Architectures and Protocols for Integrated Networks

Intra-domain and Inter-domain Routing Protocols
How is the routing table built?

- **Path finding**
  - Paths from a device to any other device. Aggregated according to address prefixes.
  - Based on network topology.
  - Networks often provide multiple alternate paths, to achieve fault tolerance, load balancing, or higher capacity/performance.

- **Path evaluation and selection**
  - Based on network resources and load.
    - E.g., prefer paths with lower delivery delay (higher bandwidth, lower propagation delay).
  - Single best path (usually) or several alternate paths (load balancing).

- **Build the routing table**
  - Derived from selected paths.
  - Destination prefix, interface, next router.

<table>
<thead>
<tr>
<th>Destination prefix</th>
<th>Interface</th>
<th>Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>N1</td>
<td>i1</td>
<td>direct</td>
</tr>
<tr>
<td>N2</td>
<td>i2</td>
<td>R2</td>
</tr>
<tr>
<td>N3</td>
<td>i2</td>
<td>R2</td>
</tr>
<tr>
<td>N4</td>
<td>i2</td>
<td>R2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Static vs. dynamic routing

- **Static (fixed) routing**
  - Routing tables are created and maintained by manually entering static routes (i.e., no dynamic route update).
  - Used in simple, stable topologies, with no/few alternatives. Hosts typically use static routes.

- **Dynamic (adaptive) routing**
  - Routing tables are created and maintained by automatically collecting routing information, using routing protocols, and computing the best paths.
  - Routes are permanently adapted to topology changes caused by failures, to network congestion, etc.
  - Used in all non-trivial networks.

- **Mixed solutions**
  - Dynamic routing is often complemented by static routes used for, e.g., backup routes, default routes, etc.
Example: Shortest path routing

- Define link metrics and path metrics
  - Define an additive link metric (cost), e.g., based on delay, data rate.
  - Define the path metric (cost): Sum of link metrics.
  - “Shortest path” routing: Select the path with the lowest metric.

- Model the network as a labeled graph
  - Nodes: routers and hosts/networks.
  - Edges: network links.
  - Labels: metric assigned to each link.
    (In general, we need a directed graph.)

- Solve shortest path graph problem
  - For each router node (as root), find the shortest path spanning tree, i.e., shortest paths to all destinations.
  - E.g., using a variant of Dijkstra’s algorithm or Bellman-Ford algorithm.
## Routing protocols design issues

<table>
<thead>
<tr>
<th><strong>Issues</strong></th>
<th><strong>Goals</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Routing information traffic</td>
<td>Minimize the number and size of the messages exchanged for collecting and updating the routing information base. The overhead (bandwidth and processing time) can become important in large IP networks (especially, in the default-free Internet core).</td>
</tr>
</tbody>
</table>
| Convergence                  | Ensure that all routers reach a consistent perspective of the network after a change of network connectivity.  
   - Complex process: many exchanged messages.  
   - Fast convergence is essential: incorrect routing can occur during the time to convergence. |
| Stability                    | Avoid frequent reconfigurations due to minor changes in network connectivity (e.g., link delay or load variation).                                                                                                 |
| Security                     | Avoid corruption of the routing tables by incorrect routing information messages sent by malicious users.                                                                                                         |
Routing in the Internet

- **Administrative boundaries and routing policies**
  - The Internet is a federation of IP networks owned and operated by different organizations.
  - Each organization manages autonomously:
    - the connectivity and routing within its own network;
    - the policies for interconnection with other networks.

- **Autonomous System (AS): Unit of routing policy**
  - An AS is a connected group of networks having a single technical administration, consistent interior routing, and a single well defined (exterior) routing policy (RFC 1786,1930).
  - It is also called *(autonomous) routing domain*.
  - An AS is uniquely identified by an AS number.

  32-bit numbers since RFC 4893, 2007. Previously, 16-bit AS numbers. A routing domain is assigned an AS number only if it has distinct inter-domain routing policies. E.g., an ISP needs an AS number, but the single-homed, stub networks of its customers do not need their own AS numbers.
Intra-domain vs. Inter-domain Routing

**Intra-domain routing protocols**
- Exchange routing information *within* a routing domain (AS).
  - Interior Gateway Protocols (IGP).
- All routers in the AS exchange routing information.
- Route finding and selection is based on *IP network topology* and *performance metrics*.
- Examples: RIP, OSPF, IS-IS.

**Inter-domain routing protocols**
- Exchange routing information *between* routing domains (AS).
  - Exterior Gateway Protocols (EGP).
- Designated AS border routers exchange routing information.
- Route finding and selection is based on *AS interconnection topology* and (mainly) *policy rules*.
- Example: BGP.
Routing protocols

Intra-domain Routing Protocols
Intra-domain routing

Distance-vector routing (DVR)
- What routing information?
  Costs of the paths to all reachable destinations.
- To what routers?
  Neighbor routers only.
- For what purpose?
  Determine for each destination the neighbor offering best path, based on distributed version of Bellman-Ford algorithm.
- Examples
  RIP (Routing Information Protocol), RFC 2453.

Link-state routing (LSR)
- What routing information?
  Costs of the links connected to the router.
- To what routers?
  All routers in the AS.
- For what purpose?
  Determine a complete network map (labeled graph), then run Dijkstra algorithm to find best path spanning tree.
- Examples
  OSPF (Open Shortest Path First), RFC 5340; IS-IS.
DVR: Basic algorithm (1)

Each router:

1. Maintains a table with all known routes (routing table)
   - Route = \{ network prefix, path metric, next hop, interface \}
   - Network prefix identifies a set of destinations with the same path.
   - No next hop for directly connected networks.

2. Advertises known routes to all neighbors (periodic/triggered)
   - List of \{ network prefix, path metric \} = Distance Vector (DV).

These DVR examples assume:

- Link metric (cost) = 1 \(\Rightarrow\)
- Path metric = link-count.
- Advertised DVs contain the path metric value in the routing table.

Note that the path metric used by the protocol RIP in the advertised DVs and the routing table is slightly different - see example later.

