AWS 白皮書

最佳實務設計模式：最佳化 Amazon S3 效能

最佳實務設計模式：最佳化 Amazon S3 效能: AWS 白皮書

Table of Contents

最佳實務設計模式：最佳化 Amazon S3 效能

首次發佈日期：2019 年 6 月 (文件修訂)

摘要

當建置從 Amazon S3 上傳及擷取儲存體的應用程式時，請依照 AWS 最佳實務建置來最佳化效能。AWS 也提供了更詳細的效能設計模式。

簡介

您的應用程式從 Amazon S3 上傳和擷取儲存體時，可輕鬆達到請求中每秒數千筆交易的效能。Amazon S3 會自動調高請求率。例如，您的應用程式可在儲存貯體中達成每個字首每秒至少 3,500 個 PUT/COPY/POST/DELETE 與 5,500 個 GET/HEAD 請求。在儲存貯體中的字首數不受限制。您可以並行讀取以提升您的讀取或寫入的效能例如，如果您在 Amazon S3 儲存貯體裡建立 10 個字首，平行讀取，您可以縮放讀取效能至每秒 55,000 讀取要求。

而 Amazon S3 上的部分資料湖應用程式在查詢超過數 PB 資料時，能掃描數百萬或數十億個物件。這些資料湖應用程式獲得單一執行個體傳輸速率，讓 Amazon EC2 執行個體充分利用網路界面，在單一執行個體上最高達到 100 Gb/s。然後，這些應用程式會彙總多個執行個體之間的傳輸量，以取得每秒多個 TB。

其他應用程式為對延遲敏感的應用程式，例如社交媒體簡訊應用程式。這些應用程式可以獲得一致的小型物件延遲 (以及適用於更大型物件的第一個位元組輸出延遲)，大約 100–200 毫秒。

其他 AWS 服務也可以協助加速不同應用程式架構的效能。例如，如果您想要單一 HTTP 連線有更高的傳輸速率或低於十毫秒的延遲，請使用 Amazon CloudFront 或 Amazon ElastiCache，以搭配 Amazon S3 進行快取。

此外，若您想要在相距遙遠的用戶端與 S3 儲存貯體之間快速傳輸資料，請使用 Amazon S3 Transfer Acceleration。Transfer Acceleration 使用 CloudFront 中遍佈全球的節點，加速地理距離的資料傳輸。

若您的 Amazon S3 工作負載使用含 AWS Key Management Service (SSE-KMS) 的伺服器端加密，請參閱《AWS Key Management Service 開發人員指南中的AWS KMS 限制，以取得使用案例所支援之請求率的資訊。

下列主題針對使用 Amazon S3 的應用程式，說明最佳化效能的最佳實務指導方針和設計模式。

此指導方針優先於先前有關 Amazon S3 效能最佳化的任何指導方針。例如，先前 Amazon S3 效能指導方針建議以雜湊字元將字首命名隨機化，以最佳化經常擷取資料時的效能。現在您不再需要為了效能隨機命名字首，而可以日期型序列模式，為字首命名。如需 Amazon S3 效能最佳化的最新資訊，請參閱效能準則與效能設計模式。

Amazon S3 的效能準則

為了讓您在 Amazon S3 上的應用程式獲得最佳效能，AWS 建議下列準則。

主題

測量效能

當最佳化效能時，請查看網路輸送量、CPU，以及動態隨機存取記憶體 (DRAM) 要求。根據這些不同資源的混合需求，可能值得評估不同的 Amazon EC2 執行個體類型。如需執行個體類型的詳細資訊，請參閱《Amazon EC2 Linux 執行個體使用者指南》中的執行個體類型。

測量效能時，使用 HTTP 分析工具來查看 DNS 查閱時間、延遲和資料傳輸速度也很有幫助。

水平擴展儲存體連線

跨許多連線傳播請求是常用的設計模式，藉此水平擴展效能。建置高效能應用程式時，請將 Amazon S3 視為一個非常大的分散式系統，而不是像傳統儲存伺服器那樣的單一網路端點。您可以向 Amazon S3 發出多個並行請求，以達到最佳效能。將這些請求分散到不同的連線，以從 Amazon S3 獲得最大可存取的頻寬。Amazon S3 對儲存貯體的連線數沒有任何限制。

