DMaaP Architecture Evolution

Ciaran Johnston
Fiachra Corcoran

2021-09-07

Contents

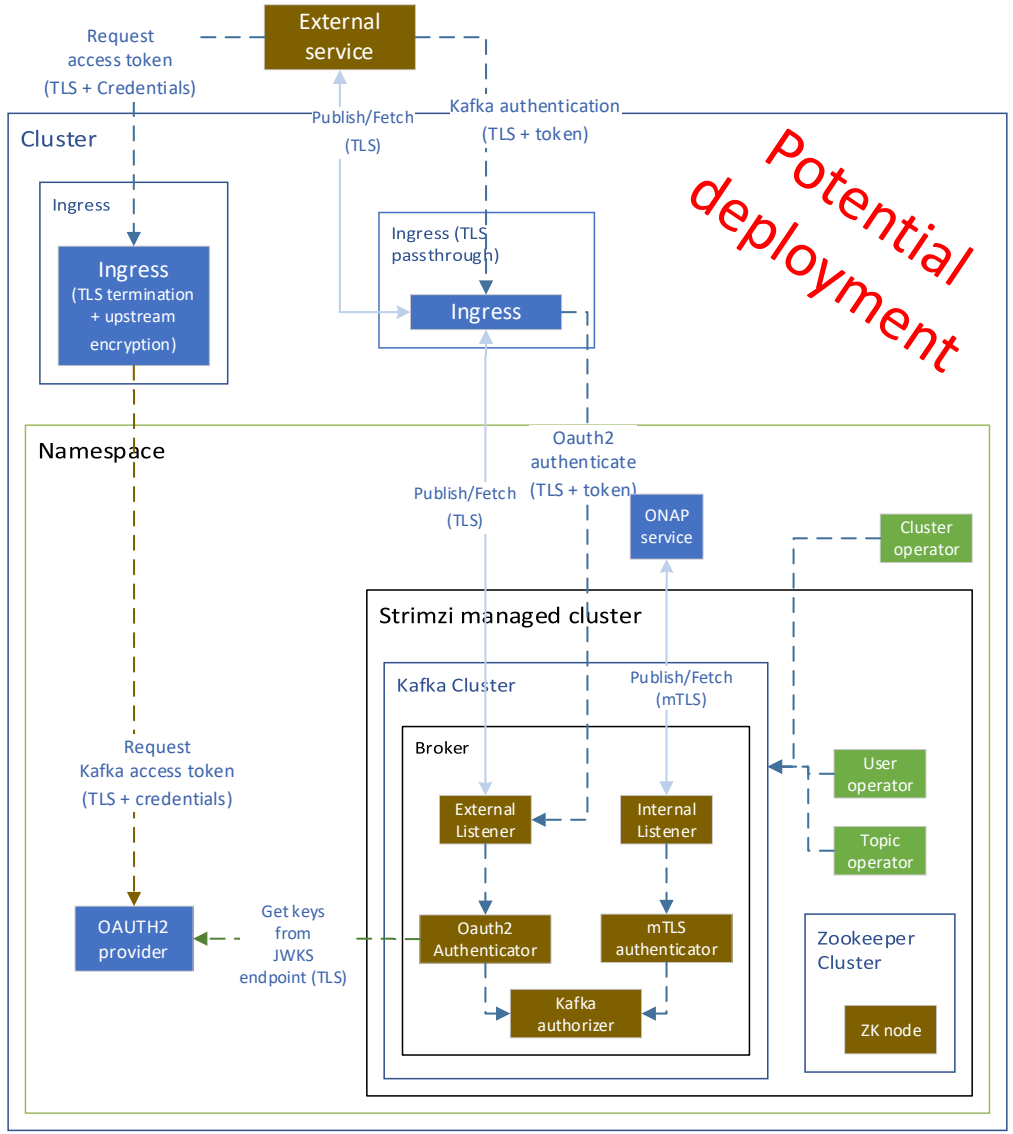
- Problem Description
- Introduction of Operator-based Deployment
- Introduction of Event Bridge Component
- Strimzi vs Alternatives
- Evolution and Deprecation Strategy
- Discussion / Next Steps