<table>
<thead>
<tr>
<th>Network</th>
<th>M</th>
<th>NH</th>
<th>IF</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1.0.0/16</td>
<td>1</td>
<td>d.c.</td>
<td>e</td>
</tr>
<tr>
<td>10.2.0.0/16</td>
<td>1</td>
<td>d.c.</td>
<td>w</td>
</tr>
<tr>
<td>10.3.0.0/16</td>
<td>2</td>
<td>C</td>
<td>w</td>
</tr>
<tr>
<td>10.4.0.0/16</td>
<td>2</td>
<td>A</td>
<td>e</td>
</tr>
<tr>
<td>10.5.0.0/16</td>
<td>2</td>
<td>A</td>
<td>e</td>
</tr>
<tr>
<td>10.6.0.0/16</td>
<td>3</td>
<td>A</td>
<td>e</td>
</tr>
</tbody>
</table>

**Distance vector (DV\(_B\))**

- \(M = \) route metric
- \(NH = \) next hop
- \(IF = \) interface
- \(d.c. = \) directly connected
DVR: Basic algorithm (2)

(3) Updates its own routing table based on DVs from neighbors

For each destination (network) not directly connected:
- Next hop = Neighbor that offers best path.
- Path metric = neighbor’s metric + shared network metric.
- Best path metric = Min \(_{all \text{ neighbors}}\) \{ Path metric \}.

<table>
<thead>
<tr>
<th>Network</th>
<th>M</th>
<th>NH</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1.0.0/16</td>
<td>1</td>
<td>d.c.</td>
</tr>
<tr>
<td>10.2.0.0/16</td>
<td>1</td>
<td>d.c.</td>
</tr>
<tr>
<td>10.3.0.0/16</td>
<td>2</td>
<td>C</td>
</tr>
<tr>
<td>10.4.0.0/16</td>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>10.5.0.0/16</td>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>10.6.0.0/16</td>
<td>3</td>
<td>A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network</th>
<th>C</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1.0.0/16</td>
<td>2</td>
<td>1+1=3</td>
</tr>
<tr>
<td>10.2.0.0/16</td>
<td>2</td>
<td>1+1=3</td>
</tr>
<tr>
<td>10.3.0.0/16</td>
<td>1</td>
<td>1+1=2</td>
</tr>
<tr>
<td>10.4.0.0/16</td>
<td>2</td>
<td>1+1=2</td>
</tr>
<tr>
<td>10.5.0.0/16</td>
<td>3</td>
<td>1+1=2</td>
</tr>
<tr>
<td>10.6.0.0/16</td>
<td>4</td>
<td>1+1=3</td>
</tr>
</tbody>
</table>

Distance vector (DV\(_B\))
Example: Network startup

- Initially, the routers know only routes to directly connected networks, learned from the IP configuration of their interfaces.
- The routers send DVs with known routes to their neighbors.
- Routing info propagates hop-by-hop until all the routers learn routes to all destinations.

E.g., assume a sequence of updates: E → A; A → B, D; B, D → C, A; etc.
Convergence time:
- Very slow for periodic updates (e.g., RIP sends updates every 30 sec).
- Much faster for triggered updates (on request or to advertise DV changes).

For the basic DVR algorithm

Convergence time:
- Very slow for periodic updates (e.g., RIP sends updates every 30 sec).
- Much faster for triggered updates (on request or to advertise DV changes).
Towards real-life DVR protocols

- **Incremental update of the routing table**
  - Routers incrementally update their routing table for each received DV, instead of storing the DVs for all neighbors.
  - For each entry in a received DV:
    - If the received path metric is **smaller** than own metric, update the route: metric = received metric, next hop = neighbor.
    - If the received path metric is **larger** than own metric, but comes from the **same neighbor**, update the route: metric = received metric.
    - Otherwise ignore the received DV entry.

- **How to handle infeasible routes?**
  - Different cases: link/interface, path, or router failures.
  - Route timeout: Delete a route if not advertised for a long time. A basic mechanism applicable in all cases, but rather slow. Example: Periodic updates every 30 sec. Declare a route invalid if not advertised for 180 sec. Delete the route after other 180 sec.
  - Notification of infeasible routes? (We'll discuss this later.)
Routing loops and “counting to infinity”

For the basic DVR algorithm

- The link E ↔10.6.0.0/16 fails.
- E believes that it can still reach 10.6.0.0/16 via A, changes its table and sends an update to A.
- What happens next? The path metric increases indefinitely.
- Routing loops. The routers cannot detect that a destination has become unreachable.
DVR enhancements (1)

- **Cause of troubles**
  - Routers ignore the actual topology. They rely on each other’s routing updates. Basic algorithm allows mutual deception!

- **What can we do?**
  - Simple enhancements can ensure convergence and avoid most routing loops. Complete elimination is difficult/inefficient.

- **Ensuring convergence: Maximum path metric**
  - The path metric is limited to a maximum value (“infinity”). Destinations with a larger value are considered unreachable.
  - What value? Conflicting requirements:
    - Small value for fast convergence. Larger than any path metric.
    - Example: RIP “infinity” = 16.
  - Inefficient solution for routing loops.
  - Useful for announcing unreachable destinations.
DVR enhancements (2)

- **Split horizon (simple)**
  - "Never advertise a route to the neighbor it was learned from."
  - More precisely: If RB is the next hop of RA to destination D, then RA does not advertise to RB a route to D.

- **Motivation**
  - It is not useful: the neighbor knows it already.
  - It is harmful: the neighbor believes it is a different path.

- **Effects**
  - Eliminates 2-router loops. Does not eliminate larger loops.
  - Speeds up convergence. Reduces routing update traffic.

