aws

使用者指南

Amazon Linux 2023

Amazon Linux 2023: 使用者指南

Table of Contents

Amazon Linux 2023 是什麼？

Amazon Linux 2023 (AL2023) 是來自 Amazon Web Services () 的新一代 Amazon Linux AWS。透過 AL2023，您可以在安全、穩定且高效能的執行時間環境中開發和執行雲端和企業應用程式。此外，您還可以取得應用程式環境，提供長期支援，讓您存取 Linux 中的最新創新。使用 AL2023 無需負擔額外費用。

AL2023 是 Amazon Linux 2 (AL2) 的後續版本。如需 AL2023 和 AL2 之間差異的資訊，請參閱 比較 AL2 與 AL2023和 AL2023 中的套件變更。

主題

發行節奏

Amazon Linux 2023 (AL2023) 已於 2023 年 3 月發行，並支援至 2029 年 6 月 30 日。支援有兩個階段：

- 標準支援 – 在此階段，版本會每季收到次要版本更新。標準支援階段將於 2027 年 6 月 30 日結束。
- 維護 – 在此階段，版本只會收到安全性更新和重大錯誤修正。這些更新會在可用時立即發佈。維護階段將於 2029 年 6 月 30 日結束。

主要版本和次要版本

在每個 Amazon Linux 版本 (主要版本、次要版本或安全版本) 中，我們都會發布新的 Linux Amazon Machine Image (AMI)。

- 主要版本發行 — 包括新功能，以及在整個堆疊中的安全性和效能改進。這些改進可能包括對核心、工具鏈、Glib C、OpenSSL，以及任何其他系統程式庫和公用程式的重大變更。Amazon Linux 的主要版本部分是以上游 Fedora Linux 發行版本的目前版本為基礎。 AWS 可能會新增或取代其他非 Fedora 上游的特定套件。
- 次要版本 — 每季更新，包含安全更新、錯誤修正，以及新功能和套件。每個次要版本都是累積的更新清單，除了新功能和套件之外，還包括安全和錯誤修正。這些發行版本可能包含最新的語言執行期，例如 PHP。另外還可能包括其他常見的套件，如 Ansible 和 Docker。

使用新版本

更新將透過新的 Amazon Machine Image (AMI) 發行版和對應的新儲存庫組合進行。預設情況下，新的 AMI 會與其指向的儲存庫耦合。不過，您可以隨著時間經過，將執行中的 Amazon EC2 執行個體指向較新的儲存庫版本，以便在執行中的執行個體套用更新。您也可以啟動最新 AMI 的新執行個體來進行更新。

長期支援政策

Amazon Linux 會為您的所有套件提供更新，並在主要版本中維持在 Amazon Linux 上建置的應用程式相容性。核心套件 (例如 glibc 程式庫、OpenSSL、OpenSSH 和 DNF 套件管理員) 會獲得 AL2023 主要發行版本壽命週期的支援。不屬於核心套件的套件會根據其特定的上游來源獲得支援。您可以執行下列命令來查看個別套件的特定支援狀態和日期。

```
$ sudo dnf supportinfo --pkg packagename
```

您可以執行下列命令來取得所有目前已安裝套件的資訊。

```
$ sudo dnf supportinfo --show installed
```

核心套件的完整清單會在預覽期間完成。如果您想查看更多核心套件包含的套件，請告訴我們。我們在收集意見回饋時進行評估。您可以透過指定 AWS 代表或在 GitHub 上的 amazon-linux-2023 儲存庫中提出問題，來提供 AL2023 的意見回饋。 https://github.com/amazonlinux/amazon-linux-2023/issues

命名和版本控制

AL2023 在標準支援的兩年期間，每三個月提供次要版本。每個發行版本都會以 0 到 N 之間的增量來識別。0 是指該版序的原始主要發行版本。所有版本都將稱為 Amazon Linux 2023。當下一個版本的 Amazon Linux 發行時，AL2023 將輸入延長支援，並接收安全性更新和重大錯誤修正的更新。

例如，AL2023 的次要版本具有以下格式：

- `2023.0.20230301`
- `2023.1.20230601`
- `2023.2.20230901`

對應的 AL2023 AMI 格式如下：

- `al2023-ami-2023.0.20230301.0-kernel-6.1-x86_64`
- `al2023-ami-2023.1.20230601.0-kernel-6.1-x86_64`
- `al2023-ami-2023.2.20230901.0-kernel-6.1-x86_64`

在特定的次要版本中，一般 AMI 發行版本會以 AMI 發行日期的時間戳記發行。

- `al2023-ami-2023.0.20230301.0-kernel-6.1-x86_64`
- `al2023-ami-2023.0.20230410.0-kernel-6.1-x86_64`
- `al2023-ami-2023.0.20230520.0-kernel-6.1-x86_64`

用於識別 AL2 或 AL2023 執行個體的建議方法從從 讀取通用平台列舉 (CPE) 字串開始`/etc/system-release-cpe`。然後，將字串拆為字段。最後，閱讀平台和版本值。

AL2023 還導入用於平台識別的新檔案：

- `/etc/amazon-linux-release` symlinks 至 `/etc/system-release`
- `/etc/amazon-linux-release-cpe` symlinks 至 `/etc/system-release-cpe`

這兩個檔案指出執行個體是 Amazon Linux。除非您想知道特定的平台和版本值，否則無需讀取檔案或將字串拆為字段。

效能與操作最佳化

Amazon 6.1 核心

- AL2023 使用 Elastic Network Adapter (ENA) 和 Elastic Fabric Adapter (EFA) 裝置的最新驅動程式。AL2023 著重於 Amazon EC2 基礎設施中硬體的效能和功能向後移植。
- 核心即時修補程式可用於 `x86_64` 和 `aarch64` 執行個體類型。這會減少頻繁重新開機的需求。
- 所有核心建置和執行時間組態都包含許多相同的 AL2 效能和操作最佳化。

基本工具鏈選項和預設建置旗標

- AL2023 套件的建置預設會啟用編譯器最佳化 (`-O2`)
- AL2023 套件的建置 `x86-64v2` 需要 `x86-64` 系統 (`-march=x86-64-v2`)，以及 `aarch64` (`-march=armv8.2-a+crypto -mtune=neoverse-n1`) 的 Graviton 2 或更新版本。
- AL2023 套件是以啟用自動向量化 (`-ftree-vectorize`) 建置。
- AL2023 套件是在啟用連結時間最佳化 (LTO) 的情況下建置。
- AL2023 使用 Rust、Clang/LLVM 和 Go 的更新版本。

套件選項和版本

- 選擇主要系統元件的反向移植包括在 Amazon EC2 基礎設施上執行的多項效能改進，尤其是 Graviton 執行個體。
- AL2023 與數個 AWS 服務 和 功能整合。這包括 AWS CLI、SSM Agent、Amazon Kinesis Agent 和 CloudFormation。
- AL2023 使用 Amazon Corretto 作為 Java 開發套件 (JDK)。
- AL2023 為較新版本提供資料庫引擎和程式設計語言執行期更新，因為它們由上游專案發布。新版本的程式設計語言執行期會在發布時新增。

在雲端環境中部署

- 基礎 AL2023 AMI 和容器映像會經常更新，以支援更換修補執行個體。
- 核心更新包含在 AL2023 AMI 更新中。這代表您不需要使用 `yum update` 和 `reboot` 等的命令來更新核心。
- 除了標準的 AL2023 AMI 之外，還提供最小 AMI 和容器映像。選擇最小 AMI，以使用執行服務所需的最少套件數量來執行環境。

- 預設情況下，AL2023 AMI 和容器會鎖定至特定版本的套件儲存庫。啟動時沒有自動更新。這表示您可以隨時掌控導入任何套件更新的時間。在推出生產前，您可以隨時在 beta/gamma 環境中進行測試。如有問題，您可以使用預先驗證的復原路徑。

與 Fedora 的關係

AL2023 會維護自己的版本，並支援與 Fedora 無關的生命週期。AL2023 提供開放原始碼軟體的更新版本、更多種類的套件，以及頻繁的發行版本。這樣可以保留熟悉的 RPM 架構作業系統。

AL2023 的正式上市 (GA) 版本無法直接與任何特定的 Fedora 發行版本相比。AL2023 GA 版本包含來自 Fedora 34、35 和 36 的元件。有些元件與 Fedora 中的元件相同，有些元件則經過修改。其他元件更接近 CentOS Stream 9 中的元件，或獨立開發的元件。Amazon Linux 核心源自 kernel.org 上的長期支援選項，並從 Fedora 中獨立選出。

自訂 cloud-init

cloud-init 套件是一種開放原始碼應用程式，可在雲端運算環境中引導 Linux 映像。如需詳細資訊，請參閱 cloud-init 文件。

AL2023 包含 cloud-init 的自訂版本。使用 cloud-init 時，您可以指定執行個體在開機時執行的操作。

啟動執行個體時，您可以使用使用者資料欄位將動作傳遞至 cloud-init。這表示您可在許多使用案例下用常用 Amazon Machine Image (AMI)，並在啟動執行個體時進行動態設定。AL2023 也會用 cloud-init 來設定 `ec2-user` 帳號。

AL2023 會在 `/etc/cloud/cloud.cfg.d` 和 `/etc/cloud/cloud.cfg` 中使用 cloud-init 操作。您可以在 `/etc/cloud/cloud.cfg.d` 目錄中建立自己的 cloud-init 操作檔案。Cloud-init 會以字母順序讀取此目錄中的所有檔案。新檔案會覆寫舊檔案中的值。cloud-init 啟動執行個體時，cloud-init 套件會執行下列設定工作：

- 設定預設的地區設定
- 設定主機名稱
- 剖析和處理使用者資料
- 產生主機私有 SSH 金鑰
- 將使用者的公有 SSH 金鑰加入至 `.ssh/authorized_keys`，以方便登入和管理
- 準備儲存庫以管理套件

- 處理使用者資料中定義的套件動作
- 執行使用者資料中的使用者指令碼
- 掛載執行個體儲存體磁碟區 (如適用)
  - 根據預設，若 `ephemeral0` 執行個體儲存體存在，且包含有效的檔案系統，執行個體儲存體會掛載在 `/media/ephemeral0`；否則將不會掛載。
  - 根據預設，對於 `m1.small` 和 `c1.medium` 執行個體類型，將掛載任何與執行個體關聯的置換磁碟區。
  - 您可用下列的 cloud-init 指令覆寫預設的執行個體儲存體磁碟區掛載：

    ```
    #cloud-config
    mounts:
    - [ ephemeral0 ]
    ```

    如需進一步掌控掛載，請參閱 cloud-init 文件中的掛載。
  - 執行個體啟動時，不會格式化支援 TRIM 的執行個體儲存體磁碟區。掛載執行個體儲存體磁碟區前，您必須先分割和格式化執行個體儲存體磁碟區。

    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磁碟區 TRIM 支援。
  - 您可在啟動執行個體時，使用 `disk_setup` 模組分割及格式化執行個體儲存體磁碟區。

    如需詳細資訊，請參閱 cloud-init 文件中的磁碟設定。

如需使用 cloud-init 與 SELinux 搭配的相關資訊，請參閱 使用 cloud-init 來啟用 `enforcing` 模式。

如需有關 cloud-init 使用者資料格式的資訊，請參閱 cloud-init 文件中的使用者資料格式。

安全更新和功能

AL2023 提供許多安全性更新和解決方案。

主題

- 受管更新
- 雲端內部的安全
- SELinux 模式
- 合規計劃

- SSH 伺服器預設
- OpenSSL 3 的主要功能

受管更新

使用 DNF和 儲存庫版本套用安全性更新。如需詳細資訊，請參閱在 AL2023 中管理套件和作業系統更新。

雲端內部的安全

安全是 AWS 與您之間的共同責任。共同責任模型將此描述為雲端安全性和雲端安全性。如需詳細資訊，請參閱Amazon Linux 2023 的安全與合規。

SELinux 模式

預設情況下，SELinux 處於啟用狀態，並在 AL2023 中設置為寬容模式。在寬容模式中，會記錄權限遭拒的情形，但不會強制執行。

SELinux 政策定義使用者、處理程序、程式、檔案和裝置的權限。使用 SELinux 時，您可以選擇兩個政策之一。這些政策是目標式或多層級安全性 (MLS)。

如需有關 SELinux 模式和政策的詳細資訊，請參閱 設定 AL2023 的 SELinux 模式 和 SELinux Project 維基。

合規計劃

獨立稽核人員會評估 AL2023 的安全性和合規性，以及許多 AWS 合規計劃。

SSH 伺服器預設

AL2023 包括 OpenSSH 8.7。OpenSSH 8.7 預設停用 `ssh-rsa` 金鑰交換算法。如需詳細資訊，請參閱預設 SSH 伺服器組態。

OpenSSL 3 的主要功能

- 憑證管理通訊協定 (CMP，RFC 4210) 同時包含 CRMF (RFC 4211) 和 HTTP 傳輸 (RFC 6712)。
- libcrypto 的 HTTP 或 HTTPS 用戶端支援 GET 和 POST 操作、重新導向、普通和 ASN.1 編碼內容、代理和逾時。

- EVP_KDF 與金鑰衍生函數搭配使用。
- EVP_MAC API 可搭配 MACs 使用。
- Linux 核心 TLS 支援。

如需詳細資訊，請參閱《OpenSSL 遷移指南》。

網路服務

該開放原始碼專案 `systemd-networkd` 在現代 Linux 發行版中廣泛使用。該專案使用類似 `systemd` 架構其餘部分的聲明式組態語言。其主要組態檔案類型為 `.network` 和 `.link` 檔案。

`amazon-ec2-net-utils` 套件會在 `/run/systemd/network` 目錄中產生介面特定組態。這些組態會在連接至執行個體時，在介面上啟用 IPv4 和 IPv6 網路。這些組態也會安裝政策路由規則，以協助確保透過對應執行個體的網路介面，將本機來源的流量路由至網路。這些規則可確保透過彈性網路界面 (ENI) 從相關聯的地址或字首路由正確的流量。如需使用 ENI 的詳細資訊，請參閱《Amazon EC2 使用者指南》中的使用 ENI。

您可以將自訂組態檔案放入 `/etc/systemd/network` 目錄，以覆寫 `/run/systemd/network` 中包含的預設組態設定並自訂此網路行為。

systemd.network 說明文件說明 `systemd-networkd` 服務如何判斷適用於特定介面的組態。它也會為 ENI 支援的界面產生稱為 的替代名稱altnames，以反映各種 AWS 資源的屬性。這些 ENI 支援的介面屬性是 ENI 附件的 `ENI ID` 和 `DeviceIndex` 欄位。使用各種工具 (例如 `ip` 命令) 時，您可以使用這些介面的屬性來參照這些介面。

AL2023 執行個體界面名稱是使用`systemd`槽命名機制產生。如需詳細資訊，請參閱 systemd.net 命名方案。

此外，AL2023 預設使用 `fq_codel` 主動佇列管理網路傳輸排程演算法。如需詳細資訊，請參閱 CoDel 概觀。

核心工具鏈套件 glibc、gcc、binutils

Amazon Linux 的套件子集指定為核心工具鏈套件。核心套件是 AL2023 的主要部分，可獲得五年的支援。我們可能會變更套件的版本，但是，長期支援適用於 Amazon Linux 發行版中包含的套件。

這三個核心套件提供一個系統工具鏈，可用在 Amazon Linux 發行版中建置大多數軟體。

套件	定義	用途
glibc 2.34	系統 C 程式庫	獲得大多數提供標準功能的二進位程式採用，以及用於程式與核心間的介面。
gcc 11.2	gcc 編譯器套件	編譯 C、C++、Fortran。
binutils 2.35	組譯工具、連結器以及其他二進位工具	操縱或檢查二進位程式。

建議對任何 glibc 程式庫進行更新後皆需重新啟動。對於控制服務的套件更新，可能只需要重新啟動服務便可套用更新。但是，系統重新啟動可確保所有先前的套件和程式庫更新都已完成。

套件管理工具

AL2023 中的預設軟體套件管理工具是 DNF。 DNF是 的後續版本YUM，AL2 中的套件管理工具。

DNF 的用途類似 YUM。許多DNF命令和命令選項與YUM命令相同。在命令列介面 (CLI) 命令中，在大多數情況下 dnf 會取代 yum。

例如，針對下列 AL2 yum命令：

```
$ sudo yum install packagename
$ sudo yum search packagename
$ sudo yum remove packagename
```

在 AL2023 中，它們會成為下列命令：

```
$ sudo dnf install packagename
$ sudo dnf search packagename
$ sudo dnf remove packagename
```

在 AL2023 中，yum 命令仍然可用，但作為指向 dnf 命令的指標。因此，當 yum 命令在 Shell 中或指令碼中使用時，所有命令和選項都與 DNF CLI 相同。有關 YUM CLI 和 DNF CLI 之間差異的詳細資訊，請參閱 DNF CLI 與 YUM 相比的變更。

如需 dnf 命令的命令和選項完整參考，請參閱 man 頁面 man dnf。如需詳細資訊，請參閱DNF 命令參考。

預設 SSH 伺服器組態

如果您使用多年前的 SSH 用戶端，連線至執行個體時可能會看到錯誤訊息。如果錯誤告訴您找不到相符的主機金鑰類型，請更新 SSH 主機金鑰以解決此問題。

預設停用 **ssh-rsa** 簽章

AL2023 包含預設組態，可停用舊版ssh-rsa主機金鑰演算法，並產生一組減少的主機金鑰。用戶端必須支援 ssh-ed25519 或 ecdsa-sha2-nistp256 主機金鑰演算法。

預設組態接受下列任何金鑰交換演算法：

- curve25519-sha256
- curve25519-sha256@libssh.org
- ecdh-sha2-nistp256
- ecdh-sha2-nistp384
- ecdh-sha2-nistp521
- diffie-hellman-group-exchange-sha256
- diffie-hellman-group14-sha256
- diffie-hellman-group16-sha512
- diffie-hellman-group18-sha512

預設情況下，AL2023 會產生 ed25519 和 ECDSA 主機金鑰。用戶端支援 ssh-ed25519 或 ecdsa-sha2-nistp256 主機金鑰演算法。當您透過 SSH 連線至執行個體時，您必須使用支援相容演算法的用戶端，例如 ssh-ed25519 或 ecdsa-sha2-nistp256。如果您需要使用其他金鑰類型，請使用使用者資料中的 cloud-config 片段覆寫生成的金鑰清單。

在下列範例中，cloud-config 使用 ecdsa 和 ed25519 金鑰產生 rsa 主機金鑰。

```
#cloud-config
 ssh_genkeytypes:
 - ed25519
 - ecdsa
 - rsa
```

如果您使用 RSA 金鑰對進行公開金鑰驗證，您的 SSH 用戶端必須支援 rsa-sha2-256 或 rsa-sha2-512 簽章。如果您使用不相容的用戶端且無法升級，請重新啟用執行個體的 ssh-rsa 支援。若要重新啟用ssh-rsa支援，請使用下列命令啟用LEGACY系統加密政策。

```
$ sudo dnf install crypto-policies-scripts
$ sudo update-crypto-policies --set LEGACY
```

如需管理主機金鑰的詳細資訊，請參閱 Amazon Linux 主機金鑰。

AL2023 中的已棄用功能

AL2 中已棄用且 AL2023 中不存在的功能會在此處記錄。這是 AL2 中存在但 AL2023 中沒有的功能和套件等功能，不會新增至 AL2023。如需 AL2 中支援功能多久的詳細資訊，請參閱 AL2 中的已棄用功能。

AL2023 中也有已棄用的功能，且在未來版本中會移除。本章說明該功能是什麼、何時不再支援，以及何時將從 Amazon Linux 移除。了解已棄用的功能可協助您部署 AL2023，並為下一個主要版本的 Amazon Linux 做好準備。

主題

compat- 套件

AL2 中任何字首為 的套件compat-都會提供，以便與尚未針對套件的現代版本重建的舊二進位檔相容。每個新的 Amazon Linux 主要版本都不會轉送先前版本的任何compat-套件。

Amazon Linux 版本 （ 例如 AL2) 中的所有compat-套件都已棄用，且不會出現在後續版本 （ 例如 AL2023) 中。我們強烈建議根據程式庫的更新版本重建軟體。

在 AL1 中停止的已棄用功能，在 AL2 中移除

本節說明 AL1 中可用的功能，且不再可用於 AL2。

> **Note**
>
> 作為 AL1 維護支援階段的一部分，某些套件的end-of-life(EOL) 日期早於 AL1 的 EOL。如需詳細資訊，請參閱 AL1 套件支援陳述式。

> **Note**
>
> 部分 AL1 功能已在舊版中停止。如需詳細資訊，請參閱 AL1 版本備註。

主題

32 位元 x86 (i686) AMIs

在 2014.09 版的 AL1 中，Amazon Linux 宣布將是生產 32 位元 AMIs的最後一個版本。因此，從 2015.03 版的 AL1 開始，Amazon Linux 不再支援以 32 位元模式執行系統。AL2 為 x86-64 主機上的 32 位元二進位檔案提供有限的執行時間支援，並且不提供開發套件來建立新的 32 位元二進位檔案。AL2023 不再包含任何 32 位元使用者空間套件。我們建議使用者在遷移至 AL2023 之前，先完成其 64 位元程式碼的轉換。

如果您需要在 AL2023 上執行 32 位元二進位檔，則可以在 AL2023 上執行的 AL2 容器內使用 AL2023 位元使用者空間。

`aws-apitools-*` 取代為 AWS CLI

在 2013 AWS CLI 年 9 月發行 之前， AWS 已提供一組命令列公用程式，實作於 Java，允許使用者進行 Amazon EC2 API 呼叫。這些工具已於 2015 年終止，並 AWS CLI 成為從命令列與 Amazon EC2 APIs互動的偏好方式。命令列公用程式集包含下列`aws-apitools-*`套件。

- `aws-apitools-as`
- `aws-apitools-cfn`
- `aws-apitools-common`
- `aws-apitools-ec2`
- `aws-apitools-elb`
- `aws-apitools-mon`

對 2017 年 3 月結束之`aws-apitools-*`套件的上游支援。儘管缺乏上游支援，但 Amazon Linux 仍繼續提供某些命令列公用程式，例如 `aws-apitools-ec2`，以提供使用者的回溯相容性。 AWS CLI

是比aws-apitools-*套件更強大且完整的工具，因為它會主動維護，並提供使用 AWS APIs的方法。

aws-apitools-* 套件已於 2017 年 3 月棄用，不會再收到更新。任何這些套件的所有使用者都應該 AWS CLI 盡快遷移到 。這些套件不存在於 AL2023 中。

AL1 也提供 aws-apitools-iam和 aws-apitools-rds套件，這些套件已在 AL1 中棄用，且從 AL2 之後不會出現在 Amazon Linux 中。

systemd 在 AL2 upstart中取代

AL2 是第一個使用 systemd init 系統的 Amazon Linux 版本，取代 AL1 upstart中的 。任何upstart特定組態都必須在從 AL1 遷移至較新版本的 Amazon Linux 的過程中進行變更。無法在 AL1 systemd上使用 ，因此從 移至 upstartsystemd只能作為移至較新主要 Amazon Linux 版本的一部分，例如 AL2 或 AL2023。

AL2 中的功能已棄用，並在 AL2023 中移除

本節說明 AL2 中提供的功能，且不再在 AL2023 中提供。

主題

- 32 位元 x86 (i686) 套件
- aws-apitools-* 取代為 AWS CLI
- awslogs 已棄用，以支持統一的 Amazon CloudWatch Logs 代理程式
- bzr 修訂控制系統
- cgroup v1
- log4j hotpatch (log4j-cve-2021-44228-hotpatch)
- lsb_release 和 system-lsb-core 套件
- mcrypt
- OpenJDK 7 (java-1.7.0-openjdk)
- Python 2.7
- rsyslog-openssl 取代 rsyslog-gnutls
- 網路資訊服務 (NIS)/ yp
- Amazon VPC 中的多個網域名稱 create-dhcp-options
- Sun RPC 中的 glibc

- audit 日誌中的 OpenSSH 金鑰指紋
- ld.gold 連結器
- ping6
- ftp 套件

32 位元 x86 (i686) 套件

在 2014.09 版的 AL1 中，我們宣布這是生產 32 位元 AMIs的最後一個版本。因此，從 2015.03 版的 AL1 開始，Amazon Linux 不再支援以 32 位元模式執行系統。AL2 在 x86-64 主機上為 32 位元二進位檔案提供有限的執行時間支援，並且不提供開發套件來建立新的 32 位元二進位檔案。AL2023 不再包含任何 32 位元使用者空間套件。我們建議客戶完成轉換為 64 位元程式碼。

如果您需要在 AL2023 上執行 32 位元二進位檔，則可以在 AL2023 上執行的 AL2 容器內使用 AL2023 位元使用者空間。

`aws-apitools-*` 取代為 AWS CLI

在 AWS CLI 2013 年 9 月發行 之前， AWS 已提供一組在 中實作的命令列公用程式Java，讓客戶能夠進行 Amazon EC2 API 呼叫。這些工具已於 2015 年棄用， AWS CLI 成為從命令列與 Amazon EC2 APIs 互動的偏好方式。這包括下列`aws-apitools-*`套件。

- `aws-apitools-as`
- `aws-apitools-cfn`
- `aws-apitools-common`
- `aws-apitools-ec2`
- `aws-apitools-elb`
- `aws-apitools-mon`

上游支援於 2017 年 3 月結束的`aws-apitools-*`套件。雖然缺乏上游支援，但 Amazon Linux 繼續運送其中一些命令列公用程式 （ 例如 `aws-apitools-ec2`)，以便為客戶提供回溯相容性。 AWS CLI 是比`aws-apitools-*`套件更強大且完整的工具，因為它會主動維護，並提供使用 AWS APIs的方法。

`aws-apitools-*` 套件已於 2017 年 3 月棄用，不會再收到更新。任何這些套件的所有使用者都應該 AWS CLI 盡快遷移到 。這些套件不存在於 AL2023 中。

awslogs 已棄用，以支持統一的 Amazon CloudWatch Logs 代理程式

awslogs 套件已在 AL2 中棄用，且不再存在於 AL2023 中。它被統一的 CloudWatch Logs 代理程式取代，可在 amazon-cloudwatch-agent套件中使用。如需詳細資訊，請參閱 Amazon CloudWatch Logs 使用者指南。

bzr 修訂控制系統

GNU Bazaar (bzr) 修訂版控制系統在 AL2 中停止，不再存在於 AL2023 中。

bzr 建議 的使用者將其儲存庫遷移至 git。

cgroup v1

AL2023 移至統一控制群組階層 (cgroup v2)，而 AL2 使用 cgroup v1。由於 AL2 不支援 cgroup v2，因此此遷移需要完成，才能移至 AL2023。

log4j hotpatch (log4j-cve-2021-44228-hotpatch)

> **Note**
>
> log4j-cve-2021-44228-hotpatch 套件已在 AL2 中棄用，並在 AL2023 中移除。

為了回應 CVE-2021-44228，Amazon Linux 針對 AL1 和 AL2 發行了適用於 Apache Log4j 的 Hotpatch 的 RPM 封裝版本。在將熱修補程式新增至 Amazon Linux 的公告中，我們注意到「安裝熱修補程式無法取代更新至可緩解 CVE-2021-44228 或 CVE-2021-45046 的 log4j 版本。」

該熱修補是一種緩解措施，讓您有時間修補 log4j。AL2023 的第一個一般可用性版本是在 CVE-2021-44228 後 15 個月，因此 AL2023 不會隨附熱修補程式（無論是否啟用）。

建議在 Amazon Linux 上執行自有 log4j 版本的客戶確認已更新至不受 CVE-2021-44228 或 CVE-2021-45046 影響的版本。

lsb_release 和 system-lsb-core 套件

在過去，某些軟體會調用 lsb_release 命令 (在 AL2 中由套件 system-lsb-core 提供) 以取得關於執行軟體所用的 Linux 發行版資訊。Linux 標準規範 (LSB) 已導入此命令，且 Linux 發行版已加以採

用。Linux 發行版已發展為使用更簡單的標準，以將此資訊保留在 /etc/os-release 和其他相關檔案內。

os-release 標準源自 systemd。如需詳細資訊，請參閱系統作業系統版本文件。

AL2023 不隨附 lsb_release 命令，也不包含 system-lsb-core 套件。軟體應完成 os-release 標準的轉換，以維持與 Amazon Linux 和其他主要 Linux 發行版的相容性。

mcrypt

程式mcrypt庫和相關聯的PHP延伸模組已在 AL2 中棄用，且不再存在於 AL2023 中。

上游已PHP棄用 7.1 中的延伸模組，該延伸模組於 2016 年 12 月首次發行，並於 2019 年 10 月發行。mcryptPHP

上游mcrypt程式庫上次在 2007 年發行版本，並且尚未從 SourceForge 在 2017 年新遞交所需的cvs修訂控制進行遷移，最近的遞交 （僅 3 年前） 是從 2011 年開始，移除提及具有維護器的專案。

建議任何剩餘的使用者mcrypt將其程式碼移植到 OpenSSL，因為 mcrypt 不會新增到 AL2023。

OpenJDK 7 (java-1.7.0-openjdk)

> **Note**
>
> AL2023 提供數個版本的 Amazon Corretto，以支援Java以 為基礎的工作負載。OpenJDK 7 套件已在 AL2 中棄用，不再存在於 AL2023 中。AL2023 中最舊的 JDK 由 Corretto 8 提供。

如需 Amazon Linux 上 Java 的詳細資訊，請參閱 AL2023 中的 Java。

Python 2.7

> **Note**
>
> AL2023 已移除 Python 2.7，任何需要 Python 的作業系統組件都編寫為配合 Python 3 運作。若要繼續使用由 Amazon Linux 提供並支援的 Python 版本，請將 Python 2 程式碼轉換為 Python 3。

如需 Amazon Linux 上的 Python 詳細資訊，請參閱 AL2023 中的 Python。

rsyslog-openssl 取代 rsyslog-gnutls

rsyslog-gnutls 套件已在 AL2 中棄用，不再存在於 AL2023 中。rsyslog-openssl 套件應該是rsyslog-gnutls任何套件使用情況的插入式取代。

網路資訊服務 (NIS)/ yp

Network Information Service (NIS)，最初稱為黃頁或YP已在 AL2 中棄用，不再存在於 AL2023 中。這包括下列套件：ypbind、 ypserv和 yp-tools。與 整合的其他套件NIS會在 AL2023 中移除此功能。

Amazon VPC 中的多個網域名稱 create-dhcp-options

在 Amazon Linux 2 中，可以將 domain-name 參數中的多個網域名稱傳遞至 create-dhcp-options，這會導致 /etc/resolv.conf 包含類似 的內容search foo.example.com bar.example.com。Amazon VPC DHCP 伺服器使用DHCP選項 15 傳送提供的網域名稱清單，僅支援單一網域名稱（請參閱 RFC 2132 第 3.17 節）。由於 AL2023 使用 systemd-networkd進行遵循的網路組態RFC，AL2 中的此意外功能不會出現在 AL2023 上

AWS CLI 和 Amazon VPC 文件包含以下內容：「某些 Linux 作業系統接受以空格分隔的多個網域名稱。不過， Windows和其他 Linux 作業系統會將該值視為單一網域，這會導致非預期的行為。如果您的DHCP選項集與 Amazon VPC 相關聯，而該 Amazon VPC 的執行個體正在執行將值視為單一網域的作業系統，請僅指定一個網域名稱。 「

在這些系統上，例如 AL2023，使用DHCP選項 15 指定兩個網域（只允許一個），而且由於網域名稱中的空格字元無效，這會導致空格字元編碼為 032，導致/etc/resolv.conf包含 search foo.exmple.com032bar.example.com。

為了支援多個網域名稱，DHCP伺服器應使用DHCP選項 119（請參閱 RFC 3397 第 2 節）。請參閱 Amazon VPC 使用者指南，了解 Amazon VPC DHCP 伺服器何時支援此功能。

Sun RPC 中的 glibc

在 中的 實作Sun RPCglibc已在 AL2 中棄用，並在 AL2023 中移除。如果需要 Sun RPC功能，建議客戶使用 libtirpc程式庫 (AL2 和 AL2023 中提供）。採用 libtirpc也可讓應用程式支援 IPv6。

此變更反映更廣泛的社群採用上游glibc移除此功能，例如從 Fedora glibc中移除Sun RPC介面，以及 Gentoo 中的類似變更。

audit 日誌中的 OpenSSH 金鑰指紋

稍後在 AL2 的生命週期中，修補程式已新增至 OpenSSH 套件，以發出用於驗證的金鑰指紋。AL2023 中不存在此功能。

ld.gold 連結器

ld.gold 連結器可在 AL2 中使用，並在 AL2023 中移除。建置明確參考gold連結器的軟體的客戶應遷移至一般 (ld.bfd) 連結器。

2.44 版 (2025 年 2 月發行） 的上游 GNU Binutils 版本備註記載移除 ld.gold：「在先前實務的變更中，在此版本中，binutils-2.44.tar tarball 不包含黃金連結器的來源。 https://lists.gnu.org/archive/html/info-gnu/2025-02/msg00001.html這是因為黃金連結器現在已棄用，除非志願者向前邁進並提議繼續開發和維護，否則最終將被移除。」

ping6

在 AL2023 中，一般ping公用程式原生支援 IPv6，且/bin/ping6不再需要單獨的 。在 AL2023 /usr/bin/ping 中， /usr/sbin/ping6是可執行檔的符號連結。

此變更遵循更廣泛的社群採用提供此功能的較新iputils版本，例如 Fedora 中的 Ping IPv6 變更。

ftp 套件

從 AL2023 開始，Amazon Linux 不再提供 AL2 中的ftp套件。此決策是我們對安全、可維護性和現代軟體開發實務的持續承諾的一部分。作為遷移至 AL2023 的一部分 （或之前），我們建議將舊版ftp套件的任何使用遷移至其替代方案之一。

背景介紹

傳統ftp套件在上游並未主動維護多年。上次對原始程式碼的重大更新發生在 2000 年代早期，而原始來源儲存庫已不再可用。雖然某些 Linux 發行版本已針對安全性漏洞實作修補程式，但程式碼庫在很大程度上仍無法維護。

建議的替代方案

AL2023 為 FTP 功能提供數種現代且積極維護的替代方案：

lftp （適用於 AL2 和 AL2023)

支援 FTP、HTTP、SFTP 和其他通訊協定的複雜檔案傳輸程式。它提供比傳統ftp用戶端更多的功能，並且會主動維護。

使用 進行安裝： dnf install lftp

`curl`（適用於 AL2 和 AL2023)

使用 URLs 傳輸資料的多用途命令列工具，支援 FTP、FTPS、HTTP、HTTPS 和許多其他通訊協定。

根據預設，AL2023 會透過 `curl-minimal`套件提供。如需更廣泛的通訊協定支援，您可以選擇`curl-full`使用 升級至 dnf swap curl-minimal curl-full。

`wget`（適用於 AL2 和 AL2023)

非互動式命令列公用程式，用於從 Web 下載檔案，並支援 HTTP、HTTPS 和 FTP 通訊協定。

使用 進行安裝：dnf install wget（並非所有 AL2023 映像預設安裝）

`sftp`（適用於 AL2 和 AL2023)

透過 SSH 操作的安全檔案傳輸通訊協定，提供加密的檔案傳輸。

根據預設，作為 OpenSSH 套件的一部分提供。

遷移考量事項

如果您的應用程式或指令碼取決於舊版`ftp`用戶端，請考慮下列遷移方法：

1. 更新指令碼以使用現代替代方案：修改指令碼以使用 `lftp`、`wget`、 `curl`或 ，`sftp`而非舊版`ftp`用戶端。
2. 檢閱套件相依性：某些應用程式可能會在其`ftp`套件中繼資料中將套件列為相依性，即使自遷移至內部使用現代通訊協定以來已很久。在這些情況下，即使`ftp`套件`/usr/bin/ftp`缺少 ，應用程式仍可在 AL2023 上正常運作。檢閱應用程式的實際需求，而不是僅依賴指定的相依性。
3. 更新應用程式相依性：對於您維護但仍宣告對`ftp`套件的相依性，但未實際使用的應用程式，請更新套件中繼資料以移除此不必要的相依性。

安全考量

FTP 通訊協定會以純文字傳輸資料，包括身分驗證憑證。對於對安全敏感的應用程式，我們強烈建議使用建議的替代工具支援的加密替代方案，例如 SFTP 或 HTTPS。

在 AL2023 中已棄用

本節說明 AL2023 中存在的功能，並且可能會在未來的 Amazon Linux 版本中移除。每個區段都會描述功能為何，以及預期何時從 Amazon Linux 中移除功能。

> **Note**
>
> 隨著 Linux 生態系統的演進和未來的 Amazon Linux 主要版本更接近發行，本節將隨著時間更新。

主題

32 位元 x86 (i686) 執行時間支援

AL2023 保留執行 32 位元 x86 (i686) 二進位檔的能力。Amazon Linux 的下一個主要版本可能不再支援執行 32 位元使用者空間二進位檔。

`aspell`

雖然 AL2023 隨附於`aspell`套件，但已棄用，並將在下一個 Amazon Linux 主要版本中移除。建議客戶遷移至現代替代，例如 `hunspell`或 `enchant2`。

AL2023 `aspell`中的 棄用遵循更廣泛的社群轉移，例如 `aspell` 中的棄用Fedora。

Berkeley 資料庫 (`libdb`)

AL2023 隨附 Berkeley 資料庫 (`libdb`) 程式庫的 5.3.28 版。這是授權從限制較少的 Sleepycat 授權變更為 GNU Affero GPLv3 (AGPL) 授權之前，Berkeley 資料庫的最新版本。

AL2023 中很少套件仍依賴於 Berkeley 資料庫 (libdb)，而且程式庫會在下一個主要版本的 Amazon Linux 中移除。

> **Note**
>
> AL2023 中的dnf套件管理員保留對 Berkeley 資料庫 (BDB) 格式rpm資料庫的唯讀支援。此支援將在 Amazon Linux 的下一個主要版本中移除。

的棄用libdb遵循更廣泛的社群轉移，例如 libdb中的棄用Fedora。

cron

cronie 套件預設安裝在 AL2 AMI 上，為排誠定期任務的傳統 crontab 方式提供支援。在 AL2023 中， 預設不包含 cronie 。因此，預設crontab不再提供 的支援。

在 AL2023 中，您可以選擇安裝cronie套件以使用傳統cron任務。由於 systemd 提供的新增功能，建議您遷移至 systemd 計時器。

未來版本的 Amazon Linux 可能是下一個主要版本，可能不再包含對傳統cron任務的支援，並完成轉換為systemd計時器。我們建議您使用 移出 cron。

IMDSv1

根據預設，AL2023 AMIs 設定為以IMDSv2僅限 模式啟動，停用 IMDSv1。仍然可以選擇在啟用 IMDSv1 的情況下使用 AL2023。Amazon Linux 的未來版本可能會強制執行IMDSv2僅限 。

如需 AMIs的 IMDS 組態詳細資訊，請參閱《Amazon EC2 使用者指南》中的設定 AMI。

pcre 第 1 版

舊版pcre套件已棄用，並將在下一個 Amazon Linux 主要版本中移除。pcre2 套件是後繼版本。雖然 AL2023 的第一個版本隨附針對 建置的有限數量套件pcre，但這些套件會在 AL2023 pcre2內遷移至 。已棄用的pcre程式庫將保留在 AL2023 中。

> **Note**
>
> 已棄用版本的 pcre將不會在 AL2023 的完整生命週期內收到安全性更新。如需pcre支援生命週期和套件將接收安全性更新時間的詳細資訊，請參閱pcre套件上的套件支援陳述式。

pcre 的棄用有利於 pcre2 遵循此方向更廣泛的社群轉移，例如 pcre 中的棄用Fedora。

System V init (**sysvinit**)

