Purple-Team Cybersecurity Roadmap — Complete (Beginner → FAANG-ready Expert)

Awesome — below is a **complete, beginner-friendly, step-by-step roadmap** to become a **purple-team software / cybersecurity engineer** capable of cracking FAANG interviews. First I'll list the **main skill-areas** you must master. After that I'll **explain each area topic-wise** (what to learn, why it matters, tools, practical exercises, and how to demonstrate it). Follow this as a checklist and build projects & evidence for each item.

🔑 Top-level list — main things you must master

1. **Computer and OS fundamentals** (Linux + Windows internals)
2. **Networking & Protocols** (TCP/IP stack, DNS, HTTP/S, TLS)
3. **Programming & Scripting** (Python + Go + Bash; C/C++ basics; JavaScript)
4. **Security fundamentals & theory** (CIA triad, risk, threat modeling)
5. **Offensive security (Red team basics)** — pentesting concepts
6. **Defensive security (Blue team basics)** — detection, monitoring, incident response
7. **Detection engineering & threat hunting** (SIEM, EDR, analytics)
8. **Malware analysis & reverse engineering (basics → intermediate)**
9. **Cloud security** (AWS/GCP/Azure fundamentals + cloud-native threats)
10. **Identity & access management (IAM), SSO, PKI**
11. **Application + DevSecOps security** (secure coding, SCA, IaC, container security)
12. **Forensics & incident response (IR)**
13. **Log pipelines & observability** (ELK, Splunk, Grafana, Prometheus)
14. **Vulnerability management & exploit awareness**
15. **Threat intelligence & frameworks (MITRE ATT&CK, kill chain)**
16. **Automation, tool development & engineering** (build detection tools, playbooks)
17. **Soft skills & process** (communication, runbooks, purple-team exercises)
18. **Certifications & credible deliverables** (projects, open source, blog posts)
19. **Interview & system design prep for FAANG** (security design, threat modeling, case studies)

Topic-wise detailed breakdown (Beginner → FAANG-ready)

I break each topic into: **What to learn**, **Why it matters**, **Tools & tech**, **Hands-on projects / exercises**, and **How to demonstrate mastery**.

1) Computer and OS fundamentals (Linux + Windows internals)

What to learn

- Linux basics: filesystems, permissions, process management, systemd, networking commands, package managers.
- Windows internals: registry, services, event logs, PowerShell, Sysinternals (ProcMon, Autoruns).
- System architecture: processes, threads, memory, kernel vs user space.

Why it matters

- Purple teamers must operate on both attacker and defender sides; deep OS knowledge lets you craft attacks and design effective detections.

Tools & tech

- Linux (Ubuntu, Kali), Windows 10/11/Server, WSL2, Sysinternals, PowerShell.

Hands-on

- Set up a home lab: one Linux VM, one Windows VM. Practice installing services, creating users, setting file ACLs.
- Use strace, top, ps, netstat/ss, journalctl on Linux; use Event Viewer + Sysinternals on Windows.

Demonstrate

- Git repo with small scripts automating Linux/Windows tasks, blog post explaining Windows event log anatomy, and a short lab write-up.

2) Networking & Protocols

What to learn

- TCP/IP fundamentals, UDP, ARP, DNS, DHCP.
- HTTP/HTTPS internals, TLS handshake, headers, cookies.
- Subnets, VLANs, NAT, routing basics.

Why it matters

- Network behavior is a major source of telemetry for detection; attacks often abuse protocols.

Tools

- Wireshark, tcpdump, nmap, netcat, curl, ss, iptables.

Hands-on

- Capture packets with Wireshark and interpret TCP three-way handshakes, HTTP requests, TLS negotiation.
- Scan & enumerate a test network with nmap. Simulate DNS tunneling and observe traffic patterns (in controlled lab).

Demonstrate

- Packet captures and analyses on GitHub, breakdown post of detecting suspicious DNS patterns.

3) Programming & Scripting (core)

What to learn

- **Python** (primary): scripting, network libraries, parsing logs, building detection prototypes, API integration.
- **Go** (recommended): build performant tools, agents, and network utilities.
- **Bash / PowerShell**: automation and admin tasks.
- **C/C++ basics**: memory model, pointers, for exploit awareness & reverse engineering.
- **JavaScript/Node.js**: web app security understanding.

Why it matters

- Purple team role demands fast tooling, automation, parser & detection rule authoring, and occasionally exploit development knowledge.

