Infoblox
CONTROL YOUR NETWORK

User Guide
Infoblox IPAM Driver for Docker

Version 1.0.0

Copyright Statements

Trademark Statements

Company Information

http://www.infoblox.com/contact/

Product Information

Hardware Models

Infoblox Advanced Appliances: PT-1400, PT-1405, PT-2200, PT-2205, PT-2205-10GE, PT-4000, and PT-4000-10GE

Network Insight Appliances: ND-800, ND-805, ND-1400, ND-1405, ND-2200, ND-2205, and ND-4000

Trinzic Appliances: TE-100, TE-810, TE-815, TE-820, TE-825, TE-1410, TE-1415, TE-1420, TE-1425, TE-2210, TE-2215, TE-2220, TE-2225, IB-4010, and IB-4020

Cloud Network Automation: CP-V800, CP-V1400, and CP-V2200

Trinzic Reporting: TR-800, TR-805, TR-1400, TR-1405, TR-2200, TR-2205, and TR-4000

DNS Cache Acceleration Appliances: IB-4030 and IB-4030-10GE

NetMRI: NetMRI-1102-A, NT-1400, NT-2200, and NT-4000

Document Number: 400-0693-000 Rev. A

Document Updated: February 3, 2017

Warranty Information

Your purchase includes a 90-day software warranty and a one year limited warranty on the Infoblox appliance, plus an Infoblox Warranty Support Plan and Technical Support. For more information about Infoblox Warranty information, refer to the Infoblox Web site, or contact Infoblox Technical Support.

Contents

Overview

Docker libnetwork is a robust Container Network Model that provides a consistent programming interface and the required network abstractions for applications. Docker can use third-party IPAM solutions through the libnetwork IPAM driver. Infoblox IPAM Driver for Docker is a Docker libnetwork driver that interfaces with the Infoblox DDI product to provide centralized IP address management services.

Organizations typically have multi-container environments including VMs, physical hosts, etc. This leads to the need to manage IP addresses in a holistic way across the organization. In this complex container deployment, Infoblox IPAM Driver for Docker helps maintain consistency in a very dynamic multi-host environment dealing with network container, network creation and deletion, and IP address allocation and release in a network.

With this approach, Infoblox IPAM Driver for Docker provides solutions to the following use cases:

- Allocating IP addresses on a given Docker network for a microservice container.
- Attaching containers on to pre-defined networks or VLANs within your environment.
- Creating a common shared network across multiple hosts of cooperating applications and associated microservices.
- Creating separate networks for different microservices-based applications across multiple hosts that do not need to interact, therefore isolating the traffic between containers.

This Guide explains how to install and use Infoblox IPAM Driver for Docker in sections *Installing Infoblox IPAM Driver for Docker* on page 4 and *Network and IP Address Management for Containers Using Infoblox IPAM Driver for Docker* on page 7.

Before Installing Infoblox IPAM Plugin for Docker

To use the Driver, you need access to the physical or virtual Infoblox DDI product, NIOS or vNIOS. For evaluation purposes, you can download a virtual version of the product from the Infoblox Download Center at *https://www.infoblox.com/infoblox-download-center*. If you are an existing Infoblox customer, you can download it from the Support site. For information about downloading and setting up vNIOS, see *Downloading vNIOS* and *Setting Up vNIOS*.

As another prerequisite, you must have a working installation of Docker. You also need to create a Docker host to interact with vNIOS. For information, see Docker documentation at *https://docs.docker.com/engine/userguide/*.

Downloading vNIOS

vNIOS is the Infoblox virtual appliance that you can download from the Infoblox Download Center.

Do the following:

1. Point your browser to *https://www.infoblox.com/infoblox-download-center*.
2. Navigate to the section **Infoblox DDI (DNS, DHCP, IPAM)**.
3. Click **Try it Now** to download the Infoblox DDI product.
4. When the registration is complete, you will receive an email with the link that takes you to the Product Evaluation Portal. In the Product Evaluation Portal, under the **Required Downloads** section, download Infoblox DDI for VMware. In the Product Evaluation Portal, you can find download links as well as instructional videos to set up vNIOS.

