

Cloud Computing Skills: Core Skills and Practice



Build a practical, role-aware cloud capability plan, not vendor trivia



For practitioners, IT generalists, architects, DevOps, platform, educators, managers



Arc: vocabulary, technical skills, architecture, security, cost, development plan



Apply concepts and produce hands-on artifacts at each checkpoint



Outcome: explain value, choose services, design, automate, measure



Market: ~317,000 US cloud openings a year, about 6x average growth



What Cloud Computing Actually Is



- On-demand network access to a shared pool of configurable resources



- Rapidly provisioned and released with minimal management effort



- Five essential characteristics: self-service, broad access, pooling, elasticity, metering



- Three service models: IaaS, PaaS, SaaS — plus FaaS as a fourth style



- Four deployment models: private, community, public, hybrid



- Enablers: fast networks, cheap servers, virtualization on commodity hardware



Regions, Zones, and the Shared Responsibility Model

- Regions are separate geographic areas; AZs are isolated locations within them
- AZs have independent power, cooling, networking, yet connect via low-latency metro fiber
- Deploy production workloads across at least two AZs; go multi-region only for resilience or data residency
- Provider secures the cloud; you secure what you put in it
- You always own your data, identities, and configuration



Provider responsibility

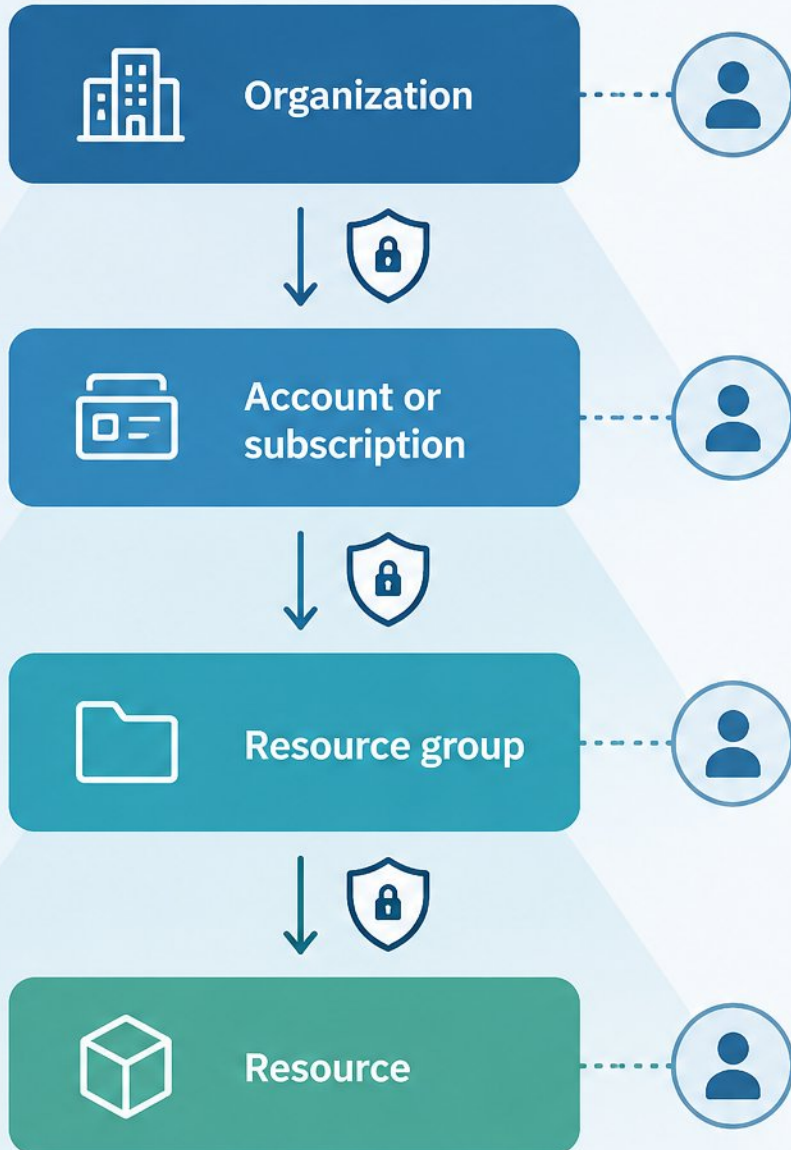
Provider secures the cloud;
you secure what you put in it



