>>> >>> AWS

Name: Henrique Tsuyoshi Yara (OPUS-software)

Date: January 24, 2023

Figure: AWS logo

>>> Índice

1. **História**
   - Linha do tempo
2. **Conceito**
   - NIST
   - 5 características
3. **AWS**
   - Origem
   - Formas de acesso
4. **Introdução**
5. **Virtual Private Cloud**
6. **EC2**
   - Tipo de instâncias
   - Amazon Machine Image (AMI)
   - Security Group
   - Autoscaling groups
   - EBS
   - Elastic IP
   - Load Balancers
7. **IAM**
   - Usuários
   - Tags
   - Policies
   - Roles
   - Relatórios
8. **IAM Identify Center**
   - Workforce identities
9. **Extra**
   - Reduce costs
   - AWS Network
   - Optimization Tips

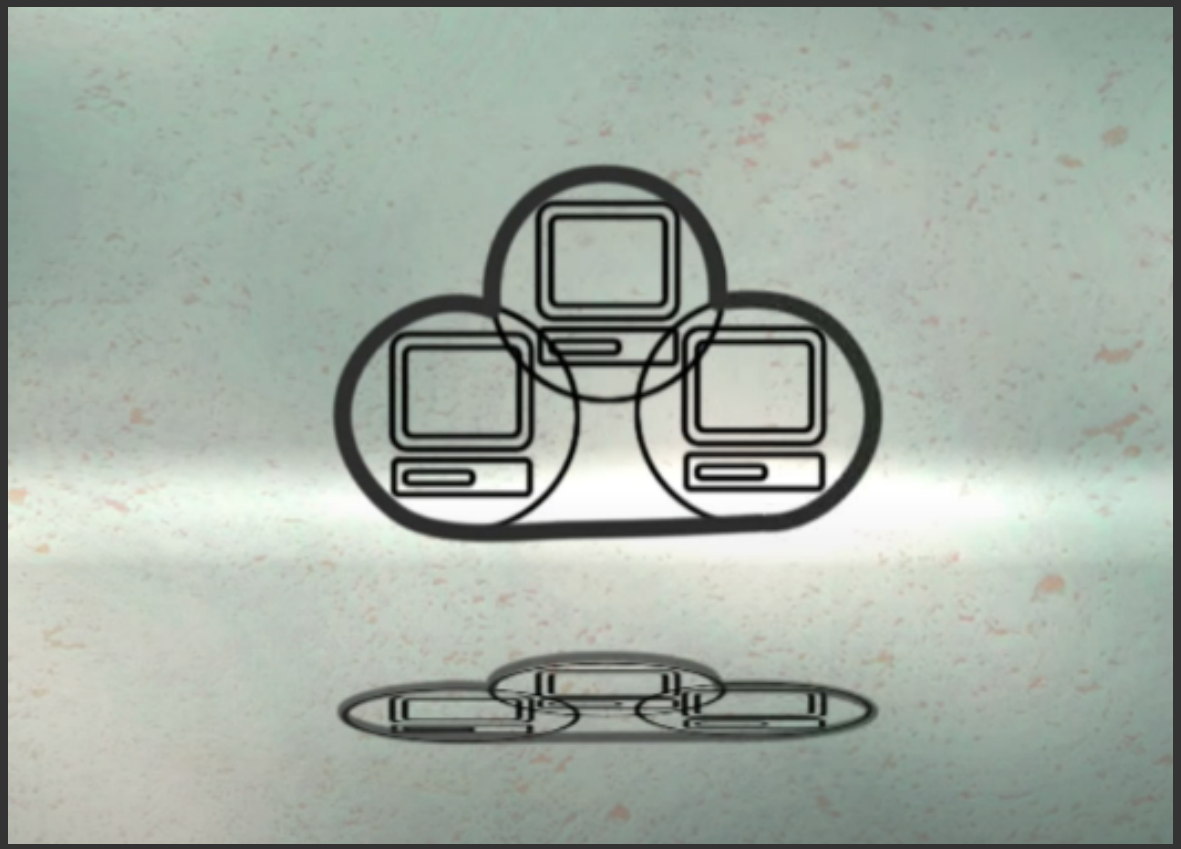
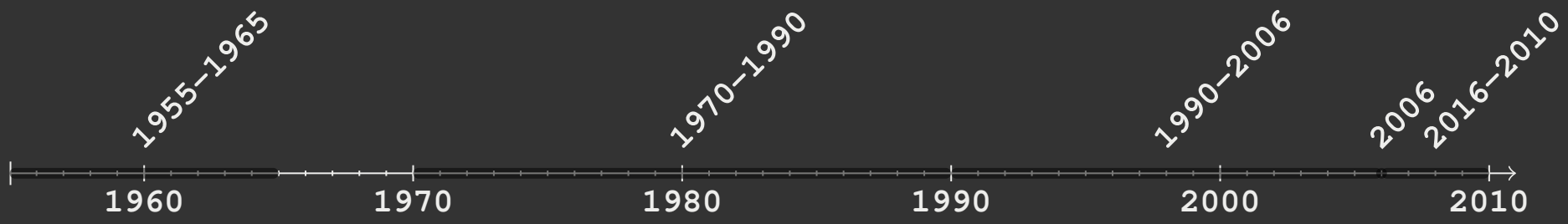


