THE ULTIMATE GUIDE TO TROUBLESHOOTING NETWORKING IN LINUX

A DEEP DIVE

Prepared By: Hosni Zaaraoui

Introduction: Why Networking Troubleshooting Matters

Networking is the backbone of any system, whether it's a personal computer, a server, or a complex cloud infrastructure. When networking issues arise, they can disrupt communication, hinder productivity, and even bring entire systems to a halt. In Linux, networking troubleshooting is a critical skill for system administrators, developers, and IT professionals. This guide will take you deep into the world of Linux networking, equipping you with the knowledge and tools to diagnose and resolve issues like a pro.

By the end of this guide, you'll have a comprehensive understanding of:

- ✓ **Essential commands** to diagnose network issues
- ✓ **Key configuration and log files** to monitor
- ✓ **Common network problems** and their solutions
- ✓ **Advanced techniques** for configuring and troubleshooting network services like DHCP (ISC & KEA)

Let's dive in!

1. Checking Network Connectivity

1.1 Using ping to Check Basic Connectivity

The ping command is the first tool in your troubleshooting arsenal. It sends ICMP echo requests to a specified host and waits for a response. This helps you determine if the host is reachable.

```
ping -c 4 google.com
```

- **If you get responses:** The network is working.
- **If there's no response:** Investigate further:
  - **Network Interface:** Is the interface up and configured correctly?
  - **DNS Settings:** Can the hostname be resolved to an IP address?
  - **Firewall Rules:** Are ICMP requests being blocked?

Tip: If google.com doesn't respond but 8.8.8.8 (Google's public DNS) does, the issue is likely with DNS resolution.

1.2 Using traceroute to Identify Network Hops

traceroute is a powerful tool for identifying where packets are being dropped along the path to the destination. It shows each hop (router) the packet passes through, along with the time taken for each hop.

```
traceroute google.com
```

– **If a hop fails:** The issue might be with that specific router or network segment.

– **High latency on a hop:** Indicates potential congestion or routing issues.

1.3 Using ip to Check Network Interfaces

The ip command is a modern replacement for ifconfig and provides detailed information about network interfaces.

```
ip a
```

– **UP and has an IP:** The interface is working correctly.

– **DOWN:** The interface is disabled. Bring it up with:

```
sudo ip link set eth0 up
```

– **No IP:** Check your DHCP or static IP configuration.

1.4 Using netstat or ss to Check Open Ports

To see which services are running and which ports are open, use ss (or netstat for older systems).

```
ss -tulnp
```

- **-t:** Show TCP connections
- **-u:** Show UDP connections
- **-l:** Show listening ports
- **-n:** Show port numbers instead of names
- **-p:** Show processes using the ports

– If a service **isn't working**, verify that it's listening on the correct port.

1.5 Checking Routing with ip route

If you're having trouble reaching external networks, check your routing table:

```
ip route show
```

– **No default route:** Your system can't access external networks. Add a default route manually:

```
sudo ip route add default via 192.168.1.1
```

2. Advanced Networking Troubleshooting Techniques

As you dive deeper into networking issues, it's crucial to go beyond the basics. Here are some advanced techniques to take your troubleshooting skills to the next level:

2.1 Use tcpdump for Network Packet Analysis

To understand what's happening on the network, capturing and analyzing packets can be incredibly helpful.

- Use tcpdump to capture packets on a specific interface:

```
sudo tcpdump -i eth0
```

– This will display **live network traffic**. You can refine it further by adding filters, such as capturing only HTTP traffic:

```
sudo tcpdump -i eth0 tcp port 80
```

Tip: Save the captured packets to a file for later analysis:

```
sudo tcpdump -i eth0 -w /tmp/capture.pcap
```

Then, analyze the file using Wireshark or tcpdump:

```
sudo tcpdump -r /tmp/capture.pcap
```

2.2 Check for Network Interface Bonding or VLAN Issues

If you're using bonding or VLANs, misconfigurations can cause network connectivity issues. Verify the configuration of bonded interfaces and VLANs using ip and vconfig.

To check bonding status:

```
cat /proc/net/bonding/bond0
```

To verify VLAN configuration:

```
ip link show type vlan
```

2.3 Test MTU (Maximum Transmission Unit) Settings

Incorrect MTU settings can cause issues, especially with VPNs or large file transfers. Test MTU by pinging with a specific packet size.

First, determine the path MTU:

```
ping -M do -s 1472 google.com
```

– If the packet doesn’t reach its destination, decrease the packet size (e.g., 1400) until it works. This gives you the MTU size that the path supports.

2.4 Debugging Routing Issues with ip rule and ip route

Sometimes routing issues aren't obvious with just ip route. Use ip rule to inspect policy routing and ensure correct routing tables are in place.

```
ip rule show
```

– You can use this command to check if any routing rules are influencing traffic and causing misrouted packets.

2.5 Check for Network Namespace Issues

In modern Linux systems, network namespaces are often used to isolate network environments for containers and virtual machines. If you're troubleshooting network connectivity in these environments, it's important to check the namespace.