Your responsibility

You always own your data,
identities, and configuration

Accounts, Identity, and Resource Hierarchy



Hierarchy: organization, account or subscription, resource group, then resource



Permissions inherit downward — broad parent grants cascade to everything beneath



Identity is the primary security boundary: separate human and workload identities



IAM is default-deny: access refused unless a policy explicitly allows it



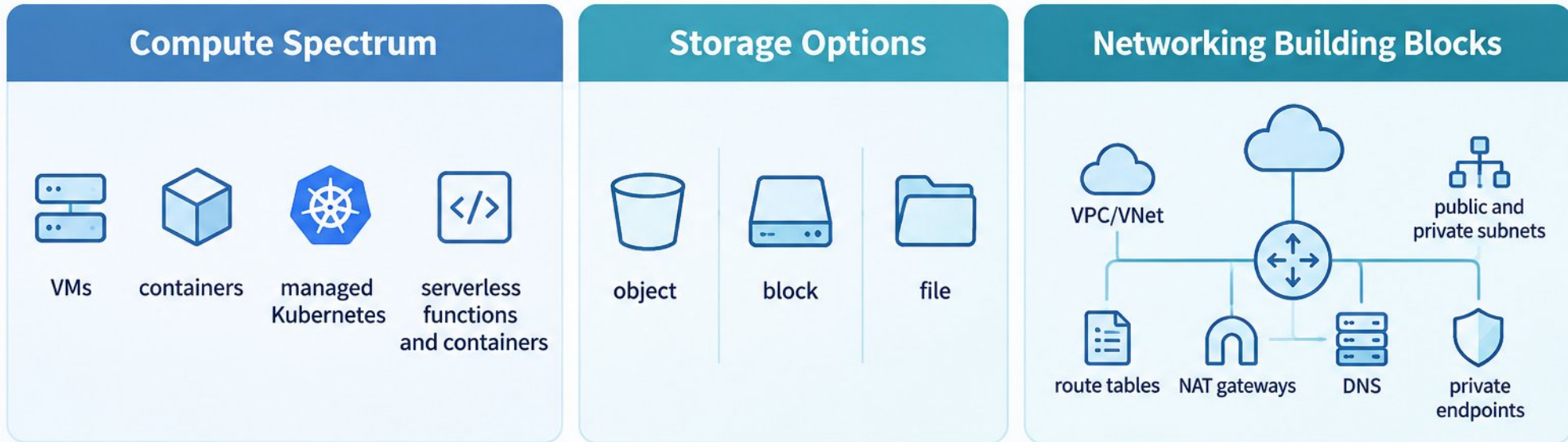
Give every workload its own narrowly scoped identity, never a shared account



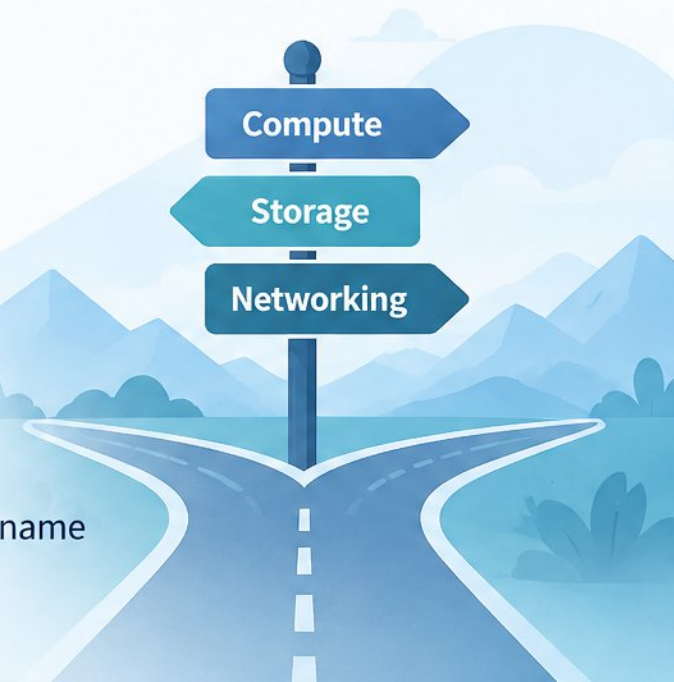
Define the hierarchy and identity model before provisioning any workload