雖然 AL2023 保留System V與服務 (init) 指令碼的回溯相容性，但作為 v254 版本的一部分，上游systemd專案已宣布棄用對 System V 服務指令碼的支援，並指出將在未來的 版本中移除支援systemd。如需詳細資訊，請參閱systemd。

AL2023 將保持System V與服務 (init) 指令碼的回溯相容性，但建議使用者遷移至使用原生systemd單位檔案，以便在從 Amazon Linux 移除System V服務 (init) 指令碼支援時做好準備，這可能是在下一個主要版本中。

EOL 套件已棄用

AL2023 中提供的每個套件都有相關的支援陳述式，涵蓋 Amazon Linux 特定資訊。這些陳述式涵蓋作業系統的核心及其生命週期，以及 the section called “PHP”和 等套件the section called “Python”，其中 AL2023 會運送多個版本，並在上游開放原始碼專案執行的期間支援每個版本。

在 AL2023 中，您可以使用套件管理員取得dnf套件支援資訊。如需詳細資訊，請參閱取得套件支援資訊。

在 Amazon Linux 主要版本結束之前不再支援套件的情況下，應該假設此套件已棄用，且不會出現在下一個 Amazon Linux 主要版本中。

對於 the section called “PHP”和 等套件the section called “Python”，其中每個主要 Amazon Linux 版本都已運送多個版本，每個版本都有不同的支援生命週期，它們可能會繼續存在於新的 Amazon Linux 主要版本中，即使與套件的主要版本很少或沒有重疊。建議在選取相依性時，記住 Amazon Linux 套件支援時間表。

比較 AL2 與 AL2023

下列主題說明 AL2 和 AL2023 之間的主要差異。

如需 AL1, AL2和 AL2023 中已棄用功能的詳細資訊，請參閱 AL2023 中的已棄用功能。

主題

- AWS CLIv2
- UEFI 偏好和安全開機
- SSH 伺服器預設組態變更
- 從 AL2 的 AL2023 核心變更 AL2
- /tmp 現在是 tmpfs
- AMI 和容器映像變更
- 比較 Amazon Linux 2 與 Amazon 2023 AMI 安裝的套件
- 比較 Amazon Linux 2 與 Amazon 2023 最小 AMI 安裝的套件
- 比較 Amazon Linux 2 與 Amazon 2023 基本容器映像安裝的套件

新增、升級和移除的套件

AL2023 包含數千個可供使用的軟體套件。如需 AL2023 中新增、升級或移除的所有套件完整清單 (與舊版 Amazon Linux 相比)，請參閱 AL2023 中的 Package 件變更。

若要請求在 AL2023 中新增或變更套件，請在 GitHub 的 amazon-linux-2023 儲存庫中提出問題。

支援各個版本

對於 AL2023，我們提供五年支援。

如需詳細資訊，請參閱發行節奏。

命名和版本控制變更

AL2023 支援 AL2 支援平台識別的相同機制。AL2023 還導入用於平台識別的新檔案。

如需詳細資訊，請參閱命名和版本控制。

最佳化

AL2023 可最佳化開機時間，以縮短從執行個體啟動到執行客戶工作負載的時間。這些最佳化涵蓋 Amazon EC2 執行個體核心組態、`cloud-init` 組態，以及作業系統套件內建的功能，例如 `kmod` 和 `systemd`。

如需關於最佳化的詳細資訊，請參閱 效能與操作最佳化。

源自多個上游

AL2023 以 RPM 為基礎，並包含源自多個版本的 Fedora 及其他發行版本的元件，例如 CentOS 9 Stream。Amazon Linux 核心源自直接從 kernel.org 發行的長期支援 (LTS) 發行版，並從其他發行版中獨立選出。

如需詳細資訊，請參閱與 Fedora 的關係。

網路系統服務

`systemd-networkd` 系統服務管理 AL2023 中的網路介面。這是與 AL2 不同的變更，且後者使用 ISC dhclient 或 `dhclient`。

如需詳細資訊，請參閱網路服務。

套件管理工具

AL2023 的預設軟體套件管理工具為 DNF。DNF 是 AL2 套件管理工具 YUM 的後續工具。

如需詳細資訊，請參閱套件管理工具。

使用 cloud-init

在 AL2023 中，cloud-init 會管理套件儲存庫。在舊版的 Amazon Linux 中，cloud-init 預設安裝安全更新。這不是 AL2023 的預設值。用於在啟動時進行更新 `releasever` 的新確定性升級功能說明 AL2023 在啟動時啟用套件更新的方法。如需詳細資訊，請參閱在 AL2023 中管理套件和作業系統更新及穩定性的確定性升級。

使用 AL2023 時，您可以將 cloud-init 與 SELinux 搭配使用。如需詳細資訊，請參閱使用 cloud-init 來啟用 `enforcing` 模式。

Cloud-init 使用 HTTP(S) 以從遠端位置載入 cloud-init 的組態內容。在舊版本中，當遠端資源無法使用時，Amazon Linux 不會提醒您。在 AL2023 中，不可用的遠端資源會產生嚴重錯誤，並且 cloud-init 執行失敗。AL2 的這種行為變更提供更安全的「失敗關閉」預設行為。

如需詳細資訊，請參閱 自訂 cloud-init 和 cloud-init 文件。

圖形桌面支援

AL2023 具有截至 2023.7 版的 GNOME 型圖形桌面環境，取代了 AL2 中使用的 MATE 桌面。此版本為使用者提供不同的桌面體驗，同時維持 AL2023 的雲端最佳化效能。GNOME 桌面環境提供各種自訂選項、系統整合功能和不同的使用者介面設計，為使用者提供先前 MATE 桌面環境的替代方案。如需詳細資訊，請參閱 AL2023 圖形桌面頁面。

編譯器三元組

AL2023針對 GCC 和 LLVM 設定編譯器三元組，以指出 amazon 是供應商。

因此，AL2 aarch64-redhat-linux-gcc 在 AL2023 上變成 aarch64-amazon-linux-gcc。

這對大多數使用者來說應該是完全透明的，而且可能只會影響在 AL2023 上建置編譯器的使用者。

32 位元 x86 (i686) 套件

在 2014.09 版的 AL1 中，已宣布它是產生 32 位元 AMIs的最後一個版本。因此，從 AL1 的 2015.03 版本開始，Amazon Linux 不再支援以 32 位元模式執行系統。AL2 針對 x86-64 主機的 32 位元二進位檔案提供有限的執行期支援，且未提供開發套件來建置新的 32 位元二進位檔案。AL2023 不再包含任何 32 位元使用者空間套件。我們建議您完成 64 位元程式碼的轉換。

如果您需要在 AL2023 上執行 32 位元二進位檔案，則可在基於 AL2023 的 AL2 容器內部使用 AL2 的 32 位元使用者空間。

lsb_release 和 system-lsb-core 套件

在過去，某些軟體會調用 lsb_release 命令 (在 AL2 中由套件 system-lsb-core 提供) 以取得關於執行軟體所用的 Linux 發行版資訊。Linux 標準規範 (LSB) 已導入此命令，且 Linux 發行版已加以採用。Linux 發行版已發展為使用更簡單的標準，以將此資訊保留在 /etc/os-release 和其他相關檔案內。

os-release 標準源自 systemd。如需詳細資訊，請參閱系統作業系統版本文件。

AL2023 不隨附 lsb_release 命令，也不包含 system-lsb-core 套件。軟體應完成 os-release 標準的轉換，以維持與 Amazon Linux 和其他主要 Linux 發行版的相容性。

Extra Packages for Enterprise Linux (EPEL)

> **Warning**
>
> AL2 epel Extra 已啟用第三方EPEL7儲存庫。自 2024-06-30 起，不再維護第三方EPEL7儲存庫。
> 這個第三方儲存庫未來不會有更新。這表示 EPEL 儲存庫中的套件不會有安全性修正。
> 本節將涵蓋 AL2023 中的套件選項EPEL。

Extra Packages for Enterprise Linux (EPEL) 是 Fedora 社群的專案，目的是為企業級 Linux 作業系統建立大量套件。該項目主要生產 RHEL 和 CentOS 套件。AL2 與 CentOS 7 高度相容。因此，許多 EPEL7 套件都在 AL2 上運作。

沒有與 AL2023 相容的二進位EPEL版本。不過，想要在 AL2023 中使用其EPEL7套件的客戶有幾個選項。有些EPEL套件在 AL2023 中具有替代方案，而其他套件則作為 的一部分提供Amazon Linux 的補充套件。

> **Warning**
>
> 僅新增旨在與 AL2023 搭配使用的儲存庫。
> 雖然專為其他 分佈設計的儲存庫可能目前有效，但無法保證他們會繼續使用 AL2023 中的任何套件更新，或並非設計用於 AL2023 的儲存庫。

此頁面提供客戶在 AL2 上使用的EPEL7套件及其 AL2023 複本的相關資訊。

對於其他套件，客戶可能可以使用適用於 Amazon Linux (SPAL) 的補充套件。SPAL 提供數千個專為 Amazon Linux 2023 建置的EPEL9套件，但 AWSSupport Plans 不涵蓋這些套件。這表示不會追蹤 SPAL 套件的 CVEs，而且只有在上游可用時才會提供修補程式。

> **Important**
>
> 使用前Amazon Linux 的補充套件，請參閱 的文件。

主題

- axel - HTTP/FTP 用戶端

- brotli 和 libbrotli - 壓縮
- collectd - 統計資料收集常駐程式
- cpulimit - CPU 用量限制器
- exim - 郵件傳輸代理程式
- fuse3 - 使用者空間中的檔案系統 (FUSE) v3
- ganglia - 分散式監控系統
- git-lfs - 版本使用 Git 控制大型檔案
- haveged - 使用HAVEGE演算法的熵來源
- inotify-tools - inotify 命令列工具
- iperf - TCP/UDP 效能基準
- jemalloc - 替代malloc實作
- libbsd - BSD 相容函數程式庫
- libserf - HTTP 用戶端程式庫
- libzstd - zstd 壓縮程式庫
- lighttpd Web 伺服器
- lshell - 受限制的 shell
- monit - 程序、檔案、目錄和裝置監控
- nodejs
- perl-Config-General
- python2-lockfile - 檔案鎖定
- python2-rsa - 純 Python RSA
- python2-simplejson - Python 2 的 JSON 常式
- rkhunter - Rootkit Hunter
- rssh - 與 OpenSSH 搭配使用的受限制 Shell
- sscg - 自我簽署的 SSL 憑證產生器
- stress - 壓力測試
- stress-ng - 壓力測試
- tmpwatch - 根據上次存取時間移除檔案
- xmlstarlet - 命令列 XML 公用程式

axel - HTTP/FTP 用戶端

axel 套件位於 中EPEL7，從未做為 Amazon Linux 的一部分運送。AL2023 中可用的替代方案為 curl和 wget。

> **Warning**
>
> axel 使用未加密http連線來探索檔案鏡像-S的選項。

強烈建議將 的任何使用遷移axel到 curl或 wget。

brotli 和 libbrotli - 壓縮

brotli 和 libbrotli套件位於 中EPEL7，而只有 brotli套件可在 AL2 核心中使用。

brotli 和 libbrotli套件都包含在 AL2023 中。

brotli 套件可以使用下列命令安裝在 AL2023 上：

```
[ec2-user ~]$ sudo dnf install brotli
```

libbrotli 套件可以使用下列命令安裝在 AL2023 上：

```
[ec2-user ~]$ sudo dnf install libbrotli
```

collectd - 統計資料收集常駐程式

collect 套件位於 EPEL7，也可在 collectd和 AL2 Extras collectd-python3 中使用。

collectd 套件包含在 AL2023 中，可透過執行下列命令來安裝：

```
[ec2-user ~]$ sudo dnf install collectd
```

cpulimit - CPU 用量限制器

在 Amazon Linux 2023 中， systemd提供限制程序或一組程序 CPU 用量的功能。此功能也可輕鬆用於任何systemd服務。

提供強大的資源控制設施systemd，可用於確保任何任務或任務群組在可以取用的資源中受到限制。如需詳細資訊，請參閱上游 systemd.resource-control 文件，以及 使用 限制 AL2023 中的程序資源使用量 systemd。

exim - 郵件傳輸代理程式

exim 套件位於 ，EPEL7先前可在 AL1 中使用。Amazon Linux 2023 同時提供 postfix和 sendmail Mail Transfer Agents (MTAs)。

fuse3 - 使用者空間中的檔案系統 (FUSE) v3

fuse3 套件 （ 包括 fuse3-libs和 fuse3-devel) 位於 中EPEL7。這些套件是 AL2023 的一部分，每個套件都可以透過執行相關的下列命令來安裝：

```
[ec2-user ~]$ sudo dnf install fuse3
```

```
[ec2-user ~]$ sudo dnf install fuse3-libs
```

```
[ec2-user ~]$ sudo dnf install fuse3-devel
```

ganglia - 分散式監控系統

ganglia 套件位於 中EPEL7，先前可在 AL1 中使用。它未與 AL2 一起運送。

上游專案有一段時間沒有活動，其中未處理一些開啟CVEs。雖然上游專案中最近有活動，但計劃不將ganglia新增至 AL2023。

git-lfs - 版本使用 Git 控制大型檔案

git-lfs 套件位於 中EPEL7。在 Amazon Linux 2023 中，git-lfs套件包含在核心儲存庫中。在 AL2023 上執行下列命令git-lfs即可安裝 ：

```
[ec2-user ~]$ sudo dnf install git-lfs
```

haveged - 使用HAVEGE演算法的熵來源

haveged 套件位於 中EPEL7。Amazon Linux 2023 已預先設定熵來源，不需要使用 haveged。

`inotify-tools` - inotify 命令列工具

`inotify-tools` 套件位於 中EPEL7，並包含在 AL2023 中。

> **Note**
>
> 在 AL2023 中， `systemd`支援路徑型啟用，可用於對事件採取動作，例如當路徑存在或變更時。
> 許多用於 `inotify-tools` 的 現在可以使用`systemd`路徑啟用，以更可靠的方式完成。如需詳細資訊，請參閱 systemd.path。

`inotify-tools` 套件包含在 AL2023 中，可透過執行下列命令來安裝：

```
[ec2-user ~]$ sudo dnf install inotify-tools
```

`iperf` - TCP/UDP 效能基準

第 2 `iperf`版套件位於 中EPEL7，也可用於 `testing` AL2 Extra。 也可用於 AL1

> **Note**
>
> `iperf3` 套件也可使用，提供 第 3 版`iperf`。

`iperf` 套件包含在 AL2023 中，可透過執行下列命令來安裝：

```
[ec2-user ~]$ sudo dnf install iperf
```

`jemalloc` - 替代`malloc`實作

`jemalloc` 套件位於 EPEL7，並可在 `lamp-mariadb10.2-php7.2`和 AL2 Extras `mariadb10.5` 中使用。

`jemalloc` 套件包含在 AL2023 中，可透過執行下列命令來安裝：

```
[ec2-user ~]$ sudo dnf install jemalloc
```

libbsd - BSD 相容函數程式庫

libbsd 套件位於 中EPEL7，也可用於 testing AL2 Extra。

libbsd 套件包含在 AL2023 中，可透過執行下列命令來安裝：

```
[ec2-user ~]$ sudo dnf install libbsd
```

您可以執行下列命令來libbsd安裝 的開發檔案。

```
[ec2-user ~]$ sudo dnf install libbsd-devel
```

libserf - HTTP 用戶端程式庫

libserf 套件位於 中EPEL7。libserf 套件在 Amazon Linux 2023 中提供。您可以執行下列命令來安裝：

```
[ec2-user ~]$ sudo dnf install libserf
```

libzstd - zstd 壓縮程式庫

libzstd 套件位於 AL2 核心，以及 中EPEL7。libzstd 套件也是 AL2023 的一部分。

```
[ec2-user ~]$ sudo dnf install libzstd
```

lighttpd Web 伺服器

lighttpd 套件位於 中EPEL7，先前可在 AL1 中使用。Amazon Linux 2023 同時提供 Apache httpd和 nginx Web 伺服器。

lshell - 受限制的 shell

lshell 套件從未做為 Amazon Linux 的一部分運送。它可在 中使用EPEL6。的 Fedora 封裝儲存庫lshell涵蓋了未封裝於 EPEL7或 Fedora 30 的原因。它也已從 Debian 移除。

上游lshell專案不再主動維護，並包含已知未修補的關鍵 CVEs：CVE-2016-6902 和 CVE-2016-6903。

Debian 錯誤中建議的替代方案rssh，在上游也未維護，作者將無法修正的安全問題視為原因。

基於這些原因，未計劃lshell將 新增至 AL2023。

monit - 程序、檔案、目錄和裝置監控

在 Amazon Linux 2023 中， systemd提供監控、啟動、停止和重新啟動服務的各種功能。這包括速率限制重新啟動、在重新啟動嘗試之間等待，以及在失敗時啟動另一個服務。如需詳細資訊，請參閱 systemd.service 文件。

在 AL2023 中， systemd也支援路徑型啟用，可用於對事件採取動作，例如當路徑存在或變更時。如需詳細資訊，請參閱 systemd.path。

systemd 單位有常見的組態選項，允許指定相依性、條件和動作，以因應成功或失敗。如需詳細資訊，請參閱 systemd.unit 文件。

提供強大的資源控制設施systemd，可用於確保任何監控任務不會使用過多的 CPU 或記憶體。如需詳細資訊，請參閱 systemd.resource-control。

nodejs

nodejs 版本 16 套件位於 中EPEL7，現在nodejs已包含在 AL2023 中。在寫入時，nodejs版本 18 和 20 皆可在 AL2023 中使用。您可以使用下列命令在 nodejs AL2023 上安裝 18：

```
[ec2-user ~]$ sudo dnf install nodejs
```

您可以使用下列命令在 nodejs AL2023 上安裝 20：

```
[ec2-user ~]$ sudo dnf install nodejs20
```

perl-Config-General

perl-Config-General 套件位於 中EPEL7，現在已包含在 AL2023 中。您可以使用下列命令在 AL2023 中安裝perl-Config-General套件：

```
[ec2-user ~]$ sudo dnf install perl-Config-General
```

也可以要求 安裝提供特定 Perl 模組的套件來DNF安裝 Perl 模組。透過此方法，您可以使用更熟悉的 Perl 模組名稱，而不是作業系統套件名稱。

```
[ec2-user ~]$ sudo dnf install 'perl(Confg::General)'
```

python2-lockfile - 檔案鎖定

python2-lockfile 套件位於 中EPEL7，AL2 包含python-lockfile套件。在 AL2023 中Python 2.7 已替換為 Python 3，因此此套件的 Python 2 變體不會新增至 AL2023。

此套件的 Python 3 版本包含在 AL2023 中。您可以使用下列其中一個命令在 AL2023 中安裝python3-lockfile套件：

```
[ec2-user ~]$ sudo dnf install python3-lockfile
```

透過要求 DNF 安裝提供特定 Python 模組的套件，也可以安裝 Python 模組。

```
[ec2-user ~]$ sudo dnf install 'python3dist(lockfile)'
```

python2-rsa - 純 Python RSA

python2-rsa 套件位於 中EPEL7，AL2 包含python2-rsa套件。在 AL2023 中Python 2.7 已替換為 Python 3，因此此套件的 Python 2 變體不會新增至 AL2023。

此套件的 Python 3 版本包含在 AL2023 中。您可以使用下列其中一個命令在 AL2023 中安裝python3-rsa套件：

```
[ec2-user ~]$ sudo dnf install python3-rsa
```

透過要求 DNF 安裝提供特定 Python 模組的套件，也可以安裝 Python 模組。

```
[ec2-user ~]$ sudo dnf install 'python3dist(rsa)'
```

python2-simplejson - Python 2 的 JSON 常式

python2-simplejson 套件位於 中EPEL7。在 AL2023 中Python 2.7 已替換為 Python 3，因此此套件的 Python 2 變體將不會新增至 AL2023。

此套件的 Python 3 版本包含在 AL2023 中。您可以使用下列命令在 AL2023 中安裝python3-simplejson套件：

```
[ec2-user ~]$ sudo dnf install python3-simplejson
```

透過要求 DNF 安裝提供特定 Python 模組的套件，也可以安裝 Python 模組。

```
[ec2-user ~]$ sudo dnf install 'python3dist(simplejson)'
```

rkhunter - Rootkit Hunter

rkhunter 套件與 包含在 AL2023 中chkrootkit。

```
[ec2-user ~]$ sudo dnf install rkhunter
```

```
[ec2-user ~]$ sudo dnf install chkrootkit
```

rssh - 與 OpenSSH 搭配使用的受限制 Shell

rssh 套件位於 中EPEL7。上游rssh套件不受維護，作者將無法修正的安全問題視為原因。

作者引用無法修正的安全問題時，不會規劃rssh將 新增至 AL2023。

sscg - 自我簽署的 SSL 憑證產生器

sscg 套件位於 AL2 核心，以及 中EPEL7。sscg 套件也是 AL2023 的一部分。

```
[ec2-user ~]$ sudo dnf install sscg
```

stress - 壓力測試

stress 套件位於 中EPEL7，也可用於 AL1

stress 套件包含在 AL2023 中，可透過執行下列命令來安裝：

```
[ec2-user ~]$ sudo dnf install stress
```

stress-ng - 壓力測試

stress-ng 套件位於 中EPEL7，也可用於 testing AL2 Extra。

stress-ng 套件包含在 AL2023 中，可透過執行下列命令來安裝：

```
[ec2-user ~]$ sudo dnf install stress-ng
```

`tmpwatch` - 根據上次存取時間移除檔案

在 Amazon Linux 2023 中，此功能由 提供`systemd-tmpfiles`。

`xmlstarlet` - 命令列 XML 公用程式

`xmlstarlet` 套件位於 中EPEL7，不適用於 AL2023。

超過 9 年未接觸上游套件（上次接觸時間是 2014 年 8 月）。另外四年前（至少自 2010 年 7 月起），對新維護器的請求已無回應。因此，未計劃將 `xmlstarlet`新增至 AL2023。

Python 2.7 已替換為 Python 3

AL2 為 Python 2.7 提供支援和安全修補直到 2025 年 6 月為止，這包含在我們對 AL2 核心套件的長期支援 (LTS) 承諾中。此支援超出對 Python 2.7 上游 Python 社群聲明的 2020 年 1 月終止服務期限。

AL2 使用`yum`套件管理員，其對 Python 2.7 具有硬相依性。在 AL2023 中，`dnf` 套件管理器已遷移到 Python 3，不再需要 Python 2.7。AL2023 已經完全轉移到 Python 3。

> **Note**
>
> AL2023 已移除 Python 2.7，任何需要 Python 的作業系統組件都編寫為配合 Python 3 運作。若要繼續使用由 Amazon Linux 提供並支援的 Python 版本，請將 Python 2 程式碼轉換為 Python 3。

如需有關 Amazon Linux 的 Python 詳細資訊，請參閱 AL2023 中的 Python。

安全性更新

Amazon Linux 2023 可改善 AL2 中的強化功能。如需詳細資訊，請參閱Amazon Linux 2023 的安全與合規。如需 AL2 核心強化變更的詳細資訊，請參閱 重視安全的核心設定變更。

主題

- SELinux

- OpenSSL 3
- IMDSV2
- 移除 log4j 熱修補 (log4j-cve-2021-44228-hotpatch)

SELinux

預設情況下，適用於 AL2023 的 Security Enhanced Linux (SELinux) 是 `enabled` 且會設為 `permissive` 模式。在 `permissive` 模式中，會記錄權限遭拒的情形，但不會強制執行。

SELinux 是 Amazon Linux 核心的安全功能，在 AL2 是 `disabled`。SELinux 是核心功能和公用程式的集合，為核心的主要子系統提供強制存取控制 (MAC) 架構。

如需詳細資訊，請參閱設定 AL2023 的 SELinux 模式。

如需有關 SELinux 儲存庫、工具和策略的詳細資訊，請參閱 SELinux Notebook、SELinux 政策類型和 SELinux Project。

OpenSSL 3

AL2023 具有 Open Secure Sockets Layer version 3 (OpenSSL 3) 加密工具組的功能。AL2023 支援 TLS 1.3 和 TLS 1.2 網路通訊協定。

AL2 預設隨附 OpenSSL 1.0.2。您可以針對 OpenSSL 1.1.1 建置應用程式。

如需有關 OpenSSL 的詳細資訊，請參閱 OpenSSL 遷移指南。

如需詳細資訊，請參閱 安全更新和功能。

IMDSV2

根據預設，使用 AL2023 AMI 啟動的任何執行個體都只需要 ，IMDSv2而且您的預設跳轉限制會設為 2，以允許容器化工作負載支援。這是透過將 `imds-support` 參數設為 `v2.0` 來完成。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的設定 AMI。 Amazon EC2

> **Note**
>
> 工作階段記號的有效時間介於 1 秒到 6 小時之間。用來導向 IMDSv2 查詢的 API 要求的位址如下：
>
> - IPv4：169.254.169.254

- IPv6：fd00:ec2::254

您可以手動覆寫這些設定IMDSv1，並使用執行個體中繼資料選項啟動屬性啟用 。您也可以使用 IAM 控制項來強制執行不同的IMDS設定。如需設定和使用執行個體中繼資料服務的詳細資訊，請參閱《Amazon EC2 使用者指南》中的使用 IMDSv2、設定新執行個體的執行個體中繼資料選項，以及修改現有執行個體的執行個體中繼資料選項。

移除 log4j 熱修補 (`log4j-cve-2021-44228-hotpatch`)

> **Note**
>
> AL2023 未隨附 `log4j-cve-2021-44228-hotpatch` 套件。

為了回應 CVE-2021-44228，Amazon Linux 針對 AL1 和 AL2 發行了適用於 Apache Log4j 的 Hotpatch 的 RPM 封裝版本。在對 Amazon Linux 加入熱修補的公告中，我們發現「安裝熱修補不能替代更新到 log4j 版本，以緩解 CVE-2021-44228 或 CVE-2021-45046。」

該熱修補是一種緩解措施，讓您有時間修補 `log4j`。AL2023 的第一個正式上市 (GA) 版本是在 CVE-2021-44228 的 15 個月後，因此 AL2023 不會隨附熱修補 (無論是否啟用)。

在 Amazon Linux 上執行自己的`log4j`版本的使用者應確保他們已更新至不受 CVE-2021-44228 或 CVE-2021-45046 影響的版本。

AL2023 提供有關 更新 AL2023 的指引，讓您隨時掌握最新的安全修補。安全建議發佈在 Amazon Linux 安全中心。

穩定性的確定性升級

透過版本控制的儲存庫功能進行確定性升級，依預設，每個 AL2023 AMI 都會鎖定到特定的儲存庫版本。您可以使用決定性升級，在套件版本和更新之間取得更高的一致性。每個主要或次要版本都包含特定的儲存庫版本。

AL2023 的新功能會預設啟用確定性升級。與 AL2 和其他舊版本使用的手動增量鎖定方法相比，這是一大改進。

如需詳細資訊，請參閱透過 AL2023 上的版本控制儲存庫進行確定性升級。

gp3 作為預設 Amazon EBS 磁碟區類型

AL2023 AMI 和 AL2 的根檔案系統都使用 XFS 檔案系統。對於 AL2023，根裝置檔案系統的 `mkfs` 選項已針對 Amazon EC2 進一步最佳化。AL2023 也支援許多其他檔案系統，您可以在其他磁碟區使用這些檔案系統，以符合您的特定需求。

AL2023 AMI 預設使用 Amazon EBS gp3 磁碟區，而 AL2 AMI 預設使用 Amazon EBS gp2 磁碟區。您可以在啟動執行個體時變更磁碟區類型。

如需 Amazon EBS 磁碟區類型的詳細資訊，請參閱 Amazon EBS 一般用途磁碟區。

如需啟動 Amazon EC2 執行個體的詳細資訊，請參閱《Amazon EC2 使用者指南》中的啟動執行個體。

整合控制群組階層 (cgroup v2)

控制群組 (cgroup) 是 Linux 核心功能，用於階層式整理處理程序並為其分配系統資源。控制組廣泛用於執行容器執行期，並由 `systemd` 執行。

AL2 支援 cgroupv1，而 AL2023 支援 cgroupv2。如果執行容器化工作負載 (例如 使用以 AL2023 為基礎的 Amazon ECS AMIs 託管容器化工作負載 時)，這就值得注意。

雖然 AL2023 仍包含可讓系統使用 執行的程式碼cgroupv1，但這不是建議或支援的組態，並且會在未來的 Amazon Linux 主要版本中完全移除。

低階 Linux 核心介面和 systemd cgroup 委託文件現有大量的說明文件。

容器外部的常見使用案例是建立對可以使用的系統資源具有限制的`systemd`單位。如需詳細資訊，請參閱 systemd.resource-control。

`systemd` 計時器取代 `cron`

`cronie` 套件預設安裝在 AL2 AMI 上，為排誠定期任務的傳統 `crontab` 方式提供支援。在 AL2023 中， 預設不包含 `cronie` 。因此，預設`crontab`不再提供 的支援。

您可以選擇性安裝 `cronie` 套件以使用傳統 `cron` 工作。由於 `systemd` 提供的新增功能，建議您遷移至 `systemd` 計時器。

改善的工具鏈：binutils、 gcc和 glibc

AL2023 包含許多與 AL2 相同的核心套件。

我們已更新以下三個 AL2023 核心工具鏈套件。

套件名稱	AL2	AL2023
glibc	2.26	2.34
gcc	7.3	11.3
binutils	2.29	2.39

如需詳細資訊，請參閱核心工具鏈套件 glibc、gcc、binutils。

如需 C、 C++和Fortran語言執行時間的詳細資訊，包括更新的預設語言標準，請參閱 AL2023 中的 C、C++ 和 Fortran。

如需關於最佳化的詳細資訊，請參閱 效能與操作最佳化。

systemd 日誌取代 rsyslog

在 AL2023 中，日誌系統套件已從 AL2 變更。AL2023 預設不會安裝 `rsyslog`，因此預設無法在 AL2 中使用文字型日誌檔，例如 `/var/log/messages`。AL2023 的預設組態為 `systemd-journal`，並可使用 `journalctl` 檢查。雖然 `rsyslog` 是 AL2023 的選用套件，但建議使用基於 `systemd` 的新 `journalctl` 介面和相關套件。如需詳細資訊，請參閱 `journalctl` 手冊頁面。

下表涵蓋一些常用syslog命令的systmed journal同等項目。

AL2 syslog命令	AL2023 systemd journal對等
`[ec2-user ~]$ cat /var/log/messages`	`[ec2-user ~]$ journalctl`
`[ec2-user ~]$ tail -f /var/log/messages`	`[ec2-user ~]$ journalctl -f`
`[ec2-user ~]$ grep foo /var/log/messages`	`[ec2-user ~]$ journalctl | grep foo`

將套件相依性降至最低

Amazon Linux 2023 可將許多套件的相依性圖表降至最低，為應用程式提供更小的佔用空間。從 AL2 的顯著變更包括 curl-minimal和 gnupg-minimal套件，可大幅減少所需套件的數量，同時保留常用的功能。

主題

curl 和 libcurl 的套件變更

AL2023 將 curl 和 libcurl 套件的常用通訊協定和功能區分為 curl-minimal 和 libcurl-minimal。這可讓大多數使用者減少磁碟、記憶體和相依性佔用量，也是 AL2023 AMI 和容器的預設套件。

如果需要 curl 的完整功能 (例如 gopher:// 的支援)，請執行下列命令來安裝 curl-full 和 libcurl-full 套件。

```
$ dnf swap libcurl-minimal libcurl-full
```

```
$ dnf swap curl-minimal curl-full
```

GNU Privacy Guard (GNUPG)

AL2023 將 gnupg2 套件的最小和完整功能區分 gnupg2-minimal 和 gnupg2-full 套件。預設僅安裝 gnupg2-minimal 套件。此套件提供在 rpm 套件中驗證數位簽章所需的最小功能。

如需 gnupg2 的更多功能，例如從金鑰伺服器下載金鑰，請確定已安裝 gnupg2-full 套件。執行下列命令以交換 gnupg2-minimal 和 gnupg2-full。

```
$ dnf swap gnupg2-minimal gnupg2-full
```

Amazon Corretto 作為預設 JVM

AL2023 隨附 Amazon Corretto 作為預設 (也是唯一) Java 開發套件 (JDK)。AL2023 中的所有 Java 型套件都是使用 建置Amazon Corretto 17。

如果您要從 AL2 遷移，您可以將 AL2 上的同等OpenJDK版本順利轉換為 Amazon Corretto。

AWS CLIv2

AL2023 隨附第 2 AWS CLI版，而 AL2 隨附第 1 版AWS CLI。

UEFI 偏好和安全開機

預設情況下，在支援 UEFI 韌體的執行個體類型上使用 AL2023 AMI 啟動的任何執行個體都會在 UEFI 模式下啟動。這是透過將 Boot Mode AMI 參數設為 `uefi-preferred` 來完成。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的開機模式。

在支援 UEFI 安全開機的 Amazon EC2 執行個體類型上，可以在 Amazon Linux 2023 中啟用安全開機。如需詳細資訊，請參閱AL2023 上的 UEFI 安全開機。

SSH 伺服器預設組態變更

對於 AL2023 AMI，我們已變更隨發行版本產生的 `sshd` 主機金鑰類型。我們也已捨棄某些舊金鑰類型，以免在啟動時產生這些金鑰類型。使用者端必須支援 `rsa-sha2-256` 和 `rsa-sha2-512` 通訊協定，或是支援 `ssh-ed25519` 使用 `ed25519` 金鑰。`ssh-rsa` 簽章會預設停用。

此外，預設 `sshd_config` 檔案中的 AL2023 組態設定包含 `UseDNS=no`。這項新設定代表 DNS 減損不太可能防止您與執行個體建立 `ssh` 工作階段。此項取捨是 `authorized_keys` 檔案中的 `from=`*`hostname.domain`*`,`*`hostname.domain`* 行項目將無法解析。由於 `sshd` 不再嘗試解析 DNS 名稱，因此必須將每個逗號分隔 *`hostname.domain`* 值轉換為對應的 IP 地址。

如需詳細資訊，請參閱預設 SSH 伺服器組態。

從 AL2 的 AL2023 核心變更 AL2

AL2023 帶來 6.1 核心，以及許多組態變更，以進一步最佳化雲端的 Amazon Linux。對於大多數使用者而言，這些變更應該是完全透明的。

IPv4 TTL

IPv4 的 TTL 是透過 設定`sysctl`，預設值存在於 中`/etc/sysctl.d/00-defaults.conf`。此值可透過一般`sysctl`方法自訂。如需詳細資訊，請參閱 `sysctl`man頁面。

AL2 將net.ipv4.ip_default_ttl值設定為 255，而 AL2023 將其設定為 127。這使得 Amazon Linux 預設值與其他主要 Linux 發行版本一致。不建議在沒有證明需要的情況下變更此預設值。

重視安全的核心設定變更

CONFIG 選項	AL2/4.14/ aarch64	AL2/4.14/ x86_64	AL2/5.10/ aarch64	AL2/5.10/ x86_64	AL2023/6 1/ aarch64	AL2023/6 1/ x86_64	AL2023/6 12/ aarch64	AL2023/6 12/ x86_64
CONFIG_I G_ON_DA _CORRUP ON	n	y	n	y	y	y	y	y
CONFIG_I FAULT_MI P_MIN_AI R	4096	4096	4096	4096	65536	65536	65536	65536
CONFIG_I VMEM	n	y	n	y	n	n	n	n
CONFIG_I VPORT	n	y	n	y	n	n	n	n
CONFIG_I RTIFY_SC RCE	n	y	n	y	y	y	y	y
CONFIG_I RDENED_U ERCOPY_F LLBACK	N/A	N/A	y	y	N/A	N/A	N/A	N/A
CONFIG_ IT_ON_AI OC_DEFAU T_ON	N/A	N/A	n	n	n	n	n	n

CONFIG 選項	AL2/4.14/ aarch64	AL2/4.14/ x86_64	AL2/5.10/ aarch64	AL2/5.10/ x86_64	AL2023/6. 1/ aarch64	AL2023/6 1/ x86_64	AL2023/6. 12/ aarch64	AL2023/6. 12/ x86_64
CONFIG_I IT_ON_FI E_DEFAUI _ON	N/A	N/A	n	n	n	n	n	n
CONFIG_I MMU_DEF/ LT_DMA_S RICT	N/A	N/A	N/A	N/A	n	n	n	n
CONFIG_I ISC_AUTO OAD	y	y	y	y	n	n	n	n
CONFIG_S HED_CORI	N/A	N/A	N/A	N/A	N/A	y	N/A	y
CONFIG_S HED_STA _END_CHI K	n	y	n	y	y	y	y	y
CONFIG_S CURITY_I ESG_RES ICT	n	n	n	n	y	y	y	y
CONFIG_S CURITY_S LINUX_D ABLE	y	y	y	y	n	n	N/A	N/A

CONFIG 選項	AL2/4.14/ aarch64	AL2/4.14/ x86_64	AL2/5.10/ aarch64	AL2/5.10/ x86_64	AL2023/6. 1/ aarch64	AL2023/6 1/ x86_64	AL2023/6. 12/ aarch64	AL2023/6. 12/ x86_64
CONFIG_S UFFLE_PA E_ALLOCA OR	N/A	N/A	y	y	y	y	y	y
CONFIG_S AB_FREEL ST_HARDE ED	n	y	y	y	y	y	y	y
CONFIG_S AB_FREEL ST_RANDO	n	n	y	y	y	y	y	y

x86-64 專屬且重視安全的核心設定變更

CONFIG 選項	AL2/4.14/x86_64	AL2/5.10/x86_64	AL2023/6.1/ x86_64	AL2023/6.12/ x86_64
CONFIG_AM D_IOMMU	y	y	y	y
CONFIG_AM D_IOMMU_V2	m	m	y	N/A
CONFIG_RA NDOMIZE_M EMORY	N/A	y	y	y

aarch64 (ARM/Graviton) 專屬且重視安全的核心設定變更

CONFIG 選項	AL2/4.14/ aarch64	AL2/5.10/ aarch64	AL2023/6.1/ aarch64	AL2023/6.12/ aarch64
CONFIG_ARM64_PTR_AUTH	N/A	y	y	y
CONFIG_ARM64_PTR_AUTH_KERNEL	N/A	N/A	y	y
CONFIG_ARM64_SW_TTBR0_PAN	y	y	y	y

/dev/mem、**/dev/kmem** 和 **/dev/port**

Amazon Linux 2023 完全停用 /dev/mem、 和 /dev/port(CONFIG_DEVMEM 和 CONFIG_DEVPORT)，以 AL2 中已有的限制為基礎。

/dev/kmem 程式碼已在 5.13 核心中從 Linux 完全移除，而在 AL2 中停用時，現在不適用於 AL2023。

此選項是核心自我保護專案建議設定之一。

FORTIFY_SOURCE

AL2023 會在所有支援的架構CONFIG_FORTIFY_SOURCE上啟用 。此功能是安全強化功能。編譯器可以確定和驗證緩衝區大小時，此功能可以偵測常見字符串和記憶體函數中的緩衝區溢出。

此選項是核心自我保護專案建議設定之一。

Line Discipline 自動載入 (**CONFIG_LDISC_AUTOLOAD**)

AL2023 核心不會自動載入行紀律，例如使用 TIOCSETD 的軟體ioctl，除非請求來自具有 CAP_SYS_MODULE許可的程序。

此選項是核心自我保護專案建議設定之一。

dmesg 無權限使用者的存取權 (**CONFIG_SECURITY_DMESG_RESTRICT**)

