Redis TLS Origination

through the sidecar

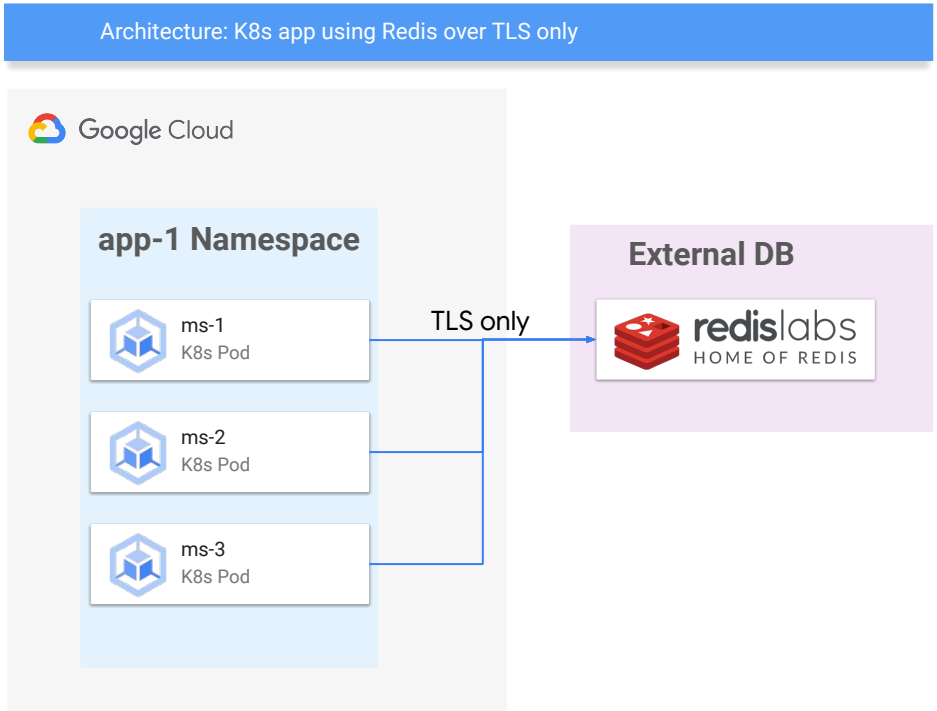
Author: Sam Stoelinga | Twitter: samosx | GitHub: samos123

Based on blog post: https://samos-it.com/posts/securing-redis-istio-tls-origniation-termination.html

What are we solving?

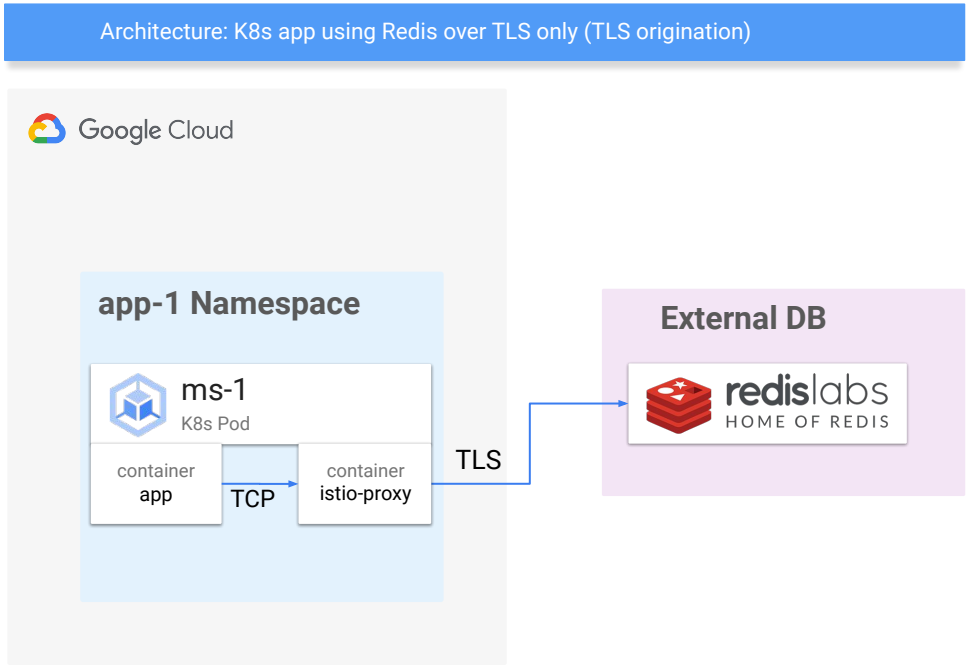
- App with multiple microservices
- external Redis TLS only
- each microservice talks to Redis

