Georgiy Shevoroshkin 13.11.2025

Computer Networks 1 | CN1

Summary

CONTENTS

1. APPLICATION LAYER (7,6,5)

Combines Layers 7 (Application), 6 (Presentation) and 5 (Session).

1.1. COMMON PORTS

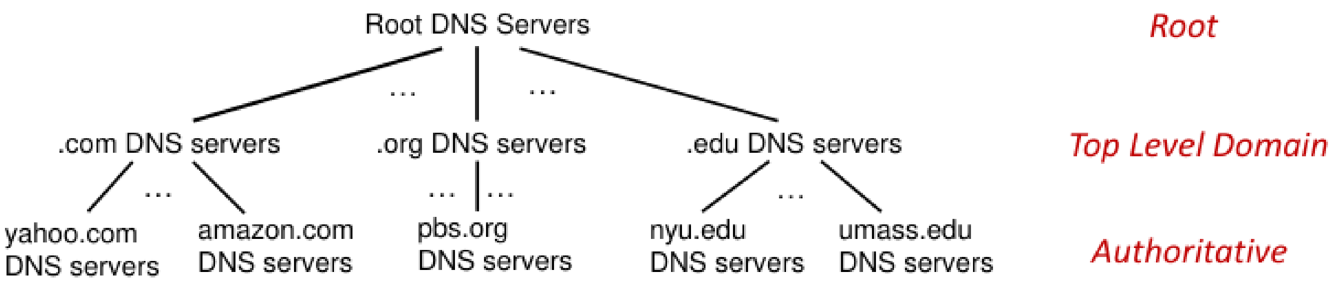
Protocol	*Port*	*Layer 4*
DNS	53	UDP, TCP
HTTP	80	TCP
HTTPS	443	TCP
FTP	20, 21	TCP
SMTP	25 (server) 587 (client)	TCP
POP3	110	TCP
DHCP	67 (server) 68 (client)	UDP

1.2. HTTP

Feature	*HTTP/1.0*	*HTTP/1.1*	*HTTP/2*	*HTTP/3*
Connection Management	One request per connection	Persistent connections by default	Multiplexing allows multiple streams	Uses QUIC for multiplexing
Request Methods	Limited (GET, POST, HEAD)	Enhanced (PUT, DELETE, OPTIONS, etc.)	Same as 1.1	Same as 1.1
Caching	Basic caching support	Improved caching with validation	Advanced caching capabilities	Same as 2 but with improved mechanisms
Header Compression	None	None	HPACK (header compression)	QPACK (header compression)
Server Push	Not supported	Not supported	Supported (automatic resource pushing)	Enhanced support for server push
Performance Improvements	None	Minor improvements over 1.0	Significant improvements in performance and latency	Further improvements in speed and efficiency
SSL/TLS Support	Not inherent	Not inherent, but commonly supported	Built-in support with ALPN (Application-Layer Protocol Negotiation)	Uses QUIC, which incorporates TLS 1.3
Transport Protocol	TCP	TCP	TCP	QUIC

1.3. DNS

Nameservers resolve domains to IP's through a distributed, hierarchical database.



Term	***Definition***
Iterated query	Local DNS server iteratively asks one server after the other, descending the domain name hierarchy step after step.
Recursive query	Local DNS server asks root server for domain, which in turn asks the TLD server, which in turn asks the authoritative server etc. until the "call stack" unwinds and returns the fully resolved domain to the query sender.
Caching	

1.3.1. Record types

Term	***Definition***
A	***name***: hostname ***value***: IPv4 address
AAAA	***name***: hostname ***value***: IPv6 address
CNAME	***name***: alias ***value***: canonical name
NS	***name***: domain ***value***: hostname of authoritateive NS for this domain
MX	***name***: domain ***value***: name of mailserver

1.4. E-MAIL

Term	***Definition***
ding	
dong	
your	
opinion	
is	
wrong	

2. TRANSPORT LAYER (4)

Segment size: 1440-1480b when using IPv4, <=1460b when using IPv6

2.1. PRIMARY RESPONSIBILITIES

- Process-to-process delivery (distinguish between multiple applications via ports)
- Ensure reliable transfer (acknowledgments, retransmissions & reordering)
- Flow control (sender does not overwhelm receiver)
- Congestion control (network is not overloaded)

Term	*Definition*
Port	***16 bit long*** numbers (0d**0**-0d**65'535**) for identifying applications to send packets to. ***Well-Known***: 0d**0**-0d**1'023** for universal TCP/IP applications, managed by the IANA. ***Registered***: 0d**1'024**-0d**49'151** for known applications, also managed by the IANA. ***Private***: 0d**49'152**-0d**65'535** for custom applications, not managed by the IANA.
Socket	Combination of *IP:Port*.
Multiplexing	Sending data from multiple sockets at sender.
Demultiplexing	Delivering segments to correct socket at receiver.
Checksum	Detect errors (i.e., flipped bits) in transmitted segment.