使用位元組範圍擷取

使用 GET Object 請求中的 Range HTTP 標頭，您可以從物件中擷取位元組範圍，僅傳輸指定的部分。您可以對 Amazon S3 使用並行連線，從同一物件內擷取不同的位元組範圍。與單一整個物件請求相比，這可協助您實現更高的彙總傳輸量。擷取大型物件的更小範圍也可讓您的應用程式在請求中斷時改善重試時間。如需詳細資訊，請參閱取得物件。

位元組範圍請求的一般大小為 8 MB 或 16 MB。如果物件是使用多部件上傳的 PUT，則良好實務為以相同部件大小 (或至少與部件界限一致) GET 它們，以取得最佳效能。GET 請求可以直接解決個別組件；例如：`GET ?partNumber=N`。

對延遲敏感之應用程式的重試請求

積極的逾時和重試可協助推動一致的延遲。由於 Amazon S3 的範圍很大，如果第一個請求很慢，則重試的請求可能會採取不同路徑且很快成功。AWS 軟體開發套件具有可設定的逾時和重試值，您可以將它們調整為特定應用程式的容錯值。

在同一 AWS 區域結合 Amazon S3 (儲存) 與 Amazon EC2 (運算)

雖然 S3 儲存貯體名稱是全球唯一，但每個儲存貯體都存放在您建立儲存貯體時選取的區域中。為了最佳化效能，建議您盡可能從同一 AWS 區域的 Amazon EC2 執行個體存取儲存貯體。這可協助減少網路延遲和資料傳輸成本。

如需資料傳輸成本的詳細資訊，請參閱 Amazon S3 定價。

使用 Amazon S3 Transfer Acceleration 將距離產生的延遲最小化

Amazon S3 Transfer Acceleration 可讓用戶端與 S3 儲存貯體之間長地理距離的檔案傳輸變得迅速、簡單又安全。Transfer Acceleration 善用 Amazon CloudFront 中遍佈全球的節點。資料到達節點時會經由最佳化的網路路徑而路由至 Amazon S3。Transfer Acceleration 非常適合在各大洲定期傳輸數 GB 到數 TB 的資料。也有助於從世界各地上傳至集中型儲存貯體的用戶端。

您可以使用 Amazon S3 Transfer Acceleration 速度比較工具，比較各 Amazon S3 區域的加速與非加速上傳速度。速度比較工具在使用和不使用 Amazon S3 Transfer Acceleration 的情況下，透過分段上傳，將檔案從您的瀏覽器傳送至各個 Amazon S3 區域。

使用 AWS 軟體開發套件的最新版本

AWS 開發套件內建支援許多建議的 Amazon S3 效能最佳化指導方針。這些開發套件提供更簡單的 API 以方便從應用程式內善用 Amazon S3，並定期更新來遵循最新的最佳實務。例如，軟體開發套件包括在發生 HTTP 503 錯誤時自動重試請求的邏輯，並投資於程式碼以回應並適應慢速連線。

這些開發套件也提供 Transfer Manager，在適當情況下會使用位元組範圍請求，以自動水平擴展連線，達到每秒數千個請求。請務必使用最新版的 AWS 軟體開發套件，來取得最新的效能最佳化特性。

使用 HTTP REST API 請求時，您也可以最佳化效能。使用 REST API 時，您應該遵循屬於軟體開發套件的相同最佳實務。允許逾時和重試慢速請求，並有多個連線允許平行擷取物件資料。如需有關使用 REST API 的詳細資訊，請參閱 Amazon Simple Storage Service API 參考。

Amazon S3 的效能設計模式

當要將應用程式設計成從 Amazon S3 上傳及擷取儲存體時，請使用我們的最佳實務設計模式，讓您的應用程式獲得最佳效能。我們也提供了《效能準則》，供您考量應用程式架構規劃。

若要最佳化效能，您可以使用下列設計模式。

主題

- 對經常存取的內容使用快取
- 適用於對延遲敏感之應用程式的逾時和重試
- 高傳輸量的水平擴展與請求並行化
- 使用 Amazon S3 Transfer Acceleration 加速異地資料傳輸

對經常存取的內容使用快取

許多將資料存放在 Amazon S3 的應用程式會提供資料「工作集」，供使用者重複請求。如果工作負載對一組常用的物件傳送重複的 GET 請求，您可以使用快取，例如 Amazon CloudFront、Amazon ElastiCache 或 AWS Elemental MediaStore，以最佳化效能。成功採用快取可以產生低延遲和高資料傳輸率。使用快取的應用程式傳送至 Amazon S3 的直接請求也較少，有助於減少請求成本。

