
# AWS - Material de Apoio
# Matheus Luis Oliveira da Silva

https://opus-aws-estags.signin.aws.amazon.com/console

▼ Definição do projeto

**Projeto 3: Cloud AWS**

Tópicos de estudo

- Cloud (história, conceito, 5 características de NIST, grandes players)
- Cloud vs Virtualização
- Cloud vs Virtual hostings/VPS
- AWS (história/origem, formas de acesso: console, sdk, cli)
- Recursos (conceito e prática)
- VPC (VPC, subnets, nat gateway, internet gateway, nacl, rede pública, rede privada)
- EC2 (Security Group, AMI, EBS, Elastic IP, Load Balancers (classic, alb, nlb, gateway, diferenças e caso de uso), autoscaling groups, tipos de instâncias)
- RDS (Tipos, Parameter groups, Snapshots)
- S3
- CloudWatch

- IAM (usuários, grupos, roles, policies)

Projeto final do estudo documentado

- Fazer uma apresentação explicando o que entendeu sobre os tópicos de estudo. Questionamentos diversos e aleatórios serão feitos sobre os tópicos.
- Apresentar uma solução que faça a integração entre 3 serviços diferentes da AWS

▼ Referências:

- https://www.alura.com.br/artigos/cloud
- https://www.ibm.com/blogs/cloud-computing/2016/08/23/a-brief-history-of-cloud-computing-2/
- https://www.ibm.com/topics/iaas-paas-saas
- https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf
- https://www.simplilearn.com/aws-stats-article
- https://www.redhat.com/pt-br/topics/cloud-computing/cloud-vs-virtualization
- https://www.hostdime.com/blog/cloud-vs-vps/
- https://www.hostinger.com/tutorials/what-is-vps-hosting
- https://www.udemy.com/course/aws-certified-solutions-architect-associate-saa-c03/learn/lecture/26097978#overview
- https://jayendrapatil.com/aws-global-vs-regional-vs-az-resources/
- https://www.gocache.com.br/dicas/para-que-serve-jenkins-e-o-que-ele-pode-fazer/
- https://github.com/ahmedtariq01/Cloud-DevOps-Learning-Resources/blob/main/AWS Learning/AWS EC2 Foundations.pdf

▼ Cloud

▼ História

- Em conceito computação em nuvem ja era trabalhada desde a decada de 50

- Na época, já existiam sistemas de multi-usuário para grandes mainframes.
- Os primeiros sistemas time-sharing surgiram na década de 60, nos primórdios do compartilhamento de recursos via rede
- Utility computing foi o termo cunhado por John McCarthy, um visionário do fornecimento da computação como um serviço para usuários comuns
- Em 70, a virtualização revolucionou o sitema de acesso compartilhado
- Na década de 90, as empresas de telecomunicação começaram a utilizar a virtualização para prover serviços com acesso compartilhado a uma mesma infraestrutura, não mais uma conexão ponto a ponto. (Cloud!)
- Em 97 o termo Cloud Computing é tratado pela primeira vez pelo professor Ramnath Chellappa

- IBM:
  - **Grid computing:** Solving large problems with parallel computing
  - **Utility computing:** Offering computing resources as a metered service
  - **SaaS:** Network-based subscriptions to applications
  - **Cloud computing:** Anytime, anywhere access to IT resources delivered dynamically as a service

▼ Conceito

▼ NIST

- *National Institute of Standards and Technology*
- "*Cloud Computing é um modelo de computação que provê um conjunto compartilhado de recursos de computação customizáveis como redes, servidores, armazenamento, aplicações e serviços."*
- Para o NIST, um sistema de cloud precisa se enquadrar em 5 caractetísticas, são elas:

    - On-Demand Self-Service, ou seja, o consumidor deve poder garantir seus recursos de computação sem a interação com algum agente humano, tudo automaticamente via serviço.
    - Broad network access, ou seja, o serviço de cloud deve estar disponível por toda a rede de internet, e acessível via qualquer dispositivo conectado a ela.
    - Resource Pooling, ou seja, os recursos computacionais devem estar disponíveis para servir múltiplos usuários ao mesmo tempo, garantindo a gerência dos recursos físicos ou virtuais dinamicamente sob a demanda do usuário. A localização exata do recurso em hardware físico não é disponibilizada, sendo fornecida pelo provedor apenas uma região mais abstrata, como o país ou região do país.
    - Rapid Elasticity, ou seja, todos os recursos provisionados devem estar a disposição do usuário ilimitadamente, podendo ser escalados ou descendidos conforme demanda automaticamente, em qualquer quantidade a qualquer momento.
    - Measured Service, ou seja, todos os recursos e serviços provisionados devem ser devidamente mensurados e relatados para o usuário, bem como possuir medidas conhecidas para administração tanto por parte do provedor quanto do usuário final.
- Formações possíveis:
    - Private cloud: disponível num âmbito local, uma empresa por exemplo, podendo estar instalada fisicamente dentro ou fora do recinto.
    - Community cloud: uma expansão em curta escala da private cloud, adicionando usuários de toda uma comunidade específica para essa cloud, um conglomerado de empresas, por exemplo.
    - Public cloud: a maior expansão possível, uma vez que a cloud deve estar disponível para todo o público em geral, sendo gerenciada por uma ou mais entidades empresariais, acadêmicas ou governamentais.

    - Hybrid cloud: uma combinação de duas ou mais formações de cloud possíveis, sendo gerenciada por uma ou mais entidades proprietárias, garantindo a conexão entre elas e fornecendo serviços de interconexão, como portabilidade de aplicações por exemplo.

▼ IaaS, ou infrastructure as a service:

- Acesso sob demanda a recursos de computação de servidor, armazenamento e rede via internet.
- O gerenciamento do hardware cabe ao provisionador do serviço de cloud
- O usuário pode administrar os recursos contratados via internet.
- Fornece varias vantagens para o usuário como:
    - Alta disponibilidade
    - Baixa latência, aumento da performance sob demanda
    - Melhor responsibilidade, sem necessidade de administração de hardware físico
    - Rápido acesso a melhores tecnologias
- Amazon Web Services, Google Cloud, IBM Cloud, Microsoft Azure

▼ PaaS, ou platform as a service:

- Agora, o provedor do serviço gerencia não só o hardware, mas tambéms alguns softwares disponíveis para os usuários:
    - servidores para teste desenvolvimento e implantação, sistemas operacionais, armazenamento, networking, banco de dados, middlewares, runtimes, frameworks, development tools, backup tools, etc…
- Geralmente possuem uma interface gráfica para os grupos de usuários utilizarem e compartilharem dos ambientes de desenvolvimento disponíveis.
- Fornece varias vantagens para o usuário como:
    - Rápido desenvolvimento
    - Facilitação no teste e implantação de novas tecnologias

        - Colaboração entre os usuários simplificada
        - Escalabilidade e gerenciamento reduzido
    - Um middleware conhecido e muito utilizado como PaaS é o wordpress, por exemplo.
    - AWS Elastic Beanstalk, Google App Engine, Microsoft Windows Azure, and Red Hat OpenShift on IBM Cloud

▼ SaaS, ou software as a service:

- Diferente dos demais, esse serviço entrega uma aplicação pronta como serviço.
- Toda a parte de servidor, armazenamento, networking, middleware e aplicação ficam sob responsabilidade do fornecedor.
- Toda a parte de atualização também fica a cargo do fornecedor, e todo o controle de uso da ferramenta pode ser administrada pelo usuário mediante cobrança proporcional ao aumento na usabilidade.
- Hoje, a maioria de nós somos usuários de SaaS, como DropBox, Emails, Trello, Slack, Teams, Canva, etc…
- Interessante notar que até mesmo alguns softwares que antes eram instalados na máquina local, como o pacote Adobe, hoje são fornecidos como SaaS, com o Adobe Creative Cloud.
- Fornece varias vantagens para o usuário como:
    - Alta escalabilidade, crie novas contas e pronto. Exemplo: nossas contas da AWS
    - Basta uma conta para utilizar de qualquer lugar.
    - Risco mínimo, testes facilitados para averiguar a eficácia da ferramenta.

