使用者指南

AWS Batch

AWS Batch: 使用者指南

Table of Contents

什麼是 AWS Batch？

AWS Batch 可協助您在 AWS 雲端 上執行批次運算工作負載。Batch 運算是開發人員、科學家和工程師存取大量運算資源的常用方式。 AWS Batch與傳統的批次運算軟體類似，消除了設定和管理所需基礎架構的無差別繁重工作。這項服務可以快速佈建資源，回應提交的任務，以便消除容量限制，降低運算成本，進而加快結果產生。

作為完全受控的服務，可AWS Batch協助您執行任何規模的批次運算工作負載。 AWS Batch自動佈建運算資源，並根據工作負載的數量和規模優化工作負載分佈。有了AWS Batch，您無需安裝或管理批次運算軟體，因此您可以專注於分析結果和解決問題。

主題

AWS Batch 的元件

AWS Batch簡化跨區域內多個可用區域執行批次工作。您可以在新的或現有的 VPC 中建立 AWS Batch 運算環境。在運算環境設置完畢並與任務佇列關聯後，您可以定義任務定義，即指定由哪個 Docker 容器映像來執行您的任務。容器映像是從容器登錄檔儲存和提取，可能來自您的 AWS 基礎設施的內部或外部。

任務

您提交到 AWS Batch 的工作單位 (如 shell 指令碼、Linux 可執行檔，或 Docker 容器映像)。它具有名稱，並使用您在任務定義中指定的參數在運算環境中作為容器化應用程式AWS Fargate或 Amazon EC2 資源上執行。任務可以透過名稱或 ID 來參照其他任務，且可能取決於其他任務是否順利完成。如需詳細資訊，請參閱任務。

任務定義

工作定義指定工作的執行方式。您可以將工作定義視為工作中資源的藍圖。您可以為您的工作提供 IAM 角色，以提供其他AWS資源的存取權。您也可以同時指定記憶體和 CPU 需求。任務定義也可以為持久性儲存控制容器屬性、環境變數和掛載點。當提交單一任務時，在任務定義中的許多規格，可以指定新的值予以覆寫。如需更多資訊，請參閱 Job 定義

任務佇列

當您送出AWS Batch工作時，會將它送到特定的工作佇列，工作所在的佇列中，直到排定到計算環境為止。您可以將一或多個計算環境與工作佇列產生關聯。您也可以指派這些運算環境的優先順序值，甚至是跨工作佇列本身。例如，您可以擁有一個高優先順序佇列，供您提交時間敏感的工作，以及低優先順序佇列，適用於運算資源較便宜時隨時執行的工作。

運算環境

運算環境是一組受管的或未受管的運算資源，用於執行任務。透過受管運算環境，您可以在多個詳細層級指定所需的運算類型 (Fargate 或 EC2)。您可以設定使用特定類型 EC2 執行個體的運算環境，例如`c5.2xlarge`或`m5.10xlarge`。或者，您可以選擇只指定要使用最新的執行個體類型。您也可以指定環境的最小、所需和最大 vCPUs 數量，以及您願意為競價型執行個體支付的數量，以隨需執行個體價格和目標 VPC 子網路的百分比表示。 AWS Batch視需要有效率地啟動、管理和終止運算類型。您也可以管理自己的運算環境。因此，您必須負責在為您AWS Batch建立的 Amazon ECS 叢集中設定和擴展執行個體。如需詳細資訊，請參閱運算環境。

開始

在 AWS Batch 主控台建立任務定義、運算環境及任務佇列，開始使用 AWS Batch。

AWS Batch第一次執行精靈可讓您選擇建立運算環境和工作佇列，以及提交 Hello World 範例工作。如果您已經擁有要啟動的 Docker 映像檔AWS Batch，則可以使用該映像檔建立工作定義，然後將其提交至佇列。如需詳細資訊，請參閱開始使用 AWS Batch。

儀表板

在AWS Batch儀表板上，您可以監視最近的工作、工作佇列和運算環境。依預設，會顯示下列儀表板 Widget：

- Job 概觀 — 如需AWS Batch工作的詳細資訊，請參閱任務。
- Job 佇列概觀 — 如需AWS Batch工作佇列的詳細資訊，請參閱Job 佇列。
- 運算環境概觀 — 如需AWS Batch運算環境的詳細資訊，請參閱運算環境。

您可以自訂顯示在「儀表板」頁面上的 Widget。下列各節說明您可以安裝的其他 Widget。

單一工作佇列

此 Widget 會顯示單一工作佇列的詳細資訊。

若要新增此 Widget，請依照下列步驟執行。

1. 開啟 AWS Batch主控台。
2. 從導覽列中，選取AWS 區域您想要的。
3. 在導覽窗格中，選擇 Dashboard (儀表板)。
4. 選擇新增小工具。
5. 對於單一工作佇列，請選擇新增小工具。
6. 對於「Job 佇列」，請選取所需的工作佇列。
7. 對於 Job 狀態，選擇您要顯示的工作狀態。
8. (選擇性) 如果您不想顯示運算環境的內容，請關閉「展示連接的計算環境」。
9. 對於「計算環境性質」，請選取所需的性質。
10選擇新增。

CloudWatch 容器洞察

此 Widget 會顯示AWS Batch運算環境和工作的彙總指標。如需更多 Container Insights 的相關資訊，請參閱 CloudWatch 容器洞察。

若要新增此 Widget，請依照下列步驟執行。

1. 開啟 AWS Batch主控台。
2. 從導覽列中，選取AWS 區域您想要的。
3. 在導覽窗格中，選擇 Dashboard (儀表板)。
4. 選擇新增小工具。
5. 如需容器見解，請選擇新增小工具。
6. 對於運算環境，請選擇所需的運算環境。
7. 選擇新增。

Job 記錄

這個小工具會在一個方便的位置顯示與您的工作不同的日誌。如需工作記錄的詳細資訊，請參閱the section called “ Job 記錄”。

若要新增此 Widget，請依照下列步驟執行。

1. 開啟 AWS Batch主控台。
2. 從導覽列中，選取AWS 區域您想要的。
3. 在導覽窗格中，選擇 Dashboard (儀表板)。
4. 選擇新增小工具。
5. 對於 Job 記錄，請選擇新增小工具。
6. 在「Job ID」中，輸入所需工作的工作 ID。
7. 選擇新增。

設定方式 AWS Batch

如果您已經註冊了 Amazon Web Services（AWS）並且正在使用 Amazon Elastic Compute Cloud (Amazon EC2) 或 Amazon Elastic Container Service (Amazon ECS)，則可以很快使用 AWS Batch。這些服務的設定程序類似。這是因為在其運算環境中 AWS Batch 使用 Amazon ECS 容器執行個體。若要 AWS CLI 與配合使用 AWS Batch ，您必須使用支援最新 AWS Batch 功能的版本。 AWS CLI 如果在中看不到 AWS Batch 功能的支援 AWS CLI，請升級至最新版本。如需詳細資訊，請參閱 http://aws.amazon.com/cli/。

> **Note**
>
> 由於 AWS Batch 使用 Amazon EC2 的元件，因此您可以使用 Amazon EC2 主控台執行許多這些步驟。

完成下列工作以進行設定 AWS Batch。如果您已完成上述任何步驟，則可以直接跳到安裝 AWS CLI.

主題

- 註冊一個 AWS 帳戶
- 建立具有管理權限的使用者
- 為您的運算環境和容器執行個體建立 IAM 角色
- 建立金鑰對
- 建立 VPC
- 建立安全群組
- 安裝 AWS CLI

註冊一個 AWS 帳戶

如果您沒有 AWS 帳戶，請完成以下步驟來建立一個。

若要註冊成為 AWS 帳戶

1. 開啟 https://portal.aws.amazon.com/billing/signup。
2. 請遵循線上指示進行。

部分註冊程序需接收來電，並在電話鍵盤輸入驗證碼。

當您註冊時 AWS 帳戶，會建立AWS 帳戶根使用者一個。根使用者有權存取該帳戶中的所有 AWS 服務 和資源。安全性最佳做法是將管理存取權指派給使用者，並僅使用 root 使用者來執行需要 root 使用者存取權的工作。

AWS 註冊過程完成後，會向您發送確認電子郵件。您可以隨時登錄 https://aws.amazon.com/ 並選擇我的帳戶，以檢視您目前的帳戶活動並管理帳戶。

建立具有管理權限的使用者

註冊後，請保護 AWS 帳戶 AWS 帳戶根使用者、啟用和建立系統管理使用者 AWS IAM Identity Center，這樣您就不會將 root 使用者用於日常工作。

保護您的 AWS 帳戶根使用者

1. 選擇 Root 使用者並輸入您的 AWS 帳戶 電子郵件地址，以帳戶擁有者身分登入。AWS Management Console在下一頁中，輸入您的密碼。

   如需使用根使用者登入的說明，請參閱 AWS 登入 使用者指南中的以根使用者身分登入。
2. 若要在您的根使用者帳戶上啟用多重要素驗證 (MFA)。

   如需指示，請參閱《IAM 使用者指南》中的為 AWS 帳戶 根使用者啟用虛擬 MFA 裝置 (主控台)。

建立具有管理權限的使用者

1. 啟用 IAM Identity Center。

   如需指示，請參閱 AWS IAM Identity Center 使用者指南中的啟用 AWS IAM Identity Center。
2. 在 IAM 身分中心中，將管理存取權授予使用者。

   若要取得有關使用 IAM Identity Center 目錄 做為身分識別來源的自學課程，請參閱《使用指南》IAM Identity Center 目錄中的「以預設值設定使用AWS IAM Identity Center 者存取」。

以具有管理權限的使用者身分登入

- 若要使用您的 IAM Identity Center 使用者簽署，請使用建立 IAM Identity Center 使用者時傳送至您電子郵件地址的簽署 URL。

如需使用 IAM 身分中心使用者登入的說明，請參閱使用AWS 登入 者指南中的登入 AWS 存取入口網站。

指派存取權給其他使用者

1. 在 IAM 身分中心中，建立遵循套用最低權限許可的最佳做法的權限集。

   如需指示，請參閱《AWS IAM Identity Center 使用指南》中的「建立權限集」。
2. 將使用者指派給群組，然後將單一登入存取權指派給群組。

   如需指示，請參閱《AWS IAM Identity Center 使用指南》中的「新增群組」。

為您的運算環境和容器執行個體建立 IAM 角色

您的 AWS Batch 運算環境和容器執行個體需要 AWS 帳戶 憑證，才能代表您呼叫其他 AWS API。建立將這些登入資料提供給您的運算環境和容器執行個體的 IAM 角色，然後將該角色與您的運算環境建立關聯。

> **Note**
>
> 在主控台首次執行體驗中，會自動為您建立運 AWS Batch 算環境和容器執行個體角色。因此，如果您打算使用 AWS Batch 控制台，則可以繼續進行下一部分。如果您打算 AWS CLI 改用，請在使用服務連結角色 AWS Batch建立您的第一個計算環境Amazon ECS 執行個體角色之前完成所述程序。

建立金鑰對

AWS 使用公開金鑰加密技術來保護執行個體的登入資訊。Linux 執行個體 (例如 AWS Batch 運算環境容器執行個體) 沒有用於 SSH 存取的密碼。您需要使用金鑰對，以安全地登入執行個體。當建立運算環境時，您會先指定金鑰對名稱，然後再提供使用 SSH 登入時的私有金鑰。

如果您尚未建立 key pair，可以使用 Amazon EC2 主控台建立金鑰組。請注意，如果您計劃啟動多個執行個體 AWS 區域，請在每個區域中建立一個 key pair。如需區域的詳細資訊，請參閱 Amazon EC2 使用者指南中的區域和可用區域。

建立一組金鑰對

1. 前往 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
2. 從導覽列中，選 AWS 區域 取 key pair 的。無論您的位置為何，您都可以選取任何可用的區域：不過，金鑰配對是特定於某個區域。例如，如果您計劃在美國西部 (奧勒岡) 區域啟動執行個體，請在相同區域中為該執行個體建立 key pair。
3. 在導覽窗格中，選擇 Key Pairs (金鑰對)、Create Key Pair (建立金鑰對)。
4. 在 Create Key Pair (建立金鑰對) 對話方塊中，在 Key pair name (金鑰對名稱) 為新的金鑰對輸入名稱，然後選擇 Create (建立)。選擇您可以記住的名稱，例如您的使用者名稱，然後再加上區域名稱。`-key-pair`例如，me-key-pair-uswest2。
5. 您的瀏覽器會自動下載私有金鑰檔案。基礎檔案名稱為您所指定的金鑰對名稱，副檔名為 `.pem`。將私有金鑰檔案存放在安全的地方。

   > **Important**
   >
   > 這是您儲存私有金鑰檔案的唯一機會。您必須在啟動執行個體時提供 key pair 的名稱，並在每次連線至執行個體時提供對應的私密金鑰。

6. 如果您在 Mac 或 Linux 電腦上使用 SSH 用戶端連線到 Linux 執行個體，請使用下列指令來設定私密金鑰檔案的權限。這樣，只有你能閱讀它。

```
$ chmod 400 your_user_name-key-pair-region_name.pem
```

如需詳細資訊，請參閱 Amazon EC2 使用者指南中的 Amazon EC2 金鑰配對。

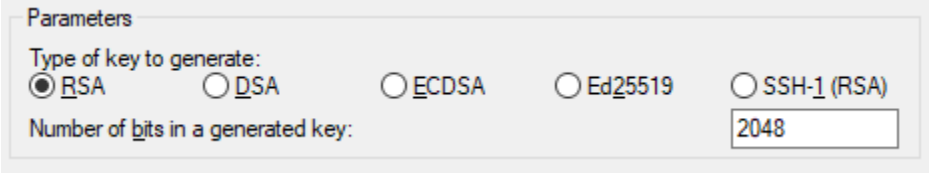
使用金鑰對連線至執行個體

若要從執行 Mac 或 Linux 的電腦連線至您的 Linux 執行個體，請使用 `.pem` 選項與您私有金鑰的路徑，將 `-i` 檔案指定給您的 SSH 用戶端。若要從執行 Windows 的電腦連線到 Linux 執行個體，請使用 MindTerm 或 PuTTY。如果您打算使用 PuTTY，請安裝它並使用下列程序將檔案轉換為`.pem`.ppk檔案。

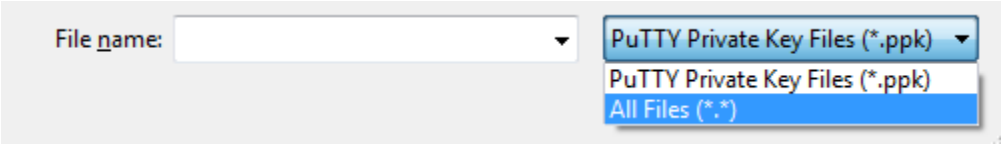
(選擇性) 若要準備使用 PuTTY 從 Windows 連線至 Linux 執行個體

1. 從 http://www.chiark.greenend.org.uk/~sgtatham/putty/ 下載並安裝 PuTTY。務必安裝整個套件。
2. 啟動 PuTTYgen（例如，從「開始」功能表中選擇「所有程式」、「膩子」和「Pu ttyGen」）。

3. 在 Type of key to generate (要產生的金鑰類型) 下，選擇 RSA (SSH-2 RSA)。如果您使用的是早期版本，PuTTYgen 選擇 SSH-2 RSA。



4. 選擇 Load (載入)。根據預設，PuTTYgen 只會顯示副檔名為 .ppk 的檔案。若要尋找您的 .pem 檔案，請選擇顯示所有類型之檔案的選項。



5. 選取您在先前程序中建立的私有金鑰檔案，然後選擇 Open (開啟)。選擇 OK (確定) 關閉確認對話方塊。
6. 選擇 Save private key (儲存私有金鑰)。PuTTYgen 會顯示有關儲存沒有密碼短語之金鑰的警告。選擇 Yes (是)。
7. 為您用於金鑰對的金鑰指定相同名稱。PuTTY 會自動新增 .ppk 副檔名。

建立 VPC

使用 Amazon Virtual Private Cloud (Amazon VPC)，您可以在已定義的虛擬網路中啟動 AWS 資源。強烈建議您在 VPC 中啟動容器執行個體。

如果您有預設 VPC，也可以略過本節並移至下一個工作建立安全群組。若要判斷您是否擁有預設 VPC，請參閱 Amazon EC2 使用者指南中的 Amazon EC2 主控台支援的平台

如需如何建立 Amazon VPC 的詳細資訊，請參閱 Amazon VPC 使用者指南中的「僅建立 VPC」。請參閱下表以決定要選取的選項。

選項	Value
要建立的資源	僅 VPC
名稱	可以選擇為 VPC 提供名稱。
IPv4 CIDR 區塊	IPv4 CIDR 手動輸入

選項	Value
	CIDR 區塊大小必須為介於 /16 和 /28 之間的大小。
IPv6 CIDR 區塊	無 IPv6 CIDR 區塊
租用	預設

如需有關 Amazon VPC 的詳細資訊，請參閱《Amazon VPC 使用者指南》中的什麼是 Amazon VPC？。

建立安全群組

安全群組就像是防火牆，用於關聯的運算環境容器執行個體，可在容器執行個體層級控制傳入及傳出流量。安全群組只能在建立該群組的 VPC 中使用。

您可以新增規則至安全群組，讓您從您的 IP 地址使用 SSH 連接到您的容器執行個體。您也可以新增允許任何位置之傳入和傳出 HTTP 和 HTTPS 存取的規則。依您的任務所需在開放連接埠新增任何規則。

請注意，如果您計劃在多個區域中啟動容器執行個體，則需要在每個區域中建立安全性群組。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的區域和可用區域。

> **Note**
>
> 您需要本機電腦的公有 IP 地址 (可以使用服務來取得)。例如，我們提供下列服務：http://checkip.amazonaws.com/ 或 https://checkip.amazonaws.com/。若要尋找其他能夠提供您 IP 地址的服務，請使用搜尋片語 "what is my IP address" (我的 IP 地址為何)。如果您透過網際網路服務供應商 (ISP) 或從沒有靜態 IP 位址的防火牆後方進行連線，請找出用戶端電腦所使用的 IP 位址範圍。

使用主控台建立安全群組

1. 在 https://console.aws.amazon.com/vpc/ 開啟 Amazon VPC 主控台。
2. 在導覽窗格中，選擇 Security Groups (安全群組)。
3. 選擇 Create Security Group (建立安全群組)。

4. 輸入安全群組的名稱和說明。您無法在建立安全群組之後變更安全群組的名稱和說明。
5. 從 VPC 中選擇 VPC。
6. (選擇性) 依預設，新安全群組只會從允許所有流量離開資源的輸出規則開始。您必須新增規則啟用任何傳入流量，或是限制傳出流量。

   AWS Batch 容器執行個體不需要開啟任何輸入連接埠。不過，您可能想要新增 SSH 規則。如此一來，您就可以登入容器執行個體，並使用 Docker 命令檢查工作中的容器。如果您希望容器執行個體託管執行 Web 伺服器的工作，也可以新增 HTTP 規則。完成下列步驟，以新增這些選用的安全群組規則。

   在 Inbound (內送) 標籤，建立以下規則然後選擇 Create (建立)：

   - 選擇 Add Rule (新增規則)。針對 Type (類型)，選擇 HTTP。針對 Source (來源)，選擇 Anywhere (隨處) (`0.0.0.0/0`)。
   - 選擇 Add Rule (新增規則)。針對 Type (類型)，選擇 SSH。在 [來源] 中，選擇 [自訂 IP]，並以無類別網域間路由 (CIDR) 標記法指定電腦或網路的公用 IP 位址。如果您的公司會分配某個範圍的地址，請指定整個範圍 (例如 `203.0.113.0/24`)。若要以 CIDR 標記法指定個別 IP 位址，請選擇 [我的 IP]。這會將路由前置詞新增`/32`至公用 IP 位址。

   > **Note**
   >
   > 基於安全理由，我們不建議您允許從所有 IP 位址 (`0.0.0.0/0`) 進行 SSH 存取您的執行個體，但只能用於測試目的，且只能在短時間內存取。

7. 您可以立即新增標籤，也可以稍後再新增。若要新增標籤，請選擇 Add new tag (新增標籤)，然後輸入標籤的索引鍵和值。
8. 選擇 Create Security Group (建立安全群組)。

若要使用命令列建立安全性群組，請參閱建立安全性群組 ()AWS CLI

如需有關安全性群組的詳細資訊，請參閱使用安全性群組。

安裝 AWS CLI

若要使用 AWS CLI 與 AWS Batch，請安裝最新 AWS CLI 版本。若要取得有關安裝 AWS CLI 或升級至最新版本的資訊，請參閱《AWS Command Line Interface 使用指南》中的〈安裝指 AWS 令行介面〉。

開始使用 AWS Batch

您可以使用 AWS Batch 首次執行精靈快速開始使用 AWS Batch。完成必要條件之後，您可以使用第一次執行精靈來建立計算環境、工作定義和工作佇列。

您也可以使用 AWS Batch 首次執行精靈來提交範例「Hello World」工作，以測試您的組態。如果您已有要在其中啟動的 Docker 映像檔 AWS Batch，則可以使用該映像檔建立工作定義。

必要條件

在啟動 AWS Batch 第一次執行精靈之前，請務必執行下列動作：

- 完成中描述的步驟設定方式 AWS Batch。
- 確認您 AWS 帳戶 具有必要的權限。

開始使用-Amazon EC2

Amazon Elastic Compute Cloud (Amazon EC2) – 在 AWS 雲端中提供可擴展的運算容量。使用 Amazon EC2 可減少前期所需的硬體投資，讓您更快速開發並部署應用程式。

您可使用 Amazon EC2 按需要啟動任意數量的虛擬伺服器，設定安全性和聯網功能以及管理儲存。使用 Amazon EC2 可擴展與縮減規模，以處理需求或熱門峰值的變更，從而降低您預測流量的需求。

建立運算環境

若要為 Amazon EC2 協調流程建立運算環境，請執行以下操作：

1. 打開AWS Batch 控制台首次運行嚮導。
2. 對於選取的協調類型，請選擇亞馬遜彈性運算雲端 (Amazon EC2)。
3. 選擇下一步。
4. 在 [名稱] 的 [運算環境組態] 區段中，為您的運算環境指定唯一的名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
5. 針對執行個體角色，請選擇已連接所需 IAM 許可的現有執行個體設定檔。此執行個體設定檔允許運算環境中的 Amazon ECS 容器執行個體對所需的 AWS API 操作進行呼叫。如需詳細資訊，請參閱 Amazon ECS 執行個體角色。

6. (選擇性) 標籤是指派給資源的標籤。若要新增標籤或 Amazon EC2 標籤，請展開標籤，然後選擇新增標籤。輸入金鑰-值配對，然後再次選擇 [新增標記]。

   > **Important**
   >
   > 如果您選擇 [新增標記]，則必須輸入金鑰-值配對，然後再次選擇 [新增標記] 或選擇 [移除標記]。

7. (選擇性) 在使用 Amazon EC2 Spot 執行個體的執行個體組態區段中，開啟使用 Spot 執行個體啟用。

8. (僅限現貨) 對於隨需價格上限百分比，請輸入您要為 Spot 資源支付的隨需定價的最大百分比。

9. (選用) (僅限競價型) 對於競價型叢集角色，請選擇現有的 Amazon EC2 Spot 叢集 IAM 角色以套用至您的競價型運算環境。如果您還沒有現有的 Amazon EC2 競價型叢集 IAM 角色，則必須先建立一個角色。如需詳細資訊，請參閱 Amazon EC2 現貨叢集角色。

   > **Important**
   >
   > 若要在建立時標記競價型執行個體，您的 Amazon EC2 競價型叢集 IAM 角色必須使用較新的 AmazonEC2 SpotFleetTaggingRole 受管政策。AmazonEC2 SpotFleetRole 受管政策沒有標記競價型執行個體所需的許可。如需詳細資訊，請參閱 建立時未標記競價型執行個體 及 the section called “標記您的 資源”。

10. 對於最低 vCPUs 數量，無論任務佇列需求為何，請選擇運算環境維護的最小 EC2 vCPUs 數量。

11. 對於所需的 vCPUs，請選擇您的運算環境隨其啟動的 EC2 vCPUs 數量。隨著任務佇列需求的增加， AWS Batch 增加所需的 vCPUs 數量並新增 EC2 執行個體。vCPUs 的數目可以增加到 vCPUs 的最大數目。隨著需求減少， AWS Batch 減少所需的 vCPUs 數量並移除執行個體。一直減少到 vCPUs 數目下限的數目。

12. 在 Minimum vCPUs (最小 vCPU 數) 中，選擇無論您的任務佇列需求為何，運算環境可擴展的最大 EC2 vCPU 數量。

13. 針對允許的執行個體類型，選擇可啟動的 Amazon EC2 執行個體類型。您可以指定例證族群以啟動這些族群中的任何例證類型 (例如`c5``c5n`、或`p3`)。或者，您可以指定族群內的特定大小 (例如`c5.8xlarge`)。金屬例證類型不在例證族群中。例如，`c5`不包括`c5.metal`. 您也可`optimal`以選擇選取符合工作佇列需求的R4執行個體類型 (從M4、和執行個體系列)。C4

> **Note**
>
> 在建立運算環境時，您為其選取的執行個體類型必須共用相同架構。例如，您無法在相同的運算環境中混合使用 x86 和 ARM 執行個體。

> **Note**
>
> AWS Batch 根據工作佇列中所需的數量調整 GPU 的比例。若要使用 GPU 排程，運算環境必須包含p2、、、、p3p4p5g3g3s、g4或g5系列中的執行個體類型。

> **Note**
>
> 目前，optimal使用、和例證族群M4中的C4R4例證類型。如果沒有來自這些例證族群的例證類型，則會使用C5M5、和R5例證族群中 AWS 區域 的例證類型。

14. 展開 Additional configuration (其他組態)。
15. (選擇性) 在「放置」群組中，輸入放置群組名稱，以將計算環境中的資源分組。
16. (選擇性) 對於 EC2 金鑰組，請在連線至執行個體時選擇公開金鑰和私密金鑰組做為安全登入資料。如需有關 Amazon EC2 金鑰配對的詳細資訊，請參閱 Amazon EC2 金鑰配對和 Linux 執行個體。
17. 如為配置策略，從允許的執行個體類型清單中選取執行個體類型時，選取要使用的配置策略。對於 EC2 隨需運算環境而言，最佳 _ FIT_漸進式通常是更好的選擇，而針對 EC2 競價型運算環境進行了優化的最佳選擇。如需詳細資訊，請參閱 the section called “分配策略”。
18. (選擇性) 對於 EC2 組態，請選擇新增 EC2 組態。選擇映像類型和映像 ID 覆寫值，以提供在運算環境中 AWS Batch 為執行個體選取 Amazon 機器映像 (AMI) 的資訊。如果未針對每個映像類型指定影像 ID 覆寫，請 AWS Batch 選取最近的 Amazon ECS 最佳化 AMI。如果未指定映像類型，則對於非 GPU、非重力執行個 AWS 體，預設值為 Amazon Linux 2。

> **Important**
>
> 若要使用自訂 AMI，請選擇映像類型，然後在 [映像 ID 覆寫] 方塊中輸入自訂 AMI ID。

Amazon Linux 2

所有以 AWS 重力為基礎的執行個體系列 (例如、、和T4g) 的預設值 C6g M6gR6g，並且可用於所有非 GPU 執行個體類型。

Amazon Linux 2 (GPU)

所有 GPU 執行個體系列的預設值 (例如P4和G4)，可用於所有非 AWS 重力型執行個體類型。

Amazon Linux

可用於非 GPU、非重 AWS 力子執行個體系列。Amazon Linux AMI 的標準支持已經結束。如需詳細資訊，請參閱 Amazon Linux AMI。

> **Note**
>
> 您為運算環境選擇的 AMI 必須與您要用於該運算環境的執行個體類型架構相符。例如，如果您的運算環境使用A1執行個體類型，則您選擇的計算資源 AMI 必須支援Arm執行個體。Amazon ECS 出售 Amazon ECS x86 的兩個Arm版本優化 Amazon Linux 2 AMI。有關詳情，請參閱 Amazon ECS Amazon 彈性容器服務開發人員指南中的 Amazon ECS 優化亞馬遜 Linux 2 AMI。

19. (選擇性) 對於啟動範本，請選取現有的 Amazon EC2 啟動範本來設定您的運算資源。系統會自動填入範本的預設版本。如需詳細資訊，請參閱 啟動範本支援。

> **Note**
>
> 在啟動範本中，您可以指定您建立的自訂 AMI。

20. (選用) 對於 Launch template version (啟動範本版本)，請輸入 $Default、$Latest 或指定要使用的版本號碼。

> **Important**
>
> 建立運算環境後，即使啟動範本的$Default或版本已更新，使用的啟動範$Latest本版本也不會變更。若要使用新的啟動範本版本，請先建立新的計算環境，然後將新的計算環境新增至現有的工作佇列。然後，從工作佇列中移除舊的計算環境，並刪除舊的計算環境。

21. 在「網路設定」區段中：

    a. 對於 V irtual Private Cloud (VPC) (VPC) ID，請選擇一個 Amazon VPC。

    b. 對於子網路， AWS 帳戶 會列出您的子網路。如果您想要建立一組自訂的子網路，請選擇 [清除子網路]，然後選擇您想要的子網路。

    > **Important**
    >
    > 運算資源必須透過虛擬私人雲端端點或多個公有 IP 地址與 Amazon ECS VPC 人雲端節點通訊。如需詳細資訊，請參閱 Amazon ECS 介面 VPC 端點 () AWS PrivateLink。如果您的執行個體未設定 VPC 端點或公用 IP 位址，您可以使用網路位址轉譯 (NAT)。如需 NAT 的詳細資訊，請參閱 NAT 閘道和建立虛擬私有雲 .

    c. 對於安全群組，請選擇要與執行個體建立關聯的 Amazon EC2 安全群組。如果您要建立一組自訂的安全性群組，請選擇 [清除安全性群組]。然後，選擇您想要的安全性群組。

22. 選擇下一步。

建立工作佇列

工作佇列會儲存您提交的工作，直到 AWS Batch 排程器在計算環境中的資源上執行工作為止。如需更多資訊，請參閱 Job 佇列

若要為 Amazon EC2 協調流程建立任務佇列，請執行以下操作：

1. 在 [名稱] 的 [Job 佇列組態] 區段中，指定運算環境的唯一名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
2. 在「優先順序」中，為工作佇列輸入介於 0 到 100 之間的整數。

   > **Important**
   >
   > 排程器會為較高的整數值指派較高的優先順 AWS Batch 序。

3. 選擇下一步。

建立任務定義

AWS Batch 工作定義指定工作的執行方式。即使每個工作都必須參照工作定義，但在執行階段仍可覆寫工作定義中指定的許多參數。

若要建立工作定義：

1. 在「一般組態」區段中：

   a. 在 [名稱] 的 [一般組態] 區段中，為您的運算環境指定唯一的名稱。名稱最多可包含 128 個字元。名稱可以包含大寫和小寫字母、數字、連字號 (-) 和底線 (_)。

   b. (選擇性) 對於執行逾時，請輸入未完成工作在後終止的時間長度 (以秒為單位)。

   > **Important**
   >
   > 最小逾時時間為 60 秒。

   c. (選擇性) 標籤是指派給資源的標籤。若要加入標籤，請展開「標籤」，然後選擇「加入標籤」。輸入金鑰-值配對，然後再次選擇 [新增標記]。

   > **Important**
   >
   > 如果您選擇 [新增標記]，則必須輸入金鑰-值配對，然後再次選擇 [新增標記] 或選擇 [移除標記]。

   d. (選擇性) 開啟傳播標籤以將標籤傳播至 Amazon 彈性容器服務任務。

2. 在「容器設定」區段中：

   a. 在「影像」中，輸入用來啟動容器的映像檔名稱。根據預設，Docker Hub 登錄中的所有映像都可以使用。您還可以在存儲庫 URL /圖像：標籤格式指定其他存儲庫。參數的長度最多可達 255 個字元。參數可以包含大寫和小寫字母、數字、連字號 (-)、底線 (_)、冒號 (:)、句點 (.)、正斜線 (/) 和數字符號 (#)。參數會對應到 Docker 遠端 API 的 [建立容器] 區段`Image`中，以及的`IMAGE`docker run參數。

   > **Note**
   >
   > Docker映像檔架構必須符合其排程所在運算資源的處理器架構。例如，Arm基於Docker映像只能Arm根據計算資源執行。

- Amazon ECR 公用儲存庫中的映像會使用完整`registry/repository[:tag]`或`registry/repository[@digest]`命名慣例 (例如`public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*)。
- Amazon ECR 儲存庫中的映像會使用完整的`registry/repository:tag`命名慣例 (例如 *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*)。
- Docker Hub 上官方儲存庫中的映像，使用的是單一名稱 (例如，`ubuntu` 或 `mongo`)。
- Docker Hub 上的其他儲存庫中的映像要求使用組織名稱 (例如，`amazon/amazon-ecs-agent`)。
- 其他線上儲存庫中的映像更進一步要求使用網域名稱 (例如，`quay.io/assemblyline/ubuntu`)。

b. 在 Command (命令) 中，指定要傳送至容器的命令。此參數會映射至 Docker Remote API 的建立容器區段中的 `Cmd` 以及 docker run 的 `COMMAND` 參數。如需 Docker `CMD` 參數的詳細資訊，請參閱 https://docs.docker.com/engine/reference/builder/#cmd。

> **Note**
>
> 您可以在命令中使用替換參數預設值及預留位置。如需詳細資訊，請參閱 參數。

c. (選擇性) 對於執行角色，請指定 IAM 角色，以授與 Amazon ECS 容器代理程式的權限，以代表您進行 AWS API 呼叫。此功能使用 Amazon ECS 身分與存取權管理角色執行任務。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 Amazon ECS 任務執行 IAM 角色。

d. (選擇性) 對於「Job 角色」設定，請選擇具有 AWS API 許可的 IAM 角色。此功能使用 Amazon ECS 身分與存取權管理角色執行任務。如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的任務 IAM 角色。

> **Note**
>
> 此處僅顯示具有 Amazon 彈性容器服務任務角色信任關係的角色。如需為任務建立 IAM 角色的詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的為任務建立 IAM 角色和政策。 AWS Batch

e. (選擇性) 您可以將參數作為鍵值對映新增至工作定義，以覆寫工作定義預設值。若要新增參數：

- 對於「參數」，選擇「新增參數」 輸入鍵值對，然後再次選擇「新增參數」。

> **Important**
>
> 如果您選擇 [新增參數]，則必須至少設定一個參數，或選擇 [移除參數]。

f. 在 vCPUs 的環境組態區段中，指定要為容器保留的 vCPUs 數目。此參數會映射到 Docker Remote API 的建立容器區段中的 `CpuShares` 以及 docker run 的 `--cpu-shares` 選項。每個 vCPU 相當於 1,024 個 CPU 共用。

g. 針對記憶體，指定要呈現給工作容器的記憶體硬性限制 (單位為 MiB)。如果您的容器嘗試超過此處指定的記憶體，則會停止容器。此參數會映射到 Docker Remote API 的建立容器區段中的 `Memory` 以及 docker run 的 `--memory` 選項。

h. 在 GPU 數目中，選擇要為容器預留的 GPU 數目。

i. (選擇性) 對於環境變數組態，請選擇新增環境變數以新增要傳遞至容器的環境變數。此參數會映射到 Docker Remote API 的建立容器區段中的 `Env` 以及 docker run 的 `--env` 選項。

j. (選擇性) 對於密碼，請選擇新增密碼，將密碼新增為名稱-值配對。這些機密會暴露在容器中。如需詳細資訊，請參閱。 Job 定義參數 ContainerProperties

k. (選擇性) 在「Linux 組態」區段中：

   i. 在 User (使用者) 中，輸入要在容器內使用的使用者名稱。此參數會映射到 Docker Remote API 的建立容器區段中的 `User` 以及 docker run 的 `--user` 選項。

   ii. 若要為工作容器提高主機執行個體的權限 (類似於`root`使用者)，請將 [已授權] 滑桿向右拖曳。此參數會映射到 Docker Remote API 的建立容器區段中的 `Privileged` 以及 docker run 的 `--privileged` 選項。

   iii. 開啟啟用初始化程序以在容器內執行`init`程序。此過程轉發信號並重新執行過程。

l. （可選）在「文件系統配置」部分中：

   i. 開啟啟用唯讀檔案系統以移除磁碟區的寫入權限。

   ii. 在「共用記憶體大小」中，輸入`/dev/shm`磁碟區的大小 (以 MiB 為單位)。

   iii. 在「最大交換大小」中，輸入容器可以使用的交換記憶體總量 (以 MiB 為單位)。

   iv. 對於「交換」，請輸入介於 0 到 100 之間的值，以指示容器的交換行為。如果您未指定值並啟用交換，則值預設為 60。若要取得更多資訊，請參閱〈〉中的交換。 Job 定義參數 ContainerProperties

   v. (選擇性) 展開其他組態。

vi. 對於 Tmpfs，請選擇「新增 tmpfs」以新增裝載。`tmpfs`

vii. 對於「裝置」，請選擇「新增裝置」以新增裝置：

A. 針對 Container path (容器路徑)，指定容器執行個體中的路徑，以公開對應到主機執行個體的裝置。如果保持此空白，則會在容器中使用主機路徑。

B. 針對 Host path (主機路徑)，指定主機執行個體中的裝置的路徑。

C. 在「權限」中，選擇要套用至裝置的一或多個權限。可用的權限包括「讀取」、「寫入」和「MCNOD」。

viii. (選擇性) 對於 Ulimit 組態，請選擇新增 ulimit 以新增容器的`ulimits`值。輸入 [名稱]、[軟限制] 及 [硬性限制] 值，然後選擇 [新增 ulimit]。

3. 選擇下一步。

建立任務。

若要建立工作，請執行下列動作：

1. 在「名稱」的「Job 組態」段落中，指定工作的唯一名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
2. 選擇下一步。

檢閱和建立

在 [檢閱並建立] 頁面上，檢閱設定步驟。如需變更，請選擇 Edit (編輯)。完成後，選擇 [建立資源]。

開始使用-Fargate

AWS Fargate 會啟動並擴展運算，以便與您為容器指定的資源需求密切相符。有了 Fargate，您不需要過度佈建或支付額外的伺服器費用。如需詳細資訊，請參閱 Fargate。

建立運算環境

若要建立 Fargate 協調流程的計算環境，請執行下列動作：

1. 打開AWS Batch 控制台首次運行嚮導。
2. 針對 [選取協調流程類型]，選擇 [F argate]。

3. 選擇下一步。
4. 在 [名稱] 的 [運算環境組態] 區段中，為您的運算環境指定唯一的名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
5. (選擇性) 標籤是指派給資源的標籤。若要加入標籤，請展開「標籤」，然後選擇「加入標籤」。輸入金鑰-值配對，然後再次選擇 [新增標記]。

   > **Important**
   >
   > 如果您選擇 [新增標記]，則必須輸入金鑰-值配對，然後再次選擇 [新增標記] 或選擇 [移除標記]。

6. (選擇性) 在使用 Fargate Spot 容量的執行個體組態區段中，開啟使用 Spot 執行個體啟用。
7. 針對 vCPUs 數目上限，輸入執行個體可使用的 vCPUs 數目上限。
8. 在「網路設定」區段中：

   a. 對於 V irtual Private Cloud (VPC) (VPC) ID，請選擇一個 Amazon VPC。

   b. 對於子網路， AWS 帳戶 會列出您的子網路。如果您想要建立一組自訂的子網路，請選擇 [清除子網路]，然後選擇您想要的子網路。

      > **Important**
      >
      > 運算資源必須透過虛擬私人雲端端點或多個公有 IP 地址與 Amazon ECS VPC 人雲端節點通訊。如需詳細資訊，請參閱 Amazon ECS 介面 VPC 端點 () AWS PrivateLink。如果您的執行個體未設定 VPC 端點或公用 IP 位址，您可以使用網路位址轉譯 (NAT)。如需 NAT 的詳細資訊，請參閱 NAT 閘道和建立虛擬私有雲 .

   c. 對於安全群組，請選擇要與執行個體建立關聯的 Amazon EC2 安全群組。如果您要建立一組自訂的安全性群組，請選擇 [清除安全性群組]。然後，選擇您想要的安全性群組。

9. 選擇下一步。

建立工作佇列

工作佇列會儲存您提交的工作，直到 AWS Batch 排程器在計算環境中的資源上執行工作為止。若要建立工作佇列：

若要建立 Fargate 協調流程的工作佇列，請執行下列動作：

1. 在 [名稱] 的 [Job 佇列組態] 區段中，指定運算環境的唯一名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
2. 在「優先順序」中，為工作佇列輸入介於 0 到 100 之間的整數。

   > **Important**
   >
   > 排程器會為較高的整數值指派較高的優先順 AWS Batch 序。

3. 選擇下一步。

建立任務定義

若要建立工作定義：

1. 在「一般組態」區段中：

   a. 在名稱中，輸入自訂工作定義名稱。

      在 [名稱] 的 [一般組態] 區段中，為您的運算環境指定唯一的名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。

   b. (選擇性) 對於執行逾時，請輸入未完成工作在後終止的時間長度 (以秒為單位)。

      > **Important**
      >
      > 最小逾時時間為 60 秒。

   c. (選擇性) 標籤是指派給資源的標籤。若要加入標籤，請展開「標籤」，然後選擇「加入標籤」。輸入金鑰-值配對，然後再次選擇 [新增標記]。

      > **Important**
      >
      > 如果您選擇 [新增標記]，則必須輸入金鑰-值配對，然後再次選擇 [新增標記] 或選擇 [移除標記]。

   d. (選擇性) 開啟傳播標籤以將標籤傳播至 Amazon 彈性容器服務任務。

2. 在 Fargate 平台配置部分：

   a. (選擇性) 對於 Fargate 平台版本，請輸入您想要的特定執行階段環境。

b. 針對「執行階段」平台，請選取 LINUX 或視窗。

c. (僅限 Windows) 針對「作業系統系列」，請選取作業系統。

d. 對於 CPU 架構，請選取您想要的 CPU 架構。

e. (選擇性) 開啟指派公用 IP 以指派公用 IP 位址。

f. 對於暫時儲存，請輸入您想要的暫時儲存空間數量。

> **Note**
>
> 依預設，會使用 20 GiB 的暫時儲存空間。若要使用額外的暫時儲存空間，請輸入介於 21 GiB 和 100 GiB 之間的值。

g. 對於執行角色，請選擇可讓 Amazon Elastic Container Service (Amazon ECS) 代理程式代表您撥 AWS 打電話的任務執行角色。例如，您可以選擇「ecsTaskExecution角色」。

3. 在「容器設定」區段中：

a. 在「影像」中，輸入用來啟動容器的映像檔名稱。根據預設，Docker Hub 登錄中的所有映像都可以使用。您還可以在存儲庫 URL /圖像：標籤格式指定其他存儲庫。參數的長度最多可達 255 個字元。可包含大寫及小寫字母、數字、連字號 (-)、底線 (_)、冒號 (:)、句點 (.)、斜線 (/) 和數字符號 (#)。參數會對應到 Docker 遠端 API 的 [建立容器] 區段Image中，以及的IMAGEdocker run參數。

> **Note**
>
> Docker映像檔架構必須符合其排程所在運算資源的處理器架構。例如，Arm基於Docker映像只能Arm根據計算資源執行。

- Amazon ECR 公用儲存庫中的映像會使用完整`registry/repository[:tag]`或`registry/repository[@digest]`命名慣例 (例如`public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*)。
- Amazon ECR 儲存庫中的映像會使用完整的`registry/repository:tag`命名慣例 (例如 *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*)。
- Docker Hub 上官方儲存庫中的映像，使用的是單一名稱 (例如，`ubuntu` 或 `mongo`)。
- Docker Hub 上的其他儲存庫中的映像要求使用組織名稱 (例如，`amazon/amazon-ecs-agent`)。

- 其他線上儲存庫中的映像更進一步要求使用網域名稱 (例如，`quay.io/assemblyline/ubuntu`)。

b. 在 Command (命令) 中，指定要傳送至容器的命令。此參數會映射至 Docker Remote API 的建立容器區段中的 `Cmd` 以及 docker run 的 `COMMAND` 參數。如需 Docker `CMD` 參數的詳細資訊，請參閱 https://docs.docker.com/engine/reference/builder/#cmd。

> **Note**
> 您可以在命令中使用替換參數預設值及預留位置。如需詳細資訊，請參閱 參數。

> **Tip**
> 選擇「資訊」以檢閱 Bash 和 JSON 程式碼範例。

c. (選擇性) 您可以將參數作為鍵值對映新增至工作定義，以覆寫工作定義預設值。若要新增參數：

- 對於「參數」，選擇「新增參數」 輸入鍵值對，然後再次選擇「新增參數」。

> **Important**
> 如果您選擇 [新增參數]，則必須至少設定一個參數，或選擇 [移除參數]。

d. (選擇性) 在「Job 角色設定」的「環境設定」區段中，選擇提供 AWS API 使用權限的 IAM 角色。

e. 在 vCPUs 的環境組態區段中，指定要為容器保留的 vCPUs 數目。此參數會映射到 Docker Remote API 的建立容器區段中的 `CpuShares` 以及 docker run 的 `--cpu-shares` 選項。每個 vCPU 相當於 1,024 個 CPU 共用。

f. 針對記憶體，指定要呈現給工作容器的記憶體硬性限制 (單位為 MiB)。如果您的容器嘗試超過此處指定的記憶體，則會停止容器。此參數會映射到 Docker Remote API 的建立容器區段中的 `Memory` 以及 docker run 的 `--memory` 選項。

g. (選擇性) 對於環境變數，請選擇新增環境變數以新增要傳遞至容器的環境變數。此參數會映射到 Docker Remote API 的建立容器區段中的 `Env` 以及 docker run 的 `--env` 選項。

4. 選擇下一步。

建立任務。

若要建立遠端工作，請執行下列操作：

1. 在「名稱」的「Job 組態」段落中，指定工作的唯一名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
2. 選擇下一步。

檢閱和建立

在 [檢閱並建立] 頁面上，檢閱設定步驟。如需變更，請選擇 Edit (編輯)。完成後，選擇 [建立資源]。

開始使 AWS Batch 用 Amazon EKS

AWS Batch 在 Amazon EKS 上是一種受管服務，用於將批次工作負載排程和擴展到現有 Amazon EKS 叢集。 AWS Batch 不會代表您建立、管理或執行 Amazon EKS 叢集的生命週期操作。 AWS Batch 協調流程可擴展和縮減由這些節點管理的節點， AWS Batch 並在這些節點上執行網繭。

AWS Batch 不會接觸與 Amazon EKS 叢集中 AWS Batch 運算環境無關的節點、auto 調整節點群組或網繭生命週期。 AWS Batch 為了有效運作，其服務連結角色需要現有 Amazon EKS 叢集中以Kubernetes角色為基礎的存取控制 (RBAC) 許可。如需詳細資訊，請參閱文件Kubernetes中的使用 RBAC 授權。

AWS Batch 需要一個Kubernetes命名空間，它可以將網繭設定為 AWS Batch 工作範圍。我們建議使用專用的命名空間，將 AWS Batch 網繭與其他叢集工作負載隔離。

獲 AWS Batch 得 RBAC 存取權並建立命名空間之後，您可以使用 API 作業將該 Amazon EKS 叢集與 AWS Batch 運算環境建立關聯。CreateComputeEnvironment任務佇列可以與這個新的 Amazon EKS 運算環境建立關聯。 AWS Batch 使用 SubmitJobAPI 作業會根據 Amazon EKS 任務定義提交任務至任務佇列。 AWS Batch 然後啟動 AWS Batch 受管理的節點，並將作業從工作佇列作為Kubernetes網繭放置到與 AWS Batch 計算環境關聯的 EKS 叢集中。

以下各節介紹了如何 AWS Batch 在 Amazon EKS 上進行設置。

內容

必要條件

在開始本教學課程之前，您必須安裝並設定建立和管理和 Amazon EKS 資源所需的下列工具 AWS Batch 和資源。

- AWS CLI：適用於使用 AWS 服務 (包括 Amazon EKS) 的命令列工具。本指南要求您使用 2.8.6 或更新版本或 1.26.0 或更新版本。若要取得更多資訊，請參閱《AWS Command Line Interface 使用指南》 AWS CLI中的〈安裝、更新和解除安裝〉。安裝之後 AWS CLI，我們建議您也對其進行配置。若要取得更多資訊，請參閱《使用指南》aws configure中的AWS Command Line Interface〈快速配置〉。
- **kubectl**：命令列工具，適用於使用 Kubernetes 叢集。本指南要求您使用版本 1.23 或更新版本。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的安裝或更新 kubectl。
- **eksctl**— 使用 Amazon EKS 叢集的命令列工具，可自動執行許多個別任務。本指南要求您使用版本 0.115.0 或更新版本。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的安裝或更新 eksctl。
- 必要的 IAM 許可 — 您使用的 IAM 安全主體必須具有使用 Amazon EKS IAM 角色和服務連結角色的許可 AWS CloudFormation，以及 VPC 和相關資源。如需詳細資訊，請參閱 IAM 使用者指南中的適用於 Amazon Elastic Kubernetes Service 的動作、資源和條件金鑰和使用服務連結角色。您必須以同一位使用者的身分完成本指南中的所有步驟。

- 建立 Amazon EKS 叢集 — 如需詳細資訊，請參閱 Amazon EKS 使用者指南eksctl中的開始使用 Amazon EKS。

> **Note**
>
> AWS Batch 僅支援具有公開存取權且可供公用網際網路存取的 API 伺服器端點的 Amazon EKS 叢集。根據預設，Amazon EKS 叢集 API 伺服器端點具有公開存取權。如需詳細資訊，請參閱 Amazon EKS 叢集端點存取控制 (英文) 中的 Amazon EKS 叢集端點存取控制。

> **Note**
>
> AWS Batch 不會為 CoreDNS 或其他部署網繭提供受管節點協調流程。如果您需要 CoredN，請參閱 Amazon EKS 使用者指南中的新增 CoreDNS Amazon EKS 附加元件。或者，用`eksctl create cluster create`來建立叢集，依預設會包含 CoreDNS。

- 權限 — 呼叫 CreateComputeEnvironmentAPI 作業以建立使用 Amazon EKS 資源之運算環境的使用者需要 `eks:DescribeCluster` API 作業的許可。使用 AWS Management Console 來建立使用 Amazon EKS 資源的運算資源需要`eks:DescribeCluster`和`eks:ListClusters`的許可。

步驟 1：準備您的 Amazon EKS 叢集 AWS Batch

所有步驟都是必需的。

1. 為 AWS Batch 工作建立專用的命名空間

   用`kubectl`於建立新的命名空間。

```
$ namespace=my-aws-batch-namespace
$ cat - <<EOF | kubectl create -f -
{
  "apiVersion": "v1",
  "kind": "Namespace",
  "metadata": {
    "name": "${namespace}",
    "labels": {
      "name": "${namespace}"
    }
  }
```

```
}
EOF
```

輸出：

```
namespace/my-aws-batch-namespace created
```

2. 透過角色型存取控制 (RBAC) 啟用存取

用kubectl於為叢集建立Kubernetes角色，以 AWS Batch 允許觀看節點和網繭，以及繫結角色。您必須針對每個 EKS 叢集執行一次此動作。

> **Note**
>
> 如需有關使用 RBAC 授權的詳細資訊，請參閱《使用指南》中的〈使用 RBAC 授權〉。Kubernetes

```
$ cat - <<EOF | kubectl apply -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: aws-batch-cluster-role
rules:
  - apiGroups: [""]
    resources: ["namespaces"]
    verbs: ["get"]
  - apiGroups: [""]
    resources: ["nodes"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["configmaps"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["apps"]
    resources: ["daemonsets", "deployments", "statefulsets", "replicasets"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["clusterroles", "clusterrolebindings"]
```

```
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: aws-batch-cluster-role-binding
subjects:
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: aws-batch-cluster-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

輸出：

```
clusterrole.rbac.authorization.k8s.io/aws-batch-cluster-role created
clusterrolebinding.rbac.authorization.k8s.io/aws-batch-cluster-role-binding created
```

建立命名空間範圍的Kubernetes角色，AWS Batch 以管理和生命週期網繭並繫結它。您必須為每個唯一命名空間執行一次此操作。

```
$ namespace=my-aws-batch-namespace
$ cat - <<EOF | kubectl apply -f - --namespace "${namespace}"
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: aws-batch-compute-environment-role
  namespace: ${namespace}
rules:
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["create", "get", "list", "watch", "delete", "patch"]
  - apiGroups: [""]
    resources: ["serviceaccounts"]
    verbs: ["get", "list"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["roles", "rolebindings"]
    verbs: ["get", "list"]
---
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: aws-batch-compute-environment-role-binding
  namespace: ${namespace}
subjects:
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: aws-batch-compute-environment-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

輸出：

```
role.rbac.authorization.k8s.io/aws-batch-compute-environment-role created
rolebinding.rbac.authorization.k8s.io/aws-batch-compute-environment-role-binding
 created
```

更新設Kubernetesaws-auth定對應，以將先前的 RBAC 權限對應至服務連結角色。 AWS Batch

```
$ eksctl create iamidentitymapping \
    --cluster my-cluster-name \
    --arn "arn:aws:iam::<your-account>:role/AWSServiceRoleForBatch" \
    --username aws-batch
```

輸出：

```
2022-10-25 20:19:57 [#]  adding identity "arn:aws:iam::<your-account>:role/
AWSServiceRoleForBatch" to auth ConfigMap
```

> **Note**
>
> 路徑aws-service-role/batch.amazonaws.com/已從服務連結角色的 ARN 中移除。這是因為aws-auth配置對映有問題。如需詳細資訊，請參閱中的路徑包含在其 ARN 中時，具有路徑的角色無法運作aws-authconfigmap。

步驟 2：建立 Amazon EKS 運算環境

AWS Batch 運算環境會定義運算資源參數，以符合批次工作負載的需求。在受管運算環境中，可 AWS Batch 協助您管理 Amazon EKS 叢集內運算資源 (Kubernetes節點) 的容量和執行個體類型。這是根據您在建立運算環境時定義的計算資源規格而定。您可以使用 EC2 隨需執行個體或 EC2 競價型執行個體。

現在AWSServiceRoleForBatch服務連結角色可以存取 Amazon EKS 叢集，您可以建立 AWS Batch 資源。首先，建立一個指向 Amazon EKS 叢集的運算環境。

```
$ cat <<EOF > ./batch-eks-compute-environment.json
{
  "computeEnvironmentName": "My-Eks-CE1",
  "type": "MANAGED",
  "state": "ENABLED",
  "eksConfiguration": {
    "eksClusterArn": "arn:aws:eks:<region>:123456789012:cluster/<cluster-name>",
    "kubernetesNamespace": "my-aws-batch-namespace"
  },
  "computeResources": {
    "type": "EC2",
    "allocationStrategy": "BEST_FIT_PROGRESSIVE",
    "minvCpus": 0,
    "maxvCpus": 128,
    "instanceTypes": [
        "m5"
    ],
    "subnets": [
        "<eks-cluster-subnets-with-access-to-internet-for-image-pull>"
    ],
    "securityGroupIds": [
        "<eks-cluster-sg>"
    ],
    "instanceRole": "<eks-instance-profile>"
  }
}
EOF
$ aws batch create-compute-environment --cli-input-json file://./batch-eks-compute-environment.json
```

備註

- 不應指定`serviceRole`參數，則會使用 AWS Batch 服務連結角色。 AWS Batch 在 Amazon EKS 上僅支援 AWS Batch 服務連結角色。
- Amazon EKS 運算環境僅`BEST_FIT_PROGRESSIVE`支援`SPOT_CAPACITY_OPTIMIZED`、和`SPOT_PRICE_CAPACITY_OPTIMIZED`配置策略。

> **Note**
>
> 我們建議您使用`SPOT_PRICE_CAPACITY_OPTIMIZED`而不是`SPOT_CAPACITY_OPTIMIZED`在大多數情況下使用。

- 有關詳情`instanceRole`，請參閱 Amazon EKS 使用者指南中的建立 Amazon EKS 節點 IAM 角色和啟用 IAM 主體存取您的叢集。如果您使用的是網繭聯網，請參閱 Amazon EKS 使用者指南中的設定 Amazon VPC CNI 外掛程式以Kubernetes將 IAM 角色用於服務帳戶。
- 取得`subnets`參數的工作子網路的一種方法是使用 Amazon EKS 受管節點群組建立的公有子網路，這些公用子網路是在建立 Amazon EKS 叢集`eksctl`時所建立的。否則，請使用具有支援提取影像之網路路徑的子網路。
- 該`securityGroupIds`參數可以使用與 Amazon EKS 叢集相同的安全群組。此命令會擷取叢集的安全性群組識別碼。

```
$ eks describe-cluster \
    --name <cluster-name> \
    --query cluster.resourcesVpcConfig.clusterSecurityGroupId
```

- 維護 Amazon EKS 運算環境是一項共同的責任。如需詳細資訊，請參閱 Kubernetes節點的共同責任。

> **Important**
>
> 在繼續之前，請務必確認運算環境狀況良好。DescribeComputeEnvironmentsAPI 操作可用於執行此操作。
>
> ```
> $ aws batch describe-compute-environments --compute-environments My-Eks-CE1
> ```
>
> 確認參`status`數不是`INVALID`。如果是，請查看原因的`statusReason`參數。如需詳細資訊，請參閱 疑難排 AWS Batch。

步驟 3：建立工作佇列並連接運算環境

```
$ aws batch describe-compute-environments --compute-environments My-Eks-CE1
```

提交至此新任務佇列的任務會在加入與運算環境相關聯之 Amazon EKS 叢集的 AWS Batch 受管節點上做為網繭執行。

```
$ cat <<EOF > ./batch-eks-job-queue.json
 {
    "jobQueueName": "My-Eks-JQ1",
    "priority": 10,
    "computeEnvironmentOrder": [
      {
        "order": 1,
        "computeEnvironment": "My-Eks-CE1"
      }
    ]
  }
EOF
$ aws batch create-job-queue --cli-input-json file://./batch-eks-job-queue.json
```

步驟 4：建立工作定義

```
$ cat <<EOF > ./batch-eks-job-definition.json
{
  "jobDefinitionName": "MyJobOnEks_Sleep",
  "type": "container",
  "eksProperties": {
    "podProperties": {
      "hostNetwork": true,
      "containers": [
        {
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "sleep",
            "60"
          ],
          "resources": {
            "limits": {
              "cpu": "1",
              "memory": "1024Mi"
            }
```

```
          }
        }
      ],
      "metadata": {
        "labels": {
          "environment": "test"
        }
      }
    }
  }
}
EOF
$ aws batch register-job-definition --cli-input-json file://./batch-eks-job-
definition.json
```

備註

- 僅支援單一容器工作。
- cpu和memory參數有一些考量。如需詳細資訊，請參閱 Amazon EKS AWS Batch 上的記憶體和 vCPU 考量。

步驟 5：提交工作

```
$ aws batch submit-job --job-queue My-Eks-JQ1 \
    --job-definition MyJobOnEks_Sleep --job-name My-Eks-Job1
$ aws batch describe-jobs --job <jobId-from-submit-response>
```

備註

- 僅支援單一容器工作。
- 請確定您熟悉cpu和memory參數的所有相關考量。如需詳細資訊，請參閱 Amazon EKS AWS Batch 上的記憶體和 vCPU 考量。
- 如需在 Amazon EKS 資源上執行任務的詳細資訊，請參閱Amazon EKS 工作機會。

(選擇性) 提交含覆寫項目的工作

此工作會覆寫傳遞至容器的命令。

```
$ cat <<EOF > ./submit-job-override.json
```

```
{
  "jobName": "EksWithOverrides",
  "jobQueue": "My-Eks-JQ1",
  "jobDefinition": "MyJobOnEks_Sleep",
  "eksPropertiesOverride": {
    "podProperties": {
      "containers": [
        {
          "command": [
            "/bin/sh"
          ],
          "args": [
            "-c",
            "echo hello world"
          ]
        }
      ]
    }
  }
}
EOF
$ aws batch submit-job --cli-input-json file://./submit-job-override.json
```

備註

- AWS Batch 在工作完成後積極清除網繭，以減少負載。Kubernetes若要檢查工作的詳細資訊，必須設定記錄。如需詳細資訊，請參閱 使用 CloudWatch 日誌監控 AWS Batch Amazon EKS 任務。
- 若要改善操作詳細資訊的可見性，請啟用 Amazon EKS 控制平面記錄。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的 Amazon EKS 控制平面記錄。
- Daemonsets和kubelets額外負荷會影響可用的 vCPU 和記憶體資源，特別是擴展和工作放置。如需更多詳細資訊，請參閱 Amazon EKS AWS Batch 上的記憶體和 vCPU 考量。

開始使用 AWS Batch Amazon EKS 私有集群

AWS Batch 這是一項受管服務，可協調 Amazon Elastic Kubernetes Service (Amazon EKS) 叢集中的批次工作負載。這包括佇列、相依性追蹤、受管理的工作重試和優先順序、網繭管理和節點調整。此功能可將現有的私有 Amazon EKS 叢集與連接起 AWS Batch 來，以大規模執行任務。您可以使用 `eksctl`(Amazon EKS 的命令列界面)、 AWS 主控台或建立包含所有其他必要資源的私有 Amazon EKS 叢集。AWS Command Line Interface對於私有 Amazon EKS 叢集的 Support 通 AWS Batch 常可在商業用途中使用 (如果 AWS 區域 有提 AWS Batch供)。

Amazon EKS 私有叢集沒有入站/輸出網際網路存取，而且只有私有子網路。Amazon VPC 端點用於啟用對其他 AWS 服務的私有存取。 `eksctl`支援使用預先存在的 Amazon VPC 和子網路建立完全私有的叢集。 `eksctl`也會在提供的 Amazon VPC 中建立 Amazon VPC 端點，並修改所提供子網路的路由表。

每個子網路都應該有一個與其相關聯的明確路由表，因為`eksctl`不會修改主路由表。您的叢集必須從 Amazon VPC 中的容器登錄中提取映像。此外，您也可以在 Amazon VPC 中建立 Amazon 彈性容器登錄，並將容器映像複製到其中以供節點提取。如需詳細資訊，請參閱將容器映像從一個存放庫複製到另一個存放庫。若要開始使用 Amazon ECR 私有儲存庫，請參閱 Amazon ECR 私有儲存庫。

您可以選擇性地使用 Amazon ECR 建立直通快取規則。為外部公用登錄建立提取快取規則之後，您可以使用 Amazon ECR 私有登錄 uriform 資源表想符 (URI) 從該外部公用登錄提取映像。然後，Amazon ECR 會建立一個儲存庫並快取映像檔。使用 Amazon ECR 私有登錄 URI 提取快取映像時，Amazon ECR 會檢查遠端登錄以查看是否有新版本的映像，並每 24 小時更新您的私有登錄一次。

內容

必要條件

在開始本教學課程之前，您必須安裝並設定建立和管理和 Amazon EKS 資源所需的下列工具 AWS Batch 和資源。您還需要建立所有必要的資源，包括 VPC、子網路、路由表、VPC 端點和 Amazon EKS 叢集。您需要使用 AWS CLI.

- AWS CLI— 用於處理 AWS 服務的命令行工具，包括 Amazon EKS。本指南要求您使用 2.8.6 或更新版本或 1.26.0 或更新版本。若要取得更多資訊，請參閱《AWS Command Line Interface 使用指南》 AWS CLI中的〈安裝、更新和解除安裝〉。

  安裝之後 AWS CLI，我們建議您對其進行配置。若要取得更多資訊，請參閱《使用指南》`aws configure`中的AWS Command Line Interface 〈快速配置〉。

- **kubectl**— 使用Kubernetes叢集的命令列工具。本指南要求您使用版本 1.23 或更新版本。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的安裝或更新 kubectl。
- **eksctl**— 可與 Amazon EKS 叢集搭配使用的命令列工具，可自動執行許多個別任務。本指南要求您使用版本 0.115.0 或更新版本。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的安裝或更新 eksctl。
- 必要 AWS Identity and Access Management (IAM) 許可 — 您使用的 IAM 安全主體必須具有使用 Amazon EKS IAM 角色和服務連結角色的許可 AWS CloudFormation，以及 VPC 和相關資源。如需詳細資訊，請參閱 IAM 使用者指南中的適用於 Amazon Elastic Kubernetes Service 的動作、資源和條件金鑰和使用服務連結角色。您必須以同一位使用者的身分完成本指南中的所有步驟。
- 建立 Amazon EKS 叢集 — 如需詳細資訊，請參閱 Amazon EKS 使用者指南eksctl中的開始使用 Amazon EKS。

> **Note**
>
> AWS Batch 不會為 CoreDNS 或其他部署網繭提供受管節點協調流程。如果您需要 CoredN，請參閱 Amazon EKS 使用者指南中的新增 CoreDNS Amazon EKS 附加元件。或者，用eksctl create cluster create來建立叢集，依預設會包含 CoreDNS。

- 權限 — 呼叫 CreateComputeEnvironmentAPI 操作以建立使用 Amazon EKS 資源之運算環境的使用者需要 eks:DescribeCluster API 作業的許可。使用 AWS Management Console 使用 Amazon EKS 資源建立運算資源需要eks:DescribeCluster和eks:ListClusters的許可。
- 使用範例設定檔，在 us-east-1 區域中建立私有 EKS 叢集。eksctl

```
kind: ClusterConfig
apiVersion: eksctl.io/v1alpha5
availabilityZones:
    - us-east-1a
    - us-east-1b
    - us-east-1d
managedNodeGroups:
    privateNetworking: true
privateCluster:
    enabled: true
    skipEndpointCreation: false
```

使用以下命令建立資源：eksctl create cluster -f clusterConfig.yaml

- Batch 管理的節點必須部署到具有您所需 VPC 介面端點的子網路。如需詳細資訊，請參閱私人叢集需求。

步驟 1：準備您的 EKS 叢集 AWS Batch

所有步驟都是必需的。

1. 為 AWS Batch 工作建立專用的命名空間

   用kubectl於建立新的命名空間。

```
$ namespace=my-aws-batch-namespace
$ cat - <<EOF | kubectl create -f -
{
  "apiVersion": "v1",
  "kind": "Namespace",
  "metadata": {
    "name": "${namespace}",
    "labels": {
      "name": "${namespace}"
    }
  }
}
EOF
```

   輸出：

```
namespace/my-aws-batch-namespace created
```

2. 透過角色型存取控制 (RBAC) 啟用存取

   用kubectl於為叢集建立Kubernetes角色，以 AWS Batch 允許觀看節點和網繭，以及繫結角色。您必須針對每個 Amazon EKS 叢集執行一次此動作。

> **Note**
>
> 如需有關使用 RBAC 授權的詳細資訊，請參閱文件中的使用 RBAC 授權。Kubernetes

```
$ cat - <<EOF | kubectl apply -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: aws-batch-cluster-role
```

```
rules:
  - apiGroups: [""]
    resources: ["namespaces"]
    verbs: ["get"]
  - apiGroups: [""]
    resources: ["nodes"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["configmaps"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["apps"]
    resources: ["daemonsets", "deployments", "statefulsets", "replicasets"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["clusterroles", "clusterrolebindings"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: aws-batch-cluster-role-binding
subjects:
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: aws-batch-cluster-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

輸出：

```
clusterrole.rbac.authorization.k8s.io/aws-batch-cluster-role created
clusterrolebinding.rbac.authorization.k8s.io/aws-batch-cluster-role-binding created
```

建立命名空間範圍的Kubernetes角色， AWS Batch 以管理和生命週期網繭並繫結它。您必須為每個唯一命名空間執行一次此操作。

```
$ namespace=my-aws-batch-namespace
$ cat - <<EOF | kubectl apply -f - --namespace "${namespace}"
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: aws-batch-compute-environment-role
  namespace: ${namespace}
rules:
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["create", "get", "list", "watch", "delete", "patch"]
  - apiGroups: [""]
    resources: ["serviceaccounts"]
    verbs: ["get", "list"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["roles", "rolebindings"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: aws-batch-compute-environment-role-binding
  namespace: ${namespace}
subjects:
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: aws-batch-compute-environment-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

輸出：

```
role.rbac.authorization.k8s.io/aws-batch-compute-environment-role created
rolebinding.rbac.authorization.k8s.io/aws-batch-compute-environment-role-binding
 created
```

更新設Kubernetesaws-auth定對應，以將先前的 RBAC 權限對應至服務連結角色。 AWS Batch

```
$ eksctl create iamidentitymapping \
    --cluster my-cluster-name \
    --arn "arn:aws:iam::<your-account>:role/AWSServiceRoleForBatch" \
    --username aws-batch
```

輸出：

```
2022-10-25 20:19:57 [#]  adding identity "arn:aws:iam::<your-account>:role/
AWSServiceRoleForBatch" to auth ConfigMap
```

> **Note**
>
> 路徑aws-service-role/batch.amazonaws.com/已從服務連結角色的 ARN 中移除。這是因為aws-auth配置對映有問題。如需詳細資訊，請參閱中的路徑包含在其 ARN 中時，具有路徑的角色無法運作aws-authconfigmap。

步驟 2：建立 Amazon EKS 運算環境

AWS Batch 運算環境會定義運算資源參數，以符合批次工作負載的需求。在受管運算環境中，可 AWS Batch 協助您管理 Amazon EKS 叢集內運算資源 (Kubernetes節點) 的容量和執行個體類型。這是根據您在建立運算環境時定義的計算資源規格而定。您可以使用 EC2 隨需執行個體或 EC2 競價型執行個體。

現在AWSServiceRoleForBatch服務連結角色可以存取 Amazon EKS 叢集，您可以建立 AWS Batch 資源。首先，建立一個指向 Amazon EKS 叢集的運算環境。

```
$ cat <<EOF > ./batch-eks-compute-environment.json
{
  "computeEnvironmentName": "My-Eks-CE1",
  "type": "MANAGED",
  "state": "ENABLED",
  "eksConfiguration": {
    "eksClusterArn": "arn:aws:eks:<region>:123456789012:cluster/<cluster-name>",
    "kubernetesNamespace": "my-aws-batch-namespace"
  },
  "computeResources": {
    "type": "EC2",
    "allocationStrategy": "BEST_FIT_PROGRESSIVE",
```

```
    "minvCpus": 0,
    "maxvCpus": 128,
    "instanceTypes": [
        "m5"
    ],
    "subnets": [
        "<eks-cluster-subnets-with-access-to-the-image-for-image-pull>"
    ],
    "securityGroupIds": [
        "<eks-cluster-sg>"
    ],
    "instanceRole": "<eks-instance-profile>"
  }
}
EOF
$ aws batch create-compute-environment --cli-input-json file://./batch-eks-compute-environment.json
```

備註

- 不應指定`serviceRole`參數，則會使用 AWS Batch 服務連結角色。 AWS Batch 在 Amazon EKS 上僅支援 AWS Batch 服務連結角色。
- Amazon EKS 運算環境僅`BEST_FIT_PROGRESSIVE`支援`SPOT_CAPACITY_OPTIMIZED`、和`SPOT_PRICE_CAPACITY_OPTIMIZED`配置策略。

  > **Note**
  >
  > 我們建議您在大多數情況下使用`SPOT_PRICE_CAPACITY_OPTIMIZED`而不是 `SPOT_CAPACITY_OPTIMIZED` n。

- 有關詳情`instanceRole`，請參閱 Amazon EKS 使用者指南中的建立 Amazon EKS 節點 IAM 角色和啟用 IAM 主體存取您的叢集。如果您使用的是網繭聯網，請參閱 Amazon EKS 使用者指南中的設定 Amazon VPC CNI 外掛程式以Kubernetes將 IAM 角色用於服務帳戶。
- 取得`subnets`參數工作子網路的一種方法是使用 Amazon EKS 受管節點群組建立的公有子網路，這些公用子網路是在建立 Amazon EKS 叢集`eksctl`時所建立的。否則，請使用具有支援提取影像之網路路徑的子網路。
- 該`securityGroupIds`參數可以使用與 Amazon EKS 叢集相同的安全群組。此命令會擷取叢集的安全性群組 ID。

```
$ eks describe-cluster \
```

```
    --name <cluster-name> \
    --query cluster.resourcesVpcConfig.clusterSecurityGroupId
```

- 維護 Amazon EKS 運算環境是共同的責任。如需詳細資訊，請參閱 Amazon EKS 中的安全性。

> **Important**
>
> 在繼續之前，請務必確認運算環境狀況良好。DescribeComputeEnvironmentsAPI 操作可用於執行此操作。
>
> ```
> $ aws batch describe-compute-environments --compute-environments My-Eks-CE1
> ```
>
> 確認參`status`數不是`INVALID`。如果是，請查看原因的`statusReason`參數。如需詳細資訊，請參閱 疑難排 AWS Batch。

步驟 3：建立工作佇列並連接運算環境

```
$ aws batch describe-compute-environments --compute-environments My-Eks-CE1
```

提交至此新任務佇列的任務會在加入與運算環境相關聯之 Amazon EKS 叢集的 AWS Batch 受管節點上做為網繭執行。

```
$ cat <<EOF > ./batch-eks-job-queue.json
 {
    "jobQueueName": "My-Eks-JQ1",
    "priority": 10,
    "computeEnvironmentOrder": [
      {
        "order": 1,
        "computeEnvironment": "My-Eks-CE1"
      }
    ]
  }
EOF
$ aws batch create-job-queue --cli-input-json file://./batch-eks-job-queue.json
```

步驟 4：建立工作定義

在工作定義的影像欄位中，不要提供公用 ECR 存放庫中影像的連結，而是提供存放在我們私人 ECR 存放庫中的映像連結。請參閱下列範例工作定義：

```
$ cat <<EOF > ./batch-eks-job-definition.json
{
  "jobDefinitionName": "MyJobOnEks_Sleep",
  "type": "container",
  "eksProperties": {
    "podProperties": {
      "hostNetwork": true,
      "containers": [
        {
          "image": "account-id.dkr.ecr.region.amazonaws.com/amazonlinux:2",
          "command": [
            "sleep",
            "60"
          ],
          "resources": {
            "limits": {
              "cpu": "1",
              "memory": "1024Mi"
            }
          }
        }
      ],
      "metadata": {
        "labels": {
          "environment": "test"
        }
      }
    }
  }
}
EOF
$ aws batch register-job-definition --cli-input-json file://./batch-eks-job-
definition.json
```

若要執行 kubectl 命令，您需要對 Amazon EKS 叢集進行私有存取。這表示叢集 API 伺服器的所有流量都必須來自叢集的 VPC 或連線網路內。

步驟 5：提交工作

```
$ aws batch submit-job - -job-queue My-Eks-JQ1 \
    - -job-definition MyJobOnEks_Sleep - -job-name My-Eks-Job1
$ aws batch describe-jobs - -job <jobId-from-submit-response>
```

備註

- 僅支援單一容器工作。
- 請確定您熟悉cpu和memory參數的所有相關考量。如需詳細資訊，請參閱 Amazon EKS AWS Batch 上的記憶體和 vCPU 考量。
- 如需在 Amazon EKS 資源上執行任務的詳細資訊，請參閱Amazon EKS 工作機會。

(選擇性) 提交含覆寫項目的工作

此工作會覆寫傳遞至容器的命令。

```
$ cat <<EOF > ./submit-job-override.json
{
  "jobName": "EksWithOverrides",
  "jobQueue": "My-Eks-JQ1",
  "jobDefinition": "MyJobOnEks_Sleep",
  "eksPropertiesOverride": {
    "podProperties": {
      "containers": [
        {
          "command": [
            "/bin/sh"
          ],
          "args": [
            "-c",
            "echo hello world"
          ]
        }
      ]
    }
  }
}
EOF
$ aws batch submit-job - -cli-input-json file://./submit-job-override.json
```

備註

- AWS Batch 在工作完成後積極清除網繭，以減少負載。Kubernetes若要檢查工作的詳細資訊，必須設定記錄。如需詳細資訊，請參閱 使用 CloudWatch 日誌監控 AWS Batch Amazon EKS 任務。
- 若要改善操作詳細資訊的可見性，請啟用 Amazon EKS 控制平面記錄。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的 Amazon EKS 控制平面記錄。
- Daemonsets和kubelets額外負荷會影響可用的 vCPU 和記憶體資源，特別是擴展和工作放置。如需詳細資訊，請參閱 Amazon EKS AWS Batch 上的記憶體和 vCPU 考量。

故障診斷

如果由啟動的節點 AWS Batch 無法存取儲存映像的 Amazon ECR 儲存庫 (或任何其他存放庫)，則您的任務可能會維持在 STARTING 狀態。這是因為網繭將無法下載映像並執行您的 AWS Batch 工作。如果您按一下由啟動的網繭名稱， AWS Batch 您應該可以看到錯誤訊息並確認問題。錯誤訊息看起來應該類似下列內容：

```
Failed to pull image "public.ecr.aws/amazonlinux/amazonlinux:2": rpc error: code =
Unknown desc = failed to pull and unpack image
"public.ecr.aws/amazonlinux/amazonlinux:2": failed to resolve reference
"public.ecr.aws/amazonlinux/amazonlinux:2": failed to do request: Head
"https://public.ecr.aws/v2/amazonlinux/amazonlinux/manifests/2": dial tcp: i/o timeout
```

如需其他常見的疑難排解案例，請參閱疑 AWS Batch難 如需根據網繭狀態進行疑難排解，請參閱如何疑難排解 Amazon EKS 中的網繭狀態？ 。

任務

工作是由開始的工作單位 AWS Batch。您可以將任務叫用為在 ECS 叢集中 Amazon ECS 容器執行個體上執行的容器化應用程式。

容器化的任務可以參考容器映像、命令和參數。如需詳細資訊，請參閱 Job 定義參數 ContainerProperties。

您可以提交大量獨立、簡單的任務

主題

提交工作

註冊工作定義後，您可以將其作為工作提交至 AWS Batch 工作佇列。您可以覆寫在執行階段工作定義中指定的許多參數。

提交任務

1. 請在以下位置開啟 AWS Batch 主控台。 https://console.aws.amazon.com/batch/

2. 從導覽列中，選取 AWS 區域 要使用的。
3. 在導覽窗格中，選擇 Jobs (任務)。
4. 選擇 [送出新工作]。
5. 在名稱中，輸入工作定義的唯一名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
6. 對於「Job 定義」，請為您的工作選擇現有的工作定義。如需詳細資訊，請參閱 建立單一節點工作定義 。
7. 對於「Job 佇列」，請選擇現有的工作佇列。如需詳細資訊，請參閱 建立工作佇列。
8. 對於 Job 相依性，請選擇新增 Job 相依性。
   - 針對 Job ID，輸入任何相依性的工作 ID。然後選擇新增工作相依性。一個工作最多可以有 20 個相依性。如需詳細資訊，請參閱 Job 相依性。
9. (僅適用於陣列任務) 在 Array size (陣列大小) 中，指定 2 至 10,000 之間的陣列大小。
10. (選擇性) 展開標籤，然後選擇 [新增標籤]，將標籤新增至資源。輸入機碼和選用值，然後選擇「新增標記」。
11. 選擇 [下一頁]。
12. 在 「Job 覆寫」區段中：
    a. (選擇性) 針對「排程優先順序」，輸入介於 0 到 100 之間的排程優先順序值。較高的值被賦予更高的優先級。
    b. (選擇性) 對於 Job 嘗試，請輸入 AWS Batch 嘗試將工作移至某個RUNNABLE狀態的次數上限。您可以輸入 1 到 10 之間的數字。如需詳細資訊，請參閱 自動化工作重試。
    c. (選擇性) 對於執行逾時，輸入逾時值 (以秒為單位)。執行逾時是未完成工作終止前的時間長度。如果嘗試超過逾時持續時間，則會停止並移至FAILED狀態。如需詳細資訊，請參閱 Job 逾時。最小值為 60 秒。

       > **Important**
       > 不要依賴在 Fargate 資源上執行的作業執行超過 14 天。14 天之後，Fargate 資源可能不再可用，因為工作可能被終止。

    d. (選擇性) 開啟傳播標籤以將標籤從任務和任務定義傳播到 Amazon ECS 任務。
13. 展開 Additional configuration (其他組態)。

14. (選擇性) 對於「重試策略條件」，請選擇「結束時新增評估」。輸入至少一個參數值，然後選擇「作業」。對於每組條件，必須將「動作」設定為「重試」或「結束」。這些動作意味著以下內容：

    - 重試 — AWS Batch 重試，直到達到您指定的作業嘗試次數為止。
    - 結束 — AWS Batch 停止重試工作。

    > **Important**
    >
    > 如果您選擇 [結束時新增評估]，請至少設定一個參數，然後選擇 [動作] 或選擇 [結束時移除評估]。

15. 對於「參數」，選擇「新增參數」以加入參數替代預留位置。然後，輸入一個鍵和一個可選的值。
16. 在容器覆寫區段中：

    a. 在 Command (命令) 中，指定要傳送至容器的命令。對於簡單指令，請像輸入指令提示一樣輸入指令。對於更複雜的命令，例如使用特殊字符），請使用 JSON 語法。

    > **Note**
    >
    > 此參數不能包含空字串。

    b. 對於 vCPUs，請輸入要為容器保留的 vCPUs 數目。此參數會映射到 Docker Remote API 的建立容器區段中的 `CpuShares` 以及 docker run 的 `--cpu-shares` 選項。每個 vCPU 相當於 1,024 個 CPU 共用。您必須指定至少 1 個 vCPU。

    c. 在記憶體中，輸入容器可用的記憶體限制。如果您的容器嘗試超過此處指定的記憶體，則會停止容器。此參數會映射到 Docker Remote API 的建立容器區段中的 `Memory` 以及 docker run 的 `--memory` 選項。您必須為單一工作指定至少 4 MiB 的記憶體。

    > **Note**
    >
    > 為了最大限度地提高資源使用率，請為特定執行個體類型的作業設定記憶體 如需詳細資訊，請參閱 運算資源記憶體管理。

    d. (選擇性) 針對 GPU 數目，選擇要為容器保留的 GPU 數目。

    e. (選擇性) 對於環境變數，請選擇新增環境變數，將環境變數新增為名稱-值配對。這些變量被傳遞到容器。

f. 選擇 [下一頁]。

g. 若要檢閱 Job，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成後，選擇 [建立工作定義]。

任務狀態

當您將工作提交至工 AWS Batch 作佇列時，工作會進入SUBMITTED狀態。任務將經過以下狀態，直到其失敗 (以 0 代碼結束) 或失敗 (以與非零代碼結束) 為止。 AWS Batch 任務可能有以下狀態：

