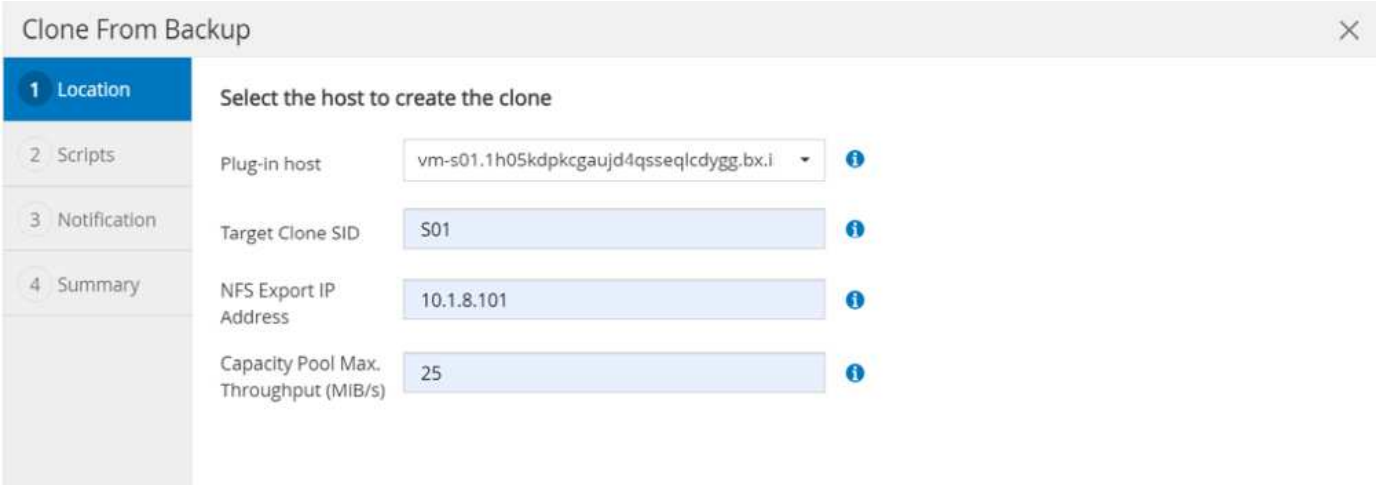
> The screenshots show a different lab setup using a FibreChannel connectivity.



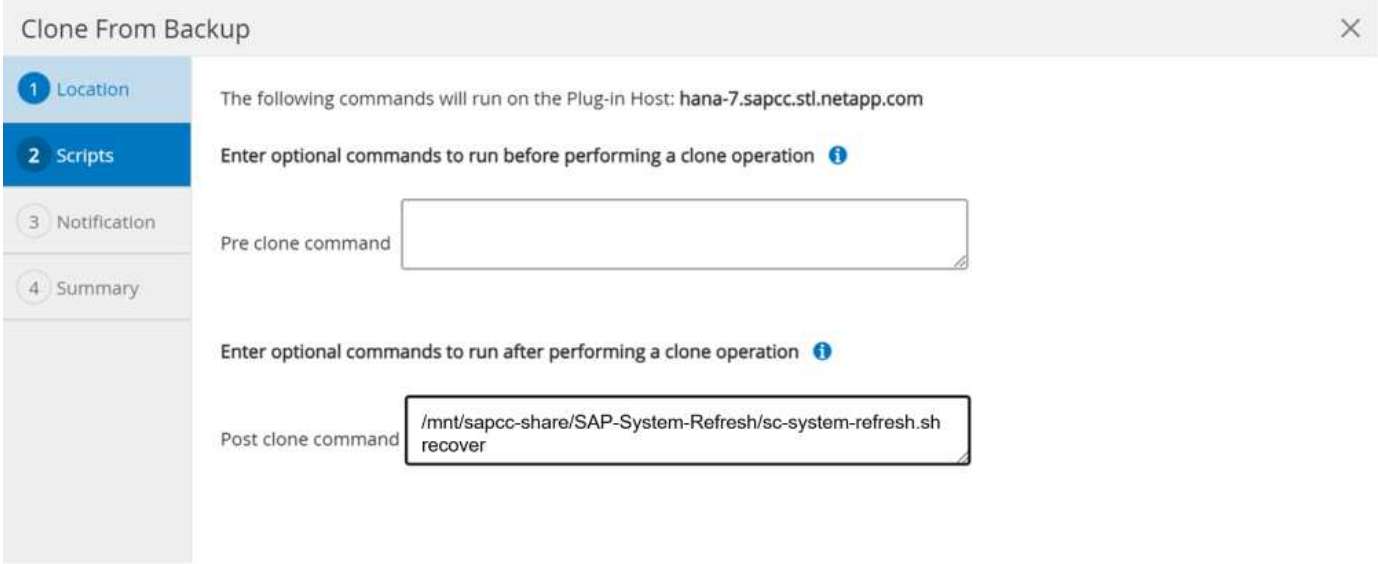


With Azure NetApp Files and a manual QoS capacity pool, you need to provide the maximum throughput for the new volume. Make sure that the capacity pool has enough headroom, otherwise the cloning workflow will fail.

The screenshots show a different lab setup running in Microsoft Azure with Azure NetApp Files.



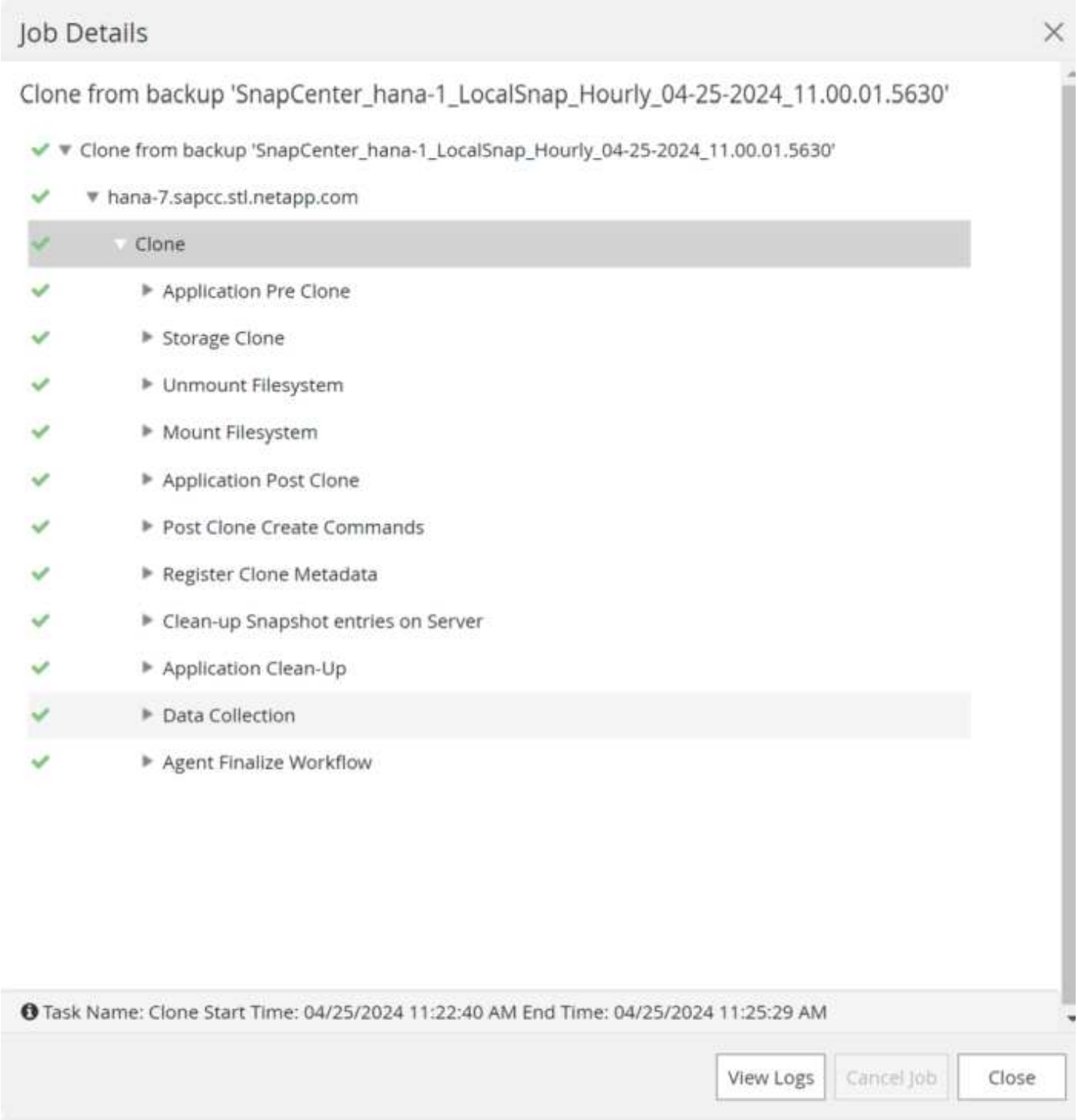
3. Enter the optional post-clone scripts with the required command-line options. With our example we use a post clone script to execute the SAP HANA database recovery.



As discussed before, the usage of the recovery script is optional. The recovery can also be done manually after the SnapCenter cloning workflow is finished.

The script for the recovery operation recovers the SAP HANA database to the point in time of the Snapshot using the clear logs operation and does not execute any forward recovery. If a forward recovery to a specific point in time is required, the recovery must be performed manually. A manual forward recovery also requires that the log backups from the source system are available at the target host.

4. The Job Details screen in SnapCenter shows the progress of the operation. The job details also show that the overall runtime including database recovery has been less than 3 minutes.

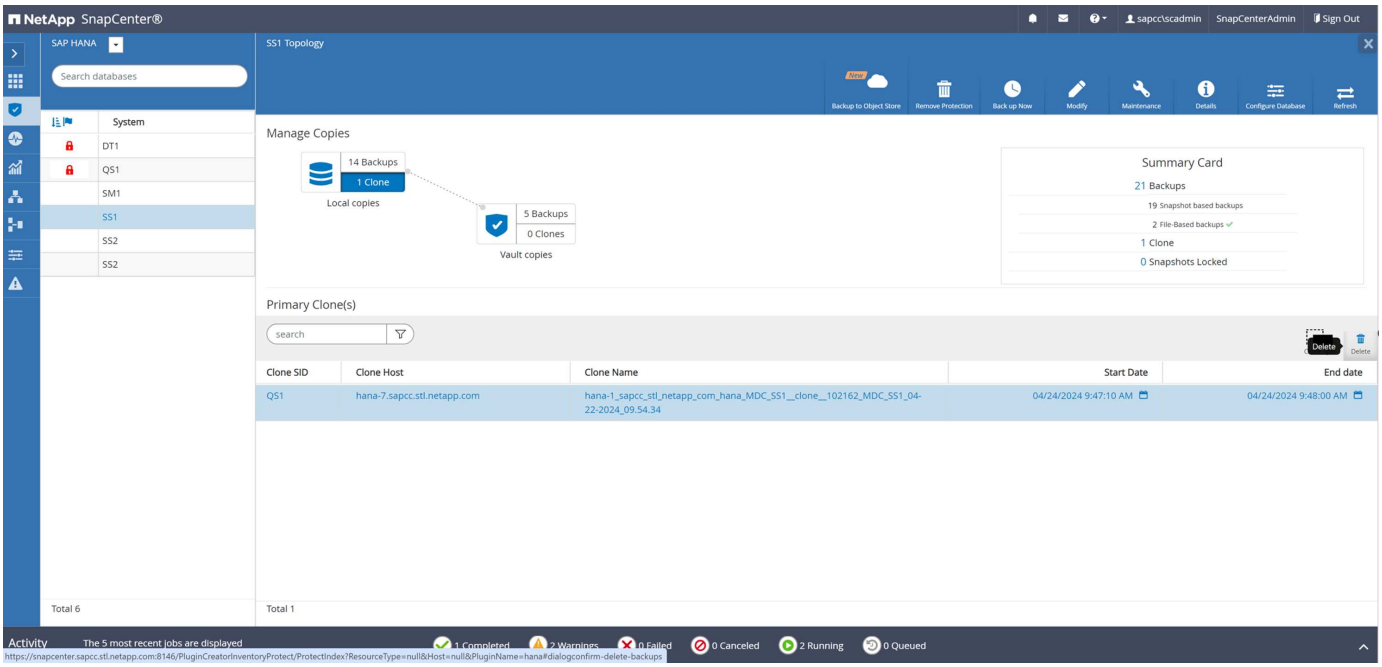


5. The logfile of the `sc-system-refresh` script shows the different steps that were executed for the recovery operation. The script reads the list of tenants from the system database and executes a recovery of all existing tenants.

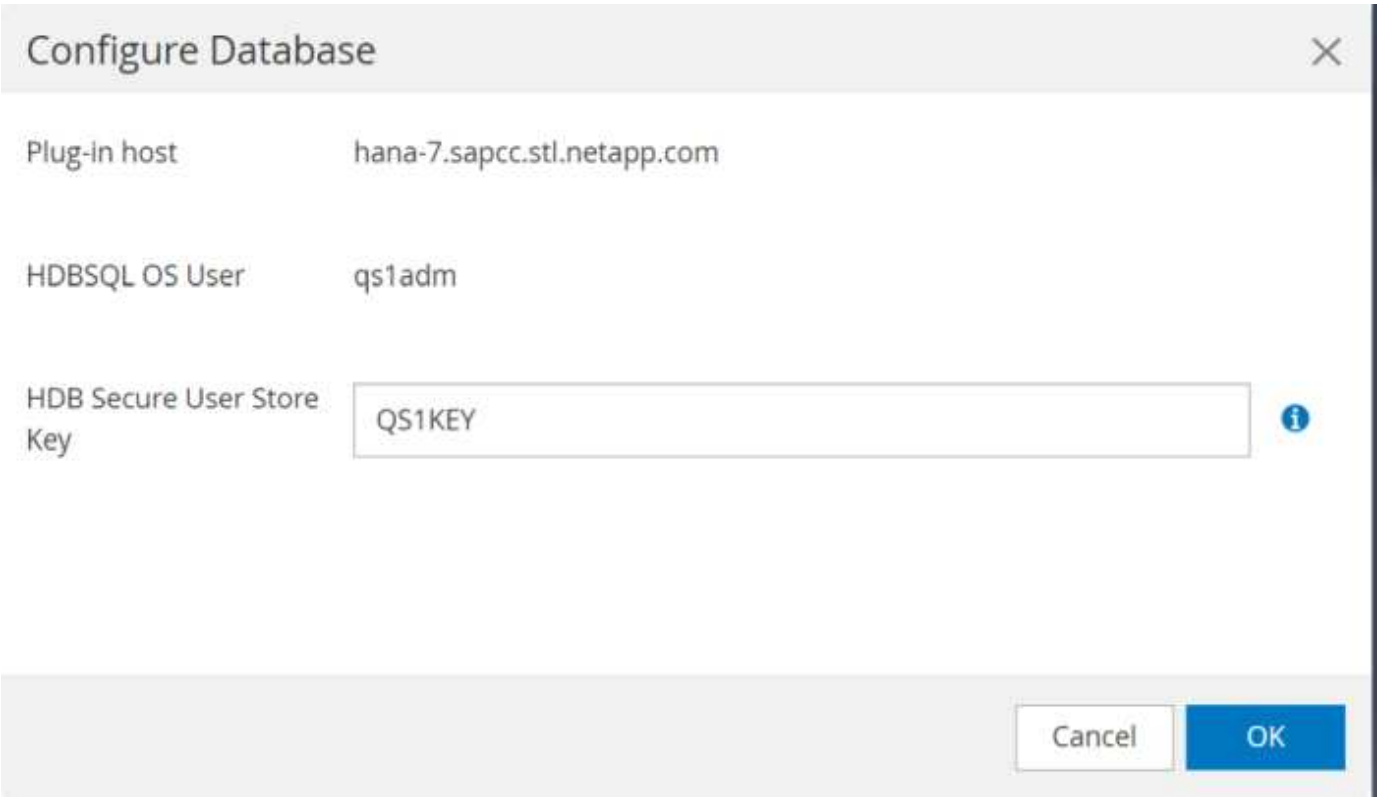
```
20240425112328###hana-7###sc-system-refresh.sh: Script version: 3.0
hana-7:/mnt/sapcc-share/SAP-System-Refresh # cat sap-system-refresh-
QS1.log
20240425112328###hana-7###sc-system-refresh.sh: ******************
Starting script: recovery operation *************************
20240425112328###hana-7###sc-system-refresh.sh: Recover system database.
```

```
20240425112328###hana-7###sc-system-refresh.sh:
/usr/sap/QS1/HDB11/exe/Python/bin/python
/usr/sap/QS1/HDB11/exe/python_support/recoverSys.py --command "RECOVER
DATA USING SNAPSHOT CLEAR LOG"
20240425112346###hana-7###sc-system-refresh.sh: Wait until SAP HANA
database is started ....
20240425112347###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425112357###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425112407###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425112417###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425112428###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425112438###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425112448###hana-7###sc-system-refresh.sh: Status: GREEN
20240425112448###hana-7###sc-system-refresh.sh: HANA system database
started.
20240425112448###hana-7###sc-system-refresh.sh: Checking connection to
system database.
20240425112448###hana-7###sc-system-refresh.sh:
/usr/sap/QS1/SYS/exe/hdb/hdbsql -U QS1KEY 'select * from sys.m_databases;'
DATABASE_NAME,DESCRIPTION,ACTIVE_STATUS,ACTIVE_STATUS_DETAILS,OS_USER,OS_G
ROUP,RESTART_MODE,FALLBACK_SNAPSHOT_CREATE_TIME
"SYSTEMDB","SystemDB-QS1-11","YES","","","","DEFAULT",?
"QS1","QS1-11","NO","ACTIVE","","","DEFAULT",?
2 rows selected (overall time 16.225 msec; server time 860 usec)
20240425112448###hana-7###sc-system-refresh.sh: Succesfully connected to
system database.
20240425112449###hana-7###sc-system-refresh.sh: Tenant databases to
recover: QS1
20240425112449###hana-7###sc-system-refresh.sh: Found inactive
tenants(QS1) and starting recovery
20240425112449###hana-7###sc-system-refresh.sh: Recover tenant database
QS1.
20240425112449###hana-7###sc-system-refresh.sh:
/usr/sap/QS1/SYS/exe/hdb/hdbsql -U QS1KEY RECOVER DATA FOR QS1 USING
SNAPSHOT CLEAR LOG
0 rows affected (overall time 22.138599 sec; server time 22.136268 sec)
20240425112511###hana-7###sc-system-refresh.sh: Checking availability of
Indexserver for tenant QS1.
20240425112511###hana-7###sc-system-refresh.sh: Recovery of tenant
database QS1 succesfully finished.
20240425112511###hana-7###sc-system-refresh.sh: Status: GREEN
20240425112511###hana-7###sc-system-refresh.sh: ********************
Finished script: recovery operation **************************
hana-7:/mnt/sapcc-share/SAP-System-Refresh
```

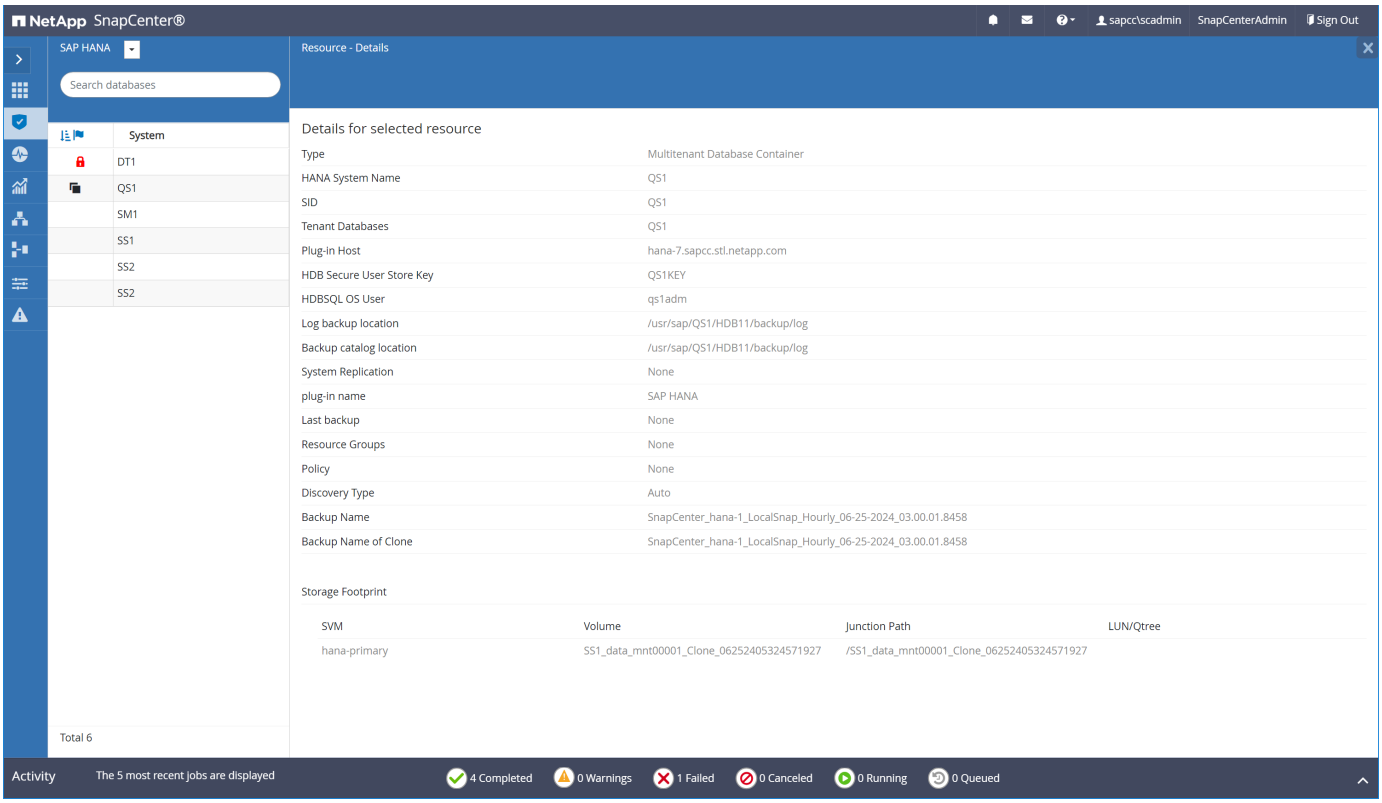
6. When the SnapCenter job is finished, the clone is visible within the topology view of the source system.



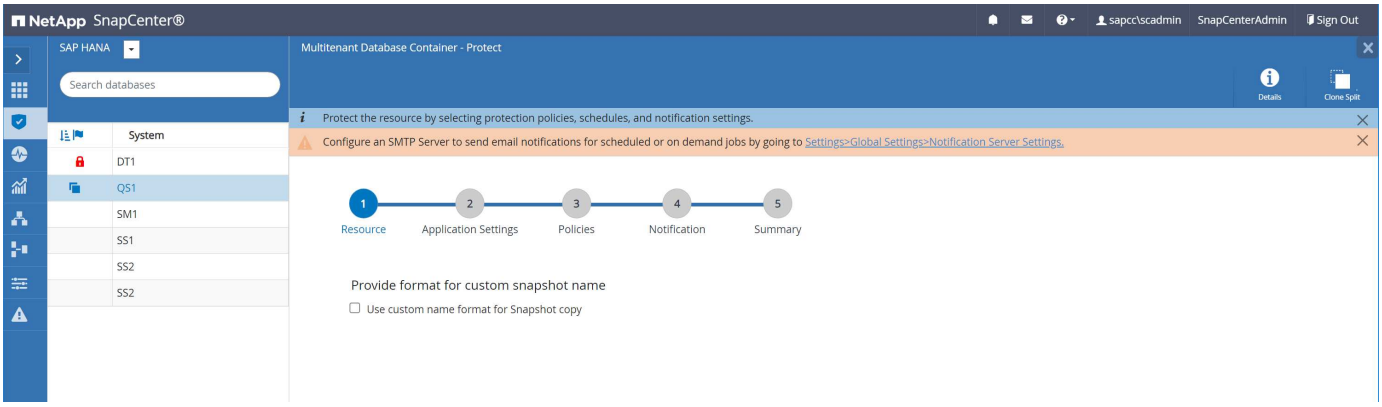
7. The SAP HANA database is now running.
8. If you want to protect the target SAP HANA system, you need to run the auto discovery by clicking on the target system resource.



When the auto discovery process is finished, the new cloned volume is listed in the storage footprint section.

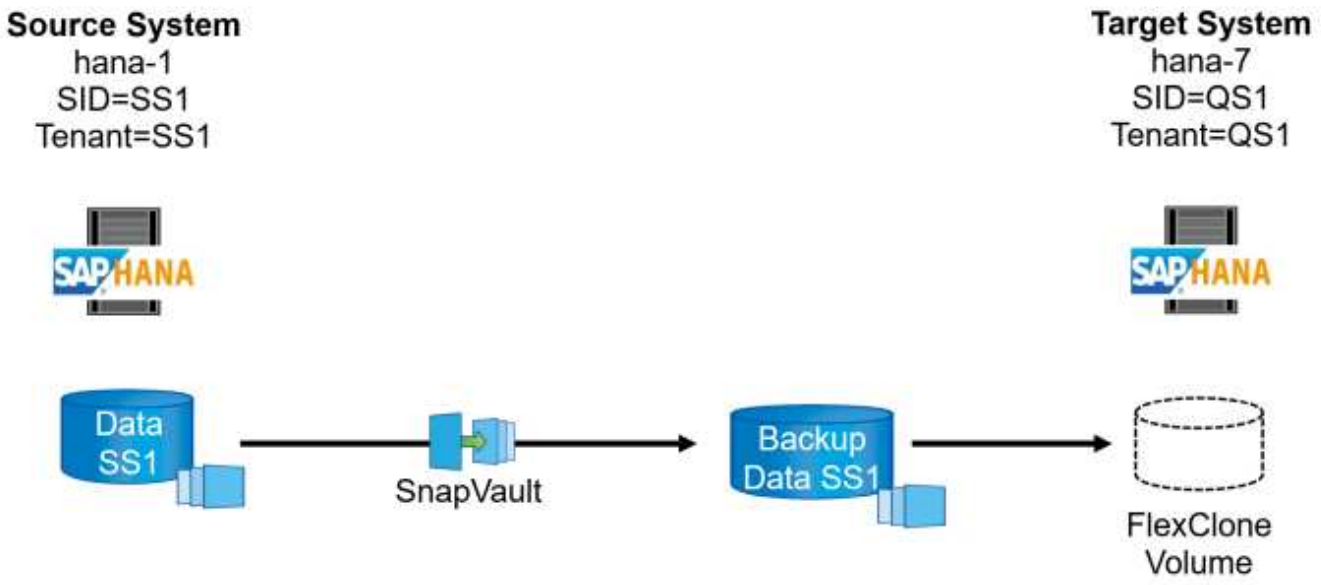


By clicking on the resource again, data protection can be configured for the refreshed QS1 system.

