Network Layer

Study-Ready Notes

Compiled by Andrew Photinakis

November 15, 2025

Contents

Chapter 4: The Network Layer — Data Plane

0.1 4.1 Overview of Network Layer

0.1.1 4.1.1 Forwarding and Routing: The Data and Control Planes

Two key functions of the network layer, **forwarding** and **routing**, map directly to the **data plane** and **control plane**, respectively.

0.1.2 1. Concept Overview: The Two Core Functions

Welcome. In our last lecture, we completed our study of the transport layer, which provides a *logical* communication service between *processes* running on different hosts. Now, we move down the stack to examine the network layer, which provides the underlying *host-to-host* communication service that the transport layer relies on.

The network layer's primary role seems simple: to move packets, called **datagrams**, from a sending host to a receiving host. However, to accomplish this, the network layer must perform two distinct and critical functions: **forwarding** and **routing**.

1. Forwarding
    - Is a *local* action.
    - The process of taking a packet that has arrived on one of a router's input links and moving it to the appropriate output link.
2. Routing (Control Plane Function)
    - Is a *network-wide* process that determines the end-to-end paths that packets take from a source host to a destination host.

0.2 Forwarding: The "Data Plane"

1. Definition
    - Forwarding refers to the router-local action of transferring a packet from an input link interface to the appropriate output link interface.
    - Forwarding is the primary function of the network layer's **data plane**, which includes per-router operations performed on a packet as it moves through the router.
2. Mechanism
    - A packet arrives at a router's input link.
    - Router examines one or more fields in the packet's header.
    - Uses header values to index into its forwarding table.
    - Value found in the forwarding table entry indicates the router's output link interface to which the packet should be forwarded.

3. Timescale & Implementation
    - Forwarding is a fast (nanosecond) action that must be performed for every packet.
    - Therefore, it is implemented in hardware.
4. Analogy
    - Forwarding is like navigating a single highway interchange or roundabout: the driver makes a local, split-second decision based on signs.

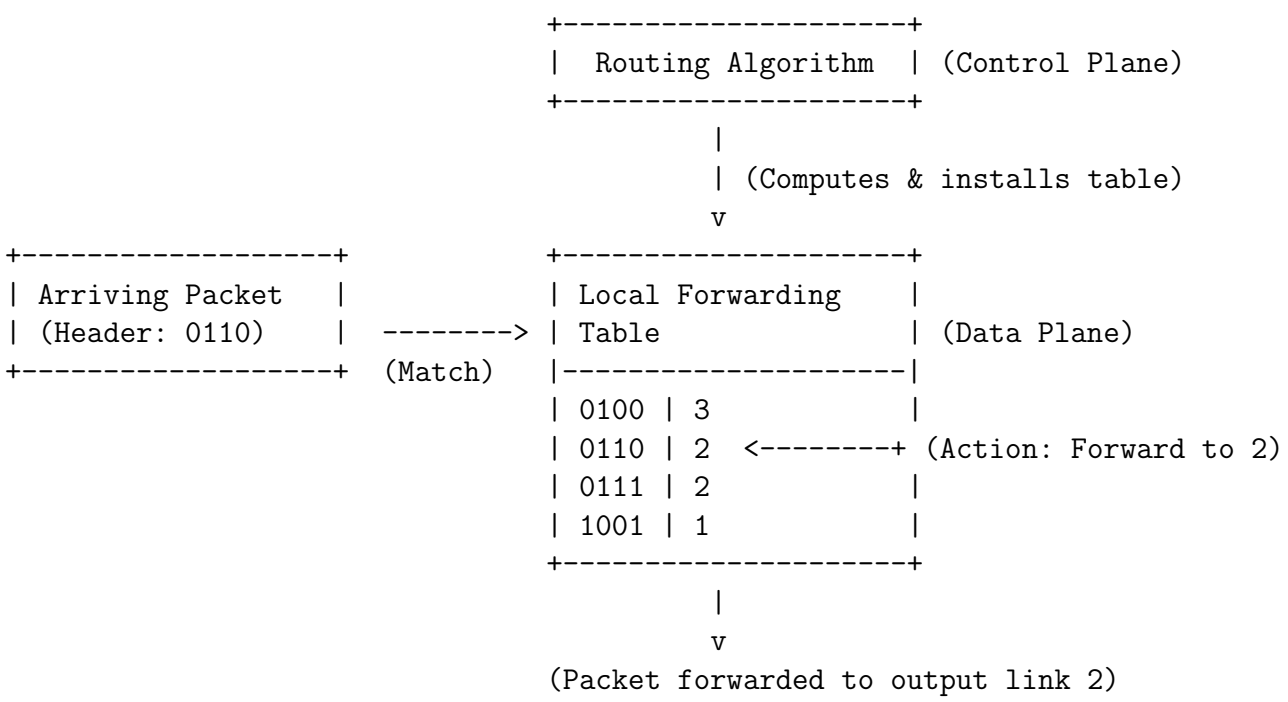
0.3 Routing (Control Plane Function)