- **Split horizon with poisoned reverse**
  - Advertise infeasible routes with infinite metric (rather than omitting them from the DV).
  - Increases the update size. In practice, it is used for limited time when a (previously feasible) route becomes infeasible. Needed to speed up convergence for the *incremental update* algorithm.
Example: State after convergence

<table>
<thead>
<tr>
<th>Router B</th>
<th>Via</th>
</tr>
</thead>
<tbody>
<tr>
<td>Route M NH</td>
<td>C A</td>
</tr>
<tr>
<td>10.1.0.0/16 1 d.c.</td>
<td>- -</td>
</tr>
<tr>
<td>10.2.0.0/16 1 d.c.</td>
<td>- -</td>
</tr>
<tr>
<td>10.3.0.0/16 2 C</td>
<td>2 3</td>
</tr>
<tr>
<td>10.4.0.0/16 2 A</td>
<td>3 2</td>
</tr>
<tr>
<td>10.5.0.0/16 2 A</td>
<td>- 2</td>
</tr>
<tr>
<td>10.6.0.0/16 3 A</td>
<td>- 3</td>
</tr>
</tbody>
</table>

| Router A | Via |
|----------|
| Route M NH | B D E |
| 10.1.0.0/16 1 d.c. | - - - |
| 10.2.0.0/16 2 B | 2 3 - |
| 10.3.0.0/16 2 D | 3 2 - |
| 10.4.0.0/16 1 d.c. | - - - |
| 10.5.0.0/16 1 d.c. | - - - |
| 10.6.0.0/16 2 E | - - 2 |

| Router C | Via |
|----------|
| Route M NH | B D |
| 10.1.0.0/16 2 B | 2 3 |
| 10.2.0.0/16 1 d.c. | - - |
| 10.3.0.0/16 1 d.c. | - - |
| 10.4.0.0/16 2 D | 3 2 |
| 10.5.0.0/16 3 B | 3 3 |
| 10.6.0.0/16 4 B | 4 4 |

| Router D | Via |
|----------|
| Route M NH | C A |
| 10.1.0.0/16 2 A | 3 2 |
| 10.2.0.0/16 2 C | 2 3 |
| 10.3.0.0/16 1 d.c. | - - |
| 10.4.0.0/16 1 d.c. | - - |
| 10.5.0.0/16 2 A | 4 2 |
| 10.6.0.0/16 3 A | 5 3 |

For DVR with split-horizon

Split-horizon:
- Eliminates trivial loops.
- Faster convergence.

Example:
- What happens if the link from E to 10.6.0.0/16 fails?
DVR enhancements (3)

- **Triggered updates**
  - "Send an update whenever the metric of a route changes, besides the periodic updates."
  - Much faster convergence (than for periodic updates), but we have to avoid "update storms".
  - Related enhancement: Allow a router to request an update (e.g., after booting up, to build up its routing table).

- **Hold-down**
  - When a route becomes infeasible, wait for a certain time (hold-down timer) before accepting worse routes to that destination. Wait until information about the incident propagates to all the routers.
  - The route is marked and *advertised as unreachable*. Even if it was not advertised earlier due to split horizon. The route may still be used to forward packets (no alternative).
  - Less routing anomalies during convergence. For triggered updates, less control traffic but longer convergence duration.
# RIP timers (Cisco RIP)

<table>
<thead>
<tr>
<th>Timer</th>
<th>Description</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update timer</td>
<td>Time interval between the transmission of periodic update messages.</td>
<td>30 sec.</td>
</tr>
<tr>
<td>Invalid timer</td>
<td>Time interval until a route is considered invalid because no updates confirming it have been received. When it expires, the route is marked and advertised as unreachable. Also, it is “hold down” for the duration of the hold-down timer.</td>
<td>180 sec.</td>
</tr>
<tr>
<td>Hold-down timer</td>
<td>Time interval during which a route is “hold down”. This occurs when a neighbor indicates a network as unreachable or the invalid timer expires. The route is advertised as unreachable.</td>
<td>180 sec.</td>
</tr>
<tr>
<td>Flush timer</td>
<td>Time interval until the route is removed from the routing table if not confirmed by updates.</td>
<td>240 sec.</td>
</tr>
<tr>
<td>Sleep timer</td>
<td>Optional delay before sending a triggered update</td>
<td></td>
</tr>
</tbody>
</table>
RIP packets

Encapsulation

UDP header
RIP Port = 520

RIP header

Up to 25 RIP route entries (DV)

RIP version 1 packet format

- Req/Resp (1)
- Version (1)
- must be zero (2)
- Address family id (2)
- must be zero (2)
- Route IPv4 address (4)
- must be zero (4)
- must be zero (4)
- Route metric (4)
- (other route entries)

RIP version 2 packet format

- Req/Resp (1)
- Version (1)
- Route Tag (2)
- Address family id (2)
- must be zero (2)
- Route IPv4 address (4)
- Route Subnet Mask (4)
- Next Hop (4)
- Route metric (4)
- (other route entries)

Main RIPv2 improvements:
- Full support for subnetting, including VLSM.
- Authentication.
- Packets are multicast not broadcast.
RIPv2 example (1/2)

The path metric used by RIP in the routing table and the advertised DVs is slightly different from the examples given earlier: Path metric = hop-count. Advertised DVs contain the metric value in the routing table incremented by 1.

224.0.0.9 = Multicast address with link scope: “All RIP-aware routers on this link”.

RIP updates received by RC

RIP: received v2 update from 172.16.2.5 on S0
172.16.1.0/24 via 0.0.0.0 in 1 hops
172.16.2.0/30 via 0.0.0.0 in 1 hops

RIP: received v2 update from 172.16.3.1 on E0
172.16.1.0/24 via 0.0.0.0 in 2 hops
172.16.2.0/30 via 0.0.0.0 in 1 hops

RIP: received v2 update from 172.17.1.2 on E1
172.18.1.0/24 via 0.0.0.0 in 2 hops
172.18.2.0/30 via 0.0.0.0 in 1 hops

RC’s routing table

R 172.16.1.0/24 [1 hops] via 172.16.2.5, S0
R 172.16.2.0/30 [1 hops] via 172.16.2.5, S0
C 172.16.2.4/30 is directly connected, S0
C 172.16.3.0/24 is directly connected, E0
C 172.17.1.0/24 is directly connected, E1
R 172.18.1.0/24 [2 hops] via 172.17.1.2, E1
R 172.18.2.0/30 [1 hops] via 172.17.1.2, E1