Problem Description

- DMaaP is a central component in the ONAP architecture, forming the backbone of the communications infrastructure
  - 60+ topic interactions across 7+ projects
- There are currently 3 team members and 2 committers in the DMaaP development team
  - Primarily Ericsson developers, with little broader community engagement
- Codebase is old, complex and contains lots of features no-one uses
  - Lots of it is effectively unmaintainable due to lack of users or knowledge
  - Significant number of unpublished dependencies – binaries available in Nexus, but no source code available for a subset of these components: https://mvnrepository.com/artifact/com.att.nsa
  - This is hindering / preventing adoption of Java 11
  - Significant concern for software governance / licensing compliance
  - Tight coupling to Zookeeper, including to share API keys between projects – significant security issue under investigation
- The value in ONAP is on the differentiating capabilities for Network Automation
- There are other open-source alternatives for doing message routing which we can reuse instead
  - We can rely on and work with the broader community to build specific features of interest to ONAP rather than building and maintaining a bespoke solution

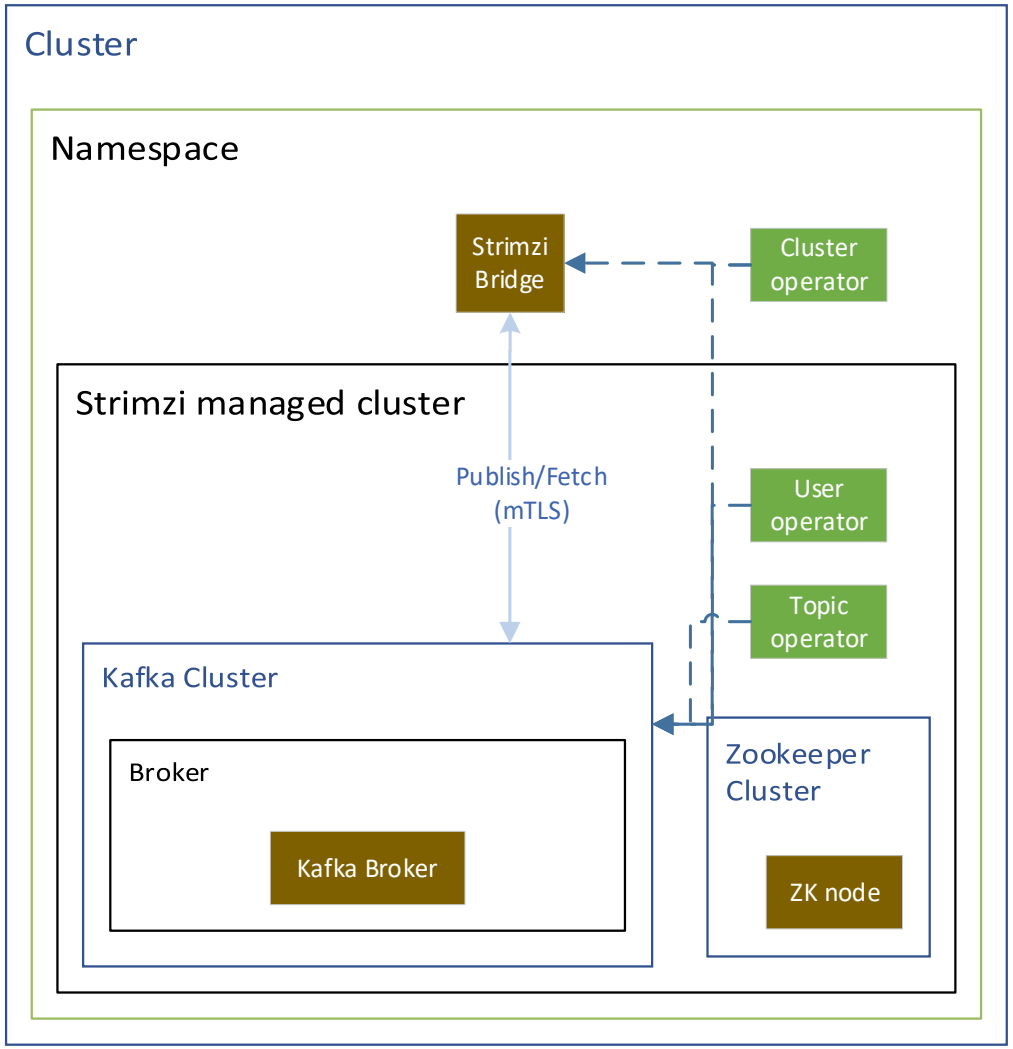
Introduction of Operator-based Deployment

