Solving the never ending requirements of authorization

Alex Olivier
@alexolivier
alex@cerbos.dev

cerbos

Hi!

Engineer -> Product

Started at Microsoft

Worked in SaaS for 10 years

MarTech, eComm, Supply Chain, Connected Fitness

Data & Infra focused

Now on open-source Cerbos

Just to be sure

AuthN ≠ AuthZ

“In this world nothing is certain but death and taxes” - Benjamin Franklin

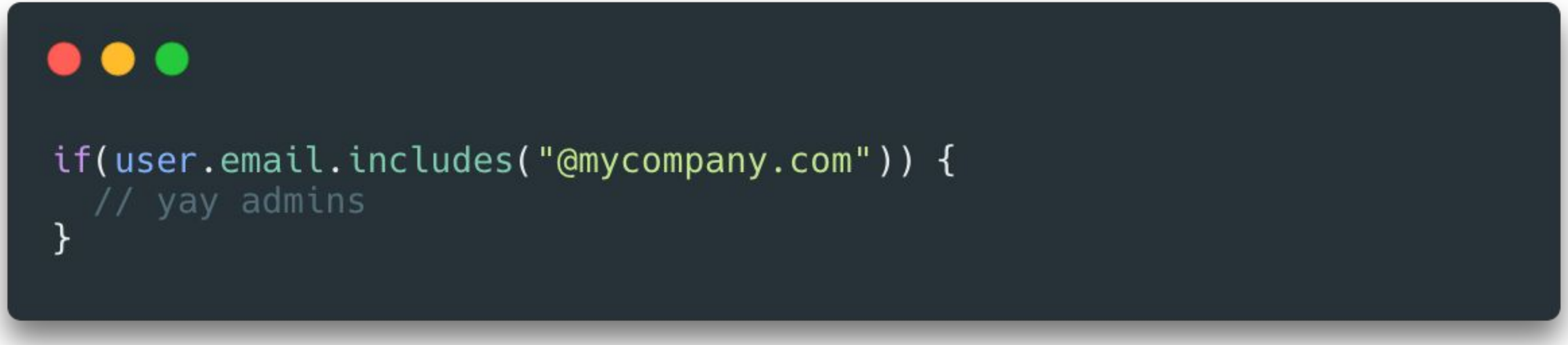
“In this world nothing is certain but death and taxes” - Benjamin Franklin

“....and ever changing authorization requirements”
- Every Product Manager

Let’s scale a company

Stage #1

The blissful days of roles

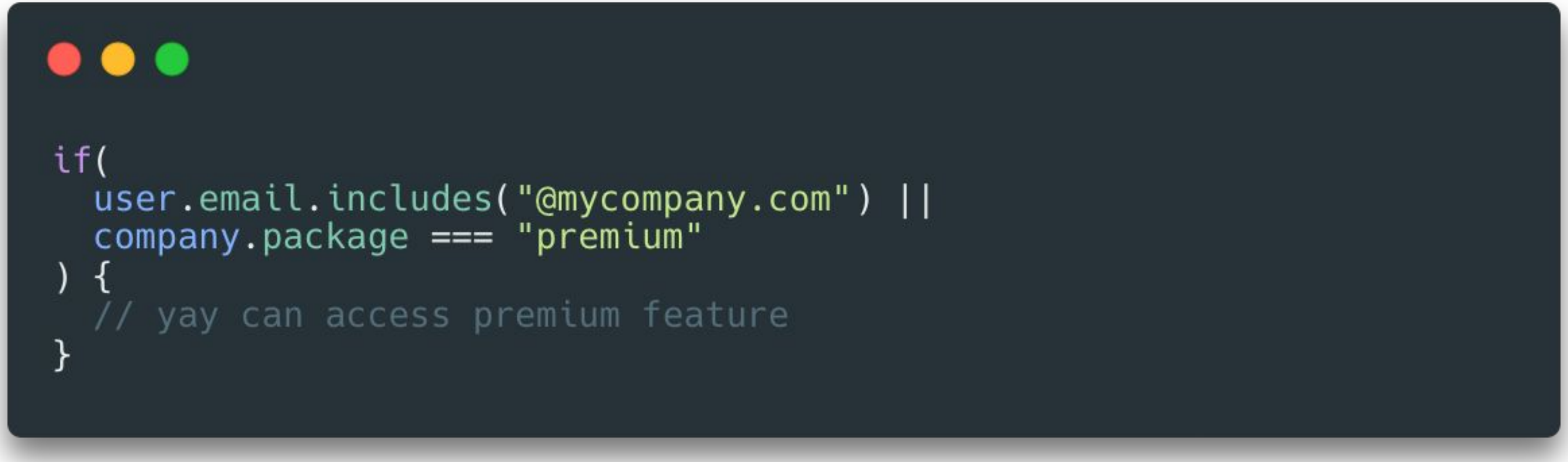


```
if(user.email.includes("@mycompany.com")) {
  // yay admins
}
```

Maybe use OAuth Scopes?

