Presentation on TCP/IP

By Te-cheng Hsu 20150713
Dept. of Electrical Engineering, NTHU

Outline

A. Origins and Backgrounds
B. Open System Interconnection (OSI)
C. Internet Protocol (IP)
D. Topics related to IP
E. Transmission Control Protocol (TCP)

Proscend IOT Team Presentation 20150708

Origins and Backgrounds

TCP/IP became a hot issue was because of e-mail and WWW service during 1980’s and 1990’s, respectively.

World Wide Web (WWW): Server-Client Structure
In 1980’s, trying to improve science research conduct

Origins and Backgrounds

HTTPd (HTTP deamon) (Server & Client)
⇒ Netscape (Client)
⇒ Apache (A Patch Server), dominates after 1996

Internet Technology: “Individually”
e.g. Ethernet & Token Ring (IBM)
⇒ APARNET, cooperating with Berkeley
implanting TCP/IP developed in 1980’s
into BSD Unix computers

IEEE standard : Ethernet & Internet

Open System Interconnection (OSI)

Before dwelling into this subject,
we first see what comprises the web:

1 node: devices with IP
2 server: forming responses
3 client (workstation)
4 router/gateway: connection of
  two different "groups" devices

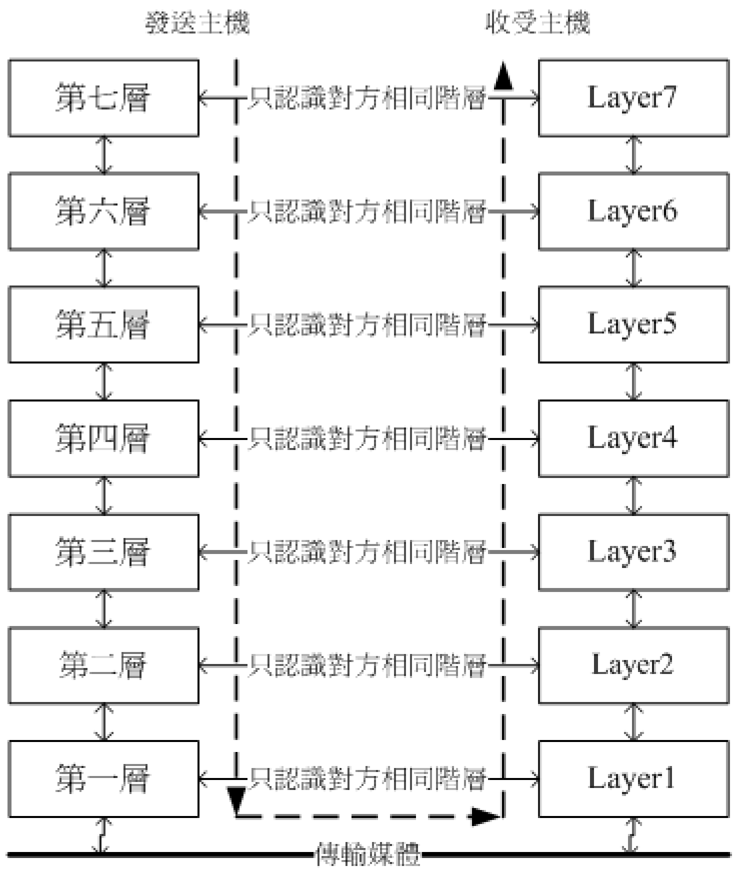
Open System Interconnection (OSI)

Now let's see what's in OSI:

By definition:
  closer to HW: layer 1
  closer to SW: layer 7

No matter Tx/Rx, they know only the same layer of each other

Package flow:
  (Tx) 7~1 => (Rx) 1~7



Open System Interconnection (OSI)

Layer 1 Physical Layer

binary decode/encode, Tx/Rx

Layer 2 Data Link Layer

(a)closer to HW:

MAC(Media Access Control): packet type

(b)clsoer to SW:

LLC(Logical Link Control): managing segments from upper layers, transforming to MAC form

Layer 3 Network Layer

1 Definition of IP (Internet Protocol)

2 Defining connections between computers (create/stop/maintain)

3 Concept of route

Open System Interconnection (OSI)

Layer 4 Transport Layer
connection technology (TCP, UDP) for Rx/Tx to ensure all segments can reach target correctly; packaging segments (big->small)

Layer 5 Session Layer
1 Ensure Internet connection creation
2 Define connection/disconnection between two ports (connection between two computers)

