Networking

K19G

2026-07-30

Table of contents

Networking

Personal notes on learning **computer networking from first principles to expert NetOps**, with labs built as **code** in Containerlab using **open-source and free network images** only.

> **Note**
>
> This book is **vendor-agnostic**. CLIs differ; models do not. Start with the **Syllabus** pages under Front Matter—several outlines are included so you can choose a spine. Diagrams use transparent SVGs (`currentColor`) so they stay readable in light and dark Quarto themes.

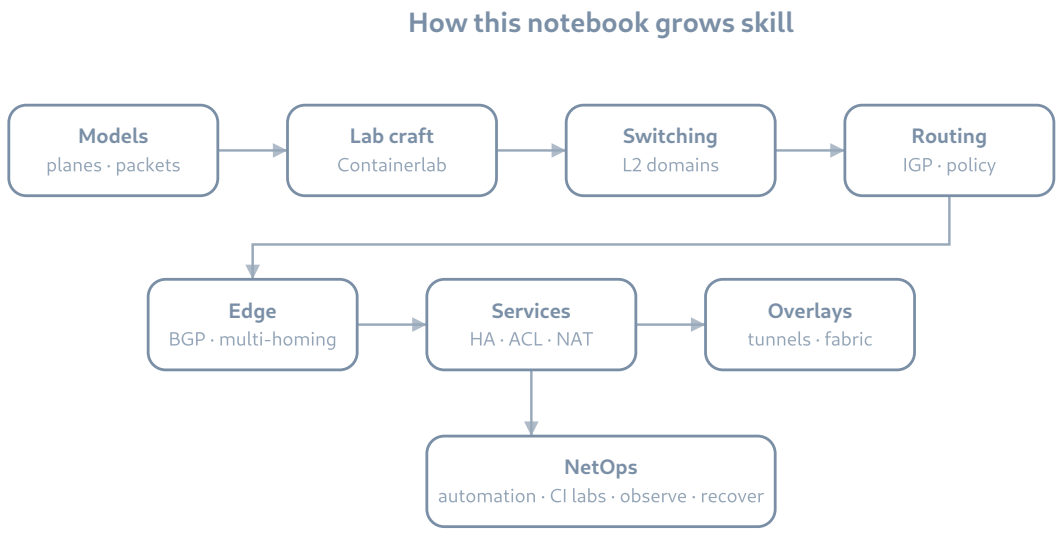


Figure 1: How skill grows in this notebook

What "done" looks like

- Design multi-site L2/L3 networks with clear underlay vs overlay roles
- Implement them in Containerlab with free images (e.g. FRR, Linux, SR Linux, SONiC, VyOS)
- Prove behavior with tables, neighbors, and packet captures—not hope
- Break links and policies on purpose; restore and document

- Automate lab lifecycle and grow into production-shaped NetOps habits

Core habits

Habit	Why
Models first	Planes, encapsulation, state machines transfer forever
Lab-as-code	Reproducible `.clab.yml` topologies in git
Predict → observe → fix → harden	How experts actually learn
Open stack	No paid image required on the main path

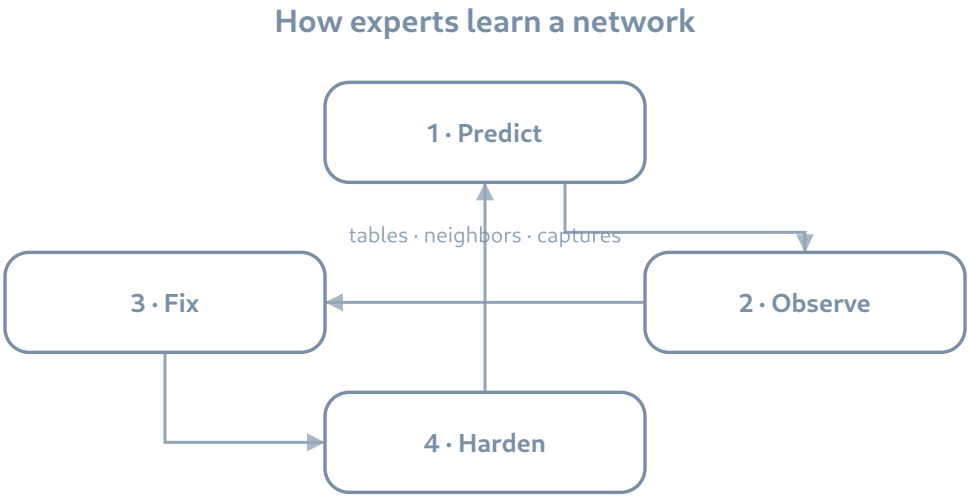


Figure 2: Verification loop

How to read the syllabi

Page	Role
Syllabus	Working outline **A** (recommended starting spine)
Syllabus 1–7	Alternate ways to organize the same destination
Syllabus options	Comparison table