▼ Casos de uso

O uso de qualquer um desses serviços traduz o uso de cloud, no qual, dependendo do objetivo da empresa contratante, cada um pode atender diferentes casos de uso, como:

- Escolher uma solução Saas, abrindo mão do controle de todas as funcionalidades e características da ferramenta utilizada, como

armazenamento de dados e segurança de tráfego. Permitindo as vezes até mesmo abrir mão de uma equipe de TI.

- Escolher uma solução PaaS, construindo sua própria aplicação, sendo responsável agora por todas as suas características, abrindo mão somente do controle e gerenciamento na fase de desenvolvimento da aplicação. Nesse caso, a equipe de TI é necessária para a criação da ferramenta.
- Escolher construir uma aplicação utilizando recursos de backend via IaaS, assim, reassumindo o controle de praticamente toda a aplicação desenvolvida bem como do ambiente de desenvolvimento, utilizando o serviço somente como uma terceirização de recursos para a sua ferramenta. Aqui a equipe de TI é indispensável, inclusive capacitada o suficiente para administrar esses novos recursos com boa eficiência.

▼ Grandes Players

- Amazon Web Services: 2006
  - Detinha, segundo publicação em 2017, mais de 33% dos serviços de IaaS e PaaS do mercado;
  - Segundo CSA, cloud security alliance, esse número cresceu para 41,5%.
- Microsoft Azure: 2008
  - Segundo CSA, detém 29% do mercado.
  - Vem crescendo no mercado, sendo o concorrente principal no mesmo nicho de tecnologia da AWS, focado em web-services.
- Google Cloud: 2008
  - Possui apenas 3% do mercado segundo CSA.
  - Focado em um nicho mais fechado, de sistemas pensados para cloud desde o princípio. Não parece se preocupar com expansão almejando chegar no nível da AWS.
- SoftLayer 2005, sendo, um dia, a majoritária do mercado → IBM Cloud, 2013
  - Possui menos de 3% do mercado segundo CSA.

▼ Cloud vs Virtualização

- Virtualização é a tecnologia que especificamente permite que um recurso computacional seja particionado e virtualizado em múltiplos recursos com multíplos fins diferentes.
- Cloud, por sua vez, está para o agrupamento, abstração e fornecimento de recursos já virtualizados sob demanda para usuários finais.
- Ou seja, os serviços de cloud usam de virtualização para particionar os recursos físicos de hardware dos provedores para os particionar e disponibilizá-los ao consumidor.

▼ Cloud vs Virtual hostings/VPS (Virtual Private Server)

- Ambos possuem um set-up parecido, em termos de hardware e funcionalidades, porém diferem muito em como são implantados e gerenciados.
- Um VPS possui, geralmente, menos máquinas a disposição do usuário, esse que, por sua vez, terá acesso a uma fatia virtualizada do hardware físico que o provedor disponibiliza.
- Com o que diz respeito a um serviço de cloud, o usuário terá acesso a recursos provindos de diversos servidores físicos, que estarão interligados e orquestrados para garantir o maior uptime possível.
- Assim, pode-se diferir serviços de VPS e Cloud através das diferenças na confiabilidade, escalabilidade, automatização no provisionamento de recursos e até mesmo na cobrança e medição de seu uso.

▼ AWS

▼ História

- 2002 → A equipe de TI lançou o que viria a ser a AWS internamente, sob os domínio da amazon.
- 2003 → Nota-se a capacidade de prestação de serviços de TI para o público externo, mediante a qualidade e potencial da equipe e dos serviços.
- 2004 → Lançamento do primeiro serviço de SQS, Simple Query Service, para o público externo à amazon.
- 2006 → Houve o relançamento, agora com serviços de SQS, S3 (SImple Storage Service) e EC2 (Elastic Compute Cloud). Assim, a

AWS sai na frente já provisionando um serviço de mensagens, de armazenamento e de computação.

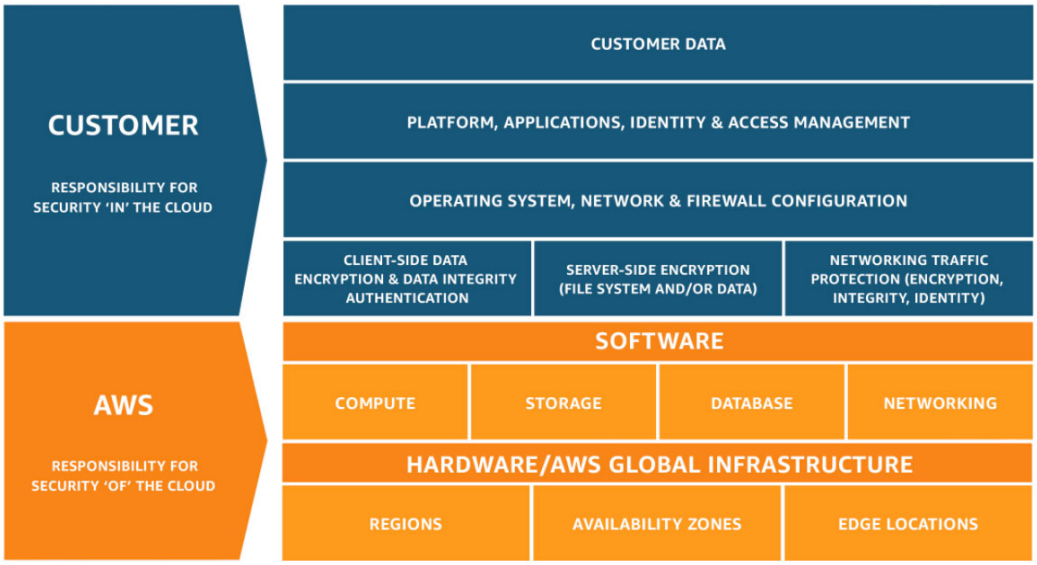
- 2007 → lançamento na Europa e crescimento acelerado para o mercado mundial: McDonald's, netflix, activision, etc…

▼ Estrutura:

- https://aws.amazon.com/pt/
- A AWS é um serviço de nuvem cujos hosts físicos estão espalhados por todo o globo.
- São 245 países atendidos, com 31 regiões
- Os serviços são divididos em regiões, como US East, US West, Asia Pacific, etc.
- Regiões basicamente são clusters de data centers e servidores.
- Cada região possui várias AZs (Availability Zones), que são data centers lógicos que possuem independência entre si, garantindo maior disponibilidade e confiabilidade, ou seja, se uma AZ sofrer um desastre, a cloud conseguirá manter o sistema de pé sem perda de dados.
- A conexão entre AZs de uma mesma região é de baixíssima latência, justamente para que haja um comportamento de cluster e uma abstração da separação geográfica do hardware utilizado.
- As chamadas Edge locations são data centers menores que se encontram nas "bordas" das AZs, servindo como um entreposto de cache de dados para garantir menor latência até os usuários.

▼ Segurança:

- Ao pensar em segurança na AWS, é preciso separar a responsabilidade da AWS e do usuário.
- A AWS como empresa é responsável por toda a parte da infraestrturua de hardware da nuvem, os data centers, os servidores e todos os componentes de hardware são de responsabilidade da AWS.
- A partir de qualquer serviço instanciado, a responsabilidade passa a ser do usuário, tendo algumas ressalvas quando se tange alguns softwares, como o software de armazenamento, ou de banco de dados por exemplo, de responsabilidade da AWS.




▼ Serviços:

- A maioria dos serviços estão disponíveis em regiões ou AZs específicas, com exceção daqueles que são globais (IAM, Route53, CloudFront, etc…)
- A cobrança por cada serviço é de acordo com o uso do usuário, sem cobrar por períodos de ociosidade.
- Entendendo a importância da cobrança pelos recursos da AWS e como usá-los corretamente.