1. Definition
    - Routing refers to the network-wide process of calculating and determining the end-to-end paths that datagrams follow from source to destination.
    - It is the primary function of the **control plane**, which governs network-wide routing logic.
2. Mechanism
    - Routing is accomplished by routing algorithms that calculate paths for datagrams.
    - Routing decisions are installed in the forwarding tables of routers.
3. Timescale & Implementation
    - Routing computations occur on much longer timescales (seconds to minutes).
    - Typically implemented in software due to their complexity.
4. Analogy
    - Routing is like planning an entire trip from Pennsylvania to Florida: the routing algorithm (driver) consults a map to select the best end-to-end path.
    - Each router on the path forwards packets according to the plan.

0.4 4.1.1.1 The Forwarding Table

The routing (control plane) controls forwarding (data plane) via the **forwarding table**.

- **Function:** Router examines header fields of incoming packets, indexes into the forwarding table, and forwards the packet according to the table entry.
- **Origin:** Entries are computed and installed by the control plane. Changes in topology or costs trigger updates.



0.5 4.1.1.2 Approaches to the Control Plane

Approach 1: Traditional Per-Router Control

- Control plane runs in each router.
- Routing components communicate via routing protocols to compute forwarding tables.
- Distributed architecture.

Approach 2: Software-Defined Networking (SDN)

- Control plane is physically separated from data plane.
- A remote controller computes forwarding tables and installs them on all routers.
- Routers act as pure data-plane devices.
- Controller runs in software, typically on a reliable data center, and communicates with routers using protocols like OpenFlow.

0.6 4.1.1 Summary

- **Forwarding:** Local, fast, hardware-level data-plane function.
- **Routing:** Network-wide, slower, software-level control-plane function.
- **Data Plane:** Handles forwarding using the forwarding table.
- **Control Plane:** Computes paths and installs forwarding table entries.

- **Forwarding Table:** Links control plane and data plane.
- **Per-Router Control:** Traditional distributed routing model.
- **Logically Centralized Control (SDN):** Centralized controller computes forwarding tables.

1 4.1.2 Network Service Model

1.1 1. Concept Overview: The "Contract" of the Network Layer

The network service model defines the characteristics of end-to-end delivery of packets between sending and receiving hosts. It represents the "contract" between the network and transport layers:

- Will the packet be delivered?
- Will it arrive intact?
- Will it arrive in order?
- Will it arrive within a certain time?
- Is there a minimum throughput guarantee?

1.2 2. Technical Mechanism: A "Menu" of Possible Services

Potential services a network layer could offer:

- **Guaranteed Delivery:** Every packet eventually arrives.
- **Guaranteed Delivery with Bounded Delay:** Delivery occurs within a specified time.
- **In-Order Packet Delivery:** Packets arrive in sending order.
- **Guaranteed Minimal Bandwidth:** Network behaves like a dedicated link with a specified bit rate.
- **Security:** Encrypt all datagrams for confidentiality.

1.3 3. Practical and Intuitive View: The Internet's *Actual* Service

The Internet provides a single service: **best-effort service**:

- Delivery is not guaranteed; packets may be lost.
- In-order delivery is not guaranteed; packets may arrive out of order.

- Delay is not guaranteed; packets may experience long delays.
- Bandwidth is not guaranteed.

Design philosophy: *Keep the network core simple.* Reliability, ordering, and congestion control are pushed to the edges (transport/application layers).

