Application Code of Conduct - Full Stack Policy as Code

Gabriel L. Manor, Director of DevRel @ Permit.io

#ossummit @gemanor

THE LINUX FOUNDATION

Permit.io

@gemanor

Hertz President's Circle

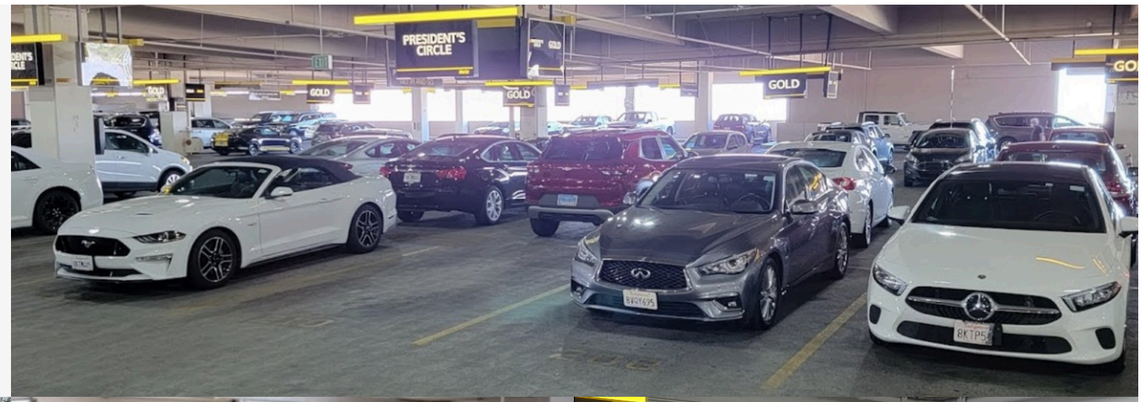
Hertz

CONFIRMATION
K3681787214

Hi, LIECHTMAN GAVRIEL D MANOR:

Welcome to Hertz Phoenix. As a loyalty member, you decide what to drive with **Hertz Ultimate Choice®**. Please note your lot assignment on our Gold Board, head straight to the **President's Circle zone** and choose your thoroughly clean car.

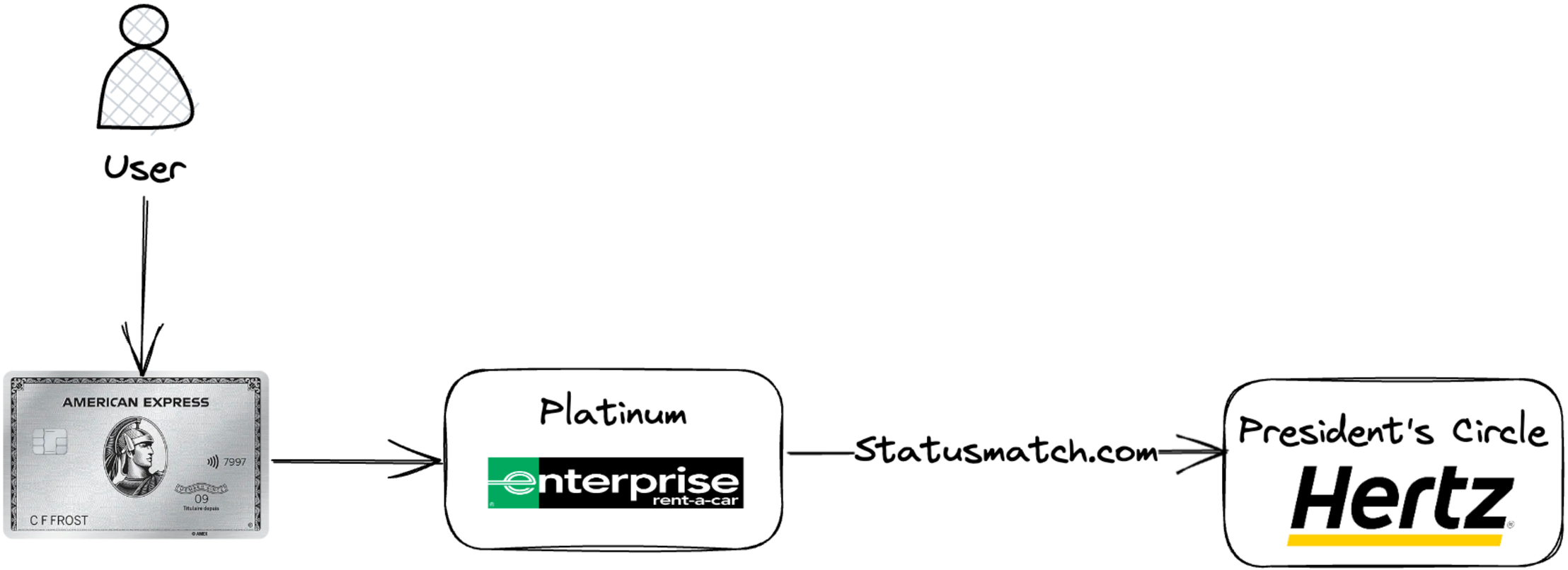
See you shortly!



