▼【本文档使用方式】：

- **当作【字典】使用。**

即学习时遇到【不理解含义的概念 or 服务】时，使用文档中的【查找 / 替换】功能，搜索对应【概念 or 服务 名称】。

- **可结合【各服务简短总结(可搜索)】使用：**

  【AWS 服务一览-CLF02｜基于 "試験にでる AWS サービスの範囲（02 版）"】

  https://sanuei.github.io/AWS-CLF02-menu/

▼【AWS CLF-C02 学习路径】：

1. 首先，请完整学习并理解 AWS（CLF-C02）的服务内容。

- **学习资源①——课程：**

  B 站搜索：【AWS 云从业者认证速通，0 基础带你入门云计算！(AWS CCP CLF-C02)】

- **学习资源②——官方培训：**

  【AWS skillbuilder 官方培训】

  https://skillbuilder.aws/learn/94T2BEN85A/aws-cloud-practitioner-essentials-/41SH4XWVQ8

  【AWS Document 官方文档】

  https://docs.aws.amazon.com/

*** 【课程】有助于理解；【官方文档】最正式、完整，且其中配有【图表&每部分例题】。**

2. 其次，请刷题练习。

- **题库①——【AWS CLF-C02 中文题库】购买题库微信：est258258**

*** 这是我刷题使用的题库。**

优点：是中文题库，方便理解；

缺点：没英文原题对照——有些 AWS 服务的中文译名会变化（正式考试时也是），所以记得一定要【对照英文原名-中文常用名】（可以自己用 ChatGPT 查对应词）进行学习！

- **题库②——【英文题库】AWS Certified Cloud Practitioner.zip**

  网盘链接: https://pan.baidu.com/s/1rFbV0Hd3-eSM-H-dotMNUw?pwd=yoyo

*** 英文题库，从网页打开，并用插件【沉浸式翻译】可进行对照理解。**

作成者：朱晓駿 ｜作成期间：2026 年 1 月

AWS CPP(CLF-C02)笔记 目录

作成者：朱晓駿 | 作成期间：2026 年 1 月

作成者：朱晓駿

作成日：2026 年 1 月 7 日

■ AWS CCP 考试（CLF-C02）精简总结

○【计算】总结

EC2 ——**弹性计算云。**

4 种**付费模式**：

On-**D**emand **I**nstances	**按需实例**(按量)	：贵、时间短(灵活)
Spot **I**nstances	**竞价实例**(按量；会中断)	：折扣+系统中断（**无工作负载**）

(可能被其它出价高的人**中断服务**，因此很多人把这个实例**【只作计算节点、不存数据】**而**【在预留实例上存储数据】**，组合使用实例→此称为**【无状态工作负载】**)

Reserved**I**nstances	**预留实例**(长期 1 年以上；绑定)	：便宜(72%折扣价)、时间长；**绑定 AZ、型号、地区**
Savings **P**lans	**节省计划**(长期 1 年以上；不绑定)	：同预留；但**不绑定**实例系列大小、操作系统、租赁或 AWS 区域

AS **(Auto Scaling)** ——

→**Cloud Watch** **[实时监控]**达负载阈值

→**ELB（负载均衡）** **[伸缩 资源容量]**扩展到【多个服务器】进行处理

Batch ——模型训练、排列组合分析 的 **批处理**

ElasticBeanstalk ——**自动化**帮我们**部署基础设施**，但仍需**自己管理。**

Lightsail ——**低配版 EC2**，但有预配置模板、上手更快更简单。

→**简易化配置服务器（基本最高 16 核 32G）**

***Lightsail 和 EC2 都是 VPS（虚拟私有服务器），即都是【计算服务器】。**

***EC2 最高可几百 G，而轻量化的 Lightsail 基本最高 16 核 32G。**

Lambda ——**关键词**是**【无服务化计算(更节省)】**和**【事件触发】**。

▼对比：

- **EC2** ——完全**自主可控**
- **Elastic Beanstalk** ——**帮助选择好**对应的基础设施，管理还是**自己管理。**
- **Lambda** ——只负责**代码的简易部署**（适用于**事件触发**类型）

作成者：朱晓駿 | 作成期间：2026 年 1 月

○【存储】总结

只有 **1 块硬盘**，只能在 1 个 EC2 上使用—— **EBS**

多个服务器 同时访问 1 块硬盘 —— **EFS**（共享文件）

高性能 同时访问 1 块硬盘 —— **FSx**

数据量大、价格低、经常使用 —— **S3**

不经常使用、深度归档 —— **S3 Glacier**

本地-云 备份 —— **Storage Gateway**

云 备份 —— **Backup**

○【数据库】总结

企业·用户管理 ——**RDS**

性能高、费用低 ——**Aurora (RDS)**

游戏进度、**点赞评论** →**键值对 DB** ——**DynamoDB**

会话、状态信息 →**内存型 DB** ——**MemoryDB for Redis**

【**关联度**】企业关系网、个性化推荐、知识图谱 →**图 DB** ——**Neptune**

○【容器】总结

- ECS ——云上 **容器（微服务；CI/CD）**
- EKS ——云上 **容器调度、编排**
- ECR ——**镜像仓库**
- Fargate ——**无服务器化容器**服务

  ECS 的 **EC2** 托管 →**云主机** [**要自己管理** →**有更大自主可控性**]

  ECS 的 **Fargate** 托管 →**无服务化** [**无需自己管理** →**自动部署、扩展、负载平衡**]

- **EC2** ——完全**自主可控**
- **Elastic Beanstalk** ——**帮助选择好**对应的基础设施，管理还是**自己管理**。
- **Lambda** ——只负责**代码的简易部署**（适用于**事件触发**类型）

○【迁移和传输】总结

AMS(MGN) ——**迁移【服务器】——不停机(不断同步、实时增量)**

DMS ——**迁移【数据库】**

SCT ——**换【数据库引擎】**

Migration Hub ——**迁移任务**的**【集中管理和调度 的 控制台】**

ADS (Application Discovery Service) ——系统不熟或太庞大 → 用 ADS 来**【发现并统计】各类应用**

SnowFamily ——**【物理传输】**→离线没网 or 数据量大(PB、TB)

TransferFamily ——**【传输】** →协议：FTP、SFTP、FTPS

○【联网与内容分发】

VPC （=云中 **内网**）

通过【**对等连接**】【**VPN**】进行 **VPC 之间的打通**。

通过 **Gateway（网关）访问互联网**。

私有子网→NAT Gateway →连接 互联网

公有子网→Internet Gateway →连接 互联网

VPC 内部 访问 S3 → VPC endpoint

- **安全组** ：附加在 **EC2 实例**上**控制流量进出。**
- **NACL** ：附加在**子网**中**控制流量进出。**

CloudFront （内容分发=CDN；缓存至更近）

Direct Connect （**混合云**；专线-传输快,搭建慢；静态）

VPN （**混合云-加密连接；公网连接；比专线便宜**

Global Accelerator （加速联网；静+动）

Route 53 （DNS；注册+管理+解析 域名；全球负载均衡）

Transit Gateway （中央枢纽 网关）

API Gateway

○【安全性身份和合规性】

59▼ ACM（AWS Certificate Manger）

60▼ AWS CloudHSM（硬件安全模块）

61▼ Amazon Detective（安全分析工具）

62▼ Cognito （账号管理；身份验证与授权：注册管理 + 第三方接入）

63▼ GuardDuty （威胁检测：AWS 账户 + 工作负载）

64▼ IAM（权限管控：分配账号+分配权限）(Identity and Access Management)

65▼ IAM Identity Center（SSO 单点登录 = AWS Single Sign-On）（用于：内部不同应用+外部访问）

66▼ Amazon Inspector(安全漏洞扫描：软件漏洞+网络暴露)

■ Inspector 和 GuardDuty 区别：

67▼ Amazon KMS(加密 服务)(Key Management Service)

68▼ Macie(发现和保护敏感数据——只发现，不能进行加密，另行用别的加密)

69▼ WAF(防火墙，阻挡对应用的攻击)Web Application firewall

70▼ Firewall Manager(WAF 防火墙的管理工具：跨账户集中配置+管理防火墙规则)

71▼ Shield(安全防护：防 DDoS 攻击+应用保护+API 端点防护)

72▼ Secrets Manager(托管密钥：存储、管理、轮转敏感信息)

73▼ Secrets Hub (托管的 集中管理和监控账户安全状态——即时洞察威胁和违规)

74▼ Artifact (提供安全合规性 文档、报告)

75▼ RAM(跨账户资源共享)、AM(审计管理)、DS (企业级目录服务（用户、账号、权限）)

