POSIX access to remote storage with OIDC AuthN/AuthZ

Federico Fornari (federico.fornari@cnaf.infn.it)

Ahmad Alkhansa (ahmad.alkhansa@cnaf.infn.it)

Introduction

- Several emerging use cases of experiments needing local POSIX access to storage provided by CNAF:
  - Test-stand TEX for Eupraxia asks for 50 TB/year to archive data that must be accessed rather frequently, so must be stored on disk.
    - The collaboration needs to access data via POSIX in read-write mode from Frascati through a software named *archiver*.
    - Only a single UNIX user is reading and writing data on disk.
  - NEWSDM: "Is it possible to access our storage area without worrying about token renewal, for example with Rclone?".
  - CMS wants to access cloud storage resources in a POSIX-like way. Multiple solutions are available (Ceph, S3, CVMFS, CernBOX), but which is the most suitable one?

STS-Wire

- https://github.com/DODAS-TS/sts-wire
- STS-Wire is a client application developed by Diego Ciangottini in GO that wraps Rclone with an AuthN/AuthZ layer for OIDC (IAM) to locally mount an S3 bucket from a remote object storage server (MinIO).
- The MinIO storage server must be configured to provide Secure Token Service (STS) functionality, that is, it must authorize the incoming access request based on a AuthZ policy.
- Diego C. implemented an integration between MinIO and Open Policy Agent (OPA) so that OPA processes the incoming IAM token and authorizes requests based on the value of a specific claim in the token.
- https://dciangot.github.io/minio-opa/

STS-Wire (cont'd)

- Anyway, STS-Wire solution encompasses a lot of problems.
- MinIO and OPA container images provided by Diego C. refer to rather old versions of these softwares:
  - MinIO image does not support modern TLS protocol versions (handshake errors).
  - MinIO is deprecating support towards Open Policy Agent.
  - The integration gets broken if MinIO and OPA are updated to latest versions.
- STS-wire client application seems unreliable when transferring files or directories with order of GB sizes.
- Thus, considering all these conditions, STS-Wire does not seem to be a good solution to satisfy the requirements of our use cases.

MinIO STS

- MinIO offers the possibilty to:
  - directly integrating OIDC AuthN establishing a trust with an external IdP
  - authorizing a request by applying a policy based on the value of a specific token claim
  - expose a server API that accepts an OIDC token as input and returns temporary S3 credentials as output of successful AuthN/AuthZ (STS/Assume Role with Web Identity)
- MinIO STS API expects the token to comprise some non-standard claims:
  - when you provide a standard token from IAM, it complains about missing "aud" claim
  - "aud" claim should be equal to OIDC client ID
  - when you provide a IAM token with "aud" != "client_id", it complains about missing "azp"
  - when you provide a IAM token with "aud" == "client_id", it complains about "invalid number of segments"
- It seems that IAM JWT profiles do not include "azp" (while Keycloak's does).
- IAM JWT profiles are not customizable, so a different solution must be taken into account to integrate MinIO and IAM in order to get S3 credentials.

MinIO with Vault-delegated STS

- Hashicorp Vault is a software that allows to securely store secrets.
- Vault is capable to interact with MinIO and get temporary S3 credentials through a plugin developed by Hashicorp people (https://github.com/StatCan/vault-plugin-secrets-minio).
- Access to Vault can be configured to be managed through OIDC AuthN protocol and supports IAM JWT profile.
- As a result, Vault can supply STS functionality for MinIO if integrated with OIDC AuthN using IAM.
- A policy must be defined in MinIO to perform operations on buckets, and that policy is linked to the role Vault assumes using MinIO Admin credentials to get temporary S3 credentials.

