4. Advanced Routing Mechanisms

PA159: Net-Centric Computing I.

Eva Hladká

Faculty of Informatics Masaryk University

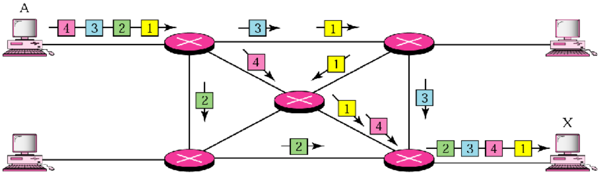
Autumn 2010

Lecture Overview I

Lecture Overview I

Routing in General

- Internet on the L3 – datagram apporach to packet switching
  - upper layer data are encapsulated into datagrams
  - datagrams (their fragments) travel through the network independently on each other
  - the global knowledge of the network's topology is problematic



- **Routing** = the process of finding a path in the network between two communicating nodes
  - the route/path has to satisfy certain constraints
  - influenced by several factors:
    - *static ones:* network topology
    - *dynamic ones:* network load

A Real Network Example

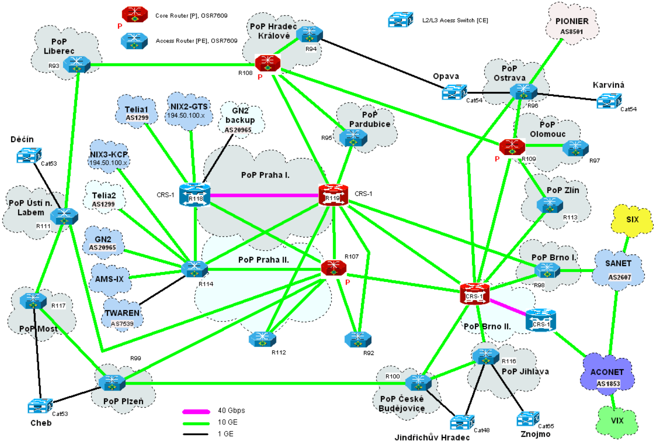


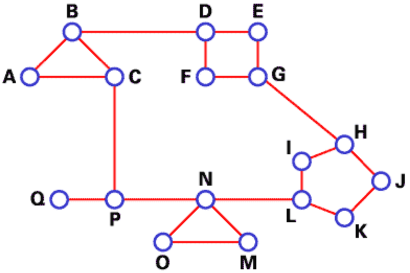
Figure: The topology of the IP/MPLS layer of the CESNET2 network.

Routing – the goal

- the main goal of routing is:
  - to find optimal paths
    - the optimality criterion is a *metric* – a cost assigned for passing through a network
  - to deliver a data packet to its receiver
- the routing *usually* does not deal with the whole packet path
  - the router deals with just a single step – to whom should be the particular packet forwarded
    - somebody "closer" to the recipient
    - so-called *hop-by-hop* principle
  - the next router then decides, what to further do with the received packet

Routing – Mathematical View

- the routing can be seen as a problem of graph theory
- a network can be represented by a graph, where:
  - nodes represent routers (identified by their IP addresses)
  - edges represent routers' interconnection (a data link)
  - edges' value = the communication cost
    - based on the employer metric – hop count, links' delay, links' usage, etc.
  - *the goal:* to find paths having minimal costs between any two nodes in the network



Routing – Mathematical View

Graph Theory Algorithms

Two very important algorithms have profound impact on data networks:
Bellman-Ford algorithm and **Dijkstra's algorithm**

- both allow to compute shortest paths from a single source
  - to a single destination – Bellman-Ford, complexity $O(LN)$
  - to all the destinations – Dijkstra, complexity $O(N^2)$ (can be improved to $O(L + NlogN)$)
- both of them have *centralized* and *distributed* variants
- variants for *widest-path computation* also exist
  - so-called *widest-path routing algorithms*
    - algorithms, that use a *non-additive concave property* to define distance cost between two nodes
    - e.g., bandwidth – the bandwidth of a path is determined by the link with the minimum available bandwidth
    - i.e., if $m(P) = min\{m(n_1, n_2), m(n_2, n_3), ..., m(n_i, n_j)\} \Rightarrow$ *concave property*
- further details:
  - PB165: Graphs and networks (prof. Matyska, doc. Hladká, dr. Rudová)

Routing – basic approaches

distributed	vs.	centralized
hop-by-hop	vs.	source-based
deterministic	vs.	stochastic
single-path	vs.	multi-path
dynamic path selection	vs.	static path selection

INTERNET

Distributed Routing – Basic Approaches

Basic approaches to distributed routing:

- *Distance Vector (DV)* – Bellman-Ford algorithm
  - the neighboring routers periodically (or when the topology changes) exchange complete copies of their routing tables
  - based on the content of received updates, a router updates its information and increments its *distance vector number*
    - a metric indicating the number of hops in the network
  - i.e., *"all pieces of information about the network just to my neighbors"*
- *Link State (LS)* – Dijkstra's algorithm
  - the routers periodically exchange information about states of the links, to which they are directly connected
  - they maintain complete information about the network topology – every router is aware of all the other routers in the network
  - once acquired, the Dijkstra algorithm is used for shortest paths computation
  - i.e., *"information about just my neighbors to everyone"*

