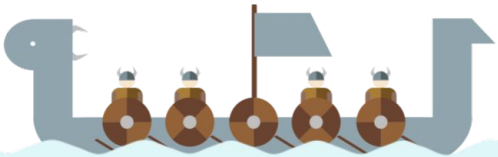
Distributed Authorization with Open Policy Agent

Anders Eknert

styra



Anders Eknert

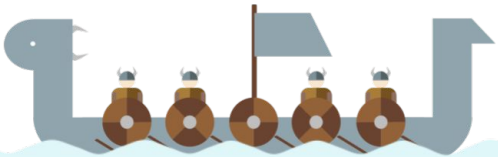
- Developer advocate at styra
- Software development
- Background in identity systems
- Three years into OPA
- Cooking and food
- Football

anderseknert@hachyderm.io
anderseknert
anderseknert
anderseknert

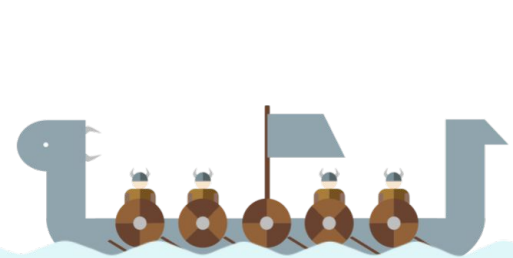
Problem:

How do we do authorization in distributed APIs?