Lab ground rules

- Prefer **free/open** node kinds
- Every serious lab: topology file + addressing plan + failure scenario
- Capture evidence (commands + pcap when useful)
- Plan CPU/RAM for node scale

Formats

HTML, PDF, and EPUB via the monorepo portal when built.

Front Matter

Syllabus

Networking Journey — Syllabus A (working)

Working title: Networking from first principles to NetOps

A personal notebook on how to learn computer networking **from scratch to expert / NetOps practice**, using **open-source and freely available lab images** (especially via Containerlab).

No vendor certification branding. No proprietary exam blueprints. Competence is defined by **what you can design, build, break, prove, and operate**.

Lab platform: Containerlab + free/open network OS and endpoint images (e.g. FRR, Linux, Nokia SR Linux community images, SONiC, VyOS, and other kinds that do not require paid licenses).

Diagrams: under `images/diagram-*.svg` — transparent SVG using `currentColor` so they work in Quarto light and dark themes.

---

0. How a world-class path is taught

0.1 Pedagogy

1. **Models before features** — planes, encapsulation, state machines
2. **Predict → observe → fix → harden** — every lab
3. **Lab-as-code** — topologies in git; rebuild from nothing
4. **Failure is curriculum** — shut links, poison routes, break MTU
5. **Vendor-agnostic language** — concepts first; CLI is a dialect
6. **Open tools only** on the main path — paid images are optional footnotes later, never required

0.2 Definition of "expert" here

You can:

- Design a multi-site L2/L3 network with clear underlay/overlay roles
- Implement it in Containerlab with free images

- Diagnose control-plane vs data-plane failures with tables + captures
- Apply routing policy without creating loops
- Automate lab lifecycle and basic config/validation
- Operate services with observability and recovery habits

0.3 Open lab stack (default)

Role	Examples (free/open)
Orchestration	Containerlab
Routing NOS	FRR, SR Linux (community), SONiC, VyOS
Endpoints	Alpine/Ubuntu/Debian containers
Capture	tcpdump, tshark
Traffic	iperf3, mtr, scapy (as needed)
Automation	Ansible, Python (scrapli/netmiko-class tools), git
CI	deploy topology smoke tests

Journey map

```
Models + lab craft
  → Host networking for labs
  → Bridging domains (L2)
  → Addressing & forwarding (L3)
  → Interior routing
  → Edge routing & policy
  → Resilience & services
  → Filtering, QoS intro, hardening
  → Overlays & fabrics
  → Automation & NetOps
  → Capstones
```

