SAP HANA SR Cost Optimized Scenario

SUSE Linux Enterprise Server for SAP Applications 12 SP1

SAP HANA SR Cost Optimized Scenario

SUSE Linux Enterprise Server for SAP Applications 12 SP1

by Fabian Herschel, Lee Martin, and Rainer Molitor

Publication Date: July 27, 2017

SUSE LLC
10 Canal Park Drive
Suite 200
Cambridge MA 02141
USA
https://www.suse.com/documentation

Contents

1 About This Guide

SUSE® Linux Enterprise Server for SAP Applications is optimized in various ways for SAP* applications. This guide provides detailed information about installing and customizing SUSE Linux Enterprise Server for SAP Applications for SAP HANA system replication in the cost optimized scenario.

1.1 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 the 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 provides two resource agents for customers to ensure the high availability of SAP HANA system replications.

1.1.1 Scale-Up vs. Scale-Out

The first set of scenarios include the architecture and development of *scale-up* solutions. For this scenarios SUSE developed the scale-up resource agent package `SAPHanaSR`. 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).

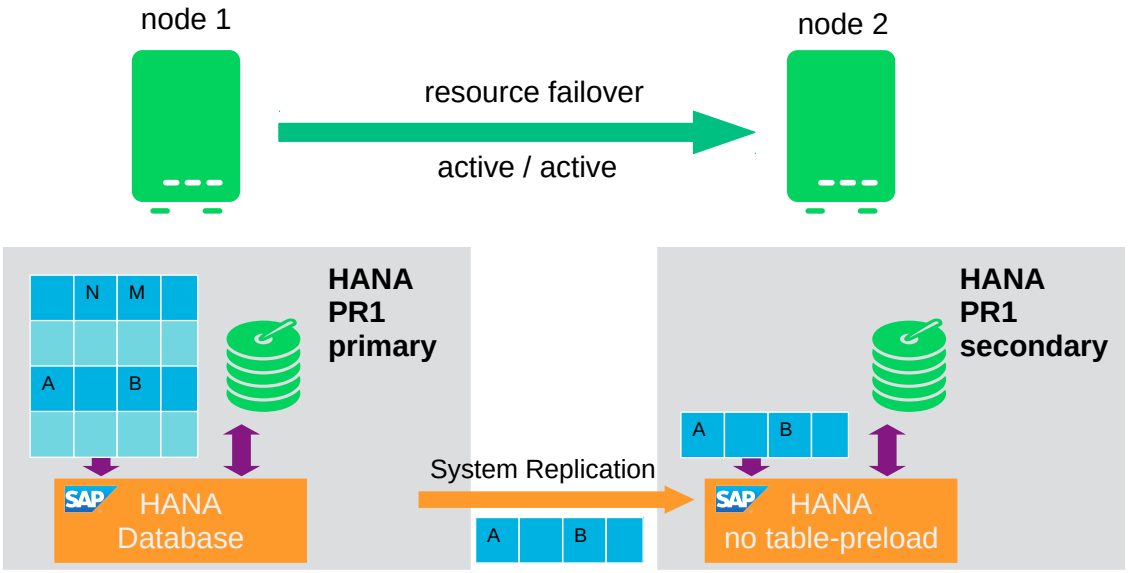

FIGURE 1.1: SAP HANA SYSTEM REPLICATION IN THE CLUSTER WITHOUT TABLE PRELOAD

The second set of scenarios include the architecture and development of *scale-out* solutions (multibox replication). For this scenarios SUSE developed the scale-out resource agent package `SAPHanaSR-ScaleOut`. 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. SAP HANA system replication automation for scale-out will be described in an own document available in the resource library at https://www.suse.com/products/sles-for-sap/resource-library/.

1.1.2 Scale-Up scenarios and resource agents

SUSE has implemented the scale-up scenario with the `SAPHana` resource agent (RA), which performs the actual check of the SAP HANA database instances.This RA 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 12 HAE cluster and gathers information about the statuses and configurations of SAP HANA system replications. It is designed as a normal (stateless) clone.

SAP HANA System replication for Scale-Up is supported in the following scenarios or use cases:

- Performance optimized ($A => B$). This scenario and setup is described in an other document in the resource library (https://www.suse.com/products/sles-for-sap/resource-library/).
  In the performance optimized scenario a HANA RDBMS A is synchronizing with a HANA RDBMS on a second node. As the HANA RDBMS on the second node is configured to preload the tables the takeover time is typically very short.

- Cost optimized ($A => B, Q$). This scenario and setup is described in this document.
  In the cost optimized scenario the second node is also used for a non-productive HANA RDBMS system (like QAS or TST). Whenever a takeover is needed the non-productive system must be stopped first. As the productive secondary system on this node must be limited in using system resources, the table preload must be switched off and a possible takeover needs longer as in the performance optimized use case.

- Multi Tier ($A => B -> C$). This scenario and setup is described in an other document in the resource library (https://www.suse.com/products/sles-for-sap/resource-library/).
  A Multi Tier system replication has an additional target, which must be connected to the secondary (chain topology).

- Multi-tenancy or MDC.
  Multi-tenancy is supported for all above scenarios and use cases. This scenario is supported since SAP HANA SPS09. The set-up and configuration from cluster point of view is the same for multi-tenancy and single container, so you can use the above documents for both kinds of scenarios. In our notation we mark a MDC like *%A*. This means MDC, performance optimized is abbreviated as $\%A => \%B$.

1.1.3 The concept of the Cost Optimized Scenario

SAP allows to run a non-productive instance of SAP HANA on the system replication site on node 2 (QAS).

In case of failure of the primary SAP HANA on node 1 the cluster first tries to restart the SAP HANA locally on this node. If the restart is not possible or if the complete node 1 is crashed, the takeover process will be triggered.

In case of a takeover the secondary (replica) of this SAP HANA on node 2 is started after the shutdown of the non-productive SAP HANA.

Alternatively you could also configure a different resource handling procedure, but we recommend to try to restart SAP HANA locally first, as a takeover with non preloaded tables could consume much time and also the needed stop of the QAS system will take additional time. Thus in many environments the local restart will be faster.

To achieve an automation of this resource handling process, we can utilize the SAP HANA resource agents included in SAPHanaSR. System replication of the productive database is done with SAPHana and SAPHanaTopology and handling of the non-productive database with the well known SAPDatabase resource agent. The idea behind is the following:

The SUSE pacemaker/openais framework is configured to run and monitor the productive SAP HANA database in system replication configuration as described in the setup guide (you need to complete a form to read the document) coming with the SAPHanaSR resource agents. The operation of the non-productive SAP HANA instance is done by the SAPDatabase resource agent because it is a single database without replication.

The automatic shutdown of the non-productive SAP HANA database (QAS) is achieved by cluster rules (anti-collocation of SAP HANA prod vs. SAP HANA non-prod), i.e. if the primary SAP HANA system (HA1) fails the anti-collocation rules for SAP HANA non-prod (QAS) are triggered and the SAPDatabase resource agent shuts down the non-productive SAP HANA database. The takeover to node 2 takes a long period of time, because the non-productive database needs to be stopped gracefully prior to takeover the productive database. This prolonged takeover time is the main disadvantage of the cost optimization scenario.

Beside the description of the concept in this guide, please also read the corresponding SAP documentation for example *Using Secondary Servers for Non-Productive systems*, ↑SAP HANA Master Guide.

1.1.4 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 needed 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 SUSE Linux development.

1.2 Additional Documentation and Resources

Chapters in this manual contain links to additional documentation resources that are either available on the system or on the Internet.

For the latest documentation updates, see http://www.suse.com/documentation.

You can also find numerous whitepapers, a best-practices guide, and other resources at the SUSE Linux Enterprise Server for SAP Applications resource library: https://www.suse.com/products/sles-for-sap/resource-library/.

1.3 Feedback

Several feedback channels are available:

Bugs and Enhancement Requests

For services and support options available for your product, refer to http://www.suse.com/support/.

To report bugs for a product component, go to https://scc.suse.com/support/requests, log in, and select *Submit New SR*.

