

STANDARD OPERATING PROCEDURE

Pacemaker Cluster Setup

on SLES

CLOUD4C SERVICES PVT LTD

Document control

Prepared By

|  |  |  |
| --- | --- | --- |
| Version | Author | Date |
| V 1.0 | Kartik Horra | 15-09-2022 |

Reviewed By

|  |  |  |
| --- | --- | --- |
| Version | Reviewer | Date |
| V1.0 | Gaurav S | 15-09-2022 |
| V1.0 | Anirban Dutta | 29-09-2022 |

Approved By

|  |  |  |
| --- | --- | --- |
| Version | Approver | Date |
| V 1.0 | Jithender Reddy | 22-09-2022 |

Change Control

|  |  |  |
| --- | --- | --- |
| Version | Change Reason | Effective Date |
| V 1.0 | Initial Release | TBD |

STATEMENT OF CONFIDENTIALITY

|  |
| --- |
| This document contains proprietary trade secret and confidential information to be used solely for evaluating CtrlS & Cloud4C Datacentres’ Ltd. The information contained herein is to be considered confidential. Customer, by receiving this document, agrees that neither this document nor the information disclosed herein, nor any part thereof, shall be reproduced or transferred to other documents, or used or disclosed to others for any purpose except as specifically authorized in writing by Ctrl S Datacentres’ Ltd. |

Contents

[Document control 2](#_Toc115364229)

[Prepared By 2](#_Toc115364230)

[Reviewed By 2](#_Toc115364231)

[Approved By 2](#_Toc115364232)

[Change Control 2](#_Toc115364233)

[STATEMENT OF CONFIDENTIALITY 2](#_Toc115364234)

[Contents 3](#_Toc115364235)

[1. Introduction 4](#_Toc115364236)

[2. Scope 4](#_Toc115364237)

[3. Preparation 4](#_Toc115364238)

[4. Installation Procedures 5](#_Toc115364239)

[4.1 Install Cluster 5](#_Toc115364240)

[4.2 Create a STONITH device on the Pacemaker cluster 10](#_Toc115364241)

[sudo crm configure property stonith-timeout=900 10](file:///D:\Anirban_Work\WIP\Murali\Pacemaker%20Cluster%20Setup%20on%20SLES%20-%20SOP_V1.docx#_Toc115364242)

[4.3 Configure Pacemaker for Azure scheduled events 11](#_Toc115364243)

[4.4 Configure the resources in Pacemaker 11](#_Toc115364244)

[sudo crm configure property maintenance-mode=false 11](file:///D:\Anirban_Work\WIP\Murali\Pacemaker%20Cluster%20Setup%20on%20SLES%20-%20SOP_V1.docx#_Toc115364245)

[4.5 High availability for SAP NetWeaver on Azure VMs on SUSE Linux Enterprise Server with NFS on Azure Files 12](#_Toc115364246)

[4.6 Prepare for SAP NetWeaver installation 12](#_Toc115364247)

[4.7 Install SAP NetWeaver ASCS/ERS 13](#_Toc115364248)

[5. Test cases 16](#_Toc115364249)

[6. Glossary 28](#_Toc115364250)

# Introduction

This document provides instructions on installing and operating the Pacemaker Cluster on SLES12/SLES15

# Scope

The scope of the document is to configure SLES cluster.

# Preparation

Ensure below products are created/installed as per the installation guide,

* Cluster VMs should be in availability sets.
* Install the HA packeges
* Gather the information from Azure APIs.
* Required informations for Azure file shares.

