Network Architectures - Study Resume

Based on your slides, here's a comprehensive resume organized by topic to help you prepare for your practical test:

---

1. NETWORK MANAGEMENT (Gestão)

Why Network Management?

- **Lower costs** through automation
- **More efficient** planning and trend prediction
- **Better service** with automatic monitoring
- **Greater knowledge** for better decisions

ISO Management Areas (FCAPS)

- **Fault Management**: Detection, isolation, correction of anomalies
- **Configuration Management**: Control/collect data from network elements
- **Accounting Management**: Measure utilization, determine costs
- **Performance Management**: Evaluate/report equipment behavior
- **Security Management**: Secure communications management

Management Models

- **Systems Management**: Covers all company aspects
- **Network Management**: Focuses on network/communications
- **Centralized**: Agent-manager model
- **Distributed**: Shared management responsibilities
- **Hierarchical**: Centralized information at root

---

2. SNMP (Simple Network Management Protocol)

Key Concepts

- **Manager/Agent Paradigm**: Client-server model
- **Transport**: UDP (connectionless)
- **MIB**: Management Information Base (contains managed objects)

SNMP Operations

Message Types:

- **GetRequest/GetNextRequest/GetBulkRequest**: Manager requests data from agent
- **SetRequest**: Manager sets MIB value
- **Response**: Agent answers request
- **Trap**: Agent informs manager of exception (unreliable)

- **InformRequest**: Reliable trap alternative

Polling vs Traps

- **Polling**: Manager periodically asks for info
  - ✓ Complete control
  - ✗ Delay and bandwidth waste
- **Traps**: Event-driven notifications
  - ✓ Info only when needed
  - ✗ More resources, can be unreliable

SNMP Security

- **Community strings**: "public" (read-only), "private" (read-write)
- Case sensitive
- Basic authentication mechanism

Object Naming

- **ISO Object Identifier Tree**: Hierarchical naming (e.g., 1.3.6.1.2.1.7.1)

RMON (Remote Monitoring)

- **RMON1**: Ethernet monitoring (RFC 1757)
- **RMON2**: Upper layer monitoring (RFC 2021)
- **9 groups**: Statistics, History, Alarm, Host, HostTopN, Matrix, Filter, Packet Capture, Event

ASN.1 & TLV Encoding

- **ASN.1**: Formal language for data representation
- **TLV**: Type-Length-Value encoding for transmission

---

3. POLICY-BASED MANAGEMENT (PBM) & COPS

Concept

- Manage network **globally**, not individual elements
- Define **policies (rules)** for network behavior

PEP-PDP Model

- **PEP** (Policy Enforcement Point): Policy targets, enforce rules
- **PDP** (Policy Decision Point): Policy consumers, make decisions
- **Repository**: Stores policy rules

COPS Protocol

- **Common Open Policy Service**
- TCP-based, maintains state synchronization
- Two client types:

- **Outsourcing (RSVP)**: PEP contacts PDP when decision needed
- **Configuration (DiffServ)**: PDP configures PEP with equipment info

---

4. CMIS/CMIP (OSI Management)

Characteristics

- **Object-oriented** approach
- Objects have attributes, generate events, execute operations
- **Intelligent agents** with on-line rules
- **Actions**: GET, SET, CREATE, DELETE, ACTION, NOTIFICATION, CANCEL_GET

GDMO

- **Guideline for Definition of Managed Objects**
- More flexible but complex compared to SNMP

CMIP vs SNMP

SNMP	CMIP
Static MIBs	Dynamic MIBs
Polling model	Event-oriented
Lightweight	Heavy
Market dominant	Telecom focused
Limited functionality	Full system management

---

5. TMN (Telecommunications Management Network)

Architecture Layers (Bottom to Top)

1. **Network Element Layer (NEL)**: Fault detection, trap generation
2. **Element Management Layer (EML)**: Alarm management, logging, maintenance
3. **Network Management Layer (NML)**: Configuration, supervision, event correlation
4. **Service Management Layer (SML)**: Service handling, SLA monitoring, charging
5. **Business Management Layer (BML)**: Policies, trends, billing reports

