

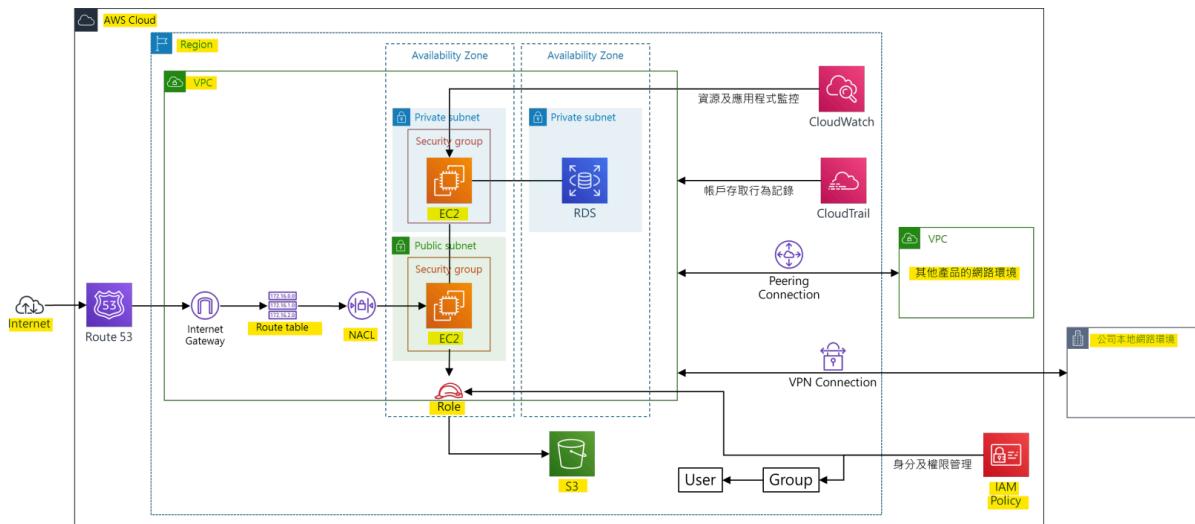


AWS

Tags	AWS
Status	Complete
Status 1	Not started

AWS 架構圖

▼ 整體架構圖



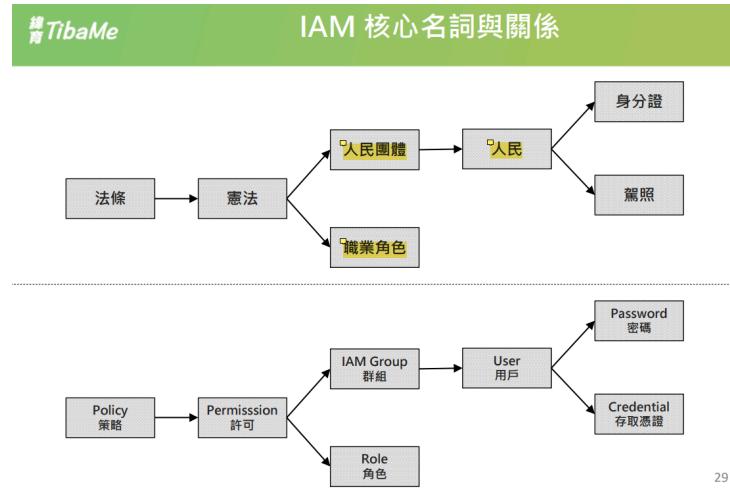
▼ 應用與協定架構圖

IAM

▼ 【IAM 介紹】Identity and Access Management 權限管理

1. 安全地控制使用者對 AWS 服務和資源的存取權限
2. 使用 IAM 建立和管理 AWS 使用者和群組

3. 可用來允許和拒絕他們存取 AWS 資源(存取審核)



29

可以針對特殊人員進行權限條款設定(AWS提供接近完全客製規範條款)

也可以針對一般使用者的通用條款，但就會需要密碼以及存取憑證(如 Google Authenticator)等驗證身分資訊

▼ 【Policy】建立與說明

Policy 是一個 JSON 格式所組成的架構

可以指定 AWS 資源的許可權，以及控制哪些 AWS 使用者和角色可以訪問這些資源。

▼ Policy 架構主要元素

1. Version : Policy 版本號

如：目前 IAM Policy 版本號為 "2012-10-17"。

2. Statement : 語句，也是最重要的元素。

一個 Policy 可以有多個 Statement，每個 Statement 定義了一個授權或拒絕訪問 AWS 資源的規則。每個 Statement 又包含以下幾個元素：

- Effect : 設定為 "Allow" 或 "Deny"，表示該規則是允許還是拒絕對資源的訪問。
- Action : 表示可以對資源進行的操作，例如 "s3:GetObject" 表示可以讀取 S3 存儲桶中的物件。
- Resource : 表示可以進行操作的 AWS 資源，
例如："arn:aws:s3:::examplebucket/*" 表示可以對名為 examplebucket 的 S3 存儲桶中所有物件進行操作。

- **Condition**：可選項目，表示**更精細的控制條件**，例如限制 IP 地址範圍等。

Policy 的組成

在探討Policy語法前，我們先來看人類世界如何描述權限的概念，再把它抽象化
Policy 的結構，也是這樣的概念



Effect、Principal、Conditional、Action、Resource (EPCAR)

Policy範例如下圖

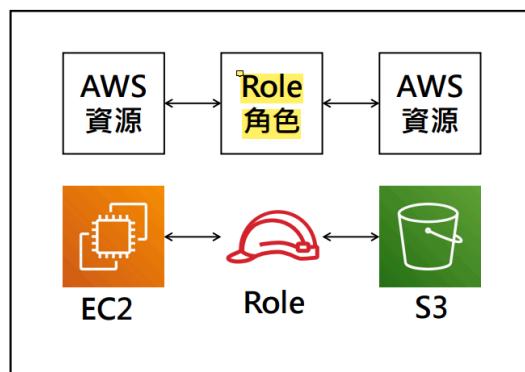
```
{
  "Version": "2012-10-17",
  "Id": "http referer policy example",
  "Statement": [
    {
      "Sid": "Allow get requests originating from www.example.com and example.com.",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::examplebucket/*",
      "Condition": {
        "StringLike": {"aws:Referer": ["http://www.example.com/*", "http://example.com/*"]}
      }
    }
  ]
}
```

Policy可分：

1. 官方預設 -> 權限較寬
2. 自定義 -> 細粒度設定權限

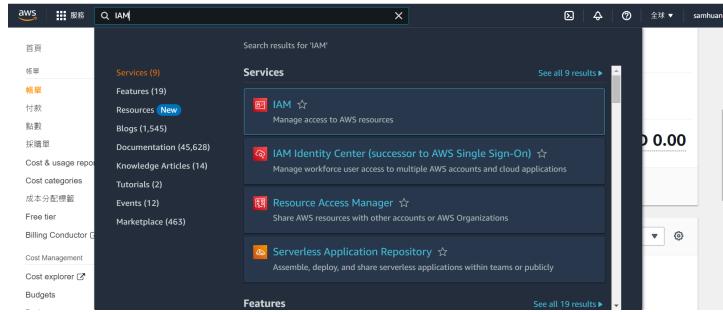
30

▼ 【Role】決定資源之間存取的權限



▼ 【建立支出預算】ROOT 神 > 創造 > IAM user 神之子(如 Linux 的 sudo)

1. 進入「帳戶」頁面，建立支出規劃



2. 帳戶資訊：之後可以使用這兩個名稱做登入

AWS 帳戶

帳戶 ID
268246653390

新增 MFA

帳戶別名
268246653390 建立

此帳戶中 IAM 使用者的登入 URL
<https://268246653390.signin.aws.amazon.com/console>

快速連結

[我的安全身份碼](#)
管理您的存取金鑰、多重要素驗證 (MFA)

3. 因為是使用 root 進入，因此會有安全警告

IAM 儀表板

安全建議

為根使用者新增 MFA
為根使用者新增 MFA - 為根使用者新增多重要素驗證 (MFA)，以提升此帳戶的安全性。

根使用者沒有作用中的存取金鑰
使用與 IAM 使用者連接的存取金鑰 (而不是根使用者的存取金鑰) 有助於提升安全性。

IAM 資源

4. 設定使用 Google Authenticator 進行驗證登入，提升安全性

設定完成後，以後就需要帶著手機，才可以登入

多重要素驗證 (MFA) [1]		
使用 MFA 提高 AWS 帳戶的安全性。使用 MFA 登入需要兩步 MFA 驗證。每個使用者最多可擁有 8 個 MFA 裝置。 進一步了解		
裝置類型	識別符	建立日期
<input type="radio"/> 虛擬	arn:aws:iam::268246653390:mfa/wrhuang	現在

移除 | 重新同步 | **啟用 MFA 裝置**

5. 建立帳號預算，類似 Google Ads 一樣，可以選擇費用計算方式，可以避免超支



6. 設定預算名稱，藉此管理專案預算 (重要)可設定帳單預算以及預警通知人，避免金額太多沒有發現

預算名稱
提供此預算的描述性名稱。

名稱必須介於 1-100 個字元之間。

輸入預算金額 (\$)
上個月的成本： -

電子郵件收件人
指定超過閾值時要通知的電子郵件收件人。

7. 預算設定完成

預算 (1) <small>資訊</small>						
<input type="text"/> 尋找預算		<input type="button" value="顯示所有預算"/>		<input type="button" value="下載 CSV"/> 動作 <small>▼</small> <input type="button" value="建立預算"/>		
<input type="checkbox"/>	名稱	▲ 關值	▼ 預算	已使用金額	預測金額	目前支出對照預算
<input type="checkbox"/>	CHI101_31_Budget	<input checked="" type="radio"/> 確定	US\$3.00	US\$0.00	-	0.00%

- ▼ 【IAM 實作 - 建立管理權限】ROOT 神 > 創造 > IAM user 神之子(如 Linux 的 sudo)
建立方法有二：

▼ 一、建立 IAM Group，針對一般使用者進行權限設定

整體操作流程如下：

建立使用者群組 > 建立群組規則 > 群組內建立角色

1. 建立使用者群組

IAM > 使用者群組 > 建立使用者群組

建立使用者群組

為群組命名

使用者群組名稱
輸入有意義的名稱以識別此群組。

CH101_31_admin

最多 128 個字元，使用英數和 '!=. @-' 字元。

2. 設定群組 Policy 權限

選擇許可政策 - 選用 (536) 此頁面列出了所有可用於此群組的政策。這些政策將應用於群組中的所有使用者。

政策名稱	類型	描述
AWSDirectConnectReadOnlyAccess	AWS 受管	Provides read only access
Amazon Glacier ReadOnlyAccess	AWS 受管	Provides read only access
AmazonMarketplaceFullAccess	AWS 受管	Provides the ability to submit offers
AWSSSOAdminGroupAdministrator	AWS 受管	Administrator access for SSO
AWSSiteToSiteReadOnlyAccess	AWS 受管	Provides read only access
AutoScalingConsoleReadOnlyAccess	AWS 受管	Provides read-only access
AmazonDMSReadOnlyTSGRole	AWS 受管	Provides access to managed databases
AWSQuickSightListIAM	AWS 受管	Allows QuickSight to list IAM users

下方紅色標記為最高權限 (如 Linux 的 sudo)

"admin" X 滾動選項條件

政策名稱	類型	描述
AWSSSOAdminGroupAdministrator	AWS 受管	Administrator access for SSO
CloudWatchAgentAdminPolicy	AWS 受管	Full permissions required
DatabaseAdministrator	AWS 受管 – 跟踪鑑權	Grants full access permissions
AWSSSOMasterAccountAdministrator	AWS 受管	Provides access within AI
AWSCloud9Administrator	AWS 受管	Provides administrator access
AWSSSOMemberAccountAdministrator	AWS 受管	Provides access within AI
SystemAdministrator	AWS 受管 – 跟踪鑑權	Grants full access permissions
<u>AdministratorAccess</u>	AWS 受管 – 跟踪鑑權	Provides full access to all services

AdministratorAccess
Provides full access to AWS services and resources.

```

1- [ { "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "*", "Resource": "*" } ] }
  
```

3. 群組內建立使用者

允許群組內的使用者做任何事情，因為整個群組的政策為 Administrator Access



4. IAM 建立完成

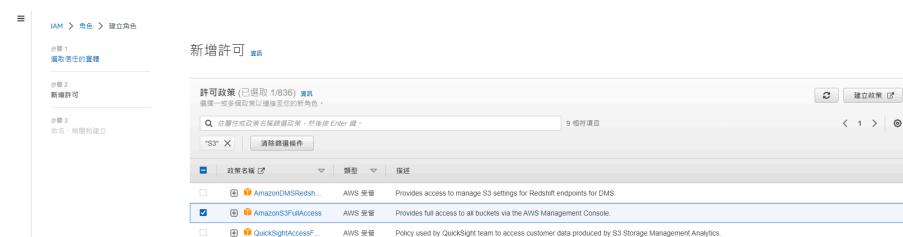
▼ 二、建立 IAM Role，針對特殊角色設定權限與規則

整體流程為：選擇角色服務 > 建立許可政策 > 建立角色

1. 進到建立角色中，先選擇建立 AWS 服務，這裡可以建立其他實體類型



2. 設定許可政策



3. 建立角色，以及詳細設定



補充：今後 IAM 會搬到 ASSO 上

▼ 【IAM 權限政策設定實作 & 權限說明】

由於 AdministratorAccess 這個政策等於是所有 su 的權限都開了，因此建議可以針對需求開立

目前所需要的功能為 EC2、ECS、RDS、S3、監控數據、查看帳單，
因此只需要開立

1. SystemAdministrator：針對基礎設施相關的管理操作，如 EC2 和 S3，但不一定包含完整的 RDS 和 ECS 管理權限。
2. AmazonRDSFullAccess：RDS。
3. CloudWatchReadOnlyAccess：能查看所有服務的監控數據和效能狀況。
4. Billing：查看所有帳單數據

▼ 【重新設定子帳號的密碼】

Step1. 使用 Root 帳號登入

Step2. 進到 IAM>人員>選擇要修改的人員管理介面中 > 選安全憑證 > 管理主控台存取

1. IAM > 人物 > Sam

2. 安全憑證

3. 管理主控台存取

Step3. 修改密碼 > 完成

管理主控台存取

停用主控台存取
停用會移除既有的密碼。

重設密碼
自動產生或設定新的自訂密碼。

自動產生的密碼

自訂密碼

顯示密碼

人員在下次登入時必須建立新的密碼
人員會自動取得 IAMUserChangePassword 政策，以允許他們變更自己的密碼。

撤銷作用中主控台工作階段
人員會自動附加以下內嵌政策，以撤銷其作用中的工作階段。

▶ AWSRevokeOlderSessions

取消 重設密碼

VPC

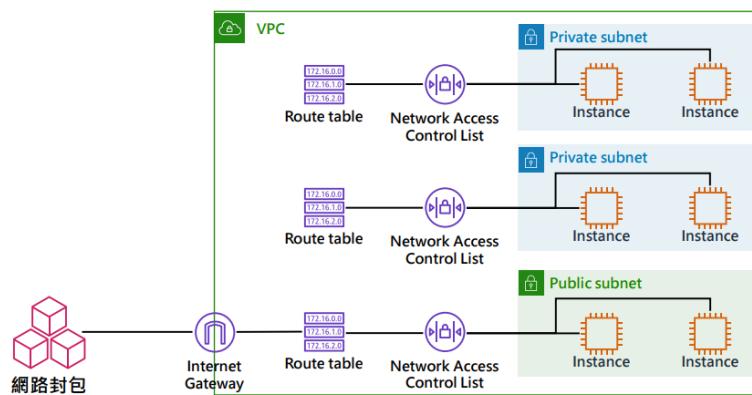
▼ 【VPC 介紹】Virtual Private Cloud 虛擬網路

可以在 AWS 中創建一個自己的**私有網路**，並控制其中運行的資源之間的連接和流量，達到完全掌控網路環境，包括 IP 位址範圍、子網路、路由表和網路閘道等

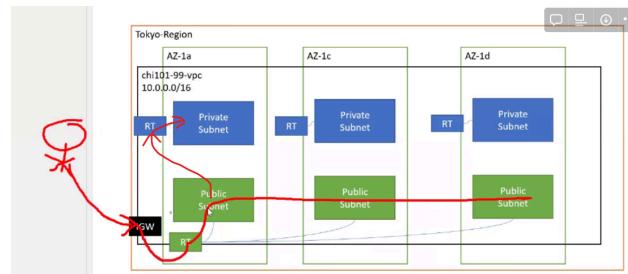
可以使用安全性控制來保護您的 VPC，例如使用網路存取控制列表（Network ACLs）和安全性群組（Security Groups）等，進一步加強對您的資源的保護

VPC 也支援與其他 **AWS 服務的整合**，例如 Amazon EC2、Amazon RDS、Amazon S3 等，讓您可以在 VPC 中運行這些服務並進行相應的配置和管理

▼ 【VPC 樣貌】



移動示意圖



網路封包是一個包含了源地址、目的地址、協定類型和數據等信息的數據包，用於在網路上傳輸數據

互聯網上的網路封包透過 **internet Gateway(IGW)** 將來自互聯網的數據包轉發到 VPC 內部，當然也可以將來自 VPC 內部的數據包轉發到互聯網上

而 **Route table(RT)** 是用於路由網路封包的一個表格，用於指定網路封包從哪裡進入和離開 VPC，每個 VPC 都有一個**默認的 Route table**，用戶可以創建和管理自己的 Route table。在 Route table 中，用戶可以指定網路封包的**路由規則**，例如將網路封包發送到 Internet Gateway 或其他 VPC。

- 網路封包的路由規則就是用於指定網路封包的轉發路徑，舉三例：

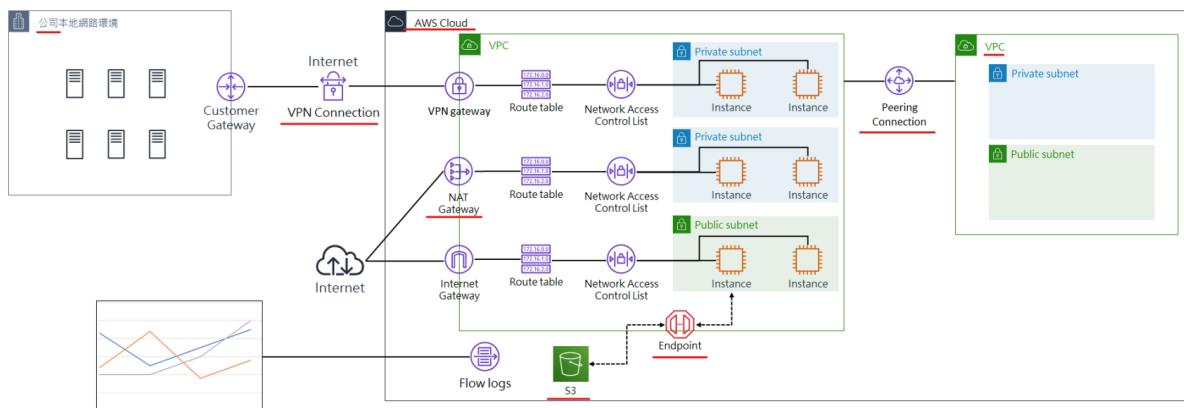
1. 網路封包進入 VPC 時 RT 會依據規則轉換到 VPC 內的指定 Instance
2. VPC 內的 Instance 與 Instance 之間傳遞，也需要經由 RT
3. VPC 內的 Instance 要將資訊網外不傳，也會需要 RT 透過 IGW 轉發到公網

Network Access Control List (NACL) 是用於控制子網(subnet)流量的一個安全層 (Subnet的靜態防火牆)，可以控制 subnet 與 subnet 之間的流量，也可以控制 VPC 進出的流量，在 NACL 中，用戶可以設置不同的入站和出站規則，例如允許或禁止特定的 IP 地址、協定或端口進行通訊

Private subnet 僅可進行內部訪問的應用程序或服務，如：運行資料庫服務或應用程序服務

Public subnet 可以運行一些需要公網訪問的應用程序或服務，例如 Web 服務或 API 服務
Instance 是在 VPC 中運行的虛擬機器

▼ 【VPC 連線方式】



1. Peering Connection (在AWS雲端上，VPC與VPC之間的連線模式)
2. VPN Connection (公司網路連上VPC的連線模式)
3. VPC Endpoint (VPC連上AWS服務的連線模式)
4. NAT Gateway (外部網路直接連上VPC私網EC2的連線模式，不用透過公有EC2)

▼ 【VPC 建立實作】建立私有網路，包括 IP 位址範圍、子網路、路由表和網路閘道

建立 VPC 機房可以提供一個完全獨立的虛擬網路環境，以實現安全性和網路隔離等需求。另外，建立多個 VPC 並使用 VPC Peering 功能可以實現多個獨立 VPC 之間的網路連通，方便不同應用程式之間的資料交換和共享。

