Detailed Explanation of DSDV (Destination-Sequenced Distance Vector Protocol)

Introduction

The **Destination-Sequenced Distance Vector (DSDV)** protocol is a proactive routing protocol designed primarily for mobile ad-hoc networks (MANETs). As a **table-driven protocol**, it ensures that every node in the network maintains an up-to-date routing table. This table is used to immediately route data packets without the need to discover routes at the time of data transmission, making DSDV effective in scenarios where low-latency routing is required.

Working Mechanism

1. **Routing Table Management:**

Each node in a DSDV-based network maintains a **routing table**, which contains several key pieces of information for each destination:

- **Destination address**: The IP address or identifier of the destination node.
- **Next hop**: The address of the next node (hop) that data should be sent to on the way to the destination.
- **Number of hops**: The number of hops (intermediate nodes) between the current node and the destination node.
- **Sequence number**: A number used to determine the freshness of the route to the destination. A higher sequence number indicates a more recent route, helping to avoid routing loops.

This table is crucial because it allows the node to determine how to forward data packets efficiently and how to handle route updates from its neighbors.

2. **Periodic Updates:**

- **Full Dumps**: Nodes periodically broadcast their entire routing table to their neighbors. This method ensures that all nodes in the network have consistent routing information.
- **Incremental Updates**: Instead of sending the entire table, nodes send only the changes (updates) to their routing tables since the last broadcast. This method reduces the overhead associated with routing updates.

These periodic broadcasts are essential in ensuring that all nodes have up-to-date information about routes to all destinations, reducing the likelihood of delays in data transmission.

3. **Sequence Numbers:**

- Each node assigns a **sequence number** to its routes, incremented each time a change occurs in its routing table (such as when the destination node moves or the route becomes unavailable).
- The **sequence number** serves two critical purposes:
  - **Freshness of the route**: Nodes use the sequence number to determine which route is the most recent. A route with a higher sequence number is considered fresher than one with a lower sequence number.

- **Prevention of loops**: Sequence numbers help prevent loops in the network. If a node hears a route update with an outdated sequence number, it knows that the route is stale and will not use it.

Example:

- Let's consider a network with nodes A, B, and C.
- Node A wants to send data to Node C, and its routing table might contain:
  - **Destination**: C
  - **Next Hop**: B
  - **Hops**: 2
  - **Sequence Number**: 5 (indicating that this route to C is fresh).
- If Node C moves, it increments its sequence number (e.g., from 5 to 6) and broadcasts this update to its neighbors. Node A will then update its routing table to reflect the new route to C with the updated sequence number.

4. **Broadcasting and Propagation of Updates**:
   - When a node learns of a better route to a destination (i.e., one with a lower hop count or a higher sequence number), it will broadcast this update to its neighbors.
   - This update may be a **full dump** (sending the entire routing table) or an **incremental update** (sending only the changed entries), depending on the protocol implementation.

Advantages of DSDV

1. **Immediate Route Availability**:
   - Since DSDV is a **proactive protocol**, it maintains routing tables that are always up-to-date. This means that routes are available immediately for any node to use without needing to wait for a route discovery process.
   - This is particularly beneficial in scenarios where low latency is critical.
2. **Loop-Free Routing**:
   - The **sequence numbers** prevent routing loops by ensuring that only the freshest routes are used. When nodes compare sequence numbers, they can avoid using outdated or looped routes.
3. **Less Route Discovery Delay**:
   - Unlike reactive routing protocols (like AODV), which discover routes on-demand when a route request is initiated, DSDV ensures that routes are always known and ready to use. This minimizes delays in data transmission.

Disadvantages of DSDV

1. **High Control Overhead**:

- One of the significant drawbacks of DSDV is the **high control overhead** due to the frequent updates. Even if no data is being transmitted, nodes must continue broadcasting their entire or updated routing tables at periodic intervals.
- This overhead can become substantial, especially in networks with many nodes, as each node's periodic updates can result in network congestion.

2. **Inefficiency in Highly Dynamic Networks**:
   - In highly dynamic environments, where nodes are frequently moving or changing network topology, DSDV can become **inefficient**. Since nodes must periodically update their tables and broadcast these updates, the system may waste bandwidth and resources on maintaining and propagating routing tables, which can be rapidly outdated.
   - The overhead of maintaining up-to-date tables in such dynamic conditions may outweigh the benefits of immediate route availability.
3. **Scalability Issues**:
   - As the size of the network increases, the size of the routing tables also grows, leading to increased memory usage and potentially larger update messages.
   - The scalability of DSDV may be limited in very large MANETs, as maintaining and broadcasting large routing tables can be a resource-intensive process.