Stage #2

Sales Team: Let’s change our product packaging

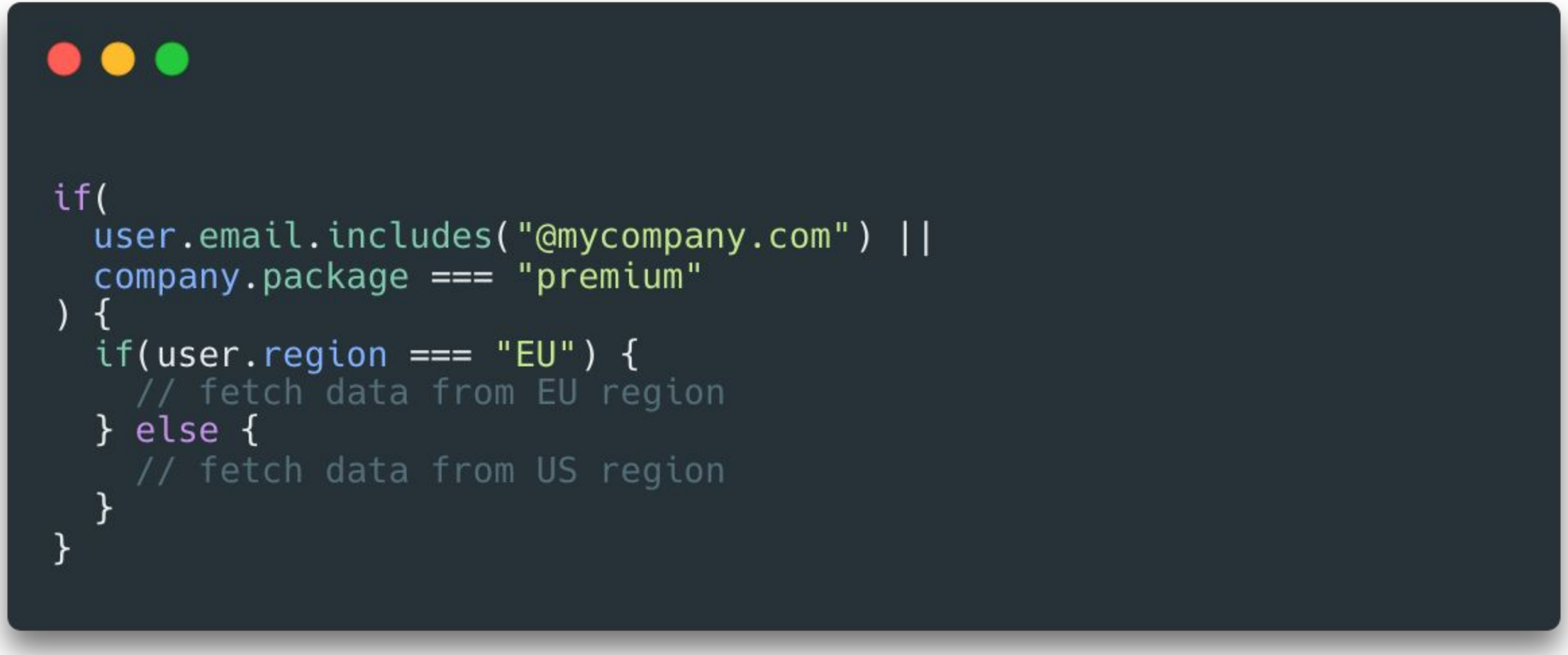


```
if(
  user.email.includes("@mycompany.com") ||
  company.package === "premium"
) {
  // yay can access premium feature
}
```

Maybe use feature flags?