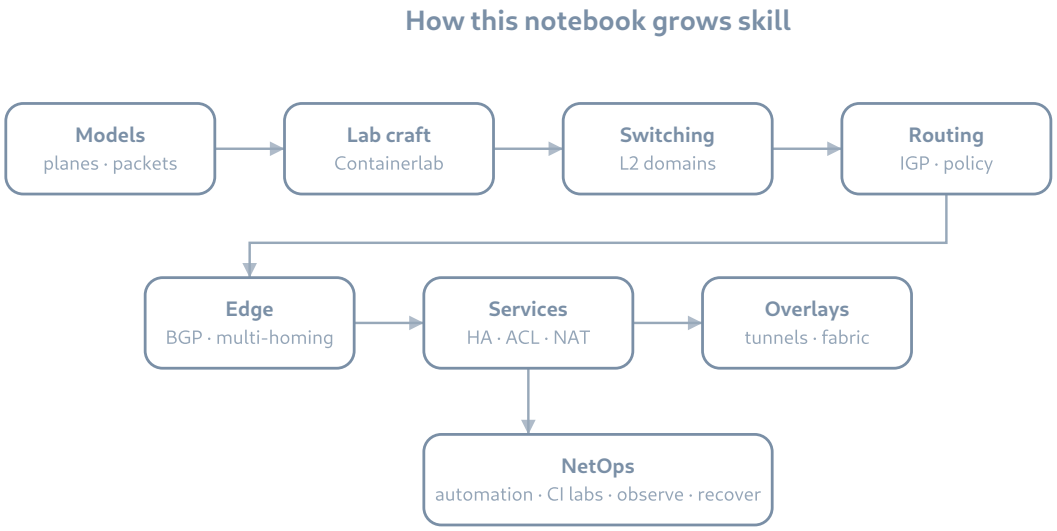


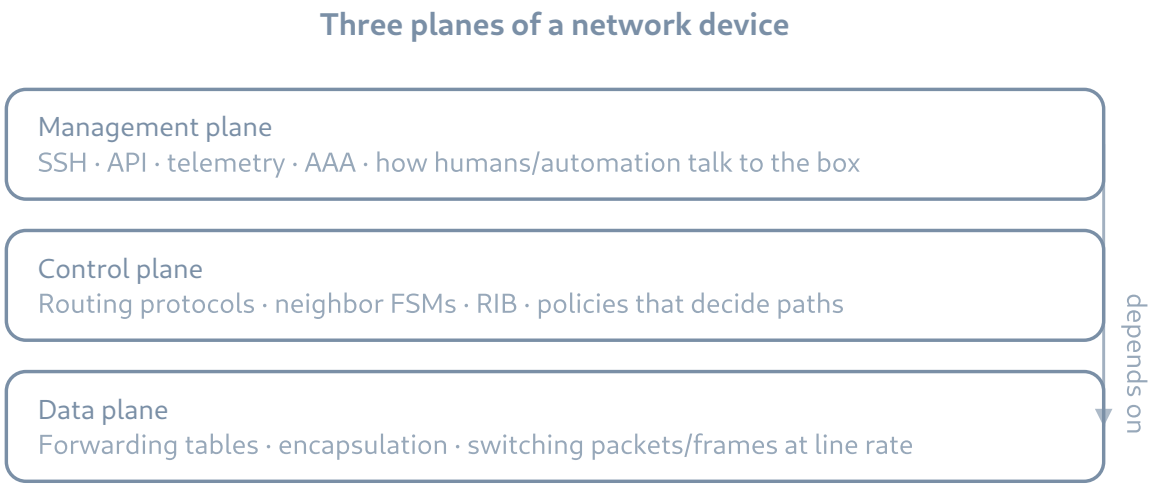
Figure 1: Learning path

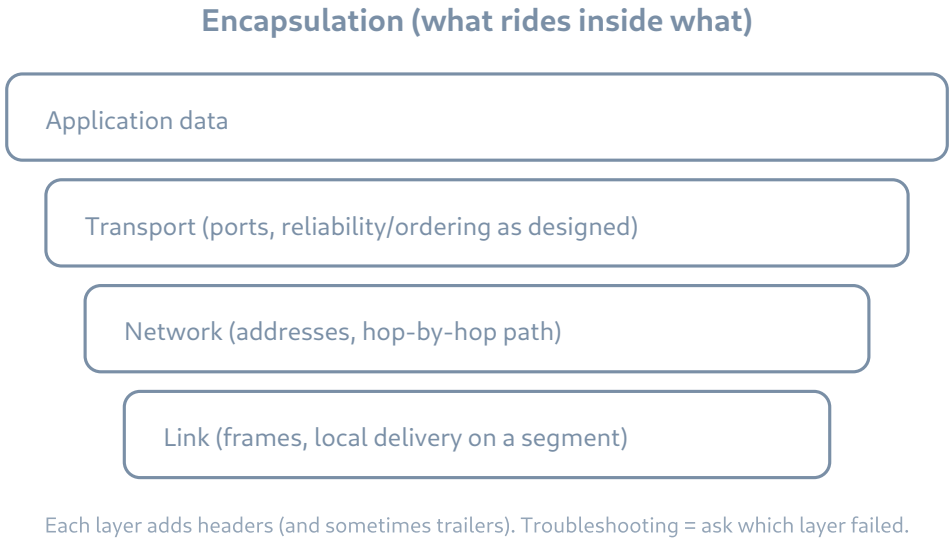
Part 00 — Front matter

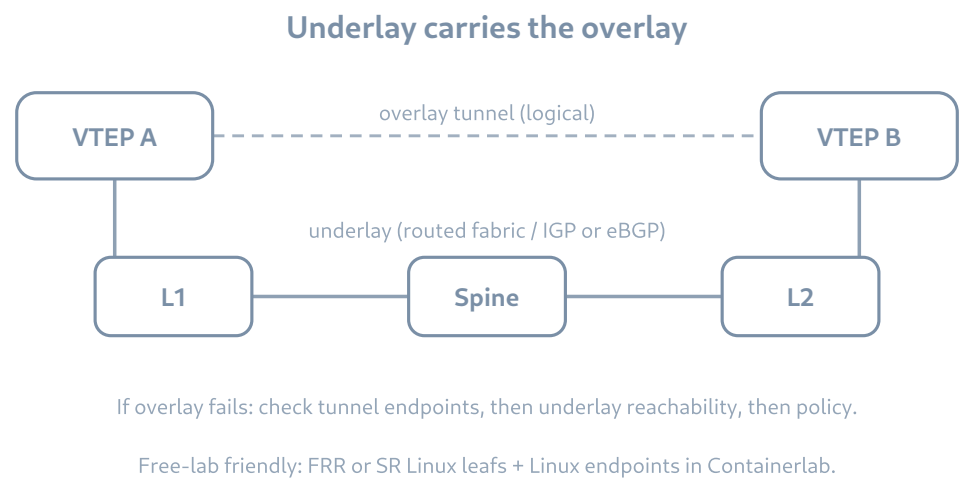
Ch 00 — Syllabus (this document + variants)

Ch 01 — How I learn networking

- Note-taking, lab journal, when to stop and rebuild
  ### Ch 02 — Mental models that never expire
- Planes, encapsulation, underlay/overlay







Ch 03 — Verification culture

- Predict → observe → fix → harden

