AWS Cloud engineer

→ Skills assessment test

Cloudfoundation team

Skills assessment framework

This skills assessment test is designed to evaluate **your knowledge**, **understanding and experience on AWS cloud services** that we're leveraging at Allianz-Trade, the uses cases are real world and tailored for Allianz-Trade ecosystem and cloud strategy.

The assessment consists of scenarios-based open questions, responses could take different forms: Architecture diagrams, technical description in bullet points, technical code (terraform/python) or any other response shape or structure.

Produced code could be either provided as an attachment to the email or published on Github/Gitlab public repo.

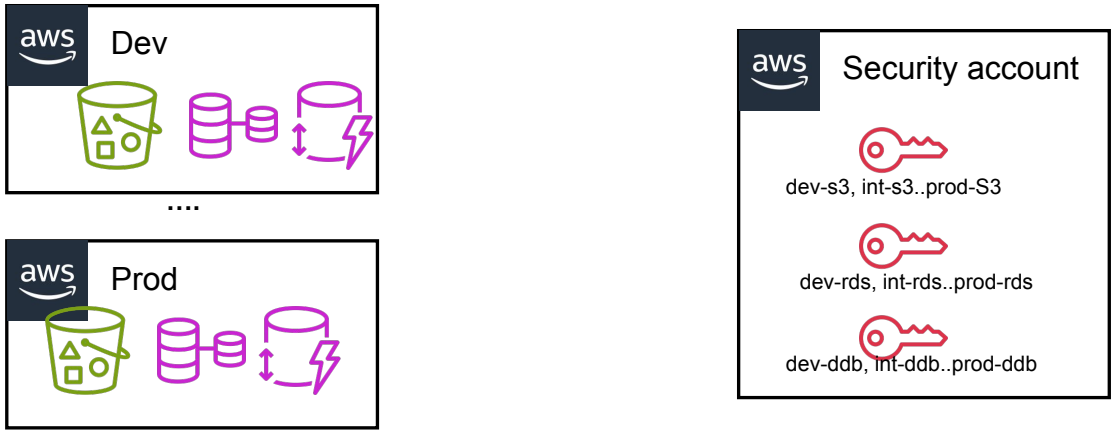
IMPORTANT

As a reminder, this file contents are intended for you only and should not be disclosed as confidential. We kindly request that you do not forward, copy, or share this file or any content of it with anyone else without our prior written consent.

Encryption management Key rotation on AWS

Scenario#1

A new requirement from our regulators dictates that we must apply a rotation on all of our KMS keys, the following diagram and technical details illustrate how we're implementing encryption on the cloud



Technical details

- Keys are managed on AWS KMS in a dedicated account, with external type (BYOK) generated on our HSM hosted onpremise
- Encryption is segregated by environment and service, each environment has it's own key for each service
- Keys are using key alias
- key policy is implementing least privilege principle

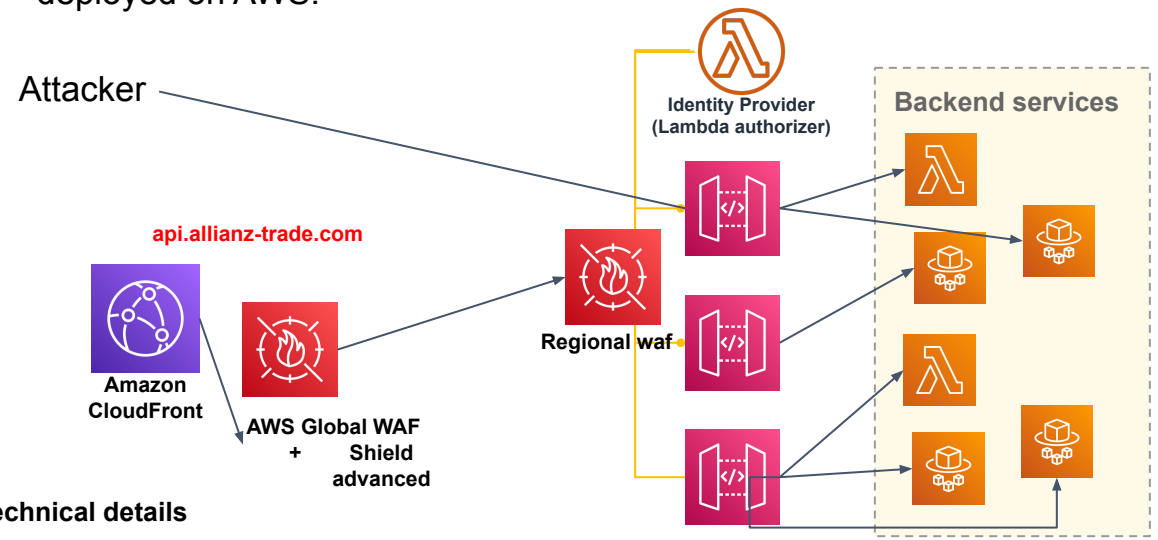
Given the architecture and the technical details, suggest a solutions/response to the following questions/requirements

1. What are the main challenges to apply key rotation ? and what impacts you can identify ?
2. From your respective, what are the steps of applying key rotation (high level description)
3. After applying the rotation on keys, we're required to have a monitoring on the resources to identify - at any given time - resources (rds, dynamodb, S3) that are not complaints (**resources where rotation is not applied**) how could we achieve this requirement with AN aws managed services ?
4. What's the best way to secure key material during their transportation from HSM to AWS KMS ?

