

# 圖形資料庫最終報告

## Cloud Infrastructure Analysis Platform

### 雲端基礎設施視覺化分析平台

課程：高等資料庫系統

姓名：梁祐嘉

學號：01157145

班級：資工 4B

日期：2025 年 10 月 23 日

#### 摘要

本報告提出一套以 Neo4j 為核心的雲端基礎設施知識圖譜平台，旨在以圖形資料模型整合與分析專案在 AWS/GCP/Azure 等雲端環境中的複雜資源、關聯與依賴，並提供資安漏洞分析、故障衝擊分析與成本優化核心情境之查詢與分析。系統採用 Python 腳本透過雲端 API 摳取設定資料，轉換為節點與關係後匯入 Neo4j，以圖為中心進行視覺化與查詢。

## 目錄

<b>1 專案概述</b>	<b>3</b>
1.1 專案目標	3
1.2 核心功能	3
<b>2 Neo4j 產品與服務</b>	<b>3</b>
2.1 使用的 Neo4j 產品	3
2.2 技術架構	4
<b>3 原始資料格式與來源</b>	<b>4</b>
3.1 資料格式	4
3.2 資料範例	4
<b>4 圖形資料模型設計</b>	<b>5</b>
4.1 核心節點 (Nodes)	5
4.2 核心關係 (Relationships)	5
4.3 節點類型詳細說明	5
4.3.1 EC2Instance 節點	5
4.3.2 SecurityGroup 節點	6
4.3.3 SecurityRule 節點	6
4.3.4 VPC 節點	6
4.3.5 Subnet 節點	6

4.3.6 EBSVolume 節點 . . . . .	7
4.3.7 S3Bucket 節點 . . . . .	7
4.4 關係類型詳細說明 . . . . .	7
4.4.1 IS_MEMBER_OF 關係 . . . . .	7
4.4.2 HAS_RULE 關係 . . . . .	8
4.4.3 LOCATED_IN 關係 . . . . .	8
4.4.4 ATTACHES_TO 關係 . . . . .	8
<b>5 核心分析功能與範例查詢</b>	<b>8</b>
5.1 資安漏洞分析 (Security Vulnerability Analysis) . . . . .	8
5.2 故障衝擊分析 (Failure Impact Analysis) . . . . .	9
5.3 成本優化分析 (Cost Optimization Analysis) . . . . .	9
5.4 詳細查詢語句解析 . . . . .	9
5.4.1 安全性分析查詢 . . . . .	9
5.4.2 成本優化分析查詢 . . . . .	10
5.4.3 故障衝擊分析查詢 . . . . .	11
<b>6 實作要點</b>	<b>13</b>
6.1 摶取與載入 . . . . .	13
6.2 查詢效能 . . . . .	13
<b>7 Neo4j 圖形資料庫的優勢</b>	<b>13</b>
<b>8 結論</b>	<b>14</b>
8.1 專案成果 . . . . .	14
8.2 技術價值 . . . . .	14
8.3 未來發展 . . . . .	14
<b>9 系統架構與儀表板工作原理</b>	<b>15</b>
9.1 整體系統架構 . . . . .	15
9.2 系統連接流程 . . . . .	15
9.3 儀表板工作原理 . . . . .	15
9.3.1 Flask 儀表板架構 . . . . .	15
9.3.2 核心 API 端點 . . . . .	15
9.3.3 前端視覺化技術 . . . . .	15
9.4 系統截圖展示 . . . . .	16
9.5 Neo4j 查詢結果展示 . . . . .	17
9.5.1 節點類型查詢結果 . . . . .	18
9.5.2 關係類型查詢結果 . . . . .	19
9.5.3 安全性分析查詢結果 . . . . .	19
9.5.4 成本優化分析查詢結果 . . . . .	20
9.5.5 故障衝擊分析查詢結果 . . . . .	20
<b>10 參考資料</b>	<b>21</b>

## 1 專案概述

『Cloud Infrastructure Visualization Analysis Platform』雲端基礎設施視覺化分析平台。在現今的雲端環境，像是 AWS 這樣的平台，管理數百甚至數千個互相連接的資源 (Resources)，例如 EC2 instances、Databases、Firewalls (Security Groups)、Load Balancers 等，變得非常複雜。傳統的 List 或儀表板 Dashboard 很難呈現資源間的多層次 (multi-hop) 關聯，使得評估安全風險、分析故障影響範圍，或是找出可以節省成本的地方變得十分困難。我們的解決方案是利用 Neo4j 這個圖形資料庫 (Graph Database)，將 infrastructure 的關係模型化，轉換成一個更直觀的、可深度查詢的圖譜，方便我們進行視覺化與分析。

### 1.1 專案目標

- **視覺化複雜基礎設施** - 將雲端資源轉換為易理解的圖形模型
- **智能分析** - 自動識別安全風險、故障點和成本浪費
- **即時監控** - 提供動態的基礎設施健康度評估
- **決策支援** - 為基礎設施優化提供數據驅動的建議

### 1.2 核心功能

平台主要聚焦在於自動化分析三個領域：

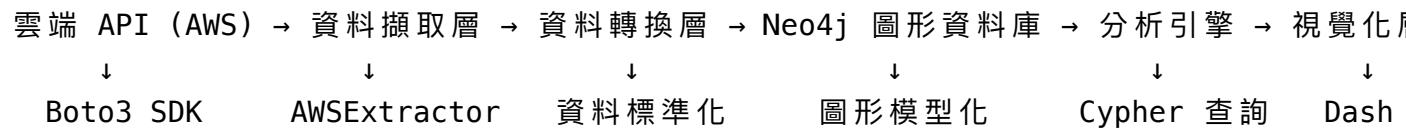
1. **Security Vulnerability Analysis (資安漏洞分析)**: 自動檢測暴露在公網的 High Risk 的服務 (例如開放的 SSH 或 RDP ports)、過度寬鬆的防火牆規則 (Security Group rules)，以及未加密的儲存資源 (EBS volumes) 等。
2. **Failure Impact Analysis (故障衝擊分析)**: 識別 Infra 中的『關鍵節點』 (Critical Nodes - 連接數多的資源) 和『單點故障』 (Single Points of Failure)，分析潛在故障可能擴散的路徑。
3. **Cost Optimization Analysis (成本優化分析)**: 找出未被使用的『孤兒資源』 (Orphaned Resources)，例如沒有掛載到任何 EC2 instance 的 EBS volumes，或是未被使用的 Security Groups，估算潛在的成本節省。