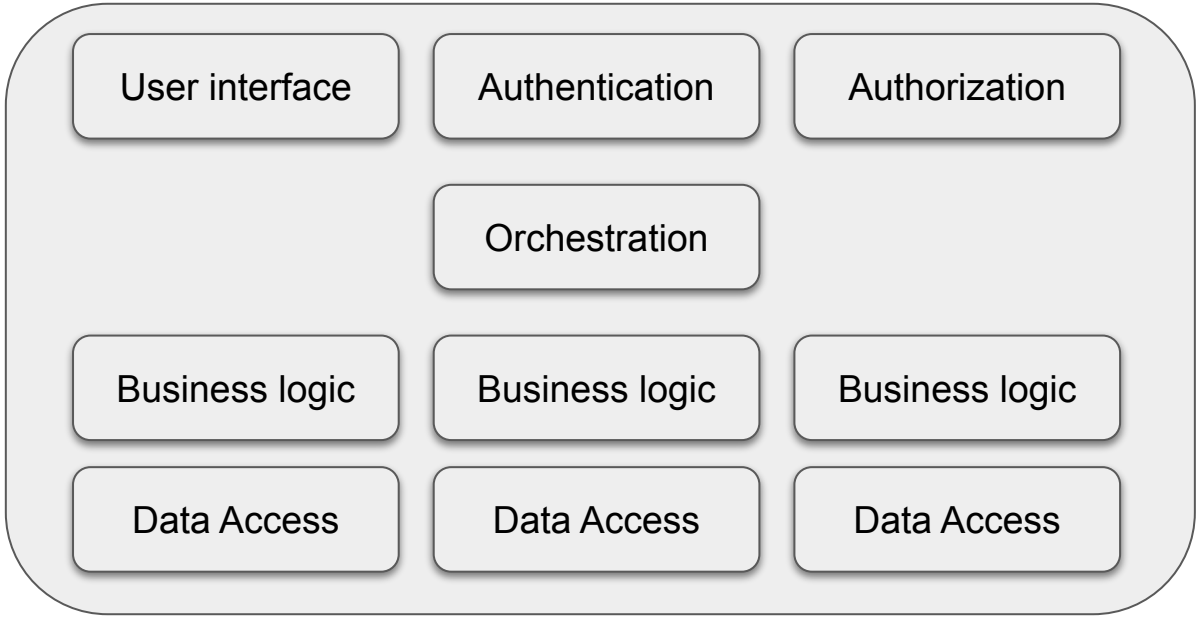
Answer:
It’s complicated



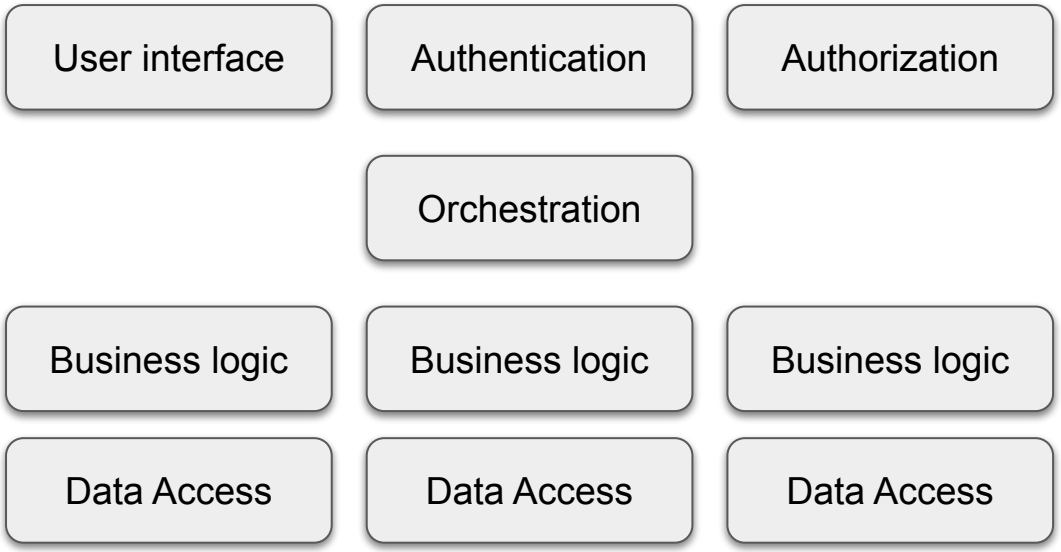
The evolution of identity

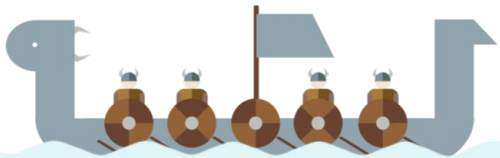


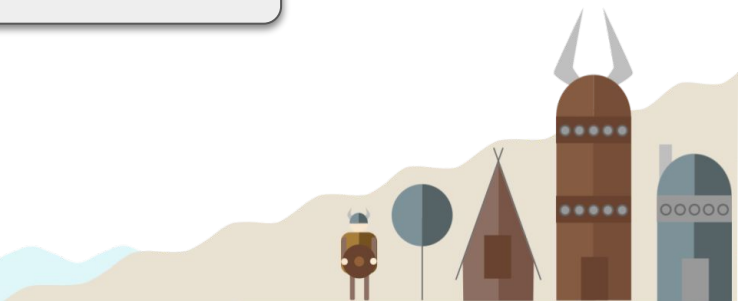
From monolith



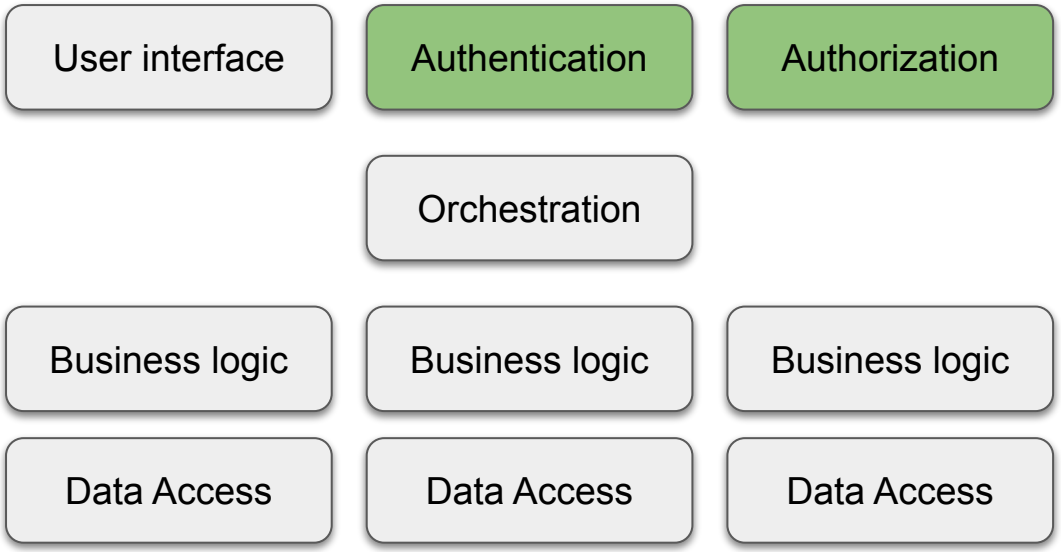
To microservices





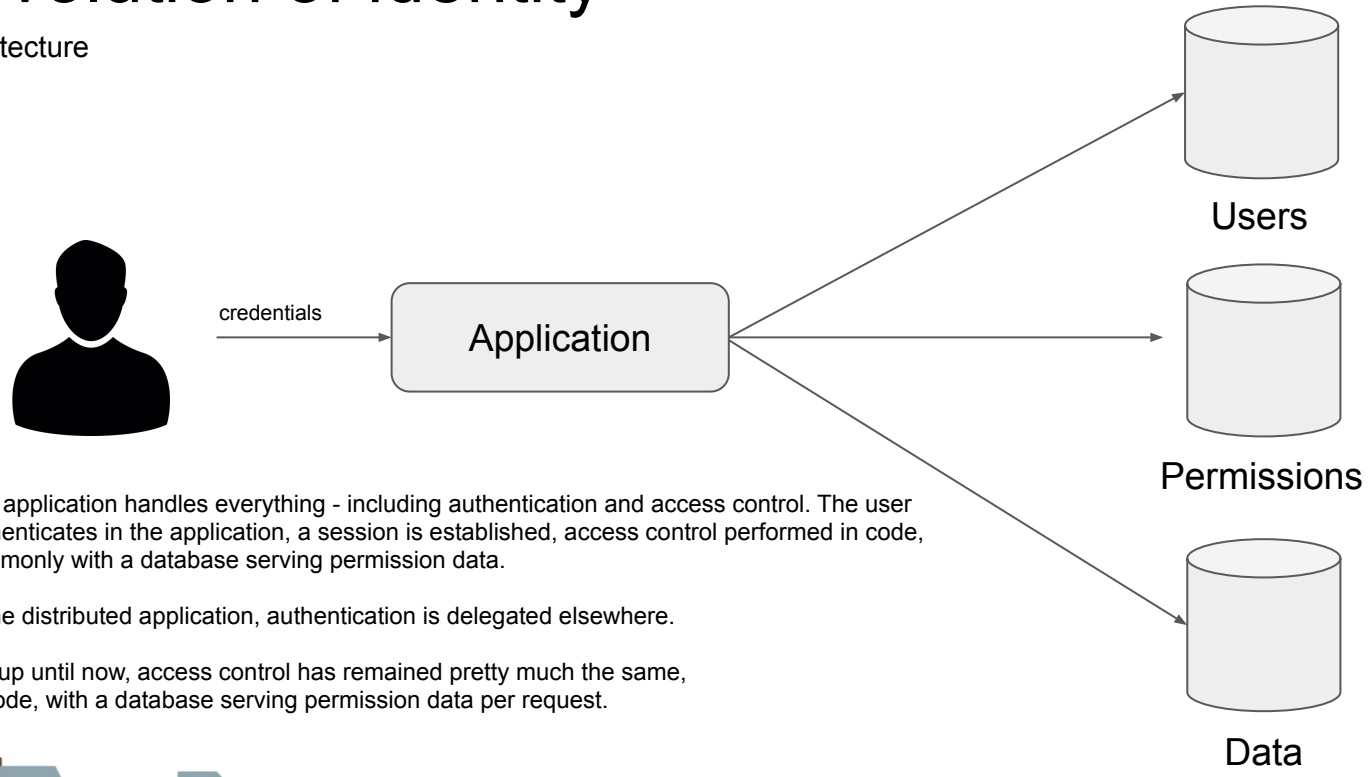


To microservices



The evolution of identity

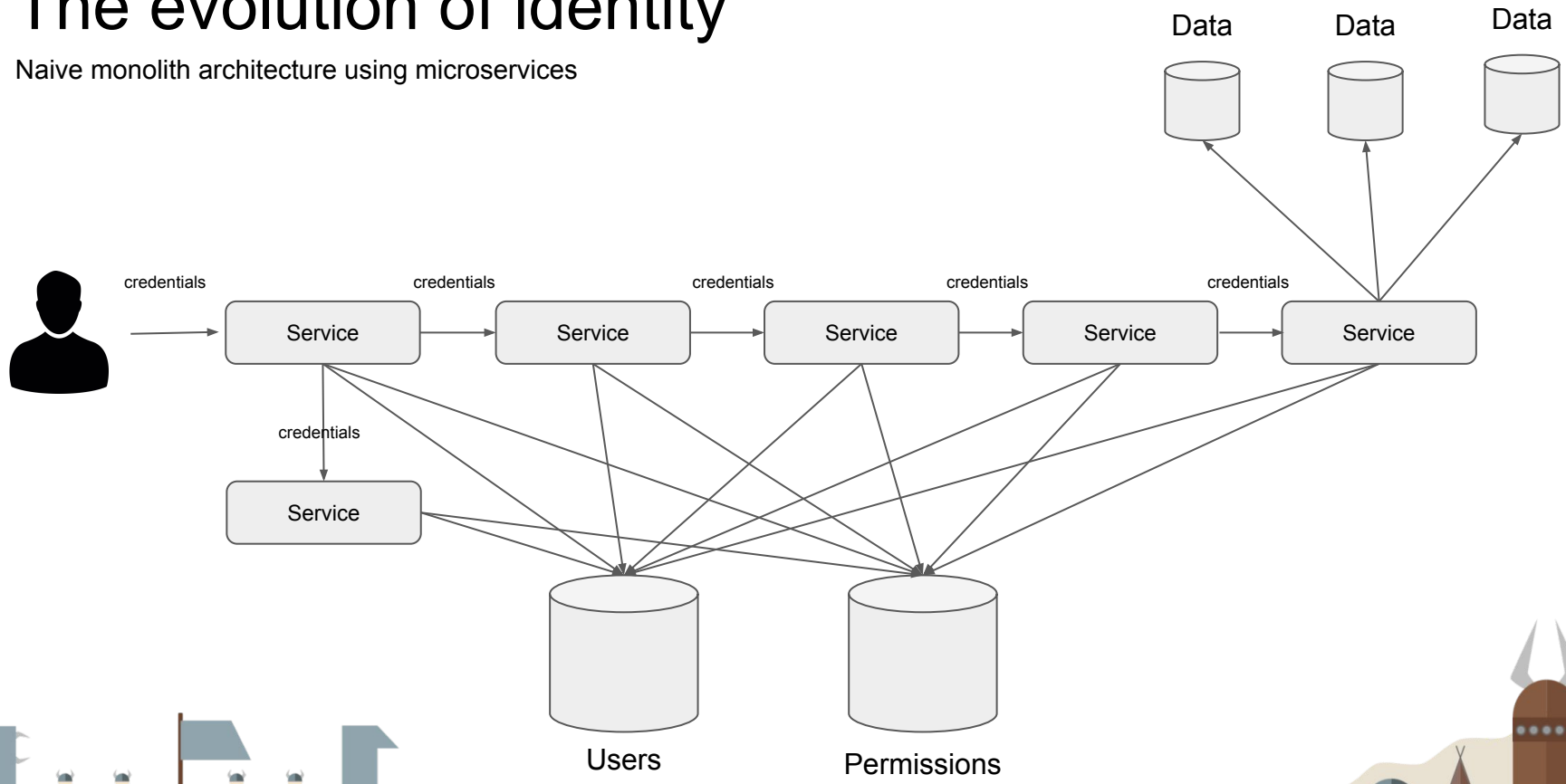
Monolith architecture



- The application handles everything - including authentication and access control. The user authenticates in the application, a session is established, access control performed in code, commonly with a database serving permission data.
- In the distributed application, authentication is delegated elsewhere.
- But up until now, access control has remained pretty much the same, in code, with a database serving permission data per request.

