# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td>4</td>
</tr>
<tr>
<td>Chapter 1 Infoblox vNIOS for AWS</td>
<td>7</td>
</tr>
<tr>
<td>Chapter 2 Using the Infoblox vNIOS for AWS API Proxy</td>
<td>45</td>
</tr>
<tr>
<td>Chapter 3 Amazon Route 53 Integration</td>
<td>65</td>
</tr>
</tbody>
</table>
Preface

The preface describes the content and organization of this guide, how to find additional product information, and how to contact Infoblox Technical Support. It includes the following topics:

- Document Overview
- Documentation Organization
- Conventions
- Related Documentation
- Customer Care
  - User Accounts
  - Software Upgrades
  - Technical Support

Document Overview

This guide introduces the Infoblox vNIOS virtual appliance for AWS, including support for Infoblox Grids in the Amazon AWS Cloud. It describes how to install the Infoblox vNIOS virtual appliance on Amazon Web Services Versions 2 and Version 4. This manual describes the following:

- How to configure the Infoblox vNIOS for AWS virtual appliance as a Grid Master and Grid member.
- How to configure DNS services on Infoblox vNIOS for AWS virtual appliances.
- How to configure vDiscovery to discover assets and networks in your Amazon Cloud.
- How to configure the AWS API Proxy on select Grid members, including Infoblox vNIOS for AWS virtual appliances, to control and forward API calls from Grid Manager and from Amazon API clients to the AWS API.
- How to configure the Infoblox Amazon Route 53 integration.

For complete information about administering Infoblox appliances, refer to the Infoblox NIOS Documentation. For the latest Infoblox documentation, visit the Infoblox Support web site at https://support.infoblox.com.

Documentation Organization

This guide covers the following topics:

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 1, Infoblox vNIOS for AWS</td>
<td>Introduction and deployment information in the Amazon Web Services Cloud, setup procedures for experienced NIOS admins who are new to AWS, and setup procedures for experienced AWS administrators who wish to use NIOS to manage their AWS virtual private clouds.</td>
</tr>
<tr>
<td>Chapter 2, Using the Infoblox vNIOS for AWS API Proxy</td>
<td>Configuration and usage of the Infoblox AWS API Proxy for the NIOS Cloud Admin account and for AWS API Clients.</td>
</tr>
<tr>
<td>Chapter 3, Amazon Route 53 Integration</td>
<td>About Amazon Route 53 Integration and how to configure DNS data synchronization from Route 53.</td>
</tr>
</tbody>
</table>

Conventions

This guide follows the Infoblox documentation style conventions, as listed in the following table.

<table>
<thead>
<tr>
<th>Style</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>bold</td>
<td>Indicates anything that you input by clicking, choosing, selecting, typing or by pressing on the keyboard.</td>
</tr>
</tbody>
</table>
Style | Usage
--- | ---
input | Signifies command line entries that you type.
variable | Signifies variables typed into the GUI that you need to modify specifically for your configuration, such as command line variables, file names, and keyboard characters.

Navigation
Infoblox technical documentation uses an arrow "->" to represent navigation through the GUI. For example, to access member information, the description is as follows:

From the Grid tab, select the Grid Manager tab -> Members tab.

Related Documentation
Other Infoblox documentation:
- Infoblox CLI Guide
- Infoblox API Documentation
- Infoblox WAPI Documentation
- Infoblox CSV Import Reference
- Infoblox Installation Guide for the Trinzic 100 Appliance
- Infoblox Installation Guide for the 800 Series Platforms
- Infoblox Installation Guide for the 805 Series Platforms
- Infoblox Installation Guide for the 1400 Series Platforms
- Infoblox Installation Guide for the 1405 Series Platforms
- Infoblox Installation Guide for the 2200 Series Platforms
- Infoblox Installation Guide for the 2205Series Platforms
- Infoblox Installation Guide for the 4000 Series Platforms
- Infoblox Installation Guide for the Infoblox-4010 Appliance
- Infoblox Installation Guide for the IB-4030 and IB-4030-10GE Appliances
- Infoblox DNS Cache Acceleration Administrator Guide
- Infoblox Installation Guide for vNIOS for Microsoft Azure
- Infoblox Installation Guide for vNIOS for AWS
- Infoblox Installation Guide for vNIOS for VMware
- Infoblox Installation Guide for vNIOS on Microsoft 2008 R2 for Hyper-V
- Infoblox Installation Guide for vNIOS for KVM Hypervisor and KVM-based OpenStack
- Infoblox Safety Guide

To provide feedback on any of the Infoblox technical documents, please e-mail techpubs@infoblox.com.

Customer Care
This section addresses user accounts, software upgrades, licenses and warranties, and technical support.

User Accounts
The Infoblox appliance ships with a default user name and password. Change the default admin account password immediately after the system is installed to safeguard its use. Make sure that the NIOS appliance has at least one administrator account with superuser privileges at all times, and keep a record of your account information in a safe...
place. If you lose the admin account password, and did not already create another superuser account, the system will need to be reset to factory defaults, causing you to lose all existing data on the NIOS appliance. You can create new administrator accounts, with or without superuser privileges.

Software Upgrades

Software upgrades are available according to the Terms of Sale for your system. Infoblox notifies you when an upgrade is available. Register immediately with Infoblox Technical Support at http://www.infoblox.com/support/customer/evaluation-and-registration to maximize your Technical Support.

Technical Support

Infoblox Technical Support provides assistance via the Web, e-mail, and telephone. The Infoblox Support web site at https://support.infoblox.com provides access to product documentation and release notes, but requires the user ID and password you receive when you register your product online at: http://www.infoblox.com/support/customer/evaluation-and-registration.
Chapter 1 Infoblox vNIOS for AWS

This chapter provides information about the Infoblox vNIOS for AWS (Amazon Web Services) virtual appliance and explains how to deploy it in the AWS public cloud. vNIOS for AWS is a virtual Infoblox appliance designed for operation in Amazon VPCs (Virtual Private Clouds). You can deploy large, robust, manageable and cost effective Infoblox Grids in your AWS cloud, or extend your private Infoblox Grid to your virtual private cloud resources in AWS.

Note
Infoblox vNIOS for AWS supports AWS Versions 2 and 4.

This chapter includes the following topics:

- About Infoblox vNIOS for AWS
  - Key Features of Infoblox vNIOS for AWS
  - About the Infoblox AWS API Proxy
  - Infoblox vNIOS for AWS AMI Shapes
- Infoblox vNIOS for AWS Deployments
  - Shared-Services VPC Deployments
  - Multiple-VPCs Full-Public-Cloud Deployment
- Configuring AWS Access for NIOS Cloud Admins
  - Assigning AWS User Credentials to the NIOS Cloud Admin Account
  - Configuring the NIOS Cloud Admin User
  - Setting Administrative Permissions for Infoblox vNIOS for AWS
- Provisioning vNIOS for AWS Using the Paid NIOS Model
- Provisioning vNIOS for AWS Using the BYOL Model
  - Obtaining the Infoblox vNIOS for AWS AMI
  - Defining Network Settings for Your New Infoblox vNIOS for AWS Instance
  - Initializing New Infoblox vNIOS for AWS Instances with the AWS User Data Field
  - Defining Storage Settings for your New Instance
  - Using AWS Tags with Infoblox Extensible Attributes to Identify Resources for IP Address Assignments
  - Defining an AWS Instance Security Group
  - Completing Your Infoblox vNIOS for AWS Instance Launch
- Provisioning Infoblox vNIOS for AWS using Elastic Scaling
  - Initial Infoblox vNIOS for AWS Configuration with Elastic Scaling
  - Defining AWS User Data Settings for Infoblox vNIOS for AWS Instances using Elastic Scaling
  - Continuing Infoblox vNIOS for AWS Instance Configuration
- Additional Configuration for vNIOS for AWS
  - Setting the DNS Name Server for the Amazon VPC
  - Starting and Stopping the vNIOS Appliance in AWS
  - Delegating NIOS Objects to the Infoblox vNIOS for AWS Grid Member
- vDiscovery on AWS VPCs
  - Credentials for vDiscovery
  - Objects Discovered and Collected by vDiscovery
  - Creating DNS Records for Discovered IP Addresses
- Using Infoblox vNIOS for AWS with AWS VPCs, Subnets and IP Addresses
  - Subnet CIDR Guidelines
  - NIOS-AWS Subnet Size Restrictions
  - NTP and Hybrid Cloud Synchronization
  - Using an Elastic IP Address
  - NIOS Treatment of AWS Public IP Addresses and Elastic IP Addresses
- Common Guidelines for Infoblox vNIOS for AWS Usage
About Infoblox vNIOS for AWS

You can use Infoblox vNIOS for AWS virtual appliances to provide enterprise-grade DNS and IPAM services across your AWS VPCs. Instead of manually provisioning IP addresses and DNS name spaces for network devices and interfaces, you can deploy an Infoblox vNIOS for AWS instance as one of the following:

- A standalone NIOS appliance to provide DNS services in your Amazon VPC.
- A virtual cloud member tied to an on-premises (non-Cloud) NIOS Grid.
- A Grid Master synchronizing with other AWS-hosted vNIOS Grid members in your Amazon VPC, and across VPCs or Availability Zones in different Amazon Regions.

Infoblox vNIOS for AWS supports fully automated allocations of IP addresses and DNS record creation for business workloads. You can reduce provisioning errors, and quickly provision and de-commission resources in your public cloud. NIOS handles all IP address management of AWS instances, provisioning and managing all private IP addresses through IPAM. You can delegate different networks in your Amazon VPC to different Infoblox vNIOS for AWS Cloud Platform Appliances for management.

After starting your Infoblox vNIOS for AWS instances, you use vDiscovery to discover, and to periodically re-discover, all resources in the networks inside your Amazon cloud. Infoblox vNIOS for AWS also supports flexible deployment options, including the Elastic Scaling feature for automatic provisioning of Infoblox vNIOS for AWS instances in your Amazon VPCs. For information about vDiscovery and Elastic Scaling, refer to the Infoblox NIOS Documentation. For requirements to use vDiscovery on AWS, see vDiscovery on AWS VPCs.

Infoblox vNIOS for AWS deployments may be part of a hybrid cloud strategy that supports managing organization assets on public clouds (e.g. Amazon) and on private clouds (e.g. OpenStack and VMware). With the inclusion of Infoblox vNIOS for AWS, the Infoblox Cloud Network Automation solution provides core network services to centralize management across the following:

- AWS EC2 public clouds
- VMware and OpenStack private clouds
- Traditional enterprise networks

You use the Grid Manager as a unified console to support all Cloud platforms, to ensure uniform DNS policies, and to provision network and IP addresses. Infoblox vNIOS for AWS also supports DNS Firewall licensing for enhanced security and malware detection within your AWS cloud. For information about Infoblox Cloud Network Automation, refer to the Infoblox NIOS Documentation.

Infoblox vNIOS for AWS supports extensions to the Amazon API. Grid members or the Grid Master can act as an API proxy to send Cloud administrators’ AWS automation, provisioning and management requests to the Amazon API. You can use NIOS configuration tasks or cloud API clients to send AWS API requests through any NIOS or vNIOS appliance designated as the API Proxy. For information, see Setting Up the Infoblox AWS API Proxy and Setting Up the Infoblox AWS API Proxy.

In this manual, you learn how to provision and manage Infoblox vNIOS for AWS instances in the Amazon Web Services public cloud platform.

Key Features of Infoblox vNIOS for AWS

You can deploy one or more Infoblox vNIOS for AWS instances in your Amazon VPC, automatically provision them to join the NIOS Grid, and manage your Amazon VPC DNS services through Grid Manager. Utilizing Cloud Network Automation with your Amazon VPC can minimize human errors by streamlining with IPAM, improve visibility of your cloud networks,
and increase the flexibility of your cloud environment.

Supported features for Infoblox vNIOS for AWS include the following:

- Run NIOS Grid members or Grid Masters in the AWS public cloud with secure connectivity and synchronization with Infoblox NIOS Grids in on-premises private networks, in a hybrid public cloud/on-premises private cloud configuration.
- Run an entire NIOS Grid in your AWS VPC.
- Support large-scale deployments spanning dozens of Amazon VPCs.
- Perform vDiscovery for all resources in your Amazon VPC (requires enabling of DNS resolvers under Grid Properties in NIOS).
- Take advantage of the pay-as-you-go licensing model (known as the Paid NIOS model) in the AWS Marketplace. When you use the Paid NIOS model to launch the vNIOS for AWS virtual appliance, the virtual appliance comes pre-installed with the following permanent licenses: vNIOS, Grid, DNS, DHCP, and CNA (Cloud Network Automation). As long as the virtual appliance is up and running, you can use the NIOS features that these licenses provide.
- Use Infoblox vNIOS for AWS to start DHCP services for the private networks outside AWS network.
- Use Elastic Scaling for flexible management of NIOS appliance licensing features across the entire Grid, including Cloud Management Platform (CMP) integration. Elastic Scaling enables automatic deployment of Infoblox vNIOS for AWS instances in your Amazon VPC.
  - Elastic Scaling allows you to use your pre-existing NIOS or vNIOS feature licenses for activation of NIOS features on newly launched Infoblox vNIOS for AWS EC2 instances.
- Use Infoblox's AWS API Proxy to send configuration directions to the Amazon API.
- DNS zone transfers to keep VPC-based Infoblox vNIOS for AWS instances in synchronization with the customer-premises NIOS Grid.
- Infoblox vNIOS for AWS is compatible with standard operations in the AWS environment.
- Synchronize DNS data from Amazon Route 53 to NIOS to achieve unified DNS data visualization across your on-premise networks and hybrid clouds.

Infoblox vNIOS for AWS also provides support for advanced configurations and scripting automation, with several use cases described in this chapter:

- Manually define the Grid Master, admin password, and temporary feature licenses for new Infoblox vNIOS for AWS instances using the AWS User Data feature.
- Manually define auto-configuration of DNS service for the entire VPC by configuring the appropriate DHCP-OPTION-SET in your AWS VPC.
- Amazon scripting automation to enable advanced AWS users to integrate API workflows into the NIOS Grid.

About the Infoblox AWS API Proxy

You can designate at least one Cloud Platform appliance (or the Grid Master) as an API Proxy to the AWS API. You can designate an Infoblox vNIOS for AWS instance as an API Proxy.

You use Grid Manager (the NIOS GUI) to set up one or more appliances as an AWS API Proxy. The following configurations support AWS API Proxy:

- Grid Master (requires the Cloud Network Automation license);
- Grid Member (requires the Cloud Platform license).

The appliance acting as the API Proxy may be located in the AWS VPC or be an on-premises appliance. The API Proxy accepts Cloud API requests from an AWS API client on a reserved TCP port. The proxy parses the Cloud API requests into generic Amazon API requests that can be processed in the Amazon cloud. The proxy then forwards those requests to Amazon services to read, write and modify cloud network data, including IP addresses.

You may also run an on-premises NIOS Cloud Platform appliance as the API Proxy. The Grid Master (with a Cloud Network Automation license) may also operate as the API Proxy. If you choose to have Infoblox vNIOS for AWS instance as the API Proxy, ensure that all SSH sessions for CLI sessions to the Infoblox vNIOS for AWS instance are routed through the network connection to the VPC.

For more information on the Infoblox AWS API Proxy, see Setting Up the Infoblox AWS API Proxy.
Infoblox vNIOS for AWS AMI Shapes and Regions

This section lists the Infoblox vNIOS for AWS appliance models and their specifications. You load the Infoblox vNIOS for AWS AMI (Amazon Machine Image) from the Infoblox Community AMI page and deploy it in your chosen VPC. For information, see Obtaining the Infoblox vNIOS for AWS AMI.

Depending on your AWS region and the NIOS release your system is running, you may need to select a different EC2 shape for the corresponding vNIOS for AWS model. The following is a summary about the AWS regions and the associated EC2 shapes:

- **For NIOS 8.1.x and earlier**: Use "m4.xlarge" for US East (Ohio), Canada (Central), EU (London), Asia Pacific (Seoul) and Asia Pacific (Mumbai) regions. These regions do not have the "m3.xlarge" shape.
- **For NIOS 8.2.x and later**: Use "r4.xlarge" for Canada (Central) and EU (London) regions. These regions do not have the "r3.xlarge" shape.
- Use the "m3.xlarge" and "r3.xlarge" shapes for the rest of all regions: US East (N. Virginia), US West (N. California), US West (Oregon), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Sydney), Asia Pacific (Tokyo), and South America (Sao Paulo).

If the required EC2 shape is not available, you can select the nearest compatible shape to create an instance. For more information on AWS instance types, see [https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-types.html](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-types.html)

The following table lists the supported vNIOS for AWS models for NIOS 8.1.x and earlier releases.

<table>
<thead>
<tr>
<th>NIOS VM Models</th>
<th>Grid Master/Grid Master Candidate (Yes/No)</th>
<th>vCPU</th>
<th>Amazon EC2 Shape</th>
<th>Amazon EC2 Shape for US East (Ohio), Canada (Central), EU (London), Asia Pacific (Seoul), and Asia Pacific (Mumbai) regions</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP-V800</td>
<td>No</td>
<td>2</td>
<td>c4.large</td>
<td>c4.large</td>
</tr>
<tr>
<td>CP-V1400</td>
<td>No</td>
<td>4</td>
<td>m3.xlarge</td>
<td>m4.xlarge</td>
</tr>
<tr>
<td>CP-V2200</td>
<td>No</td>
<td>4</td>
<td>m3.xlarge</td>
<td>m4.xlarge</td>
</tr>
<tr>
<td>TE-V820 *1</td>
<td>No</td>
<td>2</td>
<td>c4.large</td>
<td>c4.large</td>
</tr>
<tr>
<td>TE-V1420 *1</td>
<td>Yes</td>
<td>4</td>
<td>m3.xlarge</td>
<td>m4.xlarge</td>
</tr>
<tr>
<td>TE-V2220 *1</td>
<td>Yes</td>
<td>4</td>
<td>m3.xlarge</td>
<td>m4.xlarge</td>
</tr>
</tbody>
</table>

The following table lists the supported vNIOS for AWS models for NIOS 8.2.0 and later releases.
| NIOS VM Models | Grid Master/Grid Master Candidate (Yes/No) | vCPU | Amazon EC2 Shape | Amazon EC2 Shape for US East (Ohio), Asia Pacific (Seoul) and Asia Pacific (Mumbai) regions | Amazon EC2 Shape for Canada (Central) and EU (London) regions |
|----------------|------------------------------------------|------|------------------|------------------------------------------------------------------------------------------------|
| CP-V800        | No                                       | 2    | r4.large         | c4.large                                                                                 | c4.large                                       |
| CP-V1400       | No                                       | 4    | m3.xlarge        | m4.xlarge                                                                                 | m4.xlarge                                     |
| CP-V2200       | No                                       | 4    | m3.xlarge        | m4.xlarge                                                                                 | m4.xlarge                                     |
| TE-V825 *1     | Yes                                      | 2    | r4.large         | r4.large                                                                                 | r4.large                                      |
| TE-V1425 *1    | Yes                                      | 4    | r3.xlarge        | r3.xlarge                                                                                 | r4.xlarge                                     |
| TE-V2225 *1    | Yes                                      | 8    | r4.2xlarge       | r4.2xlarge                                                                                | r4.2xlarge                                    |

The following table lists the supported vNIOS for AWS IB-FLEX models for NIOS 8.3.0 and later releases. For more information, see About IB-FLEX.

<table>
<thead>
<tr>
<th>NIOS VM Models</th>
<th>Grid Master/Grid Master Candidate (Yes/No)</th>
<th>vCPU</th>
<th>Amazon EC2 Shape</th>
</tr>
</thead>
<tbody>
<tr>
<td>IB-FLEX</td>
<td>Yes</td>
<td>4</td>
<td>m4.xlarge</td>
</tr>
<tr>
<td>IB-FLEX</td>
<td>Yes</td>
<td>8</td>
<td>m4.2xlarge</td>
</tr>
<tr>
<td>IB-FLEX</td>
<td>Yes</td>
<td>16</td>
<td>m4.4xlarge</td>
</tr>
</tbody>
</table>

**NOTE:**
- *1 All TE vNIOS appliances for AWS do not support downgrading from NIOS 8.2.x to any earlier NIOS releases.
- *2 CP-V805, CP-V1405, and CP-V2205 do not support downgrading from NIOS 8.4.x to any earlier NIOS releases. Only supported in NIOS 8.4 and later releases.

The following table lists the supported vNIOS for AWS IB-FLEX models for NIOS 8.3.0 and later releases. For more information, see About IB-FLEX.
Infoblox vNIOS for AWS Deployments

Note: You can obtain the Infoblox vNIOS for AWS AMI by going to the Community AMI page in Amazon Web Services. Use 'NIOS' or 'Infoblox' as the search term to locate the AMI. For information, see Obtaining the Infoblox vNIOS for AWS AMI.

