slack
from Salesforce

Level up your edge

Do you know where your traffic comes from?

Who Am I

Radha Kumari

Staff Software Engineer / Cloud Network

Slack

Agenda

- Motivation
- User aware traffic management
- Architecture
- Rollout process
- Challenges
- Cool tricks with data
- Takeaways

Motivation

- Authentication information unavailable at the edge.
- Inability to recognise abusive vs legitimate traffic at the edge.
- Existing rate-limits only apply to the requests that are authenticated.

slack

User Aware Traffic Management

- extract **team (e.g T0QR8V06R)** and **user (e.g U9W4R407K)** information **from client API requests** at the edge.

slack

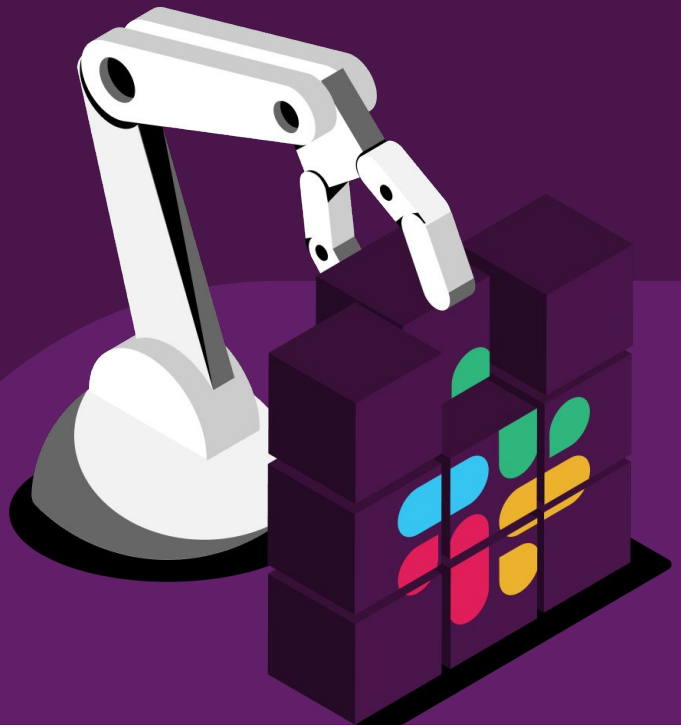
User Aware Traffic Management

- extract **team (e.g T0QR8V06R)** and **user (e.g)** information **from client API requests** at the edge.

- Enabling us to take **routing decisions on user identity.**

slack

Architecture



Architecture

- envoy-edge config contains: Listener (filterchain) → routes → clusters.

```
listeners:
  - name: listener_0
    address:
      socket_address:
        protocol: TCP
        address: 0.0.0.0
        port_value: 8000
    filter_chains:
      - filters:
          - name: envoy.filters.network.http_connection_manager
            typed_config:
              "@type": type.googleapis.com/envoy.extensions.filters.network.http_connecti
manager.v3.HttpConnectionManager
              stat_prefix: ingress_http
              route_config:
                name: local_route
                virtual_hosts:
                  - name: auth-grpc
                    domains: ["*"]
                    routes:
                      - match:
                          prefix: "/"
                        route:
                          cluster: to-admin

clusters:
  - name: to-admin
    connect_timeout: 1s
    type: STATIC
    load_assignment:
      cluster_name: to-admin
      endpoints:
        - lb_endpoints:
            - endpoint:
                address:
                  socket_address:
                    address: 0.0.0.0
                    port_value: 9000
```

slack

Architecture

- envoy-edge config contains: Listener (filterchain) → routes → clusters.
- ext-authz filter configured at envoy-edge sends requests to a GRPC Auth service deployed at the edge.

```
        http_filters:
        - name: envoy.filters.http.ext_authz
          typed_config:
            "@type": type.googleapis.com/envoy.extensions.filters.http
.ext_authz.v3.ExtAuthz
            grpc_service:
              envoy_grpc:
                cluster_name: ext-authz
            with_request_body:
              max_request_bytes: 1024
              allow_partial_message: true
```

```
clusters:
- name: ext-authz
  type: LOGICAL_DNS
  lb_policy: ROUND_ROBIN
  load_assignment:
    cluster_name: ext-authz
    endpoints:
    - lb_endpoints:
      - endpoint:
          address:
            socket_address:
              address: 127.0.0.1
              port_value: 10003
```

slack

Architecture

- envoy-edge config contains: Listener (filterchain) → routes → clusters.
- ext-authz filter configured at envoy-edge sends requests to a GRPC Auth service deployed at the edge.
- Authentication (Auth) service implements implements Envoy Authorization service.

slack

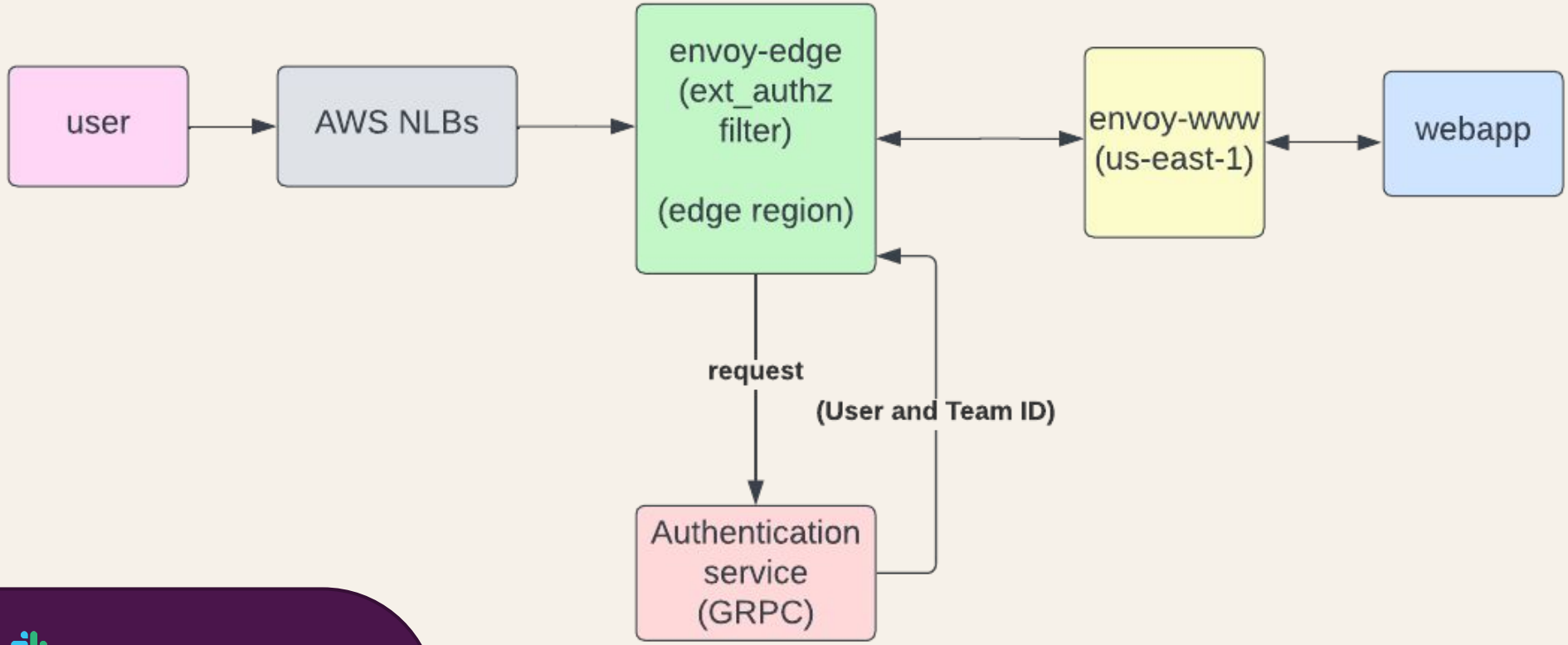
```
func buildCheckResponse(r resultBits) *envauth.CheckResponse {

        return &envauth.CheckResponse{
                Status: &status.Status{
                        Code: int32(rpc.OK),
                },
                DynamicMetadata: &structpb.Struct{
                        Fields: map[string]*structpb.Value{
                                "ext_authz_duration": {
                                        Kind: &structpb.Value_NumberValue{NumberV
                                },
                        },
                },
                HttpResponse: &envauth.CheckResponse_OkResponse{
                        OkResponse: &envauth.OkHttpResponse{
                                Headers: []*envcore.HeaderValueOption{
                                        {
                                                Header: &envcore.HeaderValue{
                                                        Key:   slackUserHeader,
                                                        Value: r.UserID,
                                                },
                                        }, {
                                                Header: &envcore.HeaderValue{
                                                        Key:   slackTeamHeader,
                                                        Value: r.TeamID,
                                                },
                                        }, {
                                                Header: &envcore.HeaderValue{
                                                        Key:   slackAuthResultHea
                                                        Value: r.result,
                                                },

.............
```