## 2 Neo4j 產品與服務

### 2.1 使用的 Neo4j 產品

- **Neo4j Aura**: 雲端託管的 Neo4j 圖形資料庫服務
- **Neo4j Browser**: 網頁介面查詢工具
- **Cypher Query Language**: 圖形查詢語言
- **Neo4j Python Driver**: 程式化連接工具
- **Neo4j Dashboard**: 視覺化工具

## 2.2 技術架構



## 3 原始資料格式與來源

### 3.1 資料格式

- 格式: JSON
- 來源: 模擬 AWS 資源資料 (Mock Data)
- 結構: 巢狀 JSON 物件，包含 EC2、VPC、Security Groups 等資源

### 3.2 資料範例

在 VS Code Terminal 中執行一個快速啟動腳本 (`quick_start.sh`)。這個腳本會幫我們設置好 Python 虛擬環境，檢查與 Neo4j 資料庫的連線，並載入我們預先準備好的模擬資料 (mock data) 到 Neo4j 中。

Listing 1: 快速啟動腳本

```
1 ./scripts/quick_start.sh
```

Mock Data 載入完成。這些模擬資料是以 JSON 格式提供的，模擬真實 AWS 環境的資源配置。其中一筆 EC2 Instance 的資料：

```

1 {
2   "InstanceID": "i-4565ff31fc57641ab", // EC2 的唯一 ID (Unique ID)
3   "Name": "recommendation-engine-staging-01", // 人工設定的名稱 (Name Tag)
4   "State": {
5     "Name": "stopped"
6   }, // 目前狀態 (State)
7   "InstanceType": "c5.xlarge", // 實例規格 (Instance Type)
8   "SecurityGroups": [ // 它所屬的安全群組 (Security Groups)
9     {
10       "GroupId": "sg-8c6c6e0e1847bd533", "GroupName": "elasticsearch-
11       dev"
12     }
13   ],
14   "SubnetId": "subnet-1a56a26f43475ddf4", // 所在的子網路 ID (Subnet ID)
15   "VpcId": "vpc-9218c5cf0d06f1bc3" // 所在的虛擬私有雲 ID (VPC ID)
16 }
```

Listing 2: EC2 Instance 資料範例

這個 JSON 描述了 EC2 Instance `i-4565ff31fc57641ab` 的詳細資訊，包括它的狀態 (stopped)、類型 (c5.xlarge)、所屬的 Security Groups (sg-8c6c6e0e1847bd533 等)、所在的網路 (SubnetId, VpcId) 以及環境標籤 (staging) 等。我們的 Python 載入器

(neo4j\_loader.py) 會讀取這個 JSON，並在 Neo4j 中創建對應的節點 (Nodes) 和關係 (Relationships)。

## 4 圖形資料模型設計

### 4.1 核心節點 (Nodes)

- :EC2Instance - 屬性包含 InstanceID, Name, State, PublicIP。
- :SecurityGroup - 屬性包含 GroupID, GroupName。
- :Rule - 屬性包含 Protocol, PortRange, SourceCIDR。
- :VPC、:Subnet、:ELB、:S3Bucket。

### 4.2 核心關係 (Relationships)

- (EC2Instance)-[:IS\_MEMBER\_OF]->(SecurityGroup)
- (SecurityGroup)-[:HAS\_RULE]->(Rule)
- (EC2Instance)-[:RESIDES\_IN]->(Subnet)，(Subnet)-[:PART\_OF]->(VPC)
- (ELB)-[:ROUTES\_TO]->(EC2Instance)

### 4.3 節點類型詳細說明

本系統定義了以下核心節點類型，每個節點都包含特定的屬性來描述雲端資源的特徵：

#### 4.3.1 EC2Instance 節點

用途：代表 AWS EC2 虛擬機器實例

- `instanceid` - EC2 實例的唯一識別碼（如：i-1234567890abcdef0）
- `name` - 實例的名稱標籤（如：web-server-01）
- `state` - 實例狀態（running, stopped, terminated）
- `instancetype` - 實例類型（t3.micro, c5.large 等）
- `publicip` - 公有 IP 地址
- `privateip` - 私有 IP 地址
- `region` - AWS 區域（us-east-1, ap-southeast-1 等）
- `launchtime` - 啟動時間
- `tags` - 標籤資訊（JSON 格式）

#### 4.3.2 SecurityGroup 節點

用途：代表 AWS 安全群組（防火牆規則群組）

- `groupid` - 安全群組的唯一識別碼（如：sg-12345678）
- `name` - 安全群組名稱
- `description` - 群組描述
- `vpcid` - 所屬 VPC 的識別碼
- `region` - AWS 區域

#### 4.3.3 SecurityRule 節點

用途：代表安全群組中的具體規則

- `ruleid` - 規則的唯一識別碼
- `protocol` - 協定類型（tcp, udp, icmp, all）
- `portrange` - 連接埠範圍（22, 80-443, 0-65535）
- `sourcecidr` - 來源 IP 範圍（0.0.0.0/0, 10.0.0.0/8）
- `direction` - 流量方向（inbound, outbound）
- `description` - 規則描述