Amazon Web Services cloud networks are called virtual private clouds (VPCs). Using Infoblox vNIOS for AWS, you can control, provision and manage IP addresses in your AWS VPCs. Supported Amazon virtual private cloud topologies include the following:

- Shared Services VPCs peered to multiple business workload VPCs.

For control through Amazon's API, you designate at least one Cloud Platform appliance (or the Grid Master) as an API Proxy to the Amazon Web Services API. NIOS administrators, and AWS users that will use NIOS to manage their VPCs, must designate an on-premise NIOS appliance or an Infoblox vNIOS for AWS instance as an API Proxy. (You may also use more than one, depending on your deployment.) You can use a computer as an AWS API client to issue AWS API calls, or run scripts to automate longer sequences of AWS API calls. For related procedures, see Setting Up the Infoblox AWS API Proxy and Using Amazon Boto as an AWS CLI API Client.

You can use any of the following as an API Proxy for your Infoblox vNIOS for AWS deployment:

- On-premises Grid Master
- On-premises Grid member
- AWS VPC-based Grid Master (Infoblox vNIOS for AWS instance)
- AWS VPC-based Grid member (Infoblox vNIOS for AWS instance)

In all cases, the instance acting as the API Proxy must have the Cloud Platform license (for a Grid Master, this is the Cloud Network Automation license).

If you already run an Infoblox Grid and want to extend that Grid to your Amazon cloud network, you can deploy one or more Infoblox vNIOS for AWS instances to join your on-premises Grid. You can use it to manage all of your cloud resources as part of IPAM management, with full DNS capabilities. This deployment type is called a hybrid, because it combines the user's public cloud (AWS) and their private network.

Shared-Services VPC Deployments

Amazon Web Services supports peering of virtual private clouds. Users can dedicate VPCs to specific purposes, such as shared services in one VPC, and business workloads in other VPCs. AWS transparently routes network traffic between them and allows separate (but not overlapping) address spaces for each. Figure 1.1 illustrates Infoblox vNIOS for AWS support for multiple VPCs through AWS VPC peering.
Characteristics of this deployment include the following:

- Organizations may use numerous VPCs in their AWS deployments, with business workload instances populating some VPCs and shared services populating a central hub VPC;
- Extension of the on-premises NIOS Grid to the organization’s Amazon VPC;
- One or more Infoblox vNIOS for AWS instances in the AWS VPC, to perform vDiscovery and management of business assets and networks;
- Regular synchronization of data from the Infoblox vNIOS for AWS instance to the on-premises Grid Master;
- Appliances providing the primary DNS and secondary DNS may be located in the on-premises network or in the VPC; DNS setup depends entirely on how the user deployment operates, without restrictions;
- All discovered Cloud objects can be managed objects in NIOS IPAM;
- You can provision the Infoblox vNIOS for AWS instance with the Elastic Scale feature, or manually configure with licenses and Grid settings;

---

Note:

If you set up an Infoblox vNIOS for AWS instance in your Amazon VPC to act as the API Proxy, all API transactions must be routed through the network connection to the VPC.
• Using the Cloud Platform license, the Infoblox vNIOS for AWS instance can operate as the API Proxy to the AWS API;
• Grid members in the on-premises network can manage VMware or OpenStack private clouds while the Infoblox vNIOS for AWS grid member manages the AWS VPC;
• Grid members in the VPC can survive locally if the connection to the Grid Master goes down, and continue to run vDiscovery tasks. It re-synchronizes with the Grid Master when the connection is restored;
• The organization's DNS namespace can extend to all objects in the VPC;
• You do not need to deploy Infoblox vNIOS for AWS instances in every VPC involved in a peering arrangement. The number and placement of Infoblox vNIOS for AWS instances should be based upon expected network object counts and expected volume of DNS query traffic in the peering VPCs;
• All peered VPCs must be fully contained in the same AWS region;
• A Infoblox vNIOS for AWS instance in the Shared Services VPC can run vDiscovery across the peering VPCs. It collects their data as a single Network View in NIOS;
• The Infoblox vNIOS for AWS instance can operate as the AWS API Proxy. It enables API access to all entities in each of the peered VPCs for which it is authoritative;
• Management of the peering VPC networks is performed through the same Amazon regional endpoint, because all VPCs are contained within the same AWS region;
• The CIDR blocks for each peered VPC are not allowed to have overlapping address spaces between one another;
• Changes to any business workflow instance in the peering VPCs are discovered by the Infoblox vNIOS for AWS instance during the next scheduled vDiscovery. Infoblox vNIOS for AWS then re-synchronizes with the Grid Master.

Multiple-VPCs Full-Public-Cloud Deployment

Another deployment type, termed "Full-Public-Cloud," places the Grid Master and all Grid members in the organization's AWS public cloud. In this example, the VPC peering model is used to illustrate a full Grid deployment. Figure 1.2 illustrates Infoblox vNIOS for AWS support for a Full-Public-Cloud with multiple VPCs.

Figure 1.2 Full Public Cloud Deployment Example
Characteristics of this deployment include the following:

- A Grid Master runs in the Shared Services VPC;
- Numerous VPCs can operate in this type of deployment; the main AWS restriction is that all VPCs peering with one another must have direct connections to each other in the same region. The topology is typically a hub-and-spoke, with the Shared Service VPC as the hub;
- User access to the Grid Master and NIOS is performed through the Amazon regional service endpoint;
- Each Infoblox vNIOS for AWS instance can be set to be authoritative for the VPC of which it is a part;
- You do not need to deploy Infoblox vNIOS for AWS instances in every VPC involved in a peering arrangement. The number and placement of Infoblox vNIOS for AWS instances should be based upon expected network object counts and expected volume of DNS query traffic generated by each of the peered VPCs;
- All VPCs are discovered and managed by default in the same Network View.

*Figure 1.3* illustrates Grid management of multiple VPCs across two or more AWS Regions.

Figure 1.3 Routed Shared Service/Peered VPCs

Characteristics of this deployment type include the following:

- A Grid Master runs in the on-premises network, which routes between each of the multiple Shared Services VPCs;
- The on-premise Grid Master or Grid member can act as the AWS API Proxy for each of regional Peered/Shared Service VPC. This is a recommended configuration due to the appliance's proximity to the administrator performing the API calls;
- The NIOS Grid extends across the organization's on-premise sites and the AWS Peered/Shared Service VPCs;
- Each Infoblox vNIOS for AWS instance can be set to be authoritative for the VPC of which it is a part;
- Each Shared Services/Peered VPC instance from each AWS Region is managed through a separate Network View in NIOS;
- You do not need to deploy Infoblox vNIOS for AWS instances in every VPC involved in a peering arrangement. The number and placement of Infoblox vNIOS for AWS instances should be based upon expected network object counts and volume of DNS query traffic generated by each of the peered VPCs;
- All VPCs are discovered and managed by default in the same Network View.
Configuring AWS Access for NIOS Cloud Admins

All AWS API requests include an Access Key ID and are signed with a corresponding Secret Access Key. These authenticate the sender of the request and verify the authenticity of the request message. AWS generates the Access Key ID and Secret Access Key as a key pair, comprising an access key credential for a specific AWS account user in the AWS Identity & Access Management (IAM) service.

As the intermediary recipient of the API requests destined for AWS, NIOS must authenticate the sender of the request and verify the authenticity of the request message. Each Access Key ID and Secret Access Key pair received by the AWS API Proxy must be assigned to a NIOS user, with sufficient privileges given by a NIOS system administrator. You can assign multiple AWS user accounts to a single NIOS Cloud Admin user account, with the required cloud-api-only NIOS group setting.

You do so by importing a simple CSV spreadsheet file with the AWS IAM access key ID/secret access key pairs and some other information for each user, or by adding existing AWS user accounts directly to NIOS through Grid Manager. For information, see Configuring the NIOS Cloud Admin User.

Note
NIOS uses the access key assignments for authorization and accounting. For example, an Amazon user account may not have authorization to create a VPC, but can launch new instances in a VPC. Another example: for a vDiscovery in a VPC, you can assign a specific AWS user account that has read access to all objects to all VPC entities (primarily, subnets and EC2 instances) to the NIOS Cloud Admin account. This level of authorization is possible in NIOS because multiple AWS user accounts with varying IAM privileges can be assigned to the NIOS Cloud admin user.

Assigning AWS User Credentials to the NIOS Cloud Admin Account

Note
In AWS, the access key credentials are used to digitally sign API calls made to AWS services. (Each access key credential has an Access Key ID and a Secret Access Key.) The secret key portion must be secured by the AWS account holder or the IAM user to whom they are assigned. As a best practice, users should rotate their access keys on a regular basis. Refer to the document AWS Security Best Practices by Amazon Web Services (http://aws.amazon.com/whitepapers/aws-security-best-practices/) and the AWS Documentation page IAM Best Practices (http://docs.aws.amazon.com/IAM/latest/UserGuide/IAMBestPractices.html) for more information.

Use the Amazon IAM features set to create an AWS user account. The AWS account needs the access key credential, comprising a key pair with an Access Key ID and a Secret Access Key, which the administrator creates when they create the account. You can obtain the access key pair only once, at the time the new user credential is created by AWS.

The credentials you use will apply directly to the NIOS Cloud Admin account, and by extension to all administrators using the Cloud Admin account to send directives to the AWS API Proxy.

Figure 1.4 Obtaining the Access Key Credential for an Amazon Account
You add these two values to the import CSV spreadsheet for each AWS user that will use the NIOS cloud account. (You can also download the credentials in a simple Excel-or-text-compatible CSV formatted file.) If the intended cloud admin user does not already have a credential, or if they need a replacement owing to not having their existing key pair on record, the administrator may create a new access key credential on AWS and make a record of the credential for use with the NIOS Cloud Admin account.

All API Query requests must be signed to authenticate the requester. By adding the AWS access key ID and secret access key to a NIOS user account mapping, you ensure a trusted connection between NIOS and AWS for all API Proxy operations, for all selected AWS users.

The import spreadsheet must contain six columns in its header, in the following exact syntax for each cell:

<table>
<thead>
<tr>
<th>header-awsuser</th>
<th>access_key_id</th>
<th>account_id</th>
<th>secret_access_key</th>
<th>nios_user_name</th>
<th>user_name</th>
</tr>
</thead>
</table>

A single record of import data reads as follows:

```
header-awsuser,access_key_id,account_id,secret_access_key,nios_user_name,user_name
awsuser,AKIAI2XEVK73NX3C4DA,337228174961,whMEGK2a2oGu9UhoABBV3lWXRPPP9LJu8t9s,clo,asmit
```

A CSV file may be edited in Excel, but you should use a text editor to verify correct formatting, and to ensure that long Account ID numeric values are not truncated to scientific notation.

**About Tenants**

You include the tenant's account ID value (account_id) in your CSV file for assigning AWS access key pairs to the Infoblox cloud account. NIOS automatically populates the tenant value as the tenant ID (a twelve-digit Amazon account ID value) unless the tenant ID is specified by the user. The tenant ID is a mandatory field in many Infoblox Web API (WAPI) requests. (You can change the tenant name at a later time.) To see tenant examples:

1. In Grid Manager, from the **Cloud** tab, select the **Tenants** tab. The **Name** and **ID** columns show the Tenant ID values.
2. Click the **Name** value for a tenant to view the **Networks** and **VMs** pages for the selected tenant.

**Configuring the NIOS Cloud Admin User**

You continue assignment of AWS users to the NIOS Cloud account by ensuring that the Cloud administrator exists in NIOS. You can either add AWS users directly to NIOS or import the CSV file containing the AWS user information.

To create the NIOS Cloud Admin account for mapping, do the following (if you have already defined a Cloud Admin, you can skip Steps 1–5 of this procedure):

1. In Grid Manager, from the **Administration** tab, select the **Administrators** tab -> **Admins** tab.
2. Expand the **Toolbar** and then click the Add icon.
3. In the **AddAdministratorWizard**, complete the following:
• **Login**: Enter the name for the new cloud administrator account. For example, you can create the name `awscloud` or simply `cloud` as the global user account for AWS.

• **Password**: Enter the local NIOS password for the account. If you want to include special characters in your password, ensure that you put the password in quotes (""") to avoid login issues. Example: "Infoblox".

• **ConfirmPassword**: Enter the same password to confirm.

4. Optionally, click **Next** to add or delete extensible attributes for this cloud admin account. For information about Extensible Attributes, see the Infoblox NIOS Documentation.

5. Save the configuration.

6. From the **Administration** tab, select the **Cloud** tab. The CMPAdministration page appears.

7. Expand the **Toolbar** and then click the **CSV** Import button. The NewCSVImportWizard appears.

8. In the first Wizard step, select **Add** and then click **Next**.

9. Click the **Choose** button to select the CSV file with your Amazon account information. In the file requester, select the file and then click **Open**. The selected file appears in the **Import type**: **Add** field.

10. Click **Next** to continue in the wizard. The **Preview File** page shows the result of the data import.

11. Click **Import**. The AWS user data is imported into the NIOS cloud admin list. The chosen AWS users now have access to the Infoblox AWS API Proxy.

---

**Setting Administrative Permissions for Infoblox vNIOS for AWS**

For operation with the AWS API Proxy, your NIOS Cloud Admin account must have read-write permissions for the following NIOS feature sets:

- IPAM permissions
- DNS Permissions
- Cloud permissions

The Cloud Admin account is assigned to the **cloud-api-only** administrative group in Grid Manager, as previously described in Assigning AWS User Credentials to the NIOS Cloud Admin Account. These permissions allow you to create all the important object types through the API Proxy in the AWS environment. You assign these permissions to the entire **cloud-api-only** administrative group in the Grid Manager.

1. From the **Administration** tab, select the **Administrators** tab -> **Permissions** tab and then select the **cloud-api-only** group in the Groups table, expand the **Toolbar** and then click **Add** -> **Global Permissions**.

2. In the Manage Global Permissions editor, from the **Group Permission** drop-down menu, ensure that **cloud-api-only** is already selected.

3. In the **Permission Type** drop-down menu, choose **IPAM Permissions**, and then select the **Read/Write** check boxes for the following: **All Network Views**, **All IPv4 Networks**, **All Hosts**, and **All IPv4 Host Addresses**.

4. Save the configuration.

5. Select the **cloud-api-only** group and then click **Add** -> **Global Permissions**.

6. In the Manage Global Permissions editor, from the **Permission Type** drop-down menu, choose **Cloud Permissions**.

   - Select the **Read/Write** check box for **All Tenants**.

7. Save the configuration.

8. Select the **cloud-api-only** group and then click **Add** -> **Global Permissions**.

9. From the **Permission Type** drop-down menu, choose **DNS Permissions**, and then select the following **Read/Write** check boxes for these categories: **Grid DNS Properties**, **All DNAME Records**, **All Alias Records**, **All DNS Views**, **All
NAPTR Records, All DNS Zones, All MX Records, All Hosts, All PTR Records, All IPv4 Host Addresses, All SRV Records, All A Records, All TXT Records, and All CNAME Records.

10. Save the configuration.

Grid Manager lists the entire set of updated cloud-api-only group permissions in the Permissions page.

Provisioning vNIOS for AWS Using the Paid NIOS Model

⚠️ Note
This section applies only to NIOS 7.2 release. The Paid NIOS model is not available for NIOS releases after 7.2.

You can obtain the Infoblox vNIOS for AWS AMI directly from the AWS Marketplace and use the pay-as-you-go pricing model to launch instances of the vNIOS for AWS.

This section describes the procedure you use to launch and provision Infoblox vNIOS for AWS using the Paid NIOS licensing model. It provides the complete sequence of procedures you perform to provision a new Infoblox vNIOS for AWS instance in AWS.

When you use the Paid NIOS model, the vNIOS for AWS comes pre-installed with the following permanent licenses: vNIOS, Enterprise, DNS, and CNA (Cloud Network Automation). As long as the virtual appliance is up and running, you can use the NIOS features that these licenses provide. Note that the CNA license is active only when the virtual appliance is configured as the Grid Master; the license has no effect on Grid members. Paid NIOS uses a one-price model, in which you pay the same price regardless of whether you configure the virtual appliance as the Grid Master or as a member. For supported vNIOS models, see Infoblox vNIOS for AWS AMI Virtual Appliance Models.

You may also use the BYOL licensing model or Elastic Scaling (dynamic licenses) to provision and configure vNIOS instances in the AWS VPC. For more information about these licensing models, see Provisioning vNIOS for AWS Using the BYOL Model and Provisioning Infoblox vNIOS for AWS using Elastic Scaling.

⚠️ Note
You cannot combine BYOL and Paid NIOS licenses on the same vNIOS member, but you can have a mix of BYOL and Paid NIOS licenses in the same Grid.

To obtain vNIOS for AWS using Paid NIOS:

1. Log in to the AWS Marketplace using your chosen AWS account.
2. Search the AWS Marketplace by entering "infoblox." The vNIOS for AWS listing appears in the search results.
3. Select the vNIOS for AWS appliance model you want to provision by clicking the appliance title.
4. Review the product details and click Continue.
5. On the Launch on EC2 page, select 1-click Launch.

When you select 1-click Launch, you will be installing the vNIOS for AWS with pre-installed licenses, and you will be charged based on the size of the EC2 instance type for the vNIOS model. You can configure the following parameters before you launch the vNIOS for AWS instance:

- **Version:** This field displays the NIOS version that is supported for the virtual appliance. You might or might not be able to select a different NIOS version from the drop-down list.
- **Region:** Expand this parameter and select the region you want to deploy the vNIOS instance. Note that the selected region might affect your monthly charges.
- **EC2 Instance Type:** This parameter displays the EC2 instance type for the vNIOS instance. Review the memory, CPU, and storage to ensure that this vNIOS model fits your business needs.
- **VPC Settings:** Expand this parameter and select the VPC and subnets in which you want to deploy the vNIOS for AWS instance.
• **Security Group**: A security group acts as a firewall that controls the traffic allowed to reach one or more instances. You can select an existing security group for this deployment or create a new one. For more information, see *Defining an AWS Instance Security Group*.

• **Key Pair**: To ensure that no one else has access to the vNIOS for AWS instance, select an EC2 key pair that the vNIOS instance uses when it is launched in the EC2.

If you select Manual Launch, you will receive a confirmation email about your subscription and details about how to launch the vNIOS instance through the EC2 console or API.

6. Review details of your settings and subscription, and then click **Launch with 1-click**.

7. Perform additional tasks for the vNIOS for AWS configuration to ensure that the virtual appliance is functioning properly. For information, see *Additional Configuration for vNIOS for AWS*.

---

### Provisioning vNIOS for AWS Using the BYOL Model

You can obtain the Infoblox vNIOS for AWS AMI by going to the Community AMI page in Amazon Web Services. Use 'NIOS' or 'Infoblox' as the search term to locate the AMI. For information, see *Obtaining the Infoblox vNIOS for AWS AMI*. This section describes the procedure you use to launch and provision an Infoblox vNIOS for AWS instance for your AWS VPC in the AWS console. This procedure supports users who want to provision Infoblox vNIOS for AWS using the BYOL (Bring Your Own Licensing) licensing model. It provides the complete sequence of procedures you perform to manually provision a new Infoblox vNIOS for AWS instance in AWS.

When you use the BYOL licensing model, you install licenses using the standard methods described in the *Infoblox NIOS Documentation*, including a set of temporary feature licenses. Ensure that you add the following licenses to the appliance: a vNIOS license for your Infoblox vNIOS for AWS instance; a DNS license to run DNS services; a DHCP license to run DHCP services in the Infoblox vNIOS instance deployed on AWS due to AWS restriction, DHCP cannot; the Enterprise (Grid) license to configure it as a Grid Master, a Grid member, or a Grid Master Candidate; and the CNA (Cloud Network Automation) license to manage cloud features on the Grid Master. All other NIOS features are available for use in Infoblox vNIOS for AWS instances and can be enabled by their respective licenses.

You may also use the Paid NIOS model or Elastic Scaling (dynamic licenses) to automatically provision and configure vNIOS instances in the AWS VPC. For more information about these licensing models, see *Provisioning vNIOS for AWS Using the Paid NIOS Model* and *Provisioning Infoblox vNIOS for AWS using Elastic Scaling*.

---

#### Notes

- You cannot combine BYOL and Paid NIOS on the same vNIOS member, but you can have a mix of BYOL and Paid NIOS licenses in the same Grid.
- DHCP services now can run on NIOS instances deployed on AWS to offer instances which are outside AWS. Due to AWS restriction, DHCP cannot be offered for instances running on AWS.

---

### Obtaining the Infoblox vNIOS for AWS AMI

You obtain the Infoblox vNIOS for AWS AMI from the AWS wizard’s Community AMIs page. Installation of the Infoblox vNIOS for AWS AMI involves a series of steps in the AWS console, during which you configure and launch a new Infoblox vNIOS for AWS instance. You may use either the BYOL or Paid NIOS model to establish your Infoblox NIOS features for your deployment of an instance.