SUBMITTED

已提交至佇列且尚未由排程器評估的工作。排程器評估任務，判斷其是否對任何其他任務的成功完成存有任何未完成的相依性。如果有相依性，任務將移至 PENDING。如果沒有相依性，任務將移至 RUNNABLE。

PENDING

位於佇列中且由於其他工作或資源的相依性而無法執行的工作。如果相依性獲得滿足，任務將移至 RUNNABLE。

RUNNABLE

佇列中的某一任務沒有未完成的相依性，因此已準備好排程傳送到主機。在對應至工作佇列的其中一個計算環境中，只要有足夠的資源可用，就會立即啟動處於此狀態的工作。不過，假如一直無法取得足夠的資源，任務將無限期停留在此狀態。

> **Note**
>
> 如果您的工作沒有進展到STARTING，請參閱工作停留在某個RUNNABLE狀態疑難排解一節。

STARTING

這些任務已排程傳送到主機，且相關的容器初始化作業正在進行中。取出容器映像且容器設置完畢並開始執行後，該任務將轉換為 RUNNING。

影像提取持續時間、Amazon EKS InitContainer 完成持續時間，以及 Amazon ECS 容器相依性解決持續時間會在啟動狀態中發生。擷取工作影像所需的時間，等同於工作處於 START 狀態的超過時間。

例如，如果擷取工作的影像需要三分鐘時間，您的工作將處於 START 狀態三分鐘。如果 InitContainer 總共需要十分鐘才能完成，那麼您的 Amazon EKS 任務將開始十分鐘。如果您的 Amazon ECS 任務中有 Amazon ECS 容器相依性集，則任務將會在開始狀態，直到解決所有容器相依性 (其執行階段) 為止。啟動不包含在逾時中；持續時間從 RUNNING 開始。如需詳細資訊，請參閱 Job 狀態。

`RUNNING`

任務在運算環境中的 Amazon ECS 容器執行個體上以容器任務的形式執行。任務的容器結束時，處理結束代碼將判斷任務為成功或失敗。`0` 結束代碼表示成功，任何非零的結束代碼則表示失敗。如果與嘗試失敗有關的任務在其選用的重試策略組態中有任何剩下的嘗試，任務將再次移至 `RUNNABLE`。如需詳細資訊，請參閱 自動化工作重試。

> **Note**
>
> `RUNNING`工作記錄可在 CloudWatch 記錄檔中找到。記錄群組為`/aws/batch/job`，記錄資料流名稱格式如下所示：*`first200CharsOfJobDefinitionName`*`/default/`*`ecs_task_id`*此格式 future 可能會改變。
> 工作到達`RUNNING`狀態後，您可以透過 DescribeJobsAPI 作業以程式設計方式擷取其記錄資料流名稱。如需詳細資訊，請參閱 Amazon CloudWatch 日誌使用者指南中的檢視傳送至 CloudWatch 日誌的日誌資料。依預設，這些記錄永遠不會過期。不過，您可以修改保留期間。如需詳細資訊，請參閱 Amazon CloudWatch 日誌使用者指南中的變更日誌中的 CloudWatch 日誌資料保留。

`SUCCEEDED`

已成功完成任務，結束代碼為 `0`。工作的`SUCCEEDED`工作狀態至少會保留 7 天。 AWS Batch

> **Note**
>
> `SUCCEEDED`工作記錄可在 CloudWatch 記錄檔中找到。記錄群組為`/aws/batch/job`，記錄資料流名稱格式如下所示：*`first200CharsOfJobDefinitionName`*`/default/`*`ecs_task_id`*此格式 future 可能會改變。
> 工作到達`RUNNING`狀態後，您可以透過 DescribeJobsAPI 作業以程式設計方式擷取其記錄資料流名稱。如需詳細資訊，請參閱 Amazon CloudWatch 日誌使用者指南中的檢視傳送至 CloudWatch 日誌的日誌資料。依預設，這些記錄永遠不會過期。不過，您可以修改保

留期間。如需詳細資訊，請參閱 Amazon CloudWatch 日誌使用者指南中的變更日誌中的 CloudWatch 日誌資料保留。

`FAILED`

任務所有的可用嘗試都失敗。工作的`FAILED`工作狀態至少會保留 7 天。 AWS Batch

> **Note**
>
> `FAILED`工作記錄可在 CloudWatch 記錄檔中找到。記錄群組為`/aws/batch/job`，記錄資料流名稱格式如下所示：*`first200CharsOfJobDefinitionName`*`/default/`*`ecs_task_id`*此格式 future 可能會改變。
> 工作到達`RUNNING`狀態後，您可以透過 DescribeJobsAPI 作業以程式設計方式擷取其記錄資料流。如需詳細資訊，請參閱 Amazon CloudWatch 日誌使用者指南中的檢視傳送至 CloudWatch 日誌的日誌資料。依預設，這些記錄永遠不會過期。不過，您可以修改保留期間。如需詳細資訊，請參閱 Amazon CloudWatch 日誌使用者指南中的變更日誌中的 CloudWatch 日誌資料保留。

AWS Batch 工作環境變數

AWS Batch 在容器作業中設定特定的環境變數。這些環境變數會針對工作內的容器提供內部檢查。您可以在應用程式的邏輯中使用這些變數的值。 AWS Batch 設置的所有變量都以`AWS_BATCH_`前綴開頭。這是一個受保護的環境變量前綴。您不能在工作定義或覆寫中將此前置詞用於您自己的變數。

以下環境變數適用於任務容器：

`AWS_BATCH_CE_NAME`

此變數會設定為放置工作的計算環境名稱。

`AWS_BATCH_JOB_ARRAY_INDEX`

只會在子陣列任務中設定此變數。陣列任務索引從 0 開始，而且每個子任務會收到一個唯一的索引號碼。例如，含 10 個子系的陣列任務有 0-9 的索引值。您可以使用此索引值，控制您陣列任務子系的區分方式。如需詳細資訊，請參閱 教學課程：使用陣列工作索引來控制工作差異化。

`AWS_BATCH_JOB_ARRAY_SIZE`

此變數會設定為父陣列工作的大小。父陣列工作的大小會傳遞至此變數中的子陣列工作。

`AWS_BATCH_JOB_ATTEMPT`

會將此變數設為任務嘗試號碼。第一次嘗試的編號為 1。如需詳細資訊，請參閱 自動化工作重試。

`AWS_BATCH_JOB_ID`

此變數設定為工 AWS Batch 作 ID。

`AWS_BATCH_JOB_KUBERNETES_NODE_UID`

此變數會設定為執行網繭之 Kubernetes 叢集中之節點物件的 Kubernetes UID。此變數僅適用於在 Amazon EKS 資源上執行的任務。如需詳細資訊，請參閱Kubernetes文件中的 UID。

`AWS_BATCH_JOB_MAIN_NODE_INDEX`

只會在多節點平行任務中設定此變數。會將此變數設為任務主要節點的索引數量。您的應用程式程式碼可以`AWS_BATCH_JOB_MAIN_NODE_INDEX`將與個別節點`AWS_BATCH_JOB_NODE_INDEX`上的項目進行比較，以判斷它是否為主節點。

`AWS_BATCH_JOB_MAIN_NODE_PRIVATE_IPV4_ADDRESS`

此變數僅在多節點 parallel 作業子節點中設定。此變數不存在於主節點上，但會設定為工作主節點的私有 IPv4 位址。您的子節點應用程式程式碼可以使用此地址與主節點通訊。

`AWS_BATCH_JOB_NODE_INDEX`

只會在多節點平行任務中設定此變數。會將此變數設為節點的節點索引數量。節點索引從 0 開始，而且每個節點皆會收到一個唯一的索引號碼。例如，含 10 個子系的多節點平行任務具有 0-9 的索引值。

`AWS_BATCH_JOB_NUM_NODES`

只會在多節點平行任務中設定此變數。此變數會設定為您為多節點 parallel 工作要求的節點數目。

`AWS_BATCH_JQ_NAME`

會將此變數設為您所提交任務的任務佇列名稱。

自動化工作重試

您可將重試策略套用至任務和任務定義，讓失敗的任務自動重試。可能的失敗情況包括：

- 容器任務有任何的非零結束代碼
- Amazon EC2 實例故障或終止
- 內部 AWS 服務錯誤或中斷

將工作提交至工作佇列並置於被視為嘗試的RUNNING狀態時。根據預設，每個任務會嘗試一次移至 SUCCEEDED 或 FAILED 任務狀態。但是，工作定義和工作提交工作流程都可用於指定嘗試 1 到 10 次之間的重試策略。如果指OnExit定了評估，它最多可以包含 5 個重試策略。如果指OnExit定了evaluate，但沒有任何重試策略相符，則會重試工作。對於不符合要結束的工作，請新增因任何原因而結束的最終項目。例如，此evaluateOnExit物件有兩個項目，其中包含動作RETRY，最後一個動作為的項目EXIT。

```
"evaluateOnExit": [
    {
        "action": "RETRY",
        "onReason": "AGENT"
    },
    {
        "action": "RETRY",
        "onStatusReason": "Task failed to start"
    },
    {
        "action": "EXIT",
        "onReason": "*"
    }
]
```

在執行時間，AWS_BATCH_JOB_ATTEMPT 環境變數設為容器的對應任務嘗試次數。第一次嘗試會編號1，後續嘗試會以遞增順序排列 (例如，2、3、4)。

例如，假設工作嘗試因任何原因而失敗，且在重試組態中指定的嘗試次數大於AWS_BATCH_JOB_ATTEMPT數目。然後，將工作放回狀RUNNABLE態。如需詳細資訊，請參閱 任務狀態。

> **Note**
>
> 取消或終止的工作不會重試。此外，因為無效的工作定義而失敗的工作也不會重試。

如需詳細資訊，請參閱重試策略建立單一節點工作定義 、提交工作和已停止工作錯誤碼。

Job 相依性

當您提交 AWS Batch 工作時，您可以指定工作相依的工作 ID。執行此動作時， AWS Batch 排程器將確保任務只在指定的相依性成功完成後執行。成功完成後，相依的任務將從 PENDING 轉為

RUNNABLE，然後轉為 STARTING 和 RUNNING。如果任何任務相依性失敗，相依的任務將自動從 PENDING 轉為 FAILED。

例如，A 任務可以對另外最多 20 個任務有相依性，必須等這 20 個任務成功後才能執行。接著您可以提交額外的任務，對 A 任務和最多 19 個其他的任務有相依性。

對於陣列任務，您可以指定 SEQUENTIAL 類型相依性，且不指定任務 ID，讓每個子陣列任務從索引 0 開始依序完成。您也可以使用任務 ID 指定 N_TO_N 類型相依性。如此一來，此任務的每個索引子系必須等待各相依性對應的索引子系完成後，才能開始。如需詳細資訊，請參閱 陣列工作。

若要提交具有相依性的 AWS Batch 工作，請參閱提交工作。

Job 逾時

您可以設定任務的逾時時間，如此一來，假如任務執行超過該時間， AWS Batch 便會終止該任務。例如，您可能有一個您知道應該只需要 15 分鐘完成的任務。有時您的應用程式會一直卡在迴圈和執行中，因此您可以設定逾時為 30 分鐘以終止卡住的任務。

> **Important**
>
> 依預設， AWS Batch 沒有工作逾時。如果您未定義工作逾時，工作會一直執行，直到容器結束為止。

您在任務定義內或是當您提交此任務時指定 attemptDurationSeconds 參數，該參數必須至少有 60 秒。在工作嘗試的時間startedAt戳記之後超過此秒數時，便 AWS Batch 會終止工作。在運算資源時，您的任務容器會收到 SIGTERM 訊號，讓您的應用程式有機會正常關閉。如果容器在 30 秒後仍在執行中，則會傳送 SIGKILL 訊號以強制關閉容器。

逾時終止會依最佳作法來處理。您不應該期望在工作嘗試超時時完全發生您的超時終止（可能需要幾秒鐘的時間）。如果您的應用程式需要精確執行逾時，您應在應用程式內實作此邏輯。如果您有大量任務同時逾時，逾時終止將採用前進先出佇列，按批次終止任務。

> **Note**
>
> AWS Batch 工作沒有逾時值上限。

如果工作因超過逾時持續時間而終止，則不會重試。如果任務嘗試自行失敗，任務會在啟用重試下進行重試，且進行新嘗試時將重新開始逾時倒數。

> **Important**
>
> 在 Fargate 資源上執行的作業無法預期執行超過 14 天。如果逾時持續時間超過 14 天，則 Fargate 資源可能不再可用，並且工作將終止。

對於陣列任務，子任務的逾時設定與父任務相同。

如需使用逾時組態提交 AWS Batch 工作的相關資訊，請參閱提交工作。

Amazon EKS 工作機會

工作是中最小的工作單位 AWS Batch。Amazon EKS 上的 AWS Batch 任務具有 one-to-one 對應至 Kubernetes網繭的對應。 AWS Batch 工作定義是工 AWS Batch 作的範本。當您送出 AWS Batch 工作時，您會參照工作定義、指定工作佇列的目標，以及提供工作的名稱。在 Amazon EKS 上任務的 AWS Batch 任務定義中，E ksProperties 參數定義了 Amazon EKS 任務 AWS Batch 上支持的一組參數。在SubmitJob要求中，eks PropertiesOverride 參數允許覆寫某些通用參數。如此一來，您就可以使用多個工作的工作定義範本。將任務分派到 Amazon EKS 叢集時，請將任務 AWS Batch 轉換為 `podspec` (`Kind: Pod`)。會`podspec`使用一些額外的 AWS Batch 參數來確保工作已正確調整比例和排程。 AWS Batch 結合標籤和污點，以確保作業僅在 AWS Batch 受管節點上執行，而且其他網繭不會在這些節點上執行。

> **Important**
>
> - 如果未在 Amazon EKS 任務定義中明確設定`hostNetwork`參數，則處於主機模式的網繭聯網模式 AWS Batch 預設為主機模式。更具體地說，會套用下列設定：`hostNetwork=true`和`dnsPolicy=ClusterFirstWithHostNet`。
> - AWS Batch 在網繭完成其工作後立即清除工作網繭。若要查看網繭應用程式記錄，請為您的叢集設定記錄服務。如需詳細資訊，請參閱 使用 CloudWatch 日誌監控 AWS Batch Amazon EKS 任務。

將執行中的工作對應至網繭和節點

正在執行`podProperties`的工作具有`podName`為目前作業嘗試設定的`nodeName`參數。使用 DescribeJobsAPI 作業來檢視這些參數。

下列為範例輸出。

```
$ aws batch describe-jobs --job 2d044787-c663-4ce6-a6fe-f2baf7e51b04
{
 "jobs": [
  {
   "status": "RUNNING",
   "jobArn": "arn:aws:batch:us-east-1:123456789012:job/2d044787-c663-4ce6-a6fe-
f2baf7e51b04",
   "jobDefinition": "arn:aws:batch:us-east-1:123456789012:job-definition/
MyJobOnEks_SleepWithRequestsOnly:1",
   "jobQueue": "arn:aws:batch:us-east-1:123456789012:job-queue/My-Eks-JQ1",
   "jobId": "2d044787-c663-4ce6-a6fe-f2baf7e51b04",
   "eksProperties": {
    "podProperties": {
     "nodeName": "ip-192-168-55-175.ec2.internal",
     "containers": [
      {
       "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
       "resources": {
        "requests": {
         "cpu": "1",
         "memory": "1024Mi"
        }
       }
      }
     ],
     "podName": "aws-batch.b0aca953-ba8f-3791-83e2-ed13af39428c"
    }
   }
  }
 ]
}
```

對於啟用重試的工作，每個已完成嘗試nodeName的podName和都會在 DescribeJobsAPI 作業的 `eksAttempts` list 參數中。目前執行中嘗試nodeName的podName和位於`podProperties`物件中。

如何將執行中的網繭對應回其工作

網繭具有標籤`jobId`，uuid指出其所屬運算環境的和。 AWS Batch 注入環境變數，以便工作的執行階段可以參考工作資訊。如需詳細資訊，請參閱 AWS Batch 工作環境變數。您可以執行下列命令來檢視此資訊。輸出如下。

```
$ kubectl describe pod aws-batch.14638eb9-d218-372d-ba5c-1c9ab9c7f2a1 -n my-aws-batch-
namespace
Name:         aws-batch.14638eb9-d218-372d-ba5c-1c9ab9c7f2a1
Namespace:    my-aws-batch-namespace
Priority:     0
Node:         ip-192-168-45-88.ec2.internal/192.168.45.88
Start Time:   Wed, 26 Oct 2022 00:30:48 +0000
Labels:       batch.amazonaws.com/compute-environment-uuid=5c19160b-
d450-31c9-8454-86cf5b30548f
              batch.amazonaws.com/job-id=f980f2cf-6309-4c77-a2b2-d83fbba0e9f0
              batch.amazonaws.com/node-uid=a4be5c1d-9881-4524-b967-587789094647
...
Status:       Running
IP:           192.168.45.88
IPs:
  IP:  192.168.45.88
Containers:
  default:
    Image:         public.ecr.aws/amazonlinux/amazonlinux:2
    ...
    Environment:
      AWS_BATCH_JOB_KUBERNETES_NODE_UID:  a4be5c1d-9881-4524-b967-587789094647
      AWS_BATCH_JOB_ID:                   f980f2cf-6309-4c77-a2b2-d83fbba0e9f0
      AWS_BATCH_JQ_NAME:                  My-Eks-JQ1
      AWS_BATCH_JOB_ATTEMPT:              1
      AWS_BATCH_CE_NAME:                  My-Eks-CE1

...
```

AWS Batch Amazon EKS 任務支援的功能

以下是在 Amazon EKS 上執行的任Kubernetes務也常見的 AWS Batch 特定功能：

- Job 相依性
- 陣列工作
- Job 逾時
- 自動化工作重試
- 公平分享排程

Kubernetes**Secrets** 和 **ServiceAccounts**

AWS Batch 支援參考KubernetesSecrets和ServiceAccounts. 您可以將網繭設定為針對服務帳戶使用 Amazon EKS IAM 角色。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的設定網繭以使用Kubernetes服務帳戶。

相關文件

- Amazon EKS AWS Batch 上的記憶體和 vCPU 考量
- 若要在 Amazon EKS 資源上建立以 GPU 為基礎的任務
- 工作停留在某個RUNNABLE狀態

陣列工作

陣列任務為共用常見參數的任務，例如任務定義、vCPU 和記憶體。它會以相關但獨立的基本作業的集合形式執行，這些工作可能會分散在多個主機上，而且可能會同時執行。陣列工作是執行非常 parallel 工作的最有效方式，例如蒙地卡羅模擬、參數化掃描或大型彩現工作。

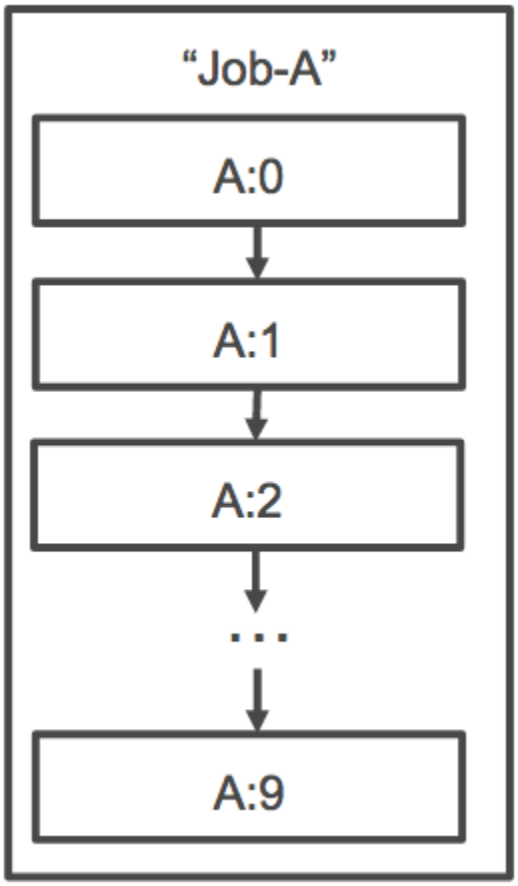
AWS Batch 陣列作業會像一般工作一樣提交。不過，您必須指定陣列大小 (2 至 10,000)，以定義陣列內應該執行的子任務數量。如果您提交陣列大小 1000 的任務，單一任務將執行並產生 1000 個子任務。陣列任務為參考或指標，用於管理所有的子任務。如此一來，您就可以透過單一查詢提交大型工作負載。在attemptDurationSeconds參數中指定的逾時會套用至每個子工作。父陣列工作沒有逾時。

當您提交陣列工作時，父陣列工作會取得一般 AWS Batch 工作 ID。每個子工作都有相同的基本 ID。但是，子工作的陣列索引會附加到父 ID 的末尾，*example_job_ID*:0例如陣列的第一個子工作。

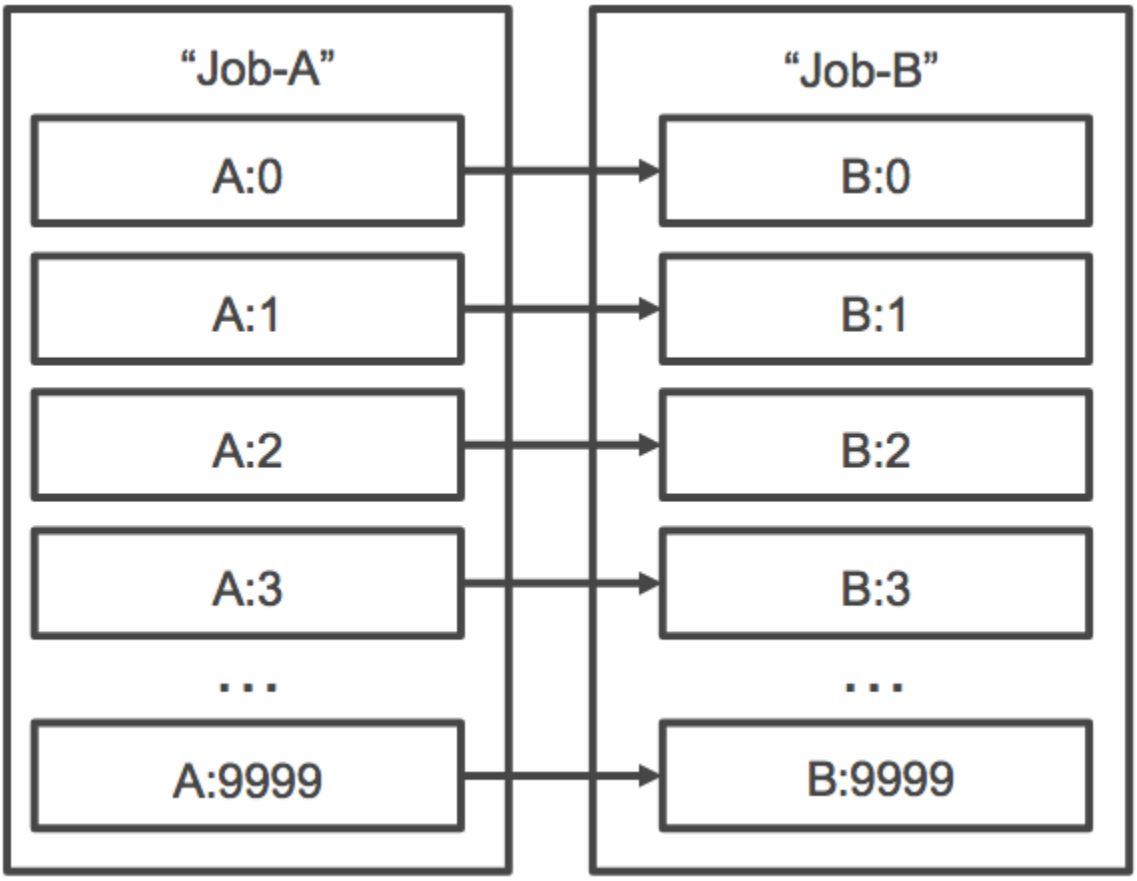
父陣列工作可以輸入SUBMITTEDPENDING、FAILED、或SUCCEEDED狀態。PENDING當任何子工作更新為時，陣列父工作會更新為RUNNABLE。如需工作相依性的詳細資訊，請參閱Job 相依性。

在執行時間，AWS_BATCH_JOB_ARRAY_INDEX 環境變數設為容器的對應任務陣列索引編號。第一個陣列工作索引編號0，後續嘗試會以遞增順序排列 (例如，1、2 和 3)。您可以使用此索引值，控制您陣列任務子系的區分方式。如需詳細資訊，請參閱 教學課程：使用陣列工作索引來控制工作差異化。

對於陣列任務的相依性，您可以指定相依性類型，例如 SEQUENTIAL 或 N_TO_N。您可以指定 SEQUENTIAL 類型相依性 (不指定任務 ID)，讓每個子陣列任務從索引 0 開始依序完成。例如，如果您提交陣列大小 100 的陣列任務，並指定 SEQUENTIAL 類型的相依性，後續將產生 100 個子任務，必須等第一個子任務完成後，下一個子任務才會開始。下圖顯示 A 任務，陣列大小 10 的陣列任務。A 任務子索引的每個任務都相依於前一個子任務。A:1 任務必須等 A:0 任務完成後才會開始。



您也可以指定 N_TO_N 類型相依性，以及陣列任務的任務 ID。如此一來，此任務的每個索引子系必須等待各相依性對應的索引子系完成後，才能開始。下圖顯示 Job A 和 Job B，這兩個陣列工作的陣列大小各為 10,000。B 任務子索引的每個任務相依於 A 任務的對應索引。B:1 任務必須等到 A:1 任務完成後才會開始。



如果您取消或終止父陣列任務，所有子任務將隨之取消或終止。您可以隨時取消或終止個別的子任務 (其將移至 FAILED 狀態)，而不會影響其他子任務。不過，如果子陣列工作失敗 (單獨或手動取消或終止)，父項工作也會失敗。

陣列工作流程範例

AWS Batch 客戶常見的工作流程是執行先決條件設定任務、針對大量輸入任務執行一系列命令，然後以彙總結果並將摘要資料寫入 Amazon S3、DynamoDB、Amazon Redshift 或 Aurora 的任務結束。

例如：

- JobA：一種標準的非陣列任務，可對 Amazon S3 儲存貯體中的物件執行快速列出和中繼資料驗證。BucketASubmitJobJSON 語法如下所示。

```
{
    "jobName": "JobA",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobA-list-and-validate:1"
}
```

- JobB：包含 10,000 個副本的陣列工作，相依於針對中JobA的每個物件執行 CPU 密集型命令，BucketA並將結果上傳至。BucketBSubmitJobJSON 語法如下所示。

```
{
    "jobName": "JobB",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobB-CPU-Intensive-Processing:1",
    "containerOverrides": {
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "4096"
            },
            {
                "type": "VCPU",
                "value": "32"
            }
        ]
   }
    "arrayProperties": {
        "size": 10000
    },
    "dependsOn": [
        {
            "jobId": "JobA_job_ID"
  }
    ]
}
```

- JobC：另外 10,000 個相N_TO_N依性模型的複製陣列工作，會針對中的每個項目執行記憶體密集型命令、將中繼資料寫入 DynamoDBBucketB，然後JobB將產生的輸出上傳至。BucketCSubmitJobJSON 語法如下所示。

```
{
    "jobName": "JobC",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobC-Memory-Intensive-Processing:1",
    "containerOverrides": {
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "32768"
            },
```

```
            {
                "type": "VCPU",
                "value": "1"
            }
        ]
    }
     "arrayProperties": {
         "size": 10000
     },
     "dependsOn": [
         {
             "jobId": "JobB_job_ID",
             "type": "N_TO_N"
         }
     ]
}
```

- JobD：執行 10 個驗證步驟的陣列任務，每個步驟都需要查詢 DynamoDB，並且可能會與上述任何 Amazon S3 儲存貯體互動。中的每個步驟都JobD執行相同的命令。不過，行為會根據工作容器內的AWS_BATCH_JOB_ARRAY_INDEX環境變數值而有所不同。這些驗證步驟會依序執行 (例如，JobD:0然後執行JobD:1)。SubmitJobJSON 語法如下所示。

```
{
    "jobName": "JobD",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobD-Sequential-Validation:1",
    "containerOverrides": {
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "32768"
            },
            {
                "type": "VCPU",
                "value": "1"
            }
        ]
    }
     "arrayProperties": {
         "size": 10
     },
     "dependsOn": [
         {
```

```
            "jobId": "JobC_job_ID"
        },
        {
            "type": "SEQUENTIAL"
        },

    ]
}
```

- JobE：最終的非陣列任務，可執行一些簡單的清理操作，並傳送 Amazon SNS 通知，其中包含管道已完成的訊息，以及輸出 URL 的連結。SubmitJobJSON 語法如下所示。

```
{
    "jobName": "JobE",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobE-Cleanup-and-Notification:1",
    "parameters": {
        "SourceBucket": "s3://JobD-Output-Bucket",
        "Recipient": "pipeline-notifications@mycompany.com"
    },
    "dependsOn": [
        {
            "jobId": "JobD_job_ID"
        }
    ]
}
```

教學課程：使用陣列工作索引來控制工作差異化

本教學課程說明如何使用AWS_BATCH_JOB_ARRAY_INDEX環境變數來區分子工作。每個子工作都會指派給此變數。此範例使用子工作的索引編號來讀取檔案中的特定行。然後，它用作業容器內的命令替換與該行號相關聯的參數。結果是，您可以有多個運行相同的 Docker 映像和命令參數的 AWS Batch 作業。然而，結果是不同的，因為陣列工作索引被用作修飾符。

在此教學課程中，您可以建立一個含有彩虹中所有顏色的文字檔案，每個顏色各為一行。然後，您可以為 Docker 容器建立入口點指令碼，將索引轉換為可用於色彩檔案中行號的值。索引從零開始，但行號從 1 開始。創建一個 Docker 文件，將顏色和索引文件複製到容器映像中，並將圖像設置ENTRYPOINT為入口點腳本。碼頭文件和資源是建立在推送到 Amazon ECR 的碼頭圖像。接著，您可以註冊使用新容器映像檔的工作定義、提交具有該工作定義的 AWS Batch 陣列工作，以及檢視結果。

必要條件

本教學課程具備下列先決條件：

- 運 AWS Batch 算環境。如需詳細資訊，請參閱 建立運算環境。
- AWS Batch 工作佇列和相關聯的計算環境。如需詳細資訊，請參閱 建立工作佇列。
- AWS CLI 安裝在您的本機系統上。如需詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的安裝 AWS Command Line Interface。
- 安裝在本機系統的 Docker。如需詳細資訊，請參閱 Docker 文件中的關於 Docker CE。

步驟 1：建立容器映像檔

您可以AWS_BATCH_JOB_ARRAY_INDEX在命令參數的工作定義中使用。不過，我們建議您建立容器映像檔，改為使用入口點指令碼中的變數。本節說明如何建立此類容器映像。

建置 Docker 容器影像

1. 建立新的目錄做為您的 Docker 影像工作空間，然後瀏覽至該目錄。
2. `colors.txt`在您的工作區目錄中建立一個名為的檔案，並將以下內容貼到其中。

```
red
orange
yellow
green
blue
indigo
violet
```

3. `print-color.sh`在您的工作區目錄中建立一個名為的檔案，並將以下內容貼到其中。

> **Note**
>
> LINE 變數設定為 AWS_BATCH_JOB_ARRAY_INDEX + 1，因為陣列索引起始為 0，但行號從 1 開始。COLOR變數會設定為與其行號相關聯的顏色。`colors.txt`

```
#!/bin/sh
LINE=$((AWS_BATCH_JOB_ARRAY_INDEX + 1))
```

```
COLOR=$(sed -n ${LINE}p /tmp/colors.txt)
echo My favorite color of the rainbow is $COLOR.
```

4. Dockerfile在您的工作區目錄中建立一個名為的檔案，並將下列內容貼到其中。此 Dockerfile 會將之前的檔案複製到您的容器，並將進入點指令碼設定為在啟動容器時執行。

```
FROM busybox
COPY print-color.sh /tmp/print-color.sh
COPY colors.txt /tmp/colors.txt
RUN chmod +x /tmp/print-color.sh
ENTRYPOINT /tmp/print-color.sh
```

5. 建置 Docker 映像。

```
$ docker build -t print-color .
```

6. 使用以下指令碼測試容器。此指令碼會在本機將AWS_BATCH_JOB_ARRAY_INDEX變數設定為 0，然後將其遞增，以模擬具有七個子系的陣列工作。

```
$ AWS_BATCH_JOB_ARRAY_INDEX=0
while [ $AWS_BATCH_JOB_ARRAY_INDEX -le 6 ]
do
    docker run -e AWS_BATCH_JOB_ARRAY_INDEX=$AWS_BATCH_JOB_ARRAY_INDEX print-color
    AWS_BATCH_JOB_ARRAY_INDEX=$((AWS_BATCH_JOB_ARRAY_INDEX + 1))
done
```

以下為其輸出。

```
My favorite color of the rainbow is red.
My favorite color of the rainbow is orange.
My favorite color of the rainbow is yellow.
My favorite color of the rainbow is green.
My favorite color of the rainbow is blue.
My favorite color of the rainbow is indigo.
My favorite color of the rainbow is violet.
```

步驟 2：將您的圖像推送到 Amazon ECR

現在，您已經構建並測試了 Docker 容器，請將其推送到映像存儲庫。此範例使用 Amazon ECR，但您可以使用其他登錄檔，例如 DockerHub。

1. 建立 Amazon ECR 映像儲存庫來存放您的容器映像檔。此範例僅使用 AWS CLI，但您也可以使用 AWS Management Console. 如需詳細資訊，請參閱 Amazon 彈性容器登錄使用者指南中的建立儲存庫。

```
$ aws ecr create-repository --repository-name print-color
```

2. 使用從上一步傳回的 Amazon ECR 儲存庫 URI 來標記您的`print-color`映像。

```
$ docker tag print-color aws_account_id.dkr.ecr.region.amazonaws.com/print-color
```

3. 登入您的 Amazon ECR 註冊表。如需詳細資訊，請參閱《Amazon Elastic Container Registry 使用者指南》中的登錄檔身分驗證。

```
$ aws ecr get-login-password \
    --region region | docker login \
    --username AWS \
    --password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

4. 將您的圖像推送到 Amazon ECR。

```
$ docker push aws_account_id.dkr.ecr.region.amazonaws.com/print-color
```

步驟 3：建立並註冊工作定義

現在您的 Docker 映像位於映像登錄中，您可以在 AWS Batch 工作定義中指定它。然後，您可以稍後使用它來運行陣列作業。此範例僅使用 AWS CLI. 不過，您也可以使用 AWS Management Console. 如需詳細資訊，請參閱 建立單一節點工作定義 。

建立任務定義

1. `print-color-job-def.json`在您的工作區目錄中建立一個名為的檔案，並將以下內容貼到其中。將映像儲存庫 URI 取代為您自己的映像 URI。

```
{
  "jobDefinitionName": "print-color",
  "type": "container",
  "containerProperties": {
    "image": "aws_account_id.dkr.ecr.region.amazonaws.com/print-color",
    "resourceRequirements": [
        {
```

```
            "type": "MEMORY",
            "value": "250"
        },
        {
            "type": "VCPU",
            "value": "1"
        }
    ]
  }
}
```

2. 使用註冊工作定義 AWS Batch。

```
$ aws batch register-job-definition --cli-input-json file://print-color-job-
def.json
```

步驟 4：提交 AWS Batch 陣列工作

註冊工作定義後，您可以提交使用新容器映像的 AWS Batch 陣列工作。

若要提交 AWS Batch 陣列工作

1. `print-color-job.json`在您的工作區目錄中建立一個名為的檔案，並將以下內容貼到其中。

> **Note**
>
> 此範例使用本節中提到的工作佇the section called “必要條件”列。

```
{
  "jobName": "print-color",
  "jobQueue": "existing-job-queue",
  "arrayProperties": {
    "size": 7
  },
  "jobDefinition": "print-color"
}
```

2. 將工作提交至工 AWS Batch 作佇列。請注意輸出中傳回的工作 ID。

```
$ aws batch submit-job --cli-input-json file://print-color-job.json
```

3. 描述任務的狀態並等待任務移至 SUCCEEDED。

步驟 5：檢視您的陣列工作記錄

工作達到SUCCEEDED狀態後，您可以從工作的容器檢視 CloudWatch 記錄。

若要在記錄中檢視工作的記 CloudWatch 錄

1. 請在以下位置開啟 AWS Batch 主控台。 https://console.aws.amazon.com/batch/
2. 在左側導覽窗格中，選擇 Jobs (任務)。
3. 對於 Job queue (任務佇列)，請選取佇列。
4. 在 Status (狀態) 區段，選擇 succeeded (已成功)。
5. 若要顯示陣列任務的所有子任務，選取在之前的區段中傳回的任務 ID。
6. 若要查看任務容器的日誌，選取其中一個子任務，然後選擇 View logs (查看日誌)。

7. 查看其他子任務日誌。每個任務都會傳回不同的彩虹顏色。

多節點 parallel 工作

您可以使用多節點 parallel 任務執行跨多個 Amazon EC2 執行個體的單一任務。透過 AWS Batch 多節點 parallel 任務，您可以執行大規模的高效能運算應用程式和分散式 GPU 模型訓練，無需直接啟動、設定和管理 Amazon EC2 資源。 AWS Batch 多節點 parallel 作業與任何支援 IP 型節點間通訊的架構相容。例子包括阿帕奇 MXNet TensorFlow，咖啡 2，或消息傳遞接口（MPI）。

多節點平行任務會以單一任務形式提交。不過，您的任務定義 (或任務提交節點覆寫) 會指定要為任務或哪些節點群組建立的節點數量。每個多節點平行任務皆包含會最先啟動的主要節點。在主要節點啟動

後，就會啟動和開始子節點。只有在主節點結束時，工作才會完成。然後停止所有子節點。如需詳細資訊，請參閱 節點群組。

多節點 parallel 作業節點是單租戶。這表示每個 Amazon EC2 執行個體上只會執行一個任務容器。

最終任務狀態 (SUCCEEDED 或 FAILED) 取決主要節點的最終任務狀態。若要取得多節點 parallel 工作的狀態，請使用提交工作時傳回的工作 ID 來描述工作。如果您需要子節點的詳細信息，請單獨描述每個子節點。您可以使用#*N*符號（從 0 開始）定址節點。例如，若要存取工作的第二個節點的詳細資料，請使用 API 作業描述 *aws_batch_job_id* #1。 AWS Batch DescribeJobs`started`、`stoppedAt`、`statusReason` 和 `exit` 多節點平行任務的資訊，將從主要節點填入。

如果您指定工作重試，則主節點失敗會導致另一次嘗試發生。子節點故障不會導致更多的嘗試發生。每次新嘗試的多節點平行任務，皆會更新該嘗試所關聯的子節點。

若要在上執行多節點 parallel 作業 AWS Batch，您的應用程式程式碼必須包含分散式通訊所需的架構和程式庫。

環境變數

在執行階段，每個節點都會設定所有 AWS Batch 工作接收的標準環境變數。此外，節點也會設定下列特定於多節點 parallel 工作的環境變數：

`AWS_BATCH_JOB_MAIN_NODE_INDEX`

會將此變數設為任務主要節點的索引數量。您的應用程式程式碼可以`AWS_BATCH_JOB_MAIN_NODE_INDEX`將與個別節點`AWS_BATCH_JOB_NODE_INDEX`上的項目進行比較，以判斷它是否為主節點。

`AWS_BATCH_JOB_MAIN_NODE_PRIVATE_IPV4_ADDRESS`

此變數僅在多節點 parallel 作業子節點中設定。此變數不存在於主節點上。會將此變數設為任務主要節點的私有 IPv4 地址。您的子節點應用程式程式碼可以使用此地址與主節點通訊。

`AWS_BATCH_JOB_NODE_INDEX`

會將此變數設為節點的節點索引數量。節點索引從 0 開始，而且每個節點皆會收到一個唯一的索引號碼。例如，含 10 個子系的多節點平行任務具有 0-9 的索引值。

`AWS_BATCH_JOB_NUM_NODES`

會將此變數設為您為多節點平行任務請求的節點數量。

節點群組

節點群組是共用相同容器屬性的相同工作節點群組。您最多可以 AWS Batch 為每個工作指定五個不同的節點群組。

每個群組可以有自己的容器映像、命令、環境變數，以此類推。例如，您可以提交要求主節點使用單一c5.xlarge執行個體的工作，以及五個c5.xlarge執行個體子節點。這些不同的節點群組中的每一個都可以指定不同的容器映像檔或每個工作要執行的命令。

或者，工作中的所有節點都可以使用單一節點群組。此外，您的應用程式程式碼可以區分節點角色，例如主節點和子節點。它會透過比較AWS_BATCH_JOB_MAIN_NODE_INDEX環境變數與其本身的值來執行此作業AWS_BATCH_JOB_NODE_INDEX。單一工作中最多可有 1,000 個節點。這是 Amazon ECS 叢集中執行個體的預設限制。您可以要求提高此限制。

> **Note**
>
> 目前，多節點平行任務中的所有節點群組皆必須使用相同的執行個體類型。

Job 週期

當您提交多節點 parallel 工作時，工作會進入該SUBMITTED狀態。然後，工作會等待任何工作相依性完成。工作也會移至狀RUNNABLE態。最後， AWS Batch 佈建執行工作並啟動這些執行個體所需的執行個體容量。

每個多節點平行任務皆包含主要節點。主節點是單一子工作，可 AWS Batch 監視以判斷提交的多節點工作結果。主要節點最先啟動，然後會移至 STARTING 狀態。在attemptDurationSeconds參數中指定的逾時值會套用至整個工作，而非節點。

當主節點在節點的容器執行之後到達RUNNING狀態時，子節點會啟動，而且它們也會移至STARTING狀態。子節點會以隨機順序出現。子節點的啟動時間或順序並不固定。為了確保節點的容器運行後，作業的所有節點都處於RUNNING狀態，您的應用程序代碼可以查詢 AWS Batch API 以獲取主節點和子節點信息。或者，應用程式程式碼可以等到所有節點上線後，才能開始任何分散式處理工作。主要節點的私有 IP 地址，在每個子節點中可做為 AWS_BATCH_JOB_MAIN_NODE_PRIVATE_IPV4_ADDRESS 環境變數使用。您的應用程式程式碼可以使用此資訊，對各任務之間的資料進行協調和通訊。

隨著個別節點結束，它們將移至 SUCCEEDED 或 FAILED，這取決於它們的結束程式碼。如果主要節點結束，則任務會視為完成，而所有子節點也會停止。如果子節點死亡，則 AWS Batch 不會對作業中的其他節點採取任何操作。如果您不希望工作繼續減少數量的節點，則必須將其納入應用程式程式碼中。這樣做會終止或取消工作。

運算環境考量

在設定使用 AWS Batch執行多節點平行任務的運算環境時，有幾點需要考慮。

- UNMANAGED運算環境不支援多節點 parallel 作業。
- 如果要將多節點 parallel 工作提交至計算環境，請在單一可用區域中建立叢集置放群組，並將其與您的計算資源建立關聯。如此可讓執行個體邏輯群組上的多節點 parallel 工作保持緊密接近，同時具有高網路流量潛力。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的置放群組。
- 使用 Spot 執行個體的運算環境不支援多節點 parallel 作業。
- AWS Batch 多節點 parallel 任務使用 Amazon ECS awsvpc 網路模式，為您的多節點 parallel 任務容器提供與 Amazon EC2 執行個體相同的聯網屬性。每個多節點平行任務容器皆會取得自己的彈性網路界面、主要私有 IP 地址及內部 DNS 主機名稱。網路界面是在與託管運算資源相同的 VPC 子網路中所建立。任何套用到您運算資源的安全群組，也會套用在它身上。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的使用 awsvpc 網路模式進行任務聯網。
- 您的運算環境可能不超過五個與其相關聯的安全群組。
- awsvpc網路模式不會為具有公用 IP 位址的多節點 parallel 作業提供彈性網路介面。若要存取網際網路，您的運算資源必須必須在設定為使用 NAT 閘道的私有子網路中啟動。如需詳細資訊，請參閱 Amazon VPC 使用者指南中的 NAT 閘道。節點間通訊必須使用私有 IP 地址或節點的 DNS 主機名稱。在公用子網路內的運算資源上執行的多節點 parallel 作業沒有輸出網路存取權。若要建立含私有子網路和 NAT 閘道的 VPC，請參閱建立虛擬私有雲 。
- 您的帳戶無法手動卸離或修改建立並連接到運算資源的彈性網路介面。這是為了防止意外刪除與執行中工作相關聯的 elastic network interface。若要釋出任務的彈性網路界面，請終止任務。
- 您的運算環境必須具有足夠的最大 vCPU，以支援您的多節點平行任務。
- Amazon EC2 執行個體配額包括執行任務所需的執行個體數量。例如，假設您的工作需要 30 個執行個體，但您的帳戶只能在一個區域中執行 20 個執行個體。然後，您的工作將陷入RUNNABLE狀態。
- 如果您在多節點 parallel 工作中指定節點群組的執行個體類型，您的計算環境必須啟動該執行個體類型。

GPU 工作

GPU 工作可協助您執行使用執行個體 GPU 的工作。

支援下列 Amazon EC2 GPU 型執行個體類型。如需詳細資訊，請參閱 Amazon EC2 G3 執行個體、Amazon EC2 G4 執行個體、Amazon EC2 Amazon EC2 G5 執行個體、Amazon EC2 P2 執行個體、Amazon EC2 P4d 執行個體和 Amazon EC2 P5 執行個體。

執行個體類型	GPU	記憶體	vCPU	記憶體	網路頻寬
g3s.xlarge	1	8 GiB	4	30.5 GiB	10 Gbps
g3.4xlarge	1	8 GiB	16	122 GiB	最高 10 Gbps
g3.8xlarge	2	16 GiB	32	244 GiB	10 Gbps
g3.16xlarge	4	32 GiB	64	488 GiB	25 Gbps
g4dn.xlarge	1	16 GiB	4	16 GiB	最高 25 Gbps
g4dn.2xlarge	1	16 GiB	8	32 GiB	最高 25 Gbps
g4dn.4xlarge	1	16 GiB	16	64 GiB	最高 25 Gbps
g4dn.8xlarge	1	16 GiB	32	128 GiB	50 Gbps
g4dn.12xlarge	4	64 GiB	48	192 GiB	50 Gbps
g4dn.16xlarge	1	16 GiB	64	256 GiB	50 Gbps
g5.xlarge	1	24 GiB	4	16 GiB	最高 10 Gbps
g5.2xlarge	1	24 GiB	8	32 GiB	最高 10 Gbps
g5.4xlarge	1	24 GiB	16	64 GiB	最高 25 Gbps
g5.8xlarge	1	24 GiB	32	128 GiB	25 Gbps
g5.16xlarge	1	24 GiB	64	256 GiB	25 Gbps
g5.12xlarge	4	96 GiB	48	192 GiB	40Gbps
g5.24xlarge	4	96 GiB	96	384 GiB	50 Gbps
g5.48xlarge	8	192 GiB	192	768 GiB	100 Gbps
p2.xlarge	1	12 GiB	4	61 GiB	高
p2.8xlarge	8	96 GiB	32	488 GiB	10 Gbps

執行個體類型	GPU	記憶體	vCPU	記憶體	網路頻寬
p2.16xlarge	16	192 GiB	64	732 GiB	20 Gbps
p3.2xlarge	1	16 GiB	8	61 GiB	最高 10 Gbps
p3.8xlarge	4	64 GiB	32	244 GiB	10 Gbps
p3.16xlarge	8	128 GiB	64	488 GiB	25 Gbps
p3dn.24xlarge	8	256 GiB	96	768 GiB	100 Gbps
p4d.24xlarge	8	320 GiB	96	1152 GiB	4x100 Gbps
p5.48xlarge	8	640 GiB	192	2 TiB	英鎊

> **Note**
>
> 中的 GPU 工作僅支援支援 NVIDIA GPU 和使用 x86_64 架構的執行個體類型。 AWS Batch例如，不支援G4ad和G5g執行個體族群。

工作定義的資源需求參數會指定要釘選到容器的 GPU 數目。在該工作期間，在該執行個體上執行的任何其他工作都無法使用這個 GPU 數目。運算環境中執行 GPU 作業的所有執行個體類型都必須來自p2p3、p4、p5、g3、g3sg4、或g5執行個體系列。如果未完成此操作，則 GPU 工作可能會卡在狀RUNNABLE態中。

不使用 GPU 的工作可以在 GPU 執行個體上執行。不過，在 GPU 執行個體上執行的成本可能會比在類似的非 GPU 執行個體上高。視特定 vCPU、記憶體和所需時間而定，這些非 GPU 工作可能會阻止 GPU 工作執行。

若要在 Amazon EKS 資源上建立以 GPU 為基礎的任務

本節介紹如何在 AWS Batch上執行 Amazon EKS GPU 工作負載。

內容

- 在 Amazon EKS 上建立以 GPU 為基礎的Kubernetes叢集
- 若要建立 Amazon EKS GPU 任務定義

- 在 Amazon EKS 叢集中執行 GPU 任務

在 Amazon EKS 上建立以 GPU 為基礎的Kubernetes叢集

在 Amazon EKS 上建立 GPU 型Kubernetes叢集之前，您必須先完成中的步驟。開始使 AWS Batch 用 Amazon EKS此外，還要考慮以下幾點：

- AWS Batch 支援搭載 NVIDIA GPU 的執行個體類型。
- 依預設， AWS Batch 選取具有與您 Amazon EKS 叢集控制平面Kubernetes版本相符的版本的 Amazon EKS 加速 AMI。

```
$ cat <<EOF > ./batch-eks-gpu-ce.json
{
  "computeEnvironmentName": "My-Eks-GPU-CE1",
  "type": "MANAGED",
  "state": "ENABLED",
  "eksConfiguration": {
    "eksClusterArn": "arn:aws:eks:<region>:<account>:cluster/<cluster-name>",
    "kubernetesNamespace": "my-aws-batch-namespace"
  },
  "computeResources": {
    "type": "EC2",
    "allocationStrategy": "BEST_FIT_PROGRESSIVE",
    "minvCpus": 0,
    "maxvCpus": 1024,
    "instanceTypes": [
      "p3dn.24xlarge",
      "p4d.24xlarge"
    ],
    "subnets": [
        "<eks-cluster-subnets-with-access-to-internet-for-image-pull>"
    ],
    "securityGroupIds": [
        "<eks-cluster-sg>"
    ],
    "instanceRole": "<eks-instance-profile>"
  }
}
EOF
```

```
$ aws batch create-compute-environment --cli-input-json file://./batch-eks-gpu-ce.json
```

AWS Batch 不會代表您管理 NVIDIA GPU 裝置外掛程式。您必須將此外掛程式安裝到 Amazon EKS 叢集中，並允許它以 AWS Batch 節點為目標。如需詳細資訊，請參閱（詳見）中的啟用 GPU Kubernetes Sup GitHub port。

若要將NVIDIA裝置外掛程式 (DaemonSet) 設定為目標 AWS Batch 節點，請執行下列指令。

```
# pull nvidia daemonset spec
$ curl -O https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/v0.12.2/nvidia-device-plugin.yml
# using your favorite editor, add Batch node toleration
# this will allow the DaemonSet to run on Batch nodes
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"

$ kubectl apply -f nvidia-device-plugin.yml
```

我們不建議您將運算型 (CPU 和記憶體) 工作負載與 GPU 工作負載混合在同一組運算環境和工作佇列中。這是因為運算工作可能會耗盡 GPU 容量。

若要連接工作佇列，請執行下列命令。

```
$ cat <<EOF > ./batch-eks-gpu-jq.json
 {
    "jobQueueName": "My-Eks-GPU-JQ1",
    "priority": 10,
    "computeEnvironmentOrder": [
      {
        "order": 1,
        "computeEnvironment": "My-Eks-GPU-CE1"
      }
    ]
  }
EOF

$ aws batch create-job-queue --cli-input-json file://./batch-eks-gpu-jq.json
```

若要建立 Amazon EKS GPU 任務定義

nvidia.com/gpu目前只支援，您設定的資源值必須是整數。您不能使用 GPU 的分數。如需詳細資訊，請參閱Kubernetes說明文件中的排程 GPU。

若要為 Amazon EKS 註冊 GPU 任務定義，請執行下列命令。

```
$ cat <<EOF > ./batch-eks-gpu-jd.json
{
    "jobDefinitionName": "MyGPUJobOnEks_Smi",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "hostNetwork": true,
            "containers": [
                {
                    "image": "nvcr.io/nvidia/cuda:10.2-runtime-centos7",
                    "command": ["nvidia-smi"],
                    "resources": {
                        "limits": {
                            "cpu": "1",
                            "memory": "1024Mi",
                            "nvidia.com/gpu": "1"
                        }
                    }
                }
            ]
        }
    }
}
EOF

$ aws batch register-job-definition --cli-input-json file://./batch-eks-gpu-jd.json
```

在 Amazon EKS 叢集中執行 GPU 任務

GPU 資源是不可壓縮的。 AWS Batch 會針對要求值等於限制值的 GPU 工作建立網繭規格。這是一項Kubernetes要求。

若要提交 GPU 工作，請執行下列命令。

```
$ aws batch submit-job --job-queue My-Eks-GPU-JQ1 --job-definition MyGPUJobOnEks_Smi --
job-name My-Eks-GPU-Job
```

```
# locate information that can help debug or find logs (if using Amazon CloudWatch Logs
 with Fluent Bit)
$ aws batch describe-jobs --job <job-id> | jq '.jobs[].eksProperties.podProperties |
 {podName, nodeName}'
{
  "podName": "aws-batch.f3d697c4-3bb5-3955-aa6c-977fcf1cb0ca",
  "nodeName": "ip-192-168-59-101.ec2.internal"
}
```

搜尋和篩選 AWS Batch 工作

您可以使用 AWS Batch 主控台列出工作佇列中的工作。但是，如果工作佇列中有許多工作，則可能很難找到特定工作。

您可以使用「搜尋」和「篩選」來列出符合指定搜尋條件的工作。

1. 開啟 AWS Batch 主控台。
2. 選擇 Jobs (任務)。
3. 開啟 [搜尋和篩選]。

> **Note**
>
> 如果您有多個工作，此程序可能需要幾分鐘的時間。

4. 在 [請選取工作佇列] 方塊中，選擇工作佇列 您要搜索的。
5. 在 [依內容或值篩選資源] 方塊中，選擇其中一個列出的屬性。
6. 選擇您要使用的運算子。例如，選擇「狀態」 =。

> **Tip**
>
> 若要使用不同的屬性或運算子，請關閉目前的條件。然後，選擇您想要的屬性和運算子。

7. 輸入或選擇屬性值。例如，輸入全部或部分工作名稱，或選擇「狀態 = 可執行」。
8. 在篩選清單中選擇您要的工作。

> **Tip**
>
> 如果看不到想要的工作，請捲動篩選清單。

Job 記錄

您可以將 AWS Batch 工作設定為將記錄資訊傳送至 CloudWatch 記錄檔。如此一來，您就可以在一個方便的位置檢視工作中的不同記錄。如需詳細資訊，請參閱 搭配使用 CloudWatch 記錄 AWS Batch。

您也可以使用 AWS Batch 主控台中的 Job 記錄來監視或疑難排解 AWS Batch 工作。

1. 開啟 AWS Batch 主控台。
2. 選擇 Jobs (任務)。
3. 在「Job 佇列」中，選擇所需的工作佇列。

> **Tip**
>
> 如果工作佇列中有多個工作，您可以開啟搜尋和篩選功能以更快地找到工作。如需詳細資訊，請參閱 搜尋和篩選 AWS Batch 工作。

4. 在狀態中，選擇您想要的工作狀態。
5. 選擇您想要的工作。
6. 在 [詳細資料] 頁面上，向下捲動至 [Job 記錄]。
7. 選擇 [擷取記錄]。
8. 對於需要授權，請輸入**OK**，然後選擇授權以接受 Amazon CloudWatch 費用。

> **Note**
>
> 若要撤銷您的 CloudWatch 費用授權：
>
> 1. 在左側導覽窗格中，選擇「權限」。
> 2. 對於 Job 記錄，請選擇編輯。
> 3. 清除「授權 Batch 使用」 CloudWatch 核取方塊。
> 4. 選擇儲存變更。

9. 檢閱工 AWS Batch 作的記錄資料。

> **Tip**
>
> 您可以根據「關鍵字」、「最大結果」和「排序」來篩選記錄。您也可以選擇其中一個預設時間間隔，或建立自訂間隔來自訂結果。

Job 信息

您可以檢閱 AWS Batch 工作資訊，例如狀態、工作定義和容器資訊。

1. 開啟 AWS Batch 主控台。
2. 選擇 Jobs (任務)。
3. 在「Job 佇列」中，選擇所需的工作佇列。

> **Tip**
>
> 如果工作佇列中有多個工作，您可以開啟搜尋和篩選功能以更快地找到工作。如需詳細資訊，請參閱 搜尋和篩選 AWS Batch 工作。

4. 選擇您想要的工作。

> **Note**
>
> 您也可以使用 AWS Command Line Interface (AWS CLI) 來檢視有關 AWS Batch 工作的詳細資訊。若要取得更多資訊，請參閱《指令參考》中的AWS CLI 描述工作。

Job 定義

AWS Batch 工作定義指定工作的執行方式。雖然每項任務皆必須參考任務定義，但任務定義中指定的許多參數均可在執行時間遭到覆寫。

目錄

任務定義中指定的部分屬性包括：

- 任務中的容器要使用的 Docker 影像
- 容器要使用多少個 vCPU 和多少記憶體
- 容器啟動時應執行的命令
- 容器啟動時應傳送到容器的 (如果有的話) 環境變數
- 容器應使用的任何資料磁碟區
- 您的工作應該用於 AWS 許可的哪些（如果有的話）IAM 角色

如需任務定義中可用參數的完整說明，請參閱 Job 定義參數 ContainerProperties。

建立單一節點工作定義

您必須先建立任務定義，接著才能在 AWS Batch 執行任務。此程序在單節點與多節點 parallel 作業之間略有不同。本主題特別介紹如何為非多節點 parallel AWS Batch 工作的工作建立工作定義。

您可以在 Amazon 彈性容器服務資源上建立多節點 parallel 任務定義。如需詳細資訊，請參閱 the section called “建立多節點 parallel 工作定義”。

主題

- 在 Amazon EC2 資源上建立單節點任務定義
- 在資源上建立單一節點工作定義 AWS Fargate
- 在 Amazon EKS 資源上建立單節點任務定義

在 Amazon EC2 資源上建立單節點任務定義

若要在 Amazon EC2 資源上建立新的任務定義：

1. 開啟主AWS Batch控台，網址為 https://console.aws.amazon.com/batch/。
2. 從導覽列中選擇AWS 區域要使用的。
3. 在左側導覽窗格中，選擇 [Job 定義]。
4. 選擇建立。
5. 對於協調類型，請選擇亞馬遜彈性運算雲端 (Amazon EC2)。
6. 對於 EC2 平台組態，請關閉啟用多節點 parallel 處理。
7. 在名稱中，輸入工作定義的唯一名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
8. (選擇性) 對於執行逾時，輸入逾時值 (以秒為單位)。執行逾時是未完成工作終止前的時間長度。如果嘗試超過逾時持續時間，則會停止該嘗試並移至某個FAILED狀態。如需詳細資訊，請參閱Job 逾時。最小值為 60 秒。
9. (選擇性) 開啟排程優先順序。輸入介於 0 到 100 之間的排程優先順序值。較高的值被賦予更高的優先級。
10. (選擇性) 對於 Job 嘗試，請輸入AWS Batch嘗試將工作移至RUNNABLE狀態的次數。輸入介於 1 到 10 之間的數字。
11. (選擇性) 對於「重試策略條件」，請選擇「結束時新增評估」。輸入至少一個參數值，然後選擇「作業」。對於每一組條件，「動作」都必須設定為「重試」或「結束」。這些動作意味著以下內容：
    - 重試 — AWS Batch 重試，直到達到您指定的作業嘗試次數為止。
    - 結束 — AWS Batch 停止重試工作。

> **Important**
>
> 如果您選擇 [結束時新增評估]，則必須至少設定一個參數，然後選擇 [動作] 或選擇 [結束時移除評估]。

12. (選擇性) 展開標籤，然後選擇 [新增標籤]，將標籤新增至資源。輸入機碼和選用值，然後選擇「新增標記」。
13. (選擇性) 開啟傳播標籤以將標籤從任務和任務定義傳播到 Amazon ECS 任務。
14. 選擇 [下一頁]。
15. 在容器設定區段中：

    a. 在「影像」中，選擇要用於工作的Docker影像。根據預設，Docker Hub 登錄檔中的映像為可用。您也可以用 *repository-url*/*image*:*tag* 指定其他儲存庫。名稱的長度最多可達 225 個字元。它可以包含大寫和小寫字母、數字、連字號 (-)、底線 (_)、冒號 (:)、正斜線 (/) 和數字符號 (#)。此參數會映射至 Docker Remote API 的建立容器區段中的 `Image` 以及 docker run 的 `IMAGE` 參數。

    > **Note**
    >
    > Docker映像檔架構必須符合其排程所在運算資源的處理器架構。例如，Arm基於Docker映像只能Arm根據計算資源執行。

    - Amazon ECR 公用儲存庫中的映像會使用完整`registry/repository[:tag]`或`registry/repository[@digest]`命名慣例 (例如`public.ecr.aws/`*registry_alias*`/`*my-web-app*`:`*latest*)。
    - Amazon ECR 儲存庫中的映像會使用完整的`registry/repository[:tag]`命名慣例 (例如 *aws_account_id*`.dkr.ecr.`*region*`.amazonaws.com/`*my-web-app*`:`*latest*)。
    - Docker Hub 上官方儲存庫中的映像，使用的是單一名稱 (例如，`ubuntu` 或 `mongo`)。
    - Docker Hub 上的其他儲存庫中的映像要求使用組織名稱 (例如，`amazon/amazon-ecs-agent`)。
    - 其他線上儲存庫中的映像更進一步要求使用網域名稱 (例如，`quay.io/assemblyline/ubuntu`)。

    b. 針對命令語法，請選擇 Bash 或 JSON。

c. 在 Command (命令) 中，指定要傳送至容器的命令。若要取得更簡單的指令，請像輸入指令提示一樣輸入指令。然後，驗證JSON結果是否正確並傳遞給Docker daemon. 對於更複雜的命令 (例如，使用特殊字元)，請使用 JSON 語法。

> **Tip**
>
> 選擇「資訊」以檢視Bash和JSON程式碼範例。

此參數會映射至 Docker Remote API 的建立容器區段中的 Cmd 以及 docker run 的 COMMAND 參數。如需 Docker CMD 參數的詳細資訊，請參閱 https://docs.docker.com/engine/reference/builder/#cmd。

> **Note**
>
> 您可以在指令中為參數取代和預留位置使用預設值。如需詳細資訊，請參閱參數。

d. (選擇性) 對於執行角色，請指定 IAM 角色，以授與 Amazon ECS 容器代理程式的權限，以代表您進行 AWS API 呼叫。此功能使用 Amazon ECS 身分與存取權管理角色執行任務。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 Amazon ECS 任務執行 IAM 角色。

e. 對於「Job 角色」設定，請選擇具有 AWS API 許可的 IAM 角色。此功能使用 Amazon ECS 身分與存取權管理角色執行任務。如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的任務 IAM 角色。

> **Note**
>
> 此處僅顯示具有 Amazon 彈性容器服務任務角色信任關係的角色。如需為任務建立 IAM 角色的詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的為任務建立 IAM 角色和政策。AWS Batch

16. 對於「參數」，選擇「新增參數」，將參數替代預留位置新增為「關鍵字」和「值」配對

17. 在「環境設定」區段中：

a. 對於 vCPUs，請輸入要為容器保留的 vCPUs 數目。此參數會映射到 Docker Remote API 的建立容器區段中的 CpuShares 以及 docker run 的 --cpu-shares 選項。每個 vCPU 相當於 1,024 個 CPU 共用。您必須指定至少 1 個 vCPU。

b. 在記憶體中，輸入容器可用的記憶體限制。如果您的容器嘗試超過您在此處指定的記憶體數量，則會停止容器。此參數會映射到 Docker Remote API 的建立容器區段中的 `Memory` 以及 docker run 的 `--memory` 選項。您必須為單一工作指定至少 4 MiB 的記憶體。

> **Note**
>
> 為了最大限度地提高資源使用率，請為特定執行個體類型的作業設定記憶體 如需詳細資訊，請參閱運算資源記憶體管理。

c. 在 GPU 數目中，選擇要為容器預留的 GPU 數目。

d. (選擇性) 對於環境變數，請選擇新增環境變數，將環境變數新增為名稱-值配對。這些變量被傳遞到容器。

e. (選擇性) 對於密碼，請選擇新增密碼，將密碼新增為名稱-值配對。這些機密會暴露在容器中。如需詳細資訊，請參閱。 Job 定義參數 ContainerProperties

18. 選擇 [下一頁]。

19. 在「Linux 設定」區段中：

a. 在 User (使用者) 中，輸入要在容器內使用的使用者名稱。此參數會映射到 Docker Remote API 的建立容器區段中的 `User` 以及 docker run 的 `--user` 選項。

b. (選擇性) 若要為工作容器提高主機執行個體的權限 (類似於`root`使用者)，請將「已授權」滑桿向右拖曳。此參數會映射到 Docker Remote API 的建立容器區段中的 `Privileged` 以及 docker run 的 `--privileged` 選項。

c. （選擇性）開啟啟用 init 程序以在容器內執行`init`處理程序。此過程轉發信號並重新執行過程。

20. （可選）在「文件系統配置」部分中：

a. 開啟啟用唯讀檔案系統以移除磁碟區的寫入權限。

b. 在「共用記憶體大小」中，輸入`/dev/shm`磁碟區的大小 (以 MiB 為單位)。

c. 在「最大交換大小」中，輸入容器可以使用的交換記憶體總量 (以 MiB 為單位)。

d. 對於「交換」，請輸入介於 0 到 100 之間的值，以指示容器的交換行為。如果您未指定值並啟用交換，則值預設為 60。若要取得更多資訊，請參閱〈〉中的交換。 Job 定義參數 ContainerProperties

e. (選擇性) 展開其他組態。

f. (選擇性) 對於 Tmpfs，請選擇新增 tmpfs 以新增裝載。`tmpfs`

g. (選擇性) 若為裝置，請選擇新增裝置以新增裝置：

i. 針對 Container path (容器路徑)，指定容器執行個體中的路徑，以公開對應到主機執行個體的裝置。如果保持此空白，則會在容器中使用主機路徑。

ii. 針對 Host path (主機路徑)，指定主機執行個體中的裝置的路徑。

iii. 在「權限」中，選擇要套用至裝置的一或多個權限。可用的權限包括「讀取」、「寫入」和「MCNOD」。

h. (選擇性) 對於磁碟區組態，請選擇新增磁碟區以建立要傳遞至容器的磁碟區清單。輸入磁碟區的名稱和來源路徑，然後選擇 [新增磁碟區]。您也可以選擇開啟啟用 EFS。

i. (選擇性) 對於掛接點，請選擇 [新增掛接點組態] 以新增資料磁碟區的掛接點。您必須指定來源磁碟區和容器路徑。這些掛載點會傳遞至容器執行個體Docker daemon上的。您也可以選擇將磁碟區設為唯讀。

j. (選擇性) 對於 Ulimit 組態，請選擇新增 ulimit 以新增容器的`ulimits`值。輸入 [名稱]、[軟限制] 及 [硬性限制] 值，然後選擇 [新增 ulimit]。

21. (選擇性) 在記錄設定區段中：

a. 對於記錄驅動程式，請選擇要使用的記錄驅動程式。如需有關可用記錄檔驅動程式的詳細資訊，請參閱中Job 定義參數 ContainerProperties的 LogDriver。

> **Note**
>
> 依預設，會使用`awslogs`記錄驅動程式。

b. 對於「選項」，請選擇「新增」選項以新增選項。輸入名稱-值配對，然後選擇 [新增] 選項。

c. 對於秘密，請選擇新增密碼。輸入名稱-值配對，然後選擇 [新增密碼] 以新增密碼。

> **Tip**
>
> 如需詳細資訊，請參閱。 Job 定義參數 ContainerProperties

22. 選擇 [下一頁]。

23. 對於 Job 定義檢閱，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成後，選擇 [建立工作定義]。

在資源上建立單一節點工作定義 AWS Fargate

若要在AWS Fargate資源上建立新的工作定義：

1. 開啟主AWS Batch控台，網址為 https://console.aws.amazon.com/batch/。
2. 在頂端導覽列中，選擇AWS 區域要使用的。
3. 在左側導覽窗格中，選擇 [Job 定義]。
4. 選擇建立。
5. 對於「協調流程」類型，請選擇「Fargate」。如需詳細資訊，請參閱AWS Batch關於 AWS Fargate。
6. 在名稱中，輸入工作定義的唯一名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
7. (選擇性) 對於執行逾時，輸入逾時值 (以秒為單位)。執行逾時是未完成工作終止前的時間長度。如果嘗試超過逾時持續時間，則會停止該嘗試並移至某個FAILED狀態。如需詳細資訊，請參閱Job 逾時。最小值為 60 秒。
8. (選擇性) 開啟排程優先順序。輸入介於 0 到 100 之間的排程優先順序值。較高的值的優先順序高於較低的值。
9. (選擇性) 展開標籤，然後選擇 [新增標籤]，將標籤新增至資源。開啟傳輸標籤以從工作和工作定義傳播標籤。
10. 在 Fargate 平台配置部分：
    a. 針對執行階段平台，請選擇運算環境架構。
    b. 針對作業系統系列，選擇運算環境的作業系統。
    c. 對於 CPU 架構，請選擇 vCPU 架構。
    d. 對於 Fargate 平台版本，請輸入LATEST或特定的執行階段環境版本。
    e. (選擇性) 開啟指派公用 IP，將公用 IP 位址指派給 Fargate 工作網路介面。對於在私有子網路中執行以將輸出流量傳送至網際網路的工作，私有子網路需要附加 NAT 閘道，才能將要求路由傳送至網際網路。您可能需要執行此操作，以便您可以提取容器映像。如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的 Amazon ECS 任務聯網。
    f. (選擇性) 對於暫時儲存，請輸入要配置給工作的暫時儲存容量。臨時存儲量必須介於 21 基博和 200 GiB 之間。根據預設，如果您未輸入值，則會配置 20 GiB 的暫時儲存空間。

> **Note**
>
> 暫時性儲存需要 Fargate 平台 1.4 版或更新版本。

g. 對於執行角色，請指定一個 IAM 角色，以授予 Amazon ECS 容器和 Fargate 代理程式的權限，以代表您進行 AWS API 呼叫。此功能使用 Amazon ECS 身分與存取權管理角色執行任務功能。如需包括組態先決條件的詳細資訊，請參閱 Amazon ECS 任務執行 IAM 角色 (英文) 中的 Amazon 彈性容器服務開發人員指南。

h. 對於 Job 嘗試，請輸入AWS Batch嘗試將工作移至某個RUNNABLE狀態的次數。輸入 1 到 10 之間的數字。

i. 選用) 對於「重試策略條件」，請選擇「結束時新增評估」。輸入至少一個參數值，然後選擇「作業」。對於每一組條件，「動作」都必須設定為「重試」或「結束」。這些動作意味著以下內容：

- 重試 — AWS Batch 重試，直到達到您指定的作業嘗試次數為止。
- 結束 — AWS Batch 停止重試工作。

> **Important**
>
> 如果您選擇「結束時新增評估」，則必須至少設定一個參數並選擇「動作」，或選擇「結束時移除評估」。

11. 選擇 [下一頁]。
12. 在容器設定區段中：

a. 在「影像」中，選擇要用於工作的 Docker 影像。根據預設，Docker Hub 登錄檔中的映像為可用。您也可以用 *repository-url*/*image*:*tag* 指定其他儲存庫。名稱最多可包含 225 個字元。可包含大寫及小寫字母、數字、連字號 (-)、底線 (_)、冒號 (:)、句點 (.)、斜線 (/) 和數字符號 (#)。此參數會映射至 Docker Remote API 的建立容器區段中的 Image 以及 docker run 的 IMAGE 參數。

> **Note**
>
> Docker映像檔架構必須符合其排程所在運算資源的處理器架構。例如，Arm基於Docker映像只能Arm根據計算資源執行。

- Amazon ECR 公用儲存庫中的映像會使用完整`registry/repository[:tag]`或`registry/repository[@digest]`命名慣例 (例如`public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*)。
- Amazon ECR 儲存庫中的映像會使用完整的`registry/repository[:tag]`命名慣例 (例如 *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*)。
- Docker Hub 上官方儲存庫中的映像，使用的是單一名稱 (例如，`ubuntu` 或 `mongo`)。
- Docker Hub 上的其他儲存庫中的映像要求使用組織名稱 (例如，`amazon/amazon-ecs-agent`)。
- 其他線上儲存庫中的映像更進一步要求使用網域名稱 (例如，`quay.io/assemblyline/ubuntu`)。

b. 針對命令語法，請選擇 Bash 或 JSON。

c. 在 Command (命令) 中，指定要傳送至容器的命令。對於簡單的命令，請像輸入命令提示字元一樣輸入指令，然後確認JSON結果是否正確。它傳遞給Docker守護進程。對於更複雜的命令 (例如，使用特殊字元)，請使用 JSON 語法。

> **Tip**
>
> 選擇「資訊」以檢視Bash和JSON程式碼範例。

此參數會映射至 Docker Remote API 的建立容器區段中的 `Cmd` 以及 docker run 的 `COMMAND` 參數。如需 Docker `CMD` 參數的詳細資訊，請參閱 https://docs.docker.com/engine/reference/builder/#cmd。

> **Note**
>
> 您可以在指令中為參數取代和預留位置使用預設值。如需詳細資訊，請參閱參數。

d. (選擇性) 將參數作為名稱與值對映新增至工作定義，以覆寫工作定義預設值。若要新增參數：

- 在參數中，選擇新增參數，輸入名稱-值配對，然後選擇新增參數。

> **Important**
>
> 如果選擇「新增參數」，則必須至少設定一個參數，或選擇「移除參數」

e. 在「環境設定」區段中：

i. 對於 Job 角色設定，請選擇具有 AWS API 許可的 IAM 角色。此功能使用 Amazon ECS 身分與存取權管理角色執行任務功能。如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的任務 IAM 角色。

> **Note**
>
> 此處僅顯示具有 Amazon 彈性容器服務任務角色信任關係的角色。如需有關如何為您的任AWS Batch務建立 IAM 角色的詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的為任務建立 IAM 角色和政策。

ii. 針對 vCPUs，請輸入要為容器保留的 vCPUs 數目。此參數會映射到 Docker Remote API 的建立容器區段中的 `CpuShares` 以及 docker run 的 `--cpu-shares` 選項。每個 vCPU 相當於 1,024 個 CPU 共用。您必須指定至少 1 個 vCPU。

iii. 在記憶體中，輸入容器可用的記憶體限制。如果您的容器嘗試超過此處指定的記憶體，則會停止容器。此參數會映射到 Docker Remote API 的建立容器區段中的 `Memory` 以及 docker run 的 `--memory` 選項。您必須為單一工作指定至少 4 MiB 的記憶體。

如果您使用 GuardDuty 執行階段監視，則 GuardDuty 安全性代理程式會產生輕微的記憶體額外負荷。因此記憶體限制必須包含 GuardDuty安全性代理程式的大小。如需有關 GuardDuty Security Agent 記憶體限制的資訊，請參閱《GuardDuty 使用手冊》中的 CPU 和記憶體限制。如需最佳實務的相關資訊，請參閱 Amazon ECS 開發人員指南中的啟用執行時期監控後，如何修復 Fargate 任務中的記憶體不足錯誤。

> **Note**
>
> 若要最大化您的資源使用率，請針對特定執行個體類型的工作優先處理記憶體。如需詳細資訊，請參閱運算資源記憶體管理。

f. (選擇性) 對於環境變數，請選擇新增環境變數，將環境變數新增為名稱-值配對。這些變量被傳遞到容器。

g. (選擇性) 對於密碼，請選擇新增密碼，將密碼新增為名稱-值配對。這些機密會暴露在容器中。如需詳細資訊，請參閱。 Job 定義參數 ContainerProperties

h. 選擇 [下一頁]。

13. (選擇性) 在「Linux 組態」區段中：

a. 對於使用者，請輸入要在容器內使用的使用者名稱。

b. 開啟啟用初始化程序以在容器內執行初始化程序。此過程轉發信號並重新執行過程。

c. 開啟啟用唯讀檔案系統以移除磁碟區的寫入權限。

d. (選擇性) 展開其他組態。

e. 對於掛接點組態，請選擇 [新增掛接點組態] 以新增資料磁碟區的掛接點。您必須指定來源磁碟區和容器路徑。這些掛載點會傳遞至容器執行個體Docker daemon上的。

f. 對於磁碟區組態，請選擇 [新增磁碟區] 以建立要傳遞至容器的磁碟區清單。輸入磁碟區的 [名稱] 和 [來源] 路徑，然後選擇 [新增磁碟區]。

g. 在記錄設定區段中：

i. (選擇性) 對於記錄驅動程式，請選擇要使用的記錄驅動程式。如需有關可用記錄檔驅動程式的詳細資訊，請參閱中Job 定義參數 ContainerProperties的 LogDriver。

> **Note**
>
> 依預設，會使用`awslogs`記錄驅動程式。

ii. (選擇性) 針對「選項」，選擇「新增」選項以新增選項。輸入名稱-值配對，然後選擇 [新增] 選項。

iii. (選擇性) 對於密碼，請選擇新增密碼以新增密碼。然後，輸入名稱-值配對，然後選擇 [新增密碼]。

> **Tip**
>
> 如需詳細資訊，請參閱。 Job 定義參數 ContainerProperties

14. 選擇 [下一頁]。

15. 對於 Job 定義檢閱，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成後，選擇 [建立工作定義]。

在 Amazon EKS 資源上建立單節點任務定義

若要在亞馬遜彈性 Kubernetes 服務資源上建立新的任務定義：

1. 請在以下位置開啟AWS Batch主控台。 https://console.aws.amazon.com/batch/

2. 在頂端導覽列中，選擇AWS 區域要使用的。
3. 在左側導覽窗格中，選擇 [Job 定義]。
4. 選擇建立。
5. 針對協調類型，請選擇彈性 Kubernetes 服務 (EKS)。
6. 在名稱中，輸入工作定義的唯一名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
7. (選擇性) 對於執行逾時，輸入逾時值 (以秒為單位)。執行逾時是未完成工作終止前的時間長度。如果嘗試超過逾時持續時間，則會停止該嘗試並移至某個FAILED狀態。如需詳細資訊，請參閱Job 逾時。最小值為 60 秒。
8. (選擇性) 開啟排程優先順序。輸入介於 0 到 100 之間的排程優先順序值。較高的值比較低的值具有較高的優先順序。
9. (選擇性) 展開標籤，然後選擇 [新增標籤]，將標籤新增至資源。
10. 選擇 [下一頁]。
11. 在 EKS pod 屬性區段中：
    a. 對於服務帳戶名稱，請輸入一個帳戶，該帳戶可為在pod.
    b. 開啟主機網路以使用Kubernetespod網路模型，並為內送連線開啟監聽通訊埠。僅針對外寄通訊關閉此設定。
    c. 針對 DNS 原則，請選擇下列其中一項：
       - 無值 (空值) — 會pod忽略Kubernetes環境中的 DNS 設定。
       - 預設 — 會pod繼承其執行所在節點的名稱解析組態。

         > **Note**
         > 如果未指定 DNS 原則，預設值不是預設 DNS 原則。相反 ClusterFirst，使用。

       - ClusterFirst— 任何與設定的叢集網域尾碼不相符的 DNS 查詢都會轉寄至繼承自節點的上游名稱伺服器。
       - ClusterFirstWithHostNet— 如果主機網絡已打開，則使用。
    d. (選擇性) 對於網繭標籤，請選擇新增網繭標籤，然後輸入名稱-值配對。

> **Important**
>
> 網繭標籤的前置詞不能包含kubernetes.io/k8s.io/、或batch.amazonaws.com/。

e. 選擇 [下一頁]。

f. 在「容器設定」區段中：

i. 在名稱中，輸入容器的唯一名稱。名稱必須以字母或數字開頭，最長可達 63 個字元。它可以包含大寫和小寫字母，數字和連字符 (-) 。

ii. 在「影像」中，選擇要用於工作的Docker影像。根據預設，Docker Hub 登錄檔中的映像為可用。您也可以用 *repository-url*/*image*:*tag* 指定其他儲存庫。名稱長度上限為 255 個字元。可包含大寫及小寫字母、數字、連字號 (-)、底線 (_)、冒號 (:)、句點 (.)、斜線 (/) 和數字符號 (#)。此參數對應到 Docker 遠端 API 的 [建立容器] 區段Image中，以及 IMAGE docker run

> **Note**
>
> Docker映像檔架構必須符合其排程所在運算資源的處理器架構。例如，Arm基於Docker映像只能Arm根據計算資源執行。

- Amazon ECR 公用儲存庫中的映像會使用完整registry/repository[:tag]或registry/repository[@digest]命名慣例 (例如public.ecr.aws/*registry_alias*/*my-web-app*:*latest*)。
- Amazon ECR 儲存庫中的映像會使用完整的registry/repository[:tag]命名慣例 (例如 *aws_account_id*.dkr.ecr.*region*.amazonaws.com/*my-web-app*:*latest*)。
- Docker Hub 上官方儲存庫中的映像，使用的是單一名稱 (例如，ubuntu 或 mongo)。
- Docker Hub 上的其他儲存庫中的映像要求使用組織名稱 (例如，amazon/amazon-ecs-agent)。
- 其他線上儲存庫中的映像更進一步要求使用網域名稱 (例如，quay.io/assemblyline/ubuntu)。

iii. (選擇性) 對於映像提取原則，請選擇擷取影像的時間。

iv. (選擇性) 在命令中，輸入要傳遞至容器的Bash或JSON命令。

v. (選擇性) 在引數中，輸入要傳遞至容器的引數。如果未提供引數，則使用容器映像命令。

g. (選擇性) 您可以將參數作為名稱與值對映新增至工作定義，以覆寫工作定義預設值。若要新增參數：

- 在參數中，輸入名稱-值配對，然後選擇新增參數。

> **Important**
>
> 如果選擇「新增參數」，則必須至少設定一個參數，或選擇「移除參數」

h. 在「環境設定」區段中：

i. 針對 vCPUs，請輸入要為容器保留的 vCPUs 數目。此參數會映射到 Docker Remote API 的建立容器區段中的 `CpuShares` 以及 docker run 的 `--cpu-shares` 選項。每個 vCPU 相當於 1,024 個 CPU 共用。您必須指定至少 1 個 vCPU。

ii. 在記憶體中，輸入容器可用的記憶體限制。如果您的容器嘗試超過此處指定的記憶體，則會停止容器。此參數會映射到 Docker Remote API 的建立容器區段中的 `Memory` 以及 docker run 的 `--memory` 選項。您必須為單一工作指定至少 4 MiB 的記憶體。

> **Note**
>
> 為了最大限度地提高資源使用率，請為特定執行個體類型的作業設定記憶體 如需詳細資訊，請參閱運算資源記憶體管理。

i. (選擇性) 對於環境變數，請選擇新增環境變數，將環境變數新增為名稱-值配對。這些變量被傳遞到容器。

j. (選擇性) 對於磁碟區掛載：

i. 選擇新增磁碟區掛載。

ii. 輸入 [名稱]，然後在裝載磁碟區的容器中輸入裝載路徑。

iii. 選擇唯讀可移除磁碟區的寫入權限。

iv. 選擇新增磁碟區掛載。

k. (選擇性) 對於以使用者身分執行，請輸入要執行容器處理序的使用者 ID。

> **Note**
>
> 使用者 ID 必須存在於映像中，容器才能執行。

l. (選擇性) 在執行身分群組中，輸入要執行容器處理程序執行階段的群組 ID。

> **Note**
>
> 群組 ID 必須存在於映像中，容器才能執行。

m. (選擇性) 若要為您的工作容器提高主機執行個體的權限 (類似於`root`使用者)，請將 [授權] 滑桿向右拖曳。此參數會映射到 Docker Remote API 的建立容器區段中的 `Privileged` 以及 docker run 的 `--privileged` 選項。

n. (選擇性) 開啟唯讀根檔案系統，移除對根檔案系統的寫入權限。

o. (選擇性) 開啟「以非根使用者身分執行」，以非 root 使用者身分執行中的容器。pod

> **Note**
>
> 如果「以非 root 身分執行」已開啟，則會在執行階段kubelet驗證映像檔，以驗證映像檔不會以 UID 0 的形式執行。

p. 選擇 [下一頁]。

12. 對於 Job 定義檢閱，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成後，選擇 [建立工作定義]。

建立多節點 parallel 工作定義

您必須先建立任務定義，接著才能在 AWS Batch 執行任務。此程序在單節點與多節點 parallel 作業之間略有不同。本主題特別介紹如何為AWS Batch多節點 parallel 工作建立工作定義。如需詳細資訊，請參閱多節點 parallel 工作。

> **Note**
>
> AWSFargate 不支援多節點 parallel 作業。

在 Amazon EC2 資源上建立多節點 parallel 任務定義

若要建立單節點任務定義，請參閱建立單一節點工作定義 。

若要在 Amazon 彈性運算雲端資源上建立多節點 parallel 任務定義：

1. 開啟主AWS Batch控台，網址為 https://console.aws.amazon.com/batch/。
2. 從導覽列中選取要使用的 AWS 區域。
3. 在導覽窗格中，選擇 [Job 定義]。
4. 選擇建立。
5. 對於協調類型，請選擇亞馬遜彈性運算雲端 (Amazon EC2)。
6. 對於「啟用多節點 parallel」，請開啟多節點 parallel。
7. 在名稱中，輸入工作定義的唯一名稱。名稱最多可包含 128 個字元，且可包含大寫和小寫字母、數字、連字號 (-) 和底線 (_)。
8. (選擇性) 針對「執行逾時」，指定您希望工作嘗試執行的秒數上限。如果嘗試超過逾時持續時間，則會停止該嘗試並移至某個FAILED狀態。如需詳細資訊，請參閱Job 逾時。
9. (選擇性) 開啟排程優先順序。輸入介於 0 到 100 之間的排程優先順序值。較高的值的優先順序高於較低的值。
10. (選擇性) 對於 Job 嘗試，請輸入AWS Batch嘗試將工作移至RUNNABLE狀態的次數。輸入介於 1 到 10 之間的數字。
11. (選擇性) 對於「重試策略條件」，請選擇「結束時新增評估」。輸入至少一個參數值，然後選擇「作業」。對於每一組條件，「動作」都必須設定為「重試」或「結束」。這些動作意味著以下內容：

    - 重試 — AWS Batch 重試，直到達到您指定的作業嘗試次數為止。
    - 結束 — AWS Batch 停止重試工作。

    > **Important**
    >
    > 如果您選擇 [結束時新增評估]，則必須至少設定一個參數，然後選擇 [動作] 或選擇 [結束時移除評估]。

12. (選擇性) 展開標籤，然後選擇 [新增標籤]，將標籤新增至資源。輸入機碼和選用值，然後選擇 [新增標記]。您也可以開啟傳播標籤，將標籤從任務和任務定義傳播到 Amazon ECS 任務。
13. 選擇 [下一頁]。

14. 針對 Number of nodes (節點數)，請輸入要在您任務中使用的總節點數量。
15. 針對 Main node (主要節點)，請輸入要用於主要節點的節點索引。預設的主要節點索引為 0。
16. 針對執行個體類型，選擇執行個體類型。

> **Note**
>
> 您選擇的執行個體類型會套用至所有節點。

17. 對於「參數」，選擇「新增參數」，將參數替代預留位置新增為「關鍵字」和「值」配對
18. 在「節點範圍」區段中：

    a. 選取新增節點範圍。這將創建一個節點範圍部分。

    b. 針對 Target nodes (目標節點)，請使用 *range_start*:*range_end* 標記法指定您節點群組的範圍。

    您最多可以為工作指定的節點建立五個節點範圍。節點範圍會使用節點的索引值，且節點索引會從 0 開始。請確定最終節點群組的範圍結束索引值小於您指定的節點數目一。例如，假設您指定了 10 個節點，而您想要使用單一節點群組。然後，您的結束範圍是 9。

    c. 在「影像」中，選擇要用於工作的Docker影像。根據預設，Docker Hub 登錄檔中的映像為可用。您也可以用 *repository-url*/*image*:*tag* 指定其他儲存庫。名稱最多可包含 225 個字元。它可以包含大寫和小寫字母、數字、連字號 (-)、底線 (_)、冒號 (:)、正斜線 (/) 和數字符號 (#)。此參數會映射至 Docker Remote API 的建立容器區段中的 Image 以及 docker run 的 IMAGE 參數。

    > **Note**
    >
    > Docker映像檔架構必須符合其排程所在運算資源的處理器架構。例如，Arm基於Docker映像只能Arm根據計算資源執行。

    - Amazon ECR 公用儲存庫中的映像會使用完整`registry/repository[:tag]`或`registry/repository[@digest]`命名慣例 (例如`public.ecr.aws/`*registry_alias*`/`*my-web-app*`:`*latest*)。
    - Amazon ECR 儲存庫中的映像會使用完整的`registry/repository[:tag]`命名慣例。例如：*aws_account_id*`.dkr.ecr.`*region*`.amazonaws.com/`*my-web-app*`:`*latest*
    - Docker Hub 上官方儲存庫中的映像，使用的是單一名稱 (例如，ubuntu 或 mongo)。

- Docker Hub 上的其他儲存庫中的映像要求使用組織名稱 (例如，`amazon/amazon-ecs-agent`)。
- 其他線上儲存庫中的映像更進一步要求使用網域名稱 (例如，`quay.io/assemblyline/ubuntu`)。

d. 針對命令語法，請選擇 Bash 或 JSON。

e. 在 Command (命令) 中，指定要傳送至容器的命令。對於簡單指令，您可以像在「空格分隔」頁籤中的指令提示下輸入指令一樣。然後，驗證JSON結果是否正確。JSON 結果會傳遞給Docker daemon. 如為更複雜的命令 (例如，包含特殊字元)，您可以切換到 JSON 索引標籤，然後在此輸入同等的字串陣列。

此參數會映射至 Docker Remote API 的建立容器區段中的 `Cmd` 以及 docker run 的 `COMMAND` 參數。如需 Docker `CMD` 參數的詳細資訊，請參閱 https://docs.docker.com/engine/reference/builder/#cmd。

> **Note**
> 您可以在指令中為參數取代和預留位置使用預設值。如需詳細資訊，請參閱參數。

f. 在 vCPU 中，指定保留給容器的 vCPU 數量。此參數會映射到 Docker Remote API 的建立容器區段中的 `CpuShares` 以及 docker run 的 `--cpu-shares` 選項。每個 vCPU 相當於 1,024 個 CPU 共用。您必須指定至少 1 個 vCPU。

g. 在 Memory (記憶體) 中，指定提供給任務容器使用的記憶體硬性限制 (MiB)。如果您的容器嘗試超過此處指定的記憶體，則會停止容器。此參數會映射到 Docker Remote API 的建立容器區段中的 `Memory` 以及 docker run 的 `--memory` 選項。您必須為單一工作指定至少 4 MiB 的記憶體。

> **Note**
> 為了最大限度地提高資源使用率，您可以為特定執行個體類型提供盡可能多的記憶體。如需詳細資訊，請參閱運算資源記憶體管理。

h. (選擇性) 針對 GPU 數目，指定工作使用的 GPU 數目。工作會在具有固定至該容器的指定 GPU 數目的容器上執行。

i. (選擇性) 對於 Job 角色，您可以指定 IAM 角色，為工作中的容器提供使用 AWS API 的權限。此功能使用 Amazon ECS 身分與存取權管理角色執行任務功能。如需包括組態先決條件的詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的適用任務的 IAM 角色。

> **Note**
>
> 對於在 Fargate 資源上執行的工作，需要工作角色。

> **Note**
>
> 此處僅顯示具有 Amazon 彈性容器服務任務角色信任關係的角色。如需為任務建立 IAM 角色的詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的為任務建立 IAM 角色和政策。AWS Batch

j. (選擇性) 對於執行角色，請指定 IAM 角色，以授與 Amazon ECS 容器代理程式的權限，以代表您進行 AWS API 呼叫。此功能使用 Amazon ECS 身分與存取權管理角色執行任務功能。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 Amazon ECS 任務執行 IAM 角色。

19. (選擇性) 展開其他組態：

a. 對於環境變數，請選擇新增環境變數，將環境變數新增為名稱-值配對。這些變量被傳遞到容器。

b. 對於 Job 角色設定，您可以指定 IAM 角色，為工作中的容器提供使用 AWS API 的許可。此功能使用 Amazon ECS 身分與存取權管理角色執行任務功能。如需包括組態先決條件的詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的適用任務的 IAM 角色。

> **Note**
>
> 對於在 Fargate 資源上執行的工作，需要工作角色。

> **Note**
>
> 此處僅顯示具有 Amazon 彈性容器服務任務角色信任關係的角色。如需有關如何為您的任AWS Batch務建立 IAM 角色的詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的為任務建立 IAM 角色和政策。

c. 對於執行角色，請指定 IAM 角色，以授與 Amazon ECS 容器代理程式的權限，以代表您進行 AWS API 呼叫。此功能使用 Amazon ECS 身分與存取權管理角色執行任務功能。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 Amazon ECS 任務執行 IAM 角色。

20. 在「安全組態」區段中：

a. (選擇性) 若要將工作的容器提升主機執行個體上的權限 (類似於root使用者)，請開啟「已授權」。此參數會映射到 Docker Remote API 的建立容器區段中的 Privileged 以及 docker run 的 --privileged 選項。

b. (選擇性) 對於使用者，請輸入要在容器內使用的使用者名稱。此參數會映射到 Docker Remote API 的建立容器區段中的 User 以及 docker run 的 --user 選項。

c. (選擇性) 對於密碼，請選擇新增密碼，將密碼新增為名稱-值配對。這些機密會暴露在容器中。如需詳細資訊，請參閱。 Job 定義參數 ContainerProperties

21. 在「Linux 設定」區段中：

a. 開啟啟用唯讀檔案系統以移除磁碟區的寫入權限。

b. (選擇性) 開啟啟用init程序以在容器內執行init處理程序。此過程轉發信號並重新執行過程。

c. 在「共用記憶體大小」中，輸入/dev/shm磁碟區的大小 (以 MiB 為單位)。

d. 在「最大交換大小」中，輸入容器可以使用的交換記憶體總量 (以 MiB 為單位)。

e. 對於「交換」，請輸入介於 0 到 100 之間的值，以指示容器的交換行為。如果您未指定值並啟用交換，則值預設為 60。若要取得更多資訊，請參閱〈〉中的交換。 Job 定義參數 ContainerProperties

f. (選擇性) 若為裝置，請選擇新增裝置以新增裝置：

i. 針對 Container path (容器路徑)，指定容器執行個體中的路徑，以公開對應到主機執行個體的裝置。如果保持此空白，則會在容器中使用主機路徑。

ii. 針對 Host path (主機路徑)，指定主機執行個體中的裝置的路徑。

iii. 在「權限」中，選擇要套用至裝置的一或多個權限。可用的權限包括「讀取」、「寫入」和「MCNOD」。

22. (選擇性) 對於掛接點，請選擇 [新增掛接點組態] 以新增資料磁碟區的掛接點。您必須指定來源磁碟區和容器路徑。這些掛載點會傳遞至容器執行個體上的Docker精靈。您也可以選擇將磁碟區設為唯讀。

23. (選擇性) 對於 Ulimit 組態，請選擇新增 ulimit 以新增容器的ulimits值。輸入 [名稱]、[軟限制] 及 [硬性限制] 值，然後選擇 [新增 ulimit]。

24. (選擇性) 對於磁碟區組態，請選擇新增磁碟區以建立要傳遞至容器的磁碟區清單。輸入磁碟區的名稱和來源路徑，然後選擇 [新增磁碟區]。您也可以選擇開啟啟用 EFS。
25. (選擇性) 對於 Tmpfs，請選擇「新增 tmpfs」以新增裝載。`tmpfs`
26. (選擇性) 在記錄設定區段中：

    a. 對於記錄驅動程式，請選擇要使用的記錄驅動程式。如需有關可用記錄檔驅動程式的詳細資訊，請參閱中Job 定義參數 ContainerProperties的 LogDriver。

    > **Note**
    > 依預設，會使用`awslogs`記錄驅動程式。

    b. 對於「選項」，請選擇「新增」選項以新增選項。輸入名稱-值配對，然後選擇 [新增] 選項。

    c. 對於秘密，請選擇新增密碼。輸入名稱-值配對，然後選擇 [新增密碼] 以新增密碼。

    > **Tip**
    > 如需詳細資訊，請參閱。 Job 定義參數 ContainerProperties

27. 選擇 [下一頁]。
28. 對於 Job 定義檢閱，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成後，選擇 [建立工作定義]。

使用建立工作定義 ContainerProperties

以下是包含單一容器的空白工作定義範本。您可以使用此範本建立工作定義，然後可將其儲存至檔案並與 AWS CLI `--cli-input-json`選項搭配使用。如需這些參數的相關資訊，請參閱 Job 定義參數 ContainerProperties。