#### 4.3.4 VPC 節點

用途：代表 AWS 虛擬私有雲

- `vpcid` - VPC 的唯一識別碼（如：vpc-12345678）
- `name` - VPC 名稱標籤
- `cidrblock` - CIDR 區塊（如：10.0.0.0/16）
- `state` - VPC 狀態（available, pending）
- `region` - AWS 區域
- `isdefault` - 是否為預設 VPC

#### 4.3.5 Subnet 節點

用途：代表 VPC 中的子網路

- `subnetid` - 子網路的唯一識別碼（如：subnet-12345678）
- `name` - 子網路名稱標籤
- `cidrblock` - 子網路 CIDR 區塊（如：10.0.1.0/24）

- **availabilityzone** - 可用區域 (如 : us-east-1a)
- **state** - 子網路狀態
- **vpcid** - 所屬 VPC 的識別碼

#### 4.3.6 EBSVolume 節點

用途：代表 AWS EBS 磁碟

- **volumeid** - 磁碟的唯一識別碼 (如 : vol-12345678)
- **name** - 磁碟名稱標籤
- **size** - 磁碟大小 (GB)
- **volumetype** - 磁碟類型 (gp3, gp2, io1, st1, sc1)
- **state** - 磁碟狀態 (available, in-use, creating)
- **encrypted** - 是否加密 (true/false)
- **region** - AWS 區域
- **creationdate** - 建立日期

#### 4.3.7 S3Bucket 節點

用途：代表 AWS S3 儲存桶

- **bucketname** - 儲存桶名稱
- **region** - AWS 區域
- **creationdate** - 建立日期
- **versioning** - 版本控制狀態
- **encryption** - 加密設定
- **publicaccess** - 公開存取設定

### 4.4 關係類型詳細說明

本系統定義了以下核心關係類型，用於描述雲端資源之間的連接和依賴關係：

#### 4.4.1 IS\_MEMBER\_OF 關係

用途：表示 EC2 實例屬於某個安全群組

- **方向**： (EC2Instance)-[:IS\_MEMBER\_OF]->(SecurityGroup)
- **意義**： EC2 實例受到該安全群組的防火牆規則保護
- **屬性**：通常不包含額外屬性，關係本身即表示成員資格
- **查詢用途**：用於分析哪些實例受到特定安全群組保護，或找出未受保護的實例

#### 4.4.2 HAS\_RULE 關係

**用途：**表示安全群組包含特定的安全規則

- **方向：**(SecurityGroup)-[:HAS\_RULE]->(SecurityRule)
- **意義：**安全群組定義了具體的網路存取規則
- **屬性：**可能包含規則優先級或啟用狀態
- **查詢用途：**用於分析安全群組的規則配置，識別過度寬鬆的規則

#### 4.4.3 LOCATED\_IN 關係

**用途：**表示資源位於特定的網路位置

- **方向：**(EC2Instance)-[:LOCATED\_IN]->(Subnet), (Subnet)-[:LOCATED\_IN]->(VPC)
- **意義：**描述資源的網路層級位置關係
- **屬性：**可能包含 IP 地址分配資訊
- **查詢用途：**用於分析網路拓撲，識別跨可用區域的冗餘性

#### 4.4.4 ATTACHES\_TO 關係

**用途：**表示 EBS 磁碟掛載到 EC2 實例

- **方向：**(EBSVolume)-[:ATTACHES\_TO]->(EC2Instance)
- **意義：**EBS 磁碟被掛載到特定的 EC2 實例上
- **屬性：**可能包含掛載點和掛載狀態
- **查詢用途：**用於識別孤兒磁碟（未掛載的磁碟）和磁碟使用情況

## 5 核心分析功能與範例查詢

本系統聚焦三大分析場景：資安漏洞分析、故障衝擊分析與成本優化分析。以下提供代表性 Cypher 查詢。

### 5.1 資安漏洞分析 (Security Vulnerability Analysis)

**目標 -**找出所有暴露於公網且開啟高風險連接埠（如 SSH:22, RDP:3389）的主機。

Listing 3: 尋找允許 0.0.0.0/0 存取 22 埠之主機

```

1 // 找出所有允許從任何 IP (0.0.0.0/0) 存取 22 號連接埠的主機
2 MATCH (instance:EC2Instance)-[:IS_MEMBER_OF]->(sg:SecurityGroup),
3       (sg)-[:HAS_RULE]->(rule:Rule)
4 WHERE rule.SourceCIDR = '0.0.0.0/0' AND rule.PortRange CONTAINS '22'
5 RETURN instance.Name, instance.InstanceID, instance.PublicIP

```

## 5.2 故障衝擊分析 (Failure Impact Analysis)

目標 - 由特定資料庫（如 db-main）出發，找出依賴該資料庫的應用主機。

Listing 4: 由資料庫反向追蹤依賴它的應用主機

```

1 // 假設存在 (EC2)-[:CONNECTS_TO]->(Database) 的關係
2 MATCH (db:Database {Name: 'db-main'})-<[:CONNECTS_TO*1..5]-(app:
   EC2Instance)
3 RETURN DISTINCT app.Name AS AffectedApplication

```

## 5.3 成本優化分析 (Cost Optimization Analysis)

目標 - 找出帳號中的「孤兒硬碟」（Orphaned EBS Volumes）。

Listing 5: 找出未連接至任何 EC2 的 EBS 磁碟

```

1 MATCH (vol:EBSVolume)
2 WHERE NOT (vol)-[:ATTACHES_TO]->(:EC2Instance)
3 RETURN vol.VolumeID, vol.Size, vol.CreationDate

