

"Router in a Stick" (Inter-Vlan Routing)

⇒ Router on a Stick is a method of Inter-Vlan Routing, which allows packets to be sent from one VLAN to another VLAN.

Inter-Vlan Routing Explained

⇒ VLANs are isolated by default, which means devices in different VLAN's cannot communicate with each other.

⇒ To enable communication b/w VLAN's, a layer 3 device (like a router or layer 3 switch) is required.

⇒ In the Router on a Stick setup:

- A single physical router interface is used to handle traffic from multiple VLAN's.

- Subinterfaces are created on the router, each representing a VLAN.

- These subinterfaces handle routing b/w the VLAN's.

Date 20
MTWTF

Date 20
MTWTF

→ How it works

1 A PC in VLAN 10 sends a packet to a PC in VLAN 20.

2 The packet is sent to the router because the destination is in a different subnet (VLAN).

3 The Router:

• Router, the packet from the VLAN 10 interfaces to the VLAN 20 subinterface.

• Sends the packet back to the switch through the trunk link.

4 The switch forwards the packet to the PC in VLAN 20.

Key features

→ This method enables communication b/w VLANs.

→ It is a cost-effective solution when a layer 3 switch is not available.

• Tricky this way b/c switch can't send b/w VLANs.

→ A switch operates at Layer 2 (Data Link Layer) it uses MAC-Address to forward frames within the same VLAN.

→ ~~Router~~ Since VLANs are logically isolated, a switch cannot forward packets between different VLAN's because it doesn't work with IP address (Layer 3).

→ Whereas a router (Layer 3, Network Layer) it uses IP Addresses to route packets b/w different networks or subset (in this case, VLANs).

→ When devices in different VLANs communicate, their traffic is routed routers subinterfaces.

Key concept:

• Switch alone: cannot forward packets b/w VLANs because it doesn't understand IP addresses or routing.

• Router can forward b/w VLANs because it works with IP Addresses and routes traffic b/w different subnets.

⇒ This is why the router on a bridge is necessary for enabling communication b/w VLAN's.

⇒ They are tagged packets coz they travel across switch ⇒ router with header (VLAN ID) of different VLAN

→ Untagged are sent within same VLAN
coz switch knows based on the port configuration (mac-address).

$$= 192.168.10.0/24 \Rightarrow \text{class C}$$

↓ 255.255.255.0

/24 indicates the subnet mask

coz total is 32 bit if it is 24 then class C, 16 → class B.

1. Why do we need the Router, not a Switch?

- **Switches** operate at **Layer 2 (Data Link Layer)**, meaning they forward frames based on **MAC addresses**. Switches can **isolate traffic** within **VLANs** but cannot **route traffic between VLANs** because they lack **Layer 3 (IP routing) functionality**.
- **Routers** operate at **Layer 3 (Network Layer)** and make decisions based on **IP addresses**. They are needed for **inter-VLAN communication** because devices in different **VLANs** belong to **different subnets**, and **routing** is required for **communication between subnets**.
- **Alternative:** A **Layer 3 Switch** can also perform **routing**, but for this scenario, we are using a **traditional Layer 2 switch**, which requires a **router** for **inter-VLAN communication**.

Key Difference:

- **Router:** Enables **routing between VLANs** (**inter-VLAN communication**).
- **Switch:** **Isolates traffic within VLANs** (**intra-VLAN communication**).

The main difference between tagged and untagged VLANs is how they handle traffic and how they're configured:

Tagged VLANs

Also known as "trunk" ports, these ports can pass traffic for multiple VLANs. They're best for larger organizations with complex networks that need strict security controls. Tagged VLANs require more detailed configuration, as each frame passing through the network needs to be specified with a VLAN tag.

Untagged VLANs

Also known as "access" ports, these ports can only accept traffic for a single VLAN. They're best for smaller organizations with less complex networks that don't require specialized security. Untagged VLANs are easier to configure because they don't require any modifications to the Ethernet frames. ↗

2. What is the Difference Between Untagged and Tagged Packets?

- **Tagged Packets:**

- Include a **VLAN ID** in the **802.1Q header**.
- Used on **trunk ports** to carry traffic for multiple VLANs.
- Allows the receiving device (**router or switch**) to identify the VLAN a packet belongs to.