```
{
    "jobDefinitionName": "",
    "type": "container",
    "parameters": {
        "KeyName": ""
    },
    "schedulingPriority": 0,
    "containerProperties": {
        "image": "",
```

```
        "vcpus": 0,
        "memory": 0,
        "command": [
            ""
        ],
        "jobRoleArn": "",
        "executionRoleArn": "",
        "volumes": [
            {
                "host": {
                    "sourcePath": ""
                },
                "name": "",
                "efsVolumeConfiguration": {
                    "fileSystemId": "",
                    "rootDirectory": "",
                    "transitEncryption": "ENABLED",
                    "transitEncryptionPort": 0,
                    "authorizationConfig": {
                        "accessPointId": "",
                        "iam": "DISABLED"
                    }
                }
            }
        ],
        "environment": [
            {
                "name": "",
                "value": ""
            }
        ],
        "mountPoints": [
            {
                "containerPath": "",
                "readOnly": true,
                "sourceVolume": ""
            }
        ],
        "readonlyRootFilesystem": true,
        "privileged": true,
        "ulimits": [
            {
                "hardLimit": 0,
                "name": "",
```

```
                "softLimit": 0
            }
        ],
        "user": "",
        "instanceType": "",
        "resourceRequirements": [
            {
                "value": "",
                "type": "MEMORY"
            }
        ],
        "linuxParameters": {
            "devices": [
                {
                    "hostPath": "",
                    "containerPath": "",
                    "permissions": [
                        "WRITE"
                    ]
                }
            ],
            "initProcessEnabled": true,
            "sharedMemorySize": 0,
            "tmpfs": [
                {
                    "containerPath": "",
                    "size": 0,
                    "mountOptions": [
                        ""
                    ]
                }
            ],
            "maxSwap": 0,
            "swappiness": 0
        },
        "logConfiguration": {
            "logDriver": "syslog",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
```

```
                }
            ]
        },
        "secrets": [
            {
                "name": "",
                "valueFrom": ""
            }
        ],
        "networkConfiguration": {
            "assignPublicIp": "DISABLED"
        },
        "fargatePlatformConfiguration": {
            "platformVersion": ""
        }
    },
    "nodeProperties": {
        "numNodes": 0,
        "mainNode": 0,
        "nodeRangeProperties": [
            {
                "targetNodes": "",
                "container": {
                    "image": "",
                    "vcpus": 0,
                    "memory": 0,
                    "command": [
                        ""
                    ],
                    "jobRoleArn": "",
                    "executionRoleArn": "",
                    "volumes": [
                        {
                            "host": {
                                "sourcePath": ""
                            },
                            "name": "",
                            "efsVolumeConfiguration": {
                                "fileSystemId": "",
                                "rootDirectory": "",
                                "transitEncryption": "DISABLED",
                                "transitEncryptionPort": 0,
                                "authorizationConfig": {
                                    "accessPointId": "",
```

```
                    "iam": "ENABLED"
                }
            }
        }
    ],
    "environment": [
        {
            "name": "",
            "value": ""
        }
    ],
    "mountPoints": [
        {
            "containerPath": "",
            "readOnly": true,
            "sourceVolume": ""
        }
    ],
    "readonlyRootFilesystem": true,
    "privileged": true,
    "ulimits": [
        {
            "hardLimit": 0,
            "name": "",
            "softLimit": 0
        }
    ],
    "user": "",
    "instanceType": "",
    "resourceRequirements": [
        {
            "value": "",
            "type": "MEMORY"
        }
    ],
    "linuxParameters": {
        "devices": [
            {
                "hostPath": "",
                "containerPath": "",
                "permissions": [
                    "WRITE"
                ]
            }
```

```
                    ],
                    "initProcessEnabled": true,
                    "sharedMemorySize": 0,
                    "tmpfs": [
                        {
                            "containerPath": "",
                            "size": 0,
                            "mountOptions": [
                                ""
                            ]
                        }
                    ],
                    "maxSwap": 0,
                    "swappiness": 0
                },
                "logConfiguration": {
                    "logDriver": "awslogs",
                    "options": {
                        "KeyName": ""
                    },
                    "secretOptions": [
                        {
                            "name": "",
                            "valueFrom": ""
                        }
                    ]
                },
                "secrets": [
                    {
                        "name": "",
                        "valueFrom": ""
                    }
                ],
                "networkConfiguration": {
                    "assignPublicIp": "DISABLED"
                },
                "fargatePlatformConfiguration": {
                    "platformVersion": ""
                }
            }
        }
    ]
},
"retryStrategy": {
```

```
        "attempts": 0,
        "evaluateOnExit": [
            {
                "onStatusReason": "",
                "onReason": "",
                "onExitCode": "",
                "action": "RETRY"
            }
        ]
    },
    "propagateTags": true,
    "timeout": {
        "attemptDurationSeconds": 0
    },
    "tags": {
        "KeyName": ""
    },
    "platformCapabilities": [
        "EC2"
    ],
    "eksProperties": {
        "podProperties": {
            "serviceAccountName": "",
            "hostNetwork": true,
            "dnsPolicy": "",
            "containers": [
                {
                    "name": "",
                    "image": "",
                    "imagePullPolicy": "",
                    "command": [
                        ""
                    ],
                    "args": [
                        ""
                    ],
                    "env": [
                        {
                            "name": "",
                            "value": ""
                        }
                    ],
                    "resources": {
                        "limits": {
```

```
                            "KeyName": ""
                        },
                        "requests": {
                            "KeyName": ""
                        }
                    },
                    "volumeMounts": [
                        {
                            "name": "",
                            "mountPath": "",
                            "readOnly": true
                        }
                    ],
                    "securityContext": {
                        "runAsUser": 0,
                        "runAsGroup": 0,
                        "privileged": true,
                        "readOnlyRootFilesystem": true,
                        "runAsNonRoot": true
                    }
                }
            ],
            "volumes": [
                {
                    "name": "",
                    "hostPath": {
                        "path": ""
                    },
                    "emptyDir": {
                        "medium": "",
                        "sizeLimit": ""
                    },
                    "secret": {
                        "secretName": "",
                        "optional": true
                    }
                }
            ]
        }
    }
}
```

> **Note**
>
> 您可以使用下列命令產生單一容器工作定義範本： AWS CLI
>
> ```
> $ aws batch register-job-definition --generate-cli-skeleton
> ```

Job 定義參數 ContainerProperties

使用的 Job 定義分為數個部分：ContainerProperties

- 工作定義名稱
- 工作定義的類型
- 參數替換佔位符默認值
- 工作的容器屬性
- 在 Amazon EKS 資源上執行任務所需的任務定義的 Amazon EKS 屬性
- 多節點 parallel 工作所需的節點屬性
- 在 Fargate 資源上執行作業所需的平台功能
- 工作定義的預設標記傳輸詳細資訊
- 工作定義的預設重試策略
- 工作定義的預設排程優先順序
- 工作定義的預設標籤
- 工作定義的預設逾時

內容

- 節點屬性
- 重試策略
- 排程優先權
- 標籤
- 逾時

Job 定義名稱

`jobDefinitionName`

您必須在註冊任務定義時指定名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。第一個以該名稱註冊的工作定義會被賦予 1 的修訂版本。後續使用該名稱註冊的任何任務定義，將取得遞增的修訂版號碼。

類型：字串

必要：是

Type

`type`

註冊任務定義時，需指定任務類型。如果作業在 Fargate 資源上執行，則`multinode`不支援。如需多節點平行任務的詳細資訊，請參閱「the section called “建立多節點 parallel 工作定義”」。

類型：字串

有效值：`container` | `multinode`

必要：是

參數

`parameters`

提交工作時，您可以指定取代預留位置或覆寫預設工作定義參數的參數。任務提交要求中的參數，優先於任務定義中的預設值。這表示您可以對使用相同格式的多個工作使用相同的工作定義。您也可以在提交時以程式設計方式變更命令中的值。

類型：字串到字串映射

必要：否

註冊任務定義時，您可以在任務容器屬性的 command 欄位使用參數替換預留位置。語法如下。

```
"command": [
    "ffmpeg",
    "-i",
    "Ref::inputfile",
    "-c",
    "Ref::codec",
    "-o",
    "Ref::outputfile"
]
```

在上述範例中，命令中有 *Ref::inputfile*、*Ref::codec* 和 *Ref::outputfile* 參數替換預留位置。您可以使用工作定義中的parameters物件來設定這些預留位置的預設值。例如，若要設定 *Ref::codec* 預留位置的預設值，您應在任務定義中指定下列各項：

```
"parameters" : {"codec" : "mp4"}
```

提交此工作定義以執行時，容器命令中的*Ref::codec*引數會取代為預設值mp4。

容器屬性

註冊工作定義時，請指定放置工作時，傳送至容器執行個體上 Docker 精靈的容器屬性清單。任務定義允許使用以下的容器屬性。針對單節點任務，這些容器屬性會設定在任務定義層級。針對多節點平行任務，每個節點群組的容器屬性會設定在 節點屬性 層級。

command

傳遞至容器的命令。此參數會映射至 Docker Remote API 的建立容器區段中的 Cmd 以及 docker run 的 COMMAND 參數。如需 Docker CMD 參數的詳細資訊，請參閱 https://docs.docker.com/engine/reference/builder/#cmd。

```
"command": ["string", ...]
```

類型：字串陣列

必要：否

`environment`

傳遞至容器的環境變數。此參數會映射到 Docker Remote API 的建立容器區段中的 `Env` 以及 docker run 的 `--env` 選項。

> **Important**
>
> 不建議您對敏感資訊 (例如憑證) 使用純文字環境變數。

> **Note**
>
> 環境變數不得以開頭`AWS_BATCH`。此命名慣例會保留給 AWS Batch 服務所設定的變數使用。

類型：金鑰值對的陣列

必要：否

`name`

環境變數的名稱。

類型：字串

必要：是，使用 `environment` 時。

`value`

環境變數的值。

類型：字串

必要：是，使用 `environment` 時。

```
"environment" : [
    { "name" : "envName1", "value" : "envValue1" },
    { "name" : "envName2", "value" : "envValue2" }
]
```

`executionRoleArn`

註冊工作定義時，您可以指定 IAM 角色。該角色為 Amazon ECS 容器代理程式提供許可，以代表您呼叫在其關聯政策中指定的 API 動作。在 Fargate 資源上執行的工作必須提供執行角色。如需詳細資訊，請參閱 AWS Batch 執行 IAM 角色。

類型：字串

必要：否

`fargatePlatformConfiguration`

在 Fargate 資源上執行之作業的平台組態。在 EC2 資源上執行的任務不得指定此參數。

類型：FargatePlatform配置對象

必要：否

`platformVersion`

AWS Fargate 平台版本用於作業，或使`LATEST`用最新的，批准的 AWS Fargate 平台版本。

類型：字串

預設：`LATEST`

必要：否

`image`

用於開始工作的影像。此字串會直接傳遞至 Docker 常駐程式。根據預設，Docker Hub 登錄檔中的映像為可用。您也可以用 *`repository-url`*`/`*`image`*`:`*`tag`* 指定其他儲存庫。允許最多 255 個字元 (大小寫)、數字、連字號、底線、等號、句號、正斜線、井號。此參數會映射至 Docker Remote API 的建立容器區段中的 `Image` 以及 docker run 的 `IMAGE` 參數。

> **Note**
>
> Docker映像檔架構必須符合其排程所在運算資源的處理器架構。例如，Arm基於Docker映像只能Arm根據計算資源執行。

- Amazon ECR 公用儲存庫中的映像會使用完整`registry/repository[:tag]`或`registry/repository[@digest]`命名慣例 (例如`public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*)。

- Amazon ECR 儲存庫中的映像會使用完整的`registry/repository:[tag]`命名慣例。例如 *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*。
- Docker Hub 上官方儲存庫中的映像，使用的是單一名稱 (例如，`ubuntu` 或 `mongo`)。
- Docker Hub 上的其他儲存庫中的映像要求使用組織名稱 (例如，`amazon/amazon-ecs-agent`)。
- 其他線上儲存庫中的映像更進一步要求使用網域名稱 (例如，`quay.io/assemblyline/ubuntu`)。

類型：字串

必要：是

`instanceType`

用於多節點平行任務的執行個體類型。所有節點平行任務中的所有節點群組皆必須使用相同的執行個體類型。此參數不適用於單節點容器作業或在 Fargate 資源上執行的工作。

類型：字串

必要：否

`jobRoleArn`

註冊工作定義時，您可以指定 IAM 角色。角色提供任務容器權限，允許其代表您呼叫相關聯政策中指定的 API 動作。如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的任務 IAM 角色。

類型：字串

必要：否

`linuxParameters`

Linux 特定的修改，會套用到容器，例如用於裝置映射的詳細資訊。

```
"linuxParameters": {
    "devices": [
        {
            "hostPath": "string",
            "containerPath": "string",
            "permissions": [
                "READ", "WRITE", "MKNOD"
```

```
            ]
        }
    ],
    "initProcessEnabled": true|false,
    "sharedMemorySize": 0,
    "tmpfs": [
        {
            "containerPath": "string",
            "size": integer,
            "mountOptions": [
                "string"
            ]
        }
    ],
    "maxSwap": integer,
    "swappiness": integer
}
```

類型：LinuxParameters 物件

必要：否

`devices`

映射到容器的裝置列表。此參數對應到 Docker Remote API Create a container (建立容器) 一節中的 `Devices`，以及 Docker run 的 `--device` 選項。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

類型：Device 物件的陣列

必要：否

`hostPath`

主機容器執行個體中可用裝置所在的路徑。

類型：字串

必要：是

containerPath

裝置暴露在容器中的路徑。如果未指定，則裝置會在與主機路徑相同的路徑上公開。

類型：字串

必要：否

permissions

容器中的裝置的許可。如果未指定權限，則會將權限設定為READWRITE、和MKNOD。

類型：字串陣列

必要：否

有效值：READ | WRITE | MKNOD

initProcessEnabled

若為 true，請在容器內執行 init 處理程序，該處理程序可轉寄訊號及獲得處理程序。此參數會映射到 docker run 的 --init 選項。在您的容器執行個體上，此參數需要 1.25 版或更新版本的 Docker Remote API。若要檢查容器執行個體的 Docker Remote API 版本，請登入容器執行個體，並執行下列命令：sudo docker version | grep "Server API version"

類型：布林值

必要：否

maxSwap

工作可以使用的交換記憶體總量 (以 MiB 為單位)。此參數將會轉換為 docker run 的 --memory-swap 選項，其中值是容器記憶體與 maxSwap 值的總和。如需詳細資訊，請參閱 Docker 文件中的 --memory-swap 詳細資訊。

如果將 maxSwap 值指定為 0，容器不會使用交換。接受的值為 0 或任何正整數。如果省略maxSwap參數，則容器會針對其執行的容器執行個體使用 swap 組態。必須設定 maxSwap 值，才能使用 swappiness 參數。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

類型：整數

必要：否

`sharedMemorySize`

`/dev/shm` 磁碟區的大小值 (以 MiB 為單位)。此參數會映射到 docker run 的 `--shm-size` 選項。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

類型：整數

必要：否

`swappiness`

您可藉此調整容器的記憶體交換行為。的`swappiness`值會`0`導致交換不會發生，除非絕對必要。為 `100` 的 `swappiness` 值導致積極地交換頁面。接受的值為介於 `0` 與 `100` 之間的整數。如果未指定 `swappiness` 參數，則會使用預設值 `60`。如果未對 `maxSwap` 指定值，則會忽略此參數。如果 `maxSwap` 設定為 0，則容器不會使用交換。此參數會映射到 docker run 的 `--memory-swappiness` 選項。

當您使用每個容器交換組態時，請考量下列事項。

- 必須在容器執行個體上啟用和配置交換空間，供容器使用。

  > **Note**
  >
  > 根據預設，Amazon ECS 最佳化 AMI 沒有啟用交換功能。您必須在執行個體上啟用交換，才能使用此功能。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的執行個體存放區交換磁碟區或如何使用交換檔分配記憶體做為 Amazon EC2 執行個體中的交換空間？ 。

- 交換空間參數僅針對使用 EC2 資源的任務定義提供支援。
- 如果從任務定義中省略 `maxSwap` 和 `swappiness` 參數，每個容器的預設 `swappiness` 值都為 60。總交換使用量限制為容器記憶體保留的兩倍。

> **Note**
> 此參數不適用於在 Fargate 資源上執行的任務。

類型：整數

必要：否

`tmpfs`

tmpfs 掛載的容器路徑、掛載選項和大小。

類型：Tmpfs 物件的陣列

> **Note**
> 此參數不適用於在 Fargate 資源上執行的任務。

必要：否

`containerPath`

掛載 tmpfs 磁碟區之容器中的絕對檔案路徑。

類型：字串

必要：是

`mountOptions`

tmpfs 磁碟區掛載選項的清單。

有效值：`defaults"" | ro "" rw "" | suid "" nosuid "|" dev "" | nodev "" exec "|" noexec "" | sync "" async "|" dirsync "|" remount "|" mand "" | nomand "" atime "|" noatime "" | diratime "" nodiratime "|" "bind" | rbind "" "|unbindable" "runbindable" | "private" "|rprivate" "|shared" rshared "|" slave "|" rslave "" | relatime "" norelatime "|" strictatime "|" nostrictatime "|」mode「|" uid "|" gid "|" "|nr_inodes" "nr_blocks" | ""mpol」`

類型：字串陣列

必要：否

`size`

tmpfs 磁碟區大小 (以 MiB 為單位)。

類型：整數

必要：是

`logConfiguration`

工作的記錄組態規格。

此參數對應到 Docker Remote API Create a container (建立容器) 一節中的 `LogConfig`，以及 Docker run 的 `--log-driver` 選項。根據預設，容器會和 Docker 常駐程式使用一樣的日誌記錄驅動程式。不過，容器可以使用與 Docker 精靈不同的記錄驅動程式，方法是在容器定義中使用此參數指定記錄驅動程式。若要針對容器使用不同的記錄驅動程式，必須在容器執行個體或其他記錄伺服器上設定記錄系統，才能提供遠端記錄選項。如需支援的不同日誌驅動程式選項的詳細資訊，請參閱 Docker 文件中的 Configure logging drivers (設定日誌驅動程式)。

> **Note**
>
> AWS Batch 目前支援 Docker 守護程式可用的記錄驅動程式子集 (以LogConfiguration資料類型顯示)。

在您的容器執行個體上，此參數需要 1.18 版或更新版本的 Docker Remote API。若要檢查容器執行個體的 Docker Remote API 版本，請登入容器執行個體，並執行下列命令：`sudo docker version | grep "Server API version"`

```
"logConfiguration": {
    "devices": [
        {
            "logDriver": "string",
            "options": {
                "optionName1" : "optionValue1",
                "optionName2" : "optionValue2"
            }
            "secretOptions": [
              {
                  "name" : "secretOptionName1",
```

```
                "valueFrom" : "secretOptionArn1"
            },
            {
                "name" : "secretOptionName2",
                "valueFrom" : "secretOptionArn2"
            }
        ]
    }
  ]
}
```

類型：LogConfiguration 物件

必要：否

`logDriver`

用於工作的記錄驅動程式。依預設，會 AWS Batch 啟用記`awslogs`錄驅動程式。根據預設，針對此參數列出的有效值是 Amazon ECS 容器代理程式可與之通訊的日誌驅動程式。

此參數對應到 Docker Remote API Create a container (建立容器) 一節中的 `LogConfig`，以及 Docker run 的 `--log-driver` 選項。依預設，工作會使用 Docker 精靈所使用的相同記錄驅動程式。不過，工作可以使用與 Docker 精靈不同的記錄驅動程式，方法是在工作定義中使用此參數指定記錄驅動程式。如果您想要為工作指定其他記錄驅動程式，則必須在計算環境中的容器執行個體上設定記錄系統。或者，您也可以在另一部記錄伺服器上進行設定，以提供遠端記錄選項。如需支援的不同日誌驅動程式選項的詳細資訊，請參閱 Docker 文件中的 Configure logging drivers (設定日誌驅動程式)。

> **Note**
>
> AWS Batch 目前支援 Docker 精靈可用的記錄驅動程式子集。未來的 Amazon ECS 容器代理程式版本可能會提供更多可用的其他日誌驅動程式。

支援的記錄驅動程式為 `awslogs`、`fluentd`、`gelf`、`json-file`、`journald`、`logentries`、`syslog` 和 `splunk`。

> **Note**
>
> 在 Fargate 資源上執行的工作僅限於`awslogs`和`splunk`記錄驅動程式。

在您的容器執行個體上，此參數需要 1.18 版或更新版本的 Docker Remote API。若要檢查容器執行個體的 Docker Remote API 版本，請登入容器執行個體，並執行下列命令：`sudo docker version | grep "Server API version"`

> **Note**
>
> 在容器執行個體上執行的 Amazon ECS 容器代理程式必須向`ECS_AVAILABLE_LOGGING_DRIVERS`環境變數註冊該執行個體上可用的記錄驅動程式。否則，放置在該執行個體上的容器將無法使用這些記錄檔設定選項。如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的 Amazon ECS 容器代理程式組態。

`awslogs`

指定 Amazon CloudWatch 日誌記錄驅動程式。如需詳細資訊，請參閱 Docker 文件中的使用 awslogs 日誌驅動程式和 Amazon CloudWatch 日誌記錄驅動程式。

`fluentd`

指定 Fluentd 記錄驅動程式。如需包括使用方式和選項在內的詳細資訊，請參閱 Docker 文件中的 Fluentd 記錄驅動程式。

`gelf`

指定 Graylog 延伸格式 (GELF) 記錄驅動程式。如需包括使用方式和選項在內的詳細資訊，請參閱 Docker 文件中的 Graylog 延伸格式記錄驅動程式。

`journald`

指定 journald 記錄驅動程式。如需包括使用方式和選項在內的詳細資訊，請參閱 Docker 文件中的日誌記錄驅動程式。

`json-file`

指定 JSON 檔案記錄驅動程式。如需包括使用方式和選項在內的詳細資訊，請參閱 Docker 文件中的 JSON 檔案記錄驅動程式。

`splunk`

指定 Splunk 記錄驅動程式。如需包括使用方式和選項在內的詳細資訊，請參閱 Docker 文件中的 Splunk 記錄驅動程式。

`syslog`

指定 syslog 記錄驅動程式。如需包括使用方式和選項在內的詳細資訊，請參閱 Docker 文件中的 Syslog 記錄驅動程式。

類型：字串

必要：是

有效值：`awslogs` | `fluentd` | `gelf` | `journald` | `json-file` | `splunk` | `syslog`

> **Note**
>
> 如果您有先前未列出的自訂驅動程式，而您想要使用 Amazon ECS 容器代理程式，則可以分叉提供的 Amazon ECS 容器代理程式專案，然後自訂它以 GitHub與該驅動程式搭配使用。我們鼓勵您為想要進行的變更提交提取請求。不過，Amazon Web Services 目前不支援執行此軟體修改副本的請求。

`options`

記錄組態選項以傳送至工作的記錄驅動程式。

在您的容器執行個體上，此參數需要 1.19 版或更新版本的 Docker Remote API。

類型：字串到字串映射

必要：否

`secretOptions`

此物件代表要傳送至日誌組態的秘密。如需詳細資訊，請參閱 指定敏感資料。

類型：物件陣列

必要：否

`name`

要在工作中設定的記錄驅動程式選項名稱。

類型：字串

必要：是

`valueFrom`

要公開給容器日誌組態的秘密的 Amazon 資源名稱 (ARN)。支援的值是秘 Secrets Manager 碼的完整 ARN，或 SSM 參數存放區中參數的完整 ARN。

> **Note**
>
> 如果 SSM 參數存放區參數與您啟動的工作位於相同 AWS 區域 的位置，則您可以使用完整的 ARN 或參數名稱。如果參數存在於不同區域，則必須指定完整 ARN。

類型：字串

必要：是

`memory`

此參數已過時，請*resourceRequirements*改用。

為工作保留的記憶體 MiB 數目。

作為如何使用的範例`resourceRequirements`，如果您的工作定義包含類似下列的語法。

```
"containerProperties": {
  "memory": 512
}
```

使用的等效語`resourceRequirements`法如下。

```
"containerProperties": {
  "resourceRequirements": [
    {
      "type": "MEMORY",
      "value": "512"
    }
  ]
}
```

類型：整數

必要：是

`mountPoints`

容器中資料磁碟區的掛載點。此參數會映射到 Docker Remote API 的建立容器區段中的 `Volumes` 以及 docker run 的 `--volume` 選項。

```
"mountPoints": [
                {
                  "sourceVolume": "string",
                  "containerPath": "string",
                  "readOnly": true|false
                }
              ]
```

類型：物件陣列

必要：否

`sourceVolume`

要掛載的磁碟區名稱。

類型：字串

必要：是，使用 `mountPoints` 時。

`containerPath`

容器上掛接主機磁碟區的路徑。

類型：字串

必要：是，使用 `mountPoints` 時。

`readOnly`

如果此數值為 `true`，容器擁有磁碟區的唯讀存取權。如果此值為 `false`，則容器可寫入磁碟區。

類型：布林值

必要：否

預設：False

`networkConfiguration`

在 Fargate 資源上執行之工作的網路組態。在 EC2 資源上執行的任務不得指定此參數。

```
"networkConfiguration": {
   "assignPublicIp": "string"
}
```

類型：物件陣列

必要：否

`assignPublicIp`

指示任務是否有公有 IP 地址。如果工作需要輸出網路存取，這是必要的。

類型：字串

有效值：`ENABLED` | `DISABLED`

必要：否

預設：`DISABLED`

`privileged`

此參數為 true 時，容器便會取得主機容器執行個體的更高許可 (類似 `root` 使用者)。此參數會映射到 Docker Remote API 的建立容器區段中的 `Privileged` 以及 docker run 的 `--privileged` 選項。此參數不適用於在 Fargate 資源上執行的任務。不要提供它或將其指定為假。

```
"privileged": true|false
```

類型：布林值

必要：否

`readonlyRootFilesystem`

此參數為 true 時，容器會取得根檔案系統的唯讀存取權。此參數會映射到 Docker Remote API 的建立容器區段中的 `ReadonlyRootfs` 以及 docker run 的 `--read-only` 選項。

```
"readonlyRootFilesystem": true|false
```

類型：布林值

必要：否

`resourceRequirements`

指派給容器的資源類型和數量。支援的資源包括 GPU MEMORY 和 VCPU。

```
"resourceRequirements" : [
  {
    "type": "GPU",
    "value": "number"
  }
]
```

類型：物件陣列

必要：否

`type`

要指派給容器的資源類型。支援的資源包括 GPU MEMORY 和 VCPU。

類型：字串

必要：是，使用 `resourceRequirements` 時。

`value`

為容器預留的指定資源數量。這些值根據指定的 `type` 而有所差異。

type="GPU"

為容器保留的記實體 GPU 數量。為工作中所有容器保留的 GPU 數目不得超過啟動工作所在運算資源上的可用 GPU 數目。

type="MEMORY"

提供給容器使用的記憶體硬性限制 (MiB)。如果您的容器嘗試使用超過此處指定的記憶體，容器便會終止。此參數對應到 Docker Remote API Create a container (建立容器) 一節中的 `Memory`，以及 Docker run 的 `--memory` 選項。您必須為單一工作指定至少 4 MiB 的記憶體。這是必要的，但可以在多個地方為多節點平行 (MNP) 任務指定。必須至少為每個節點指定一次。此參數對應到 Docker Remote API Create a container (建立容器) 一節中的 `Memory`，以及 Docker run 的 `--memory` 選項。

> **Note**
>
> 如果您嘗試透過為特定執行個體類型提供作業盡可能多的記憶體來最大化資源使用率，請參閱運算資源記憶體管理。

對於在 Fargate 資源上執行的工作，則`value`必須符合其中一個支援的值。此外，這些`VCPU`值必須是該記憶體值支援的其中一個值。

VCPU	MEMORY
0.25 vCPU	512、1024 及 2048 千 MiB 位
0.5 vCPU	1024-4096 千兆位單位 (以 1024 千兆位元為單位遞增 MiB
1 vCPU	以 1024 千兆位為單位遞增的 2048 至 8192 千 MiB 位
2 vCPU	4096-16384 千兆位單位 (以 1024 千兆 B 為單位遞增 MiB
4 vCPU	以 1024 MiB 為單位遞增的 8192-30720 MiB 兆位
8 vCPU	16384-61440 MIB (以 4096 千兆 B 為單位遞增 MiB
16 vCPU	32768-122880 千兆位單位 (以 8192 千兆位單位為單位遞增 MiB

type="VCPU"

為任務保留的 vCPU 數量。此參數對應到 Docker Remote API Create a container (建立容器) 一節中的 `CpuShares`，以及 Docker run 的 `--cpu-shares` 選項。每個 vCPU 相當於 1,024 個 CPU 共用。對於在 EC2 資源上執行的工作，您必須至少指定一個 vCPU。這是必要的，但可以在多個地方指定。必須至少為每個節點指定一次。

對於在 Fargate 資源上執行的工作，`value`必須符合其中一個支援的`MEMORY`值，而且值必須是該 VCPU 值支援的其中一個值。支援的值為 0.25、0.5、1、2、4、8 和 16。

Fargate 隨需 vCPU 資源計數配額的預設值為 6 個 vCPU。如需有關 Fargate 配額的詳細資訊，請參閱AWS . Amazon Web Services 一般參考

類型：字串

必要：是，使用 resourceRequirements 時。

secrets

公開為環境變數之工作的密碼。如需詳細資訊，請參閱 指定敏感資料。

```
"secrets": [
    {
      "name": "secretName1",
      "valueFrom": "secretArn1"
    },
    {
      "name": "secretName2",
      "valueFrom": "secretArn2"
    }
    ...
]
```

類型：物件陣列

必要：否

name

包含密碼的環境變數名稱。

類型：字串

必要：是，使用 secrets 時。

valueFrom

公開給容器的秘密。支援的值是秘 Secrets Manager 碼的完整 Amazon 資源名稱 (ARN)，或 SSM 參數存放區中參數的完整 ARN。

> **Note**
>
> 如果 SSM 參數存放區參數與您要啟動的工作位於相同 AWS 區域 的位置，則您可以使用完整 ARN 或參數的名稱。如果參數存在於不同區域，則必須指定完整 ARN。

類型：字串

必要：是，使用 secrets 時。

ulimits

容器中要設定的 ulimits 值的清單。此參數會映射到 Docker Remote API 的建立容器區段中的 Ulimits 以及 docker run 的 --ulimit 選項。

```
"ulimits": [
  {
    "name": string,
    "softLimit": integer,
    "hardLimit": integer
  }
  ...
]
```

類型：物件陣列

必要：否

name

ulimit 的 type。

類型：字串

必要：是，使用 ulimits 時。

hardLimit

ulimit 類型的硬性限制。

類型：整數

必要：是，使用 ulimits 時。

softLimit

ulimit 類型的軟性限制。

類型：整數

必要：是，使用 ulimits 時。

`user`

要在容器內使用的使用者名稱。此參數會映射到 Docker Remote API 的建立容器區段中的 `User` 以及 docker run 的 `--user` 選項。

```
"user": "string"
```

類型：字串

必要：否

`vcpus`

此參數已過時，請*resourceRequirements*改用。

為容器保留的 vCPU 數量。

作為如何使用的範例`resourceRequirements`，如果您的工作定義包含類似下列的行：

```
"containerProperties": {
  "vcpus": 2
}
```

使用的等效`resourceRequirements`行如下。

```
"containerProperties": {
  "resourceRequirements": [
    {
      "type": "VCPU",
      "value": "2"
    }
  ]
}
```

類型：整數

必要：是

`volumes`

註冊任務定義時，您可指定磁碟區清單，那些磁碟區會傳送到容器執行個體上的 Docker 協助程式。容器屬性允許使用以下參數：

```
"volumes": [
  {
    "name": "string",
    "host": {
      "sourcePath": "string"
    },
    "efsVolumeConfiguration": {
      "authorizationConfig": {
        "accessPointId": "string",
        "iam": "string"
      },
      "fileSystemId": "string",
      "rootDirectory": "string",
      "transitEncryption": "string",
      "transitEncryptionPort": number
    }
  }
]
```

name

磁碟區名稱。可以包含最多可達 255 個字元 (大小寫)、數字、連字號和底線。此名稱是參考容器定義 sourceVolume 中的 mountPoints 參數。

類型：字串

必要：否

host

host 參數內容決定資料磁碟區是否在主機容器執行個體和儲存位置中保留。如果 host 參數是空的，則 Docker 協助程式會為您的資料磁碟區指派主機路徑。但是，在與其關聯的容器停止運行之後，數據不能保證持續存在。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

類型：物件

必要：否

`sourcePath`

提供給容器的主機容器執行個體上的路徑。如果此參數是空的，則 Docker 常駐程式會為您指派主機路徑。

如果 host 參數包含 sourcePath 檔案位置，資料磁碟區將保留在主機容器執行個體上的指定位置，直到您手動將其刪除為止。如果 sourcePath 值不存在於主機容器執行個體上，Docker 常駐程式將建立該值。如果位置存在，將匯出來源路徑資料夾的內容。

類型：字串

必要：否

`efsVolumeConfiguration`

當您使用適用於任務儲存體的 Amazon Elastic File System 檔案系統時，會指定此參數。如需詳細資訊，請參閱 Amazon EFS 磁碟區。

類型：物件

必要：否

`authorizationConfig`

Amazon EFS 檔案系統的授權組態詳細資訊。

類型：字串

必要：否

`accessPointId`

要使用的 Amazon EFS 存取點 ID。如果指定存取點，則EFSVolumeConfiguration必須省略或將中指定的根目錄值設定為/。這會強制執行 EFS 存取點上設定的路徑。如果使用存取點，則必須在 EFSVolumeConfiguration 中啟用傳輸加密。如需詳細資訊，請參閱《Amazon Elastic File System 使用者指南》中的使用 Amazon EFS 存取點。

類型：字串

必要：否

`iam`

決定是否在掛接 Amazon EFS 檔案系統時使用 AWS Batch 任務定義中定義的任務 IAM 角色。如果已啟用，必須在 EFSVolumeConfiguration 中啟用傳輸加密。如果省略此

參數，系統會使用 DISABLED 的預設值。如需詳細資訊，請參閱 使用 Amazon EFS 存取點。

類型：字串

有效值：ENABLED | DISABLED

必要：否

`fileSystemId`

要使用的 Amazon EFS 檔案系統識別碼。

類型：字串

必要：否

`rootDirectory`

在 Amazon EFS 檔案系統中的目錄，其將掛載作為主機內的根目錄。如果省略此參數，使用 Amazon EFS 磁碟區的根目錄。如果您指定/，它與省略此參數具有相同的效果。長度上限為 4,096 個字元。

> **Important**
>
> 如果在中指定 EFS 存取點`authorizationConfig`，則必須省略根目錄參數或將其設定為/。這會強制執行 Amazon EFS 存取點上設定的路徑。

類型：字串

必要：否

`transitEncryption`

確定是否要對 Amazon ECS 主機和 Amazon EFS 伺服器之間 Amazon EFS 傳輸中的資料啟用加密功能。若使用 Amazon EFS IAM 授權，則必須啟用傳輸加密。如果省略此參數，系統會使用 DISABLED 的預設值。如需詳細資訊，請參閱《Amazon Elastic File System 使用者指南》中的加密傳輸中的資料。

類型：字串

有效值：ENABLED | DISABLED

必要：否

`transitEncryptionPort`

在 Amazon ECS 主機和 Amazon EFS 伺服器之間傳送加密資料時所使用的連接埠。如果您未指定傳輸加密連接埠，它會使用 Amazon EFS 掛載協助程式使用的連接埠選擇策略。該值必須介於 0 到 65,535 之間。如需詳細資訊，請參閱《Amazon Elastic File System 使用者指南》中的 EFS 掛載協助程式。

類型：整數

必要：否

Amazon EKS 屬性

有各種 Amazon ECS 型任務特定屬性的物件。對於以 Amazon ECS 為基礎的任務定義，則不得指定此選項。

`podProperties`

工作之Kubernetes網繭資源的內容。

類型：EksPod屬性物件

必要：否

`containers`

Amazon EKS Pod 上所使用容器的屬性。

類型：EksContainer 物件

必要：否

`args`

進入點的引數陣列。如果未指定，系統會使用容器映像的 CMD。這對應於中網繭之入口點部分中的`args`成員。Kubernetes環境變數參考使用容器的環境擴展。

如果沒有參考的環境變數，不會變更命令中的參考。例如，如果參考為 "$(NAME1)"，且沒有 NAME1 環境變數，命令字串會保持 "$(NAME1)"。$$ 會替換為 $，且產生的字串不會擴展。例如，$$(VAR_NAME) 會以 $(VAR_NAME) 傳遞，無論是否有 VAR_NAME 環境變數。如需詳細資訊，請參閱 Dockerfile 參考中的 CMD 和文件中的網繭定義命令和引數。Kubernetes

類型：字串陣列

必要：否

command

容器的進入點。這不是在 Shell 中執行。如果未指定，系統會使用容器映像的 ENTRYPOINT。環境變數參考使用容器的環境擴展。

如果沒有參考的環境變數，不會變更命令中的參考。例如，如果參考為 "$(NAME1)"，且沒有 NAME1 環境變數，命令字串會保持 "$(NAME1)"。$$ 會替換為 $，且產生的字串不會擴展。例如，$$(VAR_NAME) 會以 $(VAR_NAME) 傳遞，無論是否有 VAR_NAME 環境變數。無法更新進入點。如需詳細資訊，請參閱 Docker 檔案參考資料中的 ENTRYPOINT 和文件中的為容器和入口點定義命令和引數。Kubernetes

類型：字串陣列

必要：否

env

傳遞至容器的環境變數。

> **Note**
>
> 環境變數不得以 "AWS_BATCH" 開頭。此命名慣例保留給 AWS Batch 設定的變數。

類型：EksContainerEnvironmentVariable 物件陣列

必要：否

name

環境變數的名稱。

類型：字串

必要：是

value

環境變數的值。

類型：字串

必要：否

`image`

用來啟動容器的 Docker 映像檔。

類型：字串

必要：是

`imagePullPolicy`

容器的映像提取政策。支援的值為 `Always`、`IfNotPresent` 和 `Never`。此參數預設為 `IfNotPresent`。但如果指定 `:latest` 標籤，預設為 `Always`。如需詳細資訊，請參閱Kubernetes文件中的更新影像。

類型：字串

必要：否

`name`

容器的名稱。如果未指定名稱，系統會使用預設名稱 "`Default`"。Pod 中的每個容器都必須有唯一名稱。

類型：字串

必要：否

`resources`

指派給容器的資源類型和數量。支援的資源包括 `memory` `cpu` 和 `nvidia.com/gpu`。如需詳細資訊，請參閱文件中的網繭和容器的Kubernetes資源管理。

類型：EksContainerResourceRequirements 物件

必要：否

`limits`

為容器預留的資源類型和數量。這些值根據指定的 `name` 而有所差異。可以使用 `limits` 或 `requests` 物件請求資源。

memory

容器的記憶體硬性限制 (以 MiB 為單位)，使用整數，具有 "Mi" 字尾。如果您的容器嘗試使用超過指定的記憶體，容器便會終止。您必須為任務指定至少 4 MiB 的記憶體。可以在 `limits`、`requests` 或兩者中指定 `memory`。如果同時在這兩個位置指定 `memory`，則 `limits` 中指定的值必須等於 `requests` 中指定的值。

> **Note**
>
> 若要將資源使用率最大化，請為您正在使用的特定執行個體類型的任務，提供盡可能多的記憶體。如要瞭解如何作業，請參閱運算資源記憶體管理。

cpu

為容器預留的 CPU 數量。值必須是 `0.25` 的偶數倍數。可以在 `limits`、`requests` 或兩者中指定 `cpu`。如果同時在這兩個位置指定 `cpu`，則 `limits` 中指定的值至少須與 `requests` 中指定的值一樣大。

nvidia.com/gpu

為容器預留的 GPU 數量。值必須為整數。可以在 `limits`、`requests` 或兩者中指定 `memory`。如果同時在這兩個位置指定 `memory`，則 `limits` 中指定的值必須等於 `requests` 中指定的值。

類型：字串到字串映射

值長度限制：長度下限為 1。長度上限為 256。

必要：否

`requests`

為容器請求的資源類型和數量。這些值根據指定的 `name` 而有所差異。可以使用 `limits` 或 `requests` 物件請求資源。

memory

容器的記憶體硬性限制 (以 MiB 為單位)，使用整數，具有 "Mi" 字尾。如果您的容器嘗試使用超過指定的記憶體，容器便會終止。您必須為任務指定至少 4 MiB 的記憶體。可以在 `limits`、`requests` 或兩者中指定 `memory`。如果同時在兩者中指定 `memory`，則 `limits` 中指定的值必須等於 `requests` 中指定的值。

> **Note**
>
> 如果您嘗試透過為特定執行個體類型提供作業盡可能多的記憶體來最大化資源使用率，請參閱運算資源記憶體管理。

cpu

為容器預留的 CPU 數量。值必須是 `0.25` 的偶數倍數。可以在 `limits`、`requests` 或兩者中指定 `cpu`。如果同時在兩者中指定 `cpu`，則 `limits` 中指定的值至少須與 `requests` 中指定的值一樣大。

nvidia.com/gpu

為容器預留的 GPU 數量。值必須為整數。可以在 `limits`、`requests` 或兩者中指定 `nvidia.com/gpu`。如果同時在兩者中指定 `nvidia.com/gpu`，則 `limits` 中指定的值必須等於 `requests` 中指定的值。

類型：字串到字串映射

值長度限制：長度下限為 1。長度上限為 256。

必要：否

`securityContext`

任務的安全性內容。如需詳細資訊，請參閱Kubernetes文件中的設定網繭或容器的安全性內容。

類型：EksContainerSecurityContext 物件

必要：否

`privileged`

此參數為 `true` 時，容器便會取得主機容器執行個體的更高許可。權限層級與`root`使用者權限類似。預設值為 `false`。此參數對應至Kubernetes說明文件中「授權」網繭安全性`privileged`原則中的原則。

類型：布林值

必要：否

`readOnlyRootFilesystem`

此參數為 `true` 時，容器會取得根檔案系統的唯讀存取權。預設值為 `false`。此參數對應至Kubernetes說明文件中的磁碟區和檔案系統網繭安全性ReadOnlyRootFilesystem原則中的原則。

類型：布林值

必要：否

`runAsGroup`

指定此參數時，容器會以指定的群組 ID (`gid`) 執行。如果未指定此參數，預設值為映像中繼資料中指定的群組。此參數對應至RunAsGroupKubernetes說明文件中的使用者和群組網繭安全性MustRunAs原則中的原則。

類型：Long

必要：否

`runAsNonRoot`

指定此參數時，容器會以 `uid` 非 0 的使用者身分執行。如果未指定此參數，系統會強制執行此規則。此參數對應至RunAsUserKubernetes說明文件中的使用者和群組網繭安全性MustRunAsNonRoot原則中的原則。

類型：Long

必要：否

`runAsUser`

指定此參數時，容器會以指定的使用者 ID (`uid`) 執行。如果未指定此參數，預設值為映像中繼資料中指定的使用者。此參數對應至RunAsUserKubernetes說明文件中的使用者和群組網繭安全性MustRanAs原則中的原則。

類型：Long

必要：否

`volumeMounts`

磁碟區會針對適用於 Amazon EKS 任務的容器掛載。如需有關磁碟區和磁碟區掛接的詳細資訊Kubernetes，請參閱Kubernetes文件中的磁碟區。

類型：EksContainerVolumeMount 物件陣列

必要：否

mountPath

掛載磁碟區之容器上的路徑。

類型：字串

必要：否

name

掛載的磁碟區名稱。這必須符合 Pod 中任一磁碟區的名稱。

類型：字串

必要：否

readOnly

如果此數值為 true，容器擁有磁碟區的唯讀存取權。否則，容器可以寫入磁碟區。預設值為 false。

類型：布林值

必要：否

dnsPolicy

Pod 的 DNS 政策。預設值為 ClusterFirst。如果未指定 hostNetwork 參數，預設值為 ClusterFirstWithHostNet。ClusterFirst 指示任何與設定之叢集網域字尾不相符的 DNS 查詢，都會轉寄至繼承自節點的上游名稱伺服器。如果在「定RegisterJob義 API」作業dnsPolicy中未指定任何值，則「定DescribeJob義」或 DescribeJobsAPI dnsPolicy 作業都不會傳回任何值。視 hostNetwork 參數的值而定，Pod 規格設定會包含 ClusterFirst 或 ClusterFirstWithHostNet。如需詳細資訊，請參閱Kubernetes文件中的 Pod 的 DNS 原則。

有效值：Default | ClusterFirst | ClusterFirstWithHostNet

類型：字串

必要：否

`hostNetwork`

指示 Pod 是否使用主機的網路 IP 地址。預設值為 `true`。設定此值以`false`啟用網Kubernetes繭網路模型。大部分的 AWS Batch 工作負載僅限輸出，不需要針對內送連線的每個網繭配置 IP 額外負荷。如需詳細資訊，請參閱Kubernetes說明文件中的主機命名空間和網繭網路。

類型：布林值

必要：否

`serviceAccountName`

用來執行 Pod 的服務帳戶名稱。如需詳細資訊，請參閱 KubernetesAmazon EKS 使用者指南中的Kubernetes服務帳戶和設定服務帳戶以擔任 IAM 角色和Kubernetes說明文件中的網繭設定服務帳戶。

類型：字串

必要：否

`volumes`

為使用 Amazon EKS 資源的任務定義指定磁碟區。

類型：EksVolume 物件陣列

必要：否

空白目錄

指定Kubernetes`emptyDir`磁碟區的組態。將 Pod 指派給節點時，會先建立 `emptyDir` 磁碟區。只要該網繭在該節點上執行，它就會存在。`emptyDir` 磁碟區最初是空的。Pod 中的所有容器都可以讀取和寫入 `emptyDir` 磁碟區中的檔案。但 `emptyDir` 磁碟區可以掛載在每個容器中相同或不同路徑上。因任何原因將 Pod 從節點移除時，系統會永久刪除 `emptyDir` 中的資料。如需詳細資訊，請參閱文件中的 KubernetesemptyDir。

類型：EksEmpty目錄對象

必要：否

中型

存放磁碟區的媒體。預設值為空白字串，這會使用節點的儲存空間。

""

(預設) 使用節點的磁碟儲存空間。

"Memory"

使用節點 RAM 支援的 `tmpfs` 磁碟區。節點重新開機時，磁碟區的內容會遺失，且磁碟區上的任何儲存空間都會計入容器的記憶體限制。

類型：字串

必要：否

尺寸大小

磁碟區的大小上限。預設未定義大小上限。

類型：字串

長度限制：長度下限為 1。長度上限為 256。

必要：否

主機路徑

指定Kubernetes`hostPath`磁碟區的組態。`hostPath` 磁碟區會將現有檔案或目錄，從主機節點的檔案系統掛載到您的 Pod。如需詳細資訊，請參閱Kubernetes文件中的主機路徑。

類型：EksHost路徑物件

必要：否

path

要掛載至 Pod 上容器的主機檔案或目錄的路徑。

類型：字串

必要：否

name

磁碟區名稱。此名稱必須可當作 DNS 子網域名稱。如需詳細資訊，請參閱Kubernetes文件中的 DNS 子網域名稱。

類型：字串

必要：是

秘密

指定Kubernetes`secret`磁碟區的組態。如需詳細資訊，請參閱Kubernetes文件中的 secret。

類型：EksSecret 物件

必要：否

選擇性

指定是否必須定義密鑰或密鑰的密鑰。

類型：布林值

必要：否

秘密名稱

秘密的名稱。此名稱必須可當作 DNS 子網域名稱。如需詳細資訊，請參閱Kubernetes文件中的 DNS 子網域名稱。

類型：字串

必要：是

平台功能

`platformCapabilities`

工作定義所需的平台功能。如果未指定任何值，則預設為 EC2。針對在 Fargate 資源上執行的工作，會指`FARGATE`定。

> **Note**
> 如果任務在 Amazon EKS 資源上執行，則不得指定`platformCapabilities`。

類型：字串

有效值：EC2 | FARGATE

必要：否

傳播標籤

`propagateTags`

指定是否要將標籤從任務或任務定義傳播到對應的 Amazon ECS 任務。如果沒有指定值，則不會傳播標籤。標籤只能在建立工作時傳播至工作。對於具有相同名稱的標籤，任務標籤優先於任務定義標籤。如果工作和工作定義的合併標籤總數超過 50 個，則該工作會移至狀`FAILED`態。

> **Note**
>
> 如果任務在 Amazon EKS 資源上執行，則不得指定`propagateTags`。

類型：布林值

必要：否

節點屬性

`nodeProperties`

註冊多節點 parallel 工作定義時，必須指定節點屬性清單。這些節點屬性會定義工作中要使用的節點數目、主要節點索引，以及要使用的不同節點範圍。如果工作在 Fargate 資源上執行，則無法指定`nodeProperties`。請改用 `containerProperties`。任務定義允許使用以下的節點屬性。如需詳細資訊，請參閱 多節點 parallel 工作。

> **Note**
>
> 如果任務在 Amazon EKS 資源上執行，則不得指定`nodeProperties`。

類型：NodeProperties 物件

必要：否

`mainNode`

為多節點平行任務指定主要節點的節點索引。此節點索引值必須小於節點數目。

類型：整數

必要：是

`numNodes`

與多節點平行任務關聯的節點數量。

類型：整數

必要：是

`nodeRangeProperties`

與多節點平行任務關聯的節點範圍和其屬性清單。

> **Note**
>
> 節點群組是共用相同容器屬性的相同工作節點群組。您最多可以 AWS Batch 為每個工作指定五個不同的節點群組。

類型：NodeRange屬性物件的陣列

必要：是

`targetNodes`

使用節點索引值的節點範圍。`0:3` 的範圍表示節點具有 `0` 到 `3` 的索引值。如果省略起始範圍值 (`:n`)，則使用 `0` 來開始範圍。如果省略了結束範圍值 (`n:`)，則會使用最高可能的節點索引來結束範圍。您的累積節點範圍必須將所有節點納入考量 (`0:n`)。您可以巢狀化節點範圍，例如`0:10`和`4:5`。在這種情況下，`4:5`範圍屬性會覆寫`0:10`屬性。

類型：字串

必要：否

`container`

節點範圍的容器詳細資訊。如需詳細資訊，請參閱 容器屬性。

類型：ContainerProperties 物件

必要：否

重試策略

`retryStrategy`

註冊任務定義，您可以選擇性指定任務失敗後要使用的重試策略，隨此任務定義提交。在SubmitJob作業期間指定的任何重試策略都會覆寫此處定義的重試策略。根據預設，每個任務將嘗試一次。如果您指定多次嘗試，則會在工作失敗時重試。失敗嘗試的範例包括工作傳回非零結束代碼或容器執行個體終止。如需詳細資訊，請參閱 自動化工作重試。

類型：RetryStrategy 物件

必要：否

`attempts`

將任務移至 RUNNABLE 狀態的次數。您可以指定嘗試 1 至 10 次。如果 `attempts` 超過 1 次，任務失敗後將重試該次數，直到其狀態移至 RUNNABLE。

```
"attempts": integer
```

類型：整數

必要：否

`evaluateOnExit`

最多 5 個物件所組成的陣列，可指定重試或失敗工作的條件。如果指定此參數，則也必須指定 `attempts` 參數。如果`evaluateOnExit`已指定但沒有符合任何項目，則會重試工作。

```
"evaluateOnExit": [
   {
      "action": "string",
      "onExitCode": "string",
      "onReason": "string",
      "onStatusReason": "string"
   }
]
```

類型：EvaluateOn結束物件的陣列

必要：否

action

指定符合所有指定條件 (onStatusReason onReason 和 onExitCode) 時要採取的動作。這些值不區分大小寫。

類型：字串

必要：是

有效值：RETRY | EXIT

onExitCode

包含一個 glob 模式，以與工作返回ExitCode的十進制表示相匹配。模式的長度上限為 512 個字元。它只能包含數字。不能包含字母或特殊字元。可以選擇以星號 (*) 結束，以便只有字串的開頭需要完全相符。

類型：字串

必要：否

onReason

包含一個 glob 模式以與作Reason業返回的匹配。模式的長度上限為 512 個字元。它可以包含字母、數字、句點 (.)、冒號 (:) 和空格 (空格、製表符)。可以選擇以星號 (*) 結束，以便只有字串的開頭需要完全相符。

類型：字串

必要：否

onStatusReason

包含一個 glob 模式以與作StatusReason業返回的匹配。模式的長度上限為 512 個字元。它可以包含字母、數字、句點 (.)、冒號 (:) 和空格 (空格、製表符)。可以選擇以星號 (*) 結束，以便只有字串的開頭需要完全相符。

類型：字串

必要：否

排程優先權

`schedulingPriority`

使用此工作定義提交之工作的排程優先順序。這只會影響具有公平共用政策的任務佇列中的任務。排程優先順序較高的任務會排在排程優先順序較低的任務之前。

支援的最小值為 0，支援的最大值為 9999。

類型：整數

必要：否

標籤

`tags`

要與工作定義關聯的索引鍵值配對標籤。如需詳細資訊，請參閱 標記您的 AWS Batch 資源。

類型：字串到字串映射

必要：否

逾時

`timeout`

您可以設定工作的逾時持續時間，以便在工作執行時間超過該時間時 AWS Batch 終止工作。如需詳細資訊，請參閱 Job 逾時。如果工作因為逾時而終止，則不會重試。在SubmitJob作業期間指定的任何逾時設定都會覆寫此處定義的逾時設定。如需詳細資訊，請參閱 Job 逾時。

類型：JobTimeout 物件

必要：否

`attemptDurationSeconds`

AWS Batch 終止未完成工作後的持續時間 (以作業嘗試的時間`startedAt`戳記計算)。逾時最小值為 60 秒。

若為陣列任務，逾時會套用至子任務，而不是父陣列任務。

若為多節點平行 (MNP) 任務，逾時會套用至整個工作，而非個別節點。

類型：整數

必要：否

使用建立工作定義 EcsProperties

使用 AWS Batch 工作定義時 EcsProperties，您可以在不同的容器中建立硬體、感測器、3D 環境和其他模擬的模擬。您可以使用此功能以邏輯方式組織工作負載元件，並將它們與主應用程式分開。此功能可與 AWS Batch Amazon Elastic Container Service (Amazon ECS)，Amazon Elastic Kubernetes Service（Amazon EKS）和一起使用。 AWS Fargate

ContainerProperties與EcsProperties工作定義

您可以根據使用案例指示選擇使用ContainerProperties或EcsProperties工作定義。在高層級中，執行EcsProperties的 AWS Batch 工作類似於使用ContainerProperties.

使ContainerProperties用的舊式工作定義結構仍受支援。如果您目前有使用此結構的工作流程，您可以繼續執行它們。

主要的差異在於，有一個新的物件加入到工作定義中，以適應EcsProperties基於定義。

例如，ContainerProperties在 Amazon ECS 和 Fargate 上使用的任務定義具有以下結構：

```
{
   "containerProperties": {
     ...
     "image": "my_ecr_image1",
     ...
  },
...
}
```

EcsProperties在 Amazon ECS 和 Fargate 上使用的任務定義具有以下結構：

```
{
  "ecsProperties": {
    "taskProperties": [{
      "containers": [
        {
          ...
          "image": "my_ecr_image1",
```

```
      ...
    },
    {
      ...
      "image": "my_ecr_image2",
      ...
    },
```

AWS Batch API 的一般變更

以下內容進一步概述了使用EcsProperties和 EcsProperties API 資料類型時的一些主要差異：

- 中使用的許多參數都ContainerProperties會顯示在中TaskContainerProperties。一些範例包括commandimage、privileged、、secrets、和users。它們都可以在其中找到TaskContainerProperties。
- 某些TaskContainerProperties參數在傳統結構中沒有功能等價物。一些範例包括dependsOnessential、name、、ipcMode、和pidMode。如需詳細資訊，請參閱EcsTaskDetails和TaskContainerProperties。

  此外，某些ContainerProperties參數在EcsProperties結構中沒有對等項或應用程式。中taskProperties，container已被取代為，以containers便新物件最多可以接受十個元素。如需詳細資訊，請參閱容器內容和:容器RegisterJobDefinition。EcsTaskProperties
- taskRoleArn在功能上等同於jobRoleArn. 如需詳細資訊，請參閱 EcsTaskProperties：taskRoleArn和 ContainerProperties：jobRoleArn。
- 您可以在EcsProperties結構中包含一 (1) 到十 (10) 個容器。如需詳細資訊，請參閱：EcsTaskProperties容器。
- taskProperties和 instanceTypes 對象是數組，但目前只接受一個元素。例如，：工作屬性和:instanceTypes EcsProperties。NodeRangeProperty

Amazon ECS 的多容器任務定義

為了適應 Amazon ECS 的多容器結構，部分 API 資料類型有所不同。例如

- ecsProperties與單一容器定義containerProperties中的層級相同。如需詳細資訊，請參閱 AWS Batch API 參考指南EcsProperties中的。
- taskProperties包含針對 Amazon ECS 任務定義的屬性。如需詳細資訊，請參閱 AWS Batch API 參考指南EcsProperties中的。

- containers包含與單一容器定義containerProperties中的類似資訊。主要區別在於containers允許您定義多達十個容器。如需詳細資訊，請參閱《API AWS Batch 參考指南》TaskProperties中的 EC：容器。
- essential參數指示容器如何影響工作。所有客戶都必須成功完成（退出為 0），以便工作進行。如果標記為 essentail 的容器失敗 (以非 0 結束)，則工作會失敗。

  預設值為，true且至少必須將一個容器標記為essential。如需詳細資訊，請參閱 essential API 參考指南中的「AWS Batch 」。
- 透過dependsOn參數，您可以定義容器相依性的清單。如需詳細資訊，請參閱 dependsOn API 參考指南中的「AWS Batch 」。

> **Note**
>
> dependsOn清單的複雜性和相關聯的容器執行階段可能會影響工作的開始時間。如果相依性需要很長時間才能執行，則工作將保持STARTING狀態，直到完成為止。

如需有關ecsProperties和結構的詳細資訊，RegisterJobDefinition請參閱 ECs Properties 的要求語法。

Amazon EKS 的多容器任務定義

為了適應 Amazon EKS 的多容器結構，部分 API 資料類型有所不同。例如

- name是容器的唯一識別碼。單一容器不需要此物件，但在網繭中定義多個容器時則需要此物件。如果name未針對單一容器定義，則會套用預設名稱。default
- initContainers在eksPodProperties資料類型中定義。Thery 在應用程序容器之前運行，始終運行到完成，並且必須在下一個容器啟動之前成功完成。

  這些容器會向 Amazon EKS 連接器代理程式註冊，並在 Amazon Elastic Kubernetes Service 後端資料存放區中保留註冊資訊。initContainers物件最多可接受十 (10) 個元素。如需詳細資訊，請參閱Kubernetes文件中的初始化容器。

> **Note**
>
> initContainers物件可能會影響工作的開始時間。如果initContainers需要很長時間才能執行，工作將保持STARTING狀態，直到完成為止。

- shareProcessNamespace指出網繭中的容器是否可以共用相同的處理序命名空間。預設值為false。將此設定為可true讓容器在位於相同網繭中的其他容器中查看程序並發出訊號。
- 每個容器都有重要性。所有容器都必須成功完成 (以 0 結束)，工作才能成功。如果一個容器失敗 (以 0 以外的方式結束)，則工作會失敗。

如需有關eksProperties和結構的詳細資訊，RegisterJobDefinition請參閱 EKs Properties 的要求語法。

AWS Batch 工作情境使用 EcsProperties

為了說明如 AWS Batch 何根據您的需求來構建使用EcsProperties的工作定義，本主題介紹下列RegisterJobDefinition承載。您可以將這些範例複製到檔案中，根據需要自訂它們，然後使用 AWS Command Line Interface (AWS CLI) 來呼叫RegisterJobDefinition。

AWS Batch Amazon 彈性容器服務在 Amazon 彈性計算雲上的任務

```
{
    "jobDefinitionName": "multicontainer-ecs-ec2",
    "type": "container",
    "ecsProperties": {
        "taskProperties": [
          {
            "containers": [
              {
                "name": "c1",
                "essential": false,
                "command": [
                  "echo",
                  "hello world"
                ],
                "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                "resourceRequirements": [
                  {
                    "type": "VCPU",
                    "value": "2"
                  },
                  {
                    "type": "MEMORY",
                    "value": "4096"
                  }
                ]
```

```
              },
              {
                "name": "c2",
                "essential": true,
                "command": [
                  "echo",
                  "hello world"
                ],
                "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                "resourceRequirements": [
                  {
                    "type": "VCPU",
                    "value": "6"
                  },
                  {
                    "type": "MEMORY",
                    "value": "12288"
                  }
                ]
              }
            ]
          }
        ]
  }
}
```

AWS Batch Amazon ECS 的工作 AWS Fargate

```
{
    "jobDefinitionName": "multicontainer-ecs-fargate",
    "type": "container",
    "platformCapabilities": [
        "FARGATE"
    ],
    "ecsProperties": {
        "taskProperties": [
          {
            "containers": [
              {
                "name": "c1",
                "command": [
                  "echo",
                  "hello world"
```

```
                ],
                "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                "resourceRequirements": [
                  {
                    "type": "VCPU",
                    "value": "2"
                  },
                  {
                    "type": "MEMORY",
                    "value": "4096"
                  }
                ]
              },
              {
                "name": "c2",
                "essential": true,
                "command": [
                  "echo",
                  "hello world"
                ],
                "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                "resourceRequirements": [
                  {
                    "type": "VCPU",
                    "value": "6"
                  },
                  {
                    "type": "MEMORY",
                    "value": "12288"
                  }
                ]
              }
            ],
            "executionRoleArn": "arn:aws:iam::1112223333:role/ecsTaskExecutionRole"
          }
        ]
  }
}
```

AWS Batch Amazon Elastic Kubernetes Service 工作

```
{
  "jobDefinitionName": "multicontainer-eks",
```

```
  "type": "container",
  "eksProperties": {
    "podProperties": {
      "shareProcessNamespace": true,
      "initContainers": [
        {
          "name": "init-container",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "echo"
          ],
          "args": [
            "hello world"
          ],
          "resources": {
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        },
        {
          "name": "init-container-2",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "echo",
            "my second init container"
          ],
          "resources": {
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        }
      ],
      "containers": [
        {
          "name": "c1",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "echo world"
         ],
          "resources": {
```

```
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        },
        {
          "name": "sleep-container",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "sleep",
            "20"
          ],
          "resources": {
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        }
      ]
    }
  }
}
```

多節點 parallel (MNP) AWS Batch 工作，每個節點具有多個容器

```
{
  "jobDefinitionName": "multicontainer-mnp",
  "type": "multinode",
  "nodeProperties": {
    "numNodes": 6,
    "mainNode": 0,
    "nodeRangeProperties": [
      {
        "targetNodes": "0:5",
        "ecsProperties": {
          "taskProperties": [
            {
              "containers": [
                {
                  "name": "range05-c1",
                  "command": [
```

```
                    "echo",
                    "hello world"
                  ],
                  "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                  "resourceRequirements": [
                    {
                      "type": "VCPU",
                      "value": "2"
                    },
                    {
                      "type": "MEMORY",
                      "value": "4096"
                    }
                  ]
                },
                {
                  "name": "range05-c2",
                  "command": [
                    "echo",
                    "hello world"
                  ],
                  "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                  "resourceRequirements": [
                    {
                      "type": "VCPU",
                      "value": "2"
                    },
                    {
                      "type": "MEMORY",
                      "value": "4096"
                    }
                  ]
                }
              ]
            }
          ]
        }
      }
    ]
  }
}
```

使用 awslogs 日誌驅動程式

依預設，AWS Batch啟用記awslogs錄驅動程式將記錄檔資訊傳送至 CloudWatch 記錄檔。您可以使用此功能，在一個方便的位置檢視容器中的不同記錄，並防止容器記錄佔用容器執行個體上的磁碟空間。本主題可協助您在工作定義中設定awslogs記錄驅動程式。

> **Note**
>
> 在AWS Batch主控台中，您可以在建立工作定義時，在 [記錄設定] 區段中設定記錄驅動程式。awslogs

> **Note**
>
> 工作中容器記錄的資訊類型主要取決於它們的ENTRYPOINT命令。根據預設，如果您在本機執行容器 (STDOUT和 STDERR I/O 串流)，擷取的記錄會顯示您通常會在互動式終端機中看到的命令輸出。日awslogs誌驅動程序只是將這些日誌從 Docker 傳遞到 CloudWatch 日誌。如需 Docker 日誌處理方式 (包括擷取不同檔案資料或串流的替代方法) 的詳細資訊，請參閱 Docker 文件中的檢視容器或服務的日誌。

若要將容器執行個體的系統記錄檔傳送至 CloudWatch Logs，請參閱搭配使用 CloudWatch 記錄 AWS Batch。如需有關 CloudWatch 日誌的詳細資訊，請參閱 Amazon CloudWatch 日誌使用者指南中的監控 CloudWatch 日誌檔和日誌配額。

可用的 awslogs 日誌驅動程式選項

記awslogs錄驅動程式支援工AWS Batch作定義中的下列選項。如需詳細資訊，請參閱 Docker 文件中的CloudWatch 記錄記錄驅動程式。

`awslogs-region`

必要：否

指定記awslogs錄驅動程式應傳送 Docker 記錄檔的區域。根據預設，所使用的區域與工作所使用的區域相同。您可以選擇將不同區域中的工作中的所有記錄傳送至 CloudWatch 記錄中的單一區域。這樣做可以讓它們從一個位置全部可見。或者，您可以按區域將它們分開，以獲得更精細的方法。不過，當您選擇此選項時，請確定指定的記錄群組存在於您指定的 [區域] 中。

`awslogs-group`

必要：選擇性

使用此選`awslogs-group`項，您可以指定記錄驅動程式將其`awslogs`記錄串流傳送到的記錄群組。如果未指定，`aws/batch/job`則使用。

`awslogs-stream-prefix`

必要：選擇性

使用此`awslogs-stream-prefix`選項，您可以將日誌串流與指定的前置詞以及容器所屬任務的 Amazon ECS 任務 ID 建立關聯。AWS Batch如果您使用此選項指定前綴，則日誌串流會使用下列格式：