- What
  - Strimzi Kafka operator
  - Apache licensed, CNCF sandbox project
- Why?
  - Kafka configuration is complex, maintaining mapping in helm is tedious and expensive
  - Declarative scale out
  - Declarative K8 user and topic management
  - It supports additional parts of the Kafka ecosystem (e.g. Kafka connect, Mirrormaker)
  - We can take advantage of additional Kafka features (e.g. OAUTH based authentication, multiple security models on the same cluster)
  - It supports Kafka native exposure
- How?
  - Deploy the Kafka operator via helm
  - Deploy Kafka clusters via CRs
  - Topic and user operators are deployed *per cluster* by the cluster operator



Introduction of Event Bridge Component

- What
  - New REST interface for Kafka access to potentially replace DMaaP Message Router (MR)
- Why
  - Simple codebase, community driven with an active developer base – reduced maintenance for ONAP
  - Preserves Kafka semantics (at-least-once delivery guarantees, no data loss) which MR does not
  - Manages Connectivity and security towards Kafka declaratively using CRs
- How
  - Just tell the Strimzi operator to spin it up as part of OOM DMaaP charts
  - Leverages API GW or service mesh for security
- Implications
  - Alternative REST API with improved semantics in ONAP
  - Eventual deprecation and removal of Message Router from ONAP codebase
  - Topic provisioning moves to kubectl operators rather than auto-creation in MR (seen as bad practice anyway)



Strimzi vs Alternatives

- There are a number of alternatives for REST Bridge
  - Confluent REST Proxy, Kafka-pixy (designed as a sidecar rather than a bridge)
- These alternatives have pros and cons (referenced here)
  - On balance, coupled with the fact that Strimzi manages deployment and administration through operators, it is the best option for ONAP
- Strimzi is a CNCF sandbox project

Evolution Strategy – Kafka Deployment & Management

- Ongoing discussions with OOM PTL – positive feedback on the approach of consuming a 3rd party helm chart for 3rd party dependencies
- PoC under way to verify the procedure (demo TBD)
- Plan to introduce in J release assuming architecture approval
- Expected to be fairly straightforward replacement of deployment procedure
- Some challenges on the Zookeeper dependency with MR to be worked out
  - Strimzi locks down ZK for strimzi-only use
  - MR is tightly coupled to Zookeeper for cluster communication, even in single-instance mode – do we need two instances of ZK?
  - Solutions under investigation
  - Potential move to Kafka 3.0 in Strimzi which removes dependency on ZK (Q1 2022)

Evolution and Deprecation Strategy – REST API

- Analysis on existing active DMaaP MR Clients in Istanbul:
  - https://wiki.onap.org/display/DW/Active+DMaaP+clients+in+Istanbul+Release
  - Input required from other PTLs to verify completeness
- Topic creation process required – e.g. templates in OOM for topic creation during ONAP / DMaaP installation
- Migration of publishers is relatively straightforward
- Migration of subscribers is more challenging
  - Multiple client libraries in use depending on the codebase
  - Potential discussion on releasing a mapping client for MR clients to use
  - Some joined-up planning across ONAP is required to complete the migration
- Eventual deprecation and removal of MR needs to be planned in across releases – e.g. deprecation in Jakarta, removal in London
  - Can be challenging to prioritize – Global Requirement? Some other approach?

Discussion and Next Steps

- DMaaP team would like to progress with the deployment migration analysis and execution

- DMaaP team would like input on the best approach to evolving towards a more maintainable and broader community-led API for REST