Hewlett Packard
Enterprise

Server Automation

Failover, Load Balancing and High Availability

Software version: 10.22

Document release date: December 2015

Software release date: December 2015

Contents

This paper provides an overview of failover, high availability and load balancing architectures in HPE Server Automation.

Introduction

This document discusses failover, high availability and load balancing for Server Automation Ultimate Edition.

Note: There are two editions of Server Automation, Server Automation Ultimate and Server Automation Standard (Virtual Appliance). This document applies to Server Automation Ultimate.

Server Automation is data center automation software that centralizes and streamlines many data center functions and automates critical areas of your data center's server management, including:

- Server Discovery
- Operating System Provisioning
- Operating System Patching
- Software Provisioning
- Audit and Compliance
- Application Configuration
- Application Deployment
- Software Compliance

Additional details on these functions can be found in the Server Automation Overview and Architecture Guide. You can use the SA Documentation Library to find the latest version of the guides for your version of SA on the HPE Software Support Online (HPE Passport required).

Designing Server Automation architectures for high availability

Note: This paper does not address backup/restore, monitoring or disaster recovery. All three of these need to be addressed in order to create a fully resilient solution.

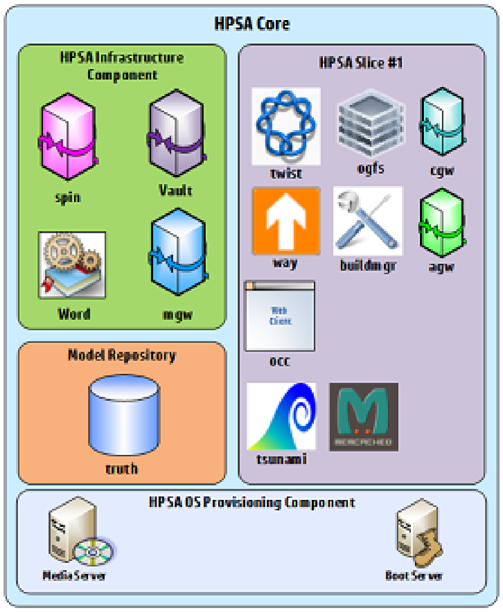


Figure 1: SA core components

The architectures described in the following sections presume that SA is installed using the standard SA Core Configurations documented in the Server Automation Installation Guide.

Server Automation can be deployed in multiple configurations which provide different degrees of resiliency. This paper does not address database HA beyond noting that Oracle RAC can be used with SA.

Server Automation components

At the most basic, an SA Core consists of the following components, as shown in Figure 1: SA Core Components

- Infrastructure
- Slice#1
- Model Repository (either local or remote)
- OS Provisioning components

An SA Core can be scaled internally by adding additional Slice components (Figure 1: SA Core Components), and externally by adding additional SA Cores and Satellites (Figure 3: SA Satellite).

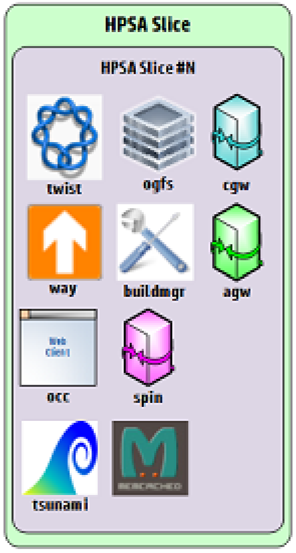


Figure 2: SA slice #N components

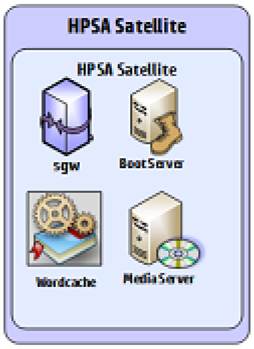


Figure 3: SA satellite components

A basic SA deployment consisting of a single SA Core is shown in Figure 4: SA Basic Single Core Installation.

