aiven

Scaling Kafka in the Cloud

All Things Open

Heikki Nousiainen
@hnousiainen
2019-10-14
Raleigh, NC

Speaker

- Heikki Nousiainen
- CTO, co-founder @ Aiven
- First contact with Apache Kafka in 2014

@hnousiainen

Google Cloud Platform

amazon web services

Microsoft Azure

UpCloud

DigitalOcean

packet

Aiven

- Based in Helsinki, Boston and Sydney
- 7 data engines now available in 6 clouds and 80 regions, virtual and bare metal instances
- Fully-managed Kafka Cloud service launched in 2016
- PostgreSQL, MySQL, Elasticsearch, Cassandra, Redis, InfluxDB

Google Cloud Platform

amazon web services

Microsoft Azure

UpCloud

DigitalOcean

packet

Kafka @ Aiven

- Managed Kafka Offering since 2016
- 500 Kafka Clusters under management
- Cluster sizes range from 3 to 30 brokers
- Over 10,000 services in total under management

Google Cloud Platform

amazon web services

Microsoft Azure

UpCloud

DigitalOcean

packet

Kafka @ Aiven

- We utilize Kafka as core message bus between all of our components
- Signaling, Metrics, Logs, Audit logs
  - Configuration/state communication
  - Logs from VMs to Kafka, from Kafka to Elasticsearch
  - Stats from VMs to Kafka, from Kafka to InfluxDB / M3

Google Cloud Platform

amazon web services

Microsoft Azure

UpCloud

DigitalOcean

packet

Kafka @ Aiven

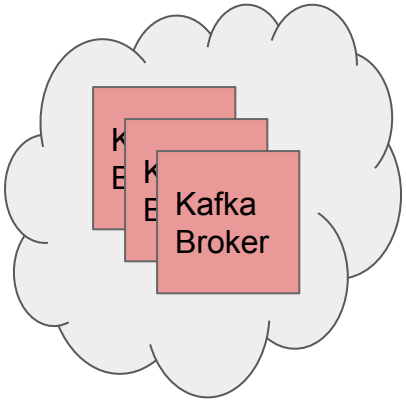
- Decoupled architecture
  - Handle various kinds of network splits
  - Avoid blocking / waits on the producer side
- Scale
  - Kafka scales near linear by number of brokers
  - Adjustable number of consumer processes
- One interface, multiple use cases
- One set of data, multiple use cases
- Security: ACL controls over read/write access

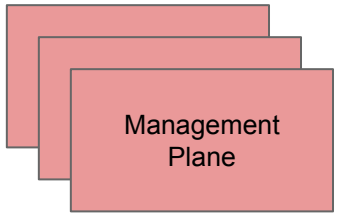
Aiven Architecture - principles

- Everything is code. Aiven is a software product.
- Automation to handle full lifecycle operations: deployment, operations and fault response.
- Converging operation: code works towards to the desired service state until established.
- Immutable infrastructure: ensure known state by enforcing rolling upgrades.

Aiven Architecture

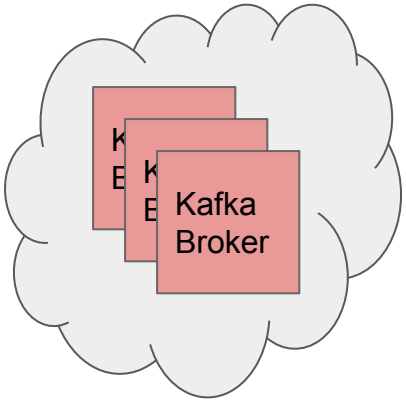
- Kafka brokers implemented as dedicated VMs
- Backed by management layer responsible for provisioning the cluster resources

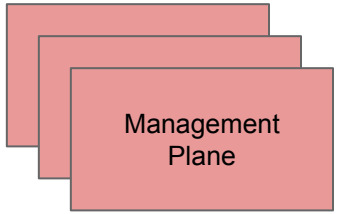




Aiven Architecture

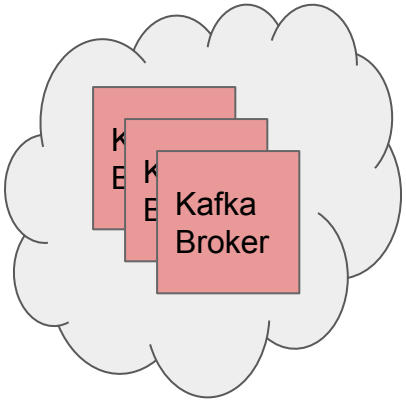
- Cluster members managed by agent software running within VM
- Per cluster ZK co-hosted on VMs
- Agent responsible for setting up and managing both ZK and Kafka
- Immutable; software updates via VM replacement

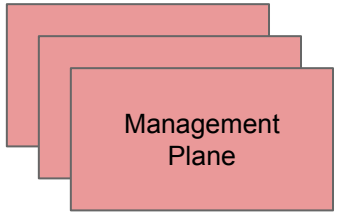




Aiven Architecture

- Management layer provisions and provides resources to the cluster.
- Management layer monitors and ensures cluster meets the specification: number of brokers, size of brokers, correct software version, broker health
- Availability Zones for fault tolerance





Our Challenges & Constraints

- Six supported cloud providers
- Immutable infrastructure, roll-forward updates
- Keeping software up-to-date
- Adjustable cluster size and broker specs
- Migration between regions and clouds
- High availability, SLAs

- Changing instances, IP addresses
- Ephemeral disks

Networking

- Span of private networks vary between clouds.
- Migration across networks can be tricky: mixed private & public addressing.
- Kafka advertises addresses in protocol endpoints also for interbroker communication.

