SAP HANA System Replication on SLES for SAP Applications

11 SP3 www.suse.com

May 19, 2014 | Setup Guide

DRAFT

SAP HANA System Replication on SLES for SAP Applications

List of authors: Yuri Shinkarev (Yury.Shinkarev@realtech.com), Fabian Herschel (Fabian.Herschel@suse.com)

TODO: Marcom: check marking of trademarks

For Novell trademarks, see the Novell Trademark and Service Mark list http://www.novell.com/company/legal/trademarks/tmlist.html . Linux* is a registered trademark of Linus Torvalds. All other third party trademarks are the property of their respective owners. A trademark symbol (®, ™ etc.) denotes a Novell trademark; an asterisk (*) denotes a third party trademark.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither Novell, Inc., SUSE LINUX Products GmbH, the authors, nor the translators shall be held liable for possible errors or the consequences thereof.

Version	Version date	Change(s)	Author
0.1	23.04.2014	Initial version	Yury Shinkarev
0.2	02.05.2014	Introduction, first review	Fabian Herschel
0.3	08.05.2014	HANA SR Setup	Yury Shinkarev
0.4	09.05.2014	Formatting, SLE HA Cluster Setup second review	Fabian Herschel
0.5	14.05.2014	Updating HANA and cluster configuration	Fabian Herschel
0.6	19.05.2014	Updating HANA and Cluster, examples and screen shots	Fabian Herschel

CONTENTS

1 INTRODUCTION

“SAP customers invest in SAP HANA” is the conclusion reached by a recent market study carried out by Pierre Audoin Consultants (PAC). In Germany alone, half of companies expect SAP HANA to become the dominant database platform in the SAP environment. In many cases, the “SAP Business Suite® powered by SAP HANA” scenario is already being discussed in concrete terms.

Naturally, SUSE is also accommodating this development by providing SUSE Linux Enterprise Server for SAP Applications – the recommended and supported operating system for SAP HANA. In close collaboration with SAP and hardware partners, therefore, SUSE will provide two resource agents for customers to ensure the high availability of SAP HANA system replications.

Two Replication Scenarios

The current first phase of the project includes the architecture and development of scale-up scenarios, which will be tested together with SAP in the coming weeks. System replication will help to replicate the database data from one computer to another computer in order to compensate for database failures (single-box replication). This is to be followed by a second project phase involving an extension for scale-out scenarios (multibox replication). With this mode of operation, internal SAP HANA high-availability (HA) mechanisms and the resource agent must work together or be coordinated with each other.

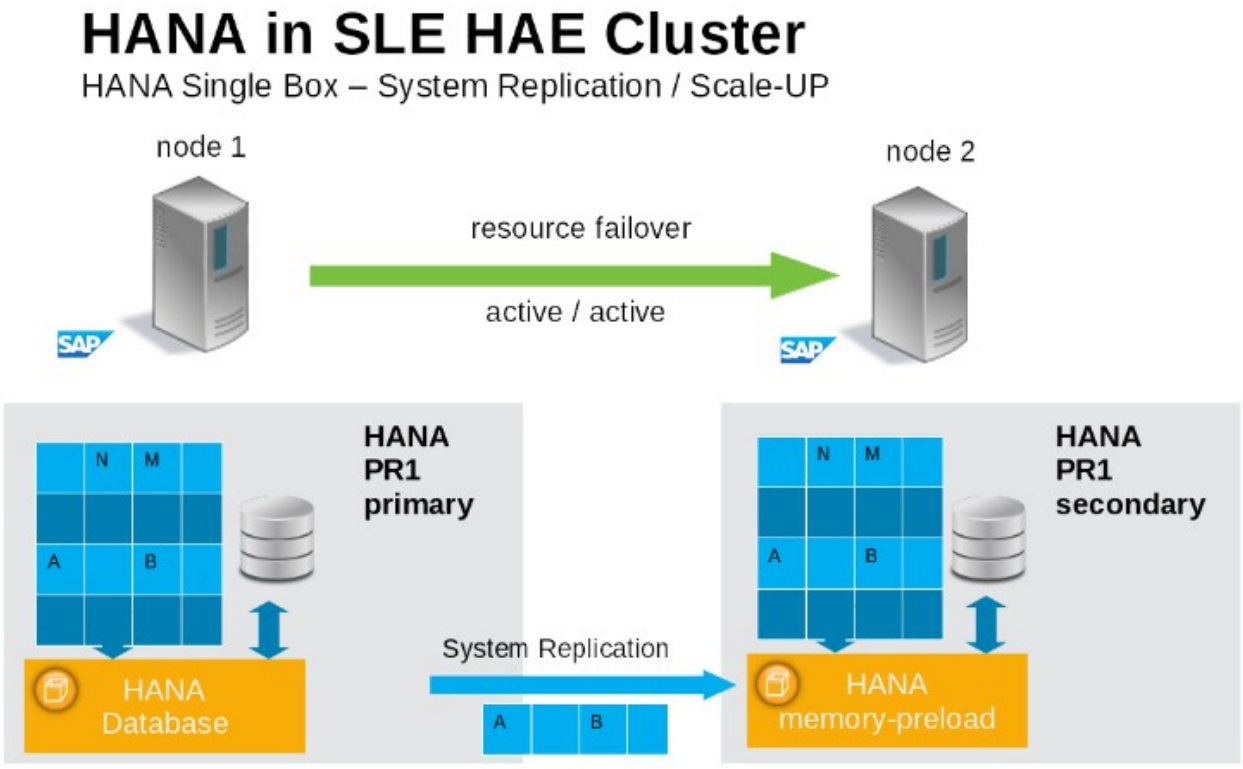


Abbildung 1: SAP HANA System Replication in the Cluster

SUSE has implemented the scale-up scenario with the **SAPHana** resource agent (RA), which performs the actual check of the SAP HANA database instances and is configured as a master/slave resource. In the scale-up scenario, the master assumes responsibility for the SAP HANA databases running in primary mode, and the slave is responsible for instances that are operated in synchronous (secondary) status.

To make configuring the cluster as simple as possible, SUSE also developed it's **SAPHanaTopology** resource agent. This runs on all nodes of an SLE 11 HAE cluster and gathers information about the statuses and configurations of SAP HANA system replications. It was designed as a normal (stateless) clone.

