CYBERSEC 2024 臺灣資安大會 | 5/14 Tue – 5/16 Thu 臺北南港展覽二館

Generative Future

DevSecOps Forum

敏捷開發生命週期下之技術託管與風險掌握

林于翔(AL)

副總經理

資誠智能風險管理諮詢公司

alvin.lin@pwc.com

Agenda

1. 現代化系統架構轉型與開發趨勢 04
2. AI運用與風險掌握 14
3. DevSecOps託管的產業趨勢 23

現代化系統架構轉型與開發趨勢

現代化系統架構轉型的困難與關鍵

需要投入大量資金

- ✓ 新架構的開發、設計和部署等成本
- ✓ 需要進行大量的測試和驗證

長期的計畫和實施過程

- ✓ 基於新系統架構，改造既有系統，需重新分析、設計、測試和驗證
- ✓ 新舊系統Parallel Run

攜手合作
共同面對和克服難題

敏捷思維、橫向合作

- ✓ 確保員工能夠適應新的工作流程
- ✓ 打破既有部門、團隊疆界

轉型過程衍生的風險

- ✓ 轉型過程中，可能會出現漏洞和風險，需要針對性地進行風險管理和減緩措施。

新世代SDLC著重解決產業實務上的痛

Generative Future

基礎架構現代化

- 應考量高可用性、高可靠性、彈性擴展、備援機制、營運不中斷等系統與環境架構；
- 配合應用系統短、中、長期發展目標與雲端發展策略，擬定基礎架構規劃重點。

Scalability

應用系統架構現代化

- 建立現代化應用系統架構與管理標準，並整合企業既有資訊標準規範；
- 舊系統轉入新架構之轉換規劃與風險掌握；
- 新舊系統架構檢視之機制建立，確保公司未來有持續自行檢視能力。

Standardization

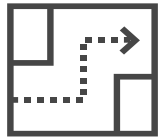
作業流程現代化

- 敏捷開發部署的作業流程設計；
- 符合企業相應主管機關所訂定之法令規範。

Speed

Estimation

資訊人才現代化



- 因應現代化架構調整後，人員角色的轉換與技能的養成；
- 系統現代化轉換期間，資訊人力資源配置、分工及職能轉型訓練。

Effective

化被動為主動的開發生命週期管理

Speed | Scalability | Standardization | Effective | Estimation

Strategic Decision Making

資源管理描述化

作業流程自動化

AI-Driven SDLC

Dev Sec Ops

開發測試框架統一化

共用資源集中化

All about Risk

pwc Responsible AI Risk

All about Maturity

NIST SSDF

IEC 62443-4-1

SAMM Software Assurance Maturity Model

(BSIMM) Building Security In Maturity Model

SDLC流程上的標準化與資安強化(以DevSecOps標準化為例)

Generative Future

1 掌握當前狀態

瞭解DevSecOps管道和實作的當前狀態，包括:

- ✓ 政策、指導方針、程式和標準
- ✓ 應用程式和軟體的一般使用情況
- ✓ 實現公共/標準安全特性
- ✓ 用於安全編碼和自動測試的現有資源、工具和流程
- ✓ 面臨的挑戰

2 設計和實施計畫

通過人員、流程和技術來回顧DevSecOps方法的設計和DevSecOps的目標狀態，例如:

- ✓ 根據矩陣計畫設計角色和職責
- ✓ 計劃和設計DevSecOps持續整合/持續交付/持續安全管道，基礎設施自動化，自動化測試和持續監控以及KPI/指標

3 實現DevSecOps架構

實現DevSecOps架構，並準備試運行如下:

- ✓ 原始程式碼遷移到新的原始程式碼管理(SCM)
- ✓ CI/CD pipeline整合
- ✓ 在CI/CD pipeline上自動構建整合
- ✓ 自動部署實現
- ✓ 自動測試工具整合
- ✓ 安全掃描工具整合
- ✓ 參照相關行業標準和慣例

4 培訓及試運行

為試運行專案團隊提供DevSecOps導入培訓，包括:

- ✓ 安全意識
- ✓ DevSecOps的採用
- ✓ 在sprint中觀察DevSecOps的試運行方法。CI/CD管道包括DEV, UAT, PRD和DR環境