Note: It is strongly recommended that you download the VMware version of the product as VMware is the platform on which the videos are based.

5. After the download is complete, install vNIOS.

See the next section for information on setting up vNIOS.

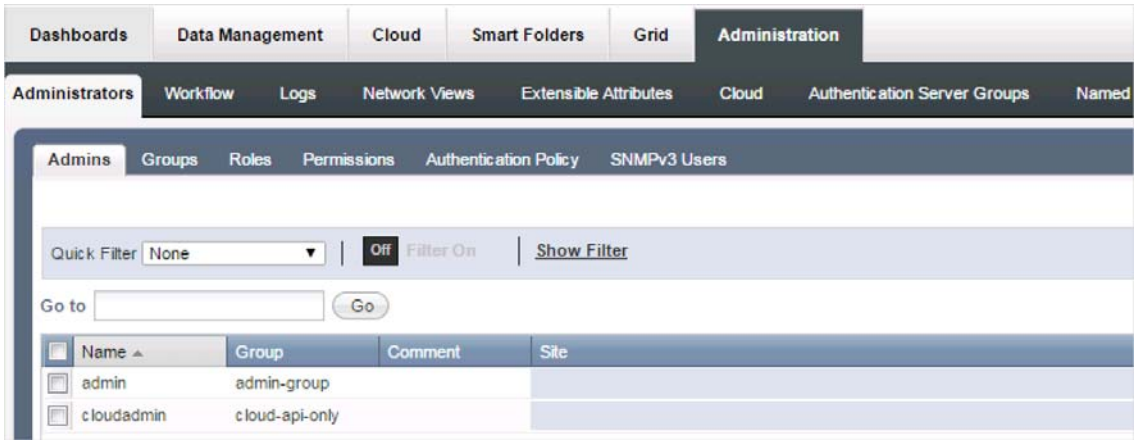
Setting Up vNIOS

In vNIOS, you need to activate the vNIOS Cloud Network Automation feature. You can do this by following one of the instructional videos In the Product Evaluation Portal as mentioned in the procedure below. Alternatively, you can configure cloud extensible attributes manually, as described in *Configuring Cloud Extensible Attributes* on page 4.

You also need to set up a user account as described below.

Do the following:

1. In the Product Evaluation Portal, scroll down to **Related Resources -> Video 1: Infoblox Cloud Network Automation Installation and Setup**.
2. Activate the vNIOS Cloud Network Automation feature by following the instruction in the later part of the video. You can skip over the section on configuring DHCP and DNS, as well as the section on vRealize Orchestrator.
3. In NIOS, create an admin account that uses cloud API.



4. Give cloud-api admin user permission to create and modify DNS views. Instructions on how to add permission to “cloud-api-only” group are included in the video. Follow the same instructions to add “All DNS Views” permission under the “DNS Permissions” permission type.

Configuring Cloud Extensible Attributes

If the Cloud Network Automation licensed feature is not activated, define the cloud extensible attributes manually.

Do the following:

1. In NIOS, go to **Administration -> Extensible Attributes**.
2. Add the following extensible attribute definitions:

Extensible Attribute	Type	Values
Cloud API Owned	List	True, False
CMP Type	String	Custom
Tenant ID	String	Custom

Installing Infoblox IPAM Driver for Docker

You can run Infoblox Docker Driver as an executable or run it as a pre-built container image. See the corresponding sections, *Running the Driver as an Executable* and *Running the Driver as a Pre-built Container Image*, below.

You can also build your own executable or container image for the Infoblox IPAM Driver for Docker. For information, see *Building Your Own Executable or Container Image* on page 6.

