Cloud Native Meetup Linz

25.02.2025

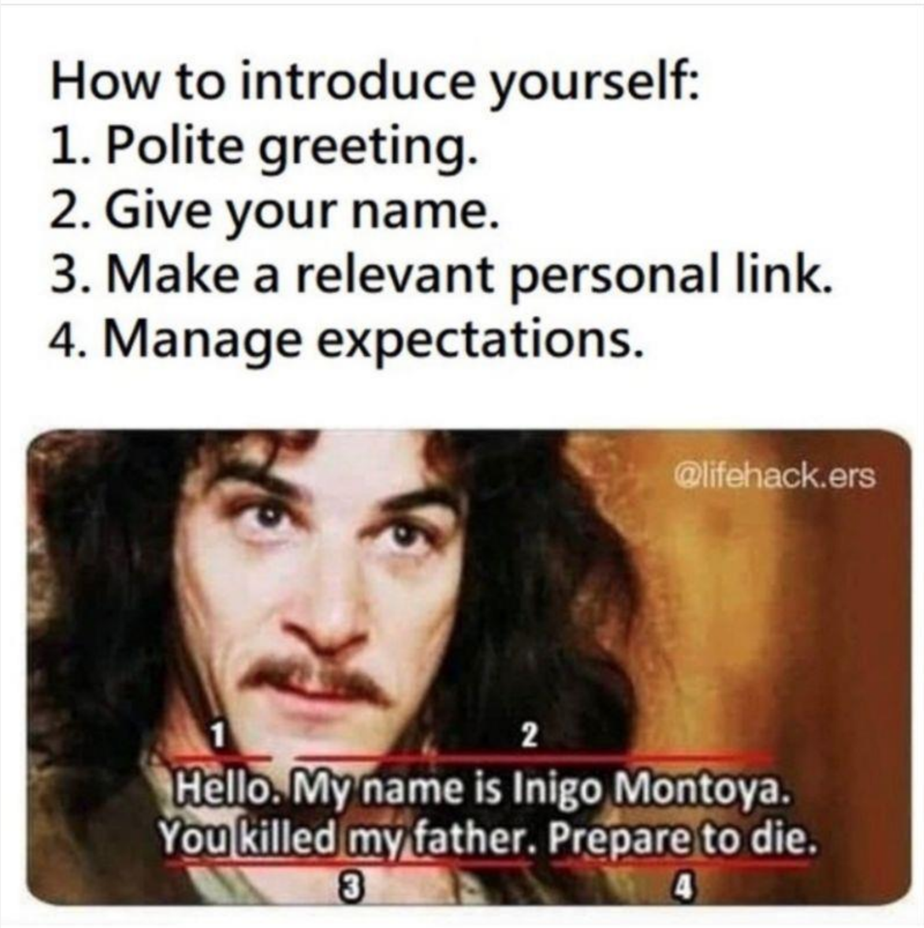
POSEDIO

Designing Zero Trust Systems

Damjan Gjurovski,
CTO of Posedio

Hello

- Head of Technology of Posedio
- Work on Software/Data/Platform Engineering
- Largest online transaction processing engine in AT
- Largest GCP developer platform in AT
- Enjoys building secure systems
- How can we build secure systems?



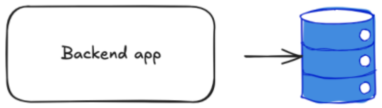
How to introduce yourself:
1. Polite greeting.
2. Give your name.
3. Make a relevant personal link.
4. Manage expectations.

Hello. My name is Inigo Montoya.
You killed my father. Prepare to die.