MinIO with Vault-delegated STS (cont'd)

MinIO policy (indigo-dc):

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "s3:CreateBucket",
          "s3:DeleteBucket",
          "s3:ListBucket",
          "s3:PutObject"
        ],
        "Resource": [
          "arn:aws:s3:::*"
        ]
      }
    ]
  }
```

Vault role (indigo):

```
vault write minio/roles/indigo
policy=indigo-dc
user_name_prefix=indigo
default_ttl=1h
max_ttl=1h
{
  "role_type": "jwt",
  "user_claim":
"preferred_username",
  "bound_claims": {
    "groups":"indigo-dc"
  },
  "policies": [
    "indigo"
  ],
  "ttl": "1h"
}
```

Vault policy (indigo)

```
path "/minio/keys/indigo"
{ capabilities = ["read", "list"] }
```

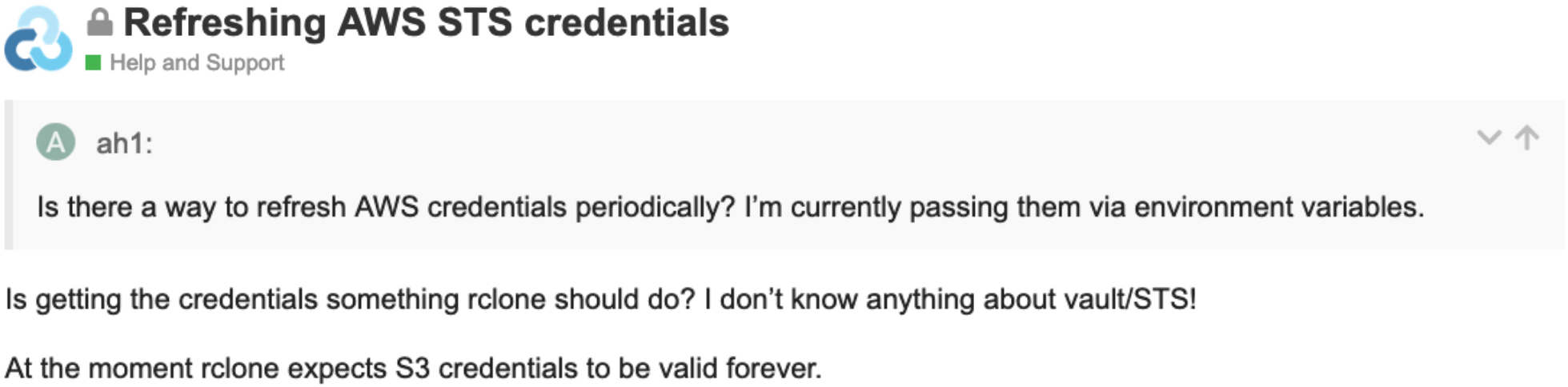
Temporary S3 credentials

Access Key = indigod9bf1775-a9ee-cc29-3eb1-bad04fef6da4
Secret Key = OJGeEiDmb1HivENDava7RFPL

- A policy must also be defined in Vault to authorize temporary S3 credentials requests from OIDC clients based on the value of a specific token claim.

Application to mount S3 bucket (Rclone?)

- You get S3 credentials valid for 1h, so how to keep your locally mounted bucket connected to the storage server when credentials expire?
- Your client application must be smart enough to automatically refresh temporary S3 credentials.
- Unfortunately, Rclone does not fit this requirement (at least for S3).