# Installation Procedures

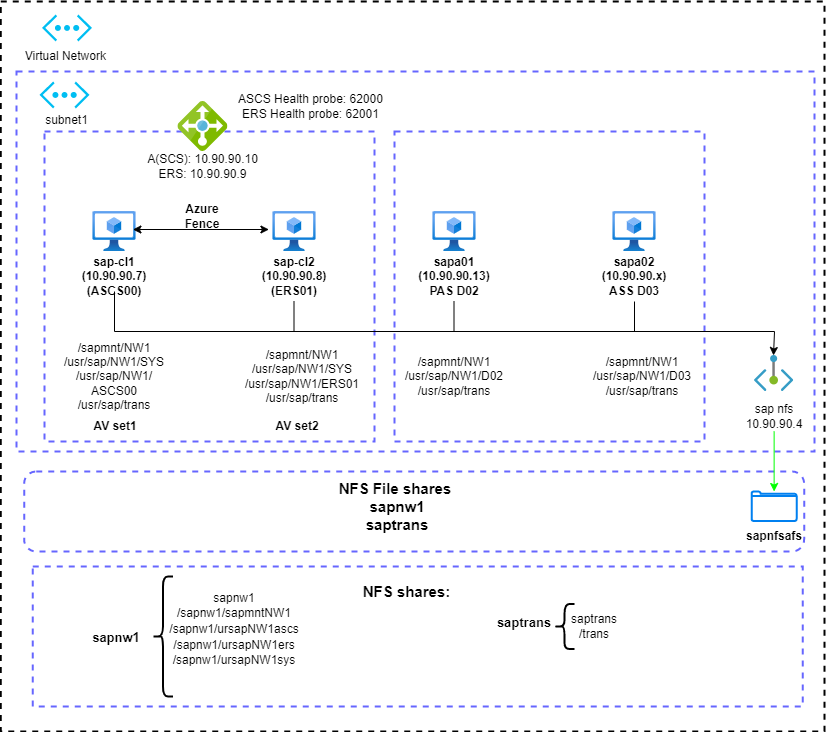


Figure – Architectural NFS

## Install Cluster

1. Update SLES on both nodes and installed the cluster Packages

# sudo zypper update

# zypper in -t pattern ha\_sles

# zypper in SAPHanaSR SAPHanaSR-doc

# zypper in sap-suse-cluster-connector3

1. Install the component, which you'll need for the cluster resources.

# sudo zypper in socat

1. Install the azure-lb component, which you'll need for the cluster resources.

# sudo zypper in resource-agents

**Note:**

Check the version of the resource-agents package, and make sure that the minimum version requirements are met:

* **SLES 12 SP4/SP5**: The version must be resource-agents-4.3.018.a7fb5035-3.30.1 or later.
* **SLES 15/15 SP1**: The version must be resource-agents-4.3.0184.6ee15eb2-4.13.1 or later.

1. Configure the operating system.

Pacemaker occasionally creates many processes, which can exhaust the allowed number. When this happens, a heartbeat between the cluster nodes might fail and lead to a failover of your resources. We recommend increasing the maximum number of allowed processes by setting the following parameter:

# Edit the configuration file

sudo vi /etc/systemd/system.conf

# Change the DefaultTasksMax

#DefaultTasksMax=512

DefaultTasksMax=4096

# Activate this setting

sudo systemctl daemon-reload

# Test to ensure that the change was successful

sudo systemctl --no-pager show | grep DefaultTasksMax

1. Enable SSH access -bidirectional

sudo ssh-keygen

# Enter file in which to save the key (/root/.ssh/id\_rsa), and then select Enter

# Enter passphrase (empty for no passphrase), and then select Enter

# Enter same passphrase again, and then select Enter

# copy the public key

sudo cat /root/.ssh/id\_rsa.pub

1. Install the fence-agents package if you're using a STONITH device, based on the Azure fence agent.

sudo zypper install fence-agents

**Important**

The installed version of the *fence-agents* package must be 4.4.0 or later to benefit from the faster failover times with the Azure fence agent, when a cluster node is fenced. If you're running an earlier version, we recommend that you update the package.

1. Install the Azure Python SDK and Azure Identity python module.

Install the Azure Python SDK on SLES 12 SP4 or SLES 12 SP5:

# You might need to activate the public cloud extension first

SUSEConnect -p sle-module-public-cloud/12/x86\_64

sudo zypper install python-azure-mgmt-compute

sudo zypper install python-azure-identity

Install the Azure Python SDK on SLES 15 or later:

# You might need to activate the public cloud extension first. In this example, the SUSEConnect command is for SLES 15 SP1

