Policy-based access control

An introduction to Open Policy Agent

Challenge:

Manage policy in increasingly distributed, complex and heterogeneous systems

Challenge:

Manage policy in increasingly distributed, complex and heterogeneous systems

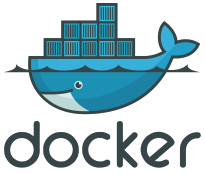






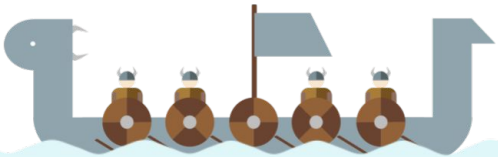





Challenge:

Manage policy in increasingly distributed, complex and heterogeneous systems



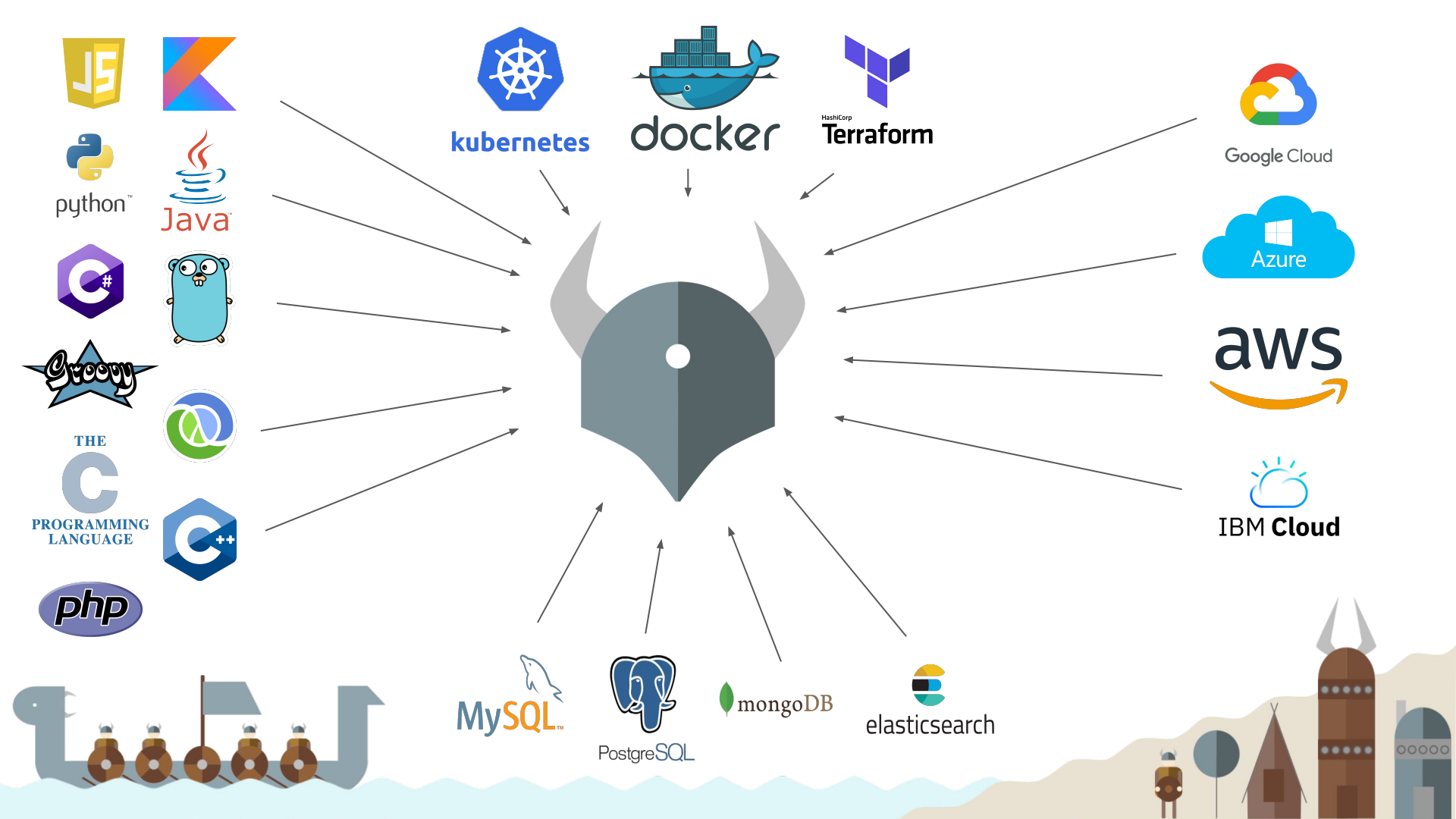

Challenge:

Manage policy in increasingly distributed, complex and heterogeneous systems

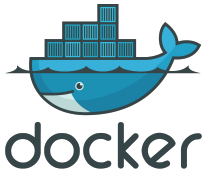
Challenge:

Manage policy in increasingly distributed, complex and heterogeneous systems

Goal:
Unify policy enforcement across the stack


kubernetes
docker
HashiCorp
Terraform
Google Cloud
Azure
aws
IBM Cloud
python
Java
Groovy
THE
C
PROGRAMMING
LANGUAGE
php
MySQL
PostgreSQL
mongoDB
elasticsearch