Customers Receive Complete Package

With both the SAPHana and SAPHanaTopology resource agents, customers will therefore be able to integrate SAP HANA system replications in their cluster. This has the advantage of enabling companies to use not only their business-critical SAP systems but also their SAP HANA databases without interruption while noticeably reducing their budgets. SUSE provides the extended solution together with best practices documentation.

SAP and hardware partners who do not have their own SAP HANA high-availability solution will also benefit from this new SUSE Linux development.

2 SUPPORTED SCENARIOS AND PREREQUISITES

For the first version of the SAPHanaSR resource agent software package we limit the support to the following scenarios and parameters:

1. Two-node clusters
2. Scale-UP (single-box to single-box) system replication
3. Both nodes are in the same network segment (layer 2)
4. Technical users and groups such as ***sid***adm are defined locally in the Linux system
5. Name resolution of the cluster nodes and the virtual IP address can be done locally on all cluster nodes
6. Time synchronization between the cluster nodes using NTP
7. There is no other SAP HANA system (like QA) on the replicating node which needs to be stopped during takeover.
8. Only one system replication for the SAP HANA database
9. Both SAP HANA instances have the same SAP Identifier (SID) and Instance-Number
10. If the cluster nodes are installed in different data centers or data center areas, the environment must match the requirements of the SLE HAE cluster product. This in special means the network latencies between the nodes and the recommended maximum distance. Please review our product documentation for SLE HAE about those recommendations.
11. As a good starting configuration for PoCs and projects we recommend to switch-off the automated registration of a failed primary. The setup `AUTOMATED_REGISTER="false"` is also the default since version 0.139.

If you need to implement a different scenario we strongly recommend to define a POC with SUSE. This POC will focus in testing the existing solution in your scenario. The limitation of most of the above items is mostly due to testing limits.

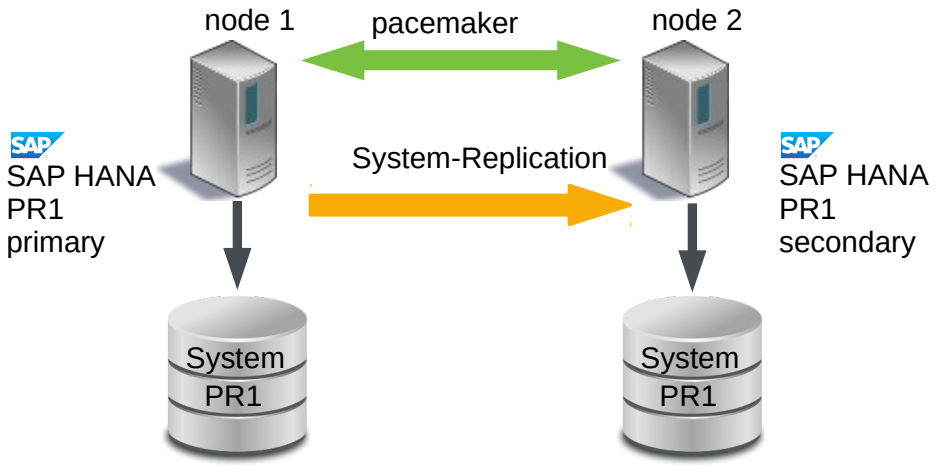
The resource agent supports SAP HANA in System replication beginning with HANA version 1.0 SPS 7 patch level 70.

Beside SAP HANA you need SAP hostagent to be installed at your system. Automated start of SAP HANA instances during system boot must be switched of.

3 SCOPE

This document describes in a short way how to setup the cluster to control SAP HANA in System Replication Scenarios. The document focuses only on the steps which need to be done to integrate an already installed and working SAP HANA with System Replication.

This setup builds a SAP HANA HA cluster in two data-centers in Walldorf and in Rot, installed on two SLES for SAP 11 SP3 virtual machines.



Direction of the System Replication will only be changed, if the parameter AUTOMATED_REGISTER is been changed to “true”.

We recommend to start with the default: AUTOMATED_REGISTER=”false”

Abbildung 2: Cluster with SAP HANA SR

Parameter	Value	Role
Custer node 1	suse01 192.168.1.10	Cluster node name and IP address.
Cluster node 2	suse02 192.168.1.11	Cluster node name and IP address.
SID	SLE	SAP Identifier (SID)
Instance number	00	Number of the SAP HANA database. For system replication als Instance Number+1 is blocked.
Network address	192.168.1.0	
Network mask	255.255.255.0	
Virtual IP address	192.168.1.20	
Storage		Storage for HDB data and log files is connected “locally” (per node; not shared)

Parameter	Value	Role
SBD	/dev/vdb	STONITH device
User Key	slehaloc	Database user key for accessing sync information at primary side.
HAWK Port	7630	

Tabelle 1: Parameters used in this document

4 HANA DATABASE INSTALLATION

Even when the document focuses on the integration of the already installed SAP HANA in already setup system replication into the SLE HAE cluster, this chapter in short says what has been done in our test environment. Please always use the official documentation from SAP to install SAP HANA and to setup the system replication.

4.1 SAP HANA Installation

4.1.1 Preparation

1. Read the SAP Installation and Setup Manuals available at the SAP Market Place
2. Download the SAP HANA Software from SAP Market Place

4.1.2 Installation and Checks

1. Install the SAP HANA Database as described in the SAP HANA Server Installation Guide
2. Verify that both databases are up and all processes of these Databases runs correctly

As Linux user *sid*adm use the command line tool “HDB” to get the overview about the running HANA processes. The output of “HDB info” should show something like shown in the following screen shot.

```
suse02:~> HDB info
```

```
USER       PID  ... COMMAND
sleadm    6561  ... -csh
sleadm    6635  ...   \_ /bin/sh /usr/sap/SLE/HDB00/HDB info
sleadm    6658  ...       \_ ps fx -U sleadm -o user,pid,ppid,pcpu,vsz,rss,args
sleadm    5442  ...  sapstart pf=/hana/shared/SLE/profile/SLE_HDB00_suse02
sleadm    5456  ...   \_ /usr/sap/SLE/HDB00/suse02/trace/hdb.sapSLE_HDB00 -d
-nw -f /usr/sap/SLE/HDB00/suse
sleadm    5482  ...       \_ hdbnameserver
sleadm    5551  ...       \_ hdbpreprocessor
sleadm    5554  ...       \_ hdbcompileserver
sleadm    5583  ...       \_ hdbindexserver
sleadm    5586  ...       \_ hdbstatisticsserver
sleadm    5589  ...       \_ hdbxsengine
sleadm    5944  ...       \_ sapwebdisp_hdb
pf=/usr/sap/SLE/HDB00/suse02/wdisp/sapwebdisp.pfl -f /usr/sap/SL
sleadm    5363  ...  /usr/sap/SLE/HDB00/exe/sapstartsrv
pf=/hana/shared/SLE/profile/SLE_HDB00_suse02 -D -u s
suse02:HDB:sleadm ~ 5>
```

