AWS 白皮書

AWS 上 5G 網路的持續整合與持續交付

AWS 上 5G 網路的持續整合與持續交付: AWS 白皮書

Table of Contents

AWS 上 5G 網路的持續整合與持續交付

發佈日期：2021 年 3 月 8 日 (文件修訂)

摘要

此白皮書介紹 5G 網路的持續整合與持續交付 (CI/CD)，以及如何使用 Amazon Web Services (AWS) 工具和服務來完全自動化 5G 網路功能的部署和升級。白皮書會詳細描述 5G 網路功能的 CI/CD 的不同階段，包括網路設定、基礎設施部署、雲端原生網路功能部署以及網路功能的持續更新。它還提供有關與開放原始碼和第三方工具整合的詳細資訊，以用於測試、可觀察性和協同運作。

本白皮書面的目標對象為通訊服務提供者 (CSP) 以及獨立軟體開發廠商 (ISV)。

簡介

在過去，行動數據網路中新網路節點或新功能的開發、實驗室和現場整合測試以及生產部署，會需要數週甚至數月的時間，才能確保任務和關鍵商業電信 (電信) 服務的穩定性。部署週期較長的原因是傳統網路節點的整體架構、多廠商環境，以及 2G、3G 和 4G 行動網路中網路實體間的許多點對點介面。

正如使用 AWS 的 5G 網路演進白皮書中所介紹，由 3GPP 標準化的 5G 行動網路，現在支援由虛擬化和容器化支援的雲端原生架構。更具體地說，推出 5G 網路並支援微服務、無狀態和基於服務架構的新模式。

此 5G 架構代表不同網路功能可以作為鬆散耦合的獨立服務運作，透過明確定義的介面和 API 彼此通訊。最重要的是，每個網路功能都可以獨立更新。5G 中的此架構轉變透過讓將網路功能更新推出更容易且更頻繁，同時透過自動化來維護測試、安全需求和標準，使得 CSP 能夠實現更高的敏捷性和營運效率。

在網路功能廠商發佈新網路功能軟體套件 (例如基於容器的網路功能中的 Docker 影像) 或新組態檔案 (如 Kubernetes 應用程式案例中的 Helm Chart) 時，CSP 的新功能整合和部署一般會開始。(Helm Chart 是檔案的集合，其描述一組相關 Kubernetes 資源)。

使用 CI/CD 的模式進行 5G 網路功能部署的想法越來越受到注目，但此想法的實際實現是電信產業的一項挑戰。

AWS 率先開發了用於軟體交付的新 CI/CD 工具，以協助各行各業快速開發和推出軟體變更，同時維護系統的穩定性和安全。這些工具包括一組軟體開發和操作 (DevOps) 服務，例如 AWS CodeStar、CodeCommit、CodePipeline、CodeBuild 和 CodeDeploy。

AWS 還使用 AWS 雲端開發套件 (AWS CDK)、AWS CloudFormation 和基於 API 的第三方工具 (如 Terraform) 來傳遞 Infrastructure as Code (IaC) 的理念。使用這些工具，AWS 可以將 AWS 內網路功能的部署程序儲存為原始程式碼，並在 CI/CD 管道中維護此 IaC 原始程式碼以實現持續交付。

本白皮書描述利用 AWS IaC 和 CI/CD 工具進行 5G 網路功能部署和更新的詳細程序。此外，本白皮書還說明與第三方工具的整合，用於測試、可觀察性和協同運作。

AWS CI/CD 工具不限於 5G 網路功能。還用於將 4G 網路的部署自動化，這使得 CSP 能夠快速且有效地部署和更新 4G 網路功能。多數 4G 網路功能會基於虛擬網路功能 (VNF)。AWS CloudFormation 之類的 AWS CI/CD 工具集可用於將 4G VNF 的部署自動化，從而為 4G 網路部署擴展規模和提升時間效率。

持續整合與持續交付

持續整合

持續整合 (CI) 是一個軟體程序，在其中，開發人員會定期將其程式碼推送到一個中央儲存庫 (如 AWS CodeCommit 或 GitHub)。每個程式碼推送都會觸發一個自動化建置，接著是執行測試。CI 的主要目標是在早期階段發現程式碼問題，提高程式碼品質，並減少驗證和發佈新軟體更新所需的時間。

持續交付與部署

持續交付 (CD) 是一個軟體程序，在其中，成品會部署到測試環境、臨時環境和生產環境。持續交付可以完全自動化，也可以在關鍵點有核准階段。這可確保部署前有所有必要的核准 (例如，發佈管理核准) 都已就定位。在持續交付正確實作時，開發人員永遠都會有已透過標準化測試程序且準備好部署的建置成品。

使用持續部署，修訂版本會自動部署到生產環境，而不需要開發人員的明確核准，使得整個軟體發行程序自動化。這樣就可以在產品生命週期的早期獲得持續的客戶回饋迴圈。

使用持續部署，遞交並透過自動測試傳遞的各項變更都會自動發行到生產中。持續交付並不表示要發行遞交的各項變更，並立即將自動化測試傳遞到生產，而是為了確保各項變更都會備妥可投入生產中。

Infrastructure as Code

如使用 AWS 的 5G 網路演進白皮書所述，IaC 是將應用程式和其環境的佈建程序和生命週期管理自動化的重要動因。網路/IT 管理員和開發人員都可以使用組態檔案起始基礎設施，而不需依賴手動執行的步驟。IaC 會將這些組態檔案視為軟體程式碼。這些檔案可用來產生一組成品：即組成作業環境的運算、儲存、網路和應用程式服務。IaC 透過自動化消除了組態偏移，因而提高基礎設施部署的速度和敏捷性。

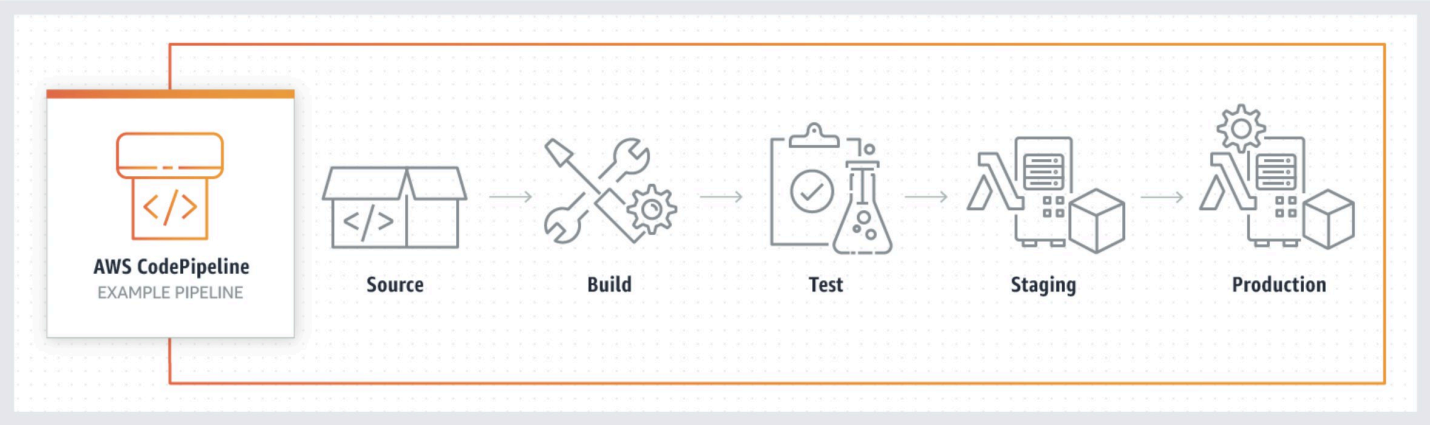
如果是在 AWS 上實作的網路功能虛擬化 (NFV)，此 IaC 架構從協同運作的觀點帶來了價值。從 Virtual Private Cloud (VPC) 建立到網路功能部署，每個步驟都可以使用程式碼編寫、以原始程式碼形式管理，並透過 AWS CodeCommit 中的版本控制維護。

用於網路功能的此 IaC 架構會產生可重複且可靠的基礎設施，而網路功能的建立和部署，可以延伸到網路分割管理和服務生命週期管理的端對端 (E2E) 自動化。AWS 提供一個全面的工具集，用於

以程式設計、描述性和宣告的方式建立、維護和部署基礎設施，運用如適用於 Kubernetes 的 AWS CloudFormation、AWS CDK 和 AWS CDK 等服務，以及所有 AWS 服務的 API 曝光。