Distributed Routing – Link State vs. Distance Vector

Link State

- *Complexity:*
  - every node has to know the cost of every link in the network ⇒ $O(nE)$ messages
  - once a link state changes, the change has to be propagated to *every* node
- *Speed of convergence:*
  - $O(n^2)$ alg., sends $O(nE)$ messages
  - sustains from oscillations
- *Robustness:*
  - wrongly functional/compromised router spreads wrong information just about the links it is directly connected to
  - every router computes routing tables on its own ⇒ separated from routing information propagation ⇒ a form of robustness
- *Usage:*
  - suitable for large networks

Distance Vector

- *Complexity:*
  - once a link state changes, the change has to be propagated just to the *closest neighbors*; it is further propagated just in cases, when the changed state leads to a change in the current shortest paths tree
- *Speed of convergence:*
  - may converge more slowly than LS
  - problems with routing loops/cycles, *count-to-infinity* problem
- *Robustness:*
  - bad computation is spread through the network ⇒ may lead to a “confusion” of other routers (bad routing tables)
- *Usage:*
  - suitable just for smaller networks

Distributed Routing – Path Vector

Path Vector (PV)

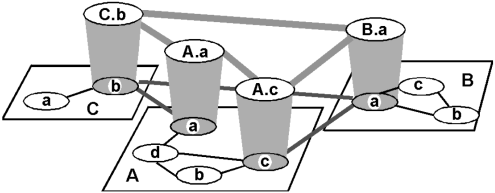
- a variant of DV routing
- in comparison with the DV, whole paths are sent in the PV (not only the end nodes)
  - allows a simple detection of loops
  - allows a definition of rules/policies (friendly vs. non-friendly ASs)

Autonomous Systems

- the goal of Internet's division into *Autonomous Systems* is:
  - a reduction of routing overhead
    - simpler routing tables, a reduction of exchanged information, etc.
  - a simplification of the whole network management
    - particular internets are managed by various institutions/organizations
- autonomous systems = domains
  - a 16bit identifier is assigned to every AS/domain
    - *Autonomous System Number (ASN)* – RFC 1930
    - assigned by *ICANN (Internet Corporation For Assigned Names and Numbers)*
  - correspond to administrative domains
    - networks and routers inside a single AS are managed by a single organization/institution
    - e.g., CESNET, PASNET, ...
  - a distinction according to the way an AS is connected to the Internet:
    - *Stub AS*
    - *Multihomed AS*
    - *Transit AS*

Autonomous Systems – routing

- separated routing because of scalability reasons:
  - *interior routing*
    - routing inside an AS
    - under the full control of AS's administrator(s)
    - the primary goal is the performance
    - so-called *Interior Gateway Protocols (IGP)* (e.g., RIP, OSPF, (E)IGRP, IS-IS)
  - *exterior routing*
    - routing among ASs
    - the primary goal is the support of defined policies and scalability
    - so-called *Exterior Gateway Protocols (EGP)* (e.g., BGP-4)
  - a cooperation of interior and exterior routing protocols is necessary



Autonomous Systems – routing

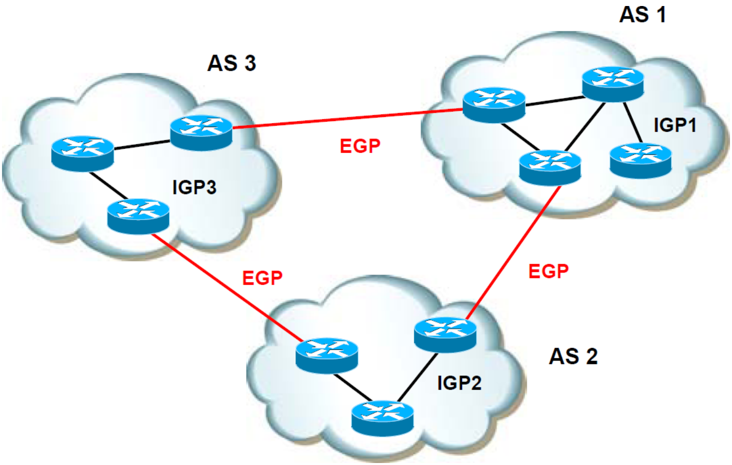


Figure: Interior (IGP) vs. Exterior (EGP) routing protocols.

Lecture Overview I

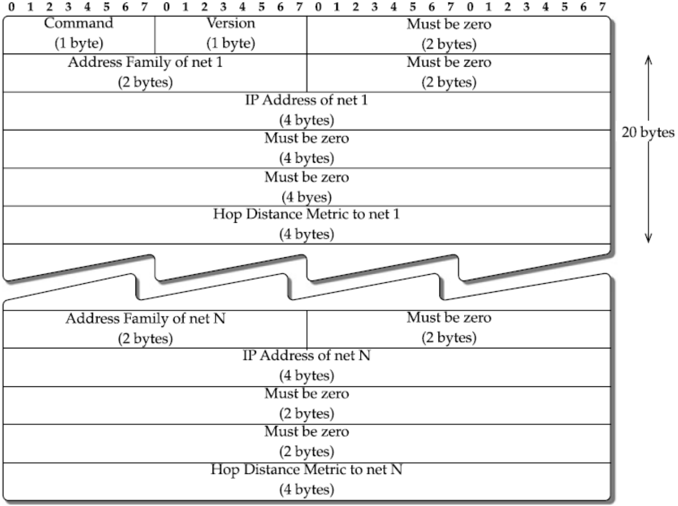
RIP protocol

Routing Information Protocol (RIP)

- the principal actor of the DV routing
  - RIPv1 (RFC 1058) – the first routing protocol used in TCP/IP-based network in an intradomain environment
  - RIPv2 (RFC 1723) – adds several features (e.g., explicit masking and an authentication of routing information)
  - RIPng (RFC 2081) – RIPv2's extension to support IPv6 addresses/networks
- the number of hops is used as a metric
  - transfer of a packet between two neighboring routers = 1 hop
- the routers send the information periodically every 30 seconds
  - messages sent over UDP protocol
  - supports triggered updates when a state of a link changes
  - timeout 180s (detection of connection errors)
- usage:
  - suitable for small networks and stable links
  - not advisable for redundant networks

RIP protocol – version 1

Message Format I.