**Table 1.1 How an AWS bill changes if the number of web shop visitors increases**

| Service | January usage | February usage | February charge | Increase |
|---|---|---|---|---|
| Visits to website | 100,000 | 500,000 | | |
| CDN | 25 M requests + 25 GB traffic | 125 M requests + 125 GB traffic | $115.00 USD | $100.00 USD |
| Static files | 50 GB used storage | 50 GB used storage | $1.15 USD | $0.00 USD |
| Load balancer | 748 hours + 50 GB traffic | 748 hours + 250 GB traffic | $19.07 USD | $1.83 USD |
| Web servers | 1 virtual machine = 748 hours | 4 virtual machines = 2,992 hours | $200.46 USD | $150.35 USD |
| Database (748 hours) | Small virtual machine + 20 GB storage | Large virtual machine + 20 GB storage | $133.20 USD | $105.47 USD |
| DNS | 2 M requests | 10 M requests | $4.00 USD | $3.20 USD |
| Total cost | | | $472.88 USD | $360.85 USD |

▼ AWS Networking Services

- Virtual Private Cloud
  - VPC – Regional

        - VPCs are created within a region
    - Subnet – **Availability Zone**
        - A subnet can span only a single Availability Zone
    - Security groups – Regional
        - A security group is tied to a region and can be assigned only to instances in the same region.
    - VPC Endpoints – Regional
        - VPC Gateway & Interface Endpoints cannot be created between a VPC and an AWS service in a different region.
    - VPC Peering – ~~Regional~~
        - VPC Peering can be performed across VPC in the same account of different AWS accounts ~~but only within the same region. They cannot span across regions~~
        - VPC Peering can now span inter-region
    - Elastic IP Address – Regional
        - Elastic IP addresses created within the region can be assigned to instances within the region only.
    - Elastic Network Interface – Availability Zone
    - Route 53 – **Global**
        - Route53 services are offered at AWS edge locations and are global
    - CloudFront – **Global**
        - CloudFront is the global content delivery network (CDN) services are offered at AWS edge locations
    - ELB, ALB, NLB, GWLB – Regional
        - Elastic Load Balancer distributes traffic across instances in multiple Availability Zones in the same region
        - Use Route 53 to route traffic to load balancers across regions.
    - Direct Connect Gateway – Global

        - is a globally available resource that can be created in any Region and accessed from all other Regions.
    - Transit Gateway – Regional
        - is a Regional resource and can connect VPCs within the same AWS Region.
        - Transit Gateway Peering can be used to attach TGWs across regions.
    - AWS Global Accelerator – Global
        - is a global service that supports endpoints in multiple AWS Regions.

# AWS Compute Services

- EC2
    - Resource Identifiers – Regional
        - Each resource identifier, such as an AMI ID, instance ID, EBS volume ID, or EBS snapshot ID, is tied to its region and can be used only in the region where you created the resource.
    - Instances – **Availability Zone**
        - An instance is tied to the Availability Zones in which you launched it. However, note that its instance ID is tied to the region.
    - EBS Volumes – **Availability Zone**
        - Amazon EBS volume is tied to its Availability Zone and can be attached only to instances in the same Availability Zone.
    - EBS Snapshot – Regional
        - An EBS snapshot is tied to its region and can only be used to create volumes in the same region and has to be copied from one region to another if needed.
    - AMIs – Regional

        - AMI provides templates to launch EC2 instances
        - AMI is tied to the Region where its files are located with Amazon S3. For using AMI in different regions, the AMI can be copied to other regions
    - Auto Scaling – Regional
        - Auto Scaling spans across multiple Availability Zones within the same region but cannot span across regions
- Cluster Placement Groups – **Availability Zone**
    - Cluster Placement groups can span across Instances within the same Availability Zones
- ECS – Regional

# AWS Storage Services

- S3 – Global but Data is Regional
    - S3 buckets are created within the selected region
    - Objects stored are replicated across Availability Zones to provide high durability but are not cross-region replicated unless done explicitly.
    - S3 cross-region replication can be used to replicate data across regions.
- DynamoDB – Regional
    - All data objects are stored within the same region and replicated across multiple Availability Zones in the same region
    - Data objects can be explicitly replicated across regions using cross-region replication
- DynamoDB Global Tables – Across Regions
    - is a new **multi-master, cross-region replication** capability of DynamoDB to support data access locality and regional fault tolerance for database workloads
- Storage Gateway – Regional

        - AWS Storage Gateway stores volume, snapshot, and tape data in the AWS region in which the gateway is activated

# AWS Identity & Security Services

- Identity Access Management – IAM
    - Users, Groups, Roles, Accounts – **Global**
        - Same AWS accounts, users, groups, and roles can be used in all regions
    - Key Pairs – **Global** or Regional
        - EC2 created key pairs are specific to the region
        - RSA key pair can be created and uploaded that can be used in all regions
- Web Access Firewall – WAF – **Global**
    - protect web applications from common web exploits and is offered at AWS edge locations globally.
- AWS Config – Regional
- AWS GuardDuty – Regional
    - findings remain in the same Regions where the underlying data was generated.
- Amazon Detective – Regional
- Amazon Inspector – Regional
- Amazon Macie – Regional
    - must be enabled on a region-by-region basis and helps view findings across all the accounts within each Region.
    - verifies that all data analyzed is regionally based and doesn’t cross AWS regional boundaries.
- AWS Security Hub – Regional.
    - supports cross-region aggregation of findings via the designation of an aggregator region.
- AWS Migration Hub – Regional.

        - runs in a single home region, however, can collect data from all regions

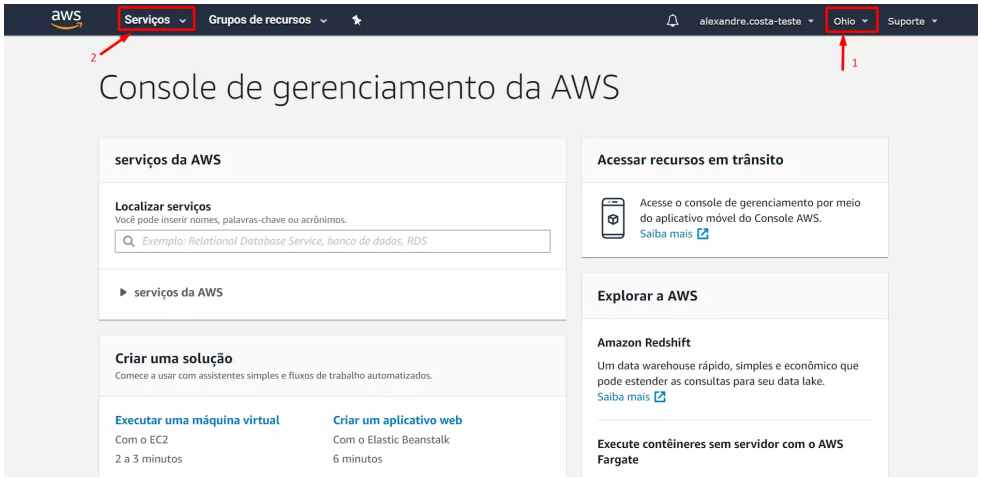
- A escolha da região a se trabalhar é muito importante por alguns fatores:
    - Compliance, ou seja, adequação as normas governamentais ou institucionais do seu país e/ou aplicação;
    - Proximity, ou seja, quanto mais próxima a região de hospedagem da sua aplicação com seus usuários finais, menor a latência em seu uso;
    - Disponibilidade de serviços, pois nem todas as regiões possuem todos os serviços;
    - Preço, claro, pois cada região tem uma tabela de preços diferente umas das outras.