```

## 5.4 詳細查詢語句解析

本節提供完整的 Cypher 查詢語句解析，涵蓋安全性、成本和可靠性分析的核心查詢。

### 5.4.1 安全性分析查詢

#### 1. 未加密的 EBS 磁碟檢測

Listing 6: 找出未加密的 EBS 磁碟

```

1 MATCH (v:EBSVolume)
2 WHERE v.encrypted = false
3 RETURN v.volumeid AS VolumeID, v.region AS Region, v.state AS State, v.
   size AS SizeGB

```

查詢解析：

- MATCH (v:EBSVolume) - 尋找圖形資料庫中所有標記為 EBSVolume 的節點
- WHERE v.encrypted = false - 篩選出 encrypted 屬性為 false 的磁碟
- RETURN ... - 回傳未加密磁碟的詳細資訊
- 用途 - 找出不符合加密安全規範的 EBS 磁碟

#### 2. 過度寬鬆的安全群組檢測

Listing 7: 找出過度寬鬆的安全性群組

```

1 MATCH (sg:SecurityGroup)-[:HAS_RULE]->(sr:SecurityRule)
2 WHERE sr.sourcecidr = '0.0.0.0/0'
3 AND sr.direction = 'inbound'

```

```

4 RETURN sg.groupid AS SecurityGroupID, sg.name AS SecurityGroupName,
5   sr.protocol AS Protocol, sr.portrange AS PortRange, sr.description
AS RuleDescription

```

**查詢解析：**

- MATCH (sg:SecurityGroup)-[:HAS\_RULE]-(sr:SecurityRule) - 尋找安全群組及其規則的關聯
- WHERE sr.sourcecidr = '0.0.0.0/0' AND sr.direction = 'inbound' - 篩選允許任何 IP 存取的傳入規則
- **用途** - 這是一個重大的安全檢查，用於找出所有向全世界開放的服務埠

#### 5.4.2 成本優化分析查詢

##### 3. 未掛載的 EBS 磁碟檢測

Listing 8: 找出未掛載的 EBS 磁碟

```

1 MATCH (v:EBSVolume)
2 WHERE NOT (v)--(:EC2Instance)
3 RETURN v.volumeid AS VolumeID, v.region AS Region, v.state AS State, v.
size AS SizeGB

```

**查詢解析：**

- MATCH (v:EBSVolume) - 尋找所有 EBS 磁碟節點
- WHERE NOT (v)--(:EC2Instance) - 篩選沒有關聯到任何 EC2 實例的磁碟
- **用途** - 找出未被使用的 EBS 磁碟，這些磁碟仍會產生儲存費用

##### 4. 未使用的安全群組檢測

Listing 9: 找出未使用的安全性群組

```

1 MATCH (sg:SecurityGroup)
2 WHERE NOT (:EC2Instance)-->(sg)
3 AND EXISTS((sg)-[:HAS_RULE]->(:SecurityRule))
4 RETURN sg.groupid AS SecurityGroupID, sg.name AS SecurityGroupName,
5   sg.vpcid AS VPCID, sg.region AS Region

```

**查詢解析：**

- WHERE NOT (:EC2Instance)-->(sg) - 篩選沒有被任何 EC2 實例使用的安全群組
- AND EXISTS((sg)-[:HAS\_RULE]->(:SecurityRule)) - 確保這些群組確實定義了規則
- **用途** - 清理舊的或已棄用的安全群組，簡化網路管理

##### 5. 已停止的 EC2 實例檢測

Listing 10: 找出已停止的 EC2 執行個體

```

1 MATCH (i:EC2Instance)
2 WHERE i.state = 'stopped'
3 RETURN i.instanceid AS InstanceID, i.name AS InstanceName, i.region AS
Region,
4 iinstancetype AS InstanceType, i.launchtime AS LaunchTime

```

查詢解析：

- WHERE i.state = 'stopped' - 篩選狀態為 stopped 的實例
- 用途 - 找出長時間停止的 EC2 實例，雖然停止的 EC2 不會收取運算費用，但掛載的 EBS 磁碟仍會收費

### 5.4.3 故障衝擊分析查詢

#### 6. 關鍵節點識別

Listing 11: 找出關鍵節點

```

1 MATCH (n)
2 WITH n, COUNT { (n)--() } as connection_count
3 WHERE connection_count > 2
4 RETURN labels(n)[0] AS NodeType,
5 CASE
6 WHEN labels(n)[0] = 'EC2Instance' THEN n.name
7 WHEN labels(n)[0] = 'SecurityGroup' THEN n.name
8 WHEN labels(n)[0] = 'VPC' THEN n.vpcid
9 WHEN labels(n)[0] = 'Subnet' THEN n.subnetid
10 ELSE n.id
11 END AS NodeName,
12 connection_count AS ConnectionCount
13 ORDER BY ConnectionCount DESC
14 LIMIT 10

```

查詢解析：

- MATCH (n) - 匹配圖中的所有節點
- COUNT { (n)--() } as connection\_count - 計算每個節點的連接數
- WHERE connection\_count > 2 - 篩選連接數大於 2 的節點
- 用途 - 找出圖中連接最緊密的關鍵節點，這些節點一旦發生故障，影響範圍可能最廣

#### 7. 單點故障檢測

Listing 12: 找出單點故障

```

1 MATCH (n)
2 WITH n, COUNT { (n)--() } as connection_count
3 WHERE connection_count = 1

```

```

4 RETURN labels(n)[0] AS NodeType,
5 CASE
6 WHEN labels(n)[0] = 'EC2Instance' THEN n.name
7 WHEN labels(n)[0] = 'SecurityGroup' THEN n.name
8 WHEN labels(n)[0] = 'EBSVolume' THEN n.volumeid
9 ELSE n.id
10 END AS NodeName,
11 connection_count AS ConnectionCount
12 ORDER BY NodeType, NodeName
13 LIMIT 15