AWS 上的 CI/CD

CI/CD 可以被描繪為管道，其中的新程式碼會在一端提交，經過一系列階段 (原始碼、建置、測試、暫存和生產) 測試，然後以生產就緒程式碼的形式發佈。



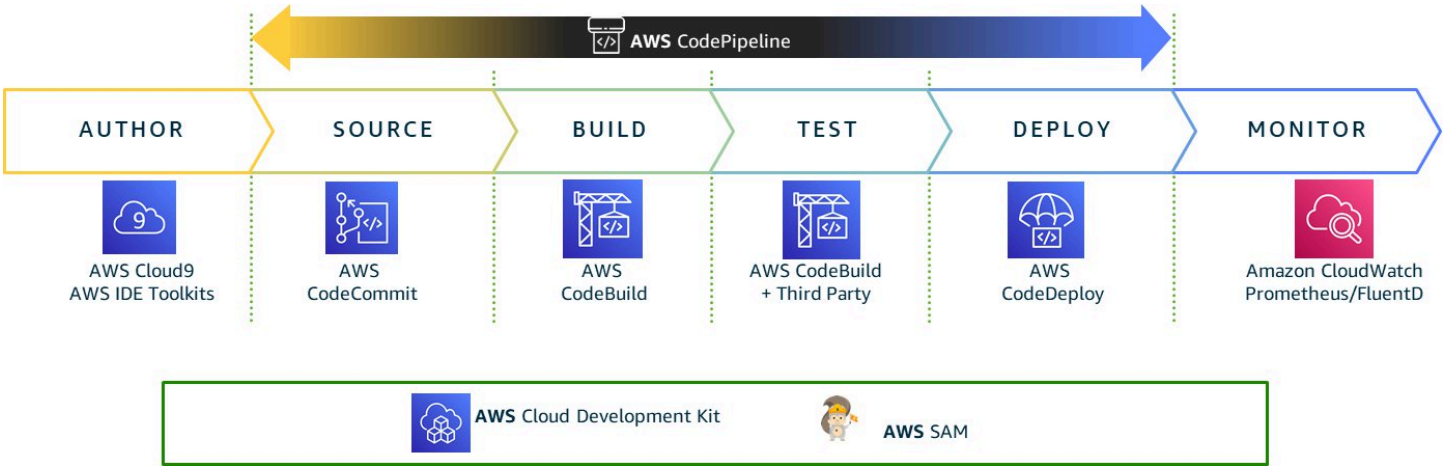
CICD 管道概觀

CI/CD 管道的每個階段都會在交付程序中以邏輯單元形式結構化。每個階段都可做為一個閘道，可檢查程式碼的特定面向。隨著程式碼在管道執行，我們會假設程式碼的品質在後期階段較高，因為它的更多面向將持續得到驗證。在早期階段發現的問題會阻止程式碼透過管道執行。測試的結果會立即傳送給團隊，而如果軟體未通過該階段，則所有進一步的建置和發行將會停止。

AWS 引入了一套完整的 CI/CD 開發人員工具，以加速軟體開發和發行週期。每次發生程式碼變更時，AWS CodePipeline 會根據定義的發行模型，將發行程序的建置、測試和部署階段自動化。這麼做可以實現快速且可靠地交付功能和更新。

程式碼管道可以與其他服務整合。這些可以是 AWS 服務，例如 Amazon Simple Storage Service (Amazon S3)，也可以是第三方產品，例如 GitHub。AWS CodePipeline 可以解決各種開發和作業使用案例，包括：

- 使用 AWS CodeBuild 編譯、建置和測試程式碼
- 將容器型應用程式持續交付至雲端
- 網路服務或特定雲端原生網路功能所需的成品 (例如描述子和容器映像) 的部署前驗證
- 容器化網路功能/虛擬網路功能 (CNF/VNF) 的功能、整合和效能測試，包括基準測試和迴歸測試
- 可靠性和災難復原 (DR) 測試。



AWS CICD 管道元件

AWS 可以使用以下 AWS 開發人員工具來設定 CI/CD 管道：

- AWS CodeCommit
- AWS CodeBuild
- AWS CodePipeline
- AWS CodeDeploy
- Amazon Elastic Container Registry
- AWS CodeStar

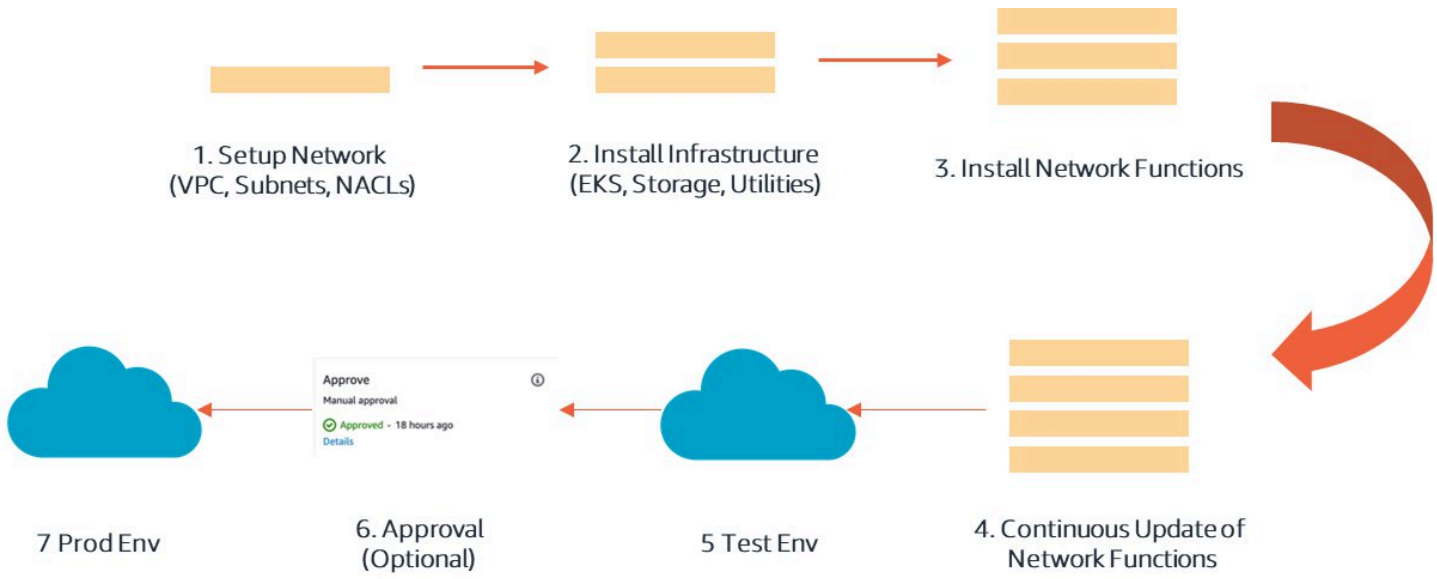
CI/CD 管道可以使用 AWS CDK 和 AWS CloudFormation 自動建立。在 NFV 網域中，此 AWS 原生自動化可以整合到管理和協同運作 (MANO) 架構以及 CSP 的服務協同運作架構中。

CI/CD 程序包含以下步驟：

- 網路設定 - AWS CDK 和 AWS CloudFormation 會起始網路先決條件的建立：
    - 聯網堆疊 (VPC、子網路、網路地址轉換 (NAT) 閘道、路由表和網際網路閘道)
- 基礎設施部署 - AWS CDK 和 AWS CloudFormation 會起始下列資源堆疊的建立：
    - 運算堆疊 (Amazon Elastic Kubernetes Service (Amazon EKS) 叢集建立、EKS 工作節點、AWS Lambda)

  - 儲存堆疊 (Amazon S3 儲存貯體、Amazon Elastic Block Store (Amazon EBS) 磁碟區和 Amazon Elastic File System (Amazon EFS))
  - 監控堆疊 (CloudWatch 、Amazon OpenSearch Service (OpenSearch Service)
  - 安全堆疊 (AWS Identity and Access Management (AWS IAM)、Amazon Elastic Compute Cloud (Amazon EC2) 安全群組、VPC 網路存取控制清單 (NACL))

- 雲端網路功能 (CNF) 部署 - 在此階段中，CNF 會使用 Kubectl 和 Helm Chart 工具部署到 EKS 叢集上。此階段還可以部署 CNF 所需的任何特定應用程式或工具以有效率地工作 (例如 Prometheus 或 Fluentd)。CNF 可以透過 Lambda 函數或使用 AWS CodeBuild 部署。
- 持續更新和部署 - 這些疊代執行的一系列步驟，用於部署會導致升級的容器/組態變更的部分變更。與 CNF 部署案例類似，可以使用 AWS 服務搭配來自 AWS CodeCommit、Amazon Elastic Container Registry (Amazon ECR) 或第三方來源系統 (如 GitLab Webhooks) 的觸發程序，將持續更新和部署自動化。