SUSEConnect -p sle-module-public-cloud/15.1/x86\_64

sudo zypper install python3-azure-mgmt-compute

sudo zypper install python3-azure-identity

**Important**

Depending on your version and image type, you might need to activate the public cloud extension for your OS release before you can install the Azure Python SDK. You can check the extension by running SUSEConnect ---list-extensions.  
To achieve the faster failover times with the Azure fence agent:

* On SLES 12 SP4 or SLES 12 SP5, install version 4.6.2 or later of the python-azure-mgmt-compute package.
* If your python-azure-mgmt-compute or python3-azure-mgmt-compute package version is 17.0.0-6.7.1, follow the instructions in [SUSE KBA](https://www.suse.com/support/kb/doc/?id=000020377) to update the fence-agents version and install the Azure Identity client library for Python module if it is missing.

1. Set up the hostname resolution.

You can either use a DNS server or modify the */etc/hosts* file on all nodes. This example shows how to use the */etc/hosts* file.

Replace the IP address and the hostname in the following commands.

**Important**

If you're using hostnames in the cluster configuration, it's essential to have a reliable hostname resolution. The cluster communication will fail if the names are unavailable, and that can lead to cluster failover delays.

The benefit of using */etc/hosts* is that your cluster becomes independent of the DNS, which could be a single point of failure too.

sudo vi /etc/hosts

Insert the following lines in the */etc/hosts*. Change the IP address and hostname to match your environment.

# IP address of the first cluster node

**10.0.0.6 prod-cl1-0**

# IP address of the second cluster node

**10.0.0.7 prod-cl1-1**

1. Install the cluster.

If you're not using SBD devices for fencing:

sudo ha-cluster-init -u

# ! NTP is not configured to start at system boot.

# Do you want to continue anyway (y/n)? **y**

# /root/.ssh/id\_rsa already exists - overwrite (y/n)? **n**

# Address for ring0 [10.0.0.6] **Select Enter**

# Port for ring0 [5405] **Select Enter**

# Do you wish to use SBD (y/n)? **n**

#WARNING: Not configuring SBD - STONITH will be disabled.

# Do you wish to configure an administration IP (y/n)? **n**

Add the node to the cluster (node2)

sudo ha-cluster-join

# ! NTP is not configured to start at system boot.

# Do you want to continue anyway (y/n)? **y**

# IP address or hostname of existing node (for example, 192.168.1.1) []**10.0.0.6**

# /root/.ssh/id\_rsa already exists - overwrite (y/n)? **n**

Change the hacluster password to the same password.

sudo passwd hacluster

Adjust the corosync settings.

sudo vi /etc/corosync/corosync.conf

Add the following bold-formatted content to the file if the values are not there or are different. Be sure to change the token to 30000 to allow memory-preserving maintenance. For more information, see the "Maintenance for virtual machines in Azure".

[...]

**token: 30000**

**token\_retransmits\_before\_loss\_const: 10**

**join: 60**

**consensus: 36000**

**max\_messages: 20**

interface {

[...]

}

transport: udpu

}

nodelist {

node {

ring0\_addr:10.0.0.6

}

node {

ring0\_addr:10.0.0.7

}

}

logging {

[...]

}

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**

}

Restart the corosync service.

sudo service corosync restart

## Create a STONITH device on the Pacemaker cluster

If you're using an Azure fence agent as STONITH, run the following commands. After you've assigned roles to both cluster nodes, you can configure the STONITH devices in the cluster.

**Note:**

The 'pcmk\_host\_map' option is required in the command only if the hostnames and the Azure VM names are not identical. Specify the mapping in the format hostname:vm-name. Refer to the bold section in the following command.