Note the following configurations of the Driver that you need to update when running the Driver, based on the vNIOS settings:

- Set grid-host to the management IP address of vNIOS.
- Set username and password to that for the cloud admin user on vNIOS.

To apply these configurations, edit the run.sh and run-container.sh shell scripts.

Note: By default, Infoblox IPAM Driver for Docker assumes that the Cloud Network Automation licensed feature is activated. If this is not the case, see *Configuring Cloud Extensible Attributes* on page 4.

Running the Driver as an Executable

Infoblox IPAM Driver for Docker accepts a number of arguments which can be listed by specifying -h. See the table below for the arguments list.

At the time of Driver startup, you can specify various input parameters: Infoblox Grid connectivity parameters, the network view name, the starting address for the network container (or a comma-separated list) and the default prefix length to be used when doing the “next available network”. The prefix lengths and network containers are specified on a per-address space basis, as you generally will have different size subnets in different address values for local and global spaces. For the input parameters list, see *Appendix A, "Input Parameters",* on page 9.

For example, following is a command that starts Infoblox IPAM Driver for Docker:

```
./ipam-driver --grid-host=192.168.124.200 --wapi-username=cloudadmin --wapi-password=cloudadmin
--local-view=local_view --local-network-container="192.168.0.0/20,192.169.0.0/22"
--local-prefix-length=25 --global-view=global_view --global-network-container="172.18.0.0/16"
--global-prefix-length=24
```

Note: You must have root permission to execute this command.

The Driver will connect to Infoblox NIOS and give output in the logs. The above command will allocate “global” subnets from 172.18.0.0/16 in network view “global-view”, with a default prefix length of 24. “Local” subnets will come from 192.168.0.0/20 and 192.169.0.0/22 in network view “local-view”, with a default prefix length of 25.

If the `conf-file` option is specified in the command line argument, it specifies a configuration file that accept the same set of parameters (except `conf_file`) in the following format:

```
[plugin_config]
driver_name="infoblox"
plugin_dir="/run/docker/plugins"

[grid_config]
grid_host="192.168.124.200"
wapi_port="443"
wapi_username="cloudadmin"
wapi_password="infoblox"
wapi_version="2.0"
ssl_verify="false"
http_request_timeout=60
http_pool_connections=10

[ipam_config]
global_view="default"
global_container="172.18.0.0/16"
global_prefix=24
local_view="default"
local_container="192.168.0.0/20"
local_prefix=25
```

If the same parameter is specified in both the configuration file and command line argument, the value specified in the command line argument takes precedence.

By default, Infoblox IPAM Driver for Docker uses Docker API version 1.22 to access Docker Remote API. The default can be overridden using the DOCKER_API_VERSION environment variable prior to running the Driver. For example:

```
DOCKER_API_VERSION=1.23
export DOCKER_API_VERSION
```

For convenience, a script called “run.sh” is provided. You can edit it to specify the desired options.

Running the Driver as a Pre-built Container Image

The best way to experiment with Infoblox IPAM Driver for Docker is to run it as a container. You can pull a pre-built Docker image from Docker Hub.

Do the following:

1. Run the Docker pull command:

   ```
   docker pull infoblox/ipam-driver
   ```

2. After successfully pulling the image, use the `docker run` command to run the Driver. For example:

   ```
   docker run -e DOCKER_API_VERSION=1.22 -v /var/run:/var/run -v /run/docker:/run/docker
   infoblox/ipam-driver --grid-host=192.168.124.200 --wapi-username=cloudadmin
   --wapi-password=cloudadmin --local-view=local_view
   --local-network-container="192.168.0.0/20,192.169.0.0/22" --local-prefix-length=25
   --global-view=global_view --global-network-container="172.18.0.0/16" --global-prefix-length=24
   ```

The -v options are necessary to provide the container access to the specified directories on the host file system.

For convenience, a script called “run-container.sh” is provided.

Building Your Own Executable or Container Image

Note: As a prerequisite, the Golang development environment must be installed. You can install it from here: *https://golang.org/doc/install*.