RIP updates sent by RC

RIP: sending v2 update to 224.0.0.9 via S0 (172.16.2.6)
172.16.3.0/24 via 0.0.0.0, metric 1
172.17.1.0/24 via 0.0.0.0, metric 1
172.18.1.0/24 via 0.0.0.0, metric 3
172.18.2.0/30 via 0.0.0.0, metric 2

RIP: sending v2 update to 224.0.0.9 via E0 (172.16.3.2)
172.16.1.0/24 via 0.0.0.0, metric 2
172.16.2.4/30 via 0.0.0.0, metric 1
172.17.1.0/24 via 0.0.0.0, metric 1
172.18.1.0/24 via 0.0.0.0, metric 3
172.18.2.0/30 via 0.0.0.0, metric 2

RIP: sending v2 update to 224.0.0.9 via E1 (172.17.1.1)
172.16.1.0/24 via 0.0.0.0, metric 2
172.16.2.0/30 via 0.0.0.0, metric 2
172.16.2.4/30 via 0.0.0.0, metric 1
RIPv2 example (2/2)

RIPv2 periodic update sent by RC to RB

Distance vector advertised by RC to RB
**OSPF overview**

- **OSPF: Open Shortest Path First**
  - IGP based on Link State Routing (LSR).

- **Hierarchical network structure**
  - The routing domain is divided into several areas interconnected by a backbone area ⇒ scalability, efficiency.
  - The topology of an area is hidden to the rest of the AS.
  - LSR within each area. Route summarization between areas. ⇒ Lower routing information traffic (flooding) and path computation overhead. Smaller routing tables.

- **Link State Routing enhancements**
  - Optimizations of the topological database and the routing info exchange (e.g., for multi-access networks with many routers).
Link State Routing (1/3)

- **Build the topological database**
  - Each router distributes the state of its own links to all routers in the domain (using reliable hop-by-hop flooding).
  - All routers collect the advertised link states and build identical copies of a database describing the topology of the domain.

![Physical network topology and link costs](image1)

![Graph model of the network topology determined by each router and stored in its local topological database](image2)

![Directed labeled graph (edges are labeled with the link metrics)](image3)
Link State Routing (2/3)

- Find the shortest paths
  - All routers independently run the same algorithm (Dijkstra) to find the shortest paths from the topological database.
  - Each router computes a tree of shortest paths, with itself as root, to all destinations.
  - The shortest paths are recomputed when the topology changes.
Derive the local routing table

- A router derives the entries in the routing table from the computed shortest paths.

Each router runs the same shortest path algorithm on the same graph (hopefully), but with a different root node - itself.

Hence each router computes a different set of paths: from itself to all destinations. However, the shortest path algorithm guarantees that the resulting routing tables are consistent.
OSPF areas (1/2)

- Two-level network hierarchy
  - The AS is structured into multiple areas interconnected by a backbone area (using physical or virtual links).
  - An area is a contiguous collection of networks, delimited by routers.
  - The backbone area also connects the AS to other ASes.
OSPF areas (2/2)

- **Types of routers**
  - Area internal routers: R1, R2, R5, R6, R8, R9, R12.
  - Area border routers (ABR): R3, R4, R7, R10, R11.
  - Backbone routers: All ABR + backbone internal routers (R5, R6).
  - AS border routers (ASBR): R5, R7. Provide links to other AS.

- **Routing information exchange**
  - **Intra-area**: Each router floods throughout an area the state of its own links that belong to the area.
  - **Inter-area**: A separate topological DB is maintained per area. The topology of an area is invisible outside the area.
  - The backbone redistributes routing info between areas.

- **Packet forwarding**
  - **Intra-area** traffic is independently handled by each area (based on its own routing info, links).
  - **Inter-area** traffic is forwarded via the backbone.
Inter-area routing info exchange

● **Outline of ABR operation**
  ● An ABR maintains a separate topological database for each area to which it belongs.
  ● Reports on the backbone topology summaries (routes) for its non-backbone areas.
  ● Injects into its non-backbone areas summaries about other areas learned on the backbone.
  ● An ABR also injects the external routes learned from AS border routers (ASBR).

● **Route distribution via the backbone**
  ● Redistribution of the routes via the backbone is similar to route advertisements in DVR.
    Similar way of computing route metrics.
  ● Redistribution can be limited for stub networks (replaced by default route).
OSPF topological database

- **Stub multi-access network**

- **Point-to-point network**

- **Multi-access network**
## Link state advertisements (LSA)

<table>
<thead>
<tr>
<th>LSA Type</th>
<th>Advertisement name</th>
<th>Advertisement description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Router (links) advertisement</td>
<td>Originated by all routers. Describes the collected states of the router's interfaces to an area. Flooded throughout a single area only.</td>
</tr>
<tr>
<td>2</td>
<td>Network (links) advertisement</td>
<td>Originated for multi-access networks by the Designated Router. Contains the list of routers connected to the network. Flooded throughout a single area only.</td>
</tr>
<tr>
<td>3, 4</td>
<td>Summary (link) advertisement</td>
<td>Originated by area border routers, and flooded throughout the advertisement's associated area. Describes a route to a destination inside the AS, but outside the area (inter-area route). - Type 3 describes routes to networks. - Type 4 describes routes to AS boundary routers.</td>
</tr>
<tr>
<td>5</td>
<td>AS external link advertisement</td>
<td>Originated by AS boundary routers, and flooded throughout the AS. Each AS external link advertisement describes a route to a destination in another AS. Default routes for the AS can also be described by AS external link advertisements.</td>
</tr>
</tbody>
</table>
- Networks in an area are assigned IP address blocks with certain prefixes. Routes to sets of blocks with a common prefix can be summarized ...
Topological DBs for areas 1 and 0

Notes:
R11 is connected to areas 2, 3, and 0 (virtual). For simplicity, point-to-point links are assumed unnumbered (but this is problematic for R5, R6).
Setup & maintenance of the topological DB

- **Objectives**
  - *Consistent routing tables* ⇒ maintain synchronized topological DBs in all routers, with fast update in case of topology changes.
  - *Low control traffic and processing overhead* ⇒ use efficient mechanisms for distributing the routing information (especially for multi-access networks).