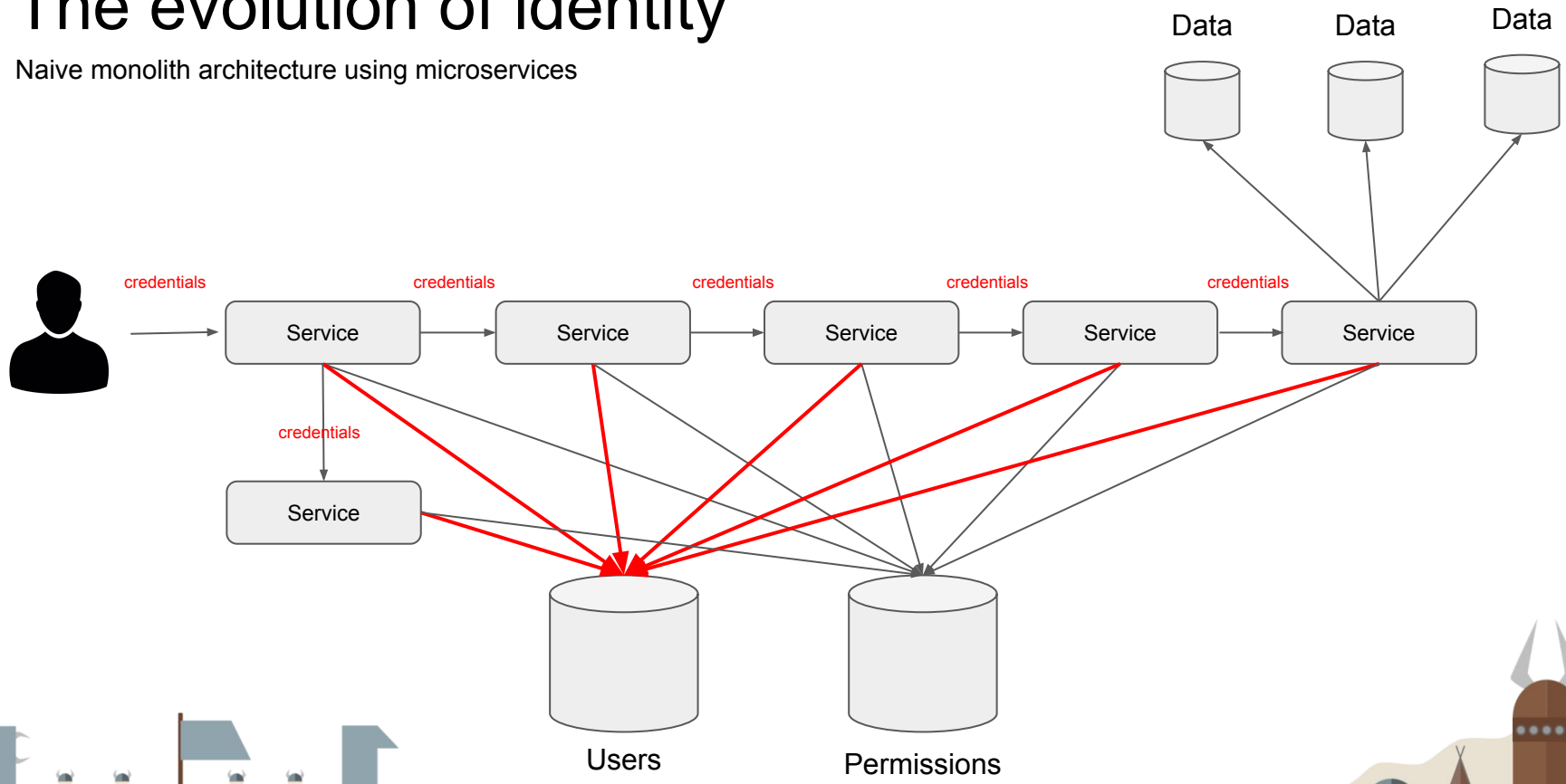
The evolution of identity

Naive monolith architecture using microservices



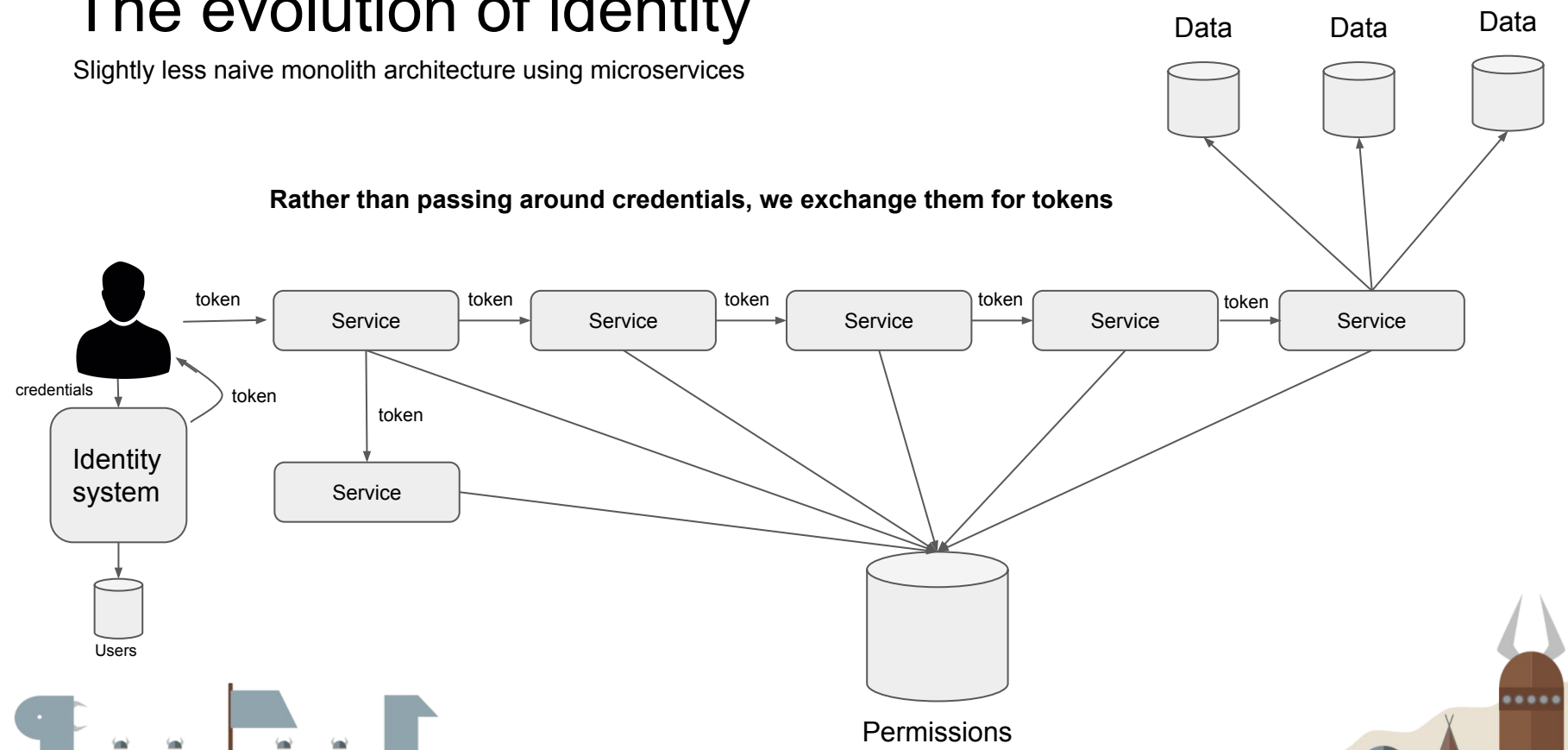
The evolution of identity

Naive monolith architecture using microservices



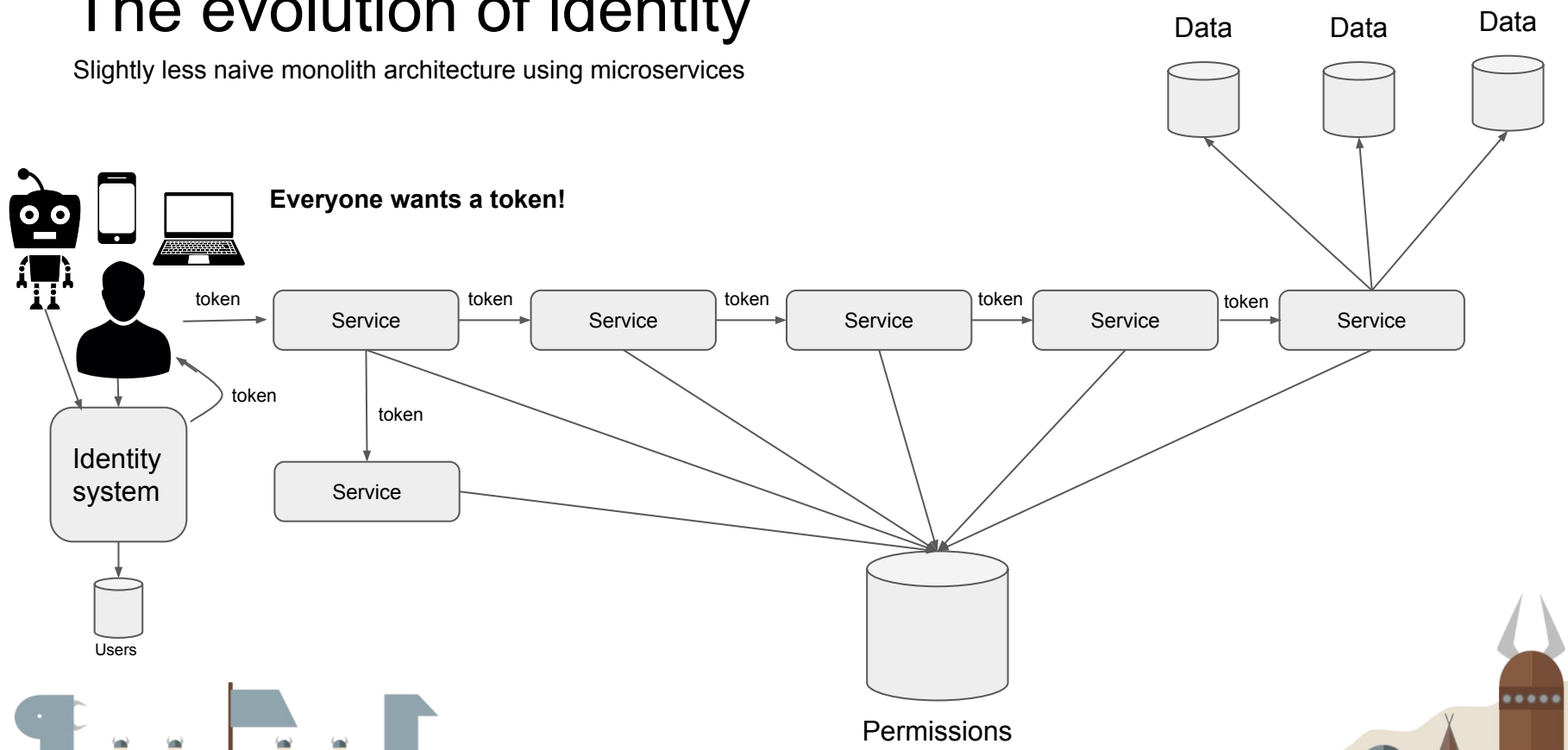
The evolution of identity

Slightly less naive monolith architecture using microservices



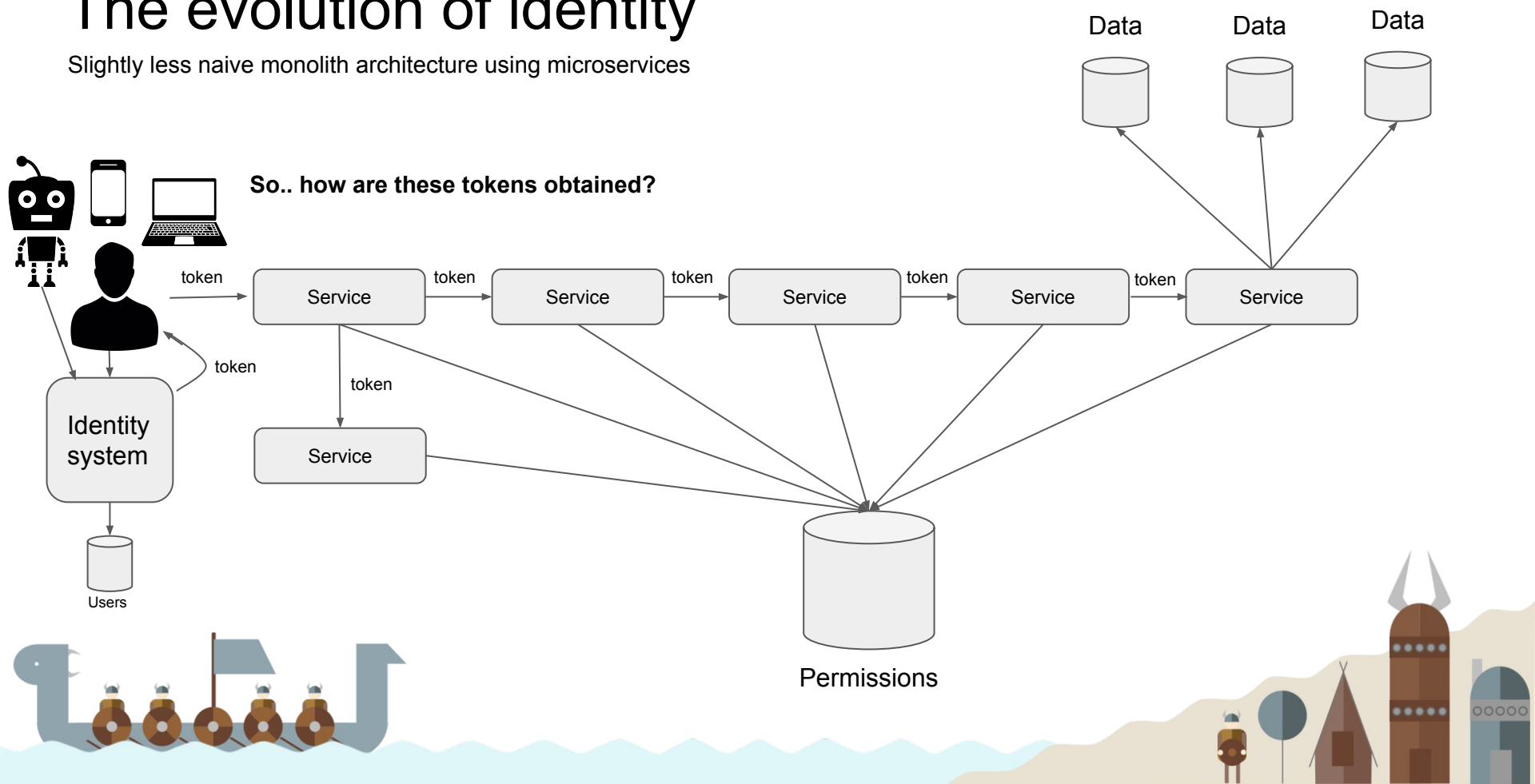
The evolution of identity

Slightly less naive monolith architecture using microservices



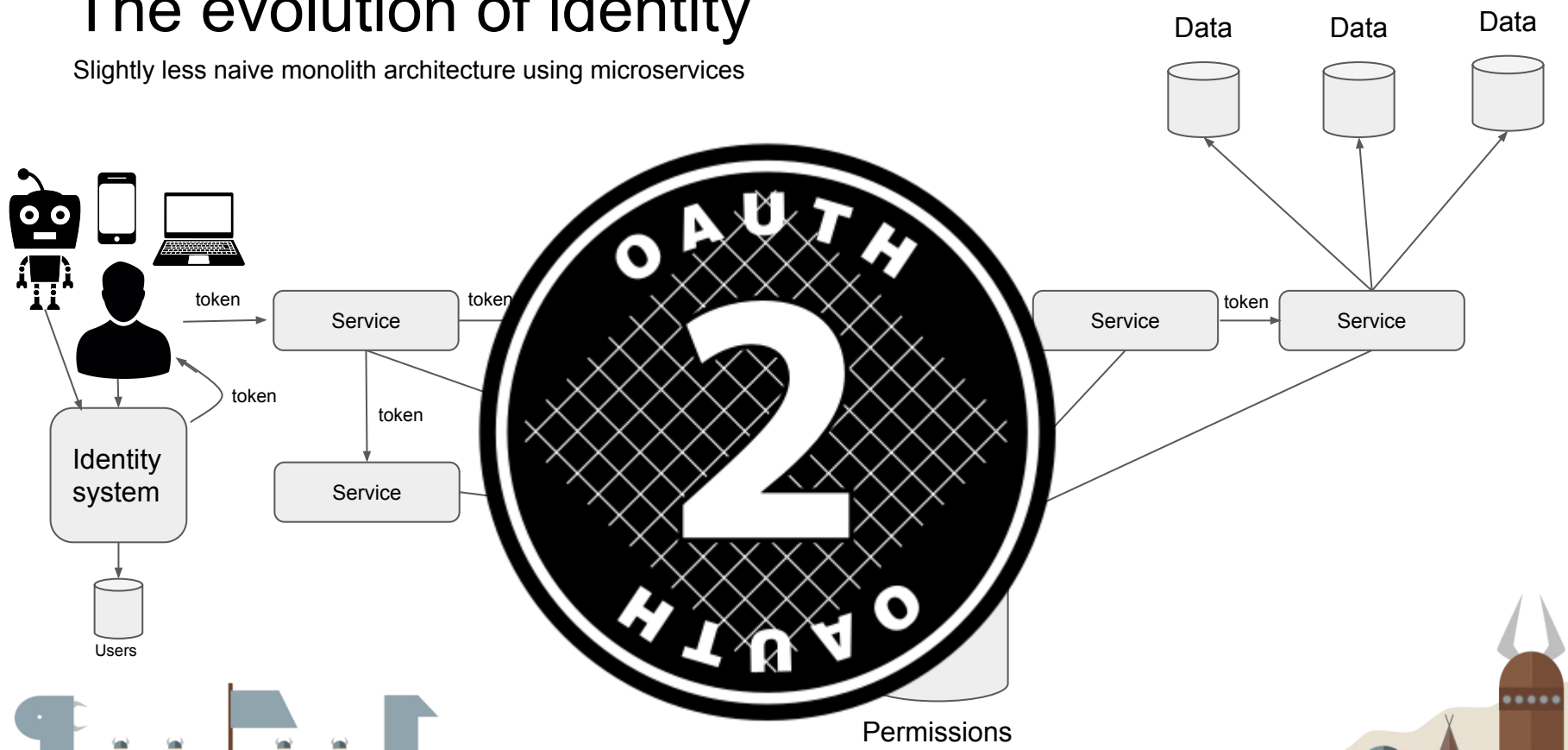
The evolution of identity

Slightly less naive monolith architecture using microservices

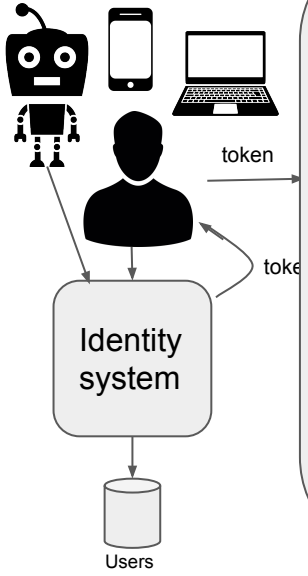


The evolution of identity

Slightly less naive monolith architecture using microservices



The evolution of identity



Data Data Data

OAuth2

Defines a set of flows for users (interactive flows) and clients (non-interactive) to authenticate at the authorization server in order to obtain *access tokens* for use as credentials to services.

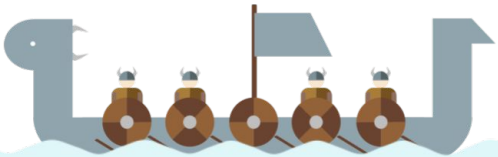
Does not detail what an access token should look like.

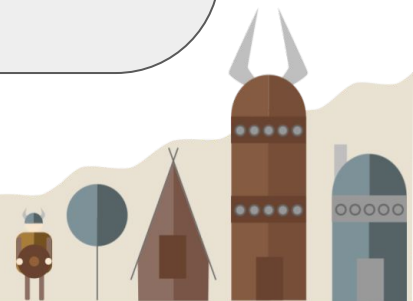
Despite labeled an “authorization framework” provides little in terms of authorization - rather about *delegation*.

Scopes provide basic boundaries for where an access token may be used.

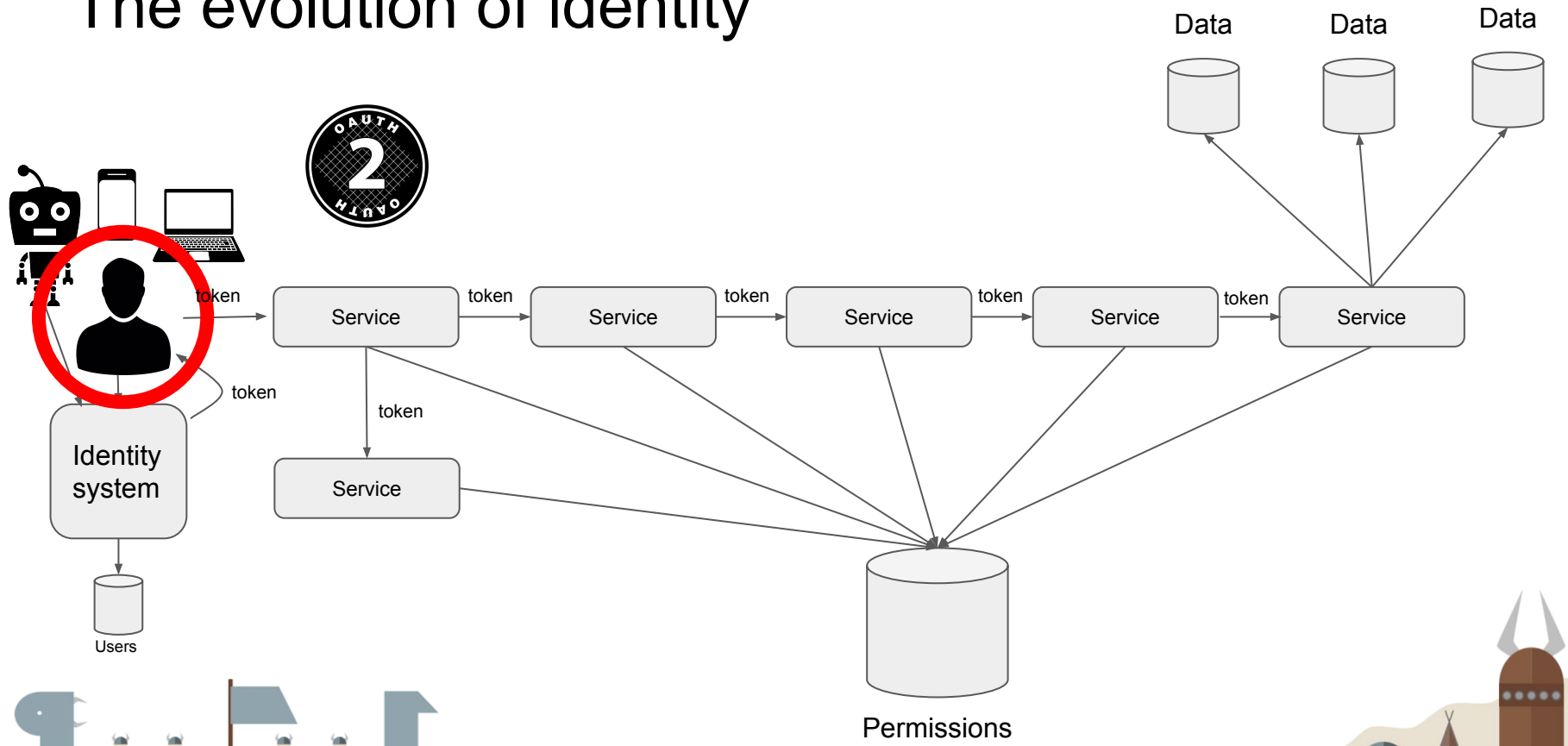
Commonly used for external identity providers, “social login”, etc.