根據預設，AL2023 不允許無權限使用者存取 dmesg。

此選項是核心自我保護專案建議設定之一。

SELinux **selinuxfs**停用

AL2023 會停用已棄用CONFIG_SECURITY_SELINUX_DISABLE的核心選項，這會啟用在載入政策之前停用 SELinux 的執行期方法。

此選項是核心自我保護專案建議設定之一。

其他核心設定變更

CONFIG 選項	AL2/4.14/ aarch64	AL2/4.14/ x86_64	AL2/5.10/ aarch64	AL2/5.10/ x86_64	AL2023/6 1/ aarch64	AL2023/6 1/ x86_64	AL2023/6 12/ aarch64	AL2023/6 12/ x86_64
CONFIG_I	100	250	100	250	100	100	100	100
CONFIG_I _CPUS	4096	8192	4096	8192	4096	8192	4096	8192
CONFIG_I NIC_ON_(PS	y	n	y	n	y	y	y	y
CONFIG_I NIC_ON_(PS_VALUI	1	0	1	0	1	1	1	1
CONFIG_I P	m	m	m	m	n	n	n	n
CONFIG_S IP	m	m	m	m	n	n	n	n

CONFIG 選項	AL2/4.14/ aarch64	AL2/4.14/ x86_64	AL2/5.10/ aarch64	AL2/5.10/ x86_64	AL2023/6 1/ aarch64	AL2023/6 1/ x86_64	AL2023/6 12/ aarch64	AL2023/6 12/ x86_64
CONFIG_ N_PV	N/A	y	N/A	n	N/A	n	N/A	n

CONFIG_HZ

x86-64 和 aarch64平台上的 AL2023 都CONFIG_HZ設為 100。

CONFIG_NR_CPUS

AL2023 會將 CONFIG_NR_CPUS 設為更接近 Amazon EC2 中 CPU 核心數量上限的數字。

意外錯誤

AL2023 核心在響起時會驚慌。此功能相當於在核心命令列以 oops=panic 啟動。

核心意外是核心偵測到內部錯誤的地方，這可能會影響系統的進一步可靠性。

PPP 和 SLIP 支援

AL2023 不支援 PPP 或 SLIP 通訊協定。

Xen PV 訪客支援

AL2023 不支援以 Xen PV 訪客身分執行。

核心檔案系統支援

AL2 中的核心將支援掛載的檔案系統中有幾項變更，以及核心將剖析的分割結構描述中的變更。

CONFIG 選項	AL2/4.14/ aarch64	AL2/4.14/ x86_64	AL2/5.10/ aarch64	AL2/5.10/ x86_64	AL2023/6 1/ aarch64	AL2023/6 1/ x86_64	AL2023/6 12/ aarch64	AL2023/6 12/ x86_64
CONFIG_ S_FS	n	m	n	m	n	n	n	n

CONFIG 選項	AL2/4.14/ aarch64	AL2/4.14/ x86_64	AL2/5.10/ aarch64	AL2/5.10/ x86_64	AL2023/6 1/ aarch64	AL2023/6 1/ x86_64	AL2023/6 12/ aarch64	AL2023/6 12/ x86_64
CONFIG_ _RXRPC	n	m	n	m	n	n	n	n
CONFIG_I D_DISKL EL	y	y	y	y	n	n	n	n
CONFIG_(AMFS	m	m	m	m	n	n	n	n
CONFIG_(AMFS_BL(KDEV	N/A	N/A	y	n	N/A	N/A	N/A	N/A
CONFIG_I _CLONE	N/A	N/A	n	n	n	n	n	n
CONFIG_I _ERA	m	n	m	n	n	n	n	n
CONFIG_I _INTEGR Y	n	m	n	m	m	m	m	m
CONFIG_I _LOG_WR ES	n	n	m	m	m	m	m	m
CONFIG_I _SWITCH	m	n	m	n	n	n	n	n
CONFIG_I _VERITY	m	n	m	n	m	m	m	m

CONFIG 選項	AL2/4.14/ aarch64	AL2/4.14/ x86_64	AL2/5.10/ aarch64	AL2/5.10/ x86_64	AL2023/6 1/ aarch64	AL2023/6 1/ x86_64	AL2023/6 12/ aarch64	AL2023/6 12/ x86_64
CONFIG_I RYPT_FS	n	m	n	m	n	n	n	n
CONFIG_I FAT_FS	N/A	N/A	m	m	m	m	m	m
CONFIG_I T2_FS	n	m	n	m	n	n	n	n
CONFIG_I T3_FS	n	m	n	m	n	n	n	n
CONFIG_(S2_FS	m	m	m	m	n	n	n	n
CONFIG_I SPLUS_F	n	m	n	m	n	n	n	n
CONFIG_I S_FS	n	m	n	m	n	n	n	n
CONFIG_ S_FS	n	n	n	n	n	n	n	n
CONFIG_I M_PARTI ON	n	y	n	y	n	n	n	n
CONFIG_I C_PARTI ON	n	y	n	y	n	n	n	n
CONFIG_I S_V2	n	m	n	m	n	n	n	n

CONFIG 選項	AL2/4.14/ aarch64	AL2/4.14/ x86_64	AL2/5.10/ aarch64	AL2/5.10/ x86_64	AL2023/6 1/ aarch64	AL2023/6 1/ x86_64	AL2023/6 12/ aarch64	AL2023/6 12/ x86_64
CONFIG_I FS_FS	n	m	n	n	n	n	n	n
CONFIG_I MFS_FS	n	m	n	m	n	n	n	n
CONFIG_S LARIS_X8 _PARTIT N	n	y	n	y	n	n	n	n
CONFIG_S UASHFS_Z TD	n	y	n	y	y	y	y	y
CONFIG_S N_PARTI ON	n	y	n	y	n	n	n	n

Andrew 檔案系統支援 (AFS)

核心在建置時，不再支援 `afs` 檔案系統。AL2 未隨附 的使用者空間支援`afs`。

cramfs 支援

核心在建置時，不再支援 `cramfs` 檔案系統。AL2023 中的後續版本是`squashfs`檔案系統。

BSD 磁碟標籤支援

核心在建置時，不再支援 BSD 磁碟標籤。如果需要讀取有 BSD 磁碟標籤的磁碟區，則可啟動各種 BSD。

裝置映射器變更

在 AL2023 核心中設定的 Device Mapper 目標有幾項變更。

eCryptFS 支援

ecryptfs 檔案系統已在 Amazon Linux 中棄用。的使用者空間元件ecryptfs存在於 AL1 中，在 AL2 中移除，AL2023 不再使用 ecryptfs支援建置核心。

exFAT

exFAT 檔案系統的支援已新增至 AL2 的 5.10 核心。使用 4.14 核心的 AL2 啟動時不存在。AL2023 會繼續支援exFAT檔案系統。

ext2、ext3 和 ext4 檔案系統

AL2023 隨附 CONFIG_EXT4_USE_FOR_EXT2選項，這表示ext4檔案系統程式碼將用於讀取舊版ext2檔案系統。

CONFIG_GFS2_FS

此核心不再使用配置 CONFIG_GFS2_FS 建置。

Apple Extended HFS 檔案系統支援 (HFS +)

在 AL2 x86-64 中，只有核心是使用 hfsplus 檔案系統支援建置的。AL2 5.15 核心不包含對任何架構的hfsplus支援。在 AL2023 中，我們在 Amazon Linux 中完成棄用hfsplus支援。

HFS 檔案系統支援

在 AL2 x86-64 中，只有核心是使用 hfs 檔案系統支援建置的。AL2 5.15 核心不包含對任何架構的hfs支援。在 AL2023 中，我們在 Amazon Linux 中完成取代hfs支援。

JFS 檔案系統支援

較舊的 AL2 x86-64 核心是透過jfs檔案系統支援所建置。AL2 5.15 核心不包含對任何架構的jfs支援。JFS 使用者空間隨附的 AL1 或 AL2。在 AL2023 中，我們在 Amazon Linux 中完成取代jfs支援。

上游 Linux 核心正在考慮移除 JFS。因此，如果您在JFS檔案系統上有資料，您應該將其遷移到另一個檔案系統。在 2024 年， JFS已從所有目前的 Amazon Linux 核心中移除。

Windows 邏輯磁碟管理員 （ 動態磁碟 ） 支援 (**CONFIG_LDM_PARTITION**)

AL2023 不再支援具有MS-DOS樣式分割區的 Windows 2000Windows XP、 或 Windows Vista 動態磁碟。此程式碼從未支援隨 推出的較新 GPT 型動態磁碟Windows Vista。

Macintosh 分割對應支援

AL2023 不再支援傳統 Macintosh 分割區映射。現代的 macOS 版本預設建立現代的 GPT 分割表，而不是這種舊類型。

NFSv2 支援

AL2023 不再支援 NFSv2，但繼續支援 NFSv3, NFSv4, NFSv4.1 和 NFSv4.2. 我們建議您遷移至 NFSv3 或更新版本。

NTFS (**CONFIG_NTFS_FS**)

從 AL2 的 5.10 核心開始，取代`ntfs`存取 Amazon Linux 上 NTFS 檔案系統的`ntfs3`程式碼。AL2023 不再包含`ntfs`程式碼，並且完全依賴`ntfs3`程式碼來存取 NTFS 檔案系統。

romfs 檔案系統

`squashfs` 檔案系統是 Amazon Linux `romfs` 檔案系統的後續系統，而 AL2023 核心的建置不再支援 `romfs`。

Solaris x86 硬碟分割格式

AL2023 不再支援 Solaris x86 硬碟分割區格式。

`squashfs` zstd 壓縮

AL2023 在所有支援的架構上新增了對zstd壓縮`squashfs`檔案系統的支援。

Sun 分割表支援

AL2023 不再包含對 Sun 分割區資料表格式 () 的支援`CONFIG_SUN_PARTITION`。

`/tmp` 現在是 **`tmpfs`**

與 Amazon Linux 2 相比，Amazon Linux 2023 對行為`/tmp`方式進行了變更。AL2 的預設組態是 `/tmp`和 `/var/tmp` 在根檔案系統上。Amazon Linux 2023 預設為使用 `tmpfs` 的 `/tmp`，限制為 RAM 的 50%，上限為 100 萬inodes。這些變更可讓 Amazon Linux 符合其他 Linux 發行版本的行為。

如需 AL2023 檔案系統配置的完整詳細資訊，請參閱 檔案系統配置一節`/var/tmp`中的 `/tmp`和 。

AMI 和容器映像變更

AMIs和容器中包含的套件有一些變更。

Amazon Linux 2023 推出 the section called “AL2023 最小容器映像”，並支援建置 the section called “建置裸機 AL2023 容器映像”。如需詳細資訊，請參閱在容器中使用 AL2023。

比較 Amazon Linux 2 與 Amazon 2023 AMI 安裝的套件

Amazon Linux 2 和 AL2023 標準 AMIs 上存在的 RPMs 比較。

套件	AL2 AMI	AL2023 AMI
acl	2.2.51	2.3.1
acpid	2.0.19	2.0.32
alternatives		1.15
amazon-chrony-config		4.3
amazon-ec2-net-utils		2.5.1
amazon-linux-extras	2.0.3	
amazon-linux-extras-yum-plugin	2.0.3	
amazon-linux-repo-s3		2023.6.20241031
amazon-linux-sb-keys		2023.1
amazon-rpm-config		228
amazon-ssm-agent	3.3.987.0	3.3.987.0
amd-ucode-firmware	20200421 (noarch)	20210208 (noarch)
at	3.1.13	3.1.23
attr	2.4.46	2.5.1

套件	AL2 AMI	AL2023 AMI
audit	2.8.1	3.0.6
audit-libs	2.8.1	3.0.6
authconfig	6.2.8	
aws-cfn-bootstrap	2.0	2.0
awscli	1.18.147	
awscli-2		2.15.30
basesystem	10.0	11
bash	4.2.46	5.2.15
bash-completion	2.1	2.11
bc	1.06.95	1.07.1
bind-export-libs	9.11.4	
bind-libs	9.11.4	9.18.28
bind-libs-lite	9.11.4	
bind-license	9.11.4	9.18.28
bind-utils	9.11.4	9.18.28
binutils	2.29.1	2.39
blktrace	1.0.5	
boost-date-time	1.53.0 (x86_64)	
boost-filesystem		1.75.0
boost-system	1.53.0 (x86_64)	1.75.0

套件	AL2 AMI	AL2023 AMI
boost-thread	1.53.0 (x86_64)	1.75.0
bridge-utils	1.5	
bzip2	1.0.6	1.0.8
bzip2-libs	1.0.6	1.0.8
ca-certificates	2023.2.68	2023.2.68
c-ares		1.19.1
checkpolicy		3.4
chkconfig	1.7.4	1.15
chrony	4.2	4.3
cloud-init	19.3	22.2.2
cloud-init-cfg-ec2		22.2.2
cloud-utils-growpart	0.31	0.31
coreutils	8.22	8.32
coreutils-common		8.32
cpio	2.12	2.13
cracklib	2.9.0	2.9.6
cracklib-dicts	2.9.0	2.9.6
cronie	1.4.11	
cronie-anacron	1.4.11	
crontabs	1.11	1.11

套件	AL2 AMI	AL2023 AMI
crypto-policies		20220428
crypto-policies-scripts		20220428
cryptsetup	1.7.4	2.6.1
cryptsetup-libs	1.7.4	2.6.1
curl	8.3.0	
curl-minimal		8.5.0
cyrus-sasl-lib	2.1.26	2.1.27
cyrus-sasl-plain	2.1.26	2.1.27
dbus	1.10.24	1.12.28
dbus-broker		32
dbus-common		1.12.28
dbus-libs	1.10.24	1.12.28
device-mapper	1.02.170	1.02.185
device-mapper-event	1.02.170	
device-mapper-event-libs	1.02.170	
device-mapper-libs	1.02.170	1.02.185
device-mapper-persistent-data	0.7.3	
dhclient	4.2.5	

套件	AL2 AMI	AL2023 AMI
dhcp-common	4.2.5	
dhcp-libs	4.2.5	
diffutils	3.3	3.8
dmidecode	3.2	
dmraid	1.0.0.rc16	
dmraid-events	1.0.0.rc16	
dnf		4.14.0
dnf-data		4.14.0
dnf-plugin-release-notification		1.2
dnf-plugins-core		4.3.0
dnf-plugin-support-info		1.2
dnf-utils		4.3.0
dosfstools	3.0.20	4.2
dracut	033	055
dracut-config-ec2	2.0	3.0
dracut-config-generic	033	055
dwz		0.14
dyninst	9.3.1 (x86_64)	10.2.1

套件	AL2 AMI	AL2023 AMI
e2fsprogs	1.42.9	1.46.5
e2fsprogs-libs	1.42.9	1.46.5
ec2-hibinit-agent	1.0.8	1.0.8
ec2-instance-connect	1.1	1.1
ec2-instance-connect-selinux	1.1	1.1
ec2-net-utils	1.7.3	
ec2-utils	1.2	2.2.0
ed	1.9	1.14.2
efibootmgr	15 (aarch64)	
efi-filesystem		5
efi-srpm-macros		5
efivar		38
efivar-libs	31 (aarch64)	38
elfutils-debuginfod-client		0.188
elfutils-default-yama-scope	0.176	0.188
elfutils-libelf	0.176	0.188
elfutils-libs	0.176	0.188
ethtool	4.8	5.15

套件	AL2 AMI	AL2023 AMI
expat	2.1.0	2.5.0
file	5.11	5.39
file-libs	5.11	5.39
filesystem	3.2	3.14
findutils	4.5.11	4.8.0
fipscheck	1.4.1	
fipscheck-lib	1.4.1	
fonts-srpm-macros		2.0.5
freetype	2.8	
fstrm		0.6.1
fuse-libs	2.9.2	2.9.9
gawk	4.0.2	5.1.0
gdbm	1.13	
gdbm-libs		1.19
gdisk	0.8.10	1.0.8
generic-logos	18.0.0	
GeoIP	1.5.0	
gettext	0.19.8.1	0.21
gettext-libs	0.19.8.1	0.21
ghc-srpm-macros		1.5.0

套件	AL2 AMI	AL2023 AMI
glib2	2.56.1	2.74.7
glibc	2.26	2.34
glibc-all-langpacks	2.26	2.34
glibc-common	2.26	2.34
glibc-gconv-extra		2.34
glibc-locale-source	2.26	2.34
glibc-minimal-langpack	2.26	
gmp	6.0.0	6.2.1
gnupg2	2.0.22	
gnupg2-minimal		2.3.7
gnutls		3.8.0
go-srpm-macros		3.2.0
gpgme	1.3.2	1.15.1
gpm-libs	1.20.7	1.20.7
grep	2.20	3.8
groff-base	1.22.2	1.22.4
grub2	2.06	
grub2-common	2.06	2.06
grub2-efi-aa64	2.06 (aarch64)	
grub2-efi-aa64-ec2	2.06 (aarch64)	2.06 (aarch64)

套件	AL2 AMI	AL2023 AMI
grub2-efi-aa64-modules	2.06 (noarch)	
grub2-efi-x64-ec2	2.06 (x86_64)	2.06 (x86_64)
grub2-pc	2.06 (x86_64)	
grub2-pc-modules	2.06 (noarch)	2.06
grub2-tools	2.06	2.06
grub2-tools-minimal	2.06	2.06
grubby	8.28	8.40
gssproxy	0.7.0	0.8.4
gzip	1.5	1.12
hardlink	1.3	
hibagent	1.1.0	
hostname	3.13	3.23
hunspell	1.3.2	1.7.0
hunspell-en	0.20121024	0.20140811.1
hunspell-en-GB	0.20121024	0.20140811.1
hunspell-en-US	0.20121024	0.20140811.1
hunspell-filesystem		1.7.0
hwdata	0.252	0.384
info	5.1	6.7
inih		49

套件	AL2 AMI	AL2023 AMI
initscripts	9.49.47	10.09
iproute	5.10.0	6.10.0
iptables	1.8.4	
iptables-libs	1.8.4	
iputils	20180629	20210202
irqbalance	1.7.0	1.9.0
jansson	2.10	2.14
jbigkit-libs	2.0	
jemalloc		5.2.1
jitterentropy		3.4.1
jq		1.7.1
json-c	0.11	0.14
kbd	1.15.5	2.4.0
kbd-legacy	1.15.5	
kbd-misc	1.15.5	2.4.0
kernel	5.10.228	6.1.112
kernel-libbpf		6.1.112
kernel-livepatch-repo-s3		2023.6.20241031
kernel-srpm-macros		1.0
kernel-tools	5.10.228	6.1.112

套件	AL2 AMI	AL2023 AMI
keyutils	1.5.8	1.6.3
keyutils-libs	1.5.8	1.6.3
kmod	25	29
kmod-libs	25	29
kpartx	0.4.9	
kpatch-runtime	0.9.4	0.9.7
krb5-libs	1.15.1	1.21.3
langtable	0.0.31	
langtable-data	0.0.31	
langtable-python	0.0.31	
less	458	608
libacl	2.2.51	2.3.1
libaio	0.3.109	0.3.111
libarchive		3.7.4
libargon2		20171227
libassuan	2.1.0	2.5.5
libattr	2.4.46	2.5.1
libbasicobjects	0.1.1	0.1.1
libblkid	2.30.2	2.37.4
libcap	2.54	2.48

套件	AL2 AMI	AL2023 AMI
libcap-ng	0.7.5	0.8.2
libcbor		0.7.0
libcollection	0.7.0	0.7.0
libcom_err	1.42.9	1.46.5
libcomps		0.1.20
libconfig	149	1.7.2
libcroco	0.6.12	
libcrypt	2.26	
libcurl	8.3.0	
libcurl-minimal		8.5.0
libdaemon	0.14	
libdb	5.3.21	5.3.28
libdb-utils	5.3.21	
libdhash		0.5.0
libdnf		0.69.0
libdrm	2.4.97	
libdwarf	20130207 (x86_64)	
libeconf		0.4.0
libedit	3.0	3.1
libestr	0.1.9	

套件	AL2 AMI	AL2023 AMI
libev		4.33
libevent	2.0.21	2.1.12
libfastjson	0.99.4	
libfdisk	2.30.2	2.37.4
libffi	3.0.13	3.4.4
libfido2		1.10.0
libgcc	7.3.1	11.4.1
libgcrypt	1.5.3	1.10.2
libgomp	7.3.1	11.4.1
libgpg-error	1.12	1.42
libibverbs		48.0
libicu	50.2	
libidn	1.28	
libidn2	2.3.0	2.3.2
libini_config	1.3.1	1.3.1
libjpeg-turbo	2.0.90	
libkcapi		1.4.0
libkcapi-hmaccalc		1.4.0
libldb		2.6.2
libmaxminddb		1.5.2

套件	AL2 AMI	AL2023 AMI
libmetalink	0.1.3	0.1.3
libmnl	1.0.3	1.0.4
libmodulemd		2.13.0
libmount	2.30.2	2.37.4
libnetfilter_conntrack	1.0.6	
libnfnetlink	1.0.1	
libnfsidmap	0.25	2.5.4
libnghttp2	1.41.0	1.59.0
libnl3	3.2.28	3.5.0
libnl3-cli	3.2.28	
libpath_utils	0.2.1	0.2.1
libpcap	1.5.3	1.10.1
libpciaccess	0.14 (x86_64)	
libpipeline	1.2.3	1.5.3
libpkgconf		1.8.0
libpng	1.5.13	
libpsl	0.21.5	0.21.1
libpwquality	1.2.3	1.4.4
libref_array	0.1.5	0.1.5
librepo		1.14.5

套件	AL2 AMI	AL2023 AMI
libreport-filesystem		2.15.2
libseccomp	2.5.2	2.5.3
libselinux	2.5	3.4
libselinux-utils	2.5	3.4
libsemanage	2.5	3.4
libsepol	2.5	3.4
libsigsegv		2.13
libsmartcols	2.30.2	2.37.4
libsolv		0.7.22
libss	1.42.9	1.46.5
libssh2	1.4.3	
libsss_certmap		2.9.4
libsss_idmap	1.16.5	2.9.4
libsss_nss_idmap	1.16.5	2.9.4
libsss_sudo		2.9.4
libstdc++	7.3.1	11.4.1
libstoragemgmt	1.6.1	1.9.4
libstoragemgmt-python	1.6.1	
libstoragemgmt-python-clibs	1.6.1	

套件	AL2 AMI	AL2023 AMI
libsysfs	2.1.0	
libtalloc		2.3.4
libtasn1	4.10	4.19.0
libtdb		1.4.7
libteam	1.27	
libtevent		0.13.0
libtextstyle		0.21
libtiff	4.0.3	
libtirpc	0.2.4	1.3.3
libunistring	0.9.3	0.9.10
libuser	0.60	0.63
libutempter	1.1.6	1.2.1
libuuid	2.30.2	2.37.4
libuv		1.47.0
libverto	0.2.5	0.3.2
libverto-libev		0.3.2
libverto-libevent	0.2.5	
libwebp	0.3.0	
libxcrypt		4.4.33
libxml2	2.9.1	2.10.4

套件	AL2 AMI	AL2023 AMI
libxml2-python	2.9.1	
libyaml	0.1.4	0.2.5
libzstd		1.5.5
linux-firmware-whence		20210208 (noarch)
lm_sensors-libs	3.4.0	3.6.0
lmdb-libs		0.9.29
logrotate	3.8.6	3.20.1
lsof	4.87	4.94.0
lua	5.1.4	
lua-libs		5.4.4
lua-srpm-macros		1
lvm2	2.02.187	
lvm2-libs	2.02.187	
lz4	1.7.5	
lz4-libs		1.9.4
make	3.82	
man-db	2.6.3	2.9.3
man-pages	3.53	5.10
man-pages-overrides	7.5.2	
mariadb-libs	5.5.68	

套件	AL2 AMI	AL2023 AMI
mdadm	4.0	
microcode_ctl	2.1 (x86_64)	2.1 (x86_64)
mlocate	0.26	
mpfr		4.1.0
mtr	0.92	
nano	2.9.8	5.8
ncurses	6.0	6.2
ncurses-base	6.0	6.2
ncurses-libs	6.0	6.2
nettle	2.7.1	3.8
net-tools	2.0	2.0
newt	0.52.15	0.52.21
newt-python	0.52.15	
nfs-utils	1.3.0	2.5.4
npth		1.6
nspr	4.35.0	4.35.0
nss	3.90.0	3.90.0
nss-pem	1.0.3	
nss-softokn	3.90.0	3.90.0
nss-softokn-freebl	3.90.0	3.90.0

套件	AL2 AMI	AL2023 AMI
nss-sysinit	3.90.0	3.90.0
nss-tools	3.90.0	
nss-util	3.90.0	3.90.0
ntsysv	1.7.4	1.15
numactl-libs	2.0.9	2.0.14
ocaml-srpm-macros		6
oniguruma		6.9.7.1
openblas-srpm-macros		2
openldap	2.4.44	2.4.57
openssh	7.4p1	8.7p1
openssh-clients	7.4p1	8.7p1
openssh-server	7.4p1	8.7p1
openssl	1.0.2k	3.0.8
openssl-libs	1.0.2k	3.0.8
openssl-pkcs11		0.4.12
os-prober	1.58	1.77
p11-kit	0.23.22	0.24.1
p11-kit-trust	0.23.22	0.24.1
package-notes-srpm-macros		0.4
pam	1.1.8	1.5.1

套件	AL2 AMI	AL2023 AMI
parted	3.1	3.4
passwd	0.79	0.80
pciutils	3.5.1	3.7.0
pciutils-libs	3.5.1	3.7.0
pcre	8.32	
pcre2	10.23	10.40
pcre2-syntax		10.40
perl	5.16.3	
perl-Carp	1.26	1.50
perl-Class-Struct		0.66
perl-constant	1.27	1.33
perl-DynaLoader		1.47
perl-Encode	2.51	3.15
perl-Errno		1.30
perl-Exporter	5.68	5.74
perl-Fcntl		1.13
perl-File-Basename		2.85
perl-File-Path	2.09	2.18
perl-File-stat		1.09
perl-File-Temp	0.23.01	0.231.100

套件	AL2 AMI	AL2023 AMI
perl-Filter	1.49	
perl-Getopt-Long	2.40	2.52
perl-Getopt-Std		1.12
perl-HTTP-Tiny	0.033	0.078
perl-if		0.60.800
perl-interpreter		5.32.1
perl-IO		1.43
perl-IPC-Open3		1.21
perl-libs	5.16.3	5.32.1
perl-macros	5.16.3	
perl-MIME-Base64		3.16
perl-mro		1.23
perl-overload		1.31
perl-overloading		0.02
perl-parent	0.225	0.238
perl-PathTools	3.40	3.78
perl-Pod-Escapes	1.04	1.07
perl-podlators	2.5.1	4.14
perl-Pod-Perldoc	3.20	3.28.01
perl-Pod-Simple	3.28	3.42

套件	AL2 AMI	AL2023 AMI
perl-Pod-Usage	1.63	2.01
perl-POSIX		1.94
perl-Scalar-List-Utils	1.27	1.56
perl-SelectSaver		1.02
perl-Socket	2.010	2.032
perl-srpm-macros		1
perl-Storable	2.45	3.21
perl-subs		1.03
perl-Symbol		1.08
perl-Term-ANSIColor		5.01
perl-Term-Cap		1.17
perl-Text-ParseWords	3.29	3.30
perl-Text-Tabs+Wrap		2021.0726
perl-threads	1.87	
perl-threads-shared	1.43	
perl-Time-HiRes	1.9725	
perl-Time-Local	1.2300	1.300
perl-vars		1.05
pinentry	0.8.1	
pkgconf		1.8.0

套件	AL2 AMI	AL2023 AMI
pkgconfig	0.27.1	
pkgconf-m4		1.8.0
pkgconf-pkg-config		1.8.0
plymouth	0.8.9	
plymouth-core-libs	0.8.9	
plymouth-scripts	0.8.9	
pm-utils	1.4.1	
policycoreutils	2.5	3.4
policycoreutils-python-utils		3.4
popt	1.13	1.18
postfix	2.10.1	
procps-ng	3.3.10	3.3.17
protobuf-c		1.4.1
psacct	6.6.1	6.6.4
psmisc	22.20	23.4
pth	2.0.7	
publicsuffix-list-dafsa	20240208	20240212
pygpgme	0.3	
pyliblzma	0.5.3	

套件	AL2 AMI	AL2023 AMI
pystache	0.5.3	
python	2.7.18	
python2-botocore	1.18.6	
python2-colorama	0.3.9	
python2-cryptography	1.7.2	
python2-dateutil	2.6.1	
python2-futures	3.0.5	
python2-jmespath	0.9.3	
python2-jsonschema	2.5.1	
python2-oauthlib	2.0.1	
python2-pyasn1	0.1.9	
python2-rpm	4.11.3	
python2-rsa	3.4.1	
python2-s3transfer	0.3.3	
python2-setuptools	41.2.0	
python2-six	1.11.0	
python3	3.7.16	3.9.16
python3-attrs		20.3.0
python3-audit		3.0.6
python3-awscrt		0.19.19

套件	AL2 AMI	AL2023 AMI
python3-babel		2.9.1
python3-cffi		1.14.5
python3-chardet		4.0.0
python3-colorama		0.4.4
python3-configobj		5.0.6
python3-cryptography		36.0.1
python3-daemon	2.2.3	2.3.0
python3-dateutil		2.8.1
python3-dbus		1.2.18
python3-distro		1.5.0
python3-dnf		4.14.0
python3-dnf-plugins-core		4.3.0
python3-docutils	0.14	0.16
python3-gpg		1.15.1
python3-hawkey		0.69.0
python3-idna		2.10
python3-jinja2		2.11.3
python3-jmespath		0.10.0
python3-jsonpatch		1.21
python3-jsonpointer		2.0

套件	AL2 AMI	AL2023 AMI
python3-jsonschema		3.2.0
python3-libcomps		0.1.20
python3-libdnf		0.69.0
python3-libs	3.7.16	3.9.16
python3-libselinux		3.4
python3-libsemanage		3.4
python3-libstorage mgmt		1.9.4
python3-lockfile	0.11.0	0.12.2
python3-markupsafe		1.1.1
python3-netifaces		0.10.6
python3-oauthlib		3.0.2
python3-pip	20.2.2	
python3-pip-wheel		21.3.1
python3-ply		3.11
python3-policycore utils		3.4
python3-prettytable		0.7.2
python3-prompt-too lkit		3.0.24
python3-pycparser		2.20

套件	AL2 AMI	AL2023 AMI
python3-pyrsistent		0.17.3
python3-pyserial		3.4
python3-pysocks		1.7.1
python3-pystache	0.5.4	
python3-pytz		2022.7.1
python3-pyyaml		5.4.1
python3-requests		2.25.1
python3-rpm		4.16.1.3
python3-ruamel-yaml		0.16.6
python3-ruamel-yaml-clib		0.1.2
python3-setools		4.4.1
python3-setuptools	49.1.3	59.6.0
python3-setuptools-wheel		59.6.0
python3-simplejson	3.2.0	
python3-six		1.15.0
python3-systemd		235
python3-urllib3		1.25.10
python3-wcwidth		0.2.5
python-babel	0.9.6	

套件	AL2 AMI	AL2023 AMI
python-backports	1.0	
python-backports-ssl_match_hostname	3.5.0.1	
python-cffi	1.6.0	
python-chardet	2.2.1	
python-chevron		0.13.1
python-configobj	4.7.2	
python-daemon	1.6	
python-devel	2.7.18	
python-docutils	0.12	
python-enum34	1.0.4	
python-idna	2.4	
python-iniparse	0.4	
python-ipaddress	1.0.16	
python-jinja2	2.7.2	
python-jsonpatch	1.2	
python-jsonpointer	1.9	
python-jwcrypto	0.4.2	
python-kitchen	1.1.1	
python-libs	2.7.18	
python-lockfile	0.9.1	

套件	AL2 AMI	AL2023 AMI
python-markupsafe	0.11	
python-pillow	2.0.0	
python-ply	3.4	
python-pycparser	2.14	
python-pycurl	7.19.0	
python-repoze-lru	0.4	
python-requests	2.6.0	
python-simplejson	3.2.0	
python-srpm-macros		3.9
python-urlgrabber	3.10	
python-urllib3	1.25.9	
pyxattr	0.5.1	
PyYAML	3.10	
qrencode-libs	3.4.1	
quota	4.01	4.06
quota-nls	4.01	4.06
rdate	1.4	
readline	6.2	8.1
rng-tools	6.8	6.14
rootfiles	8.1	8.1

套件	AL2 AMI	AL2023 AMI
rpcbind	0.2.0	1.2.6
rpm	4.11.3	4.16.1.3
rpm-build-libs	4.11.3	4.16.1.3
rpm-libs	4.11.3	4.16.1.3
rpm-plugin-selinux		4.16.1.3
rpm-plugin-systemd-inhibit	4.11.3	4.16.1.3
rpm-sign-libs		4.16.1.3
rsync	3.1.2	3.2.6
rsyslog	8.24.0	
rust-srpm-macros		21
sbsigntools		0.9.4
scl-utils	20130529	
screen	4.1.0	4.8.0
sed	4.2.2	4.8
selinux-policy	3.13.1	38.1.45
selinux-policy-targeted	3.13.1	38.1.45
setserial	2.17	
setup	2.8.71	2.13.7
setuptool	1.19.11	

套件	AL2 AMI	AL2023 AMI
sgpio	1.2.0.10	
shadow-utils	4.1.5.1	4.9
shared-mime-info	1.8	
slang	2.2.4	2.3.2
sqlite	3.7.17	
sqlite-libs		3.40.0
sssd-client	1.16.5	2.9.4
sssd-common		2.9.4
sssd-kcm		2.9.4
sssd-nfs-idmap		2.9.4
strace	4.26	6.8
sudo	1.8.23	1.9.15
sysctl-defaults	1.0	1.0
sysstat	10.1.5	12.5.6
systemd	219	252.23
systemd-libs	219	252.23
systemd-networkd		252.23
systemd-pam		252.23
systemd-resolved		252.23
systemd-sysv	219	

套件	AL2 AMI	AL2023 AMI
systemd-udev		252.23
system-release	2	2023.6.20241031
systemtap-runtime	4.5	4.8
sysvinit-tools	2.88	
tar	1.26	1.34
tbb		2020.3
tcp_wrappers	7.6	
tcp_wrappers-libs	7.6	
tcpdump	4.9.2	4.99.1
tcsh	6.18.01	6.24.07
teamd	1.27	
time	1.7	1.9
traceroute	2.0.22	2.1.3
tzdata	2024a	2024a
unzip	6.0	6.0
update-motd	1.1.2	2.2
usermode	1.111	
userspace-rcu		0.12.1
ustr	1.0.4	
util-linux	2.30.2	2.37.4

套件	AL2 AMI	AL2023 AMI
util-linux-core		2.37.4
vim-common	9.0.2153	9.0.2153
vim-data	9.0.2153	9.0.2153
vim-enhanced	9.0.2153	9.0.2153
vim-filesystem	9.0.2153	9.0.2153
vim-minimal	9.0.2153	9.0.2153
virt-what	1.18	
wget	1.14	1.21.3
which	2.20	2.21
words	3.0	3.0
xfsdump	3.1.8	3.1.11
xfsprogs	5.0.0	5.18.0
xxd	9.0.2153	9.0.2153
xxhash-libs		0.8.0
xz	5.2.2	5.2.5
xz-libs	5.2.2	5.2.5
yajl	2.0.4	
yum	3.4.3	4.14.0
yum-langpacks	0.4.2	
yum-metadata-parser	1.1.4	

套件	AL2 AMI	AL2023 AMI
yum-plugin-priorities	1.1.31	
yum-utils	1.1.31	
zip	3.0	3.0
zlib	1.2.7	1.2.11
zram-generator		1.1.2
zram-generator-defaults		1.1.2
zstd		1.5.5

比較 Amazon Linux 2 與 Amazon 2023 最小 AMI 安裝的套件

Amazon Linux 2 和 AL2023 最小 AMIs 上存在的 RPMs 比較。

套件	AL2 最小	AL2023 最小
acl	2.2.51	
alternatives		1.15
amazon-chrony-config		4.3
amazon-ec2-net-utils		2.5.1
amazon-linux-extras	2.0.3	
amazon-linux-repo-s3		2023.6.20241031
amazon-linux-sb-keys		2023.1
amd-ucode-firmware	20200421 (noarch)	20210208 (noarch)

套件	AL2 最小	AL2023 最小
audit	2.8.1	3.0.6
audit-libs	2.8.1	3.0.6
authconfig	6.2.8	
awscli-2		2.15.30
basesystem	10.0	11
bash	4.2.46	5.2.15
bind-export-libs	9.11.4	
bzip2-libs	1.0.6	1.0.8
ca-certificates	2023.2.68	2023.2.68
checkpolicy		3.4
chkconfig	1.7.4	
chrony	4.2	4.3
cloud-init	19.3	22.2.2
cloud-init-cfg-ec2		22.2.2
cloud-utils-growpart	0.31	0.31
coreutils	8.22	8.32
coreutils-common		8.32
cpio	2.12	2.13
cracklib	2.9.0	2.9.6
cracklib-dicts	2.9.0	2.9.6

套件	AL2 最小	AL2023 最小
cronie	1.4.11	
cronie-anacron	1.4.11	
crontabs	1.11	
crypto-policies		20220428
cryptsetup-libs	1.7.4	2.6.1
curl	8.3.0	
curl-minimal		8.5.0
cyrus-sasl-lib	2.1.26	2.1.27
dbus	1.10.24	1.12.28
dbus-broker		32
dbus-common		1.12.28
dbus-libs	1.10.24	1.12.28
device-mapper	1.02.170	1.02.185
device-mapper-libs	1.02.170	1.02.185
dhclient	4.2.5	
dhcp-common	4.2.5	
dhcp-libs	4.2.5	
diffutils	3.3	3.8
dnf		4.14.0
dnf-data		4.14.0

套件	AL2 最小	AL2023 最小
dnf-plugin-release-notification		1.2
dnf-plugins-core		4.3.0
dnf-plugin-support-info		1.2
dracut	033	055
dracut-config-ec2	2.0	3.0
dracut-config-generic	033	055
e2fsprogs	1.42.9	1.46.5
e2fsprogs-libs	1.42.9	1.46.5
ec2-utils	1.2	2.2.0
efibootmgr	15 (aarch64)	
efi-filesystem		5
efivar		38
efivar-libs	31 (aarch64)	38
elfutils-default-yama-scope	0.176	0.188
elfutils-libelf	0.176	0.188
elfutils-libs	0.176	0.188
expat	2.1.0	2.5.0
file	5.11	5.39

套件	AL2 最小	AL2023 最小
file-libs	5.11	5.39
filesystem	3.2	3.14
findutils	4.5.11	4.8.0
fipscheck	1.4.1	
fipscheck-lib	1.4.1	
freetype	2.8	
fuse-libs	2.9.2	2.9.9
gawk	4.0.2	5.1.0
gdbm	1.13	
gdbm-libs		1.19
gdisk	0.8.10	1.0.8
gettext	0.19.8.1	0.21
gettext-libs	0.19.8.1	0.21
glib2	2.56.1	2.74.7
glibc	2.26	2.34
glibc-all-langpacks	2.26	2.34
glibc-common	2.26	2.34
glibc-locale-source	2.26	2.34
glibc-minimal-langpack	2.26	
gmp	6.0.0	6.2.1

套件	AL2 最小	AL2023 最小
gnupg2	2.0.22	
gnupg2-minimal		2.3.7
gnutls		3.8.0
gpgme	1.3.2	1.15.1
grep	2.20	3.8
groff-base	1.22.2	1.22.4
grub2	2.06	
grub2-common	2.06	2.06
grub2-efi-aa64	2.06 (aarch64)	
grub2-efi-aa64-ec2	2.06 (aarch64)	2.06 (aarch64)
grub2-efi-aa64-modules	2.06 (noarch)	
grub2-efi-x64-ec2	2.06 (x86_64)	2.06 (x86_64)
grub2-pc	2.06 (x86_64)	
grub2-pc-modules	2.06 (noarch)	2.06
grub2-tools	2.06	2.06
grub2-tools-minimal	2.06	2.06
grubby	8.28	8.40
gzip	1.5	1.12
hardlink	1.3	
hostname	3.13	3.23