1.4 4. Terminology Clarification: Switches vs. Routers

- **Packet Switch:** Generic device moving packets from input to output interfaces.
- **Link-Layer Switch:** Layer 2 device forwarding based on MAC addresses.
- **Router:** Layer 3 device forwarding based on network-layer addresses (IP addresses).

Focus of this chapter: **routers** (Layer 3 devices).

1.5 4.1.2 Summary

- **Network Service Model:** Defines network guarantees for transport layer.
- **Best-Effort Service:** Internet's minimalist model with no delivery, order, delay, or bandwidth guarantees.
- **Packet Switch, Link-Layer Switch, Router:** Key terminology distinctions.
- Minimalist model pushes intelligence to end systems.

2 4.1 Overview of Network Layer

2.1 Core Role and Architectural Placement

1. Network layer is third layer of the internet protocol stack
2. Resides between the transport layer and link layer
3. Fundamental role is to move network layer packets, called datagrams, from a sending host to a receiving host.
4. Host-to-Host Communication
    - While transport layer provides logical communication between processes, network layer provides logical communication between hosts
5. Universal Implementation
    - Transport and application layers run only on end systems (hosts)
    - Network layer runs in every host and router in network

- It's esential b/c routers must examine datagram headers to perform their forwarding function

6. Router Protocol Stack
   - Routers are network-layer (Layer 3) devices
   - Consist of "truncated" protocol stack, implementing physical, link, and network layers
   - Do not implement transport or application layers
   - Sole purpose is to forward datagrams, not run end-user applications
7. At Each Stage
   (a) Sending Host
   - Network layer takes segments from transport layer, encapsulates them into datagrams, and sends these datagrams to its nearby router

   (b) Receiving Host
   - Network layer receives datagrams from its nearby router, extracts transport-layer segments, and delviers them up to transport layer

   (c) Routers
   - Core function to examine header of an arriving datagram and forward it to appropriate output link
   - Routers are 'Layer 3' devices
   - Typically have a 'truncated' protocol stack, implementing up to network layer but not transport or application layers

2.2 Planes

2.2.1 Data Plane

- Performs per-touer function of forwarding datagrams from a router's input link to its appropriate output link

2.2.2 Control Plane

- Performs network-wide logic that controls how datagrams are routed along an end-to-end path from source to destination host

3 4.2 What's Inside a Router?

3.1 Concept Overview: Unboxing the Data Plane

1. Primary function to transfer packets from incoming links to appropriate outgoing links

2. Task split smong four interconnected componenets

   (a) Input Ports (Hardware):

      i. Perform physical, link, and data-plane lookup for incoming packets
      ii. Is entry point for packets into router
      iii. Physically terminates incoming link, performs link-layer tasks to "unwrap" the packet, and perform lookup function to determine packet output port using forwarding table
      iv. Control packets are passed "up" to routing processor

   (b) Switching Fabric (Hardware):

      i. "Heart" of router
      ii. Purpose to connect input ports to output ports

      Internal mechanism that moves packets from input ports to output ports

   (c) Output Ports (Hardware):

      i. Exit point for packets leaving router
      ii. Receives packets from switching fabric, queues them, performs necessary link-layer and physical-layer functions to "wrap" packet into new frame and finally transmits frame onto outgoing link

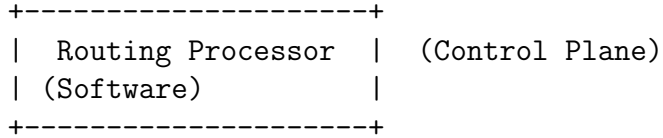
      Store, queue, and transmit packets received from switching fabric onto outgoing link

