CSS 262: Linux Administration

Linux Networking Basics

Lecture 6: IP, DNS & Network Configuration

📋 Today's Agenda

Part 1: Networking Fundamentals

- TCP/IP stack overview
- Network interfaces & naming
- IP addressing (IPv4 & IPv6)
- Subnets & CIDR notation

Part 2: Configuration & Tools

- Static IP configuration
- NetworkManager & nmcli
- Netplan (Ubuntu)
- DNS configuration

Part 3: Troubleshooting

- Connectivity testing
- Routing & ARP
- Network diagnostics
- Security considerations

Part 4: Lab Focus

- Lab 6: IP & DNS Config
- Homework 2 submission
- Reliable connectivity setup

🎯 **Learning Objective:** Configure robust network settings (Static IP, DNS) to ensure reliable connectivity.

🔄 Quick Recap: Week 5

Bash Scripting & Automation

- **Scripts**: Shebang, variables, control flow, functions
- **Text Processing**: `grep`, `sed`, `awk`, pipelines
- **Automation**: Cron jobs, systemd timers
- **Best Practices**: Strict mode, quoting, security
- **Real-world**: Log analysis, backups, health checks

✅ You should now be able to automate system tasks with Bash scripts!

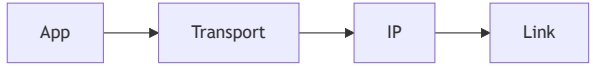
Part 1: Networking Fundamentals

Understanding the TCP/IP Stack

🌐 The TCP/IP Model

Layers (top → bottom):

1. Application – HTTP, SSH, DNS, SMTP
2. Transport – TCP/UDP, ports (22, 80, 443)
3. Internet – IP addresses, routing
4. Network Access – MAC, Ethernet/WiFi



🔌 Network Interfaces

Interface Naming Conventions

Prefix	Meaning	Example
`eth`	Ethernet (legacy)	`eth0`, `eth1`
`en`	Ethernet (predictable)	`enp0s3`, `ens33`
`wlan`	Wireless	`wlan0`
`lo`	Loopback	`lo` (127.0.0.1)
`virbr`	Virtual bridge	`virbr0`

View Interfaces

```
ip link show
ip addr show
# Legacy (deprecated)
ifconfig
```

📍 IP Addressing Basics

IPv4 (32-bit)

```
192.168.1.100
        └── Host
      └──── Subnet
    └────── Network
└────────── Class/Region
```

Private Ranges (RFC 1918)

Range	Use
10.0.0.0/8	Large networks
172.16.0.0/12	Medium networks
192.168.0.0/16	Home/Small

IPv6 (128-bit)

```
2001:0db8:85a3::8a2e:0370:7334
```

CIDR Notation

```
192.168.1.0/24  → 256 addresses
192.168.1.0/25  → 128 addresses
10.0.0.0/8      → 16M addresses
```

Key Addresses

- **Gateway**: Router (e.g., 192.168.1.1)
- **Broadcast**: 192.168.1.255 (for /24)
- **Loopback**: 127.0.0.1 (localhost)

🔀 Subnets & CIDR

Subnet Mask → CIDR

Mask	CIDR	Hosts
255.255.255.0	/24	254
255.255.255.128	/25	126
255.255.0.0	/16	65,534
255.0.0.0	/8	16M

Example: 192.168.1.100/24

- **Network**: 192.168.1.0 · **Broadcast**: 192.168.1.255
- **Usable**: 192.168.1.1 - 192.168.1.254 · **Gateway**: 192.168.1.1

Part 2: Configuration & Tools

Static IP, DNS & NetworkManager

⚙️ The `ip` Command (Modern)

View & Manage Interfaces

```
# List all interfaces
ip link show
ip addr show
ip a

# Add/remove IP address
sudo ip addr add 192.168.1.100/24 dev eth0
sudo ip addr del 192.168.1.100/24 dev eth0

# Bring interface up/down
sudo ip link set eth0 up
sudo ip link set eth0 down
```

Routing

```
ip route show
ip route add default via 192.168.1.1
```

⚠️ **Note:** `ip` changes are temporary – they reset on reboot. Use config files for persistence.

