
KubeCon
CloudNativeCon
North America 2024

AuthZEN: the “OIDC” of Authorization

Omri Gazitt

Aserto Co-founder & CEO

OpenID AuthZEN Co-chair

About Me

Dev Tech, Cloud, IAM, startups, skiing

[@omrig](#)

omri@aserto.com

Authentication != Authorization

Authentication

Did the user prove they are who they say they are?

Standards:

Developer services:

Authorization

What can the user do in the context of this app?

Standards: ?

Developer services: ?

Problems:

- Bad security[1]
- Inconsistency
- Opportunity cost

Broken Access Control: #1 !!

(1) https://owasp.org/Top10/

Authorization is finally having its moment…

KubeCon CloudNativeCon North America 2024

Zanzibar[1]

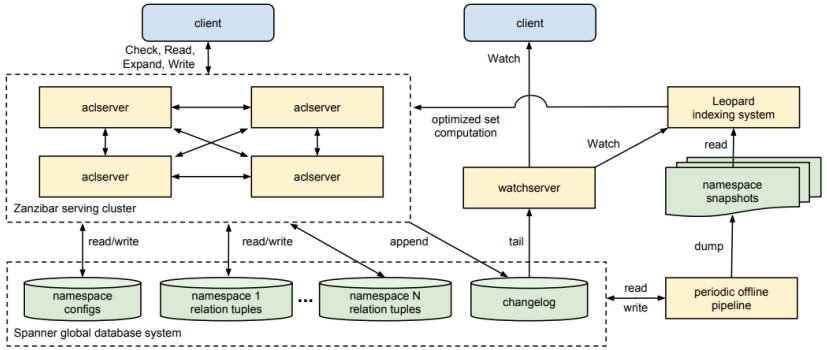
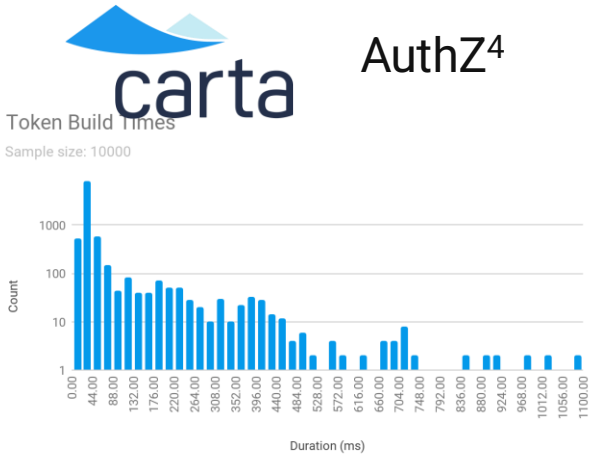
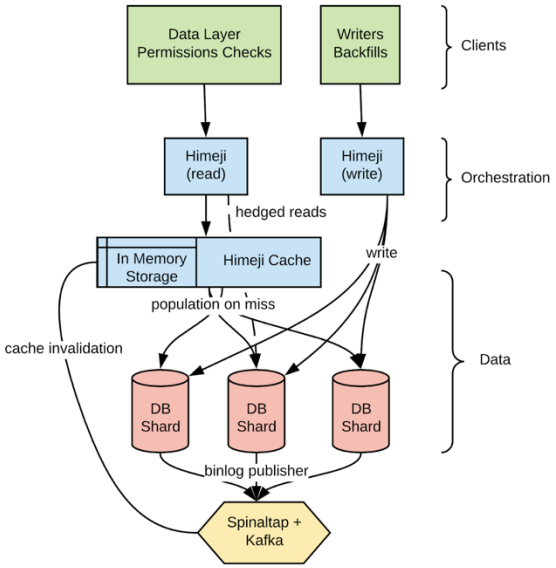


Figure 2: Zanzibar architecture. Arrows indicate the direction of data flow.