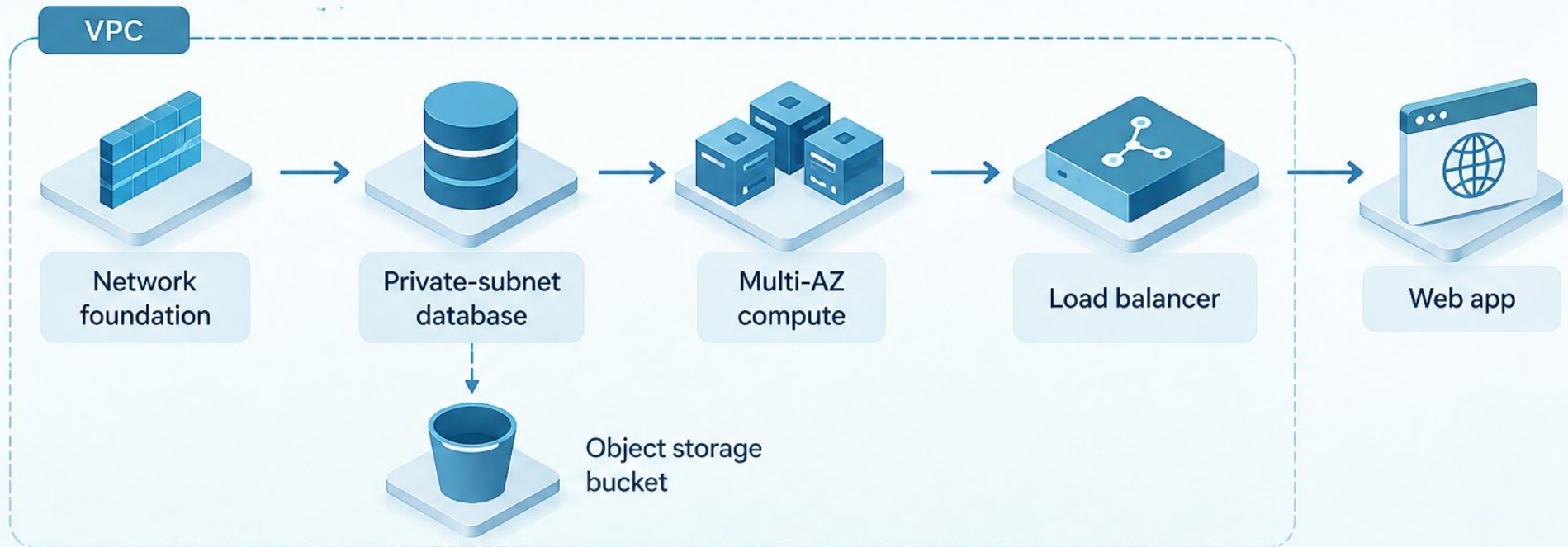
Compute, Storage, and Networking Choices



- Compute spectrum: VMs, containers, managed Kubernetes, serverless functions and containers
- Instance families matter: general purpose, compute, memory, GPU, ARM (19–37% cheaper)
- Storage: object, block, file — choose by access pattern, tier cold data for 50–80% savings
- Networking: VPC/VNet, public and private subnets, route tables, NAT gateways, DNS, private endpoints
- Names differ across AWS, Azure, and GCP — learn the pattern, then the product name



Hands-On Checkpoint: Deploy a Small Workload



Build target: web app, managed database, and object storage bucket in a VPC



Sequence: network foundation, private-subnet database, multi-AZ compute, load balancer



Instrument from the start — logging and basic metrics enabled, not added later



Acceptance: survives one AZ loss; database unreachable from the public internet



Keep cleanup steps and cost estimate for safe personal-account practice



Evidence to keep: architecture diagram, deployment steps, what broke and the fix

Identity, Secrets, and Security by Default



Least privilege: start at zero, add only what the task needs



Right-size using access analyzers and policy generation from real activity



Replace long-lived keys with federation, workload identities, and OIDC



Enforce phishing-resistant MFA; keep root accounts out of daily use



Use vaults, rotation, and dynamic secrets — never hard-code credentials



Set permission boundaries and SCPs as ceilings misconfigured policies cannot exceed



