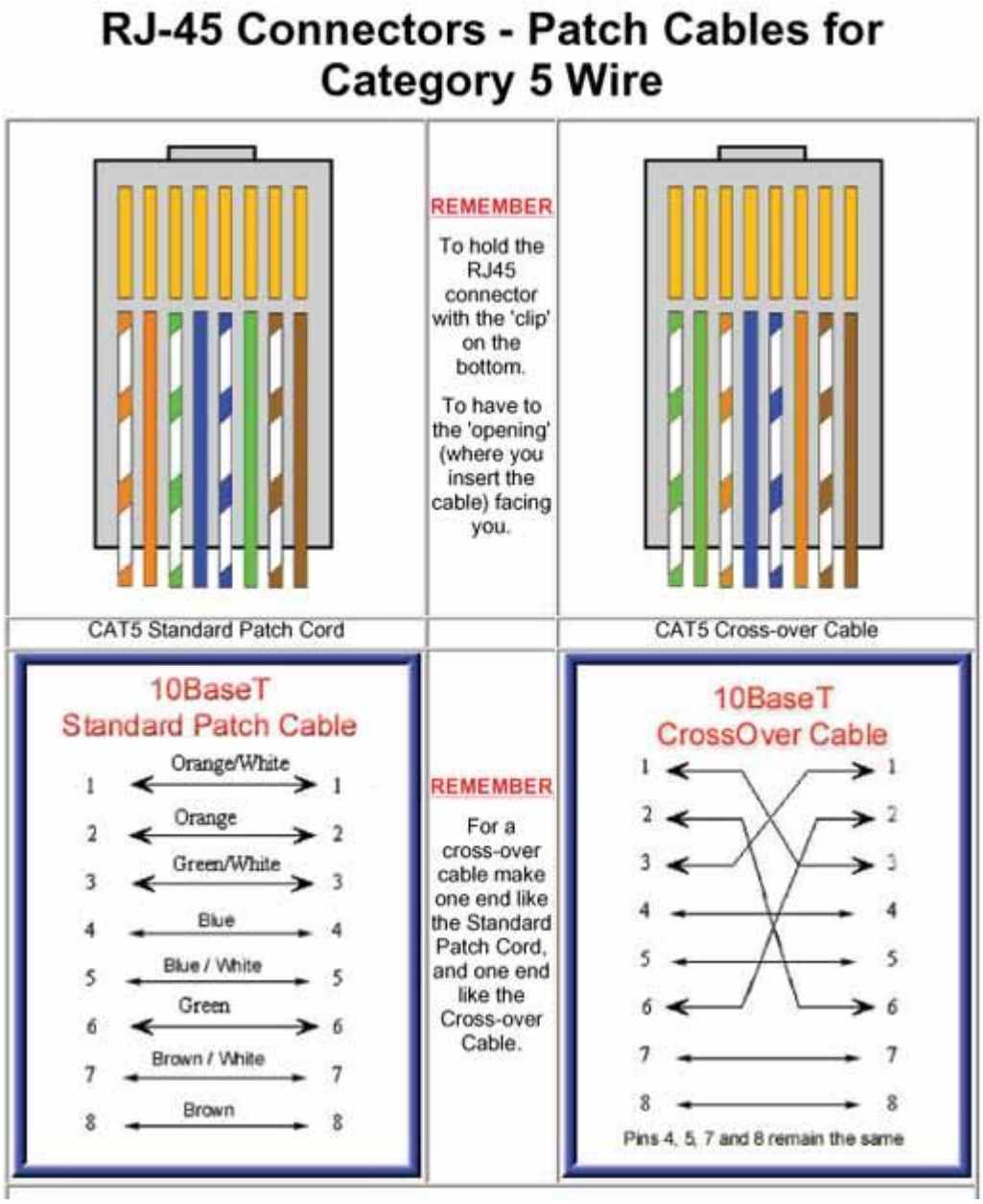
COMPUTER NETWORKS Lab 1

Prepare Ethernet Cable and check connectivity

Equipment Used: Cat5/6 Cable, Crimping Tool, RJ45 Connectors, Cable Tester

Color Coding Scheme Image Source:

https://i.pinimg.com/736x/04/e3/1a/04e31aa47bbdc1c4cbb1208437ceaf87--a-website-crossover.jpg