Architecture

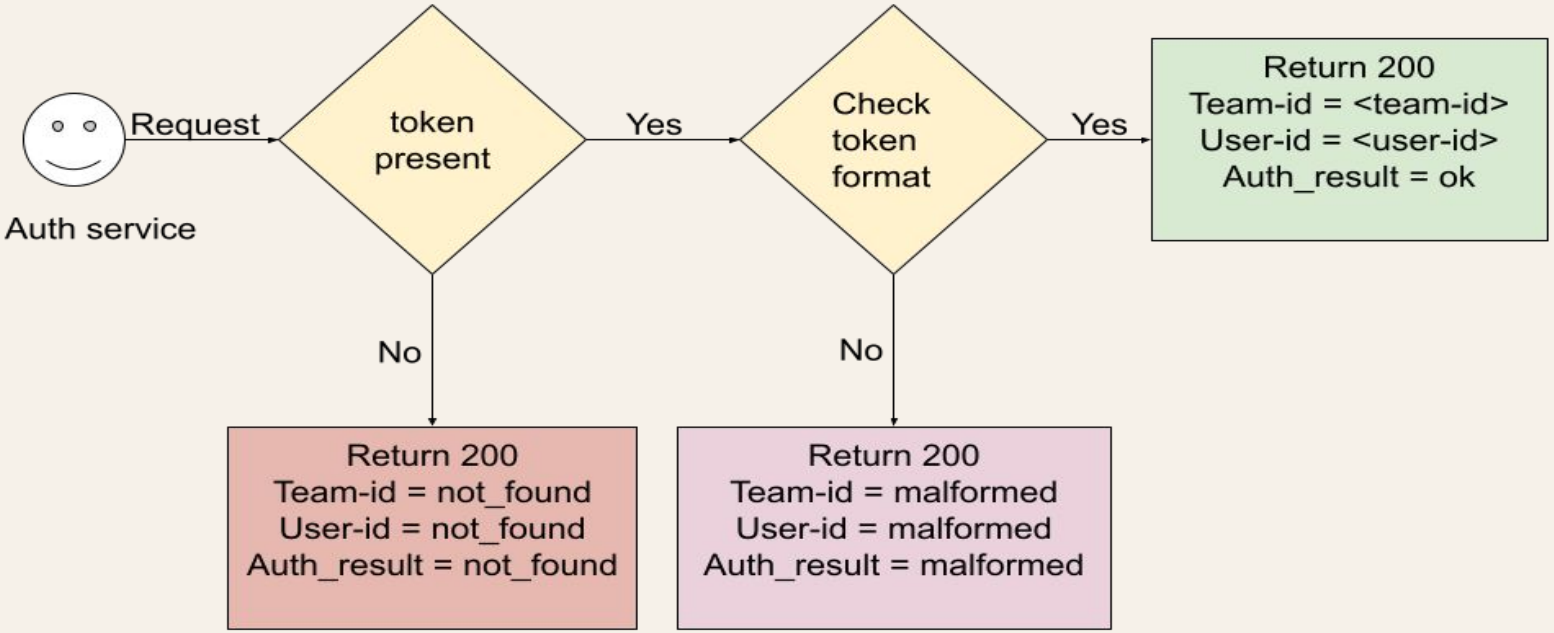
- envoy-edge config contains: Listener (filterchain) → routes → clusters.
- ext-authz filter configured at envoy-edge sends requests to a GRPC Auth service deployed at the edge.
- Authentication (Auth) service implements implements Envoy Authorization service.
- No validation, just token shape checking.

slack

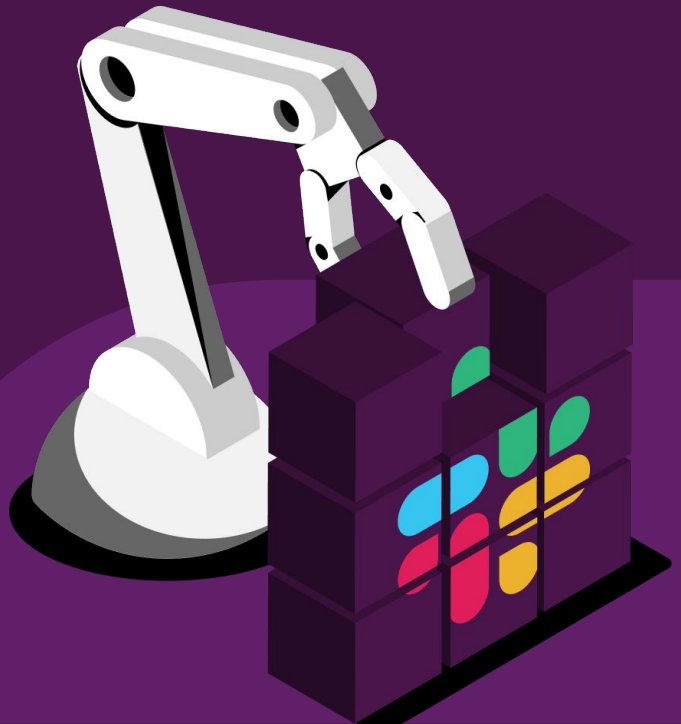
Architecture



Architecture



Rollout Process



Rollout process

Phases:

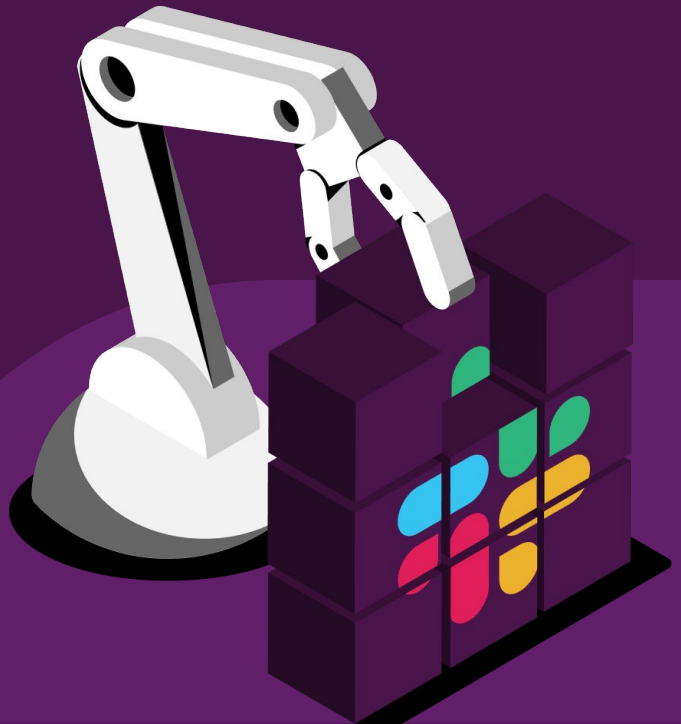
- parsing request headers for token: user awareness -> 20%
- parsing of the request body for token: user awareness -> 50%
- Add ability to rate-limit or block traffic based on user information

Process:

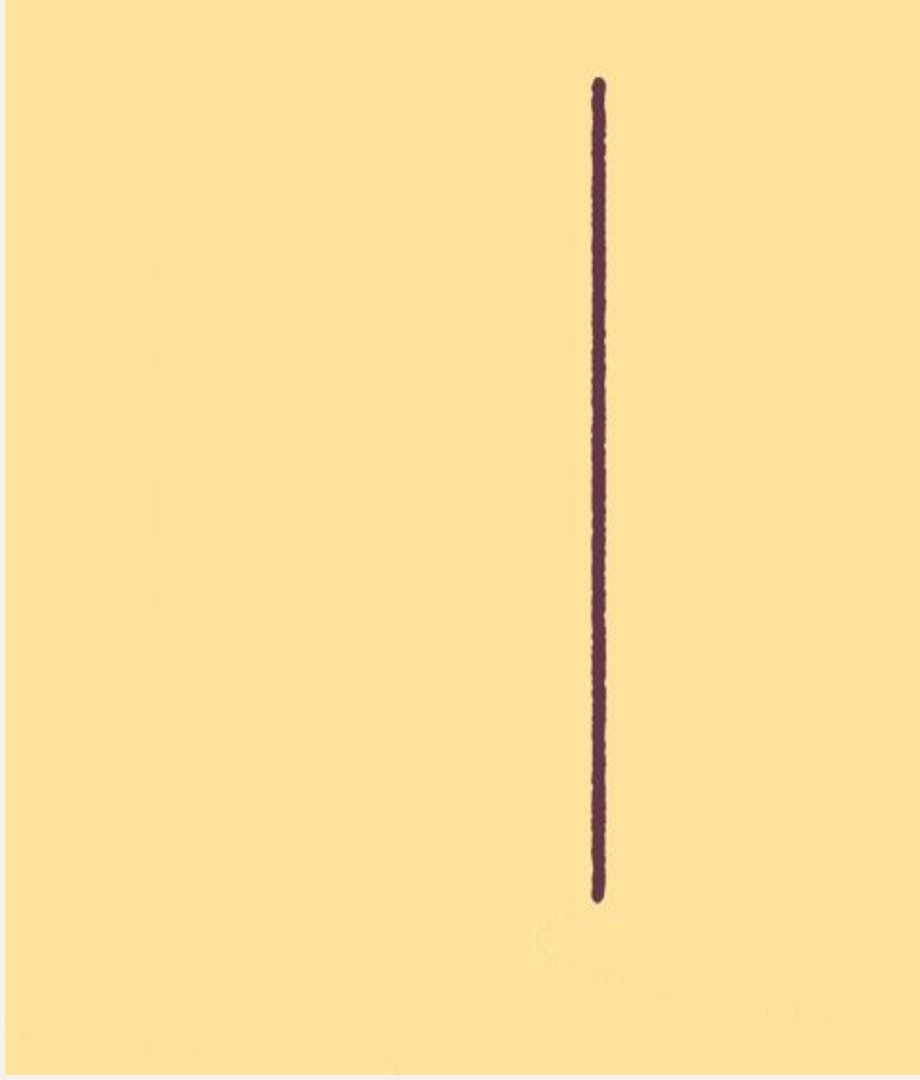
- Development → Production
- Region by region