套件	AL2 最小	AL2023 最小
hwdata		0.384
info	5.1	
inih		49
initscripts	9.49.47	10.09
iproute	5.10.0	6.10.0
iptables	1.8.4	
iptables-libs	1.8.4	
iputils	20180629	20210202
irqbalance	1.7.0	1.9.0
jansson		2.14
jitterentropy		3.4.1
jq		1.7.1
json-c		0.14
kbd		2.4.0
kbd-misc		2.4.0
kernel	4.14.355	6.1.112
kernel-libbpf		6.1.112
kernel-livepatch-repo-s3		2023.6.20241031
keyutils-libs	1.5.8	1.6.3
kmod	25	29

套件	AL2 最小	AL2023 最小
kmod-libs	25	29
kpartx	0.4.9	
krb5-libs	1.15.1	1.21.3
less	458	608
libacl	2.2.51	2.3.1
libarchive		3.7.4
libargon2		20171227
libassuan	2.1.0	2.5.5
libattr	2.4.46	2.5.1
libblkid	2.30.2	2.37.4
libcap	2.54	2.48
libcap-ng	0.7.5	0.8.2
libcbor		0.7.0
libcom_err	1.42.9	1.46.5
libcomps		0.1.20
libcroco	0.6.12	
libcrypt	2.26	
libcurl	8.3.0	
libcurl-minimal		8.5.0
libdb	5.3.21	5.3.28

套件	AL2 最小	AL2023 最小
libdb-utils	5.3.21	
libdnf		0.69.0
libeconf		0.4.0
libedit	3.0	3.1
libestr	0.1.9	
libfastjson	0.99.4	
libfdisk	2.30.2	2.37.4
libffi	3.0.13	3.4.4
libfido2		1.10.0
libgcc	7.3.1	11.4.1
libgcrypt	1.5.3	1.10.2
libgomp	7.3.1	11.4.1
libgpg-error	1.12	1.42
libicu	50.2	
libidn	1.28	
libidn2	2.3.0	2.3.2
libkcapi		1.4.0
libkcapi-hmaccalc		1.4.0
libmetalink	0.1.3	
libmnl	1.0.3	1.0.4

套件	AL2 最小	AL2023 最小
libmodulemd		2.13.0
libmount	2.30.2	2.37.4
libnetfilter_conntrack	1.0.6	
libnfnetlink	1.0.1	
libnghttp2	1.41.0	1.59.0
libpcap	1.5.3	
libpipeline	1.2.3	1.5.3
libpng	1.5.13	
libpsl	0.21.5	0.21.1
libpwquality	1.2.3	1.4.4
librepo		1.14.5
libreport-filesystem		2.15.2
libseccomp	2.5.2	2.5.3
libselinux	2.5	3.4
libselinux-utils	2.5	3.4
libsemanage	2.5	3.4
libsepol	2.5	3.4
libsigsegv		2.13
libsmartcols	2.30.2	2.37.4
libsolv		0.7.22

套件	AL2 最小	AL2023 最小
libss	1.42.9	1.46.5
libssh2	1.4.3	
libstdc++	7.3.1	11.4.1
libsysfs	2.1.0	
libtasn1	4.10	4.19.0
libtextstyle		0.21
libunistring	0.9.3	0.9.10
libuser	0.60	0.63
libutempter	1.1.6	1.2.1
libuuid	2.30.2	2.37.4
libverto	0.2.5	0.3.2
libxcrypt		4.4.33
libxml2	2.9.1	2.10.4
libyaml	0.1.4	0.2.5
libzstd		1.5.5
linux-firmware-whence		20210208 (noarch)
logrotate	3.8.6	3.20.1
lua	5.1.4	
lua-libs		5.4.4
lz4	1.7.5	

套件	AL2 最小	AL2023 最小
lz4-libs		1.9.4
make	3.82	
man-db	2.6.3	2.9.3
mariadb-libs	5.5.68	
microcode_ctl	2.1 (x86_64)	2.1 (x86_64)
mpfr		4.1.0
ncurses	6.0	6.2
ncurses-base	6.0	6.2
ncurses-libs	6.0	6.2
nettle	2.7.1	3.8
net-tools	2.0	2.0
newt	0.52.15	
newt-python	0.52.15	
npth		1.6
nspr	4.35.0	
nss	3.90.0	
nss-pem	1.0.3	
nss-softokn	3.90.0	
nss-softokn-freebl	3.90.0	
nss-sysinit	3.90.0	

套件	AL2 最小	AL2023 最小
nss-tools	3.90.0	
nss-util	3.90.0	
numactl-libs	2.0.9	2.0.14
oniguruma		6.9.7.1
openldap	2.4.44	2.4.57
openssh	7.4p1	8.7p1
openssh-clients	7.4p1	8.7p1
openssh-server	7.4p1	8.7p1
openssl	1.0.2k	3.0.8
openssl-libs	1.0.2k	3.0.8
openssl-pkcs11		0.4.12
os-prober	1.58	1.77
p11-kit	0.23.22	0.24.1
p11-kit-trust	0.23.22	0.24.1
pam	1.1.8	1.5.1
passwd	0.79	0.80
pciutils		3.7.0
pciutils-libs		3.7.0
pcre	8.32	
pcre2	10.23	10.40

套件	AL2 最小	AL2023 最小
pcre2-syntax		10.40
pinentry	0.8.1	
pkgconfig	0.27.1	
policycoreutils	2.5	3.4
popt	1.13	1.18
postfix	2.10.1	
procps-ng	3.3.10	3.3.17
psmisc	22.20	23.4
pth	2.0.7	
publicsuffix-list-dafsa	20240208	20240212
pygpgme	0.3	
pyliblzma	0.5.3	
python	2.7.18	
python2-cryptography	1.7.2	
python2-jsonschema	2.5.1	
python2-oauthlib	2.0.1	
python2-pyasn1	0.1.9	
python2-rpm	4.11.3	
python2-setuptools	41.2.0	
python2-six	1.11.0	

套件	AL2 最小	AL2023 最小
python3		3.9.16
python3-attrs		20.3.0
python3-audit		3.0.6
python3-awscrt		2019 年 19 月 0 日
python3-babel		2.9.1
python3-cffi		1.14.5
python3-chardet		4.0.0
python3-colorama		0.4.4
python3-configobj		5.0.6
python3-cryptography		36.0.1
python3-dateutil		2.8.1
python3-dbus		1.2.18
python3-distro		1.5.0
python3-dnf		4.14.0
python3-dnf-plugins-core		4.3.0
python3-docutils		0.16
python3-gpg		1.15.1
python3-hawkey		0.69.0
python3-idna		2.10
python3-jinja2		2.11.3

套件	AL2 最小	AL2023 最小
python3-jmespath		0.10.0
python3-jsonpatch		1.21
python3-jsonpointer		2.0
python3-jsonschema		3.2.0
python3-libcomps		0.1.20
python3-libdnf		0.69.0
python3-libs		3.9.16
python3-libselinux		3.4
python3-libsemanage		3.4
python3-markupsafe		1.1.1
python3-netifaces		0.10.6
python3-oauthlib		3.0.2
python3-pip-wheel		21.3.1
python3-ply		3.11
python3-policycoreutils		3.4
python3-prettytable		0.7.2
python3-prompt-toolkit		3.0.24
python3-pycparser		2.20
python3-pyrsistent		0.17.3

套件	AL2 最小	AL2023 最小
python3-pyserial		3.4
python3-pysocks		1.7.1
python3-pytz		2022.7.1
python3-pyyaml		5.4.1
python3-requests		2.25.1
python3-rpm		4.16.1.3
python3-ruamel-yaml		0.16.6
python3-ruamel-yaml-clib		0.1.2
python3-setools		4.4.1
python3-setuptools		59.6.0
python3-setuptools-wheel		59.6.0
python3-six		1.15.0
python3-systemd		235
python3-urllib3		1.25.10
python3-wcwidth		0.2.5
python-babel	0.9.6	
python-backports	1.0	
python-backports-ssl_match_hostname	3.5.0.1	

套件	AL2 最小	AL2023 最小
python-cffi	1.6.0	
python-chardet	2.2.1	
python-configobj	4.7.2	
python-devel	2.7.18	
python-enum34	1.0.4	
python-idna	2.4	
python-iniparse	0.4	
python-ipaddress	1.0.16	
python-jinja2	2.7.2	
python-jsonpatch	1.2	
python-jsonpointer	1.9	
python-jwcrypto	0.4.2	
python-libs	2.7.18	
python-markupsafe	0.11	
python-ply	3.4	
python-pycparser	2.14	
python-pycurl	7.19.0	
python-repoze-lru	0.4	
python-requests	2.6.0	
python-urlgrabber	3.10	

套件	AL2 最小	AL2023 最小
python-urllib3	1.25.9	
pyxattr	0.5.1	
PyYAML	3.10	
qrencode-libs	3.4.1	
readline	6.2	8.1
rng-tools	6.8	6.14
rootfiles	8.1	8.1
rpm	4.11.3	4.16.1.3
rpm-build-libs	4.11.3	4.16.1.3
rpm-libs	4.11.3	4.16.1.3
rpm-plugin-selinux		4.16.1.3
rpm-plugin-systemd-inhibit	4.11.3	4.16.1.3
rpm-sign-libs		4.16.1.3
rsyslog	8.24.0	
sbsigntools		0.9.4
sed	4.2.2	4.8
selinux-policy	3.13.1	38.1.45
selinux-policy-targeted	3.13.1	38.1.45
setup	2.8.71	2.13.7

套件	AL2 最小	AL2023 最小
shadow-utils	4.1.5.1	4.9
shared-mime-info	1.8	
slang	2.2.4	
sqlite	3.7.17	
sqlite-libs		3.40.0
sudo	1.8.23	1.9.15
sysctl-defaults	1.0	1.0
systemd	219	252.23
systemd-libs	219	252.23
systemd-networkd		252.23
systemd-pam		252.23
systemd-resolved		252.23
systemd-sysv	219	
systemd-udev		252.23
system-release	2	2023.6.20241031
sysvinit-tools	2.88	
tar	1.26	1.34
tcp_wrappers-libs	7.6	
tzdata	2024a	2024a
update-motd	1.1.2	2.2

套件	AL2 最小	AL2023 最小
userspace-rcu		0.12.1
ustr	1.0.4	
util-linux	2.30.2	2.37.4
util-linux-core		2.37.4
vim-data	9.0.2153	9.0.2153
vim-minimal	9.0.2153	9.0.2153
which	2.20	2.21
xfsprogs	5.0.0	5.18.0
xz	5.2.2	5.2.5
xz-libs	5.2.2	5.2.5
yum	3.4.3	4.14.0
yum-metadata-parser	1.1.4	
yum-plugin-priorities	1.1.31	
zlib	1.2.7	1.2.11
zram-generator		1.1.2
zram-generator-defaults		1.1.2
zstd		1.5.5

比較 Amazon Linux 2 與 Amazon 2023 基本容器映像安裝的套件

Amazon Linux 2 和 AL2023 基礎容器映像上存在的 RPMs 比較。

套件	AL2 容器	AL2023 容器
alternatives		1.15
amazon-linux-extras	2.0.3	
amazon-linux-repo-cdn		2023.6.20241031
audit-libs		3.0.6
basesystem	10.0	11
bash	4.2.46	5.2.15
bzip2-libs	1.0.6	1.0.8
ca-certificates	2023.2.68	2023.2.68
chkconfig	1.7.4	
coreutils	8.22	
coreutils-single		8.32
cpio	2.12	
crypto-policies		20220428
curl	8.3.0	
curl-minimal		8.5.0
cyrus-sasl-lib	2.1.26	
diffutils	3.3	

套件	AL2 容器	AL2023 容器
dnf		4.14.0
dnf-data		4.14.0
elfutils-default-yama-scope		0.188
elfutils-libelf	0.176	0.188
elfutils-libs		0.188
expat	2.1.0	2.5.0
file-libs	5.11	5.39
filesystem	3.2	3.14
findutils	4.5.11	
gawk	4.0.2	5.1.0
gdbm	1.13	
gdbm-libs		1.19
glib2	2.56.1	2.74.7
glibc	2.26	2.34
glibc-common	2.26	2.34
glibc-langpack-en	2.26	
glibc-minimal-langpack	2.26	2.34
gmp	6.0.0	6.2.1
gnupg2	2.0.22	

套件	AL2 容器	AL2023 容器
gnupg2-minimal		2.3.7
gpgme	1.3.2	1.15.1
grep	2.20	3.8
info	5.1	
json-c		0.14
keyutils-libs	1.5.8	1.6.3
krb5-libs	1.15.1	1.21.3
libacl	2.2.51	2.3.1
libarchive		3.7.4
libassuan	2.1.0	2.5.5
libattr	2.4.46	2.5.1
libblkid	2.30.2	2.37.4
libcap	2.54	2.48
libcap-ng		0.8.2
libcom_err	1.42.9	1.46.5
libcomps		0.1.20
libcrypt	2.26	
libcurl	8.3.0	
libcurl-minimal		8.5.0
libdb	5.3.21	

套件	AL2 容器	AL2023 容器
libdb-utils	5.3.21	
libdnf		0.69.0
libffi	3.0.13	3.4.4
libgcc	7.3.1	11.4.1
libgcrypt	1.5.3	1.10.2
libgomp		11.4.1
libgpg-error	1.12	1.42
libidn2	2.3.0	2.3.2
libmetalink	0.1.3	
libmodulemd		2.13.0
libmount	2.30.2	2.37.4
libnghttp2	1.41.0	1.59.0
libpsl	0.21.5	0.21.1
librepo		1.14.5
libreport-filesystem		2.15.2
libselinux	2.5	3.4
libsepol	2.5	3.4
libsigsegv		2.13
libsmartcols		2.37.4
libsolv		0.7.22

套件	AL2 容器	AL2023 容器
libssh2	1.4.3	
libstdc++	7.3.1	11.4.1
libtasn1	4.10	4.19.0
libunistring	0.9.3	0.9.10
libuuid	2.30.2	2.37.4
libverto	0.2.5	0.3.2
libxcrypt		4.4.33
libxml2	2.9.1	2.10.4
libyaml		0.2.5
libzstd		1.5.5
lua	5.1.4	
lua-libs		5.4.4
lz4-libs		1.9.4
mpfr		4.1.0
ncurses	6.0	
ncurses-base	6.0	6.2
ncurses-libs	6.0	6.2
npth		1.6
nspr	4.35.0	
nss	3.90.0	

套件	AL2 容器	AL2023 容器
nss-pem	1.0.3	
nss-softokn	3.90.0	
nss-softokn-freebl	3.90.0	
nss-sysinit	3.90.0	
nss-tools	3.90.0	
nss-util	3.90.0	
openldap	2.4.44	
openssl-libs	1.0.2k	3.0.8
p11-kit	0.23.22	0.24.1
p11-kit-trust	0.23.22	0.24.1
pcre	8.32	
pcre2		10.40
pcre2-syntax		10.40
pinentry	0.8.1	
popt	1.13	1.18
pth	2.0.7	
publicsuffix-list-dafsa	20240208	20240212
pygpgme	0.3	
pyliblzma	0.5.3	
python	2.7.18	

套件	AL2 容器	AL2023 容器
python2-rpm	4.11.3	
python3		3.9.16
python3-dnf		4.14.0
python3-gpg		1.15.1
python3-hawkey		0.69.0
python3-libcomps		0.1.20
python3-libdnf		0.69.0
python3-libs		3.9.16
python3-pip-wheel		21.3.1
python3-rpm		4.16.1.3
python3-setuptools-wheel		59.6.0
python-iniparse	0.4	
python-libs	2.7.18	
python-pycurl	7.19.0	
python-urlgrabber	3.10	
pyxattr	0.5.1	
readline	6.2	8.1
rpm	4.11.3	4.16.1.3
rpm-build-libs	4.11.3	4.16.1.3
rpm-libs	4.11.3	4.16.1.3

套件	AL2 容器	AL2023 容器
rpm-sign-libs		4.16.1.3
sed	4.2.2	4.8
setup	2.8.71	2.13.7
shared-mime-info	1.8	
sqlite	3.7.17	
sqlite-libs		3.40.0
system-release	2	2023.6.20241031
tzdata	2024a	2024a
vim-data	9.0.2153	
vim-minimal	9.0.2153	
xz-libs	5.2.2	5.2.5
yum	3.4.3	4.14.0
yum-metadata-parser	1.1.4	
yum-plugin-ovl	1.1.31	
yum-plugin-priorities	1.1.31	
zlib	1.2.7	1.2.11

比較 AL1 與 AL2023

下列主題說明 AL1 和 AL2023 之間尚未涵蓋的 AL2 比較的主要差異。

> **Note**
>
> AL1 已於 2023 年 12 月 31 日達到end-of-life (EOL)，且自 2024 年 1 月 1 日起將不會收到任何安全性更新或錯誤修正。如需 AL1 EOL 和維護支援的詳細資訊，請參閱部落格文章 Amazon Linux AMI end-of-life更新。建議您將應用程式升級至 AL2023，其中包括直到 2028 年的長期支援。

主題

支援各個版本

對於 AL2023，我們提供自發行日期起五年的支援。AL1 自 2020 年 12 月 31 日起結束標準支援，自 2023 年 12 月 31 日起結束維護支援。

如需詳細資訊，請參閱發行節奏。

`systemd` 取代 `upstart` 並成為 `init` 系統

在 AL2 中， `upstart` 已取代為 `systemd`做為`init`系統。AL2023 也會使用 `systemd`做為其`init`系統，進一步採用 的新功能`systemd`。

Python 2.6 和 2.7 已替換為 Python 3

雖然 AL1 將 Python 2.6 標記為具有 2018.03 版的 EOL，但儲存庫中仍提供套件以供安裝。AL2 隨附 Python 2.7 作為最早支援的 Python 版本，AL2023 完成轉換為 Python 3。AL2023 儲存庫中不包含 Python 2.x 版本。

如需有關 Amazon Linux 的 Python 的詳細資訊，請參閱 AL2023 中的 Python。

OpenJDK 8 作為最舊版 JDK

AL2023 隨附 Amazon Corretto 作為預設 (也是唯一) Java 開發套件 (JDK)。AL2023 中的所有 Java 型套件都是使用 建置Amazon Corretto 17。

在 AL1 中，OpenJDK 1.6.0 (`java-1.6.0-openjdk`) 在第一個 2018.03 版本中進入 EOL，而 OpenJDK 1.7.0 (`java-1.7.0-openjdk`) 則在 2020 年中進入 EOL，不過這兩個版本都可用於 AL1 儲存庫。AL2023 中可用的最早 OpenJDK 版本為 OpenJDK 8，由 提供Amazon Corretto 8。

從 Amazon Linux 1 (AL1) 的 AL1AL2023 核心變更

Kernel Live Patching

AL2023 和 AL2 新增對核心即時修補功能的支援。這可讓您修補 Linux 核心中的關鍵和重要安全漏洞，而無需重新啟動或停機。如需詳細資訊，請參閱AL2023 上的核心即時修補。

核心檔案系統支援

AL1 中的核心將支援掛載的檔案系統中有幾項變更，以及核心將剖析的分割結構描述中的變更。

CONFIG 選項	AL1/4.14/ x86_64	AL2023/6.1/ aarch64	AL2023/6.1/ x86_64	AL2023/6.12/ aarch64	AL2023/6.12/ x86_64
CONFIG_AFS_FS	m	n	n	n	n
CONFIG_AF_RXRPC	m	n	n	n	n

CONFIG 選項	AL1/4.14/ x86_64	AL2023/6.1/ aarch64	AL2023/6.1/ x86_64	AL2023/6.12/ aarch64	AL2023/6.12/ x86_64
CONFIG_BSD_DISKLABEL	y	n	n	n	n
CONFIG_CRAMFS	m	n	n	n	n
CONFIG_CRAMFS_BLOCKDEV	N/A	N/A	N/A	N/A	N/A
CONFIG_DM_CLONE	N/A	n	n	n	n
CONFIG_DM_ERA	n	n	n	n	n
CONFIG_DM_INTEGRITY	m	m	m	m	m
CONFIG_DM_LOG_WRITES	n	m	m	m	m
CONFIG_DM_SWITCH	n	n	n	n	n
CONFIG_DM_VERITY	n	m	m	m	m
CONFIG_ECRYPT_FS	m	n	n	n	n
CONFIG_EXFAT_FS	N/A	m	m	m	m

CONFIG 選項	AL1/4.14/x86_64	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_EXT2_FS	m	n	n	n	n
CONFIG_EXT3_FS	m	n	n	n	n
CONFIG_GFS2_FS	n	n	n	n	n
CONFIG_HFSPLUS_FS	m	n	n	n	n
CONFIG_HFS_FS	m	n	n	n	n
CONFIG_JFS_FS	n	n	n	n	n
CONFIG_LDM_PARTITION	y	n	n	n	n
CONFIG_MAC_PARTITION	y	n	n	n	n
CONFIG_NFS_V2	m	n	n	n	n
CONFIG_NTFS_FS	m	n	n	n	n
CONFIG_ROMFS_FS	m	n	n	n	n

CONFIG 選項	AL1/4.14/ x86_64	AL2023/6.1/ aarch64	AL2023/6.1/ x86_64	AL2023/6.12/ aarch64	AL2023/6.12/ x86_64
CONFIG_SOLARIS_X86_PARTITION	y	n	n	n	n
CONFIG_SQUASHFS_ZSTD	y	y	y	y	y
CONFIG_SUN_PARTITION	y	n	n	n	n

重視安全的核心設定變更

CONFIG 選項	AL1/4.14/ x86_64	AL2023/6.1/ aarch64	AL2023/6.1/ x86_64	AL2023/6.12/ aarch64	AL2023/6.12/ x86_64
CONFIG_BUG_ON_DATA_CORRUPTION	y	y	y	y	y
CONFIG_DEFAULT_MMAP_MIN_ADDR	4096	65536	65536	65536	65536
CONFIG_DEVMEM	y	n	n	n	n
CONFIG_DEVPORT	y	n	n	n	n

CONFIG 選項	AL1/4.14/ x86_64	AL2023/6.1/ aarch64	AL2023/6.1/ x86_64	AL2023/6.12/ aarch64	AL2023/6.12/ x86_64
CONFIG_FORTIFY_SOURCE	y	y	y	y	y
CONFIG_HARDENED_USERCOPY_FALLBACK	N/A	N/A	N/A	N/A	N/A
CONFIG_INIT_ON_ALLOC_DEFAULT_ON	N/A	n	n	n	n
CONFIG_INIT_ON_FREE_DEFAULT_ON	N/A	n	n	n	n
CONFIG_IOMMU_DEFAULT_DMA_STRICT	N/A	n	n	n	n
CONFIG_LDISC_AUTOLOAD	y	n	n	n	n
CONFIG_SCHED_CORE	N/A	N/A	y	N/A	y
CONFIG_SCHED_STACK_END_CHECK	y	y	y	y	y

CONFIG 選項	AL1/4.14/ x86_64	AL2023/6.1/ aarch64	AL2023/6.1/ x86_64	AL2023/6.12/ aarch64	AL2023/6.12/ x86_64
CONFIG_SECURITY_DMESG_RESTRICT	n	y	y	y	y
CONFIG_SECURITY_SELINUX_DISABLE	y	n	n	N/A	N/A
CONFIG_SHUFFLE_PAGE_ALLOCATOR	N/A	y	y	y	y
CONFIG_SLAB_FREELIST_HARDENED	y	y	y	y	y
CONFIG_SLAB_FREELIST_RANDOM	n	y	y	y	y

其他核心設定變更

CONFIG 選項	AL1/4.14/ x86_64	AL2023/6.1/ aarch64	AL2023/6.1/ x86_64	AL2023/6.12/ aarch64	AL2023/6.12/ x86_64
CONFIG_HZ	250	100	100	100	100
CONFIG_NR_CPUS	8192	4096	8192	4096	8192

CONFIG 選項	AL1/4.14/x86_64	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_PANIC_ON_OOPS	n	y	y	y	y
CONFIG_PANIC_ON_OOPS_VALUE	0	1	1	1	1
CONFIG_PPP	m	n	n	n	n
CONFIG_SLIP	m	n	n	n	n
CONFIG_XEN_PV	y	N/A	n	N/A	n

比較 Amazon Linux 1 (AL1) 與 Amazon 2023 AMI 安裝的套件

AL1 和 AL2023 標準 AMIs 上存在的 RPMs 比較。

套件	AL1 AMI	AL2023 AMI
acl	2.2.49	2.3.1
acpid	2.0.19	2.0.32
alsa-lib	1.0.22	
alternatives		1.15
amazon-chrony-config		4.3
amazon-ec2-net-utils		2.5.1

套件	AL1 AMI	AL2023 AMI
amazon-linux-repo-s3		2023.6.20241031
amazon-linux-sb-keys		2023.1
amazon-rpm-config		228
amazon-ssm-agent	3.2.2222.0	3.3.987.0
amd-ucode-firmware		20210208
at	3.1.10	3.1.23
attr	2.4.46	2.5.1
audit	2.6.5	3.0.6
audit-libs	2.6.5	3.0.6
authconfig	6.2.8	
aws-amitools-ec2	1.5.13	
aws-cfn-bootstrap	1.4	2.0
aws-cli	1.18.107	
awscli-2		2.15.30
basesystem	10.0	11
bash	4.2.46	5.2.15
bash-completion		2.11
bc	1.06.95	1.07.1
bind-libs	9.8.2	9.18.28
bind-license		9.18.28

套件	AL1 AMI	AL2023 AMI
bind-utils	9.8.2	9.18.28
binutils	2.27	2.39
boost-filesystem		1.75.0
boost-system		1.75.0
boost-thread		1.75.0
bzip2	1.0.6	1.0.8
bzip2-libs	1.0.6	1.0.8
ca-certificates	2023.2.62	2023.2.68
c-ares		1.19.1
checkpolicy	2.1.10	3.4
chkconfig	1.3.49.3	1.15
chrony		4.3
cloud-disk-utils	0.27	
cloud-init	0.7.6	22.2.2
cloud-init-cfg-ec2		22.2.2
cloud-utils-growpart		0.31
copy-jdk-configs	3.3	
coreutils	8.22	8.32
coreutils-common		8.32
cpio	2.10	2.13

套件	AL1 AMI	AL2023 AMI
cracklib	2.8.16	2.9.6
cracklib-dicts	2.8.16	2.9.6
cronie	1.4.4	
cronie-anacron	1.4.4	
crontabs	1.10	1.11
crypto-policies		20220428
crypto-policies-scripts		20220428
cryptsetup	1.6.7	2.6.1
cryptsetup-libs	1.6.7	2.6.1
curl	7.61.1	
curl-minimal		8.5.0
cyrus-sasl	2.1.23	
cyrus-sasl-lib	2.1.23	2.1.27
cyrus-sasl-plain	2.1.23	2.1.27
dash	0.5.5.1	
db4	4.7.25	
db4-utils	4.7.25	
dbus	1.6.12	1.12.28
dbus-broker		32
dbus-common		1.12.28

套件	AL1 AMI	AL2023 AMI
dbus-libs	1.6.12	1.12.28
dejavu-fonts-common	2.33	
dejavu-sans-fonts	2.33	
dejavu-serif-fonts	2.33	
device-mapper	1.02.135	1.02.185
device-mapper-event	1.02.135	
device-mapper-event-libs	1.02.135	
device-mapper-libs	1.02.135	1.02.185
device-mapper-persistent-data	0.6.3	
dhclient	4.1.1	
dhcp-common	4.1.1	
diffutils	3.3	3.8
dmraid	1.0.0.rc16	
dmraid-events	1.0.0.rc16	
dnf		4.14.0
dnf-data		4.14.0
dnf-plugin-release-notification		1.2
dnf-plugins-core		4.3.0

套件	AL1 AMI	AL2023 AMI
dnf-plugin-support-info		1.2
dnf-utils		4.3.0
dosfstools		4.2
dracut	004	055
dracut-config-ec2		3.0
dracut-config-generic		055
dracut-modules-growroot	0.20	
dump	0.4	
dwz		0.14
dyninst		10.2.1
e2fsprogs	1.43.5	1.46.5
e2fsprogs-libs	1.43.5	1.46.5
ec2-hibinit-agent	1.0.0	1.0.8
ec2-instance-connect		1.1
ec2-instance-connect-selinux		1.1
ec2-net-utils	0.7	
ec2-utils	0.7	2.2.0
ed	1.1	1.14.2

套件	AL1 AMI	AL2023 AMI
efi-filesystem		5
efi-srpm-macros		5
efivar		38
efivar-libs		38
elfutils-debuginfod-client		0.188
elfutils-default-yama-scope		0.188
elfutils-libelf	0.168	0.188
elfutils-libs		0.188
epel-release	6	
ethtool	3.15	5.15
expat	2.1.0	2.5.0
file	5.37	5.39
file-libs	5.37	5.39
filesystem	2.4.30	3.14
findutils	4.4.2	4.8.0
fipscheck	1.3.1	
fipscheck-lib	1.3.1	
fontconfig	2.8.0	

套件	AL1 AMI	AL2023 AMI
fontpackages-filesystem	1.41	
fonts-srpm-macros		2.0.5
freetype	2.3.11	
fstrm		0.6.1
fuse-libs	2.9.4	2.9.9
gawk	3.1.7	5.1.0
gdbm	1.8.0	
gdbm-libs		1.19
gdisk	0.8.10	1.0.8
generic-logos	17.0.0	
get_reference_source	1.2	
gettext		0.21
gettext-libs		0.21
ghc-srpm-macros		1.5.0
giflib	4.1.6	
glib2	2.36.3	2.74.7
glibc	2.17	2.34
glibc-all-langpacks		2.34
glibc-common	2.17	2.34
glibc-gconv-extra		2.34

套件	AL1 AMI	AL2023 AMI
glibc-locale-source		2.34
gmp	6.0.0	6.2.1
gnupg2	2.0.28	
gnupg2-minimal		2.3.7
gnutls		3.8.0
go-srpm-macros		3.2.0
gpgme	1.4.3	1.15.1
gpm-libs	1.20.6	1.20.7
grep	2.20	3.8
groff	1.22.2	
groff-base	1.22.2	1.22.4
grub	0.97	
grub2-common		2.06
grub2-efi-x64-ec2		2.06
grub2-pc-modules		2.06
grub2-tools		2.06
grub2-tools-minimal		2.06
grubby	7.0.15	8.40
gssproxy		0.8.4
gzip	1.5	1.12

套件	AL1 AMI	AL2023 AMI
hesiod	3.1.0	
hibagent	1.0.0	
hmaccalc	0.9.12	
hostname		3.23
hunspell		1.7.0
hunspell-en		0.20140811.1
hunspell-en-GB		0.20140811.1
hunspell-en-US		0.20140811.1
hunspell-filesystem		1.7.0
hwdata	0.233	0.384
info	5.1	6.7
inih		49
initscripts	9.03.58	10.09
iproute	4.4.0	6.10.0
iptables	1.4.21	
iputils	20121221	20210202
irqbalance	1.5.0	1.9.0
jansson		2.14
java-1.7.0-openjdk	1.7.0.321	
javapackages-tools	0.9.1	

套件	AL1 AMI	AL2023 AMI
jemalloc		5.2.1
jitterentropy		3.4.1
jpackage-utils	1.7.5	
jq		1.7.1
json-c		0.14
kbd	1.15	2.4.0
kbd-misc	1.15	2.4.0
kernel	4.14.336	6.1.112
kernel-libbpf		6.1.112
kernel-livepatch-repo-s3		2023.6.20241031
kernel-srpm-macros		1.0
kernel-tools	4.14.336	6.1.112
keyutils	1.5.8	1.6.3
keyutils-libs	1.5.8	1.6.3
kmod	14	29
kmod-libs	14	29
kpartx	0.4.9	
kpatch-runtime		0.9.7
krb5-libs	1.15.1	1.21.3
lcms2	2.6	

套件	AL1 AMI	AL2023 AMI
less	436	608
libacl	2.2.49	2.3.1
libaio	0.3.109	0.3.111
libarchive		3.7.4
libargon2		20171227
libassuan	2.0.3	2.5.5
libattr	2.4.46	2.5.1
libbasicobjects		0.1.1
libblkid	2.23.2	2.37.4
libcap	2.16	2.48
libcap54	2.54	
libcap-ng	0.7.5	0.8.2
libcbor		0.7.0
libcgroup	0.40.rc1	
libcollection		0.7.0
libcom_err	1.43.5	1.46.5
libcomps		0.1.20
libconfig		1.7.2
libcurl	7.61.1	
libcurl-minimal		8.5.0

套件	AL1 AMI	AL2023 AMI
libdb		5.3.28
libdhash		0.5.0
libdnf		0.69.0
libeconf		0.4.0
libedit	2.11	3.1
libev		4.33
libevent	2.0.21	2.1.12
libfdisk		2.37.4
libffi	3.0.13	3.4.4
libfido2		1.10.0
libfontenc	1.0.5	
libgcc		11.4.1
libgcc72	7.2.1	
libgcrypt	1.5.3	1.10.2
libgomp		11.4.1
libgpg-error	1.11	1.42
libgssglue	0.1	
libibverbs		48.0
libICE	1.0.6	
libicu	50.2	

套件	AL1 AMI	AL2023 AMI
libidn	1.18	
libidn2	2.3.0	2.3.2
libini_config		1.3.1
libjpeg-turbo	1.2.90	
libkcapi		1.4.0
libkcapi-hmaccalc		1.4.0
libldb		2.6.2
libmaxminddb		1.5.2
libmetalink		0.1.3
libmnl	1.0.3	1.0.4
libmodulemd		2.13.0
libmount	2.23.2	2.37.4
libnetfilter_conntrack	1.0.4	
libnfnetlink	1.0.1	
libnfsidmap	0.25	2.5.4
libnghttp2	1.33.0	1.59.0
libnih	1.0.1	
libnl	1.1.4	
libnl3		3.5.0
libpath_utils		0.2.1

套件	AL1 AMI	AL2023 AMI
libpcap		1.10.1
libpipeline	1.2.3	1.5.3
libpkgconf		1.8.0
libpng	1.2.49	
libpsl	0.6.2	0.21.1
libpwquality	1.2.3	1.4.4
libref_array		0.1.5
librepo		1.14.5
libreport-filesystem		2.15.2
libseccomp		2.5.3
libselinux	2.1.10	3.4
libselinux-utils	2.1.10	3.4
libsemanage	2.1.6	3.4
libsepol	2.1.7	3.4
libsigsegv		2.13
libSM	1.2.1	
libsmartcols	2.23.2	2.37.4
libsolv		0.7.22
libss	1.43.5	1.46.5
libssh2	1.4.2	

套件	AL1 AMI	AL2023 AMI
libsss_certmap		2.9.4
libsss_idmap		2.9.4
libsss_nss_idmap		2.9.4
libsss_sudo		2.9.4
libstdc++		11.4.1
libstdc++72	7.2.1	
libstoragemgmt		1.9.4
libsysfs	2.1.0	
libtalloc		2.3.4
libtasn1	2.3	4.19.0
libtdb		1.4.7
libtevent		0.13.0
libtextstyle		0.21
libtirpc	0.2.4	1.3.3
libudev	173	
libunistring	0.9.3	0.9.10
libuser	0.60	0.63
libutempter	1.1.5	1.2.1
libuuid	2.23.2	2.37.4
libuv		1.47.0

套件	AL1 AMI	AL2023 AMI
libverto	0.2.5	0.3.2
libverto-libev		0.3.2
libX11	1.6.0	
libX11-common	1.6.0	
libXau	1.0.6	
libxcb	1.11	
libXcomposite	0.4.3	
libxcrypt		4.4.33
libXext	1.3.2	
libXfont	1.4.5	
libXi	1.7.2	
libxml2	2.9.1	2.10.4
libxml2-python27	2.9.1	
libXrender	0.9.8	
libxslt	1.1.28	
libXtst	1.2.2	
libyaml	0.1.6	0.2.5
libzstd		1.5.5
linux-firmware-whence		20210208
lm_sensors-libs		3.6.0

套件	AL1 AMI	AL2023 AMI
lmdb-libs		0.9.29
log4j-cve-2021-44228-hotpatch	1.3	
logrotate	3.7.8	3.20.1
lsof	4.82	4.94.0
lua	5.1.4	
lua-libs		5.4.4
lua-srpm-macros		1
lvm2	2.02.166	
lvm2-libs	2.02.166	
lz4-libs		1.9.4
mailcap	2.1.31	
make	3.82	
man-db	2.6.3	2.9.3
man-pages	4.10	5.10
mdadm	3.2.6	
microcode_ctl	2.1	2.1
mingetty	1.08	
mpfr		4.1.0
nano	2.5.3	5.8
nc	1.84	

套件	AL1 AMI	AL2023 AMI
ncurses	5.7	6.2
ncurses-base	5.7	6.2
ncurses-libs	5.7	6.2
nettle		3.8
net-tools	1.60	2.0
newt	0.52.11	0.52.21
newt-python27	0.52.11	
nfs-utils	1.3.0	2.5.4
npth		1.6
nspr	4.25.0	4.35.0
nss	3.53.1	3.90.0
nss-pem	1.0.3	
nss-softokn	3.53.1	3.90.0
nss-softokn-freebl	3.53.1	3.90.0
nss-sysinit	3.53.1	3.90.0
nss-tools	3.53.1	
nss-util	3.53.1	3.90.0
ntp	4.2.8p15	
ntpdate	4.2.8p15	
ntsysv	1.3.49.3	1.15

套件	AL1 AMI	AL2023 AMI
numactl	2.0.7	
numactl-libs		2.0.14
ocaml-srpm-macros		6
oniguruma		6.9.7.1
openblas-srpm-macros		2
openldap	2.4.40	2.4.57
openssh	7.4p1	8.7p1
openssh-clients	7.4p1	8.7p1
openssh-server	7.4p1	8.7p1
openssl	1.0.2k	3.0.8
openssl-libs		3.0.8
openssl-pkcs11		0.4.12
os-prober		1.77
p11-kit	0.18.5	0.24.1
p11-kit-trust	0.18.5	0.24.1
package-notes-srpm-macros		0.4
pam	1.1.8	1.5.1
pam_ccreds	10	
pam_krb5	2.3.11	
pam_passwdqc	1.0.5	

套件	AL1 AMI	AL2023 AMI
parted	2.1	3.4
passwd	0.79	0.80
pciutils	3.1.10	3.7.0
pciutils-libs	3.1.10	3.7.0
pcre	8.21	
pcre2		10.40
pcre2-syntax		10.40
perl	5.16.3	
perl-Carp	1.26	1.50
perl-Class-Struct		0.66
perl-constant	1.27	1.33
perl-Digest	1.17	
perl-Digest-HMAC	1.03	
perl-Digest-MD5	2.52	
perl-Digest-SHA	5.85	
perl-DynaLoader		1.47
perl-Encode	2.51	3.15
perl-Errno		1.30
perl-Exporter	5.68	5.74
perl-Fcntl		1.13

套件	AL1 AMI	AL2023 AMI
perl-File-Basename		2.85
perl-File-Path	2.09	2.18
perl-File-stat		1.09
perl-File-Temp	0.23.01	0.231.100
perl-Filter	1.49	
perl-Getopt-Long	2.40	2.52
perl-Getopt-Std		1.12
perl-HTTP-Tiny	0.033	0.078
perl-if		0.60.800
perl-interpreter		5.32.1
perl-IO		1.43
perl-IPC-Open3		1.21
perl-libs	5.16.3	5.32.1
perl-macros	5.16.3	
perl-MIME-Base64		3.16
perl-mro		1.23
perl-overload		1.31
perl-overloading		0.02
perl-parent	0.225	0.238
perl-PathTools	3.40	3.78

