# Dynamic Routing Protocols



## THE FOUR QUESTIONS

Every dynamic routing protocol answers the same four questions. Learn the four answers for a protocol and you know the protocol — the rest is syntax.

| | Question | What it decides |
|---|---|---|
| 1 | **What messages does it use?** | how neighbours are found, how updates travel, how loss is handled |
| 2 | **What does it store, and where?** | which show command answers which question |
| 3 | **How does it measure a path?** | which path wins, and whether a second one can share the load |
| 4 | **What algorithm runs on that?** | how loops are avoided and how fast it reconverges |

Those four answers are what let you **calculate a metric** by hand, **find the loop-free alternate** before you need it, **predict how traffic will load balance**, and **say how long convergence will take** — instead of waiting to find out.

## THE THREE FAMILIES

**Distance vector — RIP, IGRP, and EIGRP by inheritance**

A router knows only what its neighbours tell it: a destination and a distance. It never sees the topology. **Routing by rumour** — which is why the whole family needs explicit loop prevention bolted on:

- **Split horizon** — never advertise a route back out the interface you learned it on
- **Route poisoning** — advertise a dead route at infinity rather than withdrawing it quietly
- **Poison reverse** — send the poison straight back to the source, overriding split horizon
- **Hold-down timer** — refuse worse information about a route that just went down
- **Triggered updates** — send the bad news immediately rather than waiting for the next cycle

**Link state — OSPF, IS-IS**

Every router floods a description of its own links. Each one assembles an **identical database** of the whole area, then runs the algorithm itself. No rumour, so no hold-downs — the loop prevention is the shared map.

**Path vector — BGP**

Each route carries the **full list of autonomous systems** it crossed. A router that sees its own AS number in that list drops the route. Loop prevention is the path itself, and it works between organisations that share no trust and no map.

**Where EIGRP sits.** It is a distance vector protocol — it learns distances from neighbours and has no map. What makes it *advanced* is DUAL: a pre-computed, mathematically proven loop-free backup, so it does not need hold-down timers to stay safe. Distance vector safety, close to link-state speed.

## ADMINISTRATIVE DISTANCE

Two protocols offering the same prefix do not compare metrics — the metrics are not in the same units. The router compares **trustworthiness** first, and only the winner's route is installed.

| Source | AD | Source | AD |
|---|---|---|---|
| **Connected** | **0** | **IS-IS** | 115 |
| **Static** | 1 | **RIP** | 120 |
| **EIGRP summary** | 5 | **ODR** | 160 |
| **eBGP** | 20 | **EIGRP external** | 170 |
| **EIGRP internal** | 90 | **iBGP** | 200 |
| **OSPF** | 110 | **Unusable** | **255** |

**The two that surprise people.** **eBGP at 20** beats every IGP — a route from outside your control is trusted over your own OSPF. And **iBGP at 200** loses to all of them. The asymmetry is deliberate, and it is why redistribution between BGP and an IGP needs filtering rather than optimism.

Lower wins. A floating static route is nothing more than a static route given a distance higher than the protocol it is backing up.

## ALL FIVE AT A GLANCE

| | RIPv2 | EIGRP | OSPFv2 | IS-IS | BGP |
|---|---|---|---|---|---|
| **Family** | Distance vector | Advanced distance vector | Link state | Link state | Path vector |
| **Scope** | **IGP — inside one autonomous system** | | | | **EGP** |
| **Algorithm** | Bellman-Ford | DUAL | Dijkstra SPF | Dijkstra SPF | Path vector |
| **Metric** | hop count | composite BW + delay | cost $10^8$÷BW | cost (default 10) | path attributes |
| **Admin distance** | **120** | **90** int 170 ext | **110** | **115** | **20** eBGP 200 iBGP |
| **Transport** | UDP 520 | IP proto 88 | IP proto 89 | Layer 2 (CLNS) | TCP 179 |
| **Multicast** | `224.0.0.9` | `224.0.0.10` | `224.0.0.5` `224.0.0.6` | L2 group MACs | unicast only |
| **Hello / hold** | no hellos 30 s updates | 5 / 15 s 60 / 180 slow | 10 / 40 s 30 / 120 NBMA | 10 / 30 s | 60 / 180 s |
| **Sends** | whole table, every 30 s | **only what changed, only when it changes** | | | |
| **Max useful hops** | **15** | no protocol limit | | | |
| **Unequal-cost load balancing** | no | **yes** `variance` | no | no | by policy |
| **On CCNA 200-301** | no | no | **yes** | no | no |

RIPv1 and IGRP are classful and cannot carry a mask. Everything in this table is classless and supports VLSM and CIDR.

