STS Role assumption to publish message (Learnings)

Context

In AWS, permission to perform any operation is governed by the IAM role of the user and policy attached to the user. To learn about IAM roles please go through this video : AWS IAM Core Concepts You NEED to Know

AWS provides AWS Security Token Service (**AWS STS**) as a web service that enables you to request temporary, limited-privilege credentials for AWS Identity and Access Management (IAM) users or for users you authenticate (federated users) (doc). To learn about STS please go through this video : AWS Security Token Service - AWS STS

If you want to play with IAM roles using AWS CLI please follow this link : https://aws.amazon.com/premiumsupport/knowledge-center/iam-assume-role-cli/

Now we will give you some hands-on with our AWS setup.

1. Go to My Apps and access **AWS Fluidly SSO**. From there go to **AWS Loan Monitor**. Here select **command line** option. You will be presented with access key, access id and secret token of **your account**(developer account). Copy the Option 1 details and paste in your local terminal. (Remember this credentials have lifetime of 12 hours)
2. Now run following command : **aws sts get-caller-identity** This will show you your account details. The same can be seen from web as well : https://us-east-1.console.aws.amazon.com/billing/home?region=us-east-1#/account. Please notice the account ARN we are assuming the role of **AWSEngineerAccessRole.**
3. We would like to list all IAM roles present in **AWS Loan Monitor.** For that please run following command : **aws iam list-roles > filename.txt**. In this output file we have interest in one role i.e. : **loan-monitor-backend-ecs-task**. This is role used by our backend service to access resource of **AWS**.
4. Now we would like to list all policies of the is role. Command for same is : **aws iam list-role-policies --role-name loan-monitor-backend-ecs-task**
5. We are interested in **sts:assume** policy. So we would like to see that policy. Command for that policy is : **aws iam get-role-policy --role-name loan-monitor-backend-ecs-task --policy-name terraform-20220909104832902900000000**
6. Above mentioned policy is defined in ecs.tf of our backend repo. : https://github.com/oaknorthbank/loan-monitor-backend/blob/ed26aa45953a46b61bc10ea5736b08092b2c7aef/terraform/ecs.tf#L144
7. Above mentioned role can also be accessed via web : https://us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/roles/details/loan-monitor-backend-ecs-task?section=permissions
8. To assume role of **loan-monitor-backend-ecs-task** command is as following :

```
aws sts assume-role --role-arn "arn:aws:iam::756570553573:role/loan-monitor-backend-ecs-task" --role-session-n

An error occurred (AccessDenied) when calling the AssumeRole operation: User: arn:aws:sts::756570553573:assume
is not authorized to perform: sts:AssumeRole on resource: arn:aws:iam::756570553573:role/loan-monitor-backend-
```

As seen from the above error statement, our developer role is allowed to assume role of **loan-monitor-backend-ecs-task.** In fact, for a user who has assumed the role of **loan-monitor-backend-ecs-task** is also not allowed to assume the same role again unless it is specified explicitly. This is the reason why we had define this permission as mentioned in point 6. Reference : AWS AssumeRole - User is not authorized to perform: sts:AssumeRole on resource

Problem

Reference : https://github.com/oaknorthbank/loan-monitor-backend/blob/8f569cb058f3a725efac6fd8bed6248dd30282e0/loan_monitor_api/workload_identity_federation.py#L8

```
aws_credentials = boto3.Session().get_credentials().get_frozen_credentials()
```

```
aws_credentials = boto3.Session().get_credentials().get_frozen_credentials()
```

Above mentioned code snippet is creating a boto session and returning the corresponding credentials. These credentials had validity of around 12 hours. So if we were trying to publish message after 12 of deployment we GCP pubsub we were getting token expiry error. We needed some function to fetch valid/refreshed credentials whenever we try to publish the message.

Solution

As discussed in the context section, STS was good option to go for. It provides us with temporary credentials to work with.

Reference : https://github.com/oaknorthbank/loan-monitor-backend/blob/main/loan_monitor_api/refreshable_boto_session.py#L14

```
role_arn = os.environ.get("IAM_ROLE_ARN")
sts_client = boto3.client("sts")
response = sts_client.assume_role(
        RoleArn=role_arn,
        RoleSessionName="loan-monitor-backend-ecs-task-session",
        DurationSeconds=ttl,
    ).get("Credentials")
```

One point to note here is that sts boto client was still valid after 12 hours but boto session was not. With help of assume role function of sts client we are able to fetch new and valid(for specified TTL) credentials every time we would like to publish message to pubsub. But to make this work we had to do 2 changes in terraform as well.

1. While running above code we got following error :

```
{     "exc_info": true,
"event": "Error in refreshing credentials: An error occurred (AccessDenied) when calling the
AssumeRole operation: User: arn:aws:sts::756570553573:assumed-role/loan-monitor-backend-ecs-task/8c803531804044ac
is not authorized to perform: sts:AssumeRole on resource: arn:aws:iam::756570553573:role/loan-monitor-backend-ecs
 "timestamp": "2022-09-08T14:40:24.016698Z" }
```

To fix this we had to specify permission to to assume role(**loan-monitor-backend-ecs-task**) even for it own user. We did so as mentioned in **point 6** above. While specifying permission we had expose ARN of IAM role as output variable. We did so by adding this line : https://github.com/oaknorthbank/tf-shared-modules/blob/3ce4d4e5987c68f78c25476fd7a06b18427f9a8e/aws_ecs_service/outputs.tf#L9

2. We exposed ARN of the role (**loan-monitor-backend-ecs-task**) in terraform as a env variable that can be used by the refresh code of python. We did so by adding this line : https://github.com/oaknorthbank/loan-monitor-backend/blob/ed26aa45953a46b61bc10ea5736b08092b2c7aef/terraform/ecs.tf#L44