slack

Challenges

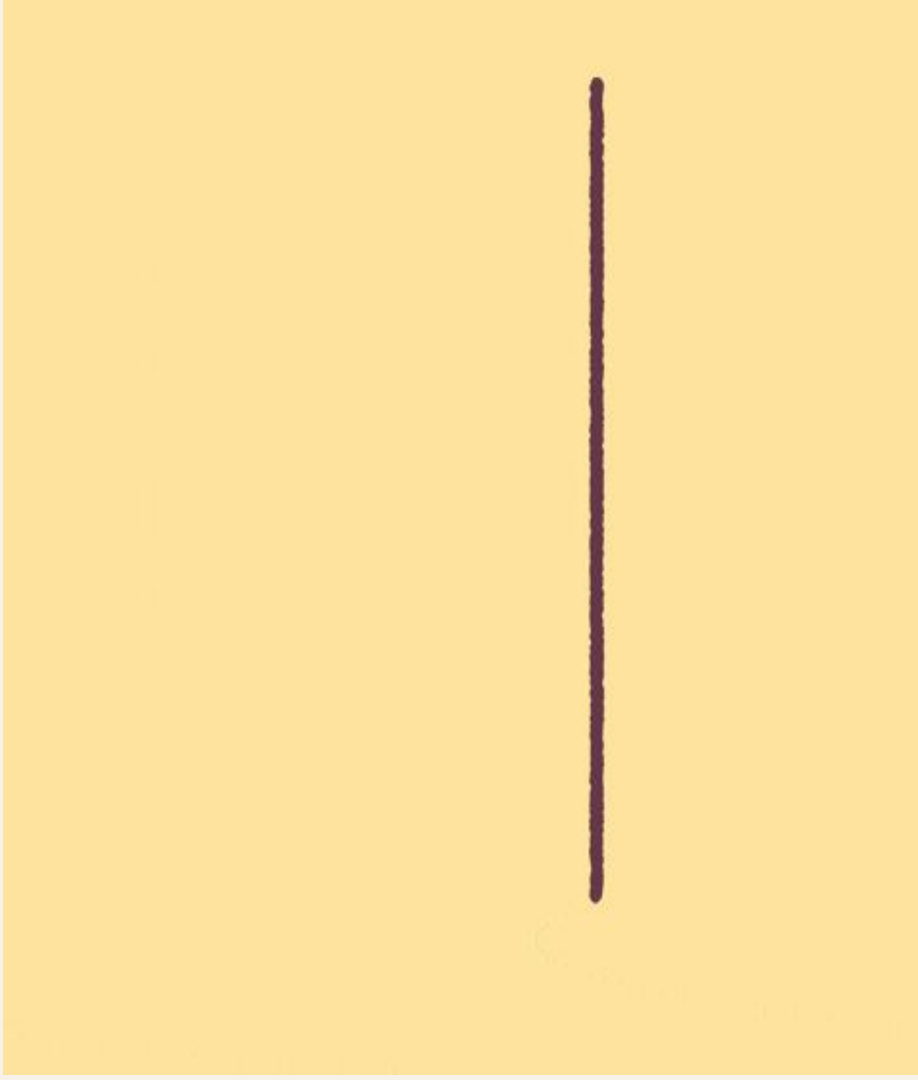


Lack of Standardization



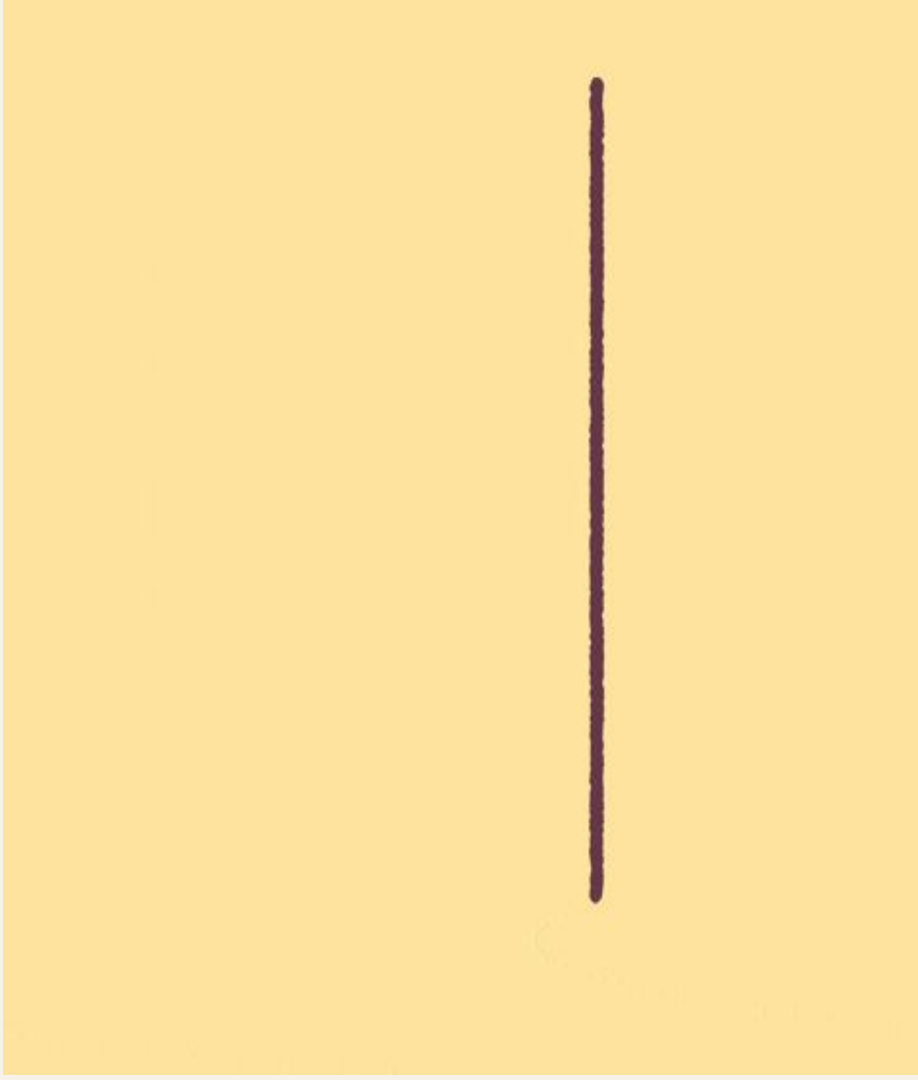
Lack of Standardization

- Most Slack clients & SDKs present **token in the authorization header**.



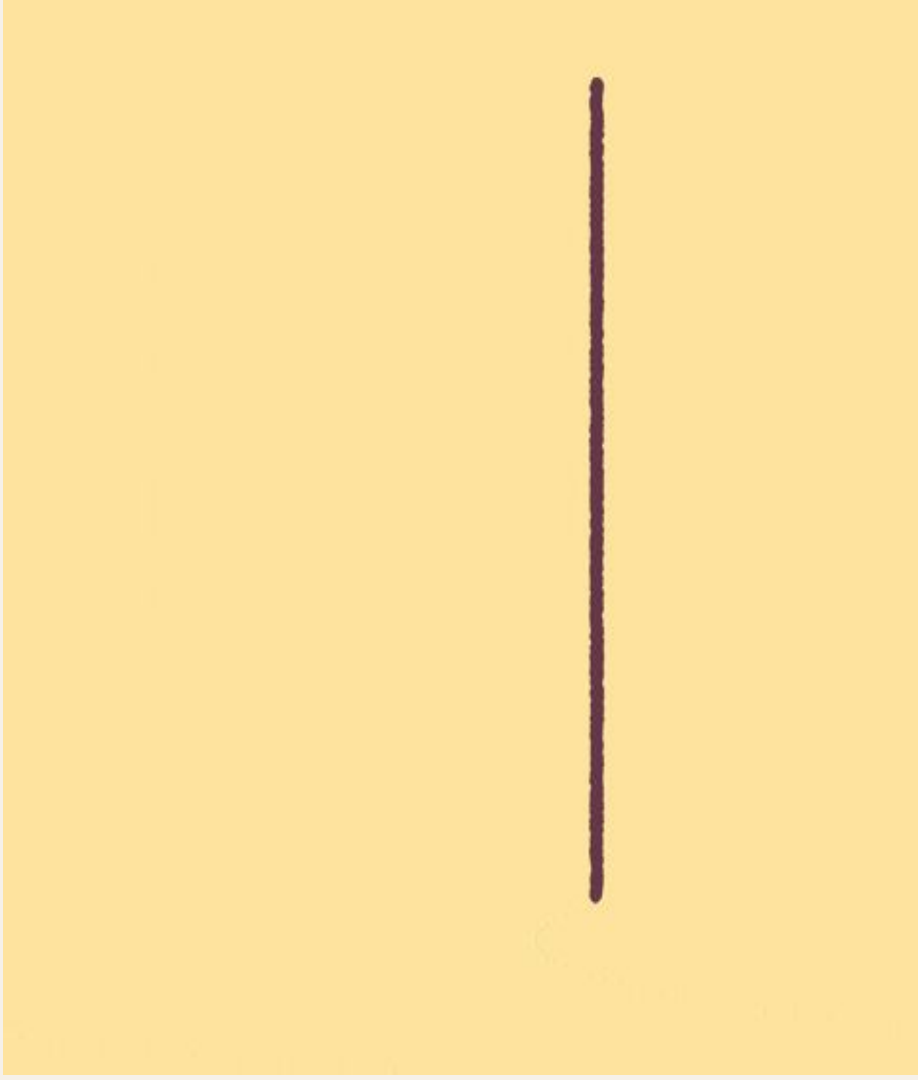
Lack of Standardization

- Most Slack clients & SDKs present **token in the authorization header**.
- Some in the first parameter of request body.