RIP protocol – version 1

Message Format II.

- **Command** – indicates, whether the message is a request (a router is asking its neighbor for DV information) or a response
- **Version** – RIP version
- **Address family identifier** – identifies the address family (set to 2 for the IP address family)
- **IP address** – the destination network (identified by a subnet or a host)
- **Metric** – hop count to the destination (a number in the range (1..16), 16 = infinity)

RIPv1 messages are *broadcast*.

RIP protocol – version 1

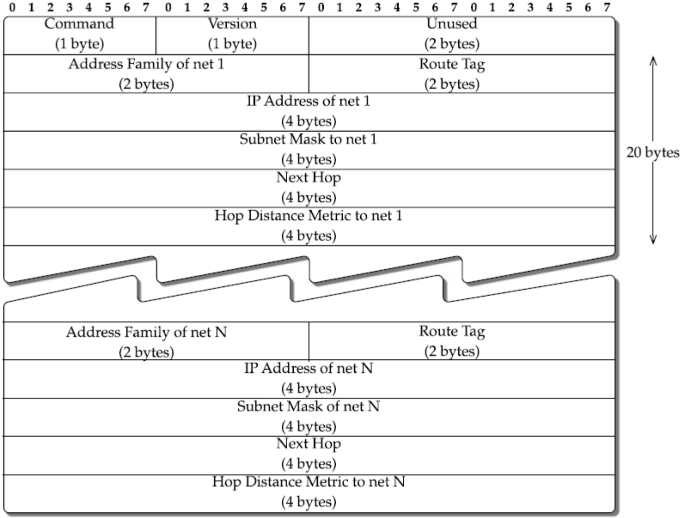
Problems Analysis

RIPv1 suffers from several problems:

- slow convergence and problems with routing loops/cycles – imposed by DV approach
- infinity $= 16 \Rightarrow$ the RIPv1 cannot be used for networks with minimal amount of hops between any two routers > 15
- has no way (no field in the messages) to indicate anything specific about the network being addressed
  - RIPv1 assumes that an address included follows a Class A, Class B, or Class C boundary implicitly
  - $\Rightarrow$ it *does NOT support variable length subnet masking*

RIP protocol – version 2

Message Format I.



RIP protocol – version 2

Message Format II.

New fields introduced by RIPv2:

- **Route tag** – used to differentiate internal routes within a RIP routing domain from external routes (the ones obtained from an external routing protocol)
- **Subnet mask** – allows routing based on subnet instead of doing classful routing (eliminates a major limitation of RIPv1)
- **Next hop** – an advertising router might want to indicate a next hop that is different from itself

RIPv2 messages are *multicast* on `224.0.0.9`.

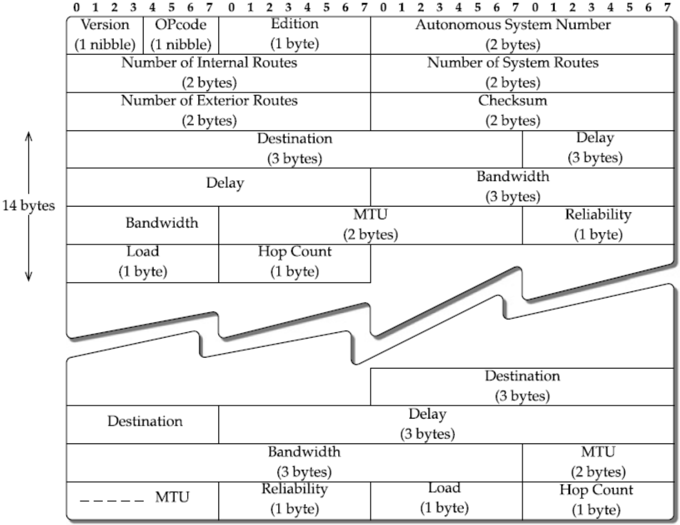
Interior Gateway Routing Protocol (IGRP)

Interior Gateway Routing Protocol (IGRP):

- developed by Cisco primarily to overcome the hop count limit and hop count metric of RIPv1
- differs from the RIPv1 in the following ways:
  - *DV updates include five different metrics for each route*
  - runs directly over IP with protocol (type field set to 9)
  - allows multiple paths for a route for the purpose of load balancing
  - external routes can be advertised
- *does NOT support variable length subnet masking*

Interior Gateway Routing Protocol (IGRP)

Message Format I.



Interior Gateway Routing Protocol (IGRP)

Message Format II.

