CSC 321 Lab 5
Hank Tsai

AWS Lab

Answer to task 1:
I believe this authentication for login to AWS management console was a decent choice, but not the best choice. It was decent because we have to know the account ID, username and password to login which is the 'I know' category of the access control. Moreover, the temporary password we used was random, which provides certain security as well. A better way to authenticate is through AWS CLI or AWS API. For AWS CLI, you have to provide an access key ID and secret access key. Multi Factor Authentication (MFA) is also the best practice that adds extra protection.

Answer to task 2:
EC2 is the AWS computing cloud that allows its customers to run applications on virtual servers and provides scalable computing capacity. It had many API errors displayed, it was either because we are not authorized to perform operations on these instances (insufficient permission). AWS S3 service is a data storage service that allows its customers to store and retrieve any amount of data at any time, from anywhere. Objects that need to be stored in Amazon S3 are contained in the buckets.

Answer to task3 :
I couldn't see the security recommendations and IAM resources because I did not have permission required to perform this operation. And in order to see, we'd have to have administrators to add the permissions. DeveloperGroupPolicy describes a series of actions that is allowed. It basically defined what is allowed and not allowed for an IAM role that is attached to the policy. We were permitted to copy the JSON because the devuser IAM role is attached to the DeveloperGroupPolicy and we have the read level access.

Answer to task4:
"s3: CreateBucket" policy allowed me to create a bucket. The upload failed because we did not have permissions to upload files and folders. We need PutObject action to provide the write permissions, then the operation will succeed. My impression on the listing of actions defined by Amazon S3 was, those permissions were so divided into small pieces. And they had way more permissions than I expected them to have. Most of the permission actions were easy for me to understand because they had the description part of it.

Answer to task5:
So since we created BucketsAccessRole in our IAM role management console. And, IAM role is an identity that has specific permissions with credentials that are valid for short durations. And roles can be assumed by entities that we trust. Since we trusted the devuser, we can switch roles. The picture was taken in the space needle of Seattle, I was up there over the summer.

Assuming roles encourage least privilege principle because it has permissions and privileges further divided. One can practice the least permission by assuming different roles.

Answer to task 6:
The role based policies refers to the policy that is attached to a specific role. In our case, devuser is our IAM role. And resource-based policy refers to the policy that is specific to certain resources, like S3 bucket in our case. It specifies actions allowed/disallowed associated with this resource. And by combining them, one can specify what role has what access to which resource.

Challenge task:
So based on the bucket resource policy, we know it has principles for another role called “OtherBucketAccessRole”. And we just had to assume that role to upload the image successfully. Other buckets are inaccessible because other buckets did not give permissions to this role.

Capital One Breach:
Errors in S3 securities policies affected the Capital One breach because of the misconfiguration along with human and non-human identities with excessive high-risk permissions. The advice I would give Capital One was to strictly practice the least privilege principle. Making sure access control is well monitored and regulated. The advice I’d give to AWS IAM is to have EC2 and other parts of services configured more rigorously to avoid things like misconfigured WAF.

What I did in this lab was following instructions in the AWS course. And explored how the permission and access control works in AWS. Something I found interesting was how thoroughly divided permissions are in AWS service.