The Driver primarily depends on libnetwork and the infoblox-go-client. You can install them using the following commands:

```
go get github.com/docker/libnetwork  # libnetwork library (This also pulls down docker engine)
go get github.com/docker/engine-api  # engine-api library (NOTE: run "make deps" to get
dependencies)
go get github.com/infobloxopen/infoblox-go-client  # Infoblox client
```

`engine-api` is used by Infoblox IPAM Driver for Docker to obtain the Docker engine ID, which is used to populate the “Tenant ID” EA.

`infoblox-go-client` is used by Infoblox IPAM Driver for Docker to interact with Infoblox.

By default, the `master` branch of `libnetwork` and `docker` is used. To build with release versions, the corresponding release tags need to be checked out. Minimum requirement for Infoblox IPAM Driver for Docker is:

```
libnetwork  0.6
Docker      1.10.0
```

It has also been tested using master. For libnetwork release information, refer to: *https://github.com/docker/libnetwork/wiki*.

After a different version of the above is checked out, rebuild the Driver.

Building the Executable

A Makefile is provided to automate the build process.

To build Infoblox IPAM Driver for Docker, type `make` in the `docker-infoblox` source directory. This creates an executable called `ipam-driver`.

Building the Container Image

To build the container image using the Dockerfile in the “docker-infoblox” directory, run the following command:

```
make image
```

Pushing the Container Image to Docker Hub

The Makefile also includes a build target to push the “ipam-driver” container image to your Docker Hub.

Do the following:

1. Setup the following environment variable:

   ```
   export DOCKERHUB_ID="your-docker-hub-id"
   ```

2. Run the following command to push the “ipam-driver” image to your Docker Hub:

   ```
   make push
   ```

Network and IP Address Management for Containers Using Infoblox IPAM Driver for Docker

To start using the Driver, you should first create a Docker network specifying the Driver using the `--ipam-driver` option:

```
sudo docker network create --ipam-driver=infoblox priv-net
```

This creates a Docker network called “priv-net” which uses “infoblox” as the IPAM driver and the default “bridge” driver as the network driver. A network will be automatically allocated from the list of network containers specified during the Driver start up.

For the command line input parameters list, see *Appendix A, "Input Parameters",* on page 9. For the configuration file parameters, see *Appendix B, "Configuration File Parameters",* on page 10.

By default, the network will be created using the default prefix length specified during the Driver start up. You can override this using the `--ipam-opt` option. For example:

```
sudo docker network create --ipam-driver=infoblox --ipam-opt="prefix-length=24" priv-net-2
```

Additionally, if you are deploying containers in a cluster on different hosts, you can specify “network-name” using the --ipam-opt option. This will be used as an identifier so that Docker networks created on different Docker hosts can share the same IP address space. For example:

```
sudo docker network create --ipam-driver=infoblox --ipam-opt="network-name=blue" blue-net
```

This will allocate a network, for example, 192.168.10.0/24, from the default address pool. Additionally, the network will be tagged in Infoblox with the network name “blue”. When the same command is issued on a different host, the Driver will look for a network on Infoblox tagged with the same name, “blue”, and will share the same network, 192.168.10.0/24, instead of allocating a new one.

After the network is created, Docker containers can be started attaching to the “priv-net” network created above. For example, the following command runs the “ubuntu” image:

```
sudo docker run -i -t --net=priv-net --name=ubuntu1 ubuntu
```

When the container is deployed, verify that the IP address assigned to the container is allocated by the Driver from NIOS by using the `ifconfig` command.

When the network is no longer needed, you can delete it with the Docker `network rm` command. At that point, Infoblox IPAM Driver for Docker will receive a `ReleasePool` API call so that the subnet can be de-allocated.