Figure: A cluster of servers drawn in a system diagram [1]

[1] Image source link

>>> Linha do tempo

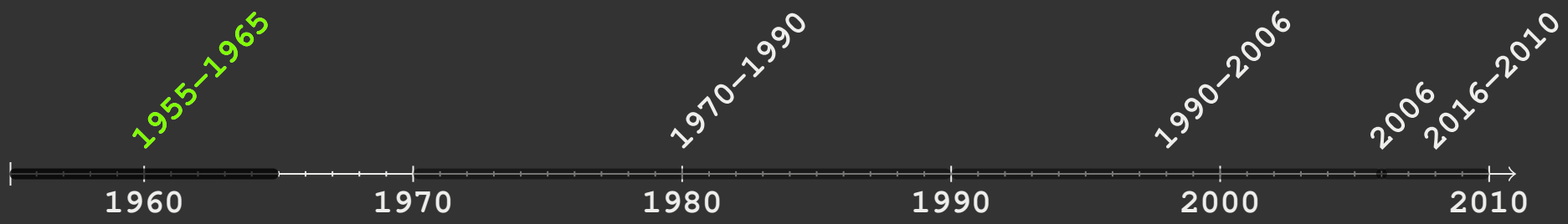
História da computação em nuvem[4]



* (1955-1965) Problemas na infraestrutura de TI
* (1970-1990) Hypervisors e a internet
* (1990-2006) Internet para todos
* (2006-2006) Precipitation
* (2006-2010) Primeiros dias da computação em nuvem

>>> Linha do tempo

Problemas na infraestrutura de TI
(1955-1965)



>>> (1955-1965) Problemas na infraestrutura de TI I

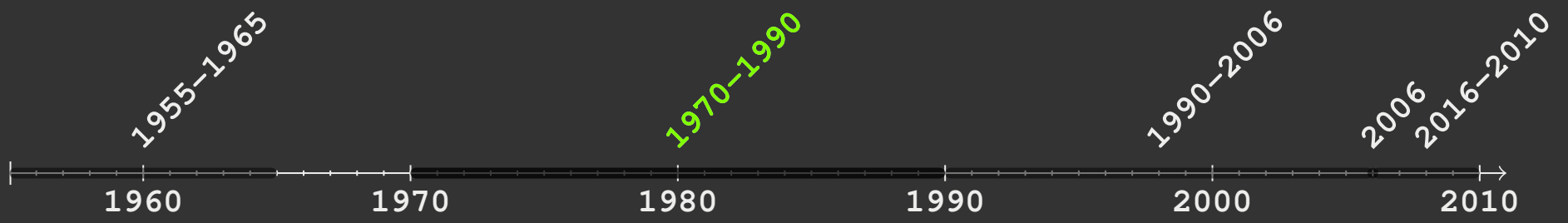
* 1942 - John Vincent Atanasoff construiu o computador ABC
* Server/Mainframe (Componentes principais):
  * Uma CPU
  * Um dispositivo de armazenamento de dados
* 1960 - Muito caro para aderir os computadores
  * Sala inteira para o servidor (Manter temperatura ideal, espaço, etc...)
  * Computador
  * Funcionários especializados
  * Problemas para adaptar o software
* Apenas empresas que com poder aquisitivo conseguiram aderir os computadores
* 1961 - John MacCharty fez uma palestra no MIT
  * Computação poderia ser vendida como água ou eletricidade [2]

>>> (1955-1965) Problemas na infraestrutura de TI II

* Mas seria necessário de uma grande evolução tecnológica

>>> Linha do tempo

Hypervisors e a internet (1970–1990)