AWS CICD 管道流程圖表

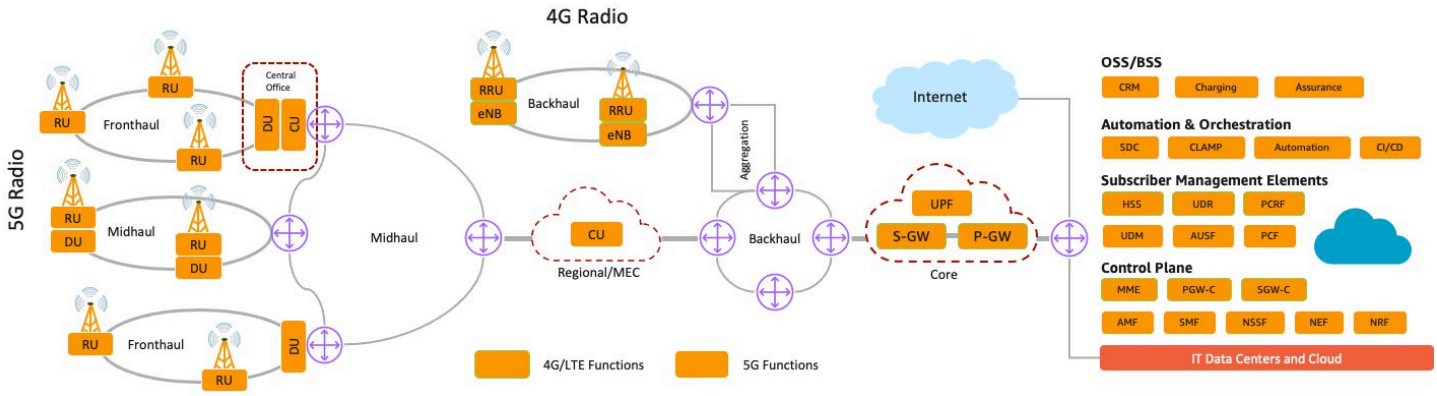
CI/CD 管道是使用 AWS CodePipeline 建置，並利用可建立模型、視覺化和自動化發佈軟體所需的步驟的持續交付服務。透過定義管道中的階段，您可以從原始程式碼儲存庫擷取程式碼，將該原始程式碼建置到可發行的成品中，測試該成品，然後將其部署到生產。只能部署成功通過這些階段的程式碼。您可以選擇將其他需求新增到您的管道中，例如，手動核准，以協助確保僅有已核准的變更會部署到生產環境中。

AWS 上的 5G 網路

5G 網路基礎設施的典型模型由 4G/5G 無線電站、前傳/中途/回傳網路、核心網路站台和電信/IT 資料中心組成。CSP 可以使用 AWS 服務來建立可擴展、靈活的 5G 網路基礎設施，同時降低前期投資成本。AWS 可用於在託管營運支援系統/商業支援系統 (OSS/BSS) 和大多數控制平面核心網路功能的區域實作虛擬網路營運中心 (NOC)。

還可以利用 AWS 來實作本機的中央辦公室 (CO) 或分散式資料中心，其中包含託管多數使用者平面功能，例如 UPF (使用者平面功能)、RAN 中央單元 (CU) 和多存取邊緣運算 (MEC) 的 AWS Outposts 機群。AWS 上的 5G 網路演進白皮書中就參考架構和 AWS 上的 5G 網路實作的優勢提供更詳細的說明。

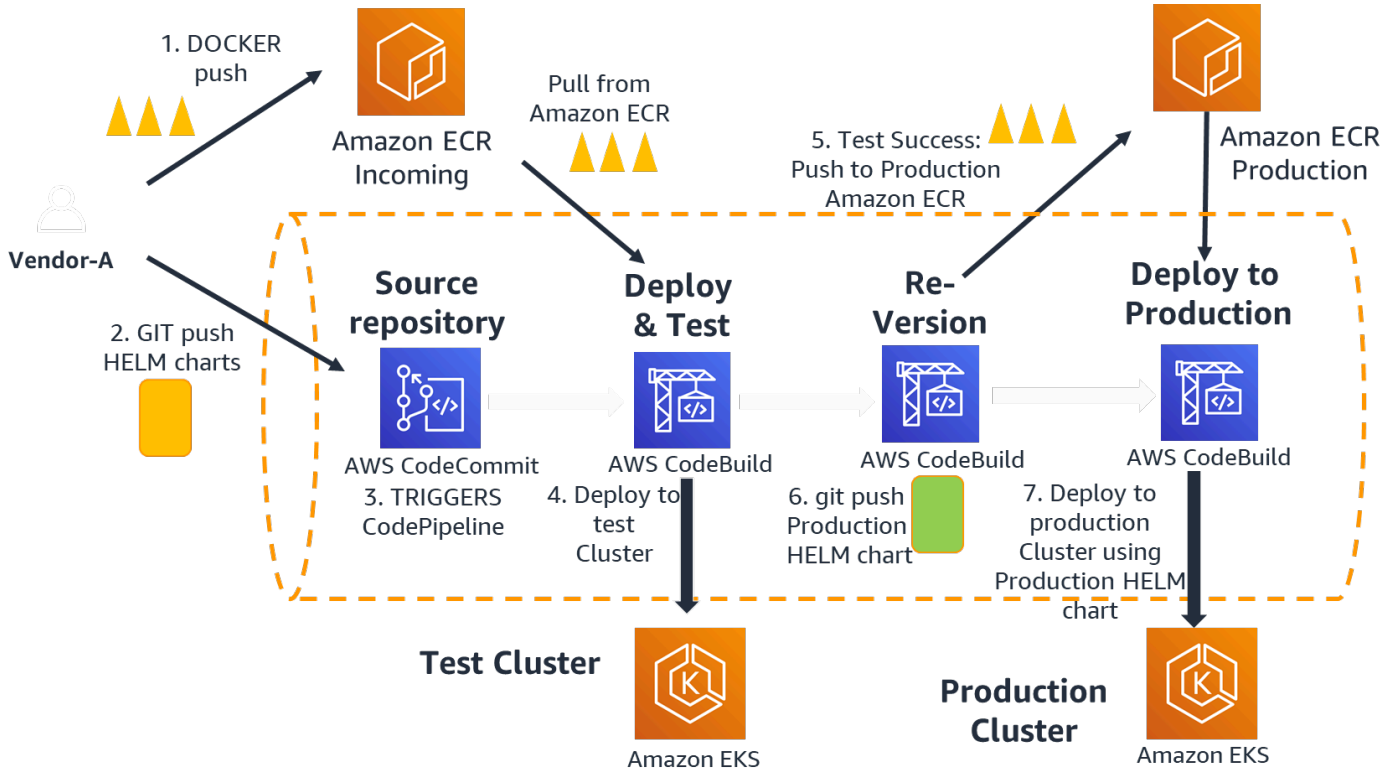
需要在 AWS 上實作 5G 網路時，本白皮書以下小節介紹的 AWS CI/CD 工具可以促進 5G 網路功能的部署、升級和生命週期管理的完全自動化。



5G 網路 E2E 架構

5G 網路中的 CI/CD

基礎設施的設計建構會以使用宣告性語言的程式碼形式儲存。這使得 CSP 能夠視需要獲得具有相同預期行為的可重複複製基礎設施。該程式碼會在程式碼儲存庫中維護，並設定一個管道來協調對已部署堆疊的更新 (例如，AWS CDK 和 AWS CloudFormation)。AWS 可協助建置 Infrastructure as Code (IaC)，以便敏捷地加入獨立軟體開發廠商 (ISV) 功能。



程式碼管道流程

透過 Helm Chart 變更雲端原生網路功能組態被視為網路功能的自動 CI/CD 管道執行的觸發程序。

AWS CodeCommit 可用於維護組態檔案，而 Amazon ECR 可用於保留容器映像。

如程式碼管道流程圖表所示，當 ISV 將新程式碼變更推送到程式碼儲存庫 (Helm Chart、組態檔案或屬性檔案) 時，將觸發程式碼管道。程式碼管道從 ECR 中提取映像，並使用 Helm Chart 來部署應用程式。新應用程式測試可以與第三方測試自動化架構整合。根據結果，CSP 可以核准生產部署。

CodePipeline 的來源階段會尋找組態檔案中的變更。來源階段的有效提供者包括 CodeCommit、Amazon S3、GitHub 或 AWS CloudFormation。透過使用 Lambda 函數來實作 Webhook，可以整合替代來源系統，因而實現 Gitlab 與 AWS CodePipeline 之間的事件驅動整合。如需詳細的實作指南，請參閱以下連結。

- Webhook 與 GitLab
- 容器登錄檔整合

CI/CD 管道設計應考量重要部署步驟，例如初始部署、測試，以及在測試結果與預期一致並對照基準驗證後提升到生產。管道程序的每個階段都會提供資料成品，其可實現比較和資料驅動型決策。

應用程式 CI/CD 管道步驟

每個階段都可視為一個單獨的任務，允許納入足以支援網路服務和雲端原生網路功能的驗證和部署工作流程。執行任務可以納入額外的第三方工具，例如流量生產生器和模擬器，實現端對端網路服務驗證。

AWS 提供完善的 AWS Step Functions (雲端原生狀態機器) 服務，其可與其他 AWS 服務原生整合，還能夠與外部系統 (例如 Jira 或測試自動化架構) 整合。