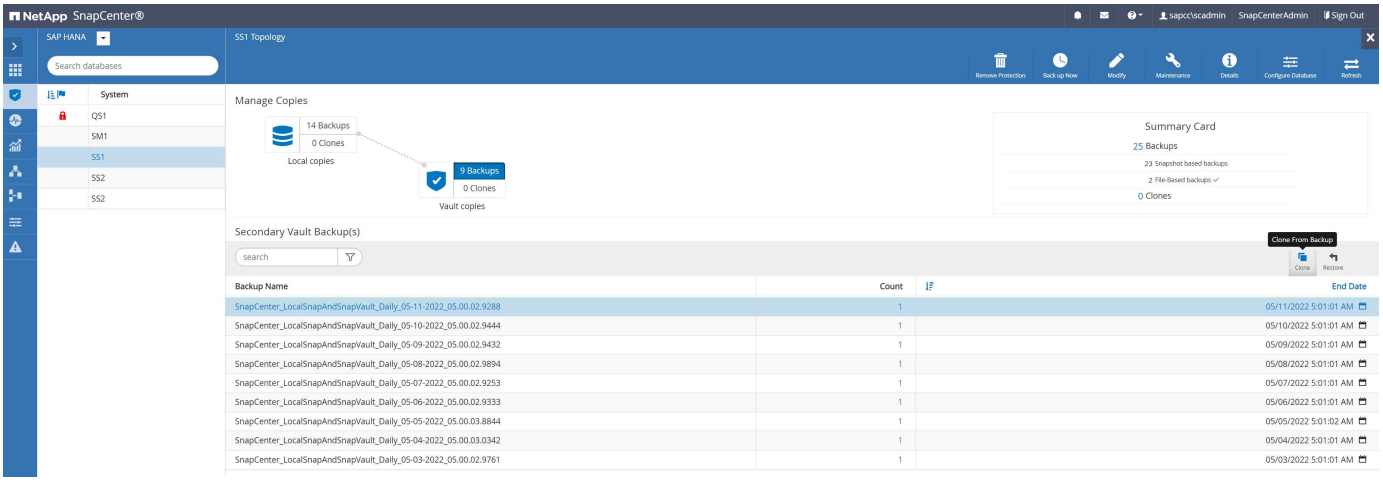


Cloning from off-site backup storage

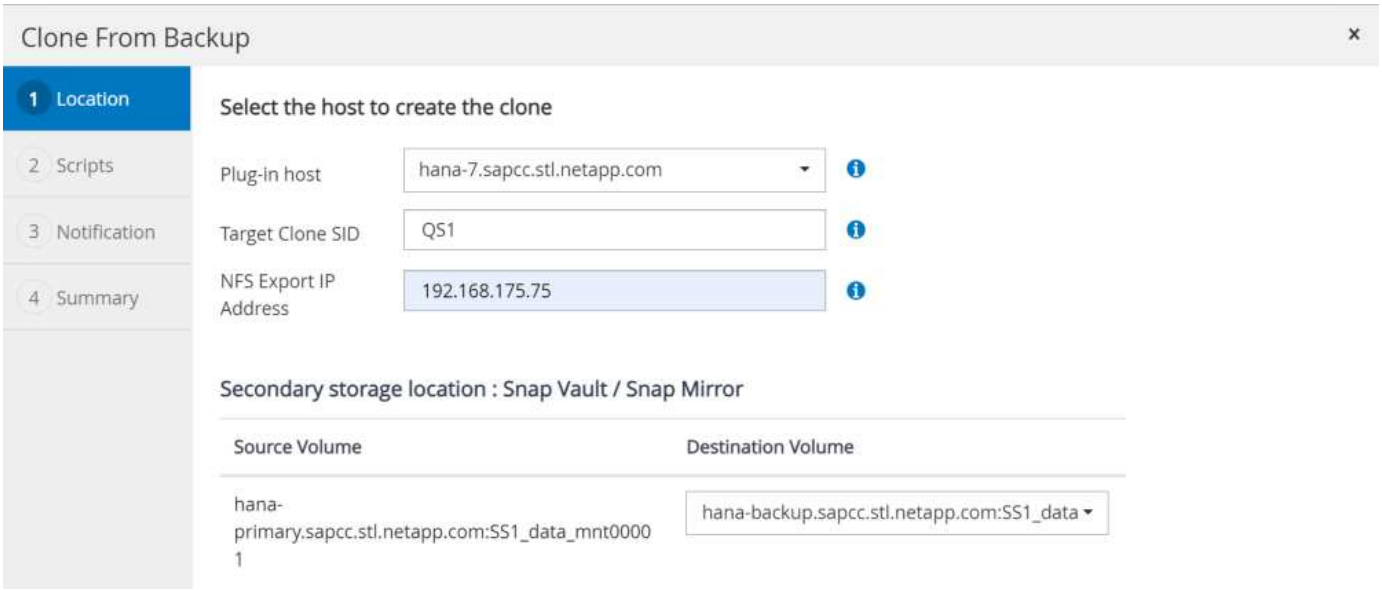
This section describes the SAP HANA system refresh workflow for which the tenant name at the source and the target system is identical to the SID. Storage cloning is executed at the off-site backup storage and further automated using the script sc-system-refresh.sh.



The only difference in the SAP HANA system refresh workflow between primary and off-site backup storage cloning is the selection of the Snapshot backup in SnapCenter. For off-site backup storage cloning, the secondary backups must be selected first, followed by the selection of the Snapshot backup.



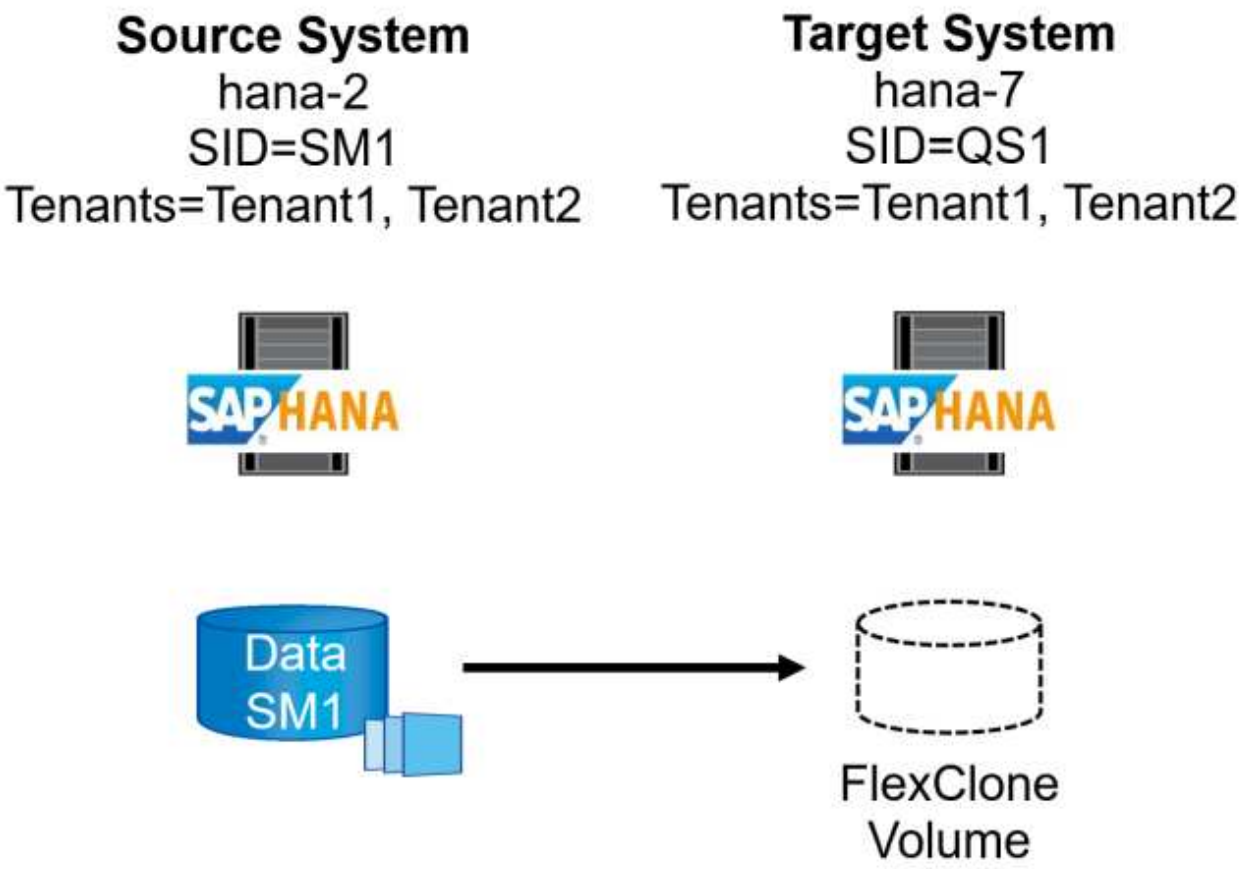
If there are multiple secondary storage locations for the selected backup, you need to choose the required destination volume.



All subsequent steps are identical to the workflow for cloning from primary storage.

Cloning a SAP HANA system with multiple tenants

This section describes the SAP HANA system refresh workflow with multiple tenants. Storage cloning is executed at the primary storage and further automated using the script `sc-system-refresh.sh`.



The required steps in SnapCenter are identical to what has been described in the section “Cloning from

primary storage with tenant name equal to SID.” The only difference is in the tenant recovery operation within the script `sc-system-refresh.sh`, where all tenants are recovered.

```
20240430070214###hana-7###sc-system-refresh.sh:
**********************************************************************
********
20240430070214###hana-7###sc-system-refresh.sh: Script version: 3.0
20240430070214###hana-7###sc-system-refresh.sh: *******************
Starting script: recovery operation **************************
20240430070214###hana-7###sc-system-refresh.sh: Recover system database.
20240430070214###hana-7###sc-system-refresh.sh:
/usr/sap/QS1/HDB11/exe/Python/bin/python
/usr/sap/QS1/HDB11/exe/python_support/recoverSys.py --command "RECOVER
DATA USING SNAPSHOT CLEAR LOG"
[140310725887808, 0.008] >> starting recoverSys (at Tue Apr 30 07:02:15
2024)
[140310725887808, 0.008] args: ()
[140310725887808, 0.008] keys: \{'command': 'RECOVER DATA USING SNAPSHOT
CLEAR LOG'}
using logfile /usr/sap/QS1/HDB11/hana-7/trace/backup.log
recoverSys started: ============2024-04-30 07:02:15 ============
testing master: hana-7
hana-7 is master
shutdown database, timeout is 120
stop system
stop system on: hana-7
stopping system: 2024-04-30 07:02:15
stopped system: 2024-04-30 07:02:15
creating file recoverInstance.sql
restart database
restart master nameserver: 2024-04-30 07:02:20
start system: hana-7
sapcontrol parameter: ['-function', 'Start']
sapcontrol returned successfully:
2024-04-30T07:02:32-04:00 P0023828 18f2eab9331 INFO RECOVERY RECOVER DATA
finished successfully
recoverSys finished successfully: 2024-04-30 07:02:33
[140310725887808, 17.548] 0
[140310725887808, 17.548] << ending recoverSys, rc = 0 (RC_TEST_OK), after
17.540 secs
20240430070233###hana-7###sc-system-refresh.sh: Wait until SAP HANA
database is started ....
20240430070233###hana-7###sc-system-refresh.sh: Status: GRAY
20240430070243###hana-7###sc-system-refresh.sh: Status: GRAY
20240430070253###hana-7###sc-system-refresh.sh: Status: GRAY
20240430070304###hana-7###sc-system-refresh.sh: Status: GRAY
```

```
20240430070314###hana-7###sc-system-refresh.sh: Status: GREEN
20240430070314###hana-7###sc-system-refresh.sh: HANA system database
started.
20240430070314###hana-7###sc-system-refresh.sh: Checking connection to
system database.
20240430070314###hana-7###sc-system-refresh.sh:
/usr/sap/QS1/SYS/exe/hdb/hdbsql -U QS1KEY 'select * from sys.m_databases;'
20240430070314###hana-7###sc-system-refresh.sh: Succesfully connected to
system database.
20240430070314###hana-7###sc-system-refresh.sh: Tenant databases to
recover: TENANT2
TENANT1
20240430070314###hana-7###sc-system-refresh.sh: Found inactive
tenants(TENANT2
TENANT1) and starting recovery
20240430070314###hana-7###sc-system-refresh.sh: Recover tenant database
TENANT2.
20240430070314###hana-7###sc-system-refresh.sh:
/usr/sap/QS1/SYS/exe/hdb/hdbsql -U QS1KEY RECOVER DATA FOR TENANT2 USING
SNAPSHOT CLEAR LOG
20240430070335###hana-7###sc-system-refresh.sh: Checking availability of
Indexserver for tenant TENANT2.
20240430070335###hana-7###sc-system-refresh.sh: Recovery of tenant
database TENANT2 succesfully finished.
20240430070335###hana-7###sc-system-refresh.sh: Status: GREEN
20240430070335###hana-7###sc-system-refresh.sh: Recover tenant database
TENANT1.
20240430070335###hana-7###sc-system-refresh.sh:
/usr/sap/QS1/SYS/exe/hdb/hdbsql -U QS1KEY RECOVER DATA FOR TENANT1 USING
SNAPSHOT CLEAR LOG
20240430070349###hana-7###sc-system-refresh.sh: Checking availability of
Indexserver for tenant TENANT1.
20240430070350###hana-7###sc-system-refresh.sh: Recovery of tenant
database TENANT1 succesfully finished.
20240430070350###hana-7###sc-system-refresh.sh: Status: GREEN
20240430070350###hana-7###sc-system-refresh.sh: *******************
Finished script: recovery operation **************************
```

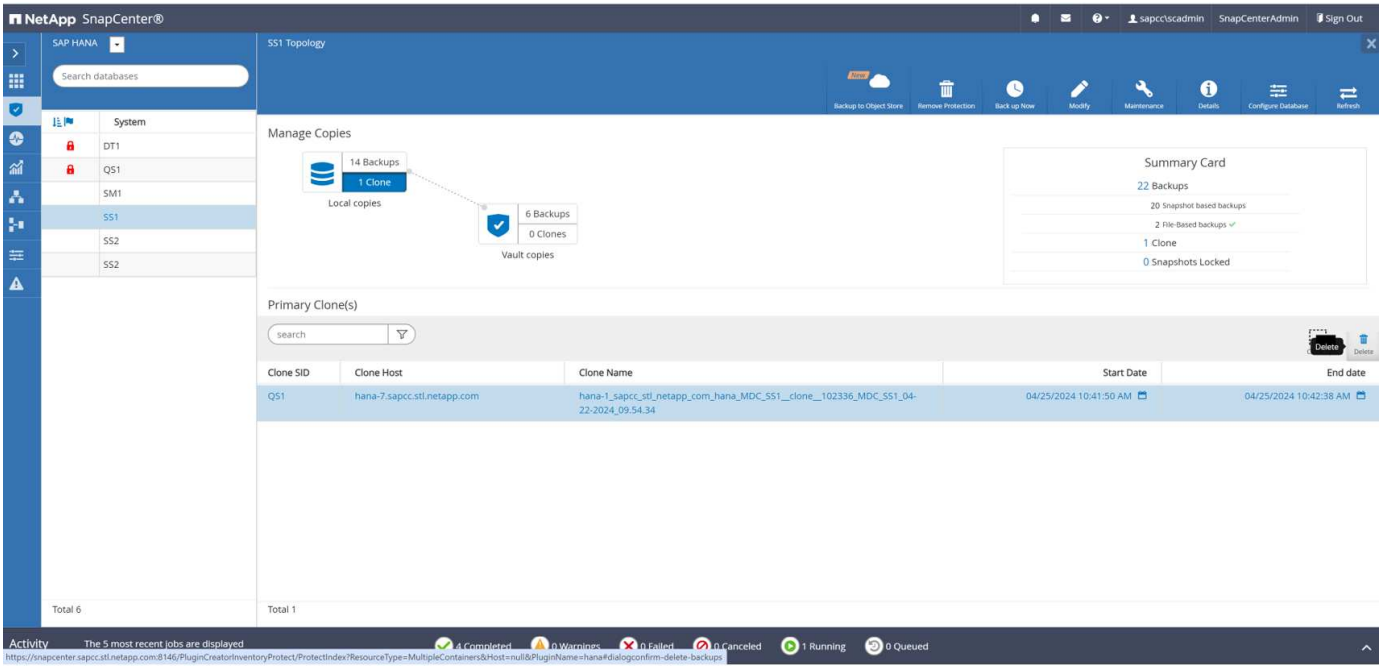
Clone delete operation

A new SAP HANA system refresh operation is started by cleaning up the target system using the SnapCenter clone delete operation.

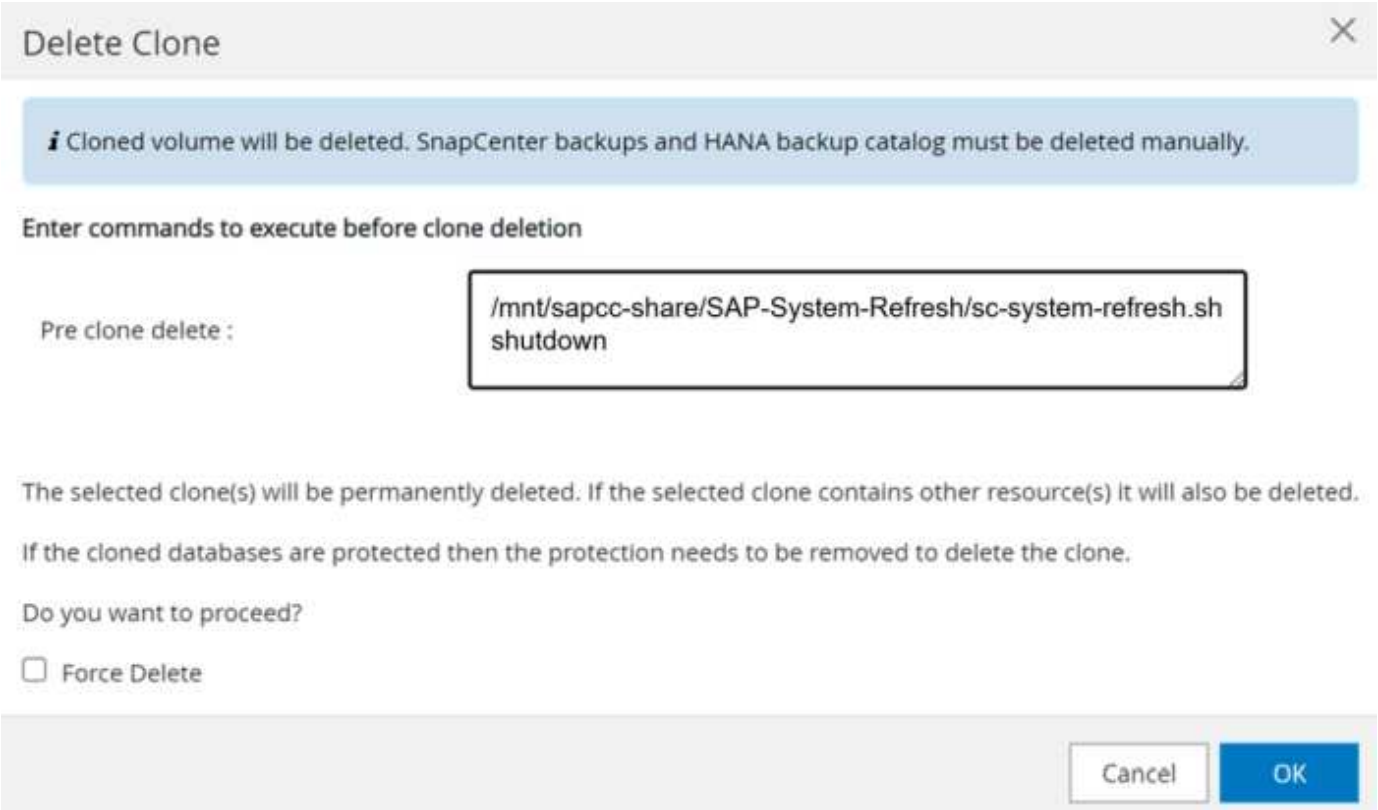
If the target SAP HANA system has been protected in SnapCenter, the protection must be removed first. Within the topology view of the target system, click Remove Protection.

The clone delete workflow is now executed with the following steps.

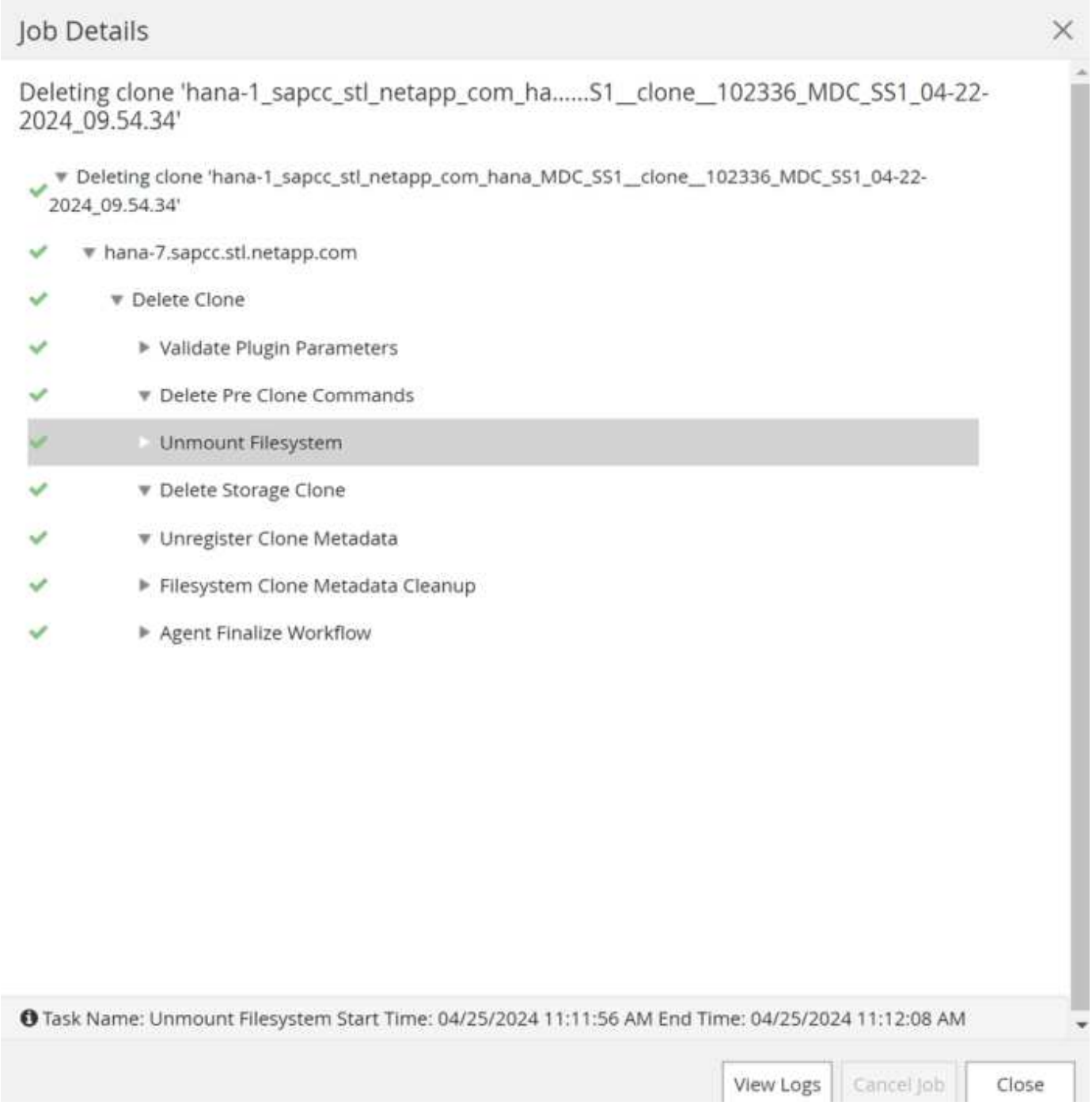
1. Select the clone within the topology view of the source system and click Delete.



2. Enter the pre-clone and unmount scripts with the required command line options.



3. The job details screen in SnapCenter shows the progress of the operation.



4. The log file of the `sc-system-refresh` script shows the shutdown and unmount operation steps.

```
20240425111042###hana-7###sc-system-refresh.sh:
**********************************************************************
********
20240425111042###hana-7###sc-system-refresh.sh: Script version: 3.0
20240425111042###hana-7###sc-system-refresh.sh: ******************
Starting script: shutdown operation *************************
20240425111042###hana-7###sc-system-refresh.sh: Stopping HANA database.
20240425111042###hana-7###sc-system-refresh.sh: sapcontrol -nr 11
-function StopSystem HDB
25.04.2024 11:10:42
StopSystem
OK
20240425111042###hana-7###sc-system-refresh.sh: Wait until SAP HANA
database is stopped ....
20240425111042###hana-7###sc-system-refresh.sh: Status: GREEN
20240425111052###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425111103###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425111113###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425111123###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425111133###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425111144###hana-7###sc-system-refresh.sh: Status: YELLOW
20240425111154###hana-7###sc-system-refresh.sh: Status: GRAY
20240425111154###hana-7###sc-system-refresh.sh: SAP HANA database is
stopped.
20240425111154###hana-7###sc-system-refresh.sh: ******************
Finished script: shutdown operation *************************
```

5. The SAP HANA refresh operation can now be started again using the SnapCenter clone create operation.

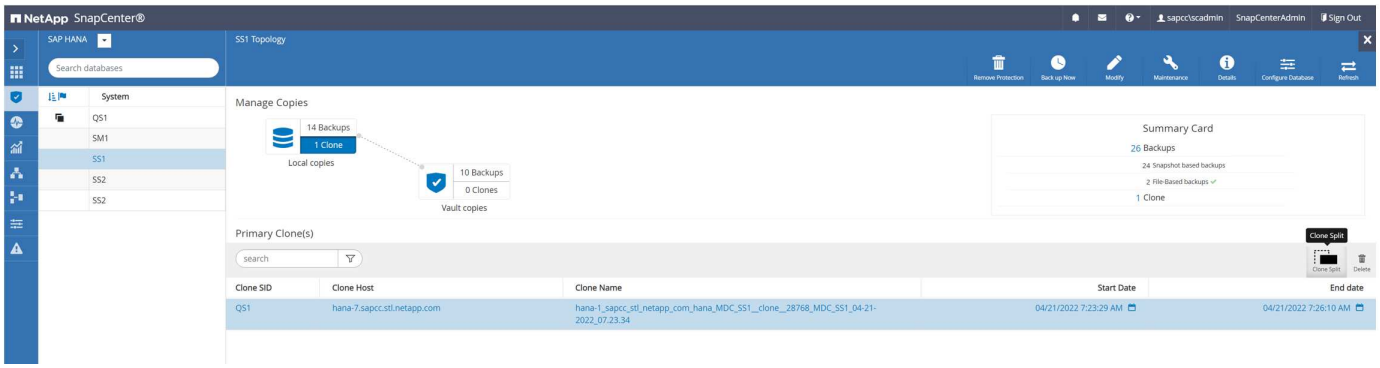
SAP HANA system refresh with clone split operation

If the target system of the system refresh operation is planned to be used for a longer timeframe, it makes sense to split the FlexClone volume as part of the system refresh operation.