- List existing namespaces:

```
ip netns
```

- To enter a network namespace:

```
sudo ip netns exec <namespace-name> bash
```

– Once inside, use traditional tools like ping, traceroute, and ip to check connectivity.

3. Key Network Configuration and Log Files

3.1 Critical Network Configuration Files

File	Purpose
`/etc/network/interfaces`	Network configuration (Debian/Ubuntu)
`/etc/sysconfig/network-scripts/`	Network configuration (RHEL/CentOS)
`/etc/hostname`	System hostname
`/etc/resolv.conf`	DNS configuration
`/etc/hosts`	Local hostname resolution

Example: Setting a Static IP (Ubuntu/Debian - /etc/network/interfaces)

```
auto eth0

iface eth0 inet static

    address 192.168.1.100

    netmask 255.255.255.0

    gateway 192.168.1.1

    dns-nameservers 8.8.8.8 8.8.4.4
```

Tip: After making changes, restart the networking service:

```
sudo systemctl restart networking
```

3.2 Checking Network Logs for Errors

Logs are your best friend when troubleshooting. Here are some useful commands:

View networking logs:

```
sudo journalctl -u networking --no-pager | tail -20
```

or

```
sudo journalctl -u NetworkManager.service --no-pager | tail -20
```

Check kernel logs for network interfaces:

```
sudo dmesg | grep -i eth
```

Monitor system logs in real-time:

```
sudo tail -f /var/log/syslog
```

4. Network Services Configuration: NetworkManager vs. netplan

When configuring network services in Linux, the tools and methods you use depend on whether your system uses NetworkManager or netplan. Below, I'll outline the necessary changes for both approaches, focusing on key tasks like setting static IPs, configuring DNS, and managing network interfaces.

4.1 Using NetworkManager

NetworkManager is a dynamic network control and configuration system that simplifies network management, especially on desktop environments. It's commonly used in distributions like Fedora, RHEL, and Ubuntu Desktop.

4.1.1 Setting a Static IP with NetworkManager

To configure a static IP using NetworkManager, you can use the nmcli command-line tool or edit configuration files.

- **Using nmcli:**

```
    nmcli connection modify <connection_name> ipv4.method
manual ipv4.addresses 192.168.1.100/24 ipv4.gateway
192.168.1.1 ipv4.dns 8.8.8.8
```

- Replace **<connection_name>** with the name of your network connection (e.g., eth0 or Wired connection 1).
- **ipv4.method manual:** Sets the interface to use a static IP.
- **ipv4.addresses:** Specifies the IP address and subnet mask.
- **ipv4.gateway:** Defines the default gateway.
- **ipv4.dns:** Configures DNS servers.

- **Restart the Connection:**

```
    nmcli connection down <connection_name> && nmcli
connection up <connection_name>
```

- **Using Configuration Files:**

Edit the connection file in /etc/NetworkManager/system-connections/:

```
    [connection]
    id=eth0
    type=ethernet

    [ipv4]
    method=manual
    addresses1=192.168.1.100/24,192.168.1.1
    dns=8.8.8.8;8.8.4.4;
```

- **Save the file and restart NetworkManager:**

```
sudo systemctl restart NetworkManager
```

4.1.2 Configuring DNS with NetworkManager

You can configure DNS settings using nmcli:

```
nmcli connection modify <connection_name> ipv4.dns
"8.8.8.8 8.8.4.4"
```

To apply changes:

```
nmcli connection up <connection_name>
```

4.1.3 Managing Network Interfaces

- **List Connections:**

```
nmcli connection show
```

- **Bring an Interface Up/Down:**

```
nmcli connection up <connection_name>

nmcli connection down <connection_name>
```

- **Add a New Connection:**

```
    nmcli connection add type ethernet ifname eth0 con-name
eth0-static
```

4.2 Using netplan

netplan is a network configuration abstraction tool used in Ubuntu Server and some desktop versions. It uses YAML configuration files to define network settings, which are then applied by the backend (e.g., NetworkManager or systemd-networkd).

4.2.1 Setting a Static IP with netplan

Edit the netplan configuration file, typically located in /etc/netplan/. For example:

```
    network:
      version: 2
      renderer: networkd  # or NetworkManager for desktop
      ethernets:
        eth0:
          dhcp4: no
          addresses:
            - 192.168.1.100/24
          gateway4: 192.168.1.1
          nameservers:
            addresses:
              - 8.8.8.8
              - 8.8.4.4
```

- **dhcp4: no:** Disables DHCP.
- **addresses**: Specifies the static IP and subnet mask.
- **gateway4:** Defines the default gateway.
- **nameservers:** Configures DNS servers.

