

- Filename: intro-to-aws-pentesting-4-3-2-lambda-priv-esc-pt-2.md
- Show Name: Intro to AWS Pentesting
- Topic Name: CloudGoat
- Episode Name: Lambda Priv Esc: Pt.2

Lambda Priv Esc: Pt.2

Objectives:

- Utilize aws pentesting tools, tactics, and procedures to solve the CloudGoat 'Lambda_PrivEsc' challenge
- Perform information gathering through enumeration of policies and permissions
- Utilize improperly set IAM User and Role permissions to exploit Lambda function for administrative access to AWS

-
- More Priv Esc with Lambda and iam:PassRole
 - The LambdaManager policy allows for any resource...
 - **Lambda:***
 - **iam:PassRole**
 - https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_passrole.html
 - Allows a service to assume a role later and perform actions on your behalf
 - This means we can...
 - Create and Invoke a Lambda Function
 - Function will run as the Lambda Service Role
 - Priv Esc to the Lambda Service Role permissions
 - How to create Lamba Function
 - Beyond the scope of this tutorial
 - Lambda Function in Python found in CG cheat sheets
 - Create python script called lambda_function.py
 - Slap code in...
 - Don't forget to modify code with Chris' username
 - SAVE...
 - Add to zip file lambda_function.py.zip
 - `zip lambda_function.py.zip lambda_function.py`
 - Now use awscli to create the lambda function in AWS
 - `aws --profile lambda_mgr lambda create-function --function-name admin_function --runtime python3.6 --role <role_arn for Debug-Role policy> --handler lambda_function.lambda_handler --zip-file fileb://lambda_function.py.zip`
 - Invoke the function
 - `aws --profile lambda_mgr lambda invoke --function-name admin_function output.txt`
 - Should get a 200 status code
 - Last step: Confirmation of success
 - Check chris profile for new managed policy
 - `aws --profile chris iam list-attached-user-policies --user-name chris-xxxxxx`
 - Should now see *AdministratorAccess* policy