Common Misconfigurations and How to Prevent Them



Misconfiguration leads cloud breaches — people and process, not exotic attacks



Recurring offenders: public storage, open sensitive ports, wildcard IAM, disabled logging



Provider profiles: AWS exposed services, Azure storage and keys, GCP IAM defaults



Weak IAM and missing logging affect 80–98% of accounts: IAM worsens at scale



Prevent: infrastructure as code, pipeline scanning, policy as code, hardened landing zones



Assume every service needs hardening — vendor defaults favor adoption, not security

Recurring Misconfigurations



Public storage



Open sensitive ports



Wildcard IAM



Disabled logging

Provider Profiles



AWS
exposed services



Azure
storage and keys



GCP
IAM defaults



Architecture Patterns for Reliability and Scale



Six well-architected pillars used as a trade-off lens, not a checklist



High availability: multi-AZ active/active or active/standby, health checks, load balancing



DR planning starts with RTO and RPO, then pilot light, warm standby, or multi-region



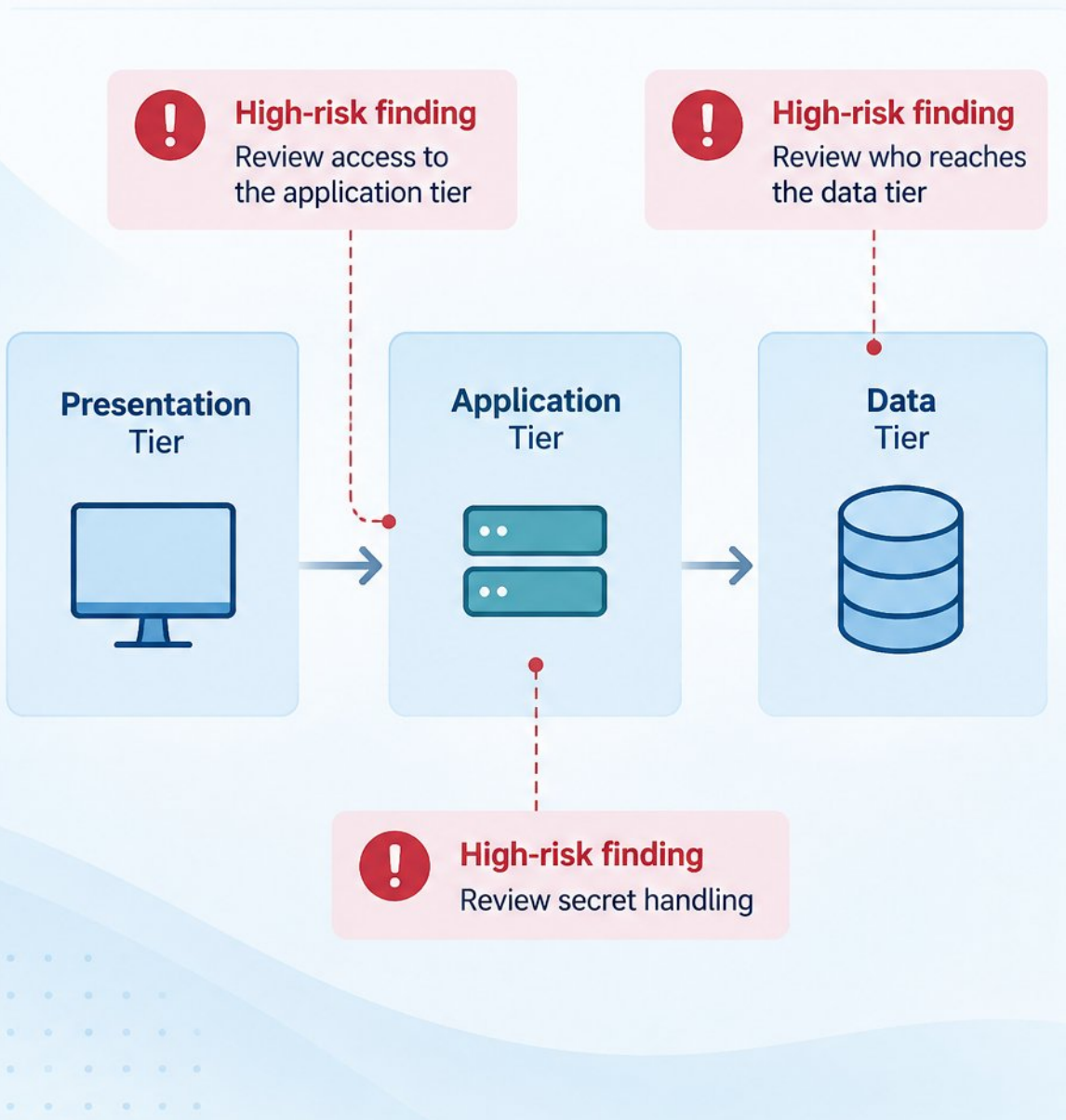
Decoupling: queues, event buses, and pub/sub replace fragile synchronous chains



