雲原生(AZURE)公務系統
WELL-ARCHITECTED FRAMEWORK
文件

摘要

是一組以質量為導向的原則、架構決策點，以及檢閱旨在協助解決方案架構設計人員為其工作負載建置技術基礎的工具。

目錄

API Portal on Azure Cloud

壹、 簡介

在 Azure Cloud 上，雲原生和容器化架構的系統設計旨在充分利用平台的彈性、高效運行能力和現代化開發工具，以滿足動態業務需求。結合 Azure 的廣泛服務和全球基礎設施，系統開發者能夠設計出能高效響應使用者請求、處理大量資料流並適應長期演變的應用程式。
此設計主要專注於：

A. 高效管理和最佳化 API 響應。
B. 資料庫系統的穩定性和可擴展性。
C. 確保應用和資料的安全性。
D. 實現資源高效使用和可持續性。

總結，本架構依據「雲原生公務系統研究設計案」需求書，設計 Azure Well-Architected Framework（WAF），需要從 Azure WAF 的五個核心支柱進行規劃，並結合需求書的背景需求和公務系統的特性。以下是詳細的規劃建議：

1. 可靠性（Reliability）
   設計目標： 確保 API Portal 能夠在公務系統高可用性需求下，持續提供服務，即使在故障或負載突增的情況下也不會影響運作。
2. 安全性（Security）
   設計目標： 保護系統免受未授權訪問、數據洩露和潛在攻擊，滿足公務系統的資訊安全規範。
3. 成本最佳化（Cost Optimization）
   設計目標： 控制系統建置與運營成本，確保在提供穩定服務的同時實現成本效益最大化。
4. 性能效率（Performance Efficiency）
   設計目標： 提供低延遲、高效能的 API 平台，滿足使用者操作體驗需求。
5. 卓越運行（Operational Excellence）
   設計目標： 實現 API 的自動化管理、持續部署和運行穩定性。

這樣的設計既滿足公務系統 API Portal 的高效、安全及穩定性要求，也最大化 Azure 資源的效益與性能。

貳、 目的

目的概述：

一、 高效：

1. 利用 Azure 服務實現快速的 API 響應與資料存取，包括針對流量優化的 API Gateway 和數據快取技術。
2. 設計具擴展性和容錯性的資料庫架構，如分布式資料處理與分片。

二、 可靠：

1. 構建具備高可用性、多區域備援和快速恢復的架構，確保即便在高峰期或災難情境下依然穩定運行。
2. 採用監控工具與自動化修復功能，快速檢測與處理異常。

三、 安全：

1. 保障應用程式 API 和資料的完整性，使用 Azure 的多層防護功能（API 管理、機密存儲、身份驗證與授權）。
2. 預防數據洩露，確保全過程符合業務或法規（如 GDPR）。

四、 可持續性：

1. 透過自動化資源調配優化運算效能並降低不必要的能源消耗。
2. 利用 Azure 可持續性工具追蹤和減少系統的碳足跡。

這些設計基準和目標促進在動態環境中高效建構、運營及持續演進，為系統提供長期價值支持。

適用範圍：Azure Cloud

參、 核心設計原則

原則簡述：

一、使用 API 管理 和 Azure Active Directory B2C 來驗證持有人令牌，以保護 Azure 和其他環境中的後端 API 的架構。

二、部署至 Azure Kubernetes Service （AKS） 的微服務應用程式,陳述基本 AKS 組態，著重在於AKS 上執行微服務架構的基礎結構和 DevOps的運作。

三、提供在 Azure 上設計任務關鍵性工作負載的指引。 它會使用雲端原生功能，將可靠性和作業效率最大化。

四、保護基礎結構的安全性服務。 藉由使用 Azure 安全性解決方案，增強 IT 環境的安全性狀態、降低弱點，以及透過架構完善的解決方案，以Microsoft最佳做法為基礎，防範缺口。

五、使用 Microsoft Defender 全面偵測回應 和 Azure 監視服務來整合，強化組織的 IT 安全性狀態。

參考微軟架構參考指引：

1. 使用 Azure API 管理 和 Azure AD B2C 保護後端 API
2. Azure Kubernetes Service 上的微服務架構
3. Azure 上任務關鍵性的基準架構
4. 使用 Azure 安全性服務打造第一道防線
5. 整合 Azure 與 Microsoft Defender XDR 安全服務
6. 使用 Azure Synapse 分析端對端