- **Mechanisms for creating/maintaining the topological DB**
  - *Neighbor discovery*: Directly connected OSPF routers discover each other (Hello protocol).
  - *Adjacency setup*: Neighbor routers set up an efficient topology for routing information exchange (adjacency relations).
  - *DB synchronization*: During adjacency setup, routers perform an initial synchronization of their topological DBs.
  - *DB update*: Routers maintain/update their topological DB by reliable flooding of LSAs following adjacency relations.
Adjacency relations

- **Point-to-point links**
  - Routers connected by a point-to-point link are always adjacent.

- **Adjacencies on multi-access networks**
  - Routers elect using the Hello protocol a Designated Router (DR) and a Backup Designated Router (BDR).
  - Adjacencies are set up only between the DR/BDR and the other routers.

- **DR responsibilities**
  - Routers on a multi-access network synchronize their topological DB with the DR: DB exchange with the DR at adjacency setup (not any-to-any), followed by LSA updates afterwards.
  - DR floods a Network-LSA which lists the routers connected to the multi-access network. It also disseminates the LSAs from the rest of the AS to the routers on its multi-access network.
Scenario: New network N1 attached to R1:
- R1 sends LS Update with Router LSA announcing link to N1 (to R4 = DR, and to BDR).
- LS Update is flooded all over the area, hop-by-hop, reliably (LS Ack, retransmission).
- LSA sequence number allows routers to distinguish duplicates and discard them.
- Each router runs SPF algorithm and updates its routing table.
- Also, the area’s ABRs send in the other areas LS Update containing a Summary LSA announcing the route to N1.
## OSPF packets

<table>
<thead>
<tr>
<th>Type</th>
<th>Packet name</th>
<th>Protocol function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hello</td>
<td>Used to discover and maintain neighbor relationships between routers, and elect the DR/BDR on multi-access networks.</td>
</tr>
<tr>
<td>2</td>
<td>Database (DB) Description</td>
<td>Contains a summary of the topological DB contents (list of LSA headers). Used to synchronize the topological DBs during adjacency setup.</td>
</tr>
<tr>
<td>3</td>
<td>Link State (LS) Request</td>
<td>Request to download specified entries in the topological DB (identified by LSA header).</td>
</tr>
<tr>
<td>4</td>
<td>Link State (LS) Update</td>
<td>Contains updates for the topological database (list of LSAs).</td>
</tr>
<tr>
<td>5</td>
<td>Link State (LS) Ack</td>
<td>Acknowledges the receipt of a Link State Update (OSPF provides a reliable transfer of routing information updates).</td>
</tr>
</tbody>
</table>

- OSPF runs directly over IP (protocol id=89).
- OSPF packets are exchanged only between adjacent routers.
  Except for the packets used in the process of discovering neighbors and setting up adjacencies.
Adjacency setup (1)

Scenario: OSPF on router R1 has just started. Router R2 is currently DR. R1 will accept R2 as DR.

- **Init**
  - Hello (RouterID = R1; Neighbors = none; DR = none; ...)
  - Hello (RouterID = R2; Neighbors = R1, R3, ...; DR = R2; ...)

**Initialize TDB exchange**

- Database Description (Seq = x, I = 1, M = 1, MS = 1)
- Database Description (Seq = y, I = 1, M = 1, MS = 1)
- Database Description (Seq = y, I = 0, M = 1, MS = 0)

**Exchange TDB: Database description**

- Database Description (Seq = y+1, I = 0, M = 1, MS = 1)
- Database Description (Seq = y+1, I = 0, M = 1, MS = 0)
- ...
Adjacency setup (2)

Router ID: R1

Adjacency state

Exchange

172.16.1.1

Database Description (Seq = y+1, I = 0, M = 0, MS = 1)

Database Description (Seq = y+1, I = 0, M = 0, MS = 0)

Loading

Exchange TDB: LSA loading
(can be interleaved with DD exchange)

LS Request (IDs of requested LSAs)

LS Update (requested LSAs)

LS Ack (acked LSA headers)

LS Update (requested LSAs)

LS Request (IDs of requested LSAs)

Full

Full

TDBs are now synchronized. Adjacency setup is finished

Router ID: R2

Router ID: R3

Router ID: R3

172.16.1.0/24

172.16.1.1

172.16.1.3

172.16.1.2

172.16.1.1

Exchange

Each router checks the received LSA headers against its own TDB to determine which LSAs need to be loaded from its peer.

Loading

LS Request/Update packets are used to request/deliver the LSAs missing from a router’s TDB. LSA loading may be interleaved with the DD exchange.

Full

The 2 routers have the same “network maps”.

© Octavian Catrina
Example: Topology DB

Topology DB for Area 1 (no auto-summary)

Topology of Area 1.
Router LSAs (RA, RB, RC) and Network LSAs (RC for N5).

Inter-area routes.
Summary LSAs (RC for N6, N7, ... N11).
LSAs in OSPF Update packet sent by RC in Area 1:
- **LS Type:** Router-LSA
  - **LS Age:** 3600 seconds
  - **Options:** 0x22 (DC, E)
  - **Link-State Advertisement Type:** Router-LSA
  - **Link State ID:** 172.17.100.1
  - **Advertising Router:** 172.17.100.1 (172.17.100.1)
  - **LS Sequence Number:** 0x80000002
  - **LS Checksum:** 0x1c42
  - **Length:** 36
  - **Flags:** 0x01 (B)
  - **Number of Links:** 1
  - **Type:** Transit ID: 172.16.3.2
  - **Data:** 172.16.3.2
  - **Metric:** 10

- **LS Type:** Network-LSA
  - **LS Age:** 3600 seconds
  - **Options:** 0x22 (DC, E)
  - **Link-State Advertisement Type:** Network-LSA
  - **Link State ID:** 172.16.3.2
  - **Advertising Router:** 172.16.3.2 (172.16.3.2)
  - **LS Sequence Number:** 0x80000001
  - **LS Checksum:** 0xb955
  - **Length:** 32
  - **Netmask:** 255.255.255.0
  - **Attached Router:** 172.17.100.1
  - **Attached Router:** 172.16.100.2

