Cloud Computing Architecture

Chapter-by-Chapter Implementation Mapping

A Strategic Learning Path for

Docker • Kubernetes • Terraform

Technical Reference Guide

Highest-ROI Reading Strategy with Implementation Translations

Based on Thomas Erl's

Cloud Computing: Concepts, Technology & Architecture

Pearson Digital Enterprise Book Series

Contents

1 Executive Summary

This document provides a comprehensive chapter-by-chapter mapping of where the *Cloud Computing: Concepts, Technology & Architecture* book most improves your effectiveness with **Docker**, **Kubernetes**, and **Terraform**. The analysis identifies enduring mental models, security primitives, and architecture patterns that reduce trial-and-error during implementation.

1.1 Document Purpose

The mapping serves three primary objectives:

1. **Prioritized Reading Strategy**: Identify the highest-ROI chapters for practitioners with limited time (approximately 30–40% of total content).
2. **Implementation Translation**: Connect theoretical concepts to concrete configurations in cloud-native tooling.
3. **Learning Phase Alignment**: Match book content to your current phase—primer, active building, or architecture hardening.

1.2 Target Audience

This guide is designed for:

- Infrastructure engineers transitioning to cloud-native platforms
- DevOps practitioners deepening their conceptual foundations
- Security engineers implementing cloud security controls
- Platform architects designing multi-cloud or hybrid environments

1.3 Official Documentation Resources

The following official documentation should be used alongside this mapping guide for implementation details:

1.3.1 Docker Documentation

- **Main Documentation**: https://docs.docker.com/
- **Getting Started**: https://docs.docker.com/get-started/
- **Build Reference**: https://docs.docker.com/build/
- **Compose**: https://docs.docker.com/compose/
- **Security**: https://docs.docker.com/engine/security/
- **Best Practices**: https://docs.docker.com/develop/develop-images/guidelines/

1.3.2 Kubernetes Documentation

- **Main Documentation**: https://kubernetes.io/docs/home/
- **Concepts**: https://kubernetes.io/docs/concepts/

- **Tasks**: https://kubernetes.io/docs/tasks/
- **Tutorials**: https://kubernetes.io/docs/tutorials/
- **API Reference**: https://kubernetes.io/docs/reference/
- **Security**: https://kubernetes.io/docs/concepts/security/

1.3.3 Terraform Documentation

- **Main Documentation**: https://developer.hashicorp.com/terraform/docs
- **Tutorials**: https://developer.hashicorp.com/terraform/tutorials
- **Language Reference**: https://developer.hashicorp.com/terraform/language
- **CLI Reference**: https://developer.hashicorp.com/terraform/cli
- **Provider Registry**: https://registry.terraform.io/
- **Best Practices**: https://developer.hashicorp.com/terraform/cloud-docs/recommended-practices

2 Highest-ROI Chapters

If reading only 30–40% of the book, prioritize these chapters for maximum practical value:

2.1 Chapter 4: Fundamental Concepts and Models

Value Proposition: Best vendor-neutral grounding for boundaries, roles, service models, and deployment models.

Direct Applications:

- Terraform module boundaries and environment modeling
  See: https://developer.hashicorp.com/terraform/language/modules
- Kubernetes multi-tenancy and namespace strategy
  See: https://kubernetes.io/docs/concepts/security/multi-tenancy/
- Docker isolation assumptions and container boundaries
  See: https://docs.docker.com/engine/security/
- Account/subscription layout for cloud providers

2.2 Chapter 6: Understanding Containerization

Value Proposition: Most directly aligned to Docker and Kubernetes fundamentals.

Core Topics:

- Container images, layers, and immutability
  See: https://docs.docker.com/build/concepts/overview/
- Container engines and orchestration concepts
  See: https://docs.docker.com/engine/

- Pods, host clusters, and overlay networks
  See: https://kubernetes.io/docs/concepts/workloads/pods/
- Sidecar, ambassador, and adapter patterns
  See: https://kubernetes.io/docs/concepts/workloads/pods/sidecar-containers/

2.3 Chapter 7: Cloud Security and Cybersecurity

Value Proposition: Provides the vocabulary and threat framing reused across all implementations.

