**Links**

AWS Landing Zone

<https://aws.amazon.com/solutions/aws-landing-zone/>

AWS Organizations

<https://aws.amazon.com/organizations/>

AWS Account

<https://aws.amazon.com/account/>

AWS SCP

<https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_scp.html>

Example Service Control Policies

<https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_example-scps.html>

AWS Calculator

<https://calculator.s3.amazonaws.com/index.html>

Dia-03 – Links

AWS Well-Architected Tool - <https://aws.amazon.com/well-architected-tool/>

AWS Well-Architected Doc - <https://aws.amazon.com/architecture/well-architected/>

The 5 Pillars of the AWS Well-Architected Framework - <https://aws.amazon.com/blogs/apn/the-5-pillars-of-the-aws-well-architected-framework/>

Lesson:

Se você já usa AWS, escolha uma aplicação, um serviço específico e tente rodar o Well-Architected nele. Não publique os resultados, use isso dentro da empresa.

Leia cada paper do Well-Architected que está na doc(veja em Links) e entenda mais degtalhes de cada Pilar do framework

Execute na sua conta pessoal da AWS o Well-Architected, criando um workload inicial para medir a qualidade de cada pilar na sua conta inicial.

Vá durante o treinamento, criando milestones, e vendo a evolução da sua conta.

No final do treinamento, vamos rever cada pilar juntos.

AWS IAM Best Practices - <https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html>

Access keys best practices - <https://medium.com/@ashishrajan/aws-security-best-practices-access-keys-cloudsecurity-facb20aa0db6>

AWS CLI - <https://aws.amazon.com/cli/>

## Today's Lesson

Crie um user com AdministratorAccess, adicione ele em grupo de Administrators. Ative MFA Virtual ou Físico para este usuário. Não utilize mais o usuário Root(email da conta) desta Account. Use uma senha de 64 caracteres no mínimo, e utilize um gerenciador de senhas. Ex.: 1Password, LastPass. Se você já tem uma conta que usa durante muito tempo, altere a senha da sua e ative MFA na Root Account. Instale o aws-cli, o utilitário linha de comando da AWS: https://aws.amazon.com/cli/ Crie um usuario do tipo Programmatic Access no IAM e attach a policy ReadOnlyAccess Apos instalar o aws-cli, configure ele usando: $ aws configure Doc em <https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-configure.html#cli-quick-configuration>

Cli:

PS C:\Users\Lucas> aws iam create-group --group-name ReadOnlyCli

{

"Group": {

"Path": "/",

"GroupName": "ReadOnlyCli",

"GroupId": "AGPASXNA3BY43H2YFIOHF",

"Arn": "arn:aws:iam::187705265721:group/ReadOnlyCli",

"CreateDate": "2022-01-23T14:45:28+00:00"

}

}

PS C:\Users\Lucas> aws iam create-user --user-name readUserCli

{

"User": {

"Path": "/",

"UserName": "readUserCli",

"UserId": "AIDASXNA3BY4TXNEZ5F7U",

"Arn": "arn:aws:iam::187705265721:user/readUserCli",

"CreateDate": "2022-01-23T14:49:53+00:00"

}

}

aws iam add-user-to-group --user-name readUsercli --group-name ReadOnlyCli

aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess --group-name

PS C:\Users\Lucas> aws iam create-access-key --user-name readUsercli

{

"AccessKey": {

"UserName": "readUserCli",

"AccessKeyId": "AKIASXNA3BY4VEFS2NU7",

"Status": "Active",

"SecretAccessKey": "tW/mt1qKpp/QjHY32MPdm9yzFuOFsQNvoYW55LdD",

"CreateDate": "2022-01-23T15:12:50+00:00"

}

}

Dia 5 – IAM

Credential Reports - <https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_getting-report.html?icmpid=docs_iam_console>

Identity Providers and Federantion - <https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers.html>

Habilitando ADFS para AWS - <https://aws.amazon.com/blogs/security/enabling-federation-to-aws-using-windows-active-directory-adfs-and-saml-2-0/>

Cross Account using IAM Roles - <https://docs.aws.amazon.com/IAM/latest/UserGuide/tutorial_cross-account-with-roles.html>

Exercicio:

## Today's Lesson

O exercício de hoje é para encerrar essa nossa primeira etapa de Foundation na AWS.

Você deve criar uma conta na AWS para ser responsável pela gestão de indentidade da sua Organization.

Basicamente um conta com IAM configurado. Ou seja, os usuarios, grupos, policies e roles estarão nessa conta.

As outras duas contas que criamos desde o início do curso, você vai utilizar para receber autenticações vindo dessa conta que vamos chamar de "Identity Account".

O cenario no final, vai ficar assim: 1 Master account 1 Identity account - usuarios e grupos criados aqui 2 Accounts com Cross Account Roles, ou seja, Roles do tipo Account, que vão autorizar usuarios vindo da conta "Identity", acessa-las através de Assume Role(Swith Role) Em uma conta, a Role da permissão de PowerUserAccess e em outra conta, ReadOnlyAccess. Execute acessos a essas contas com um usuario do IAM da Identity Account, fazendo Swith Role.