- **LS Type:** Summary-LSA (IP network)
  - **LS Age:** 3600 seconds
  - **Options:** 0x22 (DC, E)
  - **Link-State Advertisement Type:** Summary-LSA (IP network)
  - **Link State ID:** 172.18.1.0
  - **Advertising Router:** 172.17.100.1 (172.17.100.1)
  - **LS Sequence Number:** 0x80000003
  - **LS Checksum:** 0x64a9
  - **Length:** 28
  - **Netmask:** 255.255.255.0
  - **Metric:** 84

LSAs advertised by RC in Area 1:
- RC's own Router LSA.
- Network-LSA for 172.16.3.0/24: RC sends this LSA because it is DR for the transit multi-access network 172.16.3.0/24.
- Summary-LSAs: RC is ABR, so it sends in Area 1 Summary-LSAs with routes to destinations (networks) in the other areas.
Example: Adjacency setup

Adjacency setup for routers RB (172.16.3.1) and RC (172.16.3.2)
Routing protocols

Inter-domain Routing (BGP)
Internet structure

Tier-1 ISPs. Very large, global networks

Tier-2 ISPs. National or regional ISPs.

Company networks, small ISPs, ...

Transit domains

Stub domains

Customer-Provider Relationship (transit) ⇒ Hierarchical structure

Peer Relationship (shared-cost) ⇒ Shortcuts

Examples of paths
Types of routing domains

- **Transit domain**
  - Allows other routing domains to use its network infrastructure to communicate with each-other (ISPs, typically).

- **Stub domain**
  - Does not allow the use of its network infrastructure for traffic between other routing domains.
    - A stub domain is connected to at least one transit domain.
    - Can also be connected to other stub domains.
  - Different traffic patterns: content rich; access rich.
Inter-domain relationships

- **Customer-provider relationship**
  - Provider offers transit for traffic between the customer and all prefixes it can reach or part of them. Customer pays provider.

- **Peering relationship**
  - Two ASes interconnect their networks such that to allow traffic between their customers and between their internal prefixes.
  - Typically established by ISPs of similar sizes, without payment. Peering provides shortcuts and reduces transit costs.
Physical ISP interconnection

- **Dedicated links**
  - Set up between 2 ISPs that agree to interconnect their networks ⇒ mesh topology for AS interconnection.

- **Internet exchange**
  - More efficient solution: rendezvous point for many ISPs ⇒ hub and spokes physical topology.
  - Run by an independent organization.
  - A typical Internet exchange offers:
    - Collocation for ISP routers running BGP.
    - A high speed switched Ethernet LAN (1-10 Gbps links) for the interconnection of the ISP routers.
Border Gateway Protocol (BGP)

- **BGP**: The inter-domain routing protocol of the Internet
  - "BGP is running on more than 100K routers (my estimate), making it one of world’s largest and most visible distributed systems". T. Griffin, 2002.

- **BGP evolution** ("BGP was not designed, it evolved")
    - 1989: BGP-1 (RFC 1105) replaces EGP.
    - Uses CIDR, the new IP addressing & routing scheme. Deployed all over the Internet during the following years.
    - So far, it scaled up quite well with the fast growth of the Internet.
  - 2006: Updated BGP-4 specification, RFC 4271.
    - Tens of other RFCs describe various BGP-4 extensions introduced in the mean time.
Outline of BGP operation

- **Path vector routing protocol**
  - Similar to distance vector protocols.

- **BGP sessions**
  - Routers establish a *BGP session* on a *TCP connection*, exchange the *current routes* in the BGP Routing Information Base (RIB), and then send *incremental updates* of RIB changes.

- **BGP routing information (UPDATE message)**
  - *Feasible routes*: Address prefixes *reachable via the sender’s AS* and an associated set of *path attributes*.
  - *Withdrawn (infeasible) routes*: Address prefixes of previously announced routes that are *not reachable anymore via the sender’s AS*.
  - An UPDATE message can indicate both feasible routes and infeasible routes.
BGP path attributes

- **Path attributes**
  - BGP selects a feasible "best" route to a destination based on information provided in a set of attributes associated to each path, including: loop detection, next hop, local preference, etc.

- **Well-known attributes**
  - Must be recognized by all compliant BGP implementations. Are propagated to other neighbors.
    - **Well-known mandatory**: Must be specified for all paths.
    - **Well-known discretionary**: May be omitted for some paths.

- **Optional attributes**
  - Not required and not expected to be recognized by all BGP implementations. If recognized they may be propagated to other BGP routers, according to their meaning.
    - **Optional transitive**: If not recognized, may be propagated.
    - **Optional non-transitive**: If not recognized, must be discarded.
## Main BGP path attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORIGIN</td>
<td>well-known mandatory</td>
<td>Origin of the path information: 0 = intra-AS (IGP); 1 = EGP; 2 = incomplete (other means). Used to assess the trust in the source of routing info.</td>
</tr>
<tr>
<td>AS_PATH</td>
<td>well-known mandatory</td>
<td>List of all the ASes on the path: AS_SEQUENCE, AS_SET (AS_SET is needed for route aggregation). Loop detection, in/out filtering, best path selection.</td>
</tr>
<tr>
<td>NEXT_HOP</td>
<td>well-known mandatory</td>
<td>Address of the router to be used as next hop on the path to the specified prefix.</td>
</tr>
<tr>
<td>MULTI_EXIT_DISC (MED)</td>
<td>optional non-transitive</td>
<td>Value used in best path selection to discriminate among multiple entry points of a neighboring AS.</td>
</tr>
<tr>
<td>LOCAL_PREF</td>
<td>well-known</td>
<td>Value of the degree of preference for this route. Only used inside an AS (iBGP session, not eBGP).</td>
</tr>
<tr>
<td>COMMUNITY (RFC 1997)</td>
<td>optional transitive</td>
<td>32-bit tag that can be associated to any group of prefixes in order to specify various policies (can be structured as 16-bit AS number and 16-bit value). Standard (e.g., NO_EXPORT) and custom policies. Often used to control inbound traffic.</td>
</tr>
</tbody>
</table>
# BGP messages