To obtain and configure vNIOS for AWS using BYOL:

1. Log in to AWS using your chosen AWS account.
2. In the main AWS Console page, click **EC2**.
3. Click the **Launch Instance** button. The Choose AMI page of the Amazon Launch Instance wizard opens.
4. Click the **Community AMIs** tab.
5. Search for the Infoblox vNIOS for AWS AMI by entering the strings **NIOS** or **Infoblox** in the **Search Community AMIs** box. The Infoblox AMI listing appears in the search results.
6. For the Infoblox vNIOS for AWS AMI, click **Select**.
7. Select the **EC2 Instance Type** based on your requirements. See *Table 1.1* for your available options.
8. Click **Next: Configure Instance Details** to define the networking settings for your new Infoblox vNIOS for AWS instance. For information, see *Defining Network Settings for Your New Infoblox vNIOS for AWS Instance*.

### Defining Network Settings for Your New Infoblox vNIOS for AWS Instance

Infoblox vNIOS virtual appliances require two network interfaces (MGMT and LAN1) for proper Grid communications. These interfaces must be assigned to separate subnets within the same VPC.

Note that the NIOS GUI communicates through the MGMT port. If for any reason you must make changes to the MGMT port, such as swapping NICs or changing the MGMT IP address from static to dynamic, ensure that you use the same IP address for the MGMT port before and after the changes. Otherwise, you might not be able to access the NIOS GUI.

In the AWS wizard's *Configure Instance Details* page, you define the network settings for the new Infoblox vNIOS for AWS instance, including both required network interfaces.

1. Choose your VPC from the **Network** drop-down list.
   - If you have not yet created a VPC, click the **CreateNewVPC** link, and specify the name and the IP address range (in standard CIDR format) for the new VPC. (The address range you specify in this step appears as the top-level network view in the NIOS DataManagement -> IPAM page.)

2. Define the **Subnet** to which the new Infoblox vNIOS for AWS instance is assigned. Each VPC must have a default subnet; you then select this subnetvalue for your configuration:
   - If you have not yet created a subnet for your VPC, click the **Create new subnet** link.
   - In the **VPCDashboard** page, which may open in a new browser window, click **Subnets**.
   - Click **CreateSubnet**. In the VPC list, select the VPC you created in Step 1a, and enter the CIDR Block for the subnet.

**Note**

The CIDR block must be a smaller prefix than the IP address range for the VPC.

- Click **Yes, Create**.
  - You may create more than one subnet. The subnet prefix values appear in the **Subnet** field for each network interface in your AWS console.
  - For the **Auto-assign Public IP** setting, keep the **Disable** default.

Because you are creating an instance with two interfaces, AWS does not allow a Public IP assignment to the new Infoblox vNIOS for AWS instance. AWS displays a warning to this effect when you create the second interface. (You may use an Elastic IP address or a private IP address.)

3. Choose the **IAM role** for the new Infoblox vNIOS for AWS instance. Select your IAM role from the list. You may use default settings for your initial testing. It also may be defined in the AWS console on the **Identity and Access Management** page. Your AWS administrator may not allow custom IAM accounts for your deployment, so this may not be a selectable value.

**Note**

For more information about Amazon IAM, see the Amazon IAM documentation page at [http://docs.aws.amazon.com/IAM/latest/UserGuide/IAM_Introduction.html](http://docs.aws.amazon.com/IAM/latest/UserGuide/IAM_Introduction.html). For information about how Amazon IAM roles and permissions work with your Infoblox vNIOS for AWS instances to ensure secure and accurate authorization of user privileges, see *Credentials for vDiscovery*, and *Assigning AWS User Credentials to the NIOS Cloud Admin Account*. 
4. Keep the default Tenancy setting (Shared tenancy (multi-tenant hardware)). For information about Tenant settings, see About Tenants.

5. Select Network Interfaces -> eth0 and then select the default Subnet from the drop-down list. This subnet should be the same one as the subnet described in Step 1b above. (If a default subnet is in the selected VPC, it automatically appears in this field.)

**Note**
You must use two interfaces for your new Infoblox vNIOS for AWS instance: eth0 and eth1. You create a new eth1 interface for your instance. You use the eth1 interface to join the new Infoblox vNIOS for AWS instance to a NIOS Grid.

6. Click the Add Device button. A new eth1 interface listing appears.

The eth1 interface, automatically designated as such during configuration of the new Infoblox vNIOS for AWS instance, is also labeled as LAN1 in NIOS. You cannot change this setting.

**Note**
For SSH access to the new Infoblox vNIOS for AWS instance, you must always use the IP address associated with the LAN port.

- For eth1, choose the default Subnet from the drop-down list. (For more information on usage of Elastic IP addresses for interfaces in your Infoblox vNIOS for AWS instances, see Using an Elastic IP Address.)

7. Open Advanced Details to configure the User data settings for your new instance.

**Note**
In order to access the NIOS GUI when you start the vNIOS for AWS instance, you must install the vNIOS license. You can do so by setting the value "temp_license:vnios" in the User data settings. You can also use the NIOS CLI to set the temporary or permanent licenses. For more information, see Initializing New Infoblox vNIOS for AWS Instances with the AWS User Data Field.

### Initializing New Infoblox vNIOS for AWS Instances with the AWS User Data Field

You can provision the Infoblox vNIOS for AWS instance through the Advanced Details -> User data field without using Elastic Scaling. In this section, you define the administrator login settings and specify the feature licenses for the new Infoblox vNIOS for AWS instance.

1. In the Advanced Details section, define the following plain-text values in the User data field:
   a. remote_console_enabled: Enables or disables the remote SSH CLI console for a new instance (syntax: y or n).
   b. default_admin_password: Sets the password for the NIOS admin user during the first boot. This value does not have to be a default; it can be the password of any administrator that initializes the new instance. With this method, the password is defined before SSH access to the instance CLI is allowed. The minimum password length is four characters. If an invalid password is passed by this method, it will be ignored and the default "infoblox" password remains in effect for the instance. Note that if you want to include special characters in your password, ensure that you put the password in quotes (") to avoid login issues. Example: 'Infoblox'.
   c. temp_license: Defines the NIOS feature licenses for the new instance. You can list a collection of temporary license names that apply to the instance during initial boot. Using this directive allows you to quickly provision the new instance with temporary licenses without having to open a NIOS CLI session to do the same task. In order to access the NIOS GUI, you must provision the vNIOS license before you
start the vNIOS instance. Infoblox recommends that you also provision the Grid and Cloud licenses at the same time as follows: `temp_license:grid cloud vnios`. All text entries must be in all lower case. Valid license names include the following:

- **TE-** Infoblox vNIOS for AWS instances (TE-820, TE-1420 and TE-2220):
  - vnios
  - grid
  - dns
  - enterprise
  - cloud

- **NIOS license for DDI (TE-V825, TE-V1425 and TE-V2225):**
  - nios IB-Vxxxx
    where "xxxx" is the license number.

- **Cloud Platform Infoblox vNIOS for AWS instances (CP-800, CP-1400 and CP-2200):**
  - vnios
  - grid
  - dns
  - enterprise
  - cloud
  - cloud_api

*Figure 1.5* shows an example.
*Figure 1.5 Defining User Data Settings for Provisioning an Instance without Elastic Scaling*

### Advanced Details

```
#infoblox-config
remote_console_enabled: y
default_admin_password: '#$&$!'
temp_license: cloud vnios dns grid
```

All User Data settings are optional directives that can be included or left out of a configuration. For example, you can include the `remote_console_enabled` and `default_admin_password` declarations to the Elastic Scale configuration in *Figure 1.8*. The `temp_license` command setting does not interfere with or override any dynamic license assignments through Elastic Scaling (for information, see *Provisioning Infoblox vNIOS for AWS using Elastic Scaling*).

**Example:**
```
#infoblox-config
gridmaster:
ip_addr: 172.16.1.2
  remote_console_enabled: y
default_admin_password: '##$&$!'
temp_license: cloud vnios dns grid
```

**Example for adding temp licenses for TE-V825, TE-V1425 and TE-V2225 appliances using AWS User data field:**
```
#infoblox-config
remote_console_enabled: y
default_admin_password: password
```
temp_license: dns enterprise nios IB-V1425

2. Click **Next: Add Storage** to continue with setting up the instance. For information, see *Defining Storage Settings* for your New Instance.

## Defining Storage Settings for your New Instance

You use the *Add Storage* page to define the storage resources to be used by the new instance. Infoblox vNIOS for AWS instances provide a defined amount of instance data storage. The storage size varies according to the AMI you have chosen for your current instance (for information, see *Table 1.1*). You can adjust the amount of instance storage to its maximum value, and attach external storage volumes for an additional cost.

1. In the *Add Storage* page, clear the **Delete on Termination** check box. You use this setting for your Infoblox vNIOS for AWS instances, to de-couple the root partition deletion from the state of the new EC2 instance. This allows retention of the volume for debugging and event log inspection.

   Infoblox recommends keeping at least the minimum storage capacity defaults for the new Infoblox vNIOS for AWS instance.

   ![](Note) Check the top of the AWS console page to see the wizard configuration step location. Click the *Previous* button at any time to navigate to previous configuration pages.

2. Click **Next: Tag Instance** to continue setting up the new Infoblox vNIOS for AWS instance. For information, see *Using AWS Tags with Infoblox Extensible Attributes to Identify Resources for IP Address Assignments*.

## Using AWS Tags with Infoblox Extensible Attributes to Identify Resources for IP Address Assignments

![](Note) AWS Tags that have a matching tag defined in NIOS extensible attributes have the tag value replicated into NIOS.

You use the *Tag Instance* page to define name-value pairs for categorizing, searching and identifying Amazon objects such as EC2 instances, subnets, VPCs, and IP addresses. If you already have extensible attributes defined for your Infoblox Grid, you can add those same EAs to the new Infoblox vNIOS for AWS instance on this page. The tags that you define here apply only to the instance. You can choose to create the tags for the instance at a later time.

You use EAs to tag Infoblox network containers and networks, and to tag corresponding Amazon VPCs and subnets for assigning IP addresses to new resources in the cloud. Without the NIOS EA definitions, the tags defined on the AWS objects will only be meaningful in AWS and you cannot search and match against managed AWS objects in Grid Manager.

1. In the *Tag Instance* page, enter the name for the first **Key**. This key name may match a Cloud EA defined in NIOS, or you can define that EA at a later time in Grid Manager.

2. Enter the **Value** for the new tag.

*Note* For information about Cloud Extensible Attributes, see *Extensible Attributes for Cloud Objects* in the *Infoblox NIOS Documentation*.
3. Click the **Create Tag** button to add a new tag entry to the list. For more information, see *Tagging Existing AWS Objects*.
4. To add more tags to the list, create **Add Another Tag**.
5. When you are finished defining the tags, click **Next: Configure Security Group** to continue setting up the new Infoblox vNIOS for AWS instance. For information, see *Defining an AWS Instance Security Group*.

### Tagging Existing AWS Objects

Tagging existing objects in AWS is straightforward; select a VPC, subnet within a VPC, an EC2 instance or other object type residing in AWS, and then click the **Tags** tab:

*Figure 1.6 Adding Tags to AWS Objects*

You can add tags to your resources to help you organize them. For more information, see *Tagging Your Resources*.

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Remove</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Private subnet</td>
<td></td>
</tr>
<tr>
<td>Availability Zone</td>
<td>us-west-1</td>
<td></td>
</tr>
<tr>
<td>CMP Type</td>
<td>Amazon</td>
<td></td>
</tr>
<tr>
<td>Cloud API Owned</td>
<td>True</td>
<td></td>
</tr>
</tbody>
</table>

In NIOS, you define the extensible attributes for each network in the **Cloud -> Networks** page, or under IPAM within the network view, as shown in *Figure 1.7*.

*Figure 1.7 Defined EAs for Cloud Objects in NIOS*
When you consistently use AWS tags and extensible attributes in your networks, they become more useful and valuable. For example, you use Infoblox API extensions with the EAs that are appropriate for your applications. For information, see *Infoblox Extensions to the AWS API*.

**Defining an AWS Instance Security Group**

**Note**

Configure the AWS Security Group for your instance to only accept traffic for SSH (22) and HTTPS (443) from the specific computers or subnets that are used to manage the Infoblox appliance.

You use the *Configure Security Group* page to define the firewall security settings for your new Infoblox vNIOS for AWS instance. Amazon Web Services enforces a default Deny All policy for all security groups. Your new security group consists of a set of simple firewall rules that specifically allow known IP addresses and network prefixes to access your Infoblox vNIOS for AWS instance and to use specific protocols. These are defined as Inbound rules. You may create a new security group, or add new rules to an existing security group definition provided by your AWS administrator, depending on your AWS IAM privileges.

1. In the *Configure Security Group* page, you define new Inbound rules for your new instance using the following:
   - Permit SSH traffic (TCP/22) from the preferred prefix.
   - Open the port for DNS (UDP/53).
   - Permit secure Web traffic (HTTPS/443) only from a Custom IP prefix representing the network of hosts that access the vNIOS instance for management and configuration.
   - Open two ports for NIOS Grid Joining traffic:
     - UDP/1194.
     - UDP/2114.
   - Open the port for the Infoblox API Proxy (TCP/8787).
You configure a minimum of six rules based on the list above.

Note
You can also add a rule, named 'myip' or similarly, to allow access from your desktop computer to the VPC. Simply select My IP from the Source drop-down list.

Avoid using any prefixes other than those that must access the Infoblox vNIOS for AWS instances in the VPC.

3. Enter the Security group name (AWS uses a simple naming default with the prefix "launch-wizard-...").
4. Enter a Description for the new security group.
5. Click the Type drop-down list for the first rule, and choose SSH.
   - For Source, choose Custom IP and then enter the IPv4 prefix containing the computer hosts that use SSH connections to the new instance. (You may need more than one rule if you have users from multiple networks accessing your instance.)
6. Click Add Rule to create a second rule in the list.
7. Click the Type drop-down list for the second rule, and choose HTTPS.
   - For Source, select Custom IP and then enter the IPv4 prefix containing the computer hosts that connect to Grid Manager for the new Infoblox vNIOS for AWS instance. (You may need more than one rule if you have multiple networks accessing your instance.)
8. When you complete the security group configuration, click Review and Launch. The Review Instance Launch page appears.

Completing Your Infoblox vNIOS for AWS Instance Launch

The Review Instance Launch page lists breakout sections with each category of settings, beginning with AMI Details at the top. The page provides an Edit link for each category (Edit instance type, Edit security groups...) for any final changes.

1. When finished reviewing, click Launch. AWS starts the Key Pair installation. The Key Pair dialog box opens.

Note
You can choose the Proceed without a Key Pair option if you want to perform a simple deployment and then click the I Acknowledge... check box.

The Infoblox standard configuration for Infoblox vNIOS for AWS deployment requires use of a VPN connection or a direct connection to the Amazon VPC(s) on which you are deploying and operating Infoblox vNIOS for AWS instances. This connection does not require an Internet-connected IP address or a secure key pair. All AWS Proxy API operations require use of an assigned and regularly rotated AWS-generated key pair assigned to the cloud-api-only account under Grid Manager. For information, see Assigning AWS User Credentials to the NIOS Cloud Admin Account.

2. Click Review and Launch to launch your new instance. After a brief period of time, the Infoblox vNIOS for AWS instance is active in your VPC.
3. Perform additional tasks for the vNIOS for AWS configuration to ensure that the virtual appliance is functioning properly. For information, see Additional Configuration for vNIOS for AWS.

Provisioning Infoblox vNIOS for AWS using Elastic Scaling

When you provision Infoblox vNIOS for AWS instances using the NIOS Elastic Scaling feature, most procedures are similar to those used when you do not use Elastic Scaling for instance provisioning. The primary difference involves differences in the AWS Configure Instance Details page's **User data** field, including the entry of an Infoblox token and certificate pair, which you generate in the Grid Master and use one time only, when you pre-provision the Infoblox vNIOS for AWS instance.

This section provides a list of cross-references to the sequence of procedures to provision a new Infoblox vNIOS for AWS instance in AWS, and also provides the procedures specific to using Elastic Scaling.

**Note**

Ensure that you have the necessary feature licenses for each of your Infoblox vNIOS for AWS instance, including dynamic licenses to support Elastic Scaling. For information, refer to Managing Licenses in the Infoblox NIOS Documentation.

Initial Infoblox vNIOS for AWS Configuration with Elastic Scaling

To begin provisioning a new Infoblox vNIOS for AWS instance with Elastic Scaling, do the following:

1. Load the Infoblox vNIOS for AWS AMI from the Infoblox Community AMI page. For information, see Obtaining the Infoblox vNIOS for AWS AMI.
2. Click **Next: Configure Instance Details** to define networking settings and User Data configuration for your new Elastic Scaling Infoblox vNIOS for AWS instance.
   - Define the networking settings for your new Infoblox vNIOS for AWS instance, including the required two interfaces. For information, see Defining Network Settings for Your New Infoblox vNIOS for AWS Instance.
   - In the **Configure Instance Details** page, open the **Advanced Details** section, which displays the **User data** settings.
   - Continue to the following section, Defining AWS User Data Settings for Infoblox vNIOS for AWS Instances using Elastic Scaling.

Defining AWS User Data Settings for Infoblox vNIOS for AWS Instances using Elastic Scaling

When you provision using Elastic Scaling, Infoblox vNIOS for AWS instances require different User Data settings on the AWS Configure Instance Details page. In AWS, you enter User Data for new instances in plain text format. You use the following data fields for Elastic Scaling provisioning of new instances:

- The **#infoblox-config** header.
- A required **gridmaster:** field that remains blank (this signifies to AWS that the appliance that is serving the Infoblox vNIOS for AWS instance is a Grid Master). The next three fields are subordinate to this field:
  - **ip_addr**: A required field that specifies the private IP address of the current Infoblox vNIOS for AWS instance.
  - **token**: A field that provides the string for the generated token for the new instance. Here, you paste in the Infoblox NIOS Elastic Scale token for the new Infoblox vNIOS for AWS instance. The token and certificate are generated in the Grid Master when you pre-provision the vNIOS instance.
  - **certificate**: A field that provides the string for the generated Infoblox NIOS certificate credential associated with the token. The token and certificate are generated in the Grid Master when you pre-provision the vNIOS instance. The token and certificate values are valid only for a period of time or until
the new Infoblox vNIOS for AWS instance joins the Grid; at which point the token and certificate expire. For information, see About Elastic Scaling in the Infoblox NIOS Documentation.

Note
You may also use different User Data settings to manually provision new Infoblox vNIOS for AWS instances without using Elastic Scaling. You can use other fields for enabling the remote console and a default admin password. For information, see Initializing New Infoblox vNIOS for AWS Instances with the AWS User Data Field.

After you complete the configuration and launch the new Infoblox vNIOS for AWS instance, the Grid Master and the new instance communicate and the new instance automatically joins the Grid. Further communications with the instance take place through Grid Manager and the NIOS CLI.

In the Configure Instance Details page’s Advanced Details section, do the following:

1. Select the As text option above the entry field.
2. In the User data field, enter or paste in the required data fields, token and certificate information as shown in Figure 1.8. A text example:

```
#infoblox-config
gridmaster:
  ip_addr: 172.16.1.2
token: NyLGqDNK6NVREm3w0ddxyxPS913cm
Certificate: <copy and paste certificate here>
```

![Figure 1.8 Adding the Grid Master, Token and Certificate information to the AWS vNIOS Instance](image)

The token and certificate are used only once, and expire when the Infoblox vNIOS for AWS instance joins the Grid.
3. After entering your information in the Advanced Details section, click **Next: Add Storage** to continue setting up the new Infoblox vNIOS for AWS instance.

### Continuing Infoblox vNIOS for AWS Instance Configuration

After defining the settings for supporting Elastic Scaling, configuring a new Infoblox vNIOS for AWS instance uses the same procedures for configuring an appliance without Elastic Scaling. Do the following to complete instance configuration:

1. **Clicking Next: Add Storage** to define storage capacity settings for the new instance.
   - You use the *Add Storage* AWS wizard page to set storage configuration and define tags to be used as extensible attributes for the new instance. For information, see *Defining Storage Settings for your New Instance*.

   Infoblox recommends keeping at least the minimum storage capacity defaults for the new Infoblox vNIOS for AWS instance.

   ![Note](Note.png)

   Check the top of the AWS console page to see the wizard configuration step location. Click the **Previous** button at any time to navigate to previous configuration pages.

2. Click **Next: Tag Instance** to define AWS tags for matching NIOS extensible attributes.
   - You use the *Tag Instance* AWS wizard page to define name-value pairs for categorizing, searching and identifying Amazon objects such as EC2 instances, subnets, VPCs, and IP addresses. If you already have extensible attributes defined for your Infoblox Grid, you can add those same EAs to the new Infoblox vNIOS for AWS instance on this page. For information, see *Using AWS Tags with Infoblox Extensible Attributes to Identify Resources for IP Address Assignments*.

