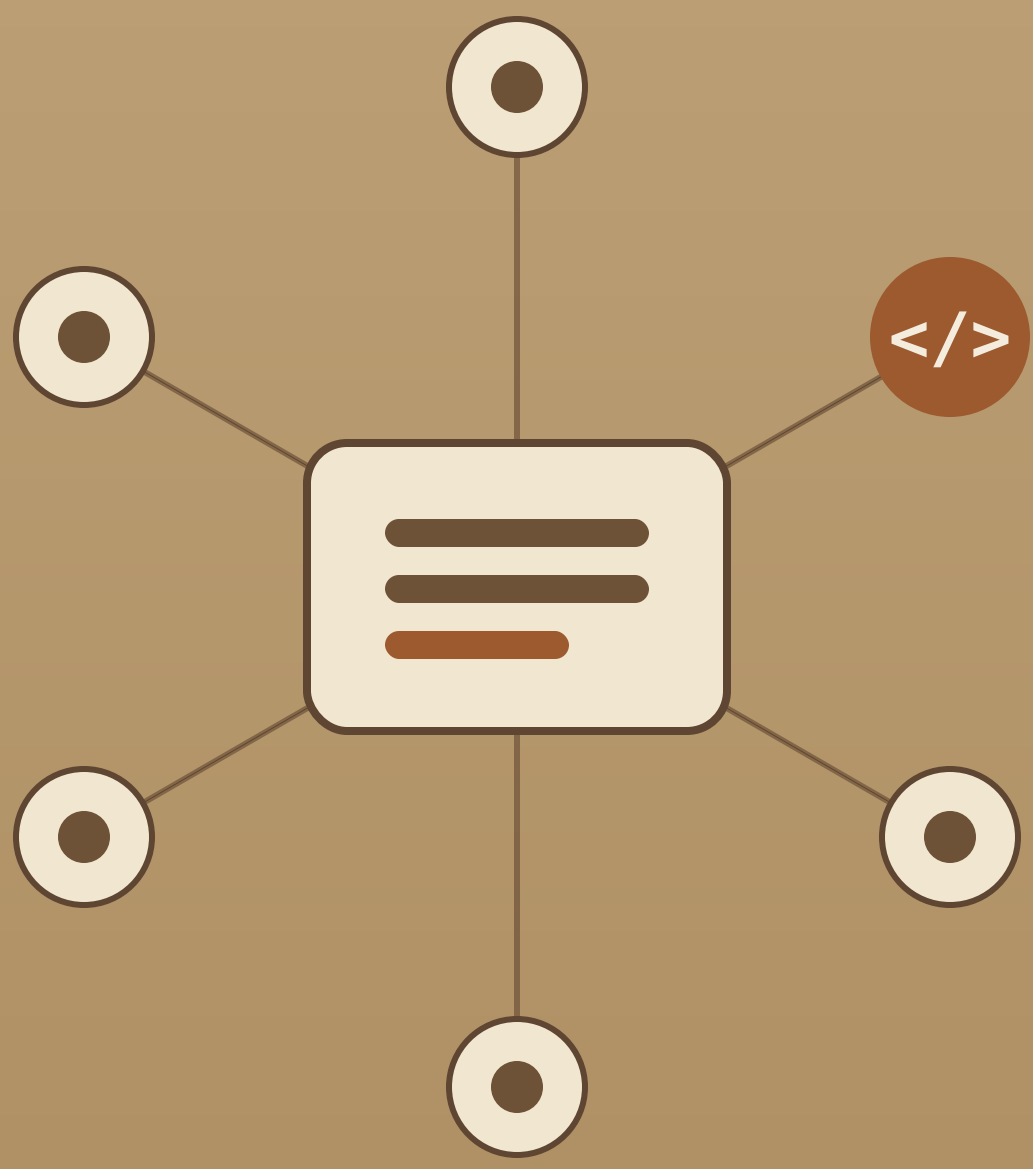
CCNA AUTOMATION · EXAM 200-901

CCNA Automation 200-901

A Complete Learning Guide for Beginners

Independent · Unofficial Study Guide



6 CHAPTERS · 28 SUB-SECTIONS · 140 QUESTIONS

Covers every topic in the 200-901 exam blueprint

JOZEF BAROŠ

AUTHOR

CCNA AUTOMATION CERTIFICATION · EXAM 200-901

CCNA Automation

200-901

A Complete Learning Guide for Beginners

Every exam topic explained from the ground up — with plain-English analogies, runnable code, exam tips, and practice quizzes.