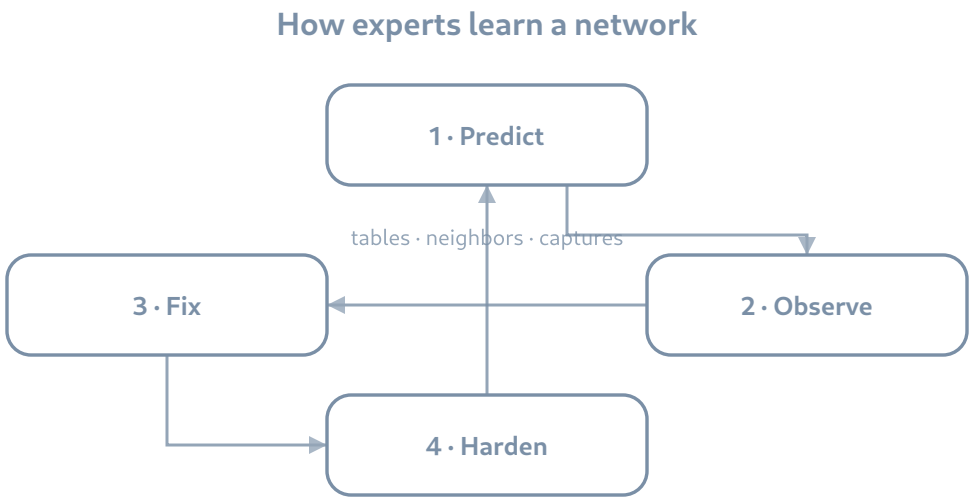


Figure 1: Verify loop

Part 01 — Lab craft (Containerlab + Linux)

Ch 01 — Linux networking for lab hosts

- namespaces, veth, bridges, routes, basic filtering
  ### Ch 02 — Containerlab fundamentals
- topology YAML, kinds, links, mgmt network, lifecycle
  ### Ch 03 — Free image matrix & resources
- what runs without licenses; RAM/CPU planning
  ### Ch 04 — Lab hygiene
- addressing plans, naming, git layout, diagram habit

Lab: deploy/destroy a triangle of open routers; export a graph.