Physical Architecture

- **Operation System (OS)**: Main management functionality
- **Mediation Equipment (MD)**: Intermediary processing
- **Workstation (WS)**: User access
- **Network Element (NE)**: Managed equipment
- **Q Adapter (QA)**: Connects non-TMN equipment
- **DCN**: Data Communication Network

Interfaces

- **Q3**: OS ↔ TMN elements (typically uses CMIP)
- **Qx**: MD ↔ NE/adapter
- **F**: WS ↔ OS/mediator
- **X**: TMN ↔ TMN/external network

---

6. QUALITY OF SERVICE (QoS)

Service Requirements

- **Packet loss**: Some apps tolerate losses (video/audio), others don't (file transfer)
- **Bandwidth**: Some need minimum (multimedia), others are elastic (email)
- **Delay/Jitter**: Critical for real-time apps (VoIP, games)

QoS Principles

1. **Packet marking**: Distinguish traffic types
2. **Policing**: Force compliance to agreed bandwidth
3. **Isolation**: Protect one class from another
4. **Efficient resource use**: Avoid waste
5. **Call admission**: Block calls if can't support requirements

---

7. TRAFFIC CONDITIONING

Leaky Bucket

- **Parameters**: p (bucket size), b (exit rate)
- Data enters at any rate, leaves at constant rate
- Limits bit rate to b

Token Bucket

- **Parameters**: b (bucket size), r (token rate), p (peak rate)
- Tokens generated at rate r
- Burst possible at rate p when tokens available
- Total data in time $t < rt + b$

Policing vs Shaping

- **Policing**: Drops or tags excess traffic
- **Shaping**: Buffers traffic, smooths bursts

---

8. CONGESTION CONTROL

Tail Drop

- Drops packets when buffer full
- **Problem**: TCP global synchronization

RED (Random Early Detection)

- **Parameters**: minQ, maxQ, AvgQ, maxP
- Drops packets **before** congestion
- Drop probability $\propto$ queue length
- Prevents global TCP synchronization

WRED (Weighted RED)

- Multiple drop levels for different priorities
- Higher priority traffic dropped last

---

9. SCHEDULING ALGORITHMS

FIFO (First In First Out)

- No QoS differentiation
- Simple but unfair

Priority Queuing

- Strict priority levels
- Higher priority always served first
- Can starve lower priority traffic

Fair Queuing (FQ)

- Equal bandwidth per non-empty queue
- Fair but inflexible

Weighted Fair Queuing (WFQ)

- Bandwidth allocation by weight
- Queue bandwidth = (weight / Σweights) × Link bandwidth
- More flexible and fair

---

10. INTEGRATED SERVICES (IntServ)

Service Classes

- **Guaranteed Service (GS)**: Delay guarantees for given bandwidth

- **Controlled Load (CL)**: Best-effort in unloaded network
- **Best Effort (BE)**: No guarantees

RSVP (Resource Reservation Protocol)

- **Signaling protocol** for IntServ
- Encapsulated in IP (protocol 46)
- **PATH messages**: Sender → Receiver (announce traffic)
- **RESV messages**: Receiver → Sender (confirm reservation)
- **Soft state**: Must be refreshed periodically

RSVP Message Flow

1. Receiver joins multicast group
2. Sender sends PATH message
3. Routers process PATH, forward downstream
4. Receiver sends RESV message
5. Routers reserve resources, forward upstream

Reservation Styles

- **Fixed Filter**: Reservation per sender
- **Wildcard Filter**: Single reservation for any sender
- **Shared Explicit**: Single reservation for listed senders

FlowSpec & FilterSpec

- **FlowSpec**: Defines traffic parameters (token bucket)
- **FilterSpec**: Identifies packets in flow (addresses/ports)

IntServ Problems

- **Does NOT scale** in core networks
- Per-flow state required
- Complex signaling

---

11. DIFFERENTIATED SERVICES (DiffServ)

Key Principles

- **No per-flow state** in core
- **Packet marking** at edges
- **Traffic aggregation** into classes
- **Scalable** for large networks