- **Untagged Packets:**

- Do not include a **VLAN ID** in the frame.
- Used on **access ports**, where the **switch** already knows the **VLAN** based on port configuration.
- Also used for traffic belonging to the **native VLAN** on trunk ports.

Aspect	Tagged Traffic	Untagged Traffic
VLAN Information	Explicitly includes VLAN ID	No VLAN ID included
Used In	Trunk ports	Access ports or native VLAN
Purpose	Carry multiple VLANs	Handle single VLAN traffic

3. In the Header of the Packet, What Does It Include?

The 802.1Q header added for tagged packets contains the following key fields:

- 1. Tag Protocol Identifier (TPID):**

- A fixed value of **0x8100** that indicates the presence of an 802.1Q tag.

- 2. Priority Code Point (PCP):**

- Used for Quality of Service (QoS) to prioritize traffic.

- 3. Drop Eligible Indicator (DEI):**

- Indicates if the frame can be dropped during congestion.

- 4. VLAN ID:**

- A 12-bit field that identifies the VLAN (0–4095). VLAN 0 and 4095 are reserved, so valid IDs are 1–4094.

For untagged packets, this header is not present, and the switch determines the VLAN from the port configuration.

4. What Does the Command `encapsulation dot1q 10` Do?

- This command:
 1. Enables 802.1Q tagging on the router's sub-interface.
 2. Configures the sub-interface to handle traffic for VLAN 10.
 3. Instructs the router to expect packets with a VLAN tag of 10 on this sub-interface.

Example:

If a tagged packet from VLAN 10 arrives on the trunk link, the router:

- Recognizes the VLAN ID (10) from the 802.1Q tag.
- Routes the packet using the sub-interface `fa0/0.10`.

5. What is the Native VLAN? Why Is It Untagged and Sent on the Trunk Port?

- Native VLAN:
 - The VLAN that carries untagged traffic on a trunk port.
 - By default, the native VLAN is VLAN 1, but it can be changed for security or design purposes.
- Why Is It Untagged?
 - To maintain compatibility with older devices that do not support VLAN tagging.
 - Any untagged frames arriving on a trunk port are assumed to belong to the native VLAN.
- Native VLAN and Trunk Port:
 - Trunk ports send traffic for the native VLAN without a tag.
 - If there's a mismatch in the native VLAN configuration between two switches, it can cause VLAN leaks, where traffic is incorrectly forwarded.

Example Command to Change the Native VLAN:

```
bash                                         Copy code

interface fa0/1
switchport trunk native vlan 99
```

This sets VLAN 99 as the native VLAN for that trunk port.

Summary

1. Router vs. Switch:
 - A router enables inter-VLAN communication, while a Layer 2 switch only handles intra-VLAN communication.
2. Tagged vs. Untagged:
 - Tagged packets (trunk) include VLAN IDs for VLAN differentiation. Untagged packets (access/native VLAN) do not include VLAN IDs.
3. 802.1Q Header:
 - Contains VLAN ID, priority info, and \downarrow ocol identifier.



Yes, the `encapsulation dot1q 10` command is part of the **trunking protocol**, specifically the **802.1Q trunking protocol**.

Explanation:

- **802.1Q (dot1Q)** is the industry-standard protocol used for VLAN tagging on trunk links.
- When you use `encapsulation dot1q 10` on a router's sub-interface, you're telling the router to:
 1. Use the **802.1Q protocol** to process tagged packets on this sub-interface.
 2. Recognize packets tagged with **VLAN ID 10**.
 3. Handle these packets as traffic belonging to **VLAN 10**.

Why Is It Part of Trunking?

- The router is connected to the switch via a **trunk port** that carries traffic for multiple VLANs.
- The trunk port on the switch tags packets with their VLAN ID using the **802.1Q protocol**.
- The router's sub-interfaces are configured with `encapsulation dot1q` to handle the tagged packets for each VLAN.

