

用 AWS 搭個高可用性網站

Allen



目錄

1. 在了解高可用之前..., 先來了解一下這之中會用到的各種 AWS 服務
2. 該怎麼做到高可用呢？
3. 來段 HA 小 Demo 吧！

在了解高可用之前...

先來了解一下這之中會用到的各種 AWS 服務

- 地區與區域
- Elastic Compute Cloud (EC2)
- 網路相關服務
- Elastic Load Balancer (ELB)
- Auto Scaling Group (ASG)
- Relational Database Service (RDS)

地區 (Region)

- AWS 在世界各地都有資料中心，分別在世界各地提供各種雲端服務
- AWS 的服務有分全球服務 (Global) 與地區服務
- 使用服務前，需要先選擇地區
- 地區都有代號，例如美國 Oregon 是 us-west-2
- 地區的選擇可能有許多考量，例如：
 - 如果想降低客戶訪問服務的延遲，應該選擇離客戶最近的地區
 - 該地區是否有提供你想要用的服務
 - 成本考量。相同服務在不同地區，會有不同的價格
 - 合規性。抱歉，我們的資料不能離開國內

AWS 有沒有台灣地區？

沒有

只有 Local Zone, 只提供很少一部分的服務

Google Cloud Platform 有提供台灣地區

嗨！



Google Cloud

服務種類

全球服務

地區服務



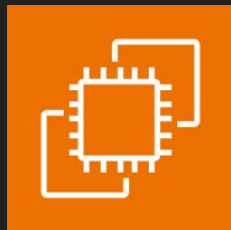
Route53



IAM



Cloudfront



EC2



RDS

(幾乎都是)

■ ■ ■

看起來是全球，但跨地區要錢的服務



S3

可用區域 (Availability Zones, AZ)

- 每個地區 (Region) 都有多個 AZ, 例如美國 Oregon (us-west-2) 就有四個
 - us-west-2a
 - us-west-2b
 - us-west-2c
 - us-west-2d
- 每個 AZ 都是獨立一座資料中心, 彼此以高速網路互相連接

Elastic Compute Cloud (EC2)

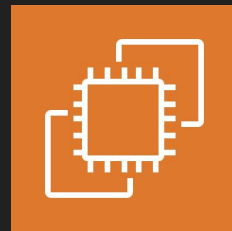
- AWS 的虛擬機器 (VM) 服務, 你可以將你的網站架在上面
- 機器有許多種規格可以選擇

t4g.small

應用場景

版本號碼

規格大小



Amazon Machine Image (AMI)

- AMI 是 EC2 的啟動映像檔，主要用途是將作業系統安裝在 EC2 上
- 有多種作業系統可以選擇 (Linux、MacOS 與 Windows)
- AMI 無法跨地區共享，如果想要跨區域，可以使用快照複製一份到其他區域
- 你也可以製作自己的 AMI，將自己的服務包裝進去，讓機器一啟動就是服務啟動的狀態
 - 可以使用 Hashicorp 的 Packer 來製作 AMI



Placement Group

指定一群 EC2 啟動位置的策略, 有三種

- 叢集 (Cluster): 將 EC2 放在同一個 AZ 內, 減少網路延遲
- 分區 (Partition): 將 EC2 分散到邏輯分區, 不同的 EC2 不會共用相同的底層硬體。適合大量分散和複寫的工作負載 (Hadoop 與 Kafka)
- 分散 (Spread): 嚴格的將 EC2 分散到不同的底層硬體

省錢小撇步, Reserved Instance、Saving Plan 與 Spot

- Reserved Instance: 預先購買特定機器規格運行一年或三年的費用, 最多可以擁有 72% 的折扣。缺點是彈性較差, 中途無法隨意更換規格
- Saving Plan: 類似點數制, 預先購買一定的點數, 後續使用機器會消耗該點數。折扣雖然沒有 RI 多, 但是較為彈性, 中途可以更換機器規格
- Spot: 以競標的方式來租用機器, 可以獲得大量折扣, 但只要競標價格低於別人的出價, 機器就會被收回。用機車的錢, 買一台勞斯萊斯, 只是這台勞斯萊斯隨時有機會被收走