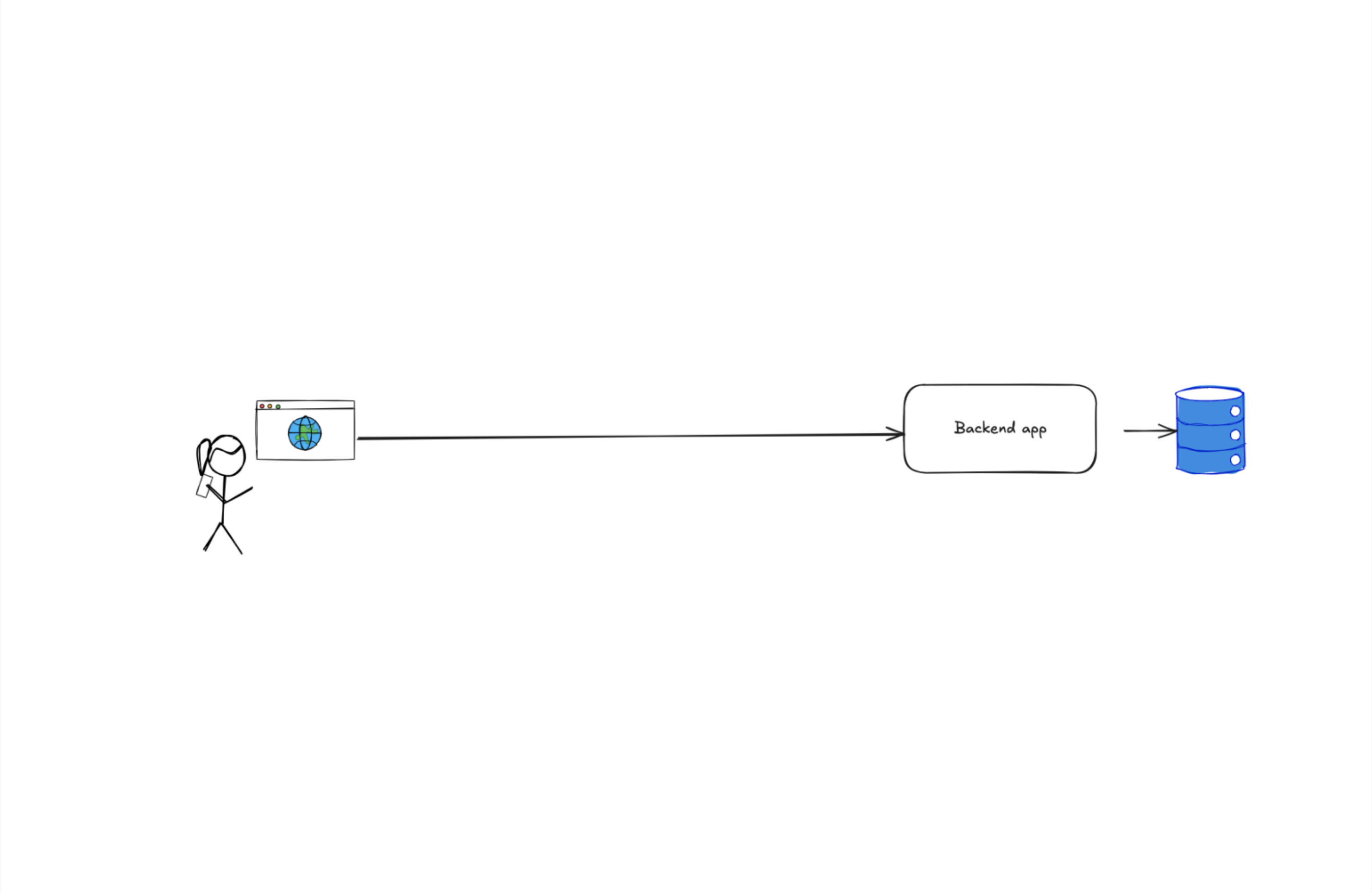
Security, the old way

01

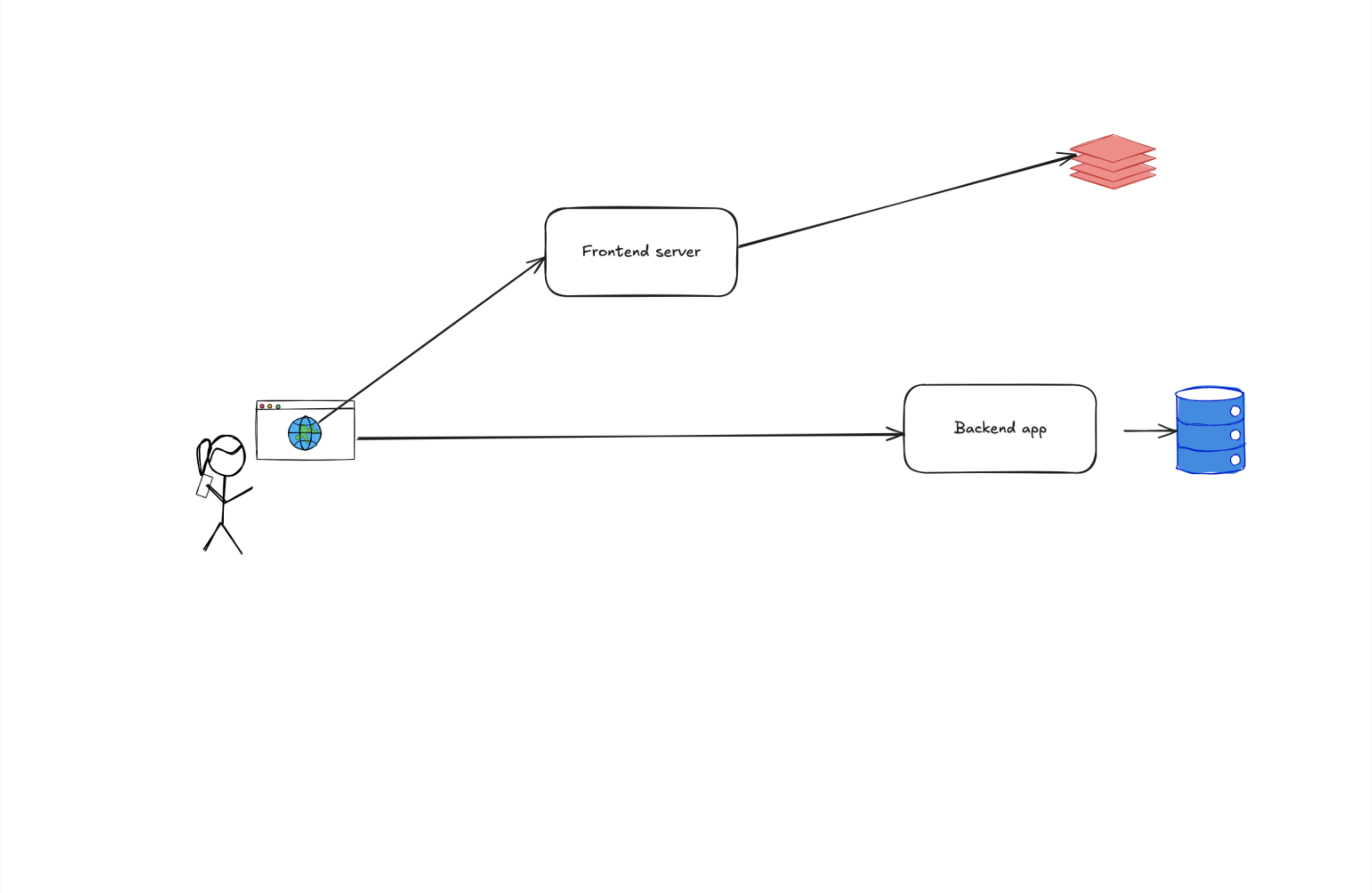
The good old days



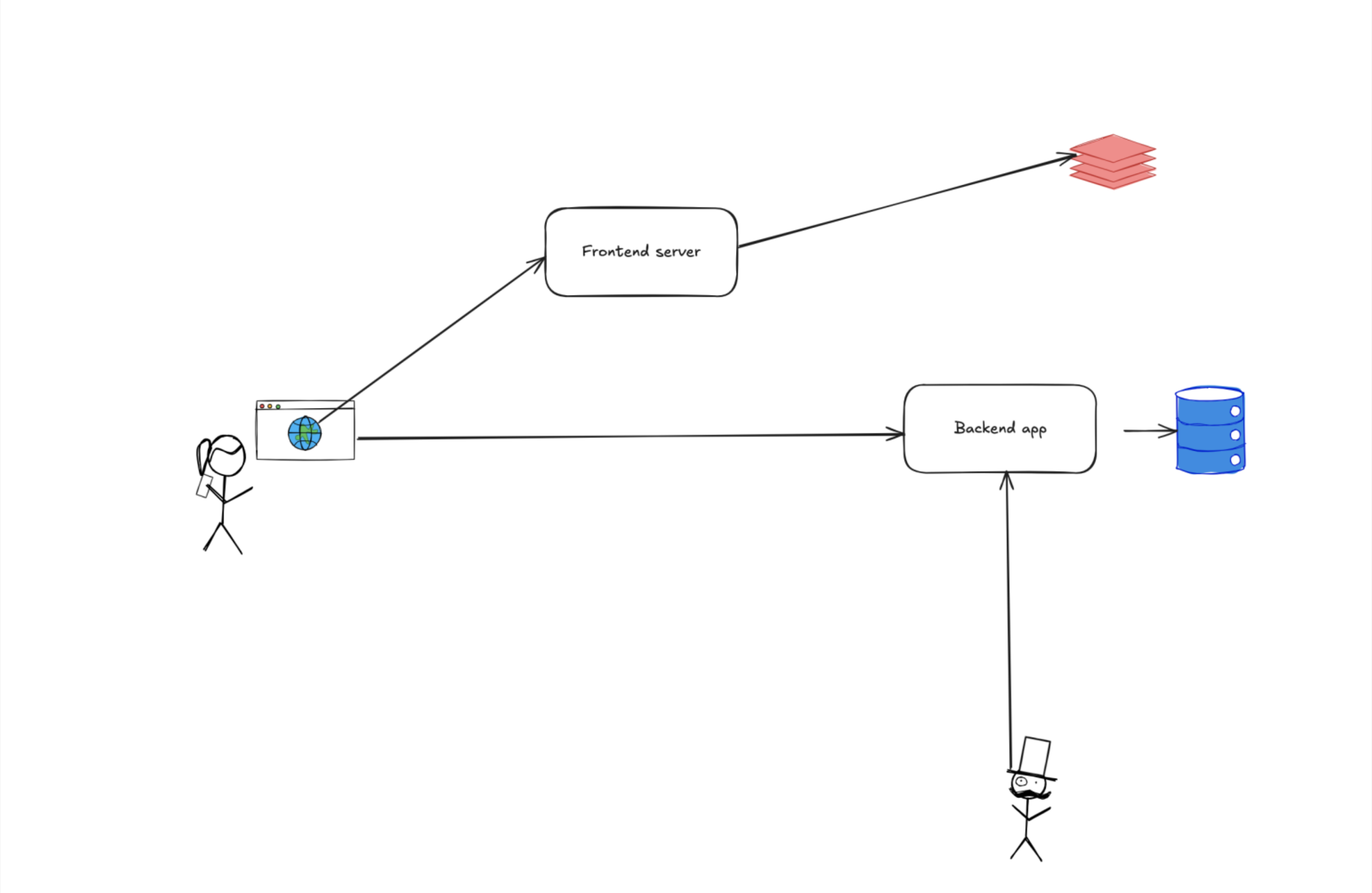
Becoming useful



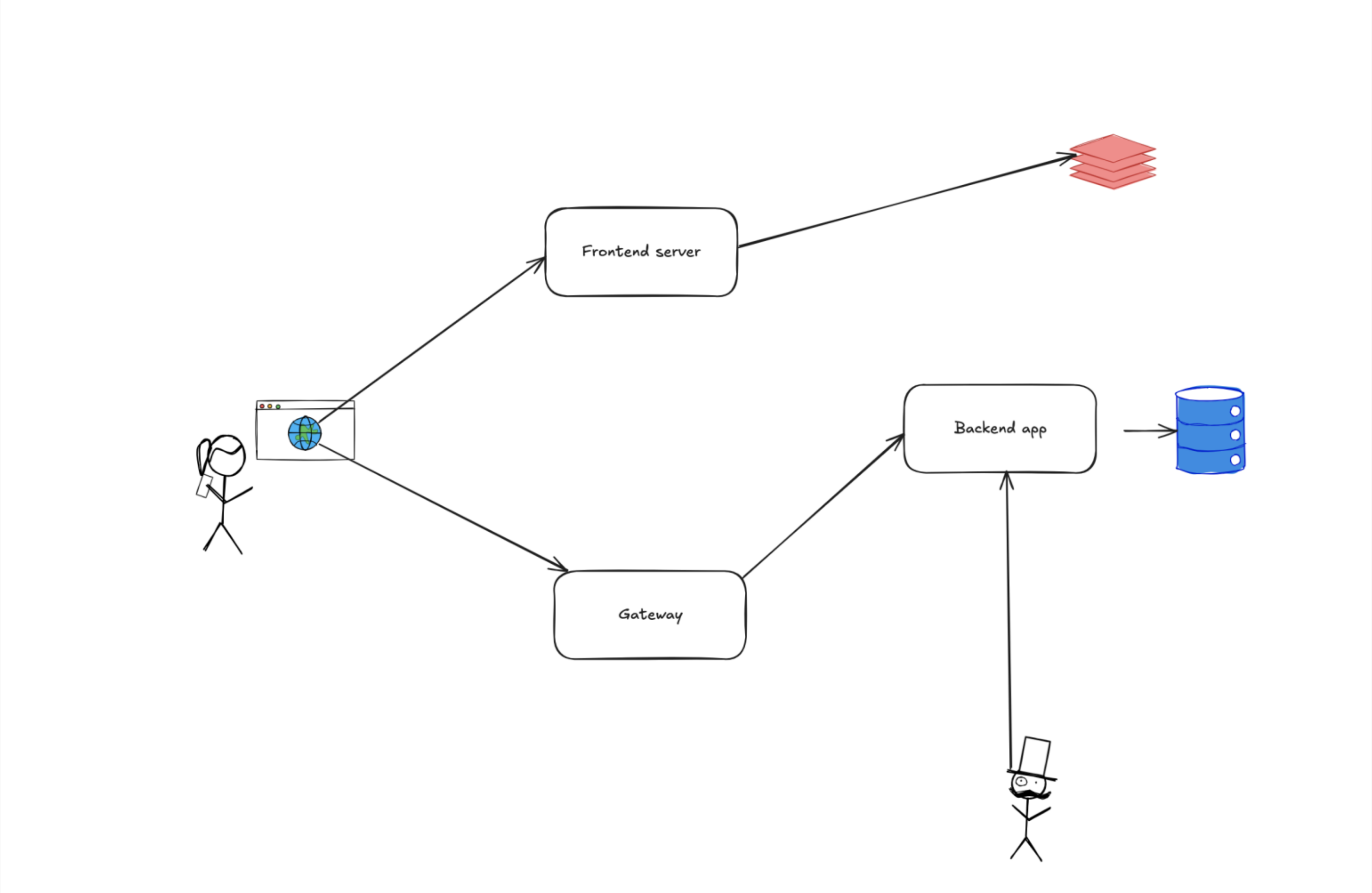
What about a nice frontend?