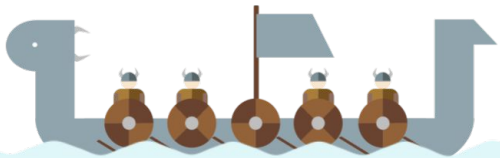







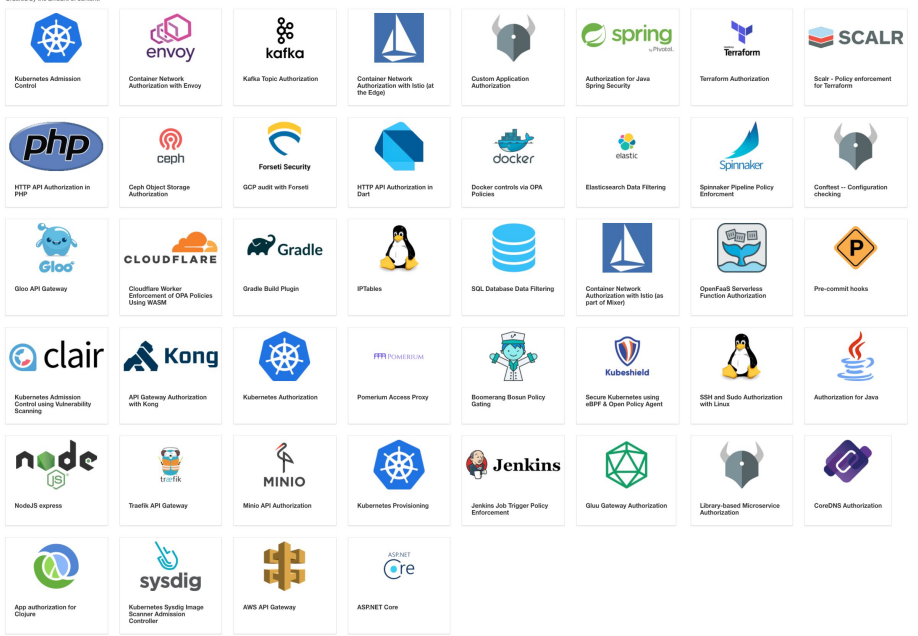


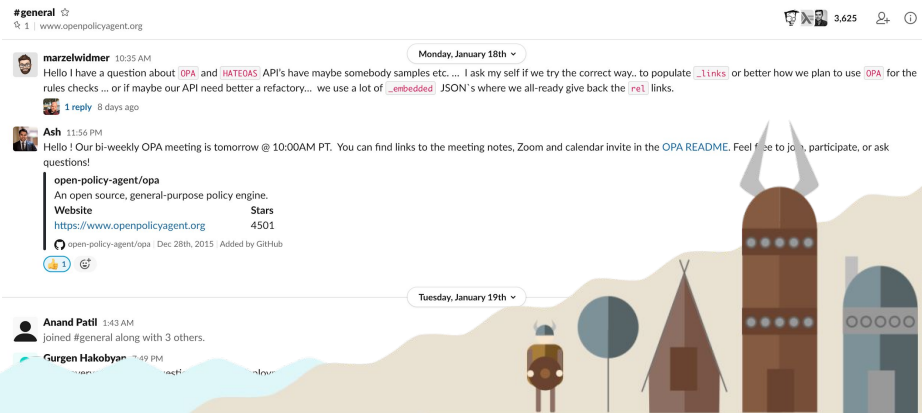

- Open source general purpose policy engine
- Unified toolset and framework for policy across the stack
- Decouples policy from application logic
- Separates policy *decision* from *enforcement*
- Policies written in declarative language Rego
- Popular use cases ranging from kubernetes admission control,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




Vibrant community

- 160 contributors
- 50+ integrations
- 5000+ Github Stars
- 4000+ Slack users
- 30+ million Docker image pulls
- Ecosystem including Conftest, Gatekeeper, VS Code and IntelliJ editor plugins.