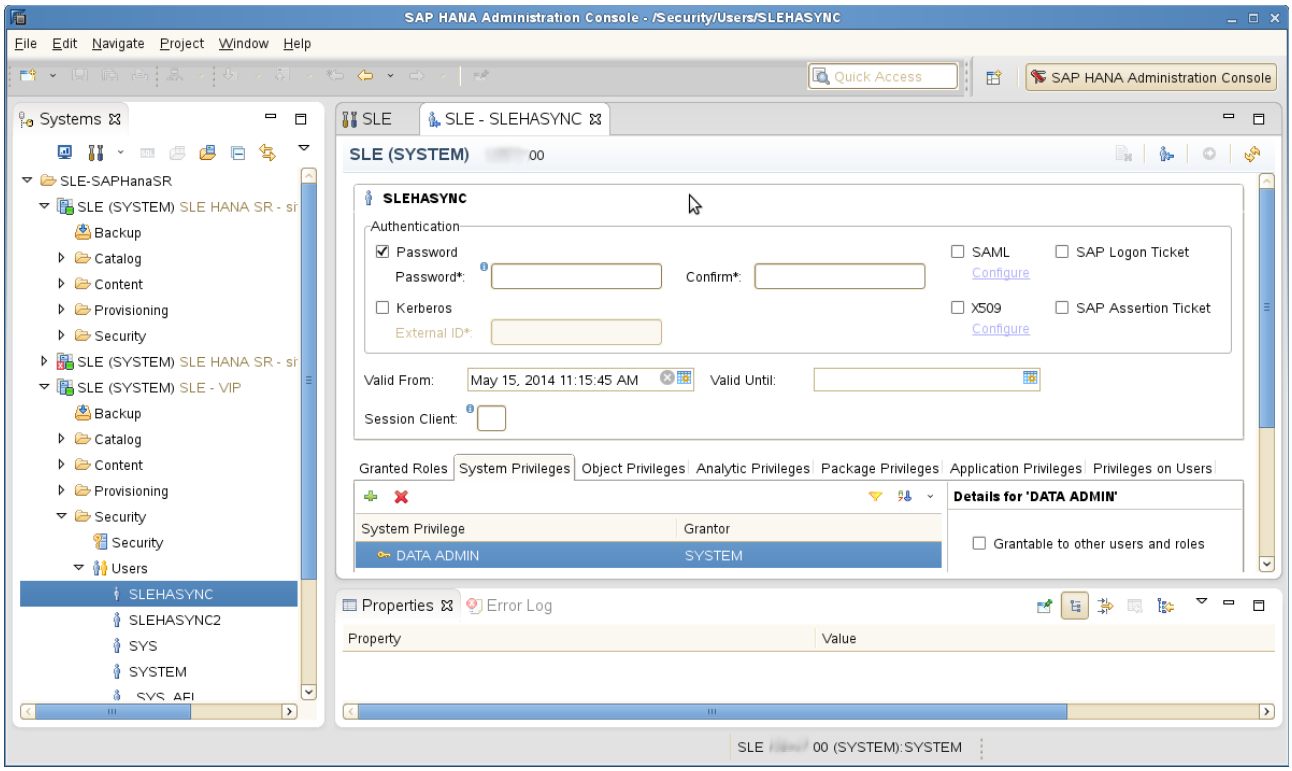
4.2 Postinstallation configuration

4.2.1 Database use slehasync

Create a user key in the secure store **as root linux user** on all cluster nodes. For a demo it would be ok use database user “system” which is of course a bit to powerful. For real scenarios it's important also to create a database user which will be referenced by the user key in the secure store. You can create such a database user in HANA studio.

Create a new user (like slehasync) and assign the system privilege “DATA_ADMIN” to the new user. If system privilege “DATA_ADMIN” is to powerful for your security policies, you need to check for more granular rights in the SAP documentation.

A good check, if your database user has enough rights to get the synchronization status is to first create a user store key and than send the query which is described at the end of this subsection.



As an alternative the user could also be created using hsbsql using comamand line.

```
suse01:~ # PATH=“$PATH:/usr/sap/SLE/HDB00/exe“

suse01:~ # hdbsql –U myadmin 'create user slehasync password L1nuxlab'

suse01:~ # hdbsql –U myadmin 'grant DATA ADMIN to slehasync'
```

While the database user need only to be created on one side (the other will get the user by the system replication), the user keys for password free access needs to be created on both nodes as the user keys are stored in the file system and not in the database.

For user key environment be sure that you use “localhost” as database host. This is needed, because the resource agent which needs the user key always should only access the local database instance.

The name of the user key “slehaloc” is a fix name used in the resource agent. The port should be setup as 3*nn*15, where nn is the SAP instance number like “00”.

```
suse01:~ # PATH=“$PATH:/usr/sap/SLE/HDB00/exe“

suse01:~ # hdbuserstore SET slehaloc localhost:30015 slehasync p45sWord
```

Verify the created setup as Linux user root. The option “-U *key*” tells hdbsql to use the credentials which are stored in the secure store. The table “dummy” is available for any database user, where each user has it's own dummy table. So far the following test only shows, that the user key has been defined correctly and we can login to the database.

```
suse01:~ # hdbsql –U slehaloc "select * from dummy"

DUMMY
"X"
```

Later you will need to check the access to the sync information. This has to be done after the system replication has been setup. The hdbsql call is desribed in section 4.3.4.

An example of the use of the user key could also be found in section 4.2.11.3 Performing a Data Backup (Batch Mode) the of Administration Guide.

4.2.2 Backup the Primary Database

Backup the primary database as described in Section 4.2.11.2 Performing a Data Backup (SQL Commands) of the SAP HANA Administration Guide.