DSCP (DiffServ Code Point)

- 6 bits in IP TOS/Traffic Class field
- Marks packet's PHB (Per-Hop Behavior)

PHB Classes

Expedited Forwarding (EF) - DSCP: 101110

- "Virtual leased line"
- Low delay, low jitter, no losses
- Strict priority queuing
- Conservative bandwidth reservation

Assured Forwarding (AF) - DSCP: aaadd0

- 4 classes (AF1-AF4), 3 drop precedences each
- Relative QoS guarantees
- In-profile packets rarely dropped
- Out-profile treated as best-effort

Default (BE) - DSCP: 000000

- Best-effort, FIFO

Edge Router Functions

- **Classifier**: Identify traffic class
- **Meter**: Check against SLA (token bucket)
- **Marker**: Set DSCP (in/out profile)
- **Dropper**: Remove out-of-profile packets
- **Shaper**: Delay out-of-profile packets

Core Router Functions

- **Forward** based on DSCP/PHB
- Simple, fast processing
- No per-flow state

DiffServ vs IntServ

IntServ	DiffServ
Per-flow	Aggregated
E2E guarantees	Per-hop behaviors
Complex signaling (RSVP)	Simple marking
Does NOT scale	Scales well
Application-oriented	Provider-oriented

Integration

- IntServ flows can map to DiffServ classes
- Edge routers translate RSVP → DSCP
- Complementary approaches

---

12. BGP (Border Gateway Protocol)

Basic Concepts

- **Path-vector protocol** (not distance-vector!)
- Routes between **Autonomous Systems (AS)**
- Uses **TCP port 179**
- BGP4 current version

AS Numbers

- **2-byte**: 1-64511 (public), 64512-65535 (private)
- **4-byte**: RFC 4893, notation X.Y (e.g., 100.1)
- **AS_TRANS (23456)**: Placeholder for backward compatibility

BGP Types

- **eBGP**: Between different AS
- **iBGP**: Within same AS
    - Requires **full mesh** or route reflectors
    - Does NOT modify AS-PATH

BGP Messages

1. **OPEN**: Establish session, negotiate capabilities
2. **UPDATE**: Advertise/withdraw routes + attributes
3. **KEEPALIVE**: Maintain session
4. **NOTIFICATION**: Error, then close session

UPDATE Message Components

- **Withdrawn routes**: No longer reachable
- **Path attributes**: Routing/policy parameters
- **NLRI**: Reachable IP networks

---

13. BGP ATTRIBUTES

Well-Known Mandatory

AS-PATH (Type: Well-known mandatory)

- Ordered list of AS numbers traversed
- Loop detection mechanism
- eBGP adds own AS, iBGP doesn't

NEXT-HOP (Type: Well-known mandatory)

- IP to reach advertising router
- eBGP: peer connection IP
- iBGP: eBGP next-hop carried into AS

- Can configure "next-hop-self"

ORIGIN (Type: Well-known mandatory)

- How BGP learned route:
    - **IGP (0)**: network command
    - **EGP (1)**: obsolete
    - **INCOMPLETE (2)**: redistribution

Well-Known Discretionary

LOCAL_PREF (Type: Well-known discretionary)

- Choose **exit point** from local AS
- **Higher** value preferred
- Propagated within AS only
- Influences **outbound** traffic

ATOMIC_AGGREGATE (Type: Well-known discretionary)

- Indicates route aggregation occurred
- Specific routes lost

Optional Transitive

AGGREGATOR (Type: Optional transitive)

- AS and router IP that performed aggregation

COMMUNITY (Type: Optional transitive)

- Group routes with common properties
- Format: AS:number (e.g., 300:1)
- Predefined: no-export, no-advertise, internet

AS4_PATH / AS4_AGGREGATOR (Type: Optional transitive)

- Carry 4-byte AS numbers
- For backward compatibility

Optional Non-Transitive

MED (Multi-Exit Discriminator) (Type: Optional non-transitive)