Example Scenario

Imagine a simple MANET with nodes **A, B, and C**:

1. Initially, Node A knows that to reach Node C, it needs to forward data through Node B. It sets up the following in its routing table:
   - **Destination**: C
   - **Next Hop**: B
   - **Hops**: 2
   - **Sequence Number**: 5
2. If Node C moves to a new position, it increments its sequence number (e.g., from 5 to 6) and broadcasts an update with the new sequence number and routing information.
3. Node A receives this updated route and adjusts its routing table to reflect the new information, ensuring that it uses the fresh route with the highest sequence number to reach Node C.
4. If Node A needs to send data to Node C, it can immediately forward the data to Node B, which will then route it to Node C without delay.

Conclusion

The **Destination-Sequenced Distance Vector (DSDV)** protocol is an effective **proactive routing protocol** for mobile ad-hoc networks that provides loop-free and immediately available routes using sequence numbers. However, its reliance on frequent periodic updates can lead to significant control

overhead, especially in highly dynamic or large-scale networks, which limits its efficiency in such environments. Despite these drawbacks, DSDV remains a useful protocol in scenarios where immediate route availability and low-latency communication are paramount.

---

2. Reactive Routing Protocols: DSR (Dynamic Source Routing) and AODV (Ad Hoc On-Demand Distance Vector)

In **reactive routing protocols**, routes are established only when needed. These protocols do not maintain a constant routing table like proactive protocols, which reduces the overhead of continuously updating routes. However, the downside is that there is a delay during route discovery. Reactive protocols are particularly useful in environments where the network topology changes infrequently, or where resources (such as bandwidth and processing power) are constrained.

The two prominent **reactive protocols** are **DSR (Dynamic Source Routing)** and **AODV (Ad Hoc On-Demand Distance Vector)**. Let's look at both in detail:

Dynamic Source Routing (DSR)

Route Discovery:

1. **Initiation**:
    - When a source node (S) needs to send data to a destination node (D) and doesn't have a known route to D, it initiates **route discovery**.
    - Node S broadcasts a **Route Request (RREQ)** message to its neighbors.
2. **Propagation**:
    - The **RREQ** message is forwarded by intermediate nodes, and along the way, each node that forwards the RREQ records the address of the node that forwarded the message (this is known as the **source route**).
    - Essentially, each intermediate node adds its own address to the RREQ to help build the full path back to the source.
3. **Route Reply**:
    - When the destination node (D) receives the **RREQ**, it sends a **Route Reply (RREP)** back to the source node (S) along the reverse path recorded in the RREQ.
    - The RREP also includes the complete route (i.e., all the intermediate nodes between the source and destination), and the nodes along the reverse path forward the RREP back to the source node.

Source Routing:

- One of the key features of **DSR** is **source routing**.
- In **source routing**, the entire path from the source to the destination is included in the packet header, meaning the source node dictates the entire route to be taken by the packet.
- This eliminates the need for intermediate nodes to maintain route information — they just forward the packet along the path provided in the header.

Example:

If node **A** wants to communicate with node **E**, and it doesn't already know the route, it would do the following:

- **Route Request (RREQ)**:
    - **A** broadcasts the RREQ message: RREQ: A → B → C → D → E.
    - Each intermediate node (B, C, D) adds its address to the path as it forwards the RREQ.
- **Route Reply (RREP)**:
    - When node **E** receives the RREQ, it replies with a **Route Reply (RREP)**: RREP: E → D → C → B → A.
    - This message travels back along the reverse path, and the full route (A → B → C → D → E) is cached by node A.

In the future, any packets from node **A** to node **E** will use this cached route (A → B → C → D → E) without needing to initiate a new route discovery.

Advantages of DSR:

- **Efficient in Small Networks**: DSR is efficient in small, low-traffic networks where the frequency of route discoveries is low, and the overhead of route maintenance is minimal.
- **Reduces Memory Overhead**: Intermediate nodes do not need to store routing information (apart from the current packet's path) since the source node dictates the full route in the packet header.