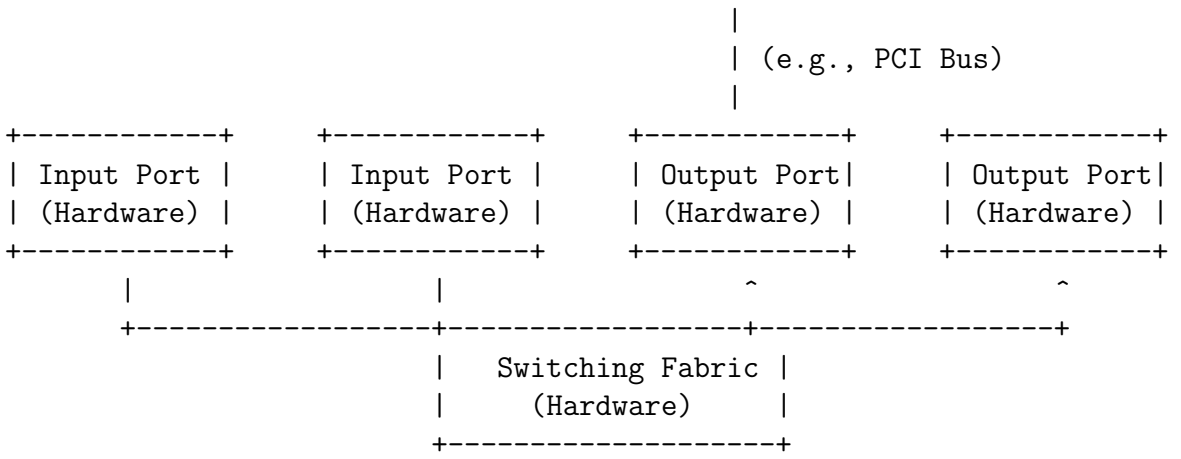
   (d) Routing Processor (Software):

      i. "Brain" of router
      ii. Primary component of control plane.
      iii. Implemented in software and runs on traditional CPU
      iv. Is not directly involved with main forwarding path
         A. In traditional router, routing processor executes routing protocols, maintains routing tables, and uses info to compute and update forwarding table
         B. In SDN router, routing processor is responsible for communicating with remote SDN controller, receiving forwarding table entries from it, and installing entriies into input ports

      Executes control-plane functions like routing protocols and maintains/computers forwarding table

3.2 The Four Components of a Router





3.3 Data Plane (Hardware) vs Control Plane (Software)

Essential to understand why such components are built the way they are

- Data Plane
  1. Reason: Speed, executes in nanosecond timescale
  2. Example:
     (a) Consider 100 Gbps input link
     (b) New packet can arrive every few nanoseconds
     (c) For a min-sized 64-byte (512 bit) IP datagram, router only has 5.12 nanoseconds (512 bits / 100 Gbps) to process before next one arrives
  3. Per-packet forwarding action far too fast for software-based implementation
- Control Plane
  1. Reason: Complexity and Timescale, control-plane functions (like new route using Dijkstra's algo or commuinicating with SDN controller) are complex and happen on much slower timescale - milliseconds or even seconds
  2. Well suited for standard CPU running a software program.

3.4 Roundabout Analogy

- Input Port: Entry road, which includes an attendant (lookup function)
- Switching Fabric: roundabout itself
- Output Port: exit ramp
- Destination-based Forwarding: Tell attendant find destination (i.e. Florida) and attendant looks it up, tells you "take 3rd exit", and you're on the way
- Generalized Forwarding: Attendants decision is more complex

1. "You're from NY? Take the slow road. You're from this state (license plate)? Take highway. Your car is not road-worthy? You're blocked"

- Bottlenecks: Analogy illustrates router's potential performance bottlenecks

1. Lookup bottleneck: Attendant is too slow. Cars pile up on entry road (input port)
2. Switching bottleneck: Roundabout is too small or slow. Cars get stuck in middle (switch fabric)
3. Output bottleneck: Too many cars want same exit ramp. Cars pil up on exit ramp (output port)

3.5 Template Example

- X

1. Y

4 4.3 The Internet Protocol (IP): IPv4, Addressing, IPv6, and More

4.1 Overview

- Delves into foundational protocols and addressing schemes of the Internet Protocol (IP) that enable the Internet's network layer.
- IP Protocol provides essential, unreliable, best-effort delviery service that underpins all higher-layer communcation
- IP Protocol is hte only network-layuer protocol used in the Internet

4.1.1 IP Protocol

- Is the only network-layer protocol in the internet
- Is the "narrow waist" of the hourglass, the single protocol that everything else - every transport protocol (TCP, UDP) and every applciaiton (HTTP, DNS) - must run on top of
- In turn must be able to run over any link-layer technology (Ethernet, WiFi, 4G, etc)