CI/CD 詳細步驟

CI/CD 可以被描繪為管道，其中的新程式碼會在一端提交，經過一系列階段 (原始碼、建置、測試、暫存和生產) 測試，然後以生產就緒程式碼的形式發佈。

以下是部署和測試的步驟。部署和組態主要可分為四個主要部分：

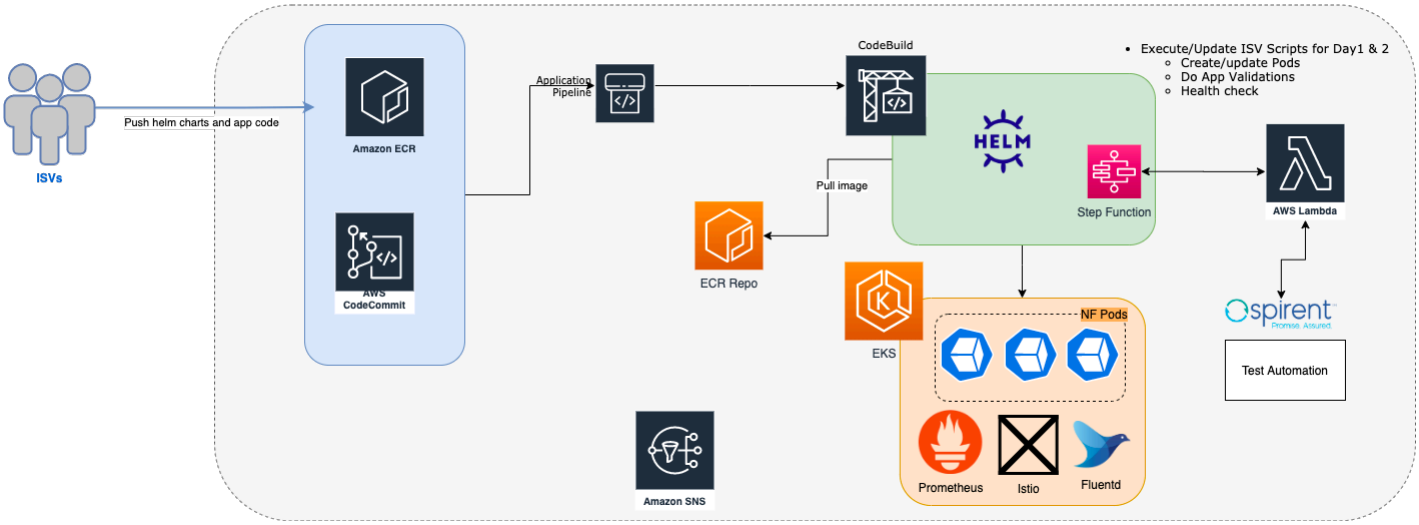
- 網路設定
- 基礎設施部署
- 雲端原生網路功能部署
- CNF 持續交付

網路設定

專注於設定基礎設施先決條件。這包括建立 VPC、網路、子網路、NACL 等。設計 IP 網路計劃，考量 ISV 和客戶的實作計劃 (例如多租用和靜態與動態配置)。此計劃可以在 AWS CDK 或 AWS CloudFormation 中編寫。執行此程式碼會部署雲端基礎設施網路先決條件。

基礎設施部署

基礎設施部署會佈建任何基礎設施元件。它包括產生 EKS 叢集和支援基礎設施 (例如 EFS、EKS 工作節點、ELB)，以及根據雲端原生網路功能需求設定叢集。根據 CNF 要求，AWS 還會為節點部署額外的網路介面，包括 Multus 介面。多數部署和組態步驟都是應用程式的一次性工作，且只有在需要作為應用程式的更新時才會更新。

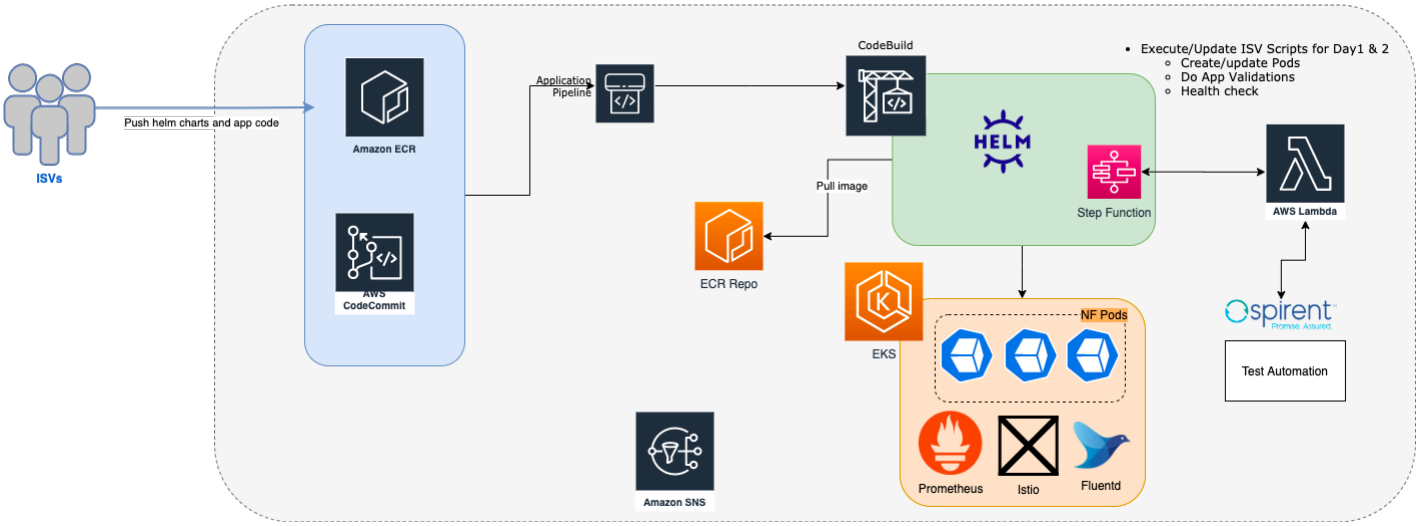


使用 CDK 的基礎設施部署

雲端原生網路功能部署

CNF 部署與應用程式部署有關。作為 CNF 部署的一部分，應用程式的 Helm Chart 會透過 CI/CD 程式碼管道實作。執行個別應用程式特定指令碼的回呼，主要涉及納入了前置和後續檢查。Helm Chart 會根據應用程式的需要依序實作，並在移至部署的下一步之前檢查 Kubernetes POD 的狀態。ISV 往往會提供一個包裝程式指令碼來執行 Helm Chart 和例行性檢查。這些 ISV 指令碼是從 AWS CodePipeline 內部叫用。作為此階段的一部分，除了 Amazon CloudWatch (會記錄和監控應用程式的雲端基礎設施) 之外，還會部署記錄和監控代理程式 (例如 Prometheus 和 Fluentd)。

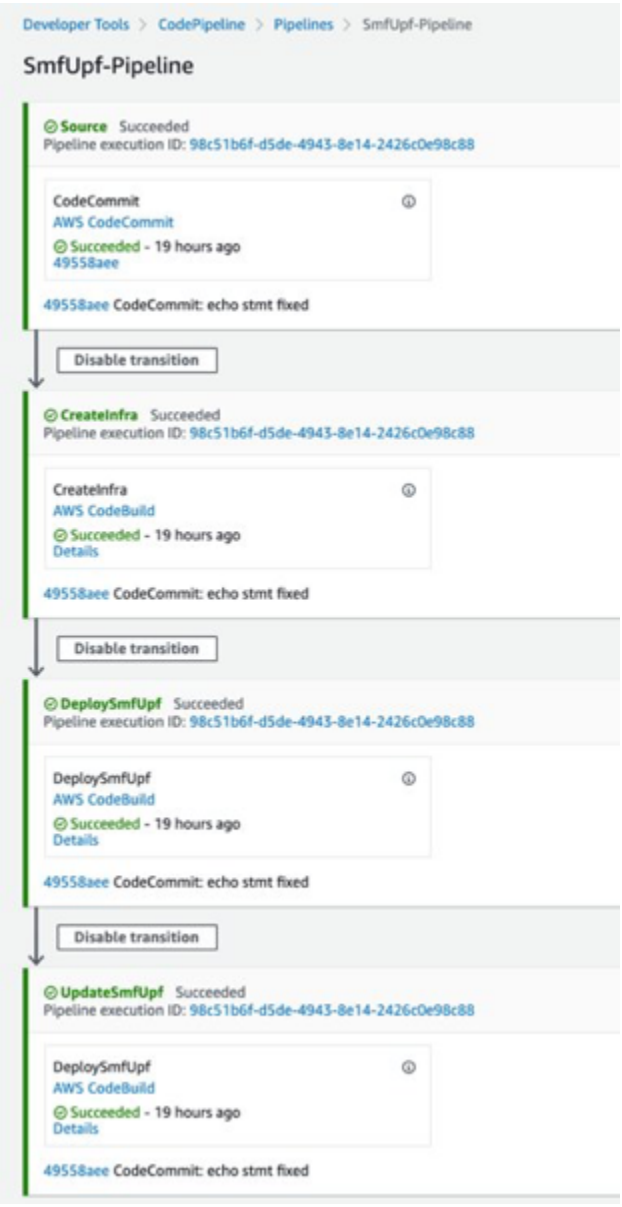
程式碼管道與第三方測試自動化架構整合。程式碼管道可以直接呼叫測試自動化架構 API，以在已部署的應用程式上執行測試，查詢測試結果，並分析結果。這會簡化應用程式的部署和測試。



應用程式部署和更新

以下是透過 AWS CodePipeline 部署使用者平面功能/工作階段管理功能 (UPF/SMF) CNF 的範例。

- 使用 CodeCommit、CodeBuild 和 CodePipeline 自動化完整的 CI/CD 程序。
- 基礎設施建立和應用程式安裝任務整合為管道的一部分。
- FluentD 和 Prometheus 代理程式已安裝並在 Amazon CloudWatch 儀表板中建立。