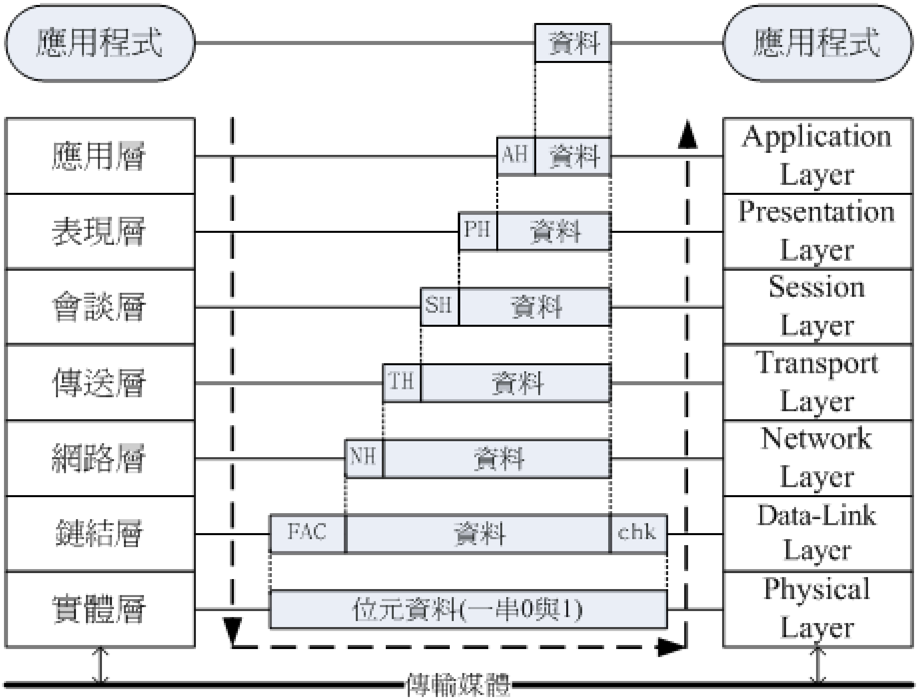
Open System Interconnection (OSI)

Layer 6 Presentation Layer

transform formats of typical applications into standard Internet formats, defining transforms between Internet ervices/programs

Layer 7 Application Layer

defining how apps can receive/transmit to apps and final representation to the users; archive and encryption=> speed and Security; encoding/ decoding



Why TCP/IP?

WHY?
Since OSI is to "rigorous", it's hard to write programs based on this interface
=> blocking further developments
=> simplifying OSI into only 4 layers

1990's email and www
=> current Internet Networks

Internet Protocol (IP)

IP abstraction (from OSI:7 to IP:4)

1 Network Access Layer

    data->device

2 Internet Layer

    adding IP headers, routing information

3 Transport Layer

    point-to-point connection

    control dataflow

    ensure data to destination correctly

  => TCP/UDP

4 Application Layer

    interface between applications,

    providing connections needed

Internet Protocol (IP)

IP packaging

32bits(IPv4); 128bits (IPv6)

32/8=4 partitions e.g. 192.168.1.3
(analogy to broadcast in streets)

InterNIC: 5 classes of IPs

Class A: 0.xxx.xxx.xxx
Class B:128.xxx.xxx.xxx
Class C:192.xxx.xxx.xxx
Class D:224.xxx.xxx.xxx
Class E:240.xxx.xxx.xxx

Note. InterNIC (Network Information Center)

Internet Protocol (IP)

in IPv4:

1 Public IP: connect to Internet

2 Private IP: LAN connections
(private IP is also for the fear that IPv4 is not enough)

Internet Protocol (IP)

However, only "Classes" will be inefficient
(think of broadcasting)

⇒ Netmask (Subnet Mask):
25-bit NetID & 7-bit HostID

Class A:255.0.0.0
Class B:255.255.0.0
Class C:255.255.255.0

(.0: server; .255: broadcast)

Internet Protocol (IP)

when a pair of (IP,Netmask) is achieved
=> AND operation=> NetID

This can be seen as a smart way to use IP address more efficiently, and two cases are considered in the following:

- when the network domain contains too many IPs
- when the network domain is too small

Internet Protocol (IP)

Case 1: too large network domain

Transfer NetID bits to HostID to divide the IPs into many small groups, nut with less amount of HostIPs (Ethernet connects to around 1200 clients/PCs at a time)

We should save at least 2 bits for HostID and we'll leave only 2 IPs available (00 for server and 11 for broadcast) Note in LAN applications we need to save at least 3 bits

Internet Protocol (IP)