User Comments

We want to hear your comments about and suggestions for this manual and the other documentation included with this product. Use the User Comments feature at the bottom of each page in the online documentation or go to http://www.suse.com/doc/feedback.html and enter your comments there.

Mail

For feedback on the documentation of this product, you can also send a mail to `doc-team@suse.de`. Make sure to include the document title, the product version and the publication date of the documentation. To report errors or suggest enhancements, provide a concise description of the problem and refer to the respective section number and page (or URL).

1.4 Documentation Conventions

The following typographical conventions are used in this manual:

- `/etc/passwd`: directory names and file names
- *`placeholder`*: replace *`placeholder`* with the actual value
- `PATH`: the environment variable PATH
- **`ls`**, `--help`: commands, options, and parameters
- `user`: users or groups
- `Alt`, `Alt`–`F1`: a key to press or a key combination; keys are shown in uppercase as on a keyboard
- *File*, *File* › *Save As*: menu items, buttons
- `amd64, em64t, ipf` This paragraph is only relevant for the architectures `amd64`, `em64t`, and `ipf`. The arrows mark the beginning and the end of the text block.
  `System z, ipseries` This paragraph is only relevant for the architectures `System z` and `ipseries`. The arrows mark the beginning and the end of the text block.
- *Dancing Penguins* (Chapter *Penguins*, ↑Another Manual): This is a reference to a chapter in another manual.

2 Supported Scenarios and Prerequisites

With the `SAPHanaSR` resource agent software package, we limit the support to Scale-Up (single-box to single-box) system replication with the following configurations and parameters:

- Two-node clusters.
- The cluster must include a valid STONITH method.
  - Any STONITH mechanism supported by SLE 12 HAE (like SDB, IPMI) is supported with SAPHanaSR.
  - This guide is focusing on the sbd fencing method as this is hardware independent.
  - If you use sbd as the fencing mechanism, you need one or more shared drives. For productive environments, we recommend more than one sbd device.
- Both nodes are in the same network segment (layer 2).
- Technical users and groups, such as `<sid>adm` are defined locally in the Linux system.
- Name resolution of the cluster nodes and the virtual IP address must be done locally on all cluster nodes.
- Time synchronization between the cluster nodes using NTP.
- Both SAP HANA instances have the same SAP Identifier (SID) and instance number.
- 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. Of particular concern are the network latencies and recommended maximum distance between the nodes. Please review our product documentation for SLE HAE about those recommendations.
- Automated registration of a failed primary after takeover.
  - As a good starting configuration for projects, we recommend to switch off the automated registration of a failed primary. The setup `AUTOMATED_REGISTER="false"` is the default. In this case, you need to register a failed primary after a takeover manually. Use SAP tools like hanastudio or hdbnsutil.
  - For optimal automation, we recommend `AUTOMATED_REGISTER="true"`.
- Automated start of SAP HANA instances during system boot must be switched off.

You need at least SAPHanaSR version 0.151, SUSE Linux Enterprise Server for SAP Applications 12 SP1 and SAP HANA SPS09 (095) for all mentioned setups.

Important: Valid STONITH Method

Without a valid STONITH method, the complete cluster is unsupported and will not work properly.

This setup-guide focuses on the cost optimized setup. The non-productive other SAP HANA system (like QAS or TST) on the replicating node which needs to be stopped during takeover is considered in this scenario.

- Performance optimized setups for faster takeover are also supported. SUSE has published a setup-guide for performance optimized scenarios too (↑ SAP HANA SR Performance Optimized Scenario).
- Multi-tier setups (A => B -> C) are supported. You need to disable automated re-registration (AUTOMATED_REGISTER = "false"), because SAP currently does not allow two targets to be connected to one primary (↑ SAP HANA SR Multi Tier Scenario).
- Multi-tenancy (MDC) databases are supported.
  - Multi-tenancy databases could be used in combination with any other setup (performance based, cost optimized and multi-tier).
  - In MDC configurations the SAP HANA RDBMS is treated as a single system including all database containers. Therefore cluster takeover decisions are based on the complete RDBMS status independent of the status of individual containers.
  - You need SAP HANA version >= SPS10 rev3 or SPS11+, if you need to stop tenants during production and like the cluster to be able to takeover. Older SAP HANA versions are marking the system replication as failed, if you stop a tenant.

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

Besides SAP HANA, you need SAP Host Agent to be installed on your system.

3 Scope of this Documentation

This document describes how to setup the cluster to control SAP HANA in System Replication Scenarios. The document focuses on the steps 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 (WDF) and in Rot (ROT), installed on two SLES for SAP 12 SP1 systems.

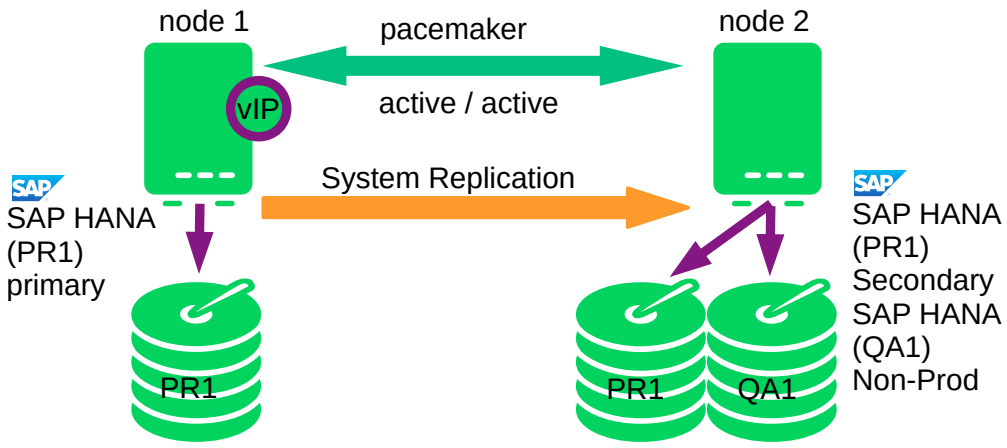

FIGURE 3.1: CLUSTER WITH SAP HANA SR - COST OPTIMIZED

TABLE 3.1: PARAMETERS USED IN THIS DOCUMENT

Parameter	Value	Role
Cluster node 1	`suse01`, `192.168.1.11`	Cluster node name and IP address.
Cluster node 2	`suse02`, `192.168.1.12`	Cluster node name and IP address.
SID	`HA1`	SAP Identifier
Instance number	`10`	Number of the SAP HANA database. For system replication also Instance Number + 1 is blocked.
SID (np)	`QAS`	SAP Identifier of the non productive database

Parameter	Value	Role
Instance number (np)	`20`	SAP Instance Number of the non productive database
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)
SBD	`/dev/disk/by-id/SBDA`	STONITH device (two for production)
HAWK Port	`7630`	
NTP Server		Address or name of your time server

4 Installing the SAP HANA Databases on both cluster nodes

Even though this document focuses on the integration of an already installed SAP HANA with already set up system replication into the pacemaker cluster, this chapter summarizes the test environment. Please always use the official documentation from SAP to install SAP HANA and to setup the system replication.

PROCEDURE 4.1: PROCEDURE OVERVIEW TO INSTALL AND CONFIGURE SAP HANA ...

1. Install the productive database on node1.
2. Install the productive database on node2.
3. Enable the system replication between both instances.
4. Reduce the memory resources of the productive system on node2.
5. Implement a SAP HANA Takeover Hook (DR provider).
6. Install the non-productive database on node2.

4.1 SAP HANA Installation

Preparation

- Read the SAP Installation and Setup Manuals available at the SAP Marketplace.
- Download the SAP HANA Software from SAP Marketplace.

PROCEDURE 4.2: INSTALLATION AND CHECKS

1. Install the SAP HANA Database as described in the SAP HANA Server Installation Guide.
2. Check if the SAP Host Agent is installed on all cluster nodes. If this SAP service is not installed, please install it now.
3. Verify that both databases are up and all processes of these databases are running correctly.