套件	AL1 AMI	AL2023 AMI
perl-Pod-Escapes	1.04	1.07
perl-podlators	2.5.1	4.14
perl-Pod-Perldoc	3.20	3.28.01
perl-Pod-Simple	3.28	3.42
perl-Pod-Usage	1.63	2.01
perl-POSIX		1.94
perl-Scalar-List-Utils	1.27	1.56
perl-SelectSaver		1.02
perl-Socket	2.010	2.032
perl-srpm-macros		1
perl-Storable	2.45	3.21
perl-subs		1.03
perl-Symbol		1.08
perl-Term-ANSIColor		5.01
perl-Term-Cap		1.17
perl-Text-ParseWords	3.29	3.30
perl-Text-Tabs+Wrap		2021.0726
perl-threads	1.87	
perl-threads-shared	1.43	
perl-Time-HiRes	1.9725	

套件	AL1 AMI	AL2023 AMI
perl-Time-Local	1.2300	1.300
perl-vars		1.05
pinentry	0.7.6	
pkgconf		1.8.0
pkgconfig	0.27.1	
pkgconf-m4		1.8.0
pkgconf-pkg-config		1.8.0
pm-utils	1.4.1	
policycoreutils	2.1.12	3.4
policycoreutils-python-utils		3.4
popt	1.13	1.18
procmail	3.22	
procps	3.2.8	
procps-ng		3.3.17
protobuf-c		1.4.1
psacct	6.3.2	6.6.4
psmisc	22.20	23.4
pth	2.0.7	
publicsuffix-list-dafsa		20240212

套件	AL1 AMI	AL2023 AMI
python27	2.7.18	
python27-babel	0.9.4	
python27-backports	1.0	
python27-backports-ssl_match_hostname	3.4.0.2	
python27-boto	2.48.0	
python27-botocore	1.17.31	
python27-chardet	2.0.1	
python27-colorama	0.4.1	
python27-configobj	4.7.2	
python27-crypto	2.6.1	
python27-daemon	1.5.2	
python27-dateutil	2.1	
python27-devel	2.7.18	
python27-docutils	0.11	
python27-ecdsa	0.11	
python27-futures	3.0.3	
python27-imaging	1.1.6	
python27-iniparse	0.3.1	
python27-jinja2	2.7.2	
python27-jmespath	0.9.2	

套件	AL1 AMI	AL2023 AMI
python27-jsonpatch	1.2	
python27-jsonpointer	1.0	
python27-kitchen	1.1.1	
python27-libs	2.7.18	
python27-lockfile	0.8	
python27-markupsafe	0.11	
python27-paramiko	1.15.1	
python27-pip	9.0.3	
python27-ply	3.4	
python27-pyasn1	0.1.7	
python27-pycurl	7.19.0	
python27-pygpgme	0.3	
python27-pyliblzma	0.5.3	
python27-pystache	0.5.3	
python27-pyxattr	0.5.0	
python27-PyYAML	3.10	
python27-requests	1.2.3	
python27-rsa	3.4.1	
python27-setuptools	36.2.7	
python27-simplejson	3.6.5	

套件	AL1 AMI	AL2023 AMI
python27-six	1.8.0	
python27-urlgrabber	3.10	
python27-urllib3	1.24.3	
python27-virtualenv	15.1.0	
python3		3.9.16
python3-attrs		20.3.0
python3-audit		3.0.6
python3-awscrt		2019 年 19 月 0 日
python3-babel		2.9.1
python3-cffi		1.14.5
python3-chardet		4.0.0
python3-colorama		0.4.4
python3-configobj		5.0.6
python3-cryptography		36.0.1
python3-daemon		2.3.0
python3-dateutil		2.8.1
python3-dbus		1.2.18
python3-distro		1.5.0
python3-dnf		4.14.0
python3-dnf-plugins-core		4.3.0

套件	AL1 AMI	AL2023 AMI
python3-docutils		0.16
python3-gpg		1.15.1
python3-hawkey		0.69.0
python3-idna		2.10
python3-jinja2		2.11.3
python3-jmespath		0.10.0
python3-jsonpatch		1.21
python3-jsonpointer		2.0
python3-jsonschema		3.2.0
python3-libcomps		0.1.20
python3-libdnf		0.69.0
python3-libs		3.9.16
python3-libselinux		3.4
python3-libsemanage		3.4
python3-libstorage mgmt		1.9.4
python3-lockfile		0.12.2
python3-markupsafe		1.1.1
python3-netifaces		0.10.6
python3-oauthlib		3.0.2
python3-pip-wheel		21.3.1

套件	AL1 AMI	AL2023 AMI
python3-ply		3.11
python3-policycoreutils		3.4
python3-prettytable		0.7.2
python3-prompt-toolkit		3.0.24
python3-pycparser		2.20
python3-pyrsistent		0.17.3
python3-pyserial		3.4
python3-pysocks		1.7.1
python3-pytz		2022.7.1
python3-pyyaml		5.4.1
python3-requests		2.25.1
python3-rpm		4.16.1.3
python3-ruamel-yaml		0.16.6
python3-ruamel-yaml-clib		0.1.2
python3-setools		4.4.1
python3-setuptools		59.6.0
python3-setuptools-wheel		59.6.0
python3-six		1.15.0

套件	AL1 AMI	AL2023 AMI
python3-systemd		235
python3-urllib3		1.25.10
python3-wcwidth		0.2.5
python-chevron		0.13.1
python-srpm-macros		3.9
quota	4.00	4.06
quota-nls	4.00	4.06
readline	6.2	8.1
rmt	0.4	
rng-tools	5	6.14
rootfiles	8.1	8.1
rpcbind	0.2.0	1.2.6
rpm	4.11.3	4.16.1.3
rpm-build-libs	4.11.3	4.16.1.3
rpm-libs	4.11.3	4.16.1.3
rpm-plugin-selinux		4.16.1.3
rpm-plugin-systemd-inhibit		4.16.1.3
rpm-python27	4.11.3	
rpm-sign-libs		4.16.1.3
rsync	3.0.6	3.2.6

套件	AL1 AMI	AL2023 AMI
rsyslog	5.8.10	
ruby	2.0	
ruby20	2.0.0.648	
ruby20-irb	2.0.0.648	
ruby20-libs	2.0.0.648	
rubygem20-bigdecimal	1.2.0	
rubygem20-json	1.8.3	
rubygem20-psych	2.0.0	
rubygem20-rdoc	4.2.2	
rubygems20	2.0.14.1	
rust-srpm-macros		21
sbsigntools		0.9.4
screen	4.0.3	4.8.0
sed	4.2.1	4.8
selinux-policy		38.1.45
selinux-policy-targeted		38.1.45
sendmail	8.14.4	
setserial	2.17	
setup	2.8.14	2.13.7
sgpio	1.2.0.10	

套件	AL1 AMI	AL2023 AMI
shadow-utils	4.1.4.2	4.9
shared-mime-info	1.1	
slang	2.2.1	2.3.2
sqlite	3.7.17	
sqlite-libs		3.40.0
sssd-client		2.9.4
sssd-common		2.9.4
sssd-kcm		2.9.4
sssd-nfs-idmap		2.9.4
strace		6.8
sudo	1.8.23	1.9.15
sysctl-defaults	1.0	1.0
sysfsutils	2.1.0	
sysstat		12.5.6
systemd		252.23
systemd-libs		252.23
systemd-networkd		252.23
systemd-pam		252.23
systemd-resolved		252.23
systemd-udev		252.23

套件	AL1 AMI	AL2023 AMI
system-release	2018.03	2023.6.20241031
systemtap-runtime		4.8
sysvinit	2.87	
tar	1.26	1.34
tbb		2020.3
tcp_wrappers	7.6	
tcp_wrappers-libs	7.6	
tcpdump		4.99.1
tcsh		6.24.07
time	1.7	1.9
tmpwatch	2.9.16	
traceroute	2.0.14	2.1.3
ttmkfdir	3.0.9	
tzdata	2023c	2024a
tzdata-java	2023c	
udev	173	
unzip	6.0	6.0
update-motd	1.0.1	2.2
upstart	0.6.5	
userspace-rcu		0.12.1

套件	AL1 AMI	AL2023 AMI
ustr	1.0.4	
util-linux	2.23.2	2.37.4
util-linux-core		2.37.4
vim-common	9.0.2120	9.0.2153
vim-data	9.0.2120	9.0.2153
vim-enhanced	9.0.2120	9.0.2153
vim-filesystem	9.0.2120	9.0.2153
vim-minimal	9.0.2120	9.0.2153
wget	1.18	1.21.3
which	2.19	2.21
words	3.0	3.0
xfsdump		3.1.11
xfsprogs		5.18.0
xorg-x11-fonts-Type1	7.2	
xorg-x11-font-utils	7.2	
xxd	9.0.2120	9.0.2153
xxhash-libs		0.8.0
xz	5.2.2	5.2.5
xz-libs	5.2.2	5.2.5
yum	3.4.3	4.14.0

套件	AL1 AMI	AL2023 AMI
yum-metadata-parser	1.1.4	
yum-plugin-priorities	1.1.31	
yum-plugin-upgrade-helper	1.1.31	
yum-utils	1.1.31	
zip	3.0	3.0
zlib	1.2.8	1.2.11
zram-generator		1.1.2
zram-generator-defaults		1.1.2
zstd		1.5.5

比較 Amazon Linux 1 (AL1) 與 Amazon Linux 2023 最小 AMI 安裝的套件

AL1 和 AL2023 最小 AMIs 上存在的 RPMs 比較。

套件	AL1 最小	AL2023 最小
acpid	2.0.19	
alternatives		1.15
amazon-chrony-config		4.3
amazon-ec2-net-utils		2.5.1
amazon-linux-repo-s3		2023.6.20241031

套件	AL1 最小	AL2023 最小
amazon-linux-sb-keys		2023.1
amd-ucode-firmware		20210208
audit	2.6.5	3.0.6
audit-libs	2.6.5	3.0.6
authconfig	6.2.8	
awscli-2		2.15.30
basesystem	10.0	11
bash	4.2.46	5.2.15
binutils	2.27	
bzip2	1.0.6	
bzip2-libs	1.0.6	1.0.8
ca-certificates	2023.2.62	2023.2.68
checkpolicy	2.1.10	3.4
chkconfig	1.3.49.3	
chrony		4.3
cloud-disk-utils	0.27	
cloud-init	0.7.6	22.2.2
cloud-init-cfg-ec2		22.2.2
cloud-utils-growpart		0.31
coreutils	8.22	8.32

套件	AL1 最小	AL2023 最小
coreutils-common		8.32
cpio	2.10	2.13
cracklib	2.8.16	2.9.6
cracklib-dicts	2.8.16	2.9.6
cronie	1.4.4	
cronie-anacron	1.4.4	
crontabs	1.10	
crypto-policies		20220428
cryptsetup-libs		2.6.1
curl	7.61.1	
curl-minimal		8.5.0
cyrus-sasl	2.1.23	
cyrus-sasl-lib	2.1.23	2.1.27
dash	0.5.5.1	
db4	4.7.25	
db4-utils	4.7.25	
dbus		1.12.28
dbus-broker		32
dbus-common		1.12.28
dbus-libs	1.6.12	1.12.28

套件	AL1 最小	AL2023 最小
device-mapper		1.02.185
device-mapper-libs		1.02.185
dhclient	4.1.1	
dhcp-common	4.1.1	
diffutils	3.3	3.8
dnf		4.14.0
dnf-data		4.14.0
dnf-plugin-release-notification		1.2
dnf-plugins-core		4.3.0
dnf-plugin-support-info		1.2
dracut	004	055
dracut-config-ec2		3.0
dracut-config-generic		055
dracut-modules-growroot	0.20	
e2fsprogs	1.43.5	1.46.5
e2fsprogs-libs	1.43.5	1.46.5
ec2-utils	0.7	2.2.0
ed	1.1	

套件	AL1 最小	AL2023 最小
efi-filesystem		5
efivar		38
efivar-libs		38
elfutils-default-yama-scope		0.188
elfutils-libelf	0.168	0.188
elfutils-libs		0.188
ethtool	3.15	
expat	2.1.0	2.5.0
file	5.37	5.39
file-libs	5.37	5.39
filesystem	2.4.30	3.14
findutils	4.4.2	4.8.0
fipscheck	1.3.1	
fipscheck-lib	1.3.1	
fuse-libs	2.9.4	2.9.9
gawk	3.1.7	5.1.0
gdbm	1.8.0	
gdbm-libs		1.19
gdisk	0.8.10	1.0.8
generic-logos	17.0.0	

套件	AL1 最小	AL2023 最小
get_reference_source	1.2	
gettext		0.21
gettext-libs		0.21
glib2	2.36.3	2.74.7
glibc	2.17	2.34
glibc-all-langpacks		2.34
glibc-common	2.17	2.34
glibc-locale-source		2.34
gmp	6.0.0	6.2.1
gnupg2	2.0.28	
gnupg2-minimal		2.3.7
gnutls		3.8.0
gpgme	1.4.3	1.15.1
grep	2.20	3.8
groff	1.22.2	
groff-base	1.22.2	1.22.4
grub	0.97	
grub2-common		2.06
grub2-efi-x64-ec2		2.06
grub2-pc-modules		2.06

套件	AL1 最小	AL2023 最小
grub2-tools		2.06
grub2-tools-minimal		2.06
grubby	7.0.15	8.40
gzip	1.5	1.12
hesiod	3.1.0	
hmaccalc	0.9.12	
hostname		3.23
hwdata	0.233	0.384
info	5.1	
inih		49
initscripts	9.03.58	10.09
iproute	4.4.0	6.10.0
iptables	1.4.21	
iputils	20121221	20210202
irqbalance		1.9.0
jansson		2.14
jitterentropy		3.4.1
jq		1.7.1
json-c		0.14
kbd	1.15	2.4.0

套件	AL1 最小	AL2023 最小
kbd-misc	1.15	2.4.0
kernel	4.14.336	6.1.112
kernel-libbpf		6.1.112
kernel-livepatch-repo-s3		2023.6.20241031
keyutils-libs	1.5.8	1.6.3
kmod	14	29
kmod-libs	14	29
krb5-libs	1.15.1	1.21.3
less	436	608
libacl	2.2.49	2.3.1
libarchive		3.7.4
libargon2		20171227
libassuan	2.0.3	2.5.5
libattr	2.4.46	2.5.1
libblkid	2.23.2	2.37.4
libcap	2.16	2.48
libcap54	2.54	
libcap-ng	0.7.5	0.8.2
libcbor		0.7.0
libcgroup	0.40.rc1	

套件	AL1 最小	AL2023 最小
libcom_err	1.43.5	1.46.5
libcomps		0.1.20
libcurl	7.61.1	
libcurl-minimal		8.5.0
libdb		5.3.28
libdnf		0.69.0
libeconf		0.4.0
libedit	2.11	3.1
libfdisk		2.37.4
libffi	3.0.13	3.4.4
libfido2		1.10.0
libgcc		11.4.1
libgcc72	7.2.1	
libgcrypt	1.5.3	1.10.2
libgomp		11.4.1
libgpg-error	1.11	1.42
libicu	50.2	
libidn	1.18	
libidn2	2.3.0	2.3.2
libkcapi		1.4.0

套件	AL1 最小	AL2023 最小
libkcapi-hmaccalc		1.4.0
libmnl	1.0.3	1.0.4
libmodulemd		2.13.0
libmount	2.23.2	2.37.4
libnetfilter_connt rack	1.0.4	
libnfnetlink	1.0.1	
libnghttp2	1.33.0	1.59.0
libnih	1.0.1	
libpipeline		1.5.3
libpsl	0.6.2	0.21.1
libpwquality	1.2.3	1.4.4
librepo		1.14.5
libreport-filesystem		2.15.2
libseccomp		2.5.3
libselinux	2.1.10	3.4
libselinux-utils	2.1.10	3.4
libsemanage	2.1.6	3.4
libsepol	2.1.7	3.4
libsigsegv		2.13
libsmartcols	2.23.2	2.37.4

套件	AL1 最小	AL2023 最小
libsolv		0.7.22
libss	1.43.5	1.46.5
libssh2	1.4.2	
libstdc++		11.4.1
libstdc++72	7.2.1	
libsysfs	2.1.0	
libtasn1	2.3	4.19.0
libtextstyle		0.21
libudev	173	
libunistring	0.9.3	0.9.10
libuser	0.60	0.63
libutempter	1.1.5	1.2.1
libuuid	2.23.2	2.37.4
libverto	0.2.5	0.3.2
libxcrypt		4.4.33
libxml2	2.9.1	2.10.4
libyaml	0.1.6	0.2.5
libzstd		1.5.5
linux-firmware-whence		20210208
logrotate	3.7.8	3.20.1

套件	AL1 最小	AL2023 最小
lua	5.1.4	
lua-libs		5.4.4
lz4-libs		1.9.4
make	3.82	
man-db		2.9.3
microcode_ctl	2.1	2.1
mingetty	1.08	
mpfr		4.1.0
ncurses	5.7	6.2
ncurses-base	5.7	6.2
ncurses-libs	5.7	6.2
nettle		3.8
net-tools	1.60	2.0
newt	0.52.11	
newt-python27	0.52.11	
npth		1.6
nspr	4.25.0	
nss	3.53.1	
nss-pem	1.0.3	
nss-softokn	3.53.1	

套件	AL1 最小	AL2023 最小
nss-softokn-freebl	3.53.1	
nss-sysinit	3.53.1	
nss-tools	3.53.1	
nss-util	3.53.1	
ntp	4.2.8p15	
ntpdate	4.2.8p15	
numactl-libs		2.0.14
oniguruma		6.9.7.1
openldap	2.4.40	2.4.57
openssh	7.4p1	8.7p1
openssh-clients		8.7p1
openssh-server	7.4p1	8.7p1
openssl	1.0.2k	3.0.8
openssl-libs		3.0.8
openssl-pkcs11		0.4.12
os-prober		1.77
p11-kit	0.18.5	0.24.1
p11-kit-trust	0.18.5	0.24.1
pam	1.1.8	1.5.1
passwd	0.79	0.80

套件	AL1 最小	AL2023 最小
pciutils	3.1.10	3.7.0
pciutils-libs	3.1.10	3.7.0
pcre	8.21	
pcre2		10.40
pcre2-syntax		10.40
pinentry	0.7.6	
pkgconfig	0.27.1	
policycoreutils	2.1.12	3.4
popt	1.13	1.18
procmail	3.22	
procps	3.2.8	
procps-ng		3.3.17
psmisc	22.20	23.4
pth	2.0.7	
publicsuffix-list-dafsa		20240212
python27	2.7.18	
python27-babel	0.9.4	
python27-backports	1.0	
python27-backports-ssl_match_hostname	3.4.0.2	

套件	AL1 最小	AL2023 最小
python27-chardet	2.0.1	
python27-configobj	4.7.2	
python27-iniparse	0.3.1	
python27-jinja2	2.7.2	
python27-jsonpatch	1.2	
python27-jsonpointer	1.0	
python27-libs	2.7.18	
python27-markupsafe	0.11	
python27-pycurl	7.19.0	
python27-pygpgme	0.3	
python27-pyliblzma	0.5.3	
python27-pyxattr	0.5.0	
python27-PyYAML	3.10	
python27-requests	1.2.3	
python27-setuptools	36.2.7	
python27-six	1.8.0	
python27-urlgrabber	3.10	
python27-urllib3	1.24.3	
python3		3.9.16
python3-attrs		20.3.0

套件	AL1 最小	AL2023 最小
python3-audit		3.0.6
python3-awscrt		0.19.19
python3-babel		2.9.1
python3-cffi		1.14.5
python3-chardet		4.0.0
python3-colorama		0.4.4
python3-configobj		5.0.6
python3-cryptography		36.0.1
python3-dateutil		2.8.1
python3-dbus		1.2.18
python3-distro		1.5.0
python3-dnf		4.14.0
python3-dnf-plugins-core		4.3.0
python3-docutils		0.16
python3-gpg		1.15.1
python3-hawkey		0.69.0
python3-idna		2.10
python3-jinja2		2.11.3
python3-jmespath		0.10.0
python3-jsonpatch		1.21

套件	AL1 最小	AL2023 最小
python3-jsonpointer		2.0
python3-jsonschema		3.2.0
python3-libcomps		0.1.20
python3-libdnf		0.69.0
python3-libs		3.9.16
python3-libselinux		3.4
python3-libsemanage		3.4
python3-markupsafe		1.1.1
python3-netifaces		0.10.6
python3-oauthlib		3.0.2
python3-pip-wheel		21.3.1
python3-ply		3.11
python3-policycoreutils		3.4
python3-prettytable		0.7.2
python3-prompt-toolkit		3.0.24
python3-pycparser		2.20
python3-pyrsistent		0.17.3
python3-pyserial		3.4
python3-pysocks		1.7.1

套件	AL1 最小	AL2023 最小
python3-pytz		2022.7.1
python3-pyyaml		5.4.1
python3-requests		2.25.1
python3-rpm		4.16.1.3
python3-ruamel-yaml		0.16.6
python3-ruamel-yaml-clib		0.1.2
python3-setools		4.4.1
python3-setuptools		59.6.0
python3-setuptools-wheel		59.6.0
python3-six		1.15.0
python3-systemd		235
python3-urllib3		1.25.10
python3-wcwidth		0.2.5
readline	6.2	8.1
rng-tools		6.14
rootfiles	8.1	8.1
rpm	4.11.3	4.16.1.3
rpm-build-libs	4.11.3	4.16.1.3
rpm-libs	4.11.3	4.16.1.3

套件	AL1 最小	AL2023 最小
rpm-plugin-selinux		4.16.1.3
rpm-plugin-systemd-inhibit		4.16.1.3
rpm-python27	4.11.3	
rpm-sign-libs		4.16.1.3
rsyslog	5.8.10	
sbsigntools		0.9.4
sed	4.2.1	4.8
selinux-policy		38.1.45
selinux-policy-targeted		38.1.45
sendmail	8.14.4	
setserial	2.17	
setup	2.8.14	2.13.7
shadow-utils	4.1.4.2	4.9
shared-mime-info	1.1	
slang	2.2.1	
sqlite	3.7.17	
sqlite-libs		3.40.0
sudo	1.8.23	1.9.15
sysctl-defaults	1.0	1.0

套件	AL1 最小	AL2023 最小
sysfsutils	2.1.0	
systemd		252.23
systemd-libs		252.23
systemd-networkd		252.23
systemd-pam		252.23
systemd-resolved		252.23
systemd-udev		252.23
system-release	2018.03	2023.6.20241031
sysvinit	2.87	
tar	1.26	1.34
tcp_wrappers-libs	7.6	
tzdata	2023c	2024a
udev	173	
update-motd	1.0.1	2.2
upstart	0.6.5	
userspace-rcu		0.12.1
ustr	1.0.4	
util-linux	2.23.2	2.37.4
util-linux-core		2.37.4
vim-data	9.0.2120	9.0.2153

套件	AL1 最小	AL2023 最小
vim-minimal	9.0.2120	9.0.2153
which	2.19	2.21
xfsprogs		5.18.0
xz	5.2.2	5.2.5
xz-libs	5.2.2	5.2.5
yum	3.4.3	4.14.0
yum-metadata-parser	1.1.4	
yum-plugin-priorities	1.1.31	
yum-plugin-upgrade-helper	1.1.31	
zlib	1.2.8	1.2.11
zram-generator		1.1.2
zram-generator-defaults		1.1.2
zstd		1.5.5

比較 Amazon Linux 1 (AL1) 與 Amazon 2023 基本容器映像安裝的套件

AL1 和 AL2023 基礎容器映像上存在的 RPMs 比較。

套件	AL1 容器	AL2023 容器
alternatives		1.15

套件	AL1 容器	AL2023 容器
amazon-linux-repo-cdn		2023.6.20241031
audit-libs		3.0.6
basesystem	10.0	11
bash	4.2.46	5.2.15
bzip2-libs	1.0.6	1.0.8
ca-certificates	2023.2.62	2023.2.68
chkconfig	1.3.49.3	
coreutils	8.22	
coreutils-single		8.32
crypto-policies		20220428
curl	7.61.1	
curl-minimal		8.5.0
cyrus-sasl-lib	2.1.23	
db4	4.7.25	
db4-utils	4.7.25	
dnf		4.14.0
dnf-data		4.14.0
elfutils-default-yama-scope		0.188
elfutils-libelf	0.168	0.188

套件	AL1 容器	AL2023 容器
elfutils-libs		0.188
expat	2.1.0	2.5.0
file-libs	5.37	5.39
filesystem	2.4.30	3.14
gawk	3.1.7	5.1.0
gdbm	1.8.0	
gdbm-libs		1.19
glib2	2.36.3	2.74.7
glibc	2.17	2.34
glibc-common	2.17	2.34
glibc-minimal-langpack		2.34
gmp	6.0.0	6.2.1
gnupg2	2.0.28	
gnupg2-minimal		2.3.7
gpgme	1.4.3	1.15.1
grep	2.20	3.8
gzip	1.5	
info	5.1	
json-c		0.14
keyutils-libs	1.5.8	1.6.3

套件	AL1 容器	AL2023 容器
krb5-libs	1.15.1	1.21.3
libacl	2.2.49	2.3.1
libarchive		3.7.4
libassuan	2.0.3	2.5.5
libattr	2.4.46	2.5.1
libblkid		2.37.4
libcap	2.16	2.48
libcap-ng		0.8.2
libcom_err	1.43.5	1.46.5
libcomps		0.1.20
libcurl	7.61.1	
libcurl-minimal		8.5.0
libdnf		0.69.0
libffi	3.0.13	3.4.4
libgcc		11.4.1
libgcc72	7.2.1	
libgcrypt	1.5.3	1.10.2
libgomp		11.4.1
libgpg-error	1.11	1.42
libicu	50.2	

套件	AL1 容器	AL2023 容器
libidn2	2.3.0	2.3.2
libmodulemd		2.13.0
libmount		2.37.4
libnghttp2	1.33.0	1.59.0
libpsl	0.6.2	0.21.1
librepo		1.14.5
libreport-filesystem		2.15.2
libselinux	2.1.10	3.4
libsepol	2.1.7	3.4
libsigsegv		2.13
libsmartcols		2.37.4
libsolv		0.7.22
libssh2	1.4.2	
libstdc++		11.4.1
libstdc++72	7.2.1	
libtasn1	2.3	4.19.0
libunistring	0.9.3	0.9.10
libuuid		2.37.4
libverto	0.2.5	0.3.2
libxcrypt		4.4.33

套件	AL1 容器	AL2023 容器
libxml2	2.9.1	2.10.4
libxml2-python27	2.9.1	
libyaml		0.2.5
libzstd		1.5.5
lua	5.1.4	
lua-libs		5.4.4
lz4-libs		1.9.4
make	3.82	
mpfr		4.1.0
ncurses	5.7	
ncurses-base	5.7	6.2
ncurses-libs	5.7	6.2
npth		1.6
nspr	4.25.0	
nss	3.53.1	
nss-pem	1.0.3	
nss-softokn	3.53.1	
nss-softokn-freebl	3.53.1	
nss-sysinit	3.53.1	
nss-tools	3.53.1	

套件	AL1 容器	AL2023 容器
nss-util	3.53.1	
openldap	2.4.40	
openssl	1.0.2k	
openssl-libs		3.0.8
p11-kit	0.18.5	0.24.1
p11-kit-trust	0.18.5	0.24.1
pcre	8.21	
pcre2		10.40
pcre2-syntax		10.40
pinentry	0.7.6	
pkgconfig	0.27.1	
popt	1.13	1.18
pth	2.0.7	
publicsuffix-list-dafsa		20240212
python27	2.7.18	
python27-chardet	2.0.1	
python27-iniparse	0.3.1	
python27-kitchen	1.1.1	
python27-libs	2.7.18	
python27-pycurl	7.19.0	

套件	AL1 容器	AL2023 容器
python27-pygpgme	0.3	
python27-pyliblzma	0.5.3	
python27-pyxattr	0.5.0	
python27-urlgrabber	3.10	
python3		3.9.16
python3-dnf		4.14.0
python3-gpg		1.15.1
python3-hawkey		0.69.0
python3-libcomps		0.1.20
python3-libdnf		0.69.0
python3-libs		3.9.16
python3-pip-wheel		21.3.1
python3-rpm		4.16.1.3
python3-setuptools-wheel		59.6.0
readline	6.2	8.1
rpm	4.11.3	4.16.1.3
rpm-build-libs	4.11.3	4.16.1.3
rpm-libs	4.11.3	4.16.1.3
rpm-python27	4.11.3	
rpm-sign-libs		4.16.1.3

套件	AL1 容器	AL2023 容器
sed	4.2.1	4.8
setup	2.8.14	2.13.7
shared-mime-info	1.1	
sqlite	3.7.17	
sqlite-libs		3.40.0
sysctl-defaults	1.0	
system-release	2018.03	2023.6.20241031
tar	1.26	
tzdata	2023c	2024a
xz-libs	5.2.2	5.2.5
yum	3.4.3	4.14.0
yum-metadata-parser	1.1.4	
yum-plugin-ovl	1.1.31	
yum-plugin-priorities	1.1.31	
yum-utils	1.1.31	
zlib	1.2.8	1.2.11

AL2023 系統需求

本節說明使用 AL2023 的系統需求。

主題

執行 AL2023 的 CPU 需求

若要執行任何 AL2023 程式碼，所使用的處理器需要符合特定的最低需求。嘗試在不符合這些要求的 CPUs 上執行 AL2023 可能會在程式碼執行的極早期導致非法指令錯誤。

最低需求適用於 Amazon EC2 上的 AL2023、 容器中的 AL2023和 Amazon EC2 之外的 AL2023。

AL2023 的 ARM CPU 需求

所有 AL2023 `aarch64`(ARM) 二進位檔都是針對 64 位元而建置。沒有可用的 32 位元ARM二進位檔，因此需要 64 位元 ARM CPU。

> **Note**
>
> 對於 ARM 型執行個體，AL2023 僅支援使用 Graviton2 或更新版本處理器的執行個體類型。AL2023 不支援 A1 執行個體。

AL2023 需要符合 ARMv8.2 標準且具有密碼編譯延伸功能的處理器 (`ARMv8.2+crypto`)。的所有 AL2023 套件`aarch64`都是使用`-march=armv8.2-a+crypto`編譯器旗標建置。雖然我們在嘗試在舊版ARM處理器上執行 AL2023 程式碼時嘗試列印正常的錯誤訊息，但第一個錯誤訊息可能是非法的指示錯誤。

> **Note**
>
> 由於 AL2023 `aarch64`基本 CPU 需求， 之前的所有Raspberry Pi系統Raspberry Pi 5都不符合最低 CPU 需求。

AL2023 的 x86-64 CPU 需求

所有 AL2023 x86-64 二進位檔都是透過傳遞-march=x86-64-v2至編譯器，為x86-64架構的x86-64v2修訂而建置。

架構的x86-64v2修訂會在基準x86-64架構上新增下列 CPU 功能：

- CMPXCHG16B
- LAHF-SAHF
- POPCNT
- SSE3
- SSE4_1
- SSE4_2
- SSSE3

這大致映射到 2009 年或之後發行的x86-64處理器。範例包括 Intel Nehalem、Atom Silvermont、 以及 AMD JaguarVIA Nano和 Eden C 微架構。

在 Amazon EC2 中，所有 x86-64 執行個體類型都支援 x86-64v2，包括 M1、C1 和 M2 執行個體系列。

未建置 32 位元 x86 (i686) AL2023 二進位檔。雖然 AL2023 保留對執行 32 位元使用者空間二進位檔的支援，但此功能已棄用，且可能在未來的主要 Amazon Linux 版本中移除。如需詳細資訊，請參閱32 位元 x86 (i686) 套件。

執行 AL2023 的記憶體 (RAM) 需求

Amazon EC2 .nano 系列執行個體類型 (t2.nano、t3a.nano、 t3.nano和 t4g.nano) 具有 512 MB RAM，這是 AL2023 的最低需求。

> **Note**
>
> 雖然 512 MB 是最低需求，但這些執行個體類型會受到記憶體限制，且功能和效能可能會受到限制。

AL2023 映像尚未在 RAM 小於 512 MB 的系統上進行測試。在少於 512 MB RAM 的 中執行以 AL2023 為基礎的容器映像，將取決於容器化工作負載。

有些工作負載，例如在某些 AL2023 版本`dnf upgrade`之間，可能需要超過 512 MB 的 RAM。因此，AL2023.3 版本`zram`已針對 RAM 小於 800 MB 的執行個體，預設推出啟用。對於容器化工作負載，這表示某些工作負載可能會在具有此記憶體數量的 AL2023 執行個體上執行良好，但在限制在此記憶體用量的容器中執行時失敗。

對於 RAM 小於 800 MB 的執行個體類型，AL2023 (截至 AL2023.3 或更新版本) 預設啟用基於 `zram` 的交換。記憶體小於 800 MB 的 Amazon EC2 執行個體類型範例包括 `t4g.nano`、`t3a.nano`、`t2.nano`、 `t3.nano`和 `t1.micro`。這表示這些執行個體類型的記憶體不足案例較少，因為 AL2023 會隨需壓縮和解壓縮記憶體分頁。如此就能使用之前需要具有更多記憶體的執行個體類型，且會犧牲執行壓縮所需 CPU 使用量的工作負載。

Amazon Linux 核心

Amazon Linux 2023 (AL2023) 在每年第一Q1發行新的長期支援 (LTS) 核心，以上游 Linux 社群的年度 LTS 核心為基礎。每個新的 LTS 核心都提供上游 Linux 核心的最新安全性修正、效能改善、硬體支援和功能。

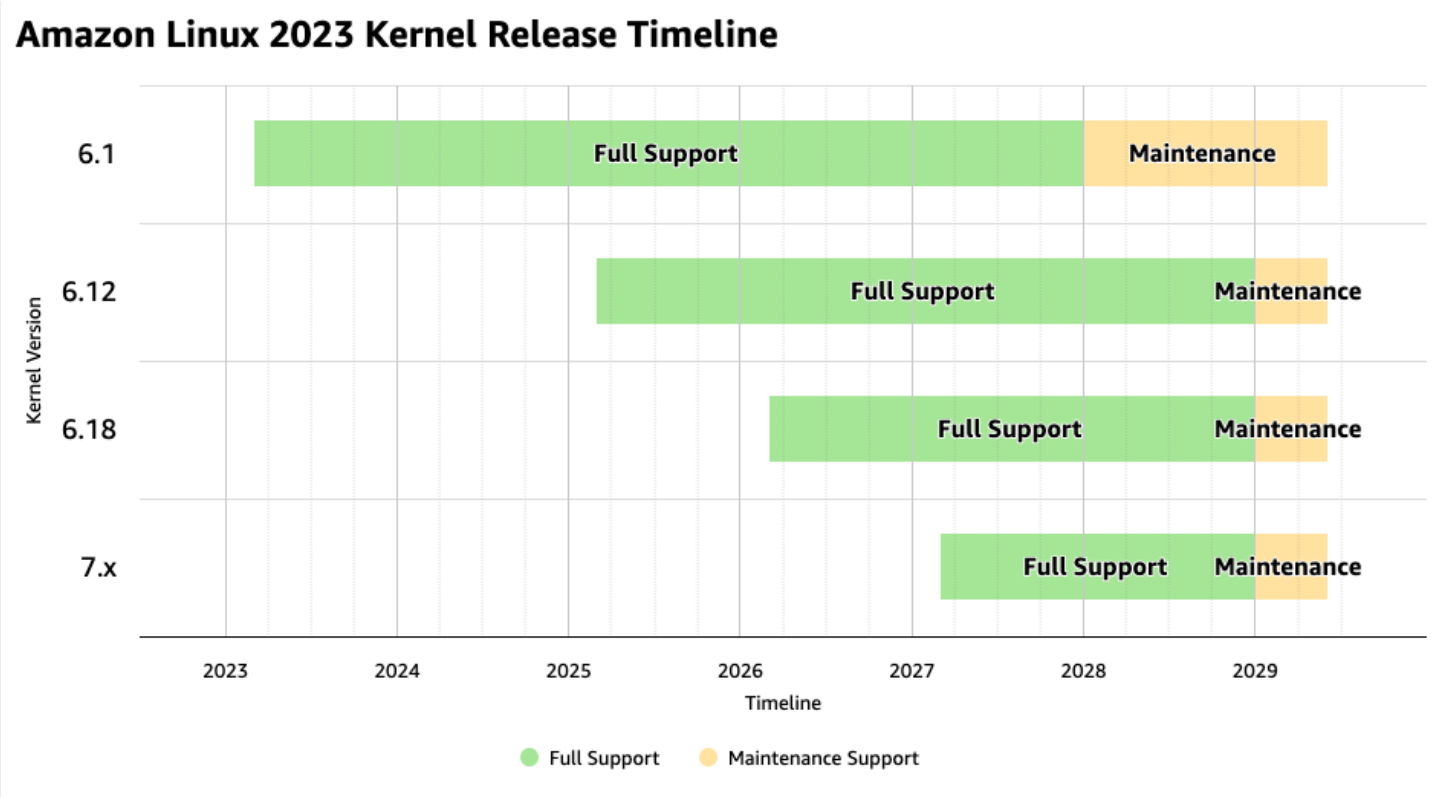
核心生命週期

每個 AL2023 LTS 核心支援四年，分為兩個階段：

1. 完整支援（第 1–2 年）– 核心透過上游 LTS 核心上的定期重新基礎收到所有 CVE 嚴重性的修正。此階段符合上游 Linux 社群的 LTS 支援時段。如果上游擴展 LTS 支援時段，Amazon Linux 將遵循相應的要求。
2. 維護支援（第 3–4 年）– 在上游 LTS 支援期間結束後，Amazon Linux 團隊會繼續向後移植主要修正重大和重要的 CVEs(CVSS 分數 7.0 以上），以及已知的漏洞。在此階段中，不會向後移植低嚴重性和中等嚴重性CVEs。

四年後，核心達到生命週期結束，不再收到安全更新。較舊的核心仍可透過儲存庫使用，您可以繼續使用它們。您不應預期任何進一步的修補程式或修正。我們建議在此日期之前升級至支援的核心。在核心的支援結束日期之後，不會再更新核心特定的 AMIs。

下表顯示目前 Amazon Linux LTS 核心的支援時間表：



核心更新

從 2026 年 6 月開始，AL2023 將每年更新預設核心。`al2023-ami-kernel-default` 一組 AMIs將更新為最新的 LTS 核心，因此新啟動的執行個體將會出現新的核心版本。

執行中的執行個體不會自動更新為新的核心。若要升級現有執行個體上的核心，您必須明確安裝新的核心套件並重新啟動。如需詳細資訊，請參閱在 AL2023 上更新 Linux 核心。

我們強烈建議您立即採用新的核心，以受益於最新的安全性和效能改進。隨著時間的推移，較舊的核心會收到較少且較慢的安全性修正。將核心更新納入現有的測試和部署管道，以在推出生產環境之前驗證相容性。

分佈之間的核心版本重疊

為了簡化 Amazon Linux 發行版本之間的遷移，目前和下一個 Amazon Linux 發行版本上至少有一個核心版本可用。這可讓您先升級到現有分發上的新核心、驗證工作負載，然後遷移到新的分發，並確信該處有相同的核心版本可用。

CVE 處理

AL2023 處理核心 CVEs如下所示：

- 關鍵和重要的 CVEs (CVSS 7.0 及更高版本） 和已知的利用漏洞會回溯到所有支援的核心。
- 在整個支援階段，透過定期上游 LTS 重新基底處理低和中 CVEs(CVSS 低於 7.0)。在維護支援階段，不會回溯這些 CVEs。如果上游 LTS 重新基底未在 60 天內解析低或中 CVE，則會在 Amazon Linux 安全中心將其標記為「未計劃修正」。