Refreshing AWS STS credentials

Help and Support

A ah1:

Is there a way to refresh AWS credentials periodically? I'm currently passing them via environment variables.

Is getting the credentials something rclone should do? I don't know anything about vault/STS!

At the moment rclone expects S3 credentials to be valid forever.

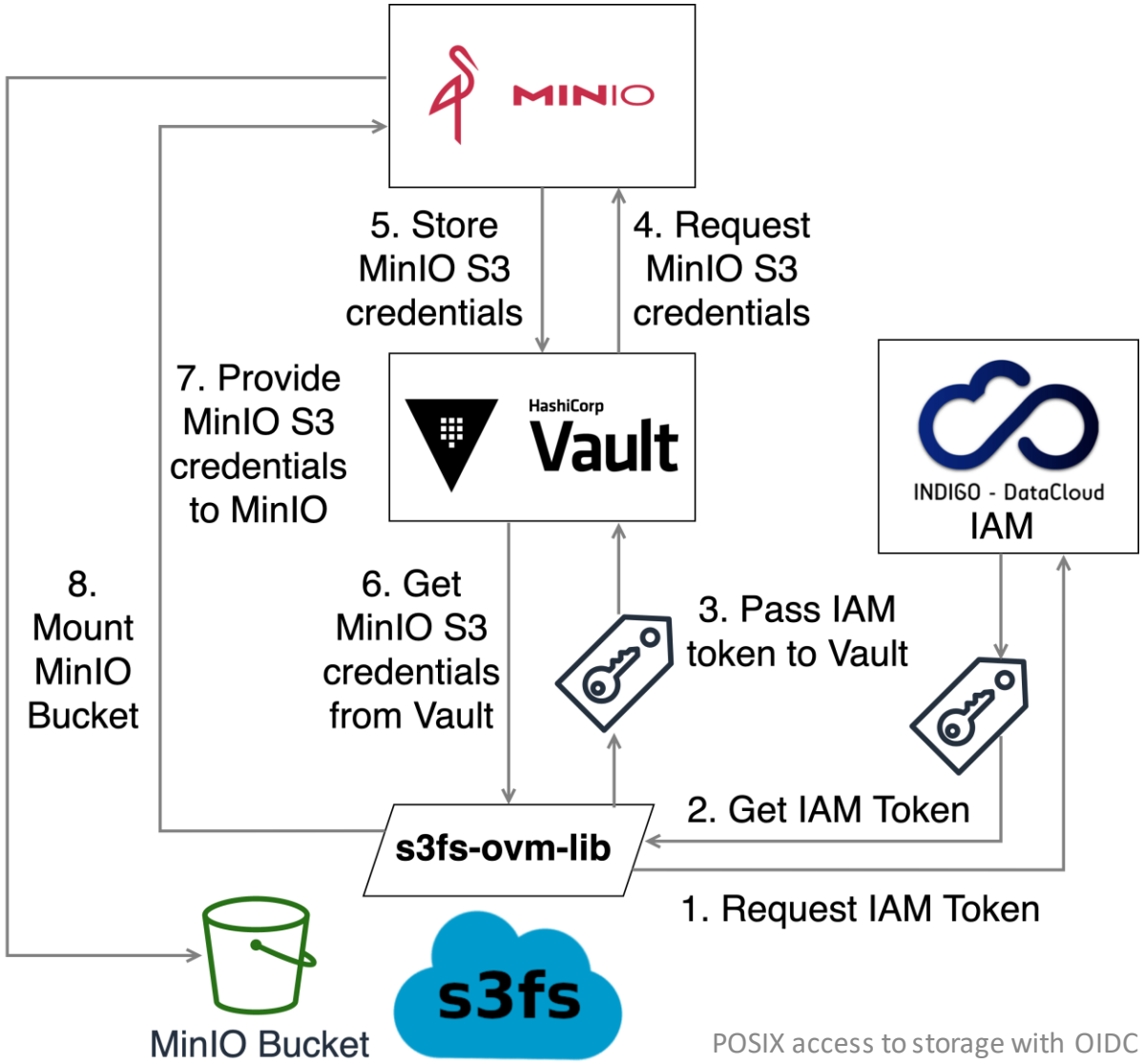
Application to mount S3 bucket (s3fs-fuse!)

- https://github.com/s3fs-fuse/s3fs-fuse/
- s3fs allows Linux, macOS, and FreeBSD to mount an S3 bucket via FUSE. s3fs preserves the native object format for files, allowing use of large subset of POSIX including reading/writing files, directories, symlinks, mode, uid/gid, and extended attributes.
- Compatible with Amazon S3, and other S3-based object stores (MinIO, Google Cloud, IBM Cloud, Oracle Cloud).
- Allows local disk data caching (optional).
- **Provides an option (credlib) to load a custom shared library that automatically refreshes tokens and credentials!**

s3fs-fuse-oidc-vault-minio-lib (s3fs-ovm-lib)

- https://baltig.infn.it/fornari/s3fs-fuse-oidc-vault-minio-lib
- s3fs-fuse-oidc-vault-minio-lib is a shared library (developed in C++) that performs credential processing of s3fs-fuse.
- It makes use of oidc-agent C++ API in order to get an access token from IAM and get the user authenticated and then authorized by Vault.
