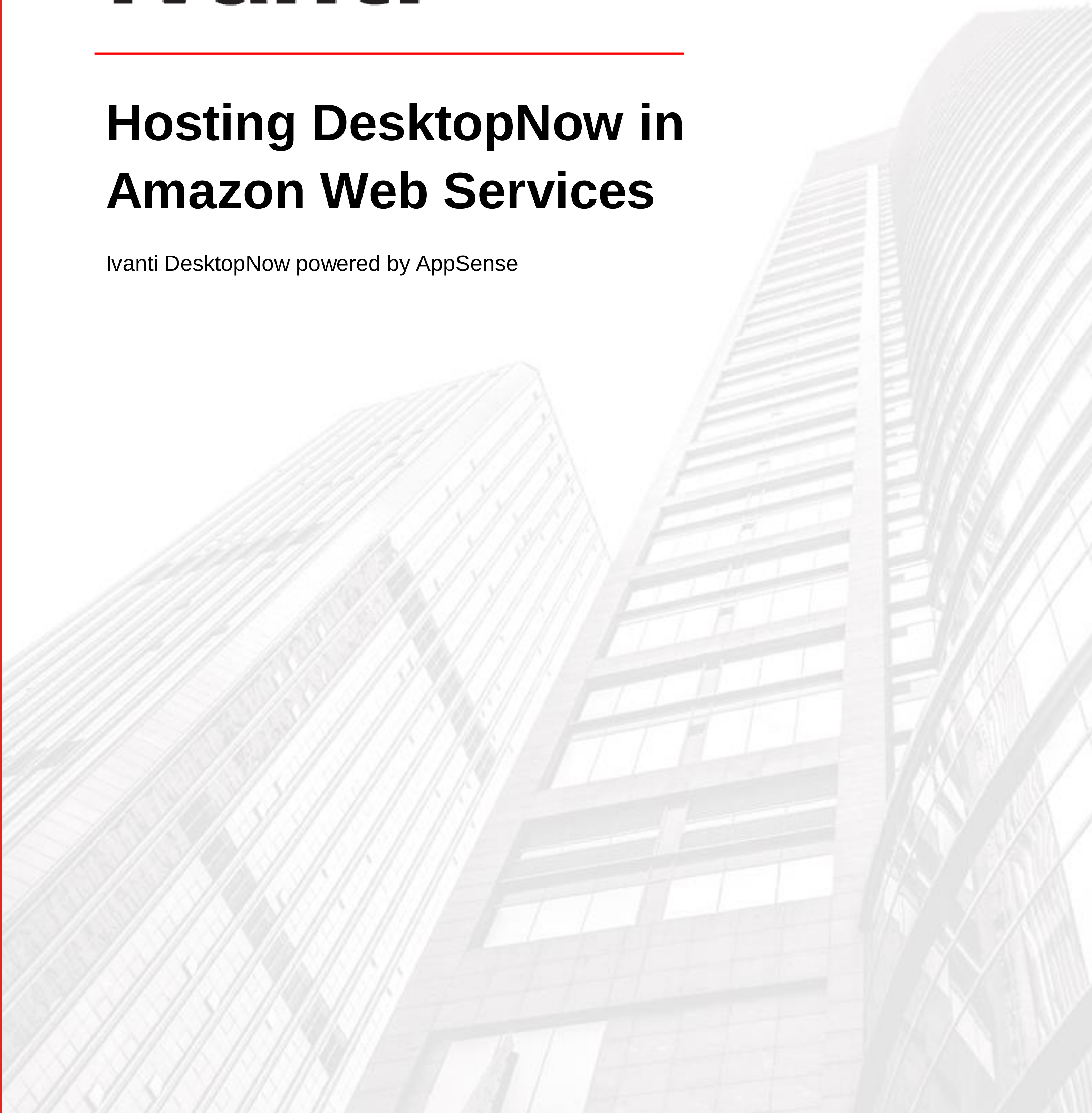
Hosting DesktopNow in Amazon Web Services

Ivanti DesktopNow powered by AppSense



Contents

Purpose of this Document

The purpose of this document is to provide Ivanti customers and Partners with a series of recommendations when working with Ivanti DesktopNow powered by AppSense and the Amazon Web Services computing platform

It should be noted that this document will not include details on the installation or configuration of Ivanti DesktopNow or the Amazon Web Services platform.

Overview

The document serves to provide the reader an overview of configuring Ivanti DesktopNow within the Amazon Web Services (AWS) platform.

Only the Ivanti components applicable to this document are detailed and discussed. For full details of Ivanti DesktopNow, consult the product documentation available at http://www.ivanti.com

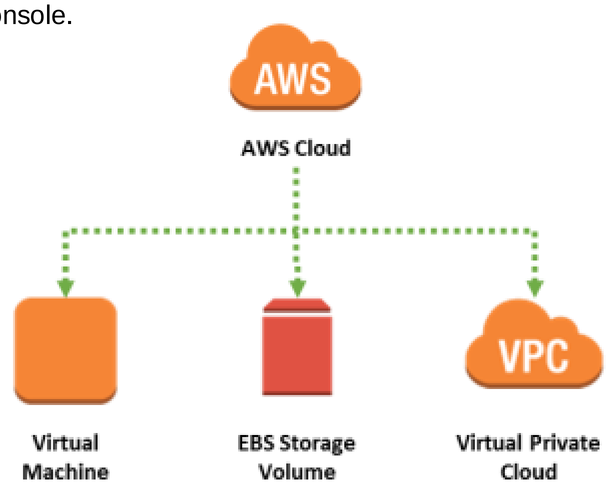
Further details relating to the AWS platform can be found here https://aws.amazon.com/?nc2=h_lg

This document is composed of three sections:

1 Non load balanced Amazon Web Services Environment

Amazon Web Services Configuration

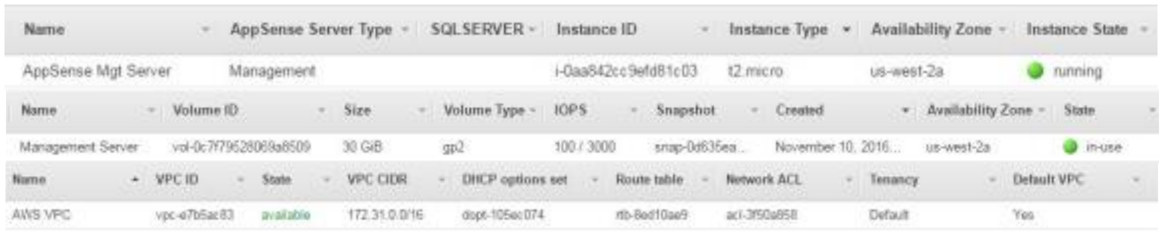
The AWS Management Console allows for the access and administration through a simple and intuitive web-based user interface. For this section of the document, the following resources were created via the AWS Management Console.



Note

With only one Virtual Machine, it is not necessary for a Virtual Private Cloud to be used. If, however, a separate Virtual Machine had been commissioned for the Microsoft SQL Server then the Virtual Private Cloud would have been required for communication between them.

The configuration when viewed from the AWS Management Console can be seen below.



Name	AppSense Server Type	SQLSERVER	Instance ID	Instance Type	Availability Zone	Instance State
AppSense Mgt Server	Management		i-0aa842cc9efd81c03	t2.micro	us-west-2a	● running

Name	Volume ID	Size	Volume Type	IOPS	Snapshot	Created	Availability Zone	State
Management Server	vol-0c7f79528069a8509	30 GiB	gp2	100 / 3000	snap-0d635ea...	November 10, 2016...	us-west-2a	● in-use

Name	VPC ID	State	VPC CIDR	DHCP options set	Route table	Network ACL	Tenancy	Default VPC
AWS VPC	vpc-e7b5ac83	available	172.31.0.0/16	dopt-105ec074	rtb-8ed10ae9	acl-3f50a858	Default	Yes

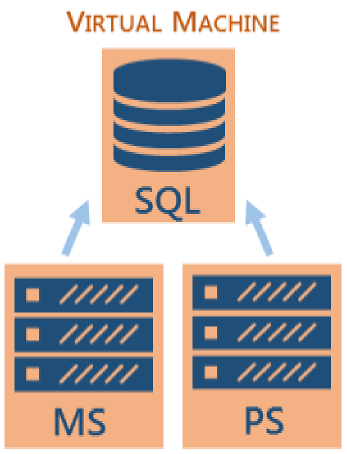
Note

AWS Storage is accessible from anywhere in the world, from any type of application, whether it is running in the cloud, on a desktop or on an on-premises server.

For the purposes of this scenario AWS Storage was used to provide access to installation media such as Ivanti DesktopNow, for example.

Amazon Web Services Virtual Machines

The AWS platform provides a flexible environment allowing for a wide range of computing solutions to be implemented. These machines can be accessed via a Remote Desktop (RDP) session in a similar way to that of an on-premises server.



MS = AppSense Management Server
PS = AppSense Personalization Server

A single Virtual Machine was created and configured as follows:

- Microsoft Windows Server 2012 R2 with the necessary IIS Roles installed
- Microsoft SQL Server 2014 Standard Edition with Service Pack 1
- Ivanti DesktopNow v10
- VPN Client

The use of a single Virtual Machine does not indicate that Microsoft SQL Server and Ivanti DesktopNow must be co-installed. This was merely a decision of simplicity rather than necessity.

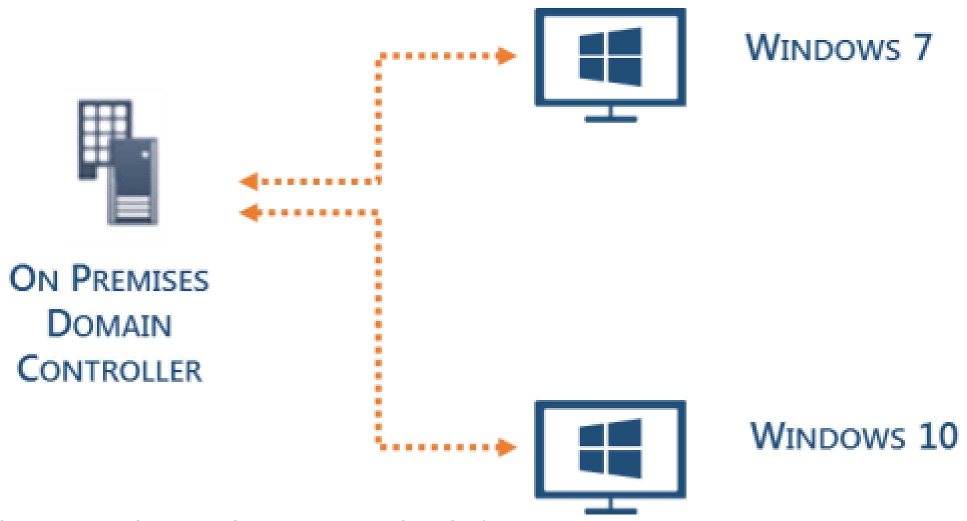
AWS has two options for hosting SQL Server workloads.

- Amazon RDS SQL Database: A SQL database that is native to the cloud, sometimes referred to as a platform as a service (PaaS) database or as a service (DBaaS) that is optimized for software as a service (SaaS) app development.
- SQL Server on AWS Virtual Machines: A SQL Server that is installed and hosted in the cloud on virtual machines, sometimes referred to as an infrastructure as a service (IaaS).

At the time of writing it was not possible to use Ivanti DesktopNow and the Amazon RDS SQL Database method for hosting the Management and Personalization databases.

On-Premises Environment

An on-premises environment was built to prove that physical desktops can be managed from the cloud, for example via an Environment Manager implementation housed within the Amazon Web Services platform. It is not intended to be a representation of a typical Ivanti customer implementation.



The on-premises environment consisted of:

- Microsoft Windows Server 2012 R2 configured as a Domain Controller
- Microsoft Windows 7 Ultimate
- Microsoft Windows 10 version 1607

In addition, a Virtual Private Network was configured to allow the AWS hosted server to access and join the on-premises domain.

