VERITAS™

High Availability and Disaster Recovery Solutions for SAP NetWeaver on AWS using Veritas InfoScale™ Enterprise

Last updated: 2017-11-27

Table of Contents

Abstract

This document provides guidelines for implementing a high availability solution for SAP NetWeaver workloads on Amazon Web Services (AWS) using Veritas InfoScale™ Enterprise. It is intended for the following audiences:

- People who deploy small SAP systems on AWS for development, testing, training, sandboxing, demonstration, or production purposes, and want to monitor and manage SAP NetWeaver-based products for high availability and disaster recovery
- SAP Basis and SAP Implementation consultants who are familiar with AWS and want to manage the availability of SAP systems on AWS using Veritas InfoScale Enterprise

This document does not replace any standard SAP documentation or AWS documentation. For information on basic SAP high availability configurations on AWS, see the AWS documentation at:

https://d0.awsstatic.com/enterprise-marketing/SAP/sap-on-aws-high-availability-guide.pdf

When installing SAP solutions on AWS, always refer to the standard SAP documentation and notes for the respective SAP solution:

- http://service.sap.com/instguides
- http://service.sap.com/notes

For more information about SAP on AWS, see the AWS documentation at:

http://aws.amazon.com/sap

Introduction to the Veritas InfoScale Enterprise solution

Veritas InfoScale Enterprise works across leading enterprise operating systems and server virtualization technologies. It is easy to deploy inside AWS instances, and to maintain and manage from AWS. AWS offers protection from major outages when you deploy applications in multiple Availability Zones (AZs). However, such solutions are not application-aware inside the Elastic Compute Cloud (EC2) instances, and they do not support traditional clustering for SAP application instances.

The following components must be protected and any associated single point of failure eliminated, because they are critical for application availability in a distributed SAP environment:

- Database instance
- Central Services instance
- Enqueue replication server

If an application fails, AWS either restarts the instance or redeploys it, which affects the overall Recovery Time Objective (RTO).

SAP databases and SAP Central services are installed on one instance at a time and are therefore single points of failure. Since multiple SAP application server (dialog) instances can be configured, they do not form a single point of failure. However, when an application deployed in the same AZ or across AZs fails, AWS restarts the application or redeploys the EC2 instances on which the application has failed. This impacts the overall Recovery Time Objective (RTO).

Organizations commonly use OS-specific custom scripts to monitor applications. InfoScale Enterprise supports the largest number of pre-built enterprise application agents that simplify such deployments, thereby eliminating the need for custom scripts. With intelligent failover capabilities, organizations can reduce the cost of redeploying instances by reducing the number of standby servers inside AWS.

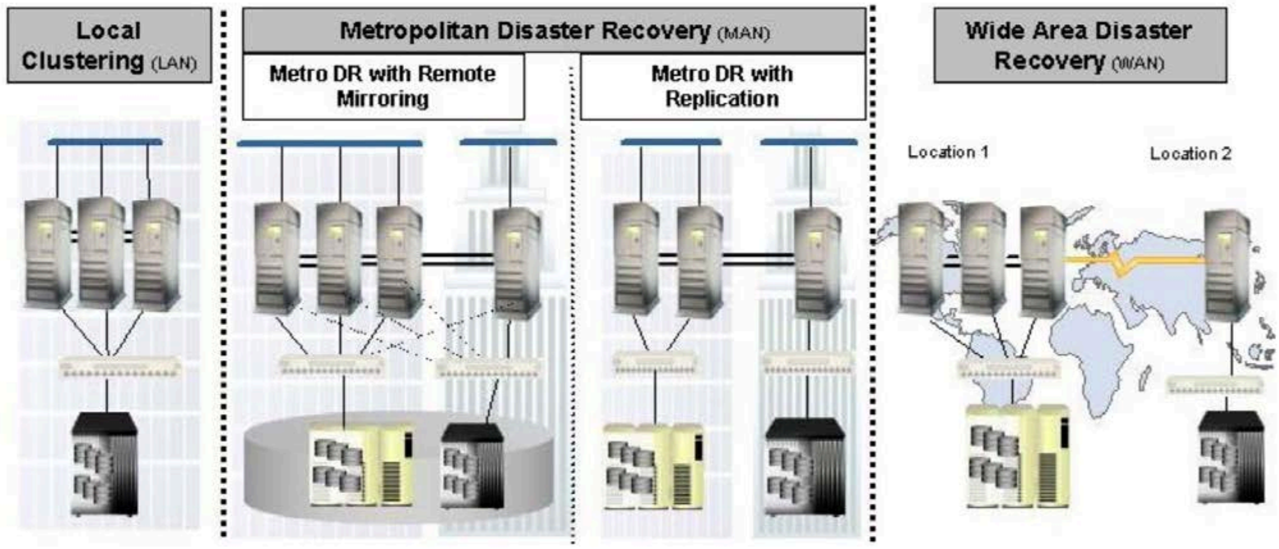
An InfoScale disaster recovery (DR) solution is simpler and faster, because it provides automated DR and non-disruptive DR testing across the AWS regions.

InfoScale Enterprise uses the following agents to provide a solution:

- AWSIP
- AWSRoute53
- SAPNW
- SAPComponents

Types of InfoScale Enterprise configurations

The following graphic depicts various availability and recovery configurations created using InfoScale Enterprise:


Local Clustering (LAN)
Metropolitan Disaster Recovery (MAN)
Metro DR with Remote Mirroring
Metro DR with Replication
Wide Area Disaster Recovery (WAN)
Location 1
Location 2

InfoScale capabilities for an SAP ecosystem

The following HA and DR capabilities of InfoScale Enterprise make it suitable for managing an SAP ecosystem:

- Business continuity with minimal application downtime during failures through complete automation for the SAP landscape
- Support for on-premises to AWS failover
- Optimal server utilization due to cost-effective failover configurations for development, testing, or production environments
- Support for cost optimization, Recovery Point Objective (RPO), and Recovery Time Objective (RTO) requirements for SAP workloads in AWS

- Support for disaster recovery (DR), which includes:
  - On-premises to AWS switchover
  - Support for new infrastructure agents in AWS to attain high availability (HA) during a disaster
  - Supported on-premises SAP agents are still relevant in AWS, because they offer:
    - Similar customer experience as on-premises
    - Flexible Storage Sharing (FSS) for data sharing in AWS across instances
    - Volume Replicator (VVR) for data movement across AWS regions or between on-premises and AWS

Replication in Veritas InfoScale Enterprise

Volume Replicator (VVR) is data-replication software designed to contribute to an effective disaster recovery plan. VVR enables you to maintain a consistent copy of application data at one or more remote locations.

VVR benefits from the robustness, ease of use, and high performance of VxVM, and at the same time, adds replication capability to VxVM. It can replicate existing VxVM configurations, and can be transparently configured while the application is active.

VVR replicates the application writes on the volumes at the source location to one or more remote locations across any distance. It provides a consistent copy of application data at the remote locations. If a disaster occurs at the source location, you can use the copy of the application data at the remote location and restart the application at the remote location.

The system on which the application is running at the source location is called the Primary system, and the one at the target location is called the Secondary system.

The volumes on the Primary system must be initially synchronized with the volumes on the Secondary system. VVR lets you initialize the application data between the primary location and the remote location using the network.

Flexible Storage Sharing in Veritas InfoScale Enterprise

The Flexible Storage Sharing (FSS) feature of InfoScale Enterprise lets you combine your distributed, high-performance, and highly available file systems with the latest storage and networking technologies. FSS lets you to unlock the true potential of Direct Attached Storage (DAS), without sacrificing performance or availability. It enables you to drive up to four times the performance at less than 20 percent of the cost of a traditional Storage Area Network (SAN) environment. FSS is not limited to a DAS-only deployment but can be used in conjunction with SAN in a hybrid deployment. Thus, you can also continue to use the AWS EBS volumes.

For details, see the Veritas FSS datasheet at:

https://www.veritas.com/content/dam/Veritas/docs/data-sheets/21327034_GA_ENT_DS-Veritas-Flexible-Storage-Sharing-EN.pdf

Considerations for using FSS or VVR across Availability Zones

Take the following considerations into account when using FSS or VVR across AZs:

- FSS can be used in active-active applications where the application can access the data on both AZs simultaneously. VVR cannot be used for active-active applications.

- When either FSS or VVR synchronous replication is used, the application throughput depends on the network characteristics, because the data needs to be mirrored or synchronized to the other AZ.
- For applications that can sustain some RPO but need high throughput, Veritas recommends using VVR asynchronous replication.
- Unlike traditional SAN storage environments, Amazon block storage such as EBS volumes can be attached to only one EC2 instance at a time. FSS can be used in AWS to provide shared storage environment to the applications running on multiple nodes in a cluster.
- VVR can be used for replication across regions in AWS.

Some of the best practices that you can adopt while deploying FSS in AWS environments to gain performance benefits are:

- Enable enhanced networking using Amazon Elastic Network Adapter (ENA) on instances to get better throughput and packet per second (PPS) performance as well as low latencies during data mirroring on volumes in FSS.

  For details on enhanced networking in AWS, see the Amazon documentation at: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/enhanced-networking-ena.html

- Choose EBS-optimized instances to get the best performance for your EBS volumes by minimizing the contention between Amazon EBS I/O and other network traffic from your instance.

  For details on EBS-optimized instances, see the Amazon documentation at: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSOptimized.html

InfoScale Enterprise agent for SAP NetWeaver

InfoScale Enterprise agents monitor specific resources within an enterprise application. They determine the status of resources and start or stop them according to external events. The InfoScale Enterprise agent for SAP NetWeaver (henceforth called the SAPNW agent) provides HA for SAP NetWeaver in a cluster. The agent brings SAP instances online, monitors them, and takes them offline. The agent monitors the system processes and server states, and can shut down the server in case of a failover.