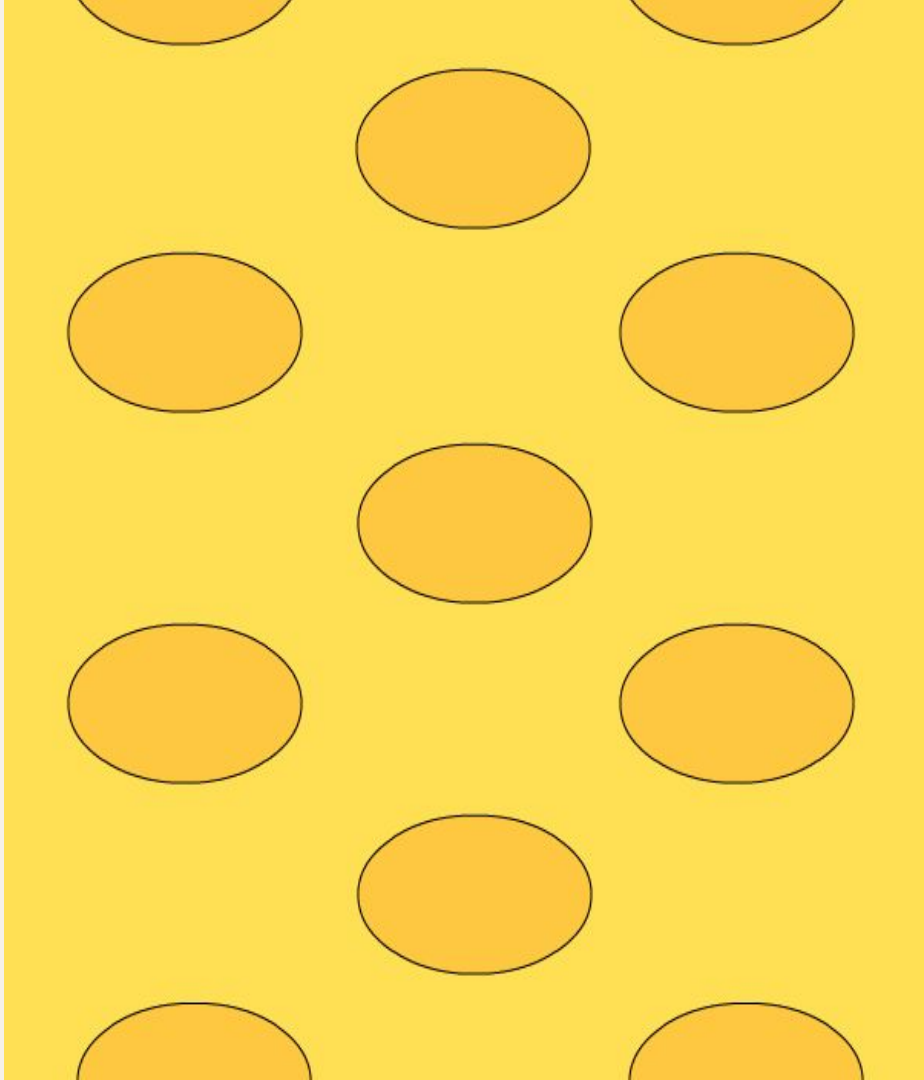


Lack of Standardization

- Most Slack clients & SDKs present **token in the authorization header**.
- Some in the first parameter of request body.
- And then there are webhooks, workflows, cache requests, third party apps, etc.