5 試運行後審查

在每個sprint結束時訪談並獲得試點團隊的回饋。

回顧DevSecOps的KPI，並在sprint中評估試運行的性能和成熟度。

安全整合左移

持續交付

持續整合

自動化測試

持續發佈

持續監控

文化的破壞與建立–IaC在敏捷開發上的流程重建

Generative Future

Speed | Scalability | Standardization

後期

Product Requirements (U/D) | Development (D) | Test (U/Q) | Infrastructure Requirements (D/S) | Infrastructure Builds (S) | SAST / DAST (Q) | Deployment (D) | Hardening (Q)

SAMPLE

早期

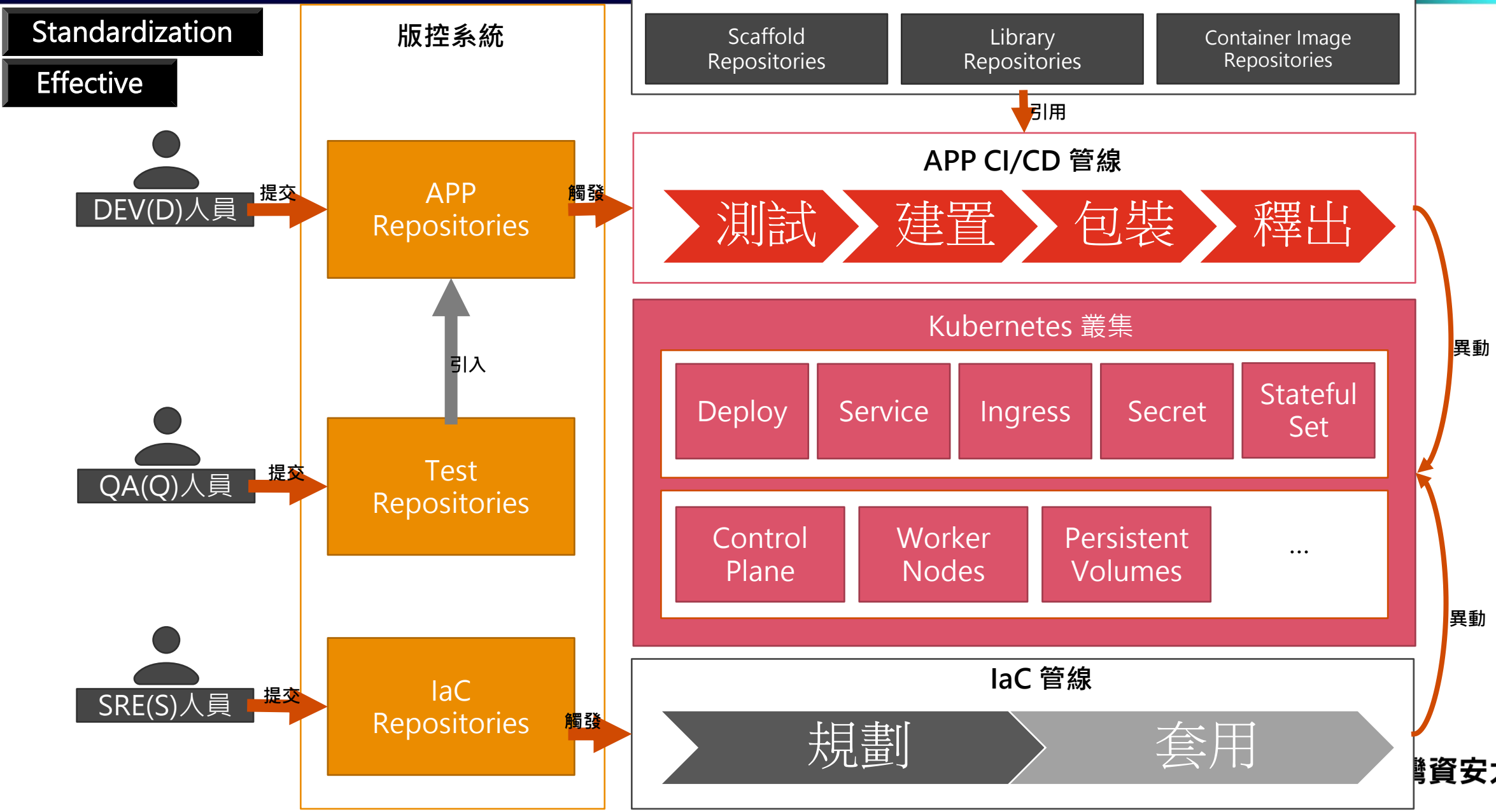
Product Requirements (U/D) | Development (D) | Deployment w/ CICD (D) | SAST / DAST / Hardening (Q) | Test (U/Q)