Solution Architect (SA) 有話要說

RI 相較於 SP, 擁有開啟機器的優先權

假設 AWS 因為出事導致伺服器資源不足, 會優先將資源給 RI 使用, 因此使用 SP 是有機會無法啟動的！

我：完全沒聽上課的老師說過

SA：這就是老師與老師傅的差距了

果然從業久了, 什麼鬼都會遇到



User Data

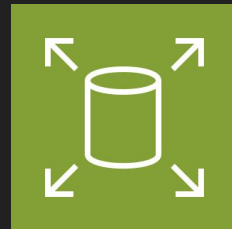
- 你可以寫好一個 Shell Script, 讓機器在啟動後立刻執行這個 Shell Script
- 可以用在機器的初始化, 例如更新或是安裝軟體
- 指令預設使用 root 權限
- 執行的結果可以在 `/var/log/cloud-init-output.log` 中查看 (Linux)

Metadata

- 用來取得 EC2 的資料, 例如 IP
- 因為 AWS 環境高度虛擬化, 連網卡都是虛擬的, 因此使用 `ifconfig` 指令是無法得知 Public IP 的資訊
- 在 EC2 上可以訪問這個網址取得個體資訊:
 - 取得 Private IP : <http://169.254.169.254/latest/meta-data/local-ipv4>
 - 取得 Public IP : <http://169.254.169.254/latest/meta-data/public-ipv4>
- 你也可以自己設定額外的 Metadata

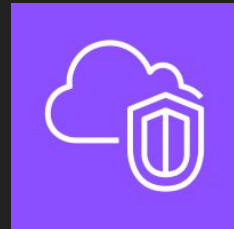
Elastic Block Storage (EBS)

- 網路硬碟，可拆卸式。可以刪除 EC2 而不刪除 EBS，保留資料
- 與 EBS 相對的是 Instance Store，雖然 I/O 比 EBS 快，但是機器刪除，資料也會跟著一起刪除



Virtual Private Cloud (VPC)

- 虛擬網路，底下可以放置你的 AWS 資源 (例如 EC2 與 RDS)
- 無料，帳號底下的每個地區預設會有一個 VPC (有開啟的話)
- 只能使用私有 IP
 - 10.0.0.0 – 10.255.255.255 (10.0.0.0/8)
 - 172.16.0.0 – 172.31.255.255 (172.16.0.0/12)
 - 192.168.0.0 – 192.168.255.255 (192.168.0.0/16)
- 可以設定的 IP 範圍為 /16 至 /28
- 底下可以切出子網路 (Subnet)



什麼是 CIDR (無類別域間路由)？

全稱是 Classless Inter-Domain Routing, 以剛剛的 VPC CIDR 為例

10.0.0.0/16

00001010.00000000.00000000.00000000

這16 個位數是固定的

這邊可以切出 Subnet 與 IP

Subnet (子網路)

- 我很常用 10.0.0.0/24, 提供的 IP 範圍是 10.0.0.0 ~ 10.0.0.255
- 每個子網路有 5 個 IP 被 AWS 預定, 無法使用 (前面 4 個與倒數最後 1 個)
 - 10.0.0.0: 網路位址 (Network Address)
 - 10.0.0.1: VPC Router
 - 10.0.0.2: DNS Server
 - 10.0.0.3: AWS 保留, 也許未來會用
 - 10.0.0.255: 廣播位址 (Broadcast Address), 只是保留, AWS 並不支援廣播

來看看 Subnet 的 CIDR

以我很常使用的 Subnet CIDR /24 為例

10.0.0.0/24

00001010.00000000.00000000.00000000

這16 個位數是固定的

24 個位數, 可以有 $2^{(24 - 16)} = 256$ 個網段

每個網段有 256 個 IP

再來看一個 CIDR 範例

假設 VPC 是 10.0.0.0/18, Subnet 切 /25

- 可以有 $2^{(25-18)}$ 個網段, 每一個網段可以有 $2^{(32 - 25)}$ 個 IP

10.0.0.0/25

00001010.00000000.00000000.00000000

這 18 個位數是固定的

25 個位數, 可以有 $2^{(25 - 18)} = 128$ 個網段

每個網段有 128 個 IP

提問！

Q: 如果你需要 29 個 IP 給機器使用, Subnet 的最小 CIDR 是？

A: 答案是 /26。/27 雖然能提供 $2^{(32 - 27)} = 32$ 個 IP, 但其中 5 個不能使用, 所以能用的 IP 是 $32 - 5 = 27$ 個