The SAP NetWeaver agent supports the following SAP instance types:

- Central Services Instance (ASCS)
- Application Server Instance (Primary Application server and Additional Application servers)
- Enqueue Replication Server Instance (ERS)

The agent supports the following types of SAP systems:

- ABAP
- Java
- Add-In (ABAP + Java)

InfoScale Enterprise integration with SAP NetWeaver using the SAP library and the Veritas connector script

The SAP NetWeaver agent enables the integration of InfoScale Enterprise with SAP NetWeaver 7.x and SAP Kernel 7.20 DCK. For this purpose, it uses an SAP-provided library (`saphascriptco.so`) and a Veritas-provided cluster connector script (`sap_symc_cluster_connector`). This integration enables the SAP sapstartsrv component to communicate SAP instance status changes that are made by SAP clients to InfoScale Enterprise.

Note: InfoScale Enterprise integration is supported only with SAP NetWeaver 7.x and SAP Kernel 7.20 DCK or later.

In a typical InfoScale cluster, when an SAP administrator changes the status of an SAP instance using an SAP client, such as sapcontrol or startsap, the following events occur:

- When the administrator stops the SAP instance, InfoScale Enterprise detects a fault and performs the clean operation.
- When the administrator starts the SAP instance, InfoScale Enterprise detects that the instance is brought online outside of its control.

You must enable communication between sapstartsrv and InfoScale Enterprise. Doing so ensures that sapstartsrv can inform InfoScale Enterprise when an SAP client is used to start or stop an assigned SAP instance. InfoScale Enterprise can then detect the correct status of the SAP instance.

InfoScale Enterprise AWSIP agent

InfoScale Enterprise provides the AWSIP agent, which lets you monitor and manage the following networking resources in AWS:

- Private IP: A private IP is a private numerical address that networked devices use to communicate with one another.
- Elastic IP: An elastic IP address is a static IPv4 address designed for dynamic cloud computing, and it is associated with your AWS account.
- Overlay IP: AWS allows you to redirect IP address traffic to an EC2 instance in a Virtual Private Network (VPC) regardless of the subnet or AZ to which it belongs. An overlay IP lets you fail over IP addresses between cluster nodes when they are spread across multiple subnets or AZs.

For details, see the InfoScale documentation at:

https://sort.veritas.com/documents/doc_details/vie/7.3/Linux/ProductGuides/

InfoScale Enterprise AWSRoute53 agent

Amazon Route 53 is a highly available and scalable cloud Domain Name System (DNS) web service. InfoScale Enterprise provides the AWSRoute53 agent to update and monitor the mapping between host names and IP addresses. The agent does the mapping for the Amazon Route 53 domain when failing over nodes across subnets. When you create a hosted zone, Amazon Route 53 automatically creates a name server (NS) record and a start of authority (SOA) record for the zone.

If the resource records need to be dynamically added and deleted from the Amazon Route 53 domain during failover, you must use the AWSRoute53 agent. The agent updates the NS with the new resource record mappings during failover, and allows the clients to connect to the failed over instance of the application.

If you do not want to use the AWSRoute53 agent, you can continue to use the legacy DNS agent for managing DNS records.

Supported use cases for SAP NetWeaver on AWS

InfoScale Enterprise can monitor and control SAP NetWeaver-based application instances in the following scenarios:

- All SAP instances on AWS
  - SAP application instances configured in one Availability Zone
  - SAP application instances configured in multiple Availability Zones
- SAP Web Dispatcher HA using InfoScale Enterprise agents
- On-premises to AWS failover (DR)
- SAP on AWS across regions

All SAP instances on AWS

In this use case, two EC2 instances each are configured for the SAP standalone enqueue server and the SAP enqueue replication server. If the SAP central service instance (message or enqueue) fails, the SAPNW agent switches over to the replication server.

Note: The failover scenario works as per SAP high availability certification guidelines.

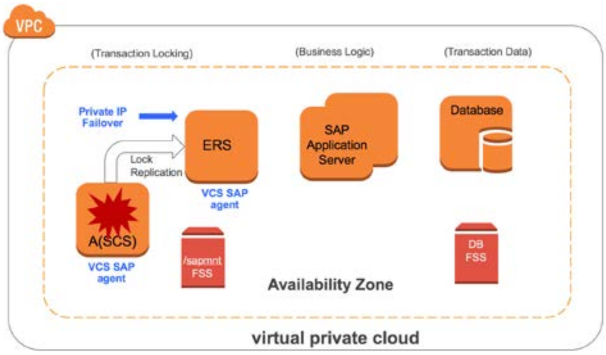
The InfoScale Enterprise SAPNW agent supports the following failover scenarios for enqueue server and enqueue replication server.

SAP application instances configured in one Availability Zone

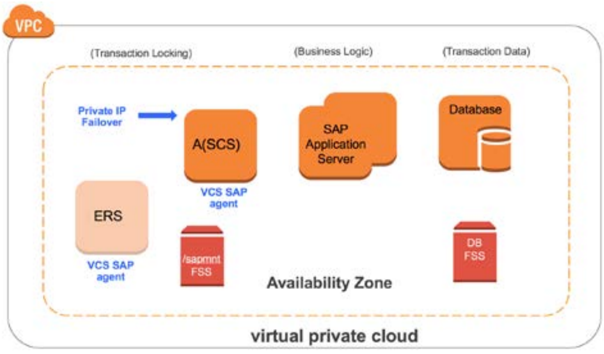
Install and configure an SAP Central Service instance with a virtual host name and ensure that the virtual host name is resolvable from all the other SAP instance hosts. In this use case enqueue server and enqueue replication server instances exist in the same AZ.

After you set the enqueuer or the replication parameter to TRUE (enque/server/replication = true), all the enqueue locks are replicated to the enqueue server and are available on the replication server. If the enqueue server instance fails or becomes unavailable, the SAPNW agent detects the fault and automatically triggers the failover of the enqueue server to the enqueue replication server node. Thus, the enqueue replication server is converted to the enqueue server.

The following graphics depicts an SAP ASCS instance fault has occurred:



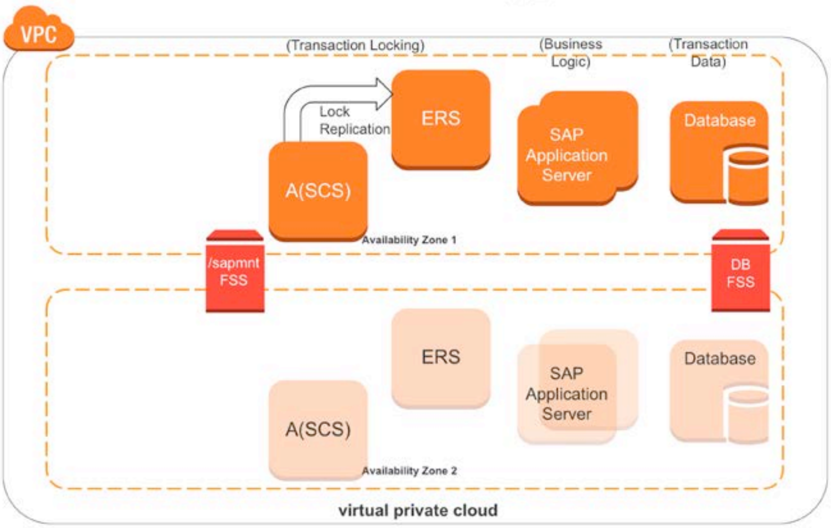
The following graphic depicts that the enqueue replication server is converted to the enqueue server after failover:



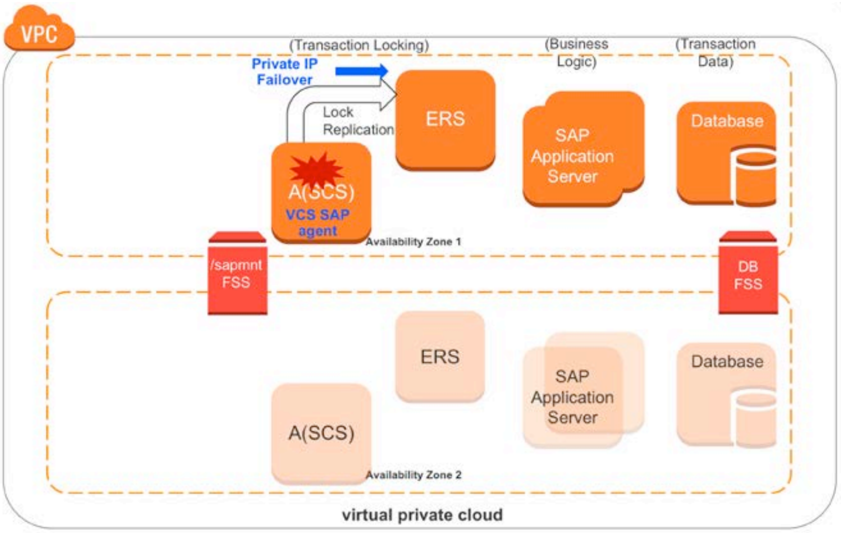
After the fault is cleared and the failback is complete, the enqueue replication server starts replicating the enqueue locks from enqueue server.