- Using Vault C++ API, it obtains S3 temporary credentials from MinIO to let s3fs-fuse locally mount a bucket.
- The Baltig project provides the library source code and the instructions to compile it, as well as a script to run s3fs in a container based on a public image downloadable from DockerHub.

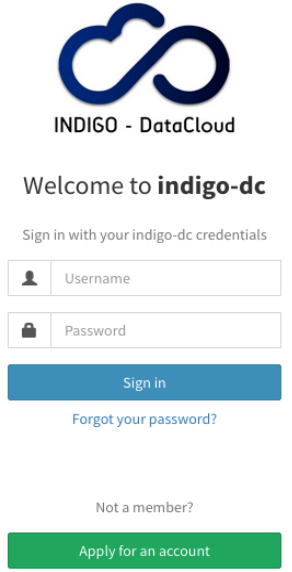
AuthN/AuthZ workflow with s3fs/Vault/MinIO

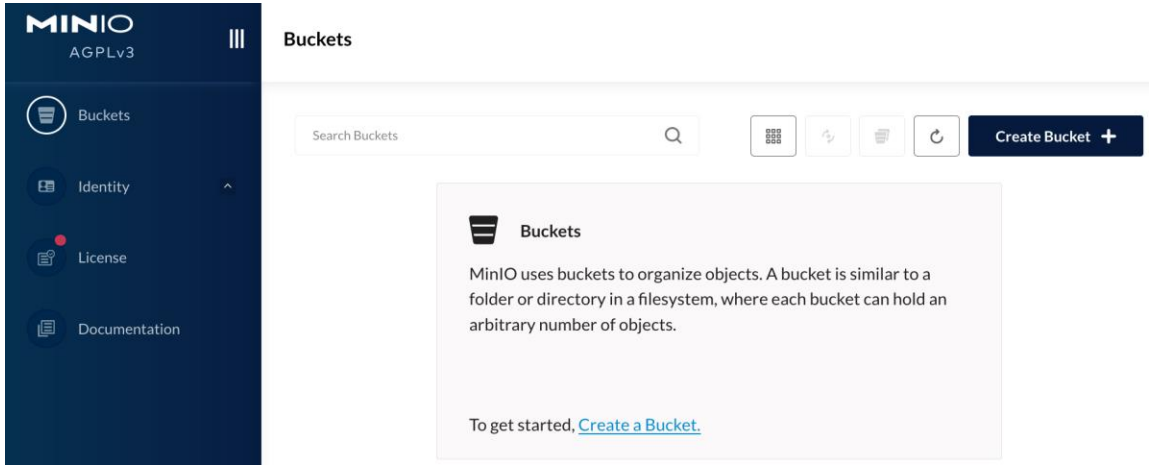


- s3fs-fuse-oidc-vault-minio-lib (s3fs-ovm-lib) takes care of temporary S3 credentials updating whenever s3fs-fuse detects expiration.
- Functionality tests have been executed to verify stable access over credential expiration time range with positive results.
- Large-scale testing activities yet to be organized and carried out.

About s3fs workflow and MinIO-OIDC integration

- MinIO provides a Console to directly access buckets from a browser.
- Console access can be configured with OIDC AuthN/AuthZ setting up trust towards external IdP (IAM) and policy based on specific token claim value.
- This way a user can access his collaboration bucket from the browser in the same way he gets access with s3fs (based on the group his IAM account belongs to).





Alternatives? Rclone + StoRM WebDAV

- https://confluence.infn.it/display/CNAFUS/Rclone%3A+a+solution+for+POSIX+access+to+remote+storage+via+WebDAV
