Technical Guide

Redundancy of virtual servers in a cloud environment

SBD Fencing with Pacemaker

May 2021

SAMSUNG SDS

Contents

1. Overview

This document covers how to configure a redundant server by utilizing a redundancy solution to ensure high availability of virtual servers in a cloud environment.

Fencing[1] settings are set to prevent split-brain[2] in a two-node cluster configuration. Among the fencing settings using SBD in a cloud environment, the main options and test items after configuration are included in this document.

2. SBD fencing

SBD Fencing is short for Storage-Based Death or STONITH[3] Block Device. It uses shared storage installed between cluster nodes, and the server's sbd daemon process decides whether or not to fence through messages written in this shared area. In the split-brain situation, if a message is sent to the node to be fenced, the node will be self-fenced by the watchdog.

Depending on the SBD usage policy, it can be set to self-fencing alone when the shared disk for SBD fails, and even if the SBD disk fails, it can be decided whether or not to use fencing according to the status of the Pacemaker's node.

SBD Fencing is more stable with hardware watchdog. It can be used as a software watchdog for SBD Fencing in a cloud environment, but it may not be processed normally due to issues such as kernel hang.

3. SBD components

The components required to set up SBD fencing are as follows:

- SBD partition: A shared data area that can communicate between nodes. Up to 3 can be added.
- SBD daemon process: Monitors the SBD area and communicates with the cluster daemon process.
- Message: Communication through message in SBD area between nodes.
- Watchdog: Periodic monitoring for self-fencing

4. How SBD fencing works

[1] As a device/method to protect data from system failure, it is a method to guarantee the integrity of shared data by disconnecting the connection to the shared resource (e.g., shared storage) of the node in case of OS resource and highly available cluster failure.

[2] This phenomenon occurs when a network is temporarily disconnected between two system groups composed of a cluster, and all nodes in the cluster recognize that each node is a primary.

[3] Shoot The Other Node In The Head

Let's take a look at how SBD fencing works.

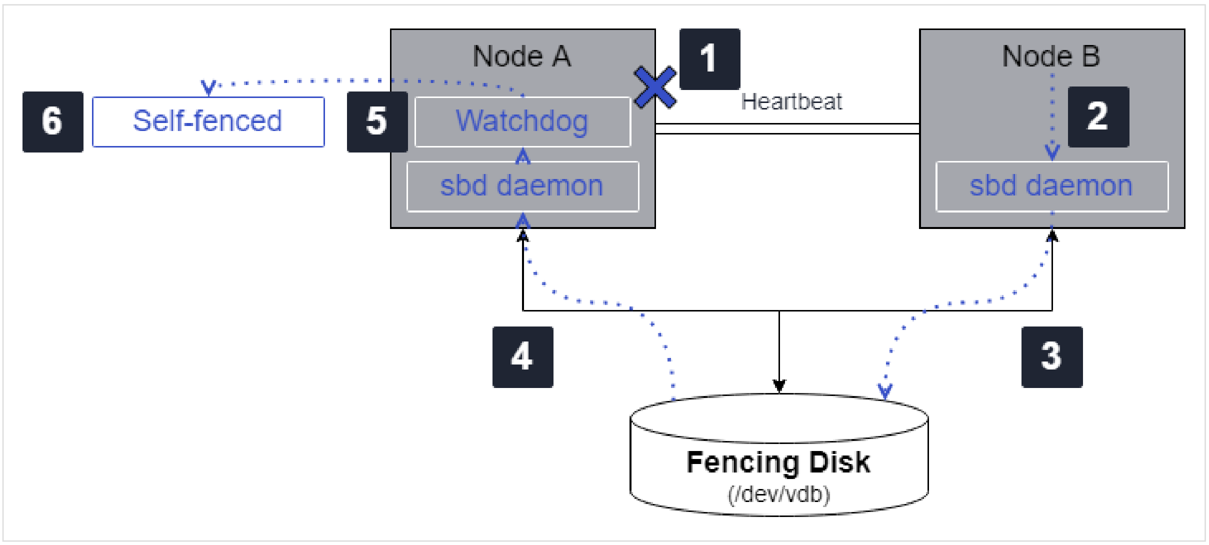


Figure 1. How SBD fencing works

1. Split-brain phenomenon occurs when heartbeat communication is stopped due to NIC down of node A.
2. Pacemaker daemon process of node B executes fencing command to sbd daemon process by interrupting heartbeat communication.
3. Node B's sbd daemon process writes fencing messages to the SBD disk.
4. Node A's sbd daemon process reads the fencing message.
5. Sbd daemon process calls watchdog.
6. Reboot the server by self-fencing by watchdog.