Permissions

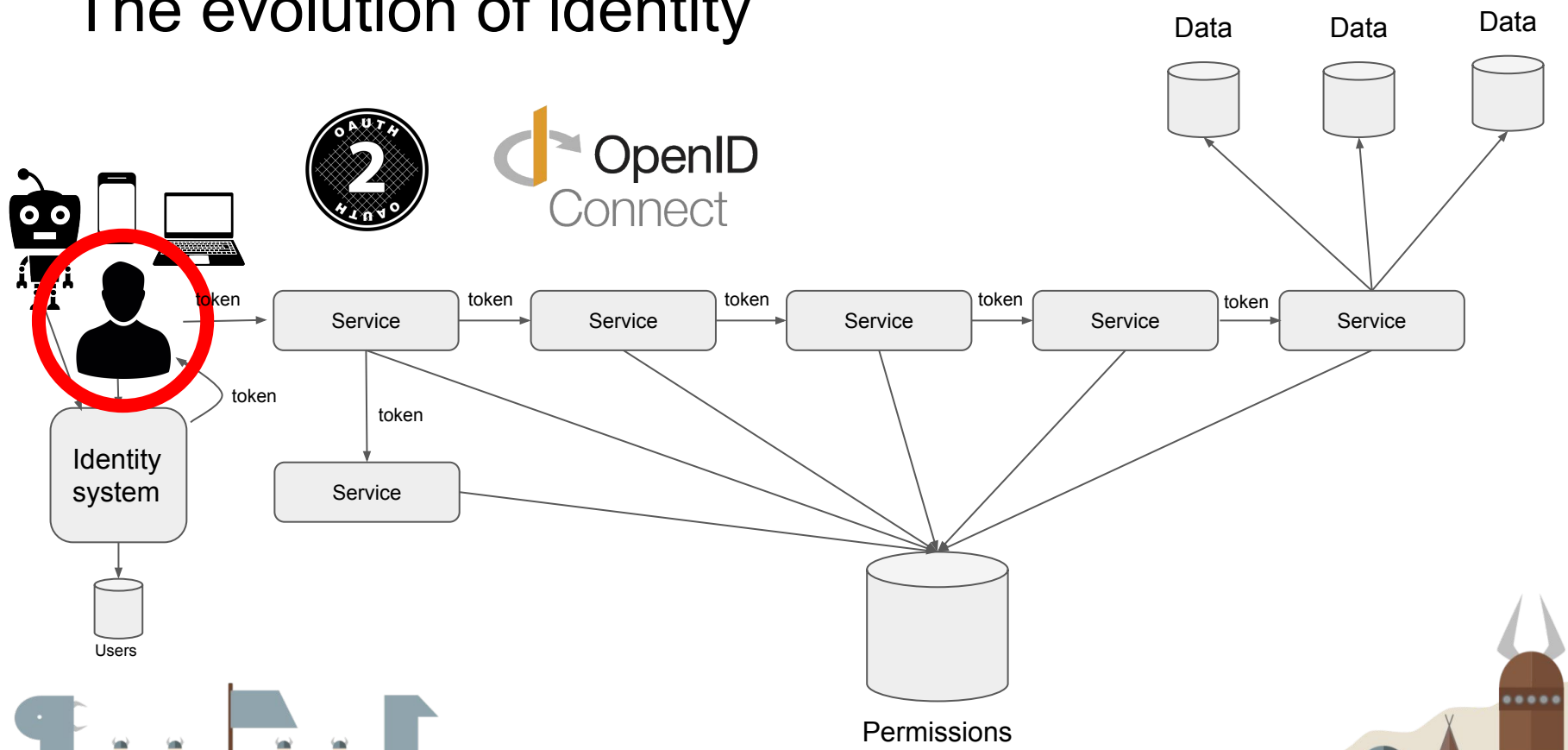




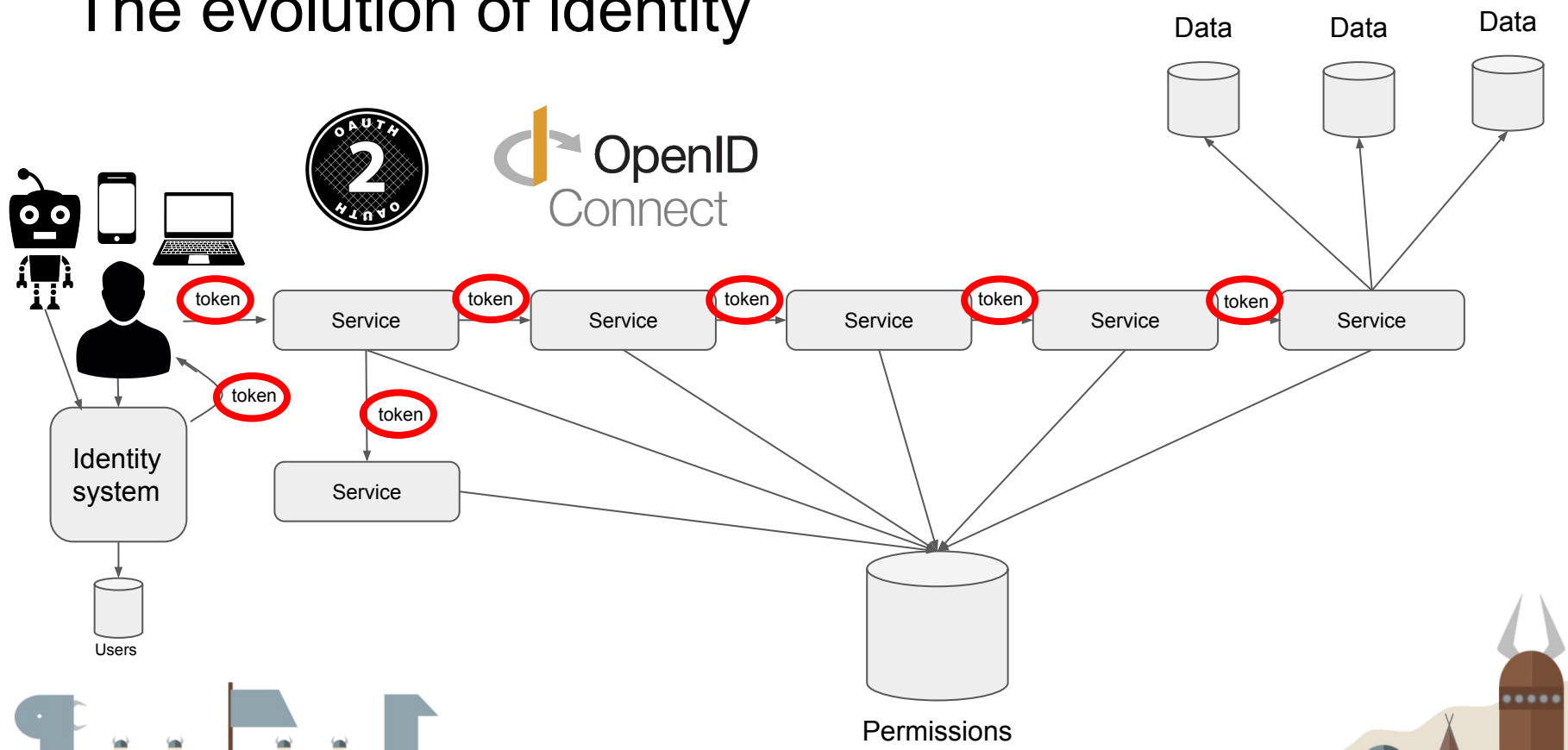
The evolution of identity



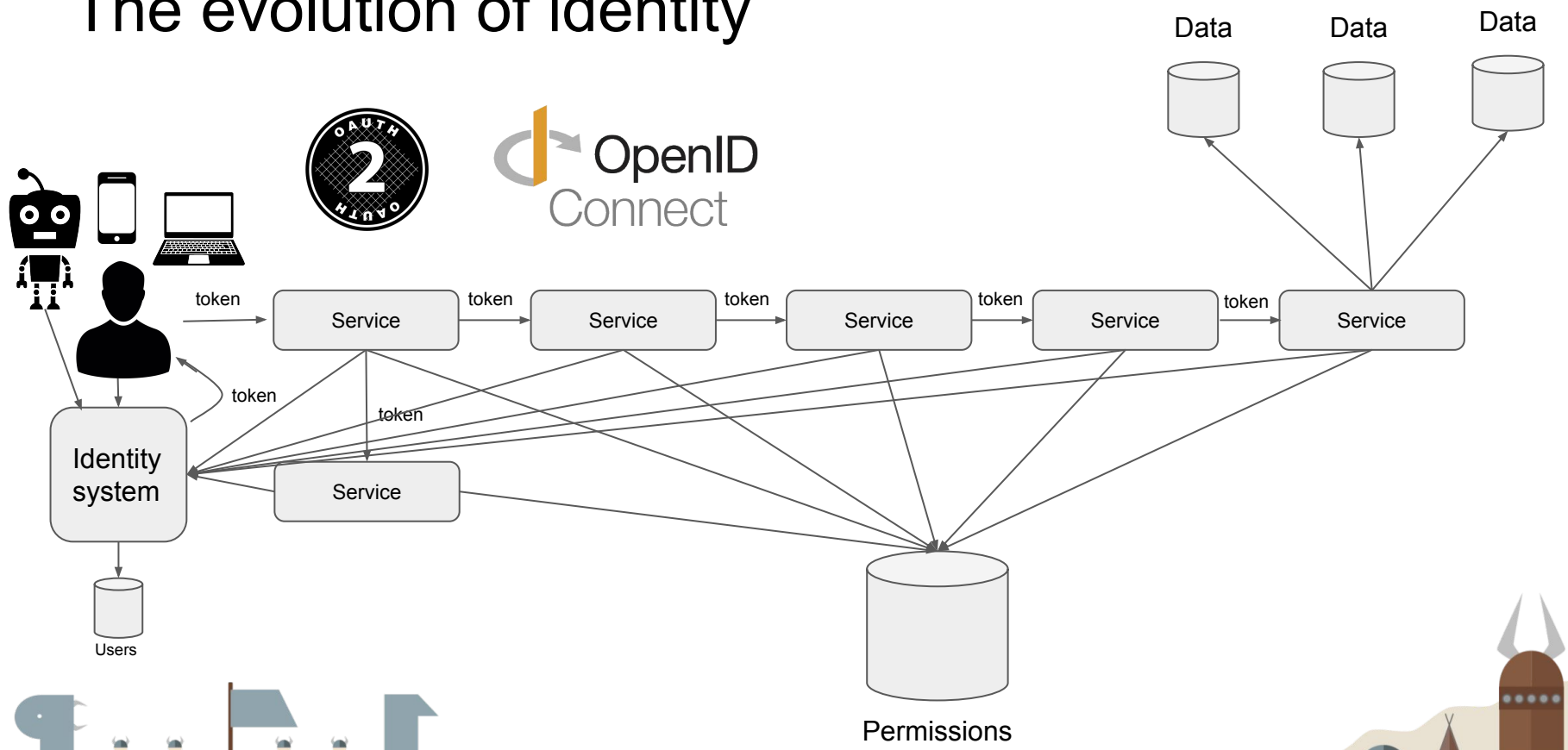
The evolution of identity



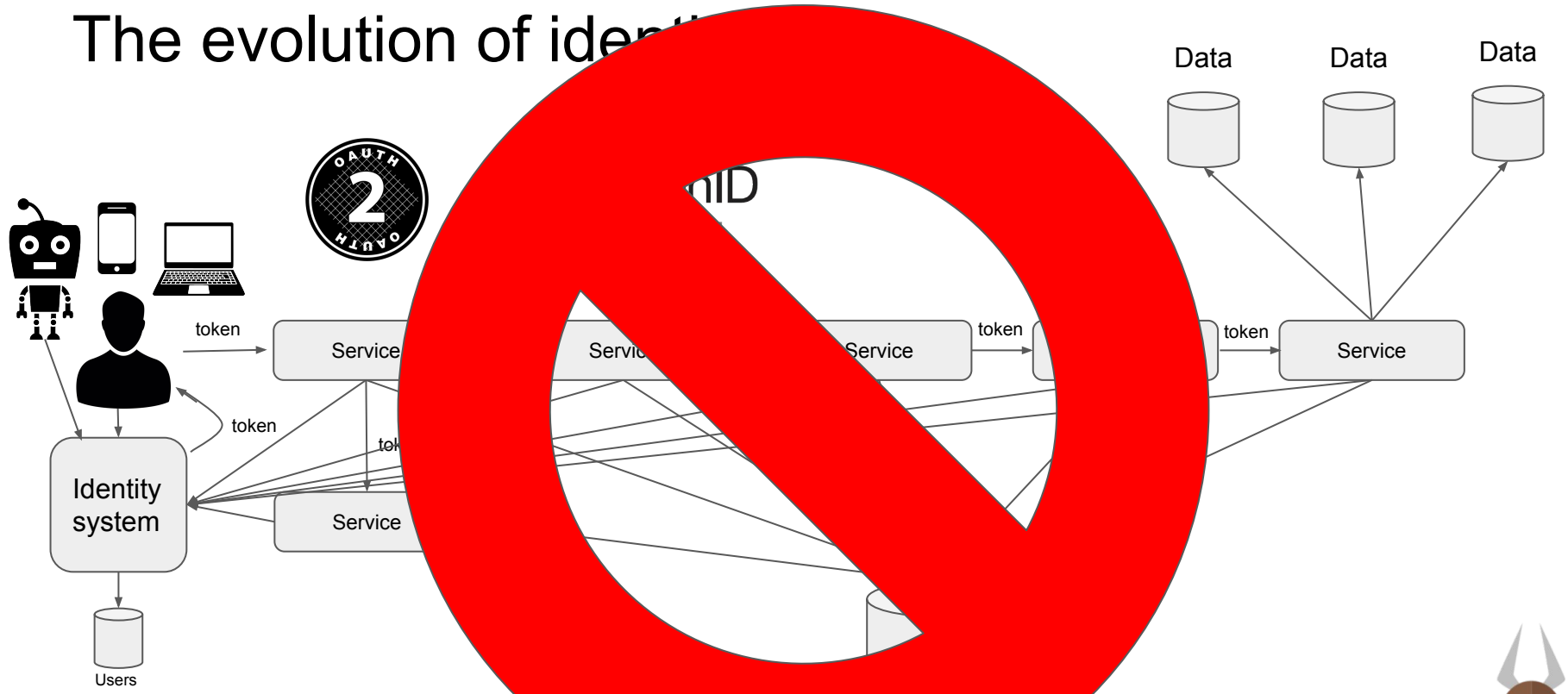
The evolution of identity



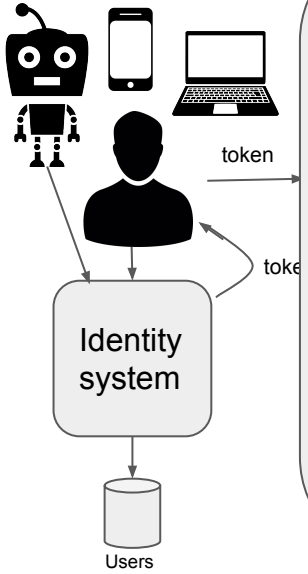
The evolution of identity



The evolution of iden



The evolution of identity



Data Data Data

JSON Web Tokens (JWTs)

A JWT is a signed self-contained collection of claims, i.e. attributes claimed to be true.

