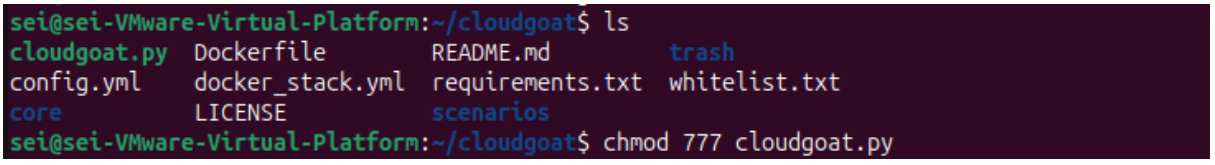
CloudGoat_[BoB13]서소영(Seo soyoung)

마감기한	@2024년 8월 13일
담당 멘토	니코 멘토님
분야	디지털 포렌식

CloudGoat Setting

```
git clone https://github.com/RhinoSecurityLabs/cloudgoat.git
```




```
cd cloudgoat
chmod 777 cloudgoat.py
pip install -r ./core/python/requirements.txt
```

```
sudo apt-get update && sudo apt-get install -y gnupg software-properti
```

```
wget -O- https://apt.releases.hashicorp.com/gpg | gpg --dearmor | sudo
```

```
gpg --no-default-keyring --keyring /usr/share/keyrings/hashicorp-archi
```

```
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg
```

```
sudo apt update
sudo apt-get install terraform
```

```
pip3 install -r ./requirements.txt
```

```
aws configure --profile seisy12
WS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]: us-east-1
Default output format [None]: json
```

```
./cloudgoat.py config profile
```

```
./cloudgoat.py config whitelist --auto
```

```
./cloudgoat.py create codebuild_secrets

 ./cloudgoat.py destroy codebuild_secrets (반드시!!)
```

https://github.com/RhinoSecurityLabs/cloudgoat/tree/master/scenarios/codebuild_secrets

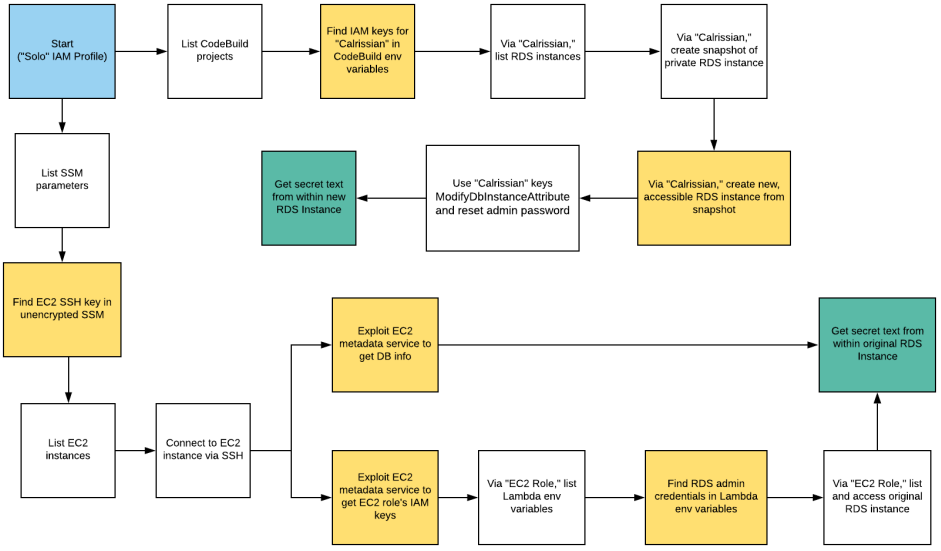
cloudgoat_codebuild_secrets

Summary

Starting as the IAM user Solo, the attacker first enumerates and explores CodeBuild projects, finding unsecured IAM keys for the IAM user Calrissian therein. Then operating as Calrissian, the attacker discovers an RDS database. Unable to access the database's contents directly, the attacker can make clever use of the RDS snapshot functionality to acquire the scenario's goal: a pair of secret strings.

Alternatively, the attacker may explore SSM parameters and find SSH keys to an EC2 instance. Using the metadata service, the attacker can acquire the EC2 instance-profile's keys and push deeper into the target environment, eventually gaining access to the original database and the scenario goal inside (a pair of secret strings) by a more circuitous route.

Note: This scenario may require you to create some AWS resources, and because CloudGoat can only manage resources it creates, you should remove them manually before running `./cloudgoat destroy`.