5. SBD fencing configuration options

Let's take a look at the settings for the main options for configuring SBD fencing.

1. Pacemaker sets options in the /etc/sysconfig/sbd file.
2. For a description of the main options, refer to the following table.
3. If the environment variable is not defined in the /etc/sysconfig/sbd file, additional options can be set in SBD_OPTS.
   Check if the main options above are within the file and add them if not.

Option	Description	Recommended value
SBD_PACEMAKER (-P)	✓ Even if SBD disk fails, fencing does not occur if the pacemaker node status is normal. ✓ In a general cluster environment that is not for IO fencing purposes, it is recommended to use yes by default.	yes
SBD_WATCHDOG (-W)	✓ Fencing using watchdog	yes
SBD_WATCHDOG_TIMEOUT	✓ Timeout time when the SBD watchdog device is not connected. After this time, the node is fencing. However, if SBD_PACEMAKER=yes is set and the node status is normal, fencing does not occur. ✓ The watchdog timeout time is determined by the storage IO delay time. ✓ If the SBD device is multipath or iSCSI, the timeout is set to the time required to detect a path error and switch to the next path. In the case of multipath, set greater than max_polling_interval (default: 4 * polling_interval-10~20)	10 ~ 20
msgwait timeout (-4 <timeout>)	✓ The time at which a message written to the SBD device's node slot is considered delivered ✓ Set to twice the SBD_WATCHDOG_TIMEOUT ✓ Pacemaker's stonith-timeout (the time to wait until the entire STONITH operation is completed) should be set higher than the (msgwait timeout + 20%) value. stonith-timeout >= msgwait timeout + msgwait timeout * 20%	20 ~ 40

Table 1. Main options of SBD fencing

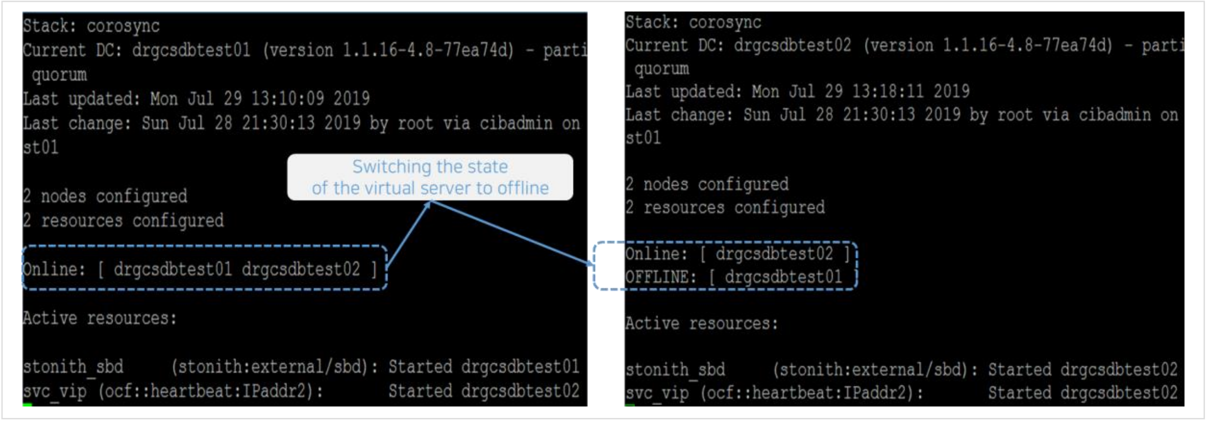
```
# cat /etc/sysconfig/sbd

SBD_DEVICE="/dev/vdi"

# Whether to enable the pacemaker integration.
#
SBD_PACEMAKER=yes
# -P option (Enable if the option is once, Disable if twice (-P -P) or none)
SBD_STARTMODE=always
# -S 0
SBD_DELAY_START=no
SBD_WATCHDOG_DEV=/dev/watchdog
SBD_WATCHDOG=yes
# -W option (Enable if the option is once, Disable if twice (-W -W) or none)
SBD_WATCHDOG_TIMEOUT=10

## Type: string
## Default: ""
#
# Additional options for starting sbd
#
SBD_OPTS=""
# Additional option
```

Figure 2. /etc/sysconfig/sbd file

6. Test results of SDB fencing

Perform a failover test using SBD fencing and check if it works properly.

