Computer Networks: Network Layer (Comprehensive Exam Guide)

1. Network Layer Design Issues and Services

The **Network Layer** is the third layer in OSI and TCP/IP models and is responsible for **packet delivery from source to destination** across multiple networks (internetwork). It deals with:

- Addressing and routing
- Path determination
- Forwarding (packet switching)
- Congestion control
- Quality of Service

Key Design Issues:

1. **Services to Transport Layer:** Connection oriented (virtual circuit) or Connectionless (datagram).
2. **Routing:** Select best path among multiple routes (dynamic/static, adaptive/non-adaptive).
3. **Addressing:** Unique logical (network) addresses (IPv4/IPv6).
4. **Packetizing & Forwarding:** Convert segments to packets, send via best route.
5. **Traffic Control:** Prevent Congestion, manage flow, avoid bottlenecks.
6. **Reliable Delivery (Error Control):** Ensure error-free and in-sequence delivery.
7. **Fragmentation & Reassembly:** Handle different Maximum Transmission Units (MTU) over hops.

Network Layer Services:

- **Connectionless Packet Delivery (Datagram):** Each packet routed independently.
- **Connection-Oriented Service (Virtual Circuit):** Path established before any data transfer.

2. Application of Network Layers

Applications utilizing the network layer:

- **Internet communication:** Emails, web browsing (HTTP), file transfers (FTP).
- **VoIP & Video Streaming:** Real-time data routed efficiently.
- **Online gaming:** Low-latency packet switching.
- **Remote access, VPNs:** Secure routing across public networks.
- **IoT Connectivity:** Billions of devices identified/logged/communicated.

3. Virtual Circuit vs Datagram Networks (Tabular Comparison)

Aspect	Virtual Circuit	Datagram Network
Concept	Connection-oriented, circuit set up before data	Connectionless, each packet handled independently
Path	Fixed path for session	Dynamic routing per packet
Addressing	Circuit/VC number (lightweight)	Full src-dest address in every packet
Resource allocation	Pre-allocated (reserved resources)	On demand (no reservation)
Setup/Tear down	Setup and teardown required	None
Reliability	High (in-order, reliable)	Best effort, may be unordered
Examples	ATM, Frame Relay	Internet (IP)
Overhead	Higher	Lower
Cost	More expensive (maintenance)	Cheaper (flexible)
Congestion response	All packets may be delayed/dropped	Individual packets may reroute
Use cases	Telephone, financial, MPLS	Internet data, VoIP, email

In summary:
Virtual Circuits are great for predictable, reliable sessions (voice, banking).
Datagram networks suit bursty, best-effort, and flexible communication (Internet).

4. Routing Algorithms

Routing is the process of choosing paths for packets in a network. Key routing algorithm types:

a) Distance Vector Routing

- Each router maintains a vector (table) of minimum distance to every node.
- Routers share distance vectors with neighbors at intervals; update table on change.
- **Metric:** Hop count (RIP), delay, cost etc.
- **Convergence:** Slow, count-to-infinity problem.
- **Example protocols:** RIP (Routing Information Protocol).

How it works:
Each router sends its entire routing table to immediate neighbors. If a neighbor advertises a new, shorter path to a destination, the router updates its table accordingly.

b) Shortest Path Algorithm

Main method: **Dijkstra's Algorithm (Link-State Routing)**

- Each router learns full network topology.
- Computes shortest paths to every node using weights (costs).
- All routers flood their link state info; all compute same shortest path tree.
- **Convergence:** Fast, robust.
- **Example protocols:** OSPF (Open Shortest Path First).

Dijkstra's Steps:

1. Initialize all distances as infinity except for the starting node (0).
2. Visit unvisited node with smallest distance.
3. Update adjacent nodes with minimum cost if shorter path found.
4. Repeat until all nodes processed.

c) Hierarchical Routing

- Used in large networks; divides network into regions or **autonomous systems (AS)**.
- **Intra-domain routing:** Within AS (OSPF, IS-IS).
- **Inter-domain routing:** Between AS (BGP—Border Gateway Protocol).
- Reduces complexity: routers handle only summary info for other regions.

Example: The Internet backbone uses BGP for inter-domain routing, and OSPF/IS-IS within large organizations.

5. Congestion Prevention Policies

Congestion occurs when demand > network capacity. **Prevention Policies:**

1. **Admission Control:** New flows are refused if load is high.
2. **Traffic Shaping:** Smooth out bursts using algorithms (leaky/token bucket).
3. **Fair Scheduling/Queuing:** Priority given to critical flows (voice over video over bulk files).
4. **Packet Dropping:** Proactively drops packets (tail drop, RED) to signal sources to slow down.
5. **Resource Reservation:** Reserve bandwidth for critical applications (QoS).
6. **Packet Marking:** Mark packets for priority handling during congestion.

6. Congestion Control Algorithms

a) Leaky Bucket Algorithm

- **Analogy:** Leaky water bucket—water added at various rates, leaks at fixed rate.
- **Mechanism:** Incoming packets buffered. Sent at a constant (leak) rate. Overflow drops packets.
- **Application:** Ensures **constant output rate**, prevents burst overload.

