The Control Plane

Network-Layer Functions

- The **data plane** is reponsible for **forwarding packets** (moving them from the router's input, to the router's output).
- The **control plane** is responsible for **determining the route** taken by **packets** from **source to destination**.

Structuring the Control Plane

- There are two ways to **structure the network control plane**:
  1. **Per-router control (traditional)** — Each **router** has a **routing algorithm** that is used to determine **where to route the packet**.
  2. **Logically centralized control (software defined networking)** — Remote controller computes, and **installs a fordwarding table** in the **routers**.

Routing Protocols

Routing Protocols

- The **goal** of a **routing protocol** is to **determine "good" routes** from the **sending host** to the **receiving host** through a network of routers.
- In order to acheive that goal, **each router** needs to know **what it is directly conntected to**, and what **those routers are connected to**.
- A **path** is a sequence of **routers** that packets must **traverse** from the **inital sending host** to the **final destination host**.
- A **"good" route** is a **route** that is the **fastest, least congested, and of least "cost"**.

Routing Graphs

- A **routing graph** is a tuple $G = (N, E)$ where N is a set of routers $\{n_1, n_2, \cdots, n_j\}$ and E is a set of links $\{e_1, e_2, \cdots, e_k\}$.
- The **cost** of a **link** $l \in E$ is defined as a function $C : E \to \mathbb{R} \cup \{\infty\}$, denoted by $C_{a,b}$ where $a, b \in N$ are the routers that the link l is contected to.

Routing Algorithms

- A **routing algorithm** is an **algorithm** that is used to determine the **a "good" path** that a **packet** should take to get from **a sending host** to a **receiving host**.
- **Route classificatons**:
  1. **Static Routes** — Static routes are routes that do **do not change**, or that **change very slowly over time**.
  2. **Dynamic Routes** — Dynamic routers are routes that **change quickly over time**, or have a **quickly chaning cost**.
- **Routing algorithm classifications**:
  1. **Link State Algorithms (Global)** — Link state algorithms are used when **all routers** have a **complete topology of the network, and know the cost of each route**.
     - An example of link state algorithms is Dijkstra's link-state routing algorithm.

2. **Distance Vector Algorithms (Decentralized)** — Distance vector algorithms are used routers initially **only know the link cost to attached neighbors**. This algorithm is **iterative**, and information needs to be **exchanged with neighboring routers**.

Dijkstra's Link-State Routing Algorithm

- Notations:
  1. $C_{x,y}$ — The direct link cost from node x to node y. If x and y are not directly connected, $C_{x,y} = \infty$.
  2. $D(v)$ — The current least-cost-path cost estimate from source to destination v.
  3. $p(v)$ — Predecessor node along path from source to v.
  4. N' — The set of nodes whose least-cost-path is definitively known.
- The algorithm:

```
begin
    N' = {u}
    for all nodes v
        if v is adjacent to u
            D(v) = Cu,v
        else
            D(v) = ∞

    Loop
        find w net in N' such that D(w) is minimum
        add w to N'
        update D(v) for all v adjacent to w and not in N'
        D(v) = min( D(v), D(w) + Cw,v )
    Until all nodes are in N'
end
```

- The complexity of this algorithm is $O(n^2)$.
- There are more efficient implementations that are $O(n \log n)$.

The Distance Vector Algorithm

- The **Distance Vector Algorithm** is based on **Bellman-Ford's equation**: Let $D_x(y)$ denote the cost of the least-cost path from x to y. Then $D_x(y) = min\{c_{x,v} + D_v(y)\}$.
- From time to time, each node sends its own distance vector estimate to neighboring nodes. This distance vector can be used by the distance vector algorithm to allow nodes to compute the least-cost path.