Case 2: too small network domain

We can use Classless Inter-Domain Routing (CIDR, 不分級IP) to develop “Supernet”
The netmask length is arbitrary (VLSM)

If we have an IP like this:
192.168.1.1/18 then
- 3 supernets: 192.168.1.1/21
- 3 subnets: 192.168.1.1/15

Ethernet Protocol: CSMA/CD

CSMA/CD(IEEE 802.3 standard):
Carrier Sense Multiple Access with Collision Detection

card: hardware address
transition: card-to-card
=> no. == MAC (not able to modify)

Ethernet Protocol: CSMA/CD

1 Carrier Sense:
  before sending packets, make sure nobody's using it

2 Multiple Access:
  all machines connected to the hub can reach the data => encryption

3 Collision Detection:
  no data sending at the same time, retry if collided

Ethernet Protocol: CSMA/CD

A large packet -> a few small packets
(1500B per time) in MAC format

46~1500B (64B for collision detection)
(<46B, add some additional bits)
=> Maximum Transmission Unit (MTU)

Internet Protocol (IP)

Address Resolution Protocol (ARP) 網路位址解析
&& Reverse ARP (RARP)

=> ARP table: recording MAC
(last for 20 minutes)

ICMP (enclosed in IP packets) 網際網路訊息控制協定
error detection/report, ensuring connection status and connection correctness

Internet Protocol (IP)

Address Resolution Protocol (ARP)

1 Every client will build an ARP form in ARP cache, recording IP and hardware address (MAC) (with a finite life time)
2 Broadcast ARP packet if IP's hardware address is NOT found in the table
3 If IP's hardware address is correct, take the request and save/update the ARP table and reply
4 Update table (server)=> start transmission
5 Sending end don't get ARP reply: failed

Internet Protocol (IP)

Internet Control Message Protocol (ICMP)

Main functions: (covered by HW mostly)
- server exist?
- connection; take care of router data
- redirection
- data flow control

connection status: (Type, Code)

Note. about PING in Linux
- echo-request(type8), echo-reply(type0)

Domain Name System (DNS)

machine name to machine IP

a full set of settings to connect to Internet
IP, Netmask, Network, Broadcast, Gateway, DNS
=> only IP, Netmask, Default Gateway, DNS

If we have multiple DNS servers, then the retrieved IP is randomly chosen from one of the DNS servers=> important to synchronize all servers: Master v.s. Slave

- Master: renew by user (by hands)
- Slave: synchronization or noticing larger serial number (means newer)

Dynamic Host Configuration Protocol (DHCP)

IP setups controlled by DHCP server,
taking care of DHCP request from client
=> Automatic/Dynamic allocation

Suitable for DHCP:
- a lot of mobile devices or PC in this area

NOT suitable for DHCP:
- no DHCP server or few devices

DHCP should be setup along with a DNS server
to efficiently recognizing all clients

UDP (User Datagram Protocol 用戶資料協定)

no response no severe check (used with TCP)
(faster than TCP since no three-way handshake)
=> good for immediate data transfer

(applicable when the correctness of the data
is NOT of much importance, e.g. webcam or LIVE)

Transmission Control Protocol (TCP)

連接導向TCP v.s. 非連接導向UDP
ensure to destination correctly

TCP headers

* Source/Destination Port
* Sequence Number: 封包拆裝
* ......
* code (control flag): 6 bits (1:on)

- URG (Urgent)
- ACK (Acknowledge) (response packet)

Transmission Control Protocol (TCP)

- PSH (Push Function) (send immediately)
- RST (Reset) (forced disconnection)
- SYN (Synchronous)(take care simultaneously/ require for connection);
- FIN (Finish) (agree disconnection?/ end of data transmission);

* Window (if window==0: buffer full)
* Checksum (交叉比對 不符合重新發送封包)
* ......

Transmission Control Protocol (TCP)

TCP Three-way handshake

A. Sending packets: request for connection
   SYN=1(主動連線) & say,
   Sequence number=10001 (port>1024)
B. Packet sending/transmission:
   provide packets with SYN=1
   ack = 10001 + 1 = 10002
   (for client confirmation)
   Server output: Sequence=20001, for
   example, wait for response from client

Transmission Control Protocol (TCP)

C.if ACK is correct & agree to transmit data
=> send packet (ACK=1) again to server
with ack = 20001 + 1 = 20002

D. if server receive ACK = 1 ack = 20002
=> connection success

Reference

1 Vbird

2 Computer Foundation

http://www.study-area.org/network/network_ip.htm