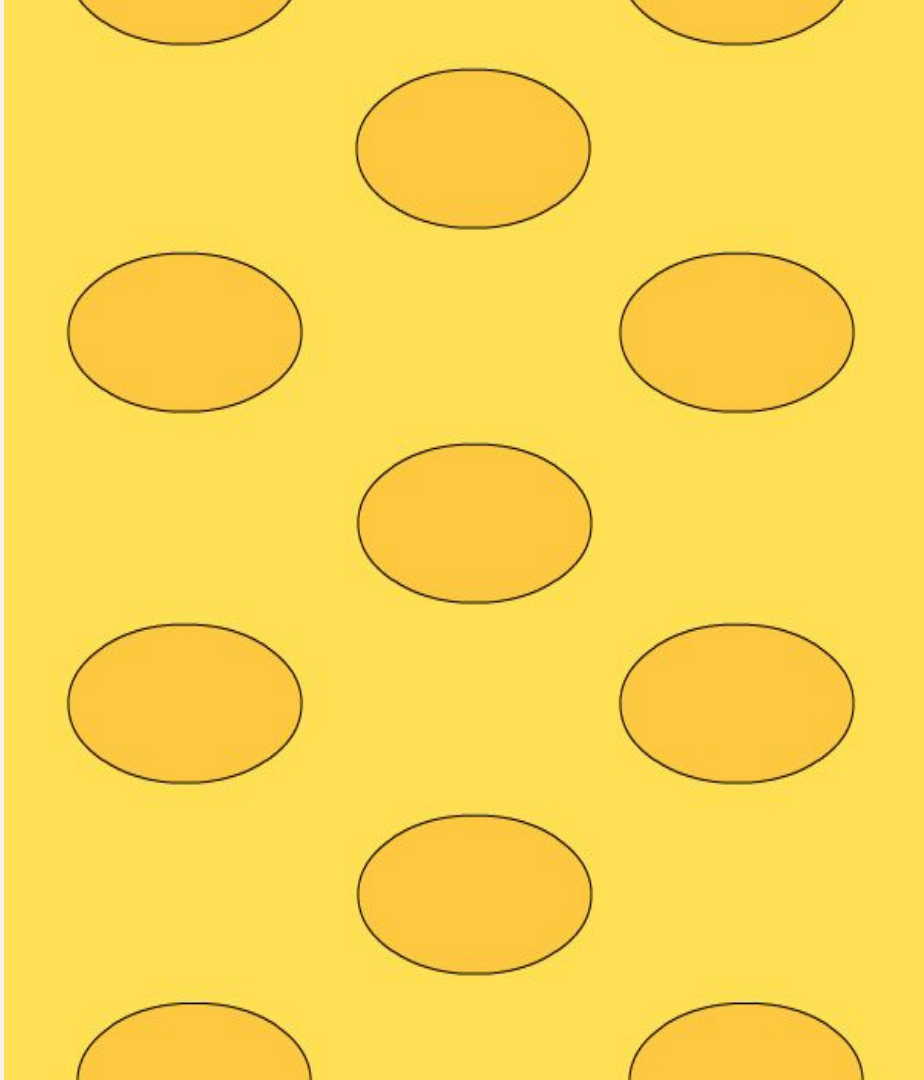


Whack-a-mole



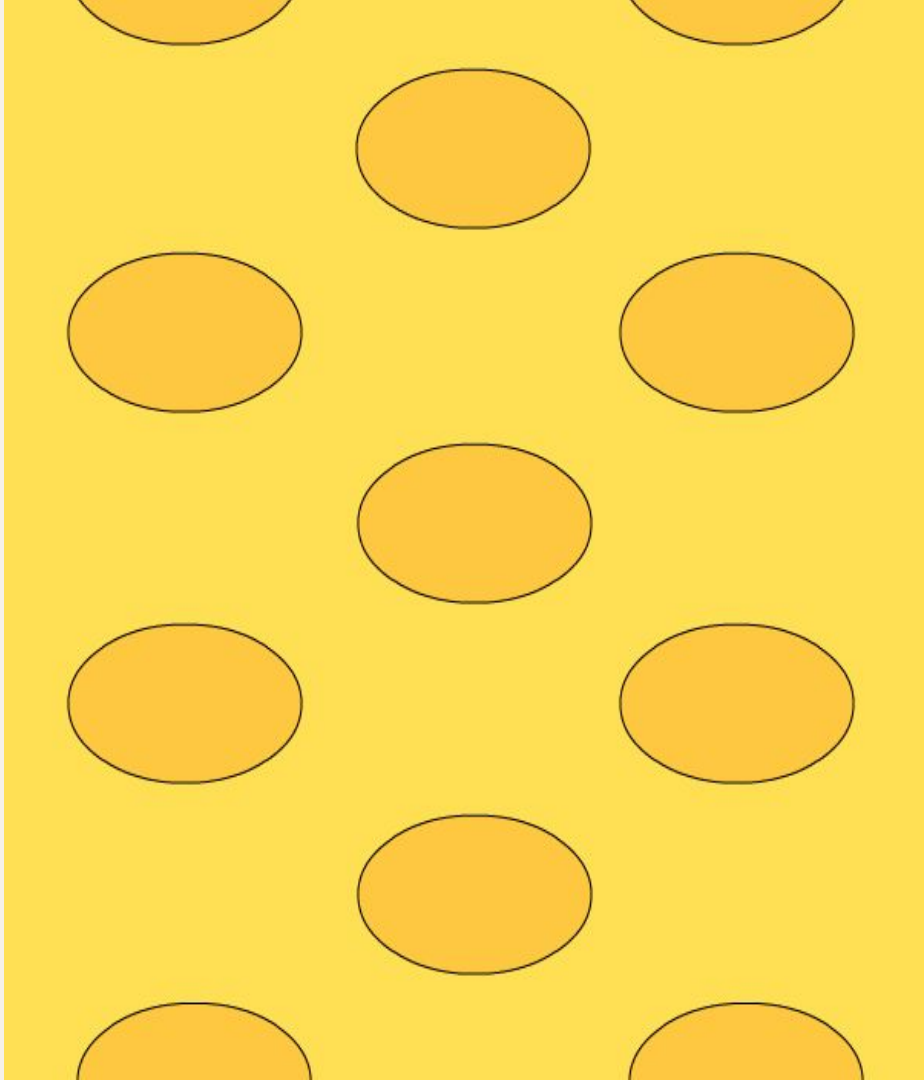
Whack-a-mole

- Look for token in the Authorization header and in the body, **whats next?**



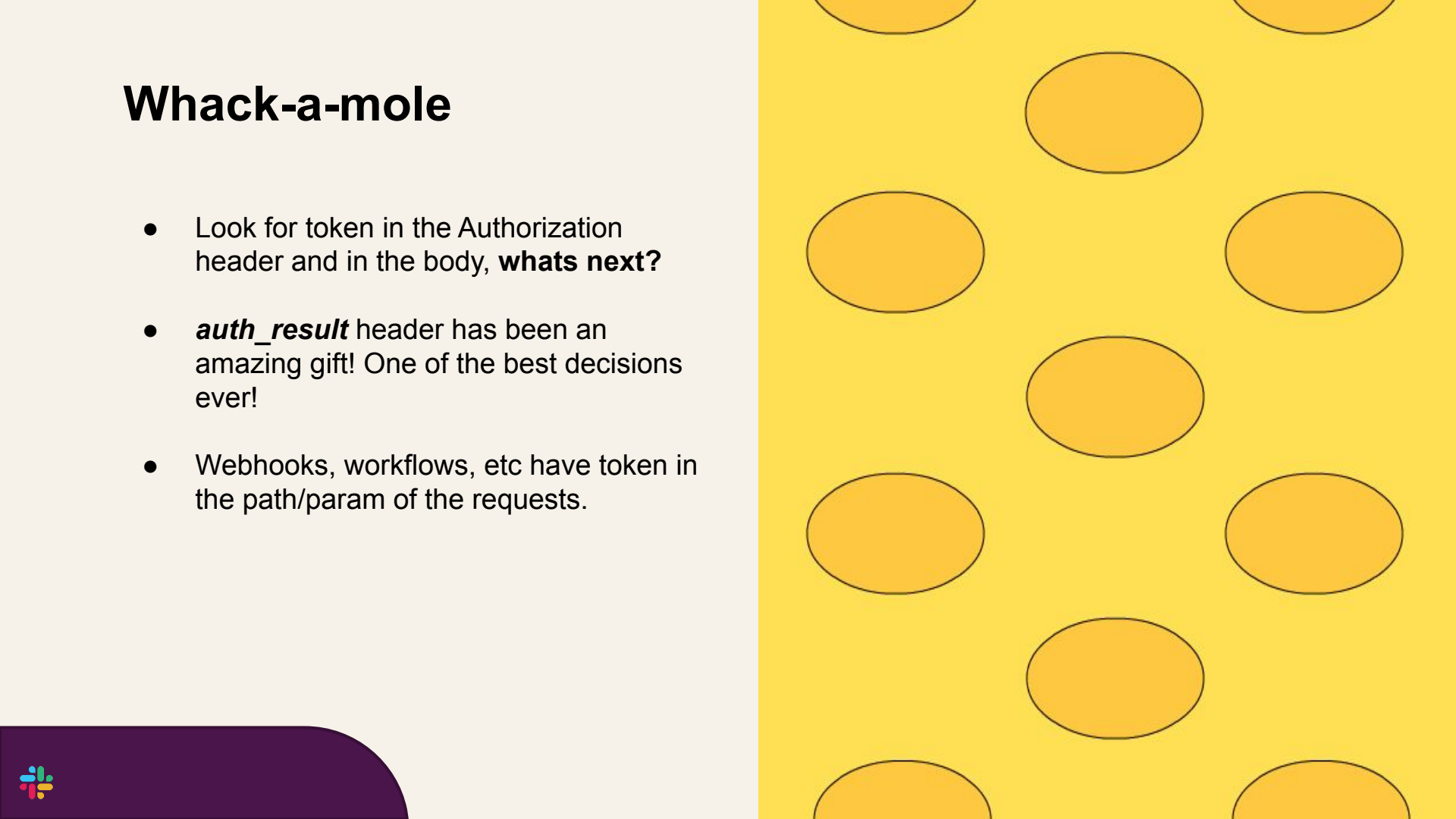
Whack-a-mole

- Look for token in the Authorization header and in the body, **whats next?**
- ***auth_result*** header has been an amazing gift! One of the best decisions ever!

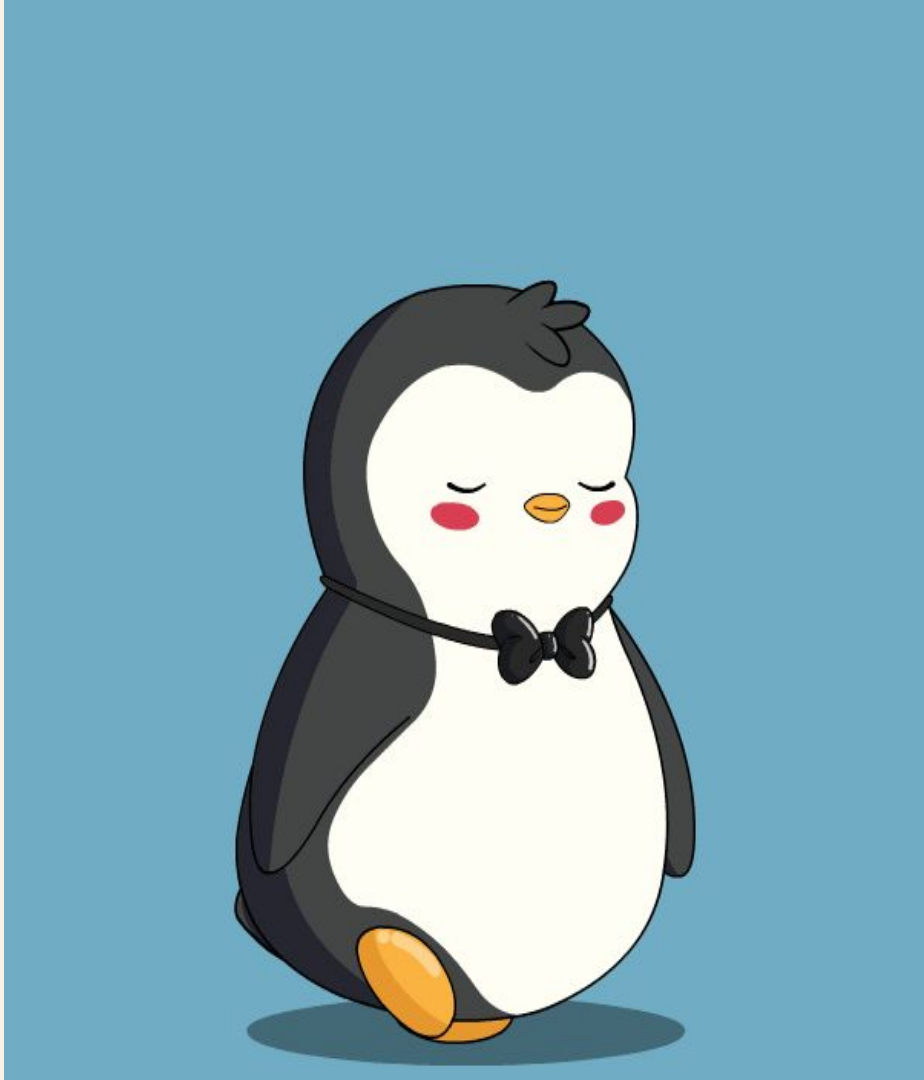


Whack-a-mole

- Look for token in the Authorization header and in the body, **whats next?**
- ***auth_result*** header has been an amazing gift! One of the best decisions ever!
- Webhooks, workflows, etc have token in the path/param of the requests.