Permit.io

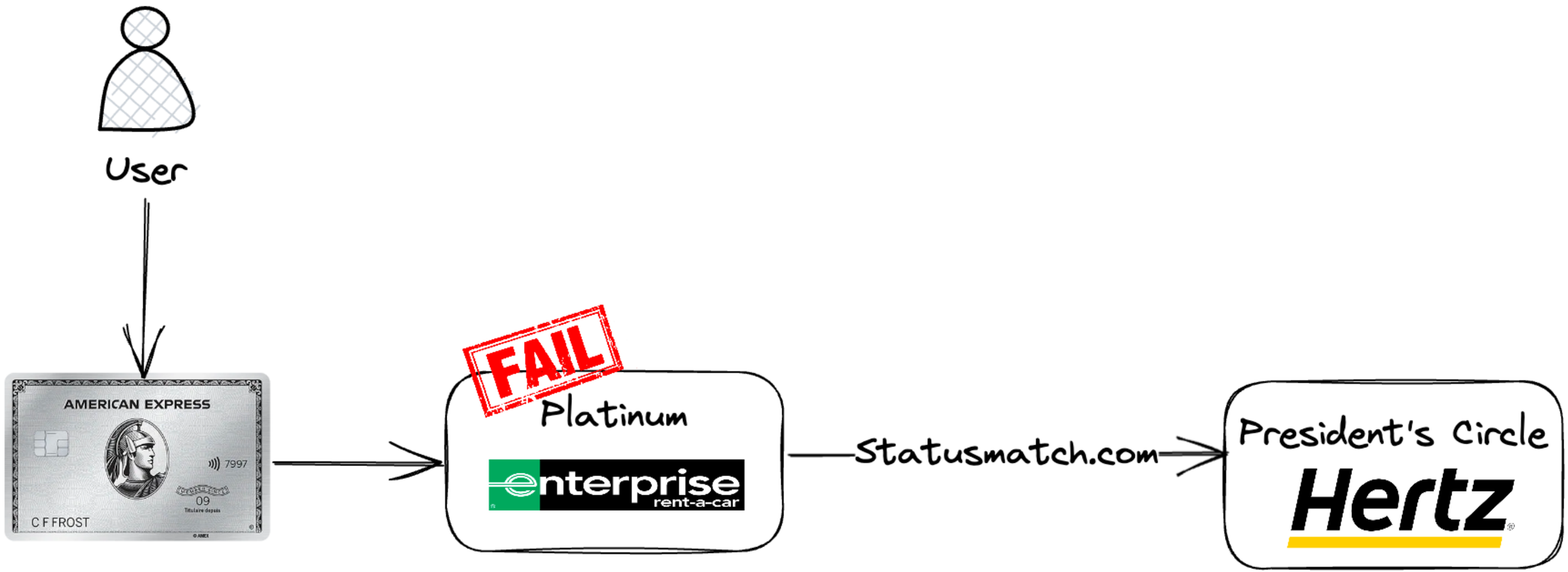
@gemanor

President's Circle®	
Qualification	15 Rentals or $3,000



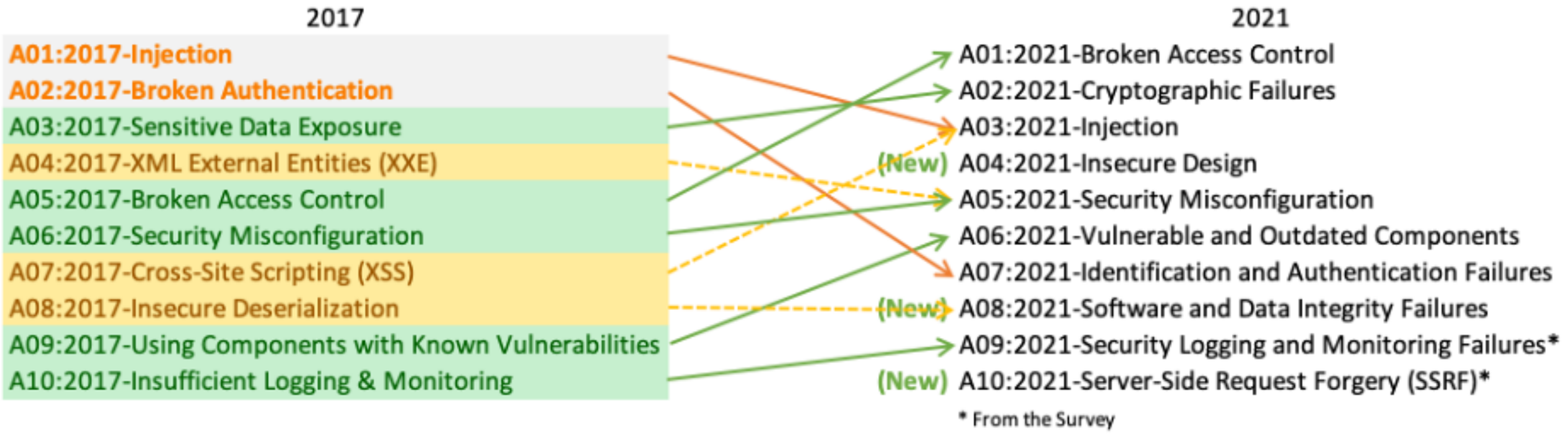
Permit.io

@gemanor



Permit.io

@gemanor


2017
A01:2017-Injection
A02:2017-Broken Authentication
A03:2017-Sensitive Data Exposure
A04:2017-XML External Entities (XXE)
A05:2017-Broken Access Control
A06:2017-Security Misconfiguration
A07:2017-Cross-Site Scripting (XSS)
A08:2017-Insecure Deserialization
A09:2017-Using Components with Known Vulnerabilities
A10:2017-Insufficient Logging & Monitoring
2021
A01:2021-Broken Access Control
A02:2021-Cryptographic Failures
A03:2021-Injection
(New) A04:2021-Insecure Design
A05:2021-Security Misconfiguration
A06:2021-Vulnerable and Outdated Components
A07:2021-Identification and Authentication Failures
(New) A08:2021-Software and Data Integrity Failures
A09:2021-Security Logging and Monitoring Failures*
(New) A10:2021-Server-Side Request Forgery (SSRF)*
* From the Survey

We Will Do RBAC Later

- Every developer on every new application

Permit.io

@gemanor

The Auth Comfort Zone

Authentication 😌

Verify who the user is

```
if (!user) {
    return;
}
```

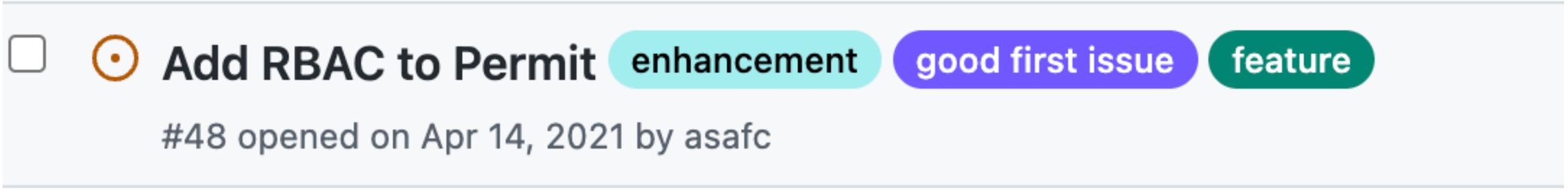
Authorization 😔

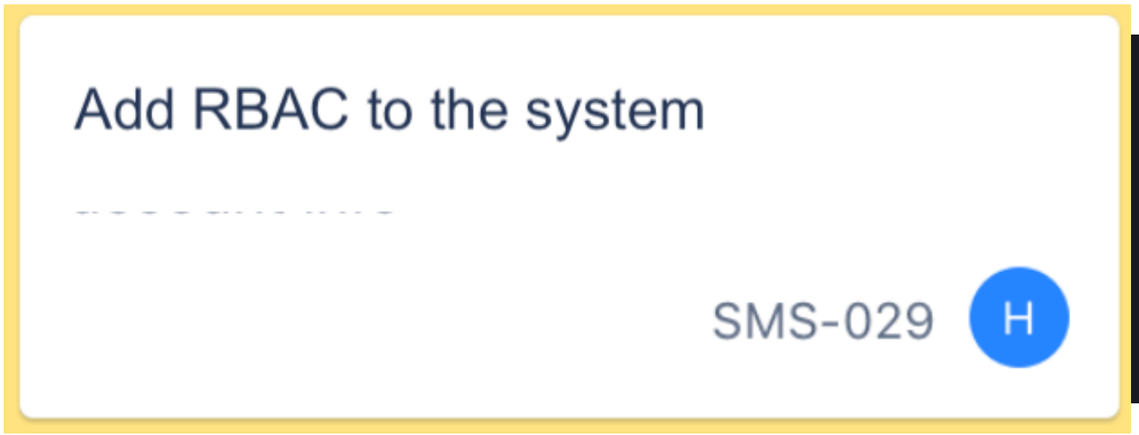
Check what a user can do

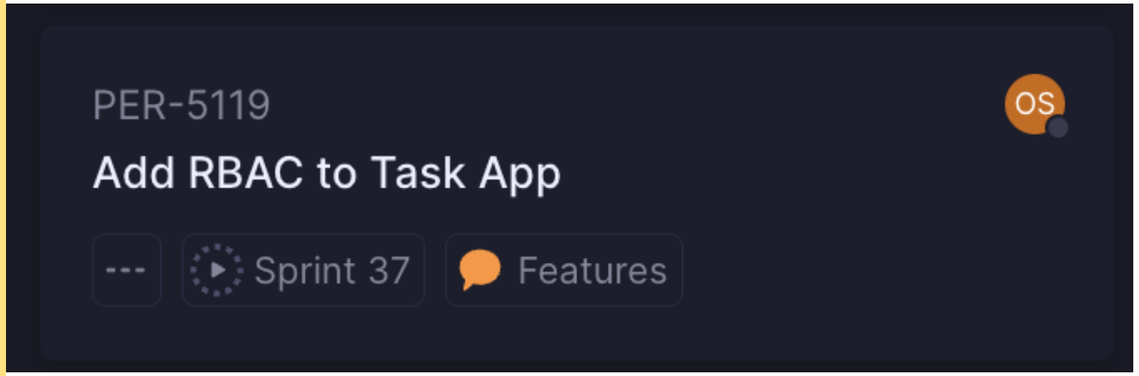
```
if (!allowed(
        user,
        action,
        resource
    )) {
    return;
}
```

Permit.io

@gemanor







Permit.io

@gemanor

“Nah, man. RBAC is pretty easy.”

- Derek, age 24

“Nah, man. RBAC is pretty easy.”

- Derek, age 24

Gabriel L. Manor

Director of DevRel @ Permit.io

🎖️ **Struggling with authorization for the last 8y**

Not an ethical hacker, zero awards winner, dark mode hater.