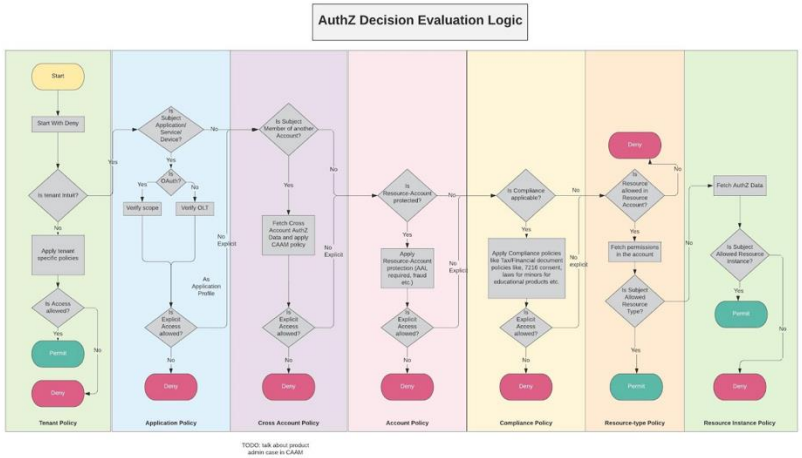
AuthZ[4]



Himeji[3]



AuthZ[2]



Product Access Service[5]

TRACK
Security: Establishing & Maintaining Customer Trust

SESSION
Authorization at Netflix Scale

Travis Nelson
Senior Software Engineer @Netflix

QCon plus NOVEMBER 01 - 12, 2021 PLUS.QCONFERENCES.COM

(1) https://research.google/pubs/pub48190/
(2) https://medium.com/intuit-engineering/authz-intuits-unified-dynamic-authorization-system-bea554d18f91
(3) https://medium.com/airbnb-engineering/himeji-a-scalable-centralized-system-for-authorization-at-airbnb-341664924574
(4) https://medium.com/building-carta/authz-cartas-highly-scalable-permissions-system-782a7f2c840f
(5) https://www.infoq.com/presentations/authorization-scalability/

Cloud-native authorization changes the game

	"Old-school"	Cloud-native	
WHAT	Coarse-grained, tenant-level permissions	**Fine-grained**: resource-level permissions	… Principle of least privilege
HOW	Authorization "spaghetti logic" embedded in the application	**Policy-based**: authorization logic extracted out of the application	… Separation of duties
WHEN	Permissions evaluated at login time, scopes embedded in access token	**Real-time**: permissions evaluated before granting access to resource	… Continuous enforcement

Fine-grained access control evolution

Access Control Lists: 1980's-1990's (UNIX, NT CACL)

ACL

Does Alice have read access to this file?

Role-based Access Control: 1990's-2000's (LDAP, AD)

RBAC

Is Bob in the sales-admin role?

Attribute-based Access Control: 2000's-2010's (XACML)

ABAC

Is Mallory in the sales department, is the document in the sales folder, and is it currently working hours in the US?

Relationship-based Access Control – 2020 (Zanzibar)

ReBAC

Does Eve have read access to this document if she's in the sales group, the document is in the sales folder, and the sales group is in the "editor" relation on the sales folder?

Policy-based access management

Lift access control logic out of the application and into its own policy-as-code artifact

Policy written in Rego (Open Policy Agent / Topaz)

post.rego

src > policies > __id > post.rego

```
1   package peoplefinder.POST.api.users.__id
2
3   default allowed = false
4
5   default visible = true
6
7   default enabled = true
8
9   allowed {
10    props = input.user.attributes.properties
11    props.department == "Operations"
12  }
13
14  allowed {
15    dir.is_manager_of(input.user.id, input.resource.id)
16  }
17
18  enabled {
19    allowed
20  }
21
```

Application code uses middleware to call authz service

```
// use checkAuthz as middleware in the route dispatch path
app.get("/api/users", checkJwt, checkAuthz, async (req, res) => {
  const users = await directory.getUsers(req);
  if (users) {
    res.status(200).send(users);
  } else {
    res.status(403).send('something went wrong');
  }
});
```