Check connectivity among nodes using different configurations

Private IP Address Ranges:

- 10.0.0.0/8 IP addresses: 10.0.0.0 -- 10.255.255.255.
- 172.16.0.0/12 IP addresses: 172.16.0.0 -- 172.31.255.255.
- 192.168.0.0/16 IP addresses: 192.168.0.0 – 192.168.255.255

Equipment Used: 8-port switch, Ethernet cables, USB interface based Ethernet NIC cards
Operating System Ubuntu 16.04 LTS

Task 1:
Connect two nodes to a switch
Configure their IP address in a same subnet (specify static IP address and subnet mask using GUI)
Check connectivity among nodes using ping

Task 2:
Configure IP addresses of two nodes in different subnet (specify static IP address and subnet mask using GUI)
Check connectivity among nodes using ping.

Task 4:
Configure IP addresses of three nodes in the following manner

- Node 1: 10.100.1.1/8, Default Gateway 10.100.1.2
- Node 2:
  - NIC 1 10.100.1.2/8
  - NIC 2 192.168.1.2/24
- Node 3: 192.168.1.1/24, Default Gateway 192.168.1.2

Check connectivity among nodes using ping:

- Node 1 to Node2 (10.100.1.2)
- Node 1 to Node2 (192.168.1.2/24)

- Node 3 to Node2 (192.168.1.2/24)
- Node 3 to Node2 (10.100.1.2)

- Node 1 to Node 3

Try to reason about the obtained results

On Node 2, open the following configuration file /etc/sysctl.conf and enable IP Forwarding (by removing the comment from the relevant line). Following is the relevant description extracted from http://www.linuxhomenetworking.com/wiki/index.php/Quick_HOWTO_:_Ch03_:_Linux_Networking#Configuring_IP_Forwarding

For your Linux server to become a router, you have to enable **packet forwarding**. In simple terms packet forwarding enables packets to flow through the Linux box from one network to another. The Linux kernel configuration parameter to activate this is named `net.ipv4.ip_forward` and can be found in the file `/etc/sysctl.conf`. Remove the "#" from the line related to packet forwarding.

Before:

```
# Disables packet forwarding
net.ipv4.ip_forward=0
```

After:

```
# Enables packet forwarding
net.ipv4.ip_forward=1
```

This enables packet forwarding only when you reboot at which time Linux will create a file in one of the subdirectories of the special RAM memory-based `/proc` filesystem. To activate the feature immediately you have to force Linux to read the `/etc/sysctl.conf` file with the sysctl command using the `-p` switch. Here is how it's done:

```
[root@bigboy tmp] sysctl -p
sysctl -p
net.ipv4.ip_forward = 1
net.ipv4.conf.default.rp_filter = 1
kernel.sysrq = 0
kernel.core_uses_pid = 1
[root@bigboy tmp]#
```

Restart the network service of Node 2 (“sysctl -p <file>” or **restart the system**) and check connectivity between Node 1 to Node 3

Use route command to see the routing table

External Link: http://www.yourownlinux.com/2013/07/how-to-configure-ubuntu-as-router.html. You may find other similar documentation

Configure multiple routes
Node 1: IP 10.100.1.1, default gateway 10.100.1.2, alternate gateway 172.16.1.2
Node 2: IP 10.100.1.2, 192.168.1.2, 172.16.1.2 (three NIC)
Node 3: IP 192.168.1.3, default gateway 192.168.1.2, alternate gateway 172.16.1.2

Check connectivity between node 1 and 3 by disabling one of the gateway.
Use Traceroute to reason about connectivity

Set up

Node 1: IP 10.100.1.1, default gateway 10.100.1.2
Node 2: IP 10.100.1.2, 192.168.1.2, 172.16.1.2 (three NIC)
Node 3: IP 192.168.1.3, 172.16.1.3 (two NIC); default gateway 192.168.1.2

Check connectivity between node 1 and 3 (both interfaces) and reason about the results.