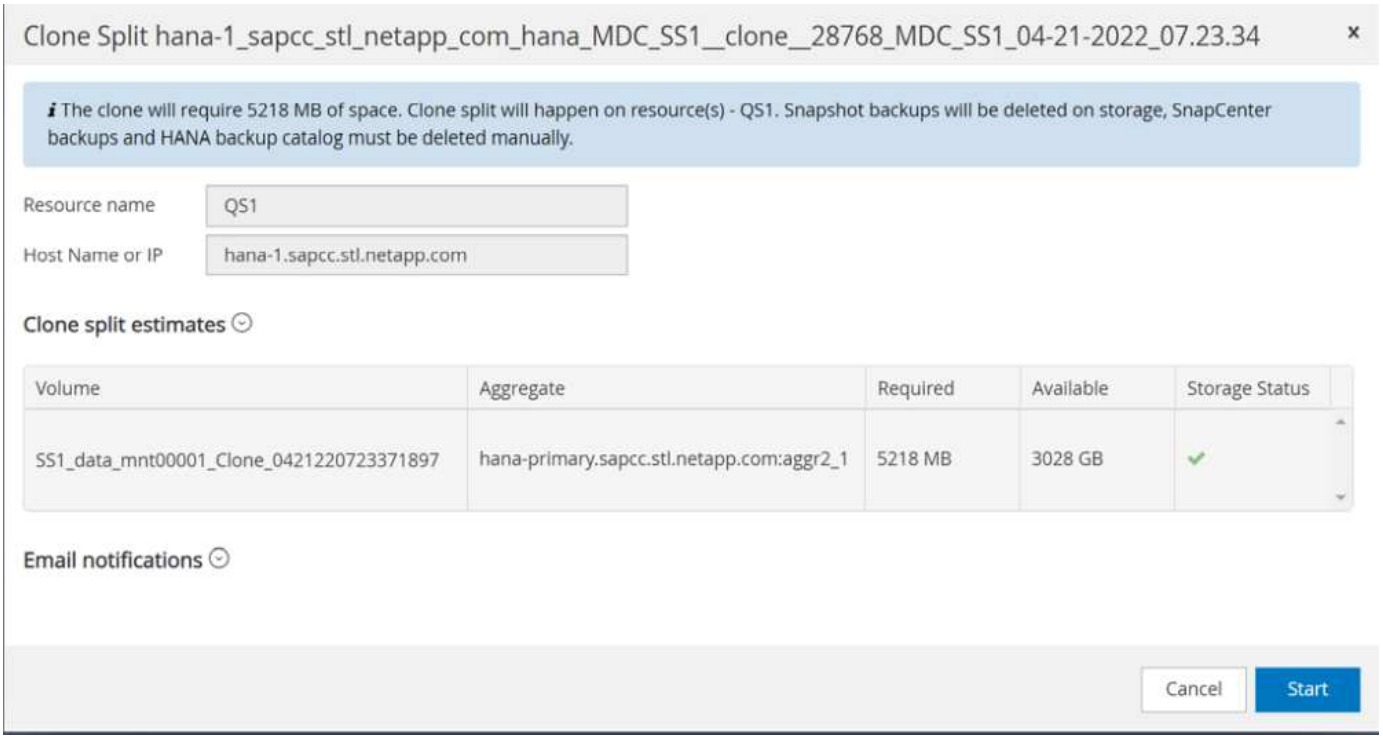
> The clone split operation does not block the use of the cloned volume and can therefore be executed at any time while the SAP HANA database is in use.

> With Azure NetApp Files, the clone split operation is not available, since Azure NetApp Files always splits the clone after creation.

The clone split workflow in SnapCenter is initiated in the topology view of the source system by selecting the clone and clicking on clone split.

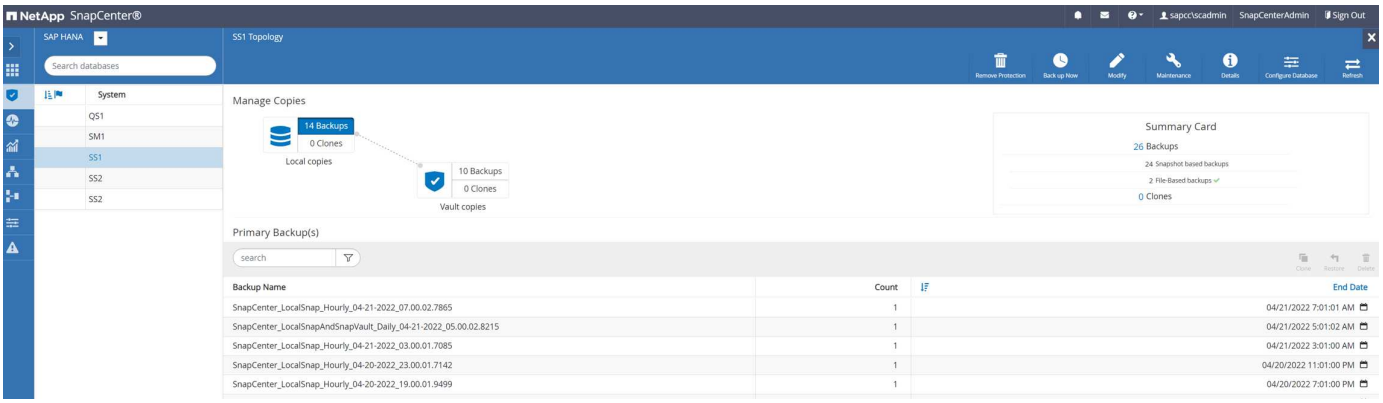


A preview is shown in the next screen, which provides information on the required capacity for the split volume.



The SnapCenter job log shows the progress of the clone split operation.

In the resource view in SnapCenter the target system QS1 is now not marked as a cloned resource anymore. When going back to the topology view of the source system, the clone is not visible anymore. The split volume is now independent from the Snapshot backup of the source system.



The refresh workflow after a clone split operation looks slightly different than the operation without clone split. After a clone split operation, there is no clone delete operation required, because the target data volume is not a FlexClone volume anymore.

The workflow consists of the following steps:

1. If the target SAP HANA system has been protected in SnapCenter, the protection must be removed first.
2. The SAP HANA database must be shut down, the data volume must be unmounted and the fstab entry created by SnapCenter must be removed. These steps need to be executed manually.
3. Now the SnapCenter clone create workflow can be executed as described in sections before.
4. After the refresh operation, the old target data volume still exists and it must be deleted manually with, for example, ONTAP System Manager.

SnapCenter workflow automation with PowerShell scripts

In the previous sections, the different workflows were executed using the SnapCenter UI. All the workflows can also be executed with PowerShell scripts or REST API calls, allowing further automation. The following sections describe basic PowerShell script examples for the following workflows.

- Create clone
- Delete clone

> The example scripts are provided as is and are not supported by NetApp.

All scripts must be executed in a PowerShell command window. Before the scripts can be run, a connection to the SnapCenter server must be established using the `Open-SmConnection` command.

Create clone

The simple script below demonstrates how a SnapCenter clone create operation can be executed using

PowerShell commands. The SnapCenter `New-SmClone` command is executed with the required command line option for the lab environment and the automation script discussed before.

```
$BackupName='SnapCenter_hana-1_LocalSnap_Hourly_06-25-2024_03.00.01.8458'
$JobInfo=New-SmClone -AppPluginCode hana -BackupName $BackupName
-Resources @\{"Host"="hana-1.sapcc.stl.netapp.com";"UID"="MDC\SS1"}
-CloneToInstance hana-7.sapcc.stl.netapp.com -postclonecreatecommands
'/mnt/sapcc-share/SAP-System-Refresh/sc-system-refresh.sh recover'
-NFSExportIPs 192.168.175.75 -CloneUid 'MDC\QS1'
# Get JobID of clone create job
$Job=Get-SmJobSummaryReport | ?\{$_.JobType -eq "Clone" } | ?\{$_.JobName
-Match $BackupName} | ?\{$_.Status -eq "Running"}
$JobId=$Job.SmJobId
Get-SmJobSummaryReport -JobId $JobId
# Wait until job is finished
do \{ $Job=Get-SmJobSummaryReport -JobId $JobId; write-host $Job.Status;
sleep 20 } while ( $Job.Status -Match "Running" )
Write-Host " "
Get-SmJobSummaryReport -JobId $JobId
Write-Host "Clone create job has been finshed."
```

The screen output shows the execution of the clone create PowerShell script.

```
PS C:\Windows\system32> C:\NetApp\clone-create.ps1
SmJobId : 110382
JobCreatedDateTime :
JobStartDateTime : 6/26/2024 9:55:34 AM
JobEndDateTime :
JobDuration :
JobName : Clone from backup 'SnapCenter_hana-1_LocalSnap_Hourly_06-25-
2024_03.00.01.8458'
JobDescription :
Status : Running
IsScheduled : False
JobError :
JobType : Clone
PolicyName :
JobResultData :
Running
Running
Running
Running
Running
Running
Running
Running
Running
Completed
SmJobId : 110382
JobCreatedDateTime :
JobStartDateTime : 6/26/2024 9:55:34 AM
JobEndDateTime : 6/26/2024 9:58:50 AM
JobDuration : 00:03:16.6889170
JobName : Clone from backup 'SnapCenter_hana-1_LocalSnap_Hourly_06-25-
2024_03.00.01.8458'
JobDescription :
Status : Completed
IsScheduled : False
JobError :
JobType : Clone
PolicyName :
JobResultData :
Clone create job has been finshed.
```

Delete clone

The simple script below demonstrates how a SnapCenter clone delete operation can be executed using

PowerShell commands. The SnapCenter `Remove-SmClone` command is executed with the required command line option for the lab environment and the automation script discussed before.

```
$CloneInfo=Get-SmClone |?\{$_.CloneName -Match "hana-
1_sapcc_stl_netapp_com_hana_MDC_SS1" }
$JobInfo=Remove-SmClone -CloneName $CloneInfo.CloneName -PluginCode hana
-PreCloneDeleteCommands '/mnt/sapcc-share/SAP-System-Refresh/sc-system-
refresh.sh shutdown QS1' -UnmountCommands '/mnt/sapcc-share/SAP-System-
Refresh/sc-system-refresh.sh umount QS1' -Confirm: $False
Get-SmJobSummaryReport -JobId $JobInfo.Id
# Wait until job is finished
do \{ $Job=Get-SmJobSummaryReport -JobId $JobInfo.Id; write-host
$Job.Status; sleep 20 } while ( $Job.Status -Match "Running" )
Write-Host " "
Get-SmJobSummaryReport -JobId $JobInfo.Id
Write-Host "Clone delete job has been finshed."
PS C:\NetApp>
```

The screen output shows the execution of the clone –delete.ps1 PowerShell script.

```
PS C:\Windows\system32> C:\NetApp\clone-delete.ps1
SmJobId : 110386
JobCreatedDateTime :
JobStartDateTime : 6/26/2024 10:01:33 AM
JobEndDateTime :
JobDuration :
JobName : Deleting clone 'hana-
1_sapcc_stl_netapp_com_hana_MDC_SS1__clone__110382_MDC_SS1_04-22-
2024_09.54.34'
JobDescription :
Status : Running
IsScheduled : False
JobError :
JobType : DeleteClone
PolicyName :
JobResultData :
Running
Running
Running
Running
Completed
SmJobId : 110386
JobCreatedDateTime :
JobStartDateTime : 6/26/2024 10:01:33 AM
JobEndDateTime : 6/26/2024 10:02:38 AM
JobDuration : 00:01:05.5658860
JobName : Deleting clone 'hana-
1_sapcc_stl_netapp_com_hana_MDC_SS1__clone__110382_MDC_SS1_04-22-
2024_09.54.34'
JobDescription :
Status : Completed
IsScheduled : False
JobError :
JobType : DeleteClone
PolicyName :
JobResultData :
Clone delete job has been finshed.
PS C:\Windows\system32>
```

SAP system clone with SnapCenter

This section provides a step-by-step description for the SAP system clone operation, which can be used to set up a repair system to address logical corruption.

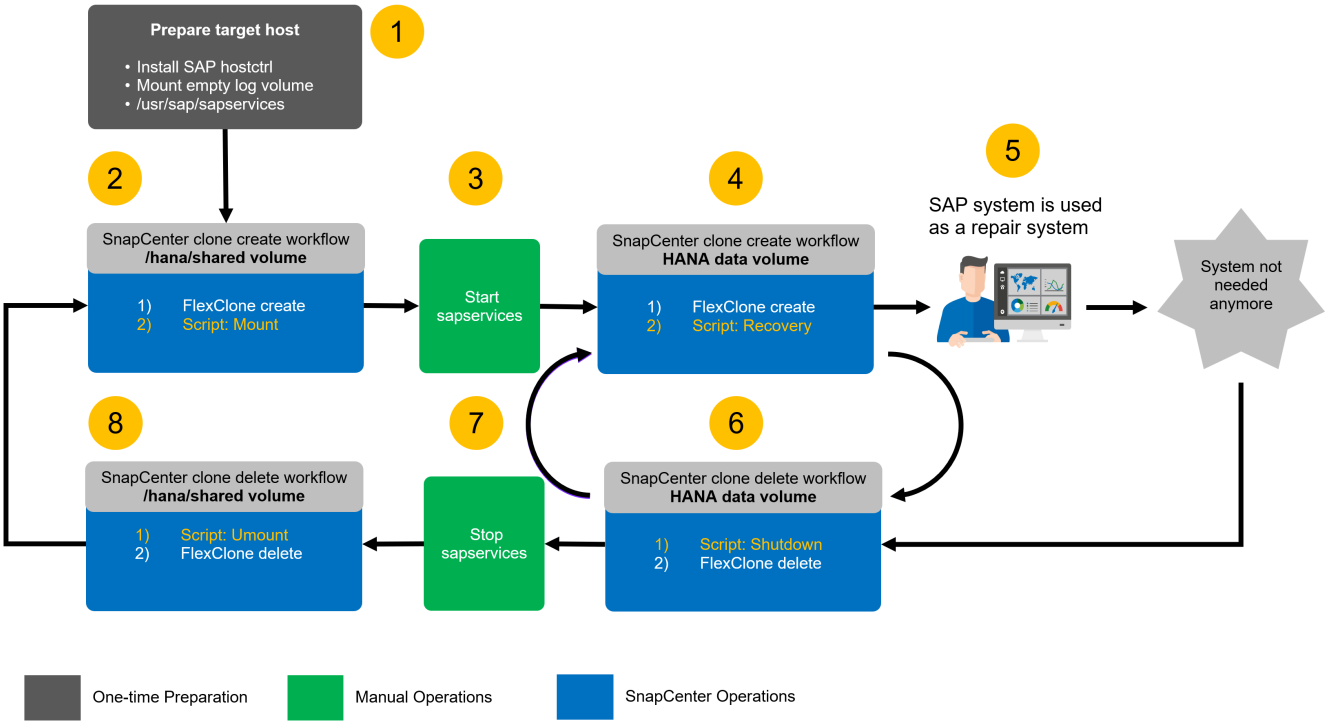
The figure below summarizes the required steps for an SAP system clone operation using SnapCenter.

1. Prepare the target host.
2. SnapCenter clone create workflow for the SAP HANA shared volume.
3. Start SAP HANA services.
4. SnapCenter clone create workflow for the SAP HANA data volume including database recovery.
5. The SAP HANA system can now be used as a repair system.

> If you must reset the system to a different Snapshot backup, then step 6 and step 4 are sufficient. The SAP HANA shared volume can continue to be mounted.

If the system is not needed anymore, the clean-up process is performed with the following steps.

6. SnapCenter clone delete workflow for the SAP HANA data volume including database shutdown.
7. Stop SAP HANA services.
8. SnapCenter clone delete workflow for the SAP HANA shared volume.



Prerequisites and limitations

The workflows described in the following sections have a few prerequisites and limitations regarding the SAP HANA system architecture and the SnapCenter configuration.

- The described workflow is valid for single host SAP HANA MDC systems. Multiple host systems are not supported.
- The SnapCenter SAP HANA plug-in must be deployed on the target host to enable the execution of automation scripts.
- The workflow has been validated for NFS. The automation `script sc-mount-volume.sh`, which is used to mount the SAP HANA shared volume, does not support FCP. This step must be either done manually or by extending the script.

- The described workflow is only valid for the SnapCenter 5.0 release or higher.

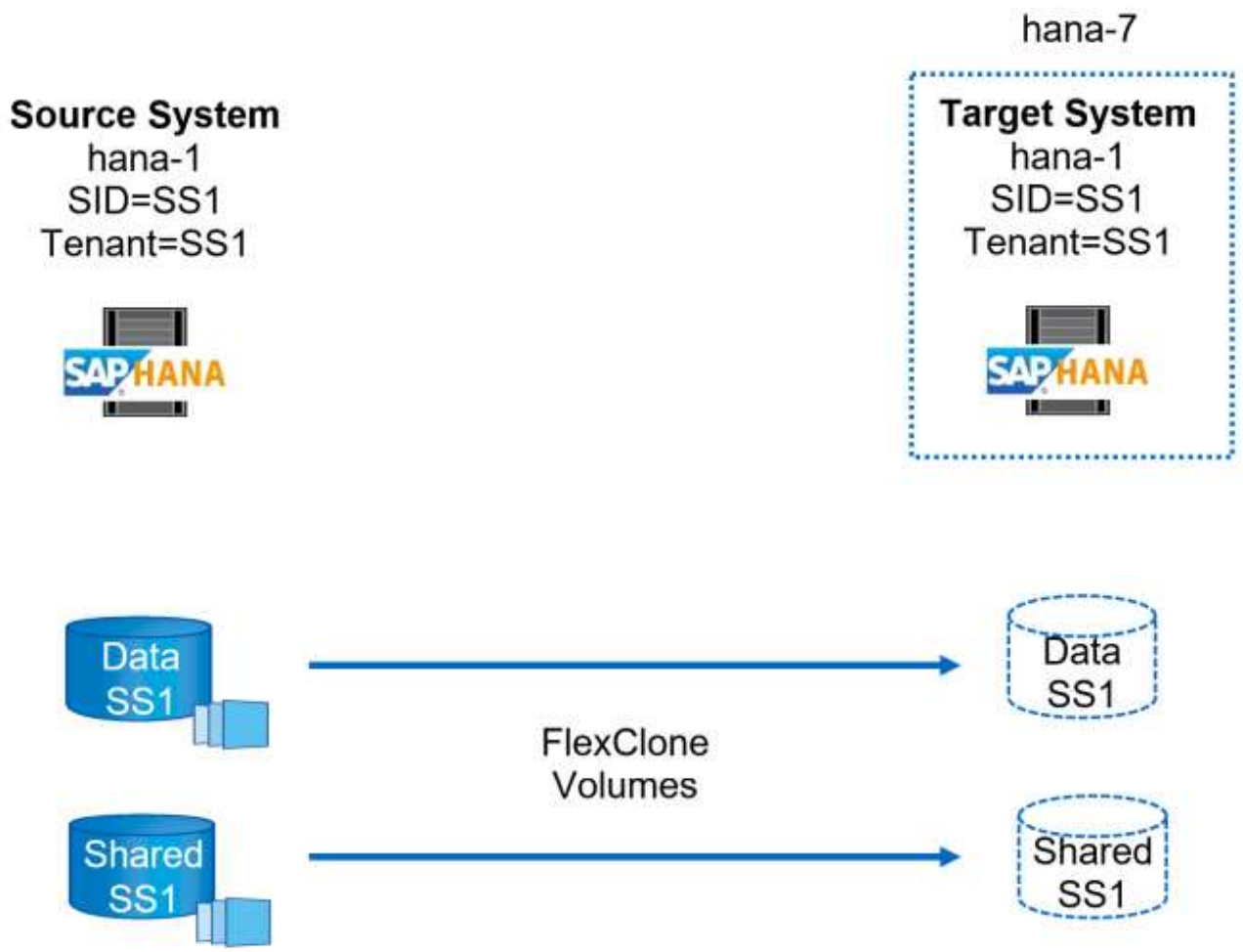
Lab setup

The figure below shows the lab setup used for system clone operation.

The following software versions were used:

- SnapCenter 5.0
- SAP HANA systems: HANA 2.0 SPS6 rev.61
- SLES 15
- ONTAP 9.7P7

All SAP HANA systems must be configured based on the configuration guide SAP HANA on NetApp AFF systems with NFS. SnapCenter and the SAP HANA resources were configured based on the best practice guide SAP HANA Backup and Recovery with SnapCenter.



Target host preparation

This section describes the preparation steps required at a server that is used as a system clone target.

During normal operation, the target host might be used for other purposes, for example, as an SAP HANA QA or test system. Therefore, most of the described steps must be executed when the system clone operation is

requested. On the other hand, the relevant configuration files, like `/etc/fstab` and `/usr/sap/sapservices`, can be prepared and then put in production simply by copying the configuration file.

The target host preparation also includes shutting down the SAP HANA QA or test system.

Target server host name and IP address

The host name of the target server must be identical to the host name of the source system. The IP address can be different.

> Proper fencing of the target server must be established so that it cannot communicate with other systems. If proper fencing is not in place, then the cloned production system might exchange data with other production systems.

> In our lab setup, we changed the host name of the target system only internally from the target system perspective. Externally the host was still accessible with the hostname hana-7. When logged into the host, the host itself is hana-1.

Install required software

The SAP host agent software must be installed at the target server. For full information, see the SAP Host Agent at the SAP help portal.

The SnapCenter SAP HANA plug-in must be deployed on the target host using the add host operation within SnapCenter.

Configure users, ports, and SAP services

The required users and groups for the SAP HANA database must be available at the target server. Typically, central user management is used; therefore, no configuration steps are necessary at the target server. The required ports for the SAP HANA database must be configured at the target hosts. The configuration can be copied from the source system by copying the /etc/services file to the target server.

The required SAP services entries must be available at the target host. The configuration can be copied from the source system by copying the `/usr/sap/sapservices` file to the target server. The following output shows the required entries for the SAP HANA database used in the lab setup.

```
#!/bin/sh
LD_LIBRARY_PATH=/usr/sap/SS1/HDB00/exe:$LD_LIBRARY_PATH;export
LD_LIBRARY_PATH;/usr/sap/SS1/HDB00/exe/sapstartsrv
pf=/usr/sap/SS1/SYS/profile/SS1_HDB00_hana-1 -D -u ss1adm
limit.descriptors=1048576
```

Prepare log and log backup volume

Because you do not need to clone the log volume from the source system and any recovery is performed with the clear log option, an empty log volume must be prepared at the target host.

Because the source system has been configured with a separate log backup volume, an empty log backup volume must be prepared and mounted to the same mount point as at the source system.

```
hana-1:/# cat /etc/fstab
192.168.175.117:/SS1_repair_log_mnt00001 /hana/log/SS1/mnt00001 nfs
rw,vers=3,hard,timeo=600,rsize=1048576,wsize=1048576,intr,noatime,nolock 0
0
192.168.175.117:/SS1_repair_log_backup /mnt/log-backup nfs
rw,vers=3,hard,timeo=600,rsize=1048576,wsize=1048576,intr,noatime,nolock 0
0
```

Within the log volume hdb*, you must create subdirectories in the same way as at the source system.

```
hana-1:/ # ls -al /hana/log/SS1/mnt00001/
total 16
drwxrwxrwx 5 root root 4096 Dec 1 06:15 .
drwxrwxrwx 1 root root 16 Nov 30 08:56 ..
drwxr-xr-- 2 ss1adm sapsys 4096 Dec 1 06:14 hdb00001
drwxr-xr-- 2 ss1adm sapsys 4096 Dec 1 06:15 hdb00002.00003
drwxr-xr-- 2 ss1adm sapsys 4096 Dec 1 06:15 hdb00003.00003
```

Within the log backup volume, you must create subdirectories for the system and the tenant database.

```
hana-1:/ # ls -al /mnt/log-backup/
total 12
drwxr-xr-- 2 ss1adm sapsys 4096 Dec 1 04:48 .
drwxr-xr-- 2 ss1adm sapsys 4896 Dec 1 03:42 ..
drwxr-xr-- 2 ss1adm sapsys 4096 Dec 1 06:15 DB_SS1
drwxr-xr-- 2 ss1adm sapsys 4096 Dec 1 06:14 SYSTEMDB
```

Prepare file system mounts

You must prepare mount points for the data and the shared volume.

With our example, the directories `/hana/data/SS1/mnt00001`, `/hana/shared` and `usr/sap/SS1` must be created.

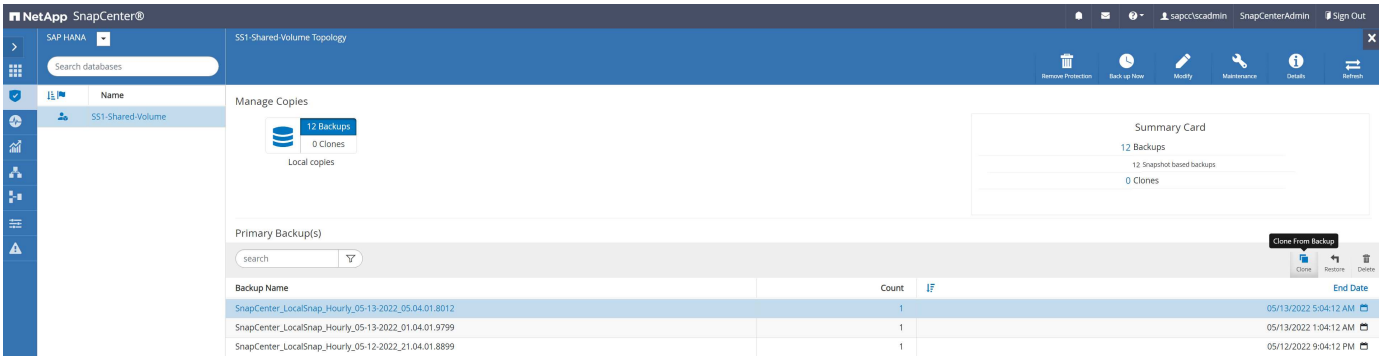
Prepare script execution

You must add the scripts, that should be executed at the target system to the SnapCenter allowed commands config file.

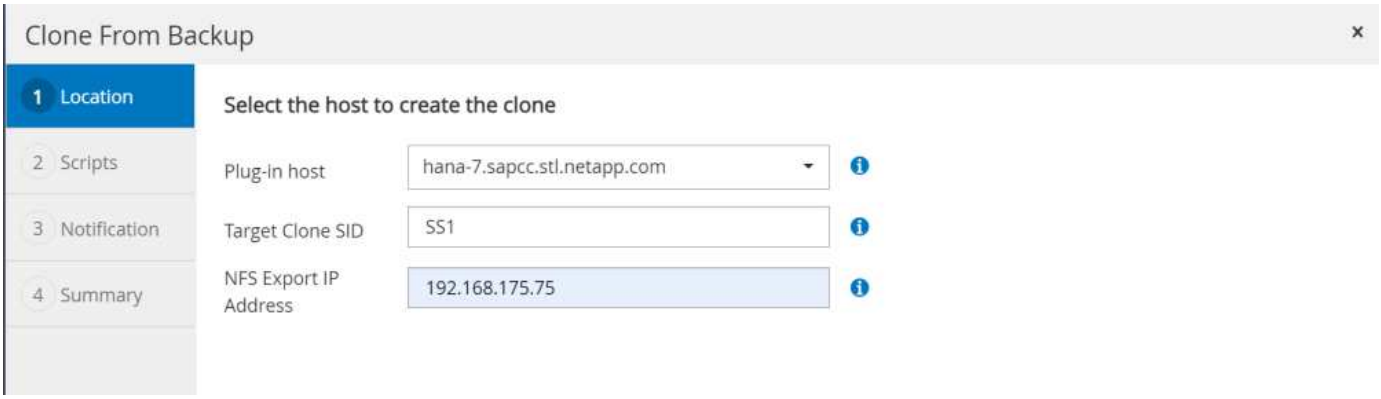
```
hana-7:/opt/NetApp/snapcenter/scc/etc # cat
/opt/NetApp/snapcenter/scc/etc/allowed_commands.config
command: mount
command: umount
command: /mnt/sapcc-share/SAP-System-Refresh/sc-system-refresh.sh
command: /mnt/sapcc-share/SAP-System-Refresh/sc-mount-volume.sh
hana-7:/opt/NetApp/snapcenter/scc/etc #
```

Cloning the HANA shared volume

1. Select a Snapshot backup from the source system SS1 shared volume and click Clone.

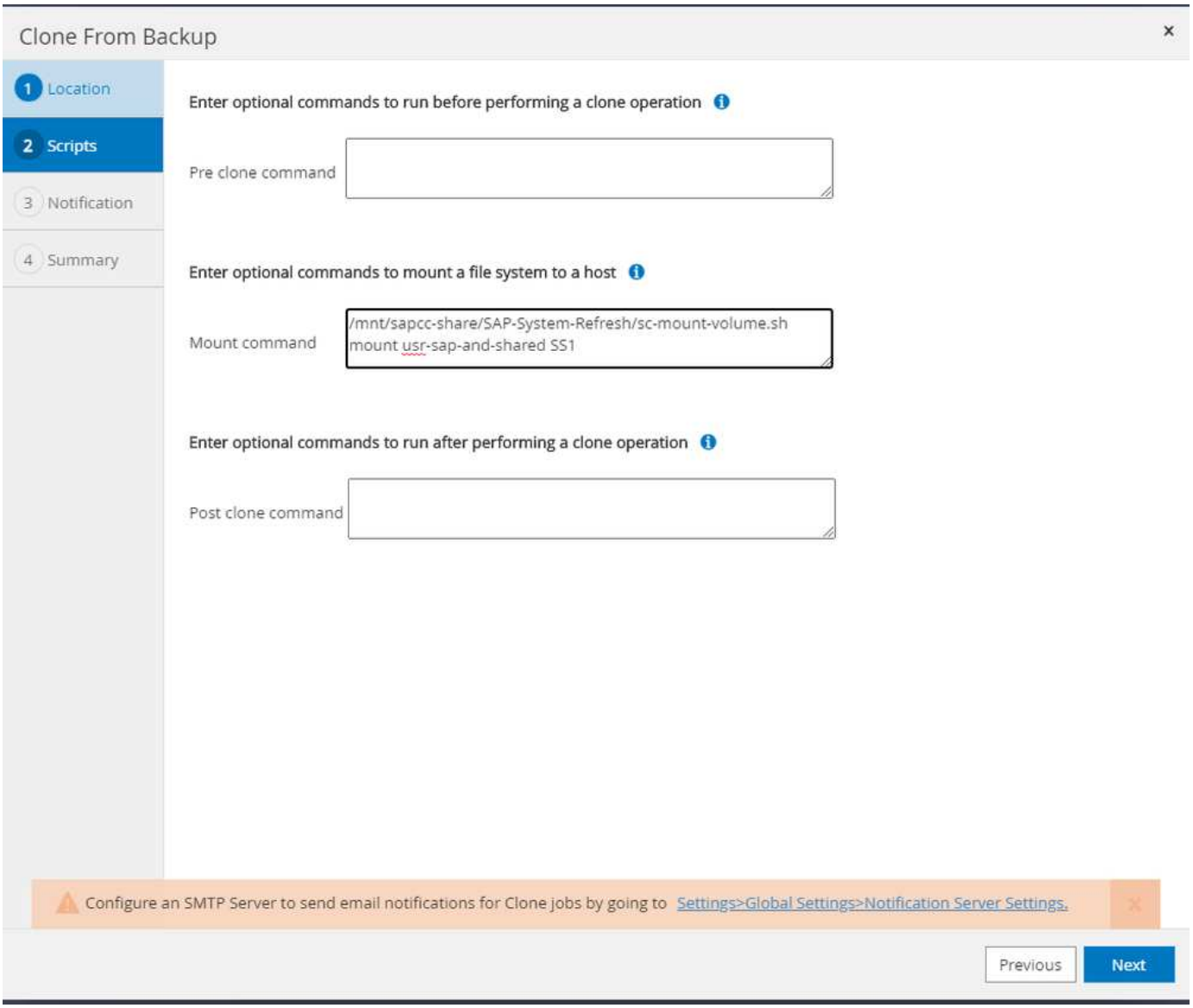


1. Select the host where the target repair system has been prepared. The NFS export IP address must be the storage network interface of the target host. As target SID keep the same SID as the source system. In our example SS1.