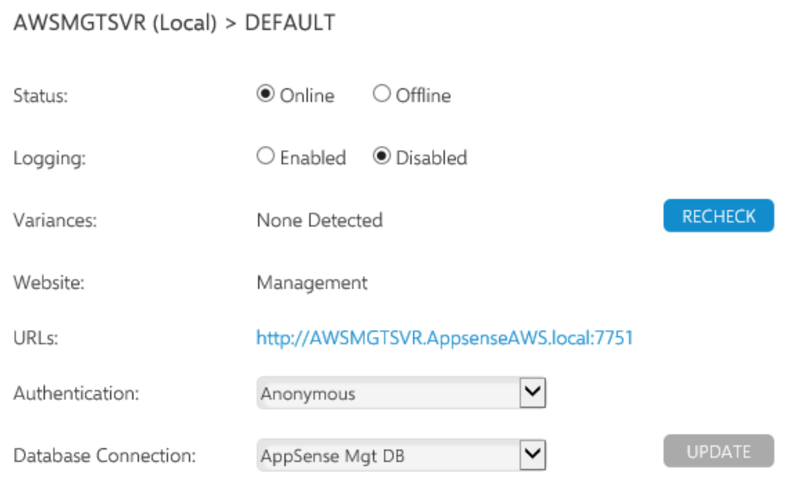
Ivanti DesktopNow Configuration

The AWS hosted Microsoft Windows Server 2012 R2 Virtual Machine was joined to the on- premises domain. DesktopNow v10 was installed using the Suite Installer and the Server Configuration Portal used to create the following databases within the workload of the SQL Server on an AWS Virtual Machine:

- Ivanti_MgtDB
- Ivanti_PersDB

Ivanti Management Server Configuration

The Ivanti Management Server was configured in the following way:

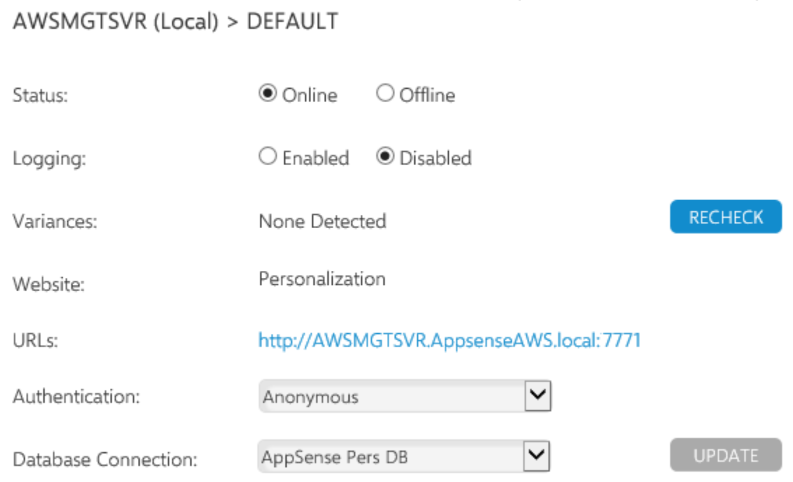


AWSMGTSVR (Local) > DEFAULT

Status:	● Online ○ Offline	
Logging:	○ Enabled ● Disabled	
Variances:	None Detected	RECHECK
Website:	Management	
URLs:	http://AWSMGTSVR.AppsenseAWS.local:7751	
Authentication:	Anonymous	
Database Connection:	AppSense Mgt DB	UPDATE

Ivanti Personalization Server Configuration

The Ivanti Personalization Server was configured in the following way:

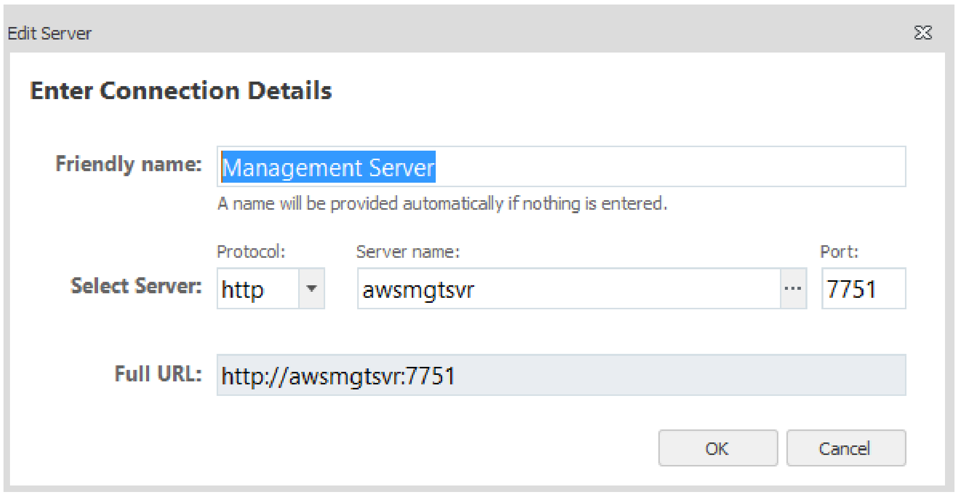


AWSMGTSVR (Local) > DEFAULT

Status:	● Online ○ Offline	
Logging:	○ Enabled ● Disabled	
Variances:	None Detected	RECHECK
Website:	Personalization	
URLs:	http://AWSMGTSVR.AppsenseAWS.local:7771	
Authentication:	Anonymous	
Database Connection:	AppSense Pers DB	UPDATE

Consoles

The Ivanti Management Center and Environment Manager consoles were configured to connect to the respective AWS hosted servers. There was no bespoke configuration required.



Enter Connection Details

Friendly name: Management Server

A name will be provided automatically if nothing is entered.

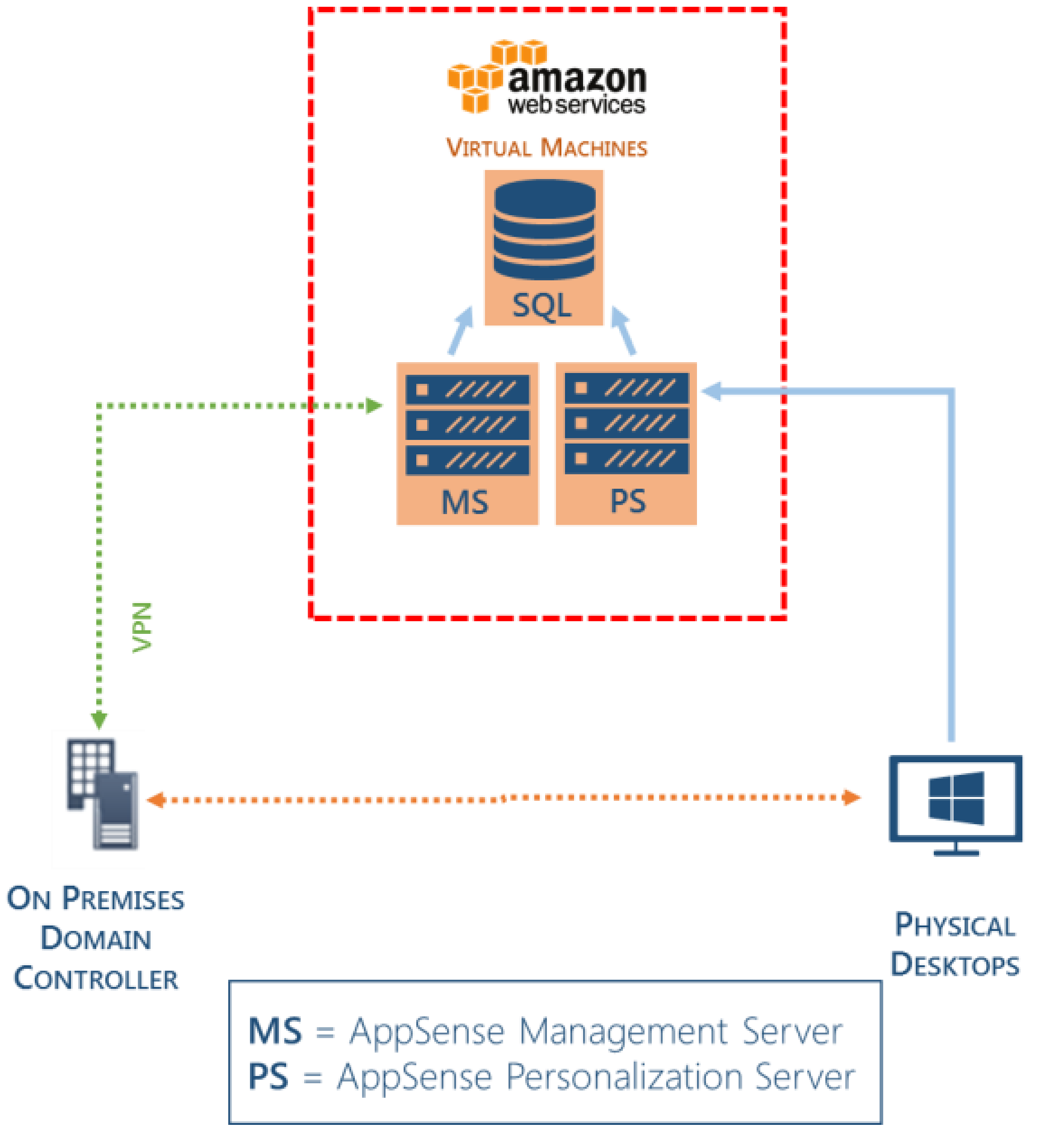
Select Server: Protocol: http | Server name: awsmgtsvr | Port: 7751

Full URL: http://awsmgtsvr:7751

Typical configuration such as Membership Rules and Access Credentials were configured and the agents for the Ivanti Management Center, Application Manager, Environment Manager and Performance Manager deployed. Again, no bespoke configuration was required.

Overall Configuration

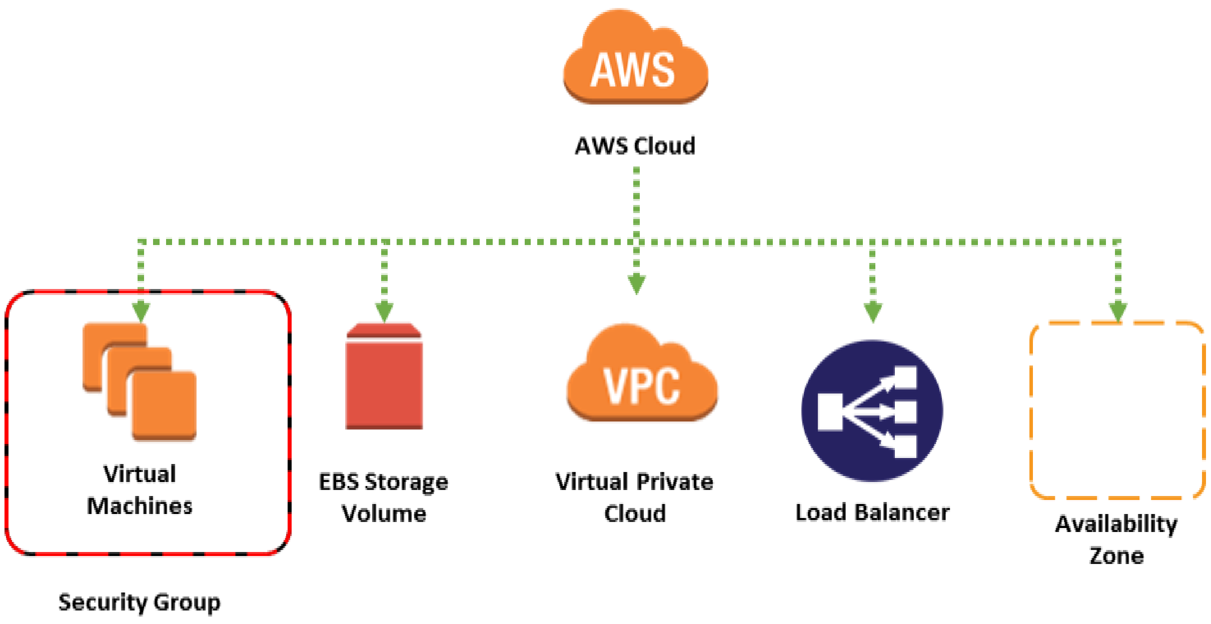
The diagram below provides an overview of the configuration of Ivanti DesktopNow and the Amazon Web Services platform in a non load balanced environment.


