Kubernetes Enterprise Readiness Checklist

Comprehensive guide of 77 factors to evaluate and implement production-grade Kubernetes management across technical, operational, and financial dimensions

Engineering Optimization *(14 items)*

- ❑ Rightsizing workloads and infrastructure including requests and limits
- ❑ Autoscaling, with traffic prediction including HPA & VPA, event-driven autoscaling (e.g., KEDA)
- ❑ Adaptability to new releases
- ❑ Node provisioning optimization (static pools, Karpenter)
- ❑ Set resource quotas and namespace limits
- ❑ Set pod disruption budgets
- ❑ Set pod topology spread constraints
- ❑ Set network policies and security configurations
- ❑ Determine storage class optimization
- ❑ Set image optimization and container security policies
- ❑ Set multi-tenancy parameters
- ❑ Determine cluster upgrade strategies
- ❑ Set idle resource management policies
- ❑ Anomaly detection

Financial Optimization (8 items)

- ❑ Buy Reserved Instances / CUD
- ❑ Establish savings plans
- ❑ Set spot instance strategies
- ❑ Set idle resource management policies
- ❑ Determine dev/staging environment scheduling
- ❑ Set resource cleanup policies
- ❑ Plan scheduled shutdowns
- ❑ Billing anomaly detection

Availability (10)

- ❑ High availability design
- ❑ Failover testing
- ❑ Circuit breaking
- ❑ Rate limiting
- ❑ Load testing
- ❑ Chaos engineering
- ❑ Multi-zone vs single-zone
- ❑ Data consistency
- ❑ Backup strategies
- ❑ Disaster recovery plan

Infrastructure Foundation (4)

- ❑ Management & Metrics - Infrastructure
- ❑ Logging & Observability Platform
- ❑ SLOs & SLA Framework
- ❑ RBAC and security controls

Operational Enablers (11)

- ❑ Backup Infrastructure and Systems
- ❑ GitOps Tooling and Templates
- ❑ CI/CD Pipeline Integration
- ❑ Service Mesh Implementation
- ❑ Topology visualization tools
- ❑ Management reporting policies
- ❑ Backup solutions
- ❑ Compliance and audit requirements
- ❑ GitOps practices
- ❑ CI/CD pipeline integration
- ❑ Service mesh considerations

Cloud Specific Factors (10)

AWS

- ❑ EKS or other distribution
- ❑ ASG (auto scaling groups)
- ❑ EKS Auto Mode
- ❑ Choose EC2 vs Fargate

Azure

- ❑ AKS or other distribution
- ❑ Azure VM Scale Sets integration

GCP

- ❑ GKE or other distribution
- ❑ GKE Cluster Autoscaler
- ❑ GKE Node Auto-provisioning
- ❑ GKE Autopilot

Architecture & Integration (6)

- ❑ Cloud-native service integration (databases, caching, messaging)
- ❑ Identity and access management
- ❑ Network architecture (VPC, subnets, ingress/egress)
- ❑ Load balancer options
- ❑ Storage (block, file, object)
- ❑ Other managed services integration

Monitoring/Management (14)

Documentation and Governance

- ❑ Documentation Requirements
- ❑ Training and Knowledge Sharing
- ❑ Change Control Processes
- ❑ Compliance and Audit Management

Operational Processes

- ❑ Monitoring Implementation and Alerts
- ❑ Incident Response and Management
- ❑ Performance Testing and Optimization
- ❑ Capacity Planning and Forecasting
- ❑ New Release Scoring

Maintenance Activities

- ❑ Upgrade and Patch Management
- ❑ General System Maintenance
- ❑ Backup Execution and Verification
- ❑ Disaster Recovery Policies & Testing

Cost and Chargeback Management

- ❑ Chargeback and showback
- ❑ Cost allocation

Kubernetes Management Maturity Spectrum

Level 0: Manual Operations	Level 1: Basic Automation	Level 2: Standardized Automation	Level 3: Intelligent Operations (AI Copilot)	Level 4: Autonomy (AI Autopilot)	Level 5: Advanced Autonomy
• All cluster management tasks performed manually • Manual scaling and resource allocation • Manual troubleshooting and incident response • No automation for routine tasks	• Monitoring tools in place with basic alerting • Simple scripts for common tasks • HPA/VPA for basic workload scaling • Manual intervention required for most operations	• GitOps workflow implementation • Automated CI/CD pipelines • Rule-based autoscaling for workloads • Predefined responses to common incidents • Automated backup and recovery processes	• AI-assisted anomaly detection • Predictive scaling based on historical patterns • Automated remediation for known issues • Self-healing capabilities for common failures • Human approval required for significant changes	• Automated cost optimization • AI makes autonomous scaling decisions • Self-tuning performance parameters • Predictive maintenance and upgrades • Human oversight primarily for exceptions	• AI manages complete cluster lifecycle • Autonomous resource optimization across clusters • Self-evolving policies based on performance analysis • Automatic adaptation to new workload patterns • Cross-cluster resource balancing and optimization