一、■ 云概念

▼【地域、可用区】：

region（地域）

available zone（可用区）

datacenter

local zones（本地区域）

edge location（边缘站点）

▼【云种类】：

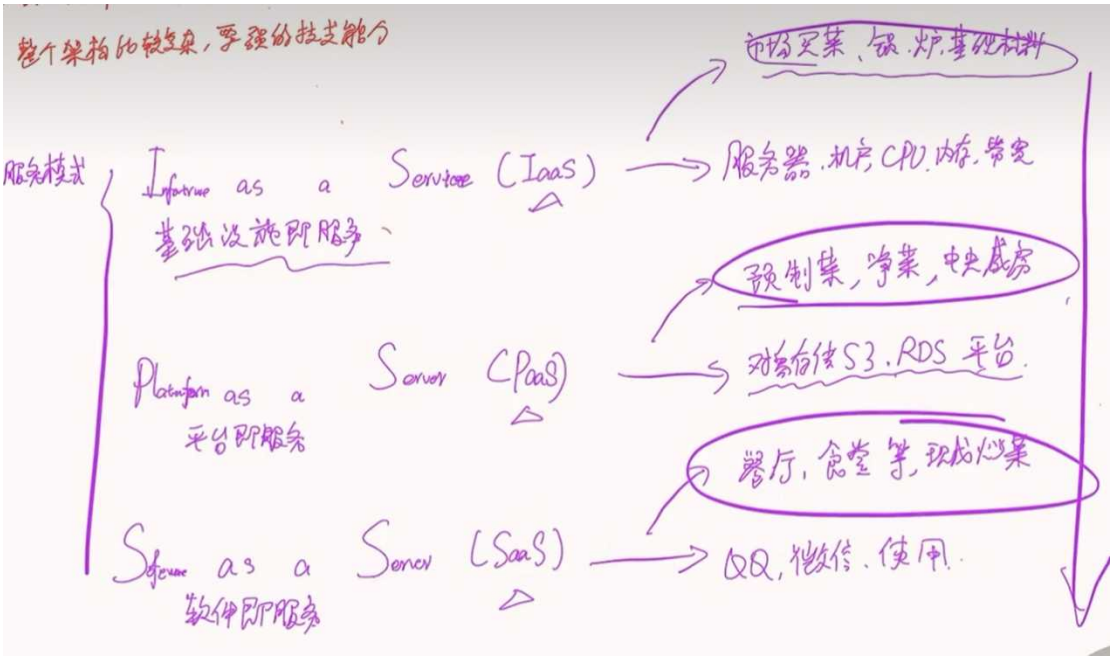
公有云、私有云、混合云

▼【云服务模式】：

IaaS（基础设施 即服务）——给锅，自己买菜、做菜（infrastructure）

PaaS（平台 即服务）——给预制菜，自己做菜

SaaS（软件 即服务）——直接上菜，直接吃



▼【服务】：

（官网——产品与服务：https://aws.amazon.com/cn/products/）

●计算：

云服务器 EC2、lamda（无服务器化的计算服务）、ECS、EKS

●存储：

对象存储 S3、NES（文件共享）、EBS(块存储)、EFS（完全托管型文件系统）

●网络：

VPC——route53、CloudFront 缓存、（GA）、DC、TransitGateway

●数据库：

RDS 关系型数据库——for mysql、Ms、Aurora、Neptune

非关系型数据库——redis、MongoDB、DynamoDB

●其他：

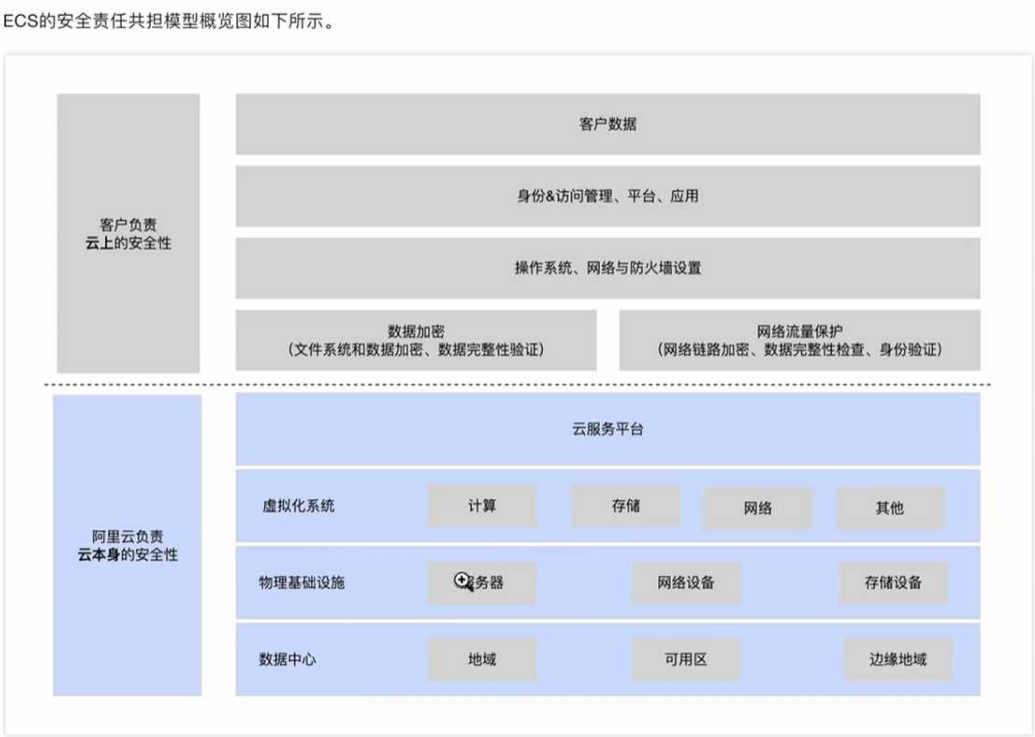
身份验证、AI 大模型、监控、审计、安全合规

▼【付费模式】:pay as you go（即用即付）

按量付费：按小时（1h0.5 刀、2.5h1.5 刀）、按分钟、按算力（0.5cpu0.5memory）

包年包月（预留）（1 年 8 折，3 年 7 折）

▼【责任共担模型】：（见下图）



▼【SLA 协议（服务等级协议）】：

一句话概括：承诺 99.99%时间服务正常，否则按比例赔付。

用处：

如果服务器坏了，客户向你索赔。

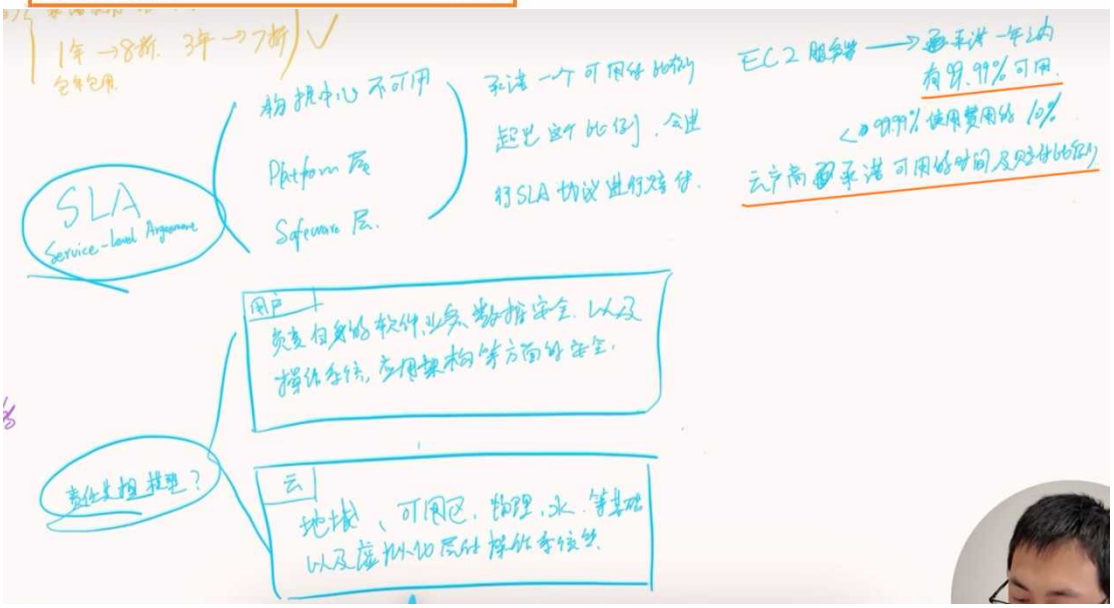
但按照【**责任共担模型**】——这是 AWS 服务商的责任，所以你公司不该赔付、该 AWS 服务商赔付；

而赔付的多少，按照【**SLA 服务等级协议**】的比例赔付，如服务时间为 **95~99.0%**，则赔付 **30%的费用**。

区域级 SLA

对于同时部署在同一区域（或者如果给定区域只有一个 AZ，则至少两个区域）的两个或多个 AZ 中的所有运行中实例的 Amazon EC2，AWS 将尽商业上的合理努力，使得在任何月度账单周期内，在每种情况下，Amazon EC2 在每个 AWS 区域的每月正常运行时间百分比至少达到 99.99%（“区域级 SLA”）。如果 Amazon EC2 不符合区域级 SLA，您将有资格获得如下所述的服务抵扣额度。

每月正常运行时间百分比	服务抵扣额度百分比
小于 99.99% 但等于或大于 99.0%	10%
小于 99.0% 但等于或大于 95.0%	30%
小于 95.0%	100%



▼【IT 财务模型】：

CAPEX ： (一次性)

资本支出——固定财务资产（一次性租服务器）、无形资产（运维、技术人员）

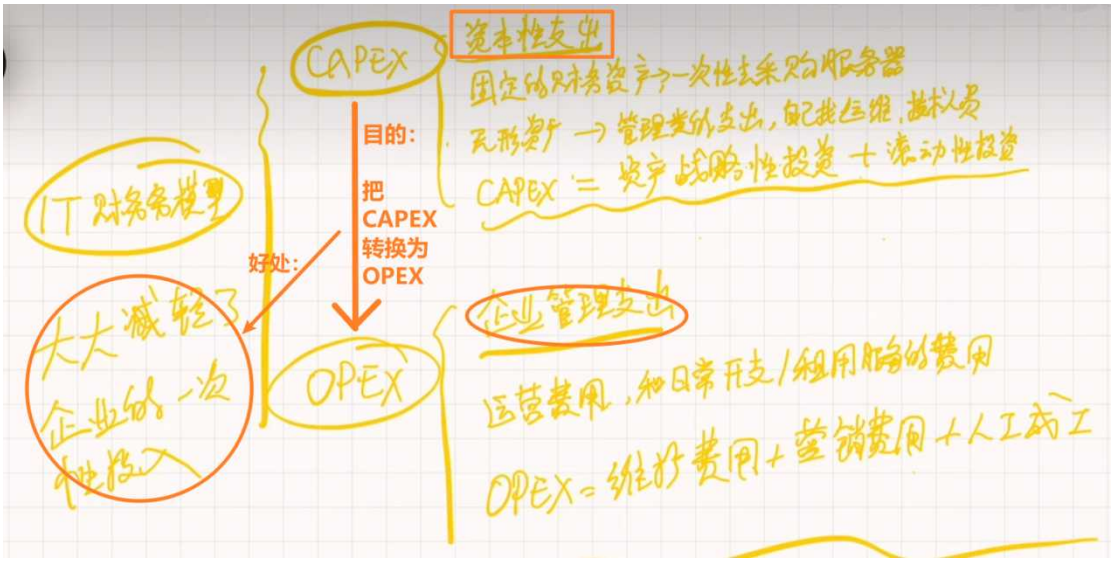
CAPEX=资产战略性投资+滚动性投资

OPEX ： (持续性)

运营支出——运营费用、日常开支/租用服务的费用

OPEX=维护费用+营销费用+人工成本

目的：【把 CAPEX 转换为 OPEX】——好处：大大减轻了企业的一次性支出。



▼【两个框架】CAF、WAF

● （CAF）Cloud Adoption Framework：

定义：利用最佳实践，让企业实现云转型 的实践指导（方法论）。

简述：【**理论框架**】；企业云转型(**企业上云**) 的 **方法论。**

■ **业务的【六大方面】：业务、人员、治理、平台、安全、运营**

Business、People / Personnel、Governance、Platform、Security、Operations (常简写为 Ops)

平台角度：

- **数据 架构/工程**：在云中构建、管理和优化数据。
- **云基础 架构设计**
- **应用程序 架构**
- **集成和持续交付（CI/CD）**
- **产品管理（变更、发布）**

运营角度——配置管理 + 补丁管理 + 事件管理——可观察性

治理角度——云采用业务成果和收益管理+程序和项目管理+风险管理（管理风险、确保合规及优化资源使用）

业务视角——组合管理 + 数据科学（通过数据分析和洞察，驱动商业决策）

■ **CAF【云转型阶段】**

阶段	英文	核心含义（一句话）	考试关键词
构想	Envision	明确为什么上云、上云要达成什么目标	**展示云如何加速业务成果**、价值、KPI
准备	Align / Prepare	建立组织、治理、安全、能力基础	组织**准备**、**治理**、技能
规划	Plan	制定迁移路线图与实施计划	迁移策略、优先级
发布	Launch	执行迁移并上线工作负载	迁移、部署、上线
扩展	Scale	大规模推广云使用，优化架构	自动化、扩展性
改进	Optimize	持续优化成本、性能、安全	成本优化、持续改进

● （WAF）Well-Architected Framework：

简述：【**技术框架**】。

■ **良好架构六大方面（WAF 的【支柱】-最佳实践）：**

可靠性、安全性、性能效率、成本优化、卓越运营、可持续性

Reliability、Security、Performance efficiency、Cost optimization、Operational excellence、Sustainability

共同点：都是指导企业上好云、管好云的白皮书、理论。

不同点：CAF 大方向理论指导；WAF 具体实践技术。

Table 1. The pillars of the AWS Well-Architected Framework

Name	Description
Operational excellence	The ability to support development and run workloads effectively, gain insight into their operations, and to continuously improve supporting processes and procedures to deliver business value.
Security	The security pillar describes how to take advantage of cloud technologies to protect data, systems, and assets in a way that can improve your security posture.
Reliability	The reliability pillar encompasses the ability of a workload to perform its intended function correctly and consistently when it's expected to. This includes the ability to operate and test the workload through its total lifecycle. This paper provides in-depth, best practice guidance for implementing reliable workloads on AWS.
Performance efficiency	The ability to use computing resources efficiently to meet system requirements, and to maintain efficiency as demand changes and technologies evolve.
Cost optimization	The ability to run systems to deliver business value at the lowest price point.
Sustainability	The ability to continually improve sustainability impacts by reducing energy consumption and increasing efficiency across all components of a workload by maximizing the benefits from provisioned resources and minimizing the total resources required.

二、■ 云产品资源

○【计算】

EC2、AS、Batch、ElasticBeanstalk、Lightsail、Lambda

15▼ EC2

●EC2 精简定义：

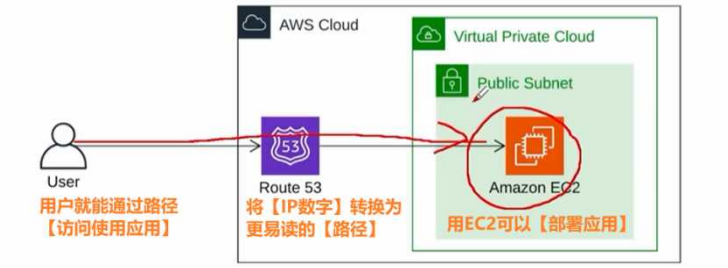
infrastructure 层服务；**弹性计算**(收缩、扩展)服务→实现**优化**；**降本、增加灵活性**。

我们：**租户**

产品：**实例**

●EC2 的运用（案例）：

（结合下图案例理解）①托管网站、②弹性计算加快处理数据、③数据备份



●付费模式：

On-Demand Instances **按需实例(按量)** ：贵、时间短(灵活)

Spot Instances **竞价实例(按量；会中断)** ：折扣+系统中断（**无工作负载**）

(可能被其它出价高的人**中断服务**，因此很多人把这个实例**【只作计算节点、不存数据】**而**【在预留实例上存储数据】**，组合使用实例→此称为**【无状态工作负载】**)

ReservedInstances **预留实例(长期 1 年以上；绑定)** ：便宜(72%折扣价)、时间长；**绑定 AZ、型号、地区**

Savings Plans **节省计划(长期 1 年以上；不绑定)** ：同预留；但**不绑定实例系列大小、操作系统、租赁或 AWS 区域**

↑**适用于可变负载（即业务量忽高忽低）、灵活地域和机器都能打折！**

Dedicated Host **专用主机** :当用户想要利用他们现有的**每个套接字、每个核心或每个虚拟机**的**软件许可证**，在 AWS 上运行 Microsoft Windows 服务器时需要。这允许**全面的自带许可证（BYOL）**支持。

付费模式：

On-Demand Instances 按需实例，使用按需实例，您只需要按小时或秒数支付计算容量，无需长期购买。

按需实例推荐用途：

- 希望拥有低成本和 EC2 提供的灵活性，且不想支付预付款或签订长期合同的用户
- 具有短期、难应付或无法预测且不能中断的工作负载的应用程序
- 首次在 EC2 上开发或测试的应用程序

https://aws.amazon.com/cn/ec2/pricing/on-demand/

Spot Instances 竞价实例，这种模式的核心特点包括折扣售卖和系统中断机制。用户可以以低于按量计费实例的价格购买这些资源，但存在一定风险，即系统可能会在后续自动回收这些折扣售卖的实例。与按需价格相比，可享受高达 90% 的折扣。

竞价型实例建议用于：

- 容错或无状态工作负载
- 可以在异构硬件上运行的应用程序
- 开始时间和结束时间灵活的应用程序

https://aws.amazon.com/cn/ec2/spot/pricing/

Reserved Instances 预留实例，一次性签署一年或三年期合同，并获得高达 72% 的账单折扣作为回报。预留实例与指定AZ和实例型号绑定。

按需容量预留建议用于：

- 需要容量保证的业务关键型事件或工作负载
- 需要满足高可用性监管要求的工作负载
- 灾难恢复

https://docs.aws.amazon.com/zh_cn/AWSEC2/latest/UserGuide/ec2-capacity-reservations.html

Savings Plans 节省计划，是一种灵活的定价模式，这种定价模式为EC2 实例的使用提供了更低的价格，而不考虑实例系列、大小、操作系统、租赁或 AWS 区域。与 EC2 预留实例一样，在一年或三年内使用特定数量计算能力的承诺，相比于按需价格，该模式可协助将费用减少多达 72%。

建议将节省计划用于：

- 承诺使用情况和稳定使用情况
- 希望在继续省钱的同时利用最新计算产品的用户

https://aws.amazon.com/cn/blogs/china/new-saving services/

16▼ Auto Scaling

- **Auto Scaling 精简定义：**

把用户多时达到**特定负载阈值**，扩展到【多个服务器】进行处理→**按设定好的条件，实时监控、并伸缩 资源容量**。

AWS Auto Scaling 是亚马逊提供的一项自动化服务，旨在根据应用程序的需求自动调整 AWS 资源的容量，以应对流量的变化，并确保应用程序的性能和可用性，同时最大程度地降低成本。

AWS Auto Scaling 可以监控您的应用程序并自动调整容量，从而以尽可能低的成本来保持稳定、可预测的性能。该服务可以提供一个简单而功能强大的用户界面，让您可以为 Amazon EC2 实例和 Spot 队列、Amazon ECS 任务、Amazon DynamoDB 表和索引以及 Amazon Aurora 副本等资源制定扩展计划。如果您已经在使用 Amazon EC2 Auto Scaling 来动态扩展 Amazon EC2 实例，那么现在可以将其与 AWS Auto Scaling 结合使用，为其他 AWS 服务扩展其他资源。有了 AWS Auto Scaling，您的应用程序就始终能在合适的时间获得合适的资源。

