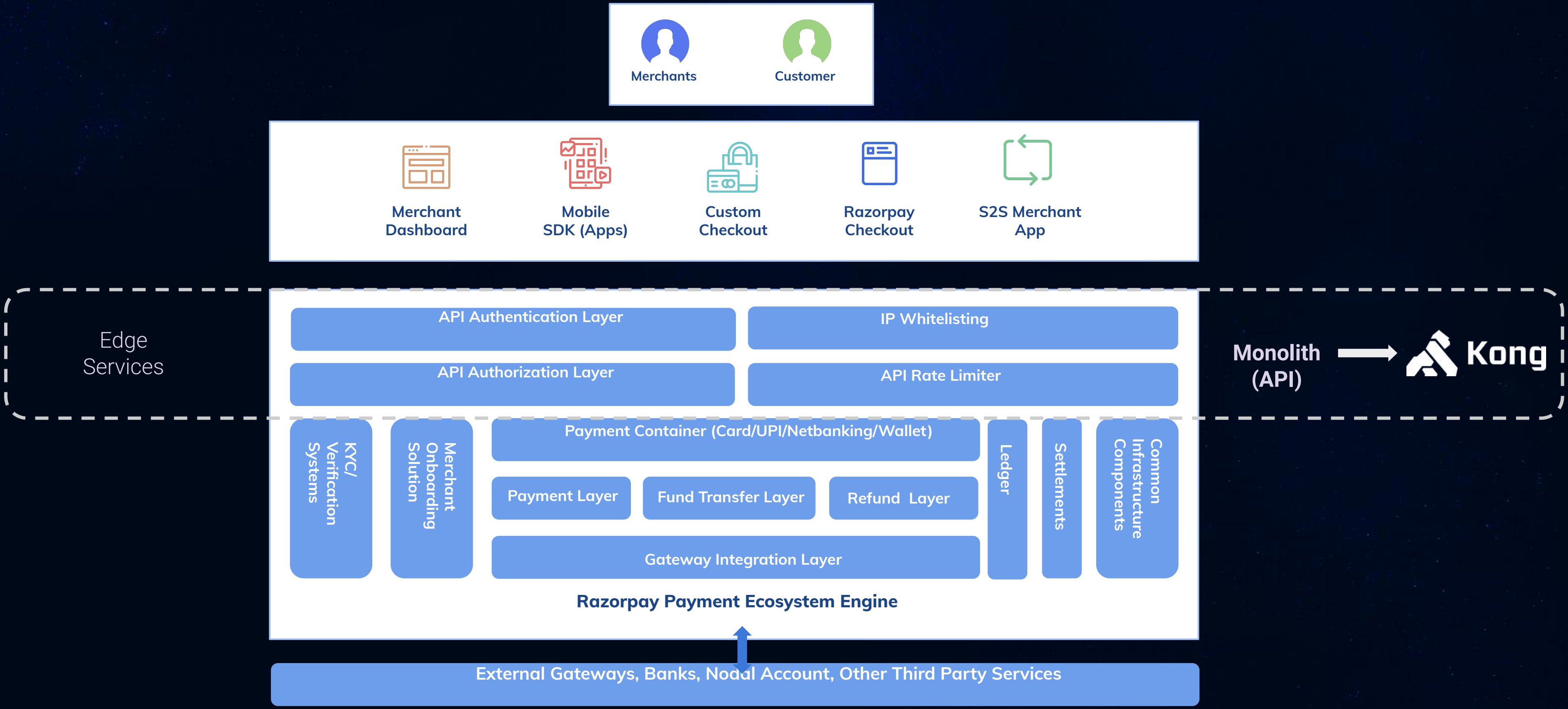
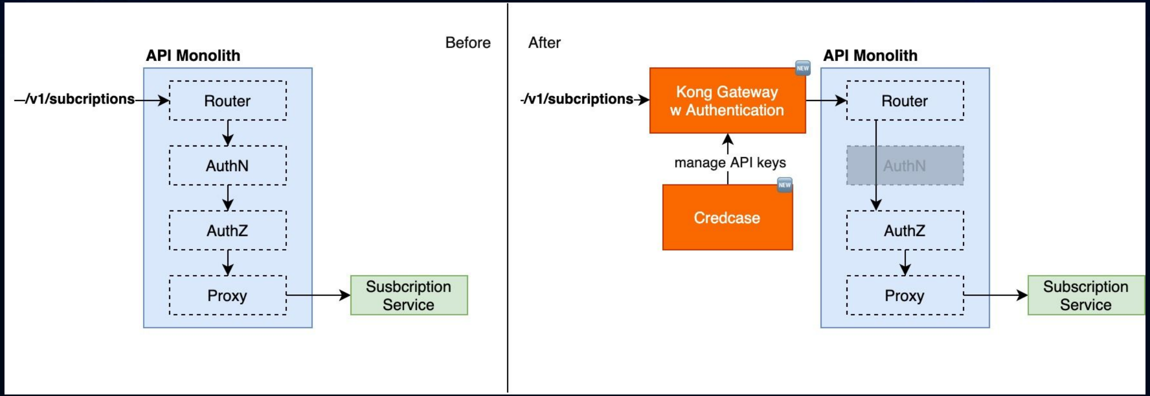
Scaling Authentication and Authorization for a million merchants using Kong