Security Domains Addressed:

- RBAC design and network policy
  See: https://kubernetes.io/docs/reference/access-authn-authz/rbac/
- Secrets management and image trust
  See: https://kubernetes.io/docs/concepts/configuration/secret/
- Terraform guardrails and policy-as-code
  See: https://developer.hashicorp.com/sentinel/docs
- Shared responsibility model understanding

2.4 Chapters 8–12: Mechanisms

Value Proposition: Maps cleanly to "what you actually configure" in cloud and Kubernetes environments.

Coverage Includes:

- Infrastructure: perimeters, virtual servers, hypervisors, storage, containers
  See: https://kubernetes.io/docs/concepts/storage/
- Specialized: scaling listeners, load balancers, failover systems, clusters, state management
  See: https://kubernetes.io/docs/concepts/services-networking/
- Security (Access): encryption, PKI, SSO, firewalls, VPN, IAM, IDS, MFA
  See: https://kubernetes.io/docs/concepts/security/
- Security (Data): DLP, backup/recovery, traffic monitoring, malware analysis
- Management: remote admin, resource management, SLA/billing management
  See: https://developer.hashicorp.com/terraform/language/state

2.5 Chapters 13–15: Architectures

Value Proposition: Essential for designing clusters and environments beyond tutorial-level implementations.

Architecture Patterns:

- Workload distribution and resource pooling
- Zero downtime and disaster recovery
- VPC patterns and data sovereignty

- Multi-cloud and federated architectures

2.6 Chapters 16–18: Delivery, Cost, and SLA

Value Proposition: Translates into concrete FinOps and SRE requirements.

Operational Implications:

- Terraform state and backend configuration
  See: https://developer.hashicorp.com/terraform/language/settings/backends
- Kubernetes autoscaling policies
  See: https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/
- Cost management and pricing model understanding
- Service quality metrics and SLA definitions

3 Chapter-by-Chapter Implementation Mapping

3.1 Reading Legend

Category	Description
Depth	*Skip* / *Skim* / *Read* / *Study*
When	*Before* (primer) / *During* (while building) / *After* (architecture hardening)
Value Rating	Low / Medium / **High** (bold indicates critical value)

Table 1: Legend for interpreting the chapter mapping table

3.2 Complete Chapter Mapping

Chapter	Primary Value	Docker	K8s	TF	Depth	When
1: Introduction	Sets expectations; helps avoid reading as "how-to manual"	Indirect	Indirect	Indirect	Skim	Before
2: Case Study Background	Realistic constraints, governance, roadmaps, tradeoffs	Indirect	Med	Med	Skim	During/ After
3: Understanding Cloud Computing	Business drivers, risk framing, trust boundaries, portability	Low	Med	Med	Skim	Before

Continued on next page...

Chapter	Primary Value	Docker	K8s	TF	Depth	When
4: Fundamental Concepts & Models	Core mental models: roles, boundaries, elasticity, IaaS/PaaS/SaaS	Med	**High**	**High**	**Read/ Study**	Before/ During
5: Cloud-Enabling Technology	Networks, DC tech, virtualization, multitenancy, APIs	Med	**High**	**High**	Read	Before/ During
6: Understanding Containerization	Images, layers, engines, pods, networks, sidecar patterns	**High**	**High**	Med	**Study**	Before/ During
7: Cloud Security & Cybersecurity	Threat vocabulary, trust boundaries, shared responsibility	**High**	**High**	**High**	**Study**	Before/ During
8: Cloud Infrastructure Mechanisms	Perimeters, virtual servers, hypervisors, storage, containers	Med	**High**	**High**	Read	During
9: Specialized Cloud Mechanisms	Scaling, load balancers, SLA monitors, failover, clusters	Med	**High**	**High**	Read	During/ After
10: Access-Oriented Security	Encryption, PKI, SSO, firewalls, VPN, IAM, IDS, MFA	Med	**High**	**High**	Read/ Study	During
11: Data-Oriented Security	DLP, backup/recovery, traffic/log monitoring, malware analysis	Med	**High**	**High**	Read	During/ After
12: Cloud Management Mechanisms	Remote admin, resource mgmt, SLA mgmt, billing mgmt	Low	Med	**High**	Skim/ Read	After
13: Fundamental Cloud Architectures	Workload distribution, pooling, elastic capacity, multicloud	Low	**High**	**High**	Read	After
14: Advanced Cloud Architectures	Zero downtime, DR, data sovereignty, VPC, rapid provisioning	Low	**High**	**High**	Study (selected)	After
15: Specialized Cloud Architectures	Edge/fog, persistent network config, federated architectures	Low	Med	Med	Skim	After/ Optional
16: Delivery Model Considerations	Provider vs consumer perspective, operational responsibilities	Low	Med	**High**	Read	Before/ After

Continued on next page...