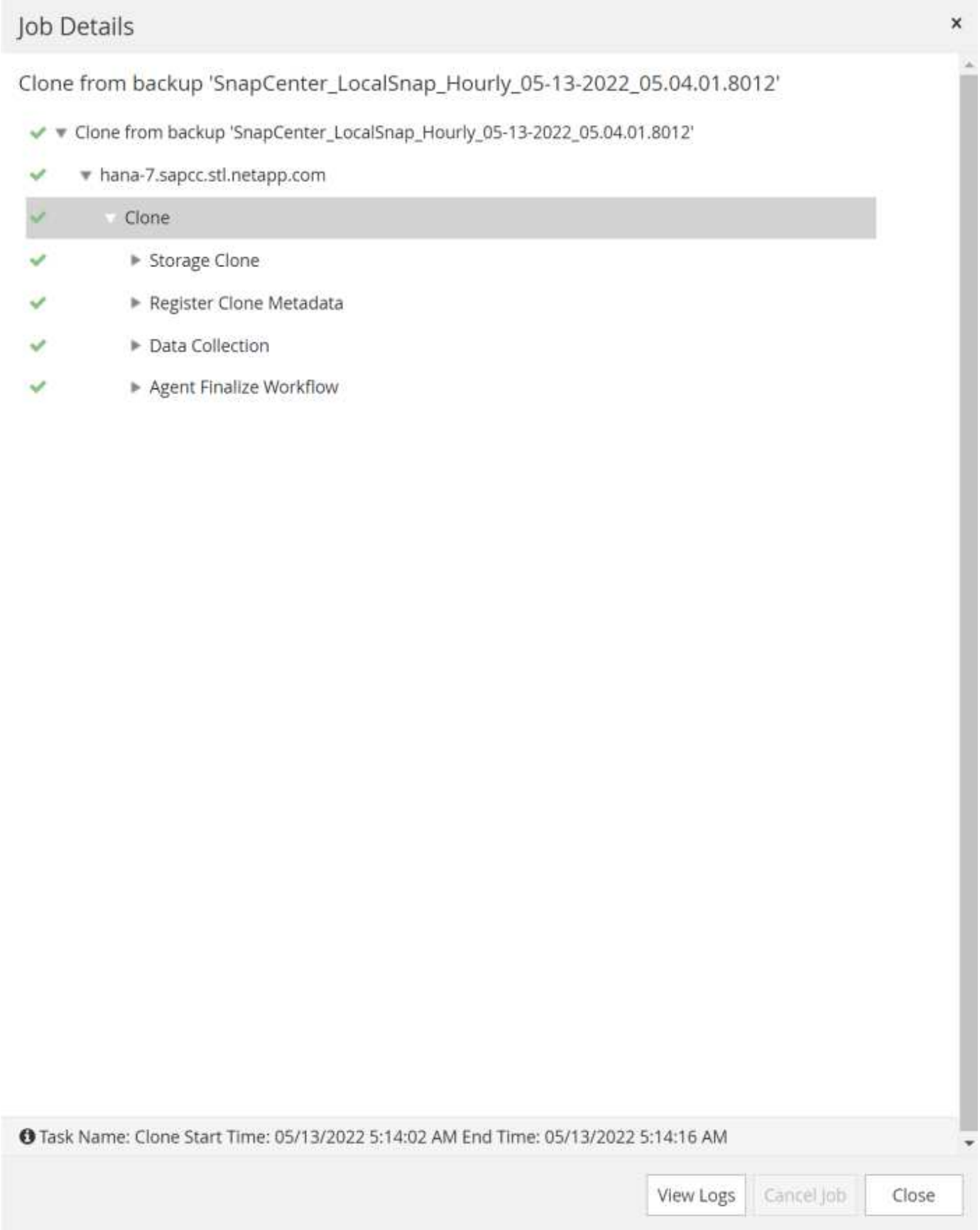


3. Enter the mount script with the required command line options.

> The SAP HANA system uses a single volume for `/hana/shared` as well as for `/usr/sap/SS1`, separated in subdirectories as recommended in the configuration guide SAP HANA on NetApp AFF systems with NFS. The script `sc-mount-volume.sh` supports this configuration using a special command line option for the mount path. If the mount path command line option is equal to usr-sap-and-shared, the script mounts the subdirectories shared and usr-sap in the volume accordingly.



4. The Job Details screen in SnapCenter shows the progress of the operation.



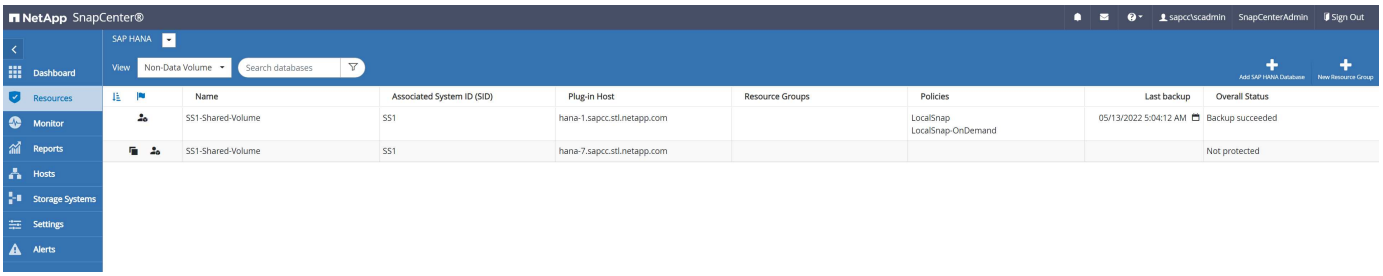
5. The logfile of the sc-mount-volume.sh script shows the different steps executed for the mount operation.

```
20201201041441###hana-1###sc-mount-volume.sh: Adding entry in /etc/fstab.
20201201041441###hana-1###sc-mount-volume.sh:
192.168.175.117://SS1_shared_Clone_05132205140448713/usr-sap /usr/sap/SS1
nfs
rw,vers=3,hard,timeo=600,rsize=1048576,wsize=1048576,intr,noatime,nolock 0
0
20201201041441###hana-1###sc-mount-volume.sh: Mounting volume: mount
/usr/sap/SS1.
20201201041441###hana-1###sc-mount-volume.sh:
192.168.175.117:/SS1_shared_Clone_05132205140448713/shared /hana/shared
nfs
rw,vers=3,hard,timeo=600,rsize=1048576,wsize=1048576,intr,noatime,nolock 0
0
20201201041441###hana-1###sc-mount-volume.sh: Mounting volume: mount
/hana/shared.
20201201041441###hana-1###sc-mount-volume.sh: usr-sap-and-shared mounted
successfully.
20201201041441###hana-1###sc-mount-volume.sh: Change ownership to ss1adm.
```

6. When the SnapCenter workflow is finished, the /usr/sap/SS1 and the /hana/shared filesystems are mounted at the target host.

```
hana-1:~ # df
Filesystem 1K-blocks Used Available Use% Mounted on
192.168.175.117:/SS1_repair_log_mnt00001 262144000 320 262143680 1%
/hana/log/SS1/mnt00001
192.168.175.100:/sapcc_share 1020055552 53485568 966569984 6% /mnt/sapcc-
share
192.168.175.117:/SS1_repair_log_backup 104857600 256 104857344 1%
/mnt/log-backup
192.168.175.117:/SS1_shared_Clone_05132205140448713/usr-sap 262144064
10084608 252059456 4% /usr/sap/SS1
192.168.175.117:/SS1_shared_Clone_05132205140448713/shared 262144064
10084608 252059456 4% /hana/shared
```

7. Within SnapCenter, a new resource for the cloned volume is visible.



8. Now that the /hana/shared volume is available, the SAP HANA services can be started.

```
hana-1:/mnt/sapcc-share/SAP-System-Refresh # systemctl start sapinit
```

9. SAP Host Agent and sapstartsrv processes are now started.

```
hana-1:/mnt/sapcc-share/SAP-System-Refresh # ps -ef |grep sap
root 12377 1 0 04:34 ? 00:00:00 /usr/sap/hostctrl/exe/saphostexec
pf=/usr/sap/hostctrl/exe/host_profile
sapadm 12403 1 0 04:34 ? 00:00:00 /usr/lib/systemd/systemd --user
sapadm 12404 12403 0 04:34 ? 00:00:00 (sd-pam)
sapadm 12434 1 1 04:34 ? 00:00:00 /usr/sap/hostctrl/exe/sapstartsrv
pf=/usr/sap/hostctrl/exe/host_profile -D
root 12485 12377 0 04:34 ? 00:00:00 /usr/sap/hostctrl/exe/saphostexec
pf=/usr/sap/hostctrl/exe/host_profile
root 12486 12485 0 04:34 ? 00:00:00 /usr/sap/hostctrl/exe/saposcol -l -w60
pf=/usr/sap/hostctrl/exe/host_profile
ss1adm 12504 1 0 04:34 ? 00:00:00 /usr/sap/SS1/HDB00/exe/sapstartsrv
pf=/usr/sap/SS1/SYS/profile/SS1_HDB00_hana-1 -D -u ss1adm
root 12582 12486 0 04:34 ? 00:00:00 /usr/sap/hostctrl/exe/saposcol -l -w60
pf=/usr/sap/hostctrl/exe/host_profile
root 12585 7613 0 04:34 pts/0 00:00:00 grep --color=auto sap
hana-1:/mnt/sapcc-share/SAP-System-Refresh #
```

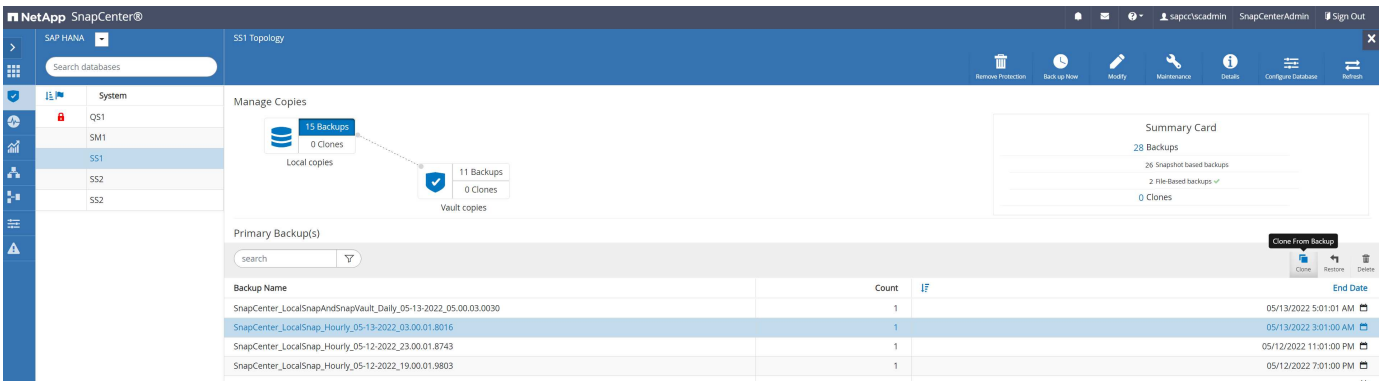
Cloning additional SAP application services

Additional SAP application services are cloned in the same way as the SAP HANA shared volume as described in the section “Cloning the SAP HANA shared volume.” Of course, the required storage volume(s) of the SAP application servers must be protected with SnapCenter as well.

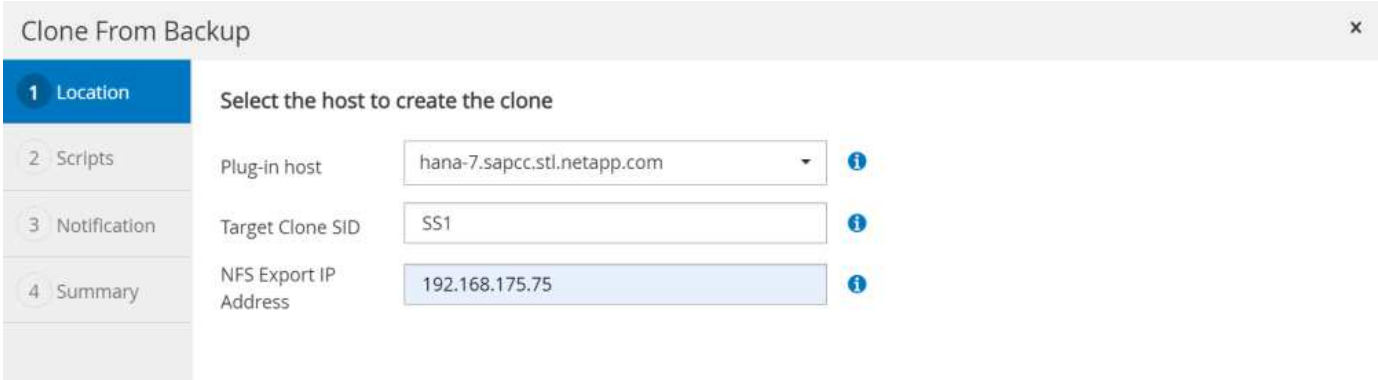
You must add the required services entries to /usr/sap/sapservices, and the ports, users, and the file system mount points (for example, /usr/sap/SID) must be prepared.

Cloning the data volume and recovery of the HANA database

1. Select an SAP HANA Snapshot backup from the source system SS1.

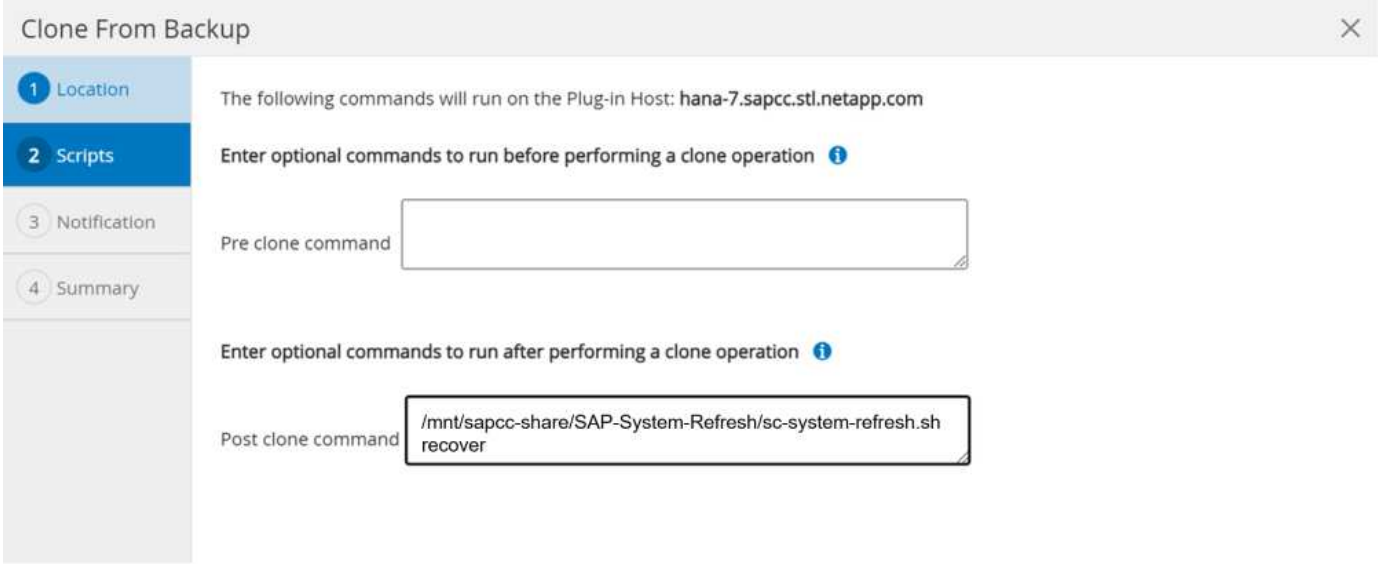


2. Select the host where the target repair system has been prepared. The NFS export IP address must be the storage network interface of the target host. As target SID keep the same SID as the source system. In our example SS1

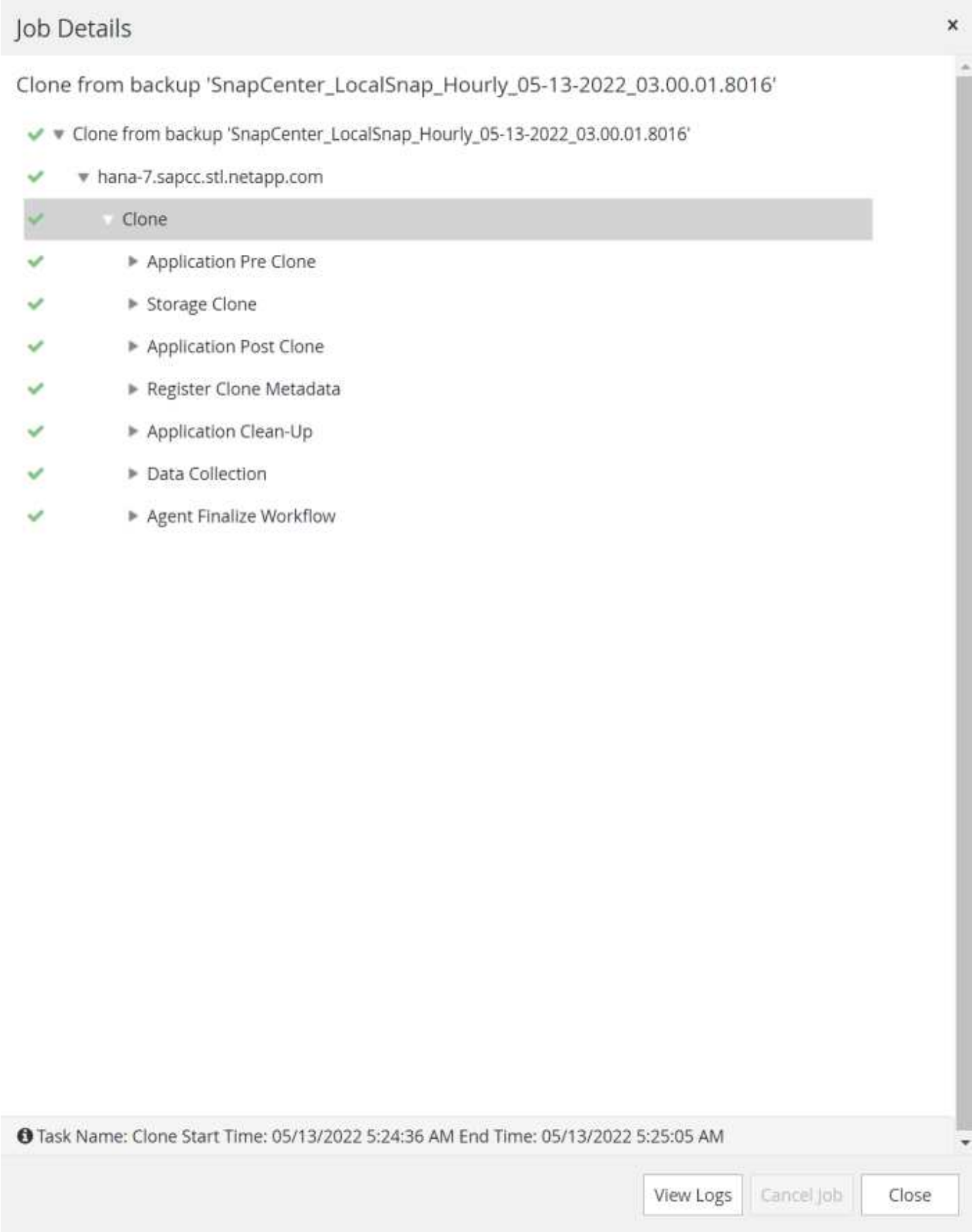


3. Enter the post-clone scripts with the required command line options.

> The script for the recovery operation recovers the SAP HANA database to the point in time of the Snapshot operation and does not execute any forward recovery. If a forward recovery to a specific point in time is required, the recovery must be performed manually. A manual forward recovery also requires that the log backups from the source system are available at the target host.



The job details screen in SnapCenter shows the progress of the operation.



The logfile of the `sc-system-refresh` script shows the different steps that are executed for the mount and the recovery operation.

```
20201201052124###hana-1###sc-system-refresh.sh: Recover system database.
20201201052124###hana-1###sc-system-refresh.sh:
/usr/sap/SS1/HDB00/exe/Python/bin/python
/usr/sap/SS1/HDB00/exe/python_support/recoverSys.py --command "RECOVER
DATA USING SNAPSHOT CLEAR LOG"
20201201052156###hana-1###sc-system-refresh.sh: Wait until SAP HANA
database is started ....
20201201052156###hana-1###sc-system-refresh.sh: Status: GRAY
20201201052206###hana-1###sc-system-refresh.sh: Status: GREEN
20201201052206###hana-1###sc-system-refresh.sh: SAP HANA database is
started.
20201201052206###hana-1###sc-system-refresh.sh: Source system has a single
tenant and tenant name is identical to source SID: SS1
20201201052206###hana-1###sc-system-refresh.sh: Target tenant will have
the same name as target SID: SS1.
20201201052206###hana-1###sc-system-refresh.sh: Recover tenant database
SS1.
20201201052206###hana-1###sc-system-refresh.sh:
/usr/sap/SS1/SYS/exe/hdb/hdbsql -U SS1KEY RECOVER DATA FOR SS1 USING
SNAPSHOT CLEAR LOG
0 rows affected (overall time 34.773885 sec; server time 34.772398 sec)
20201201052241###hana-1###sc-system-refresh.sh: Checking availability of
Indexserver for tenant SS1.
20201201052241###hana-1###sc-system-refresh.sh: Recovery of tenant
database SS1 succesfully finished.
20201201052241###hana-1###sc-system-refresh.sh: Status: GREEN
After the recovery operation, the HANA database is running and the data
volume is mounted at the target host.
hana-1:/mnt/log-backup # df
Filesystem 1K-blocks Used Available Use% Mounted on
192.168.175.117:/SS1_repair_log_mnt00001 262144000 760320 261383680 1%
/hana/log/SS1/mnt00001
192.168.175.100:/sapcc_share 1020055552 53486592 966568960 6% /mnt/sapcc-
share
192.168.175.117:/SS1_repair_log_backup 104857600 512 104857088 1%
/mnt/log-backup
192.168.175.117:/SS1_shared_Clone_05132205140448713/usr-sap 262144064
10090496 252053568 4% /usr/sap/SS1
192.168.175.117:/SS1_shared_Clone_05132205140448713/shared 262144064
10090496 252053568 4% /hana/shared
192.168.175.117:/SS1_data_mnt00001_Clone_0421220520054605 262144064
3732864 258411200 2% /hana/data/SS1/mnt00001
```

The SAP HANA system is now available and can be used, for example, as a repair system.

Where to find additional information and version history

To learn more about the information described in this document, refer to the following documents and/or websites:

- SAP Business Application and SAP HANA Database Solutions (netapp.com)
- TR-4614: SAP HANA Backup and Recovery with SnapCenter
- TR-4436: SAP HANA on NetApp All Flash FAS Systems with Fibre Channel Protocol
- TR-4435: SAP HANA on NetApp All Flash FAS Systems with NFS
- TR-4926: SAP HANA on Amazon FSx for NetApp ONTAP - Backup and recovery with SnapCenter
- TR-4953: NetApp SAP Landscape Management Integration using Ansible
- TR-4929: Automating SAP system copy operations with Libelle SystemCopy (netapp.com)
- Automating SAP system copy, refresh, and clone workflows with ALPACA and NetApp SnapCenter
- Automating SAP system copy, refresh, and clone workflows with Avantra and NetApp SnapCenter

Version	Date	Document Version History
Version 1.0	February 2018	Initial release.
Version 2.0	February 2021	Complete rewrite covering SnapCenter 4.3 and improved automation scripts. New workflow description for system refresh and system clone operations.
Version 3.0	May 2022	Adapted to changed workflow with SnapCenter 4.6 P1
Version 4.0	July 2024	Document covers NetApp systems on-premises, FSx for ONTAP and Azure NetApp Files New SnapCenter 5.0 operations mount and unmount during clone create and delete workflows Added specific steps for Fibre Channel SAN Added specific steps for Azure NetApp Files Adapted and simplified `sc-system-refresh` script Included required steps for enabled SAP HANA volume encryption

Automating SAP system copy operations with Libelle SystemCopy

TR-4929: Automating SAP system copy operations with Libelle SystemCopy

Holger Zecha, Tobias Brandl, NetApp
Franz Diegruber, Libelle

In today's dynamic business environment, companies must provide ongoing innovation and react quickly to changing markets. Under these competitive circumstances, companies that implement greater flexibility in their work processes can adapt to market demands more effectively.

Changing market demands also affect a company's SAP environments such that they require regular

integrations, changes, and updates. IT departments must implement these changes with fewer resources and over shorter time periods. Minimizing risk when deploying those changes requires thorough testing and training which require additional SAP systems with actual data from production.