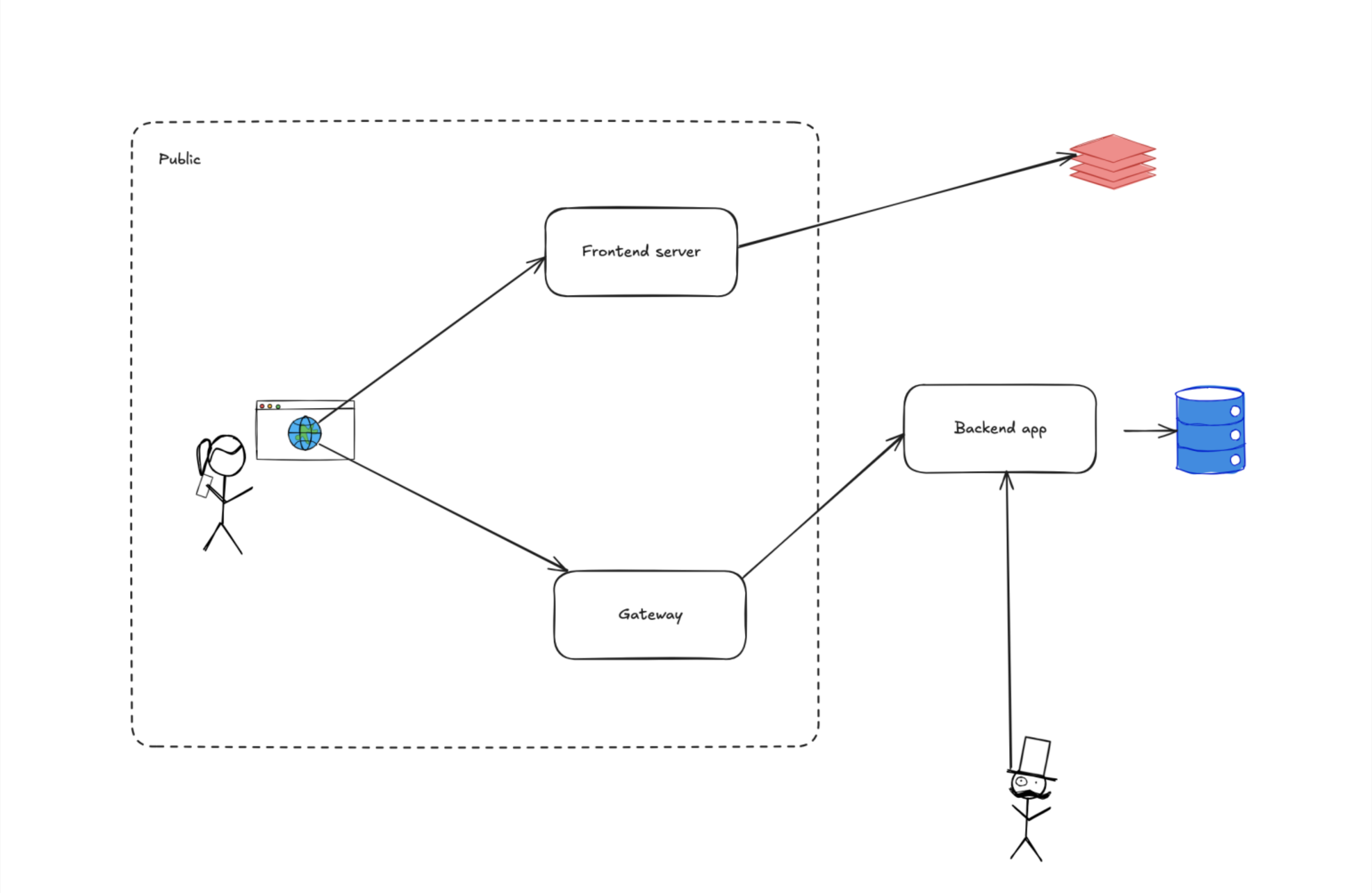
Admin access needed



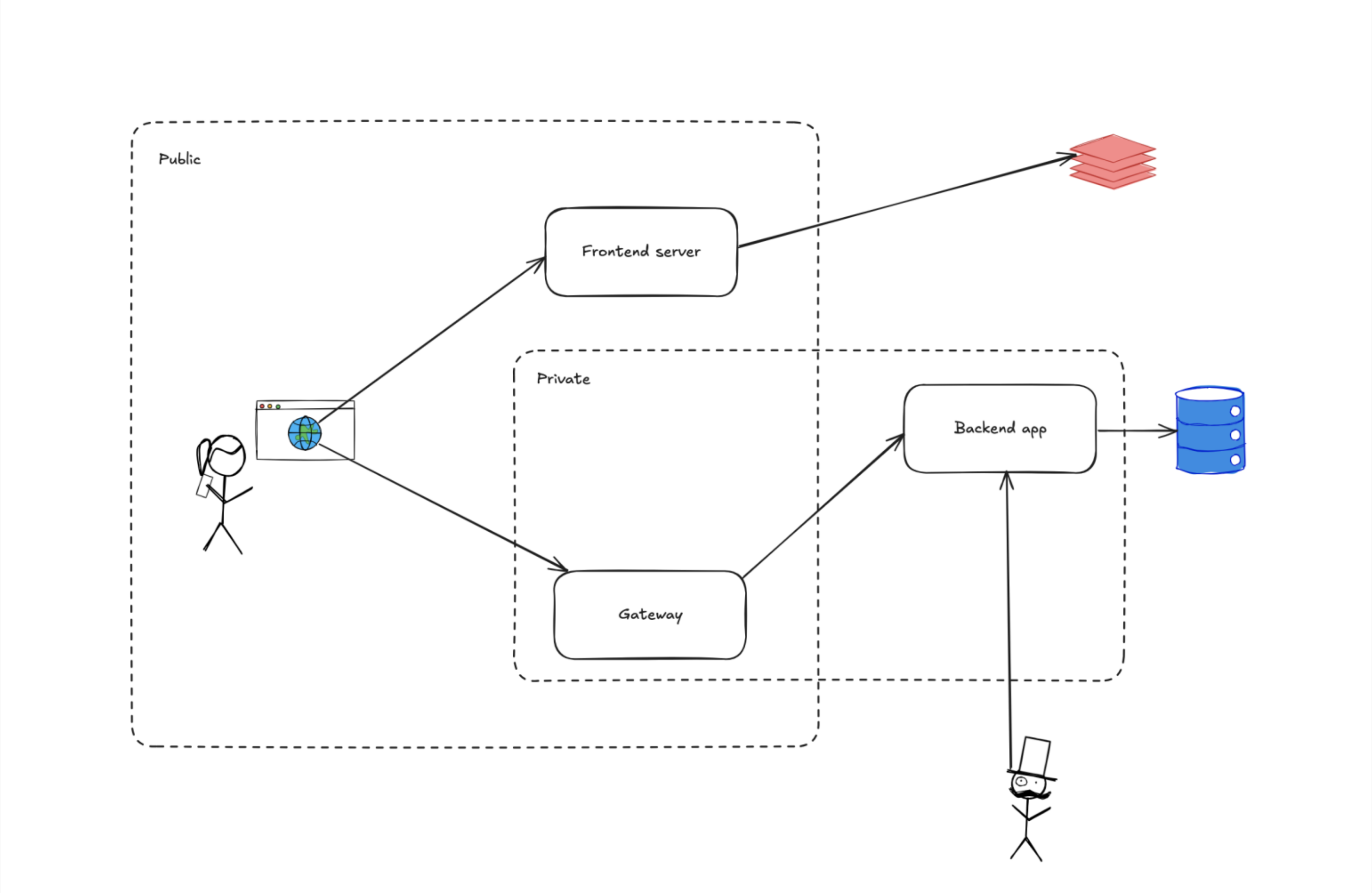
Load balancing to the rescue



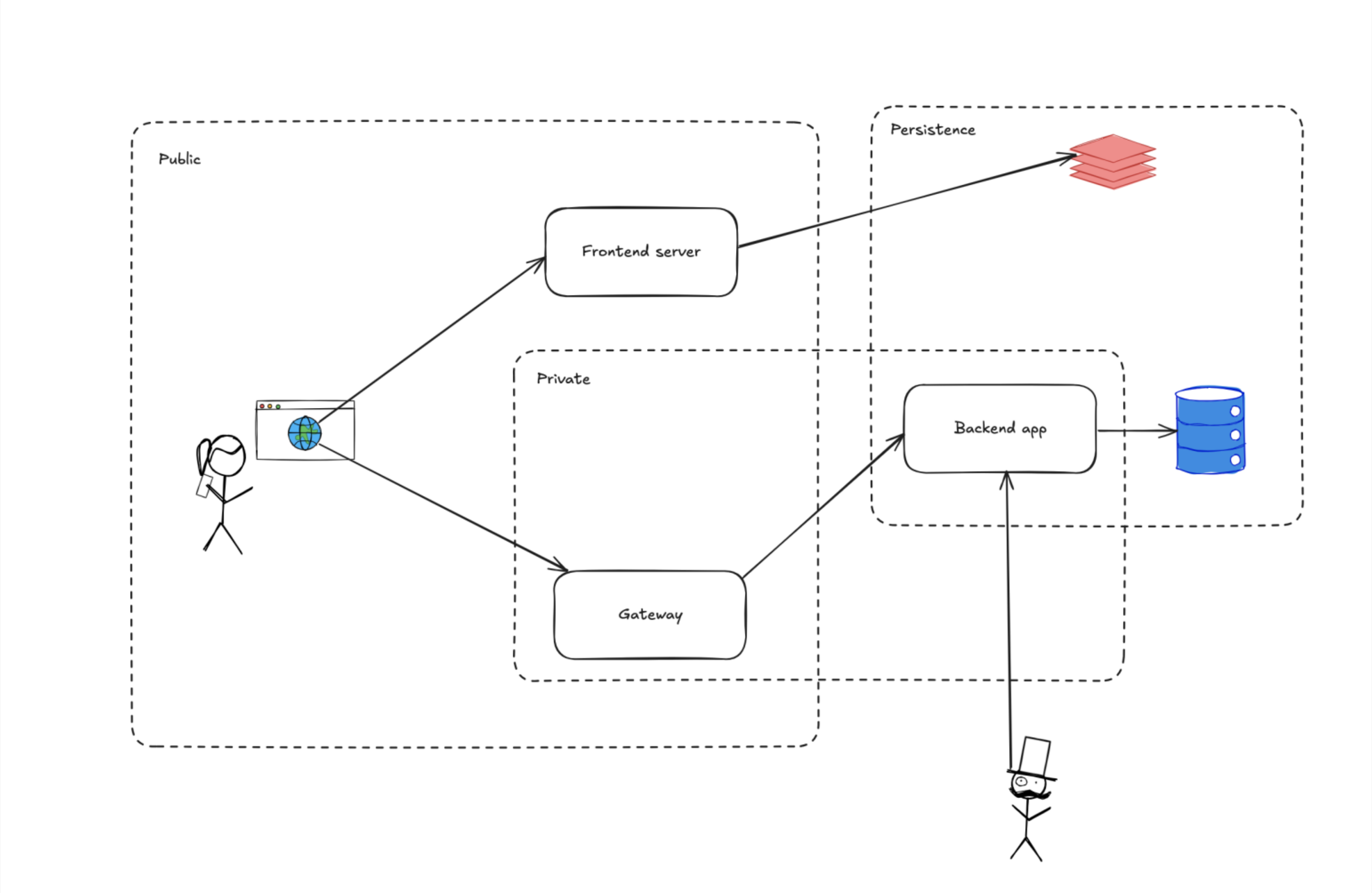
Who can access our services



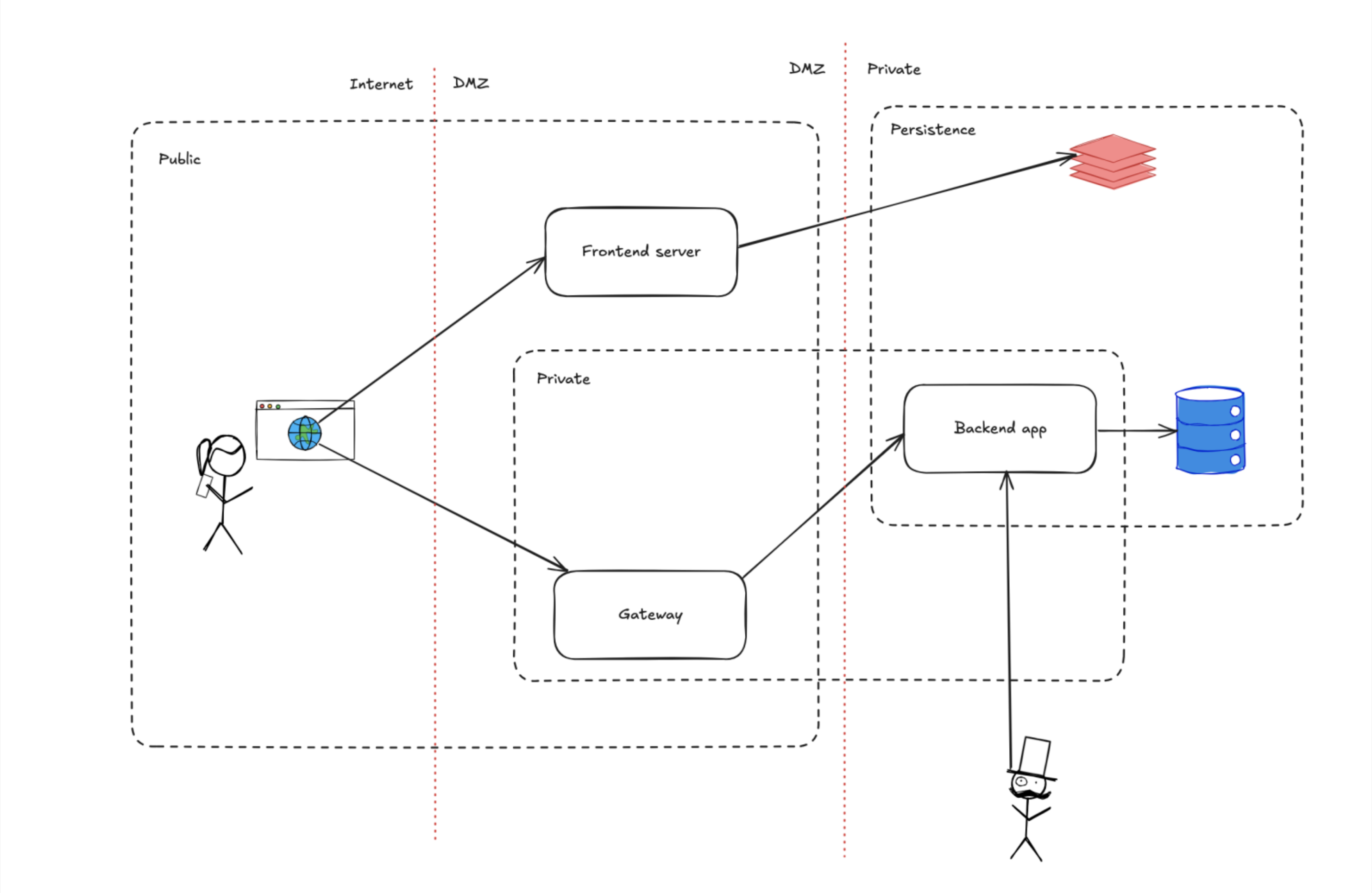
Let’s keep things private



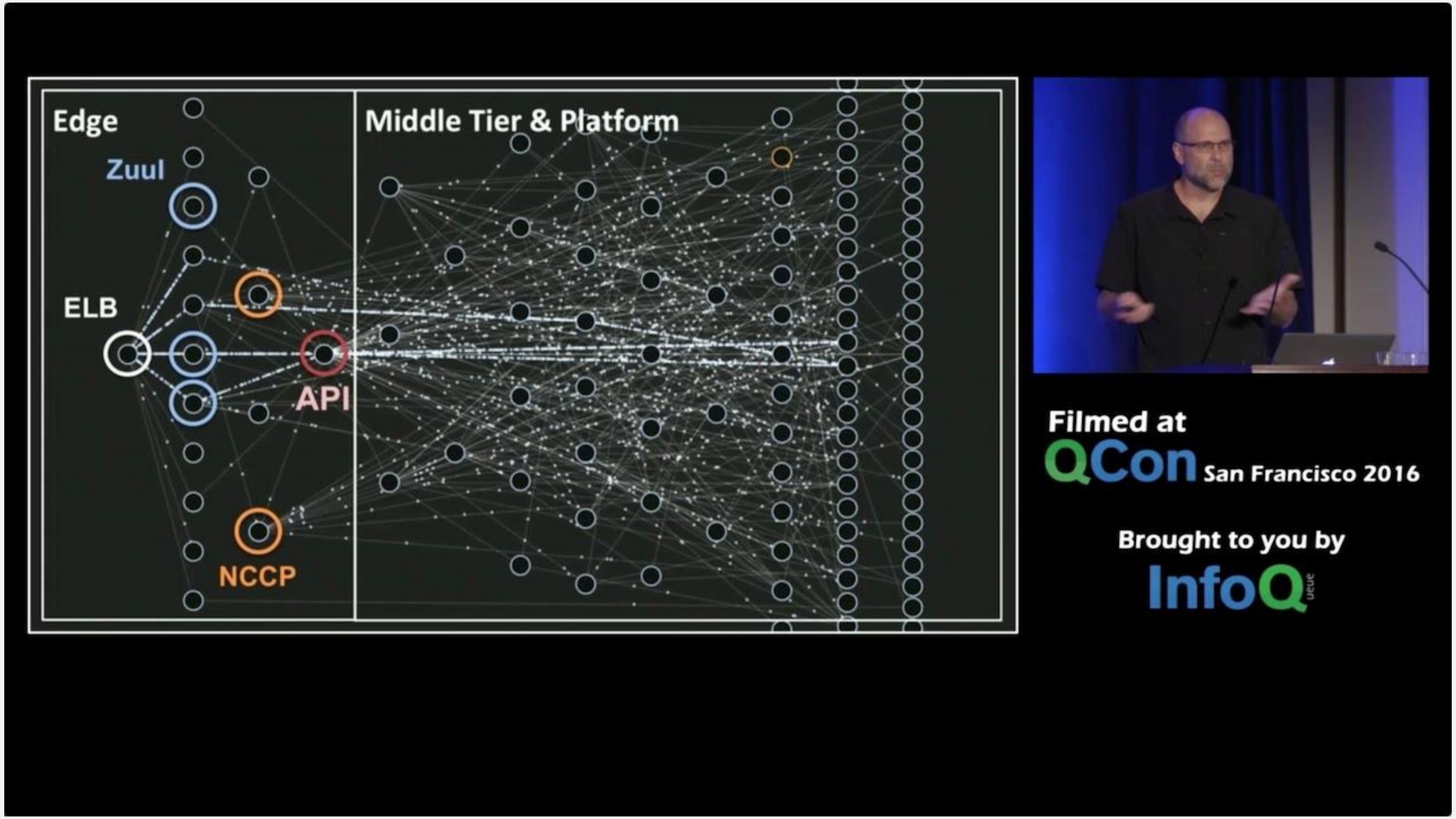
The crown jewels



Compartmentalisation is the solution



Or is it?


Edge
Zuul
ELB
API
NCCP
Middle Tier & Platform
Filmed at
QCon San Francisco 2016
Brought to you by
InfoQ

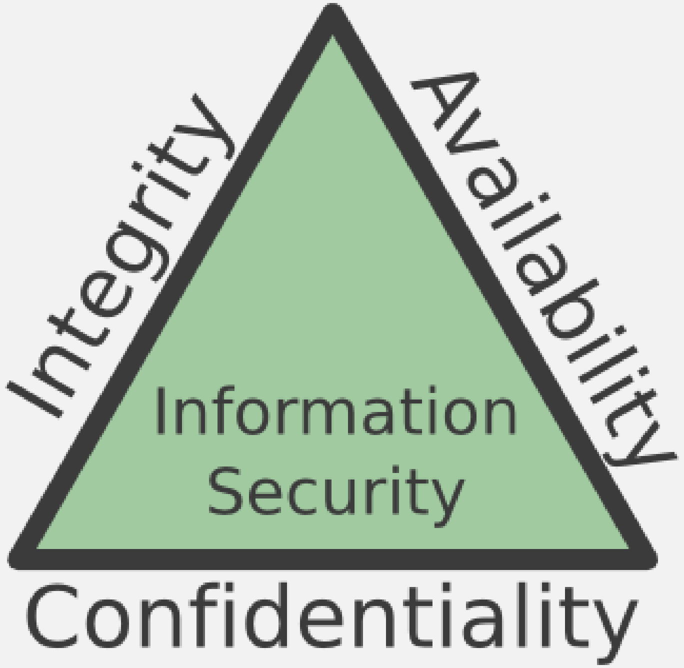