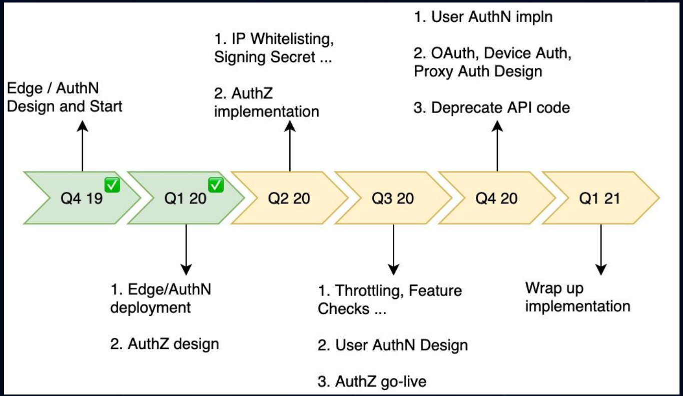
The Problem Statement



Edge Services - Current State and Roadmap

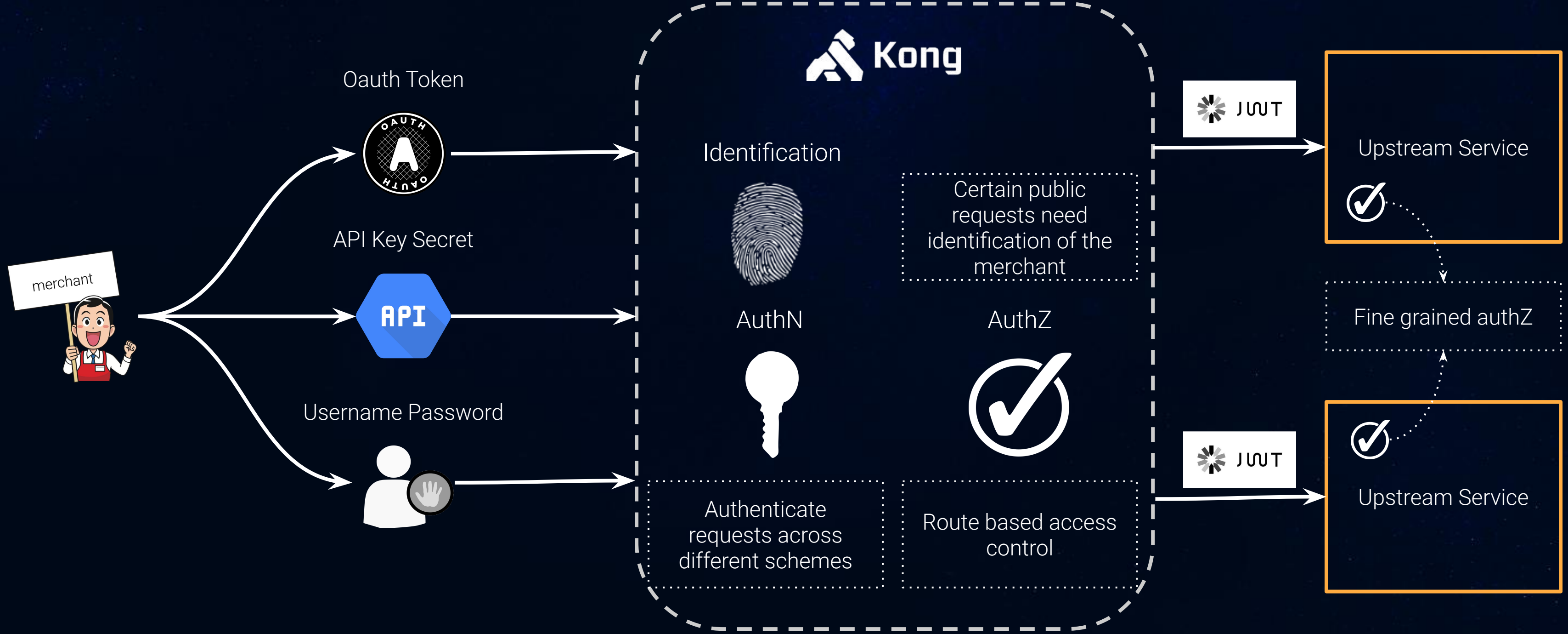


Current State

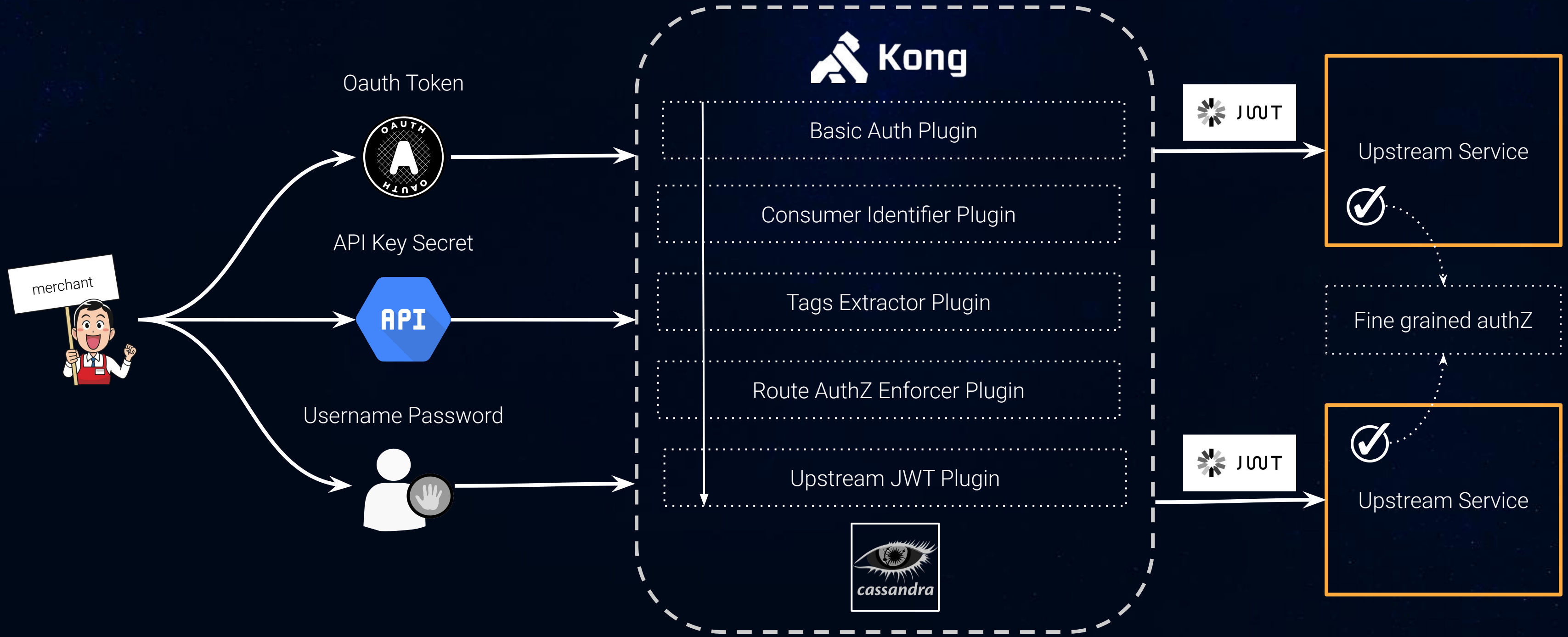