- Suggest preferred path to external AS
- **Lower** value preferred
- Influences **inbound** traffic
- Not propagated beyond neighboring AS

Cisco-Specific

WEIGHT (Cisco only, local)

- Local to router, not advertised
- **Highest** value preferred

- First in selection process

---

14. BGP PATH SELECTION

Order of preference:

1. **Highest WEIGHT** (Cisco only)
2. **Highest LOCAL_PREF**
3. **Locally originated** route
4. **Shortest AS-PATH**
5. **Lowest ORIGIN** (IGP < EGP < INCOMPLETE)
6. **Lowest MED**
7. **eBGP over iBGP**
8. **Closest IGP neighbor**

---

15. ADVANCED BGP CONCEPTS

BGP Synchronization

- Don't advertise iBGP route to eBGP until IGP learns it
- Ensures AS can forward transit traffic
- Usually disabled in modern networks

Route Reflectors

- Avoids full iBGP mesh requirement
- **Route Reflector (RR)**: Reflects routes between clients
- **Clients**: Peer only with RR
- **Non-clients**: Full mesh among themselves
- RRs should peer with each other

Private AS Numbers

- Range: 64512-65535
- Used for customer networks
- ISP removes with `remove-private-as` command
- Not suitable for multi-homing

BGP Communities (Practical)

- Control routing policy
- Example: Set LOCAL_PREF based on community
- Common format: ISP_AS:action_code
- Enables customer traffic engineering

Route Filtering

- **Prefix lists**: Filter by network prefix
- **AS-PATH filters**: Filter by AS path (regex)
- **Route maps**: Complex conditional logic
    - Match conditions (prefix, AS-PATH, community)
    - Set actions (LOCAL_PREF, MED, community)

Redistribution

- **IGP → BGP**: Announces internal networks (OK)
- **BGP → IGP**: Fills internal tables (usually BAD)
    - Increases routing table size
    - Slows convergence
    - Use default routes instead

BGP over Tunnels

- Solves BGP/IGP routing conflicts
- IP-IP tunnel between BGP peers
- Tunnel IPs distributed via IGP
- BGP next-hop is tunnel endpoint

---

16. MP-BGP (Multi-Protocol BGP)

Purpose

- Extends BGP for multiple address families:
    - IPv6 Unicast/Multicast
    - MPLS VPN (IPv4/IPv6)
    - L2VPN (VXLAN EVPN)
    - 6PE (IPv6 over MPLS)

New Attributes

MP_REACH_NLRI (Optional non-transitive)

- Reachable destinations
- AFI/SAFI (Address Family Identifier / Sub-AFI)
- Next-hop info
- NLRI (Network Layer Reachability Info)

MP_UNREACH_NLRI (Optional non-transitive)

- Unreachable destinations

Capability Negotiation

- Exchanged in OPEN message
- Includes: AFI/SAFI support, Route Refresh, Outbound Filtering

17. MPLS (Multi-Protocol Label Switching)

Purpose

- Simplifies packet forwarding
- Enables **Traffic Engineering**
- Single protocol layer
- Supports multiple services

Key Concepts

- **Label**: 20-bit identifier
- **FEC** (Forwarding Equivalence Class): Group of packets with same treatment
- **LSP** (Label Switched Path): Path through MPLS network
- **LSR** (Label Switch Router): Core router doing label swapping
- **LER** (Label Edge Router): Edge router (PE) doing label push/pop

Label Format

```
[Label 20 bits][Exp 3 bits][S 1 bit][TTL 8 bits]
```

- **Label**: Value
- **Exp**: Experimental (QoS)
- **S**: Bottom of Stack (1=last label)
- **TTL**: Time to Live

Label Operations

- **PUSH**: Add label (ingress)
- **SWAP**: Change label (core)
- **POP**: Remove label (egress or PHP)

PHP (Penultimate Hop Popping)

- Second-to-last router pops label
- Reduces load on egress PE
- Egress PE only does IP lookup

Label Stacking

- Multiple labels in a stack
- Outer label: transport in core
- Inner label(s): services, VPNs
- Bottom of Stack bit identifies last label