SA 有話要說

VPC 切出 /16 是非常大的, 應該根據 IP 的需求來切 VPC 與 Subnet 的 CIDR

而切得太剛好可能不利於未來擴充, 因此適時保持一些彈性切多點 IP 也可以

SA: 切 Subnet 是門學問



Internet Gateway (IGW)

- 讓 VPC 底下的資源 (例如 EC2) 能上網, 還有能被外部訪問
- Subnet 有分 Public Subnet 與 Private Subnet, 有連上 IGW 的 Subnet 就是 Public Subnet
- 一個 VPC 只能有一個 IGW

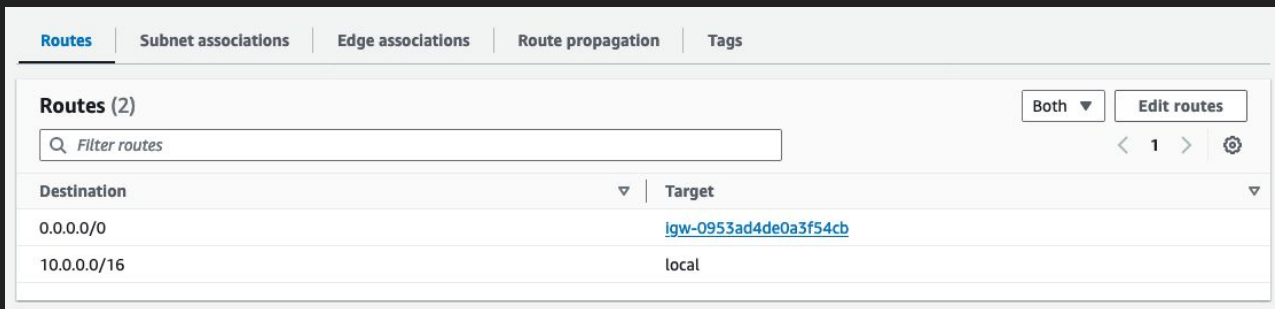
IGW 建立之後記得編輯 Route Table
讓 Subnet 連上 IGW

不然無法上網



Route Table

- 一個 Subnet 會有一個對應的 Route Table
- 預設的 Route Table 可以讓 VPC 底下所有 Subnet 互相連接



The screenshot shows the AWS Management Console interface for a Route Table. The top navigation bar includes tabs for 'Routes', 'Subnet associations', 'Edge associations', 'Route propagation', and 'Tags'. The 'Routes' tab is selected. Below the tabs, the title 'Routes (2)' is displayed next to a 'Both' dropdown and an 'Edit routes' button. A search bar labeled 'Filter routes' is present. The main content area contains a table with two columns: 'Destination' and 'Target'. The first row shows the destination '0.0.0.0/0' and the target 'igw-0953ad4de0a3f54cb'. The second row shows the destination '10.0.0.0/16' and the target 'local'.

Destination	Target
0.0.0.0/0	igw-0953ad4de0a3f54cb
10.0.0.0/16	local

有 IGW 的 Route Table 範例

172.16.0.0

172.16.1.0

172.16.2.0

NAT Gateway

- NAT (Network Address Translate), 意思是網路位址轉譯, 原本目的在於解決 IP 不夠用的問題
- 需要 IGW 上網
- 讓 VPC 底下的資源 (例如 EC2) 能上網, 但不能被外部訪問
- NAT Gateway 會放在 Public Subnet 中, Private Subnet 會使用 NAT Gateway 上網
- 非常貴, 每月保底 30 美金 (每小時 0.045 美金), 還有額外的流量費用



