CYBERSEC 2025 臺灣資安大會 | 4/15-4/17 臺北南港展覽二館

secure SOFTWARE & DEVSECOPS FORUM

SECURE SOFTWARE & DEVSECOPS FORUM

你的雲端開發環境值得更好的保護 - 雲端開發常見的迷思與風險

TEAM CYBERSECURITY

Archer Tsai

ASUSTek / Digital Security Center

Security Architecture Strategist

加映聯名場次

4/15

4/15 (二) 16:15 - 17:00　7F 701D　Exposure Management 論壇

老闆：我全都要。爭什麼，摻在一起做成情資

企業面臨的網路威脅日益複雜和多樣化。ASUS透過外部攻擊面管理（EASM）結合情資資源，提升企業的可視性與即時性，有效保障資訊安全。本次介紹將深入探討ASUS如何整合各種威脅情資，從多元化的資料來源中蒐集相關資訊，並通過自動化手段加強安全防護。...

陳星賀 (Hardy Chen)

許宗仁 (TJ Hsu)

4/16

4/16 (三) 16:15 - 16:45　4F 4C　Live Translation Session　SecOps 論壇

Zero to FIRST

我將和會眾分享申請加入FIRST（Forum of Incident Response and Security Teams）的過程和意義。隨著全球資訊安全威脅的日益嚴峻，華碩致力於為全球用戶與合作夥伴提供安全可靠的產品與服務，進一步強化資安韌性，並推動ESG永續發展。在這個過程中，我將解析如何利用SIM3 v2 interim Self Assessment Tool進行自我評估，全面了解CSIRT/PSIRT的成熟度，...

4/17

4/17 (四) 10:15 - 10:45　4F 4C　Live Translation Session　雲端安全論壇

Secure by Design 雲端安全管理架構設計

目前常見企業安全防護或是組織分工常以產品、資安治理標準框架的角度規劃，或以安全政策、資安認證、安全監控或是威脅情資為主，容易見樹不見林或是見林不見樹。卻忽略資訊系統或是網路設計的安全規劃才是強化體質正本清源的做法，本講題將以在趨勢科技與華碩電腦資安部門專職 18 年 Security Architect 的經驗出發，設計出一個Secure by Design 與 Security by...

Will Lin

4/17

4/17 (四) 12:40 - 13:10　4F 4C　Live Translation Session　Lunch Learning Session　雲端安全論壇

不時默默地以 Admin 操作的雲端管理員

本次議程將以中立角度，從甲方與乙方的觀點出發，與會眾共同探討使用雲服務時的管理與技術風險考量。議程以企業管理資源有限為前提，針對希望入門雲端安全的資安從業者，深入探討在實務操作中面臨的各種挑戰與經驗。我們將介紹常見的雲端技術實務問題及應對方法，並透過實際案例分析不同情境的使用風險。主題涵蓋分散與集中式雲端管理經驗、身份與存取管...

黃星評 (Kuro Huang)

蔡幸育 (Archer Tsai)

4/17

4/17 (四) 14:00 - 14:30　7F 701F　Secure Software & DevSecOps 論壇

你的雲端開發環境值得更好的保護

傳統系統開發上，只有系統正式上線前，才會需要經過完善的掃描和保護，也只有正式系統的安全性會受到重視。

但在公有雲進行開發時，其實從資源建立的那一刻開始，系統就已經開始面對外在的種種挑戰。

而這些POC、開發、測試或Demo環境被攻破後，可能帶來的危害其實也遠比想像中來的大。...

Archer Tsai

語音辨識 → 系統整合 → 雲端開發 → DevSecOps → 資安架構師

核心技術	系統開發	DevSecOps	資安架構師
• 效果與運算速度至上 • 不在乎週邊系統架構	• 系統準時上線 • 功能與架構符合規格	• 系統穩定不出事 • 順利通過資安認證	• 系統設計合乎規範 • 系統不容易被攻破

雲端時代帶來的改變

- 硬體設定流程快到以分鐘計
- 運算資源原生在雲端
- 開發流程從機房搬到雲端
- **那我們的開發思維上雲了嗎?**

講在開始之前

- 資安就是花費、便利性與安全性的妥協
- 沒有絕對的安全，沒有完美的資安防護工具
- 小心配置你的起司，避免讓洞疊在一起

常見迷思

雲端環境有通過各種資安認證，應該是安全的吧?