Roadmap

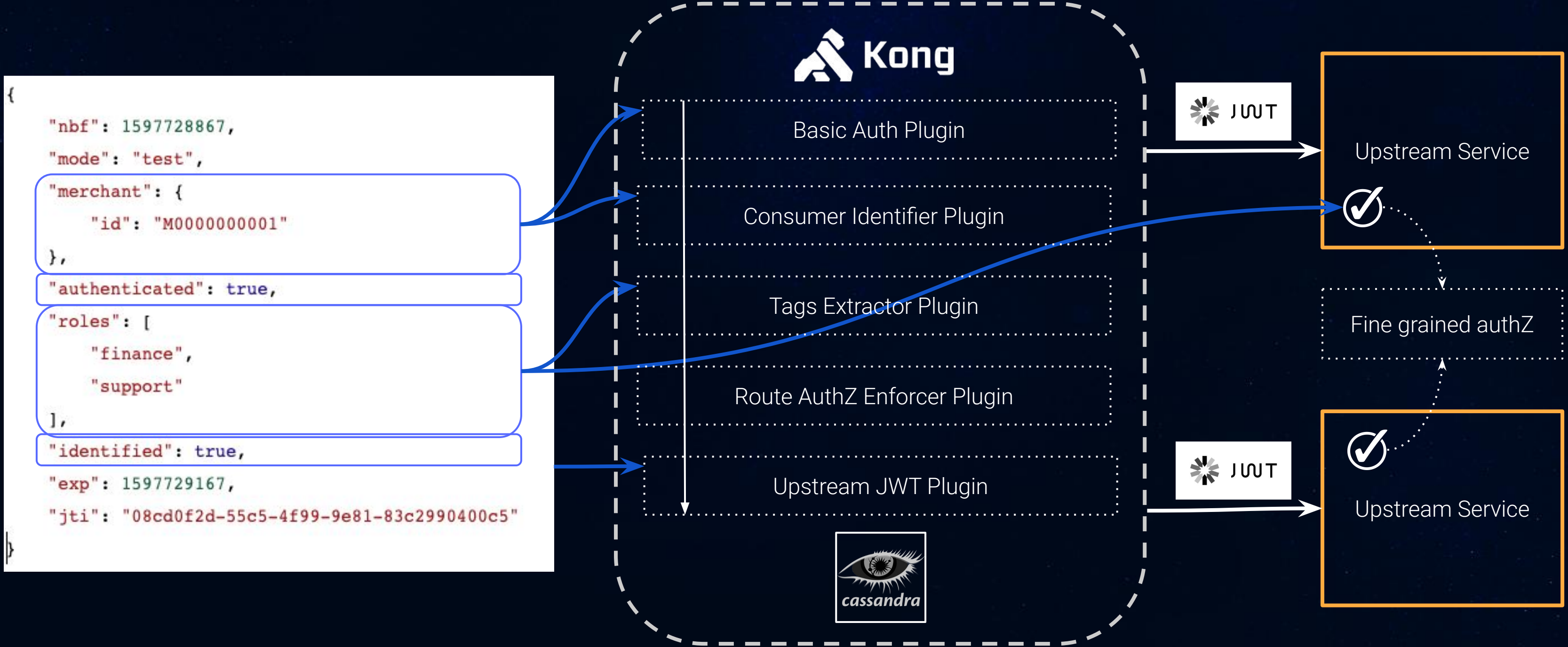
Authentication, Identification and Authorization