▼ Formas de Acesso

- Existem algumas formas de se acessar a API da AWS para administrar os recursos de uma aplicação em nuvem, são elas:

    ▼ Console:

    - O console da AWS oferece uma interface amigável para que o usuário consiga ter acesso rápido a qualquer serviço oferecido pela AWS.
    - Pela praticidade, é muito utilizado para ambientes de teste e desenvolvimento.
    - Como segurança, esse método oferece senhas em combinação com multi factor authentication, MFA.



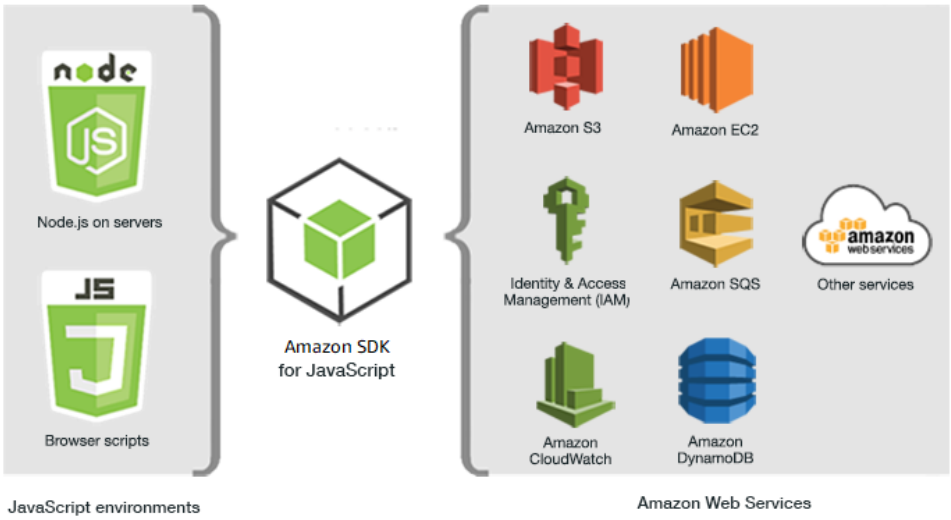

▼ CLI:

- É possível também acessar todos os recursos da AWS via terminal, com a AWS CLI.
- Com ela, é possível automatizar algumas tarefas, daí a sua importância, fora a praticidade de se usar o terminal para quem está habituado.
- Com ela também é possível criar infraestruturas a partir de blueprints, adicionar objetos a uma já existente via arquivos e/ou gerenciar e monitorar uma infraestrutura e rede em execução.
- Uma ótima opção para serviços de integração e entrega contínuas, como jenkins, por exemplo, pois há facilitações ao se trabalhar com a CLI em ambientes que se utilizam dessas metodologias, facilitando a administração da infraestrutura da aplicação para contínuas melhorias e manutenções.
- O Jenkins, em especial, já saiu na frente com forte integração com tecnologias de cloud e virtualização, kubernetes, containers docker e serviços de nuvem como a AWS, por exemplo.
- O CLI pode ser acessado através de Access Keys, que são geradas através do console para posterior login no terminal.

```
~ aws ec2 describe-instances
{
    "Reservations": [
        {
            "Groups": [],
            "Instances": [
                {
                    "AmiLaunchIndex": 0,
                    "ImageId": "ami-01893222c83843146",
                    "InstanceId": "i-0d9c1f5ccff0d8314",
                    "InstanceType": "t3.nano",
                    "LaunchTime": "2022-02-07T14:04:39+00:00",
                    "Monitoring": {
                        "State": "disabled"
                    },
                    "Placement": {
                        "AvailabilityZone": "us-east-1a",
                        "GroupName": "",
                        "Tenancy": "default"
                    },
                    "PrivateDnsName": "ip-172-31-74-160.ec2.internal",
                    "PrivateIpAddress": "172.31.74.160",
                    "ProductCodes": [],
                    "PublicDnsName": "ec2-3-237-171-227.compute-1.amazonaws.com",
                    "PublicIpAddress": "3.237.171.227",
                    "State": {
```

▼ SDK:

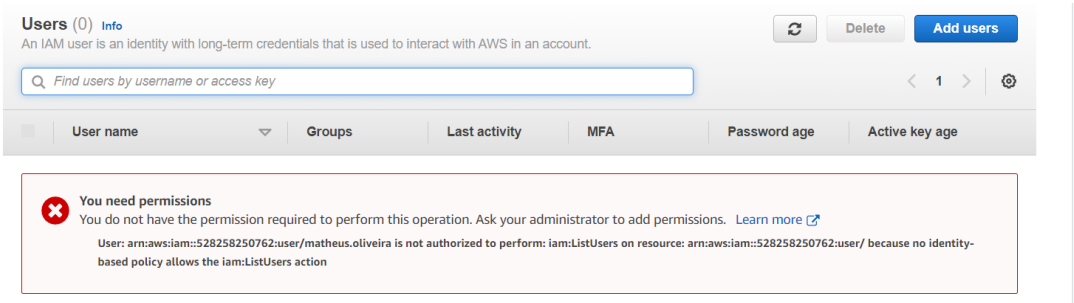
- Para aplicações que possuem integração com serviços da AWS é necessário se utilizar das SDK's.
- As SDK's (software development kits) conectam as API's de diversas tecnologias, como javascript, python, php, java, c++, Go, NodeJS, etc… com a API da AWS




▼ IAM

▼ Usuarios, Grupos e Policies

- IAM → Access Management → Users/Policies
- É um serviço Global no AWS, Identity and Access Management
- Uma conta Root é criada para que administre as contas de acesso a nuvem.
- Num time, existem vários **usuários**, que podem pertencer a um ou mais **grupos**, porém um grupo jamais contém grupos dentro dele.
- Através de JSONs documents, pode-se criar uma **police**, a qual define todas as permissões que um determinado usuário ou grupo possui dentro da nuvem.




- Uma policy pode ser anexada a um grupo, ou a um usuário diretamente. Isso faz com que usuários adicionados a determinados grupos, tenham,

por herança, todas as policies anexadas ao(s) seu(s) grupo(s), além das policies direcionadas a ele diretamente.

- Os **Roles,** por sua vez, se destinam aos serviços, ou seja, são “papeis” que os próprios serviços da AWS podem precisar para acessar determinados recursos e realizar determinadas ações.
    - Por exemplo, se um EC2 precisa acessar todos os s3 existentes na nuvem. Para isso, o desenvolvedor deve criar uma Role para EC2, que permita full acesso aos serviços s3. Depois, basta anexar essa Role em sua instância em execução.

▼ Segurança do IAM

- Password Policy:
    - IAM → Account Settings → Chance Password Policy
    - Esse é talvez o jeito mais simples de se aumentar a segurança de uma conta na AWS, uma vez que com esse componente é possível especificar restrições para a administração da senha dos usuários, obrigar que os usuários as altere e melhore, se for necessário.
- MFA, Multi Factor Authentication:
    - Account → My Security Credentials → MFA → Activate
    - Diferente de apenas se usar uma password, o usuário combina a sua senha com um token de autenticação para o dispositivo que está usando.
    - Para ter acesso a conta com MFA, então, o usuário deve saber a sua senha e autenticar seu dispositivo no momento do login.
    - Existem aplicativos de terceiros que permitem a geração de keys para a autenticação como Google authenticator ou Authy, por exemplo. Também é possível adquirir um hardware físico para que seja feita a autenticação.
    - O MFA é ativado pelo próprio usuário, no entanto é possível garantir que todos os usuários de uma conta ativem o seu MFA
    - https://docs.aws.amazon.com/singlesignon/latest/userguide/how-to-configure-mfa-device-enforcement.html
- IAM-identity-center

        - Como sucessor do aws SSO, o IIS tem como objetivo centralizar a gestão de usuários de uma organização num único serviço.
        - Nele é possível administrar permissões e restrições para os usuários de uma mesma conta, bem como impor regras para o acesso a cloud.
        - Com ele, assim como o single-sign-on, também é possível centralizar num só lugar os logins dos usuários a diversas aplicações.