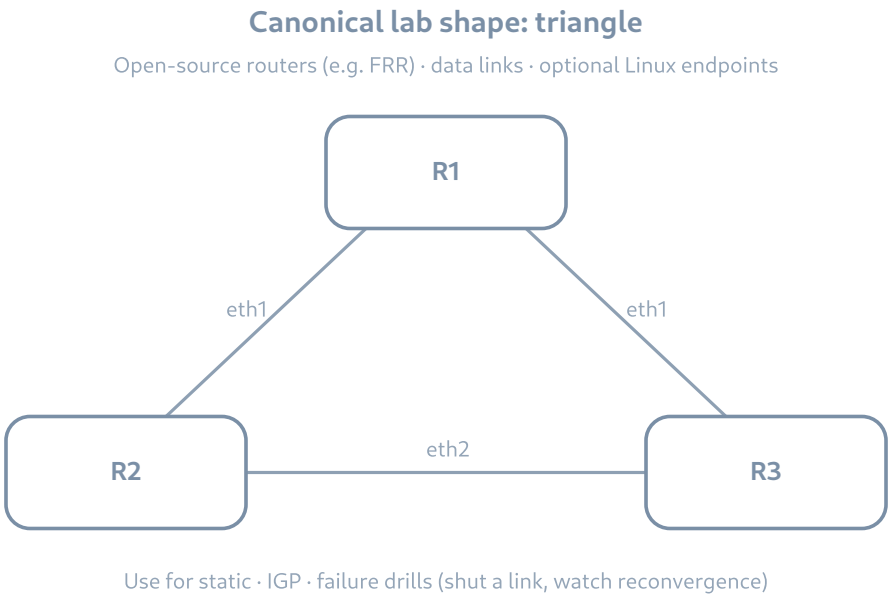


Figure 1: Triangle lab

Checkpoint: topology-as-code is muscle memory.

Part 02 — Bridging domains (layer-2)

Ch 01 — Ethernet & MAC learning

Ch 02 — VLANs & trunks

Ch 03 — Loop prevention (spanning tree family)

Ch 04 — Link aggregation

Ch 05 — L2 failure drills

Labs: multi-switch VLAN/trunk; root failover; native-VLAN mismatch.

Checkpoint: you can heal a broken multi-switch domain with evidence.

Part 03 — Addressing & basic forwarding

Ch 01 — IPv4 structure, subnetting, summarization

Ch 02 — IPv6 essentials for dual-stack labs

Ch 03 — Static routing & floating statics

Ch 04 — Host vs router behavior (ARP/ND)

Labs: dual-stack; summary black-hole; static failover.

Part 04 — Interior routing

Ch 01 — Dynamic routing ideas (distance-vector vs link-state)

Ch 02 — Link-state IGP in one area (adjacency, flood, SPF)

Ch 03 — Multi-area link-state design

Ch 04 — IPv6 IGP notes

Ch 05 — Redistribution & loop prevention

Labs: triangle IGP; cost change; multi-area; filtered redistribution.

Checkpoint: reconvergence is predictable and documented.

Part 05 — Edge routing & policy

Ch 01 — Inter-domain routing fundamentals

Ch 02 — Session types & path selection mental model

Ch 03 — Policy (prefix filters, attributes, route-maps as idea)

Ch 04 — Scaling internal meshes (route reflection concepts)

Ch 05 — Dual-homing & traffic engineering

Labs: dual upstream; policy steer; session loss drill.

Part 06 — Resilience & network services

Ch 01 — First-hop redundancy concepts

Ch 02 — Address assignment & relay

Ch 03 — Translation at edges (NAT family)

Ch 04 — Time, name, and log hygiene in labs

Labs: active/standby first hop; NAT edge; DHCP relay.

Part 07 — Policy enforcement, QoS intro, hardening

Ch 01 — Packet filters (ACL thinking, direction, implicit deny)

Ch 02 — Protecting the control plane (concepts)

Ch 03 — QoS vocabulary (classify, mark, queue, shape/police)

Ch 04 — Device & lab hardening checklist

Labs: filter lockout/recovery; simple mark/class demo.

Part 08 — Overlays, tunnels, multicast

Ch 01 — Tunneling ideas (why encapsulate)

Ch 02 — Site connectivity patterns

Ch 03 — DC overlay intro (VXLAN-class data plane)

Ch 04 — EVPN-class control plane (conceptual + lab if free images allow)

Ch 05 — Multicast essentials

Labs: point-to-point tunnel; mini overlay fabric with open NOS.