Stage #3

Sales Team: Let’s sell into another region

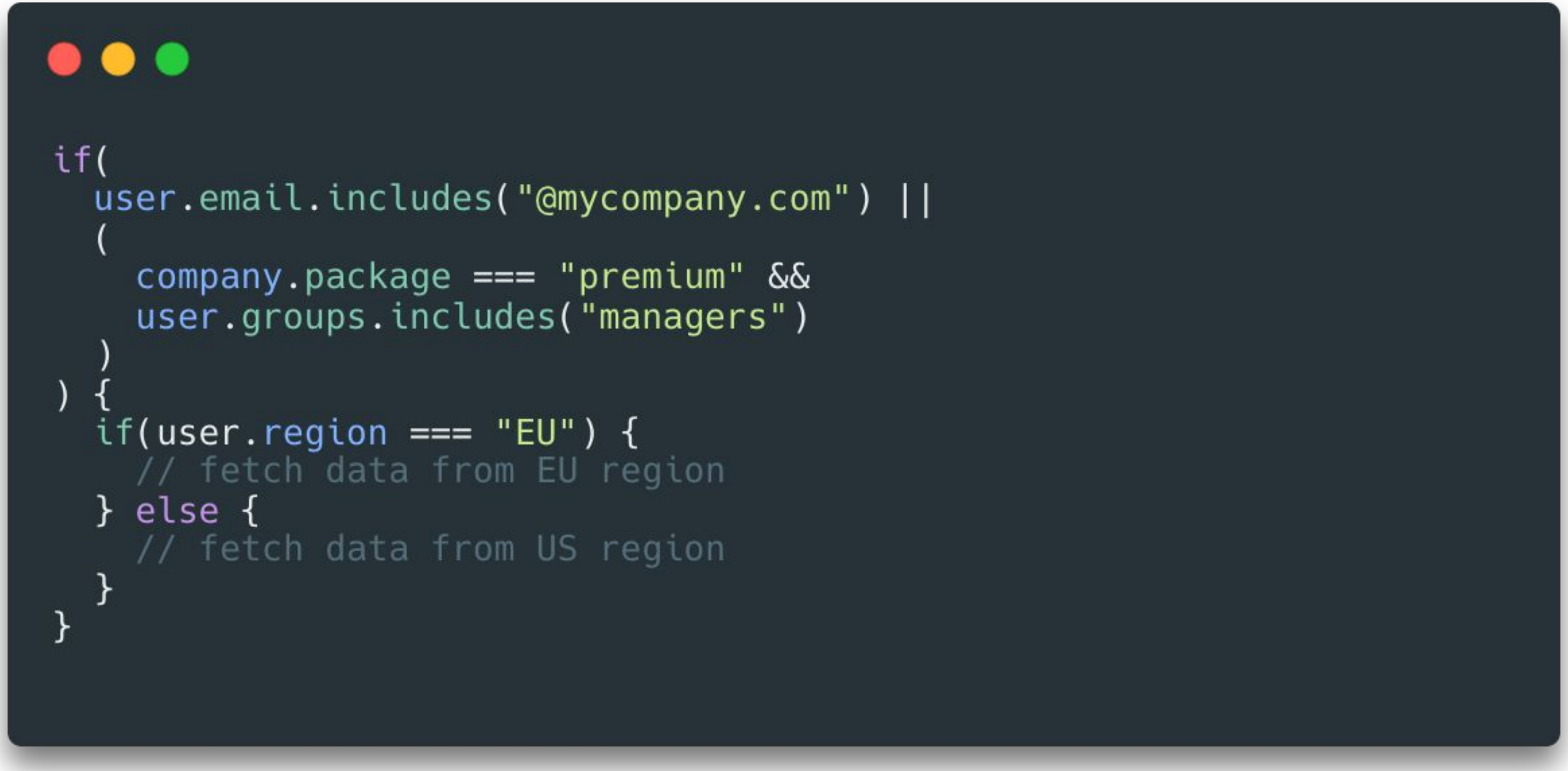


```
if(
  user.email.includes("@mycompany.com") ||
  company.package === "premium"
) {
  if(user.region === "EU") {
    // fetch data from EU region
  } else {
    // fetch data from US region
  }
}
```

Maybe use OAuth Scopes or
CASL/CanCanCan?