```
prefix-name/default/ecs-task-id
```

`awslogs-datetime-format`

必要：否

此選項會以 Python `strftime` 格式定義多行開始模式。日誌消息由匹配模式的行以及不匹配模式的任何以下行組成。因此，符合的行是日誌訊息之間的分隔符號。

使用此格式的一個使用案例範例是用於剖析輸出，例如堆疊傾印，在其他情形下這可能會記錄在多個項目中。正確的模式可允許將它擷取在單一項目中。

如需詳細資訊，請參閱awslogs-datetime-format。

如果 `awslogs-datetime-format` 和 `awslogs-multiline-pattern` 都設定，則一律以此選項優先。

> **Note**
>
> 多行記錄會執行常規表達式剖析並比對所有日誌訊息。這可能會對記錄效能造成負面影響。

`awslogs-multiline-pattern`

必要：否

此選項使用規則表達式來定義多行開始模式。日誌消息由匹配模式的行以及不匹配模式的任何以下行組成。因此，匹配的行是日誌消息之間的分隔符。

如需詳細資訊，請參閱 Docker 文件awslogs-multiline-pattern中的。

如果同時設定 awslogs-datetime-format，會忽略此選項。

> **Note**
>
> 多行記錄會執行常規表達式剖析並比對所有日誌訊息。這可能會對記錄效能造成負面影響。

awslogs-create-group

必要：否

指定您是否希望自動建立日誌群組。若未指定此選項，則預設為 false。

> **Warning**
>
> 不建議使用此選項。建議您在每個工作嘗試建立記錄群組時，使用 CloudWatch 記錄CreateLogGroupAPI 動作預先建立記錄群組，從而增加工作失敗的機會。

> **Note**
>
> 您的執行角色的 IAM 政策必須包含logs:CreateLogGroup權限，然後才能嘗試使用awslogs-create-group。

在工作定義中指定記錄組態

依預設，會AWS Batch啟用記awslogs錄驅動程式。本節說明如何自訂工作的awslogs記錄組態。如需詳細資訊，請參閱建立單一節點工作定義 。

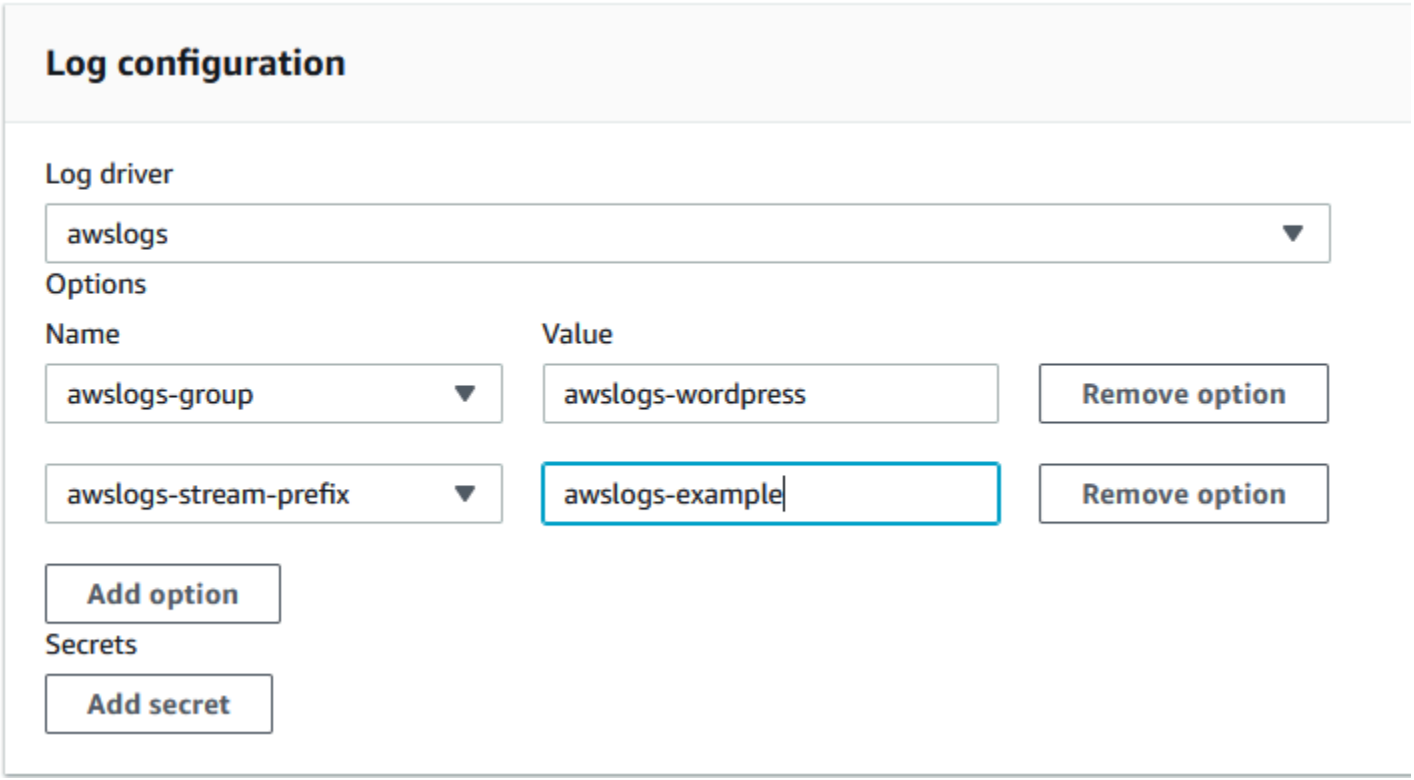
下列記錄組態 JSON 片段具有針對每個工作指定的logConfiguration物件。一種是將日誌發送到名為的日誌組的 WordPress 工作awslogs-wordpress，另一個用於將日誌發送到名為的日誌組的 MySQL 容器awslogs-mysql。兩個容器使用的日誌串流前綴皆為 awslogs-example。

```
"logConfiguration": {
    "logDriver": "awslogs",
    "options": {
        "awslogs-group": "awslogs-wordpress",
        "awslogs-stream-prefix": "awslogs-example"
```

```
    }
}
```

```
"logConfiguration": {
    "logDriver": "awslogs",
    "options": {
        "awslogs-group": "awslogs-mysql",
        "awslogs-stream-prefix": "awslogs-example"
    }
}
```

在AWS Batch主控台中，會指定wordpress工作定義的記錄組態，如下圖所示。



在工作定義記錄組態中使用記awslogs錄驅動程式註冊工作定義之後，您可以提交具有該工作定義的工作，以開始將記錄檔傳送至 CloudWatch 記錄檔。如需詳細資訊，請參閱 提交工作。

指定敏感資料

使用時AWS Batch，您可以將敏感資料儲存在機AWS Secrets Manager密或AWS Systems Manager參數存放區參數中，然後在工作定義中參照這些資料，將敏感資料插入工作。

密碼可以透過下列方式顯示在工作中：

- 若要將敏感資料插入容器做為環境變數，請使用`secrets`工作定義參數。
- 若要參照工作記錄組態中的機密資訊，請使用工`secretOptions`作定義參數。

主題

使用 Secrets Manager 指定敏感資料

使用時 AWS Batch，您可以將敏感資料插入工作，方法是將敏感資料儲存在機 AWS Secrets Manager 密中，然後在工作定義中參照這些資料。Secrets Manager 密碼中儲存的機密資料可以作為環境變數公開給工作，也可以做為記錄組態的一部分公開給工作。

當您將秘密做為環境變數插入時，可以指定 JSON 金鑰或要插入的秘密版本。此程序可協助您控制公開給工作的機密資料。如需有關秘密版本控制的詳細資訊，請參閱《AWS Secrets Manager 使用者指南》中的 AWS Secrets Manager重要術語和概念。

使用 Secrets Manager 指定敏感資料的注意事項

使用 Secrets Manager 指定工作的敏感資料時，應考慮下列事項。

- 若要使用特定 JSON 金鑰或密碼版本插入機密，運算環境中的容器執行個體必須安裝 Amazon ECS 容器代理程式的 1.37.0 版或更新版本。不過，我們建議您使用最新版的容器代理程式。如需有關檢查代理程式版本和更新至最新版本的資訊，請參閱 Amazon 彈性容器服務開發人員指南中的更新 Amazon ECS 容器代理程式。

  若要將密碼的完整內容插入為環境變數，或在記錄組態中插入密碼，您的容器執行個體必須具有容器代理程式的 1.23.0 版或更新版本。
- 僅支援儲存文字資料的密碼，這些密碼是使用 CreateSecretAPI `SecretString` 參數建立的密碼。不支援儲存二進位資料的密碼，這些密碼是使用 CreateSecretAPI `SecretBinary` 參數建立的機密。
- 使用參考 Secrets Manager 密碼的工作定義來擷取工作的機密資料時，如果您同時使用介面虛擬私人雲端端點，則必須為 Secrets Manager 建立介面 VPC 人雲端端點。如需詳細資訊，請參閱《AWS Secrets Manager 使用者指南》中的搭配使用 Secrets Manager 與 VPC 端點。
- 工作初始啟動時，會將機密資料插入您的工作中。如果密碼隨後更新或輪替，工作不會自動接收更新的值。您必須啟動新工作，才能強制服務啟動具有更新密碼值的全新工作。

AWS Batch 密碼所需的 IAM 許可

若要使用此功能，您必須具有執行角色，並在工作定義中參照它。這可讓容器代理程式提取必要的 Secrets Manager 資源。如需詳細資訊，請參閱 AWS Batch 執行 IAM 角色。

若要提供對您所建立之 Secrets Manager 密碼的存取權，請手動將下列權限新增為內嵌原則至執行角色。如需詳細資訊，請參閱 IAM 使用者指南中的新增和移除 IAM 政策。

- `secretsmanager:GetSecretValue` - 如果您要參考 Secrets Manager 秘密，則需要此項目。
- `kms:Decrypt` - 只有在您的秘密使用自訂的 KMS 金鑰而非預設金鑰時，才需要此項目。您的自訂金鑰的 ARN 應該新增為資源。

下列內嵌政策範例新增必要許可。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:<secret_name>",
        "arn:aws:kms:<region>:<aws_account_id>:key/<key_id>"
      ]
    }
  ]
}
```

插入敏感資料作為環境變數

在工作定義中，您可以指定下列項目：

- 包`secrets`含要在工作中設定之環境變數名稱的物件
- Secrets Manager 秘密的 Amazon Resource Name (ARN)
- 包含要呈現給工作之敏感資料的其他參數

下列範例示範了必須為 Secrets Manager 秘密指定的完整語法。

```
arn:aws:secretsmanager:region:aws_account_id:secret:secret-name:json-key:version-
stage:version-id
```

以下部分說明其他參數。這些參數是可選的。但是，如果不使用它們，則必須包含冒號:才能使用默認值。以下提供範例深入說明。

`json-key`

使用您要設為環境變數值的值，來指定金鑰/值對中的金鑰名稱。僅支援 JSON 格式的值。如果您沒有指定 JSON 金鑰，則會使用秘密的完整內容。

`version-stage`

指定您要使用之秘密版本的預備標籤。如果指定了版本預備標籤，就無法指定版本 ID。如果未指定版本階段，則預設會擷取具有 AWSCURRENT 階段標籤的秘密。

預備標籤會用來在不同版本的秘密更新或輪換時加以追蹤。每個版本的秘密都有一或多個預備標籤和 ID。如需詳細資訊，請參閱AWS Secrets Manager 使用指南中的 AWS Secrets Manager 主要術語和概念。

`version-id`

針對您要使用的秘密版本，指定其唯一識別符。如果指定了版本 ID，就無法指定版本預備標籤。如果未指定版本 ID，則預設會擷取具有 AWSCURRENT 階段標籤的秘密。

版本 ID 會用來在不同版本的秘密更新或輪換時加以追蹤。每個版本的秘密都有 ID。如需詳細資訊，請參閱AWS Secrets Manager 使用指南中的 AWS Secrets Manager 主要術語和概念。

容器定義範例

下列範例示範您可以在容器定義中參考 Secrets Manager 秘密的方法。

Example 參考完整秘密

以下是任務定義的程式碼片段，顯示參考 Secrets Manager 秘密全文時的格式。

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
```

```
    }]
  }]
}
```

Example 參考秘密中的特定金鑰

以下顯示來自命令的範例輸出，該get-secret-value命令會顯示密碼的內容，以及與密碼相關聯的版本暫存標籤和版本 ID。

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "VersionId": "871d9eca-18aa-46a9-8785-981dd39ab30c",
    "SecretString": "{\"username1\":\"password1\",\"username2\":\"password2\",
\"username3\":\"password3\"}",
    "VersionStages": [
        "AWSCURRENT"
    ],
    "CreatedDate": 1581968848.921
}
```

在 ARN 結尾指定金鑰名稱，來在容器定義中參考上一個輸出的特定金鑰。

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::"
    }]
  }]
}
```

Example 參考特定秘密版本

以下示範 describe-secret 命令的範例輸出，會顯示秘密的未加密內容，以及所有版本秘密的中繼資料。

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
```

```
    "Description": "Example of a secret containing application authorization data.",
    "RotationEnabled": false,
    "LastChangedDate": 1581968848.926,
    "LastAccessedDate": 1581897600.0,
    "Tags": [],
    "VersionIdsToStages": {
        "871d9eca-18aa-46a9-8785-981dd39ab30c": [
            "AWSCURRENT"
        ],
        "9d4cb84b-ad69-40c0-a0ab-cead36b967e8": [
            "AWSPREVIOUS"
        ]
    }
}
```

在 ARN 結尾指定金鑰名稱，來在容器定義中參考上一個輸出的特定版本預備標籤。

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf::AWSPREVIOUS:"
    }]
  }]
}
```

在 ARN 結尾指定金鑰名稱，來在容器定義中參考上一個輸出的特定版本 ID。

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf::9d4cb84b-ad69-40c0-a0ab-cead36b967e8"
    }]
  }]
}
```

Example 參考秘密的特定金鑰和版本預備標籤

以下說明如何參考秘密中的特定金鑰和特定版本預備標籤。

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1:AWSPREVIOUS:"
    }]
  }]
}
```

若要指定特定的金鑰和版本 ID，請使用下列語法。

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::9d4cb84b-ad69-40c0-a0ab-cead36b967e8"
    }]
  }]
}
```

將敏感資料插入日誌組態

在您的工作定義中，當指定一個時，`logConfiguration`您可以`secretOptions`使用要在容器中設定的記錄驅動程式選項名稱，以及包含要呈現給容器之敏感資料的 Secrets Manager 密碼的完整 ARN 來指定。

以下是工作定義的片段，顯示參考 Secret Manager 密碼時的格式。

```
{
  "containerProperties": [{
    "logConfiguration": [{
      "logDriver": "splunk",
      "options": {
        "splunk-url": "https://cloud.splunk.com:8080"
      },
      "secretOptions": [{
        "name": "splunk-token",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      }]
```

```
    }]
  }]
}
```

建立 AWS Secrets Manager 密碼

您可以使用 Secrets Manager 主控台為您的敏感資料建立秘密。如需詳細資訊，請參閱 《AWS Secrets Manager 使用者指南》中的建立基本秘密。

建立基本秘密

使用 Secrets Manager 為您的敏感資料建立秘密。

1. 開啟位於的 Secrets Manager 主控台https://console.aws.amazon.com/secretsmanager/。
2. 選擇 Store a new secret (存放新機密)。
3. 針對 Select secret type (選取秘密類型)，選擇 Other type of secrets (其他秘密類型)。
4. 將自訂秘密詳細資訊指定為 Key (金鑰) 與 Value (值) 對。例如，您可以指定 `UserName` 的金鑰，然後提供適當的使用者名稱作為其值。新增名為 `Password` 的第二個金鑰，再輸入密碼文字作為其值。您也可以新增資料庫名稱、伺服器位址或 TCP 連接埠的項目。您可以新增任意數量的對組用以存放您需要的資訊。

   或者，您可以選擇 Plaintext (純文字) 標籤，然後以任何您喜歡的方式輸入秘密值。
5. 選擇您要用來 AWS KMS 加密密碼中受保護文字的加密金鑰。如果您未選擇一個金鑰，則 Secrets Manager 將查看帳戶是否有預設金鑰，若有便會使用該金鑰。如果預設金鑰不存在，Secrets Manager 將自動為您建立一個。您也可以選擇 Add new key (新增金鑰) 來建立專供此秘密使用的自訂 KMS 金鑰。若要建立自己的 KMS 金鑰，您必須擁有在帳戶中建立 KMS 金鑰的許可。
6. 選擇下一步。
7. 針對 Secret name (秘密名稱)，請輸入可選的路徑與名稱，例如 **`production/MyAwesomeAppSecret`** 或 **`development/TestSecret`**，然後選擇 Next (下一步)。您也可以選擇性新增描述，協助您日後回憶起此秘密的用途。

   秘密名稱必須是 ASCII 字母、數字或下列任一字元：/_+=.@-
8. (選用) 在此階段，您可以為秘密設定輪換。針對此程序，維持 Disable automatic rotation (停用自動輪換)，然後選擇 Next (下一步)。

   如需如何設定新密碼或現有密碼輪換的相關資訊，請參閱輪換您的 AWS Secrets Manager 密碼。
9. 檢視您的設定，然後選擇 Store secret (存放秘密) 以儲存您在 Secrets Manager 中輸入作為新秘密的所有內容。

使用 Systems Manager 參數存放區指定敏感資料

使用時 AWS Batch，您可以將敏感資料插入容器，方法是將敏感資料儲存在 AWS Systems Manager 參數存放區參數中，然後在容器定義中參考這些資料。

主題

使用 Systems Manager 參數存放區指定敏感資料的注意事項

使用 Systems Manager 參數存放區參數來指定容器的敏感資料時，應考慮以下事項。

- 此功能要求您的容器執行個體具有 1.23.0 版或更新版本的容器代理程式。不過，我們建議您使用最新版的容器代理程式。如需有關檢查代理程式版本和更新至最新版本的資訊，請參閱 Amazon 彈性容器服務開發人員指南中的更新 Amazon ECS 容器代理程式。
- 當容器初始啟動時，機密資料會插入工作的容器中。如果後續更新或輪換秘密或參數存放區參數，則容器不會自動收到更新的值。您必須啟動新工作，才能強制啟動具有更新密碼的全新工作。

AWS Batch 密碼所需的 IAM 許可

若要使用此功能，您必須具有執行角色，並在工作定義中參照它。這可讓 Amazon ECS 容器代理程式提取必要的 AWS Systems Manager 資源。如需詳細資訊，請參閱 AWS Batch 執行 IAM 角色。

若要提供對您建立之 AWS Systems Manager 參數存放區參數的存取權，請手動將下列權限新增為內嵌原則至執行角色。如需詳細資訊，請參閱 IAM 使用者指南中的新增和移除 IAM 政策。

- `ssm:GetParameters` - 如果您在任務定義中參考 Systems Manager 參數存放區參數，才需要此項目。
- `secretsmanager:GetSecretValue` - 如果您直接參考 Secrets Manager 秘密，或者您的 Systems Manager 參數存放區參數參考任務定義中的 Secrets Manager 秘密，才需要此項目。
- `kms:Decrypt` - 只有在您的秘密使用自訂 KMS 金鑰而非預設金鑰時，才需要此項目。您的自訂金鑰的 ARN 應該新增為資源。

下列內嵌政策範例新增必要許可：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetParameters",
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:ssm:<region>:<aws_account_id>:parameter/<parameter_name>",
        "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:<secret_name>",
        "arn:aws:kms:<region>:<aws_account_id>:key/<key_id>"
      ]
    }
  ]
}
```

插入敏感資料作為環境變數

在您的容器定義內，將 `secrets` 指定為要在容器中設定的環境變數名稱，以及 Systems Manager 參數存放區參數 (含有要呈現給容器的敏感資料) 的完整 ARN。

以下是工作定義的片段，顯示參考 Systems Manager 參數存放區參數時的格式。如果「Systems Manager 參數存放區」參數與您正在啟動的工作位於相同的區域中，則您可以使用完整 ARN 或參數名稱。如果參數存在於不同區域，則必須指定完整 ARN。

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

將敏感資料插入日誌組態

在您的容器定義內，指定 `logConfiguration` 時，您可用要在容器中設定的日誌驅動程式選項名稱指定 `secretOptions`，以及 Systems Manager 參數存放區參數 (含有要呈現給容器的敏感資料) 的完整 ARN。

> **Important**
>
> 如果「Systems Manager 參數存放區」參數與您正在啟動的工作位於相同的區域中，則您可以使用完整 ARN 或參數名稱。如果參數存在於不同區域，則必須指定完整 ARN。

以下是工作定義的片段，顯示參考 Systems Manager 參數存放區參數時的格式。

```
{
  "containerProperties": [{
    "logConfiguration": [{
      "logDriver": "fluentd",
      "options": {
        "tag": "fluentd demo"
      },
      "secretOptions": [{
        "name": "fluentd-address",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
      }]
    }]
  }]
}
```

建立 AWS Systems Manager 參數存放區參數

您可以使用 AWS Systems Manager 主控台為您的機密資料建立「Systems Manager 參數存放區」參數。如需詳細資訊，請參閱 《AWS Systems Manager 使用者指南》中的演練：在命令中建立和使用參數 (主控台)。

建立參數存放區參數

1. 請在以下位置開啟 AWS Systems Manager 主控台。 https://console.aws.amazon.com/systems-manager/
2. 在導覽窗格中，選擇 Parameter Store (參數存放區)、Create parameter (建立參數)。

3. 針對 Name (名稱)，輸入階層和參數名稱。例如，輸入 `test/database_password`。
4. 針對 Description (描述)，輸入選擇性描述。
5. 在「類型」中，選擇「字串」StringList、、或SecureString。

> **Note**
>
> - 如果您選擇 SecureString，則會顯示 KMS 金鑰識別碼欄位。如果您不提供 KMS 金鑰 ID、KMS 金鑰 ARN、別名名稱或別名 ARN，系統會使用 `alias/aws/ssm`。這是 Systems Manager 的預設 KMS 金鑰。若要避免使用此金鑰，請選擇自訂金鑰。如需詳細資訊，請參閱 《AWS Systems Manager 使用者指南》中的使用安全字串參數。
> - 當您在主控台使用 `key-id` 參數及自訂 KMS 金鑰別名名稱或別名 ARN 建立安全字串參數時，您必須在別名前面指定字首 `alias/`。以下是 ARN 範例：
>
>   ```
>   arn:aws:kms:us-east-2:123456789012:alias/MyAliasName
>   ```
>
>   下列是別名名稱範例：
>
>   ```
>   alias/MyAliasName
>   ```

6. 針對 Value (值)，輸入一個值。例如 `MyFirstParameter`。如果您選擇此選項 SecureString，則會完全依照您輸入的值進行遮罩。
7. 選擇 Create parameter (建立參數)。

工作的私人登錄驗證

使用的工作專用登錄驗證 AWS Secrets Manager 可讓您安全地儲存您的認證，然後在工作定義中參照這些認證。這提供了一種方法，可以引用存在於私人登錄中的容器映像檔，而這些映像檔 AWS 之外需要在工作定義中進行驗證。Amazon EC2 執行個體和 Fargate 上託管的任務支援此功能。

> **Important**
>
> 如果您的任務定義參考儲存在 Amazon ECR 中的影像，則此主題不適用。如需詳細資訊，請參閱 《Amazon Elastic Container Registry 使用者指南》中的搭配使用 Amazon ECR 映像與 Amazon ECS。

對於 Amazon EC2 執行個體上託管的任務，此功能需要容器代理程式的版本`1.19.0`或更新版本。不過，我們建議您使用最新版的容器代理程式。如需如何檢查代理程式版本並更新至最新版本的相關資訊，請參閱 Amazon 彈性容器服務開發人員指南中的更新 Amazon ECS 容器代理程式。

對於在 Fargate 上託管的工作，此功能需要平台版本`1.2.0`或更高版本。如需相關資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 AWS Fargate Linux 平台版本。

在您的容器定義內，使用您所建立的秘密的詳細資訊來指定 `repositoryCredentials` 物件。您參考的密碼可以來自與使用它的工作不 AWS 區域 同的帳戶，也可能來自不同的帳戶。

> **Note**
>
> 使用 AWS Batch API、或 AWS SDK 時 AWS CLI，如果密碼存在於與您正在啟動的工 AWS 區域 作相同，則您可以使用完整的 ARN 或密碼名稱。如果此秘密已存在於不同帳戶中，則必須指定秘密的完整 ARN。使用時 AWS Management Console，必須始終指定密碼的完整 ARN。

以下是顯示必要參數的工作定義片段：

```
"containerProperties": [
  {
    "image": "private-repo/private-image",
    "repositoryCredentials": {
      "credentialsParameter":
 "arn:aws:secretsmanager:region:123456789012:secret:secret_name"
    }
  }
]
```

私有登錄檔身分驗證所需的 IAM 許可

使用此功能需要執行角色。這可讓容器代理程式提取容器映像。如需詳細資訊，請參閱 AWS Batch 執行 IAM 角色。

若要提供對您建立之密碼的存取權，請將下列權限新增為內嵌原則至執行角色。如需詳細資訊，請參閱新增和移除 IAM 政策。

- `secretsmanager:GetSecretValue`

- `kms:Decrypt` - 只有在您的金鑰使用自訂 KMS 金鑰而非預設金鑰時，才需要此項目。您的自訂金鑰的 Amazon Resource Name (ARN) 必須新增為資源。

下列為新增許可的內嵌政策範例。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region:123456789012:secret:secret_name",
        "arn:aws:kms:region:123456789012:key/key_id"
      ]
    }
  ]
}
```

使用私有登錄身分驗證

建立基本秘密

用 AWS Secrets Manager 於為您的私人登錄認證建立密碼。

1. 請在以下位置開啟 AWS Secrets Manager 主控台。 https://console.aws.amazon.com/secretsmanager/
2. 選擇儲存新機密。
3. 針對 Select secret type (選取秘密類型)，選擇 Other type of secrets (其他秘密類型)。
4. 選取 Plaintext (純文字)，然後使用下列格式輸入您的私有登錄登入資料：

   ```
   {
     "username" : "privateRegistryUsername",
     "password" : "privateRegistryPassword"
   }
   ```

5. 選擇下一步。

6. 針對 Secret name (秘密名稱)，請輸入選用的路徑與名稱，例如 **`production/MyAwesomeAppSecret`** 或 **`development/TestSecret`**，然後選擇 Next (下一步)。您也可以選擇性新增描述，協助您日後回憶起此秘密的用途。

   秘密名稱必須是 ASCII 字母、數字或下列任一字元：`/_+=.@-`。

7. (選用) 在此階段，您可以為秘密設定輪換。針對此程序，維持 Disable automatic rotation (停用自動輪換)，然後選擇 Next (下一步)。

   如需有關如何設定新密碼或現有密碼輪換的指示，請參閱輪換您的 AWS Secrets Manager 密碼。

8. 檢閱您的設定，然後選擇 Store secret (存放秘密) 以儲存您在 Secrets Manager 中輸入作為新秘密的所有內容。

註冊工作定義，然後在 [私人登錄] 下方開啟 [私人登錄驗證]。然後，在 Secrets Manager ARN 或名稱中，輸入密碼的 Amazon Resource Name (ARN)。如需更多詳細資訊，請參閱 私有登錄檔身分驗證所需的 IAM 許可。

Amazon EFS 磁碟區

Amazon Elastic File System (Amazon EFS) 提供簡單、可擴展的檔案儲存，以便與您的任AWS Batch務搭配使用。利用 Amazon EFS，儲存容量即可有彈性。它會在您加入和移除檔案時自動縮放。您的應用程式可在需要時具備所需的儲存容量。

您可以搭AWS Batch配使用 Amazon EFS 檔案系統，跨容器執行個體叢集匯出檔案系統資料。如此一來，您的工作就可以存取相同的永久性儲存體。但您必須在 Docker 常駐程式啟動之前，先設定您的容器執行個體 AMI，以掛載 Amazon EFS 檔案系統。此外，您的工作定義必須參考容器執行個體上的磁碟區掛載，才能使用檔案系統。以下各節可協助您開始在搭配使用 Amazon EFS AWS Batch。

Amazon EFS 磁碟區考量事項

使用 Amazon EFS 磁碟區時應考慮以下事項：

- 對於使用 EC2 資源的任務，Amazon EFS 檔案系統支援已新增為公開預覽，其中包含 Amazon ECS 最佳化 AMI 版本 (20191212含容器代理程式版本 1.35.0)。不過，Amazon EFS 檔案系統支援已透過 Amazon ECS 最佳化 AMI 版本 20200319 (包含 Amazon EFS 存取點和 IAM 授權功能的容器代理程式版本 1.38.0) 提供正式推出。我們建議您使用 Amazon ECS 最佳化 AMI 版本20200319或更新版本，以利用這些功能。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 Amazon ECS 最佳化 AMI 版本。

> **Note**
>
> 如果您建立自己的 AMI，則必須使用容器代理程式 1.38.0 或更新版本 (1.38.0-1 或更新`ecs-init`版本)，並在 Amazon EC2 執行個體上執行下列命令。這一切都是為了啟用 Amazon ECS 卷插件。這些命令取決於您是否使用 Amazon Linux 2 或 Amazon Linux 作為基礎映像。
>
> Amazon Linux 2
>
> ```
> $ yum install amazon-efs-utils
> systemctl enable --now amazon-ecs-volume-plugin
> ```
>
> Amazon Linux
>
> ```
> $ yum install amazon-efs-utils
> sudo shutdown -r now
> ```

- 對於使用 Fargate 資源的任務，在使用平台 1.4.0 或更新版本時，已新增 Amazon EFS 檔案系統支援。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 AWSFargate 平台版本。
- 使用 Fargate 資源在任務中指定 Amazon EFS 磁碟區時，Fargate 會建立負責管理 Amazon EFS 磁碟區的主管容器。主管容器使用少量工作的記憶體。查詢任務中繼資料第 4 版端點時，可看見監督容器。如需詳細資訊，請參閱 AWS Fargate 的 Amazon Elastic Container Service 使用者指南中的任務中繼資料端點版本 4。

使用 Amazon EFS 存取點

Amazon EFS 存取點是 EFS 檔案系統中應用程式特定的進入點，可協助您管理應用程式對共用資料集的存取。如需有關 Amazon EFS 存取點及如何控制對它們的存取的詳細資訊，請參閱《Amazon Elastic File System 使用者指南》中的使用 Amazon EFS 存取點。

存取點可以針對透過存取點提出的所有檔案系統要求，強制執行使用者身分 (包括使用者的 POSIX 群組)。存取點也可以針對檔案系統強制執行不同的根目錄，讓用戶端只能存取指定目錄或其子目錄中的資料。

> **Note**
>
> 建立 EFS 存取點時，您可以在檔案系統上指定要做為根目錄的路徑。當您在AWS Batch工作定義中使用存取點 ID 來參照 EFS 檔案系統時，必須省略根目錄或將其設定為/這會強制在 EFS 存取點上設定的路徑。

您可以使用AWS Batch工作 IAM 角色強制執行特定應用程式使用特定存取點。透過結合 IAM 政策與存取點，您可以輕鬆地為應用程式提供特定資料集的安全存取權。此功能使用 Amazon ECS 身分與存取權管理角色執行任務功能。如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的任務 IAM 角色。

在任務定義中指定 Amazon EFS 檔案系統

若要為您的容器使用 Amazon EFS 檔案系統磁碟區，您必須在任務定義中指定磁碟區和掛接點組態。下列工作定義 JSON 程式碼片段會顯示容器`volumes`和`mountPoints`物件的語法：

```
{
    "containerProperties": [
        {
            "image": "amazonlinux:2",
            "command": [
                "ls",
                "-la",
                "/mount/efs"
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEfsVolume",
                    "containerPath": "/mount/efs",
                    "readOnly": true
                }
            ],
            "volumes": [
                {
                    "name": "myEfsVolume",
                    "efsVolumeConfiguration": {
                        "fileSystemId": "fs-12345678",
                        "rootDirectory": "/path/to/my/data",
                        "transitEncryption": "ENABLED",
                        "transitEncryptionPort": integer,
```

```
                        "authorizationConfig": {
                            "accessPointId": "fsap-1234567890abcdef1",
                            "iam": "ENABLED"
                        }
                    }
                }
            ]
        }
    ]
}
```

`efsVolumeConfiguration`

類型：物件

必要：否

只有使用 Amazon EFS 磁碟區時才會指定此參數。

`fileSystemId`

類型：字串

必要：是

要使用的 Amazon EFS 檔案系統識別碼。

`rootDirectory`

類型：字串

必要：否

在 Amazon EFS 檔案系統中的目錄，其將掛載作為主機內的根目錄。如果省略此參數，使用 Amazon EFS 磁碟區的根目錄。指定 / 的效果與忽略此參數的效果相同。它的長度最多可以有 4,096 個字符。

> **Important**
>
> 如果在中指定 EFS 存取點`authorizationConfig`，則必須省略根目錄參數或將其設定為`/`。這會強制執行 EFS 存取點上設定的路徑。

`transitEncryption`

類型：字串

有效值：`ENABLED | DISABLED`

必要：否

決定是否為AWS Batch主機和 Amazon EFS 伺服器之間傳輸的 Amazon EFS 資料啟用加密。若使用 Amazon EFS IAM 授權，則必須啟用傳輸加密。如果省略此參數，系統會使用 `DISABLED` 的預設值。如需詳細資訊，請參閱《Amazon Elastic File System 使用者指南》中的加密傳輸中的資料。

`transitEncryptionPort`

類型：整數

必要：否

在AWS Batch主機和 Amazon EFS 伺服器之間傳送加密資料時要使用的連接埠。如果您未指定傳輸加密連接埠，它會使用 Amazon EFS 掛載協助程式使用的連接埠選擇策略。該值必須介於 0 到 65,535 之間。如需詳細資訊，請參閱《Amazon Elastic File System 使用者指南》中的 EFS 掛載協助程式。

`authorizationConfig`

類型：物件

必要：否

Amazon EFS 檔案系統的授權組態詳細資訊。

`accessPointId`

類型：字串

必要：否

要使用的存取點 ID。如果指定存取點，則`efsVolumeConfiguration`必須省略中的根目錄值或將其設定為`/`。這會強制執行 EFS 存取點上設定的路徑。如果使用存取點，則必須在 `EFSVolumeConfiguration` 中啟用傳輸加密。如需詳細資訊，請參閱《Amazon Elastic File System 使用者指南》中的使用 Amazon EFS 存取點。

`iam`

類型：字串

有效值：ENABLED | DISABLED

必要：否

決定是否在掛接 Amazon EFS 檔案系統時使用AWS Batch任務定義中定義的任務 IAM 角色。如果已啟用，必須在 EFSVolumeConfiguration 中啟用傳輸加密。如果省略此參數，系統會使用 DISABLED 的預設值。如需執行 IAM 角色的詳細資訊，請參閱AWS Batch 執行 IAM 角色。

工作定義範例

以下任務定義範例說明，如何使用環境變數、參數替換和磁碟區掛載等常見模式。

使用環境變數

下列任務定義範例使用環境變數來指定檔案類型和 Amazon S3 URL。此特定範例出自建立簡單的「提取與執行」AWS Batch 任務運算部落格文章。部落格文章中描述的fetch_and_run.sh指令碼會使用這些環境變數從 S3 下載myjob.sh指令碼，並宣告其檔案類型。

即使在此範例中將命令和環境變數硬式編碼到工作定義中，您也可以指定命令和環境變數覆寫，以使工作定義更具用途。

```
{
    "jobDefinitionName": "fetch_and_run",
    "type": "container",
    "containerProperties": {
        "image": "123456789012.dkr.ecr.us-east-1.amazonaws.com/fetch_and_run",
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "2000"
            },
            {
                "type": "VCPU",
                "value": "2"
            }
        ],
        "command": [
            "myjob.sh",
            "60"
        ],
```

```
        "jobRoleArn": "arn:aws:iam::123456789012:role/AWSBatchS3ReadOnly",
        "environment": [
            {
                "name": "BATCH_FILE_S3_URL",
                "value": "s3://my-batch-scripts/myjob.sh"
            },
            {
                "name": "BATCH_FILE_TYPE",
                "value": "script"
            }
        ],
        "user": "nobody"
    }
}
```

使用參數替換

以下任務定義範例說明，如何允許替換參數和設定預設值。

Ref:: 區段中的 command 宣告用於設定的替換參數的預留位置。提交使用此任務定義的任務時，您要指定參數覆寫以填入這些值，例如 inputfile 和 outputfile。接下來的parameters部分設定了的預設值codec，但您可以視需要覆寫該參數。

如需詳細資訊，請參閱參數。

```
{
    "jobDefinitionName": "ffmpeg_parameters",
    "type": "container",
    "parameters": {"codec": "mp4"},
    "containerProperties": {
        "image": "my_repo/ffmpeg",
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "2000"
            },
            {
                "type": "VCPU",
                "value": "2"
            }
        ],
        "command": [
            "ffmpeg",
```

```
            "-i",
            "Ref::inputfile",
            "-c",
            "Ref::codec",
            "-o",
            "Ref::outputfile"
        ],
        "jobRoleArn": "arn:aws:iam::123456789012:role/ECSTask-S3FullAccess",
        "user": "nobody"
    }
}
```

測試 GPU 功能

以下工作定義範例測試 使用 GPU 工作負載 AMI 中所述的 GPU 工作負載 AMI 是否正確設定。此範例工作定義會從中執行 TensorFlow 深層 MNIST 分類器範例。 GitHub

```
{
    "containerProperties": {
        "image": "tensorflow/tensorflow:1.8.0-devel-gpu",
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "32000"
            },
            {
                "type": "VCPU",
                "value": "8"
            }
        ],
        "command": [
            "sh",
            "-c",
            "cd /tensorflow/tensorflow/examples/tutorials/mnist; python mnist_deep.py"
        ]
    },
    "type": "container",
    "jobDefinitionName": "tensorflow_mnist_deep"
}
```

您可以使用前面呼叫的 JSON 文字建立檔案，`tensorflow_mnist_deep.json`然後使用下列命令註冊AWS Batch工作定義：

```
aws batch register-job-definition --cli-input-json file://tensorflow_mnist_deep.json
```

多節點 parallel 工作

下列任務定義範例說明多節點平行任務。如需詳細資訊，請參閱 AWSCompute 部落格中AWS Batch的使用多節點 parallel 工作建立緊密結合的分子動力學工作流程。

```
{
  "jobDefinitionName": "gromacs-jobdef",
  "jobDefinitionArn": "arn:aws:batch:us-east-2:123456789012:job-definition/gromacs-
jobdef:1",
  "revision": 6,
  "status": "ACTIVE",
  "type": "multinode",
  "parameters": {},
  "nodeProperties": {
    "numNodes": 2,
    "mainNode": 0,
    "nodeRangeProperties": [
      {
        "targetNodes": "0:1",
        "container": {
          "image": "123456789012.dkr.ecr.us-east-2.amazonaws.com/gromacs_mpi:latest",
          "resourceRequirements": [
              {
                  "type": "MEMORY",
                  "value": "24000"
              },
              {
                  "type": "VCPU",
                  "value": "8"
              }
          ],
          "command": [],
          "jobRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
          "ulimits": [],
          "instanceType": "p3.2xlarge"
        }
      }
    ]
  }
}
```

Job 佇列

工作會提交至它們所在的工作佇列，直到排定在運算環境中執行為止。一個 AWS 帳戶可以有多個工作佇列。例如，您可以建立將 Amazon EC2 隨需執行個體用於高優先順序任務的佇列，以及使用 Amazon EC2 Spot 執行個體執行低優先順序任務的另一個佇列。Job 佇列具有排程器使用的優先順序，決定應先評估哪些佇列中的工作以便執行。

主題

建立工作佇列

您必須先建立任務佇列，接著才能在 AWS Batch 提交任務。建立工作佇列時，您可以將一或多個計算環境與佇列產生關聯，並指派偏好順序。

您也可以設定工作佇列的優先順序，以決定 AWS Batch 排程器放置工作的順序。這意味著，如果計算環境與多個工作佇列相關聯，則優先順序較高的工作佇列會被指定為優先順序。

建立 Fargate 作佇列

建立遠端工作佇列

1. 開啟主AWS Batch控台，網址為 https://console.aws.amazon.com/batch/。
2. 從導覽列中選取要使用的 AWS 區域。
3. 在導覽窗格中，選擇 [Job 佇列]。
4. 選擇建立。
5. 對於「協調流程」類型，請選擇「Fargate」。
6. 在名稱中，輸入工作佇列的唯一名稱。名稱最多可包含 128 個字元，並且可以包含大寫和小寫字母、數字和底線 (_)。
7. 在「優先順序」中，輸入工作佇列優先順序的整數值。優先順序較高的 Job 佇列會在與相同計算環境相關聯的較低優先順序工作佇列之前執行。優先順序會依遞減順序決定。例如，優先順序值為 10 的任務佇列的排程優先順序會高於優先順序值為 1 的任務佇列。

8. (選擇性) 對於 Amazon 資源名稱 (ARN) 的排程政策，請選擇現有的排程政策。
9. 對於連線的運算環境，請從清單中選取一或多個要與工作佇列產生關聯的運算環境。按照您希望佇列嘗試放置工作佇列的順序選取運算環境。工作排程器會使用您選取運算環境的順序來決定啟動指定工作的計算環境。運算環境必須處於狀態，才能將它們與工作佇列產生關VALID聯。您可以將多達三個運算環境與工作佇列建立關聯。

   > **Note**
   >
   > 與工作佇列相關聯的所有運算環境都必須共用相同的佈建模式。 AWS Batch不支援在單一工作佇列中混合佈建模型。

10. 對於計算環境順序，請選擇向上和向下箭頭來設定您想要的順序。
11. 選擇 [建立工作佇列] 以完成並建立工作佇列。

創建亞 Amazon EC2 任務隊列

若要建立 Amazon EC2 任務佇列

1. 開啟主AWS Batch控台，網址為 https://console.aws.amazon.com/batch/。
2. 從導覽列中選取要使用的 AWS 區域。
3. 在導覽窗格中，選擇 [Job 佇列]。
4. 選擇建立。
5. 對於協調類型，請選擇亞馬遜彈性運算雲端 (Amazon EC2)。
6. 在名稱中，輸入工作佇列的唯一名稱。名稱最多可包含 128 個字元，並且可以包含大寫和小寫字母、數字和底線 (_)。
7. 在「優先順序」中，輸入工作佇列優先順序的整數值。優先順序較高的 Job 佇列會在與相同計算環境相關聯的較低優先順序工作佇列之前執行。優先順序會依遞減順序決定。例如，優先順序值為 10 的任務佇列的排程優先順序會高於優先順序值為 1 的任務佇列。
8. (選擇性) 對於 Amazon 資源名稱 (ARN) 的排程政策，請選擇現有的排程政策。
9. 對於連線的運算環境，請從清單中選取一或多個要與工作佇列產生關聯的運算環境。按照您希望佇列嘗試放置工作佇列的順序選取運算環境。工作排程器會使用您選取運算環境的順序來決定啟動指定工作的計算環境。運算環境必須處於狀態，才能將它們與工作佇列產生關VALID聯。您可以將多達三個運算環境與工作佇列建立關聯。如果您沒有現有的運算環境，請選擇 [建立運算環境]

> **Note**
>
> 與工作佇列相關聯的所有運算環境都必須共用相同的佈建模式。 AWS Batch不支援在單一工作佇列中混合佈建模型。

10. 對於計算環境順序，請選擇向上和向下箭頭來設定您想要的順序。
11. 選擇 [建立工作佇列] 以完成並建立工作佇列。

創建 Amazon EKS 任務隊列

若要建立 Amazon EKS 任務佇列

1. 開啟主AWS Batch控台，網址為 https://console.aws.amazon.com/batch/。
2. 從導覽列中選取要使用的 AWS 區域。
3. 在導覽窗格中，選擇 [Job 佇列]。
4. 選擇建立。
5. 對於協調類型，請選擇 Amazon Elastic Kubernetes Service (Amazon EKS)。
6. 在名稱中，輸入工作佇列的唯一名稱。名稱最多可包含 128 個字元，並且可以包含大寫和小寫字母、數字和底線 (_)。
7. 在 Priority (優先順序)，為任務佇列的優先順序輸入整數值。優先順序較高的 Job 佇列會在與相同計算環境相關聯的較低優先順序工作佇列之前執行。優先順序會依遞減順序決定。例如，優先順序值為 10 的任務佇列的排程優先順序會高於優先順序值為 1 的任務佇列。
8. (選擇性) 對於 Amazon 資源名稱 (ARN) 的排程政策，請選擇現有的排程政策。
9. 對於連線的運算環境，請從清單中選取一或多個要與工作佇列產生關聯的運算環境。按照您希望佇列嘗試放置工作佇列的順序選取運算環境。工作排程器會使用您選取運算環境的順序來決定啟動指定工作的計算環境。運算環境必須處於狀態，才能將它們與工作佇列產生關`VALID`聯。您可以將多達三個運算環境與工作佇列建立關聯。

> **Note**
>
> 與工作佇列相關聯的所有運算環境都必須共用相同的佈建模式。 AWS Batch不支援在單一工作佇列中混合佈建模型。

> **Note**
>
> 與任務佇列關聯的所有運算環境皆必須共用相同的架構。AWS Batch 不支援在單一任務佇列中混用運算環境架構類型。

10. 對於計算環境順序，請選擇向上和向下箭頭來設定您想要的順序。
11. 選擇 [建立工作佇列] 以完成並建立工作佇列。

Job 佇列範本

以下是空的工作佇列範本。您可以使用此樣板來建立工作佇列。然後，您可以將此工作佇列儲存到檔案中，並將其與 AWS CLI `--cli-input-json`選項搭配使用。如需這些參數的詳細資訊，請參閱 AWS Batch API 參考CreateJobQueue中的。

```
{
   "computeEnvironmentOrder": [
      {
         "computeEnvironment": "",
         "order": 0
      }
   ],
   "jobQueueName": "",
   "jobStateTimeLimitActions": [
      {
         "state": "RUNNABLE",
         "action": "CANCEL",
   "maxTimeSeconds": 0,
         "reason": ""

      }
   ],
   "priority": 0,
   "schedulingPolicyArn": "",
   "state": "ENABLED",
   "tags": {
      "KeyName": ""
   }
}
```

> **Note**
>
> 您可以使用下列 AWS CLI 命令產生前面的工作佇列範本。
>
> ```
> $ aws batch create-job-queue --generate-cli-skeleton
> ```

Job 佇列參數

Job 佇列分為四個基本元件：名稱、狀態、優先順序和計算環境順序。本節將分析這些與元件相關聯的元件。

主題

Job 佇列名稱

`jobQueueName`

工作佇列的名稱。可以包含最多達 128 個字元 (大小寫)、數字和底線。

類型：字串

必要：是

Job 佇列狀態時間限制動作

jobStateTimeLimitActions

AWS Batch 對保留在工作佇列標頭處於指定狀態超過指定時間的工作執行的一組動作。 AWS Batch 將在通過後maxTimeSeconds執行每個動作。（注意：最小值maxTimeSeconds為 600（10 分鐘），其最大值為 86,400（24 小時）。）

類型：JobStateTimeLimitActions 物件陣列

必要：否

優先順序

priority

任務佇列的優先順序。當關聯相同的運算環境時，優先順序較高的任務佇列 (或 priority參數更高的整數值) 會先進行評估。優先順序係以遞減順序決定，例如，在決定排程偏好時，優先順序值 10 的任務佇列會先於優先順序值 1 的任務佇列。所有運算環境都必須是 Amazon EC2 (EC2或SPOT) 或 Fargate (FARGATE或FARGATE_SPOT)。Amazon EC2 和 Fargate 運算環境不能混合使用。

類型：整數

必要：是

排程政策

schedulingPolicyArn

任務佇列的排程政策的 Amazon 資源名稱 (ARN)。沒有排程原則的 Job 佇列會以先進先出 (FIFO) 模式排程。工作佇列具有排程原則後，即可將其取代，但無法移除。沒有排程原則的工作佇列會排程為 FIFO 工作佇列，且無法新增排程原則。具有排程原則的工作佇列最多可以有 500 個作用中的公平共用識別碼。達到上限時，任何新增公平共用識別碼的工作都會失敗。

類型：字串

必要：否

State

`state`

工作佇列的狀態。如果工作佇列狀態為 ENABLED (預設值)，則可接受工作。如果任務佇列狀態為 DISABLED，則無法將新任務新增至佇列，但佇列中已有的任務可以完成。

類型：字串

有效值：ENABLED | DISABLED

必要：否

運算環境順序

`computeEnvironmentOrder`

運算環境的設定對應到一個任務佇列以及彼此之間的相對順序。任務排程器使用此參數來判斷應執行特定任務的運算環境。運算環境必須先處於 VALID 狀態，才能與任務佇列建立關聯。您可以將多達三個運算環境與工作佇列建立關聯。所有運算環境都必須是 Amazon EC2 (EC2或SPOT) 或 Fargate (FARGATE或FARGATE_SPOT)。Amazon EC2 和 Fargate 運算環境不能混合使用。

> **Note**
>
> 與工作佇列相關聯的所有運算環境都必須共用相同的架構。 AWS Batch 不支援在單一工作佇列中混合運算環境架構類型。

類型：ComputeEnvironmentOrder 物件陣列

必要：是

`computeEnvironment`

運算環境的 Amazon Resource Name (ARN)。

類型：字串

必要：是

`order`

運算環境順序。運算環境會嘗試遞增排序。例如，如果兩個運算環境關聯至一個任務佇列，較低`order`整數值的運算環境會優先嘗試任務放置。

標籤

`tags`

要與工作佇列關聯的索引鍵值配對標籤。如需詳細資訊，請參閱 標記您的 AWS Batch 資源。

類型：字串到字串映射

必要：否

檢視工作佇列狀態

建立工作佇列並送出工作之後，能夠監視其進度非常重要。您可以使用「Job 詳細資訊」頁面來檢閱、管理和監視工作佇列。

檢視工作佇列資訊

在 AWS Batch 主控台中，選取導覽窗格中的 Job 佇列，然後選擇所需的工作佇列以檢視其詳細資料。您可以在此頁面複查和管理工作佇列，以及查看佇列作業的其他相關資訊，例如工作佇列快照、工作狀態限制、環境順序、標記以及工作佇列的 JSON 程式碼。

Job 佇列詳情

本節提供工作佇列的簡介和維護選項。需要注意的是，您可以在本節中找到 Amazon 資源名稱（ARN）是非常重要的。

若要透過尋找此資訊 AWS Command Line Interface，請搭配工`DescribeJobQueues`作佇列名稱或對應的 ARN 使用作業。

Job 佇列快照

此段落提供佇列中前 100 個`RUNNABLE`工作的靜態清單。您可以使用搜尋欄位，透過搜尋結果區段中的任何欄位搜尋資訊來縮小清單範圍。快照結果區域中的工作會根據工作佇列的執行策略來排序。對於 first-in-first-out (FIFO) 工作佇列，工作的順序是根據提交時間而定。對於AWS Batch 公平共用排程 (FSS) 工作佇列，工作的順序是根據工作優先順序和共用使用量而定。

因為結果是工作佇列的快照，所以結果清單不會自動更新。若要更新清單，請選擇區段頂端的重新整理。選擇 Job 的名稱超連結以瀏覽至「工作詳細資訊」，並檢視工作的狀態及其他相關資訊。

若要透過尋找此資訊 AWS CLI，請搭配工`GetJobQueueSnapshot`作佇列名稱或對應的 ARN 使用作業。

Job 狀態限制

您可以使用此索引標籤來檢閱工作在取消前可維持在某個RUNNABLE狀態的時間長度的組態資訊。

若要透過尋找此資訊 AWS CLI，請搭配工`DescribeJobQueues`作佇列名稱或對應的 ARN 使用作業。

環境秩序

如果您的工作佇列在多個環境中執行，此索引標籤會提供其順序和概觀。

若要透過尋找此資訊 AWS CLI，請搭配工`DescribeJobQueues`作佇列名稱或對應的 ARN 使用作業。

標籤

使用此標籤可以檢閱和管理與此工作佇列相關聯的標籤。

JSON

使用此索引標籤可複製與此工作佇列相關聯的 JSON 程式碼。然後，您可以針對 AWS CloudFormation 範 AWS CLI 本和指令碼重複使用 JSON。

Job 排程

AWS Batch 排程器會評估提交至工作佇列的工作的時間、位置以及如何執行。如果您在建立工作佇列時未指定排程原則，工 AWS Batch 作排程器會預設為先進先出 (FIFO) 策略。FIFO 策略可能會導致重要的工作被「卡住」後面提交的工作。透過指定不同的排程原則，您可以根據自己的特定需求配置運算資源。

> **Note**
>
> 如果您要排定工作執行的特定順序，請使用中的`dependsOn`參數`SubmitJob`來指定每個工作的相依性。

如果您建立排程原則並將其附加至工作佇列，則會開啟公平共用排程。如果工作佇列具有排程原則，則排程原則會決定工作的執行順序。如需詳細資訊，請參閱 排程原則。

共用識別碼

您可以使用共用識別碼來標記工作，並區分使用者和工作負載。 AWS Batch 排程器會使用公式來追蹤每個*(T * weightFactor)*公平共用識別碼的使用情況，其中*T*是一段時間內的 vCPU 使用率。排程器會從共用識別碼中挑選使用量最低的工作。您可以使用公平共用識別碼，而不會覆寫它。

> **Note**
>
> 共用識別碼在工作佇列中是唯一的，不會跨工作佇列彙總。

您可以設定排程優先順序，以設定工作在共用識別碼上執行的順序。排程優先順序較高的工作會先排定。如果您未指定排程原則，則提交至工作佇列的所有作業都會以 FIFO 順序排程。當您提交工作時，您無法指定共用識別碼或排程優先順序。

> **Note**
>
> 除非明確覆寫，否則附加的計算資源會平均分配給所有共用識別碼

公平分享排程

公平共用排程提供一組控制項以協助排程工作。

> **Note**
>
> 如需排程原則參數的詳細資訊，請參閱排程原則參數。

- 共用衰減秒數 — AWS Batch 排程器用來計算每個公平共用識別碼的公平份額百分比的期間 (以秒為單位)。值為零表示只測量目前的使用情況。較長的衰減時間會給予更多的重量。

  > **Note**
  >
  > 衰減的時間週期計算方式為：*shareDecaySeconds + OrderMinutes*其中*OrderMinutes* 是以分鐘為單位的時間順序。

- 計算保留區 — 防止單一共用識別碼中的工作耗盡附加至工作佇列的所有資源。保留比率是*computeReservation/100)^ActiveFairShares*活躍的公平共享標識符的數量。ActiveFairShares

  > **Note**
  >
  > 如果共用識別碼具有SUBMITTED、、PENDINGRUNNABLE、或RUNNING狀態的工作STARTING，則會被視為使用中共用識別碼。在衰減期限到期後，共用識別碼會被視為非作用中。

- 權重係數 — 共用識別碼的權重係數。預設值為 1。較低的值可讓共用識別碼中的工作執行，或為共用識別碼提供額外的執行階段。例如，使用加權係數為 0.125 (1/8) 的共用識別碼的工作，會指派八倍於使用共用識別碼 (加權係數為 1) 的工作計算資源。

  > **Note**
  >
  > 只有在需要更新預設加權係數 1 時，才需要定義此屬性。

當 Job 佇列處於使用中狀態且正在處理工作時，您可以透過RUNNABLE工作佇列快照檢閱前 100 個工作的清單。如需詳細資訊，請參閱檢視工作佇列狀態。

運算環境

任務佇列會對應至一或多個運算環境。運算環境包含用於執行容器化批次任務的 Amazon ECS 容器執行個體。特定計算環境也可以對應至一個或多個工作佇列。在工作佇列中，每個關聯的運算環境都有排程器使用的順序來決定準備好執行的工作將在何處執行。如果第一個計算環境的狀態為VALID且具有可用資源，則會將工作排程到該計算環境中的容器執行個體。如果第一個計算環境的狀態為INVALID或無法提供適當的計算資源，則排程器會嘗試在下一個計算環境上執行工作。

主題

受管理運算環境

您可以使用受管運算環境來 AWS Batch 管理環境中運算資源的容量和執行個體類型。這是根據您在建立運算環境時定義的計算資源規格而定。您可以選擇使用 Amazon EC2 隨需執行個體和 Amazon EC2 競價型執行個體。或者，您也可以在受管理的運算環境中使用 Fargate 和 Fargate Spot 容量。使用 Spot 執行個體時，您可以選擇性地設定最高價格。如此一來，Spot 執行個體只會在競價型執行個體價格低於隨需價格的指定百分比時啟動。

> **Important**
>
> 在 Windows containers on AWS Fargate上不支援 Fargate 端競價型執行個體。如果將工作提交至僅使用 Fargate Spot 運算環境的工作佇列，則會封鎖工作佇列。 FargateWindows

受管運算環境會將 Amazon EC2 執行個體啟動到您指定的 VPC 和子網路，然後在 Amazon ECS 叢集中註冊這些執行個體。Amazon EC2 執行個體需要外部網路存取權，才能與 Amazon ECS 服務端點通訊。某些子網路不提供具有公有 IP 地址的 Amazon EC2 執行個體。如果您的 Amazon EC2 執行個體沒有公有 IP 地址，則必須使用網路位址轉譯 (NAT) 來取得此存取權。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 NAT 閘道。如需如何建立 VPC 的詳細資訊，請參閱建立虛擬私有雲 。

根據預設， AWS Batch 受管運算環境會針對運算資源使用最新、核准的 Amazon ECS 最佳化 AMI 版本。不過，出於各種原因，您可能需要建立自己的 AMI 以用於受管理的運算環境。如需詳細資訊，請參閱 計算資源 AMI。

> **Note**
>
> AWS Batch 建立之後，不會自動升級運算環境中的 AMI。例如，當 Amazon ECS 最佳化 AMI 的較新版本發佈時，它不會更新您運算環境中的 AMI。您必須負責客體作業系統的管理。這包括任何更新和安全性修補程式。您也必須負責安裝在運算資源上的任何其他應用程式軟體或公用程式。有兩種方法可以使用新 AMI 進行 AWS Batch 工作。原始方式是完成以下步驟：
>
> 1. 新建內有新 AMI 的運算環境。
> 2. 將運算環境新增至現有的任務佇列。
> 3. 將較早的運算環境從任務佇列移除。
> 4. 刪除較早的運算環境。
>
> 2022 年 4 月，增 AWS Batch 加了對更新計算環境的增強支援。如需詳細資訊，請參閱 更新運算環境。若要使用運算環境的增強型更新功能來更新 AMI，請遵循下列規則：
>
> - 請勿設定服務角色 (`serviceRole`) 參數，或將其設定為服AWSServiceRoleForBatch務連結角色。
> - 將配置策略 (`allocationStrategy`) 參數設定為`BEST_FIT_PROGRESSIVE`、`SPOT_CAPACITY_OPTIMIZED`或`SPOT_PRICE_CAPACITY_OPTIMI`
> - 將更新至最新映像版本 (`updateToLatestImageVersion`) 參數設定為`true`。

- 請勿在imageId、imageIdOverride(中 ec2Configuration) 或啟動範本 (launchTemplate) 中指定 AMI ID。在這種情況下，請 AWS Batch 選取啟動基礎設施更新時支援的 AWS Batch 最新 Amazon ECS 最佳化 AMI。或者，您可以在imageId或imageIdOverride參數中指定 AMI ID，或由LaunchTemplate屬性識別的啟動範本。變更任何這些屬性都會啟動基礎結構更新。如果在啟動範本中指定 AMI ID，則無法透過在imageId或imageIdOverride參數中指定 AMI ID 來取代它。只能透過指定不同的啟動範本來取代它。或者，如果啟動範本版本設定為$Default或$Latest，則透過為啟動範本設定新的預設版本 (如果有的話$Default)，或將新版本新增至啟動範本 (如果有的話$Latest)。

  如果遵循這些規則，則啟動基礎結構更新的任何更新都會導致 AMI ID 被重新選取。如果啟動範本 (launchTemplate) 中的version設定設為$Latest或$Default，則會在基礎結構更新時評估啟動範本的最新或預設版本，即使未更新也launchTemplate是如此。

建立多節點 parallel 工作時的考量

AWS Batch 建議您建立專用的運算環境，以執行多節點 parallel (MNP) 作業和非 MNP 工作。這是因為在受管理的運算環境中建立運算容量的方式所致。建立新的受管理運算環境時，如果您指定的minvCpu值大於零，則只 AWS Batch 會建立執行個體集區以用於非 MNP 工作。如果提交多節點 parallel 工作，則 AWS Batch 會建立新的執行個體容量以執行多節點 parallel 作業。如果在設定了minvCpus或maxvCpus值的相同計算環境中同時執行單節點和多節點 parallel 工作，則如果無法使用所需的計算資源， AWS Batch 則在建立執行新工作所需的計算資源之前，等待目前的工作完成。

未受管理的運算環境

在未受管運算環境中，由您管理自己的運算資源。您必須確認用於運算資源的 AMI 是否符合 Amazon ECS 容器執行個體 AMI 規格。如需詳細資訊，請參閱 運算資源 AMI 規格 及 建立計算資源 AMI。

> **Note**
>
> AWS 未受管理的運算環境不支援 Fargate 資源。

建立非受管運算環境之後，請使用 DescribeComputeEnvironmentsAPI 作業來檢視計算環境詳細資料。尋找與環境相關聯的 Amazon ECS 叢集，然後將您的容器執行個體手動啟動到該 Amazon ECS 叢集。

下列 AWS CLI 指令也提供 Amazon ECS 叢集 ARN。

```
$ aws batch describe-compute-environments \
    --compute-environments unmanagedCE \
    --query "computeEnvironments[].ecsClusterArn"
```

如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的啟動 Amazon ECS 容器執行個體。啟動運算資源時，請指定資源向下列 Amazon EC2 使用者資料註冊的 Amazon ECS 叢集 ARN。以先前的命令取*ecsClusterArn*代叢集 ARN。

```
#!/bin/bash
echo "ECS_CLUSTER=ecsClusterArn" >> /etc/ecs/ecs.config
```

計算資源 AMI

根據預設， AWS Batch 受管運算環境會針對運算資源使用最新、核准的 Amazon ECS 最佳化 AMI 版本。不過，您可能想要建立自己的 AMI，以便用於受管理和未受管理的運算環境。如果您需要以下任何一項，我們建議您創建自己的 AMI：

- 增加 AMI 根或資料磁碟區的儲存大小
- 新增支援 Amazon EC2 執行個體類型的執行個體儲存磁碟
- 自訂 Amazon ECS 容器代理程式
- 自訂泊塢視窗
- 設定 GPU 工作負載 AMI 以允許容器存取受支援的 Amazon EC2 執行個體類型上的 GPU 硬體

> **Note**
>
> 建立運算環境之後， AWS Batch 不會升級運算環境中的 AMI。 AWS Batch 當有更新版本的 Amazon ECS 最佳化 AMI 可用時，也不會更新運算環境中的 AMI。您必須負責客體作業系統的管理。這包括任何更新和安全性修補程式。您也必須負責安裝在運算資源上的任何其他應用程式軟體或公用程式。要為您的 AWS Batch 工作使用新的 AMI，請執行以下操作：
>
> 1. 新建內有新 AMI 的運算環境。
> 2. 將運算環境新增至現有的任務佇列。
> 3. 將較早的運算環境從任務佇列移除。
> 4. 刪除較早的運算環境。

2022 年 4 月，增 AWS Batch 加了對更新計算環境的增強支援。如需詳細資訊，請參閱 更新運算環境。若要使用運算環境的增強型更新功能來更新 AMI，請遵循下列規則：

- 請勿設定服務角色 (serviceRole) 參數，或將其設定為服AWSServiceRoleForBatch務連結角色。
- 將配置策略 (allocationStrategy) 參數設定為BEST_FIT_PROGRESSIVESPOT_CAPACITY_OPTIMIZED、或SPOT_PRICE_CAPACITY_OPTIMIZED。
- 將更新至最新映像版本 (updateToLatestImageVersion) 參數設定為true。
- 請勿在imageId、imageIdOverride(中 ec2Configuration) 或啟動範本 (launchTemplate) 中指定 AMI ID。如果您未指定 AMI ID，請 AWS Batch 選取啟動基礎設施更新時 AWS Batch 支援的最新 Amazon ECS 最佳化 AMI。或者，您可以在imageId或imageIdOverride參數中指定 AMI ID。或者，您可以指定由LaunchTemplate屬性識別的啟動範本。變更任何這些屬性都會啟動基礎結構更新。如果在啟動範本中指定 AMI ID，則無法透過在imageId或imageIdOverride參數中指定 AMI ID 來取代 AMI 識別碼。AMI ID 只能透過指定不同的啟動範本來取代。如果啟動範本版本設定為$Default或$Latest，則可以透過設定啟動範本的新預設版本 (if$Default) 或將新版本新增至啟動範本 (如果$Latest) 來取代 AMI ID。

如果遵循這些規則，則啟動基礎結構更新的任何更新都會導致 AMI ID 重新選取。如果啟動範本 (launchTemplate) 中的version設定設為$Latest或$Default，則會在基礎結構更新時評估啟動範本的最新或預設版本，即使launchTemplate未更新也是如此。

主題

運算資源 AMI 規格

基本 AWS Batch 運算資源 AMI 規格包含下列項目：

必要

- 在 HVM 虛擬化類型 AMI 上執行至少 3.10 版 Linux 核心的現代 Linux 發行版本。不支援視窗容器。

> **Important**
>
> 多節點 parallel 任務只能在已安裝`ecs-init`套件的 Amazon Linux 執行個體上啟動的運算資源上執行。建議您在建立運算環境時使用預設的 Amazon ECS 最佳化 AMI。您可以透過不指定自訂 AMI 來執行此操作。如需詳細資訊，請參閱 多節點 parallel 工作。

- Amazon ECS 容器代理程式。建議您使用最新的 版本。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的安裝 Amazon ECS 容器代理程式。
- 啟動 Amazon ECS 容器代理程式時，必須使用`ECS_AVAILABLE_LOGGING_DRIVERS`環境變數將日誌驅動程式指定為可用的日誌驅動程式。`awslogs`如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的 Amazon ECS 容器代理程式組態。
- 運行至少版本 1.9 的 Docker 守護進程以及任何 Docker 運行時依賴項。如需詳細資訊，請參閱 Docker 文件中的檢查執行時間相依性。

> **Note**
>
> 我們建議使用隨附的 Docker 版本，並使用您正在使用的對應 Amazon ECS 代理程式版本進行測試。Amazon ECS 為 Amazon ECS 優化 AMI 的 Linux 變體提供了一個更改日誌。GitHub如需詳細資訊，請參閱變更記錄。

建議

- 用於執行和監控 Amazon ECS 代理程式的初始化和保姆程序。Amazon ECS 最佳化 AMI 使用`ecs-init`新啟動程序，而其他作業系統可能會使用。`systemd`如需詳細資訊和範例，請參閱 Amazon 彈性容器服務開發人員指南中的範例容器執行個體使用者資料組態指令碼。如需詳細資訊`ecs-init`，請參閱中的`ecs-init`專案 GitHub。受管運算環境至少需要 Amazon ECS 代理程式在開機時啟動。如果 Amazon ECS 代理程式未在您的運算資源上執行，則無法接受來自 AWS Batch的任務。

Amazon ECS 最佳化 AMI 已預先設定這些需求和建議。我們建議您使用 Amazon ECS 最佳化 AMI 或 Amazon Linux AMI 與為您的運算資源安裝的`ecs-init`套件搭配使用。如果您的應用程式需要特定作

業系統或 Docker 版本尚未在這些 AMI 中提供，請選擇其他 AMI。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 Amazon ECS 優化 AMI。

建立計算資源 AMI

您可以建立自己的自訂運算資源 AMI，以用於受管和未受管理的運算環境。如需說明，請參閱 運算資源 AMI 規格。然後，在建立自訂 AMI 之後，您可以建立使用該 AMI 的運算環境，您可以將工作佇列與之建立關聯。最後，開始將工作提交到該佇列。

建立自訂計算資源 AMI

1. 選擇一個基礎 AMI 開始。基礎 AMI 必須使用 HVM 虛擬化。基本 AMI 不能是視窗 AMI。

   > **Note**
   >
   > 您為運算環境選擇的 AMI 必須與您要用於該運算環境的執行個體類型架構相符。例如，如果您的運算環境使用A1執行個體類型，則您選擇的計算資源 AMI 必須支援Arm執行個體。Amazon ECS 出售 Amazon ECS x86 的兩個Arm版本優化 Amazon Linux 2 AMI。有關詳情，請參閱 Amazon ECS Amazon 彈性容器服務開發人員指南中的 Amazon ECS 優化亞馬遜 Linux 2 AMI。

   Amazon ECS 最佳化 Amazon Linux 2 AMI 是受管運算環境中運算資源的預設 AMI。Amazon ECS 最佳化的 Amazon Linux 2 AMI 已 AWS Batch 由 AWS 工程師預先設定和測試。這是一個最小的 AMI，您可以開始使用並獲得 AWS 快速運行的計算資源。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 Amazon ECS 最佳化 AMI。

   或者，您可以選擇另一個 Amazon Linux 2 變體，然後使用下列命令安裝`ecs-init`套件。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的在 Amazon Linux 2 EC2 執行個體上安裝 Amazon ECS 容器代理程式：

   ```
   $ sudo amazon-linux-extras disable docker
   $ sudo amazon-linux-extras install ecs-init
   ```

   例如，如果您想要在運 AWS Batch 算資源上執行 GPU 工作負載，您可以從 Amazon Linux 深度學習 AMI 著手。然後，配置 AMI 以運行 AWS Batch 作業。如需詳細資訊，請參閱 使用 GPU 工作負載 AMI。

> **Important**
>
> 您可以選擇不支持該`ecs-init`軟件包的基本 AMI。但是，如果您這樣做，則必須設定一種方法，以便在開機時啟動 Amazon ECS 代理程式並保持其執行。您也可以檢視數個用於啟動和監控 Amazon ECS 容器代理程式的使用`systemd`者資料組態指令碼範例。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的範例容器執行個體使用者資料組態指令碼

2. 使用適合 AMI 的儲存選項，從您選取的基礎 AMI 啟動執行個體。如果您選擇的執行個體類型支援，您可以設定連接的 Amazon EBS 磁碟區或執行個體儲存磁碟區的大小和數量。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的啟動執行個體和 Amazon EC2 執行個體存放區。
3. 使用 Connect 至您的執行個體，SSH並執行任何必要的設定工作。這可能包括以下任何或所有步驟：

   - 安裝 Amazon ECS 容器代理程式。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的安裝 Amazon ECS 容器代理程式。
   - 設定指令碼，以設置執行個體存放區磁碟區的格式。
   - 將執行個體存放區磁碟區或 Amazon EFS 檔案系統新增至`/etc/fstab`檔案，以便在開機時掛接。
   - 配置 Docker 選項，例如啟用調試或調整基本圖像大小。
   - 安裝套件或複製檔案。

   如需詳細資訊，請參閱 Amazon EC2 使用者指南中的使用安全殼層連線到 Linux 執行個體。
4. 如果您在執行個體上啟動 Amazon ECS 容器代理程式，則必須先停止該代理程式並移除任何持續性資料檢查點檔案，然後再建立 AMI。否則，如果您不這樣做，代理程式就不會在從 AMI 啟動的執行個體上啟動。

   a. 停用 Amazon ECS 容器代理程式。

      - Amazon ECS 最佳化 Amazon Linux 2 AMI：

        ```
        sudo systemctl stop ecs
        ```

      - Amazon ECS 最佳化 Amazon Linux AMI：

        ```
        sudo stop ecs
        ```

b. 移除持續性資料檢查點檔案。依預設，這些檔案位於目/var/lib/ecs/data/錄中。使用下面的命令刪除這些文件，如果有的話。

```
sudo rm -rf /var/lib/ecs/data/*
```

5. 從執行中的執行個體建立新的 AMI。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的建立支援 Linux AMI。

若要搭配使用您的新 AMI AWS Batch

1. 建立新 AMI 之後，使用新 AMI 建立運算環境。要做到這一點， 選擇圖像類型，然後在圖像 ID 中輸入自定義 AMI ID 建立 AWS Batch 計算環境時的取代方塊。欲了解更多信息， 請參閱the section called “使用 EC2 資源建立受管運算環境”。

> **Note**
>
> 您為運算環境選擇的 AMI 必須與您要用於該運算環境的執行個體類型架構相符。例如，如果您的運算環境使用A1執行個體類型，則您選擇的計算資源 AMI 必須支援Arm執行個體。Amazon ECS 出售 Amazon ECS x86 的兩個Arm版本優化 Amazon Linux 2 AMI。有關詳情，請參閱 Amazon ECS Amazon 彈性容器服務開發人員指南中的 Amazon ECS 優化亞馬遜 Linux 2 AMI。

2. 建立任務佇列，並與新的運算環境建立關聯。如需詳細資訊，請參閱 建立工作佇列。

> **Note**
>
> 與工作佇列相關聯的所有運算環境都必須共用相同的架構。 AWS Batch 不支援在單一工作佇列中混合運算環境架構類型。

3. (選用) 將範例任務提交到新的任務佇列。如需詳細資訊，請參閱工作定義範例、建立單一節點工作定義 及提交工作。

使用 GPU 工作負載 AMI

若要在您的 AWS Batch 運算資源上執行 GPU 工作負載，您必須搭配 GPU 支援來使用 AMI。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的在 Amazon ECS 和 Amazon ECS 最佳化 AMI 上使用 GPU。

在受管運算環境中，如果運算環境指定任何p2p3、p4、、p5、g3g3sg4、或g5執行個體類型或執行個體系列，則AWS Batch使用 Amazon ECS GPU 最佳化 AMI。

在非受管運算環境中，建議使用 Amazon ECS GPU 最佳化 AMI。您可以使用AWS Command Line Interface或AWS Systems Manager參數存放區GetParameter和GetParametersByPath操作來擷取建議的 Amazon ECS GPU 最佳化 AMI 的中繼資料。GetParameters

> **Note**
>
> p5執行個體系列僅支援與 Amazon ECS GPU 最佳化 AMI 相同或更新20230912的版本，而且與和執行個體類型不相p2容。g2如果您需要使用p5執行個體，請確定您的運算環境不包含p2或g2執行個體，並使用最新的預設 Batch AMI。建立新的計算環境將使用最新的 AMI，但如果您要將計算環境更新為包含p5，則可以將ComputeResource內容設定updateToLatestImageVersion為，以確保您使用的是最新true的 AMI。如需 GPU 執行個體的 AMI 相容性的詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的在 Amazon ECS 上使用 GPU。

下列範例顯示如何使用 GetParameter 命令。

AWS CLI

```
$ aws ssm get-parameter --name /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended \
                        --region us-east-2 --output json
```

輸出將 AMI 資訊包含在Value參數中。

```
{
    "Parameter": {
        "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended",
        "LastModifiedDate": 1555434128.664,
        "Value": "{\"schema_version\":1,\"image_name\":\"amzn2-ami-ecs-gpu-
hvm-2.0.20190402-x86_64-ebs\",\"image_id\":\"ami-083c800fe4211192f\",\"os\":\"Amazon
 Linux 2\",\"ecs_runtime_version\":\"Docker version 18.06.1-ce\",\"ecs_agent_version
\":\"1.27.0\"}",
        "Version": 9,
        "Type": "String",
        "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended"
```

```
    }
}
```

Python

```
from __future__ import print_function

import json
import boto3

ssm = boto3.client('ssm', 'us-east-2')

response = ssm.get_parameter(Name='/aws/service/ecs/optimized-ami/amazon-linux-2/
gpu/recommended')
jsonVal = json.loads(response['Parameter']['Value'])
print("image_id   = " + jsonVal['image_id'])
print("image_name = " + jsonVal['image_name'])
```

這時輸出只會包含 AMI ID 和 AMI 名稱：

```
image_id   = ami-083c800fe4211192f
image_name = amzn2-ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs
```

下面的實例演示了使用 GetParameters.