sudo crm configure property stonith-enabled=true

crm configure property concurrent-fencing=true

# replace the bold string with your subscription ID, resource group of the VM, tenant ID, service principal application ID and password

sudo crm configure primitive rsc\_st\_azure stonith:fence\_azure\_arm \

params subscriptionId="**subscription ID**" resourceGroup="**resource group**" tenantId="**tenant ID**" login="**application ID**" passwd="**password**" \

pcmk\_monitor\_retries=4 pcmk\_action\_limit=3 power\_timeout=240 pcmk\_reboot\_timeout=900 **pcmk\_host\_map="prod-cl1-0:prod-cl1-0-vm-name;prod-cl1-1:prod-cl1-1-vm-name"** \

op monitor interval=3600 timeout=120

sudo crm configure property stonith-timeout=900

## Configure Pacemaker for Azure scheduled events

Azure offers scheduled events. Scheduled events are provided via the metadata service and allow time for the application to prepare for such events as VM shutdown, VM redeployment, and so on. Resource agent azure-events monitors for scheduled Azure events. If events are detected and the resource agent determines that another cluster node is available, the azure-events agent will place the target cluster node in standby mode to force the cluster to migrate resources away from the VM with pending Azure scheduled events. To achieve that, you must configure additional Pacemaker resources.

Make sure that the package for the azure-events agent is already installed and up to date.

# sudo zypper info resource-agents

## Configure the resources in Pacemaker

#Place the cluster in maintenance mode

sudo crm configure property maintenance-mode=true

#Create Pacemaker resources for the Azure agent

sudo crm configure primitive rsc\_azure-events ocf:heartbeat:azure-events op monitor interval=10s

sudo crm configure clone cln\_azure-events rsc\_azure-events

#Take the cluster out of maintenance mode

## sudo crm configure property maintenance-mode=false

**Note:**

After you've configured the Pacemaker resources for the azure-events agent, if you place the cluster in or out of maintenance mode, you might get warning messages such as:

WARNING: cib-bootstrap-options: unknown attribute 'hostName\_ **hostname'**

WARNING: cib-bootstrap-options: unknown attribute 'azure-events\_globalPullState'

WARNING**:** cib-bootstrap-options: unknown attribute 'hostName\_ **hostname'**

These warning messages can be ignored.

## High availability for SAP NetWeaver on Azure VMs on SUSE Linux Enterprise Server with NFS on Azure Files

This article describes how to deploy and configure VMs, install the cluster framework, and install an HA SAP NetWeaver system, using [NFS on Azure Files](https://docs.microsoft.com/en-us/azure/storage/files/files-nfs-protocol). The example configurations use VMs that run on SUSE Linux Enterprise Server (SLES).

## Prepare for SAP NetWeaver installation

Create the shared directories on both nodes

sudo mkdir -p /sapmnt/NW1

sudo mkdir -p /usr/sap/trans

sudo mkdir -p /usr/sap/NW1/SYS

sudo mkdir -p /usr/sap/NW1/ASCS00

sudo mkdir -p /usr/sap/NW1/ERS01

Mount the file systems that will not be controlled by the Pacemaker cluster on both nodes

vi /etc/fstab

# Add the following lines to fstab, save and exit

sapnfs.file.core.windows.net:/sapnfsafs/saptrans /usr/sap/trans nfs vers=4,minorversion=1,sec=sys 0 0

sapnfs.file.core.windows.net:/sapnfsafs/sapnw1/sapmntNW1 /sapmnt/NW1 nfs vers=4,minorversion=1,sec=sys 0 0

sapnfs.file.core.windows.net:/sapnfsafs/sapnw1/usrsapNW1sys/ /usr/sap/NW1/SYS nfs vers=4,minorversion=1,sec=sys 0 0

# Mount the file systems

mount –a

## Install SAP NetWeaver ASCS/ERS

1. **Create a file systems, virtual IP resource and health-probe for the ASCS instance**

**Important**

We recommend using azure-lb resource agent, which is part of package resource-agents, with the following package version requirements:

* For SLES 12 SP4/SP5, the version must be at least resource-agents-4.3.018.a7fb5035-3.30.1.
* For SLES 15 and above, the version must be at least resource-agents-4.3.0184.6ee15eb2-4.13.1.

sudo crm configure primitive fs\_NW1\_ASCS Filesystem device='sapnfs.file.core.windows.net:/sapnfsafs/sapnw1/usrsapNW1ascs' directory='/usr/sap/NW1/ASCS00' fstype='nfs' options='sec=sys,vers=4.1' \

op start timeout=60s interval=0 \

op stop timeout=60s interval=0 \

op monitor interval=20s timeout=40s

sudo crm configure primitive vip\_NW1\_ASCS IPaddr2 \

params ip=10.90.90.10 \

op monitor interval=10 timeout=20

sudo crm configure primitive nc\_NW1\_ASCS azure-lb port=62000

sudo crm configure group g-NW1\_ASCS fs\_NW1\_ASCS nc\_NW1\_ASCS vip\_NW1\_ASCS \

meta resource-stickiness=3000

Make sure that the cluster status is ok and that all resources are started. It is not important on which node the resources are running.