Disadvantages of DSR:

- **High Delay During Route Discovery**: The process of broadcasting RREQs and waiting for RREPs to return causes a delay before data can be transmitted.
- **Large Packet Headers**: The size of the packet header increases with the number of nodes in the route. For example, the route from A to E with 5 intermediate nodes would have a packet header containing 5 addresses, which increases the overhead.

Ad Hoc On-Demand Distance Vector Protocol (AODV)

Route Discovery:

- Like DSR, **AODV** also uses the **Route Request (RREQ)** and **Route Reply (RREP)** process to discover routes when required. However, there are key differences in how AODV handles routing information and maintains routes.

1. **Route Request**:
    - When a source node (S) needs to send data to a destination (D), it broadcasts an **RREQ** message.

- The intermediate nodes that receive the RREQ do not store the entire route (unlike DSR). Instead, they only store the **next hop** on the route.

2. **Route Reply**:
    - When the destination node (D) or an intermediate node with a valid route to the destination receives the RREQ, it sends a **RREP** message back to the source node (S), along the reverse path.
    - The RREP only includes the next hop for each intermediate node and is propagated along the reverse route to the source node.
3. **Route Maintenance**:
    - In **AODV**, nodes use **sequence numbers** to ensure that routes are fresh and up-to-date.
    - If a link along an established route breaks, the nodes involved send a **Route Error (RERR)** message to inform other nodes that the route is no longer valid.

Example:

If node **A** wants to send data to node **D**, the steps would be:

- **Route Request (RREQ)**:
    - A broadcasts the RREQ: RREQ: A → B → C → D.
    - Intermediate nodes store only the next hop (e.g., node B stores node C as the next hop).
- **Route Reply (RREP)**:
    - When node **D** receives the RREQ, it sends a **RREP** back to node **A**: RREP: D → C → B → A.
    - The RREP is forwarded along the reverse path, and each intermediate node updates its routing table to store the next hop for the destination.
    - Only the next hop is stored at each intermediate node, which reduces memory usage compared to DSR.

Advantages of AODV:

- **Reduces Memory Usage**: Since only the next hop is stored at intermediate nodes, AODV uses less memory compared to DSR, which stores the entire route at each node.
- **Adaptable to Topology Changes**: AODV can quickly adapt to changes in network topology because it continuously checks for fresh routes via sequence numbers.

Disadvantages of AODV:

- **Route Discovery Latency**: Similar to DSR, AODV experiences delays during route discovery due to the process of broadcasting RREQs and waiting for RREPs.
- **Frequent Broadcasting in Large Networks**: As the network size increases, the frequency of RREQ broadcasts may also increase, leading to higher network congestion.

Comparison Between DSR and AODV:

Feature	DSR	AODV
Route Discovery	Full path included in packet headers.	Only next hop stored at intermediate nodes.
Route Maintenance	No explicit route maintenance (cached routes).	Explicit route maintenance using RERR messages.
Route Discovery Delay	Higher due to complete route propagation.	Lower due to minimal route information propagation.
Memory Usage	Higher memory usage (entire path cached).	Lower memory usage (only next hop stored).
Scalability	Not as scalable due to large headers.	More scalable for larger networks.

In **AODV (Ad hoc On-Demand Distance Vector)**, the route discovery process is different from DSR because **AODV does not store the full path** in the packet header. Instead, **each node stores only the next hop** for the destination in a routing table. Here's the step-by-step explanation of how AODV establishes a route.

In DSR (Dynamic Source Routing), the process of route discovery and route storage is different compared to AODV. Let's break it down step by step to understand how nodes store the entire path and form the route.

Step-by-Step Process in DSR:

1. **Source Node Needs a Route:**
    - **Node A (source node) wants to send data to Node E (destination node), but it doesn't know the route to E.**
2. **Route Request (RREQ):**
    - **Node A broadcasts a Route Request (RREQ) to all its neighbors.**
    - **The RREQ packet contains:**
        - **Source address (Node A).**
        - **Destination address (Node E).**
        - **A unique identifier (to avoid processing old RREQs).**
        - **A list of nodes (starting with just Node A) that have forwarded this request.**