目前調整兩個地方：

1. 只需要自動產生名稱標籤欄位填入我們的專案名稱

2. 調整可用區域 (AZ) 數量

VPC > 您的 VPC > 建立 VPC

建立 VPC

VPC 是由 AWS 物件 (例如 Amazon EC2 實行產地) 所攝入的 AWS 環境資源部分。將清單移至資源導航，將相關資訊反白。

VPC 設定

要建立的資源 **資源**
選擇 VPC、子網和 VPC 和其他聯繫資源。
 建立 VPC VPC 和更多

自動產生名稱標籤 **資源**
輸入各項資源的名稱，此項將應用於 VPC 中的所有資源自動產生名稱標籤。
 自動產生
chi10131

IPv4 CIDR 雜湧 **資源**
使用 CIDR 雜湧來自動分配 IP 和您 VPC 的大小。
10.0.0.0/16 65,536 IPs

IPv6 CIDR 雜湧 **資源**
 IPv6 CIDR 雜湧
 由 Amazon 提供的 IPv6 CIDR 雜湧

選用 **資源**
預設

可選區域 (AZ) 數量 **資源**
每個子網有其自身的可用區域數量。建議至少要有兩個可用區域，以維持彈性可用性。
1 2 3
► 白訂 AZ

公用子網的數量 **資源**
每個子網有其自身的公用子網數量。如果 Web 服務需要讓使用者直接存取網站內容，請考慮為子網提供一個公用子網。

0 3
私有子網的數量 **資源**

預覽

VPC **啟用詳細資訊**
您的 AWS VPC 連接埠

子網 (6)
此 VPC 中的子網
chi10131-vpc

ap-northeast-1a
chi10131-subnet-public1-ap-northeast-1a
chi10131-subnet-private1-ap-northeast-1a
ap-northeast-1c
chi10131-subnet-public2-ap-northeast-1c
chi10131-subnet-private2-ap-northeast-1c
ap-northeast-1d
chi10131-subnet-public3-ap-northeast-1d
chi10131-subnet-private3-ap-northeast-1d

路徑表 (4)
所建立之子網與子網間的路徑表
chi10131-rtb-public
chi10131-rtb-private1-ap-northeast-1a
chi10131-rtb-private2-ap-northeast-1c
chi10131-rtb-private3-ap-northeast-1d

網路連線 (2)
其他網域的連線
chi10131-igw
chi10131-vpc-s3

跑完後就完成 VPC 建立了

VPC > 您的 VPC > 建立 VPC > 建立 VPC 資源

建立 VPC 工作流程

建立子網路
38%

▼ 詳細資訊

建立 VPC: vpc-04536846313a6946a

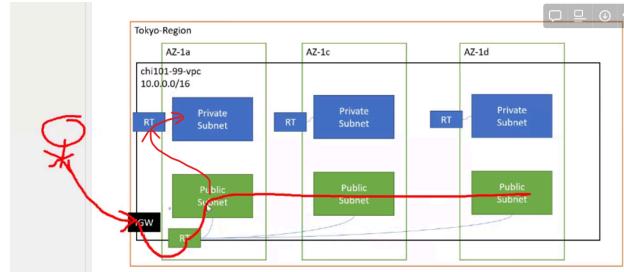
啟用 DNS 主機名稱
 啟用 DNS 解析
 延遲 VPC 的建立: vpc-04536846313a6946a
 建立 S3 端點: vpc-e067eff0a3e575e87
 建立子網路: subnet-0be08d0500fd8e88
 建立子網路: subnet-09e9fbabfa88ed7b
 建立子網路: subnet-09d75994b04cd1ea8
 建立子網路: subnet-0c71ab99f5631c3f8
 建立子網路: subnet-0bc834728a7daee0d
 建立子網路
 建立網際網路路閘道
 將網際網路路閘道連接至 VPC
 建立路由路

VPC > 您的 VPC > vpc-04536846313a6946a / robo_advisor-vpc

詳細資訊

VPC ID	狀態	DNS 主機名稱	DNS 解析
vpc-04536846313a6946a	Available	已啟用	已啟用
範圍 Default	DHCP 地址池 dopt-0312eb09e5179c5	主節點名 rtb-0x2edf8e420d5461	主節點 ACL act-065dbd5a70a3e161f
網域 VPC 子網	IPv4 CIDR 10.0.0.0/16	IPv6 佈署 -	IPv6 CIDR (網路邊界對組) -
網域位址使用指標 已停用	Route 53 Resolver DNS 防火牆規則對組 -	網域 ID 268246653390	-

概念圖：



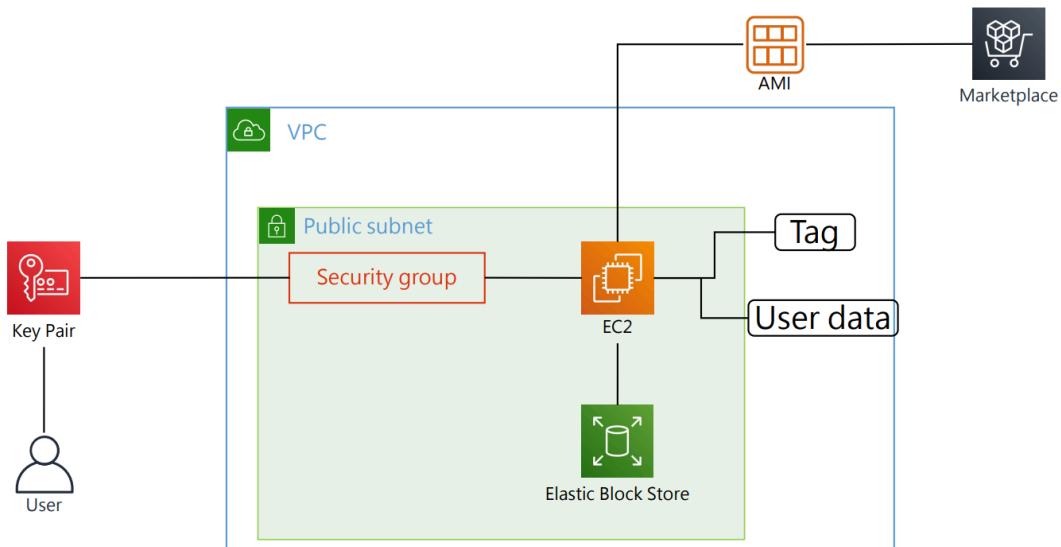
EC2

▼ 【EC2 介紹】Elastic Computing Cloud

EC2 是 AWS 提供的虛擬設備服務，透過下方圖是可以了解到建置 EC2 的流程



▼ 【EC2 組成架構圖】



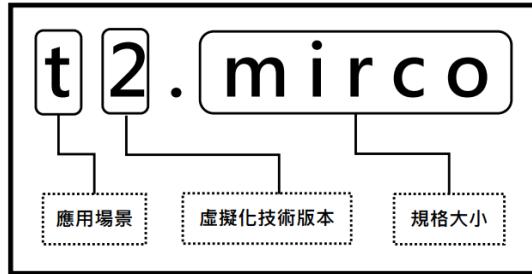
▼ 【核心名詞】

▼ Amazon Machine Image(AMI)：虛擬主機的系統映像檔

AMI 就是 EC2 用來快速開啟虛擬主機的系統映像檔，在創建 EC2 實例時，可以選擇一個適合自己需求的 AMI，然後啟動這個 AMI 所描述的虛擬機器。這樣就可以在幾分鐘內創建一個符合自己所需的 EC2。
坊間會將自己製作好的 AMI 放在 AMI Marketplace 讓人選購。

▼ Instance Type : 機器規格

機器規格，可以選擇CPU、記憶體、硬碟空間、網路傳輸速度等等



▼ User data : 自動環境建構指令，類似YAML 檔

當我們希望運行新的虛擬主機的時候，有些軟體或設定就已經自動載入完成，就可以將資訊放到 User Data 內，當第一次開機的時候，就會自動執行該檔案內的指令完成環境建置，類似 docker-composed 的 YAML 檔

▼ Elastic Block Service(EBS) : 塊級存儲的服務

EBS 可以為 EC2 實例提供持久的塊級存儲，並且可以為 EC2 實例提供快速和靈活的存儲。EBS 硬碟的大小和類型，例如，SSD 或 HDD

塊級存儲 (Block-level storage) 是指將數據劃分為一個個固定大小的塊 (block)，每個塊可以獨立地被讀取、寫入和管理。每個塊都有自己的地址，這樣就可以在需要時隨時訪問和操作數據。

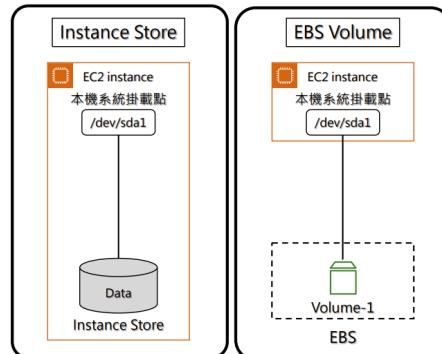
相比於對象存儲或檔案存儲等存儲方式，塊級存儲通常具有更高的性能和更低的延遲。因為每個塊都可以直接被訪問，而不需要將整個對象或檔案進行讀取或寫入。這使得塊級存儲在需要高效、低延遲存儲和訪問大量小文件的場景中具有優勢，例如虛擬機器 (VM) 的儲存、資料庫系統、高性能運算和分佈式文件系統等。

但還是有劣勢，例如：

1. 無法靈活地擴展存儲空間
2. 管理塊級存儲的複雜度可能會比其他存儲方式高
3. 需要額外的硬體設備來管理和操作塊，增加系統的成本和複雜度

▼ Instance Store : 另外一種儲存方式，專用於儲存暫時性資料

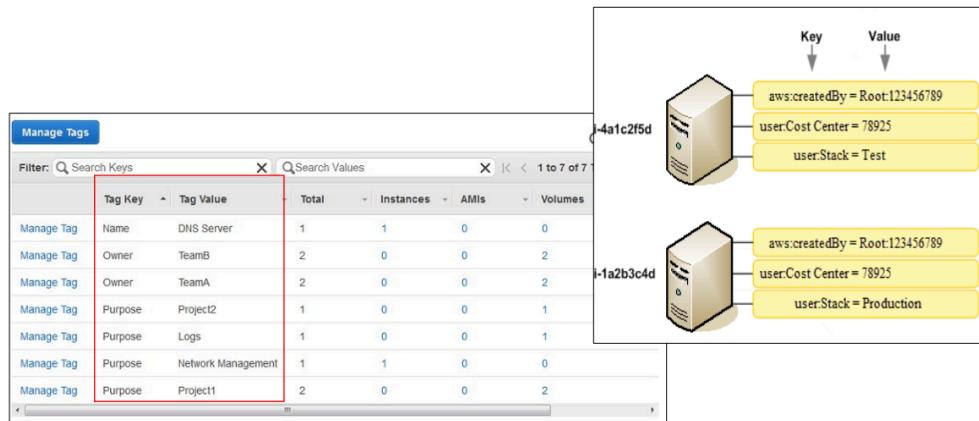
與虛擬主機放置在同一台實體機上，關機開機後，存放在裡面的資料則清空。



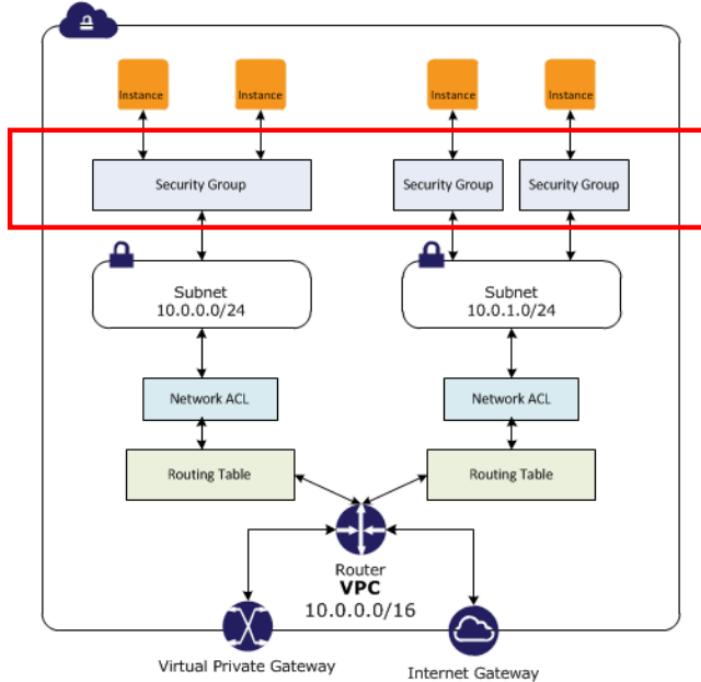
Instance Store & EBS 對照圖

58

▼ Tag : 我方自定義資源，作為資源管理與權限設定的用途



▼ Security Group : 防火牆，用來管理虛擬主機的網路流量



▼ Key Pair : 遠端虛擬主機，預設不開放帳密登入，通通使用金鑰做遠端登入

EC2 的虛擬主機連線需要使用金鑰。

此金鑰可從 EC2 上進行申請或自行匯入。

若從 EC2 上進行申請，金鑰發放後，為了安全，不予補發。

Select an existing key pair or create a new key pair X

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

▲

Key pair name:

Tip: You have to download the **private key file (*.pem file)** before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

▼ 【購買方式】

主要有六種，隨需、預留、排程、專用、競標、Saving plans(未來新趨勢)

其中「隨需」最貴，但也是有使用才會花錢

▼ 【EC2 建立實作】 EC2 最核心的運算服務-軟硬體設定

▼ 點選啟動執行個體



▼ 輸入名稱和標籤

Name and tags 資訊

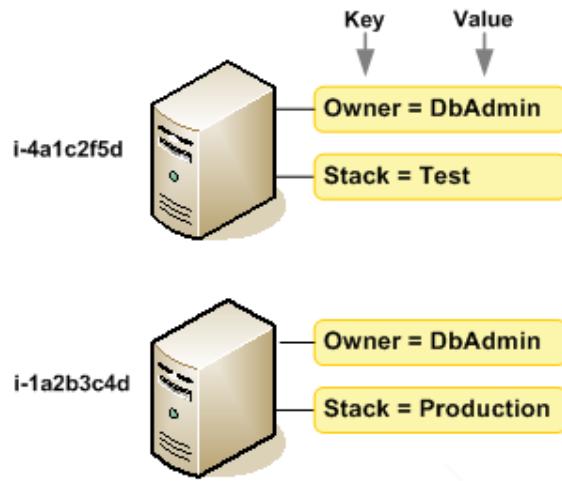
名稱	<input type="text" value="robo_advisor"/>	Add additional tags
▼ Name and tags 資訊		
金鑰 資訊	值 資訊	Resource types 資訊
<input type="text" value="Name"/> X	<input type="text" value="輸入值"/> X	<input type="text" value="Select resource ty..."/> X
新增標籤		
49 remaining (Up to 50 tags maximum)		

Name：建議為EC2實例取一個該實例用途的名稱，例如"Web Server"或"Database Server"

Tag：可以用來分類 AWS 資源，因此可以依據目的、擁有者、環境...進行標註

Key：建議為EC2實例取一個該實例用途的名稱，例如"Web Server"或"Database Server"，或是也可以用來作為 Owner 的概念經營，在 Key 欄位中填入負責人的姓名或團隊的名稱

Value：用來描述 Key



▼ 【Key Pair】

管理員會有一組公鑰與私鑰，使用者會要求一組公鑰，獲取金鑰後會進行加密，然後與私鑰比對，就可以允許進入

▼ 開始選購作業系統、應用程式伺服器和應用程式範本

▼ 應用程式和作業系統映像 (Amazon Machine Image) 資訊

AMI 是一種範本，包含啟動執行個體所需的軟體組態 (作業系統、應用程式伺服器和應用程式)。如果在下面看不到所尋找的內容，請搜尋或瀏覽 AMI

搜尋我們的完整型錄，包括 1,000 個應用程式和作業系統映像

[快速入門](#)

Amazon Linux 	macOS 	Ubuntu 	Windows 	Red Hat 	S >
---	--	---	--	--	-----

[瀏覽更多 AMI](#)
包括來自 AWS、Marketplace 和社群的 AMI

可以點選瀏覽更多 AMI 也可以到 AMI Marketplace 下載，我們這邊選用

「Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type」，因為是 AWS 自己開發的 Linux 環境會比較穩定好用

▼ 執行個體類型，選擇CPU、記憶體、硬碟空間、網路傳輸速度等等

Instance Type : t2.micro

▼ 建立開機器的金鑰

如果未來要使用虛擬機登入，就會需要使用 `ppk` 金鑰，但如果選擇錯誤也沒關係，有軟體可以協助將 `pem` 轉 `ppk` 選擇

金鑰名稱沒有要特別注意的事情，記住就好，建立完成後會提供一個 `ppk` 檔，記得留好，遺失不會補發



▼ 網路設定，設定防火牆規則，以此用來管理虛擬主機的網路流量

點選編輯 > 選擇 VPC 機房 > 設定兩組防火牆規則，一組SSH，一組HTTP，這樣可以防止其他不允許進來的安全環境 > 設定儲存空間 > 設定進階詳細資訊(User data)

在設定 EC2 的網路時，安全群組名稱應該取一個具有代表性的名稱，以便使用者在之後的管理中能夠快速識別該安全群組的用途。

一般來說，可以依據該安全群組所要保護的服務或應用程式進行命名，例如 Web Server 安全群組、Database 安全群組等等。

另外，也可以在名稱中加入該安全群組所屬的 VPC 名稱或環境名稱，以更清楚地區分不同網路和環境的安全群組。



▼ 建立兩組防火牆規則，一組SSH，一組HTTP

這是出於安全性和最佳實踐考慮的原因

▼ SSH防火牆規則

SSH是一種安全的遠程登錄協議，用於對EC2實例進行遠程管理和訪問。建立一組SSH防火牆規則可以限制僅允許特定IP地址或IP地址範圍從安全連接進行SSH訪問到您的實例。這樣可以減少不必要的風險，確保只有授權的用戶可以訪問和管理實例。

▼ HTTP防火牆規則

HTTP是用於Web應用程序的通信協議，用於向EC2實例提供網頁訪問。建立一組HTTP防火牆規則可以限制僅允許特定IP地址或IP地址範圍通過HTTP協議訪問您的應用程序。這樣可以確保只有授權的用戶可以訪問您的Web應用程序，同時減少潛在的安全風險和未經授權的訪問。

通過這樣的安全組設置，您可以細粒度控制進入EC2實例的網絡流量。只有具有相應權限和IP地址的用戶才能通過SSH或HTTP協議訪問您的實例，這有助於保護實例的安全性並減少潛在的風險。同時，建議使用更強大的密碼和密鑰來增加SSH訪問的安全性，並實施其他安全措施以保護您的應用程序和數據。

描述 - 必要 資訊
robo_advisor created 2023-04-30T02:04:25.080Z

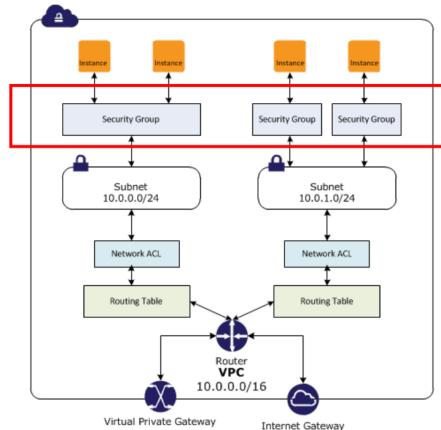
傳入安全群組規則

▼ 安全群組規則 1 (TCP, 22, 0.0.0.0/0)

類型 資訊 ssh	通訊協定 資訊 TCP	連接埠範圍 資訊 22
來源類型 資訊 隨處	來源 資訊 新增 CIDR、前綴清單或安全群	描述 - optional 資訊 例如：用於管理員桌面的 SSH 0.0.0.0/0 X

▼ 安全群組規則 2 (TCP, 80, 0.0.0.0/0)

類型 資訊 HTTP	通訊協定 資訊 TCP	連接埠範圍 資訊 80
來源類型 資訊 隨處	來源 資訊 新增 CIDR、前綴清單或安全群	描述 - optional 資訊 例如：用於管理員桌面的 SSH



▼ 設定儲存空間

▼ 設定儲存 資訊

進階

1x GiB 根磁碟區 (未加密)

ⓘ 符合免費方案資格的客戶可獲得最多 30 GB 的 EBS 一般用途 (SSD) 或磁性儲存空間 X

新增新磁碟區

0 x 檔案系統 編輯

▼ 設定 User data



*安裝apache web server的指令

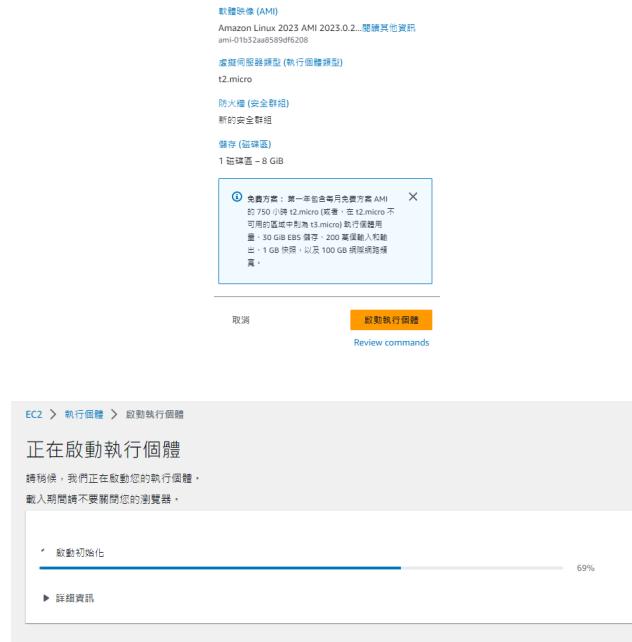
```
#!/bin/bash
sudo yum install -y httpd
sudo service httpd start
sudo chkconfig httpd on
```

