AWS IAM Assume Role Setup for Teradata OTF

⚠ Teradata Confidential - Internal use only

This document provides guidelines on how to setup AWS assume role to allow Teradata Vantage to access (read/write) customer data in Amazon S3 bucket.

Pre-requisites

For this guide we are assuming we have two separate AWS accounts, one that represents Teradata AWS account and other would be Customer side AWS account where the storage account is located. The goal here is to access data residing in the Customer's storage account from Teradata Vantage using STS AssumeRole capability of AWS for cross account data access.

EC2 instances on Teradata AWS account should be configured with an IAM Instance Role and have it attached to it. The policy for that IAM role should allow `"sts:AssumeRole"` action, which allows that attached role to assume some another role.

Example JSON snippet of a policy on Teradata AWS account that allows STS:

```
{
    "Action": [
        "sts:AssumeRole"
    ],
    "Effect": "Allow",
    "NotResource": "arn:aws:ssm:*:915556001112:role/*",
    "Sid": "NOSAssumeRole"
}
```

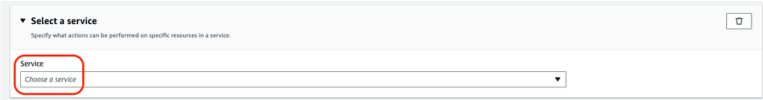
Configurations on Customer AWS account

NOTE: All instructions in this section are to be performed by our customer in their AWS account management console.

IAM Policy

We will first be creating an IAM policy that would allow access from Teradata to customer's S3 bucket.

1. Log on to AWS Management console.
2. Either from home dashboard OR search bar on the top, select/type "**IAM**".
3. AWS recommends using regional STS endpoints to reduce latency and hence now we will make sure we have enabled endpoints for all the regions that we intend to use STS service from.
   a. Under **Access Management** from left pane, select **Account Settings**.
   b. You will see **Region Name**, **Endpoint** and **STS Status**. If the desired region/endpoint has an inactive status, flip the switch to activate it.
4. Go to **Policies** from the left pane. Here we will create a new policy for Teradata access. Click **Create Policy**.
5. Under **Select a service**, select **Glue**. (Note: skip steps 5 & 6 and go to step 7 if Glue catalog is not used)



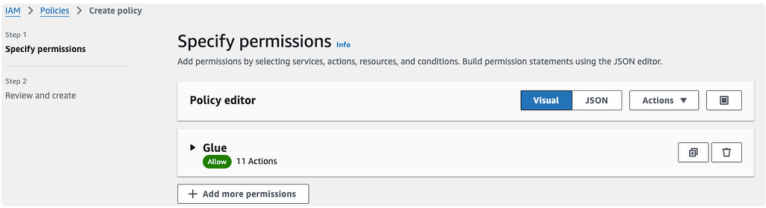
   a. Under **Actions allowed**, add appropriate permissions to the policy that you want Teradata to do on your Glue catalog.

```
"glue:GetDatabase",
"glue:GetDatabases",
"glue:GetTable",
"glue:GetTables",
"glue:CreateDatabase",
"glue:CreateTable",
"glue:DeleteDatabase",
"glue:DeleteTable",
"glue:UpdateTable",
```

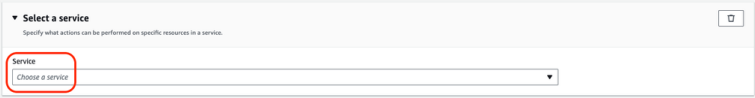
```
"glue:UpdateDatabase"
```

b. Under **Resources**, add ARN of a **Catalog, Database and Table** to restrict those action to specific databases and tables only.

