Practical Tasks SSM Parameter Store

**Task1 – Create App parameters in SSM Parameter Store (AWS console)**

We should create the following parameters for dev and prod environments of the test app:

* /my-app/dev/db-url
* /my-app/dev/db-password
* /my-app/prod/db-url
* /my-app/prod/db-password

1. Go to **Systems Manager** console and open **Parameter Store**. You can find it in **Application Management** section at the left panel menu.
2. Click on **Create parameter** in the right panel
3. Fill in **Parameter details**

* **Name** - /my-app/dev/db-url
* **Description** – Database URL for my app in Dev
* **Type** – String
* **Data type** – text
* **Value** – dev.database.dreamdb.org:3306

1. Click on **Create Parameter**
2. Now when the parameter has been created click on the parameter name. You can see Overview, History and Tags tabs that contain useful information about the parameter including parameter versions and their values.
3. Get back to **Parameter store**. Just click on the **Parameter Store** hyperlink at the top of the page.
4. Click on **Create parameter** and fill in Parameter details for the second parameter of the Dev environment:
   * + - * **Name** - /my-app/dev/db-password
         * **Description** – Database password for my app in Dev
         * **Type** – SecureString
         * **KMS key source** – My current account
         * **KMS key ID** – alias/aws/ssm
         * **Value** – DevPassword
5. Click on **Create parameter**
6. Create parameters for Prod env
7. Database URL

* **Name** - /my-app/prod/db-url
* **Description** – Database URL for my app in Prod
* **Type** – String
* **Data type** – text
* **Value** – prod.database.dreamdb.org:3306

1. Database password
   * + **Name** - /my-app/prod/db-password
     + **Description** – Database password for my app in prod
     + **Type** – SecureString
     + **KMS key source** – My current account
     + **KMS key ID** – alias/aws/ssm
     + **Value** – ProdPassword

We will use all these parameters in the next tasks

**Task2 – Work with App parameters stored in SSM Parameter Store (AWS CLI)**

1. Open **AWS CloudShell** window. You can find the **CloudShell** icon at the top of AWS console and click on it.
2. It can take some minutes until **CloudShell** is completely running and you can see Linux console.
3. In order to get Dev parameter type the following command in **CloudShell**:
   * *aws ssm get-parameters --names /my-app/dev/db-url /my-app/dev/db-password* and press **Enter**.

As you can see in the command output the db-password value is encrypted as we has use **SecureString** type for it.

1. In order to decrypt the db-password value use additional parameter --*with-decryption*:
   * *aws ssm get-parameters --names /my-app/dev/db-url /my-app/dev/db-password --with-decryption*
2. Another option to get all parameters under the path is to use the following command:
   * aws ssm get-parameters-by-path --path /my-app/dev

So you can see both db-url and db-password values in output

1. In order to get parameters recursively from several folders nested from the single path you can use the following command:
   * aws ssm get-parameters-by-path --path /my-app/ --recursive

In output you should be able to see all parameters for both Dev and Prod envs

**Task3 – Use App parameters stored in SSM Parameter Store with AWS Lambda**

1. Open AWS **Lambda** consolein and click on **Create function**
2. Fill in **Function name**, e.g. test-parameter-store
3. For Runtime choose **Python 3.8**
4. For Permissions choose **Create a new role with basic Lambda permissions**. At the bottom of this section you can find the AIM role name. Write down it as we will use it later in this task for giving additional permissions to Lambda for access to Parameter Store
5. Click on **Create function**
6. In the left panel of **Code source** find **lambda\_function.py** and click on it with the right button and then click on Open in the context menu.
7. Replace the default script with the following python script

import json

import boto3

region\_name = "<current\_region\_name>"

ssm = boto3.client("ssm", region\_name)

def lambda\_handler(event, context):

db\_url = ssm.get\_parameters(Names=["/my-app/dev/db-url"])

print(db\_url)

db\_password = ssm.get\_parameters(Names=["/my-app/dev/db-password"])

print(db\_password)

return "worked!"

1. Find your current AWS region (e.g. region=us-east-1) in the page URL and update **region\_name** variable in the pyton code with the real region value
2. Click on **Deploy**
3. Click on **Test** and select Create new test event
4. Fill in **Event name**, e.g. **mytes**t and click on **Create**
5. Click on **Test**
6. You should see an error **“AccessDeniedException when calling the GetParameter operation” because your Lambda is not authorized to perform this operation**. You need to fix this
7. Open **AWS IAM** console in a new tab of your browser.
8. Click on **Roles** in the left panel and enter the name of Lambda IAM role (created in 4-th step) in the Search field. Then open it.
9. Open **Permissions** tab and click on **Add inline policy**
10. Fill in policy parameters:

* Service=Systems Manager
* Action=Read: GetParameters, GetParameterbyPath
* Resources:(Add ARN) Region=Any, Account=Any, Parameter name= my-app/\*

1. Click on Review policy and check that everything looks good
2. Fill in the policy name – SSMAccessForMyApp and Then click on Create Policy
3. Get back to the browser tab with where you have Lambda console opened and Refresh the page
4. Click on **Test**
5. If everything is fine you can see **Response** “worked!” in **Execution results**
6. In **Function logs** you can see parameters with their values there. By the way “/my-app/dev/db-password” value is encrypted.
7. If you want to decrypt it, you should change the following line in the python script:

db\_password = ssm.get\_parameters(name=[“/my-app/dev/db-password”], WithDecryption=True)

1. Click on **Save**
2. Click on **Test** and you can see “worked!” message in **Execution results**
3. In **Function logs** section you can find that /my-app/dev/db-password value is decrypted now.