2.2. TCP

Connection-oriented, bidirectional, reliable, managed data flow.

Term	*Definition*
Handshake	Agreement on **starting sequence numbers**, **maximum segment size** and **window scaling**. 1) SEQ 2) SEQ+ACK 3) ACK
FIN	Termination of a connection. 1) FIN 2) FIN+ACK 3) ACK
Round Trip Time	*RTT* is the time it takes for a packet to be sent to the receiver and acknowledged back to the sender.
Buffer size	Maximum amount of data (measured in bytes) that can be stored in memory while waiting to be processed or transmitted.
Maximum Segment Size	*MSS* is the maximum payload size of a TCP packet. In IPv4 networks, typically, the size of the MSS is **1460 bytes** because it is encapsulated in the data link layer Ethernet frame size of **1500 bytes**.

2.2.1. Reliability

Term	*Definition*
Sequence numbers	*SEQ* ensures that the packets arrive or can be reassembled in order.
Acknowledgement	*ACK* ensures that the receiver gets all of the packets.

Term	*Definition*
Retransmission timeout	If an acknowledgment is not received before the timer for a segment expires, a retransmission timeout occurs, and the segment is **automatically retransmitted**.
Packet loss rate	Measures how many packets of the ones being sent actually arrive.

2.2.2. Throughput

Term	*Definition*
Throughput	Denoted by *T*, is the amount of data that can be transmitted during a specified time. $T = \frac{W}{R} \leq C_{L3}$
Continuous sending	Sender transmits a stream of data packets in the given window size **without waiting for acknowledgments**.
Delayed ACK	Receiver waits for a short period to acknowledge **multiple segments** with a **single ACK**.
Selective ACK	Instead of asking for a retransmission of all missing segments, *SACK* (specified by the receiver) allows the sender to send only the lost segments, significantly improving efficiency.