If you have (for example) created a backup database user and a userkey “hanabackup” you can create an initial backup using the following command:

```
suse01:~ # hdbsql –U hanabackup "BACKUP DATA USING FILE ('backup')"
```

If the user key hanabackup has not been defined please use an instance/user/password combination for login.

Without a valid backup you could not bring SAP HANA into a system replication configuration.

4.3 HDB System Replication

For more information read Section 4.1.2 Setting up System Replication with hdbnsutil of SAP_HANA_Administration_Guide

4.3.1 Enable Primary Node

As Linux user *sid*adm enable the system replication at the primary node. You need to

define a site name (like “WALLDORF”). This site name must be uniqueue for all SAP HANA databases which are connected via system replication. This means the secondary must have a different site name.

```
suse01:~> hdbnsutil –sr_enable –name=WALLDORF
```

```
checking local nameserver:
checking for active nameserver …
nameserver is running, proceeding …
configuring ini files …
successfully enabled system as primary site …
done.
```

4.3.2 Verify the state of system replication

The command line tool hdbnsutil can be used to check the system replication mode and site name.

```
suse01:~> hdbnsutil –sr_state
```

```
checking for active or inactive nameserver …
System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~
mode: primary
site id: 1
site name: WALLDORF
Host Mappings:
~~~~~~~~~~~~~~
done.
```

The mode changed from “none” to “primary” and the site has now a site name and a site id.

4.3.3 Enable the Secondary Node

The SAP HANA database instance on the secondary side must be stopped before the instance could be registered for the system replication. You could use your preferred method to stop the instance (like HDB or sapcontrol). After the database instance is stopped successfully you could register the instance using hdbnsutil. Again use Linux user *sid*adm.

```
suse02:~> hdbnsutil –sr_register --remoteHost=suse01 --remoteInstance=00
--mode=sync --name=ROT
```

```
adding site …
checking for inactive nameserver …
nameserver suse02:30001 not responding.
collecting information …
updating local ini files …
done.
```

Now start the database instance again and verify the system replication status. At secondary side the mode should be one of „sync“, „syncmem“ or „async“. The mode depends on what have been defined during registration of the client.

The remoteHost is the primary node in our case, the remoteInstance is the database instanace number (here 00).

4.3.4 Checking the access to sync information

Now please check if the database user has enough rights to access the synchronisation information.

```
suse01:~ # hdbsql –U slehaloc 'select distinct REPLICATION_STATUS from
SYS.M_SERVICE_REPLICATION'
```

If you get a result like “error”, “unknown”, “syncing” or “active” than the user is able to access the synchronization info.

5 CONFIGURATION OF THE CLUSTER AND SAP HANA DATABASE INTEGRATION

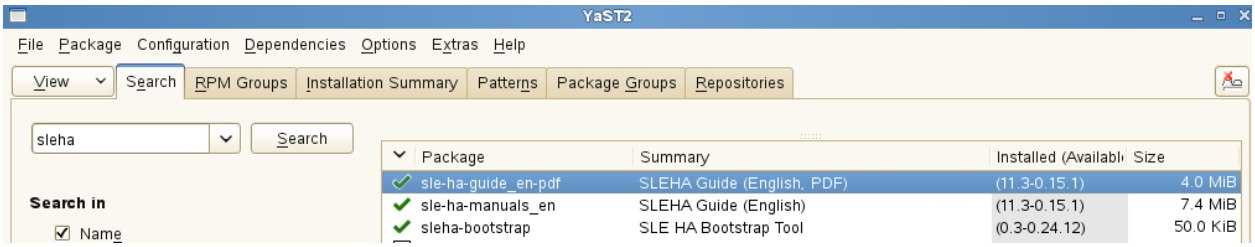
This chapter describes the configuration of the cluster software SUSE Linux Enterprise High Availability Extension, which is part of the SUSE Linux Enterprise Server for SAP Applications, and SAP HANA Database Integration.

5.1 Installation

If not already done in forehand, install the software packages with YaST. Alternatively, you can install them from the command line with zypper. This has to be done on both nodes.

```
suse01:~> zypper in -t pattern ha_sles
```

For more information please read Section 3.2.2 Initial Cluster Setup of of SUSE Linux Enterprise High Availability ExtensionSLEHA



5.2 Basic Cluster Configuration

The first step is to setup the base cluster framework. For convenience, use YaST2 or the sleha-init script. Depending on your needs, you should later add a second ring and also change to UCAST communication.

5.3 Initial cluster setup using sleha-init

For more information please read Section 3.4 Automatic Cluster Setup of of SUSE Linux Enterprise High Availability ExtensionSLEHA

Create an initial setup, using sleha-init command and follow the dialogs.

```
sleha-init
```

So far we have configured the basic cluster framework including:

- ssh keys
- csync2 to transfer configuration files
- SBD (at least one device)
- corosync (at least one ring)
- HAWK web interface

As requested by sleha-init, change the passwords of the user hacluster.

5.4 Adapting the Corosysnc and sbd Configuration

Depending on your needs, it is possible to add a second ring and also change to UCAST communication as described in 3.2.1 of Best Practices for SAP on SUSE Linux Enterprise.

Stop the already running cluster using “rcopenais stop”. After setup of the corosync config and sbd parameters we start the cluster again.

5.4.1 Corosync conf

Adjust following blocks in the file /etc/corosync/corosync.conf, see also example on the end of this document.

```
totem {
     ...
      Interface {
            #Network Address to be bind for this interface setting
            bindnetaddr:    10.17.140.0
            member {
                  memberaddr:     192.168.1.36
            }
            member {
                  memberaddr:     192.168.1.191
            }
      #The multicast port to be used
      mcastport:      5405
      #The ringnumber assigned to this interface setting
      ringnumber:     0
      }
      #How many threads should be used to encypt and sending message.
     threads:        4
     #
     transport:      udpu
      #How many token retransmits should be attempted before forming
      # a new configuration.
     token_retransmits_before_loss_const:    10
      #To make sure the auto-generated nodeid is positive
     clear_node_high_bit:    new
  …
}
```

5.4.2 Adapting sbd config

If you use the newest updates of the pacemaker packages from the SUSE maintenances-channels, you can also use the -P option (*Check Pacemaker quorum and node health*), which enables the cluster nodes not to self-fence if SBDs are lost, but pacemaker communication is still available. The timeout parameter -t defines the reprobe of the SBD device. The timeout is defined in seconds.