Infrastructure Requirements (D/S) | Infrastructure Builds w/ IaC (S)

後期

Deployment (D)

IaC版控標準化實現


Standardization
Effective
版控系統
成品庫
Scaffold Repositories
Library Repositories
Container Image Repositories
引用
APP CI/CD 管線
測試
建置
包裝
釋出
DEV(D)人員
提交
APP Repositories
觸發
引入
QA(Q)人員
提交
Test Repositories
Kubernetes 叢集
Deploy
Service
Ingress
Secret
Stateful Set
Control Plane
Worker Nodes
Persistent Volumes
...
異動
異動
SRE(S)人員
提交
IaC Repositories
觸發
IaC 管線
規劃
套用

自動化模式: 描述模式 & 命令模式

Effective | Standardization

Declarative (描述模式)

表達我們想要什麼

跟廚師說: 我想要一份沙拉

我想要一台 Oracle DB

Imperative (命令模式)

表達我們如何得到它

跟廚師說: 請給我生菜、番茄和醬汁

先安裝 Server, 再安裝 Oracle 需要的 Dependency, 再安裝 Oracle, 調整參數...

不如我們寫成腳本?

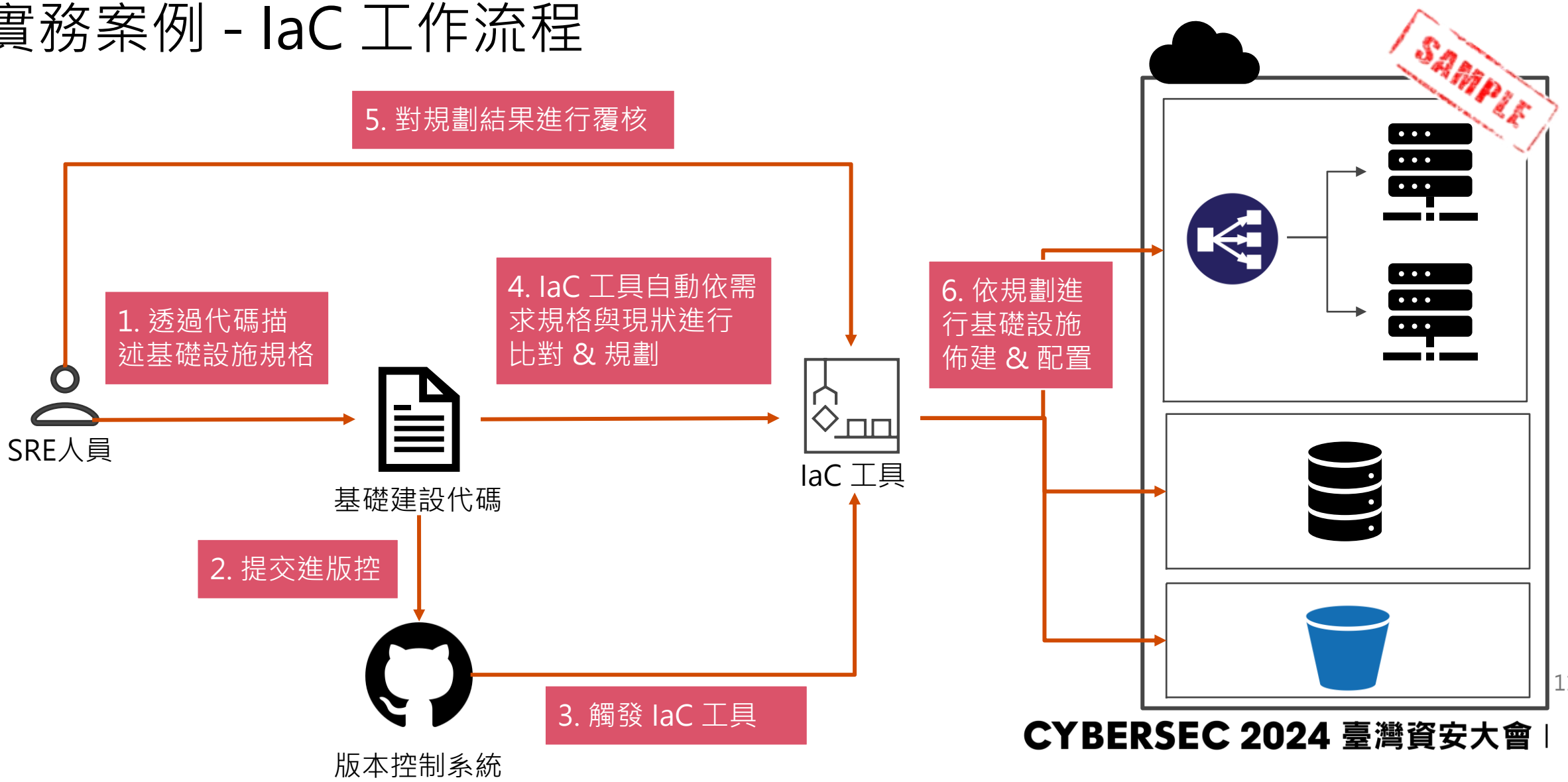
Idempotence (冪等性)