## 1 — THE MESSAGES

**OSPF — five packet types, all inside IP protocol 89**

| Type | Packet | Does what |
|---|---|---|
| 1 | **Hello** | finds neighbours, agrees parameters, elects DR/BDR, keeps the adjacency alive |
| 2 | **Database Description** | a headline list of the LSAs I hold, so we can compare without sending everything |
| 3 | **Link State Request** | send me the ones I am missing or hold an older copy of |
| 4 | **Link State Update** | the LSAs themselves — the only packet carrying topology |
| 5 | **Link State Ack** | confirms an LSU. OSPF acknowledges its own flooding |

**EIGRP — IP protocol 88, carried by RTP**

| Op | Packet | Does what |
|---|---|---|
| 5 | **Hello** | finds and holds neighbours. **Unreliable** — never acknowledged |
| 1 | **Update** | routing information. Reliable, so it must be acknowledged |
| 3 | **Query** | I lost a route and have no feasible successor — does anyone have one? |
| 4 | **Reply** | the answer to a query, sent even when the answer is no |
| — | **Ack** | a Hello carrying an acknowledgement number and no data |

Queries are the expensive ones. A router with no feasible successor goes **Active** and must wait for every reply; if one never comes it goes **Stuck-in-Active** and tears the neighbour down. Summarisation exists to stop queries travelling.

**BGP — over an established TCP session, port 179**

| Type | Message | Does what |
|---|---|---|
| 1 | **Open** | version, my AS, my router ID, hold time, capabilities. Any mismatch and the session never forms |
| 2 | **Update** | prefixes advertised with their path attributes, and prefixes withdrawn |
| 4 | **Keepalive** | every 60 s, because TCP alone will not tell you fast enough |
| 3 | **Notification** | something is wrong — and it always closes the session |

RIP has two: **Request** and **Response**. A response is the entire routing table, every 30 seconds, whether anything changed or not.

## 2 — THE DATA STRUCTURES

All three keep the **same three tables**. Only the names change. Read them left to right and you are reading the protocol's decision, one stage at a time.

| who am I talking to | everything I have heard | what I actually use |
|---|---|---|

| | Neighbours | Everything learned | Chosen |
|---|---|---|---|
| **OSPF** | Adjacency table `show ip ospf neighbor` | Link State Database `show ip ospf database` | Global routing table (RIB) `show ip route` One table, shared by every source. This is the only one the forwarding plane ever consults. |
| **EIGRP** | Neighbor table `show ip eigrp neighbors` | Topology table `show ip eigrp topology` | |
| **BGP** | Neighbor table `show ip bgp summary` | BGP table `show ip bgp` | |

Whatever is running, start with `show ip protocols` — every protocol, its timers, networks, neighbours and distances, on one screen. It is the first command to type and the last one people learn.

**Which table to open.** Not forming an adjacency → **neighbour table**. Adjacency up but the prefix is missing → **topology or database**. Prefix is there but traffic takes the wrong path → **routing table**, and then administrative distance. Working the three in order finds nearly everything.

### What sits in the middle differs

- **OSPF's database is a map**, identical on every router in the area. Yours differing from a neighbour's is itself the fault.
- **EIGRP's topology table is a list of offers** — every path a neighbour reported, with its metric, usable or not.
- **BGP's table is a policy workspace.** One prefix commonly has several paths; exactly one is marked best and only that one is offered to the RIB.

## ROUTE SELECTION — TWO QUESTIONS

Only two decisions are ever being made, and confusing them is the source of most arguments about routing tables:

| | |
|---|---|
| 1 | Does the route go into the routing table at all? |
| 2 | Once it is in the table, is it the one I actually use? |

Question 1 is about **installing**; question 2 is about **forwarding**, answered per packet, by destination address.

### Same prefix, different protocols — administrative distance decides

```
O  10.10.10.0/24 [110/65]   via 1.1.1.1
D  10.10.10.0/24 [90/3072] via 2.2.2.2
```

The **D** route is installed. 90 is lower than 110, so EIGRP is trusted over OSPF, and the OSPF route does not appear at all. **This only applies when the two routes are identical** — administrative distance is compared only between routes with the same address *and* the same mask. Different mask, different route, and no comparison happens.

### Different prefixes — no contest, both are installed

```
O  10.10.10.0/25 [110/65]   via 1.1.1.1
D  10.10.10.0/24 [90/3072] via 2.2.2.2
```

Both go in. They are distinct destinations, so AD never gets involved. Which one carries a given packet is question 2 — and the answer changes with the address:

- `10.10.10.1` → the **O** route. /25 is the longer match
- `10.10.10.190` → the **D** route. The /25 covers only .0–.127, so it does not match at all

### Same prefix, same protocol — lowest metric

```
D  10.10.10.0/24 [90/3072]  via 1.1.1.1
D  10.10.10.0/24 [90/81774] via 2.2.2.2
```

Same source, so AD is a tie and the metric breaks it. 3072 wins.