💡 **< Scenario Start(s) >**

→

IAM User "Solo"

💡 **< Scenario Goal(s) >**

→ A pair of secret strings stored in a secure RDS database.

```
(env) sei@sei-VMware-Virtual-Platform:~/cloudgoat$ tree
.
├── cloudgoat.py
├── codebuild_secrets_cgidfjzufpipzz
│   ├── assets
│   │   ├── buildspec.yml
│   │   ├── lambda.py
│   │   └── lambda.zip
│   ├── cheat_sheet_calrissian.md
│   ├── cheat_sheet_solo.md
│   ├── cloudgoat
│   ├── cloudgoat.pub
│   ├── manifest.yml
│   ├── README.md
│   ├── start.sh
│   ├── start.txt
│   └── terraform
│       ├── codebuild.tf
│       ├── data_sources.tf
│       ├── ec2.tf
│       ├── iam.tf
│       ├── lambda.tf
│       ├── outputs.tf
│       ├── provider.tf
│       ├── rds.tf
│       ├── ssm-parameters.tf
│       ├── terraform.tfstate
│       ├── variables.tf
│       └── vpc.tf
├── config.yml
├── core
│   ├── python
│   │   ├── commands.py
│   │   ├── help_text.py
│   │   ├── __pycache__
│   │   │   ├── commands.cpython-312.pyc
│   │   │   ├── help_text.cpython-312.pyc
│   │   │   └── utils.cpython-312.pyc
│   │   ├── python_terraform
│   │   │   ├── __init__.py
```

- tree - 디렉토리 내 파일을 트리 형식으로 보여준다.

  tree - Displays the files in a directory in a tree format.

```
(env) sei@sei-VMware-Virtual-Platform:~/cloudgoat/codebuild_secrets_cgidfjzufpipzz$ cat start.txt
cloudgoat_output_aws_account_id = 831926608298
cloudgoat_output_solo_access_key_id = AKIA4DMVQUGVNEZUJCHP
cloudgoat_output_solo_secret_key = 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
```

```
cat start.txt
```

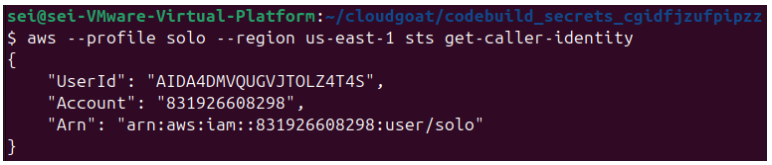
💡 cloudgoat_output_aws_account_id = 831926608298
cloudgoat_output_solo_access_key_id = AKIA4DMVQUGVNEZUJCHP
cloudgoat_output_solo_secret_key =
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

```
sei@sei-VMware-Virtual-Platform:~/cloudgoat/codebuild_secrets_cgidf
jzufpipzz$ aws configure --profile solo
AWS Access Key ID [None]: AKIA4DMVQUGVNEZUJCHP
AWS Secret Access Key [None]: hOcHOi88HlnjgOIEdxljV12Fd3/mc8ex209eR
S5F
Default region name [None]: us-east-1
Default output format [None]:
```

```
aws configure --profile solo
```

- solo라는 이름을 가진 user가 발견되어, 해당 계정의 credential을 등록한다.

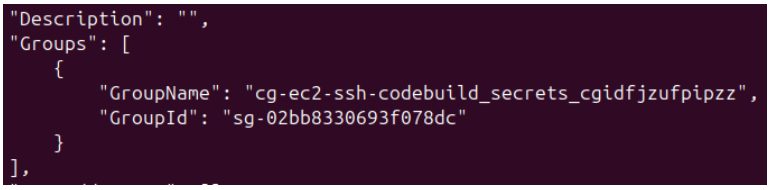
  A user named "solo" has been found, and the credentials for that account are being registered.




```
aws --profile solo --region us-east-1 sts get-caller-identity
```

- solo의 AWS 계정 ID, 사용자 ID, ARN이 확인된다.

  The AWS account ID, user ID, and ARN for the user "solo" are verified.




```
aws ec2 describe-instances --profile solo
```

- 해당 명령어를 사용하여 현재 계정에서 실행 중인 모든 EC2 인스턴스의 정보를 나열한다.

  The command is used to list information about all running EC2 instances in the current account.

- publicIP : 23.20.28.120
- instance id : 084cf29ee93d523ae
- security-group-id : 02bb8330693f078dc


```
aws ec2 describe-security-groups --profile solo
```

- AWS 계정에서 사용 가능한 보안 그룹의 상세정보 조회한다.

  Retrieve detailed information about the security groups available in the AWS account.Retrieve detailed information about the security groups available in the AWS account.

- 22번 포트를 통해서 SSH 연결을 할 수 있다.

  SSH connections can be made through port 22.


```
aws ssm describe-parameters --profile solo
```

- 파라미터 정보를 출력한다.

  Output the parameter information.

- "ARN": "arn:aws:ssm:us-east-1:831926608298:parameter/cg-ec2-private-key-codebuild_secrets_cgidfjzufpipzz”
- "Name": "cg-ec2-private-key-codebuild_secrets_cgidfjzufpipzz”


```
aws ssm get-parameter --name cg-ec2-private-key-codebuild_secrets_cgi
```

- cg-ec2-private-key-codebuild_secrets_cgidfjzufpipzz라는 이름의 파라미터를 조회한다.

  Retrieve the parameter named "cg-ec2-private-key-codebuild_secrets_cgidfjzufpipzz".


```
aws lambda list-functions --region us-east-1
```

- AWS Lambda 함수의 목록을 조회한다.

  Retrieve a list of AWS Lambda functions.