Custom Plugins



The JWT Token



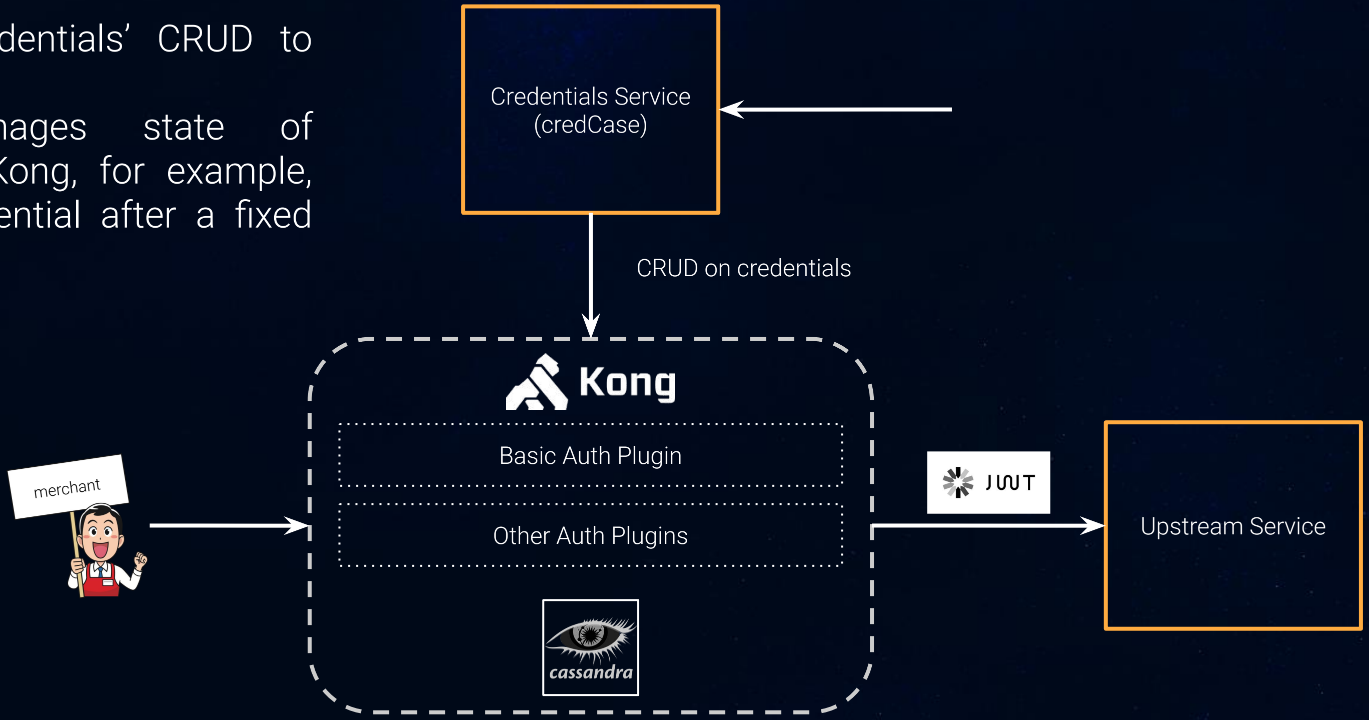
```
{
    "nbf": 1597728867,
    "mode": "test",
    "merchant": {
        "id": "M0000000001"
    },
    "authenticated": true,
    "roles": [
        "finance",
        "support"
    ],
    "identified": true,
    "exp": 1597729167,
    "jti": "08cd0f2d-55c5-4f99-9e81-83c2990400c5"
}
```