6 CHAPTERS · 28 SUB-SECTIONS · 140 PRACTICE QUESTIONS

Covers every topic in the 200-901 exam blueprint

An independent, unofficial study guide — not affiliated with or endorsed by Cisco Systems, Inc.

JOZEF BAROŠ

AUTHOR

Table of Contents

Chapter 1 — Network Fundamentals

Welcome to the very first chapter. Before you can teach a computer to **automate** a network, you have to understand what a network actually is and how its pieces talk to each other. Think of this chapter as learning the rules of the road before you build a self-driving car. Everything in the rest of this book — APIs, Python scripts, Cisco platforms, infrastructure-as-code — sits on top of the ideas you meet here.

On the official exam this material lives in **Domain 6.0 (Network Fundamentals)** and is worth about **15%** of your score. We put it first on purpose: it is the foundation. We will explain every term as if you have never seen a network before, lean on everyday analogies, and finish each part with a five-question Cisco-style quiz so the ideas stick.

Here is the map of this chapter:

Sub-section	What you will learn	Exam focus
1.1	MAC addresses and VLANs — how devices are named and grouped	Describe
1.2	IP addresses, subnets, routes, gateways — how data finds its way	Describe
1.3	Switches, routers, firewalls, load balancers, and topology diagrams	Describe / Interpret
1.4	The three planes and core IP services: DHCP, DNS, NAT, SNMP, NTP	Describe
1.5	Ports, connectivity troubleshooting, and how the network limits apps	Describe / Diagnose

> ✅ **Exam Tip — Describe, do not configure**
>
> Domain 6.0 asks you to **describe** and **recognize** — what a thing is, what it is for, and how to read a diagram. You are not asked to type long router configurations from memory. Aim to recognize the right answer, not to be a network engineer (yet).

1.1 How Networks Talk: MAC Addresses & VLANs

What is a network, really?

Imagine a big apartment building full of people who want to pass notes to each other. A **network** is just that: a set of devices (computers, phones, printers, routers) connected so they can pass notes — we call those notes **frames** and **packets** — back and forth. For any note to arrive, the sender needs two things: a way to **name** the receiver, and a **path** to reach them. This sub-section is about naming at the lowest level.

MAC addresses: a device's permanent name tag

Every network card is born with a unique sticker baked into its hardware called a **MAC address** (Media Access Control address). Think of it like the serial number stamped on the back of a toy: it never changes, and no two are supposed to be the same. A MAC address is **48 bits** long and is written as six pairs of hexadecimal digits, like this:

A MAC address — six pairs of hex, often separated by colons

```
00:1A:2B:3C:4D:5E
|————|  |————|
  OUI       Device ID
(who made    (which exact
 the card)    card it is)
```

The first half (the **OUI**, Organizationally Unique Identifier) says who manufactured the card — Cisco, Apple, Intel. The second half is a unique number that vendor assigned to that one card. Put them together and you get a name tag that is unique in the whole world.

Because the MAC address is built into the hardware, it works only on the **local** network — the same building. It is like shouting a person's full name across one room: useful in that room, useless across the city. To cross between buildings you need IP addresses, which we meet in 1.2.

Reading the vendor out of a MAC address in Python

```
mac = "00:1A:2B:3C:4D:5E"
oui = mac.replace(":", "")[:6]      # first 3 bytes
print("Vendor prefix (OUI):", oui)  # -> 001A2B

# Is this a broadcast frame? (all ones = FF:FF:FF:FF:FF:FF)
print("Broadcast?", mac.upper() == "FF:FF:FF:FF:FF:FF")
```

> 💡 **Did you know? — The 'locally administered' bit**
>
> The second hex digit of a MAC tells you secrets. If it is 2, 6, A, or E, the address was set by software, not the factory. That is why your phone can use a different, random MAC on each Wi-Fi network to protect your privacy.