Store and version policy just like application code

Every policy change is part of a git changelog

Policy can be evolved by security team, decoupled from app

Policy can be built into an immutable image and signed (https://github.com/opcr-io/policy)

Real-time access checks

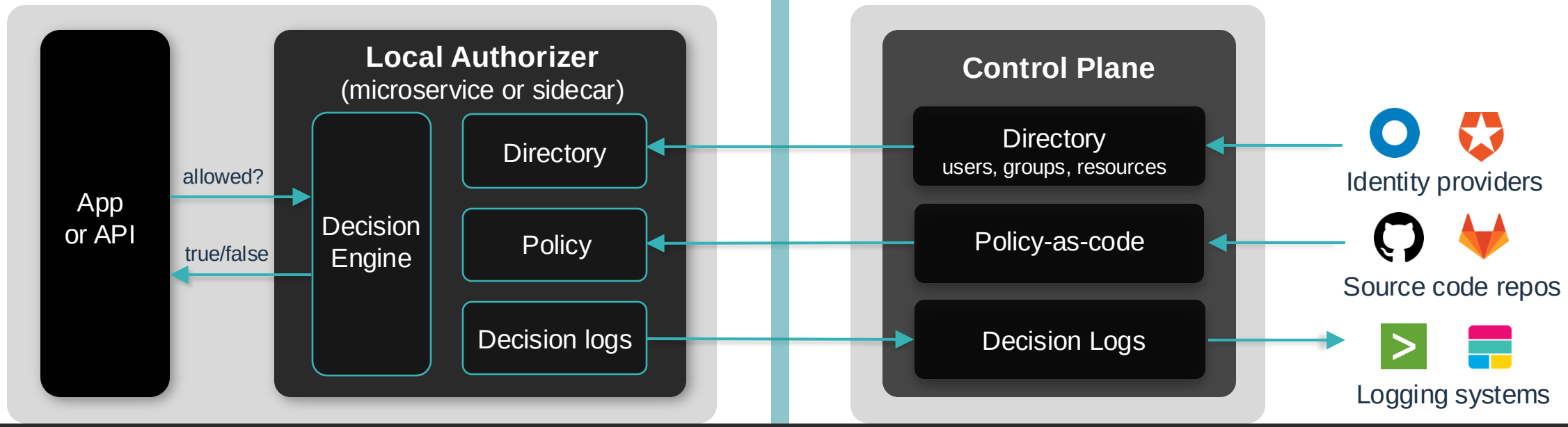
Done correctly, authorization is a distributed systems problem

Authorize locally

- Authorization is in the critical path of every application request
- Requires 100% availability at milliseconds of latency
- Must be deployed right next to the application / microservice
- Compute decision using cached policy, user, and resource data

Manage centrally

- Control plane manages policies, user directory, resource data
- Decision log collection and aggregation
- High-speed event and data fabric between control plane and edge



Cloud-native authorization: ecosystems, standards & OSS

Policy as code (ABAC)

Policy as data (ReBAC)