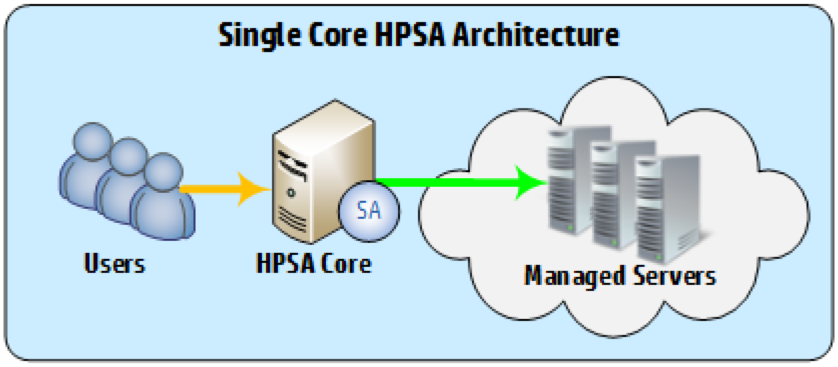


Figure 4: SA basic single core installation

In this design, the SA core is a single point of failure – both user connections and server management will fail if the SA core fails, as shown in Figure 5: SA Single Core Installation points of failure.

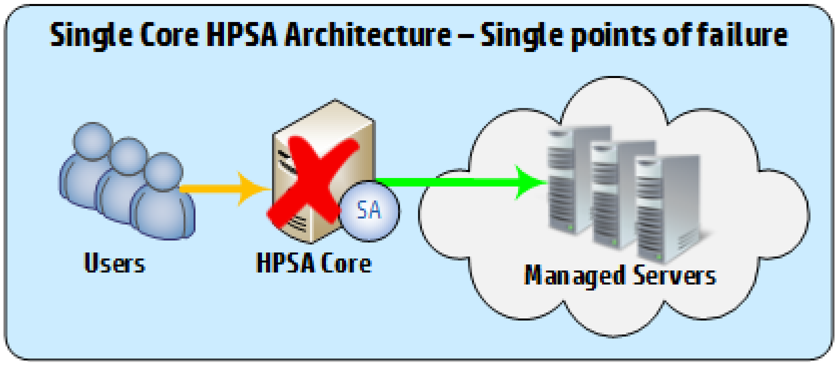


Figure 5: SA single core installation points of failure

In-core load balancing

This information is included for troubleshooting purposes; modifying the load balancing configuration is **NOT** recommended, and typically results in unexpected behaviors and failures.

An SA Core consists of an Infrastructure, Slice#1, Model Repository (either local or remote), OS Provisioning component and one or more additional Slice components.

In a multi-slice configuration, the SA infrastructure component automatically load balances various services across the slices in the core, allowing additional slices to fail transparently[1].

Note: If the Infrastructure component fails, all components in the core MUST be shut down until the Infrastructure is recovered.

The following SA Slice components are load balanced between active slices:

SA Slice Component	Load Balancing Mode	Description
Build Manager (buildmgr)	ORDERED	Connect to the first slice listed
Command Center (occ)	TLS_LC	Use a sticky TLS session to the slice with the least number of connections
Global File System (hub)	STICKY	Use a sticky connection to a randomly selected slice
Secondary Data Access Engine (secondary spin)	STICKY	Use a sticky connection to a randomly selected slice
Software Repository (word)	STICKY	Use a sticky connection to a randomly selected slice
Web Services Data Access Engine (twist)	STICKY	Use a sticky connection to a randomly selected slice
Command Engine (way)	STICKY	Use a sticky connection to a randomly selected slice

If the following Slice components are enabled and fail, the default Software Repository functionality will be used.

SA Slice Component
Software Repository Accelerator (tsunami)
Memcache

[1] Obviously a slice failure may affect performance, depending on task load.