1. **sudo yum install -y httpd**：這個命令使用 **yum** 套件管理器安裝Apache HTTP Server (**httpd**) 軟體包。**y** 參數用於自動確認安裝過程中的提示，以確保無需手動確認安裝。
2. **sudo service httpd start**：這個命令啟動安裝的Apache HTTP Server服務。透過這個命令，EC2實例將開始執行HTTP服務，使其能夠接收和處理HTTP請求。
3. **sudo chkconfig httpd on**：這個命令設置Apache HTTP Server在EC2實例啟動時自動啟用。通過這個命令，您確保了每次EC2實例重新啟動時HTTP服務都會自動啟動。

總的來說，這些命令的目的是在EC2實例啟動時自動安裝和配置Apache HTTP Server，使其能夠正常運行和處理HTTP請求。這對於託管網站或應用程序的EC2實例

非常有用，因為它可以自動完成HTTP服務的設置，而無需手動進行安裝和啟動。

▼ 規格確定後，就可以啟用 EC2



正在啟動執行個體
正在啟動您的執行個體，
載入期間請不要關閉您的瀏覽器。
啟動初始化
詳細資訊

▼ 接下來就是等 EC2 機器開機

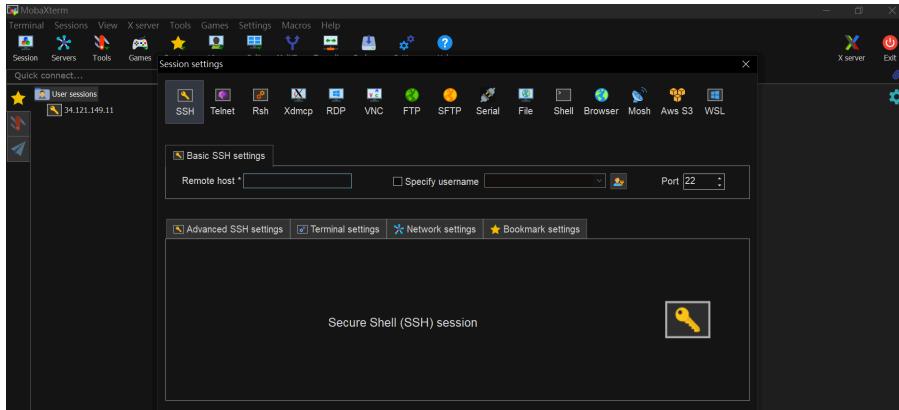


▼ 開機完成後就會顯示執行中

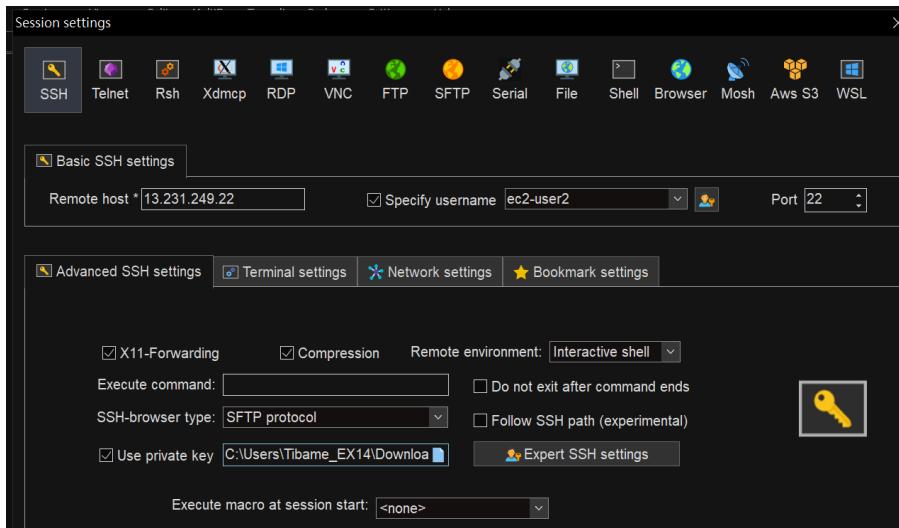


▼ 【EC2 連線虛擬主機】 - 使用 MobaXterm 手動/自動安裝 apache web server

使用我們剛剛建立的金鑰檔案



會使用 公有 IPv4 地址 , 機器如果關機或開機, 就會更新, 所以如果要架設網站, 就會需要使用靜態 IP (詳見下方補充)



在執行介面中輸入：以下語法，建立「apache web server」「輸入公有IP查看網頁」

※安裝apache web server的指令

sudo yum install -y httpd

sudo service httpd start

sudo chkconfig httpd on

如果要自動安裝的流程，建立 EC2 差不多，差異只在最後的 User data 中輸入，下方程式碼，即可建立公有IP網頁、安裝apache web server

```
#!/bin/bash
yum install -y httpd
service httpd start
chkconfig httpd on
```

S3

▼ 【S3 介紹】Simple Storage Service

安全、耐久、可高度擴展的雲端儲存(專門給開發人員的雲端儲存空間)，Amazon S3 是易於使用的物件儲存，

且具備一個簡單的 Web 服務界面，可讓您在 Web 上隨處存放和擷取任意數量的資料。

▼ 【S3 觀念】

CDN (Cloud Front) 將所有資訊(多媒體物件、HTTPS ...)，分散到各國 CDN 節點，進行儲存備份，資料(S3)取用的人，可以找最近的節點進行取用(S3)

CDN 也可以防範駭客，因為只會攻擊到 CDN 節點，節點都會清洗到乾淨後，回傳到原始資料庫

▼ 【S3 名詞說明】

Bucket：存儲桶，名字全球唯一

Object：即為放在bucket內的檔案，每個帳號有軟性限制(Soft limit)，最多只能有一百個bucket。

每個bucket可存儲的物件數量無上限，單個物件大小最大可支援到5TB

下方為儲存級別，但會持續更新，因此建議參考官方文件

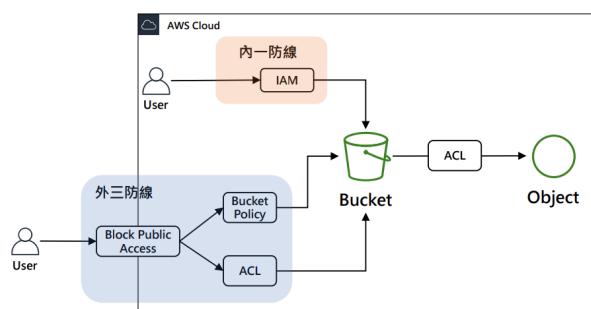
<https://aws.amazon.com/tw/s3/storage-classes/>

儲存級別 (Storage-Level)						
	S3 標準	S3 智慧型分層*	S3 標準 – IA	S3 單區域 – IA†	S3 Glacier	S3 Glacier Deep Archive
專為耐用性而設計	99.999999999% (11 9's)					
專為可用性而設計	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
可用性 SLA	99.9%	99%	99%	99%	99.9%	99.9%
可用區域	≥3	≥3	≥3	1	≥3	≥3
每個物件的最低容量費用	無	無	128 KB	128 KB	40 KB	40 KB
最短儲存持續時間費用	無	30 天	30 天	30 天	90 天	180 天
擷取費用	無	無	每 GB 擷取	每 GB 擷取	每 GB 擷取	每 GB 擷取
第一位元組延遲	毫秒	毫秒	毫秒	毫秒	選取分鐘或小時	選取小時
儲存體類型	物件	物件	物件	物件	物件	物件
生命週期轉換	是	是	是	是	是	是

* 因為 S3 單區域 – IA 將資料存放在單一 AWS 可用區域中，因此存放在這個儲存類別中的資料會在可用區域損壞時遺失

72

▼ 【S3 管理權限】NACL



在 AWS 中， Network Access Control List (簡稱 NACL)

NACL 是一種**有序的防火牆規則集**，可應用於 VPC 的子網路和子網路對應的子網路 ACL，以允許或拒絕通過 VPC 中每個子網路的流量。

NACL 可以用來**控制特定 IP 位址或區域的流量**，也可以**控制某些 TCP 或 UDP 端口上的流量**，而且可以**對進出 VPC 的流量進行控制**。

NACL 是在子網路層級上進行操作，而不是在單個 EC2 實例層級上進行操作。NACL 通常用於網路層面的安全需求，例如防火牆，而不是對 EC2 實例上的應用程序進行控制。

總的來說，NACL 可以用來增加網路層級的安全性，通過過濾流量控制 EC2 實例和子網路的進出流量，從而**保護 VPC 的安全性**。

▼ 【NACL 的控制權限】

參考資料：ACL (Access Control List)

<https://ithelp.ithome.com.tw/articles/10221185>

使用者(user)：針對使用者設定權限

群組(group)：針對群組設定權限

預設屬性(mask)：該目錄新建檔案/目錄時，**規範新資料的預設權限**。

▼ 【NACL、Security Group】 防火牆設定

NACL 不帶狀態防火牆，進出都需要驗證

Security Group 帶狀態防火牆，只對進入把關，對於離開不做管理

▼ 【S3 建立實作】

▼ 1. 建立儲存貯體



▼ 2. 設定 ACL

物件擁有權 Info
控制從其他 AWS 帳戶寫入此儲存貯體之物件的擁有權，以及存取控制清單 (ACL) 的使用。物件擁有權決定誰可以指定對物件的存取權。

ACL 已停用 (建議)
此儲存貯體中的所有物件都由此帳戶擁有。僅使用政策指定此儲存貯體及其物件的存取權。

ACL 已啟用
此儲存貯體中的物件可由其他 AWS 帳戶擁有。可使用 ACL 指定此儲存貯體及其物件的存取權。

⚠️ 除非您需要個別控制每個物件的存取，或讓物件寫入器擁有上傳的資料，否則建議您停用 ACL。使用儲存貯體政策 (而非 ACL) 與帳戶外的使用者共享資料，可簡化許可管理及稽核。

▼ 3. 設定公開權限

因為課程需要公開，所以解除，但這樣會對資源有疑慮

此儲存貯體的「封鎖公開存取」設定

系統是透過存取控制清單 (ACL)、儲存貯體政策、存取點政策或所有這些項目將公有存取權授與儲存貯體和物件。為了確保此儲存貯體和物件的公有存取權已封鎖，請開啟「封鎖所有公有存取權」，這些設定僅適用於此儲存貯體及其存取點。AWS 建議您開啟「封鎖所有公有存取權」，但在套用任何這些設定之前，確保您的應用程式能在沒有公有存取權的情況下正常運作。如果您需要此儲存貯體或物件的一些公有存取層級，您可以在下方自訂個別設定，以滿足您的特定儲存使用案例。[進一步了解](#)

封鎖所有公開存取權
開啟此設定等同於開啟以下所有個別設定。下列每個設定都是相互獨立的。

- 封鎖透過新的存取控制清單 (ACL) 授予的儲存貯體和物件的公開存取權**
S3 將封鎖套用至剛新增儲存貯體或物件的公開存取權，並防止針對現有儲存貯體和物件建立新的公開存取 ACL。此設定不會變更任何現有的允許使用 ACL 公開存取 S3 資源權限。
- 封鎖透過任何存取控制清單 (ACL) 授予的儲存貯體和物件的公開存取權**
S3 會忽略授與儲存貯體和物件公開存取權的所有 ACL。
- 封鎖透過新的公開儲存貯體或存取點政策授予的對儲存貯體和物件的公開存取權**
S3 將封鎖新的儲存貯體和存取點政策，該政策授與儲存貯體和物件的公開存取權。此設定不會變更任何現有的允許公開存取 S3 資源的政策。
- 封鎖透過任何公開儲存貯體或存取點政策授予的對儲存貯體和物件的公有和跨帳戶存取權**
S3 將忽略對儲存貯體或存取點的公開和跨帳戶存取，這些儲存貯體採用授與儲存貯體和物件公開存取權的政策。

⚠️ **關閉「封鎖所有公有存取權」可能導致此儲存貯體及其內的物件變成公有狀態**
AWS 建議您開啟「封鎖所有公有存取權」，除非經過驗證的特定使用案例 (例如靜態網站託管) 需要公有存取權。

▼ 4. 儲存貯體版本控制

儲存貯體版本控制

版本控制是在相同儲存貯體中保留物件多個變體的方法。您可以使用版本控制來保留、擷取和還原存放在 Amazon S3 儲存貯體中每個物件的每個版本。透過版本控制，您可以從非預期使用者動作和應用程式失敗中輕鬆復原。[進一步了解](#)

儲存貯體版本控制

停用

啟用

▼ 5. 建立資料夾與上傳檔案，步驟如同 google drive 一樣建立資料夾

The screenshot shows the Amazon S3 console interface. At the top, it says 'Amazon S3 > 儲存容器 > chi101-31 > 建立資料夾'. Below that, there's a section titled '建立資料夾' with an 'Info' link. A note says: '您的儲存容器政策可能會封鎖資料夾建立。如果您將儲存容器中的物件分組，當您建立資料夾時，S3 會使用您指定的名稱後接斜線 (/) 來建立物件。此物件隨後會在主控制台顯示為資料夾。進一步了解'.

In the main area, there's a '資料夾' section with a '資料夾名稱' input field containing 'photos'. Below it, there's a '伺服器端加密' section with a note: '伺服器端加密可保護靜態資料'.

At the bottom, a green banner says: '已成功建立資料夾 "photos".' and '操作已成功完成'.

Then, the 'chi101-31' bucket page is shown with tabs for 物件、屬性、許可、指標、管理、存取點. The '物件' tab is selected, showing one item: 'photos/' (資料夾). The '屬性' tab is selected in the bottom navigation bar, showing detailed information about the 'cat001.jpg' file:

- 名稱: cat001.jpg
- 類型: JPEG
- 上次修改時間: 2023年4月25日 pm4:51:55 CST
- 大小: 54.9 KB
- S3 URI: <https://chi101-31.s3-ap-northeast-1.amazonaws.com/photos/cat001.jpg>
- Amazon Resource Name (ARN): arn:aws:s3:::chi101-31/photos/cat001.jpg
- 置存標籤 (ETag): 0001665f18275ce146779ee88e564a63
- 物件 URL: <http://chi101-31.s3-ap-northeast-1.amazonaws.com/photos/cat001.jpg>

檔案資料建立完成後，可以查看檔案，但因為目前檔案尚未公開，所以會出現下方圖示

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```

▼<Error>
  <Code>AccessDenied</Code>
  <Message>Access Denied</Message>
  <RequestId>EHTWB429QC8G4A2Z</RequestId>
  <HostId>JIErzfLtJ0hR13nP3wkd01f1AAMifDEFH3al4dxRGdWPPEwEAnXyuMDcq5nnvZIeJ4JjsgAdbo=</HostId>
</Error>

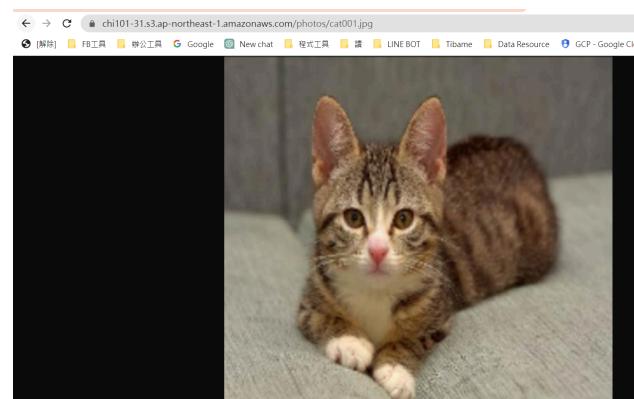
```

只要「使用 ACL 設為公有」



The screenshot shows the 'Set Public' dialog in the AWS S3 console. It includes a warning message about public access and a table listing objects in the 'photos/' folder. A prominent orange '設為公有' (Set Public) button is at the bottom right.

再次執行連結後，就可以成功讓他人查閱檔案了



S3 靜態網站託管

CLI

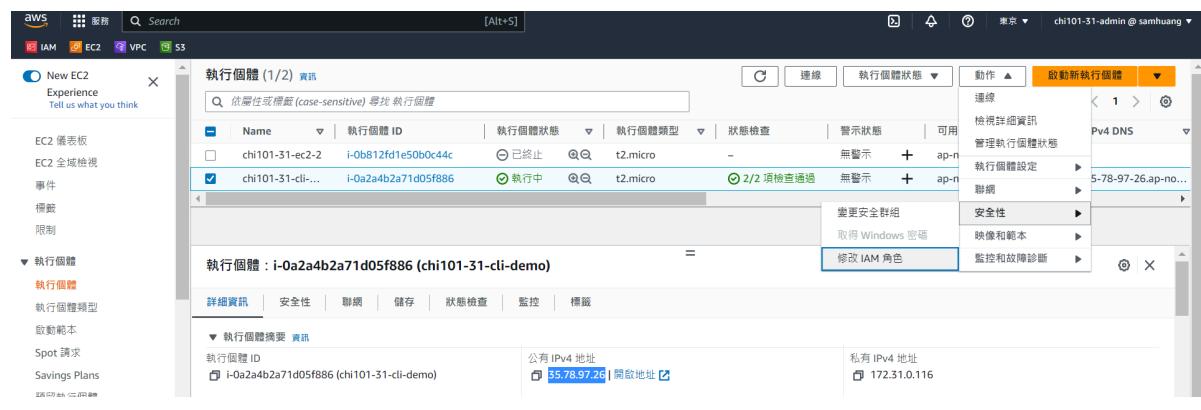
▼ 【CLI 介紹】AWS Command-Line Interface

聲明式文件(寫好文件後，下一個指令啟動，就可以直接幫我們安裝好所有資源，效果與 docker composed 一樣，boto3 自動化執行程式，文件格式可以是 yaml 或 JSON，作業流程為 yaml 壓縮成 JSON 再呼叫 API 然後開始建立我們的環境以及功能)

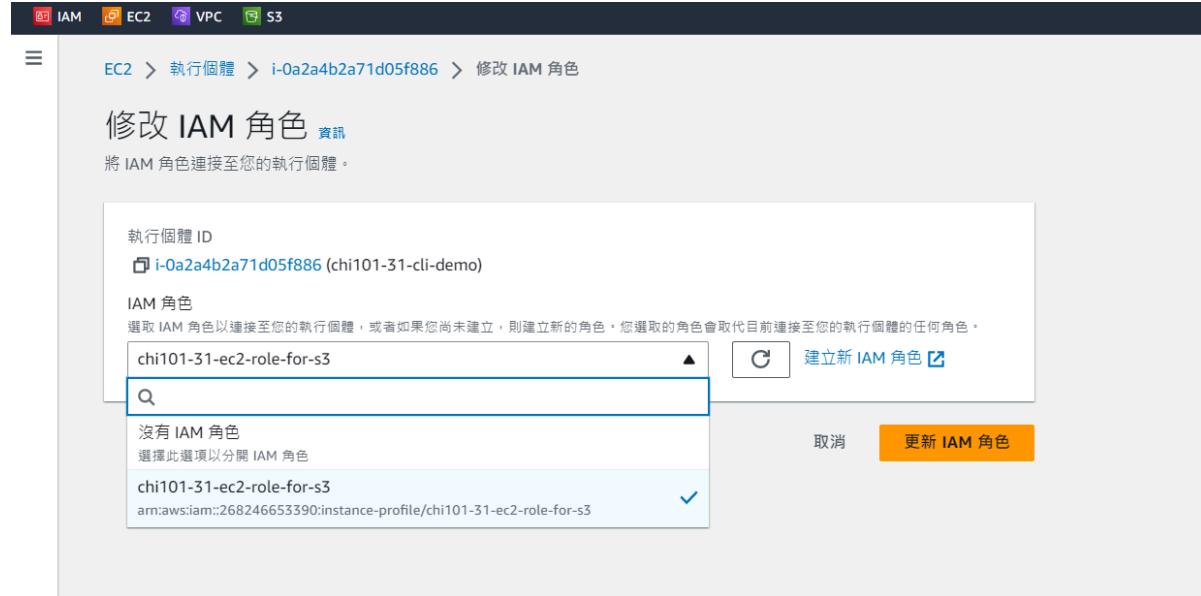
CLI 通常是透過命令列介面進行操作的工具，不過，有些工具可能會提供額外的圖形化介面

可以參考文件轉寫方法

<https://aws.amazon.com/tw/sdk-for-python/>



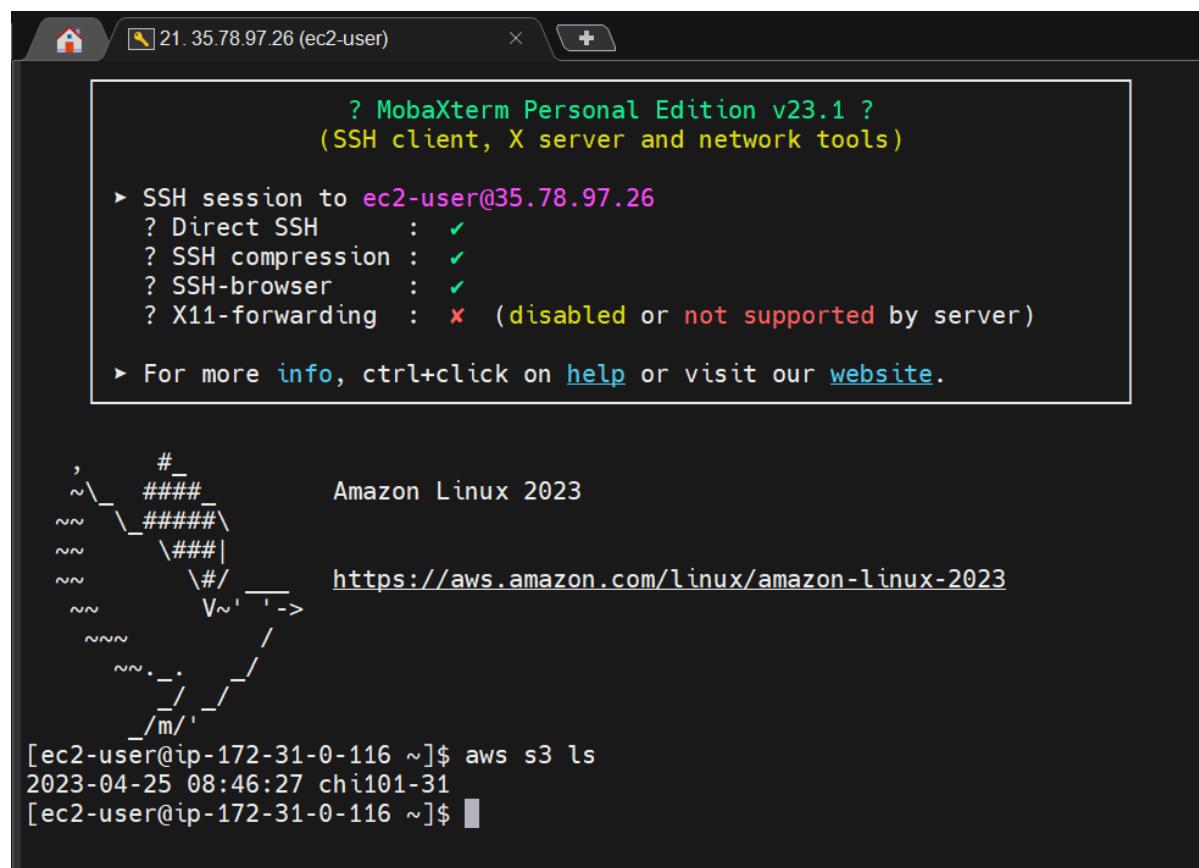
The screenshot shows the AWS Lambda console interface. On the left, there's a sidebar with navigation links like 'New EC2 Experience', 'EC2 價表板', 'EC2 全域檢視', '事件', '標籤', '限制', '執行個體', '執行個體類型', '啟動範本', 'Spot 請求', and 'Savings Plans'. The main area displays the '執行個體 (1/2) 資訊' section. It lists two executions: one completed (chi101-31-ec2-2) and one currently running (chi101-31-cli-demo). The running execution's status is '執行中' (Running), and its public IPv4 address is listed as 55.78.97.26. The right side of the screen contains various tabs and buttons for managing the execution, such as '啟動新執行個體', '連線', '執行個體狀態', '動作', '檢視詳細資訊', '管理執行個體狀態', '執行個體設定', '聯網', '安全性', '映像和範本', and '監控和故障診斷'.



The screenshot shows the AWS IAM console with the URL 'EC2 > 執行個體 > i-0a2a4b2a71d05f886 > 修改 IAM 角色'. The main area is titled '修改 IAM 角色' (Modify IAM Role). It shows a dropdown menu for selecting an IAM role, which is currently set to 'chi101-31-ec2-role-for-s3'. Below the dropdown, there's a note: '沒有 IAM 角色 選擇此選項以分開 IAM 角色'. At the bottom right, there are '取消' (Cancel) and '更新 IAM 角色' (Update IAM Role) buttons, with the latter being highlighted in orange.