>>> (1970–1990) Hypervisors e a internet I

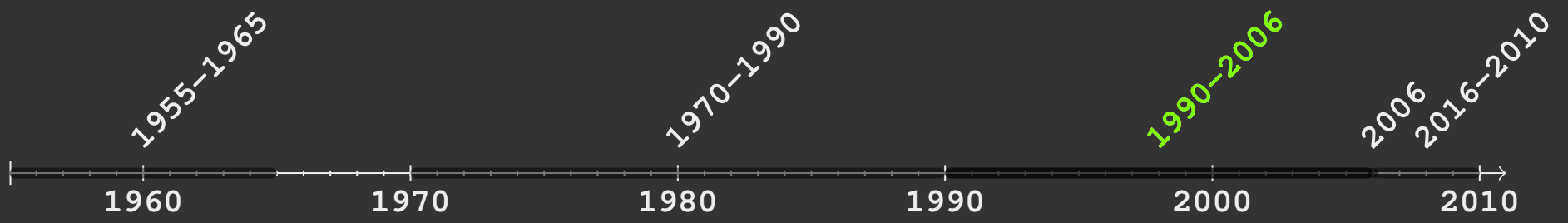
* Diminiur os custos da adoção do computador
  * Múltiplos usuários podem compartilhar o mesmo armazenamento e o poder de processamento da CPU
* 1970 – Nasceu a tecnologia da virtualização
  * Um computador pode ser particionado em vários partes
  * Cada parte pode rodar um código independente
  * Cada parte é chamada de Máquina Virtual (VM)
  * O server principal é chamado de *host*
* 1983 – A internet nasceu
  * Começou pelo projeto ARPANet para comunicação de professores de universidades
* Foi introduzida a arquitetura cliente-servidor, o cliente e o server eram os mainframes interagindo com código e informação conectados por cabos (Internet)

>>> (1970-1990) Hypervisors e a internet II

* Conforme os números de páginas foram crescendo o número de servers cresceram
  * Os servers se moveram para datacenters

>>> Linha do tempo

Internet para todos
(1990-2006)

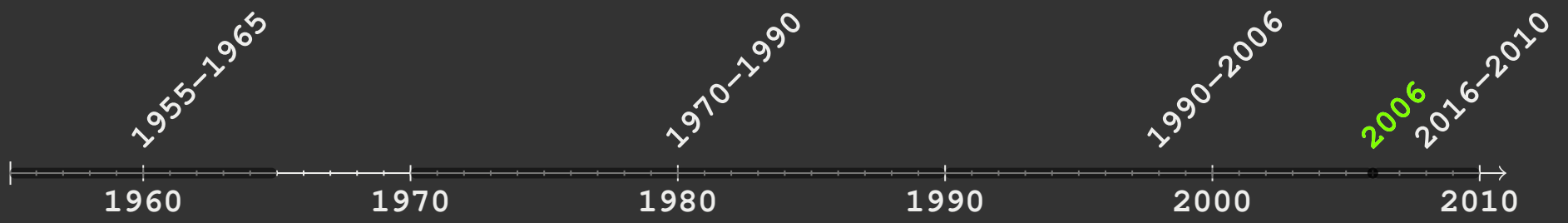


>>> (1990-2006) Internet para todos I

* A empresas maiores precisavam de datacenters maiores
* Problemas:
  * Ociosidade quando seu website não está sendo acessado. Trazendo prejuízo para as empresas com os servers ativos
  * A escalabilidade era feita de forma manual e precisava de dispositivos físicos. Seria muito difícil acompanhar o crescimento da aplicação
  * As atualizações eram feitas manualmente, e precisariam possívelmente consertar as máquinas. Perdendo um tempo precioso
  * A experiência do usuário não era boa. A distância do server impacta o delay da página

>>> Linha do tempo

Precipitation
(2006)