amazon web services
Virtual Machines
SQL
MS
PS
VPN
On Premises Domain Controller
Physical Desktops
MS = AppSense Management Server
PS = AppSense Personalization Server

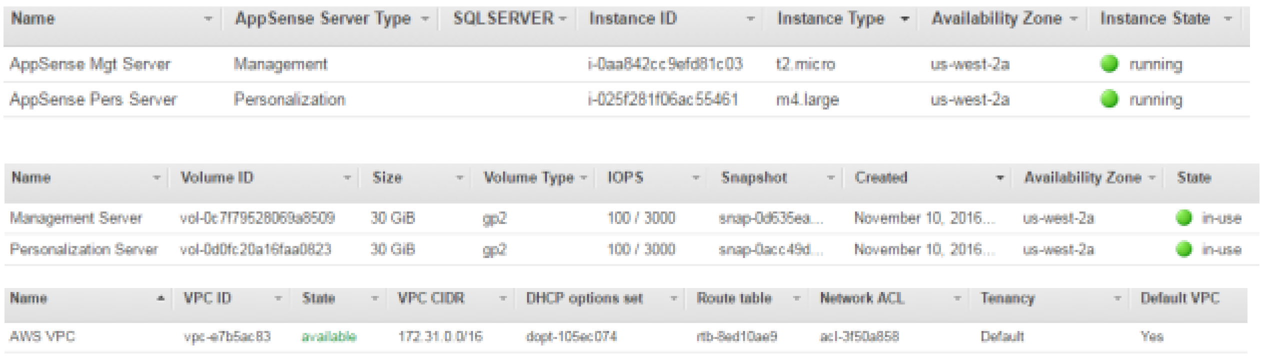
2 Basic Load balanced Amazon Web Services Environment

Amazon Web Services Configuration

The following resources were created within the AWS Management Console.



As can be seen that when compared to a none load balanced environment a number of additional AWS components are required. The configuration when viewed from the AWS Management Console can be seen below.



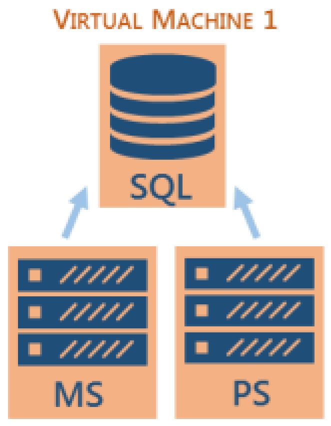
Name	AppSense Server Type	SQLSERVER	Instance ID	Instance Type	Availability Zone	Instance State
AppSense Mgt Server	Management		i-0aa842cc9efd81c03	t2.micro	us-west-2a	● running
AppSense Pers Server	Personalization		i-025f281f06ac55461	m4.large	us-west-2a	● running

Name	Volume ID	Size	Volume Type	IOPS	Snapshot	Created	Availability Zone	State
Management Server	vol-0c7f79528069a8509	30 GiB	gp2	100 / 3000	snap-0d635ea...	November 10, 2016...	us-west-2a	● in-use
Personalization Server	vol-0d0fc20a16faa0823	30 GiB	gp2	100 / 3000	snap-0acc49d...	November 10, 2016...	us-west-2a	● in-use

Name	VPC ID	State	VPC CIDR	DHCP options set	Route table	Network ACL	Tenancy	Default VPC
AWS VPC	vpc-e7b5ac83	available	172.31.0.0/16	dopt-105ec074	rtb-8ed10ae9	acl-3f50a858	Default	Yes

Amazon Web Services Virtual Machines

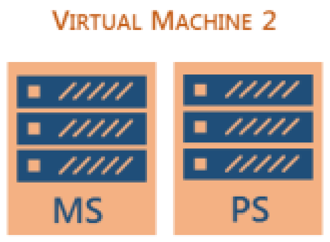
The AWS platform provides a flexible environment allowing for a wide range of computing solutions to be implemented. These machines can be access via a Remote Desktop (RDP) session in a similar way to that of an on-premises server.



MS = AppSense Management Server
PS = AppSense Personalization Server

Virtual Machine 1 was created and configured as follows:

- Microsoft Windows Server 2012 R2 with the necessary IIS Roles installed
- Microsoft SQL Server 2014 Standard Edition with Service Pack 1
- Ivanti DesktopNow v10
- VPN Client



MS = AppSense Management Server
PS = AppSense Personalization Server

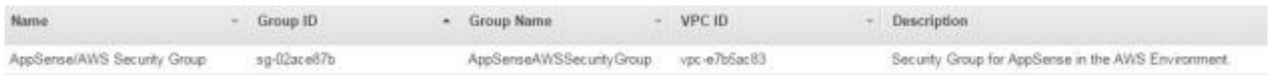
Virtual Machine 2 was created and configured as follows:

- Microsoft Windows Server 2012 R2 with the necessary IIS Roles installed
- Ivanti DesktopNow v10
- VPN Client

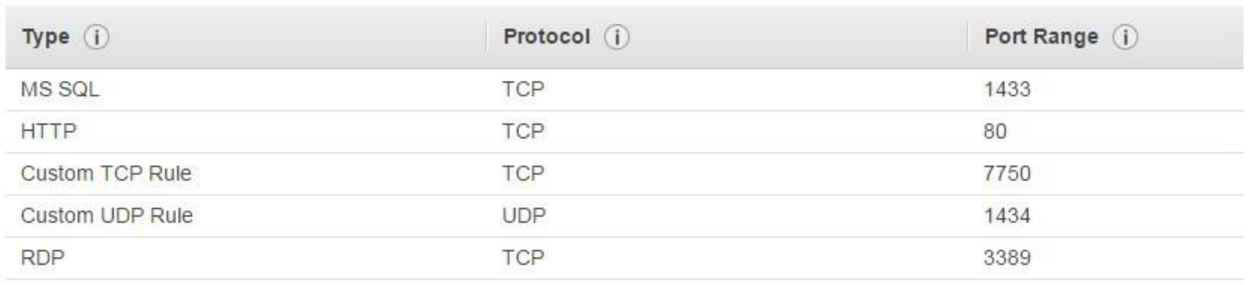
Security Group

A *security group* acts as a virtual firewall that controls the traffic for one or more instances. From within the AWS Management Console, you associate one or more security groups with the instance. You add rules to each security group that allow traffic to or from its associated instances.

In this instance a single security group was created to be used on the Virtual Network subnet.



Name	Group ID	Group Name	VPC ID	Description
AppSense/AWS Security Group	sg-02ace87b	AppSenseAWSSecurityGroup	vpc-e7b5ac83	Security Group for AppSense in the AWS Environment

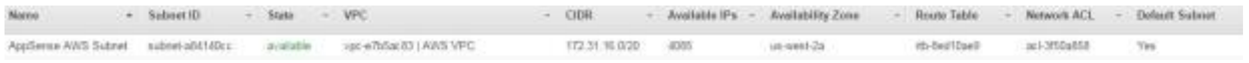


Type	Protocol	Port Range
MS SQL	TCP	1433
HTTP	TCP	80
Custom TCP Rule	TCP	7750
Custom UDP Rule	UDP	1434
RDP	TCP	3389

Virtual Private Cloud

A virtual private cloud (VPC) is a virtual network dedicated to an AWS account. It is logically isolated from other virtual networks in the AWS cloud. You can launch AWS resources, such as Amazon EC2 instances, into a VPC.

The following subnet was created within the Amazon Web Services Management Console.



Name	Subnet ID	State	VPC	CIDR	Available IPs	Availability Zone	Route Table	Network ACL	Default Subnet
AppSense AWS Subnet	subnet-a04140cc	available	vpc-e7b5ac83 | AWS VPC	172.31.16.0/20	4085	us-west-2a	rtb-5ed10ae0	acl-3f50a858	Yes

Virtual Machine instances were then added to the AWS Virtual Private Cloud.

Availability Zone

When working with two or more Virtual Machines within the AWS platform you should use an Availability Zone for each application tier. As an example, you might place domain controllers in one Availability Zone, SQL Servers in a second, and Web Servers in a third. Without this grouping, AWS is unable to distinguish between the application tiers for each Virtual Machine.

This could lead to a single point of failure in the hardware infrastructure causing an outage or a planned maintenance event rebooting all Virtual Machines in the same application tier simultaneously.

The two Virtual Machines that are used in this configuration were added to an Availability Zone.

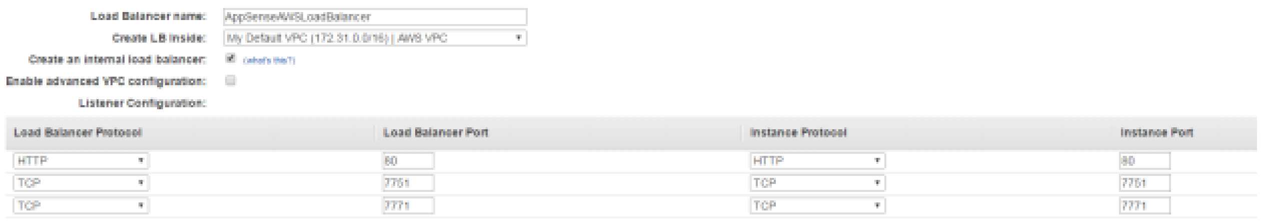
Network Load Balancer

Load Balancing distributes incoming application traffic across multiple EC2 instances, in multiple Availability Zones. This increases the fault tolerance of your services.

Within AWS the load balancer serves as a single point of contact for clients, which increases availability. You can add and remove instances from your load balancer as your needs change, without disrupting the overall flow of requests to an application or service.

Creation of a load balancer from within the Amazon Web Service Management Console is driven by a wizard, each of the configuration steps are shown below in the order that they are completed.

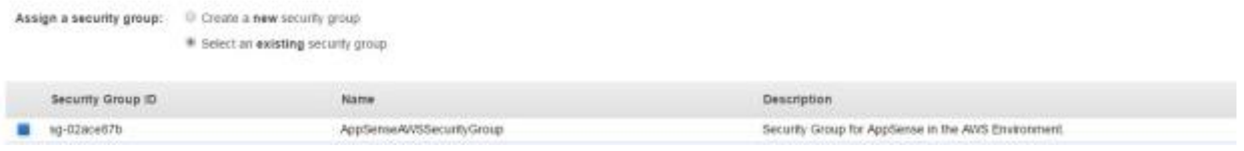
Step One



Load Balancer name:	AppSenseAWSLoadBalancer
Create LB inside:	My Default VPC (172.31.0.0/16) | AWS VPC
Create an internal load balancer:	✓ (what's this?)
Enable advanced VPC configuration:	☐
Listener Configuration:	

Load Balancer Protocol	Load Balancer Port	Instance Protocol	Instance Port
HTTP	80	HTTP	80
TCP	7751	TCP	7751
TCP	7771	TCP	7771

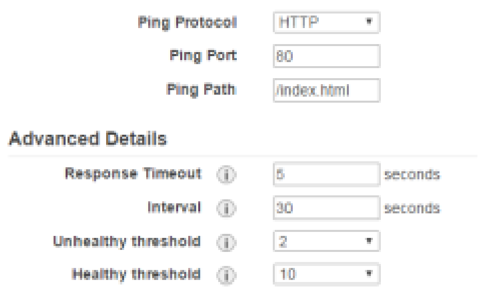
Step Two



Assign a security group: Create a **new** security group [✓ Select an **existing** security group]

	Security Group ID	Name	Description
■	sg-02ace67b	AppSenseAWSSecurityGroup	Security Group for AppSense in the AWS Environment