- **AS 补充：**

ELB（负载均衡）

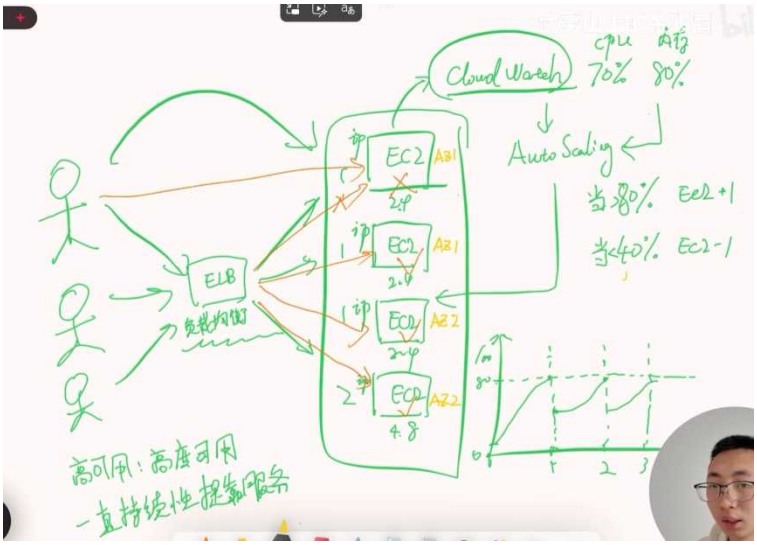
CloudWatch：

监控 CPU 和内存的负载。

例如，一台 CPU 负载到 80%时，自动加到 2 条服务器，每台 40%起的负载，三台 30%多起的负载……

高可用：

一直持续性提供服务。——业务**至少在【2 个可用区】(AZ1、AZ2)上**。

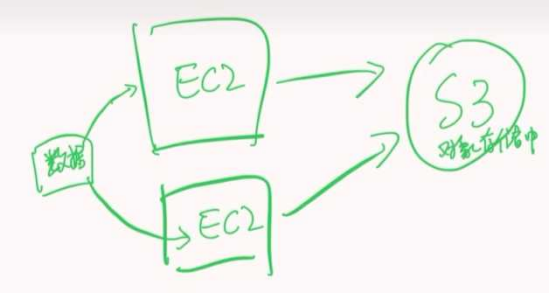


- **【请求→…→存储】的流程：**

用户请求 → 数据库 → ELB →EC2 → S3（对象存储）

*注意①： 也就是说，EC2 不接收请求，而是开机后从数据库中拉取已有请求。
因此可以经过 ELB 均衡处理分配负载。

*注意②： 有【用户交互】才需要设置【ELB(负载均衡)】。



17▼ Ligthsail（低配版 EC2）=AWS 的“新手/小项目一站式服务器”——快速搭建

关键词：

固定价格——月费清晰、可预测

一体化——**计算 + 存储 + 网络**

预配置——提供 WordPress / LAMP 等模板

可升级——可迁移到 EC2

Amazon Lightsail 是亚马逊提供的一种简化的虚拟私有服务器（VPS）服务，旨在帮助开发人员快速、轻松地搭建和管理虚拟服务器。Lightsail 提供了预配置的计算资源、网络、存储和数据传输选项，用户可以通过简单的界面选择所需的配置，轻松部署应用程序和网站，而无需担心复杂的基础架构管理细节。

精简定义：

低配版 EC2，但有预配置模板、上手更快更简单。→**简易化配置服务器**（基本最高 16 核 32G）

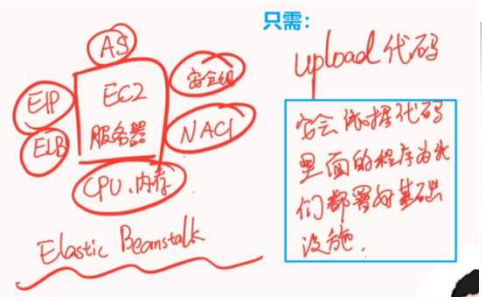
***Lightsail 和 EC2 都是 VPS（虚拟私有服务器），即都是【服务器】。**

***EC2 最高可几百 G，而轻量化的 Lightsail 基本最高 16 核 32G。**

18▼ Elastic Beanstalk（自动部署基础设施）

精简定义：

【**自动化**帮我们**部署基础设施**】，我们只需【**上传代码**】即可。



★★★AWS Elastic Beanstalk

AWS Elastic Beanstalk 是一项用于简化应用程序部署和管理的托管服务。它允许开发人员上传他们的应用程序代码，并自动处理底层的部署、扩展、负载平衡和监控任务，从而使开发人员能够专注于应用程序的开发而不必担心基础架构的管理细节。

借助 Elastic Beanstalk，您可以在AWS云中快速部署和管理应用程序，而不必了解运行这些应用程序的基础设施。Elastic Beanstalk 可降低管理的复杂性，但不会影响选择或控制。您只需上传应用程序，Elastic Beanstalk 将自动处理有关容量预配置、负载均衡、扩展和应用程序运行状况监控的部署细节。

Elastic Beanstalk 支持在 Go、Java、.NET、Node.js、PHP、Python 和 Ruby 中开发的应用程序。在部署应用程序时，Elastic Beanstalk 会构建选定的受支持的平台版本，并预配置一个或多个AWS资源（如 Amazon EC2 实例）来运行应用程序。

案例展示

Case1: Web 应用部署
一家初创公司开发了一个新的 Web 应用，并希望将其部署到云端。他们选择使用 AWS Elastic Beanstalk 来托管他们的应用程序。通过 Elastic Beanstalk，他们可以轻松地上传应用程序代码，并利用 Elastic Beanstalk 的自动扩展和负载平衡功能来管理应用程序的部署和运行。这样，他们可以快速地将应用程序部署到云端，并在应用程序的需求增加时自动扩展计算资源，以确保应用程序的稳定性和性能。

Case2: API 服务部署
一家软件公司开发了一组 API 服务，用于支持其移动应用程序和网站。他们选择使用 AWS Elastic Beanstalk 来部署和管理他们的 API 服务。通过 Elastic Beanstalk，他们可以轻松地将 API 服务部署到云端，并利用 Elastic Beanstalk 的自动负载平衡和监控功能来确保 API 服务的高可用性和可靠性。这样，他们可以专注于开发和维护 API 服务的功能，而无需担心基础架构的管理问题。

Case3: 后端应用部署
一家电子商务公司需要将其后端应用程序部署到云端，并确保应用程序的高可用性和可扩展性。他们选择使用 AWS Elastic Be... 来管理他们的后端应用程序。通过 Elastic Beanstalk，他们可以轻松地部署后端应用程序，并利用 Elastic Beanstalk 的... 来根据应用程序的需求动态分配计算资源。这样，他们可以确保后端应用程序始终能够满足用户的需求，并在流量增加时... 保性能。

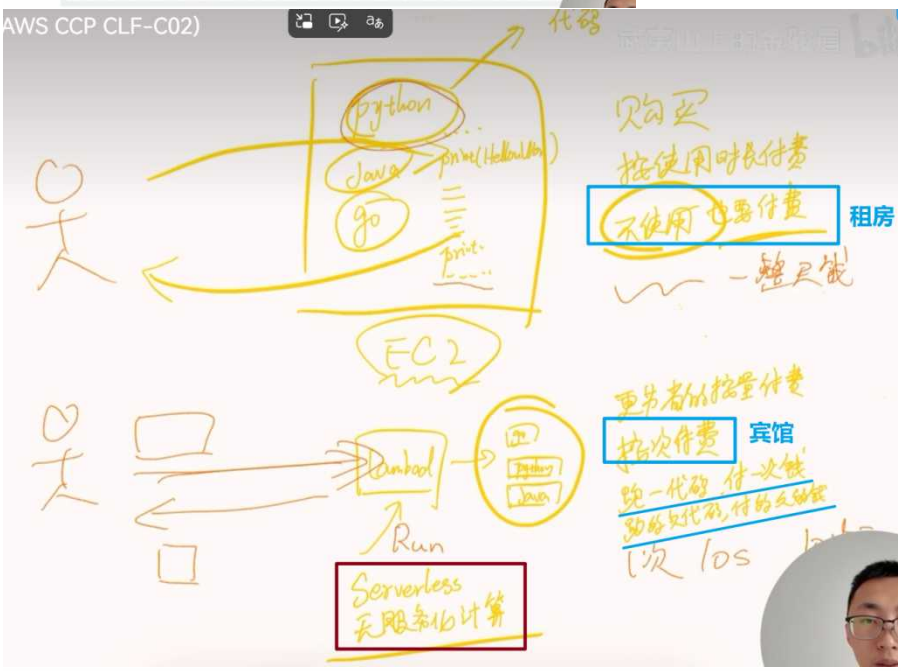
19▼ Lambda（无服务器化；Lambda 函数的最大执行时间是 15 分钟；按次节省；事件触发）

精简定义：

按次付费，比 EC2（按时长付费）**更节省的服务器**；**关键词**是【**无服务化计算**】和【**事件触发**】。

★★★AWS Lambda

AWS Lambda 是亚马逊提供的一项无服务器计算服务，它允许开发人员在无需管理服务器的情况下运行代码。
Lambda 可以自动扩展以处理任何规模的请求，并且只会收取实际执行代码的费用，而不会收取任何预订或固定费用。



案例展示

Case1: 图像处理服务
一家社交媒体平台需要对用户上传的图像进行处理，例如缩放、裁剪或添加水印。他们使用 AWS Lambda 创建了一个图像处理服务。当用户上传图像时，Lambda 函数会自动触发并对图像进行处理，然后将处理后的图像存储回亚马逊 S3 存储桶。这样，他们可以根据需要处理任意数量的图像，并且只需支付实际执行代码的费用。

Case2: 日志分析
一家网络安全公司需要对大量的网络日志进行实时分析，以检测潜在的安全威胁。他们使用 AWS Lambda 创建了一个日志分析服务。当新的日志数据到达亚马逊 CloudWatch 日志组时，Lambda 函数会自动触发并对日志数据进行分析，识别可能的安全事件，并发送警报通知给安全团队。通过 Lambda，他们可以实现高效的实时日志分析，而无需担心基础设施的管理。

Case3: 后端 API
一家在线零售商需要构建后端 API 来支持其移动应用和网站。他们使用 AWS Lambda 创建了一组后端 API。每个 API 端点都与一个 Lambda 函数关联，负责处理特定的请求。例如，一个 Lambda 函数负责处理用户注册请求，另一个负责处理订单查询请求。这样，他们可以快速搭建可扩展的后端 API，而不必担心服务器的管理和扩展。

作成者：朱晓駿 | 作成期间：2026 年 1 月

20▼ AWS Batch

关键词：

机器学习模型训练、模拟、规模分析 的 **批处理**。

应用领域：生命科学（大量排列组合研究靶向药）、**自动驾驶系统**（多容器组合作业，减少准备、加快开发）

★★☆AWS Batch

AWS Batch 是亚马逊提供的一项批量计算服务，旨在帮助用户高效地处理大规模的计算工作负载。AWS Batch 可以自动调度、运行和监控批处理作业，用户无需管理底层的计算资源，可以专注于编写和提交作业。AWS Batch 提供了灵活的配置选项，包括不同类型的计算环境和作业队列，以满足不同应用场景的需求。

案例展示

Case1: 科学计算

一个科研团队需要对大量的科学数据进行分析和计算。他们利用 AWS Batch 创建了一个计算环境，并将计算任务提交到 AWS Batch 的作业队列中。AWS Batch 自动调度并运行这些任务，并根据需求动态分配计算资源。通过 AWS Batch，他们可以高效地完成科学计算任务，加速研究进程。

Case2: 生命科学

生物制药和基因组学公司依赖高性能计算来推动产品上市。AWS Batch 简化了不同应用领域的操作，例如计算化学、临床建模、分子动力学，以及基因组测序测试和分析等。在药物筛选期间，AWS Batch 使研究科学家能够高效地搜索小分子库，从而确定最有可能绑定到药物靶标（通常是蛋白受体或酶）的结构。这一过程有助于药物设计，有潜力促进更有效药物和疗法的开发。对于 DNA 测序，在生物信息学家完成对基因组序列的初步分析并生成原始文件后，他们可以利用 AWS Batch 自动执行二次分析，包括将原始 DNA 读取内容汇编成完整的基因组序列，从而减少错误。

Case3: 自动驾驶数据

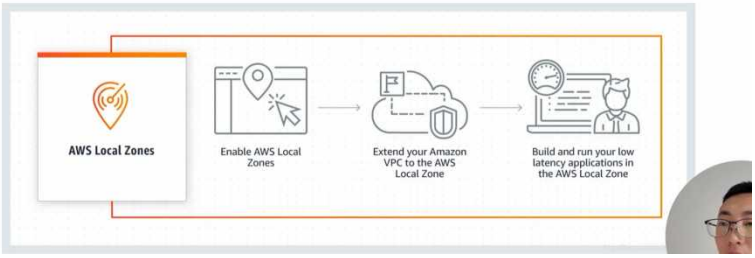
汽车公司在开发和测试自动驾驶汽车（AV）以及高级驾驶员辅助系统（ADAS）时依赖模拟。工程师使用容器，将模拟中的每项要素（车辆传感器、交通和 3D 环境）建模成更小的模块化组件。由于能够使用 AWS Batch 运行多容器作业，您可以享受到 AWS 的高级扩展、调度和成本优化功能，而无需将您的系统重构为一个复杂的整体式容器。相反，您可以使用多个更小的模块化容器代表不同的系统组件。此功能通过减少作业准备步骤来加快开发速度，消除了构建额外内部工具的需求，并简化了开发（Dev）、IT 运营（Ops）和调试。

21▼ Local Zones

关键词：

☆☆☆**AWS Local Zones**

AWS Local Zones 是亚马逊提供的一种区域性基础设施扩展服务，它提供了与 AWS 区域相似的功能和服务，但位于距离主要 AWS 区域更近的地理位置，以降低延迟并提高性能。Local Zones 允许客户在更接近其用户和数据的地方部署应用程序，并提供与主要 AWS 区域相同的可靠性和安全性。



22▼ AWS Outposts（混合云【云→本地】；低延时）

关键词：

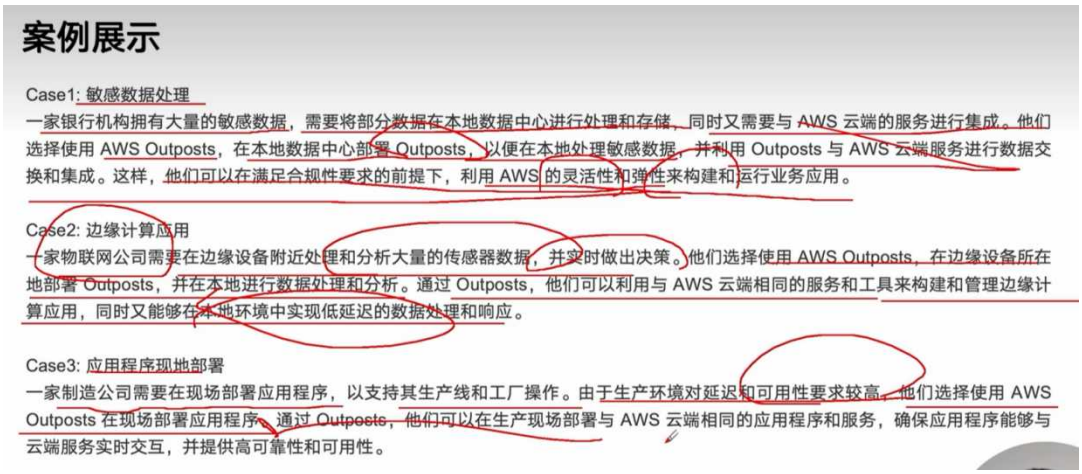
混合云；【**云服务**→部署到→**本地数据中心**】（将 **AWS 基础设施和服务部署到本地数据中心**）