Chapter	Primary Value	Docker	K8s	TF	Depth	When
17: Cost Metrics & Pricing	FinOps basics: network egress, instance allocations, storage I/O	Low	Med	**High**	Read	During/ After
18: Service Quality & SLAs	SRE alignment: availability, MTBF/MTTR, capacity, scalability	Low	**High**	**High**	Read	After
Appendix B: Containerization Technologies	Docker/K8s component glossary and terminology alignment	Med	Med	Low	Skim	During

Table 2: Chapter-by-chapter mapping to Docker, Kubernetes, and Terraform

4 Concept-to-Implementation Translations

This section maps theoretical concepts from the book to concrete implementation decisions, identifying where the book prevents expensive mistakes.

4.1 Trust Boundary vs Organizational Boundary

Source: Chapters 4 and 7

Technology	Implementation Translation
Kubernetes	Namespace/tenancy strategy, RBAC scoping, network policy segmentation, cluster vs namespace isolation decisions (https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/)
Terraform	Account/subscription separation, environment boundaries, state isolation (https://developer.hashicorp.com/terraform/language/state/workspaces)
Docker	Container isolation assumptions, security context boundaries (https://docs.docker.com/engine/security/seccomp/)

4.2 Shared Responsibility Model

Source: Chapters 3, 7, and 16

What You Must Secure Yourself:

- Identity and Access Management (IAM) configuration
- Logging and audit trail implementation
- Encryption key ownership and rotation

- Cluster hardening and CIS benchmark compliance
- Image provenance and supply chain security
- Backup strategy and disaster recovery

4.3 Elasticity and Measured Usage

Source: Chapters 4, 17, and 18

Technology	Implementation Translation
Kubernetes	HPA (Horizontal Pod Autoscaler), VPA (Vertical Pod Autoscaler), Cluster Autoscaler policies (https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/)
Terraform	Autoscaling groups, node group configurations, scaling policies as code (https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/autoscaling_group)
FinOps	Guardrails, budget alerts, cost allocation tags, reserved capacity planning

4.4 Container Immutability and Image Layering

Source: Chapter 6

Build Discipline Requirements:

- Reproducible builds with pinned dependencies
  See: https://docs.docker.com/build/building/best-practices/
- Minimal base images (distroless, Alpine, scratch)
  See: https://docs.docker.com/build/building/base-images/
- Defined patch cadence and vulnerability scanning
- Software Bill of Materials (SBOM) generation
  See: https://docs.docker.com/build/metadata/attestations/sbom/
- Image signing and verification (Cosign, Notary)
  See: https://docs.docker.com/engine/security/trust/

4.5 Logical Network Perimeter

Source: Chapters 8 and 10

Context	Implementation Translation
Cloud Infrastructure	VPC/VNet segmentation, security groups, firewall rules, private endpoints
Kubernetes	Network policies, ingress/egress controls, service mesh (Istio, Linkerd) (https://kubernetes.io/docs/concepts/services-networking/network-policies/)
Terraform	Network module design, security group rules as code, private link configurations (https://developer.hashicorp.com/terraform/tutorials/networking)

4.6 State Management Database

Source: Chapter 9

Context	Implementation Translation
Kubernetes	Control plane state (etcd), understanding consistency and availability tradeoffs (https://kubernetes.io/docs/tasks/administer-cluster/configure-upgrade-etcd/)
Terraform	Remote state backends (S3, GCS, Azure Blob), state locking, state encryption (https://developer.hashicorp.com/terraform/language/settings/backends/configuration)
Applications	Stateful workload decisions, PersistentVolume strategies, operator patterns (https://kubernetes.io/docs/concepts/storage/persistent-volumes/)

4.7 Audit, SLA, and Usage Monitors

Source: Chapters 8, 9, and 18

Observability Requirements:

- Logging requirements (structured logs, retention policies, centralization)
- Metrics collection (Prometheus, CloudWatch, Datadog)
- Distributed tracing (Jaeger, Zipkin, X-Ray)
- Defining "done" as observable and measurable
- SLA/SLO/SLI definitions and error budgets

4.8 Failover and DR Architectures

Source: Chapters 14 and 18

Pattern	Implementation Considerations
Multi-zone	Single region, multiple availability zones, synchronous replication
Multi-region	Multiple regions, asynchronous replication, higher latency tolerance
Backup/Restore	Cold standby, RTO measured in hours, lowest cost
Active-Active	Zero/minimal RTO, highest cost, complex state synchronization

4.9 Hardened Images

Source: Chapter 10

Golden Image Patterns:

- CIS benchmark compliance for node images
- Immutable infrastructure alignment
- Automated image building pipelines (Packer, image-builder)
- Regular patching and rotation schedules
- Security scanning integration (Trivy, Grype, Clair)

4.10 Portability Limits

Source: Chapters 3, 4, 13, and 17

Realistic Multi-cloud Posture:

- **Portable**: Application containers, Kubernetes manifests, Terraform modules (with abstraction)
- **Accept Lock-in**: Managed services where they provide significant leverage (managed databases, ML services, serverless)
- **Strategy**: Portability where it matters (applications), acceptance of provider lock-in where it buys operational efficiency

5 Practical Learning Path

5.1 Phase 1: Before Starting Labs (Fast Primer)

Reading Focus: Chapter 4, targeted sections of Chapters 5 and 7

Objectives:

- Understand cloud delivery models (IaaS, PaaS, SaaS) and their boundaries
- Grasp organizational and trust boundary concepts
- Internalize the shared responsibility model
- Develop threat vocabulary for security discussions

Time Investment: 4–6 hours

5.2 Phase 2: While Learning Docker and Kubernetes

Reading Focus: Chapter 6 + relevant sections of Chapters 10–11

Recommended Documentation:

- Docker Getting Started: https://docs.docker.com/get-started/
- Kubernetes Basics Tutorial: https://kubernetes.io/docs/tutorials/kubernetes-basics/

Chapter 6 Deep Dive:

- Container images, layers, and immutability principles
- Container engines and orchestration fundamentals
- Pod concepts and multi-container patterns
- Network overlay and service discovery

Security Integration (Chapters 10–11):

- IAM and RBAC design principles
  See: https://kubernetes.io/docs/reference/access-authn-authz/
- Encryption at rest and in transit
- Hardened image creation and maintenance
- Monitoring and audit logging
  See: https://kubernetes.io/docs/tasks/debug/debug-cluster/audit/

Time Investment: 8–12 hours

5.3 Phase 3: While Learning Terraform

Reading Focus: Chapters 8–9 + Chapters 16–18

Recommended Documentation:

- Terraform Getting Started: https://developer.hashicorp.com/terraform/tutorials/aws-get-started
- Terraform Language: https://developer.hashicorp.com/terraform/language
- Provider Registry: https://registry.terraform.io/browse/providers

Infrastructure Mechanisms (Chapters 8–9):

- Logical network perimeter design
- Virtual server and storage provisioning
- Scaling listeners and load balancer patterns
- State management and backend architecture
  See: https://developer.hashicorp.com/terraform/language/state

Operational Considerations (Chapters 16–18):

- Delivery model responsibilities (what you must manage)
- Cost metrics and pricing model understanding
- SLA definitions and SRE metric alignment

Time Investment: 6–10 hours

5.4 Phase 4: Designing Production Environments

Reading Focus: Chapters 13–14 + Chapter 18

Architecture Patterns (Chapters 13–14):

- Workload distribution and resource pooling
- Dynamic scalability and elastic capacity
- Zero downtime and disaster recovery architectures
- VPC architecture and data sovereignty

SRE Alignment (Chapter 18):

- Service quality metrics (availability, reliability, performance)
- SLA guidelines and definitions
- MTBF, MTTR, and recovery objectives