Note: If you want to modify the network view configuration of an existing instance of the Driver, you need to restart the Docker engine as well as restarting the Driver with the new configuration. On the other hand, if you want to deploy a new, e.g. additional, instance of the Driver with a different set of configurations including network views, you can start a new instance specifying a different `--driver-name` option. For example, `--driver-name infoblox-2`. To use the new Driver instance, simply specify the new Driver name, in this case “infoblox-2”, via the `--ipam-driver` option when creating a Docker network.

Related Documentation

Other Infoblox IPAM Driver for Docker documentation:

- *Release Notes for the Infoblox IPAM Driver for Docker*

See also related Infoblox NIOS documentation:

- *Infoblox NIOS Administrator Guide*
- *Infoblox CLI Guide*
- *Infoblox API Documentation*

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.

Appendix A Input Parameters

Following is the list of command line input parameters for Infoblox IPAM Driver for Docker.

Parameter	Type	Description
-conf-file	string	File path of configuration file.
-driver-name	string	Name of Infoblox IPAM Driver for Docker. The default is “infoblox”.
-global-network-container	string	The container from which subnets will be allocated when no subnet is specified during network creation. The default is “172.18.0.0/16”.
-global-prefix-length	unit	The default CIDR prefix length when allocating a global subnet. The default is “24”.
-global-view	string	Infoblox network view for global address space. The default is “default”.
-grid-host	string	IP of the Infoblox Grid host. The default is “192.168.124.200”.
-local-network-container	string	The container from which subnets will be allocated when no subnet is specified during network creation. The default is “192.168.0.0/16”.
-local-prefix-length	unit	The default CIDR prefix length when allocating a local subnet. The default is 24.
-local-view	string	Infoblox network view for local address space. The default is “default”.
-plugin-dir	string	Docker plugin directory where the Driver socket is created. The default is “/run/docker/plugins”.
-ssl-verify	string	Specifies whether to verify server certificate (true/false). If a file path is specified, it is assumed to be a certificate file and is used to verify server certificate.
-wapi-password	string	Infoblox WAPI user account password.
-wapi-port	string	Infoblox WAPI port. The default is “443”.
-wapi-username	string	Infoblox WAPI user account name.
-wapi-version	string	Infoblox WAPI version. The default is “2.0”.
-http-request-timeout	string	Infoblox WAPI request timeout in seconds. The default is “60”.
-http-pool-connections	string	Infoblox WAPI request connection pool size. The default is “10”.

Appendix B Configuration File Parameters

Following is the list of configuration file parameters for Infoblox IPAM Driver for Docker.

Parameter	Type	Description
driver_name	string	Name of Infoblox IPAM Driver for Docker. The default is “infoblox”.
global_network_container	string	The container from which subnets will be allocated when no subnet is specified during network creation. The default is “172.18.0.0/16”.
global_prefix_length	unit	The default CIDR prefix length when allocating a global subnet. The default is “24”.
global_view	string	Infoblox network view for global address space. The default is “default”.
grid_host	string	IP of the Infoblox Grid host. The default is “192.168.124.200”.
local_network_container	string	The container from which subnets will be allocated when no subnet is specified during network creation. The default is “192.168.0.0/16”.
local_prefix_length	unit	The default CIDR prefix length when allocating a local subnet. The default is 24.
local_view	string	Infoblox network view for local address space. The default is “default”.
plugin_dir	string	Docker plugin directory where the Driver socket is created. The default is “/run/docker/plugins”.
ssl_verify	string	Specifies whether to verify server certificate (true/false). If a file path is specified, it is assumed to be a certificate file and is used to verify server certificate.
wapi_password	string	Infoblox WAPI user account password.
wapi_port	string	Infoblox WAPI port. The default is “443”.
wapi_username	string	Infoblox WAPI user account name.
wapi_version	string	Infoblox WAPI version. The default is “2.0”.
http_request_timeout	string	Infoblox WAPI request timeout in seconds. The default is “60”.
http_pool_connections	string	Infoblox WAPI request connection pool size. The default is “10”.