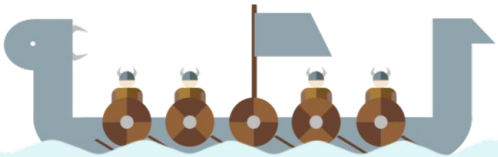
Tokens are created by an issuer. Claims often (but not always) provided by the backing identity provider.

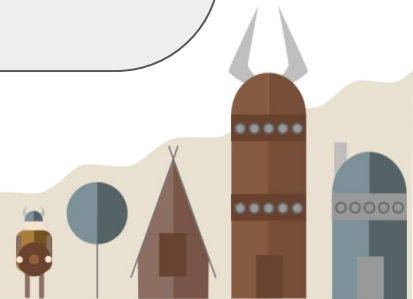
Expiry time (and other standard attributes) of JWT included in payload.

JWTs are immutable - no claim may be changed without breaking signature verification.

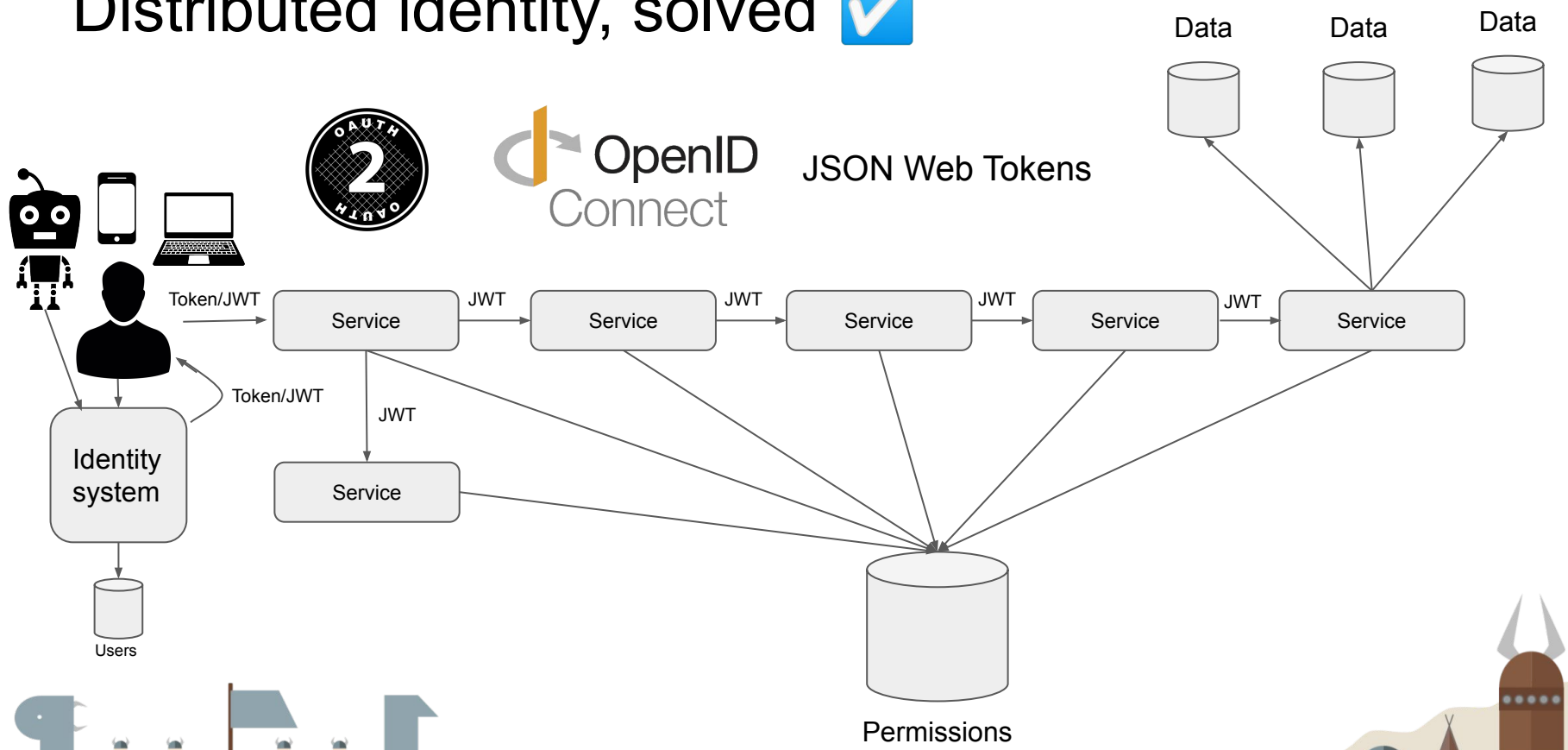
JWTs are everywhere - libraries for both encoding and decoding available for all languages and platforms.

Permissions



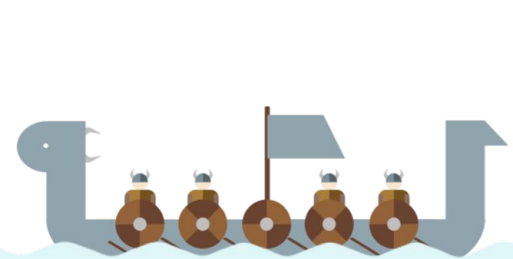


Distributed identity, solved ☑️

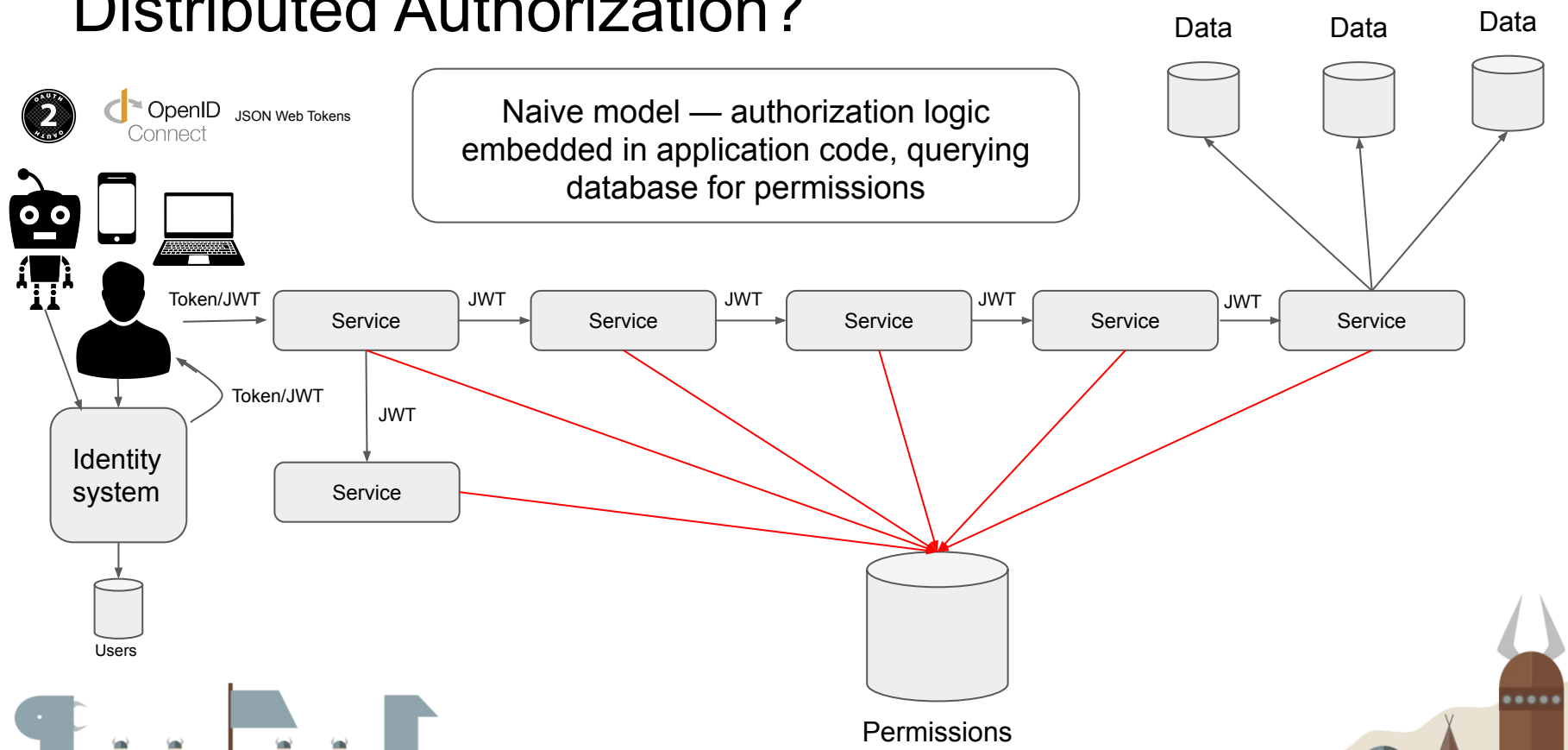


Great, now do authorization

The evolution of access control

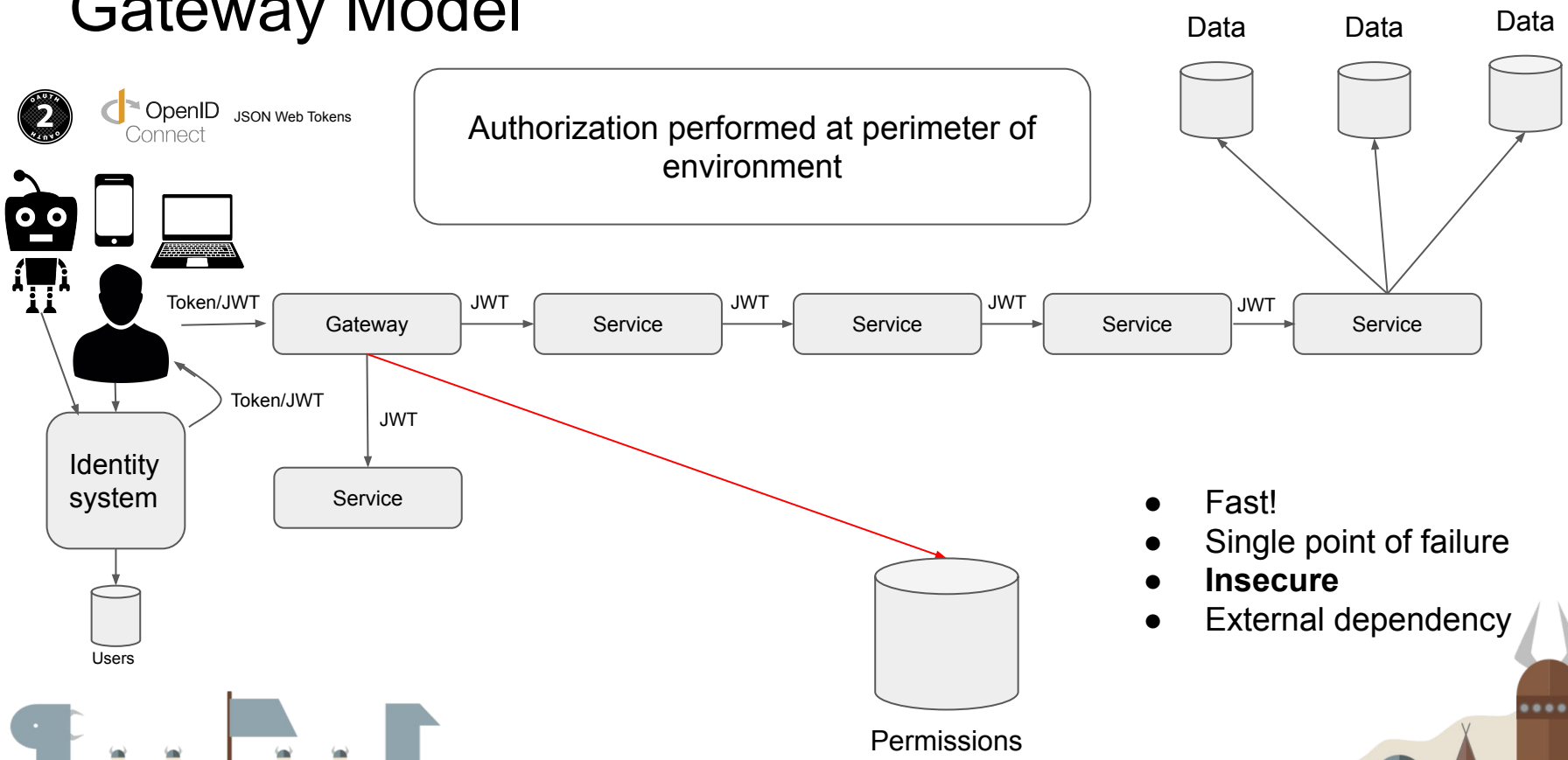


Distributed Authorization?