SAP application instances configured in multiple Availability Zones

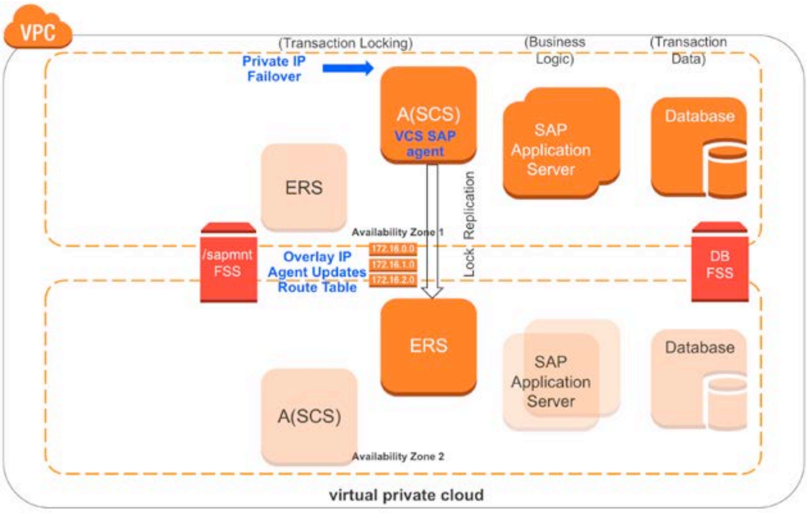
In this use case, the SAP enqueue and SAP enqueue replication instances are initially running in Availability Zone 1. The SAP dialog and SAP database instances are either in the same AZ or spread across multiple AZs.



If the SAP enqueue server fails, the SAPNW agent automatically fails over to the enqueue replication server and loads the replicated enqueue locks from the enqueue replication server. In this case, the InfoScale Enterprise AWSIP and IP agents manage the virtual IP failover.



The following graphic depicts that the enqueue replication server is active and the enqueue transaction locks are being replicated to Availability Zone 2:



After the Faulted state is cleared and maintenance is done on the previous enqueue (ASCS) server, you can restart the enqueue replication server on the previous enqueque server.

You can switch over the enqueue and enqueue replication instances across AZs without any loss of enqueue transaction locks by using the SAPNW and SAPComponents agents.

The InfoScale Enterprise application agents manage to eliminate a single point of failure for the SAP database and SAP dialog instances across AZs.

For details, see the *Cluster Server Agent for SAP NetWeaver Installation and Configuration Guide* at:

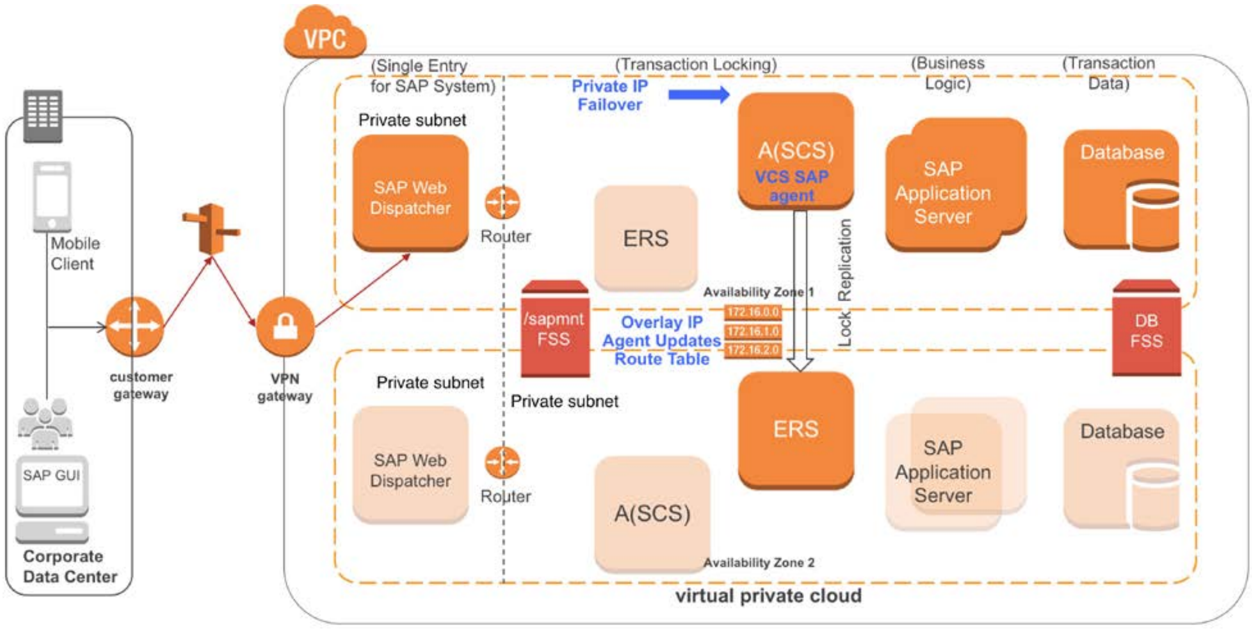
https://sort.veritas.com/agents/detail/7018

SAP Web Dispatcher HA using InfoScale Enterprise agents

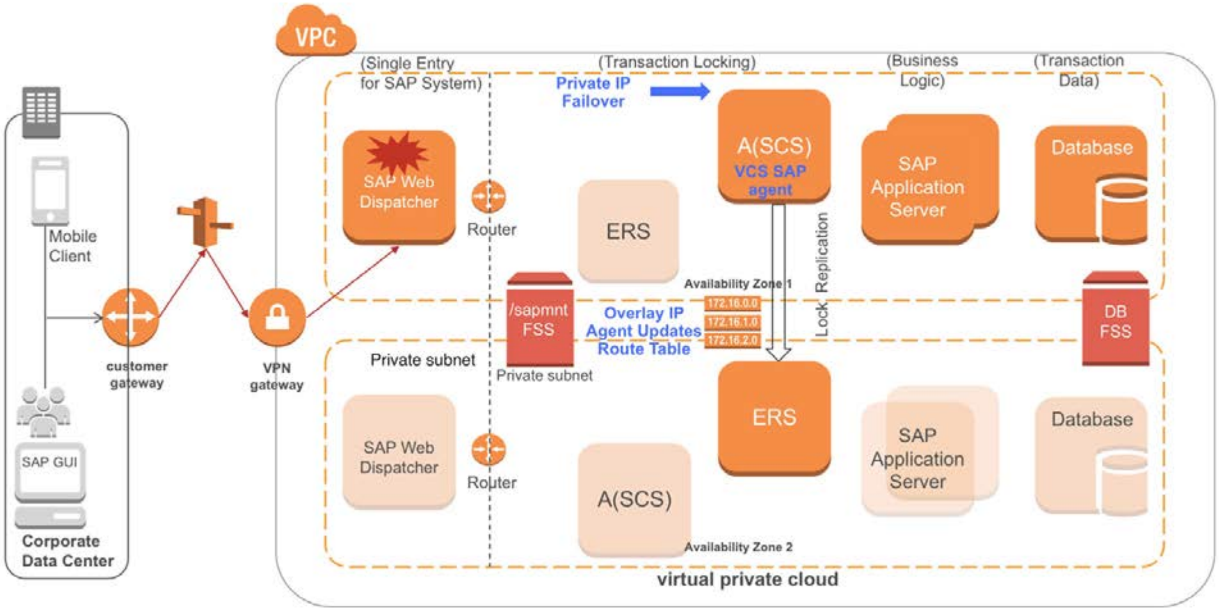
The SAPComponents agent and the AWSRoute53 agent work together to monitor and control SAP Web Dispatcher instances across AZs. In this use case, SAP Web Dispatcher is configured and managed with a private IP instead of the Amazon elastic IP. The virtual IPs are customized in different AZs with the same hostname.

If SAP Web Dispatcher becomes unavailable in an AZ, the SAPComponents agent fails it over to another AZ using same virtual hostname. In this scenario, clients can connect with the same hostname across AZs and this can be managed by the AWSRoute53 agent.

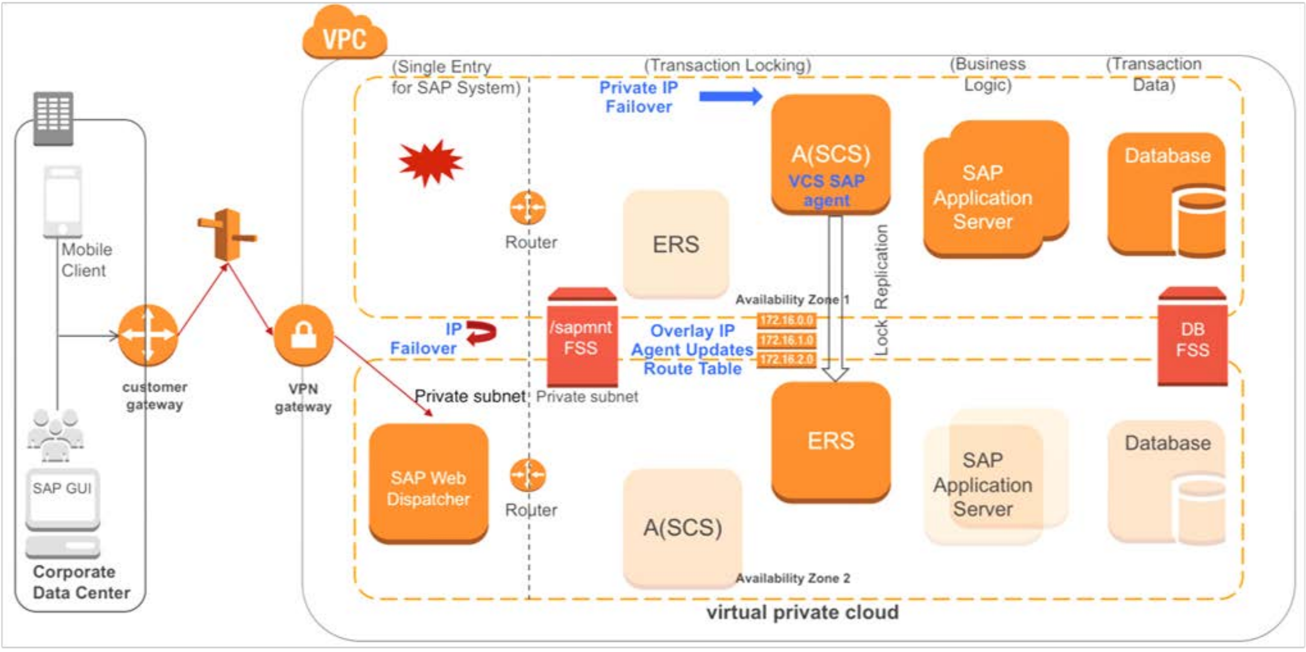
The following graphic depicts a typical SAP landscape that spans multiple AZs. The SAP Web Dispatcher instance is active in Availability Zone 1 and in standby mode in Availability Zone 2.