- **Apply the Configuration:**

```
sudo netplan apply
```

4.2.2 Configuring DNS with netplan

DNS settings are configured under the nameservers section in the netplan YAML file:

```
nameservers:
  addresses:
    - 8.8.8.8
    - 8.8.4.4
```

Apply the changes:

```
sudo netplan apply
```

4.2.3 Managing Network Interfaces

- **Check Current Configuration:**

```
netplan status
```

- **Debug Configuration:**

```
sudo netplan --debug apply
```

- **Regenerate Configuration:**

If you’re switching from NetworkManager to netplan or vice versa, regenerate the configuration:

```
sudo netplan generate
```

4.3 Key Differences Between NetworkManager and netplan

Feature	**NetworkManager**	**netplan**
Primary Use Case	Desktop environments, dynamic networks	Server environments, static configurations
Configuration Method	nmcli or GUI tools	YAML files
Backend	Manages connections directly	Uses systemd-networkd or NetworkManager
Ease of Use	More interactive, user-friendly	More declarative, suited for automation

4.4 Switching Between NetworkManager and netplan

If your system uses both tools, you can specify which one to use in the netplan configuration:

```
network:
  version: 2
  renderer: NetworkManager  # or networkd
```

– **Renderer:** Determines the backend for netplan. Use NetworkManager for desktop systems and networkd for servers.

5. Common Network Problems and Fixes

5.1 DNS Issues

If you're able to ping an IP address but not a domain name, your DNS might be misconfigured. To verify DNS settings, check /etc/resolv.conf or use dig.

```
dig google.com
```

– If you get no response or an error, update /etc/resolv.conf with the correct DNS servers.

```
nameserver 8.8.8.8

nameserver 8.8.4.4
```

Tip: Use systemd-resolve on systems with systemd:

```
systemd-resolve --status
```

5.2 DHCP Issues

If your system isn't receiving an IP address from the DHCP server, ensure that the DHCP client is running and properly configured.

```
sudo systemctl status dhclient
```

If the DHCP server is misbehaving, restart the service:

```
sudo systemctl restart isc-dhcp-server
```

- Also, ensure that the DHCP server has available IP addresses to assign.

5.3 Firewall Blocking Traffic

Sometimes, firewalls block outgoing or incoming traffic, even if the network is working fine. Check the status of your firewall:

```
sudo ufw status
```

- If you're using iptables, inspect rules with:

```
sudo iptables -L
```

- If necessary, add rules to allow traffic on specific ports, like HTTP or SSH:

```
sudo ufw allow 80/tcp

sudo ufw allow 22/tcp
```

5.4 Interface Not Coming Up

If your network interface isn't coming up after a reboot or configuration change, check dmesg for any kernel errors related to the interface.

```
sudo dmesg | grep eth0
```

- Common issues could be related to incorrect drivers or hardware problems. You can also try reloading the network interface:

```
sudo ifdown eth0 && sudo ifup eth0
```

5.5 Slow Network Speed

If the network is slow, identify the bottleneck by testing different parts of the network. First, run a speed test from the client to the server:

```
iperf3 -c <server-ip>
```

– Check the bandwidth and latency, and use mtr to identify network congestion:

```
mtr <destination-ip>
```

– You can also verify if network interfaces have mismatched speeds by using:

```
ethtool eth0
```

– If necessary, adjust the MTU or disable offloading features on the network interface.

6. Troubleshooting and Configuring Network Services

6.1 DHCP Troubleshooting (ISC and KEA)

- **Checking if DHCP is Running:**

```
sudo systemctl status isc-dhcp-server
```

or for KEA DHCP:

```
sudo systemctl status kea-dhcp4-server
```

- **Analyzing DHCP Logs:**

```
sudo tail -f /var/log/syslog | grep dhcp
```

  – Look for lease errors or configuration issues.

- **Configuring DHCP Server (ISC):**

File: /etc/dhcp/dhcpd.conf

```
subnet 192.168.1.0 netmask 255.255.255.0 {
  range 192.168.1.100 192.168.1.200;
  option routers 192.168.1.1;
  option domain-name-servers 8.8.8.8, 8.8.4.4;
}
```

- **Configuring DHCP Server (KEA DHCP - JSON Format):**

File: /etc/kea/kea-dhcp4.conf

```
    {
      "subnet4": [
        {
          "subnet": "192.168.1.0/24",
          "pools": [{ "pool": "192.168.1.100 -
192.168.1.200" }],
          "routers": ["192.168.1.1"],
          "option-data": [{ "name": "domain-name-servers",
"data": "8.8.8.8, 8.8.4.4" }]
        }
      ]
    }
```

Tip: Restart the service after changes:

```
    sudo systemctl restart isc-dhcp-server
```

or

```
    sudo systemctl restart kea-dhcp4-server
```

Best Practices for Network Troubleshooting

- **Check Physical Connections First:** Ensure cables, WiFi, and NICs are functioning.
- **Restart Networking Services:** Instead of rebooting the system, restart specific services.
- **Test Connectivity Step-by-Step:** Start with ping, then traceroute, and finally check logs.
- **Use Logs to Diagnose Issues:** Don’t just restart services—understand the root cause.
- **Secure Your Network:** Implement firewalls, SSH keys, and VPNs to protect your systems.