1

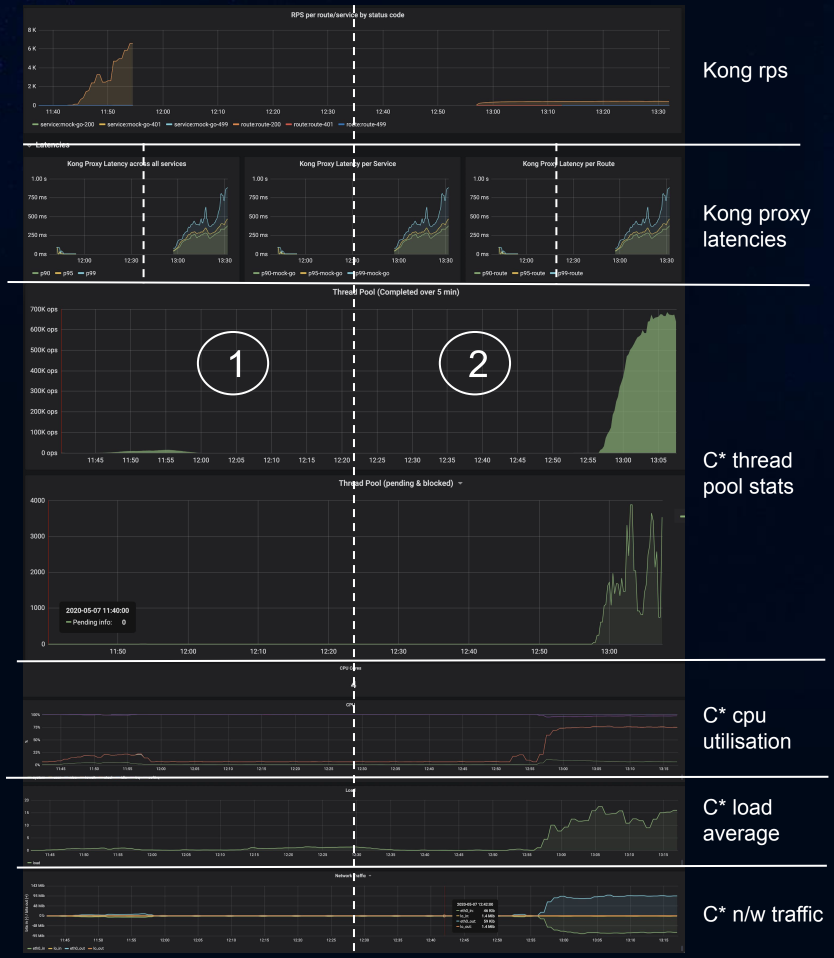
Authentication

Authentication

- Credentials Service is the control plane to manage various types of credentials.
- It exposes credentials' CRUD to other services
- It also manages state of credentials in Kong, for example, expiring a credential after a fixed duration.



Stress Testing Basic Auth Plugin in Kong



1	2
With basic-auth credentials warmed up in the cache	Without basic-auth credentials warmed up in the cache

- When basic-auth credentials were not cached, kong proxy latency spiked under load.
- There was also a spike in C* thread pools, cpu utilisation and n/w traffic.
- The issue was identified to be happening due to username being a secondary in Kong's C* schema, and C* is not efficient with secondary indexes.

Modifying basic-auth plugin with username as primary key fixed the performance bottleneck.