Step Three

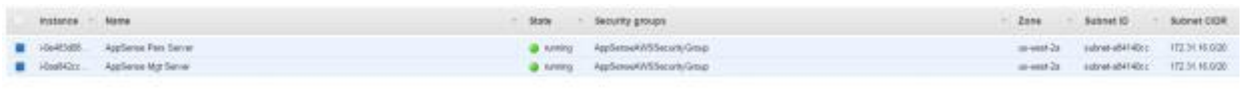


Ping Protocol	HTTP
Ping Port	80
Ping Path	/index.html

Advanced Details

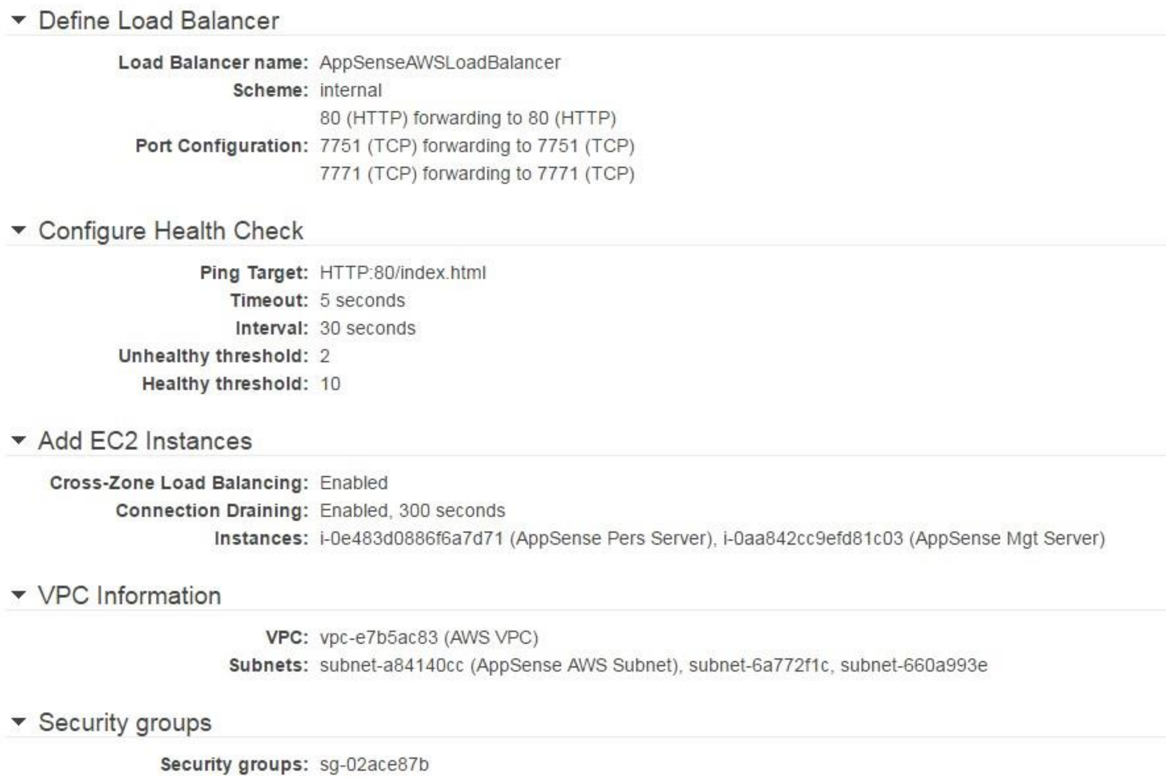
Response Timeout	5	seconds
Interval	30	seconds
Unhealthy threshold	2	
Healthy threshold	10	

Step Four



	Instance	Name	State	Security groups	Zone	Subnet ID	Subnet CIDR
■	i-0e4f3d85...	AppSense Pers Server	running	AppSenseAWSSecurityGroup	us-west-2a	subnet-a84140cc	172.31.16.0/20
■	i-0aa942cc...	AppSense Mgt Server	running	AppSenseAWSSecurityGroup	us-west-2a	subnet-a84140cc	172.31.16.0/20

Upon completion of the configuration wizard the following load balancing configuration was available.



▾ Define Load Balancer

Load Balancer name:	AppSenseAWSLoadBalancer
Scheme:	internal
Port Configuration:	80 (HTTP) forwarding to 80 (HTTP) 7751 (TCP) forwarding to 7751 (TCP) 7771 (TCP) forwarding to 7771 (TCP)

▾ Configure Health Check

Ping Target:	HTTP:80/index.html
Timeout:	5 seconds
Interval:	30 seconds
Unhealthy threshold:	2
Healthy threshold:	10

▾ Add EC2 Instances

Cross-Zone Load Balancing:	Enabled
Connection Draining:	Enabled, 300 seconds
Instances:	i-0e483d0886f6a7d71 (AppSense Pers Server), i-0aa842cc9efd81c03 (AppSense Mgt Server)

▾ VPC Information

VPC:	vpc-e7b5ac83 (AWS VPC)
Subnets:	subnet-a84140cc (AppSense AWS Subnet), subnet-6a772f1c, subnet-660a993e

▾ Security groups

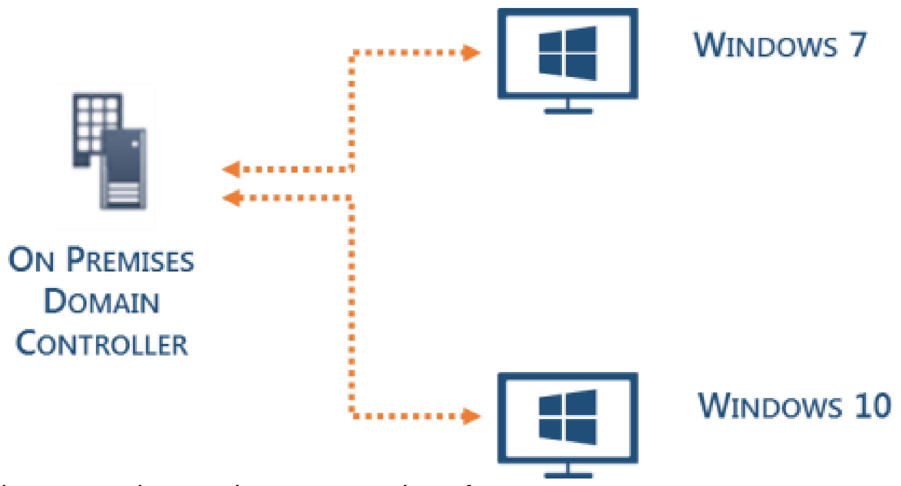
Security groups:	sg-02ace87b

Storage Account

AWS storage was configured to allow installation media such as Ivanti DesktopNow to be made available to all Virtual Machines.

On-Premises Environment

The on-premises environment was built to prove that physical desktops can be managed from the cloud, for example via an Environment Manager implementation housed within the Amazon Web Services platform.



The on-premises environment consists of:

- Microsoft Windows Server 2012 R2 configured as a Domain Controller
- Microsoft Windows 7 Ultimate
- Microsoft Windows 10 version 1607

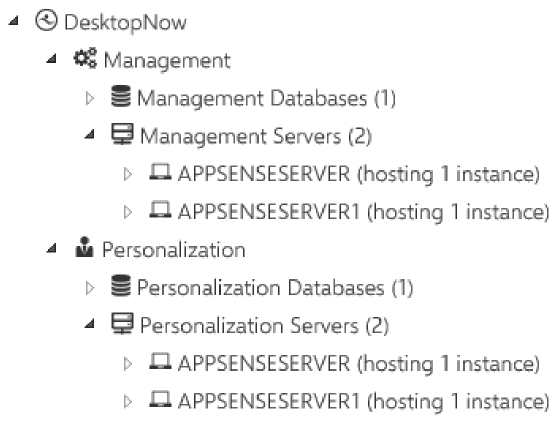
In addition, a Virtual Private Network was configured to allow the Amazon Web Services hosted servers to join and access the on-premises domain.

Ivanti DesktopNow Configuration

The AWS hosted Microsoft Windows Server 2012 R2 Virtual Machines were joined to the on- premises domain. DesktopNow v10 was installed using the Suite Installer on both of the Virtual Machines. Finally, the Server Configuration Portal was used to create the following databases within the workload of the SQL Server on one of the AWS Virtual Machines:

- Ivanti_MgtDB
- Ivanti_PersDB

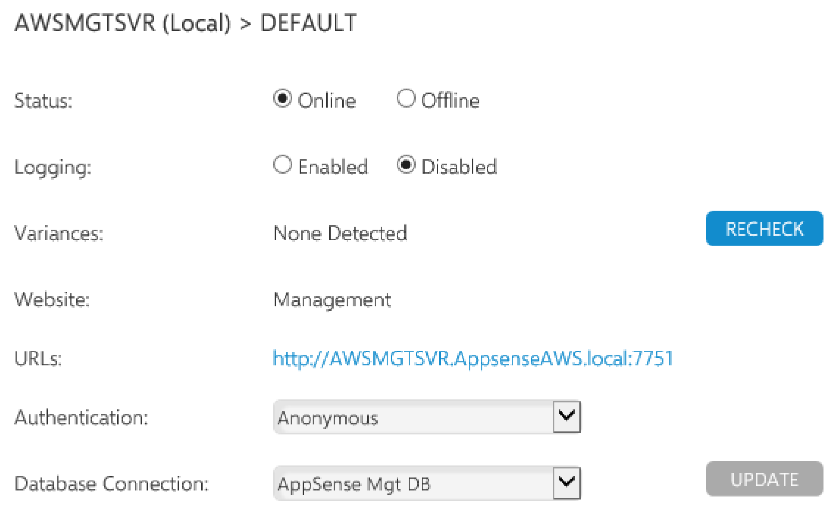
Each Virtual Machine was then configured to host an instance of the Ivanti Management and Personalization Server.



```
◢ DesktopNow
    ◢ Management
        ▷ Management Databases (1)
        ◢ Management Servers (2)
            ▷ APPSENSESERVER (hosting 1 instance)
            ▷ APPSENSESERVER1 (hosting 1 instance)
    ◢ Personalization
        ▷ Personalization Databases (1)
        ◢ Personalization Servers (2)
            ▷ APPSENSESERVER (hosting 1 instance)
            ▷ APPSENSESERVER1 (hosting 1 instance)
```

Ivanti Management Server Configuration

Both Ivanti Management Servers were configured in the following way:



AWSMGTSVR (Local) > DEFAULT

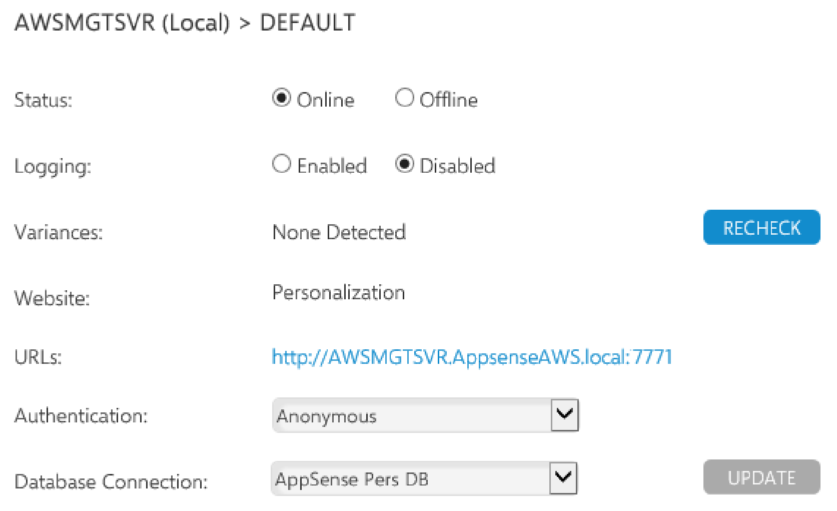
Status:	◉ Online ○ Offline	
Logging:	○ Enabled ◉ Disabled	
Variances:	None Detected	RECHECK
Website:	Management	
URLs:	http://AWSMGTSVR.AppsenseAWS.local:7751	
Authentication:	Anonymous	
Database Connection:	AppSense Mgt DB	UPDATE

Note

When using a load balanced configuration, it is necessary to set the Authentication method to Anonymous.