SA Multimaster Mesh – simple core failover

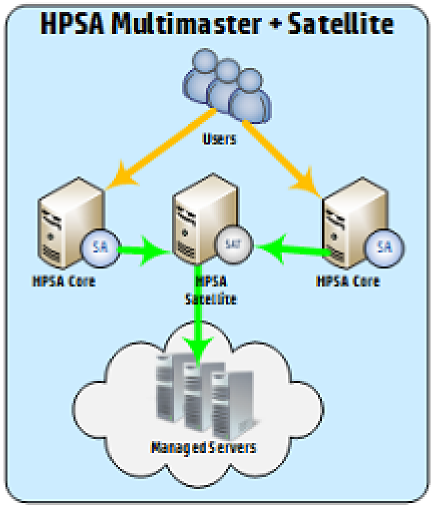


Figure 6: SA Multimaster + satellite architecture

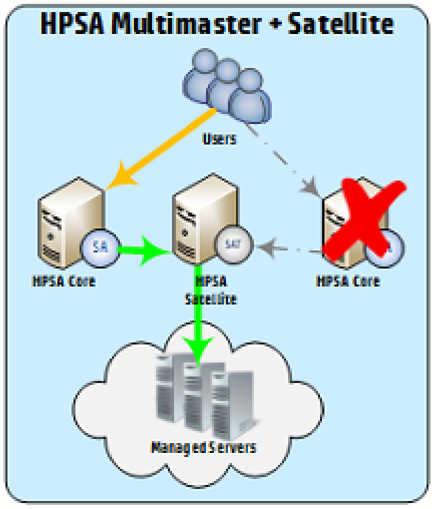
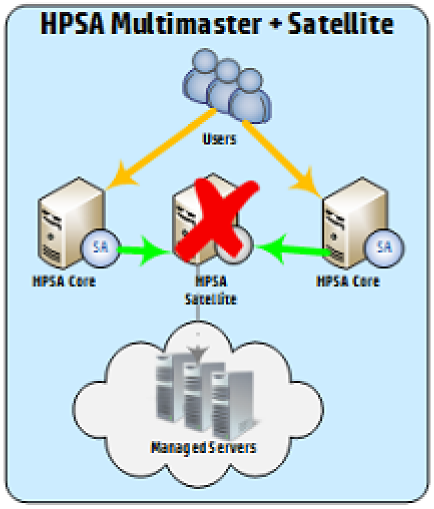
The first HA Architecture that we consider here is a basic Multimaster Mesh, with two SA Cores, and a single satellite (Figure 6: SA Multimaster + Satellite Architecture).

The primary advantage of this configuration is the ability to continue to manage servers and serve users in the event of a single core failure.

A Satellite has been added in this configuration, as servers which are managed directly by a core will become unreachable if that core fails; satellites can be configured to fail between cores. This ensures that server management can continue if one of the cores fails (Figure 4: SA Multimaster + Satellite - Core Failure).

If the Satellite fails, users will still be able to connect to the SA cores, but server management will not be available (Figure 8: SA Multimaster + Satellite - Satellite Failure).

In this configuration, users can connect to either SA core, but must know the address of the core that they wish to connect to in the event of a failure.

SA Multimaster + Satellite Failure Conditions	
 HPSA Multimaster + Satellite Users SA HPSA Core SAT HPSA Satellite HPSA Core Managed Servers *Figure 7: SA Multimaster + satellite - core failure*	 HPSA Multimaster + Satellite Users SA HPSA Core HPSA Satellite SA HPSA Core Managed Servers *Figure 8: SA Multimaster + satellite - satellite failure*

Simple core failover configuration

This section describes the components and additional configuration required to implement this solution.

1. Install the First (Primary) Core with a Secondary Core (Multimaster Mesh) as described in the Server Automation Installation Guide.
2. Install the SA Satellite as described in the Server Automation Installation Guide, being sure to specify the same name for both the Satellite Facility and Satellite Realm (must be different from the Core Facility names), eg: SA10SAT

   The Satellite Gateway name uniquely identifies the satellite, and is typically something similar to <Satellite Facility Name><number>, eg: SATFACILITY01

3. Perform the remaining configuration tasks and finalize the satellite installation.
4. Edit the gateway properties file and modify section 3 as follows:

```
# 3) This Gateway should have at least one outbound tunnel.
#    Please uncomment one the lines below and replace the IP
#    and port (i.e., 10.0.0.10:2001) with the IP and TunnelDst
#    port for your Core-side Gateway component.
#    ip:port:cost:bw   (bw in kbits/sec)

opswgw.TunnelSrc=<core1 ip>:2001:100:0:/var/opt/opsware/crypto/opswgw-SA1010SAT01/opswgw.pem
opswgw.TunnelSrc=<core2 ip>:2001:200:0:/var/opt/opsware/crypto/opswgw-SA1010SAT01/opswgw.pem
#opswgw.TunnelSrc=10.0.0.11:2001:200:0:/var/opt/opsware/crypto/opswgw-SA1010SAT01/opswgw.pem
```

   These two lines tell the satellite to create encrypted tunnels to the management gateways on Core1 and Core2, with the ‘100’ and ‘200’ indicating which tunnel will be preferred (the lower number takes priority).

   Note: Each tunnel MUST have a different priority. Setting the same priority will result in unpredictable failures.

   In this case, the configuration means that the satellite will send traffic to Core1 unless Core1 is down. If Core1 is down, the satellite will select the tunnel with the next lowest priority, which would be Core2 in this example.