2

Authorization

Authorization Requirements

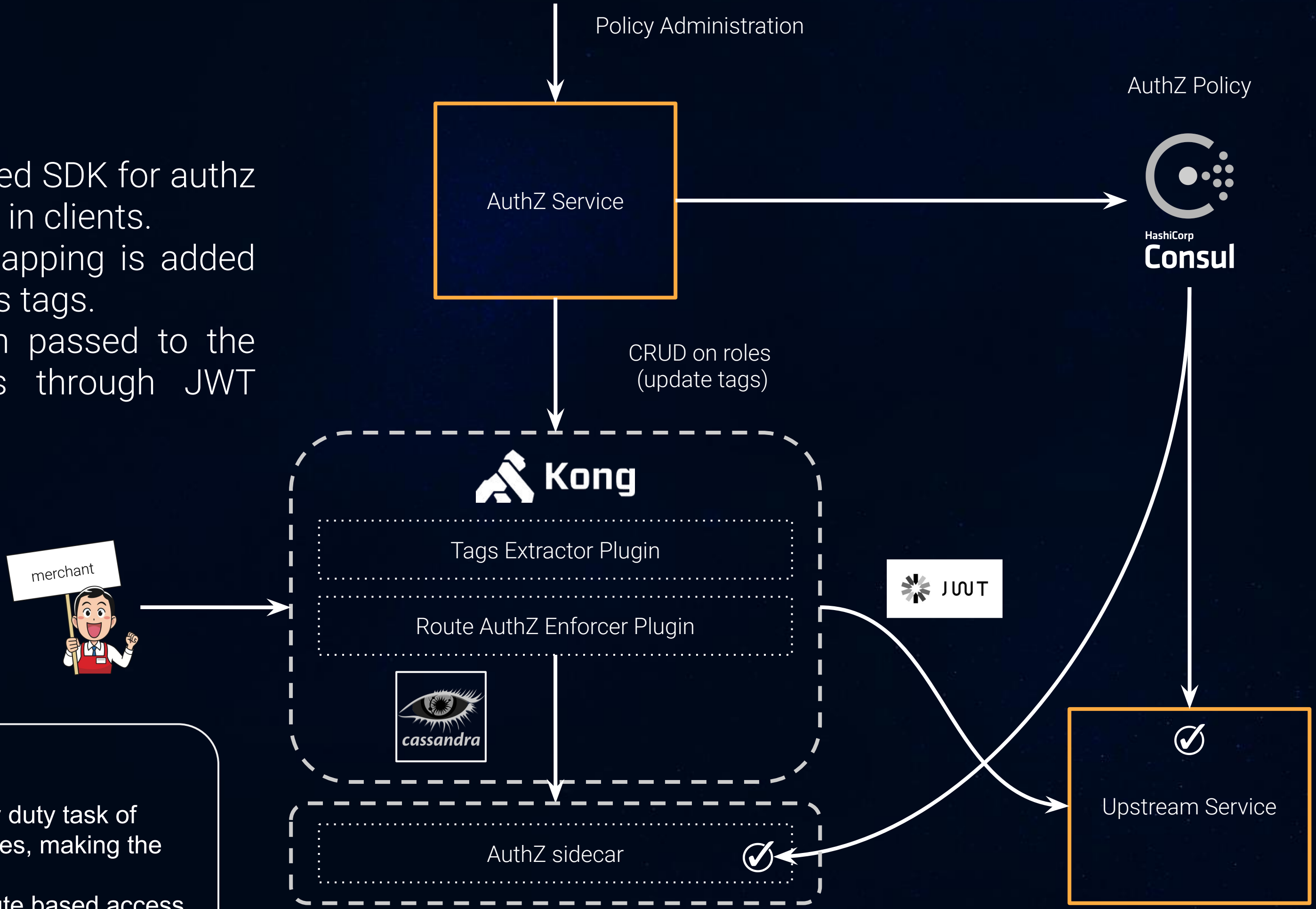
- Various models of authorization were needed to be supported. For example, ACL, RBAC, ABAC etc.
- Route based authorization enforcement needed to be done at Kong.
- Fine grained access control is responsibility of services.
- Services to have flexibility in terms of defining their PERM meta model schema (Policy, Effect, Request, Matchers).
- Authorization enforcement should be low latency operation for a service.
- Authorization service and enforcement design to be efficient in terms of network calls or memory footprint in services.

Some examples of authorization policies to be enforced in our ecosystem

➔ Restrict specific users from accessing a section in dashboard.
➔ Restrict user to access only records created by him/her
➔ Restrict users to access specific properties of an entity
➔ A partner to access specific entities of his merchant’s account, but restrict from accessing other data.
➔ Restrict access to a specific API route for a given key secret.

Authorization

- We use casbin based SDK for authz policy enforcement in clients.
- Subject to roles mapping is added to Kong's entities as tags.
- The roles are then passed to the upstream services through JWT token.



Kong helps in

1. Offloading the heavy duty task of mapping users to roles, making the SDK light weight.
2. AuthZ checks for route based access control

Community Contributions from Razorpay

- Support for different read and write consistency levels in Cassandra. This is required as we wanted a higher consistency level on writes for durability but lower on reads for speed.
  https://github.com/Kong/kong/commit/eeab2ec8ab0bcf76b21e3e8d5dfb58e81ee3ed88

- A performance fix in basic-auth plugin to avoid redundant fetches form the database.
  https://github.com/Kong/kong/commit/448b8fcbfb0aacc98fd48071d7ae5728037621a8

- Triaged and reported performance results with basic-auth plugin.
  https://discuss.konghq.com/t/primary-key-for-cassandra-in-basic-auth-plugin/6169

- Triaged and reported an issue with DNS timeouts.
  https://discuss.konghq.com/t/kong-pongo-hangs-with-kong-plugin/5517

Thanks