<table>
<thead>
<tr>
<th>BGP message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPEN</td>
<td>Used to establish a BGP session between two routers (includes mutual identification and optional capabilities negotiation). Each router sends an OPEN message, which is acknowledged (if accepted) by a KEEPALIVE message.</td>
</tr>
<tr>
<td>KEEPALIVE</td>
<td>Used to verify the liveness of the BGP session. A KEEPALIVE message is sent if no other message was transmitted during HoldTime/3 seconds. A BGP session is canceled if no message was received during HoldTime seconds.</td>
</tr>
<tr>
<td>UPDATE</td>
<td>Used to exchange routing information on a BGP session. Can carry a set of feasible routes with common attribute-set (prefixes and attribute-set), and/or a set of withdrawn routes (prefixes).</td>
</tr>
<tr>
<td>NOTIFICATION</td>
<td>Used to inform the other BGP router of an error. After sending or receiving this message a BGP session is closed.</td>
</tr>
<tr>
<td>ROUTE_REFRESH</td>
<td>Message added in 2000 to enable a graceful restart of a BGP session (e.g., when policies are reconfigured).</td>
</tr>
</tbody>
</table>
**BGP session overview**

Two routers establish a BGP session only if they are configured to do so. Similarly, they exchange only the routes that are allowed by configuration (i.e., according to the policies of their ASes).

![Diagram of BGP session establishment]

**TCP connection establishment (to port 179)**

**BGP session establishment**

- OPEN (Version = 4, My-AS = 100, Hold-Time = 180, BGP-Identifier = 10.1.3.1, Optional-Param.)

**KEEPALIVE**

- OPEN (Version = 4, My-AS = 200, Hold-Time = 180, BGP-Identifier = 10.1.3.2, Optional-Param.)

**KEEPALIVE**

**Exchange of current routes in RIB**

- UPDATE (Path-Attributes-Len = ..., Path-Attributes = ..., NLRI = 140.1.128.0/18, 160.2.0.0/16)

- UPDATE (Path-Attributes-Len = ..., Path-Attributes = ..., NLRI = 190.3.192.0/18, 170.4.0.0/16)

**Incremental updates (RIB changes) using UPDATE:**
- Announce new feasible routes.
- Withdraw infeasible routes.

**Periodic KEEPALIVE messages (e.g., at Hold-Time/3)**

**Session established**

NLRI = Network Layer Reachability Information = IP address prefix
Example: BGP session setup

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>209</td>
<td>474.350000</td>
<td>10.1.254.1</td>
<td>10.1.254.2</td>
<td>BGP</td>
<td>OPEN Message</td>
</tr>
<tr>
<td>210</td>
<td>474.475000</td>
<td>10.1.254.2</td>
<td>10.1.254.1</td>
<td>BGP</td>
<td>OPEN Message, KEEPALIVE</td>
</tr>
<tr>
<td>211</td>
<td>474.538000</td>
<td>10.1.254.1</td>
<td>10.1.254.2</td>
<td>BGP</td>
<td>KEEPALIVE Message</td>
</tr>
<tr>
<td>225</td>
<td>504.677000</td>
<td>10.1.254.1</td>
<td>10.1.254.2</td>
<td>BGP</td>
<td>KEEPALIVE Message</td>
</tr>
</tbody>
</table>

Transmission Control Protocol, Src Port: bgp (179), Dst Port: 38368 (38368), Seq: 1, Ack:

**Border Gateway Protocol**

**OPEN Message**
- Marker: 16 bytes
- Length: 45 bytes
- Type: OPEN Message (1)
- Version: 4
- My AS: 65000
- Hold time: 180
- BGP identifier: 10.1.254.2

Optional parameters length: 16 bytes
- Optional parameters
  - Capabilities Advertisement (8 bytes)
    - Parameter type: Capabilities (2)
    - Parameter length: 6 bytes
  - Multiprotocol extensions capability (6 bytes)
  - Capabilities Advertisement (4 bytes)
    - Parameter type: Capabilities (2)
    - Parameter length: 2 bytes
  - Route refresh capability (2 bytes)
  - Capabilities Advertisement (4 bytes)

**KEEPALIVE Message**
- Marker: 16 bytes
- Length: 19 bytes
- Type: KEEPALIVE Message (4)
Example: BGP Update (1)

UPDATE Message advertising a feasible route to 10.10.0.0/16

Attributes of the BGP route to the prefix specified below

Prefix of the BGP route: 10.10.0.0/16
Example: BGP Update (2)

A BGP UPDATE message can advertise feasible routes and also announce withdrawn routes (previously advertised routes which are no longer feasible).
BGP path vector routing (1)

The **AS_PATH attribute** lists the ASes on the path. It is used for loop detection, for policy-based routing, and as basic path metric (AS count).

The **NEXT_HOP attribute** indicates the IP address of the router used to reach the prefix.
BGP path vector routing (2)

We assume that AS 20 is the provider, and AS 10 and AS 30 are its customers. AS 20 provides transit for its customers, hence it announces the routes to them.

R2 and R3 exchange the current routes in their tables.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS_PATH</th>
<th>NEXT_HOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>140.1.0.0/16</td>
<td>internal</td>
<td>. .</td>
</tr>
<tr>
<td>150.1.0.0/16</td>
<td>20</td>
<td>11.1.1.2</td>
</tr>
</tbody>
</table>

Router R1: BGP routing table

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS_PATH</th>
<th>NEXT_HOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>140.1.0.0/16</td>
<td>internal</td>
<td>. .</td>
</tr>
<tr>
<td>150.1.0.0/16</td>
<td>20</td>
<td>11.1.1.2</td>
</tr>
</tbody>
</table>

Router R2: BGP routing table

AS 20

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS_PATH</th>
<th>NEXT_HOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>150.1.0.0/16</td>
<td>internal</td>
<td>. .</td>
</tr>
</tbody>
</table>