Traditional SAP lifecycle-management approaches to provision these systems are primarily based on manual processes. These manual processes are often error-prone and time-consuming, delaying innovation and the response to business requirements.

NetApp solutions for optimizing SAP lifecycle management are integrated into SAP AnyDBs and SAP HANA databases. In addition, NetApp integrates into SAP lifecycle management tools, combining efficient application-integrated data protection with the flexible provisioning of SAP test systems.

While these NetApp solutions solve the issue of efficiently managing enormous amounts of data even for the largest databases, full end-to-end SAP system- copy and refresh operations have to include pre- and post-copy activities to completely change the identity of the source SAP system to the target system. SAP describes the required activities in their SAP homogenous system copy guide. To further reduce the number of manual processes and to improve the quality and stability of a SAP system copy process, our partner Libelle has developed the Libelle SystemCopy (LSC) tool. We have jointly worked with Libelle to integrate the NetApp solutions for SAP system copies into LSC to provide full end-to-end automated system copies in record time.

Application-integrated Snapshot copy operation

The ability to create application-consistent NetApp Snapshot copies on the storage layer is the foundation for the system copy and system clone operations described in this document. Storage-based Snapshot copies are created with the NetApp SnapCenter Plug-In for SAP HANA or SAP Any DBs on native NetApp ONTAP systems or by using the Microsoft Azure Application Consistent Snapshot tool (AzAcSnap) and interfaces provided by the SAP HANA and Oracle database running in Microsoft Azure. When using SAP HANA, SnapCenter and AzAcSnap register Snapshot copies in the SAP HANA backup catalog so that the backups can be used for restore and recovery as well as for cloning operations.

Off-site backup and/or disaster recovery data replication

Application-consistent Snapshot copies can be replicated on the storage layer to an off-site backup site or a disaster recovery site controlled by SnapCenter on-premises. Replication is based on block changes and is therefore space and bandwidth efficient. The same technology is available for SAP HANA and Oracle systems running in Azure with Azure NetApp Files by using the Cross Region Replication (CRR) feature to efficiently replicate Azure NetApp Files volumes between Azure regions.

Use any Snapshot copy for SAP system copy or clone operations

NetApp technology and software integration allows you to use any Snapshot copy of a source system for an SAP system copy or clone operation. This Snapshot copy can be either selected from the same storage that is used for the SAP production systems, the storage that is used for off-site backups (such as Azure NetApp Files backup in Azure), or the storage at the disaster recovery site (Azure NetApp Files CRR target volumes). This flexibility allows you to separate development and test systems from production if required and covers other scenarios, such as the testing of disaster recovery at the disaster recovery site.

Automation with integration

There are various scenarios and use cases for the provisioning of SAP test systems, and you might also have different requirements for the level of automation. NetApp software products for SAP integrate into database and lifecycle management products from SAP and other third-party vendors (for example, Libelle) to support different scenarios and levels of automation.

NetApp SnapCenter with the plug-in for SAP HANA and SAP AnyDBs or AzAcSnap on Azure is used to

provision the required storage- volume clones based on an application-consistent Snapshot copy and to execute all required host and database operations up to a started SAP database. Depending on the use case, SAP system copy, system clone, system refresh, or additional manual steps such as SAP postprocessing might be required. More details are covered in the next section.

A fully automated, end-to-end provisioning or refresh of SAP test systems can be performed by using Libelle SystemCopy (LSC) automation. The integration of SnapCenter or AzAcSnap into LSC is described in more detail in this document.

Libelle SystemCopy

Libelle SystemCopy is a framework-based software solution to create fully automated system and landscape copies. With the proverbial touch of a button, QA and test systems can be updated with fresh production data. Libelle SystemCopy supports all conventional databases and operating systems, provides its own copy mechanisms for all platforms but, at the same time, integrates backup/restore procedures or storage tools such as NetApp Snapshot copies and NetApp FlexClone volumes. The activities that are necessary during a system copy are controlled from outside the SAP ABAP stack. In this way, no transports or other changes are required in the SAP applications. Generally, all steps necessary to successfully complete a system copy procedure can be categorized into four steps:

- **Check phase.** Check the involved system environments.
- **Pre phase.** Prepare the target system for a system copy.
- **Copy phase.** Provide a copy of the actual production database to the target system from the source.
- **Post phase.** All tasks after the copy to complete the homogeneous system copy procedure and to provide an updated target system.

During the copy phase, NetApp Snapshot and FlexClone functionality is used to minimize the time needed to a couple of minutes even for the largest databases.

For the Check, Pre, and Post phases, LSC comes with 450+ preconfigured tasks covering 95% of typical refresh operations. As a result, LSC embraces automation following SAP standards. Due to the software-defined nature of LSC, system refresh processes can be easily adjusted and enhanced to meet the specific needs of customer SAP environments.

Use cases for SAP system refresh and cloning

There are multiple scenarios in which data from a source system must be made available to a target system:

- Regular refresh of quality assurance and test and training systems
- Creating break fix or repair system environments to address logical corruption
- Disaster recovery test scenarios

Although repair systems and disaster recovery test systems are typically provided using SAP system clones (which don’t require extensive post-processing operations) for refreshed test and training systems, these post-processing steps must be applied to enable coexistence with the source system. Therefore, this document focuses on SAP system refresh scenarios. More details about the different use cases can be found in the technical report TR-4667: Automating SAP HANA System Copy and Clone Operations with SnapCenter.

The remainder of this document is separated into two parts. The first part describes the integration of NetApp SnapCenter with Libelle SystemCopy for SAP HANA and SAP AnyDBs systems running on NetApp ONTAP systems on-premises. The second part describes the integration of AzAcSnap with LSC for SAP HANA systems running in Microsoft Azure with Azure NetApp Files provided. Although the underlaying ONTAP technology is identical, Azure NetApp Files provides different interfaces and tool integration (for example,

AzAcSnap) compared to native ONTAP installation.

SAP HANA system refresh with LSC and SnapCenter

This section describes how to integrate LSC with NetApp SnapCenter. The integration between LSC and SnapCenter supports all SAP- supported databases. Nevertheless, we must differentiate between SAP AnyDBs and SAP HANA because SAP HANA provides a central communication host that is not available for SAP AnyDBs.

The default SnapCenter agent and database plug-in installation for SAP AnyDBs is a local installation from the SnapCenter agent in addition to the corresponding database plug-in for the database server.

In this section, the integration between LSC and SnapCenter is described using an SAP HANA database as an example. As previously stated for SAP HANA, there are two different options for the installation of the SnapCenter agent and SAP HANA database plug-in:

- **A standard SnapCenter agent and SAP HANA Plug-in installation.** In a standard installation, the SnapCenter agent and the SAP HANA Plug-in are locally installed on the SAP HANA database server.
- **A SnapCenter installation with a central communication host.** A central communication host is installed with the SnapCenter agent, the SAP HANA Plug-in, and the HANA database client that handles all database-related operations needed to back up and restore an SAP HANA database for several SAP HANA systems in the landscape. Therefore, a central communication host does not need to have a complete SAP HANA database system installed.

For more details regarding these different SnapCenter agents and SAP HANA database plug-in installation options, see the technical report TR-4614: SAP HANA backup and recovery with SnapCenter.

The following sections highlight the differences between integrating LSC with SnapCenter using either the standard installation or the central communication host. Notably, all configuration steps that are not highlighted are the same regardless of the installation option and the database used.

To perform an automated Snapshot copy-based backup from the source database and create a clone for the new target database, the described integration between LSC and SnapCenter uses the configuration options and scripts described in TR-4667: Automating SAP HANA System Copy and Clone Operations with SnapCenter.

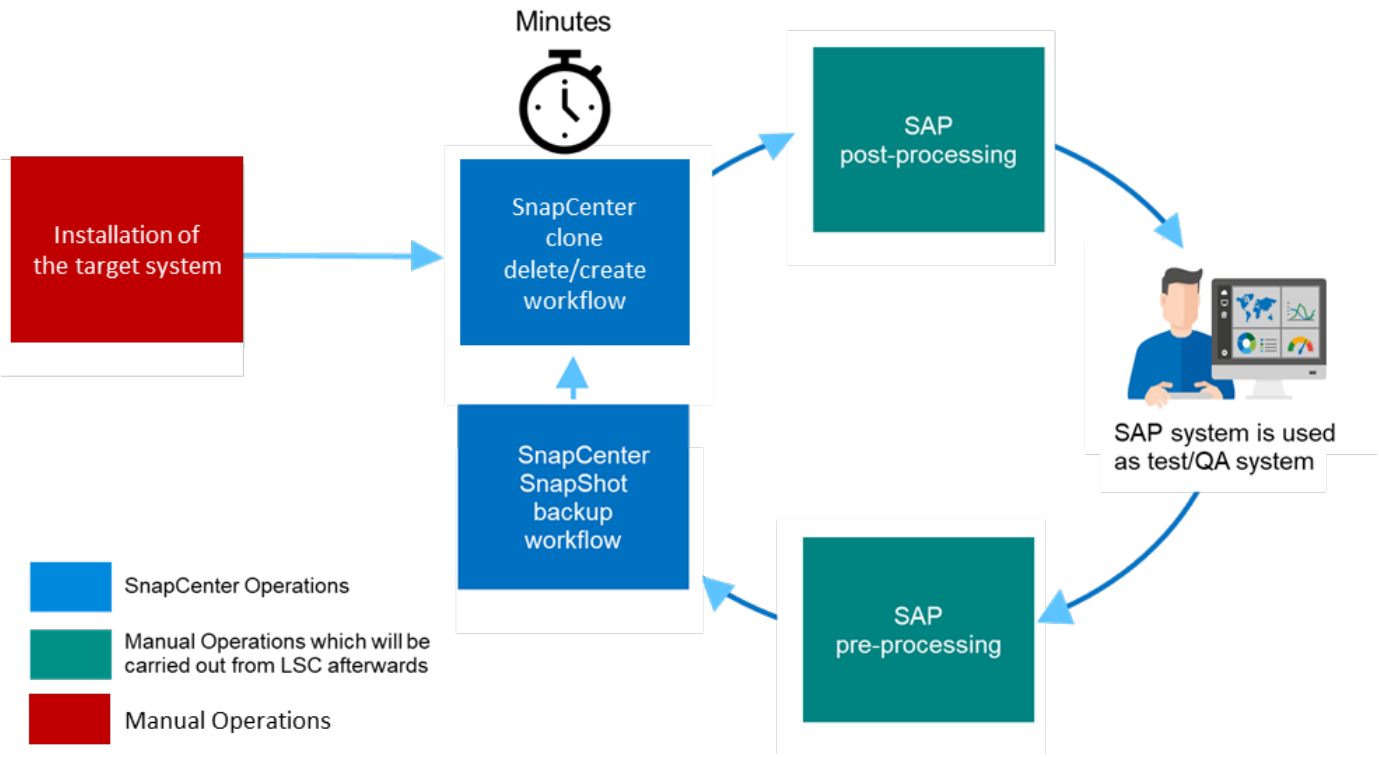
Overview

The following figure shows a typical high-level workflow for an SAP system refresh lifecycle with SnapCenter without LSC:

1. A one-time, initial installation and preparation of the target system.
2. Manual preprocessing (exporting licenses, users, printers, and so on).
3. If necessary, the deletion of an already existing clone on the target system.
4. The cloning of an existing Snapshot copy of the source system to the target system performed by SnapCenter.
5. Manual SAP post-processing operations (importing licenses, users, printers, disabling batch jobs, and so on).
6. The system can then be used as test or QA system.
7. When a new system refresh is requested, the workflow restarts at step 2.

SAP customers know that the manual steps colored in green in the figure below are time consuming and error prone. When using LSC and SnapCenter integration, these manual steps are carried out with LSC in a reliable and repeatable manner with all necessary logs needed for internal and external audits.

The following figure provides an overview of the general SnapCenter-based SAP system refresh procedure.



Prerequisites and limitations

The following prerequisites must be fulfilled:

- SnapCenter must be installed. The source and target system must be configured in SnapCenter, either in a standard installation or by using a central communication host. Snapshot copies can be created on the source system.
- The storage backend must be configured properly in SnapCenter, as shown in the image below.

Storage Connections

Name	IP	Cluster Name	User Name	Controller License
svm-trident		grenada.muccbc.hq.netapp.com		✔
svm-sap02	10.65.58.253	grenada.muccbc.hq.netapp.com		✔
svm-sap01	10.65.58.252	grenada.muccbc.hq.netapp.com		✔

The next two images cover the standard installation in which the SnapCenter agent and the SAP HANA Plug-in are installed locally on each database server.

The SnapCenter agent and the appropriate database plug-in must be installed on the source database.

Name	Type	System	Plug-in	Version	Overall Status
sap-lnx35.muccbc.hq.netapp.com	Linux	Stand-alone	UNIX, SAP HANA	4.3.1	Running

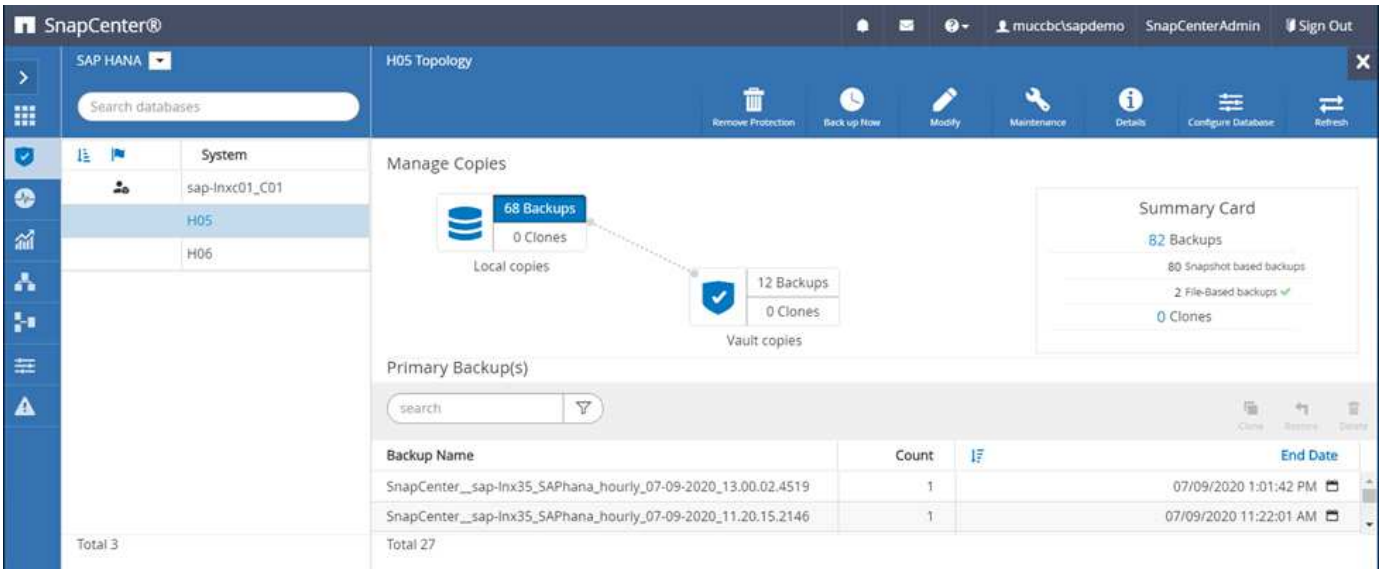
The SnapCenter agent and the appropriate database plug-in must be installed on the target database.

☐	sap-lnx36.muccbc.hq.netapp.com	Linux	Stand-alone	UNIX, SAP HANA	4.3.1	Running

The following image portrays central communication-host deployment in which the SnapCenter agent, the SAP HANA Plug-in, and the SAP HANA database client are installed on a centralized server (such as the SnapCenter Server) to manage several SAP HANA systems in the landscape.

The SnapCenter agent, the SAP HANA database plug-in, and the HANA database client must be installed on the central communication host.

The backup for the source database must be configured properly in SnapCenter so that the Snapshot copy can be successfully created.



The LSC master and the LSC worker must be installed in the SAP environment. In this deployment, we also installed the LSC master on the SnapCenter Server and the LSC worker on the target SAP database server, which should be refreshed. More details are described in the following section "Lab setup."

Documentation resources:

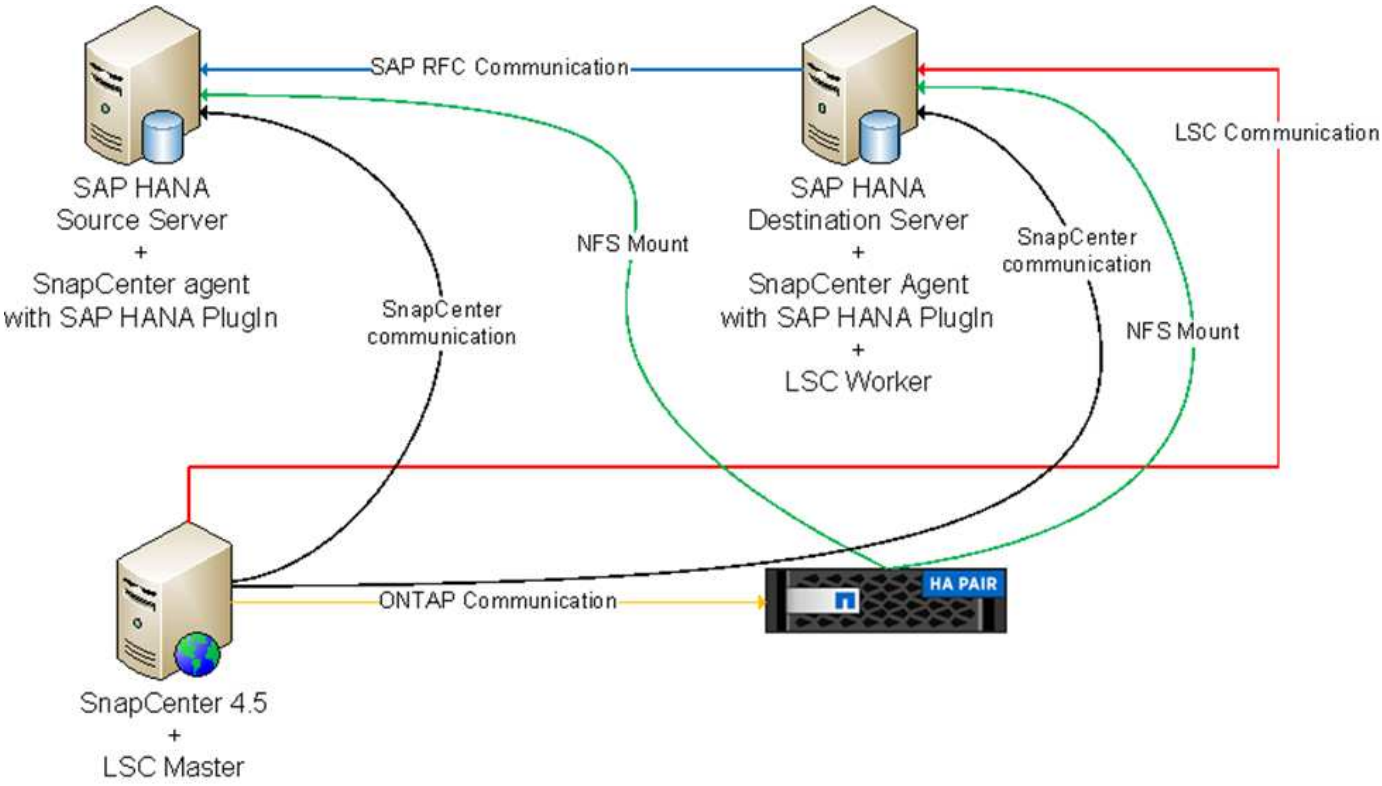
- SnapCenter Documentation Center
- TR-4700: SnapCenter Plug-In for Oracle Database
- TR-4614: SAP HANA Backup and Recovery with SnapCenter
- TR-4667: Automating SAP HANA System Copy and Clone Operations with SnapCenter
- TR-4769 -SnapCenter Best Practices and Sizing Guidelines
- SnapCenter 4.6 Cmdlet Reference Guide

Lab setup

This section describes an example architecture that was set up in a demo data center. The setup was divided into a standard installation and an installation using a central communication host.

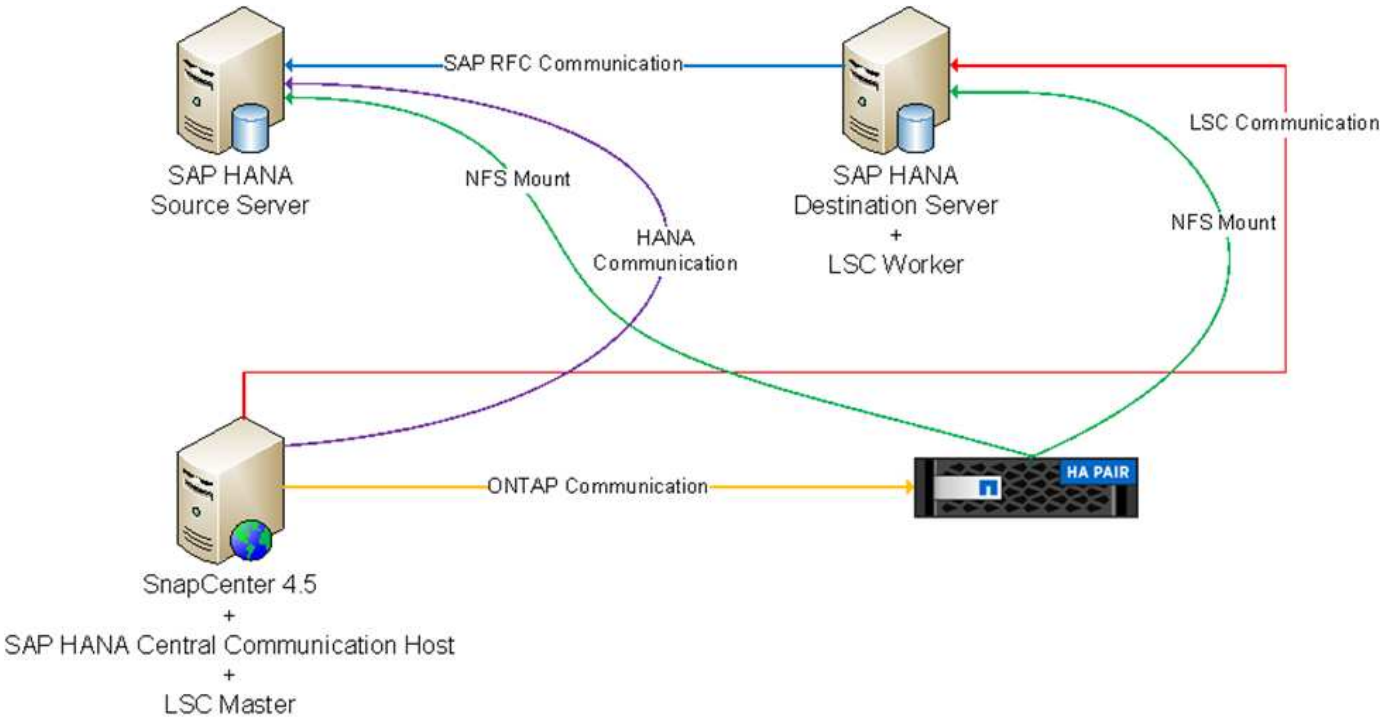
Standard installation

The following figure shows a standard installation in which the SnapCenter agent together with the database plug-in was installed locally on the source and the target database server. In the lab setup, we installed the SAP HANA Plug-in. In addition, the LSC worker was also installed on the target server. For simplification and to reduce the number of virtual servers, we installed the LSC master on the SnapCenter Server. The communication between the different components is illustrated in the following figure.



Central communication host

The following figure shows the setup using a central communication host. In this configuration, the SnapCenter agent together with the SAP HANA Plug-in and the HANA database client was installed on a dedicated server. In this setup, we used the SnapCenter Server to install the central communication host. In addition, the LSC worker was again installed on the target server. For simplification and to reduce the number of virtual servers, we decided to also install the LSC master on the SnapCenter Server. The communication between the different components is illustrated in the figure below.



Initial one-time preparation steps for Libelle SystemCopy

There are three main components of an LSC installation:

- **LSC master.** As the name suggests, this is the master component that controls the automatic workflow of a Libelle-based system copy. In the demo environment, the LSC master was installed on the SnapCenter Server.
- **LSC worker.** An LSC worker is the part of the Libelle software that typically runs on the target SAP system and executes the scripts required for the automated system copy. In the demo environment, the LSC worker was installed on the target SAP HANA application server.
- **LSC satellite.** An LSC satellite is a part of the Libelle software that runs on a third-party system on which further scripts must be executed. The LSC master can also fulfill the role of an LSC satellite system at the same time.

We first defined all the involved systems inside LSC, as shown in the following image

- **172.30.15.35.** The IP address of the SAP source system and the SAP HANA source system.
- **172.30.15.3.** The IP address of the LSC master and the LSC satellite system for this configuration. Because we installed the LSC master on the SnapCenter Server, the SnapCenter 4.x PowerShell Cmdlets are already available on this Windows host because they were installed during the SnapCenter Server installation. So, we decided to enable the LSC satellite role for this system and execute all SnapCenter PowerShell Cmdlets on this host. If you use a different system, make sure you install the SnapCenter PowerShell Cmdlets on this host according to the SnapCenter documentation.
- **172.30.15.36.** The IP address of the SAP destination system, the SAP HANA destination system, and the LSC worker.

Instead of IP addresses, host names, or fully qualified domain names can also be used.

The following image shows the LSC configuration of the master, worker, satellite, SAP source, SAP target, source database, and target database.

System Identifier	Worker	Source SAP	Source Database	Target SAP	Target Database	Satellite System
172.30.15.35		☑	☑	☐	☐	☐
172.30.15.3	172.30.15.3:9000	☐	☐	☐	☐	☑
172.30.15.36	172.30.15.36:9000	☐	☐	☑	☑	☐

For the main integration, we must again separate the configuration steps into the standard installation and the installation using a central communication host.

Standard installation

This section describes the configuration steps needed when using a standard installation where the SnapCenter agent and the necessary database plug-in are installed on the source and target systems. When using a standard installation, all tasks needed to mount the clone volume and to restore and recover the target system are carried out from the SnapCenter agent that is running on the target database system on the server itself. This allows access to all the clone-related details that are available through environmental variables from the SnapCenter agent. Therefore, you only need to create one additional task in the LSC copy phase. This task carries out the Snapshot copy process on the source database system and the clone and restore and recovery process on the target database system. All SnapCenter related tasks are triggered by using a PowerShell script that is entered in the LSC task `NTAP_SYSTEM_CLONE`.

The following image shows LSC task configuration in the copy phase.