同一份內容，
在任何的系統起始狀態下，
無論執行幾次結果都是相同的

實務案例 - IaC 工作流程

Speed | Scalability | Standardization

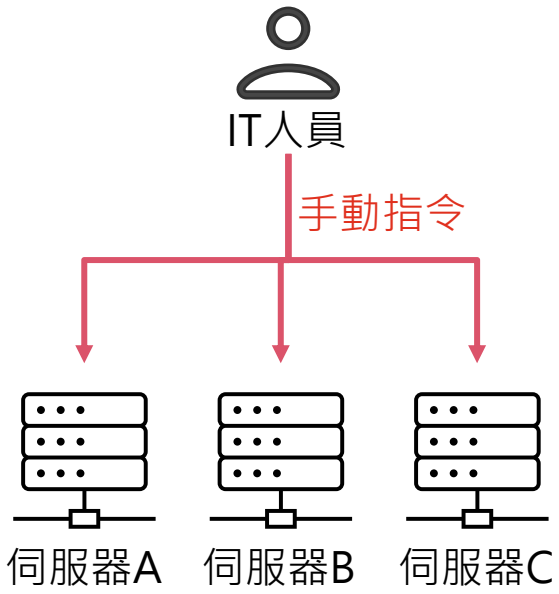
實務案例 - IaC 工作流程



傳統基礎設施建構與 IaC 的效益考量

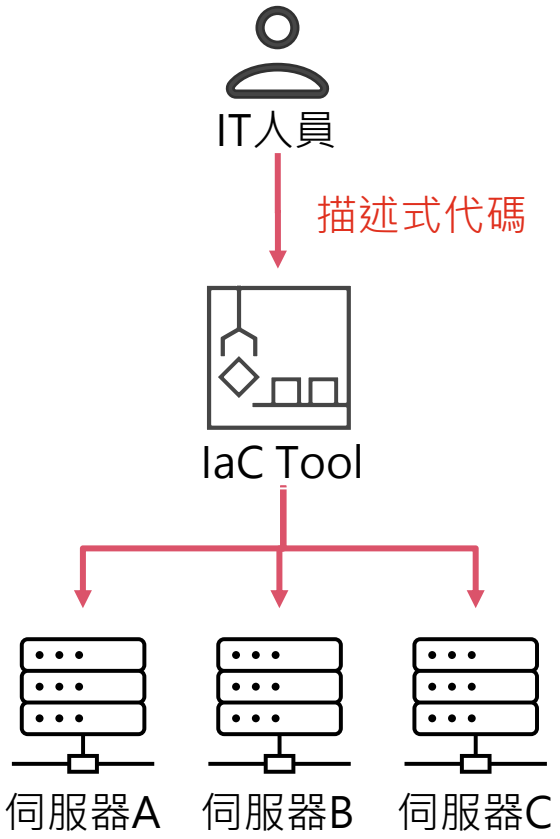
Effective | Standardization | Estimation

傳統基礎設施 (手動設定)