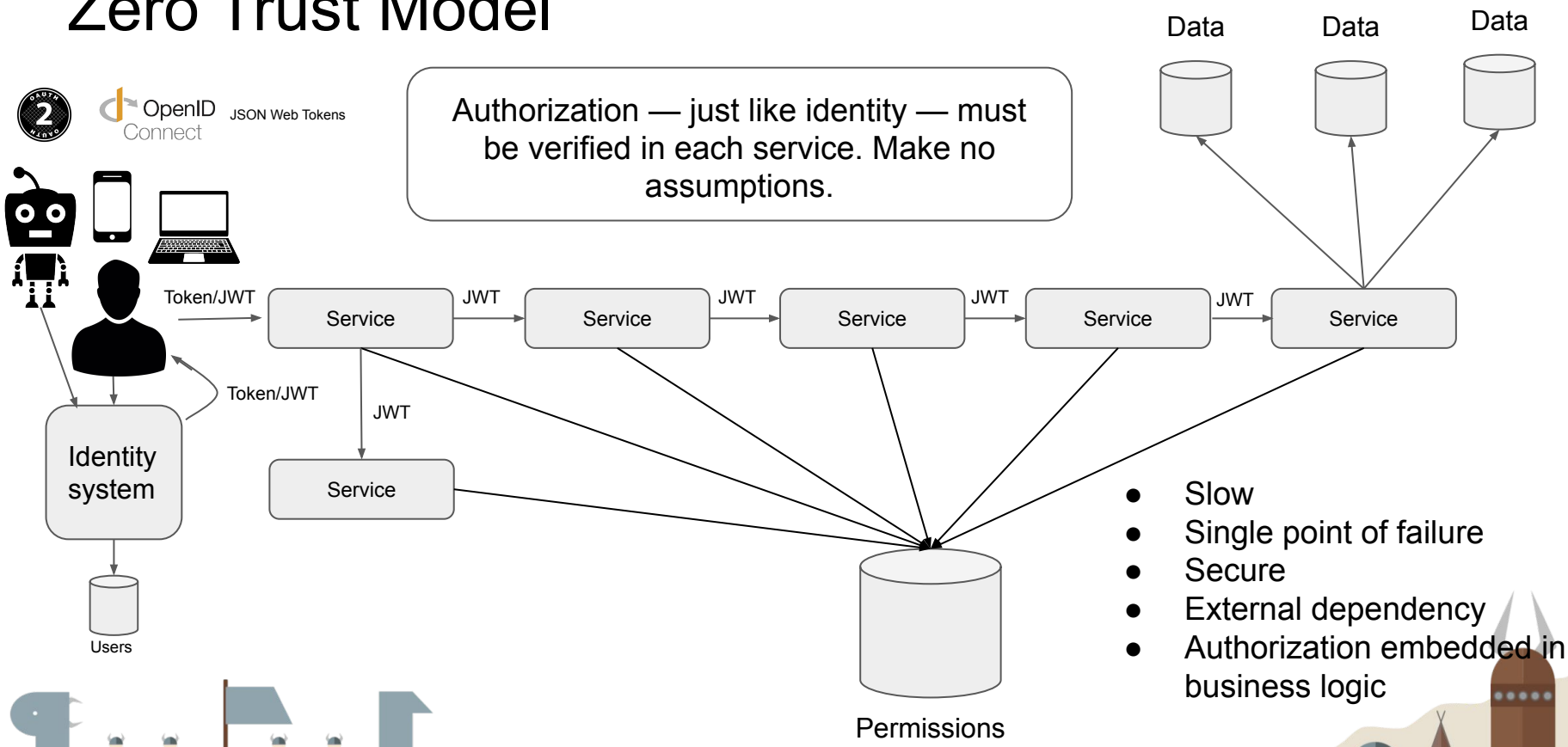
So, where should we do authorization?

Gateway Model



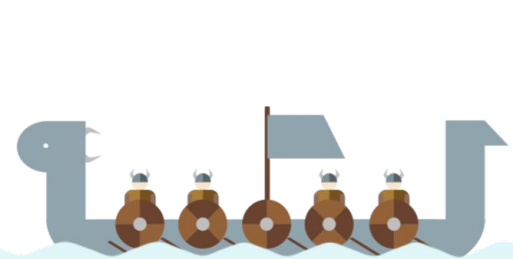
- Fast!
- Single point of failure
- **Insecure**
- External dependency

Zero Trust Model

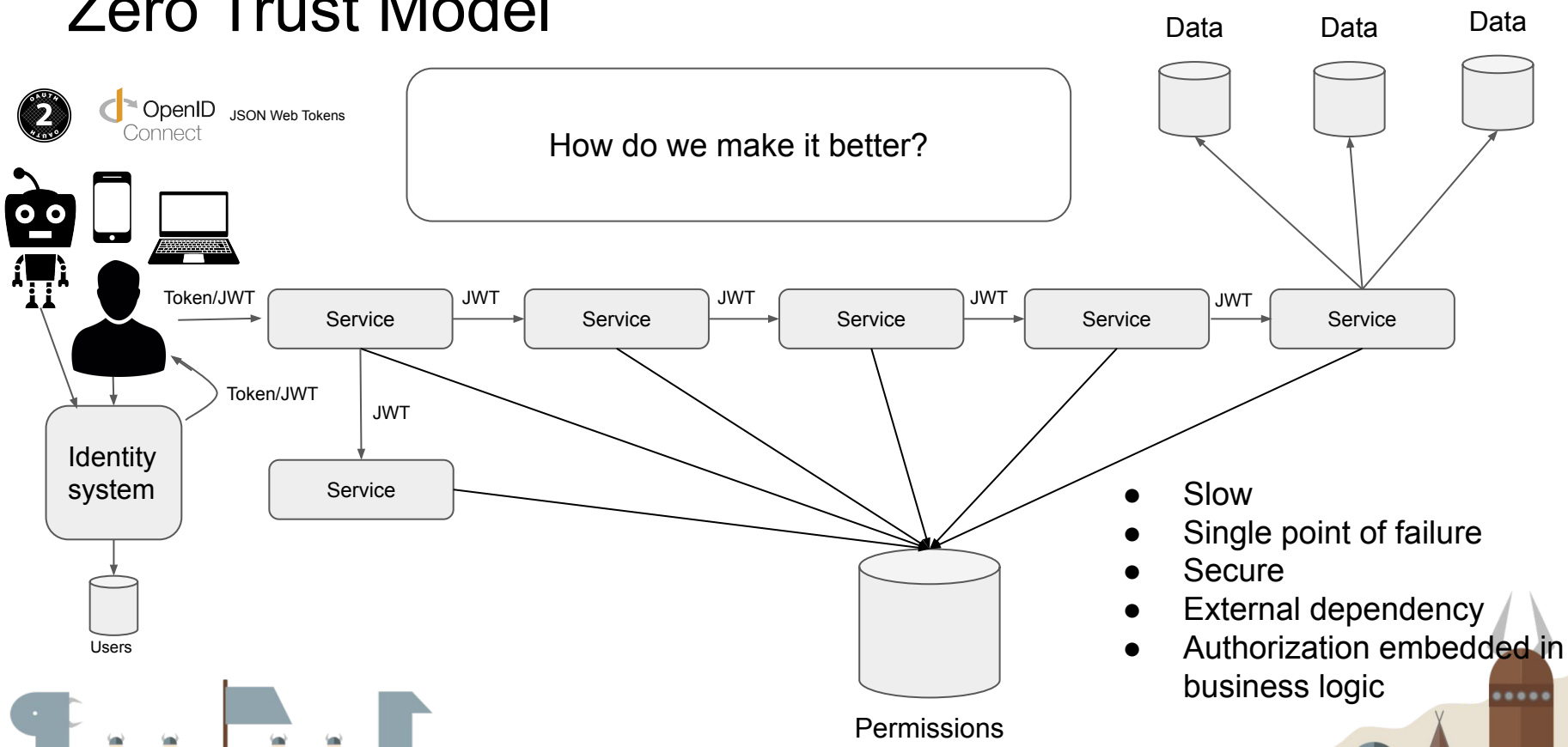


- Slow
- Single point of failure
- Secure
- External dependency
- Authorization embedded in business logic