```
命令提示字元
Microsoft Windows [版本 10.0.19044.2846]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。
C:\Users\Tibame_EX14>aws s3 ls
Unable to locate credentials. You can configure credentials by running "aws configure".
C:\Users\Tibame_EX14>aws configure
AWS Access Key ID [None]: AKIAT45F7DHHKINNYHRL
AWS Secret Access Key [None]: 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
Default region name [None]: ap-northeast-1c
Default output format [None]: text
C:\Users\Tibame_EX14>
```

建立桶子，類似資料夾的意思，之後可以在裡面建立資料



```
? MobaXterm Personal Edition v23.1 ?
(SSH client, X server and network tools)

▶ SSH session to ec2-user@35.78.97.26
? Direct SSH      : ✓
? SSH compression : ✓
? SSH-browser    : ✓
? X11-forwarding : ✘ (disabled or not supported by server)

▶ For more info, ctrl+click on help or visit our website.

,
~\_#
~~\_#####
~~ \###|
~~   \#/   ___ https://aws.amazon.com/linux/amazon-linux-2023
~~     V~'__->
~~~   /
~~-. /_/
~/m/'[ec2-user@ip-172-31-0-116 ~]$ aws s3 ls
2023-04-25 08:46:27 chi101-31
[ec2-user@ip-172-31-0-116 ~]$
```

可以使用「」

```
~`.-. /_`/  
/_m/'  
[ec2-user@ip-172-31-0-116 ~]$ aws s3 ls  
2023-04-25 08:46:27 chi101-31  
[ec2-user@ip-172-31-0-116 ~]$ ls  
[ec2-user@ip-172-31-0-116 ~]$ touch ./app.py  
[ec2-user@ip-172-31-0-116 ~]$ ls  
app.py  
[ec2-user@ip-172-31-0-116 ~]$
```

▼ 【專題使用情境】情境、安裝、使用

▼ 【情境】

因為 docker 每天都會關閉，所以需要重新連線，為了讓 docker 不會關閉，因此需要設定 CLI 腳本，讓 docker 不會關閉

▼ 【安裝】

安裝指定版本的 CLI，可以直接參考官方操作文件

https://docs.aws.amazon.com/zh_tw/AmazonECS/latest/developerguide/create-public-private-vpc.html

RDS

▼ 【建立前，請先看一下 IAM 權限說明】

<https://www.notion.so/AWS-c51ac45c666d488d85c77d0953701dc7?pvs=4#11dcb79cbf348037b350d4c696987a75>

▼ 【Amazon CloudWatch Logs】日誌

日誌記錄到Amazon CloudWatch Logs中，以便於後續的監控、故障排除和分析，以下為類型及差異說明：

▼ 稽核日誌（Audit Logs） 比較偏向管理者的整體資料庫監控

稽核日誌主要用於記錄系統和資源的活動和變更，以便後續審查和安全監控。這些日誌通常包含關鍵的事件，例如使用者登入和登出、資源的讀取和寫入操作、系統設定的更改等。

▼ 錯誤日誌（Error Logs） 異常發生時的紀錄

錯誤日誌用於記錄應用程序或系統中發生的錯誤、異常和例外情況。這些日誌通常包含錯誤訊息、堆棧跟蹤和其他相關資訊，有助於故障排除和問題分析。

▼ **一般日誌 (General Logs) 例行性紀錄**

一般日誌是應用程序或系統生成的常規訊息的記錄。這些日誌可以包含應用程序的運行狀態、操作的詳細訊息、事件通知等。一般日誌可以用於監控應用程序的運行情況、追蹤操作流程和記錄重要事件。

▼ **緩慢查詢日誌 (Slow Query Logs) 用於優化資料庫查詢速度**

緩慢查詢日誌用於記錄執行時間超過預設閾值的數據庫查詢。這些日誌對於識別性能瓶頸、調試和優化數據庫操作非常有用。它可以提供查詢的執行時間、索引使用情況、鎖定狀態等詳細資訊

▼ **【資料庫附加元件】**

建立資料庫後，會談出這樣的視窗，以下是視窗內容個別說明

▼ **ElastiCache 快取服務**

AWS提供的一種快取服務，用於提高應用程序的可擴展性和性能。它支援兩種主要的快取引擎，即Redis和Memcached。

快取服務可以減輕RDS資料庫的負載，提高讀取操作的效能，並降低應用程序的延遲

▼ **RDS Proxy 資料庫連接代理**

在 RDS 中創建一個代理層，用於管理與資料庫的連接和查詢，藉此處理應用程序和資料庫之間的連接，並提供一些額外的功能和優化，以提高資料庫的可用性、擴展性和效能

主要功能和好處如下：

▼ **資料庫連接池管理**

RDS Proxy負責管理與資料庫的連接池，自動處理連接的建立和回收，從而減少了應用程序的資源消耗和連接管理的複雜性。

▼ **高可用性和容錯**

RDS Proxy提供了內置的故障轉移和容錯功能。它可以自動檢測並重新路由連接到可用的資料庫實例，以確保應用程序的連接不會因為單一資料庫實例的故障而中斷。

▼ **效能優化**

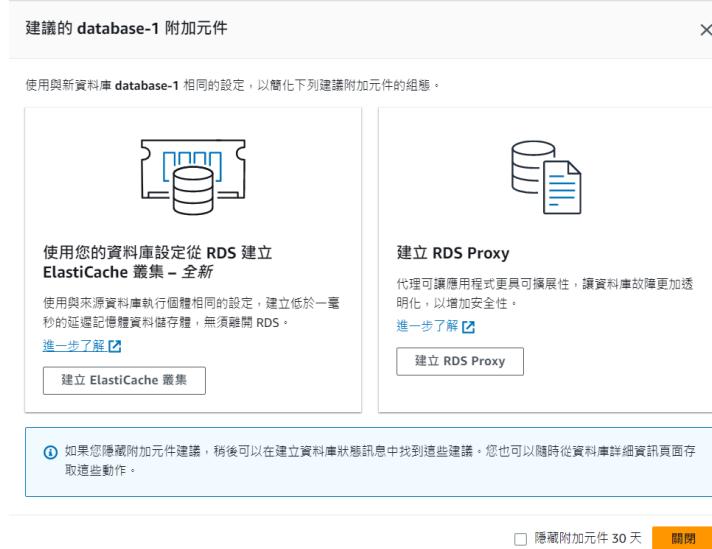
RDS Proxy支援連接池緩存和預熱功能，可以提高連接的重用率和資料庫查詢的執行效能。它還提供了一些優化機制，如快取查詢結果和分散式查詢分派，以提高整

體的查詢效能。

- 重用率：盡可能地重複使用現有的連接，而不是頻繁地創建新的連接。這可以通過連接池的機制來實現
- 連接池：是一個管理和維護資料庫連接的池子，它在應用程序初始化時預先創建一定數量的資料庫連接，並將它們保存在池中。當應用程序需要與資料庫交互時，它可以從連接池中獲取一個可用的連接，執行查詢或更新操作，然後將連接返回到池中以供重複使用。

▼ 安全性增強

RDS Proxy支援IAM（Identity and Access Management）身份驗證和授權，可以更精細地控制應用程序對資料庫的訪問權限。它還可以通過提供TLS（Transport Layer Security）加密和認證，增強資料在傳輸過程中的安全性。



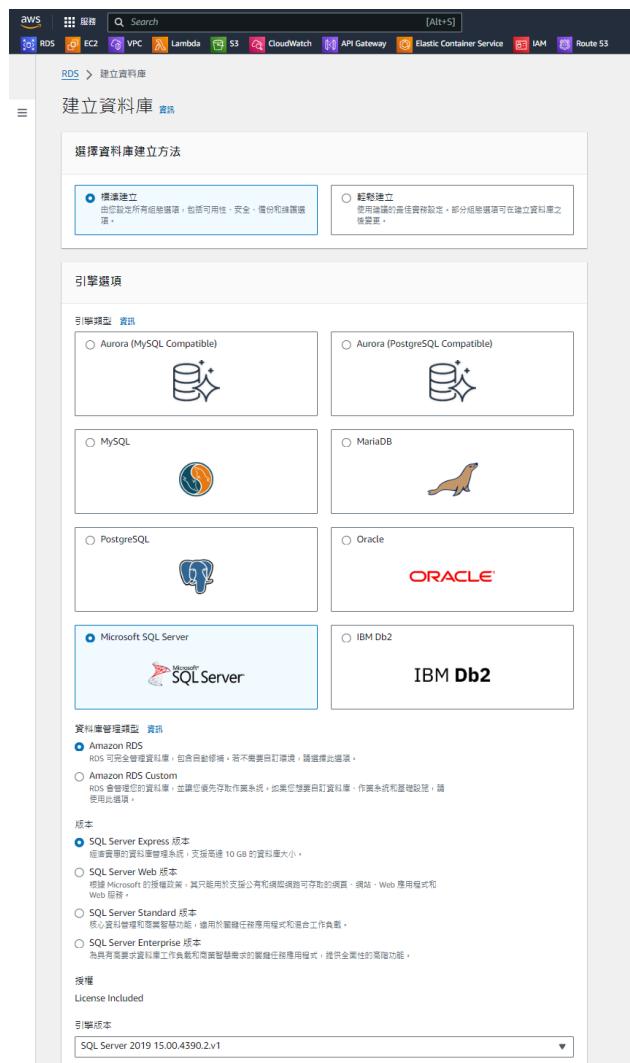
但在小型 side project 應該不需要，等到未來有效能需求再開啟即可

▼ 【實際建立操作】SQL server 為例(MySQL 同理)

請依據下圖進行設定

▼ 圖一

注意，我們使用免費版本 Express，引擎版本無須設定，使用建議即可



▼ 圖二

範本
選擇範例範本，以符合您的使用案例。

開發/測試
此執行個體適用於生產環境以外的開發用途。

免費方案
使用 RDS 免費方案來開發新的應用程式、測試現有應用程式，或透過 Amazon RDS 取得實作經驗。[資訊](#)

設定

資料庫執行個體識別符 [資訊](#)
為資料庫執行個體輸入一個名稱。該名稱必須為所有資料庫執行個體的唯一名稱，該執行個體的擁有者是在您在目前 AWS 區域中的 AWS 帳戶。

資料庫執行個體識別符並不會區分大小寫，但全以小寫的形式儲存（例如 "mydbinstance"）。限制：必須包含 1 到 60 個英數字元或連字號。第一個字元必須是字母，不可連續包含兩個連字號。不得以連字號結尾。

▼ 認證設定

主要使用者名稱 [資訊](#)
針對您資料庫執行個體的主要使用者，輸入登入 ID。

1 到 16 個英數字元。第一個字元必須是字母。

憑證管理
您可以使用 AWS Secrets Manager 或管理主要使用者憑證。

以 AWS Secrets Manager 管理 - 最安全
RDS 會為您產生密碼，並使用 AWS Secrets Manager 在整個生命週期中管理密碼。

自我管理
建立您自己的密碼，或讓 RDS 建立您管理的密碼。

自動產生密碼
Amazon RDS 可為您產生密碼，或者您可以指定自己的密碼。

主要密碼 [資訊](#)

Password strength Strong

最低限制：至少 8 個可列印的 ASCII 字元。不得包含下列任何符號：/ ' " @

確認主要密碼 [資訊](#)

執行個體組態

以下的資料庫執行個體組態選項受限於您在上面選取的引擎所支援的選項。

資料庫執行個體類別 [資訊](#)

▼ 隱藏篩選條件

包含上一代的類別

爆量類別 (包括 t 類別)

2 vCPUs 1 GiB RAM 網路：高達 2,085 Mbps

▼ 圖三

執行個體組態

以下的資料庫執行個體選項僅在您在上面選取的引擎所支援的選項。

資料庫執行個體類別: [資訊](#)

隱藏舊選項

古董上一代的類別

標量類別 (包括 t 類別)

db.t3.micro

2 vCPUs | 1 GiB RAM | 網路：高達 2,085 Mbps

儲存體

儲存體類型: [資訊](#)

佈建 IOPS SSD (io2) 儲存磁碟現在可供使用。

一般用途 SSD (gp2)

標準效能取決於磁碟區大小

配置儲存: [資訊](#)

20 GiB

配置的儲存值必須為 20 GiB 至 16,384 GiB

修改資料庫執行個體的儲存後，資料庫執行個體的狀態將轉為儲存優化狀態。儲存優化作業完成時，您的執行個體仍可使用。 [進一步了解](#)

▶ 儲存自動擴展

連線: [資訊](#)

選擇資源

選擇是否為此資料庫設定連接資源的連線。設定連線會自動變更連接設定，讓資料庫可以連線至此資料庫。

不要連接至 EC2 連算資源

請勿為此資料庫設定連接資源的連線。您可以稍後手動設定連接資源的連線。

連接至 EC2 連算資源

設定此資料庫的 EC2 連算資源的連線。

Virtual Private Cloud (VPC) 資訊

選擇 VPC。VPC 是為此資料庫執行個體的虛擬網路環境。

Default VPC (vpc-09af418ac2e426cfb)

3 子網段, 3 可用區域

列出與此資料庫子網路群組相關的 VPC。

① 在資料庫建立完成後，您無法變更其 VPC。

資料庫子網路群組: [資訊](#)

選擇資料庫子網路群組，指定資料庫執行個體可以在您選取的 VPC 中使用哪些子網路與 IP 范圍。

default-vpc-09af418ac2e426cfb

3 子網段, 3 可用區域

公開存取: [資訊](#)

是

RDS 會將公有 IP 地址推派給資料庫。Amazon EC2 執行個體和 VPC 外部的其他資源可以連接至您的資料庫。VPC 內的資源也可以連接至資料庫。選擇一或多個 VPC 安全群組，以指定哪些資源可以連接至資料庫。

否

RDS 不會將公有 IP 地址推派給資料庫。只有 Amazon EC2 執行個體和 VPC 內的其他資源才能連接至您的資料庫。選擇一或多個 VPC 安全群組，以指定哪些資源可以連接至資料庫。

VPC 安全群組 (防火牆): [資訊](#)

選擇一或多個 VPC 安全群組，以允許存取您的資料庫。請確保安全群組規則允許適當的傳入流量。

選擇現有

選擇現有的 VPC 安全群組

新建

建立新的 VPC 安全群組

現有的 VPC 安全群組:

選擇一或多個選項

default

可用區域: [資訊](#)

無偏好設定

RDS Proxy:

RDS Proxy 是全監管、高可用性的資料庫代理，可提升應用程式可擴展性、復原能力和安全性。

建立 RDS Proxy: [資訊](#)

RDS 會自動為代理建立 IAM 角色和 Secrets Manager 密碼。RDS Proxy 會產生額外成本。如需詳細資訊，請參閱 [Amazon RDS Proxy 定義](#)。

憑證認證機構 - 連用: [資訊](#)

使用同服務憑證可透過驗證是否連線至 Amazon 資料庫提供多一層的安全防護。方法是檢查在您佈建的所有資料庫上自動安裝的同服務憑證。

rds-ca-1sa2048-g1 (預設)

到期日: May 26, 2061

如果您未選取憑證認證機構，則 RDS 會為您選擇憑證認證機構。

▶ 其他組態

▼ 圖四

Microsoft SQL Server Windows 身份驗證

選取一個目錄，以便其中允許授權網域使用者透過此使用 Windows 身份驗證的 SQL Server 執行個體進行身分驗證。

啟用 Microsoft SQL Server Windows 身份驗證

標籤 - 選用
標籤包含區分大小寫的索引鍵/值對。

沒有與資源相關聯的標籤。

新增標籤
您最多可以再新增 50 個標籤。

監控

績效詳情 資訊
 開啟 Performance Insights

保留期間 資訊
7 天 (免費方案)

AWS KMS 金鑰 資訊
(default) aws/rds

帳戶
911167896067

KMS 金鑰 ID
e9ad170c-6525-402a-8860-be6854a286e4

⚠ 在啟用績效詳情之後，您將無法變更 KMS 金鑰。

▶ 其他組態
資料庫選項, 備份 已開啟, 恢復 已關閉, 緊急, CloudWatch Logs, 刪除保護 已關閉.