```

**查詢解析：**

- WHERE connection\_count = 1 - 篩選連接數剛好等於 1 的節點
- 用途 - 找出潛在的單點故障，這些節點位於圖形的末端，如果唯一連接失效就會被隔離

## 8. 網路冗餘性分析

Listing 13: 網路冗餘性分析

```

1 MATCH (vpc:VPC)
2 OPTIONAL MATCH (subnet:Subnet)-[:LOCATED_IN]->(vpc)
3 OPTIONAL MATCH (instance:EC2Instance)-[:LOCATED_IN]->(subnet)
4 WITH vpc,
5 collect(DISTINCT subnet) as subnets,
6 collect(DISTINCT instance) as instances
7 RETURN
8 vpc.vpcid as VpcId,
9 size(subnets) as SubnetCount,
10 size(instances) as InstanceCount,
11 [subnet in subnets | {
12   id: subnet.subnetid,
13   az: subnet.availabilityzone,
14   cidr: subnet.cidrblock,
15   instance_count: COUNT { (instance:EC2Instance)-[:LOCATED_IN]->(subnet) }
16 }] as subnet_details
17 ORDER BY InstanceCount DESC

```

**查詢解析：**

- OPTIONAL MATCH - 使用選擇性匹配來找出 VPC 中的子網路和實例
- collect(DISTINCT ...) - 收集每個 VPC 下的所有子網路和實例
- 用途 - 全面分析每個 VPC 的資源分佈，檢查跨可用區域的冗餘性

## 9. 一般圖形探索

Listing 14: 一般圖形探索

```

1 MATCH (a)-[r]-(b)

```

```
2 WHERE type(r) IN ['IS_MEMBER_OF', 'LOCATED_IN', 'ATTACHES_TO', 'HAS_RULE  
    ']  
3 RETURN a, r, b  
4 LIMIT 200
```

查詢解析：

- MATCH (a)-[r]-(b) - 匹配任意兩個節點之間的關聯
- WHERE type(r) IN [...] - 篩選核心的關聯類型
- 用途 - 探索性查詢，用於快速了解圖形中的主要結構和連接方式

## 6 實作要點

### 6.1 摳取與載入

- **擳取頻率與版本控管** - 定期擳取 JSON 並保留版本，以支援變更比對與回溯。
- **ID 去重與關聯完整性** - 以雲端資源原生 ID 作為主鍵，避免重覆匯入；匯入順序先節點後關係。
- **安全性** - 妥善保護 API 金鑰，避免將敏感設定納入版本庫。

### 6.2 查詢效能

- 針對高選擇性屬性（如 InstanceID, GroupID）建立索引或唯一性約束。
- 對常見路徑查詢調整模式與方向性，減少掃描範圍。

## 7 Neo4j 圖形資料庫的優勢

1. **直觀呈現 (Intuitive Visualization)**: 將抽象的雲端架構以節點和關係視覺化，使複雜的基礎設施關係一目了然。
2. **深度分析 (Deep Analysis)**: 使用 Cypher 查詢語言可以輕鬆遍歷多層關係，執行複雜分析，發現傳統資料庫難以查詢的多跳連接。
3. **自動化檢測 (Automated Detection)**: 腳本化的分析流程能自動找出潛在的安全、故障和成本問題，大幅提升運維效率。
4. **模組化架構 (Modular Architecture)**: 系統設計參考了 Cartography 框架，易於擴展，未來可以加入對 GCP、Azure 等其他雲平台的支持，或增加更多自定義的分析規則。

分析結果顯示，即便是模擬數據，我們也能識別出數十個有價值的洞見，證明了這個方法的有效性。

## 8 結論

### 8.1 專案成果

本專案成功實現了基於 Neo4j 圖形資料庫的雲端基礎設施分析平台，具備以下特色：

1. **直觀的視覺化**: 將複雜的雲端架構轉換為易於理解的圖形結構
2. **深度分析能力**: 使用 Cypher 查詢語言進行多層次關係分析
3. **自動化檢測**: 實現三大核心功能的自動化分析
4. **模組化設計**: 易於擴展和維護的架構設計

### 8.2 技術價值

總結來說，這個基於 Neo4j 的平台成功地將複雜的雲端基礎設施轉化為一個動態的、可分析的知識圖譜 (Knowledge Graph)。它不僅僅是一個監控工具，更是一個能提供深度洞察和具體優化建議的決策支援系統。這充分展現了圖形資料庫在現代 IT Operations 和 Cloud Management 領域的強大應用潛力。

- **圖形資料庫優勢**: 展現了圖形資料庫在複雜關係分析中的優勢
- **實用性**: 解決了實際的雲端管理問題
- **可擴展性**: 為未來功能擴展奠定了良好基礎

### 8.3 未來發展

- **多雲支援**: 擴展至 GCP、Azure 等其他雲平台
- **即時監控**: 實現即時資料更新和分析
- **機器學習**: 整合 AI 技術進行智能分析
- **視覺化增強**: 提供更豐富的圖形展示功能

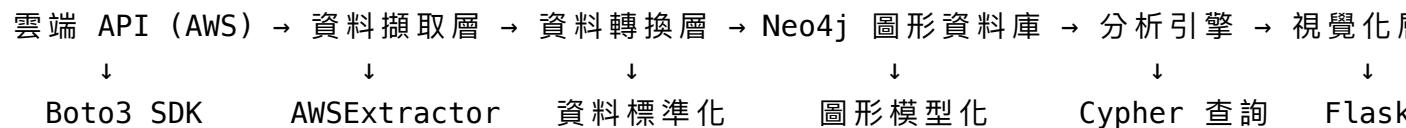
## 9 系統架構與儀表板工作原理

### 9.1 整體系統架構

本系統採用模組化架構設計，主要組件包括：

1. 資料擷取層 - 從 AWS API 或模擬資料中擷取雲端資源資訊
2. 資料轉換層 - 將原始資料轉換為圖形資料模型
3. Neo4j 圖形資料庫 - 儲存節點和關係，支援複雜查詢
4. 分析引擎 - 執行安全性、故障衝擊和成本優化分析
5. 視覺化儀表板 - 提供互動式 Web 介面展示分析結果