XACML
2001

ALFA
2014

AuthZEN
2024

Zanzibar
2020

NGAC
2016

Open Policy Agent

Casbin

TOPAZ

OpenFGA™

spicedb

cerbos

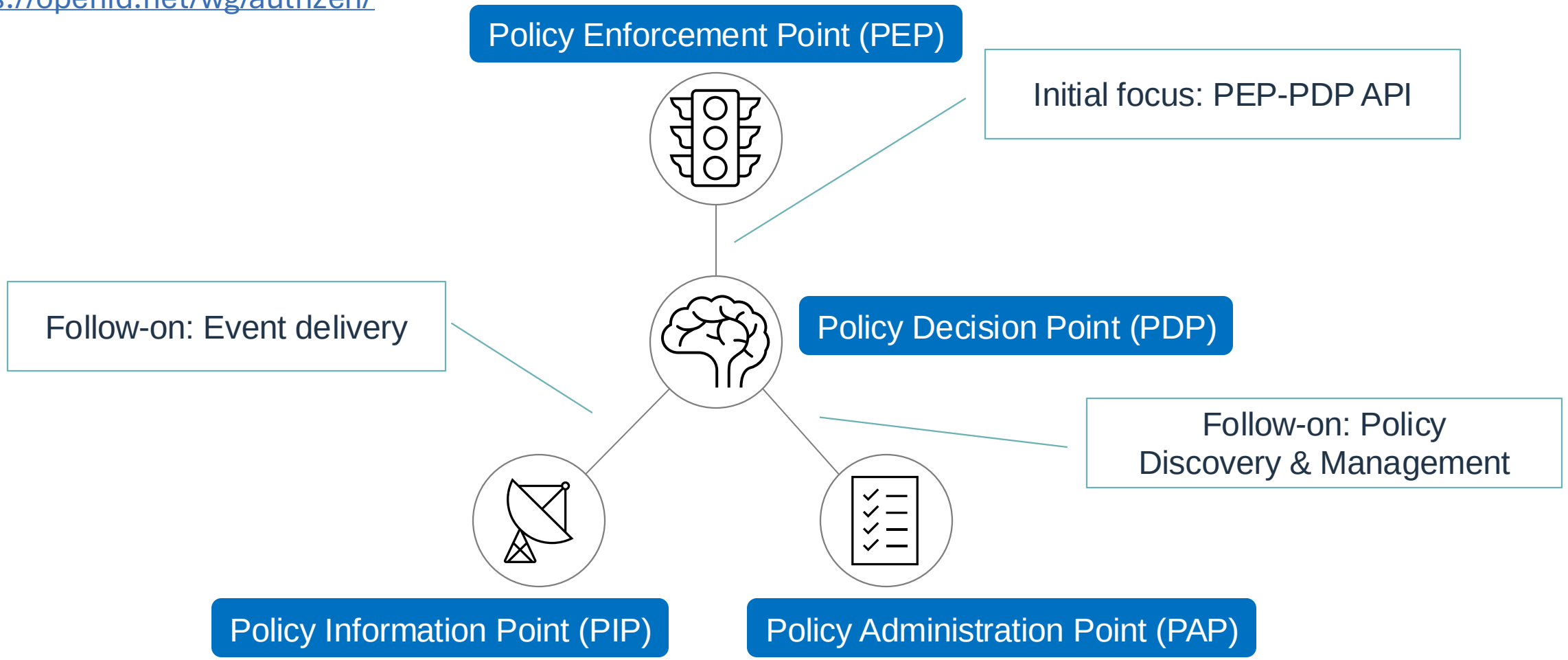
CEDAR

ORY

OSO

The AuthZEN Charter

https://openid.net/wg/authzen/



AuthZEN 1.0 PEP-PDP Implementer's Draft (ratified today!)