5 4.3.1 IPv4 Datagram Format

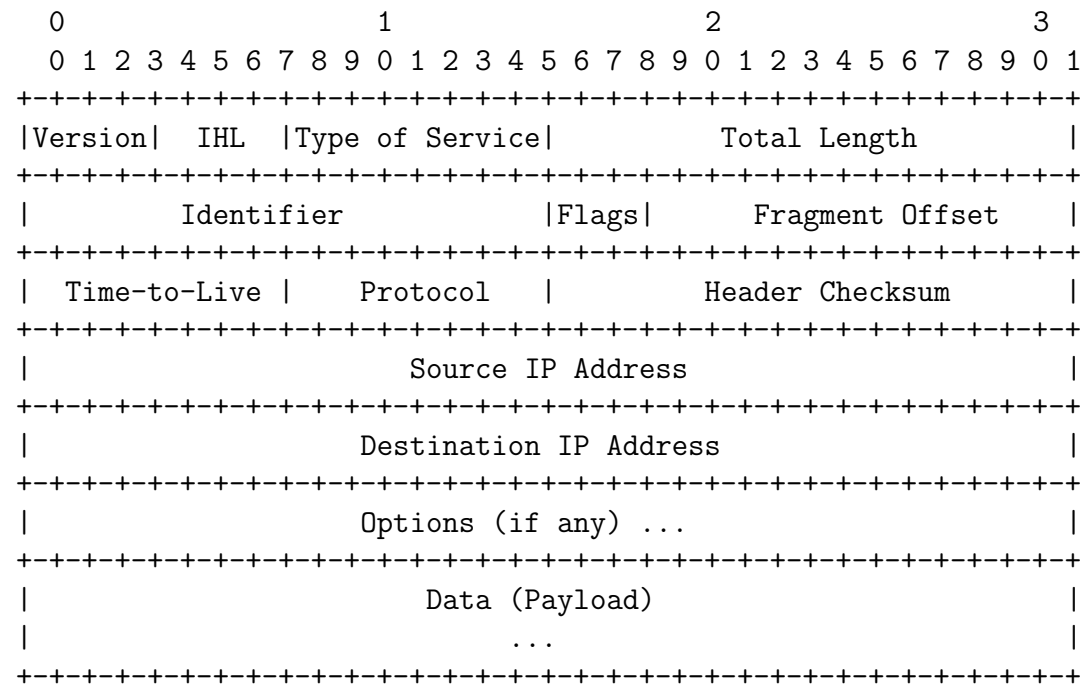
5.1 Concept Overview

Datagram:

1. The Internet's network-layer packet is known as a **datagram**.
2. Fundamental to understanding IP's operations.
3. Format is defined in **RFC 791**.
4. IPv4 datagram has a variable-length header:
   - Typically 20 bytes long when no options are present.
   - Followed by a payload (data field).
5. Composed of two main parts:
   (a) **Header:** Contains all information a router needs to make forwarding decisions.
   (b) **Data (Payload):** Contains the data being transported, typically a transport-layer segment (like TCP or UDP).

5.2 Technical Mechanism: The Datagram Format

- A textual representation of the IPv4 header:



- Key Header Fields and Their Purpose:

1. **Version (4 bits):**
   (a) Specifies IP protocol verstion

   (b) For IPv4, this is always 4.
   (c) Route examines this field first to know how to parse the rest of the header (e.g. to know that it should expect an IPv4 header, not IPv6 header)
2. **IHL (Internet Header Length) (4 bits):**
   (a) Specifies length of IP header in 32-bit words
   (b) Field is necessary because 'Options' field can make header variable-length
   (c) Required to find stasrt of a data/payload fields
   (d) Typical heder length is 20 bytes (5 words)
3. **Type of Service (TOS) (8 bits):** Indicates priority and routing preferences.
   (a) Included to allow different types of IP datagrams (e.g. real-time vs non-real time) to be distinguished and serviced differently (e.g. with priority or low-delay)
   (b) Bits are also used in Explicit Congestion Notification (ECN)
4. **Datagram Length (16 bits):** Length of the entire datagram (header + payload).
   (a) Total length of IP datagraam (header + payload) in bytes
   (b) Practial View:
      i. A 16-bit field allows for theoretical maximum datagram size of 65,535 bytes
      ii. However, datagrams are rarely larger than 1,500 bytes because they must fit within the Maximum Transmission Unit (MTU) of udnerlying link-layer to avoid fragmentation
5. **Identifier, Flags, Fragmentation Offset (32 bits total):** Identifies fragments of the original datagram.
   (a) Fields are all used for IP fragmentation
   (b) If router needs to forward a datagram onto a link with an MTU smaller than datagram's size, it can "fragment" datagram into multiple smaller datagrams
   (c) Such smaller datagreams are reassembled only at final destination
   (d) This is a slow, complex process, and IPv6 gets rid of it entirely for routes
6. **Time-to-Live (TTL) (8 bits):** Limits the datagram's lifetime to prevent infinite loops.
   (a) Critical safety mechanism to ensure datagrams don't circulate forever in network (e.g. in a routing loop)
   (b) TTL field is decremented by one at every router is passes through
   (c) If router receives a datagram with a TTL of 1, it decrements it to 0, discards the datagram, and sends an ICMP error message back to source
7. **Protocol (8 bits):** Specifies the transport protocol in the payload (e.g., TCP, UDP).
   (a) Binds network layer to transport layer