The following graphic depicts that the SAPComponents agent has detected a fault and is ready to fail over the instance to another AZ.



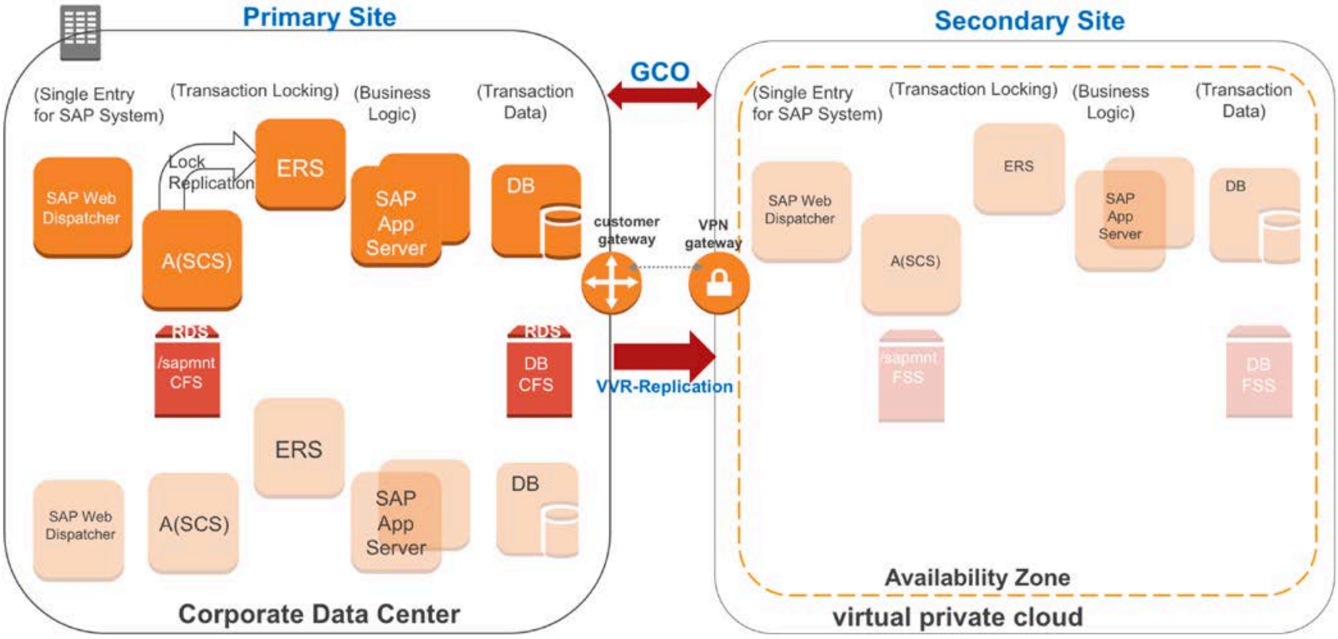
The following graphic depicts that the SAP Web Dispatcher has failed over to Availability Zone 2. The AWSRoute53 agent follows the failover policy and changes the DNS records for the Web Dispatcher host name. This ensures easy of connectivity for web-based clients.



On-premises to AWS failover (DR)

In this use case, all the SAP application instances are on-premises (primary site). Shared components like sapmnt, trans, database data volumes and log volumes are configured with InfoScale CFS. VVR manages the replication of data across sites. On AWS (secondary site), the storage volumes are allocated from Amazon EBS and shared components are configured with FSS.

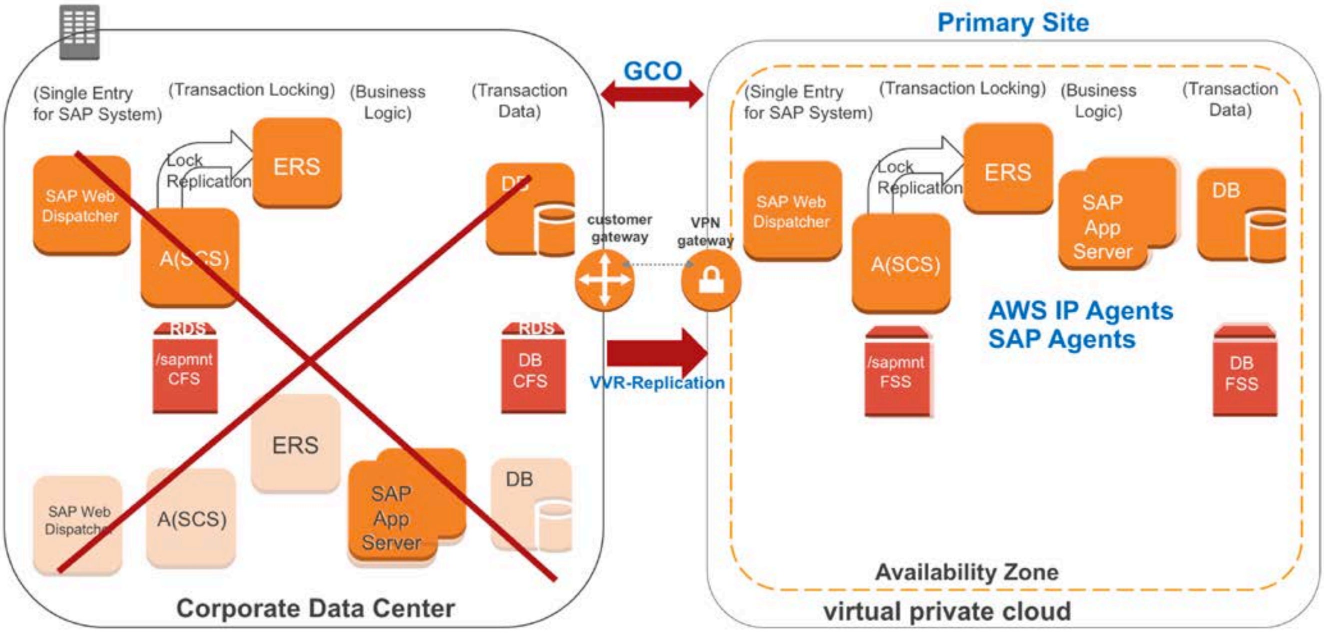
The following graphic depicts a DR scenario with replication between on-premises and AWS managed by VVR:



When a disaster occurs at the primary site, all the SAP application and database instances are migrated to AWS. This can be triggered automatically or done manually by bringing the SAP applications online in AWS, which then becomes the primary site.

The AWSIP and AWSRoute53 agents manage the virtual IPs and virtual hostnames in AWS.

The following graphic depicts the environment after failover:



Configuring DR between on-premises and AWS

To configure a DR environment between an on-premises site and the AWS cloud, perform the following tasks:

1. Configure a VPN to connect the corporate data center and the AWS virtual private cloud.
2. Install and configure Veritas InfoScale Enterprise on all the systems that must be part of the cluster.

   For specific instructions, see the *Veritas InfoScale Installation Guide*.

3. Allocate SAP-recommended LUNs, and create CFS disk groups and volumes using VxVM and VxFS commands.
4. Mount the following SAP mount points:

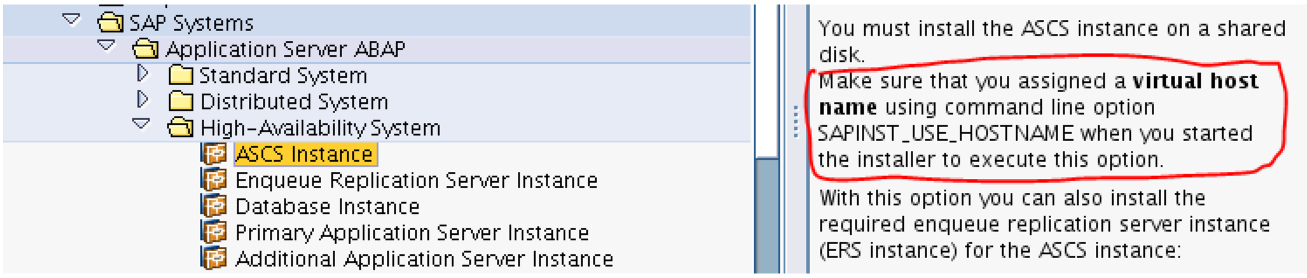
   ```
   /sapmnt/<SID>
   /usr/sap/trans
   /usr/sap/<SID>/ASCSxx
   /usr/sap/<SID>/ERSxx
   /usr/sap/<SID>/DVEBMGSxx
   /usr/sap/<SID>/Dxx
   ```

   For specific instructions, see the following Veritas documentation *on the* SORT *website*:

   - *Storage Foundation and High Availability Configuration and Upgrade Guide*
   - *Storage Foundation Cluster File System High Availability Administrator's Guide*
   - *Storage Foundation Cluster File System High Availability Configuration and Upgrade Guide*
   - *Veritas InfoScale Disaster Recovery Implementation Guide*

5. Install and configure the SAP stack on the on-premises systems.

   Enterprises can run SAP applications like ERP, CRM, SCM, SRM, and the complete SAP NetWeaver ABAP, JAVA, and ADDIN technology stacks on AWS. InfoScale Enterprise provides HA and the desired software-defined storage (SDS) capabilities for SAP NetWeaver on AWS.
   While deploying SAP NetWeaver, you must choose the High Availability option with virtual hostnames, as follows:
   `SAPINST_USE_HOSTNAME=<virtual_host_name>`
   Ensure that individual instances are configured with different host names:



You must install the ASCS instance on a shared disk.
Make sure that you assigned a **virtual host name** using command line option SAPINST_USE_HOSTNAME when you started the installer to execute this option.
With this option you can also install the required enqueue replication server instance (ERS instance) for the ASCS instance:

   Refer to the SAP documentation for creating a similar SAP environment in AWS.