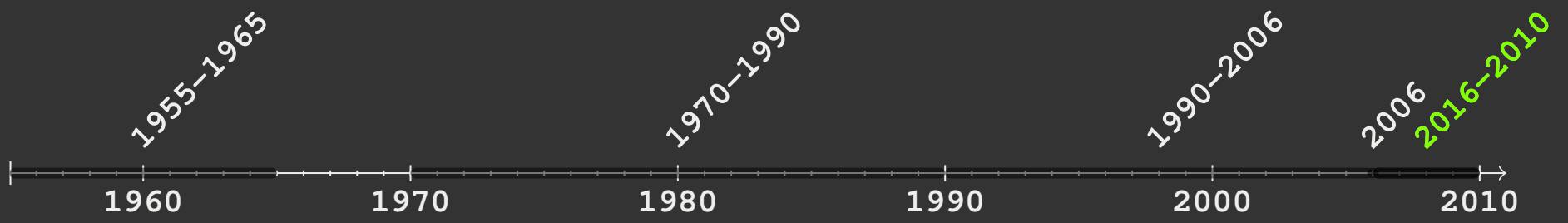


>>> (2006) Precipitação I

* Grandes empresas (*Google, Amazon, eBay, etc...*)
  * *Criaram data centers com centenas/milhares de servers de alta qualidade no mundo inteiro*
  * *Por causa dos prejuízos decidiram começar a alugar essas máquinas*
* *2006 - "Cloud Computing" foi introduzino no mundo como forma de aluguel de poder computacional*
* *A Amazon chamou essa divisão de Amazon Web Services (AWS)*
* *A AWS foi a primeira de muitos provedores de cloud*
* *Depois de 50 anos o sonho de John McCharty foi realizado*

>>> Linha do tempo

Primeiros dias da computação em nuvem
(2006-2010)



>>> (2006-2010) Primeiros dias da computação em nuvem I

* Por 6 anos a AWS estabeleceu seu monopólio na área de nuvem
* A computação em nuvem permitiu que as empresa pudessem dar um foco maior no desenvolvimento
  * Os dados são criptografados e seguros
  * Dados são armazenados com redundância em nuvem
  * Escalabilidade
  * De fácil deploy
  * Alta disponibilidade