18. MPLS PROTOCOLS

LDP (Label Distribution Protocol)

- Dynamic label distribution
- **Discovery**: Hello messages (UDP multicast 224.0.0.2)
- **Session**: TCP port 646
- **Messages**:
    - Discovery: Hello
    - Session: Initialization, KeepAlive
    - Advertisement: Label Mapping, Withdraw, Release
    - Notification: Errors

LDP Operation

1. Routers send periodic Hellos
2. Router with higher IP initiates TCP session
3. Exchange Initialization messages
4. Exchange KeepAlives
5. Exchange label bindings
6. Session maintained with periodic KeepAlives

RSVP-TE (Traffic Engineering)

- Evolution of RSVP for MPLS
- **Explicit routes** (ER): Defined path
- **Label Request/Label** objects
- **Record Route**: Path tracking
- **Session Attribute**: Priority, tunnel name

RSVP-TE Messages

- **PATH**: Sender → Receiver, request label
- **RESV**: Receiver → Sender, provide label
- Similar to IntServ RSVP but with labels

OSPF-TE Extensions

- Type 10 Opaque LSAs (area flooding)
- **Router Address TLV**: Stable router ID
- **Link TLV**: Link attributes
    - Link type, ID
    - Local/remote IP
    - TE metric
    - Max bandwidth
    - Max reservable bandwidth
    - Unreserved bandwidth (8 priorities)
    - Admin group

---

19. MPLS L3 VPN

Key Concepts

- **VRF** (Virtual Routing and Forwarding): Separate routing table per customer
- **RD** (Route Distinguisher): Makes routes unique (64-bit)
- **RT** (Route Target): Controls import/export (64-bit)
- **VPNv4**: RD:IPv4 address (96 bits)

Devices

- **CE** (Customer Edge): Customer router
- **PE** (Provider Edge): VRF-aware, runs MP-BGP
- **P** (Provider): Core router, label switching only

VRF Components

- Separate routing table
- Associated with interfaces
- RD: Makes customer routes unique globally
- RT: Controls which routes to import/export

RD vs RT

- **RD**: Makes route unique (ASN:number or IP:number)
    - E.g., 100:1
- **RT**: Policy tool, extended community
    - Export RT: attached to routes sent
    - Import RT: which routes to accept
    - Allows complex topologies (hub-spoke, extranet)

MP-BGP for VPN

- **Address Family**: VPNv4 (AFI=1, SAFI=128)
- **MP_REACH_NLRI**: Carries RD:prefix, RT, label
- Only PEs run MP-BGP (not P routers!)
- Core is "BGP-free"

Label Stack in VPN

- **Outer label**: IGP label (get to PE)
- **Inner label**: VPN label (identify VRF)
- P routers only see outer label
- Penultimate P pops outer label
- Egress PE uses inner label to select VRF

Packet Flow

1. CE sends IP packet to PE
2. PE performs VRF lookup

3. PE pushes 2 labels [IGP, VPN]
4. P routers swap IGP label
5. Penultimate P pops IGP label (PHP)
6. Egress PE uses VPN label to select VRF
7. Egress PE pops VPN label, forwards IP to CE

---

20. TRAFFIC ENGINEERING (TE)

Problem

- Routing protocols use shortest path
- Doesn't consider link utilization
- Can cause congestion on some links while others underutilized

Solution

- **Manipulate traffic paths** to fit network
- MPLS enables TE through explicit paths
- Override shortest path routing

TE Requirements

1. Specify path constraints (bandwidth, delay)
2. Extend topology database (resource info)
3. Find paths that meet constraints
4. Signal to reserve resources
5. Set up LSP along path
6. Map traffic to appropriate LSPs

Implementation

- **OSPF-TE/ISIS-TE**: Distribute link state + TE info
- **RSVP-TE**: Signal explicit paths, reserve bandwidth
- **CSPF** (Constrained Shortest Path First): Calculate TE paths