預估每月成本

您可享有 Amazon RDS 免費方案 12 個月，在每個行事曆月，免費方案均允許您免費使用下列 Amazon RDS 資源：

- 單一可用區 db.t2.micro、db.t3.micro 或 db.t4g.micro 執行個體中 750 小時的 Amazon RDS。
- 20 GB 的一般用途儲存 (SSD)。
- 20 GB，用於自動化備份儲存和任何使用者啟動的資料庫快照。

進一步了解 AWS 免費方案。 [\[\]](#)

當免費用量逾期或應用程式的用量超過免費用量方案時，您只需依據標準的按用量付費服務費率付費，如 [Amazon RDS 定價頁面](#)。 [\[\]](#) 中所述

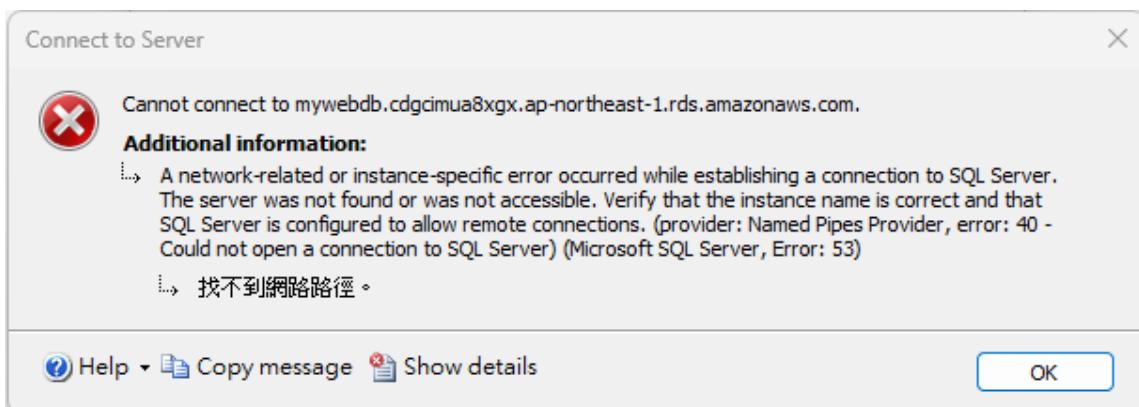
💡 您有責任確保您對搭配 AWS 服務使用的任何第三方產品或服務擁有所有必要的權利。

取消 **建立資料庫**

[CloudShell](#) [意見回饋](#)

▼ 【連線錯誤排除】SQL server

當我的 RDS 建立完成後，使用 SSMS 進行連線，但出現下方錯誤訊息，大多狀況是 Security Groups (防火牆相關) 問題，因此需要開放公開 IP



修改 DB 的設定



設定為公開存取

連線

資料庫子網路群組
default-vpc-09af418ac2e426cfb

安全群組
要與此資料庫執行個體關聯的資料庫安全群組清單。
選擇安全群組
▼
default X

憑證認證機構 資訊
使用伺服器憑證可透過驗證是否連線至 Amazon 資料庫來提供多一層的安全防護。方法是檢查在您佈建的所有資料庫上自動安裝的伺服器憑證。
rds-ca-rsa2048-g1 (預設)
到期日: May 26, 2061

▼ 其他組態

公開存取
 可公開存取
RDS 會將公有 IP 地址指派給資料庫。Amazon EC2 執行個體和 VPC 外部的其他資源可以連接至您的資料庫。VPC 內的資源也可以連接至資料庫。選擇一或多個 VPC 安全群組，以指定哪些資源可以連接至資料庫。
 不可公開存取
未將 IP 地址指派給資料庫執行個體。EC2 執行個體和 VPC 外部的裝置無法建立連線。

資料庫連接埠
指定資料庫執行個體將用於應用程式連線的 TCP/IP 連接埠。應用程式連線字串必須指定連接埠號。資料庫安全群組和您的防火牆必須允許該連接埠的連線。[進一步了解](#)

1433

等待狀態欄未顯示修改完成後，就可以再次測試連線

如果還是有錯誤，通常會是 VPC 安全群組的防火牆，要如同下方截圖一樣的 IP 才是正確，如果顯示 VPC 的英文內容，建議可以重新建立資料庫

Amazon RDS

代理別符

無代理
您沒有任何代理。

安全群組規則 (2)

安全群組	類型	規則
mywebdb (sg-00baa429f8556a14d)	CIDR/IP - Inbound	106.105.64.26/32
mywebdb (sg-00baa429f8556a14d)	CIDR/IP - Outbound	0.0.0.0/0

複寫 (1)

資料庫別符	角色	區域與可用區域 (AZ)	複寫來源	複寫狀態	延遲
mvwebdb	執行備註	an-northeast-1c	-	-	-

因為上圖顯示的 Inbound 規則，限制 106.105..../32 這個 IP 才可以進入，算是一個白名單的概念，因此如果需要指定特定連線，就會需要設定新的白名單，例如下方，我需要兩個 IP 可以進入

安全群組規則 (3)

安全群組	類型	規則
myWebDBMySQL (sg-03bfe54675487e121)	CIDR/IP - Inbound	106.105.64.26/32
myWebDBMySQL (sg-03bfe54675487e121)	CIDR/IP - Inbound	211.23.210.148/32
myWebDBMySQL (sg-03bfe54675487e121)	CIDR/IP - Outbound	0.0.0.0/0

補充，如果想要限制指定的後端 server 才可以取得數據可以設定 Outbound，但需求不常見

▼ 【快照 Snapshot 轉換成資料庫】SQL server

▼ Step1、先建立一個快照

Aurora and RDS > 資料庫

資料庫 (1)

資料庫別符

aspnex

動作

- 暫時停止
- 重新啟動
- 刪除
- 立即修補
- 在下一個快照修補
- 設定 EC2 連線
- 設定 Lambda 連線
- 從 EC2 資料庫複製資料 - 全新
- 建立 Aurora 僅供讀取副本
- 將資料庫移至另一個可用區
- 建立快照 **(highlighted)**
- 還原至當前點
- 還原快照
- 啟動資料庫活動串流
- 建立 ElastiCache 繪集

Aurora and RDS > 快照

Aurora and RDS

- 儀表板
- 資料庫
- 查詢編輯器
- 績效詳情
- 快照**
- 匯出至 Amazon S3
- 自動備份
- 預留執行個體
- 代理

快照

手動 系統 與我共享 公開 備份服務 匯出至 Amazon S3

手動快照 (12)

快照名稱	引擎版本	資料庫
apnex	15.00.4073.23.v1	aspnex
aspnex-dev	15.00.4390.2.v1	aspnex
aspnex-test-env-final-snapshot	15.00.4073.23.v1	aspnex
aspxnex-test01snapshot	15.00.4073.23.v1	aspnex

▼ Step2、設定新資料庫細項，如安全性之類的，內容都可以「參考原本資料庫」的設定

Aurora and RDS > 快照

快照

手動 系統 與我共享 公開 備份服務 匯出至 Amazon S3

手動快照 (12)

快照名稱	引擎版本	資料庫執行個體或叢集	快照建立時間	資料庫執行個體建
apnex	15.00.4073.23.v1	aspnex	May 11, 2022, 15:07 (UTC+08:00)	October 28, 2021
aspnex-dev	15.00.4390.2.v1	aspnex	July 17, 2025, 09:56 (UTC+08:00)	October 28, 2021
aspnex-test-env-final-snapshot	15.00.4073.23.v1	aspnex-test-env	October 05, 2022, 17:10 (UTC+08:00)	January 12, 2022, 12:03 (UTC+08:00)
aspnex-test01-snapshot	15.00.4073.23.v1	aspnex-test01	October 18, 2022, 10:53 (UTC+08:00)	October 05, 2022, 17:16 (UTC+08:00)

操作：還原快照

還原快照

資料庫執行個體設定

DB 引擎：SQL Server Express 版本

授權模式：License Included

設定

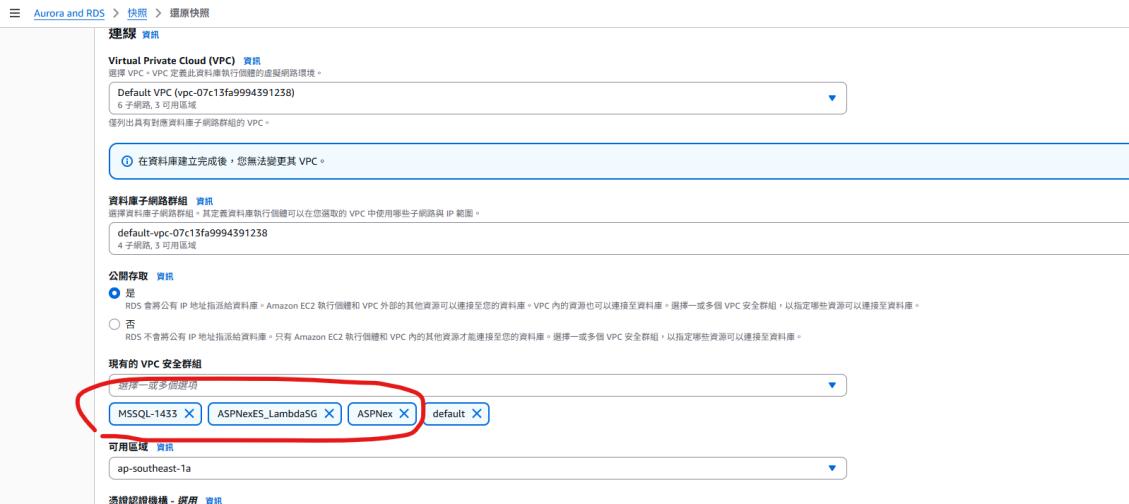
資料庫快照 ID：aspnex-dev

資料庫執行個體識別符：aspnex-dev

執行個體組態

資料庫執行個體類別：資料庫

捲儲歸還條件：包含上一代的類別



▼ Step3、建立完成後，就會有新的資料庫可以使用，請記得要進行連線測試



▼ 【RDS 資料庫備份】

由於 AWS 的 RDS 沒有硬碟儲存的機制，因此無法像是本地端的資料庫備份的方式一樣處理資料

Boto3

▼ 【介紹】

AWS Boto3 是 AWS (Amazon Web Services) 提供的官方 Python 軟體開發套件，用於與 AWS 服務進行交互。它為 Python 開發者提供了一個方便的界面，可以通過編寫 Python 程式碼來使用和管理 AWS 資源。

Boto3 提供了一個豐富的 API，使開發者能夠通過 Python 直接與 AWS 服務進行互動，例如創建、管理和配置 EC2 實例、S3 存儲桶、RDS 資料庫等等。開發者可以使用 Boto3 來執行各種操作，例如創建、查詢、更新和刪除 AWS 資源，並管理 AWS 服務的各種配置選項。

Boto3 的特點包括：

1. 易於使用：Boto3 提供了簡潔且直觀的 API，使開發者能夠快速上手並開始使用 AWS 服務。
2. 全面的服務支持：Boto3 支持 AWS 提供的幾乎所有服務，包括計算、存儲、資料庫、網絡、人工智慧、安全等等。
3. 強大的功能：Boto3 提供了豐富的功能，例如處理 AWS 資源的 CRUD（創建、讀取、更新、刪除）操作、執行 AWS 服務的 API 請求、處理身份驗證和授權等。
4. 高度可擴展：Boto3 允許開發者構建自定義的功能和工具，以滿足特定的需求和場景。

使用 Boto3，開發者可以在 Python 程式中輕鬆集成和自動化 AWS 服務的操作，並根據需要進行擴展和自定義。這使得在 AWS 環境中進行自動化、腳本化和開發工作變得更加容易和便捷。

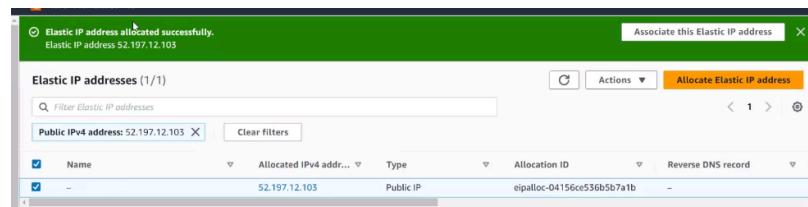
ECS

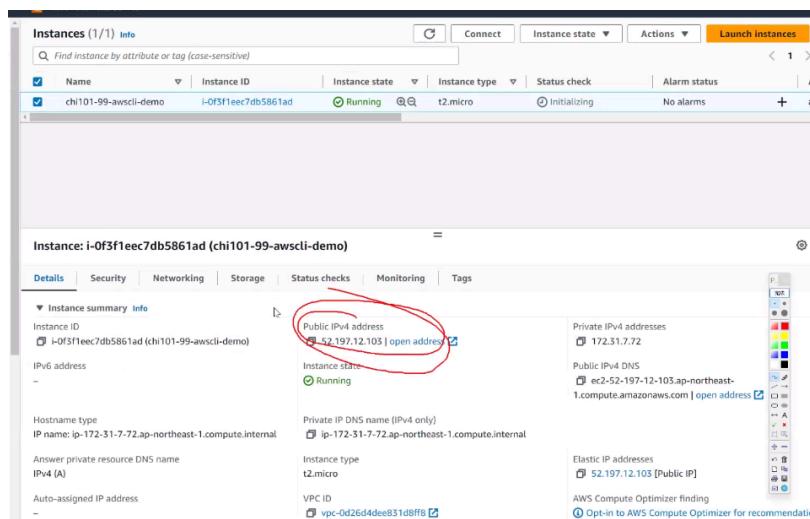
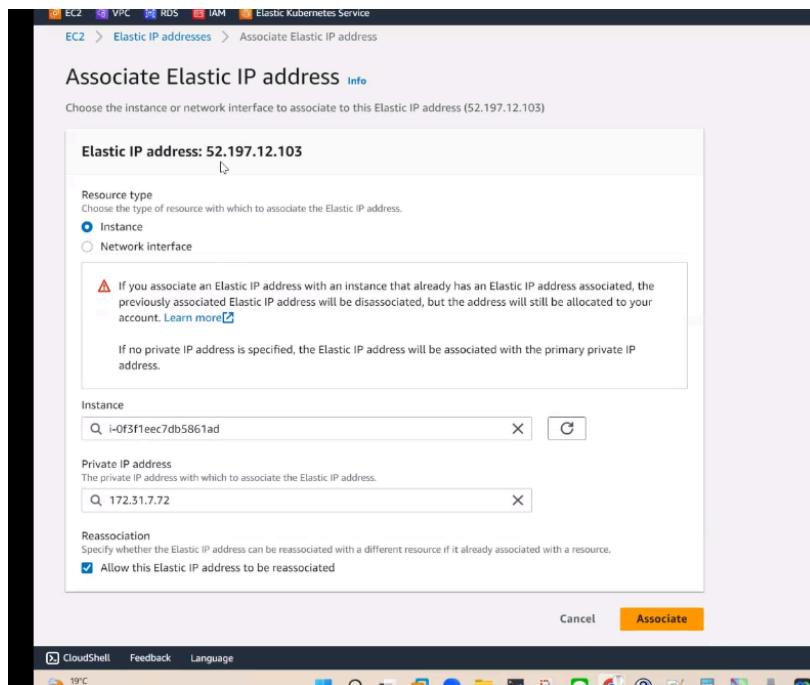
- ▼ 【介紹】
- ▼ 【安裝 CLI】
- ▼ 【使用 CLI】

補充

- ▼ 【靜態 IP 建置】

VU





▼ 【其他課程補充】

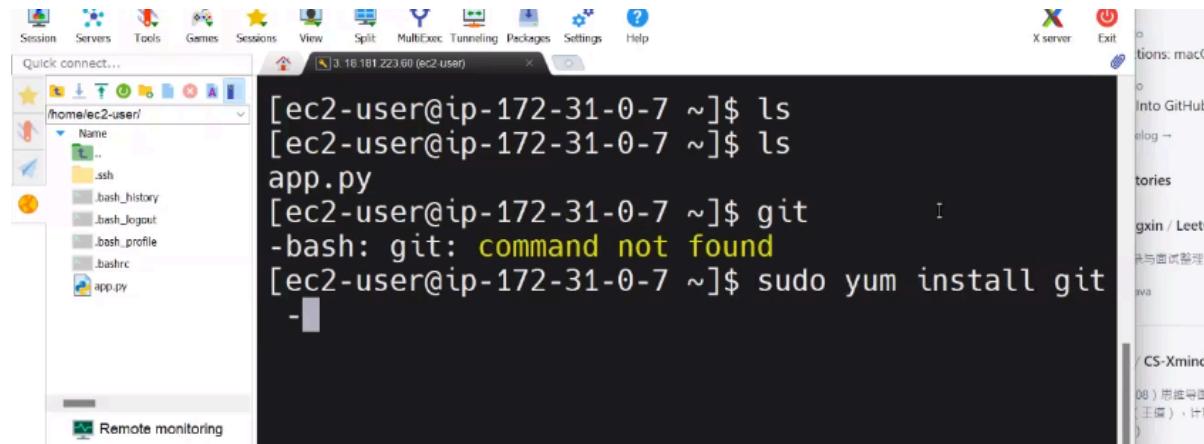
serveo 結合 docker 可以用來製作一個 HTPS 憑證，可以做為測試用的網站，但有安全的問題，所以只能自己做不是和公司

只要擁有 Agent 就可以控制電腦(大量，如總師令)，應用如木馬程式，獲取 cookie session，google 就會認定你是帳號擁有者，就可以獲取帳號權限

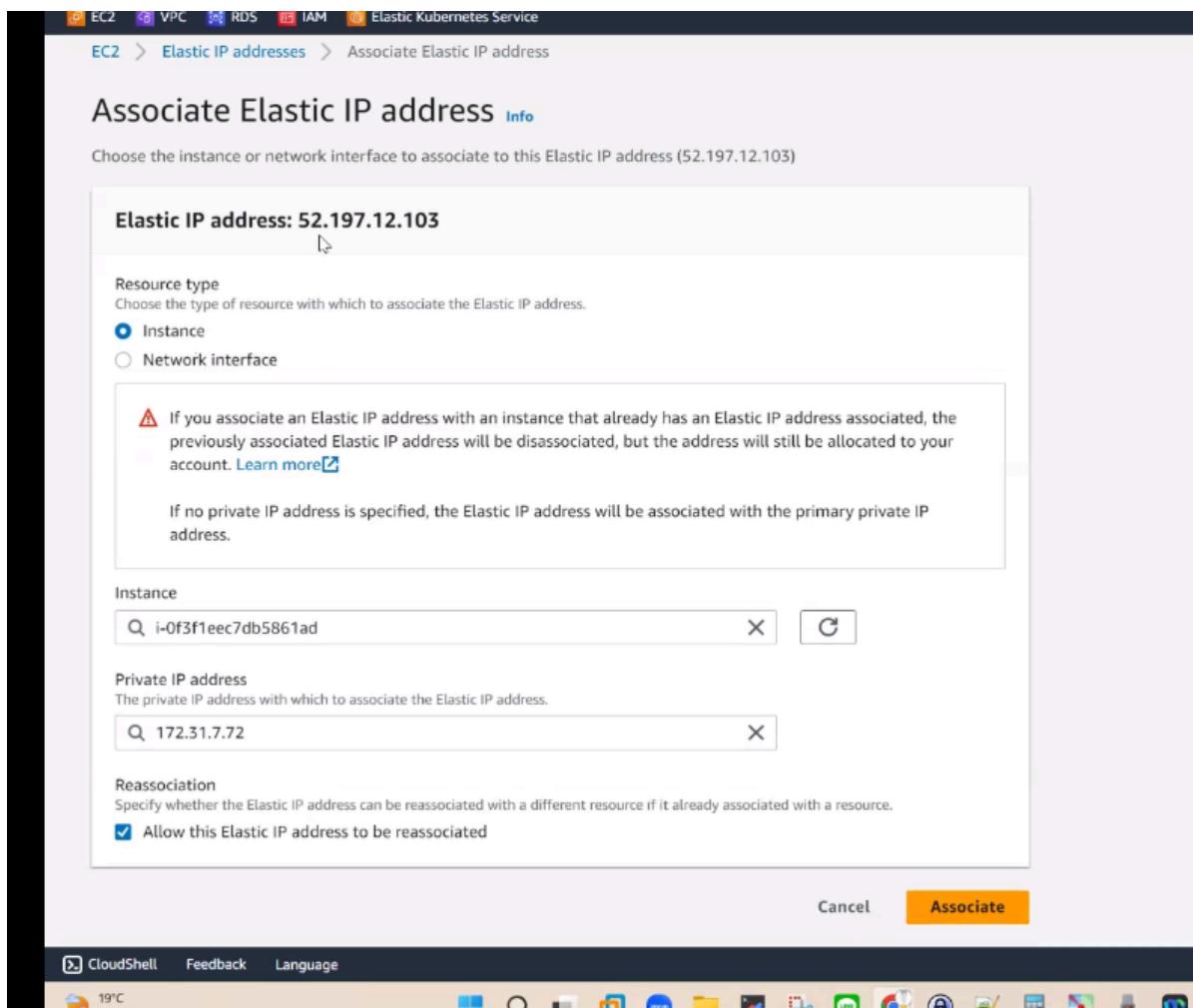
Wireshark 網路封包剖析器，可以解析、監聽 server 所有流量

MVC 系統架構，需要了解(使用者介面>前端工程師框架、使用者介面；商業邏輯 > function define；資料庫 > DBA)

Day_06 - MVC 與三層架構 - iT 邦幫忙::一起幫忙解決難題，拯救 IT 人的一天
(ithome.com.tw)



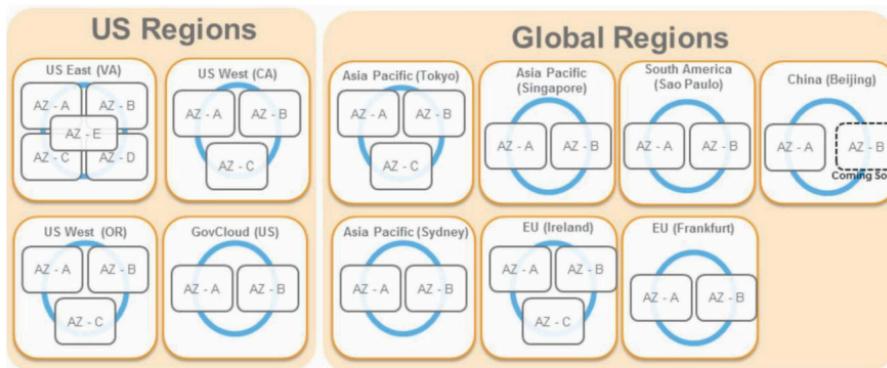
lightsail (虛擬私有伺服器(VPS)) 簡單快速建立網站，可以在上面建立一個個人伺服器來使用，也不容易因為共用伺服器而被攻擊



▼ 【A-Z and Region】

兩個以上的 A-Z(Available Zone)才可以成為 region

A-Z 數據機房之間距離須超過 100 公里



- ▼ 雲端原生 <https://www.oracle.com/tw/cloud/cloud-native/what-is-cloud-native/>
- ▼ CNCF Cloud Native Computing Foundation (cncf.io)
- ▼ 全受管的關聯式資料庫- Amazon RDS - AWS
[全受管的關聯式資料庫 – Amazon RDS – Amazon Web Services](#)

▼ AWS Lambda ?