雲端環境有通過各種資安認證，應該是安全的吧

- 資安認證的是公有雲廠商提供的服務合規性
  - 使用者怎麼用，會決定最終產品安全性
  - 一台汽車通過再多安規，行車安全還是控制在駕駛手上

- 很多雲端服務使用者或設定流程只顧到方便使用，而忽略安全性

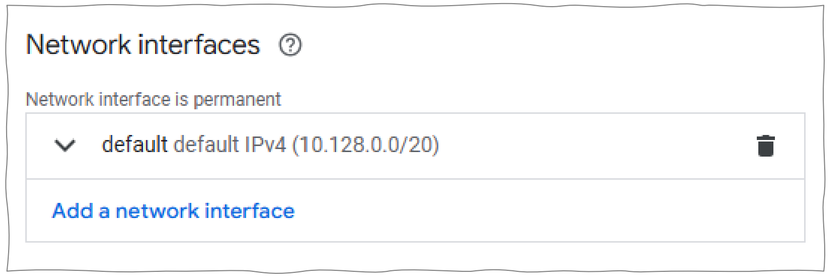
雲端環境有通過各種資安認證，應該是安全的吧

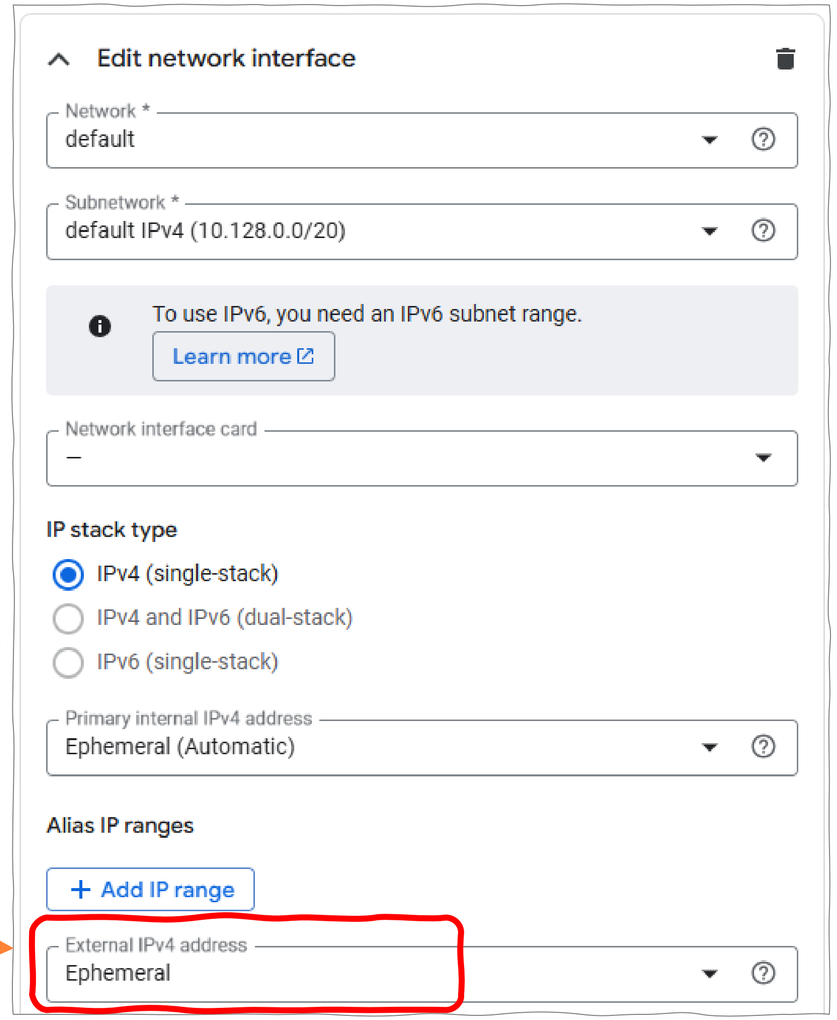
Insecure

~~Secure~~ by Default

Insecure

Create a VM on GCP



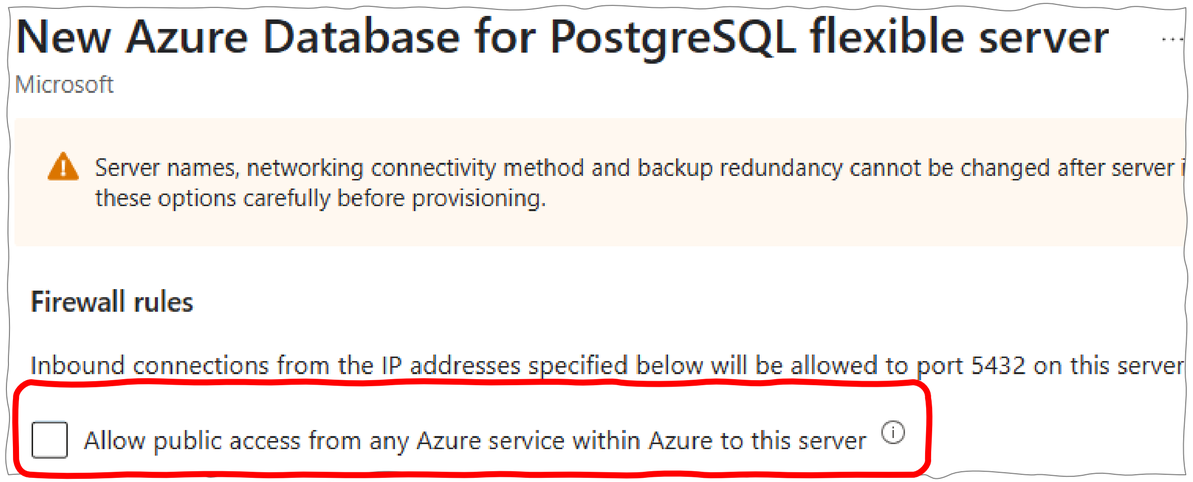


一直按下一步，就會得到不安全的環境

一不小心就直接對外開放了

雲端環境有通過各種資安認證，應該是安全的吧

Insecure

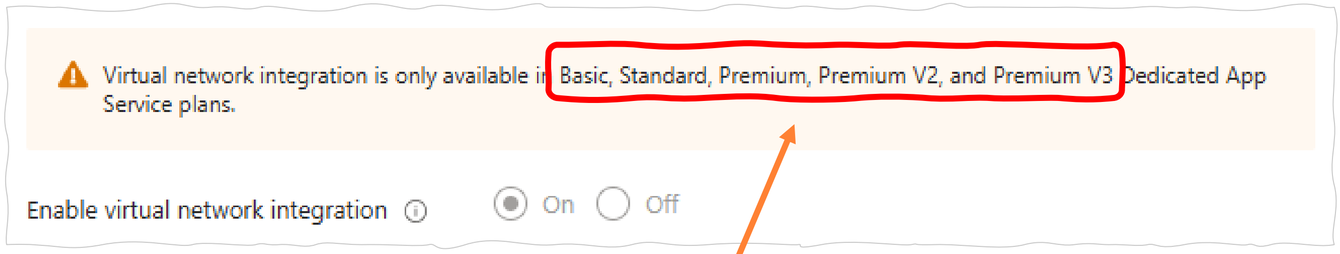


勾起來，全世界的Azure 服務都能連到你家

雲端環境有通過各種資安認證，應該是安全的吧

Insecure

- 部分Free/Basic方案防護功能有限
  - 只能公開對外
  - 無法與虛擬網路整合
  - 無法使用安全設定，如備份等



Free Tier 沒人權

雲端環境有通過各種資安認證，應該是安全的吧

Insecure

- 仔細了解並確認每個設定值
- 妥善設定對外開放範圍，如 Public IP, 防火牆