6. Click on **Add more permissions** in the bottom.



7. Under **Select a service**, select **S3**.

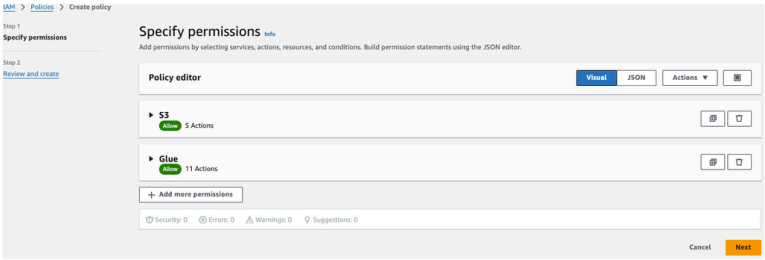


a. Under **Actions allowed**, add appropriate permissions to the policy that you want Teradata to do on your storage.

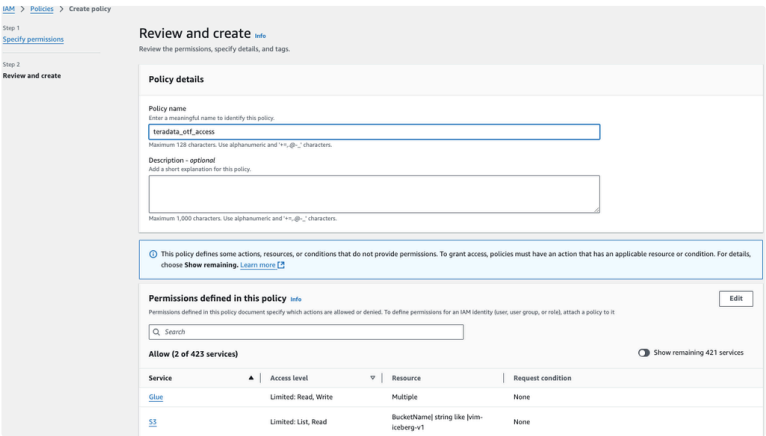
```
"s3:PutObject",
"s3:GetObject",
"s3:DeleteObject",
"s3:ListBucket",
"s3:ListAllMyBuckets",
"s3:GetBucketLocation",
```

b. Under **Resources**, add ARN of a **bucket** to restrict those action to one or more buckets only.

8. Click **Next**.



9. Give this policy a name, something like "**teradata_otf_access**".



10. Click **Create policy**.
11. Note down this policy name as we would need it when creating **IAM Role** below.

Example

Here is an example of a JSON policy for Glue and S3, restricting access to a specific Glue database, table and S3 bucket named. In addition, this example also outlines the read and write access into separate sections:

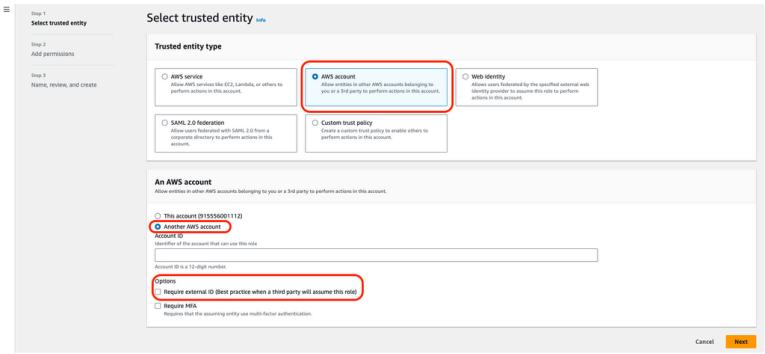
```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "ReadGlue",
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:GetTable",
                "glue:GetTables"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:158228458290:catalog",
                "arn:aws:glue:us-west-2:158228458290:database/<database>",
                "arn:aws:glue:us-west-2:158228458290:table/<database>/<table>"
            ]
        },
        {
            "Sid": "WriteGlue",
            "Effect": "Allow",
            "Action": [
                "glue:CreateDatabase",
                "glue:CreateTable",
                "glue:DeleteDatabase",
                "glue:DeleteTable",
                "glue:UpdateTable",
                "glue:UpdateDatabase"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:158228458290:catalog",
                "arn:aws:glue:us-west-2:158228458290:database/<database>",
                "arn:aws:glue:us-west-2:158228458290:table/<database>/<table>"
            ]
        },
        {
            "Sid": "ReadS3",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:ListAllMyBuckets",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket>",
                "arn:aws:s3:::<bucket>/*"
            ]
        },
        {
            "Sid": "WriteS3",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket>",
                "arn:aws:s3:::<bucket>/*"
            ]
        }
    ]
}
```