>>> (2006-2010) Primeiros dias da computação em nuvem

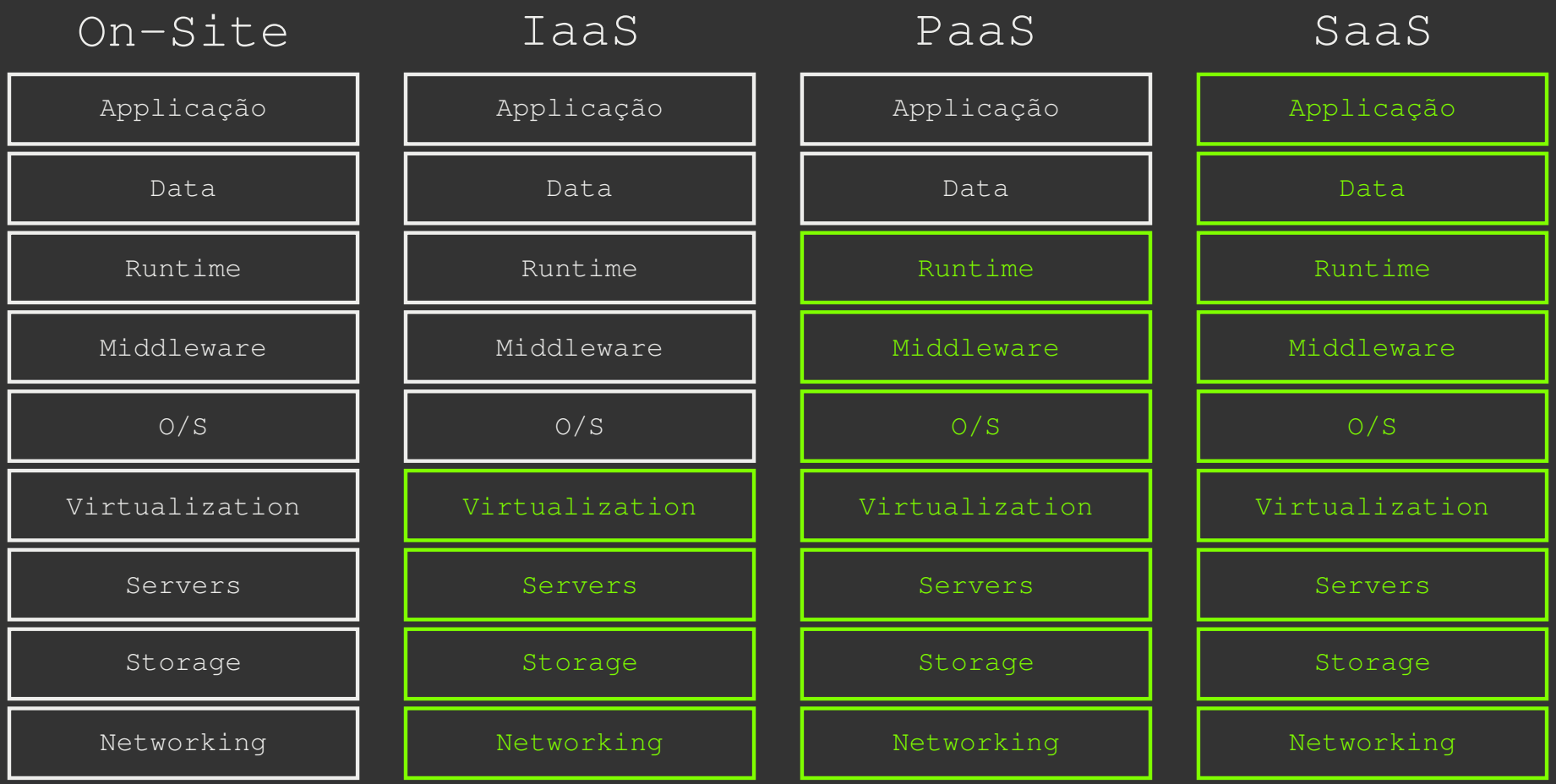


Figure: Imagem retirado do site da redhat[2]

[2] RedHat IaaS, PaaS e SaaS

>>> O que é cloud para o NIST

* Um modelo para habilitar o acesso por rede a um conjunto compartilhado de recursos de computação e precisa ser:
  * Ubíquo (Pode ser encontrado em todos os lugares)
  * Conveniente
  * Sob demanda
* **Recursos de computação:** Redes, servidores, armazenamento, aplicações e serviços
* Esses recursos devem ser provisionados e liberados com o mínimo de esforço de gerenciamento ou interação com o provedor de serviços.

>>> 5 Características

* Conceito que reúne vários *softwares* e utiliza de virtualização
* Possui algumas características específicas (NIST):
    * Autoserviço sob demanda
    * Amplo acesso a rede
    * Pool de recursos
    * Rápida elaticidade
    * Serviços mensuráveis

>>> Auto-serviço sob demanda

* O consumidor pode providionar por conta própra **Recursos de computação**
* Não necessita da intervenção humana dos provedores de serviço

>>> Amplo acesso por rede

* Os **Recursos de computação** estão disponíveis através da rede
* São acessados através de mecanismos padronizados que promovem o uso por dispositivos, clientes leves ou ricos de diversas plataformas (Smartphones, tablets, laptops ou desktops)

>>> Agrupamento de recursos

* Os **Recusos de computação** do provedor são agrupados para atender a múltiplos consumidores em modalidade multi-inquilinos (Recursos físicos e virtuais diferentes dinamicamentes atribuídos e reatribuídos conforme a demanda dos consumidores)
* Há uma certa independência de localização geográfica, uma vez que o consumidor em geral não controla ou conhece a localização exata dos recursos fornecidos
* Mas pode ser capaz de especificar a localização em um nível de abstração mais alto (país, estado, datacenter)

>>> Elasticidade rápida

* Os recursos podem ser provisionados e liberados elasticamente, em alguns casos automaticamentes, para rapidamente aumentar ou diminuir de acordo com a demanda
* Para o consumidor, os recursos disponíveis para provisionamento muitas vezes parecem ser ilimitados e podem ser alocados em qualquer quantidade e a qualquer tempo

>>> Serviços mensurado

* Os sistemas na nuvem automaticamente controlam e otimizam o uso dos recursos através de medições em um nível de abstração apropriado para o tipo de serviço (como armazenamento, processamento, comunicação de ree e contas de usuário ativas)
* A utilização de recursos pode ser monitorada, controlada e informada, gerando transparência tanto para o fornecedor como para o consumidor do serviço utilizado

>>> Origem AWS I

* 2006 - Amazon Web Services começou a oferecer infraestrutura de TI como forma de serviços web
  * Low Cost (Pay-as-you-go)
  * Agility and Instant Elasticity
  * Open and Flexible
  * Secure (PCI DSS Level 1, ISO27001, etc...)
* Instância (2006):
  * 1.7 GHz Xeon Processor
  * 1.75 GB of RAM
  * 160 GB of local disk
  * 250 Mbps network bandwidth
* Instância (2019):
  * 4.0 GHz Xeon Processor (z1d instance)
  * 24 TiB of RAM (High Memory instances)
  * 60 TB of NVMe local storage (I3en.metal instances)
  * 100 Gbps network bandwidth