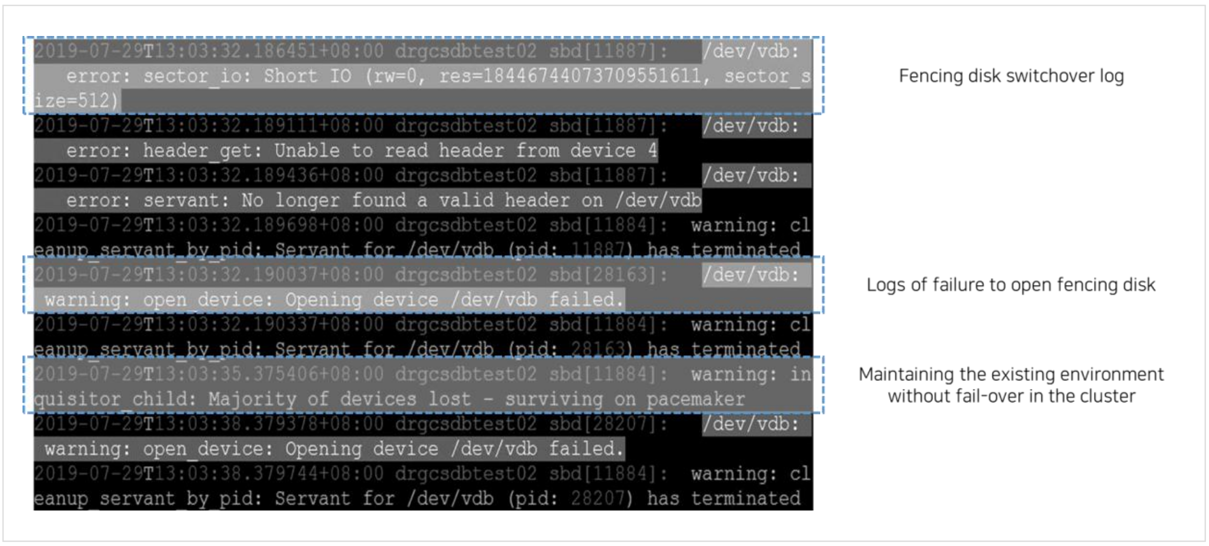
1. Test Case 1. Failover verification in case of forced reboot of primary virtual server



```
Stack: corosync
Current DC: drgcsdbtest01 (version 1.1.16-4.8-77ea74d) - parti
 quorum
Last updated: Mon Jul 29 13:10:09 2019
Last change: Sun Jul 28 21:30:13 2019 by root via cibadmin on
st01

2 nodes configured
2 resources configured

Online: [ drgcsdbtest01 drgcsdbtest02 ]

Active resources:

stonith_sbd     (stonith:external/sbd): Started drgcsdbtest01
svc_vip (ocf::heartbeat:IPaddr2):       Started drgcsdbtest02
```

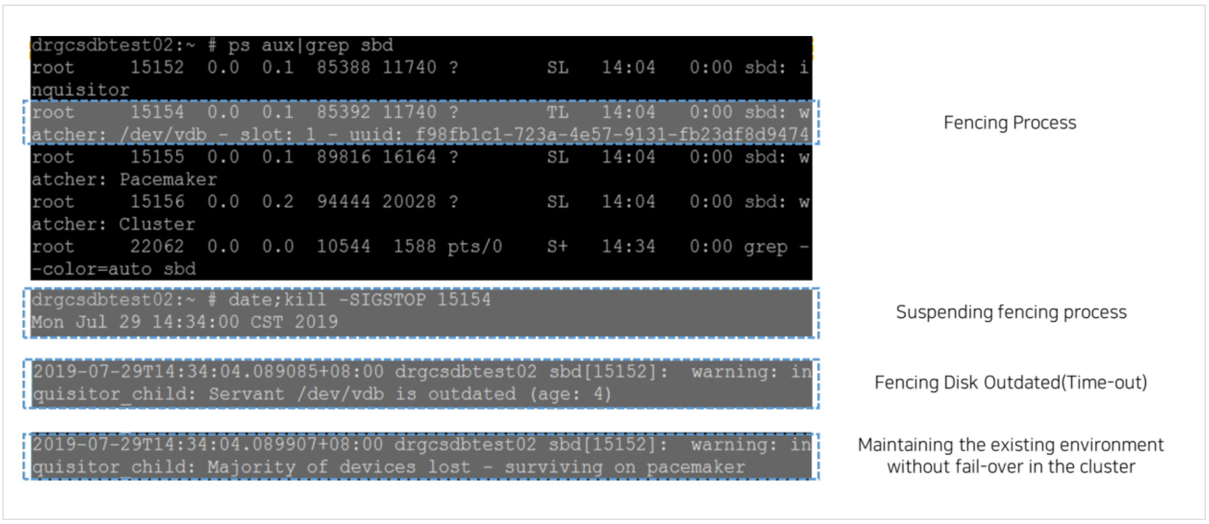
```
Stack: corosync
Current DC: drgcsdbtest02 (version 1.1.16-4.8-77ea74d) - parti
 quorum
Last updated: Mon Jul 29 13:18:11 2019
Last change: Sun Jul 28 21:30:13 2019 by root via cibadmin on
st01

2 nodes configured
2 resources configured

Online: [ drgcsdbtest02 ]
OFFLINE: [ drgcsdbtest01 ]

Active resources:

stonith_sbd     (stonith:external/sbd): Started drgcsdbtest02
svc_vip (ocf::heartbeat:IPaddr2):       Started drgcsdbtest02
```