- **Version** – set to 1
- **Opcode** – ≈ *Command* field in RIPv1
- **Edition** – counter incremented by the sender (prevents from receiving an old update)
- **Autonomous system number** – ID number of an IGRP process
- **Number of interior routes** – a field to indicate the number of routing entries in an update message that are subnets of a directly connected network
- **Number of system routes** – a counterpart of the number of interior routes
- **Number of exterior routes** – the number of route entries that are default networks
- **Checksum** – value calculated on the entire IGRP packet (header + entries)

- **Destination –** the destination network for which the distance vector is generated (just *3 B are used!*)
- **Delay**, **Bandwidth**, **Reliability**, **Load –** fields for *composite metric computation*
- **Hop count –** a number between 0 and 255 used to indicate the number of hops to the destination
- **MTU –** the smallest MTU of any link along the route to the destination

IGRP messages are *multicast* on `224.0.0.10`.

Interior Gateway Routing Protocol (IGRP)

Composite Metric Computation I.

The IGRP uses a composite metric to compute a link cost:

- included to provide flexibility to compute better or more accurate routes from a link cost rather than just using a hop count
- based on four factors: *bandwidth (B)*, *delay (D)*, *reliability (R)*, and *load (L)*
  - along with five nonnegative real-number coefficients (K1, K2, K3, K4, K5) for **weighting these factors**
    - set on the routers
- The composite metric, C ("cost of a link"), is given as follows:

$$C = \begin{cases} (K_1 \times B + K_2 \times \dfrac{B}{256 - L} + K_3 \times D) \times (\dfrac{K_5}{R + K_4}), & \text{if } K_5 \neq 0 \quad (1) \\ K_1 \times B + K_2 \times \dfrac{B}{256 - L} + K_3 \times D, & \text{if } K_5 = 0 \quad (2) \end{cases}$$

Interior Gateway Routing Protocol (IGRP)

Composite Metric Computation II.

- example: $\frac{K_5}{R+K_4}$ considers the reliability of a link
  - i.e., if $K_5 = 0$ (the above part is not included), all the links have the same level of reliability
- the default, often used case: $K_1 = K_3 = 1$ and $K_2 = K_4 = K_5 = 0$
  - the composite metric reduces: $C_{default} = B + D$
  - How can we compare bandwidth (kbps, Mbps) with delay (sec, milisec)?
    - a transformation process is necessary to map the raw parameters to a comparable level
    - see the literature
- further details:
  Medhi, D. and Ramasamy, K.: Network Routing: Algorithms, Protocols, and Architectures.

Interior Gateway Routing Protocol (IGRP)

Analysis

- the protocol message includes all the different metric components rather than the composite metric
  - $\Rightarrow$ the composite metric is left to a router to be computed
- it is extremely important to ensure that each router is configured with the same value of the coefficients K_1, K_2, K_3, K_4, K_5
  - if NOT set equally, the routers' view of the shortest paths would be different
    - may cause routing problems

Enhanced Interior Gateway Routing Protocol (EIGRP)

Enhanced Interior Gateway Routing Protocol (EIGRP):

- another routing protocol developed by Cisco
- it enhances IGRP in many ways (e.g., it provides loop-free routing, provides reliable delivery, allows variable length subnet masking, etc.)
- the composite metric remains the same as in IGRP
- originally designed for IPv4 only, IPv6 version proposed afterwards

DV Protocols Comparison

Protocol	RIPv1	RIPv2	IGRP	EIGRP	RIPng
Address Family	IPv4	IPv4	IPv4	IPv4	IPv6
Metric	Hop	Hop	Composite	Composite	Hop
Information Communication	Unreliable, broadcast	unreliable, multicast	Unreliable, multicast	Reliable, multicast	Unreliable, multicast
Routing Computation	Bellman–Ford	Bellman–Ford	Bellman–Ford	Diffusing computation	Bellman–Ford
VLSM/CIDR	No	Yes	No	Yes	v6-based
Remark	Slow convergence; split horizon	Slow convergence; split horizon	Slow convergence; split horizon	Fast, loop-free convergence; chatty protocol	Slow convergence; split horizon

Figure: Comparison of protocols in the distance vector protocol family.

Lecture Overview I

Open Shortest Path First (OSPF) I.

Open Shortest Path First (OSPF)

- currently the mostly used LS protocol
  - gathers link state information from available routers and constructs a topology map of the network
- metric: *cost*
  - NO hop-count
  - a number (in the range between 1 and 65535) assigned to each router's network interface
  - the lower the number is, the better the link/path is (i.e., will be preferred)
  - by default, every interface is automatically assigned a cost derived from the link's throughput
    - $cost = 100000000/bandwidth$ (bw in bps)
    - might be manually edited

Open Shortest Path First (OSPF) II.

- features:
  - *message authentication*
    - up to OSPFv2
    - OSPFv3 (running on IPv6) no longer supports protocol-internal authentication (instead, it relies on IPv6 protocol security (IPsec))
  - *routing areas*
    - next layer of hierarchy – autonomous systems can be divided into subdomains (*routing areas*)
    - to simplify administration and optimize traffic and resource utilization (lower amount of messages exchanged among same-area routers)
  - *load-balancing*
    - OSPF can make use of more outgoing links with the same (lowest) cost
    - so-called *Equal-Cost MultiPath (ECMP)*
  - *CIDR/Variable Length Subnet Mask support*
- OSPF messages are encapsulated directly in IP datagrams (protocol number 89)
  - OSPF handles its own error detection and correction functions
  - multicast is used for OSPF messages delivery (`224.0.0.5` and `224.0.0.6` for IPv4, `FF02::5` and `FF02::6` for IPv6)

Open Shortest Path First (OSPF) III.