Stage #4

Sales Team: Let’s sell to ‘enterprise’ organisations

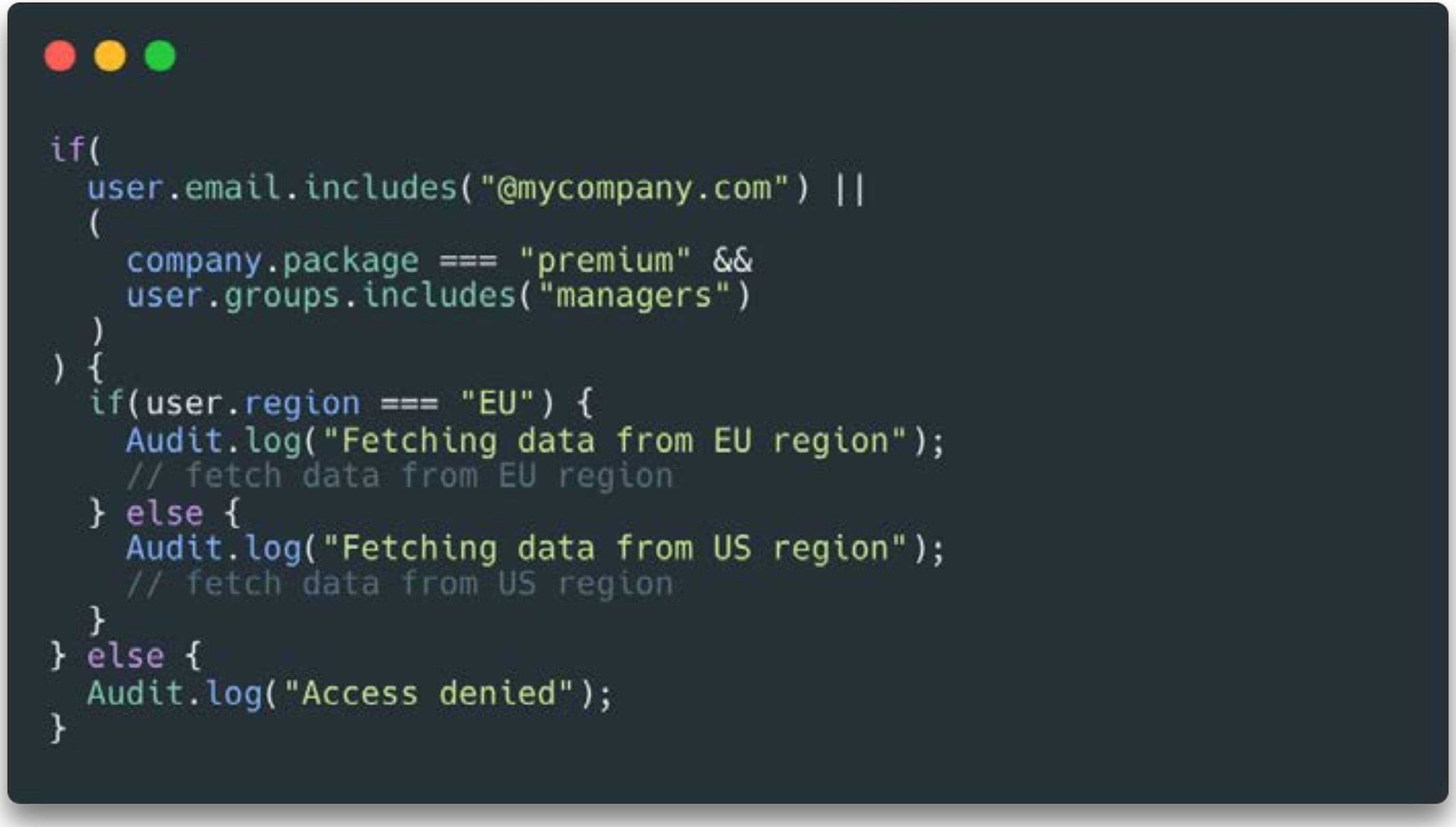


```
if(
  user.email.includes("@mycompany.com") ||
  (
    company.package === "premium" &&
    user.groups.includes("managers")
  )
) {
  if(user.region === "EU") {
    // fetch data from EU region
  } else {
    // fetch data from US region
  }
}
```

Maybe use directory attributes with
custom logic?

Stage #5

New CISO: Let’s get ISO27001/SOC2

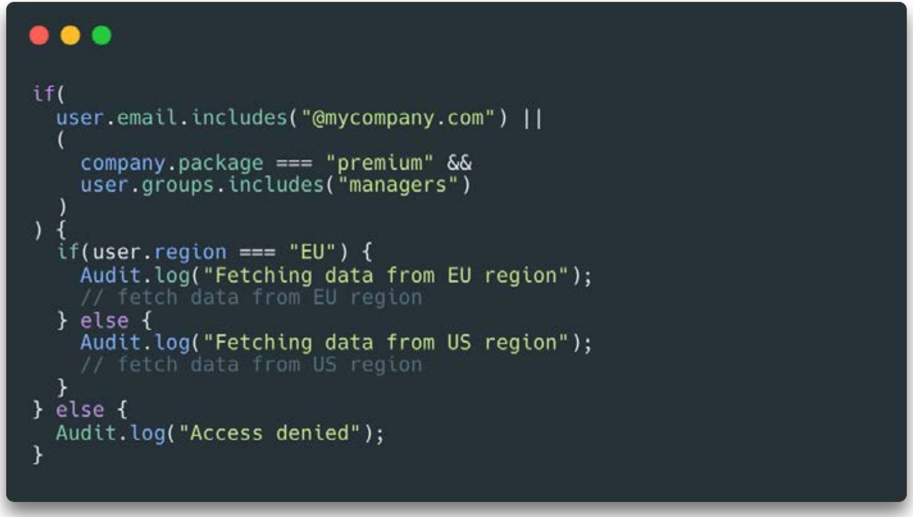


```
if(
  user.email.includes("@mycompany.com") ||
  (
    company.package === "premium" &&
    user.groups.includes("managers")
  )
) {
  if(user.region === "EU") {
    Audit.log("Fetching data from EU region");
    // fetch data from EU region
  } else {
    Audit.log("Fetching data from US region");
    // fetch data from US region
  }
} else {
  Audit.log("Access denied");
}
```

Maybe use a logging library?

Stage #6

Eng: We need microservices!