6. Configure replication between the on-premises site and AWS. Ensure that all the required ports from your data center to AWS are enabled for replication.

   For specific instructions, see the *Veritas InfoScale Replication Administrator's Guide*.

7. Prepare the could environment as follows:

    1. Create the Amazon EC2 instances.
    2. Attach the required SSD or standard Amazon EBS volumes.
    3. Configure disk groups and volumes using the VxVM, VxFS, and FSS commands.
    4. Configure FSS for data sharing between EBS volumes on the EC2 instances.
    5. Mount all the SAP shared volumes and the SAP database volumes.

    For specific instructions, see the *Storage Foundation Cluster File System High Availability Administrator's Guide*.
    Veritas used the following EBS volume types for testing the SAP NetWeaver configurations in AWS:

    - gp2 (SSD)
    - Standard (magnetic disks)

    However, FSS supports all the EBS volume types.

8. Configure InfoScale cluster service groups and resources for the SAP application instances, the AWSIP agent, and the AWSRoute53 agent.

    See “Sample configurations” on page 16.

9. Initially, bring the entire SAP stack online in the on-premises data center.
10. Ensure that the initial data synchronization is complete and that VVR has started replicating data to AWS (secondary site).

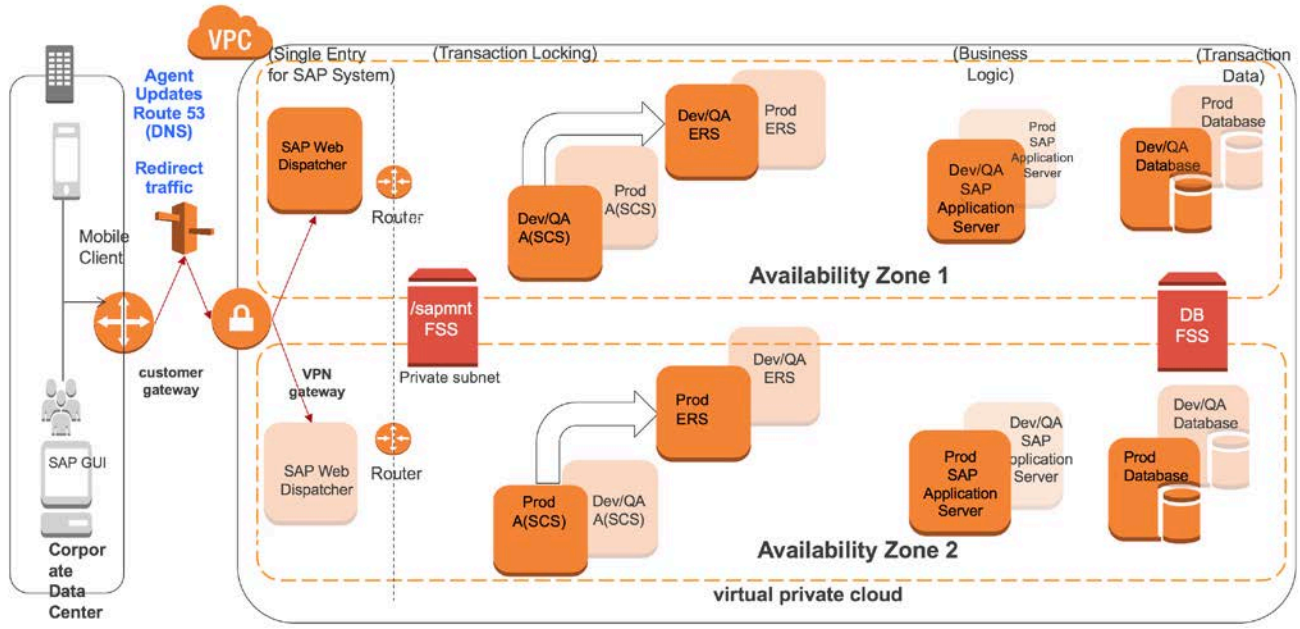
SAP on AWS across regions

Failover scenarios in AWS across different regions are congruent to the failover scenario between an on-premises site and AWS. So, you can set up similar configurations for failover across AWS regions too, and use VVR for data replication and protection. Doing so lets you achieve better RPO and RTO in case of disaster at an active site (region). You can continue to use FSS to manage storage configurations within an AWS region.

Optimization of SAP instances inside AWS

AWS lets you create and optimize SAP instances for development, testing, or production environments. If an SAP application instance outage occurs, InfoScale Enterprise fails over the instances between the designated SAP systems without disrupting the client connections. Thus, it reduces the overall Total Cost of Ownership (TCO) in case of a disaster or an outage of SAP instances on AWS.

The following graphic depicts that if an SAP production application instance fails, InfoScale Enterprise detects the failed instance and moves that particular instance to either development or testing nodes.



⇒ Active instance

⇒ Standby instance

However, you must ensure that the development or testing instances cannot be moved to production instance nodes. Development or testing instances can fail over or switch over among each other.

Sample configurations

This section lists the sample configurations for an InfoScale cluster that provides disaster recovery (DR) between an on-premises site and AWS for SAP. Each configuration includes service groups and resources for the SAP application instances, the AWSIP agent, and the AWSRoute53 agent.