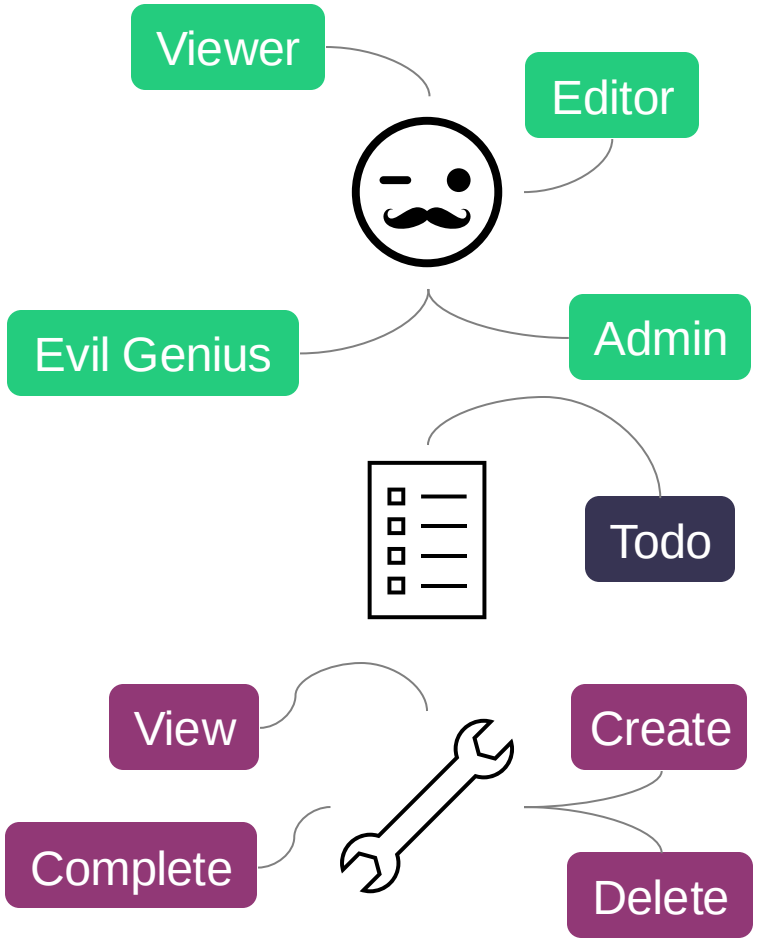
https://openid.github.io/authzen/

```
{
  "subject": {
    "type": "user",
    "id": "CiRm…2Fs"
  },
  "action": {
    "name": "can_delete_todo"
  },
  "resource": {
    "type": "todo",
    "id": "1"
    "properties": {
      "ownerID": "beth@the-smiths.com"
    }
  }
}
```

```
{
  "decision": true
}
```

First interop use-case: Todo application

https://authzen-interop.net/docs/scenarios/todo



Todo Application

This document lists the request and response payloads for each of the API requests in the Todo interop scenario.

TIP

This is a copy of the payload document defined by the AuthZEN WG. The definitive document can be found here.

Overview of the scenario

The Todo application manages a shared todo list between a set of users.

There are 5 actions that the Todo application supports, each with a permission associated with it:

Action	Permission
View a user's information	`can_read_user`
View all Todos	`can_read_todos`
Create a Todo	`can_create_todo`
(Un)complete a Todo	`can_update_todo`
Delete a Todo	`can_delete_todo`