   (b) Used only at final destination host to indicate which transport-layer protocol should receive datagram's payload
      i. 6 = TCP
      ii. 17 = UDP
      iii. 1 = ICMP (Internet Control Message Protocol)
8. **Header Checksum (16 bits):** Error-checking for the header.
   (a) Performs error detection on header only, not on payload
   (b) Computed using 1s complement arithmetic over 2-byte units of the header
   (c) Must be recomputed at every router because TTL field changes
   (d) Why only header?
      i. Transport layer (TCP/UDP) performs its own checksum on payload
      ii. TTL field changes at every route. Means header must be altered. Checksum must be recomputer and restored at every single router, which is time consuming
9. **Source and Destination IP Address (32 bits each):** Addresses of originating and final destination interfaces
   (a) Fields contain 32-bit IP addresses of original src and final dest
   (b) Src address inserted by sending host
   (c) Dest address obtained from a DNS lookup
10. **Options (if any) (variable length):** Optional fields for control or security.
   (a) Field allows IP header to be extended, but rarely used
   (b) Can be problematic because makes header length variable, complkicating and slowing down router processing
11. **Data (Payload):** Actual transported data (e.g., TCP/UDP segment).

5.3 Template Example

- X

   1. Y

6 4.3.2 IPv4 Addressing

6.1 Concept Overview: Interfaces and Subnets

- IP Addresses and Interfaces

   1. IP addr is 32 bits long
   2. IP addr associated with a router or host interface, not device itself
   3. Router has multiple interfaces (one per link), and thus multiple IP addrs

4. Hosts typically have one interface (and one IP address) connecting them to network
5. Addresses are writting in dotted-decimal notation

6.2 Technical Mechanism: The IP Subnet

- IP addresses not assigned randomly
- Portion of address is determined by network interface connected to
- /24 notation is subnet mask
    1. Indicates that leftmost 24 bits define subnet, and remaining (32-24) = 8 bits identify specific interfaces on that subnet
- Recipe for Identifying Subnets
    1. Detach every interface from its host or source
    2. This creates "islands" of isolated networks
    3. Each island is a subnet
- Subnets
    1. IP subnet is a network that connects multiple host interfaces and router interfaces,
    2. Forms isolated network without intervening routers
- Subnet Addressing
    1. Interfaces on given subnet share same high-order bits of their IP addresses
    2. Notation (CIDR)
        (a) A subnet address is denoted by form a.b.c.d/x, where /x indicates number of high-order bits that constitute network portion of address

6.3 Concept: CIDR (Classless Inter-Domain Routing)

1. Is a flexible IP address assignment strategy that replaced older class-based system
2. Introduced /x notation, where x specifies how many bits of the address form network prefix

- Mechanism
    1. CIDR generalizes traditional subnet and hsot division
        (a) In address a.b.c.d/x
        (b) x most significant bits are prefix
        (c) Remaining 32 - x bits are host portion

2. Allows networks to have variable-length prefixes, making address allocation more flexible

- Scalability via Address Aggregation

1. Organizations are assigned contiguous blocks of IP addresses with a common prefix
2. Routers outside of org only need one entry in their forwarding tables.
   (a) "To reach any address startingt with 200.23.16.0/20, forward to that orgs ISP"
3. Ability to use single prefix to advertise many networks = address aggregation (or route aggregation)
4. Essential for keeping global routing tables small and manageble