Permit.io

@gemanor

```
// Middleware
if (req.user.roles.indexOf(role) === -1) {
    return res.send(403);
}

// Endpoint
@authz('admin')
const Document = () => {
    ...
}
```

Permit.io

@gemanor

```
// Middleware
const hasRole = roles.filter((r) => (
    req.user.roles.indexOf(role) > -1
))
if (!hasRole) {
    return res.send(403);
}

// Endpoint
@authz('admin')
@authz('paid')
const Document = () => {
    ...
}
```

Permit.io

@gemanor

```
// Middleware
if (args.length === 2 && req.user[args[0]] !== args[1]) {
    return res.send(403);
}
if (req.user.roles.indexOf(role) > -1) {
    return res.send(403);
}

// Endpoint
@authz('admin')
@authz('location', 'eu')
const Document = () => {
    ...
}
```

Permit.io

@gemanor

```
// Middleware
if (typeof args[0] === 'object') {
    return AuthzDesicion(args[0])
}
if (args.length === 2 && req.user[args[0]] !== args[1]) {
    return res.send(403);
}
if (req.user.roles.indexOf(role) > -1) {
    return res.send(403);
}

// Endpoint
@authz('admin')
@authz('location', 'eu')
@authz({
    ...
})
const Document = () => {
    ...
}
```

Permit.io

@gemanor

Implement RBAC to an Application

Permit.io

@gemanor

~~Implement RBAC to Application~~

Enforce Permissions in Applications

Permit.io

@gemanor

User | Action | Resource

Does *[User]* Allowed to Perform *[Action]* on *[Resource]*

Does a Monkey Allowed to Eat a Banana

Permit.io

@gemanor

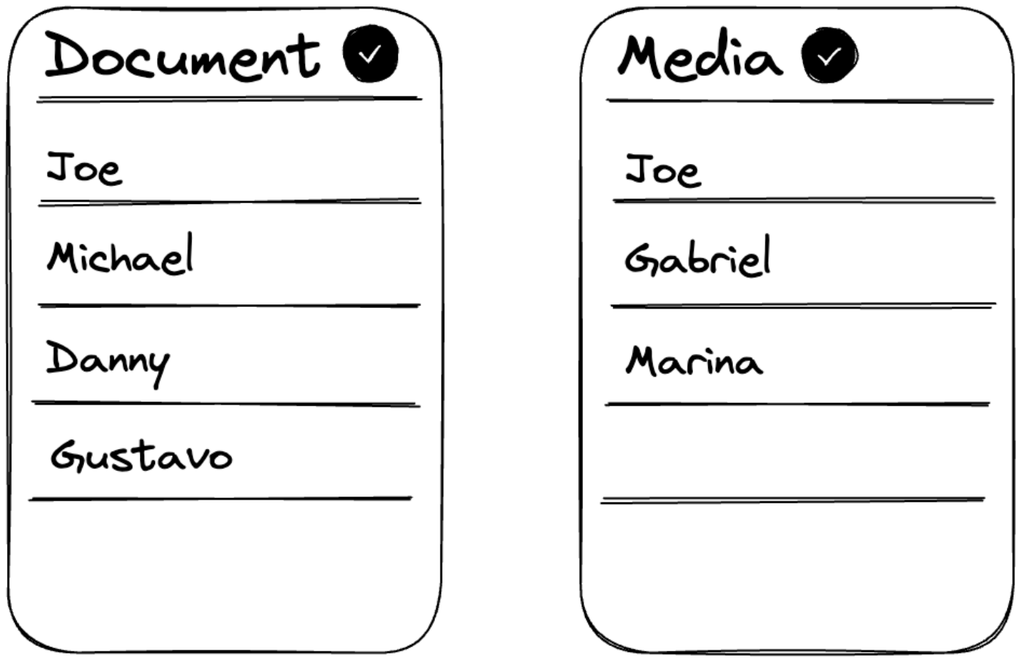
ACL - Access Control List

RBAC - Role-Based Access Control

ABAC - Attribute-Based Access Control

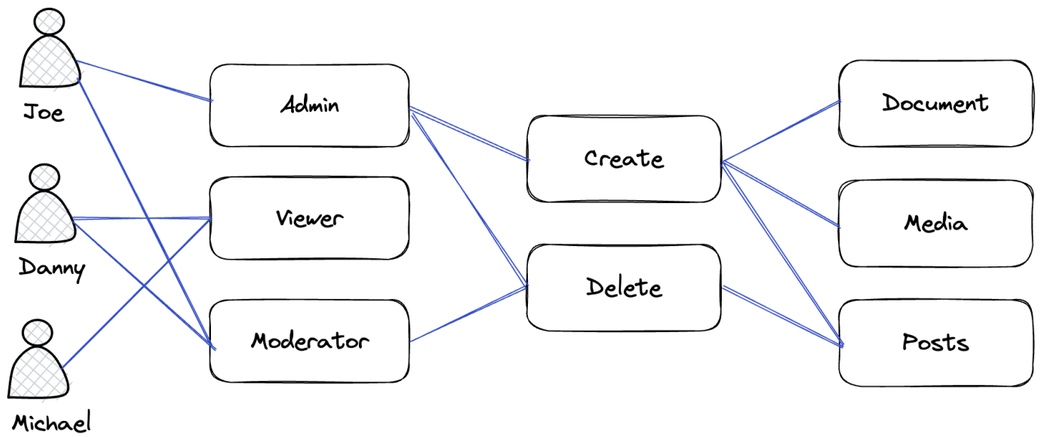
ReBAC - Relationship-Based Access Control

ACL - Access Control List



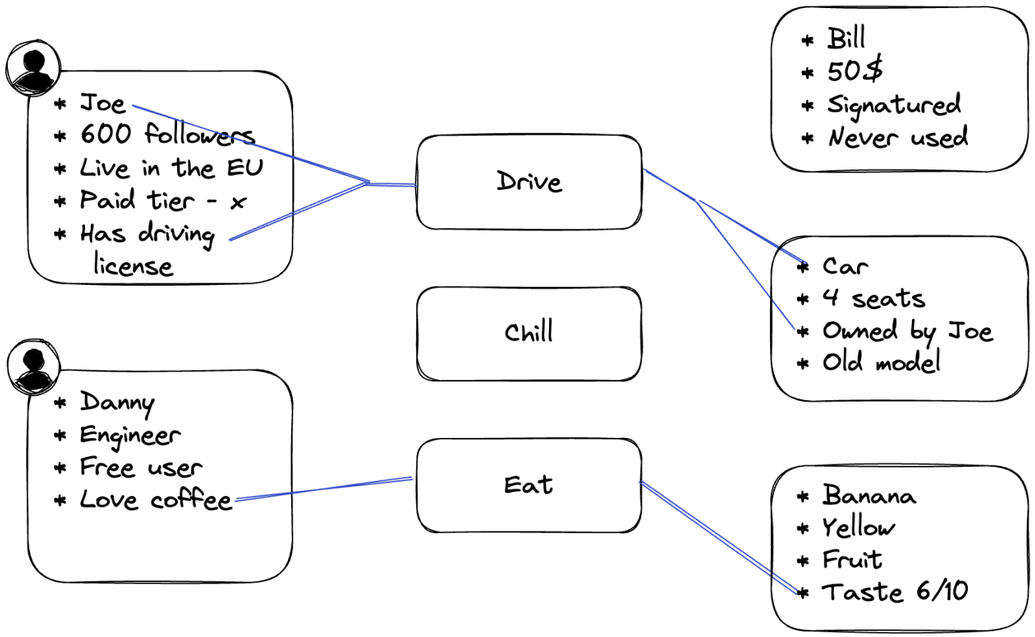
- EOL model
- Widely used in IT systems/ networks
- No segmentation/attribution support
- Hard to scale

RBAC - Role Based Access Control



- The widely-used model for app authorization
- 🤓 Easy to define, use and audit
- No resource inspection
- Limited scalablity