多台不一致
時間消耗
風險高

Iac (自動設定)



多台一致
效率高
可追朔

多維度架構下的轉型構面

Generative Future

Top to Down Standardization

基礎建設進入生命週期

Infrastructure as Code

Dev. to Dep. with full SDLC

ANSIBLE Terraform CHEF puppet

人員	面向	資源管理描述化	作業流程自動化	開發測試框架統一化	共用資源集中化
SRE	基礎建設	• IaC 版控 • 設定管理 (CM) • Vault 管理	• IaC 資源規劃與佈建 • Auto Scaling • 監控與告警	• IaC 範式	• 資料視覺化 • API Gateway
DEV	軟體架構	• CI/CD 過程 • 容器建置過程 • 部署資源請求	• Scaffolding • Code-First Migration • CI/CD	• AOP 分層架構 • ORM、DTO建模範式 • SSO、IAM • Flow Engine • ETL • 前後端分離 • OpenAPI Spec • 報表	• Logging • SSO、IAM • 資料視覺化 (BI) • SCM • Artifact
QA	品質確保	• Test Case 版控 • Test Data 版控	• UI Test (E2E) • API Test (IT) • Unit Test (UT)	• 工具輔助測試產生 • Data Driven	• Issue Tracking

AI運用與風險掌握

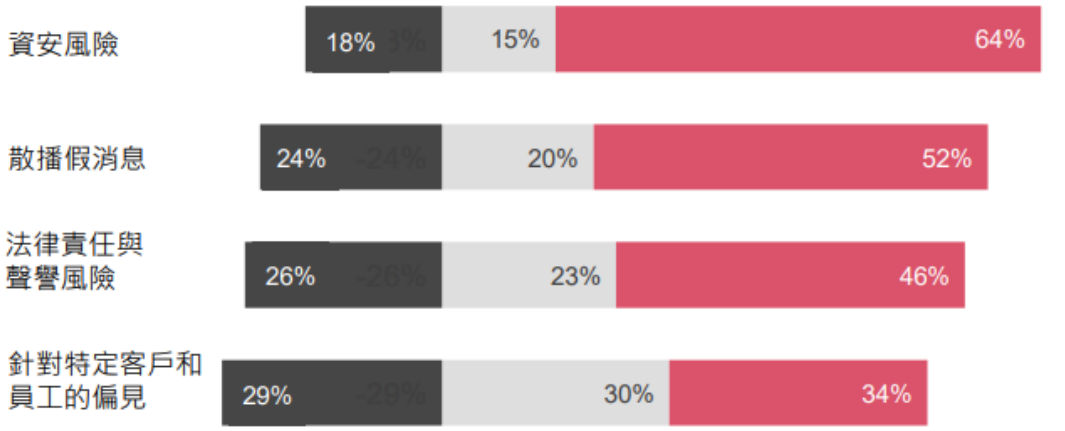
全球與臺灣CEO均關注生成式AI風險

Generative Future

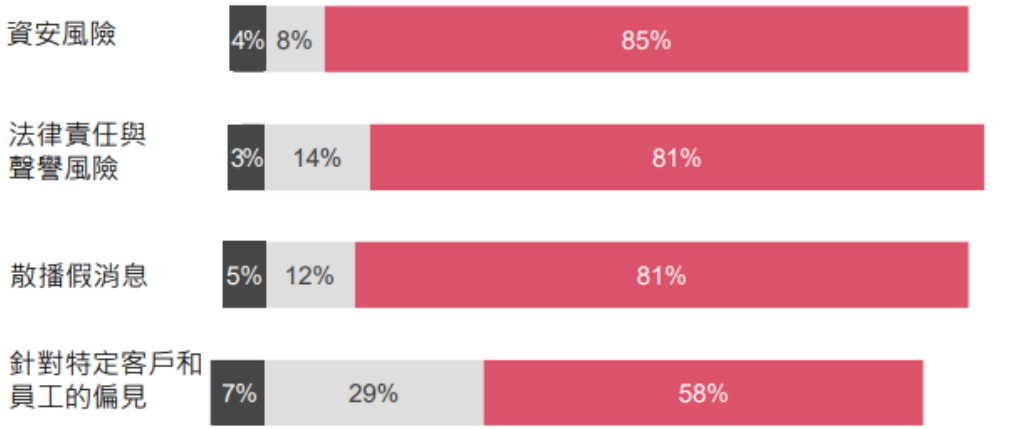
對於生成式AI的風險，同意或不同意的程度為何？ 無意見 不同意 同意

2024臺灣企業領袖調查

全球整體



臺灣整體



Source | PwC Taiwan 2024臺灣企業領袖調查、PwC 27th Annual Global CEO Survey

Base | 2023全球整體CEOs=4,410, 2024全球整體CEOs=4,702；2023臺灣整體CEOs=216, 2024臺灣整體CEOs=212。

