aws-auth-operator

Blending Okta and AWS federated users into Kubernetes access control I October 2020
AWS UserGroup Berlin 20.10.2020

TIER

Daniel & Daniel

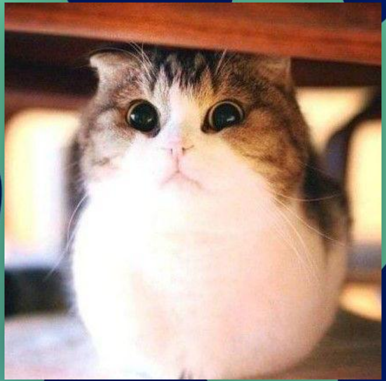
Daniel Ciaglia - Infrastructure Lead

#changemobilityforgood utilising the best infrastructure landscape and tooling
#aws #kubernetes #hashicorp

Daniel Hahn - Senior Backend/Devops Developer

Developing in startups for over 10 years

TIER



TIER as a company

We are here to **lead the way** towards **seamless and sustainable mobility**.

Together with public and private organizations, **we are rethinking urban transportation** and helping **create** a **clean, sustainable** and **better connected tomorrow** with cities **free from pollution and congestion**.

TIER in numbers

- **3** major compute environments
- **~80** nodes in production EKS
- PostgreSQL **Auroras**, **RabbitMQ**, some **λ**, **Redshift**...
- **60+** Developers

- **60k** vehicle (scooters + mopeds)
- **30+ Million** rides
- **80** Cities, **10** countries
- **900+** employees

Infrastructure as Code

No exceptions!

TIER

Policy: no manual creation or change of AWS or Kubernetes resources.

Never! Nowhere!

Solution:

- Global use of **terraform**
- **One** infrastructure repository **per team**
- **Terraform as CI** with final approval of administrative team*
- **Sandbox** account with **Administrator permissions** for **everybody****

*Bottleneck, subject to change
**Cleanup of sandbox is a different topic

Show me your source

Team infrastructure project

```
<team-infra-repository>

├── production/
│   ├── aws.tf
│   ├── okta.tf
│   ├── kubernetes.tf
│   ├── vault.tf
│   ├── example-service-foo/
│   │   ├── eu-central-1
│   │   │   └── main.tf
│   │   ├── eu-west-3
│   │   └── main.tf
│   ├── example-service-bar/
│   └── main.tf
│
├── staging/
│   ├── example-service-foo
│   └── example-service-bar
│
└── sandbox/
```

```
<production/main.tf>

module "example-service-foo" {
  source      = "./example-service-foo"
  environment = "production"
  team        = "ring-0"
  service     = "example-service-foo"
  region      = "global"
}
```

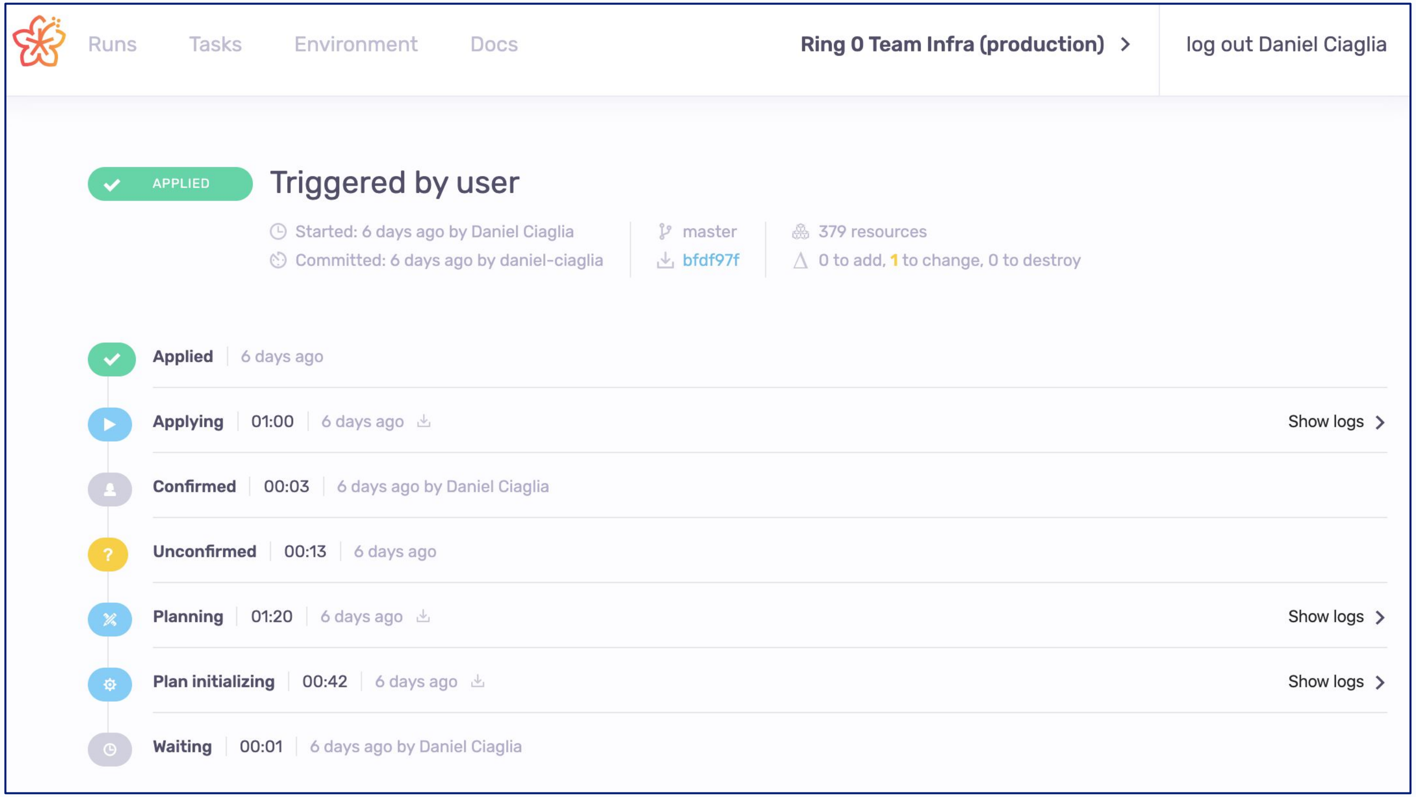
```
<production/example-service-foo/main.tf>

module "example-service-foo-eu-central-1" {
  source      = "./eu-central-1"
  service     = var.service
  region      = "eu-central-1"
  team        = var.team
  environment = var.environment
}
```

TIER

Compile the Infrastructure

Terraform CI



Triggered by user

Started: 6 days ago by Daniel Ciaglia
Committed: 6 days ago by daniel-ciaglia
master
bfdf97f
379 resources
0 to add, 1 to change, 0 to destroy

Applied | 6 days ago

Applying | 01:00 | 6 days ago — Show logs

Confirmed | 00:03 | 6 days ago by Daniel Ciaglia

Unconfirmed | 00:13 | 6 days ago

Planning | 01:20 | 6 days ago — Show logs

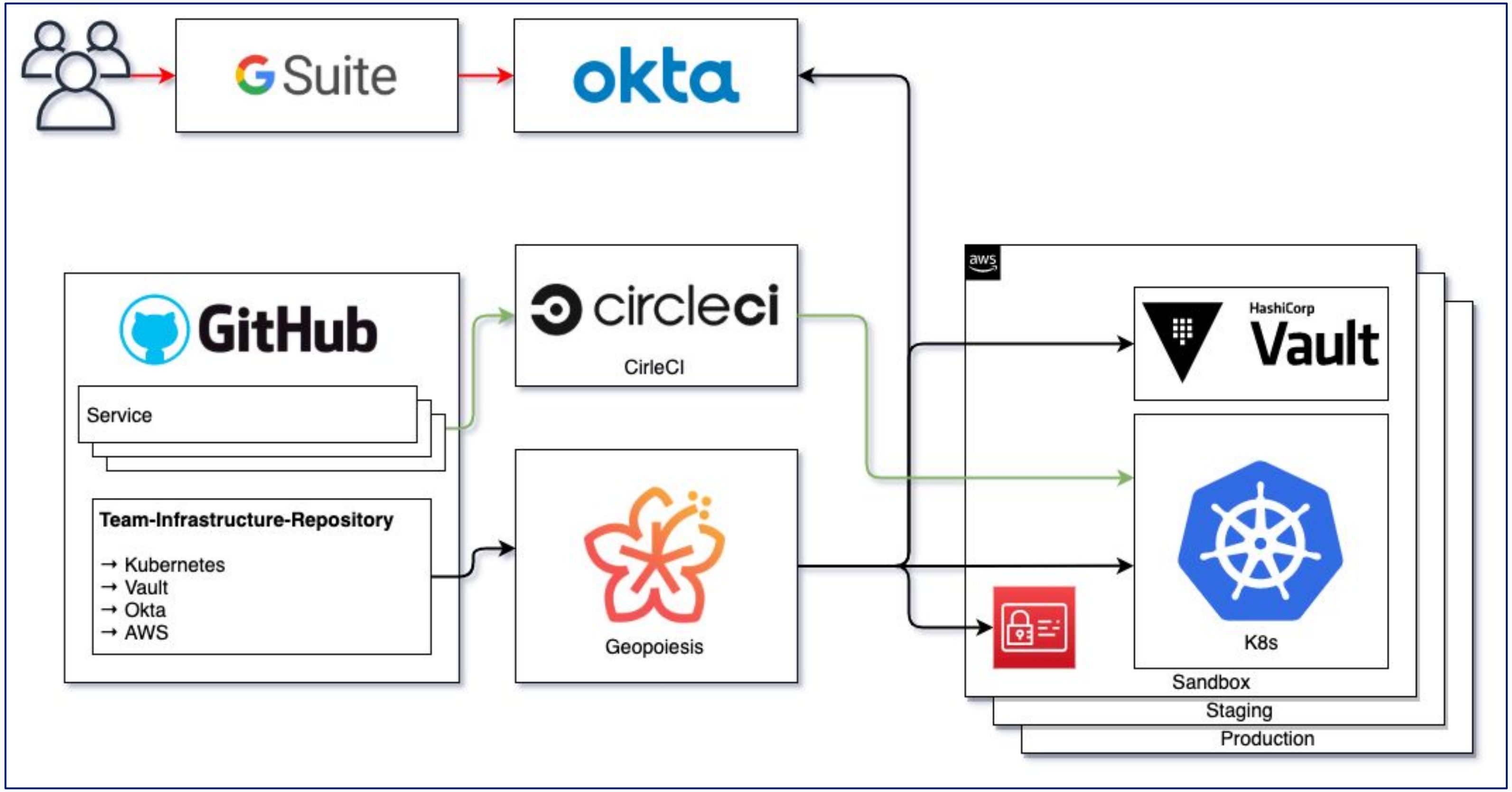
Plan initializing | 00:42 | 6 days ago — Show logs

Waiting | 00:01 | 6 days ago by Daniel Ciaglia

TIER

The moving parts

TIER tooling

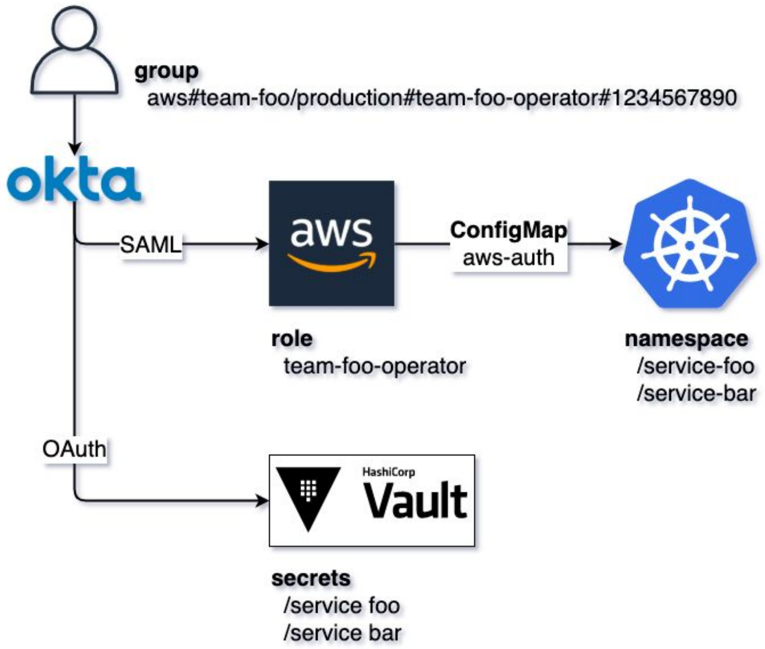


Authn|Authz

Mapping humans to kubernetes

A user is able to

- Log in to Okta and has **groups assigned**
- Access **Vault** (via WebUI or CLI) and has access to certain secrets
- Assume a role in **AWS** (via WebUI or eg. **aws-okta** CLI)
- Access **EKS** via **kubectl** and has permissions on certain namespaces



TIER

Welcome to the show

k get cm -n kube-system aws-auth -o yaml

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.

apiVersion: v1
Data:
  mapRoles: |
    - rolearn: arn:aws:iam::555555555555:role/devel-worker-nodes-NodeInstanceRole-74RF4UBDUKL6
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
  mapUsers: |
    - userarn: arn:aws:iam::555555555555:user/admin
      username: admin
      groups:
        - system:masters
    - userarn: arn:aws:iam::111122223333:user/ops-user
      username: ops-user
      groups:
        - system:masters
```

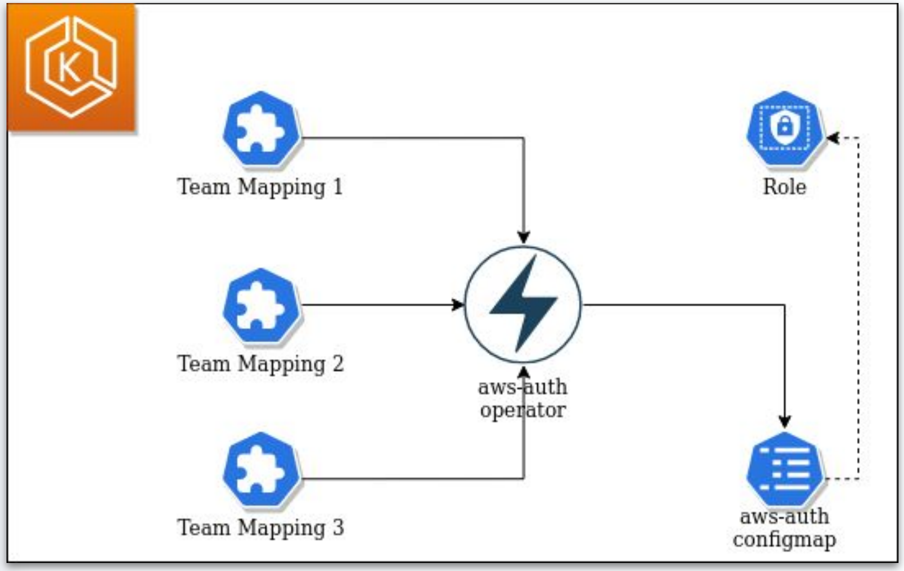
The problem

Central config map -- read by one, touched by many

- configured by terraform eks module
- Manual change error prone (type errors)
- No log of changes
- Access rights only visible from within k8s
- Only admins can add group rights
- No validation

TIER

The solution



aws-auth-operator

- Central management entity for configmap
- Decentralized configuration
- Monitored process
- Fragments in custom resources
- Operator based on kopf framework

TIER

Pitfalls and failsafes and best practices

- **Who adds what**
  solved through Geopoiesis for now
- **Adding CRDs via terraform**
  there's an alpha solution for that
- Protected mapping to **save initial configuration**

What happened when (audit)

- Log of all changes to stdout
- Configuration in code
- Possibly more integrations through kubernetes events

TIER

Reality

k get awsm search-and-ride -o yaml

```
apiVersion: tier.app/v1
kind: AwsAuthMapping

metadata:
  name: search-and-ride

spec:
  mappings:
  - groups:
    - search-and-ride-viewers
    - search-and-ride-editors
    rolearn: arn:aws:iam::0123456789:role/team-search-and-ride-operator
    username: team-search-and-ride-operator
```

TIER

Module

Integration in terraform

```
# Custom Module to apply CRD
# new (untested by us) provider - https://registry.terraform.io/providers/hashicorp/kubernetes-alpha/

module "aws-auth" {
  source                 = "terraform.tier-services.io/tier/aws-auth/kubernetes"
  version                = "~> 1.0"
  clustername            = module.k8s-data.eks-cluster-id
  region                 = "eu-central-1"
  cluster_endpoint       = module.k8s-data.eks-cluster-endpoint
  cluster_ca_certificate = module.k8s-data.eks-ca-data-base64
  team                   = local.metadata.team
  map_roles = [
     {
       rolearn  = aws_iam_role.operator_role.arn
       username = local.auth.operator_role_name
       groups   = ["${local.metadata.team}-viewers", "${local.metadata.team}-editors"]
     },
     {
       rolearn  = aws_iam_role.spectator_role.arn
       username = local.auth.spectator_role_name
       groups   = ["${local.metadata.team}-viewers"]
     },
  ]
}
```

TIER

Rolebinding

Integration in terraform

```
# Cluster Role Binding to map group to actual ClusterRole/Role

resource "kubernetes_cluster_role_binding" "viewers" {
  metadata {
       name = "${local.metadata.team}-viewers"
  }
  role_ref {
       api_group = "rbac.authorization.k8s.io"
       kind      = "ClusterRole"
       name      = "tier-view"
  }
  subject {
       kind       = "Group"
       name       = "${local.metadata.team}-viewers"
       api_group = "rbac.authorization.k8s.io"
  }
}
```

TIER

Star me
Fork me

- TierMobility/aws-auth-operator
- TierMobility/aws-auth-operator/helm
- It has documentation

Last bits

Tooling involved

- **Okta** - https://www.okta.com/
- **Terraform** - https://www.terraform.io/
- **Geopoiesis** - https://spacelift.io/
- **AWS EKS** - https://aws.amazon.com/eks
- **Kopf operator framework** - https://github.com/zalando-incubator/kopf
- **AWS Auth** - https://docs.aws.amazon.com/eks/latest/userguide/add-user-role.html
- **AWS Okta CLI** - https://github.com/segmentio/aws-okta
- *Terraform k8s provider (alpha)* - https://registry.terraform.io/providers/hashicorp/kubernetes-alpha/

Summary

The **aws-auth-operator** re-constructs the central **aws-auth configuration** (configuring access from AWS world to kubernetes land) **based on individual fragments**, allowing a flexible setup on the ever-changing teams.

BE BOLD.

daniel.hahn@tier.app // https://www.linkedin.com/in/daniel-hahn-92351472
daniel.ciaglia@tier.app // https://www.linkedin.com/in/danielciaglia

What happened, when, how?

Example audit log

```
[2020-10-13 23:00:26,530] Kopf.objects [INFO] [kube-system/aws-auth] Change to aws-auth configmap:
 [("add",
    "mapRoles",
    [
      (30,
      {
        "username":"example-operator",
        "groups":[
          "example-team-infra-viewers",
          "example-team-infra-editors"
        ],
        "rolearn":"arn:aws:iam::0123456789:role/example-operator"
      }),
      (31,
      {
        "username":"example-spectator",
        "groups":[
          "example-team-infra-viewers"
        ],
        "rolearn":"arn:aws:iam::0123456789:role/example-spectator"
      })
    ])
  ]
```

TIER