   Note: Gateway customizations (i.e. adding a new tunnel to Core 2) should be moved from the opswgw.properties file to the opswgw.custom file (/etc/opt/opsware/opswgw-<gateway_name>/opswgw.custom) to preserve those customizations during a Server Automation upgrade.

5. Perform the remaining configuration tasks.

SA Multimaster Mesh – core and satellite failover

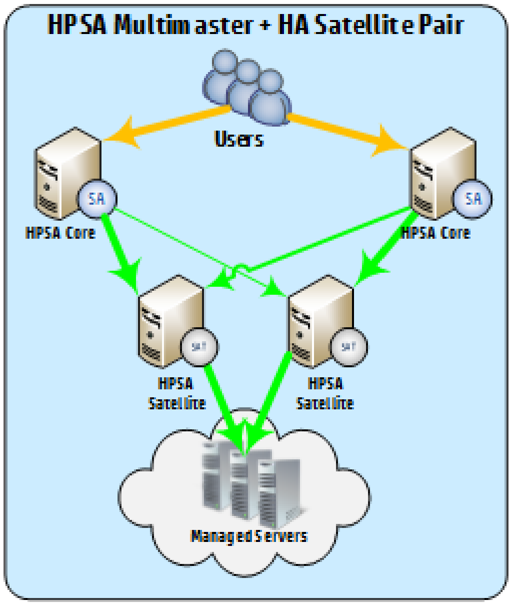
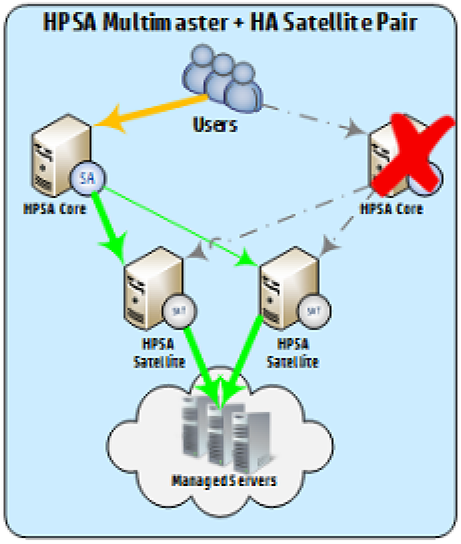
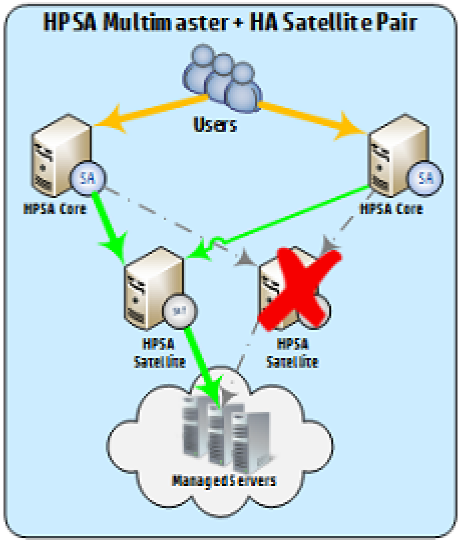


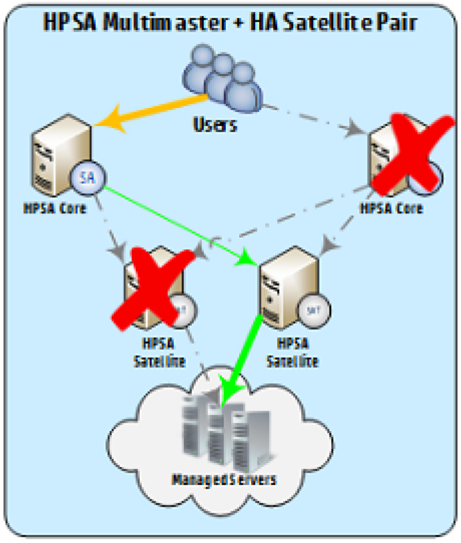
Figure 9: SA Multimaster + HA satellite pair

This design improves on the initial SA Multimaster Mesh design through the introduction of an HA Satellite pair in place of the single satellite in SA Multimaster Mesh – Core and Satellite Failover.

This design ensures that services can continue transparently in the event of a core, satellite, or core and satellite failure.