```
# /etc/sysconfig/sbd
```

```
SBD_DEVICE="/dev/vdb;/dev/vdc"
SBD_OPTS="-W -P -t 300"
```

5.4.3 Verify the sbd device

It's a good practice to check, if the sbd device could be accessed from both nodes and does contain valis records.

```
suse01:~ # sbd -d /dev/vdb dump
```

```
==Dumping header on disk /dev/vdb
Header version     : 2.1
UUID               : 0f4ea13e-fab8-4147-b9b2-3cdcfff07f86
Number of slots    : 255
Sector size        : 512
Timeout (watchdog) : 20
Timeout (allocate) : 2
Timeout (loop)     : 1
Timeout (msgwait)  : 40
==Header on disk /dev/vdb is dumped
```

To check the current sbd entries for the various cluster nodes you can use “sbd list”. If all entries are “clear” no fencing task is marked in the sbd device.

```
suse01:~ # sbd -d /dev/vdb list
```

```
0       suse01  clear
1       suse02  clear
```

For more information over SBD configuration parameters, please read section 17.1.3 of SUSE Linux Enterprise High Availability ExtensionSLEHA:
https://www.suse.com/documentation/sle_ha/pdfdoc/book_sleha/book_sleha.pdf

Now it's time to restart the cluster at the first node again (rcopenais start).

5.5 Cluster configuration on the second node

The second node of the two nodes cluster cound be integrated by starting the command “sleha-join”. This command just asks for the IP address or name of the first cluster node. Than all needed configuration files are copied over. As a result the cluster is started on both nodes.

```
sleha-join
```

5.6 Install the SAPHanaSR Resource Agent Package

Now the resource agents to control the SAP HANA system replication need to installed at both cluster nodes.

```
suse01:~> zypper install SAPHanaSR
```

Before general availability via the SUSE update channels for SUSE Linux Enterprise

Server for SAP Applications you can use the following syntax to install a package stored in a file system with local access:

```
suse01:~> zypper install SAPHanaSR-0.140-4.1.noarch.rpm
```

5.7 Check the Cluster for the first Time

Now it's time to check and optionally start the cluster for the first time on both nodes.

```
suse01:~ # rcopenais status
suse02:~ # rcopenais status

suse01:~ # rcopenais start
suse02:~ # rcopenais start
```

Check the cluster status with crm_mon. We use the option "-r" to see also resources, which are configured but stopped.

```
crm_mon -r
```

The command will show the "empty" cluster and will print something like in the following screen output. The most interesting information for now is that there are two nodes in status "online" and the message "partition with quorum".

```
Last updated: Fri Apr 25 09:44:59 2014
Last change: Fri Apr 25 08:30:58 2014 by root via cibadmin on suse01
Stack: classic openais (with plugin)
Current DC: suse01 - partition with quorum
Version: 1.1.9-2db99f1
2 Nodes configured, 2 expected votes
6 Resources configured.
Online: [ suse01 suse02]
Full list of resources:
stonith-sbd     (stonith:external/sbd): Started suse01
```

5.8 Configure the Cluster using crmsh

This section describes how to configure Constraints, Resources, Bootstrap and STONITH using the ***crm configure*** Shell command. as described in section 7.1.2 of SUSE Linux Enterprise High Availability ExtensionSLEHA

Use the crmsh to add the objects to CRM. Copy the following examples to a local file, edit the file and than load the configuration to the CRM

```
suse01:~ # vi crm-fileXX

suse01:~ # crm configure load update crm-fileXX
```

5.8.1 Cluster bootsrap and more

The first example defines the cluster bootstrap options, the resource and operation

defaults.

```
vi crm-bs.txt
# enter the following to crm-bs.txt

property $id="cib-bootstrap-options" \
        no-quorum-policy="ignore" \
        stonith-enabled="true" \
        stonith-action="reboot" \
        stonith-timeout="150s"
rsc_defaults $id="rsc-options" \
        resource-stickiness="1000" \
        migration-threshold="5000"
op_defaults $id="op-options" \
        record-pending="false" \
        timeout="600"

# now we load the file to the cluster
crm configure load update crm-bs.txt
```

5.8.2 STONITH device

The next configuration part defines a SBD disk STONITH resource.

```
vi crm-sbd.txt
# enter the following to crm-sbd.txt


primitive stonith-sbd stonith:external/sbd



# now we load the file to the cluster
crm configure load update crm-bs.txt
```

For alternative IPMI/ILO setup see our cluster product documentation.

5.8.3 SAPHanaTopology

First we define the group of resources needed, before the HANA instances can be started edit the changes in a txt file f.e. resource01.txt an load these with the command: crm configure load update /resource01.txt

```
vi crm-saphanatop.txt
# enter the following to crm-saphanatop.txt

primitive rsc_SAPHanaTopology_SLE_HDB00 ocf:suse:SAPHanaTopology \
        operations $id="rsc_sap2_SLE_HDB00-operations" \
        op monitor interval="10" timeout="600" \
        op start interval="0" timeout="600" \
        op stop interval="0" timeout="300" \
        params SID="SLE" InstanceNumber="00"
clone cln_SAPHanaTopology_SLE_HDB00 rsc_SAPHanaTopology_SLE_HDB00 \
        meta is-managed="true" clone-node-max="1" target-role="Started"
```

```
# now we load the file to the cluster
crm configure load update crm-saphanatop.txt
```

The most important parameters here are SID and InstanceNumber, which are in the SAP market quite self explaining. Beside these parameters the timeout values or the operations (start, monitor, stop) are typical tunables.

5.8.4 SAPHana

First we define the group of resources needed, before the HANA instances can be started edit the changes in a txt file f.e. resource02.txt an load these with the command: crm configure load update /resource02.txt

```
vi crm-saphana.txt
# enter the following to crm-saphana.txt

primitive rsc_SAPHana_SLE_HDB00 ocf:suse:SAPHana \
        operations $id="rsc_sap_SLE_HDB00-operations" \
        op start interval="0" timeout="3600" \
        op stop interval="0" timeout="3600" \
        op promote interval="0" timeout="3600" \
        op monitor interval="60" role="Master" timeout="700" \
        op monitor interval="61" role="Slave" timeout="700" \
        params SID="SLE" InstanceNumber="00" PREFER_SITE_TAKEOVER="1" \
        DUPLICATE_PRIMARY_TIMEOUT="1200"
ms msl_SAPHana_SLE_HDB00 rsc_SAPHana_SLE_HDB00 \
        meta is-managed="true" notify="true" clone-max="2" clone-node-max="1"
target-role="Started"

# now we load the file to the cluster
crm configure load update crm-saphana.txt
```

The most important parameters here are SID and InstanceNumber, which are in the SAP market quite self explaining. Beside these parameters the timeout values or the operations (start, promote, monitors, stop) are typical tuneables.

5.8.5 The virtual ip address

The last resource to be added to the cluster is covering the virtual IP address.