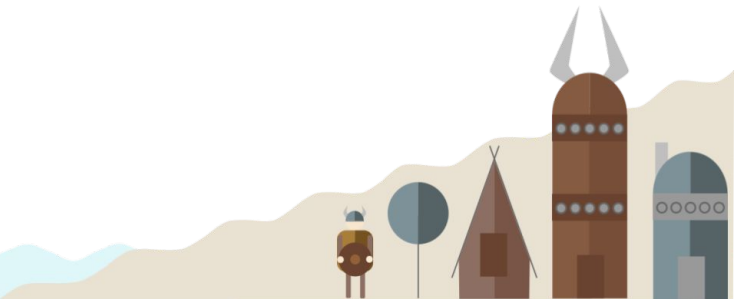




Production users

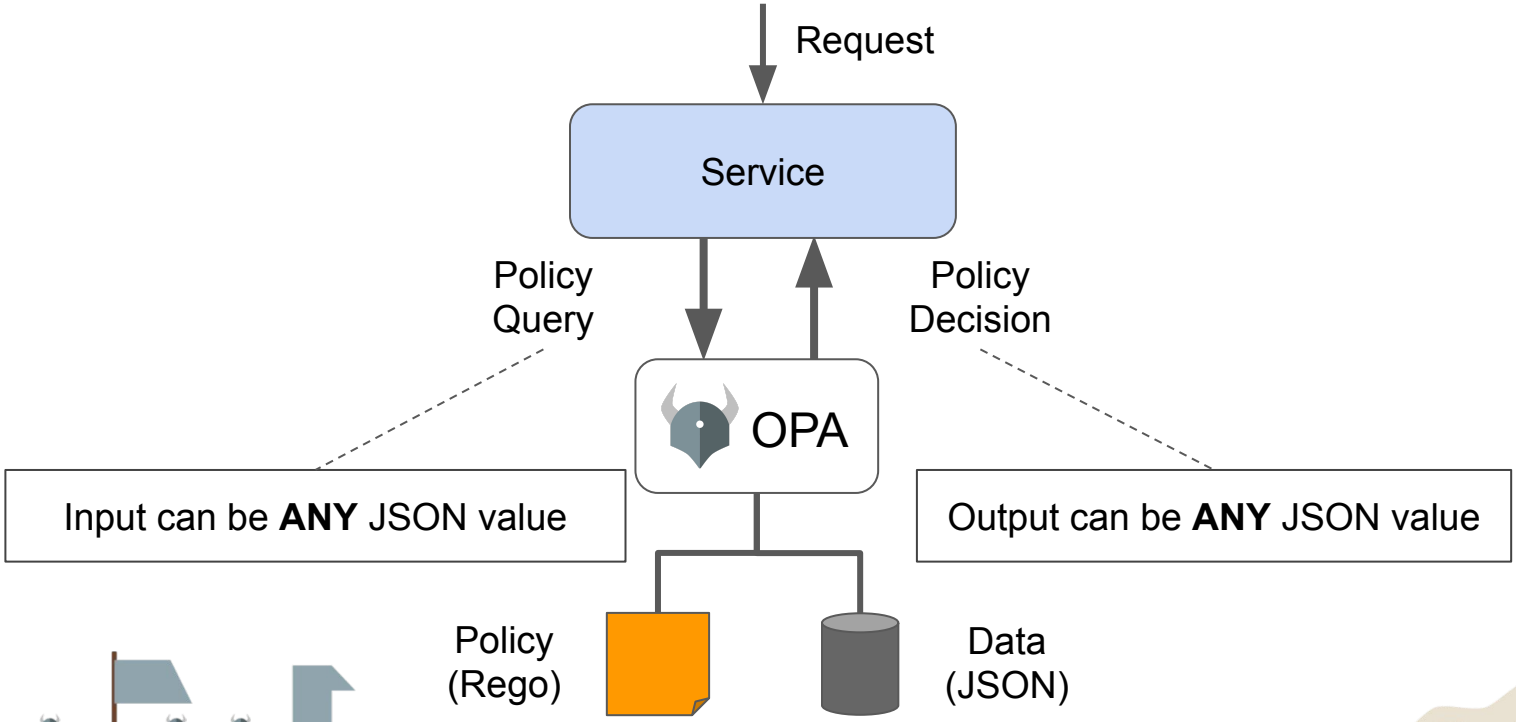
Kelsey Hightower
@kelseyhightower

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.