In this configuration, users can connect to either SA core, but must know the address of the core that they wish to connect to in the event of a failure.

SA Multimaster Mesh – Core or Satellite Failure	
 HP SA Multimaster + HA Satellite Pair Users SA HPSA Core HPSA Core HPSA Satellite HPSA Satellite Managed Servers *mFigure 10: SA Multimaster Mesh - core and satellite failover - core failure*	 HP SA Multimaster + HA Satellite Pair Users SA HPSA Core SA HPSA Core HPSA Satellite HPSA Satellite Managed Servers *Figure 11: SA Multimaster Mesh - core and satellite failover - satellite failure*

SA Multimaster Mesh – Core or Satellite Failure
 HP SA Multimaster + HA Satellite Pair Users SA HPSA Core HPSA Core HPSA Satellite HPSA Satellite Managed Servers *Figure 12: SA Multimaster Mesh - core and satellite failure*

Core and satellite failover configuration

This section describes the components and additional configuration required to implement this solution.

1. Install the First (Primary) Core with a Secondary Core (Multimaster Mesh) as described in the Server Automation Installation Guide.

2. Install the first SA Satellite as described in the Server Automation Installation Guide, being sure to specify the same name for both the Satellite Facility and Satellite Realm (must be different from the Core Facility names), eg: SA10SAT
   The Satellite Gateway name uniquely identifies the satellite, and is typically something similar to <Satellite Facility Name><number>, eg: SATFACILITY01
3. After the satellite installation is complete, edit the gateway properties file and modify section 3 as follows:

```
# 3) This Gateway should have at least one outbound tunnel.
#    Please uncomment one the lines below and replace the IP
#    and port (i.e., 10.0.0.10:2001) with the IP and TunnelDst
#    port for your Core-side Gateway component.
#    ip:port:cost:bw   (bw in kbits/sec)

opswgw.TunnelSrc=<core1 ip>:2001:100:0:/var/opt/opsware/crypto/opswgw-SA1010SAT01/opswgw.pem
opswgw.TunnelSrc=<core2 ip>:2001:200:0:/var/opt/opsware/crypto/opswgw-SA1010SAT01/opswgw.pem
#opswgw.TunnelSrc=10.0.0.11:2001:200:0:/var/opt/opsware/crypto/opswgw-SA1010SAT01/opswgw.pem
```

These two lines tell the satellite to create encrypted tunnels to the management gateways on Core1 and Core2, with the ‘100’ and ‘200’ indicating which tunnel will be preferred (the lower number takes priority).

Note: Each tunnel MUST have a different priority. Setting the same priority will result in unpredictable failures.

In this case, the configuration means that the satellite will send traffic to Core1 unless Core1 is down. If Core1 is down, the satellite will select the tunnel with the next lowest priority, which would be Core2 in this example.

Note: Gateway customizations (i.e. adding a new tunnel to Core 2) should be moved from the opswgw.properties file to the opswgw.custom file (/etc/opt/opsware/opswgw-<gateway_name>/opswgw.custom) to preserve those customizations during a Server Automation upgrade.

4. Install the second SA Satellite as described in the Server Automation Installation Guide, being sure to specify the same name for both the Satellite Facility and Satellite Realm (must be different from the Core Facility names), eg: SA10SAT. The Satellite Facility and Realm name MUST be the same for both satellites in an HA pair.

   The Satellite Gateway name uniquely identifies the satellite, and is typically something similar to <Satellite Facility Name><number>, eg: SATFACILITY02

5. After the satellite installation is complete, edit the gateway properties file and modify section 3 as follows:

```
# 3) This Gateway should have at least one outbound tunnel.
#    Please uncomment one the lines below and replace the IP
#    and port (i.e., 10.0.0.10:2001) with the IP and TunnelDst
#    port for your Core-side Gateway component.
#    ip:port:cost:bw   (bw in kbits/sec)

opswgw.TunnelSrc=<core1 ip>:2001:100:0:/var/opt/opsware/crypto/opswgw-SA1010SAT01/opswgw.pem
opswgw.TunnelSrc=<core2 ip>:2001:200:0:/var/opt/opsware/crypto/opswgw-SA1010SAT01/opswgw.pem
#opswgw.TunnelSrc=10.0.0.11:2001:200:0:/var/opt/opsware/crypto/opswgw-SA1010SAT01/opswgw.pem
```

These two lines tell the satellite to create encrypted tunnels to the management gateways on Core1 and Core2, with the ‘100’ and ‘200’ indicating which tunnel will be preferred (the lower number takes priority).