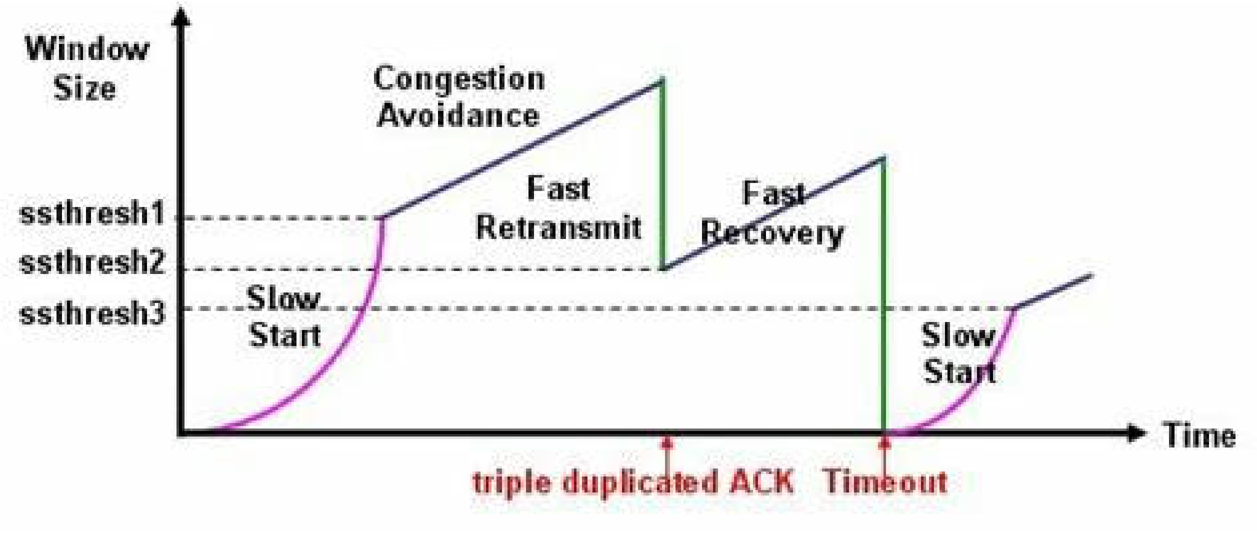
2.2.3. Flow control

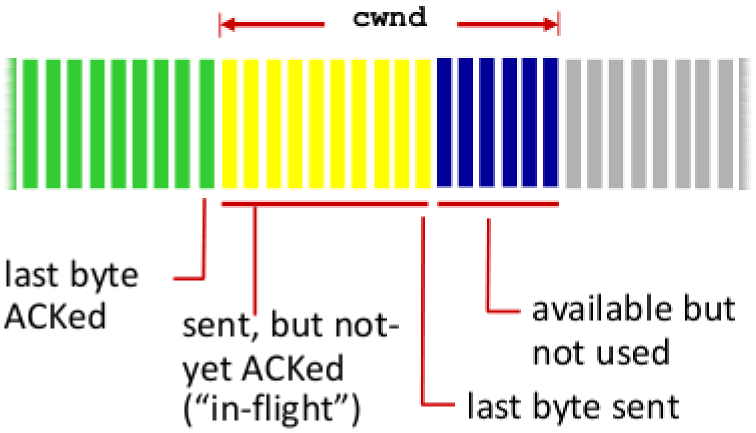
So that the sender does not overwhelm the receiver.

Term	*Definition*
Window Size	Denoted by *W*, is a *16 bit* number sent with each packet by the receiver inside of the **rwnd** header field, indicating the amount of data he still has space for.
Window scale	Used when the TCP window size needs to be increased beyond the traditional maximum of 65,535 bytes due to the demands of high-speed networks. If the handshake header includes the **window scale option** and the packet header includes the **scaling factor** then the effective window size is calculated as such: $\textbf{window size} * \textbf{scaling factor}$

2.2.4. Congestion control

To prevent network congestion.



Term	*Definition*
Congestion window	 cwnd last byte ACKed sent, but not-yet ACKed ("in-flight") available but not used last byte sent
Sliding Window	Describes the process of the congestion window sliding to the right after receiving ACKs.
Slow start	Gradual growth (doubling **cwnd** every **RTT**) within the congestion window size at the start of a connection or after a period of state of no activity. ***Purpose***: Allows the sender to probe the available bandwidth in a controlled way.
Congestion avoidance	Transition from sluggish start to congestion avoidance segment after accomplishing a threshold. ***Purpose***: Maintains a truthful share of the community bandwidth even as heading off excessive congestion.
Fast Retransmit	Detects packet loss through duplicate acknowledgments and triggers speedy retransmission without waiting for the **retransmission timeout**. ***Purpose***: Speeds up the recuperation method with the aid of retransmitting lost packets without looking ahead to a timeout.
Fast Recovery	Enters a quick healing state after detecting packet loss, lowering congestion window and transitioning to congestion avoidance. ***Purpose***: Accelerates healing from congestion by way of avoiding a complete go back to slow begin after packet loss.
AIMD	Adjusts the congestion window size based on network situations following the **Additive Increase, Multiplicative Decrease** principle. ***Purpose***: Provides a balanced approach by way of linearly growing the window all through congestion avoidance and halving it on packet loss.

2.3. UDP

2.4. QUIC

Actually a layer 7 Protocol, running on top of UDP

3. NETWORK LAYER (3)

Packet size: **1500b**

3.1. SUBNETTING

Dividing a */X* network into *n* amount of */Y* subnets: $2^{Y-X} = n$.

Eg: Dividing a */16* network into */24* subnets will yield *256* subnets, because $2^{24-16} = 2^8 = 256$

3.2. IPV6

3.2.1. Glossary

Term	*Definition*
Extension Header	Additional headers used in IPv6 to provide optional information. These can define aspects like payload size, routing, or fragmentation.
DHCPv6	Dynamic Host Configuration Protocol for IPv6; this allows servers to assign IPv6 addresses dynamically from a pool, similar to DHCP for IPv4.
NAT64	Network Address Translation from IPv6 to IPv4 and vice versa; it facilitates communication between IPv6 and IPv4 networks.
Neighbor Discovery Protocol **(NDP)**	A protocol in IPv6 for discovering other network nodes, determining their link-layer addresses, and ensuring that addresses are valid and reachable.
Neighbor Solicitation	
Router Advertisement **(RA)**	A message sent by routers to announce their presence along with various link parameters.
Router Solicitation **(RS)**	A message sent by hosts to request additional information from routers.
Internet Control Message Protocol **(ICMPv6)**	A crucial part of IPv6 that handles error messages and operational queries, with an expanded role compared to ICMP in IPv4.
MTU	Maximum Transmission Unit; the size of the largest packet that can be sent in a single frame over a network medium. IPv6 can handle larger MTUs compared to IPv4.