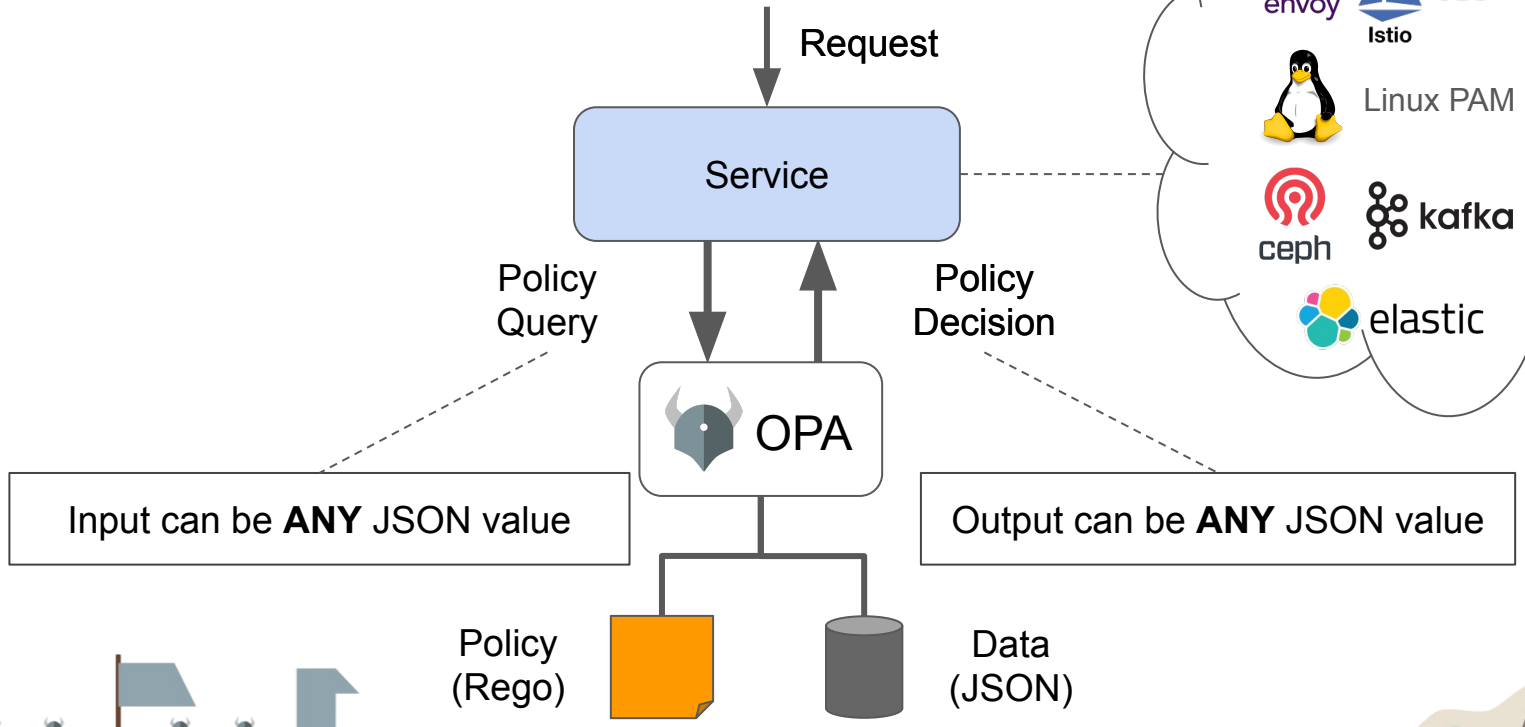


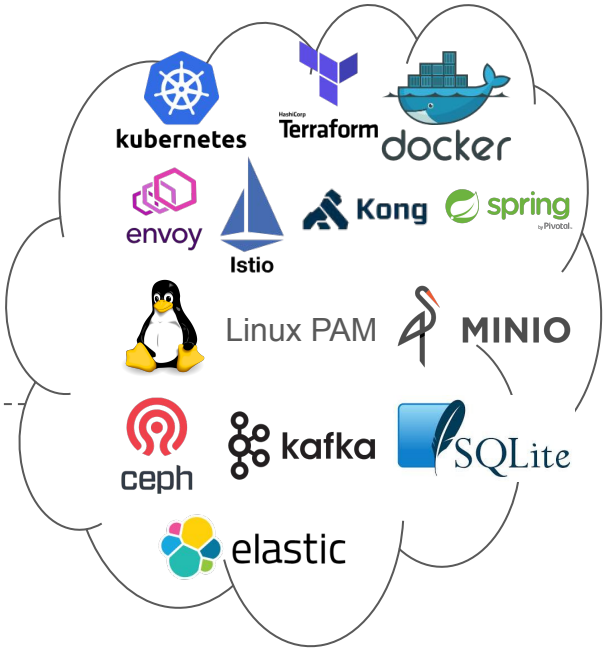
OPA and Rego

Policy decision model



Policy decision model





Deployment

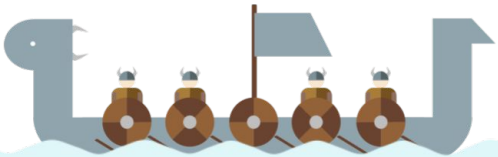
- OPA runs as a lightweight self-contained server binary.
- OPA ideally deployed as close to service as possible. This usually means on the same host, as a daemon or in a sidecar container.
- Applications communicate with the OPA server through its REST API.
- Go library available for Go applications.
- Envoy/Istio based applications. WASM.