As there can be one default gateway, we can ping only one of the interface of Node 3 from Node 1. We observe that the messages arrive at the Node 3 from Node 1 but are not echoed back to Node 1 from one of the interfaces. To ping both the interfaces of Node 3 from Node 1, we need to have additional routing table and associated routing table entry at Node 3 for a route to Node 1. Following information is taken from https://www.thomas-krenn.com/en/wiki/Two_Default_Gateways_on_One_System

Problem Description

You have built two or more network cards into one Linux system and each of these cards has its own default gateway. By default, you can only have one default gateway on a system. The case described would lead to asynchronous routing, whereby the router would reject the packets as appropriate.

Solution

The iproute2 program, which is included in all current Linux distributions and already installed even, as a rule, can be used for the solution of this problem. Normally, a Linux system only has one routing table, in which only one default gateway can make entries. With iproute2, you have the ability to setup an additional routing table, for one thing, and allow this table to be used by the system based on rules, for another.

Initial Position

We will assume that we have two interfaces: eth0 and eth1. The two networks that should be used are 192.168.0.0/24 and 172.16.0.0/16. The gateway IP addresses are 192.168.1.2 and 172.16.1.2 for the netoworks 192.168.0.0/24 and 172.16.0.0/24, respectively. Under Debian/Ubuntu, the initial configuration would appear as follows. /etc/network/interfaces

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).
# The loopback network interface

auto lo
iface lo inet loopback

# The primary network interface

allow-hotplug eth0
iface eth0 inet static
    address 192.168.1.3
    netmask 255.255.255.0
    gateway 192.168.1.2

# The secondary network interface
allow-hotplug eth1
iface eth1 inet static
    address 172.16.1.3
    netmask 255.255.0.0
```

Adding a Second Routing Table

To add a new routing table, the file, /etc/iproute2/rt_tables must be edited. We will call the routing table "rt2" and set its preference to 1. The named file should then appear as follows.

```
#
# reserved values
#
255     local
254     main
253     default
0       unspec
#
# local
#
#1      inr.ruhep
1 rt2
```

Configuring the New Routing Table

From this point, four commands are needed to achieve our goal. First, the new routing table needs to be populated, which is done using the following command.

```
ip route add 172.16.0.0/16 dev eth1 src 172.16.1.3 table rt2
ip route add default via 172.16.1.2 dev eth1 table rt2
```

The first command says that the network, 172.16.0.0/16, can be reached through the eth1 interface. The second command sets the default gateway.

Routing Rules

So that the system knows when to use our new routing table, two rules must be configured.

```
ip rule add from 172.16.1.3/32 table rt2
ip rule add to 172.16.1.3/32 table rt2
```

These rules say that both traffic from the IP address, 172.16.1.3, as well as traffic directed to or through this IP address, should use the rt2 routing table.

Making the Configuration permanent

The ip rule and ip route commands will become invalid after a re-boot, for which reason they should become part of a script (for example, /etc/rc.local) that will be executed once the network has been started after booting. For Debian, these command can also be written directly into the /etc/network/interfaces file, which would then appear as follows.

```
iface eth1 inet static
   address 172.16.1.3
   netmask 255.255.0.0
   post-up ip route add 172.16.0.0/16 dev eth1 src 172.16.1.3 table rt2
```

```
post-up ip route add default via 172.16.1.2 dev eth1 table rt2
post-up ip rule add from 172.16.1.3/32 table rt2
post-up ip rule add to 172.16.1.3/32 table rt2
```

More than Two Network Cards or Gateways

If there are more than two networks, a routing table can be created for each additional network analogous to the example presented above.

Testing the Configuration

The following commands can be used to ensure that the rules as well as the routing entries are working as expected.

```
ip route list table rt2
ip rule show
```