Sample configuration for an on-premises InfoScale cluster

The following code snippet is a sample configuration file (`main.cf`) for an on-premises InfoScale cluster:

```
include "OracleASMTypes.cf"

include "types.cf"

include "CFSTypes.cf"

include "CRSResource.cf"

include "CSSD.cf"

include "CVMTypes.cf"

include "Db2udbTypes.cf"

include "MultiPrivNIC.cf"
include "OracleTypes.cf"
include "PrivNIC.cf"
include "SAPNWTypes.cf"
include "SybaseTypes.cf"

cluster sapaws_onprem (
      UserNames = { admin = aPQiPKpMQlQQoYQkPN }
      ClusterAddress = "10.209.58.229"
      Administrators = { admin }
      HacliUserLevel = COMMANDROOT
      )

remotecluster sapawsclus (
      ClusterAddress = "10.239.2.75"
      ConnectTimeout = 3000
      SocketTimeout = 3000
      )

heartbeat Icmp (
      ClusterList = { sapawsclus }
      Arguments @sapawsclus = { "10.239.2.75" }
      )

system saprhe7 (
      )

system saprhel27 (
      )

group ClusterService (
      SystemList = { saprhel27 = 0, saprhe7 = 1 }
```

```
AutoStartList = { saprhel27, saprhe7 }

ClusterFailOverPolicy = Manual

OnlineRetryLimit = 3

OnlineRetryInterval = 120

)


Application wac (

      StartProgram = "/opt/VRTSvcs/bin/wacstart"

      StopProgram = "/opt/VRTSvcs/bin/wacstop"

      MonitorProcesses = { "/opt/VRTSvcs/bin/wac" }

      RestartLimit = 3

      )


IP webip (

      Device = ens192

      Address = "10.209.58.229"

      NetMask = "255.255.252.0"

      )


NIC csgnic (

      Device = ens192

      )


wac requires webip

webip requires csgnic



// resource dependency tree

//

//    group ClusterService

//    {
      //    Application wac
      //        {
      //        IP webip
      //            {
      //            NIC csgnic
      //            }
      //        }
      //    }



group Oracle_Database (
      SystemList = { saprhe7 = 1, saprhel27 = 0 }
      ClusterList = { sapawsclus = 1, sapaws_onprem = 0 }
      ClusterFailOverPolicy = Manual
      )

      IP IP (
            Critical = 0
            Device = ens192
            Address = "10.209.58.226"
            NetMask = "255.255.252.0"
            )

      Mount SAP_DB_Mount (
            Critical = 0
            MountPoint = "/oracle"
            BlockDevice = "/dev/vx/dsk/sapdbdg/sapdbdg_vol"
            FSType = vxfs
            MountOpt = rw
            FsckOpt = "-y"
            )
```

```
NIC NIC (
      Enabled = 0
      Device = ens192
      )

Netlsnr Listener (
      Critical = 0
      Owner = oraqas
      Home = "/oracle/QAS/121"
      )

Oracle Oracle_resource (
      Critical = 0
      Sid = QAS
      Owner = oraqas
      Home = "/oracle/QAS/121"
      )

requires group RVG_Owner_SG online local firm
IP requires NIC
Listener requires Oracle_resource
Oracle_resource requires IP
Oracle_resource requires SAP_DB_Mount



// resource dependency tree
//
//    group Oracle_Database
//    {
//    Netlsnr Listener
```

```
      //        {
      //        Oracle Oracle_resource
      //            {
      //            IP IP
      //                {
      //                NIC NIC
      //                }
      //            Mount SAP_DB_Mount
      //            }
      //        }
      //    }


group RVG_Owner_SG (
      SystemList = { saprhel27 = 0, saprhe7 = 1 }
      )

      IP LogOwner_IP_1 (
            Device = ens192
            Address = "10.209.58.225"
            NetMask = "255.255.252.0"
            )

      NIC NIC_1 (
            Device = ens192
            )

      requires group RVGgroup online local firm
      LogOwner_IP_1 requires NIC_1
      // resource dependency tree
      //
      //    group RVG_Owner_SG
      //    {
      //    IP LogOwner_IP_1
      //        {
      //        NIC NIC_1
      //        }
      //    }


group RVGgroup (
      SystemList = { saprhel27 = 0, saprhe7 = 1 }
      Parallel = 1
      AutoStartList = { saprhel27, saprhe7 }
      )

      CVMVolDg DB_DiskGroup (
            Critical = 0
            CVMDiskGroup = sapdbdg
            CVMVolume = { sapdbdg_vol, sapsrl_vol }
            CVMActivation = sw
            )

      RVGShared data_rvg (
            Critical = 0
            RVG = sapdbrvg
            DiskGroup = sapdbdg
            )

      requires group cvm online local firm
      data_rvg requires DB_DiskGroup



      // resource dependency tree
      //
      //    group RVGgroup
      //    {
      //    RVGShared data_rvg
      //        {
      //        CVMVolDg DB_DiskGroup
      //        }
      //    }



group cvm (
      SystemList = { saprhel27 = 0, saprhe7 = 1 }
      AutoFailOver = 0
      Parallel = 1
      AutoStartList = { saprhel27, saprhe7 }
      )

      CFSMount sapmnt_res (
            Critical = 0
            MountPoint = "/sapmnt"
            BlockDevice = "/dev/vx/dsk/sapmntdg/sapmntdg_vol"
            MountOpt = rw
            )

      CFSfsckd vxfsckd (
            )
```

```
CVMCluster cvm_clus (
      CVMClustName = sapaws_onprem
      CVMNodeId = { saprhel27 = 0, saprhe7 = 1 }
      CVMTransport = gab
      CVMTimeout = 200
      )

CVMVolDg sapmntdg_res (
      CVMDiskGroup = sapmntdg
      CVMVolume = { sapmntdg_vol }
      CVMActivation = sw
      )

CVMVxconfigd cvm_vxconfigd (
      Critical = 0
      CVMVxconfigdArgs = { syslog }
      )

ProcessOnOnly vxattachd (
      Critical = 0
      PathName = "/bin/sh"
      Arguments = "- /usr/lib/vxvm/bin/vxattachd root"
      RestartLimit = 3
      )

cvm_clus requires cvm_vxconfigd
sapmnt_res requires sapmntdg_res
sapmnt_res requires vxfsckd
vxfsckd requires cvm_clus
```

```
      // resource dependency tree
      //
      //    group cvm
      //    {
      //    CFSMount sapmnt_res
      //        {
      //        CVMVolDg sapmntdg_res
      //        CFSfsckd vxfsckd
      //            {
      //            CVMCluster cvm_clus
      //                {
      //                CVMVxconfigd cvm_vxconfigd
      //                }
      //            }
      //        }
      //    ProcessOnOnly vxattachd
      //    }



group sap_aas_sg (
      SystemList = { saprhel27 = 0, saprhe7 = 1 }
      ClusterList = { sapawsclus = 1, sapaws_onprem = 0 }
      ClusterFailOverPolicy = Manual
      )

      IP sap_aas_ip (
            Device = ens192
            Address = "10.209.58.228"
            NetMask = "255.255.252.0"
            )
```

```
NIC sap_aas_nic (
      Device = ens192
      )

SAPNW sap_aas_res (
      Critical = 0
      ResLogLevel = TRACE
      InstProfile = "/usr/sap/QAS/SYS/profile/QAS_D02_qasdi"
      SAPAdmin = qasadm
      ProcMon = dw
      EnvFile = "/home/qasadm/.login"
      )

SAPNW sap_aas_sapstartsrv (
      ResLogLevel = TRACE
      InstType = SAPSTARTSRV
      InstProfile = "/usr/sap/QAS/SYS/profile/QAS_D02_qasdi"
      SAPAdmin = qasadm
      ProcMon = sapstartsrv
      EnvFile = "/home/qasadm/.login"
      )

requires group Oracle_Database online global soft
sap_aas_ip requires sap_aas_nic
sap_aas_res requires sap_aas_sapstartsrv
sap_aas_sapstartsrv requires sap_aas_ip


// resource dependency tree
//
//    group sap_aas_sg
```

```
      //    {
      //    SAPNW sap_aas_res
      //        {
      //        SAPNW sap_aas_sapstartsrv
      //            {
      //            IP sap_aas_ip
      //                {
      //                NIC sap_aas_nic
      //                }
      //            }
      //        }
      //    }


group sap_ascs_sg (
      SystemList = { saprhe7 = 0, saprhel27 = 1 }
      ClusterList = { sapawsclus = 1, sapaws_onprem = 0 }
      ClusterFailOverPolicy = Manual
      PreOnline = 1
      )

      IP sap_ascs_ip (
            Critical = 0
            Device = ens192
            Address = "10.209.58.222"
            NetMask = "255.255.252.0"
            )

      NIC sap_ascs_nic (
            Critical = 0
            Device = ens192
```

```
        )

SAPNW sap_ascs_res (
        ResLogLevel = TRACE
        InstType = ENQUEUE
        InstProfile = "/usr/sap/QAS/SYS/profile/QAS_ASCS00_qasascs"
        SAPAdmin = qasadm
        ProcMon = "en ms"
        EnvFile = "/home/qasadm/.login"
        )

SAPNW sap_ascs_sapstartsrv (
        Critical = 0
        ResLogLevel = TRACE
        InstType = SAPSTARTSRV
        InstProfile = "/usr/sap/QAS/SYS/profile/QAS_ASCS00_qasascs"
        SAPAdmin = qasadm
        ProcMon = sapstartsrv
        EnvFile = "/home/qasadm/.login"
        ToleranceLimit = 5
        RestartLimit = 5
        )

requires group cvm online local firm
sap_ascs_ip requires sap_ascs_nic
sap_ascs_res requires sap_ascs_sapstartsrv
sap_ascs_sapstartsrv requires sap_ascs_ip



// resource dependency tree
//
```

```
      //    group sap_ascs_sg
      //    {
      //    SAPNW sap_ascs_res
      //        {
      //        SAPNW sap_ascs_sapstartsrv
      //            {
      //            IP sap_ascs_ip
      //                {
      //                NIC sap_ascs_nic
      //                }
      //            }
      //        }
      //    }


group sap_ers_sg (
      SystemList = { saprhe7 = 0, saprhel27 = 1 }
      AutoStart = 0
      ClusterList = { sapawsclus = 1, sapaws_onprem = 0 }
      Authority = 1
      ClusterFailOverPolicy = Manual
      PreOnline = 1
      AutoRestart = 0
      )

      IP sap_ers_ip (
            Device = ens192
            Address = "10.209.58.224"
            NetMask = "255.255.252.0"
            )
```

```
NIC sap_ers_nic (
      Device = ens192
      )

SAPNW sap_ers_res (
      ResLogLevel = TRACE
      InstType = ENQREP
      InstProfile = "/usr/sap/QAS/SYS/profile/QAS_ERS10_qasers"
      SAPAdmin = qasadm
      EnqSrvResName = sap_ascs_res
      ProcMon = er
      EnvFile = "/home/qasadm/.login"
      )

SAPNW sap_ers_sapstartsrv (
      InstType = SAPSTARTSRV
      InstProfile = "/usr/sap/QAS/SYS/profile/QAS_ERS10_qasers"
      SAPAdmin = qasadm
      ProcMon = sapstartsrv
      EnvFile = "/home/qasadm/.login"
      )

requires group cvm online local firm
sap_ers_ip requires sap_ers_nic
sap_ers_res requires sap_ers_sapstartsrv
sap_ers_sapstartsrv requires sap_ers_ip



// resource dependency tree
//
//    group sap_ers_sg
```

```
      //    {
      //    SAPNW sap_ers_res
      //        {
      //        SAPNW sap_ers_sapstartsrv
      //            {
      //            IP sap_ers_ip
      //                {
      //                NIC sap_ers_nic
      //                }
      //            }
      //        }
      //    }



group sap_pas_sg (
      SystemList = { saprhe7 = 0, saprhel27 = 1 }
      ClusterList = { sapawsclus = 1, sapaws_onprem = 0 }
      ClusterFailOverPolicy = Manual
      )

      IP sap_pas_ip (
            Critical = 0
            Device = ens192
            Address = "10.209.58.230"
            NetMask = "255.255.252.0"
            )

      NIC sap_pas_nic (
            Critical = 0
            Device = ens192
            )
```

```
SAPNW sap_pas_res (
      ResLogLevel = TRACE
      InstProfile = "/usr/sap/QAS/SYS/profile/QAS_DVEBMGS01_qaspas"
      SAPAdmin = qasadm
      ProcMon = dw
      EnvFile = "/home/qasadm/.login"
      )

SAPNW sap_pas_sapstartsrv (
      Critical = 0
      ResLogLevel = TRACE
      InstType = SAPSTARTSRV
      InstProfile = "/usr/sap/QAS/SYS/profile/QAS_DVEBMGS01_qaspas"
      SAPAdmin = qasadm
      ProcMon = sapstartsrv
      EnvFile = "/home/qasadm/.login"
      )

requires group Oracle_Database online global soft
sap_pas_ip requires sap_pas_nic
sap_pas_res requires sap_pas_sapstartsrv
sap_pas_sapstartsrv requires sap_pas_ip



// resource dependency tree
//
//    group sap_pas_sg
//    {
//    SAPNW sap_pas_res
//        {
//        SAPNW sap_pas_sapstartsrv
//            {
//            IP sap_pas_ip
//                {
//                NIC sap_pas_nic
//                }
//            }
//        }
//    }
```