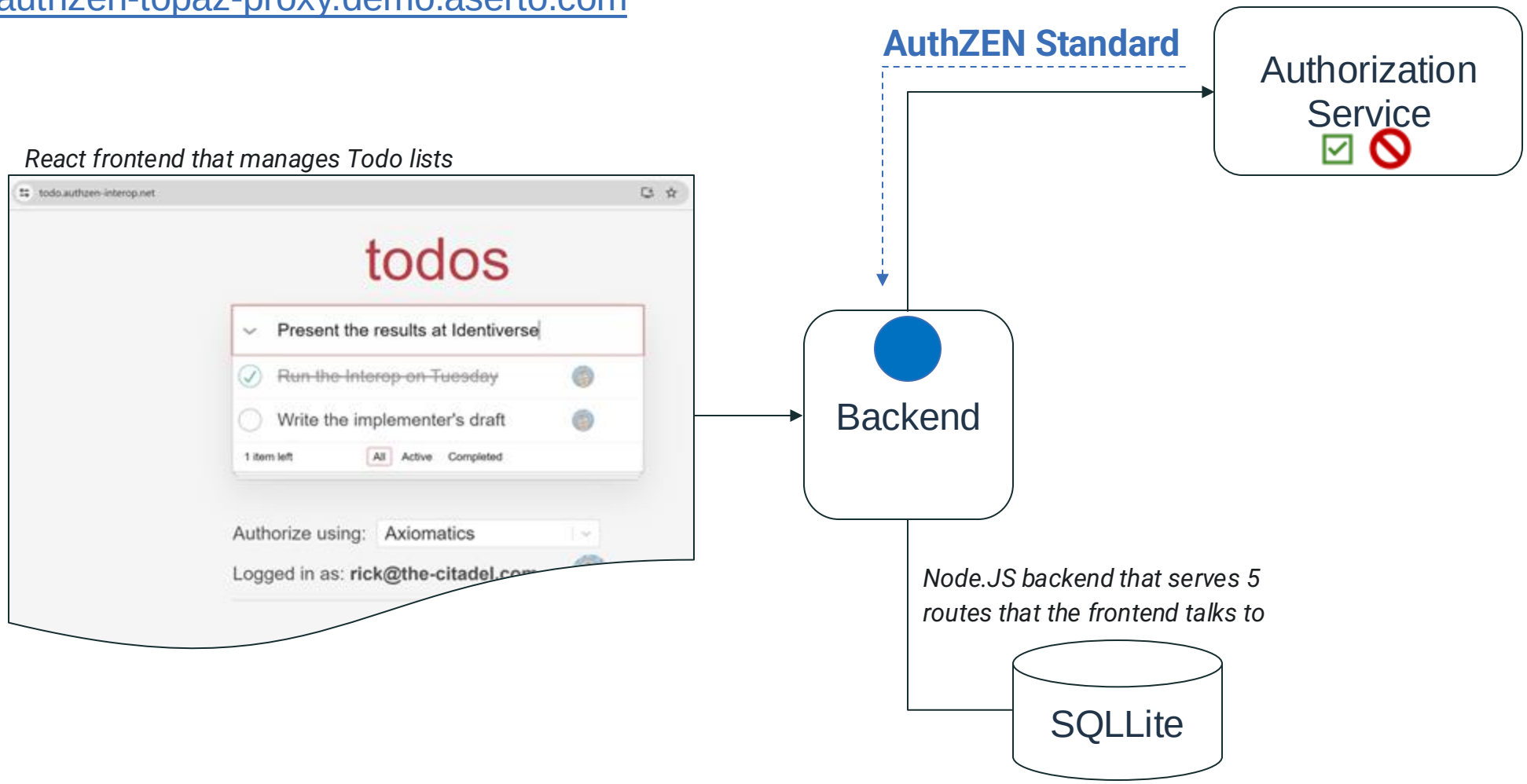
There are four roles defined:

- `viewer` - able to view the shared todo list (`can_read_` their picture) (`can_read_user`)
- `editor` - `viewer` + the ability to create new Todos (`c` *user*
- `admin` - `editor` + the ability to delete any Todos (`car`
- `evil_genius` - `editor` + the ability to edit Todos that