APIs-as-a-Product Public and private APIs

Scenario#2

We're leveraging AWS to build and ship our APIs, both for internal usage (API-driven integration between applications) and public usage (customers, brokers..Etc). The following schema depicts how our APIs are deployed on AWS.



Technical details

- All the APIs are "by design" public (even those who are not used publicly)
- APIs developed and maintained by different teams, but exposed through **unique endpoint (api.allianz-trade.com**
- All the APIs are protected with a global AWS WAFv2 and shield-advanced
- Backend microservices are either lambda functions or internal ALBs backing ECS fargate microservices

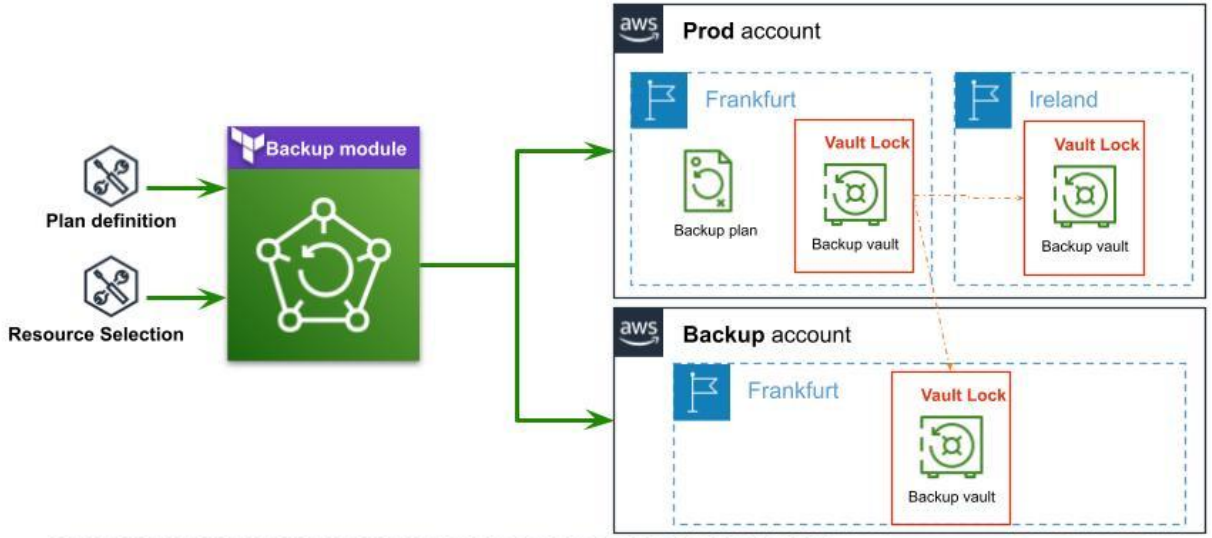
Given the architecture and the technical details, suggest a solutions/response to the following questions/requirements

1. What weaknesses can you see on the current architecture ?
2. We would like to change the exposition of these APIs. APIs intended for internal usage must go private, and APIs for both internal and external usage must be exposed internally and externally, hence, internal calls (from our network) must no longer go through internet -> cloudfront -> apigw -> backend service. what would the new architecture be, simple efficient and with the less impact possible ?
3. In the current architecture how cloudfront could be configured to route traffic to multiple APIGWs based on path (path-based routing) ?
4. If we want to protect our regional APIGW endpoints (which are public) from traffic that "Bypass" the cloudfront/WAF and directly reach APIGW endpoints, what would you suggest as a solution ?

Backup policy Laveraging AWS Backup

Scenario#4

A new requirement dictates to implement a cloud backup policy on AWS using AWS Backup service, automation is key when it comes to deploying at a scale the backup policy, cloudfoundation team came up with a design validated by security, compliance and architecture team illustrated underneath:



technical details & requirements

Plan definition

- Backup frequency
- Backup retention
- Backup encryption

Resource selection

- All supported resources with `ToBackup=true Owner=<owner@eulerhermes.com.com`

X-region/X-account copy

- Enable Cross-Region with defined frequency, retention & key
- Enable Cross-account with defined frequency, retention & key

WORM protection

- Enable Vault Lock to prevent malicious & accidental backup deletion

Given the architecture and the technical details and the requirements, <u>implement a terraform module that meets all the requirements.</u>

the module does not necessarily need to be production-ready. the purpose is only to validate your terraform skills and knowledge.