ABAC - Attribute Based Access Control

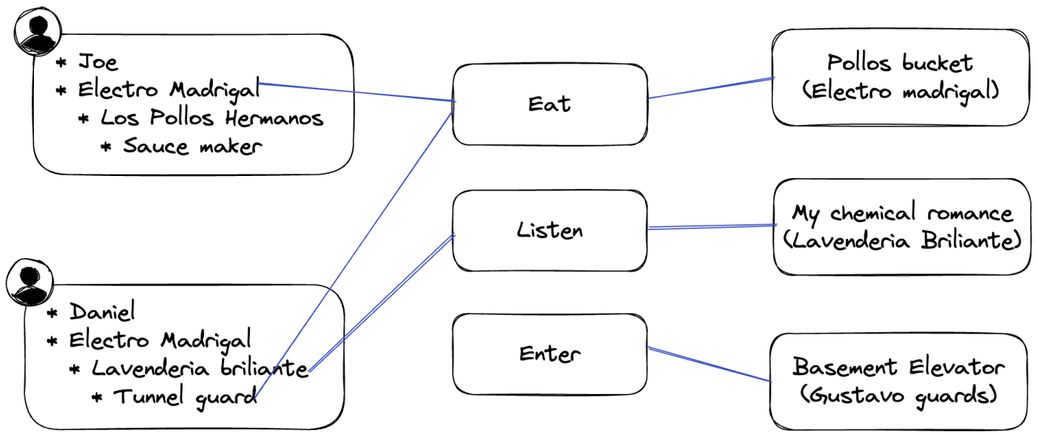


- The most robust model for inspection and desicion making
- Configuration could be hard
- Easy to handle multiple data sources
- 🚀 Highly scalable

Permit.io

@gemanor

ReBAC - Relationship Based Access Control



- Best fit for consumer-style applications
- Support in reverse indices and search for allowed data
- Easy to scale for users (>1b) hard in desicion's performance

```
// Middleware
if (typeof args[0] === 'object') {
    return AuthzDesicion(args[0])
}
if (args.length === 2 && req.user[args[0]] !== args[1]) {
    return res.send(403);
}
if (req.user.roles.indexOf(role) > -1) {
    return res.send(403);
}

// Endpoint
@authz('admin')
@authz('location', 'eu')
@authz({
    ...
})
const Document = () => {
    ...
}
```

Permit.io

@gemanor

```
// Middleware
if (typeof args[0] === 'object') {
    return AuthzDesicion(args[0])
}
if (args.length === 2 && req.user[args[0]] !== args[1]) {
    return res.send(403);
}
if (req.user.roles.indexOf(role) > -1) {
    return res.send(403);
}

// Endpoint
@authz('admin')
const Document = () => {
    const enrichedUser = getSocialData(req.user);
    if (enrichedUser.lastGeolocation !== 'eu') {
        return;
    }
    ...
}
```

Permit.io

@gemanor

```
// Middleware
if (typeof args[0] === 'object') {
    return AuthzDesicion(args[0])
}
if (args.length === 2 && req.user[args[0]] !== args[1]) {
    return res.send(403);
}
if (req.user.roles.indexOf(role) > -1) {
    return res.send(403);
}

// Endpoint
@authz('admin')
const Document = () => {
    const enrichedUser = getSocialData(req.user);
    if (enrichedUser.lastGeolocation !== 'eu') {
        return;
    }
    ...
}
```

Permit.io

@gemanor

Jessie's New Stack

- 👺 DevOps
- Frontend
- Data
- Persistent
- 🦀 Language that no one use

Jessie's New Skillset

- 🚀 Performance
- 📈 Monitoring
- 🤓 Audit logs
- 👥 User Management
- 🌩️ Ops

Jessie's New Headache 🤕

- Compliance
- Posture
- Centralized vs Decantralized
- Product managers

Authorization Top 5 Best-Practices

- Agnostic to the permissions model
- Seperate policy from code
- Support the whole stack
- Performance and monitoring features
- Unified policy management

Permit.io

@gemanor

Contracts Creates Better Relationships

👩🏽‍💻 *Especially in Human <> Machine Relationships* 🤖

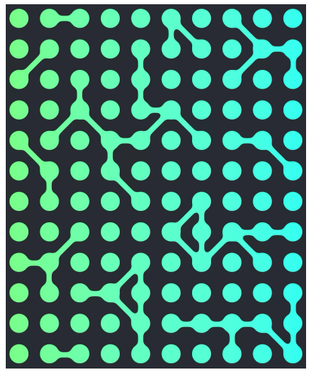

Permit.io

@gemanor

Open Policy Agent

AWS Cedar

Google Zanzibar



```rego
package app.rbac

import future.keywords.contains
import future.keywords.if
import future.keywords.in

default allow := false

allow if user_is_admin

allow if {
    some grant in user_is_granted
    input.action == grant.action
    input.type == grant.type
}

user_is_admin if "admin" in data.user_roles[input.user]

user_is_granted contains grant if {
    some role in data.user_roles[input.user]
    some grant in data.role_grants[role]
}
```

```rego
package app.abac

import future.keywords.if

default allow := false

allow if user_is_owner

allow if {
    user_is_employee
    action_is_read
}

allow if {
    user_is_employee
    user_is_senior
    action_is_update
}

allow if {
    user_is_customer
    action_is_read
    not pet_is_adopted
}

user_is_owner if data.user_attributes[input.user].title == "owner"

user_is_employee if data.user_attributes[input.user].title == "employee"

user_is_customer if data.user_attributes[input.user].title == "customer"

user_is_senior if data.user_attributes[input.user].tenure > 8

action_is_read if input.action == "read"

action_is_update if input.action == "update"

pet_is_adopted if data.pet_attributes[input.resource].adopted == true
```

Permit.io

@gemanor

Open Policy Agent

- 👍 Wide ecosystem
- 👍 Rego is comprehensive and robust
- Rego is complex 🤔
- Data cache replica
- Enforcement plugins


```
permit(
    principal in Role::"admin",
    action in [
        Action::"task:update",
        Action::"task:retrieve",
        Action::"task:list"
    ],
    resource in ResourceType::"task"
);
```

```
permit (
    principal,
    action in
        [Action::"UpdateList",
         Action::"CreateTask",
         Action::"UpdateTask",
         Action::"DeleteTask"],
    resource
)
when { principal in resource.editors };
```

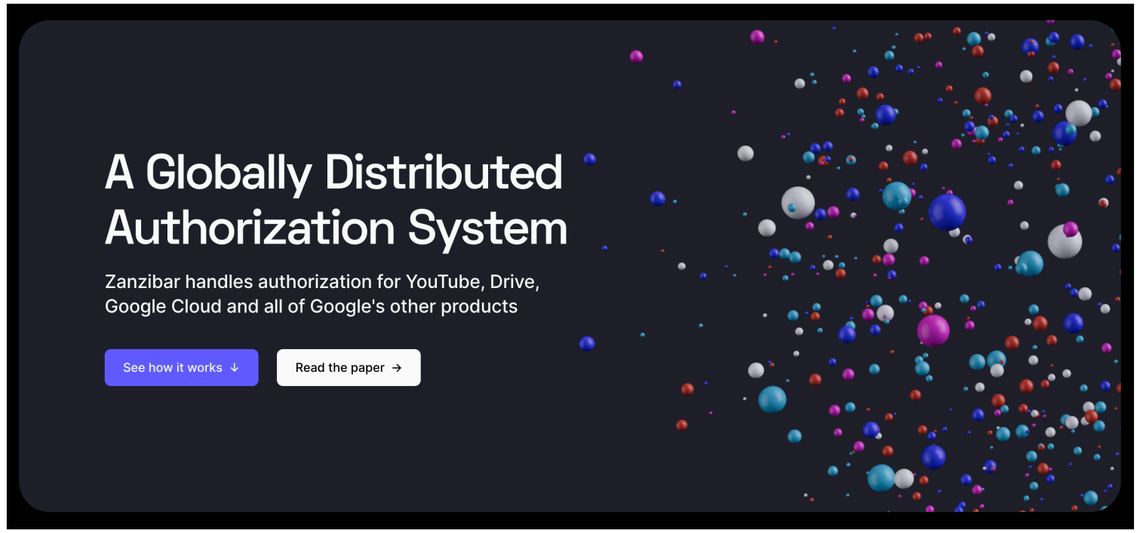
```
permit (
    principal,
    action,
    resource
)
when {
    resource has owner &&
    resource.owner == principal
};
```


