- Static vs. Dynamic Routing -

Static vs. Dynamic Routing

There are two basic methods of building a routing table:

- **Static Routing**
- **Dynamic Routing**

A **static** routing table is created, maintained, and updated by a network administrator, *manually*. A static route to *every* network must be configured on *every* router for full connectivity. This provides a granular level of control over routing, but quickly becomes impractical on large networks.

Routers will *not* share static routes with each other, thus reducing CPU/RAM overhead and saving bandwidth. However, static routing is *not fault-tolerant*, as any change to the routing infrastructure (such as a link going down, or a new network added) requires manual intervention. Routers operating in a purely static environment cannot seamlessly choose a better route if a link becomes unavailable.

Static routes have an Administrative Distance (AD) of **1**, and thus are always preferred over dynamic routes, unless the default AD is changed. A static route with an adjusted AD is called a **floating static route**, and is covered in greater detail in another guide.

A **dynamic** routing table is created, maintained, and updated by a *routing protocol* running on the router. Examples of routing protocols include **RIP** (Routing Information Protocol), **EIGRP** (Enhanced Interior Gateway Routing Protocol), and **OSPF** (Open Shortest Path First). Specific dynamic routing protocols are covered in great detail in other guides.

Routers *do* share dynamic routing information with each other, which increases CPU, RAM, and bandwidth usage. However, routing protocols are capable of dynamically choosing a different (or better) path when there is a change to the routing infrastructure.

Do not confuse *routing* protocols with *routed* protocols:

- A ***routed*** protocol is a Layer 3 protocol that applies logical addresses to devices and routes data between networks (such as IP)
- A ***routing*** protocol dynamically builds the network, topology, and next hop information in routing tables (such as RIP, EIGRP, etc.)

Static vs. Dynamic Routing (continued)

The following briefly outlines the advantages and disadvantages of *static* routing:

Advantages of Static Routing

- Minimal CPU/Memory overhead
- No bandwidth overhead (updates are not shared between routers)
- Granular control on how traffic is routed

Disadvantages of Static Routing

- Infrastructure changes must be manually adjusted
- No “dynamic” fault tolerance if a link goes down
- Impractical on large network

The following briefly outlines the advantages and disadvantages of *dynamic* routing:

Advantages of Dynamic Routing

- Simpler to configure on larger networks
- Will dynamically choose a different (or better) route if a link goes down
- Ability to load balance between multiple links

Disadvantages of Dynamic Routing

- Updates are shared between routers, thus consuming bandwidth
- Routing protocols put additional load on router CPU/RAM
- The choice of the “best route” is in the hands of the routing protocol, and not the network administrator

Dynamic Routing Categories

There are two distinct categories of dynamic routing protocols:
- **Distance-vector protocols**
- **Link-state protocols**

Examples of distance-vector protocols include **RIP** and **IGRP**. Examples of link-state protocols include **OSPF** and **IS-IS**.

EIGRP exhibits both distance-vector and link-state characteristics, and is considered a *hybrid* protocol.

Distance-vector Routing Protocols

All **distance-vector** routing protocols share several key characteristics:
- **Periodic** updates of the **full** routing table are sent to routing neighbors.
- Distance-vector protocols suffer from slow convergence, and are highly susceptible to loops.
- Some form of *distance* is used to calculate a route's metric.
- The **Bellman-Ford algorithm** is used to determine the shortest path.

A distance-vector routing protocol begins by advertising directly-connected networks to its neighbors. These updates are sent *regularly* (RIP – every 30 seconds; IGRP – every 90 seconds).

Neighbors will add the routes from these updates to their own routing tables. Each neighbor trusts this information *completely*, and will forward their full routing table (connected *and* learned routes) to every other neighbor. Thus, routers fully (and blindly) rely on neighbors for route information, a concept known as **routing by rumor**.

There are several disadvantages to this behavior. Because routing information is propagated from neighbor to neighbor via periodic updates, distance-vector protocols suffer from slow convergence. This, in addition to blind faith of neighbor updates, results in distance-vector protocols being highly susceptible to routing loops.

Distance-vector protocols utilize some form of **distance** to calculate a route's metric. RIP uses **hopcount** as its distance metric, and IGRP uses a composite of **bandwidth** and **delay**.

Link-State Routing Protocols

Link-state routing protocols were developed to alleviate the convergence and loop issues of distance-vector protocols. Link-state protocols maintain three separate tables:

- **Neighbor table** – contains a list of all neighbors, and the interface each neighbor is connected off of. Neighbors are formed by sending **Hello** packets.
- **Topology table** – otherwise known as the "link-state" table, contains a map of all links within an **area**, including each link's status.
- **Shortest-Path table** – contains the *best* routes to each particular destination (otherwise known as the "routing" table")

Link-state protocols do *not* "route by rumor." Instead, routers send updates advertising the *state* of their *links* (a **link** is a directly-connected network). All routers know the state of all existing links within their **area**, and store this information in a **topology** table. All routers within an area have *identical* topology tables.

The best route to each link (network) is stored in the **routing** (or **shortest-path**) table. If the state of a link changes, such as a router interface failing, an advertisement containing ***only* this link-state change** will be sent to all routers within that area. Each router will adjust its topology table accordingly, and will calculate a new *best* route if required.

By maintaining a consistent topology table among all routers within an area, link-state protocols can **converge very quickly** and are **immune to routing loops.**

Additionally, because updates are sent only during a link-state change, and contain *only* the change (and not the full table), link-state protocols are **less bandwidth intensive** than distance-vector protocols. However, the three link-state tables **utilize more RAM and CPU** on the router itself.

Link-state protocols utilize some form of **cost**, usually based on bandwidth**,** to calculate a route's metric. The **Dijkstra formula** is used to determine the shortest path.