### 9.2 系統連接流程



### 9.3 儀表板工作原理

#### 9.3.1 Flask 儀表板架構

系統使用 SimpleDashboard 類別提供基於 Flask 的輕量級 Web 儀表板：

- 資料載入 - 從 JSON 分析結果文件載入資料
- API 端點 - 提供 RESTful API 供前端調用
- 圖形視覺化 - 整合 vis-network 進行互動式圖形展示
- 即時分析 - 支援動態查詢和結果更新

#### 9.3.2 核心 API 端點

Listing 15: 儀表板 API 端點

```

1 @app.route('/api/data')      # 獲取所有分析數據
2 @app.route('/api/security')  # 獲取安全分析結果
3 @app.route('/api/failure')   # 獲取故障衝擊分析結果
4 @app.route('/api/cost')      # 獲取成本優化分析結果

```

#### 9.3.3 前端視覺化技術

- vis-network - 互動式圖形網路視覺化
- Chart.js - 統計圖表展示
- FontAwesome - 圖標系統
- 韻應式設計 - 支援多種螢幕尺寸

## 9.4 系統截圖展示



fig 1: 系統主頁面 - 顯示整體分析概覽

(a) 安全性分析結果

(b) 故障衝擊分析

(c) 成本優化分析

fig 2: 三大核心分析功能截圖 - 安全性分析、故障衝擊分析、成本優化分析

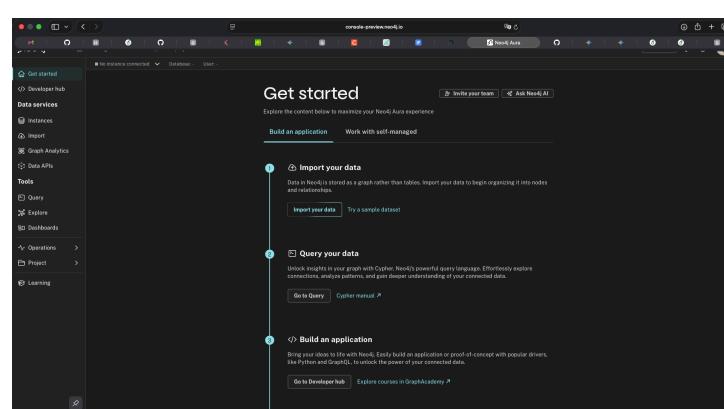


fig 3: Neo4j Aura 雲端資料庫 - 圖形資料儲存和查詢

## 9.5 Neo4j 查詢結果展示

本節展示在 Neo4j Aura 中執行的各種 Cypher 查詢結果，驗證系統的分析功能。