There are 5 users defined (based on the "Rick & Morty" cart

Interop architecture & Demo

https://authzen-topaz-proxy.demo.aserto.com



Interoperable implementations as of Nov 2024

Compliant with: 1.1 Preview, 1.0 Implementers Draft, 1.0 Preview

Aserto

AXIOMATICS

cerbos

INDY KITE

Open Policy Agent

Ping Identity

plainID

sgnl

STRATA

TOPAZ

Compliant with: 1.0 Preview

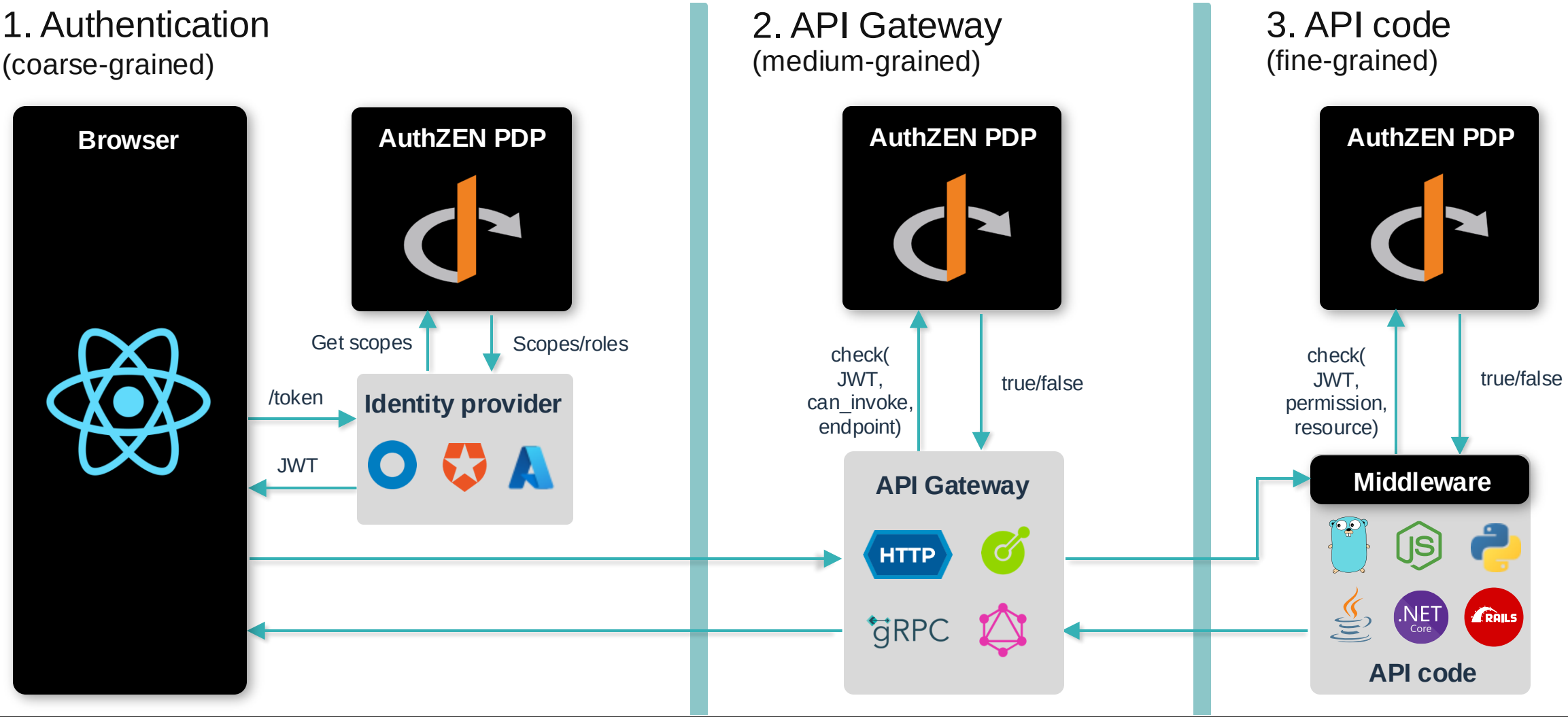
Permit.io

Rock Solid Knowledge

THALES

Enforcement points

Zero-trust means defense-in-depth



What's Next?

Resource Search API: find all the resources that a subject can access

Subject Search API: find all the subjects that can access a resource

Work with relying parties (Workday, SFDC, etc) to externalize authorization

Work with API Gateway vendors to wire AuthZEN into their products

Create additional interop scenarios

Add more implementations (especially ReBAC systems)

Pursue policy discovery/management and event delivery into PDP/PIP

Where to find us

Meeting notes & Design docs: https://hackmd.io/@oidf-wg-authzen

AuthZEN mailing list: https://openid.net/wg/authzen

GitHub: https://github.com/openid/authzen

OpenID Slack: #wg-authzen

Questions? Connect with me this week!

- Open Policy Containers: Project Pavilion 17B: Thu 1:30-5pm
- The Policy Engines Showdown: Fri 2pm

@omrig

omri@aserto.com

aserto.com/slack

aserto.com/contact

The principles of cloud-native authorization

Fine-grained	Support a consistent model (RBAC, ABAC, ReBAC) that fits the application domain
Policy-based	Extract policy out of the app and into its own repo, and build into a signed image
Real-time	Authorization is a local call, executing over fresh user / resource data
Centrally managed	Policy and directory/resource data are centrally managed
Compliance & forensics	Decision logs are aggregated and stored centrally
Developer-centric	Authorization with a single line of code
Integrates easily	Identity providers, source code repos, artifact registries, logging systems, API gateways
Cloud-native and open	Ecosystem effects of using k8s-native technologies like Open Policy Agent, Topaz, OCI