Message Format I.

0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
Version (1 byte)	Type (1 byte)	Packet Length (2 bytes)	
Router ID (4 bytes)			
Area ID (4 bytes)			
Checksum (2 bytes)		Authentication Type (2 bytes)	
Authentication (4 bytes)			
Authentication (4 bytes)			

Figure: OSPF packet common header.

OSPF messages:

- *Hello Packet*
- *Database Description Packet*
- *Link State Request Packet*
- *Link State Update Packet*
- *Link State Acknowledgement Packet*

Intermediate System To Intermediate System (IS-IS) I.

- **Intermediate System To Intermediate System (IS-IS)**
  - standardized by the ISO as a mechanism for communication between network devices (termed *Intermediate Systems*)
    - developed at the same time as the OSPF
  - originally designed for ISO-developed OSI Network Layer service called *CLNS (Connectionless Network Service)*
  - later extended to support routing of IP datagrams – called *Integrated IS-IS* or *Dual IS-IS*
    - RFC 1195
- *key similarities with the OSPF:*
  - both protocols provide network hierarchy through two-level areas
  - both protocols use *Hello packets* to initially form adjacencies and then continue to maintain them
  - both protocols support variable length subnet masks
  - both protocols maintain a link state database and perform shortest path computation using the Dijkstra's algorithm

Intermediate System To Intermediate System (IS-IS) II.

- *key differences with the OSPF:*
  - while OSPF packets are encapsulated in IP datagrams, IS-IS packets are encapsulated directly in link layer frames
  - IS-IS's run on top of layer 2 makes it relatively safer from spoofs or attacks
  - IS-IS is neutral regarding the type of network addresses for which it can route
    - easily adapted to support IPv6
    - OSPF needed a major overhaul (OSPFv3) in order to support IPv6
  - IS-IS allows overload declaration – an overloaded router may not be considered in path computation
  - OSPF's link metric value is in the range 1 to $65,535$, while IS-IS's metric value is in the range 0 to 63 (narrow metric)
    - further extended to the range 0 to $16,777,215$ (wide metric)
  - OSPF provides a richer set of extensions and added features
  - IS-IS is less "chatty" and can scale to support larger networks

Lecture Overview I

Border Gateway Protocol (BGP) I.

Border Gateway Protocol (BGP)

- currently version 4 (*BGP-4*)
  - RFC 1771
- proposed due to Internet's grow and demands on complex topologies support
  - supports redundant topologies, deals with loops/cycles, etc.
- used to communicate information about networks currently residing in an autonomous system to other autonomous systems
  - the exchange is done by setting up a communication session between bordering autonomous systems
  - the communication channel is set on top of the TCP protocol
    - the BGP relies on a fully reliable transport protocol
- allows a definition of routing rules (policies)
- uses a hop count metric
- uses CIDR for paths' aggregation

Border Gateway Protocol (BGP) II.

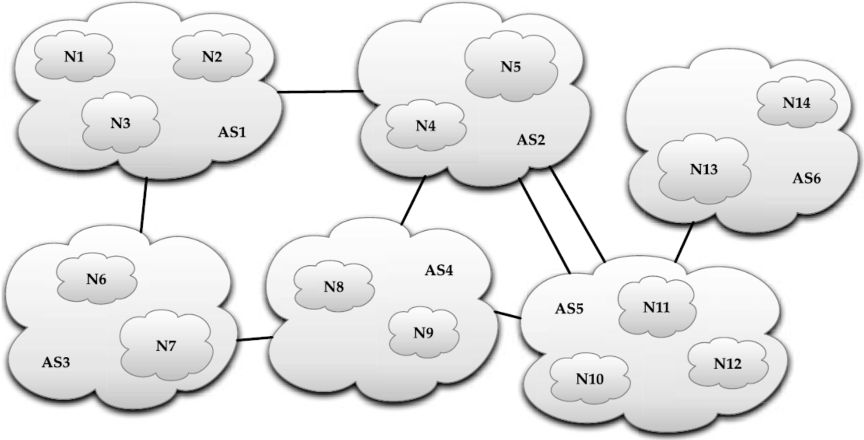


Figure: The BGP's view of the Internet architecture.

Border Gateway Protocol (BGP) III.

Advertisements

- the BGP basis upon *advertisements* sent among BGP peers:
  - sent through reliable point-to-point communication channels
    - TCP, port 179
  - an advertisement consists of:
    - a destination network address (using CIDR notation)
    - path attributes (e.g., the ASs on the path, next-hop router, etc.)
- once paths are advertised to an AS, a *routing policy* takes place
  - a routing policy defines, which ASs are allowed to transit data through the particular AS, to which ASs the data are allowed to be forwarded, etc.
  - peering contracts are big bussiness (no standards exist)
  - if a routing policy is not defined, the shortest path is chosen

Border Gateway Protocol (BGP) III.