Switches and the MAC address table

The device that moves frames around inside one building is a **switch**. A switch is like a smart mail clerk on one floor: when a frame arrives, it reads the destination MAC address and sends the frame out only the

one port where that device lives. To do this it keeps a cheat-sheet called the **MAC address table** — a list of which MAC address is plugged into which port. It learns this automatically by watching the source address of every frame that comes in.

VLANs: invisible walls inside one switch

Now imagine that apartment building has one big shared hallway, but you want the accountants to be separated from the guests — same building, different groups. A **VLAN** (Virtual Local Area Network) does exactly that. It lets one physical switch pretend to be several separate switches by tagging frames with a number (the **VLAN ID**, 1 to 4094). Devices in VLAN 10 can talk to each other but cannot directly reach devices in VLAN 20, even if they are plugged into the same box.

Why bother? Three big reasons: **security** (keep sensitive groups apart), **organization** (group by department, not by where the cable physically goes), and **less noise** (a broadcast in VLAN 10 does not bother VLAN 20).

Putting a switch port into VLAN 10 (Cisco IOS style — read it, do not memorize it)

```
Switch(config)# vlan 10
Switch(config-vlan)# name ACCOUNTING
Switch(config)# interface GigabitEthernet0/1
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 10
```

🌍 **From the Field — Why automation engineers care**

In a real data center you might have hundreds of switches and thousands of ports. Nobody puts each port into a VLAN by hand — that is precisely the boring, repeatable work you will automate later in this book with tools like Ansible and Python. Understanding VLANs now means your future scripts will make sense.

⚠️ **Common Pitfall — Confusing a VLAN with a subnet**

A VLAN is a Layer-2 grouping (MAC level); a subnet is a Layer-3 grouping (IP level). They usually map one-to-one in practice, so people mix them up. On the exam, remember: VLANs separate **broadcast domains**; subnets separate **IP ranges**. Two different layers, two different jobs.

Summary

A network is just devices passing notes. Each network card has a permanent 48-bit **MAC address** — a worldwide-unique name tag whose first half identifies the vendor (OUI). A **switch** moves frames inside one local network by learning which MAC lives on which port in its MAC address table. A **VLAN** carves one physical switch into several logical ones using a VLAN ID, separating groups for security, organization, and quieter broadcasts. MAC addresses work only locally; crossing between networks needs IP.

Key Takeaways

MAC address = 48 bits, six hex pairs; **OUI = vendor** half. **Switch** = forwards frames using the MAC table, learned from source addresses. **VLAN** = logical network on a switch, ID **1–4094**, separates broadcast domains. VLANs improve **security, organization, broadcast control**. **VLAN ≠ subnet** — Layer 2 vs Layer 3.

Knowledge Check — 1.1 MAC Addresses & VLANs

Q1. How many bits long is a standard MAC address?

- **A.** 32 bits
- **B.** 64 bits
- **C.** 48 bits
- **D.** 128 bits

Correct answer: C. A MAC address is 48 bits, written as six pairs of hexadecimal digits. 32 bits is an IPv4 address; 128 bits is IPv6.

Q2. What does the first half (OUI) of a MAC address identify?

- **A.** The hardware vendor that made the card
- **B.** The VLAN the device belongs to
- **C.** The device's IP address
- **D.** The switch port number

Correct answer: A. The OUI (Organizationally Unique Identifier) identifies the manufacturer. The second half uniquely identifies the specific card.

Q3. What is the main purpose of a VLAN?

- **A.** To assign IP addresses automatically
- **B.** To encrypt all traffic on a switch
- **C.** To increase the physical cable length
- **D.** To logically separate devices into different broadcast domains on shared hardware

Correct answer: D. A VLAN splits one physical switch into several logical networks, separating broadcast domains for security and organization. It does not assign IPs (that is DHCP) or encrypt traffic.

Q4. How does a switch learn which MAC address is on which port?

- **A.** By asking a DNS server
- **B.** By reading the source MAC address of incoming frames
- **C.** By reading the destination IP address
- **D.** An administrator must type every entry manually

Correct answer: B. A switch populates its MAC address table dynamically by recording the source MAC of each frame it receives on a port.

Q5. Which statement is TRUE about MAC addresses?

- **A.** They are used for communication within the local network
- **B.** They route traffic across the internet
- **C.** They change every time a device reboots
- **D.** They are 4 bytes long

Correct answer: A. MAC addresses operate at Layer 2 for local delivery. Routing across networks uses IP addresses (Layer 3). The factory MAC does not change on reboot, and it is 6 bytes (48 bits).

