

1.

- Serverless solves: Eliminates infrastructure management, provides automatic fine-grained scaling, charges only for execution time.
- Traditional microservices: Require constant resource provisioning, manual scaling configuration, and pay for idle resources.
- Serverless better: Event processing with sporadic traffic - like image thumbnailing on upload. Saves cost by scaling to zero.
- Serverless worse: High-traffic consistent workloads - like video streaming. Cold starts and execution limits hurt performance.

2. Beyond Kubernetes networking: Advanced traffic control (canary, circuit breaking), detailed observability (tracing, metrics), automatic mTLS encryption, protocol support beyond HTTP.

3. Intercepts all network traffic to/from application container. Provides: traffic management, security policies, observability data collection. Needed to separate operational concerns from business logic.

4.

- Features: Weighted routing, retries, circuit breaking, fault injection, mirroring.
- Production use: Canary deployments - route 1% traffic to new version; Circuit breaking – stop sending requests to failing services.

5. Knative uses Kubernetes HPA with custom metrics and can scale to zero. Scales based on request queue length/concurrency. Up: When requests exceed current capacity. Down: After sustained no traffic (default 60s), scales to zero pods

6. Manages event delivery between sources and consumers. Supports event-driven architectures with: multiple event sources (Kafka, HTTP),

event routing, filtering, transformation, reliable delivery.

7.

- Knative leverages Kubernetes primitives to provide serverless experience through these abstractions
- Hides: Deployments, Services, HPA, Ingress.
- Developers just provide container image. Benefits: No infrastructure knowledge needed, automatic scaling and networking.

8. Unified interface for ML model deployment. Simplifies: automatic scaling, traffic splitting, model framework configuration, standardized endpoints.

9.

Data flow:

- Istio/Istio Gateway - Handles external traffic routing, TLS termination
- Knative - Manages autoscaling, request queuing
- KServe - Model serving, format transformation
- Kubernetes - Container orchestration, resource allocation
- Bottlenecks: Cold starts, model loading, network hops between layers, resource limits.

10.

- Weighted routing for gradual rollouts (5%, 10%, 25%), retries for temporary failures, circuit breaking for bad versions.
- Pros vs manual: Precise traffic control, instant rollback, real-time metrics.
- Cons: Configuration complexity, resource overhead, operational learning curve