### Same prefix, same protocol, same metric — both

```
D  10.10.10.0/25 [90/3072] via 1.1.1.1
D  10.10.10.0/25 [90/3072] via 2.2.2.2
```

Both installed, traffic shared across them, up to `maximum-paths` — four by default.

## LONGEST MATCH ROUTING

Once routes are installed, **the protocol that put them there stops mattering.** Forwarding compares the destination address against every entry and takes the one with the longest prefix. Administrative distance and metric have already done their work.

```
O   10.10.10.0/25 [110/78]   via 1.1.1.1
D   10.10.10.0/24 [90/3333] via 2.2.2.2
R   10.10.10.0/26 [120/2]    via 3.3.3.3
S*  0.0.0.0/0     [1/0]      via 4.4.4.4
```

Four different protocols with four different distances, and all four routes are installed — because all four prefixes are different.

| Destination | Next hop | Why |
|---|---|---|
| 10.10.10.1 | **3.3.3.3** | /26 covers .0–.63 and is the longest match |
| 10.10.10.66 | **1.1.1.1** | past the /26; the /25 covers .0–.127 |
| 10.10.10.190 | **2.2.2.2** | past both; only the /24 still matches |
| 10.10.11.1 | **4.4.4.4** | nothing matches — the default route catches it |

**Read that table again.** The RIP route wins for the first address, with the *worst* administrative distance on the page. Nothing is wrong: RIP won the install for the /26 because no other protocol offered a /26, and forwarding never reconsiders the source. This is why "my better protocol is being ignored" is almost always a mask question, not a distance question.

### The codes in the first column

| | |
|---|---|
| **C / L** | connected / local (the interface's own /32) |
| **S / S*** | static / static default candidate |
| **D / D EX** | EIGRP internal / external |
| **O / O IA** | OSPF intra-area / inter-area |
| **O E1 / E2** | OSPF external — E1 adds the internal cost, E2 does not |
| **i L1 / L2** | IS-IS level 1 / level 2 |
| **R / B** | RIP / BGP |

## CONFIGURATION SIDE BY SIDE

```
R1(config)#router rip
R1(config-router)#version 2
R1(config-router)#no auto-summary
R1(config-router)#network 10.0.0.0
```
Classful network only — RIP takes no wildcard

```
R1(config)#router ospf 100
```
Process ID is **local**. It need not match the neighbour
```
R1(config-router)#router-id 1.1.1.1
R1(config-router)#network 10.1.1.0 0.0.0.255 area 0
```
Wildcard mask, and every network statement names an area

```
R1(config)#router eigrp 100
```
Autonomous system number — this one **must match** the neighbour
```
R1(config-router)#no auto-summary
R1(config-router)#network 10.1.1.0 0.0.0.255
```
Wildcard, and no area — EIGRP has none

```
R1(config)#router bgp 65000
```
One BGP process per router, and the AS number is who you are
```
R1(config-router)#neighbor 203.0.113.2 remote-as 65001
```
Peers are configured one at a time, by address. Nothing is discovered
```
R1(config-router)#network 198.51.100.0 mask 255.255.255.0
```
Here `network` means *advertise this prefix*, and it must already be in the routing table — a different meaning from every line above

**Three meanings of one word.** In RIP, OSPF and EIGRP, `network` means *run the protocol on interfaces matching this*. In BGP it means *originate this prefix*. Same keyword, different job — and the single most common reason a first BGP configuration advertises nothing.

# Dynamic Routing Protocols



## 3 — THE METRICS

### RIP — hop count

Every router is one hop. A gigabit path four routers long loses to a serial line three routers long. **15** is the maximum; **16 means unreachable**, and that ceiling is also its loop prevention of last resort.

### OSPF — cost

$$\texttt{cost} = \frac{10^8 \text{ (reference bandwidth)}}{\text{interface bandwidth in bits/s}}$$

| | |
|---|---|
| T1 | 64 |
| 10 Mb | 10 |
| 100 Mb | 1 |
| 1 Gb+ | 1 |

Costs are summed along the path and always rounded down, never below 1. Which is the problem: past 100 Mb every link costs 1 and OSPF stops being able to tell them apart. Raise the reference bandwidth with `auto-cost reference-bandwidth` — on **every router**, or the comparison is meaningless.

### EIGRP — composite

```
metric = 256 × [ K1·BW + (K2·BW)/(256−Load) + K3·Delay ] ×
                    K5/(Reliability+K4)
```

| K value | Component | Default |
|---|---|---|
| K1 | Bandwidth | **1** |
| K2 | Load | 0 |
| K3 | Delay | **1** |
| K4, K5 | Reliability | 0 |
| — | MTU | **never used** |

With the defaults K2, K4 and K5 are zero, the whole right-hand term drops out, and what is left is:

$$\texttt{metric} = 256 \times (\, 10^7/\text{slowest BW in kbps} + \text{total delay in tens of } \mu s \,)$$

**Slowest bandwidth on the path, total delay along it.** That is the whole metric in practice. Load and reliability are left off deliberately — they change, and a metric that changes makes a network that never settles.