肆、　系統架構概述

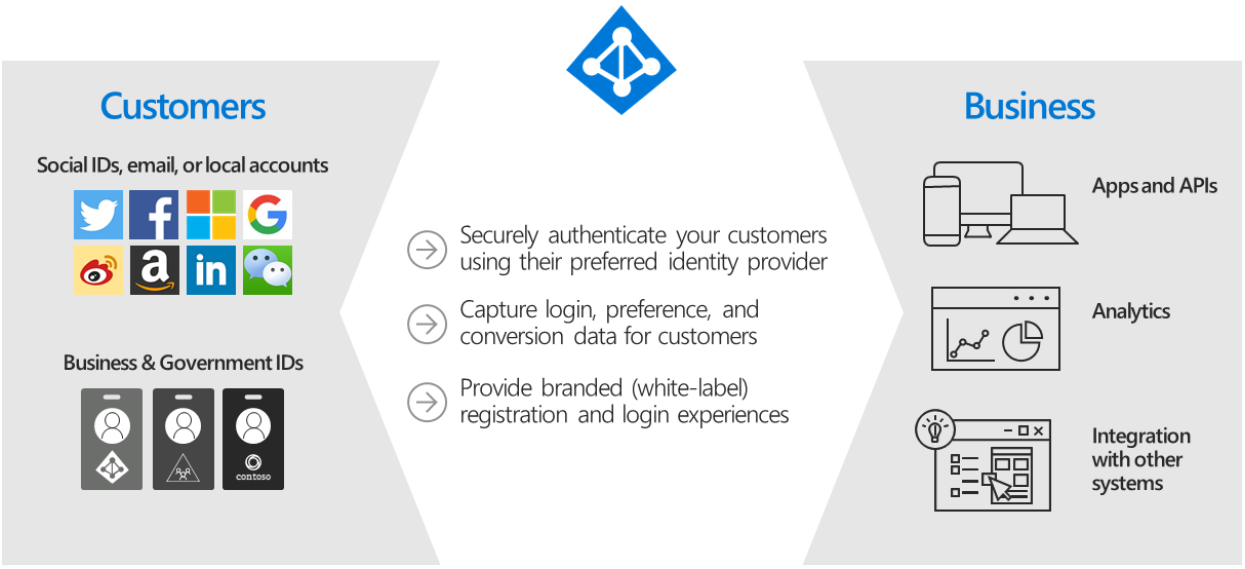
一、系統架構說明：

1. 使用 Azure Active Directory B2C 來驗證持有人令牌，以保護 Azure 後端資源的架構。
2. 使用 API 管理 以保護後端 API 的架構。
3. 使用DevOps 流程將微服務應用程式部署至 Azure Kubernetes Service （AKS）環境中。
4. 提供在 Azure 上設計任務關鍵性工作負載的指引。 它會使用雲端原生功能，將可靠性和作業效率最大化。
5. 由使用 Azure 安全性解決方案，增強 IT 環境的安全性狀態、降低弱點，以Microsoft最佳做法為基礎，防範缺口。

二、系統各元件工作流程說明：

Azure Active Directory B2C 提供企業對客戶身分識別即服務。 用戶會使用其慣用的社交、企業或本機帳戶身分識別取得應用程式和 API 的單一登入。

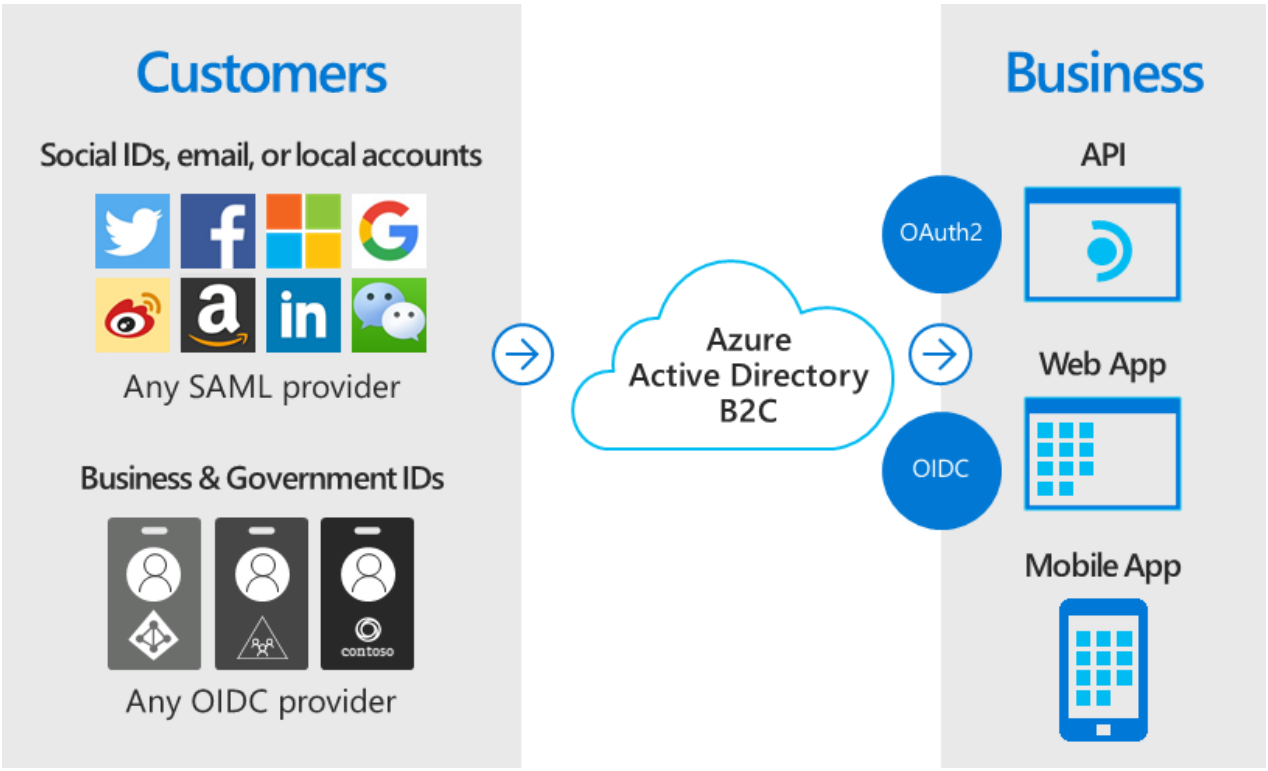
參考指引: https://learn.microsoft.com/zh-tw/azure/active-directory-b2c/overview



三、Azure Active Directory B2C：

Azure Active Directory B2C以使用者提供的身分識別存取單一登入

Azure AD B2C 使用標準型驗證通訊協定，包括 OpenID Connect、OAuth 2.0 與安全性聲明標記語言 (SAML)。 它會與大多數現代化應用程式和商業現成軟體整合。