EC2

Amazon EC2:

https://aws.amazon.com/pt/ec2/

Tipos e preços:

https://aws.amazon.com/pt/ec2/pricing/

Amazon EBS:

https://aws.amazon.com/pt/ebs/

Montar o EBS no Linux:

https://docs.aws.amazon.com/pt\_br/AWSEC2/latest/UserGuide/ebs-using-volumes.html

Amazon Security Group:

https://docs.aws.amazon.com/pt\_br/AWSEC2/latest/UserGuide/ec2-security-groups.html

**AWS EC2 Metadata:** <https://docs.aws.amazon.com/pt_br/AWSEC2/latest/UserGuide/ec2-instance-metadata.html>

**Snapshot:** <https://docs.aws.amazon.com/pt_br/AWSEC2/latest/UserGuide/ebs-creating-snapshot.html>

**AMIs:** <https://docs.aws.amazon.com/pt_br/AWSEC2/latest/WindowsGuide/AMIs.html>

**Placement Groups:** <https://docs.aws.amazon.com/pt_br/AWSEC2/latest/WindowsGuide/placement-groups.html>

**Lifecycle Manager:** <https://docs.aws.amazon.com/pt_br/AWSEC2/latest/UserGuide/snapshot-lifecycle.html>

VPC

VPC => <https://docs.aws.amazon.com/pt_br/vpc/latest/userguide/VPC_Subnets.html>

Calculadora Subnet => <http://www.subnet-calculator.com/>

Internet Gateway => <https://docs.aws.amazon.com/pt_br/vpc/latest/userguide/VPC_Internet_Gateway.html>

Route Tables => <https://docs.aws.amazon.com/pt_br/vpc/latest/userguide/VPC_Route_Tables.html>

VPN Client-site

**VPN**

<https://docs.aws.amazon.com/pt_br/vpn/latest/s2svpn/SetUpVPNConnections.html>

<https://docs.aws.amazon.com/pt_br/vpn/latest/s2svpn/VPC_VPN.html>

<https://docs.aws.amazon.com/pt_br/vpn/latest/s2svpn/VPNRoutingTypes.html>

<https://docs.aws.amazon.com/pt_br/vpn/latest/s2svpn/VPNTunnels.html>

**VPC Quotas**

<https://docs.aws.amazon.com/vpc/latest/userguide/amazon-vpc-limits.html>

RDS & DynamoDB

O primeiro exercício é pra fortalecer os conhecimentos de RDS.

Crie um rds, free tier, com o nome de wordpress, usando a engine MySQL

O serviço que vamos subir em uma EC2, será um Wordpress, aquela engine para Blog e gerenciador de conteúdo.

Faça o setup do Wordpress, seguindo a doc oficial aqui https://wordpress.org/support/article/how-to-install-wordpress/

Você vai editar o arquivo de config do Wordpress, como está na doc, e apontar para o seu MySQL rodando no RDS, com host, porta, usuário e senha.

Como o RDS está sobre SSL/TLS, você precisa instalar esse certificado na sua EC2, de onde vai partir os request para o banco.

O segundo exercício é sobre DynamoDB.

Crie uma tabela com o nome de "hosts", com as propriedades de name(primary key), ip(sort key) e outras propriedades que você acha necessário para guardar informações de servers, VMS, instâncias, etc.

Toda a interação com essa tabela, sera feita usando Python, como no mundo real, como uma aplicação de verdade.

Se você quiser utilizar outra linguagem, fique a vontade, minha sugestão é usar Python!

Você vai utilizar a doc oficial do Dynamo, aqui https://docs.aws.amazon.com/pt\_br/amazondynamodb/latest/developerguide/GettingStarted.html

Tem SDK(Software Development Kit) para várias linguagens.

Seu código precisa conseguir, recuperar todos os items da tabela, criar um novo item, recuperar um item específico usando o ip como parametro de busca e outra função/método que busque por name do host. E por último, um trecho de código para deletar um item usando o name como parâmetro.

Publique no github esse exemplo de código com a integração do DynamoDB.

Boa sorte!

----

AWS Config

Conformance Packs

<https://docs.aws.amazon.com/config/latest/developerguide/conformance-packs.html>

Custom Rules

<https://docs.aws.amazon.com/config/latest/developerguide/evaluate-config_develop-rules.html>

Examples Custom Rules

<https://github.com/awslabs/aws-config-rules>

AWS Config Multiple Accounts

<https://aws.amazon.com/blogs/aws/aws-config-update-aggregate-compliance-data-across-accounts-regions>

Using terraform for authorization and aggregator

<https://www.terraform.io/docs/providers/aws/r/config_aggregate_authorization.html>

<https://www.terraform.io/docs/providers/aws/r/config_configuration_aggregator.html>

SNS

O exercicio que vamos fazer, vai ser um publicador(pub) em Python, vai enviar mensagens em um tópico.

As mensagens enviadas lá, serão pedidos de clientes do nosso e-commerce.

Voce vai publicar um json lá, e o subscription vai ser um endpoint https(sub) em Python, para processar essa mensagem,

e dar sequência no pedido de compra do cliente no nosso e-commerce.

O documento json deve conter essas chaves:

order\_id,

client\_id,

shipment\_id,

status

User a criatividade na hora de publicar e criar esse document json :)

Use a documentação oficial para escrever o pub e o sub - em PT-BR aqui https://docs.aws.amazon.com/pt\_br/sns/latest/dg/sns-http-https-endpoint-as-subscriber.html

e EN-US aqui https://docs.aws.amazon.com/sns/latest/dg/sns-http-https-endpoint-as-subscriber.html

Leia outros códigos no github pra te ajudar nessa trabalho, mas lembre-se, quebre um pouco a cabeça antes!

E claro, mande pra mim ou no grupo se precisar de alguma coisa!

Abraços e bons estudos!

mprado