▼ Recursos

- Elastic BeanStalk: Esse serviço tem como propósito facilitar o deploy e scaling de aplicações WEB. A partir do código da aplicação, o EBS criará a infraestrutura necessária para manter a aplicação no ar, com load balancing, auto scaling, monitoramento etc. Ele não tem tarifa, a cobrança é feita a partir dos recursos que ele próprio cria.
- CloudFormation: segue a mesma linha do EB, porém com outra forma de deployment de infraestrutura, sendo essas vi arquivo de texto ou ate mesmo uma ferramenta visual de montagem da infra.
- CodeCommit: Esse serviço se assemelha ao github, tem a exata mesma proposta, porém tudo hospedado na estrutura de cloud da AWS.
- ECS: Ferramenta da AWS para lidar com containers docker. Opção do Fargate launch type garante a infraestrutura do cluster automatizada pela AWS, enquanto a EC2 launch type confere ao usuário a configuração de suas máquinas e de seu cluster, baseado em instâncias de EC2.

▼ VPC

▼ VPC

- a AWS Virtual Private Cloud é uma porção isolada da nuvem publica da AWS.
- É usada para criar redes privadas para recursos de EC2.
- É possível conectar a rede privada, com todos os seus serviços e topologia interna configurados, com a internet com uma subnet exposta e um internet gateway, sem comprometer a estrutura interna privada.
- Geralmente uma estrutura de multi-VPC é criada para separar os recursos em desenvolvimento, produção, teste, etc.

▼ Subnets

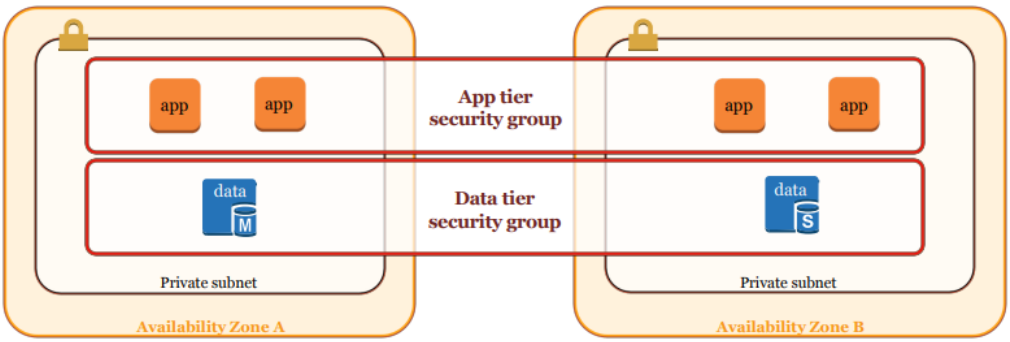
- Uma subnet é um conjunto de endereços IPs de uma VPC, divididos por um range CIDR.
- A subnet private é destinada aos recursos que não serão expostos na internet, ou seja, é voltada para databases, processos backend, armazenamentos, etc.
- A publica, por outro lado, terá exposição para a internet, sendo voltada basicamente para web servers.
- Cada subnet deve estar contida em somente uma AZ.

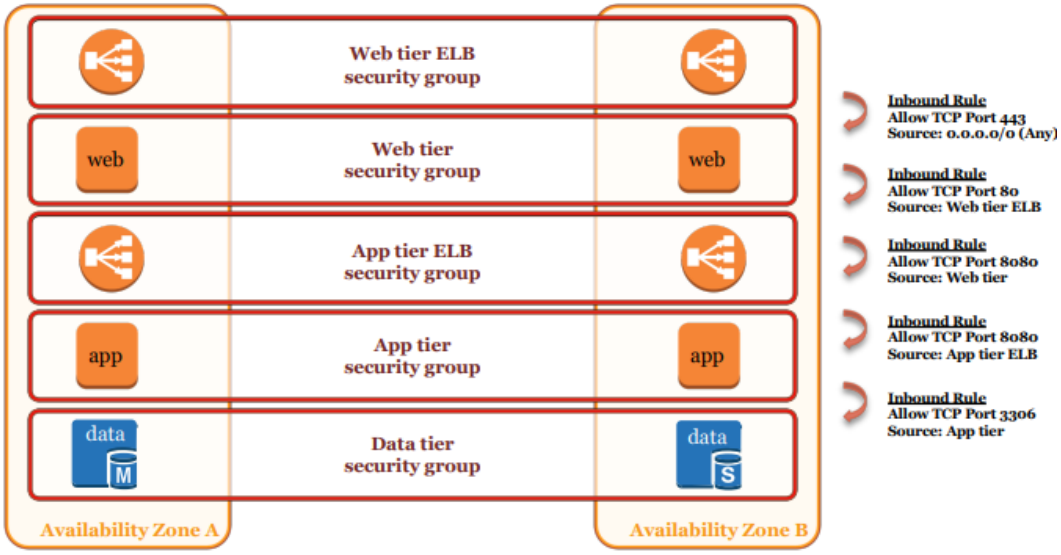
▼ Route Tables

- As RTs possuem todas as rotas de tráfego de rede, pareando IPs de saída com seus respectivos destinos.
- A VPC por padrão possui uma RT principal, podendo ter outras RTs customizadas.
- Cada subnet pode ter apenas uma RT.

▼ Security groups

- Funciona como um firewall virtual que controla o tráfego de entrada e saída dentre os recursos.
- Uma requisição que é autorizada automaticamente possui sua resposta autorizada também.
- São definidos blocos de endereço de instancias ou outros security groups para descrever as permissões de tráfego.
- Pode-se, por exemplo, separar os Apps dos servidores de dados.






▼ Network ACLs

- As network access control lists permitem ou negam o acesso a nível de subnet.
- As NACLs recebem as requisições do roteador e definem se podem ou não entrar, ou sair das subnets as quais a requisição se destina.
- As principais diferenças entre security groups e NACLs são:

| **Security group** | **Network ACL** |
|---|---|
| Operates at the instance level | Operates at the subnet level |
| Applies to an instance only if it is associated with the instance | Applies to all instances deployed in the associated subnet (providing an additional layer of defense if security group rules are too permissive) |
| Supports allow rules only | Supports allow rules and deny rules |
| Evaluates all rules before deciding whether to allow traffic | Evaluates rules in order, starting with the lowest numbered rule, when deciding whether to allow traffic |
| Stateful: Return traffic is allowed, regardless of the rules | Stateless: Return traffic must be explicitly allowed by the rules |