📡 NetworkManager & nmcli

NetworkManager

- Default on most modern Linux (Ubuntu, Fedora, RHEL)
- Handles wired, wireless, VPN
- Persistent configuration

nmcli – Command-Line Interface

```
# List connections
nmcli connection show

# Show device status
nmcli device status

# Add static IP connection
nmcli connection add type ethernet con-name "Static-LAN" \
  ifname eth0 ipv4.addresses 192.168.1.100/24 \
  ipv4.gateway 192.168.1.1 ipv4.dns "8.8.8.8 8.8.4.4" \
  ipv4.method manual
```

📡 nmcli: Modify & Activate

Modify Existing Connection

```
# Change to static IP
nmcli connection modify "Static-LAN" ipv4.addresses 192.168.1.100/24
nmcli connection modify "Static-LAN" ipv4.gateway 192.168.1.1
nmcli connection modify "Static-LAN" ipv4.dns "8.8.8.8 8.8.4.4"
nmcli connection modify "Static-LAN" ipv4.method manual

# Or DHCP
nmcli connection modify "Static-LAN" ipv4.method auto
```

Activate Connection

```
nmcli connection up "Static-LAN"
nmcli connection down "Static-LAN"
```

Interactive TUI

```
nmtui
```

📄 Netplan (Ubuntu 18.04+)

YAML Configuration

Netplan reads `/etc/netplan/*.yaml` and generates config for NetworkManager or systemd-networkd.

```yaml
# /etc/netplan/01-netcfg.yaml
network:
  version: 2
  ethernets:
    enp0s3:
      addresses:
        - 192.168.1.100/24
      gateway4: 192.168.1.1
      nameservers:
        addresses:
          - 8.8.8.8
          - 8.8.4.4
```

Apply Changes

```bash
sudo netplan apply
sudo netplan try   # Rollback if connection lost
```

📄 Netplan: DHCP Example

DHCP Configuration

```yaml
# /etc/netplan/01-netcfg.yaml
network:
  version: 2
  ethernets:
    enp0s3:
      dhcp4: true
      dhcp4-overrides:
        use-dns: true
        use-routes: true
```

Multiple Interfaces

```yaml
network:
  version: 2
  ethernets:
    enp0s3:
      dhcp4: true
    enp0s8:
      addresses:
        - 10.0.0.10/24
      gateway4: 10.0.0.1
```

🌐 DNS Configuration

What is DNS?

- **Domain Name System**: Resolves hostnames → IP
- **Example**: `google.com` → `142.250.185.46`

Configuration Files

```
# Primary config
/etc/resolv.conf

# Example content:
nameserver 8.8.8.8
nameserver 8.8.4.4
search example.local
```

systemd-resolved (Modern)

```
# Often manages resolv.conf
ls -l /etc/resolv.conf   # May be symlink
resolvectl status
```

Testing DNS

```
# Resolve hostname
nslookup google.com
dig google.com
host google.com

# Using getent
getent hosts google.com
```

Common Public DNS

Provider	Primary	Secondary
Google	8.8.8.8	8.8.4.4
Cloudflare	1.1.1.1	1.0.0.1
Quad9	9.9.9.9	149.112.112.112

📝 Traditional Config: Debian/Ubuntu

`/etc/network/interfaces` (legacy, ifupdown)

```
# Loopback
auto lo
iface lo inet loopback

# Static IP
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

# DHCP
iface eth0 inet dhcp
```

Apply

```
sudo systemctl restart networking
# Or
sudo ifup eth0
sudo ifdown eth0
```

Part 3: Troubleshooting

Connectivity & Diagnostics

🔍 Connectivity Testing

ping – Test Reachability

```
ping -c 4 8.8.8.8
ping -c 4 google.com
ping -c 4 192.168.1.1
```

traceroute – Path Discovery

```
traceroute google.com
traceroute -n 8.8.8.8   # No DNS lookup
# Or
tracepath google.com
```

Check Port Connectivity

```
nc -zv 192.168.1.1 22    # Test SSH port
telnet 192.168.1.1 80    # Test HTTP (if installed)
curl -v http://example.com
```

📊 Network Diagnostics

View Current Config

```
ip addr show
ip route show
ip link show
```

DNS Resolution

```
dig google.com
dig @8.8.8.8 google.com
nslookup google.com
```

ARP Table

```
ip neigh show
arp -a
```

Packet Capture (Advanced)

```
sudo tcpdump -i eth0 -c 10
sudo tcpdump -i eth0 port 22
```