As Linux user `<sid>adm` use the command line tool **`HDB`** to get an overview of running HANA processes. The output of **`HDB`** `info` should show something like shown in the following screen-shot:

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

```
USER           PID  ...  COMMAND
ha1adm         6561 ...  -csh
ha1adm         6635 ...    \_ /bin/sh /usr/sap/HA1/HDB10/HDB info
ha1adm         6658 ...        \_ ps fx -U ha1 -o user,pid,ppid,pcpu,vsz,rss,args
ha1adm         5442 ...  sapstart pf=/hana/shared/HA1/profile/HA1_HDB10_suse02
ha1adm         5456 ...   \_ /usr/sap/HA1/HDB10/suse02/trace/hdb.sapHA1_HDB10 -d
-nw -f /usr/sap/HA1/HDB10/suse
ha1adm         5482 ...       \_ hdbnameserver
ha1adm         5551 ...       \_ hdbpreprocessor
ha1adm         5554 ...       \_ hdbcompileserver
ha1adm         5583 ...       \_ hdbindexserver
ha1adm         5586 ...       \_ hdbstatisticsserver
ha1adm         5589 ...       \_ hdbxsengine
ha1adm         5944 ...       \_ sapwebdisp_hdb
pf=/usr/sap/HA1/HDB10/suse02/wdisp/sapwebdisp.pfl -f /usr/sap/SL
ha1adm         5363 ...  /usr/sap/HA1/HDB10/exe/sapstartsrv
pf=/hana/shared/HA1/profile/HA1_HDB10_suse02 -D -u s
```

4.2 Postinstallation configuration

4.2.1 Back Up the Primary Database

Back up the primary database as described in the SAP HANA Administration Guide, Section *SAP HANA Database Backup and Recovery*. We provide an example with SQL Commands:

```
suse01:~ # hdbsql -u system -i 10 "BACKUP DATA USING FILE ('backup')"
```

If you have (for example) created a backup database user and a user key *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 cannot bring SAP HANA into a system replication configuration.

4.2.2 HDB System Replication

For more information read the *Setting Up System Replication* Section of the SAP HANA Administration Guide.

PROCEDURE 4.3: SET UP SYSTEM REPLICATION ON PRIMARY AND SECONDARY SYSTEMS

1. Enable system replication on the primary system (`hdbnsutil -sr_enable`)
2. Register the secondary system with the primary system (`hdbnsutil -sr_register`)
3. Start the secondary system.

4.2.2.1 Enable Primary Node

As Linux user `<sid>adm` enable the system replication at the primary node. You need to define a site name (like *WDF*). This site name must be unique 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=WDF
checking local nameserver:
checking for active nameserver ...
nameserver is running, proceeding ...
configuring ini files ...
successfully enabled system as primary site ...
done.
```

4.2.2.2 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: WDF
Host Mappings:
~~~~~~~~~~~~~~
done.
```

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

4.2.2.3 Enable the Secondary Node

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

```
suse02:~> HDB stop
...
suse02:~> hdbnsutil -sr_register --name=ROT \
     --remoteHost=suse01 --remoteInstance=10 \
     --replicationMode=sync --operationMode=logreplay
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. On the secondary node, the mode should be one of „SYNC“, „SYNCMEM“ or „ASYNC“. The mode depends on the sync option defined during the registration of the client.

```
suse02:~> HDB start
...
suse02:~> hdbnsutil -sr_state
...
System Replication State
~~~~~~~~~~~~~~~~~~~~~~~~

mode: sync
site id: 2
site name: ROT
active primary site: 1
...
```

The remoteHost is the primary node in our case, the remoteInstance is the database instance number (here 10).

To view the replication state of the whole SAP HANA cluster use the following command as `<sid>adm` user on the primary node.

```
suse01:~> HDBSettings.sh systemReplicationStatus.py
...
```

4.2.3 Reducing memory resources of the productive SAP HANA system on node2

Please keep in mind that parameters for memory allocation and table preload of SAP HANA must be set according to the SAP HANA documentation. A very helpful SAP SCN document could be found at: DOC-47702

Reduce the memory parameters in global.ini for the secondary instance, so the system resources are sufficient for the SLE and QAS database (see hints in this section below)

TABLE 4.1: GLOBAL.INI PARAMETERS

Parameter	Description
[ha_dr_provider_<className>]	
provider	ClassName of Python class provided by the hook script
path	Path to the hook script
execution_order	Execution order for multiple hooks
[memorymanager]	
global_allocation_limit	Maximum amount of memory used by the SAP HANA memory manager
[system_replication]	
preload_column_tables	Switch preload of tables on or off

```
 [ha_dr_provider_srTakeover]
provider = srTakeover
path = /hana/shared/srHook
execution_order = 1

[memorymanager]
global_allocation_limit = <size-in-GB>

[system_replication] ...
preload_column_tables = false
```

4.2.4 Implementing the srTakeover-Hook

The parameters added to global.ini imply that there should be a srTakeover hook installed on the second node. We provide a sample code which needs to be adapted for your environment. Currently you need to provide a user name / password combination inside the hook. We are trying to improve that in the future together with SAP at the LinuxLab.

The srTakeover-Hook is based on sample code of SAP as well as the hook provided by DELL in SAP Note 2196941.

This hook is given "as-it-is". It must be installed at node 2 as /hana/shared/srHook/srTakeover.py to undo the changes to global_allocation_limit and preload_column_tables in case of a takeover.

```
"""
Sample for a HA/DR hook provider.

When using your own code in here, please copy this file to location on /hana/shared
 outside the HANA installation.
This file will be overwritten with each hdbupd call! To configure your own changed
 version of this file, please add
to your global.ini lines similar to this:

    [ha_dr_provider_<className>]
    provider = <className>
    path = /hana/shared/haHook
    execution_order = 1

For all hooks, 0 must be returned in case of success.

Set the following variables : dbinst Instance Number [e.g. 00 - 99 ]
                              dbuser Username [ e.g. SYSTEM ]
                              dbpwd  user password [ e.g. SLES4sap ]
                              dbport port where db listens for sql [e.g 30013 or 30015]
"""

dbuser="SYSTEM"
dbpwd="manager"
dbinst="00"
dbport="30013"

stmnt1 = "ALTER SYSTEM ALTER CONFIGURATION ('global.ini','SYSTEM') UNSET
 ('memorymanager','global_allocation_limit') WITH RECONFIGURE"
stmnt2 = "ALTER SYSTEM ALTER CONFIGURATION ('global.ini','SYSTEM') UNSET
 ('system_replication','preload_column_tables') WITH RECONFIGURE"…
from hdb_ha_dr.client import HADRBase, Helper
```

```
import os, time, dbapi

class srTakeover(HADRBase):

    def __init__(self, *args, **kwargs):
        # delegate construction to base class
        super(srTakeover, self).__init__(*args, **kwargs)

    def about(self):
        return {"provider_company" :       "SUSE",
                "provider_name" :          "srTakeover", # provider name = class name
                "provider_description" :   "Replication takeover script to set parameters
 to default.",
                "provider_version" :       "1.0"}

    def startup(self, hostname, storage_partition, system_replication_mode, **kwargs):
        self.tracer.debug("enter startup hook; %s" % locals())
        self.tracer.debug(self.config.toString())
        self.tracer.info("leave startup hook")
        return 0

    def shutdown(self, hostname, storage_partition, system_replication_mode, **kwargs):
        self.tracer.debug("enter shutdown hook; %s" % locals())
        self.tracer.debug(self.config.toString())
        self.tracer.info("leave shutdown hook")
        return 0

    def failover(self, hostname, storage_partition, system_replication_mode, **kwargs):
        self.tracer.debug("enter failover hook; %s" % locals())
        self.tracer.debug(self.config.toString())
        self.tracer.info("leave failover hook")
        return 0

    def stonith(self, failingHost, **kwargs):
        self.tracer.debug("enter stonith hook; %s" % locals())
        self.tracer.debug(self.config.toString())
        # e.g. stonith of params["failed_host"]
        # e-g- set vIP active
        self.tracer.info("leave stonith hook")
        return 0

    def preTakeover(self, isForce, **kwargs):
        """Pre takeover hook."""
        self.tracer.info("%s.preTakeover method called with isForce=%s" %
 (self.__class__.__name__, isForce))
        if not isForce:
            # run pre takeover code
```