>>> Formas de acesso

* **Console:** Permite gerenciar a infraestrutura e os recursos da aws com uma interface web
* **SDK:** Simplifica o uso dos serviços da AWS provendo bibliotecas para os desenvolvedores
  * Tem suporte para: Java, .NET, C++, PHP, etc...
* **CLI:** É uma ferramenta para gerenciar os serviços da AWS. É possível controlar múltiplos serviços usando a linha de comando e é possível automatizar usando scripts

>>> Recursos

* Cada recurso vai ter um **Amazon Resource name** (Identificador único)

>>> Free Tier

* São recursos que podem ser usadas de graça na Amazon

>>> Calculadora

* É utilizada para calcular o custo total de algum recurso
  * Calculadora antiga
  * Calculadora nova

>>> Regiões

* Cada região tem um preço diferente
* Uma região é composta de zonas de disponibilidade
* Algumas regiões podem ter mais serviços que outras
* **OBS:** É bom saber se juridicamente a gente pode armazenar os dados fora do Brasil
  * Regiões e zonas de disponibilidade
  * Serviços regionais
* **OBS:** Tráfegos entre zonas de disponibilidade ou regiões podem acabar sendo cobrados

>>> Zonas de disponibilidade

* Compõem as regiões
  * Serviços regionais
* **OBS:** Tráfegos entre zonas de disponibilidade ou regiões podem acabar sendo cobrados

>>> Status AWS

* Para verificar o status das zonas de disponibilidade/regiões ou recursos
  * Status AWS
* **OBS:** Tráfegos entre zonas de disponibilidade ou regiões podem acabar sendo cobrados

>>> Pontos de presença

* Pontos de cache utilizado pela AWS (É possível usar *CNDs*)

>>> Virtual Private Cloud (VPC)

* VPCs são isoladas entre si, mas podem ser configuradas para se comunicarem
* Cada região tem uma VPC padrão, mas é recomendada criar sua própria VPC para o ambiente de produção
* Dentro de uma VPC é possível criar uma subnet
* As subnets são aplicadas em AZs (Zonas de disponibilidade)
* Subnet:
  * Pública: Pode ser acessada remotamente por qualquer lugar
  * Privada: Só vai ser acessível por dentro da AWS
* **VPC wizard** tem algumas configurações pré-definidas de VPC
* Lembrar de verificar e configurar:
  * DHCP options set
  * DNS resolution
  * DNS hostname

>>> NAT Gateway

* Instâncias dentro de subnets privadas podem conectar com serviços fora da VPC, mas instâncias de fora não podem iniciar ocnexões com essas instâncias
  * Public: Instâncias em subnets privadas podem conectar com a internet
  * Private: Instâncias em subnets privadas podem conectar com outros VPCs
* Cobrança por hora de uso e quantidade em GBs de dados processados

>>> Internet Gateway

* Permite a comunicação do seu VPC com a internet
* Libera a entrada e a saída de determinado **Route Table**
* Não tem custo

>>> Route table

* Associa as **subnets**
* Se a **Route table** não tiver uma rota default ela não está pública

>>> Security Groups X NetworkACL

* **Securty Groups**
  * Opera no nível de instância (Primeira camada de defesa)
  * Apenas regras de liberação
  * Stateful: o tráfego de retorno é automaticamente permitido, independentemente de quaisquer regras
  * Aplica-se a uma instância somente quando especificado o grupo de segurança
* **NetworkACL**
  * Regra de segurança da rede (Como se fosse um *firewall*)
  * Regras de liberação e negação
  * Stateless: o tráfego de retorno deve ser explicitamente permitido pelas regras
  * Aplica a todas as instâncias nas sub-redes

>>> NetworkACL

* Cada regra vai ter uma prioridade
* É bom deixar um espaço entre cada regra para possíveis regras futuras (Ex: deixar 10 espaços entre cada regra)
* **OBS**: É bom liberar portas efêmeras (1024-65535). São usadas para comunicações de saída através do protocolo de rede TCP/IP