Permit.io

@gemanor

AWS Cedar

- 👍 First desgined as application authz language
- 👍 Backed by AWS
- Just released 🤔
- ReBAC via ABAC
- Benchmark winner



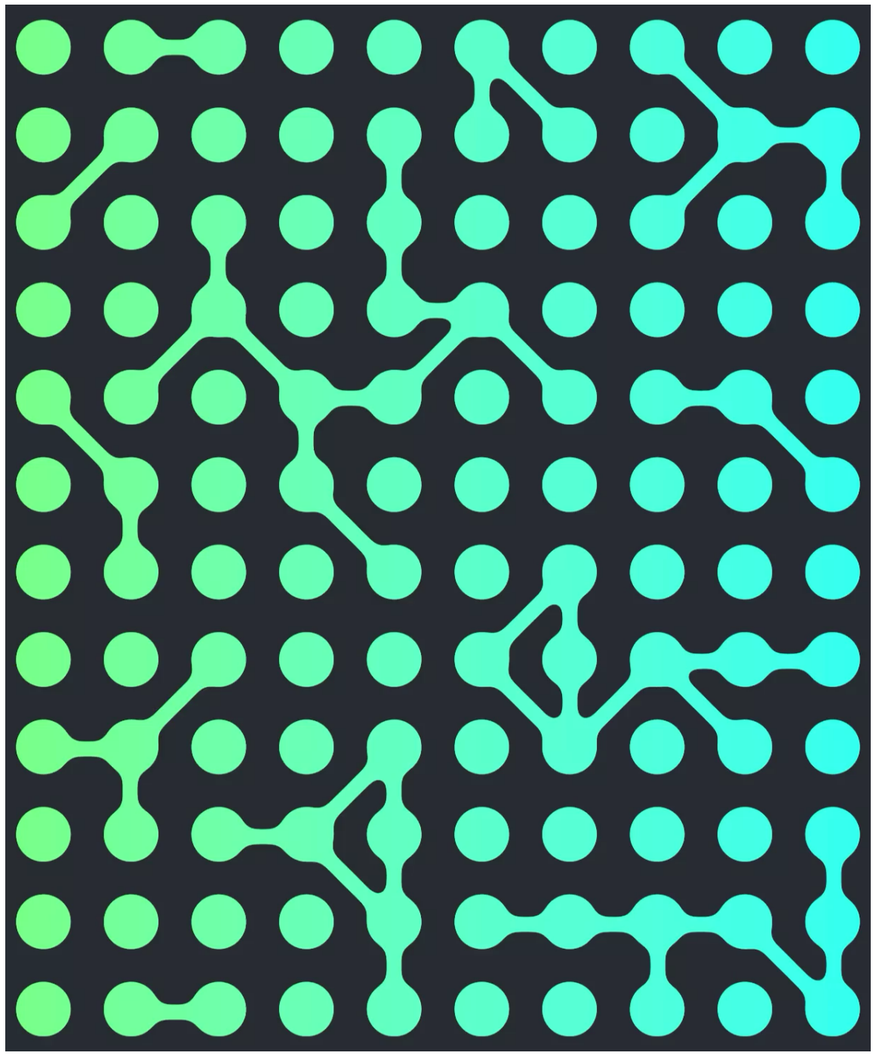

```
name: "doc"

relation { name: "owner" }

relation {
 name: "editor"
 userset_rewrite {
  union {
   child { _this {} }
   child { computed_userset { relation: "owner" } }
} } }

relation {
 name: "viewer"
 userset_rewrite {
  union {
   child { _this {} }
   child { computed_userset { relation: "editor" } }
   child { tuple_to_userset {
     tupleset { relation: "parent" }
     computed_userset {
       object: $TUPLE_USERSET_OBJECT  # parent folder
       relation: "viewer"
   } } }
} } }
```

Permit.io

@gemanor

OpenFGA Implementation

```
{
  "schema_version": "1.1",
  "type_definitions": [{
      "type": "user"
    }, {
      "type": "document",
      "relations": {
        "reader": {"this": {}},
        "writer": {"this": {}},
        "owner": {"this": {}}
      },
      "metadata": {
        "relations": {
          "reader": {
            "directly_related_user_types": [{"type": "user"}]
          },
          "writer": {
            "directly_related_user_types": [{"type": "user"}]
          },
          "owner": {
            "directly_related_user_types": [{"type": "user"}]
          }
        }
      }
    }
  ]
}
```



Permit.io

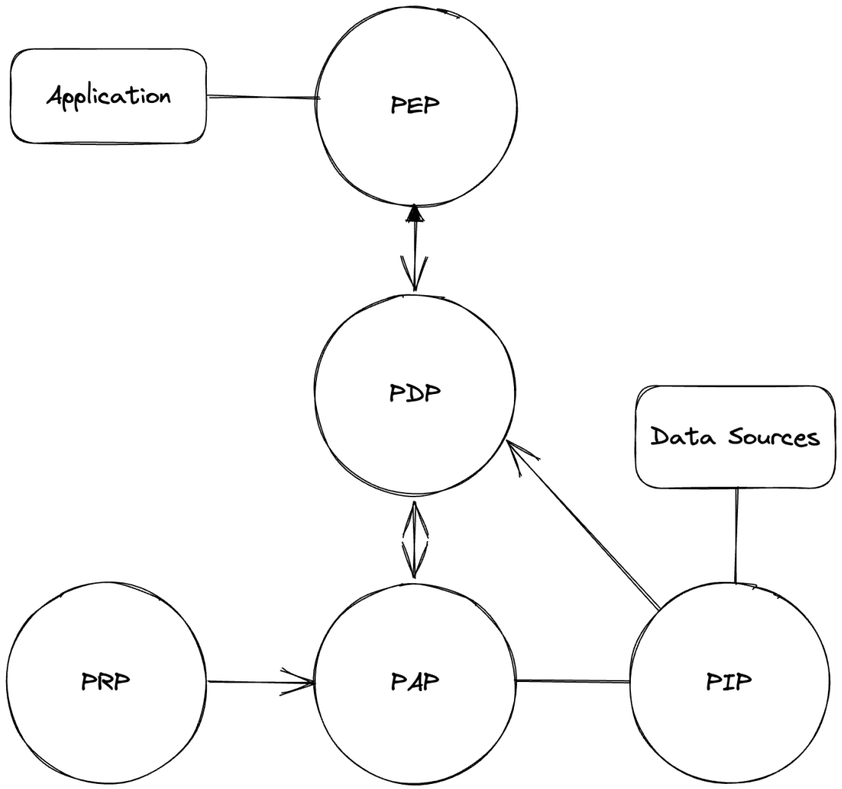
@gemanor

Google Zanzibar - OpenFGA

- Simple ReBAC
- Complex RBAC/ABAC
- Reverse indices support
- Backed by OKTA
- Stateful