将 AWS 计算和存储 **扩展到 本地数据中心**→**低延迟**；

使能**在 本地环境中构建和运行 AWS 云端一致的应用**。

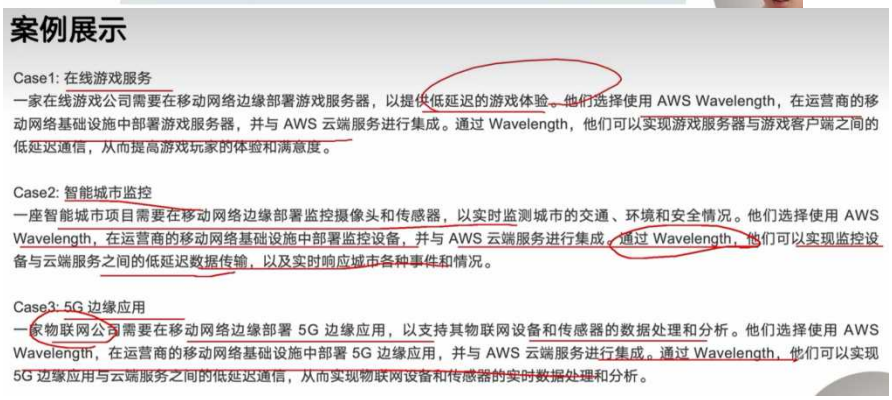
应用场景：

本地处理敏感数据，AWS 云服务器交换和集成数据。



23▼ AWS Wavelength（5G 边缘计算→低延迟(数据传输)）

关键词：5G 边缘计算基础设施



○【存储】

24▼ EBS（= 一块 硬盘）→文件是操作系统自己在 EBS 上建的

持久性；高可靠性；

可根据需求 弹性扩容；

EBS 只可挂载到 1 个服务器上。

25▼ EFS（=多个实例 可**共享访问 同数据**）

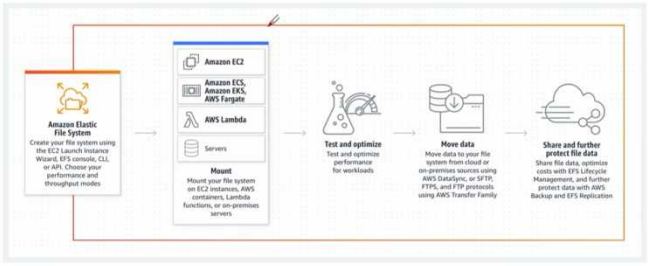
精简：**多实例**访问**同数据**。

应用：**文件共享**（方便多实例工作协作）、**数据文件备份**（云端**安全可靠性、且可按需扩容**）

★★★Amazon Elastic File System (Amazon EFS)

Amazon Elastic File System (Amazon EFS) 是亚马逊提供的一种托管文件存储服务，可在多个 EC2 实例之间提供可扩展的、高性能的共享文件存储。

Amazon EFS 是一种通用的、高可用性的文件存储服务，适用于各种不同类型的工作负载，如应用程序部署、文件共享、数据备份等。



案例展示

Case1: 应用程序部署

一家软件开发公司正在部署其新的微服务应用程序。他们选择使用 Amazon EFS 来存储应用程序的共享配置文件和静态资源，以便多个 EC2 实例可以共享相同的文件系统。这样，他们的开发团队可以方便地访问和更新共享的配置文件，同时确保所有实例都使用相同的配置。

Case2: 媒体文件存储

一家媒体公司需要存储大量的媒体文件，如音乐、视频和图像。他们使用 Amazon EFS 创建了一个共享文件系统，将其挂载到多个 EC2 实例上。这样，他们的团队可以方便地共享和访问媒体文件，并且可以在多个实例之间实现协作和文件共享，从而提高工作效率。

Case3: 数据备份存储

一家企业需要一个可靠的文件备份解决方案，以备份其重要数据。他们使用 Amazon EFS 创建了一个备份文件系统，并将其挂载到他们的备份服务器上。这样，他们可以将备份数据存储在 Amazon EFS 上，以确保数据的安全性和可靠性，并且可以根据需要扩展存储容量以应对备份数据的增长。

26▼ FSx（高性能：低延迟+高吞吐量；文件储存、共享系统）

关键词：低延迟；高吞吐量（大规模、高性能计算）；可扩展

本质：高性能**文件共享**系统。

★★☆Amazon FSx

Amazon FSx 是亚马逊提供的一种托管文件存储服务，它为用户提供了可扩展的、高性能的文件存储解决方案，可用于在云中运行各种工作负载。

Amazon FSx 通常提供更高性能的文件存储，适用于需要低延迟和高吞吐量的工作负载，如大规模数据分析、高性能计算（HPC）等。

案例展示

Case1: Windows 文件共享

一家企业需要在 AWS 上部署 Windows 文件共享服务，以便员工可以方便地共享和访问文件。他们选择使用 Amazon FSx for Windows File Server 来部署文件共享服务。Amazon FSx 提供了一个托管的 Windows 文件系统，可以轻松地与现有的 Windows 环境集成，并为用户提供了与本地文件共享服务类似的体验。

Case2: HPC 文件存储

一家科学研究机构需要一个高性能的文件系统来支持其高性能计算 (HPC) 工作负载。他们选择使用 Amazon FSx for Lustre 来部署 Lustre 文件系统。Amazon FSx for Lustre 提供了一个可扩展的、高性能的文件系统，可用于存储和处理大规模的科学数据，以支持复杂的 HPC 应用程序和工作流程。

Case3: SAP 文件共享

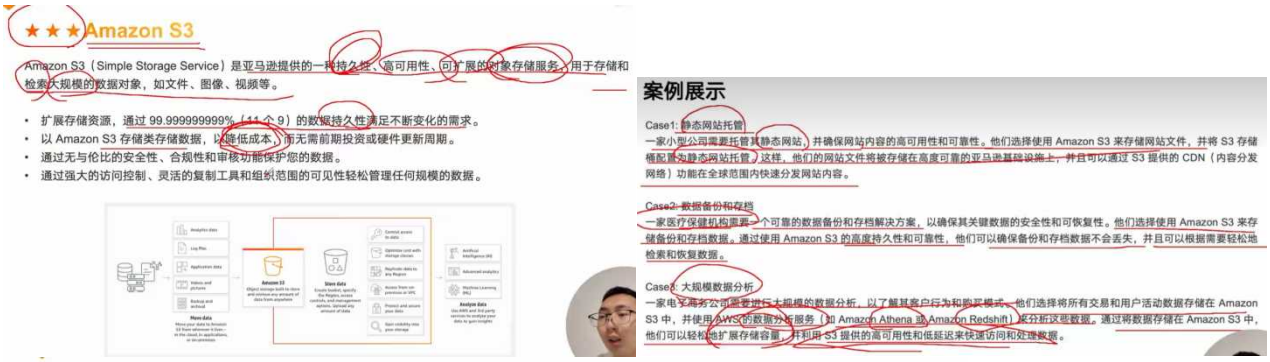
一家制造公司正在迁移其 SAP 系统到 AWS 上，并需要一个可靠的文件共享解决方案来存储 SAP 数据和文档。他们选择使用 Amazon FSx for Windows File Server 来部署 SAP 文件共享服务。Amazon FSx for Windows File Server 提供了与 SAP 系统兼容的高可靠性、可扩展的 Windows 文件系统，可以满足其对文件共享的需求，并确保数据的安全性和可靠性。

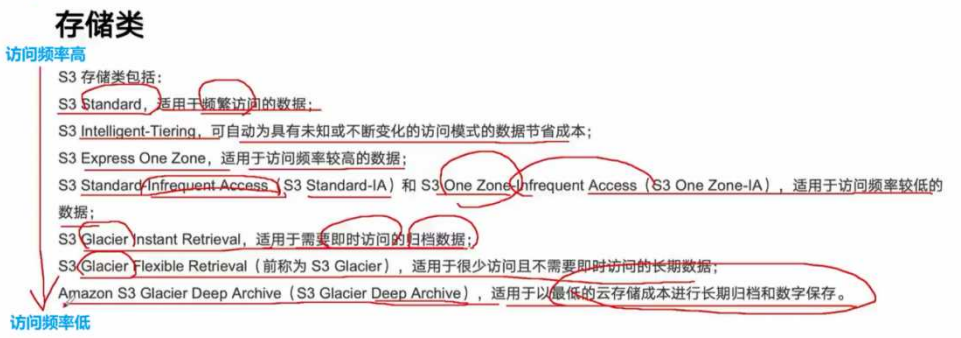
27▼ S3（关键词：检索大规模数据、数据分析；对象存储；降低成本；无服务器）= 云端网盘 / 文件仓库

* **持久性、高可用性(高度耐用)、可扩展性** 的 **对象存储**服务。
* **检索**大规模的数据对象（如文件、图像、视频等）。

*** S3（简单存储服务）**是**无服务器**的，用户**无需管理存储服务器**

应用场景：静态网页托管；数据存储；大规模(PB)数据分析





■ **对象**存储（S3）、**块**存储（EBS）、**文件**存储（EFS）**具体是 存什么** 呢?

类型	AWS 服务	本质在“存什么”
对象存储	Amazon **S3**	**文件本身**（一个个独立文件）
块存储	Amazon **EBS**	**磁盘本身**（给服务器用的硬盘）
文件存储	Amazon **EFS**	**共享文件夹**（多人/多机共用）

28▼ S3 Glacier（存储 长期但很少访问 的数据）

★★★Amazon S3 Glacier

Amazon S3 Glacier 是 Amazon S3 的一种存储类别，专门用于长期存储和归档数据，具有低成本、高可靠性和安全性的特点。Glacier 主要用于存储需要长期保存但很少需要访问的数据。

目前S3 Glacier有3种存储类：
Amazon S3 Glacier Instant Retrieval 即时检索存储类。即时检索为每季度访问一次且需要毫秒级检索的长期数据提供成本最低的存储。成本最多降低 68%。
Amazon S3 Glacier Flexible Retrieval 灵活检索存储类。为每年访问 1-2 次且异步检索的归档数据提供低成本存储，成本最多降低 10%。
Amazon S3 Glacier Deep Archive 深度存档存储类。提供最低成本的存储，比灵活检索低 75%，适用于每年访问少于一次且异步检索的长期归档数据。

案例展示

Case1: 数据归档
一家法律公司需要对其大量的案件文件进行长期归档，并确保这些数据安全可靠。他们选择使用 Amazon S3 Glacier 来存储归档文件。通过将文件存储在 Glacier 中，他们可以以极低的成本存储大量数据，并且可以根据需要随时检索和访问这些归档文件。

Case2: 科学研究数据存储
一个科学研究机构需要存储其历史上产生的大量科学数据，并确保这些数据长期保存和可靠。他们选择使用 Amazon S3 Glacier 来存储科学研究数据。通过将数据存储在 Glacier 中，他们可以以低廉的价格存储大规模的数据，并且可以利用 Glacier 提供的安全性和持久性来确保数据的安全和完整性。

Case3: 数字存档
一个图书馆或博物馆需要对其珍贵的数字文物进行长期存储和保存。他们选择使用 Amazon S3 Glacier 来存储这些数字存档。通过将数字文物存储在 Glacier 中，他们可以以极低的成本保存大量的数字文物，并且可以通过 Glacier 提供的高度可靠性和安全性来确保这些文物的长期保存。

29▼ Storage Gateway (混合云网关——【本地-云】互相传输【备份文件】 | 本地部署 SG + 云端与 S3 集成)

- 混合云：**本地 备份到 云端、或 云端 备份到 本地** ——（类似 百度云盘备份文件）
- 关键词：**混合云→ 本地部署 SG + 云端与 S3 集成；提供几乎无限的云储存访问权限。**

* 可为本地应用程序提供**对存储在 AWS 云中的数据的低延迟访问**。它使本地应用程序能够访问和处理云中的数据，确保数据具有最小的延迟可用。

★★☆AWS Storage Gateway

AWS Storage Gateway 是一项服务，它允许您在本地环境和 AWS 云之间无缝地集成存储解决方案。它提供了一种简单、安全的方式，让您可以将本地应用程序连接到云存储服务，如 Amazon S3、Amazon Glacier、Amazon EBS 等。

案例展示

Case1: 混合云备份
一家中型企业需要一种备份解决方案，以确保其关键数据在本地和云端之间具有高可用性和冗余性。他们使用 AWS Storage Gateway 部署了一个混合云备份解决方案。他们将 Storage Gateway 部署在本地数据中心，并配置了备份策略，将数据备份到 Amazon S3。这样，他们可以通过 AWS 云服务来管理备份数据，并在需要时轻松地将备份数据恢复到本地或云端。

Case2: 本地文件共享
一家小型工程公司需要一个简单而有效的文件共享解决方案，以便员工在不同的办公地点之间共享和访问工程图纸和文件。他们使用 AWS Storage Gateway 部署了一个本地文件共享解决方案。他们将 Storage Gateway 部署在本地办公室，并将其配置为与 Amazon S3 集成。这样，员工可以通过本地网络访问文件共享，并且所有数据都存储在 AWS 云中，具有高可用性和可靠性。

Case3: 灾难恢复
一家大型企业需要一个可靠的灾难恢复解决方案，以确保其业务连续性和数据安全性。他们使用 AWS Storage Gateway 部署了一个灾难恢复解决方案。他们在本地数据中心和 AWS 云中各部署一个 Storage Gateway，并设置数据复制策略，将关键数据备份到云端。这样，他们可以在发生灾难时快速恢复业务，并确保数据的安全性和可靠性。