Message Types

- *OPEN* – initiates a BGP session between a pair of BGP routers
  - allows routers to introduce themselves and to announce their capabilities
  - includes router's authentication information
- *UPDATE*
  - used to advertise routing information from one BGP router to another ("push model")
  - used to withdraw a previously announced advertisment
    - the advertised information is valid *until being explicitly withdrawn!*
- *KEEPALIVE*
  - exchanged when there is no other traffic
  - allows the BGP routers to distinguish between a failed connection and a BGP peer that has nothing to say
- *NOTIFICATION* – used to close a session or to report an error
  - e.g., rejecting an OPEN message or reporting a problem with UPDATE message
- *ROUTE-REFRESH* – a specific request to re-advertise all of the routes in router's routing table using UPDATE messages
  - not defined in the original BGP-4 (RFC 1771), but added by RFC 2918

Border Gateway Protocol (BGP) IV.

Routing table size

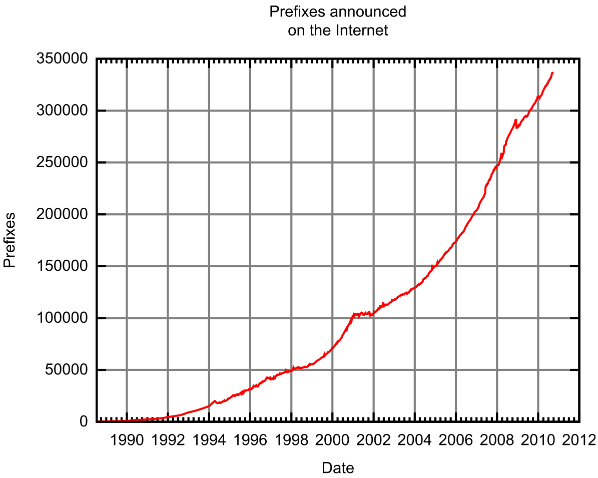


Figure: The growth of the BGP Table.

Border Gateway Protocol (BGP) IV.

Number of ASs on the Internet

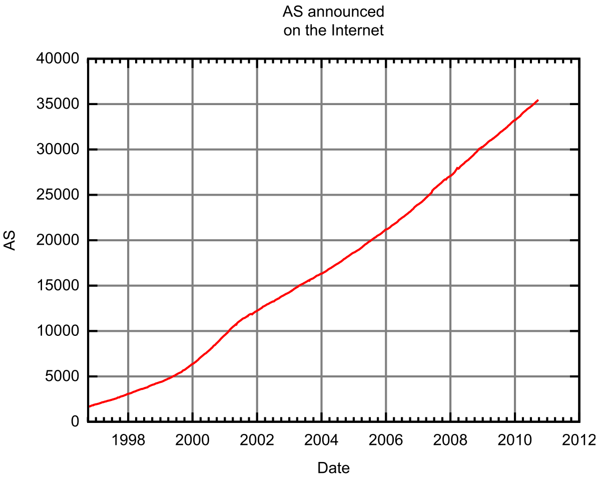


Figure: The number of autonomous systems on the Internet.

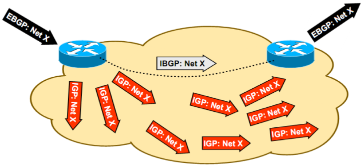
Border Gateway Protocol (BGP) V.

Internal BGP (IBGP)

The basic problem: *How to make external destinations (ASs) reachable from all the routers within an AS?*

⇒ **Internal BGP (IBGP)**

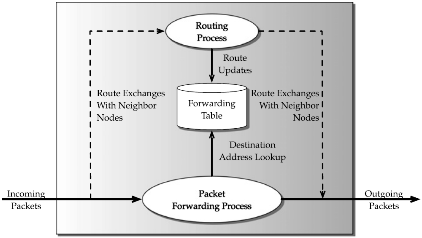
- a mechanism to provide information about adjacent ASs to internal routers of a particular AS
  - all IBGP peers within a same AS are fully meshed
  - peer announces routes received via eBGP (external BGP) to IBGP peers
  - **but:** IBGP peers do not announce routes received via IBGP to other IBGP peers
  - the learned routes are further distributed via interior routing protocol (IGP)



Lecture Overview I

Router Functions

- a router must perform two fundamental tasks: *routing* and *packet forwarding*
  - the **routing process** constructs a view of the network topology and computes the best paths
    - based on the information exchanged between neighboring routers using routing protocols
    - the best paths are stored in a data structure called a *forwarding table*
  - the **packet forwarding process** moves a packet from an input interface ("ingress") to the appropriate output interface ("egress")
    - based on the information contained in the forwarding table
    - the performance of the forwarding process determines the overall performance of the router



Router Functions

Basic forwarding functions I.

IP Header Validation

- every IP packet arriving at a router needs to be validated
  - e.g., the version number of the protocol is correct, the header length is valid, checksum is correct, etc.

Packet Lifetime Control

- decrementing the TTL field to prevent packets from getting caught in the routing loops forever
- it the TTL is zero or negative, the packet is discarded
  - and an ICMP message is generated and sent to the original sender

Checksum Recalculation

- since the value of the TTL has been modified, the header checksum needs to be updated

Router Functions

Basic forwarding functions II.

Route Lookup

- packet destination address is used to search the forwarding table for determining the output port

Fragmentation

- the router needs to split the packet into multiple fragments when the MTU of the outgoing link is smaller than the size of the packet that needs to be transmitted

Handling IP Options

- a packet may indicate that it requires special processing needs at the router

Router Functions

Complex forwarding functions

Packet Classification

- for distinguishing packets, a router might need to examine not only the destination IP address but also other fields
  - such as source address, destination port, and source port, etc.