執行最新的可用核心是取得最新安全性、效能和功能更新的最佳方式。

您應該做什麼

1. 為您的應用程式實作持續整合 (CI) – 在對您的生產設定進行任何變更之前，請確定已在測試階段正確驗證。在驗證中包含核心更新。
2. 持續使用最新的核心 – 在推出時立即採用每個新的年度 LTS 核心。較新的核心更快收到安全性修正。使用`al2023-ami-kernel-default`一系列 AMIs來自動位於 Amazon Linux 團隊建議的核心版本。
3. 自動化更新 – 使用 AWS Systems Manager 等工具來管理整個機群的核心更新。
4. 規劃重新啟動 – 每次核心更新都需要重新啟動。在維護排程中建置重新啟動時段。

AL2023 圖形桌面

Amazon Linux 2023 提供選用、輕量型、雲端最佳化的圖形介面，以 2023.7 版的 GNOME 為基礎。此現代桌面環境透過 Firefox 等內建工具提供增強的生產力功能，以安全地瀏覽，同時維持 Amazon DCV 和 VNC 對遠端存取的支援。

> **Note**
>
> AL2023 會密切遵循上游 Firefox 延伸支援版本 (ESR) 版本，並盡快更新至下一個 ESR。如需詳細資訊，請參閱 Firefox ESR 版本行事曆和 Firefox 版本備註。

相關主題

如需安裝圖形桌面環境的詳細資訊，請參閱下列文件：

- 教學課程：在 AL2023 上安裝 GNOME 桌面環境

Amazon Linux 的補充套件

本節介紹 Amazon Linux (SPAL) 的補充套件，並概述其優點和限制，以及報告套件相關問題的指導方針。

主題

什麼是 Amazon Linux （或 SPAL) 的補充套件？

Amazon Linux (SPAL) 的補充套件是專用套件儲存庫，可讓您存取衍生自 Enterprise Linux 9 (EPEL9) 額外套件的數千個額外套件。這些套件補充了核心 Amazon Linux 2023 中可用的現有軟體。

SPAL 提供與 AL2023 相容的預先建置套件，可簡化軟體部署，無需客戶自行從原始碼建置套件。這可節省軟體安裝過程中的時間和精力。

> **Note**
>
> SPAL 適用於所有AWS商業區域，包括 AWSGovCloud (US) 區域和中國，適用於發行版本為 `2023.9.20251117`或更新版本的 AL2023 執行個體。

優勢

SPAL 為 Amazon Linux 2023 使用者提供了幾個關鍵優勢：

- 擴展 AL2023 使用案例 — 存取核心 AL2023 儲存庫以外的熱門額外套件，例如 `pandoc` `GDAL`或 `drbd-utils`，可讓客戶支援多個業務和開發需求。
- 簡化 AL2023 套件管理 — 透過提供為 AL2023 預先建置的套件，可消除從來源建置其他套件的程序，節省時間並降低編譯錯誤的風險。

- 簡化 AL2 到 AL2023 遷移 — SPAL 儲存庫可讓工作負載從 AL2 無縫遷移到 AL2023，包括依賴 EPEL7 套件的工作負載。

支援 SPAL 套件

SPAL 套件不受與核心 AL2023 套件相同層級的支援所涵蓋，這些套件在 Amazon Linux 2023 的整個生命週期內都獲得支援。

> **Important**
>
> 使用 SPAL 之前，客戶必須仔細評估下列考量：
>
> - AWSSupport Plans 不涵蓋 SPAL 套件。
> - SPAL 套件從上游 EPEL9 提供「原樣」。
> - SPAL 套件將不會收到AWS CVE 安全性追蹤。
> - SPAL 套件在可用時，只會從上游 EPEL9 接收安全性修補程式和錯誤修正。

報告套件相關問題

如果您遇到 SPAL 套件的問題，建議您先檢查上游 EPEL9 儲存庫中對應的套件是否發生相同的問題。為此，請參閱上游 EPEL 儲存庫上的問題清單。

如果問題不存在於 EPEL9，請在 Amazon Linux 2023 GitHub 儲存庫中建立問題，因為這表示問題專屬於 SPAL 套件建置或組態。

此方法可確保由適當的維護者解決問題，並有助於 SPAL 和上游 EPEL 套件的整體品質。

> **Note**
>
> 回報的問題將盡力處理。

相關主題

如需有關在系統上設定 SPAL 的資訊，請參閱下列文件頁面：

- 教學課程：在 AL2023 上設定 SPAL 儲存庫

教學課程：在 AL2023 上設定 SPAL 儲存庫

適用於 Amazon Linux (SPAL) 的補充套件是 AL2023 的額外套件儲存庫，可讓客戶存取數千個開放原始碼套件。

下列教學課程可協助您在 AL2023 執行個體上設定 SPAL 儲存庫。透過安裝儲存庫，您可以存取 SPAL 中提供的所有 RPM 套件。安裝後，您可以使用套件管理員在系統上安裝和使用這些套件。

目錄

先決條件

本教學假設您已使用 AL2023 發行版本 2023.9.20251117 或更新版本啟動執行個體。如需詳細資訊，請參閱 Amazon EC2 上的 AL2023和 更新 AL2023頁面。

檢查先決條件

- 若要驗證執行個體是否符合先決條件，您可以檢查system-release系統上安裝的 版本。

  若要檢查套件的版本，您可以使用下列命令。

  ```
  [ec2-user ~]$ rpm -qi system-release
  ```

  命令會顯示套件的相關資訊，包括主要版本。

  ```
  Name        : system-release
  Version     : 2023.9.20251117
  ...
  ```

> **Note**
>
> 請務必system-release安裝最新版本的 。您可以執行 sudo dnf upgrade 以更新至最新版本。

在您的系統上安裝 SPAL

1. 在您的系統上安裝 spal-release套件。這會將.repo組態檔案和 GPG 金鑰新增至您的系統。

```
[ec2-user ~]$ sudo dnf install spal-release
```

> **Note**
>
> 在安裝期間，會顯示支援陳述式。陳述式說明 SPAL 的支援範圍和限制。請花時間仔細檢閱此資訊。

2. 確認 SPAL 儲存庫組態已成功新增至您的系統。

```
[ec2-user ~]$ cat /etc/yum.repos.d/amazonlinux-spal.repo
```

您應該會看到系統上設定的兩個儲存庫： amazonlinux-spal 和 amazonlinux-spal-source

您也可以執行 來檢查設定的儲存庫清單dnf repolist。

```
[ec2-user ~]$ dnf repolist --all
```

> **Note**
>
> 需要 --all旗標才能查看啟用和停用的儲存庫。

這兩個 SPAL 儲存庫都應該可供使用。請注意，Amazon Linux 2023 SPAL 儲存庫 - 來源套件儲存庫預設為停用。

```
repo id                    repo name
 status
amazonlinux-spal           Amazon Linux 2023 SPAL repository
 enabled
amazonlinux-spal-source    Amazon Linux 2023 SPAL repository - Source packages
 disabled
```

3. （選用）啟用來源儲存庫。

> **Note**
>
> RPM 來源 (SRPM) 儲存庫通常預設為停用，因為開發人員主要用於建置套件，而非最終使用者用於軟體安裝。當您使用需要來源套件的命令時，DNF 會自動啟用來源儲存庫，例如 `dnf download --source package`。
> 您不需要為一次性來源套件操作手動啟用來源儲存庫。只有在您想要在系統上從 SPAL 重建 SRPMs時，才執行此步驟。

若要在您的系統上永久啟用 Amazon Linux 2023 SPAL 儲存庫 - 來源套件儲存庫，請執行下列命令：

```
[ec2-user ~]$ sudo dnf config-manager --enable amazonlinux-spal-source
```

安裝 SPAL 套件

- 執行 `dnf install`命令，在您的系統上安裝 SPAL 套件。

```
[ec2-user ~]$ sudo dnf install package
```

> **Note**
>
> 您可以使用 `dnf list` 查看 SPAL 套件的完整清單。
>
> ```
> [ec2-user ~]$ dnf list --repo=amazonlinux-spal
> ```

> **Note**
>
> SPAL 是版本控制的儲存庫。請務必`system-release`安裝最新版本的 ，以查看最新的套件清單。
> 如需確定性更新的詳細資訊，您可以檢查 透過 AL2023 上的版本控制儲存庫進行確定性升級

從系統解除安裝 SPAL 儲存庫

1. 使用 `dnf remove`命令移除 SPAL 儲存庫組態。

```
[ec2-user ~]$ sudo dnf remove spal-release
```

2. 透過執行 `dnf repolist`命令來驗證儲存庫是否已移除。

```
[ec2-user ~]$ dnf repolist
```

> **Important**
>
> 從系統移除 SPAL 儲存庫組態並不會移除安裝在系統上的任何 SPAL 套件。

相關主題

如需 Amazon Linux 儲存庫補充套件的詳細資訊，請參閱下列文件：

- Amazon Linux 的補充套件

在 AL2023 上執行應用程式

本節涵蓋在 Amazon Linux 2023 (AL2023) 上執行應用程式的方法，包括管理應用程式啟動 （ 和重新啟動 ） 的時間，以及控制資源用量。

主題

使用 限制 AL2023 中的程序資源使用量 systemd

在 Amazon Linux 2023 (AL2023) 上，建議使用 `systemd` 來控制程序或程序群組可以使用哪些資源。使用 `systemd` 是功能強大且易於使用的替代，可用於`cgroups`手動操作或使用 等公用程式`cpulimit`，先前僅適用於第三方EPEL儲存庫中的 Amazon Linux。

如需完整資訊，請參閱 systemd.resource-control 的上游`systemd`文件，或 AL2023 執行個體`systemd.resource-control`上的 man頁面。 https://www.freedesktop.org/software/systemd/man/latest/systemd.resource-control.html

以下範例將使用 `stress-ng` CPU 壓力測試 （ 來自 `stress-ng`套件 ） 來模擬 CPU 繁重的應用程式，以及`memcached`模擬記憶體繁重的應用程式。

下列範例涵蓋將 CPU 限制放置在一次性命令上，以及將記憶體限制放在服務上。`systemd` 提供的大多數資源限制條件都可以在`systemd`將執行程序的任何位置使用，並且可以同時使用多個資源限制條件。以下範例僅限於單一限制條件，以供 illustrive 使用。

使用 **`systemd-run`** 執行一次性命令的資源控制

雖然通常與系統服務相關聯，但非根使用者`systemd`也可以使用 來執行服務、排程計時器或執行一次性程序。在下列範例中，我們將使用 `stress-ng`做為範例應用程式。在第一個範例中，我們將在`ec2-user`預設帳戶中使用 `systemd-run` 執行它，而在第二個範例中，我們將對其 CPU 使用量設定限制。

Example在命令列systemd-run上使用 來執行程序，而非限制資源使用量

1. 確保套件已安裝，因為我們將使用此`stress-ng`套件做為範例。

```
[ec2-user ~]$ sudo dnf install -y stress-ng
```

2. 使用 systemd-run執行 10 秒的 CPU 壓力測試，而不會限制其可以使用的 CPU 數量。

```
[ec2-user ~]$ systemd-run --user --tty --wait --property=CPUAccounting=1 stress-ng
 --cpu 1 --timeout 10
Running as unit: run-u6.service
Press ^] three times within 1s to disconnect TTY.
stress-ng: info:  [339368] setting to a 10 second run per stressor
stress-ng: info:  [339368] dispatching hogs: 1 cpu
stress-ng: info:  [339368] successful run completed in 10.00s
Finished with result: success
Main processes terminated with: code=exited/status=0
Service runtime: 10.068s
CPU time consumed: 9.060s
```

--user 選項會指示 systemd-run以登入身分的使用者身分執行命令、 --tty選項表示TTY已連接 、 --wait 表示等待服務完成，而 --property=CPUAccounting=1選項會指示systemd-run記錄執行程序所使用的 CPU 時間。--property 命令列選項可用來傳遞可在systemd.unit組態檔案中設定的systemd-run設定。

當指示在 CPU 上放置負載時，stress-ng程式將使用所有可用的 CPU 時間，在您要求它執行的期間執行其測試。對於真實世界應用程式，可能想要限制程序的總執行時間。在下列範例中，我們會要求stress-ng執行的時間比使用 對其設定的最大持續時間限制更長systemd-run。

Example在命令列systemd-run上使用 來執行程序，將 CPU 用量限制為 1 秒

1. 確保stress-ng已安裝 以執行此範例。
2. LimitCPU 屬性相當於 ulimit -t，其將限制允許此程序在 CPU 上使用的時間上限。在此情況下，由於我們要求執行 10 秒壓力，而且將 CPU 用量限制為 1 秒，因此命令將收到SIGXCPU訊號並失敗。

```
[ec2-user ~]$ systemd-run --user --tty --wait --property=CPUAccounting=1 --
property=LimitCPU=1 stress-ng --cpu 1 --timeout 10
Running as unit: run-u12.service
Press ^] three times within 1s to disconnect TTY.
stress-ng: info:  [340349] setting to a 10 second run per stressor
stress-ng: info:  [340349] dispatching hogs: 1 cpu
```

```
stress-ng: fail:  [340349] cpu instance 0 corrupted bogo-ops counter, 1370 vs 0
stress-ng: fail:  [340349] cpu instance 0 hash error in bogo-ops counter and run
 flag, 3250129726 vs 0
stress-ng: fail:  [340349] metrics-check: stressor metrics corrupted, data is
 compromised
stress-ng: info:  [340349] unsuccessful run completed in 1.14s
Finished with result: exit-code
Main processes terminated with: code=exited/status=2
Service runtime: 1.201s
CPU time consumed: 1.008s
```

更常見的是，您可能想要限制特定程序可以使用的 CPU 時間百分比。在下列範例中，我們將限制 可以使用的 CPU 時間百分比stress-ng。對於真實世界的服務，建議您限制背景程序可以使用的 CPU 時間上限百分比，以便為提供使用者請求的程序釋出資源。

Example使用 systemd-run 將程序限制在一個 CPU 上 10% 的 CPU 時間

1. 確保stress-ng已安裝 以執行此範例。
2. 我們將使用 CPUQuota 屬性來告知 ，systemd-run限制要執行之命令的 CPU 使用量。我們不會限制程序可以執行的時間量，以及程序可以使用的 CPU 數量。

```
[ec2-user ~]$ systemd-run --user --tty --wait --property=CPUAccounting=1 --
property=CPUQuota=10% stress-ng --cpu 1 --timeout 10
Running as unit: run-u13.service
Press ^] three times within 1s to disconnect TTY.
stress-ng: info:  [340664] setting to a 10 second run per stressor
stress-ng: info:  [340664] dispatching hogs: 1 cpu
stress-ng: info:  [340664] successful run completed in 10.08s
Finished with result: success
Main processes terminated with: code=exited/status=0
Service runtime: 10.140s
CPU time consumed: 1.014s
```

請注意，CPU會計如何告訴我們，當服務執行 10 秒時，它只會耗用 1 秒的實際 CPU 時間。

設定 以systemd限制 CPU、記憶體、聯網和 IO 的資源用量的方法有很多。如需完整systemd文件，請參閱 systemd.resource-control 的上游文件，或 AL2023 執行個體systemd.resource-control上的 man頁面。

在幕後， systemd 使用 Linux 核心的功能，例如 cgroups實作這些限制，同時避免您手動設定這些限制。的 Linux 核心文件cgroup-v2包含有關cgroups工作的廣泛詳細資訊。

systemd 服務中的資源控制

有幾個參數可以新增到 systemd 服務的 [Service]區段，以控制系統資源使用量。這些包括硬性限制和軟性限制。如需每個選項的確切行為，請參閱 systemd.resource-control 的上游systemd文件，或 AL2023 執行個體systemd.resource-control上的 man頁面。

常用的限制是MemoryHigh指定記憶體用量的限流限制、MemoryMax設定硬性上限（達到上限後，即叫用 OOM Killer) 和 CPUQuota（如上一節所示）。您也可以設定權重和優先順序，而不是固定數字。

Example使用 systemd 設定 服務的記憶體用量限制

在此範例中，我們將為 設定硬性記憶體用量限制memcached，即簡單的鍵值快取，並顯示如何針對該服務叫用 OOM Killer，而不是整個系統。

1. 首先，我們需要安裝此範例所需的套件。

```
[ec2-user ~]$ sudo dnf install -y memcached libmemcached-awesome-tools
```

2. 啟用 ，memcached.service然後啟動 服務，讓 memcached 執行。

```
[ec2-user ~]$ sudo systemctl enable memcached.service
Created symlink /etc/systemd/system/multi-user.target.wants/memcached.service # /
usr/lib/systemd/system/memcached.service.
[ec2-user ~]$ sudo systemctl start memcached.service
```

3. 檢查 memcached.service是否正在執行。

```
[ec2-user ~]$ sudo systemctl status memcached.service
# memcached.service - memcached daemon
     Loaded: loaded (/usr/lib/systemd/system/memcached.service; enabled; preset:
 disabled)
     Active: active (running) since Fri 2025-01-31 22:36:42 UTC; 1s ago
   Main PID: 356294 (memcached)
      Tasks: 10 (limit: 18907)
     Memory: 1.8M
        CPU: 20ms
     CGroup: /system.slice/memcached.service
     ##356294 /usr/bin/memcached -p 11211 -u memcached -m 64 -c 1024 -l
 127.0.0.1,::1
```

```
Jan 31 22:35:36 ip-1-2-3-4.us-west-2.compute.internal systemd[1]: Started
 memcached.service - memcached daemon.
```

4. 現在memcached已安裝並執行 ，我們可以觀察它是否可運作，方法是將一些隨機資料插入快取

   在 /etc/sysconfig/memcached CACHESIZE變數中，預設會設為 64，這表示 64 MB。透過將比快取大小上限更多的資料插入快取，我們可以看到快取已填滿，而某些項目會使用 移出memcached-tool，而 memcached.service 使用的記憶體約為 64MB。

```
[ec2-user ~]$ for i in $(seq 1 150); do dd if=/dev/random of=$i bs=512k count=1;
 memcp -s localhost $i; done
[ec2-user ~]$ memcached-tool localhost display
  #  Item_Size  Max_age   Pages   Count   Full?  Evicted Evict_Time OOM
  2     120B         0s       1       0      no        0        0    0
 39   512.0K         4s      63     126     yes       24        2    0
[ec2-user ~]$ sudo systemctl status memcached.service
# memcached.service - memcached daemon
     Loaded: loaded (/usr/lib/systemd/system/memcached.service; enabled; preset:
 disabled)
     Active: active (running) since Fri 2025-01-31 22:36:42 UTC; 7min ago
   Main PID: 356294 (memcached)
      Tasks: 10 (limit: 18907)
     Memory: 66.7M
        CPU: 203ms
     CGroup: /system.slice/memcached.service
             ##356294 /usr/bin/memcached -p 11211 -u memcached -m 64 -c 1024 -l
 127.0.0.1,::1

Jan 31 22:36:42 ip-1-2-3-4.us-west-2.compute.internal systemd[1]: Started
 memcached.service - memcached daemon.
```

5. 使用 MemoryMax 屬性為 設定硬性限制memcached.service，如果命中，則會叫用 OOM Killer。將服務新增至覆寫檔案，即可為服務設定其他選項。您可以直接編輯/etc/systemd/system/memcached.service.d/override.conf檔案或使用 的 edit命令以互動方式完成此操作systemctl。

```
[ec2-user ~]$ sudo systemctl edit memcached.service
```

   將以下項目新增至覆寫，以設定服務的 32MB 記憶體硬性限制。

```
[Service]
MemoryMax=32M
```

6. systemd 告知 重新載入其組態

```
[ec2-user ~]$ sudo systemctl daemon-reload
```

7. 請注意，memcached.service 目前正在執行，記憶體限制為 32MB。

```
[ec2-user ~]$ sudo systemctl status memcached.service
# memcached.service - memcached daemon
     Loaded: loaded (/usr/lib/systemd/system/memcached.service; enabled; preset:
 disabled)
    Drop-In: /etc/systemd/system/memcached.service.d
             ##override.conf
     Active: active (running) since Fri 2025-01-31 23:09:13 UTC; 49s ago
   Main PID: 358423 (memcached)
      Tasks: 10 (limit: 18907)
     Memory: 1.8M (max: 32.0M available: 30.1M)
        CPU: 25ms
     CGroup: /system.slice/memcached.service
             ##358423 /usr/bin/memcached -p 11211 -u memcached -m 64 -c 1024 -l
 127.0.0.1,::1

Jan 31 23:09:13 ip-1-2-3-4.us-west-2.compute.internal systemd[1]: Started
 memcached.service - memcached daemon.
```

8. 服務會在使用少於 32MB 的記憶體時正常運作，我們可以檢查將少於 32MB 的隨機資料載入快取，然後檢查服務的狀態。

```
[ec2-user ~]$ for i in $(seq 1 30); do dd if=/dev/random of=$i bs=512k count=1;
 memcp -s localhost $i; done
```

```
[ec2-user ~]$ sudo systemctl status memcached.service
# memcached.service - memcached daemon
     Loaded: loaded (/usr/lib/systemd/system/memcached.service; enabled; preset:
 disabled)
    Drop-In: /etc/systemd/system/memcached.service.d
             ##override.conf
     Active: active (running) since Fri 2025-01-31 23:14:48 UTC; 3s ago
   Main PID: 359492 (memcached)
```

```
      Tasks: 10 (limit: 18907)
     Memory: 18.2M (max: 32.0M available: 13.7M)
        CPU: 42ms
     CGroup: /system.slice/memcached.service
             ##359492 /usr/bin/memcached -p 11211 -u memcached -m 64 -c 1024 -l
 127.0.0.1,::1

Jan 31 23:14:48 ip-1-2-3-4.us-west-2.compute.internal systemd[1]: Started
 memcached.service - memcached daemon.
```

9. 我們現在可以嘗試memcached使用預設memcached組態的完整 64MB 快取，讓 使用超過 32MB 的記憶體。 64MB

```
[ec2-user ~]$ for i in $(seq 1 150); do dd if=/dev/random of=$i bs=512k count=1;
 memcp -s localhost $i; done
```

您會發現在上述命令的某些時間點，memcached伺服器發生連線錯誤。這是因為 OOM Killer 已因我們對程序施加的限制而終止程序。系統的其餘部分將正常運作，OOM Killer 不會考慮任何其他程序，因為它只是我們限制memcached.service的 。

```
[ec2-user ~]$ sudo systemctl status memcached.service
# memcached.service - memcached daemon
     Loaded: loaded (/usr/lib/systemd/system/memcached.service; enabled; preset:
 disabled)
    Drop-In: /etc/systemd/system/memcached.service.d
             ##override.conf
     Active: failed (Result: oom-kill) since Fri 2025-01-31 23:20:28 UTC; 2s ago
   Duration: 2.901s
    Process: 360130 ExecStart=/usr/bin/memcached -p ${PORT} -u ${USER} -m
 ${CACHESIZE} -c ${MAXCONN} $OPTIONS (code=killed, signal=KILL)
   Main PID: 360130 (code=killed, signal=KILL)
        CPU: 94ms

Jan 31 23:20:25 ip-1-2-3-4.us-west-2.compute.internal systemd[1]: Started
 memcached.service - memcached daemon.
Jan 31 23:20:28 ip-1-2-3-4.us-west-2.compute.internal systemd[1]:
 memcached.service: A process of this unit has been killed by the OOM killer.
Jan 31 23:20:28 ip-1-2-3-4.us-west-2.compute.internal systemd[1]:
 memcached.service: Main process exited, code=killed, status=9/KILL
Jan 31 23:20:28 ip-1-2-3-4.us-west-2.compute.internal systemd[1]:
 memcached.service: Failed with result 'oom-kill'.
```

使用 限制 AL2023 中的程序資源使用量 cgroups

雖然建議使用 使用 進行資源控制 systemd，但本節涵蓋基本libcgroup-tools公用程式的基本使用量，以限制程序的 CPU 和記憶體使用量。這兩種方法都是使用 cpulimit公用程式的替代方案，先前在 中找到EPEL。

以下範例涵蓋執行stress-ng壓力測試（來自stress-ng套件），同時使用libcgroup-tools套件中的公用程式限制其 CPU 和記憶體用量，以及 中的調校sysfs。

在命令列**libcgroup-tools**上使用 來限制資源使用量

1. 安裝 libcgroup-tools 套裝服務。

```
[ec2-user ~]$ sudo dnf install libcgroup-tools
```

2. cgroup 使用 memory和 cpu控制器建立 ，並為其命名 (our-example-limits)。使用 -a和 -t選項，允許使用者ec2-user控制 的調校 cgroup

```
[ec2-user ~]$ sudo cgcreate -a ec2-user -t ec2-user -g memory,cpu:our-example-limits
```

現在有一個/sys/fs/cgroup/our-example-limits/目錄，其中包含可用於控制每個可調校的檔案。

> **Note**
>
> Amazon Linux 2 使用 cgroup-v1 cgroup-v2 而非用於 AL2023。在 AL2 上，sysfs路徑不同，而且 擁有的 /sys/fs/cgroup/memory/our-example-limits和 /sys/fs/cgroup/cpu/our-example-limits 目錄將ec2-user包含可用於控制 限制的檔案cgroup。

3. 將 中所有程序的記憶體用量限制cgroup在 1 億位元組。

```
[ec2-user ~]$ echo 100000000 > /sys/fs/cgroup/our-example-limits/memory.max
```

> **Note**
>
> Amazon Linux 2 使用 cgroup-v1而非 cgroup-v2 Amazon Linux 2023 使用的 。這表示某些可調整項目不同。若要限制 AL2 上的記憶體用量，請改用下列可調校功能。

```
[ec2-user ~]$ echo 10000000 > /sys/fs/cgroup/memory/our-example-limits/
memory.limit_in_bytes
```

4. 將 中所有程序的 CPU 用量限制cgroup為 10%。檔案的格式cpu.max為 $MAX $PERIOD，限制群組$MAX針對每個 消耗 $PERIOD。

```
[ec2-user ~]$ echo 10000 100000 > /sys/fs/cgroup/our-example-limits/cpu.max
```

   Amazon Linux 2 使用 cgroup-v1而非 cgroup-v2 Amazon Linux 2023 使用的 。這表示某些可調整項目不同，包括如何限制 CPU 用量。

5. 以下範例會在 中執行 our-example-limits stress-ng (可透過執行 來安裝dnf install -y stress-ng)cgroup。stress-ng 命令執行時，您可以使用 觀察top其限制為 10% CPU的時間。

```
[ec2-user ~]$ sudo cgexec -g memory,cpu:our-example-limits stress-ng --cpu 1
```

6. 移除 cgroup 進行清除

```
[ec2-user ~]$ sudo cgdelete -g memory,cpu:our-example-limits
```

的 Linux 核心文件cgroup-v2包含其運作方式的廣泛詳細資訊。cpu 和記憶體控制器的文件涵蓋了如何使用每個可調整選項的詳細資訊。

在 上使用 AL2023 AWS

您可以設定 AL2023 以與其他 搭配使用 AWS 服務。例如，您可以在啟動 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體時選擇 AL2023 AMI。

對於這些設定程序，您可以使用 AWS Identity and Access Management (IAM) 服務。如需 IAM 的完整資訊，請參閱下列參考資料：

- AWS Identity and Access Management (IAM)
- IAM 使用者指南

主題

- 入門 AWS
- Amazon EC2 上的 AL2023
- 在容器中使用 AL2023
- 上的 AL2023 AWS Elastic Beanstalk
- 在 中使用 AL2023 AWS CloudShell
- 使用以 AL2023 為基礎的 Amazon ECS AMIs 託管容器化工作負載
- 在 AL2023 上使用 Amazon Elastic File System
- 使用在 AL2023 上建置的 Amazon EMR
- 在 中使用 AL2023 AWS Lambda

入門 AWS

註冊 AWS 帳戶

如果您沒有 AWS 帳戶，請完成下列步驟來建立一個。

註冊 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或簡訊，並在電話鍵盤輸入驗證碼。

當您註冊 時 AWS 帳戶，AWS 帳戶根使用者會建立 。根使用者有權存取該帳戶中的所有 AWS 服務 和資源。作為安全最佳實務，請將管理存取權指派給使用者，並且僅使用根使用者來執行需要根使用者存取權的任務。

AWS 會在註冊程序完成後傳送確認電子郵件給您。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。

建立具有管理存取權的使用者

註冊 後 AWS 帳戶，請保護 AWS 帳戶根使用者、啟用 AWS IAM Identity Center和建立 管理使用者，以免將根使用者用於日常任務。

保護您的 AWS 帳戶根使用者

1. 選擇根使用者並輸入 AWS 帳戶 您的電子郵件地址，以帳戶擁有者AWS 管理主控台身分登入 。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)。

   如需說明，請參閱《IAM 使用者指南》中的為您的 AWS 帳戶 根使用者 （ 主控台 ） 啟用虛擬 MFA 裝置。

建立具有管理存取權的使用者

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 Identity Center 中，將管理存取權授予使用者。

   如需使用 IAM Identity Center 目錄 做為身分來源的教學課程，請參閱AWS IAM Identity Center《 使用者指南》中的使用預設值設定使用者存取權 IAM Identity Center 目錄。

以具有管理存取權的使用者身分登入

- 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 Identity Center 使用者登入的說明，請參閱AWS 登入 《 使用者指南》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 Identity Center 中，建立一個許可集來遵循套用最低權限的最佳實務。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的建立許可集。
2. 將使用者指派至群組，然後對該群組指派單一登入存取權。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的新增群組。

授予程式設計存取權

如果使用者想要與 AWS 外部互動，則需要程式設計存取 AWS 管理主控台。授予程式設計存取權的方式取決於正在存取的使用者類型 AWS。

若要授予使用者程式設計存取權，請選擇下列其中一個選項。

哪個使用者需要程式設計存取權？	到	根據
IAM	（建議）使用主控台登入資料做為臨時登入資料，以簽署對 AWS CLI、 AWS SDKs程式設計請求。 AWS APIs	請依照您要使用的介面所提供的指示操作。 • 如需 AWS CLI，請參閱AWS Command Line Interface 《 使用者指南》中的登入以進行 AWS 本機開發。 • AWS SDKs請參閱 AWS SDKs和工具參考指南中的登入以進行 AWS 本機開發。
人力資源身分 (IAM Identity Center 中管理的使用者)	使用暫時登入資料簽署對 AWS CLI、 AWS SDKs程式設計請求。 AWS APIs	請依照您要使用的介面所提供的指示操作。

哪個使用者需要程式設計存取權？	到	根據
		• 如需 AWS CLI，請參閱AWS Command Line Interface 《使用者指南》中的設定 AWS CLI 要使用 AWS IAM Identity Center的 。 • AWS SDKs、工具和 AWS APIs，請參閱 AWS SDKs 和工具參考指南中的 IAM Identity Center 身分驗證。
IAM	使用暫時登入資料簽署對 AWS CLI、 AWS SDKs程式設計請求。 AWS APIs	遵循《IAM 使用者指南》中將臨時登入資料與 AWS 資源搭配使用中的指示。
IAM	(不建議使用) 使用長期登入資料來簽署對 AWS CLI、 AWS SDKs程式設計請求。 AWS APIs	請依照您要使用的介面所提供的指示操作。 • 如需 AWS CLI，請參閱AWS Command Line Interface 《使用者指南》中的使用 IAM 使用者憑證進行身分驗證。 • AWS SDKs和工具，請參閱 AWS SDKs和工具參考指南中的使用長期憑證進行身分驗證。 • 對於 AWS APIs，請參閱《IAM 使用者指南》中的管理 IAM 使用者的存取金鑰。

Amazon EC2 上的 AL2023

使用下列其中一個程序來啟動具有 AL2023 AMI 的 Amazon EC2 執行個體。您可以選擇標準 AMI 或最小 AMI。如需標準 AMI 和最小 AMI 之間差異的詳細資訊，請參閱比較 AL2023 標準 (預設) 與最小 AMI。

主題

使用 Amazon EC2 主控台啟動 AL2023

使用 Amazon EC2 主控台來啟動 AL2023 AMI。

> **Note**
>
> 對於 ARM 型執行個體，AL2023 僅支援使用 Graviton2 或更新版本處理器的執行個體類型。AL2023 不支援 A1 執行個體。

請遵循下列步驟，以從 Amazon EC2 主控台並使用 AL2023 AMI 啟動 Amazon EC2 執行個體。

使用 AL2023 AMI 啟動 EC2 執行個體

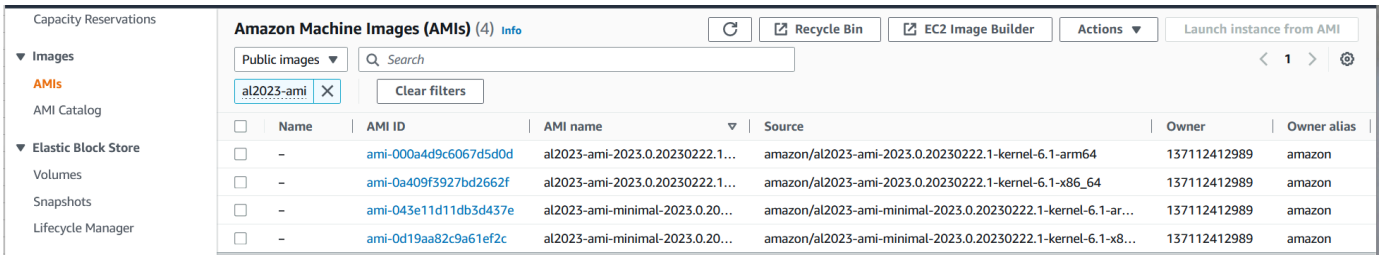
1. 前往 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。
2. 在導覽窗格中，選擇 AMI。
3. 從下拉式功能表中選擇 Public images (公有映像)。
4. 在搜尋欄位中輸入 **`al2023-ami`**。

> **Note**
>
> 確保 Amazon 出現在 Owner alias (所有者別名) 欄中。

5. 從清單選取映像。在 Source (來源) 下，您可以判斷 AMI 是標準或最小。AL2023 AMI 名稱可使用以下格式來解釋：

   ```
   'al2023-[ami || ami-minimal]-2023.0.[release build date].[build number]-kernel-[version number]-[arm64 || x86_64]'
   ```

6. 下圖顯示 AL2023 AMI 的部分清單。



如需啟動 Amazon EC2 執行個體的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 Linux 執行個體入門。 Amazon EC2

使用 SSM 參數和 啟動 AL2023 AWS CLI

在 中 AWS CLI，您可以使用 AMI 的 SSM 參數值來啟動新的 AL2023 執行個體。更具體來說，請使用下列清單的其中一個動態 SSM 參數值，並在 SSM 參數值/ 前面加 `/aws/service/ami-amazon-linux-latest/`。您可以在 AWS CLI中使用此方式來啟動執行個體。

- `al2023-ami-kernel-default-arm64` 適用於 arm64 架構
- `al2023-ami-minimal-kernel-default-arm64` 適用於 arm64 架構 (最小 AMI)
- `al2023-ami-kernel-default-x86_64` 適用於 x86_64 架構
- `al2023-ami-minimal-kernel-default-x86_64` 適用於 x86_64 架構 (最小 AMI)

> **Note**
>
> 每個##項目皆為範例參數。請以您自己的資訊取代。

```
$ aws ec2 run-instances \
  --image-id \
    resolve:ssm:/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-default-x86_64 \
  --instance-type m5.xlarge \
  --region us-east-1 \
  --key-name aws-key-us-east-1 \
  --security-group-ids sg-004a7650
```

`--image-id` 旗標指定 SSM 參數值。

`--instance-type` 旗標指定執行個體的類型和大小。此旗標必須與您選取的 AMI 類型相容。

`--region` 旗標會指定您建立執行個體 AWS 區域 的 。

`--key-name` 旗標指定用於連線至執行個體 AWS 區域的 金鑰。如果您沒有提供位於建立執行個體所在區域的金鑰，就無法使用 SSH 連線至執行個體。

`--security-group-ids` 旗標指定確定輸入和輸出網路流量存取權限的安全群組。

> **Important**
>
> AWS CLI 需要您指定現有的安全群組，以允許透過連接埠 從遠端機器存取執行個體TCP:22。如果沒有指定的安全群組，新執行個體會放置在預設安全群組中。在預設安全群組中，您的執行個體只能與 VPC 中的其他執行個體連線。

如需詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的啟動、列出及終止 Amazon EC2 執行個體。

使用 啟動最新的 AL2023 AMI CloudFormation

若要使用 啟動 AL2023 AMI CloudFormation，請使用下列其中一個範本。

> **Note**
>
> x86_64 和 Arm64 AMI 各自需要不同的執行個體類型。如需詳細資訊，請參閱 Amazon EC2 執行個體類型

JSON 範本：

```
{
  "Parameters": {
    "LatestAmiId": {
      "Type": "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>",
      "Default": "/aws/service/ami-amazon-linux-latest/al2023-ami-minimal-kernel-
default-x86_64"
    }
  },
  "Resources": {
    "MyEC2Instance": {
      "Type": "AWS::EC2::Instance",
      "Properties": {
        "InstanceType": "t2.large",
        "ImageId": {
          "Ref": "LatestAmiId"
        }
      }
    }
  }
}
```

YAML 範本：

```
Parameters:
  LatestAmiId:
    Type: 'AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>'
    Default: '/aws/service/ami-amazon-linux-latest/al2023-ami-minimal-kernel-default-
x86_64'

Resources:
  Instance:
    Type: 'AWS::EC2::Instance'
    Properties:
      InstanceType: 't2.large'
      ImageId: !Ref LatestAmiId
```

如有需要，請務必替換「預設」部分末尾的 AMI 參數。以下參數值可用於：

- `al2023-ami-kernel-6.1-arm64` 適用於 arm64 架構
- `al2023-ami-minimal-kernel-6.1-arm64` 適用於 arm64 架構 (最小 AMI)
- `al2023-ami-kernel-6.1-x86_64` 適用於 x86_64 架構

- `al2023-ami-minimal-kernel-6.1-x86_64` 適用於 x86_64 架構 (最小 AMI)

以下是動態核心規格。預設核心版本會隨著每次重大核心版本更新而自動變更。

- `al2023-ami-kernel-default-arm64` 適用於 arm64 架構
- `al2023-ami-minimal-kernel-default-arm64` 適用於 arm64 架構 (最小 AMI)
- `al2023-ami-kernel-default-x86_64` 適用於 x86_64 架構
- `al2023-ami-minimal-kernel-default-x86_64` 適用於 x86_64 架構 (最小 AMI)

使用特定 AMI ID 啟動 AL2023

您可以使用 AMI ID 啟動特定的 AL2023 AMI。您可以透過查看 Amazon EC2 控制台中的 AMI 列表，來確定需要哪個 AL2023 AMI ID。或者，您可以使用 AWS Systems Manager。如果您使用系統管理員，請務必從上一節列出的 AMI 別名中選取 AMI 別名。如需詳細資訊，請參閱使用 AWS Systems Manager 參數存放區查詢最新的 Amazon Linux AMI IDs。

AL2023 AMI 棄用和生命週期

每個新的 AL2023 發布都含有新的 AMI。當 AMI 註冊時，會標記棄用日期。每個 AL2023 AMI 的棄用日期都是自發行日起 90 天，以符合針對每個核心版本提供的 AL2023 上的核心即時修補 時間段。

> **Note**
>
> 90 天棄用日期是指個別 AMI，並不是指 AL2023 發行節奏 或產品支援期限。

如需 AMI 棄用的詳細資訊，請參閱《Amazon EC2 使用者指南》中的棄用 AMI。

定期使用更新的 AMI 啟動執行個體，以確保執行個體使用最新的安全性更新 (包括更新的核心) 啟動。如果您啟動舊版 AMI 並套用更新，執行個體會有一段時間沒有最新的安全性更新。為確保您使用最新的 AMI，建議使用 SSM 參數。