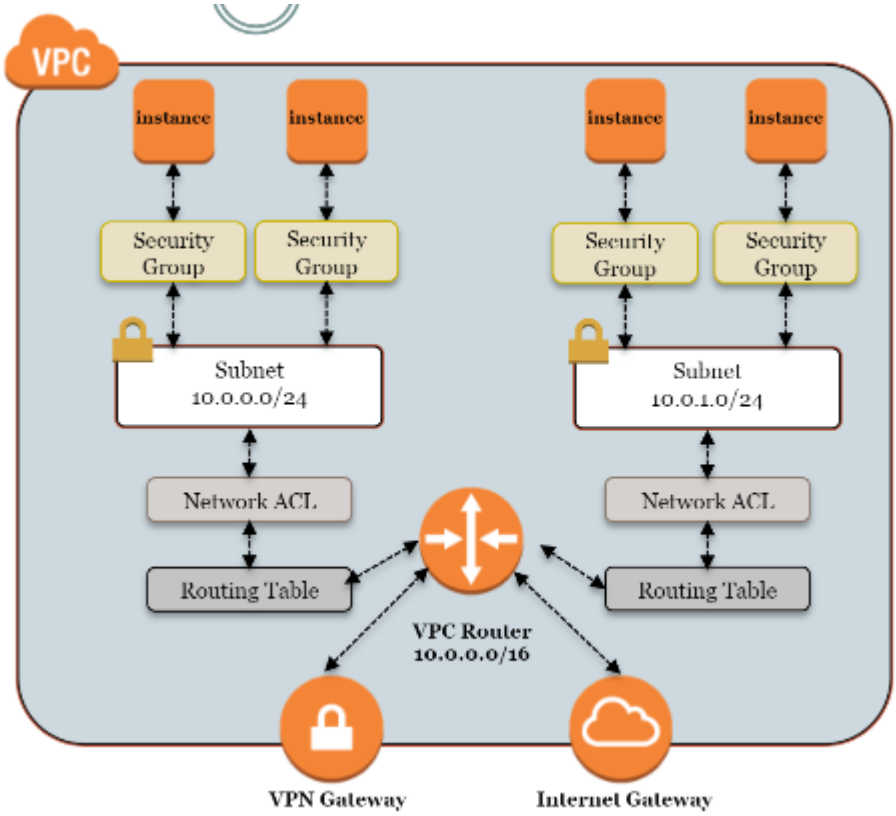
▼ internet gateway e nat gateway

- São a porta de acesso para que instancias possam se comunicar com a internet.
- Um nat gateway é usado para expor uma instancia de um asubnet privada para a internet, sem comprometer sua segurança, uma vez que a rota passará, ou por uma instancia EC2 configurada como NAT que está numa subnet pública, ou então por um VPC NAT gateway.
- Ambas as opções são válidas, mas possuem algumas diferenças, são elas:

| | VPC NAT gateway | NAT instance |
|---|---|---|
| Availability | Highly available by default | Use script to manage failover |
| Bandwidth | Bursts to 10 Gbps | Based on bandwidth of instance type |
| Maintenance | Managed by AWS | Managed by you |
| Security | NACLs | Security groups and NACLs |
| Port forwarding | Not supported | Supported |

▼ Resumo:

- Uma VPC garante para o desenvolvedor ferramentas para proteção de rede da sua aplicação, com vários componentes.
- Os componentes para criação de uma VPC robusta são:
  - VPC em si
  - Subnets
  - Route Tables, Security Groups e NACLs
  - Virtual Private Gateway
  - AWS Direct Connect
  - Internet Gateway
  - Elastic IPs e Load Balances

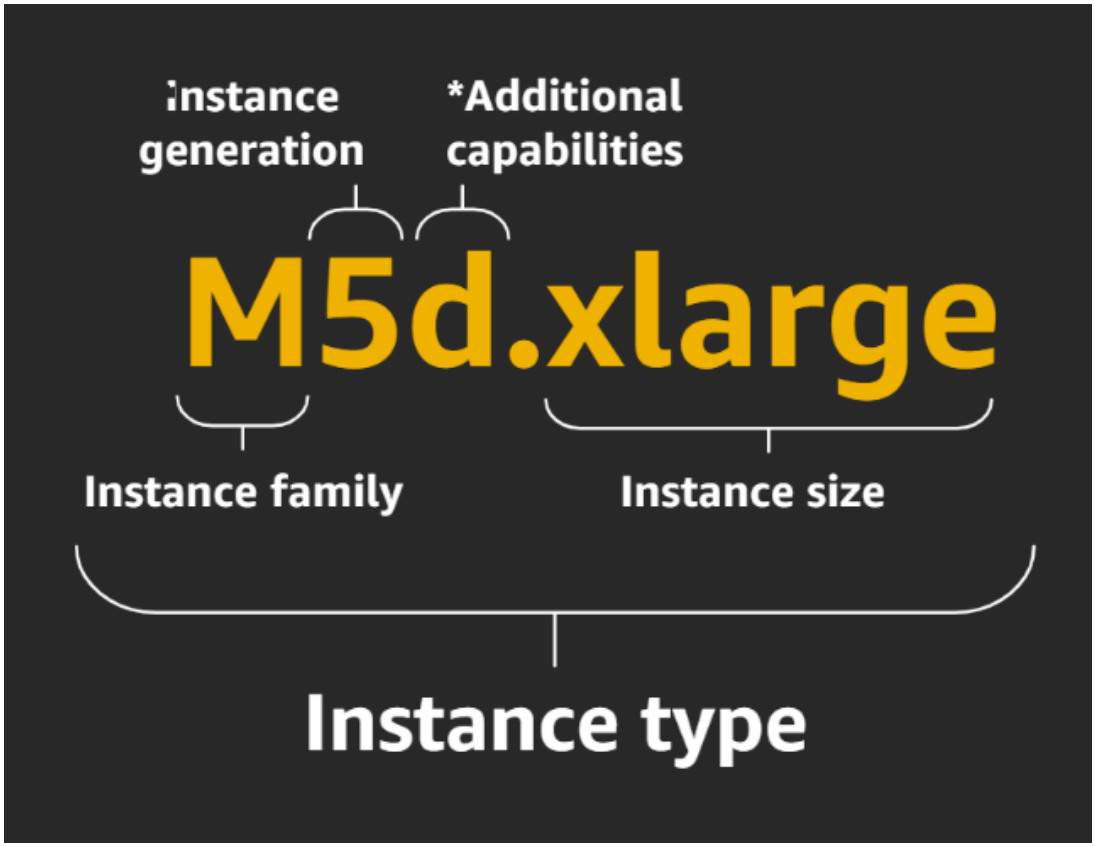



▼ EC2

- Elastic Compute Cloud, são instancias de máquinas virtuais disponibilizadas pela AWS.
- A cobrança é feita sob demanda, ou seja, quanto mais recursos usar, maior será a tarifa.

▼ Tipos de Instancia:

- Os tipos de instância são extremamente variados, sendo identificados por um código conforme demonstra a imagem




- O tipo fornece algumas “dicas” de quais características a sua instância terá, como sua familia, o seu tamanho, geração ou características adicionais:

  ▼ Famílias:

  - R: RAM aprimorada, máquinas com grande quantidade de memória RAM, para fins de alto processamento em memória.
  - C: CPU aprimorada, máquinas com grande capacidade de processamento, voltadas a computação e/ou banco de dados.
  - M: Máquinas balanceadas, voltadas para uso geral, aplicações web, serviços, etc.
  - I: Voltada para banco de dados, máquinas que precisam de um grande fluxo de input e output.
  - G: GPU aprimorada, voltada a multiprocessamento em placa de vídeo, como renderização de vídeos, aplicações de IA, etc.
  - T2/T3:
    - São as burstable instances, que garantem um uso de CPU base conhecido, porém permitem um overload de CPU

acima da base, até um certo limite.

- Esse limite é atingido através do uso de créditos de burst, ou seja, quando os créditos de overload é atingido, a CPU reduzirá massivamente seu desempenho, adicionando novos créditos com o tempo após ela retornar ao seu estado padrão.
- Ideal para aplicações que possuem períodos ociosos em contraste com picos de uso.
- Caso constantemente a instancia utilizada processe com poucos créditos disponíveis, fica evidente a necesisdade de selecionar uma máquina mais adequada para esse uso.
- Existem também ambas as versões sem o limite superior. Porém a cobrança pode ser inesperada, dado que é lastreada no número de créditos utilizados, sem aviso prévio.

▼ Tamanho:

- O tamanho das instâncias também possui um padrão. As vCPUS disponíveis são calculadas com o nThreads*nCores*CPUS do host.
- Instancias da mesma família possuem o mesmo vCPU ratio umas das outras.
- Entretanto, entre um tamanho e outro, a diferença da quantidade de memoria da CPU e do armazenamento dobra de acordo com a mudança de tamanho.
- Ao planejar uma nova infraestrutura ou um upgrade em uma existente é possível usar uma calculadora para solicitar um limite maior de vCPUs para o uso.