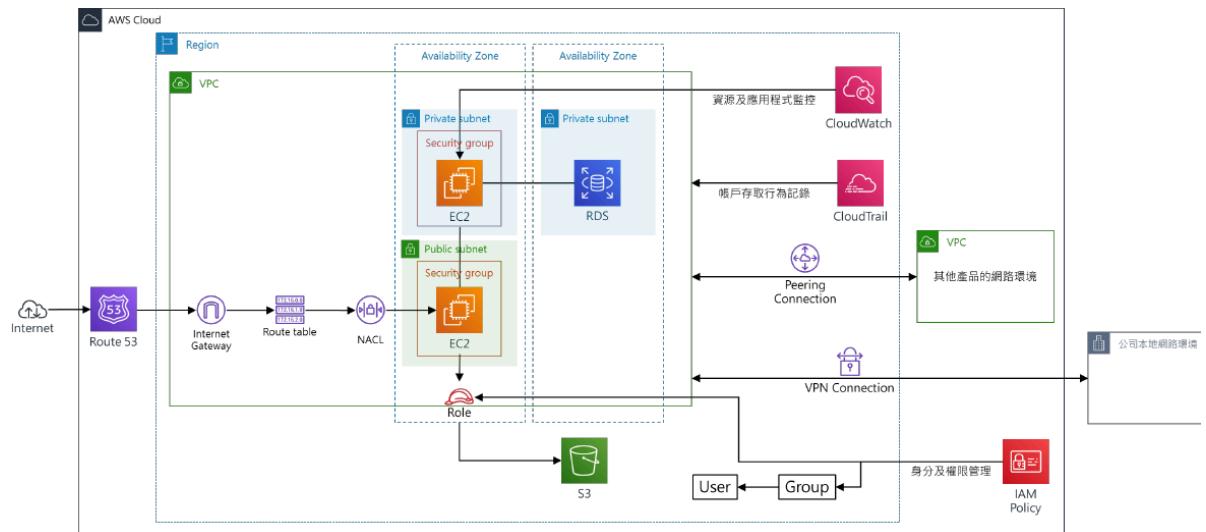
<https://ap-southeast-2.console.aws.amazon.com/lambda/home?region=ap-southeast-2#/begin>

▼ AWS tail 是什麼？日誌信息

AWS tail 是 AWS CLI 中的一個命令，可以用來在命令列介面中實時追蹤 AWS 服務中產生的日誌。該命令可以從指定的 CloudWatch Logs 日誌組中獲取和顯示最新的日誌事件，顯示方式類似於 Linux 中的 tail 命令。使用 AWS tail 可以幫助開發人員和系統管理員快速獲取和監控 AWS 中的日誌信息，以便進行故障排除、調試和優化等工作。

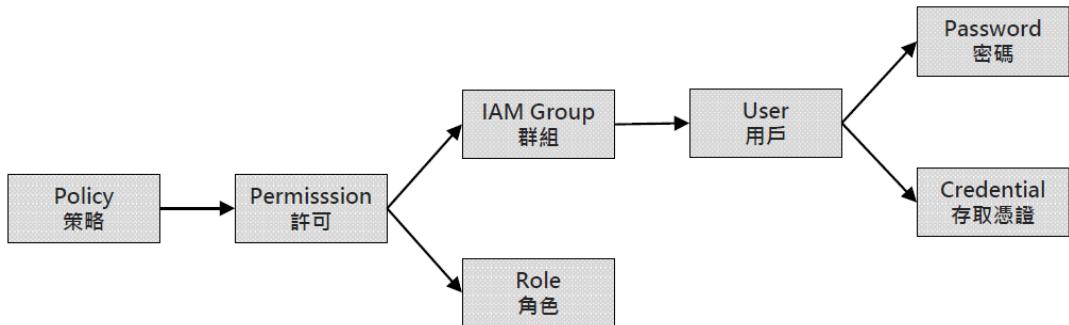
可以到 AWS 的 CloudTrail 中建立

▼ AWS 雲端架構



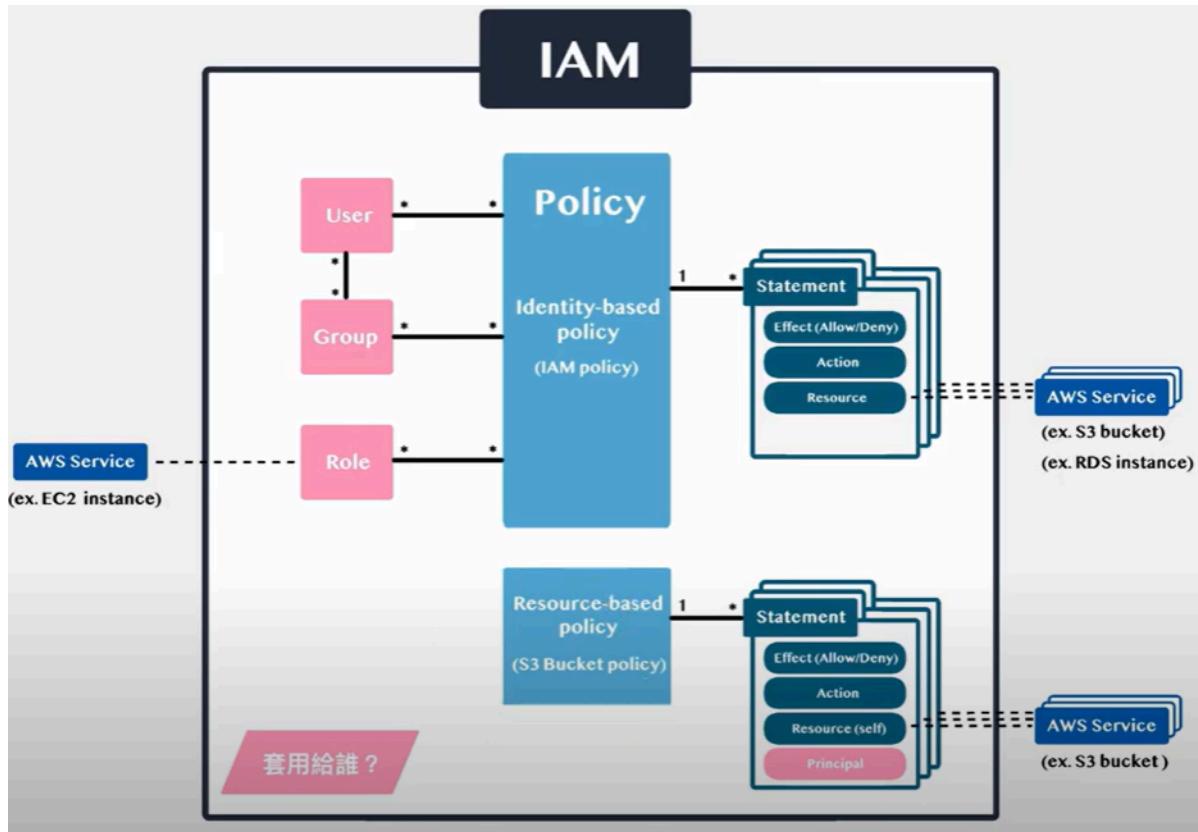
AWS Identity and Access Management (IAM)

- 控制 User 對 AWS 服務和資源的存取權限。
- 可以使用 IAM 建立和管理 AWS 使用者和群組。
- 使用各種許可來允許和拒絕他們存取 AWS 資源。



- Users: End Users 像是人、員工、公司等。

- Groups: Users 的集合，在 group 的每個user 都會繼承 group 的 permissions
- Policies: 由 document 創建而成，又稱為 Policy documents，JSON 格式，Documents 會給 permissions 紿 User/Group/Role。
- Roles: 創建 roles 之後指派給 AWS Resources，不需要 password 或 access key，Roles 比 User 好管理。



Role 主要目的是給不是 User,Group(需要登入帳號)來使用IAM的服務。

EX: 創建 EC2 instance 時需要填入Role欄位，透過此方式就可以把AWS Service 連結到 IAM中，可以使用其他AWS Service(S3 bucket, RDS...)。

Resource-based policy :偏向於某個Service下的規範。

EX: S3 bucket中設定Principal 來決定那個User,Group可使用這個服務。

套用給誰? Identity-based policy與Resource-based policy 差異:

Identity-based policy :是Policy level下的Statement,User, Group, Role一次套用(類似人的角度)。

Resource-based policy: 是Statement level下的Principal, 每次都需要撰寫，跨Service則無法使用。(類似資源的角度)。

AWS Key Management Service (KMS)

- **Regional**, 建立、管理加密金鑰
- 管理 **customer master keys (CMKs)**
- **FIPS 140-2 Level 2**
- 種類：Customer Managed、AWS Managed CMK、AWS Owned CMK

Type	Can View	Can Manage	Dedicated to My Account
Customer Managed	Yes	Yes	Yes
AWS Managed CMK	Yes	No	Yes
AWS Owned CMK	No	Yes	No

- 必須徹底理解每個種類，常考該情境應使用何種 Encryption 搭配何種 Key Management Service

AWS CloudHSM

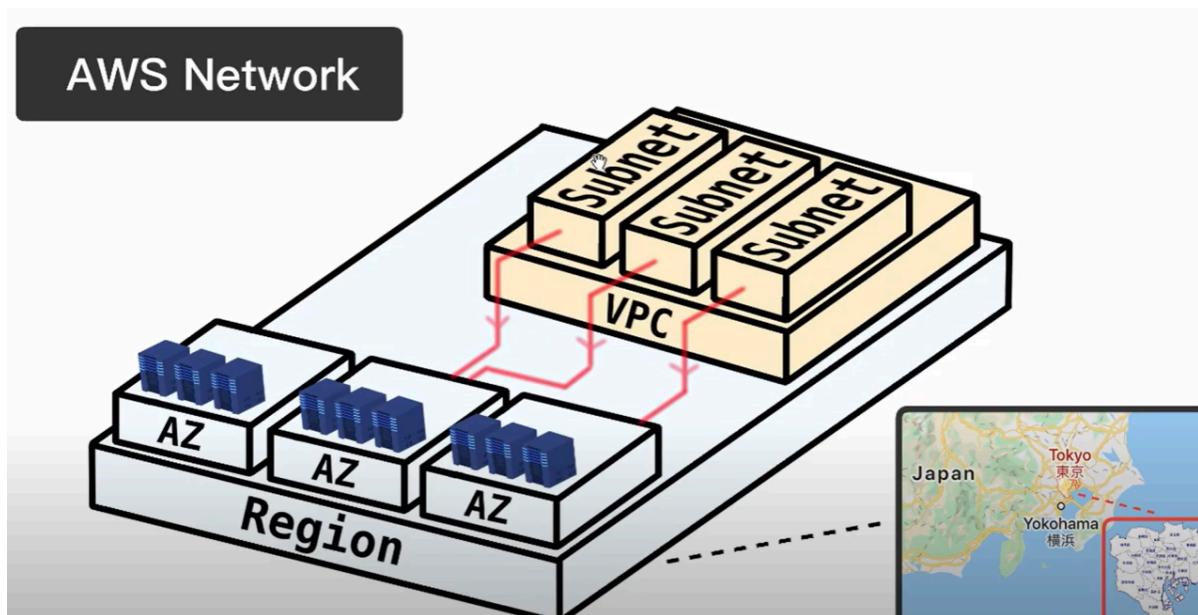
- 以雲端為基礎的硬體安全模組 (Hardware Security Module 簡稱 HSM)，在 AWS 雲端產生和使用自己的加密金鑰。
- **FIPS 140-2 Level 3**補充：Level 2 是 KMS
- 產業標準 API，而非 AWS API

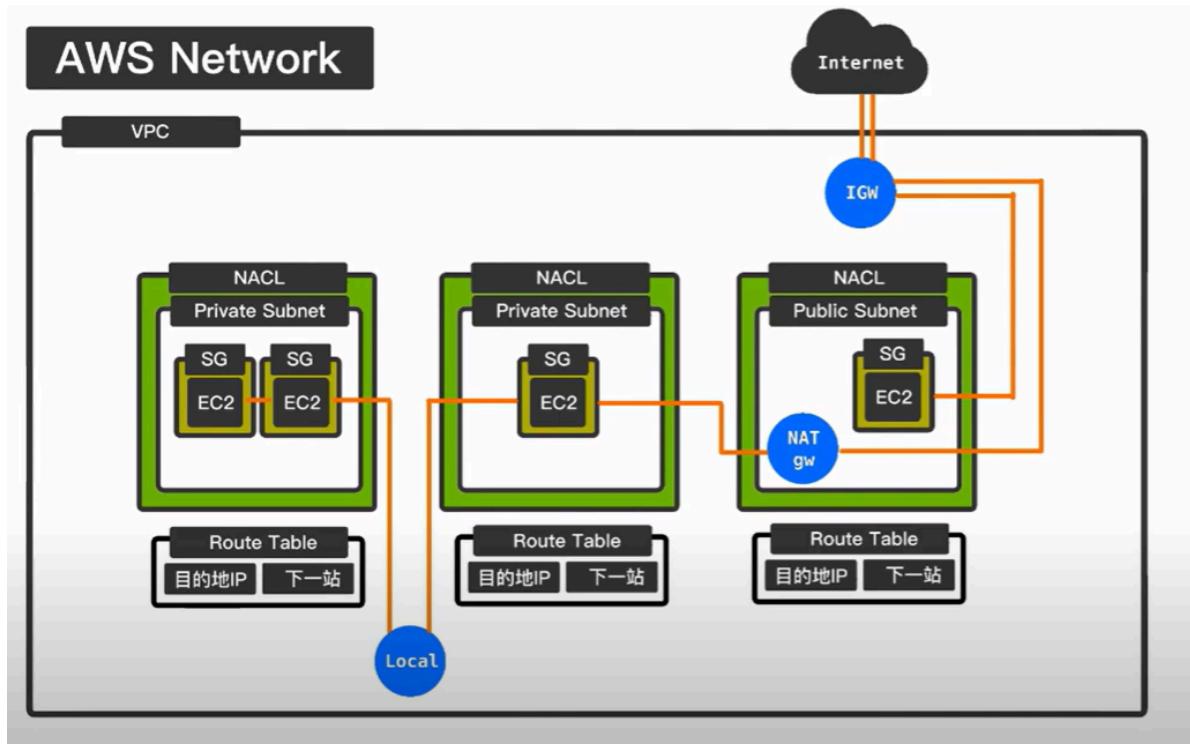


AWS VPC (Virtual Private Cloud)

提供的虛擬網路環境，可在此進行網路與IP配置，業界多用於產品或專案的資訊架構。

- Virtual Private Cloud，想像 VPC 是在雲端的虛擬 data center
- 包含 **Subnets、Security Groups、Route Tables、Internet Gateways (或 Virtual Private Gateways)、Access Control Lists**
- 1 Subnet = 1 AZ不能有一個 Subnet 延展至多個 AZs，但可以一個 AZ 有多個 Subnets
- Security Groups - Stateful ; Network Access Control Lists - Stateless.
- NO TRANSITIVE PEERING





Network Access Control List(NACL) : 給Subnet使用的靜態防火牆，為Subnet過濾所有進出的網路封包。

Security Group(SG): 類似EC2的防火牆概念。

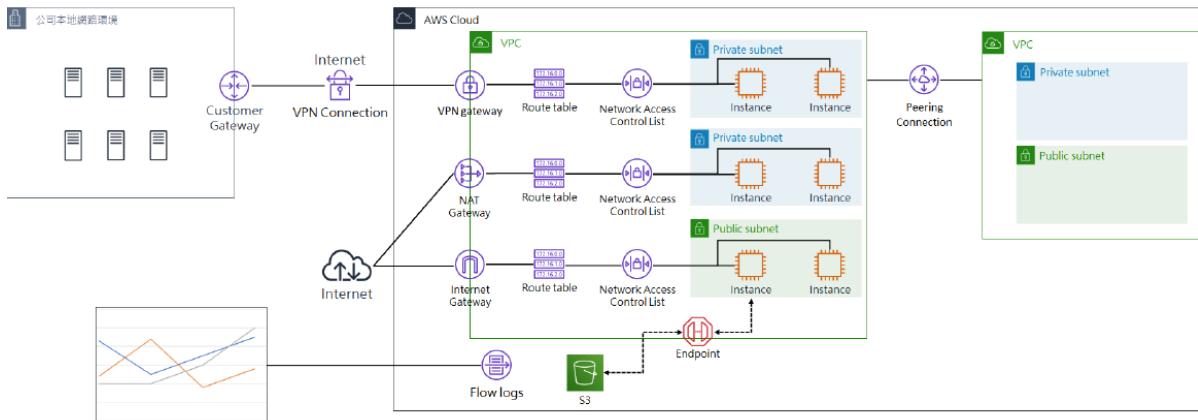
Internet Gateway(IGW): 連通網際網路的閘道口，VPC內的機器必須從此閘道傳輸對外流量，連線至外部網際網路。

Subnet: VPC之下的虛擬網路環境，在雲服務中開啟的虛擬主機，都會放在Subnet。

Route Table: 內外部網路封包的流量導引，網路封包憑藉此表，導流進出VPC環境。

NAT Gateway :外部網路直接連上VPC私網EC2的連線模式，不用透過公有EC2。

VPC 總體架構



EC2(Elastic Computing Cloud)



EC2 Storage types

- [Amazon Elastic Block Store\(EBS\)](#)：提供 block-level storage，分成 Magnetic, General purpose SSD, provisioned IOPS SSD。
- [Instance Storage](#)：提供暫時的 storage，EC2 instance shut-down 資料就會消失。

Elastic File System (EFS)

- File storage 服務，屬於 high durable storage，可以跟 EC2 intance 共享檔案資料 (NFS Protocol)，資料可以儲存 across multi-AZ (單一 region)，提供 [POSIX](#)。

EC2 建立流程



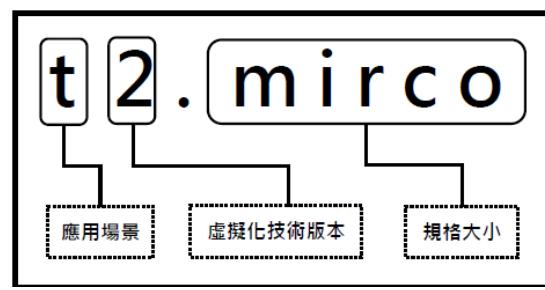
AMI Marketplace: 坊間企業會將自己製作的AMI，放在AMI Marketplace平台，讓AWS的使用者自由選購，拿來開啟機器。

Amazon Machine Image(AMI): EC2 用來快速開啟虛擬主機的系統映像檔。

note: 在雲端租用虛擬主機，需要快速建置環境，業界常用做法，預先製作一個模板，用這模板快速製作主機。

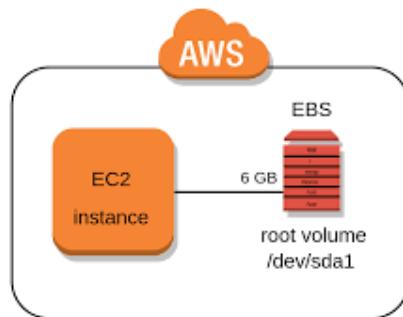
Instance Type: 按不同的應用場景去選擇需要的機器規格。

Instance Name	vCPUs	Memory	Local Storage	EBS-Optimized Bandwidth	Network Bandwidth
z1d.large	2	16 GiB	1 x 75 GB NVMe SSD	Up to 2.333 Gbps	Up to 10 Gbps
z1d.xlarge	4	32 GiB	1 x 150 GB NVMe SSD	Up to 2.333 Gbps	Up to 10 Gbps
z1d.2xlarge	8	64 GiB	1 x 300 GB NVMe SSD	2.333 Gbps	Up to 10 Gbps
z1d.3xlarge	12	96 GiB	1 x 450 GB NVMe SSD	3.5 Gbps	Up to 10 Gbps
z1d.6xlarge	24	192 GiB	1 x 900 GB NVMe SSD	7.0 Gbps	10 Gbps
z1d.12xlarge	48	384 GiB	2 x 900 GB NVMe SSD	14.0 Gbps	25 Gbps
r5.large	2	16 GiB		Up to 3.5 Gbps	Up to 10 Gbps
r5.xlarge	4	32 GiB		Up to 3.5 Gbps	Up to 10 Gbps
r5.2xlarge	8	64 GiB		Up to 3.5 Gbps	Up to 10 Gbps
r5.4xlarge	16	128 GiB		3.5 Gbps	Up to 10 Gbps
r5.12xlarge	48	384 GiB		7.0 Gbps	10 Gbps
r5.24xlarge	96	768 GiB		14.0 Gbps	25 Gbps