Tools

- Python (requests, scapy, pandas), Go toolchain, pip, virtualenv, PowerShell Core.

Hands-on

- Build a Python script to parse web server logs and flag anomalous requests.
- Implement a simple TCP port scanner in Go.
- Create PowerShell scripts to collect Windows event logs.

Demonstrate

- GitHub repo with practical scripts, unit tests, README and usage examples.

4) Security fundamentals & theory

What to learn

- CIA triad, threat modeling, risk assessment, OWASP Top 10, secure design principles (least privilege, defense-in-depth).

Why it matters

- Provides mental models to evaluate tradeoffs, design controls, and communicate risks to stakeholders.

Hands-on

- Do threat modeling on a sample web app (data flows, attack surface, mitigations).

Demonstrate

- A threat model document and mitigation plan for a sample system on your portfolio.

5) Offensive security (Red team basics)

What to learn

- Reconnaissance, scanning, exploitation basics (non-destructive), lateral movement concepts, persistence, C2 basics conceptually (no malicious use).
- Web app pentesting basics: SQLi, XSS, auth flaws (ethically, in lab).

Why it matters

- Purple teams need to know attack techniques to craft detections and test defenses.

Tools

- Kali tools: nmap, metasploit (conceptual use), burpsuite (intercepting proxy), sqlmap (lab), Responder (lab), BloodHound (AD enumeration).

Hands-on

- Enroll in intentionally vulnerable environments: OWASP Juice Shop, DVWA, WebGoat, TryHackMe labs, HackTheBox (learn legally).

Demonstrate

- Write ethical pentest summaries of your lab findings (no exploits shown), mitigation suggestions, and detection rules you created in response.

⚠️ Ethical note: Always practice on legal, controlled environments and never attack real systems without written permission.

6) Defensive security (Blue team basics)

What to learn

- Log sources, telemetry mapping, detection types (signature, heuristic, behavior).
- EDR basics (what it collects), alert triage, SOC workflows (Triage → Investigate → Contain → Remediate).

Tools

- Elastic stack (ELK), Splunk, OSSEC/Wazuh, Windows Event Logs, Sysmon, Velociraptor, OSQuery, Carbon Black, CrowdStrike (conceptual).

Hands-on

- Install Sysmon on Windows, centralize logs to ELK, write Kibana queries and dashboards.
- Create simple detection rules (e.g., unusual PowerShell child process spawn).

Demonstrate

- Dashboard screenshots, detection POC, playbook for triage of a simulated alert.

7) Detection engineering & threat hunting

What to learn

- MITRE ATT&CK mapping, analytics development, baseline noise reduction, false positive management, statistical detection vs rules.
- Query languages: KQL (in Azure), Splunk SPL, Elasticsearch DSL.

Why it matters

- Purple teamers design detections and validate them.

Hands-on

- Map simulated red-team actions to ATT&CK and author detections for each.
- Run hunts: pivot through logs to find stealthy behavior (e.g., credential dumping indicators).

Demonstrate

- A portfolio collection of ATT&CK-mapped detections, tuning notes, and a retrospective showing FP reduction.

8) Malware analysis & reverse engineering

What to learn

- Static analysis basics, dynamic analysis in sandbox, assembly reading basics, reversing simple binaries.
- Tools: IDA/Ghidra, x64dbg, strings, radare2.

Hands-on

- Reverse small intentionally vulnerable binaries, sandbox behavior in a VM, extract IoCs.

Demonstrate

- Malware analysis write-up (safe/sanitized), indicators of compromise, proposed detection logic.

9) Cloud security (AWS/GCP/Azure)

What to learn

- Cloud fundamentals, IAM, VPCs, security groups, cloud logging (CloudTrail, Stackdriver), serverless pitfalls, container orchestration security (Kubernetes).
- Cloud misconfigurations and remediation.

Tools

- AWS Console, Azure Portal, GCP console, Terraform, kubectl, kube-bench, kube-hunter.

Hands-on

- Create a cloud lab account (free tier), deploy sample app, enable logging, intentionally misconfigure an S3 bucket (locally) and detect access. Map cloud events to ATT&CK.

Demonstrate

- Cloud sec checklist, remediation steps, Terraform configs, and detection playbooks.

10) Identity & Access Management (IAM), SSO, PKI

What to learn

- Principals, roles, policies, OIDC/OAuth2/SAML, MFA, certificate basics, token lifetimes.