        - **The request packet is propagated by intermediate nodes.**

3. **Forwarding the RREQ:**
    - **Node B receives the RREQ from Node A. It then forwards the RREQ to its neighbors, adding Node B to the list of nodes in the packet.**
    - **The updated RREQ now contains the path: A → B.**
    - **Node C receives the RREQ from Node B and forwards it with the list updated to A → B → C.**
    - **This process continues as the RREQ is forwarded along the network, and each node adds itself to the list.**
        - **After Node D forwards it, the path in the RREQ would be A → B → C → D.**
        - **Finally, Node E receives the RREQ and sees that it is the destination.**
4. **Route Reply (RREP):**
    - **Node E receives the RREQ and responds with a Route Reply (RREP).**
    - **The RREP is sent back along the reverse path, using the list of nodes in the RREQ to form the reverse route.**
    - **So, Node E sends the RREP to Node D, which forwards it to Node C, and so on, until it reaches Node A.**
    - **The RREP includes the full path in the header (the same path list that was accumulated in the RREQ).**
5. **Route Establishment:**
    - **Now, Node A knows the entire path: A → B → C → D → E.**
    - **The nodes in the path (like B, C, D, E) also know the path because they added themselves to the list in the RREQ.**
    - **Node A can now send data packets to Node B, B forwards it to C, C to D, and finally D to E.**

Key Point in DSR:

- **In DSR, each node stores the whole route in the packet header.**
- **Intermediate nodes add themselves to the list of nodes in the RREQ.**
- **When the RREP is sent back, it carries the full path as a list of nodes.**
- **The entire route is included in the header of each packet sent from the source to the destination, which is called source routing.**
    - **For example, if the path is A → B → C → D → E, Node A will include this full list in the packet's header, and each node will use the list to forward the packet to the next hop.**

Summary:

- **In DSR, the full path is included in the packet's header. Each node adds itself to the list of nodes in the RREQ.**
- **The destination (E) responds with a RREP, and the path is carried back to the source.**
- **This allows Node A to know the entire route (A → B → C → D → E) and use it for packet forwarding.**

Step-by-Step Process in AODV:

1. **Source Node Needs a Route:**
    - **Node A** (source) wants to send data to **Node E** (destination) but doesn't know the route.
2. **Route Request (RREQ):**
    - **Node A** broadcasts a **Route Request (RREQ)** packet to its neighbors.
    - The **RREQ** contains:
        - Source address (Node A).
        - Destination address (Node E).
        - A sequence number (to ensure the freshness of the route and avoid loops).
        - A hop count (initially 0).
    - The hop count increments at each node to indicate how far the packet has traveled.
3. **Forwarding the RREQ:**
    - **Node B** receives the **RREQ** from **Node A**.
        - **Node B**:
            - Increases the hop count by 1.
            - Updates its **routing table** with:
                - Destination: **A**.
                - Next hop: **A**.
                - Hop count: 1.
            - Forwards the **RREQ** to its neighbors.
    - **Node C** receives the **RREQ** from **Node B**, increments the hop count, updates its routing table (Destination: A, Next hop: B, Hop count: 2), and forwards the **RREQ**.
    - This process continues until the **RREQ** reaches the destination **Node E**.
4. **Route Reply (RREP):**

- When **Node E** receives the **RREQ**, it prepares a **Route Reply (RREP)** packet.
- The **RREP** contains:
  - Destination address (Node E).
  - Source address (Node A).
  - Sequence number (ensuring the route is fresh).
  - Hop count (indicating how far **E** is from **A**).
- **Node E** sends the **RREP** back to the source **Node A**, using the reverse path stored in the routing tables of intermediate nodes.

5. **Forwarding the RREP:**
   - **Node D** receives the **RREP** from **Node E**.
     - Updates its routing table with:
       - Destination: **E**.
       - Next hop: **E**.
       - Hop count: 1.
     - Forwards the **RREP** to **Node C**.
   - **Node C** updates its routing table (Destination: E, Next hop: D, Hop count: 2) and forwards the **RREP** to **Node B**.
   - Finally, **Node B** forwards the **RREP** to **Node A**.
6. **Route Established:**
   - Now, **Node A** has a route to **Node E** in its routing table:
     - Destination: **E**.
     - Next hop: **B**.
     - Hop count: 4.
   - Each intermediate node (B, C, D) has the next-hop information for the route to **E**.

Key Point in AODV:

- **AODV does not store the full path** in the packet header like DSR.
- Instead, each node maintains a **routing table** with:
  - Destination address.
  - Next hop.
  - Hop count.

- Routes are created dynamically, and only the next hop is stored at each node, reducing memory overhead.