User data: 希望在機器運行後，有些軟體或設定就已經自動載入完成。EC2在啟用新虛擬主機的時候，會自動執行User data裡面的指令，來完成虛擬主機內的環境設定。內容為shell script，使用時，預設身分即為root，故指令前頭不需要加sudo，若部屬環境不如預想時，可從/var/cloud-init.log看端倪。

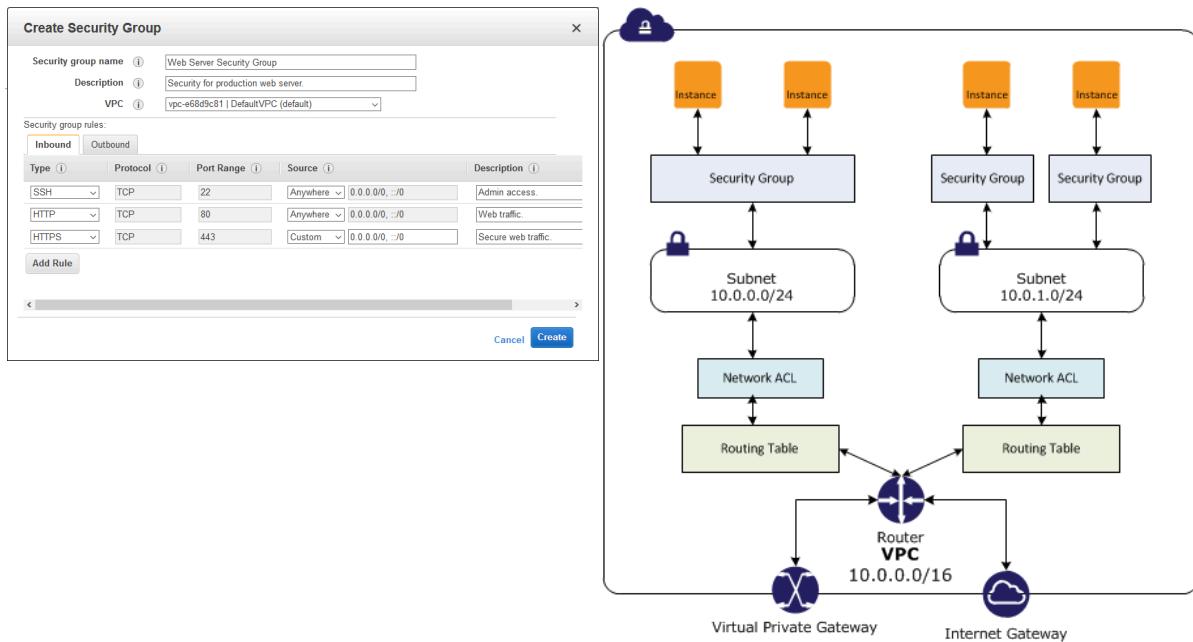
Elastic Block Service(EBS): 電腦需要硬碟來做資料儲存。EC2在啟動虛擬主機(Instance)時，也許要協派一個硬碟給Instance作為作業系統硬碟。可在創建時，設定硬碟空間。



Tag: AWS開放我方自定義資源，作為後續資源管理與權限設定的用途。這個自定義的方式，我們稱為Tag。

Manage Tags						
	Filter: <input type="text"/> Search Keys <input type="button" value="X"/>	<input type="text"/> Search Values <input type="button" value="X"/>				
	Tag Key	Tag Value	Total	Instances	AMIs	Volumes
Manage Tag	Name	DNS Server	1	1	0	0
Manage Tag	Owner	TeamB	2	0	0	2
Manage Tag	Owner	TeamA	2	0	0	2
Manage Tag	Purpose	Project2	1	0	0	1
Manage Tag	Purpose	Logs	1	0	0	1
Manage Tag	Purpose	Network Management	1	1	0	0
Manage Tag	Purpose	Project1	2	0	0	2

Security Group:在雲端上，每一台虛擬主機都會有網路流量的進出，流量的安全控管需要特別注意。EC2為虛擬主機(Instance) 設置了Security Group 防火牆，用來管理虛擬主機的網路流量。



Key Pair:遠端虛擬主機，預設不開放帳密登入，通通使用金鑰做遠端登入。**Key Pair**俗稱的金鑰，我們可在這裡選擇用什麼金鑰登入遠端虛擬主機。

This dialog box provides instructions for creating a key pair. It states that a key pair consists of a public key stored by AWS and a private key file stored by the user. For Windows AMIs, the private key file is required to obtain the password used to log into the instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

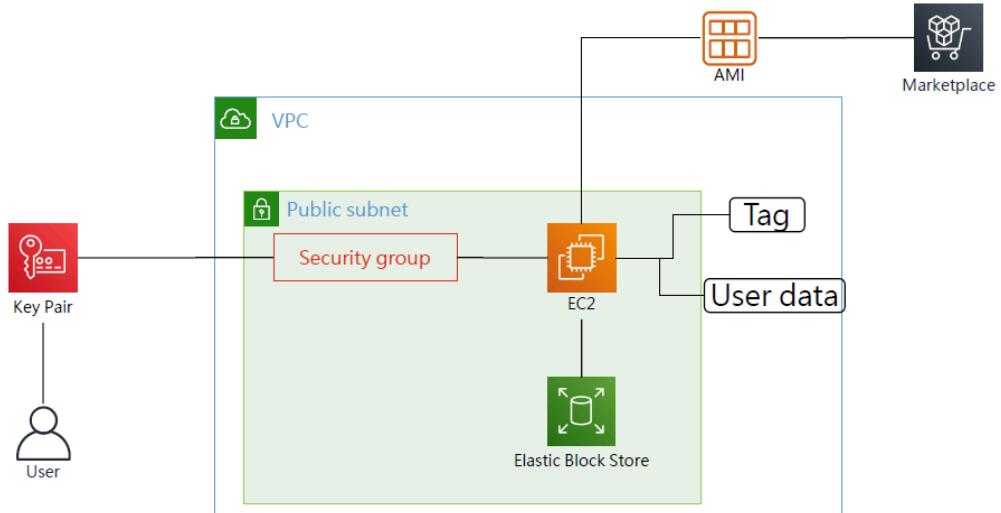
A note below says: "Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI."

A dropdown menu titled 'Choose an existing key pair' is shown, with 'mykey' selected. A red box highlights this dropdown and the selected key name.

At the bottom, there is an acknowledgement checkbox: "I acknowledge that I have access to the selected private key file (mykey.pem), and that without this file, I won't be able to log into my instance." This checkbox is checked.

Buttons at the bottom are 'Cancel' and 'Launch Instances'.

EC2 的總體架構



Amazon Simple Storage Service (S3)

安全、耐久、可高度擴展的雲端儲存。

Amazon S3是易於使用的物件儲存，且具備一個簡單的Web 服務界面，可讓您在Web 上隨處存放和擷取任意數量的資料。

Bucket: 存儲桶，名字全球唯一。

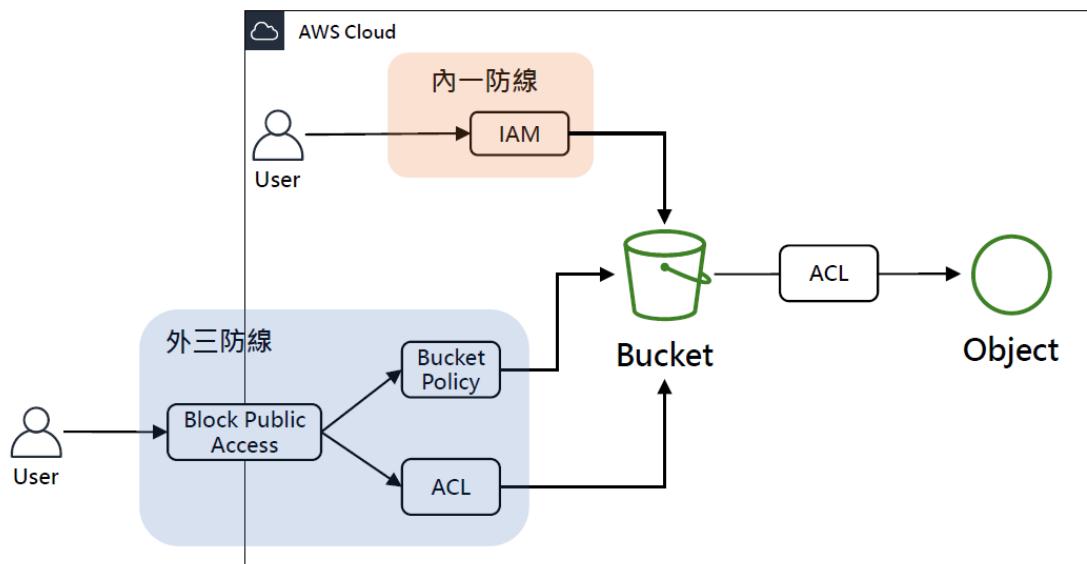
Object :即為放在bucket內的檔案每個帳號有軟性限制(Soft limit)，只能有一百個bucket。每個bucket可存儲的物件數量無上限，單個物件大小最大可支援到5 TB。

儲存級別 (Storage-Level)

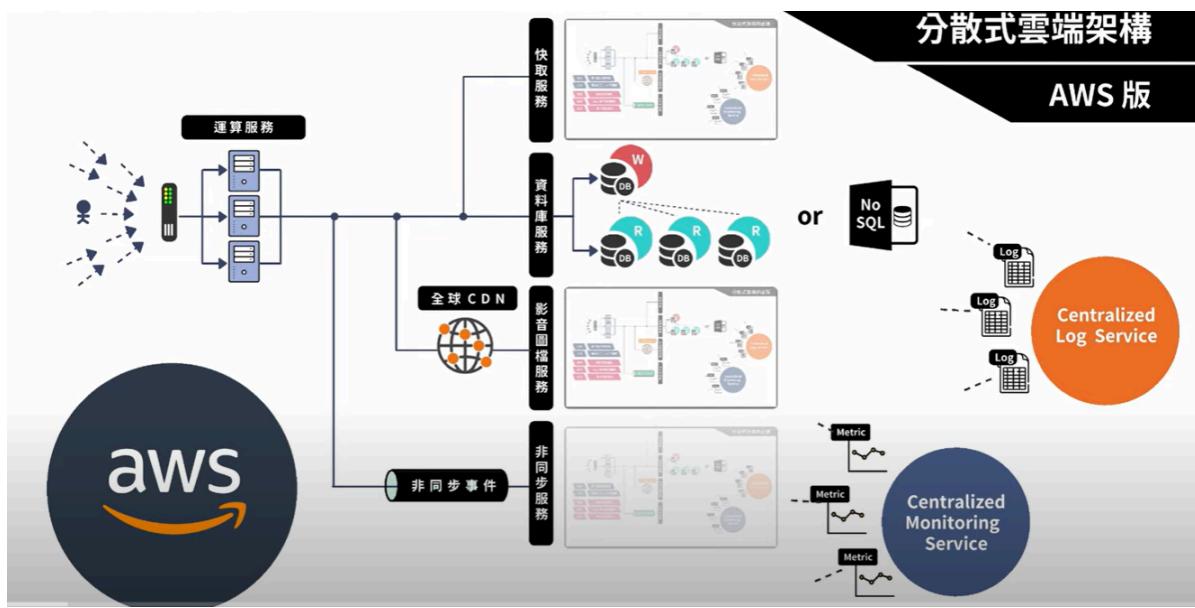
	S3 標準	S3 智慧型分層*	S3 標準 – IA	S3 單區域 – IA†	S3 Glacier	S3 Glacier Deep Archive
專為耐用性而設計	99.999999999% (11 9's)					
專為可用性而設計	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
可用性 SLA	99.9%	99%	99%	99%	99.9%	99.9%
可用區域	≥3	≥3	≥3	1	≥3	≥3
每個物件的最低容量費用	無	無	128 KB	128 KB	40 KB	40 KB
最短儲存持續時間費用	無	30 天	30 天	30 天	90 天	180 天
擷取費用	無	無	每 GB 擷取	每 GB 擷取	每 GB 擷取	每 GB 擷取
第一位元組延遲	毫秒	毫秒	毫秒	毫秒	選取分鐘或小時	選取小時
儲存體類型	物件	物件	物件	物件	物件	物件
生命週期轉換	是	是	是	是	是	是

† 因為 S3 單區域 – IA 將資料存放在單一 AWS 可用區域中，因此存放在這個儲存類別中的資料會在可用區域損壞時遺失

S3 權限管理機制

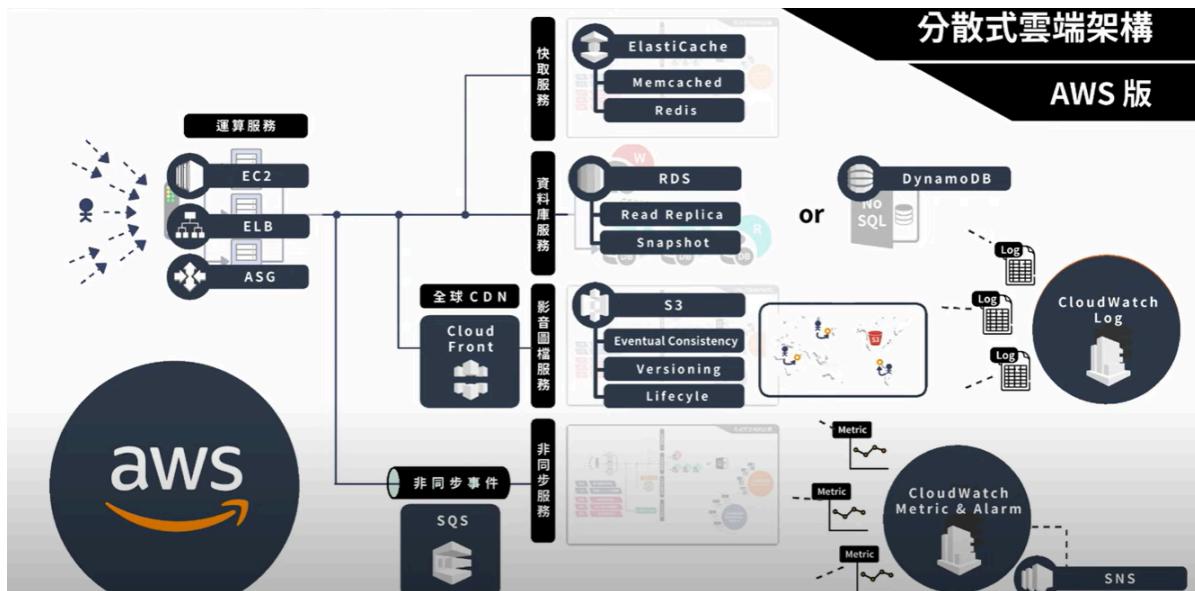


AWS 分散式架構



情境:

有大量流量從運算服務進來，會把請求送到資料庫服務(資料庫非常耗時且消耗大)，透過快取服務，影音圖片相關透過全球CDN讓各地的使用者更快取得資訊，以及非同步事件交由非同步服務處理，最後整體監測給監測系統及集中式的LOG管理系統處理。

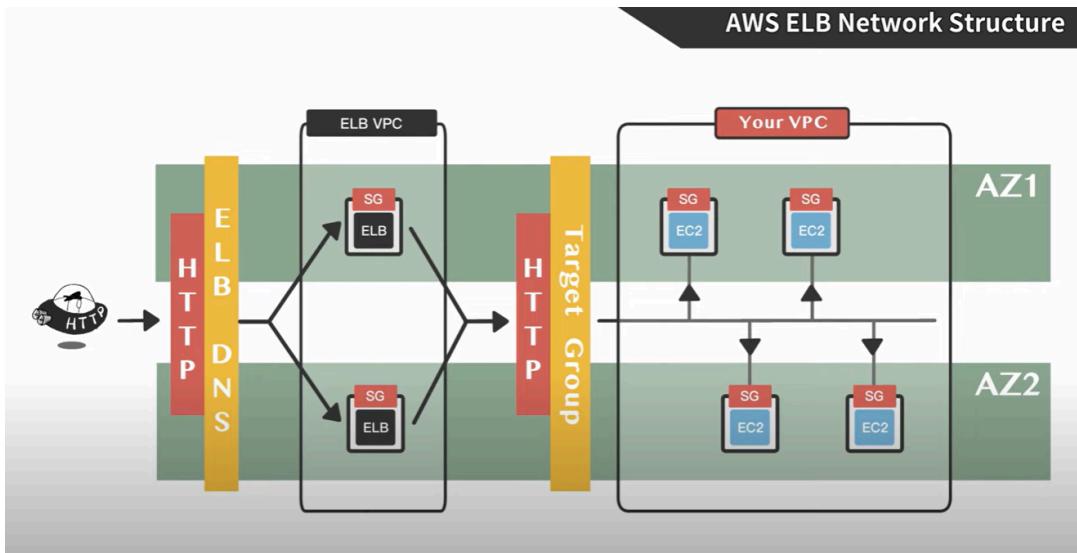


運算服務: EC2, 為了應付大量流量會有ELB (Elastic Load Balance)搭配ASG(Auto Scaling Group)來自動根據流量大小增減EC2的使用量。

Elastic Load Balancer (ELB)

ELB 是 high availability 和 scalability 重要的一環，分成以下三類：

- [Classic ELB](#) (較少出現)
- [Application Load Balancer \(ALB\)](#)
- [Network Load Balancer \(NLB\)](#).



資料庫服務: RDS (Relational Database Service), Read Replica 可應付大量讀取請求, Snapshot 處理資料備份。 NoSQL: DynamoDB 可應付大量讀取寫入請求。

[Amazon Relational Database \(RDS\)](#)

- 種類：SQL, MySQL, PostgreSQL, Oracle, Aurora, MariaDB
- Backup 功能分兩種：Automated Backups、Database Snapshots
- Multi-AZ：用途 Disaster Recovery (災難處理)
- Read Replicas：用途為 Increase Performance (增加效能)

[DynamoDB](#)

- NoSQL，儲存在 SSD storage
- 跨越三個不同地理區的 data centers
- Read Consistency (讀取一致性模型)
 - Eventual Consistent Reads (Default)：最終一致性，best read performance
 - Strongly Consistent Reads：強制一致性

快取服務Elastic Cache: Memcached, Redis

Elasticache

- 用途：優化 database 與 web application 的 performance
- 種類：Redis & Memcached
- Redis : Multi-AZ, 可以做 backups 和 restores
- Memcached : 當需要 scale horizontally (水平擴展)

影音圖檔服務: S3(本身就是典型的分散式架構)。

[Eventual Consistency]資料可能短時間不同步但最終會一致。

Versioning:預防重要檔案誤刪, Life cycle:根據檔案使用頻率來節省使用成本。

全球服務CDN: Cloud Front



Ex: 第一次要取得影音檔案時需要到美洲的S3取得資訊，但後續附近的使用想取得時可就近取得。

CloudFront

- Content delivery network (CDN) 服務 → 看到 CDN 聯想 CloudFront
- **Edge location** : 資料被 cached 的地方
- **Origin** : 所有檔案的 origin, 可以是 S3, EC2, ELB, 或 Route53
- **Distribution • Web Distribution** — 特別用於 Websites • **RTMP Distribution** — 用於 Media Streaming
- Edge locations 不單單只是 READ only, 你也可以 write (如：放 object 在上面)
- 物件被 cache 的時長基於 **TTL (Time To Live)**
- 可以清除 cache, 但會被收費

- Geo restriction 功能

非同步事件:SQS(Simple Queue Service), 可放入EC2, Lambda,公司內部主機等。

集中管理機制:

LOG服務:CloudWatch LOG

監測服務:CloudWatch Metric & Alarm - Metric 蒐集監測資訊(CPU數值,硬碟使用量),如果有異常會發起Alarm→SNS(Simple Notification Service)寄發通知訊息。