6.4 Obtaining and Managing IP Addresses

1. Getting a block of addresses (for an ISP or large org)
   (a) Global authority is ICANN (Internet Corporation for Assigned Names and Numbers)
   (b) ICANN allocates address blocks to Regional Internet Registries (RIRs)
   (c) An ISP (like Comcast, or Verizon) gets its address blocks from its RIR
   (d) An org gets its address blocks from its ISP
   (e) Hierarchical allocation allows for route aggregation
2. Getting a host address (DHCP)
   (a) Once org has blocks of addresses (e.g. 68.85.2.0/24) it needs to assign individual addresses to its hosts
   (b) Done automatically by Dynamic Host Configuration Protocol (DHCP)
   (c) DHCP - is a plug and play protocol. When your laptop connects to a network, uses DHCP to automatically get
      i. Its IP address (e.g. 68.85.2.101)
      ii. Its subnet mask (e.g. /24)
      iii. IP address of default gateway (first hop router)
      iv. IP address of local DNS server
   (d) How DHCP works?
      i. DHCP Discover
         A. New host client sends broadcast message (Dest IP: 255.255.255.255, Souce IP: 0.0.0.0)
         B. Asks "is there a DHCP server out there?"

ii. DHCP Offer
   A. A DHCP server (typically on the router) receives the discover and replies with offer message
   B. Includes proposing an IP address, lease time, etc.
iii. DHCP Request
   A. Client formally requests offered address by sending a broadcast "request" message
   B. This tells any other servers that sent offers that they weren't chosen
iv. DHCP ACK
   A. Server confirms allocation with a "DHCP ACK" message
   B. Client can now use IP address

6.5 Template Example

- X
   1. Y

7 4.4 Generalized Forwarding and SDN

7.1 Concept Overview

- Traditional Model of Fowarding
   1. Router makes forwarding decision based on single field in packet header, dest IP address
   2. Router sole job to perform lookup on dest IP address (using longest prefix matching) and send packet to correct output port
- Generalized Forwarding
   1. Describes a "match-plus-action" paradigm where a packet switch (which could be a router or a switch) makes decision based on wide range of fields in packet header, spanning multiple protocol layers
   2. Match-plus-action abstraction
      (a) Instead of just matching dest IP, switch can match on any combination of fields, such as src IP, dest port, source MAC addresses, VLAN tag, etc.
      (b) Action:
         i. Instead of just forwarding, switch can perform variety of actions, such as
         A. Forward packet to one or more output ports (e.g. unicast, multicast, or broadcast)
         B. Load balance packet across mutliple output ports
         C. Rewrite head values

D. Block/drop packet
E. Send packet to special server or controller for further processing (e.g. deep packet inspection)

3. Flow Table
   (a) Generalizes simple forwarding table into a more complex flow table, flow table is "program" for packet switch's data plane
4. Connection to SDN
   (a) Generalized forwarding model is data-plane foundation of Software-Defined Networking
   (b) Because "match" and "action" rules can be complex and interdepedent, they're not computed by switch itself
   (c) Instead computed, installed, and updated by a remote, logically centralized controller

7.2 OpenFlow Flow Table

- match plus action table is called flow table. Each entry in table has three key components

  1. Set of header field values (the "match"): Patterns used to match against incoming packet. Matching performed very fast, by TCAMs
  2. Set of counters (the "statistics"): Counters are updated as packets match rule. Crucial for network management and analystics. Coutners track number of packets matched by rule and time since rule was last updated
  3. Set of Actions (the "action"): Instructiosn to execute when a packet matches a rule. Can be a list of actions (e.g. "rewrite source port to 80, then forward to port 3")

- If packet arrives and matches no entry in flow table, can either be dropped, or more commonly, forward to remote controller for a decision. Controller can decide what to do and potentially install a new flow rule for this packet's flow

7.3 Template Example

- X

  1. Y

8 4.5 Middleboxes

8.1 Concept Overview: Beyond the Router

- Middlebox defined as "any intermediary box performing functions apart from normal, standard functions of an IP router on data path between source host and destination

host"

- Sit on forwarding path but modify, inspect, or filter packets based on policies often related to security, performance, op application-specific reqs
- Often violate end-to-end principle by processing information up to application layer (layer 7)or modifying network layer (layer 3) headers, thereby breaking strict separation of layers