30▼ AWS Backup（【云上】存储）

- **案例：**

  数据库备份 ；

  存储卷备份（**搬家公司一点点搬**）；

  服务器镜像备份（**直接屋子整个平移**）

■ Backup & Storage Gateway 区别：

Backup ——【**云上**】存储。

Storage Gateway ——网关，【**本地-云**】互相传输存储。

●31【存储 总结】：

只有 **1 块硬盘**，只能在 1 个 EC2 上使用—— **EBS**

多个服务器 同时访问 1 块硬盘 —— **EFS**（共享文件）

高性能 同时访问 1 块硬盘 —— **FSx**

海量数据、**低成本**、**经常使用** —— **S3**

不经常使用、深度归档 —— **S3 Glacier**

混合云 备份 —— **Storage Gateway**

云上 备份 —— **Backup**

○【数据库】

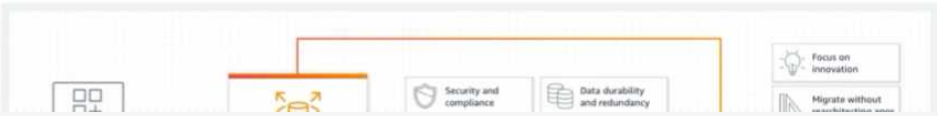
32▼ Amazon RDS (关系型数据库)

- **把 RDS 运维托管给 AWS。**

★★★Amazon RDS

Amazon Relational Database Service（RDS）是亚马逊提供的一种托管关系型数据库服务，可用于在云中轻松设置、运行和扩展关系型数据库。

目前支持：
RDS for MySQL
RDS for PostgreSQL
RDS for MariaDB
Amazon Aurora
RDS for SQL Server
RDS for Oracle
RDS for Db2



案例展示

Case1: 电子商务网站
一家电子商务公司需要一个可靠的数据库解决方案来存储其产品信息、订单数据和用户信息。他们选择使用 Amazon RDS 来托管他们的数据库。他们创建了一个 MySQL 或 PostgreSQL 数据库实例，并将其用于存储产品目录、订单记录和用户信息。通过 Amazon RDS，他们可以确保其数据库具有高可用性、可伸缩性和安全性，从而保证其网站的顺利运行。

Case2: 移动应用后端
一家创业公司正在开发一款新的移动应用程序，并需要一个可靠的后端数据库来存储用户数据、内容信息和应用状态。他们选择使用 Amazon RDS 来托管他们的后端数据库。他们创建了一个 Amazon Aurora 或 Microsoft SQL Server 数据库实例，并将其用于存储用户配置、应用内容和交互数据。通过 Amazon RDS，他们可以轻松地管理和扩展其后端数据库，以满足不断增长的用户需求。

Case3: 企业内部系统
一家大型企业需要一个可靠的数据库解决方案来支持其内部业务系统，如人力资源管理、财务管理和供应链管理。他们选择使用 Amazon RDS 来托管他们的企业数据库。他们创建了一个 Oracle 或 Microsoft SQL Server 数据库实例，并将其用于存储企业数据和业务应用数据。通过 Amazon RDS，他们可以确保其数据库具有高可用性、可伸缩性和安全性，从而支持其内部业务系统的顺利运行。

■ MySQL & Aurora 区别：

MySQL ——开源

Aurora ——Amazon 自研 → 非开源 + 性能优于 MySQL + 费用低于 MySQL + 可自动扩缩容

33▼ Aurora (兼容 + 性能和可靠性 更高 + 价格更低 + 可自动扩缩容)

- 兼容 MySQL 和 PostgreSQL 的功能；
  且 性能（吞吐量）和可靠性 更高；
  且 价格 是 商业数据库的十分之一。

■ 有服务器 & 无服务器 概念：

【有服务器】本义 = 指需要自己管理服务器 ≈手动预设

——手动 Provision (预置) 或管理 EC2 实例。

【有服务器化】≈预定付费

——时间段 A 内定量，不管时间段 A 内有没有使用，都按时间付钱。

【无服务器】本义 = 指不需要自己管理服务器 ≈自动计算

——AWS 自动处理计算资源的伸缩和维护，无需 Provision(预置)或管理 EC2 实例。

【无服务器化】≈按需付费

——用多少，付多少钱。

34▼ DynamoDB (无服务器+NoSQL(键值对型)+完全托管+任何规模下均个位数毫秒级的性能)

- 无服务器+NoSQL+完全托管+任何规模下均个位数毫秒级的性能
- 冷启动——即 瞬间启动
- DynamoDB 是 键值对型数据库（key-value）

冷启动和热启动好像是在胡扯，感觉类似于热插拔那种，不关机插拔，表示还有状态，个人理解这边冷启动就是表示没有状态的，热启动就是表示有状态的环境下

■ 关系型 DB & 非关系型 DB 概念：

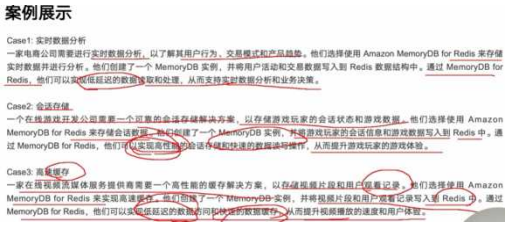
关系型 DB ——表格，行列 11 对应。 ——适用关系型 DB：用户订单数据

非关系型 DB ——见下图多种都是 ——适用非关系型 DB：点赞评论、游戏进度成就、设备状态配置

* 【内存型数据库】是 NoSQL 的一种。▼

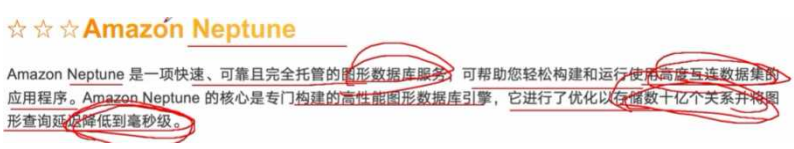
35▼ MemoryDB for Redis (内存型 NoSQL；兼容 Redis，可实现超快性能)

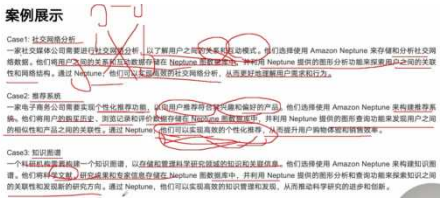
- 案例：实时数据分析；会话存储(临时状态存储)；高速缓存
- **做题技巧：见到需要【低延迟、缓存】类型的 DB，就选内存型数据库。**



36▼ Neptune (高性能图形 NoSQL；高关联度；存储数十亿关系，查询延迟降到毫秒级)

- **高性能图形 NoSQL；高关联度；存储数十亿关系，查询延迟降到毫秒级**
- **案例：社交网络分析（分析关联度→个性推荐）；知识图谱**





●37【数据库 总结】：

企业·用户管理 ——RDS

性能高、费用低 ——Aurora

游戏进度、**点赞评论** →键值对 DB ——DynamoDB

会话、状态信息 →内存型 DB ——MemoryDB for Redis

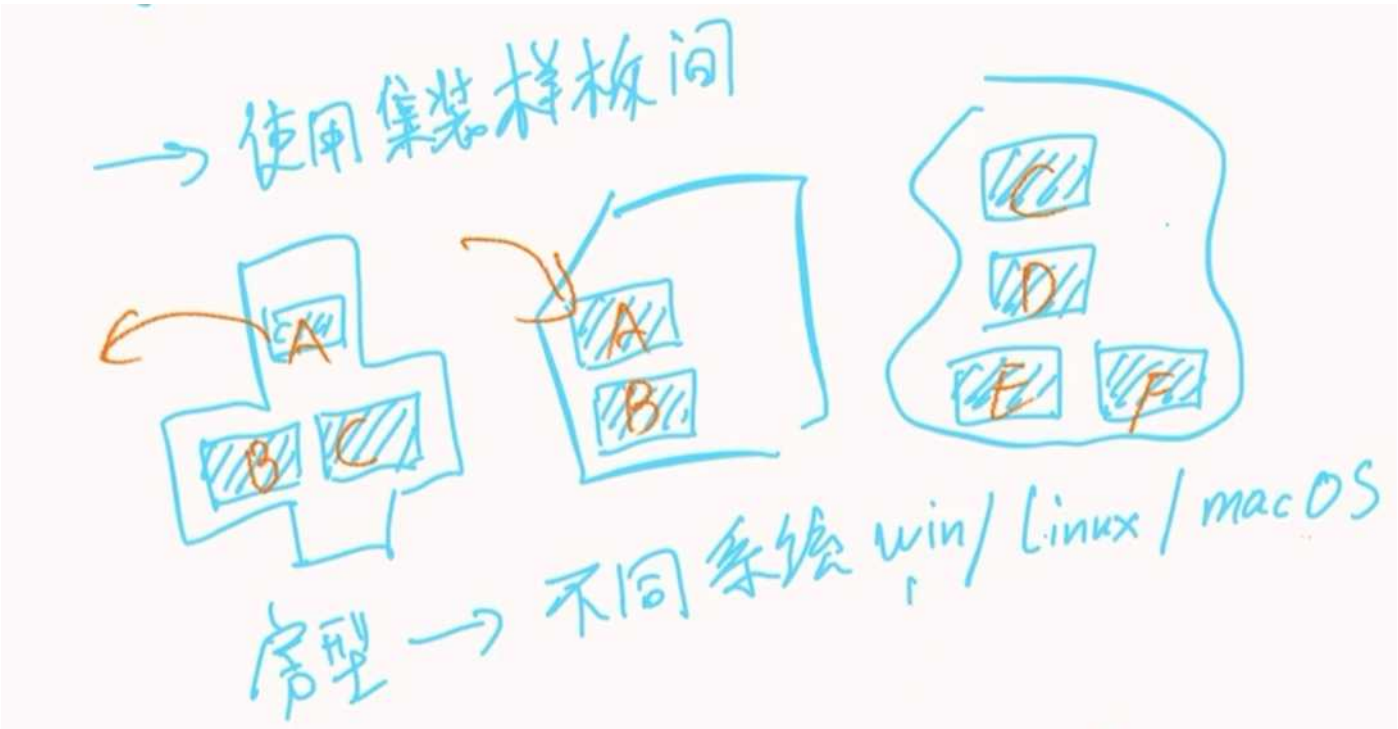
【**关联度**】企业关系网、个性化推荐、知识图谱 →图 DB ——Neptune

○【容器】

■ 概念：什么是【容器】？

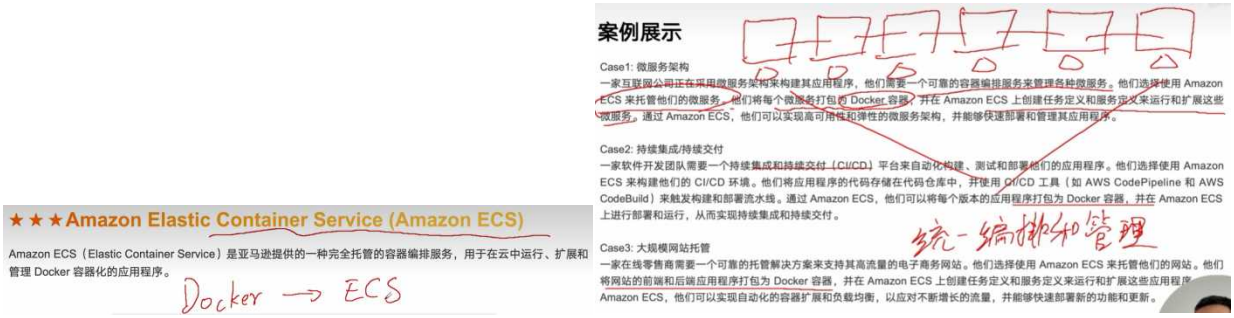
答：容器 = 把“应用 + 运行环境”一起打包，在哪都能用同样方式运行

标准定义：



38▼ ECS (=云上托管的 Docker 容器——打包应用进容器)

- Elastic Container Service
- 案例：把微服务打包为 Docker 容器来统一编排和管理、持续集成和交付(CI/CD)、大规模网站托管



39▼ EKS (=云上托管的 K8S——编排容器；管理调度容器，简化跨环境应用管理)

- Elastic Kubernetes(K8s) Service
- 案例：容器编排服务来管理微服务；持续集成和交付(CI/CD)自动化构建、测试、部署应用；混合云部署(托管工作负载、管理 K8S 集群，统一容器管理和部署，简化跨环境应用管理)

40▼ Fargate (无服务器容器计算引擎；并列概念 是 EC2 弹性计算云)

- Fargate 含义：无服务器容器托管 / 无服务器计算引擎

  ECS 的 EC2 托管　→云主机　[要自己管理　→有更大自主可控性]

  ECS 的 Fargate 托管　→无服务化　[无需自己管理　→自动部署、扩展、负载平衡]

■ 解题技巧：微服务 or 持续集成/交互

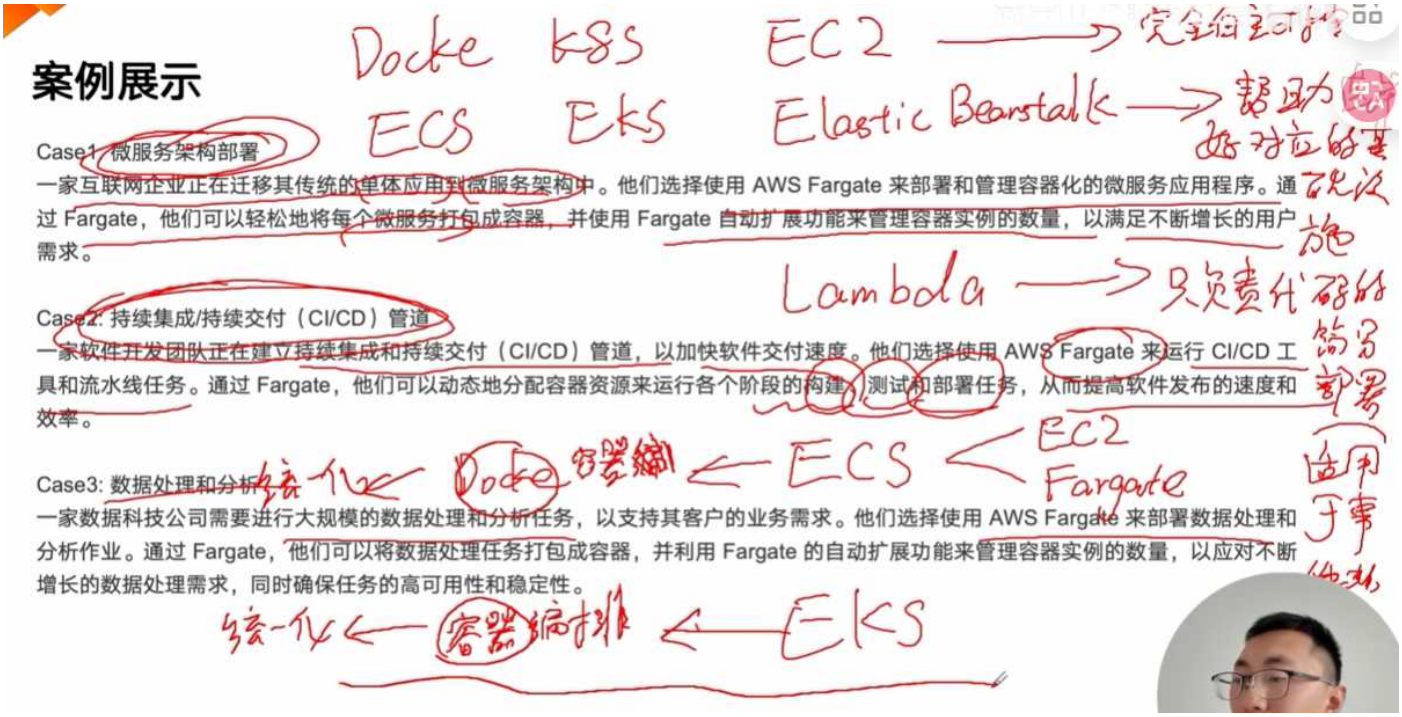
见【微服务架构部署】和【持续集成/持续交互(CI/CD)】

——就想到【Docker→ECS】【K8S→EKS】

■ 区分：EC2 & Elastic Beanstalk & Lambda

- **EC2** ——完全**自主可控**
- **Elastic Beanstalk** ——**帮助选择好**对应的基础设施，管理还是**自己管理**。
- **Lambda** ——只负责**代码的简易部署**（适用于**事件触发**类型）