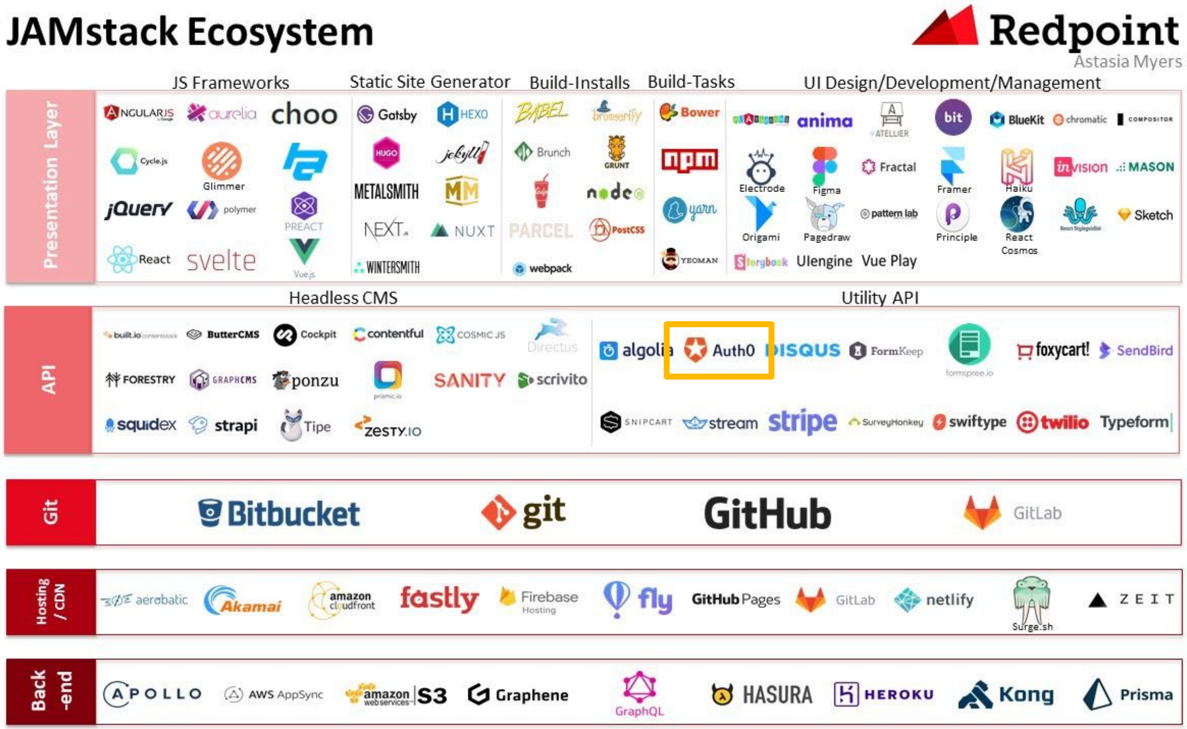
```
if(
  user.email.includes("@mycompany.com") ||
  (
    company.package === "premium" &&
    user.groups.includes("managers")
  )
) {
  if(user.region === "EU") {
    Audit.log("Fetching data from EU region");
    // fetch data from EU region
  } else {
    Audit.log("Fetching data from US region");
    // fetch data from US region
  }
} else {
  Audit.log("Access denied");
}
```

Translate

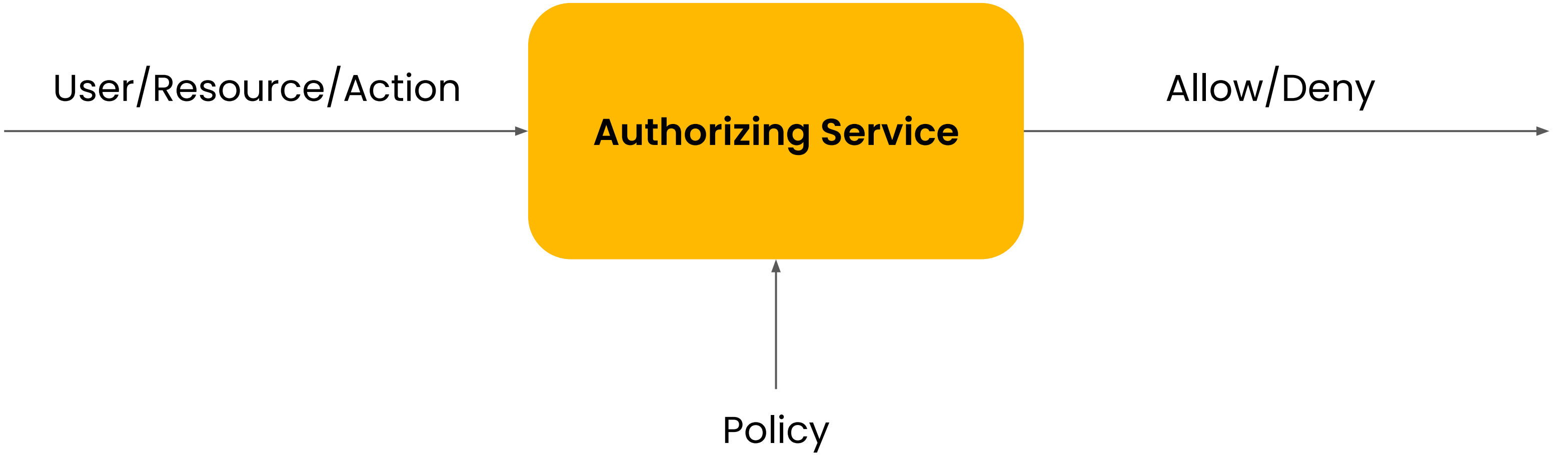
Use ???