- For WebDAV remote storage, Rclone allows the user to provide a command (oidc-agent) for the application to automatically renew tokens.
- This way, Rclone can locally mount a WebDAV storage area (SA) providing POSIX access to the client.
- However, Rclone needs a local cache (size is customizable) to be setup in the user's home directory. Thus:
  - Read/write operations are always limited by local cache size.
  - Not suitable to mount SA on CNAF UI (homes are exported from remote file system).

Topics for internal discussion

- S3 vs. WebDAV – POSIX access
  - Which one performs better? Large-scale tests needed!
- Which use cases shall we support?
  - Remote home mounted on user's laptop?
    - s3fs + MinIO might be good but AuthZ policies should be tailored on user basis (scalable?).
  - Remote SA mounted on user's laptop?
    - Rclone + WebDAV seems good but limited by local cache.
  - Remote SA mounted on CNAF UI?
    - Rclone + WebDAV seems not good (cache on remote homes file system).
    - s3fs + MinIO might be good (no cache but SA must be exported via S3).
- Do we really want to maintain MinIO and Vault (next to IAM)?
  - Maybe it would be easier to maintain Ceph.
    - Integration between RADOS Gateway and OIDC provider (Keycloak) seems not trivial.

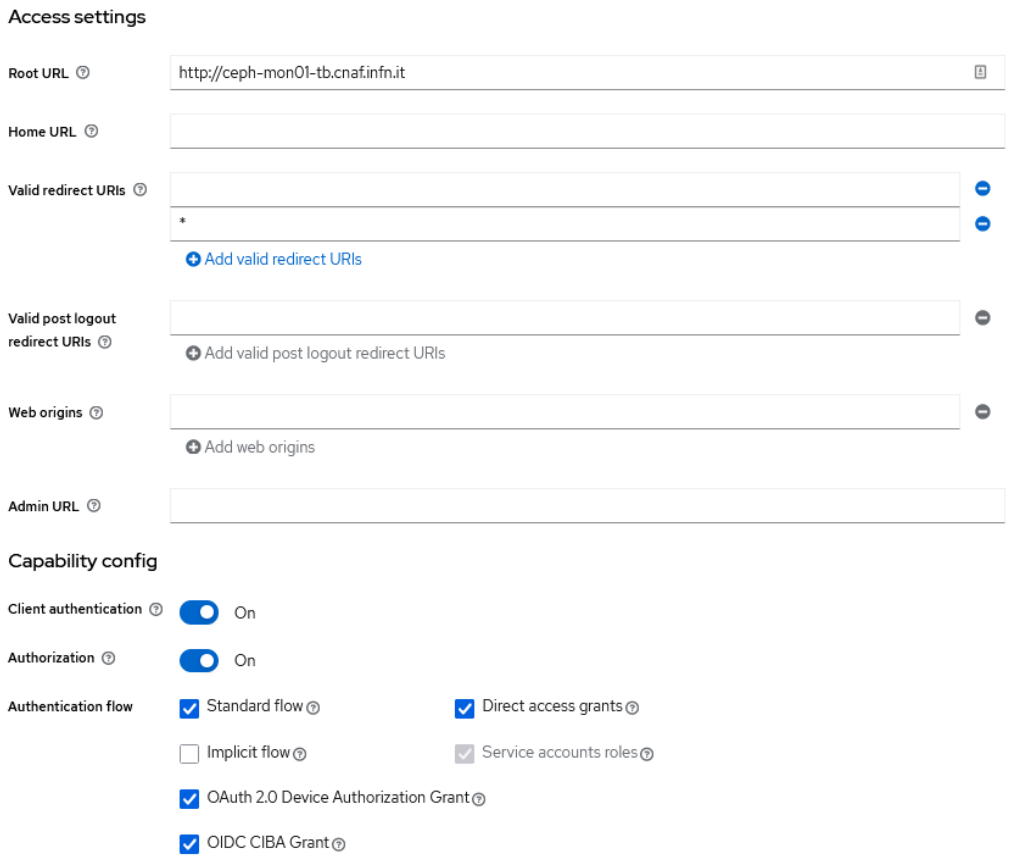
Thank you very much!
Ahmad, the floor is yours!

Integrating RadosGW with Keycloak

