[DEMO] Configuring AWS access credentials

Christophe • August 27, 2023

For this course, I highly recommend that you create a separate AWS account. Either that or make sure that the account you plan on using HAS NO PRODUCTION RESOURCES!

This is extremely important, because the actions we are going to take in this course will not only be destructive, they're also going to create vulnerabilities in your AWS account. You do not want that running alongside any production resources.

Take the time to pause here and create a separate AWS account. Or if you've never created an account before, then you'll want to do that now.

You can follow the instructions from either of these pages to do that and it's pretty straight forward so I won't be showing it for the sake of time:

- https://docs.aws.amazon.com/accounts/latest/reference/manage-acct-creating.html
- https://repost.aws/knowledge-center/create-and-activate-aws-account

Once you have your new AWS account, we're ready to get started with this lesson.

The first step we want to take is to create a new IAM user. This user is going to be the user that creates the lab environment and resources for us. As a general best practice, we should never use the root user after we've created our account except to create other users that we will use for day-to-day administration. If you need more information on that, please refer to our Introduction to AWS Security course.

Steps to creating an IAM user, policies, access keys

With that, pull up IAM from the AWS console and go to `Users`. We'll create a new user so that we can give them specific permissions.

Name this user whatever you'd like:

```
pacu-example
```

We do not need to provide console access so leave that unchecked.

Normally I'd recommend using groups to provide permissions but this is a throwaway example so we'll skip that step to save time.

`Create User`

We now need to provide permissions.

Switch over to the `JSON` tab and copy/paste this policy (available in the description below):

```
{
        "Version": "2012-10-17",
        "Statement": [
                {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Action": [
                "iam:Get*",
                "iam:List*",
                "iam:Put*",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Resource": "*"
        }
        ]
}

```

This is the policy that we'll start with for the course, and then we'll updating it as we go along for other scenarios.

This user here is providing access a specific S3 bucket and its data, but whoever set up these permissions made a massive mistake, which is that they provided this user with very broad IAM permissions as well.

Something like this could easily happen in the real world when someone is moving quickly and copy/pasting other policies or using pre-created policies without:

1. Understanding what the policy actually does
2. Without properly testing policies

I have seen stuff like this in production before, this is definitely not an exaggeration.

Go ahead and create this policy and give it a name, like:

```
pacu-example-policy
```

Once the policy is created, we need to go back to our prior tab where we were creating the user, refresh the permissions policies, and then search for our new policy.

Then, select it and apply it to the user, and then create the user.

Now that our example user has this policy attached, we need to generate `Security Credentials` by going to the Users page, clicking on our new user, and going to the Security Credentials tab.

From there, scroll down until you see `Access Keys` and `Create access key`.

For this video, our use case will be to access the AWS API via the CLI, so we can select `Command Line Interface (CLI)`.

You'll get a warning asking you instead to use the `AWS CloudShell` or `AWS CLI V2`, but go ahead and click the checkbox confirming you understand and click on `Next`.

We don't need to set a description, so click on `Create access key`.

We now have our AWS access keys, so we need to copy them and put them into Pacu.

If you don't have Pacu up and running yet, then go ahead and spin it up with whatever method you used.

Give your session name. It can be anything you want, like cybr-course

Then, type in the command `set_keys`

It will ask you to create a `Key Alias` which is optional but it definitely doesn't hurt. This can be anything you want, it doesn't matter. I'll leave mine empty.

Next, you need to paste your `Access key ID`

After that you need to paste in your `Secret access key`

This is the key that you should never give out to anybody else. Right after I'm done recording this, I'll delete this key pair because otherwise you'll be able to use it to access my AWS environment.

Next we can give it a `session token` for temp AWS keys, but we don't need that for this example.

You now have access keys configured for this session, and we can start running some commands as our `pacu-example` user and its permissions.

That concludes this lesson, which sets us up for the following lesson. So go ahead and complete it, and I'll see you in the next!