Data patterns: replication, read replicas, sharding, and tested restore procedures



Architecture Review Checkpoint: Critique a Three-Tier App



Exercise: assess a simple three-tier app against six well-architected pillars



Reliability: how many AZs, failover behaviour, tested RTO and RPO



Security: who reaches the data tier, secret handling, per-component least-privilege identity



Cost and sustainability: idle resources, over-provisioning, storage tiering, deliberate region choice



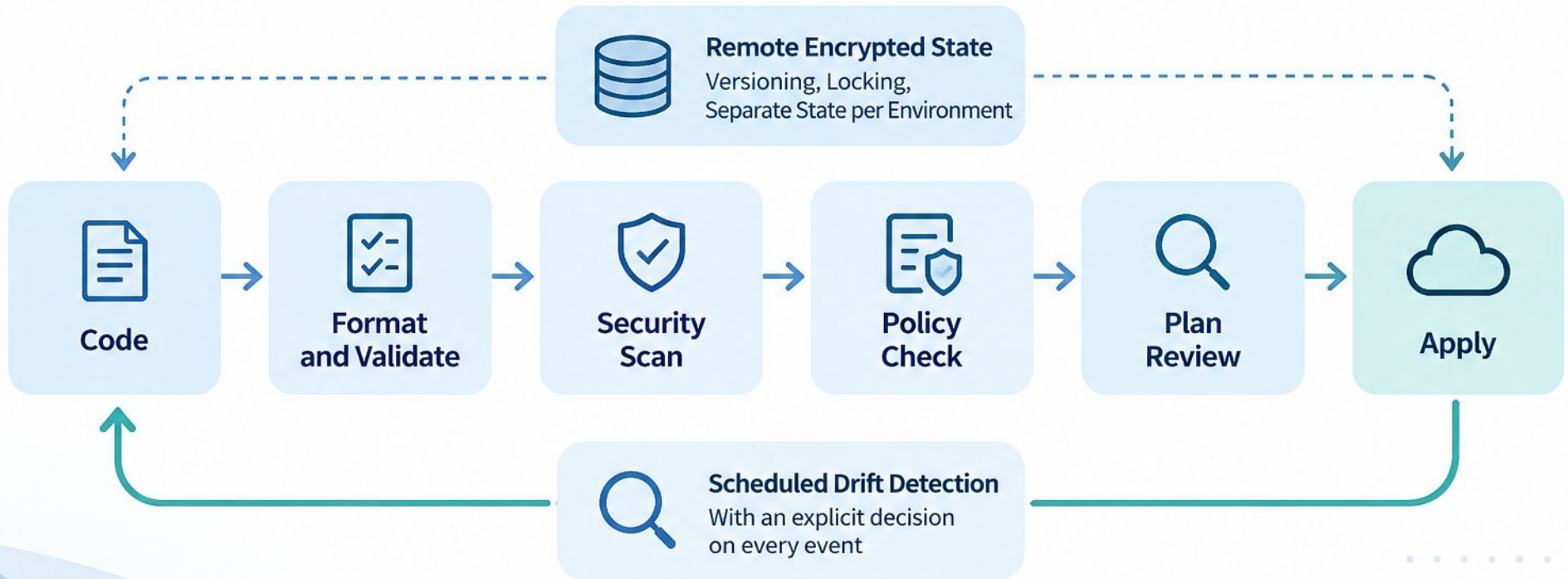
Operational excellence: what is deployed as code, what is observed, rollback path



