Cloud Deployment Models

Part 1

Cloud Deployment Models (Why They Matter)

- Deployment models describe *where* cloud resources run and *who* controls them
- Choices affect cost, scalability, compliance, and operational complexity
- Most organizations combine models rather than choosing only one
- This sets context for real-world architectures (not “one cloud to rule them all”)

Roadmap and Learning Objectives

- Define deployment models and distinguish them from service models
- Understand public vs private trade-offs (control, elasticity, compliance)
- Introduce hybrid and multi-cloud at a conceptual and practical level
- Build a decision lens you can apply to real workloads
- End with “how orgs actually evolve over time” and why

Deployment Model vs Service Model (Common Confusion)

- Service model = *what layer is managed for you* (IaaS/PaaS/SaaS/FaaS)
- Deployment model = *where it runs and who owns the infrastructure*
- You can mix them (e.g., SaaS + hybrid identity; IaaS in public cloud; PaaS in public cloud)
- Treat them as orthogonal choices, not a single continuum

What Is a Cloud Deployment Model?

- Defines the location and ownership of infrastructure
- Independent of the service model (IaaS, PaaS, SaaS)
- Shapes security boundaries and network design
- Influences governance and compliance strategy

Key Dimensions to Compare (A Practical Checklist)

- **Ownership/operations**: who patches, upgrades, and responds to incidents?
- **Tenancy/isolation**: dedicated vs shared resources; how isolation is achieved
- **Connectivity**: internet-only vs private connectivity options
- **Elasticity**: can you scale quickly without procurement cycles?
- **Compliance & data locality**: where data can live and how it's audited
- **Tooling & governance**: policies, identity, logging, monitoring consistency

Public Cloud: Definition and Characteristics

- Infrastructure owned and operated by a cloud provider
- Resources shared across many customers (multitenancy)
- Accessed over the public internet or private links
- Highly elastic and globally available

Public Cloud: Why It's Attractive

- Rapid scaling and fast provisioning
- No upfront hardware investment
- Easier experimentation: create, test, delete resources quickly
- Global footprints can reduce latency and improve resiliency options
- Access to managed services beyond basic compute/storage

Public Cloud: Constraints and Risk Areas

- Less control over physical location
- Compliance and data residency can be challenging
- Misconfiguration risk shifts “security” toward identity/policy excellence
- Cost variability: pay-per-use can surprise without governance
- Dependency risk: outages/quotas, and service changes outside your control

Public Cloud: Common Enterprise Patterns

- “Landing zone” baseline: account/subscription structure, policies, logging
- Network segmentation: VPC/VNet design, subnets, egress control, firewalls
- Identity-first: centralized IAM, SSO, least privilege, break-glass procedures
- Standard deployments: templates/IaC + approved service catalog
- Guardrails: tagging, cost controls, policy-as-code, security scanning

Private Cloud: Definition and Characteristics

- Infrastructure dedicated to a single organization
- Can be on-premises or hosted by a provider
- Uses cloud-style virtualization and automation
- Greater control and isolation

Private Cloud: What You Gain

- Stronger control over environment details (hardware, network, physical access)
- Predictability for performance-sensitive or tightly governed workloads
- Easier to enforce bespoke security controls (when required)
- Potentially simpler data locality assurance (depending on footprint)
- Integration with legacy systems may be more straightforward (same environment)

When Private Cloud Makes Sense

- Strict regulatory or compliance requirements
- Predictable workloads
- Strong internal IT capabilities
- When isolation requirements exceed what your public-cloud design can satisfy
- When hardware dependencies exist (special appliances, constrained environments)

Private Cloud: Trade-offs and Failure Modes

- Higher cost and lower elasticity than public cloud
- Capacity planning returns: you must plan and purchase for peak
- Operational load remains high (patching, upgrades, DR planning, staffing)
- Risk: “private cloud” becomes “virtualized on-prem” without true self-service
- Risk: automation gaps lead to slower delivery than expected

Cloud Deployment Models

Part 2

Hybrid Cloud: Combining Public and Private

- Mixes public cloud with private infrastructure
- Workloads span environments
- Common during cloud migration
- Requires strong integration and networking