Time Investment: 8–12 hours

6 Complete Book Structure Reference

6.1 Part I: Fundamental Cloud Computing

6.1.1 Chapter 3: Understanding Cloud Computing

- **3.1 Origins and Influences**: Brief history, definitions, business drivers (cost reduction, business agility), technology innovations (clustering, grid computing, capacity planning, virtualization, containerization, serverless)
- **3.2 Basic Concepts and Terminology**: Cloud, container, IT resource, on premises, cloud consumers/providers, scaling (horizontal/vertical), cloud service, cloud service consumer
- **3.3 Goals and Benefits**: Increased responsiveness, reduced investments, increased scalability, increased availability and reliability
- **3.4 Risks and Challenges**: Overlapping trust boundaries, shared security responsibility, cyber threat exposure, reduced governance control, limited portability, compliance issues, cost overruns

6.1.2 Chapter 4: Fundamental Concepts and Models

- **4.1 Roles and Boundaries**: Cloud provider, cloud consumer, cloud broker, cloud service owner, cloud resource administrator, organizational boundary, trust boundary

- **4.2 Cloud Characteristics**: On-demand usage, ubiquitous access, multitenancy, elasticity, measured usage, resiliency
- **4.3 Cloud Delivery Models**: IaaS, PaaS, SaaS, comparing and combining delivery models, cloud delivery submodels
- **4.4 Cloud Deployment Models**: Public clouds, private clouds, multiclouds, hybrid clouds

6.1.3 Chapter 5: Cloud-Enabling Technology

- **5.1 Networks and Internet Architecture**: ISPs, packet switching, router interconnectivity, physical network, transport/application layer protocols, connectivity/bandwidth/latency issues
- **5.2 Cloud Data Center Technology**: Virtualization, standardization, autonomic computing, remote operation, high availability, security-aware design, facilities, hardware (computing, storage, network)
- **5.3 Modern Virtualization**: Hardware independence, server consolidation, resource replication, OS-based virtualization, hardware-based virtualization, containers, virtualization management
- **5.4 Multitenant Technology**
- **5.5 Service Technology and Service APIs**: REST services, web services, service agents, service middleware, web-based RPC

6.1.4 Chapter 6: Understanding Containerization

- **6.1 Origins and Influences**: Brief history, containerization and cloud computing
- **6.2 Fundamental Virtualization and Containerization**: OS basics, virtualization basics (physical/virtual servers, hypervisors, virtualization types), containerization basics (containers, images, engines, pods, hosts, clusters, networks)
- **6.3 Understanding Containers**: Container hosting, containers and pods, instances and clusters, package management, orchestration, container networks
- **6.4 Understanding Container Images**: Image types and roles, immutability, abstraction, build files, layers
- **6.5 Multi-Container Types**: Sidecar container, adapter container, ambassador container

6.1.5 Chapter 7: Understanding Cloud Security and Cybersecurity

- **7.1 Basic Security Terminology**: Confidentiality, integrity, availability, authenticity, security controls, mechanisms, policies
- **7.2 Basic Threat Terminology**: Risk, vulnerability, exploit, zero-day, security/data breach, data leak, threat, attack, attacker, attack vector and surface
- **7.3 Threat Agents**: Anonymous attacker, malicious service agent, trusted attacker, malicious insider
- **7.4 Common Threats**: Traffic eavesdropping, malicious intermediary, DoS, insufficient authorization, virtualization attack, overlapping trust boundaries, containerization attack, malware,

insider threat, social engineering, botnet, privilege escalation, brute force, RCE, SQL injection, tunneling, APT

- **7.6 Additional Considerations**: Flawed implementations, security policy disparity, contracts, risk management

6.2 Part II: Cloud Computing Mechanisms

6.2.1 Chapter 8: Cloud Infrastructure Mechanisms

- Logical Network Perimeter
- Virtual Server
- Hypervisor
- Cloud Storage Device (storage levels, network/object/database storage interfaces)
- Cloud Usage Monitor (monitoring agent, resource agent, polling agent)
- Resource Replication
- Ready-Made Environment
- Container

6.2.2 Chapter 9: Specialized Cloud Mechanisms

- Automated Scaling Listener
- Load Balancer
- SLA Monitor (polling agent, monitoring agent)
- Pay-Per-Use Monitor
- Audit Monitor
- Failover System (active-active, active-passive)
- Resource Cluster
- Multi-Device Broker
- State Management Database

6.2.3 Chapter 10: Cloud Security Access-Oriented Mechanisms

- Encryption (symmetric, asymmetric)
- Hashing
- Digital Signature
- Cloud-Based Security Groups
- Public Key Infrastructure (PKI) System
- Single Sign-On (SSO) System