- We utilize IPv6 overlay network between cluster nodes
- Consistent addressing regardless of the location or network

Networking

- Point-to-point IPv6 tunnels over IPv4 IPSec connections
- Libreswan
- Managed with whack commandline client
- Dynamically select the best routing option between any two VMs

- All internode communication authenticated and encrypted

- Closely tracking Wireguard progress as well

ZooKeeper

- Distributed, consensus based synchronization & coordination service
- Used by Kafka for controller election and cluster state management

- Changes in ZK cluster used to require restarts
- ZooKeeper 3.5 supports dynamic reconfiguration
- Add/remove nodes, switch roles
- Released May 20th, 2019
- In use at Aiven for 3+ years

ZooKeeper

- Dynamic reconfig requires quorum

- Fault recovery adds new node(s), removes disappeared ones
- Roll-forward upgrade adds new node(s), removes obsoleted nodes

- Both failure and positive cases handled using the same procedure

Controlled Kafka Broker restarts

- There are more and more settings that can be configured dynamically
- Some Kafka settings require broker restart
  - inter.broker.protocol.version
  - auto.create.topics.enable
  - offsets.retention.minutes

- For availability, avoid restarting all brokers at the same time

Controlled Kafka Broker restarts

- Agent detects a change requiring broker restart
- ZooKeeper based locks ensure that a single broker restart is initiated and completed at any one time

Controlled Kafka Broker restarts

- Same protocol applies to automated error recovery
- Handle Kafka Broker misbehaviour:
  - Deadlocks
  - Memory leaks / thread exhaustion
  - ZK induced cluster state mismatches

- Detect and correct automatically

Active Partition Management

- Agents perform continuous monitoring of partition placement
- Automatic adjustment for:
  - Satisfying number of alive replicas
  - Satisfying data replication across zones
  - Balance storage usage
  - Balance partition leader assignments

- Controlled draining of decommissioned brokers
  - Maintain replication level, AZ spread
  - Ensure clients receive “Not Leader For Partition”

Active Partition Management

- Calculate and maintain optimal plan in ZK
- Detect deviations from the plan
- Select and request a set of partition reassignments
  - /admin/reassign_partitions
  - Throttle amount of ongoing requests
  - Optimize & balance broker to broker links

- Currently coupled with our agent logic
- I'm looking to split & Open Source as a standalone daemon

Kafka Cluster operations

- Loss of Kafka Broker is detected by management plane
- New VM resource is created and joined into the cluster created
- A new Broker ID assigned

- Agent adjusts the ZK cluster memberships
- Partition placement uses the new resource and restores replication levels

Kafka Cluster operations

- Management plane detects:
  - Required security software updates
  - Requested Kafka version updates
  - Mismatching VM types - scaling
  - Mismatching VM number - scaling
  - Mismatching VM locations - migration

- Create new VMs
- Partition manager drains outgoing brokers - data migration
- Old brokers recycled

Monitoring

- JMX metrics via Jolokia
- ZooKeeper keys and values
- Logs
- Health checks, metadata calls

- Utilized by the local agent for decision making and corrective actions
- Streamed with Telegraf to central processing using Kafka

Monitoring

- JMX metrics via Jolokia
  - UnderReplicatedPartitions (*)
  - IsrShrinksPerSec
  - RequestHandlerAvgIdlePercent / NetworkProcessorAvgIdlePercent
  - LogCleanerManager:time-since-last-run-ms

- For topics / partitions with stable workloads, monitor & alert on high/low messages / sec bands

Monitoring

- ZooKeeper keys and values
  - /admin/reassign_partitions
  - /brokers/topics/...

- E.g. /admin/reassign_partitions update age
- Sometimes Kafka gets stuck:
  - Ever tries to implement reassignment on a broker that's gone forever
  - Failed partition leadership assignments

- On timeout, trigger Kafka controller re-election

Monitoring

- Logs
  - Exceptions & Errors
  - ReplicaFetchError
  - LeaderAndIsr requests
  - Log Truncations
  - Derived metrics, errors / minute

- Can be quite noisy
- Local decisions, alerts on metrics, centralized collection

- Journald to Kafka to Elasticsearch

Keeping up with Kafka versions

- We've used Kafka since version 0.8.2
- Extremely good backwards compatibility support
- Quality steadily improving:
  - Memory Leaks
  - Resource starvation
    - Deadlocks
    - Thread exhaustion (Exceptions)
  - Falling out of sync with the cluster
  - 2.1 => TLA+ modeled replication protocol improvements
- Generally a lot less broker restarts / forced controller re-elections

Break Things!

- Kafka is critical component to our infrastructure
- Practice makes perfect

- Terminate broker & ZK processes & VMs
- Induce network splits and errors
- Destroy volumes
- Spot & pre-emptible instance types

- Verify availability and capability to operate clusters

Thank you!

@hnousiainen
htn@aiven.io

https://aiven.io