3.2.2. Special addresses

Term	*Definition*
Link-local Address	***FE80::/10*** Used for local communication between devices on the same network segment.
Global Unicast Address	***2000::/3*** A globally routable address, these addresses are equivalent to public IPv4 addresses and can be reached over the internet.
Unique Local Address **(ULA)**	***FC00::/7*** An address for local communication that is not routable on the global internet, similar to private addresses in IPv4.
Multicast Address	***FF00::/8*** An address that enables a single packet to be sent to multiple destinations simultaneously.
Anycast Address	An address assigned to multiple interfaces, where a packet sent to an anycast address is routed to the nearest (in terms of routing distance) interface.
Reserved Address	Certain ranges in IPv6 are reserved for future use or specific functions. For example, addresses starting with ***::/128*** are reserved for unspecified addresses.
Documentation Address	***2001:DB8::/32*** Designated specifically for use in documentation and examples, ensuring it does not conflict with real-world addresses.
Link-local Multicast Address	***FF02::/16*** Part of the link-local address range; it enables devices to communicate within a local network without requiring an external routing address.

3.2.3. Header

Term	*Definition*
Version	Always 6 with IPv6. IPv4 would be 4.
Flow Label	For identifying packets that require special handling, like real-time streaming.
Traffic Class	Priority or type of traffic.
Payload Length	Size of the payload in bytes.
Next Header	Type of optional header following the IPv6 header.
Hop Limit	Maximum number of hops a packet can take before being discarded.

3.2.4. Stateless Address Autoconfiguration (SLAAC)

A method for automatically configuring IPv6 addresses without a DHCP server, relying on local network information.

3.2.4.1. Autoconfigure link-local address

Mac address: **70:07:12:34:56:78**

1) Flip **7th** bit: 7**2**:07:12:34:56:78
2) Insert **FFEE** in the middle: 7207:12**FF:EE**34:5678
3) Combine with link-local prefix: **FE80::**7207:12FF:EE34:5678

New address: **FE80::7207:12FF:EE34:5678**

3.2.4.2. Perform Duplicate Address Detection (DAD)

To make sure that the address is actually unique in the local segment.
Upon configuring an IPv6 address, every node joins a **multicast group** identified by the address ***FF02::1:FFxx:xxxx*** where xx:xxxx are the **last 6 hexadecimal values** in the IPv6 unicast address, eg. FF02::1:FF**34:5678**

3.2.4.3. TBD...

3.3. IPV4

3.3.1. Network classes (private nets)

Term	*Definition*
A	**10.0.0.0** - 10.255.255.255 (10*/8* prefix)
B	**172.16.0.0** - 172.31.255.255 (172.16*/12* prefix)
C	**192.168.0.0** - 192.168.255.255 (192.168*/16* prefix)

3.3.2. Subnetting

3.3.2.1. Calculating subnet mask

/24 = *1111 1111 . 1111 1111 . 1111 1111 . 0000 0000* = 255.255.255.0
/10 = *1111 1111 . 1100 0000 . 0000 0000 . 0000 0000* = 255.192.0.0

3.3.2.2. Calculating increment

Address increment = $\frac{\textbf{amount of addresses}}{\mathbf{256}}$ or $\mathbf{2^{8-(\text{mask mod } 8)}}$
Let there be 4 subsequent networks starting with 10.0.0.0, each being /20
Amount of addresses = $\mathbf{2^{32-20} = 2^{12} = 4096}$. Increment = $\mathbf{\frac{4096}{256} = 16}$
Alternatively: $\mathbf{2^{8-(20 \bmod 8)} = 2^{8-4} = 2^4 = 16}$
Networks = 10.0.**0**.0/20, 10.0.**16**.0/20, 10.0.**32**.0/20, 10.0.**48**.0/20

3.4. ROUTING

3.4.1. Data plane

- local, per-router function
- determines how datagram arriving on router input port is forwarded to router output port

3.4.2. Control plane

- network-wide logic
- determines how datagram is routed among routers along end-to-end path from source to destination host

3.4.3. Static

3.4.4. Dynamic

3.4.4.1. Algorithms

3.4.4.1.1. Dijkstra’s algorithm (Link State)

3.4.4.1.2. ? algorithm (Distance vector)

3.4.4.2. Protocols

3.4.4.2.1. OSPF (Open Shortest Path First)

3.4.4.2.2. BGP (Border Gateway Protocol)

4. CISCO

4.1. ROUTER SETUP

```
Router> enable
Router# configure terminal
Router(config)#
```

4.2. INTERFACES

4.2.1. Static IP Assignment

```
Router(config)# interface GigabetEthernet 0/0/1
Router(config-if)# ip address 172.16.0.0 255.255.255.252
Router(config-if)# no shutdown
Router(config-if)# exit
```