```
            # run pre-check, return != 0 in case of error => will abort takeover
            return 0
        else:
            # possible force-takeover only code
            # usually nothing to do here
            return 0
    def postTakeover(self, rc, **kwargs):
        """Post takeover hook."""
        self.tracer.info("%s.postTakeover method called with rc=%s" %
(self.__class__.__name__, rc))

        if rc == 0:
            # normal takeover succeeded
            conn = dbapi.connect('localhost',dbport,dbuser,dbpwd)
            cursor = conn.cursor()
            cursor.execute(stmnt1)
            cursor.execute(stmnt2)
            return 0

        elif rc == 1:
            # waiting for force takeover
            conn = dbapi.connect('localhost',dbport,dbuser,dbpwd)
            cursor = conn.cursor()
            cursor.execute(stmnt1)
            cursor.execute(stmnt2)
            return 0
        elif rc == 2:
            # error, something went wrong
            return 0
```

In the same directory (/hana/shared/srHook) you need to install some python files from the SAP HANA client software to enable the hook to run the database connect and sql queries.

dbapi.py, _init_.py, resultrow.py

After changing the global.ini on node 2 and implementing the srTakeover hook you should start the productive HANA database secondary and check if the parameters are working.

4.2.5 Manual SAP HANA takeover test

Test if takeover can be achieved by administrator interaction (i.e. manually). After the sr_takeover, check if the global.ini parameters on the node2 have been changed by the sr-Takeover hook call (for example with the SAP HANA Administration Console).

4.2.6 Manual re-establish SAP HANA SR to original state

1. Bring the systems back to the original state:
2. takeover SLE to node 1
3. wait till sync state is active
4. stop SLE on node 2
5. re-register node 2 as secondary
6. reconfigure global.ini
7. start SLE on node2

Please note that while you are switching back to original system replication direction you must also re-change the global.ini as the srTakeover hook should have deleted the parameters for memory limitation and table preload.

4.2.7 Install the non-productive SAP HANA database (QAS)

Install the non-productive SAP HANA database (QAS) on node 2. Please keep in mind that parameters for memory usage of SAP HANA must be set according to the SAP HANA documentation of SAP and you have to provide separate storage for the additional SAP HANA database.

See also: SAP_HANA_Administration_Guide_en.pdf (Chapter 5).

A maybe feasible starting point to tune the resource limitation is that the secondary on the node 2 should hold 10% of the resources, but at least enough memory to start without table preload. The resources for the QAS system must also be limited not to get into conflict with the secondary productive database. Exact values for resource optimization are heavily depending on the customers workload and project situation and should be discussed together with SAP.

The sizing is discussed in SAP Document 47702, page 7ff.

Before installing the QAS system, we recommend you to stop the productive secondary to avoid resource bottle necks, because the memory consumption of the QAS system could only be reduced in a post installation task.

PROCEDURE 4.4: POST-INSTALLATION TASKS FOR THE QAS DATABASE

1. Stop QAS database

2. SAPHOSTAGENT needs to be a current version (SAP Note 2138524 - SAPHOSTAGENT on SAP HANA: ListDatabases function: Error: Command execution failed)
3. SAP HANA Client must be installed
4. Creating a key for SAPHOSTAGENT Monitoring (as qasadm) see also SAP notes 1625203 and 2023587):

```
suse02:~> hdbsql -u system -i 10 -nlocalhost:31013 'CREATE USER SC PASSWORD
 L1nuxLab'
suse02:~> hdbsql -u system -i 10 -nlocalhost:31013 'GRANT MONITORING TO SC'
suse02:~> hdbsql -u system -i 10 -nlocalhost:31013 'ALTER USER SC  DISABLE PASSWORD
 LIFETIME'
suse02:~> hdbsql -u sc -i 10 -nlocalhost:31013 'SELECT * FROM DUMMY'
suse02:~> hdbsql -u sc -p  L1nuxLab -i 10 -nlocalhost:31013 'SELECT * FROM DUMMY'
suse02:~> hdbuserstore SET QASSAPDBCTRL localhost:31013 SC  L1nuxLab
suse02:~> hdbsql -U QASSAPDBCTRL 'select * from Dummy'
```

5. Adapt global.ini to limit memory resources

```
[memorymanager]
global_allocation_limit = <size-in-GB>
```

6. Start QAS and secondary SAP HANA database
7. Check if everything works as configured and expected.

4.2.8 Shutdown all SAP HANA databases

To stop the SAP HANA databases login on node 1 (suse01) as *sid-of-prd*adm and call "HDB stop". On node 2 we need to stop the productive and the QAS database. So you need to login and run "HDB stop" as *sid-of-prd* adm and as *sid-of-qas* adm. In our setup we have SLE for the production and QAS for the non-productive database system.

```
suse01:~# su - sleadm -c "HDB stop"
suse02:~# su - sleadm -c "HDB stop"
suse02:~# su - qasadm -c "HDB stop"
```

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.

PROCEDURE 5.1: INSTALL AND CONFIGURE THE CLUSTER

1. Install the `High Avaiability` pattern and the `SAPHanaSR Resource Agents`.
2. Basic Cluster Configuration.
3. Configure Cluster Properties and Resources.

5.1 Installation

If have not already done, install the pattern `High Availability` on both nodes. To do so, for example, use **`zypper`**:

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

Now the `Resource Agents` for controlling the SAP HANA system replication needs to be installed at both cluster nodes.

```
suse01:~> zypper in SAPHanaSR SAPHanaSR-doc
```

For more information, see *Installation and Basic Setup*, ↑SUSE Linux Enterprise High Availability Extension.

5.2 Basic Cluster Configuration

The first step is to set up the basic cluster framework. For convenience, use YaST2 or the ha-cluster-init script. It is strongly recommended to add a second corosync ring, change to UCAST communication and adjust the timeout values to your environment.

5.2.1 Initial cluster setup using `ha-cluster-init`

For more information, see *Automatic Cluster Setup*, ↑SUSE Linux Enterprise High Availability Extension.

Create an initial setup, using ha-cluster-init command and follow the dialogs. This is only to be done on the first cluster node.

```
suse01:~> ha-cluster-init
```

This command configures 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

Important: Change the password of the user `hacluster`

As requested by ha-cluster-init, change the password of the user `hacluster`.

5.2.2 Adapting the Corosync and sbd Configuration

It is recommended to add a second corosync ring and to change to UCAST communication as described in *3.2.1 of Best Practices for SAP on SUSE Linux Enterprise*.

Stop the already running cluster by using `systemctl stop pacemaker`. After setup of the corosync config and sbd parameters, start the cluster again.

5.2.2.1 Corosync Configuration

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

```
totem {
```

```
    ...

 crypto_cipher: none
 crypto_hash: none

 interface {
  ringnumber: 0
  bindnetaddr: 192.168.1.0
  mcastport: 5405
  ttl:  1
 }
    #Transport protocol
 transport:      udpu

}
nodelist {
        node {
        #ring0 address
        ring0_addr:     192.168.1.11

        }
        node {
        #ring0 address
        ring0_addr:     192.168.1.12

        }
 }
```

5.2.2.2 Configure a watchdog

It is highly recommended that you configure your Linux system to load a watchdog driver with hardware assistance (as is available on most modern systems), such as hpwdt, iTCO_wdt, or others. As a fall-back, you can use the softdog module.

No other software must access the watchdog timer; it can only be accessed by one process at any given time. Some hardware vendors ship systems management software that use the watchdog for system resets (f.e. HP ASR daemon). Such software has to be disabled if the watchdog is to be used by SBD.

5.2.2.3 Adapting sbd config

You can skip this section, if you do not have any sbd devices, but be sure to implement an other supported fencing mechanism. If you use the newest updates of the pacemaker packages from the SUSE maintenance 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.

Please see the sbd man page for further details of the parameters -S <n>, -P, -W.

TABLE 5.1: SBD OPTIONS

Parameter	Description
-W	Use watchdog. It is mandatory to use a watchdog. SBD does not work reliable without watchdog. Please refer to the SLES manual and SUSE TIDs 7016880 for setting up a watchdog. This is equivalent to `SBD_WATCHDOG="yes"`
-S 1	Start mode. If set to one, sbd will only start if the node was previously shutdown cleanly or if the slot is empty. This is equivalent to `SBD_STARTMODE="clean"`
-P	Check Pacemaker quorum and node health. This is equivalent to `SBD_PACEMAKER="yes"`

In the following, replace /dev/disk/by-id/SBDA and /dev/disk/by-id/SBDB by your real sbd device names.