Part 09 — Fabrics & multi-node design

Ch 01 — Clos / leaf-spine

Ch 02 — Underlay design choices

Ch 03 — Multi-implementation labs (FRR + SR Linux + Linux endpoints)

Labs: small leaf-spine; break a leaf; watch convergence.

Part 10 — NetOps, automation, observability

Ch 01 — Config as code against lab nodes

Ch 02 — Containerlab in CI (smoke deploy)

Ch 03 — Telemetry & captures as operations

Ch 04 — Runbooks & incident notes

Labs: Ansible/Python push; CI deploy; intentional outage write-up.

Part 11 — Capstones

C1 — Campus-style dual core (open images)

C2 — Dual-homed site to “provider” edge

C3 — Mini fabric with overlay

C4 — Incident week: break, detect, restore, prevent

Part 99 — Appendices

A — Addressing plan templates

B — Free image matrix & host sizing

C — Verification cheatsheet (vendor-neutral show/intent)

D — Diagram library index

E — Resources (RFCs, Containerlab docs, free labs, manuals)

F — Deliberate exclusions (wireless deep-dive, paid-only NOS, exam cram)

Writing order (when content starts)

1. Models + Containerlab craft + diagrams
2. L2 + single-area IGP (highest ROI)
3. Edge routing dual-home
4. Services & filters
5. Multi-area / policy depth
6. Overlay/fabric
7. NetOps/CI

Changelog

Date	Note
2026-07-24	Initial multi-syllabus exploration
2026-07-24	Vendor-agnostic rewrite; open images only; expert pedagogy; diagrams

Syllabus 1 — Expert practitioner

Syllabus 1 — Expert practitioner (spiral)

Vendor-agnostic. Open lab images only. Destination: expert NetOps competence.

Thesis

Experts debug **planes**, **encapsulation**, and **state**—not brand menus. Spiral depth beats feature bingo.

Arcs

0. Models (planes, encaps, underlay/overlay) + verification culture
1. Lab craft (Containerlab + Linux networking)
2. Packets & tables (MAC, ARP/ND, RIB vs forwarding)
3. Bridging domains until boring
4. Interior routing truth (adjacency, flood/SPF, multi-area)
5. Edge routing & policy (sessions, path choice, dual-homing)
6. Resilience & services (FHRP-class, NAT, relays)
7. Filters, QoS vocabulary, hardening
8. Overlays & fabrics
9. NetOps (automation, CI labs, observe, runbooks)

Definition of done

Broken multi-protocol lab → restore with evidence → prevent recurrence → all in free Containerlab images.

Syllabus 2 — Project-driven milestones

Syllabus 2 — Project-driven milestones

Open images · Containerlab · theory only when blocked.

M	Build	Theory pulled
M0	Host ready; first deploy/destroy	Linux net, images
M1	Triangle routers, static + pcap	IP, static
M2	Multi-switch VLANs/trunks	Bridging, tags
M3	Loop-prevention failover	Spanning tree family
M4	IGP single area	Link-state basics
M5	Multi-area IGP	Hierarchy, filtering
M6	Dual-homed edge routing	Inter-domain, policy intro
M7	First-hop HA + relay	Resilience, services
M8	Filters + edge translation	ACL thinking, NAT
M9	Site tunnel	Overlay basics
M10	Mini leaf-spine	Fabric underlay
M11	Overlay fabric taste	VXLAN/EVPN-class ideas
M12	Automate config against lab	NetOps scripting
M13	CI smoke deploy	Lab-as-code
M14	Capstone multi-site	Integration

Each M: `.clab.yml` · address plan · expected state · break · recovery · diagram.

Syllabus 3 — Tools & lab stack

Syllabus 3 — Open tools & lab stack

Must-learn platform

Containerlab · Docker/containerd · Linux bridges/veth · tcpdump/tshark · iperf3/mtr

Free/open node kinds (main path)

FRR · Linux endpoints · Nokia SR Linux (community) · SONiC · VyOS · other libre kinds as useful

Docs & design

git · addressing plans · diagram-as-code · this Quarto book

Automation & CI

Ansible / Nornir / Python · CI runners deploying `.clab.yml` · optional gNMI

NetOps-adjacent (later)

NetBox-class SoT · metrics/logs stacks · backup/restore tools · analysis tools (Batfish/Suzieq awareness)

