NX-OS Bash Shell DNS Configuration

Contents

Introduction

This document describes the steps used to configure DNS servers within Bash to allow for the resolution of DNS hostnames to IP addresses.

Cisco Nexus 3000 and 9000 series devices allow access to NX-OS's underlying Linux system through Bash (Bourne-Again SHell). Bash enables system management and monitoring through a Linux environment. For more information about Bash on NX-OS, refer to the Bash chapter of the Cisco Nexus 9000 Series NX-OS Programmability Guide.

It may be necessary to translate human-friendly domain names into numeric IP addresses while performing normal tasks in the Bash shell. Such tasks include using the `curl` or `wget` utilities to access resources from a web server or to download Docker images using the `docker pull` command.

Prerequisites

Requirements

This document is not restricted to specific software and hardware versions.

> **Note**: The Bash shell is to be enabled on your Cisco Nexus device. Refer to the "Accessing Bash" section of the Bash chapter in the Cisco Nexus 9000 Series NX-OS Programmability Guide for instructions to enable the Bash shell.

Components Used

The information in this document is based on these software and hardware versions:

- Nexus 9000 platform starting from NX-OS Release 6.1(2)I2(1)
- Nexus 3000 platform starting from NX-OS Release 6.0(2)U4(1)

The information in this document was created from devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Bash Shell DNS Configuration

The Linux environment accessed through the Bash shell utilizes the **/etc/resolv.conf** file to store DNS configuration, similarly to most other Unix-like operating systems.

1. Log in to the Bash shell as the root user through the `run bash sudo su -` command.

```
Nexus# run bash sudo su -
root@Nexus#whoami
root
```

2. View the current contents of the **/etc/resolv.conf** file. In this example, the file is empty.

```
root@Nexus#cat /etc/resolv.conf
root@Nexus#
```

3. Open the **/etc/resolv.conf** file for editing using the `vi` text editor.

```
root@Nexus#vi /etc/resolv.conf
```

4. Press the `i` key to enter INSERT mode, then enter your desired configuration. Refer to the **/etc/resolv.conf File Format** section of this document for more information about how the format of configuration within the **/etc/resolv.conf** file.

5. After the file is modified, press the `ESC` key to exit INSERT mode, then enter `:x` to save all changes to the file and close it.

Bash Shell DNS Verification

Once the changes are made to the Bash shell's DNS configuration, verify that the changes result in successful domain name resolution. The simplest method of testing domain name resolution is through using the `ping` utility using a domain hostname as a target. This document demonstrates how to verify valid DNS configuration using a test host of **test.cisco.com** and DNS servers of 192.168.2.1 and 192.168.2.2.

Step 1. Verify the Use of a Valid Network Namespace to Test.

By default, the Bash shell uses the **default** network namespace unless instructed otherwise. Network namespaces are logically equivalent to NX-OS VRFs, and the `ip netns` the command displays a list of namespaces that are available to the Bash shell, as demonstrated below:

```
root@Nexus#ip netns
EXAMPLE-VRF (id: 2)
management (id: 1)
default (id: 0)
```

A valid network namespace to test with is one that has IP connectivity to the DNS nameservers configured in the **/etc/resolv.conf** file, as well as IP connectivity to the IP address that your test host resolves to.