Ivanti Personalization Server Configuration

Both Ivanti Personalization Servers were configured in the following way:



AWSMGTSVR (Local) > DEFAULT

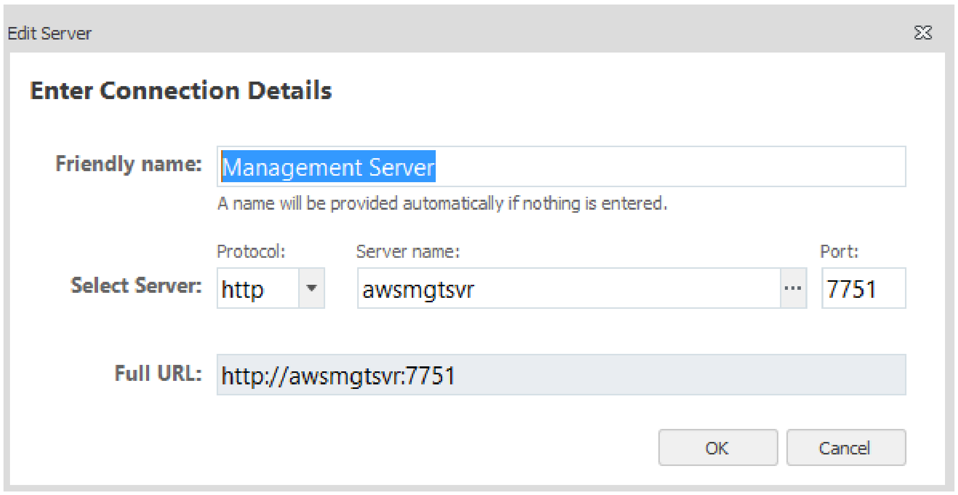
Status:	◉ Online ○ Offline	
Logging:	○ Enabled ◉ Disabled	
Variances:	None Detected	RECHECK
Website:	Personalization	
URLs:	http://AWSMGTSVR.AppsenseAWS.local:7771	
Authentication:	Anonymous	
Database Connection:	AppSense Pers DB	UPDATE

Note

When using a load balanced configuration, it is necessary to set the Authentication method to Anonymous.

Consoles

The Ivanti Management Center and Environment Manager consoles were configured to connect to the respective AWS hosted servers. There was no bespoke configuration required.


Edit Server

Enter Connection Details

Friendly name: Management Server

A name will be provided automatically if nothing is entered.

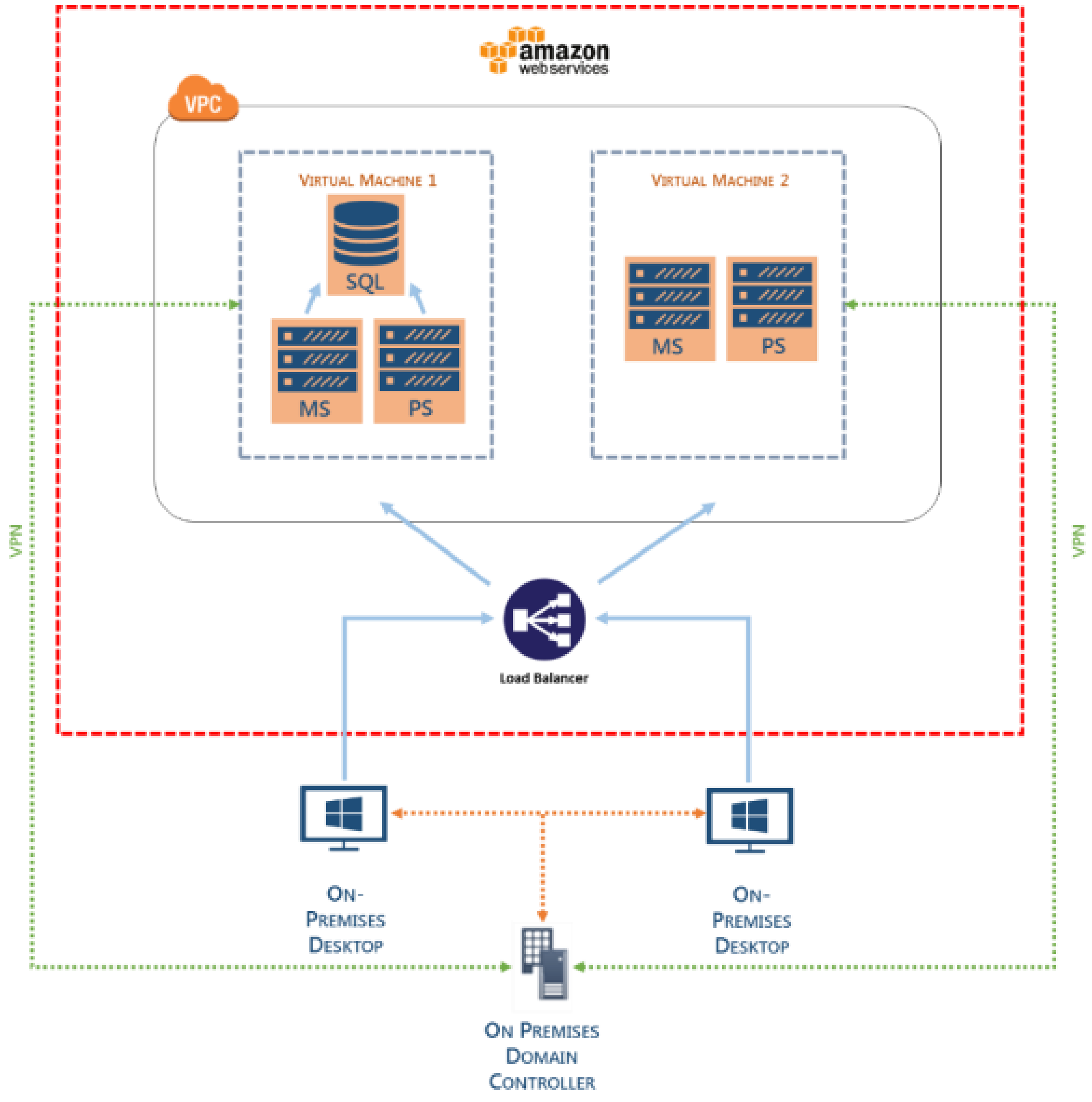
Select Server: Protocol: http | Server name: awsmgtsvr | Port: 7751

Full URL: http://awsmgtsvr:7751

Typical configuration such as Membership Rules and Access Credentials were configured and the agents for the Ivanti Management Center, Application Manager, Environment Manager and Performance Manager deployed. Again, no bespoke configuration was required.

Overall Configuration

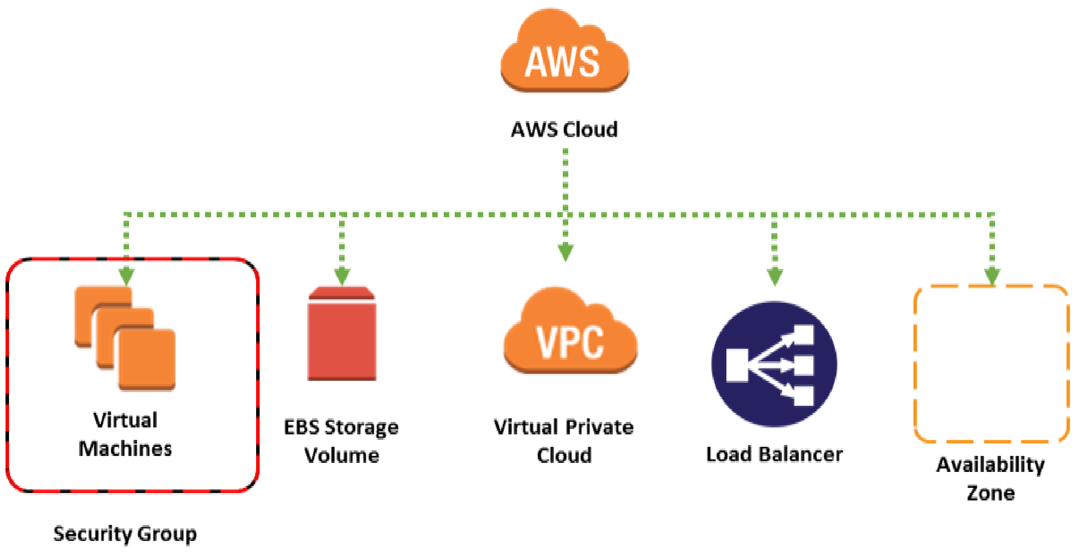
The diagram below provides an overview of the configuration of Ivanti DesktopNow and Amazon Web Services platform in a load balanced environment.



3 Advanced Load balanced Amazon Web Services Environment

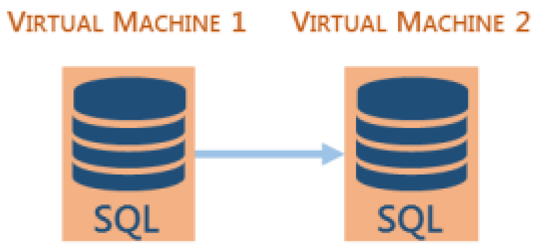
Amazon Web Services Configuration

The following resources were created within the AWS Management Console.



Amazon Web Services Virtual Machines

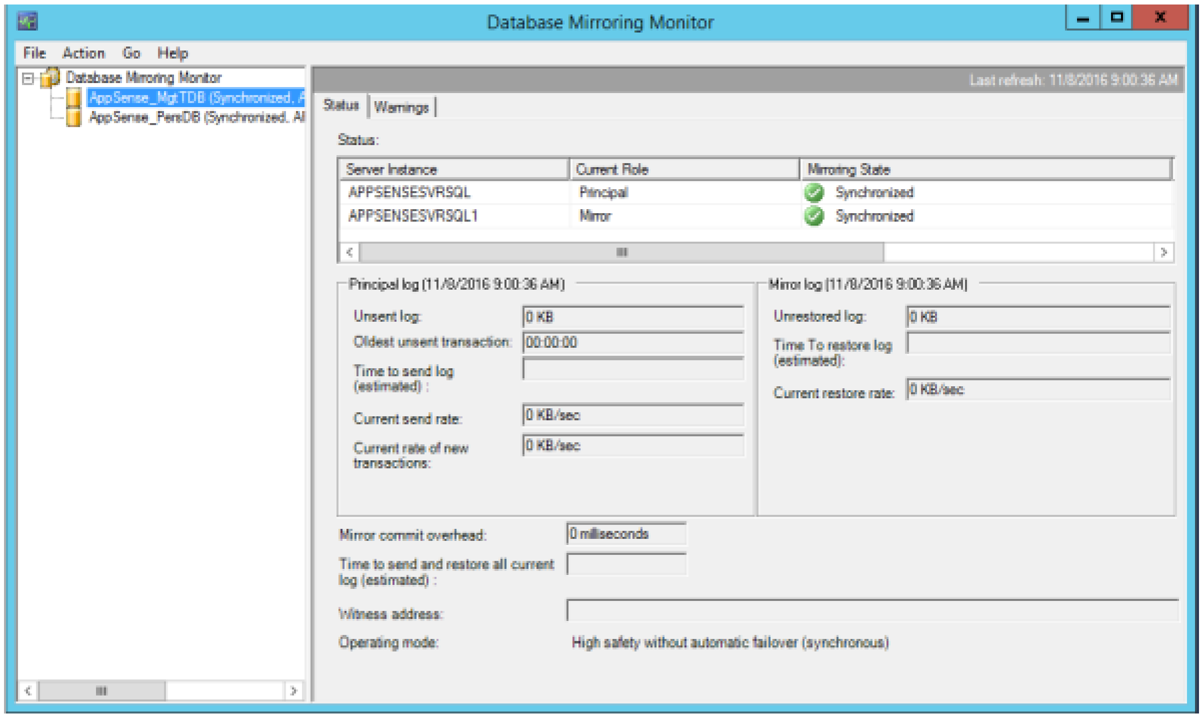
The AWS platform provides a flexible environment allowing for a wide range of computing solutions to be implemented. These machines can be access via a Remote Desktop (RDP) session in a similar way to that of an on-premises server.