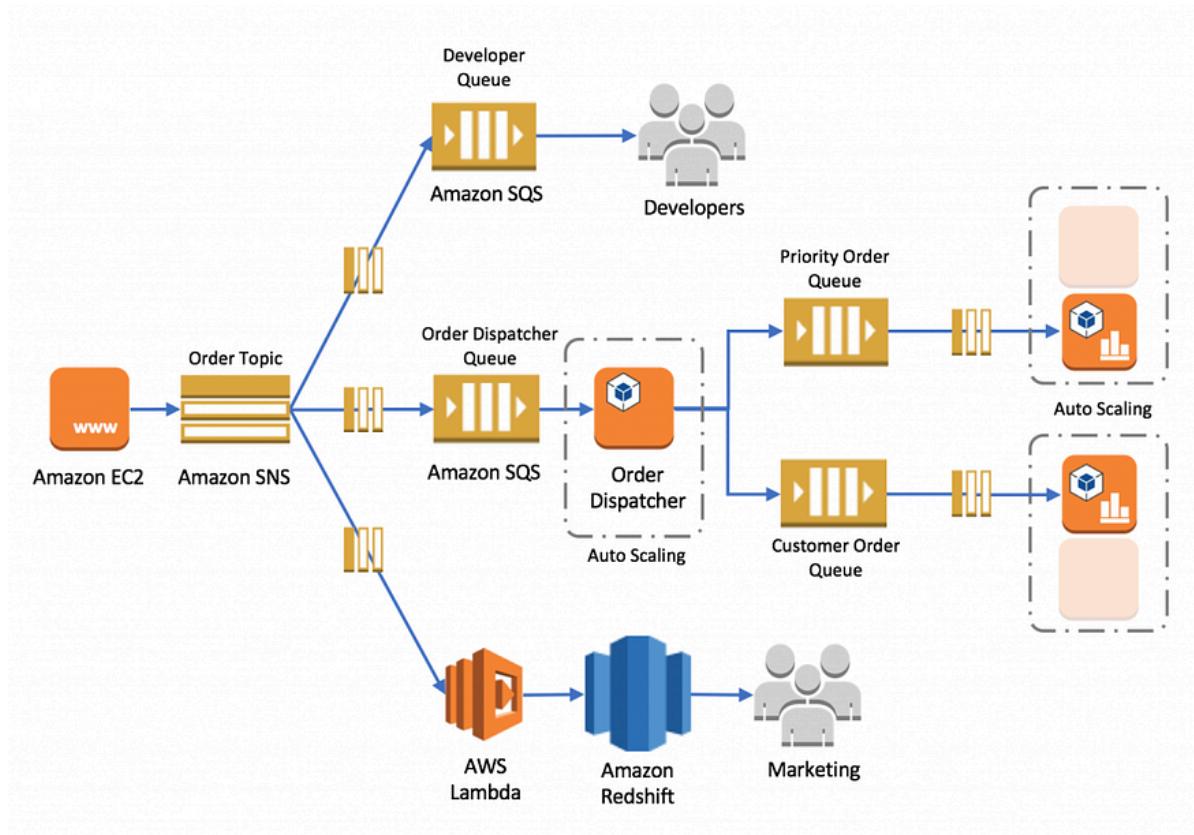
Design Cost-Optimized Architectures

我們需要知道如何將成本納入考量來建構雲端架構。

- 價格是基於 Compute, Storage, Data transfer
- 了解 EC2 價格模型，基於各式種類如：Compute-optimized, Memory-optimized, Storage-optimized 等
- 何時使用 reserved instances 取代 on-demand 或 spot instances
- 不需持續運作的可以用 Lambda
- 知道使用哪個 data storage service (EBS, S3, RDS) 與其種類
- 閱讀 [How AWS Pricing works 白皮書](#)

設計 decoupling (解耦) 系統

雲端架構的解耦元素非常重要。



解耦系統範例：SQS + Auto Scaling，此外注意 SNS, SNS, Kinesis 相關應用。

AWS 上雲實作紀錄

▼ 【創建 Robo Advisor】

▼ 【VPC 創建】

IPv4 CIDR 區塊

▼ 【創建 EC2】

- 記得要設定公網域 IPV4
- 進到 mobaXterm 後，如果無法順利執行 docker compose up -d，會需要先安裝docker、docker-compose，操作順序如下

```
sudo yum install -y python3-pip
```

```
sudo yum install -y docker
# 下載 Docker Compose 二進位檔案
sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

# 授予執行權限
sudo chmod +x /usr/local/bin/docker-compose

# 確認安裝
docker-compose --version

# 重新啟動 server、docker
sudo service docker start
sudo systemctl start docker

# 可以確認狀態
sudo service docker status
sudo systemctl status docker

# 執行 docker compose
sudo /usr/local/bin/docker-compose up -d
```

可以使用「`sudo usermod -a -G docker $USER`」將當前用戶添加到 docker 群組中，以便能夠在不使用 sudo 的情況下運行 Docker 命令

- 建立 root 帳號，需要輸入 `sudo passwd root`

▼ 【掛載 https】

▼ 安裝 Nginx

可以先查詢 nginx version <http://nginx.org/en/download.html>

1. 安裝 Nginx：`wget http://nginx.org/download/nginx-1.25.1.tar.gz`
2. 解壓縮源碼包：`tar -zxvf nginx-1.25.1.tar.gz`
3. 安裝編譯所需的依賴項，配置和編譯 Nginx 方法為：進入解壓縮後的目錄，創建 `configure`，先輸入「`./configure`」然後輸入「`make`」，安裝編譯完成的 Nginx 「`sudo make install`」

- 過程中如果出現錯誤訊息，就需要依依排除，如「checking for OS + Linux 5.10.179-171.711.amzn2.x86_64 ×86_64 checking for C compiler ... not found」，就會須要安裝 C 編譯器，執行「sudo yum groupinstall "Development Tools"」
完成後就可以繼續上一個步驟操作
- 錯誤訊息「./configure: error: the HTTP rewrite module requires the PCRE library. You can either disable the module by using --without-http_rewrite_module option, or install the PCRE library into the system, or build the PCRE library statically from the source with nginx by using --with-pcre=<path> option.」
可以先安裝部分套件

```
# 安裝 PCRE 開發套件
sudo yum install pcre-devel
# 安裝其他相依套件
sudo yum install zlib-devel openssl-devel
# 清除緩存
sudo yum clean all
# 重新執行
./configure
# 繼續編譯和安裝
make
sudo make install
```

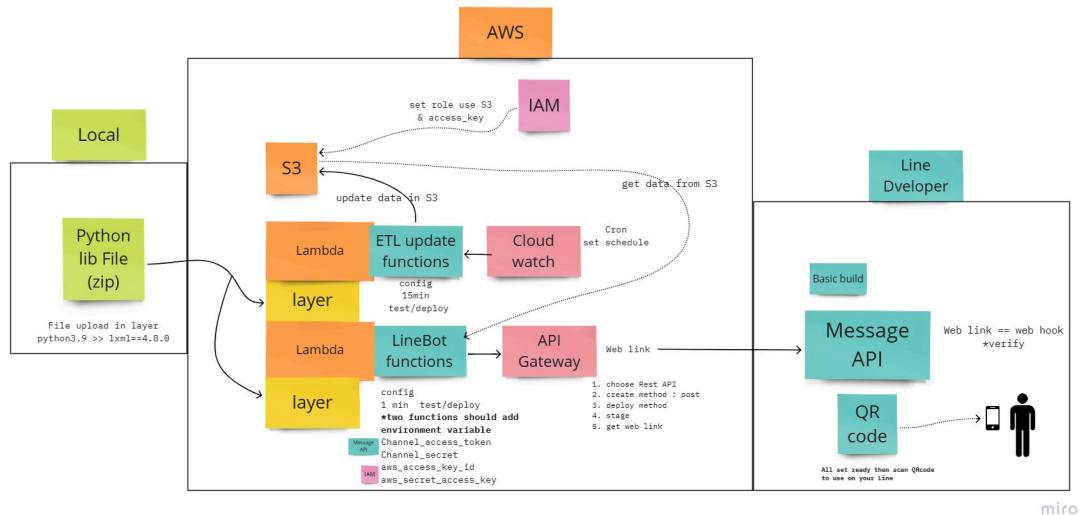
▼ 【創建 Nginx 配置文件】

▼ 【負載平衡器、目標群組】

mysql、https、http

▼ 【創建 Line chatbot】

▼ 流程圖



▼ 【如何將 EC2 公有 IPv4 地址，映射到自己買的網域】

▼ 【Godaddy 買網域並搭載進去】

DNS 管理
roboardvisor.website

DNS 記錄

新增記錄

類型	名稱	資料	TTL	刪除	編輯
A	@	13.230.179.229	600 秒		

▼ 【接下來進到 MobaXterm 開始操作】

▼ 【查看是否成功掛載】

輸入「nslookup 網站」 >> 「nslookup roboardvisor.website」

```
[ec2-user@ip-172-16-17-52 ~]$ nslookup roboadvisor.website
Server:      172.16.0.2
Address:     172.16.0.2#53

Non-authoritative answer:
Name:   roboadvisor.website
Address: 13.230.179.229
```

▼ 【使用 Nginx 將網站於前景呈現】

▼ 【先安裝 Nginx】

1. 更新套件庫：「sudo apt update」如果是 AWS Linux 「sudo yum update」
2. 安裝 Nginx 「sudo yum install nginx」，但因為 AWS Linux 的關係，需要使用「sudo amazon-linux-extras install nginx1」才可以安裝
3. 檢查 Nginx 服務狀態 「sudo systemctl status nginx」
4. 如果您已經安裝了 Nginx，但服務未啟動，您可以使用以下命令啟動 Nginx 服務：「sudo systemctl start nginx」
5. 如果您想要在開機時自動啟動 Nginx 服務，可以使用以下命令啟用 Nginx 服務：「sudo systemctl enable nginx」

▼ 【編輯 Nginx 配置文件】

進到 nginx.conf 資料路徑，etc/nginx/nginx.conf

編輯 Nginx 配置文件，配置反向代理的規則

```
server {
    listen 80;
    server_name roboadvisor.website;

    location / {
        proxy_pass http://13.230.179.229:5000; # 設定反向代理的目標 IP 和埠號
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
```

```
    }  
}
```

▼ 【重新啟動 Nginx 服務】

輸入「`sudo service nginx restart`」測試設置：打開您的瀏覽器並訪問 <http://roboadvisor.website>。應該可以看到您的網站正確呈現

先移動到目錄「`cd /etc/nginx/sites-available/`」

▼ 【接下來將 HTTP 轉為 HTTPS】申請 SSL 到建立的過程

前往網站「Let's Encrypt」取得 SSL 憑證

但在取得之前環境可能需要調整安裝 pip 套件管理員，流程如下

一、安裝 Python 的開發工具套件，請輸入「`sudo yum install python3-devel`」

二、下載並執行 `get-pip.py` 腳本來安裝 pip，輸入「`sudo curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py`」

三、輸入「`sudo python3 get-pip.py`」

完成後，就可以開始安裝套件，取得certbot

一、輸入「`pip install certbot`」安裝 certbot 套件

二、執行 certbot 命令來取得 SSL/TLS 憑證「`sudo certbot certonly --nginx`」

如果系統找不到指令，這可能是因為 Certbot 沒有正確安裝或無法找到可執行檔的路徑。請確認您已經正確安裝 Certbot。

在 AWS Linux 環境中，您可以嘗試使用以下命令來安裝 Certbot：

「`sudo amazon-linux-extras install epel`」

安裝 Certbot

「`sudo yum install certbot python2-certbot-nginx`」

完成安裝後再次輸入

「`sudo certbot certonly --nginx`」

這時候，系統會要求要輸入 email，然後要你選你要申請 SSL 的網站

系統要求您輸入電子郵件地址的原因是為了讓 Certbot 在憑證到期前提醒您續訂憑證。當憑證接近到期時，Certbot 將向您提供的電子郵件地址發送相關的提醒通知。

輸入完電子郵件地址後，Certbot 將繼續進行驗證和獲取 SSL/TLS 憑證的過程

三、接下來會收到驗證信件，驗證後就可以使用 <https://roboadvisor.website/> 來開啟

四、接下配置 Nginx 以使用 SSL：

編輯 Nginx 的配置文件 /etc/nginx/nginx.conf

並添加以下配置以啟用 SSL：

```
server {  
    ...  
    listen 443 ssl;  
    server_name roboadvisor.website;  
    ssl_certificate /etc/nginx/ssl/your_certificate.crt;  
    ssl_certificate_key /etc/nginx/ssl/your_private_key.key;  
    ...  
}  
# cert.pem chain.pem fullchain.pem privkey.pem README  
server {  
    ...  
    listen 443 ssl;  
    server_name roboadvisor.website;  
    ssl_certificate /etc/letsencrypt/live/roboadvisor.website/fullchain.pe  
m;  
    ssl_certificate_key /etc/letsencrypt/live/roboadvisor.website/privkey.p  
em;  
    ...  
}
```

Certbot 將憑證檔案保存在 /etc/letsencrypt/live/your_domain/ 目錄中，其中 your_domain 是您的網域名稱(roboadvisor.website)。該目錄中的檔案包括 cert.pem (SSL 憑證檔案)、privkey.pem (私鑰檔案) 和 fullchain.pem (包含憑證和中間證書的完整憑證鏈)。

五、輸入「`sudo systemctl reload nginx`」重新啟動 SSL

六、如果遇到狀況可以調整權限設定，輸入以下指令

「`sudo chmod 600 /etc/letsencrypt/live/roboadvisor.website/cert.pem`」、

「`sudo chmod 600 /etc/letsencrypt/live/roboadvisor.website/privkey.pem`」

最終Nginx 的配置文件應該要調整成如下：

```
user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log;
pid /run/nginx.pid;

events {
    worker_connections 1024;
}

http {
    log_format main '$remote_addr - $remote_user [$time_local] "$request"
                    '$status $body_bytes_sent "$http_referer"
                    '"$http_user_agent" "$http_x_forwarded_for"';

    access_log /var/log/nginx/access.log main;

    sendfile      on;
    tcp_nopush    on;
    tcp_nodelay   on;
    keepalive_timeout 65;
    types_hash_max_size 4096;

    include       /etc/nginx/mime.types;
    default_type  application/octet-stream;

    # Load modular configuration files from the /etc/nginx/conf.d directory.
    # See http://nginx.org/en/docs/ngx_core_module.html#include
    # for more information.
    include /etc/nginx/conf.d/*.conf;

    server {
        listen 80;
        server_name roboardvisor.website;
        return 301 https://$host$request_uri;
```

```

}

server {
    listen 443 ssl http2;
    server_name roboadvisor.website;

    ssl_certificate /etc/letsencrypt/live/roboadvisor.website/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/roboadvisor.website/privkey.pem;

    location / {
        proxy_pass http://54.95.184.190:5000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}

```

請確保您已經在 `/etc/letsencrypt/live/roboadvisor.website/` 目錄下擁有正確的 Let's Encrypt 憑證和私鑰文件。如果需要，您可以將憑證和私鑰文件的路徑更改為正確的位置。

應用新的配置文件後，請重新載入 Nginx 以使更改生效：「`sudo systemctl reload nginx`」

接下來就可以在 <https://roboadvisor.website/> 看到成果了

▼ 【Lambda function URL 映射到自己買的網域】 AWS Route 53 <筆記未完成>

要將您的 Lambda 函數 URL 映射到一個自定義域名，您可以透過以下步驟來完成：

1. **註冊域名**：確保您已擁有一個域名。如果您還沒有，可以透過 Amazon Route 53 或其他域名註冊服務購買一個。
2. **建立 SSL/TLS 證書**：對於您的自定義域名，您需要一個 SSL/TLS 證書以保證安全連接。您可以透過 AWS Certificate Manager (ACM) 免費獲取並管理這樣的證書。在 ACM 中，申請一個證書，並根據指示完成域名所有權的驗證。
3. **設定 API Gateway**：

- 創建一個新的 REST 或 HTTP API。
- 為您的 Lambda 函數建立一個新的觸發器。
- 建立資源和方法，然後將它們連接到您的 Lambda 函數。

4. 建立自定義域名：

- 在 API Gateway 控制台中，為您的 API 建立一個自定義域名。
- 選擇剛才在 ACM 中創建的 SSL/TLS 證書。
- 將自定義域名指向您的 API。

5. 建立基礎路徑映射：

- 在自定義域名設定中，建立一個新的基礎路徑映射。
- 將基礎路徑映射至您的 API 部署階段。

6. 更新 DNS 設定：

- 在您的域名提供商或 Amazon Route 53 中，將您的自定義域名的 DNS 記錄指向 API Gateway 提供的域名。
- 通常這會是一條類型為 CNAME 或 A 的記錄（取決於您的設定和域名提供商）。

7. 等待 DNS 傳播：DNS 更新可能需要一些時間來傳播。這可能需要幾分鐘到幾小時不等。

完成以上步驟後，您的自定義域名應該會指向您的 Lambda 函數。當您訪問自定義域名時，它會透過 API Gateway 轉發請求到您的 Lambda 函數。

如果您在過程中遇到任何問題，請提供更多詳情，以便我能提供進一步的幫助。

▼ 【前後端部屬流程】Vue3 + .NET Core8 (S3、ECS、CloudFront)

<https://chatgpt.com/share/6738b268-2730-8001-a029-80dc5a1b812b>

▼ GPT 內容資訊

AWS 部署 VUE3 API

你說：

我有一個問題，我現在有一個 VUE3(前端畫面)、.NET core8(API)，我該如何部屬的 AWS 上呢？

我不需要前端再使用 API 的時候，暴露我的 API server 真實地址，而是與前端一樣會有 Domain 保護

ChatGPT 說：

ChatGPT

在 AWS 上部署 Vue3 前端和 .NET Core 8 API 的方式，並同時隱藏 API 真實地址，以下是建議的完整步驟：

1. 準備前端與後端專案

前端 (Vue3)

1. 確保前端專案已打包 (`npm run build` 或 `yarn build`)。
 - 生成的靜態檔案通常位於 `dist` 資料夾。
2. 打包完成後，確認是否正確設定了 API 的 Proxy (或透過 Nginx 進行設定)。

後端 (.NET Core 8 API)

1. 確保 API 可用於生產環境，並已使用 Docker 或其他方法進行打包 (建議容器化)。

- **Dockerfile:**

```
dockerfile
複製程式碼
FROM mcr.microsoft.com/dotnet/aspnet:8.0 AS base
WORKDIR /app
COPY ..
EXPOSE 5000
ENTRYPOINT ["dotnet", "YourApp.dll"]
```

- 若需要 HTTPS，可以設定 `nginx` 或直接設定憑證。

2. 建立 AWS 服務

前端 (Vue3 部署到 S3 和 CloudFront)

1. 建立 S3 儲存桶

- 進入 AWS S3，建立一個儲存桶，選擇 **無公共存取阻止**。
- 上傳打包後的 `dist` 資料夾內容。

- 啟用「靜態網站託管」，並設定 `index.html` 為入口文件。

2. 設定 CloudFront

- 建立 CloudFront 發布分配 (Distribution)。
- 指定 S3 儲存桶作為來源 (Origin)。
- 啟用 HTTPS (使用 AWS 提供的免費 ACM 憑證)。
- 設定快取行為，將所有路徑 (`/*`) 指向 S3，確保 404 頁面正確處理。

後端 (.NET Core 8 API 部署到 ECS)

1. 建立 ECR 儲存庫

- 上傳 Docker 映像至 AWS Elastic Container Registry (ECR)。
- 推送命令範例：

```
bash
複製程式碼
docker tag your-image-name:latest <your-account-id>.dkr.ecr.<region>.amazonaws.com/your-repository-name:latest
docker push <your-account-id>.dkr.ecr.<region>.amazonaws.com/your-repository-name:latest
```

2. 建立 ECS

- 建立 ECS 叢集 (Cluster)。
- 配置 Fargate 作為執行環境。
- 設定服務和任務，並使用從 ECR 部署的 Docker 映像。
- 暴露的 API 埠應只允許 ALB 存取。

3. 設定 Application Load Balancer (ALB)

- 建立 ALB，並指向 ECS 的後端服務。
- 配置 HTTPS，並使用 ACM 的憑證。
- 限制 ALB 僅接收來自 CloudFront 的流量。

3. 隱藏 API 實地地址

透過 CloudFront 保護 API

1. 為 API 建立一個 CloudFront 發布分配

- 指定 ALB 為 CloudFront 的來源。
- 啟用 **Origin Access Control (OAC)**，以限制來源請求。

2. 使用 Lambda@Edge 或 Rewrite URL

- 如果需要對 API 地址進一步處理，可以使用 Lambda@Edge 來重寫請求。

3. 前端 API 地址配置

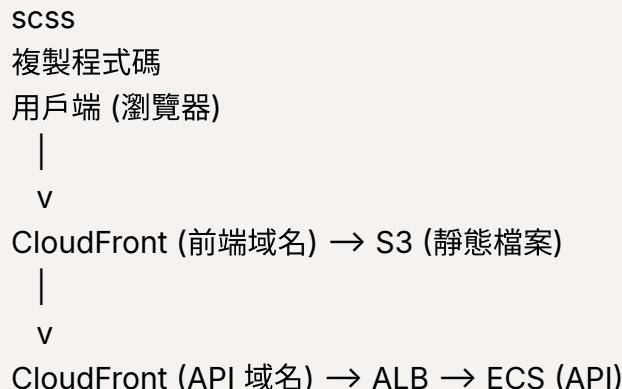
- 在前端，配置 API 的 Base URL 為 CloudFront 的域名，例如：

```
javascript  
複製程式碼  
const API_BASE_URL = "https://your-cloudfront-domain/api";
```

4. 測試與驗證

- 確保前端與 API 的請求都通過 CloudFront 域名。
- 驗證 HTTPS 是否正常，並測試是否能正常存取 API。

5. 整體架構圖



如需進一步協助設定，可以提供你的架構細節或問題，我會再針對需求提供解決方案！

[繼續此對話](#)

[檢舉內容](#) | [使用條款](#) | [隱私權政策](#) |

AWS 流量監管

▼ 【IP 流量監管】

AWS 架構圖

▼