AWS CLI

```
$ aws ssm get-parameters --names  /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended/image_name \
                                  /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended/image_id \
                          --region us-east-2 --output json
```

這時輸出會包含個別參數的完整中繼資料：

```
{
    "InvalidParameters": [],
    "Parameters": [
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_id",
```

```
            "LastModifiedDate": 1555434128.749,
            "Value": "ami-083c800fe4211192f",
            "Version": 9,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_id"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_name",
            "LastModifiedDate": 1555434128.712,
            "Value": "amzn2-ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs",
            "Version": 9,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_name"
        }
    ]
}
```

Python

```
from __future__ import print_function

import boto3

ssm = boto3.client('ssm', 'us-east-2')

response = ssm.get_parameters(
            Names=['/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_name',
                   '/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_id'])
for parameter in response['Parameters']:
    print(parameter['Name'] + " = " + parameter['Value'])
```

輸出包括 AMI ID 和 AMI 名稱，使用名稱的完整路徑。

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_id =
 ami-083c800fe4211192f
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_name = amzn2-
ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs
```

下列範例顯示如何使用 GetParametersByPath 命令。

AWS CLI

```
$ aws ssm get-parameters-by-path --path /aws/service/ecs/optimized-ami/amazon-
linux-2/gpu/recommended \
                                  --region us-east-2 --output json
```

輸出包括指定路徑下所有參數的完整中繼資料。

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
ecs_agent_version",
            "LastModifiedDate": 1555434128.801,
            "Value": "1.27.0",
            "Version": 8,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/ecs_agent_version"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
ecs_runtime_version",
            "LastModifiedDate": 1548368308.213,
            "Value": "Docker version 18.06.1-ce",
            "Version": 1,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/ecs_runtime_version"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_id",
            "LastModifiedDate": 1555434128.749,
            "Value": "ami-083c800fe4211192f",
            "Version": 9,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_id"
        },
        {
```

```
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_name",
            "LastModifiedDate": 1555434128.712,
            "Value": "amzn2-ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs",
            "Version": 9,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_name"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
os",
            "LastModifiedDate": 1548368308.143,
            "Value": "Amazon Linux 2",
            "Version": 1,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/os"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
schema_version",
            "LastModifiedDate": 1548368307.914,
            "Value": "1",
            "Version": 1,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/schema_version"
        }
    ]
}
```

Python

```
from __future__ import print_function

import boto3

ssm = boto3.client('ssm', 'us-east-2')

response = ssm.get_parameters_by_path(Path='/aws/service/ecs/optimized-ami/amazon-
linux-2/gpu/recommended')
for parameter in response['Parameters']:
```

```
    print(parameter['Name'] + " = " + parameter['Value'])
```

輸出包括指定路徑中所有參數名稱的值，並使用名稱的完整路徑。

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/ecs_agent_version =
 1.27.0
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/ecs_runtime_version =
 Docker version 18.06.1-ce
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_id =
 ami-083c800fe4211192f
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_name = amzn2-
ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/os = Amazon Linux 2
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/schema_version = 1
```

如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的擷取 Amazon ECS 最佳化 AMI 中繼資料。

Amazon 棄用

Amazon Linux AMI（也稱為 Amazon Linux 1）在 2023 年 12 月 31 日達到其使用壽命終止。 AWS Batch 已經結束對 Amazon Linux AMI 的支援，因為自 2024 年 1 月 1 日起，它將不會收到任何安全性更新或錯誤修正。如需有關 Amazon Linux 的詳細資訊 end-of-life，請參閱 AL 常見問題集。

我們建議您將現有以 Amazon Linux 為基礎的運算環境更新至 Amazon Linux 2023，以防止意外的工作負載中斷，並繼續接收安全性和其他更新。

您使用 Amazon Linux AMI 的運算環境可能會在 2023 年 12 月 31 end-of-life 日之後繼續運作。不過，這些運算環境將不再接收來自的任何新軟體更新、安全性修補程式或錯誤修正 AWS。之後，您有責任在 Amazon Linux AMI 上維護這些運算環境 end-of-life。我們建議將 AWS Batch 運算環境遷移到 Amazon Linux 2023 或 Amazon Linux 2，以維持最佳的性能和安全性。

有關 AWS Batch 從 Amazon Linux AMI 遷移到 Amazon 2023 或 Amazon Linux 2 的幫助，請參閱更新運算環境- AWS Batch。

啟動範本支援

AWS Batch 支援將 Amazon EC2 啟動範本與您的 EC2 運算環境搭配使用。使用啟動範本，您可以修改 AWS Batch 運算資源的預設組態，而無需建立自訂 AMI。

> **Note**
>
> AWS Fargate 資源不支援啟動範本。

您必須先建立啟動範本，才能將它與運算環境建立關聯。您可以在 Amazon EC2 主控台中建立啟動範本。或者，您可以使用 AWS CLI 或 AWS SDK。例如，下列 JSON 檔案代表一個啟動範本，該範本會針對預設 AWS Batch 計算資源 AMI 調整 Docker 資料磁碟區的大小，並將其設定為加密。

```
{
    "LaunchTemplateName": "increase-container-volume-encrypt",
    "LaunchTemplateData": {
        "BlockDeviceMappings": [
            {
                "DeviceName": "/dev/xvda",
                "Ebs": {
                    "Encrypted": true,
                    "VolumeSize": 100,
                    "VolumeType": "gp2"
                }
            }
        ]
    }
}
```

您可以透過將 JSON 儲存到呼叫的檔案`lt-data.json`並執行下列 AWS CLI 命令，以建立先前的啟動範本。

```
aws ec2 --region <region> create-launch-template --cli-input-json file://lt-data.json
```

如需有關啟動範本的詳細資訊，請參閱 Amazon EC2 使用者指南中的從啟動範本啟動執行個體。

如果您使用啟動範本建立您的運算環境，您可以將以下現有的運算環境參數移至您的啟動範本：

> **Note**
>
> 假設這些參數中的任何一個 (Amazon EC2 標籤除外) 都在啟動範本和運算環境組態中指定。然後，計算環境參數優先。Amazon EC2 標籤會在啟動範本和運算環境組態之間合併。如果標籤的鍵發生衝突，則計算環境配置中的值優先。

- Amazon EC2 key pair
- Amazon EC2 AMI ID
- 安全群組 ID
- Amazon EC2 標籤

下列啟動範本參數會被忽略 AWS Batch：

- 執行個體類型 (在您建立運算環境時，指定所需的執行個體類型)
- 執行個體角色 (在您建立運算環境時，指定所需的執行個體角色)
- 網路界面子網路 (在您建立運算環境時，指定所需的子網路)
- 執行個體市場選項 (AWS Batch 必須控制 Spot 執行個體組態
- 停用 API 終止 (AWS Batch 必須控制執行個體生命週期)

AWS Batch 只會在基礎結構更新期間，以新的啟動範本版本更新啟動範本。如需詳細資訊，請參閱 更新運算環境。

啟動範本中的 Amazon EC2 使用者資料

您可以在執行個體啟動時由雲端初始化執行的啟動範本中提供 Amazon EC2 使用者資料。您的使用者資料可以執行常見的設定案例，包括但不限於下列各項：

- 包括使用者或群組
- 安裝套件
- 建立分區和檔案系統

啟動範本中的 Amazon EC2 使用者資料必須採用 MIME 多部分封存格式。這是因為您的使用者資料會與設定運算資源所需的其他 AWS Batch 使用者資料合併。您可以將多個使用者資料區塊組合在一起成為單一 MIME 分段檔案。例如，您可能想要將設定 Docker 精靈的雲端啟動器與寫入 Amazon ECS 容器代理程式組態資訊的使用者資料殼層指令碼結合在一起。

如果您正在使用 AWS CloudFormation，則可以將該AWS::CloudFormation::Init類型與 cfn-init 輔助程序腳本一起使用，以執行常見的配置方案。

MIME 分段檔案包含下列元件：

- 內容類型和部分邊界宣告：`Content-Type: multipart/mixed; boundary="==BOUNDARY=="`
- MIME 版本宣告：`MIME-Version: 1.0`
- 包含下列元件的一或多個使用者資料區塊：
  - 表示用戶數據塊的開始的開口邊界：`--==BOUNDARY==`。您必須將此邊界前的直線保持空白。
  - 區塊的內容類型宣告：`Content-Type: text/cloud-config; charset="us-ascii"`。如需內容類型的詳細資訊，請參閱「Cloud-Init 說明文件」。您必須將內容類型宣告之後的行保留為空白。
  - 用戶數據的內容，例如 shell 命令或`cloud-init`指令列表。
- 表示 MIME 多部分檔案結尾的閉合邊界：`--==BOUNDARY==--`。您必須將封閉邊界前的直線保持空白。

以下是可用來建立自己的 MIME 多部分檔案的範例。

> **Note**
>
> 如果您將使用者資料新增至 Amazon EC2 主控台中的啟動範本，則可以將其以純文字形式貼入。或者，您可以從文件上傳它。如果您使用 AWS CLI 或 AWS SDK，則必須先`base64`對使用者資料進行編碼，然後在呼叫 Tem CreateLaunchplate 時將該字串提交為`UserData`參數值，如此 JSON 檔案所示。
>
> ```
> {
>     "LaunchTemplateName": "base64-user-data",
>     "LaunchTemplateData": {
>         "UserData":
>  "ewogICAgIkxhdW5jaFRlbXBsYXRlTmFtZSI6ICJpbmNyZWFzZS1jb250YWluZXItdm9sdW..."
>     }
> }
> ```

範例

- 範例：掛載現有的 Amazon EFS 檔案系統
- 範例：覆寫預設的 Amazon ECS 容器代理程式組態
- 範例：為 Lustre 檔案系統掛載現有的亞馬遜 FSx

範例：掛載現有的 Amazon EFS 檔案系統

Example

此範例 MIME 多部分檔案會將運算資源設定為安裝amazon-efs-utils套件，並在上掛接現有的 Amazon EFS 檔案系統。/mnt/efs

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
- amazon-efs-utils

runcmd:
- file_system_id_01=fs-abcdef123
- efs_directory=/mnt/efs

- mkdir -p ${efs_directory}
- echo "${file_system_id_01}:/ ${efs_directory} efs tls,_netdev" >> /etc/fstab
- mount -a -t efs defaults

--==MYBOUNDARY==--
```

範例：覆寫預設的 Amazon ECS 容器代理程式組態

Example

此範例 MIME 多段檔案會覆寫運算資源預設的 Docker 影像清除設定。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
echo ECS_IMAGE_CLEANUP_INTERVAL=60m >> /etc/ecs/ecs.config
echo ECS_IMAGE_MINIMUM_CLEANUP_AGE=60m >> /etc/ecs/ecs.config

--==MYBOUNDARY==--
```

範例：為 Lustre 檔案系統掛載現有的亞馬遜 FSx

Example

此範例 MIME 多部分檔案會設定計算資源，以便從「額外程式庫」安裝`lustre2.10`套件，並掛載 Lustre 檔案系統的現有 FSx，以及的掛載名稱。`/scratch` `fsx`這個例子是針對 Amazon Linux 2。如需其他 Linux 發行版的安裝說明，請參閱《安裝 Lustre 用戶端》中的《Amazon FSx for Lu stre 戶端使用者指南》。如需詳細資訊，請參閱 Amazon FSx 使用者指南中的自動掛載 Amazon FSx for Lustre 檔案系統。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

runcmd:
- file_system_id_01=fs-0abcdef1234567890
- region=us-east-2
- fsx_directory=/scratch
- amazon-linux-extras install -y lustre2.10
- mkdir -p ${fsx_directory}
- mount -t lustre ${file_system_id_01}.fsx.${region}.amazonaws.com@tcp:fsx
 ${fsx_directory}

--==MYBOUNDARY==--
```

在容器內容的磁碟區和 mountPoints 成員中，掛載點必須對應到容器中。

```
{
    "volumes": [
        {
            "host": {
                "sourcePath": "/scratch"
            },
            "name": "Scratch"
        }
    ],
    "mountPoints": [
        {
            "containerPath": "/scratch",
            "sourceVolume": "Scratch"
```

```
        }
    ],
}
```

建立運算環境

您必須先建立運算環境 AWS Batch，才能在中執行工作。您可以建立受管運算環境，在其中根據您的規格 AWS Batch 管理環境中的 Amazon EC2 執行個體或 AWS Fargate 資源。或者，您也可以建立非受管運算環境，在其中處理環境中的 Amazon EC2 執行個體組態。

> **Important**
>
> 在下列情況下，不支援 Fargate 競價型執行個體：
>
> - 在具有 ARM64 架構的 Amazon Linux 容器上。
> - Windows containers on AWS Fargate
>
> 如果將工作提交至僅使用 Fargate Spot 運算環境的工作佇列，則在這些情況下，工作佇列將會遭到封鎖。

內容

使用 AWS Fargate 資源建立受管理的運算環境

1. 請在以下位置開啟 AWS Batch 主控台。 https://console.aws.amazon.com/batch/
2. 從導覽列中，選取 AWS 區域 要使用的。
3. 在導覽窗格中，選擇 Compute environments (運算環境)。
4. 選擇建立。
5. 設定運算環境。

> **Note**
>
> Windows containers on AWS Fargate 工作的運算環境至少必須有一個 vCPU。

a. 針對 [運算環境] 組態，選擇 [Fargate]。

b. 對於「名稱」，請為您的計算環境指定唯一的名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。

c. 針對服務角色，請選擇可讓 AWS Batch 服務代表您呼叫所需 AWS API 作業的服務連結角色。例如，選擇 AWSServiceRoleForBatch。如需詳細資訊，請參閱 服務連結角色權限 AWS Batch。

d. (選擇性) 展開標籤。若要新增標籤，請選擇 Add tag (新增標籤)。然後，輸入「金鑰」名稱和選用「值」。選擇 Add tag (新增標籤)。

e. 選擇 [下一頁]。

6. 在「執行個體組態」區段中：

a. (選擇性) 若要使用 Fargate 定點容量，請開啟 Fargate 定點。如需 Fargate 點的相關資訊，請參閱使用 Amazon EC2 現貨和遠端點。

b. 針對 vCPUs 數目上限，無論工作佇列需求為何，都可以選擇運算環境可向外擴充至的 vCPUs 數目上限。

c. 選擇 [下一頁]。

7. 設定網路。

> **Important**
>
> 運算資源需要存取，才可以與 Amazon ECS 服務端點通訊。可透過介面 VPC 端點或透過具備公有 IP 地址的運算資源來實現。
> 如需介面 VPC 端點的詳細資訊，請參閱 Amazon Elastic Container Service 開發人員指南中的 Amazon ECS 介面 VPC 端點 (AWS PrivateLink)。
> 如果您沒有設定介面 VPC 端點，且運算資源沒有公有 IP 地址，則它們必須使用網路地址轉譯 (NAT) 來提供此存取。如需詳細資訊，請參閱 Amazon VPC 使用者指南中的 NAT 閘道。如需詳細資訊，請參閱 the section called “建立 VPC”。

a. 針對 V irtual Private Cloud (VPC) (VPC) ID，請選擇您要啟動執行個體的 VPC。

b. 對於子網路，請選擇要使用的子網路。依預設，所選 VPC 內的所有子網路均可使用。

> **Note**
>
> AWS Batch 在 Fargate 目前不支持 Local Zones。如需詳細資訊，請參閱本機區域、Wavelength 區域和 Amazon 彈性容器服務開發人員指南 AWS Outposts中的 Amazon ECS 叢集。

c. 在 Security groups (安全群組) 中，選擇連接至您的執行個體的安全群組。根據預設，會選擇您的 VPC 預設的安全群組。

d. 選擇 [下一頁]。

8. 對於「檢閱」，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成後，請選擇 [建立運算環境]。

使用 EC2 資源建立受管運算環境

1. 請在以下位置開啟 AWS Batch 主控台。 https://console.aws.amazon.com/batch/
2. 從導覽列中，選取 AWS 區域 要使用的。
3. 在導覽窗格中，選擇 Compute environments (運算環境)。
4. 選擇建立。
5. 設定環境。

   a. 對於運算環境組態，請選擇亞馬遜彈性運算雲端 (Amazon EC2)。

   b. 針對協調類型，選擇受管理。

   c. 對於「名稱」，請為您的計算環境指定唯一的名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。

   d. (選用) 針對服務角色，請選擇服務連結角色，讓 AWS Batch 服務代表您呼叫所需 AWS API 作業。例如，選擇 AWSServiceRoleForBatch。如需詳細資訊，請參閱 服務連結角色權限 AWS Batch。

   e. 在 Instance role (執行個體角色) 中，選擇建立新的執行個體描述檔，或使用附有所需 IAM 許可的現有執行個體描述檔。此執行個體設定檔允許為您的運算環境建立的 Amazon ECS 容器執行個體代表您呼叫所需的 AWS API 操作。如需詳細資訊，請參閱 Amazon ECS 執行個體角色。如果您選擇建立新的執行個體描述檔，會為您建立所需的角色 (`ecsInstanceRole`)。

   f. (選擇性) 展開標籤。

g. (選用) 對於 EC2 標籤，請選擇 [新增標籤]，將標籤新增至在運算環境中啟動的資源。然後，輸入「金鑰」名稱和選用「值」。選擇 Add tag (新增標籤)。

h. (選擇性) 針對「標籤」，選擇「新增標籤」。然後，輸入「金鑰」名稱和選用「值」。選擇 Add tag (新增標籤)。

   如需詳細資訊，請參閱 標記您的 AWS Batch 資源。

i. 選擇 [下一頁]。

6. 在「執行個體組態」區段中：

a. (選擇性) 若要啟用使用 Spot 執行個體，請開啟 Spot。如需詳細資訊，請參閱 Spot 執行個體。

b. (僅限競價型) 對於隨需價格上限百分比，請選擇競價型執行個體價格與執行個體啟動前該執行個體類型的隨需價格相比的最高百分比。例如，如果您的最高價為 20%，則 Spot 價格必須低於該 EC2 執行個體目前隨需價格的 20%。您一律會支付最低價 (市價) 且絕不超過您的最大百分比。如果您將此欄位空，預設值是隨需價格的 100%。

c. (僅限競價型) 對於競價型叢集角色，請選擇現有的 Amazon EC2 Spot 叢集 IAM 角色以套用至您的競價型運算環境。如果您還沒有現有的 Amazon EC2 競價型叢集 IAM 角色，則必須先建立一個角色。如需詳細資訊，請參閱 Amazon EC2 現貨叢集角色。

> **Important**
>
> 若要在建立時標記競價型執行個體，您的 Amazon EC2 競價型叢集 IAM 角色必須使用較新的 AmazonEC2 SpotFleet TaggingRole 受管政策。AmazonEC2 SpotFleet 角色受管政策沒有標記 Spot 執行個體所需的權限。如需詳細資訊，請參閱 建立時未標記競價型執行個體 及 the section called “標記您的 資源”。

d. 對於最小 vCPUs 數量，無論工作佇列需求為何，請選擇運算環境維護的 vCPUs 數目下限。

e. 對於所需的 vCPUs，請選擇您的運算環境啟動時使用的 vCPUs 數量。隨著任務佇列需求增加， AWS Batch 可以增加運算環境的所需 vCPU 數，並新增 EC2 執行個體 (最多達最大 vCPU 數)。隨著需求減少， AWS Batch 可以減少運算環境的所需 vCPU 數，並移除執行個體 (最少可達最小 vCPU 數)。

f. 針對 vCPUs 數目上限，無論工作佇列需求為何，都可以選擇運算環境可向外擴充至的 vCPUs 數目上限。

g. 針對允許的執行個體類型，選擇可啟動的 Amazon EC2 執行個體類型。您可以指定例證族群以啟動這些族群中的任何例證類型 (例如c5c5n、或p3)。或者，您可以指定族群內的特定大

小 (例如c5.8xlarge)。金屬例證類型不在例證族群中。例如，c5不包括c5.metal. 您也可optimal以選擇選取符合工作佇列需求的R4執行個體類型 (從M4、和執行個體系列)。C4

> **Note**
>
> 在建立運算環境時，您為其選取的執行個體類型必須共用相同架構。例如，您無法在相同的運算環境中混合使用 x86 和 ARM 執行個體。

> **Note**
>
> AWS Batch 會根據工作佇列中所需的數量來調整 GPU 的規模。若要使用 GPU 排程，運算環境必須包含p2、、、、、p3p4p5g3g3s、g4或g5系列中的執行個體類型。

> **Note**
>
> 目前，optimal使用、和例證族群M4中的C4R4例證類型。如果沒有來自這些例證族群的例證類型，則會使用C5M5、和R5例證族群中 AWS 區域 的例證類型。

h. 展開 Additional configuration (其他組態)。

i. (選擇性) 在「放置」群組中，輸入放置群組名稱，以將計算環境中的資源分組。

j. (選擇性) 對於 EC2 金鑰組，請在連線至執行個體時選擇公開金鑰和私密金鑰組做為安全登入資料。如需有關 Amazon EC2 金鑰配對的詳細資訊，請參閱 Amazon EC2 金鑰配對和 Linux 執行個體。

k. 如為配置策略，從允許的執行個體類型清單中選取執行個體類型時，選取要使用的配置策略。對於 EC2 隨需運算環境、最佳容量優化以及針對 EC2 競價型運算環境優化的最佳選擇，最佳化通常是最佳選擇。如需詳細資訊，請參閱 the section called “分配策略”。

l. (選擇性) 對於 EC2 組態，請選擇映像類型和映像 ID 覆寫值，以提供資訊， AWS Batch 以便為運算環境中的執行個體選取 Amazon 機器映像 (AMI)。如果未針對每個映像類型指定影像 ID 覆寫，請 AWS Batch 選取最近的 Amazon ECS 最佳化 AMI。如果未指定映像類型，則對於非 GPU、非重力執行個 AWS 體，預設值為 Amazon Linux 2。

> **Important**
>
> 若要使用自訂 AMI，請選擇映像類型，然後在 [映像 ID 覆寫] 方塊中輸入自訂 AMI ID。

Amazon Linux 2

所有以 AWS 重力為基礎的執行個體系列 (例如、、和T4g) 的預設值 C6g M6gR6g，並且可用於所有非 GPU 執行個體類型。

Amazon Linux 2 (GPU)

所有 GPU 執行個體系列的預設值 (例如P4和G4)，可用於所有非 AWS 重力型執行個體類型。

Amazon Linux

可用於非 GPU、非重 AWS 力子執行個體系列。Amazon Linux AMI 的標準支持已經結束。如需詳細資訊，請參閱 Amazon Linux AMI。

> **Note**
>
> 您為運算環境選擇的 AMI 必須與您要用於該運算環境的執行個體類型架構相符。例如，如果您的運算環境使用A1執行個體類型，則您選擇的計算資源 AMI 必須支援Arm執行個體。Amazon ECS 出售 Amazon ECS x86 的兩個Arm版本優化 Amazon Linux 2 AMI。有關詳情，請參閱 Amazon ECS Amazon 彈性容器服務開發人員指南中的 Amazon ECS 優化亞馬遜 Linux 2 AMI。

m. (選擇性) 對於啟動範本，請選取現有的 Amazon EC2 啟動範本來設定您的運算資源。系統會自動填入範本的預設版本。如需詳細資訊，請參閱 啟動範本支援。

> **Note**
>
> 在啟動範本中，您可以指定您建立的自訂 AMI。

n. (選用) 對於 Launch template version (啟動範本版本)，請輸入 `$Default`、`$Latest` 或指定要使用的版本號碼。

> **Important**
>
> 如果啟動範本的 version 參數為`$Default`或`$Latest`，則會在基礎結構更新期間評估指定啟動範本的預設或最新版本。如果預設選取了不同的 AMI ID，或選取了啟動範本的最新版本，則會在更新中使用該 AMI ID。如需詳細資訊，請參閱 the section called “更新 AMI 識別碼”。

o. 選擇 [下一頁]。

7. 在「網路設定」區段中：

> **Important**
>
> 運算資源需要存取，才可以與 Amazon ECS 服務端點通訊。可透過介面 VPC 端點或透過具備公有 IP 地址的運算資源來實現。
> 如需介面 VPC 端點的詳細資訊，請參閱 Amazon Elastic Container Service 開發人員指南中的 Amazon ECS 介面 VPC 端點 (AWS PrivateLink)。
> 如果您沒有設定介面 VPC 端點，且運算資源沒有公有 IP 地址，則它們必須使用網路地址轉譯 (NAT) 來提供此存取。如需詳細資訊，請參閱 Amazon VPC 使用者指南中的 NAT 閘道。如需詳細資訊，請參閱 the section called “建立 VPC”。

a. 對於 V irtual Private Cloud (VPC) (VPC) ID，請選擇要啟動執行個體的 VPC。

b. 對於子網路，請選擇要使用的子網路。依預設，所選 VPC 內的所有子網路均可使用。

> **Note**
>
> AWS Batch 在 Amazon EC2 上支持 Local Zones。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的 L ocal Zones 機區域和本機區域、Wavelength 區域中的 Amazon ECS 叢集和 AWS Outposts Amazon 彈性容器服務開發人員指南。

c. (選擇性) 針對安全性群組，請選擇要連結至執行個體的安全性群組。根據預設，會選擇您的 VPC 預設的安全群組。

8. 選擇 [下一頁]。

9. 對於「檢閱」，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成後，請選擇 [建立運算環境]。

使用 EC2 資源建立非受管運算環境

1. 請在以下位置開啟 AWS Batch 主控台。 https://console.aws.amazon.com/batch/
2. 從導覽列中，選取 AWS 區域 要使用的。
3. 在 [運算環境] 頁面上，選擇 [建立]。
4. 設定環境。
   a. 對於運算環境組態，請選擇亞馬遜彈性運算雲端 (Amazon EC2)。
   b. 針對協調類型，選擇 [未受管理]。
5. 對於「名稱」，請為您的計算環境指定唯一的名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
6. (選用) 針對服務角色，請選擇可讓 AWS Batch 服務代表您呼叫所需 AWS API 作業的角色。例如，選擇 AWSBatchServiceRole。如需詳細資訊，請參閱 the section called “使用服務連結角色”。
7. 針對 vCPUs 數目上限，無論工作佇列需求為何，都可以選擇運算環境可向外擴充至的 vCPUs 數目上限。
8. (選擇性) 展開標籤。若要新增標籤，請選擇 Add tag (新增標籤)。然後，輸入「金鑰」名稱和選用「值」。選擇 Add tag (新增標籤)。如需詳細資訊，請參閱 標記您的 AWS Batch 資源。
9. 選擇 [下一頁]。
10. 對於「檢閱」，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成後，請選擇 [建立運算環境]。

使用 Amazon EKS 資源建立受管運算環境

1. 請在以下位置開啟 AWS Batch 主控台。 https://console.aws.amazon.com/batch/
2. 從導覽列中，選取 AWS 區域 要使用的。
3. 在導覽窗格中，選擇 Compute environments (運算環境)。
4. 選擇建立。
5. 對於運算環境組態，請選擇 Amazon Elastic Kubernetes Service (Amazon EKS)。
6. 對於「名稱」，請為您的計算環境指定唯一的名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。
7. 針對執行個體角色，請選擇已連接所需 IAM 許可的現有執行個體設定檔。

> **Note**
>
> 若要在 AWS Batch 主控台中建立運算環境，請選擇具有`eks:ListClusters`和`eks:DescribeCluster`權限的執行個體設定檔。

8. 對於 EKS 叢集，請選擇現有的 Amazon EKS 叢集。
9. 在命名空間中，輸入Kubernetes命名空間以將您的 AWS Batch 處理序分組到叢集中。
10. (選擇性) 展開標籤。選擇「新增標籤」，然後輸入鍵值配對。
11. 選擇 [下一頁]。
12. (選用) 對於使用 EC2 競價型執行個體，請開啟啟用使用競價型執行個體以使用 Amazon EC2 競價型執行個體。
13. (僅限競價型) 對於隨需價格上限百分比，請選擇競價型執行個體價格與執行個體啟動前該執行個體類型的隨需價格相比的最高百分比。例如，如果您的最高價為 20%，則 Spot 價格必須低於該 EC2 執行個體目前隨需價格的 20%。您一律會支付最低價 (市價) 且絕不超過您的最大百分比。如果您將此欄位空，預設值是隨需價格的 100%。
14. (僅限競價型) 對於競價型叢集角色，請為SPOT運算環境選擇 Amazon EC2 競價型叢集 IAM 角色。

> **Important**
>
> 如果將配置策略設定為`BEST_FIT`或未指定，則需要此角色。

15. (選擇性) 對於最低 vCPUs 數量，無論工作佇列需求為何，都可以選擇運算環境維護的 vCPUs 數目下限。
16. (選擇性) 對於 vCPUs 數目上限，無論工作佇列需求為何，都可選擇運算環境可擴充至的 vCPUs 數目上限。
17. 針對允許的執行個體類型，選擇可啟動的 Amazon EC2 執行個體類型。您可以指定例證族群以啟動這些族群中的任何例證類型 (例如`c5`c5n、或p3)。或者，您可以指定族群內的特定大小 (例如，`c5.8xlarge`)。金屬例證類型不在例證族群中。例如，`c5`不包括`c5.metal`. 您也可以選擇選`optimal`取執行個體類型 (從C4M4、和R4執行個體系列)，因為您需要符合工作佇列需求的執行個體類型。

> **Note**
>
> 在建立運算環境時，您為其選取的執行個體類型必須共用相同架構。例如，您無法在相同的運算環境中混合使用 x86 和 ARM 執行個體。

> **Note**
>
> AWS Batch 根據工作佇列中所需的數量調整 GPU 的比例。若要使用 GPU 排程，運算環境必須包含p2、、、、、p3p4p5g3g3s、g4或g5系列中的執行個體類型。

> **Note**
>
> 目前，optimal 使用 C4、M4 和 R4 執行個體系列中的執行個體類型。如果沒有來自這些例證族群的例證類型，則會使用C5M5、和R5例證族群中 AWS 區域 的例證類型。

18. (選擇性) 展開其他組態。

    a. (選擇性) 在「放置」群組中，輸入放置群組名稱，以將計算環境中的資源分組。

    b. 對於「配置策略」，請選擇「最佳 _ 漸進」。

    c. （可選）對於 Amazon 機器映像（AMI）配置，請選擇添加亞馬遜機器映像（amis）配置。然後，選擇「影像類型」、輸入「影像 ID 取代」，然後輸入Kubernetes版本。

    > **Important**
    >
    > 若要使用自訂 AMI，請選擇映像類型，然後在 [映像 ID 覆寫] 方塊中輸入自訂 AMI ID。

    > **Note**
    >
    > 如果未針對每個映像類型指定影像 ID 覆寫，請 AWS Batch 選取最近的 Amazon ECS 最佳化 AMI。如果未指定映像類型，則對於非 GPU、非重力執行個 AWS 體，預設值為 Amazon Linux 2。

Amazon Linux 2

所有以 AWS 重力為基礎的執行個體系列 (例如、、和T4g) 的預設值 C6g M6gR6g，並且可用於所有非 GPU 執行個體類型。

Amazon Linux 2 (GPU)

所有 GPU 執行個體系列的預設值 (例如P4和G4)，可用於所有非 AWS 重力型執行個體類型。

d. (選擇性) 對於 Launch 範本，請選擇現有的啟動範本。

e. (選擇性) 對於 Launch 範本版本**$Default**，請輸入**$Latest**、或版本號碼。

19. 選擇 [下一頁]。
20. 對於 V irtual Private Cloud (VPC) (VPC) ID，請選擇要啟動執行個體的 VPC。
21. 對於子網路，請選擇要使用的子網路。依預設，所選 VPC 內的所有子網路均可使用。

> **Note**
>
> AWS Batch 在 Amazon EKS 支持 Local Zones。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的 Amazon EKS 和 L AWS ocal Zones。

22. (選擇性) 針對安全性群組，請選擇要連結至執行個體的安全性群組。依預設，會選取 VPC 的預設安全性群組。
23. 選擇 [下一頁]。
24. 對於「檢閱」，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成後，請選擇 [建立運算環境]。

運算環境範本

下列範例顯示空的計算環境範本。您可以使用此範本來建立您的運算環境，隨後可儲存至檔案並搭配 AWS CLI`--cli-input-json` 選項使用。如需這些參數的詳細資訊，請參閱 AWS BatchAPI 參考CreateComputeEnvironment中的。

```
{
    "computeEnvironmentName": "",
    "type": "UNMANAGED",
    "state": "DISABLED",
```

```
    "unmanagedvCpus": 0,
    "computeResources": {
        "type": "EC2",
        "allocationStrategy": "BEST_FIT_PROGRESSIVE",
        "minvCpus": 0,
        "maxvCpus": 0,
        "desiredvCpus": 0,
        "instanceTypes": [
            ""
        ],
        "imageId": "",
        "subnets": [
            ""
        ],
        "securityGroupIds": [
            ""
        ],
        "ec2KeyPair": "",
        "instanceRole": "",
        "tags": {
            "KeyName": ""
        },
        "placementGroup": "",
        "bidPercentage": 0,
        "spotIamFleetRole": "",
        "launchTemplate": {
            "launchTemplateId": "",
            "launchTemplateName": "",
            "version": ""
        },
        "ec2Configuration": [
            {
                "imageType": "",
                "imageIdOverride": "",
                "imageKubernetesVersion": ""
            }
        ]
    },
    "serviceRole": "",
    "tags": {
        "KeyName": ""
    },
    "eksConfiguration": {
        "eksClusterArn": "",
```

```
        "kubernetesNamespace": ""
    }
}
```

> **Note**
>
> 您可以使用下列AWS CLI指令產生先前的計算環境範本。
>
> ```
> $ aws batch create-compute-environment --generate-cli-skeleton
> ```

運算環境參數

運算環境分為幾個基本元件：運算環境的名稱、類型和狀態、運算資源定義 (如果是受管運算環境)、Amazon EKS 組態 (如果使用 Amazon EKS 資源)、用於提供 IAM 許可的服務角色 AWS Batch，以及運算環境的標籤。

主題

運算環境名稱

`computeEnvironmentName`

您的運算環境名稱。名稱最多可包含 128 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。

類型：字串

必要：是

Type

`type`

運算環境類型。選`MANAGED`擇 AWS Batch 管理您定義的 EC2 或 Fargate 運算資源。如需詳細資訊，請參閱 運算資源。選擇`UNMANAGED`管理您自己的 EC2 運算資源。

類型：字串

有效值：`MANAGED` | `UNMANAGED`

必要：是

State

`state`

運算環境狀態。

如果狀態為`ENABLED`，則 AWS Batch 排程器會嘗試在環境中放置工作。這些工作來自計算資源上的關聯工作佇列。如果管理運算環境，則執行個體會根據工作佇列需求自動向外擴充或擴充。

如果狀態為`DISABLED`，則 AWS Batch 排程器不會嘗試在環境中放置工作。處於`STARTING`或`RUNNING`狀態的工作會繼續正常進行。處於此`DISABLED`狀態的受管理運算環境不會向外延展。

> **Note**
>
> 處於某個`DISABLED`狀態的運算環境可能會繼續產生帳單費用。若要避免額外費用，請關閉然後刪除運算環境。有關更多信息，請參閱AWS Billing 用戶指南DeleteComputeEnvironment中的 AWS Batch API 參考和避免意外費用中的。

執行個體閒置時，例證會縮小至該`minvCpus`值。不過，執行個體大小不會變更。例如，假設`c5.8xlarge`執行個體的`minvCpus`值為`4`且`desiredvCpus`值為`36`。此執行個體不會縮減為`c5.large`執行個體。

類型：字串

有效值：`ENABLED` | `DISABLED`

必要：否

運算資源

`computeResources`

由運算環境管理的運算資源詳細資訊。如需詳細資訊，請參閱 運算環境。

類型：ComputeResource 物件

必要：受管理的運算環境需要此參數

`type`

運算環境類型。您可以選擇使用 EC2 隨需執行個體 (EC2) 和 EC2 競價型執行個體 (SPOT)，或在受管運算環境中使用 Fargate 容量 (FARGATEFARGATE_SPOT) 和遠門競價型容量 ()。如果您選擇 SPOT，您還必須使用 `spotIamFleetRole` 參數指定 Amazon EC2 Spot Fleet 角色。如需詳細資訊，請參閱 Amazon EC2 現貨叢集角色。

有效值：`EC2 | SPOT | FARGATE | FARGATE_SPOT`

必要：是

`allocationStrategy`

如果沒有足夠的最適合 EC2 執行個體類型的執行個體配置策略可用於運算資源。這可能是因為 AWS 區域 或 Amazon EC2 服務限制中的執行個體類型可用性所致。如需詳細資訊，請參閱 分配策略。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

`BEST_FIT` (default)

AWS Batch 選取最符合工作需求的執行個體類型，並選取具有最低成本執行個體類型偏好設定的執行個體類型。如果所選執行個體類型的其他執行個體無法使用，請 AWS Batch 等待其他執行個體可用。如果沒有足夠的可用執行個體，或者達到 Amazon EC2 服務限制，則在目前正在執行的任務完成之後才會執行其他任務。這種配置策略可以降低成本，但可

能限制擴展。如果搭配使用 Spot 叢集BEST_FIT，則必須指定 Spot 叢集 IAM 角色。使用BEST_FIT配置策略的計算資源不支援基礎結構更新，也無法更新某些參數。如需詳細資訊，請參閱 更新運算環境。

> **Note**
>
> BEST_FIT不支援使用 Amazon EKS 資源的運算環境。

BEST_FIT_PROGRESSIVE

使用足夠大的其他執行個體類型，以符合佇列中工作的需求。針對每個單元 vCPU 以較低的成本偏好執行個體類型。如果先前選取的執行個體類型的其他執行個體無法使用，請選 AWS Batch 取新的執行個體類型。

SPOT_CAPACITY_OPTIMIZED

(僅適用於 Spot 執行個體運算資源) 使用其他大小足以滿足佇列中任務需求的執行個體類型。偏好不太可能中斷的執行個體類型。

SPOT_PRICE_CAPACITY_OPTIMIZED

(僅適用於競價型執行個體運算資源) 價格和容量最佳化配置策略會查看價格和容量，以選擇最不可能中斷且價格最低的競價型執行個體集區。

> **Note**
>
> 我們建議您使用SPOT_PRICE_CAPACITY_OPTIMIZED而不是SPOT_CAPACITY_OPTIMIZED在大多數情況下使用。

使用隨需或競價型執行個體的SPOT_CAPACITY_OPTIMIZED、和SPOT_PRICE_CAPACITY_OPTIMIZED策BEST_FIT略，以及使用 Spot 執行個體的策略，AWS Batch 可能需maxvCpus要超過以符合您的容量需求。BEST_FIT_PROGRESSIVE在此情況下， AWS Batch 永遠不會超maxvCpus過單一執行個體以上。

有效值：BEST_FIT | BEST_FIT_PROGRESSIVE | SPOT_CAPACITY_OPTIMIZED | SPOT_PRICE_CAPACITY_OPTIMIZED

必要：否

`minvCpus`

即使運算環境為`DISABLED`，環境仍維護的 vCPUs 數目下限。

> **Note**
> 此參數不適用於在 Fargate 資源上執行的任務。

類型：整數

必要：否

`maxvCpus`

AWS Batch 運算環境可支援的 vCPUs 數目上限。

> **Note**
> 使用隨需或競價型執行個體的`SPOT_CAPACITY_OPTIMIZED`、和`SPOT_PRICE_CAPACITY_OPTIMIZED`配置策`BEST_FIT`略，以及使用 Spot 執行個體的策略， AWS Batch 可能需`maxvCpus`要超過以符合您的容量需求。`BEST_FIT_PROGRESSIVE`在此情況下， AWS Batch 永遠不會超`maxvCpus`過單一執行個體以上。例如， AWS Batch 使用計算環境中指定的執行個體不超過一個執行個體。

類型：整數

必要：否

`desiredvCpus`

運算環境中所需的 vCPUS 數目。 AWS Batch 根據工作佇列需求，在最小值和最大值之間修改此值。

> **Note**
> 此參數不適用於在 Fargate 資源上執行的任務。

類型：整數

必要：否

`instanceTypes`

可啟動的執行個體類型。此參數不適用於在 Fargate 資源上執行的任務。請勿指定它。您可以指定例證族群以啟動這些族群中的任何例證類型 (例如`c5`、`c5n`、或`p3`)。或者，您可以指定族群內的特定大小 (例如`c5.8xlarge`)。請注意，金屬例證類型不在例證族群中 (例如`c5`不包含) `c5.metal`。您還可以選擇 `optimal`，選取符合您任務佇列需求的執行個體類型 (C4、M4、R4 執行個體系列)。

> **Note**
>
> 在建立運算環境時，您為其選取的執行個體類型必須共用相同架構。例如，您無法在相同的運算環境中混合使用 x86 和 ARM 執行個體。

> **Note**
>
> 目前，`optimal` 使用 C4、M4 和 R4 執行個體系列中的執行個體類型。如果沒有來自這些例證族群的例證類型，則會使用 C5、M5 和 R5 例證族群中 AWS 區域 的例證類型。

類型：字串陣列

必要：是

`imageId`

此參數已過時。

用於運算環境啟動的執行個體的 Amazon Machine Image (AMI) ID。此參數會由 `Ec2Configuration` 結構的 `imageIdOverride` 成員覆寫。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

> **Note**
>
> 您為運算環境選擇的 AMI 必須與您要用於該運算環境的執行個體類型架構相符。例如，如果您的運算環境使用A1執行個體類型，則您選擇的計算資源 AMI 必須支援Arm執行個體。Amazon ECS 出售 Amazon ECS x86 的兩個Arm版本優化 Amazon Linux 2 AMI。有關詳情，請參閱 Amazon ECS Amazon 彈性容器服務開發人員指南中的 Amazon ECS 優化亞馬遜 Linux 2 AMI。

類型：字串

必要：否

`subnets`

啟動運算資源的 VPC 子網路。這些子網路必須位於相同的 VPC 中。Fargate 算資源最多可包含 16 個子網路。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 和子網路。

> **Note**
>
> AWS Batch 在 Amazon EC2 和 Amazon EKS AWS Batch 上支持 Local Zones。如需詳細資訊，請參閱 Amazon EC2 使用者指南、Amazon EKS 和 L oc al Zones 中的 L AWS ocal Zones，以及本機區域、Wavelength 區域中的 Amazon ECS 叢集以及 Amazon 彈性容器服務開發人員指南AWS Outposts中的本機區域。
> AWS Batch 在 Fargate 目前不支持 Local Zones。

更新運算環境時，如果您提供 VPC 子網路的空白清單，則 Fargate 和 EC2 運算資源之間產生的行為會有所不同。針對 Fargate 運算資源，系統會猶如未指定此參數且未進行變更一樣來提供空白清單。針對 EC2 運算資源，提供空白清單會從運算資源中移除 VPC 子網路。如果您變更 VPC 子網路，則需要更新運算環境的基礎結構。Fargate 和 EC2 運算資源都是這種情況。如需詳細資訊，請參閱 更新運算環境。

類型：字串陣列

必要：是

`securityGroupIds`

已與在運算環境中啟動之執行個體建立關聯的 Amazon EC2 安全群組。必須指定一或多個安全群組，這些群組必須位於 `securityGroupIds` 或是使用 `launchTemplate` 中參照的啟動範

本。對於在 Fargate 資源上執行的工作，且必須至少包含一個安全性群組，此參數是必要的。（Fargate 不支持啟動模板。）如果同時使用 `securityGroupIds` 和 `launchTemplate` 來指定安全群組，則會使用 `securityGroupIds` 中的值。

更新運算環境時，如果您提供安全群組的空白清單，則 Fargate 和 EC2 運算資源之間產生的行為會有所不同。針對 Fargate 運算資源，系統會猶如未指定此參數且未進行變更一樣來提供空白清單。針對 EC2 運算資源，提供空白清單會從運算資源中移除安全群組。如果您變更安全群組，則需要更新運算環境的基礎結構。Fargate 和 EC2 運算資源都是這種情況。如需詳細資訊，請參閱 更新運算環境。

類型：字串陣列

必要：是

`ec2KeyPair`

用於在運算環境中啟動的執行個體的 EC2 key pair。您可以使用此金鑰對，經由 SSH 登入您的執行個體。更新運算環境時，如果您變更 EC2 金鑰組，則需要對運算環境進行基礎設施更新。如需詳細資訊，請參閱 更新運算環境。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

類型：字串

必要：否

`instanceRole`

要連接到運算環境中 Amazon EC2 執行個體的 Amazon ECS 執行個體設定檔。此參數不適用於在 Fargate 資源上執行的任務。請勿指定它。您可以指定執行個體描述檔的簡稱或完整的 Amazon Resource Name (ARN)。例如 `ecsInstanceRole` 或 `arn:aws:iam::`*`aws_account_id`*`:instance-profile/ecsInstanceRole`。如需詳細資訊，請參閱 Amazon ECS 執行個體角色。

更新計算環境時，如果您變更此設定，則需要更新運算環境的基礎結構。如需詳細資訊，請參閱更新運算環境。

類型：字串

必要：否

`tags`

要套用至在運算環境中啟動之 EC2 執行個體的金鑰值配對標籤。舉例來說，您可以將 `"Name": "AWS Batch Instance - C4OnDemand"` 指定為標籤，以讓運算環境中的每個執行個體都擁有該名稱。這對於在 Amazon EC2 主控台中識別您的 AWS Batch 執行個體很有幫助。使用 AWS BatchListTagsForResourceAPI 操作時，看不到這些標籤。

更新運算環境時，如果您變更 EC2 標籤，則需要更新運算環境的基礎設施。如需詳細資訊，請參閱 更新運算環境。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

類型：字串到字串映射

必要：否

`placementGroup`

要與運算資源建立關聯的 Amazon EC2 置放群組。此參數不適用於在 Fargate 資源上執行的任務。請勿指定它。如果您打算將多節點 parallel 工作提交至您的計算環境，請考慮建立叢集置放群組，並將其與您的運算資源產生關聯。這可讓單一可用區域中執行個體邏輯群組上的多節點平行任務，保持較高網路流量潛力。如需詳細資訊，請參閱 Amazon EC2 Linux 執行個體使用者指南中的放置群組。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

類型：字串

必要：否

`bidPercentage`

EC2 Spot 執行個體價格在啟動執行個體之前，與該執行個體類型的隨需價格進行比較時所能獲得的最高百分比。例如，如果您的最大百分比為 20%，則 Spot 價格必須小於該 EC2 執行個體

目前隨需價格的 20%。您一律會支付最低價 (市價) 且絕不超過您的最大百分比。如果您將此欄位空，預設值是隨需價格的 100%。對於大多數使用案例，我們建議將此欄位留白。

更新運算環境時，如果您變更出價百分比，則需要更新運算環境的基礎結構。如需詳細資訊，請參閱 更新運算環境。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

必要：否

`spotIamFleetRole`

套用至 SPOT 運算環境的 Amazon EC2 Spot Fleet IAM 角色的 Amazon Resource Name (ARN)。如果將配置策略設為 BEST_FIT，或是若沒有指定配置策略，則此角色為必要項目。如需詳細資訊，請參閱 Amazon EC2 現貨叢集角色。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

> **Important**
>
> 若要在建立時標記 Spot 執行個體，此處指定的競價型叢集 IAM 角色必須使用較新的 AmazonEC2 SpotFleet TaggingRole 受管政策。先前建議的 AmazonEC2 SpotFleet 角色受管政策沒有標記 Spot 執行個體所需的許可。如需詳細資訊，請參閱 建立時未標記競價型執行個體。

類型：字串

必要：SPOT 運算環境必須擁有此參數。

`launchTemplate`

要與您運算資源相關聯的選用啟動範本。此參數不適用於在 Fargate 資源上執行的任務。請勿指定它。您在CreateComputeEnvironment或 UpdateComputeEnvironmentAPI 作業中指定的任何

其他計算資源參數會覆寫啟動範本中的相同參數。若要使用啟動範本，您必須在請求中擇一指定啟動範本 ID 或啟動範本名稱，而非同時指定兩者。如需詳細資訊，請參閱 啟動範本支援。

更新計算環境時，若要移除自訂啟動範本並使用預設啟動範本，請將啟動範本規格的`launchTemplateId`或`launchTemplateName`成員設定為空字串。從計算環境中移除啟動範本並不會移除啟動範本中指定的 AMI (如果啟動範本是使用的 AMI)。若要更新從啟動範本中選取的 AMI，必須將`updateToLatestImageVersion`參數設定為`true`。更新計算環境時，如果您變更啟動範本，則需要更新運算環境的基礎結構。如需詳細資訊，請參閱 更新運算環境。

類型：LaunchTemplateSpecification

object

必要：否

`launchTemplateId`

啟動範本的 ID。

類型：字串

必要：否

`launchTemplateName`

啟動範本的名稱。

類型：字串

必要：否

`version`

啟動範本的版本編號 `$Latest` 或 `$Default`。

如果數值為 `$Latest`，則會使用最新版本的啟動範本。如果數值為 `$Default`，則會使用啟動範本的預設版本。在基礎結構更新期間，如果為計算環境指定`$Default`了`$Latest`或，請 AWS Batch 重新評估啟動範本版本，並可能使用不同版本的啟動範本。這是即使未在更新中指定啟動範本。

預設：`$Default`。

類型：字串

必要：否

`ec2Configuration`

提供用於為 EC2 運算環境中執行個體選取 Amazon 機器映像 (AMI) 的資訊。如果`Ec2Configuration`未指定，則預設值為 Amazon Linux 2 (`ECS_AL2`)。在 2021 年 3 月 31 日之前，非 GPU、非重力子執行個 AWS 體的預設值為 Amazon Linux (`ECS_AL1`)。

更新計算環境時，如果您變更此參數，則需要更新運算環境的基礎結構。如需詳細資訊，請參閱更新運算環境。

> **Note**
>
> 此參數不適用於在 Fargate 資源上執行的任務。

類型：Ec2Configuration 物件陣列

必要：否

`imageIdOverride`

用於在運算環境中啟動的執行個體 (符合映像檔類型) 的 AMI ID。此設定會覆寫 `computeResource` 物件中的 `imageId` 集。

類型：字串

必要：否

`imageKubernetesVersion`

運算環境的Kubernetes版本。如果您未指定值，系統會使用 AWS Batch 支援的最新版本。

類型：字串

長度限制：長度下限為 1。長度上限為 256。

必要：否

`imageType`

與執行個體類型相符的映像類型，用於選取 AMI。ECS 和 EKS 資源支援的值不同。

ECS

若未指定 imageIdOverride 參數，則會使用最近的 Amazon ECS 最佳化 Amazon Linux 2 AMI (ECS_AL2)。如果在更新中指定了新映像類型，但未指定imageIdOverride參數imageId或參數，則會使用該映像類型支援的最新 Amazon ECS 最佳化 AMI。 AWS Batch

ECS_AL2

Amazon Linux 2 – 對於所有非 GPU 執行個體系列為預設。

ECS_AL2_NVIDIA

Amazon Linux 2 (GPU)：所有 GPU 執行個體系列的預設值 (例如P4和G4)，可用於所有非 AWS 重力型執行個體類型。

ECS_AL1

Amazon Linux。Amazon Linux 已經達到了標準 end-of-life 的支持。如需詳細資訊，請參閱 Amazon Linux AMI。

EKS

若未指定 imageIdOverride 參數，系統會使用最近的 Amazon EKS 最佳化的 Amazon Linux AMI (EKS_AL2)。如果在更新中指定了新映像類型，但未指定參數imageId或imageIdOverride參數，則會使用 AWS Batch 支援該映像類型的最新 Amazon EKS 最佳化 AMI。

EKS_AL2

Amazon Linux 2 – 對於所有非 GPU 執行個體系列為預設。

EKS_AL2_NVIDIA

Amazon Linux 2 (加速)：所有 GPU 執行個體系列的預設值 (例如P4和G4)，可用於所有非 AWS 重力型執行個體類型。

類型：字串

長度限制：長度下限為 1。長度上限為 256。

必要：是

Amazon EKS 配置

支援 AWS Batch 運算環境之 Amazon EKS 叢集的組態。必須有叢集，才能建立運算環境。

`eksClusterArn`

Amazon EKS 叢集的 Amazon Resource Name (ARN)。例如，`arn:aws:eks:us-east-1:123456789012:cluster/ClusterForBatch`。

類型：字串

必要：是

`kubernetesNamespace`

Amazon EKS 叢集的命名空間。 AWS Batch 管理此命名空間中的網繭。值不能留白或為 null。長度必須少於 64 個字元、不能設定為 `default`、不能以 "kube-" 開頭，且必須符合此規則表達式：`^[a-z0-9]([-a-z0-9]*[a-z0-9])?$`。如需詳細資訊，請參閱 Kubernetes 文件中的命名空間。

類型：字串

必要：是

類型：EksConfiguration物件

必要：否

服務角色

`serviceRole`

IAM 角色的完整 Amazon 資源名稱 (ARN)，可讓您代表您撥 AWS Batch 打其他 AWS 服務的電話。如需詳細資訊，請參閱 使用服務連結角色 AWS Batch。建議您不要指定服務角色。如此一來，就 AWS Batch 會使用AWSServiceRoleForBatch服務連結角色。

> **Important**
>
> 如果您的帳戶已建立 AWS Batch 服務連結角色 (AWSServiceRoleForBatch)，則除非您在此指定角色，否則依預設會在您的計算環境中使用該角色。如果您的帳戶中不存在 AWS Batch 服務連結角色，且此處未指定角色，則服務會嘗試在您的帳戶中建立 AWS Batch 服務連結角色。如需 AWSServiceRoleForBatch 服務連結角色的詳細資訊，請參閱服務連結角色權限 AWS Batch。
>
> 如果運算環境是使用AWSServiceRoleForBatch服務連結角色建立的，則無法將其變更為使用一般 IAM 角色。同樣地，如果使用一般 IAM 角色建立運算環境，則無法變更為使用AWSServiceRoleForBatch服務連結角色。若要更新需要基礎結構更新才能變更的運算環

境參數，必須使用AWSServiceRoleForBatch服務連結角色。如需詳細資訊，請參閱 更新運算環境。

如果您指定的角色具有路徑以外的路徑/，請確定指定完整角色 ARN (建議)，或在角色名稱前面加上路徑。

> **Note**
>
> 視您建立 AWS Batch 服務角色的方式而定，其 Amazon 資源名稱 (ARN) 可能包含`service-role`路徑前置詞。當您僅指定服務角色的名稱時，會 AWS Batch 假設 ARN 不使用`service-role`路徑前置詞。因此，我們建議您在建立運算環境時，指定您服務角色的完整 ARN。

類型：字串

必要：否

標籤

`tags`

要與計算環境關聯的金鑰值配對標籤。如需詳細資訊，請參閱 標記您的 AWS Batch 資源。

類型：字串到字串映射

必要：否

EC2 組態設定

AWS Batch 針對 EC2 和 EC2 競價型運算環境使用 Amazon ECS 最佳化的 AMI。默認為 Amazon Linux 2（`ECS_AL2`）。在 2021 年 3 月 31 日之前，非 GPU、非重力子執行個 AWS 體的預設值為 Amazon Linux (`ECS_AL1`)。

> **Note**
>
> AWS Batch 還支持 Amazon 2023。

Amazon Linux AMI（也稱為 Amazon Linux 1）在 2023 年 12 月 31 日達到其使用壽命終止。 AWS Batch 已經結束對 Amazon Linux AMI 的支援，因為自 2024 年 1 月 1 日起，它將不會收到任何安全性更新或錯誤修正。如需有關 Amazon Linux 的詳細資訊 end-of-life，請參閱 AL 常見問題集。

我們建議您將現有的 Amazon Linux 運算環境更新至 Amazon Linux 2023，以防止意外的工作負載中斷，並繼續接收安全性和其他更新。

您使用 Amazon Linux AMI 的運算環境可能會在 2023 年 12 月 31 end-of-life 日之後繼續運作。不過，這些運算環境將不再接收來自的任何新軟體更新、安全性修補程式或錯誤修正 AWS。之後，您有責任在 Amazon Linux AMI 上維護這些運算環境 end-of-life。我們建議將 AWS Batch 運算環境遷移到 Amazon Linux 2023 或 Amazon Linux 2，以維持最佳的性能和安全性。

有關 AWS Batch 從 Amazon AMI 遷移到 Amazon Linux 2023 或 Amazon Linux 2 的幫助，請參閱更新運算環境- AWS Batch

分配策略

建立受管運算環境時，請從`instanceTypes`指定的最符合工作需求的執行個體類型中 AWS Batch 選取執行個體類型。配置策略會定義 AWS Batch 需要額外容量時的行為。此參數不適用於在 Fargate 資源上執行的任務。請勿指定此參數。

`BEST_FIT` (default)

AWS Batch 選取最符合工作需求的執行個體類型，並使用成本最低執行個體類型的偏好設定。如果所選執行個體類型的其他執行個體無法使用，請 AWS Batch 等待其他執行個體可用。如果沒有足夠的可用執行個體，或者使用者達到 Amazon EC2 服務配額，則在目前正在執行的任務完成之前，不會執行其他任務。這種配置策略可以降低成本，但可能限制擴展。如果搭配使用 Spot 叢集`BEST_FIT`，則必須指定 Spot 叢集 IAM 角色。 `BEST_FIT`更新運算環境時不支援。如需詳細資訊，請參閱 更新運算環境。

> **Note**
>
> AWS Batch 管理您帳戶中的 AWS 資源。具有 BEST_FIT 配置策略的運算環境最初依預設使用啟動組態。但是，在新 AWS 帳戶中使用啟動配置將隨著時間的推移而受到限制。因此，從 2024 年 4 月下旬開始，新建立的 BEST_FIT 運算環境將預設為啟動範本。如果您的服務角色缺乏管理啟動範本的權限， AWS Batch 可能會繼續使用啟動設定。現有的運算環境將繼續使用啟動組態。

`BEST_FIT_PROGRESSIVE`

AWS Batch 選取其他大小足以符合佇列中工作需求的執行個體類型。建議使用每個單元 vCPU 成本較低的執行個體類型。如果先前選取的執行個體類型的其他執行個體無法使用，請選 AWS Batch 取新的執行個體類型。

`SPOT_CAPACITY_OPTIMIZED`

AWS Batch 選取一或多個足以符合佇列中工作需求的執行個體類型。建議使用較不可能中斷的執行個體類型。此配置策略僅適用於 Spot 執行個體運算資源。

`SPOT_PRICE_CAPACITY_OPTIMIZED`

價格和容量最佳化分配策略會考慮價格和容量，來選擇最不可能中斷且價格最低的 Spot 執行個體集區。此配置策略僅適用於 Spot 執行個體運算資源。

> **Note**
>
> 我們建議您使用`SPOT_PRICE_CAPACITY_OPTIMIZED`而不是`SPOT_CAPACITY_OPTIMIZED`在大多數情況下使用。

`BEST_FIT_PROGRESSIVE`和`BEST_FIT`策略使用隨需執行個體或 Spot 執行個體，且`SPOT_CAPACITY_OPTIMIZED`和`SPOT_PRICE_CAPACITY_OPTIMIZED`策略使用 Spot 執行個體。但是， AWS Batch 可能需要超過`maxvCpus`以符合您的容量需求。在此情況下， AWS Batch 永遠不會超`maxvCpus`過單一執行個體以上。

更新運算環境

建立使用 EC2 資源的運算環境後，您可以直接更新運算環境的許多設定。但是，變更某些設定需要 AWS Batch取代運算環境中的執行個體。

對於使用 Fargate 資源的運算環境，您可以更新下列項目。

- `securityGroupIds`
- `subnets`
- `desiredvCpus`
- `maxvCpus`
- `minvCpus`

AWS Batch有兩種更新機制。第一個是擴展更新，其中執行個體會從運算環境中新增或移除。第二個是基礎結構更新，會取代運算環境中的執行個體。基礎結構更新所花費的時間比擴展更新要長得多。

如果使用更新計算環境AWS Batch，則僅變更這些設定會導致擴展更新：所需的 vCPUs (desiredvCpus)、最大 vCPUs (maxvCpus)、最小 vCPUs (minvCpus)、服務角色 (serviceRole) 和狀態 ()。state

> **Note**
>
> 更新desiredvCpus設定時，值必須介於minvCpus和maxvCpus值之間。
> 此外，更新的desiredvCpus值必須大於或等於目前desiredvCpus值。如需詳細資訊，請參閱the section called “更新desiredvCpus設定時出現錯誤訊息”。

如果在 UpdateComputeEnvironmentAPI 動作中變更了下列任何設定，請AWS Batch啟動基礎結構更新。基礎結構更新需要將服務角色設定為 AWSServiceRoleForBatch(預設值)，且配置策略為BEST_FIT_PROGRESSIVESPOT_CAPACITY_OPTIMIZED、或SPOT_PRICE_CAPACITY_OPTIMIZED。 BEST_FIT不支援。除了服務角色之外，可以針對擴展更新變更的所有設定也可以變更基礎結構更新。

> **Note**
>
> 我們建議您在大多數情況下使用SPOT_PRICE_CAPACITY_OPTIMIZED而不是SPOT_CAPACITY_OPTIMIZED n。

在基礎結構更新期間，運算環境的狀態會變更為UPDATING。使用更新的設定啟動新執行個體。新的工作會在新執行個體上排程。目前正在執行的工作會根據基礎結構更新原則傳送。如需詳細資訊，請參閱UpdateComputeEnvironment和 AWS Batch API 參考中的 UpdatePolicy。

在UpdatePolicy資料類型中，請考慮下列案例：

> **Note**
>
> 在這些情況下，下列情況為真。當執行個體終止時，執行中的工作會停止。依預設，這些工作不會重試。若要在執行個體終止後重試其中一項工作，請設定工作重試策略。如需詳細資訊，請參閱《AWS Batch 使用者指南》中的 the section called “自動化工作重試”。

- 如果terminateJobsOnUpdate設定設為true，則執行中的工作會在基礎結構更新期間終止。會忽略此jobExecutionTimeoutMinutes設定。
- 如果terminateJobsOnUpdate設定設為false，則在基礎結構更新發生後，工作可以執行額外的時間。這個額外的時間是在設置中jobExecutionTimeoutMinutes配置的。依預設，設jobExecutionTimeoutMinutes定為 30 分鐘。

當運算環境中的容量變為可用時，會以更新的設定啟動新執行個體，並在新執行個體上啟動任務。當所有工作在具有舊設定的執行個體上完成時，舊執行個體就會終止。可用容量表示所需的 vCPUs 數目低於最小執行個體類型所需的 vCPUs 數目至少達到最小執行個體類型所需的 vCPUs 數目上限。

基礎架構更

若要變更運算環境的某些設定，必須進行基礎結構更新。如果變更下列任一設定，就會啟動基礎結構更新：

> **Important**
>
> 運算環境必須使用AWSServiceRoleForBatch服務連結角色，才能進行需要更新基礎結構的變更。
> 如果運算環境使用服務連結角色，則無法將其變更為使用一般 IAM 角色。同樣地，如果運算環境具有一般 IAM 角色，則無法將其變更為使用服務連結角色。因此，您只能在使用服務連結角色建立的運算環境上執行基礎結構更新。

- 配置策略 (allocationStrategy、必須是BEST_FIT_PROGRESSIVESPOT_CAPACITY_OPTIMIZED、或SPOT_PRICE_CAPACITY_OPTIMIZED。 如果原始配置策略是BEST_FIT，則不支援基礎結構更新。)

  > **Note**
  >
  > 我們建議您在大多數情況下使用SPOT_PRICE_CAPACITY_OPTIMIZED而不是SPOT_CAPACITY_OPTIMIZED n。

- 出價百分比 (bidPercentage)
- EC2 組態設定 (ec2Configuration)
- 金鑰配對 (ec2KeyPair)

- 影像識別碼 (imageId)
- 執行個體角色 (instanceRole)
- 執行個體類型 (instanceTypes)
- 啟動範本 (launchTemplate)
- 放置群組 (placementGroup)
- 安全性群組 (securityGroupIds)
- VPC 子網路 () subnets
- 標籤 (tags)
- 計算環境類型 (type，可以是EC2或之一SPOT)
- 是否要更新至基礎結構更新AWS Batch期間所支援的最新 AMI updateToLatestImageVersion

更新 AMI 識別碼

在基礎結構更新期間，計算環境的 AMI ID 可能會變更，具體取決於是否在這三個設定中指定 AMI。AMI 是在中指定的 imageId (中computeResources)、imageIdOverride (中) 或中ec2Configuration指定的啟動範本中launchTemplate指定的。假設在這些設定中未指定任何 AMI ID，且該updateToLatestImageVersion設定為true。然後，所支援的最新 Amazon ECS 最佳化 AMI 會用AWS Batch於任何基礎設施更新。

如果在這些設定中至少有一個指定 AMI ID，則更新取決於提供更新前所使用的 AMI ID 的設定。建立運算環境時，選取 AMI ID 的優先順序首先是啟動範本，然後是imageId設定，最後是imageIdOverride設定。但是，如果使用的 AMI ID 來自啟動範本，則更新imageId或imageIdOverride設定並不會更新 AMI ID。更新從啟動範本中選取的 AMI ID 的唯一方法是更新啟動範本。如果啟動範本的 version 參數為$Default或$Latest，則會評估指定啟動範本的預設或最新版本。如果預設選取了不同的 AMI ID，或選取了啟動範本的最新版本，則會在更新中使用該 AMI ID。

如果未使用啟動範本來選取 AMI ID，則會使用imageId或imageIdOverride參數中指定的 AMI 識別碼。如果同時指定兩者，則會使用在imageIdOverride參數中指定的 AMI ID。

假設運算環境使用、或launchTemplate參數指定的 AMI 識別碼 imageIdimageIdOverride，而且您想要使用支援的最新 Amazon ECS 最佳化 AMI。AWS Batch然後，更新必須移除提供 AMI ID 的設定。對於imageId，這需要為該參數指定一個空字串。對於imageIdOverride，這需要為ec2Configuration參數指定空字串。

如果 AMI ID 來自啟動範本，您可以變更為由AWS Batch下列其中一種方式支援的最新 Amazon ECS 最佳化 AMI：

- 指定`launchTemplateId`或`launchTemplateName`參數的空字串，以移除啟動範本。這會移除整個啟動範本，而不是單獨移除 AMI ID。
- 如果啟動範本的更新版本未指定 AMI ID，則必須將`updateToLatestImageVersion`參數設定為`true`。

Amazon EKS 運算環境

開始使 AWS Batch 用 Amazon EKS提供建立 EKS 運算環境的簡短指南。本節提供有關 Amazon EKS 運算環境的更多詳細資訊。

主題

預設 AMI 選擇

當您建立 Amazon EKS 運算環境時，您不需要指定 Amazon 機器映像 (AMI)。 AWS Batch 根據您在CreateComputeEnvironment請求中指定的Kubernetes版本和執行個體類型，選取 Amazon EKS 最佳化 AMI。一般而言，我們建議您使用預設 AMI 選項。有關 Amazon EKS 優化 AMI 的更多信息，請參閱 Amazon EKS 用戶指南中的 Amazon EKS 優化 Amazon Linux AMI。

執行下列命令以查看為您的 Amazon EKS 運算環境選取了哪種 AMI 類型 AWS Batch 。下列範例是非 GPU 執行個體類型。

```
# compute CE example: indicates Batch has chosen the AL2 x86 or ARM EKS 1.29 AMI,
 depending on instance types
$ aws batch describe-compute-environments --compute-environments My-Eks-CE1 \
    | jq '.computeEnvironments[].computeResources.ec2Configuration'
[
```

```
  {
    "imageType": "EKS_AL2",
    "imageKubernetesVersion": "1.29"
  }
]
```

下列範例是 GPU 執行個體類型。

```
# GPU CE example: indicates Batch has choosen the AL2 x86 EKS Accelerated 1.29 AMI
$ aws batch describe-compute-environments --compute-environments My-Eks-GPU-CE \
    | jq '.computeEnvironments[].computeResources.ec2Configuration'
[
  {
    "imageType": "EKS_AL2_NVIDIA",
    "imageKubernetesVersion": "1.29"
  }
]
```

支援的 Kubernetes 版本

AWS Batch 在 Amazon EKS 上目前支持以下Kubernetes版本：

- 1.29
- 1.28
- 1.27
- 1.26
- 1.25
- 1.24
- 1.23

當您使用 API 作業或 `CreateComputeEnvironment` API 作業建立
或`UpdateComputeEnvironment`更新運算環境時，您可能會看到類似下列的錯誤訊息。如果您在中指定不支援的Kubernetes版本，就會發生這個問題`EC2Configuration`。

```
At least one imageKubernetesVersion in EC2Configuration is not supported.
```

若要解決此問題，請刪除計算環境，然後使用支援的Kubernetes版本重新建立該環境。

您可以在 Amazon EKS 叢集上執行次要版本升級。例如，即使不支援次要版本，您1.yy也可以1.xx將叢集從升級為。

不過，在主要版本更新INVALID之後，運算環境狀態可能會變更為。例如，如果您執行從1.xx到的主要版本升級2.yy。如果主要版本不受支援 AWS Batch，您會看到類似下列的錯誤訊息。

```
reason=CLIENT_ERROR - ... EKS Cluster version [2.yy] is unsupported
```

更新運算環境的Kubernetes版本

您可以使用更新運算環境的Kubernetes版本 AWS Batch，以支援 Amazon EKS 叢集升級。運算環境的Kubernetes版本是 Amazon EKS AMI 版本，適用於 AWS Batch 啟動以執行任務的Kubernetes節點。您可以在更新 Amazon EKS 叢集控制平面Kubernetes版本之前或之後，在其 Amazon EKS 節點上執行版本升級。建議您在升級控制平面之後更新節點。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的更新 Amazon EKS 叢集Kubernetes版本。

若要升級運算環境的Kubernetes版本，請使用 UpdateComputeEnvironmentAPI 作業。

```
$ aws batch update-compute-environment \
    --compute-environment <compute-environment-name> \
    --compute-resources \
      'ec2Configuration=[{imageType=EKS_AL2,imageKubernetesVersion=1.23}]'
```

Kubernetes節點的共同責任

維護運算環境是共同的責任。

- 請勿變更或移除 AWS Batch 節點、標籤、污點、命名空間、啟動範本或 auto 調度資源群組。請勿將污染新增至 AWS Batch 受管節點。如果您進行上述任何變更，則無法支援您的運算環境，也會發生包括閒置執行個體在內的故障。
- 請勿將網繭鎖定至 AWS Batch 受管理節點。如果您將網繭鎖定到受管節點，則會發生中斷的調整和卡住的工作佇列。執行不 AWS Batch 在自我管理節點或受管節點群組上使用的工作負載。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的受管節點群組。
- 您可以DaemonSet將 a 定位為在 AWS Batch 受管節點上執行。如需詳細資訊，請參閱 DaemonSet 在 AWS Batch 受管節點上執行。

AWS Batch 不會自動更新計算環境 AMI。您有責任更新它們。執行下列命令，將 AMI 更新為最新的 AMI 版本。

```
$ aws batch update-compute-environment \
    --compute-environment <compute-environment-name> \
    --compute-resources 'updateToLatestImageVersion=true'
```

AWS Batch 不會自動升級Kubernetes版本。執行下列命令，將電腦環境的Kubernetes版本更新為 *1.23*。

```
$ aws batch update-compute-environment \
    --compute-environment <compute-environment-name> \
    --compute-resources \
      'ec2Configuration=[{imageType=EKS_AL2,imageKubernetesVersion=1.23}]'
```

更新為較新的 AMI 或Kubernetes版本時，您可以指定是否在更新工作時終止作業 (`terminateJobsOnUpdate`)，以及如果執行中的工作未完成，則要等待多久才取代執行個體 (`jobExecutionTimeoutMinutes`.) 如需詳細資訊[更新運算環境](#)，請參閱 [UpdateComputeEnvironment](#)API 作業中設定的基礎結構更新原則 ([UpdatePolicy](#))。

DaemonSet在 AWS Batch 受管節點上執行

AWS Batch 設置 AWS Batch 管理Kubernetes節點上的污點。您可以使用以下命令DaemonSet將 a 定位在 AWS Batch 受管節點上運行`tolerations`。

```
tolerations:
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
```

另一種方法是使用以下內容`tolerations`。

```
tolerations:
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
  effect: "NoSchedule"
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
  effect: "NoExecute"
```

使用啟動範本自訂

AWS Batch 在 Amazon EKS 支持啟動模板。您的啟動範本可以執行的動作存在限制。

Important

AWS Batch 運行/etc/eks/bootstrap.sh。請勿/etc/eks/bootstrap.sh在啟動範cloud-inituser-data本或指令碼中執行。您可以將參數以外的其他參--kubelet-extra-args數新增至 bootstrap.sh。若要執行此操作，請在/etc/aws-batch/batch.config檔案中設定AWS_BATCH_KUBELET_EXTRA_ARGS變數。如需詳細資訊，請參閱下列範例。

Note

如果啟動範本在呼CreateComputeEnvironment叫之後變更，則UpdateComputeEnvironment必須呼叫以評估要取代的啟動範本版本。

主題

添加kubelet額外的參數

AWS Batch 支持向kubelet命令添加額外的參數。如需支援參數的清單，請參閱Kubernetes文件kubelet中的。在下面的例子中，*--node-labels mylabel=helloworld*被添加到kubelet命令行。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
mkdir -p /etc/aws-batch

echo AWS_BATCH_KUBELET_EXTRA_ARGS=\"--node-labels mylabel=helloworld\" >> /etc/aws-
batch/batch.config
```

```
--==MYBOUNDARY==--
```

設定容器執行階段

您可以使用 AWS Batch CONTAINER_RUNTIME環境變數在受管理節點上設定容器執行階段。下列範例會將容器執行階段設定為bootstrap.sh執行containerd時間。如需詳細資訊，請參閱Kubernetes文件containerd中的。

> **Note**
>
> 環CONTAINER_RUNTIME境變數相當於的--container-runtime選項bootstrap.sh。如需詳細資訊，請參閱Kubernetes文件Options中的。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
mkdir -p /etc/aws-batch

echo CONTAINER_RUNTIME=containerd >> /etc/aws-batch/batch.config

--==MYBOUNDARY==--
```

掛接 Amazon EFS 磁碟區

您可以使用啟動範本將磁碟區掛接到節點。在下列範例中，會使用cloud-configpackages和runcmd設定。如需詳細資訊，請參閱cloud-init文件中的 Cloud 設定範例。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
- amazon-efs-utils
```

```
runcmd:
- file_system_id_01=fs-abcdef123
- efs_directory=/mnt/efs

- mkdir -p ${efs_directory}
- echo "${file_system_id_01}:/ ${efs_directory} efs _netdev,noresvport,tls,iam 0 0"
 >> /etc/fstab
- mount -t efs -o tls ${file_system_id_01}:/ ${efs_directory}

--==MYBOUNDARY==--
```

若要在工作中使用此磁碟區，必須將其加入至 EksProperty 參數中。RegisterJobDefinition下列範例是工作定義的很大一部分。

```
{
    "jobDefinitionName": "MyJobOnEks_EFS",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["ls", "-la", "/efs"],
                    "resources": {
                        "limits": {
                            "cpu": "1",
                            "memory": "1024Mi"
                        }
                    },
                    "volumeMounts": [
                        {
                            "name": "efs-volume",
                            "mountPath": "/efs"
                        }
                    ]
                }
            ],
            "volumes": [
                {
                    "name": "efs-volume",
                    "hostPath": {
                        "path": "/mnt/efs"
```

```
                    }
                }
            ]
        }
    }
}
```

在節點中，Amazon EFS 磁碟區會掛接在目/mnt/efs錄中。在 Amazon EKS 任務的容器中，磁碟區會掛接在/efs目錄中。

IPv6 支援

AWS Batch 支援具有 IPv6 地址的 Amazon EKS 叢集。無需自訂即可獲得 AWS Batch 支援。不過，在開始之前，我們建議您檢閱 Amazon EKS 使用者指南中將 IPv6 地址指派給網繭和服務中概述的考量和條件。

運算資源記憶體管理

Amazon ECS 容器代理程式在運算環境中註冊運算資源時，代理程式必須判斷運算資源可用於為您的任務保留多少記憶體。由於平台記憶體額外負荷和系統核心佔用的記憶體，此數字與 Amazon EC2 執行個體安裝的記憶體數量不同。舉例而言，m4.large 執行個體安裝了 8 GiB 的記憶體。但是，當計算資源註冊時，這並不總是轉換為可用於作業的 8192 MiB 記憶體。

假設您為工作指定 8192 MiB，而且沒有任何計算資源有 8192 MiB 或更大的記憶體可用來滿足此需求。然後，工作無法放置在您的計算環境中。如果您使用的是受管運算環境，則AWS Batch必須啟動較大的執行個體類型以容納要求。

預設的 AWS Batch 運算資源 AMI 也會預留 32 MiB 的記憶體供 Amazon ECS 容器代理程式和其他重要系統程序使用。此記憶體無法用於工作分配。如需詳細資訊，請參閱預留系統記憶體。

Amazon ECS 容器代理程式會使用 Docker ReadMemInfo() 函數來查詢作業系統可用的記憶體總量。Linux 提供了命令行實用程序來確定總內存。

Example - 判定 Linux 記憶體總量

該free命令返回由操作系統識別的總內存。

```
$ free -b
```

以下是執行 Amazon ECS 最佳化 Amazon Linux AMI 的m4.large執行個體的範例輸出。

```
             total       used       free     shared    buffers     cached
Mem:    8373026816  348180480 8024846336      90112   25534464  205418496
-/+ buffers/cache:  117227520 8255799296
```

這個執行個體有 8373026816 個位元組的記憶體總計。這表示有 7985 MiB 可用於工作。

預留系統記憶體

如果您在工作中佔用計算資源上的所有記憶體，您的工作可能會與記憶體的關鍵系統處理程序競爭，並可能導致系統故障。Amazon ECS 容器代理程式提供稱為`ECS_RESERVED_MEMORY`的組態變數。您可以使用此組態變數，從配置給工作的集區中移除指定數目的 MiB 記憶體。這可為重要系統程序有效地預留記憶體。

預設的 AWS Batch 運算資源 AMI 會預留 32 MiB 的記憶體供 Amazon ECS 容器代理程式和其他重要系統程序使用。

檢視運算資源記憶體

您可以在 Amazon ECS 主控台或透過 DescribeContainerInstancesAPI 作業檢視運算資源註冊的記憶體量。如果您嘗試為特定執行個體類型提供作業盡可能多的記憶體，以最大限度地提高資源使用率，則可以觀察該計算資源可用的記憶體，然後指派工作那麼多記憶體。

若要檢視計算資源記憶體

1. 開啟主控台，網址為 https://console.aws.amazon.com/ecs/v2。
2. 選擇 [叢集]，然後選擇主控要檢視之運算資源的叢集。

   您運算環境的叢集名稱以您運算環境的名稱開頭。
3. 選擇基礎結構。
4. 在容器執行個體下，選擇容器執行個體。
5. [資源和網路] 區段顯示計算資源的已註冊記憶體和可用記憶體。

   已註冊的記憶體值是第一次啟動 Amazon ECS 時向 Amazon ECS 註冊的運算資源，而「可用記憶體」值則是尚未分配給任務的值。

Amazon EKS AWS Batch 上的記憶體和 vCPU 考量

AWS Batch在 Amazon EKS 中，您可以指定可供容器使用的資源。例如，您可以為 vCPU 和記憶體資源指定`requests`或`limits`值。

以下是指定 vCPU 資源的限制：

- 至少必須指定一個 vCPU `requests` 或`limits`值。
- 一個 vCPU 單元相當於一個實體或虛擬核心。
- vCPU 值必須以整數輸入，或以 0.25 的遞增方式輸入。
- 最小的有效 vCPU 值為 0.25。
- 如果同時指定兩者，則`requests`值必須小於或等於該`limits`值。如此一來，您就可以同時設定軟體和硬體 vCPU 組態。
- 無法以毫升形式指定 vCPU 值。例如，`100m`不是有效的值。
- AWS Batch使用該`requests`值進行擴展決策。如果未指定`requests`值，則會將該`limits`值複製到該`requests`值。

以下是指定記憶體資源的限制條件：

- 至少必須指定一個記憶體`requests`或`limits`值。
- 記憶體值必須在 mebibytes (MiBs) 中。
- 如果同時指定兩者，則`requests`值必須等於該`limits`值。
- AWS Batch使用該`requests`值進行擴展決策。如果未指定`requests`值，則會將`limits`值複製到該`requests`值。

以下是指定 GPU 資源的限制：

- 如果同時指定兩者，則`requests`值必須等於該`limits`值。
- AWS Batch使用該`requests`值進行擴展決策。如果未指定`requests`值，則會將該`limits`值複製到該`requests`值。

工作定義範例

Amazon EKS 任務定義AWS Batch上的以下內容可設定軟體 vCPU 共用。這可讓 AWS Batch Amazon EKS 使用執行個體類型的所有 vCPU 容量。但是，如果有其他工作正在執行，則會配置最多 2 vCPUs 的工作。記憶體限制為 2 GB。

```
{
    "jobDefinitionName": "MyJobOnEks_Sleep",
    "type": "container",
```

```
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["sleep", "60"],
                    "resources": {
                        "requests": {
                            "cpu": "2",
                            "memory": "2048Mi"
                        }
                    }
                }
            ]
        }
    }
}
```

Amazon EKS 任務定義AWS Batch上的以下內容的`request`值為，1並將最多 4 vCPUs 分配給任務。

```
{
    "jobDefinitionName": "MyJobOnEks_Sleep",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["sleep", "60"],
                    "resources": {
                        "requests": {
                            "cpu": "1"
                        },
                        "limits": {
                            "cpu": "4",
                            "memory": "2048Mi"
                        }
                    }
                }
            ]
        }
    }
}
```

Amazon EKS 任務定義AWS Batch上的以下內容將 vCPU `limits` 值設定為，1並將記憶體limits值設定為 1 GB。

```
{
    "jobDefinitionName": "MyJobOnEks_Sleep",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["sleep", "60"],
                    "resources": {
                        "limits": {
                            "cpu": "1",
                            "memory": "1024Mi"
                        }
                    }
                }
            ]
        }
    }
}
```

將 Amazon EKS AWS Batch 上的任務AWS Batch轉譯為 Amazon EKS 網繭時，請將該值AWS Batch複製到該limits值。requests這是如果沒有指定requests值。當您提交上述範例工作定義時，網繭spec如下所示。

```
apiVersion: v1
kind: Pod
...
spec:
  ...
  containers:
    - command:
        - sleep
        - 60
      image: public.ecr.aws/amazonlinux/amazonlinux:2
      resources:
        limits:
          cpu: 1
          memory: 1024Mi
```

```
        requests:
          cpu: 1
          memory: 1024Mi
      ...
```

節點 CPU 和記憶體保留

AWS Batch依賴 vCPU 和記憶體保留區的`bootstrap.sh`檔案預設邏輯。如需有關`bootstrap.sh`檔案的詳細資訊，請參閱 bootstrap.sh。調整 vCPU 和記憶體資源的大小時，請考慮以下範例。

> **Note**
>
> 如果沒有執行任何執行個體，vCPU 和記憶體保留最初可能會影響AWS Batch擴展邏輯和決策。執行執行個體之後，AWS Batch調整初始配置。

節點 CPU 保留範例

CPU 保留值是使用執行個體可用的 vCPUs 總數來計算，以毫秒為單位。

vCPU 編號	保留百分比
1	6%
2	1%
3-4	0.5%
4 及以上	0.25%

使用前面的值，下列情況為真：

- 具有 2 個 vCPUs 的`c5.large`執行個體的 CPU 保留值為 70 公尺。這是通過以下方式計算：（1* 60）+（1* 10）= 70 米。
- 具有 96 個 vCPUs 的`c5.24xlarge`執行個體的 CPU 保留值為 310 公尺。這是通過以下方式計算：（1* 60）+（1* 10）+（2* 5）+（92* 2.5）= 310 米。

在此範例中，有 1930 個 (計算出 2000-70) 毫米 vCPU 單位可用於在執行個體上執行工作。c5.large假設您的工作需要 2 (2*1000 m) vCPU 單位，則該工作不適用於單一執行個體。c5.large但是，需要 1.75 vCPU 單元的工作適合。

節點記憶體保留範例

記憶體保留值以 MB 為單位，使用下列項目計算：

- 執行個體容量 (MB)。例如，一個 8 GB 的執行個體是 7,748 MiB 個。
- kubeReserved值。此kubeReserved值是為系統精靈保留的記憶體容量。此kubeReserved值的計算方式如下：((11* 執行個體類型支援的網繭數目上限) + 255)。如需執行個體類型所支援之網繭數目上限的相關資訊，請參閱 eni-max-pods. txt
- HardEvictionLimit值。當可用記憶體低於該HardEvictionLimit值時，執行個體會嘗試收回 Pod。

#################### _ ###-#11######-255-###HardEvictionLimit*)).

c5.large執行個體最多可支援 29 個網繭。對於HardEvictionLimit值為 100 MiB 的 8 GB c5.large 執行個體，可分配的記憶體為 7074。MiB這是以下列方式計算：(7748-(11 * 29) -255 -100) = 7074 千 MiB。在這個範例中，8,192 個MiB工作不適合這個執行個體，即使它是 8 gibibyte (GiB) 執行個體。

DaemonSets

使用時DaemonSets，請考慮下列事項：

- 如果 Amazon EKS 執行個體AWS Batch上沒有執行，一開始DaemonSets可能會影響AWS Batch擴展邏輯和決策。 AWS Batch一開始會針對預期配置 0.5 個 vCPU 單元和 500 MiB。DaemonSets執行執行個體之後，AWS Batch調整初始配置。
- 如果DaemonSet定義了 vCPU 或記憶體限制，則AWS Batch在 Amazon EKS 任務上擁有的資源較少。我們建議您盡可DaemonSets能減少指派給AWS Batch工作的數量。

排程原則

您可以使用排程原則來設定工作佇列中的計算資源在使用者或工作負載之間的配置方式。使用排程原則，您可以將不同的公平共用識別碼指派給工作負載或使用者。 AWS Batch為每個公平共用識別碼指定一段時間內可用資源總數的百分比。

公平份額百分比是使用shareDecaySeconds和shareDistribution值來計算。您可以將共用減少時間指派給原則，為公平共用分析增加時間。增加時間會給時間帶來更多的重量，而且對於定義的重量更少。您可以透過指定計算保留區，保留未使用中的公平共用識別碼的計算資源。如需詳細資訊，請參閱 排程原則參數。

主題

建立排程原則

您必須先建立排程原則，才能使用排程原則建立工作佇列。建立排程原則時，您可以將一或多個公平共用識別碼或公平共用識別碼前置碼與佇列的權重建立關聯，並選擇性地將衰減期間和計算保留區指派給原則。

建立排程原則

1. 開啟主AWS Batch控台，網址為 https://console.aws.amazon.com/batch/。
2. 從導覽列中選取要使用的「區域」。
3. 在導覽窗格中，選擇 [排程原則] > [建立]。
4. 在 [名稱] 中，輸入排程原則的唯一名稱。可以包含最多可達 128 個字元 (大小寫)、數字、連字號和底線。
5. (選擇性) 針對共用衰減秒數，輸入排程原則共用衰減時間的整數值。較長的共用衰減時間會在排程工作時考慮較長的計算資源使用量。如果公平共用識別碼最近沒有使用計算資源，則使用公平共用識別碼的工作可以暫時使用比該公平共用識別碼所允許的權重更多的運算資源。
6. (選擇性) 對於 Compute 保留區，請為排程原則的計算保留區輸入整數值。計算保留區將保留一些 vCPUs，以用於目前非作用中的公平共用識別碼。

   保留比例為 (*computeReservation*/100)^*ActiveFairShares*，其中 ActiveFairShares 為作用中公平共用識別碼的數量。

例如，computeReservation值 50 表示如果只有一個公平共用識別碼，則AWS Batch應保留最大可用 VCPU 的 50%；如果有兩個公平共用識別碼，則應保留 25%；如果有三個公平共用識別碼，則應保留 12.5%。computeReservation值 25 表示如果只有一個公平共用識別碼，則AWS Batch應保留最大可用 VCPU 的 25%；如果有兩個公平共用識別碼，則應保留 6.25%；如果有三個公平共用識別碼，則應保留 1.56%。

7. 在「共用屬性」區段中，您可以為每個公平共用識別碼指定公平共用識別碼和權重，以便與排程原則產生關聯。

   a. 選擇新增共用識別碼。

   b. 針對共用識別碼，指定公平共用識別碼。如果字串以 '*' 結尾，則會變成公平共用識別碼前綴，用於比對工作的公平共用識別碼。排程政策中的所有公平共用識別碼和公平共用識別碼前置詞都必須是唯一且不能重疊。例如，您不能在相同的排程原則中使用公平共用識別碼前置詞「usera*」和公平共用識別碼「UserA1」。

   c. 對於「線粗係數」，指定公平共用識別碼的相對權重。預設值為 1.0。較低的值對於運算資源的優先順序較高。如果使用公平共用識別碼前置詞，則具有以前綴開頭的公平共用識別碼的工作將共用權重係數。這會有效地增加這些工作的加權係數，降低其個別優先順序，但保持公平共用識別碼字首的相同權重係數。

8. (選擇性) 在「標記」區段中，您可以指定要與排程原則產生關聯的每個標籤的金鑰和值。如需詳細資訊，請參閱標記您的 AWS Batch 資源。

9. 選擇 [提交] 以完成並建立您的排程原則。

排程原則範本

空白的排程原則範本如下所示。您可以使用此範本建立排程原則，然後將其儲存至檔案並與AWS CLI--cli-input-json選項搭配使用。如需這些參數的詳細資訊，請參閱 AWS BatchAPI 參考CreateSchedulingPolicy中的。

```
{
    "name": "",
    "fairsharePolicy": {
        "shareDecaySeconds": 0,
        "computeReservation": 0,
        "shareDistribution": [
            {
                "shareIdentifier": "",
                "weightFactor": 0.0
            }
```

```
        ]
    },
    "tags": {
        "KeyName": ""
    }
}
```

> **Note**
>
> 您可以使用下列AWS CLI命令產生前面的工作佇列範本。
>
> ```
> $ aws batch create-scheduling-policy --generate-cli-skeleton
> ```

排程原則參數

排程原則分為三個基本元件：排程原則的名稱、公平共用原則和標籤。

主題

- 排程原則名稱
- 公平共享政策
- 標籤

排程原則名稱

`name`

排程原則的名稱。可以包含最多可達 128 個字元 (大小寫)、數字、連字號和底線。

類型：字串

必要：是

公平共享政策

`fairsharePolicy`

排程政策的公平共用政策。