▼ Pricing:

- On Demand: As de uso mais comum e menos pré-definidas, ou seja, uma vez que são instanciadas, a sua cobrança é feita por hora e todos os recursos utilizados por ela podem ser alterados mediante cobrança proporcional.

- Reserved Instances: Diferente das On Demand, o usuário deve previamente definir os limites de uso de recursos da sua aplicação, bem como o período de tempo a qual ocupará esses recursos. Um ponto positivo é o desconto na cobrança desse tipo de instância, dado devido a previsão do uso, provavelmente.
- Spot Instances: Instancias Spot são interessantes para workloads que sejam tolerantes a interrupções, uma vez que a AWS fornece esse recurso a preços baixos porém sob a chancela de interrrompê-lo caso algum outro usuário venha a requisitá-lo pelo preço cheio. Ainda assim é interessante pelo grande desconto oferecido, podendo ser usadas como um excedente para um pico do auto-scaling, ou então para alguma tarefa sem deadline rígido, por exemplo.
- Dedicated hosts: Uma máquina para uso exclusivo do usuário, facilitando os trâmites de licenciamento de determinados softwares que são vinculados ao servidor de instalação.

▼ Security Group:

- Security Groups são um conjunto de firewalls para limitar o acesso de rede a sua máquina.
- Com esse serviço é possível criar uma lista de protocolos, portas e IP's os quais serão autorizados a interagirem com a instância criada.
- Com a devida autorização, é possível acessar a instância criada via SSH, por exemplo, com o uso da chave criada.
- Para acessar via SSH:
  - Alterar a visibilidade da chave: chmod 400 chave.txt
  - Conectar via ssh: ssh -i "chave.txt" ec2-x-x-x….amazonaws
- Para troca de arquivos via SCP:
  - Instalar SCP na instancia, caso não exista: `sudo yum install -y openssh-clients`
  - Enviar o arquivo do host para a instancia: `scp -i /path/key-pair-name.pem /path/SampleFile.txt instance-user-name@my-instance-public-dns-name:~`
- Em ambos o casos, um passo de verificação de segurança pode ser feito, comparando as fingerprints da instância, evitando um ataque man-

in-the-middle.

    - Primeiro é preciso coletar via CLI, as fingerprints da instância desejada com: `aws ec2 get-console-output --instance-id` *`instance_id`* `--output text >` *`temp.txt`*
    - Depois, é preciso comparar as chaves com a chave mostrada no aviso, caso elas não seja idênticas, um possível ataque pode estar ocorrendo.

▼ AMI:

- É possível salvar o estado atual de uma EC2 numa Amazon Machine Image.
- A partir dessas imagens, as EC2 podem ser executadas.

▼ EBS:

- O Amazon Elastic Block Store foi desenvolvido para provisionar um armazenamento consistente e prático para as EC2.
- Com ele é possível automatizar backups, por exemplo, ou então gerenciar os volumes de forma prática e eficiente.
- A cobrança é feita baseada no tamanho dos volumes e snapshots e na taxa de output de dados.

▼ Elastic IP:

- A cobrança é feita somente quando a instância anexada está ociosa, ou possui mais de um endereço IP.
- Esse serviço confere a uma EC2 um IPv4 fixo, com o qual é possível acessar a instância via internet.
- Um bom uso do ElasticIP é remapear o endereço de uma para outra instância, de modo a não quebrar o endereço que estiver em execução por motivos de instabilidade, como manutenção, por exemplo.
- Também é possível associar o endereço num registro de DNS, para nomear os domínios aos quais o endereço pertence.

▼ Load Balancers:

- A partir de um ponto de acesso, é possível utilizar um Elastic Load Balance para balancear a carga de requisições entre os componentes conectados a eles.

- É possível apontar um ELB para instâncias EC2, containers e para endereços IP.
- É possível operar em uma ou em múltiplas AZs.
- application lb: Voltado para tráfego HTTP e HTTPS
- network lb: Voltado para tráfego TCP
- classic lb: Voltado para estrutura clássica legado do EC2

▼ Autoscaling:

- Permite aumentar ou diminuir o número de instâncias EC2 automaticamente.
- É possível definir algumas variáveis de comportamento do autoscaling, similar ao hpa do k8s.
- É possível utilizar o modo dinâmico, que confere à AWS a decisão de autoscaling, ou então o modo preditivo, com o qual o administrador define manualmente momentos e quantidades de instâncias para serem escaladas.
- Esse serviço não tem valor adicional.

▼ RDS

- Relational Database Service, é o serviço que fornece acesso a bancos de dados relacionais para os serviços da AWS.
- Tem características que garantem acesso rápido e boas méticas de input e output
- Se utilizam da virtualização via EC2 para manter o servidor no ar,  porém toda a complexidade da máquina é abstraída, expondo apenas o banco de dados.

▼ Tipos

- Amazon Aurora - Recomendada, obviamente, pela AWS.
- MySql, MariaDb, PostgreSQL, Oracle, Microsoft SQL Server.
- Também existem diferentes tipos de armazenamento:
  - SSD de propósito geral: Recomendado para ambientes de teste e desenvolvimento.

        - Provisionamento de IOPS SSD: Recomendado para produção, pois foi feito para suportar grandes volumes de I/O, com baixa latência e alta consistência.
        - Magnetic: Disco legado para sistemas que ainda precisem desse tipo de armazenamento, sequer é recomendado pela AWS.

    ▼ Security Groups

        - Amazon RDS Security groups é o serviço que provê a capacidade de agrupar certos IP's que terão acesso ao RDS, limitando qualquer tráfico de rede externo a esse grupo.
        - Também é possível passar um EC2 security group inteiro para o RDS, que passará a aceitar tráfego de todos os EC2 desse grupo.

    ▼ Snapshots

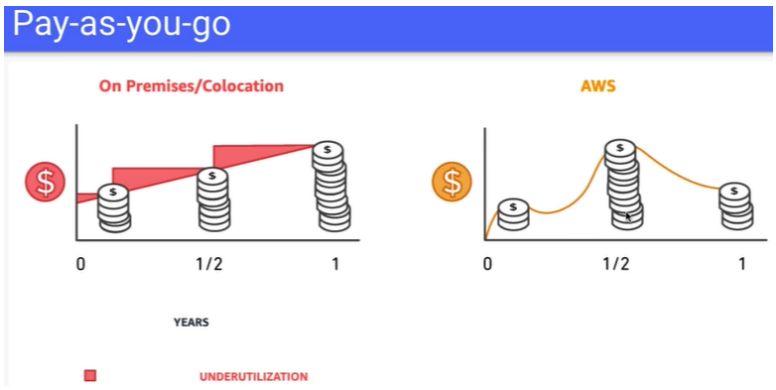
        - Criando um Snapshot de uma instancia do banco, por poucos minutos haverá uma pausa no I/O, que pode variar de acordo com o tipo do banco.
        - É interessante transferir o snapshot feito para um Amazon S3, de forma a integrar e facilitar a recuperação a partir deste snapshot.

▼ S3

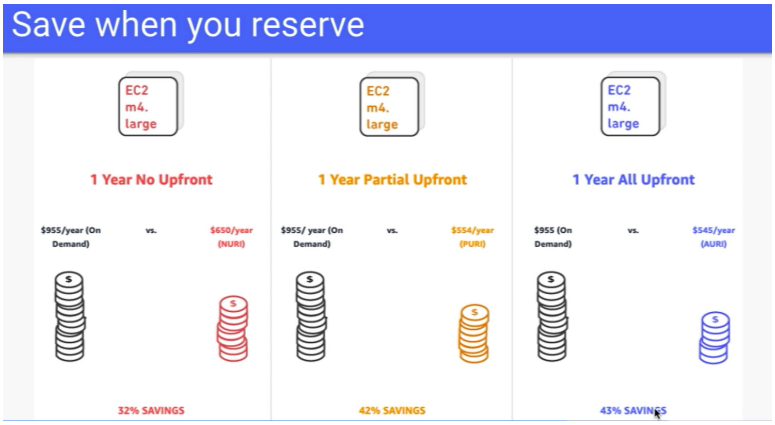
- Simple Storage Service, possui design com estabilidade estimada em 99,9%.
- Baseado em objetos:
- arquivos de 0 a 5 TB
- O objeto possui uma Key para o identificar, e este é armazenado dentro de estruturas chamadas de buckets.
- O S3 possui classes, as quais definem características do uso e de comportamento do armazenamento, como velocidade de recuperação, frequência de uso etc.
- Os buckets devem ser configurados com suas respectivas restrições de acesso, através das policies.
- É possível replicar um S3 entre regiões ou entre AZs.