A New Approach

Build v ‘Buy’ // *X*-aaS



“Authorization-as-a-Service”?



User/Resource/Action

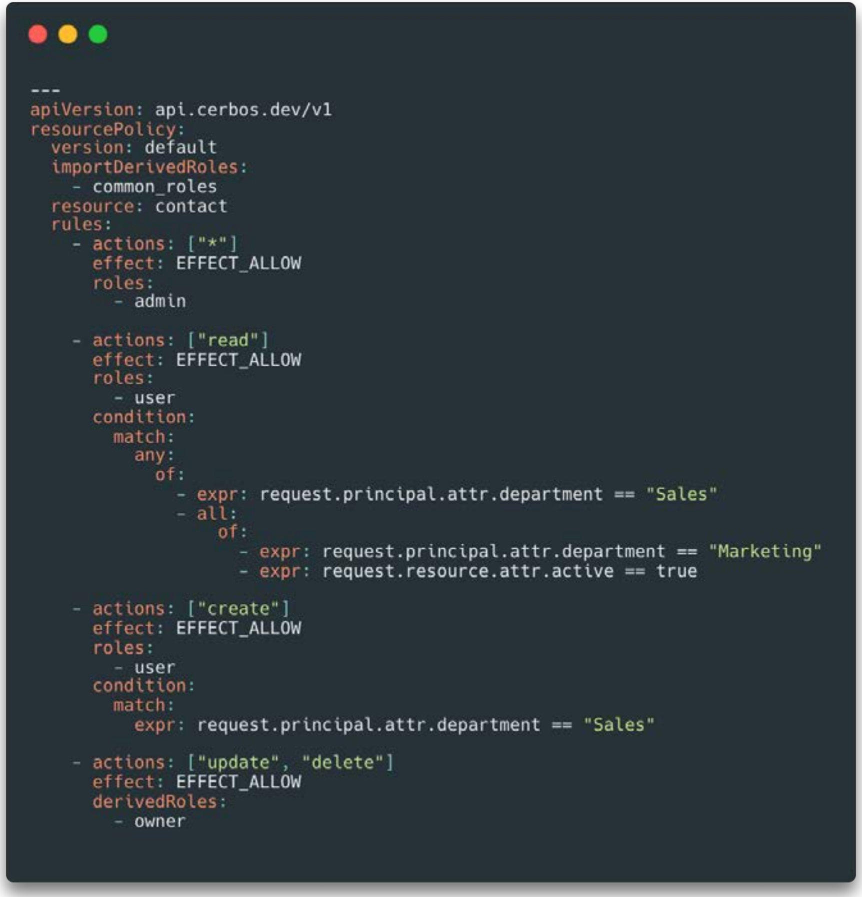
Authorizing Service

Allow/Deny

Policy

Code -> Policy

Complex authorization logic can now be defined in easy to read YAML configuration files decoupled from your application code



```yaml
---
apiVersion: api.cerbos.dev/v1
resourcePolicy:
  version: default
  importDerivedRoles:
    - common_roles
  resource: contact
  rules:
    - actions: ["*"]
      effect: EFFECT_ALLOW
      roles:
        - admin

    - actions: ["read"]
      effect: EFFECT_ALLOW
      roles:
        - user
      condition:
        match:
          any:
            of:
              - expr: request.principal.attr.department == "Sales"
              - all:
                  of:
                    - expr: request.principal.attr.department == "Marketing"
                    - expr: request.resource.attr.active == true

    - actions: ["create"]
      effect: EFFECT_ALLOW
      roles:
        - user
      condition:
        match:
          expr: request.principal.attr.department == "Sales"

    - actions: ["update", "delete"]
      effect: EFFECT_ALLOW
      derivedRoles:
        - owner
```