Virtual Machine’s 1 and 2 were created as follows:

- Microsoft Windows Server 2012 R2
- Microsoft SQL Server 2014 Standard Edition with Service Pack 1 with Database Mirroring enabled.
- VPN Client

Database Mirroring was configured using Microsoft Best Practices. The following Microsoft TechNet article can be used as a starting point. The following illustrates the high-level configuration.



Database Mirroring Monitor
- AppSense_MgtTDB (Synchronized, A
- AppSense_PersDB (Synchronized, Al

Last refresh: 11/8/2016 9:00:36 AM

Status:

Server Instance	Current Role	Mirroring State
APPSENSESVRSQL	Principal	✓ Synchronized
APPSENSESVRSQL1	Mirror	✓ Synchronized

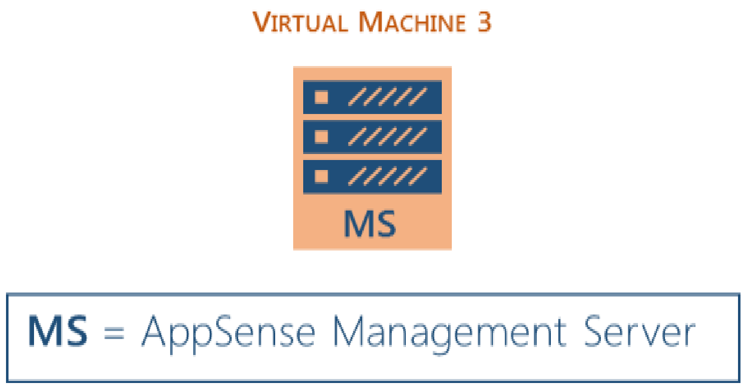
Principal log (11/8/2016 9:00:36 AM)

Unsent log:	0 KB
Oldest unsent transaction:	00:00:00
Time to send log (estimated) :	
Current send rate:	0 KB/sec
Current rate of new transactions:	0 KB/sec

Mirror log (11/8/2016 9:00:36 AM)

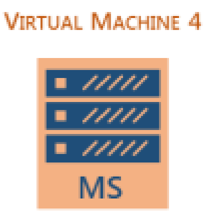
Unrestored log:	0 KB
Time To restore log (estimated):	
Current restore rate:	0 KB/sec

Mirror commit overhead:	0 milliseconds
Time to send and restore all current log (estimated) :	
Witness address:	
Operating mode:	High safety without automatic failover (synchronous)



Virtual Machine 3 was created and configured as follows:

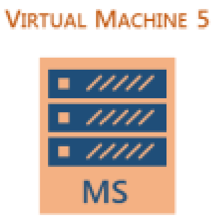
- Microsoft Windows Server 2012 R2 with the necessary IIS Roles installed
- Ivanti Management Server
- VPN Client



MS = AppSense Management Server

Virtual Machine 4 was created and configured as follows:

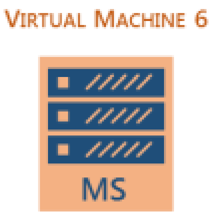
- Microsoft Windows Server 2012 R2 with the necessary IIS Roles installed
- Ivanti Management Server
- VPN Client



PS = AppSense Presonalization Server

Virtual Machine 5 was created and configured as follows:

- Microsoft Windows Server 2012 R2 with the necessary IIS Roles installed
- Ivanti Management Server
- VPN Client



PS = AppSense Presonalization Server

Virtual Machine 6 was created and configured as follows:

- Microsoft Windows Server 2012 R2 with the necessary IIS Roles installed
- Ivanti Management Server
- VPN Client

Virtual Machine 7

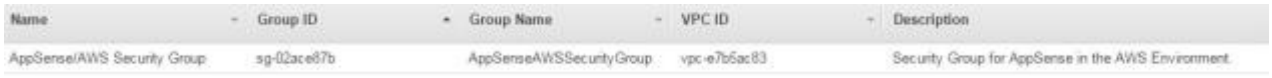
Virtual Machine 7 was created and configured as follows:

- Microsoft Windows 10 version 1607
- VPN Client

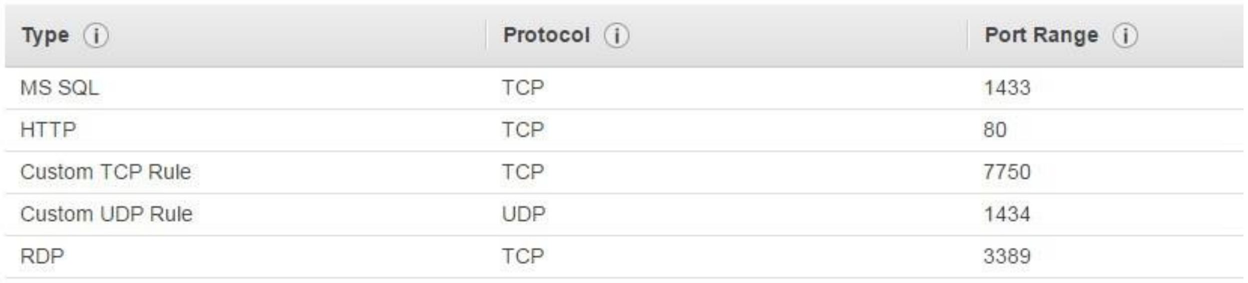
Security Group

A *security group* acts as a virtual firewall that controls the traffic for one or more instances. From within the AWS Management Console, you associate one or more security groups with the instance. You add rules to each security group that allow traffic to or from its associated instances.

In this instance a single security group was created to be used on the Virtual Network subnet.



Name	Group ID	Group Name	VPC ID	Description
AppSense/AWS Security Group	sg-02ace87b	AppSenseAWSSecurityGroup	vpc-e7b5ac83	Security Group for AppSense in the AWS Environment

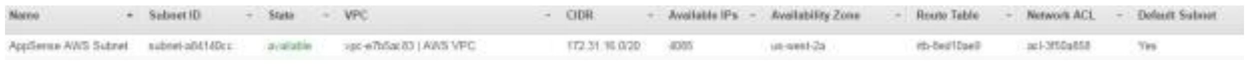


Type	Protocol	Port Range
MS SQL	TCP	1433
HTTP	TCP	80
Custom TCP Rule	TCP	7750
Custom UDP Rule	UDP	1434
RDP	TCP	3389

Virtual Private Cloud

A virtual private cloud (VPC) is a virtual network dedicated to an AWS account. It is logically isolated from other virtual networks in the AWS cloud. You can launch AWS resources, such as Amazon EC2 instances, into a VPC.

The following subnet was created within the Amazon Web Services Management Console.



Name	Subnet ID	State	VPC	CIDR	Available IPs	Availability Zone	Route Table	Network ACL	Default Subnet
AppSense AWS Subnet	subnet-a84140cc	available	vpc-e7b5ac83 | AWS VPC	172.31.16.0/20	4085	us-west-2a	rtb-8ed10ae8	acl-3f50a858	Yes

Virtual Machine instances were then added to the AWS Virtual Private Cloud.

Availability Zone

When working with two or more Virtual Machines within the AWS platform you should use an Availability Zone for each application tier. As an example, you might place domain controllers in one Availability Zone, SQL Servers in a second, and Web Servers in a third. Without this grouping, AWS is unable to distinguish between the application tiers for each Virtual Machine.

This could lead to a single point of failure in the hardware infrastructure causing an outage or a planned maintenance event rebooting all Virtual Machines in the same application tier simultaneously.

The seven Virtual Machines that are used in this configuration were added to an Availability Zone.

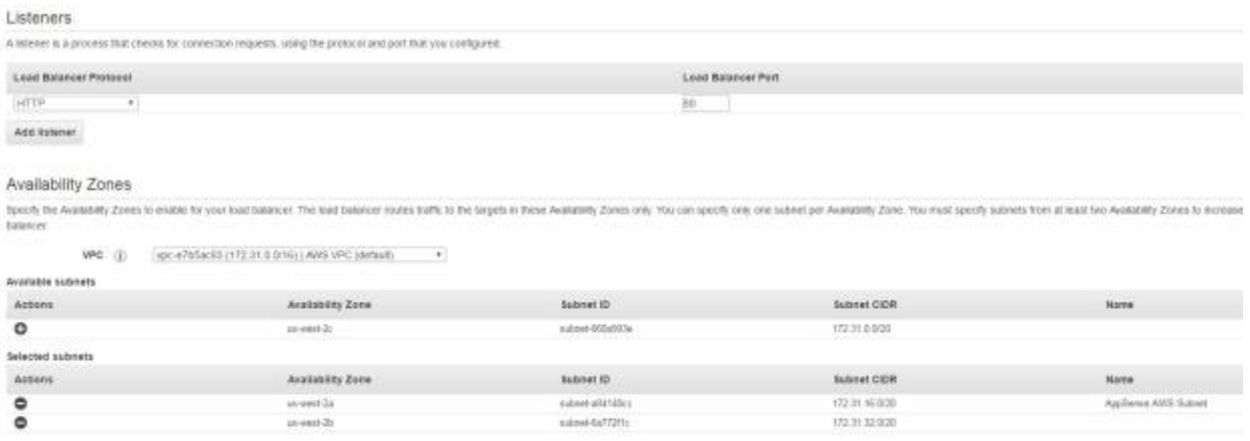
Network Load Balancer

Load Balancing distributes incoming application traffic across multiple EC2 instances, in multiple Availability Zones. This increases the fault tolerance of your services.

Within AWS the load balancer serves as a single point of contact for clients, which increases availability. You can add and remove instances from your load balancer as your needs change, without disrupting the overall flow of requests to an application or service.

Creation of an Application load balancer from within the Amazon Web Service Management Console is driven by a wizard, each of the configuration steps are shown below in the order that they are completed.

Step One



Listeners

A listener is a process that checks for connection requests, using the protocol and port that you configured.

Load Balancer Protocol	Load Balancer Port
HTTP	80

Add listener

Availability Zones

Specify the Availability Zones to enable for your load balancer. The load balancer routes traffic to the targets in these Availability Zones only. You can specify only one subnet per Availability Zone. You must specify subnets from at least two Availability Zones to increase balancer

VPC: vpc-e7b5ac63 (172.31.0.0/16) | AWS VPC (default)

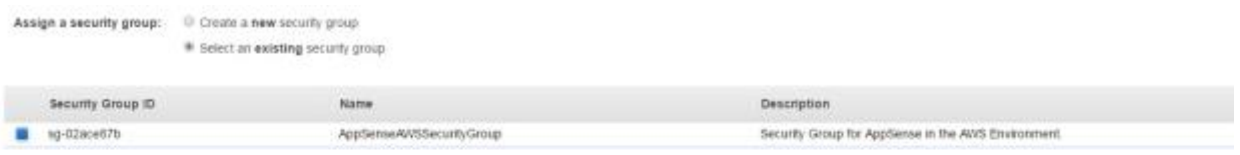
Available subnets

Actions	Availability Zone	Subnet ID	Subnet CIDR	Name
⊕	us-west-2c	subnet-002a697e	172.31.0.0/20	

Selected subnets

Actions	Availability Zone	Subnet ID	Subnet CIDR	Name
⊖	us-west-2a	subnet-a84140cc	172.31.16.0/20	AppSense AWS Subnet
⊖	us-west-2b	subnet-6a772f1c	172.31.32.0/20	

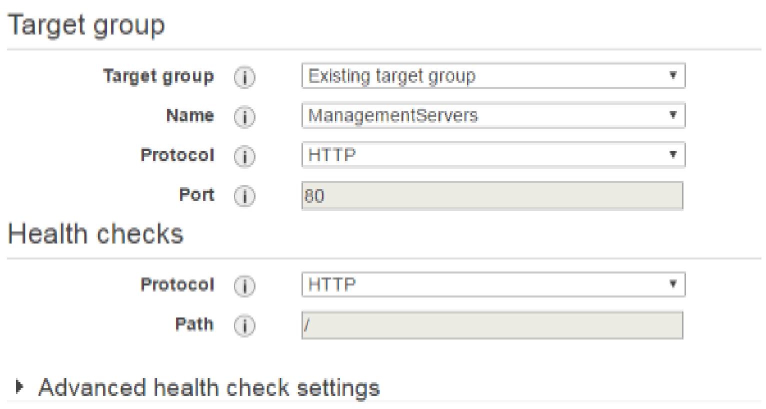
Step Two



Assign a security group: Create a **new** security group [✓ Select an **existing** security group]