3. Click **Next: Configure Security Group** to configure the security settings for your new Infoblox vNIOS for AWS instance.
   - You use the *Configure Security Group* AWS wizard page to define the firewall security settings for the new Infoblox vNIOS for AWS instance. For information, see *Defining an AWS Instance Security Group*.

4. Click **Next: Review and Launch** to review instance settings and launch your new Infoblox vNIOS for AWS instance.
   - The *Review Instance Launch* page lists breakout sections with each category of settings, beginning with *AMI Details* at the top. The page provides an *Edit* link for each category (*Edit instance type, Edit security groups…*) for any final changes. Complete your Infoblox vNIOS for AWS instance configuration and launch the instance. For information, see *Completing Your Infoblox vNIOS for AWS Instance Launch*.

5. Perform additional tasks for the vNIOS for AWS configuration to ensure that the virtual appliance is functioning properly. For information, see *Additional Configuration for vNIOS for AWS*.

### Additional Configuration for vNIOS for AWS

After you successfully launch the vNIOS for AWS instance, complete the following tasks to set up your virtual appliance so it is operational. Note that some of these tasks are optional depending on your network configuration and business requirements.

- Configure the DNS settings for your VPC and its resources. For information, see *Setting the DNS Name Server for the Amazon VPC*.
- To test the starting and stopping of the virtual appliance. For information, see *Starting and Stopping the vNIOS Appliance in AWS*. 

---

Copyright ©2020, Infoblox, Inc. All rights reserved.

Rev. B
Perform a DNS Zone transfer (AXFR) to the new Infoblox vNIOS for AWS instance. For information, refer to Enabling Zone Transfers in the Infoblox NIOS Documentation.

Delegation of the required VPC networks and other objects for management by the Infoblox vNIOS for AWS instance. For information, see Delegating NIOS Objects to the Infoblox vNIOS for AWS Grid Member.

If necessary, perform a DNS Zone transfer (AXFR) to the Infoblox vNIOS for AWS instance you designate as the name server in the Amazon VPC. For information, see Enabling Zone Transfers in the Infoblox NIOS Documentation. (This process excludes the use of AWS Route 53.)

Perform vDiscovery of all virtual machines and subnets within the AWS VPC. For information, see Configuring vDiscovery Jobs in the Infoblox NIOS Documentation.

Perform vDiscovery of all virtual machines and subnets within the AWS VPC. For information, see vDiscovery on AWS VPCs.

For required Amazon-specific IAM permission settings to support vDiscovery, see Credentials for vDiscovery.

If you wish to use your Infoblox vNIOS for AWS instance as an AWS API Proxy, see Setting Up the Infoblox AWS API Proxy.

To set up your new Infoblox vNIOS for AWS instance in your Infoblox Grid, refer to Deploying a Grid in the Infoblox NIOS Documentation.

Infoblox recommends that you secure all network connections to your new Amazon deployments to allow traffic only from the specific network addresses involved in your deployment. Your AWS corporate account should use a VPN configuration or direct connection associated with one or more virtual private clouds under the account. You can access the AWS VPN endpoints through your on-premise firewall. Ensure that the correct infrastructure is in place to allow secure communication to your AWS cloud. For examples in the Amazon documentation, refer to http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Scenario3.html.

Setting the DNS Name Server for the Amazon VPC

After configuration and launching of your new instance, you configure it so that your Amazon VPC is within your DNS domain. To do this, you change your virtual private cloud (VPC) DHCP-OPTION-SET configuration in the Amazon console to point to your organization's Infoblox-managed DNS domain. Complete the following:

1. Open the AWS Console and go to the main VPC Dashboard page.
2. Select your Amazon VPC from the list. Ensure that its check box is selected.
3. Click DHCP OptionsSets in the left-pane menu.
4. Click the CreateDHCPOptionsSet button.

Amazon does not allow you to edit existing DHCP Options Sets in the console. You can delete them and create new ones with different settings.

5. Enter the following values as necessary:

   - **Domain name**: This is the top-level DNS domain for the organization. (Required.)
   - **Domain name servers**: The IP addresses of the DNS servers in the organization. (Required.)
   - **NTP Servers**: The Network Time Protocol server for your organization.
   - **NetBIOS name servers** and **NetBIOS Node Type**: For Microsoft support only.

Figure 1.9 shows an example.

Figure 1.9 Setting the VPC's DHCP Options to bind the VPC DNS to the DNS Domain
6. Click Yes, Create.

You should ensure that no other DHCP Option sets conflict with the settings that you define here.

Starting and Stopping the vNIOS Appliance in AWS

For Infoblox vNIOS for AWS instances, Infoblox recommends using the shutdown mechanisms provided in the NIOS UI or in the NIOS CLI. In the NIOS UI, go to Cloud -> Cloud Platform Members, select the Infoblox vNIOS for AWS member, and then click Stop in the Toolbar.

To start and stop Infoblox vNIOS for AWS instances in your AWS VPC, do the following:

1. Go to the top-level Amazon Web Service console page.
2. Select Compute -> EC2.
3. Select Resources -> ( ) Running instances. AWS displays the list of all instances currently running in your VPC.
4. Right-click the Infoblox vNIOS for AWS instance and choose Instance State -> Stop. AWS may take a moment to stop the instance operation. You may also choose from the following:
   - Reboot: Reboots the instance;
   - Terminate: Spins down the Infoblox vNIOS for AWS instance and erases it and its disk contents from the VPC (unless you have selected the **Delete on Termination** feature check box for storage settings, in which case the disk information from the instance remains available for reference in the VPC).

To restart an AWS vNIOS instance, do the following:

1. Go to the top-level AWS console.
2. Select Compute -> EC2.
3. Select Resources -> ( ) RunningInstances.
4. Right-click the stopped vNIOS instance (it shows an orange Stopped icon) and choose InstanceState -> Start.

**Note**
Wait a few minutes after you power on the virtual appliance for the CLI prompt to appear while the appliance initializes.

5. To access Grid Manager on the instance, open a Web browser and enter https://<your_ip_address> as the URL.

Delegating NIOS Objects to the Infoblox vNIOS for AWS Grid Member

You can delegate networks and resources in your Amazon VPC for control by your Infoblox vNIOS for AWS instances. This process is called delegation.

Authority delegation in Cloud Network Automation assigns full and exclusive control of IP addresses and DNS name spaces to a Cloud Platform Appliance, such as an Infoblox vNIOS for AWS instance in an AWS VPC. You can perform authority delegation only through the Grid Master.

When you delegate the authority of IP addresses and DNS name spaces to a Cloud Platform Appliance, the Grid Master loses its authority over the scope of delegation for these IP addresses and name spaces along with any objects within them. That authority is given to the Cloud Platform Appliance.

For a complete discussion of the process of authority delegation, refer to the Infoblox NIOS Documentation.

vDiscovery on AWS VPCs

For the NIOS vDiscovery feature to work on AWS VPCs with the Infoblox vNIOS for AWS instance on public or private subnets, you configure the DNS Resolver setting in the Grid Properties editor in NIOS to add the IP address of the upstream DNS server within AWS. The DNS server must resolve both the user-provided AWS service endpoint and the host name iam.amazonaws.com to the NIOS configuration. You define the setting for the Grid.

To configure DNS resolver for the Grid, complete the following in Grid Manager:

1. From the Grid tab -> Grid Manager tab -> Members tab, expand the Toolbar, and then click Grid Properties.
2. In the Grid Properties editor, do the following:
   - Click the DNS Resolver tab and select the Enable DNS Resolver check box if it is not already selected.
   - In the Name Servers list, click Add to add the IP address of the upstream DNS server to the list.
   - Enter the IP address and press Enter.
3. Save the configuration. The changes may take a brief period of time to become active.

The following figures illustrate AWS cloud-based and on-premises-based appliances communicating with the AWS endpoints to initiate vDiscovery for their VPCs:

*Figure 1.10 Infoblox vNIOS for AWS Appliance Routing to Endpoints for vDiscovery Tasks*
By adding the DNS resolver, communication by the Infoblox vNIOS for AWS appliance to the endpoints is automatic for vDiscovery. Figure 1.11 illustrates the same process for an on-premises NIOS Cloud Platform appliance:

Figure 1.11 On-Premises NIOS Appliance Configured for vDiscovery Tasks
You can also set up a proxy server in your data center so you can perform vDiscovery through the proxy server. For information about how to configure a proxy server on your NIOS virtual appliance, refer to the Infoblox NIOS Documentation.

Note
Network routing may also be required to enable the member to communicate with the AWS endpoints.
Credentials for vDiscovery

When you configure a vDiscovery job through the Infoblox GUI (Grid Manager), you can choose to use **Instance Profile** or **IAM Credential** for authentication.

An instance profile is a container for an IAM role that you use to pass role information to an EC2 instance when the instance is up and running. Select this option if you want to collect information from AWS by waiving a user's credentials and using configuration of a predefined IAM role to get a temporary token that allows API calls. Note that you must first configure the option for "instance profile" in AWS, define an IAM role in the instance profile, and then set permissions for this role before you can select this option in NIOS. Otherwise, this option is disabled. When you select this, you do not need to provide user credentials for vDiscovery jobs.

You can also select IAM credentials if you want to authenticate by using IAM roles to grant secure access to AWS resources from your EC2 instances when they are up and running. When you select this authentication method, you must provide the Access Key ID and Secret Access Key for the AWS endpoint. This is the secret key pair for the administrator account that executes the vDiscovery job.

Before assignment to the NIOS cloud admin account, AWS users need the following AWS IAM permissions to use the vDiscovery feature to discover the resources in their VPCs and manage them through IPAM:

- `iam:GetUser`
- `ec2:DescribeVpcs`
- `ec2:DescribeSubnets`
- `ec2:DescribeRouteTables`
- `ec2:DescribeAddresses`
- `ec2:DescribeNetworkInterfaces`
- `ec2:DescribeInstances`

For more information about how to configure vDiscovery, refer to *Configuring vDiscovery Jobs* in the Infoblox NIOS Documentation.

**Objects Discovered and Collected by vDiscovery**

A number of different network object types are discovered, collected and added to the NIOS database during the vDiscovery process. You may convert some object types to Managed objects in NIOS IPAM.

- Virtual Private Clouds
- Availability Zone
- Tenants
- Subnets
- EC2 Instances (virtual machines);
- IP Addresses

---

**Note**

In AWS, access keys are used to digitally sign API calls made to AWS services. Each access key credential is comprised of an access key ID and a secret key. The secret key portion must be secured by the AWS account holder or the IAM user to whom they are assigned. As a best practice, users should rotate their access keys on a regular basis. Refer to the document *AWS Security Best Practices* by Amazon Web Services ([http://aws.amazon.com/whitepapers/aws-security-best-practices/](http://aws.amazon.com/whitepapers/aws-security-best-practices/)) and the AWS Documentation page IAM Best Practices ([http://docs.aws.amazon.com/IAM/latest/UserGuide/IAMBestPractices.html](http://docs.aws.amazon.com/IAM/latest/UserGuide/IAMBestPractices.html)) for more information.
Creating DNS Records for Discovered IP Addresses

When you configure vDiscovery jobs, you can enable the appliance to automatically create DNS records for discovered virtual instances in your AWS VPCs. When you enable this feature, NIOS automatically adds Host records or A and PTR records to the authoritative zones for the discovered IP addresses based on your configuration. You can also enter a formula that NIOS uses to create the DNS names for the discovered IP addresses based on their VM parameters such as vm_name or discovered_name for data discovered through AWS. By doing so, NIOS is able to discover public and private IP addresses by looking up the corresponding DNS names.

Discovered data includes IP addresses for the VMs and associated information such as VM ID, VM Name, Tenant ID, and others. Note that corresponding zones must already exist in order for NIOS to add DNS records. Otherwise, NIOS does not add any DNS records and it logs a message to the syslog.

NIOS automatically adds DNS records based on the following conditions:

- The corresponding DNS zones must already exist in the NIOS database. NIOS does not automatically create DNS zones for the records.
- To create a PTR records, the corresponding reverse-mapping zone must exist.
- A DNS zone cannot be associated with more than one DNS view. NIOS does not create DNS records for zones that are associated with multiple DNS views.
- NIOS adds new DNS records only if the VM name for the discovered IP address is available and there is no conflict between the discovered data and the associated network view.

The following matrix captures some scenarios about how vDiscovery handles various actions and what the outcome is for the information on the Cloud Platform appliance and in the NIOS database.

**Note**

vDiscovery modifies records that are created by the vDiscovery process only. It does not create or update DNS records that are originally created by other admin users.

<table>
<thead>
<tr>
<th>Actions and Conditions</th>
<th>Cloud Platform Data before vDiscovery</th>
<th>Cloud Platform Data after vDiscovery</th>
<th>NIOS Data before vDiscovery</th>
<th>NIOS Data after vDiscovery</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add new VM (vma) on Cloud Platform appliance</td>
<td>No data for vma</td>
<td>10.10.1 vma.corp1.com</td>
<td>Zone: corp1.com</td>
<td>Zone: corp1.com Host record: vma.corp1.com (10.10.10.1)</td>
</tr>
<tr>
<td>Automatic creation of Host records</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>In NIOS: existing zone corp1.com; no DNS records</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Add new VM (vma) on Cloud Platform appliance                | No data for vma                        | 10.10.10.1 vma.corp1.com            | Zone: corp1.com Host record: vma.corp1.com (10.10.10.1) | Zone: corp1.com Host record: vma.corp1.com (10.10.10.1) |
| Automatic creation of Host records                          |                                       |                                     |                            |                           |
| In NIOS: existing zone corp1.com; existing Host record (originally created by vDiscovery or admin) |                                   |                                     |                            |                           |
### Actions and Conditions

<table>
<thead>
<tr>
<th>Cloud Platform Data before vDiscovery</th>
<th>Cloud Platform Data after vDiscovery</th>
<th>NIOS Data before vDiscovery</th>
<th>NIOS Data after vDiscovery</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add new interface to existing VM (vma) with the same discovered name on Cloud Platform appliance</strong></td>
<td>10.10.10.1 vma.corp1.com</td>
<td>10.10.10.1 vma.corp1.com</td>
<td><strong>Zone:</strong> corp1.com Host record: vma.corp1.com (10.10.10.1) <strong>Zone:</strong> corp1.com Host record: vma.corp1.com (10.10.10.2)</td>
</tr>
<tr>
<td><strong>Automatic creation of Host records</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>In NIOS:</strong> existing zone corp1.com; existing Host record <em>(originally created by vDiscovery)</em></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cloud Platform Data before vDiscovery</th>
<th>Cloud Platform Data after vDiscovery</th>
<th>NIOS Data before vDiscovery</th>
<th>NIOS Data after vDiscovery</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add new interface to existing VM (vma) with the same discovered name on Cloud Platform appliance</strong></td>
<td>10.10.10.1 vma.corp1.com</td>
<td>10.10.10.1 vma.corp1.com</td>
<td><strong>Zone:</strong> corp1.com Host record: vma.corp1.com (10.10.10.1) <strong>Zone:</strong> corp1.com Host record: vma.corp1.com (10.10.10.1)</td>
</tr>
<tr>
<td><strong>Automatic creation of Host records</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>In NIOS:</strong> existing zone corp1.com; existing Host record <em>(originally created by admin)</em></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cloud Platform Data before vDiscovery</th>
<th>Cloud Platform Data after vDiscovery</th>
<th>NIOS Data before vDiscovery</th>
<th>NIOS Data after vDiscovery</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add new interface to existing VM (vma) with different discovered name (vmb) on Cloud Platform appliance</strong></td>
<td>10.10.10.1 vma.corp1.com</td>
<td>10.10.10.1 vma.corp1.com</td>
<td><strong>Zone:</strong> corp1.com Host record: vma.corp1.com (10.10.10.1) <strong>Zone:</strong> corp1.com Host record: vma.corp1.com (10.10.10.1) Host record: vmb.corp1.com (10.10.10.2)</td>
</tr>
<tr>
<td><strong>Automatic creation of Host records</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>In NIOS:</strong> existing zone corp1.com; existing Host record <em>(originally created by admin)</em></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cloud Platform Data before vDiscovery</th>
<th>Cloud Platform Data after vDiscovery</th>
<th>NIOS Data before vDiscovery</th>
<th>NIOS Data after vDiscovery</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Add new interface to existing VM (vma) with different discovered name (vmb) on Cloud Platform appliance</strong></td>
<td>10.10.10.1 vma.corp1.com</td>
<td>10.10.10.1 vma.corp1.com</td>
<td><strong>Zone:</strong> corp1.com Host record: vma.corp1.com (10.10.10.1) <strong>Zone:</strong> corp1.com Host record: vma.corp1.com (10.10.10.1) Host record: vmb.corp1.com (10.10.10.2)</td>
</tr>
<tr>
<td><strong>Automatic creation of Host records</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>In NIOS:</strong> existing zone corp1.com; existing Host record <em>(originally created by admin)</em></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Actions and Conditions

<table>
<thead>
<tr>
<th>Actions and Conditions</th>
<th>Cloud Platform Data before vDiscovery</th>
<th>Cloud Platform Data after vDiscovery</th>
<th>NIOS Data before vDiscovery</th>
<th>NIOS Data after vDiscovery</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Remove existing VM (vma) on Cloud Platform appliance</td>
<td>10.10.10.1</td>
<td>No data for vma</td>
<td>Zone: corp1.com</td>
<td>Zone: corp1.com</td>
</tr>
<tr>
<td>• Automatic creation of Host records</td>
<td>vma.corp1.com</td>
<td></td>
<td>Host record: vma.corp1.com</td>
<td></td>
</tr>
<tr>
<td>• <strong>In NIOS:</strong> existing zone corp1.com; existing Host record (<em>originally created by vDiscovery</em>)</td>
<td></td>
<td></td>
<td>(10.10.10.1)</td>
<td></td>
</tr>
<tr>
<td>• Remove existing VM (vma) on Cloud Platform appliance</td>
<td>10.10.10.1</td>
<td>No data for vma</td>
<td>Zone: corp1.com</td>
<td>Zone: corp1.com</td>
</tr>
<tr>
<td>• Automatic creation of Host records</td>
<td>vma.corp1.com</td>
<td></td>
<td>Host record: vma.corp1.com</td>
<td></td>
</tr>
<tr>
<td>• <strong>In NIOS:</strong> existing zone corp1.com; existing Host record (<em>originally created by admin</em>)</td>
<td></td>
<td></td>
<td>(10.10.10.1)</td>
<td></td>
</tr>
<tr>
<td>• Remove existing interface (10.10.10.2) from VM (vma) with different discovered name (vmb) on Cloud Platform appliance</td>
<td>10.10.10.1</td>
<td>10.10.10.1</td>
<td>Zone: corp1.com</td>
<td>Zone: corp1.com</td>
</tr>
<tr>
<td>• Automatic creation of Host records</td>
<td>vma.corp1.com</td>
<td>vmb.corp1.com</td>
<td>Host record: vmb.corp1.com</td>
<td></td>
</tr>
<tr>
<td>• <strong>In NIOS:</strong> existing zone corp1.com; existing Host record (<em>originally created by vDiscovery</em>)</td>
<td></td>
<td></td>
<td>(10.10.10.2)</td>
<td></td>
</tr>
<tr>
<td>• Remove existing interface (10.10.10.2) from VM (vma) with different discovered name (vmb) on Cloud Platform appliance</td>
<td>10.10.10.1</td>
<td>10.10.10.1</td>
<td>Zone: corp1.com</td>
<td>Zone: corp1.com</td>
</tr>
<tr>
<td>• Automatic creation of Host records</td>
<td>vma.corp1.com</td>
<td>vmb.corp1.com</td>
<td>Host record: vmb.corp1.com</td>
<td></td>
</tr>
<tr>
<td>• <strong>In NIOS:</strong> existing zone corp1.com; existing Host record (<em>originally created by admin</em>)</td>
<td></td>
<td></td>
<td>(10.10.10.2)</td>
<td></td>
</tr>
<tr>
<td>Actions and Conditions</td>
<td>Cloud Platform Data before vDiscovery</td>
<td>Cloud Platform Data after vDiscovery</td>
<td>NIOS Data before vDiscovery</td>
<td>NIOS Data after vDiscovery</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------------------</td>
<td>--------------------------------------</td>
<td>-------------------------------------</td>
<td>-----------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>• Update record name (from vma to vm1) for the existing interface (10.10.10.1) on Cloud Platform appliance</td>
<td>10.10.10.1 vma.corp1.com</td>
<td>10.10.10.1 vm1.corp1.com</td>
<td>Zone: corp1.com Host record: vma.corp1.com (10.10.10.1)</td>
<td>Zone: corp1.com Host record: vm1.corp1.com (10.10.10.1)</td>
</tr>
<tr>
<td>• Automatic creation of Host records</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• In NIOS: existing zone corp1.com; existing Host record (originally created by vDiscovery)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Update record name (from vma to vm1) for the existing interface (10.10.10.1) on Cloud Platform appliance</td>
<td>10.10.10.1 vma.corp1.com</td>
<td>10.10.10.1 vm1.corp1.com</td>
<td>Zone: corp1.com Host record: vma.corp1.com (10.10.10.1)</td>
<td>Zone: corp1.com Host record: vm1.corp1.com (10.10.10.1)</td>
</tr>
<tr>
<td>• Automatic creation of Host records</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• In NIOS: existing zone corp1.com; existing Host record (originally created by admin)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Automatic creation of Host records</td>
<td>10.10.10.1 vma.corp1.com</td>
<td>10.10.10.1 vm1.corp1.com</td>
<td>Zone: corp1.com Host record: vma.corp1.com (10.10.10.1)</td>
<td>Zone: corp1.com Host record: vm1.corp1.com (10.10.10.1)</td>
</tr>
<tr>
<td>• Change FQDN template from ${discover_name} to ${vm_name}</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• In NIOS: existing zone corp1.com; existing Host record (originally created by vDiscovery)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Automatic creation of Host records</td>
<td>10.10.10.1 vma.corp1.com</td>
<td>10.10.10.1 vm1.corp1.com</td>
<td>Zone: corp1.com Host record: vma.corp1.com (10.10.10.1)</td>
<td>Zone: corp1.com Host record: vm1.corp1.com (10.10.10.1)</td>
</tr>
<tr>
<td>• Change FQDN template from ${discover_name} to ${vm_name}</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• In NIOS: existing zone corp1.com; existing Host record (originally created by admin)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Actions and Conditions