Example in AODV:

If **Node A** wants to send data to **Node E**:

- **Node A** starts with an RREQ → Broadcasts to neighbors.
- **Node E** responds with an RREP → Travels back along the reverse path.
- **Routing tables** at each node are updated during this process:
    - **Node A** knows: **E → via B → Hop count 4**.
    - **Node B** knows: **E → via C → Hop count 3**, and so on.

Comparison with DSR:

- In **DSR**, the **entire path** is stored in the packet header.
- In **AODV**, only the **next hop** is stored in each node's routing table. This makes AODV more efficient for larger networks.

Hybrid Routing Protocol: Zone Routing Protocol (ZRP)

Introduction

- ZRP combines **proactive** and **reactive** routing strategies to overcome their individual limitations:
    - **Proactive protocols** maintain up-to-date routes but have high overhead in dynamic networks.
    - **Reactive protocols** establish routes on demand, reducing overhead but causing delays during route discovery.
- ZRP aims to balance routing **overhead** and **latency** by dividing the network into **zones**. Each zone is defined by a **fixed radius** (measured in hops).

Working Mechanism

1. Intra-Zone Routing (Proactive Approach)

- **Proactive routing** is applied within each zone.
- Each node maintains a **routing table** for all nodes within its zone (nodes within the zone radius).
- Periodic updates ensure routes to nearby nodes are always available.

- This reduces latency for local communication since routes are pre-established.

2. Inter-Zone Routing (Reactive Approach)

- **Reactive routing** is used for communication between nodes in **different zones**.
- When a source node needs to communicate with a destination outside its zone:
    - The source node identifies a **border node** (a node at the edge of its zone).
    - The border node initiates a **route discovery** process (similar to reactive protocols like AODV or DSR).
- This strategy minimizes the overhead of maintaining routing information for distant nodes.

Example

Scenario:

- Consider a network with a **zone radius** of **2 hops**:
    - **Node A** wants to communicate with **Node F**, which is **4 hops away**.

Steps:

1. **Intra-Zone Routing:**
    - **Node A** checks its routing table and finds that **Node F** is outside its zone (since F is 4 hops away, and A's zone radius is 2 hops).
    - A identifies **Node C** as a **border node** (node at the edge of A's zone).
2. **Inter-Zone Routing:**
    - **Node C** initiates a **reactive route discovery** process to locate **Node F**.
    - A **Route Request (RREQ)** is sent, and **Node F** responds with a **Route Reply (RREP)**.
3. **Route Establishment:**
    - Once the route is discovered, **Node A** can send data to **Node F** via the path: **A → B → C → D → F**.

Advantages of ZRP

1. **Reduced Control Overhead:**
    - Within a zone, proactive routing minimizes delays since routes are pre-computed.
    - For distant nodes, reactive routing avoids the overhead of maintaining unnecessary routes.
2. **Low Latency for Local Communication:**
    - Communication within the zone benefits from pre-established routes, ensuring fast data delivery.

3. **Scalability:**
    - ZRP scales well in larger networks by confining proactive updates to local zones.

Disadvantages of ZRP

1. **Performance Depends on Zone Radius:**
    - A small radius increases inter-zone routing overhead (as many destinations lie outside the zone).
    - A large radius increases intra-zone overhead (as routing tables grow and require frequent updates).
2. **Complex Zone Maintenance:**
    - Dynamic networks with frequent topology changes make maintaining zones and updating routing tables challenging.

Taxonomy of Routing Protocols

Routing protocols in **Mobile Ad Hoc Networks (MANETs)** can be classified based on:

1. Routing Information

- **Proactive:** Maintain up-to-date routes to all destinations (e.g., DSDV).
- **Reactive:** Discover routes only when needed (e.g., DSR, AODV).
- **Hybrid:** Combine proactive and reactive approaches (e.g., ZRP).

2. Topology

- **Flat Routing:** All nodes have equal roles and responsibilities (e.g., AODV, DSR).
- **Hierarchical Routing:** Nodes are grouped into clusters with specific roles (e.g., Clusterhead Gateway Switch Routing, CGSR).

3. Route Metrics

- **Hop Count:** Number of hops to reach the destination.
- **Energy Efficiency:** Routes that conserve battery life.
- **Delay:** Routes with the least latency.

Conclusion

ZRP is a hybrid protocol designed to balance the strengths of proactive and reactive routing. It minimizes overhead for distant routes using reactive discovery and ensures low latency for local communication through proactive updates. Properly selecting the zone radius is critical to achieving optimal performance in dynamic networks.

4o