---

21. LAYER 2 VPN (VXLAN & EVPN)

Datacenter Evolution

- **Traditional**: 3-tier (access-aggregation-core), STP-based
- **Modern**: Spine-Leaf (CLOS), Layer 3 to access, ECMP

VXLAN (Virtual Extensible LAN)

- **Encapsulates** L2 frames in L4 UDP
- **UDP port**: 4789
- **VNI** (VXLAN Network Identifier): 24 bits (vs 12-bit VLAN)

- **VTEP** (VXLAN Tunnel Endpoint): Encap/decap device

VXLAN Header

[Outer Ether][Outer IP][UDP][VXLAN][Inner Ether][Payload]

- Allows L2 over L3 network
- Scales beyond VLAN limits

Problem: Flood and Learn

- Unknown unicast/multicast/broadcast flooded
- MAC learning from data plane
- Inefficient at scale

BGP EVPN Solution

- **Control plane** for VXLAN
- Uses MP-BGP (AFI=25, SAFI=70)
- Distributes MAC/IP info without flooding

EVPN Route Types

Type 2 (MAC/IP Advertisement)

- Announces MAC + IP + next-hop
- Sent when new MAC learned
- Enables unicast forwarding

Type 3 (Inclusive Multicast)

- Announces VTEP address
- Creates BUM (Broadcast, Unknown unicast, Multicast) distribution list
- Uses **Ingress Replication**
- Sent when new VTEP added

Type 5 (IP Prefix)

- Announces IP prefixes
- Enables L3 VPN over VXLAN
- Alternative to MPLS L3 VPN

BGP EVPN Peering

- iBGP or eBGP between leafs
- **Route Reflectors** (usually spines) for iBGP
- Private AS per leaf for eBGP

EVPN Extended Communities

- **Encapsulation**: VXLAN (type 8)
- **Route Target**: VNI membership

---

22. CDN (Content Delivery Networks)

Purpose

- **Reduce latency** to end users
- **Reduce load** on origin server
- **Avoid congestion** in network
- Handle **flash crowds**

CDN Generations

1. **1st Gen (90s)**: Static caching, web acceleration
2. **2nd Gen (2000s)**: Dynamic content, multimedia
3. **3rd Gen (2010s)**: Cloud integration, UGC, mobile
4. **4th Gen (2020+)**: Edge computing, MEC, federation

CDN Components

- **Origin server**: Original content source
- **Surrogate servers**: Cached copies at edge
- **Distribution infrastructure**: Moves content to surrogates
- **Request routing**: Directs clients to best surrogate
- **Accounting**: Logs, reports

CDN vs Caching Proxy

CDN	Caching Proxy
Content provider controlled	ISP controlled
Proactive	Reactive
Global distribution	Local to ISP
Accounting to content owner	Reduces ISP bandwidth

Request Routing Methods

DNS Redirection

- Client requests www.foo.com
- Authoritative DNS returns CNAME → cdn.example.com
- CDN DNS returns IP of nearby surrogate
- Short TTL for flexibility
- ✓ Uses existing infrastructure
- ✗ Only sees DNS server IP

URL Rewriting

- Origin rewrites URLs to point to CDN
- E.g., `<img src="cdn.example.com/image.jpg">`

HTTP Redirection

- Origin server sends HTTP 302
- Redirects to surrogate

Anycast

- Same IP announced from multiple locations
- BGP routes to "nearest"

Content Types

1. **Base HTML**: May stay at origin
2. **First-party static** (images, CSS, JS): Ideal for CDN
3. **Third-party** (analytics, ads): Often separate CDN

CDN Challenges

- **Consistency**: Keep surrogates synchronized
- **Session state**: Share across surrogates
- **Security**: Distributed authentication
- **Cost**: Multiple PoPs to operate

PRACTICAL TEST TIPS

Wireshark Analysis - What to Look For

SNMP

- UDP port 161 (agent), 162 (trap)
- Community string visible in cleartext
- Message types: GetRequest, GetResponse, SetRequest, Trap
- OID structure (1.3.6.1...)
- MIB values