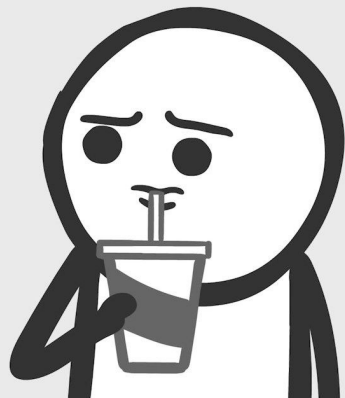
能不能別用 NAT Gateway？它好貴

想省錢可以自己使用 IPTables 來搭建一個 NAT Instance，但要自己處理 Scaling

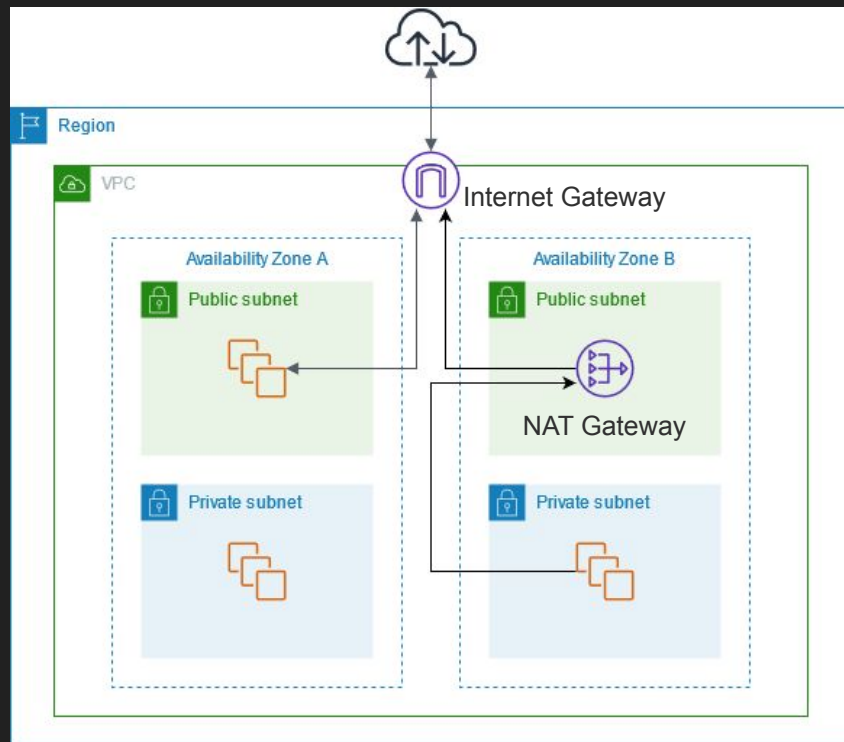
NAT Gateway 可以負擔很大的流量，為 AWS 的黑科技之一

我：能不能別用 NAT Gateway？

SA 的表情 →



常見的 Public/Private Subnet 架構圖



剛剛都說 IPv4, 來說說 IPv6

- VPC 與 Subnet 也可以使用 IPv6, 原本 IPv4 預設是一定會有的, 無法單純只用 IPv6, 但後來 AWS 支援 IPv6-Only
- 注意 IPv6 是沒有私有位址的, 全部都是公有位址

你可能想問: 全部公有是不是很危險?

資料庫也暴露在外

IPv6 的奇特功能 Egress-only Gateway

- 功能類似於 NAT Gateway, 讓 VPC 底下的資源可以上網, 但是外部無法直接訪問
- Egress-only Gateway 只支援 IPv6, 是無料的 !

我之前本來想用這個來省錢
結果發現 GitHub 不支援 IPv6

只好繼續使用 NAT

Security Group

- 剛剛說的都設定好了，但機器還是無法上網？你可能忘記設定 Security Group
- Security Group 的功能類似防火牆，可以設定白名單允許流量出入
 - Inbound: 從外面進來 AWS 的流量
 - Outbound: 從 AWS 出去外面的流量，預設全開

Inbound rules (1)								
<div><input type="text" value="Search"/></div> <div><div></div><div>Manage tags</div><div>Edit inbound rules</div></div> <div>< 1 > ⚙</div>								
<input type="checkbox"/>	Name ▾	Security group rule ID ▾	IP version ▾	Type ▾	Protocol ▾	Port range ▾	Source	
<input type="checkbox"/>	-	sgr-02e73d90664e798a4	-	All traffic	All	All	sg-07fdb3ca7aba7e394 / default	

Outbound rules (1)								
<div><input type="text" value="Search"/></div> <div><div></div><div>Manage tags</div><div>Edit outbound rules</div></div> <div>< 1 > ⚙</div>								
<input type="checkbox"/>	Name ▾	Security group rule ID ▾	IP version ▾	Type ▾	Protocol ▾	Port range ▾	Destination	
<input type="checkbox"/>	-	sgr-02dbd6066c007bc32	IPv4	All traffic	All	All	0.0.0.0/0	

Security Group 的特性

- Security Group 只有允許 (Allow) 規則, 沒有阻擋 (Deny) 規則
- 在決定是否允許流量之前會評估所有規則
- Security Group 是有狀態的 (Stateful), 能進來就能出去, 能出去就能進來
 - 因此修改 Inbound 的規則已足夠, Outbound 預設的全開規則不太會更動
- Security Group 可以使用在 EC2、ENI 與 RDS ... 等, 當防火牆幫忙過濾流量