```
vi crm-vip.txt
# enter the following to crm-vip.txt

primitive rsc_ip_SLE_HDB00 ocf:heartbeat:IPaddr2 \
        meta target-role="Started" is-managed="true" \
        operations $id="rsc_ip_SLE_HDB00-operations" \
        op monitor interval="10s" timeout="20s" \
        params ip="192.168.1.20"

# now we load the file to the cluster
```

```
crm configure load update crm-vip.txt
```

In most installations only the parameter ip needs to be set to the virtual ip address to be presented to the client systems.

5.8.6 Constraints

Two constraints are organizing the correct placement of the virtual IP address for the client database access and the start order between the two resource agents SAPHana and SAPHanaTopology.

```
vi crm-cs.txt
# enter the following to crm-cs.txt

colocation col_saphana_ip_SLE_HDB00 2000: rsc_ip_SLE_HDB00:Started \
   msl_SAPHana_SLE_HDB00:Master
order ord_SAPHana_SLE_HDB00 2000: cln_SAPHanaTopology_SLE_HDB00 \
   msl_SAPHana_SLE_HDB00

# now we load the file to the cluster
crm configure load update crm-cs.txt
```

6 ADMINISTRATION

6.1 DO's and DON'T Dos

In your project you should...

- Define STONITH before adding other resources to the cluster
- Do intensive testing
- Tune the timeouts of the operations of SAPHana and SAPHanaTopology
- Start with PREFER_SITE_TAKEOVER=”true”, AUTOMATED_REGISTER=”false” and DUPLICATE_PRIMARY_TIMEOUT=”7200”

It's strong recommended to avoid...

- Rapidly change / change back cluster configuration such as: Set node standby and online again; stopping/starting the master/slave resource
- Creating a cluster without proper time synchronization or unstable name resolutions for hosts, users and groups

6.2 Monitoring

You can use the High Availability Web Console (HAWK), SAP HANA Studio and different command line tools for cluster status requests.

6.2.1 HAWK – Cluster Status and more

You can use an internet browser to check the cluster status. If you setup the cluster using sleha-init and you had installed all packages as described above, your system will provide a very useful web interface. You can use this grafical web interface to get an overview of the complete cluster status, doing administrative tasks or even configure resources and cluster boostrap parameters. Read our product manuals for a complete documentaion of this powerful user interface.

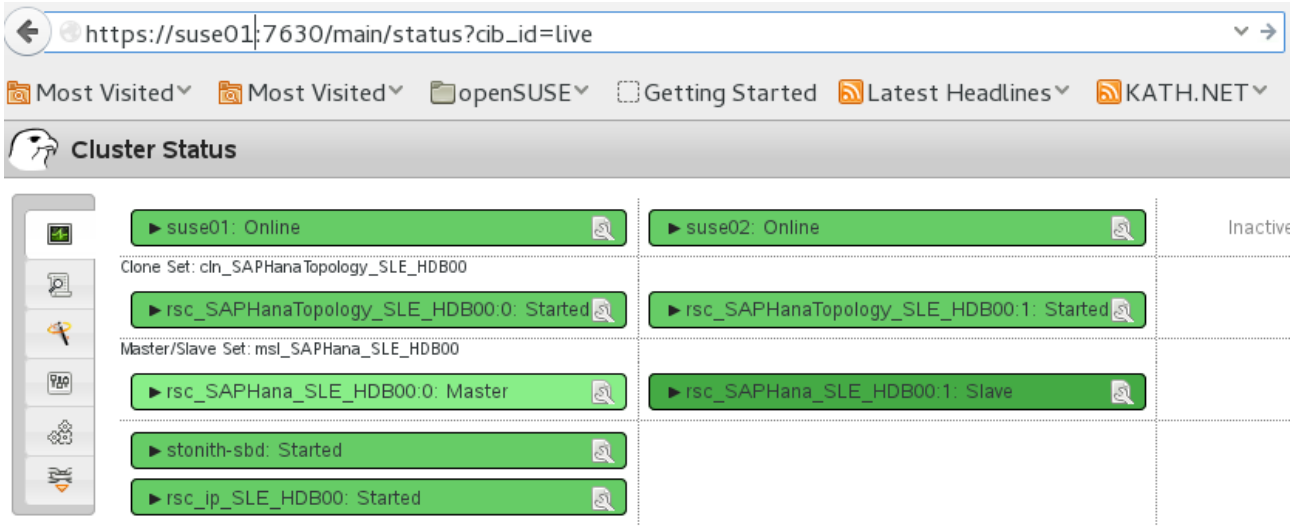


Abbildung 3: Cluster Status in HAWK

6.2.2 SAP HANA Studio

Database specific administration and checks could be done with the SAP HANA studio.

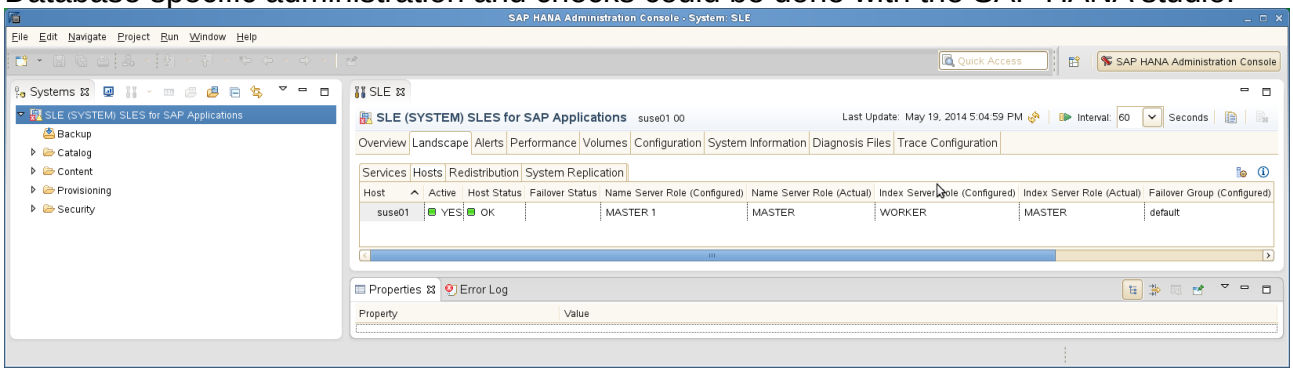


Abbildung 4: SAP HANA Studio - Landscape

6.2.3 Cluster Command-Line Tools

A simple overview can be get by calling crm_mon. Using option “-r” shows also stopped but already configured resources. Option “-1” tells crm_mon to just output the status once instead of periodically.