UPF/SMF CNF 的部署範例

CNF 持續交付

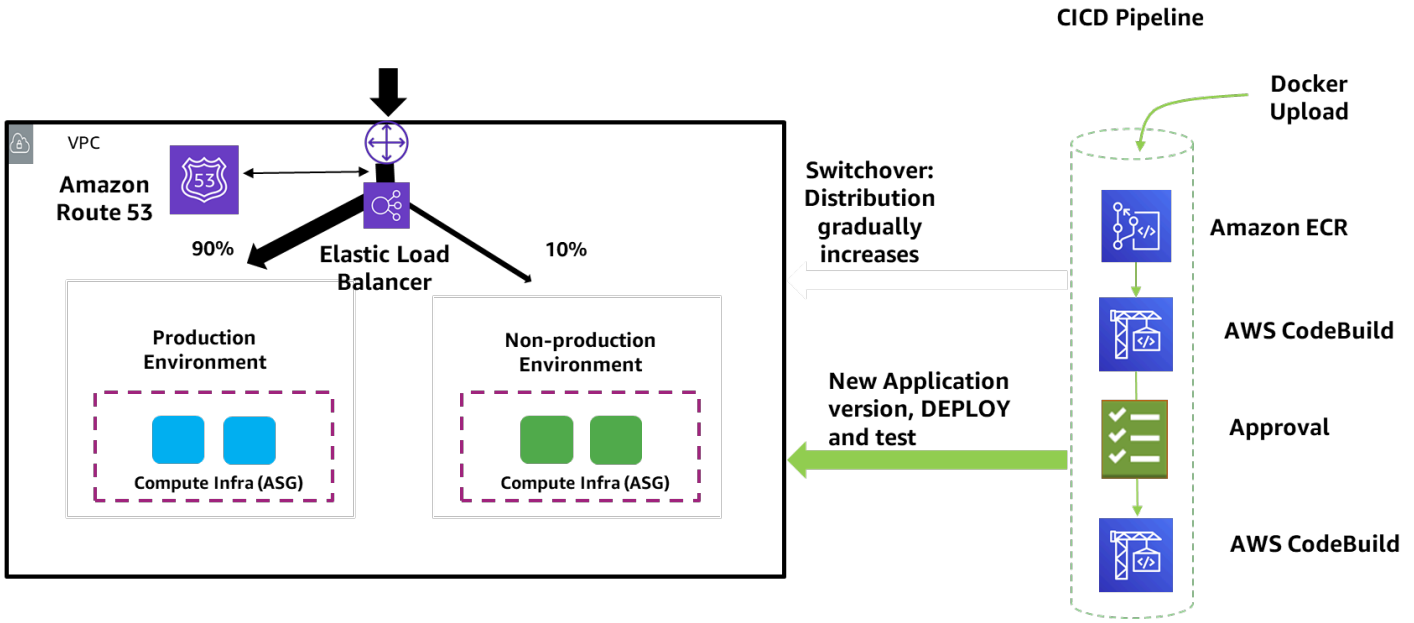
此步驟包含重複執行的一系列步驟，用於部署會導致升級的容器/組態變更的部分變更。CNF 持續交付會透過管道自動化，並且為個別應用程式特定。AWS 使用標準 Helm Chart 來更新特定的 CNF。程式碼管道會針對應用程式更新狀態進行預先檢查和後續檢查。更新的 CI/CD 管道還與測試自動化架構整合，以執行自動化測試。這種抽象會實現網路功能的全新部署。

CNF 持續交付和部署可大致分為以下幾類：

- 應用程式升級 - 多數應用程式升級都是 Kuberbetes 應用程式 POD 內的變更。這些更新可以透過程式碼管道自動套用。多數 CNF 會透過提供多個應用程式 POD 執行個體來支援就地升級。多個執行個體允許輪流升級方法。並非所有應用程式 POD 變更都支援 Helm 升級。管道會將這些變化考慮在內，並視需要使用 Helm 安裝/刪除。
- 主要升級 - 主要升級主要是資料庫結構描述變更。此變更無法在不會導致某些停機時間的情況下套用。這些變更的標準方法是刪除應用程式並重新建立相關的 Pod。在此程序期間，應用程式可能無法使用。下列工具會用於升級：

    - AWS CloudFormation 可讓客戶描述和佈建 JSON 或 YAML 範本中的所有基礎設施資源。AWS CloudFormation 透過由 Lambda 支援的自訂資源提供功能強大的擴展機制。客戶可以將 AWS CloudFormation 延伸到 AWS 資源之外，還可以在其他環境中佈建需要的資源；例如，混合環境中的內部部署資源。AWS CDK 為開發人員提供使用較高階的熟悉程式設計語言 (例如 Python、TypeScript、JavaScript、Java 和 C#) 來建置程式碼，接著將該程式碼編譯為較低階的 AWS CloudFormation JSON 格式，然後加以部署的能力。
    - 藍綠部署 - AWS 在測試和生產環境中支援並建議使用藍/綠和基於 canary 的部署。藍/綠部署可讓客戶在受控的環境中測試新應用程式版本。它們提供一種簡單且順暢的方法來切換至生產流量。基於 canary 的部署會透過使用一小部分的生產流量測試非生產綠色環境，以發現生產流量引起的任何問題，藉此延伸此概念。將對內部模擬測試流量和少量生產流量測試新應用程式版本，這使得使用者在切換到生產流量之前有信心。生產流量會逐漸增加，直到切換完成。實作涉及加權 DNS 和加權 ELB 目標群組。
    - 透過設定 AWS CodePipeline 使用藍/綠和基於 canary 的部署階段，可以實現自動化。核准階段一開始可以在佈建期間手動驅動，但之後應該完全自動化。在測試環境中，最好在部署到生產環境之前，一律使用回復動作測試，以驗證向前和向後相容性。具有服務網格叢集上的藍/綠部署取決於為實現順暢轉換，最終應用程式和路由閘道為服務網格提供的支援。

- AWS Systems Manager 提供整合的使用者介面，因此您可以檢視來自 CI/CD 部署的網路功能所使用的多個 AWS 服務的作業資料。Systems Manager 可讓您對整個 AWS 資源自動化作業任務。



Canary 部署

安全性

安全是關鍵元素。以下是在部署應用程式時 AWS CI/CD 程序考量的安全步驟清單。

- 來源 - 配置給廠商的 ECR 儲存庫已設定將「推送時掃描」標記啟用，使得 Docker 影像的任何上傳都將立即受限於安全掃描。任何已知的常見漏洞和曝光 (CVE) 都將標記為通知。除了 ECR，當廠商將圖表放入 AWS CodeCommit 儲存庫，還會要求他們將搭配 Secrets Manager 使用的任何密碼加密，而不是純文字。
- 成品完整性 - 無論是靜態 (使用 AWS 受管金鑰) 還是傳輸 (使用 SSL/TLS)，都會在整個管道中對使用的成品加密。
- IAM 使用者和角色 - 為使用者或資源提供的許可會基於最低權限原則。如果您在不同服務中的資源操作，則應該有需要設定的跨 IAM 角色信任關係。例如，AWS CodeBuild 需要許可才能在 Amazon EKS 叢集上執行命令。
- 稽核 - AWS CloudTrail 提供的稽核功能會追蹤服務和使用者操作間的每個 API 呼叫，並實現對過去事件的評估。

- 映像漏洞掃描 - 上傳到 Amazon ECR 的 CNF 映像會經過自動掃描，以掃描安全漏洞。掃描結果的報告可於 AWS Management Console 中取得，也可以透過 API 擷取。然後，可將結果傳送給 CSP 操作員以採取矯正措施，包括替換該 CNF 映像。

安全檢查會在管道的不同階段進行，確保新上傳的映像是安全的，並符合所需的合規檢查，以便可以向 CSP 傳送通知以進行核准：

- 容器登錄檔將掃描任何開啟的 CVE 漏洞。
- 在測試階段期間，系統會檢查組態是否存在資訊洩漏、已知的個人身分識別資訊 (PII) 模式，從而觸發合規檢查規則，以發現意外開啟的 TCP/UDP 連接埠和 DOS 漏洞之類的問題。
- 向後和前向相容性會經過驗證，以確保升級/回復的安全性。

除了應用程式之外，確保成品在階段間的加密傳輸 (無論是靜態還是在運輸中)，以佈建管道安全性非常重要。

可觀測性

AWS 依預設會針對在 AWS 上部署的 5G CNF 實現可觀察性。這會透過 Amazon CloudWatch 啟用。CloudWatch 讓您全面掌握雲端資源和應用程式。

在此程序期間，Amazon CloudWatch 有四個主要步驟：

1. 收集 - 從 AWS 和內部部署伺服器上執行的所有 AWS 資源、應用程式和服務收集指標和日誌。
2. 監控 - 使用 CloudWatch 儀表板將應用程式和基礎設施視覺化，並排關聯日誌和指標以疑難排解，並使用 CloudWatch 警示設定警示。
3. 行動 - 使用 CloudWatch Events 和 AWS Auto Scaling 自動回應對作業的變更。
4. 分析 - 使用 CloudWatch Metric Math，最多達 1 秒的指標、延長資料保留 (15 個月) 和即時分析。

Amazon CloudWatch 代理程式會安裝在客戶的 Kubernetes 叢集中。代理程式支援 Prometheus 組態、探索和指標提取功能，能夠增添並將所有高保真度 Prometheus 指標和中繼資料作為嵌入式指標格式 (EMF) 發佈到 CloudWatch Logs。