Next Station: Architecture

The Policy System Building Blocks

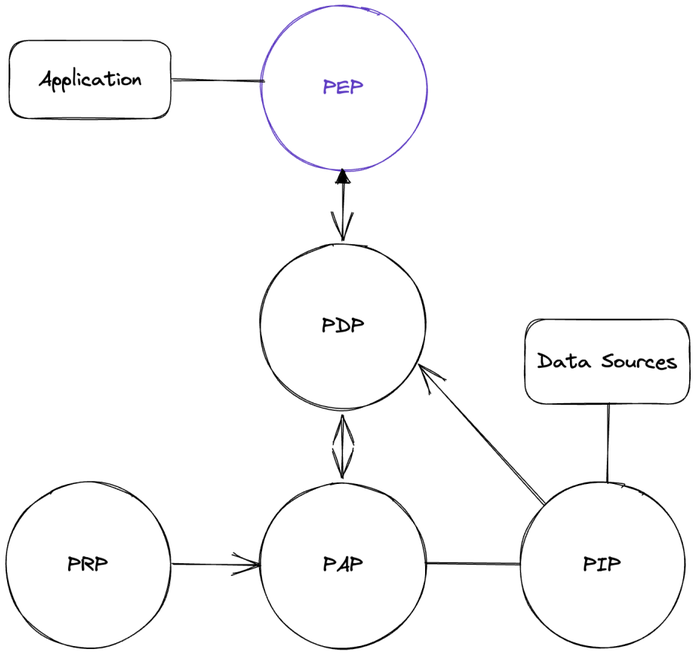


- Policy Enforcement Point
- Policy Decision Point
- Policy Retrieval Point
- Policy Information Point
- Policy Administration Point

Permit.io

@gemanor

PEP - Policy Enforcement Point

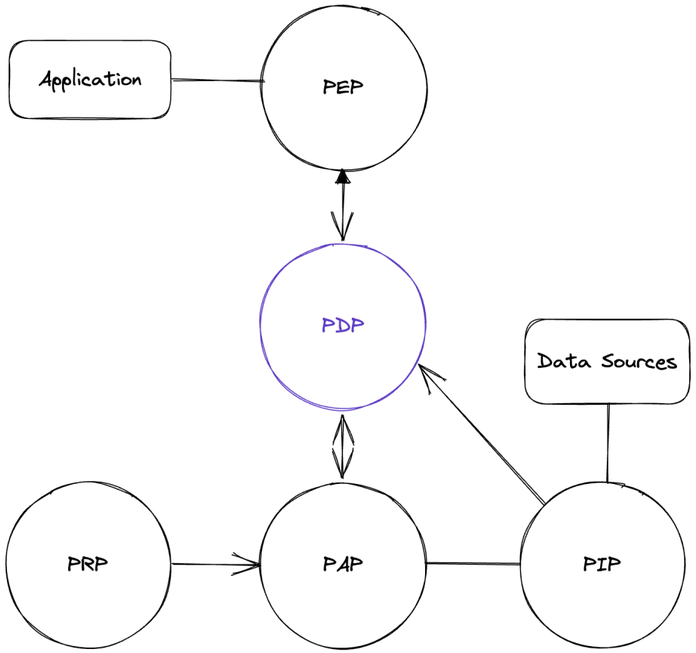


- Enforce policy decisions in the application
- An abstract representation of `check` function `check(user, action, resource)`
- Policy contract agnostic
- Could be in application code or as a plugin
- In ReBAC should include search/list allowed object endpoint

Permit.io

@gemanor

PDP - Policy Desicion Point

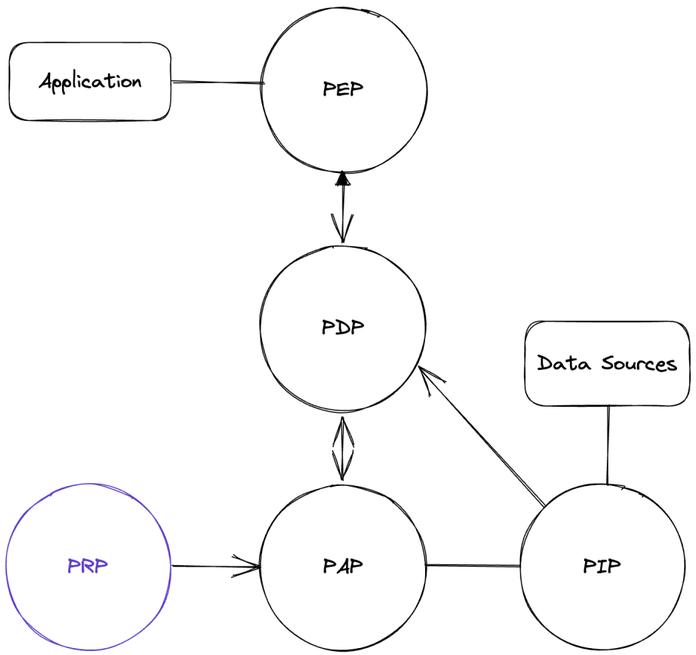


- Decision engine
- Stateless and cache enabled
- Serve request in <10ms
- Should run as a sidecar to the application

Permit.io

@gemanor

PRP - Policy Retrieval Point

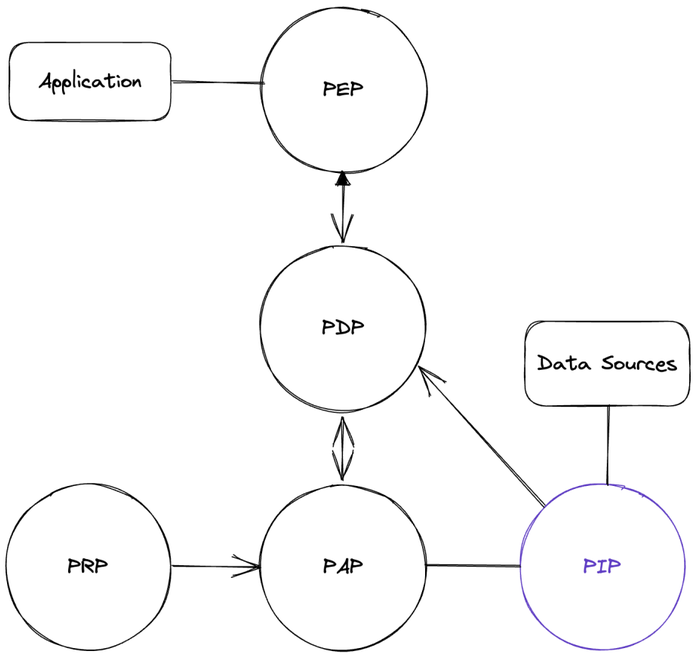


- Policy config storage
- GitOps enabled git repository
- Use IaC principles
- Branches represents environments