企業需要評估和管理，所遭遇到的人工智慧風險

Generative Future

根據行業環境和落實的人工智慧的性質，風險的影響和可能性會有所不同，這些風險可能包括：

效能

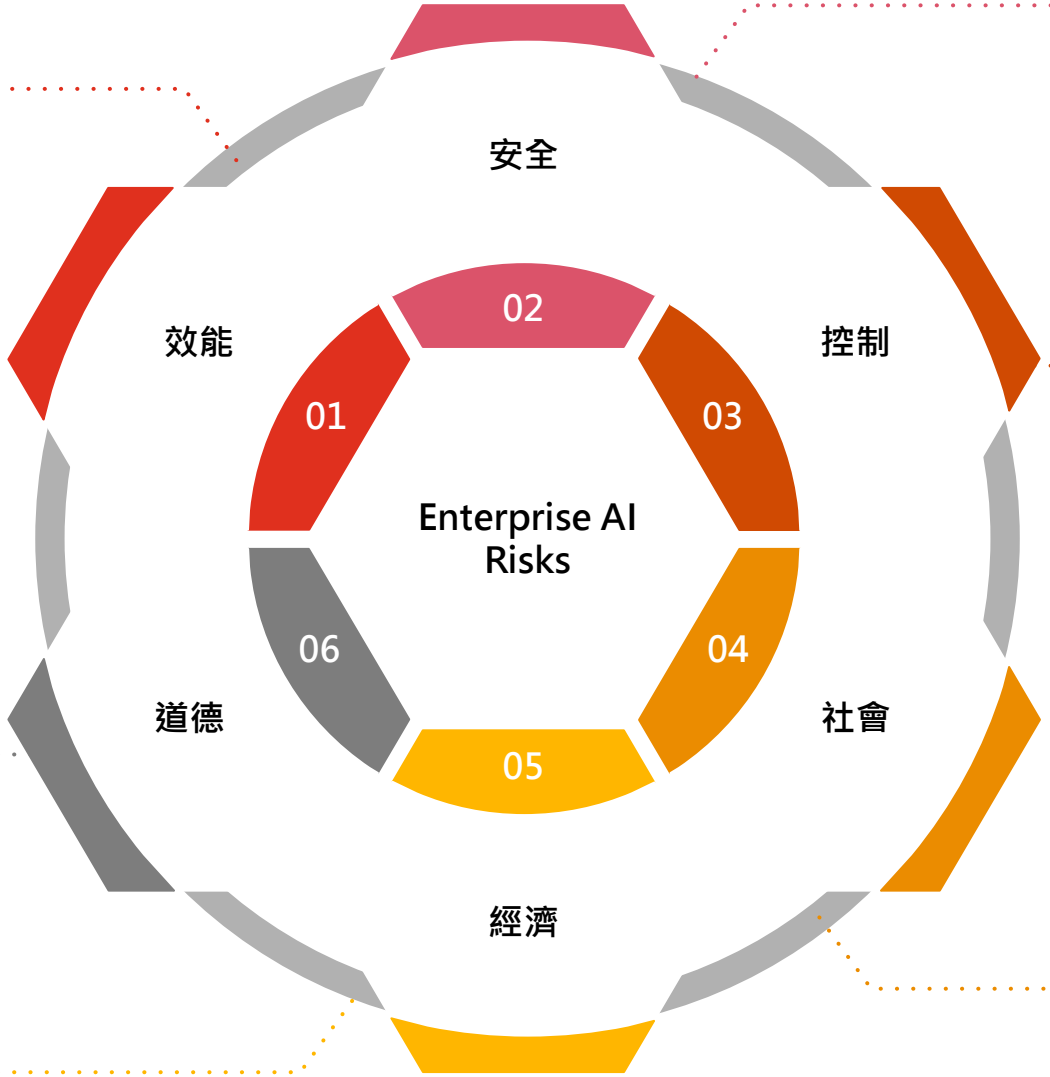
錯誤風險
偏差風險
不透明風險
效能不穩定風險

道德

缺乏價值觀
價值調整
目標風險

經濟

工作取代
責任風險



安全

進階持續性攻擊風險
網路入侵風險
隱私風險
開放原始碼風險

控制

流程中缺乏人工介入判斷
無法控制及檢測流程

社會

名譽身望風險
歧視風險

人工智慧對於攻擊者和防禦者的潛在好處及限制

Generative Future

世界各地

未針對特定產業

概要

自從人工智慧聊天機器人出現以來，人工智慧一直處於在網路安全中討論的重點 - 不論是從攻擊者或是防禦者的角度出發都是。

PwC 資安團隊分析和評估該技術，可能對威脅領域及防禦人工智慧攻擊造成的影響，並觀察到了下列現象：

- 人工智慧可能會降低駭客進入的門檻，這可能會增加攻擊的數量及成功率，特別是社交工程和網路釣魚活動。
- 與仰賴人工智慧解決方案的第三方相比，傳統的安全解決方案(例如: 端點偵測及回應)是目前較實用的防禦解決方案。

建議處置方式

預防

- 通過確保只有必要的服務可以公開存取並實施強大的外圍邊界防禦，以評估及減少攻擊點的暴露。
- 定期對可外部連線的資產進行弱點掃描，以發現漏洞或是錯誤配置。
- 如果實施開放原始碼的AI工具，需要考慮限制和潛在風險(例如資料隱私)，並確保關鍵資料不被曝露。

偵測

- 確保實施一個強大的入侵偵測系統，以檢測、監控和限制潛在惡意連線流量。
- 利用人工智慧和機器學習，模擬現有病毒株的惡意軟體變種，進行惡意軟體特徵碼的改進與檢測。

AI 風險治理不會只是一個時間點的查核

Generative Future

AI 風險治理的基礎是一個 end-to-end 的企業治理框架
重點關注於整體識別出的營運風險和相應的控制，
伴隨著組織的 AI 營運策略 - 從高層企業策略往下延伸至系統開發。

DevSecOps託管的產業趨勢

企業要解決的問題是什麼?