Order through tools

1) Containerlab+FRR+Linux → 2) captures → 3) L2 kinds → 4) IGP/edge on FRR → 5) multi-impl (SRL/SONiC) → 6) automation → 7) CI → 8) homelab services as modules of knowledge

Syllabus 4 — Community path

Syllabus 4 — Community-shaped path

What practitioners repeat

Lab daily · start small · capture packets · one platform deep · free stacks teach most ideas · finish a path · document failures · routing policy is the professional gate · automate only after manual competence

Phases

1. Containerlab comfortable
2. Bridging mini-campus
3. IP + single-area IGP
4. Services & filters
5. Self-check: design, build, break, recover
6. Multi-area + edge routing + policy
7. Tunnels / fabric taste
8. Automation + CI
9. Portfolio of 3 polished labs

Footguns

Too many images early · no address plan · skipping L2 · edge routing before IGP · automation without reading tables · never failing the lab on purpose

Syllabus 5 — Containerlab-first

Syllabus 5 — Containerlab-first

Thesis

If labs are the product, master **topology patterns** first; attach protocols as patterns evolve.

Blocks

B1 Platform: kinds, links, mgmt, binds, graph, troubleshoot deploy
B2 Pattern library: 2-node, triangle, square, dual-home, leaf-spine, multi-AS
B3 Protocols attached in order: L2 → static/IGP → edge routing → services/filters → overlay
B4 Chaos: link down, delay/loss, process restart
B5 Automation & CI smoke
B6 Map patterns to real campus/DC/WAN (still vendor-neutral)

Open images only on the main path.

Syllabus 6 — Edge & fabric emphasis

Syllabus 6 — Edge & fabric emphasis

Foundations still required (models, L2, IP, single-area IGP)—then tilt depth toward:

1. Multi-area interior routing & filtering
2. Edge routing policy & dual-homing
3. Tunneling & secure site connect
4. Clos fabrics, underlay design
5. Overlay data/control planes (VXLAN/EVPN-class) with free NOS
6. Multicast essentials
7. NetOps for multi-node labs

Capstone: “provider edge + fabric + site” reduced topology, all open images.

Syllabus 7 — Topic coverage map

Syllabus 7 — Coverage map (checklist)

Use to guarantee long-term completeness. Mark: not started / conceptual / labbed / operated.

Concepts

Planes · encapsulation · underlay/overlay · state machines · RIB vs FIB · MTU · anycast · ECMP

Bridging

MAC learning · VLANs/trunks · STP family · LAG · MLAG concepts · L2 loops/drills

Addressing

IPv4/IPv6 · VLSM · summarization · ND/ARP · dual-stack

Interior routing

Link-state single/multi-area · DV awareness · redistribution · filtering · IPv6 IGP

Edge routing

Sessions · path selection · policy · scaling meshes · dual-homing · communities-class ideas

Services & HA

FHRP-class · DHCP/relay · NAT · NTP/DNS/logging hygiene

Security & QoS

Filters · control-plane protection concepts · classify/mark/queue · hardening

Overlays & DC

GRE-class tunnels · VXLAN-class · EVPN-class · leaf-spine · multi-impl labs

Multicast

ASM/SSM ideas · PIM sparse essentials

Lab & NetOps

Containerlab · free kinds · captures · traffic tools · Ansible/Python · CI · telemetry · runbooks · SoT awareness

Exclusions (unless later promoted)

Paid-only NOS as requirement · exam cram · deep wireless · full optical transport

Syllabus options — comparison

Syllabus options — comparison

Shared destination: vendor-agnostic networking expert / NetOps, **open lab images**, Containerlab.

ID	Chapter title	Organizing idea
A	Syllabus	First principles → L2 → L3 → edge → overlay → NetOps
1	Expert practitioner	Spiral models + failure-first
2	Project-driven	Milestones M0–M14
3	Tools & lab stack	Free tool inventory
4	Community path	Practitioner consensus
5	Containerlab-first	Topology patterns first
6	Edge & fabric	BGP/DC-shaped depth
7	Coverage map	Exhaustive checklist

How to choose: pick a spine (A/1/2/4/5/6); use **3** and **7** so personal + production open tooling is not forgotten.

Diagrams (theme-friendly): `images/diagram-*.svg` (transparent, `currentColor`).