Computer Networks

Redes de Computadores

Bruno Carvalho
up201606517

João Malheiro
up201605926

Carlos Daniel Gomes
up201603404

December 22, 2018

Contents

Summary

Introduction

Part 1: Download application

Part 2: Network

Configure an IP Network

We start by creating a private network 172.16.30.0/24 for computers *tux31* and *tux34*, connecting directly their interfaces `tux31@eth0` and `tux34@eth0`. Interface `tux31@eth0` is given IP address 172.16.30.1/24, and `tux34@eth0` is given 172.16.30.254/24. We test the connectivity of *tux31* and *tux34* with a simple `ping` command.

Cable configuration:
`tux31@eth0`—`tux34@eth0`

Commands:
tux31:
```
ifconfig eth0 up
ifconfig eth0 172.16.30.1/24
route add default gw 172.16.30.254
```
tux34:
```
ifconfig eth0 up
ifconfig eth0 172.16.30.254/24
```

Questions

What are the ARP packets and what are they used for? ARP request packets are broadcast in a network by a host to retrieve the MAC address of the machine with a certain IPv4 address. The protocol specifies the packet is sent to every host machine in the network, and only the one with the requested IPv4 address responds with its MAC address. In other words, the ARP protocol serves to convert 32-bit logical IP addresses to 48-bit physical MAC addresses.

What are the MAC and IP addresses of ARP packets and why? The ARP request packet includes the IP and MAC addresses of its source host and the IP address of its target host, whose MAC address it wants to retrieve. The ARP reply includes all four addresses. For example, *tux31* sent an ARP request for 172.16.30.254 on network 172.16.30.0/24, which reached `tux34@eth0` (the only other host on the network). Its ARP reply was MAC address `00:21:5a:5a:7d:7a`.

What packets does the ping command generate? The `ping` commands generates ICMP `ECHO_REQUEST` packets, and expects ICMP `ECHO_REPLY` responses.

What are the MAC and IP addresses of the ping packets? The source IP is 172.16.30.1 (`tux31@eth0`) and the destination IP is 172.16.30.254 (`tux34@eth0`) for the `ECHO_REQUEST` packets, and the reverse of `ECHO_REPLY` packets. Since the packets are directly transmitted, the source and destination MAC addresses match the IP addresses.

How to determine if a receiving Ethernet frame is ARP, IP, ICMP? For Ethernet frames, check the EtherType header field on the MAC frame: `0x0800` for IPv4 and `0x0806` for ARP. Inside the IP packet, the protocol header field indicates the upper layer protocol: 1 for ICMP, 6 for TCP, 17 for UDP, etc.

00:0f:fe:8b:e4…	ff:ff:ff:ff:ff:ff	ARP	Who has 172.16.30.254? Tell 172.16.30.1
00:21:5a:5a:7d…	00:0f:fe:8b:e4:4d	ARP	172.16.30.254 is at 00:21:5a:5a:7d:74
172.16.30.1	172.16.30.254	ICMP	Echo (ping) request id=0x39cc, seq=1/256, ttl=64
172.16.30.254	172.16.30.1	ICMP	Echo (ping) reply id=0x39cc, seq=1/256, ttl=64
172.16.30.1	172.16.30.254	ICMP	Echo (ping) request id=0x39cc, seq=2/512, ttl=64
172.16.30.254	172.16.30.1	ICMP	Echo (ping) reply id=0x39cc, seq=2/512, ttl=64
172.16.30.1	172.16.30.254	ICMP	Echo (ping) request id=0x39cc, seq=3/768, ttl=64
172.16.30.254	172.16.30.1	ICMP	Echo (ping) reply id=0x39cc, seq=3/768, ttl=64
172.16.30.1	172.16.30.254	ICMP	Echo (ping) request id=0x39cc, seq=4/1024, ttl=64
172.16.30.254	172.16.30.1	ICMP	Echo (ping) reply id=0x39cc, seq=4/1024, ttl=64
172.16.30.1	172.16.30.254	ICMP	Echo (ping) request id=0x39cc, seq=5/1280, ttl=64
172.16.30.254	172.16.30.1	ICMP	Echo (ping) reply id=0x39cc, seq=5/1280, ttl=64

Figure 1: *tux31* pings *tux34*

How to determine the length of a receiving frame? For Ethernet frames there is no header field with the frame length — the whole frame must be read and the measured. For IPv4 packets, the header has two length fields: one for header length (4bits, offset 4 bits) and another for packet length (2 bytes, offset 16 bits).