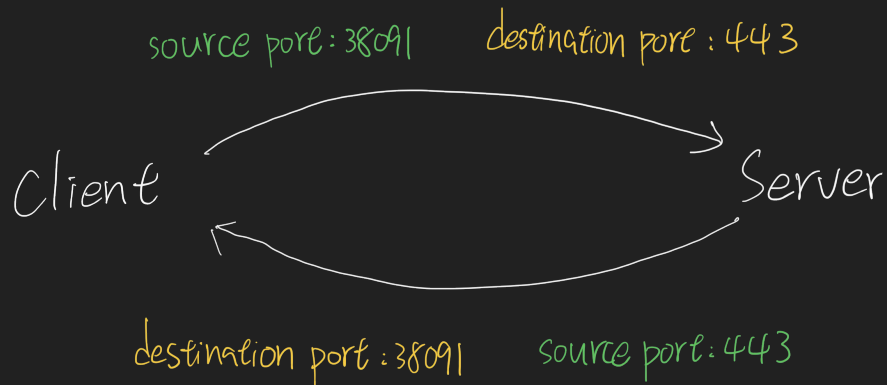
Network Access Control List (NACL)

- 與 Security Group 功能類似，也是用來過濾流量
- 但是 NACL 是無狀態 (Stateless) 的，出去與回來的流量都要設定規則。否則出去的回不來，能進來的出不去
- 規則有優先順序，使用數字來決定優先級。數字越小，優先級越高
- NACL 是使用在 Subnet 上



Ephemeral Ports (臨時通訊埠)

- 因為 NACL 是無狀態的, 因此需要考慮客戶端會使用的臨時通訊埠
- 如果我想開啟一個 HTTP 端點, NACL 除了在 Inbound 與 Outbound 設定允許 80 port, 我还需要設定臨時通訊埠
- 當客戶端連到我的 HTTP 端點時, 客戶端會使用一個臨時通訊埠來接收流量
- Linux kernels 的臨時通訊埠為 32768-61000



Control traffic to subnets using network ACLs

Security Group VS NACL

	Security Group	NACL
狀態性	有狀態	無狀態
使用對象	使用在 EC2 與 RDS 上	使用在 Subnet 上
流量規則設定	只能允許	可以允許與阻擋
規則判斷	會評估所有規則來判斷	使用數字決定優先級

我有話要說

真的很少用到 NACL, 大多場景用 SG 已足夠。但因為兩者功能類似, 在 SAA 考試中很常拿來比較。

Elastic Load Balancer (ELB)

- Load Balancer 可以將進來的流量分散至不同的機器上，減少單一機器的負擔
- 可以設定 Health Check, 如果目標機器不健康，會停止將流量轉過去



ELB 的種類

- Application Load Balancer (ALB): 屬於 Layer 7, 支援 HTTP、HTTPS 與 WebSocket。不支援固定 IP (注意 gRPC 也支援, 因為 gRPC 是跑在 HTTP/2 上)
- Network Load Balancer (NLB): 屬於 Layer 4, 支援 UDP 與 TCP, 可以承受較大的流量 (百萬/秒)。支援固定 IP
- Gateway Load Balancer: 可以將流量轉送至第三方軟體, 例如防火牆與威脅偵測。使用 GENEVE 協議

Application Load Balancer (ALB)

- ALB 預設是跨區域 (AZ) 的
- ALB 並不支援固定 IP, 建立後會有一個 Domain Name, 如果想改用自己的 Domain Name, 可以使用 CNAME
- 可以根據 Hostname、URL Path 或是 Query String, 將流量轉送到不一樣的目標
- 想知道客戶端的真實 IP, 可以查看 Header 中的 X-Forwarded-For

ALB 的 Target Group

ALB 轉發流量的目標, 稱為 Target Group, Target Group 可以是

- EC2
- Auto Scaling Group (一群 EC2)
- ECS
- Lambda

SA 有話要說

ALB 背後其實也是機器，如果你預計會有很大的流量進來，可以跟 AWS Support 事先說好加開機器，以免到時候 ALB 掛掉

我：AWS 的服務還真是周到

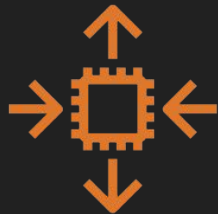
SA: Support 要花錢的

我的表情 →



Auto Scaling Group (ASG)