- 仔細設定人員存取權限
- 評估使用防毒、防駭等監控軟體

Secure

常見迷思

開發中網站的網址只有自己知道，應該是安全的吧

開發中網站的網址只有自己知道，應該是安全的吧

Insecure

- 駭客已經是24*7全年無休在自動化掃描網路環境
  ✓甚至連入侵都自動化了

- 搜尋引擎的Robot也是全年無休自動化掃瞄
  ✓可設Robots.txt，但防君子不防小人

- 公開在Internet上的東西，
  只有發現時間的早晚，沒有不被知道的

開發中網站的網址只有自己知道，應該是安全的吧

- 開發中站台對外曝光的影響
  - 商業機密提前曝光
  - 競爭對手搶先上市，或提出反制
  - 使用者誤認是正式資訊，提出客訴

開發中網站的網址只有自己知道，應該是安全的吧

- 不要有Public IP，或利用防火牆限制來源IP
- 利用登入機制，避免未經授權存取
- 開發中的站台不要圖方便掛在正式站台下

常見迷思

只是從正式資料庫讀資料沒有寫入，應該是安全的吧

只從正式資料庫讀資料沒有寫入，應該是安全的吧

- 開發中站台容易有各種漏洞，造成資料外洩
- 人為疏失可能導致正式資料損毀
- 真實資料外洩的後果可能超乎你的想像
  - 臺灣個資法最高罰款1500萬新台幣，可連續處罰
  - 歐洲GDPR最高可罰2000萬歐元 (約7億新台幣)，或4%全球營收
  - 2014 YAHOO資料外洩，最終導致上億美元賠償，加上收購價下跌2.5億美元

