RootUsers Guides, tutorials, reviews and news for System Administrators.

How To Troubleshoot DNS Client Issues in Linux

Posted by Jarrod on September 29, 2015 Go to comments Leave a comment (4)

DNS resolution is an important service, without it functioning properly domain names will not be correctly resolved to IP addresses preventing other network services from working correctly. Therefore it is equally important to know how to troubleshoot DNS issues on a Linux client and fix any problems to reduce disruption.

There are multiple potential points of failure during the DNS lookup process such as at the system performing the lookup, at the DNS cache, or on an external DNS server. Here we will cover how to check these and perform various tests to identify where exactly the problem lies.

Local Server Configuration

First off it's important to understand the 'hosts' section of the /etc/nsswitch.conf file, the default configuration for hosts is shown below.

```
hosts:      files dns myhostname
```

Essentially this means that host name resolution will be performed in the order specified, left to right. First files will be checked, followed by DNS.

As files are first these will be checked first, this references the local /etc/hosts file which contains static host name to IP address mappings. This file takes priority over any DNS resolution, any changes to the file will be placed straight into the DNS cache of that local server. Below is an example line of configuration from /etc/hosts

```
1.1.1.1     google.com
```

As this entry is in our host file locally, if we try to reach google.com our local machine will think that 1.1.1.1 is the correct IP address of google.com and will not perform a DNS lookup. This is demonstrated below by trying to ping google.com, DNS is not consulted as there is a hosts file entry which takes priority.

```
[root@centos ~]# ping google.com
PING google.com (1.1.1.1) 56(84) bytes of data.
```

If there is no entry in the hosts file DNS will be used next as per /etc/nsswitch.conf. The servers used for DNS resolution will be specified in the /etc/resolv.conf file, below is an example configuration of this file.

```
nameserver 192.168.0.1
```

In this case all DNS queries of our system will go to the DNS server at 192.168.0.1. Other secondary and tertiary DNS servers can also be specified here as backups.

Testing DNS

For DNS resolution to succeed to 192.168.0.1, the DNS server at 192.168.0.1 will need to accept TCP and UDP traffic over port 53 from our server. A port scanner such as the nmap tool can be used to confirm if the DNS server is available on port 53 as shown below.

Note: To install nmap run 'yum install nmap -y'.

```
[root@centos ~]# nmap -sU -p 53 192.168.0.1
Starting Nmap 6.40 ( http://nmap.org ) at 2015-08-26 15:22 AEST
Nmap scan report for 192.168.0.1
Host is up (0.00091s latency).
PORT   STATE         SERVICE
53/udp open|filtered domain
MAC Address: 02:00:79:55:00:0D (Unknown)
Nmap done: 1 IP address (1 host up) scanned in 0.29 seconds

[root@centos ~]# nmap -sT -p 53 192.168.0.1
Starting Nmap 6.40 ( http://nmap.org ) at 2015-08-26 15:22 AEST
Nmap scan report for 192.168.0.1
Host is up (0.00099s latency).
PORT   STATE SERVICE
53/tcp open  domain
MAC Address: 02:00:79:55:00:0D (Unknown)
Nmap done: 1 IP address (1 host up) scanned in 0.07 seconds
```

It's worth noting that scanning UDP with nmap is not reliable due to the nature of UDP, this is why the state is listed as open or filtered. We can clearly see that TCP 53 is definitely open and responding which is a good sign, if the state was reported as filtered the next thing to investigate would be the connectivity to the DNS server, in particular any firewall running on the DNS server would need to be configured to allow TCP and UDP port 53 traffic in.

By running a packet capture we can view any DNS queries over the network, in this example we are running tcpdump to our local DNS server at 192.168.0.1 and we can see our request from 192.168.0.100 requesting the A record of google.com as well as the response of 216.58.220.142 which is returned from our local DNS server.

```
[root@testing ~]# tcpdump -n host 192.168.0.1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
15:29:52.439222 IP 192.168.0.100.32811 > 192.168.0.1.domain: 8134+ A? google.com.
15:29:52.440153 IP 192.168.0.1.domain > 192.168.0.100.32811: 8134 1/0/0 A 216.58.22
```

The Domain Information Groper (dig) tool can be used to perform DNS queries as demonstrated below. We are again querying for google.com and we are again returned the A record IP address of 216.58.220.142.