Amazon CloudFront 是快速的內容交付網路 (CDN)，在分散各地的大量連接點 (PoP) 之中，可直接快取來自 Amazon S3 的資料。在可能從多個區域或透過網際網路存取物件時，CloudFront 允許在存取物件的使用者附近快取資料。這樣能夠高效能傳遞熱門的 Amazon S3 內容。如需有關 CloudFront 的資訊，請參閱 Amazon CloudFront 開發人員指南。

Amazon ElastiCache 受管的記憶體內快取。ElastiCache 可讓您佈建將物件快取在記憶體中的 Amazon EC2 執行個體。此快取會導致 GET 延遲數量級減少，以及下載傳輸量顯著增加。若要使用 ElastiCache，請修改應用程式邏輯，將熱門物件移入快取，而在向 Amazon S3 請求熱門物件之前，先檢查快取中有無這些物件。有關使用 ElastiCache 來提高 Amazon S3 GET 效能的範例，請參閱部落格文章：使用 Amazon ElastiCache for Redis 讓 Amazon S3 效能飛快。

AWS Elemental MediaStore 是專為來自 Amazon S3 的影片工作流程和媒體交付而打造的快取和內容分發系統。MediaStore 特別針對影片提供端對端儲存 API，且建議用於需要高效能的影片工作負載。如需有關 MediaStore 的資訊，請參閱 AWS Elemental MediaStore 使用者指南。

適用於對延遲敏感之應用程式的逾時和重試

在某些情況下，應用程式會收到 Amazon S3 的回應，這表示有必要重試。Amazon S3 會將儲存貯體和物件名稱對應至相關聯的物件資料。若應用程式產生高請求率 (通常對少數物件維持每秒超過 5,000 個請求的速率)，則其可能會收到 HTTP 503 slowdown 回應。如果發生這些錯誤，每個 AWS 軟體開發套件會使用指數退避來實作自動重試邏輯。如果您不是使用 AWS 軟體開發套件，則應該在收到 HTTP 503 錯誤時實作重試邏輯。如需有關退避技術的資訊，請參閱《Amazon Web Services 一般參考》中的 AWS 中的錯誤重試與指數退避。

Amazon S3 會隨著持續的新請求率而自動擴展，動態地最佳化效能。當 Amazon S3 在內部針對新請求率最佳化時，您將會暫時收到 HTTP 503 請求回應，直到最佳化完成為止。在 Amazon S3 於內部針對新請求率來最佳化效能之後，通常就能處理所有請求而不必重試。

對於需要低延遲的應用程式，Amazon S3 建議追蹤並積極重試較慢的操作。當您重試請求時，我們建議對 Amazon S3 使用新連線，並執行全新的 DNS 查閱。

當您進行大量易變大小的請求 (例如，超過128 MB) 時，我們建議追蹤正要實現的傳輸量並重試最慢的 5% 請求。當您提出更小的要求 (例如，少於 512 KB)，其中中位數延遲通常在幾十毫秒範圍內時，良好實務為 2 秒後重試 GET 或 PUT 操作。如果需要額外重試，最佳實務為退避。例如，我們建議 2 秒後發出一次重試，再過 4 秒後發出第二次重試。

如果您的應用程式對 Amazon S3 提出固定大小的請求，則這些請求的回應時間都應該會更一致。在此情況下，簡單策略為識別最慢的 1% 請求，然後重試它們。即使單次重試也常常有效地減少延遲。

若您是使用 AWS Key Management Service (AWS KMS) 進行伺服器端加密，請參閱《AWS Key Management Service 開發人員指南》中的配額，以取得使用案例所支援之請求率的相關資訊。

高傳輸量的水平擴展與請求並行化

Amazon S3 是非常大的分散式系統。為了協助您善用其規模，建議您水平擴展對 Amazon S3 服務端點的並行請求。除了在 Amazon S3 內分發要求，這種擴展方法還有助於將負載分散至網路上的多個路徑。

對於高傳輸量傳輸，Amazon S3 建議使用對 GET 或 PUT 資料並行使用多個連線的應用程式。例如，AWS Java 開發套件中的 Amazon S3 Transfer Manager 支援此作法，其他大多數 AWS 開發套件也提供類似的概念。對於部分應用程式，您可以實現並行連線，方法為在不同的應用程式執行緒中或在不同的應用程式執行個體中並行啟動多個請求。採取的最佳方法取決於您的應用程式，以及您正在存取的物件結構。