1. **Create a file system, virtual IP resource and health-probe for the ERS instance**

sudo crm configure primitive fs\_NW1\_ERS Filesystem device='sapnfs.file.core.windows.net:/sapnfsafs/sapnw1/usrsapNW1ers' directory='/usr/sap/NW1/ERS01' fstype='nfs' options='sec=sys,vers=4.1' \

op start timeout=60s interval=0 \

op stop timeout=60s interval=0 \

op monitor interval=20s timeout=40s

sudo crm configure primitive vip\_NW1\_ERS IPaddr2 \

params ip=10.90.90.9 \

op monitor interval=10 timeout=20

sudo crm configure primitive nc\_NW1\_ERS azure-lb port=62101

sudo crm configure group g-NW1\_ERS fs\_NW1\_ERS nc\_NW1\_ERS vip\_NW1\_ERS

Make sure that the cluster status is ok and that all resources are started. It is not important on which node the resources are running.

Once OS cluster configuration done, then we will hand over the servers to SAP team to install the applications, once application installation done, they will provide the below profiles to configure the application under cluster

ASCS profile: /sapmnt/NW1/profile/NW1\_ASCS00\_sapascs

ERS Profile: /sapmnt/NW1/profile/NW1\_ERS01\_sapers

1. **Create the SAP cluster resources**

sudo crm configure property maintenance-mode="true"

sudo crm configure primitive rsc\_sap\_NW1\_ASCS00 SAPInstance \

operations \$id=rsc\_sap\_NW1\_ASCS00-operations \

op monitor interval=11 timeout=60 on-fail=restart \

params InstanceName=NW1\_ASCS00\_sapascs START\_PROFILE="/sapmnt/NW1/profile/NW1\_ASCS00\_sapascs" \

AUTOMATIC\_RECOVER=false \

meta resource-stickiness=5000 failure-timeout=60 migration-threshold=1 priority=10

sudo crm configure primitive rsc\_sap\_NW1\_ERS01 SAPInstance \

operations \$id=rsc\_sap\_NW1\_ERS01-operations \

op monitor interval=11 timeout=60 on-fail=restart \

params InstanceName=NW1\_ERS01\_sapers START\_PROFILE="/sapmnt/NW1/profile/NW1\_ERS01\_sapers" AUTOMATIC\_RECOVER=false IS\_ERS=true \

meta priority=1000

sudo crm configure modgroup g-NW1\_ASCS add rsc\_sap\_NW1\_ASCS00

sudo crm configure modgroup g-NW1\_ERS add rsc\_sap\_NW1\_ERS01

sudo crm configure colocation col\_sap\_NW1\_no\_both -5000: g-NW1\_ERS g-NW1\_ASCS

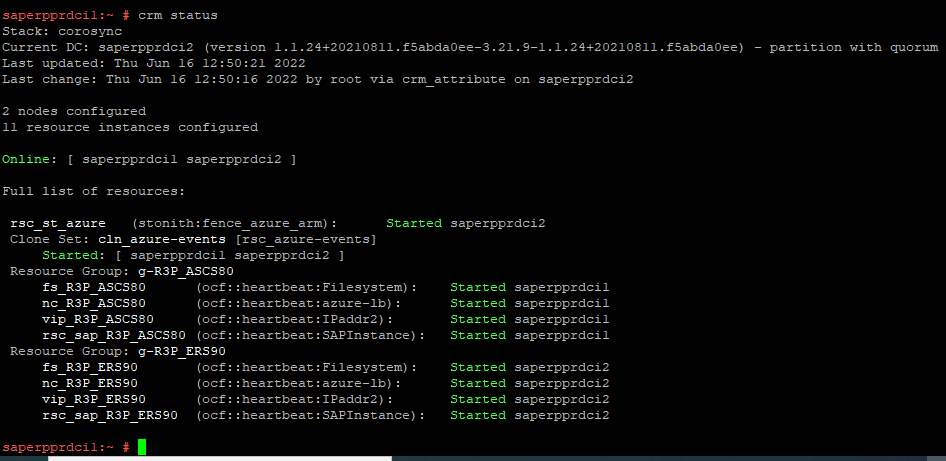
sudo crm configure location loc\_sap\_NW1\_failover\_to\_ers rsc\_sap\_NW1\_ASCS00 rule 2000: runs\_ers\_NW1 eq 1