Note: Dig is provided by the bind-utils package which can be installed with 'yum install bind-utils'.

```
[root@testing ~]# dig google.com

; <<>> DiG 9.9.4-RedHat-9.9.4-18.el7_1.3 <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 32536
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             65      IN      A       216.58.220.142
```

The status of the dig query correctly returned the IP address from our local DNS server at 192.168.0.1 and the status was NOERROR, which is returned when the query has been successfully resolved. Response codes can help you in the troubleshooting process, for a full list of them refer to RFC 5395.

Test Authoritive DNS Server

With dig we can also directly query the authoritative name servers for a domain, these are the DNS servers that hold the authoritative records for the domains DNS zone – the source of truth. If a correct response is

received from the authoritative DNS server but not when querying against your own DNS server then you should investigate why your local DNS server is not able to resolve the record.

To get the name servers of a domain we can use the 'whois' command as shown below. This is part of the whois package and can be installed with 'yum install whois -y' if not already present.

```
[root@testing ~]# whois google.com | grep -i "name server"
   Name Server: NS1.GOOGLE.COM
   Name Server: NS2.GOOGLE.COM
   Name Server: NS3.GOOGLE.COM
   Name Server: NS4.GOOGLE.COM
```

As shown google.com currently has 4 authoritative name servers, if we run a dig directly against any of these we should receive an authoritative response, that is an up to date and non cached response straight from the source rather than from our local DNS server. In the below example we have run our query against @ns1.google.com

```
[root@testing ~]# dig @NS1.GOOGLE.COM google.com

; <<>> DiG 9.9.4-RedHat-9.9.4-18.el7_1.3 <<>> @NS1.GOOGLE.COM google.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 3477
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             300     IN      A       216.58.220.142
```

While the A record returned is the same in this instance, note that in this dig response we now have the "aa" flag in the header which represents that this is an authoritative answer and is not a cached response. If we run this same dig command again, the 300 second TTL that was returned in the answer section will continually state that the TTL is 300 seconds as the response is authoritative.

However if we were to run this dig without specifying @ns1.google.com we would be querying our 192.168.0.1 DNS server which is not authoritative for the google.com domain, after the first result the record will be cached locally. This can be confirmed by running the dig command again, as the TTL value will drop down until it reaches 0 and is removed from the cache completely.

By querying the authoritative name server directly we ensure that we are getting the most up to date response rather than a potential old cached response from our own local DNS server or local DNS cache.

Summary

As DNS is an important service being able to troubleshoot it is a useful skill. By default Linux will first check it's local host file /etc/hosts before querying DNS servers defined in /etc/resolv.conf. It is important to confirm that the correct DNS servers have been specified within this file and that you can connect to them on TCP/UDP port 53. DNS queries can be checked with the dig command, either against the local DNS server or against the authoritative name server for the domain which will provide an up to date non cached result.

This post is part of our Red Hat Certified Engineer (RHCE) exam study guide series. For more RHCE related posts and information check out our full RHCE study guide.

Share this:

How To, Linux DNS, Linux, RHCE, troubleshooting

← Edit the XenServer Storage Heartbeat How To Change Hostname In Linux →

Leave a comment ? 4 Comments.

thegeekaid December 18, 2016 at 12:55 pm Reply

Thank you, very informative post

anonuser May 17, 2017 at 2:52 pm Reply

thank you for a posting a very useful & valuable article..

Incognito May 30, 2017 at 3:04 am Reply

Thanks, this is explained very clear!

sean March 19, 2018 at 1:44 am Reply

Thanks a lot. It is great

Leave a Comment

NOTE - You can use these HTML tags and attributes:

```
<a href="" title=""> <abbr title=""> <acronym title=""> <b> <blockquote
cite=""> <cite> <code> <del datetime=""> <em> <i> <q cite=""> <s>
<strike> <strong>
```

NAME

EMAIL

Website URL

Notify me of follow-up comments by email.

Notify me of new posts by email.

SUBMIT

Subscribe

Receive new post notifications by email for free!
Unsubscribe any time.

Email Address

Subscribe

Recent Posts

- Create and edit text files – RHEL 8 RHCSA
- Create, delete, copy, and move files and directories – RHEL 8 RHCSA
- Create hard and soft links – RHEL 8 RHCSA

AMD Radeon VII – Hashcat Benchmark

How To Enable Ping In Windows Server 2019 Firewall

Categories

Command Examples

Exam Guides

How To

Linux

Security

Technology

Uncategorized

Windows

XenServer