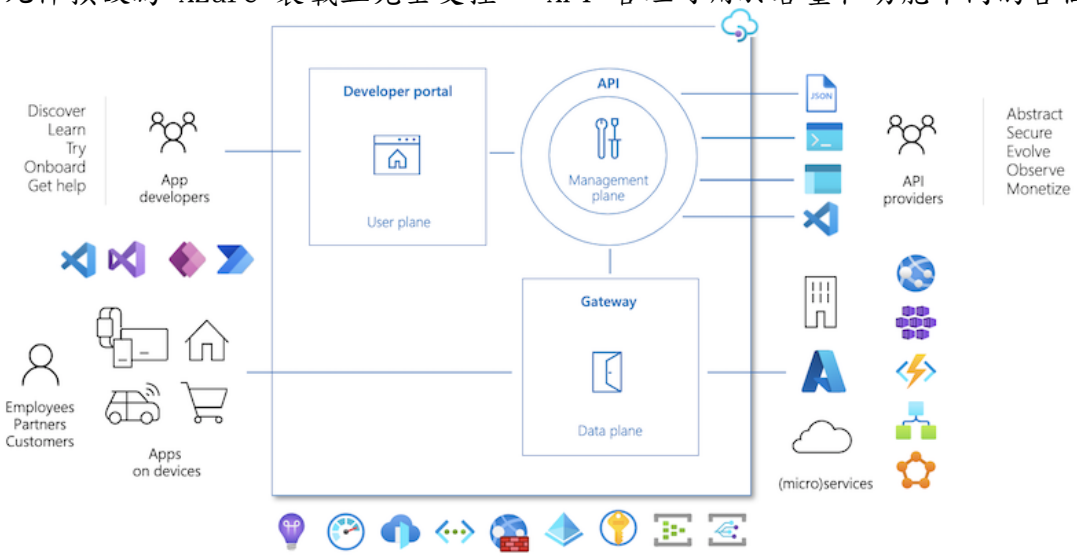
Azure AD B2C 可充當 Web 應用程式、行動應用程式和 API 的中央驗證授權單位，為上述項目建置單一登入（SSO）解決方案。

四、Azure API Management：

Azure API 管理（Azure API Management, APIM） 是 Microsoft Azure 提供的一項 雲端服務，用於幫助開發者安全、高效地公開和管理 API。通過 Azure API 管理，企業能夠為內部應用、合作夥伴或開發者社群提供 API 的完整管理方案，包括公開、保護、監控以及優化 API 的使用和性能。

參考指引: https://learn.microsoft.com/zh-tw/azure/api-management/api-management-key-concepts

Azure API 管理是由「API 閘道」、「管理平面」和「開發人員入口網站」所組成。 這些元件預設為 Azure 裝載且完全受控。 API 管理可用於容量和功能不同的各個階層。



五、Azure DevOps 流程：

是一套以 Microsoft Azure DevOps 服務為基礎的軟體開發生命週期管理流程，用於支持現代 DevOps 實踐。透過該流程，開發團隊能夠有效地計劃、開發、建置、測試、部署和監控應用程式，實現自動化、協作和持續交付。

此參考架構提供 Azure Resource Manager 範本 來布建雲端資源及其相依性。 使用 [Azure Resource Manager 範本][arm-template] 時，您可以使用 Azure DevOps Services 在幾分鐘內布建不同的環境，例如復寫生產案例。 這可讓您只在需要時節省成本並布建負載測試環境。

請考慮遵循工作負載隔離準則來建構 ARM 範本，工作負載通常會定義為任意功能單位;例如，您可以針對叢集有個別的範本，然後針對相依服務使用其他範本。 工作負載隔離可讓 DevOps 執行持續整合和持續傳遞 （CI/CD），因為每個工作負載都會由其對應的 DevOps 小組相關聯及管理。

參考指引:

https://learn.microsoft.com/zh-tw/azure/architecture/reference-architectures/containers/aks-microservices/aks-microservices

部署 （CI/CD） 考慮

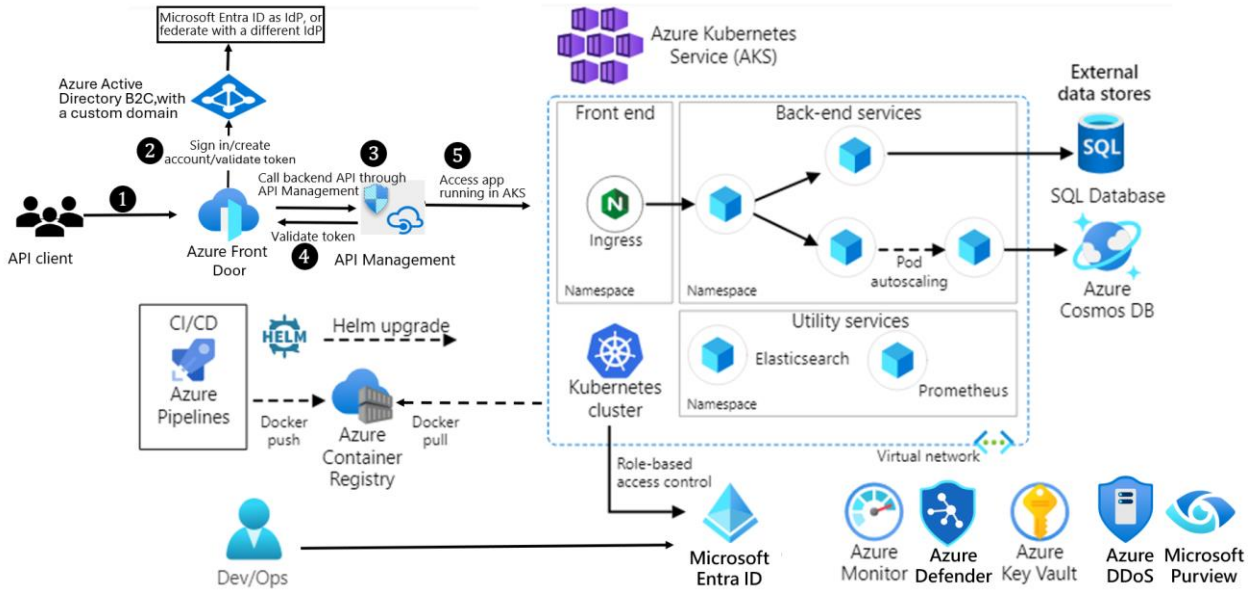
微服務架構的 如果沒有良好的 CI/CD 程序，就無法達到微服務所承諾的靈活度。故本文建議一些解決問題的方法：