Amazon CloudWatch Container Insights 可將容器化應用程式探索和收集 Prometheus 指標自動化。它會自動收集、篩選並在儀表板中建立視覺化的彙總自訂 CloudWatch 指標。

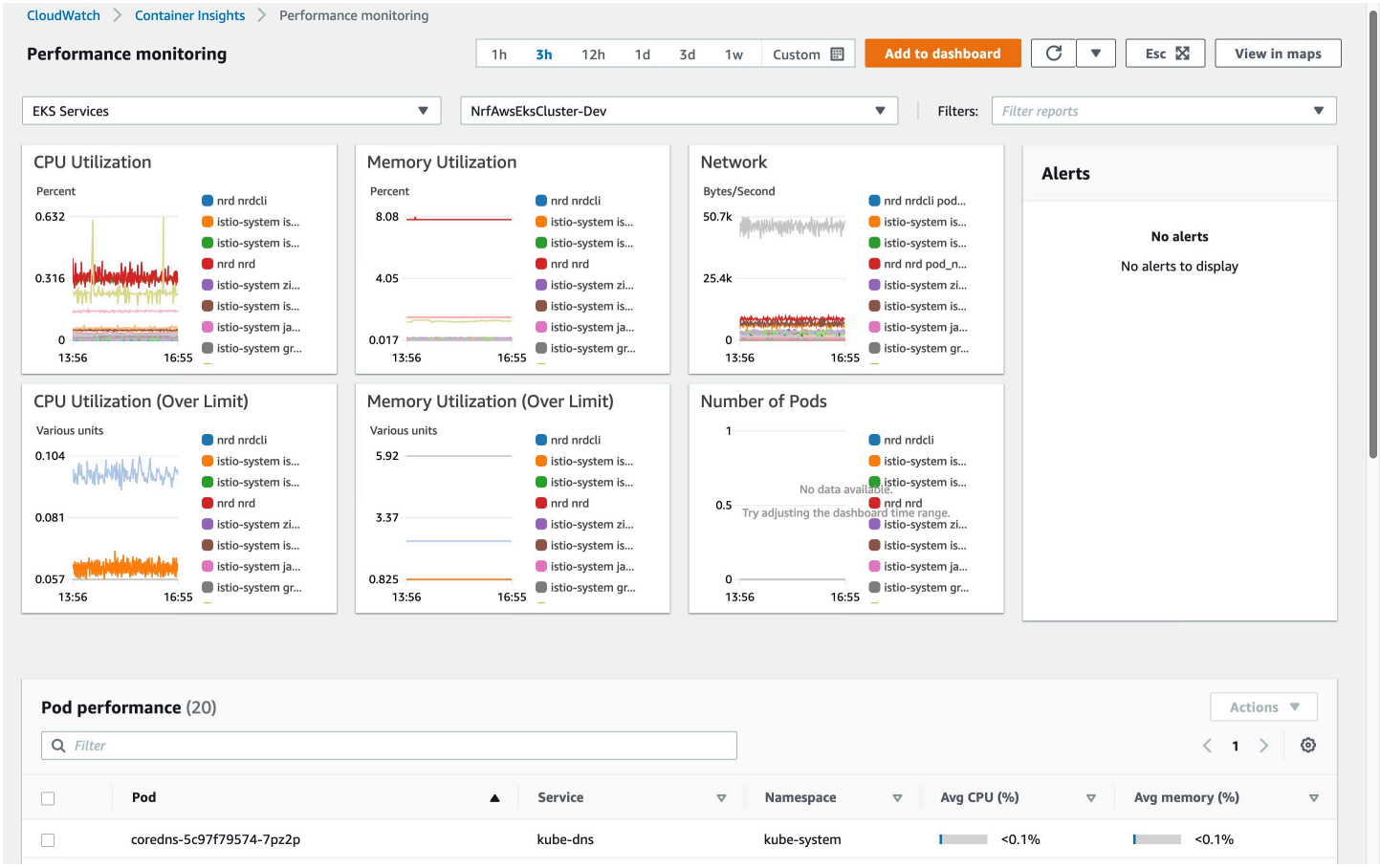
每個事件都會針對一組精選的完全可設定的指標維度，將指標資料點建立為 CloudWatch 自訂指標。將彙總的 Prometheus 指標發佈為 CloudWatch 自訂指標統計數字，可以減少監控、警示和疑難排解

效能問題和故障所需的指標數量。您還可以使用 CloudWatch Logs Insights 查詢語言來分析高保真度 Prometheus 指標，以隔離會影響容器化環境運作狀態和效能的特定容器 Pod 和標籤。

AWS CloudTrail 提供此可見性，跨服務記錄每個 API 呼叫。AWS Config 提供合規驗證的功能。AWS 使用各種服務 (如 AWS X-Ray 和 AWS CloudTrail) 為客戶提供額外的監控選項，包括指標、日誌、應用程式、基礎設施和管道的事件。

- AWS 可以原生整合開放原始碼指標工具，例如 Prometheus、Fluentd 等。
- Prometheus 指標可以進一步擷取 Amazon CloudWatch 或 OpenSearch Service 以進一步分析。
- AWS 會使用 fluentD 作為從各種系統收集日誌的標準機制。為此專案使用和設定了該相同機制。

如需如何設定此機制的詳細資訊，請參閱將 FluentD 設定為 DaemonSet 以將日誌傳送到 CloudWatch Logs。



Amazon CloudWatch 監控指標的範例

使用第三方和開放原始碼工具的 CI/CD 協同運作

協同運作層使用 IaC 來部署和設定執行 5G 網路功能所需的底層基礎設施。此層應設計為模組化、可攜式且可重複使用。

該基礎設施會遵循雲端原生最佳實務，具高可用性、備援且可擴展。

如稍早的小節所示範，您可以使用 AWS 雲端開發套件來達成基礎設施的部署。這可以使用 HashiCorp 的 Terraform 來完成。

Terraform

Terraform 是一項開放原始碼 IaC 軟體工具，其提供一致的命令列介面 (CLI) 工作流程來管理數百個雲端服務。Terraform 會將雲端 API 編碼為宣告式組態檔案。

對於使用 Terraform 的部署，請使用 CDK 中使用的相同原則。程式碼會以模組形式建立結構，使您可以根據廠商的需求自訂和重複使用聯網元件。

組態都是參數化的，使得您能夠根據提供者和 ISV 的建議對部署完全量身訂做。

網路功能部署可分為兩個階段：

- 需要 AWS 基礎設施會透過中央儲存庫建立和管理。
- 組態和程式碼會集中儲存在 GitHub 儲存庫中。

建立先決條件後，即可透過在上一個階段中設定的應用程式管道來部署網路功能。

基礎設施部署

基礎設施部署包括成功部署和設定網路功能的所有先決條件。

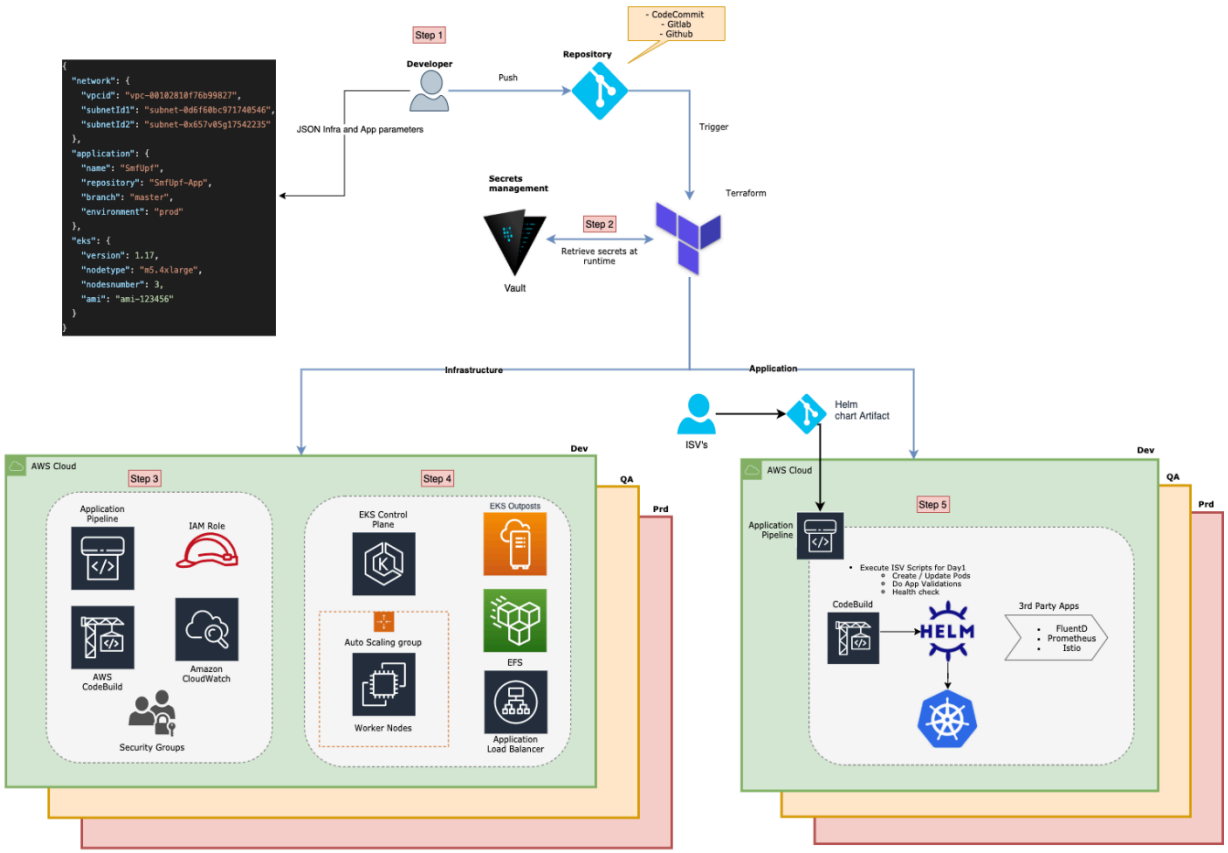
作為此階段的一部分建立的部分元件包括：

- 聯網 - VPC、公有和私有子網路、路由、負載平衡器
- 運算 - Kubernetes (Vmware Tanzu、Amazon EKS 或 AWS Outposts)、Amazon EC2 執行個體主要節點和工作節點、Auto Scaling 群組
- 儲存 - Amazon EFS、Amazon EBS、Amazon S3 儲存貯體
- 安全性 - IAM 角色、安全群組