Note: Each tunnel MUST have a different priority. Setting the same priority will result in unpredictable failures.

Note: Tunnel priority MUST be set to the same values on both satellites.

In this case, the configuration means that the satellite will send traffic to Core1 unless Core1 is down. If Core1 is down, the satellite will select the tunnel with the next lowest priority, which would be Core2 in this example.

Note: Gateway customizations (i.e. adding a new tunnel to Core 2) should be moved from the opswgw.properties file to the opswgw.custom file (/etc/opt/opsware/opswgw-<gateway_name>/opswgw.custom) to preserve those customizations during a Server Automation upgrade.

6. Perform the remaining configuration tasks.

SA Multimaster Mesh – core, satellite and end-user access failover

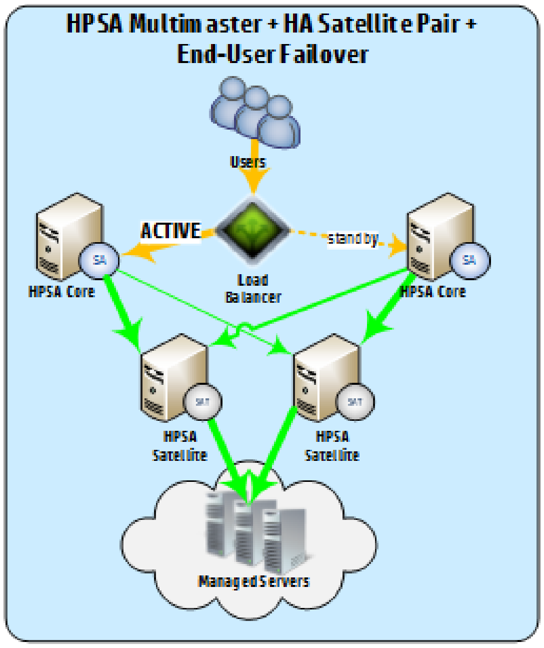


Figure 13: SA Multimaster Mesh - core, satellite and end-user access failover

This design builds on the previous SA Multimaster Mesh – Core and Satellite Failover design to include end-user access failover.

Both a single connection point and transparent end-user failover are provided by the use of an external load balancer.

The load balancer MUST be configured as follows, to avoid issues with SA internal replication and load balancing:

- Load balancer MUST be configured to point the SA core Infrastructure server[2] in an active/standby configuration[3].
- Sticky SSL sessions MUST be configured

HA summary

The following table outlines the HA capabilities available in a single SA Core or SA Multimaster Mesh configuration with the specified SA Components.

SA Component	SA Core	SA Multimaster Mesh
SA Core	n/a	Core Failover
+additional slice component bundle instances	Load Balance*	Core Failover Load Balance*

[2] The SA application provides load balancing capabilities across slices within a given SA Core. This load balancing takes place automatically, and is controlled by the Infrastructure component; configuring an external load balancer to load balance across each slice in an SA core will have unpredictable effects, at best.
Additionally, if the Infrastructure component in a core fails, the entire core MUST be shut down until the infrastructure (and any other affected components) can be recovered.

[3] Briefly – it is possible to have the load balancer route to either the primary or secondary infrastructure based on the source network of the client (though DNS and multiple VIPs with different route configurations on the LB), but the client MUST NEVER route to both the primary and secondary concurrently.

+satellites		Satellite Failover between cores
+satellite HA pair(s)	Agent Failover†	Core Failover Satellite Failover between cores Satellite Failover Agent Failover†
+additional slice component bundle instances and satellites	Load Balance*	Core Failover Satellite Failover between cores Load Balance*
+additional slice component bundle instances and satellite HA pair(s)	Agent Failover† Load Balance*	Core Failover Satellite Failover between cores Agent Failover† Load Balance*

*for certain components

†for agents managed via satellite HPE pair(s)