Do it **RIGHT.**

What is security?

02

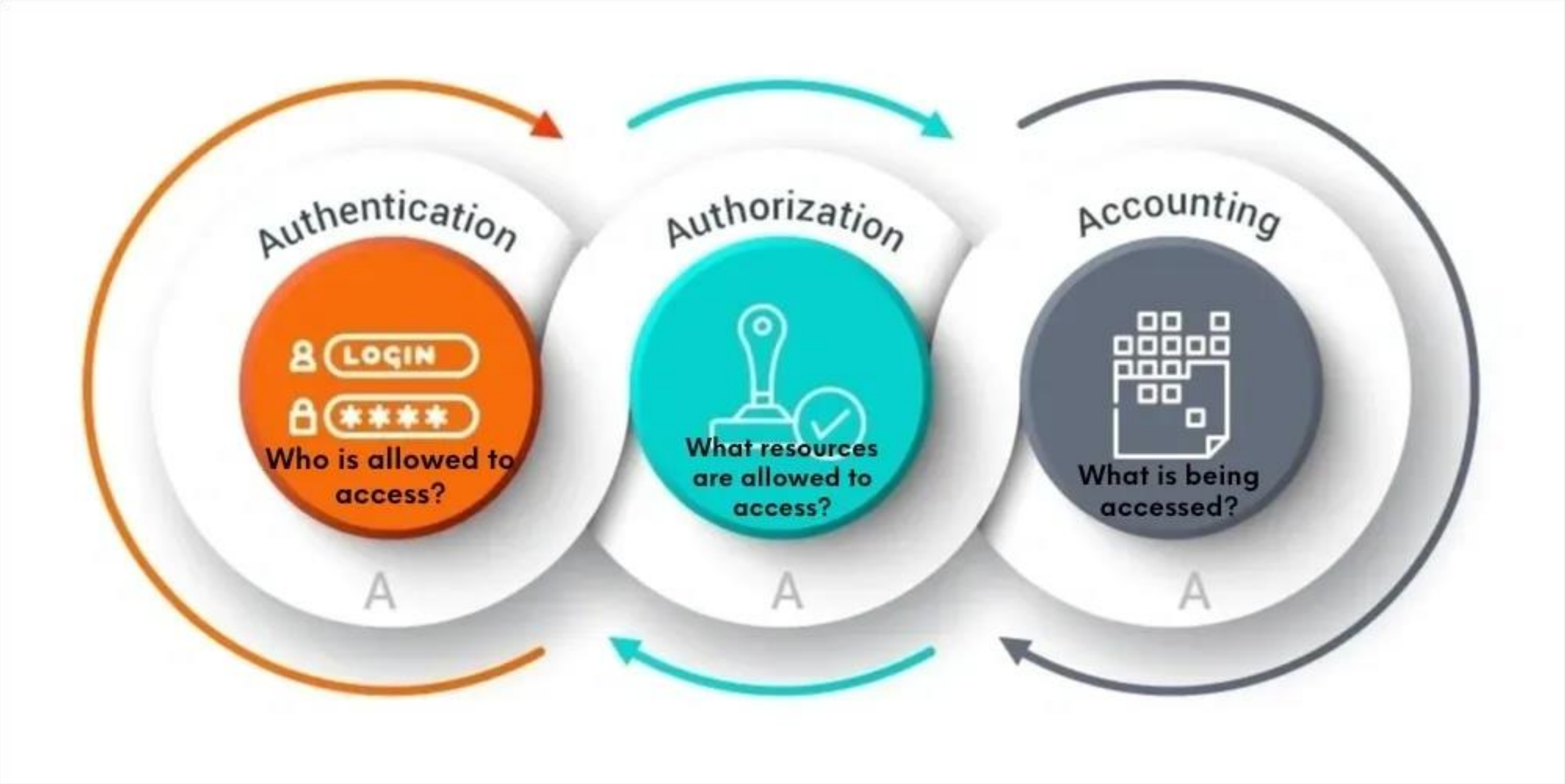
The glossary

CIA triad



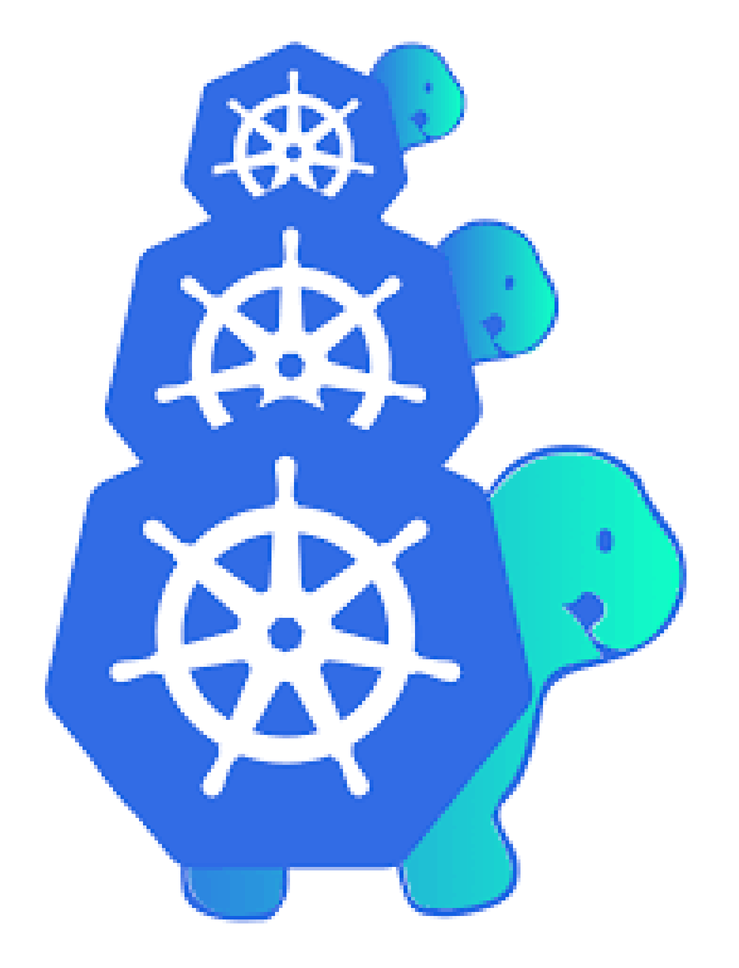
The glossary

Triple A



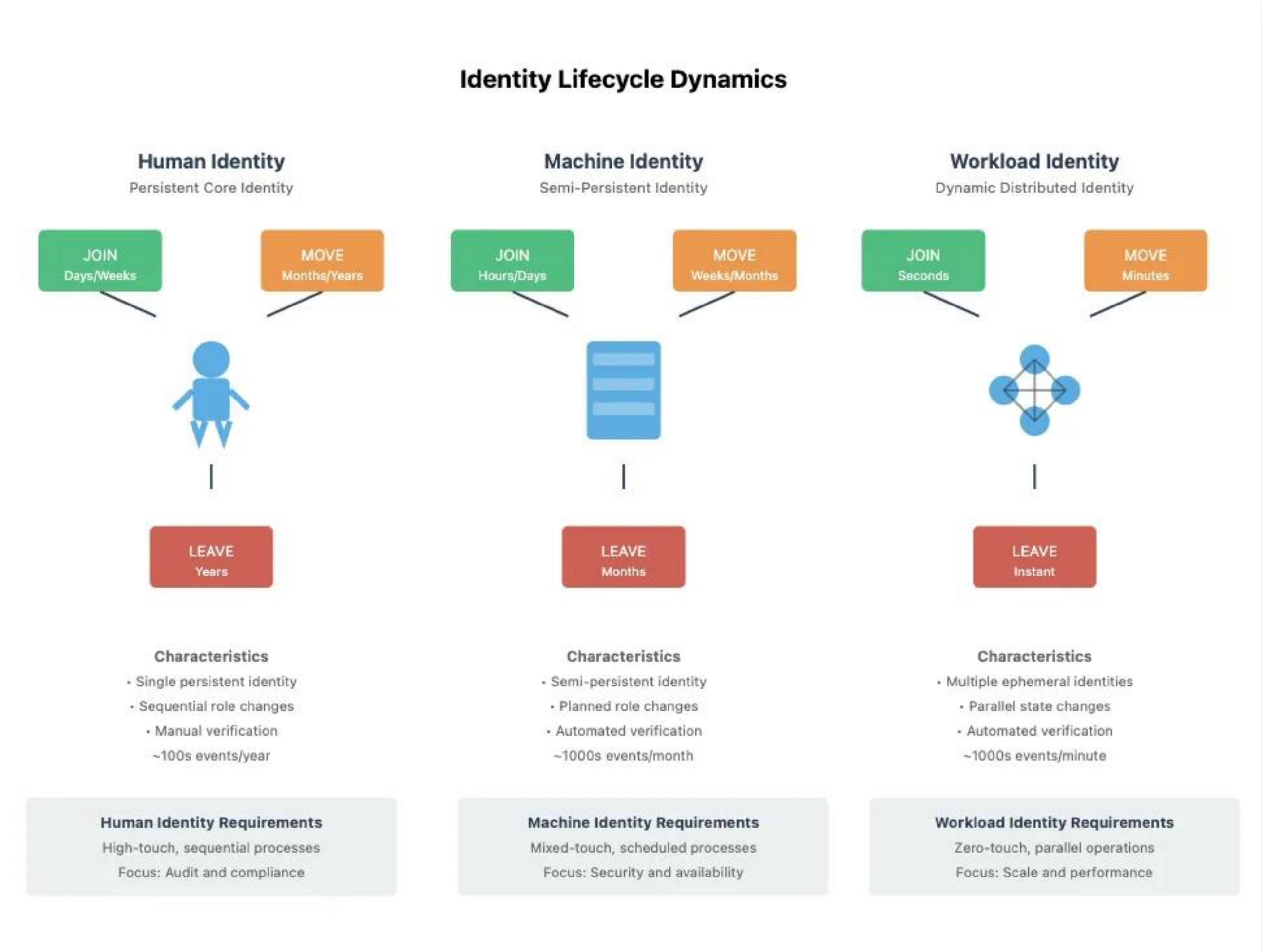
The glossary

Root of trust



The glossary

Identity



Identity Lifecycle Dynamics

Human Identity
Persistent Core Identity

Characteristics

- Single persistent identity
- Sequential role changes
- Manual verification

~100s events/year

Human Identity Requirements
High-touch, sequential processes
Focus: Audit and compliance