▼ Cobrança e precificação:

- Basicamente, toda precificação da amazon gira em torno de três pilares: computação, armazenamento e tráfego de dados, geralmente de saída, da amazon para fora.
- O case de sucesso da AWS: Pague pelo que for usar, e somente por isso.

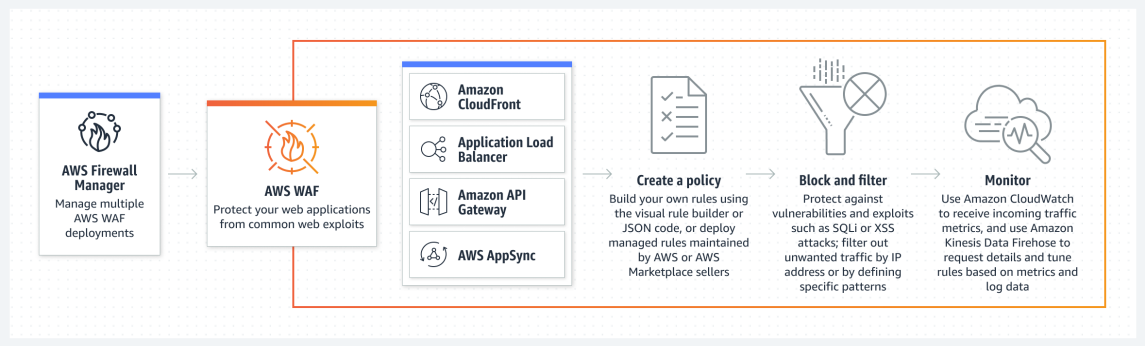



- Cobranças reservadas e sob demanda:
  - Como a AWS funciona sob demanda, no seu caso de uso principal, é possível adquirir descontos caso haja reserva de recursos, uma vez que o produto principal de disponibilidade de recursos ilimitada não será utilizado.

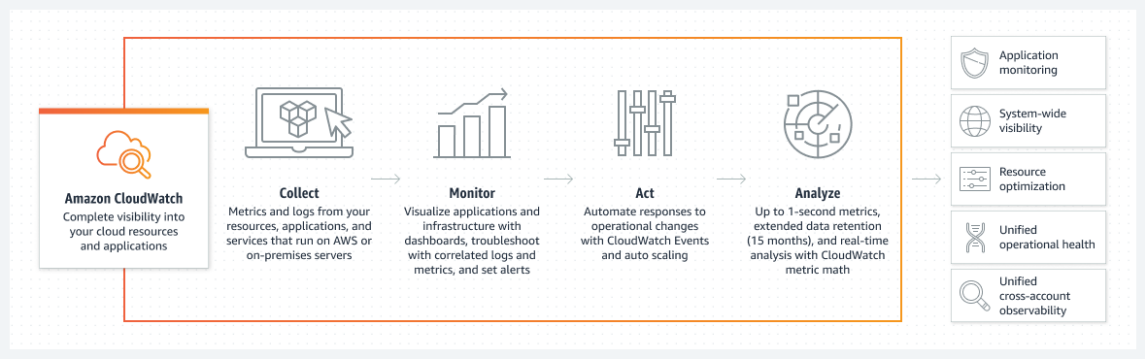



▼ Alguns serviços da AWS focados em segurança:

- AWS WAF: Esse serviço tem a função de proteger a aplicação WEB dos ataques mais comuns desse meio. Esse serviço é precificado com base no número de regras que são definidas nele, sejam as padrões, de controle de tráfego WEB, sql injection etc, ou as personalizadas, criadas pelo usuário. Atua na camada 7, de aplicação.




- AWS Shield: Funcionando junto do AWS WAF, este tem foco em monitorar o tráfego de rede para evitar ataques do tipo Distribuited Denial of Service, DDoS, filtrando requisições suspeitas que passem a consumir recursos em demasia sem motivo aparente.
- Cloud Trail: Uma espécie de sysLog, servindo para auditorar todos os eventos que aconteceram numa determinada cloud.
- Cloud Watch: Também serve para o monitoramento da nuvem, porém com maior abrangência. O cloudWatch coleta dados operacionais (como custo dos serviços em execução), monitoramento de logs, métricas e eventos. É possível configurar alarmes baseados em algumas métricas que podem gerar ações, como por exemplo parar ou escalonar um serviço com base num possível aumento ou redução de tráfego.




## ▼ Aplicação

### ▼ Apresentação prática:

- Mostrar MFA na conta pessoal
- configurar o AWS CLI
  - `chmod 400 projetoAWS-keypairEC2.pem`
  - aws configure

    - Colocar as chaves de acesso
    - aws s3 ls
- Terraform apply
- Acessar aplicação no navegador
- Acessar bastion via ssh
    - eval $(ssh-agent)
    - ssh-add `projetoAWS-keypairEC2-DEV.pem`
    - `ssh -A ubuntu@ec2-52-11-32-80.us-west-2.compute.amazonaws.com`
    - comparar fingerprint: `aws ec2 get-console-output --instance-id instance_id --output text > temp.txt`
- Acessar Frontend e Backend pelo bastion e mostrar logs do cloud init
    - Acessar via ssh ambos com IP privado
    - Matar o processo node rodando
        - `sudo killall -9 node`
    - Reiniciar ambos os servidores
        - `sudo node frontend`
        - `MYSQL_HOST="rdsmysql.cotblzgylx7o.us-west-2.rds.amazonaws.com" MYSQL_USER="admin" MYSQL_PASSWORD="admin123" MYSQL_DATABASE="mySQLDB" node index` **vulnerabilidade 1**
- Acessar banco de dados via backend
    -
    - `use mySQLDB`
    - `SELECT * FROM registros`
- Mostrar nginx reverse proxy na instancia e explicar redirecionamentos
    - Mostrar fetch feito no frontend para o reverse proxy
- Aplicar teste de carga e mostrar cloud watch e alarms
    - `locust -f locust.py`
- Mostrar Elastic IP, Route 53, e todos os Security Groups

- Mostrar Role para acessar Bucket
- Explicar motivo da NAT gateway
- Cálculo de custo CHAT GPT
- Duvidas:
  - Seria o elastic ip a solução final para nao reconfiguração de arquivos ou githubactions eh uma alternativa
  - Os problemas de exposição de credenciais sao resolvidos atraves de quais componenetes?
  - Como fazer um frontend pro navegador do host encontrar uma rota do route 53
  - Conectar EC2 num bucket privado, qual o melhor caminho?

aws ec2 describe-instances

mostrar s3

mostrar integração EC2 com S3 via IAM Role

Mostrar tarefas do script sh

Mostrar conexão com RDS

`mysql -h rdsmysql.cotblzgylx7o.us-west-2.rds.amazonaws.com -u admin -p`

Mostrar vpcs e security group chaining

Mostrar reverse proxy na ec2 de frontend

▼ TERRAFORM

▼ Comandos iniciais

- `terraform init`
- `terraform plan`
- `terraform apply`
- `terraform destroy`

▼ ANSIBLE

▼ Comandos iniciais

- `ansible-playbook`
  - `-u ubuntu`
  - `--private-key keyName.pem`
  - `-i hosts.yml`