- 持續整合：程式代碼變更經常合併至主要分支。 自動化建置和測試程式可確保主要分支中的程式代碼一律是生產品質。
- 持續傳遞：傳遞 CI 程式的任何程式代碼變更都會自動發佈至類似生產環境的環境。 部署到實時生產環境可能需要手動核准，但否則為自動化。 目標是您的程式代碼應該一律 準備好， 部署至生產環境。
- 持續部署：通過上述兩個步驟的程式代碼變更會自動部署到生產 。

參考指引:

https://learn.microsoft.com/zh-tw/azure/architecture/microservices/ci-cd

六、API Portal on Azure Cloud建議架構說明：



工作流程：

1. 為了取得應用程式的存取權，API 用戶端會藉由提供使用者名稱和密碼等認證來進行驗證。 IdP 在此解決方案中是 Azure AD B2C，但可以使用不同的識別碼。
2. 驗證要求會透過 Azure Front Door 移至 Azure AD B2C，其已設定自定義網域進行登入。 Azure AD B2C 會驗證使用者，並將 JSON Web 令牌 （JWT） 持有人令牌傳回給使用者。
3. 用戶端會觸發存取後端 API 的事件。 此事件可能是按下 Web 應用程式或行動裝置上的按鈕，或直接呼叫後端 API 的端點。
4. 要求會經過 Azure Front Door，其後端會對應至 API 管理 的公用端點。 API 管理 攔截要求，並使用其 validate-jwt 原則，對 Azure AD B2C 驗證持有人令牌。 如果令牌無效，API 管理 以 401 碼回應來拒絕要求。
5. 如果令牌有效，API 管理 將要求轉送至適當的後端 API執行在Azure Kubernetes Services (AKS)環境中。
6. Azure Kubernetes Service (AKS)。 AKS 是 Azure 雲端中裝載的受控 Kubernetes 叢集。 Azure 會管理 Kubernetes API 服務，只需要管理代理程式節點。
7. **Microsoft Entra ID**：AKS 會使用 Microsoft Entra 身分識別來建立和管理其他 Azure 資源，Microsoft用戶端應用程式中也建議使用 Entra 識別碼進行用戶驗證。
8. **Azure Container Registry**：使用 Container Registry 來儲存部署至叢集的私人 Docker 映射。 AKS 可以使用其Microsoft Entra 身分識別向 Container Registry 進行驗證。
9. **Azure Pipelines**：是 Azure DevOps Services 的一部分，並執行自動化組建、測試和部署。
10. **Helm** 是 Kubernetes 的套件管理員，可將 Kubernetes 對象組合並一般化成單一單位，可發佈、部署、版本設定和更新。
11. **Azure 監視器**會收集並儲存 Azure 服務的計量和記錄、應用程式遙測資料和平台計量。 使用此資料來監視應用程式、設定警示、儀表板，以及執行失敗的根本原因分析。

參考指引:

1. https://learn.microsoft.com/zh-tw/azure/architecture/solution-ideas/articles/protect-backend-apis-azure-management
2. https://learn.microsoft.com/zh-tw/azure/architecture/reference-architectures/containers/aks-microservices/aks-microservices

工具與資源

1. **Microsoft Entra ID**：提供的 雲端身份和存取管理解決方案，它是 Microsoft Entra 家族的一部分，專注於身份和存取管理，幫助組織保護用戶身份、簡化應用程式存取，並增強混合工作環境中的安全性與生產力。
2. **Azure Front Door**：提供的一項 全球分佈式的負載平衡與應用交付網關服務。它專為 Web 應用程式優化，幫助企業在全球範圍內快速、安全地交付應用內容。Azure Front Door 集成內容交付網路（CDN）和應用層負載平衡功能，實現高可用性、加速性能和增強安全性。
3. **Azure DevOps**：是 Microsoft 提供的一套 端到端 DevOps 工具和服務平台，專為支援現代化應用開發而設計，旨在整合開發（Development）和運維（Operations）流程，幫助團隊高效協作、快速交付高品質軟體並持續改進系統。
4. **Azure Container Registry**：是 Microsoft Azure 提供的 全託管型容器映像檔儲存庫服務，用於 存儲、管理和分發容器映像檔。ACR 支援 Docker 和 Open Container Initiative（OCI）標準，與其他 Azure 服務緊密整合，方便用戶將應用程式的容器化流程無縫融入 Azure 環境。

5. **Azure Kubernetes Service**：是 Microsoft Azure 提供的一項 完全託管型 Kubernetes 容器編排服務，用於簡化 Kubernetes 群集的部署與管理，幫助用戶運行、擴展和運維容器化應用。它支持自動化許多複雜的 Kubernetes 管理任務，例如群集佈建、升級和監控，使開發者可以專注於應用開發和交付。