sudo crm configure order ord\_sap\_NW1\_first\_start\_ascs Optional: rsc\_sap\_NW1\_ASCS00:start rsc\_sap\_NW1\_ERS01:stop symmetrical=false

sudo crm node online sap-cl1

sudo crm configure property maintenance-mode="false"

Now Finaly below is the cluster output



# Test cases

The following tests are a copy of the test cases in the best practices guides of SUSE. They are copied for your convenience. Always also read the best practices guides and perform all additional tests that might have been added.

1. **Test HAGetFailoverConfig, HACheckConfig and HACheckFailoverConfig**

Run the following commands as <sapsid>adm on the node where the ASCS instance is currently running. If the commands fail with FAIL: Insufficient memory, it might be caused by dashes in your hostname. This is a known issue and will be fixed by SUSE in the sap-suse-cluster-connector package.

Copy

# HACheckConfig

# OK

# state, category, description, comment

# SUCCESS, SAP CONFIGURATION, Redundant ABAP instance configuration, 2 ABAP instances detected

# SUCCESS, SAP CONFIGURATION, Redundant Java instance configuration, 0 Java instances detected

# SUCCESS, SAP CONFIGURATION, Enqueue separation, All Enqueue server separated from application server

# SUCCESS, SAP CONFIGURATION, MessageServer separation, All MessageServer separated from application server

# SUCCESS, SAP CONFIGURATION, ABAP instances on multiple hosts, ABAP instances on multiple hosts detected

# SUCCESS, SAP CONFIGURATION, Redundant ABAP SPOOL service configuration, 2 ABAP instances with SPOOL service detected

# SUCCESS, SAP STATE, Redundant ABAP SPOOL service state, 2 ABAP instances with active SPOOL service detected

nw1-cl-0:nw1adm 54> sapcontrol -nr **00** -function HAGetFailoverConfig

# 15.08.2018 13:50:36

# HAGetFailoverConfig

# OK

# HAActive: TRUE

# HAProductVersion: Toolchain Module

# HASAPInterfaceVersion: Toolchain Module (sap\_suse\_cluster\_connector 3.0.1)

# HADocumentation: https://www.suse.com/products/sles-for-sap/resource-library/sap-best-practices/

# HAActiveNode:

# HANodes: nw1-cl-0, nw1-cl-1

nw1-cl-0:nw1adm 55> sapcontrol -nr 00 -function HACheckConfig

# 15.08.2018 14:00:04

# SUCCESS, SAP STATE, ABAP instances with ABAP SPOOL service on multiple hosts, ABAP instances with active ABAP SPOOL service on multiple hosts detected

# SUCCESS, SAP CONFIGURATION, Redundant ABAP BATCH service configuration, 2 ABAP instances with BATCH service detected

# SUCCESS, SAP STATE, Redundant ABAP BATCH service state, 2 ABAP instances with active BATCH service detected

# SUCCESS, SAP STATE, ABAP instances with ABAP BATCH service on multiple hosts, ABAP instances with active ABAP BATCH service on multiple hosts detected

# SUCCESS, SAP CONFIGURATION, Redundant ABAP DIALOG service configuration, 2 ABAP instances with DIALOG service detected

# SUCCESS, SAP STATE, Redundant ABAP DIALOG service state, 2 ABAP instances with active DIALOG service detected

# SUCCESS, SAP STATE, ABAP instances with ABAP DIALOG service on multiple hosts, ABAP instances with active ABAP DIALOG service on multiple hosts detected