2. Test Case 2. Failover verification through fencing disk removal


Fencing disk switchover log
Logs of failure to open fencing disk
Maintaining the existing environment without fail-over in the cluster

```
2019-07-29T13:03:32.186451+08:00 drgcsdbtest02 sbd[11887]:   /dev/vdb:
   error: sector_io: Short IO (rw=0, res=18446744073709551611, sector_s
ize=512)
2019-07-29T13:03:32.189111+08:00 drgcsdbtest02 sbd[11887]:   /dev/vdb:
   error: header_get: Unable to read header from device 4
2019-07-29T13:03:32.189436+08:00 drgcsdbtest02 sbd[11887]:   /dev/vdb:
   error: servant: No longer found a valid header on /dev/vdb
2019-07-29T13:03:32.189698+08:00 drgcsdbtest02 sbd[11884]:  warning: cl
eanup_servant_by_pid: Servant for /dev/vdb (pid: 11887) has terminated
2019-07-29T13:03:32.190037+08:00 drgcsdbtest02 sbd[28163]:   /dev/vdb:
 warning: open_device: Opening device /dev/vdb failed.
2019-07-29T13:03:32.190337+08:00 drgcsdbtest02 sbd[11884]:  warning: cl
eanup_servant_by_pid: Servant for /dev/vdb (pid: 28163) has terminated
2019-07-29T13:03:35.375406+08:00 drgcsdbtest02 sbd[11884]:  warning: in
quisitor_child: Majority of devices lost - surviving on pacemaker
2019-07-29T13:03:38.379378+08:00 drgcsdbtest02 sbd[28207]:   /dev/vdb:
 warning: open_device: Opening device /dev/vdb failed.
2019-07-29T13:03:38.379744+08:00 drgcsdbtest02 sbd[11884]:  warning: cl
eanup_servant_by_pid: Servant for /dev/vdb (pid: 28207) has terminated
```

3. Test Case 3. Failover verification through fencing process suspended situation


Fencing Process
Suspending fencing process
Fencing Disk Outdated(Time-out)
Maintaining the existing environment without fail-over in the cluster

```
drgcsdbtest02:~ # ps aux|grep sbd
root      15152  0.0  0.1  85388 11740 ?        SL   14:04   0:00 sbd: i
nquisitor
root      15154  0.0  0.1  85392 11740 ?        TL   14:04   0:00 sbd: w
atcher: /dev/vdb - slot: 1 - uuid: f98fb1c1-723a-4e57-9131-fb23df8d9474
root      15155  0.0  0.1  89816 16164 ?        SL   14:04   0:00 sbd: w
atcher: Pacemaker
root      15156  0.0  0.2  94444 20028 ?        SL   14:04   0:00 sbd: w
atcher: Cluster
root      22062  0.0  0.0  10544  1588 pts/0    S+   14:34   0:00 grep -
-color=auto sbd
drgcsdbtest02:~ # date;kill -SIGSTOP 15154
Mon Jul 29 14:34:00 CST 2019

2019-07-29T14:34:04.089085+08:00 drgcsdbtest02 sbd[15152]:  warning: in
quisitor_child: Servant /dev/vdb is outdated (age: 4)

2019-07-29T14:34:04.089907+08:00 drgcsdbtest02 sbd[15152]:  warning: in
quisitor_child: Majority of devices lost - surviving on pacemaker
```

The main options and test results for Fencing configuration using SBD have been explained. We hope that it will help prevent split-brain phenomenon by utilizing SBD Fencing when configuring redundancy for virtual servers in a cloud environment.