只從正式資料庫讀資料沒有寫入，應該是安全的吧

- 真實業務資料避免放在開發環境
- 真實業務資料應該有妥善權限管制，維運與開發權限應該拆開
- 正式環境與開發環境要有網路隔離，以免非預期存取

常見迷思

用SSH Key登入主機或GitHub，應該是安全的吧

用SSH Key登入主機或GitHub，應該是安全的吧

Insecure

- 未經保護的SSH Key，相當於把密碼貼在螢幕上
- 任何人只要取得金鑰，就可以存取系統

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/arc
Enter passphrase (empty for no passphrase):
```

大多數工程師都會自動跳過的選項

用SSH Key登入主機或GitHub，應該是安全的吧

- **在SSH Key加上passphrase**
  - ✓**增加盜用難度**
  - ✓**避免被自動化惡意軟體使用**
  - ✓**達成類似雙因子認證效果**
  - ✓**增加駭客橫向移動難度**

常見迷思

API Key寫在檔案裡只要不上傳，應該就是安全的吧

API Key寫在檔案裡只要不上傳，應該就是安全的吧

- 明碼存放的機密資訊就是一種風險
- 容易因操作失誤上傳到codebase
- 能碰到開發環境的人可能比你預期的多
  - 雲端管理員
  - 共用環境的開發者
  - CNAPP等監控Agent

API Key寫在檔案裡只要不上傳，應該就是安全的吧

- 永遠不要把API Key或密碼寫在檔案裡
- 利用金鑰庫保存機敏資訊，要用再拿
  - Azure KeyVault
  - AWS/GCP KMS
  - 自建HashiVault

常見迷思

團隊共用帳號登入系統開發，應該是安全的吧

團隊共用帳號登入系統開發，應該是安全的吧

- 多一個人持有登入資訊就多一分洩漏風險
- 無法明確劃分責任
- 事故發生時難以追蹤洩漏點
- 人員離職或轉調時難以收回權限

團隊共用帳號登入系統開發，應該是安全的吧

- 每個人都有獨立帳號，並避免用Local Account
- 利用SSO登入，並套用MFA機制
- 定期確認環境中有效帳號
- 必須共用帳號時需要有明確控管程序

常見迷思

免費開發環境又不花錢，用完放著應該是安全的吧

免費開發環境又不花錢，用完放著應該是安全的吧

- 費用不該是使用雲端的唯一考量
- 被忽視的閒置環境，被攻破也沒人知道
- 隨時有新的資安漏洞，沒有更新維護的系統容易出事

免費開發環境又不花錢，用完放著應該是安全的吧

- 不用的開發環境就關機或刪除
- 善用IaC和CVS以快速生成或復原開發狀態

常見迷思

為了除錯或開發方便先開個後門，應該是安全的吧

為了除錯或開發方便先開個後門，應該是安全的吧

- 開發留下的後門或已淘汰功能常常是重大漏洞來源
  - 萬能密碼
  - https://***.com/?debug

- 一時疏忽流到正式環境可能造成災難

為了除錯或開發方便先開個後門，應該是安全的吧

- 以正規方式進行開發除錯，如透過固定跳板機
- 利用Github等CVS版控系統管理程式碼，明確分隔開發中程式與正式版本
- 落實程式碼合併前的Code Review
- 功能取消或淘汰時，同時移除所有相關程式和端點

常見迷思

開發環境而已不用特別做系統更新

開發環境而已不用特別做系統更新

- 駭客其實不在乎是開發或正式環境
- 新的系統漏洞隨時可能被發現
- 今天的安全環境明天不一定還安全

開發環境而已不用特別做系統更新

Insecure

- 開發環境也需要套用安全更新
- 善用Serverless/PaaS 降低維護成本
- 不用的開發環境就關機或刪除
- 妥善隔離開發環境與正式環境

Secure

常見迷思

大家都在用的開源套件，
應該是安全的吧

大家都在用的開源套件，應該是安全的吧

- 安全的套件用不安全的方式使用就是不安全的
- 使用多年的知名套件也可能突然爆出漏洞
  - Log4j v2.0 在2014年發佈，2021年才爆出嚴重資安漏洞

大家都在用的開源套件，應該是安全的吧

- 使用開源套件要詳讀使用手冊
- 使用開源套件要隨時留意安全更新

對資安來說，沒有開發或正式環境的區別

Best Practice

Best Practice

- 開發環境不要直接暴露在公網
  - ✓ 設定跳板機並以防火牆保護，不使用跳板機時關機
- 確實隔離開發環境與正式環境
- 開發與正式環境使用不同 API Key
- 開發用API Key設定較短期限與較低使用額度
- 限制API Key呼叫來源IP

Best Practice

- 以IaC (Infrastructure as Code)建立及管理開發環境

  ✓利用程式碼管理雲端基礎設施配置

  ✓開發環境統一且容易管控

  ✓容易確保環境安全

CYBERSEC 2025
臺灣資安大會

THANK YOU