Packet Translation

- a router that acts as a gateway to a NAT network needs to support network address translation

Traffic Prioritization

- a router might need to guarantee a certain quality of service to meet service level agreements

Router Functions

Routing process functions

Routing Protocols

- routers need to implement different routing protocols (e.g., OSPF, BGP, and RIP) for maintaining peer relationships by sending and receiving route updates from adjacent routers

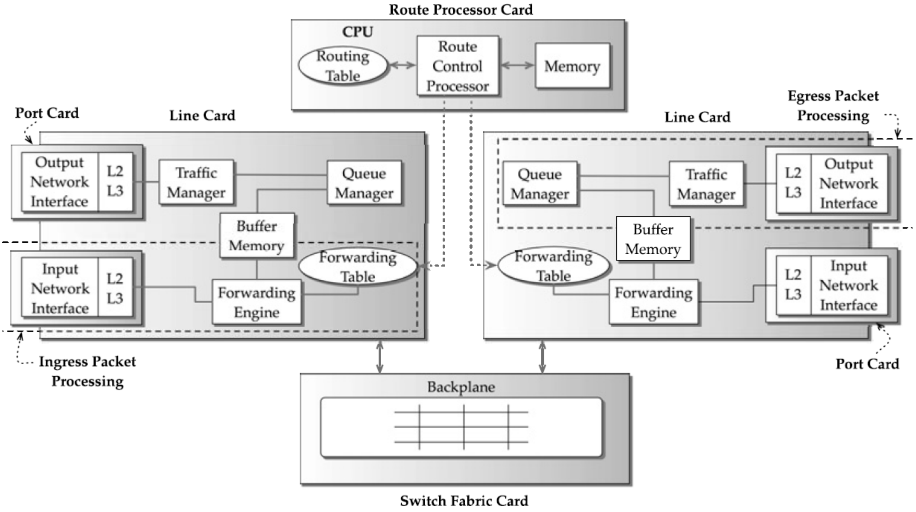
System Configuration

- a router needs to implement various functions enabling the operators to configure various administrative tasks
    - configuring the interfaces, routing protocol keep alives, rules for classifying packets, etc.

Router Management

- in addition to the configuration tasks, the router needs to be monitored for continuous operation
    - e.g., SNMP support

Router Elements



Router Elements II.

Network Interfaces

- a network interface contains many ports that provide the connectivity to physical network links
  - a port is specific to a particular type of network physical medium (Ethernet, Sonet, etc.)

Forwarding Engines

- responsible for deciding to which network interface the incoming packet should be forwarded
  - by consulting a *forwarding table* = **Address/Route Lookup**

Queue Manager

- provides buffers for temporary storage of packets when an outgoing link from a router is overbooked
- when these buffer queues overflow due to congestion, the queue manager selectively drops packets

Traffic Manager

- responsible for prioritizing and regulating the outgoing traffic, depending on the desired level of service

Router Elements III.

Backplane

- provides connectivity for the network interfaces
  - packets from an incoming network interface can be transferred to the outgoing network interface

Route Control Processor

- responsible for implementing and executing routing protocols
  - maintains a *routing table* that is updated whenever a route change occurs
    - based on the contents of the routing table, the *forwarding table* is computed and updated
- runs the software to configure and manage the router
- performs complex packet-by-packet operations
  - e.g., handling errors during packet processing
    - e.g., sending an ICMP message to the origin when packet's destination address cannot be found in the forwarding table

(a) Routing table

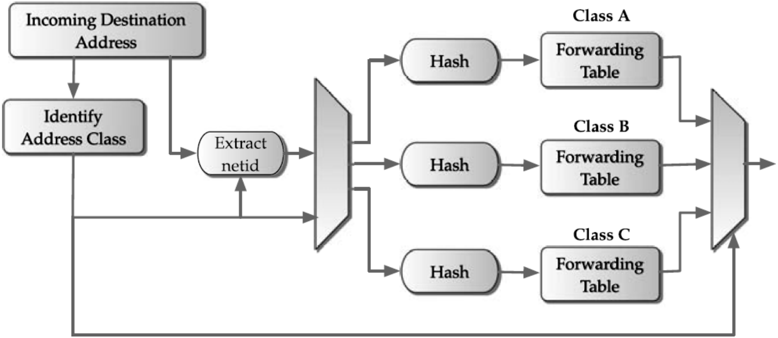
IP prefix	Next hop
10.5.0.0/16	192.168.5.254

(b) Forwarding table

IP prefix	Interface	MAC address
10.5.0.0/16	eth0	00:0F:1F:CC:F3:06

Address Lookup with Classful Addressing

- with the classful addressing scheme, the forwarding of packets is straightforward
  - routers need to examine only the network part of the destination address
  - ⇒ the forwarding table needs to store just a single entry for routing the packets destined to all the hosts attached to a given network



Address Lookup with CIDR – Longest Prefix Matching

- address lookup with CIDR is *more difficult* since:
  1. a destination IP address does not explicitly carry the netmask information
  2. the prefixes in the forwarding table against which the destination address needs to be matched can be of arbitrary lengths

Address Lookup with CIDR – Longest Prefix Matching

Requirements I.

