



OIDC Web Identity for Ceph RGW

Temporary S3 credentials for users, pods, and cloud portals



Tadas Sutkaitis

DevOps Engineer, VEXXHOST, Inc.

[linkedin.com/in/tadasas](https://www.linkedin.com/in/tadasas)

Ceph Tech Talks • October 2026

Four live proofs

01

Terraform / OpenTofu

Provision the OIDC provider, role, policies, and demo bucket as code.

02

radosgw-assume

Use the role from the CLI with temporary credentials.

03

Pod identity

Obtain RGW credentials from a projected Kubernetes identity token.

04

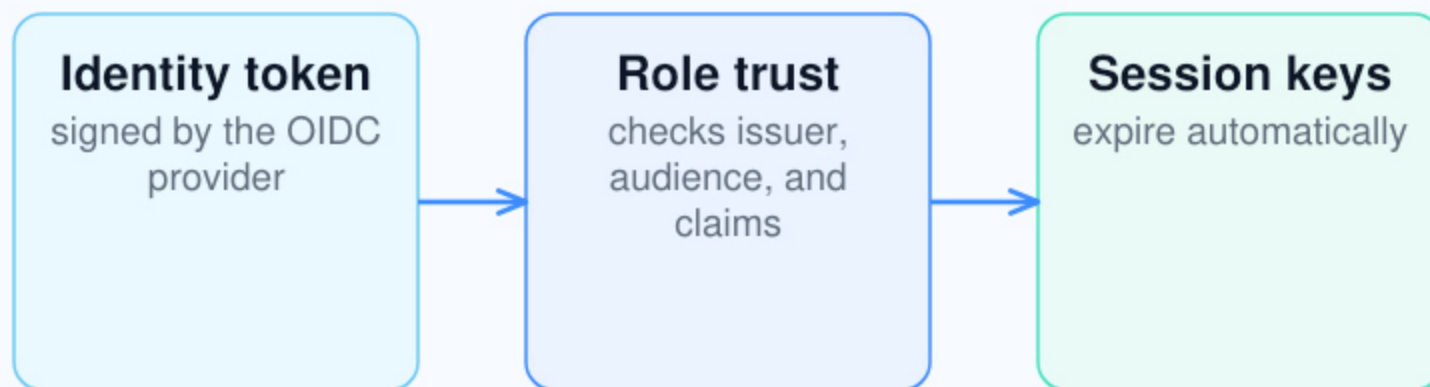
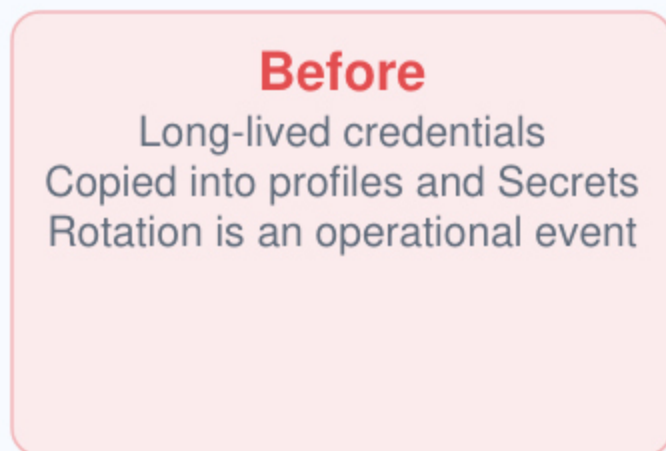
Sunrise

Turn one OpenStack login into project-aware S3 and IAM access.

One trust model. No long-lived S3 credentials.

What web identity changes

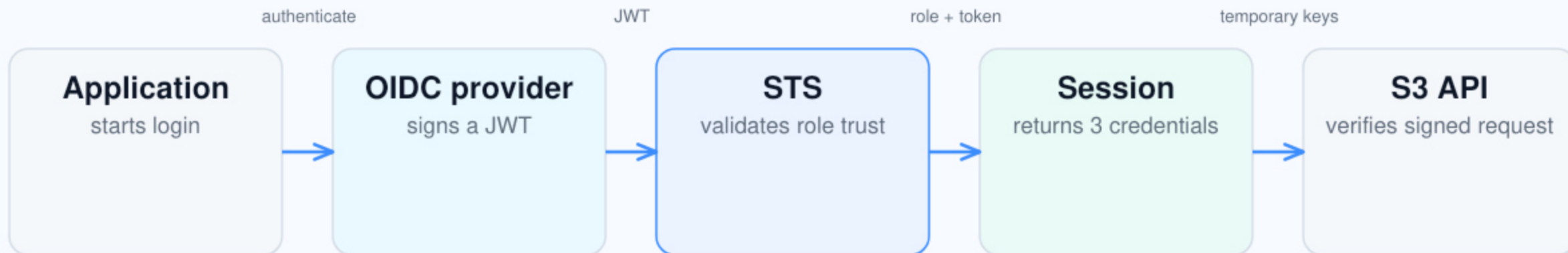
Use an OIDC token to receive temporary credentials for a trusted role.



Keycloak, Okta, GitHub, or another standards-compliant OIDC provider can supply the token.

No existing AWS-style access key is required for AssumeRoleWithWebIdentity.

AssumeRoleWithWebIdentity



Default AWS session: 1 hour • request 15 minutes up to the role maximum

S3 signing stays the same. Only the credential source changes.

Two policy boundaries

Trust policy

WHO may start a session

Principal: OIDC provider

Action: sts:AssumeRoleWithWebIdentity

Conditions: aud, sub, or other claims

Permission policy

WHAT the session may do

Actions: s3:GetObject and friends

Resources: selected buckets or prefixes

Conditions: optional context restrictions

Effective session = role permissions $\cap$ optional session policy

How Ceph RGW implements the pattern



Gateway settings

```
rgw_s3_auth_use_sts = true  
rgw_sts_key = <shared signing key>
```

Ceph-specific behavior

STS and S3 share the RGW endpoint.
A shadow user comes from the token subject.
Claim conditions and session tags can scope access.

The API exists. The workflow is missing

01

Useful API reference

The STS page lists operations, parameters, gateway settings, and small boto3 examples.

02

Thin production story

It does not connect OIDC setup, role lifecycle, rollout, import, drift, and rollback into one workflow.

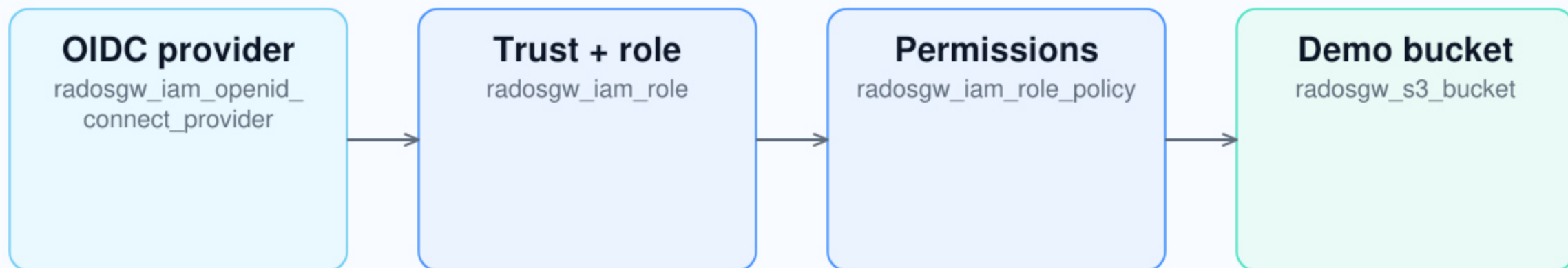
03

Few client patterns

There is little operational guidance for CLI profiles, desktop clients, pods, or web portals.

The API exists. Operators still have to assemble a repeatable deployment and client workflow.

Identity becomes reviewable infrastructure



```
resource "radosgw_s3_bucket" "demo" { bucket = "ceph-tech-talks-bucket01" }
```

Review • repeat • import • detect drift

<https://github.com/fitbeard/terraform-provider-radosgw>

Live demo 1: Terraform / OpenTofu

1

Plan

Confirm the OIDC provider, role, policies, and bucket.

2

Apply

Create the resources through RGW Admin and S3 APIs.

3

Inspect

Show role ARN, trust audience, and bucket name.

4

Set the boundary

The role may reach the demo bucket—nothing broader.

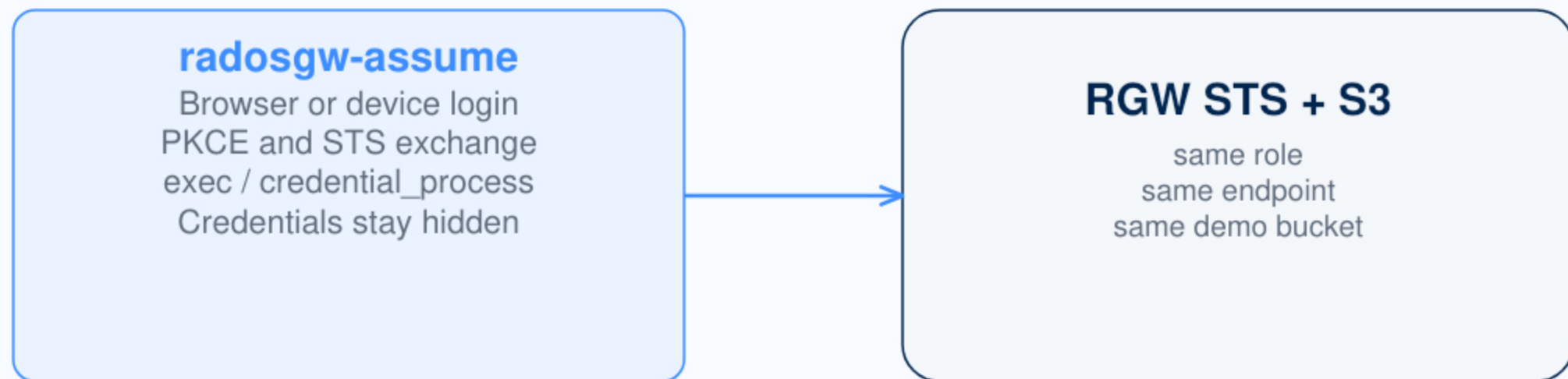
Terraform CLI

```
$ terraform plan  
$ terraform apply
```

OpenTofu CLI

```
$ tofu plan  
$ tofu apply
```

Temporary credentials for existing tools



Existing tools keep their workflow and use the same role permissions.

<https://github.com/fitbeard/radosgw-assume>

Live demo 2: radosgw-assume

01

Login

Let radosgw-assume open the OIDC flow.

02

CLI access

List the provisioned bucket and upload a small object.

03

Desktop access

Use Cyberduck with the same RGW STS role ARN.

04

Cross-check

See the CLI-uploaded object and download it.

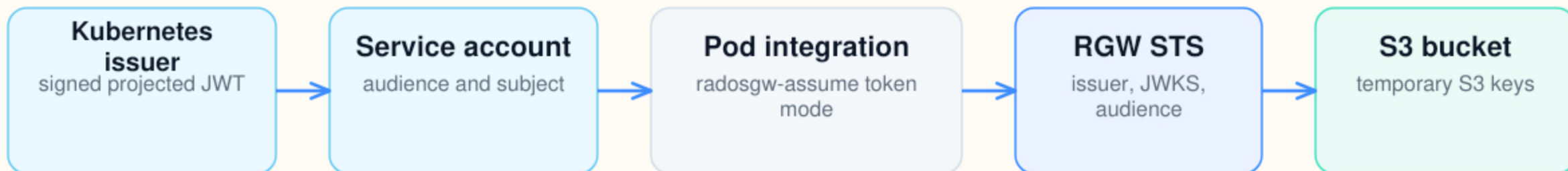
```
$ radosgw-assume exec -p demo -- aws s3 ls s3://ceph-tech-talks-bucket01
```

Show the session in use. Never expose the token or secret key.

github.com/fitbeard/radosgw-assume

Kubernetes as a trusted OIDC issuer

<https://github.com/aws/amazon-eks-pod-identity-webhook/pull/302>



Production cluster requirements

Stable HTTPS issuer

[service-account-issuer](#)

Sets the JWT iss claim and discovery base

Reachable public keys

[service-account-jwks-uri](#)

Discovery and JWKS reachable from RGW

Audience for RGW

[api-audiences](#)

Projected token audience matches role trust

Public: discovery metadata and JWKS public keys. **Private:** projected pod JWTs.

Live demo 3: pod identity

1

Verify the issuer

Fetch OIDC discovery and JWKS through the same network path used by RGW.

2

Inspect the workload

Show role ARN, token-file path, RGW endpoint, and only the iss, aud, and sub claims.

3

Run the S3 request

Access the provisioned bucket with temporary credentials and no static S3 Secret.

```
aws --endpoint-url $AWS_ENDPOINT_URL s3 ls s3://ceph-tech-talks-bucket01
```

Sunrise brings S3 and IAM into OpenStack

VEXXHOST's modern, open-source dashboard for project-aware cloud workflows.

<https://github.com/vexxhost/sunrise>

OpenStack context

Projects, regions, and the service catalog stay visible to the user.

Cloud workflows

Instances, images, networks, volumes, Kubernetes, and object storage.

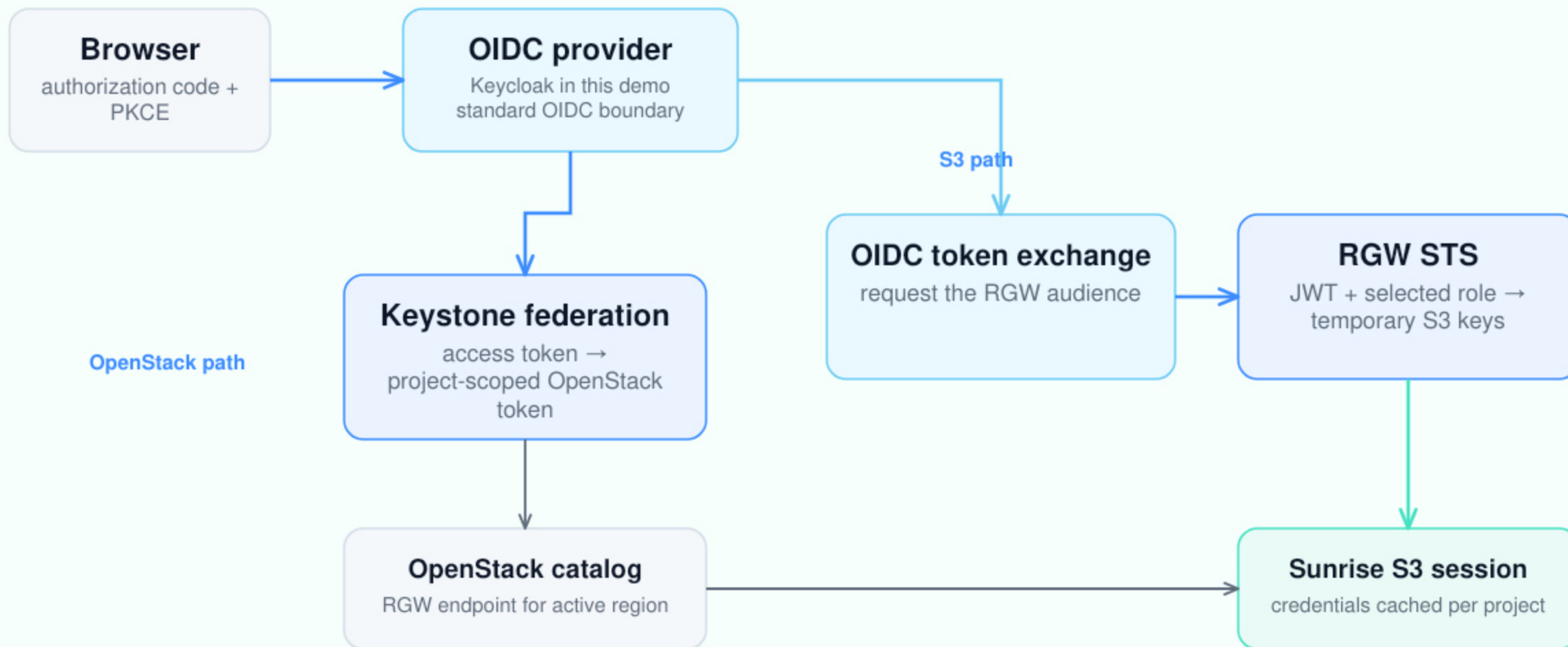
One browser login

OIDC authorization code with PKCE
Keystone federation behind it.

Implementation

Open source, built with Next.js and TypeScript.

One login, two cloud sessions



S3 + IAM in Sunrise

Buckets and objects

- List and create buckets
- Browse objects and prefixes
- Upload and download
- Use the active regional endpoint

IAM roles

- List project roles
- Inspect trust policy
- Inspect permission policies
- Select the project role for STS

Session boundary

- OIDC token exchange
- AssumeRoleWithWebIdentity
- Temporary credentials
- Refresh per active project

Project switch ⇒ Keystone scope + RGW role + S3 endpoint switch

Live demo 4: Sunrise

01

Sign in

Use the normal VEXXHOST
OIDC login.

02

Select context

Choose an OpenStack
project and region.

03

Inspect IAM

Open the RGW role, trust
policy, and permissions.

04

Use S3

Browse the provisioned
bucket and open the demo
object.

Success condition

One login. Project-aware OpenStack and S3 access. No exposed S3 credentials.

Temporary S3 credentials can be a platform primitive

- 1 Provision OIDC providers, roles, policies, and buckets as reviewed infrastructure.
- 2 Give workstation tools and Kubernetes pods the same short-lived credential path.
- 3 Make S3 and IAM feel native inside the user's OpenStack project context.

github.com/fitbeard/terraform-provider-radosgw • github.com/fitbeard/radosgw-assume • github.com/vexxhost/sunrise

Questions?