- Ceph testbed runs within Cloud@CNAF providing S3 access through its RadosGW API.
- Following Ceph Documentation is difficult, so choosing Keycloak as IAM system ensured better comprehension.
- Keycloak runs as a docker container on a VM deployed at Cloud@CNAF.
- Networking security is considered by setting security groups that limit traffic from the public network.

Keycloak Client Configuration

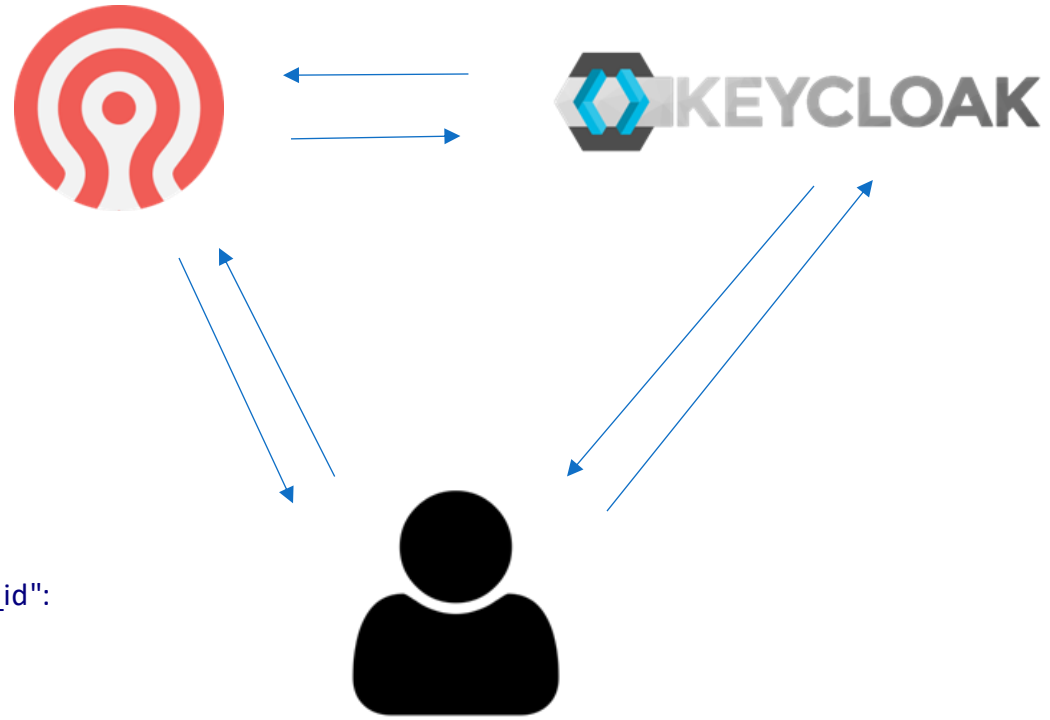
- Create a OIDC client that redirects to RadosGW endpoint.
- Allow client authentication and authorization in a browserless environment.
- Allow token issue to client via HTTP request.



Setting up RadosGW with STS

- Create OIDC provider in RadosGW.
- Create role with IAM policy to authorize keycloak client's requests.
- Assume role when a web identity token is presented with proper claim values.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Principal": {
    "Federated": [
     "arn:aws:iam:::oidc-provider/keycloak-
demo.cloud.cnaf.infn.it:8443/realms/demo"
    ]
   },
   "Action": [
    "sts:AssumeRoleWithWebIdentity"
   ],
   "Condition": {
    "StringEquals": {
     "keycloak-
demo.cloud.cnaf.infn.it:8443/realms/demo:app_id":
"account"
    }
   }
  }
 ]
}
```



Next Steps

- Adding Policies to manage user access to Ceph.
- Investigating integration of RadosGW with IAM.
- Foreseen usage of s3fs-fuse to provide POSIX-like mounting of S3.