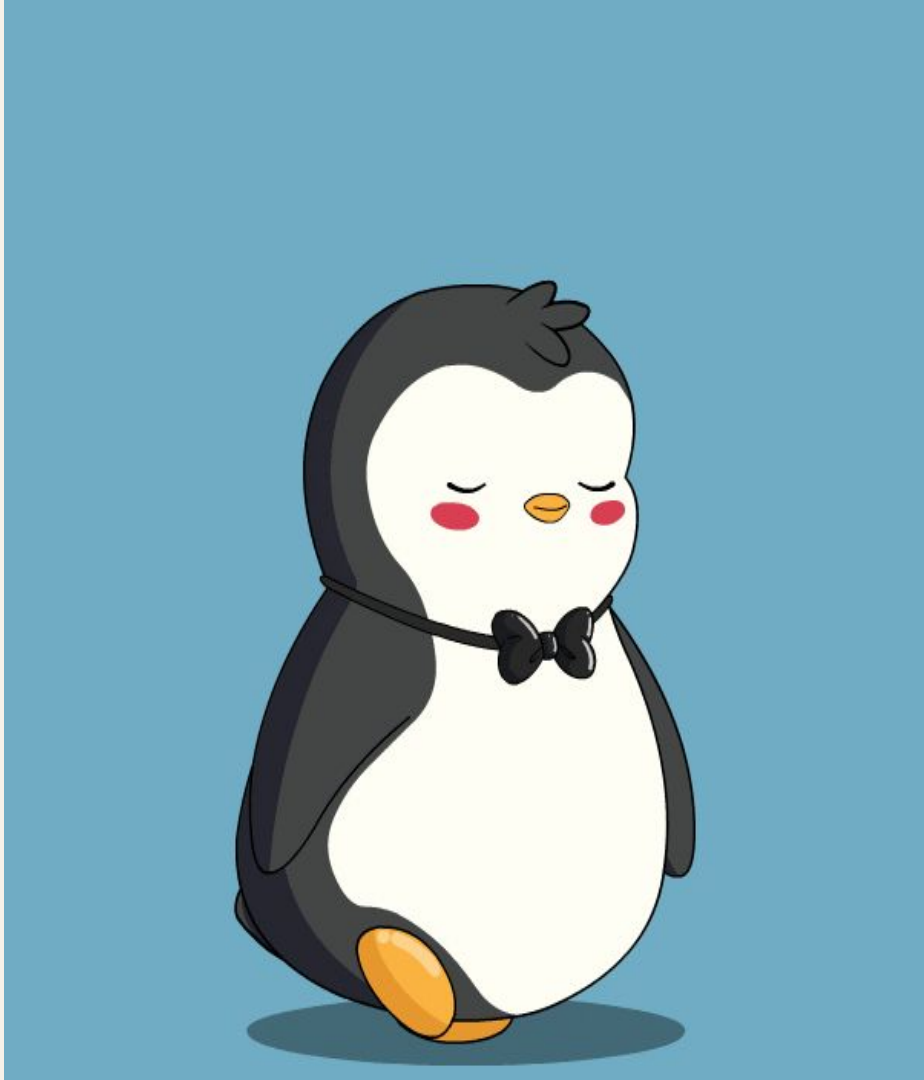


Latency



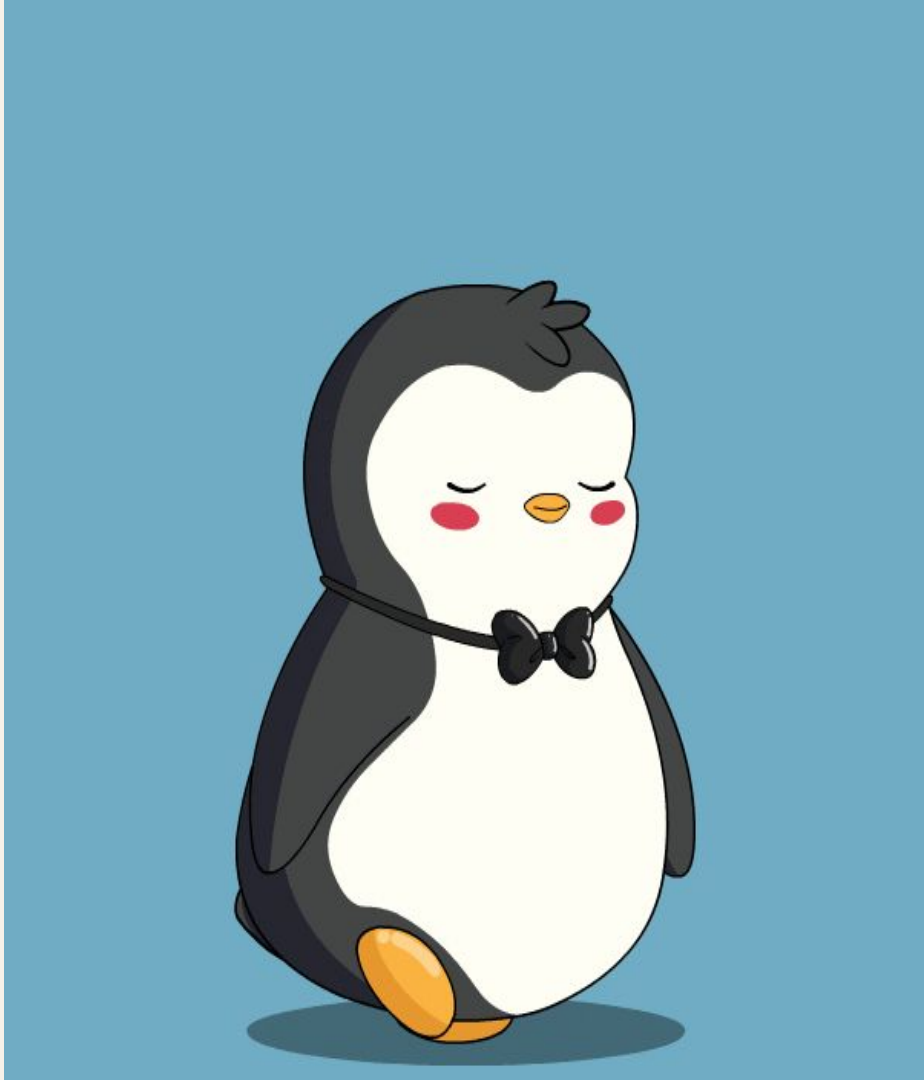
Latency

- **Extra network hop** between envoy-edge and the backend.



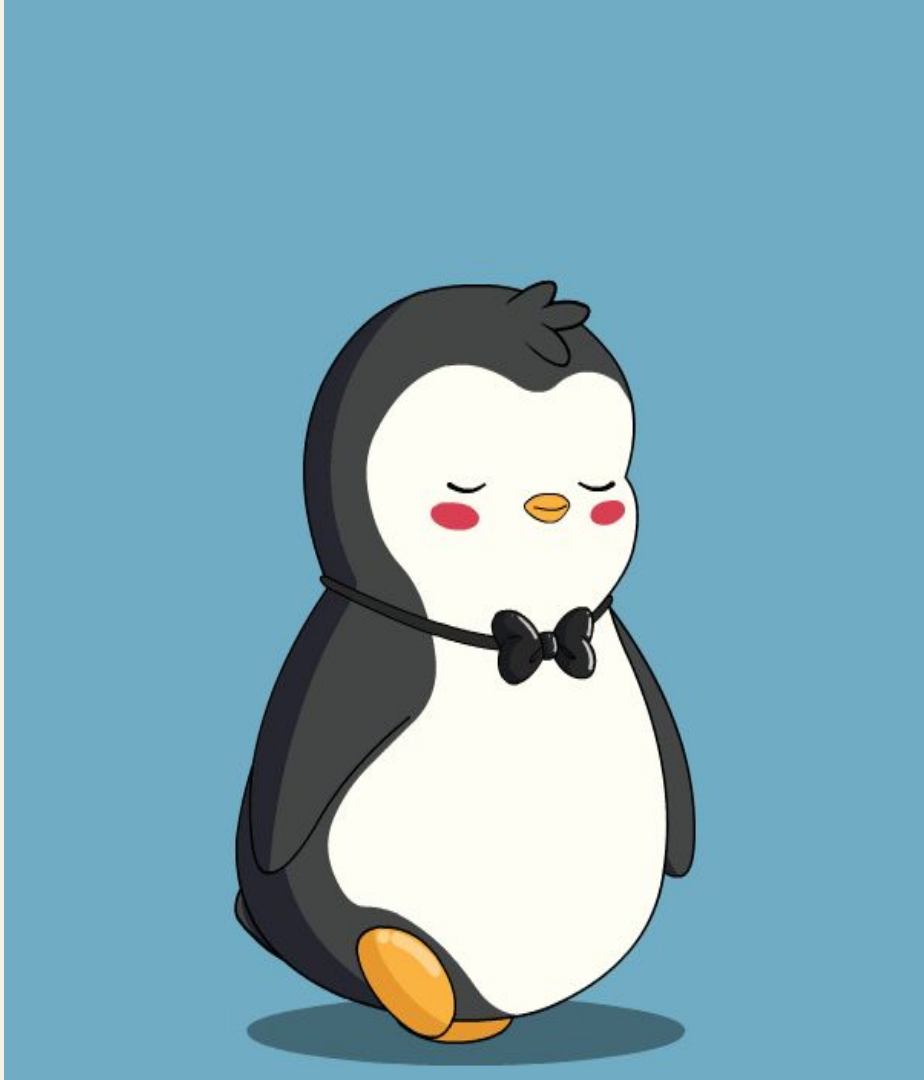
Latency

- **Extra network hop** between envoy-edge and the backend.
- Came up with the **budget based on duration** of the requests.



Latency

- **Extra network hop** between envoy-edge and the backend.
- Came up with the **budget based on duration** of the requests.
- **P99 is less than 5ms** between envoy-edge and auth service.



Snowflake requests

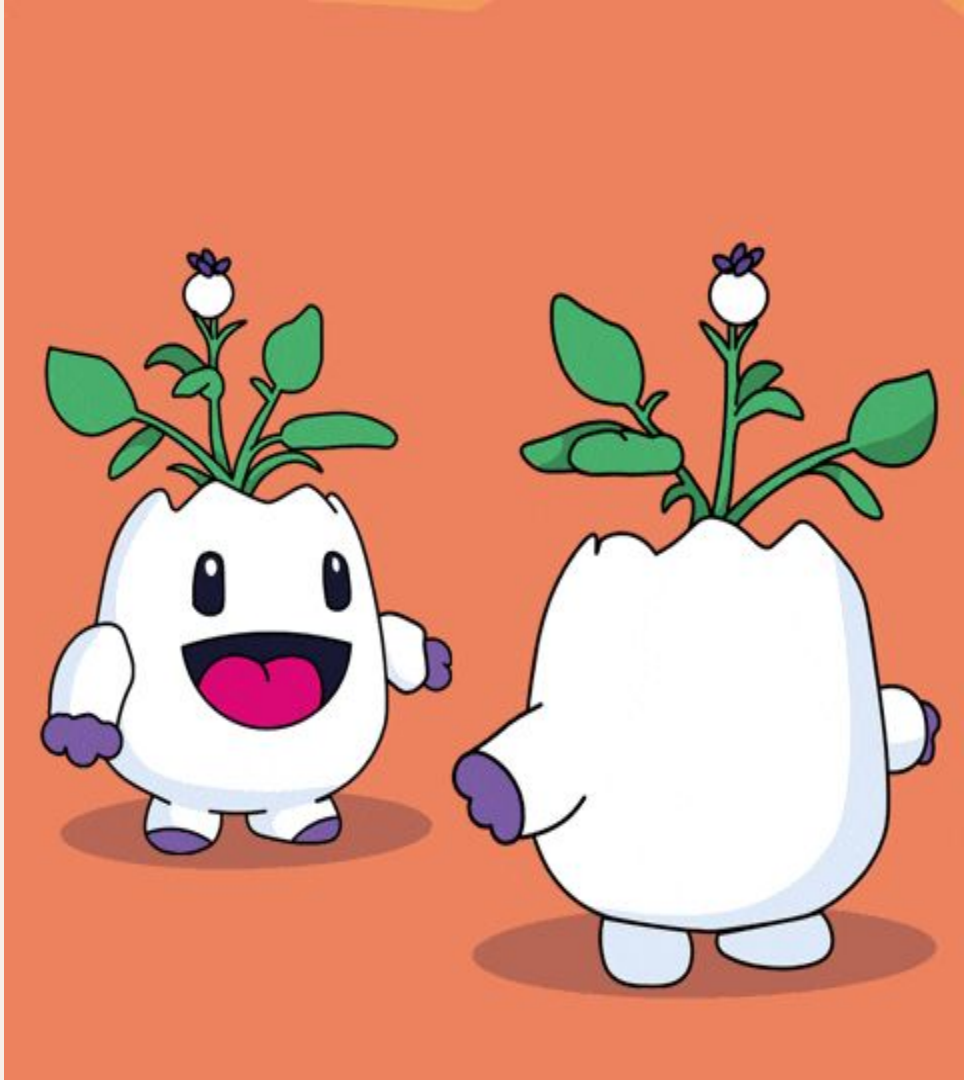
Snowflake requests

- Some API requests are very very **very big requests, like files upload**.