Key Points About Trunk Protocol:

1. **802.1Q:** A standard for VLAN tagging on trunk links. It adds a 4-byte VLAN tag to Ethernet frames.
2. **Native VLAN:** Frames for the native VLAN are untagged on the trunk.
3. **Trunk Link:** Used to carry traffic for multiple VLANs between switches or between a switch and a router.

What Happens with `encapsulation dot1q 10`:

1. A packet tagged with VLAN ID 10 is received by the router on the trunk link.
2. The router's sub-interface (`fa0/0.10`) recognizes the VLAN ID **10** in the 802.1Q tag.
3. The router processes the packet based on its IP address and forwards it appropriately, potentially tagging it with a different VLAN ID if inter-VLAN routing is required.

So yes, `encapsulation dot1q 10` is directly related to the trunk protocol (802.1Q) and is used for tagging and identifying VLAN traffic on trunk links. Let me know if you'd like further clarification!

802.1Q it is the trunking protocol ? it is assigning the vlan Id of the sub interface ?



Yes, 802.1Q is the industry-standard trunking protocol used for VLAN tagging, and it is also responsible for assigning VLAN IDs to sub-interfaces in the "Router on a Stick" configuration. Let me clarify both aspects:

1. Is 802.1Q the Trunking Protocol?

- Yes, 802.1Q is a trunking protocol.
- It is the most commonly used protocol for carrying traffic for multiple VLANs on a single trunk link between devices (e.g., between a switch and a router or between switches).
- It adds a 4-byte VLAN tag to Ethernet frames to identify the VLAN a frame belongs to when it traverses a trunk link.

Why is it a Trunking Protocol?

- Trunking is the process of carrying traffic for multiple VLANs over a single physical link.
- 802.1Q tags are added to Ethernet frames on trunk links to differentiate VLAN traffic.
- Without 802.1Q tagging, a trunk port cannot determine which VLAN a frame belongs to.

2. Does encapsulation dot1Q Assign the VLAN ID to the Sub-Interface?

- Yes, the `encapsulation dot1Q` command assigns the VLAN ID to the sub-interface on the router.

How It Works:

1. Sub-Interface Creation:

- A router sub-interface is logically tied to a specific VLAN.
- For example, `Fa0/0.10` is tied to VLAN 10.

2. 802.1Q Encapsulation:

- The `encapsulation dot1Q 10` command:
 - Enables 802.1Q tagging for the sub-interface.
 - Specifies that the sub-interface will handle traffic tagged with VLAN ID 10.

3. Traffic Flow:

- When a tagged frame with VLAN ID 10 arrives at the router, it is processed by `Fa0/0.10`.
- When the router sends traffic to VLAN 10, it tags the frames with VLAN ID 10 before sending them back to the switch.

Key Concepts of `encapsulation dot1Q`

- It does not create the VLAN; the VLAN must already exist on the switch.
- It maps VLAN traffic to the router's sub-interface.
- It allows the router to:
 - Recognize VLAN-tagged traffic arriving from the trunk.
 - Route traffic between VLANs (inter-VLAN routing).

Example Workflow

1. A frame tagged with VLAN ID 10 arrives at the router from the switch.
2. The router checks its sub-interface configuration:
 - `Fa0/0.10` is configured with `encaps ↓ ion dot1Q 10`.
 - The router identifies that this frame belongs to VLAN 10 and processes it.

Summary

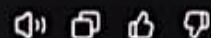
1. 802.1Q:

- It is the trunking protocol used to tag VLAN traffic.
- It enables multiple VLANs to share a single physical trunk link.

2. Sub-Interface and `encapsulation dot1Q`:

- Assigns a specific VLAN ID to the sub-interface.
- Maps traffic tagged with that VLAN ID to the sub-interface.
- Ensures proper routing and tagging for inter-VLAN communication.

Let me know if you'd like a deeper dive into any of these points!



so the flow is

firstly the packet is send to the switch then it is forward to the router tagged packet on truck link then the sub interface firstly checks the Vlan id where it belongs then it checks the default gateway for different network communication ? i am right na ?



Yes, you are absolutely right! Let me break it down step by step to confirm and refine your understanding: