Store your environment variables with **AWS Secret Manager**

AWS Secret Manager, Advantages, Disadvantages, Examples

AWS Secret Manager

The WHAT:

- A fully managed service to store and retrieve secrets securely
- Supports rotation, auditing, and fine-grained access control
- Integrates easily with AWS services and custom applications

The WHY:

- Centralized management of secrets
- Very easy integration with AWS services
- Automated secret rotation
- Secure and encrypted storage

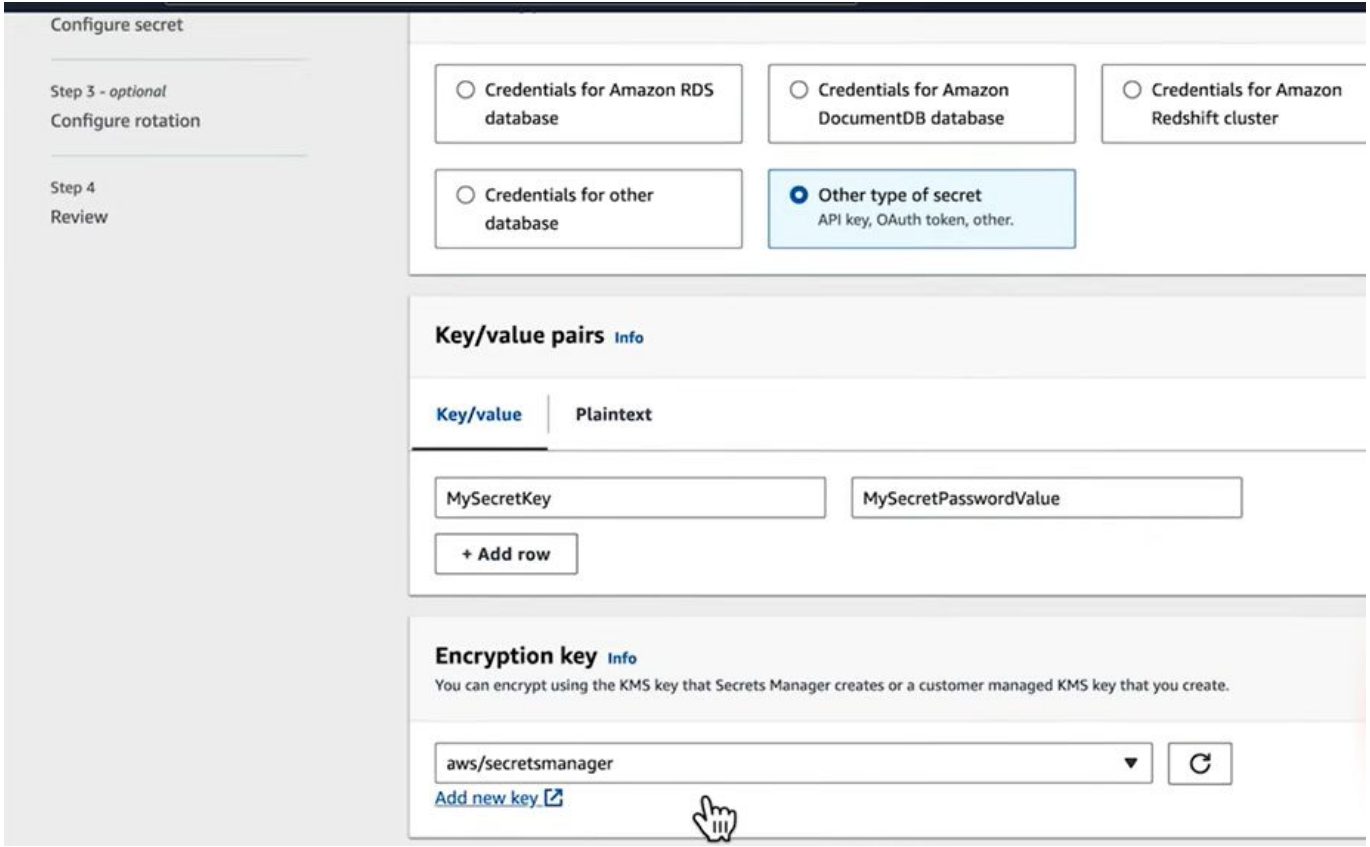
Advantages and disadvantages of using AWS Secret Manager vs local storage of secrets

AWS Secret Manager	Local storage .env
- Secrets encrypted at rest and in transit. ✅ - Granular access control with IAM. ✅	- `.env` or config files can be leaked through: ❌ • Git commits • Server misconfiguration - Easy to use and create✅
- Seamless integration with AWS services ✅ - Secure and encrypted storage (KMS-based) ✅ - Extra costs ❌	- Need to copy every time the secrets to new services (Servers, CI pipelines, Environments) ❌ - No extra costs ✅
- All access and updates logged in CloudTrail (must for compliance (SOC2, ISO, HIPAA) ✅ - Supports rotation for RDS, Redshift, and custom apps ✅ - Deeply tied to AWS services ❌	- No log by default of accessing the secrets ❌ - No automatic rotation of leaked/expired secrets ❌ - Fast read time. Secrets are loaded directly from disk or environment variables ✅

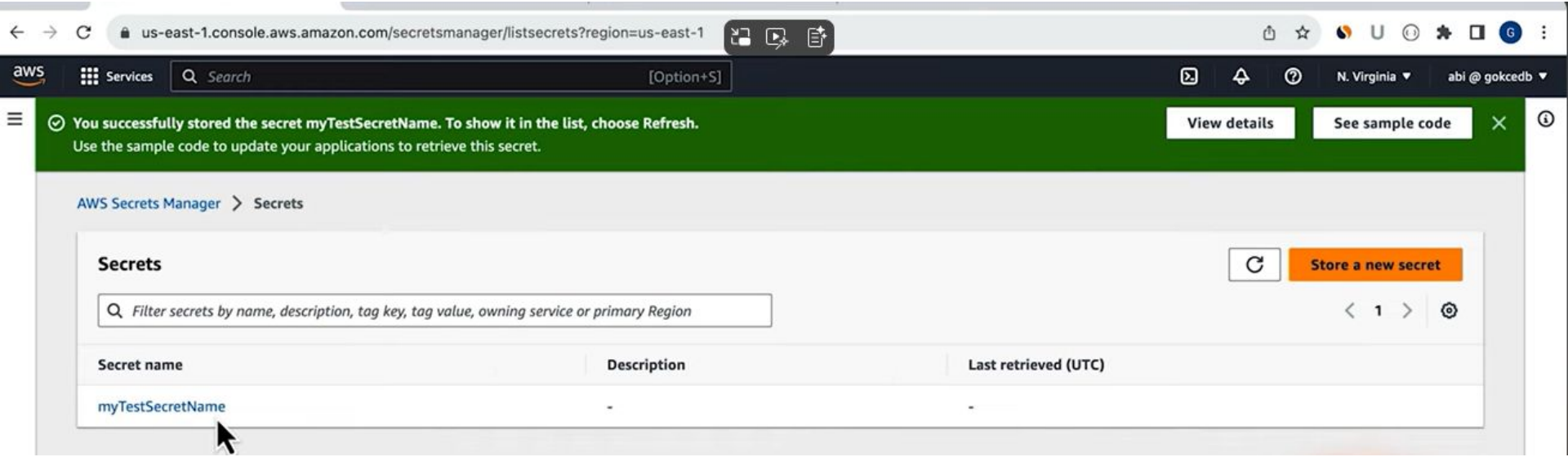
DEMO

Let’s create a new secret and consume it

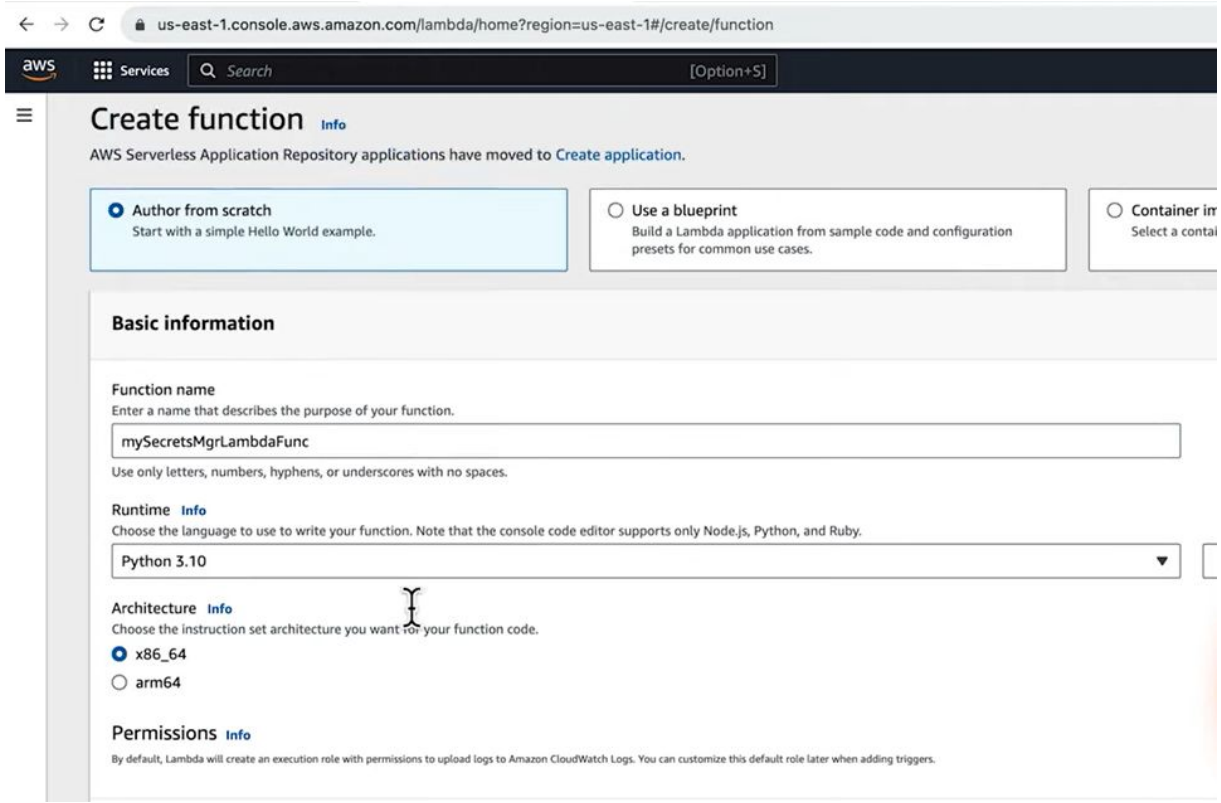
Select the secret type



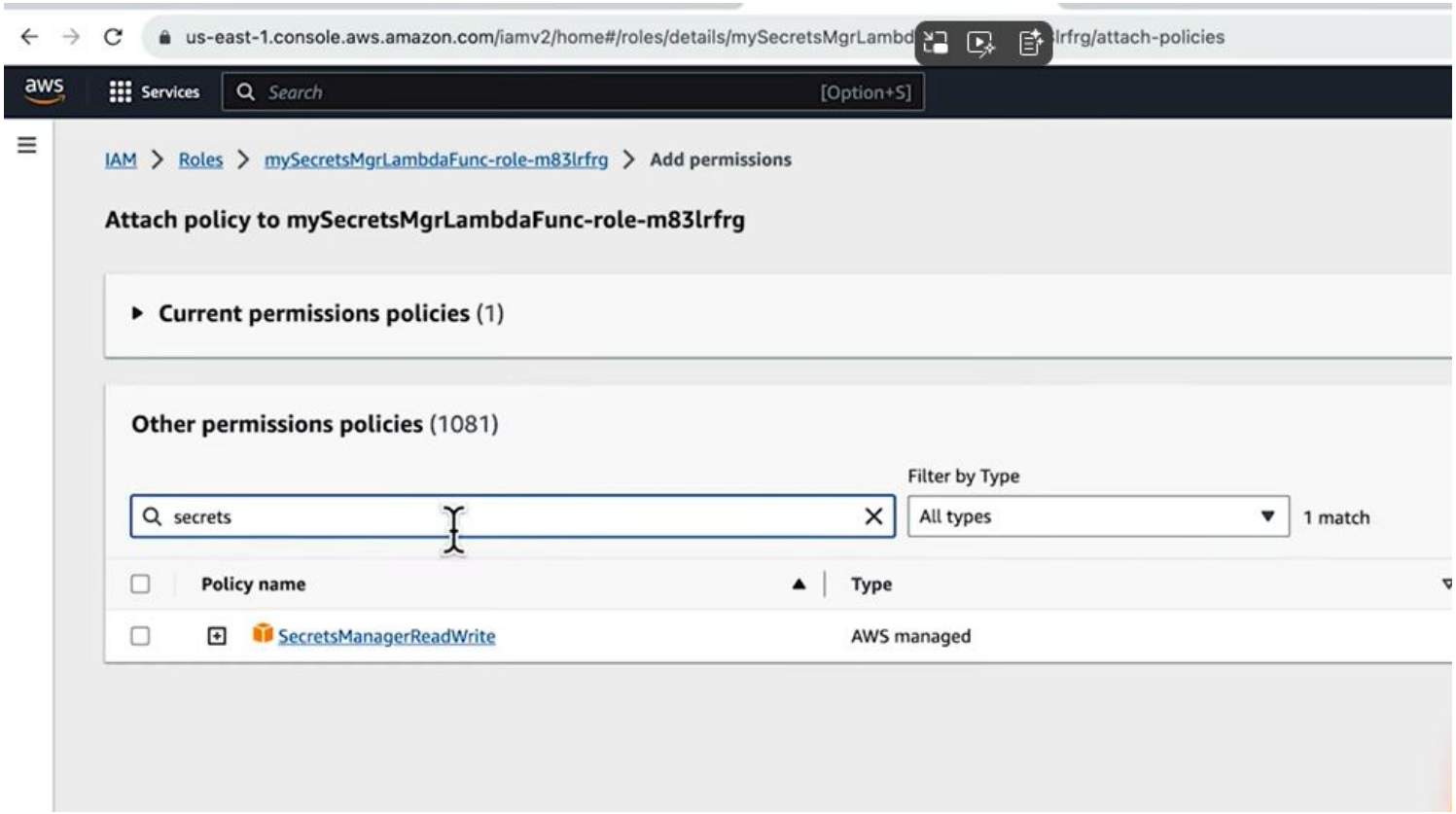
Check if secret was created successfully



Create an AWS Lambda function with Python



Add permissions to use the Secrets Manager



Access the secret

```python
import boto3
from botocore.exceptions import ClientError

def lambda_handler(event, context):
    secret_name: str = "myTestSecretName"
    secret_json = get_secret(secret_name)
    print(secret_json)

    return {
        'statusCode': 200,
        'body': json.dumps("Hello from lambda")
    }

def get_secret(secret_name):
    region_name: str = "us-east-1"

    session = boto3.session.Session()
    client = session.client(
        service_name='secretsmanager'
        region_name=region_name
    )

    try:
        secret_resp = client.get_secret_value(
            SecretId=secret_name
        )
    except ClientError as e:
        raise e

    secret = get_secret_value_response('SecretString')

    return secret
```

Run and see the result