```
# /etc/sysconfig/sbd
SBD_DEVICE="/dev/disk/by-id/SBDA;/dev/disk/by-id/SBDB"
SBD_WATCHDOG="yes"
SBD_PACEMAKER="yes"
SBD_STARTMODE="clean"
SBD_OPTS=""
```

Note

This equates to the SUSE Linux Enterprise 11 settings **SBD_OPTS="-W -P -S 1"**.

5.2.2.4 Verify the sbd device

You can skip this section if you do not have any sbd devices, but make sure to implement a supported fencing mechanism.

It's a good practice to check, if the sbd device can be accessed from both nodes and does contain valid records. Check this for all devices configured in /etc/sysconfig/sbd.

```
suse01:~ # sbd -d /dev/disk/by-id/SBDA dump
==Dumping header on disk /dev/disk/by-id/SBDA
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/disk/by-id/SBDA is dumped
```

The timeout values in our sample are only start values, which need to be tuned to your environment.

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/disk/by-id/SBDA list
0     suse01      clear
1     suse02      clear
```

For more information on SBD configuration parameters, please read section *Storage-based Fencing*, ↑SUSE Linux Enterprise High Availability Extension and TIDs 7016880 and 7008216.

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

5.2.3 Cluster configuration on the second node

The second node of the two nodes cluster could be integrated by starting the command **ha-cluster-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.

```
# ha-cluster-join
```

5.2.4 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:~ # systemctl status pacemaker
suse02:~ # systemctl status pacemaker
suse01:~ # systemctl start pacemaker
suse02:~ # systemctl start pacemaker
```

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

```
# crm_mon -r
```

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

5.3 Configure Cluster Properties and Resources

This section describes how to configure constraints, resources, bootstrap and STONITH using the crm configure shell command as described in section *Configuring and Managing Cluster Resources (Command Line)*, ↑SUSE Linux Enterprise High Availability Extension.

Use the command **crm** to add the objects to CRM. Copy the following examples to a local file, edit the file and than load the configuration to the CIB:

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

5.3.1 Cluster bootstrap and more

The first example defines the cluster bootstrap options, the resource and operation defaults. The stonith-timeout should be greater than 1.2 times the SBD msgwait timeout.

```
suse01:~ # 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" \
                 timeout="600"
```

Now we add the configuration to the cluster.

```
suse01:~ # crm configure load update crm-bs.txt
```

5.3.2 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 \
    params pcmk_delay_max="15" \
```

```
    op monitor interval="15" timeout="15"
```

Again we add the configuration to the cluster.

```
suse01:~ # crm configure load update crm-sbd.txt
```

For alternative IPMI/ILO setup see our cluster product documentation. An example for an IPMI STONITH resource can be found in the appendix (section 7.5) of this document.

5.3.3 Using IPMI as fencing mechanism

For alternative IPMI/ILO setup see our cluster product documentation. An example for an IPMI STONITH resource can be found in the appendix (section 7.5) of this document.

To use IPMI the remote management boards must be compatible with the IPMI standard.

For the IPMI based fencing you need to configure a primitive per cluster node. Each resource is responsible to fence exactly one cluster node. You need to adapt the IP addresses and login user / password of the remote management boards to the STONITH resource agent. We recommend to create a special STONITH user instead of providing root access to the management board. Location rules must guarantee that a host should never run its own STONITH resource.

5.3.4 Using other fencing mechanisms

We recommend to use SBD (best practice) or IPMI (second choice) as STONITH mechanism. The SUSE Linux Enterprise High Availability product also supports additional fencing mechanism not covered here.

For further information about fencing, see SUSE Linux Enterprise High Availability Guide.

5.3.5 SAPHanaTopology

Next we define the group of resources needed, before the HANA instances can be started. Prepare the changes in a text file, for example crm-saphanatop.txt, and load these with the command: **`crm configure load update crm-saphanatop.txt`**

```
# vi crm-saphanatop.txt
# enter the following to crm-saphanatop.txt
primitive rsc_SAPHanaTopology_HA1_HDB10 ocf:suse:SAPHanaTopology \
        operations $id="rsc_sap2_HA1_HDB10-operations" \
```

```
        op monitor interval="10" timeout="600" \
        op start interval="0" timeout="600" \
        op stop interval="0" timeout="300" \
        params SID="HA1" InstanceNumber="10"
clone cln_SAPHanaTopology_HA1_HDB10 rsc_SAPHanaTopology_HA1_HDB10 \
        meta clone-node-max="1" interleave="true"
```

Additional information about all parameters could be found with the command **man** ocf_suse_SAPHanaTopology

Again we add the configuration to the cluster.

```
suse01:~ # crm configure load update crm-saphanatop.txt
```

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

5.3.6 SAPHana

Next we define the group of resources needed, before the HANA instances can be started. Edit the changes in a text file, for example crm-saphana.txt and load these with the command: **crm configure load update crm-saphana.txt**

TABLE 5.2: TYPICAL RESOURCE AGENT PARAMETER SETTINGS FOR DIFFERENT SCENARIOS

Parameter	Performance Optimized	Cost Optimized	Multi-Tier
PREFER_SITE_TAKEOVER	true	false	false / true
AUTOMATED_REGISTER	false / true	false / true	false
DUPLICATE_PRIMARY_TIMEOUT	7200	7200	7200

TABLE 5.3: DESCRIPTION OF IMPORTANT RESOURCE AGENT PARAMETERS

Parameter	Description
PREFER_SITE_TAKEOVER	Defines whether RA should prefere to takeover to the seconadry instance instead of restarting the failed primary locally.

Parameter	Description
AUTOMATED_REGISTER	Defines whether a former primary should be automatically registered to be secondary of the new primary. With this parameter you can adapt the level of system replication automation. If set to `false` the former primary must be manually registered. The cluster will not start this SAP HANA RDBMS till it is registerd to avoid double primary up situations.
DUPLICATE_PRIMARY_TIMEOUT	Time difference needed between two primary time stamps, if a dual-primary situation occurs. If the time difference is less than the time gap, than the cluster hold one or both instances in a "WAITING" status. This is to give a admin the chance to react on a failover. If the complete node of the former primary crashed, the former primary will be registered after the time difference is passed. If "only" the SAP HANA RDBMS has crashed, then the former primary will be registered immediately. After this registration to the new primary all data will be overwritten by the system replication.

Additional information about all parameters could be found with the command `man ocf_suse_SAPHana`

```
# vi crm-saphana.txt
# enter the following to crm-saphana.txt
primitive rsc_SAPHana_HA1_HDB10 ocf:suse:SAPHana \
        operations $id="rsc_sap_HA1_HDB10-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="HA1" InstanceNumber="10" PREFER_SITE_TAKEOVER="true" \
        DUPLICATE_PRIMARY_TIMEOUT="7200" AUTOMATED_REGISTER="false"
ms msl_SAPHana_HA1_HDB10 rsc_SAPHana_HA1_HDB10 \
```

```
        meta clone-max="2" clone-node-max="1" interleave="true"
```

We add the configuration to the cluster.

```
suse01:~ # crm configure load update crm-saphana.txt
```

The most important parameters here are again SID and InstanceNumber. Beside these parameters the timeout values for the operations (start, promote, monitors, stop) are typical tuneables.

5.3.7 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_HA1_HDB10 ocf:heartbeat:IPaddr2 \
        operations $id="rsc_ip_HA1_HDB10-operations" \
        op monitor interval="10s" timeout="20s" \
        params ip="192.168.1.20"
```

We load the file to the cluster.

```
suse01:~ # 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.

5.3.8 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_HA1_HDB10 3000: rsc_ip_HA1_HDB10:Started \
    msl_SAPHana_HA1_HDB10:Master
order ord_SAPHana_HA1_HDB10  Optional: cln_SAPHanaTopology_HA1_HDB10 \
    msl_SAPHana_HA1_HDB10
```

We load the file to the cluster.

