**SAP Cluster Monitoring using CloudWatch and SNS**

Contents

[1. Document Summary 1](#_Toc104395148)

[2. AWS Services used in this setup: 2](#_Toc104395149)

[3. Architecture 2](#_Toc104395150)

[4. Configure Redhat Pacemaker alerts 3](#_Toc104395151)

[4.a Confirm pacemaker cluster has been setup 3](#_Toc104395152)

[4.b Install RedHat pacemaker alert agents 3](#_Toc104395153)

[4.b.1 Run the commands below on the primary EC2 cluster instance: 3](#_Toc104395154)

[4.b.2 Run the commands below on the secondary EC2 cluster instance: 4](#_Toc104395155)

[5. Configure CloudWatch Log Group and SNS Notification 4](#_Toc104395156)

[5.a Create a CloudWatch log group 4](#_Toc104395157)

[5.b Create Metric filters 5](#_Toc104395158)

[5.c Create AWS CloudWatch Metric Alarm 7](#_Toc104395159)

[6. Configure AWS CloudWatch Agent 11](#_Toc104395160)

[6.a Check if CloudWatch agent has been installed: 11](#_Toc104395161)

[6.b Run AWS CloudWatch configuration wizard: 11](#_Toc104395162)

[6.c Start the AWS CloudWatch agent on the clustered EC2 instances: 13](#_Toc104395163)

[7. References 15](#_Toc104395164)

# 1. Document Summary

This document will outline the steps required for monitoring and alerting the RedHat pacemaker cluster for SAP application and HANA database resources using AWS cloudwatch and SNS (Simple Notification Service). The AWS resources can be created through terraform or AWS console and pacemaker commands will be used to perform cluster alerting configuration.

Often in an SAP project, once cluster is setup, monitoring of the cluster takes a back seat and the AWS Infrastructure team has to either monitor clusters manually or follow a reactive approach where the SAP Basis team will inform when the cluster has an issue since it affects the SAP application and/or SAP HANA database.

This solution will provide a proactive approach for the SAP Infrastructure and even the Basis team to detect issues on the cluster for SAP resources in a near real time manner.

# 2. AWS Services used in this setup:

* AWS EC2
* AWS CloudWatch
* AWS SNS

# 3. Architecture



# 4. Configure Redhat Pacemaker alerts

A Pacemaker cluster is an event-driven system, where an event might be a resource or node failure, a configuration change, or a resource starting or stopping. You can configure Pacemaker cluster alerts to take some external action when a cluster event occurs.

RedHat pacemaker provides alert agents which can take certain actions based on events that occur in the cluster. Below are the steps to install and configure these alert agents.

## 4.a Confirm pacemaker cluster has been setup

This will confirm that the basic cluster setup to monitor SAP services has been set up using the “pcs status” command.

## 4.b Install RedHat pacemaker alert agents

### 4.b.1 Run the commands below on the primary EC2 cluster instance:

install --mode=0755 /usr/share/pacemaker/alerts/alert\_file.sh.sample /var/lib/pacemaker/alert\_file.sh

touch /var/log/pcmk\_alert\_file.log

chown hacluster:haclient /var/log/pcmk\_alert\_file.log

chmod 600 /var/log/pcmk\_alert\_file.log

pcs alert create id=alert\_file description="Log events to a file." path=/var/lib/pacemaker/alert\_file.sh

pcs alert recipient add alert\_file id=my-alert\_logfile value=/var/log/pcmk\_alert\_file.log



Use command “pcs alert” to check the alert resource created:





### 4.b.2 Run the commands below on the secondary EC2 cluster instance:

install --mode=0755 /usr/share/pacemaker/alerts/alert\_file.sh.sample /var/lib/pacemaker/alert\_file.sh

touch /var/log/pcmk\_alert\_file.log

chown hacluster:haclient /var/log/pcmk\_alert\_file.log

chmod 600 /var/log/pcmk\_alert\_file.log



You do not need to run the steps to create the alert id on the secondary cluster instance



# 5. Configure CloudWatch Log Group and SNS Notification

The creation of the AWS cloudwatch log group, cloudwatch metric filters and the cloudwatch alarms can be handled by Terraform or performed through AWS console.

## 5.a Create a CloudWatch log group

The AWS CloudWatch agent will transfer the pacemaker alert file to the cloudwatch log group as a log stream.



After performing step no. 6, you will see below log streams coming from the cluster servers.



## 5.b Create Metric filters

Metric filters will help to search for a pattern in the cloudwatch log streams, in our case for example, we are searching for “stop <cluster resource name>”. Once this pattern is identified, it will update a custom metric.

Goto the cloudwatch log group created and you will find the metric filters tab:



For the SAP Application (ASCS/SCS and ERS) high availability setup, you will create below filters that monitor the ASCS/SCS and ERS cluster resources:



When the metric filter identifies the said pattern, it will update the cloudwatch custom metric “sapcluster\_xq1” to a value of 1.

CloudWatch alarm “SAP-Cluster-QA1-XQ1” will monitor the metric sapcluster\_xq1 and send out a SNS notification as soon as the value changes to 1 indicating that the cluster resource has stopped and action needs to be taken.



Similarly, if there is SAP HANA database cluster, below filter can be setup:



## 5.c Create AWS CloudWatch Metric Alarm

Below are the two cloudwatch alarms for the SAP ASCS/SCS and ERS and SAP HANA database clusters.



For the SAP HANA cluster, it was observed that the metric value to be tracked is anything greater than 0 as compared to the SAP application cluster where the tracked metric value is 1.

Whenever there is a cluster issue, the value will change to 1 and once the resource is started again, the value drops to 0 as seen below.

Below diagram shows how custom cloudwatch metric looks for our SAP XQ1 system:



Below are the properties of the cloudwatch alarms created for SAP ASCS/SCS and ERS clusters:



The alarm will trigger a SNS notification as seen here, the SNS topic can be an email to the Infrastructure/Basis team:



Below are the properties of the cloudwatch alarms created for SAP HANA DB clusters respectively:



The alarm will trigger a SNS notification as seen here, the SNS topic can be an email to the Infrastructure/Basis team:



# 6. Configure AWS CloudWatch Agent

AWS CloudWatch agent will transfer the contents of the pacemaker alert log file created in the above steps to an AWS CloudWatch log group/stream.

## 6.a Check if CloudWatch agent has been installed:



## 6.b Run AWS CloudWatch configuration wizard:

In this step, we will configure the cloudwatch agent to track the pacemaker alert log file.

First, we will configure the primary cluster instance by run the command:

sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-config-wizard





Select the below choices:



Here you will enter the name of the cloudwatch log group to which the pcs alert log file will be sent as a log stream.



Perform the same steps on the secondary cluster EC2 instance:

sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-config-wizard





## 6.c Start the AWS CloudWatch agent on the clustered EC2 instances:

Run the below command on the primary and secondary cluster EC2 instances to start the cloudwatch agent:

sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config -m ec2 -s -c file:/opt/aws/amazon-cloudwatch-agent/bin/config.json

After you start the cloudwatch agent, the cloudwatch log group/streams will start populating the pcmk\_alert\_file.log contents which will then be read by the metric filters to identify ‘stop’ cluster resource events.



Below is how the SNS notifications look when a cluster resource is stopped:



# 7. References

<https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html-single/high_availability_add-on_reference/index#ch-alertscripts-HAAR>

<https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/create-cloudwatch-agent-configuration-file-wizard.html>

<https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/install-CloudWatch-Agent-commandline-fleet.html>