6. **Azure SQL Database**：是 Microsoft Azure 提供的一種 完全託管的關聯式資料庫即服務 (Database-as-a-Service, DBaaS)。它基於 Microsoft SQL Server 技術，旨在幫助用戶快速部署和運行資料庫，而無需管理底層基礎架構，例如硬體、作業系統或資料庫軟體。Azure SQL Database 適合雲端應用的資料存儲需求，並支持彈性擴展、高可用性和安全性。
7. **Azure Cosmos DB**：是 Microsoft 提供的一種 全託管分散式 NoSQL 資料庫服務，設計為全球分佈式、高可用且高彈性，適用於運行現代化應用程序。Cosmos DB 支援多種資料模型和 API，讓用戶能夠根據需求構建應用，提供快速、可預測的性能和業務持續性。
8. **Azure Monitor**：是 Microsoft 提供的一套 完整的監控和診斷解決方案，用於收集、分析和操作 Azure 資源及其他雲端或本地環境的性能和健康狀態數據。Azure Monitor 幫助用戶深入了解系統運行狀況，識別問題並優化性能，提升應用的可用性和可靠性。
9. **Azure Defender**：是一種 雲安全態勢管理解決方案 (Cloud Security Posture Management, CSPM) 和 工作負載保護服務 (Cloud Workload Protection, CWP)，專為幫助企業保護其雲端和混合環境資源而設計。Azure Defender 能主動識別威脅、提供安全建議，並保護 Azure 資源、混合環境及多雲資源，實現全面的雲安全管理。
10. **Azure Key Vault**：是一種 Microsoft Azure 提供的 雲端密鑰管理服務，用於集中管理和保護敏感信息，例如加密密鑰、憑據和機密配置資料。它幫助開發者安全地存儲和控制對這些機密的存取，同時滿足合規性需求並增強應用程序的安全性。
11. **Azure DDoS**：是 Microsoft Azure 提供的一項 分散式阻斷服務攻擊 (DDoS) 防護服務，旨在保護 Azure 應用、服務和資源免受大規模、分散式的網絡攻擊。DDoS 攻擊通常試圖使目標資源不可用，Azure DDoS 提供了高效的防護，幫助確保應用可用性並減少潛在的業務損失。
12. **Azure API Management**：是 Microsoft Azure 提供的一項 API 管理平台，旨在幫助組織更簡單、有效地建立、保護、管理和監控 API（應用程式介面）。該服務設計用來提高 API 的安全性、可伸縮性、可重用性並改善整體運維效率，並促進組織內部或與合作夥伴之間的 API 資源共享和整合。
13. **Microsoft Purview**：是一套 資料治理、風險管理與合規性解決方案，專為企業幫助管理和保護其資料資產而設計。它提供全面的資料管理功能，幫助組織以統一的方式進行資料分類、標籤、監控和保護，以確保企業資料的合規性、隱私保護並提升資料治理效能。

伍、 安全性

一、角色型存取控制（RBAC）

Kubernetes 和 Azure 都有角色型存取控制的機制（RBAC）：

- Azure RBAC 可控制對 Azure 中資源的存取，包括建立新 Azure 資源的能力。許可權可以指派給使用者、群組或服務主體。 （服務主體是應用程式所使用的安全性身分識別。
- Kubernetes RBAC 控制 Kubernetes API 的許可權。 例如，建立 Pod 和列出 Pod 是可透過 Kubernetes RBAC 授權或拒絕用戶的動作。

二、秘密管理和應用程式認證

應用程式和服務通常需要認證，讓它們能夠連線到外部服務，例如 Azure 儲存體 或 SQL 資料庫。確保這些認證安全，而不會洩漏認證。

針對 Azure 資源，其中一個選項是使用受控識別。 受控識別的概念是應用程式或服務具有儲存在 Microsoft Entra ID 中的身分識別，並使用此身分識別向 Azure 服務進行驗證。應用程式或服務在 Microsoft Entra 識別符中建立服務主體，並使用 OAuth 2.0 令牌進行驗證。 執行中的進程程式代碼可以透明地取得要使用的令牌。

另外針對不支援受控識別的 Azure 服務、第三方服務、API 密鑰等等。 以下是一些安全地儲存秘密的選項：

- **Azure Key Vault**：在 AKS 中，您可以將一或多個秘密從 金鑰保存庫 掛接為磁碟區。 磁碟區會從 金鑰保存庫 讀取秘密。 然後Pod可以像一般磁碟區一樣讀取秘密。
- **HashiCorp Vault**： Kubernetes 應用程式可以使用 Microsoft Entra 受控識別向 HashiCorp Vault 進行驗證。
- **Kubernetes 秘密**：秘密會儲存在etcd中，這是分散式索引鍵/值存放區。 AKS 會在待用時加密 etcd。

三、容器和 Orchestrator 安全性

以下是保護 Pod 和容器的建議做法：

- **威脅監視**： 使用 適用於容器 的Defender Microsoft監視威脅（或第三方功能）。 如果您要在 VM 上裝載容器，請使用 適用於伺服器的 Microsoft Defender 或第三方功能。
- **弱點監視**：使用 Azure Marketplace 提供的 適用於雲端的 Microsoft Defender 或第三方解決方案，持續監視映射和執行容器的已知弱點。
- **使用 ACR 工作將映像修補自動化**：這是 Azure Container Registry 的功能。容器映像是從圖層建置而來。 基底層包括 OS 映像和應用程式架構映像，例如 ASP.NET Core 或 Node.js。 基底映像通常是從應用程式開發人員上游建立，並由其他專案維護者維護。
- **將映像儲存在受信任的私人登錄** 中，例如 Azure Container Registry 或 Docker Trusted Registry。 使用 Kubernetes 中的驗證許可 Webhook，以確保 Pod 只能從信任的登錄提取映像。
- **套用最低許可權** 原則
  - 請勿以特殊許可權模式執行容器。 特殊許可權模式可讓容器存取主機上的所有裝置。
  - 可能的話，請避免在容器內以根目錄的形式執行進程。 容器不會從安全性觀點提供完全隔離，因此最好以非特殊許可權使用者身分執行容器進程。

參考指引:

https://learn.microsoft.com/zh-tw/azure/architecture/reference-architectures/containers/aks-microservices/aks-microservices