Snowflake requests

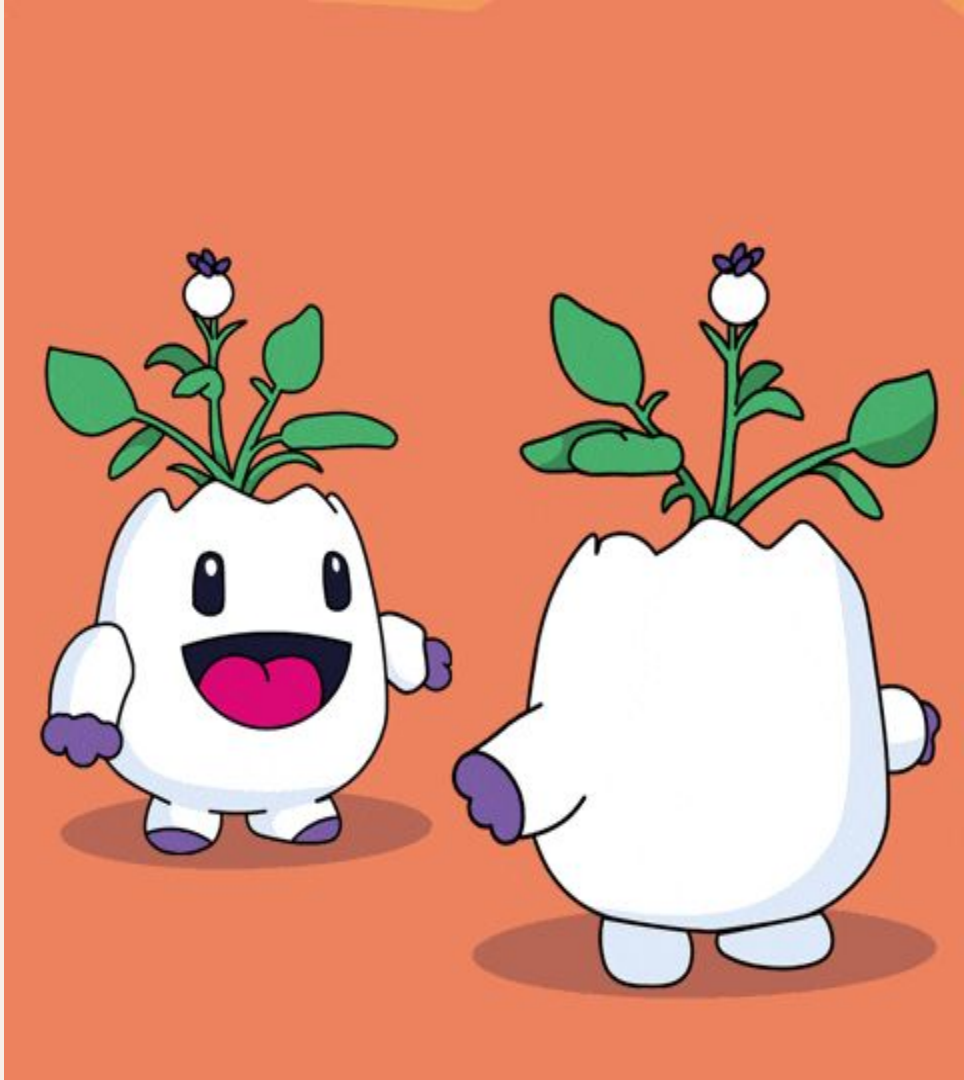
- Some API requests are very very **very big requests, like files upload**.
- We **can’t buffer indefinitely**.

Ownership



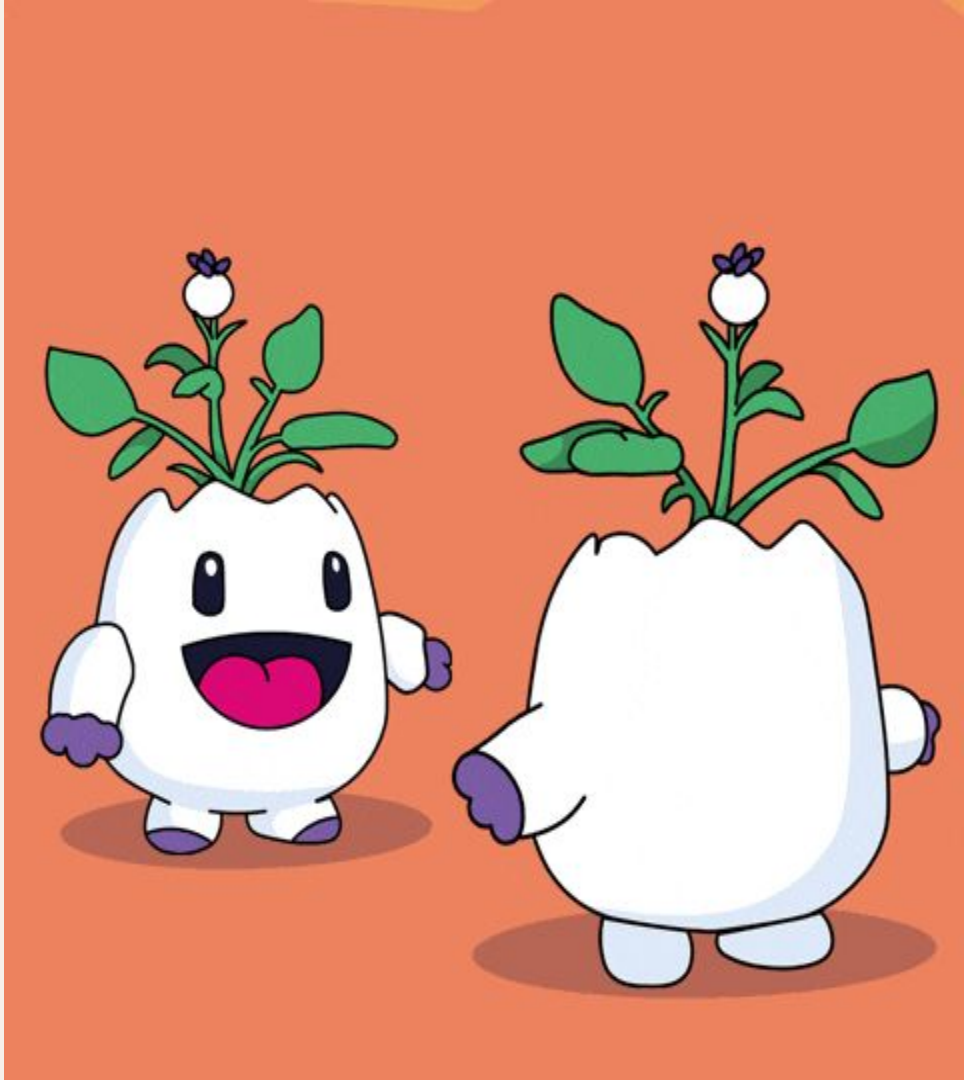
Ownership

- Initially owned by Demand Engineering and Sign-in Infra team.



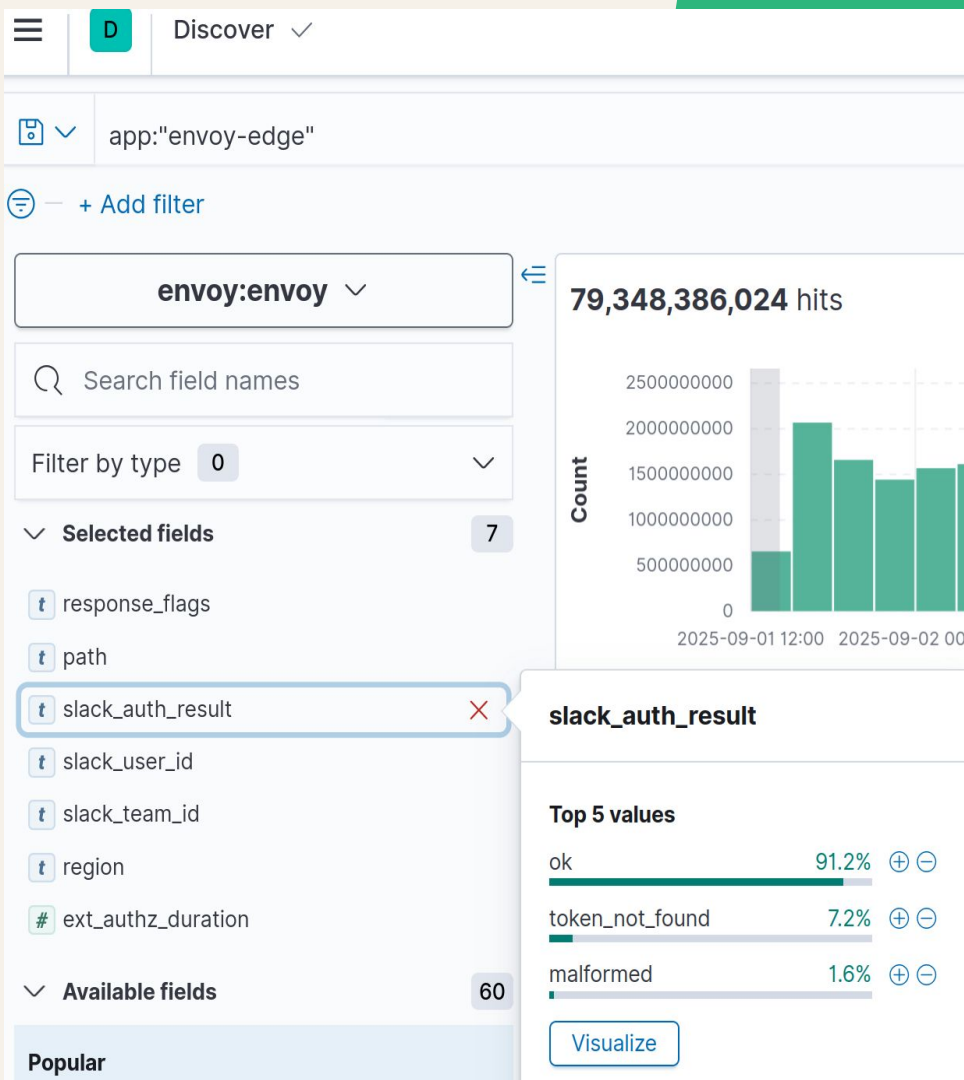
Ownership

- Initially owned by Demand and Sign-in Infra team
- Learn to operate in a **distributed ownership** model