<table>
<thead>
<tr>
<th>Cloud Platform Data before vDiscovery</th>
<th>Cloud Platform Data after vDiscovery</th>
<th>NIOS Data before vDiscovery</th>
<th>NIOS Data after vDiscovery</th>
</tr>
</thead>
</table>
| • Change vDiscovery task configuration from creation of Host record to A and PTR records  
  • In NIOS: existing zone corp1.com; existing Host record (originally created by vDiscovery) | 10.10.10.1 vma.corp1.com | 10.10.10.1 vma.corp1.com | Zone: corp1.com  
  Host record: vma.corp1.com  
  (10.10.10.1)  
  Zone: corp1.com  
  A record: vma.corp1.com  
  (10.10.10.1) |
| • Change vDiscovery task configuration from creation of Host record to A and PTR records  
  • In NIOS: existing zone corp1.com; existing Host record (originally created by admin) | 10.10.10.1 vma.corp1.com | 10.10.10.1 vma.corp1.com | Zone: corp1.com  
  Host record: vma.corp1.com  
  (10.10.10.1)  
  A record: vma.corp1.com  
  (10.10.10.1) |

To enable the appliance to automatically create DNS records, complete the following in Grid Manager:

1. For a new vDiscovery job: From the Data Management tab, select the IPAM tab, then select vDiscovery -> New from the Toolbar; or from the Cloud tab, select vDiscovery -> New from the Toolbar.
   or
   To modify an existing job: From the Data Management tab, select the IPAM tab and click vDiscovery -> Discovery Manager from the Toolbar, or from the Cloud tab, select vDiscovery -> Discovery Manager from the Toolbar. In the vDiscovery Job Manager editor, click the Action icon ✽ next to a selected job and select Edit from the menu.

2. In step four of the vDiscovery Job wizard, or in the Data Consolidation tab of the vDiscovery Job Properties editor, complete the following:
   - For every newly discovered IP address, create: Select this check box to enable NIOS to automatically create or update DNS records for discovered VM instances if the records were originally created by vDiscovery.
     - Host: Select this to automatically create Host records for discovered VMs.
     - A & PTR Record: Select this to automatically create A and PTR records for discovered VMs. Note that the DNS zones and reverse-mapping zones to which the records belong must exist in NIOS. Otherwise, vDiscovery does not create the records.
     - The DNS name will be computed from the formula: Enter the formula that NIOS uses to create FQDNs for discovered VMs. You can use the auto-generated FQDNs for DNS resolution purposes. You must use the syntax of $\{parameter\name\}$ for this formula. For AWS, this field supports the vm_name and discovered_name parameters. For example, when you enter $\{vm_name\}.corp100.com$ and the discovered vm_name = XYZ, the DNS name for this IP becomes XYZ.corp100.com. When you enter $\{discover_name\}$ here and the discovered name for the IP is ip-172-31-1-64.us-west-1.compute.internal, the DNS name for this IP is ip-172-31-1-64.us-west-1.compute.internal.

   **Note:**
   If the $\{vm\_name\}$ parameter of an instance contains any special character, the appliance will not be able to identify this instance and will convert it to a managed VM using the vm_id parameter.
Using Infoblox vNIOS for AWS with AWS VPCs, Subnets and IP Addresses

An AWS VPC automatically maps to a network view under NIOS. The main AWS subnet comprising the complete collection of IP addresses for the AWS VPC is automatically mapped to a NIOS network container. No subnets in the network container are allowed to have overlapping IPs or subnets.

Different AWS VPCs can contain overlapping IP address spaces as they are logically isolated from each other in AWS.

Subnet CIDR Guidelines

AWS places limits on how you can structure subnets within a VPC. All subnets within each AWS VPC are in a flat hierarchy, where you cannot define new subnetworks beyond the first level of subnets defined in the VPC. AWS does not provide for a containment hierarchy within subnets.

Figure 1.12 Subdividing VPCs into Subnets in IPAM

In IPAM, any subnet in an AWS address space can be treated as a network container or simply as a network. Figure 1.12 shows two subnets that appear as subordinate network containers to the top-level container, which represents the actual VPC. Though the subnets can appear as containers, no smaller subnets can be created within them. For AWS, functionality as a subnet is exactly the same. Carefully consider the size of the subnets in your AWS virtual private cloud, because you cannot define subnets that are subordinate to those at the top level.

NIOS-AWS Subnet Size Restrictions

AWS allows you to create a subnet in your VPC that uses the same CIDR prefix and mask as for the host VPC, and add new instances to it. For example, consider a VPC prefix 10.0.0.0/16. In AWS, an administrator is allowed to create a subnet with the same prefix 10.0.0.0/16 and to run service instances within it. NIOS does not allow subnets in an AWS VPC that use the same prefix and mask as the VPC network container, and does not discover or recognize resources within that prefix. You will also not be able to create new objects in that subnet. Ensure that all VPCs to be managed or discovered through Grid Manager only use subnets with CIDR mask values that are smaller than the CIDR denoting the VPC. For example, the host VPC has a CIDR mask of /16, and the largest subnet has a mask of /18.
NTP and Hybrid Cloud Synchronization

Ensure that all appliances in a hybrid Grid configuration (for example, an on-premises Grid Master with cloud-based Grid members) are correctly time synchronized. If the Grid Master is not time-correct with all cloud-based Grid members, you will receive AWS authentication errors when attempting to issue directives through the API Proxy.

Using an Elastic IP Address

You may use either an AWS Elastic IP Address or a private IP address, associated with LAN1 on the vNIOS instance, to access the Infoblox vNIOS for AWS instance. (AWS automatically associates the LAN1 port on the vNIOS instance with the `eth1` interface on the VM.) VPN access or an AWS Elastic IP is required for access to Infoblox vNIOS for AWS instances. Using an SSH terminal program such as PuTTY, you connect to the LAN1 IP address for CLI access to the instance.

If your VPC does not use a VPN connection (instead, for example, using the AWS Direct Connect service), you can assign an AWS Elastic IP address to the `eth1` port of your Infoblox vNIOS for AWS instance. You do so through the AWS console. Elastic IP addresses are created and associated with your AWS account and can be allocated and re-allocated to different instances as needed.

NIOS Treatment of AWS Public IP Addresses and Elastic IP Addresses

AWS Public IP addresses originate from Amazon's Public IP address pool. Infoblox vNIOS for AWS instances in the AWS cloud do not manage these public IP addresses, and you do not use NIOS IPAM functionality for them. The Infoblox appliance stores the relationships between private and public IP addresses and enables public IP addresses to be used for creation of DNS records.

Note
For more information about AWS Elastic IP addresses, see the AWS documentation at http://aws.amazon.com/articles/1346. Note that Elastic IP addresses are cost items in AWS, and may not be necessary for operating and accessing your instance.

Note
Infoblox vNIOS for AWS does not support using the API Proxy with Elastic IP addresses.
NIOS stores AWS public IP addresses as Host records with addresses that do not belong to any IPAM network. These are mapped to a DNS zone that is common for multiple NIOS Cloud Platform members. This DNS zone must reside in a non-delegated network view (not managed by any specific Cloud Platform appliance) and the zone uses all associated Cloud Platform members as Grid primaries that are expected to create host records in this zone. NIOS treats Elastic IP addresses in the same way. NIOS stores Elastic IPs as Host records without relationship to any IPAM network, and they are not managed under NIOS IPAM.

![Note]
Infoblox does not recommend opening the API Proxy to the public Internet. If it is necessary to do so, pay close attention to your Security Group settings. For information, see [Defining an AWS Instance Security Group](#).

Common Guidelines for Infoblox vNIOS for AWS Usage

General guidelines for use of Infoblox vNIOS for AWS instances in AWS include the following:

- **Where is your deployment?** You need to know the Amazon Regions, VPCs and Subnets in which you deploy your Infoblox vNIOS for AWS instances, and which of these VPCs and Subnets you delegate to Infoblox vNIOS for AWS instances for management.
- **Who will configure and manage the deployment?** Amazon IAM (Identity and Access Management) configurations in your VPC should be configured correctly to work with Infoblox vNIOS for AWS. Your AWS accounts need the correct IAM permissions for cloud object configuration, discovery, and feature configuration. Infoblox vNIOS for AWS requires use of AWS IAM administrative accounts for full Amazon virtual private cloud integration with the Infoblox Grid. For details, see [Credentials for vDiscovery](#) and [Assigning AWS User Credentials to the NIOS Cloud Admin Account](#).
- **How will you secure your deployment?** Network connections to your Infoblox vNIOS for AWS instances should be further locked down using the AWS Security Groups feature, to allow access only from the specific network addresses involved in your deployment.
- **Where will your Infoblox AWS API Proxy be located?** If you intend to use an Infoblox vNIOS for AWS instance as an AWS API Proxy for management of the Grid in the AWS cloud, ensure that your instance acting as the API Proxy will be reachable through the connection to the Amazon VPC.
- **In large deployments, placement of Infoblox vNIOS for AWS instances can depend on the number of DNS queries and business workloads prevailing in each VPC.** VPCs with relatively small amounts of DNS query traffic, or with services that do not produce heavy traffic flows, can be managed through an Infoblox vNIOS for AWS instance in a peering VPC. The managing instance, however, may have larger memory and storage requirements based on the scale of the networks for which it is expected to be authoritative.
- **Schedule vDiscovery tasks for non-peak time periods when workflow demand is not as heavy.**
Chapter 2 Using the Infoblox vNIOS for AWS API Proxy

This chapter describes how to use the Infoblox AWS API Proxy on Infoblox vNIOS for AWS virtual appliances. Infoblox supports the Amazon Web Services API through Amazon API clients and through NIOS provisioning actions. To enable this capability, you designate one or more NIOS or vNIOS appliances in the Grid as AWS API Proxies. You then configure NIOS so that all AWS API clients automatically send all Amazon API requests to the AWS API proxy. The AWS API Proxy accepts Cloud API requests from an AWS API client on a reserved TCP port. The AWS API proxy parses the Cloud API requests into generic Amazon API requests that can be processed in the Amazon cloud. The AWS API proxy then forwards those requests to Amazon services to read, write and modify cloud network data, including IP addresses. When an Infoblox vNIOS for AWS virtual appliance acts as an AWS API Proxy, the appliance requires delegation to ensure that it can modify the objects specified in the API query. The AWS API Proxy can access objects elsewhere in the Grid for which it is authoritative in the API request.

This chapter includes the following topics:

- Setting Up the Infoblox AWS API Proxy
  - Creating a Custom Zone for the API Proxy
  - Configuring the Proxy API Zone
  - Configuring Cloud API Properties to Support the API Proxy
  - Setting up a Grid Member as the API Proxy
- Infoblox Extensions to the AWS API
  - Setting and Searching Extensible Attributes
  - NIOS-to-AWS Object Correlations
- Using Amazon Boto as an AWS CLI API Client
  - Setting Up an API Client
  - Assigning the AWS Service Endpoints to the AWS API Proxy
- AWS API/Proxy Use Cases
  - Creating New VPC with Specific Tenant and Network View
  - Creating New Instance in EA-Selected Subnet
  - Allocating and Associating an Amazon Elastic IP

Setting Up the Infoblox AWS API Proxy

Note
You can designate the Grid Master (with the Cloud Network Automation license), an on-premises Cloud Platform appliance, or an Infoblox vNIOS for AWS instance (with the Cloud Platform license) as the API Proxy for AWS. Appliances with the Cloud Platform license (on a Grid member) or the Cloud Network Automation license (on the Grid Master) are the only instances eligible to operate as the AWS API Proxy. Infoblox recommends setting up your AWS API Proxy to be in close proximity to the systems that act as clients to communicate with the API Proxy.

You use Grid Manager (the NIOS GUI) to set up an appliance as an AWS API Proxy. The following configurations support AWS API Proxy:

- Grid Master (requires the Cloud Network Automation license);
- Grid Member (requires the Cloud Platform license).
Creating a Custom Zone for the API Proxy

Note This procedure creates a custom DNS zone on the Grid Master. To create the custom DNS zone on a Grid member, see Setting up a Grid Member as the API Proxy.

To use the API Proxy through NIOS, you configure the Grid's Amazon API support by defining a custom DNS zone for your API Proxy. The new zone contains an A record and at least one CNAME alias. An A (address) record is a DNS resource record that maps a domain name (in this case, the API Proxy's FQDN) to an IPv4 address. You then use the CNAME records to assign each AWS service endpoint to the API Proxy's FQDN.

You must define a forward-mapping zone for the API Proxy.

Consider a use case where you want to use the API Proxy to send API directives to two different AWS regional service endpoints:

- ec2.us-west-1.amazonaws.com
- ec2.us-west-2.amazonaws.com

To use the Proxy to issue API calls to more than one AWS service endpoint, you need to use fully-qualified domain names for your API Proxy configuration. You begin by creating a new zone that will be authoritative for all API Proxy operations in your Grid.

1. Log in to Grid Manager, and do the following:
2. From the Data Management tab, select the DNS tab, expand the Toolbar, and then click Add -> Zone -> Add Authoritative Zone.
3. In the Add Authoritative Zone wizard, click Add an authoritative forward-mapping zone and then click Next.
4. Specify the following:
   - **Name:** Enter the domain name for the new zone. Omit the trailing period (" . ") that signifies the root zone. As an example, enter the name "api-proxy" to ensure that the new zone is easily identifiable.
   - **Comment:** Enter a descriptive comment about the zone.
5. Click Next to continue in the wizard:
   - Select Use the set of nameservers, and then click the Add icon and choose Add Grid Primary. If you have only a single appliance, such as the Grid Master or a single Cloud Platform appliance, clicking the Add icon opens the Add Grid Primary panel.
   - In the Add Grid Primary panel, select the name servers for the new zone. If no member is displayed, click Select to specify the Grid member or Grid Master. As the Grid Primary for this zone, you select the appliance acting as the AWS API Proxy. When there are multiple members, Grid Manager displays the MemberSelector dialog box, from which you select the primary name server for the zone.
   - Click Add to add the selected appliance to the list of primary name servers for the zone.

Note For information about selecting a primary DNS server, refer to Specifying a Primary Server in the Infoblox NIOS Documentation.

6. Save the configuration. Click Restart if it appears at the top of the page.
Configuring the Proxy API Zone

You configure a set of DNS records for the new API Proxy zone. For API clients, you create a single A record and at least one CNAME record for the zone. Each CNAME record provides the assignments for each AWS service endpoint to the appliance acting as the AWS API Proxy. Do the following:

1. In the Data Management tab -> DNS tab -> Zones panel, select the new API Proxy zone, expand the Toolbar, and then click Add -> Record -> A Record.
2. In the Add A Record wizard, do the following:
   - Name: Enter the hostname of the appliance or virtual appliance to be applied to the new A record. The displayed zone name should be the API Proxy zone you just created. If no zone name appears, or if you need to specify your new API Proxy zone, click Select Zone. When there are multiple zones, Grid Manager displays the Zone Selector dialog box. Select the API Proxy zone name in the dialog box and then click OK. The name you enter is prefixed to the DNS zone name, and the complete name becomes the FQDN (fully qualified domain name) of the host. For example, if the zone name is api-proxy and you enter gridmaster as the host name, then the FQDN for your A record would be, in this example, gridmaster.api-proxy.
   - In the IP Addresses section, enter the IP address of the appliance or virtual appliance (the Grid Master, in this example) acting as the API Proxy.
   - Comment: Optionally, enter descriptive information about the A record.
3. Save the configuration. Click Restart if it appears at the top of the page. If a message appears requesting "Cannot create a PTR record for <ip_address> because there is no reverse-mapping zone in the system. Only the A record will be created. Do you want to continue?", click Yes. You now create the CNAME records for each of the AWS service endpoints.
4. In the Data Management tab -> DNS tab -> Zones panel, select the new API Proxy zone, expand the Toolbar, and then click Add -> Record -> CNAME Record.
5. In the Add CNAME Record wizard, do the following:
   - Alias: The zone name you selected appears to the right of the Alias field. Enter the alias for the canonical name. For API Proxy, Infoblox recommends making the alias recognizable for the relationship to AWS regional endpoints. For example, enter ec2-us-west-1 as a recognizable alias for the CNAME record that you will associate with the AWS region ec2.us-west-1.amazonaws.com. Click Select Zone to select a DNS zone from the Zone Selector dialog box. If you have only one zone, Grid Manager displays the zone name here when you click Select Zone.
   - Canonical Name: This field displays the domain name of either the current zone or the last selected zone. To add a CNAME record to the new API Proxy zone, enter the complete canonical (or official) name of the host that acts as the API Proxy. Example: gridmaster.api-proxy. This value derives from the A record.
6. Save the configuration. Click Restart if it appears at the top of the page. To add a second CNAME record, do the following:
7. In the Data Management tab -> DNS tab -> Zones panel, select the API Proxy zone, expand the Toolbar, and then click Add -> Record -> Add CNAME Record.
8. In the Add CNAME Record wizard, do the following:
   - Alias: Click Select Zone to select a DNS zone from the Zone Selector dialog box. If you have only one zone, Grid Manager displays the zone name here when you click Select Zone. Then, enter the alias for the canonical name. For API Proxy, Infoblox recommends making the alias recognizable for the relationship to AWS regional endpoints. For example, enter ec2-us-west-2 as a recognizable alias for the CNAME record that you will associate with the AWS region ec2.us-west-2.amazonaws.com.
   - Canonical Name: This field displays the domain name of either the current zone or the last selected zone. To add a CNAME record to the new API Proxy zone, enter the canonical name of the host that acts as the API Proxy. Example: gridmaster.api-proxy.
9. Save the configuration. Click Restart if it appears at the top of the page.

Configuring Cloud API Properties to Support the API Proxy

You complete AWS API Proxy support in NIOS by defining the API endpoint assignments for the API Proxy to the FQDNs for each AWS service endpoint. Do the following:
1. From the **Cloud** tab, click **Grid Cloud API Properties** from the Toolbar. Configuration done for the Grid only applies to the current Grid Master; it is not inherited by other Cloud Platform Appliances.

2. In the **Grid Cloud API Properties** editor, select the **API Proxy** tab, and then complete the following:
   - **Port**: Enter **8787** as the API Proxy port number. You can change the default value, but it must match the value for your AWS VPC configuration.
   - **API Endpoint Mapping**: Enter the complete FQDN for the API Proxy in the **API Proxy FQDN** field. You use the matching CNAME alias from the zone configuration as the prefix for the FQDN, with the hostname for the API Proxy. Example: `ec2-us-west-1.api-proxy`. Enter the complete AWS endpoint FQDN in the **AWS Endpoint FQDN** field. Example: `ec2.us-west-1.amazonaws.com`.
   - Enter any other **API Endpoint Mapping** instances if you want to use the API Proxy with other AWS service endpoints. You must have a matching CNAME alias for each mapping.

3. Select the **Enable the API Proxy for AWS on the Grid Master** check box.

4. Save the configuration.

---

### Setting up a Grid Member as the API Proxy

To perform the API Properties setup for a Cloud Platform Appliance to operate as the AWS API Proxy, ensure that the member has the Cloud Platform license installed and that the Grid Master delegates authority to the member.