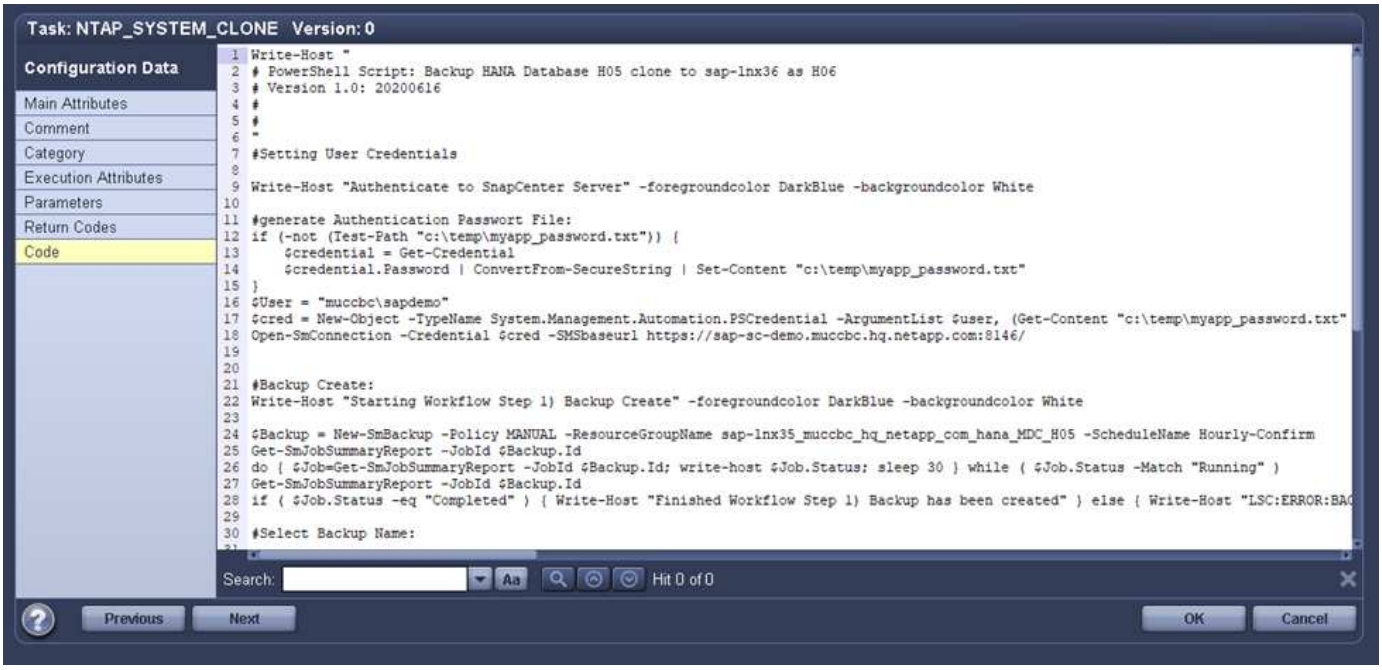
copy	Copy Phase		phase
copy 1	NTAP_SYSTEM_CLONE	NetApp SnapShot and Clone	psh
copy 2	NTAP_SYSTEM_CLONE_CP	NetApp SnapShot and Clone	psh
copy 3	NTAP_MNT_RECOVER_CP	Mount Volume and Recover HANA Database	cmd
copy 4	LPDBBCKP	Backup Source DB in Filesystem	lsh
copy 5	LPDBCPYFLS	Copy DB Backup Files From Source to Target System	lsh
copy 6	LTDBRESTORE	Restore DB Files	lsh
copy 7	LTDBRESTORE_TENANT	Restore DB Files for Tenant Database	lsh
post	**Post Phase**		**phase**

The following image highlights the configuration of the `NTAP_SYSTEM_CLONE` process. Because you are executing a PowerShell script, this Windows PowerShell script is executed on the satellite system. In this instance, this is the SnapCenter Server with the installed LSC master that also acts as a satellite system.

Because LSC must be made aware of whether the Snapshot copy, cloning, and recovery operation has been successful, you must define at least two return code types. One code is for a successful execution of the script, and the other code is for a failed execution of the script, as shown in the following image.

- `LSC:OK` must be written from the script to standard out if the execution was successful.
- `LSC:ERROR` must be written from the script to standard out if the execution has failed.

The following image shows part of the PowerShell script that must run to execute a Snapshot-based backup on the source database system and a clone on the target database system. The script is not intended to be complete. Rather, the script shows how integration between LSC and SnapCenter can look and how easy it is to set it up.



Because the script is executed on the LSC master (which is also a satellite system), the LSC master on the SnapCenter Server must be run as a Windows user that has appropriate permissions to execute backup and

cloning operations in SnapCenter. To verify whether the user has appropriate permission, the user should be able execute a Snapshot copy and a clone in the SnapCenter UI.

There is no need to run the LSC master and the LSC satellite on the SnapCenter Server itself. The LSC master and the LSC satellite can run on any Windows machine. The prerequisite for running the PowerShell script on the LSC satellite is that the SnapCenter PowerShell cmdlets have been installed on the Windows Server.

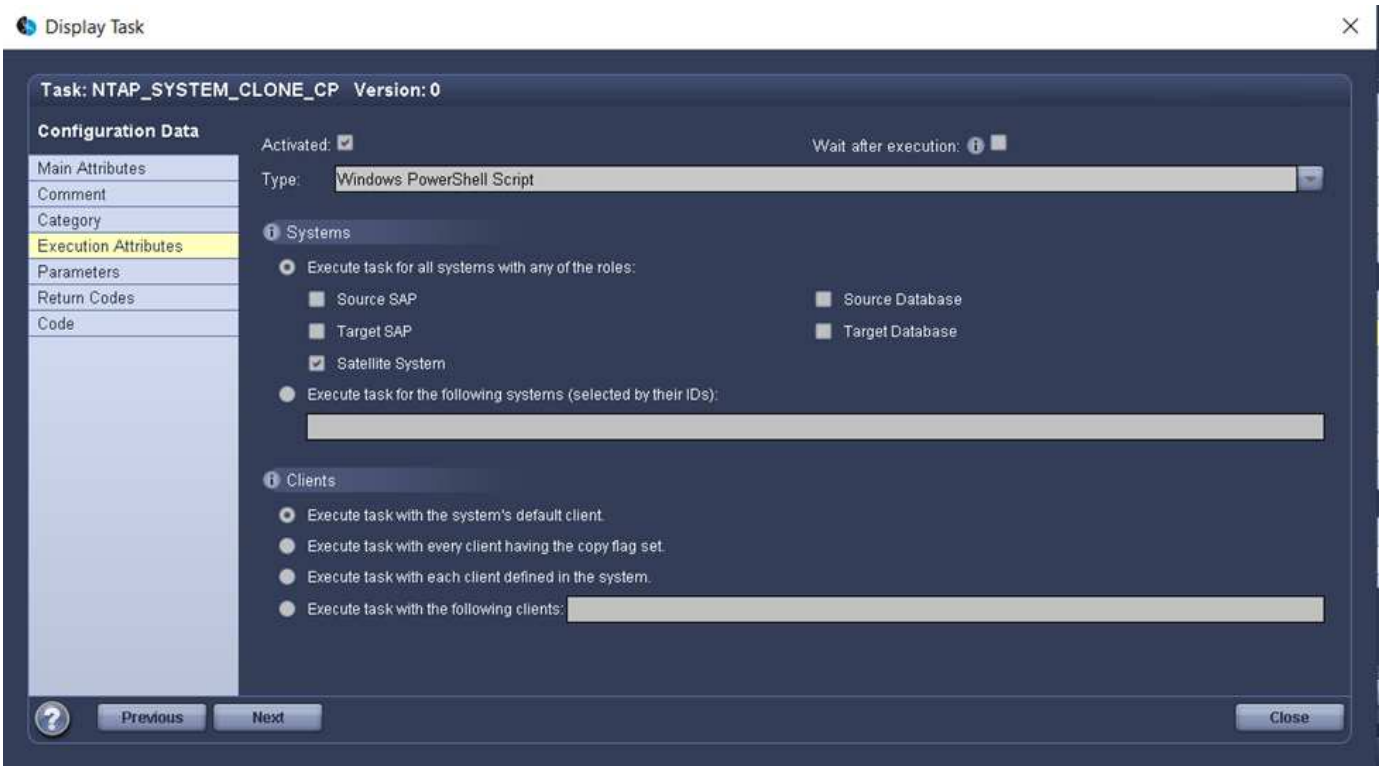
Central communication host

For integration between LSC and SnapCenter using a central communication host, the only adjustments that must be made are performed in the copy phase. The Snapshot copy and the clone are created using the SnapCenter agent on the central communication host. Therefore, all details about the newly created volumes are only available on the central communication host and not on the target database server. However, these details are needed on the target database server to mount the clone volume and to carry out the recovery. This is the reason why two additional tasks are needed in the copy phase. One task is executed on the central communication host and one task is executed on the target database server. These two tasks are shown in the image below.

- **NTAP_SYSTEM_CLONE_CP.** This task creates the Snapshot copy and the clone using a PowerShell script that executes the necessary SnapCenter functions on the central communication host. This task therefore runs on the LSC satellite, which in our instance is the LSC master that runs on Windows. This script collects all details about the clone and the newly created volumes and hands it over to the second task `NTAP_MNT_RECOVER_CP`, which runs on the LSC worker that runs on the target database server.
- **NTAP_MNT_RECOVER_CP.** This task stops the target SAP system and the SAP HANA database, unmounts the old volumes, and then mounts the newly created storage clone volumes based on the parameters that were passed through from the previous task `NTAP_SYSTEM_CLONE_CP`. The target SAP HANA database is then restored and recovered.

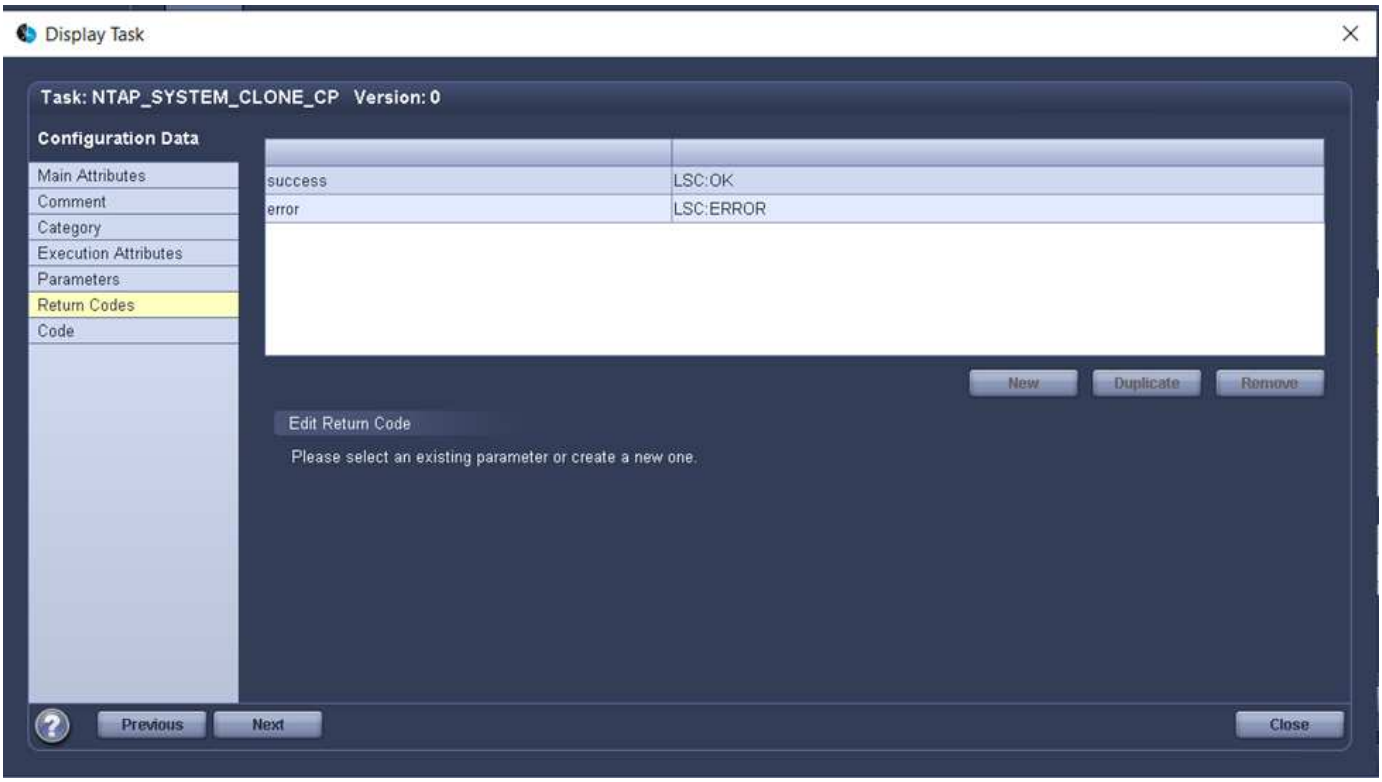
	copy	Copy Phase		phase
	copy 1	NTAP_SYSTEM_CLONE	NetApp SnapShot and Clone	psh
	copy 2	NTAP_SYSTEM_CLONE_CP	NetApp SnapShot and Clone	psh
	copy 3	NTAP_MNT_RECOVER_CP	Mount Volume and Recover HANA Database	cmd
	copy 4	LPDBBCKP	Backup Source DB in Filesystem	lsh
	copy 5	LPDBCPYFLS	Copy DB Backup Files From Source to Target System	lsh
	copy 6	LTDBRESTORE	Restore DB Files	lsh
	copy 7	LTDBRESTORE_TENANT	Restore DB Files for Tenant Database	lsh
	post	Post Phase		phase

The following image highlights the configuration of the task `NTAP_SYSTEM_CLONE_CP`. This is the Windows PowerShell script that is executed on the satellite system. In this instance, the satellite system is the SnapCenter Server with the installed LSC master.

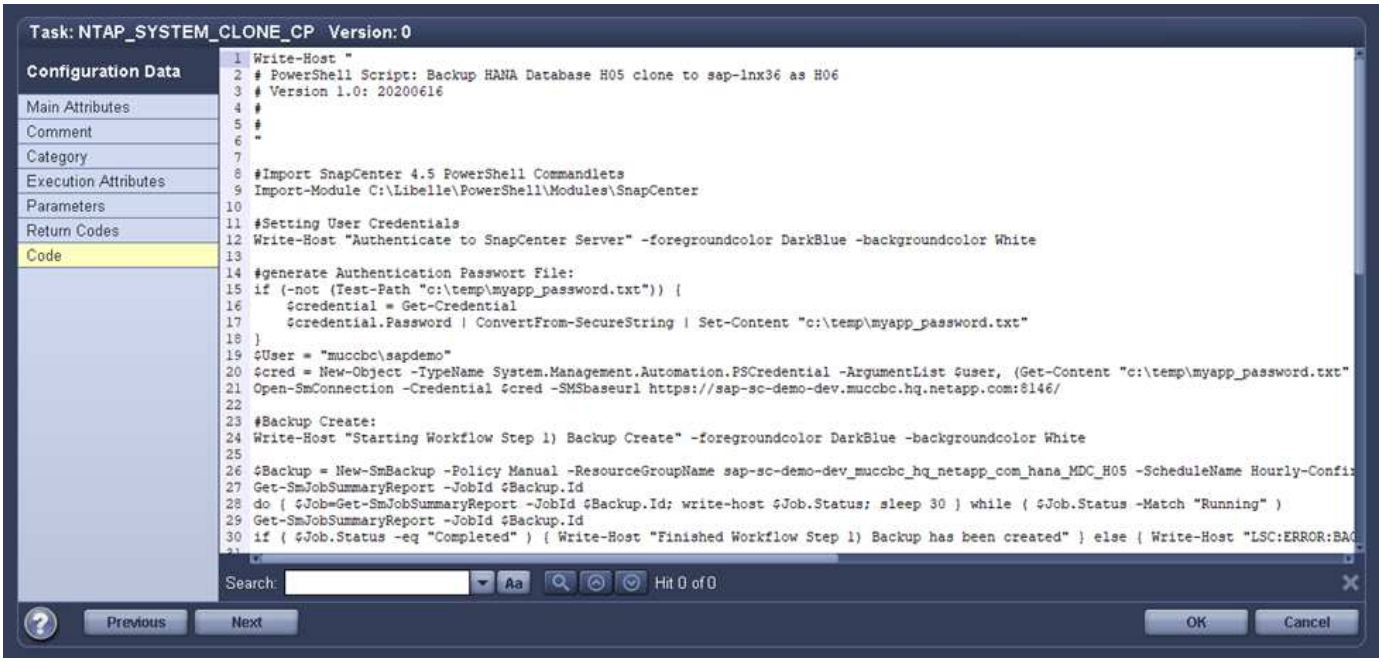


Because LSC must be aware of whether the Snapshot copy and cloning operation was successful, you must define at least two return code types: one return code for a successful execution of the script and the other for a failed execution of the script, as shown in the image below.

- `LSC:OK` must be written from the script to standard out if the execution was successful.
- `LSC:ERROR` must be written from the script to standard out if the execution failed.



The following image shows part of the PowerShell script that must run to execute a Snapshot copy and a clone using the SnapCenter agent on the central communication host. The script is not meant to be complete. Rather, the script is used to show how integration between LSC and SnapCenter can look and how easy it is to set it up.

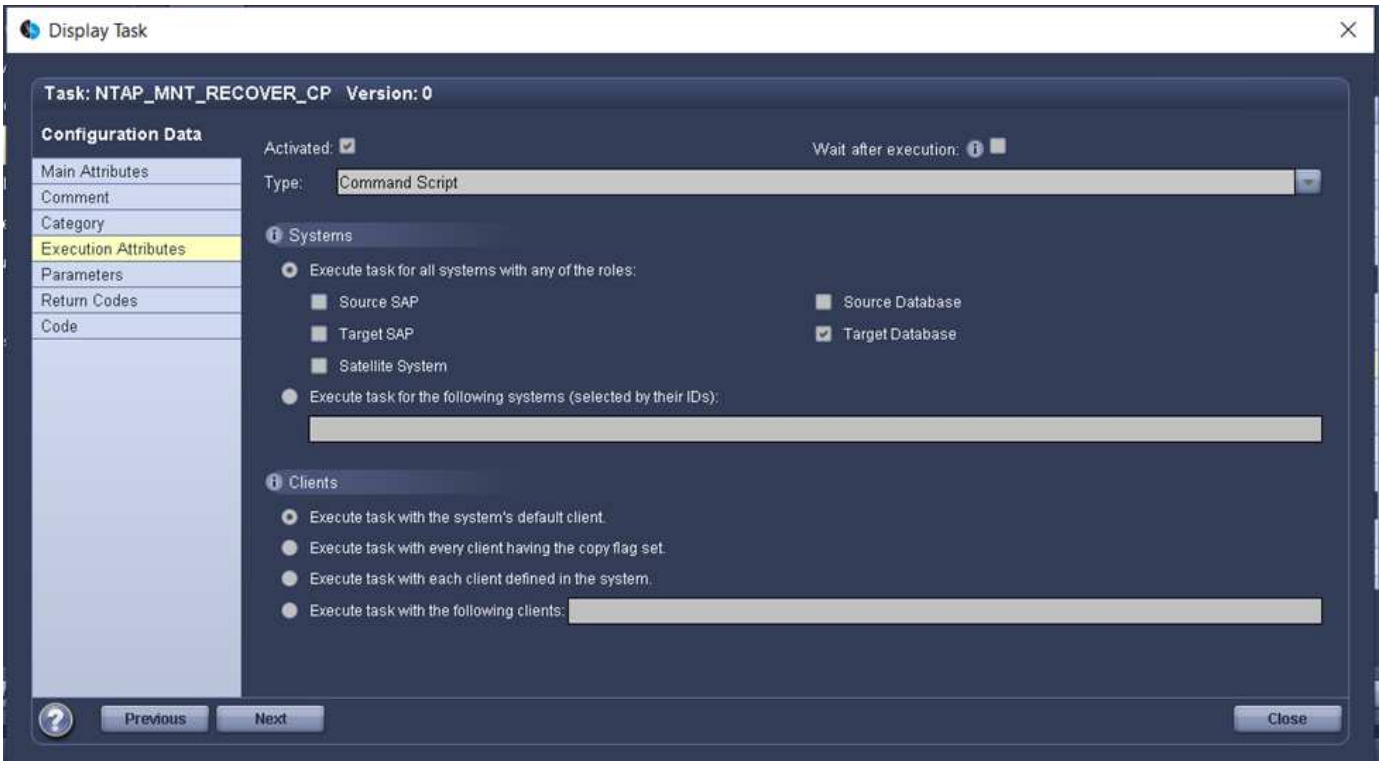


As previously mentioned, you must hand over the name of the clone volume to the next task `NTAP_MNT_RECOVER_CP` to mount the clone volume on the target server. The name of the clone volume, also known as the junction path, is stored in the variable `$JunctionPath`. The handover to a subsequent LSC task is achieved through a custom LSC variable.

```
echo $JunctionPath > $_task(current, custompath1)_$
```

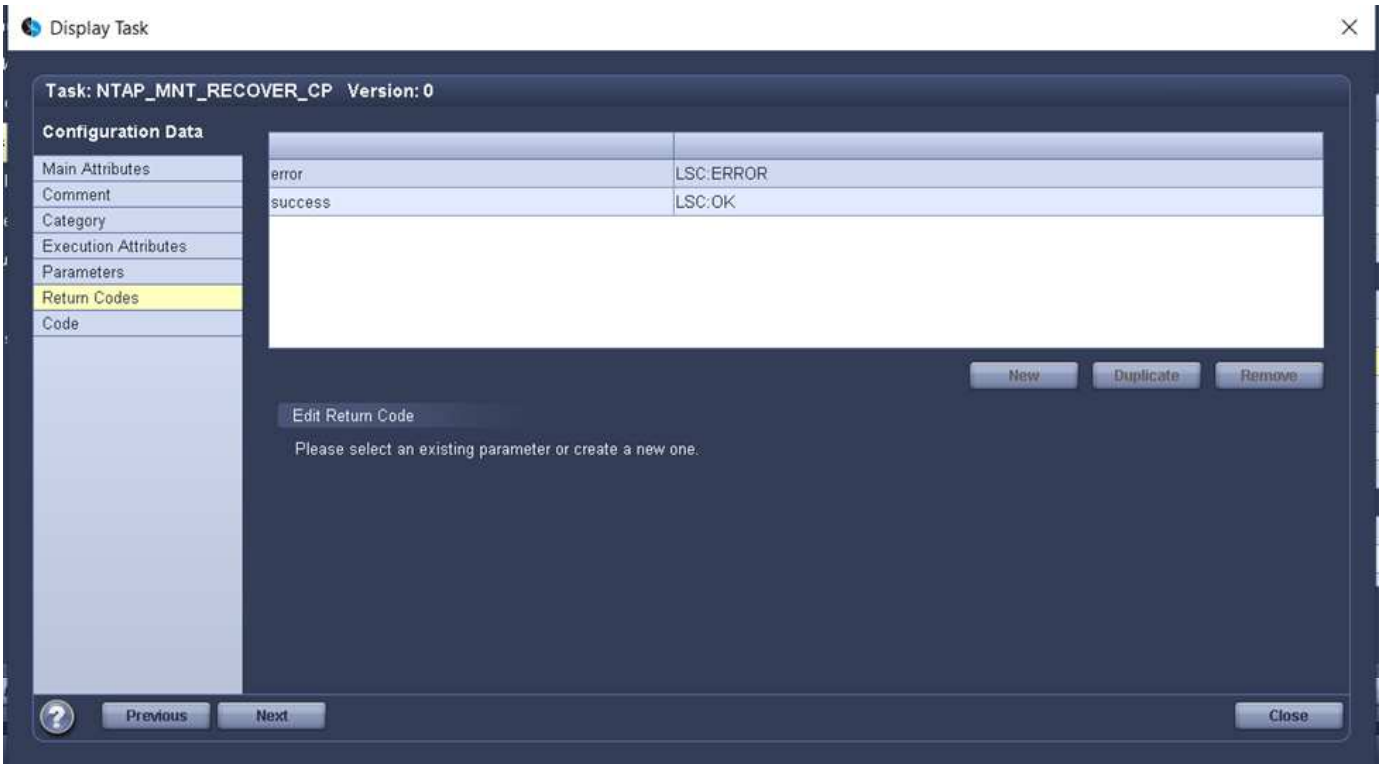
Because the script is executed on the LSC master (which is also a satellite system), the LSC master on the SnapCenter Server must run as a Windows user that has appropriate permissions to execute the backup and cloning operations in SnapCenter. To verify whether it has the appropriate permissions, the user should be able execute a Snapshot copy and clone in the SnapCenter GUI.

The following figure highlights the configuration of the task `NTAP_MNT_RECOVER_CP`. Because we want to execute a Linux Shell script, this is a command script executed on the target database system.



Because LSC must be made aware of mounting the clone volumes and whether restoring and recovering the target database was successful, we must define at least two return code types. One code is for a successful execution of the script, and one is for a failed execution of the script, as is shown in the following figure.

- `LSC:OK` must be written from the script to standard out if the execution was successful.
- `LSC:ERROR` must be written from the script to standard out if the execution failed.



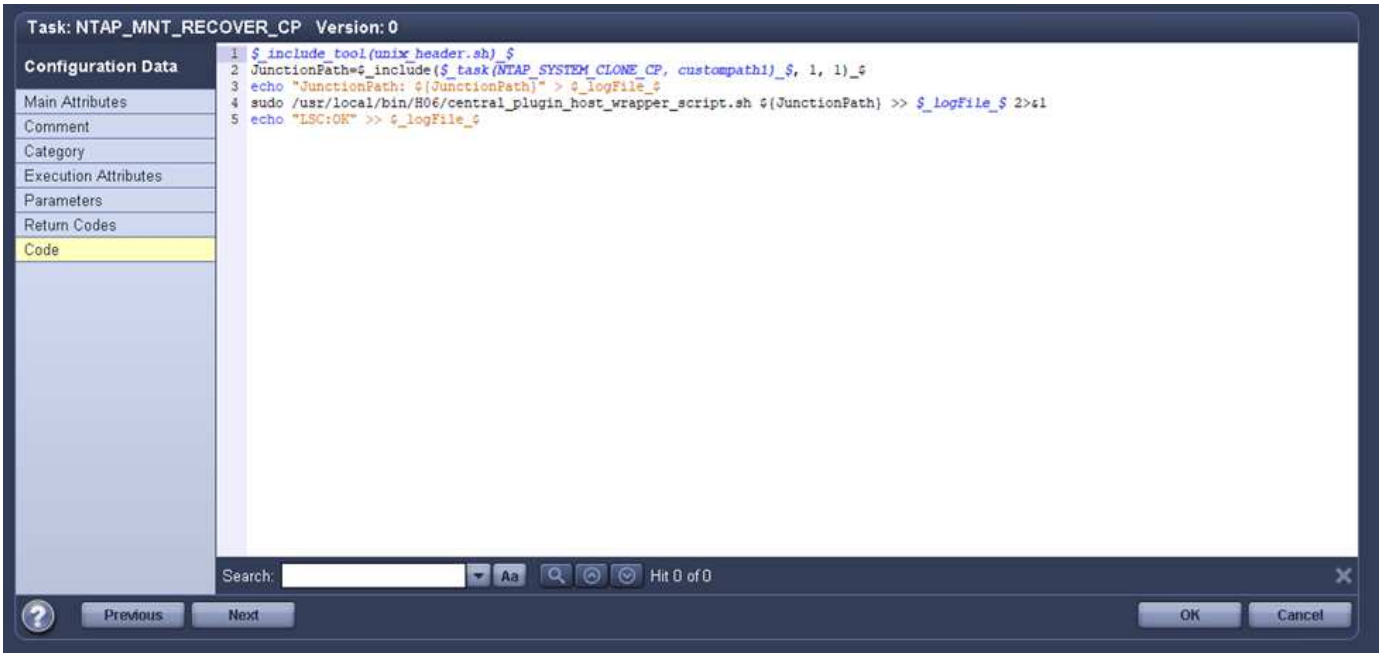
The following figure shows part of the Linux Shell script used to stop the target database, unmount the old

volume, mount the clone volume, and restore and recover the target database. In the previous task, the junction path was written into an LSC variable. The following command reads this LSC variable and stores the value in the `$JunctionPath` variable of the Linux Shell script.

```
JunctionPath=$_include($_task(NTAP_SYSTEM_CLONE_CP, custompath1)_$, 1,
1)_$
```

The LSC worker on the target system runs as `<sidaadm>`, but mount commands must be run as the root user. This is why you must create the `central_plugin_host_wrapper_script.sh`. The script `central_plugin_host_wrapper_script.sh` is called from the task `NTAP_MNT_RECOVERY_CP` using the `sudo` command. Using the `sudo` command, the script runs with UID 0 and we are able to carry out all subsequent steps, such as unmounting the old volumes, mounting the clone volumes, and restoring and recovering the target database. To enable script execution using `sudo`, the following line must be added in `/etc/sudoers`:

```
hn6adm ALL=(root)
NOPASSWD:/usr/local/bin/H06/central_plugin_host_wrapper_script.sh
```

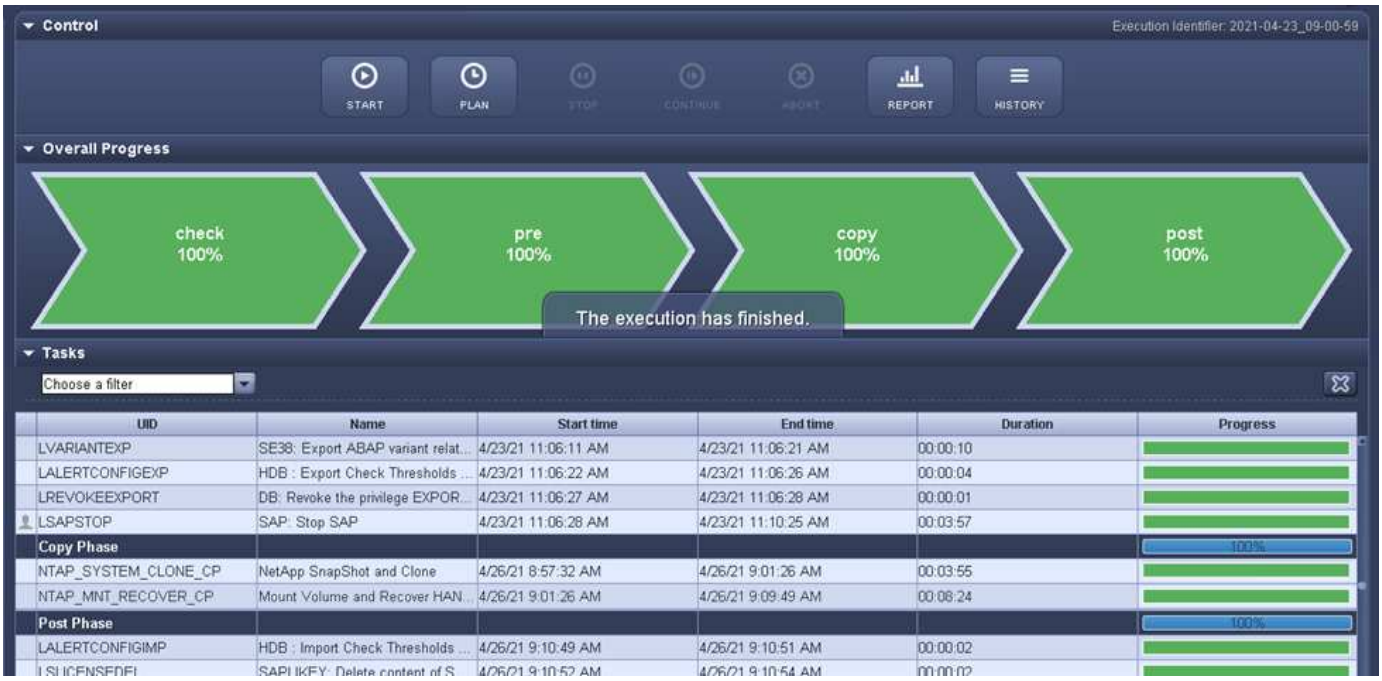


SAP HANA system refresh operation

Now that all necessary integration tasks between LSC and NetApp SnapCenter have been carried out, starting a fully automated SAP system refresh is a one-click task.