Why Hybrid Happens (Even If It's Not the End Goal)

- Stepwise migration: move parts while minimizing business disruption
- Data gravity: some datasets/systems remain on private infrastructure longer
- Regulatory or residency constraints for specific components
- Latency and integration requirements (legacy systems and internal networks)
- Organizational reality: different teams adopt at different speeds

Hybrid Integration Pattern: Network Connectivity

- Site-to-site VPNs over the internet (quick start, variable performance)
- Dedicated private connections (more consistent, higher throughput)
- Routing and segmentation: what can talk to what across environments
- DNS strategy: name resolution across boundaries
- Failure planning: what happens when the link drops?

Hybrid Integration Pattern: Identity and Access

- Identity and access federation across environments
- Single sign-on and centralized policy enforcement (roles, groups, MFA)
- Least privilege across boundaries: avoid “everyone is admin everywhere”
- Account lifecycle automation (provisioning/deprovisioning)
- Auditability: consistent logs for who accessed what and when

Hybrid Integration Pattern: Unified Monitoring and Management

- Unified monitoring and management is a core requirement
- Standard telemetry: logs, metrics, traces across environments
- Incident response: one playbook, one on-call model, clear escalation
- Asset inventory: know what's running where (and who owns it)
- Policy enforcement: consistent configuration baselines and drift detection

Hybrid Cloud: Common Pitfalls (and Mitigations)

- "Two of everything": duplicated tooling and fragmented teams → standardize interfaces
- Network complexity surprises → design and test routing/security early
- Data sync issues → define authoritative sources and replication strategy
- Inconsistent security posture → unify identity, logging, and policy controls
- Migration stalls → define target-state milestones and retire legacy dependencies

Multi-Cloud: Definition and Motivation

- Uses services from multiple cloud providers
- Avoids single-vendor dependence
- Enables best-of-breed service selection
- Often increases architectural complexity

Multi-Cloud: When It's Worth Considering

- Business requirements for vendor risk reduction or negotiating leverage
- Regulatory constraints that differ by geography/provider
- Mergers/acquisitions (inherited clouds)
- Specialized services needed in one cloud (with a deliberate scope)
- Resilience strategy (but only if you can operate it competently)

Challenges of Multi-Cloud (What Actually Gets Hard)

- Inconsistent APIs and services
- Higher operational and tooling overhead
- Networking and data transfer complexity
- Requires mature cloud governance
- Skills matrix: teams must know multiple platforms deeply (not shallowly)

Multi-Cloud Architecture Patterns (Ways to Reduce Pain)

- Keep the “portable core” consistent (identity approach, logging standards, IaC practices)
- Standardize on containers/orchestration where it genuinely helps
- Minimize cross-cloud data movement (treat data gravity as a first-class constraint)
- Use abstraction selectively (don’t hide differences that matter operationally)
- Decide “where not to be multi-cloud” (scope it to avoid accidental sprawl)

Comparing Deployment Models (One-Slide Summary)

- Public: maximum elasticity, minimal control
- Private: maximum control, higher cost
- Hybrid: balance of flexibility and compliance
- Multi-cloud: resilience and leverage, highest complexity
- Key reminder: complexity is also a “cost” (people/time/risk), not just dollars

Strategic Factors in Choosing a Model

- Regulatory and data locality requirements
- Cost predictability vs flexibility
- Organizational cloud maturity
- Risk tolerance and vendor strategy
- Integration constraints (legacy, latency, data gravity, operating model)

Deployment Models in Practice (How Orgs Evolve)

- Most organizations evolve over time
- Common path: start hybrid, move selectively to public cloud
- Choices change as needs change (growth, regulation, org maturity)
- No single “correct” model for all cases
- Good practice: periodically reassess “why this lives here” for major workloads

Wrap-Up: A Simple Decision Framework

- Step 1: List constraints (compliance, locality, latency, integration)
- Step 2: Identify variability of demand (spiky vs predictable) and desired elasticity
- Step 3: Assess operational capability (skills, tooling, on-call maturity)
- Step 4: Choose the simplest model that satisfies constraints (avoid complexity by default)
- Step 5: Document governance: identity, network boundaries, logging, and ownership