```
suse01:~ # crm configure load update crm-cs.txt
```

5.3.9 Add the cluster resource for the non-productive SAP HANA database

This snippet describes the cluster configuration part for the SAP HANA QAS database:

We refer to the cluster nodes as suse01 and suse02 respectively:

```
  primitive rsc_SAP_QAS_HDB20 ocf:heartbeat:SAPDatabase \
    params DBTYPE="HDB" SID="QAS" InstanceNumber="20" \
    MONITOR_SERVICES="hdbindexserver|hdbnameserver" \
    op start interval="0" timeout="600" \
    op monitor interval="120" timeout="700" \
    op stop interval="0" timeout="300" \
    meta priority="100"
```

5.3.10 Adding the cluster rules for the automatic shutdown of SAP HANA QAS

We refer to the cluster nodes as suse01 and suse02 respectively:

```
We refer to the cluster nodes as suse01 and suse02 respectively:
location loc_QAS_never_on_suse01 rsc_SAP_QAS_HDB20 -inf: suse01

colocation col_QAS_never_with_HA1ip -inf: rsc_SAP_QAS_HDB20:Started \
    rsc_ip_HA1_HDB10

order ord_QASstop_before_HA1-promote inf: rsc_SAP_QAS_HDB20:stop \
    msl_SAPHana_HA1_HDB10:promote
```

6 Testing the Cluster

As with any cluster testing is crucial. Please make sure that all test cases derived from customer expectations are implemented and passed fully. Else the project is likely to fail in production use

TABLE 6.1: TEST CASE EXAMPLES

Nr	Compo-nent	Descripion	Status
1	Primary	D: Public LAN failure: The loss of connectivity of the node on the Public LAN. Simulated in this case by shutting down the NIC handling this traffic. E: Cluster Reaction: 1. Virtual IP Address off-lined, “stopped”. 2. Promotion of Replication HDB instance on suse02. 3. Promotion forces “Stop” of none-REP HDB [QAS] 4. REP-HDB has srTakeover action performed and is promoted from Slave to Master “live” HDB instance. 5. Successful REP-HDB state change to Master restarts Virtual IP-address.y Recover Action: 1. Successfully Restore Network Connectivity to failed node. 2. Within “planned downtime” reconfigure system to initial state. A: As expected results	Pass
2	Primary [suse01]	**D:** System Freeze [Simulated Kernel Panic] Loss of communications by CRM [Cluster Resource Manager] with this node. Simulated by `kill -9` on the corosync daemon. **E:** Cluster Reaction: 1. Secondary node issues stonith to Primary node – Result: Primary node is powerd off. 2. Power-off results Virtual IP address “Stopped”.	Pass

Nr	Compo-nent	Descripion	Status
		3. Promotion of Replication HDB instance on suse02. 4. Promotion forces “Stop” of none-REP HDB [QAS]. 5. REP-HDB has srTakeover action performed and is promoted from Slave to Master “live” HDB instance. 6. Successful REP-HDB state change to Master restarts Virtual IP-address. R Recover Action: 1. Successfully fix and restart primary node. 2. Restart cluster software, if not automatic. 3. Within “planned downtime” reconfigure system to initial state. A: As Expected results.	

7 Administration

7.1 Do and Don't

In your project, you should:

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

In your project, avoid:

- Rapidly changing/changing back cluster configuration, such as: Setting nodes to standby and online again or stopping/starting the master/slave resource.
- Creating a cluster without proper time synchronization or unstable name resolutions for hosts, users and groups
- Adding location rules for the clone, master/slave or IP resource. Only location rules mentioned in this setup guide are allowed.
- As "migrating" or "moving" resources in crm-shell, HAWK or other tools would add client-prefer location rules this activities are completely forbidden.

7.2 Monitoring and Tools

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

7.2.1 HAWK – Cluster Status and more

You can use an internet browser to check the cluster status.

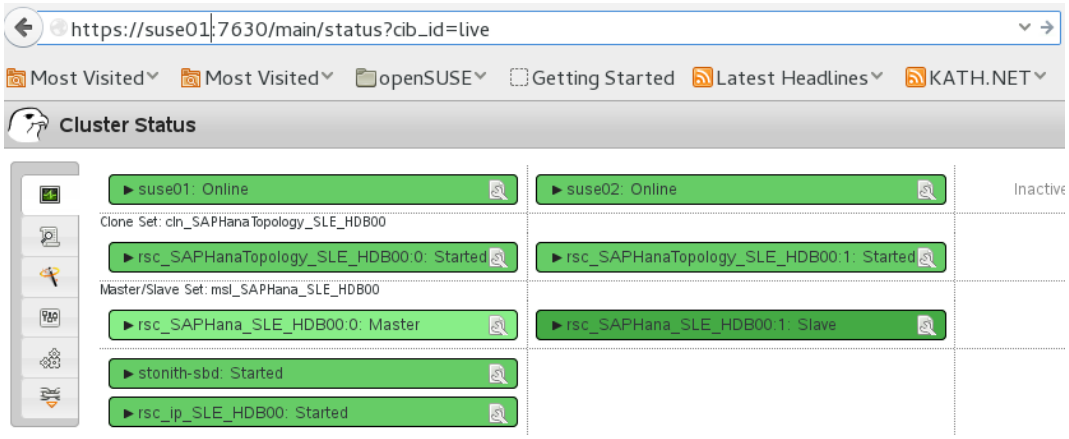

FIGURE 7.1: CLUSTER STATUS IN HAWK

If you set up the cluster using ha-cluster-init and you have installed all packages as described above, your system will provide a very useful web interface. You can use this graphical web interface to get an overview of the complete cluster status, doing administrative tasks or even configure resources and cluster bootstrap parameters. Read our product manuals for a complete documentation of this powerful user interface.

7.2.2 SAP HANA Studio

Database-specific administration and checks can be done with SAP HANA studio.

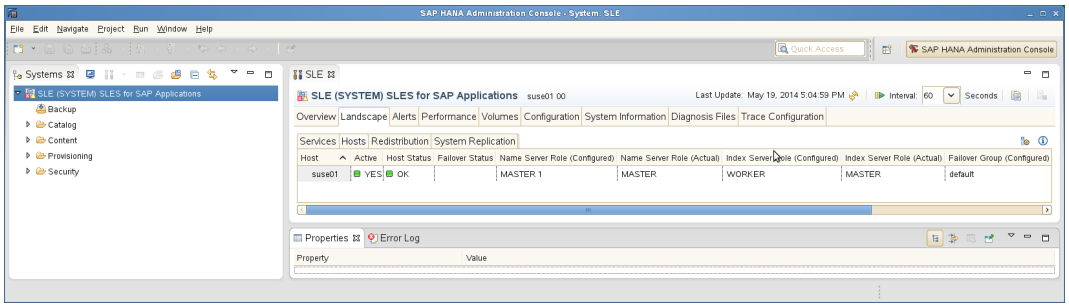

FIGURE 7.2: SAP HANA STUDIO – LANDSCAPE

7.2.3 Cluster Command-Line Tools

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

```
Last updated: Fri Jun 17 14:14:08 2016
Last change: Fri Jun 17 14:13:09 2016 by root via crm_attribute on suse01
Stack: corosync
Current DC: suse02 (version 1.1.13-14.7-6f22ad7) - partition with quorum
2 nodes and 6 resources configured

Online: [ suse01 suse02 ]

Full list of resources:

 stonith-sbd    (stonith:external/sbd): Started suse01
 Clone Set: cln_SAPHanaTopology_HA1_HDB10 [rsc_SAPHanaTopology_HA1_HDB10]
     Started: [ suse01 suse02 ]
 Master/Slave Set: msl_SAPHana_HA1_HDB10 [rsc_SAPHana_HA1_HDB10]
     Masters: [ suse01 ]
     Slaves: [ suse02 ]
 rsc_ip_HA1_HDB10       (ocf::heartbeat:IPaddr2):       Started suse01
```

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 **`SAPHanaSR-showAttr`**. The internal values, storage place and their parameter names may change in the next versions. The command **`SAPHanaSR-showAttr`** 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 **`SAPHanaSR-showAttr`** is a test program only and should not be used for automated system monitoring.

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

7.2.4 SAP HANA LandscapeHostConfiguration

To check the status of a SAPHana database and to figure out if the cluster should react, you can use the script landscapeHostConfiguration to be called as Linux user `<sid>adm`.