- **ECS**——云上 **容器**
- **EKS**——云上 **容器编排**



41▼ ECR (=仓库 →注册表的托管)

- **Elastic Container Registry(注册表、登记处)**
- **案例：** **▲容器镜像存储和部署、**
  ▲持续集成/交付（commit→推送到 ECR→镜像构建到云端 Docker）、
  ▲多环境部署（不同环境 存储到 不同镜像仓库→选择不同的、更稳定的版本镜像，进行版本控制管理）

*** Docker 镜像=环境**

●42【容器 总结】：

- ECS ——云上 **容器（微服务；CI/CD）**
- EKS ——云上 **容器调度、编排**
- ECR ——**镜像仓库**
- Fargate ——**无服务器化容器**服务

  ECS 的 **EC2** 托管 →**云主机** [**要自己管理** →**有更大自主可控性**]

  ECS 的 **Fargate** 托管 →**无服务化** [**无需自己管理** →**自动部署、扩展、负载平衡**]

  - **EC2** ——完全**自主可控**
  - **Elastic Beanstalk** ——**帮助选择好**对应的基础设施，管理还是**自己管理**。
  - **Lambda** ——只负责**代码的简易部署**（适用于**事件触发**类型）

○【迁移和传输】

数据库和主机→迁移上→**云**

或

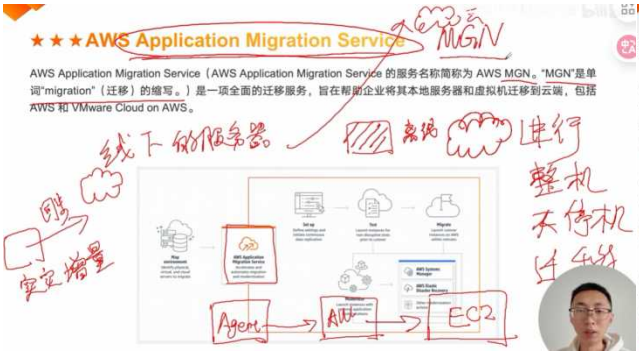
其它云→迁移到→**AWS 云**

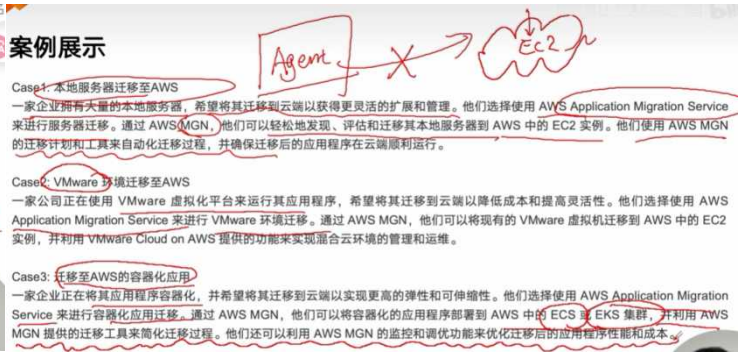
43▼ Application Discovery Service (Discovery Agent；管理类软件)

- **运行 Discovery Agent 收集信息→生成信息清单→洞察并管理软件**

44▼ AMS / MGN (【服务器】线下迁移上云：不停机(不断同步、实时增量)迁移)

- **全称：Application Migration Service**（区分【AMS】：AWS 托管服务——Managed Services（AMS）——托管 AWS 环境的日常运维）
- **功能：将线下服务器迁移上云端——AMS**
- **迁移方式：**进行**整机的(服务器)不停机(不断同步、实时增量)**迁移





■ 补充：Discovery Agent 的【服务器迁移】过程

【Agent】——不断地收集新增数据到云中 EC2

→ 发现 云中 EC2 和 线下数据 一致时

→ 切断 Agent 和 EC2 的连接

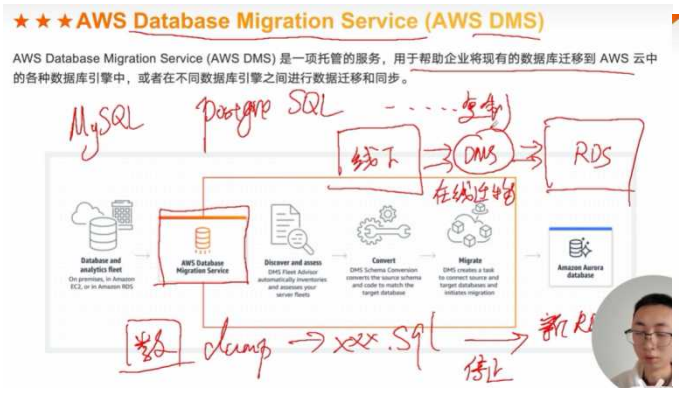
→ 把用户流量引导到 EC2 中去

→ 完成【服务器迁移】

45▼ DMS（【数据库】在线迁移）

- Database Migration Service
- 【数据库】在线迁移—— 在线 = 不停机
- 案例：跨数据库引擎迁移（源端→目标端）；数据库版本升级；跨云迁移 ↓

迁移（用 DMS）	迁移原因	源端	目标端
跨 数据库引擎	Aurora 高可用、可扩展	MySQL	AWS 的 Aurora
数据库 版本升级	数据量大→手动成本和风险高	旧版本 Oracle	
跨 云	服务质量、费用	某云（运行着 SQL Server）	AWS 的 RDS SQL Server



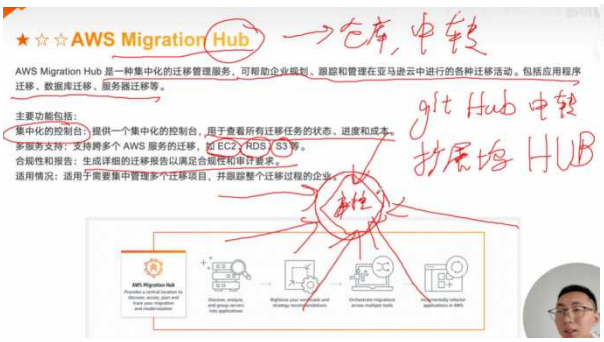
46▼ Migration Hub（集中化控制台，不是产品）

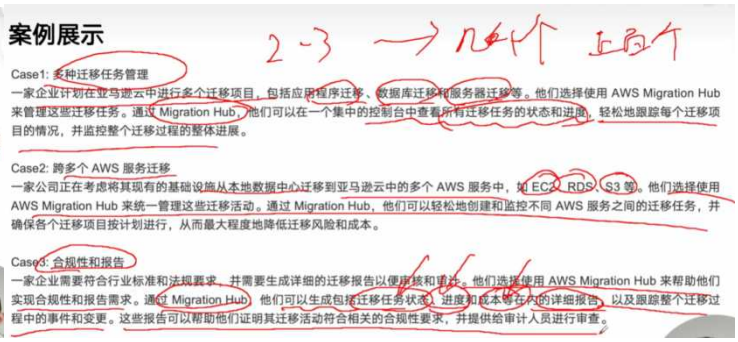
- 【Hub】：中转的仓库。
- 案例：

  多种迁移任务管理 （程序、数据库、服务器 的迁移）、

  跨多个 AWS 服务迁移（EC2、RDS、S3 的迁移）、

  合规性报告 （迁移任务状态、进度、成本等的详细报告）





47▼ SCT (数据库迁移工具：换 数据库引擎)

- Schema Conversion(转换) Tool

  Schema(模式、架构)

- 案例：Oracle→RDS PostgreSQL；SQL Server→Aurora；IBM Db2→Redshift(数据仓库)
- 数据库（小）——小组小型（冰箱、书柜）

  数据仓库（大）——企业大型（图书馆）

48▼ Snow Family (方便部署 &高速 &量大的物理传输家族)

- 做题 关键词：离线环境/网络连接不稳定[物理]、大量数据迁移[量大&高速]

- Snow Family（家族）

  ——Snowcone、Snowball、

  Snowball Edge（单台 80TB→适合迁移数百 TB）、

  Snowmobile（50PB~最高 100PB）

- 简记：

  Snowball=大号移动硬盘 ——巨大量数据(PB 级)、高速传输到云端 的 “移动硬盘”

  Snowcone ——任何地方部署、超便捷 的 数据传输和边缘计算

- 下单方式：

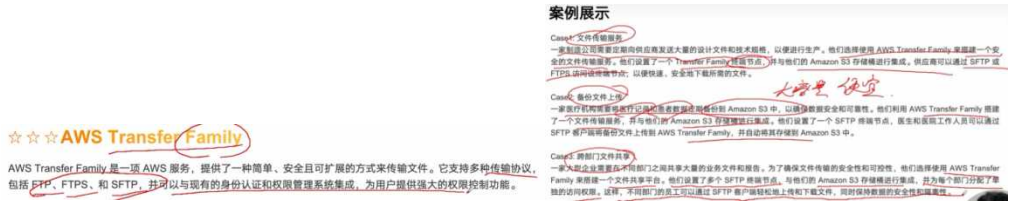
  AWS 下单填写容量(30T/50T/80T)→物理寄到公司→拷贝数据到 Snowball

  →打电话运走 Snowball→AWS 负责把数据拷到云端

- 基础补充：MB→GB→TB→PB （每级 1000 倍关系）

49▼ Transfer Family (文件传输服务，支持多种传输协议：FTP、FTPS、SFTP)

- 复习：大容量、便宜 的存储——选 S3
- 做题关键词：看到 FTP、FTPS、SFTP——选 Transfer Family



●50【迁移和传输 总结】：

AMS(MGN) ——**迁移【服务器】**——**不停机(不断同步、实时增量)**
DMS ——**迁移【数据库】**
SCT ——**换【数据库引擎】**

Migration Hub ——**迁移任务的【集中管理和调度 的 控制台】**
ADS (Application **Discovery** Service) ——系统不熟或太庞大 → 用 ADS 来**【发现并统计】各类应用**

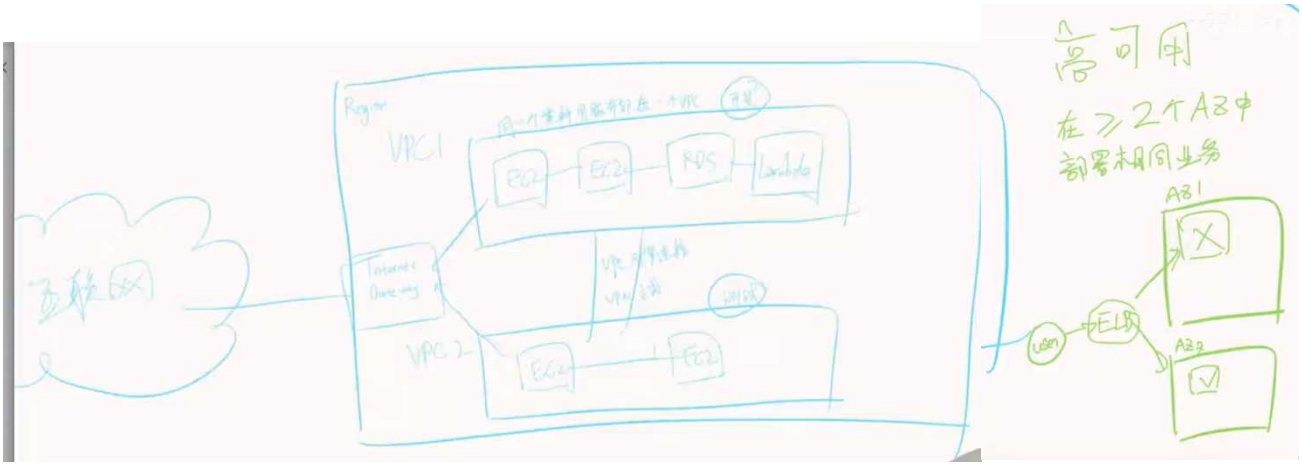
SnowFamily ——**【物理传输】**→离线没网 or 数据量大(PB、TB)
TransferFamily ——**【传输】** →协议：**FTP、SFTP、FTPS**