	Security Group ID	Name	Description
■	sg-02ace67b	AppSenseAWSSecurityGroup	Security Group for AppSense in the AWS Environment

Step Three



Target group

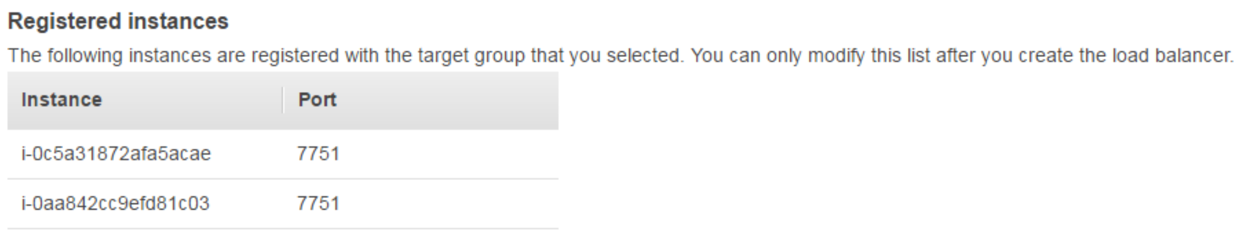
Target group	Existing target group
Name	ManagementServers
Protocol	HTTP
Port	80

Health checks

Protocol	HTTP
Path	/

Advanced health check settings

Step Four

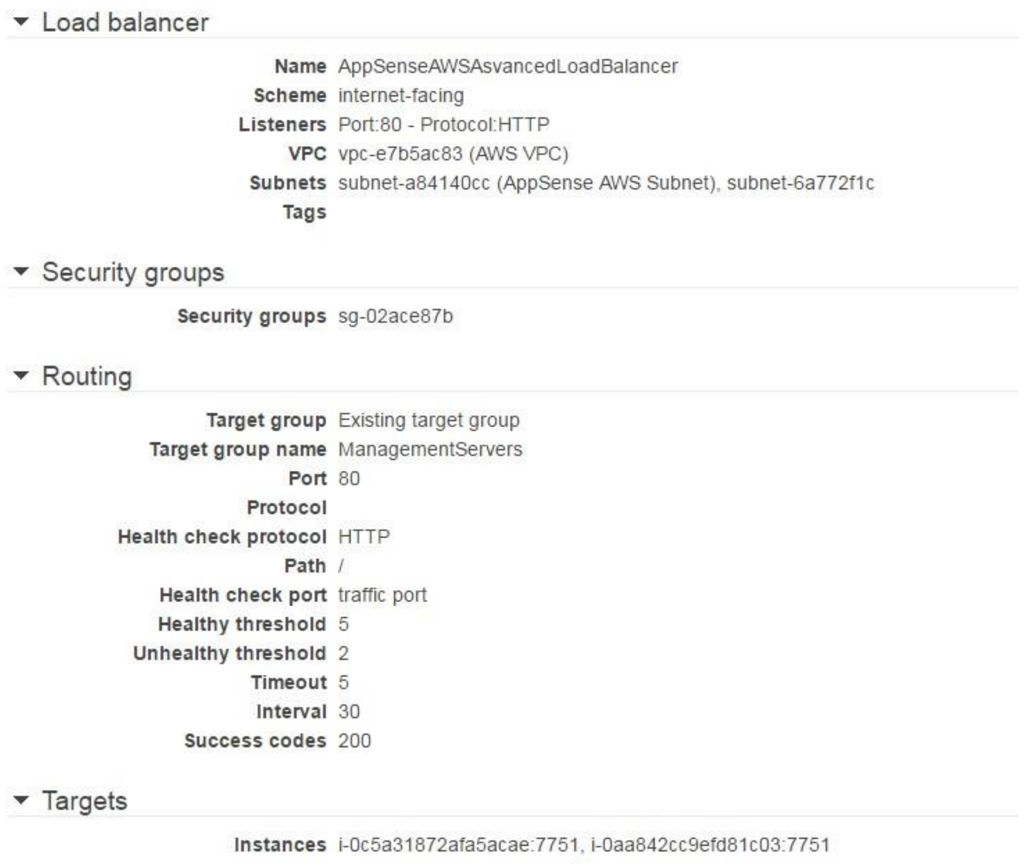


Registered instances

The following instances are registered with the target group that you selected. You can only modify this list after you create the load balancer.

Instance	Port
i-0c5a31872afa5acae	7751
i-0aa842cc9efd81c03	7751

Upon completion of the configuration wizard the following load balancing configuration was available.



Load balancer

Name	AppSenseAWSAsvancedLoadBalancer
Scheme	internet-facing
Listeners	Port:80 - Protocol:HTTP
VPC	vpc-e7b5ac83 (AWS VPC)
Subnets	subnet-a84140cc (AppSense AWS Subnet), subnet-6a772f1c
Tags	

Security groups

Security groups	sg-02ace87b

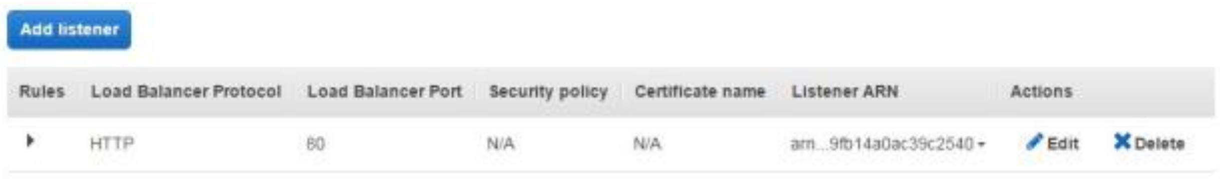
Routing

Target group	Existing target group
Target group name	ManagementServers
Port	80
Protocol	
Health check protocol	HTTP
Path	/
Health check port	traffic port
Healthy threshold	5
Unhealthy threshold	2
Timeout	5
Interval	30
Success codes	200

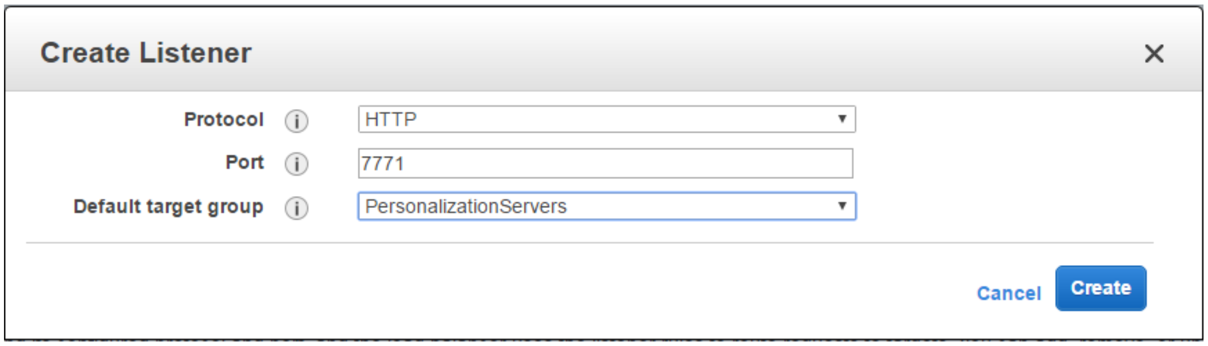
Targets

Instances	i-0c5a31872afa5acae:7751, i-0aa842cc9efd81c03:7751

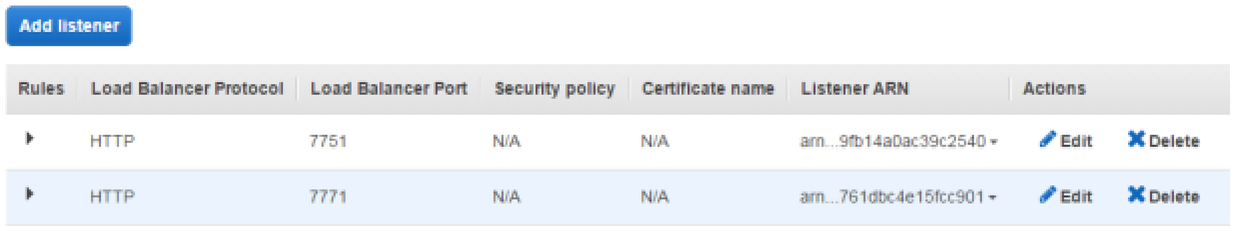
As we are using the same load balancer to route traffic to both our Ivanti Management and Personalization servers we must now add in a second target group. This is achieved by adding a new listener to the load balancer.



Rules	Load Balancer Protocol	Load Balancer Port	Security policy	Certificate name	Listener ARN	Actions	
▸	HTTP	80	N/A	N/A	arn...9fb14a0ac39c2540 ▾	Edit	Delete



Protocol: HTTP
Port: 7771
Default target group: PersonalizationServers



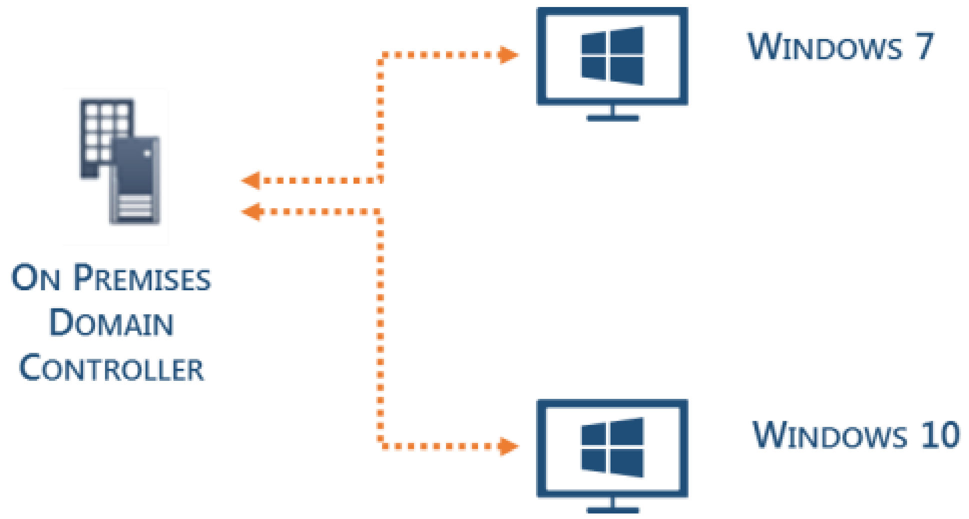
Rules	Load Balancer Protocol	Load Balancer Port	Security policy	Certificate name	Listener ARN	Actions	
▸	HTTP	7751	N/A	N/A	arn...9fb14a0ac39c2540 ▾	Edit	Delete
▸	HTTP	7771	N/A	N/A	arn...761dbc4e15fcc901 ▾	Edit	Delete

Storage Account

AWS storage was configured to allow installation media such as Ivanti DesktopNow to be made available to all Virtual Machines.

On-Premises Environment

The on-premises environment was built to prove that physical desktops can be managed from the cloud, for example via an Environment Manager implementation housed within the Amazon Web Services platform.



The on-premises environment consists of:

- Microsoft Windows Server 2012 R2 configured as a Domain Controller
- Microsoft Windows 7 Ultimate
- Microsoft Windows 10 version 1607

In addition, a Virtual Private Network was configured to allow the AWS hosted Virtual Machines to join and access the on-premises domain.

