What foundational IT topics should I master before starting DevOps

Short answer: master operating systems, networking, programming/scripting, version control, basic security, and cloud fundamentals before starting DevOps. These foundations make CI/CD, IaC, and troubleshooting much easier.[1] [2]

Operating systems

- Linux essentials: filesystem layout, permissions, users/groups, processes, systemd services, logs with journalctl, and shell navigation/SSH usage. These skills underpin every deployment and outage fix.[2] [1]
- Resource basics: CPU, memory, disk, and I/O; know how to check utilization and diagnose common performance issues quickly.[1] [2]

Networking

- IP addressing and subnets, routing, TCP vs UDP, and common ports/protocols like HTTP/HTTPS and SSH to understand how apps communicate.[2] [1]
- Practical tools: ip, ss, ping, traceroute, and dig for DNS; these are first-line tools during connectivity and DNS incidents.[1] [2]

Programming and scripting

- Bash for automation: loops, functions, exit codes, and text processing (grep, awk, sed) to glue systems together.[2] [1]
- Python fundamentals: CLI with argparse, requests for REST calls, JSON/YAML handling, and basic unit testing to build reliable utilities.[1] [2]

Version control and collaboration

- Git proficiency: branching, pull requests, code reviews, resolving conflicts, and tags/releases to manage changes safely and traceably.[3] [1]
- Commit hygiene: small, descriptive commits and conventional messages that feed into CI/CD pipelines clearly.[3] [1]

Web and APIs

- HTTP model: methods, status codes, headers, and cookies; how REST endpoints are designed and tested. This is essential for CI test stages and health checks.[2] [1]
- Basic API testing: Postman collections and writing a simple automated check to validate services continuously.[1] [2]

Containers

- Docker concepts: images vs containers, Dockerfiles, multi-stage builds, environment variables, volumes, and health checks for portable deployments.[4] [3]
- Local workflows: build, tag, run, exec, and logs to debug containerized apps before pushing to registries.[4] [3]

Security fundamentals

- Principles: least privilege, secrets handling, key management, and patching to reduce attack surface from day one.[3] [2]
- Network security: firewalls/security groups basics and how to reason about inbound vs outbound rules safely.[5] [2]

Cloud fundamentals

- Core building blocks: compute, storage, networking, identity, regions/AZs, and pricing levers so designs are realistic and cost-aware.[6] [3]
- High availability and DR: multi-AZ thinking, backups/snapshots, and understanding RTO/RPO to frame reliable architectures.[7] [3]

Observability mindset

- Logs, metrics, traces: what to capture and how to use signals to detect and resolve issues quickly during rollouts. This underpins CI/CD gates and rollback logic.[3] [2]
- Runbooks: step-by-step procedures for common failures to make responses consistent and auditable.[2] [1]

Practice targets

- Daily 60–90 minutes: one Linux/networking exercise, one Bash/Python task, and one Git action (branch/PR) to build muscle memory before advanced DevOps topics.[3] [1]
- Weekly mini-project: containerize a small API, write Postman tests, and document troubleshooting steps to tie foundations together.[2] [3]

⁂

1. productivity.documentation
2. learning.technical_concepts
3. learning.technical_skills

4. tools.docker_workflow
5. tools.aws_networking
6. tools.aws_infrastructure
7. tools.aws_scaling
8. DevOps.jpg