Listening Ports

```
ss -tuln
netstat -tuln
```

Network Speed Test

```
iperf3 -s    # Server
iperf3 -c server_ip  # Client
```

Routing

Default Route

```
# Show default gateway
ip route show default
route -n

# Add default route (temporary)
sudo ip route add default via 192.168.1.1
```

Static Routes

```
# Add route to specific network
sudo ip route add 10.0.0.0/8 via 192.168.1.254

# Delete route
sudo ip route del 10.0.0.0/8
```

Routing Table

```
ip route show
# default via 192.168.1.1 dev eth0
# 192.168.1.0/24 dev eth0 proto kernel scope link src 192.168.1.100
```

🔧 Troubleshooting Flowchart

Decision flow:

1. Can't reach internet → Can you `ping` gateway?
2. No → Check IP config, cable, interface
3. Yes → Can you `ping 8.8.8.8` ?
4. No → Check routing, firewall
5. Yes → Can you resolve `google.com` ?
6. No → Check DNS config
7. Yes → Connectivity OK ✓

🔧 Troubleshooting: Quick Checks

1. `ip addr` – Do I have an IP?
2. `ping <gateway>` – Is gateway reachable?
3. `ping 8.8.8.8` – Is internet reachable?
4. `ping google.com` – Is DNS working?

Network Security Basics

Server Hardening

- **Minimize listening services**: `ss -tuln`
- **Disable unused interfaces**
- **Use static IPs** for servers (predictable)
- **Restrict SSH** to specific IPs (later weeks)

DNS Security

- Use trusted DNS (avoid MITM)
- Consider DNS over HTTPS (DoH)
- Validate certificates: `openssl s_client`

Information Disclosure

```
# Don't expose internal hostnames
# Use /etc/hosts for internal resolution
echo "192.168.1.10  internal-db" >> /etc/hosts
```

Network Namespaces (Advanced)

- Isolate network per process
- Used by containers (Docker)

📊 Summary: Linux Networking

Key Concepts Covered

1. **TCP/IP Stack**: Layers, ports, IPs
2. **Interfaces**: Naming, `ip link`, `ip addr`
3. **IP Addressing**: IPv4, CIDR, private ranges
4. **Configuration**: nmcli, Netplan, interfaces
5. **DNS**: resolv.conf, resolution tools
6. **Troubleshooting**: ping, traceroute, ss
7. **Routing**: Default gateway, static routes

Configuration Tools by Distro

Distro	Tool
Ubuntu 18+	Netplan
Fedora/RHEL	nmcli / NetworkManager
Debian (legacy)	/etc/network/interfaces
Arch	systemd-networkd / NetworkManager

🎯 Learning Objectives: Did We Achieve?

By now, you should be able to:

- ✅ Explain the TCP/IP stack and interface naming
- ✅ Configure static IP addresses
- ✅ Use nmcli and Netplan for network config
- ✅ Configure DNS resolution
- ✅ Troubleshoot connectivity (ping, traceroute)
- ✅ View routing tables and ARP
- ✅ Apply basic network security practices

🎓 **Next Week:** Package Management & Repositories – Software compilation, Lab 7, Quiz 1!

🧪 Lab Practice: IP & DNS Config

Exercise 1: Static IP

Configure VM with static IP (192.168.1.100/24), gateway, DNS. Verify with `ip addr`, `ping`, `nslookup`.

Exercise 2: Dual interface

Add second interface (NAT + Host-only). Configure one DHCP, one static. Test both.

Exercise 3: DNS troubleshooting

Break DNS (wrong nameserver), diagnose with `dig` / `nslookup`, fix and verify.

Exercise 4: Connectivity script

Script checks: interface, route, ping gateway, ping 8.8.8.8, DNS. Output one-line status per check.

🔗 Additional Resources

Documentation

- ip(8) man page - Full ip command reference
- Netplan documentation - Ubuntu network config
- NetworkManager nmcli - nmcli reference

Books & Guides

- *"UNIX and Linux System Administration Handbook"* - Networking chapter
- Arch Wiki: Network configuration
- Red Hat: Configuring network connections

Practice

- Set up a lab with multiple VMs and static IPs
- Practice netplan/nmcli on different distros

Questions?

Next: Package Management & Repositories