Great, we’re back where we started

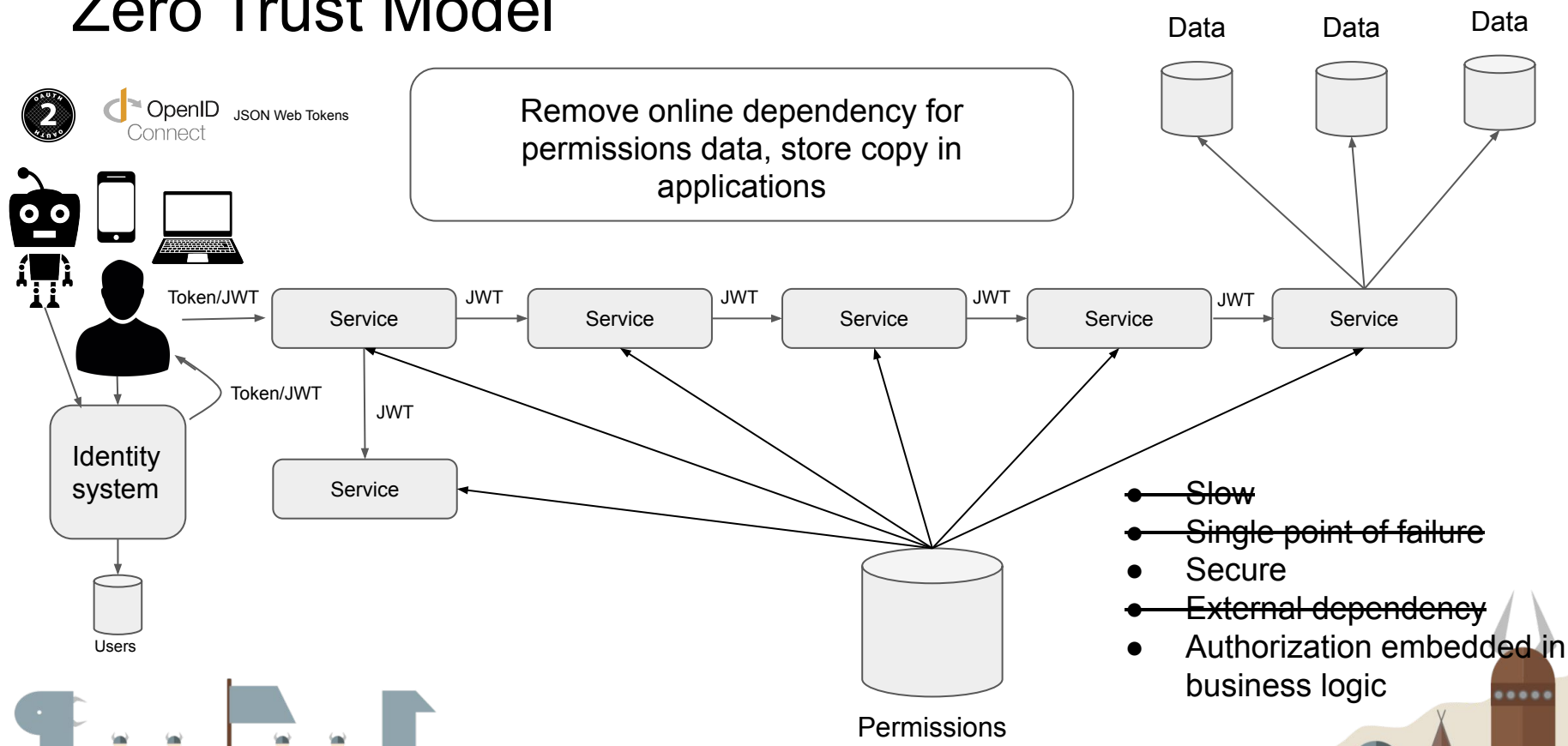


Zero Trust Model

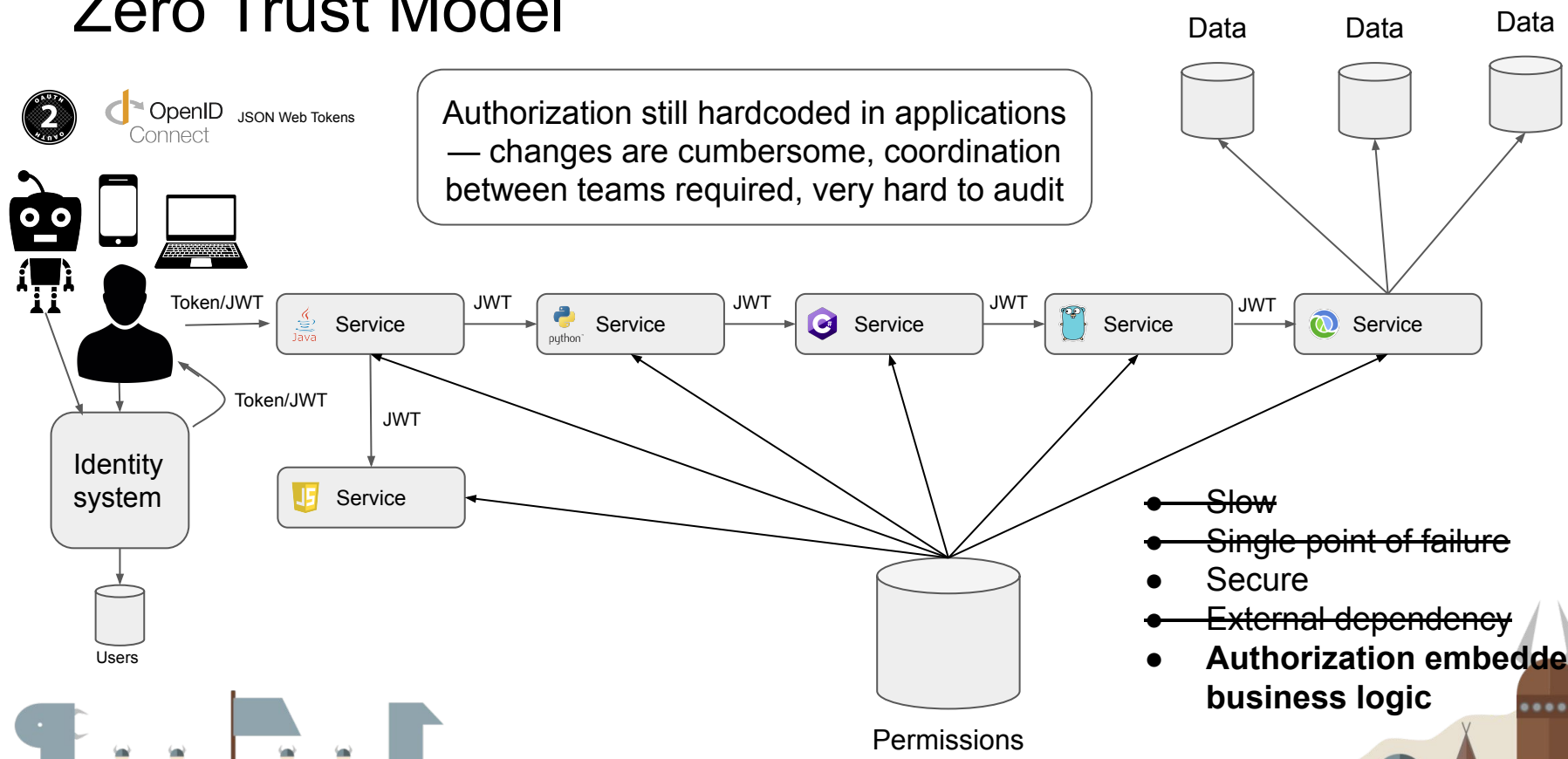


- Slow
- Single point of failure
- Secure
- External dependency
- Authorization embedded in business logic

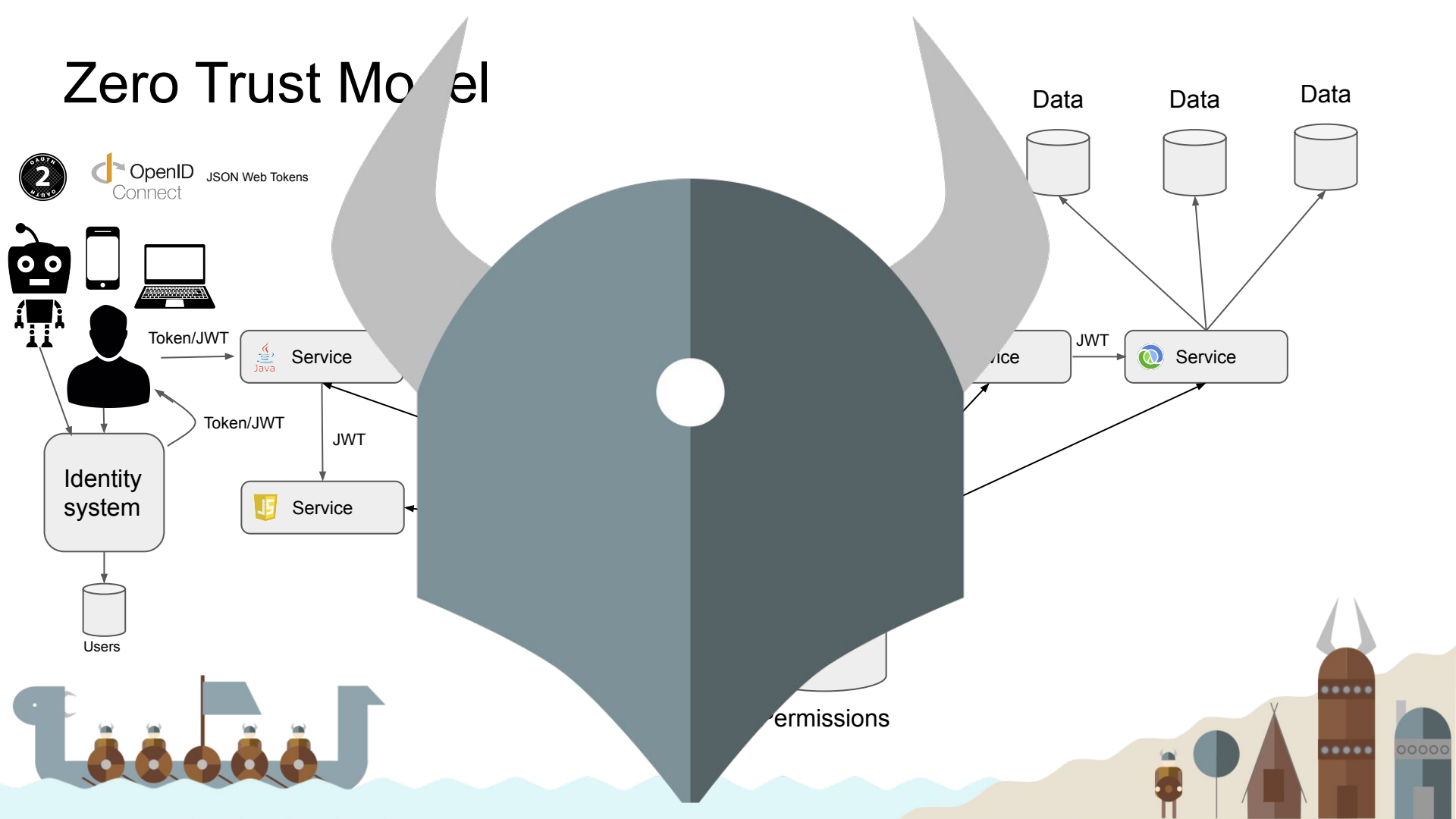
Zero Trust Model



Zero Trust Model



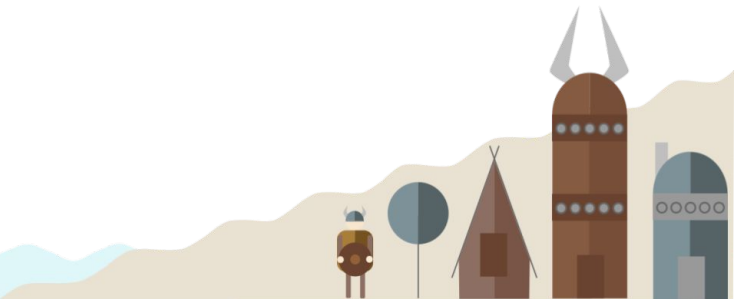
Zero Trust Model



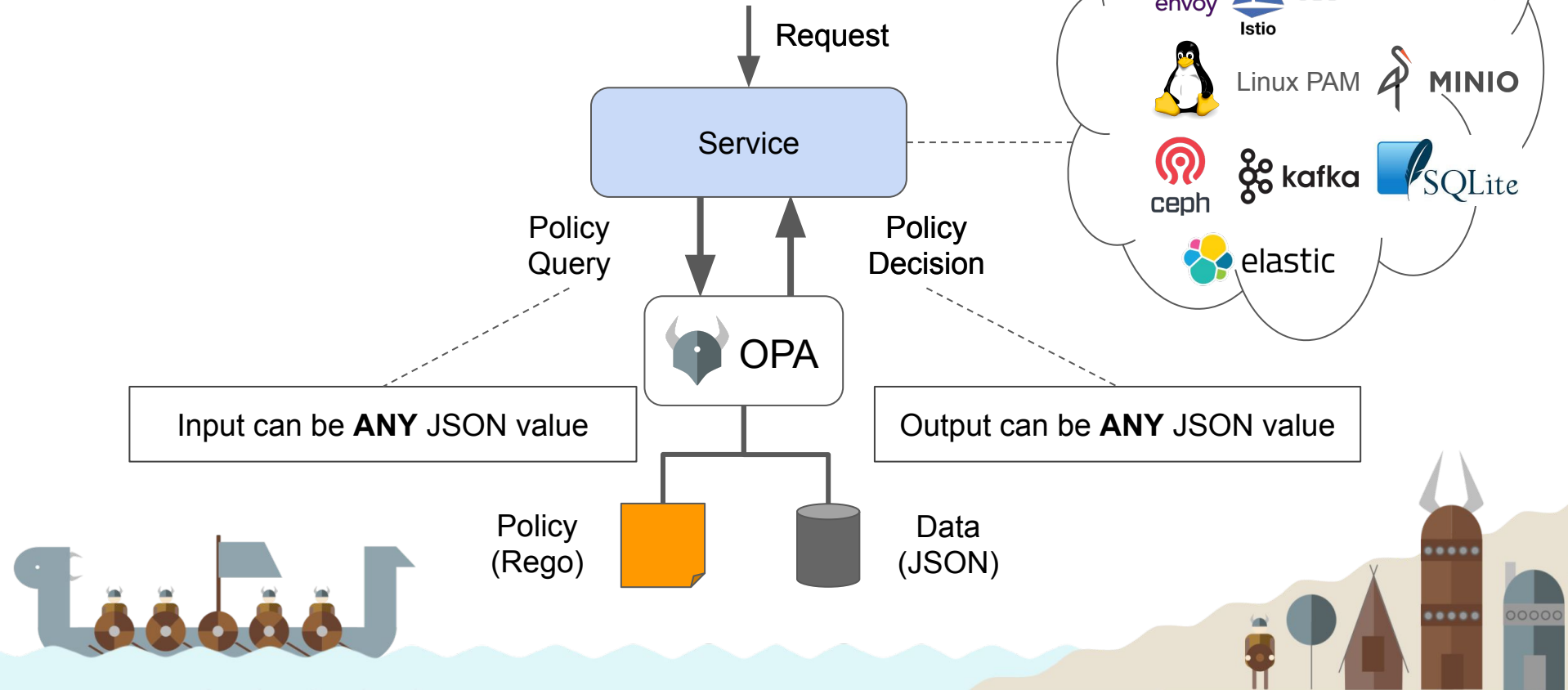
- Open source general purpose policy engine
- Unified toolset and framework for policy across the stack
- Decouples policy from application logic
- Separates policy *decision* from *enforcement*
- Policies written in declarative language Rego
- Popular use cases ranging from kubernetes admission control, microservice authorization, infrastructure, data source filtering, to CI/CD pipeline policies and many more.

Kelsey Hightower
@kelseyhightower

The Open Policy Agent project is super dope! I finally have a framework that helps me translate written security policies into executable code for every layer of the stack.