- 管道 - CodePipeline、CodeBuild
- CloudWatch - CloudWatch、Prometheus、FluentD

以下是由 Terraform 協調的基礎設施順序，並在下圖中說明：

1. 開發人員使用 IaC 程式碼填入儲存在中央儲存庫的 JSON 檔案。該檔案包含關於所需基礎設施組態的資訊，例如，執行個體大小、Kubernetes 版本、網路資訊和應用程式儲存庫詳細資訊。
2. 在執行時間從 HashiCorp 保存庫或 AWS Secrets Manager 擷取金鑰。
3. 部署和設定基礎設施元件 (聯網、運算、儲存和安全)。
4. 具有託管網路功能 Pod 的工作節點的 Amazon EKS 叢集隨即會部署。也可以在 AWS Outposts 上部署 Amazon EKS，以支援需要靠近資料中心的工作負載。
5. 建立並設定應用程式管道，以聆聽網路功能儲存庫中的變更。每次將程式碼推送到設定的儲存庫分支時，管道都會自動觸發網路功能的建置、測試和部署。
6. 可收集和集中日誌和指標的可觀察性工具會作為服務部署在所有節點中，並提供可以在 Grafana 或 OpenSearch 儀表板中視覺化的近乎即時的資料



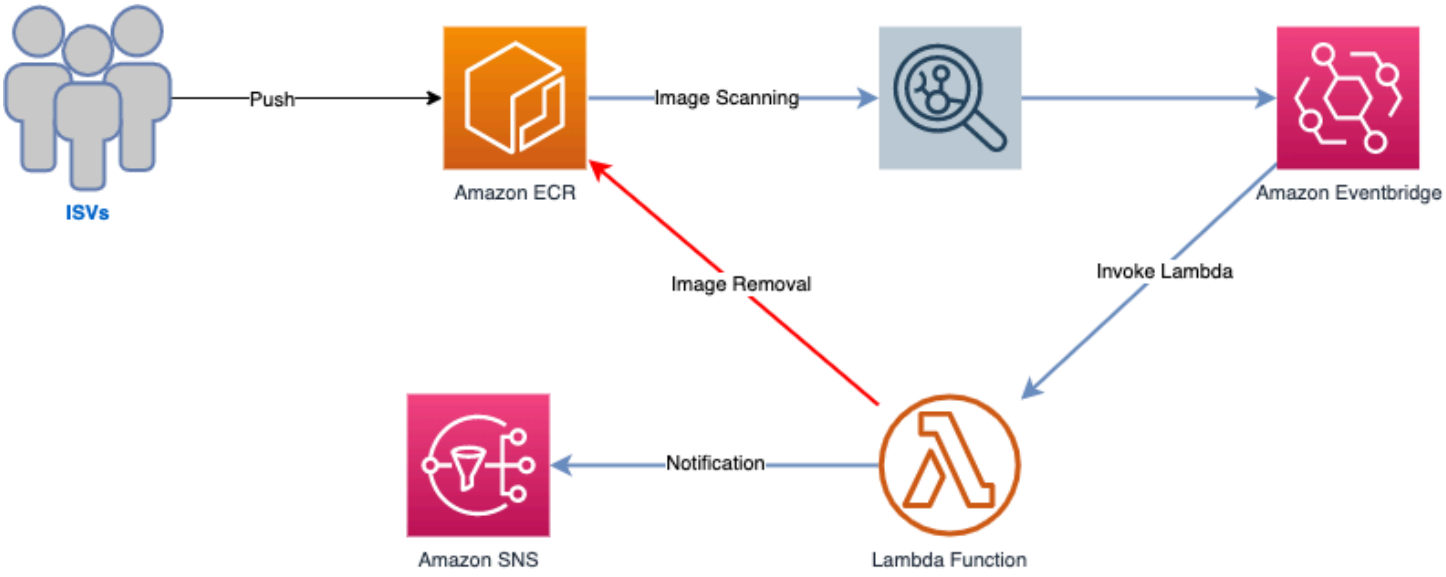
網路功能部署和組態

網路功能部署和組態

在先前階段中建立的管道可讓 ISV 和提供者能夠分散和最佳化網路功能的部署。管道已連線並聆聽應用程式儲存庫中的變更，這已在上圖步驟 1 中的 JSON 檔案中設定。

為了檢查由第三方發佈的映像，我們會部署並設定一個漏洞掃描解決方案，可協助識別容器映像中的軟體漏洞。掃描解決方案會自動檢查推送到 Amazon ECR 的所有新映像。如需 ECR 映像掃描的詳細資訊，請參閱映像掃描。

下圖示範映像漏洞掃描解決方案的架構。



映像漏洞掃描解決方案的架構

應用程式管道可設定為由掃描結果後映像中的變更觸發或由儲存庫中的直接變更觸發。例如，建立新的 Helm 映像時。

下列清單是建立/升級網路功能的順序，如下圖所示：

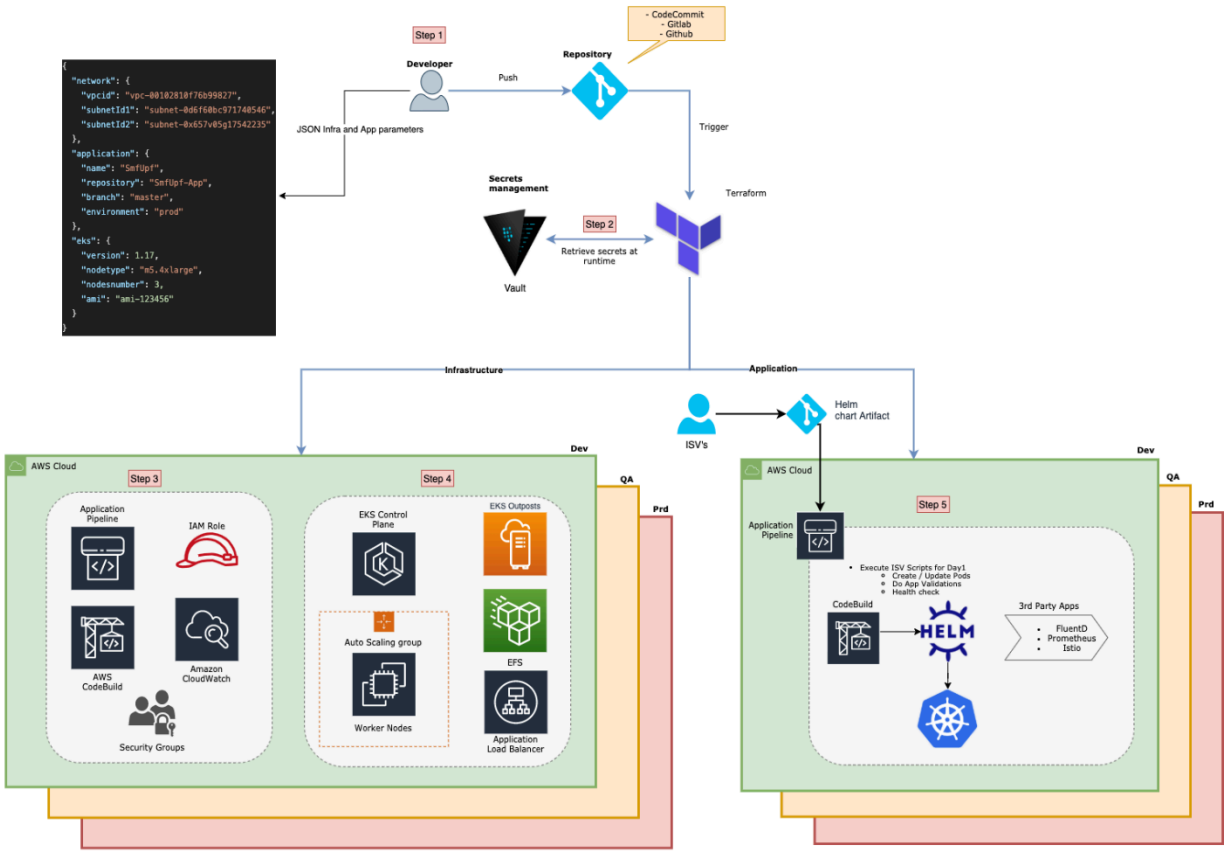
ISV 會將新映像發佈至 Amazon ECR。(如果映像經過核准，則會觸發應用程式管道。