Machine Identity
Semi-Persistent Identity

Characteristics

- Semi-persistent identity
- Planned role changes
- Automated verification

~1000s events/month

Machine Identity Requirements
Mixed-touch, scheduled processes
Focus: Security and availability

Workload Identity
Dynamic Distributed Identity

Characteristics

- Multiple ephemeral identities
- Parallel state changes
- Automated verification

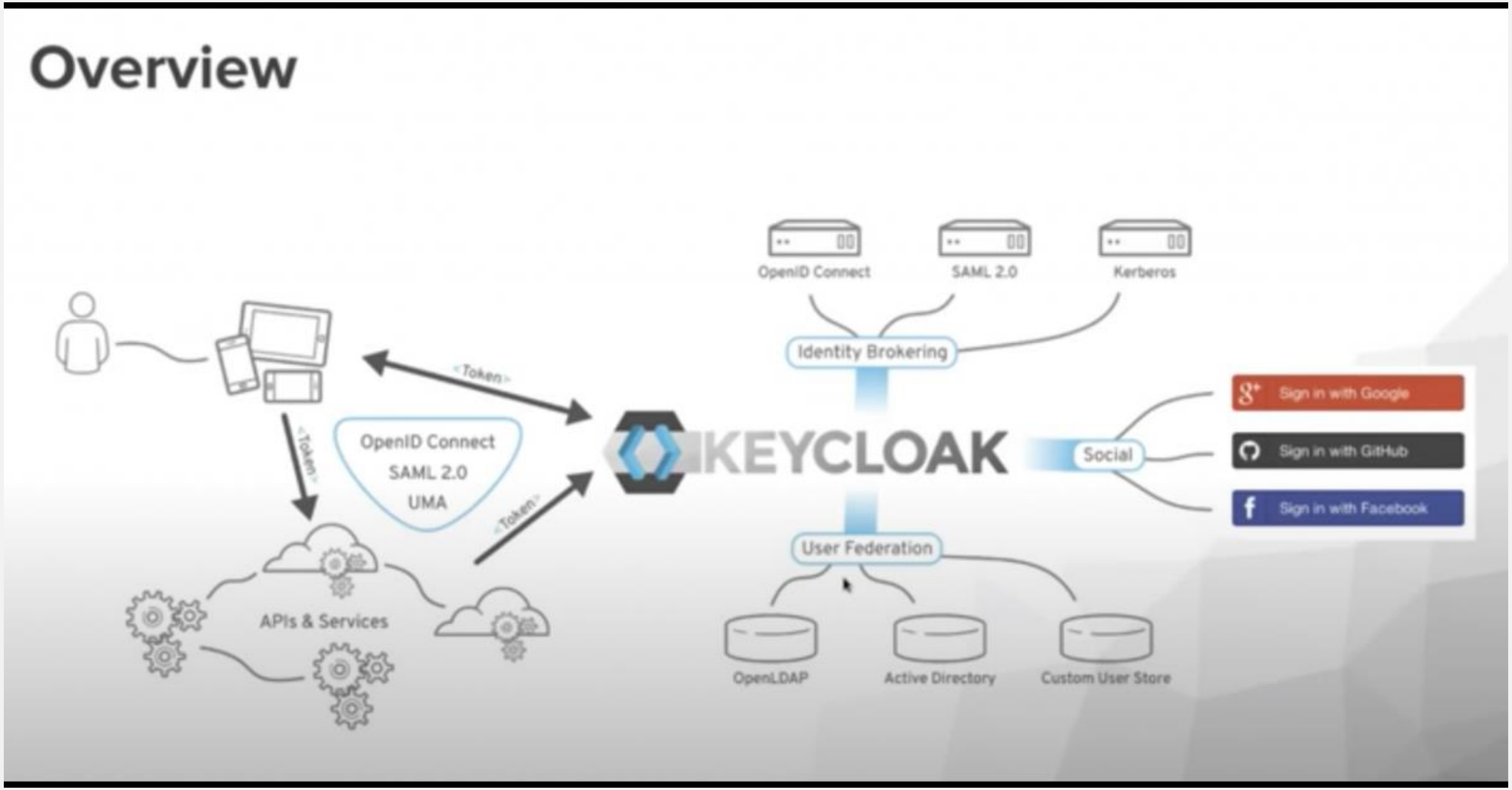
~1000s events/minute

Workload Identity Requirements
Zero-touch, parallel operations
Focus: Scale and performance