What is the loopback interface and why is it important? 172.0.0.0/8 is a range of 2^{24} IPv4 addresses representing the host machine. Useful for testing or running client-server services in the host. It is generally not represented in the routing table. The address 172.0.0.1 is assigned the local loopback interface `lo`, but the whole range is private and may be used.

Implement two virtual LANs in a switch

Now we are going to create a second private network 172.16.31.0/24 for *tux32*, and connect our two networks to two virtual LANs in the switch: vlan 30 for network 172.16.30.0/24 and vlan 31 for network 172.16.31.0/24. Naturally, we connect `tux31@eth0` and `tux34@eth0` to vlan 30 and `tux32@eth0` to vlan 31. Notice there is no connection between the networks yet, so *tux32* cannot communicate with neither *tux31* or *tux34*.

Cable configuration:
```
tux31@eth0—sw Fa0/1
tux34@eth0—sw Fa0/4
tux32@eth0—sw Fa0/2
```

Commands:
tux32:
```
ifconfig eth0 up
ifconfig eth0 172.16.31.1/24
```
switch:
```
configure terminal
vlan 30
exit
vlan 31
exit
interface fastethernet 0/1
switchport mode access
switchport access vlan 30
exit
interface fastethernet 0/4
switchport mode access
switchport access vlan 30
exit
interface fastethernet 0/2
switchport mode access
switchport access vlan 31
end
```

Questions

How to configure vlan 30? On the right.

How many broadcast domains are there? How can you conclude it from the logs? The two broadcast domains are the two networks 172.16.30.0/24 and 172.16.31.0/24 because they are not connected.

Configure a Router in Linux

In this configuration we enable a new interface `tux34@eth2` and connect it to network 172.16.31.0/24 through vlan 31. This way *tux34* connects the two networks, and enabling IP forwarding allows *tux31* and *tux32* to communicate.

Cable configuration:
```
tux31@eth0—sw Fa0/1
tux34@eth0—sw Fa0/4
tux34@eth2—sw Fa0/7
tux32@eth0—sw Fa0/2
```

Commands:
tux34:
```
ifconfig eth2 up
ifconfig eth2 172.16.31.253/24
echo 1 > /.../ip_forward
echo 0 > /.../icmp_echo_ignore_broadcasts
```
tux32:
```
route add -net 172.16.30.0/24 gw 172.16.31.253
```
switch:
```
configure terminal
interface fastethernet 0/7
switchport mode access
switchport access vlan 41
end
```

Questions

What routes are there in the tuxes? What are their meaning? The routes listed by command `ip route` are as follows:

- *tux31*: `default via 172.16.30.254 dev eth0`
  Default gateway of *tux31*. If no other route matches the destination of an IP packet being sent, it is sent to 172.16.30.254 (through interface `eth0`).
- *tux31*: `172.16.30.0/24 dev eth0`
  Means *tux31* is directly connected to all hosts in the network 172.16.30.0/24 through network interface `eth0`.

- *tux34*: `172.16.30.0/24 dev eth0`
  Means *tux34* is directly connected to network 172.16.30.0/24 through interface `eth0`.
- *tux34*: `172.16.31.0/24 dev eth2`
  Means *tux34* is directly connected to network 172.16.31.0/24 through interface `eth2`.
- *tux32*: `172.16.30.0/24 via 172.16.31.253 dev eth0`
  Means that *tux32* is (indirectly) connected to the network 172.16.30.0/24 through gateway 172.16.31.253. So any IP packet whose destination is the network 172.16.30.0/24 will be sent to address 172.16.31.253 (who *tux32* expects to forward).
- *tux32*: `172.16.31.0/24 dev eth0`
  Means *tux32* is directly connected to network 172.16.31.0/24 through interface `eth0`.

What information does an entry of the forwarding table contain? The primary information are the *Destination* (host or networks) and the *Gateway*. Packets destined to a certain *Destination* are routed to the respective *Gateway*. For gateway entries, the host is not directly connected to the *Destination* but it always directly connected to the *Gateway*.