您可以使用 AWS 軟體開發套件，來直接發出 GET 和 PUT 請求，而非使用 AWS 軟體開發套件中管理傳輸的操作。此方法可讓您更直接調整工作負載，同時仍能受益於軟體開發套件支援重試，以及處理任何可能發生的 HTTP 503 回應。當您在區域內將大型物件從 Amazon S3 下載至 Amazon EC2 時，我們通常建議您對物件的位元組範圍提出並行請求，精細程度為 8–16 MB。針對所需網路輸送量的每個 85–90 MB/s，提出一個並行請求。若要使 10 Gb/s 網路界面卡 (NIC) 飽和，您可以透過個別連線使用大約 15 個並行請求。您可以透過更多連線來擴增並行請求，使更快的 NIC 飽和，例如 25 Gb/s 或 100 Gb/s NIC。

當您調整要同時發出的請求數時，測量效能很重要。我們建議從一次提出單一請求開始。測量要實現的網路頻寬，以及您的應用程式在處理資料時其他資源的使用情形。然後，您可以識別瓶頸資源 (亦即，具有最高用量的資源)，因此識別可能有用的請求數。例如，如果一次處理一個請求導致使用 25% CPU，則其建議最多只能容納四個並行請求。

測量是必要的操作，且隨著請求率的增加，確認資源用量是值得的。

如果您的應用程式使用 REST API 直接向 Amazon S3 發出請求，我們建議您使用 HTTP 連線集區，並針對一系列請求重複使用每個連線。避免每個請求的連線設定可移除在每個請求上執行 TCP 緩慢啟動和 Secure Sockets Layer (SSL) 交握的需求。如需使用 REST API 的資訊，請參閱 Amazon S3 REST API 簡介。

最後，值得注意 DNS，並仔細檢查請求是否分散在廣泛的 Amazon S3 IP 地址集區。Amazon S3 的 DNS 查詢會在大量 IP 端點之中循環進行。但是快取解析程式或重複使用單一 IP 地址的應用程式碼不會受益於地址多樣性和隨之而來的負載平衡。網路公用程式工具 (例如 `netstat` 命令列工具) 可以顯示用來與 Amazon S3 通訊的 IP 地址，我們提供指導方針供 DNS 組態使用。如需這些準則的詳細資訊，請參閱請求路由。

使用 Amazon S3 Transfer Acceleration 加速異地資料傳輸

在分散全球的用戶端與使用 Amazon S3 的區域應用程式之間，Amazon S3 Transfer Acceleration 可有效地將地理距離所導致的延遲降至最低或消除。Transfer Acceleration 使用 CloudFront 中遍佈全球的節點來傳輸資料。AWS 邊緣網路在超過 50 個位置中具有存在點。目前用於透過 CloudFront 分發內容，並快速回應對 Amazon Route 53 的 DNS 查詢。

邊緣網路也有助於加速進出 Amazon S3 的資料傳輸。它非常適合於各大洲之間傳輸資料的應用程式、具有快速網際網路連線的應用程式、使用大型物件的應用程式，或具有許多內容要上傳的應用程式。當資料到達節點時，資料會經由最佳化的網路路徑而路由至 Amazon S3。一般而言，離 Amazon S3 區域越遠，使用 Transfer Acceleration 改善速度越明顯。

您可以在新的或現有的儲存貯體上設定 Transfer Acceleration。您可以使用個別的 Amazon S3 Transfer Acceleration 端點來使用 AWS 節點。測試 Transfer Acceleration 是否可提升用戶端請求效能的最佳方式，就是使用 Amazon S3 Transfer Acceleration 速度比較工具。網路組態和狀況會隨著時間和位置而有所不同。因此，只有在 Amazon S3 Transfer Acceleration 有可能改善上傳效能時，才會向您收取傳輸費用。如需使用具有不同 AWS 開發套件之 Transfer Acceleration 的資訊，請參閱 Amazon S3 Transfer Acceleration 範例。

作者群

此文件的作者包括：

- Amazon S3 副總裁 Mai-Lan Tomsen Bukovec
- Amazon S3 資深首席工程師 Andy Warfield
- Amazon S3 首席工程師 Tim Harris

文件修訂

若要收到此白皮書更新的通知，請訂閱 RSS 摘要。

update-history-change	update-history-description	update-history-date
已更新	已檢閱技術準確性	2021 年 3 月 10 日
初版	初版	2019 年 6 月 1 日

聲明

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