四、Azure 安全性效能評定

每個安全性控制指的是一個或多個特定的 Azure 安全性服務。架構參考根據 ASB 文件顯示了其中一些服務及其控制編號。 這些控制向包括：

控制定義域	描述
網路安全性 (NS)	網路安全性涵蓋保護及保護網路的控制項，包括保護虛擬網路、建立私人連線、防止及減輕外部攻擊，以及保護 DNS。
身分識別管理 (IM)	身分識別管理涵蓋使用身分識別和存取管理系統建立安全身分識別和存取控制的控制項，包括使用單一登入、增強式驗證、受控識別（和服務主體，）應用程式、條件式存取和帳戶異常監視。
特殊權限存取 (PA)	Privileged Access 涵蓋保護租使用者和資源特殊許可權存取的控制措施，包括保護系統管理模型、系統管理帳戶和特殊許可權存取工作站的一系列控制項，以防止刻意和不小心的風險。
資料保護 (DP)	資料保護涵蓋待用資料保護的控制、傳輸中，以及透過授權的存取機制，包括使用存取控制、加密、金鑰管理和憑證管理來探索、分類、保護及監視敏感性資料資產。
資產管理 (AM)	資產管理涵蓋控制項，以確保資源的安全性可見度和控管，包括安全性人員許可權的建議、資產清查的安全性存取，以及管理服務和資源核准，（清查、追蹤和更正）。
記錄與威脅偵測 (LT)	記錄和威脅偵測涵蓋在雲端上偵測威脅的控制項，以及啟用、收集及儲存雲端服務的稽核記錄，包括啟用偵測、調查和補救程式，以及控制以在雲端服務中產生具有原生威脅偵測的高品質警示;它也包含使用雲端監視服務收集記錄、使用 SIEM 集中處理安全性分析、時間同步處理和記錄保留。
事件回應 (IR)	事件回應涵蓋事件回應生命週期中的控制項 - 準備、偵測和分析、內含專案及事件後活動，包括使用 Azure 服務（，例如雲端和 Sentinel Microsoft Defender）和/或其他雲端服務，將事件回應程式自動化。
態勢與弱點管理 (PV)	狀態和弱點管理著重于評估及改善雲端安全性狀態的控制項，包括弱點掃描、滲透測試和補救，以及雲端資源中的安全性設定追蹤、報告和更正。
端點安全性 (ES)	端點安全性涵蓋端點偵測和回應的控制，包括針對雲端環境中端點使用端點偵測和回應 (EDR) 和反惡意程式碼服務。
備份與復原 (BR)	備份和復原涵蓋控制項，以確保在不同服務層級執行、驗證及保護的資料和組態備份。
DevOps 安全性 (DS)	DevOps 安全性涵蓋與 DevOps 程式中安全性工程和作業相關的控制項，包括部署重要的安全性檢查（，例如靜態應用程式安全性測試、弱點管理），以確保整個 DevOps 程式的安全性;它也包含威脅模型化和軟體等常見主題，
治理與策略 (GS)	治理和策略提供指引，以確保一致的安全性策略和記載的治理方法來引導及維持安全性保證，包括建立不同雲端安全性功能的角色和責任、統一的技術策略和支援原則和標準。

參考指引:

https://learn.microsoft.com/zh-tw/security/benchmark/azure/overview

五、Network保護

Label	描述
NSG	附加到網路介面或子網路的免費服務。 NSG 可讓您使用 IP 位址範圍和連接埠來過濾傳入和傳出連線的 TCP 或 UDP 通訊協定流量。
DDOS	在虛擬網路上實施 DDoS 保護，協助您減緩不同類型的 DDoS 攻擊。
TLS/SSL	TLS/SSL 為大部分交換資訊的 Azure 服務（例如 Azure 儲存空間和 Web Apps）提供傳輸中加密。
私人連結	為 Azure 服務建立私人網路的服務，可讓您為最初暴露於網際網路的 Azure 服務建立私人網路。
私人端點	建立網路介面並將其附加到 Azure 服務。 私人端點是 Private Link 的一部分。 此設定可讓服務透過使用私人端點，成為您虛擬網路的一部分。

參考指引:

https://learn.microsoft.com/zh-tw/azure/architecture/solution-ideas/articles/azure-security-build-first-layer-defense

六、基礎結構和端點保護

Label	描述
Keyvault	金鑰保存庫可儲存 FIPS 140-2 Level 2 或 3 的金鑰、密碼和證書的服務。

參考指引:

https://learn.microsoft.com/zh-tw/azure/architecture/solution-ideas/articles/azure-security-build-first-layer-defense

七、應用程式和數據保護

Label	描述
Frontdoor + WAF	內容傳遞網路 (CDN)。 Front Door 結合多個存在點，為存取服務的使用者提供更好的連線，並加入 WAF。
API 管理	為 API 呼叫提供安全性並跨環境管理 API 的服務。
PenTest	在環境（包括 Azure 資源）中執行滲透測試的一套最佳實務。
記憶體 SAS 令牌	共用存取權杖，允許其他人存取您的 Azure 儲存體帳戶。
私人端點	建立網路介面並將其附加到您的儲存體帳戶，以將其設定在 Azure 上的私人網路內。
加密 (Azure 儲存體)	透過靜態加密保護您的儲存體帳戶。
SQL 稽核	會追蹤資料庫事件，並將其寫入 Azure 儲存體帳戶中的稽核記錄。
弱點評估	協助您探索、追蹤並修復潛在資料庫弱點的服務。
加密 (Azure SQL)	透明資料加密 (TDE) 可在資料靜止時加密，有助於保護 Azure SQL 資料庫服務。

參考指引:
https://learn.microsoft.com/zh-tw/azure/architecture/solution-ideas/articles/azure-security-build-first-layer-defense

