Security - Identity and Access - Technical exercise

Using code (your preferred language), create three AWS IAM Users, one AWS IAM Role and two AWS IAM Policies meeting the following requirements:

- The two policies should be attached to the role.
- One policy should allow:
    - Read only - S3,
    - Full Access - EC2
- One policy should Deny:
    - IAM user creation
- The users you create should be able to assume the Role as long as they have an MFA authenticated session.
- Store all config in a S3 bucket so that you have a master copy.
- Create documentation that explains the steps required to achieve the end result.

Propose a solution for providing access for users in a large to medium size company's AWS environment.

Things to think about:

- How would it work with many users?
- Many Roles?
- Many Accounts?

Optional Questions (pick at most one, feel free to skip this question):

1. AWS

   Create an AWS Lambda function that checks which IAM roles a user can assume and outputs the results to a S3 bucket.

2. Active Directory

   Using your preferred language, please create a script that would interact with ActiveDirectory to:

    - Gets a list of all users in the domain with expired passwords
    - Creates a new Organizational Unit called BotUsers
    - Creates new user in the BotUsers OU called *bot_perfmon*
    - Adds this user to the built in *Performance Monitor Users* group

3. OKTA Development

   Using code (your preferred language/sdk):
   - print out a list of users email addresses in an OKTA tenant
   - print out a list of inactive user emails in an OKTA tenant
   - add a user to a group called “Okta_Admin”

4. React/Javascript
   - Create a *AWSAccountAliasRequest* component
     - Assume the *userName* is passed as props from its Parent Component
     - Create a field that accepts the AWS account ID (12 digits only)
     - Create a field that accepts the preferred alias (Allows a string of characters consisting of lowercase letters, digits, and dashes. Cannot start or finish with a dash nor can have two dashes in a row. Maximum length of 50)
     - Have a Submit button that displays the *userName*, the account ID, and the account alias on the screen when clicked.

5. Python

   Given a csv file of [server access logs]() for an S3 bucket, use Python to process the logs and extract the key information.
   - Extract the role arn and timestamp from each access log
     Example values:
     i. arn:aws:sts::043321448045:assumed-role/RougeDeveloperRole
     ii. 06/Feb/2019:00:00:38 +0000
   - Create a new csv file that contains the role arn and timestamp for all logs
   - You do not need to know anything about server access logs to complete this task
   - Please provide us with your complete code and the output csv file it generates from the sample csv file below
   - Sample input csv file (5 rows and 1 column, with one log per row):

```
"79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be awsexamplebucket1
[06/Feb/2019:00:00:38 +0000] 192.0.2.3 arn:aws:sts::043321448045:assumed-role/RougeDeveloperRole
3E57427F3EXAMPLE REST.GET.VERSIONING - ""GET /awsexamplebucket1?versioning HTTP/1.1"" 200 -
113 - 7 - ""-"" ""S3Console/0.4"" -
s9lzHYrFp76ZVxRcpX9+5cjAnEH2ROuNkd2BHfIa6UkFVdtjf5mKR3/eTPFvsiP/XV/VLi31234= SigV2
ECDHE-RSA-AES128-GCM-SHA256 AuthHeader awsexamplebucket1.s3.us-west-1.amazonaws.com
TLSV1.1"
"79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be awsexamplebucket1
[06/Feb/2019:00:00:38 +0000] 192.0.2.3 arn:aws:sts::043321448045:assumed-role/Administrator
891CE47D2EXAMPLE REST.GET.LOGGING_STATUS - ""GET /awsexamplebucket1?logging HTTP/1.1"" 200 -
242 - 11 - ""-"" ""S3Console/0.4"" -
9vKBE6vMhrNiWHZmb2L0mXOcqPGzQOI5XLnCtZNPxev+Hf+7tpT6sxDwDty4LHBUOZJG96N1234= SigV2
ECDHE-RSA-AES128-GCM-SHA256 AuthHeader awsexamplebucket1.s3.us-west-1.amazonaws.com
TLSV1.1"
"79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be awsexamplebucket1
[06/Feb/2019:00:00:38 +0000] 192.0.2.3 arn:aws:sts::043321448045:assumed-role/RougeDeveloperRole
```

```
A1206F460EXAMPLE REST.GET.BUCKETPOLICY - ""GET /awsexamplebucket1?policy HTTP/1.1"" 404
NoSuchBucketPolicy 297 - 38 - ""-"" ""S3Console/0.4"" -
BNaBsXZQQDbssi6xMBdBU2sLt+Yf5kZDmeBUP35sFoKa3sLLeMC78iwEIWxs99CRUrbS4n11234= SigV2
ECDHE-RSA-AES128-GCM-SHA256 AuthHeader awsexamplebucket1.s3.us-west-1.amazonaws.com
TLSV1.1"
"79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be awsexamplebucket1
[06/Feb/2019:00:01:00 +0000] 192.0.2.3 arn:aws:sts::043321448045:assumed-role/ReadOnly
7B4A0FABBEXAMPLE REST.GET.VERSIONING - ""GET /awsexamplebucket1?versioning HTTP/1.1"" 200 -
113 - 33 - ""-"" ""S3Console/0.4"" -
Ke1bUcazaN1jWuUlPJaxF64cQVpUEhoZKEG/hmy/gijN/I1DeWqDfFvnpybfEseEME/u7ME1234= SigV2
ECDHE-RSA-AES128-GCM-SHA256 AuthHeader awsexamplebucket1.s3.us-west-1.amazonaws.com
TLSV1.1"
"79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be awsexamplebucket1
[06/Feb/2019:00:01:57 +0000] 192.0.2.3 arn:aws:sts::043321448045:assumed-role/RougeDeveloperRole
DD6CC733AEXAMPLE REST.PUT.OBJECT s3-dg.pdf ""PUT /awsexamplebucket1/s3-dg.pdf HTTP/1.1"" 200 -
- 4406583 41754 28 ""-"" ""S3Console/0.4"" -
10S62Zv81kBW7BB6SX4XJ48o6kpcl6LPwEoizZQQxJd5qDSCTLX0TgS37kYUBKQW3+bPdrg1234= SigV4
ECDHE-RSA-AES128-SHA AuthHeader awsexamplebucket1.s3.us-west-1.amazonaws.com TLSV1.1"
```

Please share your work with us privately via email at sia-seniors@xero.com. Don't make your code public.