1.2 Finding the Way: IP Addresses, Subnets, Routes & Gateways

IP addresses: the mailing address of a device

A MAC address is like your name; an **IP address** is like your full mailing address. It tells the network not just **who** you are but **where** you are, so a note can travel across many buildings and cities to reach you. The most common form, **IPv4**, is **32 bits** written as four numbers from 0 to 255, separated by dots:

An IPv4 address and what each part means

```
192.168.  1.       10
|———————|         |
 Network part      Host part
 (which street)    (which house)
```

Part of the address says **which network** (the street) and part says **which device** (the house number). But how do we know where the street part ends and the house part begins? That is the job of the **subnet mask**.

Subnet masks and prefixes: drawing the street boundary

A **subnet mask** marks which bits of the IP are the network part. A common one is `255.255.255.0`, which we can also write as a **prefix** `/24` — meaning the first 24 bits are the network. Think of `/24` as saying: the first three numbers are the street name; the last number is the house. So `192.168.1.10/24` and `192.168.1.20/24` are neighbours on the same street and can talk directly; `192.168.2.10/24` is on a different street and needs help to be reached.

Python's ipaddress module does the subnet math for you

```
import ipaddress

net = ipaddress.ip_network("192.168.1.0/24")
print("Network address:", net.network_address)   # 192.168.1.0
print("Broadcast:", net.broadcast_address)        # 192.168.1.255
print("Usable hosts:", net.num_addresses - 2)     # 254
print("Subnet mask:", net.netmask)                # 255.255.255.0

a = ipaddress.ip_address("192.168.1.10")
b = ipaddress.ip_address("192.168.2.10")
print("Same /24 network?", a in net, b in net)    # True False
```

> 💡 **Did you know? — Why minus two?**
>
> In every IPv4 subnet, the first address is reserved as the **network address** (the street's name) and the last is the **broadcast address** (shout to everyone on the street). So a /24 has 256 addresses but only 254 are usable by real devices.

The default gateway: the door to the outside world

If a device wants to reach someone on a **different** street, it cannot get there alone. It hands the packet to its **default gateway** — almost always the local **router** — which knows how to forward it onward. The gateway is like the mailroom at the building's front door: anything addressed outside the building goes

there first. A device with the wrong gateway is like a letter with no stamp — it can reach neighbours but never leaves the street.

Routes: the map routers use

A **route** is a single line in a router's map that says, in effect, 'to reach network X, send the packet toward Y.' Routers stack many of these lines into a **routing table** and always pick the most specific match. A special route, the **default route** (`0.0.0.0/0`), means 'anything I do not have a specific rule for, send this way' — usually toward the internet.

A simplified routing table — read it top to bottom

```
Destination        Next hop / Interface     Meaning
192.168.1.0/24     directly connected       my own street
10.0.0.0/8         via 192.168.1.254        the company network
0.0.0.0/0          via 192.168.1.1          everything else (internet)
```

🌍 **From the Field — Reading routes is an exam and a job skill**

Later in this book you will write scripts that pull routing tables from devices via APIs and decide whether the network is healthy. Being able to glance at a table and say 'that 0.0.0.0/0 is the default route to the internet' is the same skill the exam tests and the job demands.

⚠️ **Common Pitfall — Forgetting hosts on different subnets need a router**

Two devices with the same IP network/prefix talk directly. The instant their network parts differ, traffic MUST pass through a gateway/router — even if the cable physically connects them. A classic troubleshooting miss is two PCs that 'should' talk but are in different subnets with no router between them.

Summary

An **IP address** locates a device across networks; IPv4 is 32 bits in four dotted numbers. The **subnet mask** (or prefix like /24) splits the address into a network part and a host part, deciding who is a local neighbour. Devices on the same network talk directly; to reach other networks they send packets to their **default gateway** (the router). Routers consult a **routing table** of routes, preferring the most specific, with `0.0.0.0/0` as the catch-all default route.