4.2.2. DHCP Assignment

```
Router(config)# interface GigabetEthernet 0/1/1
Router(config-if)# ip address dhcp
Router(config-if)# no shutdown
Router(config-if)# exit
```

4.2.3. Show

```
Router(config)# do show ip interface brief
Router# show ip interface brief
Router# show ip interface GigabetEthernet 0/0/1
```

4.3. VLAN

```
Switch(config)# vlan 120
Switch(config-if)# name vlan-server
Switch(config-if)# exit
```

4.3.1. Assign IP

```
Switch(config)# interface vlan 120
Switch(config-if)# ip address 10.120.0.10 255.255.255.0
```

4.3.2. Access Port

```
Switch(config)# interface GigabitEthernet 0/0/1
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 120
Switch(config-if)# exit
Switch(config)# interface GigabitEthernet 0/0/1-5
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 120
Switch(config-if)# exit
```

4.3.3. Access Port

```
Switch(config-if)# switchport mode trunk
```

4.3.4. VTP (Virtual Trunk Protocol)

4.3.4.1. Server

```
Switch(config)# vtp domain ins
Switch(config)# vtp mode server
```

4.3.4.2. Client

```
Switch(config)# vtp domain ins
Switch(config)# vtp mode client
```

4.3.5. LACP (Link Aggregation Control Protocol)

```
Switch(config-if)# channel-group 5 mode active
Switch(config-if)# channel-group 5 mode passive
```

4.3.6. Load Balancing

```
Switch(config)# port-channel load-balance <strategy>
```

4.3.7. STP (Spanning Tree Protocol)

4.3.7.1. Bridge priority

```
Switch(config)# spanning-tree vlan 1 priority <priority>
```

4.3.7.2. Interface costs

```
Switch(config-if)# spanning-tree cost 100
```

4.3.7.3. PortFast mode

```
Switch(config-if)# spanning-tree portfast
```

4.3.7.4. Show

```
Switch# show spanning-tree
Switch# show spanning-tree root
```

4.4. ROUTING

4.4.1. Static

4.4.1.1. IPv4

```
Router(config-if)# ip route <destination_network_id> <subnet_mask> <next_hop_router>
<adminitrative_distance>?
Router(config-if)# ip route 10.0.0.0 255.0.0.0 192.168.1.1
```

4.4.1.2. IPv6

```
Router(config-if)# ip route <ipv6_prefix> <outgoing_interface> <next-hop>
<administrative_distance>?
```

```
Router(config-if)# ipv6 route 2001:db8:2103:a::/64 GigabitEthernet1/0/1
fe80::ba27:ebff:fea8:3e50
```

4.4.2. OSPF

4.4.2.1. IPv4

```
Router(config)# router ospf <process-id>
Router(config-if)# ip ospf <process-id> area <area-nr>
```

4.4.2.2. IPv6

```
Router(config)# ipv6 router ospf <process-id>
Router(config-if)# ipv6 ospf <process-id> area <area-nr>
```

4.4.3. Show

```
Router# show ip route
Router# show ip ospf route
```

4.5. DHCP

4.5.1. Create Pool

```
Router#(config-if) ip dhcp pool DEV
 network 192.168.1.0 255.255.255.0
 default-router 192.168.1.1
 dns-server 1.1.1.1 8.8.8.8
 lease 5
 domain-name enterprise.com
```

4.5.2. Relay Agent

```
Router(config-if)# ip helper-address 176.16.12.10
```

4.6. NAT

4.6.1. IF Inside

```
Router(config-if)# ip nat inside
```

4.6.2. IF Outside

```
Router(config-if)# ip nat outside
```

4.6.3. ACL (Access Control List)

```
Router(config-if)# access-list 1 permit 192.168.1.0 0.0.0.255
```

4.6.4. PAT (Nat overload)

```
Router(config-if)# ip nat inside source list 1 interface GigabitEthernet0/1 overload
```

4.7. IPV6

```
Router(config-if)# ipv6 enable
```

4.7.1. DHCPv6

```
Router(config-if)# ipv6 dhcp client pd MY_PREFIX
Router(config-if)# ipv6 address autoconfig default
```

4.8. TROUBLESHOOTING

4.8.1. Ping

```
Router# ping <destination-ip> source <interface-name>
```

4.8.2. Traceroute

```
Router# traceroute <destination-ip> source <interface-name> numeric
```

5. BINARY, DECIMAL, HEX

0x**A4 6A** = 0b**1010 0100 0110 1010** = 0d**42'090**
0x**04 B0** = 0b**0100 1011 0000** = 0d**1'200**
0x**01 D4 C0** = 0b**0001 1101 0100 1100 0000** = 0d**120'000**