Permit.io

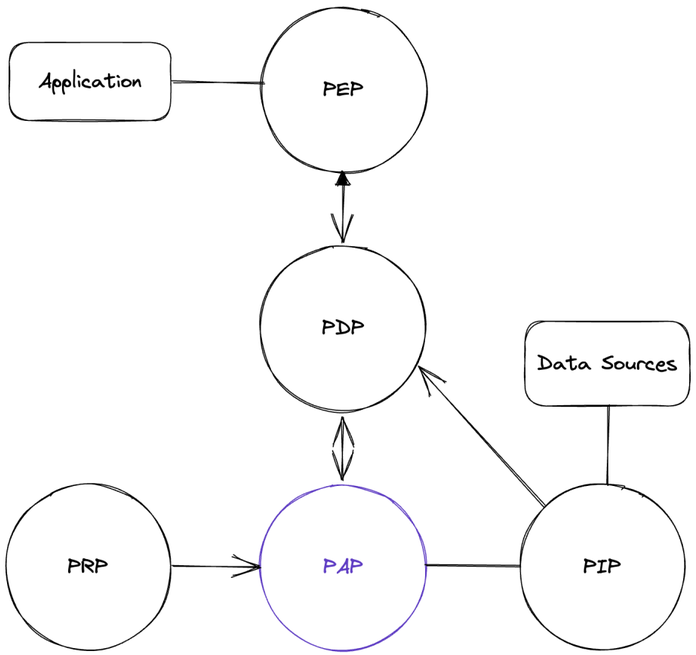
@gemanor

PIP - Policy Informantion Point

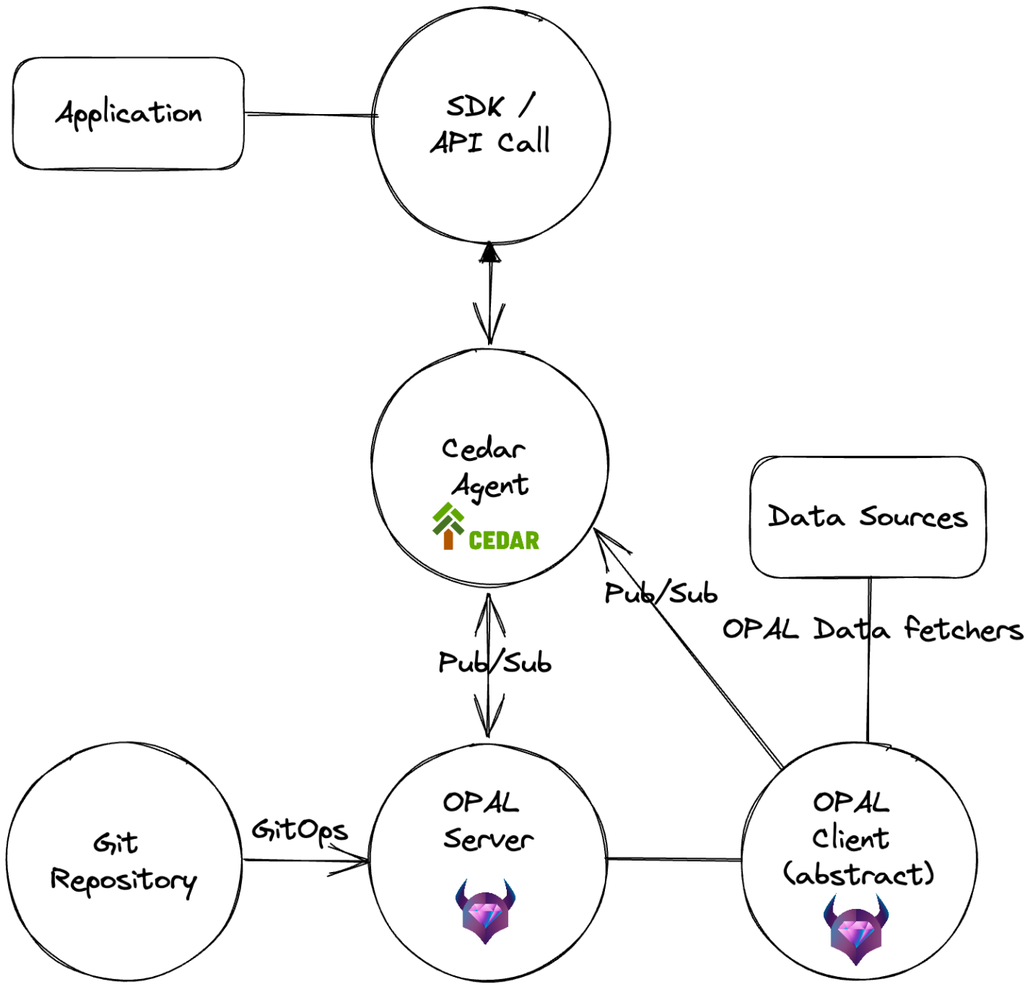


- Every kind of data to enrich the policy desicion making
- Traditionally - IDM
- Currently - data fetchers
- Pub/Sub with PDP

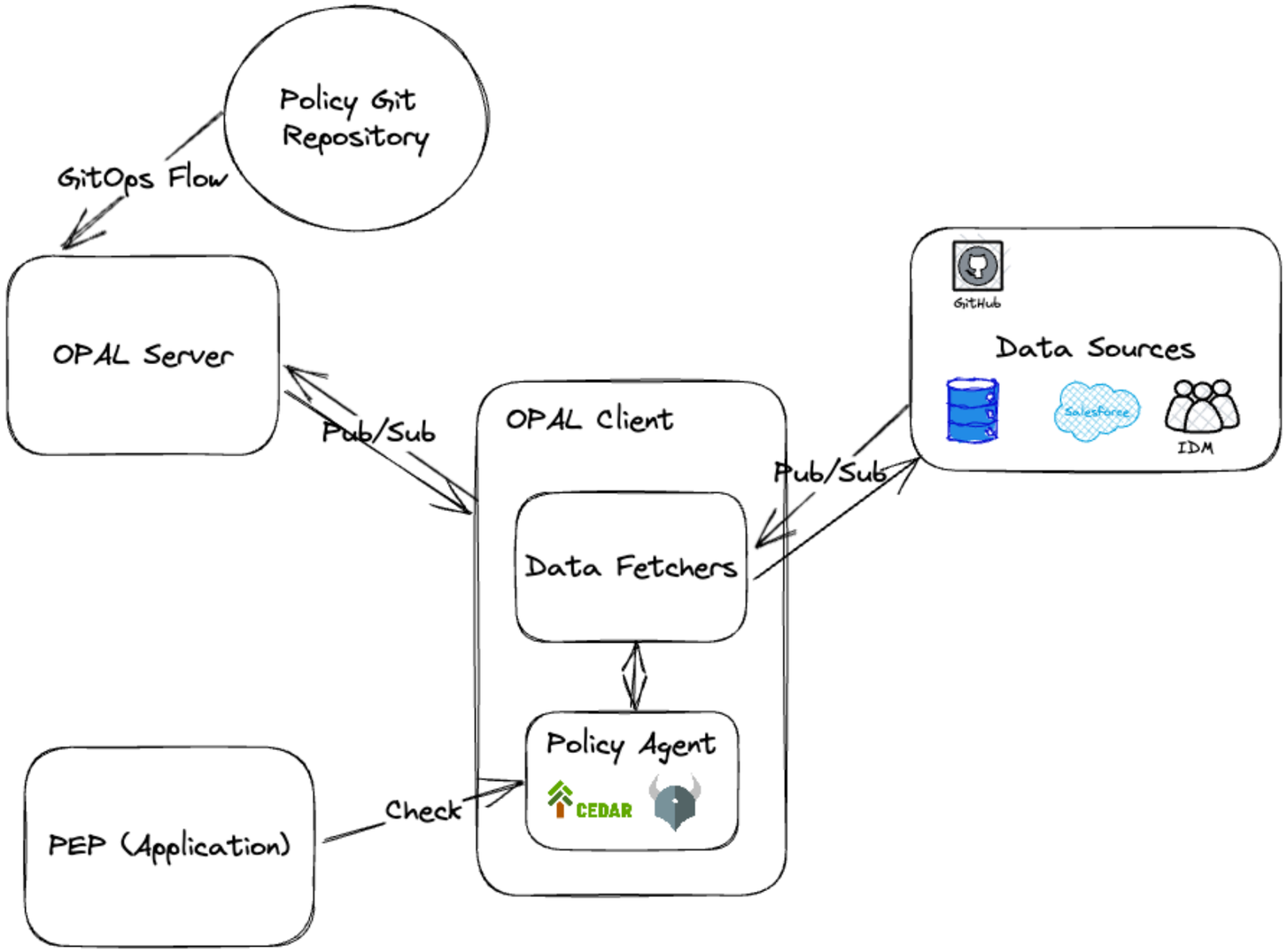
PAP - Policy Administration Point



- Keep the PDP/PIP/PRP synced
- The PDP "backend"
- Make sure all the PDP data is up to date
- System configuration one-stop-shop
- Scale PDPs



- PEP - Call the PDP APIs (HTTP/REST)
- PDP - Cedar Agent `https://github.com/permit-policy` `https://github.com/permitio/cedar-agent`
- PAP - OPAL server `https://github.com/permitio/opal`
- PRP - Git Repository
- PIP - OPAL Client data fetchers