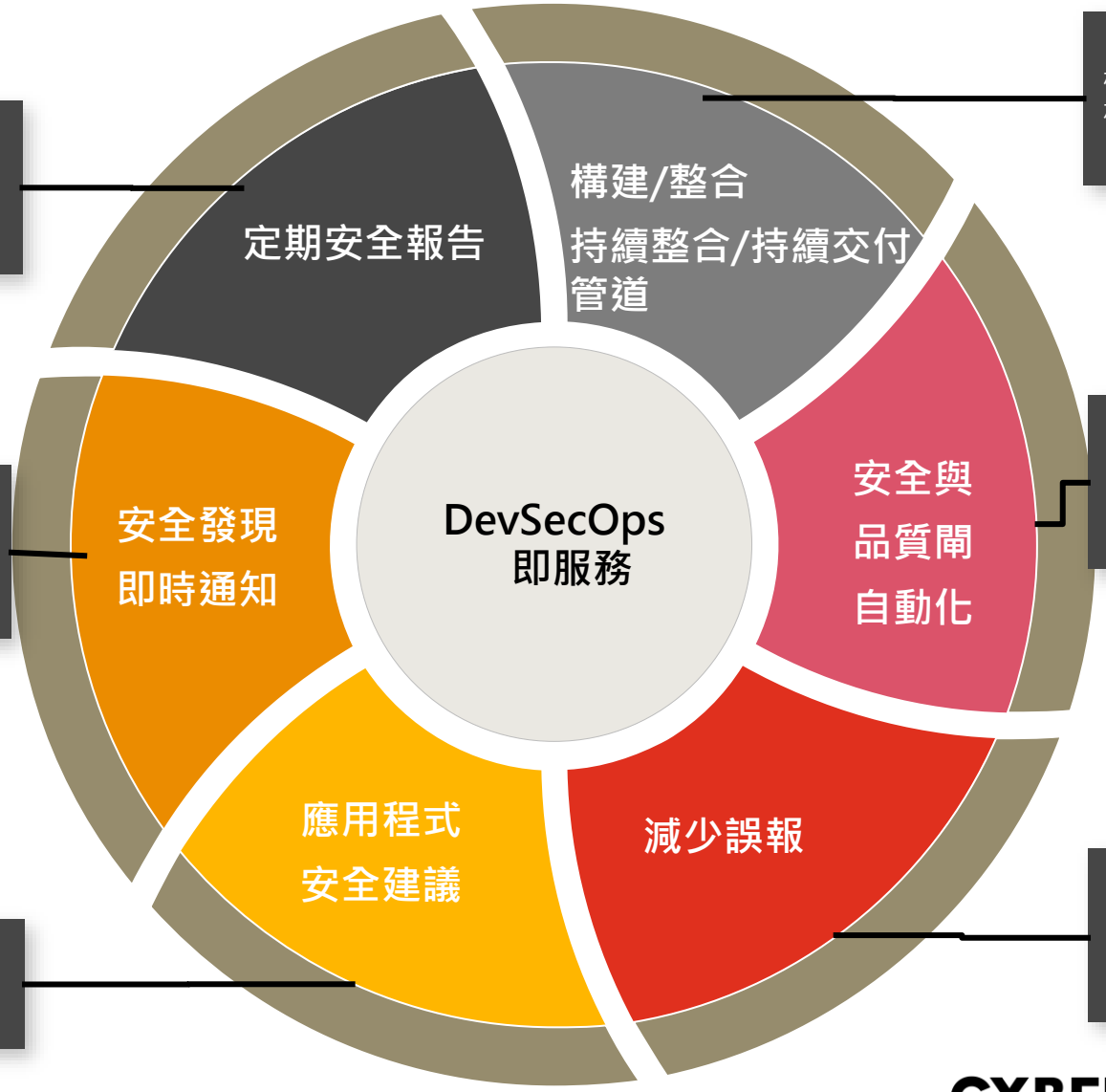
產業常見問題:

- 缺乏具備專門知識的人力
- 缺乏維護管道的資源和能力
- 購買不同工具許可證的巨大成本
- 應用程式開發中的弱點會導致安全問題

企業尋求資源:

- 提供技術專家
- 解放IT和安全團隊，讓他們專注於提高業務安全性
- 說明提高IT功能效率並節省成本
- 在早期階段檢測安全問題並修復潛在問題/違規行為

產業對於DevSecOps託管的期望是什麼？



為技術團隊提供每週報告，為管理層提供每月總結，從而全面瞭解應用程式的安全狀況

構建/整合現有的持續整合/持續交付管道，確保一個順利和高效的開發過程

利用Jira和電子郵件作為管道即時通知安全發現

整合自動化的安全工具和品質閘，識別漏洞並保持高應用標準