Begin by following the steps in the previous procedures:

- **Creating a Custom Zone for the API Proxy**
- **Configuring the Proxy API Zone**

Then, do the following:

1. From the **Cloud** tab, select the **Members** tab -> **member** check box, and then click the Action icon 🍃 and select **Edit** from the menu. Configuration done at the member level applies only to the Grid member.

2. In the **MemberCloudAPIProperties** editor, select the **APIProxy** tab, and then complete the following:
   - **Port**: Enter **8787** as the AWS API Proxy port number. You can change the default value, but it must match the value for your AWS VPC configuration.
   - **APIEndpointMapping**: Enter the complete FQDN for the AWS API Proxy in the **APIProxyFQDN** field. You use the matching CNAME alias from the zone configuration as the prefix for the FQDN, combined with the hostname for the AWS API Proxy. Example: `ec2-us-west-1.api-proxy`. Enter the complete AWS endpoint FQDN in the **AWSEndpointFQDN** field. Example: `ec2.us-west-1.amazonaws.com`.
   - Enter more **APIEndpointMapping** instances if you want to use the AWS API Proxy with other AWS service endpoints. You must have a matching CNAME alias for each mapping.

3. Select the **Enable the API Proxy for AWS on the Member** check box.

4. Save the configuration.

---

### Infoblox Extensions to the AWS API

**AWS API extensions from Infoblox provide extensive support in AWS for both DNS and IPAM functionality in NIOS, by adding enhancements to the standard AWS API parameters.**

**AWS API requests are either GET or POST directives. An AWS query is embedded in the URL or may be part of the request body of a POST request.**
The following table summarizes Infoblox API extension parameters for AWS. The second column indicates which AWS API Actions that each Infoblox extension parameter may be used against.

**Table 2.1 Infoblox Extension Parameters to Amazon API**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>AWS Actions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#X-IB-Tenant-ID={tenant-id}</td>
<td>CreateVpc, CreateSubnet, RunInstances, AllocateAddress,</td>
<td>Specifies the Tenant ID to be used in creating corresponding NIOS objects.</td>
</tr>
<tr>
<td></td>
<td>AssignPrivateIpAddresses, CreateNetworkInterface</td>
<td></td>
</tr>
<tr>
<td>#X-IB-Tenant-Name={tenant-name}</td>
<td>CreateVpc, CreateSubnet, RunInstances, AllocateAddress,</td>
<td>Specifies the Tenant Name for the Tenant ID. Will update the Tenant Name if it's different.</td>
</tr>
<tr>
<td></td>
<td>AssignPrivateIpAddresses, CreateNetworkInterface</td>
<td></td>
</tr>
<tr>
<td>#X-IB-Network-View={view-name}</td>
<td>CreateVpc</td>
<td>Specifies the Network View Name where VPC (Network Container) must be created. If the specified Network View doesn't exist, then returns an error to client.</td>
</tr>
<tr>
<td>#X-IB-SET-EA-{name}={value}</td>
<td>CreateVpc, CreateSubnet, RunInstances, AllocateAddress,</td>
<td>Attaches EA with (name) and (value) to object. Note that EAs may be set with the AllocateAddress action.</td>
</tr>
<tr>
<td></td>
<td>AssignPrivateIpAddresses, CreateNetworkInterface</td>
<td></td>
</tr>
<tr>
<td>#X-IB-EA-EQ-{name}={value}</td>
<td>CreateVpc, CreateSubnet, RunInstances, AllocateAddress,</td>
<td>Matches object with EA (name) that is equal (case insensitive) to (value).</td>
</tr>
<tr>
<td></td>
<td>AssignPrivateIpAddresses, CreateNetworkInterface</td>
<td></td>
</tr>
<tr>
<td>#X-IB-EA-EX-{name}={value}</td>
<td>CreateVpc, CreateSubnet, RunInstances, AllocateAddress,</td>
<td>Matches object with EA (name) that is equal (exact match) to (value).</td>
</tr>
<tr>
<td></td>
<td>AssignPrivateIpAddresses, CreateNetworkInterface</td>
<td></td>
</tr>
<tr>
<td>#X-IB-EA-NE-{name}={value}</td>
<td>CreateVpc, CreateSubnet, RunInstances, AllocateAddress,</td>
<td>Matches object with EA (name) that is not equal to (value)</td>
</tr>
<tr>
<td></td>
<td>AssignPrivateIpAddresses, CreateNetworkInterface</td>
<td></td>
</tr>
<tr>
<td>#X-IB-EA-RE-{name}={value}</td>
<td>CreateVpc, CreateSubnet, RunInstances, AllocateAddress,</td>
<td>Matches object with EA (name) that matches regular expression (value) (unanchored)</td>
</tr>
<tr>
<td></td>
<td>AssignPrivateIpAddresses, CreateNetworkInterface</td>
<td></td>
</tr>
<tr>
<td>#X-IB-EA-LE-{name}={value}</td>
<td>CreateVpc, CreateSubnet, RunInstances, AllocateAddress,</td>
<td>Matches object with EA (name) that is less than or equal to (value)</td>
</tr>
<tr>
<td></td>
<td>AssignPrivateIpAddresses, CreateNetworkInterface</td>
<td></td>
</tr>
<tr>
<td>#X-IB-EA-GE-{name}={value}</td>
<td>CreateVpc, CreateSubnet, RunInstances, AllocateAddress,</td>
<td>Matches object with EA (name) that is greater than or equal to (value)</td>
</tr>
<tr>
<td></td>
<td>AssignPrivateIpAddresses, CreateNetworkInterface</td>
<td></td>
</tr>
</tbody>
</table>
### Table 2.1: Setting and Searching Extensible Attributes

<table>
<thead>
<tr>
<th>Parameter</th>
<th>AWS Actions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#X-IB-Record-Type={'fixedaddress' OR 'ipv4reservation' OR 'Host' OR 'A' OR 'CNAME' OR 'PTR'}</td>
<td>RunInstances, AllocateAddress, AssignPrivateIpAddresses, AssociateAddress, CreateNetworkInterface</td>
<td>Suggests which type of NIOS record should be created. Can have multiple records per API. If both fixed address and ipv4reservation records are supplied, then the last one wins while parsing.</td>
</tr>
<tr>
<td>#X-IB-Host-Name={FQDN}</td>
<td>RunInstances, AllocateAddress, AssignPrivateIpAddresses, AssociateAddress, CreateNetworkInterface</td>
<td>Creates Host Record with FQDN for the object. The dns zone must already exist.</td>
</tr>
<tr>
<td>#X-IB-Host-Aliases={FQDN}</td>
<td>RunInstances, AllocateAddress, AssignPrivateIpAddresses, AssociateAddress, CreateNetworkInterface</td>
<td>Adds Aliases to the Host Record.</td>
</tr>
<tr>
<td>X-IB-Host-Usage={'DNS' OR 'DHCP' OR 'IPAM-Only'}</td>
<td>RunInstances, AllocateAddress, AssignPrivateIpAddresses, AssociateAddress, CreateNetworkInterface</td>
<td>Controls Host Record configuration. DNS will configure Host Record for DNS resolutions, DHCP will configure Host record for DHCP requests, IPAM-Only will disable both DNS and DHCP. Multiple request enhancement allowed; if conflicting, the last one wins.</td>
</tr>
<tr>
<td>#X-IB-A-Name={FQDN}</td>
<td>RunInstances, AllocateAddress, AssignPrivateIpAddresses, AssociateAddress, CreateNetworkInterface</td>
<td>Creates address record with FQDN for the object. The DNS zone must already exist.</td>
</tr>
<tr>
<td>#X-IB-PTR-Name={FQDN}</td>
<td>RunInstances, AllocateAddress, AssignPrivateIpAddresses, AssociateAddress, CreateNetworkInterface</td>
<td>Creates address and PTR records with FQDN for the object. The DNS zone must already exist.</td>
</tr>
<tr>
<td>#X-IB-CNAME-Name={alias}</td>
<td>RunInstances, AllocateAddress, AssignPrivateIpAddresses, AssociateAddress, CreateNetworkInterface</td>
<td>Creates CNAME with alias for the object. This parameter requires that either #X-IB-Host-Record, #X-IB-A-Record or #X-IB-A-PTR-Record also be present to provide an FQDN.</td>
</tr>
<tr>
<td>#X-IB-DNS-View={view-name}</td>
<td>RunInstances, AllocateAddress, AssignPrivateIpAddresses, AssociateAddress, CreateNetworkInterface</td>
<td>Specifies DNS View name in which various DNS records should be created. DNS View must exist in NIOS, if not found, an error is returned.</td>
</tr>
</tbody>
</table>

### Setting and Searching Extensible Attributes

Setting and searching extensible attributes is an important part of the Infoblox enhancements to AWS API parameters. The most important Amazon API workflows, including CreateVpc, CreateSubnet and RunInstances, allow you to set extensible attribute values for the object when it is created in NIOS after the AWS workflow executes. In Table 2.1, you simply use the following request parameter:

#X-IB-Set-EA-{name}={value}

For searching extensible attribute values, use:

#X-IB-EA-{EQ|EC|NE|RE|LE|GE}-{name}={value}

The six search criteria are the following:

- **EQ**: Equal
- **EX**: Equal Case Sensitive (Exact)
- **NE**: Not Equal

---

**Copyright ©2020, Infoblox, Inc. All rights reserved.**

Rev. B
The extensible attributes must be defined and widely used in the NIOS Grid to be meaningful in the AWS context. Extensible attributes that are expected to be used in AWS workflows also must be set as Cloud Extensible Attributes under NIOS. For information, see the Infoblox NIOS Documentation.

A search example, showing a search across two extensible attributes where a match only occurs if both values match:

```
https://ec2.amazonaws.com/?Action=RunInstances
&ImageId=ami-60a54009#X-IB-EA-EX-Department=Engineering#X-IB-EA-EX-Location=California
```

All Extensible Attribute searches use a logical AND construct, in which the positive match occurs only when all specified EAs match.

### Specifying a Network View API Parameter

By default, the AWS API Proxy creates new virtual private clouds (VPC) in the NIOS default network view, which is named **default**. Doing so disallows the creation of overlapping IP address spaces in VPCs, which is supported by AWS. To avoid this, you can use a custom Infoblox API parameter to specify a different NIOS network view for each new virtual private cloud. (The network view must be defined under NIOS before specifying it in AWS API calls.) For the CreateVpc API workflow, an example:

```
resp = conn.create_subnet(vpc_id='vpc-67b36602#X-IB-Network-View=foo',
cidr_block='10.10.10.0/28')
```

### NIOS-to-AWS Object Correlations

*Table 2.2* lists the mappings of AWS API actions to Infoblox NIOS workflows.

*Table 2.2 AWS API Mappings to Infoblox NIOS Workflows*
<table>
<thead>
<tr>
<th>AWS API Action</th>
<th>Input Parameters (Inc. Infoblox)</th>
<th>Input Examples</th>
<th>Effect in NIOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateSubnet</td>
<td>VpcId={vpc-id} &amp;CidrBlock={next-available-network} / {subnet}</td>
<td>VpcId=vpc-6a7b8c9d &amp;CidrBlock=10.1.1.0/26</td>
<td>Create Network with provided subnet within a Network Container identified by VPC ID EA.</td>
</tr>
<tr>
<td></td>
<td>VpcId={vpc-id} &amp;CidrBlock=next-available-network / {subnet}</td>
<td>VpcId=vpc-6a7b8c9d &amp;CidrBlock=next-available-network/26</td>
<td>Creates Network in the Network Container specified by VPC ID EA using get-next-network with provided subnet.</td>
</tr>
<tr>
<td></td>
<td>VpcId=None &amp;CidrBlock=next-available-network / {subnet} #X-IB-EA-EQ-{name} = {value}</td>
<td>VpcId=none &amp;CidrBlock=next-available-network/16 #X-IB-EA-EQ-Site=Bombay #X-IB-EA-EQ-Department=finance</td>
<td>Creates Network with provided subnet mask in a Network Container that matches given set of EAs.</td>
</tr>
<tr>
<td>RunInstances</td>
<td>SubnetId={id} &amp;PrivateIpAddress={ip}</td>
<td>SubnetId=subnet-1a2b3c4d &amp;PrivateIpAddress=172.1.42.42</td>
<td>Creates FIXEDADDRESS record for IP addresses with instance information as EAs.</td>
</tr>
<tr>
<td>AWS API Action</td>
<td>Input Parameters (Inc. Infoblox)</td>
<td>Input Examples</td>
<td>Effect in NIOS</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------------------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>SubnetId={id}</td>
<td>SubnetId=subnet-1a2b3c4d</td>
<td>Allocates next available IP address from the network with provided SubnetId. Creates FIXEDADDRESS record with instance information as EAs.</td>
<td></td>
</tr>
<tr>
<td>SubnetId=None #X-IB-EA-EQ- (name) ={value}</td>
<td>SubnetId=None #X-IB-EA-EQ-Site=London #X-IB-EA-EQ-AvailabilityZone=us-east-1b</td>
<td>Allocates next available IP address from the network with provided EAs. Creates FIXEDADDRESS record with VM-ID provided by Amazon.</td>
<td></td>
</tr>
<tr>
<td>SubnetId={id} &amp;PrivateIpAddress={ip} #X-IB-Host-Record={FQDN}</td>
<td>SubnetId=subnet-1a2b3c4d &amp;PrivateIpAddress=172.16.42.42 #<a href="mailto:X-IB-Host-Record@mail.infoblox.com">X-IB-Host-Record@mail.infoblox.com</a> OR SubnetId=subnet-1a2b3c4d &amp;PrivateIpAddress=172.16.42.42 #<a href="mailto:X-IB-Host-Record@mail.infoblox.com">X-IB-Host-Record@mail.infoblox.com</a> #X-IB-CNAME=webmail.infoblox.com OR SubnetId=subnet-1a2b3c4d &amp;PrivateIpAddress=172.16.42.42 #<a href="mailto:X-IB-Host-Record@mail.infoblox.com">X-IB-Host-Record@mail.infoblox.com</a> #X-IB-DNS-View=Internal</td>
<td>Creates Host Record (and/or other various DNS records) using the FQDN supplied in request. Creates a FIXEDADDRESS record with VM-ID provided by Amazon.</td>
<td></td>
</tr>
<tr>
<td>SubnetId={id} #X-IB-Host-Record={FQDN}</td>
<td>SubnetId=subnet-1a2b3c4d #<a href="mailto:X-IB-Host-Record@mail.infoblox.com">X-IB-Host-Record@mail.infoblox.com</a> OR SubnetId=subnet-1a2b3c4d #<a href="mailto:X-IB-Host-Record@mail.infoblox.com">X-IB-Host-Record@mail.infoblox.com</a> #X-IB-DNS-View=Internal</td>
<td>Allocates next available IP address from the network with provided SubnetId. Creates Host Record using the FQDN supplied in request. Creates FIXEDADDRESS record with VM-ID provided by Amazon. Note that only one instance can be created with this request (MaxCount=1).</td>
<td></td>
</tr>
<tr>
<td>AWS API Action</td>
<td>Input Parameters (Inc. Infoblox)</td>
<td>Input Examples</td>
<td>Effect in NIOS</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RunInstances (cont.)</td>
<td>SubnetId=None #X-IB-EA-EQ-</td>
<td>SubnetId=None #X-IB-EA-EQ-Site=London #X-IB-EA-EQ-AvailabilityZone=us-east-1b #<a href="mailto:X-IB-Host-Record@mail.infoblox.com">X-IB-Host-Record@mail.infoblox.com</a> OR Subnet-Id=Non #X-IB-EA-EQ-Site=London #X-IB-EA-EQ-AvailabilityZone=us-east-1b #<a href="mailto:X-IB-Host-Record@mail.infoblox.com">X-IB-Host-Record@mail.infoblox.com</a> #X-IB-DNS-View=Internal</td>
<td>Allocates next available IP address from the network with provided EAs. Creates Host Record using the FQDN supplied by client. Creates a FIXEDADDRESS record with VM-ID provided by Amazon. Note that only one Instance can be created with this request (MaxCount=1).</td>
</tr>
<tr>
<td></td>
<td>(name)={value} #X-IB-Host-Record= {FQDN}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AllocateAddress</td>
<td>Domain=vpc</td>
<td>Domain=vpc</td>
<td>In this case, FQDN is not provided under any Infoblox request enhancement, so we make this a pass-thru request.</td>
</tr>
<tr>
<td>AssignPrivateIPAddresses</td>
<td>NetworkInterfaceId={id} &amp;PrivateIpAddress. {N}={ip}</td>
<td>NetworkInterfaceId=eni-d83388b1 &amp;PrivateIpAddress. 1=10.0.2.1 &amp;PrivateIpAddress. 2=10.0.2.11</td>
<td>Creates FIXEDADDRESS record with given IP address for the interface (potential-ly an instance) provided in request.</td>
</tr>
<tr>
<td></td>
<td>NetworkInterfaceId=eni-d83388b1 &amp;SecondaryPrivateIpAddress.Count={2}</td>
<td>NetworkInterfaceId=eni-d83388b1 &amp;SecondaryPrivateIpAddress.Count={2}</td>
<td>Obtains IP address from the Subnet of primary IP address of the interface provided in the request. Creates FIXEDADDRESS record with given IP address for the interface (potential-ly an instance) provided in the request.</td>
</tr>
<tr>
<td>AWS API Action</td>
<td>Input Parameters (Inc. Infoblox)</td>
<td>Input Examples</td>
<td>Effect in NIOS</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AssociateAddress</td>
<td>AllocationId={id}</td>
<td>AllocationId=eipalloc-572 3d13e</td>
<td>Updates Host Record of Elastic IP with various EAs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AllocationId=eipalloc-572 3d13e #X-IB-Host-Record={FQDN}</td>
<td>Updates Host Record of Elastic IP with various EAs.</td>
</tr>
<tr>
<td>AttachNetworkInterface</td>
<td>InstanceId={id} &amp;NetworkInterfaceId={id}</td>
<td>InstanceId=i-9cc316fe &amp;NetworkInterfaceId=eni-ffda3197</td>
<td>Updates FIXEDADDRESS record with Attachment ID EA.</td>
</tr>
<tr>
<td>CreateNetworkInterface</td>
<td>SubnetId={id} &amp;PrivateIpAddress={ip}</td>
<td>SubnetId=subnet-b2a249da &amp;PrivateIpAddress=10.0.2.140</td>
<td>Creates FIXEDADDRESS record with given IP address for the interface provided in request.</td>
</tr>
<tr>
<td>CreateNetworkInterface</td>
<td>SubnetId={id}</td>
<td>SubnetId=subnet-b2a249da</td>
<td>Obtains IP address from the Network of SubnetId provided in the request. Creates FIXEDADDRESS with IP address for the interface.</td>
</tr>
</tbody>
</table>