八、身分識別

Label	描述
RBAC	Azure 角色型存取控制 (Azure RBAC) 透過使用基於使用者 Microsoft Entra 認證的細微權限，協助您管理 Azure 服務的存取。
MFA	多重要素驗證提供使用者名稱和密碼以外的其他驗證類型。
標識碼保護	識別保護是 Microsoft Entra ID 的安全性服務，每天分析數以萬億計的訊號，以辨識並保護使用者免於威脅。
PIM	Privileged Identity Management (PIM) 是 Microsoft Entra ID 的安全性服務。 它可協助您暫時提供Microsoft Entra ID（例如使用者管理員）和 Azure 訂用帳戶的超級用戶許可權（例如角色型 存取控制 系統管理員或 金鑰保存庫 系統管理員）。
Cond Acc	條件存取是一種智慧型安全性服務，使用您針對各種條件定義的原則來封鎖或授予使用者存取權限。

參考指引:
https://learn.microsoft.com/zh-tw/azure/architecture/solution-ideas/articles/azure-security-build-first-layer-defense

九、整合 Azure 與 Microsoft Defender XDR 安全服務
使用 Microsoft Defender 全面偵測回應 和 Azure 監視服務來整合這些安全性功能。

1. **Azure 監視器**是涵蓋多個 Azure 監控服務的總體平台。 它包括記錄管理、計量和 Application Insights 等。 它也提供可供使用和管理警示的儀表板集合。
2. **雲端的 Microsoft Defender** 提供虛擬機器 (VM)、儲存體、應用程式和其他資源的建議，以協助 IT 環境符合各種法規標準，例如 ISO 和 PCI。 同時，適用於雲端的 Defender 提供系統的安全性態勢分數，以協助您追蹤環境的安全性。 適用於雲端的 Defender 也會根據其收集和分析的記錄，提供自動警示。 適用於雲端的 Defender 以前稱為 Azure 資訊安全中心。
3. **Log Analytics** 是最重要的服務之一。 它負責儲存用來建立警示、深入解析和事件的所有記錄和警示。 Microsoft Sentinel 可在 Log Analytics 上運作。
4. **Microsoft Sentinel** 的運作方式就像 Log Analytics 的外觀一樣。 當 Log Analytics 儲存來自各種來源的記錄和警示時，Microsoft Sentinel 提供 API，以協助從各種來源提取記錄。 這些來源包括內部部署 VM、Azure VM、來自 Microsoft Defender XDR 和其他服務的警示。 Microsoft Sentinel 會讓記錄相互關聯，以提供 IT 環境中發生狀況的深入解析，避免誤判為真。
5. **網路監看員**提供了工具，可對 Azure 虛擬網路中的資源進行監視、診斷、檢視計量，以及啟用或停用記錄。
6. **流量分析**是「網路監看員」的一部分，可在網路安全性群組 (NSG) 的記錄上運作。流量分析提供許多儀表板，能夠從 Azure 虛擬網路中的輸出和輸入連接彙總計量。
7. **Application Insights** 著重於應用程式，為即時 Web 應用程式提供可擴展的效能管理和監視，包括對 .NET、Node.js、Java 和 Python 等各種平台的支援。
8. **Azure 儲存體分析**會執行記錄，並為儲存體帳戶提供計量。 您可以使用其資料來追蹤要求、分析使用趨勢，以及診斷儲存體帳戶的問題。

參考指引:
https://learn.microsoft.com/zh-tw/azure/architecture/solution-ideas/articles/microsoft-365-defender-security-integrate-azure

十、 管理資料隱私權和保護
Microsoft Purview：

1. 使用 Microsoft Purview 來探索數據資產、數據分類和敏感度，以涵蓋整個組織數據環境。
2. Microsoft Purview 可協助您使用使用者所需的特定商務術語來維護 商務詞彙 ，以了解數據集所代表的意義語意，以及它們在整個組織中使用的方式。
3. 可以 註冊所有數據源 ，並將其組織成 集合，這也可作為元數據的安全性界限。
4. 設定 定期掃描 ，以自動編錄及更新組織中數據資產的相關元數據。 Microsoft Purview 也可以根據 Azure Data Factory 或 Azure Synapse 管線的信息自動新增 數據譜 系資訊。
5. 根據在一般掃描期間套用的預先設定或海關規則，數據分類 和數據 敏感度 標籤可以自動新增至您的數據資產。
6. 數據控管專業人員可以使用 Microsoft Purview 所產生的報告和 深入解析 ，以控制整個數據環境，並保護組織免於發生任何安全性和隱私權問題。

Referen to:
1. https://learn.microsoft.com/zh-tw/azure/architecture/example-scenario/dataplate2e/data-platform-end-to-end
2. https://learn.microsoft.com/zh-tw/purview/legacy/concept-insights

陸、 成本最佳化

- 建置小組文化，瞭解預算、費用、報告和成本追蹤。
  成本優化是在組織的各種層級進行。 請務必瞭解目前工作負載如何符合組織目標和 FinOps 做法。 檢視業務單位、資源組織和集中式稽核原則可採用標準化財務系統。

- 只花在達到投資最高報酬所需的專案上。
  每個架構決策都有直接或間接的財務影響。 瞭解與組建與購買選項、技術選擇、計費模型和授權、訓練、作業等相關聯的成本。

- 最大化資源和作業的使用。 將它們套用至解決方案的交涉功能和非功能需求。
  服務和供應專案提供各種功能和定價層。 **購買一組功能之後，請避免使用量過低**。 尋找在層中最大化投資的方法。 同樣地，請持續評估計費模型，以根據目前的生產工作負載，找出更符合使用量的模型。

