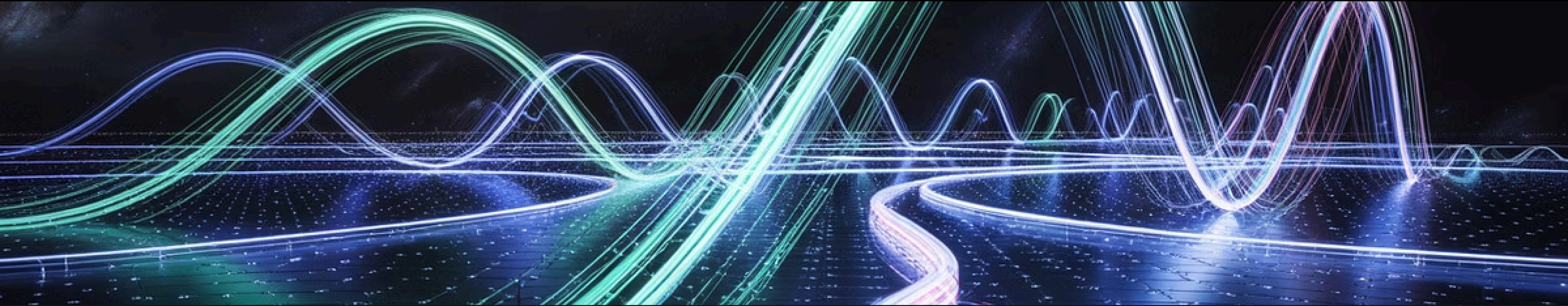
Scaling Kafka on Kubernetes

Cloud-Native Streaming for 100,000+ Messages Per Second

Shalini Katyayani Koney
Walmart
Conf42 Kube Native



The Challenge

Unprecedented Data Volume

Containerized applications generating millions of messages per second requiring real-time processing

Cloud-Native Requirements

Need for Kubernetes-native streaming platforms that scale dynamically with workload demands

Performance at Scale

Maintaining low latency and high throughput while managing stateful streaming workloads

Why Kafka on Kubernetes?

Container Orchestration

Leverage Kubernetes' native capabilities for automatic scaling, self-healing, and resource management of Kafka clusters

Cloud-Native Benefits

Seamless integration with cloud services, persistent storage, and service mesh architectures

Dynamic Scaling

Horizontal pod autoscaling adapts to traffic patterns while maintaining streaming performance

Architecture Foundation

StatefulSet Configuration

Kafka brokers deployed as StatefulSets ensuring ordered deployment, stable network identities, and persistent storage

Persistent Volume Strategy

Optimized storage classes for Kafka logs with high IOPS and low latency requirements

Service Mesh Integration

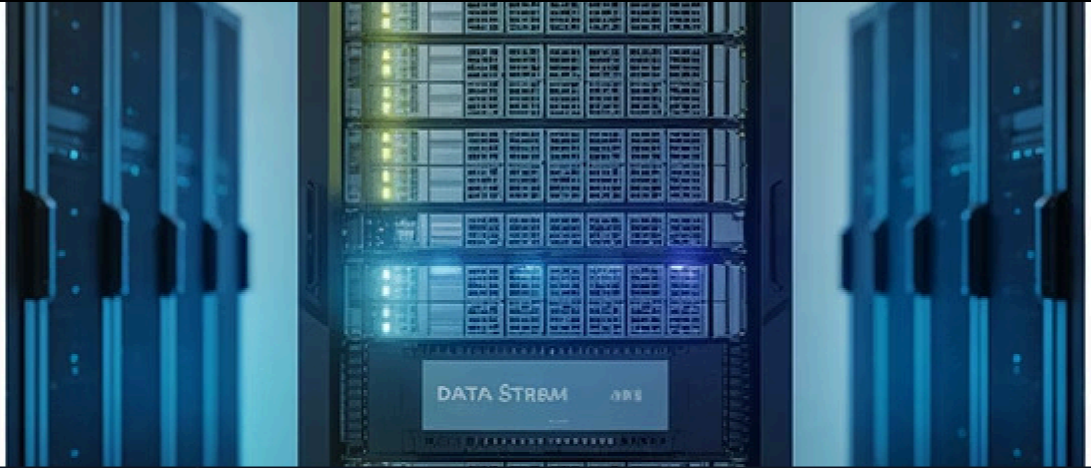
Secure inter-pod communication with traffic management and observability

Pod Resource Allocation

CPU & Memory Strategy

- CPU requests: 2-4 cores per broker
- Memory: 8-16GB heap size optimization
- Resource limits prevent noisy neighbors
- QoS class: Guaranteed for critical workloads

Proper resource allocation ensures consistent performance under varying loads while preventing resource contention in multi-tenant clusters.