```
"fairsharePolicy": {
   "computeReservation": number,
   "shareDecaySeconds": number,
   "shareDistribution": [
      {
         "shareIdentifier": "string",
         "weightFactor": number
      }
   ]
}
```

類型：物件

必要：否

computeReservation

用於為尚未使用的公平共用識別碼保留部分可用最大 VCPU 的值。

保留比例為 (*computeReservation*/100)^*ActiveFairShares*，其中 ActiveFairShares 為作用中公平共用識別碼的數量。

例如，computeReservation值 50 表示如果只有一個使用中的公平共用識別碼，則AWS Batch應保留最大可用 VCPU 的 50%；如果有兩個作用中的公平共用識別碼，則應保留 25%；如果有三個作用中的公平共用識別碼，則應保留 12.5%。computeReservation值 25 表示如果只有一個使用中的公平共用識別碼，則AWS Batch應保留最大可用 VCPU 的 25%；如果有兩個作用中的公平共用識別碼，則應保留 6.25%；如果有三個作用中的公平共用識別碼，則應保留 1.56%。

類型：整數

有效範圍：最小值為 0。最大值 99。

必要：否

shareDecaySeconds

用於計算每個正在使用的公平股份識別碼的公平份額百分比的時間段。若此值為零 (0)，表示僅應測量目前的使用量。衰減允許最近執行的任務比先前執行的任務具有更多權重。

類型：整數

有效範圍：最小值為 0。最大值為 604800（一週）。

必要：否

shareDistribution

包含公平共用原則之公平共用識別碼權重的物件陣列。未包含的公平共用識別碼的預設權數為1.0。

```
"shareDistribution": [
   {
      "shareIdentifier": "string",
      "weightFactor": number
   }
]
```

類型：陣列

必要：否

shareIdentifier

公平共用識別碼或公平共用識別碼字首。如果字符串以 '*' 結尾，則此字符串為以該前綴開頭的公平共享標識符指定公平共享標識符前綴。例如，如果值是UserA*且為 1，並且有兩個以開頭的公平共用識別碼UserA，則每個這些公平共用識別碼的權重將為 2；如果有五個這樣的公平共用識別碼，則每個識別碼的權重為 5。weightFactor

公平共用政策中的公平共用識別碼和公平共用識別碼前綴清單不能重疊。例如，您不能UserA-1在相同的公平共用政策中使用UserA*的公平共用識別碼前置詞和公平共用識別碼。

類型：字串

必要：是

weightFactor

公平共用識別碼的權重係數。預設值為 1.0。較低的值對於運算資源的優先順序較高。例如，使用權重係數為 0.125 (1/8) 之共用識別碼的任務，與使用權重係數為 1 之共用識別碼的任務相較之下，會取得 8 倍的運算資源。

支援的最小值為 0.0001，支援的最大值為 999.9999。

類型：浮點數

必要：否

標籤

`tags`

要與排程原則相關聯的索引鍵值配對標籤。如需詳細資訊，請參閱標記您的 AWS Batch 資源。

類型：字串到字串映射

必要：否

使用主控台中的 Step Functions 狀態機器協調AWS Batch工作 AWS Batch

您可以使用AWS Batch主控台來檢視有關 Step Functions 狀態機器及其使用之功能的詳細資料。

章節

- 檢視狀態機器詳細資料
- 編輯狀態機器
- 執行狀態機器

檢視狀態機器詳細資料

主AWS Batch控台會顯示目前狀態機器的清單，其中至少包AWS 區域含一個提交工作的AWS Batch工作流程步驟。

選擇狀態機器以檢視代表工作流程的圖形。以藍色反白顯示的步驟代表AWS Batch工作。使用圖表控制項目來放大、縮小和置中圖表。

> **Note**
>
> 在狀態機器定義 JsonPath中動態參考AWS Batch工作時，無法在AWS Batch主控台中顯示函數詳細資料。相反地，工作名稱會列為動態參照，且圖形中對應的步驟會變成灰色。

檢視狀態機器詳細資料

1. 開啟由 Step Functions 支援的AWS Batch主控台 [工作流程協調] 頁面。
2. 選擇狀態機器。
   <result>

   AWS Batch 主控台會開啟詳細資料頁面。
   </result>

如需詳細資訊，請參閱 AWS Step Functions 開發人員指南中的 Step Functions。

編輯狀態機器

當您要編輯狀態機器時，會AWS Batch開啟 Step Functions 主控台的編輯定義頁面。

編輯狀態機器

1. 開啟由 Step Functions 支援的AWS Batch主控台 [工作流程協調] 頁面。
2. 選擇狀態機器。
3. 選擇編輯。

   Step Functions 主控台會開啟 Edit definition (編輯定義) 頁面。
4. 編輯狀態機器，然後選擇儲存。

如需有關編輯狀態機器的詳細資訊，請參閱 AWS Step Functions 開發人員指南中的 Step Functions 狀態機器語言。

執行狀態機器

當您想要執行狀態機器時，會AWS Batch開啟 Step Functions 主控台的 [新增執行] 頁面。

執行狀態機器

1. 開啟由 Step Functions 支援的AWS Batch主控台 [工作流程協調] 頁面。
2. 選擇狀態機器。
3. 選擇 Execute (執行)。

   Step Functions 主控台會開啟 New execution (新執行) 頁面。
4. (選用) 編輯狀態機器並選擇開始執行。

如需執行狀態機器的詳細資訊，請參閱 AWS Step Functions 開發人員指南中的 Step Functions 狀態機器執行概念。

AWS Batch關於 AWS Fargate

AWSFargate 是一種技術，您可以使用它AWS Batch來執行容器，而無需管理伺服器或 Amazon EC2 執行個體叢集。使用 AWS Fargate，就不再需要佈建、設定或擴展虛擬機器的叢集來執行容器。這樣一來即無須選擇伺服器類型、決定何時擴展叢集，或最佳化叢集壓縮。

使用 Fargate 資源執行任務時，您可以將應用程式封裝在容器中、指定 CPU 和記憶體需求、定義聯網和 IAM 政策，以及啟動應用程式。每個 Fargate 作業都有自己的隔離界限，不會與其他工作共用基礎核心、CPU 資源、記憶體資源或 elastic network interface。

內容

何時使用 Fargate

我們建議在大多數情況下使用 Fargate。Fargate 會啟動並擴展運算，以便與您為容器指定的資源需求密切相符。有了 Fargate，您不需要過度佈建或支付額外的伺服器費用。您也不需要擔心基礎結構相關參數 (例如執行個體類型) 的具體細節。當運算環境需要擴充時，在 Fargate 資源上執行的工作可以更快速地開始。通常，啟動新的 Amazon EC2 執行個體需要幾分鐘的時間。不過，在 Fargate 上執行的工作可以在大約 30 秒內佈建。所需的確切時間取決於多種因素，包括容器映像大小和作業數量。

不過，如果您的任務需要下列任何一項，我們建議您使用 Amazon EC2：

- 超過 16 個 vCPUs
- 超過 120 GB 的記憶體
- 一個 GPU
- 自定義 Amazon 機器映像（AMI）
- 任何 Linux 參數參數的參數

如果您有大量的任務，建議您使用 Amazon EC2 基礎設施。例如，如果同時執行的工作數目超過 Fargate 節流限制。這是因為使用 EC2 時，任務可以以比 Fargate 資源更高的速率分派給 EC2 資源。

此外，當您使用 EC2 時，可以同時執行更多工作。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 AWS Fargate 服務配額。

Fargate 上的 Job 定義

AWS BatchFargate 上的工作不支援所有可用的工作定義參數。某些參數完全不受支援，而另一些參數對於 Fargate 工作的行為不同。

下列清單說明 Fargate 工作中無效或受其他限制的工作定義參數。

`platformCapabilities`

必須指定為FARGATE。

```
"platformCapabilities": [ "FARGATE" ]
```

`type`

必須指定為container。

```
"type": "container"
```

`containerProperties` 中的參數

`executionRoleArn`

必須為在 Fargate 資源上執行的工作指定。如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的任務 IAM 角色。

```
"executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole"
```

`fargatePlatformConfiguration`

（可選，僅適用於 Fargate 工作定義）。指定 Fargate 平台版本，或LATEST指定最新平台版本。的可能值platformVersion為1.3.01.4.0、和 LATEST (預設值)。

```
"fargatePlatformConfiguration": { "platformVersion": "1.4.0" }
```

`instanceType`, `ulimits`

不適用於在 Fargate 資源上執行的工作。

`memory`, `vcpus`

這些設定必須在 `resourceRequirements`

`privileged`

請勿指定此參數，或指定`false`。

```
"privileged": false
```

`resourceRequirements`

記憶體和 vCPU 需求都必須使用支援的值來指定。在 Fargate 資源上執行的工作不支援 GPU 資源。

如果您使用 GuardDuty 執行階段監視，則 GuardDuty 安全性代理程式會產生輕微的記憶體額外負荷。因此記憶體限制必須包含 GuardDuty安全性代理程式的大小。如需有關 GuardDuty Security Agent 記憶體限制的資訊，請參閱《GuardDuty 使用手冊》中的 CPU 和記憶體限制。如需最佳實務的相關資訊，請參閱 Amazon ECS 開發人員指南中的啟用執行時期監控後，如何修復 Fargate 任務中的記憶體不足錯誤。

```
"resourceRequirements": [
  {"type": "MEMORY", "value": "512"},
  {"type": "VCPU",   "value": "0.25"}
]
```

`linuxParameters` 中的參數

`devices`, `maxSwap`, `sharedMemorySize`, `swappiness`, `tmpfs`

不適用於在 Fargate 資源上執行的工作。

`logConfiguration` 中的參數

`logDriver`

只`splunk`有`awslogs`和受支援。如需詳細資訊，請參閱使用 awslogs 日誌驅動程式。

成員 `networkConfiguration`

`assignPublicIp`

如果私有子網路沒有連接 NAT 閘道來傳送流量至網際網路，則`assignPublicIp`必須是 "ENABLED"。如需詳細資訊，請參閱AWS Batch 執行 IAM 角色。

Fargate 上的 Job 佇列

AWS BatchFargate 上的工作佇列基本上是不變的。唯一的限制是列在中的運算環境都computeEnvironmentOrder必須是 Fargate 運算環境 (FARGATE或FARGATE_SPOT)。EC2 和 Fargate 運算環境不能混合使用。

Fargate 上的運算環境

AWS BatchFargate 上的運算環境不支援所有可用的運算環境參數。某些參數完全不受支援。其他人對 Fargate 有特定要求。

下列清單說明 Fargate 作業中無效或受其他限制的計算環境參數。

`type`

此參數必須是MANAGED。

```
"type": "MANAGED"
```

computeResources對象中的參數

`allocationStrategy`, `bidPercentage`, `desiredvCpus`, `imageId`, `instanceTypes`, `ec2Configuration`, `ec2KeyPair`, `instanceRole`, `launchTemplate`, `minvCpus`, `placementGroup`, `spotIamFleetRole`

這些不適用於 Fargate 運算環境，因此無法提供。

`subnets`

如果此參數中列出的子網路未附加 NAT 閘道，則必須將工作定義中的assignPublicIp參數設定為ENABLED。

`tags`

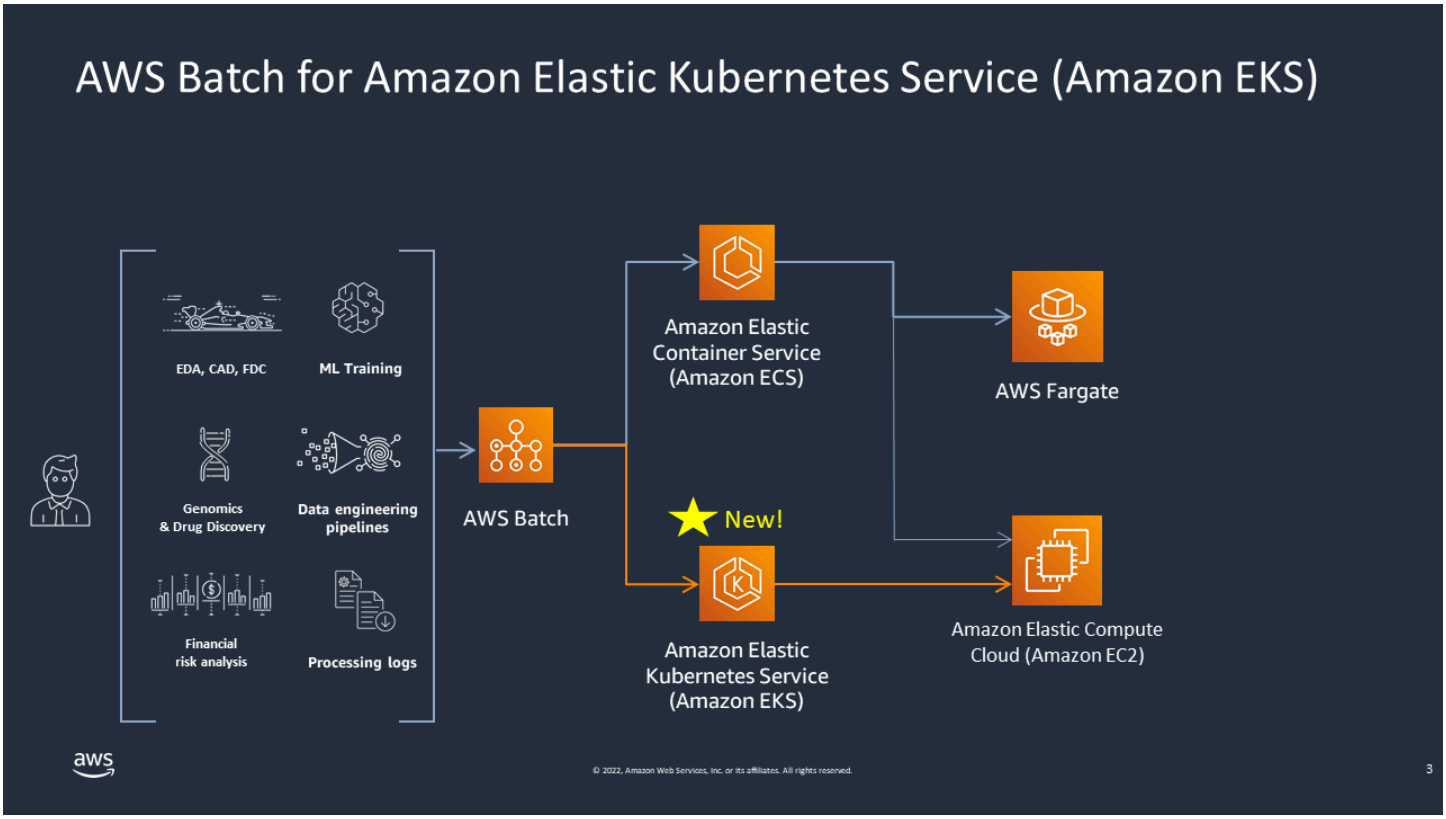
這不適用於 Fargate 運算環境，因此無法提供。若要為 Fargate 計算環境指定標籤，請使用computeResources物件中沒有的tags參數。

`type`

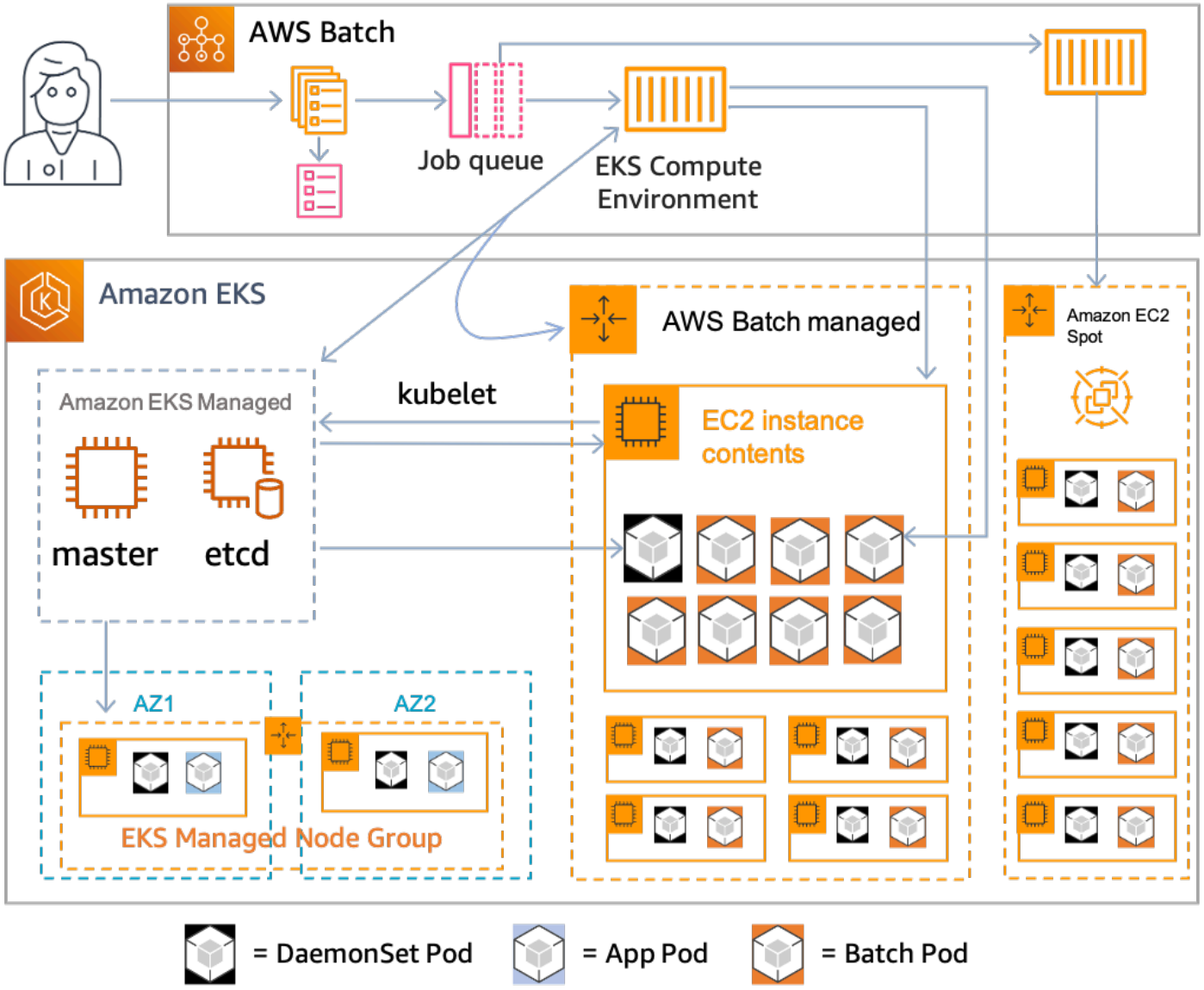
此必須為 FARGATE 或 FARGATE_SPOT。

```
"type": "FARGATE_SPOT"
```

AWS Batch 在 Amazon EKS 上



AWS Batch 透過提供受管批次功能，簡化 Amazon EKS 叢集上的批次工作負載。這包括佇列、相依性追蹤、受管理的工作重試和優先順序、網繭管理和節點調整。 AWS Batch 可以處理多個可用區域和多個 Amazon EC2 執行個體類型和大小。 AWS Batch 整合多個 Amazon EC2 Spot 最佳實務，以容錯的方式執行工作負載，減少中斷情況。您可 AWS Batch 以放心地執行少數隔夜工作或數百萬個關鍵任務工作。



AWS Batch 這是一項受管服務，可協調Kubernetes叢集中由 Amazon 彈性 Kubernetes 服務 (Amazon EKS) 管理的批次工作負載。 AWS Batch 使用「覆蓋」模型在叢集外部執行此協調作業。由於 AWS Batch 是受管理的服務，因此叢集中沒有要安裝或管理的Kubernetes元件 (例如操作員或自訂資源)。 AWS Batch 您只需要將叢集設定為可與 API 伺服器通訊的角色型存取控制 (RBAC)。 AWS Batch Kubernetes AWS Batch 呼叫 Kubernetes API 以建立、監視和刪除Kubernetes網繭和節點。

AWS Batch 具有內建的縮放邏輯，可根據工作佇列負載調整Kubernetes節點，並在工作容量分配方面進行最佳化。當工作佇列為空時，將節點 AWS Batch 縮減至您設定的最小容量，預設為零。 AWS Batch 管理這些節點的完整生命週期，並用標籤和污點裝飾節點。如此一來，其他Kubernetes工作負載就不會放置在由管理的節點上 AWS Batch。例外情況是`DaemonSets`，它們可以鎖定 AWS Batch 節點，以提供正確執行作業所需的監視和其他功能。此外， AWS Batch 不會在叢集中未管理的節點上執行作業 (特別是網繭)。如此一來，您就可以針對叢集上的其他應用程式使用個別的擴展邏輯和服務。

若要將工作提交給 AWS Batch，您可以直接與 AWS Batch API 互動。 AWS Batch 將任務轉譯為，podspecs然後建立請求，將網繭放置在 Amazon EKS 叢集 AWS Batch 中管理的節點上。您可以使用諸如檢視執行中kubectl的網繭和節點之類的工具。當網繭完成執行時， AWS Batch 會刪除其建立的網繭，以維持較低的Kubernetes系統負載。

您可以透過將有效的 Amazon EKS 叢集與 AWS Batch. 然後將任務 AWS Batch 佇列附加到該佇列，並使用podspec對等屬性註冊 Amazon EKS 任務定義。最後，使用參照工作定義的 SubmitJobAPI 作業來提交工作。如需詳細資訊，請參閱 開始使 AWS Batch 用 Amazon EKS。

Elastic Fabric Adapter

Elastic Fabric Adapter (EFA) 是一種用於加速高效能運算 (HPC) 應用程式的裝置。如果符合以下條件， AWS Batch 支援使用 EFA 的應用程式。

- 如需支援 EFA 的執行個體類型清單，請參閱 Amazon EC2 使用者指南中支援的執行個體類型。

  > **Tip**
  >
  > 若要查看中支援 EFA 的執行個體類型清單 AWS 區域，請執行下列命令。然後，交叉參照 AWS Batch 主控台中可用執行個體類型清單傳回的清單。
  >
  > ```
  > $ aws ec2 describe-instance-types  --region us-east-1  --filters Name=network-info.efa-supported,Values=true  --query "InstanceTypes[*].[InstanceType]"  --output text | sort
  > ```

- 如需支援 EFA 的作業系統清單，請參閱支援的作業系統。
- AMI 已載入 EFA 驅動程式。
- EFA 的安全群組必須允許往返於其本身的所有傳入和傳出流量。
- 所有使用 EFA 的執行個體都必須位於相同的叢集置放群組中。
- 任務定義必須包含 `hostPath` 設定為 `/dev/infiniband/uverbs0` 的 `devices` 成員，以允許 EFA 裝置傳遞到容器。如果`containerPath`已指定，則也必須將其設定為`/dev/infiniband/uverbs0`。如果已設定 `permissions`，它必須設定為 `READ` | `WRITE` | `MKNOD`。

  多節點 parallel 工作和單節點容器工作的`LinuxParameters`成員位置不同。下列範例顯示差異，但缺少必要值。

  Example 多節點平行任務範例

  ```
  {
    "jobDefinitionName": "EFA-MNP-JobDef",
    "type": "multinode",
    "nodeProperties": {
      ...
      "nodeRangeProperties": [
        {
          ...
          "container": {
  ```

```
          ...
          "linuxParameters": {
            "devices": [
              {
                "hostPath": "/dev/infiniband/uverbs0",
                "containerPath": "/dev/infiniband/uverbs0",
                "permissions": [
                    "READ", "WRITE", "MKNOD"
                ]
              },
            ],
          },
        },
      },
    ],
  },
}
```

Example 單一節點容器任務範例

```
{
  "jobDefinitionName": "EFA-Container-JobDef",
  "type": "container",
  ...
  "containerProperties": {
    ...
    "linuxParameters": {
      "devices": [
        {
          "hostPath": "/dev/infiniband/uverbs0",
        },
      ],
    },
  },
}
```

如需有關 EFA 的詳細資訊，請參閱 Amazon EC2 使用者指南中的彈性網狀架構配接器。

AWS BatchIAM 政策、角色和許可

根據預設，使用者沒有使用 AWS Batch API、AWS Batch主控台或AWS CLI. AWS Batch 若要允許使用者執行這些動作，請建立 IAM 政策以授與使用者特定資源和 API 作業的權限。然後，將策略附加到需要這些權限的使用者或群組。

當您將策略附加到使用者或使用者群組時，該策略會允許或拒絕在特定資源上執行特定工作的權限。如需詳細資訊，請參閱 IAM 使用者指南中的許可和政策。如需管理和建立自訂 IAM 政策的詳細資訊，請參閱管理 IAM 政策。

AWS Batch代表您打電話給其他AWS 服務人。因此，AWS Batch必須使用您的認證進行身份驗證。更具體地說，AWS Batch透過建立提供這些許可的 IAM 角色和政策來進行驗證。然後，它會在您建立運算環境時將角色與您的運算環境產生關聯。有關詳情Amazon ECS 執行個體角色，請參閱《IAM 使用者指南》中的 IAM 角色、使用服務連結角色和建立將許可委派給AWS服務的角色。

入門

IAM 政策必須授予或拒絕許可才能使用一或多個AWS Batch動作。

主題

- 政策結構
- AWS Batch API 動作支援的資源層級許可
- 政策範例
- AWS Batch 受管政策
- 建立 AWS Batch IAM 政策
- Amazon ECS 執行個體角色
- Amazon EC2 現貨叢集角色
- EventBridge IAM 角色

政策結構

下列主題說明 IAM 政策的結構。

主題

- 政策語法
- AWS Batch 動作
- AWS Batch 的 Amazon Resource Name
- 檢查使用者擁有必要的許可

政策語法

IAM 政策為包含一或多個陳述式的 JSON 文件。每個陳述式的結構如下所示。

```
{
  "Statement":[{
    "Effect":"effect",
    "Action":"action",
    "Resource":"arn",
    "Condition":{
      "condition":{
    "key":"value"
    }
      }
    }
  ]
}
```

陳述式由各種元素組成：

- Effect (效果)：效果 可以是 `Allow` 或 `Deny`。根據預設，使用者沒有使用資源和 API 動作的權限。因此，所有請求都被拒絕。明確允許覆寫預設值。明確拒絕覆寫任何允許。
- 動作：動作是您授與或拒絕權限的特定 API 動作。如需有關如何指定動作的指示，請參閱AWS Batch 動作。
- Resource (資源)：受動作影響的資源。使用某些 AWS Batch API 動作，您可以在策略中包含可由動作建立或修改的特定資源。若要在陳述式中指定資源，請使用它的 Amazon Resource Name (ARN)。如需詳細資訊，請參閱 AWS Batch API 動作支援的資源層級許可 及 AWS Batch 的 Amazon Resource Name。如果 AWS Batch API 作業目前不支援資源層級權限，請包含萬用字元 (*)，以指定所有資源都可以受到動作的影響。
- Condition (條件)：條件為選擇性。您可以使用它們來控制何時政策開始生效。

如需有關的 IAM 政策陳述式範例的詳細資訊AWS Batch，請參閱建立 AWS Batch IAM 政策。

AWS Batch 動作

在 IAM 政策陳述式中，您可以從任何支援 IAM 的服務指定任何 API 動作。對於AWS Batch，請使用下列前置詞搭配 API 動作的名稱：batch:(例如batch:SubmitJob和batch:CreateComputeEnvironment)。

若要在單一陳述式中指定多個動作，請以逗號分隔每個動作。

```
"Action": ["batch:action1", "batch:action2"]
```

您也可以透過包含萬用字元 (*) 來指定多個動作。例如，您可以指定名稱以「描述」一詞開頭的所有動作。

```
"Action": "batch:Describe*"
```

若要指定所有 AWS Batch API 動作，請包含萬用字元 (*)。

```
"Action": "batch:*"
```

如需AWS Batch動作清單，請參閱 AWS BatchAPI 參考中的動作。

AWS Batch 的 Amazon Resource Name

每個 IAM 政策聲明都適用於您使用其 Amazon 資源名稱 (ARN) 指定的資源。

Amazon 資源名稱（ARN）具有以下一般語法：

```
arn:aws:[service]:[region]:[account]:resourceType/resourcePath
```

服務

服務 (例如，batch)。

region

資源的 (例如，us-east-2)。AWS 區域

account

AWS 帳戶 ID，不含連字號 (例如，123456789012)。

resourceType

資源類型 (例如，compute-environment)。

resourcePath

識別資源的路徑。您可以在路徑中使用萬用字元 (*)。

AWS BatchAPI 作業目前支援數個 API 作業的資源層級權限。如需詳細資訊，請參閱AWS Batch API 動作支援的資源層級許可。若要指定所有資源，或者特定 API 動作不支援 ARN，請在元素中加入萬用字Resource元 (*)。

```
"Resource": "*"
```

檢查使用者擁有必要的許可

在將 IAM 政策投入生產環境之前，請確定該政策授予使用者使用所需特定 API 動作和資源的許可。

若要這麼做，請先建立用於測試目的的使用者，然後將 IAM 政策附加到測試使用者。接著，以測試使用者的身分提出請求。您可以在主控台或 AWS CLI 中提出測試請求。

> **Note**
>
> 您也可以使用 IAM 政策模擬器來測試政策。如需政策模擬器的詳細資訊，請參閱 IAM 使用者指南中的使用 IAM 政策模擬器。

如果政策未授予使用者預期的許可，或授予過多許可，您可以視需要調整政策並重新測試。重新測試，直到您取得所要的結果。

> **Important**
>
> 政策變更的散佈可能需要幾分鐘時間才能生效。因此，我們建議您至少在測試原則更新之前預留五分鐘的時間。

如果授權檢查失敗，請求將傳回包含診斷資訊的編碼訊息。您可使用 DecodeAuthorizationMessage 動作將訊息解碼。如需詳細資訊，請參閱 DecodeAuthorizationMessageAWS Security Token ServiceAPI 參考和AWS CLI命令參考decode-authorization-message中的。

AWS Batch API 動作支援的資源層級許可

「資源層級權限」一詞指的是指定允許使用者對其執行動作之資源的能力。 AWS Batch部分支援資源層級權限。對於某些AWS Batch動作，您可以根據必須符合的條件來控制允許使用者使用這些動作的時間。您也可以根據允許使用者使用的特定資源進行控制。例如，您可以授予使用者提交任務的許可，但僅限特定任務佇列，且僅能藉由特定的任務定義來達成。

下列清單說明目前支援資源層級權限的 AWS Batch API 動作。此清單也說明每個動作的支援資源、資源 ARN 和條件索引鍵。

> **Important**
>
> 如果 AWS Batch API 動作未列在此清單中，則不支援資源層級權限。如果 AWS Batch API 動作不支援資源層級權限，您可以授與使用者使用該動作的權限。但是，您必須為政策聲明的資源元素包含萬用字元 (*)。

動作

CancelJob, CreateComputeEnvironment, CreateJobQueue, CreateSchedulingPolicy, DeleteComputeEnvironment, DeleteJobQueue, DeleteSchedulingPolicy, DeregisterJobDefinition, ListTagsForResource, RegisterJobDefinition, SubmitJob, TagResource, TerminateJob, UntagResource, UpdateComputeEnvironment, UpdateSchedulingPolicy, UpdateJobQueue

CancelJob

取消AWS Batch佇列中的工作。

Resource

任務

arn: AW: ##:##:##:##/jobId

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

CreateComputeEnvironment

建立AWS Batch運算環境。

Resource

運算環境

arn: awn: ##:##:##:####/compute-environment-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

條件鍵

aws:RequestTag/${TagKey} (字串)

根據要求中傳遞的標籤篩選動作。

aws:TagKeys (字串)

根據要求中傳遞的標籤鍵篩選動作。

CreateJobQueue

建立AWS Batch工作佇列。

Resource

運算環境

arn: awn: ##:##:##:####/compute-environment-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

Job 佇列

arn: aws: ##:##:##:####/####

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

排程原則

arn: awn: ##:##:##:####/scheduling-policy-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

條件鍵

aws:RequestTag/${TagKey} (字串)

根據要求中傳遞的標籤篩選動作。

aws:TagKeys (字串)

根據要求中傳遞的標籤鍵篩選動作。

DeleteComputeEnvironment

刪除計AWS Batch算環境。

Resource

運算環境

arn: awn: ##:##:##:####/compute-environment-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

CreateSchedulingPolicy

建立AWS Batch排程原則。

Resource

排程原則

arn: awn: ##:##:##:####/scheduling-policy-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

條件鍵

aws:RequestTag/${TagKey} (字串)

根據要求中傳遞的標籤篩選動作。

aws:TagKeys (字串)

根據要求中傳遞的標籤鍵篩選動作。

DeleteJobQueue

刪除指定的任務佇列。刪除工作佇列最後會刪除佇列中的所有工作。刪除工作的速率約為每秒 16 個工作。

Resource

Job 佇列

arn: aws: ##:##:##:####/####

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

DeleteSchedulingPolicy

刪除指定的排程策略。

Resource

排程原則

arn: awn: ##:##:##:####/scheduling-policy-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

DeregisterJobDefinition

取消註冊AWS Batch工作定義。

Resource

Job 定義

arn: aws: ##:##:##:####/####:###

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

ListTagsForResource

列出指定資源的標籤。

Resource

運算環境

arn: awn: ##:##:##:####/compute-environment-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

任務

arn: AW: ##:##:##:##/jobId

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

Job 定義

arn: aws: ##:##:##:####/####:###

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

Job 佇列

arn: aws: ##:##:##:####/####

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

排程原則

arn: awn: ##:##:##:####/scheduling-policy-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

RegisterJobDefinition

註冊一個AWS Batch定義。

Resource

Job 定義

arn: awn: ##:##:##:####/####

條件鍵

batch:AWSLogsCreateGroup(布林值)

當此參數為 true 時，便會awslogs-group為記錄檔建立。

batch:AWSLogsGroup (字串)

記錄檔所在的awslogs群組。

batch:AWSLogsRegion (字串)

記錄傳送目的地的地區。

batch:AWSLogsStreamPrefix (字串)

日awslogs誌流前綴。

batch:Image (字串)

用來啟動工作的泊塢視窗影像。

batch:LogDriver (字串)

用於工作的記錄驅動程式。

batch:Privileged(布林值)

當此參數為 true 時，工作的容器會在主機容器執行個體上獲得提升的權限。

batch:User (字串)

要在工作容器內使用的使用者名稱或數字 uid。

aws:RequestTag/${TagKey} (字串)

根據要求中傳遞的標籤篩選動作。

aws:TagKeys (字串)

根據要求中傳遞的標籤鍵篩選動作。

SubmitJob

從AWS Batch工作定義提交工作。

Resource

任務

arn: AW: ##:##:##:##/jobId

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

Job 定義

arn: aws: ##:##:##:####/#### [: ##]

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

> **Note**
>
> 只有在任務定義 Amazon 資源名稱 (ARN) 的格式為時，才能使用此金鑰 arn:aws:batch:*region*:*account_number*:job-definition/*definition-name*:*revision*.

Job 佇列

arn: aws: ##:##:##:####/####

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

TagResource

標記指定的資源。

Resource

運算環境

arn: awn: ##:##:##:####/compute-environment-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

任務

arn: AW: ##:##:##:##/jobId

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

Job 定義

arn: aws: ##:##:##:####/####:###

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

Job 佇列

arn: aws: ##:##:##:####/####

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

排程原則

arn: awn: ##:##:##:####/scheduling-policy-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

條件鍵

aws:RequestTag/${TagKey} (字串)

根據要求中傳遞的標籤篩選動作。

aws:TagKeys (字串)

根據要求中傳遞的標籤鍵篩選動作。

TerminateJob

終止工作佇列中的AWS Batch工作。

Resource

任務

arn: AW: ##:##:##:##/jobId

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

UntagResource

取消標記指定的資源。

Resource

運算環境

arn: awn: ##:##:##:####/compute-environment-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

任務

arn: AW: ##:##:##:##/jobId

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

Job 定義

arn: aws: ##:##:##:####/####:###

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

Job 佇列

arn: aws: ##:##:##:####/####

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

排程原則

arn: awn: ##:##:##:####/scheduling-policy-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

條件鍵

aws:TagKeys (字串)

根據要求中傳遞的標籤鍵篩選動作。

UpdateComputeEnvironment

更新AWS Batch運算環境。

Resource

運算環境

arn: awn: ##:##:##:####/compute-environment-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

UpdateJobQueue

更新任務佇列。

Resource

Job 佇列

arn: aws: ##:##:##:####/####

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

排程原則

arn: awn: ##:##:##:####/scheduling-policy-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

UpdateSchedulingPolicy

更新排程原則。

Resource

排程原則

arn: awn: ##:##:##:####/scheduling-policy-name

條件鍵

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

AWS BatchAPI 動作的條件金鑰

AWS Batch定義 IAM 政策Condition元素中使用的下列條件金鑰。您可以使用這些索引鍵來調整套用原則陳述式的條件。若要檢視所有服務可用的全域條件金鑰，請參閱 IAM 使用者指南中的可用全域條件金鑰。

batch:AWSLogsCreateGroup(布林值)

當此參數為 true 時，便會awslogs-group為記錄檔建立。

batch:AWSLogsGroup (字串)

記錄檔所在的awslogs群組。

batch:AWSLogsRegion (字串)

記錄檔傳送到的AWS 區域位置。

batch:AWSLogsStreamPrefix (字串)

日awslogs誌流前綴。

batch:Image (字串)

用來啟動工作的泊塢視窗影像。

batch:LogDriver (字串)

用於工作的記錄驅動程式。

batch:Privileged(布林值)

當此參數為 true 時，工作的容器會在主機容器執行個體上獲得更高的權限 (類似於 root 使用者)。

aws:ResourceTag/${TagKey} (字串)

根據與資源相關聯的標籤篩選動作。

aws:RequestTag/${TagKey} (字串)

根據要求中傳遞的標籤篩選動作。

batch:ShareIdentifier (字串)

根據傳送至的shareIdentifier參數篩選動作SubmitJob。

aws:TagKeys (字串)

根據要求中傳遞的標籤鍵篩選動作。

batch:User (字串)

要在工作容器內使用的使用者名稱或數字使用者 ID (uid)。

政策範例

下列範例顯示可用來控制使用者擁有之權限的原則陳述式AWS Batch。

範例

- 唯讀存取
- 在工作提交時限制為 POSIX 使用者、Docker 映像檔、權限等級和角色
- 工作提交時限制為工作定義字首
- 限制為工作佇列
- 當條件所有鍵匹配字符串時拒絕操作
- 當任何條件鍵符合字串時拒絕動作
- 使用batch:ShareIdentifier條件鍵

唯讀存取

下列政策授予使用者使用名稱開頭為`Describe`和的所有 AWS Batch API 動作的權限`List`。

除非另一個陳述式授予他們這樣做的權限，否則使用者無權對資源執行任何動作。默認情況下，他們被拒絕使用 API 操作的權限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "batch:Describe*",
                "batch:List*"
            ],
            "Resource": "*"
        }
    ]
}
```

在工作提交時限制為 POSIX 使用者、Docker 映像檔、權限等級和角色

下列原則可讓 POSIX 使用者管理自己的一組受限工作定義。

########################## A_ ##########JobDef

第一個陳述式也使用條件式內容金鑰來限制任務定義內 containerProperties 的 POSIX 使用者、權限狀態，以及容器映像值。如需詳細資訊，請參閱《AWS Batch API 參考》 中的 RegisterJobDefinition。在此範例中，工作定義只能在 POSIX 使用者設定為nobody時註冊。特殊權限旗標設定為false。最後，在 Amazon ECR 儲存庫myImage中將映像設定為。

> **Important**
>
> Docker 會uid從容器映像檔內將user參數解析為該使用者。在大多數情況下，這可以在容器映像中的/etc/passwd文件中找到。透過在工作定義和任何相關聯的 IAM 政策中使用直接uid值，即可避免此名稱解析。AWS BatchAPI 作業和 batch:User IAM 條件金鑰都支援數值。

使用第三個陳述式，限制只有工作定義的特定角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "batch:RegisterJobDefinition"
            ],
            "Resource": [
                "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/JobDefA_*"
            ],
            "Condition": {
                "StringEquals": {
                    "batch:User": [
                        "nobody"
                    ],
                    "batch:Image": [
                        "<aws_account_id>.dkr.ecr.<aws_region>.amazonaws.com/myImage"
                    ]
                },
                "Bool": {
                    "batch:Privileged": "false"
                }
            }
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "batch:DeregisterJobDefinition"
            ],
            "Resource": [
                "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/JobDefA_*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::<aws_account_id>:role/MyBatchJobRole"
            ]
        }
    ]
}
```

工作提交時限制為工作定義字首

使用下列原則，將工作提交至任何以 *JobDefA* 開頭的工作定義名稱的工作佇列。

> **Important**
>
> 在限制任務提交的資源層級存取範圍時，您必須同時提供任務佇列和任務定義資源類型。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:SubmitJob"
      ],
      "Resource": [
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/JobDefA_*",
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-queue/*"
      ]
```

```
    }
  ]
}
```

限制為工作佇列

使用下列原則，將工作送至具有任何工作定義名稱為 queue1 的特定工作佇列。

> **Important**
>
> 在限制任務提交的資源層級存取範圍時，您必須同時提供任務佇列和任務定義資源類型。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:SubmitJob"
      ],
      "Resource": [
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/*",
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-queue/queue1"
      ]
    }
  ]
}
```

當條件所有鍵匹配字符串時拒絕操作

 # batch:Image (#### ID) #######string1## (#########) #######string2####### ##### RegisterJobDefinitionAPI ###batch:LogDriver AWS Batch評估每個容器上的條件鍵。當工作跨越多個容器 (例如多節點 parallel 作業) 時，容器可能會有不同的組態。如果在一個陳述式中評估多個條件索引鍵，則會使用AND邏輯來組合它們。因此，如果多個條件索引鍵中的任何一個不符合容器，則不會對該容器套用此Deny效果。相反地，相同工作中的不同容器可能會遭到拒絕。

如需的條件金鑰清單AWS Batch，請參閱服務授權參考AWS Batch中的條件金鑰。除了batch:ShareIdentifier，所有batch條件鍵都可以用這種方式使用。batch:ShareIdentifier條件鍵是為工作定義的，而不是工作定義。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Deny",
      "Action": "batch:RegisterJobDefinition",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "batch:Image": "string1",
          "batch:LogDriver": "string2"
        }
      }
    }
  ]
}
```

當任何條件鍵符合字串時拒絕動作

*# batch:Image (#### ID) #######string1## (#########) #######string2###### ###### RegisterJobDefinitionAPI ###batch:LogDriver*當工作跨越多個容器 (例如多節點 parallel 作業) 時，容器可能會有不同的組態。如果在一個陳述式中評估多個條件索引鍵，則會使用AND邏輯來組合它們。因此，如果多個條件索引鍵中的任何一個不符合容器，則不會對該容器套用此Deny效果。相反地，相同工作中的不同容器可能會遭到拒絕。

如需的條件金鑰清單AWS Batch，請參閱服務授權參考AWS Batch中的條件金鑰。除了batch:ShareIdentifier，所有batch條件鍵都可以用這種方式使用。batch:ShareIdentifier條件鍵是為工作定義的，而不是工作定義。)

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Deny",
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "batch:Image": [
            "string1"
          ]
        }
      }
    },
    {
      "Effect": "Deny",
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "batch:LogDriver": [
            "string2"
          ]
        }
      }
    }
  ]
}
```

使用batch:ShareIdentifier條件鍵

使用下列原則，將使用工jobDefA作定義的工作提交至具有lowCpu共用識別碼的jobqueue1工作佇列。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:SubmitJob"
      ],
      "Resource": [
        "arn:aws::batch:<aws_region>:<aws_account_id>:job-definition/JobDefA",
        "arn:aws::batch:<aws_region>:<aws_account_id>:job-queue/jobqueue1"
      ],
      "Condition": {
        "StringEquals": {
          "batch:ShareIdentifier": [
            "lowCpu"
          ]
        }
      }
    }
  ]
}
```

AWS Batch 受管政策

AWS Batch提供受管理的原則，您可以附加至使用者，以提供使用AWS Batch資源和 API 作業的權限。您可以直接套用此政策，或用它做為起點來建立您自己的政策。如需這些政策中提及之每個 API 作業的詳細資訊，請參閱 AWS BatchAPI 參考中的動作。

AWSBatchFullAccess

此政策允許完整的 AWS Batch 管理員存取權。

```
{
  "Version":"2012-10-17",
  "Statement":[
```

```
    {
      "Effect":"Allow",
      "Action":[
        "batch:*",
        "cloudwatch:GetMetricStatistics",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeVpcs",
        "ec2:DescribeImages",
        "ec2:DescribeLaunchTemplates",
        "ec2:DescribeLaunchTemplateVersions",
        "ecs:DescribeClusters",
        "ecs:Describe*",
        "ecs:List*",
        "eks:DescribeCluster",
        "eks:ListClusters",
        "logs:Describe*",
        "logs:Get*",
        "logs:TestMetricFilter",
        "logs:FilterLogEvents",
        "iam:ListInstanceProfiles",
        "iam:ListRoles"
      ],
      "Resource":"*"
    },
    {
      "Effect":"Allow",
      "Action":[
        "iam:PassRole"
      ],
      "Resource":[
        "arn:aws:iam::*:role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/service-role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/ecsInstanceRole",
        "arn:aws:iam::*:instance-profile/ecsInstanceRole",
        "arn:aws:iam::*:role/iaws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/aws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/AWSBatchJobRole*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
```

```
        "iam:CreateServiceLinkedRole"
      ],
      "Resource":"arn:aws:iam::*:role/*Batch*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "batch.amazonaws.com"
          }
      }
    }
  ]
}
```

建立 AWS Batch IAM 政策

您可以建立特定的 IAM 政策，以限制帳戶中使用者可存取的呼叫和資源。然後，您可以將這些策略附加到使用者。

當您將原則附加至使用者或使用者群組時，此原則會允許或拒絕使用者針對特定資源上特定工作的權限。如需詳細資訊，請參閱 IAM 使用者指南中的許可和政策。如需如何管理和建立自訂 IAM 政策的指示，請參閱管理 IAM 政策。

Amazon ECS 執行個體角色

AWS Batch運算環境中會填入 Amazon ECS 容器執行個體。他們在本機執行 Amazon ECS 容器代理程式。Amazon ECS 容器代理程式會代表您呼叫各種 AWS API 作業。因此，執行代理程式的容器執行個體需要這些服務的 IAM 政策和角色，才能辨識該代理程式屬於您。您必須建立 IAM 角色和執行個體設定檔，以便容器執行個體啟動時使用。否則，您無法建立運算環境並在其中啟動容器執行個體。此要求適用於使用或不使用 Amazon 提供的 Amazon ECS 優化 AMI 啟動的容器執行個體。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 Amazon ECS 容器執行個體 IAM 角色。

Amazon ECS 執行個體角色和執行個體設定檔會在主控台首次執行體驗中自動為您建立。但是，您可以按照以下步驟檢查您的帳戶是否已具有 Amazon ECS 執行個體角色和執行個體設定檔。下列步驟也涵蓋如何附加受管 IAM 政策。

在 IAM 主控台中檢查 **`ecsInstanceRole`**

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇角色。
3. 搜尋 `ecsInstanceRole` 的角色清單。如果角色不存在，請使用下列步驟建立角色。

a. 選擇建立角色。

b. 對於 Trusted entity type (信任的實體類型)，請選擇 AWS 服務。

c. 對於常見使用案例，請選擇 EC2。

d. 選擇下一步。

e. 對於許可政策，請搜索亞馬遜 EC2 角色。ContainerServicefor

f. 選擇亞馬遜 ContainerServicefor EC2Role 旁邊的核取方塊，然後選擇下一步。

g. 針對 Role Name (角色名稱)，輸入 `ecsInstanceRole`，然後選擇 Create Role (建立角色)。

> **Note**
>
> 如果您使用AWS Management Console為 Amazon EC2 建立角色，則主控台會建立與該角色名稱相同的執行個體設定檔。

或者，您可以使AWS CLI用建立 `ecsInstanceRole` IAM 角色。下列範例會建立具有信任政策和AWS受管政策的 IAM 角色。

建立 IAM 角色和執行個體設定檔 (AWS CLI)

1. 建立下列信任原則，並將其儲存在名為的文字檔中`ecsInstanceRole-role-trust-policy.json`。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": { "Service": "ec2.amazonaws.com"},
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. 使用「創建角色」命令來創建角色。`ecsInstanceRole`在`assume-role-policy-document`參數中指定信任原則檔案位置。

```
$ aws iam create-role \
```

```
    --role-name ecsInstanceRole \
    --assume-role-policy-document file://ecsInstanceRole-role-trust-policy.json
```

以下是回應範例。

```
{
    "Role": {
         "Path": "/",
         "RoleName: "ecsInstanceRole",
         "RoleId": "AROAT46P5RDIY4EXAMPLE",
         "Arn": "arn:aws:iam::123456789012:role/ecsInstanceRole".
        "CreateDate": "2022-12-12T23:46:37.247Z",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service: "ec2.amazonaws.com"
                    }
                    "Action": "sts:AssumeRole",
                 }
              ]
          }
        }
```

3. 使用指create-instance-profile令建立名為的執行個體設定檔ecsInstanceRole。

> **Note**
>
> 您需要在和 AWS API 中將角色和執行個體設定檔建立為個別動作。AWS CLI

```
$ aws iam create-instance-profile --instance-profile-name ecsInstanceRole
```

以下是回應範例。

```
{
    "InstanceProfile": {
        "Path": "/",
        "InstanceProfileName": "ecsInstanceRole",
```

```
        "InstanceProfileId": "AIPAT46P5RDITREXAMPLE",
        "Arn": "arn:aws:iam::123456789012:instance-profile/ecsInstanceRole",
        "CreateDate": "2022-06-30T23:53:34.093Z",
        "Roles": [],    }
}
```

4. 使用 add-role-to-instance-profile 指令將ecsInstanceRole角色新增至ecsInstanceRole執行個體設定檔。

```
aws iam add-role-to-instance-profile \
    --role-name ecsInstanceRole --instance-profile-name ecsInstanceRole
```

5. 使用命attach-role-policy令將AmazonEC2ContainerServiceforEC2RoleAWS受管理的策略附加到ecsInstanceRole角色。

```
$ aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role \
    --role-name ecsInstanceRole
```

Amazon EC2 現貨叢集角色

如果您建立使用 Amazon EC2 Spot 叢集執行個體的受管運算環境，則必須建立AmazonEC2SpotFleetTaggingRole政策。此原則授與 Spot 叢集權限，以代表您啟動、標記和終止執行個體。在您的 Spot Fleet 請求中指定角色。您還必須擁有 Amazon EC2 競價型AWSServiceRoleForEC2Spot和競價型叢集的和AWSServiceRoleForEC2SpotFleet服務連結角色。請使用下列指示來建立所有這些角色。如需詳細資訊，請參閱 IAM 使用者指南中的使用服務連結角色和建立將權限委派給AWS服務的角色。

主題

- 在中建立 Amazon EC2 現貨叢集角色 AWS Management Console
- 建立 Amazon EC2 競價型叢集角色 AWS CLI

在中建立 Amazon EC2 現貨叢集角色 AWS Management Console

為 Amazon EC2 競價型叢集建立 **AmazonEC2SpotFleetTaggingRole** IAM 服務連結角色

1. 開啟位於 https://console.aws.amazon.com/iam/ 的 IAM 主控台。

2. 對於「存取管理」，選擇「角色」
3. 對於角色，請選擇建立角色。
4. 從 [選取信任的實體類型的信任實體] 中，選擇AWS 服務。
5. 對於其他使用案例 AWS 服務，請選擇 EC2，然後選擇 [EC2-競價型叢集標記]。
6. 選擇下一步。
7. 從 [原則名稱] 的 [權限原則] 中，確認AmazonEC2SpotFleetTaggingRole。
8. 選擇下一步。
9. 對於「名稱」、「檢閱」和「建立」：

    a. 在角色名稱中，輸入用於識別角色的名稱。
    b. 在說明中，輸入策略的簡短說明。
    c. (選擇性) 對於步驟 1：選取信任的實體，請選擇編輯以修改程式碼。
    d. (選擇性) 對於步驟 2：新增權限，請選擇編輯以修改程式碼。
    e. (選擇性) 在 [新增標籤] 中，選擇 [新增標籤] 以將標籤新增至資源。
    f. 選擇建立角色。

> **Note**
>
> 在過去，Amazon EC2 競價型叢集角色有兩個受管政策。
>
> - AmazonEC2 SpotFleetRole：這是競價型叢集角色的原始受管政策。不過，我們不再建議您搭配使用它AWS Batch。此原則不支援在運算環境中使用AWSServiceRoleForBatch服務連結角色所必須的 Spot 執行個體標記。如果您先前使用此原則建立 Spot 叢集角色，請將新建議的原則套用至該角色。如需詳細資訊，請參閱建立時未標記競價型執行個體。
> - 亞馬遜 EC2 SpotFleetTaggingRole：此角色提供標記 Amazon EC2 競價型執行個體的所有必要許可。使用此角色，便能在 AWS Batch 運算環境中執行 Spot 執行個體標記。

建立 Amazon EC2 競價型叢集角色 AWS CLI

為您的競價型叢集運算環境建立亞馬遜 SpotFleetTaggingRole IAM 角色

1. 使用執行下列命令AWS CLI。

```
$ aws iam create-role --role-name AmazonEC2SpotFleetTaggingRole \
```

```
    --assume-role-policy-document '{
    "Version":"2012-10-17",
    "Statement":[
       {
         "Sid":"",
         "Effect":"Allow",
         "Principal": {
           "Service":"spotfleet.amazonaws.com"
          },
          "Action":"sts:AssumeRole"
        }
      ]
    }'
```

2. 若要將亞馬遜 EC2 SpotFleetTaggingRole 受管身分與存取權管理政策附加到您的 AmazonEC2 SpotFleetTaggingRole 角色，請使用. AWS CLI

```
$ aws iam attach-role-policy \
  --policy-arn \
    arn:aws:iam::aws:policy/service-role/AmazonEC2SpotFleetTaggingRole \
  --role-name \
    AmazonEC2SpotFleetTaggingRole
```

若要為 Amazon EC2 競價型建立 **AWSServiceRoleForEC2Spot** IAM 服務連結角色

> **Note**
>
> 如果 AWSServiceRoleForEC2Spot IAM 服務連結角色已存在，您會看到類似下列的錯誤訊息。
>
> ```
> An error occurred (InvalidInput) when calling the CreateServiceLinkedRole
>  operation:
> Service role name AWSServiceRoleForEC2Spot has been taken in this account,
>  please try a different suffix.
> ```

- 使用執行下列命令AWS CLI。

```
$ aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
```

為 Amazon EC2 競價型叢集建立 **AWSServiceRoleForEC2SpotFleet** IAM 服務連結角色

> **Note**
>
> 如果 AWSServiceRoleForEC2SpotFleet IAM 服務連結角色已存在，您會看到類似下列的錯誤訊息。
>
> ```
> An error occurred (InvalidInput) when calling the CreateServiceLinkedRole
>  operation:
> Service role name AWSServiceRoleForEC2SpotFleet has been taken in this account,
>  please try a different suffix.
> ```

- 使用執行下列命令AWS CLI。

```
$ aws iam create-service-linked-role --aws-service-name spotfleet.amazonaws.com
```

EventBridge IAM 角色

Amazon EventBridge 提供近乎即時的系統事件串流，用來描述AWS資源變更。 AWS Batch工作可作為 EventBridge 目標使用。利用可快速設定的簡單規則，匹配事件並提交 AWS Batch 任務以回應事件。您必須具有代表您執行AWS Batch工作的權限， EventBridge 才能提交具有 EventBridge 規則和目標的AWS Batch工作。

> **Note**
>
> 在將AWS Batch佇列指定為目標的 EventBridge 主控台中建立規則時，您可以建立此角色。如需範例演練，請參閱 AWS Batch 作為 EventBridge 目標的工作。您可以使用 EventBridge IAM 主控台手動建立角色。如需指示，請參閱《IAM 使用者指南》中的使用自訂信任政策 (主控台) 建立角色。

EventBridge IAM 角色的信任關係必須為events.amazonaws.com服務主體提供擔任該角色的能力。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

確保附加到 EventBridge IAM 角色的政策允許您的資源`batch:SubmitJob`許可。在下列範例中，AWS Batch提供`AWSBatchServiceEventTargetRole`受管理的原則來提供這些權限。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:SubmitJob"
       ],
      "Resource": "*"
    }
  ]
}
```

AWS Batch Amazon 事件流 EventBridge

您可以使用 Amazon 的 AWS Batch 事件串流 EventBridge 接收有關任務佇列中目前任務狀態的近乎即時的通知。

您可以使用 EventBridge 來獲得有關 AWS Batch 服務的進一步見解。更具體地說，您可以使用它來檢查工作進度，構建自 AWS Batch 定義工作流程，生成使用情況報告或指標，或構建自己的儀表板。使用 AWS Batch 和 EventBridge，您不需要排程和監控程式碼，來持續輪詢工 AWS Batch 作狀態變更。相反地，您可以使用各種 Amazon EventBridge 目標以非同步方式處理 AWS Batch 任務狀態變更。其中包括 AWS Lambda Amazon 簡單佇列服務、Amazon 簡單通知服務或 Amazon Kinesis Data Streams。

AWS Batch 事件串流中的事件會確保至少傳送一次。如果傳送重複的事件，則事件會提供足夠的資訊來識別重複項目。如此一來，您就可以比較事件的時間戳記和工作狀態。

AWS Batch 工作可作為 EventBridge 目標使用。使用簡單的規則，您可以匹配事件並提交 AWS Batch 作業以響應它們。如需詳細資訊，請參閱什麼是 EventBridge？ 在 Amazon 用 EventBridge 戶指南。您也可以使用 EventBridge 來排程使用cron或評估運算式在特定時間自行觸發的自動動作。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的建立按排程執行的 Amazon EventBridge 規則。如需範例演練，請參閱 AWS Batch 作為 EventBridge 目標的工作。如需使用 EventBridge 排程器的相關資訊，請參閱 Amazon 使用 EventBridge 者指南中的設定 Amazon EventBridge 排程器。

主題

- AWS Batch 活動
- 使用使用 AWS 者通知 AWS Batch
- AWS Batch 作為 EventBridge 目標的工作
- 教學課程：聆聽 AWS Batch EventBridge
- 教學：針對失敗的 Job 務事件傳送 Amazon 簡易通知服務警示

AWS Batch 活動

AWS Batch 將工作狀態變更事件傳送至 EventBridge。 AWS Batch 追蹤工作的狀態。如果先前提交的工作狀態變更，則會叫用事件。例如，如果狀RUNNING態中的工作移至狀FAILED態。這些事件便歸類為任務狀態變更事件。

> **Note**
>
> AWS Batch future 可能會新增其他事件類型、來源和詳細資料。如果您以程式設計方式還原序列化事件 JSON 資料，請確定您的應用程式已準備好處理未知的屬性。這是為了避免在添加這些附加屬性時出現問題。

任務狀態變更事件

只要現有 (先前提交的) 工作變更狀態，就會建立事件。如需 AWS Batch 工作狀態的詳細資訊，請參閱任務狀態。

> **Note**
>
> 不會為初始工作提交建立事件。

Example 任務狀態變更事件

Job 狀態變更事件會以下列格式傳遞。此`detail`區段類似於 API 參考中 DescribeJobsAPI 作業傳回的JobDetail物件。AWS Batch 如需有關 EventBridge 參數的詳細資訊，請參閱 Amazon EventBridge 使用者指南中的事件和事件模式。

```
{
    "version": "0",
    "id": "c8f9c4b5-76e5-d76a-f980-7011e206042b",
    "detail-type": "Batch Job State Change",
    "source": "aws.batch",
    "account": "123456789012",
    "time": "2022-01-11T23:36:40Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-ba5a-4727fcce14a8"
    ],
    "detail": {
        "jobArn": "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-
ba5a-4727fcce14a8",
        "jobName": "event-test",
        "jobId": "4c7599ae-0a82-49aa-ba5a-4727fcce14a8",
        "jobQueue": "arn:aws:batch:us-east-1:123456789012:job-queue/
PexjEHappyPathCanary2JobQueue",
```

```
        "status": "RUNNABLE",
        "attempts": [],
        "createdAt": 1641944200058,
        "retryStrategy": {
            "attempts": 2,
            "evaluateOnExit": []
        },
        "dependsOn": [],
        "jobDefinition": "arn:aws:batch:us-east-1:123456789012:job-definition/first-
run-job-definition:1",
        "parameters": {},
        "container": {
            "image": "137112412989.dkr.ecr.us-east-1.amazonaws.com/amazonlinux:latest",
            "command": [
                "sleep",
                "600"
            ],
            "volumes": [],
            "environment": [],
            "mountPoints": [],
            "ulimits": [],
            "networkInterfaces": [],
            "resourceRequirements": [
                {
                    "value": "2",
                    "type": "VCPU"
                }, {
                    "value": "256",
                    "type": "MEMORY"
                }
            ],
            "secrets": []
        },
        "tags": {
            "resourceArn": "arn:aws:batch:us-
east-1:123456789012:job/4c7599ae-0a82-49aa-ba5a-4727fcce14a8"
        },
        "propagateTags": false,
        "platformCapabilities": []
    }
}
```

Job 佇列封鎖的事件

只要 AWS Batch 偵測到RUNNABLE狀態中的工作並因此封鎖佇列，就會在 Amazon E CloudWatch vents 中建立事件。如需支援之封鎖佇列原因的相關資訊，請參閱封鎖的工作佇列訊息範例。DescribeJobsAPI 動作的statusReason欄位中也有相同的原因。

Example 任務狀態變更事件

Job 狀態變更事件會以下列格式傳遞。此detail區段類似於 API 參考中 DescribeJobsAPI 作業傳回的JobDetail物件。AWS Batch 如需有關 EventBridge參數的詳細資訊，請參閱 Amazon EventBridge 使用者指南中的事件和事件模式。

```
{
    "version": "0",
    "id": "c8f9c4b5-76e5-d76a-f980-7011e206042b",
    "detail-type": "Batch Job Queue Blocked",
    "source": "aws.batch",
    "account": "123456789012",
    "time": "2022-01-11T23:36:40Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-
ba5a-4727fcce14a8",
        "arn:aws:batch:us-east-1:123456789012:job-queue/PexjEHappyPathCanary2JobQueue"
    ],
    "detail": {
        "jobArn": "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-
ba5a-4727fcce14a8",
        "jobName": "event-test",
        "jobId": "4c7599ae-0a82-49aa-ba5a-4727fcce14a8",
        "jobQueue": "arn:aws:batch:us-east-1:123456789012:job-queue/
PexjEHappyPathCanary2JobQueue",
        "status": "RUNNABLE",
        "statusReason": "blocked-reason"
        "attempts": [],
        "createdAt": 1641944200058,
        "retryStrategy": {
            "attempts": 2,
            "evaluateOnExit": []
        },
        "dependsOn": [],
        "jobDefinition": "arn:aws:batch:us-east-1:123456789012:job-definition/first-
run-job-definition:1",
```

```
        "parameters": {},
        "container": {
            "image": "137112412989.dkr.ecr.us-east-1.amazonaws.com/amazonlinux:latest",
            "command": [
                "sleep",
                "600"
            ],
            "volumes": [],
            "environment": [],
            "mountPoints": [],
            "ulimits": [],
            "networkInterfaces": [],
            "resourceRequirements": [
                {
                    "value": "2",
                    "type": "VCPU"
                }, {
                    "value": "256",
                    "type": "MEMORY"
                }
            ],
            "secrets": []
        },
        "tags": {
            "resourceArn": "arn:aws:batch:us-
east-1:123456789012:job/4c7599ae-0a82-49aa-ba5a-4727fcce14a8"
        },
        "propagateTags": false,
        "platformCapabilities": []
    }
}
```

使用使用 AWS 者通知 AWS Batch

您可以使用「使用AWS 者通知」 來設定傳送管道，以接收有關 AWS Batch 事件的通知。當事件符合您指定的規則時，便會收到通知。您可以透過多個管道接收事件通知，包括電子郵件、AWS Chatbot 聊天通知或 AWS Console Mobile Application 推送通知。您也可以在主控台通知中心查看通知。使用者通知支援彙總，可減少您在特定事件期間收到的通知數目。

若要在中設定使用者通知 AWS Batch：

1. 開啟 AWS Batch 主控台。

2. 選擇 Dashboard (儀表板)。
3. 選擇「設定通知」。
4. 在 [AWS 使用者通知] 中，選擇 [建立通知組態

如需如何設定及檢視使用者通知的相關資訊，請參閱開始 AWS 使用使用者通知。

AWS Batch 作為 EventBridge 目標的工作

亞馬遜 EventBridge提供近乎即時的系統事件串流，用於描述 Amazon Web 服務資源中的變更。通常， AWS Batch 在 Amazon 彈性容器服務上，Amazon Elastic Kubernetes Service 和 AWS Fargate 任務可作為目標使用。 EventBridge 使用簡單的規則，您可以匹配事件並提交 AWS Batch 作業以響應它們。如需詳細資訊，請參閱什麼是 EventBridge？ 在 Amazon 用 EventBridge 戶指南。

您也可以使用 EventBridge 來排程使用cron或評估運算式在特定時間叫用的自動動作。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的建立按排程執行的 Amazon EventBridge 規則。

有關如何建立在事件符合事件模式時執行的規則的詳細資訊，請參閱 Amazon EventBridge 使用者指南中的建立對事件做出反應的 Amazon EventBridge 規則。

作為 EventBridge 目標的 AWS Batch 工作常見使用案例包括下列使用案例：

- 排定的工作會以固定的時間間隔執行。例如，只有在 Amazon EC2 競價型cron執行個體較便宜的情況下，才會在低使用時間內執行任務。
- AWS Batch 工作會執行以回應已登入的 API 作業 CloudTrail。例如，每當物件上傳到指定的 Amazon S3 儲存貯體時，就會提交任務。每次發生這種情況時， EventBridge 輸入轉換器都會將對象的存儲桶和密鑰名稱傳遞給 AWS Batch 參數。

> **Note**
>
> 在這個案例中，所有相關 AWS 資源都必須位於相同的區域中。這包括 Amazon S3 儲存貯體、 EventBridge 規則和 CloudTrail 日誌等資源。

在您可以提交具有 EventBridge 規則和目標的 AWS Batch 工作之前， EventBridge 服務需要數個權限才能執行 AWS Batch 工作。在將 AWS Batch 工作指定為目標的 EventBridge 主控台中建立規則時，您也可以建立此角色。如需有關此角色必要的服務主體和 IAM 權限的詳細資訊，請參閱 EventBridge IAM 角色。

建立排定的 AWS Batch 工作

下列程序涵蓋如何建立排程 AWS Batch 工作和所需的 EventBridge IAM 角色。

若要建立排定的 AWS Batch 工作 EventBridge

> **Note**
>
> 此程序適用 AWS Batch 於所有 Amazon ECS、Amazon EKS 和 AWS Fargate 工作。

1. 在以下位置打開 Amazon EventBridge 控制台 https://console.aws.amazon.com/events/。
2. 從導覽列中，選取 AWS 區域 要使用的。
3. 在導覽窗格中，選擇規則。
4. 選擇建立規則。
5. 對於「名稱」，請為您的計算環境指定唯一的名稱。名稱最多可包含 64 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。

   > **Note**
   >
   > 在同一個區域和同一個事件匯流排上，規則不能與另一個規則同名。

6. (選擇性) 在說明中，輸入規則的說明。
7. 針對事件匯流排，選擇要與此規則建立關聯的事件匯流排。如果您想要此規則匹配來自您的帳戶的事件，請選取預設值。當您帳戶 AWS 服務 中的某個事件發出時，它始終會進入您帳戶的默認事件總線。
8. (選擇性) 如果您不想立即執行規則，請關閉所選匯流排上的規則。
9. 針對規則類型，選擇排程。
10. 選擇「繼續」建立規則或「下一步」。
11. 針對 Schedule pattern (排程模式)，執行下列其中一項動作：

    - 選擇在特定時間執行的精細排程，例如上午 8:00 PST 在每個月的第一個星期一，然後輸入一個 cron 表達式。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的 Cron 運算式。
    - 選擇以一般費率執行的排程，例如每 10 分鐘執行一次。 ，然後輸入比率表示式。

12. 選擇下一步。
13. 對於 Target types (目標類型)，選擇 AWS 服務。

14. 對於選取目標，請選擇 Batch 工作佇列。然後，設定下列項目：

    - Job queue (任務佇列)：輸入任務佇列的 Amazon Resource Name (ARN) 以排程任務。
    - Job definition (任務定義)：輸入用於任務之任務定義的名稱，及其修訂版或完整 ARN。
    - Job name (任務名稱)：輸入任務的名稱。
    - Array size (陣列大小)：(選擇性) 輸入任務要執行多個副本的陣列大小。如需詳細資訊，請參閱陣列工作。
    - Job attempts (任務嘗試)：(選擇性) 輸入任務失敗時的重試次數。如需詳細資訊，請參閱 自動化工作重試。
15. 對於 Batch 工作佇列目標類型， EventBridge 需要將事件傳送至目標的權限。 EventBridge 可以建立規則執行所需的 IAM 角色。執行以下任意一項：

    - 若要自動建立 IAM 角色，請選擇為此特定資源建立新角色。
    - 若要使用已建立的 IAM 角色，請選擇 [使用現有角色]。
16. (選用) 展開 Additional settings (其他設定)。

    a. 在「設定目標輸入」中，選擇事件中的文字在傳送至目標之前的處理方式。
    b. 對於事件的保留時間上限，請指定未處理事件保留多久的時間間隔。
    c. 對於「重試嘗試」，請輸入重試事件的次數。
    d. 對於無效字母佇列，請選擇處理未處理事件的選項。如有必要，請指定要用作無效字母佇列的 Amazon SQS 佇列。
17. (選用) 選擇新增其他目標，為此規則新增另一個目標。
18. 選擇下一步。
19. (選擇性) 對於標籤，請選擇「新增標籤」以新增規則的資源標籤。如需詳細資訊，請參閱 Amazon EventBridge 標籤。
20. 選擇下一步。
21. 對於「檢閱和建立」，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成時，請選擇 Create rule (建立規則)。

如需有關建立規則的詳細資訊，請參閱 Amazon EventBridge 使用者指南中的建立按排程執行的 Amazon 規 EventBridge 則。

使用事件模式建立規則

下列程序涵蓋如何使用事件模式建立規則。

若要建立在事件符合定義的模式時，將事件傳送至目標的規則

> **Note**
>
> 此程序適用 AWS Batch 於所有 Amazon ECS、Amazon EKS 和 AWS Fargate 工作。

1. 在以下位置打開 Amazon EventBridge 控制台 https://console.aws.amazon.com/events/。
2. 從導覽列中，選取 AWS 區域 要使用的。
3. 在導覽窗格中，選擇規則。
4. 選擇建立規則。
5. 對於「名稱」，請為您的計算環境指定唯一的名稱。名稱最多可包含 64 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。

   > **Note**
   >
   > 在同一個區域和同一個事件匯流排上，規則不能與另一個規則同名。

6. (選擇性) 在說明中，輸入規則的說明。
7. 針對事件匯流排，選擇要與此規則建立關聯的事件匯流排。如果您想要此規則匹配來自您的帳戶的事件，請選取預設值。當您帳戶 AWS 服務 中的某個事件發出時，它始終會進入您帳戶的默認事件總線。
8. (選擇性) 如果您不想立即執行規則，請關閉所選匯流排上的規則。
9. 針對規則類型，選擇具有事件模式的規則。
10. 選擇下一步。
11. 對於事件來源，請選擇AWS 事件或 EventBridge 合作夥伴事件。
12. (選擇性) 對於範例事件：
    a. 針對範例事件類型，選擇AWS 事件。
    b. 對於範例事件，請選擇 Batch Job 狀態變更。
13. 針對建立方法，選取使用模式表單。
14. 對於事件模式：
    a. 在 Event source (事件來源)，選擇 AWS 服務。
    b. 對於 AWS 服務，選擇「Batch」。

c. 對於事件類型，請選擇 Batch Job 狀態變更。

15. 選擇下一步。
16. 對於 Target types (目標類型)，選擇 AWS 服務。
17. 在「選取目標」中，選擇目標類型。例如，選擇 Batch 工作佇列。然後指定下列項目：

    - Job queue (任務佇列)：輸入任務佇列的 Amazon Resource Name (ARN) 以排程任務。
    - Job definition (任務定義)：輸入用於任務之任務定義的名稱，及其修訂版或完整 ARN。
    - Job name (任務名稱)：輸入任務的名稱。
    - Array size (陣列大小)：(選擇性) 輸入任務要執行多個副本的陣列大小。如需詳細資訊，請參閱陣列工作。
    - Job attempts (任務嘗試)：(選擇性) 輸入任務失敗時的重試次數。如需詳細資訊，請參閱 自動化工作重試。
18. 對於 Batch 工作佇列目標類型， EventBridge 需要將事件傳送至目標的權限。 EventBridge 可以建立規則執行所需的 IAM 角色。執行以下任意一項：

    - 若要自動建立 IAM 角色，請選擇為此特定資源建立新角色。
    - 若要使用您之前建立的 IAM 角色，請選擇 Use existing role (使用現有角色)。
19. (選用) 展開 Additional settings (其他設定)。

    a. 在「設定目標輸入」中，選擇處理事件中文字的方式。

    b. 對於事件的保留時間上限，請指定未處理事件保留多久的時間間隔。

    c. 對於「重試嘗試」，請輸入重試事件的次數。

    d. 對於無效字母佇列，請選擇處理未處理事件的選項。如有必要，請指定要用作無效字母佇列的 Amazon SQS 佇列。
20. (選擇性) 選擇「新增其他目標」以新增其他目標。
21. 選擇下一步。
22. (選擇性) 對於標籤，請選擇新增標籤以新增資源標籤。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的 Amazon EventBridge 標籤。
23. 選擇下一步。
24. 對於「檢閱和建立」，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成後，請選擇 [建立規則]。

    如需有關建立規則的詳細資訊，請參閱 Amazon EventBridge 使用者指南中的建立可對事件做出反應的 Amazon EventBridge 規則。

使用 EventBridge 輸入轉換器按排程將事件資訊傳遞至 AWS Batch Target

您可以使用 EventBridge 輸入轉換器將事件資訊傳遞至工作提交 AWS Batch 中。如果您因為其他 AWS 事件資訊而呼叫工作，這可能會特別有用。其中一個範例是將物件上傳到 Amazon S3 儲存貯體。您也可以在容器的命令中使用具有參數替代值的工作定義。 EventBridge 輸入變壓器可以根據事件資料提供參數值。

然後，您會建立一個 AWS Batch 事件目標，從啟動它的事件剖析資訊，並將其轉換為`parameters`物件。工作執行時，觸發器事件中的參數會傳遞至工作容器的命令。

> **Note**
>
> 在這個案例中，所有資 AWS 源 (例如 Amazon S3 儲存貯體、 EventBridge 規則和 CloudTrail 日誌) 都必須位於同一個區域。

建立使用輸入變壓器的 AWS Batch 目標的步驟

1. 在以下位置打開 Amazon EventBridge 控制台 https://console.aws.amazon.com/events/。
2. 從導覽列中，選取 AWS 區域 要使用的。
3. 在導覽窗格中，選擇規則。
4. 選擇建立規則。
5. 對於「名稱」，請為您的計算環境指定唯一的名稱。名稱最多可包含 64 個字元。可以包含大小寫字母、數字、連字號 (-) 和底線 (_)。

   > **Note**
   >
   > 規則的名稱不能 AWS 區域 與相同事件匯流排中的另一個規則相同。

6. (選擇性) 在說明中，輸入規則的說明。
7. 針對事件匯流排，選擇要與此規則建立關聯的事件匯流排。如果您想要此規則匹配來自您的帳戶的事件，請選取預設值。當您帳戶 AWS 服務 中的某個事件發出時，它始終會進入您帳戶的默認事件總線。
8. (選擇性) 如果您不想立即執行規則，請關閉所選匯流排上的規則。
9. 針對規則類型，選擇排程。
10. 選擇「繼續」建立規則或「下一步」。

11. 針對 Schedule pattern (排程模式)，執行下列其中一項動作：

    - 選擇在特定時間執行的精細排程，例如上午 8:00 PST 在每個月的第一個星期一，然後輸入一個 cron 表達式。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的 Cron 運算式。
    - 選擇以一般費率執行的排程，例如每 10 分鐘執行一次。 ，然後輸入比率表示式。

12. 選擇下一步。
13. 對於 Target types (目標類型)，選擇 AWS 服務。
14. 對於選取目標，請選擇 Batch 工作佇列。然後，設定下列項目：

    - Job queue (任務佇列)：輸入任務佇列的 Amazon Resource Name (ARN) 以排程任務。
    - Job definition (任務定義)：輸入用於任務之任務定義的名稱，及其修訂版或完整 ARN。
    - Job name (任務名稱)：輸入任務的名稱。
    - Array size (陣列大小)：(選擇性) 輸入任務要執行多個副本的陣列大小。如需詳細資訊，請參閱陣列工作。
    - Job attempts (任務嘗試)：(選擇性) 輸入任務失敗時的重試次數。如需詳細資訊，請參閱 自動化工作重試。

15. 對於 Batch 工作佇列目標類型， EventBridge 需要將事件傳送至目標的權限。 EventBridge 可以建立規則執行所需的 IAM 角色。執行以下任意一項：

    - 若要自動建立 IAM 角色，請選擇為此特定資源建立新角色。
    - 若要使用已建立的 IAM 角色，請選擇 [使用現有角色]。

16. (選用) 展開 Additional settings (其他設定)。
17. 在 Additional settings (其他設定) 區段中，針對 Configure target input (設定目標輸入)，選擇 Input Transformer (輸入轉換器)。
18. 選擇設定輸入轉換器。
19. (選擇性) 對於範例事件：

    a. 針對範例事件類型，選擇AWS 事件。

    b. 對於範例事件，請選擇 Batch Job 狀態變更。

20. 在 Target input transformer (目標輸入轉換器) 區段中，針對 Input path (輸入路徑)，指定透過觸發事件剖析的值。例如，若要剖析「Batch Job 狀態變更」事件，請使用下列 JSON 格式。

```
{
    "instance": "$.detail.jobId",
    "state": "$.detail.status"
```

```
}
```

21. 在「範本」中，輸入以下內容。

```
{
    "instance": <jobId> ,
    "status": <status>
}
```

22. 選擇確認。
23. 對於事件的保留時間上限，請指定未處理事件保留多久的時間間隔。
24. 對於「重試嘗試」，請輸入重試事件的次數。
25. 對於無效字母佇列，請選擇處理未處理事件的選項。如有必要，請指定要用作無效字母佇列的 Amazon SQS 佇列。
26. (選擇性) 選擇「新增其他目標」以新增其他目標。
27. 選擇下一步。
28. (選擇性) 對於標籤，請選擇新增標籤以新增資源標籤。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的 Amazon EventBridge 標籤。
29. 選擇下一步。
30. 對於「檢閱和建立」，請檢閱組態步驟。如需變更，請選擇 Edit (編輯)。完成後，請選擇 [建立規則]。

教學課程：聆聽 AWS Batch EventBridge

在本教學課程中，您會設定一個簡單AWS Lambda函數，以偵聽AWS Batch工作事件並將它們寫入 CloudWatch 記錄資料流。

必要條件

此教學課程假設您有一個運作中的運算環境，和已準備好要接受任務的任務佇列。如果您沒有可從中擷取事件的執行中計算環境和工作佇列，請按照中的步驟開始使用 AWS Batch建立一個。在本教學課程結束時，您可以選擇性地將工作提交至此工作佇列，以測試您是否已正確設定 Lambda 函數。

步驟 1：建立 Lambda 函數

在此程序中，您可以建立簡單的 Lambda 函數，做為AWS Batch事件串流訊息的目標。

若要建立目標 Lambda 函數

1. 前往 https://console.aws.amazon.com/lambda/ 開啟 AWS Lambda 主控台。
2. 依序選擇 Create function (建立函數)、Author from scratch (從頭開始撰寫)。
3. 針對 函數名稱 ，請輸入 batch-event-stream-handler。
4. 針對執行階段，選擇 Python 3.8。
5. 選擇 建立函式。
6. 在「程式碼原始碼」區段中，編輯範例程式碼以符合下列範例：

```
import json


def lambda_handler(event, _context):
    # _context is not used
    del _context
    if event["source"] != "aws.batch":
        raise ValueError("Function only supports input from events with a source
 type of: aws.batch")

    print(json.dumps(event))
```

這是一個簡單的 Python 3.8 函數，用於打印由發送的事件AWS Batch。如果所有項目都設定正確，則在本教學課程結束時，事件詳細資料會顯示在與此 Lambda 函數相關聯的 CloudWatch 記錄資料流中。

7. 選擇部署。

步驟 2：註冊事件規則

在本節中，您會建立一個 EventBridge 事件規則，以擷取來自AWS Batch資源的工作事件。此規則會擷取來自定義其帳戶AWS Batch內的所有事件。工作訊息本身包含事件來源的相關資訊，包括提交事件來源的工作佇列。您可以使用此資訊以程式設計方式篩選和排序事件。

> **Note**
>
> 如果您使用AWS Management Console建立事件規則，主控台會自動新增 IAM 許可，EventBridge 以呼叫您的 Lambda 函數。不過，如果您要使用建立事件規則AWS CLI，則必

須明確授與權限。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的事件和事件模式。

若要建立 EventBridge 規則

1. 在以下位置打開 Amazon EventBridge 控制台 https://console.aws.amazon.com/events/。
2. 在導覽窗格中，選擇 Rules(規則)。
3. 選擇 Create rule (建立規則)。
4. 輸入規則的名稱和描述。

   在同一個區域和同一個事件匯流排上，規則不能與另一個規則同名。
5. 針對 Event bus (事件匯流排)，選擇要與此規則建立關聯的事件匯流排。如果您想要此規則匹配來自您的帳戶的事件，請選取 AWS default event bus (預設事件匯流排)。當您帳戶中的 AWS 服務發出事件時，一律會前往您帳戶的預設事件匯流排。
6. 針對 Rule type (規則類型) 選擇 Rule with an event pattern (具有事件模式的規則)。
7. 選擇 Next (下一步)。
8. 在 Event source (事件來源) 中，選擇 Other (其他)。
9. 對於事件模式，選取自訂模式 (JSON 編輯器)。
10. 將下列的事件模式貼到文字區域。