**MTU is carried but never calculated with.** It is a tie-breaker only. Being able to say that is a reliable way to sort people who have read about EIGRP from people who have run it.

## BGP HAS NO METRIC

BGP compares **path attributes**, in a fixed order, stopping at the first one that decides. There is no single number, because BGP is not choosing the fastest path — it is choosing the one policy says to use.

| | Attribute | Prefer |
|---|---|---|
| 0 | Next hop reachable | or the path is not considered at all |
| 1 | **Weight** (Cisco only) | highest — local to one router, never advertised |
| 2 | **Local preference** | highest — how your AS picks its way out. Default 100 |
| 3 | Locally originated | your own route beats a learned one |
| 4 | **AS path length** | shortest — the closest thing BGP has to a metric |
| 5 | Origin | IGP < EGP < Incomplete |
| 6 | **MED** | lowest — how a neighbour AS asks you to come in |
| 7 | eBGP over iBGP | external first |
| 8 | IGP metric to next hop | lowest — the nearest exit door |
| 9 | Oldest eBGP path | stability over novelty |
| 10 | Lowest router ID, then neighbour address | a deterministic coin toss |

**Which direction each one steers.** **Weight** and **local preference** control traffic *leaving* your AS and you can rely on them. **MED** and **AS path prepending** try to influence traffic *arriving*, and are only ever a request — the other AS is free to ignore both. Inbound traffic engineering is persuasion, not configuration.

## 4 — THE ALGORITHMS

### Bellman-Ford — RIP

Add your own cost to the distance a neighbour reported, keep the lowest, tell everyone else. Simple, and slow to notice a lie: without split horizon and hold-downs it will count to infinity.

### Dijkstra / SPF — OSPF, IS-IS

Build the shortest-path tree with yourself at the root, over a database every router in the area holds identically. A tree has no loops by construction, so no hold-downs are needed. The cost is CPU: any change inside the area makes everyone recompute, which is exactly what **areas** exist to contain.

### DUAL — EIGRP

| Term | Meaning |
|---|---|
| Feasible distance (FD) | my best total metric to the destination |
| Reported distance (RD) | the neighbour's own metric to it, as it told me |
| Feasibility condition | **RD < FD** |
| Successor | lowest total metric. Goes in the routing table |
| Feasible successor | meets the condition. Stays in the topology table, ready |
| Passive / Active | converged / querying for a path |

**Why the condition is that and not something else.** A neighbour whose own distance is already shorter than yours cannot be routing through you — if it were, its metric would include yours and be larger. So `RD < FD` proves the backup is loop free *without* looking at the topology. That single inequality is the whole trick, and it is why EIGRP fails over in under a second when a feasible successor exists.

No feasible successor and the route goes **Active**: query every neighbour, wait for every reply. That is the slow path, and summarisation is how you stop it spreading.

### Path vector — BGP

Carry the whole AS path with the route. See your own AS number in it and drop it. Loop prevention that needs no map, no shared metric and no trust — which is why it, and not an IGP, runs between organisations.

Between iBGP peers the AS path never grows, so it cannot detect a loop. The rule instead is that **a route learned from an iBGP peer is never passed to another iBGP peer** — and that rule is the entire reason full mesh, route reflectors and confederations exist.

## CONVERGENCE AND LOAD BALANCING

| | Reconverges by | Speed |
|---|---|---|
| RIP | waiting out invalid (180 s), hold-down and flush (240 s) timers | minutes |
| OSPF | flooding one LSA, then every router in the area reruns SPF | seconds |
| EIGRP | promoting the feasible successor already in the topology table | **sub-second** |
| EIGRP | *with no* feasible successor — go Active, query, wait | seconds+ |
| BGP | deliberately damped — hold time 180 s, advertisement intervals, policy | slowest |

BGP being slow is a design decision, not a defect. A protocol carrying the global table cannot afford to react to every flap it hears.

### Load balancing

| | Equal cost | Unequal cost |
|---|---|---|
| RIP | 4 by default | no |
| OSPF | 4 by default | no — not possible |
| EIGRP | 4 by default | **yes — `variance`** |
| BGP | one best path unless `maximum-paths` | by policy |

`maximum-paths` raises the equal-cost count on any of them. `variance 2` tells EIGRP to also use any feasible successor up to twice the successor's metric — and traffic is shared in **inverse proportion to metric**, not evenly.