Policy decision model

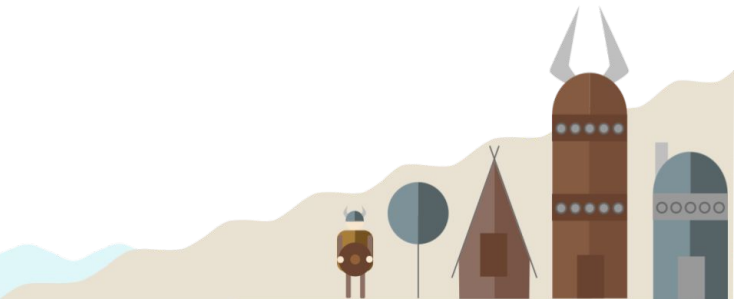


Deployment model

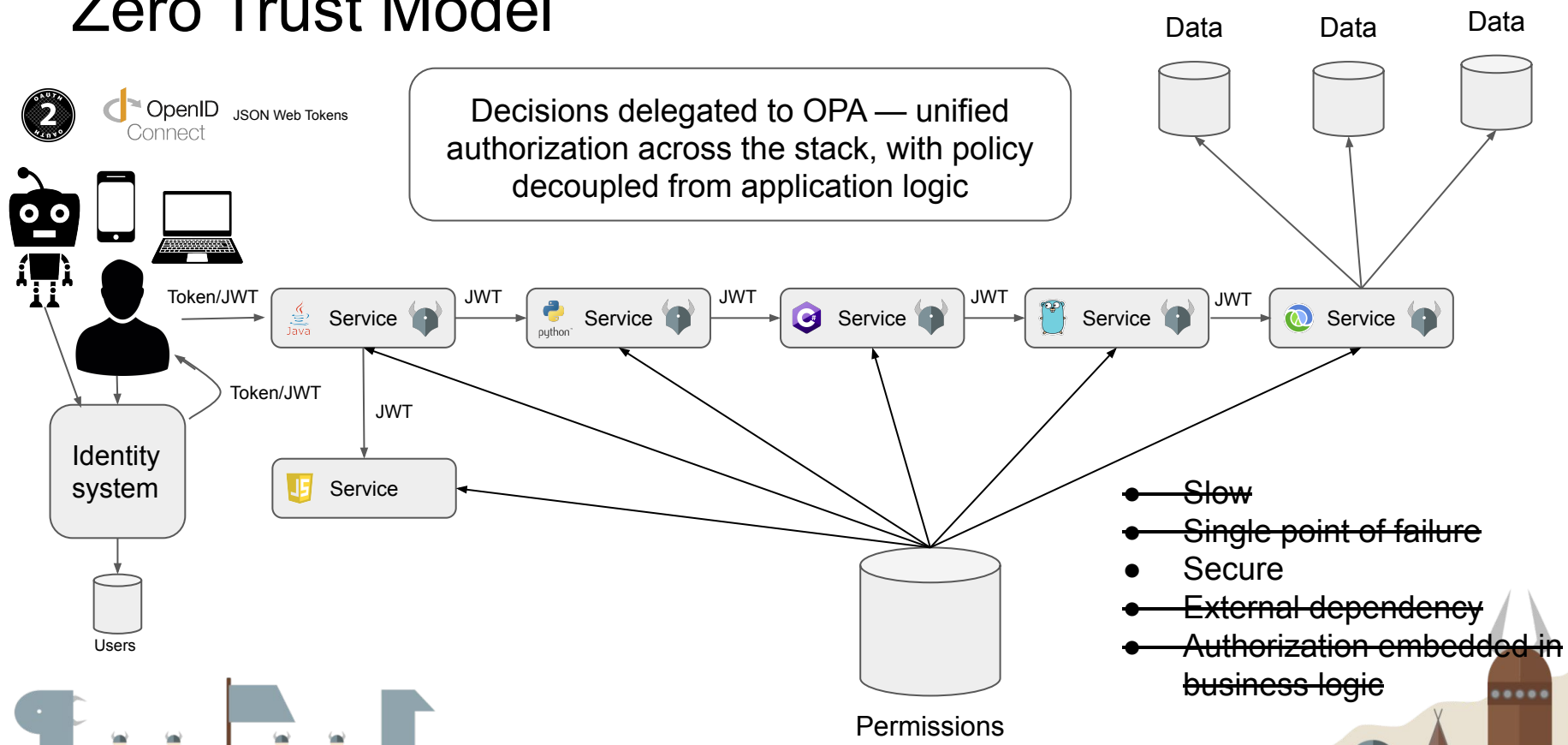
- OPA runs as a lightweight self-contained server binary
- OPA ideally deployed as close to service as possible. This usually means running on the same host, either as a daemon or in a sidecar deployment
- Applications communicate with the OPA server through its REST API
- Go library available for Go applications
- Envoy/Istio based applications. Wasm, Intermediate Representation (IR), more...

Policy authoring and Rego

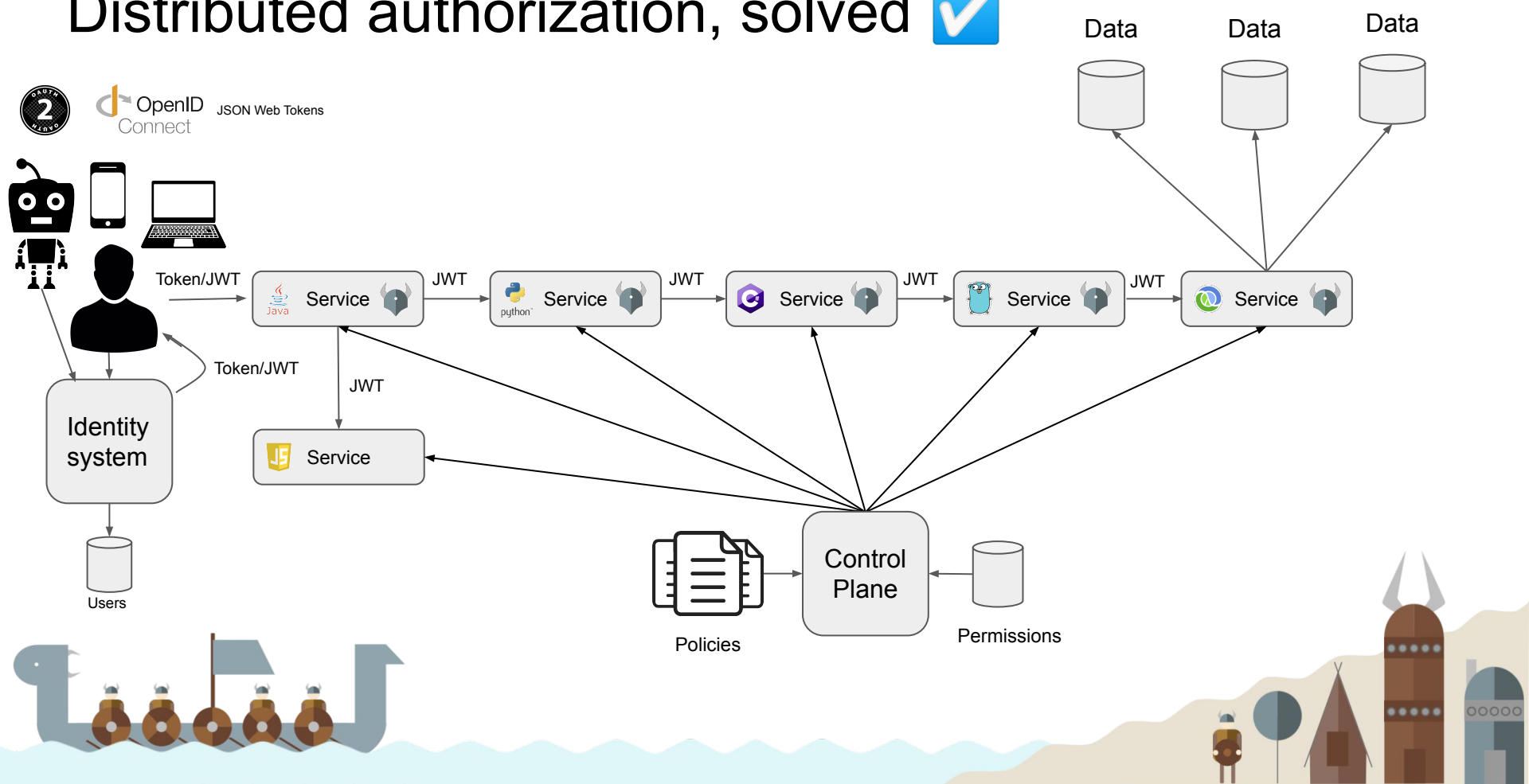
- Rego — declarative high-level policy language used by OPA.
- Policy consists of any number of rules.
- Rules commonly return true/false but may return any type available in JSON, like strings, lists and objects.
- Policy testing is easy with provided unit test framework.
- Well documented! https://www.openpolicyagent.org/docs/latest/
- Try it out! https://play.openpolicyagent.org/



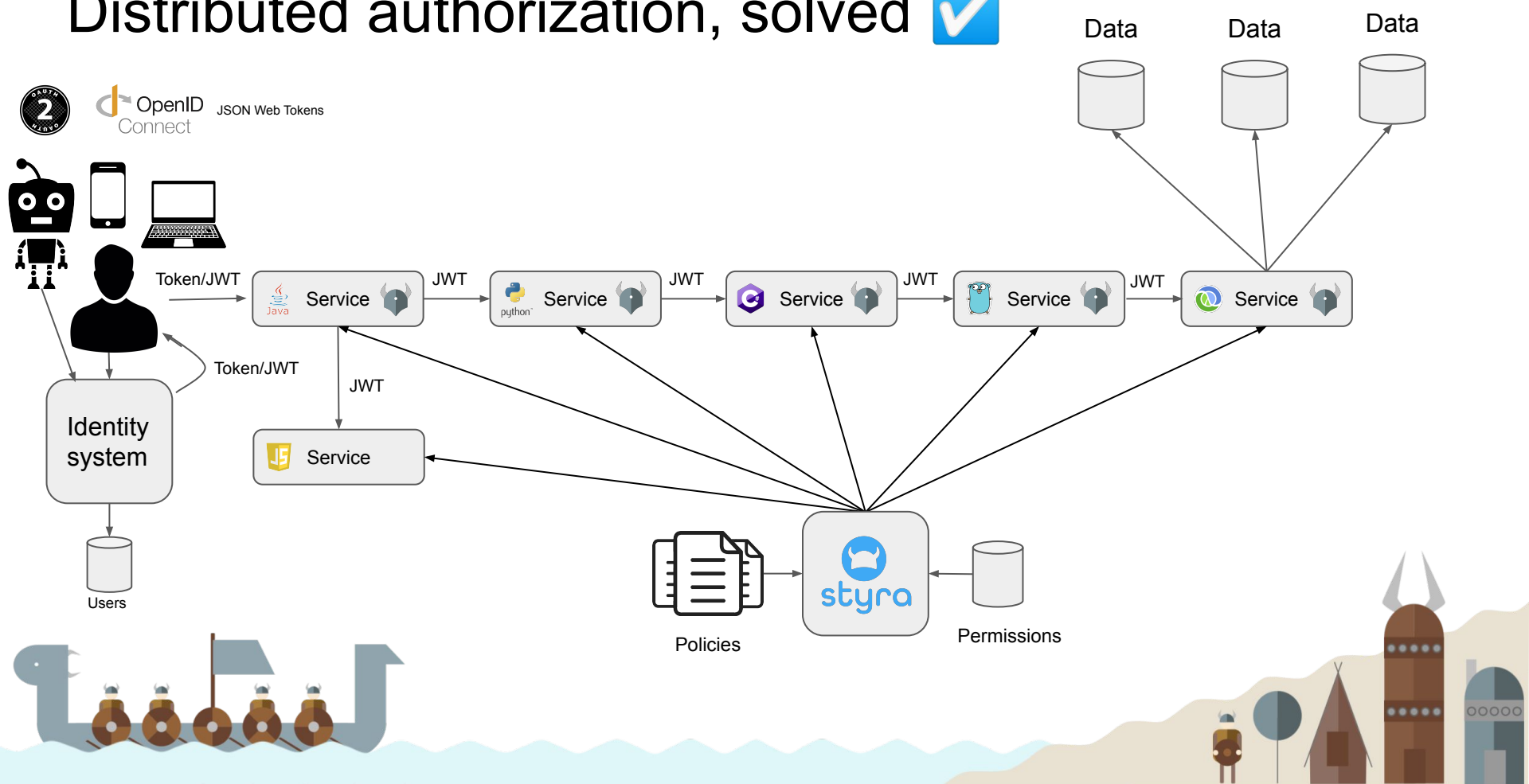
Zero Trust Model



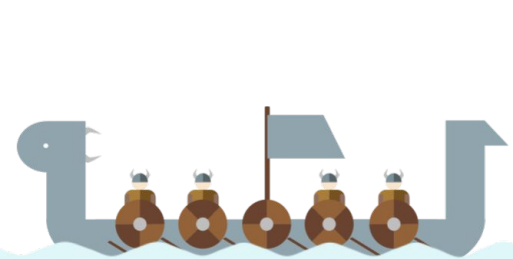
Distributed authorization, solved ☑️



Distributed authorization, solved ☑️

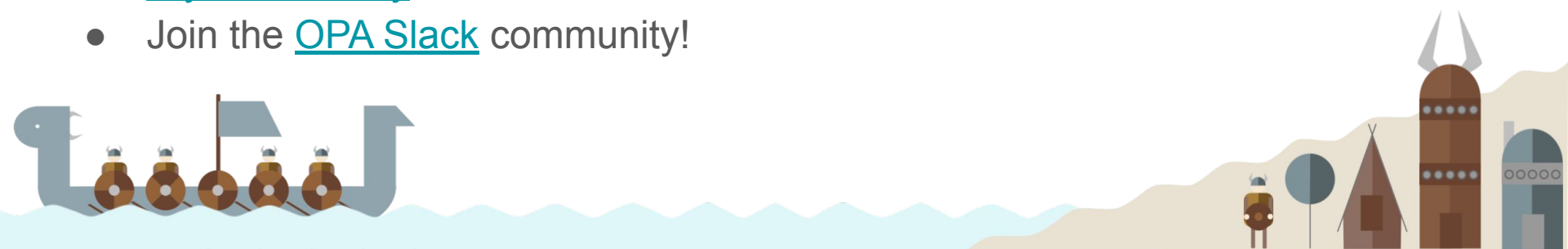


Distributed authorization solved!



Getting started

- Start small – write a few simple policies and tests.
- Browse the OPA documentation. Get a feel for the basics and the built-ins.
- Consider possible applications near to you - previous apps and libraries you've worked with. Consider the informal policies it dealt with.
- Delegate policy responsibilities to OPA. Again, start small! Perhaps a single endpoint to begin somewhere. Deploy and build experience.
- Scale up - management capabilities, logging, bundle server - Styra DAS!
- Styra Academy
- Join the OPA Slack community!



Questions?

Thank you!