1. Tedious configuration in the application
2. Improved visibility into Redis traffic



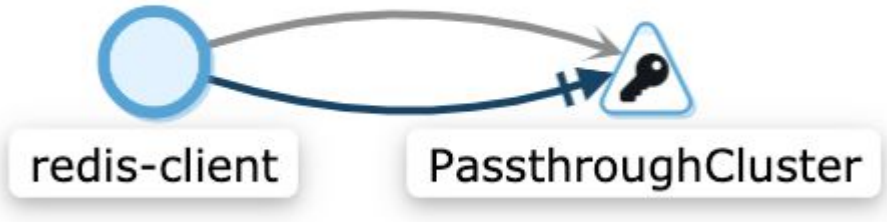
Istio TLS Origination

- app talks unencrypted TCP to Redis
- Sidecar istio-proxy encrypts the Redis traffic and sends to external redis
- App doesn't need to configure certs
- Traffic becomes more "visible"

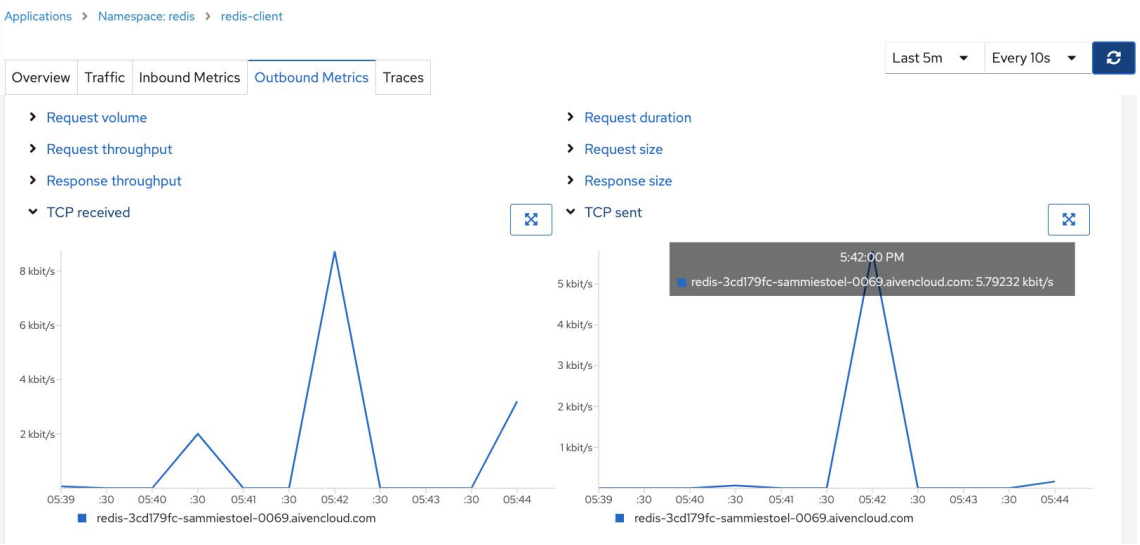


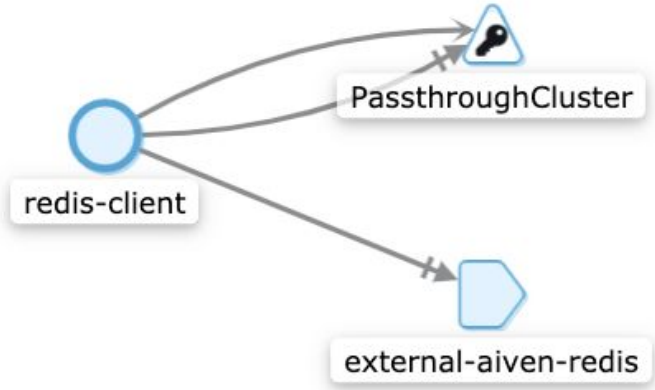
How traffic looks like before TLS origination

- Unclear where traffic is going
- No metrics available



How it looks after TLS origination





How to do Redis TLS origination with the sidecar?

1. Create **ServiceEntry** for external service such that Istio knows about Redis
2. Create **DestinationRule** to configure TLS origination for Redis

```yaml
apiVersion: networking.istio.io/v1beta1
kind: ServiceEntry
metadata:
  name: external-aiven-redis
  namespace: redis
spec:
  hosts:
  - redis-1425a1d9-google-bc39.aivencloud.com
  location: MESH_EXTERNAL
  resolution: DNS
  ports:
  - number: 16222
    name: tcp-redis
    protocol: TCP
```

```yaml
apiVersion: networking.istio.io/v1beta1
kind: DestinationRule
metadata:
  name: external-aiven-redis
  namespace: redis
spec:
  host: redis-1425a1d9-google-bc39.aivencloud.com
  trafficPolicy:
    tls:
      mode: SIMPLE
      caCertificates: /etc/ssl/certs/ca-certificates.crt
```

Demo time

1. Deploy the redis-client with a sidecar, however no ServiceEntry and no DestinationRule
   Expectation: Should fail when trying to connect over plain TCP
2. Create DestinationRule and ServiceEntry
   Expectation: Ability to connect over redis through unencrypted TLS