The following figure shows the task `NTAP``SYSTEM``CLONE` in a standard installation. As you can see, creating a Snapshot copy and a clone, mounting the clone volume on the target database server, and restoring and recovering the target database took approximately 14 minutes. Remarkably, with Snapshot and NetApp FlexClone technology, the duration of this task remains nearly the same, independent of the size of the source database.

The following figure shows the two tasks `NTAP_SYSTEM_CLONE_CP` and `NTAP_MNT_RECOVERY_CP` when using a central communication host. As you can see, creating a Snapshot copy, a clone, mounting the clone volume on the target database server, and restoring and recovering the target database took approximately 12 minutes. This is more or less the same time needed to carry out these steps when using a standard installation. Again, Snapshot and NetApp FlexClone technology enables the consistent, rapid completion of these tasks, independent of the size of the source database.



SAP HANA system refresh with LSC, AzAcSnap, and Azure NetApp Files

Using Azure NetApp Files for SAP HANA, Oracle, and DB2 on Azure provides customers with the advanced data management and data protection features of NetApp ONTAP with the native Microsoft Azure NetApp Files service. AzAcSnap is the foundation for very fast SAP system refresh operations to create application-consistent NetApp Snapshot copies

of SAP HANA and Oracle systems (DB2 is not currently supported by AzAcSnap).

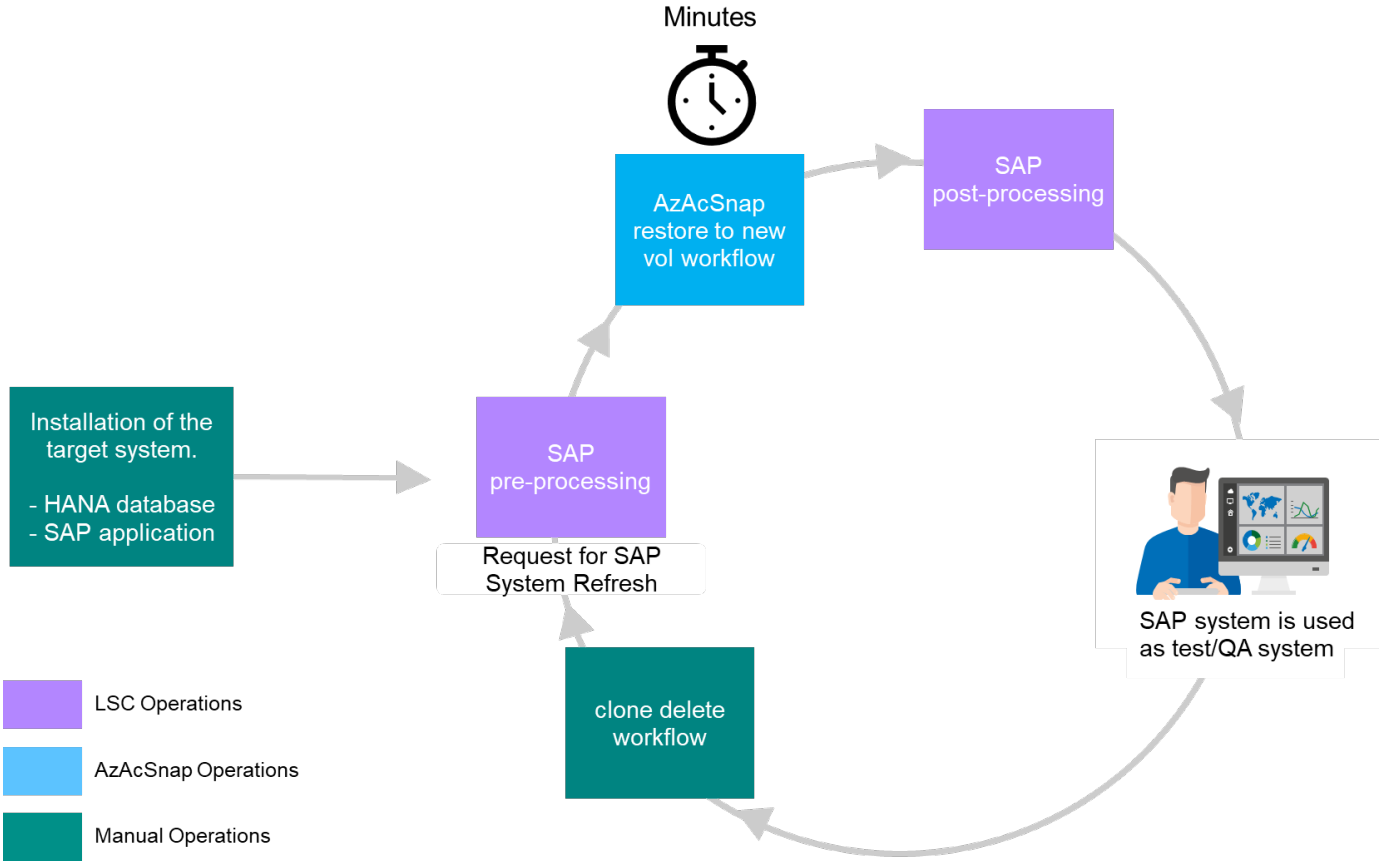
Snapshot copy backups, which are created either on-demand or on a regular basis as part of the backup strategy, can then be efficiently cloned to new volumes and used to quickly refresh target systems. AzAcSnap provides the workflows necessary to create backups and clone them to new volumes, while Libelle SystemCopy performs the pre- and post-processing steps necessary for a full end-to-end system refresh.

In this chapter, we describe an automated SAP system refresh using AzAcSnap and Libelle SystemCopy using SAP HANA as the underlying database. Because AzAcSnap is also available for Oracle, the same procedure can also be implemented using AzAcSnap for Oracle. Other databases might be supported by AzAcSnap in the future, which would then enable system copy operations for those databases with LSC and AzAcSnap.

The following figure shows a typical high-level workflow of an SAP system refresh lifecycle with AzAcSnap and LSC:

- A one-time, initial installation and preparation of the target system.
- SAP preprocessing operations performed by LSC.
- Restoring (or cloning) an existing Snapshot copy of the source system to the target system performed by AzAcSnap.
- SAP post-processing operations performed by LSC.

The system can then be used as a test or QA system. When a new system refresh is requested, the workflow restarts with step 2. Any remaining cloned volumes must be deleted manually.

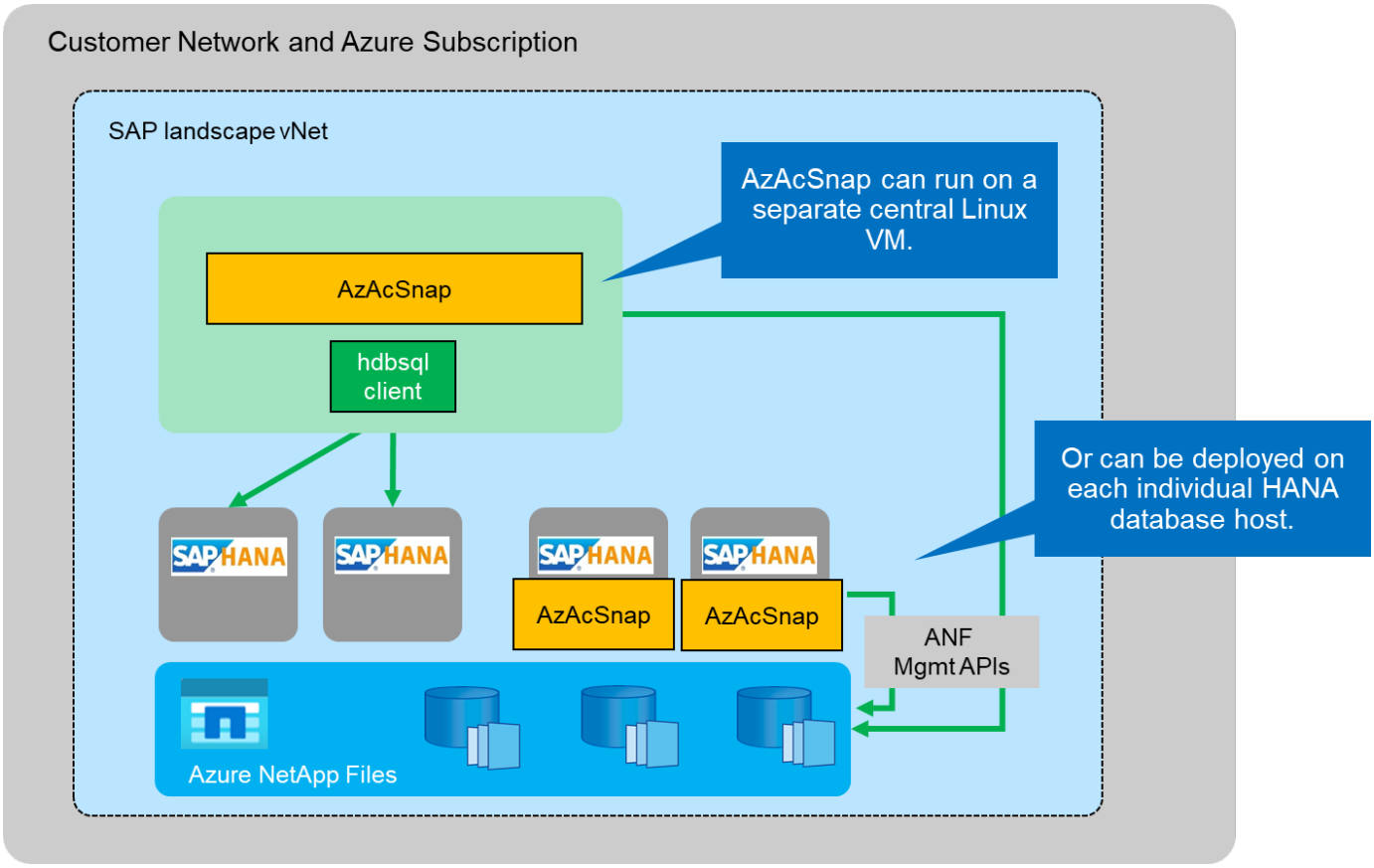


Prerequisites and limitations

The following prerequisites must be fulfilled.

AzAcSnap installed and configured for the source database

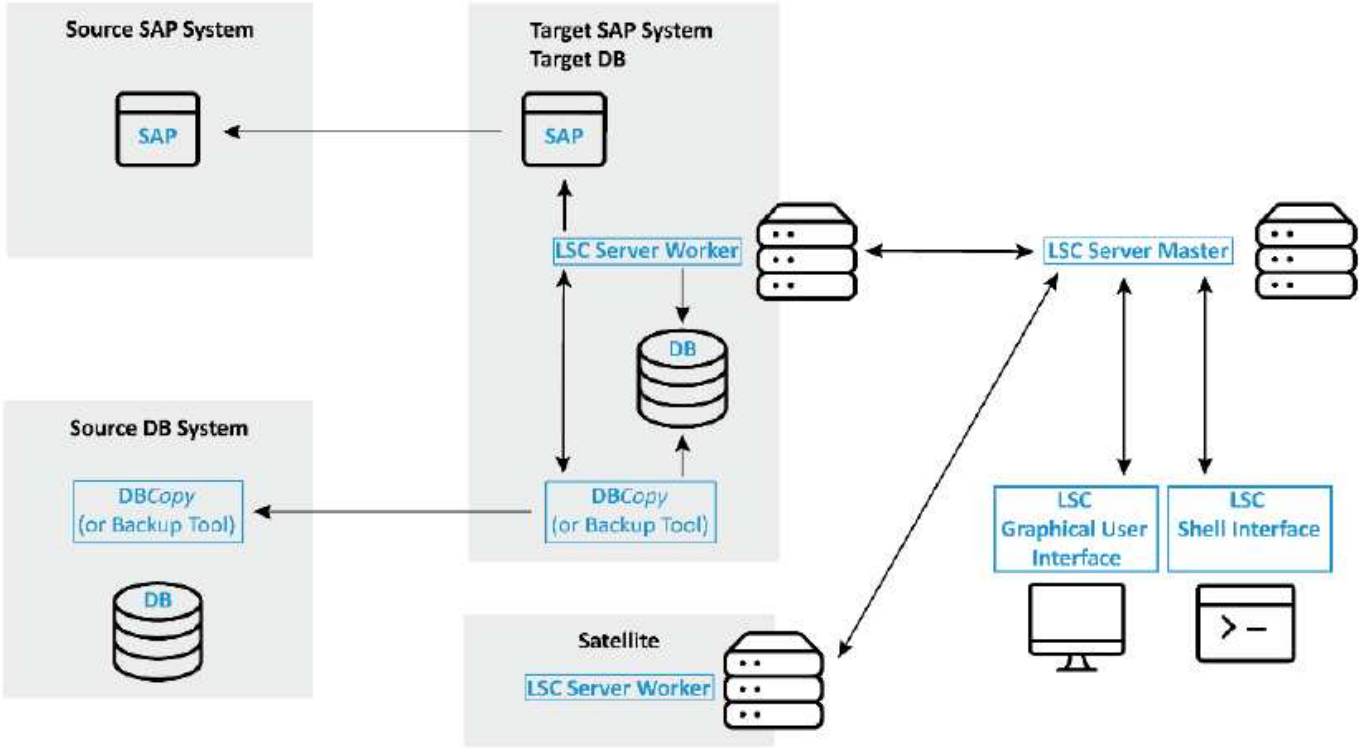
In general, there are two deployments options for AzAcSnap, as is shown in the following picture.



AzAcSnap can be installed and run on a central Linux VM for which all DB configuration files are stored centrally and AzAcSnap has access to all databases (through the hdbsql client) and the configured HANA userstore keys for all these databases. With a decentralized deployment, AzAcSnap is installed individually on each database host where typically only the DB configuration for the local database is stored. Both deployment options are supported for LSC integration. However, we followed a hybrid approach in the lab setup for this document. AzAcSnap was installed on a central NFS share along with all DB configuration files. This central installation share was mounted on all VMs under `/mnt/software/AZACSNAP/snapshot-tool`. The execution of the tool was then performed locally on the DB VMs.

Libelle SystemCopy installed and configured for source and target SAP system

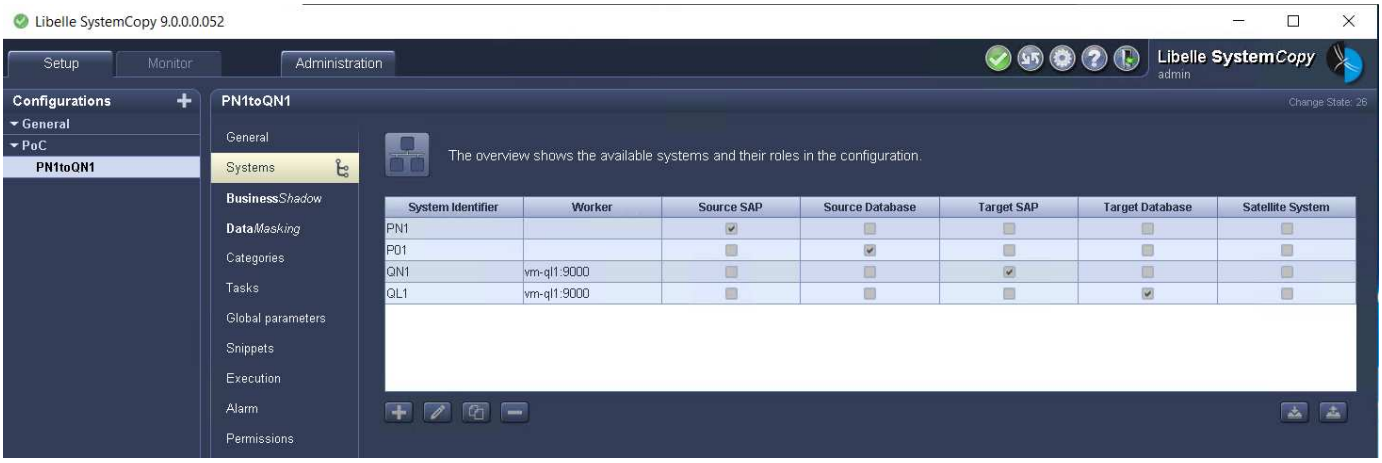
Libelle SystemCopy deployments consist of the following components:



- **LSC Master.** As the name suggests, this is the master component that controls the automatic workflow of a Libelle-based system copy.
- **LSC Worker.** An LSC worker usually runs on the target SAP system and executes the scripts required for the automated system copy.
- **LSC Satellite.** An LSC satellite runs on a third-party system on which further scripts must be executed. The LSC master can also fulfill the role of an LSC satellite system.

The Libelle SystemCopy (LSC) GUI must be installed on a suitable VM. In this lab setup, the LSC GUI was installed on a separate Windows VM, but it can also run on the DB host together with the LSC worker. The LSC worker must be installed at least on the VM of the target DB. Depending on your chosen AzAcSnap deployment option, additional LSC worker installations might be required. You must have an LSC worker installation on the VM where AzAcSnap is executed.

After LSC is installed, the basic configuration for the source and the target database must be performed according to the LSC guidelines. The following images shows the configuration of the lab environment for this document. See the next section for details about the source and the target SAP systems and databases.



You should also configure a suitable standard task list for the SAP systems. For more details about the installation and configuration of LSC, consult the LSC user manual that is part of the LSC installation package.

Known limitations

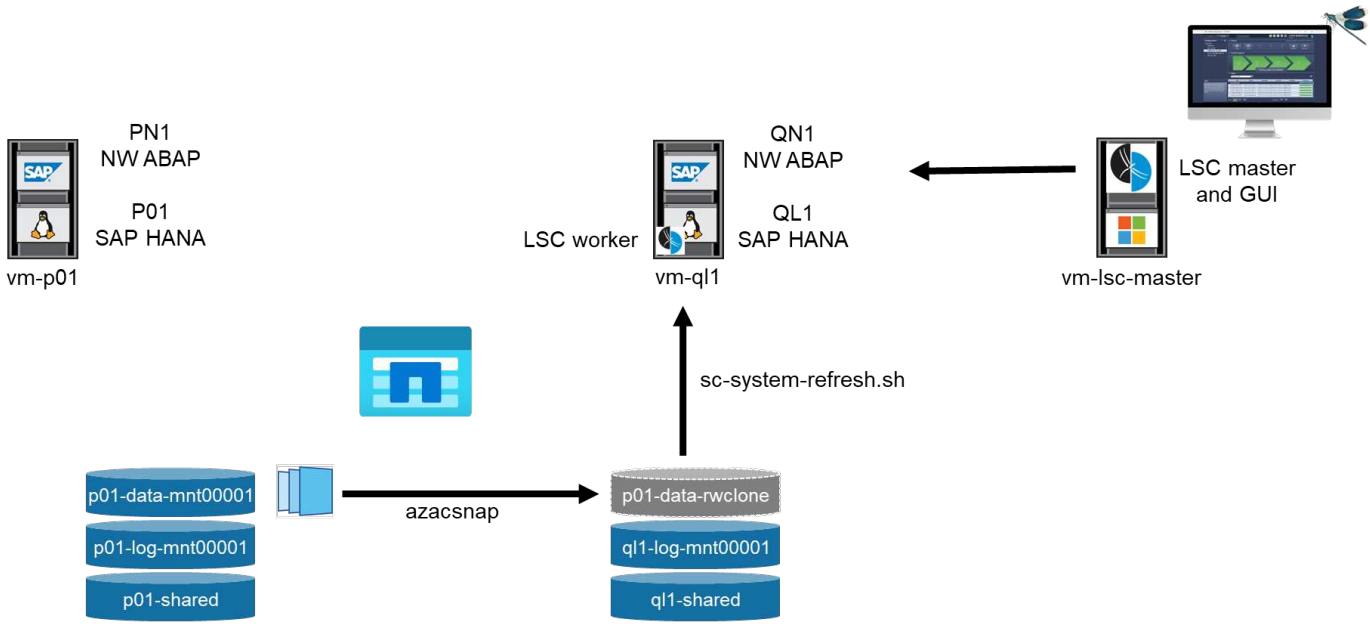
The AzAcSnap and LSC integration described here only works for SAP HANA single-host databases. SAP HANA multiple-host (or scale-out) deployments can also be supported, but such deployments require a few adjustments or enhancements to the LSC custom tasks for the copy phase and the underlaying scripts. Such enhancements are not covered in this document.

SAP system refresh integration always uses the latest successful Snapshot copy of the source system to perform the refresh of the target system. If you would like to use other older Snapshot copies, the corresponding logic in the ZAZACSNAPRESTORE custom task must be adjusted. This process is out of scope for this document.

Lab setup

The lab setup consists of a source SAP system and a target SAP system, both running on SAP HANA single-host databases.

The following picture shows the lab setup.



It contains the following systems, software versions, and Azure NetApp Files volumes:

- **P01.** SAP HANA 2.0 SP5 database. Source database, single host, single user tenant.
- **PN1.** SAP NetWeaver ABAP 7.51. Source SAP system.
- **vm-p01.** SLES 15 SP2 with AzAcSnap installed. Source VM hosting P01 and PN1.
- **QL1.** SAP HANA 2.0 SP5 database. System refresh target database, single host, single-user tenant.
- **QN1.** SAP NetWeaver ABAP 7.51. System refresh target SAP system.
- **vm-ql1.** SLES 15 SP2 with LSC worker installed. Target VM hosting QL1 and QN1.
- LSC master version 9.0.0.0.052.
- **vm- lsc-master.** Windows Server 2016. Hosts LSC master and LSC GUI.

- Azure NetApp Files volumes for data, log, and shared for P01 and QL1 mounted on the dedicated DB hosts.
- Central Azure NetApp Files volume for scripts, AzAcSnap installation, and configuration files mounted on all VMs.

Initial one-time preparation steps

Before the first SAP system refresh can be executed, you must integrate Azure NetApp Files Snapshot copy-and-cloning-based storage operations executed by AzAcSnap. You must also execute an auxiliary script for starting and stopping the database and mounting or unmounting the Azure NetApp Files volumes. All required tasks are performed as custom tasks in LSC as part of the copy phase. The following picture shows the custom tasks in the LSC task list.

Phase	UID	Name	Type
pre 77	LREVOKEEXPORT	DB: Revoke the privilege EXPO...	cmd
pre 78	*LJAVACONFEXP*	*JAVA: Backup java config files...*	*cmd*
pre 79	*LSTOPSLTJOBS*	*LTRC: Stop all replication jobs ...*	*lsh*
pre 80	*LSAPSTOP*	*SAP: Stop SAP*	*intv*
pre 81	LSTOPSAPSYSTEM	Stops all SAP instances (appli...	lsh
copy	**Copy Phase**		**phase**
copy 1	ZSCCOPYSHUTDOWN	Shutdown HANA DB	cmd
copy 2	ZSCCOPYUMOUNT	Unmount data volumes	cmd
copy 3	ZAZACSNAPRESTORE	Restore snapshot backup of so...	cmd
copy 4	ZSCCOPYMOUNT	Mount data volumes	cmd
copy 5	ZSCCOPYRECOVER	Recover target DB based on sn...	cmd
post	**Post Phase**		**phase**
post 1	*LCHNGHDBPWD*	*HDB : Restore the password fo...*	*cmd*
post 2	LHDBLICIMP	HANA DB License Import	lsh
post 3	LALERTCONFIGIMP	HDB : Import Check Threshold...	lsh

All five copy tasks are described here in more detail. In some of these tasks, a sample script `sc-system-refresh.sh` is used to further automate the required SAP HANA database recovery operation and the mount and unmount of the data volumes. The script uses an `LSC: success` message in the system output to indicate a successful execution to LSC. Details about custom tasks and available parameters can be found in the LSC user manual and the LSC developer guide. All tasks in this lab environment are executed on the target DB VM.

> The sample script is provided as is and is not supported by NetApp. You can request the script by email to ng-sapcc@netapp.com.

Sc-system-refresh.sh configuration file

As mentioned before, an auxiliary script is used to start and stop the database, to mount and unmount the Azure NetApp Files volumes, and to recover the SAP HANA database from a Snapshot copy. The script `sc-system-refresh.sh` is stored on the central NFS share. The script requires a configuration file for each target database that must be stored in the same folder as the script itself. The configuration file must have the following name: `sc-system-refresh-<target DB SID>.cfg` (for example `sc-system-refresh-QL1.cfg` in this lab environment). The configuration file used here uses a fixed/hard-coded source DB SID. With a few changes, the script and the config file can be enhanced to take the source DB SID as an input parameter.