```
{
  "source": [
    "aws.batch"
  ]
}
```

   此規則適用於所有AWS Batch群組和每個AWS Batch活動。或者，您可以建立一個更針對性的規則，來篩選掉一些結果。
11. 選擇 Next (下一步)。
12. 在 Target types (目標類型) 欄位中，選擇 AWS service (服務)。
13. 對於選取目標，選擇 Lambda 函數，然後選取您的 Lambda 函數。
14. (選用) 針對 Additional settings (其他設定)，執行下列動作：

    a. 針對 Maximum age of event (事件的最長存留期)，輸入介於一分鐘 (00:01) 到 24 小時 (24:00) 之間的某個值。

b. 針對 Retry attempts (重試嘗試)，輸入介於 0 到 185 之間的某個數。

c. 對於無效字母佇列，請選擇是否使用標準 Amazon SQS 佇列做為無效字母佇列。EventBridge 如果符合此規則的事件未成功傳遞至目標，則會將符合此規則的事件傳送至無效字母佇列。執行以下任意一項：

- 選擇 None (無)，即不使用無效字母佇列。
- 選擇 Select an Amazon SQS queue in the current AWS account to use as the dead-letter queue (選取當前帳戶中的 Amazon SQS 佇列以用作無效字母佇列)，然後從下拉式清單中選取要使用的佇列。
- 選擇 Select an Amazon SQS queue in an other AWS account as a dead-letter queue (選取其他 帳戶中的 Amazon SQS 佇列做為無效字母佇列)，然後輸入要使用的佇列的 ARN。您必須將以資源為基礎的政策附加至佇列，以授與傳送訊息給該佇列的 EventBridge 權限。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的授予無效字母佇列的許可。

15. 選擇 Next (下一步)。
16. (選用) 為規則輸入一或多個標籤。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的 Amazon EventBridge 標籤。
17. 選擇 Next (下一步)。
18. 檢閱規則的詳細資訊，然後選擇 Create rule (建立規則)。

步驟 3：測試組態

您現在可以透過將工作提交至工作佇列來測試 EventBridge 組態。如果一切都設定正確，則會觸發 Lambda 函數，並將事件資料寫入該函數的 CloudWatch 記錄日誌串流。

若要測試組態

1. 請在以下位置開啟AWS Batch主控台。 https://console.aws.amazon.com/batch/
2. 提交新的 AWS Batch 任務。如需詳細資訊，請參閱提交工作。
3. 請在以下位置開啟 CloudWatch 主控台。 https://console.aws.amazon.com/cloudwatch/
4. 在導覽窗格上，選擇 Logs (日誌)，然後選取 Lambda 函數的日誌群組 (例如，/aws/lambda/*my-function*)。
5. 選取日誌串流，以檢視事件資料。

教學：針對失敗的 Job 務事件傳送 Amazon 簡易通知服務警示

在本教學課程中，您會設定僅擷取工作已移至某個FAILED狀態之工作 EventBridge 事件的事件的事件規則。在本教學課程結束時，您也可以選擇性地將工作送至此工作佇列。這是為了測試您是否已正確設定 Amazon SNS 警示。

必要條件

此教學課程假設您有一個運作中的運算環境，和已準備好要接受任務的任務佇列。如果您沒有可從中擷取事件的執行中計算環境和工作佇列，請按照中的步驟開始使用 AWS Batch建立一個。

步驟 1：建立並訂閱 Amazon SNS 主題

在此教學課程中，您會設定 Amazon SNS 主題，做為新事件規則的事件目標。

建立 Amazon SNS 主題

1. 在 https://console.aws.amazon.com/sns/v3/home 開啟 Amazon SNS 主控台。
2. 選擇 Topics (主題)、Create topic (建立主題)。
3. 針對類型，選擇標準。
4. 在「名稱」中，輸入**JobFailedAlert**並選擇「建立主題」。
5. 在JobFailedAlert畫面上，選擇 [建立訂閱]。
6. 對於通訊協定，選擇電子郵件。
7. 對於 Endpoint (端點)，輸入您目前能存取的電子郵件地址，並選擇 Create subscription (建立訂閱)。
8. 檢查您的電子郵件帳戶，並等待收到訂閱確認的電子郵件訊息。您收到訊息時，請選擇 Confirm subscription (確認訂閱)。

步驟 2：註冊事件規則

接著，註冊事件規則，使其只擷取任務失敗的事件。

若要註冊您的 EventBridge 規則

1. 在以下位置打開 Amazon EventBridge 控制台 https://console.aws.amazon.com/events/。
2. 在導覽窗格中，選擇規則。

3. 選擇建立規則。
4. 輸入規則的名稱和描述。

   在同一個區域和同一個事件匯流排上，規則不能與另一個規則同名。
5. 針對事件匯流排，選擇要與此規則建立關聯的事件匯流排。如果您想要此規則匹配來自您的帳戶的事件，請選取 AWS 預設事件匯流排。當您帳戶中的某個 AWS 服務發出活動時，它始終會進入您帳戶的預設事件匯流排。
6. 針對規則類型，選擇具有事件模式的規則。
7. 選擇下一步。
8. 在事件來源中，選擇其他。
9. 對於事件模式，選取自訂模式 (JSON 編輯器)。
10. 將下列的事件模式貼到文字區域。

```
{
  "detail-type": [
    "Batch Job State Change"
  ],
  "source": [
    "aws.batch"
  ],
  "detail": {
    "status": [
      "FAILED"
    ]
  }
}
```

   此程式碼會 EventBridge 定義與工作狀態為的任何事件相符的規則`FAILED`。如需有關事件模式的詳細資訊，請參閱 Amazon EventBridge 使用者指南中的事件和事件模式。
11. 選擇下一步。
12. 在目標類型欄位中，選擇 AWS 服務。
13. 在 [選取目標] 中選擇 SNS 主題，然後選擇 [主題] 做為 [主題] JobFailedAlert。
14. (選用) 針對其他設定，請執行下列動作：

    a. 針對 Maximum age of event (事件的最長存留期)，輸入介於一分鐘 (00:01) 到 24 小時 (24:00) 之間的某個值。

    b. 針對重試嘗試，輸入介於 0 到 185 之間的某個數。

c. 對於無效字母佇列，請選擇是否使用標準 Amazon SQS 佇列做為無效字母佇列。EventBridge 如果符合此規則的事件未成功傳遞至目標，則會將符合此規則的事件傳送至無效字母佇列。執行以下任意一項：

- 選擇無，即不使用無效字母佇列。
- 選擇選取目前 AWS 帳戶中的 Amazon SQS 佇列作為無效字母佇列，然後從下拉式清單中選取要使用的佇列。
- 選擇選取其他 AWS 帳戶中的 Amazon SQS 佇列做為無效字母佇列，然後輸入要使用的佇列 ARN。您必須將以資源為基礎的政策附加至佇列，以授與傳送訊息給該佇列的 EventBridge 權限。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的授予無效字母佇列的許可。

15. 選擇下一步。
16. (選用) 為規則輸入一或多個標籤。如需詳細資訊，請參閱 Amazon EventBridge 使用者指南中的 Amazon EventBridge 標籤。
17. 選擇下一步。
18. 檢閱規則的詳細資訊，然後選擇建立規則。

步驟 3：測試您的規則

為了測試您的規則，提交在使用非零結束代碼啟動後不久結束的任務。如果您的事件規則設定正確，您應該會在幾分鐘內收到包含事件文字的電子郵件訊息。

測試規則

1. 開啟主 AWS Batch 控台，網址為 https://console.aws.amazon.com/batch/。
2. 提交新 AWS Batch 工作。如需詳細資訊，請參閱 提交工作。針對任務的命令，將結束容器的此命令換成結束代碼 1。

```
/bin/sh, -c, 'exit 1'
```

3. 檢查您的電子郵件，確認您收到失敗的作業通知的電子郵件警示。

替代規則：已封鎖 Batch Job 佇列

若要建立監視「已封鎖 Batch Job 佇列」的事件規則，請重複本教學課程中的步驟，並進行下列變更：

1. 在步驟 1 中，用*BlockedJobQueue*作主題名稱。

2. 在步驟 2 中，在 JSON 編輯器中使用下列模式：

```
{
   "detail-type": [
     "Batch Job Queue Blocked"
   ],
   "source": [
     "aws.batch"
   ]
}
```

搭配使用 CloudWatch 記錄 AWS Batch

您可以在 EC2 資源上設定任 AWS Batch 務，將詳細的日誌資訊和指標傳送到 CloudWatch 日誌。這樣做，您可以在一個方便的位置查看作業中的不同日誌。如需有關 CloudWatch 日誌的詳細資訊，請參閱什麼是 Amazon CloudWatch 日誌？ 在 Amazon 用 CloudWatch 戶指南。

> **Note**
>
> 根據預設， AWS Fargate 容器的 CloudWatch 記錄處於開啟狀態。

若要開啟和自訂記錄 CloudWatch 記錄，請檢閱下列一次性設定工作：

- 對於以 EC2 資源為基礎的 AWS Batch 運算環境，請將 IAM 政策新增至`ecsInstanceRole`角色。如需詳細資訊，請參閱 the section called “新增 CloudWatch 日誌 IAM 政策”。
- 建立包含詳細 CloudWatch 監控的 Amazon EC2 啟動範本，然後在建立 AWS Batch 運算環境時指定範本。您也可以在現有映像上安裝 CloudWatch代理程式，然後在 AWS Batch 首次執行精靈中指定映像。
- (選擇性) 設定 awslog 驅動程式。您可以新增參數，以變更 EC2 和 Fargate 資源上的預設行為。如需詳細資訊，請參閱 the section called “使用 awslogs 日誌驅動程式”。

新增 CloudWatch 日誌 IAM 政策

您必須先建立使用記錄 API 的 IAM 政策，才能將 CloudWatch 日誌資料和詳細指標傳送至 CloudWatch 記錄檔。建立 IAM 政策後，請將其附加到`ecsInstanceRole`角色。

> **Note**
>
> 如果原`ECS-CloudWatchLogs`則未附加至`ecsInstanceRole`角色，基本量度仍可傳送至 CloudWatch 記錄檔。但是，基本指標不包括日誌數據或詳細指標，例如可用磁盤空間。

AWS Batch 運算環境使用 Amazon EC2 資源。使用 AWS Batch 首次執行精靈建立計算環境時，AWS Batch 會建立`ecsInstanceRole`角色並使用該角色設定環境。

如果您未使用首次執行精靈，則可以在 AWS Command Line Interface 或 AWS Batch API 中建立計算環境時指定`ecsInstanceRole`角色。如需詳細資訊，請參閱AWS CLI 命令參考或 AWS Batch API 參考。

若要建立 **ECS-CloudWatchLogs** IAM 政策

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇政策。
3. 選擇 Create policy (建立政策)。
4. 選擇 JSON，然後輸入下列政策：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": [
                "arn:aws:logs:*:*:*"
            ]
        }
    ]
}
```

5. 選擇下一步：標籤。
6. (選擇性) 在 [新增標記] 中，選擇 [新增標記] 將標籤新增至原則。
7. 選擇下一步：檢閱。
8. 在 [檢閱原則] 頁面上，針對 [名稱] 輸入**ECS-CloudWatchLogs**，然後輸入選擇性的說明。
9. 選擇建立政策。

將 **ECS-CloudWatchLogs** 政策連接至 **ecsInstanceRole**

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇角色。

3. 選擇 `ecsInstanceRole`。如果角色不存在，請按照中的程序Amazon ECS 執行個體角色建立角色。
4. 選擇「新增權限」，然後選擇「附加策略」
5. 選擇 ECS-CloudWatch 記錄原則，然後選擇 [附加原則]。

安裝和設定 CloudWatch 代理程式

您可以建立包含 CloudWatch 監控的 Amazon EC2 啟動範本。如需詳細資訊，請參閱從啟動範本啟動執行個體和 Amazon EC2 使用者指南中的進階詳細資訊。

您也可以在現有的 Amazon EC2 AMI 上安裝 CloudWatch 代理程式，然後在 AWS Batch 首次執行精靈中指定映像。如需詳細資訊，請參閱安裝 CloudWatch 代理程式和入門 AWS Batch。

> **Note**
>
> AWS Fargate 資源不支援啟動範本。

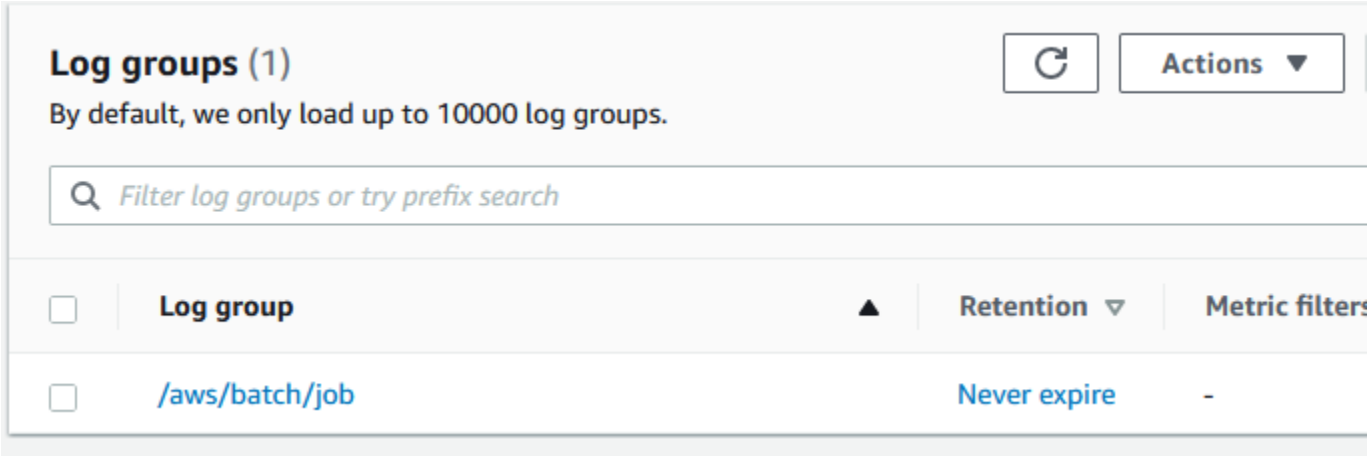
檢視 CloudWatch 記錄

您可以在中檢視和搜尋 CloudWatch 記錄檔記錄 AWS Management Console。

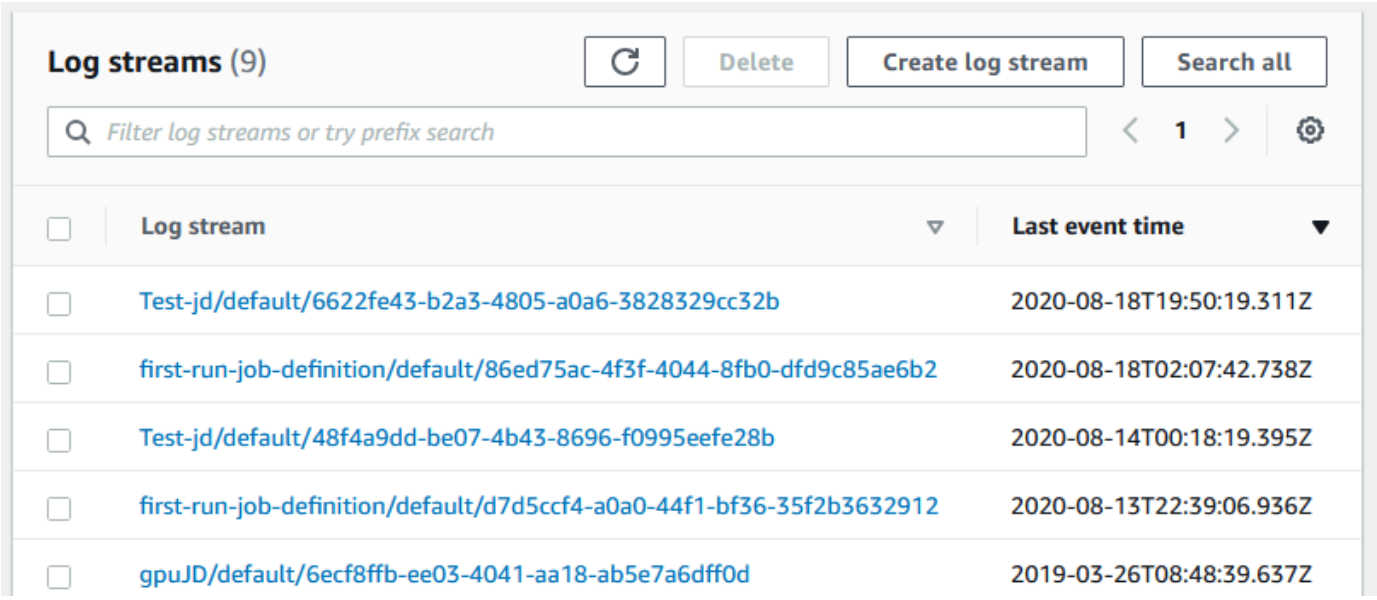
> **Note**
>
> 資料可能需要幾分鐘的時間才會顯示在 CloudWatch 記錄中。

若要檢視您的 CloudWatch 記錄資料

1. 請在以下位置開啟 CloudWatch 主控台。 https://console.aws.amazon.com/cloudwatch/
2. 在左側導覽窗格中，選擇「記錄檔」，然後選擇「記錄群組」。

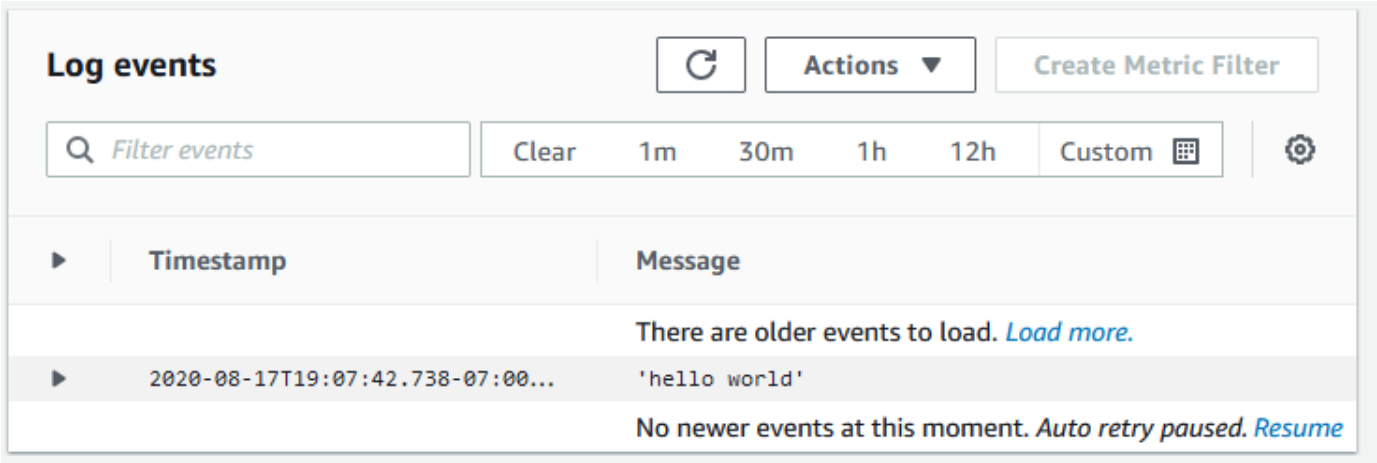


3. 選擇要檢視的日誌群組。



4. 選擇要檢視的日誌串流。依預設，串流會以任務名稱的前 200 個字元和 Amazon ECS 任務 ID 識別。

> **Tip**
>
> 若要下載記錄串流資料，請選擇 [動作]。


Log events
Actions
Create Metric Filter
Filter events
Clear
1m
30m
1h
12h
Custom
Timestamp
Message
There are older events to load. Load more.
2020-08-17T19:07:42.738-07:00...
'hello world'
No newer events at this moment. Auto retry paused. Resume

使用 CloudWatch 日誌監控 AWS Batch Amazon EKS 任務

您可以使用 Amazon CloudWatch Logs 在一個位置監控、存放和檢視所有日誌檔。使用 CloudWatch 記錄檔，您可以搜尋、篩選和分析來自多個來源的記錄資料。

您可以下載包含外掛程式的Fluent Bit影像，以便在 CloudWatch 日誌中監控 AWS Batch Amazon EKS 任務。AWS Fluent Bit是一個開源日誌處理器和轉發器，既 Docker 又Kubernetes兼容。我們建議您使用Fluent Bit日誌路由器，因為它的資源密集比Fluentd. 如需詳細資訊，請參閱使AWS用流利位元影像。

必要條件

將`CloudWatchAgentServerPolicy`原則附加至 Worker 節點的AWS Identity and Access Management原則。如需詳細資訊，請參閱驗證必要條件。

安AWS裝流利位

如需有關如何安裝AWSFluent Bit和建立 CloudWatch 群組的指示，請參閱使用 CloudWatch代理程式和設定Fluent Bit或快速入門Fluent Bit。

> **Tip**
>
> 請記住，AWS Batch節點上Fluent Bit使用了 .5 CPU 和 100 MB 的內存。這會減少工AWS Batch作的總可用容量。當您調整工作大小時，請考慮這一點。

開啟AWS Batch節點的「流利位元」

若要確保記Fluent Bit錄在受AWS Batch管節點上 DaemonSet 執行，請修改Fluent Bit DaemonSet 容許值：

```
tolerations:
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
```

AWS Batch CloudWatch 容器洞察

CloudWatch Container Insights 會從您的AWS Batch運算環境和任務中收集、彙總和摘要指標和記錄。這些指標包括 CPU、記憶體、磁碟和網路使用率。您可以將這些指標新增至 CloudWatch 儀表板。

營運資料的收集形式為「效能日誌事件」。這些項目使用的是結構化的 JSON 結構描述，可擷取高基數資料並大量地進行存放。根據此資料，在運算環境和工作層級 CloudWatch 建立更高層級的彙總指標作為 CloudWatch 指標。如需詳細資訊，請參閱 Amazon CloudWatch 使用者指南中的適用於 Amazon ECS 的容器洞見結構化日誌。

> **Important**
>
> CloudWatch 容器深入解析以自訂指標的方式收費 CloudWatch。如需詳細資訊，請參閱 Amazon CloudWatch 活動定價

開啟容器深入解析

您可以針對AWS Batch運算環境開啟容器深入解析。

1. 開啟 AWS Batch主控台。
2. 選擇運算環境。
3. 選擇您想要的運算環境。
4. 對於容器深入解析，請針對運算開啟容器深入解析 環境。

> **Tip**
>
> 您可以選取預設間隔來彙總指標或建立自訂 間隔。

依預設，會顯示下列測量結果。如需 Amazon ECS 容器洞察指標的完整清單，請參閱 Amazon CloudWatch 使用者指南中的 Amazon ECS 容器洞察指標。

- **`JobCount`**— 在計算環境中執行的工作數目。
- **`ContainerInstanceCount`**— 執行 Amazon ECS 代理程式並在運算環境中註冊的 Amazon 彈性運算雲端執行個體數量。

- **MemoryReserved**— 由計算環境工作保留的記憶體。只有在其工作定義中已定義記憶體保留區的工作，才會收集此測量結果。
- **MemoryUtilized**— 計算環境工作正在使用的記憶體。只有在其工作定義中已定義記憶體保留區的工作，才會收集此測量結果。
- **CpuReserved**— 由計算環境工作保留的 CPU 單位。只有在其工作定義中已定義 CPU 保留區的工作，才會收集此測量結果。
- **CpuUtilized**— 計算環境中工作所使用的 CPU 單位。只有在其工作定義中已定義 CPU 保留區的工作，才會收集此測量結果。
- **NetworkRxBytes**-接收的位元組數。此指標僅適用於使用awsvpc或橋接網路模式之作業中的容器。
- **NetworkTxBytes**— 傳輸的位元組數。此指標僅適用於使用awsvpc或橋接網路模式之作業中的容器。
- **StorageReadBytes**— 從儲存區讀取的位元組數。
- **StorageWriteBytes**— 寫入儲存區的位元組數目。

使用 AWS CloudTrail 記錄 AWS Batch API 呼叫

AWS Batch與 (提供中的使用者AWS CloudTrail、角色或服務所採取的動作記錄) 的AWS服務整合AWS Batch。 CloudTrail 擷取AWS Batch作為事件的所有 API 呼叫。擷取的呼叫包括從 AWS Batch 主控台進行的呼叫，以及針對 AWS Batch API 操作的程式碼呼叫。如果您建立追蹤，您可以啟用持續交付 CloudTrail 事件到 Amazon S3 儲存貯體，包括AWS Batch. 如果您未設定追蹤，您仍然可以在 [事件歷程記錄] 中檢視 CloudTrail 主控台中最近的事件。使用收集的資訊 CloudTrail，您可以判斷提出的要求 AWS Batch、提出要求的 IP 位址、提出要求的人員、提出要求的時間，以及其他詳細資訊。

若要進一步了解 CloudTrail，請參閱使AWS CloudTrail用者指南。

AWS Batch 中的 資訊 CloudTrail

CloudTrail 在您創建AWS帳戶時，您的帳戶已啟用。當活動發生在中時AWS Batch，該活動會與事件歷史記錄中的其他AWS服務 CloudTrail 事件一起記錄在事件中。您可以檢視、搜尋和下載 AWS 帳戶的最新事件。如需詳細資訊，請參閱檢視具有事 CloudTrail 件記錄的事件。

如需您 AWS 帳戶中正在進行事件的記錄 (包含 AWS Batch 的事件)，請建立線索。追蹤可 CloudTrail 將日誌檔交付到 Amazon S3 儲存貯體。根據預設，當您在主控台建立線索時，線索會套用到所有 AWS 區域。該追蹤會記錄來自 AWS 分割區中所有區域的事件，並將日誌檔案交付到您指定的 Amazon S3 儲存貯體。此外，您還可以設定其他AWS服務，以進一步分析 CloudTrail 記錄中收集的事件資料並採取行動。如需詳細資訊，請參閱下列內容：

- 建立追蹤的概觀
- CloudTrail 支援的服務與整合
- 設定的 Amazon SNS 通知 CloudTrail
- 從多個區域接收 CloudTrail 記錄檔並從多個帳戶接收 CloudTrail 記錄檔

所有AWS Batch操作都由記錄 CloudTrail 並記錄在 https://docs.aws.amazon.com/batch/latest/APIReference/ 中。例如，對 `SubmitJob`、`ListJobs` 及 `DescribeJobs` 區段的呼叫，都會在 CloudTrail 日誌檔案中產生項目。

每一筆事件或日誌項目都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 該請求是否使用根或 IAM 使用者憑證提出。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。

- 該請求是否由另一項 AWS 服務提出。

如需詳細資訊，請參閱 CloudTrail 使用者身分元素。

了解 AWS Batch 日誌檔項目

追蹤是一種組態，可讓事件以日誌檔的形式傳遞到您指定的 Amazon S3 儲存貯體。 CloudTrail 記錄檔包含一或多個記錄項目。事件代表來自任何來源的單一請求，並包含有關請求的動作、動作的日期和時間、請求參數等資訊。 CloudTrail 日誌檔不是公有 API 呼叫的排序堆疊追蹤，因此不會以任何特定順序顯示。

下列範例顯示示範CreateComputeEnvironment動作的 CloudTrail 記錄項目。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE:admin",
    "arn": "arn:aws:sts::012345678910:assumed-role/Admin/admin",
    "accountId": "012345678910",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2017-12-20T00:48:46Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::012345678910:role/Admin",
        "accountId": "012345678910",
        "userName": "Admin"
      }
    }
  },
  "eventTime": "2017-12-20T00:48:46Z",
  "eventSource": "batch.amazonaws.com",
  "eventName": "CreateComputeEnvironment",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.1",
  "userAgent": "aws-cli/1.11.167 Python/2.7.10 Darwin/16.7.0 botocore/1.7.25",
  "requestParameters": {
```

```
    "computeResources": {
      "subnets": [
        "subnet-5eda8e04"
      ],
      "tags": {
        "testBatchTags": "CLI testing CE"
      },
      "desiredvCpus": 0,
      "minvCpus": 0,
      "instanceTypes": [
        "optimal"
      ],
      "securityGroupIds": [
        "sg-aba9e8db"
      ],
      "instanceRole": "ecsInstanceRole",
      "maxvCpus": 128,
      "type": "EC2"
    },
    "state": "ENABLED",
    "type": "MANAGED",
    "computeEnvironmentName": "Test"
  },
  "responseElements": {
    "computeEnvironmentName": "Test",
    "computeEnvironmentArn": "arn:aws:batch:us-east-1:012345678910:compute-environment/
Test"
  },
  "requestID": "890b8639-e51f-11e7-b038-EXAMPLE",
  "eventID": "874f89fa-70fc-4798-bc00-EXAMPLE",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "012345678910"
}
```

建立虛擬私有雲

運算環境中的運算資源需要外部網路存取才能AWS Batch與 Amazon ECS 服務端點通訊。不過，您可能有要在私有子網路中執行的工作。若要靈活地在公用或私有子網路中執行作業，請建立同時具有公用和私有子網路的 VPC。

您可以使用 Amazon Virtual Private Cloud (Amazon VPC) 將AWS資源啟動到您定義的虛擬網路中。本主題提供 Amazon VPC 精靈的連結，以及可選取的選項清單。

建立 VPC

如需如何建立 Amazon VPC 的詳細資訊，請參閱 Amazon VPC 使用者指南中的僅建立 VPC，並使用下表決定要選取的選項。

選項	值
要建立的資源	僅 VPC
名稱	可以選擇為 VPC 提供名稱。
IPv4 CIDR 區塊	IPv4 CIDR 手動輸入 CIDR 區塊大小必須為介於 /16 和 /28 之間的大小。
IPv6 CIDR 區塊	無 IPv6 CIDR 區塊
租用	預設

如需有關 Amazon VPC 的詳細資訊，請參閱《Amazon VPC 使用者指南》中的什麼是 Amazon VPC？。

後續步驟

建立 VPC 之後，請考慮下列步驟：

- 如果您的公有和私有資源需要入站網路存取，則為其建立安全群組。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的使用安全群組。
- 建立 AWS Batch 受管運算環境，將運算資源啟動到您的新 VPC。如需詳細資訊，請參閱建立運算環境。如果您在AWS Batch主控台中使用運算環境建立精靈，則可以指定剛建立的 VPC，以及要啟動執行個體的公用或私有子網路。
- 建立對應至新計算環境的AWS Batch工作佇列。如需詳細資訊，請參閱建立工作佇列。
- 建立任務定義來執行您的任務。如需詳細資訊，請參閱建立單一節點工作定義 。
- 將任務和任務定義提交到新的任務佇列。這項工作落在您使用新 VPC 和子網路建立的運算環境中。如需詳細資訊，請參閱 提交工作。

中的安全性 AWS Batch

雲安全 AWS 是最高的優先級。身為 AWS 客戶，您可以從資料中心和網路架構中獲益，這些架構是為了滿足對安全性最敏感的組織的需求而建置的。

安全是 AWS 與您之間共同的責任。共同責任模型將其描述為雲端「的」安全性和雲端「中」的安全性：

- 雲端的安全性 — AWS 負責保護在 AWS 雲端中執行 AWS 服務的基礎架構。 AWS 還為您提供可以安全使用的服務。若要深入瞭解適用於的規範遵循計劃 AWS Batch，請參閱合規計劃的AWS 服務範圍範圍)。
- 雲端內部的安全：您的責任取決於所使用的 AWS 服務。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規。

本文件可協助您瞭解如何在使用時套用共同責任模型 AWS Batch。下列主題說明如何設定 AWS Batch 以符合安全性與合規性目標。您也會學到如何使用其他可協助您監控和保護 AWS Batch 資源的 AWS 服務。

主題

- 的 Identity and Access Management AWS Batch
- 使 AWS Batch 用介面端點存取
- 符合性驗證 AWS Batch
- 基礎結構安全 AWS Batch

的 Identity and Access Management AWS Batch

AWS Identity and Access Management (IAM) 可協助管理員安全地控制 AWS 資源存取權。 AWS 服務 IAM 管理員控制哪些人可以通過身份驗證 (登入) 和授權 (具有權限) 來使用 AWS Batch 資源。您可以使用 IAM AWS 服務 ，無需額外付費。

主題

- 物件
- 使用身分驗證

- 使用政策管理存取權
- 如何與 IAM AWS Batch 搭配使用
- AWS Batch 執行 IAM 角色
- 以身分識別為基礎的原則範例 AWS Batch
- 預防跨服務混淆代理人
- 疑難排解 AWS Batch 身分和存取
- 使用服務連結角色 AWS Batch
- AWS 受管理的政策 AWS Batch

物件

您使用 AWS Identity and Access Management (IAM) 的方式會有所不同，具體取決於您在進行的工作 AWS Batch。

服務使用者 — 如果您使用 AWS Batch 服務執行工作，則管理員會為您提供所需的認證和權限。當您使用更多 AWS Batch 功能來完成工作時，您可能需要其他權限。了解存取的管理方式可協助您向管理員請求正確的許可。若您無法存取 AWS Batch中的某項功能，請參閱 疑難排解 AWS Batch 身分和存取。

服務管理員 — 如果您負責公司的 AWS Batch 資源，您可能擁有完整的存取權 AWS Batch。決定您的服務使用者應該存取哪些 AWS Batch 功能和資源是您的工作。接著，您必須將請求提交給您的 IAM 管理員，來變更您服務使用者的許可。檢閱此頁面上的資訊，了解 IAM 的基本概念。若要進一步瞭解貴公司如何搭配使用 IAM AWS Batch，請參閱如何與 IAM AWS Batch 搭配使用。

IAM 管理員：如果您是 IAM 管理員，建議您掌握如何撰寫政策以管理 AWS Batch存取權的詳細資訊。若要檢視可在 IAM 中使用的 AWS Batch 基於身分的政策範例，請參閱。以身分識別為基礎的原則範例 AWS Batch

使用身分驗證

驗證是您 AWS 使用身分認證登入的方式。您必須以 IAM 使用者身分或假設 IAM 角色進行驗證 (登入 AWS)。 AWS 帳戶根使用者

您可以使用透過 AWS 身分識別來源提供的認證，以聯合身分識別身分登入。 AWS IAM Identity Center (IAM 身分中心) 使用者、貴公司的單一登入身分驗證，以及您的 Google 或 Facebook 登入資料都是聯合身分識別的範例。您以聯合身分登入時，您的管理員先前已設定使用 IAM 角色的聯合身分。當您使 AWS 用同盟存取時，您會間接擔任角色。

根據您的使用者類型，您可以登入 AWS Management Console 或 AWS 存取入口網站。如需有關登入的詳細資訊 AWS，請參閱《AWS 登入 使用指南》 AWS 帳戶中的如何登入您的。

如果您 AWS 以程式設計方式存取，請 AWS 提供軟體開發套件 (SDK) 和命令列介面 (CLI)，以使用您的認證以加密方式簽署要求。如果您不使用 AWS 工具，則必須自行簽署要求。如需使用建議的方法自行簽署請求的詳細資訊，請參閱 IAM 使用者指南中的簽署 AWS API 請求。

無論您使用何種身分驗證方法，您可能都需要提供額外的安全性資訊。例如， AWS 建議您使用多重要素驗證 (MFA) 來增加帳戶的安全性。如需更多資訊，請參閱 AWS IAM Identity Center 使用者指南中的多重要素驗證和 IAM 使用者指南中的在 AWS中使用多重要素驗證 (MFA)。

AWS 帳戶 根使用者

當您建立時 AWS 帳戶，您會從一個登入身分開始，該身分可完整存取該帳戶中的所有資源 AWS 服務和資源。此身分稱為 AWS 帳戶 root 使用者，可透過使用您用來建立帳戶的電子郵件地址和密碼登入來存取。強烈建議您不要以根使用者處理日常任務。保護您的根使用者憑證，並將其用來執行只能由根使用者執行的任務。如需這些任務的完整清單，了解需以根使用者登入的任務，請參閱 IAM 使用者指南中的需要根使用者憑證的任務。

聯合身分

最佳作法是要求人類使用者 (包括需要系統管理員存取權的使用者) 使用與身分識別提供者的同盟，才能使用臨時登入資料進行存取 AWS 服務 。

聯合身分識別是來自企業使用者目錄的使用者、Web 身分識別提供者、Identi ty Center 目錄，或使用透過身分識別來源提供的認證進行存取 AWS 服務 的任何使用者。 AWS Directory Service同盟身分存取時 AWS 帳戶，他們會假設角色，而角色則提供臨時認證。

對於集中式存取權管理，我們建議您使用 AWS IAM Identity Center。您可以在 IAM Identity Center 中建立使用者和群組，也可以連線並同步到自己身分識別來源中的一組使用者和群組，以便在所有應用程式 AWS 帳戶 和應用程式中使用。如需 IAM Identity Center 的相關資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

IAM 使用者和群組

IAM 使用者是您內部的身分，具 AWS 帳戶 有單一人員或應用程式的特定許可。建議您盡可能依賴暫時憑證，而不是擁有建立長期憑證 (例如密碼和存取金鑰) 的 IAM 使用者。但是如果特定使用案例需要擁有長期憑證的 IAM 使用者，建議您輪換存取金鑰。如需更多資訊，請參閱 IAM 使用者指南中的為需要長期憑證的使用案例定期輪換存取金鑰。

IAM 群組是一種指定 IAM 使用者集合的身分。您無法以群組身分簽署。您可以使用群組來一次為多名使用者指定許可。群組可讓管理大量使用者許可的程序變得更為容易。例如，您可以擁有一個名為 IAMAdmins 的群組，並給予該群組管理 IAM 資源的許可。

使用者與角色不同。使用者只會與單一人員或應用程式建立關聯，但角色的目的是在由任何需要它的人員取得。使用者擁有永久的長期憑證，但角色僅提供暫時憑證。如需進一步了解，請參閱 IAM 使用者指南中的建立 IAM 使用者 (而非角色) 的時機。

IAM 角色

IAM 角色是您 AWS 帳戶 內部具有特定許可的身分。它類似 IAM 使用者，但不與特定的人員相關聯。您可以切換角色，在中暫時擔任 IAM 角色。 AWS Management Console 您可以透過呼叫 AWS CLI 或 AWS API 作業或使用自訂 URL 來擔任角色。如需使用角色的方法更多相關資訊，請參閱 IAM 使用者指南中的使用 IAM 角色。

使用暫時憑證的 IAM 角色在下列情況中非常有用：

- 聯合身分使用者存取 – 若要向聯合身分指派許可，請建立角色，並為角色定義許可。當聯合身分進行身分驗證時，該身分會與角色建立關聯，並獲授予由角色定義的許可。如需有關聯合角色的相關資訊，請參閱 IAM 使用者指南中的為第三方身分提供者建立角色。如果您使用 IAM Identity Center，則需要設定許可集。為控制身分驗證後可以存取的內容，IAM Identity Center 將許可集與 IAM 中的角色相關聯。如需有關許可集的資訊，請參閱 AWS IAM Identity Center 使用者指南中的許可集。
- 暫時 IAM 使用者許可 – IAM 使用者或角色可以擔任 IAM 角色來暫時針對特定任務採用不同的許可。
- 跨帳戶存取權 – 您可以使用 IAM 角色，允許不同帳戶中的某人 (信任的委託人) 存取您帳戶中的資源。角色是授予跨帳戶存取權的主要方式。但是，對於某些策略 AWS 服務，您可以將策略直接附加到資源（而不是使用角色作為代理）。若要了解跨帳戶存取權角色和資源型政策間的差異，請參閱 IAM 使用者指南中的 IAM 角色與資源類型政策的差異。
- 跨服務訪問 — 有些 AWS 服務 使用其他 AWS 服務功能。例如，當您在服務中進行呼叫時，該服務通常會在 Amazon EC2 中執行應用程式或將物件儲存在 Amazon Simple Storage Service (Amazon S3) 中。服務可能會使用呼叫主體的許可、使用服務角色或使用服務連結角色來執行此作業。
  - 轉寄存取工作階段 (FAS) — 當您使用 IAM 使用者或角色在中執行動作時 AWS，您會被視為主體。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS 會使用主體呼叫的權限 AWS 服務，並結合要求 AWS 服務 向下游服務發出要求。只有當服務收到需要與其 AWS 服務 他資源互動才能完成的請求時，才會發出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱《轉發存取工作階段》。
  - 服務角色 – 服務角色是服務擔任的 IAM 角色，可代表您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需更多資訊，請參閱 IAM 使用者指南中的建立角色以委派許可給 AWS 服務。

- 服務連結角色 — 服務連結角色是連結至. AWS 服務服務可以擔任代表您執行動作的角色。服務連結角色會顯示在您的中， AWS 帳戶 且屬於服務所有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。
- 在 Amazon EC2 上執行的應用程式 — 您可以使用 IAM 角色來管理在 EC2 執行個體上執行的應用程式以及發出 AWS CLI 或 AWS API 請求的臨時登入資料。這是在 EC2 執行個體內儲存存取金鑰的較好方式。若要將 AWS 角色指派給 EC2 執行個體並提供給其所有應用程式，請建立連接至執行個體的執行個體設定檔。執行個體設定檔包含該角色，並且可讓 EC2 執行個體上執行的程式取得暫時憑證。如需更多資訊，請參閱 IAM 使用者指南中的利用 IAM 角色來授予許可給 Amazon EC2 執行個體上執行的應用程式。

若要了解是否要使用 IAM 角色或 IAM 使用者，請參閱 IAM 使用者指南中的建立 IAM 角色 (而非使用者) 的時機。

使用政策管理存取權

您可以透 AWS 過建立原則並將其附加至 AWS 身分識別或資源來控制中的存取。原則是一個物件 AWS ，當與身分識別或資源相關聯時，會定義其權限。 AWS 當主參與者 (使用者、root 使用者或角色工作階段) 提出要求時，評估這些原則。政策中的許可決定是否允許或拒絕請求。大多數原則會 AWS 以 JSON 文件的形式儲存在中。如需 JSON 政策文件結構和內容的更多相關資訊，請參閱 IAM 使用者指南中的 JSON 政策概觀。

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

預設情況下，使用者和角色沒有許可。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

IAM 政策定義該動作的許可，無論您使用何種方法來執行操作。例如，假設您有一個允許 `iam:GetRole` 動作的政策。具有該原則的使用者可以從 AWS Management Console AWS CLI、或 AWS API 取得角色資訊。

身分型政策

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立身分類型政策，請參閱 IAM 使用者指南中的建立 IAM 政策。

身分型政策可進一步分類成內嵌政策或受管政策。內嵌政策會直接內嵌到單一使用者、群組或角色。受管理的策略是獨立策略，您可以將其附加到您的 AWS 帳戶. 受管政策包括 AWS 受管政策和客戶管理

的策略。若要了解如何在受管政策及內嵌政策間選擇，請參閱 IAM 使用者指南中的在受管政策和內嵌政策間選擇。

資源型政策

資源型政策是連接到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。主參與者可以包括帳戶、使用者、角色、同盟使用者或。 AWS 服務

資源型政策是位於該服務中的內嵌政策。您無法在以資源為基礎的政策中使用 IAM 的 AWS 受管政策。

存取控制清單 (ACL)

存取控制清單 (ACL) 可控制哪些委託人 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

Amazon S3 和 Amazon VPC 是支援 ACL 的服務範例。 AWS WAF若要進一步了解 ACL，請參閱 Amazon Simple Storage Service 開發人員指南中的存取控制清單 (ACL) 概觀。

其他政策類型

AWS 支援其他較不常見的原則類型。這些政策類型可設定較常見政策類型授予您的最大許可。

- 許可界限 – 許可範圍是一種進階功能，可供您設定身分型政策能授予 IAM 實體 (IAM 使用者或角色) 的最大許可。您可以為實體設定許可界限。所產生的許可會是實體的身分型政策和其許可界限的交集。會在 `Principal` 欄位中指定使用者或角色的資源型政策則不會受到許可界限限制。所有這類政策中的明確拒絕都會覆寫該允許。如需許可範圍的更多相關資訊，請參閱 IAM 使用者指南中的 IAM 實體許可範圍。
- 服務控制策略（SCP）— SCP 是 JSON 策略，用於指定中組織或組織單位（OU）的最大權限。AWS Organizations AWS Organizations 是一種用於分組和集中管理您企業擁 AWS 帳戶 有的多個服務。若您啟用組織中的所有功能，您可以將服務控制政策 (SCP) 套用到任何或所有帳戶。SCP 限制成員帳戶中實體的權限，包括每個 AWS 帳戶根使用者帳戶。如需組織和 SCP 的更多相關資訊，請參閱 AWS Organizations 使用者指南中的 SCP 運作方式。
- 工作階段政策 – 工作階段政策是一種進階政策，您可以在透過編寫程式的方式建立角色或聯合使用者的暫時工作階段時，作為參數傳遞。所產生工作階段的許可會是使用者或角色的身分型政策和工作

階段政策的交集。許可也可以來自資源型政策。所有這類政策中的明確拒絕都會覆寫該允許。如需更多資訊，請參閱 IAM 使用者指南中的工作階段政策。

多種政策類型

將多種政策類型套用到請求時，其結果形成的許可會更為複雜、更加難以理解。要了解如何在涉及多個政策類型時 AWS 確定是否允許請求，請參閱《IAM 使用者指南》中的政策評估邏輯。

如何與 IAM AWS Batch 搭配使用

在您使用 IAM 管理存取權限之前 AWS Batch，請先了解哪些 IAM 功能可搭配使用 AWS Batch。

您可以搭配使用的 IAM 功能 AWS Batch

IAM 功能	AWS Batch 支持
身分型政策	是
資源型政策	否
政策動作	是
政策資源	是
政策條件索引鍵	是
ACL	否
ABAC (政策中的標籤)	是
臨時憑證	是
主體許可	是
服務角色	是
服務連結角色	是

若要深入瞭解如何以 AWS Batch 及其他 AWS 服務如何使用大多數 IAM 功能，請參閱 IAM 使用者指南中的搭配 IAM 使用的AWS 服務。

以身分識別為基礎的原則 AWS Batch

支援身分型政策	是

身分型政策是可以連接到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立身分類型政策，請參閱《IAM 使用者指南》中的建立 IAM 政策。

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及在何種條件下允許或拒絕動作。您無法在身分型政策中指定主體，因為這會套用至連接的使用者或角色。如要了解您在 JSON 政策中使用的所有元素，請參閱《IAM 使用者指南》中的 IAM JSON 政策元素參考。

AWS Batch的身分型政策範例

若要檢視以 AWS Batch 身分為基礎的原則範例，請參閱。以身分識別為基礎的原則範例 AWS Batch

的政策動作 AWS Batch

支援政策動作	是

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼条件下可以對什麼資源執行哪些動作。

JSON 政策的 `Action` 元素描述您可以用來允許或拒絕政策中存取的動作。原則動作通常與關聯的 AWS API 作業具有相同的名稱。有一些例外狀況，例如沒有相符的 API 操作的僅限許可動作。也有一些作業需要政策中的多個動作。這些額外的動作稱為相依動作。

政策會使用動作來授予執行相關聯動作的許可。

若要查看 AWS Batch 動作清單，請參閱服務授權參考 AWS Batch中的定義動作。

中的策略動作在動作之前 AWS Batch 使用下列前置詞：

```
batch
```

如需在單一陳述式中指定多個動作，請用逗號分隔。

```
"Action": [
      "batch:action1",
      "batch:action2"
         ]
```

您也可以使用萬用字元 (*) 來指定多個動作。例如，若要指定開頭是 Describe 文字的所有動作，請包含以下動作：

```
"Action": "batch:Describe*"
```

若要檢視以 AWS Batch 身分為基礎的原則範例，請參閱。以身分識別為基礎的原則範例 AWS Batch

的政策資源 AWS Batch

支援政策資源	是

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Resource JSON 政策元素可指定要套用動作的物件。陳述式必須包含 Resource 或 NotResource 元素。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。您可以針對支援特定資源類型的動作 (稱為資源層級許可) 來這麼做。

對於不支援資源層級許可的動作 (例如列出操作)，請使用萬用字元 (*) 來表示陳述式適用於所有資源。

```
"Resource": "*"
```

若要查看 AWS Batch 資源類型及其 ARN 的清單，請參閱服務授權參考資料 AWS Batch中的定義資源。若要了解您可以使用哪些動作指定每個資源的 ARN，請參閱 AWS Batch定義的動作。

若要檢視以 AWS Batch 身分為基礎的原則範例，請參閱。以身分識別為基礎的原則範例 AWS Batch

AWS Batch的政策條件索引鍵

支援服務特定政策條件金鑰	是

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Condition 元素 (或 Condition 區塊) 可讓您指定使陳述式生效的條件。Condition 元素是選用項目。您可以建立使用條件運算子的條件運算式 (例如等於或小於)，來比對政策中的條件和請求中的值。

若您在陳述式中指定多個 Condition 元素，或是在單一 Condition 元素中指定多個索引鍵， AWS 會使用邏輯 AND 操作評估他們。如果您為單一條件索引鍵指定多個值，請使用邏輯OR運算來 AWS 評估條件。必須符合所有條件，才會授與陳述式的許可。

您也可以在指定條件時使用預留位置變數。例如，您可以只在使用者使用其 IAM 使用者名稱標記時，將存取資源的許可授予該 IAM 使用者。如需更多資訊，請參閱 IAM 使用者指南中的 IAM 政策元素：變數和標籤。

AWS 支援全域條件金鑰和服務特定條件金鑰。若要查看所有 AWS 全域條件金鑰，請參閱《IAM 使用者指南》中的AWS 全域條件內容金鑰。

若要查看 AWS Batch 條件索引鍵清單，請參閱服務授權參考 AWS Batch中的條件金鑰。若要瞭解您可以使用條件索引鍵的動作和資源，請參閱動作定義者 AWS Batch。

若要檢視以 AWS Batch 身分為基礎的原則範例，請參閱。以身分識別為基礎的原則範例 AWS Batch

以屬性為基礎的存取控制 (ABAC) 搭配 AWS Batch

支援 ABAC (政策中的標籤)	是

屬性型存取控制 (ABAC) 是一種授權策略，可根據屬性來定義許可。在中 AWS，這些屬性稱為標籤。您可以將標籤附加到 IAM 實體 (使用者或角色) 和許多 AWS 資源。為實體和資源加上標籤是 ABAC 的第一步。您接著要設計 ABAC 政策，允許在主體的標籤與其嘗試存取的資源標籤相符時操作。

ABAC 在成長快速的環境中相當有幫助，並能在政策管理變得繁瑣時提供協助。

若要根據標籤控制存取，請使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 條件金鑰，在政策的條件元素中，提供標籤資訊。

如果服務支援每個資源類型的全部三個條件金鑰，則對該服務而言，值為 Yes。如果服務僅支援某些資源類型的全部三個條件金鑰，則值為 Partial。

如需 ABAC 的詳細資訊，請參閱《IAM 使用者指南》中的什麼是 ABAC?。如要查看含有設定 ABAC 步驟的教學課程，請參閱《IAM 使用者指南》中的使用屬性型存取控制 (ABAC)。

使用臨時登入資料 AWS Batch

支援臨時憑證	是

當您使用臨時憑據登錄時，某些 AWS 服務 不起作用。如需其他資訊，包括哪些 AWS 服務 與臨時登入資料搭配AWS 服務 使用，請參閱 IAM 使用者指南中的 IAM。

如果您使用除了使用者名稱和密碼以外的任何方法登入，則您正在 AWS Management Console 使用臨時認證。例如，當您 AWS 使用公司的單一登入 (SSO) 連結存取時，該程序會自動建立暫時認證。當您以使用者身分登入主控台，然後切換角色時，也會自動建立臨時憑證。如需切換角色的詳細資訊，請參閱《IAM 使用者指南》中的切換至角色 (主控台)。

您可以使用 AWS CLI 或 AWS API 手動建立臨時登入資料。然後，您可以使用這些臨時登入資料來存取 AWS。 AWS 建議您動態產生臨時登入資料，而非使用長期存取金鑰。如需詳細資訊，請參閱 IAM 中的暫時性安全憑證。

AWS Batch的跨服務主體權限

支援轉寄存取工作階段 (FAS)	是

當您使用 IAM 使用者或角色在中執行動作時 AWS，您會被視為主體。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS 會使用主體呼叫的權限 AWS 服務，並結合要求 AWS 服務 向下游服務發出要求。只有當服務收到需要與其 AWS 服務 他資源互動才能完成的請求時，才會發出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱《轉發存取工作階段》。

的服務角色 AWS Batch

支援服務角色	是

服務角色是服務擔任的 IAM 角色，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需更多資訊，請參閱 IAM 使用者指南中的建立角色以委派許可給 AWS 服務。

> **Warning**
>
> 變更服務角色的權限可能會中斷 AWS Batch 功能。只有 AWS Batch 提供指引時，才能編輯服務角色。

服務連結角色 AWS Batch

支援服務連結角色	是

服務連結角色是一種連結至. AWS 服務服務可以擔任代表您執行動作的角色。服務連結角色會顯示在您的中， AWS 帳戶 且屬於服務所有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

如需建立或管理服務連結角色的詳細資訊，請參閱可搭配 IAM 運作的AWS 服務。在表格中尋找服務，其中包含服務連結角色欄中的 Yes。選擇是連結，以檢視該服務的服務連結角色文件。

AWS Batch 執行 IAM 角色

執行角色授予 Amazon ECS 容器和代 AWS Fargate 理程式代表您進行 AWS API 呼叫的權限。

> **Note**
>
> Amazon ECS 容器代理程式 1.16.0 版及更新版本支援執行角色。

需要執行 IAM 角色，具體取決於您的任務需求。您可以針對與您的帳戶相關聯的不同目的和服務擁有多個執行角色。

> **Note**
>
> 如需 Amazon ECS 執行個體角色的相關資訊，請參閱Amazon ECS 執行個體角色。如需有關服務角色的資訊，請參閱如何與 IAM AWS Batch 搭配使用。

Amazon ECS 提供AmazonECSTaskExecutionRolePolicy受管政策。此原則包含上述常見使用案例的必要權限。針對下列特殊使用案例，您可能需要將內嵌政策新增至您的執行角色。

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecr:GetAuthorizationToken",
        "ecr:BatchCheckLayerAvailability",
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchGetImage",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "*"
    }
  ]
}
```

您可以使用下列程序來檢查您的帳戶是否已具有執行角色，並視需要附加受管 IAM 政策。

在 IAM 主控台中檢查 **ecsTaskExecutionRole**

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇角色。
3. 搜尋 `ecsTaskExecutionRole` 的角色清單。如果找不到角色，請參閱建立執行 IAM 角色。如果找到角色，請選擇要檢視附加策略的角色。
4. 在 [權限] 索引標籤上，確認已將 AmazonECS TaskExecution RolePolicy 管理原則附加至角色。如果已附加原則，表示您的執行角色已正確設定。如未連接，請按照以下子步驟連接政策。

   a. 選擇 [新增權限]，然後選擇 [附加原則]

   b. 搜索 AmazonECS TaskExecution RolePolicy。

   c. 勾選 AmazonECs TaskExecution RolePolicy 政策左側的核取方塊，然後選擇附加政策。
5. 選擇 Trust relationships (信任關係)。
6. 確認信任關係包含下列政策。如果信任關係符合以下策略，則會正確設定角色。如果信任關係不相符，請選擇 [編輯信任原則]，輸入下列內容，然後選擇 [更新原則]。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
```

```
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

建立執行 IAM 角色

如果您的帳戶還沒有執行角色，請使用下列步驟建立角色。

若要建立 **ecsTaskExecutionRole** IAM 角色

1. 在以下網址開啟 IAM 主控台：https://console.aws.amazon.com/iam/。
2. 在導覽窗格中，選擇角色。
3. 選擇 Create Role (建立角色)。
4. 針對信任的實體類型，選擇 AWS 服務。
5. 對於服務或使用案例，請選擇 EC2。然後再次選擇 EC2。
6. 選擇下一步。
7. 對於權限策略，請搜索 AmazonECS TaskExecution RolePolicy。
8. 選擇 AmazonECs TaskExecution RolePolicy 政策左側的核取方塊，然後選擇 [下一步]。
9. 在角色名稱中，輸入，`ecsTaskExecutionRole`然後選擇建立角色。

以身分識別為基礎的原則範例 AWS Batch

根據預設，使用者和角色不具備建立或修改 AWS Batch 資源的權限。他們也無法使用 AWS Management Console、 AWS Command Line Interface (AWS CLI) 或 AWS API 來執行工作。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

若要了解如何使用這些範例 JSON 政策文件建立 IAM 身分型政策，請參閱《IAM 使用者指南》中的建立 IAM 政策。

如需有關由定義的動作和資源類型的詳細資訊 AWS Batch，包括每個資源類型的 ARN 格式，請參閱服務授權參考 AWS Batch中的動作、資源和條件索引鍵。

主題

政策最佳實務

以身分識別為基礎的政策會決定某人是否可以建立、存取或刪除您帳戶中的 AWS Batch 資源。這些動作可能會讓您的 AWS 帳戶產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管原則並邁向最低權限權限 — 若要開始將權限授與使用者和工作負載，請使用可授與許多常見使用案例權限的AWS 受管理原則。它們在您的 AWS 帳戶. 建議您透過定義特定於您使用案例的 AWS 客戶管理政策，進一步降低使用權限。如需更多資訊，請參閱 IAM 使用者指南中的 AWS 受管政策或任務職能的AWS 受管政策。
- 套用最低許可許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的權限。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。如需使用 IAM 套用許可的更多相關資訊，請參閱 IAM 使用者指南中的 IAM 中的政策和許可。
- 使用 IAM 政策中的條件進一步限制存取權 – 您可以將條件新增至政策，以限制動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。您也可以使用條件來授與對服務動作的存取權 (如透過特定) 使用這些動作 AWS 服務，例如 AWS CloudFormation。如需更多資訊，請參閱 IAM 使用者指南中的 IAM JSON 政策元素：條件。
- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您編寫安全且實用的政策。如需更多資訊，請參閱 IAM 使用者指南中的 IAM Access Analyzer 政策驗證。
- 需要多因素身份驗證 (MFA) — 如果您的案例需要 IAM 使用者或根使用者 AWS 帳戶，請開啟 MFA 以獲得額外的安全性。若要在呼叫 API 作業時請求 MFA，請將 MFA 條件新增至您的政策。如需更多資訊，請參閱 IAM 使用者指南中的設定 MFA 保護的 API 存取。

如需 IAM 中最佳實務的相關資訊，請參閱 IAM 使用者指南中的 IAM 安全最佳實務。

使用控 AWS Batch 制台

若要存取 AWS Batch 主控台，您必須擁有最少一組權限。這些權限必須允許您列出和檢視有關 AWS 帳戶. AWS Batch 如果您建立比最基本必要許可更嚴格的身分型政策，則對於具有該政策的實體 (使用者或角色) 而言，主控台就無法如預期運作。

您不需要為僅對 AWS CLI 或 AWS API 進行呼叫的使用者允許最低主控台權限。反之，只需允許存取符合他們嘗試執行之 API 操作的動作就可以了。

若要確保使用者和角色仍可使用 AWS Batch 主控台，請同時將 AWS Batch `ConsoleAccess`或受`ReadOnly` AWS 管理的原則附加至實體。如需詳細資訊，請參閱《IAM 使用者指南》中的新增許可到使用者。

允許使用者檢視他們自己的許可

此範例會示範如何建立政策，允許 IAM 使用者檢視附加到他們使用者身分的內嵌及受管政策。此原則包含在主控台上或以程式設計方式使用 AWS CLI 或 AWS API 完成此動作的權限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

預防跨服務混淆代理人

混淆代理人問題屬於安全性問題，其中沒有執行動作許可的實體可以強制具有更多許可的實體執行該動作。在中 AWS，跨服務模擬可能會導致混淆的副問題。在某個服務 (呼叫服務) 呼叫另一個服務 (被呼叫服務) 時，可能會發生跨服務模擬。可以操縱呼叫服務來使用其許可，以其不應有存取許可的方式對其他客戶的資源採取動作。為了預防這種情況， AWS 提供的工具可協助您保護所有服務的資料，而這些服務主體已獲得您帳戶中資源的存取權。

建議您在資源策略中使用aws:SourceArn和aws:SourceAccount全域條件前後關聯索引鍵，以限制將其他服務 AWS Batch 提供給資源的權限。如果 aws:SourceArn 值不包含帳戶 ID (例如 Amazon S3 儲存貯體 ARN)，您必須使用這兩個全域條件內容金鑰來限制許可。如果同時使用這兩個全域條件內容金鑰，且 aws:SourceArn 值包含帳戶 ID，則在相同政策陳述式中使用 aws:SourceAccount 值和 aws:SourceArn 值中的帳戶時，必須使用相同的帳戶 ID。如果您想要僅允許一個資源與跨服務存取相關聯，則請使用 aws:SourceArn。如果您想要允許該帳戶中的任何資源與跨服務使用相關聯，請使用 aws:SourceAccount。

的值aws:SourceArn必須是 AWS Batch 儲存的資源。

防範混淆代理人問題的最有效方法是使用 aws:SourceArn 全域條件內容索引鍵，以及資源的完整 ARN。如果不知道資源的完整 ARN，或者如果您指定了多個資源，請使用 aws:SourceArn 全域內容條件索引鍵搭配萬用字元 (*) 來表示 ARN 的未知部分。例如 arn:aws:*servicename*:*:*123456789012*:*。

下列範例顯示如何在中使用aws:SourceArn和aws:SourceAccount全域條件前後關聯鍵字 AWS Batch 來避免混淆的副問題。

範例 1：僅存取一個計算環境的角色

下列角色只能用於存取一個計算環境。工作名稱必須指定為，*因為工作佇列可與多個計算環境相關聯。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "Service": "batch.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:batch:us-east-1:123456789012:compute-environment/testCE",
            "arn:aws:batch:us-east-1:123456789012:job/*"
          ]
        }
      }
    }
  ]
}
```

範例 2：存取多個運算環境的角色

以下角色可用於存取多個計算環境。工作名稱必須指定為，*因為工作佇列可與多個計算環境相關聯。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "batch.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:batch:us-east-1:123456789012:compute-environment/*",
            "arn:aws:batch:us-east-1:123456789012:job/*"
          ]
        }
```

```
      }
    }
  ]
}
```

疑難排解 AWS Batch 身分和存取

使用下列資訊可協助您診斷和修正使用和 IAM 時可能會遇到的 AWS Batch 常見問題。

主題

- 我未獲授權，不得在 AWS Batch中執行動作
- 我沒有授權執行 iam：PassRole
- 我想允許 AWS 帳戶以外的人員存取我的 AWS Batch 資源

我未獲授權，不得在 AWS Batch中執行動作

如果 AWS Management Console 告訴您您沒有執行動作的授權，則您必須聯絡您的管理員以尋求協助。您的管理員是提供您使用者名稱和密碼的人員。

下列範例錯誤會在 `mateojackson` 使用者嘗試使用主控台檢視一個虛構 *`my-example-widget`* 資源的詳細資訊，但卻無虛構 `batch:`*`GetWidget`* 許可時發生。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 batch:GetWidget on resource: my-example-widget
```

在此情況下，Mateo 會請求管理員更新他的政策，允許他使用 *`my-example-widget`* 動作存取 `batch:`*`GetWidget`* 資源。如需授與傳遞角色之權限的詳細資訊，請參閱授與使用者將角色傳遞至 AWS 服務的權限。

我沒有授權執行 iam：PassRole

如果您收到錯誤，告知您未獲授權執行 `iam:PassRole` 動作，您的政策必須更新，允許您將角色傳遞給 AWS Batch。

有些 AWS 服務 允許您將現有角色傳遞給該服務，而不是建立新的服務角色或服務連結角色。如需執行此作業，您必須擁有將角色傳遞至該服務的許可。

名為 `marymajor` 的 IAM 使用者嘗試使用主控台在 AWS Batch中執行動作時，發生下列範例錯誤。但是，動作要求服務具備服務角色授予的許可。Mary 沒有將角色傳遞至該服務的許可。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在這種情況下，Mary 的政策必須更新，允許她執行 `iam:PassRole` 動作。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我想允許 AWS 帳戶以外的人員存取我的 AWS Batch 資源

您可以建立一個角色，讓其他帳戶中的使用者或您組織外部的人員存取您的資源。您可以指定要允許哪些信任物件取得該角色。針對支援基於資源的政策或存取控制清單 (ACL) 的服務，您可以使用那些政策來授予人員存取您的資源的許可。

如需進一步了解，請參閱以下內容：

- 若要瞭解是否 AWS Batch 支援這些功能，請參閱如何與 IAM AWS Batch 搭配使用。
- 若要了解如何提供對您所擁有資源 AWS 帳戶 的存取權，請參閱 IAM 使用者指南中您擁有的另一 AWS 帳戶 個 IAM 使用者提供存取權限。
- 若要了解如何將資源存取權提供給第三方 AWS 帳戶，請參閱 IAM 使用者指南中的提供第三方 AWS 帳戶 擁有的存取權。
- 若要了解如何透過聯合身分提供存取權，請參閱 IAM 使用者指南中的將存取權提供給在外部進行身分驗證的使用者 (聯合身分)。
- 若要了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱 IAM 使用者指南中的 IAM 角色與資源型政策的差異。

使用服務連結角色 AWS Batch

AWS Batch 使用 AWS Identity and Access Management (IAM) 服務連結角色。服務連結角色是直接連結到 AWS Batch的唯一 IAM 角色類型。服務連結角色由預先定義， AWS Batch 並包含服務代表您呼叫其他服 AWS 務所需的所有權限。

服務連結角色可讓您 AWS Batch 更輕鬆地設定，因為您不需要手動新增必要的權限。 AWS Batch 定義其服務連結角色的權限，除非另有定義，否則只 AWS Batch 能擔任其角色。定義的許可包括信任政策和許可政策，且該許可政策無法附加至其他 IAM 實體。

> **Note**
>
> 執行下列其中一項動作，以指定 AWS Batch 計算環境的服務角色。

- 使用空字串做為服務角色。這可讓您 AWS Batch 建立服務角色。
- 以下列格式指定服務角色：`arn:aws:iam::account_number:role/aws-service-role/batch.amazonaws.com/AWSServiceRoleForBatch`

若要取得更多資訊，請參閱《 AWS Batch 使用指南》the section called “角色名稱或 ARN 不正確”中的。

您必須先刪除服務連結角色的相關資源，才能將其刪除。如此可保護您 AWS Batch 的資源，避免您不小心移除資源的存取許可。

如需關於支援服務連結角色的其他服務的資訊，請參閱可搭配 IAM 運作的AWS 服務，並尋找服務連結角色欄顯示為是的服務。選擇具有連結的是，以檢視該服務的服務連結角色文件。

服務連結角色權限 AWS Batch

AWS Batch 使用名為AWSServiceRoleForBatch的服務連結角色。AWSServiceRoleForBatch角色可 AWS Batch 讓您建立和管理 AWS 資源。

服AWSServiceRoleForBatch務連結角色會信任`batch.amazonaws.com`服務主體擔任該角色。

名為的 IAM 政策BatchServiceRolePolicy允許 AWS Batch 對特定資源完成以下動作：

- `autoscaling`— 允許創 AWS Batch 建和管理 Amazon EC2 Auto Scaling 資源。 AWS Batch 為大多數運算環境建立和管理 Amazon EC2 Auto Scaling 群組。
- `ec2`— AWS Batch 允許控制 Amazon EC2 執行個體的生命週期，以及建立和管理啟動範本和標籤。AWS Batch 針對某些 EC2 競價型運算環境建立和管理 EC2 競價型叢集請求。
- `ecs`-允許 AWS Batch 建立和管理 Amazon ECS 叢集、任務定義和任務執行任務。
- `eks`-允許描述 AWS Batch 用於驗證的 Amazon EKS 叢集資源。
- `iam`- AWS Batch 允許驗證所有者提供的角色並將其傳遞給 Amazon EC2，Amazon EC2 Auto Scaling 和 Amazon ECS。
- `logs`— 可建 AWS Batch 立和管理 AWS Batch 工作的記錄群組和記錄資料流。

您必須設定許可，IAM 實體 (如使用者、群組或角色) 才可建立、編輯或刪除服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的服務連結角色許可。

建立服務連結角色 AWS Batch

您不需要手動建立一個服務連結角色。當您 CreateComputeEnvironment 在 AWS Management Console AWS CLI、或 AWS API 中且未指定`serviceRole`參數值時，會為您 AWS Batch 建立服務連結角色。

> **Important**
>
> 此服務連結角色可以顯示在您的帳戶，如果您於其他服務中完成一項動作時，可以使用支援此角色的功能。此外，如果您在 2021 年 3 月 10 日之前使用該 AWS Batch 服務，則該服務開始支援服務連結角色時，請在您的帳戶中 AWS Batch 建立該 AWSServiceRoleForBatch 角色。若要進一步了解，請參閱我的 IAM 帳戶中出現的新角色。

若您刪除此服務連結角色，之後需要再次建立，您可以在帳戶中使用相同程序重新建立角色。當您時 CreateComputeEnvironment，請再次為您 AWS Batch 建立服務連結角色。

編輯下列項目的服務連結角色 AWS Batch

使用時 AWS Batch，您無法編輯 AWSServiceRoleForBatch 服務連結角色。因為可能有各種實體會參考服務連結角色，所以您無法在建立角色之後變更其名稱。然而，您可使用 IAM 來編輯角色描述。如需更多資訊，請參閱 IAM 使用者指南中的編輯服務連結角色。

允許 IAM 實體編輯 AWSServiceRoleForBatch 服務連結角色的說明

將下列陳述式新增至權限原則。這可讓 IAM 實體編輯服務連結角色的說明。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:UpdateRoleDescription"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/batch.amazonaws.com/
AWSServiceRoleForBatch",
    "Condition": {"StringLike": {"iam:AWSServiceName": "batch.amazonaws.com"}}
}
```

刪除的服務連結角色 AWS Batch

如果您不再需要使用需要服務連結角色的功能或服務，建議您刪除該角色。如此一來，您就沒有未使用的實體不受主動監視或維護。然而，在手動刪除服務連結角色之前，您必須先清除資源。

允許 IAM 實體刪除 AWSServiceRoleForBatch 服務連結角色

將下列陳述式新增至權限原則。這可讓 IAM 實體刪除服務連結角色。

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/batch.amazonaws.com/
AWSServiceRoleForBatch",
    "Condition": {"StringLike": {"iam:AWSServiceName": "batch.amazonaws.com"}}
}
```

清除服務連結角色

在使用 IAM 刪除服務連結角色之前，您必須先確認該角色沒有作用中工作階段，並刪除在單一分割區中所有 AWS 區域中使用該角色的所有 AWS Batch 運算環境。

檢查服務連結角色是否有作用中工作階段

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 在導覽窗格中，選擇 [角色]，然後選取 AWSServiceRoleForBatch 名稱 (而非核取方塊)。
3. 在 Summary (摘要) 頁面上，選擇 Access Advisor (存取 Advisor)，然後檢閱服務連結角色的近期活動。

> **Note**
>
> 如果您不知道 AWS Batch 是否正在使用該 AWSServiceRoleForBatch 角色，則可以嘗試刪除該角色。如果服務使用的是角色，則該角色將無法刪除。您可以檢視使用角色的「區域」。如果服務正在使用該角色，您必須先等到工作階段結束，才能刪除該角色。您無法撤銷服務連結角色的工作階段。

若要移除 AWSServiceRoleForBatch 服務連結角色所使用的 AWS Batch 資源

您必須先刪除所有 AWS 區域中使用此 AWSServiceRoleForBatch 角色的所有 AWS Batch 計算環境，才能刪除 AWSServiceRoleForBatch 角色。

1. 請在以下位置開啟 AWS Batch 主控台。 https://console.aws.amazon.com/batch/

2. 從導覽列中選取要使用的「區域」。
3. 在導覽窗格中，選擇 Compute environments (運算環境)。
4. 選取運算環境。
5. 選擇停用。等待「狀態」變更為「已停用」。
6. 選取運算環境。
7. 選擇刪除。選擇刪除計算環境，確認您要刪除計算環境。
8. 針對在所有區域中使用服務連結角色的所有計算環境重複步驟 1—7。

刪除 IAM (主控台) 中的服務連結角色

您可以使用 IAM 主控台刪除服務連結角色。

刪除服務連結角色 (主控台)

1. 登入 AWS Management Console 並開啟 IAM 主控台，網址為 https://console.aws.amazon.com/iam/。
2. 在 IAM 主控台的導覽窗格中，選擇角色。然後選取旁邊的核取方塊 AWSServiceRoleForBatch，而不是名稱或列本身。
3. 選擇 Delete role (刪除角色)。
4. 在確認對話方塊中，檢閱服務上次存取資料，以顯示每個所選取角色上次存取 AWS 服務的時間。這可協助您確認角色目前是否作用中。如果您想要繼續進行，請選擇 Yes, Delete (是，刪除) 來提交服務連結角色以進行刪除。
5. 查看 IAM 主控台通知，監視服務連結角色刪除的進度。因為 IAM 服務連結角色刪除不同步，所以在您提交角色進行刪除之後，刪除任務可能會成功或失敗。
   - 如果任務成功，則會從清單中移除角色，而且成功通知會出現在頁面頂端。
   - 如果任務失敗，您可以從通知中選擇 View details (檢視詳細資訊) 或 View Resources (檢視資源)，以了解刪除失敗的原因。如果刪除因角色使用服務資源而失敗，則服務傳回該資訊時，通知會包含資源清單。您接著可以清除資源，並重新提交刪除。

> **Note**
>
> 根據服務所傳回的資訊，您可能需要重複此程序數次。例如，您的服務連結角色可能會使用六個資源，而且您的服務可能傳回其中五項的相關資訊。如果您清除五個資源，並

重新提交刪除角色，則刪除會失敗，而且服務會報告還有一個資源。服務可能會傳回所有資源、其中一些資源，或未報告任何資源。

- 如果任務失敗，而且通知未包含資源清單，則服務可能未傳回該資訊。若要瞭解如何清除該服務的資源，請參閱使用 IAM 的AWS 服務。請在表格中找到您的服務，然後選擇 Yes (是) 連結，檢視該服務的服務連結角色文件。

刪除 IAM ()AWS CLI中的服務連結角色

您可以使用的 IAM 命令 AWS Command Line Interface 來刪除服務連結角色。

刪除服務連結角色 (CLI)

1. 因為無法刪除正在使用或具有相關聯資源的服務連結角色，所以您必須提交刪除要求。如果不符合這些條件，則可以拒絕該請求。您必須從回應中擷取 `deletion-task-id`，以檢查刪除任務的狀態。輸入下列命令，以提交服務連結角色刪除要求：

```
$ aws iam delete-service-linked-role --role-name AWSServiceRoleForBatch
```

2. 使用下列命令，以檢查刪除任務的狀態：

```
$ aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
```

刪除任務的狀態可以是 NOT_STARTED、IN_PROGRESS、SUCCEEDED 或 FAILED。如果刪除失敗，則呼叫會傳回失敗原因，以進行疑難排解。如果刪除因角色使用服務資源而失敗，則服務傳回該資訊時，通知會包含資源清單。您接著可以清除資源，並重新提交刪除。

> **Note**
>
> 根據服務所傳回的資訊，您可能需要重複此程序數次。例如，您的服務連結角色可能會使用六個資源，而且您的服務可能傳回其中五項的相關資訊。如果您清除五個資源，並重新提交刪除角色，則刪除會失敗，而且服務會報告還有一個資源。服務可能會傳回所有資源，其中一些資源。或者，它可能不會報告任何資源。要了解如何清理未報告任何資源的服務的資源，請參閱與 IAM 搭配使用的AWS 服務。請在表格中找到您的服務，然後選擇 Yes (是) 連結，檢視該服務的服務連結角色文件。

刪除 IAM (AWS API) 中的服務連結角色

您可以使用 IAM API 刪除服務連結角色。

刪除服務連結角色 (API)

1. 要提交服務鏈接卷的刪除請求，請致電DeleteServiceLinkedRole。在請求中，指定 AWSServiceRoleForBatch 角色名稱。

   因為無法刪除正在使用或具有相關聯資源的服務連結角色，所以您必須提交刪除要求。如果不符合這些條件，則可以拒絕該請求。您必須從回應中擷取 `DeletionTaskId`，以檢查刪除任務的狀態。
2. 要檢查刪除狀態，請致電GetServiceLinkedRoleDeletionStatus。在請求中，指定 `DeletionTaskId`。

   刪除任務的狀態可以是 `NOT_STARTED`、`IN_PROGRESS`、`SUCCEEDED` 或 `FAILED`。如果刪除失敗，則呼叫會傳回失敗原因，以進行疑難排解。如果刪除因角色使用服務資源而失敗，則服務傳回該資訊時，通知會包含資源清單。您接著可以清除資源，並重新提交刪除。

   > **Note**
   >
   > 根據服務所傳回的資訊，您可能需要重複此程序數次。例如，您的服務連結角色可能會使用六個資源，而且您的服務可能傳回其中五項的相關資訊。如果您清除五個資源，並重新提交刪除角色，則刪除會失敗，而且服務會報告還有一個資源。服務可能會傳回所有資源、其中一些資源，或未報告任何資源。若要瞭解如何清除未報告任何資源之服務的資源，請參閱使用 IAM 的AWS 服務。請在表格中找到您的服務，然後選擇 Yes (是) 連結，檢視該服務的服務連結角色文件。

支援 AWS Batch 服務連結角色的區域

AWS Batch 支援在所有提供服務的區域中使用服務連結角色。如需詳細資訊，請參閱 AWS Batch 端點。

AWS 受管理的政策 AWS Batch

您可以使用 AWS 受管理的原則，為您的團隊和佈建的 AWS 資源簡化身分存取管理。 AWS 受管政策涵蓋各種常見使用案例，依預設可在您的 AWS 帳戶中使用，並代表您進行維護和更新。您無法變更

AWS 受管理原則中的權限。如果您需要更大的彈性，也可以選擇建立 IAM 客戶受管政策。如此一來，您就可以僅為團隊佈建的資源提供他們所需的確切權限。

如需 AWS 受管政策的詳細資訊，請參閱 IAM 使用者指南中的AWS 受管政策。

AWS 服務會代表您維護及更新 AWS 受管理的政策。 AWS 服務會定期將其他權限新增至受 AWS 管理的策略。 AWS 當新功能啟動或作業可用時，最有可能會更新受管理的策略。這些更新會自動影響附加原則的所有身分識別 (使用者、群組和角色)。但是，它們不會移除權限或破壞您現有的權限。

此外，還 AWS 支援跨多個服務之工作職能的受管理原則。例如，`ReadOnlyAccess` AWS 受管理的策略提供對所有 AWS 服務和資源的唯讀存取權。當服務啟動新功能時，會為新作業和資源新 AWS 增唯讀權限。如需任務職能政策的清單和說明，請參閱 IAM 使用者指南中有關任務職能的AWS 受管政策。

AWS 受管理策略：BatchServiceRolePolicy

`AWSServiceRoleForBatch`服務連結角色會使用受BatchServiceRolePolicy管 IAM 政策。這允 AWS Batch 許代表您執行操作。您無法將此政策連接至 IAM 實體。如需詳細資訊，請參閱 使用服務連結角色 AWS Batch。

此原則可 AWS Batch 針對特定資源完成下列動作：

- `autoscaling`— 允許創 AWS Batch 建和管理 Amazon EC2 Auto Scaling 資源。 AWS Batch 為大多數運算環境建立和管理 Amazon EC2 Auto Scaling 群組。
- `ec2`— AWS Batch 允許控制 Amazon EC2 執行個體的生命週期，以及建立和管理啟動範本和標籤。AWS Batch 針對某些 EC2 競價型運算環境建立和管理 EC2 競價型叢集請求。
- `ecs`-允許 AWS Batch 建立和管理 Amazon ECS 叢集、任務定義和任務執行任務。
- `eks`-允許描述 AWS Batch 用於驗證的 Amazon EKS 叢集資源。
- `iam`- AWS Batch 允許驗證所有者提供的角色並將其傳遞給 Amazon EC2，Amazon EC2 Auto Scaling 和 Amazon ECS。
- `logs`— 可建 AWS Batch 立和管理 AWS Batch 工作的記錄群組和記錄資料流。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "AWSBatchPolicyStatement1",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeAccountAttributes",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeImages",
                "ec2:DescribeImageAttribute",
                "ec2:DescribeSpotInstanceRequests",
                "ec2:DescribeSpotFleetInstances",
                "ec2:DescribeSpotFleetRequests",
                "ec2:DescribeSpotPriceHistory",
                "ec2:DescribeSpotFleetRequestHistory",
                "ec2:DescribeVpcClassicLink",
                "ec2:DescribeLaunchTemplateVersions",
                "ec2:RequestSpotFleet",
                "autoscaling:DescribeAccountLimits",
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:DescribeLaunchConfigurations",
                "autoscaling:DescribeAutoScalingInstances",
                "autoscaling:DescribeScalingActivities",
                "eks:DescribeCluster",
                "ecs:DescribeClusters",
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTaskDefinition",
                "ecs:DescribeTasks",
                "ecs:ListClusters",
                "ecs:ListContainerInstances",
                "ecs:ListTaskDefinitionFamilies",
                "ecs:ListTaskDefinitions",
                "ecs:ListTasks",
                "ecs:DeregisterTaskDefinition",
                "ecs:TagResource",
                "ecs:ListAccountSettings",
                "logs:DescribeLogGroups",
                "iam:GetInstanceProfile",
                "iam:GetRole"
            ],
            "Resource": "*"
        },
```

```
        {
            "Sid": "AWSBatchPolicyStatement2",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/batch/job*"
        },
        {
            "Sid": "AWSBatchPolicyStatement3",
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/batch/job*:log-stream:*"
        },
        {
            "Sid": "AWSBatchPolicyStatement4",
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateOrUpdateTags"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement5",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn",
                        "ecs-tasks.amazonaws.com"
                    ]
```

```
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement6",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "spot.amazonaws.com",
                        "spotfleet.amazonaws.com",
                        "autoscaling.amazonaws.com",
                        "ecs.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement7",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateLaunchTemplate"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement8",
            "Effect": "Allow",
            "Action": [
                "ec2:TerminateInstances",
                "ec2:CancelSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "ec2:DeleteLaunchTemplate"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
```

```
                    "aws:ResourceTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement9",
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateLaunchConfiguration",
                "autoscaling:DeleteLaunchConfiguration"
            ],
            "Resource":
"arn:aws:autoscaling:*:*:launchConfiguration:*:launchConfigurationName/AWSBatch*"
        },
        {
            "Sid": "AWSBatchPolicyStatement10",
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:SetDesiredCapacity",
                "autoscaling:DeleteAutoScalingGroup",
                "autoscaling:SuspendProcesses",
                "autoscaling:PutNotificationConfiguration",
                "autoscaling:TerminateInstanceInAutoScalingGroup"
            ],
            "Resource":
"arn:aws:autoscaling:*:*:autoScalingGroup:*:autoScalingGroupName/AWSBatch*"
        },
        {
            "Sid": "AWSBatchPolicyStatement11",
            "Effect": "Allow",
            "Action": [
                "ecs:DeleteCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:StopTask"
            ],
            "Resource": "arn:aws:ecs:*:*:cluster/AWSBatch*"
        },
        {
            "Sid": "AWSBatchPolicyStatement12",
            "Effect": "Allow",
```

```
            "Action": [
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:StopTask"
            ],
            "Resource": "arn:aws:ecs:*:*:task-definition/*"
        },
        {
            "Sid": "AWSBatchPolicyStatement13",
            "Effect": "Allow",
            "Action": [
                "ecs:StopTask"
            ],
            "Resource": "arn:aws:ecs:*:*:task/*/*"
        },
        {
            "Sid": "AWSBatchPolicyStatement14",
            "Effect": "Allow",
            "Action": [
                "ecs:CreateCluster",
                "ecs:RegisterTaskDefinition"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement15",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:launch-template/*",
                "arn:aws:ec2:*:*:placement-group/*",
                "arn:aws:ec2:*:*:capacity-reservation/*",
```

```
                "arn:aws:ec2:*:*:elastic-gpu/*",
                "arn:aws:elastic-inference:*:*:elastic-inference-accelerator/*",
                "arn:aws:resource-groups:*:*:group/*"
            ]
        },
        {
            "Sid": "AWSBatchPolicyStatement16",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement17",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": [
                        "RunInstances",
                        "CreateLaunchTemplate",
                        "RequestSpotFleet"
                    ]
                }
            }
        }
    ]
}
```