```
suse01:~ # crm_mon -r1
```

```
Last updated: Fri May 16 17:18:56 2014
Last change: Fri May 16 17:17:45 2014 by root via crm_attribute on suse01
Stack: classic openais (with plugin)
Current DC: suse01 - partition with quorum
Version: 1.1.10-f3eeaf4
2 Nodes configured, 2 expected votes
6 Resources configured
```

```
Online: [ suse01 suse02 ]

Full list of resources:

 stonith-sbd    (stonith:external/sbd): Started suse01
 Clone Set: cln_SAPHanaTopology_SLE_HDB00 [rsc_SAPHanaTopology_SLE_HDB00]
     Started: [ suse01 suse02 ]
 Master/Slave Set: msl_SAPHana_SLE_HDB00 [rsc_SAPHana_SLE_HDB00]
     Masters: [ suse01 ]
     Slaves: [ suse02 ]
```

See the manual page crm_mon(8) for details.

To show some of the SAPHana, SAPHanaTopology **resource agent internal** values you can call the program show_SAPHanaSR_attributes. The internal values, storage place and their parameter names may change in the next versions. The commands show_SAPHanaSR_attributes will always fetch the values from the correct storage place. Do not use cluster commands like crm_attribute to fetch the values directly from the cluster, because in such cases your methods will be broken, when we need to move an attribute to a different storage place or even out of the cluster. For first show_SAPHanaSR_attributes is a test program only and should not used for automated system monitoring.

```
suse01:~ # /usr/share/SAPHanaSR/tests/show_SAPHanaSR_attributes
```

```
Host \ Attr clone_state remoteHost roles       ... site     srmode sync_state ...
--------------------------------------------------------------------------------
suse01      PROMOTED    suse02     4:P:master1:... WALLDORF sync   PRIM       ...
suse02      DEMOTED     suse01     4:S:master1:... ROT      sync   SOK        ...
```

6.2.4 SAP HANA LandscapeHostConfiguration

To check the status of a SAPHana database and to figure out of the cluster should react or not, you can use the landscapeHostConfiguration python script to be called as Linux user *sid*adm.

```
suse01:~> python /usr/sap/SLE/HDB00/exe/python_support/landscapeHostConfigurati-
on.py
```

```
| Host   | Host   | Host   | Failover | ... NameServer  | NameServer  | IndexServer | IndexServer |
|        | Active | Status | Status   | ... Config Role | Actual Role | Config Role | Actual Role |
| ------ | ------ | ------ | -------- | ... ----------- | ----------- | ----------- | ----------- |
| suse01 | yes    | ok     |          | ... master 1    | master      | worker      | master      |

overall host status: ok
```

Following the SAP HA guideline the SAPHana resource agent interprets the return codes in the following way:

Return Code	Interpretation
4	SAP HANA database is up and ok. The cluster does interprete this as a correctly running database.
3	SAP HANA database is up and in status info. The cluster does interprete this as a correctly running database.

Return Code	Interpretation
2	SAP HANA database is up and in status warning. The cluster does interprete this as a correctly running database.
1	SAP HANA database is down. This could trigger also an takeover, if the database should be up and is not down by intention.
0	Internal Script Error – to be ignored.

7 ADDENDUM

7.1 Useful links, manuals and SAP Notes

7.1.1 SUSE Best practices and more

- Best Practices for SAP on SUSE Linux Enterprise
- Fail-Safe Operation of SAP HANA®: SUSE Extends Its High-Availability Solution

7.1.2 SUSE product documentation

The SUSE product manuals and documentation could be loaded at www.suse.com/documentation.

- https://www.suse.com/documentation/sles_for_sap/index.html
- SUSE Linux Enterprise High Availability Extension SLEHA

7.1.3 SAP product documentation

- SAP HANA Server Installation Guide
- http://help.sap.com/hana/SAP_HANA_Server_Installation_Guide_en.pdf
  SAP HANA Platform SPS 07 - Document Version: 1.1 - 13-03-2014
- http://help.sap.com/hana/SAP_HANA_Administration_Guide_en.pdf
  SAP HANA Platform SPS 07 - Document Version: 1.1 - 13-03-2014

7.1.4 SAP Notes

- 1310037 - SUSE LINUX Enterprise Server 11: Installation notes
- 1824819 - SAP HANA DB: Recommended OS settings for SLES 11 / SLES for SAP Applications 11 SP2
- 1876398 - Network configuration for System Replication in HANA SP6
- 611361 - Hostnames of SAP servers

7.1.5 Additional useful links

- http://scn.sap.com/community/business-continuity/blog/2013/08/26/storage-based-fencing-in-mixed-high-availability-environments

7.2 Sample of sleha-init configuration.