- **在不重新設計、重新交涉或犧牲功能或非功能需求的情況下提升效率。**
  利用機會來優化現有資源和作業的公用程式和成本。

- **隨著您的工作負載隨著生態系統演進，持續調整投資大小。**
  透過評估生產工作負載時， **預期架構、商務需求、程式，甚至是小組結構的變更**。 軟體開發生命週期 (SDLC) 實務可能需要演進。
  應該仔細評估所有變更對成本的影響。 定期監視變更和 ROI 趨勢，並評估您是否需要調整功能和非功能需求。

參考指引:
https://learn.microsoft.com/zh-tw/azure/well-architected/cost-optimization/principles

柒、 可靠性

- 收集商務需求，並將焦點放在工作負載的預定公用程式上。
  需求必須涵蓋工作負載特有的用戶體驗、數據、工作流程和特性。 *需求程序的結果必須清楚陳述預期*。 在指定投資的情況下，目標必須能夠達成並與小組交涉。它們必須記載為推動技術選擇、實作和作業。

- 工作負載必須繼續以完整或縮減的功能運作。
  應該預期元件故障、平台中斷、效能降低、資源可用性有限，以及其他錯誤都會發生。 在系統中建置復原功能，使其 容錯且可正常降級。

- 工作負載必須能夠預期並從所有範圍的大多數失敗中復原，且對用戶體驗和商務目標造成最少的中斷。
  即使是高度復原的系統也需要 災害準備方法，同時在架構設計和工作負載作業中。 在數據層上，您應該有可在損毀時修復工作負載狀態的策略。

- **在作業中保持左移，以預期失敗狀況。**
  利在開發生命週期*中早期和經常測試失敗*，並判斷效能對可靠性的影響。 為了進行根本原因分析和事後分析，您必須在小組之間擁有相依性狀態和持續失敗的共用可見度。 *來自可觀察系統的深入解析、診斷和警示是* 有效事件管理和持續改善的基礎。

- **避免過度建立架構設計、應用程式程式代碼和作業。**
  通常是您移除的專案，而不是您新增會導致最可靠解決方案的專案。 *簡單性可減少控件的介面區*，將效率不佳和潛在的設定錯誤或非預期的互動降到最低。

參考指引:
https://learn.microsoft.com/zh-tw/azure/well-architected/reliability/principles

捌、 卓越營運績效

- **讓開發和營運小組能夠透過共同作業、共同責任和擁有權的心態合作，持續改善其系統設計和流程。**
  良好的 DevOps 文化特性會在共同責任中茁壯成長。 開發和營運小組應將其目標和優先順序與客戶的期望保持一致，並牢記業務重點。 開發小組應該讓營運小組參與意見反應迴圈，讓改善能同樣地推動上游且其他小組能夠受益。

- **藉由標準化開發做法、強制執行品質大門，以及透過系統性變更管理追蹤進度和成功，將生產力優化。**
  開發小組會負責在發行前解決工作負載問題，併產生最少的摩擦。 請留意開發人員的效率，並 **針對快速轉換週期**進行優化，從程式代碼撰寫到測試結果。 實作有效且正確的程式，以規劃和標準化技術活動，並推動小組和專案關係人之間的共識。

- **深入了解系統、衍生深入解析，並做出數據驅動決策。**
  由監視工作負載**並納入 Azure 架構的所有要素，來持續**改善品質。 藉由提供必要的數據、統計數據和趨勢，讓小組和專案關係人能夠跨許多面向做出短期和長期決策。 從數據中學習並推動改善。

- **使用可預測性達到所需的部署狀態。**
  在工作負載的裝載平臺、應用程式、數據和設定資源之間，一致地達到所有環境中可預測性的目標。 **部署機制必須能夠自動化、測試、監視和版本控制**。 它應該模組化，並準備好視需要執行。

- **以更快速完成的軟體自動化取代重複的手動工作，並透過更高的一致性和精確度，並降低風險。**
  工作負載可能具有涉及小組成員進行平凡、重複且耗時的工作，而不需要人類智力的工作流程。 視頻率而定，可能會花費相當長的時間進行這些工作，在工作負載成長時投入更多時間。

- **在部署程式中實作護欄，以將錯誤或非預期狀況的影響降到最低。**
  部署程式必須遵循標準作業程式。 **任何變更都必須以相同層級的嚴謹**進行部署。 此原則同樣適用於程式代碼、組態和所有相關成品。 關鍵是儘早套用安全做法，以便您在生產環境中具有可預測性。

參考指引:
https://learn.microsoft.com/zh-tw/azure/well-architected/operational-excellence/principles

玖、 效能效率

- **已定義預期的用戶體驗，而且有一個策略可針對預先建立的商務需求來開發基準檢驗和測量目標。**

  效能和可靠性目標之間有強大的關聯性，可協助判斷效能、可用性和復原方面的服務品質。 如果沒有清楚的定義，測量、警示及測試效能是一項挑戰。 建立目標並透過一段時間的測試來識別實際數字之後，可以針對這些目標實作持續測試的自動化。

- **提供足夠的供應，以解決預期的需求。**

  測量效能牽涉到 **測量基準** ，在開發生命週期初期必須建立有效的效能管理基礎。

- **保護系統在使用中且隨著其演進而降低效能。**

  保護系統免於變更 ，使其不會滑回效能目標。 **在開發程式中整合測試和監視**。

  使用實際負載在生產環境中測試系統的效能，並在生產環境之前使用自動化測試來模擬該負載。

- **改善已定義效能目標內的系統效率，以增加工作負載值。**

  效率優化工作可讓工作負載使用較低的資源耗用量。 它們可能會導致工作負載處於過度布建狀態，並具有備用容量。 使用該容量來改善系統的可靠性。 消除容量以改善系統的成本。 或重新規劃容量，以支援現有資源上的新產品功能。

參考指引:

https://learn.microsoft.com/zh-tw/azure/well-architected/performance-efficiency/principles