CodePipeline 會從 Amazon ECR 中提取新映像，並使用 CodeBuild 將映像部署到 Kubernetes。Helm 命令可用於升級網路功能。

部署映像後，將觸發測試即服務 (TaS)。TaS 會驗證新部署，並將在壓力下的網路功能效能的相關資料和指標集中化。

日誌和指標被收集並集中在 OpenSearch 和 Grafana 中。也可以設定 Datadog、Istio 和 Prometheus 之類的第三方，以提供額外的可觀察性。

還可以部署能協調網路資源的 MANO，並與解決方案整合。它會使用收集的資料來執行自動化的動作，例如網路分割和服務品質 (QoS) 自動擴展。



應用程式管道

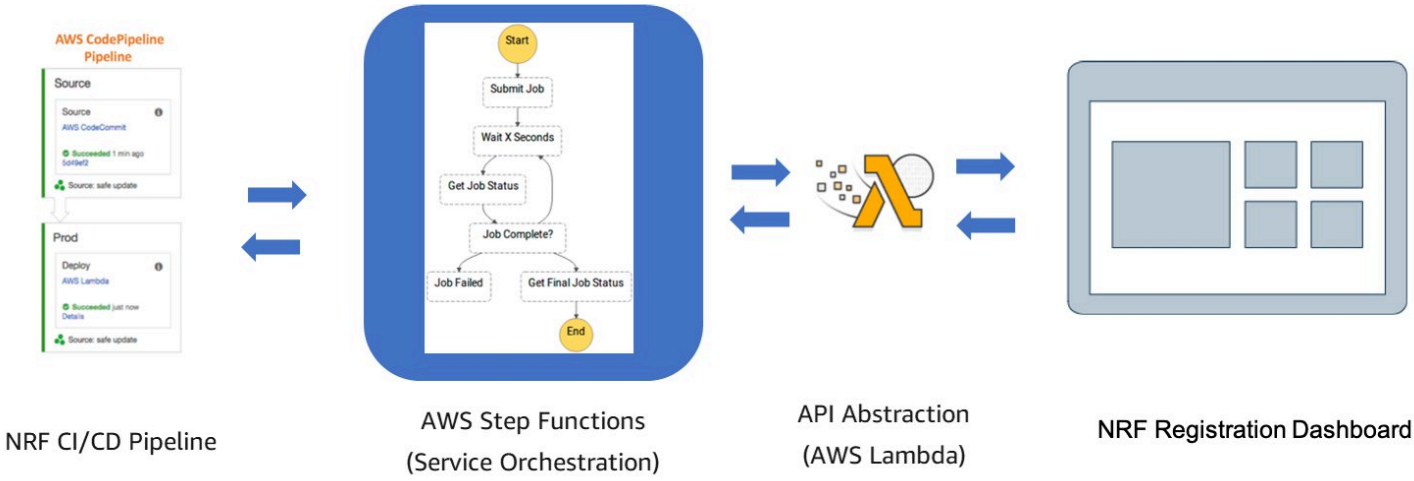
測試

電信特定測試自動化架構可以整合到程式碼管道中。程式碼管道與步驟函數整合，以協調與測試自動化架構的整合。AWS Step Functions 會使用多個 Lambda 函數，透過 API 呼叫叫用測試自動化架構 (第三方工具)。步驟函數一開始將取得測試 ID，執行測試，並從測試自動化架構中取得結果。然後，步驟函數會分析結果並將其傳遞到程式碼管道，以便核准用於生產部署的應用程式。核准可以是自動化或視需要在程式碼管道中手動維護。對於 CSP 來說，這是將部署從測試環境提升到生產環境的重要步驟。整合所需的高階 API 可分為以下方式：

- 取得內容

- 執行特定測試案例
- 停止測試案例
- 取得結果

呼叫外部 REST API 的複雜性是使用 AWS Step Functions 建立模型，其會允許標準建構呼叫平行流程、等候結果、基於條件進行分支，並整合 REST API 與 AWS CodePipeline。



測試流程

CI/CD 和協同運作

持續整合與持續交付是伴隨著雲端原生架構與其應用於 5G 的方式的整體自動化哲學的一部分。協同運作是此哲學的另一個面向，預期它會對網路上發生的任何變更為動態且具回應性。協同運作和 CI/CD 預期將緊密耦合，以保證服務運作正常，並將造成的服務中斷最小化。CI/CD 與協同運作之間的整合預計在兩個方面：

- 將修補程式和升級套用至系統，需要以對任何即時服務造成的中斷最少的方式管理和協調。例如，協同運作可以動態判斷應推出更新的最佳時間。
- CI/CD 感知協同運作允許在部署升級期間根據所採用的部署模型策略 (canary、線性或一次全部) 轉移流量。

一般來說，協同運作解決方案會在 CI/CD 管道之上執行，以允許協同運作將控管階段引入這些管道，並且可以接觸進行中的升級週期。

結論

CI/CD 為開發人員和應用程式團隊在幾分鐘內部署新應用程式程式碼提供一條清楚且高效的途徑。AWS 擁有豐富的工具，可幫助開發人員整合、測試和部署新程式碼，包括 AWS CodePipeline、AWS CodeCommit、AWS CodeBuild、AWS CodeDeploy 和許多其他程式碼。本文件會介紹如何使用 AWS 服務來建立 CI/CD 流程，用於以完全自動化的方式部署 5G 網路功能，包括完成新程式碼部署所需的不同步驟。它還說明如何將第三方測試自動化架構整合到 CI/CD 流程，以及如何使用 Terraform 之類的第三方工具。

作者群

此文件的作者包括：

- Amazon Web Services AWS 電信資深顧問 Hisham Elshaer
- Amazon Web Services AWS 電信首席顧問 Vara Prasad Talari
- Amazon Web Services AWS 電信首席顧問 Rabi Abdel
- Amazon Web Services 共用交付資深顧問 Franco Bontorin
- Amazon Web Services 共用交付顧問 Pragtideep Singh
- Amazon Web Services 全球客戶雲端基礎結構架構師 Subbarao Duggisetty
- Amazon Web Services AWS 電信資深合作夥伴解決方案架構師 Young Jung

文件修訂

若要收到此白皮書更新的通知，請訂閱 RSS 摘要。

update-history-change	update-history-description	update-history-date
初始出版	白皮書初始出版	2021 年 3 月 8 日

深入閱讀

如需其他資訊，請參閱：

- 在 AWS 上實作持續整合與持續交付 (白皮書)
- AWS 上的電信營運等級行動封包核心網路 (白皮書)
- 使用 AWS 的 5G 網路演進 (白皮書)

縮略字

- AMF - 存取和行動性管理功能
- API - 應用程式設計介面
- AUSF - 身分驗證伺服器功能
- BSS - 商業支援系統
- CDK - Cloud Development Kit
- CI/CD - 持續整合與持續交付
- CLI - 命令列介面
- CNF - 雲端原生或容器化網路功能
- CSP - 通訊服務提供者
- CU - RAN 中央單元
- CVE - 常見的漏洞和風險
- DoS - 拒絕服務
- DR - 災難復原
- DU - RAN 分散式單元
- E2E - 端對端
- ECR - Elastic Container Registry
- EFS - Elastic File System
- EKS - Elastic Kubernetes Service
- EPC - 演進的封包核心
- IaC - Infrastructure as Code
- ISV - 獨立軟體開發廠商
- MANO - 管理和協同運作
- MEC - 多存取邊緣運算
- NACL - 網路存取控制清單
- NAT - 網路位址轉譯
- NF - 網路功能
- NFV - 網路功能虛擬化
- NFVO - 網路功能虛擬化協調器

- NOC - 網路營運中心
- NRF - 網路儲存庫功能
- OSS - 營運支援系統
- PII - 個人識別資訊
- QoS - 服務品質
- RAN - 無線電存取網路
- SBI - 基於服務的介面
- SMF - 工作階段管理功能
- SSL - Secure Sockets Layer
- TaS - 測試即服務
- TCP - 傳輸控制通訊協定
- TLS - Transport Layer Security
- UDM - 整合資料管理
- UDP - 使用者資料包通訊協定
- UPF - 使用者平面功能
- VIM - 虛擬化基礎設施管理員
- VNF - 虛擬網路功能
- VPC - Virtual Private Cloud

聲明

客戶應負責對本文件中的資訊自行進行獨立評估。本文件：(a) 僅供參考之用，(b) 代表目前的 AWS 產品供應與實務，如有變更恕不另行通知，以及 (c) 不構成 AWS 及其附屬公司、供應商或授權人的任何承諾或保證。AWS 產品或服務以「現況」提供，不提供任何明示或暗示的擔保、主張或條件。AWS 對其客戶之責任與義務，應受 AWS 協議之約束，且本文件並不屬於 AWS 與其客戶間之任何協議的一部分，亦非上述協議之修改。