Permit.io

@gemanor

PEP Deployment Strategies

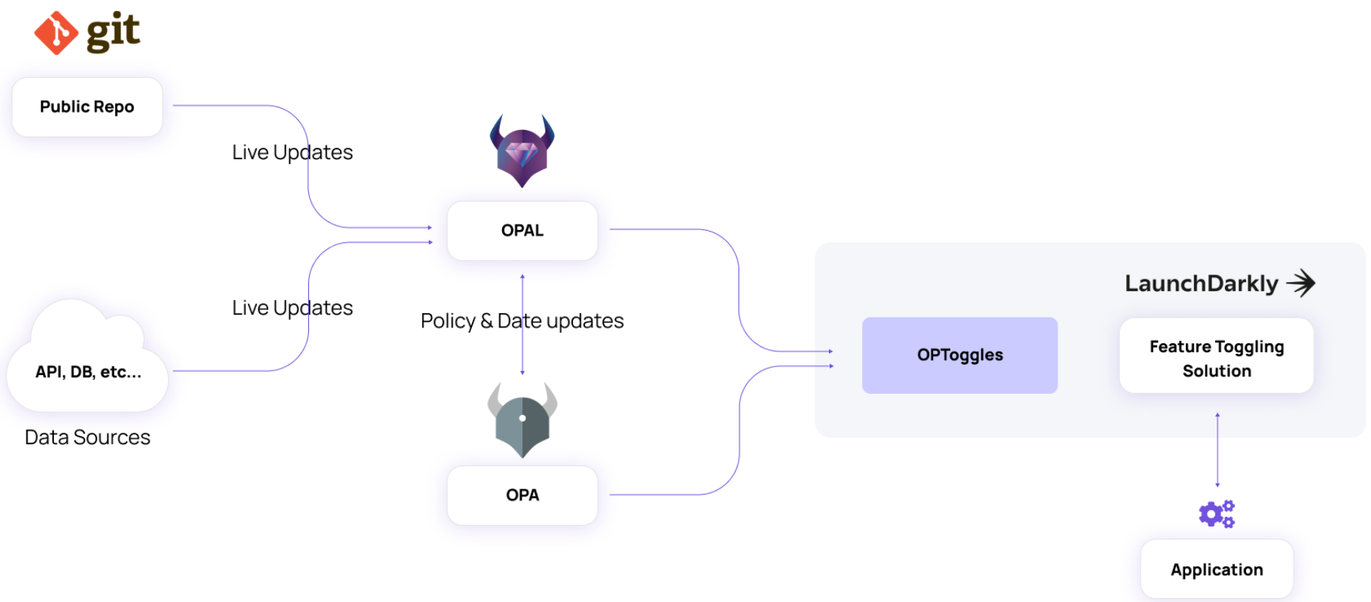
- Application level
  - HTTP API calls | Middlewares | SDKs | API Gateway plugins
- Persistence Layer
  - DB access plugins
- Adminssions
  - K8s controllers | TF plugins

CASL - The Frontend PEP SDK

```
import { createMongoAbility, AbilityBuilder } from '@casl/ability';

// define abilities
const { can, cannot, build } = new AbilityBuilder(createMongoAbility);

can('read', ['Post', 'Comment']);
can('manage', 'Post', { author: 'me' });
can('create', 'Comment');

// check abilities
const ability = build();

ability.can('read', 'Post') // true
```

Permit.io

@gemanor



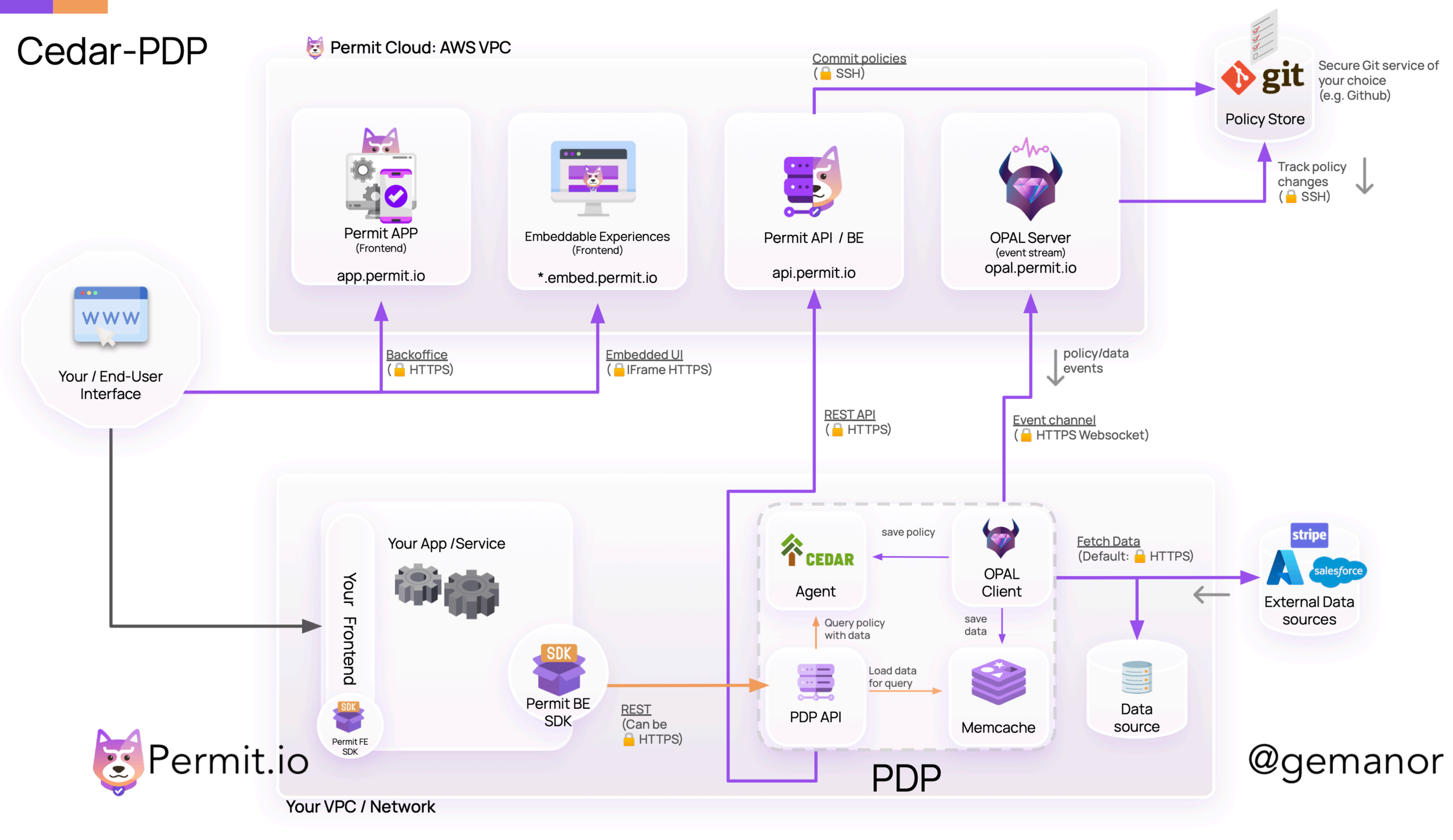
https://github.com/permitio/optoggles

Permit.io

@gemanor

Demo time 🍿

Cedar-PDP


Permit Cloud: AWS VPC
Commit policies
(SSH)
Secure Git service of your choice (e.g. Github)
Policy Store
Track policy changes
(SSH)
Permit APP
(Frontend)
app.permit.io
Embeddable Experiences
(Frontend)
*.embed.permit.io
Permit API / BE
api.permit.io
OPAL Server
(event stream)
opal.permit.io
Your / End-User Interface
Backoffice
(HTTPS)
Embedded UI
(IFrame HTTPS)
policy/data events
REST API
(HTTPS)
Event channel
(HTTPS Websocket)
Your App /Service
Your Frontend
SDK
Permit FE SDK
SDK
Permit BE SDK
REST
(Can be HTTPS)
CEDAR
Agent
save policy
OPAL Client
Fetch Data
(Default: HTTPS)
stripe
salesforce
External Data sources
Query policy with data
save data
PDP API
Load data for query
Memcache
Data source
PDP
Permit.io
Your VPC / Network
@gemanor

Thank You 🙏

Show your love to OPAL with a GitHub Star ⭐ 👉

Find more about OPAL on opal.ac

Follow me on Twitter @gemanor

Permit.io

@gemanor