積極地與系統管理者溝通，提供量身定制的應用程式安全建議和指導

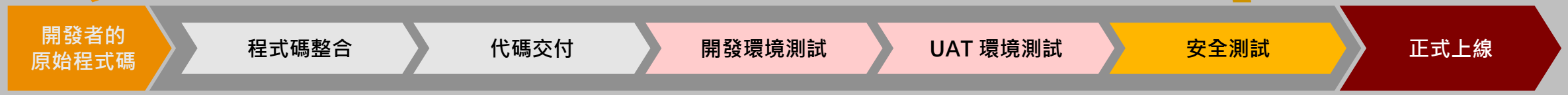
完善自動掃描功能，最大限度地減少誤報，提供準確和可操作的見解

實務案例 – 在應用DevSecOps託管前後的變化

Generative Future

應用DevSecOps託管之前...

- 資安團隊需要自行安裝掃描工具
- 漏洞掃描警報必須由資安團隊手動準備，然後請求開發團隊在上線之前修復已識別的問題



應用DevSecOps託管之後...

- 採用成熟構建的DevSecOps pipeline，與各種掃描工具整合，使團隊能夠簡化軟體發展和交付週期，同時整合傳統的IT功能，以提高整體執行效能與降低成本

託管安全性掃描以滿足合規性需求，以便團隊可以更多地關注開發

共同自助平臺可即時存取以查看掃描結果和修復