### 9.5.1 節點類型查詢結果

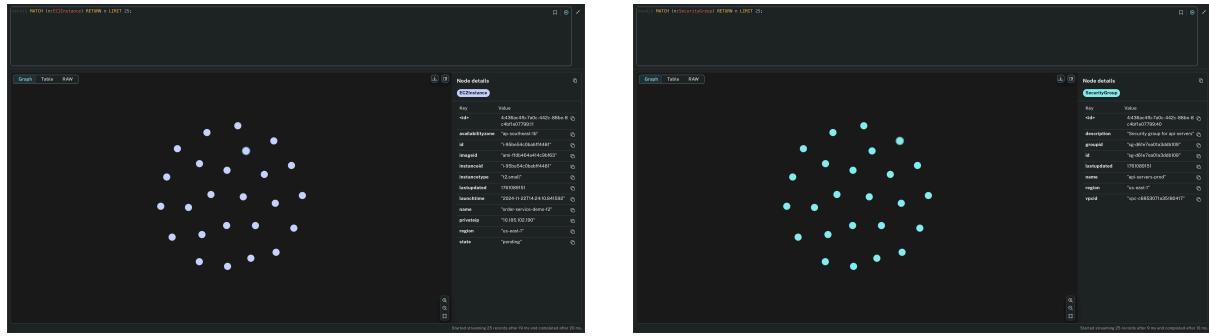


fig 4: 核心節點類型查詢結果

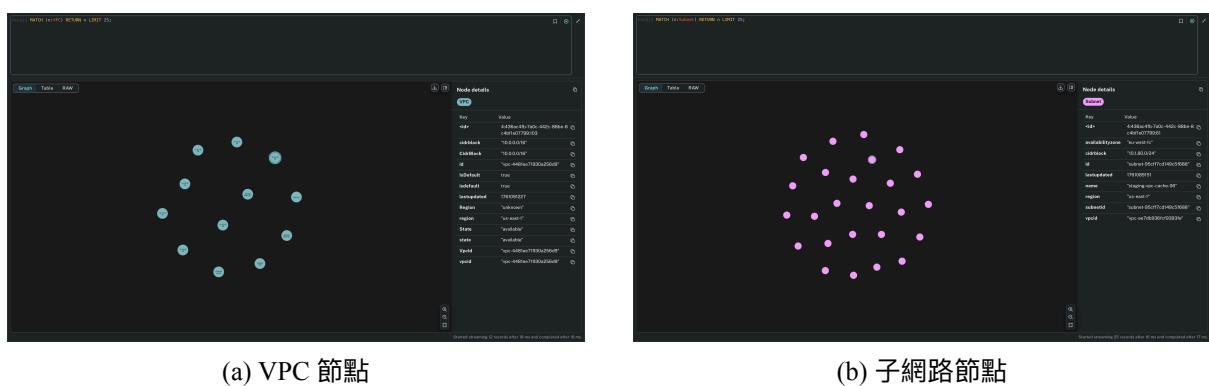


fig 5: 網路基礎設施節點

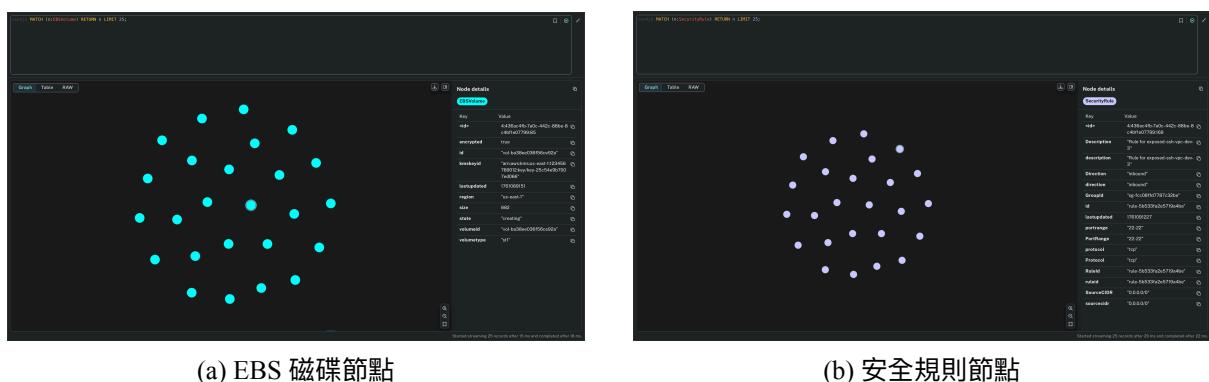


fig 6: 儲存和安全規則節點

### 9.5.2 關係類型查詢結果

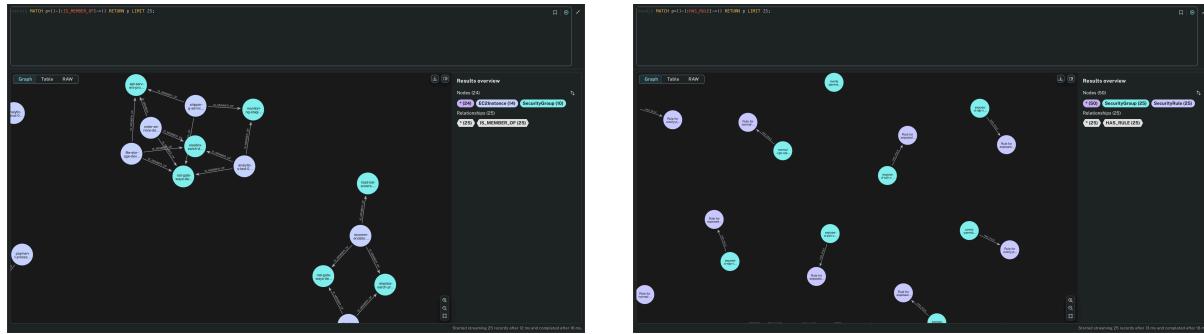


fig 7: 安全群組相關關係

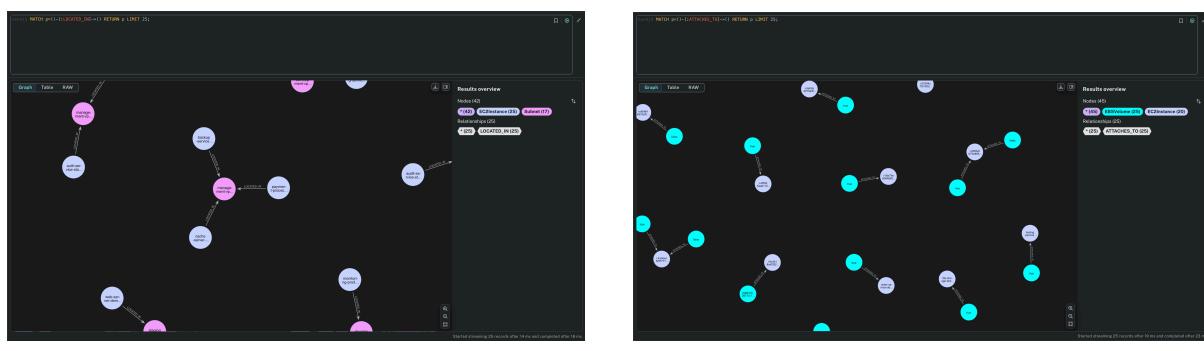


fig 8: 位置和掛載關係

### 9.5.3 安全性分析查詢結果

Security: Unencrypted EBS Volumes					
Region	State	SizeGiB	VolumeID	Region	State
us-east-1	in-use	341	vvol-702a14e4eab9b0d0	us-east-1	in-use
us-west-2	available	259	vvol-8af22ea1f51a4	us-east-1	in-use
us-west-2	available	93	vvol-1f11a1f5b7a4798	us-east-1	in-use
us-west-2	available	62	vvol-601aef12735a2	us-east-1	in-use
us-west-2	available	21	vvol-b425a15a2c02	us-east-1	in-use
us-west-2	available	81	vvol-36c02650a7664a	us-east-1	in-use