8.2 A Taxonomy of Middlebox Services

Functions categorized into three categories

- NAT Translation
  1. NAT boxes are middleboxes that perform private network addressing
  2. Rewrite datagrams header's source/dest IP addresses and, transport-layers src/dest port numbers
- Security Services
  1. Firewalls
     (a) Most common security middlebox
     (b) Block traffic based on wide range of header-field values (IP addresses, port numbers, protocol type)
  2. Instrusion Detection/Prevention System (IDS/IPS)
     (a) Devices perform "deep packet inspection", examining packets payload
     (b) Can redirect packets for further analysis, detect known attack patterns ("signatures"), and filter malicious packets
  3. Application-Level Filters
     (a) Email gateways are prime example
     (b) Set on email delivery path and filter messags based on content, blocking spam, phishing attempts, and security threats
- Performance Enhancement
  1. Load Balancers: Devices intercept incoming service requests and distribute them across a farm of backend servers to balance computational load
  2. Content Caches: Devices store copies of popular content close to users to reduce latency and save network bandwidth
  3. Traffic Compressors: Compress data streams on the fly to save bandwidth, especially over expensive or slow links

8.3 The Problem with Middleboxes: Cost and Complexity

- Capital Cost: each new function (firewall, NAT, load balancer) has required its own specialized, often proprietary, hardware "box"
- Operational Cost: each box runs its own separate software stack and requires separate management, configuration, and upgrades. Admin needs special skills to manage such

    1. Y

8.4 The Solution: Network Function Virtualization (NFV)

- NFV's goal is to "unbundle" function from dedicated hardware
- Mechanism

    1. Instead of buying firewall appliance, network operator buys commodity hardware (standard servers, switches, and storage)
    2. Run middlebox functions as software on top of generic hardware, using virtual machines or containers

- Conenction to SDN

    1. NFV is logical extension of SDN
    2. SDN unbundles router, separating control-plane software from data-plan hardware
    3. NFV unbundles all other network functions (middleboxes) in same way

8.5 Architectural Principles of the Internet (A Sidebar)

- The "Architectural Abomination" View

    1. Original Sin: Middleboxes violate clean separation between network layer and layers above/below it
    2. Layer-Breakers
        (a) Original Internet had a "simple" network core (routers just forward IP packets) and "smart" edges (hosts run transport and application logic)
        (b) NAT box is a router (L3) that peeks into and rewrites port numbers (L4)
        (c) Firewall is a router (L3) that blocks packets based on TCP flags (L4) or even HTTP URLs (L7)
        (d) Email gateway (L3) filters packets based on application-layer content (email body)

- The "Pragmatic" View

    1. Counter argument is middleboxes "exist for important and permanent reasons"

2. Fill critical needs (security, address translation) that original architecture failed to provide
3. Argument from this camp is that we will ahve more, not fewer, middleboxes in future, so must learn how to build and manage them properly (e.g. using NFV and SDN)

- The IP Hourglass (The "Narrow Waist")
  1. Concept: Internet protocol stack has narrow waist at network layer
  2. Above Waist: Innumerable application-layer protocols and transport protocols
  3. Below Waist: Innumerable link-layer technologies
  4. Wait: One single, universisal protocol: IP

8.6 4.5 Section-Wide Summary

Key Terms

- **Middlebox**: In-network device performing non-standard IP forwarding functions such as NAT, firewalling, load balancing.
- **Network Function Virtualization (NFV)**: Implementing middlebox functions in software on commodity hardware.
- **IP Hourglass**: The Internet architecture with a narrow waist at the IP layer.
- **End-to-End Argument**: Idea that intelligence should reside at end hosts, while the network core remains simple.

Core Relationships

- Middleboxes address practical needs (security, address scarcity) that original Internet architecture did not solve.
- NFV improves cost, flexibility, and deployment by separating middlebox software from hardware.
- Middleboxes challenge the IP Hourglass (by adding complexity to the core) and the End-to-End Argument.

Key Insights / Takeaways

- The IP core is no longer simple: it is filled with middleboxes essential for modern networking.
- SDN (match-plus-action) provides a way to program and manage middlebox-like behaviors centrally.
- The tension between architectural purity (End-to-End) and practical necessity (Middleboxes) will define the future of networking.

8.7 Template Example

- X
    1. Y