# SUCCESS, SAP STATE, SCS instance running, SCS instance status ok

# SUCCESS, SAP CONFIGURATION, SAPInstance RA sufficient version (nw1-ascs\_NW1\_00), SAPInstance includes is-ers patch

# SUCCESS, SAP CONFIGURATION, Enqueue replication (nw1-ascs\_NW1\_00), Enqueue replication enabled

# SUCCESS, SAP STATE, Enqueue replication state (nw1-ascs\_NW1\_00), Enqueue replication active

nw1-cl-0:nw1adm 56> sapcontrol -nr 00 -function HACheckFailoverConfig

# 15.08.2018 14:04:08

# HACheckFailoverConfig

# OK

# state, category, description, comment

# SUCCESS, SAP CONFIGURATION, SAPInstance RA sufficient version, SAPInstance includes is-ers patch

1. **Manually migrate the ASCS instance**

Resource state before starting the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-0

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Run the following commands as root to migrate the ASCS instance.

nw1-cl-0:~ # crm resource migrate rsc\_sap\_NW1\_ASCS00 force

# INFO: Move constraint created for rsc\_sap\_NW1\_ASCS00

nw1-cl-0:~ # crm resource unmigrate rsc\_sap\_NW1\_ASCS00

# INFO: Removed migration constraints for rsc\_sap\_NW1\_ASCS00

# Remove failed actions for the ERS that occurred as part of the migration

nw1-cl-0:~ # crm resource cleanup rsc\_sap\_NW1\_ERS02

Resource state after the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-0

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

1. **Test HAFailoverToNode**

Resource state before starting the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-0

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

Run the following commands as <sapsid>adm to migrate the ASCS instance.

nw1-cl-0:nw1adm 55> sapcontrol -nr 00 -host nw1-ascs -user nw1adm <password> -function HAFailoverToNode ""

# run as root

# Remove failed actions for the ERS that occurred as part of the migration

nw1-cl-0:~ # crm resource cleanup rsc\_sap\_NW1\_ERS02

# Remove migration constraints

nw1-cl-0:~ # crm resource clear rsc\_sap\_NW1\_ASCS00

#INFO: Removed migration constraints for rsc\_sap\_NW1\_ASCS00

Resource state after the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-0

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

1. **Simulate node crash**

Resource state before starting the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-0

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Run the following command as root on the node where the ASCS instance is running

nw1-cl-0:~ # echo b > /proc/sysrq-trigger

If you use SBD, Pacemaker should not automatically start on the killed node. The status after the node is started again should look like this.

Online: [ nw1-cl-1 ]

OFFLINE: [ nw1-cl-0 ]

Full list of resources:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Failed Actions:

\* rsc\_sap\_NW1\_ERS02\_monitor\_11000 on nw1-cl-1 'not running' (7): call=219, status=complete, exitreason='none',

last-rc-change='Wed Aug 15 14:38:38 2018', queued=0ms, exec=0ms

Use the following commands to start Pacemaker on the killed node, clean the SBD messages, and clean the failed resources.

# run as root

# list the SBD device(s)

nw1-cl-0:~ # cat /etc/sysconfig/sbd | grep SBD\_DEVICE=

# SBD\_DEVICE="/dev/disk/by-id/scsi-36001405772fe8401e6240c985857e116;/dev/disk/by-id/scsi-36001405034a84428af24ddd8c3a3e9e1;/dev/disk/by-id/scsi-36001405cdd5ac8d40e548449318510c3"

nw1-cl-0:~ # sbd -d /dev/disk/by-id/scsi-36001405772fe8401e6240c985857e116 -d /dev/disk/by-id/scsi-36001405034a84428af24ddd8c3a3e9e1 -d /dev/disk/by-id/scsi-36001405cdd5ac8d40e548449318510c3 message nw1-cl-0 clear

nw1-cl-0:~ # systemctl start pacemaker

nw1-cl-0:~ # crm resource cleanup rsc\_sap\_NW1\_ASCS00

nw1-cl-0:~ # crm resource cleanup rsc\_sap\_NW1\_ERS02

Resource state after the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

1. **Test manual restart of ASCS instance**

Resource state before starting the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

Create an enqueue lock by, for example edit a user in transaction su01. Run the following commands as <sapsid>adm on the node where the ASCS instance is running. The commands will stop the ASCS instance and start it again. If using enqueue server 1 architecture, the enqueue lock is expected to be lost in this test. If using enqueue server 2 architecture, the enqueue will be retained.