○【联网与内容分发】

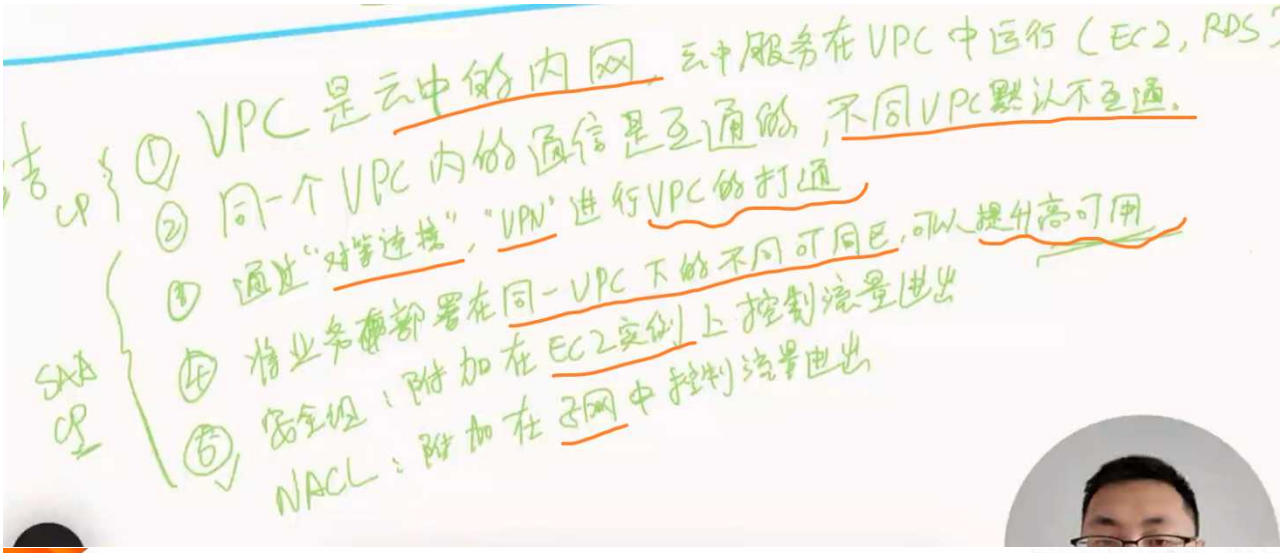
51▼ VPC （=云中 内网）

*IDC = 内网

↓【同一种类服务】(开发、测试...)部署在同一个 VPC 下。 // 通过 Gateway（网关）访问互联网。



总结：



案例展示

云上网络基础

Case1: 企业内部网络扩展

一家企业正在考虑将其内部网络扩展到 AWS 云中，并希望能够在云中运行一些应用程序或服务。他们选择使用 Amazon VPC 来创建一个虚拟网络，并将其扩展到 AWS 云中。通过 Amazon VPC，他们可以创建和管理虚拟网络中的子网、路由表、网络访问控制列表（ACL）等，以实现对应用程序和数据的安全和隔离。

EC2 安全组 网络ACL（NACL）

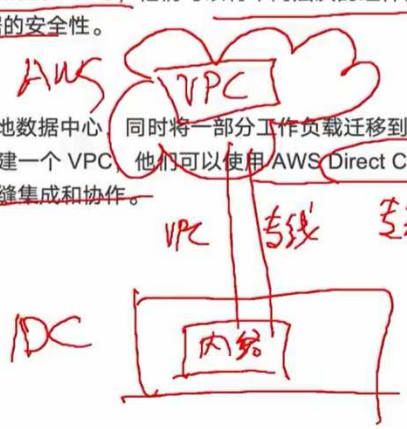
Case2: 多层应用程序部署

一家公司正在构建一个多层次的应用程序，并希望能够在 AWS 云中部署这个应用程序。他们选择使用 Amazon VPC 来创建一个多层次的网络架构，包括公共子网、私有子网和数据库子网。通过 Amazon VPC，他们可以将不同层次的组件部署在不同的子网中，并使用安全组和网络 ACL 来控制流量的访问，以保护应用程序和数据的安全性。

EC2 子网

Case3: 混合云环境连接

一家公司正在实施混合云架构，部署了一些应用程序和服务在本地数据中心，同时将一部分工作负载迁移到 AWS 云中。为了实现安全可靠的连接，他们选择使用 Amazon VPC。通过在 AWS 云中创建一个 VPC，他们可以使用 AWS Direct Connect 或 VPN 连接将本地数据中心连接到 AWS 云中的 VPC，从而实现混合云环境的无缝集成和协作。

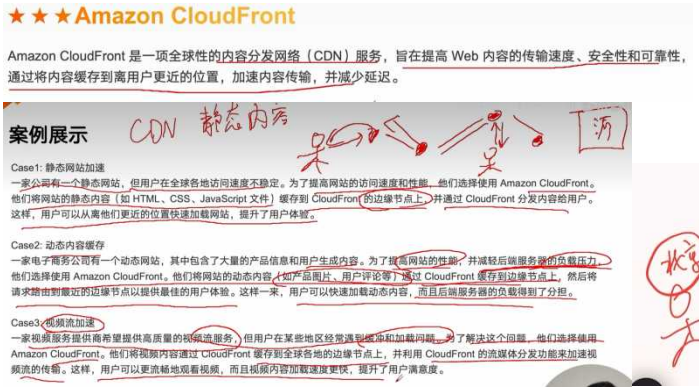


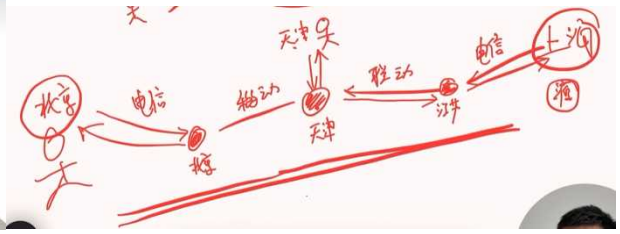
52▼ CloudFront (内容分发=CDN；缓存至更近)

关键词：

内容分发(CDN)。

缓存到用户更近位置——加速传输、减少延迟；提升网站性能、减轻后端服务器压力。





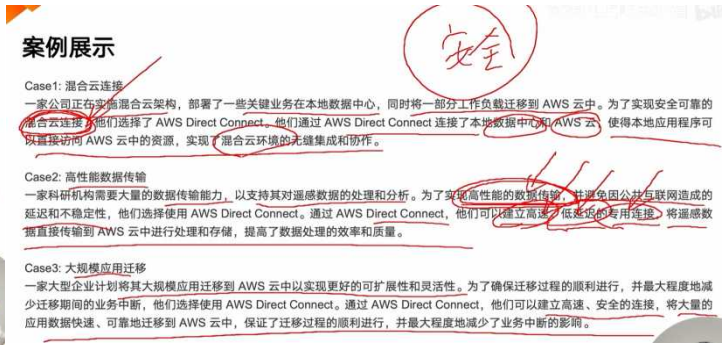
53▼ Direct Connect （混合云；专线-传输快,搭建慢；静态）

关键词：

- **【静态 使用】**
- **(IDC 私网-云)混合云；**
- **专线连接——**

  传输安全、高速低延迟；

  搭建专线慢且贵，因要物理挖土拉线



54▼ Global Accelerator （加速联网；静+动）

关键词：

- **【静+动 使用】；AWS 自家服务**
- **加速服务——路由到最近的 AWS 边缘位置**

55▼ Route 53 （DNS；注册+管理+解析 域名；全球负载均衡）

关键词：域名系统(DNS 解析)——

- **注册+管理 域名**
- **解析域名——域名→IPv4 地址**
- **全球负载均衡——每周导向每周的服务器、中国导向中国服务器→减少访问时长**

56▼ VPN （【IDC-云】加密连接——混合云；公网连接；比专线 便宜）

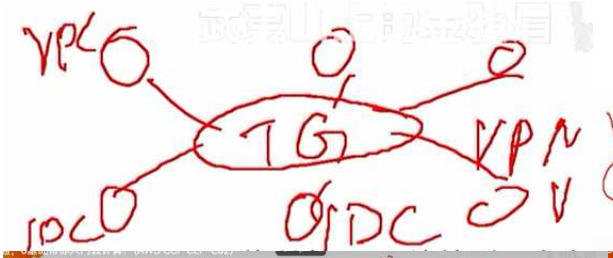
- **IDC-云 加密连接——混合云**
- **公网连接**
- **比专线 便宜**

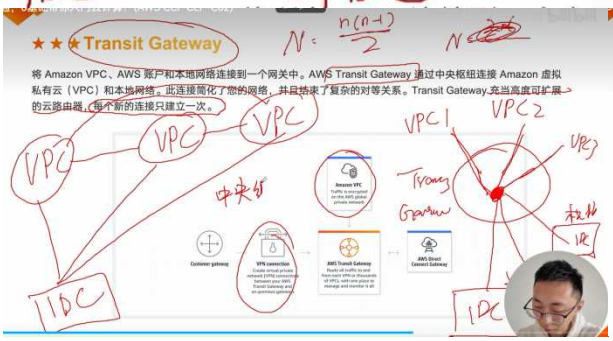
■ VPN(加密通道) & VPC(云上私网) 对比

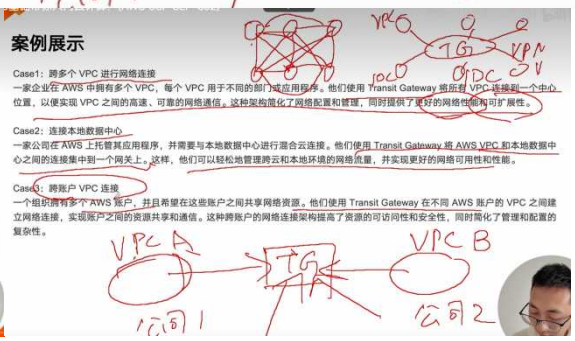
项目	VPN	VPC
全称	Virtual Private **Network**	Virtual Private **Cloud**
本质	**连接方式 / 加密通道**	**网络环境 / 内网** **云上的“私有局域网”**
解决什么	**把两端网络连起来**	在云上建私有网络
典型场景	**远程访问**	AWS 云服务器网络

57▼ Transit Gateway （中央枢纽 网关）

- **中央枢纽 网关 连接 VPC——高度可扩展、每个新连接只要建立一次**
- **案例①：不分云上云下——跨 VPC、VPN、IDC（即跨环境）都可接入传输**
- **案例②：跨 AWS 账户 VPC 连接（即跨公司） 也可**







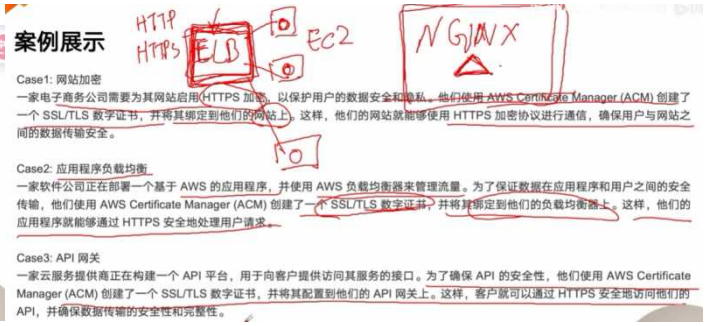
58▼ API Gateway

- **API 是（前后端交互的）接口；API Gateway 是 接口连接。**
- **案例①：统一标准，实现不同微服务之间的通信**

○【安全性身份和合规性】

59▼ ACM（AWS Certificate Manger)

- **购买创建 和 托管管理 HTTPS 证书**（SSL/TLS 证书）的**保管类服务。**



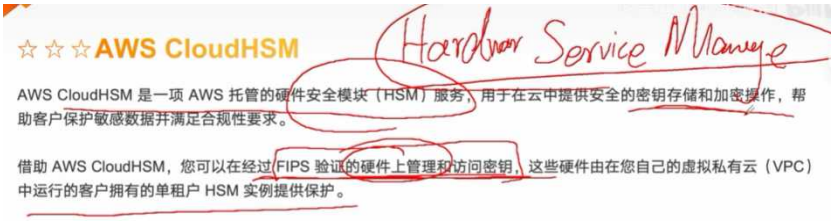
* 补充：HTTP 非加密；HTTPS 加密。

60▼ AWS CloudHSM（硬件安全模块）

- HSM——Hardware Security Module——**硬件安全模块**

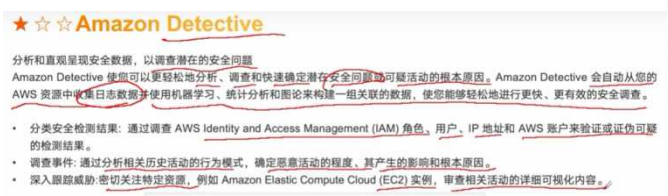
  以高度安全的方式轻松生成和管理加密密钥。它旨在通过确保加密密钥安全存储和管理来保护数据。

- 案例：数据加密；数字签名；合规性要求（安全性、隐私性）



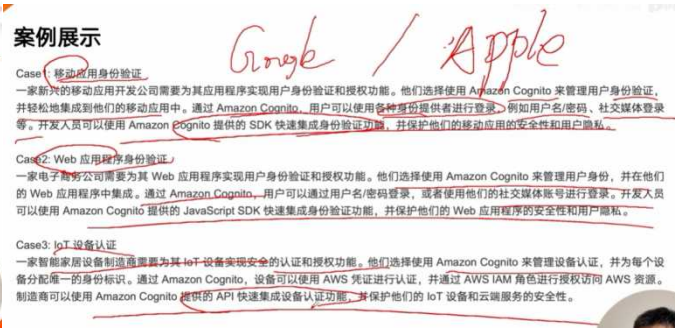
61▼ Amazon Detective（安全分析工具）

- 分析、呈现安全数据，调查潜在安全问题。



62▼ Cognito （账号管理；身份验证与授权：注册管理 + 第三方接入）

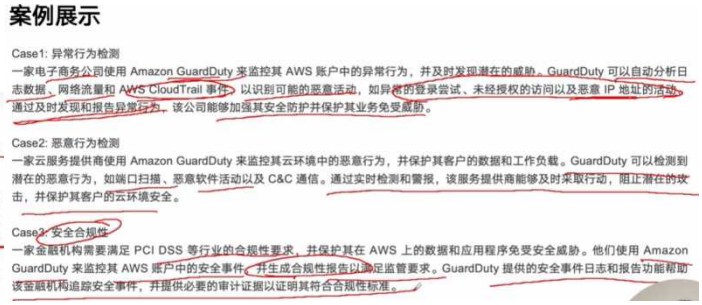
- **注册管理 + 第三方接入**（如案例③的设备认证）
- 案例：移动应用身份验证；Web 应用**身份验证**；IoT **设备认证**



*** 补充：【IoT】=物联网（如：智能家居）**

63▼ GuardDuty （威胁检测：AWS 账户 + 工作负载）

- **威胁检测**服务——监控 AWS 账户 + 工作负载 的恶意活动
- **案例**：异常行为检测；恶意行为检测；安全合规性

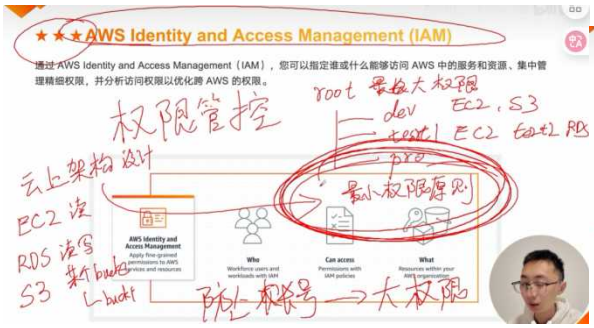


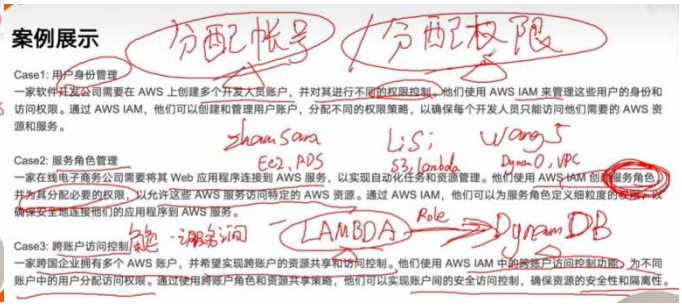
64▼ IAM （权限管控：分配账号+分配权限）(Identity and Access Management)