Why it matters

- Identity is the primary attack vector in modern infra.

Hands-on

- Simulate weak IAM policies in cloud lab, create an identity-based detection for privilege escalation.

Demonstrate

- Documented IAM review and remediation plan, short write-up on secure token usage.

11) Application + DevSecOps security

What to learn

- Secure SDLC, SAST/DAST, dependency scanning (SCA), CI/CD pipeline security, IaC security, container image scanning.

Tools

- GitHub Actions/GitLab CI, SonarQube, Dependabot, Trivy, Snyk, Checkov.

Hands-on

- Integrate SCA and DAST tools into a CI pipeline and demonstrate a blocked insecure build. Secure a Dockerfile and scan images.

Demonstrate

- CI pipeline examples, screenshots, PRs showing remediation of vulnerabilities.

12) Forensics & Incident Response (IR)

What to learn

- Evidence collection, timeline building, memory forensics, disk imaging basics, chain of custody (for enterprise IR).
- Triage steps, containment strategies, and post-incident reporting.

Tools

- Volatility, Autopsy, FTK Imager, Velociraptor.

Hands-on

- Simulate an incident in lab: collect logs, build a timeline, identify root cause, and prepare an incident report.

Demonstrate

- Incident report template, IR playbook, and a postmortem write-up.

13) Log pipelines & observability

What to learn

- Log ingestion, normalization, retention, enrichment, tagging, indexing and querying. Monitor metrics, traces, and logs.

Tools

- ELK stack, Splunk, Grafana, Prometheus, Loki.

Hands-on

- Build log pipeline from Linux/Windows hosts to ELK, create dashboards and alerts for suspicious activity.

Demonstrate

- Live dashboards or screenshots; explain chosen KPIs and alerts.

14) Vulnerability management & exploit awareness

What to learn

- CVE lifecycle, asset inventory, prioritization (CVSS + context), patching strategies, and exploitability vs impact.

Hands-on

- Run a vulnerability scanner on a lab network, classify findings, and write remediation tickets.

Demonstrate

- VM-scanning reports, remediation tracking, and improvement metrics.

15) Threat intelligence & frameworks (MITRE ATT&CK)

What to learn

- ATT&CK matrix mapping, TTPs, IOC vs TTP, threat actor profiling.

Hands-on

- Map a simulated intrusion to ATT&CK and produce an intelligence brief.

Demonstrate

- ATT&CK mapping artifacts and threat brief for leadership.

16) Automation, tool development & engineering

What to learn

- Build internal tooling and automation (playbooks, SOAR, scripted detection deployments). Learn REST APIs, multi-threading, logging, and testing.

Tools

- Python, Go, Elastic API, Splunk REST API, SOAR platforms (Palo Alto Cortex XSOAR, Demisto conceptual).

Hands-on

- Build a detection deployment script that takes a YARA/Sigma rule and deploys to your test environment.

Demonstrate

- GitHub repo with tools, unit tests, usage docs.

17) Soft skills & process (critical for FAANG)

What to learn

- Communicating technical analysis to non-technical stakeholders, writing runbooks, presenting postmortems, running purple-team exercises, mentoring.

Hands-on

- Lead a simulated tabletop exercise, present findings to a mock leadership audience.

Demonstrate

- Slide decks of past tabletop or purple exercises; crisp incident postmortem.

18) Certifications & credible deliverables

Certs to consider (optional but valued)

- Entry: eJPT (Pentesting), CompTIA Security+ (foundational).
- Mid: OSCP (offensive credibility), GCIA/GCFA/GCTI (SANS/GIAC) for detection/forensics.
- Senior: CISSP (broad security mgmt), AWS/GCP/Azure Security certs.

Deliverables

- GitHub projects (tools, detections), blog posts, writeups, public dashboards, contributions to open-source security projects.

19) FAANG Interview & system design prep for security roles

What to learn

- System-design with security constraints, threat modeling for large systems, detection architecture at scale, measurable SLAs for detection/MTTR.
- Behavioral interviewing: STAR method, stories of impact.

Interview focus areas

- Design a scalable detection pipeline (ingestion → enrichment → indexing → detection → alerting → SOAR).
- Given a threat scenario, propose instrumentation + metrics to detect it.
- Explain tradeoffs in detection sensitivity, retention costs, and privacy.