How can we secure our systems

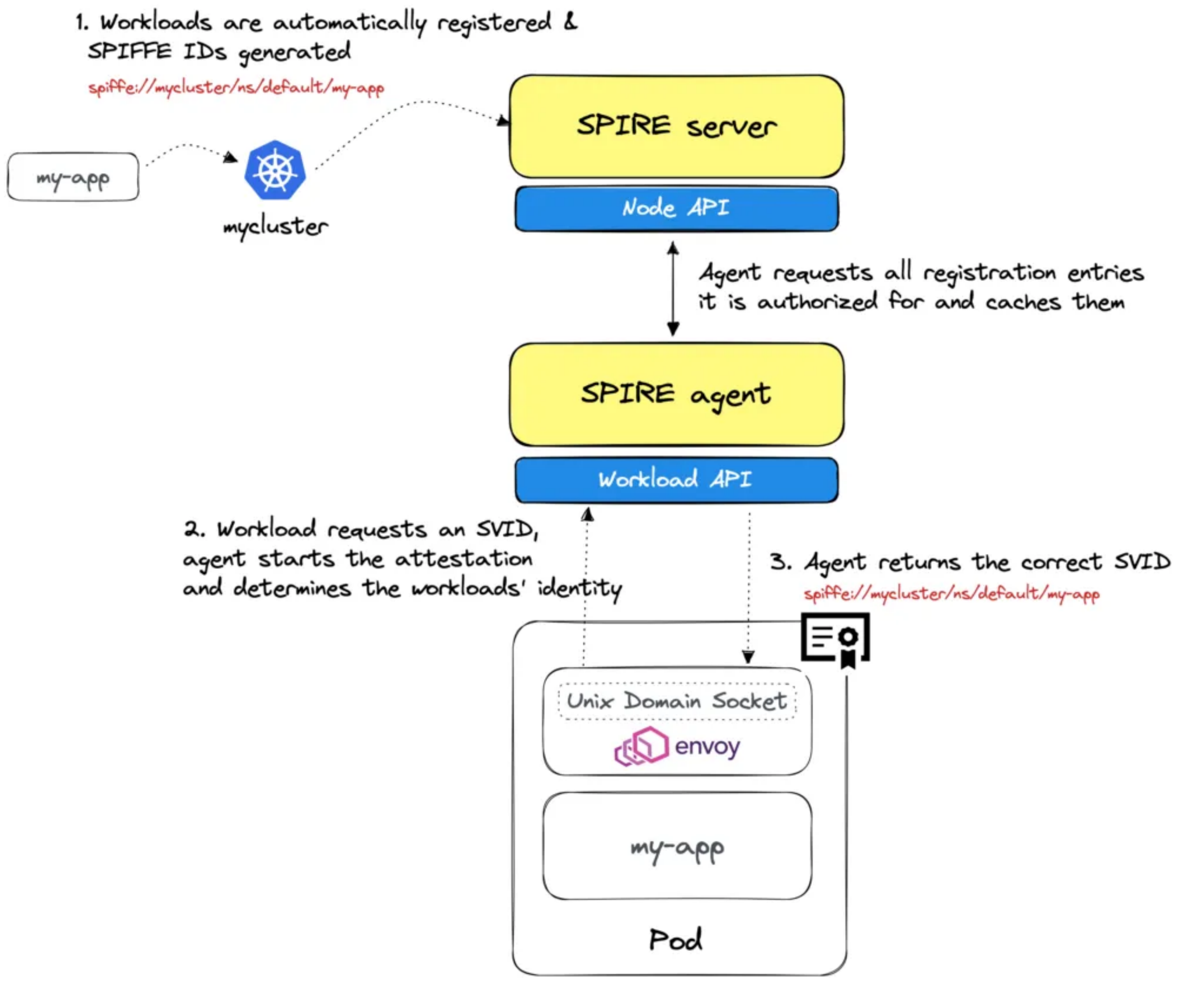
03

IdP - Keycloak

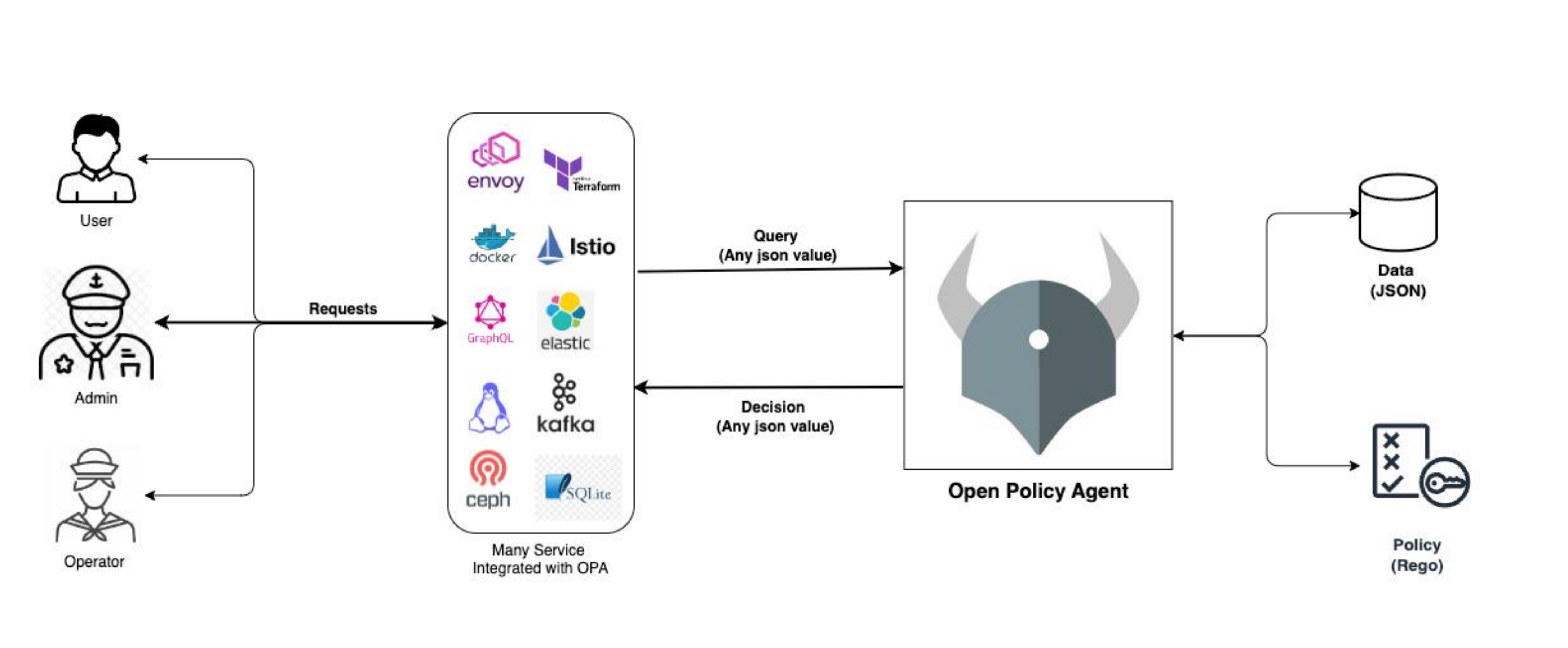


Overview

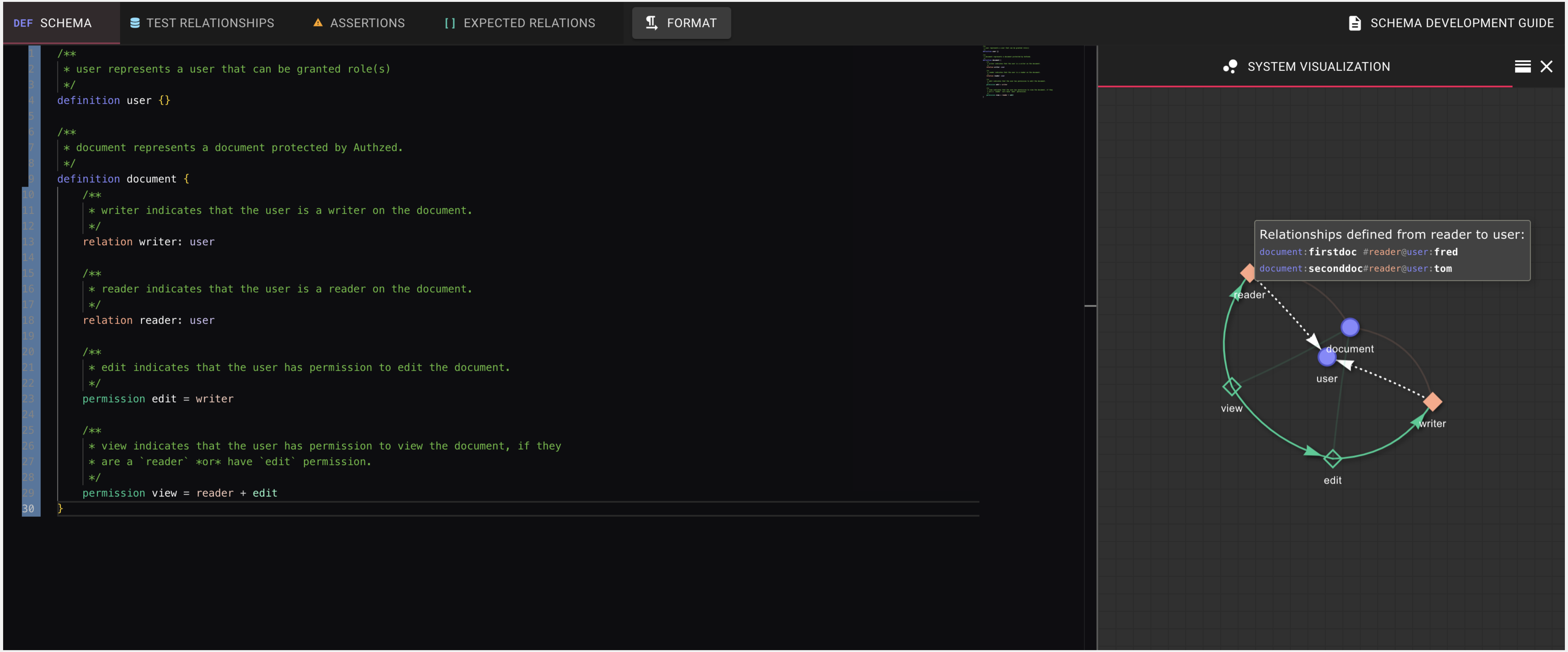
Workload Identity – SPIFFIE/SPIRE


1. Workloads are automatically registered & SPIFFE IDs generated
spiffe://mycluster/ns/default/my-app
my-app
mycluster
SPIRE server
Node API
Agent requests all registration entries it is authorized for and caches them
SPIRE agent
Workload API
2. Workload requests an SVID, agent starts the attestation and determines the workloads' identity
3. Agent returns the correct SVID
spiffe://mycluster/ns/default/my-app
Unix Domain Socket
envoy
my-app
Pod

Policy - OPA


User
Admin
Operator
Requests
envoy
Terraform
docker
Istio
GraphQL
elastic
kafka
ceph
SQLite
Many Service
Integrated with OPA
Query
(Any json value)
Decision
(Any json value)
Open Policy Agent
Data
(JSON)
Policy
(Rego)

Permissions - SpiceDB

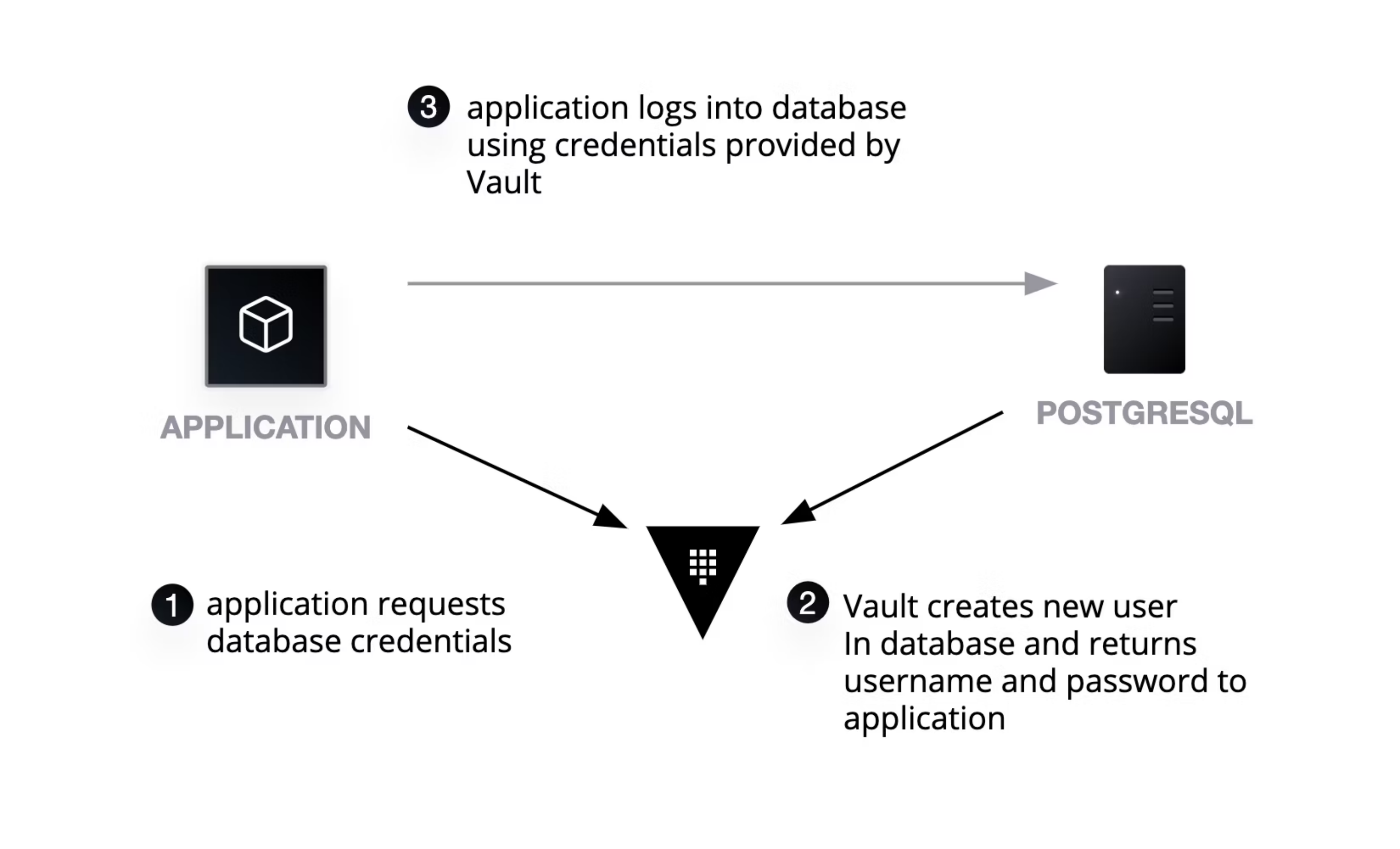


```
/**
 * user represents a user that can be granted role(s)
 */
definition user {}

/**
 * document represents a document protected by Authzed.
 */
definition document {
    /**
     * writer indicates that the user is a writer on the document.
     */
    relation writer: user

    /**
     * reader indicates that the user is a reader on the document.
     */
    relation reader: user

    /**
     * edit indicates that the user has permission to edit the document.
     */
    permission edit = writer

    /**
     * view indicates that the user has permission to view the document, if they
     * are a `reader` *or* have `edit` permission.
     */
    permission view = reader + edit
}
```

Relationships defined from reader to user:

```
document:firstdoc #reader@user:fred
document:seconddoc#reader@user:tom
```