Sample configuration for an InfoScale cluster in AWS

The following code snippet is a sample configuration file (main.cf) for an InfoScale cluster in AWS:

```
include "types.cf"
include "CFSTypes.cf"
include "CRSResource.cf"
include "CSSD.cf"
include "CVMTypes.cf"
include "Db2udbTypes.cf"
include "MultiPrivNIC.cf"
include "OracleTypes.cf"
include "PrivNIC.cf"
include "SAPNWTypes.cf"
include "SybaseTypes.cf"

cluster sapawsclus (
      UserNames = { admin = GPQiPKpMQlQQoYQkPN }
      ClusterAddress = "10.239.2.75"
      Administrators = { admin }
      )

remotecluster sapaws_onprem (
      ClusterAddress = "10.209.58.229"
      ConnectTimeout = 3000
      SocketTimeout = 3000
      )

heartbeat Icmp (
      ClusterList = { sapaws_onprem }
      Arguments @sapaws_onprem = { "10.209.58.229" }
      )

system ip-10-239-2-104 (
      )
system ip-10-239-2-201 (
      )
system ip-10-239-3-193 (
      )
system ip-10-239-3-203 (
      )
system ip-10-239-3-238 (
      )
system ip-10-239-3-251 (
      )
group ClusterService (
      SystemList = { ip-10-239-3-251 = 0, ip-10-239-3-238 = 1,
              ip-10-239-3-203 = 2,
              ip-10-239-2-104 = 3,
              ip-10-239-2-201 = 4,
              ip-10-239-3-193 = 5 }
      AutoStartList = { ip-10-239-3-251, ip-10-239-3-238, ip-10-239-2-104,
              ip-10-239-3-203,
              ip-10-239-2-201,
              ip-10-239-3-193 }
      OnlineRetryLimit = 3
      OnlineRetryInterval = 120
      )

      AWSIP awsgcoip (
            PrivateIP = "10.239.2.75"
            Device = eth0
            AWSBinDir = "/usr/local/bin"
            )

      Application wac (
            StartProgram = "/opt/VRTSvcs/bin/wacstart"
            StopProgram = "/opt/VRTSvcs/bin/wacstop"
            MonitorProcesses = { "/opt/VRTSvcs/bin/wac" }
            RestartLimit = 3
            )

      IP webip (
            Device = eth0
            Address = "10.239.2.75"
            NetMask = "255.255.254.0"
            )

      NIC csgnic (
            Device = eth0
            )

      awsgcoip requires webip
      wac requires webip
      webip requires csgnic
```

```
    // resource dependency tree
    //
    //    group ClusterService
    //    {
    //    AWSIP awsgcoip
    //        {
    //        IP webip
    //            {
    //            NIC csgnic
    //            }
    //        }
    //    Application wac
    //        {
    //        IP webip
    //            {
    //            NIC csgnic
    //            }
    //        }
    //    }


group Oracle_Database (
    SystemList = { ip-10-239-3-251 = 0, ip-10-239-3-238 = 1 }
    AutoFailOver = 0
    ClusterList = { sapawsclus = 1, sapaws_onprem = 0 }
    Authority = 1
    AutoStartList = { ip-10-239-3-238, ip-10-239-3-251 }
    ClusterFailOverPolicy = Manual
    AutoRestart = 0
```

```
        )

        AWSIP DB_AWS_IP (
                PrivateIP = "10.239.2.72"
                Device = eth0
                AWSBinDir = "/usr/local/bin"
                )

        IP DB_IP (
                Critical = 0
                Device = eth0
                Address = "10.239.2.72"
                NetMask = "255.255.254.0"
                )

        Mount SAP_DB_Mount (
                Critical = 0
                MountPoint = "/oracle"
                BlockDevice = "/dev/vx/dsk/sapdbdg/sapdbdg_vol"
                FSType = vxfs
                MountOpt = rw
                FsckOpt = "-y"
                )

        NIC NIC (
                Device = eth0
                )

        Netlsnr Listener (
                Critical = 0
                Owner = qasadm
```

```
            Home = "/oracle/QAS/121"
            )

    Oracle Oracle_resource (
            Critical = 0
            Sid = QAS
            Owner = oraqas
            Home = "/oracle/QAS/121"
            )

    requires group RVG_Owner_SG online local firm
    DB_AWS_IP requires NIC
    DB_IP requires DB_AWS_IP
    Listener requires Oracle_resource
    Oracle_resource requires DB_IP
    Oracle_resource requires SAP_DB_Mount



    // resource dependency tree
    //
    //      group Oracle_Database
    //      {
    //      Netlsnr Listener
    //          {
    //          Oracle Oracle_resource
    //              {
    //              Mount SAP_DB_Mount
    //              IP DB_IP
    //                  {
    //                  AWSIP DB_AWS_IP
    //                      {
      //                    NIC NIC
      //                    }
      //                }
      //            }
      //        }
      //    }



group RVG_Owner_SG (
      SystemList = { ip-10-239-3-251 = 0, ip-10-239-3-238 = 1 }
      AutoStartList = { ip-10-239-3-251, ip-10-239-3-238 }
      )

      AWSIP LogOwner_AWS_IP_1 (
            PrivateIP = "10.239.2.76"
            Device = eth0
            AWSBinDir = "/usr/local/bin"
            )

      IP LogOwner_IP_1 (
            Device = eth0
            Address = "10.239.2.76"
            NetMask = "255.255.254.0"
            )

      NIC NIC_1 (
            Device = eth0
            )

      requires group RVGgroup online local firm
      LogOwner_AWS_IP_1 requires NIC_1
```

```
        LogOwner_IP_1 requires LogOwner_AWS_IP_1



        // resource dependency tree
        //
        //    group RVG_Owner_SG
        //    {
        //    IP LogOwner_IP_1
        //        {
        //        AWSIP LogOwner_AWS_IP_1
        //            {
        //            NIC NIC_1
        //            }
        //        }
        //    }



group RVGgroup (
        SystemList = { ip-10-239-3-251 = 0, ip-10-239-3-238 = 1 }
        Parallel = 1
        AutoStartList = { ip-10-239-3-251, ip-10-239-3-238 }
        )

        CVMVolDg DB_DiskGroup (
                Critical = 0
                CVMDiskGroup = sapdbdg
                CVMActivation = sw
                )

        requires group cvm online local hard
```

```
      // resource dependency tree
      //
      //      group RVGgroup
      //      {
      //      CVMVolDg DB_DiskGroup
      //      }



group cvm (
      SystemList = { ip-10-239-3-251 = 0, ip-10-239-3-238 = 1,
              ip-10-239-2-104 = 2,
              ip-10-239-3-203 = 3,
              ip-10-239-2-201 = 4,
              ip-10-239-3-193 = 5 }
      AutoFailOver = 0
      Parallel = 1
      AutoStartList = { ip-10-239-3-251, ip-10-239-3-238, ip-10-239-2-104,
              ip-10-239-3-203,
              ip-10-239-2-201,
              ip-10-239-3-193 }
      )

      CFSMount sapmnt_res (
            Critical = 0
            MountPoint = "/sapmnt"
            BlockDevice = "/dev/vx/dsk/sapmntdg/sapmntdg_vol"
            MountOpt = rw
            RestartLimit = 5
            ToleranceLimit = 5
            )
```

```
CFSfsckd vxfsckd (
      )

CVMCluster cvm_clus (
      CVMClustName = sapawsclus
      CVMNodeId = { ip-10-239-2-104 = 2, ip-10-239-3-203 = 3,
              ip-10-239-3-238 = 1,
              ip-10-239-3-251 = 0,
              ip-10-239-2-201 = 4,
              ip-10-239-3-193 = 5 }
      CVMTransport = gab
      CVMTimeout = 200
      )

CVMVolDg sapmnt_dg (
      CVMDiskGroup = sapmntdg
      CVMVolume = { sapmntdg_vol }
      CVMActivation = sw
      )

CVMVxconfigd cvm_vxconfigd (
      Critical = 0
      CVMVxconfigdArgs = { syslog }
      )

ProcessOnOnly vxattachd (
      Critical = 0
      PathName = "/bin/sh"
      Arguments = "- /usr/lib/vxvm/bin/vxattachd root"
      RestartLimit = 3
      )

cvm_clus requires cvm_vxconfigd

sapmnt_dg requires cvm_clus

sapmnt_res requires sapmnt_dg

sapmnt_res requires vxfsckd

vxfsckd requires cvm_clus



// resource dependency tree

//

//    group cvm

//    {

//    CFSMount sapmnt_res

//        {

//        CVMVolDg sapmnt_dg

//            {

//            CVMCluster cvm_clus

//                {

//                CVMVxconfigd cvm_vxconfigd

//                }

//            }

//        CFSfsckd vxfsckd

//            {

//            CVMCluster cvm_clus

//                {

//                CVMVxconfigd cvm_vxconfigd

//                }

//            }

//        }

//    ProcessOnOnly vxattachd
```

```
        //      }



group sap_aas_sg (
        SystemList = { ip-10-239-3-193 = 0, ip-10-239-3-238 = 1 }
        ClusterList = { sapawsclus = 1, sapaws_onprem = 0 }
        Authority = 1
        ClusterFailOverPolicy = Manual
        )

        AWSIP AAS_AWS_IP (
                Critical = 0
                PrivateIP = "10.239.2.74"
                Device = eth0
                AWSBinDir = "/usr/local/bin"
                )

        IP sap_aas_ip (
                Critical = 0
                Device = eth0
                Address = "10.239.2.74"
                NetMask = "255.255.254.0"
                )

        NIC sap_aas_nic (
                Device = eth0
                )

        SAPNW sap_aas_res (
                Critical = 0
                ResLogLevel = TRACE
                EnvFile = "/home/qasadm/.login"
                SAPAdmin = qasadm
                InstProfile = "/usr/sap/QAS/SYS/profile/QAS_D02_qasdi"
                ProcMon = dw
                )

        SAPNW sap_aas_sapstartsrv (
                Critical = 0
                ResLogLevel = TRACE
                EnvFile = "/home/qasadm/.login"
                SAPAdmin = qasadm
                InstProfile = "/usr/sap/QAS/SYS/profile/QAS_D02_qasdi"
                InstType = SAPSTARTSRV
                ProcMon = sapstartsrv
                RestartLimit = 5
                )

        requires group Oracle_Database online global soft
        AAS_AWS_IP requires sap_aas_nic
        sap_aas_ip requires AAS_AWS_IP
        sap_aas_res requires sap_aas_sapstartsrv
        sap_aas_sapstartsrv requires sap_aas_ip



        // resource dependency tree
        //
        //      group sap_aas_sg
        //      {
        //      SAPNW sap_aas_res
        //          {
        //          SAPNW sap_aas_sapstartsrv
```

```
      //              {
      //              IP sap_aas_ip
      //                  {
      //                  AWSIP AAS_AWS_IP
      //                      {
      //                      NIC sap_aas_nic
      //                      }
      //                  }
      //              }
      //          }
      //      }


group sap_ascs_sg (
      SystemList = { ip-10-239-2-201 = 0, ip-10-239-3-203 = 1 }
      ClusterList = { sapawsclus = 1, sapaws_onprem = 0 }
      Authority = 1
      ClusterFailOverPolicy = Manual
      PreOnline = 1
      )

      AWSIP ASCS_AWS_IP (
            Critical = 0
            PrivateIP = "10.239.2.70"
            Device = eth0
            AWSBinDir = "/usr/local/bin"
            )

      IP sap_ascs_ip (
            Critical = 0
            Device = eth0
            Address = "10.239.2.70"
            NetMask = "255.255.254.0"
            )

    NIC sap_ascs_nic (
            Critical = 0
            Device = eth0
            )

    SAPNW sap_ascs_res (
            ResLogLevel = TRACE
            EnvFile = "/home/qasadm/.login"
            SAPAdmin = qasadm
            InstProfile = "/usr/sap/QAS/SYS/profile/QAS_ASCS00_qasascs"
            InstType = ENQUEUE
            ProcMon = "en ms"
            )

    SAPNW sap_ascs_sapstartsrv (
            Critical = 0
            ResLogLevel = TRACE
            EnvFile = "/home/qasadm/.login"
            SAPAdmin = qasadm
            InstProfile = "/usr/sap/QAS/SYS/profile/QAS_ASCS00_qasascs"
            InstType = SAPSTARTSRV
            ProcMon = sapstartsrv
            ToleranceLimit = 5
            RestartLimit = 5
            )

    requires group cvm online local firm
```

```
        sap_ascs_ip requires ASCS_AWS_IP

        sap_ascs_ip requires sap_ascs_nic

        sap_ascs_res requires sap_ascs_sapstartsrv

        sap_ascs_sapstartsrv requires sap_ascs_ip



        // resource dependency tree

        //

        //    group sap_ascs_sg

        //    {

        //    SAPNW sap_ascs_res

        //        {

        //        SAPNW sap_ascs_sapstartsrv

        //            {

        //            IP sap_ascs_ip

        //                {

        //                AWSIP ASCS_AWS_IP

        //                NIC sap_ascs_nic

        //                }

        //            }

        //        }

        //    }



group sap_ers_sg (

        SystemList = { ip-10-239-3-203 = 0, ip-10-239-2-201 = 1 }

        AutoStart = 0

        ClusterList = { sapawsclus = 1, sapaws_onprem = 0 }

        ClusterFailOverPolicy = Manual

        PreOnline = 1

        AutoRestart = 0
        )

        AWSIP ERS_AWS_IP (
                Critical = 0
                PrivateIP = "10.239.2.71"
                Device = eth0
                AWSBinDir = "/usr/local/bin"
                )

        IP sap_ers_ip (
                Critical = 0
                Device = eth0
                Address = "10.239.2.71"
                NetMask = "255.255.254.0"
                )

        NIC sap_ers_nic (
                Device = eth0
                )

        SAPNW sap_ers_res (
                ResLogLevel = TRACE
                EnvFile = "/home/qasadm/.login"
                SAPAdmin = qasadm
                InstProfile = "/usr/sap/QAS/SYS/profile/QAS_ERS10_qasers"
                InstType = ENQREP
                ProcMon = er
                EnqSrvResName = sap_ascs_res
                )

        SAPNW sap_ers_sapstartsrv (
      EnvFile = "/home/qasadm/.login"
      SAPAdmin = qasadm
      InstProfile = "/usr/sap/QAS/SYS/profile/QAS_ERS10_qasers"
      InstType = SAPSTARTSRV
      ProcMon = sapstartsrv
      )

requires group cvm online local firm
sap_ers_ip requires ERS_AWS_IP
sap_ers_ip requires sap_ers_nic
sap_ers_res requires sap_ers_sapstartsrv
sap_ers_sapstartsrv requires sap_ers_ip



// resource dependency tree
//
//    group sap_ers_sg
//    {
//    SAPNW sap_ers_res
//        {
//        SAPNW sap_ers_sapstartsrv
//            {
//            IP sap_ers_ip
//                {
//                AWSIP ERS_AWS_IP
//                NIC sap_ers_nic
//                }
//            }
//        }
//    }
```

```
group sap_pas_sg (
        SystemList = { ip-10-239-2-104 = 0, ip-10-239-3-251 = 1 }
        ClusterList = { sapawsclus = 1, sapaws_onprem = 0 }
        Authority = 1
        ClusterFailOverPolicy = Manual
        )

        AWSIP PAS_AWS_IP (
                Critical = 0
                PrivateIP = "10.239.2.73"
                Device = eth0
                AWSBinDir = "/usr/local/bin"
                )

        IP sap_pas_ip (
                Critical = 0
                Device = eth0
                Address = "10.239.2.73"
                NetMask = "255.255.254.0"
                )

        NIC sap_pas_nic (
                Critical = 0
                Device = eth0
                )

        SAPNW sap_pas_res (
                Critical = 0
                ResLogLevel = TRACE
                EnvFile = "/home/qasadm/.login"
```

```
      SAPAdmin = qasadm
      InstProfile = "/usr/sap/QAS/SYS/profile/QAS_DVEBMGS01_qaspas"
      ProcMon = dw
      )

SAPNW sap_pas_sapstartsrv (
      Critical = 0
      ResLogLevel = TRACE
      EnvFile = "/home/qasadm/.login"
      SAPAdmin = qasadm
      InstProfile = "/usr/sap/QAS/SYS/profile/QAS_DVEBMGS01_qaspas"
      InstType = SAPSTARTSRV
      ProcMon = sapstartsrv
      ToleranceLimit = 3
      RestartLimit = 5
      )

requires group Oracle_Database online global soft
sap_pas_ip requires PAS_AWS_IP
sap_pas_ip requires sap_pas_nic
sap_pas_res requires sap_pas_sapstartsrv
sap_pas_sapstartsrv requires sap_pas_ip



// resource dependency tree
//
//    group sap_pas_sg
//    {
//    SAPNW sap_pas_res
//        {
//        SAPNW sap_pas_sapstartsrv
//            {
//            IP sap_pas_ip
//                {
//                AWSIP PAS_AWS_IP
//                NIC sap_pas_nic
//                }
//            }
//        }
//    }
```