AWS 管理策略：AWSBatchServiceRole策略

名為的角色權限原則AWSBatchServiceRole AWS Batch 允許對特定資源完成下列動作：

受AWSBatchServiceRole管 IAM 政策通常由名為的角色使用，AWSBatchServiceRole並包含以下許可。遵循授與最少權限的標準安全性建議，可以使用AWSBatchServiceRole受管理的策略作為指南。如果您的使用案例不需要受管政策中授予的任何許可，請建立自訂政策並僅新增您需要的許可。這個 AWS Batch less-error-prone受管理的原則和角色可用於大部分的運算環境類型，但服務連結角色的使用方式較適合用於更佳範圍和改善的受管理體驗。

- `autoscaling`— 允許創 AWS Batch 建和管理 Amazon EC2 Auto Scaling 資源。 AWS Batch 為大多數運算環境建立和管理 Amazon EC2 Auto Scaling 群組。
- `ec2`— AWS Batch 允許管理 Amazon EC2 執行個體的生命週期，以及建立和管理啟動範本和標籤。AWS Batch 針對某些 EC2 競價型運算環境建立和管理 EC2 競價型叢集請求。
- `ecs`-允許 AWS Batch 建立和管理 Amazon ECS 叢集、任務定義和任務執行任務。
- `iam`- AWS Batch 允許驗證所有者提供的角色並將其傳遞給 Amazon EC2，Amazon EC2 Auto Scaling 和 Amazon ECS。
- `logs`— 可建 AWS Batch 立和管理 AWS Batch 工作的記錄群組和記錄資料流。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSBatchPolicyStatement1",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeAccountAttributes",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeImages",
                "ec2:DescribeImageAttribute",
                "ec2:DescribeSpotInstanceRequests",
                "ec2:DescribeSpotFleetInstances",
                "ec2:DescribeSpotFleetRequests",
                "ec2:DescribeSpotPriceHistory",
                "ec2:DescribeSpotFleetRequestHistory",
                "ec2:DescribeVpcClassicLink",
                "ec2:DescribeLaunchTemplateVersions",
                "ec2:CreateLaunchTemplate",
```

```
                "ec2:DeleteLaunchTemplate",
                "ec2:RequestSpotFleet",
                "ec2:CancelSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "ec2:TerminateInstances",
                "ec2:RunInstances",
                "autoscaling:DescribeAccountLimits",
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:DescribeLaunchConfigurations",
                "autoscaling:DescribeAutoScalingInstances",
                "autoscaling:DescribeScalingActivities",
                "autoscaling:CreateLaunchConfiguration",
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:SetDesiredCapacity",
                "autoscaling:DeleteLaunchConfiguration",
                "autoscaling:DeleteAutoScalingGroup",
                "autoscaling:CreateOrUpdateTags",
                "autoscaling:SuspendProcesses",
                "autoscaling:PutNotificationConfiguration",
                "autoscaling:TerminateInstanceInAutoScalingGroup",
                "ecs:DescribeClusters",
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTaskDefinition",
                "ecs:DescribeTasks",
                "ecs:ListAccountSettings",
                "ecs:ListClusters",
                "ecs:ListContainerInstances",
                "ecs:ListTaskDefinitionFamilies",
                "ecs:ListTaskDefinitions",
                "ecs:ListTasks",
                "ecs:CreateCluster",
                "ecs:DeleteCluster",
                "ecs:RegisterTaskDefinition",
                "ecs:DeregisterTaskDefinition",
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:UpdateContainerAgent",
                "ecs:DeregisterContainerInstance",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogGroups",
```

```
                "iam:GetInstanceProfile",
                "iam:GetRole"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AWSBatchPolicyStatement2",
            "Effect": "Allow",
            "Action": "ecs:TagResource",
            "Resource": [
                "arn:aws:ecs:*:*:task/*_Batch_*"
            ]
        },
        {
            "Sid": "AWSBatchPolicyStatement3",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn",
                        "ecs-tasks.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement4",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "spot.amazonaws.com",
                        "spotfleet.amazonaws.com",
                        "autoscaling.amazonaws.com",
                        "ecs.amazonaws.com"
                    ]
                }
```

```
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement5",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "RunInstances"
                }
            }
        }
    ]
}
```

AWS 受管理策略：AWSBatchFullAccess

此原AWSBatchFullAccess則會授與 AWS Batch 動作對 AWS Batch 資源的完整存取權。它還授予 Amazon EC2、Amazon ECS、亞馬遜 EKS 和 IAM 服務的描述和列出動作存取權限。 CloudWatch如此一來，IAM 身分 (使用者或角色) 都可以檢視代表他們建立的 AWS Batch 受管資源。最後，此政策還允許將選定的 IAM 角色傳遞給這些服務。

您可以附加AWSBatchFullAccess到 IAM 實體。 AWS Batch 也會將此原則附加至允許代表您執 AWS Batch 行動作的服務角色。

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "batch:*",
        "cloudwatch:GetMetricStatistics",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeVpcs",
        "ec2:DescribeImages",
```

```
        "ec2:DescribeLaunchTemplates",
        "ec2:DescribeLaunchTemplateVersions",
        "ecs:DescribeClusters",
        "ecs:Describe*",
        "ecs:List*",
        "eks:DescribeCluster",
        "eks:ListClusters",
        "logs:Describe*",
        "logs:Get*",
        "logs:TestMetricFilter",
        "logs:FilterLogEvents",
        "iam:ListInstanceProfiles",
        "iam:ListRoles"
      ],
      "Resource":"*"
    },
    {
      "Effect":"Allow",
      "Action":[
        "iam:PassRole"
      ],
      "Resource":[
        "arn:aws:iam::*:role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/service-role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/ecsInstanceRole",
        "arn:aws:iam::*:instance-profile/ecsInstanceRole",
        "arn:aws:iam::*:role/iaws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/aws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/AWSBatchJobRole*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "iam:CreateServiceLinkedRole"
      ],
      "Resource":"arn:aws:iam::*:role/*Batch*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "batch.amazonaws.com"
          }
      }
    }
  ]
```

```
}
```

AWS BatchAWS 受管理策略的更新

檢視 AWS Batch 自此服務開始追蹤這些變更以來的 AWS 受管理策略更新詳細資料。如需有關此頁面變更的自動警示，請訂閱「 AWS Batch 文件歷史記錄」頁面上的 RSS 摘要。

變更	描述	日期
BatchServiceRolePolicy政策已更新	已更新以新增描述 Spot 叢集請求歷史記錄和 Amazon EC2 Auto Scaling 活動的支援。	2023 年 12 月 5 日
AWSBatchServiceRole已新增原則	已更新為新增陳述式 ID、授`ec2:DescribeSpotFleetRequestHistory` 與與 AWS Batch 權限`autoscaling:DescribeScalingActivities` 。	2023 年 12 月 5 日
BatchServiceRolePolicy政策已更新	已更新以新增對描述 Amazon EKS 叢集的支援。	2022 年 10 月 20 日
AWSBatchFullAccess政策已更新	已更新為新增列出和描述 Amazon EKS 叢集的支援。	2022 年 10 月 20 日
BatchServiceRolePolicy政策已更新	已更新為新增對受管理之 Amazon EC2 容量保留群組的支援 AWS Resource Groups。如需詳細資訊，請參閱 Amazon EC2 使用者指南中的使用容量保留群組。	2022 年 5 月 18 日
BatchServiceRolePolicy和AWSBatchServiceRole政策已更新	已更新為新增說明 Amazon EC2 中 AWS Batch 受管執行個體狀態的支援，以便取代運作狀態不良的執行個體。	2021 年 12 月 6 日

變更	描述	日期
BatchServiceRolePolicy政策已更新	已更新，可在 Amazon EC2 中新增對置放群組、容量保留、彈性 GPU 和 Elastic Inference 資源的支援。	2021 年 3 月 26 日
BatchServiceRolePolicy已新增原則	透過AWSServiceRoleForBatch服務連結角色的BatchServiceRolePolicy受管理原則，您可以使用由所管理的服務連結角色。 AWS Batch有了這個原則，您就不需要維護自己的角色，就能在您的運算環境中使用。	2021 年 3 月 10 日
AWSBatchFullAccess-新增新增服務連結角色的權限	新增 IAM 許可，以允許將AWSServiceRoleForBatch服務連結角色新增至帳戶。	2021 年 3 月 10 日
AWS Batch 開始追蹤變更	AWS Batch 開始追蹤其 AWS 受管理策略的變更。	2021 年 3 月 10 日

使 AWS Batch 用介面端點存取

您可 AWS PrivateLink 以使用在 VPC 和 AWS Batch. 您可以 AWS Batch 像在 VPC 中一樣進行存取，而無需使用網際網路閘道、NAT 裝置、VPN 連線或 AWS Direct Connect 連線。VPC 中的執行個體不需要公用 IP 位址即可存取 AWS Batch。

您可以建立由 AWS PrivateLink提供支援的介面端點來建立此私有連線。我們會在您為介面端點啟用的每個子網中建立端點網路介面。這些是請求者管理的網路介面，可作為目的地為 AWS Batch之流量的進入點。

如需詳細資訊，請參閱AWS PrivateLink 指南中的介面 VPC 端點。

的注意事項 AWS Batch

設定的介面端點之前 AWS Batch，請先檢閱AWS PrivateLink 指南中的介面端點內容和限制。

AWS Batch 支援透過介面端點呼叫其所有 API 動作。

在設定的介面 VPC 端點之前 AWS Batch，請注意下列考量事項：

- 使用 Fargate 資源啟動類型的任務不需要 Amazon ECS 的介面虛擬私人雲端節點，但您可能需要接口 VPC 端點 AWS Batch、Amazon ECR、Secrets Manager 或 Amazon CloudWatch 日誌，如以下幾點所述。
  - 若要執行任務，您必須為 Amazon ECS 建立介面 VPC 人雲端端點。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的介面 VPC 端點 (AWS PrivateLink)。
  - 若要允許您的任務從 Amazon ECR 提取私有映像檔，您必須為 Amazon ECR 建立介面 VPC 私人雲端節點。如需詳細資訊，請參閱 《Amazon Elastic Container Registry 使用者指南》中的介面 VPC 端點 (AWS PrivateLink)。
  - 若要允許您的工作從 Secrets Manager 提取敏感資料，您必須為 Secrets Manager 員建立介面 VPC 人雲端端點。如需詳細資訊，請參閱 《AWS Secrets Manager 使用者指南》中的搭配使用 Secrets Manager 與 VPC 端點。
  - 如果您的 VPC 沒有網際網路閘道，而您的工作使用記`awslogs`錄驅動程式將記錄資訊傳送至 CloudWatch 記錄檔，則必須為 CloudWatch 記錄檔建立介面 VPC 端點。如需詳細資訊，請參閱 Amazon CloudWatch 日誌使用指南中的將日 CloudWatch 誌與界面 VPC 端點搭配使用。
- 使用 EC2 資源的任務需要其啟動的容器執行個體，才能執行 Amazon ECS 容器代理程式的版本`1.25.1`或更新版本。如需詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 Amazon ECS Linux 容器代理程式版本。
- VPC 端點目前不支援跨區域請求。請確實在計劃發出 AWS Batch API 呼叫的相同區域中建立端點。
- 透過 Amazon Route 53，VPC 端點僅支援 Amazon 提供的 DNS。如果您想要使用自己的 DNS，您可以使用條件式 DNS 轉送。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 DHCP 選項集。
- 連接到 VPC 端點的安全群組，必須允許從 VPC 的私有子網路，透過 443 埠傳入的連線。
- AWS Batch 在下列情況中不支援 VPC 介面端點： AWS 區域
  - 亞太區域 (大阪) (`ap-northeast-3`)
  - 亞太區域 (雅加達) (`ap-southeast-3`)

建立的介面端點 AWS Batch

您可以建立介面端點以 AWS Batch 使用 Amazon VPC 主控台或 AWS Command Line Interface (AWS CLI)。如需詳細資訊，請參閱《AWS PrivateLink 指南》中的建立介面端點。

建立 AWS Batch 使用下列服務名稱的介面端點：

```
com.amazonaws.region.batch
```

例如：

```
com.amazonaws.us-east-2.batch
```

在分aws-cn區中，格式不同：

```
cn.com.amazonaws.region.batch
```

例如：

```
cn.com.amazonaws.cn-northwest-1.batch
```

如果您為介面端點啟用私有 DNS，您可以 AWS Batch 使用其預設的區域 DNS 名稱向 API 要求。例如 `batch.us-east-1.amazonaws.com`。

如需詳細資訊，請參閱AWS PrivateLink 南中的透過介面端點存取服務。

為您的介面端點建立端點政策

端點政策為 IAM 資源，您可將其連接至介面端點。預設端點策略允許 AWS Batch 透過介面端點進行完整存取。若要控制允許 AWS Batch 從您的 VPC 存取，請將自訂端點原則附加到介面端點。

端點政策會指定以下資訊：

- 可以執行動作 (AWS 帳戶、使用者和 IAM 角色) 的主體。
- 可執行的動作。
- 可供執行動作的資源。

如需詳細資訊，請參閱《AWS PrivateLink 指南》中的使用端點政策控制對服務的存取。

範例：用於動作的 VPC 端點原則 AWS Batch

以下是自訂端點政策的範例。當您將此原則附加到介面端點時，它會授與所有資源上所有主參與者所列 AWS Batch 動作的存取權。

```
{
   "Statement": [
```

```
      {
         "Principal": "*",
         "Effect": "Allow",
         "Action": [
            "batch:SubmitJob",
            "batch:ListJobs",
            "batch:DescribeJobs"
         ],
         "Resource":"*"
      }
   ]
}
```

符合性驗證 AWS Batch

若要瞭解 AWS 服務 是否屬於特定規範遵循方案的範圍內，請參閱AWS 服務 遵循規範計劃方案中的，並選擇您感興趣的合規方案。如需一般資訊，請參閱AWS 規範計劃AWS。

您可以使用下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱下載中的報告中的 AWS Artifact。

您在使用時的合規責任取決 AWS 服務 於資料的敏感性、公司的合規目標以及適用的法律和法規。AWS 提供下列資源以協助遵循法規：

- 安全性與合規性快速入門指南 — 這些部署指南討論架構考量，並提供部署以安全性和合規性 AWS 為重點的基準環境的步驟。
- 在 Amazon Web Services 上架構 HIPAA 安全性與合規性 — 本白皮書說明公司如何使用建立符合 HIPAA 資格的應 AWS 用程式。

> **Note**
>
> 並非所有人 AWS 服務 都符合 HIPAA 資格。如需詳細資訊，請參閱 HIPAA 資格服務參照。

- AWS 合規資源AWS — 此工作簿和指南集合可能適用於您的產業和所在地。
- AWS 客戶合規指南 — 透過合規的角度瞭解共同的責任模式。這份指南總結了在多個架構 (包括美國國家標準技術研究所 (NIST)、支付卡產業安全標準委員會 (PCI) 和國際標準化組織 (ISO)) 中，保 AWS 服務 護指引並對應至安全控制的最佳實務。
- 使用AWS Config 開發人員指南中的規則評估資源 — 此 AWS Config 服務會評估您的資源組態符合內部實務、產業準則和法規的程度。

- AWS Security Hub— 這 AWS 服務 提供了內部安全狀態的全面視圖 AWS。Security Hub 使用安全控制，可評估您的 AWS 資源並檢查您的法規遵循是否符合安全業界標準和最佳實務。如需支援的服務和控制清單，請參閱 Security Hub controls reference。
- Amazon GuardDuty — 透過監控環境中的 AWS 帳戶可疑和惡意活動，藉此 AWS 服務 偵測您的工作負載、容器和資料的潛在威脅。 GuardDuty 可協助您滿足特定合規性架構所要求的入侵偵測需求，如 PCI DSS 等各種合規性需求。
- AWS Audit Manager— 這 AWS 服務 有助於您持續稽核您的 AWS 使用情況，以簡化您管理風險的方式，以及遵守法規和業界標準的方式。

基礎結構安全 AWS Batch

作為託管服務， AWS Batch 受到 AWS 全球網絡安全的保護。有關 AWS 安全服務以及如何 AWS 保護基礎結構的詳細資訊，請參閱AWS 雲端安全 若要使用基礎架構安全性的最佳做法來設計您的 AWS 環境，請參閱安全性支柱架構良 AWS 好的架構中的基礎結構保護。

您可以使用 AWS 已發佈的 API 呼叫透 AWS Batch 過網路進行存取。使用者端必須支援下列專案：

- Transport Layer Security (TLS)。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 具備完美轉送私密(PFS)的密碼套件，例如 DHE (Ephemeral Diffie-Hellman)或 ECDHE (Elliptic Curve Ephemeral Diffie-Hellman)。現代系統(如 Java 7 和更新版本)大多會支援這些模式。

此外，請求必須使用存取金鑰 ID 和與 IAM 主體相關聯的私密存取金鑰來簽署。或者，您可以透過 AWS Security Token Service (AWS STS) 來產生暫時安全憑證來簽署請求。

您可以從任何網路位置呼叫這些 API 作業，但支 AWS Batch 援以資源為基礎的存取原則，其中可能包含以來源 IP 位址為基礎的限制。您也可以使用 AWS Batch 政策來控制來自特定 Amazon Virtual Private Cloud 端 (Amazon VPC) 端點或特定 VPC 的存取。實際上，這會將對特定 AWS Batch 資源的網路存取從網路內的特定 VPC 隔離出來 AWS 。

標記您的 AWS Batch 資源

為協助您管理 AWS Batch 資源，您可以用標籤形式將您自己的中繼資料指派給每個資源。本主題說明標籤並示範如何建立它們。

目錄

標籤基本概念

標籤是您指派給 AWS 資源的標籤。每個標籤皆包含由您定義的一個金鑰與一個選用值。

標籤可讓您分類 AWS 資源，例如依用途、擁有者或環境。當您有許多相同類型的資源時，您可以依據先前指派的標籤，快速識別特定的資源。例如，您可以為 AWS Batch 服務定義一組標籤，協助您追蹤每個服務的擁有者和堆疊層級。建議您為每個資源類型設計一組一致的標籤金鑰。

標籤不會自動指派給您的資源。新增標籤後，您可以隨時編輯標籤索引鍵和值，或從資源移除標籤。如果您刪除資源，也會刪除任何該資源的標籤。

標籤對 AWS Batch 來說不具有任何語意意義，並會嚴格解譯為字元字串。您可以將標籤的值設為空白字串，但您無法將標籤的值設為 Null。若您將與現有標籤具有相同鍵的標籤新增到該資源，則新值會覆寫舊值。

您可以使用 AWS Management Console、AWS CLI 和 AWS Batch API 來使用標籤。

若您使用 AWS Identity and Access Management (IAM)，您可以控制您的 AWS 帳戶中的哪些使用者具有建立、編輯和刪除標籤的許可。

標記您的 資源

您可以標記新的或現有的AWS Batch計算環境、工作、工作定義、工作佇列和排程原則。

如果您使用 AWS Batch 主控台，您可以在新資源建立時將標籤套用到新資源，或隨時在相關資源頁面上使用 Tags (標籤) 索引標籤，將標籤套用到現有的資源。

如果您使用的是 AWS Batch API、AWS CLI 或 AWS 開發套件，您可以在相關 API 動作上使用 `tags` 參數，將標籤套用到新資源，或使用 `TagResource` API 動作，將標籤套用到現有的資源。如需詳細資訊，請參閱TagResource。

有些資源建立動作可讓您在建立資源時指定資源的標籤。如果無法在資源建立時套用標籤，則資源建立程序會失敗。這可確保您要在建立時標記的資源是以指定的標籤建立，不然就根本不會建立。如果您在建立時標記資源，則不需要在建立資源之後執行自訂標記指令碼。

下表說明可標記的 AWS Batch 資源，以及可在建立時標記的資源。

AWS Batch 資源的標記支援

資源	支援標籤	支援標籤傳播	支援在建立時標記 (AWS Batch API、AWS CLI、AWS 開發套件)
AWS Batch運算環境	是	沒有 計算環境標籤不會傳播到任何其他資源。資源的標籤會在CreateComputeEnvironmentAPI 作業中傳遞的 ComputeResources 物件的標籤成員中指定。	是
AWS Batch 工作	是	是	是
AWS Batch工作定義	是	否	是
AWS Batch 任務佇列	是	否	是
AWS Batch排程原則	是	否	是

標籤限制

以下基本限制適用於標籤：

- 每一資源最多標籤數 – 50

- 對於每一個資源，每個標籤金鑰必須是唯一的，且每個標籤金鑰只能有一個值。
- 索引鍵長度上限 - 128 個 UTF-8 Unicode 字元
- 值的長度上限 - 256 個 UTF-8 Unicode 字元
- 如果您的標記結構描述用於多個 AWS 服務和資源，請記得，其他服務可能限制允許的字元。通常允許的字元包括：可用 UTF-8 表示的英文字母、數字和空格，還有以下字元：+ - = . _ : / @。
- 標籤鍵與值皆區分大小寫。
- 請勿使用 `aws:`、`AWS:` 或其任何大小寫組合做為索引鍵或值的字首，因為這已預留給 AWS 使用。您不可編輯或刪除具此字首的標籤金鑰或值。具有此前置字元的標籤不會計入您的 tags-per-resource 限制。

透過主控台使用標籤

使用主AWS Batch控台，您可以管理與新的或現有運算環境、工作、工作定義和工作佇列相關聯的標籤。

在建立個別資源時新增標籤

您可以在建立AWS Batch運算環境、工作、工作定義、工作佇列和排程原則時新增標籤。

在個別資源上新增和刪除標籤

AWS Batch 可讓您直接從資源的頁面新增或刪除與叢集相關聯的標籤。

在個別資源上新增或刪除標籤

1. 請在以下位置開啟AWS Batch主控台。 https://console.aws.amazon.com/batch/
2. 在導覽列中，選擇要使用的「區域」。
3. 在導覽窗格中，選擇資源類型 (例如，Job 佇列)。
4. 選擇特定資源，然後選擇 「編輯標籤」。
5. 視需要新增或刪除標籤。

    - 若要新增標籤 — 在清單結尾的空白文字方塊中指定鍵和值。
    - 若要刪除標籤，請選擇標籤旁邊的

      Delete icon
      按鈕。

6. 針對您要新增或刪除的每個標籤重複此程序，然後選擇 [編輯標籤] 以完成。

透過 CLI 或 API 使用標籤

使用下列 AWS CLI 命令或 AWS Batch API 操作來新增、更新、列出及刪除資源的標籤。

AWS Batch 資源的標記支援

任務	API 動作	AWS CLI	AWS Tools for Windows PowerShell
新增或覆寫一或多個標籤。	TagResource	tag-resource	添加蝙蝠 ResourceTag
刪除一或多個標籤。	UntagResource	untag-resource	刪除蝙蝠 ResourceTag
列出資源的標籤	ListTagsForResource	list-tags-for-resource	獲取蝙蝠 ResourceTag

下列範例示範如何使用 AWS CLI 來標記或取消標記資源。

範例 1：標記現有資源

以下命令會標記現有的資源。

```
aws batch tag-resource --resource-arn resource_ARN --tags team=devs
```

範例 2：取消標記現有的資源

以下命令會從現有的資源刪除標籤。

```
aws batch untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

範例 3：列出資源的標籤

以下命令列出與現有資源相關聯的標籤。

```
aws batch list-tags-for-resource --resource-arn resource_ARN
```

有些資源建立動作可讓您在建立資源時指定標籤。下列動作支援在建立時新增標籤。

任務	API 動作	AWS CLI	AWS Tools for Windows PowerShell
建立運算環境	CreateComputeEnvironment	create-compute-environment	新蝙蝠 ComputeEnvironment
建立工作佇列	CreateJobQueue	create-job-queue	新蝙蝠 JobQueue
建立排程原則	CreateSchedulingPolicy	create-scheduling-policy	新蝙蝠 SchedulingPolicy
註冊工作定義	RegisterJobDefinition	register-job-definition	寄存器蝙蝠 JobDefinition
提交任務	SubmitJob	提交工作	提交蝙蝠工作

AWS Batch Service Quotas

下表提供無法變更的AWS Batch服務配額。每個配額都是區域特定的。

資源	配額
任務佇列的數量上限 如需詳細資訊，請參閱Job 佇列。	50
跨 Amazon ECS 和 Amazon EKS 運算環境的最大數量。如需詳細資訊，請參閱運算環境。	50
每個 Amazon EKS 叢集的運算環境數目上限。	5
每個工作佇列的運算環境數目上限	3
任務相依性上限數量	20
最大任務定義大小 (適用於 `RegisterJobDefinition` API 操作)	24 KiB
最大任務承載大小 (適用於 `SubmitJob` API 操作)	30 KiB
陣列任務的最大陣列大小	10000
`SUBMITTED` 狀態任務的最大數量	1000000
每個作業帳戶每秒的最大交易數 (TPS) `SubmitJob`	50

根據您的使用方式AWS Batch，可能會套用額外的配額。若要了解有關 Amazon EC2 配額的資 Service Quotas，請參閱中的 AWS 一般參考. 如需有關 Amazon ECS 配額的詳細資訊，請參閱中的 Amazon ECS Service Quotas。AWS 一般參考如需 Amazon EKS 配額的詳細資 Service Quotas，請參閱. AWS 一般參考

疑難排 AWS Batch

您可能需要疑難排解與計算環境、工作佇列、工作定義或工作相關的問題。本章說明如何疑難排解及解決 AWS Batch 環境中的此類問題。

AWS Batch 使用 IAM 政策、角色和許可，並在 Amazon EC2、Amazon ECS 和亞馬 Amazon Elastic Kubernetes Service 基礎設施上執行。 AWS Fargate若要疑難排解與這些服務相關的問題，請參閱下列內容：

- IAM 使用者指南中的 IAM 疑難排解
- Amazon 彈性容器服務開發人員指南中的 Amazon ECS 故障排除
- Amazon EKS 用戶指南中的 Amazon EKS 故障排除
- 疑難排解 Amazon EC2 使用者指南中的 EC2 執行個體

內容

    - 無效的Kubernetes名稱
    - 刪除的運算環境
    - 節點未加入 Amazon EKS 叢集
- AWS Batch 在 Amazon EKS 上的工作卡在狀態 RUNNABLE
- 確認已aws-auth ConfigMap正確設定
- RBAC 權限或綁定未正確配置

AWS Batch

INVALID運算環境

您可能已錯誤地設定受管理的運算環境。如果這樣做，運算環境會進入INVALID狀態，而且無法接受放置的工作。下列各節說明可能的原因，以及如何根據原因進行疑難排解。

角色名稱或 ARN 不正確

運算環境進入INVALID狀態的最常見原因是 AWS Batch 服務角色或 Amazon EC2 競價型叢集角色的名稱或 Amazon 資源名稱 (ARN) 不正確。這在使用 AWS CLI 或 AWS SDK 建立的運算環境中較為常見。當您在中建立運算環境時 AWS Management Console，可 AWS Batch 協助您選擇正確的服務或 Spot 叢集角色。但是，假設您手動輸入名稱或 ARN 並輸入錯誤。然後，產生的計算環境也是INVALID。

不過，假設您在 AWS CLI 命令或 SDK 程式碼中手動輸入 IAM 資源的名稱或 ARN。在這種情況下，AWS Batch 無法驗證字符串。相反， AWS Batch 必須接受壞值並嘗試創建環境。如果 AWS Batch 無法建立環境，環境會移至某個INVALID狀態，並且您會看到下列錯誤。

如為無效的服務角色：

```
CLIENT_ERROR - Not authorized to perform sts:AssumeRole (Service:
AWSSecurityTokenService; Status Code: 403; Error Code: AccessDenied;
Request ID: dc0e2d28-2e99-11e7-b372-7fcc6fb65fe7)
```

如為無效的 Spot Fleet 角色：

```
CLIENT_ERROR - Parameter: SpotFleetRequestConfig.IamFleetRole
is invalid. (Service: AmazonEC2; Status Code: 400; Error Code:
InvalidSpotFleetRequestConfig; Request ID: 331205f0-5ae3-4cea-
```

bac4-897769639f8d) Parameter: SpotFleetRequestConfig.IamFleetRole is invalid

這個問題的一個常見原因是下列情況。您只會在使用 AWS CLI 或 AWS 軟體開發套件時指定 IAM 角色的名稱，而不是完整的 Amazon 資源名稱 (ARN)。根據您建立角色的方式而定，ARN 可能包含aws-service-role路徑前置詞。例如，如果您使用中的程序手動建立 AWS Batch 服務角色使用服務連結角色 AWS Batch，則服務角色 ARN 可能如下所示。

```
arn:aws:iam::123456789012:role/AWSBatchServiceRole
```

不過，如果您今天建立服務角色做為主控台第一次執行精靈的一部分，您的服務角色 ARN 可能如下所示。

```
arn:aws:iam::123456789012:role/aws-service-role/AWSBatchServiceRole
```

如果您將服務 AWS Batch 層級原則 (AWSBatchServiceRole) 附加至非服務角色，也可能會發生這個問題。例如，在這個案例中，您可能會收到類似下列的錯誤訊息：

```
CLIENT_ERROR - User: arn:aws:sts::account_number:assumed-role/batch-replacement-role/
aws-batch is not
     authorized to perform: action on resource ...
```

若要解決此問題，請執行下列其中一個動作。

- 建立 AWS Batch 計算環境時，請為服務角色使用空字串。
- 以下列格式指定服務角色：arn:aws:iam::*account_number*:role/aws-service-role/batch.amazonaws.com/AWSServiceRoleForBatch

當您僅在使用 AWS CLI 或 AWS SDK 時指定 IAM 角色的名稱時， AWS Batch 假設您的 ARN 不使用aws-service-role路徑前綴。因此，建議您在建立運算環境時為 IAM 角色指定完整 ARN。

若要修復設定錯誤的運算環境，請參閱修復運INVALID算環境。

修復運INVALID算環境

當您的運算環境處於某個INVALID狀態時，請更新該環境以修復無效的參數。對於角色名稱或 ARN 不正確，請使用正確的服務角色更新計算環境。

修復設定錯誤的運算環境

1. 開啟主 AWS Batch 控台，網址為 https://console.aws.amazon.com/batch/。
2. 從導覽列中，選取 AWS 區域 要使用的。
3. 在導覽窗格中，選擇 Compute environments (運算環境)。
4. 在 Compute environments (運算環境) 頁面，選擇要編輯的運算環境旁的選項按鈕，然後選擇 Edit (編輯)。
5. 在 [更新運算環境] 頁面上，對於服務角色，選擇要搭配運算環境使用的 IAM 角色。 AWS Batch 主控台只會顯示與運算環境有正確信任關係的角色。
6. 選擇 Save (儲存)，更新運算環境。

工作停留在某個RUNNABLE狀態

假設您的計算環境包含運算資源，但您的工作進度不超過RUNNABLE狀態。然後，很可能會阻止工作被放置在計算資源上，並導致您的工作佇列被阻止。以下說明如何知道您的工作是否正在等待輪流或卡住並封鎖佇列。

如果 AWS Batch 偵測到頭部有RUNNABLE工作並封鎖佇列，您將收到來自 Amazon E CloudWatch vents 的封鎖工作佇列事件，其原因是。作為ListJobs和 DescribeJobs API 調用的一部分，也將相同的原因更新到statusReason字段中。

或者，您可以透過CreateJobQueue和 UpdateJobQueueAPI 動作來設定jobStateTimeLimitActions參數。

> **Note**
>
> 目前，您唯一可以使用的動作jobStateLimitActions.action是取消工作。

此jobStateTimeLimitActions參數可用來指定在特定狀態下的工作上 AWS Batch 執行的一組動作。您可以透過欄位設定時間閾值 (以秒為單maxTimeSeconds位)。

當工作處於已定義的RUNNABLE狀態時statusReason， AWS Batch 會在經過之後執行指定maxTimeSeconds的動作。

例如，您可以將jobStateTimeLimitActions參數設定為等待RUNNABLE狀態中正在等待足夠容量變為可用的任何工作，最多等待 4 小時。您可以在取消工作之前statusReason將其設

定maxTimeSeconds為CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY和 144000，並允許下一個工作前進到工作佇列的頭部來執行此操作。

以下是偵測到工作佇列遭到封鎖時所 AWS Batch 提供的原因。此清單提供從ListJobs和 DescribeJobs API 動作傳回的訊息。這些值也與您可以為參數定義的相同jobStateLimitActions.statusReason值。

1. 原因：所有連接的運算環境都有容量不足錯誤。系統提出要求時， AWS Batch 會偵測出容量不足錯誤的 Amazon EC2 執行個體。手動取消工作或將jobStateTimeLimitActions參數設定為開啟statusReason，可讓後續工作移至佇列的標頭。

   - **statusReason**工作卡住時的消息：CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY - Service cannot fulfill the capacity requested for instance type [instanceTypeName]
   - **reason**用於**jobStateTimeLimitActions**：CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY
   - **statusReason**取消作業後的消息：Canceled by JobStateTimeLimit action due to reason: CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY

   請注意：

   a. AWS Batch 服務角色需要此偵測的autoscaling:DescribeScalingActivities權限才能運作。如果您使用AWSServiceRoleForBatch服務連結角色 (SLR) 或AWSBatchServiceRolePolicy受管理的原則，則不需要採取任何動作，因為其權限原則已更新。

   b. 如果您使用 SLR 或受管理的策略，則必須新增autoscaling:DescribeScalingActivities和ec2:DescribeSpotFleetRequestHistory權限，以便在RUNNABLE中接收封鎖的工作佇列事件和更新的工作狀態。此外， AWS Batch 需要這些權限才能透過jobStateTimeLimitActions參數執行cancellation動作，即使這些動作是在工作佇列中設定的。

   c. 在多節點 parallel (MNP) 任務的情況下，如果連接的高優先順序 Amazon EC2 運算環境發insufficient capacity生錯誤，即使優先順序較低的運算環境確實遇到此錯誤，它也會封鎖佇列。

2. 原因：所有運算環境的maxvCpus參數都小於工作需求。手動取消工作或將jobStateTimeLimitActions參數設定為開啟statusReason，可讓後續工作移至佇列的標頭。您也可以選擇增加主要運算環境的maxvCpus參數，以符合封鎖工作的需求。

- **statusReason**工作卡住時的消息：MISCONFIGURATION:COMPUTE_ENVIRONMENT_MAX_RESOURCE - CE(s) associated with the job queue cannot meet the CPU requirement of the job.
- **reason**用於**jobStateTimeLimitActions**：MISCONFIGURATION:COMPUTE_ENVIRONMENT_MAX_RESOURCE
- **statusReason**取消作業後的消息：Canceled by JobStateTimeLimit action due to reason: MISCONFIGURATION:COMPUTE_ENVIRONMENT_MAX_RESOURCE

3. 原因：所有運算環境都沒有符合工作需求的執行個體。當工作要求資源時，AWS Batch 偵測到沒有連接的計算環境能夠容納傳入的工作。手動取消工作或將jobStateTimeLimitActions參數設定為開啟statusReason，可讓後續工作移至佇列的標頭。或者，您可以重新定義運算環境允許的執行個體類型，以新增必要的工作資源。
   - **statusReason**工作卡住時的消息： MISCONFIGURATION:JOB_RESOURCE_REQUIREMENT - The job resource requirement (vCPU/memory/GPU) is higher than that can be met by the CE(s) attached to the job queue.
   - **reason**用於**jobStateTimeLimitActions**：MISCONFIGURATION:JOB_RESOURCE_REQUIREMENT
   - **statusReason**取消作業後的消息：Canceled by JobStateTimeLimit action due to reason: MISCONFIGURATION:JOB_RESOURCE_REQUIREMENT
4. 原因：所有計算環境都有服務角色問題。若要解決此問題，請將您的服務角色權限與AWS Batch 受管理服務角色權限進行比較，並解決任何差距。

   最佳做法是將 AWS Batch SLR 用於運算環境，以避免類似的錯誤。

   手動取消工作或將jobStateTimeLimitActions參數設定為開啟statusReason，可讓後續工作移至佇列的標頭。如果不解決服務角色問題，下一個工作也可能也會遭到封鎖。最好是手動調查並解決此問題。
   - **statusReason**工作卡住時的消息：MISCONFIGURATION:SERVICE_ROLE_PERMISSIONS – Batch service role has a permission issue.
   - **reason**用於**jobStateTimeLimitActions**：MISCONFIGURATION:SERVICE_ROLE_PERMISSIONS
   - **statusReason**取消作業後的消息：Canceled by JobStateTimeLimit action due to reason: MISCONFIGURATION:SERVICE_ROLE_PERMISSIONS
5. 原因：所有運算環境都無效。如需詳細資訊，請參閱INVALID計算環境。注意：您無法透過jobStateTimeLimitActions參數設定可程式化動作來解決此錯誤。

- **statusReason**工作卡住時的消息：`ACTION_REQUIRED - CE(s) associated with the job queue are invalid.`

6. 原因： AWS Batch 偵測到封鎖的佇列，但無法判斷原因。注意：您無法透過`jobStateTimeLimitActions`參數設定可程式化動作來解決此錯誤。如需疑難排解的詳細資訊，請參閱 Re: POST 中為什麼我的 AWS Batch 工作卡在 RUNNABLE AWS 中。
   - **statusReason**工作卡住時的消息：`UNDETERMINED - Batch job is blocked, root cause is undetermined.`

如果您沒有收到來自 E CloudWatch vents 的事件，或者您收到了未知的原因事件，以下是造成此問題的一些常見原因。

未在計算資源上設定**awslogs**記錄驅動程式

AWS Batch 工作會將其記錄資訊傳送至 CloudWatch 記錄檔。若要啟用此功能，您必須設定運算資源使用 `awslogs` 日誌驅動程式。假設您將運算資源 AMI 基於 Amazon ECS 優化 AMI（或 Amazon Linux）。然後，默認情況下，該驅動程序與`ecs-init`軟件包註冊。現在假設您使用不同的基礎 AMI。然後，您必須在啟動 Amazon ECS 容器代理程式時，透過`ECS_AVAILABLE_LOGGING_DRIVERS`環境變數確認日誌驅動程式是否已指定為可用的日誌驅動程式。`awslogs`如需詳細資訊，請參閱 運算資源 AMI 規格 及 建立計算資源 AMI。

資源不足

如果您的工作定義指定的 CPU 或記憶體資源多於運算資源所能配置的資源，則您的任務永遠不會放置。例如，假設您的工作指定了 4 GiB 的記憶體，而您的計算資源少於可用的記憶體。然後，工作無法放置在這些計算資源上。在此情況下，您必須減少任務定義中所指定的記憶體，或在環境中加入更多運算資源。部分記憶體會保留給 Amazon ECS 容器代理程式和其他關鍵系統程序使用。如需詳細資訊，請參閱 運算資源記憶體管理。

運算資源無法存取網際網路

運算資源需要存取，才可以與 Amazon ECS 服務端點通訊。可透過介面 VPC 端點或透過具備公有 IP 地址的運算資源來實現。

如需介面 VPC 端點的詳細資訊，請參閱 Amazon Elastic Container Service 開發人員指南中的 Amazon ECS 介面 VPC 端點 (AWS PrivateLink)。

如果您沒有設定介面 VPC 端點，且運算資源沒有公有 IP 地址，則它們必須使用網路地址轉譯 (NAT) 來提供此存取。如需詳細資訊，請參閱 Amazon VPC 使用者指南中的 NAT 閘道。如需詳細資訊，請參閱 the section called “建立 VPC”。

達到 Amazon EC2 實例限制

您的帳戶可以在其中啟動的 Amazon EC2 執行個體數量取決 AWS 區域 於您的 EC2 執行個體配額。某些執行個體類型也有 per-instance-type配額。如需有關帳戶 Amazon EC2 執行個體配額的詳細資訊，包括如何請求提高限制，請參閱 Amazon EC2 使用者指南中的 Amazon EC2 服務限制。

未安裝 Amazon ECS 容器代理

Amazon ECS 容器代理程式必須安裝在 Amazon 機器映像 (AMI) 上，才能 AWS Batch 執行任務。Amazon ECS 容器代理程式預設會安裝在 Amazon ECS 最佳化的 AMI 上。如需 Amazon ECS 容器代理程式的詳細資訊，請參閱 Amazon 彈性容器服務開發人員指南中的 Amazon ECS 容器代理程式。

如需詳細資訊，請參閱為什麼我的 AWS Batch 工作停滯在RUNNABLE狀態？ 在 Re：帖子。

建立時未標記競價型執行個體

自 2017 年 10 月 25 日起，支援 AWS Batch 運算資源的競價型執行個體標記。之前，Amazon EC2 Spot 叢集角色建議的 IAM 受管政策 (`AmazonEC2SpotFleetRole`) 未包含在啟動時標記 Spot 執行個體的許可。新建議的 IAM 受管政策稱為`AmazonEC2SpotFleetTaggingRole`。它支援在啟動時標記 Spot 執行個體。

若要在建立時修正競價型執行個體標記，請遵循下列程序，將目前建議的 IAM 受管政策套用至 Amazon EC2 Spot 叢集角色。如此一來，任何 future 使用該角色建立的 Spot 執行個體都有權在建立執行個體標籤時套用。

將目前的 IAM 受管政策套用至您的 Amazon EC2 競價型叢集角色

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇角色，然後選擇您的 Amazon EC2 競價型叢集角色。
3. 選擇連接政策。
4. 選擇亞馬遜 EC2 SpotFleet TaggingRole 並選擇附加政策。
5. 再次選擇您的 Amazon EC2 競價型叢集角色以移除先前的政策。
6. 選取 AmazonEC2 SpotFleet 角色政策右側的 x，然後選擇「分離」。

競價型執行個體未縮小

AWS Batch 於 2021 年 3 月 10 日推出AWSServiceRoleForBatch服務連結角色。如果未在計算環境的`serviceRole`參數中指定角色，則會使用此服務連結角色作為服務角色。但是，假設服務連結角色用於 EC2 競價型運算環境，但使用的 Spot 角色不包括 AmazonEC2 SpotFleet TaggingRole 受管政策。然後，競價型執行個體就不會縮小。因此，您將收到錯誤訊息，並顯示下列訊息：「您無權執行此作業。」 使用下列步驟來更新您在`spotIamFleetRole`參數中使用的 Spot 叢集角色。如需詳細資訊，請參閱 IAM 使用者指南中的使用服務連結角色和建立將權限委派給 AWS 服務的角色。

主題

- 將 AmazonEC2 SpotFleet TaggingRole 受管政策附加到您的競價型叢集角色 AWS Management Console
- 將 AmazonEC2 SpotFleet TaggingRole 受管政策附加到您的競價型叢集角色 AWS CLI

將 AmazonEC2 SpotFleet TaggingRole 受管政策附加到您的競價型叢集角色 AWS Management Console

將目前的 IAM 受管政策套用至您的 Amazon EC2 競價型叢集角色

1. 前往 https://console.aws.amazon.com/iam/ 開啟 IAM 主控台。
2. 選擇角色，然後選擇您的 Amazon EC2 競價型叢集角色。
3. 選擇連接政策。
4. 選擇亞馬遜 EC2 SpotFleet TaggingRole 並選擇附加政策。
5. 再次選擇您的 Amazon EC2 競價型叢集角色以移除先前的政策。
6. 選取 AmazonEC2 SpotFleet 角色政策右側的 x，然後選擇「分離」。

將 AmazonEC2 SpotFleet TaggingRole 受管政策附加到您的競價型叢集角色 AWS CLI

這些命令範例假設您的 Amazon EC2 競價型叢集角色命名為「亞馬*# EC2 SpotFleet* 角色」。如果您的角色使用不同的名稱，請調整要相符的指令。

將 AmazonEC2 SpotFleet TaggingRole 受管政策附加到您的競價型叢集角色

1. 若要將亞馬遜 EC2 SpotFleet TaggingRole 受管身分與存取權管理政策附加到您的 *AmazonEC2 SpotFleet #*色，請使用. AWS CLI

```
$ aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEC2SpotFleetTaggingRole
 \
    --role-name AmazonEC2SpotFleetRole
```

2. 若要將 AmazonEC2 SpotFleet 角色受管 IAM 政策與您的 *AmazonEC2 SpotFleet ##*分離出來，請使用. AWS CLI

```
$ aws iam detach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEC2SpotFleetRole \
    --role-name AmazonEC2SpotFleetRole
```

無法擷取 Secrets Manager 密碼

如果您將 AMI 與早於 1.16.0-1 版本的 Amazon ECS 代理程式搭配使用，則必須使用 Amazon ECS 代理程式組態變數ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true才能使用此功能。您可以在建立該執行個體時，將其新增至./etc/ecs/ecs.config檔案至新的容器執行個體。或者，您也可以將其新增至現有的執行個體。如果將其新增至現有的執行個體，則必須在新增 ECS 代理程式之後重新啟動該代理程式。如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的 Amazon ECS 容器代理程式組態。

無法覆寫工作定義資源需求

傳遞至SubmitJob的容器覆寫結構memory和vcpus成員中指定的記憶體和 vCPU 覆寫無法覆寫工作定義的資源需求結構中指定的記憶體和 vCPU 需求。

如果您嘗試覆寫這些資源需求，您可能會看到下列錯誤訊息：

「此值是以棄用的鍵提交的，並且可能與工作定義的資源需求提供的值衝突。」

若要更正此問題，請在容器覆寫的資源需求成員中指定記憶體和 vCPU 需求。例如，如果您的記憶體和 vCPU 覆寫已在下列幾行中指定。

```
"containerOverrides": {
   "memory": 8192,
   "vcpus": 4
}
```

將它們更改為以下內容：

```
"containerOverrides": {
   "resourceRequirements": [
      {
         "type": "MEMORY",
         "value": "8192"
      },
      {
         "type": "VCPU",
         "value": "4"
      }
   ],
}
```

對工作定義的容器屬性物件中指定的記憶體和 vCPU 需求進行相同的變更。例如，如果在以下幾行中指定了您的記憶體和 vCPU 需求。

```
{
   "containerProperties": {
      "memory": 4096,
      "vcpus": 2,
}
```

將它們更改為以下內容：

```
"containerProperties": {
   "resourceRequirements": [
      {
         "type": "MEMORY",
         "value": "4096"
      },
      {
         "type": "VCPU",
         "value": "2"
      }
   ],
}
```

更新desiredvCpus設定時出現錯誤訊息

使用 AWS Batch API 更新所需的 vCPUs (desiredvCpus) 設定時，您會看到下列錯誤訊息。

```
Manually scaling down compute environment is not supported. Disconnecting
job queues from compute environment will cause it to scale-down to
minvCpus.
```

如果更新的值小於目前desiredvCpusdesiredvCpus值，就會發生這個問題。當您更新desiredvCpus值時，下列兩項都必須成立：

- desiredvCpus值必須介於minvCpus和maxvCpus值之間。
- 更新後的desiredvCpus值必須大於或等於目前desiredvCpus值。

AWS Batch 在 Amazon EKS 上

主題

INVALID運算環境

您可能已錯誤地設定受管理的運算環境。如果這樣做，運算環境會進入INVALID狀態，而且無法接受放置的工作。下列各節說明可能的原因，以及如何根據原因進行疑難排解。

不支援Kubernetes版本

當您使用 API 作業或 CreateComputeEnvironment API 作業建立或UpdateComputeEnvironment更新運算環境時，您可能會看到類似下列的錯誤訊息。如果您在中指定不支援的Kubernetes版本，就會發生這個問題EC2Configuration。

```
At least one imageKubernetesVersion in EC2Configuration is not supported.
```

若要解決此問題，請刪除計算環境，然後使用支援的Kubernetes版本重新建立該環境。

您可以在 Amazon EKS 叢集上執行次要版本升級。例如，即使不支援次要版本，您1.yy也可以1.xx將叢集從升級為。

不過，在主要版本更新INVALID之後，運算環境狀態可能會變更為。例如，如果您執行從1.xx到的主要版本升級2.yy。如果主要版本不受支援 AWS Batch，您會看到類似下列的錯誤訊息。

```
reason=CLIENT_ERROR - ... EKS Cluster version [2.yy] is unsupported
```

若要解決此問題，請在使用 API 作業建立或更新運算環境時指定支援的Kubernetes版本。

AWS Batch 在 Amazon EKS 上目前支持以下Kubernetes版本：

- 1.29
- 1.28
- 1.27
- 1.26
- 1.25
- 1.24
- 1.23

執行個體設定檔不存在

如果指定的執行個體設定檔不存在，Amazon EKS AWS Batch 上的運算環境狀態會變更為INVALID。您會在statusReason參數中看到類似下列的錯誤設定。

```
CLIENT_ERROR - Instance profile arn:aws:iam::...:instance-profile/<name> does not exist
```

若要解決此問題，請指定或建立使用中執行個體設定檔。如需詳細資訊，請參閱《Amazon EKS 使用者指南》中的 Amazon EKS 節點 IAM 角色。

無效的Kubernetes名稱

如果 AWS Batch 在 Amazon EKS 上無法驗證運算環境的命名空間，則運算環境狀態會變更為INVALID。例如，如果命名空間不存在，則可能會發生此問題。

您會在statusReason參數中看到類似下列的錯誤訊息集。

```
CLIENT_ERROR - Unable to validate Kubernetes Namespace
```

如果下列任一條件成立，就可能會發生這個問題：

- CreateComputeEnvironment呼叫中的Kubernetes命名空間字串不存在。如需詳細資訊，請參閱CreateCompute環境。
- 管理命名空間所需的角色型存取控制 (RBAC) 權限未正確設定。
- AWS Batch 無法存取 Amazon EKS Kubernetes API 伺服器端點。

若要解決此問題，請參閱 確認已aws-auth ConfigMap正確設定。如需詳細資訊，請參閱開始使 AWS Batch 用 Amazon EKS。

刪除的運算環境

假設您先刪除 Amazon EKS 叢集，然後再刪除 Amazon EKS 運算環境 AWS Batch 上的連接。然後，計算環境狀態會變更為INVALID。在這個案例中，如果您重新建立具有相同名稱的 Amazon EKS 叢集，則運算環境無法正常運作。

若要解決此問題，請刪除 Amazon EKS AWS Batch 上的運算環境，然後重新建立。

節點未加入 Amazon EKS 叢集

AWS Batch 在 Amazon EKS 上，如果確定並非所有節點都加入 Amazon EKS 叢集，則會縮減運算環境的規模。 AWS Batch 在 Amazon EKS 上縮減運算環境時，運算環境狀態會變更為INVALID。

> **Note**
>
> AWS Batch 不會立即變更運算環境狀態，以便您可以偵錯問題。

您會在statusReason參數中看到類似下列項目的錯誤訊息集：

```
Your compute environment has been INVALIDATED and scaled down because
none of the instances joined the underlying ECS Cluster. Common issues
preventing instances joining are the following: VPC/Subnet configuration
preventing communication to ECS, incorrect Instance Profile policy
preventing authorization to ECS, or customized AMI or LaunchTemplate
configurations affecting ECS agent.
```

```
Your compute environment has been INVALIDATED and scaled down because
none of the nodes joined the underlying Amazon EKS Cluster. Common issues
preventing nodes joining are the following: networking configuration
preventing communication to Amazon EKS Cluster, incorrect Amazon EKS
```

```
Instance Profile or Kubernetes RBAC policy preventing authorization
to Amazon EKS Cluster, customized AMI or LaunchTemplate configurations
affecting Amazon EKS/Kubernetes node bootstrap.
```

使用預設的 Amazon EKS AMI 時，造成此問題的最常見原因如下：

- 執行個體角色未正確設定。如需詳細資訊，請參閱《Amazon EKS 使用者指南》中的 Amazon EKS 節點 IAM 角色。
- 子網路設定不正確。如需詳細資訊，請參閱 Amazon EKS VPC 和子網路需求和 Amazon EKS 使用者指南中的考量事項。
- 未正確設定安全性群組。如需詳細資訊，請參閱 Amazon EKS 安全群組要求和注意事項 (在 Amazon EKS 使用者指南中)。

> **Note**
>
> 您也可能會在 Personal Health Dashboard (PHD) 中看到錯誤通知。

AWS Batch 在 Amazon EKS 上的工作卡在狀態 **RUNNABLE**

當您使aws-authConfigMap用建立受管節點群組或節點群組時，會自動建立並套用至叢集eksctl。aws-authConfigMap一開始會建立允許節點加入叢集。不過，您也可以使用新增aws-authConfigMap以角色為基礎的存取控制 (RBAC) 存取權限給使用者和角色。

若要確認已aws-authConfigMap正確設定：

1. 擷取下列項目中的對應角色 aws-authConfigMap：

   ```
   $ kubectl get configmap -n kube-system aws-auth -o yaml
   ```

2. 確認roleARN是否設定如下。

   ```
   rolearn: arn:aws:iam::aws_account_number:role/AWSServiceRoleForBatch
   ```

   > **Note**
   >
   > 您也可以檢閱 Amazon EKS 控制平面日誌。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的 Amazon EKS 控制平面記錄。

若要解決某個工作停留在某個RUNNABLE狀態的問題，建議您使用重新套kubectl用資訊清單。如需詳細資訊，請參閱 步驟 1：準備您的 Amazon EKS 叢集 AWS Batch。或者，您可以使kubectl用手動編輯 aws-authConfigMap. 如需詳細資訊，請參閱 Amazon EKS 使用者指南中的啟用對叢集的 IAM 使用者和角色存取。

確認已aws-auth ConfigMap正確設定

若要確認已aws-authConfigMap正確設定：

1. 擷取中的對應角色aws-authConfigMap。

```
$ kubectl get configmap -n kube-system aws-auth -o yaml
```

2. 確認roleARN是否設定如下。

```
rolearn: arn:aws:iam::aws_account_number:role/AWSServiceRoleForBatch
```

> **Note**
>
> 路徑aws-service-role/batch.amazonaws.com/已從服務連結角色的 ARN 中移除。這是因為aws-auth配置對映有問題。如需詳細資訊，請參閱當路徑包含在中的 ARN 中時，具有路徑的角色無法運作aws-authconfigmap。

> **Note**
>
> 您也可以檢閱 Amazon EKS 控制平面日誌。如需詳細資訊，請參閱 Amazon EKS 使用者指南中的 Amazon EKS 控制平面記錄。

若要解決某個工作停留在某個RUNNABLE狀態的問題，建議您使用重新套kubectl用資訊清單。如需詳細資訊，請參閱 步驟 1：準備您的 Amazon EKS 叢集 AWS Batch。或者，您可以使kubectl用手動編輯 aws-authConfigMap. 如需詳細資訊，請參閱 Amazon EKS 使用者指南中的啟用對叢集的 IAM 使用者和角色存取。

RBAC 權限或綁定未正確配置

如果您遇到任何 RBAC 權限或繫結問題，請確認aws-batchKubernetes角色可以存取命名空間：Kubernetes

```
$ kubectl get namespace namespace --as=aws-batch
```

```
$ kubectl auth can-i get ns --as=aws-batch
```

您也可以使用命**kubectl describe**令來檢視叢集角色或Kubernetes命名空間的授權。

```
$ kubectl describe clusterrole aws-batch-cluster-role
```

下列為範例輸出。

```
Name:         aws-batch-cluster-role
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources                                       Non-Resource URLs  Resource Names
 Verbs
  ---------                                       -----------------  --------------
 -----
  configmaps                                      []                 []
 [get list watch]
  nodes                                           []                 []
 [get list watch]
  pods                                            []                 []
 [get list watch]
  daemonsets.apps                                 []                 []
 [get list watch]
  deployments.apps                                []                 []
 [get list watch]
  replicasets.apps                                []                 []
 [get list watch]
  statefulsets.apps                               []                 []
 [get list watch]
  clusterrolebindings.rbac.authorization.k8s.io   []                 []
 [get list]
  clusterroles.rbac.authorization.k8s.io          []                 []
 [get list]
  namespaces                                      []                 []
 [get]
```

```
$ kubectl describe role aws-batch-compute-environment-role -n my-aws-batch-namespace
```

下列為範例輸出。

```
Name:         aws-batch-compute-environment-role
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources                                  Non-Resource URLs  Resource Names  Verbs
  ---------                                  -----------------  --------------  -----
  pods                                       []                 []              [create
 get list watch delete patch]
  serviceaccounts                            []                 []              [get list]
  rolebindings.rbac.authorization.k8s.io     []                 []              [get list]
  roles.rbac.authorization.k8s.io            []                 []              [get list]
```

若要解決這個問題，請重新套用 RBAC 權限和命令。rolebinding如需更多詳細資訊，請參閱 步驟 1：準備您的 Amazon EKS 叢集 AWS Batch。

AWS Batch 最佳實務

您可以使用AWS Batch大規模執行各種高要求的運算工作負載，而無需管理複雜的架構。 AWS Batch工作可用於流行病學、遊戲和機器學習等領域的各種使用案例。

本主題涵蓋使用時應考慮的最佳做法，以AWS Batch及如何在使用時執行和最佳化工作負載的指導AWS Batch。

主題

何時使用 AWS Batch

AWS Batch以低成本大規模執行工作，並提供佇列服務和成本最佳化的擴充能力。但是，並非每個工作負載都適合使用AWS Batch.

- 短工作 — 如果工作只執行幾秒鐘，則排定批次工作的額外負荷可能會比工作本身的執行時間更長。因應措施是在binpack您提交任務之前一起工作AWS Batch。然後，配置您的任AWS Batch務以迭代任務。例如，將個別任務引數暫存到 Amazon DynamoDB 資料表中，或當做 Amazon S3 儲存貯體中的檔案。請考慮將工作分組，以便每個工作執行 3-5 分鐘。完成工作binpack之後，在工AWS Batch作中循環瀏覽任務群組。
- 必須立即執行的工作 — AWS Batch 可以快速處理工作。不過，AWS Batch這是排程器，可針對成本效能、工作優先順序和輸送量進行最佳化。 AWS Batch可能需要時間來處理您的請求。如果您在幾秒鐘內需要回應，那麼使用 Amazon ECS 或 Amazon EKS 的服務型方法更適合。

大規模執行的檢查清單

在 5 萬個或更多 vCPUs 上執行大量工作負載之前，請考慮下列檢查清單。

> **Note**
>
> 如果您計劃在一百萬個或更多 vCPUs 上執行大型工作負載，或需要大規模執行的指導，請聯絡您的AWS團隊。

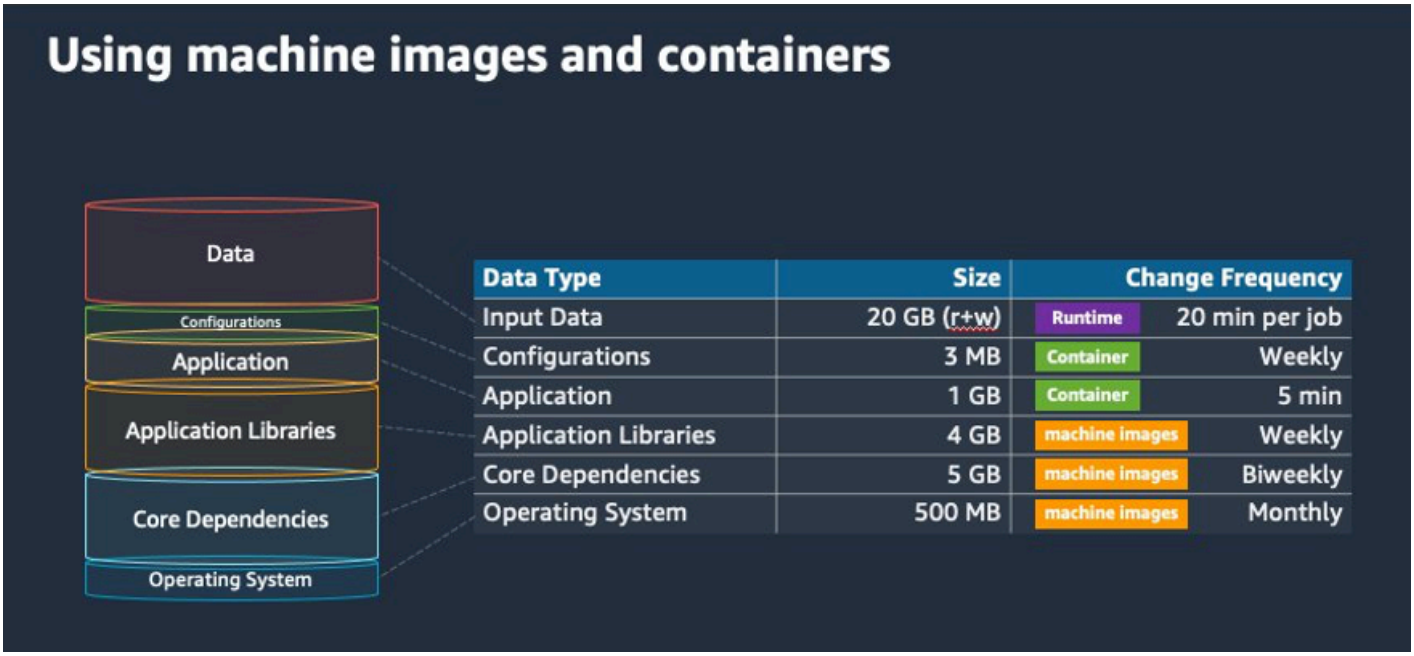
- 檢查您的 Amazon EC2 配額 — 在的 Service Quotas 面板中查看您的 Amazon EC2 配額 (也稱為限制) AWS Management Console。如有必要，請求增加 Amazon EC2 執行個體尖峰數量的配額。請記住，Amazon EC2 競價型和 Amazon 隨需執行個體有不同的配額 如需詳細資訊，請參閱開始使用 Service Quotas。
- 驗證每個區域的 Amazon 彈性區塊存放區配額 — 每個執行個體使用 GP2 或 GP3 磁碟區作為作業系統。根據預設，每個配額AWS 區域為 300 TiB。不過，每個執行個體都會使用計數做為此配額的一部分。因此，當您驗證每個區域的 Amazon 彈性區塊存放區配額時，請務必考慮這一點。如果達到配額，就無法建立更多執行個體。如需詳細資訊，請參閱 Amazon 彈性區塊存放區端點和配額
- 使用 Amazon S3 進行儲存 — Amazon S3 提供高輸送量，並有助於根據每個可用區域中的任務和執行個體數量來佈建多少儲存量的猜測。如需詳細資訊，請參閱最佳實務設計模式：最佳化 Amazon S3 效能。
- 逐步擴展以及早找出瓶頸 — 對於在一百萬個或更多 vCPUs 上執行的工作，請從降低開始並逐漸增加，以便及早找出瓶頸。例如，首先在 5 萬個 vCPUs 上執行。然後，將計數增加到 20 萬個 vCPUs，然後增加 50 萬個 vCPUs，依此類推。換句話說，繼續逐漸增加 vCPU 計數，直到達到所需的 vCPUs 數量為止。
- 監控以及早識別潛在問題 — 若要避免大規模執行時可能發生的中斷和問題，請務必同時監控應用程式和架構。即使從 1 千個 vCPUs 擴展到 5 千個 vCPU，也可能會發生中斷。您可以使用 Amazon CloudWatch Logs 檢閱日誌資料，或使用用戶端程式庫使用 CloudWatch 內嵌指標。如需詳細資訊，請參閱CloudWatch 記錄代理程式參考和 `aws-embedded-metrics`

最佳化容器和 AMI

容器大小和結構對於您執行的第一組工作很重要。如果容器大於 4 GB，則尤其如此。容器圖像是內置的層。該層由 Docker 使用三個並發線程並行檢索。您可以使用參數增加並發線程的`max-concurrent-downloads`數量。如需詳細資訊，請參閱 D ockerd 文件。

雖然您可以使用較大的容器，但我們建議您最佳化容器結構和大小，以加快啟動時間。

- 較小的容器獲取速度更快 — 更小的容器可以帶來更快的應用程序啟動時間。若要減少容器大小，請將不常更新的程式庫或檔案卸載至 Amazon 機器映像 (AMI)。您也可以使用 bind 掛載來存取您的容器。如需詳細資訊，請參閱繫結裝載。
- 創建尺寸均勻的圖層並分解大圖層-每個圖層由一個線程檢索。因此，較大的圖層可能會對您的工作啟動時間產生重大影響。我們建議最大圖層大小為 2 GB，作為較大容器大小和更快的啟動時間之間的良好權衡。您可以執行指`docker history your_image_id`令來檢查容器映像檔結構和圖層大小。如需詳細資訊，請參閱 Docker 文件。
- 使用 Amazon Elastic 容器登錄做為您的容器儲存庫 — 當您同時執 parallel 數千個任務時，自我管理的儲存庫可能會失敗或限制輸送量。Amazon ECR 可大規模運作，並可處理具有多達一百萬個 vCPUs 的工作負載。



選擇正確的運算環境資源

AWS Fargate與 Amazon EC2 相比，所需的初始設定和組態更少，而且可能更容易使用，尤其是在您第一次使用時。搭配使用 Fargate，您無需管理伺服器、處理容量規劃，或出於安全性而隔離容器工作負載。

如果您有下列需求，建議您使用 Fargate 執行個體：

- 您的工作必須快速啟動，特別是不到 30 秒。
- 您的任務需求為 16 個 vCPUs 以下、沒有 GPU，以及 120 GiB 的記憶體 (含) 以下。

如需詳細資訊，請參閱何時使用 Fargate。

如果您有下列需求，建議您使用 Amazon EC2 執行個體：

- 您需要增強對執行個體選取的控制，或需要使用特定的執行個體類型。
- 您的任務需要AWS Fargate無法提供的資源，例如 GPU、更多記憶體、自訂 AMI 或 Amazon 彈性網狀架構配接器。
- 您需要高層級的輸送量或並行。
- 您需要自訂 AMI、Amazon EC2 啟動範本，或存取特殊的 Linux 參數。

使用 Amazon EC2，您可以根據特定需求更精細地調整工作負載，並在需要時大規模執行。

Amazon EC2 按需或 Amazon EC2 現貨

大多數AWS Batch客戶使用 Amazon EC2 Spot 執行個體是因為比隨需執行個體節省成本。但是，如果您的工作負載執行數小時且無法中斷，則隨需執行個體可能更適合您。您始終可以先試用 Spot 執行個體，並在必要時切換到隨需。

如果您有以下要求和期望，請使用 Amazon EC2 隨需執行個體：

- 工作的執行時間超過一小時，您無法容忍工作負載中斷。
- 您的整體工作負載有嚴格的 SLO (服務層級目標)，而且無法增加運算時間。
- 您需要的執行個體更有可能發生中斷情況。

如果您有下列需求和期望，請使用 Amazon EC2 競價型執行個體：

- 工作的執行階段通常為 30 分鐘以內。
- 作為工作負載的一部分，您可以容忍潛在的中斷和工作重新排程。有關詳情，請參閱 Spot 執行個體建議程式。
- 如果中斷，可以從檢查點重新啟動長時間執行的作業。

您可以先在 Spot 執行個體上提交，然後使用隨需執行個體做為後援選項，以混合兩種購買模式。例如，在連接到 Amazon EC2 Spot 執行個體上執行之運算環境的佇列上提交任務。如果任務中斷，請從 Amazon catch 取事件，並將其 EventBridge 與競價型執行個體回收產生關聯。然後，使用AWS Lambda函數或AWS Step Functions將工作重新提交至隨選佇列。如需詳細資訊教學：針對失敗的 Job

務事件傳送 Amazon 簡易通知服務警示，請參閱處理 Amazon EC2 Spot 執行個體中斷的最佳實務和 AWS Batch使用 Step Functions 進行管理。

> **Important**
>
> 針對隨需運算環境使用不同的執行個體類型、大小和可用區域，以維持 Amazon EC2 Spot 執行個體集區的可用性並降低中斷率。

使用 Amazon EC2 競價最佳實務 AWS Batch

當您選擇 Amazon Elastic Compute Cloud (EC2) 競價型執行個體時，您可能可以優化工作流程以節省成本，有時候可以顯著節省成本。如需詳細資訊，請參閱 Amazon EC2 競價型的最佳實務。

若要最佳化您的工作流程以節省成本，請考慮下列 Amazon EC2 Spot 最佳實務AWS Batch：

- AWS Batch選擇**SPOT_CAPACITY_OPTIMIZED**配置策略 — 從最深的 Amazon EC2 競價型容量集區中選擇 Amazon EC2 執行個體。如果您擔心中斷，這是一個合適的選擇。如需詳細資訊，請參閱分配策略。
- 多樣化執行個體類型 — 為了使您的執行個體類型多樣化，請考慮相容的大小和系列，然後根據價格或可用性進行AWS Batch選擇。例如，請考慮`c5.24xlarge`做為`c5.12xlarge`或`c5a`、、`c5n`和`c5dm5`、和`m5d`族群的替代品。如需詳細資訊，請參閱彈性瞭解執行個體類型和可用區域。
- 減少任務執行時間或檢查點 — 我們建議您不要在使用 Amazon EC2 Spot 執行個體時執行需要一小時或更長時間的任務，以避免中斷。如果你劃分或檢查你的工作成由 30 分鐘或更少的小部分，你可以顯著減少中斷的可能性。
- 使用自動重試 — 為避免工作中斷，請為AWS Batch工作設定自動重試。Batch 工作可能因下列任一原因而中斷：傳回非零結束代碼、發生服務錯誤或執行個體回收。您最多可以設定 10 次自動重試。首先，我們建議您至少設定 1-3 次自動重試。如需追蹤 Amazon EC2 競價型中斷的相關資訊，請參閱競價型中斷儀表板。

  對於AWS Batch，如果您設定了 retry 參數，工作會放置在工作佇列的前面。也就是說，工作被優先考慮。當您建立工作定義或在中提交工作時AWS CLI，您可以設定重試策略。如需詳細資訊，請參閱提交工作。

```
$ aws batch submit-job --job-name MyJob \
    --job-queue MyJQ \
    --job-definition MyJD \
```

```
    --retry-strategy attempts=2
```

- 使用自訂重試 — 您可以為特定應用程式結束代碼或執行個體回收設定作業重試策略。在下列範例中，如果主機導致失敗，則最多可重試五次工作。但是，如果工作因其他原因而失敗，則工作會結束並將狀態設定為FAILED。

```
"retryStrategy": {
    "attempts": 5,
    "evaluateOnExit":
    [{
        "onStatusReason" :"Host EC2*",
        "action": "RETRY"
    },{
      "onReason" : "*"
        "action": "EXIT"
    }]
}
```

- 使用 Spot 中斷儀表板 — 您可以使用 Spot 中斷儀表板追蹤 Spot 中斷。該應用程式提供回收的 Amazon EC2 Spot 執行個體上的指標，以及 Spot 執行個體所在的可用區域。如需詳細資訊，請參閱 Spot 中斷儀表板

常見錯誤和疑難排解

中的錯誤AWS Batch通常發生在應用程序級別，或者是由不符合特定作業需求的實例配置引起的。其他問題包括工作停留在RUNNABLE狀態中，或是運算環境卡在某個狀INVALID態中。如需疑難排解工作停滯RUNNABLE狀態的詳細資訊，請參閱工作停留在某個RUNNABLE狀態。如需疑難排解INVALID狀態中運算環境的相關資訊，請參閱INVALID運算環境。

- 檢查 Amazon EC2 競價型 vCPU 配額 — 確認您目前的服務配額符合任務需求。例如，假設您目前的服務配額為 256 個 vCPUs，而該工作需要 10,000 個 vCPUs。然後，服務配額不符合工作要求。如需詳細資訊和疑難排解指示，請參閱 Amazon EC2 服務配額和如何增加 Amazon EC2Resources 的服務配額？ 。
- 工作會在應用程式執行前失敗 — 有些工作可能會因為DockerTimeoutError錯誤或錯CannotPullContainerError誤而失敗。如需疑難排解資訊，請參閱如何解決中的 DockerTimeoutError "" 錯誤AWS Batch？ 。
- IP 位址不足 — VPC 和子網路中的 IP 位址數量可能會限制您可以建立的執行個體數目。使用無類別網域間路由 (CIDR) 提供比執行工作負載所需的更多 IP 位址。如有必要，您也可以建立具有大型位址空間的專用 VPC。例如，您可以在中建立具有多個 CIDR 的 VPC，10.x.0.0/16並在每個可用

區域中建立一個子網路 (CIDR 為)。10.x.y.0/17在這個例子中，x 介於 1-4 之間，y 是 0 或 128。此組態在每個子網路中提供 36,000 個 IP 位址。

- 確認執行個體已向 Amazon EC2 註冊 — 如果您在 Amazon EC2 主控台中看到您的執行個體，但 Amazon ECS 叢集中沒有 Amazon 彈性容器服務容器執行個體，則 Amazon ECS 代理程式可能不會安裝在 Amazon 機器映像 (AMI) 上。亞馬遜 ECS 代理程式、AMI 中的 Amazon EC2 資料或啟動範本可能也未正確設定。若要隔離根本原因，請建立個別的 Amazon EC2 執行個體，或使用 SSH 連接到現有的執行個體。如需詳細資訊，請參閱 Amazon ECS 容器代理程式組態、Amazon ECS 日誌檔案位置和。計算資源 AMI
- 檢閱AWS儀表板 — 檢閱AWS儀表板以確認預期的工作狀態，以及計算環境是否如預期擴展。您也可以檢閱中的工作記錄 CloudWatch。

- 確認您的執行個體已建立 — 如果已建立執行個體，表示您的運算環境會如預期擴展。如果您的執行個體並未建立，請在您的運算環境中尋找要變更的相關子網路。如需詳細資訊，請參閱驗證 Auto Scaling 群組的縮放比例活動。

  我們也建議您驗證執行個體是否能滿足您的相關工作要求。例如，工作可能需要 1 TiB 的記憶體，但計算環境使用的是限制為 192 GB 記憶體的 C5 執行個體類型。
- 確認您的執行個體已被請求 AWS Batch — 檢查 Auto Scaling 群組歷史記錄以確認您的執行個體是否被請求AWS Batch。這表示 Amazon EC2 如何嘗試取得執行個體。如果您收到錯誤訊息，指出 Amazon EC2 Spot 無法取得特定可用區域中的執行個體，這可能是因為可用區域未提供特定的執行個體系列。
- 確認執行個體是否向 Amazon ECS 註冊 — 如果您在 Amazon EC2 主控台中看到執行個體，但 Amazon ECS 叢集中沒有 Amazon ECS 容器執行個體，則 Amazon ECS 代理程式可能未安裝在 Amazon 機器映像 (AMI) 上。此外，亞馬遜 ECS 代理程式、AMI 中的 Amazon EC2 資料或啟動範本可能未正確設定。若要隔離根本原因，請建立個別的 Amazon EC2 執行個體，或使用 SSH 連接到現有的執行個體。如需詳細資訊，請參閱CloudWatch 代理程式組態檔案：日誌區段、Amazon ECS 日誌檔位置和計算資源 AMI。
- 開立支援票證 — 如果您在進行一些疑難排解後仍然遇到問題，並有支援方案，請開立支援票證。在支援票證中，請務必包含有關問題、工作負載細節、組態和測試結果的資訊。如需詳細資訊，請參閱比較AWS Support方案。
- 檢閱AWS Batch和 HPC 論壇 — 如需詳細資訊，請參閱AWS Batch和 HPC 論壇。
- 檢閱AWS Batch執行階段監控儀表板 — 此儀表板使用無伺服器架構從 Amazon ECS 和 Amazon EC2 擷取事件AWS Batch，以提供任務和執行個體的見解。如需詳細資訊，請參閱AWS Batch執行階段監視儀表板解決

文件歷史紀錄

下表說明文件自初始發行版本以來的重要變更 AWS Batch。我們也會經常更新文件，以處理您傳送給我們的意見回饋。

變更	描述	日期
更新了 AWS Batch 支持的 Amazon EKS 版本	更新了 AWS Batch 支援移除 1.22 版的 Amazon EKS 版本。	2024年3月11日
更新了 AWS Batch 支持的 Amazon EKS 版本	更新了 AWS Batch 支援包含 1.29 版本的 Amazon EKS 版本。	2024 年 2 月 29 日
自動化工作重試	更正了代碼示例。	2024 年 2 月 29 日
添加對多容器作業的支持 AWS Batch	為 Amazon 彈性容器服務、亞馬遜彈性 Kubernetes 服務和添加 AWS Batch 對多容器任務的支援。 AWS Fargate	2024年2月28日
更新了 AWS Batch 支持的 Amazon EKS 版本	更新了 AWS Batch 支援包含 1.28 版本的 Amazon EKS 版本	2024年1月27日
已更新BatchServiceRolePolicy 和 AWSBatchServiceRole	BatchServiceRolePolicy 已更新以新增描述 Spot 叢集請求歷史記錄和 Amazon EC2 Auto Scaling 活動的支援。 AWSBatchServiceRole 已更新為新增陳述式 ID、授`ec2:DescribeSpotFleetRequestHistory` 與與 AWS Batch 權限`autoscali`	2023 年 12 月 5 日

	ng:DescribeScaling Activities 。	
AWS Batch 在 Amazon EKS 上	AWS Batch 新增對在 Amazon EKS 叢集上執行任務的支援。	2022 年 10 月 25 日
跨服務混淆副預防 AWS Batch	AWS Batch 現在提供混淆的副安全性問題的因應措施，當不同實體強制執行動作時，就會產生此問題。	2022 年 6 月 6 日
介面 VPC 端端點 ()AWS PrivateLink	增加了對配置由 AWS PrivateLink支持的接口 VPC 端點的支持。這表示您可以在 VPC 之間建立私人連線， AWS Batch 而不需要透過 NAT 執行個體、VPN 連線或 AWS Direct Connect存取。	2022 年 4 月 15 日
增強的計算環境更新	AWS Batch 增強運算環境的支援更新。	2022 年 4 月 14 日
AWS 受管策略更新-更新現有策略	AWS Batch 更新現有的受管理策略。	2021 年 12 月 6 日
公平分享排程	AWS Batch 新增將排程原則新增至工作佇列的支援。	2021 年 11 月 9 日
Amazon EFS	AWS Batch 新增將 Amazon EFS 檔案系統新增至您的任務定義的支援。	2021 年 4 月 1 日
新增服務連結角色	AWS Batch 新增AWSServiceRoleForBatch服務連結角色。	2021 年 3 月 10 日
AWS Fargate 支持	AWS Batch 添加了對在 Fargate 資源上運行作業的支持。	2020 年 12 月 3 日

Amazon 2 支持	AWS Batch 增加了對使用 EC2 組態參數在運算環境中自動選擇 Amazon Linux 2 AMI 的支援。	2020 年 11 月 24 日
增強的重試策略	AWS Batch 增強工作的重試策略。現在，工作可以重試或停止進一步的重試，方法是將工作`StatusReason` 的`ExitCodeReason`、或與模式相符。	2020 年 10 月 20 日
資源標記	AWS Batch 新增支援將中繼資料標籤新增至您的運算環境、工作定義、工作佇列和工作。	2020 年 10 月 7 日
秘密	AWS Batch 增加了對將秘密傳遞給工作的支持。	2020 年 10 月 1 日
日誌	AWS Batch 添加了為作業指定其他日誌驅動程序的支持。	2020 年 10 月 1 日
分配策略	AWS Batch 添加了對選擇實例類型的多種策略的支持。	2019 年 10 月 16 日
全民福利局支援	AWS Batch 增加了對 Elastic Fabric Adapter (EFA) 備的支持。	2019 年 8 月 2 日
GPU 排程	AWS Batch 添加 GPU 調度。使用此功能，您可以指定每個任務需要的 GPU 數量，並相應地 AWS Batch 擴展執行個體。	2019 年 4 月 4 日

多節點 parallel 工作	AWS Batch 增加了對多節點 parallel 作業的支持。您可以使用此功能執行跨越多個 Amazon EC2 執行個體的單一任務。	2018 年 11 月 19 日
資源層級許可	AWS Batch 支援數個 API 作業的資源層級權限。	2018 年 11 月 12 日
Amazon EC2 啟動模板支持	AWS Batch 新增對搭配運算環境使用啟動範本的支援。	2018 年 11 月 12 日
AWS Batch 工作逾時	AWS Batch 添加對作業超時的支持。有了這項支援，您可以為工作設定特定的逾時持續時間，如果工作執行的時間超過預期，就 AWS Batch 會終止工作。	2018 年 4 月 5 日
AWS Batch 作為 EventBridge 目標的工作	AWS Batch 工作可作為 EventBridge 目標使用。通過創建簡單的規則，您可以匹配事件並提交 AWS Batch 作業以響應它們。	2018 年 3 月 1 日
CloudTrail 稽核 AWS Batch	CloudTrail 可以稽核對 AWS Batch API 動作的呼叫。	2018 年 1 月 10 日
陣列工作	AWS Batch 增加了對陣列工作的支援。您可以將陣列工作用於參數掃描和蒙地卡羅工作負載。	2017 年 11 月 28 日
擴充 AWS Batch 標記	AWS Batch 擴展對標記功能的支持。您可以使用此函數為在受管運算環境中啟動的 Amazon EC2 Spot 執行個體指定標籤。	2017 年 10 月 26 日

AWS Batch 事件串流 EventBridge	AWS Batch 會加入的事件串流 EventBridge。您可以使用 AWS Batch 事件串流接收有關提交至工作佇列之工作狀態的近乎即時的通知。	2017 年 10 月 24 日
自動化工作重試	AWS Batch 添加了對作業重試的支持。透過此更新，您可以將重試策略套用至工作和工作定義，以便在工作失敗時自動重試。	2017 年 3 月 28 日
AWS Batch 一般可用性	AWS Batch 引入，旨在讓您在 AWS 雲端.	2017 年 1 月 5 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。