Deliverable: three high-risk findings and three concrete remediations, presented to the group

Automation and Infrastructure as Code

- Manual console changes: root cause of most misconfigurations and drift
- Declarative tools for the resource graph; imperative code for migrations
- Terraform and OpenTofu for multi-cloud; CloudFormation, Bicep for native
- Remote encrypted state, versioning, locking, separate state per environment
- Scan, policy-check, and review the plan before any apply
- Scheduled drift detection with an explicit decision on every event



Observability, SLOs, and Incident Response

- Metrics, logs, traces: correlate the three, don't read in isolation
- SLI measures user-perceived quality; SLO sets the target; error budget = 100% - SLO
- 99.9% over 30 days $\approx$ 43 min; 99.5% $\approx$ 3.6 hours; never set 100%
- Budget policy: green normal releases, yellow extra approval, red feature freeze
- Burn-rate alerts use a long and short window to cut alert fatigue
- Runbooks and blameless postmortems make response repeatable



SLO and Error Budget

SLI measures user-perceived quality

SLO sets the target

error budget = 100% - SLO



Availability Examples

99.9% over 30 days $\approx$ 43 min

99.5% $\approx$ 3.6 hours



never set 100%

Error Budget Policy



green normal releases



yellow extra approval



red feature freeze

Burn-Rate Alerts



Burn-rate alerts use a long and short window to cut alert fatigue