- **NetworkInterfaceId** is the ID of the network interface.
- **SecondaryPrivateIpAddressCount** is the count of secondary private IP addresses.
- **SecondaryPrivateIpAddress** is the secondary private IP address.
- EAs: EA-EQ-{name}={value}
<table>
<thead>
<tr>
<th>AWS API Action</th>
<th>Input Parameters (Inc. Infoblox)</th>
<th>Input Examples</th>
<th>Effect in NIOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>SubnetId=None</td>
<td>&lt;ac:structured-macro ac:name=&quot;uninitialized wiki-markup&quot; ac:schema-version=&quot;1&quot; ac:macro-id=&quot;1fc26dbc-3bfb-4c2c-8e67-a4d3186c1891&quot;&gt;<a href="">ac:plain-text-body</a>&lt;![CDATA[#X-IB-EA={name}={value}[^SecondaryPrivateIPAddressCount=3]]&lt;/ac:plain-text-body&gt;&lt;/ac:structured-macro&gt;</td>
<td>Searches Network using EAs provided in request, Obtains IP address(es) from Network. Creates FIXEDADDRESS with IP address(es) for the interface.</td>
<td></td>
</tr>
<tr>
<td>AWS API Action</td>
<td>Input Parameters (Inc. Infoblox)</td>
<td>Input Examples</td>
<td>Effect in NIOS</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>SubnetId</strong></td>
<td><strong>&amp;PrivatelpAddress=x({ip})</strong> &amp;<strong>#X-IB-Host-Record={FQDN}</strong></td>
<td><strong>SubnetId=subnet-1a2b3c4d &amp;PrivateIpAddress=172.16.42.42</strong> <strong>#<a href="mailto:X-IB-Host-Record@mail.infoblox.com">X-IB-Host-Record@mail.infoblox.com</a> OR SubnetId=subnet-1a2b3c4d</strong> &amp; <strong>PrivateIpAddress=172.16.42.42</strong> <strong>#<a href="mailto:X-IB-Host-Record@mail.infoblox.com">X-IB-Host-Record@mail.infoblox.com</a></strong></td>
<td><strong>Creates Host Record (and/or other various DNS records) using the FQDN supplied in request. Creates a FIXEDADDRESS record with IP address(es) for the interface.</strong></td>
</tr>
<tr>
<td><strong>SubnetId</strong></td>
<td><strong>None</strong> &amp; <strong>#X-IB-EA-EQ-nn={name}</strong> &amp; <strong>#X-IB-Host-Record={FQDN}</strong></td>
<td><strong>SubnetId=None</strong> &amp; <strong>#X-IB-EA-EQ-Site=London</strong> &amp; <strong>#X-IB-EA-EQ-AvailabilityZone=us-east-1b</strong> &amp; <strong>#<a href="mailto:X-IB-Host-Record@mail.infoblox.com">X-IB-Host-Record@mail.infoblox.com</a> OR SubnetId=None</strong> &amp; <strong>#X-IB-EA-EQ-Site=London</strong> &amp; <strong>#X-IB-EA-EQ-AvailabilityZone=us-east-1b</strong> &amp; <strong>#<a href="mailto:X-IB-Host-Record@mail.infoblox.com">X-IB-Host-Record@mail.infoblox.com</a></strong></td>
<td><strong>Allocates next available IP address from the network with provided EAs. Creates Host Record using the FQDN supplied by client. Creates a FIXEDADDRESS record with IP address(es) for the interface.</strong></td>
</tr>
<tr>
<td><strong>CreateTags</strong></td>
<td><strong>ResourceId.1={resource_id}</strong> &amp; <strong>Tag.1.Key={Name}</strong> &amp; <strong>Tag.1.Value={value}</strong></td>
<td><strong>ResourceId.1=vpc-11223344 &amp; Tag.1.Key=Name &amp; Tag.1.Value=TestLab OR ResourceId.1=subnet-11223344 &amp; Tag.1.Key=Site &amp; Tag.1.Value=Chicago OR ResourceId.1=i-11223344 &amp; Tag.1.Key=Name &amp; Tag.1.Value=MyVM1</strong></td>
<td><strong>If name of the ‘Key’ matches with any defined EA in NIOS, update/create EA of the corresponding resource (Network Container, Network or FIXEDADDRESS record) in NIOS</strong></td>
</tr>
<tr>
<td><strong>DeleteNetworkInterface</strong></td>
<td><strong>NetworkInterfaceId={id}</strong></td>
<td><strong>NetworkInterfaceId=eni-11223344</strong></td>
<td><strong>Deletes the Host records and frees addresses associated with the interface</strong></td>
</tr>
<tr>
<td>AWS API Action</td>
<td>Input Parameters (Inc. Infoblox)</td>
<td>Input Examples</td>
<td>Effect in NIOS</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------------</td>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DeleteSubnet</td>
<td>SubnetId={id}</td>
<td>SubnetId=subnet-600D00D</td>
<td>Deletes Network</td>
</tr>
<tr>
<td>DeleteTags</td>
<td>ResourseId.(N)={resource_id} &amp;Tag.(N)...</td>
<td>ResourceId.1=vpc-11223344 &amp;Tag.1.Key=Name</td>
<td>Reset EA value of Network Container or Network or FIXEDADDRESS record if tag Name is a defined EA in NIOS</td>
</tr>
<tr>
<td>DeleteVpc</td>
<td>VpcId={id}</td>
<td>VpcId=vpc-11223344</td>
<td>Deletes Network Container</td>
</tr>
<tr>
<td>DetachNetworkInterface</td>
<td>AttachmentId={id}</td>
<td>AttachmentId=eni-attach-d94b09b0</td>
<td>Updates Host Records with instance specific information</td>
</tr>
<tr>
<td>DisassociateAddress</td>
<td>AssociationId={id}</td>
<td>AssociationId=eipassoc-aa7486c3</td>
<td>Update Host Record of EIP and remove association EAs</td>
</tr>
<tr>
<td></td>
<td>PublicIp=54.43.32.21</td>
<td>PublicIp=54.43.32.21</td>
<td>Passthru - EC2 classic not supported</td>
</tr>
<tr>
<td>ReleaseAddress</td>
<td>AllocationId={id}</td>
<td>AllocationId=eipalloc-5723d13e</td>
<td>Deletes Host Record corre- sponding to the Elastic IP</td>
</tr>
<tr>
<td>StartInstances</td>
<td>N/A</td>
<td>N/A</td>
<td>Triggers vDiscovery to see status of public and private IP associations with the instance</td>
</tr>
<tr>
<td>StopInstances</td>
<td>N/A</td>
<td>N/A</td>
<td>Triggers vDiscovery to see status of public and private IP associations with the instance</td>
</tr>
<tr>
<td>TerminateInstance</td>
<td>InstanceId.(N)={id}</td>
<td>InstanceId.1=i-11223344</td>
<td>Deletes Host Records for all IPs owned by this instance.</td>
</tr>
</tbody>
</table>
Using Amazon Boto as an AWS CLI API Client

For advanced users, Amazon Web Services supports the Boto Python scripting interface for finer-grained control of instances in your Amazon virtual private clouds. You use the Boto Python interface on a separate computer as an AWS API client. Knowledge of Python and Bash scripting and use of Boto are required for effective use of API clients with the AWS API Proxy.

Setting Up an API Client

You begin the Amazon API client configuration by defining the DNS CNAME aliases for the Infoblox vNIOS for AWS instance that will act as the AWS API Proxy. (For this section, we assume use of a vNIOS instance in the VPC as the AWS API Proxy, and a Linux system as the API client.) You define the aliases against the IP address of the vNIOS instance for every region in which the Infoblox vNIOS for AWS instance is intended to operate.

Example:

us-west-blox-gw1.corp100.com
us-east-blox-gw1.corp100.com

You edit the API client computer’s /etc/hosts file to redirect the client’s API requests, that would normally go to the EC2 endpoint, to the IP address of the NIOS appliance of the AWS API Proxy.

Example:

127.0.0.1 localhost localhost.localdomain
::1 localhost localhost.localdomain
...
Note how two aliases point to the same IP address, hence to the same instance hosting the AWS API Proxy. You can have more than one AWS API Proxy in the Grid.

When you want to use an API client to issue requests to the Infoblox AWS API Proxy, you can use the Boto Python interface to Amazon Web Services. Using the following configuration, all directives from this interface go to the AWS API Proxy. Add the following settings to your Boto configuration file, which should be located in your /home directory (assuming Linux as the terminal system):

```
[Boto]
dbg = 1
num_retries = 0
https_validate_certificates = False
endpoints_path = /home/<admin_acct_name>/endpoints.json
logging.basicConfig(filename="boto.log", level=logging.DEBUG)

[Credentials]
aws_access_key_id = <aws-access-key>
aws_secret_access_key = <aws-secret-key>
```

As shown above, the Boto interface refers to an endpoints.json file (it can be named by any name, so long as the script calls the correct file) to refer to the DNS CNAME aliases that are already defined on the API client. This file needs to be located in the `/home/<admin_acct_name>/` directory. This file locally modifies the standard set of AWS regional EC2 endpoints to assign as many as needed to the DNS hostname aliases defined on the API client.

Example:

```
"ec2": {
    "ap-northeast-1": "ec2.ap-northeast-1.amazonaws.com",
    "ap-southeast-1": "ec2.ap-southeast-1.amazonaws.com",
    "ap-southeast-2": "ec2.ap-southeast-2.amazonaws.com",
    "cn-north-1": "ec2.cn-north-1.amazonaws.com.cn",
    "eu-west-1": "ec2.eu-west-1.amazonaws.com",
    "sa-east-1": "ec2.sa-east-1.amazonaws.com",
    "us-east-1": "us-east-blox-gw1.corp100.com",
    "us-west-1": "us-west-blox-gw1.corp100.com",
    "us-west-2": "ec2.us-west-2.amazonaws.com",
    "eu-central-1": "ec2.eu-central-1.amazonaws.com"
}
```

These statements redirect the client’s API requests to the Amazon EC2 regional endpoints, towards the NIOS host acting as the AWS API Proxy.

In the JSON file, API requests refer to the standard Amazon service endpoint values in each record ("us-east-1"). You also enter these standard values as part of assignments to the AWS API Proxy configuration (for information, see Assigning the AWS Service Endpoints to the AWS API Proxy). These definitions assign the AWS API Proxy, with each of its DNS host names, to the endpoints. (These host names are defined on the appliance, as described in Setting the DNS Name Server for the Amazon VPC.)

The following example illustrates how to connect to the AWS EC2 endpoint using the aforementioned Boto setup:

```
vpc_conn = boto.vpc.connect_to_region("us-west-1", port=8787)
```

**Note**
Port 8787 is the default AWS API port.

Continue your setup in the following section, Assigning the AWS Service Endpoints to the AWS API Proxy.
Assigning the AWS Service Endpoints to the AWS API Proxy

You use the values in the endpoints.json file to assign the EC2 service endpoint mappings to the vNIOS host, changing the values from the standard Amazon ones. You add these values to your vNIOS API Proxy configuration. For example, consider the JSON entry:

```
"us-west-1": "us-west-blox-gw1.corp100.com",
```

Instead of using the standard ec2.us-west-1.amazonaws.com value as the endpoint, you can use a DNS alias you configured for the NIOS appliance or Infoblox vNIOS for AWS instance (us-west-blox-gw1.corp100.com), as a substitute for the service endpoint of the particular Amazon region. You change these values for all EC2 service endpoints that you plan to use for Infoblox vNIOS for AWS. This enforces use of the AWS API Proxy in place of the service endpoints.

To perform the setup for the NIOS or vNIOS Cloud Platform Appliance that will operate as the AWS API Proxy, do the following:

1. From the Grid tab, select the GridManager tab, and then click the Cloud-API service.
2. Select the appliance that will run the API Proxy (the member must have the Cloud Network Appliance license) by selecting the Services tab -> cloud_member check box, and then click Edit.
3. In the General tab -> AdministratorsallowowedtomakeWAPIrequestsontheGridMaster section, select Setofadministrators and then click Add -> Local. You will see the complete list of configured Cloud user accounts. You choose the accounts from which API requests are allowed. In a default configuration, only the cloud/local account appears.
4. Click the APIProxy tab -> AmazonWebServices tab, and then click the EnableService check box. Ensure that it is enabled.
5. For the Port, enter the value for the TCP port (typically 8787).
6. For the APIEndpointMapping, click Add.
   - APIProxyFQDN: For the Infoblox API Proxy this value consists of the DNS host name for the Infoblox vNIOS for AWS appliance. Along with the JSON configuration described above, these values enforce the placement of the Cloud Platform appliance as the AWS API Proxy, which receives all AWS API calls issued by API clients.
   - AWSEndpointFQDN: Enter the Amazon regional endpoint to which the API Proxy will send its processed API requests. Example: ec2.us-west-1.amazonaws.com.
7. Save the configuration.

⚠️ Note
The Cloud API service must be restarted in NIOS for configuration changes to take effect. The selected vNIOS member shows Cloud API Service in its Service Status column.

All API Request parameter names are case-insensitive. Infoblox recommends retaining the cases shown for all elements in this document.

AWS API Proxy Use Cases

⚠️ Note
All sample code in this section is written in Python using the Amazon Boto AWS API client library.

Before pre-provisioning and launching Infoblox vNIOS for AWS instances, ensure that you have completed the following:

- Configure the AWS API Proxy based on your deployment, as described in Setting Up the Infoblox AWS API Proxy;
- For CLI operations, configure an Amazon API client system to work with your API Proxy, as described in Setting Up an API Client;
• Obtain and install pool licenses and create license pools on the Grid Master. (For information on license pools and dynamic licensing, see the *Infoblox NIOS Documentation.*) To obtain your pool licenses, consult your Infoblox Sales representative.

The following examples rely on the following two declarations:

```python
vpc_conn = boto.vpc.connect_to_region("eu-west-1", port=8787, validate_certs=False)
e2_conn = boto.ec2.connect_to_region("eu-west-1", port=8787)
```

The first declaration initializes a connection object in Python for the AWS VPC API endpoint. The second declaration initializes a connection object to the AWS EC2 API endpoint.

### Creating New VPC with Specific Tenant and Network View

This example shows the recommended API Client process to set the AWS VPC's DHCP options to specify a Infoblox vNIOS for AWS instance as the DNS server. This ensures that the NIOS Grid is the management for the VPC's DNS domain.

The network view must already be created in NIOS.

You use the `CreateVpc` workflow to define a new virtual private cloud. In the sample python script, you can issue the following sample directives:

```python
## Create a VPC with an explicit CIDR in a specified network view
## Network view MUST exist in NIOS
cidr_and_exts = '10.69.69.0/24#X-IB-Network-View=My-network'
vpc = vpc_conn.create_vpc(cidr_block=cidr_and_exts)
print 'VPC ID: %s, State: %s' % (vpc.id, vpc.state)
```

The following example adds another Infoblox extension to specify the tenant for the new virtual private cloud.

```python
## Create a VPC with CIDR in a network view under a specified Tenant
## Maximum sizes for VPCs under Amazon are /16
cidr_and_exts = '10.60.60.0/24#X-IB-Network-View=My-network#X-IB-Tenant-ID=Tenant-1'
vpc = vpc_conn.create_vpc(cidr_block=cidr_and_exts)
print 'VPC ID: %s, State: %s' % (vpc.id, vpc.state)
```

The `vpc_id` is the value Amazon passes to NIOS.

### Creating New Subnet in VPC with no Extensible Attributes

You use the `CreateSubnet` workflow to create a new Subnet in an Amazon VPC. The following example is relatively straightforward, without use of any Infoblox extensions to search or define EA values:

```python
## Create a Subnet under a VPC with the next available subnet range in the VPC
## THE NEXT-AVAILABLE-NETWORK argument is not native to AWS.
## Without NIOS, this won't work.
```

The next example shows how you can pass an explicit prefix value to create the new subnet.

```python
## or pass explicitly - which AWS DOES support) cird_and_exts - '10.10.1.1/26'
subnet = vpc_conn.create_subnet(vpc_id=vpc.id, cidr_block=cidr_and_exts)
print 'Subnet ID: %s, State: %s' % (subnet.id, subnet.state)
```
Adjust the CIDR and prefix values to suit your requirements.

The subnet.id value is passed to NIOS, and is used for other operations including creating new instances as described in the following section.

Creating New Instance in EA-Selected Subnet

The following script segment calls an Amazon virtual machine shape ID and refers to the AWS subnet for the VPC created in the previous section, Creating New Subnet in VPC with no Extensible Attributes.

You use the *Run Instances* workflow to define a new virtual private cloud, combined with Infoblox extensions built into the API query request.

```python
## Start a t2.micro VM instance in AWS on the subnet
## created above, with the next available IP address on the subnet.
## REQUIRES A VPC AND A SUBNET
reservation = ec2_conn.run_instances(
    "ami-7f0ae93b",
    subnet_id = subnet.id,
    ## THIS subnet.id value is passed from the prior subnet script.
    instance_type="t2.micro")
inst = reservation.instances[0]
print 'Started instance %s with private IP address %s, status: %s %s'
    (inst.id, inst.private_ip_address, inst._state.name)
```

The IP for the new instance is automatically selected by NIOS from the subnet you previously created.

The next usage example includes the following:

- Use of an Extensible Attribute to select the VPC subnet in which to provision the VM;
- Specifying an FQDN for the Host record.

```python
## Start a VM instance in AWS on the subnet created above, with the next available IP
## address on the subnet, using the specific AWS VM shape,
## and create a DNS record with the specified host name.
## The DNS zone corp100.com must be created manually using a subnet search by EA.
reservation = ec2_conn.run_instances(
    "ami-7f0ae93b",
    subnet_id = 'None#X-IB-EA-EQ-Subnet-ID=172.16.0.0'
        '#X-IB-Host-Name=myvm3.corp100.com',
    instance_type="t2.micro")
inst = reservation.instances[0]
print 'Started instance %s with private IP address %s, status: %s %s'
    (inst.id, inst.private_ip_address, inst._state.name)
```

Allocating and Associating an Amazon Elastic IP

This short script segment allocates an Amazon Elastic IP to be used for an instance. The Elastic IP always needs to have a Host name (X-IB-Host-Name) and the Infoblox Network View extension (X-IB-Network-View).

```python
# stringing together two extensions
domain_ext = 'vpc#X-IB-Host-Name=myvm-pub.corp100.com#X-IB-Network-View=My-network'
eip = ec2_conn.allocate_address(domain=domain_ext)
print eip
```

Before associating an Elastic IP, the VPC needs to be attached to an AWS internet gateway. This is a one-time manual step that is done in the AWS console and cannot be performed through the API.

```python
# alloc-id will be logged in /tmp/boto.log for a successful
# allocate_address all. Not shown in the shell. You have to grab it from the log file
# or from the AWS console.
alloc_id = ''
assoc_id = ec2_conn.associate_address(instance_id=inst.id,
    # Created in the previous process
    public_ip=eip, allocation_id=alloc_id,
    private_ip_address=inst.private_ip_address)
```

Afterwards, in the DNS view, you will see the Elastic IP in NIOS.
Chapter 3 Amazon Route 53 Integration

This chapter describes what Amazon Route 53 is and how you can synchronize DNS data from Amazon Route 53 to NIOS to achieve unified DNS data visualization across your on-premise networks and hybrid clouds. This chapter includes the following topics:

- Amazon Route 53 Integration Overview
- Amazon Route 53 Hosted Zones
- Best Practices for Using Infoblox Amazon Route 53 Integration
- Managing Amazon Route 53 Sync Groups
  - Viewing Amazon Route 53 Sync Groups
  - Viewing Sync Task Details
  - Modifying Amazon Route 53 Sync Groups
  - Deleting Amazon Route 53 Sync Groups
  - Viewing Route 53 Hosted Zones in NIOS
  - Viewing Route 53 Resource Records in NIOS

Amazon Route 53 Integration Overview

Amazon Route 53 is a cloud DNS web service designed to route end user requests to Internet applications and resources by resolving domain names into IP addresses and vice versa. It connects user requests to infrastructure that runs in AWS, such as Amazon EC2 instances and load balancers. It can also route users to infrastructure outside of AWS. In Amazon Route 53, you organize DNS records into "hosted zones" that you configure through the Route 53 API. Infoblox NIOS provides the capability to synchronize with Amazon Route 53 and integrate hosted zones with the NIOS database so you can view Route 53 DNS data through a unified console.

There are two types of hosted zones: public and private. For more information, see Amazon Route 53 Hosted Zones. Note that private hosted zones created using Amazon Route 53 cannot resolve resources outside of AWS VPCs nor can it respond to DNS requests outside of the VPCs. If your cloud configuration involves on-premise networks and AWS VPCs in the AWS public cloud, you can address these limitations by integrating DNS data in AWS VPCs with NIOS for a unified DNS data visualization and management. You can also assign a NIOS appliance to serve DNS for imported hosted zones. For more information about Amazon Route 53, refer to the Amazon Route 53 documentation.

The Infoblox Amazon Route 53 integration feature offers the following:

- Synchronization of DNS data from your AWS VPCs to the NIOS database (note that this is a one-way synchronization).
- A unified console (Grid Manager) across your enterprise networks and AWS hosted zones.
- Consolidated DNS and IPAM views for all DNS data through Grid Manager.

Note
To integrate Amazon Route 53 DNS data with NIOS, you must have the Cloud Network Automation license installed on the Grid Master.

Figure 3.1 illustrates how you can utilize the Infoblox Amazon Route 53 integration feature to achieve centralized DNS data visualization. In a Grid that consists of on-premise networks and an AWS public cloud, you define two Grid members to which Route 53 data is synchronized. The DNS data is synchronized from Amazon Route 53, and then transferred from the members to the Grid Master to be stored in the NIOS database. DNS clients (in the enterprise data center) can then query NIOS for the imported Route 53 DNS data. You can also view the imported DNS data through Grid Manager. Note that all synchronization is done at the hosted zone level from Amazon Route 53 to NIOS, NOT vice versa.

Figure 3.1 Amazon Route 53 Integration
Amazon Route 53 Hosted Zones

In Amazon Route 53, there are two types of hosted zones:

- **Public Hosted Zones**: Contain information about routing traffic and resource record sets for domains and subdomains of queries that come from the public Internet and are resolved within the AWS infrastructure.
- **Private Hosted Zones**: Contain information about routing traffic and resource record sets for domains and subdomains of queries that come from instances and resources of any given AWS VPCs and are resolved within one or more AWS VPCs.