Ivanti DesktopNow Configuration

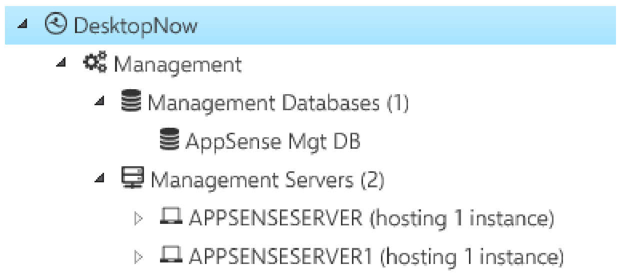
The AWS hosted Microsoft Windows Server 2012 R2 Virtual Machines were joined to the on- premises domain. DesktopNow v10 was installed using the Suite Installer on each of the Ivanti Virtual Machines. Finally, the Server Configuration Portal was used to create the following databases within the workload of the SQL Server.

- Ivanti_MgtDB
- Ivanti_PersDB

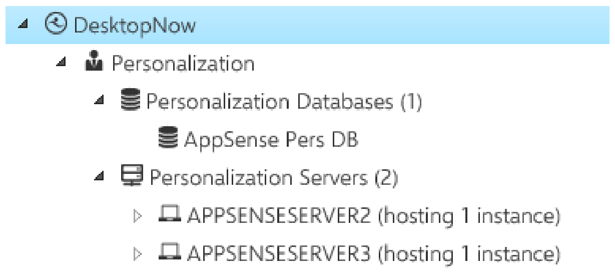
Note

These databases were both configured to use Database Mirroring.

Each Virtual Machine was then configured to host either an instance of the Ivanti Management or Personalization Server.



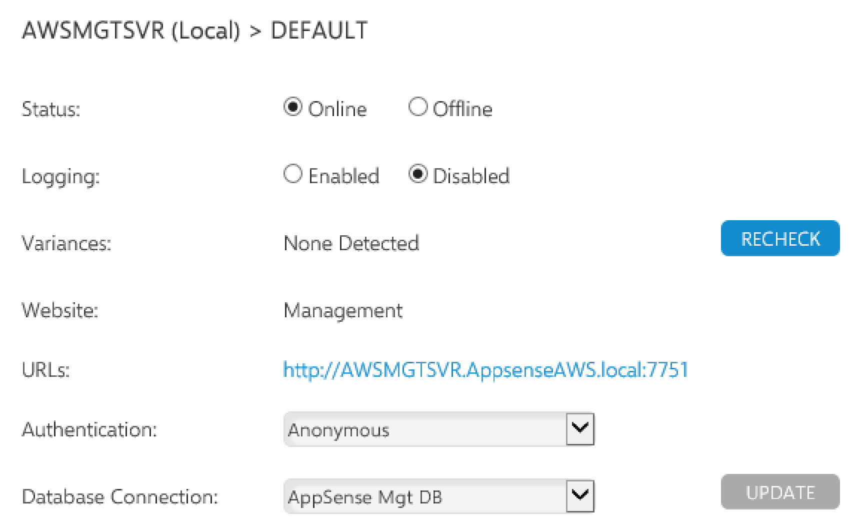
DesktopNow
- Management
  - Management Databases (1)
    - AppSense Mgt DB
  - Management Servers (2)
    - APPSENSESERVER (hosting 1 instance)
    - APPSENSESERVER1 (hosting 1 instance)



DesktopNow
- Personalization
  - Personalization Databases (1)
    - AppSense Pers DB
  - Personalization Servers (2)
    - APPSENSESERVER2 (hosting 1 instance)
    - APPSENSESERVER3 (hosting 1 instance)

Ivanti Management Server Configuration

Both Ivanti Management Servers were configured in the following way:


AWSMGTSVR (Local) > DEFAULT

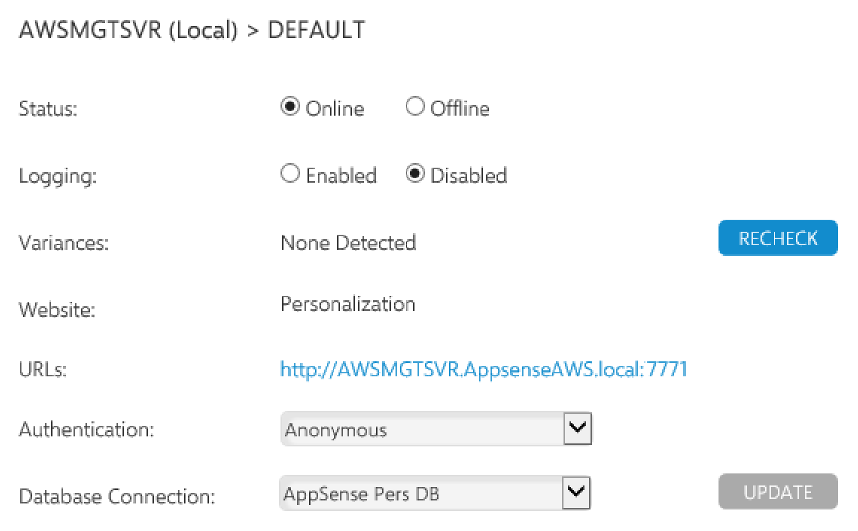
Status:	[✓ Online] Offline
Logging:	Enabled [✓ Disabled]
Variances:	None Detected
Website:	Management
URLs:	http://AWSMGTSVR.AppsenseAWS.local:7751
Authentication:	Anonymous
Database Connection:	AppSense Mgt DB

Note

When using a load balanced configuration, it was necessary to set the Authentication method to Anonymous.

Ivanti Personalization Server Configuration

Both Ivanti Personalization Servers were configured in the following way:


AWSMGTSVR (Local) > DEFAULT

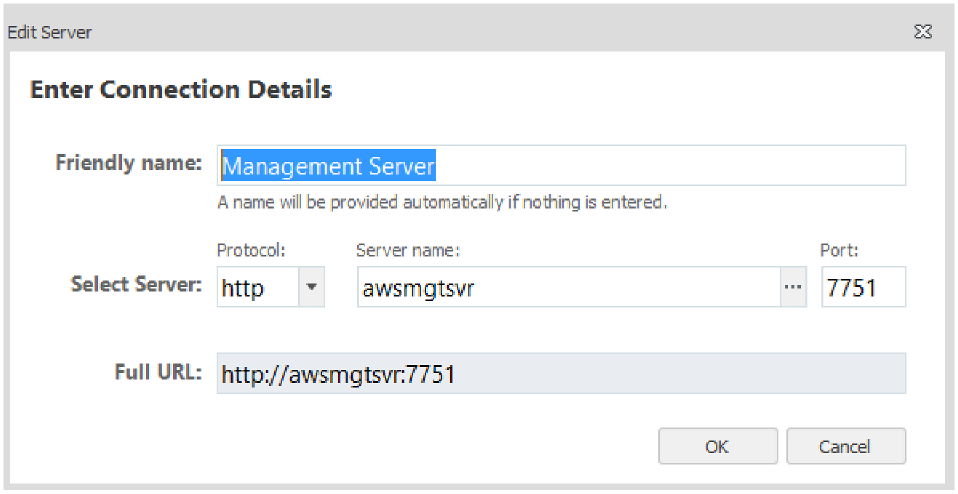
Status:	[✓ Online] Offline
Logging:	Enabled [✓ Disabled]
Variances:	None Detected
Website:	Personalization
URLs:	http://AWSMGTSVR.AppsenseAWS.local:7771
Authentication:	Anonymous
Database Connection:	AppSense Pers DB

Note

When using a load balanced configuration, it was necessary to set the Authentication method to Anonymous.

Consoles

The Ivanti Management Center and Environment Manager consoles were configured to connect to the respective AWS hosted servers. There was no bespoke configuration required.



Edit Server

Enter Connection Details

Friendly name: Management Server

A name will be provided automatically if nothing is entered.

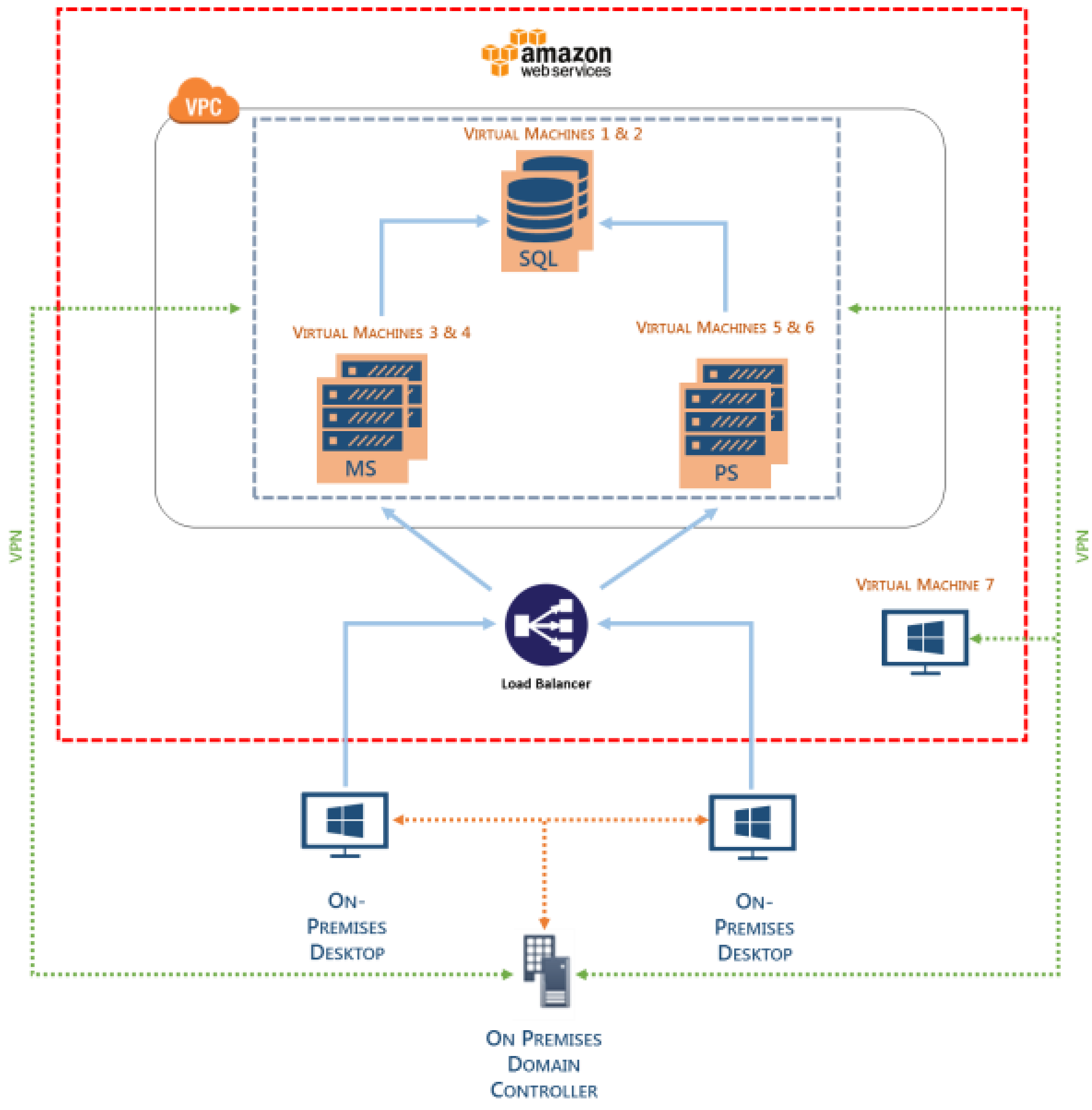
Select Server: Protocol: http | Server name: awsmgtsvr | Port: 7751

Full URL: http://awsmgtsvr:7751

Typical configuration such as Membership Rules and Access Credentials were configured and the agents for the Ivanti Management Center, Application Manager, Environment Manager and Performance Manager deployed. Again, no bespoke configuration was required.

Overall Configuration

The diagram below provides an overview of the configuration of Ivanti DesktopNow and Amazon Web Services platform in a load balanced environment.



Additional Reading

- Amazon EC2 - Virtual Server Hosting
- Getting Started with Amazon EC2