The table displayed by `route -n` shows further information. *Metric* scores a given route in terms of cost. It can contain any number of values that help a router or host determine the best route among multiple routes to a destination. A packet will generally be sent through the route with the lowest metric. The most basic metric is typically based on path length, hop count or delay. Some *Flags* are U for Up, meaning the route is up; G for Gateway, meaning the route is to a gateway; H for host, meaning the route's destination is a complete host address; D means the route was created by a redirect; and M means the route was modified by a redirect. *Iface* is the network interface used for the route, naturally.

What ARP messages, and associated MAC addresses, are observed and why? ARP Request/Reply pairs which are required for IP communication:
- *tux31* (172.16.30.1) asks MAC address of 172.16.30.254 (*tux34*)
- *tux34* (172.16.31.253) asks MAC address of 172.16.31.1 (*tux32*)

When the ARP table is clear, the link layer needs to request the MAC address of the destination IP with an ARP request.

The observed MAC addresses are then:
- `tux31@eth0` (172.16.30.1): `00:0f:fe:8b:e4:4d`
- `tux34@eth0` (172.16.30.254): `00:21:5a:5a:7d:74`
- `tux34@eth2` (172.16.31.253): `00:01:02:21:83:0e`
- `tux32@eth0` (172.16.31.1): `00:21:5a:61:30:63`

What ICMP packets are observed and why? When *tux31* pings *tux32*, ICMP `ECHO_REQUEST` packets are sent with source 172.16.30.1 and destination 172.16.31.1. These are routed from `tux31@eth0` through *tux34* to `tux32@eth0`, and back for the `ECHO_REPLY` packets. These packets can be observed in both of *tux34*'s interfaces.

What are the IP and MAC addresses associated to ICMP packets and why? Since ICMP packets are IPv4 packets – they live in the network layer – their IP addresses are fixed. When *tux31* pings *tux32*, the source IP is 172.16.30.1 and destination IP is 172.16.31.1. However, the MAC addresses reflect the hops the packet takes through the network. From *tux31* to *tux34*, the source MAC address is that of `tux31@eth0` and the destination MAC address is that of `tux34@eth0`. From *tux34* to *tux32*, the source MAC address is that of `tux34@eth2` and the destination MAC address is that of `tux32@eth0`. For the `ECHO_REPLY` packets it's the reverse, naturally.

Configure a Commercial Router and Implement NAT

To allow both networks to communicate with the Internet we connect one of the router's interfaces to network 172.16.31.0/24 and implement NAT, with allowance for *tux31* and *tux32*.

Cable configuration:
`tux31@eth0—sw Fa0/1`
`tux34@eth0—sw Fa0/4`
`tux34@eth2—sw Fa0/7`
`tux32@eth0—sw Fa0/2`
`rt Giga0/1—sw Fa0/9`
`rt Giga0/0—172.16.1.254`

Commands:
tux44:
`route add default gw 172.16.31.254`
tux42:
`route add default gw 172.16.31.254`
switch:
```
configure terminal
interface fastethernet 0/9
switchport mode access
switchport access vlan 31
end
```

Questions

How to configure a static route in a commercial router?
The command's basic syntax is

```
ip route DestinationIP Mask GatewayIP
```

What are the paths followed by the packets in the experiments carried out and why?

- *tux31* pings 172.16.30.254 at *tux34*:
  172.16.30.1(*tux31*) → 172.16.30.254(*tux34*)
- *tux31* pings 172.16.31.253 at *tux34*:
  172.16.30.1(*tux31*) → 172.16.30.254(*tux34*)
- *tux31* pings 172.16.31.1 at *tux32*:
  172.16.30.1(*tux31*) → 172.16.30.254(*tux34*) → 172.16.31.1(*tux32*)
- *tux31* pings 172.16.31.254 at *router*:
  172.16.30.1(*tux31*) → 172.16.30.254(*tux34*) → 172.16.31.254(*rt*)
- *tux31* pings 172.16.1.39 at *router*:
  172.16.30.1(*tux31*) → 172.16.30.254(*tux34*) → 172.16.31.254(*rt*)
- *tux31* pings 172.16.1.254 (with NAT implemented in the router):
  172.16.30.1(*tux31*) → 172.16.30.254(*tux34*) → 172.16.31.254(*rt*) → 172.16.1.254(*lab*)

How to configure NAT in a commercial router?

What does NAT do?

DNS

TCP connections

Conclusion

References