```
suse01:~> HDBSettings.sh landscapeHostConfiguration.py
| Host   | Host   | ... NameServer   | NameServer  | IndexServer | IndexServer |
```

```
|        | Active | ... Config Role  | Actual Role | Config Role | Actual Role |
| ------ | ------ | ... ------------ | ----------- | ----------- | ----------- |
| suse01 | yes    | ... 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:

TABLE 7.1: INTERPRETATION OF RETURN CODES

Return Code	Interpretation
4	SAP HANA database is up and OK. The cluster does interpret this as a correctly running database.
3	SAP HANA database is up and in status info. The cluster does interpret this as a correctly running database.
2	SAP HANA database is up and in status warning. The cluster does interpret this as a correctly running database.
1	SAP HANA database is down. If the database should be up and is not down by intention, this could trigger a takeover.
0	Internal Script Error – to be ignored.

7.3 Maintenance

It is highly recommended to register your systems to either a local SUSE Manager or SMT or remote with SUSE Customer Center to be able to receive updates to the OS or HAE.

7.3.1 Updating the OS and Cluster

Update of SUSE Linux Enterprise Server for SAP Applications packages including cluster software you should follow the rolling update procedure defined in the product documentation of SUSE Linux Enterprise High Availability *Upgrading Your Cluster and Updating Software Packages*, ↑ High Availability Administration Guide.

7.3.2 Updating SAP HANA

For updating SAP HANA database systems in system replication you need to follow the defined SAP processes. This section describes the steps to be done before and after the update procedure to get the system replication automated again.

PROCEDURE 7.1: TOP LEVEL STEPS TO UPDATING SAP HANA IN THE CLUSTER

1. Prepare the cluster not to react on the maintenance work to be done on the SAP HANA database systems. Set the master/slave resource to be unmanaged and the cluster nodes in maintenance mode.
   For the master/slave resource set the unmange status: `crm resource unmanage master-slave-resource`
   For all cluster nodes set the ready status: `crm node maintenance node`.
2. Complete the SAP Update process for both SAP HANA database systems. This process is described by SAP.
3. After the SAP HANA update is complete on both sites, tell the cluster about the end of the maintenance process.
   For all cluster nodes set the ready status: `crm node ready node`.
4. As we expect the primary/secondary roles to be exchanged after the maintenance, tell the cluster to forget about this states and to reprobe the updated SAP HANA database systems.
   `crm resource cleanup master-slave-resource`.
5. In the last step we tell the cluster to manage the SAP HANA database systems again.
   `crm resource manage master-slave-resource`.

7.3.3 Migrating a HANA primary

In the following procedures we assume the primary to be running on node1 and the secondary on node2. The goal is to "exchange" the roles of the nodes, so finally the primary should run on node2 and the secondary should run on node1.

There are different methods to get the excahnge of the roles done, the following procedure shows how to tell the cluster to "accept" a role change done with native HANA commands.

PROCEDURE 7.2: MIGRATING A HANA PRIMARY WITH UNMANAGED MASTER/SLAVE RESOURCE

1. Set the master/slave resource to be unmanaged. This could be done on any cluster node.

`crm resource unmanage` *master-slave-resource-name*

For the cost optimized scanario you also need to set the primitive of the non-productive database to be unmanaged.

2. Stop the primary SAP HANA database system. Enter the command in our example on node1 as user *sid*adm.

   `HDB stop`

3. For the cost optimized scanario you also need to stop the non-productive database to allow the takeover. Enter the command in our example on node2 as the *sid*adm of the non-productive system.

   `HDB stop`

4. Start the takeover process on the secondary SAP HANA database system. Enter the command in our example on node2 as user *sid*adm.

   `hdbnsutil -sr_takeover`

5. Register the former primary to become the new secondary. Enter the command in our example on node1 as user *sid*adm.

   `hdbnsutil -sr_register --remoteHost=suse02 --remoteInstance=10 --replicationMode=sync --name=WDF --operationMode=logreplay`

6. Start the new secondary SAP HANA database system. Enter the command in our example on node1 as user *sid*adm.

   `HDB start`

   Wait some time till SAPHanaSR-showAttr shows both SAP HANA database systems to be up again (field roles must start with the digit 4).

7. Tell the cluster to forget about the former master/slave roles and to re-monitor the failed master. The command could be submitted on any cluster node as user root.

   `crm resource cleanup` *master-slave-resource-name*

   For the cost optimized scanario you also need to set the primitive of the non-productive database to be cleaned-up.

8. Set the master/slave resource to the status managed again. The command could be submitted on any cluster node as user root.

   `crm resource manage` *master-slave-resource-name*

   For the cost optimized scanario you also need to set the primitive of the non-productive database to be managed. Please wait until the master/slave is full up and the primary is promoted (master). You can check this using the command `crm_mon -1r`

Now we explain how to use the cluster to partially automize the migration.

PROCEDURE 7.3: MIGRATING A HANA PRIMARY USING CLUSTER MIGRATION COMMANDS

1. Create a migrate awqy from this node rule.
   crm resource migrate *sapHanaResource* **force**
   Because of we used the migrate away rule the cluster will stop the current primary and run a promote on the secondary site, if the system replication was in sync before. You should not migrate the primary, if the status of the system replication is not in sync (SFAIL).
   Wait till the secondary has completly taken over to be the new primary.
2. If you have setup `AUTOMATED_REGISTER="true"` you can skip this step. In other cases you now need to regsiter the old primary. Enter the command in our example on node1 as user *sid*adm.
   hdbnsutil -sr_register --remoteHost=suse02 --remoteInstance=10 --replicationMode=sync --name=WDF --operationMode=logreplay
3. Unmigrate the resource to allow the cluster to start the new secondary.
   crm resource unmigrate *sapHanaRsource*

PROCEDURE 7.4: MIGRATING A HANA PRIMARY USING CLUSTER NODE STATUS STANDBY

1. Set the primary node to be standby.
   crm node standby *suse01*
   The cluster will stop the primary SAP HANA database and if the system replication was in sync process the takeover on the secondary site
   Wait till the former secondary has completly taken over to be the new primary.
2. If you have setup `AUTOMATED_REGISTER="true"` you can skip this step. In other cases you now need to regsiter the old primary. Enter the command in our example on node1 as user *sid*adm.
   hdbnsutil -sr_register --remoteHost=suse02 --remoteInstance=10 --replicationMode=sync --name=WDF --operationMode=logreplay
3. Set the standby node to be online again.
   crm node online *suse01*

A Useful Links, Manuals, and SAP Notes

A.1 SUSE Best Practices and More

- Best Practices for SAP on SUSE Linux Enterprise: https://www.suse.com/products/sles-for-sap/resource-library/sap-best-practices.html
- Fail-Safe Operation of SAP HANA®: SUSE Extends Its High-Availability Solution: http://scn.sap.com/community/hana-in-memory/blog/2014/04/04/fail-safe-operation-of-sap-hana-suse-extends-its-high-availability-solution
- HOW TO SET UP SAPHanaSR IN THE COST OPTIMIZED SAP HANA SR SCENARIO http://scn.sap.com/docs/DOC-65899

A.2 SUSE Product Documentation

The SUSE product manuals and documentation can be downloaded at www.suse.com/documentation.