Key Takeaways

IPv4 = 32 bits, four octets 0–255. **Subnet mask / prefix** marks the network bits (`/24` = `255.255.255.0`). **First address = network**, **last = broadcast**, so a /24 has **254 usable hosts**. **Default gateway** = router that reaches other networks. **0.0.0.0/0** = default route. Same subnet = direct; different subnet = via router.

Knowledge Check — 1.2 IP Addresses, Subnets & Gateways

Q1. How many bits is an IPv4 address?

- **A.** 48 bits
- **B.** 32 bits
- **C.** 64 bits
- **D.** 128 bits

Correct answer: B. IPv4 is 32 bits (four 8-bit octets). 48 bits is a MAC address; 128 bits is IPv6.

Q2. What does the subnet mask determine?

- **A.** The MAC address of the gateway
- **B.** The speed of the connection
- **C.** The DNS server to use
- **D.** Which part of the IP address is the network and which is the host

Correct answer: D. The subnet mask marks the network bits versus host bits, deciding which addresses are local neighbours.

Q3. A host needs to send traffic to a device on a different subnet. Where does it send the packet first?

- **A.** To its default gateway (the router)
- **B.** Directly to the destination's MAC address
- **C.** To the broadcast address
- **D.** To the DNS server

Correct answer: A. Traffic destined for another subnet is sent to the default gateway, which forwards it onward. Direct MAC delivery only works within the same subnet.

Q4. How many usable host addresses are in a /24 network?

- **A.** 256
- **B.** 255
- **C.** 254
- **D.** 128

Correct answer: C. A /24 has 256 total addresses, minus the network and broadcast addresses, leaving 254 usable for hosts.

Q5. What does the route 0.0.0.0/0 represent in a routing table?

- **A.** A broadcast to all devices
- **B.** The loopback address
- **C.** An invalid or blackhole route
- **D.** The default route — used for any destination with no more specific match

Correct answer: D. 0.0.0.0/0 is the default route: the least specific match, used when no other route fits, typically pointing toward the internet.

1.3 The Building Blocks: Switches, Routers, Firewalls & Load Balancers

Now that we can name devices (MAC) and locate them (IP), let us meet the machines that move and guard the traffic. Picture a busy city. Each of these devices plays one clear role — once you know the role, reading a network diagram becomes easy.

Switch — the floor's mail clerk

A **switch** connects devices inside the **same** local network and forwards frames using MAC addresses, as we saw in 1.1. It works at **Layer 2**. Analogy: a clerk on one office floor who hand-delivers mail between desks on that floor. Fast, local, MAC-based.

Router — the city's intersection

A **router** connects **different** networks together and forwards packets using IP addresses. It works at **Layer 3**. Analogy: a road intersection with signs pointing to other towns. Whenever traffic must leave its home subnet, a router decides which way it goes. The router is also your **default gateway**.

Firewall — the security guard at the door

A **firewall** inspects traffic and allows or blocks it based on rules — by address, port, or application. Analogy: a guard with a clipboard checking who is allowed in and out. A firewall does not speed traffic up; its whole job is to decide **yes** or **no** for safety.

Load balancer — the restaurant host

A **load balancer** sits in front of several identical servers and spreads incoming requests across them so no single server is overwhelmed. Analogy: a restaurant host who seats arriving guests at whichever table is free, keeping the wait short. If one server dies, the load balancer simply stops sending it guests.

Device	Layer	Decides using	Everyday analogy
Switch	Layer 2	MAC address	Floor mail clerk (local delivery)
Router	Layer 3	IP address	City intersection (between networks)
Firewall	L3–L7	Rules: address, port, app	Security guard with a clipboard
Load balancer	L4–L7	Server health / load	Restaurant host seating guests

Reading a topology diagram

A **topology diagram** is a map of how these devices connect. The exam may show you one and ask what a symbol means or where traffic flows. You do not need to memorize Cisco icons perfectly — just follow the logic: hosts connect to switches, switches connect to a router, the router connects through a firewall to the internet, and load balancers sit in front of server farms.

A tiny text topology — trace the path from PC to internet

```
PC1 ---\
         [ Switch ] --- [ Router ] --- [ Firewall ] --- Internet
PC2 ---/                    |
                       [ Load Balancer ]
                        /     |     \
                     Web1    Web2    Web3
```

Trace it: PC1 sends a frame to the switch (Layer 2). The switch hands anything off-subnet to the router (Layer 3). The router passes it through the firewall, which checks the rules, then out to the internet. Requests coming back to the web servers hit the load balancer first, which picks Web1, Web2, or Web3.