Storage Optimization

1

Storage Classes

SSD-backed storage classes with high IOPS for log segments and low-latency access patterns

2

Volume Configuration

Persistent volume claims sized for retention policies and replication factors

3

Performance Tuning

File system optimizations and mount options for maximum throughput

Network Policies & Performance

1. **Network Segmentation**

   Isolated network policies for Kafka clusters ensuring security without performance degradation

2. **Load Balancing**

   Service configurations optimized for streaming workloads with session affinity

3. **Bandwidth Optimization**

   Network performance tuning for high-throughput message processing

Horizontal Pod Autoscaling

Custom Metrics

HPA based on Kafka-specific metrics like consumer lag and partition throughput

Scaling Policies

Intelligent scaling algorithms that account for stateful nature of Kafka brokers

Traffic Adaptation

Dynamic adjustment to varying message volumes while maintaining partition balance

Multi-Cluster Deployment

Helm Charts

Standardized deployments across environments with configurable parameters

Global Distribution

Cross-region replication for disaster recovery and reduced latency

Data Consistency

MirrorMaker 2.0 for reliable cross-cluster data synchronization

GitOps Workflow

Infrastructure as Code

Kafka cluster configurations managed through Git repositories with version control

Automated Deployments

CI/CD pipelines triggering cluster updates based on configuration changes

Rollback Capabilities

Safe deployment practices with automated rollback on configuration errors

Monitoring & Observability

Monitoring Stack

- **Prometheus Integration**

  JMX metrics exported for comprehensive cluster monitoring

- **Grafana Dashboards**

  Real-time visualization of throughput, latency, and resource utilization

- **Alerting Rules**

  Proactive notifications for performance degradation and capacity thresholds

Performance Metrics

100K+

Messages/Seco nd

Sustained throughput with optimized configurations

<5ms

Latency P99

Low latency maintained under high load

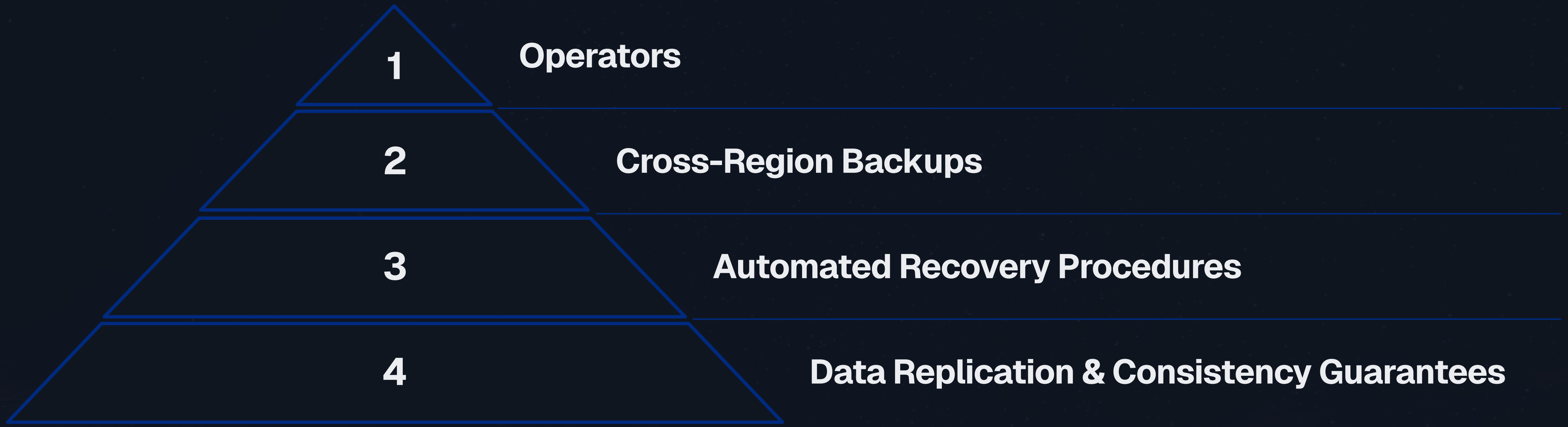
99.9%

Availability

High availability with zero-downtime deployments

These metrics demonstrate the effectiveness of cloud-native Kafka deployments in production environments handling demanding streaming workloads.

Disaster Recovery


1
Operators
2
Cross-Region Backups
3
Automated Recovery Procedures
4
Data Replication & Consistency Guarantees

Kubernetes operators provide automated backup, restoration, and failover capabilities ensuring business continuity for critical streaming applications.

Key Takeaways

Strategic Planning

Proper resource allocation and storage optimization are critical for sustained high-throughput performance

Cloud-Native Approach

Leverage Kubernetes-native patterns for scaling, monitoring, and operational excellence

Production Ready

Implement comprehensive monitoring, GitOps workflows, and disaster recovery from day one

Thank You

Shalini Katyayani Koney
Walmart
Conf42 Kube Native