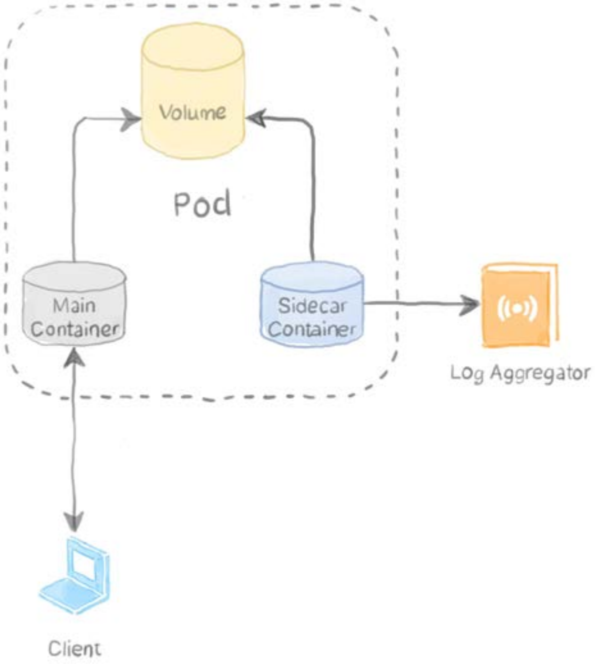
Rise of sidecars

Driven by k8s

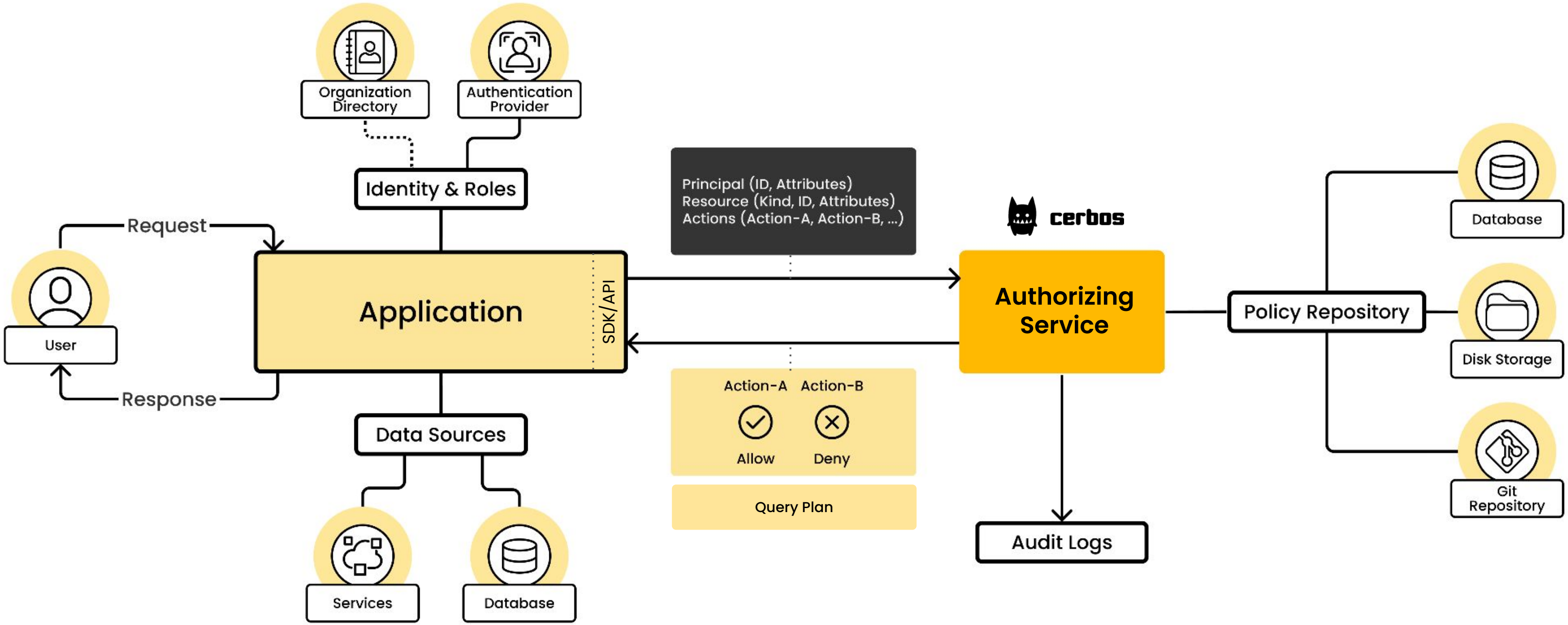
Used for AuthN, logging, metrics etc.