- 可以使用 Cloudwatch 來監測機器的指標，例如 CPU 使用率
- 可以設定觸發 Scale In/Out 的政策 (Scaling Policy)，例如 CPU 使用率達到 80% 時，增加一台機器
- 你可以設定啟動機器的模板 (Launch Template)，讓機器在一啟動後就是準備好的狀態



Scaling Policy 的種類

- Simple/Step Policy: 當機器 CPU 使用率達到 70%, 增加一台機器; 當機器 CPU 使用率低於 30%, 減少一台機器
- Target Tracking Policy: 當一群機器的平均 CPU 使用率達到 70%, 持續加碼, 直到平均 CPU 使用率未達 70%
- Schedule Policy: 已經知道何時會有大流量, 在大流量時段前增加機器來處理
- Predictive Scaling: 讓 AWS 根據過往的流量紀錄, 智慧的幫你調整機器數量

Launch Template

- 讓機器一啟動就是服務運行的狀態，減少準備時間並加快 Scale Out 的速度
- Launch Template 其中包含許多資訊，例如：
 - AMI + 機器類型
 - User Data
 - EBS
 - Security Groups
 - SSH Key Pair

Relational Database Service (RDS)

- 全託管的 RMDBS 服務, 有常見的 MySQL、MariaDB 與 PostgreSQL
- 還有 AWS 經由 MySQL 與 PostgreSQL 魔改後的 Aurora
 - Aurora 效能為 MySQL 的數倍, 但是沒有開源



IAM DB Authentication

- 不使用密碼, 改為使用 Identity and Access Management (IAM) 取得短暫的 Token 來進行連線
- 是官方建議的做法

Multi-AZ Deployment

- RDS 可以設定 Mutli-AZ Deployment, AWS 會在不同的區域部署多台 RDS Instance
- RDS Instance 會分為 Primary 與 Standby
- Standby 不開放讀寫
- 同步 (Synchronize) 備份 Primary 的資料
- 如果 Primary 掛掉, AWS 會修改 CNAME, 將流量導向至 Standby

Read Replica

- 如果想分散 Primary 資料庫的讀取負擔, 可以開啟 Read Replica
- Read Replica 只開放讀, 不開放寫
- 會異步 (Asynchronous) 備份 Primary 的資料

介紹了這麼多服務，該怎麼做到高可用呢？

該怎麼做到高可用？

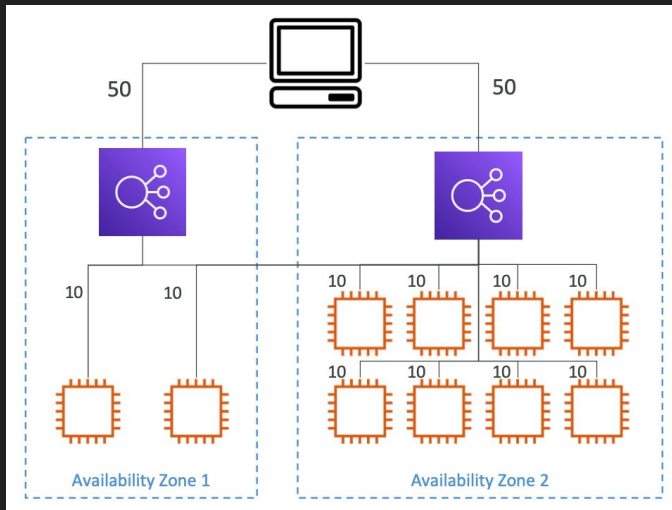
- 當流量越來越大，服務仍能正常運行
- 當某個區域失效時，服務仍能正常運行

當流量越來越大，服務仍能正常運行

- 應該有機制監測機器的各項指標，如 CPU 使用率與記憶體使用率
- 當流量提高，導致機器的負載加重時，我們可以利用 Scale Out 來處理提高的流量，降低個別機器的負載
- Load Balancer 會將流量轉送至新啟動的機器，降低原本機器的負載