>>> Network Optimization

- Quantas VPCs eu preciso?
  - Depende de quanto achar necessário para separar a aplicação
  - É comum uma VPC Para AWS Network Firewall ou Firewall com Gateway Load Balancer.
  - Talvez VPCs para Ingress e Egress centralizados
- Tamanho da VPC
  - Evitar alocar /16 endereços IP padrão para todas as VPCs
  - Alguns recursos precisam de IPs livres (Load balancer precisa de 8 IPs livres)
  - É possível usar o IPAM (VPC IP Address Manager)[3] para gerenciar os IPs nas redes
  - OBS: É possível usar o cloudWatch junto com o IPAM para verificar se os endereços IPs estão acabando
- Quantas subnets por VPC?
  - Pelo menos 1 subnet por VPC
  - Caso a aplicação esteja em várias AZs, é recomendado pelo menos uma subnet por AZ
  - OBS: Quando uma subnet é colocada em uma AZ não é possível mudar

* Oferece instâncias onde podemos escolher: Processador, Sistema Operacional, Armazenamento, Rede

Figure: Tipo das instâncias

* São para propósito geral, podem ser usadas em: Web/App servers, Enterprise apps, Gaming servers, caching Fleets, Analytics applications, Dev/Test Environments, etc...

>>> Instâncias de propósito geral

* **Instâncias M5**: Um equilíbrio entre memória, poder computacional e velocidade de rede. Proporção de memória para vCPU é de 4:1
* **Instâncias T3**: Tem uma linha base de performace da CPU e tem a possibilidade de passar a linha base. Usado para workloads que não usam a CPU constantemente.
* **Instâncias A1**: Workloads que precisam escalar em múltiplos cores, rodar instruções ARM, etc...

>>> Instâncias Memory-intensive workloads

* Usado para: Banco de dados de alta performace, Análise de Big Data, Cache de memória, etc...
* **R5 Instances**: Usado para workloads que processam data sets grandes em memória. Proporção de memória para vCPU é de 8:1
* **X1/X1e Instances**: Proporção de memória para vCPU é de 16:1 e 32:1
* **High memory instances**: Certificado para rodar SAP HANA. Possui 6 até 24 TB de memória

>>> Instâncias Compute-intensive workloads

* Usado para: High-perf computing (HPC), Multiplayer Gaming, Video encoding, etc...
* **C5 Instances**: Alta performace por um preço baixo. Proporção de memória para vCPU é 2:1
* **z1d Instances**: ALta performace em uma única thread. Processador mais rápido em nuvem de 4.0 GHz. Proporção de memória para vCPU é de 8:1

>>> Instâncias Storage-intensive workloads

* Usado para:
  * Alta operações de I/O. Ex: High-perf databases, Real-time analytics, No SQL databases, etc...
  * Muito armazenamento. Ex: Big Data, Kafka, HDFS, Log processing...
* **Instâncias I3/I3en**: Otimizadas para operações de I/O com pouca latencia
* **Instâncias D2**: Custo baixo por armazenamento e suporta alta taxas de transferências
* **Instâncias H1**: Aplicações de custo baixo que usam altas transferências de dadso e adesso sequencial para grandes Data Sets. Mais vCPUS e memória por TB que o D2

>>> Workloads de computação acelerada

* Usado para: Machine learning (PLN, Reconhecimento de imagem e vídeo, etc...), HPC (Dinamica dos flúidos, Química computacional, etc...), Gráficos (Codificação de vídeo, Modelação 3D e renderização, etc...)
* Podem ser usados:
  * CPU
  * GPU
  * FPGA
  * ASICs

>>> Instâncias de computação acelerada

* Usado para:
  * Alta operações de I/O. Ex: High-perf databases, Real-time analytics, No SQL databases, etc...
  * Muito armazenamento. Ex: Big Data, Kafka, HDFS, Log processing...
* **Instâncias I3/I3en**: Otimizadas para operações de I/O com pouca latencia
* **Instâncias D2**: Custo baixo por armazenamento e suporta alta taxas de transferências
* **Instâncias H1**: Aplicações de custo baixo que usam altas transferências de dadso e adesso sequencial para grandes Data Sets. Mais vCPUS e memória por TB que o D2

>>> Amazon Machine Images (AMIs)

* Amazon Maintained
  * Imagens de Windows e Linux
  * Recebem Updates pela amazon em cada região
  * Amazon Linux 2 (5 anos de suporte)
* Marketplace Maintained
  * São gerenciados e mantidos pelos parceiros da AWS
* Your Machine Images
  * AMIs que foram criadas de instâncias EC2
  * Podem ser privadas, compartilhadas com outras contas ou publicadas na comunidade

>>> Load Balancers

* Classic
* alb
* nlb
* gateway

>>> IAM

* Identity and Acess Management
* Boas práticas:
  * Habilitar MFA
  * Criar um usuário padrão para cada pessoa do time e dar permissões (Não usar o **root**)
  * Usar grupos para atribuir permissões
  * Aplicar uma política de senhas do IAM
* OBS: É universal, funciona em todas as regiões