Useful as co-located with instances

kubernetes

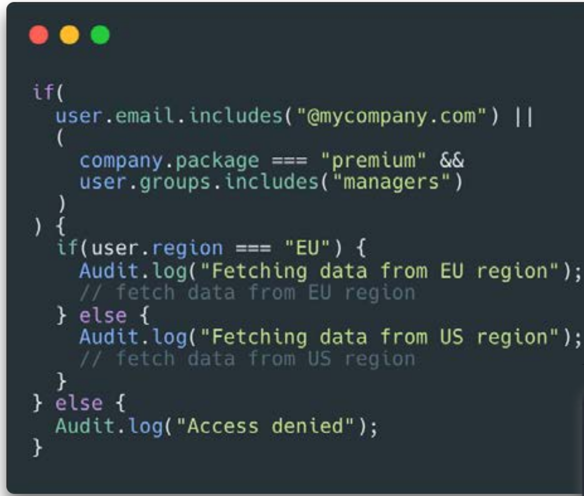


In practice



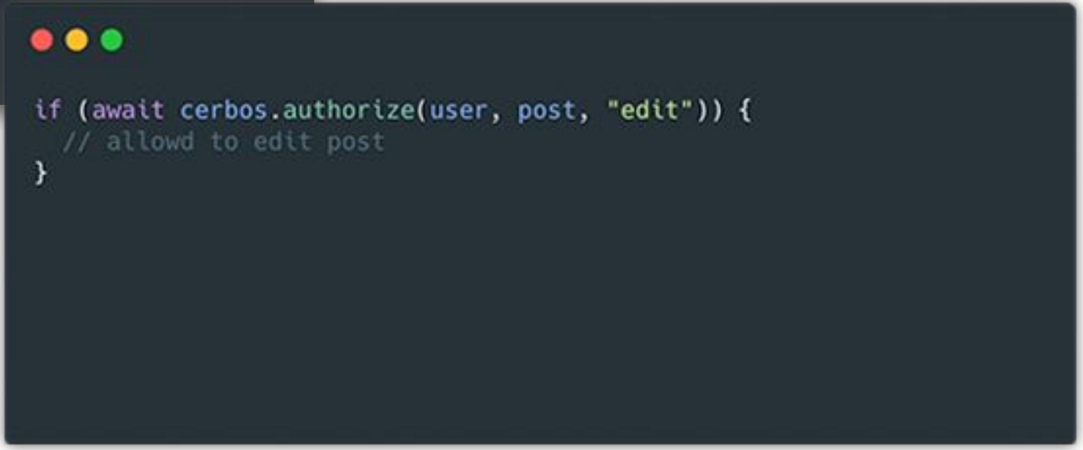
Principal (ID, Attributes)
Resource (Kind, ID, Attributes)
Actions (Action-A, Action-B, ...)

Before



```
if(
  user.email.includes("@mycompany.com") ||
  (
    company.package === "premium" &&
    user.groups.includes("managers")
  )
) {
  if(user.region === "EU") {
    Audit.log("Fetching data from EU region");
    // fetch data from EU region
  } else {
    Audit.log("Fetching data from US region");
    // fetch data from US region
  }
} else {
  Audit.log("Access denied");
}
```

After



```
if (await cerbos.authorize(user, post, "edit")) {
  // allowd to edit post
}
```

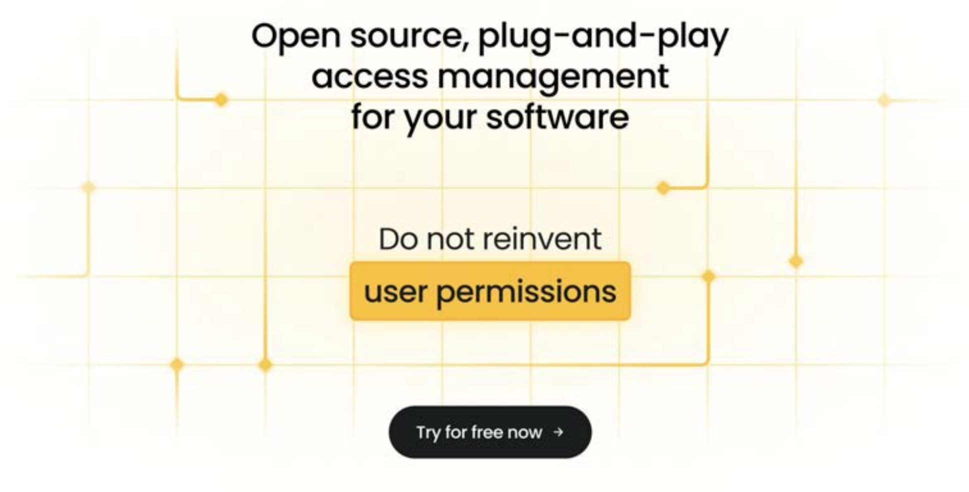
Advantages

- Logic is defined centrally
- Policy can evolve independently to code
- Language, framework and architecture agnostic
- GitOps friendly
- Consistent audit trail

Challenges

- Another service to deploy, run and scale
- A new DSL for writing policies in some cases
- A new component in the critical path

cerbos



Open source, plug-and-play access management for your software

Do not reinvent

user permissions

https://cerbos.dev

https://github.com/cerbos

Swag!

Solving the never ending requirements of authorization

Alex Olivier
@alexolivier
alex@cerbos.dev