- Hardened Virtual Server Image
- Firewall
- Virtual Private Network (VPN)
- Biometric Scanner
- Multi-Factor Authentication (MFA) System
- Identity and Access Management (IAM) System
- Intrusion Detection System (IDS)
- Penetration Testing Tool
- User Behavior Analytics (UBA) System
- Third-Party Software Update Utility
- Network Intrusion Monitor
- Authentication Log Monitor
- VPN Monitor

6.2.4 Chapter 11: Cloud Security Data-Oriented Mechanisms

- Digital Virus Scanning and Decryption System
- Malicious Code Analysis System
- Data Loss Prevention (DLP) System
- Trusted Platform Module (TPM)
- Data Backup and Recovery System
- Activity Log Monitor
- Traffic Monitor
- Data Loss Protection Monitor

6.2.5 Chapter 12: Cloud Management Mechanisms

- Remote Administration System
- Resource Management System
- SLA Management System
- Billing Management System

6.3 Part III: Cloud Computing Architecture

6.3.1 Chapter 13: Fundamental Cloud Architectures

- Workload Distribution Architecture
- Resource Pooling Architecture

- Dynamic Scalability Architecture
- Elastic Resource Capacity Architecture
- Service Load Balancing Architecture
- Cloud Bursting Architecture
- Elastic Disk Provisioning Architecture
- Redundant Storage Architecture
- Multicloud Architecture

6.3.2 Chapter 14: Advanced Cloud Architectures

- Hypervisor Clustering Architecture
- Virtual Server Clustering Architecture
- Load-Balanced Virtual Server Instances Architecture
- Nondisruptive Service Relocation Architecture
- Zero Downtime Architecture
- Cloud Balancing Architecture
- Resilient Disaster Recovery Architecture
- Distributed Data Sovereignty Architecture
- Resource Reservation Architecture
- Dynamic Failure Detection and Recovery Architecture
- Rapid Provisioning Architecture
- Storage Workload Management Architecture
- Virtual Private Cloud Architecture

6.3.3 Chapter 15: Specialized Cloud Architectures

- Direct I/O Access Architecture
- Direct LUN Access Architecture
- Dynamic Data Normalization Architecture
- Elastic Network Capacity Architecture
- Cross-Storage Device Vertical Tiering Architecture
- Intra-Storage Device Vertical Data Tiering Architecture
- Load-Balanced Virtual Switches Architecture
- Multipath Resource Access Architecture
- Persistent Virtual Network Configuration Architecture

- Redundant Physical Connection for Virtual Servers Architecture
- Storage Maintenance Window Architecture
- Edge Computing Architecture
- Fog Computing Architecture
- Virtual Data Abstraction Architecture
- Metacloud Architecture
- Federated Cloud Application Architecture

6.4 Part IV: Working with Clouds

6.4.1 Chapter 16: Cloud Delivery Model Considerations

- **Cloud Provider Perspective**: Building IaaS environments (data centers, scalability, reliability, monitoring, security), equipping PaaS environments, optimizing SaaS environments
- **Cloud Consumer Perspective**: Working with IaaS/PaaS/SaaS environments, IT resource provisioning considerations

6.4.2 Chapter 17: Cost Metrics and Pricing Models

- **Business Cost Metrics**: Up-front and ongoing costs, additional costs
- **Cloud Usage Cost Metrics**: Network usage (inbound, outbound, intra-cloud WAN), server usage (on-demand, reserved), cloud storage device usage, cloud service usage
- **Cost Management Considerations**: Pricing models, multicloud cost management

6.4.3 Chapter 18: Service Quality Metrics and SLAs

- **Service Quality Metrics**:
  - Availability metrics (availability rate, outage duration)
  - Reliability metrics (MTBF, reliability rate)
  - Performance metrics (network/storage/server/web app capacity, instance starting time, response time, completion time)
  - Scalability metrics (storage horizontal, server horizontal/vertical)
  - Resiliency metrics (MTSO, MTSR)
- **SLA Guidelines**: Scope, service quality guarantees, definitions, financial credits, exclusions

6.5 Part V: Appendices

6.5.1 Appendix B: Common Containerization Technologies

Docker (https://docs.docker.com/):

