XPipe: A Decentralized, Token-Orchestrated Pipeline Architecture

Abstract

This document outlines the architecture of **XPipe**, a decentralized PaaS designed to support and orchestrate distributed CI/CD pipelines without relying on traditional API Gateway infrastructure. The system employs secure, token-based communication for direct service invocation and leverages OpenVMS as a transactive orchestration layer to manage pipeline state consistency, diagnostics, and rollback strategies.

1. Introduction

XPipe is a modular service mesh architecture designed to offer fault tolerance, high throughput, and secure communication in a Kubernetes environment. The platform integrates RKE2 as its orchestration layer and builds on Ingress-managed endpoints secured by direct service-level tokens.

2. Architecture Overview

The infrastructure is built on:

- RKE2 Kubernetes Cluster with 3 worker nodes + 1 control plane
- Cilium CNI
- Longhorn storage (RAID-1 SSD per node)
- Round Robin DNS with public ingress controller exposure

3. Token Authentication Layer

Each microservice is secured by a statically defined token that must be passed via the `X-Token-XPipe` header. These tokens serve as private keys and are validated against internal secrets stored per-service.

4. CI/CD Flow Integration

As depicted below, XPipe integrates Git-based tagging with pipeline triggering, quality inspection (via SonarQube), and container image management with JFrog. Configuration data is retrieved from CMDBuild.

CI/CD Pipeline Diagram

5. OpenVMS: Orchestration & Transactional Ledger

A key innovation is the delegation of feedback control and pipeline orchestration to an OpenVMS node, interfaced over TCP. When a pipeline is initiated:

- OpenVMS logs the execution context atomically
- Tracks probe signals (liveness/readiness)
- Automatically triggers rollback or diagnostic pods on failure

This provides resilience akin to database-level transactions in pipeline orchestration, with unmatched auditability.

6. Benefits

- No API Gateway = no central point of failure
- Full horizontal scalability
- Audit trail and recovery via OpenVMS
- Secure token-based routing across services

7. Considerations

This architecture assumes the client can cache ACLs after login, and token revocation requires a new session. Session consistency is ensured via sticky ingress and Redis-based state backend.

8. Conclusion

XPipe demonstrates how modern container-native systems can integrate legacy-proven technologies like OpenVMS to build secure, resilient, and scalable PaaS architectures.