References

SAP notes for SAP on AWS:

- 1588667: Overview of related SAP Notes and Web-Links
- 1656099: Supported SAP, DB/OS and AWS EC2 products
- 1656250: Support prerequisites

SAP on Linux:

- 171356: SAP Software on Linux: Essential information
- SAP SCN Page on Linux
- SAP on Linux in general (FAQ)
- Red Hat Enterprise Linux Knowledge Repository
- SUSE Linux Enterprise Server 11 documentation

Amazon documentation:

- https://aws.amazon.com/sap/
- https://d0.awsstatic.com/enterprise-marketing/SAP/sap-on-aws-high-availability-guide.pdf
- http://awsmedia.s3.amazonaws.com/sap_on_aws_implementation_guide_v2.1.pdf

Veritas documentation:

- https://www.veritas.com/product/software-defined-storage/infoscale-enterprise
- https://www.veritas.com/content/trial/en/us/veritas-infoscale-solutions.html
- https://sort.veritas.com/documents/doc_details/vie/7.3/Linux/ProductGuides/
- https://sort.veritas.com/agents/detail/6860
- https://sort.veritas.com/agents/detail/7018

About Veritas Technologies LLC

Veritas Technologies LLC enables organizations to harness the power of their information, with solutions designed to serve the world’s largest and most complex heterogeneous environments. Veritas works with 86 percent of Fortune 500 companies today, improving data availability and revealing insights to drive competitive advantage.

For specific country offices and contact numbers, please visit our website.

Veritas World Headquarters
500 East Middlefield Road
Mountain View, CA 94043
+1 (650) 933 1000
www.veritas.com