Copy

nw1-cl-1:nw1adm 54> sapcontrol -nr 00 -function StopWait 600 2

The ASCS instance should now be disabled in Pacemaker

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Stopped (disabled)

Start the ASCS instance again on the same node.

nw1-cl-1:nw1adm 54> sapcontrol -nr 00 -function StartWait 600 2

The enqueue lock of transaction su01 should be lost and the back-end should have been reset. Resource state after the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

1. **Kill message server process**

Resource state before starting the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

Run the following commands as root to identify the process of the message server and kill it.

nw1-cl-1:~ # pgrep ms.sapNW1 | xargs kill -9

If you only kill the message server once, it will be restarted by sapstart. If you kill it often enough, Pacemaker will eventually move the ASCS instance to the other node. Run the following commands as root to clean up the resource state of the ASCS and ERS instance after the test.

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

1. Kill enqueue server process

Resource state before starting the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

nw1-cl-0:~ # crm resource cleanup rsc\_sap\_NW1\_ASCS00

nw1-cl-0:~ # crm resource cleanup rsc\_sap\_NW1\_ERS02

Resource state after the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Run the following commands as root on the node where the ASCS instance is running to kill the enqueue server.

nw1-cl-0:~ #

#If using ENSA1

pgrep en.sapNW1 | xargs kill -9

#If using ENSA2

pgrep -f enq.sapNW1 | xargs kill -9

The ASCS instance should immediately fail over to the other node, in the case of ENSA1. The ERS instance should also fail over after the ASCS instance is started. Run the following commands as root to clean up the resource state of the ASCS and ERS instance after the test.

nw1-cl-0:~ # crm resource cleanup rsc\_sap\_NW1\_ASCS00

nw1-cl-0:~ # crm resource cleanup rsc\_sap\_NW1\_ERS02

Resource state after the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

1. **Kill enqueue replication server process**

Resource state before starting the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

Run the following command as root on the node where the ERS instance is running to kill the enqueue replication server process.

nw1-cl-0:~ # pgrep er.sapNW1 | xargs kill -9

If you only run the command once, sapstart will restart the process. If you run it often enough, sapstart will not restart the process and the resource will be in a stopped state. Run the following commands as root to clean up the resource state of the ERS instance after the test.

nw1-cl-0:~ # crm resource cleanup rsc\_sap\_NW1\_ERS02

Resource state after the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

1. **Kill enqueue sapstartsrv process**

Resource state before starting the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-0

Run the following commands as root on the node where the ASCS is running.

nw1-cl-1:~ # pgrep -fl ASCS00.\*sapstartsrv

# 59545 sapstartsrv

nw1-cl-1:~ # kill -9 59545

The sapstartsrv process should always be restarted by the Pacemaker resource agent. Resource state after the test:

stonith-sbd (stonith:external/sbd): Started nw1-cl-1

Resource Group: g-NW1\_ASCS

fs\_NW1\_ASCS (ocf::heartbeat:Filesystem): Started nw1-cl-1

nc\_NW1\_ASCS (ocf::heartbeat:azure-lb): Started nw1-cl-1

vip\_NW1\_ASCS (ocf::heartbeat:IPaddr2): Started nw1-cl-1

rsc\_sap\_NW1\_ASCS00 (ocf::heartbeat:SAPInstance): Started nw1-cl-1

Resource Group: g-NW1\_ERS

fs\_NW1\_ERS (ocf::heartbeat:Filesystem): Started nw1-cl-0

nc\_NW1\_ERS (ocf::heartbeat:azure-lb): Started nw1-cl-0

vip\_NW1\_ERS (ocf::heartbeat:IPaddr2): Started nw1-cl-0

rsc\_sap\_NW1\_ERS02 (ocf::heartbeat:SAPInstance): Started nw1-cl-0