RSVP

- IP protocol 46
- PATH messages (sender → receiver)
- RESV messages (receiver → sender)
- Objects: SESSION, SENDER_TEMPLATE, FLOWSPEC
- Refresh period (soft state)

BGP

- TCP port 179
- OPEN: AS number, hold time, capabilities
- UPDATE: NLRI, withdrawn routes, path attributes
- KEEPALIVE: empty message
- AS-PATH attribute format
- Community attribute

COPS

- TCP connection
- Client-Type (RSVP=1, DiffServ=2)

- REQ (request), DEC (decision)
- Handle for flow identification

MPLS

- Look between L2 and L3 headers
- Label stack (multiple labels possible)
- TTL decrement
- Bottom of Stack bit

VXLAN

- UDP port 4789
- VNI in VXLAN header
- Inner Ethernet frame
- Outer IP = VTEP addresses

True/False - Common Traps

❌ **SNMP is connection-oriented** → FALSE (uses UDP) √ **RSVP creates soft state** → TRUE (requires refresh) ❌ **IntServ scales well in core networks** → FALSE (per-flow state) √ **DiffServ aggregates traffic into classes** → TRUE ❌ **iBGP modifies AS-PATH** → FALSE (only eBGP does) √ **Local Preference is used within an AS** → TRUE ❌ **MED influences outbound traffic** → FALSE (inbound traffic) √ **MPLS can use label stacking** → TRUE ❌ **LDP uses UDP for sessions** → FALSE (TCP port 646) √ **VRF provides routing separation** → TRUE ❌ **VXLAN uses 12-bit identifiers** → FALSE (24-bit VNI) √ **EVPN Type-2 announces MAC/IP** → TRUE

Completion - Key Terms to Know

- **SNMP community string**: Authentication mechanism
- **MIB**: Management Information Base
- **RSVP PATH**: Sender announces traffic characteristics
- **RSVP RESV**: Receiver confirms reservations
- **Token bucket parameters**: r (rate), b (bucket), p (peak)
- **RED parameters**: minQ, maxQ, AvgQ, maxP
- **IntServ service types**: GS, CL, BE
- **DiffServ PHB**: EF, AF, BE
- **AS-PATH**: Ordered list of AS numbers
- **LOCAL_PREF**: Chooses exit point (higher wins)
- **MED**: Suggests entry point (lower wins)
- **WEIGHT**: Cisco-specific (highest wins)
- **VRF**: Virtual Routing and Forwarding
- **RD**: Route Distinguisher (makes routes unique)
- **RT**: Route Target (import/export control)
- **FEC**: Forwarding Equivalence Class
- **LSP**: Label Switched Path
- **PHP**: Penultimate Hop Popping
- **VTEP**: VXLAN Tunnel Endpoint
- **VNI**: VXLAN Network Identifier (24-bit)

- **Ingress Replication**: EVPN multicast method

QUICK REFERENCE TABLES

Port Numbers

Protocol	Port	Transport
SNMP Agent	161	UDP
SNMP Trap	162	UDP
BGP	179	TCP
LDP	646	TCP/UDP
RSVP	-	IP (46)
COPS	-	TCP
VXLAN	4789	UDP

QoS Comparison

Feature	IntServ	DiffServ
Granularity	Per-flow	Aggregate
Signaling	RSVP	None
Scalability	Poor	Good
Guarantees	Hard	Soft
Complexity	High	Medium
Where used	Edge	Core

BGP Attribute Summary

Attribute	Type	Higher/Lower	Scope
WEIGHT	Cisco	Higher	Local
LOCAL_PREF	WK-D	Higher	AS
AS-PATH	WK-M	Shorter	Global
ORIGIN	WK-M	IGP best	Global
MED	Opt-NT	Lower	Neighbor AS

Label Operations

Operation	Where	Function
PUSH	Ingress LER	Add label
SWAP	LSR	Change label
POP	Egress LER or PHP	Remove label

Good luck with your test on Monday! Focus on understanding the **concepts** and **relationships** between protocols rather than just memorizing facts. The practical nature of the test suggests you'll need to **recognize protocols in Wireshark** and **understand their behavior**.