Diagram:

```
+------------+
| Leaky |
-->| Bucket |--> [Constant Out]
+------------+
|
[Input may burst; output is smooth]
```

b) Token Bucket Algorithm

- **Analogy:** Bucket filled with tokens at steady rate. Each token permits one packet transmission.
- **Mechanism:** Packets transmitted only if enough tokens available (allows bursts up to bucket size).
- **Application: Allows bursty traffic** but controls average rate.

Diagram:

```
[Tokens drip in @ rate R]
+------------+
| Token |----+
| Bucket | | Packets out ONLY
+------------+ | if tokens present
^ | |
| +------> Data in
[Refill] [Excess packets wait or drop]
```

Leaky Bucket vs Token Bucket:

Feature	Leaky Bucket	Token Bucket
Output Rate	Constant	Variable (bounded)
Burst Support	No	Yes
Use Case	Traffic smoothing	Rate limiting with bursts

7. Network Layer Protocol Headers

a) IPv4 Header Format

- **20-60 bytes**, variable length (options)
- Main fields:
    - **Version** (4 bits)
    - **Header Length** (4 bits)
    - **Type of Service** (8 bits)
    - **Total Length** (16 bits)
    - **Identification** (16 bits)
    - **Flags** (3 bits)
    - **Fragment Offset** (13 bits)
    - **Time to Live** (8 bits)
    - **Protocol** (8 bits)
    - **Header Checksum** (16 bits)
    - **Source Address** (32 bits)
    - **Destination Address** (32 bits)
    - **Options** (if any)

Figure 1: IPv4 Packet Header Structure

b) IPv6 Header Format

- **Fixed at 40 bytes** (simplified format for speed)
- Main fields:
    - **Version** (4 bits)
    - **Traffic Class** (8 bits)
    - **Flow Label** (20 bits)
    - **Payload Length** (16 bits)
    - **Next Header** (8 bits)
    - **Hop Limit** (8 bits)
    - **Source Address** (128 bits)
    - **Destination Address** (128 bits)

Figure 2: IPv6 Packet Header Structure

8. Comparison: IPv4 vs IPv6

Side-by-side Feature Chart

IPv4 vs IPv6 Comparison

Feature	IPv4	IPv6
Header Size	20-60 bytes (variable)	40 bytes (fixed)
Address Length	32 bits	128 bits
# of Addresses	2^32 (~4.3 billion)	2^128 (~340 undecillion)
Address Format	Decimal (x.x.x.x) e.g. 192.168.0.1	Hexadecimal (x:x:x:x:x:x:x:x) e.g. 2600::1
Header Checksum	Yes (computed per hop)	No (removed)
Fragmentation	Done by router/host	Only by source
QoS Support	Type of Service (ToS)	Traffic Class + Flow Label
Security	Optional	Mandatory

Figure 3: Comprehensive Comparison: IPv4 vs IPv6 Protocol Features

Tabular Summary

Feature	IPv4	IPv6
Header Size	20-60 bytes (variable)	40 bytes (fixed)
Address Length	32 bits	128 bits
Address Space	$\approx$4.3 billion	$\approx$340 undecillion
Address Format	Decimal, dot-separated (A.B.C.D)	Hexadecimal, colon-separated
Checksum	Yes	No
Fragmentation	Routers and hosts	Only by sender (source)
Security	Optional (IPSec)	Mandatory (IPSec)
Quality of Service	ToS/DSCP	Traffic Class, Flow Label
Broadcast	Yes	No (uses multicast only)
Processing Speed	Lower (variable header)	Higher (fixed header)
Options	In header	Extensions

References

[1] GeeksforGeeks, "Design Issues in Network Layer", https://www.geeksforgeeks.org/computer-networks/design-issues-in-network-layer/
[2] GeeksforGeeks, "Distance Vector Routing (DVR) Protocol", https://www.geeksforgeeks.org/computer-networks/distance-vector-routing-dvr-protocol/
[3] Wikipedia, "Dijkstra's algorithm", https://en.wikipedia.org/wiki/Dijkstra's_algorithm
[4] GeeksforGeeks, "Token Bucket Algorithm", https://www.geeksforgeeks.org/computer-networks/token-bucket-algorithm/
[5] GeeksforGeeks, "Congestion Control in Computer Networks", https://www.geeksforgeeks.org/computer-networks/congestion-control-in-computer-networks/
[6] GeeksforGeeks, "Difference Between IPv4 and IPv6", https://www.geeksforgeeks.org/computer-networks/differences-between-ipv4-and-ipv6/
[7] AWS, "IPv4 vs IPv6 - Difference Between Internet Protocol Versions", https://aws.amazon.com/compare/the-difference-between-ipv4-and-ipv6/
[8] W3Schools, "IP Addressing", https://www.w3schools.com/
[9] NetworkAcademy.IO, "IPv4 & IPv6 header differences", https://www.networkacademy.io/ccna/ipv4-vs-ipv6

[10] TutorialsPoint, “Comparison of Virtual-Circuit and Datagram Networks”, https://www.tutorialspoint.com/comparison-of-virtual-circuit-and-datagram-networks
[11] Dr. Balvinder Taneja, “Leaky bucket and token bucket algorithm”, https://drbtaneja.com/leaky-bucket-and-token-bucket-algorithm/