Router R3: BGP routing table

AS 30

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS_PATH</th>
<th>NEXT_HOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>160.1.0.0/16</td>
<td>internal</td>
<td>. .</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS_PATH</th>
<th>NEXT_HOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>160.1.0.0/16</td>
<td>internal</td>
<td>. .</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS_PATH</th>
<th>NEXT_HOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>160.1.0.0/16</td>
<td>20</td>
<td>11.1.2.1</td>
</tr>
<tr>
<td>140.1.0.0/16</td>
<td>20 10</td>
<td>11.1.2.1</td>
</tr>
</tbody>
</table>

Router R2: BGP routing table

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS_PATH</th>
<th>NEXT_HOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>150.1.0.0/16</td>
<td>internal</td>
<td>. .</td>
</tr>
</tbody>
</table>

Router R3: BGP routing table

<table>
<thead>
<tr>
<th>Prefix</th>
<th>AS_PATH</th>
<th>NEXT_HOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>150.1.0.0/16</td>
<td>20</td>
<td>11.1.2.1</td>
</tr>
</tbody>
</table>

R2 and R3 exchange the current routes in their tables.
BGP path vector routing (3)

Router R1: BGP routing table
Prefix | AS_PATH | NEXT_HOP
--- | --- | ---
140.1.0.0/16 | internal | . . .
150.1.0.0/16 | 20 | 11.1.1.2

Router R2: BGP routing table
Prefix | AS_PATH | NEXT_HOP
--- | --- | ---
150.1.0.0/16 | internal | . . .
140.1.0.0/16 | 10 | 11.1.1.1
160.1.0.0/16 | 30 | 11.1.2.2

Router R3: BGP routing table
Prefix | AS_PATH | NEXT_HOP
--- | --- | ---
160.1.0.0/16 | internal | . . .
150.1.0.0/16 | 20 | 11.1.2.1
140.1.0.0/16 | 20 10 | 11.1.2.1

UPDATE (Prefix = 160.1.0.0/16, AS_PATH = 20 30, NEXT_HOP = 11.1.1.2)

Suppose that the BGP session between AS 20 and AS 30 is lost (link failure, R3 restart, etc). Then, R2 removes from its table the routes learned in this session and sends an update to R1.

R2 sends an incremental update to R1.

UPDATE (Withdrawn-Routes = 160.1.0.0/16)

R2 sends an incremental update to R1.

R1’s table returns to previous state.
BGP policy-based routing

Route import policies:
For each BGP peer
- Import filter selects the received routes accepted from that peer.
- Path attribute manipulation influences best route selection by this BGP router.

Control of outbound traffic:
Imported route participates in the selection of the best path toward that prefix.

Route export policies:
For each BGP peer
- Export filter selects the routes announced to that peer.
- Path attribute manipulation influences the best route selection by that peer.

Control of inbound traffic:
Announcing a route enables the other AS to route traffic to that prefix via this AS.
Best route selection

- **Overview**
  - The algorithm operates on the set of routes in Adj-RIBs-In after applying import policies (filter, tweak attributes, assign local preference). It selects one best route for each destination.

- **Sequence of selection steps for each destination**
  - Discard all the routes for which NEXT_HOP is inaccessible.
  - Discard all the routes for which AS_PATH contains the local AS or a loop.
  - Let \( n \) be the number of routes left.
  - If \( n > 1 \), select the routes with the largest LOCAL_PREF.
  - If \( n > 1 \), select the routes with the smallest number of ASes in AS_PATH.
  - If \( n > 1 \), select the routes with the lowest ORIGIN (IGP<EGP<INCOMPLETE).
  - If \( n > 1 \) and routes from the same AS, select the routes with the lowest MED.
  - If \( n > 1 \) and a route learned from eBGP, discard routes learned from iBGP.
  - If \( n > 1 \), select the routes with shortest path (local metric) to NEXT_HOP.
  - If \( n > 1 \), select the route from the BGP speaker with the lowest BGP Id.
Example: Route filtering

Note: Assume that AS2, AS3, AS4, AS5 have many other interconnections, not shown in this picture. In particular, AS2 and AS4 are also connected to other Tier-1 and Tier-2 ISPs, and have many other customers.
Example

AS 20 has two border routers, R2 and R3.
How can R2 advertise to R3 the routes learned from R1?
How can R3 advertise to R2 the routes learned from R4?

AS 20 runs an IGP for interior routing.
Could this IGP distribute the BGP routes as well?
No. The IGP cannot carry the BGP path attributes and does not scale up to BGP routing table size.

Solution: Propagate BGP routes within an AS using BGP.
EBGP vs. IBGP

- EBGP session: BGP routers in different ASs
- IBGP session: BGP routers in the same AS

**EBGP session**
- Directly connected routers (usually).
- Do not use LOCAL_PREF.
- Update AS_PATH, NEXT_HOP attributes in advertised paths.
- Advertise the best route to each destination. Usually, an export filter for each session.

**IBGP session**
- Need not directly connect the routers.
- Use LOCAL_PREF attribute.
- No change to AS_PATH and NEXT_HOP attributes.
- Advertise only best routes learned from EBGP (not from IBGP, to avoid loops). Usually, no export filter.  
  ⇒ Full mesh of iBGP sessions.
Example: EBGP + IBGP + IGP

Note how the path attributes change during route propagation.
Scaling up iBGP

How to scale up iBGP to a large domain?

- Routes learned from an iBGP session cannot be advertised on iBGP sessions, to avoid routing loops.
- Hence we must set up a full mesh of iBGP speakers.
- For N routers, each router must establish N-1 sessions, for a total of N(N-1)/2 sessions. Does not scale up. ASes can have 100s of routers or more.

Solution 1: Confederations

- Divide a large AS into smaller sub-domains.
- Use an iBGP full mesh inside each sub-domain, and eBGP between sub-domains.

Solution 2: Route reflectors

- Establish one or more routers, called route reflectors, that forward the iBGP routes to a cluster of other routers.
Route reflectors

- From iBGP full mesh to iBGP with route reflector

**Interactions between a router reflector and its clients**

- R2 and R3 are configured as clients of the route reflector R1. Clients establish iBGP sessions only with their route reflector.
- When a route reflector receives a route update from its clients or from eBGP, it computes the best route and then advertises it to its clients.