One can utilize the `ip netns exec {namespace} {desired-command}` command to execute a command `{desired-command}`in the namespace `{namespace}`. Alternatively, one can execute the Bash shell within the context of a specific namespace with the `ip netns exec {namespace} bash` command. The former methodology is used in the example here, it is verified that the **management** namespace has IP connectivity with the IP address owned by the test.cisco.com host (which is 192.168.2.100) and both DNS servers (192.168.2.1 and 192.168.2.2).

```
root@Nexus#ip netns exec management ping 192.168.2.100 -c 5
PING 192.168.2.100 (192.168.2.100) 56(84) bytes of data.
64 bytes from 192.168.2.100: icmp_seq=1 ttl=59 time=0.277 ms
64 bytes from 192.168.2.100: icmp_seq=2 ttl=59 time=0.284 ms
64 bytes from 192.168.2.100: icmp_seq=3 ttl=59 time=0.280 ms
64 bytes from 192.168.2.100: icmp_seq=4 ttl=59 time=0.274 ms
64 bytes from 192.168.2.100: icmp_seq=5 ttl=59 time=0.297 ms

--- 192.168.2.100 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4001ms
rtt min/avg/max/mdev = 0.274/0.282/0.297/0.017 ms

root@Nexus#ip netns exec management ping 192.168.2.1 -c 5
PING 192.168.2.1 (192.168.2.1) 56(84) bytes of data.
64 bytes from 192.168.2.1: icmp_seq=1 ttl=59 time=0.277 ms
64 bytes from 192.168.2.1: icmp_seq=2 ttl=59 time=0.284 ms
64 bytes from 192.168.2.1: icmp_seq=3 ttl=59 time=0.280 ms
64 bytes from 192.168.2.1: icmp_seq=4 ttl=59 time=0.274 ms
64 bytes from 192.168.2.1: icmp_seq=5 ttl=59 time=0.297 ms

--- 192.168.2.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4001ms
rtt min/avg/max/mdev = 0.274/0.282/0.297/0.017 ms

root@Nexus#ip netns exec management ping 192.168.2.2 -c 5
PING 192.168.2.2 (192.168.2.2) 56(84) bytes of data.
64 bytes from 192.168.2.2: icmp_seq=1 ttl=59 time=0.277 ms
64 bytes from 192.168.2.2: icmp_seq=2 ttl=59 time=0.284 ms
64 bytes from 192.168.2.2: icmp_seq=3 ttl=59 time=0.280 ms
64 bytes from 192.168.2.2: icmp_seq=4 ttl=59 time=0.274 ms
64 bytes from 192.168.2.2: icmp_seq=5 ttl=59 time=0.297 ms

--- 192.168.2.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4001ms
rtt min/avg/max/mdev = 0.274/0.282/0.297/0.017 ms
```

Step 2. Verify the Functioning of DNS Resolution using the Test Host's Hostname.

Use the `ping` utility with a target of the test host's hostname. If ICMP Replies are received from the test host, and the IP address contained within the ICMP Replies is the IP address we expect the

hostname to resolve to, then DNS resolution is confirmed to be working within the Bash shell.

This example here demonstrates how the `ping` utility is used within the **management** namespace to verify correct DNS resolution. Note how the domain hostname of **test.cisco.com** resolves to **192.168.2.100**, which is the IP address we expect that hostname to resolve to.

```
root@Nexus#ip netns exec management ping test.cisco.com -c 5
PING test.cisco.com (192.168.2.100) 56(84) bytes of data.
64 bytes from test.cisco.com (192.168.2.100): icmp_seq=1 ttl=59 time=0.617 ms
64 bytes from test.cisco.com (192.168.2.100): icmp_seq=2 ttl=59 time=0.341 ms
64 bytes from test.cisco.com (192.168.2.100): icmp_seq=3 ttl=59 time=0.310 ms
64 bytes from test.cisco.com (192.168.2.100): icmp_seq=4 ttl=59 time=0.379 ms
64 bytes from test.cisco.com (192.168.2.100): icmp_seq=5 ttl=59 time=0.296 ms

--- test.cisco.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4004ms
rtt min/avg/max/mdev = 0.296/0.388/0.617/0.119 ms
```