如需使用 SSM 參數啟動執行個體的詳細資訊，請參閱：

- 使用 SSM 參數和 啟動 AL2023 AWS CLI
- 使用 啟動最新的 AL2023 AMI CloudFormation

連線至 AL2023 執行個體

使用 SSH 或 AWS Systems Manager 連線到您的 AL2023 執行個體。

使用 SSH 連線至您的執行個體

如需如何使用 SSH 連線至執行個體的指示，請參閱《Amazon EC2 使用者指南》中的使用 SSH 連線至 Linux 執行個體。

使用 連線至您的執行個體 AWS Systems Manager

如需如何使用 AWS Systems Manager 連接到 AL2023 執行個體的指示，請參閱《Amazon EC2 使用者指南》中的使用 Session Manager 連接到 Linux 執行個體。

使用 Amazon EC2 Instance Connect

AL2023 AMI 不包含最小 AMI，預設會隨附安裝的 EC2 Instance Connect 代理程式。若要將 EC2 Instance Connect 與從最小 AMI 啟動的 AL2023 執行個體搭配使用，您必須安裝 `ec2-instance-connect`套件。如需使用 EC2 Instance Connect 的指示，請參閱《Amazon EC2 使用者指南》中的使用 EC2 Instance Connect 連線至 Linux 執行個體。

比較 AL2023 標準和最小 AMIs

您可以使用標準 （預設） 或最小 AL2023 AMI 來啟動 Amazon EC2 執行個體。如需如何使用標準或最小 AMI 類型啟動 Amazon EC2 執行個體的說明，請參閱 Amazon EC2 上的 AL2023。

標準 AL2023 AMI 隨附所有最常用的應用程式和工具。如果您想快速開始並且對自訂 AMI 不感興趣，建議使用標準 AMI。

最小 AL2023 AMI 是基本且簡化的版本，僅包含執行作業系統 (OS) 所需的最基本工具和公用程式。如果您希望將作業系統的使用空間降到最低，建議您使用最小 AMI。最小 AMI 可稍微降低磁碟空間使用率，並提高長期成本效益。如果您想要更小的作業系統，且不介意手動安裝工具和應用程序，則適合使用最小 AMI。

容器映像更接近套件組中的 AL2023 最小 AMI。

比較 Amazon Linux 2023 映像安裝的套件

AL2023 AMI、Minmal AMI 和 Container 映像上存在的 RPMs 比較。

套件	AMI	最小 AMI	容器
acl	2.3.1		
acpid	2.0.32		
alternatives	1.15	1.15	1.15
amazon-chrony-config	4.3	4.3	
amazon-ec2-net-utils	2.5.1	2.5.1	
amazon-linux-repo-cdn			2023.6.20241031
amazon-linux-repo-s3	2023.6.20241031	2023.6.20241031	
amazon-linux-sb-keys	2023.1	2023.1	
amazon-rpm-config	228		
amazon-ssm-agent	3.3.987.0		
amd-ucode-firmware	20210208 (noarch)	20210208 (noarch)	
at	3.1.23		
attr	2.5.1		
audit	3.0.6	3.0.6	
audit-libs	3.0.6	3.0.6	3.0.6

套件	AMI	最小 AMI	容器
aws-cfn-bootstrap	2.0		
awscli-2	2.15.30	2.15.30	
basesystem	11	11	11
bash	5.2.15	5.2.15	5.2.15
bash-completion	2.11		
bc	1.07.1		
bind-libs	9.18.28		
bind-license	9.18.28		
bind-utils	9.18.28		
binutils	2.39		
boost-filesystem	1.75.0		
boost-system	1.75.0		
boost-thread	1.75.0		
bzip2	1.0.8		
bzip2-libs	1.0.8	1.0.8	1.0.8
ca-certificates	2023.2.68	2023.2.68	2023.2.68
c-ares	1.19.1		
checkpolicy	3.4	3.4	
chkconfig	1.15		

套件	AMI	最小 AMI	容器
chrony	4.3	4.3	
cloud-init	22.2.2	22.2.2	
cloud-init-cfg-ec2	22.2.2	22.2.2	
cloud-utils-growpart	0.31	0.31	
coreutils	8.32	8.32	
coreutils-common	8.32	8.32	
coreutils-single			8.32
cpio	2.13	2.13	
cracklib	2.9.6	2.9.6	
cracklib-dicts	2.9.6	2.9.6	
crontabs	1.11		
crypto-policies	20220428	20220428	20220428
crypto-policies-scripts	20220428		
cryptsetup	2.6.1		
cryptsetup-libs	2.6.1	2.6.1	
curl-minimal	8.5.0	8.5.0	8.5.0
cyrus-sasl-lib	2.1.27	2.1.27	

套件	AMI	最小 AMI	容器
cyrus-sasl-plain	2.1.27		
dbus	1.12.28	1.12.28	
dbus-broker	32	32	
dbus-common	1.12.28	1.12.28	
dbus-libs	1.12.28	1.12.28	
device-mapper	1.02.185	1.02.185	
device-mapper-libs	1.02.185	1.02.185	
diffutils	3.8	3.8	
dnf	4.14.0	4.14.0	4.14.0
dnf-data	4.14.0	4.14.0	4.14.0
dnf-plugin-release-notification	1.2	1.2	
dnf-plugins-core	4.3.0	4.3.0	
dnf-plugin-support-info	1.2	1.2	
dnf-utils	4.3.0		
dosfstools	4.2		
dracut	055	055	

套件	AMI	最小 AMI	容器
dracut-config-ec2	3.0	3.0	
dracut-config-generic	055	055	
dwz	0.14		
dyninst	10.2.1		
e2fsprogs	1.46.5	1.46.5	
e2fsprogs-libs	1.46.5	1.46.5	
ec2-hibinit-agent	1.0.8		
ec2-instance-connect	1.1		
ec2-instance-connect-selinux	1.1		
ec2-utils	2.2.0	2.2.0	
ed	1.14.2		
efi-filesystem	5	5	
efi-srpm-macros	5		
efivar	38	38	
efivar-libs	38	38	

套件	AMI	最小 AMI	容器
elfutils-debuginfod-client	0.188		
elfutils-default-yama-scope	0.188	0.188	0.188
elfutils-libelf	0.188	0.188	0.188
elfutils-libs	0.188	0.188	0.188
ethtool	5.15		
expat	2.5.0	2.5.0	2.5.0
file	5.39	5.39	
file-libs	5.39	5.39	5.39
filesystem	3.14	3.14	3.14
findutils	4.8.0	4.8.0	
fonts-srpm-macros	2.0.5		
fstrm	0.6.1		
fuse-libs	2.9.9	2.9.9	
gawk	5.1.0	5.1.0	5.1.0
gdbm-libs	1.19	1.19	1.19
gdisk	1.0.8	1.0.8	
gettext	0.21	0.21	

套件	AMI	最小 AMI	容器
gettext-libs	0.21	0.21	
ghc-srpm-macros	1.5.0		
glib2	2.74.7	2.74.7	2.74.7
glibc	2.34	2.34	2.34
glibc-all-langpacks	2.34	2.34	
glibc-common	2.34	2.34	2.34
glibc-gconv-extra	2.34		
glibc-locale-source	2.34	2.34	
glibc-minimal-langpack			2.34
gmp	6.2.1	6.2.1	6.2.1
gnupg2-minimal	2.3.7	2.3.7	2.3.7
gnutls	3.8.0	3.8.0	
go-srpm-macros	3.2.0		
gpgme	1.15.1	1.15.1	1.15.1
gpm-libs	1.20.7		
grep	3.8	3.8	3.8
groff-base	1.22.4	1.22.4	
grub2-common	2.06	2.06	

套件	AMI	最小 AMI	容器
grub2-efi-aa64-ec2	2.06 (aarch64)	2.06 (aarch64)	
grub2-efi-x64-ec2	2.06 (x86_64)	2.06 (x86_64)	
grub2-pc-modules	2.06	2.06	
grub2-tools	2.06	2.06	
grub2-tools-minimal	2.06	2.06	
grubby	8.40	8.40	
gssproxy	0.8.4		
gzip	1.12	1.12	
hostname	3.23	3.23	
hunspell	1.7.0		
hunspell-en	0.20140811.1		
hunspell-en-GB	0.20140811.1		
hunspell-en-US	0.20140811.1		
hunspell-filesystem	1.7.0		
hwdata	0.384	0.384	
info	6.7		
inih	49	49	

套件	AMI	最小 AMI	容器
initscripts	10.09	10.09	
iproute	6.10.0	6.10.0	
iputils	20210202	20210202	
irqbalance	1.9.0	1.9.0	
jansson	2.14	2.14	
jemalloc	5.2.1		
jitterentropy	3.4.1	3.4.1	
jq	1.7.1	1.7.1	
json-c	0.14	0.14	0.14
kbd	2.4.0	2.4.0	
kbd-misc	2.4.0	2.4.0	
kernel	6.1.112	6.1.112	
kernel-libbpf	6.1.112	6.1.112	
kernel-livepatch-repo-s3	2023.6.20241031	2023.6.20241031	
kernel-srpm-macros	1.0		
kernel-tools	6.1.112		
keyutils	1.6.3		
keyutils-libs	1.6.3	1.6.3	1.6.3

套件	AMI	最小 AMI	容器
kmod	29	29	
kmod-libs	29	29	
kpatch-runtime	0.9.7		
krb5-libs	1.21.3	1.21.3	1.21.3
less	608	608	
libacl	2.3.1	2.3.1	2.3.1
libaio	0.3.111		
libarchive	3.7.4	3.7.4	3.7.4
libargon2	20171227	20171227	
libassuan	2.5.5	2.5.5	2.5.5
libattr	2.5.1	2.5.1	2.5.1
libbasicobjects	0.1.1		
libblkid	2.37.4	2.37.4	2.37.4
libcap	2.48	2.48	2.48
libcap-ng	0.8.2	0.8.2	0.8.2
libcbor	0.7.0	0.7.0	
libcollection	0.7.0		
libcom_err	1.46.5	1.46.5	1.46.5
libcomps	0.1.20	0.1.20	0.1.20
libconfig	1.7.2		

套件	AMI	最小 AMI	容器
libcurl-minimal	8.5.0	8.5.0	8.5.0
libdb	5.3.28	5.3.28	
libdhash	0.5.0		
libdnf	0.69.0	0.69.0	0.69.0
libeconf	0.4.0	0.4.0	
libedit	3.1	3.1	
libev	4.33		
libevent	2.1.12		
libfdisk	2.37.4	2.37.4	
libffi	3.4.4	3.4.4	3.4.4
libfido2	1.10.0	1.10.0	
libgcc	11.4.1	11.4.1	11.4.1
libgcrypt	1.10.2	1.10.2	1.10.2
libgomp	11.4.1	11.4.1	11.4.1
libgpg-error	1.42	1.42	1.42
libibverbs	48.0		
libidn2	2.3.2	2.3.2	2.3.2
libini_config	1.3.1		
libkcapi	1.4.0	1.4.0	
libkcapi-hmaccalc	1.4.0	1.4.0	

套件	AMI	最小 AMI	容器
libldb	2.6.2		
libmaxminddb	1.5.2		
libmetalink	0.1.3		
libmnl	1.0.4	1.0.4	
libmodulemd	2.13.0	2.13.0	2.13.0
libmount	2.37.4	2.37.4	2.37.4
libnfsidmap	2.5.4		
libnghttp2	1.59.0	1.59.0	1.59.0
libnl3	3.5.0		
libpath_utils	0.2.1		
libpcap	1.10.1		
libpipeline	1.5.3	1.5.3	
libpkgconf	1.8.0		
libpsl	0.21.1	0.21.1	0.21.1
libpwquality	1.4.4	1.4.4	
libref_array	0.1.5		
librepo	1.14.5	1.14.5	1.14.5
libreport-filesystem	2.15.2	2.15.2	2.15.2
libseccomp	2.5.3	2.5.3	
libselinux	3.4	3.4	3.4

套件	AMI	最小 AMI	容器
libselinux-utils	3.4	3.4	
libsemanage	3.4	3.4	
libsepol	3.4	3.4	3.4
libsigsegv	2.13	2.13	2.13
libsmartcols	2.37.4	2.37.4	2.37.4
libsolv	0.7.22	0.7.22	0.7.22
libss	1.46.5	1.46.5	
libsss_certmap	2.9.4		
libsss_idmap	2.9.4		
libsss_nss_idmap	2.9.4		
libsss_sudo	2.9.4		
libstdc++	11.4.1	11.4.1	11.4.1
libstoragemgmt	1.9.4		
libtalloc	2.3.4		
libtasn1	4.19.0	4.19.0	4.19.0
libtdb	1.4.7		
libtevent	0.13.0		
libtextstyle	0.21	0.21	
libtirpc	1.3.3		

套件	AMI	最小 AMI	容器
libunistring	0.9.10	0.9.10	0.9.10
libuser	0.63	0.63	
libutempter	1.2.1	1.2.1	
libuuid	2.37.4	2.37.4	2.37.4
libuv	1.47.0		
libverto	0.3.2	0.3.2	0.3.2
libverto-libev	0.3.2		
libxcrypt	4.4.33	4.4.33	4.4.33
libxml2	2.10.4	2.10.4	2.10.4
libyaml	0.2.5	0.2.5	0.2.5
libzstd	1.5.5	1.5.5	1.5.5
linux-firmware-whence	20210208 (noarch)	20210208 (noarch)	
lm_sensors-libs	3.6.0		
lmdb-libs	0.9.29		
logrotate	3.20.1	3.20.1	
lsof	4.94.0		
lua-libs	5.4.4	5.4.4	5.4.4
lua-srpm-macros	1		
lz4-libs	1.9.4	1.9.4	1.9.4
man-db	2.9.3	2.9.3	

套件	AMI	最小 AMI	容器
man-pages	5.10		
microcode_ctl	2.1 (x86_64)	2.1 (x86_64)	
mpfr	4.1.0	4.1.0	4.1.0
nano	5.8		
ncurses	6.2	6.2	
ncurses-base	6.2	6.2	6.2
ncurses-libs	6.2	6.2	6.2
nettle	3.8	3.8	
net-tools	2.0	2.0	
newt	0.52.21		
nfs-utils	2.5.4		
npth	1.6	1.6	1.6
nspr	4.35.0		
nss	3.90.0		
nss-softokn	3.90.0		
nss-softokn-freebl	3.90.0		
nss-sysinit	3.90.0		
nss-util	3.90.0		
ntsysv	1.15		
numactl-libs	2.0.14	2.0.14	

套件	AMI	最小 AMI	容器
ocaml-srpm-macros	6		
oniguruma	6.9.7.1	6.9.7.1	
openblas-srpm-macros	2		
openldap	2.4.57	2.4.57	
openssh	8.7p1	8.7p1	
openssh-clients	8.7p1	8.7p1	
openssh-server	8.7p1	8.7p1	
openssl	3.0.8	3.0.8	
openssl-libs	3.0.8	3.0.8	3.0.8
openssl-pkcs11	0.4.12	0.4.12	
os-prober	1.77	1.77	
p11-kit	0.24.1	0.24.1	0.24.1
p11-kit-trust	0.24.1	0.24.1	0.24.1
package-notes-srpm-macros	0.4		
pam	1.5.1	1.5.1	
parted	3.4		
passwd	0.80	0.80	
pciutils	3.7.0	3.7.0	

套件	AMI	最小 AMI	容器
pciutils-libs	3.7.0	3.7.0	
pcre2	10.40	10.40	10.40
pcre2-syntax	10.40	10.40	10.40
perl-Carp	1.50		
perl-Class-Struct	0.66		
perl-constant	1.33		
perl-DynaLoader	1.47		
perl-Encode	3.15		
perl-Errno	1.30		
perl-Exporter	5.74		
perl-Fcntl	1.13		
perl-File-Basename	2.85		
perl-File-Path	2.18		
perl-File-stat	1.09		
perl-File-Temp	0.231.100		
perl-Getopt-Long	2.52		
perl-Getopt-Std	1.12		
perl-HTTP-Tiny	0.078		

套件	AMI	最小 AMI	容器
perl-if	0.60.800		
perl-interpreter	5.32.1		
perl-IO	1.43		
perl-IPC-Open3	1.21		
perl-libs	5.32.1		
perl-MIME-Base64	3.16		
perl-mro	1.23		
perl-overload	1.31		
perl-overloading	0.02		
perl-parent	0.238		
perl-PathTools	3.78		
perl-Pod-Escapes	1.07		
perl-podlators	4.14		
perl-Pod-Perldoc	3.28.01		
perl-Pod-Simple	3.42		
perl-Pod-Usage	2.01		
perl-POSIX	1.94		

套件	AMI	最小 AMI	容器
perl-Scalar-List-Utils	1.56		
perl-SelectSaver	1.02		
perl-Socket	2.032		
perl-srpm-macros	1		
perl-Storable	3.21		
perl-subs	1.03		
perl-Symbol	1.08		
perl-Term-ANSIColor	5.01		
perl-Term-Cap	1.17		
perl-Text-ParseWords	3.30		
perl-Text-Tabs+Wrap	2021.0726		
perl-Time-Local	1.300		
perl-vars	1.05		
pkgconf	1.8.0		
pkgconf-m4	1.8.0		
pkgconf-pkg-config	1.8.0		

套件	AMI	最小 AMI	容器
policycoreutils	3.4	3.4	
policycoreutils-python-utils	3.4		
popt	1.18	1.18	1.18
procps-ng	3.3.17	3.3.17	
protobuf-c	1.4.1		
psacct	6.6.4		
psmisc	23.4	23.4	
publicsuffix-list-dafsa	20240212	20240212	20240212
python3	3.9.16	3.9.16	3.9.16
python3-attrs	20.3.0	20.3.0	
python3-audit	3.0.6	3.0.6	
python3-awscrt	0.19.19	0.19.19	
python3-babel	2.9.1	2.9.1	
python3-cffi	1.14.5	1.14.5	
python3-chardet	4.0.0	4.0.0	
python3-colorama	0.4.4	0.4.4	
python3-configobj	5.0.6	5.0.6	

套件	AMI	最小 AMI	容器
python3-cryptography	36.0.1	36.0.1	
python3-daemon	2.3.0		
python3-dateutil	2.8.1	2.8.1	
python3-dbus	1.2.18	1.2.18	
python3-distro	1.5.0	1.5.0	
python3-dnf	4.14.0	4.14.0	4.14.0
python3-dnf-plugins-core	4.3.0	4.3.0	
python3-docutils	0.16	0.16	
python3-gpg	1.15.1	1.15.1	1.15.1
python3-hawkey	0.69.0	0.69.0	0.69.0
python3-idna	2.10	2.10	
python3-jinja2	2.11.3	2.11.3	
python3-jmespath	0.10.0	0.10.0	
python3-jsonpatch	1.21	1.21	
python3-jsonpointer	2.0	2.0	

套件	AMI	最小 AMI	容器
python3-jsonschema	3.2.0	3.2.0	
python3-libcomps	0.1.20	0.1.20	0.1.20
python3-libdnf	0.69.0	0.69.0	0.69.0
python3-libs	3.9.16	3.9.16	3.9.16
python3-libselinux	3.4	3.4	
python3-libsemanage	3.4	3.4	
python3-libstoragemgmt	1.9.4		
python3-lockfile	0.12.2		
python3-markupsafe	1.1.1	1.1.1	
python3-netifaces	0.10.6	0.10.6	
python3-oauthlib	3.0.2	3.0.2	
python3-pip-wheel	21.3.1	21.3.1	21.3.1
python3-ply	3.11	3.11	
python3-policycoreutils	3.4	3.4	

套件	AMI	最小 AMI	容器
python3-prettytable	0.7.2	0.7.2	
python3-prompt-toolkit	3.0.24	3.0.24	
python3-pycparser	2.20	2.20	
python3-pyrsistent	0.17.3	0.17.3	
python3-pyserial	3.4	3.4	
python3-pysocks	1.7.1	1.7.1	
python3-pytz	2022.7.1	2022.7.1	
python3-pyyaml	5.4.1	5.4.1	
python3-requests	2.25.1	2.25.1	
python3-rpm	4.16.1.3	4.16.1.3	4.16.1.3
python3-ruamel-yaml	0.16.6	0.16.6	
python3-ruamel-yaml-clib	0.1.2	0.1.2	
python3-setools	4.4.1	4.4.1	
python3-setuptools	59.6.0	59.6.0	

套件	AMI	最小 AMI	容器
python3-setuptools-wheel	59.6.0	59.6.0	59.6.0
python3-six	1.15.0	1.15.0	
python3-systemd	235	235	
python3-urllib3	1.25.10	1.25.10	
python3-wcwidth	0.2.5	0.2.5	
python-chevron	0.13.1		
python-srpm-macros	3.9		
quota	4.06		
quota-nls	4.06		
readline	8.1	8.1	8.1
rng-tools	6.14	6.14	
rootfiles	8.1	8.1	
rpcbind	1.2.6		
rpm	4.16.1.3	4.16.1.3	4.16.1.3
rpm-build-libs	4.16.1.3	4.16.1.3	4.16.1.3
rpm-libs	4.16.1.3	4.16.1.3	4.16.1.3
rpm-plugin-selinux	4.16.1.3	4.16.1.3	

套件	AMI	最小 AMI	容器
rpm-plugin-systemd-inhibit	4.16.1.3	4.16.1.3	
rpm-sign-libs	4.16.1.3	4.16.1.3	4.16.1.3
rsync	3.2.6		
rust-srpm-macros	21		
sbsigntools	0.9.4	0.9.4	
screen	4.8.0		
sed	4.8	4.8	4.8
selinux-policy	38.1.45	38.1.45	
selinux-policy-targeted	38.1.45	38.1.45	
setup	2.13.7	2.13.7	2.13.7
shadow-utils	4.9	4.9	
slang	2.3.2		
sqlite-libs	3.40.0	3.40.0	3.40.0
sssd-client	2.9.4		
sssd-common	2.9.4		
sssd-kcm	2.9.4		
sssd-nfs-idmap	2.9.4		
strace	6.8		

套件	AMI	最小 AMI	容器
sudo	1.9.15	1.9.15	
sysctl-defaults	1.0	1.0	
sysstat	12.5.6		
systemd	252.23	252.23	
systemd-libs	252.23	252.23	
systemd-networkd	252.23	252.23	
systemd-pam	252.23	252.23	
systemd-resolved	252.23	252.23	
systemd-udev	252.23	252.23	
system-release	2023.6.20241031	2023.6.20241031	2023.6.20241031
systemtap-runtime	4.8		
tar	1.34	1.34	
tbb	2020.3		
tcpdump	4.99.1		
tcsh	6.24.07		
time	1.9		
traceroute	2.1.3		
tzdata	2024a	2024a	2024a

套件	AMI	最小 AMI	容器
unzip	6.0		
update-motd	2.2	2.2	
userspace-rcu	0.12.1	0.12.1	
util-linux	2.37.4	2.37.4	
util-linux-core	2.37.4	2.37.4	
vim-common	9.0.2153		
vim-data	9.0.2153	9.0.2153	
vim-enhanced	9.0.2153		
vim-filesystem	9.0.2153		
vim-minimal	9.0.2153	9.0.2153	
wget	1.21.3		
which	2.21	2.21	
words	3.0		
xfsdump	3.1.11		
xfsprogs	5.18.0	5.18.0	
xxd	9.0.2153		
xxhash-libs	0.8.0		
xz	5.2.5	5.2.5	
xz-libs	5.2.5	5.2.5	5.2.5
yum	4.14.0	4.14.0	4.14.0

套件	AMI	最小 AMI	容器
zip	3.0		
zlib	1.2.11	1.2.11	1.2.11
zram-generator	1.1.2	1.1.2	
zram-generator-defaults	1.1.2	1.1.2	
zstd	1.5.5	1.5.5	

在容器中使用 AL2023

> **Note**
>
> 如需如何使用 AL2023 在 Amazon ECS 上託管容器化工作負載的詳細資訊，請參閱 適用於 Amazon ECS 容器主機的 AL2023。

根據使用案例，AL2023 有多種可在容器內使用的方式。AL2023 基本容器映像 最類似 Amazon Linux 2 容器映像和 AL2023 最小 AMI。

對於進階使用者，我們提供 AL2023.2 版本中介紹的最小容器映像，以及描述如何建置準系統容器的文件。

AL2023 也可用於託管容器化工作負載，無論是以 AL2023 為基礎的容器映像，或是以其他 Linux 發行版為基礎的容器。您可以直接使用 適用於 Amazon ECS 容器主機的 AL2023 或使用提供的容器執行期套件。`docker`、`containerd` 和 `nerdctl` 套件可以在 AL2023 上安裝和使用。

主題

- 使用 AL2023 基礎容器映像
- AL2023 最小容器映像
- 建置裸機 AL2023 容器映像
- 比較 Amazon Linux 2023 容器映像安裝的套件
- 比較 Amazon Linux 2023 最小 AMI 與容器映像安裝的套件

使用 AL2023 基礎容器映像

AL2023 容器映像是從包含在 AL2023 AMI 中的相同軟體元件所建置。該映像可用於任何環境中並作為 Docker 工作負載的基礎映像。如果在 Amazon Elastic Compute Cloud (Amazon EC2) 中的應用程式中使用 Amazon Linux AMI，則可以使用 Amazon Linux 容器映像對應用程式進行容器化。

在本機開發環境中使用 Amazon Linux 容器映像，然後使用 Amazon Elastic Container Service (Amazon ECS) 將應用程式推送至 AWS 。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。

Amazon Linux 容器映像可在 Amazon ECR Public 使用。您可以透過指定的 AWS 代表提供 AL2023 的意見回饋，或在 GitHub 的 amazon-linux-2023 儲存庫中提出問題。

從 Amazon ECR Public 中提取 Amazon Linux 容器映像

1. 向 Amazon Linux Public 登錄檔驗證您的 Docker 用戶端。驗證字符有效時間為 12 小時。如需詳細資訊，請參閱《Amazon Elastic Container Registry 使用者指南》中的私有登錄檔身分驗證。

   > **Note**
   >
   > 使用最新版本的 第 2 AWS CLI 版支援 get-login-password命令。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。

   ```
   $ aws ecr-public get-login-password --region us-east-1 | docker login --username
    AWS --password-stdin public.ecr.aws
   ```

   輸出如下。

   ```
   Login succeeded
   ```

2. 使用 docker pull 命令提取 Amazon Linux 容器映像。若要在 Amazon ECR 公共映像庫上查看 Amazon Linux 容器映像，請參閱 Amazon ECR 公共映像庫 - amazonlinux。

   > **Note**
   >
   > 當您提取 AL2023 Docker 容器映像時，您可以使用下列其中一種格式的標籤：

- 若要取得 AL2023 容器映像的最新版本，請使用 `:2023` 標籤。
- 若要取得 AL2023 的特定版本，您可以使用下列格式：
  - `:2023.[0-7 release quarter].[release date].[build number]`

以下範例使用標籤 `:2023` 並提取 AL2023 最新可用的容器映像。

```
$ docker pull public.ecr.aws/amazonlinux/amazonlinux:2023
```

3. (選用) 在本機執行容器。

```
$ docker run -it --security-opt seccomp=unconfined public.ecr.aws/amazonlinux/
amazonlinux:2023 /bin/bash
```

從 Docker Hub 提取 AL2023 容器映像

1. 使用 docker pull 命令提取 AL2023 容器映像。

```
$ docker pull amazonlinux:2023
```

2. (選用) 在本機執行容器。

```
$ docker run -it amazonlinux:2023 /bin/bash
```

> **Note**
>
> AL2023 的容器映像僅使用 dnf 套件管理員來安裝軟體套件。這表示沒有 amazon-linux-extras 或同等的命令可用於其他軟體。

AL2023 最小容器映像

> **Note**
>
> 標準 AL2023 容器映像適用於大多數使用案例，而適應最小容器映像可能比適應 AL2023 基礎容器映像更有效。

在 AL2023.2 中引入的 AL2023 最小容器映像與基本容器映像不同，因為它只包含安裝其他套件所需的裸機最小套件。最小容器映像的設計是一組最小的套件，而不是一組方便的套件。

AL2023 最小容器映像是由 AL2023 中已有的軟體元件建置而成。最小容器映像的主要差異是使用 `microdnf`來提供`dnf`套件管理員，而不是全功能Python型 `dnf`。這可讓最小容器映像更小，因為沒有包含在 AL2023 AMIs 和基礎容器映像中的完整`dnf`套件管理員功能集。

AL2023 最小容器映像會形成 `provided.al2023` AWS Lambda 執行時間環境的基礎。

如需最小容器映像中包含之套件的詳細清單，請參閱 比較 Amazon Linux 2023 容器映像安裝的套件。

最小容器映像的大小

由於 AL2023 最小容器映像包含的套件少於 AL2023 基礎容器映像，因此也明顯較小。下表比較 Amazon Linux 目前和過去版本的容器映像選項。

> **Note**
>
> 映像大小如 Amazon ECR 公共映像庫的 Amazon Linux 所示。

影像	版本	映像大小	注意
Amazon Linux 1 (AL1)	2018.03.0.20230918.0	62.3MB	僅限 `x86-64`
Amazon Linux 2	2.0.20230926.0	64.2MB	`aarch64` 比 `x86-64` 大 1.6 MB
Amazon Linux 2023 基本容器映像	2023.2.20231002.0	52.4MB	

影像	版本	映像大小	注意
Amazon Linux 2023 最小容器映像	2023.2.20231002.0-minimal	35.2MB	

使用 AL2023 最小容器映像

AL2023 最小容器映像可在 上使用，ECR`2023-minimal`標籤一律指向最新的 AL2023 型最小容器映像，而`minimal`標籤可能會更新為較 AL2023 更新的 Amazon Linux 版本。

您可以使用 `docker`搭配下列範例來提取這些標籤：

```
$ docker pull public.ecr.aws/amazonlinux/amazonlinux:minimal
```

```
$ docker pull public.ecr.aws/amazonlinux/amazonlinux:2023-minimal
```

下列範例顯示`Dockerfile`採用最小容器映像並在其上安裝 GCC 的 ：

```
FROM public.ecr.aws/amazonlinux/amazonlinux:2023-minimal
RUN dnf install -y gcc && dnf clean all
```

建置裸機 AL2023 容器映像

AL2023 容器映像是從包含在 AL2023 AMI 中的相同軟體元件所建置。它包含的軟體可讓基礎容器層的行為類似於在 Amazon EC2 執行個體上執行的行為，例如套件管理員 `dnf`。本節說明如何從頭開始建構容器，只包含應用程式所需的裸機最低相依性。

> **Note**
>
> 標準 AL2023 容器映像適用於大多數使用案例。使用標準容器映像可讓您輕鬆地以映像為基礎進行建置。裸機容器映像可讓您更難以在映像上建置。

為應用程序建立具有最小相依性的容器

1. 決定執行期相依性。這將根據您的應用程式而有所不同。

2. 構建一個可建置 FROM scratch 的 Dockerfile / Containerfile。下列的 Dockerfile 範例可用來建置只包含 bash shell 及其相依性的容器。

```
FROM public.ecr.aws/amazonlinux/amazonlinux:2023 as build
RUN mkdir /sysroot
RUN dnf --releasever=$(rpm -q system-release --qf '%{VERSION}') \
  --installroot /sysroot \
  -y \
  --setopt=install_weak_deps=False \
  install bash

FROM scratch
COPY --from=build /sysroot /
WORKDIR /
ENTRYPOINT ["/bin/bash"]
```

- 此 Dockerfile 的運作方式：

  1. 啟動名為 build 的 AL2023 容器。此容器將用於引導準系統容器，此容器本身不會部署，而是生成要部署的容器。
  2. 建立 /sysroot 目錄。此目錄將會是 build 容器安裝準系統容器所需相依性的目錄。在後續步驟中，/sysroot 路徑會封裝成準系統映像的根目錄。

     使用 --installroot 選項並以此方式 dnf，是建立其他 AL2023 映像的方式。這是 dnf 用來執行安裝程式和映像建立工具的功能。
  3. 調用 dnf 以將套件安裝到 /sysroot。

     rpm -q system-release --qf '%{VERSION}' 命令查詢 (-q) system-release 套件、設定查詢格式 (--qf) 以印出受查詢的套件版本 (%{VERSION} 變數是 RPM 版本的 rpm 變數)。

     將 dnf 的 --releasever 引數設為 build 容器中的 system-release 版本，每次發行 Amazon Linux 的更新版容器基本映像時，Dockerfile 就能用來重建準系統容器。

     您可以將 設定為--releasever任何 Amazon Linux 2023 版本，例如 2023.10.20260216。這樣做意味著build容器將作為最新的 AL2023 版本執行，但無論目前的 AL2023 版本為何，從 2023.10.20260216 建置準系統容器。

`--setopt=install_weak_deps=False` 組態選項會告訴 dnf 只安裝必要的相依性，而非推薦或建議的相依性。

4. 將已安裝的系統複製到空白 (`FROM scratch`) 容器的根目錄中。
5. 在此情況 `/bin/bash` 下，將 ENTRYPOINT 社為所需的二進位。

3. 建立目錄，並將步驟 2 範例的內容新增至名為 `Dockerfile` 的檔案。

```
$ mkdir al2023-barebones-bash-example
 $ cd al2023-barebones-bash-example
 $ cat > Dockerfile <<EOF
FROM public.ecr.aws/amazonlinux/amazonlinux:2023 as build
RUN mkdir /sysroot
RUN dnf --releasever=$(rpm -q system-release --qf '%{VERSION}') \
  --installroot /sysroot \
  -y \
  --setopt=install_weak_deps=False \
  install bash && dnf --installroot /sysroot clean all

FROM scratch
COPY --from=build /sysroot /
WORKDIR /
ENTRYPOINT ["/bin/bash"]
EOF
```

4. 執行下列命令來建置容器。

```
$ docker build -t al2023-barebones-bash-example
```

5. 使用下列命令執行容器，以查看僅 `bash` 容器的最小程度。

```
$ docker run -it --rm al2023-barebones-bash-example
bash-5.2# rpm
bash: rpm: command not found
bash-5.2# du -sh /usr/
bash: du: command not found
bash-5.2# ls
bash: ls: command not found
bash-5.2# echo /bin/*
/bin/alias /bin/bash /bin/bashbug /bin/bashbug-64 /bin/bg /bin/catchsegv /bin/cd /
bin/command /bin/fc /bin/fg /bin/gencat /bin/getconf /bin/getent /bin/getopts /
bin/hash /bin/iconv /bin/jobs /bin/ld.so /bin/ldd /bin/locale /bin/localedef /
```

```
bin/pldd /bin/read /bin/sh /bin/sotruss /bin/sprof /bin/type /bin/tzselect /bin/
ulimit /bin/umask /bin/unalias /bin/wait /bin/zdump
```

如需更實際的範例，下列程序會為顯示 Hello World! 的 C 應用程式建置容器。

1. 建立空白目錄，然後新增 C 原始碼和 Dockerfile。

```
$ mkdir al2023-barebones-c-hello-world-example
$ cd al2023-barebones-c-hello-world-example
$ cat > hello-world.c <<EOF
#include <stdio.h>
int main(void)
{
  printf("Hello World!\n");
  return 0;
}
EOF

$ cat > Dockerfile <<EOF
FROM public.ecr.aws/amazonlinux/amazonlinux:2023 as build
COPY hello-world.c /
RUN dnf -y install gcc
RUN gcc -o hello-world hello-world.c
RUN mkdir /sysroot
RUN mv hello-world /sysroot/
RUN dnf --releasever=$(rpm -q system-release --qf '%{VERSION}') \
  --installroot /sysroot \
  -y \
  --setopt=install_weak_deps=False \
  install glibc && dnf --installroot /sysroot clean all

FROM scratch
COPY --from=build /sysroot /
WORKDIR /
ENTRYPOINT ["/hello-world"]
EOF
```

2. 使用下列命令來建置容器。

```
$ docker build -t al2023-barebones-c-hello-world-example .
```

3. 使用下列命令來執行容器。

```
$ docker run -it --rm al2023-barebones-c-hello-world-example
Hello World!
```

比較 Amazon Linux 2023 容器映像安裝的套件

AL2023 基礎容器映像上出現RPMs 與 AL2023 最小容器映像上出現RPMs 的比較。

套件	容器	最小容器
alternatives	1.15	1.15
amazon-linux-repo-cdn	2023.6.20241031	2023.6.20241031
audit-libs	3.0.6	3.0.6
basesystem	11	11
bash	5.2.15	5.2.15
bzip2-libs	1.0.8	1.0.8
ca-certificates	2023.2.68	2023.2.68
coreutils-single	8.32	8.32
crypto-policies	20220428	20220428
curl-minimal	8.5.0	8.5.0
dnf	4.14.0	
dnf-data	4.14.0	4.14.0
elfutils-default-yama-scope	0.188	

套件	容器	最小容器
elfutils-libelf	0.188	
elfutils-libs	0.188	
expat	2.5.0	
file-libs	5.39	5.39
filesystem	3.14	3.14
gawk	5.1.0	5.1.0
gdbm-libs	1.19	
glib2	2.74.7	2.74.7
glibc	2.34	2.34
glibc-common	2.34	2.34
glibc-minimal-langpack	2.34	2.34
gmp	6.2.1	6.2.1
gnupg2-minimal	2.3.7	2.3.7
gobject-introspection		1.73.0
gpgme	1.15.1	1.15.1
grep	3.8	3.8
json-c	0.14	0.14
keyutils-libs	1.6.3	1.6.3
krb5-libs	1.21.3	1.21.3

套件	容器	最小容器
libacl	2.3.1	2.3.1
libarchive	3.7.4	3.7.4
libassuan	2.5.5	2.5.5
libattr	2.5.1	2.5.1
libblkid	2.37.4	2.37.4
libcap	2.48	2.48
libcap-ng	0.8.2	0.8.2
libcom_err	1.46.5	1.46.5
libcomps	0.1.20	
libcurl-minimal	8.5.0	8.5.0
libdnf	0.69.0	0.69.0
libffi	3.4.4	3.4.4
libgcc	11.4.1	11.4.1
libgcrypt	1.10.2	1.10.2
libgomp	11.4.1	
libgpg-error	1.42	1.42
libidn2	2.3.2	2.3.2
libmodulemd	2.13.0	2.13.0
libmount	2.37.4	2.37.4
libnghttp2	1.59.0	1.59.0

套件	容器	最小容器
libpeas		1.32.0
libpsl	0.21.1	0.21.1
librepo	1.14.5	1.14.5
libreport-filesystem	2.15.2	2.15.2
libselinux	3.4	3.4
libsepol	3.4	3.4
libsigsegv	2.13	2.13
libsmartcols	2.37.4	2.37.4
libsolv	0.7.22	0.7.22
libstdc++	11.4.1	11.4.1
libtasn1	4.19.0	4.19.0
libunistring	0.9.10	0.9.10
libuuid	2.37.4	2.37.4
libverto	0.3.2	0.3.2
libxcrypt	4.4.33	
libxml2	2.10.4	2.10.4
libyaml	0.2.5	0.2.5
libzstd	1.5.5	1.5.5
lua-libs	5.4.4	5.4.4
lz4-libs	1.9.4	1.9.4

套件	容器	最小容器
microdnf		3.10.0
microdnf-dnf		3.10.0
mpfr	4.1.0	4.1.0
ncurses-base	6.2	6.2
ncurses-libs	6.2	6.2
npth	1.6	1.6
openssl-libs	3.0.8	3.0.8
p11-kit	0.24.1	0.24.1
p11-kit-trust	0.24.1	0.24.1
pcre2	10.40	10.40
pcre2-syntax	10.40	10.40
popt	1.18	1.18
publicsuffix-list-dafsa	20240212	20240212
python3	3.9.16	
python3-dnf	4.14.0	
python3-gpg	1.15.1	
python3-hawkey	0.69.0	
python3-libcomps	0.1.20	
python3-libdnf	0.69.0	
python3-libs	3.9.16	