- Current online documentation of SLES for SAP https://www.suse.com/documentation/sles_for_sap/
- Current online documentation of SUSE Linux Enterprise High Availability Extension https://www.suse.com/documentation/sle_ha/
- Tuning guide for SUSE Linux Enterprise Server https://www.suse.com/documentation/sles11/book_sle_tuning/data/book_sle_tuning.html
- Storage admin guide for SUSE Linux Enterprise Server https://www.suse.com/documentation/sles11/stor_admin/data/bookinfo.html
- Release notes https://www.suse.com/releasenotes/
- TID Estimate correct multipath timeout http://www.suse.com/support/kb/doc.php?id=7008216
- TID How to load the correct watchdog kernel module http://www.suse.com/support/kb/doc.php?id=7016880

- TID Performance issues after upgrade from SLES11 SP1 to SP2 or SP3 http://www.suse.com/support/kb/doc.php?id=7011982
- TID Addressing file system performance issues on NUMA machines http://www.suse.com/support/kb/doc.php?id=7008919
- TID Low write performance on SLES 11 servers with large RAM https://www.suse.com/support/kb/doc.php?id=7010287
- TID Overcommit Memory in SLES https://www.suse.com/support/kb/doc.php?id=7002775
- SLES technical information https://www.suse.com/products/server/technical-information/
- XFS file system https://www.suse.com/communities/conversations/xfs-the-file-system-of-choice/

A.3 SAP Product Documentation

- SAP HANA Installation and Update Guide http://help.sap.com/hana/SAP_HANA_Server_Installation_Guide_en.pdf
- SAP HANA Administration Guide http://help.sap.com/hana/SAP_HANA_Administration_Guide_en.pdf

A.4 SAP Notes

- 1310037 (http://service.sap.com/sap/support/notes/1310037) SUSE LINUX Enterprise Server 11: Installation notes
- 1824819 (http://service.sap.com/sap/support/notes/2240716) SAP HANA DB: Recommended OS settings for SLES 11 / SLES for SAP Applications 11 SP4
- 1876398 (http://service.sap.com/sap/support/notes/1876398) Network configuration for System Replication in HANA SP6
- 611361 (http://service.sap.com/sap/support/notes/611361) Hostnames of SAP servers
- 1275776 (http://service.sap.com/sap/support/notes/1275776) Preparing SLES for Sap Environments

- 1514967 (http://service.sap.com/sap/support/notes/1514967) SAP HANA: Central Note
- 1523337 (http://service.sap.com/sap/support/notes/1523337) SAP In-Memory Database 1.0: Central Note
- 1501701 (http://service.sap.com/sap/support/notes/1501701) Single Computing Unit Performance and Sizing
- 1944799 (http://service.sap.com/sap/support/notes/1944799) SAP HANA Guidelines for SLES Operating System Installation
- 1954788 (http://service.sap.com/sap/support/notes/1954788) SAP HANA DB: Recommended OS settings for SLES 11 / SLES for SAP Applications 11 SP3
- 1855805 (http://service.sap.com/sap/support/notes/1855805) Recommended SLES 11 packages for HANA support on OS level
- 1890444 (http://service.sap.com/sap/support/notes/1890444) Slow HANA system due to CPU power save mode
- 1867783 (http://service.sap.com/sap/support/notes/1867783) XFS Data Inconsistency Bug with SLES 11 SP2
- 1888072 (http://service.sap.com/sap/support/notes/1888072) SAP HANA DB: Indexserver crash in strcmp sse42
- 1846872 (http://service.sap.com/sap/support/notes/1846872) "No space left on device" error reported from HANA

B Examples

B.1 Example `ha-cluster-init` Configuration

```
suse01:~ # ha-cluster-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/disk/by-id/SBDA
All data on /dev/disk/by-id/SBDA 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://192.168.1.11:7630/
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
```

B.2 Example Cluster Configuration

The following complete crm configuration is for a two-node cluster (suse01, suse02) and a SAP HANA database with SID HA1 and instance number 10. The virtual IP address in the example is 192.168.1.20

```
node suse01
node suse02

primitive rsc_SAPHanaTopology_HA1_HDB10 ocf:suse:SAPHanaTopology \
        operations $id="rsc_sap2_HA1_HDB10-operations" \
        op monitor interval="10" timeout="300" \
        op start interval="0" timeout="300" \
        op stop interval="0" timeout="300" \
        params SID="HA1" InstanceNumber="10"
primitive rsc_SAPHana_HA1_HDB10 ocf:suse:SAPHana \
        operations $id="rsc_sap_HA1_HDB10-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="HA1" InstanceNumber="10" PREFER_SITE_TAKEOVER="false"
DUPLICATE_PRIMARY_TIMEOUT="7200" AUTOMATED_REGISTER=“false“
primitive rsc_ip_HA1_HDB10 ocf:heartbeat:IPaddr2 \
        operations $id="rsc_ip_HA1_HDB10-operations" \
        op monitor interval="10s" timeout="20s" \
        params ip="192.168.1.20"
primitive stonith-sbd stonith:external/sbd \
        params pcmk_delay_max="15" \
        op monitor interval="15" timeout=“15“
primitive rsc_SAPDatabase_QAS_HDB20 ocf:heartbeat:SAPDatabase \
           params DBTYPE="HDB" SID="QAS" InstanceNumber="20" \
```

```
           MONITOR_SERVICES="hdbindexserver|hdbnameserver" \
           op start interval="0" timeout="600" \
           op monitor interval="120" timeout="700" \
           op stop interval="0" timeout="300" \
           meta priority="100"
ms msl_SAPHana_HA1_HDB10 rsc_SAPHana_HA1_HDB10 \
        meta clone-max="2" clone-node-max="1" interleave="true"
clone cln_SAPHanaTopology_HA1_HDB10 rsc_SAPHanaTopology_HA1_HDB10 \
        meta clone-node-max="1" interleave="true"
colocation col_saphana_ip_HA1_HDB10 2000: \
        rsc_ip_HA1_HDB10:Started msl_SAPHana_HA1_HDB10:Master
order ord_SAPHana_HA1_HDB10 2000: \
        cln_SAPHanaTopology_HA1_HDB10 msl_SAPHana_HA1_HDB10
location loc_QAS_never_on_suse01 rsc_SAP_QAS_HDB20 -inf: suse01
colocation col_QAS_never_with_HA1-ip -inf: rsc_SAPHana_QAS_HDB20:Started \
        rsc_ip_HA110
order ord_QAS_stop_before_HA1-promote inf: rsc_SAPHana_QAS_HDB20:stop \
        msl_SAPHana_HA1_HDB10:promote
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" \
        timeout="600"
```

B.3 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.

```
# Please read the corosync.conf.5 manual page

totem {
 version: 2
 secauth: off
 cluster_name: hacluster
 clear_node_high_bit: yes
```

```
# Following are old corosync 1.4.x defaults from SLES
# token:  5000
# token_retransmits_before_loss_const: 10
# join:  60
# consensus: 6000
# vsftype: none
# max_messages: 20
# threads: 0

 crypto_cipher: none
 crypto_hash: none

 interface {
  ringnumber: 0
  bindnetaddr: 192.168.1.0
  mcastport: 5405
  ttl:  1
 }
    #Transport protocol
 transport:      udpu

}
nodelist {
        node {
        #ring0 address
        ring0_addr:     192.168.1.11

        }
        node {
        #ring0 address
        ring0_addr:     192.168.1.12

        }
 }

logging {
 fileline: off
 to_stderr: no
 to_logfile: no
 logfile: /var/log/cluster/corosync.log
 to_syslog: yes
 debug:  off
 timestamp: on
 logger_subsys {
  subsys: QUORUM
  debug: off
```

```
 }
}
quorum {
 # Enable and configure quorum subsystem (default: off)
 # see also corosync.conf.5 and votequorum.5
 provider: corosync_votequorum
 expected_votes: 2
 two_node: 1
}
```

B.4 Example for the IPMI STONITH Method

```
primitive rsc_suse01_stonith stonith:external/ipmi \
params hostname="suse01" ipaddr="192.168.1.101" userid="stonith" \
passwd="k1llm3" interface="lanplus" \
op monitor interval="1800" timeout="30"
...
primitive rsc_suse02_stonith stonith:external/ipmi \
params hostname="suse02" ipaddr="192.168.1.102" userid="stonith" \
passwd="k1llm3" interface="lanplus" \
op monitor interval="1800" timeout="30"
...
location loc_suse01_stonith rsc_suse01_stonith -inf: suse01
location loc_suse02_stonith rsc_suse02_stonith -inf: suse02
```