IAM Role

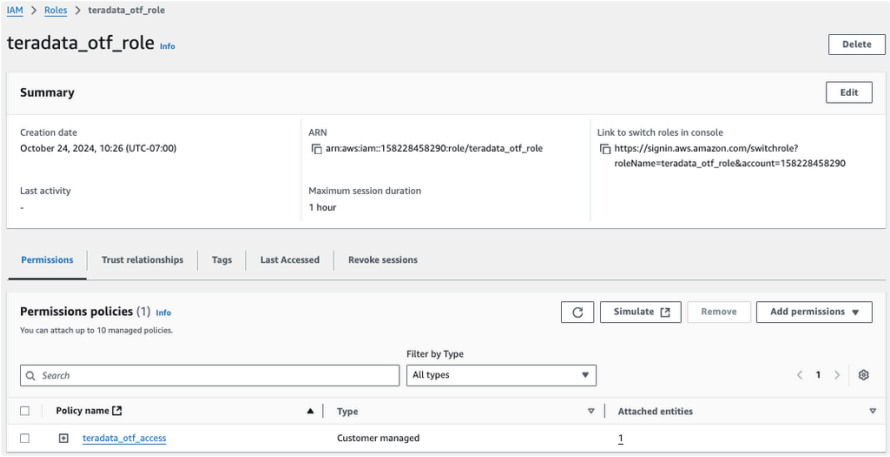
We will now create a role to grant an external account (Teradata's account) to access customer's S3 data.

1. Log on to AWS Management console.
2. Either from home dashboard OR search bar on the top, select/type "**IAM**".
3. Under **Access Management** from left pane, select **Roles**.
4. Click **Create role**.
5. For **Trusted entity type**, select **AWS account**.
6. Under AWS account section, select **Another AWS account**.



a. Under **Account ID**, provide **Teradata account ID** to be the identifier of the account that can use this role.

7. Under **Options**, select **Require external ID**. You can increase the security of your role by requiring an optional external identifier, which prevents "confused deputy" attacks. This is recommended if you do not own or have administrative access to the account that can assume this role. The external ID can include any characters that you choose. To assume this role, users must be in the trusted account and provide this exact external ID.
   a. Enter **Teradata Vantage Site ID** (which can be obtained from Teradata's accounts team).
   b. Please note this down as you will require it later when setting up Authentication object.
8. Click **Next**.
9. Search or select the policy "**teradata_otf_access**" that was created under **IAM Policy** section.
10. Click **Next**.
11. Add a name to this role, something like "**teradata_otf_role**" and click **Create role**.
12. Navigate to the details of this role (if it does not automatically) and record the **ARN**. This **Role ARN** will be used when creating authorization object.



Example

For VCE, Here is an example JSON of a customer's IAMRole trust relationship that includes VCE system's IAMRole name in the principal section to assume this customer role. Additionally, an External ID string (shown as a dummy "TDICAMSICRPVP01" below) is added as condition to assume the role that will be validated with every STS request. This resolves confused deputy situation.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": ["arn:aws:iam::349718792561:role/TDICAMSICRPVP01-IC-td-ecosystem-DBMppRole"]
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "sts:ExternalId": "TDICAMSICRPVP01"
                }
            }
        }
    ]
}
```

For VCL, the customer IAMRole should include the VCL POG and COG IAM Role names in the principal section. Here is an example

```

```

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": ["arn:aws:iam::xxxx04618xxx:role/tcedbbe9/tenants/01/tcedbbe9-01-pog20250205064011660200000888",
                        "arn:aws:iam::xxxx04618xxx:role/tcedbbe9/tenants/01/tcedbbe9-01-pog20250205064011660200000999"
                       ]
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "sts:ExternalId": "TDICAMSICRPVP01"
                }
            }
        }
    ]
}
```

Authorization object

For creating Authorization object that would use Assume Role feature and AWS STS service, please have the recorded values handy. The details that we will need are:

1. External ID, captured in step #7 under IAM Role section.
2. Role ARN, captured in step #12 under IAM Role section.

Few example of authorization object to be used for Assume roles and its use with OTF datalake:

Examples to create an authorization and associate it with OTF Datalake

```
CREATE AUTHORIZATION otf_assume_role_auth
USING
AUTHSERVICETYPE 'ASSUME_ROLE'
ROLENAME 'arn:aws:iam::158228458290:role/teradata_otf_role'
EXTERNALID '1234';


CREATE DATALAKE iceberg_glue
EXTERNAL SECURITY  CATALOG otf_assume_role_auth,
EXTERNAL SECURITY  STORAGE otf_assume_role_auth
USING
catalog_type ('glue')
storage_region ('us-west-2')
storage_location ('s3://vim-iceberg-v1/')
TABLE FORMAT iceberg;
```

Validation

Using `aws` CLI

If our customer has access to Teradata Vantage instance console, they can run following `aws` CLI commands to validate STS assume role setup by SSH into a primary cluster database node.

```
# Obtain token on Teradata instance using/assuming customer's role along with Teradata provided external id.
<CUSTOMER_ROLE_ARN> = The value captured in step #12 under IAM Role section.
<EXTERNAL_ID> = The value captured in step #7 under IAM Role section.
<SESSION_NAME> = Any string to name this session.
> aws sts assume-role --role-arn <CUSTOMER_ROLE_ARN> --external-id <EXTERNAL_ID> --role-session-name <SESSION_NAME>

Example:
# aws sts assume-role --role-arn arn:aws:iam::158228458290:role/teradata_otf_role --external-id 1234 --role-session-name vimsession
{
    "Credentials": {
        "AccessKeyId": "ASIASJVZGVMZHI52SXO2",
        "SecretAccessKey": "vqIEITXsp+zNJ3BFZGkx4fasLvsRTiOyd6RBAhKu",
        "SessionToken":
"IQoJb3JpZ2luX2VjEHIaCXVzLWVhc3QtMSJGMEQCIDLyWLHLVlWTw7TsrwUJEk/m7NJNLf2TEOYed1NvE5rmAiBnPKCcvDTueMv8H6Tc00/yNkQYjdTot9B0BGdXOjBGQiqgAgjb/////////8B
EAEaDDE1ODIyODQ1ODI5MCIMgR5j7uOW5iWiTu/MKvQBpsFYwq70QnG+bAZgCK3mke4J0MalTf1mLn7/0JkIyGd2woOOxql5vm9IJbbVt12HJ+3QwwIJzit4Fyj1b2d1FtAifWYCJHUziOEY/6xML
s7MQ7KCxOZ/92UVKSPcNyzaX2eTCO/0T55gF7jEIP4uSku5ZUGe4hI/3j5zIBn/TdXl4pOlfHnmf9epRJeA1LQYQQZMgEGFWZt65mCd+DWZMVoueyLDTr1wBoyF9ft+EgyrInbaj082KtCyo1CAM0
/vm9EIpb5jbAd+ZpAB2qUDOvhUh9/P9yycj7jNijTLbEr+5/qEQBWm8uAz60ffuenapo1hzDC7juq4BjqeAWubMhVbiPN+vLyP1skRhLqYSjAegnoEctr3nKrGN4A3bOkk14gQAHD/4nFB3yxNQ4i
rUoNz8HgIzMDxFWlBqlVI/tWvegJgGGCp533LXaIs/zKNY7UuEchaeUzg6Nwymdv8IKpYC3URzHhfNL2pJ8wJEYHQQJXBJWHtFWeqYstBWa0CKZF4U8clIr4hjbiYwIUpKBHeGI72sWogGrtw",
        "Expiration": "2024-10-24T18:43:23Z"
    },
```

```
    "AssumedRoleUser": {
        "AssumedRoleId": "AROASJVZGVMZCJKJ6Z7AL:vimsession",
        "Arn": "arn:aws:sts::158228458290:assumed-role/teradata_otf_role/vimsession"
    }
}
```