The following parameters must be adjusted according to the specific environment:

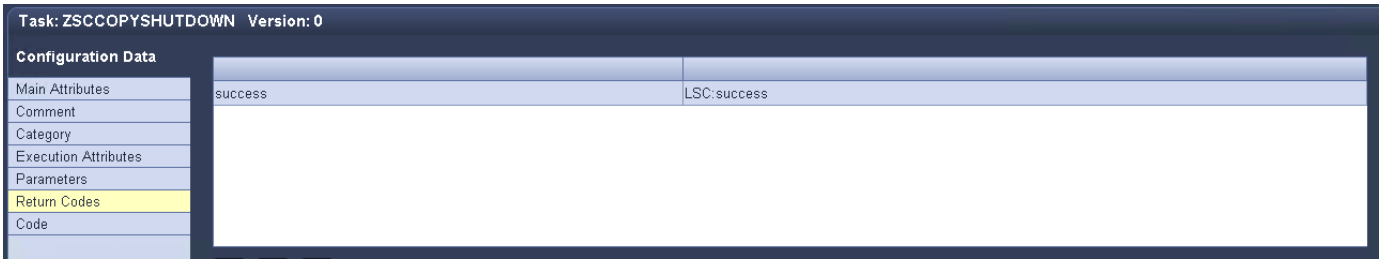
```
# hdbuserstore key, which should be used to connect to the target database
KEY="QL1SYSTEM"
# single container or MDC
export P01_HANA_DATABASE_TYPE=MULTIPLE_CONTAINERS
# source tenant names { TENANT_SID [, TENANT_SID]* }
export P01_TENANT_DATABASE_NAMES=P01
# cloned vol mount path
export CLONED_VOLUMES_MOUNT_PATH=`tail -2
/mnt/software/AZACSNAP/snapshot_tool/logs/azacsnap-restore-azacsnap-
P01.log | grep -oe "[0-9]*\.[0-9]*\.[0-9]*\.[0-9]*:/.* "`
```

ZSCCOPYSHUTDOWN

This task stops the target SAP HANA database. The Code section of this task contains the following text:

```
$_include_tool(unix_header.sh)_$
sudo /mnt/software/scripts/sc-system-refresh/sc-system-refresh.sh shutdown
$_system(target_db, id)_$ > $_logfile_$
```

The script `sc-system-refresh.sh` takes two parameters, the `shutdown` command and the DB SID, to stop the SAP HANA database using sapcontrol. The system output is redirected to the standard LSC logfile. As mentioned before, an `LSC: success` message is used to indicate successful execution.



ZSCCOPYUMOUNT

This task unmounts the old Azure NetApp Files data volume from the target DB operating system (OS). The code section of this task contains the following text:

```
$_include_tool(unix_header.sh)_$
sudo /mnt/software/scripts/sc-system-refresh/sc-system-refresh.sh umount
$_system(target_db, id)_$ > $_logfile_$
```

The same scripts as in the previous task is used. The two parameters passed are the `umount` command and the DB SID.

ZAZACSNAPRESTORE

This task runs AzAcSnap to clone the latest successful Snapshot copy of the source database to a new

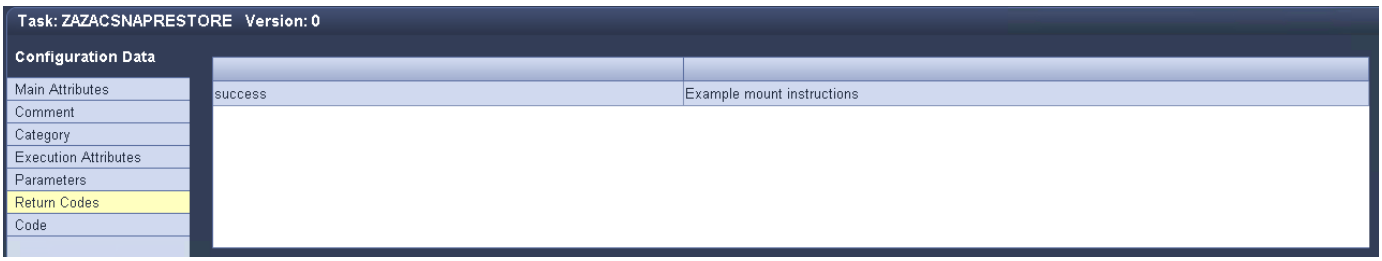
volume for the target database. This operation is equivalent to a redirected restore of backup in traditional backup environments. However, the Snapshot copy and cloning functionality enables you to perform this task within seconds even for the largest databases, whereas, with traditional backups, this task could easily take several hours. The code section of this task contains the following text:

```
$_include_tool(unix_header.sh)_$
sudo /mnt/software/AZACSNAP/snapshot_tool/azacsnap -c restore --restore
snaptovol --hanasid $_system(source_db, id)_$
--configfile=/mnt/software/AZACSNAP/snapshot_tool/azacsnap
-$_system(source_db, id)_$.json > $_logfile_$
```

Full documentation for the AzAcSnap command line options for the `restore` command can be found in the Azure documentation here: Restore using Azure Application Consistent Snapshot tool. The call assumes that the json DB configuration file for the source DB can be found on the central NFS share with the following naming convention: `azacsnap-<source DB SID>. json`, (for example, `azacsnap-P01.json` in this lab environment).

> Because the output of the AzAcSnap command cannot be changed, the default `LSC: success` message cannot be used for this task. Therefore, the string `Example mount instructions` from the AzAcSnap output is used as a successful return code. In the 5.0 GA version of AzAcSnap, this output is only generated if the cloning process was successful.

The following figure shows the AzAcSnap restore to new volume success message.



ZSCCOPYMOUNT

This task mounts the new Azure NetApp Files data volume on the OS of the target DB. The code section of this task contains the following text:

```
$_include_tool(unix_header.sh)_$
sudo /mnt/software/scripts/sc-system-refresh/sc-system-refresh.sh mount
$_system(target_db, id)_$ > $_logfile_$
```

The sc-system-refresh.sh script is used again, passing the `mount` command and the target DB SID.

ZSCCOPYRECOVER

This task performs an SAP HANA database recovery of the system database and the tenant database based on the restored (cloned) Snapshot copy. The recovery option used here is to specific database backup, such as no additional logs, are applied for forward recovery. Therefore, the recovery time is very short (a few minutes at most). The runtime of this operation is determined by the startup of the SAP HANA database that

happens automatically after the recovery process. To speed up the startup time, the throughput of the Azure NetApp Files data volume can be increased temporarily if needed as described in this Azure documentation: Dynamically increasing or decreasing volume quota. The code section of this task contains the following text:

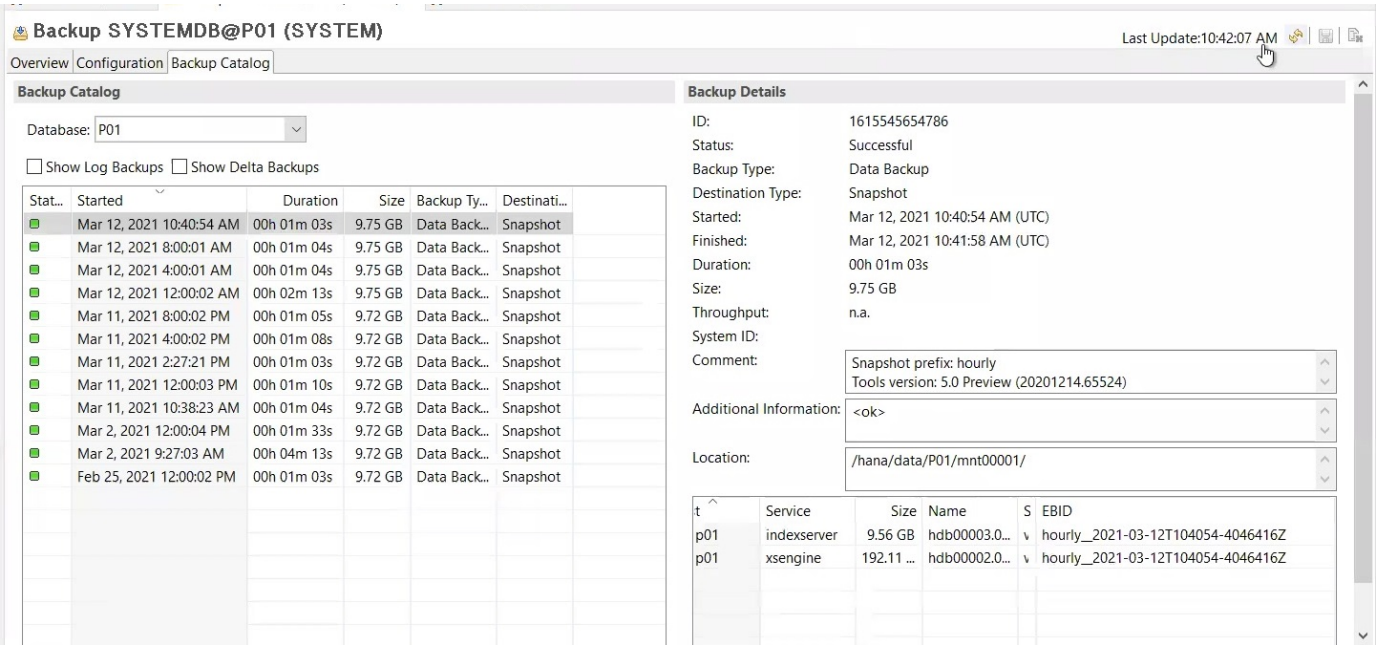
```
$_include_tool(unix_header.sh)_$
sudo /mnt/software/scripts/sc-system-refresh/sc-system-refresh.sh recover
$_system(target_db, id)_$ > $_logfile_$
```

This script is used again with the `recover` command and the target DB SID.

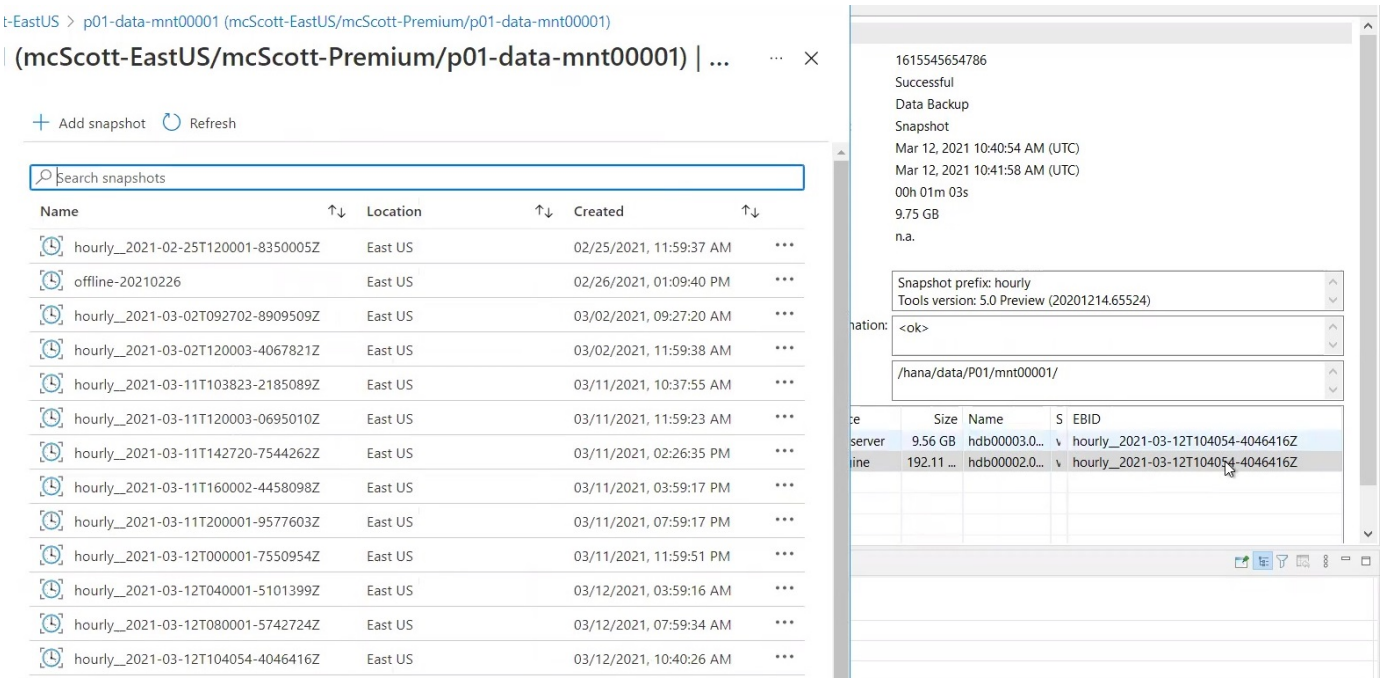
SAP HANA system refresh operation

In this section a sample refresh operation of lab systems shows the main steps of this workflow.

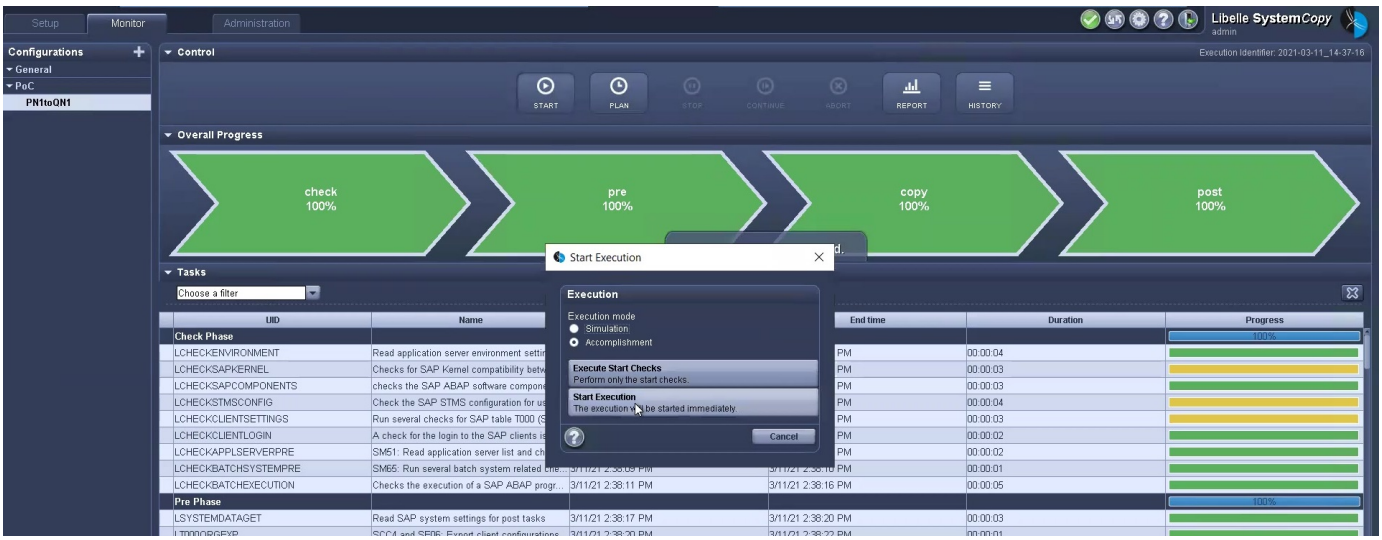
Regular and on-demand Snapshot copies have been created for the P01 source database as listed in the backup catalog.



For the refresh operation, the latest backup from March 12th was used. In the backup details section, the external backup ID (EBID) for this backup is listed. This is the Snapshot copy name of the corresponding Snapshot copy backup on the Azure NetApp Files data volume as shown in the following picture.

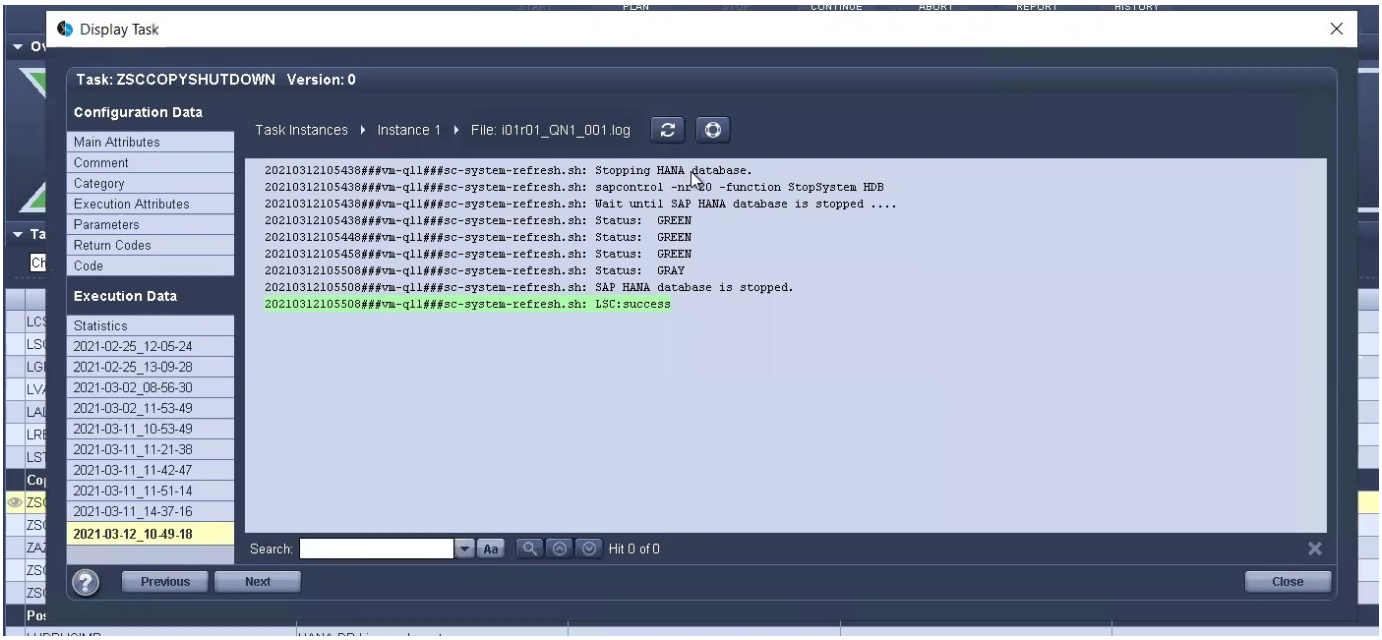


To start the refresh operation, select the correct configuration in the LSC GUI, and then click Start Execution.

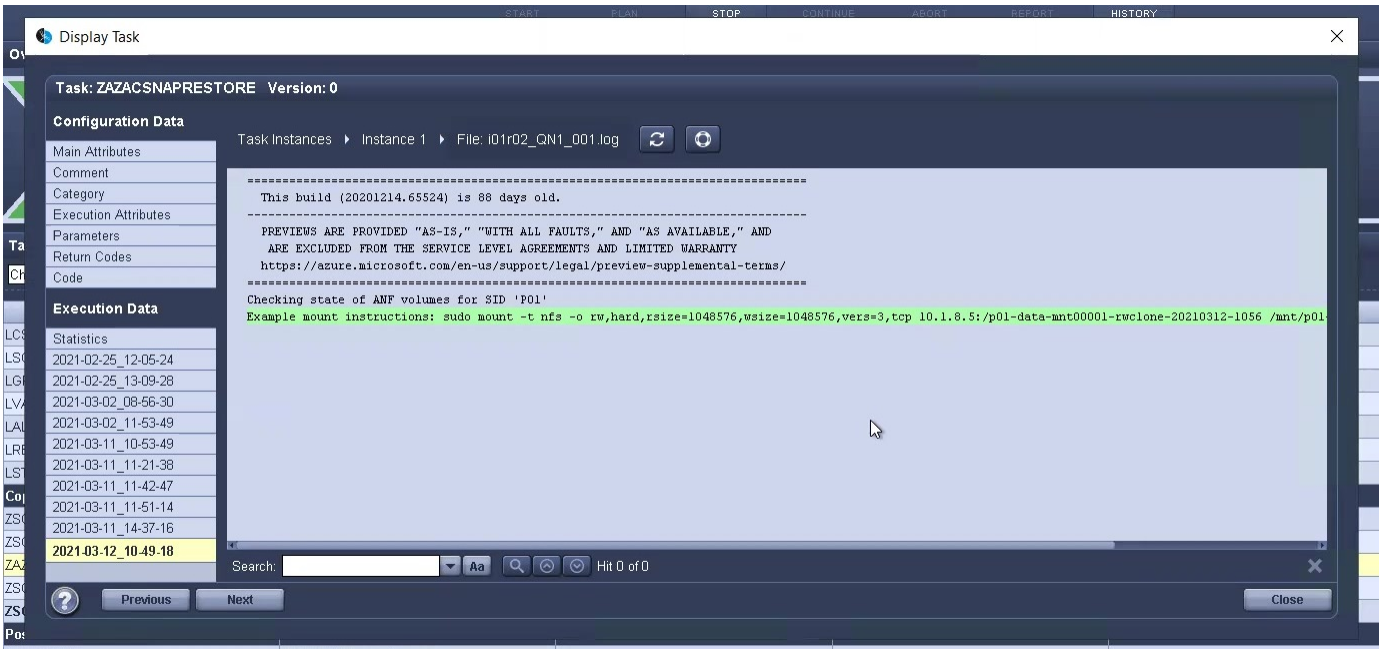


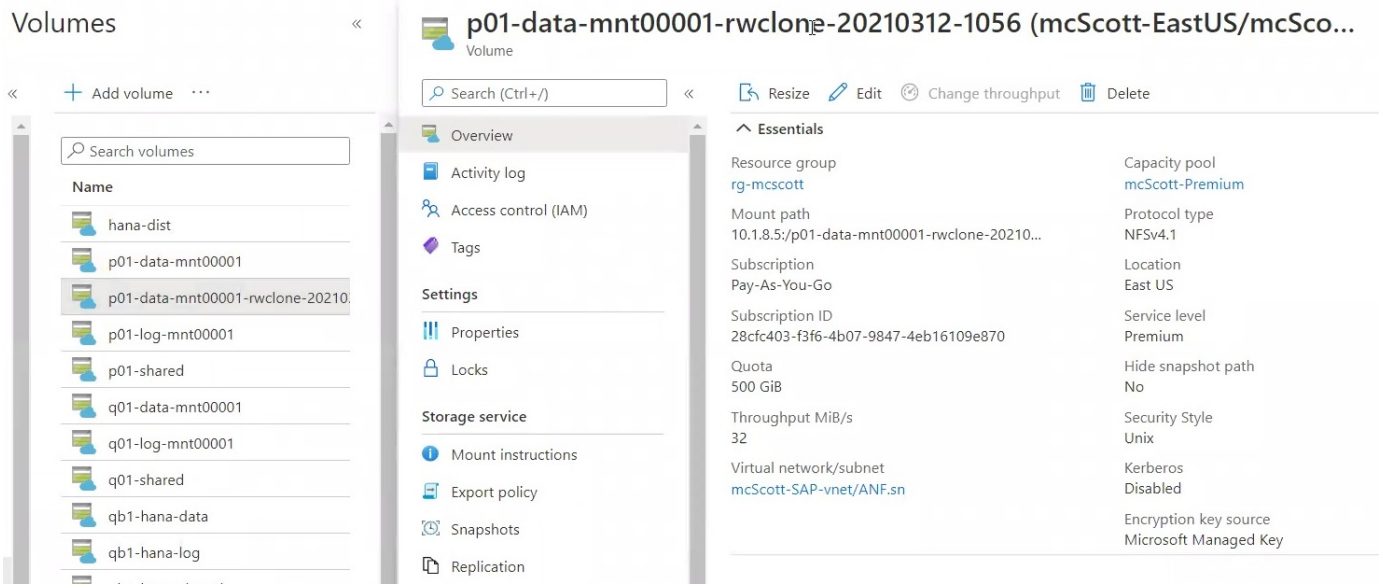
LSC starts to execute the tasks of the Check phase followed by the configured tasks of the Pre phase.

As the last step of the Pre phase, the target SAP system is stopped. In the following Copy phase, the steps described in the previous section are executed. First, the target SAP HANA database is stopped, and the old Azure NetApp Files volume is unmounted from the OS.

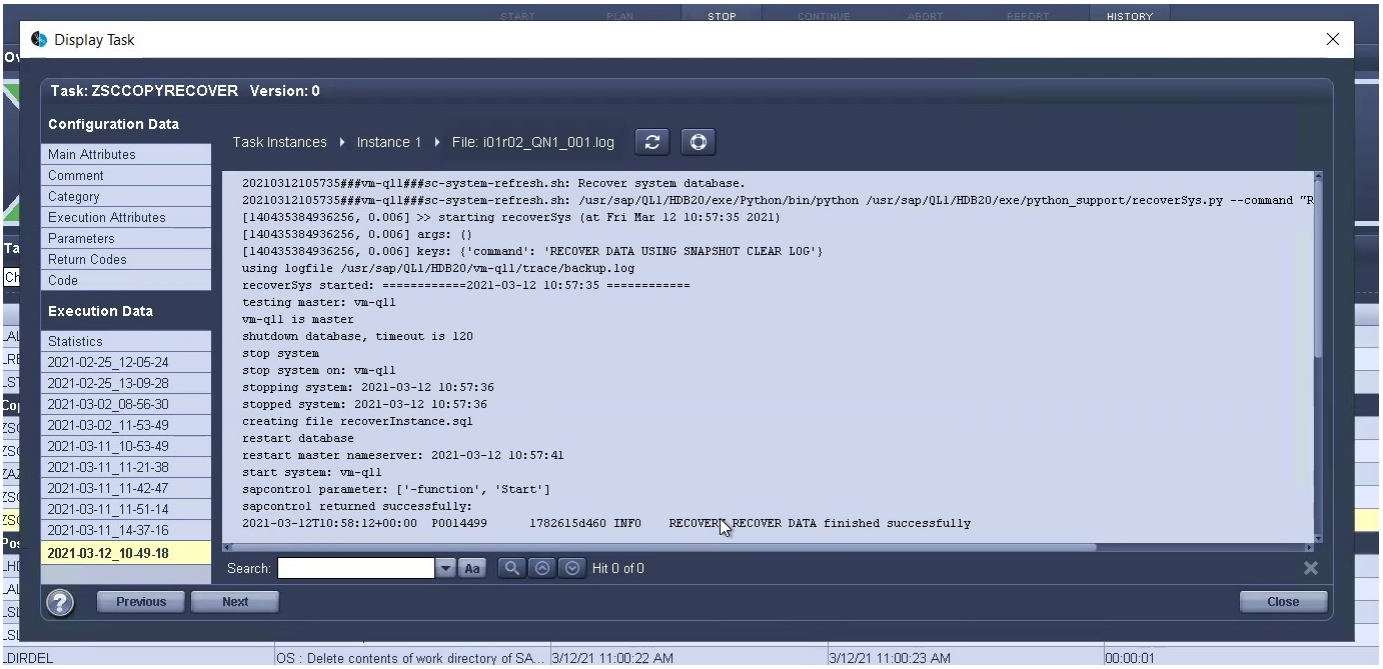


The ZAZACSNAPRESTORE task then creates a new volume as a clone from the existing Snapshot copy of the P01 system. The following two pictures show the logs of the task in the LSC GUI and the cloned Azure NetApp Files volume in the Azure portal.





This new volume is then mounted on the target DB host and the system database and the tenant database are recovered using the containing Snapshot copy. After successful recovery, the SAP HANA database is started automatically. This startup of the SAP HANA database occupies most of the time of the Copy phase. The remaining steps typically finish in a few seconds to a few minutes, regardless of the size of the database. The following picture shows how the system database is recovered using SAP- provided python recovery scripts.



After the Copy phase, LSC continues with all the defined steps of the Post phase. When the System Refresh process finishes completely, the target system is up and running again and fully usable. With this lab system, the total runtime for the SAP system refresh was roughly 25 minutes, of which the Copy phase consumed just under 5 minutes.

Where to find additional information and version history

To learn more about the information that is described in this document, review the following documents and/or websites:

- NetApp Product Documentation

  https://docs.netapp.com

Version history

Version	Date	Document Version History
Version 1.0	April 2022	Initial release.