SA component failure impact

	SA Component	Failure Result (Core)	Failure Result (Mesh)
Model Repository (truth)		Core failure	Core failure; Mesh continues
Infrastructure	Primary Data Access Engine (spin)	Core failure	Core failure; Mesh continues
	Management Gateway (mgw)	Core failure	Core failure; Mesh continues
	Model Repository Multimaster Component (vault)	Core failure	Core failure; Mesh continues
	Software Repository Store (word)	Core failure	Core failure; Mesh continues
OS Prov	Media Server	OS Provisioning failure	OS Provisioning failure
	Boot Server	OS Provisioning Failure	OS Provisioning Failure
Slice #1	Core Gateway / Agent Gateway (cgw / agw)	Core failure	Core failure; Mesh continues
	Command Center (occ)	User Access failure	Core failure; Mesh continues
	Global File System (ogfs)	Core failure	Core failure; Mesh continues
	Web Services Data Access Engine (twist)	Core failure	Core failure; Mesh continues
	Build Manager (buildmgr)	OS Provisioning failure	Core failure; Mesh continues
	Command Engine (way)	Job failure / Core failure	Core failure; Mesh continues
	Software Repository Accelerator (tsunami)	n/a	n/a
	Memcache	n/a	n/a
Slice #x	Core Gateway / Agent Gateway (cgw / agw)	Slice failure	n/a
	Command Center (occ)	Slice failure	n/a
	Global File System (ogfs)	Slice failure	n/a
	Web Services Data Access Engine (twist)	Slice failure	n/a
	Secondary Data Access Engine (spin)	Slice failure	n/a

	Build Manager (buildmgr)	OS Provisioning failure	n/a
	Command Engine (way)	Slice failure	n/a
	Software Repository Accelerator (tsunami)	n/a	n/a
	Memcache	n/a	n/a

Sample F5 load balancer configuration

```
lb.example.com:


wideip {
   name         "lb.example.com"
   pool_lbmode  rr
   partition "Common"
   pool         "lb.example.com_443"
}
pool {
   name           "lb.example.com_443"
   ttl            30
   monitor all "https"
   preferred      ratio
   alternate      ratio
   partition "Common"

   member         10.100.1.10:443   ratio 100
   member         10.100.2.10443   ratio 0
```

Glossary

Term	Description
SA Core	An SA Core is a set of Core Components that work together to allow you to discover servers on your network, add those servers to a Managed Server Pool, and then provision, configure, patch, monitor, audit, and maintain those servers from a centralized SA Client interface. The SA Client provides a single interface to all the information and management capabilities of SA. The servers that the Core Components are installed on are called Core Servers. Core Components, even if distributed to multiple hosts are still considered part of a single SA Core.
SA Core Components	SA Core Components are separated into bundles which must be installed as a unit. Those bundles are: – Model Repository (one per core) – Infrastructure Components (one per core) – OS Provisioning Components (typically one per core) – Slice Components (minimum one per core)
Model Repository (truth)	The Model Repository requires an Oracle database. All SA components work from or update a data model maintained in the Model Repository. The Model Repository stores the following information:

	– An inventory of all servers under SA management – An inventory of the hardware associated with these servers, including memory, CPUs, storage capacity, and so on. – Information about managed server configuration. – An inventory of the operating systems, system software, and applications installed on managed servers. – An inventory of OS Provisioning operating system installation media (the media itself is stored in the OS Provisioning Media Server). – An inventory of software available for installation and the software policies that control how the software is configured and installed. The software installation media itself is stored in the Software Repository. – Authentication and security information.
Infrastructure Components	The Infrastructure Component Bundle contains the following components: – Primary Data Access Engine (spin) – Management Gateway (mgw) – Model Repository Multimaster Component (vault) – Software Repository Store (word)
OS Provisioning Components	The OS Provisioning Component Bundle contains the following components: – Media Server – Boot Server
Slice Components	The Slice Component Bundle contains the following components: – Core Gateway / Agent Gateway (cgw / agw) – Command Center (occ) – Global File System (ogfs) – Web Services Data Access Engine (twist) – Secondary Data Access Engine (spin) – Build Manager (buildmgr) – Command Engine (way) – Software Repository (word) – Software Repository Accelerator (tsunami) – Memcache
Slice Load Balancing Modes	– STICKY: Send the connection to a working target based on a priority list randomized by a hash of the source IP and source Realm – LC: Send connection to a working target with the least number of connections. – RR: Send connection to the next working target in a round-robin fashion. – ORDERED: Send connection in using the precedence order as supplied to the rule. – TLS_STICKY: Use an SSLv3/TLSv1.0 session ID to send the connection back to the previous target based on a session ID cache. If the target is in error, or the session ID is missing from the cache, fall back to STICKY mode to make a new selection. – TLS_LC: Similar to TLS_STICKY mode, but falls back to LC mode (least connections). – TLS_RR: Similar to TLS_STICKY mode, but falls back to RR mode (round-robin).
SA Satellite	A Satellite installation can be a solution for remote sites that do not have a large enough number of potentially Managed Servers to justify a full SA Core installation. A Satellite installation allows you to install only the minimum necessary Core Components on the Satellite host which then accesses the Primary Core's database and other services through an SA Gateway connection. A Satellite installation typically consists of, at minimum, a Satellite Gateway and a Software Repository Cache and still allows you to fully manage servers at a remote facility. The Software Repository Cache contains local copies of software packages to be installed on Managed Servers in the Satellite while the Satellite Gateway handles communication with the Primary Core. You can optionally install the OS Provisioning Boot Server and Media Server on the Satellite host to support remote OS Provisioning.
SA Satellite Components	The basic SA Satellite Component Bundle contains the following components: – Satellite Gateway (sgw) – Software Repository Cache (wordcache) An OS Provisioning Satellite adds the following components: – Media Server – Boot Server