當某個區域失效時，服務仍能正常運行

- 即使單一區域因為天災人禍導致失效，其他區域可以立即補上
- Load Balancer 會停止將流量轉送至失效的區域



當某區域失效，LB 可以把流量轉送至其他區域

SAA 考題

有一間公司正在運行一個購物網站，該網站需要 6 台機器。目前公司有 3 個 AZ，分別是 eu-east-2a、eu-east-2b 與 eu-east-2c。

請問下列哪一種部署方式可以讓服務擁有容錯性，且耗費的成本最少？

- A. 2 台在 eu-east-2a, 2 台在 eu-east-2b, 2 台在 eu-east-2c
- B. 3 台在 eu-east-2a, 3 台在 eu-east-2b, 3 台在 eu-east-2c
- C. 2 台在 eu-east-2a, 4 台在 eu-east-2b, 2 台在 eu-east-2c
- D. 6 台在 eu-east-2a, 6 台在 eu-east-2b, 0 台在 eu-east-2c

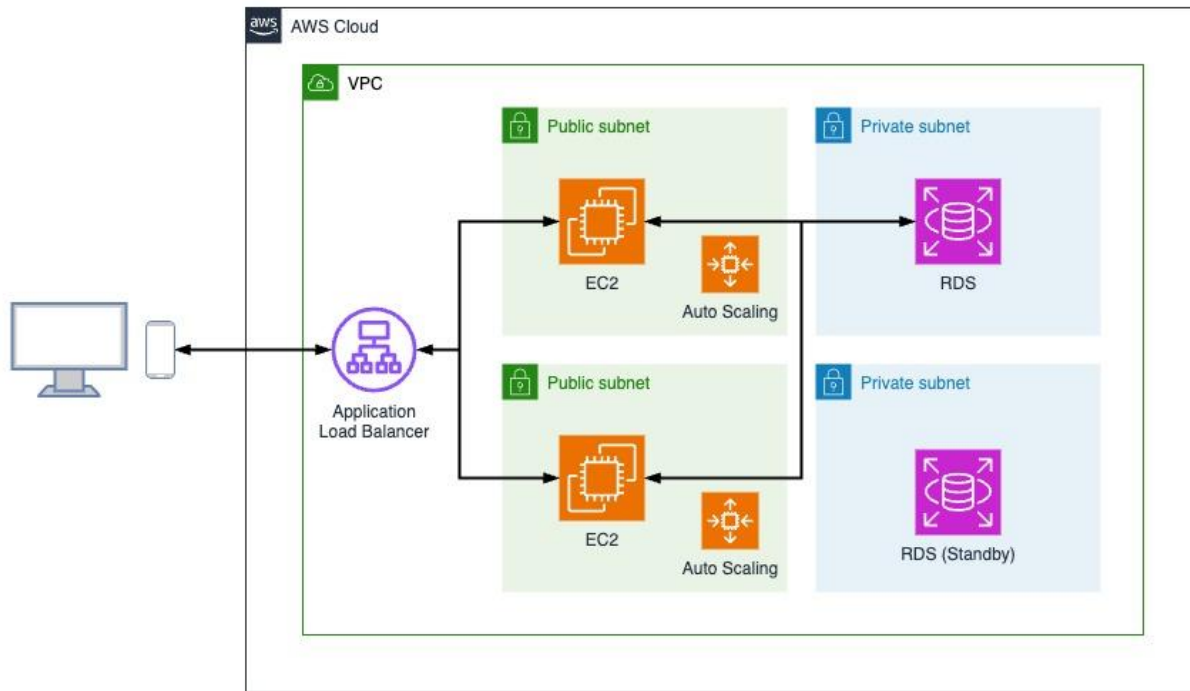
題目背後的意思，是指如果某個AZ 掛掉，剩下的機器數量能否保持在6 台以上。

只有 B 與 D 符合這個條件。而B 的總機器數量少於D 的總機器數量($9 < 12$)，所以成本更低，故選B。



來段 HA 小 Demo 吧！

HA 簡易架構



SA 有話要說

有沒有遇過 AZ 間彼此無法互通的情況？

這時候 Standby 就會以為 Primary 掛掉，然後 Standby 就會莫名其妙的起來了

要做到高可用並不容易啊 ...

我：真的是從業久了，什麼鬼都會遇到

* 這種情形俗稱 Split-brain 問題

參考資料

- [AWS SOA 學習筆記- High Availability & Scalability | 小信豬的原始部落](#)
- [Why we need to configure ephemeral ports on NACL for web servers?](#)
- [Amazon VPC 常見問答集](#)
- [Udemy - Ultimate AWS Certified Solutions Architect Associate SAA-C03](#)