Policy authoring and Rego

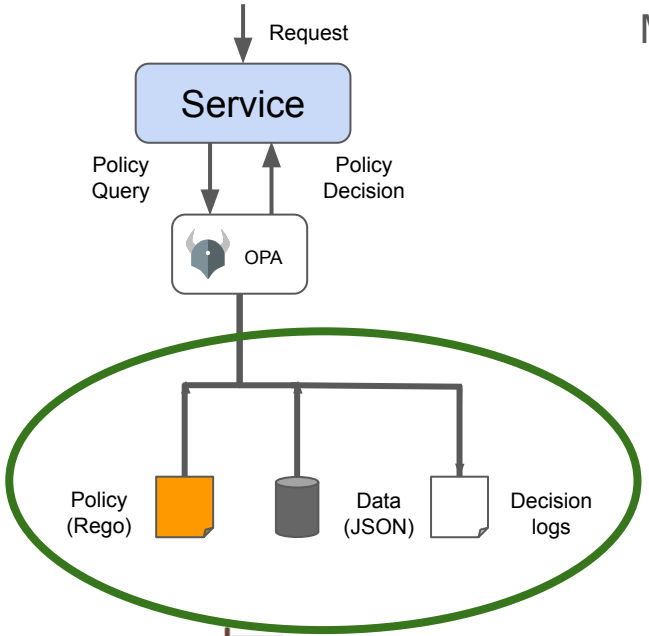
- Declarative high-level policy language used by OPA.
- Policy consists of any number of rules.
- Rules commonly return true/false but may return any type available in JSON, like strings, lists and objects.
- 150+ built-in functions: JWTs, date/time, CIDR math ,etc.
- Policy testing is easy with provided unit test framework.
- Well documented! https://www.openpolicyagent.org/docs/latest/
- Try it out! https://play.openpolicyagent.org/

Policy data

- JSON Web Tokens
- As part of query input
- Push data
- Bundle API
- http.send function from inside policy



Management APIs



Managing OPA at scale

- Bundle API - distribute policy and data from a central location
- Decision log API - allow OPA instances to report back on any decisions made. This may be used for auditing as well as for refinement of policies.
- Status API - allows OPA to send status and health updates to the management server.
- Discovery API - provides OPA instances the option to periodically fetch configuration.

Demo

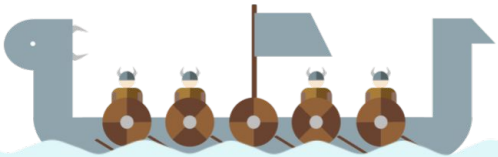
CI/CD

CI/CD deployment

- Running HTTP server not always ideal in this context (but could be!)
- “opa eval” command to do one-off evaluation of data against policy.
- Conftest
    - Runs policy-based tests on data like configuration and deployment resources.
    - Works with many non-JSON formats (XML, HCL, INI, Dockerfiles, and many more).

CI/CD use cases

- Validation
- Triggering
- Kubernetes admission control



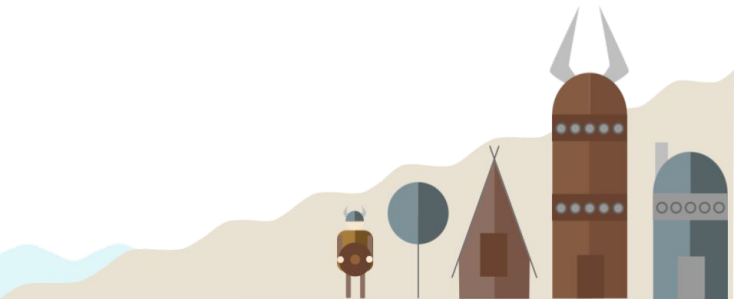
Validation

Most common use case:

- **Deployment policies**
  - Do not deploy during certain windows (e.g. black friday, no one on-call)
  - Require sign-off for production deployments

- **Shift-left**
  - Developers want feedback as early as possible.
  - Run OPA inside of pipelines to validate K8s manifests early (locally, pre-merge hook)
  - Terraform and other IaC solutions (CloudFormation, ARM templates, etc.)

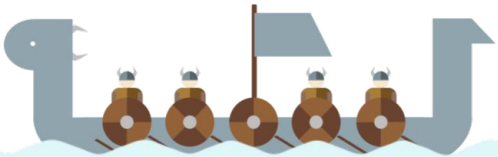
- **Configuration file testing**
  - Using Conftest to test Dockerfiles, npm files, etc.