```
suse01:~ # sleha-init
```

```
ntp on
Enabling sshd service
```

```
Generating ssh key
Configuring csync2
csync2 is already configured - overwrite? [y/N] y
Generating csync2 shared key (this may take a while)...done
Enabling csync2 service
Enabling xinetd service
csync2 checking files
Enabling sshd service
Generating ssh key
Configuring csync2
csync2 is already configured - overwrite? [y/N] y
Generating csync2 shared key (this may take a while)...done
Enabling csync2 service
Enabling xinetd service
csync2 checking files
Configure Corosync:
This will configure the cluster messaging layer. You will need
to specify a network address over which to communicate (default
is eth0's network, but you can use the network address of any
active interface), a multicast address and multicast port.
/etc/corosync/corosync.conf already exists - overwrite? [y/N] y
Network address to bind to (e.g.: 192.168.1.0) [192.168.1.0]
Multicast address (e.g.: 239.x.x.x) [239.107.222.58] 238.50.0.1
Multicast port [5404]
Configure SBD:
If you have shared storage, for example a SAN or iSCSI target,
you can use it avoid split-brain scenarios by configuring SBD.
This requires a 1 MB partition, accessible to all nodes in the
cluster. The device path must be persistent and consistent
across all nodes in the cluster, so /dev/disk/by-id/* devices
are a good choice. Note that all data on the partition you
specify here will be destroyed.
Do you wish to use SBD? [y/N] y
Path to storage device (e.g. /dev/disk/by-id/...) [] /dev/vdb
All data on /dev/vdb will be destroyed
Are you sure you wish to use this device [y/N] y
Initializing SBD......done
Enabling hawk service
HA Web Konsole is now running, to see cluster status go to:
https://192.168.1.10:7630/
Log in with username 'hacluster', password 'linux'
WARNING: You should change the hacluster password to something more secure!
Enabling openais service
Waiting for cluster........done
Loading initial configuration
Done (log saved to /var/log/sleha-bootstrap.log)
Change the hacluster password
```

7.3 Sample Cluster configuration in crmsh format

The following complete crm configuration is for a two node cluster (suse01, suse02) and a SAP HANA database with SID SLE and instance number 00. The virtual IP address in the example is 192.168.1.20.

```
node suse01
node suse02

primitive rsc_SAPHanaTopology_SLE_HDB00 ocf:suse:SAPHanaTopology \
        operations $id="rsc_sap2_SLE_HDB00-operations" \
        op monitor interval="10" timeout="300" \
        op start interval="0" timeout="300" \
        op stop interval="0" timeout="300" \
        params SID="SLE" InstanceNumber="00"
primitive rsc_SAPHana_SLE_HDB00 ocf:suse:SAPHana \
        operations $id="rsc_sap_SLE_HDB00-operations" \
        op monitor interval="61" role="Slave" timeout="700" \
        op start interval="0" timeout="3600" \
        op stop interval="0" timeout="3600" \
        op promote interval="0" timeout="3600" \
        op monitor interval="60" role="Master" timeout="700" \
        params SID="SLE" InstanceNumber="00" PREFER_SITE_TAKEOVER="1"
DUPLICATE_PRIMARY_TIMEOUT="1200"
primitive rsc_ip_SLE_HDB00 ocf:heartbeat:IPaddr2 \
        meta target-role="Started" is-managed="true" \
        operations $id="rsc_ip_SLE_HDB00-operations" \
        op monitor interval="10s" timeout="20s" \
        params ip="192.168.1.20"
primitive stonith-sbd stonith:external/sbd
ms msl_SAPHana_SLE_HDB00 rsc_SAPHana_SLE_HDB00 \
        meta is-managed="true" notify="true" clone-max="2" clone-node-max="1" target-role="Started"
clone cln_SAPHanaTopology_SLE_HDB00 rsc_SAPHanaTopology_SLE_HDB00 \
        meta is-managed="true" clone-node-max="1" target-role="Started"
colocation col_saphana_ip_SLE_HDB00 2000: rsc_ip_SLE_HDB00:Started
order ord_SAPHana_SLE_HDB00 2000: cln_SAPHanaTopology_SLE_HDB00 msl_SAPHana_SLE_HDB00
msl_SAPHana_SLE_HDB00:Master
property $id="cib-bootstrap-options" \
        dc-version="1.1.10-f3eeaf4" \
        cluster-infrastructure="classic openais (with plugin)" \
        expected-quorum-votes="2" \
        no-quorum-policy="ignore" \
        stonith-enabled="true" \
        stonith-action="reboot" \
        stonith-timeout="150s" \
        last-lrm-refresh="1398346620"
rsc_defaults $id="rsc_default-options" \
        resource-stickiness="1000" \
        migration-threshold="5000"
op_defaults $id="op_defaults-options" \
        record-pending="false" \
        timeout="600"
```

7.4 Example for /etc/corosync/corosync.conf

The following file shows a typical corosync configuration with one ring. Please view SUSE product documentation about details and about additional rings.

```
aisexec {
        #Group to run aisexec as. Needs to be root for Pacemaker
        group:  root
```

```
        #User to run aisexec as. Needs to be root for Pacemaker
        user:   root
}
service {
        #Default to start mgmtd with pacemaker
        use_mgmtd:      yes
        #Use logd for pacemaker
        use_logd:       yes
        ver:    0
        name:   pacemaker
}
totem {
        #The mode for redundant ring. None is used when only 1 interface
        #specified, otherwise, only active or passive may be choosen
        rrp_mode:       none
        #How long to wait for join messages in membership protocol. in ms
        join:   60
        #The maximum number of messages that may be sent by one processor
        #on receipt of the token.
        max_messages:   20
        #The virtual synchrony filter type used to indentify a primary
        #component. Change with care.
 vsftype:        none
        #Timeout for a token lost. in ms
        token:  5000
        #How long to wait for consensus to be achieved before starting a
        #new round of membership configuration.
        consensus:      6000
        interface {
                #Network Address to be bind for this interface setting
                bindnetaddr:    192.168.1.0.0
                member {
                        memberaddr:     192.168.1.10
                }
                member {
                        memberaddr:     192.168.1.11
                }
                #The multicast port to be used
                mcastport:      5405
                #The ringnumber assigned to this interface setting
                ringnumber:     0
        }
        #HMAC/SHA1 should be used to authenticate all message
        secauth:        on
        #The only valid version is 2
        version:        2
        #How many threads should be used to encypt and sending message.
        #Only have meanings when secauth is turned on
        threads:        4
        #
        transport:      udpu
        #How many token retransmits should be attempted before forming a
        #new configuration.
        token_retransmits_before_loss_const:    10
        #To make sure the auto-generated nodeid is positive
        clear_node_high_bit:    new
}
logging {
        #Log to a specified file
        to_logfile:     no
        #Log to syslog
        to_syslog:      yes
        #Whether or not turning on the debug information in the log
        debug:  off
        #Log timestamp as well
        timestamp:      off
        #Log to the standard error output
        to_stderr:      no
```

```
        #Logging file line in the source code as well
        fileline:       off
        #Facility in syslog
        syslog_facility:        daemon
}
amf {
        #Enable or disable AMF
        mode:   disable
}
```