Lookup Speed

- Internet traffic measurements show that roughly 50 % of the packets that arrive at a router are TCP-acknowledgment packets, which are typically 40-byte long
- thus, the prefix lookup has to happen in the time it takes to forward such a minimum-size packet (40 bytes)
  - known as *wire-speed forwarding*
- wire-speed forwarding for:
  - 1 Gbps link ⇒ prefix lookup should not exceed 320 nanosec
  - 10 Gbps link ⇒ prefix lookup should not exceed 32 nanosec
  - 40 Gbps link ⇒ prefix lookup should not exceed 8 nanosec

1 Gbps computed as: $\frac{40\ bytes \times 8\ bits/byte}{1 \times 10^9\ bps} = 320\ nanosec$

Address Lookup with CIDR – Longest Prefix Matching

Requirements II.

Memory Usage

- i.e., the amount of memory consumed by the data structures of the algorithm
- a memory-efficient algorithm can effectively use the fast but small cache memory

Scalability

- algorithms are expected to scale both in speed and memory as the size of the forwarding table increases

Updatability

- route changes occur fairly frequently
  - rates varying from a few prefixes per second to a few hundred prefixes per second
- ⇒ the route changes require updating the forwarding table data structure in the order of milliseconds or less

Address Lookup with CIDR – Longest Prefix Matching

Algorithms I.

Naive Algorithms

- the simplest algorithm for finding the best matching prefix is a *linear search of prefixes*
- time complexity is $O(N)$
  - N ... number of prefixes in a forwarding table
  - useful if there are very few prefixes to search; otherwise the search time degrades as N becomes large

Trie-based Algorithms

- *note: "trie" comes from "retrieval"*, not from "tree"
- several variants proposed:
  - Binary Tries
  - Multibit Tries
  - Compressed Multibit Tries

Address Lookup with CIDR – Longest Prefix Matching

Algorithms II.

P_1	0*
P_2	00001*
P_3	001*
P_4	1*
P_5	1000*
P_6	1001*
P_7	1010*
P_8	1011*
P_9	111*

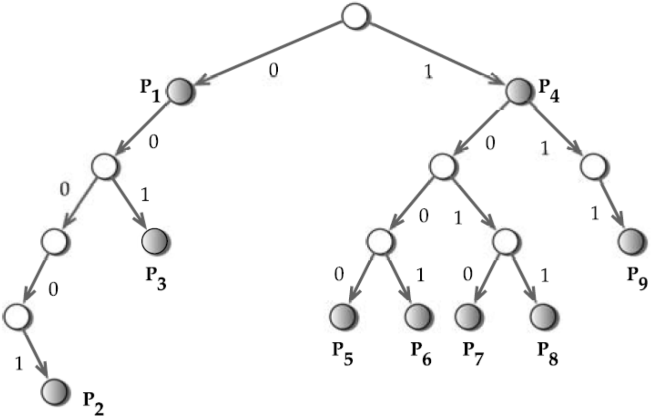


Figure: Binary trie data structure example.

Address Lookup with CIDR – Longest Prefix Matching

Algorithms II.

Other Approaches

- Search by Length Algorithms
- Search by Value Approaches
- Hardware Algorithms
  - RAM-Based Lookup, Ternary CAM-Based Lookup, Multibit Tries in Hardware, etc.

Further details:

- *Medhi, D. and Ramasamy, K.: Network Routing: Algorithms, Protocols, and Architectures.*

IP Packet Filtering and Classification I.

Importance of **Packet Classification/Filtering**:

- *Providing preferential treatment for different types of traffic*
  - to provide different service guarantees for different types of traffic, an ISP might maintain different paths for the same source and destination addresses
- *Flexibility in accounting and billing*
  - an ISP needs flexible accounting and billing based on the traffic type
    - $\Rightarrow$ different traffic can be charged at different prices
- *Preventing malicious attacks*
  - the ability to identify malicious packets and drop them at the point of entry
- *etc.*

IP Packet Filtering and Classification II.

The criteria for classification are expressed in terms of *rules* or *policies*

- using the header fields of the packets
  - ⇒ the forwarding engine needs to examine packet fields other than the destination address to identify the context of the packets
  - and to perform required processing/actions in order to satisfy user requirements
- a collection of such rules/policies – *rule/policy database*, *flow classifier* or simply *classifier*
- each rule specifies:
  - a *flow* to which a packet may belong (based on expressed conditions)
    - exact match, prefix match, range match, regular expression match, etc.
  - an *action* which has to be applied to packets belonging to the flow
    - like permit, deny, encrypt, etc.
- a packet may match more than one rule in the classifier
  - a *cost* is associated with each rule to determine an unambiguous match
  - ⇒ the goal is to find the rule with the least cost that matches a packet's header
    - when the rules are placed in the order based on their cost → the goal is to find the *earliest matching rule*

IP Packet Filtering and Classification

Algorithms

- *Naive Algorithms*
  - storing the rules in a linked list in the order of increasing cost
  - storage efficient, but seach-time inefficient (does not scale)
- *Two-dimensional Solutions*
  - Hierarchical Tries, Set Pruning Tries, Grid-of-Tries
- *d-dimensional Solutions*
- *Divide and Conquer Approaches*
  - Lucent Bit Vector, Aggregated Bit Vector, Cross-Producting, Recursive Flow Classification
- *Tuple Space Approaches*
- *Decision Tree Approaches*
  - Hierarchical Intelligent Cuttings (HiCuts), HyperCuts,
- *Hardware-Based Solutions*
  - Ternary Content Addressable Memory (TCAM)

Further details:

Medhi, D. and Ramasamy, K.: Network Routing: Algorithms, Protocols, and Architectures.