- Docker Server, Docker Client, Docker Registry
  See: https://docs.docker.com/get-started/docker-overview/

- Docker Objects
  See: https://docs.docker.com/get-started/docker-concepts/the-basics/what-is-an-image/
- Docker Swarm (Container Orchestrator)
  See: https://docs.docker.com/engine/swarm/

Kubernetes (https://kubernetes.io/docs/):

- Kubernetes Node (Host)
  See: https://kubernetes.io/docs/concepts/architecture/nodes/
- Kubernetes Pod
  See: https://kubernetes.io/docs/concepts/workloads/pods/
- Kubelet, Kube-Proxy
  See: https://kubernetes.io/docs/reference/command-line-tools-reference/kubelet/
- Container Runtime (Container Engine)
  See: https://kubernetes.io/docs/setup/production-environment/container-runtimes/
- Cluster
  See: https://kubernetes.io/docs/concepts/architecture/
- Kubernetes Control Plane
  See: https://kubernetes.io/docs/concepts/overview/components/

7 Customization by Deployment Pattern

The reading strategy should be tailored based on your target deployment pattern. The following sections outline prioritization adjustments for common scenarios.

7.1 Single Cloud, Single Cluster

Priority Adjustments:

- **Increase**: Chapter 6 (containerization depth), Chapter 10 (access security)
- **Decrease**: Chapter 15 (specialized architectures), multicloud sections of Chapter 13
- **Focus**: Getting fundamentals right before scaling complexity

7.2 Single Cloud, Multi-Cluster

Priority Adjustments:

- **Increase**: Chapter 14 (advanced architectures, especially clustering and failover), Chapter 9 (specialized mechanisms)
- **Focus**: Cluster federation, workload distribution, consistent policy across clusters

7.3 Multi-Cloud or Hybrid

Priority Adjustments:

- **Increase**: Chapter 4 (deployment models), Chapters 13 and 15 (multicloud and federated architectures), Chapter 17 (cost management across providers)
- **Focus**: Portability tradeoffs, unified observability, cross-cloud networking

7.4 Regulated Environment (HIPAA, PCI-DSS, SOC 2)

Priority Adjustments:

- **Increase**: Chapter 7 (security and cybersecurity), Chapters 10–11 (security mechanisms), Chapter 14 (data sovereignty)
- **Focus**: Audit logging, encryption requirements, compliance controls, data residency

8 Outcome-Based Reading Targets

After completing the recommended reading, you should be able to define and implement the following:

8.1 After Chapter 4 and 7

- Define trust boundaries and their implementation in namespaces and RBAC
- Articulate shared responsibility boundaries for your deployment model
- Document threat model vocabulary for architecture decisions

8.2 After Chapter 6

- Design container image build pipelines with layering optimization
  See: https://docs.docker.com/build/cache/
- Implement multi-container pod patterns (sidecar, ambassador, adapter)
  See: https://kubernetes.io/docs/concepts/workloads/pods/init-containers/
- Configure container networking and service discovery
  See: https://kubernetes.io/docs/concepts/services-networking/service/

8.3 After Chapters 8–11

- Design VPC/network segmentation with security groups
  See: https://developer.hashicorp.com/terraform/tutorials/aws/aws-vpc
- Implement IAM policies and RBAC configurations
  See: https://kubernetes.io/docs/reference/access-authn-authz/rbac/
- Configure encryption (at rest, in transit) and secrets management
  See: https://kubernetes.io/docs/concepts/configuration/secret/
- Deploy monitoring, logging, and audit infrastructure

8.4 After Chapters 13–14

- Design autoscaling policies (HPA, VPA, cluster autoscaler)
  See: https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale-walkthrough/
- Implement disaster recovery and failover strategies
- Configure load balancing and traffic distribution
  See: https://kubernetes.io/docs/concepts/services-networking/ingress/
- Plan zero-downtime deployment strategies
  See: https://kubernetes.io/docs/concepts/workloads/controllers/deployment/

8.5 After Chapters 16–18

- Define SLOs/SLIs with error budgets
- Implement cost allocation and FinOps controls
- Configure Terraform state backends with appropriate locking and encryption
  See: https://developer.hashicorp.com/terraform/language/settings/backends/s3
- Document operational runbooks aligned with SLA requirements

---

End of Document