Incident Response



Runbooks



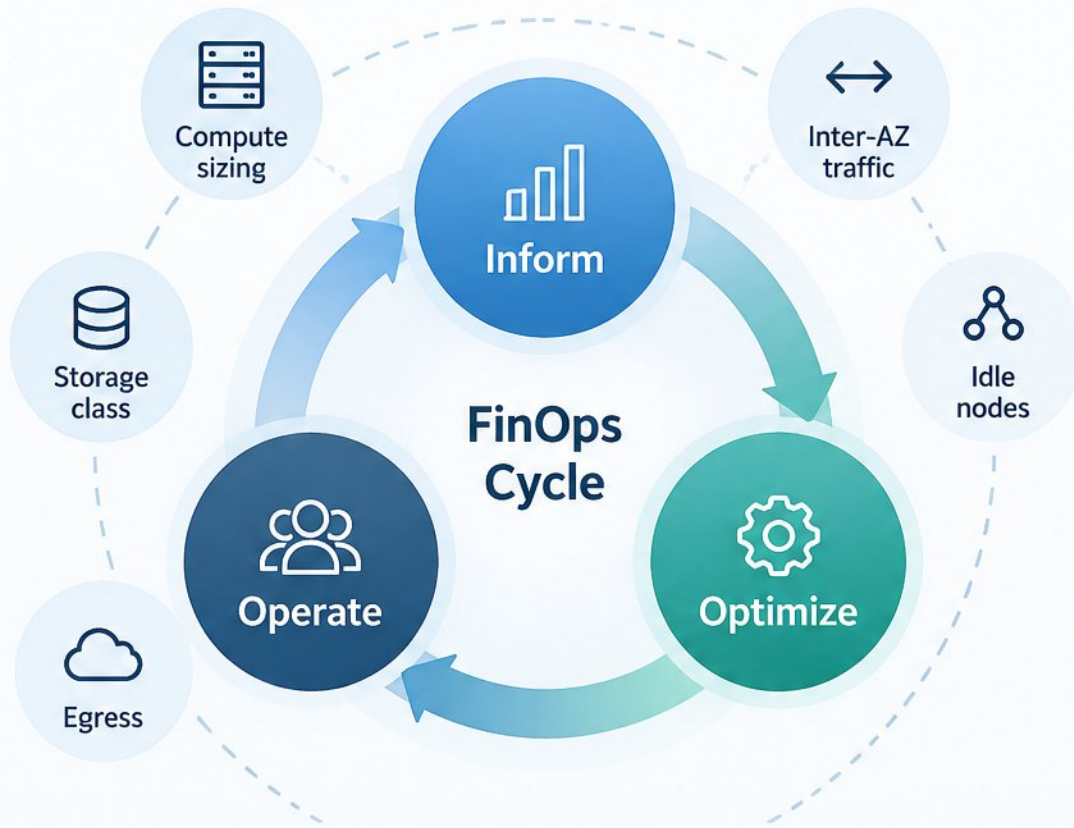
Blameless postmortems



Make response repeatable



Cost, Governance, and Sustainability



Unit Economics



Cost per customer or transaction

Tags at Creation



FinOps cycle: Inform, Optimize, Operate — visibility before optimization



Watch compute sizing, storage class, egress, inter-AZ traffic, idle nodes



Levers in order: kill idle, rightsize, then commit the stable baseline



Enforce tags at creation: environment, owner, cost center, project, managed-by



Unit economics — cost per customer or transaction — shows if spend is healthy



Landing zones and guardrails make new accounts compliant by default

Building a Personal and Team Cloud Capability Plan



Role-based skill maps: engineer, architect, platform, security, manager



Go deep on one platform first, then add Terraform and Kubernetes



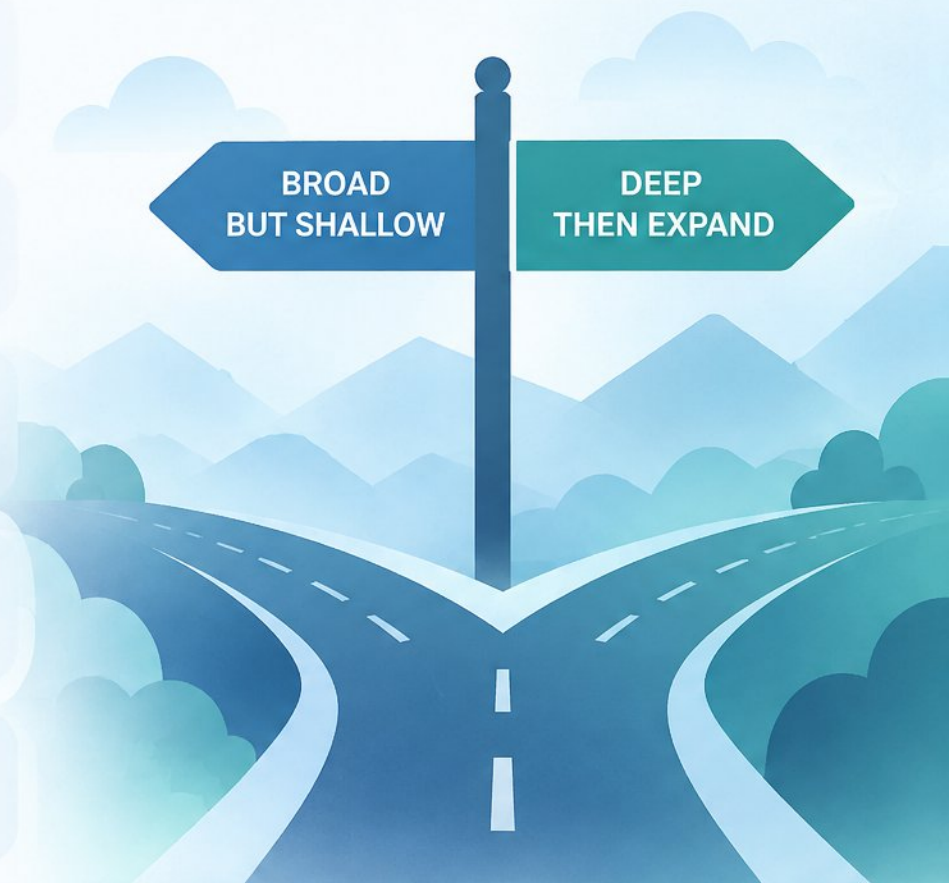
Sequence certifications: foundational, then associate, then specialty



Assess real skill with portfolio artifacts and design reviews



90-day plan: one workload, one certification, one measurable outcome



90-day plan



One workload



One certification



One review cadence



One measurable outcome

PersonWise.

PersonWise • AI digital-human course platform



Scan the code or click anywhere to watch this course live, with voice Q&A

personwise.ai/t/6086a2d6/public