- **云上架构设计**需要遵循【**最小权限原则**】——防止账号泄露时，大权限被盗。
- **案例**：用户**身份管理**；服务**角色管理**；**跨账户访问**控制

*** 补充：【服务角色】——云服务之间 调用。**

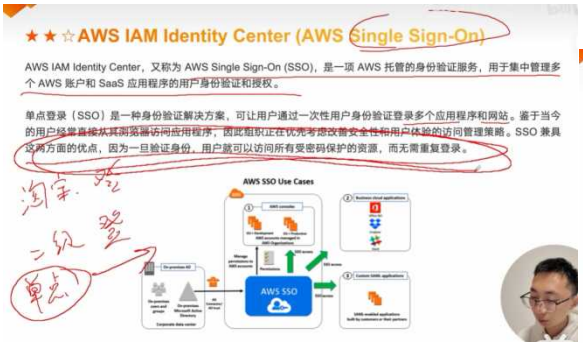
如，若 lambda 需要访问 DynamDB，就要给 lambda 分配一个【角色】，才能对 DynamDB 进行一些操作调用。

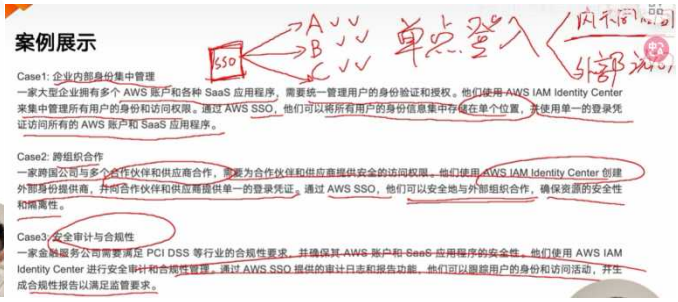




65▼ IAM Identity Center (SSO 单点登录 = AWS Single Sign-On) (用于：内部不同应用+外部访问)

- **SSO 单点登录**——即，**登录一次** 就可以 **访问所有受密码保护的资源**，而**无需重复登录**。
- 用于——内部不同应用+外部访问





66▼ Amazon Inspector(安全漏洞扫描：软件漏洞+网络暴露)

■ Inspector 和 GuardDuty 区别：

Inspector ——**安全漏洞扫描：软件漏洞+网络暴露**

GuardDuty ——**威胁监测：AWS 账户 + 工作负载**

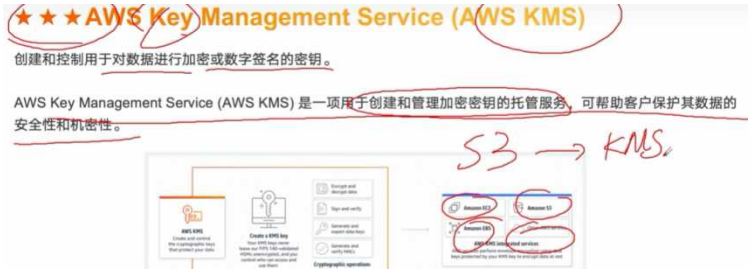
■ KMS 和 HMS 区别：

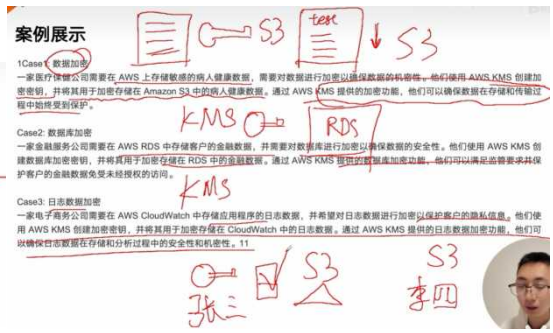
KMS ——密钥 加密

HMS ——硬件 管理

67▼ Amazon KMS(加密 服务)(Key Management Service)

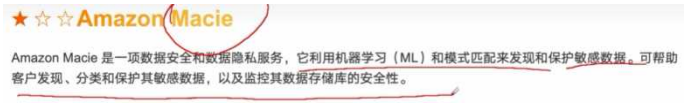
- 创建和管理 **加密密钥** 的托管服务
- 案例：数据·数据库·日志数据 **加密**

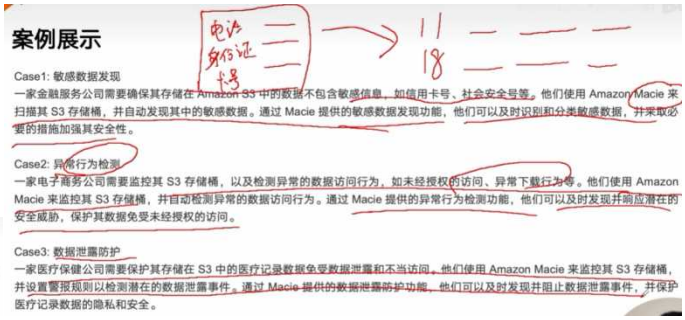




68▼ Macie(发现和保护敏感数据——只发现，不能进行加密，另行用别的加密)

- **保护 数据安全和隐私** 服务
- 案例：敏感数据发现；异常行为检测；数据泄露防护 **(发现敏感数据，防止明文保存，确保数据不泄露)**

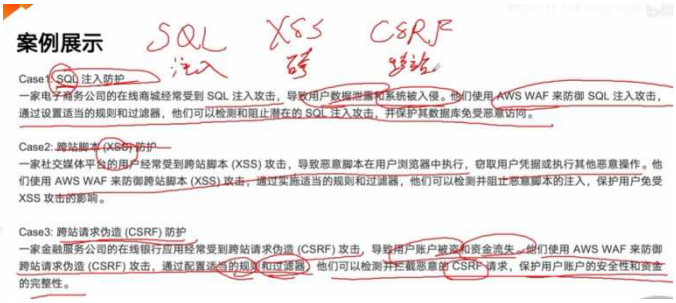




69▼ WAF(防火墙，阻挡对应用的攻击)Web Application Firewall

*** 此 WAF 并非【良好架构框架】！**

- **阻挡对应用的攻击**：如，**SQL 注入**、**跨站脚本攻击(XSS)**、**跨站请求伪造(CSRF)**
- 常见关键词：**防止数据泄露、系统入侵、账户被盗、资金流失**→用 WAF 防御



70▼ Firewall Manager(WAF 防火墙的管理工具：跨账户集中配置+管理防火墙规则)

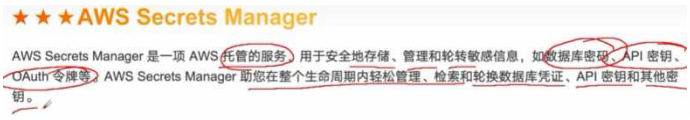
- 跨账户集中配置 和 管理防火墙规则
- 案例：VPC 中的**安全组集中管理**；Web 应用**防火墙的管理**；合规性审计和报告

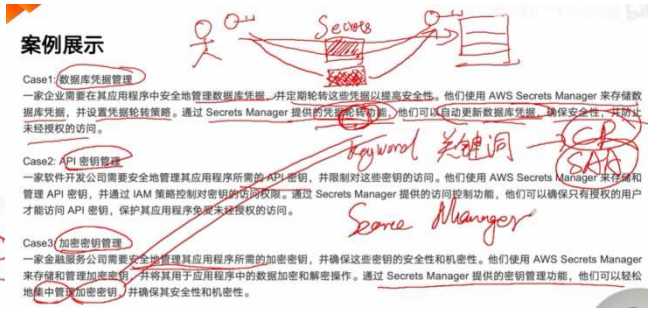
71▼ Shield(安全防护：防 DDoS 攻击+应用保护+API 端点防护)

- DDoS(分布式拒绝服务)保护服务，保护应用免受网络攻击。
  - * **DDoS 攻击**——给容量 100 人的服务器塞 1000 个请求，**拥堵→服务器瘫痪**
    Shield 防御——判断并丢弃异常请求。
- 案例：

72▼ Secrets Manager(托管密钥：存储、管理、轮转敏感信息)

- 存储、管理、**轮转(更改)**敏感信息——如，数据库密码、**API 密钥**、OAuth 令牌
- 案例：数据库凭证管理；API 密钥管理；加密密钥管理





73▼ Secrets Hub (托管的 集中管理和监控账户安全状态——即时洞察威胁和违规)

■Hub——集中式管理工具(枢纽)

- **集中管理和监控 AWS 账户安全状态——提供威胁和违规行为的即时洞察**

74▼ Artifact (提供安全合规性 文档、报告)

75▼ RAM(跨账户资源共享)、AM(审计管理)、DS (企业级目录服务（用户、账号、权限）)

- **IAM 管“云资源权限”； Directory Service 管“人和账号”**
- **Active Directory = 企业里“统一管理账号、电脑和权限”的系统**
  - **打比方来理解：**

    AD = 公司的人事 + 门禁 + 钥匙管理系统

    人事表：谁是员工

    门禁： 能进哪些门

    钥匙： 能用哪些系统

☆☆☆AWS Resource Access Manager (AWS RAM)

AWS RAM 可帮助您的组织或组织单元（OU）内的 AWS 账户之间安全地共享资源，还可针对支持的资源类型与 IAM 角色和 IAM 用户共享，并实现跨账户资源的集中管理和控制。

案例展示

Case1: 跨账户资源共享

一家跨国企业拥有多个 AWS 账户，希望在这些账户之间共享特定的资源，如 Amazon S3 存储桶、Amazon RDS 数据库等。他们使用 AWS RAM 创建资源共享组，并将所需的资源添加到共享组中。通过 AWS RAM 提供的资源共享功能，他们可以轻松地在不同账户之间共享资源，实现资源的集中管理和协作。

Case2: 组织内部资源管理

一家大型企业拥有多个部门和团队，希望在组织内部共享特定的 AWS 资源，以实现资源的最大化利用和优化。他们使用 AWS RAM 创建资源共享组，并将所需的资源分配给不同的部门和团队。通过 AWS RAM 提供的资源共享功能，他们可以在组织内部方便地共享和管理资源，提高资源利用率和效率。

Case3: 合作伙伴资源访问

一家软件开发公司与多个合作伙伴合作开发应用程序，并希望在开发过程中共享特定的 AWS 资源，如 Amazon EC2 实例、Amazon S3 存储桶等。他们使用 AWS RAM 创建资源共享组，并向合作伙伴授予对所需资源的访问权限。通过 AWS RAM 提供的资源共享功能，他们可以安全地与合作伙伴共享资源，实现合作开发项目的顺利进行。

☆☆☆AWS Directory Service

AWS Directory Service for Microsoft Active Directory 又称为 AWS Managed Microsoft AD，可以激活目录感知型工作负载和 AWS 资源，在 AWS 云中创建和管理 Microsoft Active Directory (AD) 或其他目录服务，帮助客户简化目录集成和管理工作，从而让您可以在 AWS 上使用托管的 AD。

案例展示

Case1: 企业用户身份管理
一家企业需要在 AWS 云中实现与其本地 Active Directory（AD）的集成，以便统一管理用户身份和访问控制。他们选择使用 AWS Directory Service 来创建一个托管的 Microsoft AD，然后将其与其本地 AD 进行连接。这样，企业可以通过 AWS Directory Service 管理和同步用户身份，实现跨云和本地环境的统一身份管理。

Case2: Windows 应用程序部署
一家软件开发公司需要在 AWS 云中部署基于 Windows 的应用程序，并需要一个中心化的用户身份验证解决方案。他们选择使用 AWS Directory Service 创建一个托管的 Microsoft AD，并将其作为他们应用程序的身份提供者。通过 AWS Directory Service，用户可以使用他们在 Microsoft AD 中的凭证登录和访问 Windows 应用程序，从而实现方便的身份验证和访问控制。

Case3: AWS 资源访问控制
一家大型企业需要对其在 AWS 云中的资源实施严格的访问控制和权限管理。他们选择使用 AWS Directory Service 创建一个托管的 Microsoft AD，并将其与 AWS Identity and Access Management (IAM) 进行集成。通过 AWS Directory Service，企业可以将其 Microsoft AD 中的用户和组织结构映射到 AWS IAM 中，从而实现对 AWS 资源的精细化访问控制和权限管理。

☆☆☆AWS Audit Manager

持续审计您的 AWS 使用情况，以简化风险与合规性的评估

AWS Audit Manager 是一项 AWS 服务，可帮助企业自动化合规性审计过程，管理合规性工作流程，并生成合规性报告。

案例展示

Case1: 合规性审计
一家金融服务公司需要进行定期的合规性审计，以确保其 AWS 云环境符合监管标准和内部策略。他们使用 AWS Audit Manager 创建了合规性框架，并配置了一系列合规性规则和要求。AWS Audit Manager 自动化执行审计程序，收集审计结果，并生成合规性报告，帮助他们快速识别和解决潜在的合规性问题。

Case2: 安全性审计
一家电子商务公司需要对其 AWS 资源进行安全性审计，以确保其云环境的安全性和完整性。他们使用 AWS Audit Manager 创建了安全性审计框架，并配置了一系列安全性规则和要求。AWS Audit Manager 自动化执行审计程序，收集安全性评估数据，并生成安全性审计报告，帮助他们评估其云环境的安全性，并采取必要的措施加强安全防护。

Case3: 合作伙伴合规性
一家软件公司计划将其产品部署到 AWS Marketplace 上，并需要通过 AWS 合作伙伴认证。为了满足认证要求，他们需要进行合规性审计，并生成合规性报告。他们使用 AWS Audit Manager 创建了合规性框架，并根据 AWS 的合规性标准配置了一系列合规性规则。AWS Audit Manager 自动化执行审计程序，收集合规性数据，并生成合规性审计报告，帮助他们证明其产品符合 AWS 的安全和合规性要求，顺利完成合作伙伴认证流程。

作成者：朱暁駿 | 作成期间：2026 年 1 月