**Pod Kill with Delayed Start**  
  
This experiment simulates **a pod failure followed by incremental network latency** affecting the ledger writer service in the bank-of-anthos namespace.

The steps are:

1. **Pod Kill:** A ledger writer pod is forcefully killed.
2. **Network Latency 80ms (2 min):** Introduces **80ms delay** in network traffic.
3. **Network Latency 60ms (2 min):** Reduces latency to **60ms** after the previous delay.
4. **Network Latency 40ms (2 min):** Further reduces to **40ms**.
5. **Network Latency 20ms (2 min):** Finally, latency drops to **20ms** before normal operation.

This **delayed start** means the application first faces a pod failure, and as it recovers, it gradually encounters decreasing network delays.

**Why Are We Using These?**

This test helps evaluate:  
✅ **Application Resilience:** How ledger writer handles pod failures and network slowdowns.  
✅ **Network Stability:** Impact of high-to-low latency changes on service performance.  
✅ **Recovery Handling:** Whether the service can auto-recover without manual intervention.  
✅ **User Experience:** Effects of pod failures and network slowness on end-users (e.g., transaction delays in a banking app).

**Path: /root/Kalyani/** **pod-kill-delayed-start  
  
How to Eradicate These Issues?  
1. Mitigating Pod Kill Issues**

* High Availability (HA): Use multiple replicas for ledger writer in a Deployment or Stateful Set.
* Pod Disruption Budgets (PDBs): Prevent all instances from being killed at once.
* Liveness/Readiness Probes: Ensure Kubernetes restarts unhealthy pods.
* Horizontal Pod Autoscaler (HPA): Scale up pods during high load.

**2. Handling Network Latency**

* Istio Traffic Management: Use Istio Circuit Breakers and Retries to handle delayed responses.
* CDN or Caching: Reduce dependency on real-time network calls.
* Optimized Load Balancing: Configure Service Mesh (Istio/Linkerd) to reroute traffic dynamically.
* Network Policies & QoS: Prioritize critical requests over non-essential traffic.

**3. Continuous Monitoring & Alerting**

* Prometheus & Grafana: Monitor latency and pod restarts.
* Chaos Engineering Pipelines: Automate tests using tools like Chaos Mesh and detect weak points.