If the namespace used to test DNS resolution has IP connectivity to the Internet, one can ping **cisco.com** to verify that external domain names can be resolved in addition to internal domain names. This is particularly important if one needs to use utilities such as `curl` and `wget` against public web servers. The example here demonstrates how the `ping` utility can be used within the **management** namespace (which has IP connectivity to the Internet) to verify correct external DNS resolution.

```
root@Nexus#ip netns exec management ping cisco.com -c 5
PING cisco.com (72.163.4.161) 56(84) bytes of data.
64 bytes from www1.cisco.com (72.163.4.161): icmp_seq=1 ttl=239 time=29.2 ms
64 bytes from www1.cisco.com (72.163.4.161): icmp_seq=2 ttl=239 time=29.2 ms
64 bytes from www1.cisco.com (72.163.4.161): icmp_seq=3 ttl=239 time=29.3 ms
64 bytes from www1.cisco.com (72.163.4.161): icmp_seq=4 ttl=239 time=29.2 ms
64 bytes from www1.cisco.com (72.163.4.161): icmp_seq=5 ttl=239 time=29.2 ms

--- cisco.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4005ms
rtt min/avg/max/mdev = 29.261/29.283/29.335/0.111 ms
```

/etc/resolv.conf File Format

Some common configuration parameters are explained here. Ensure to modify all the configuration parameters to match your environment.

- **domain {domain-name.tld}** - Defines a default domain name `{domain-name.tld}` to append to hostnames that do not end with a period. There can only be one `domain` entry within the **/etc/resolv.conf** file.
- **search {domain-name-1.tld} [domain-name-2.tld...]** - Defines a space-delimited list of domain names (`{domain-name-1.tld}` and, optionally, `[domain-name-2.tld]`) to append to hostnames.

  Note: The `domain` and `search` entries are mutually exclusive - only one can be in use at a time. If both entries are included in the **/etc/resolv.conf** file, the entry that appears last in the file is used.

- **nameserver {address-1}** - Defines an IP address `{address-1}` for a DNS server where DNS resolution requests are forwarded to. Multiple `nameserver` entries are allowed within a single

file, up to a maximum of three.

Examples

This example shows the contents of the **/etc/resolv.conf** file where the default domain of the environment is cisco.com and DNS servers within the environment have IP addresses of 192.168.2.1 and 192.168.2.2. In this scenario, if the Bash shell needs to resolve the IP address for a device with a hostname of **foo**, it appends **cisco.com** to the end of the hostname such that the Fully Qualified Domain Name (FDQN) of the host is **foo.cisco.com**.

```
domain cisco.com
nameserver 192.168.2.1
nameserver 192.168.2.2
```

The following example shows the contents of the **/etc/resolv.conf** file where either the **cisco.com** or the **bar.com** domain names may be used to resolve DNS hostnames. DNS servers within the environment have IP addresses of 192.168.2.1 and 192.168.2.2. In this scenario, if the Bash shell needs to resolve the IP address of a device with a hostname of **foo**, it attempts to resolve **foo.cisco.com** first, then attempt to resolve **foo.bar.com** next if the resolution for **foo.cisco.com** fails.

```
search cisco.com bar.com
nameserver 192.168.2.1
nameserver 192.168.2.2
```

Related Information

- **Cisco Nexus 9000 Series NX-OS Programmability Guide, Release 9.x**
- **Cisco Nexus 9000 Series NX-OS Programmability Guide, Release 7.x**
- **Cisco Nexus 9000 Series NX-OS Programmability Guide, Release 6.x**
- **Cisco Nexus 3000 Series NX-OS Programmability Guide, Release 9.x**
- **Cisco Nexus 3000 Series NX-OS Programmability Guide, Release 7.x**
- **Cisco Nexus 3000 Series NX-OS Programmability Guide, Release 6.x**
- **Cisco Nexus 3500 Series NX-OS Programmability Guide, Release 9.x**
- **Cisco Nexus 3500 Series NX-OS Programmability Guide, Release 7.x**
- **Cisco Nexus 3500 Series NX-OS Programmability Guide, Release 6.x**
- **Cisco Nexus 3600 Series NX-OS Programmability Guide, Release 9.x**
- **Cisco Nexus 3600 Series NX-OS Programmability Guide, Release 7.x**
- **Programmability and Automation with Cisco Open NX-OS**