Security: Overly Permissive SG					
Protocol	PortRange	RuleDescription	SecurityGroupID	SecurityGroupName	Region
tcp	80-80	Rule for normal sg-0000000000000000	sg-0000000000000000	normal-sg-0000000000000000	us-east-1
tcp	0-443	Rule for https sg-0000000000000001	sg-0000000000000001	https-sg-0000000000000001	us-east-1
tcp	22-22	Rule for ssh sg-0000000000000002	sg-0000000000000002	ssh-sg-0000000000000002	us-east-1
tcp	3389-3389	Rule for RDP sg-0000000000000003	sg-0000000000000003	rdp-sg-0000000000000003	us-east-1
tcp	80-80	Rule for http sg-0000000000000004	sg-0000000000000004	http-sg-0000000000000004	us-east-1
tcp	0-65535	Rule for everyting sg-0000000000000005	sg-0000000000000005	everyting-sg-0000000000000005	us-east-1
tcp	22-22	Rule for ssh sg-0000000000000006	sg-0000000000000006	ssh-sg-0000000000000006	us-east-1
tcp	2389-2389	Rule for port 2389 sg-0000000000000007	sg-0000000000000007	port-2389-sg-0000000000000007	us-east-1

(a) 未加密 EBS 磁碟檢測

(b) 過度寬鬆的安全群組檢測

fig 9: 安全性分析查詢結果

### 9.5.4 成本優化分析查詢結果

The image displays two separate CloudWatch Metrics Insights query interfaces. Both queries are titled 'Cost' and involve filtering metrics related to AWS Lambda and EC2 instances.

**(a) 孤兒 EBS 磁碟檢測**

```

    MATCH :EBSVolume
    WHERE i:EC2Instance
    RETURN instanceid AS InstanceID, i.name AS InstanceName, i.region AS Region, i.launchtime AS LaunchTime
  
```

**(b) 未使用的安全群組檢測**

```

    MATCH :SecurityGroup
    WHERE i:EC2Instance
    RETURN securitygroupname AS SecurityGroupName, i.name AS InstanceName, i.region AS Region
  
```

fig 10: 成本優化分析查詢結果

This screenshot shows a single CloudWatch Metrics Insights query interface with the title 'Cost: Stopped EC2 Instances'. The query filters EC2 instances based on their state.

```

    MATCH (:EC2Instance)
    WHERE i.state = 'stopped'
    RETURN instanceid AS InstanceID, i.name AS InstanceName, i.region AS Region, iinstancetype AS InstanceType, i.launchtime AS LaunchTime
  
```

fig 11: 已停止的 EC2 實例檢測

### 9.5.5 故障衝擊分析查詢結果

The image shows two CloudWatch Metrics Insights query interfaces for fault impact analysis.

**(a) 關鍵節點識別**

```

    // Failure Impact: Critical Nodes
    MATCH (n:FailureImpact {critical: true})
    WITH n, COUNT(i) AS connection_count
    WHERE i:EC2Instance
    RETURN n.name AS NodeName, connection_count
  
```

**(b) 單點故障檢測**

```

    // Failure Impact: Single Point of Failure
    MATCH (n:FailureImpact {single_point_of_failure: true})
    WITH n, COUNT(i) AS connection_count
    WHERE i:EC2Instance
    RETURN n.name AS NodeName, connection_count
  
```

fig 12: 故障衝擊分析查詢結果

Failure Impact: Network Redundancy Analysis			
Query Settings		Table	
Natural language		Cypher	
			VpcId SubnetCount InstanceCount subnet_details
1 MATCH (vpc:VPC)			vpc-77769925fa0c22123 0 0 []
2 OPTIONAL MATCH (subnet:Subnet)-[:LOCATED_IN]-(vpc)			vpc-ae7db936fcf9393fe 0 0 []
3 OPTIONAL MATCH (instance:Ec2Instance)-[:LOCATED_IN]-(subnet)			vpc-c6683071a35180417 0 0 []
4 WITH vpc,			vpc-bfe9208d77145ba4b 0 0 []
5 collect(DISTINCT subnet) as subnets,			vpc-4e2382667ae2d581b 0 0 []
6 collect(DISTINCT instance) as instances			vpc-4481ee71930a256d9 0 0 []
7 RETURN			vpc-01421fe8f00821a84 0 0 []
8 vpc.vpcId as VpcId,			vpc-bc2594ec5c835a1fb 0 0 []
9 size(subnets) as SubnetCount,			null 0 0 []
10 size(instances) as InstanceCount,			null 0 0 []
11 UNWIND subnets as subnet,			
12 UNWIND instances as instance			
13 WHERE subnet.vpcId = vpc.vpcId			
14 AND instance.vpcId = vpc.vpcId			
15 AND instance.subnetId = subnet.id			
16 AND instance.availabilityZone = subnet.availabilityZone			
17 AND instance.blockDeviceMapping = subnet.blockDeviceMapping			
18 } AS subnet_details			
19 ORDER BY InstanceCount DESC			
Showing 1-10 of 12 results			
Recent			
// 8. Failure Impact: Network Redundancy Analysis (網路冗餘性分析)			
MATCH (vpc:VPC)			
OPTIONAL MATCH (subnet:Subnet)-[:LOCATED_IN]-(vpc)			
OPTIONAL MATCH (instance:Ec2Instance)-[:LOCATED_IN]-(subnet)			
WITH vpc,			
collect(DISTINCT subnet) as subnets,			
collect(DISTINCT instance) as instances			

fig 13: 網路冗餘性分析

## 10 參考資料

1. Neo4j Documentation: <https://neo4j.com/docs/>
2. Cypher Query Language: <https://neo4j.com/docs/cypher-manual/>
3. AWS Well-Architected Framework: <https://aws.amazon.com/architecture/well-architected/>
4. Cartography Project: <https://github.com/lyft/cartography>
5. 專案 GitHub 儲存庫: <https://github.com/itsYoga/cloud-infra-analysis>

—