Secrets - Vault



3. application logs into database using credentials provided by Vault

1. application requests database credentials

2. Vault creates new user In database and returns username and password to application

mTLS - ISTIO

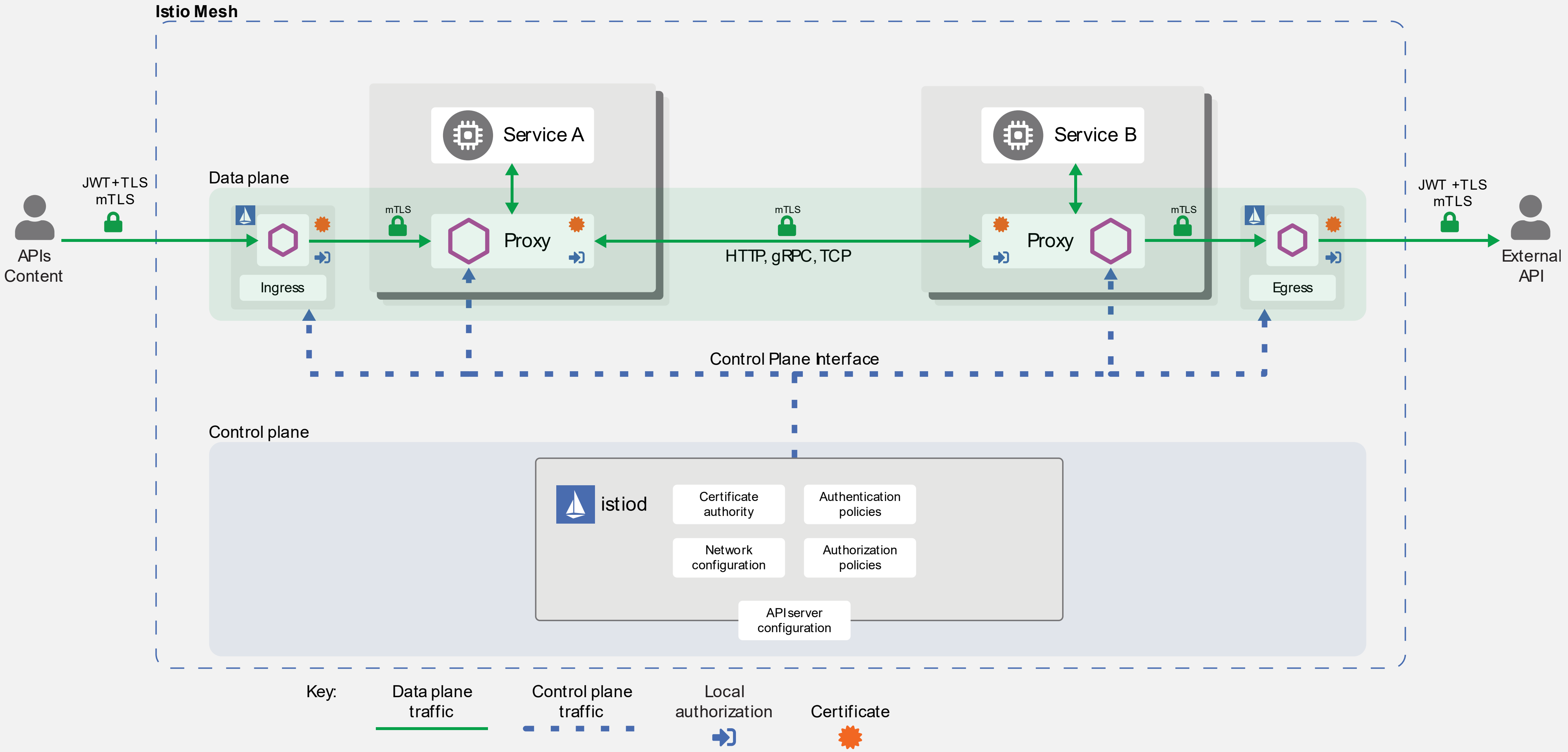
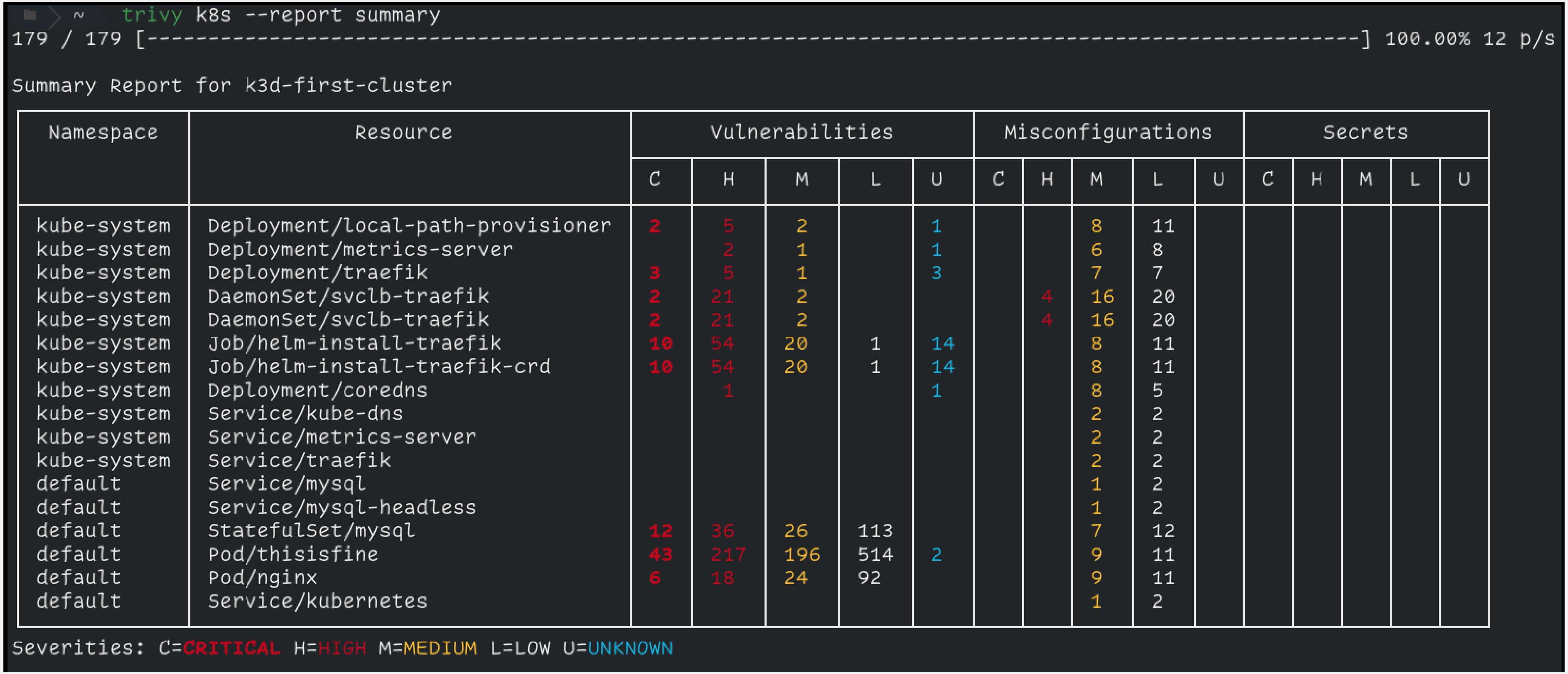
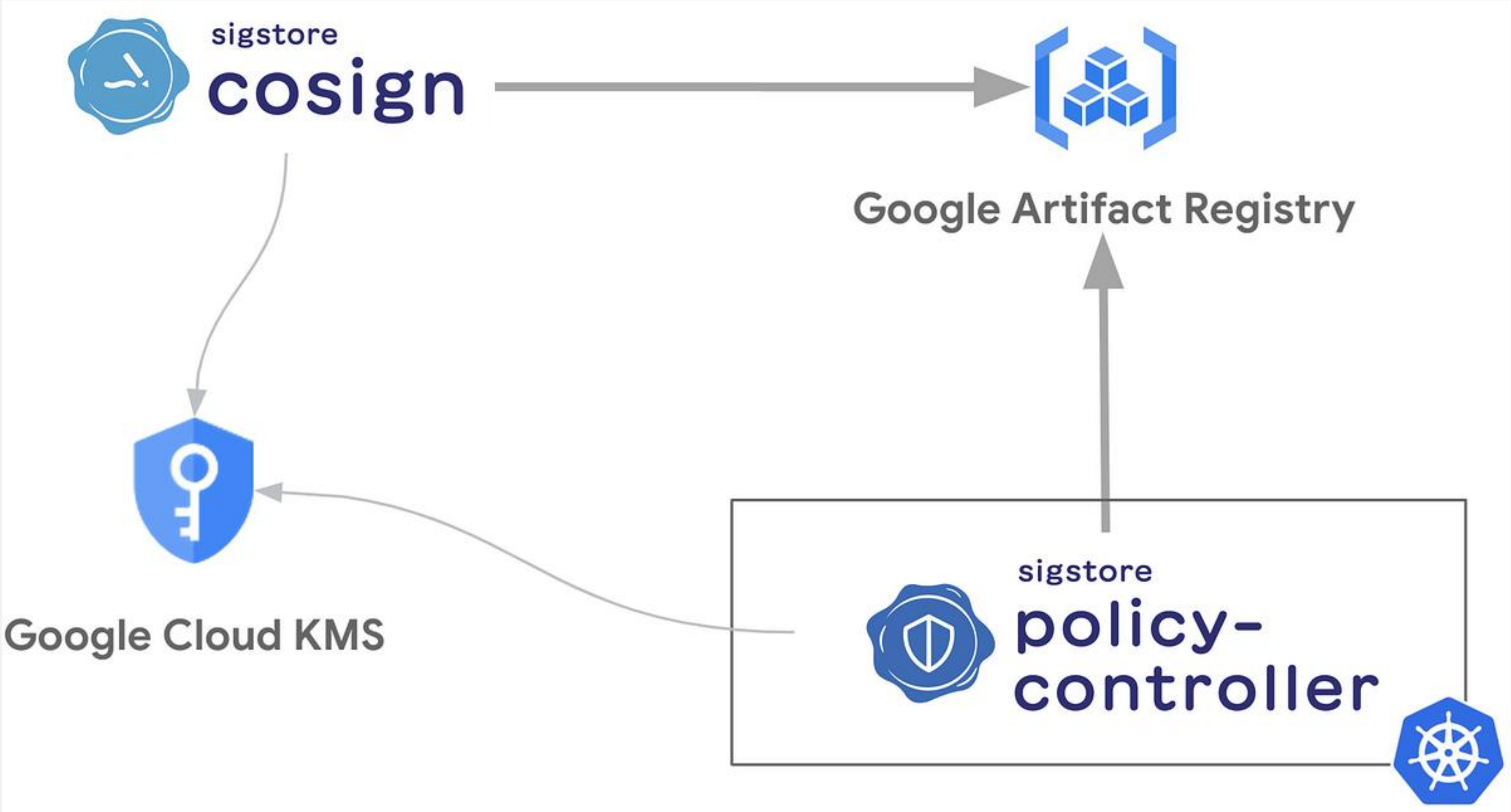

Istio Mesh
Data plane
APIs Content
JWT+TLS mTLS
Ingress
mTLS
Service A
Proxy
mTLS
HTTP, gRPC, TCP
Service B
Proxy
mTLS
Egress
JWT +TLS mTLS
External API
Control Plane Interface
Control plane
istiod
Certificate authority
Authentication policies
Network configuration
Authorization policies
API server configuration
Key:
Data plane traffic
Control plane traffic
Local authorization
Certificate