The Amazon Route 53 GUI displays details about hosted zones. For each hosted zone, you can view information such as domain name, hosted name type, record set count, name servers (for public hosted zones) and VPCs (for private hosted zones). Note that the name servers (for public hosted zones) to which zone information is deployed are selected randomly by Route 53.
Each hosted zone supports a resource record set that includes records such as A/AAAA, Alias, PTR, NS, SOA, MX, TXT, CNAME, SRV, and SPF. In the Amazon Route 53 GUI, you can view resource record details such as record name, record type, TTL value, record value, and routing policy. Resource records imported from Amazon Route 53 to NIOS are mapped to their corresponding NIOS resource record types, except for SPF records. Amazon Route 53 SPF records are mapped to TXT records in NIOS, and Route 53 aliases are mapped to CNAME records in NIOS.

Note
Hosted zones imported from Amazon Route 53 are managed by Route 53 only. If you add or manipulate any Route 53 data in NIOS, the changes will be overwritten in subsequent synchronization with Amazon Route 53. Adding any NIOS specific supported records (HOST, BULKHOST, DNAME) and NS records would result in synchronization inconsistencies/failures.

Best Practices for Using Infoblox Amazon Route 53 Integration

- Ensure that the time on the NIOS or vNIOS appliance is synchronized with the actual time so that AWS Route 53 synchronization functions properly.
- Depending on the number of hosted zones and resource records, the synchronization of Route 53 data could consume significant amount of memory and database capacity. Therefore, when you configure the Grid member to pull Amazon Route 53 data, ensure that the Grid member has the capacity to handle bulk import of DNS data. Infoblox recommends that you select a member that is not running other services and can handle the synchronization load for this feature.
- To reduce unnecessary data synchronization and to ensure optimal performance in your Grid, use filters to specify the hosted zones that you want to import into the NIOS database when you configure sync tasks. For information about how to configure sync tasks, see Configuring Amazon Route 53 Sync Groups.
- Hosted zones imported from Amazon Route 53 are managed by Route 53 only. If you add or manipulate any Route 53 DNS data in NIOS, the changes will be overwritten in subsequent synchronization with Amazon Route 53. Infoblox recommends that you do not add or modify any Route 53 data through NIOS.
- NIOS does not synchronize Route 53 data to a network view whose authority is delegated to another Grid member. When you configure a sync group, ensure that you select a network view that is not delegated.
- Amazon Route 53 zones and records cannot be synchronized to NIOS database when a zone is signed as they are encrypted.
- Amazon supports multiple values for a resource record set. After data synchronization, NIOS create multiple records (one for each value that is specified in Route 53).
- NIOS does not import NS and SOA records for Route 53 hosted zones. When you configure a primary and secondary name servers to serve Route 53 hosted zones, NIOS creates the NS records and use the default SOA records for these zones.
- SPF records from Route 53 are stored as TXT records in NIOS.
- NIOS does not support the following Route 53 data:
  - Duplicate records (or records using the same name and same record type) in a hosted zone using "non-simple" routing policies
  - Duplicate Route 53 public hosted zones
  - "Tags" in Route 53 (similar to extensible attributes in NIOS)
- During creation of Route 53 record, name with special characters < or & are not allowed.

Configuring Amazon Route 53 Integration

To import DNS zone data from Amazon Route 53 to NIOS, you must first complete a few prerequisites, as described in Prerequisites for Amazon Route 53 Integration. After you complete all the prerequisites, you must then create a sync group in which you add one or more sync tasks. For a sync group, you define the Grid member on which the synchronization occurs, the synchronization interval, the type of hosted zones to import and other configurations. For information about how to configure sync groups, see Configuring Amazon Route 53 Sync Groups.
Prerequisites for Amazon Route 53 Integration

Before you configure sync groups and sync tasks in NIOS, complete the following prerequisites:

1. Ensure that you have installed the Cloud Network Automation license on the Grid Master. For information about licenses, refer to the Infoblox NIOS Documentation.
2. Set up AWS user accounts and record the AWS credentials for these accounts. You may need the credentials when configuring Route 53 sync tasks. For information about how to set up AWS account, see the AWS documentation. You can also configure AWS accounts and credentials through Grid Manager, as described in Configuring AWS Access for NIOS Cloud Admins.

3. Ensure that the time on the NIOS or vNIOS appliance is synchronized with the actual time so that AWS Route 53 synchronization functions properly. You can configure NTP servers on the NIOS appliance and enable the NTP service to synchronize time on the appliance. For information about how to set up the NTP server, refer to the Infoblox NIOS Documentation.
4. Configure DNS resolvers on the Grid member that is synchronizing Route 53 data so the AWS API can reach the Route 53 endpoints. For information about how to configure DNS resolvers, refer to the Infoblox NIOS Documentation.

Configuring Amazon Route 53 on NIOS

To configure Amazon Route 53 integration, complete the following:

1. Create an Amazon Route 53 sync group and add sync tasks to the sync group, as described in Configuring Amazon Route 53 Sync Groups.
2. Optionally, if you want NIOS to serve DNS for the synchronized hosted zones from Amazon Route 53, configure the primary and secondary servers accordingly. For information about how to do that, refer to the Infoblox NIOS Documentation.

After you set up Amazon Route 53 integration, you can do the following:

- View all configured Amazon Route 53 sync groups, as described in Viewing Amazon Route 53 Sync Groups.
- View detailed information about the configured sync groups, as described in Viewing Sync Task Details.
- Modify sync groups and their sync tasks, as described in Configuring Amazon Route 53 Sync Groups.
- View imported DNS data from Route 53, as described in Viewing Route 53 Hosted Zones in NIOS and Viewing Route 53 Resource Records in NIOS.

Note

All sync tasks in the same sync group are performed for the same AWS user account.

Note

The AWS Route 53 job can get stuck during its run due to the following reasons:
- Loss of network connectivity with the member node. To prevent this scenario, ensure that the member node is connected to the network.
- RabbitMQ queue overflow. This can be confirmed with the error message "error:The AMQP connection was closed" found in Administration -> Logs -> Syslog in the NIOS GUI, Grid Manager. To fix this scenario, restart all NIOS services or reboot the NIOS node.
Configuring Amazon Route 53 Sync Groups

You can configure an Amazon Route 53 sync group to include multiple synchronization tasks for different hosted zones in the same Route 53 end point. Before you create a sync group, ensure that you have configured the AWS user accounts (on the NIOS appliance) you want to use for configuring sync tasks. Note that all sync tasks in the same sync group are performed for the same AWS user account. When you disable individual sync tasks, the appliance skips those sync tasks during synchronization with Amazon Route 53. For information about prerequisites, see Prerequisites for Amazon Route 53 Integration.

When you configure a sync group, you can define a network view in which synchronized data resides. You cannot change the network view for the sync group once you save the configuration. If you want to change the network view for subsequent synchronization, create a new sync group. If you want to remove stale DNS data in a specific network view and then remove the data accordingly. You can also use the CSV Import feature to export this data for removal. For information about extensible attributes and CSV Import, refer to the Infoblox NIOS Documentation.

You can also select a specific DNS view so you can synchronize Route 53 zones and records from AWS into NIOS. This way, you can serve all those zones in a consolidated way from NIOS by querying a single Grid member. Depending on which network view you have selected, you may or may not be able to select a specific DNS view for consolidating your Route 53 zones and records. Ensure that you understand the various scenarios about how the appliance handles the consolidated data before you configure the Consolidate zone data into this DNS view option while adding or modifying a Route 53 sync group, as described in Creating Route 53 Sync Groups.

Creating Route 53 Sync Groups

To create a Route 53 sync group and add sync tasks, complete the following:

1. Ensure that you have installed the Cloud Network Automation license on the Grid Master. For information about licenses, refer to the Infoblox NIOS Documentation.
2. Log in to Grid Manager (the Infoblox GUI).
3. From the Grid tab, click the Amazon tab.
4. Expand the Toolbar and then click the Add icon.
5. In the Add Amazon Route 53 Sync Group Wizard, complete the following:
   - **Sync Group Name**: Enter the name of the Amazon Route 53 sync group.
   - **Disable Synchronization**: Select this to disable synchronization for this sync group. This allows you to keep the current configuration, including all sync tasks in the group, and enable them at a later time.
   - **Member**: Click Select to choose the Grid member that will pull DNS data from Amazon Route 53. Infoblox suggests that you select a member that is not running other services and can handle the synchronization load for this feature. If you have only one Grid member in the Grid, the appliance automatically displays the member name here. Select Clear to remove the current member. You can also specify a proxy server to pull data from Amazon Route 53. For information about how to set up a proxy server, refer to the Infoblox NIOS Documentation.
   - **Credentials**: Select the method you want to use to authenticate the connection between the Grid member and AWS for this sync group. You can select one of the following:
     - **Use instance profile**: An instance profile is a container for an IAM role that you use to pass role information to an EC2 instance when the instance is up and running. Select this option if you want to collect information from AWS by waiving a user's credentials and using configuration of a predefined IAM role to get a temporary token that allows cloud API calls. Note that you must first configure the option for "instance profile" in AWS, define an IAM role in the instance profile, and then set permissions for this role before you can select this option. Otherwise, this option is disabled. When you select this, you do not need to provide user credentials.
     - **Use IAM credential**: Select this if you want to authenticate by using IAM roles to grant secure access to AWS resources from your EC2 instances. Click Select to choose the IAM role and use its credentials to access AWS resources from your EC2 instances when they are up and running.

For more information about instance profiles and IAM roles, refer to the AWS documentation.
• **Synchronize Route 53 data into**: Select the network view to which you want the appliance to add synchronized data.
  
  - **This network view**: From the drop-down list, select the NIOS network view to which you want to add synchronized data. The default network view is displayed by default. When you select this option, you can choose to consolidate zone data into a specified DNS view by enabling the **Consolidate zone data into this DNS view** option and selecting a specific DNS view.

  **Note**
  When you synchronize Route 53 data from two or more different AWS endpoints, you must assign each AWS endpoint to a different network view.

  • **The tenant's network view (if it does not exist, create a new one)**: This option is recommended. When you select this option, the synchronized data is saved to the tenant's network view. If the network view does not exist, the appliance creates it (only if a cloud license is installed in the Grid). The appliance uses tenant information to create a new NIOS network view for the synchronized data. For example, AWS tenants by default are associated with the 12-digit user account number (such as 2233441247523), which is the identifier for all objects that are created by that account in AWS. This tenant value becomes the identifier for the new network view as its data is synchronized.

  **Note**
  You cannot modify the network view selection once you save the configuration. Create a new sync group if you want to change the network view. When you remove an old sync task from a sync group, the data remains in the database and you can manually remove the old data by searching for all Route 53 zones that are associated with a particular network view; or you can use CSV import and export the stale data you want to remove from the database.

• **Consolidate zone data into this DNS view**: Depending on which network view you have selected to synchronize Route 53 zone data, you may or may not be able to select a specific DNS view to which the zone data is being synchronized and consolidated. Note that NIOS supports up to 19 VPCs per zone. Consider the following scenarios before selecting or deselecting this option:

  - If you have selected a NIOS network view to add synchronized DNS data, you can select a specific DNS view to which you add the synchronized Route 53 zone data. When you select this option, all zone data will be synchronized into the selected DNS view. If there are duplicate zones, the appliance places them in an order based on their VPC names and adds the first duplicate zone to the corresponding DNS view (depending on your configuration). It then creates new DNS views for subsequent zones that have the same zone name. For example, if your DNS view is "corp100view", the first duplicate zone is added to "corp100view", the second duplicate zone to "corp100view_1", and so on until all duplicate zones are added to their corresponding DNS views.

    If you choose to synchronize Route 53 data into a NIOS network view but you do not select this option, you are not allowed to select a specific DNS view and the appliance synchronizes all private zones into a newly created DNS view using the name "private\%", where \% stands for the key of the DNS view. A new DNS view is created for each VPC in which the zones reside. On the other hand, all public zones are synchronized into the default DNS view, and all duplicate zones are ignored.

  - If you have selected to add synchronized DNS data to a tenant's network view, you are not allowed to select a specific DNS view for the synchronized data. In this case, the appliance synchronizes all private zones into a newly created DNS view using the name "private\%", where \% stands for the key of the DNS view. A new DNS view is created for each VPC in which the zones reside. On the other hand, all public zones are synchronized into the default DNS view, and all duplicate zones are ignored.

  **Note**
  You must not perform a Route 53 sync on multiple DNS views that reside in the same network view. Performing a Route 53 sync in more than one DNS view deletes the data from the other DNS views in which synchronization has taken place. To prevent this, create multiple network views each having a single DNS view and perform Route 53 sync on each of the DNS views.
• **Comment**: Enter additional information about this sync group. In the Sync Tasks section, do the following:

```
Note
All sync tasks in the same sync group are performed for the same AWS user account. Create a new sync group if you want to synchronize data using another AWS user account.
```

• **Sync Tasks**: Click the Add icon to add a sync task to this group. Grid Manager displays the Add Task Task panel. Complete the following in the panel and then click **Add** to add the task to the **Sync Tasks** table:

  • **Name**: Enter the name of the sync task. Use a name that best represents the task so you can differentiate it from other tasks.
  
  • **Public Hosted Zone**: Select this if you want to synchronize data from the Route 53 public hosted zones. In Amazon Route 53, public hosted zones contain information about routing traffic and resource record sets for domains and sub domains of queries that come from the public Internet and are resolved within the AWS infrastructure.
  
  • **Private Hosted Zone**: Select this if you want to synchronize data from the Route 53 private hosted zones. In Amazon Route 53, private hosted zones contain information about routing traffic and resource record sets for a domain and its sub domains of queries that come from instances and resources of any given AWS VPCs and are resolved within one or more AWS VPCs.
  
  • **Filter**: You can add a filter to select a specific zone or zones for synchronization purposes. To specify multiple zones, use commas to separate the values. You can also use wildcard characters in the filter. For example, you can enter “*abc*, ab?c.com, [a-z].com” in this field.
  
  • **Interval**: Define how often you want the synchronization to happen by entering the time interval and selecting the interval unit from the drop-down list.
  
  • **Disable Synchronization**: Select this to disable synchronization for this specific task. This allows you to keep the current configuration for the task and enable it at a later time.

Click **Add** to save the sync task. Click the Add icon again to add more tasks. Grid Manager displays the following information for each saved task in the Sync Tasks table:

  • **Name**: The sync task name.
  
  • **AWS User**: The AWS credential for this task.
  
  • **Interval**: The synchronization interval.
  
  • **Filter**: The filter you entered for synchronizing data from specified zones.

6. Save the configuration.

Managing Amazon Route 53 Sync Groups

After you add your sync groups, you can view detailed information about them. You can also modify certain properties for an existing sync group or delete one when necessary, as follows:

• View the list of your sync groups, as described in **Viewing Amazon Route 53 Sync Groups**

• View detailed information about a specific sync group, as described in **Viewing Sync Task Details**

• Modify properties of an existing sync group, as described in **Modifying Amazon Route 53 Sync Groups**

• Delete an existing sync group, as described in **Deleting Amazon Route 53 Sync Groups**

• View imported Route 53 hosted zones and resource records in NIOS, as described in **Viewing Route 53 Hosted Zones in NIOS** and **Viewing Route 53 Resource Records in NIOS**.

```
Note
To perform any tasks in NIOS, you must first log in to Grid Manager. For detailed information about tasks that you perform in NIOS, such as how to view zones and resource records, refer to the **Infoblox NIOS Documentation**.
```

Viewing Amazon Route 53 Sync Groups

To view configured Amazon Route 53 sync groups, complete the following:
1. From the Grid tab, click the Amazon tab.
2. Grid Manager displays the following information for each sync group in the Grid:
   - **Name**: The name of the sync group.
   - **Status**: Displays the current overall status of the sync group. This field can display one of the following status:
     - **OFFLINE**: No synchronization has started or no sync task has been configured.
     - **OK**: All synchronization is successful.
     - **ERROR**: One or more sync tasks failed. You can drill down to the sync task level to find out which sync task has failed. You can also check the syslog for more information about the failure.
   - **AWS Credentials**: Displays the Amazon account ID and AWS user name. This field is hidden by default.
   - **Member**: The name of the Grid member on which the synchronization occurs.
   - **Disabled**: Indicates whether the sync group is disabled or not. This displays **Yes** if the sync group is disabled. When you disable a sync group, synchronization for all tasks in the group is disabled but the appliance keeps your configuration intact.

### Note
Route 53 does not support two records having the same Fully Qualified Domain Name (FQDN). If you encounter Uniqueness Violation error for any record, then that particular record will not be getting updated in NIOS from Route 53. You will receive a warning message in **syslog** and **infoblox.log**. However, sync jobs will continue to sync for other zones and records.

You can also do the following in this tab:
- Click the Add icon to add a new sync group.
- Click the Action icon 🕵️ for a specific sync group to perform the following:
  - **Edit**: Modify certain properties for the sync group.
  - **View sync details**: Get details about the sync group in a different dialog box. For more information, see Viewing Sync Task Details.
  - **Delete**: Remove the sync group from the list.
- Create a quick filter to save frequently used filter criteria:
  1. In the filter section, click **Show Filter** and define filter criteria for the quick filter.
  2. Click **Save** and complete the configuration in the **Save Quick Filter** dialog box. The appliance adds the quick filter to the quick filter drop-down list in the panel. Note that global filters are prefixed with [G], local filters with [L], and system filters with [S].
- Use filters and the **Go to** function to narrow down the list. With the autocomplete feature, you can just enter the first few characters of an object name in the **Go to** field and select the object from the possible matches.
- Sort the data in ascending and descending order by column.
- Print or export the data in this tab.

### Viewing Sync Task Details
After you configure a sync group, you can view more details about this group by clicking **View sync details** from the Action menu 🕵️.

To view sync group details:
1. From the Grid tab, click the Amazon tab.
2. Click the Action icon 🕵️ next to the selected sync group, and then select **View sync details** from the menu.
3. Grid Manager displays the following information for the selected sync group:
   - **Sync Group Name**: The name of the sync group.
   - **Name**: The name of the sync task.
• **Status**: The current status of the sync task. You can hover your mouse over the status to display a tooltip that gives you more information about the status.
• **Disabled**: Indicates whether this sync task has been disabled or not.
• **Zone Count**: The total number of synchronized hosted zones.
• **Interval**: Displays how often the synchronization happens for this task.
• **AWS Credentials**: Displays the AWS user account that is used to establish the connection between the Grid member and AWS. Note that all the sync tasks you add to this sync group belong to the same AWS user account. This column is hidden by default.
• **Filter**: Displays the zone or a list of zones that you have configured for this sync task. To specify multiple zones, use comma to separate the values. You can use wildcard characters in the filter. For example, you can enter "*abc*, ab?c.com, [a-z].com" in this field.
• **Last Run**: The timestamp when this task was last run and synchronized with Amazon Route 53.

You can click the Action icon 📚 next to a sync task to perform the following:

• **Run Task Now**: Performs synchronization immediately for the selected task.

## Modifying Amazon Route 53 Sync Groups

To modify an existing sync group, complete the following:

1. From the Grid tab, click the Amazon tab.
2. In the Amazon Route 53 Sync Group table, click the Action icon 📚 next to the selected sync group, and then select Edit from the menu.
3. In the **General** tab of the Amazon Route 53 Sync Group Properties editor, modify the properties as described in Configuring Amazon Route 53 Sync Groups.
4. Save the configuration.

## Deleting Amazon Route 53 Sync Groups

To delete an existing sync group, complete the following:

1. From the Grid tab, click the Amazon tab.
2. In the Amazon Route 53 Sync Group table, select the check box of a sync group, and then click the Delete icon.
3. In the confirmation dialog, click Yes.

## Viewing Route 53 Hosted Zones in NIOS

You can view the imported Route 53 hosted zone data in NIOS, as follows:

1. From the Data Management tab -> DNS tab, click the Zones tab.
2. Grid Manager displays all imported Route 53 zones in this panel. You can filter Route 53 hosted zones by the extensible attribute "DNS Source" = "AWS Route53." Note that the name and comment for each Route 53 zone appear under its corresponding fields (Name and Comments) in NIOS.
3. To view detailed information about a specific Route 53 zone, select the zone check box, and then click the Edit icon.
4. In the Zone editor, select the Route 53 tab. The appliance displays Route 53 zone properties such as VPC, caller reference ID, resource record set count, name server, and delegation set ID. Note that this read-only data is not used for serving DNS.

## Viewing Route 53 Resource Records in NIOS

You can view imported DNS data for each Route 53 resource record, as follows:

1. From the Data Management tab -> DNS tab, click the Zones tab.
2. Click the Route 53 zone link to drill down to the **Records** tab. Grid Manager displays all the associated records for the selected hosted zone in this panel. Note that the name and TTL value for each Route 53 record appear under its corresponding fields (**Name** and **TTL**) in NIOS.

3. Select a record check box and click the Edit icon.

4. In the **Record** editor, select the **Route 53** tab. Depending on the record you select, the appliance displays Route 53 resource record properties such as record type, set ID, weight, region, failover, geolocation continent code, geolocation country code, geolocation subdivision code, alias hosted zone ID, alias evaluate target health, and health check ID. Note that this read-only data is not used for serving DNS.