✅ **Exam Tip — Match the verb to the layer**

If a question mentions **MAC addresses**, think **switch (Layer 2)**. If it mentions **IP addresses or routing between networks**, think **router (Layer 3)**. If it mentions **allow/block or security policy**, think **firewall**. If it mentions **spreading traffic across servers**, think **load balancer**.

⚠️ **Common Pitfall — Calling a switch a router (or vice versa)**

Both forward traffic, so beginners blur them. Anchor on the layer: switch = local, MAC, Layer 2; router = between networks, IP, Layer 3. A frame staying on one subnet never needs a router; a packet leaving the subnet always does.

🌍 **From the Field — Why this matters for automation**

Every Cisco platform you will script later — Catalyst Center, Meraki, ACI — ultimately manages these same four roles. When an API returns a list of 'devices,' knowing whether each is a switch, router, or firewall tells you what you can sensibly ask it to do.

Summary

Four building blocks carry and guard traffic. A **switch** forwards frames locally by MAC (Layer 2). A **router** forwards packets between networks by IP (Layer 3) and is the default gateway. A **firewall** allows or blocks traffic by rule for security. A **load balancer** spreads requests across identical servers for capacity and resilience. A **topology diagram** maps how they connect; read it by following traffic from host → switch → router → firewall → internet, with load balancers in front of server farms.

Key Takeaways

Switch = L2, MAC, local. Router = L3, IP, between networks, = gateway. Firewall = allow/block by rule (security). Load balancer = spread load across servers. Read **topologies** by tracing the traffic path and matching each device to its role.

Knowledge Check — 1.3 Network Building Blocks

Q1. Which device forwards traffic between different networks using IP addresses?

- **A.** Switch
- **B.** Firewall
- **C.** Load balancer
- **D.** Router

Correct answer: D. A router operates at Layer 3 and forwards packets between networks based on IP. A switch is Layer 2 (MAC) and stays local.

Q2. At which layer does a traditional switch primarily operate?

- **A.** Layer 3
- **B.** Layer 2
- **C.** Layer 4
- **D.** Layer 7

Correct answer: B. A traditional switch forwards frames at Layer 2 using MAC addresses.

Q3. What is the primary purpose of a firewall?

- **A.** To assign IP addresses to clients
- **B.** To balance load across servers
- **C.** To allow or block traffic based on security rules
- **D.** To convert MAC addresses to IP addresses

Correct answer: C. A firewall enforces security policy, permitting or denying traffic by address, port, or application. It is not for addressing or load distribution.

Q4. A company runs three identical web servers and wants to spread incoming requests evenly. Which device does this?

- **A.** Load balancer
- **B.** Switch
- **C.** DNS server
- **D.** Repeater

Correct answer: A. A load balancer distributes incoming requests across multiple servers and can stop sending traffic to a failed one.

Q5. In a typical topology, what does a host send its off-subnet traffic to first?

- **A.** Directly to the internet
- **B.** The switch it is connected to, which forwards it toward the router
- **C.** The firewall, bypassing all other devices
- **D.** The load balancer

Correct answer: B. The host frames go to its local switch first; the switch hands off-subnet traffic toward the router (the gateway), which then reaches the firewall and internet.

END OF FREE SAMPLE

You've just read the opening of the book

This sample included the Chapter 1 overview and its first three sub-sections — each with everyday analogies, runnable code, exam tips, common-pitfall warnings, and a complete five-question Knowledge Check. That is the exact rhythm of all 28 sub-sections.

THE COMPLETE GUIDE GIVES YOU

- ✓ All 6 chapters — 113 pages covering every domain of the 200-901 blueprint
- ✓ 28 sub-sections, each taught from the ground up with vivid analogies
- ✓ 140 Cisco-style practice questions, every answer fully explained
- ✓ Hands-on Python, REST API, Docker, Ansible, and NETCONF / RESTCONF code
- ✓ A 48-term glossary and exam-day Summary + Key Takeaways for fast revision

Get the complete guide

Available on LeanPub and Etsy

Search: CCNA Automation 200-901 — Complete Learning Guide

by Jozef Baroš