Managed Server	A Server with the SA Server Agent installed which is managed by SA.
SA Server Agent	An SA Server Agent is intelligent software that is installed on all servers that you want SA to manage. After an agent is installed on an agentless server, the agent register the server with the SA Core which then adds that server to its pool of Managed Servers. The SA Agent also receives user initiated commands from the Core and takes the appropriate action on the server it is installed on, such as software installation and removal, software and hardware configuration, server status reporting, auditing, and so on. During agent registration, SA assigns each server a unique ID (the Machine ID (MID)) and stores this ID in the Model Repository.
Facility	A Facility is a construct that typically represents a collection of servers that a single SA core manages through the data about the managed environment stored in its Model Repository. A facility typically represents a specific geographical location, such as Sunnyvale, San Francisco, or New York, or, commonly, a specific data center. A new Facility is automatically created for each SA Core, and each SA Core MUST belong to a separate Facility. A Facility contains one or more Realms. A Facility is a permissions boundary within SA, that is, a user's permissions in one Facility do not carry over to another. Every Managed Server is assigned to a single facility. When a device initially registers with the SA Core, it is assigned to the facility associated with the gateway through which it is registering. A managed server can only be a member of one Facility at a time.
Realms	Realms are a SA construct that allow SA to manage servers on different networks in the same Facility without fear of IP address conflicts. A realm is a unique identifier, appended to the IP address of a device in a Facility's network that allows SA Gateways to uniquely identify devices on different networks in a Multimaster Mesh that may have conflicting IP addresses. A Realm is a logical entity that defines an IP namespace within which all Managed Server IP addresses must be unique. However, servers that are assigned to different Realms can have duplicate IP addresses and still be uniquely identified within SA by their Realm membership. Realms are interconnected by gateways in what can be described as a gateway mesh — a single interconnected network of SA Gateways. When you create and name a new Facility during installation, a default Realm is also created with the same name as the Facility. For example, when you create the Facility, Datacenter, the installation also creates a Realm named Datacenter. Subsequent Realms in that facility could be named Datacenter001, Datacenter002, and so on. Managed servers in each realm are uniquely identified by the combination of the Realm name and the IP address.

Send documentation feedback

If you have comments about this document, you can send them to hpe_sa_docs@hpe.com.

Legal notices

Warranty

The only warranties for Hewlett Packard Enterprise products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Hewlett Packard Enterprise shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Restricted rights legend

Confidential computer software. Valid license from Hewlett Packard Enterprise required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor’s standard commercial license.

Copyright notice

Trademark notices

Adobe® is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

UNIX® is a registered trademark of The Open Group.

RED HAT READY™ Logo and RED HAT CERTIFIED PARTNER™ Logo are trademarks of Red Hat, Inc.

The OpenStack word mark and the Square O Design, together or apart, are trademarks or registered trademarks of OpenStack Foundation in the United States and other countries, and are used with the OpenStack Foundation’s permission.

Documentation updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to the following URL and sign-in or register: https://softwaresupport.hp.com/

Select Manuals from the Dashboard menu to view all available documentation. Use the search and filter functions to find documentation, whitepapers, and other information sources.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your Hewlett Packard Enterprise sales representative for details.

Support

Visit the Hewlett Packard Enterprise Software Support Online web site at https://softwaresupport.hp.com/