Triggering

Creating policy based workflows where OPA decides what builds/deploys

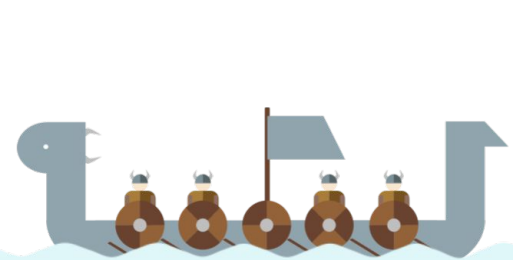
- Load context from AWS EC2, GitHub, deployment environments, etc.
- Policies analyze context and decide what jobs need to run
  - Images to build (e.g., every commit)
  - Tests to execute
  - Images to garbage collect from ECR
- **Output from policy is a workflow spec**



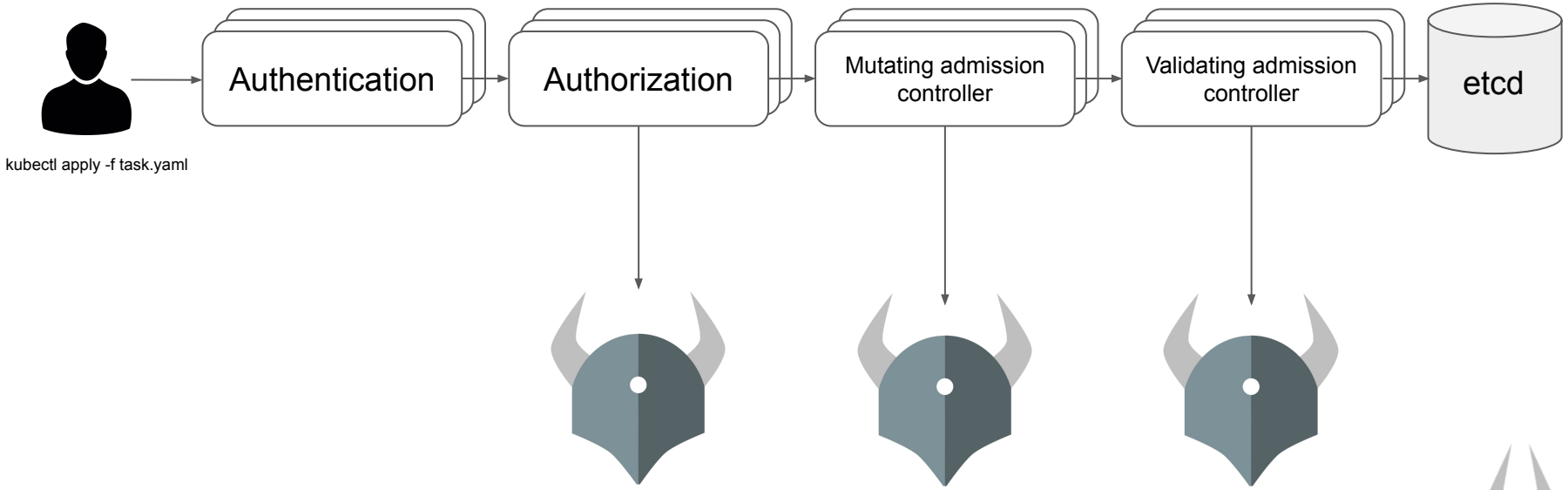
Kubernetes Admission Controller

Policy guardrails around Kubernetes clusters

- One of the most common use cases for OPA.
- Allows policy to validate (and deny) resources as they are created/updated.
- Mutating admission controller allows mutating resource.
- CI/CD context: works just as well for CRDs such as those used by e.g. Tekton, allowing rules around which tasks or pipelines should be allowed.



The Kubernetes API



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 - consider management, logging, bundle server, etc.
- Styra Academy - https://academy.styra.com
- Styra DAS - http://www.styra.com/das-free
- Join the OPA Slack community! - https://openpolicyagent.slack.com

Thank you!