Image scanning - Trivy

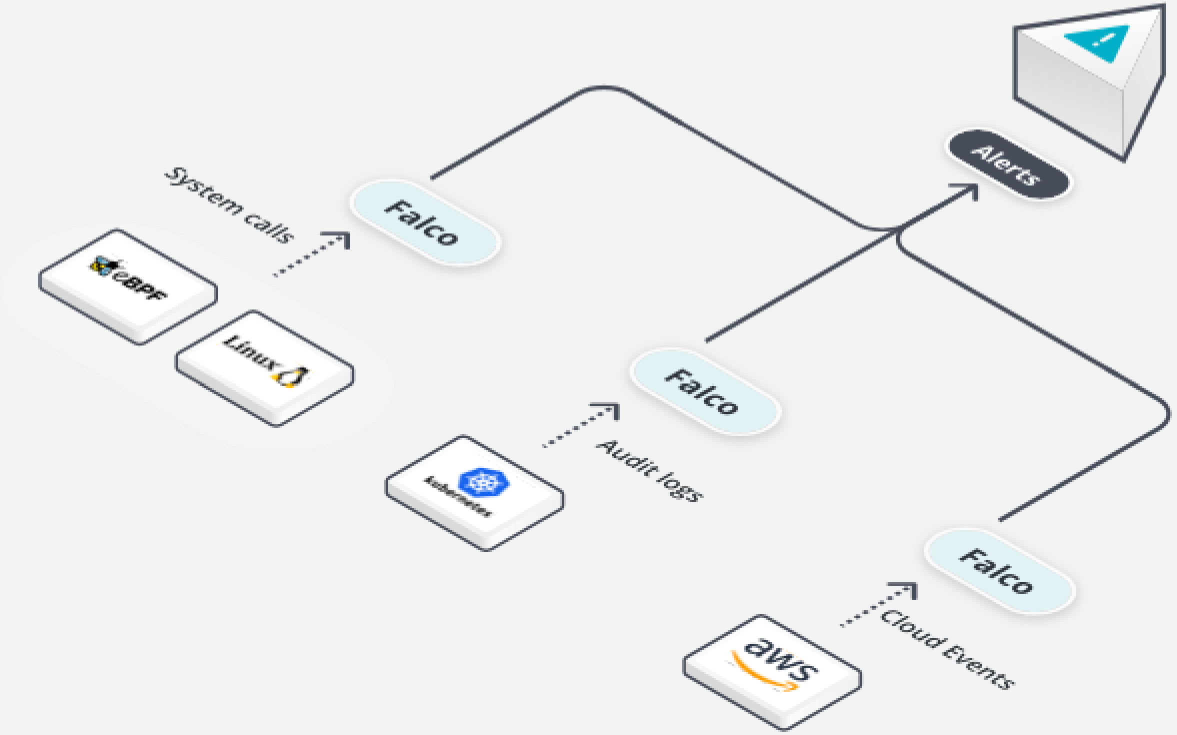


```
~  trivy k8s --report summary
179 / 179 [---------------------------------------------------------------------------------------------------] 100.00% 12 p/s

Summary Report for k3d-first-cluster
┌─────────────┬────────────────────────────────────┬─────────────────────────────┬─────────────────────────────┬─────────────────────────────┐
│  Namespace  │              Resource              │       Vulnerabilities       │      Misconfigurations      │           Secrets           │
│             │                                    ├─────┬─────┬─────┬─────┬─────┼─────┬─────┬─────┬─────┬─────┼─────┬─────┬─────┬─────┬─────┤
│             │                                    │  C  │  H  │  M  │  L  │  U  │  C  │  H  │  M  │  L  │  U  │  C  │  H  │  M  │  L  │  U  │
├─────────────┼────────────────────────────────────┼─────┼─────┼─────┼─────┼─────┼─────┼─────┼─────┼─────┼─────┼─────┼─────┼─────┼─────┼─────┤
│ kube-system │ Deployment/local-path-provisioner  │  2  │  5  │  2  │     │  1  │     │     │  8  │ 11  │     │     │     │     │     │     │
│ kube-system │ Deployment/metrics-server          │     │  2  │  1  │     │  1  │     │     │  6  │  8  │     │     │     │     │     │     │
│ kube-system │ Deployment/traefik                 │  3  │  5  │  1  │     │  3  │     │     │  7  │  7  │     │     │     │     │     │     │
│ kube-system │ DaemonSet/svclb-traefik            │  2  │ 21  │  2  │     │     │     │  4  │ 16  │ 20  │     │     │     │     │     │     │
│ kube-system │ DaemonSet/svclb-traefik            │  2  │ 21  │  2  │     │     │     │  4  │ 16  │ 20  │     │     │     │     │     │     │
│ kube-system │ Job/helm-install-traefik           │ 10  │ 54  │ 20  │  1  │ 14  │     │     │  8  │ 11  │     │     │     │     │     │     │
│ kube-system │ Job/helm-install-traefik-crd       │ 10  │ 54  │ 20  │  1  │ 14  │     │     │  8  │ 11  │     │     │     │     │     │     │
│ kube-system │ Deployment/coredns                 │     │  1  │     │     │  1  │     │     │  8  │  5  │     │     │     │     │     │     │
│ kube-system │ Service/kube-dns                   │     │     │     │     │     │     │     │  2  │  2  │     │     │     │     │     │     │
│ kube-system │ Service/metrics-server             │     │     │     │     │     │     │     │  2  │  2  │     │     │     │     │     │     │
│ kube-system │ Service/traefik                    │     │     │     │     │     │     │     │  2  │  2  │     │     │     │     │     │     │
│ default     │ Service/mysql                      │     │     │     │     │     │     │     │  1  │  2  │     │     │     │     │     │     │
│ default     │ Service/mysql-headless             │     │     │     │     │     │     │     │  1  │  2  │     │     │     │     │     │     │
│ default     │ StatefulSet/mysql                  │ 12  │ 36  │ 26  │ 113 │     │     │     │  7  │ 12  │     │     │     │     │     │     │
│ default     │ Pod/thisisfine                     │ 43  │ 217 │ 196 │ 514 │  2  │     │     │  9  │ 11  │     │     │     │     │     │     │
│ default     │ Pod/nginx                          │  6  │ 18  │ 24  │ 92  │     │     │     │  9  │ 11  │     │     │     │     │     │     │
│ default     │ Service/kubernetes                 │     │     │     │     │     │     │     │  1  │  2  │     │     │     │     │     │     │
└─────────────┴────────────────────────────────────┴─────┴─────┴─────┴─────┴─────┴─────┴─────┴─────┴─────┴─────┴─────┴─────┴─────┴─────┴─────┘
Severities: C=CRITICAL H=HIGH M=MEDIUM L=LOW U=UNKNOWN
```

Image signing – cosign (honourable mention – chainguard)

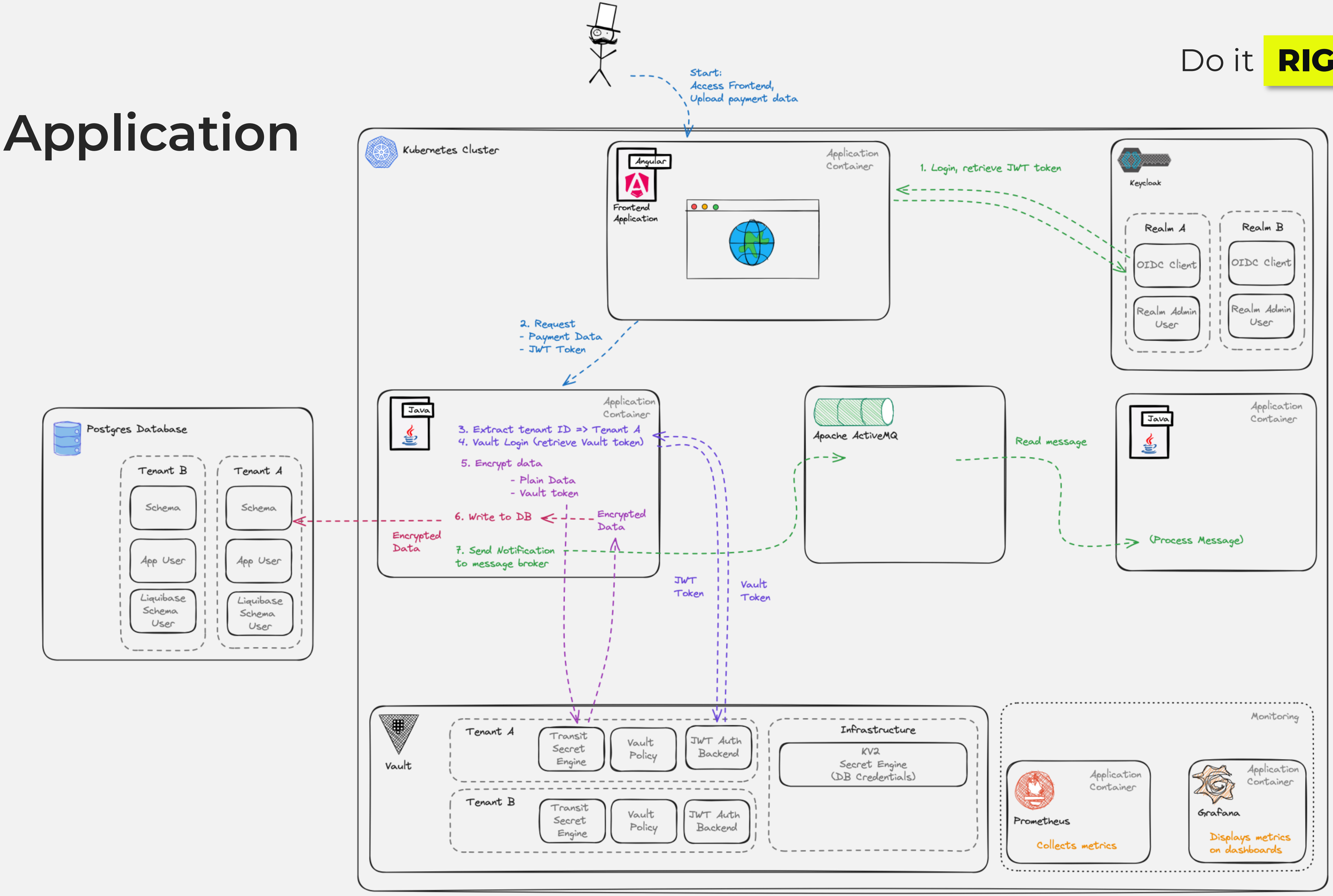


Threat detection - Falco



Do it **RIGHT.**

The Application



Start:
Access Frontend,
Upload payment data

Kubernetes Cluster

Angular
Frontend Application
Application Container

1. Login, retrieve JWT token

Keycloak

Realm A
OIDC Client
Realm Admin User

Realm B
OIDC Client
Realm Admin User

2. Request
- Payment Data
- JWT Token

Java
Application Container

3. Extract tenant ID => Tenant A
4. Vault Login (retrieve Vault token)
5. Encrypt data
- Plain Data
- Vault token

6. Write to DB

Encrypted Data

7. Send Notification to message broker

JWT Token

Vault Token

Postgres Database

Tenant B
Schema
App User
Liquibase Schema User

Tenant A
Schema
App User
Liquibase Schema User

Encrypted Data

Apache ActiveMQ

Read message

Java
Application Container
(Process Message)

Vault

Tenant A
Transit Secret Engine
Vault Policy
JWT Auth Backend

Tenant B
Transit Secret Engine
Vault Policy
JWT Auth Backend

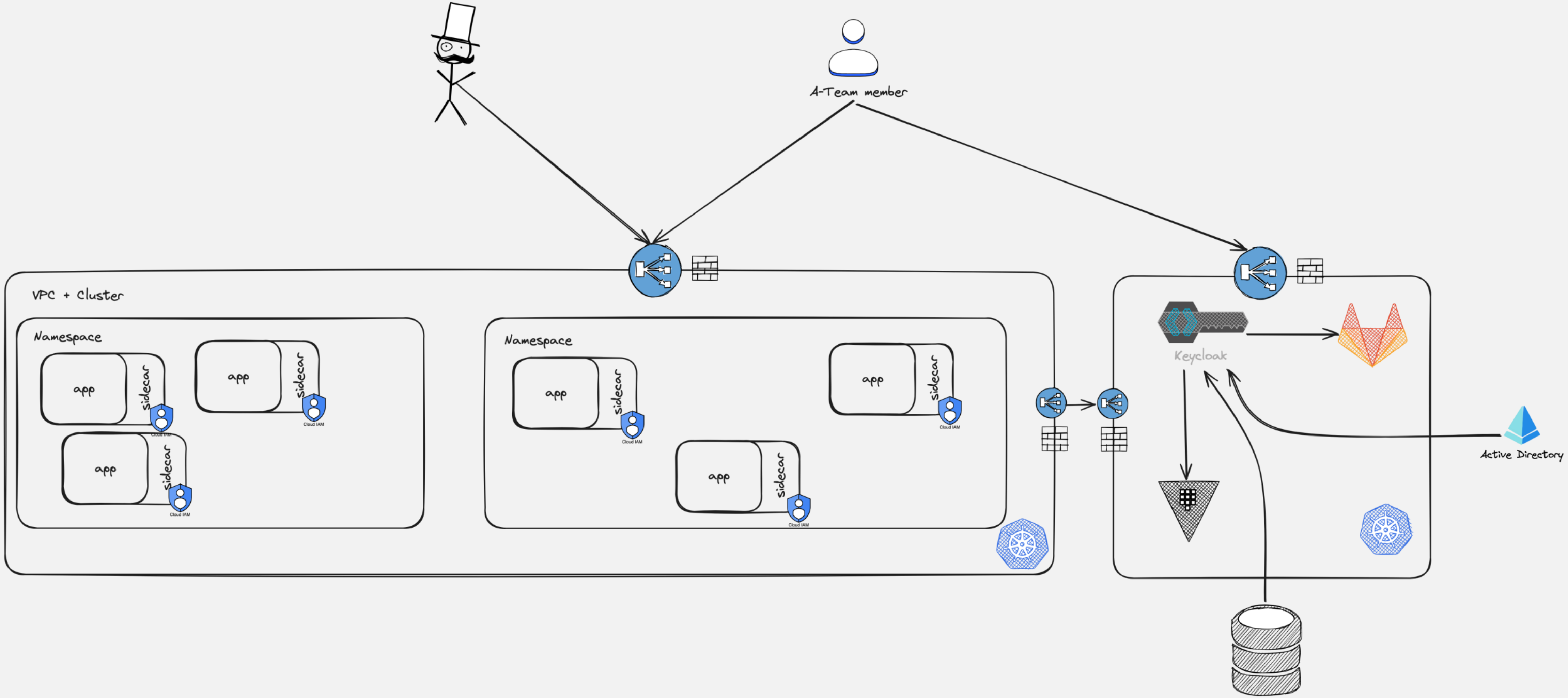
Infrastructure
KV2 Secret Engine (DB Credentials)

Monitoring

Prometheus
Application Container
Collects metrics

Grafana
Application Container
Displays metrics on dashboards

The Platform



CONTACT US:

Weyringergasse 1-3/DG
1040 Wien

www.posedio.com
office@posedio.com

THANK YOU!