Prepare

- Mock system design sessions, whiteboard threat models, and practice clear 10-minute walk-throughs of your security projects.

Demonstrate

- Portfolio with architecture diagrams, threat model docs, and playbooks you authored.

Practical roadmap order (how to progress logically)

1. **Foundations:** Computer/OS fundamentals + basic networking + Python + security fundamentals.
2. **Blue team basics:** Logging, Sysmon, ELK, simple detections.
3. **Red team basics:** Web app pentesting labs, recon, safe environment.

4. **Detection engineering:** ATT&CK mapping, SIEM queries, hunts.
5. **Cloud & DevSecOps:** Secure CI/CD, container security, cloud logging.
6. **Forensics & malware analysis:** Add depth for incident response.
7. **Tooling & automation:** Build detection and automation tools in Python/Go.
8. **Scale & architecture:** Learn to design enterprise detection ecosystems and prepare FAANG-level system design.
9. **Polish:** Real labs, open source contributions, public writeups, mock interviews, soft skills.

Key projects to build for your portfolio (must-have)

- **Detection Pack:** 10+ detection rules mapped to MITRE ATT&CK with test cases (GitHub).
- **SIEM Lab:** ELK / Splunk lab with dashboards and alert pipeline.
- **Threat-Hunt Case Study:** From detection → triage → containment for a simulated intrusion.
- **Tooling:** Python/Go tools (log parsers, lightweight agent, SIEM integrator).
- **Cloud Security Project:** Demonstrated cloud misconfig remediation and cloud log detection.
- **Purple Team Report:** Run a purple-team exercise (simulate attacks, tune detections) and publish the results + improvements.

Best programming languages for Purple-Team / FAANG security

1. **Python** — primary: rapid prototyping, data parsing, analytics, detection rule generation.
2. **Go** — performant tooling, agents, network tools; excellent for production tools.
3. **Bash / PowerShell** — essential for automation and Windows-specific ops.
4. **C / C++ (basics)** — understand memory corruption and low-level malware behavior.
5. **JavaScript / Node.js** — web security and serverless understanding.

(Recommendation: be excellent at Python and at least competent in Go and PowerShell/Bash.)

Learning resources (types — use legally)

- Interactive labs: TryHackMe, HackTheBox, OWASP Juice Shop (for web), RangeForce, SANS NetWars (if accessible).
- Books & reading: defensive security books, incident response, Linux & Windows internals.
- Blogs & communities: vendor blogs (CrowdStrike, Microsoft Security), Mandiant, ThreatIntel feeds.
- Open source: Sigma rules, YARA, MITRE ATT&CK resources.
- YouTube / conference talks: DEFCON, Black Hat, RSA (for research & current trends).

How to prepare specifically for FAANG interviews

- **Show scale mindset:** discuss telemetry at petabyte/tenant scale, cost tradeoffs (storage vs retention), and distributed ingestion.
- **Architecture + code:** be able to whiteboard an end-to-end detection pipeline and show real code for components (parsers, enrichment).
- **Data structures & algorithms:** strong fundamentals — many FAANG security interviews expect problem solving ability.
- **Behavioral:** STAR stories showing impact, cross-team collaboration, leadership in incident handling.
- **Open source evidence:** public repos, blog posts, presentations — FAANG recruiters value demonstrable impact.

Daily / weekly habits to level-up

- Do 30–60 minutes of labs/challenges (TryHackMe / HTB).
- Weekly: one medium project (detection rule, script, blog post).
- Participate in Capture-the-Flag (CTF) and purple-team communities.
- Read one new security research or vendor blog each week.

Ethics & legal considerations

- Always practice on authorized or intentionally vulnerable environments.
- Never share exploit code for active zero-days. Focus on detection, mitigation, and defensive knowledge.

Final checklist — what to have ready before applying to FAANG

- Solid projects: SIEM lab, detection pack, cloud security case.
- Public GitHub with well-documented code + tests.
- 2–3 technical writeups or blog posts explaining your work.
- Certifications and contribution evidence (optional but helpful).
- Strong interview prep: system design + security case studies + DSA practice.

If you want, I can now:

- Turn this into a **study schedule** (Beginner / Intermediate / Advanced stages) with weekly milestones and concrete lab tasks — or
- Generate **template projects** (a starter SIEM lab, sample detection rule repository, or a Python detection tool scaffold) and the exact commands/files to run locally.

Which of those do you want next? (I'll produce it immediately.)