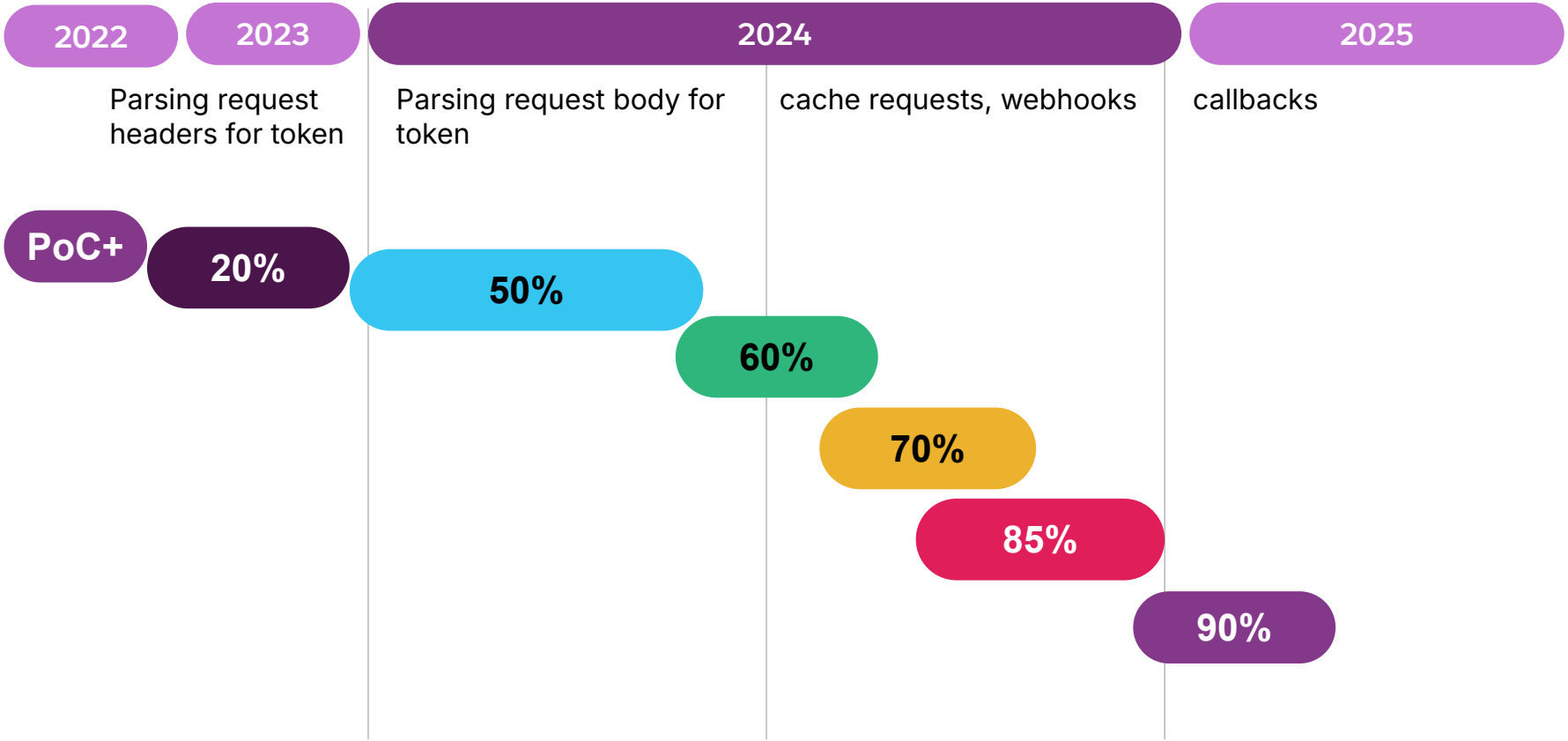


Where are we now

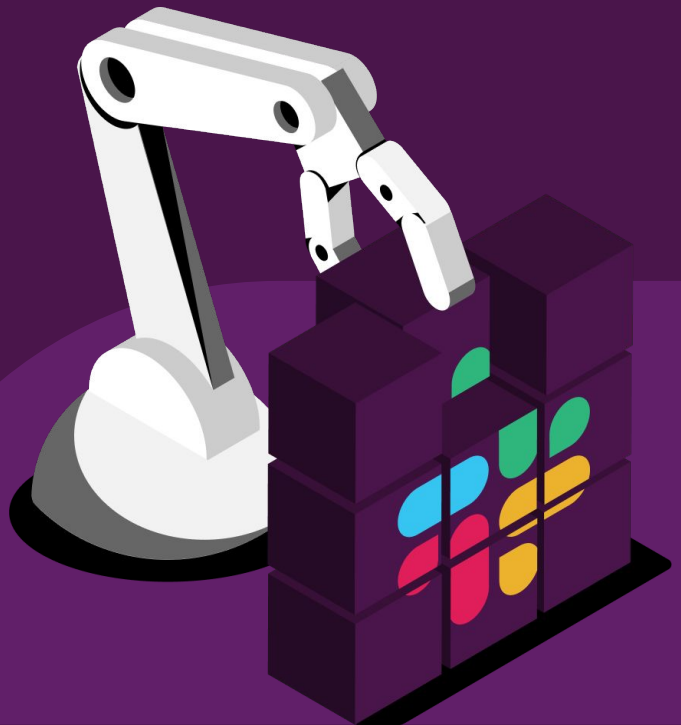
- User awareness percentage is about **90%**.
- Token validation at auth service.
- Parity with application monolith.



user awareness percentage progress



Cool tricks with user data…



Smarter load balancing 🎯

- More reliable identification of abusive traffic at edge.

slack

Smarter load balancing 🎯

- More reliable identification of abusive traffic at edge.
- Rate-limiting based on bots, unauthenticated traffic, etc.

slack

Smarter load balancing 🎯

- More reliable identification of abusive traffic at edge.
- Rate-limiting based on bots, unauthenticated traffic, etc.
- Select routing destinations based on customer identity.

slack

Smarter load balancing 🎯

- More reliable identification of abusive traffic at edge.
- Rate-limiting based on bots, unauthenticated traffic, etc.
- Select routing destinations based on customer identity.
- Routing suspected abuse traffic to low-priority ‘quarantine’ cluster.

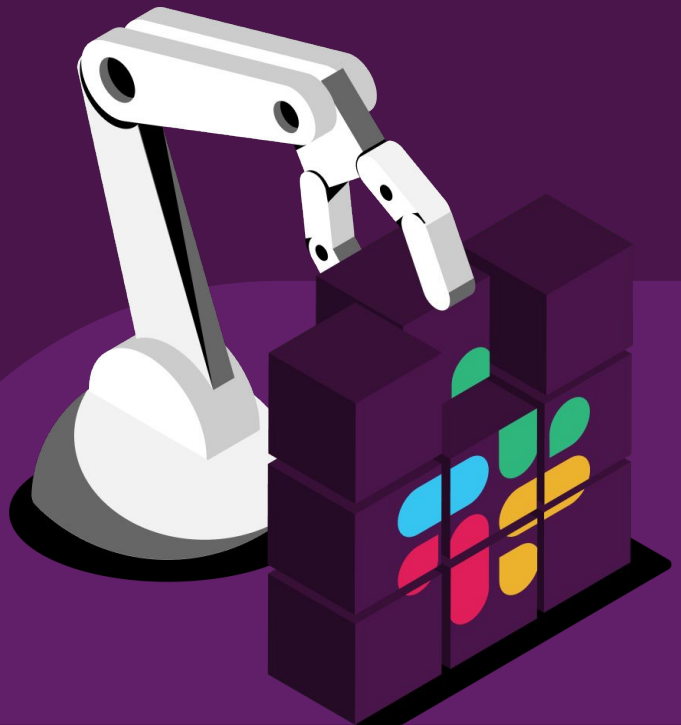
slack

Troubleshooting 🚀

- Troubleshooting
  - we have logs with (anonymised) user and team IDs, which helps understanding traffic patterns in incidents.
- Completely eliminated certain classes of incidents.

slack

Takeaways 🎁



- Ext-authz can be used to turbocharge your edge load balancing 

slack

- Ext-authz can be used to turbocharge your edge load balancing 
- Increased visibility into where your traffic is coming from unlocks a lot of capabilities 

slack

- Ext-authz can be used to turbocharge your edge load balancing 
- Increased visibility into where your traffic is coming from unlocks a lot of capabilities 
- Standardize where you can, it really helps 

slack

- Ext-authz can be used to turbocharge your edge load balancing. 
- Increased visibility into where your traffic is coming from unlocks a lot of capabilities. 
- Standardize where you can, it really helps 
- Embrace the edge cases! 

slack

slack
from Salesforce

Thank you 

Q&A