>>> IAM - Users

* **Programmatic access**
* Enables an access key ID and secret access key for the AWS API, CLI, SDK, and other development tools.
  * Instalar o CLI para ter acesso ao AWS
  * Dar acesso de um bucket para uma aplicação
* *AWS Management Console access*
* Enables a password that allows users to sign-in to the AWS Management Console.
  * Precisa dar permissão para esse usuário

>>> IAM - Tags

* Servem para a gente identificar serviços
* É possível fazer um relatório de faturamento baseado em *Tags*
* **OBS:** É possível ter até 50 tags por serviço

>>> IAM - Políticas Pt.1

* É uma boa prática criar grupos com permissões para os usuários. E não colocar permissões diretamente no usuário
* Permissões mais específicas são mais fortes (Permissão de usuário prevalece contra permissão de grupo)
* Políticas de senha
    * Exigir que o usuário use senhas fortes
    * Expiração de senhas
    * Impedir reutilização de senhas
    * Etc...

>>> IAM - Políticas Pt.2

* Políticas de acesso
  * As políticas podem ser definidas por um arquivo Json
  * Pode ser usado políticas prontas ou criar políticas específicas
  * Então as políticas podem ser atribuídas em usuários/grupos

>>> Funções/Roles

* Dar permissões para:
    * Recursos
        * Ex: Dar permissão para uma instância acessar um bucket
    * Outras contas AWS
    * Federações do SAML 2.0
    * Identidade web (Login Google, amazon, etc...)

>>> Relatórios de acesso

* Relatórios de credenciais
  * Lista de todas as credenciais geradas
* Access Analyzer: Gera um relatório de políticas pra a gente ver o que precisa ser modificado. é possível arquivar, resolver, etc...

>>> IAM Identify Center

* Expande as capacidades da AWS IAM
* Provê um lugar centralizado para administrar os usuários e seus acessos para contas AWS e aplicações cloud
  * Workforce identities
  * Application assignments for SAML applications
  * Identity Center enabled applications
  * Multi-account permissions
  * AWS access portal

>>> IAM - Users

* a

>>> Usando Spot (com Coiled)

* Spots podem salvar muito dinheiro [5], mas você precisa:
  * Encontrar máquinas Spots o suficiente na sua região
  * Lidar com seu desaparecimento

>>> Escolher AZ e Região [1]

* Verificar se na sua zona tem Spots o suficiente
  * Usar o Spot placement score para ver qual região ou az tem mais chance de suprir as necessidades do usuário
* Deixar todos os Spots em uma zona só (Evitar custos de transferência de dados por zona)

>>> Fallback

* Se os Spots não forem o suficiente é possível colocar instâncias *On-demand para suprir as necessidades*
  * *Spot: Vai alocar apenas Spots*
  * *Spot-with-fallback: Vai alocar Spots e vai alocar on-demand caso seja necessário*
  * *On-demand: Se quiser que tudo seja estável*

>>> Burstable Instances

* Vai existir cobrança se usar mais CPU do que foi definido
* Caso a instância use menos CPU do que foi definido a instância vai juntar créditos de CPU para usar

>>> Outros

* EBS Disks
* Network traffic for EC2 instances
* Nat Gateways

>>> AWS Network Optimization Tips

* Read this [3]

>>> Referencias I

[1] Coiled. *AWS Cost Explorer Tips and Tricks*. URL: https://blog.coiled.io/blog/aws-cost-explorer-tips/?ck_subscriber_id=2013077611 (visited on 01/20/2023).

[2] Simson Garfinkel. *The Cloud Imperative*. URL: https://www.technologyreview.com/2011/10/03/190237/the-cloud-imperative/ (visited on 01/20/2023).

[3] Bentzen M., Looney B., and Kumar R. *AWS Network Optimization Tips*. URL: https://aws.amazon.com/blogs/networking-and-content-delivery/aws-network-optimization-tips/?ck_subscriber_id=2013077611 (visited on 01/20/2023).

>>> Referencias II

[4] Ankit R Sanghvi. *History of Cloud Computing*. URL: https://www.cohesive.so/blog/the-history-of-cloud-computing (visited on 01/20/2023).

[5] Nat Tabris. *Save Money with Spot*. URL: https://www.coiled.io/blog/save-money-with-spot (visited on 01/20/2023).