套件	容器	最小容器
python3-pip-wheel	21.3.1	
python3-rpm	4.16.1.3	
python3-setuptools-wheel	59.6.0	
readline	8.1	8.1
rpm	4.16.1.3	4.16.1.3
rpm-build-libs	4.16.1.3	
rpm-libs	4.16.1.3	4.16.1.3
rpm-sign-libs	4.16.1.3	
sed	4.8	4.8
setup	2.13.7	2.13.7
sqlite-libs	3.40.0	3.40.0
system-release	2023.6.20241031	2023.6.20241031
tzdata	2024a	
xz-libs	5.2.5	5.2.5
yum	4.14.0	
zlib	1.2.11	1.2.11

比較 Amazon Linux 2023 最小 AMI 與容器映像安裝的套件

AL2023 最小 AMI 上出現RPMs 與 AL2023 基礎和最小容器映像上出現RPMs 的比較。

套件	最小 AMI	容器	最小容器
alternatives	1.15	1.15	1.15
amazon-chrony-config	4.3		
amazon-ec2-net-utils	2.5.1		
amazon-linux-repo-cdn		2023.6.20241031	2023.6.20241031
amazon-linux-repo-s3	2023.6.20241031		
amazon-linux-sb-keys	2023.1		
amd-ucode-firmware	20210208 (noarch)		
audit	3.0.6		
audit-libs	3.0.6	3.0.6	3.0.6
awscli-2	2.15.30		
basesystem	11	11	11
bash	5.2.15	5.2.15	5.2.15
bzip2-libs	1.0.8	1.0.8	1.0.8
ca-certificates	2023.2.68	2023.2.68	2023.2.68
checkpolicy	3.4		
chrony	4.3		
cloud-init	22.2.2		

套件	最小 AMI	容器	最小容器
cloud-init-cfg-ec2	22.2.2		
cloud-utils-growpart	0.31		
coreutils	8.32		
coreutils-common	8.32		
coreutils-single		8.32	8.32
cpio	2.13		
cracklib	2.9.6		
cracklib-dicts	2.9.6		
crypto-policies	20220428	20220428	20220428
cryptsetup-libs	2.6.1		
curl-minimal	8.5.0	8.5.0	8.5.0
cyrus-sasl-lib	2.1.27		
dbus	1.12.28		
dbus-broker	32		
dbus-common	1.12.28		
dbus-libs	1.12.28		
device-mapper	1.02.185		

套件	最小 AMI	容器	最小容器
device-mapper-libs	1.02.185		
diffutils	3.8		
dnf	4.14.0	4.14.0	
dnf-data	4.14.0	4.14.0	4.14.0
dnf-plugin-release-notification	1.2		
dnf-plugins-core	4.3.0		
dnf-plugin-support-info	1.2		
dracut	055		
dracut-config-ec2	3.0		
dracut-config-generic	055		
e2fsprogs	1.46.5		
e2fsprogs-libs	1.46.5		
ec2-utils	2.2.0		
efi-filesystem	5		
efivar	38		
efivar-libs	38		

套件	最小 AMI	容器	最小容器
elfutils-default-yama-scope	0.188	0.188	
elfutils-libelf	0.188	0.188	
elfutils-libs	0.188	0.188	
expat	2.5.0	2.5.0	
file	5.39		
file-libs	5.39	5.39	5.39
filesystem	3.14	3.14	3.14
findutils	4.8.0		
fuse-libs	2.9.9		
gawk	5.1.0	5.1.0	5.1.0
gdbm-libs	1.19	1.19	
gdisk	1.0.8		
gettext	0.21		
gettext-libs	0.21		
glib2	2.74.7	2.74.7	2.74.7
glibc	2.34	2.34	2.34
glibc-all-langpacks	2.34		
glibc-common	2.34	2.34	2.34

套件	最小 AMI	容器	最小容器
glibc-locale-source	2.34		
glibc-minimal-langpack		2.34	2.34
gmp	6.2.1	6.2.1	6.2.1
gnupg2-minimal	2.3.7	2.3.7	2.3.7
gnutls	3.8.0		
gobject-introspection			1.73.0
gpgme	1.15.1	1.15.1	1.15.1
grep	3.8	3.8	3.8
groff-base	1.22.4		
grub2-common	2.06		
grub2-efi-aa64-ec2	2.06 (aarch64)		
grub2-efi-x64-ec2	2.06 (x86_64)		
grub2-pc-modules	2.06		
grub2-tools	2.06		
grub2-tools-minimal	2.06		
grubby	8.40		

套件	最小 AMI	容器	最小容器
gzip	1.12		
hostname	3.23		
hwdata	0.384		
inih	49		
initscripts	10.09		
iproute	6.10.0		
iputils	20210202		
irqbalance	1.9.0		
jansson	2.14		
jitterentropy	3.4.1		
jq	1.7.1		
json-c	0.14	0.14	0.14
kbd	2.4.0		
kbd-misc	2.4.0		
kernel	6.1.112		
kernel-libbpf	6.1.112		
kernel-livepatch-repo-s3	2023.6.20241031		
keyutils-libs	1.6.3	1.6.3	1.6.3
kmod	29		

套件	最小 AMI	容器	最小容器
kmod-libs	29		
krb5-libs	1.21.3	1.21.3	1.21.3
less	608		
libacl	2.3.1	2.3.1	2.3.1
libarchive	3.7.4	3.7.4	3.7.4
libargon2	20171227		
libassuan	2.5.5	2.5.5	2.5.5
libattr	2.5.1	2.5.1	2.5.1
libblkid	2.37.4	2.37.4	2.37.4
libcap	2.48	2.48	2.48
libcap-ng	0.8.2	0.8.2	0.8.2
libcbor	0.7.0		
libcom_err	1.46.5	1.46.5	1.46.5
libcomps	0.1.20	0.1.20	
libcurl-minimal	8.5.0	8.5.0	8.5.0
libdb	5.3.28		
libdnf	0.69.0	0.69.0	0.69.0
libeconf	0.4.0		
libedit	3.1		
libfdisk	2.37.4		

套件	最小 AMI	容器	最小容器
libffi	3.4.4	3.4.4	3.4.4
libfido2	1.10.0		
libgcc	11.4.1	11.4.1	11.4.1
libgcrypt	1.10.2	1.10.2	1.10.2
libgomp	11.4.1	11.4.1	
libgpg-error	1.42	1.42	1.42
libidn2	2.3.2	2.3.2	2.3.2
libkcapi	1.4.0		
libkcapi-hmaccalc	1.4.0		
libmnl	1.0.4		
libmodulemd	2.13.0	2.13.0	2.13.0
libmount	2.37.4	2.37.4	2.37.4
libnghttp2	1.59.0	1.59.0	1.59.0
libpeas			1.32.0
libpipeline	1.5.3		
libpsl	0.21.1	0.21.1	0.21.1
libpwquality	1.4.4		
librepo	1.14.5	1.14.5	1.14.5
libreport-filesystem	2.15.2	2.15.2	2.15.2

套件	最小 AMI	容器	最小容器
libseccomp	2.5.3		
libselinux	3.4	3.4	3.4
libselinux-utils	3.4		
libsemanage	3.4		
libsepol	3.4	3.4	3.4
libsigsegv	2.13	2.13	2.13
libsmartcols	2.37.4	2.37.4	2.37.4
libsolv	0.7.22	0.7.22	0.7.22
libss	1.46.5		
libstdc++	11.4.1	11.4.1	11.4.1
libtasn1	4.19.0	4.19.0	4.19.0
libtextstyle	0.21		
libunistring	0.9.10	0.9.10	0.9.10
libuser	0.63		
libutempter	1.2.1		
libuuid	2.37.4	2.37.4	2.37.4
libverto	0.3.2	0.3.2	0.3.2
libxcrypt	4.4.33	4.4.33	
libxml2	2.10.4	2.10.4	2.10.4
libyaml	0.2.5	0.2.5	0.2.5

套件	最小 AMI	容器	最小容器
libzstd	1.5.5	1.5.5	1.5.5
linux-firmware-whence	20210208 (noarch)		
logrotate	3.20.1		
lua-libs	5.4.4	5.4.4	5.4.4
lz4-libs	1.9.4	1.9.4	1.9.4
man-db	2.9.3		
microcode_ctl	2.1 (x86_64)		
microdnf			3.10.0
microdnf-dnf			3.10.0
mpfr	4.1.0	4.1.0	4.1.0
ncurses	6.2		
ncurses-base	6.2	6.2	6.2
ncurses-libs	6.2	6.2	6.2
nettle	3.8		
net-tools	2.0		
npth	1.6	1.6	1.6
numactl-libs	2.0.14		
oniguruma	6.9.7.1		
openldap	2.4.57		
openssh	8.7p1		

套件	最小 AMI	容器	最小容器
openssh-clients	8.7p1		
openssh-server	8.7p1		
openssl	3.0.8		
openssl-libs	3.0.8	3.0.8	3.0.8
openssl-pkcs11	0.4.12		
os-prober	1.77		
p11-kit	0.24.1	0.24.1	0.24.1
p11-kit-trust	0.24.1	0.24.1	0.24.1
pam	1.5.1		
passwd	0.80		
pciutils	3.7.0		
pciutils-libs	3.7.0		
pcre2	10.40	10.40	10.40
pcre2-syntax	10.40	10.40	10.40
policycoreutils	3.4		
popt	1.18	1.18	1.18
procps-ng	3.3.17		
psmisc	23.4		
publicsuffix-list-dafsa	20240212	20240212	20240212
python3	3.9.16	3.9.16	

套件	最小 AMI	容器	最小容器
python3-attrs	20.3.0		
python3-audit	3.0.6		
python3-awscrt	0.19.19		
python3-babel	2.9.1		
python3-cffi	1.14.5		
python3-chardet	4.0.0		
python3-colorama	0.4.4		
python3-configobj	5.0.6		
python3-cryptography	36.0.1		
python3-dateutil	2.8.1		
python3-dbus	1.2.18		
python3-distro	1.5.0		
python3-dnf	4.14.0	4.14.0	
python3-dnf-plugins-core	4.3.0		
python3-docutils	0.16		
python3-gpg	1.15.1	1.15.1	
python3-hawkey	0.69.0	0.69.0	

套件	最小 AMI	容器	最小容器
python3-idna	2.10		
python3-jinja2	2.11.3		
python3-jmespath	0.10.0		
python3-jsonpatch	1.21		
python3-jsonpointer	2.0		
python3-jsonschema	3.2.0		
python3-libcomps	0.1.20	0.1.20	
python3-libdnf	0.69.0	0.69.0	
python3-libs	3.9.16	3.9.16	
python3-libselinux	3.4		
python3-libsemanage	3.4		
python3-markupsafe	1.1.1		
python3-netifaces	0.10.6		
python3-oauthlib	3.0.2		

套件	最小 AMI	容器	最小容器
python3-pip-wheel	21.3.1	21.3.1	
python3-ply	3.11		
python3-policycoreutils	3.4		
python3-prettytable	0.7.2		
python3-prompt-toolkit	3.0.24		
python3-pycparser	2.20		
python3-pyrsistent	0.17.3		
python3-pyserial	3.4		
python3-pysocks	1.7.1		
python3-pytz	2022.7.1		
python3-pyyaml	5.4.1		
python3-requests	2.25.1		
python3-rpm	4.16.1.3	4.16.1.3	
python3-ruamel-yaml	0.16.6		

套件	最小 AMI	容器	最小容器
python3-ruamel-yaml-clib	0.1.2		
python3-setools	4.4.1		
python3-setuptools	59.6.0		
python3-setuptools-wheel	59.6.0	59.6.0	
python3-six	1.15.0		
python3-systemd	235		
python3-urllib3	1.25.10		
python3-wcwidth	0.2.5		
readline	8.1	8.1	8.1
rng-tools	6.14		
rootfiles	8.1		
rpm	4.16.1.3	4.16.1.3	4.16.1.3
rpm-build-libs	4.16.1.3	4.16.1.3	
rpm-libs	4.16.1.3	4.16.1.3	4.16.1.3
rpm-plugin-selinux	4.16.1.3		
rpm-plugin-systemd-inhibit	4.16.1.3		
rpm-sign-libs	4.16.1.3	4.16.1.3	

套件	最小 AMI	容器	最小容器
sbsigntools	0.9.4		
sed	4.8	4.8	4.8
selinux-policy	38.1.45		
selinux-policy-targeted	38.1.45		
setup	2.13.7	2.13.7	2.13.7
shadow-utils	4.9		
sqlite-libs	3.40.0	3.40.0	3.40.0
sudo	1.9.15		
sysctl-defaults	1.0		
systemd	252.23		
systemd-libs	252.23		
systemd-networkd	252.23		
systemd-pam	252.23		
systemd-resolved	252.23		
systemd-udev	252.23		
system-release	2023.6.20241031	2023.6.20241031	2023.6.20241031
tar	1.34		
tzdata	2024a	2024a	

套件	最小 AMI	容器	最小容器
update-motd	2.2		
userspace-rcu	0.12.1		
util-linux	2.37.4		
util-linux-core	2.37.4		
vim-data	9.0.2153		
vim-minimal	9.0.2153		
which	2.21		
xfsprogs	5.18.0		
xz	5.2.5		
xz-libs	5.2.5	5.2.5	5.2.5
yum	4.14.0	4.14.0	
zlib	1.2.11	1.2.11	1.2.11
zram-generator	1.1.2		
zram-generator-defaults	1.1.2		
zstd	1.5.5		

上的 AL2023 AWS Elastic Beanstalk

AWS Elastic Beanstalk 是一種用於部署和擴展 Web 應用程式和服務的服務。您只要上傳程式碼，Elastic Beanstalk 即會自動處理部署作業；無論是容量佈建、負載平衡、自動擴展或應用程式運作狀態監控，該服務都能滿足所需。如需詳細資訊，請參閱 AWS Elastic Beanstalk。

若要使用 Elastic Beanstalk，您須建立應用程式、將應用程式版本 (採用諸如 Java .war 檔案之應用程式原始碼套件的形式) 上傳至 Elastic Beanstalk，然後提供應用程式的部分資訊。Elastic Beanstalk 會自動啟動環境，並建立和設定執行程式碼所需的 AWS 資源。如需詳細資訊，請參閱《AWS Elastic Beanstalk 開發人員指南》。

Elastic Beanstalk Linux 平台使用 Amazon EC2 執行個體，而這些執行個體執行 Amazon Linux。自 2023 年 8 月 4 日起，Elastic Beanstalk 提供以下基於 Amazon Linux 2023 的平台分支：Docker、Tomcat、Java SE、Node.js、PHP 和 Python。Elastic Beanstalk 正在努力在更多 Elastic Beanstalk 平台上發佈對 AL2023 的支援。

您可以在《Elastic Beanstalk 開發人員指南》的「Elastic Beanstalk Linux 平台」一節中查看 Elastic Beanstalk 平台支援和目前建置在 AL2023 上的完整平台清單。

您可以在 Elastic Beanstalk 版本備註中查看新 Elastic Beanstalk 平台的版本備註以及現有平台的版本

在 中使用 AL2023 AWS CloudShell

AWS CloudShell 是以瀏覽器為基礎的預先驗證 Shell，您可以直接從 啟動 AWS 管理主控台。您可以從 AWS 管理主控台 幾種不同的方式導覽至 CloudShell。如需詳細資訊，請參閱如何開始使用 AWS CloudShell？

AWS CloudShell目前以 Amazon Linux 2 為基礎的 將遷移至 AL2023。從 2023 年 12 月 4 日開始，遷移至 AL2023 的所有 都將開始推出。 AWS 區域 如需有關 CloudShell 遷移到 AL2023 的詳細資訊，請參閱AWS CloudShell 從 Amazon Linux 2 遷移到 Amazon Linux 2023。

使用以 AL2023 為基礎的 Amazon ECS AMIs 託管容器化工作負載

> **Note**
>
> 如需如何在容器內使用 AL2023 的詳細資訊，請參閱 容器中的 AL2023。

Amazon Elastic Container Service (Amazon ECS) 為全受管容器協同運作服務，可讓您輕鬆部署、管理和擴展容器化應用程式。Amazon ECS 是全受管服務，內建 AWS 組態和操作最佳實務。它與 AWS 和第三方工具整合，例如 Amazon Elastic Container Registry (Amazon ECR) 和 Docker。這種整合可讓團隊能夠更輕鬆地專注於建置應用程式，而無需為環境分心。您可以在雲端跨 AWS 區域執行和擴展容器工作負載，而無需進行控制平面管理等複雜任務。

您可以使用以 AL2023 為基礎的 Amazon ECS 最佳化 AMI，在 AL2023 上託管容器化工作負載。如需詳細資訊，請參閱 Amazon ECS 最佳化 AMI

與 AL2 相比，Amazon ECS 的 AL2023 變更 AL2

如同 AL2, AL2023提供以 Amazon ECS Linux 執行個體身分執行 所需的基本套件。在 AL2 中，`containerd`、 `docker`和 `ecs-init`套件可透過 取得`amazon-linux-extras`，而 AL2023 會在核心儲存庫中包含這些套件。

透過版本控制的儲存庫功能進行確定性升級時，每個 AL2023 AMI 預設都會鎖定到特定的儲存庫版本。AL2023 Amazon ECS 最佳化 AMI 也是如此。在部署之前，您可以仔細管理和測試環境的所有更新，並提供在發生問題時還原至先前 AMI 內容的簡單方法。如需此 AL2023 功能的詳細資訊，請參閱透過 AL2023 上的版本控制儲存庫進行確定性升級。

AL2023 透過 AL2 AL2。如需詳細資訊，請參閱整合控制群組階層 (cgroup v2)。

> **Note**
>
> 2AL2023.2.20230920 之前的 AL2023 版本（第一個 AL2023.2 版本）包含 cgroup Out-of-Memory `systemd` (OOM) 處理的 錯誤。cgroup 中的所有程序一律遭到終止，而不是 OOM-Killer 一次選擇一個程序，這是預期的行為。
> 與 AL2 行為相比，這是迴歸，截至 AL2023 的 2023.2AL202320230920 版本為止是固定的。

建置 Amazon ECS 最佳化 AMI 的程式碼可在 amazon-ecs-ami GitHub 專案上取得。版本備註說明哪些 AL2023 版本對應到哪些 Amazon ECS AMI 版本。

自訂基於 AL2023 的 Amazon ECS 最佳化 AMI

> **Important**
>
> 我們建議您使用 Amazon ECS 最佳化 AL2023 AMI。如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的 Amazon ECS 最佳化 AMI。

您可以使用與 Amazon ECS 用來建立自訂 AMI 相同的建置指令碼。如需詳細資訊，請參閱 Amazon ECS 最佳化 Linux AMI 建置指令碼。

在 AL2023 上使用 Amazon Elastic File System

Amazon Elastic File System (Amazon EFS) 提供無伺服器、完全彈性的檔案儲存功能，讓您無需佈建或管理儲存容量和效能，即可分享檔案資料。Amazon EFS 可隨需擴展至 PB 級，而不會中斷應用程式，並可隨著您新增和移除檔案而自動擴展及縮減。因爲 Amazon EFS 採用簡單的 Web 服務介面，您可以快速輕鬆地建立和設定檔案系統。此服務會為您管理所有檔案儲存基礎設施，這表示您可以避免部署、修補和維護複雜檔案系統組態的複雜性。

Amazon EFS 支援網路檔案系統第 4 版 (NFSv4.1 和 NFSv4.0) 協定，因此您目前使用的應用程式和工具都可以與 Amazon EFS 無縫配合使用。包括 Amazon EC2、Amazon ECS 和 在內的多個運算執行個體 AWS Lambda可以同時存取 Amazon EFS 檔案系統。因此，EFS 檔案系統可針對在一個以上運算執行個體或伺服器執行的工作負載和應用程式，提供共同的資料來源。

在 AL2023 上安裝 **amazon-efs-utils**

`amazon-efs-utils` 套件可在要安裝和用於存取 Amazon EFS 檔案系統的 AL2023 儲存庫中使用。

在 AL2023 上安裝 **amazon-efs-utils** 套件

- `amazon-efs-utils` 使用下列命令安裝 。

```
$ dnf -y install amazon-efs-utils
```

在 AL2023 上掛載 Amazon EFS 檔案系統

安裝 `amazon-efs-utils` 之後，您可以在 AL2023 執行個體上掛載 Amazon EFS 檔案系統。

在 AL2023 上掛載 Amazon EFS 檔案系統

- 若要使用檔案系統 ID 進行掛載，請使用下列命令。

```
sudo mount -t efs file-system-id efs-mount-point/
```

您也可以掛載檔案系統，以便使用 TLS 加密傳輸中的資料，或使用 DNS 名稱或掛載目標 IP，而不是檔案系統 ID。如需詳細資訊，請參閱使用 EFS 掛載協助程式在 Amazon Linux 執行個體上掛載。

使用在 AL2023 上建置的 Amazon EMR

Amazon EMR 是一種 Web 服務，可讓您輕鬆地使用 Apache Hadoop 和 AWS提供的服務，以高效率地處理大量資料。

AL2023 型 Amazon EMR 版本

Amazon EMR 7.0.0 版是建置於 AL2023 的第一個版本。在此版本中，AL2023 是 Amazon EMR 的基礎作業系統，將 AL2023 的所有優勢帶入 Amazon EMR。如需詳細資訊，請參閱 Amazon EMR 7.0.0 版本備註。

EKS 上基於 Amazon EMR 的 AL2023

EKS 6.13 上的 Amazon EMR 是導入 AL2023 作為選項的第一個版本。在此版本中，您可以將 Spark 與 AL2023 同時啟動作為作業系統，並搭配 Java 17 執行期。如需詳細資訊，請參閱 Amazon EMR on EKS 6.13 版本備註和所有 Amazon EMR on EKS 版本備註。

在 中使用 AL2023 AWS Lambda

您可以使用 執行程式碼 AWS Lambda，而無需佈建或管理伺服器。您只要按實際使用的運算時間付費即可，未執行程式碼時不必支付任何費用。可以針對幾乎任何類型的應用程式或後端服務執行程式碼，且通常都不需要管理。您只需上傳程式碼，Lambda 就會處理執行程式碼及擴展所需的各項工作，藉此維持高可用性。

AL2023 **`provided.al2023`**受管執行時間和容器映像

`provided.al2023` 基本執行期是以 AL2023 最小容器映像為基礎，並提供以 AL2023 為基礎的 Lambda 受管執行期和容器基礎映像。由於`provided.al2023`執行時間是以 AL2023 最小容器映像為基礎，因此遠小於 40 MB，遠小於 109 MB 的`provided.al2`執行時間。

如需詳細資訊，請參閱 Lambda 執行期和使用 Lambda 容器映像。

AL2023 型 Lambda 執行時間

受管語言執行時間的未來版本，例如 Node.js 20、Python3.12、21 Java 和 .NET 8，是以 AL2023 為基礎，並將使用 `provided.al2023` 做為基礎映像，如 AL2023 型執行時間的公告中所述。

AL2023 型 Lambda 函數

- AL2023 Lambda 函數寫入 Go

- AL2023 Lambda 函數寫入 Rust

如需詳細資訊，請參閱《 AWS Lambda 開發人員指南》中的 Lambda 執行時間。

教學課程

下列教學課程說明如何使用執行 Amazon Linux 2023 (AL2023) 的 Amazon EC2 執行個體來執行常見任務。AL2023 如需影片教學課程，請參閱AWS 教學影片和實驗室。

如需 AL2 說明，請參閱《Amazon EC2 使用者指南》中的執行 Linux 的 Amazon EC2 執行個體教學課程。 Amazon EC2

教學課程

- 教學課程：在 AL2023 上安裝 LAMP Web 伺服器
- 教學課程：在 AL2023 上設定 SSL/TLS
- 教學課程：託管 AL2023 上的 WordPress 部落格
- 教學課程：在 AL2023 上將 Redis 6 轉換為 Valkey
- 教學課程：在 AL2023 上安裝 GNOME 桌面環境
- 教學課程：在 AL2023 上設定 TigerVNC 伺服器
- 在 AL2023 核心上使用多世代 LRU (MGLRU)

教學課程：在 AL2023 上安裝 LAMP Web 伺服器

下列程序可協助您在 AL2023 執行個體（有時稱為 LAMP Web 伺服器或 LAMP 堆疊）上安裝支援 PHP 和 MariaDB（社群開發的 MySQL 分支）的 Apache Web 伺服器。您可以使用此伺服器託管一個靜態網站或部署一個將資訊讀取和寫入資料庫的動態 PHP 應用程式。

> **Important**
>
> 這些程序旨在與 AL2023 搭配使用。如果您要在不同的發行版 (例如 Ubuntu 或 Red Hat Enterprise Linux) 上設定 LAMP Web 伺服器，本教學不適用於您。若為 Ubuntu，請參閱以下 Ubuntu 社群文件：ApacheMySQLPHP。若是其他發行版，請參閱其特定文件。

任務

- 步驟 1：準備 LAMP 伺服器
- 步驟 2：測試您的 LAMP 伺服器
- 步驟 3：保護資料庫伺服器
- 步驟 4：(選用) 安裝 phpMyAdmin

- 疑難排解
- 相關主題

步驟 1：準備 LAMP 伺服器

先決條件

- 本教學課程假設您已使用 AL2023 啟動新的執行個體，其公有 DNS 名稱可從網際網路連線。如需詳細資訊，請參閱Amazon EC2 上的 AL2023。您也必須先設定您的安全群組，允許 SSH (連接埠 22)、HTTP (連接埠 80) 和 HTTPS (連接埠 443) 連線。如需這些先決條件的詳細資訊，請參閱《Amazon EC2 使用者指南》中的授權 Linux 執行個體的傳入流量。
- 下列程序會安裝 AL2023 上可用的最新 PHP 版本，目前為 8.1。若您預計不使用此教學所提及的 PHP 應用程式，改使用其他 PHP 應用程式，您應檢查其與 PHP 8.1 的相容性。

準備 LAMP 伺服器

1. 連線到您的執行個體。如需詳細資訊，請參閱連線至 AL2023 執行個體。
2. 為確保所有軟體套件皆為最新版本，請對您的執行個體執行快速軟體更新。本程序可能需費時幾分鐘，但確定您擁有最新的安全更新和錯誤修正至關重要。

   -y 選項不要求確認就會安裝更新。如果您要先檢查更新再安裝，則可以略過此選項。

   ```
   [ec2-user ~]$ sudo dnf upgrade -y
   ```
3. 安裝適用於 AL2023 的最新版本 Apache Web 伺服器和 PHP 套件。

   ```
   [ec2-user ~]$ sudo dnf install -y httpd wget php-fpm php-mysqli php-json php php-devel
   ```
4. 安裝 MariaDB 軟體套件。使用 dnf install 命令可以同時安裝多個軟體套件和所有相關的依存項目。

   ```
   [ec2-user ~]$ sudo dnf install mariadb105-server
   ```

   您可以使用下列命令來檢視這些套件的目前版本：

   ```
   [ec2-user ~]$ sudo dnf info package_name
   ```

範例：

```
[root@ip-172-31-25-170 ec2-user]# dnf info mariadb105
Last metadata expiration check: 0:00:16 ago on Tue Feb 14 21:35:13 2023.
Installed Packages
Name         : mariadb105
Epoch        : 3
Version      : 10.5.16
Release      : 1.amzn2023.0.6
Architecture : x86_64
Size         : 18 M
Source       : mariadb105-10.5.16-1.amzn2023.0.6.src.rpm
Repository   : @System
From repo    : amazonlinux
Summary      : A very fast and robust SQL database server
URL          : http://mariadb.org
License      : GPLv2 and LGPLv2
Description  : MariaDB is a community developed fork from MySQL - a multi-user,
 multi-threaded
             : SQL database server. It is a client/server implementation consisting
 of
             : a server daemon (mariadbd) and many different client programs and
 libraries.
             : The base package contains the standard MariaDB/MySQL client programs
 and
             : utilities.
```

5. 啟動 Apache Web 伺服器。

```
[ec2-user ~]$ sudo systemctl start httpd
```

6. 使用 systemctl 命令來設定 Apache Web 伺服器在每次系統開機時啟動。

```
[ec2-user ~]$ sudo systemctl enable httpd
```

要確認 httpd 已啟用，您可以執行以下命令：

```
[ec2-user ~]$ sudo systemctl is-enabled httpd
```

7. 如果您尚未這樣做，請新增安全規則以允許對內 HTTP (連接埠 80) 連線到您的執行個體。根據預設，launch-wizard-*N* 安全群組在啟動期間已針對您的執行個體建立完畢。如果您未新增安全群組規則，該群組只會包含允許 SSH 連線的單一規則。

   a. 前往 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。

   b. 在左側的導覽窗格中，選取 Instances (執行個體)，然後選取您的執行個體。

   c. 在 Security (安全性) 標籤上，檢視對內規則。您應該會看到下列規則：

   ```
   Port range   Protocol     Source
   22           tcp          0.0.0.0/0
   ```

   > **Warning**
   >
   > 使用 `0.0.0.0/0` 可讓所有 IPv4 地址使用 SSH 存取您的執行個體。通常在測試環境中短暫進行此作業是沒有問題的，但用在生產環境則不安全。在生產環境中，您應只授權特定 IP 地址或特定範圍的地址存取您的執行個體。

   d. 如果允許 HTTP (連接埠 80) 連線的傳入規則不存在，您必須現在新增規則。選擇安全群組的連結。使用 中的程序，請參閱授權 Linux 執行個體的傳入流量，新增具有下列值的新傳入安全規則：

   - 類型：HTTP
   - Protocol (通訊協定)：TCP
   - Port Range (連接埠範圍)：80
   - Source (來源)：自訂

8. 測試您的 Web 伺服器。在 Web 瀏覽器中，輸入執行個體的公有 DNS 地址 (或公有 IP 地址)。如果 `/var/www/html` 中沒有內容，則您應該會看到 Apache 測試頁，其會顯示 "It works!" (成功了！) 訊息。

   您可以使用 Amazon EC2 主控台取得執行個體的公有 DNS (請查看 Public IPv4 DNS (公有 IPv4 DNS) 欄；如果此欄為隱藏，請選擇 Preferences (偏好設定) (齒輪狀圖示) 並將 Public IPv4 DNS (公有 IPv4 DNS) 選項切換為開啟)。

   確認執行個體的安全性群組包含允許連接埠 80 上的 HTTP 流量的規則。如需詳細資訊，請參閱新增規則至安全群組。

> **Important**
>
> 如果您未使用 Amazon Linux，則可能也需要在您的執行個體上設定防火牆以允許這些連線。如需如何設定防火牆的詳細資訊，請參閱針對您特定散發的文件。

Apache httpd 提供保存在稱為 Apache 文件根目錄中的檔案。Amazon Linux Apache 文件根目錄是 /var/www/html，預設情況下由根擁有。

若要允許 ec2-user 帳戶操作此目錄中的檔案，您必須修改目錄的所有權和許可。有多種方法可以達成這件任務。在本教學中，您會將 ec2-user 新增至 apache 群組以向 apache 群組授予 /var/www 目錄的所有權，並指派寫入許可。

設定檔案許可

1. 將您的使用者 (在此案例中為 ec2-user) 新增至 apache 群組。

```
[ec2-user ~]$ sudo usermod -a -G apache ec2-user
```

2. 登出並重新登入，以取得新的群組並驗證您的成員資格。

   a. 登出 (使用 exit 命令或關閉終端機視窗)：

```
[ec2-user ~]$ exit
```

   b. 若要在 apache 群組中驗證您的會員資格，請重新連線至您的執行個體，然後執行下列命令：

```
[ec2-user ~]$ groups
ec2-user adm wheel apache systemd-journal
```

3. 將 /var/www 的群組所有權及其內容變更為 apache 群組。

```
[ec2-user ~]$ sudo chown -R ec2-user:apache /var/www
```

4. 若要新增群組寫入許可並在將來的子目錄上設定群組 ID，請變更 /var/www 及其子目錄的目錄許可。

```
[ec2-user ~]$ sudo chmod 2775 /var/www && find /var/www -type d -exec sudo chmod
 2775 {} \;
```

5. 若要新增群組寫入許可，請以遞迴方式變更 /var/www 及其子目錄的檔案許可：

```
[ec2-user ~]$ find /var/www -type f -exec sudo chmod 0664 {} \;
```

現在，ec2-user (以及 apache 群組未來的任何成員) 可以新增、刪除和編輯 Apache 文件根目錄中的檔案，所以您可以新增內容 (例如靜態網站或 PHP 應用程式)。

保護您的 Web 伺服器 (選擇性)

執行 HTTP 通訊協定的 Web 伺服器不會為其傳送或接收的資料提供傳輸安全性。當您使用 Web 瀏覽器連線到 HTTP 伺服器時，您前往的 URL、您收到的網頁內容以及您提交的任何 HTML 表單內容 (包括密碼)，網路路徑的任何一處的竊聽者都可以看到。保護您的 Web 伺服器的最佳實務是安裝對 HTTPS (HTTP Secure) 的支援，HTTPS 使用 SSL/TLS 加密保護您的資料。

如需在伺服器上啟用 HTTPS 的相關資訊，請參閱 教學課程：在 AL2023 上設定 SSL/TLS。

步驟 2：測試您的 LAMP 伺服器

如果伺服器已安裝且正在執行，且檔案許可設定正確，則您的 ec2-user 帳戶應該能夠在可透過網際網路使用的 /var/www/html 目錄中建立 PHP 檔案。

測試您的 LAMP 伺服器

1. 在 Apache 文件根資料夾中建立 PHP 檔案。

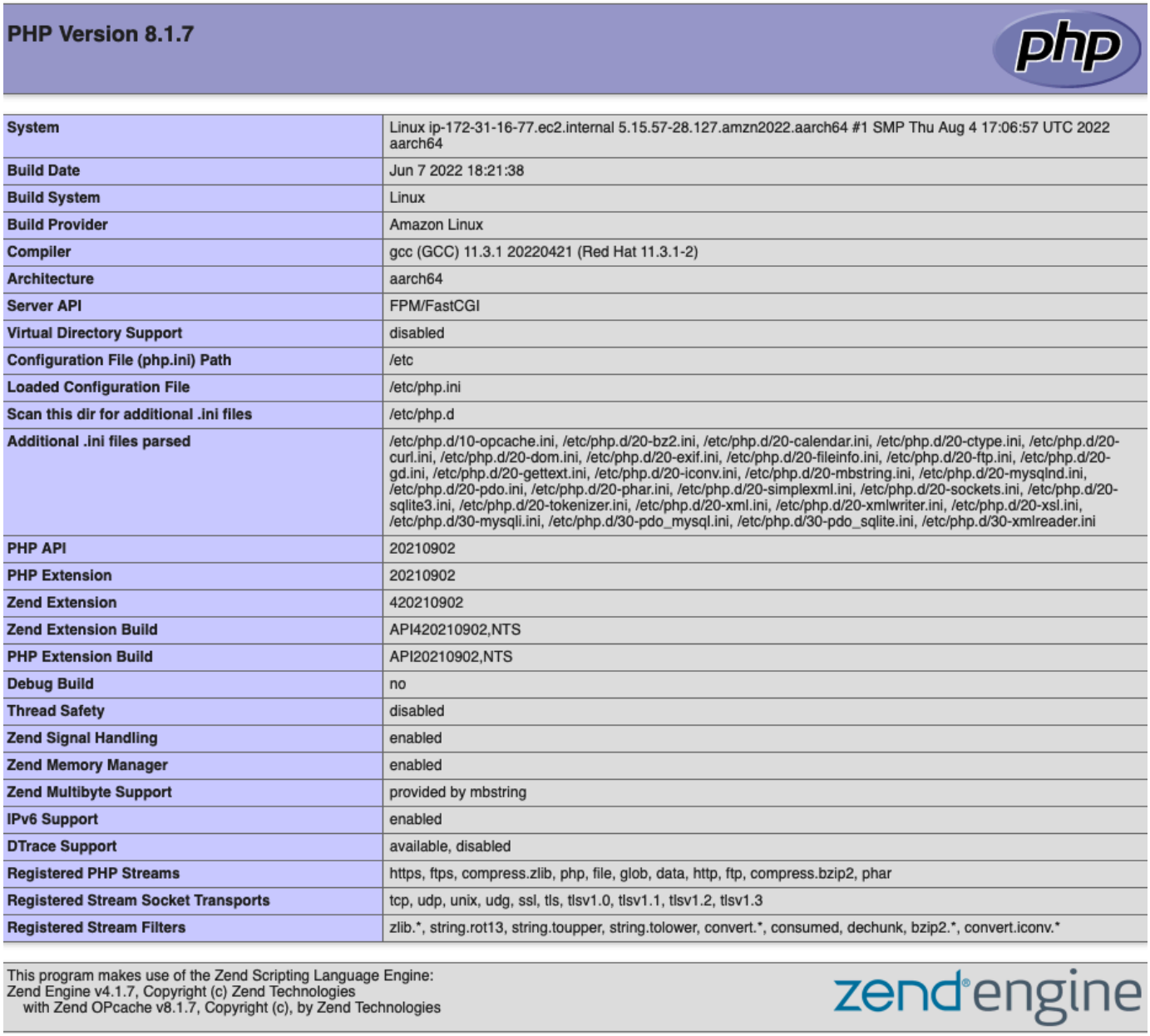
```
[ec2-user ~]$ echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php
```

   如果您嘗試執行此命令時出現拒絕許可錯誤，請嘗試登出並重新登入，以取得您在 設定檔案許可 設定的適當群組許可。

2. 在 Web 瀏覽器中，輸入您剛才建立的檔案 URL。此 URL 為您執行個體的公有 DNS 地址，其後跟隨斜線和檔案名稱。例如：

```
http://my.public.dns.amazonaws.com/phpinfo.php
```

   您現在應該會看見 PHP 資訊頁面：



PHP Version 8.1.7

System	Linux ip-172-31-16-77.ec2.internal 5.15.57-28.127.amzn2022.aarch64 #1 SMP Thu Aug 4 17:06:57 UTC 2022 aarch64
Build Date	Jun 7 2022 18:21:38
Build System	Linux
Build Provider	Amazon Linux
Compiler	gcc (GCC) 11.3.1 20220421 (Red Hat 11.3.1-2)
Architecture	aarch64
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/10-opcache.ini, /etc/php.d/20-bz2.ini, /etc/php.d/20-calendar.ini, /etc/php.d/20-ctype.ini, /etc/php.d/20-curl.ini, /etc/php.d/20-dom.ini, /etc/php.d/20-exif.ini, /etc/php.d/20-fileinfo.ini, /etc/php.d/20-ftp.ini, /etc/php.d/20-gd.ini, /etc/php.d/20-gettext.ini, /etc/php.d/20-iconv.ini, /etc/php.d/20-mbstring.ini, /etc/php.d/20-mysqlnd.ini, /etc/php.d/20-pdo.ini, /etc/php.d/20-phar.ini, /etc/php.d/20-simplexml.ini, /etc/php.d/20-sockets.ini, /etc/php.d/20-sqlite3.ini, /etc/php.d/20-tokenizer.ini, /etc/php.d/20-xml.ini, /etc/php.d/20-xmlwriter.ini, /etc/php.d/20-xsl.ini, /etc/php.d/30-mysqli.ini, /etc/php.d/30-pdo_mysql.ini, /etc/php.d/30-pdo_sqlite.ini, /etc/php.d/30-xmlreader.ini
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902,NTS
PHP Extension Build	API20210902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, compress.bzip2, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, convert.*, consumed, dechunk, bzip2.*, convert.iconv.*

This program makes use of the Zend Scripting Language Engine:
Zend Engine v4.1.7, Copyright (c) Zend Technologies
with Zend OPcache v8.1.7, Copyright (c), by Zend Technologies

如果你未看見此頁面，請確認 `/var/www/html/phpinfo.php` 檔案在前述步驟中正確建立。您也可以使用下列命令來確認已安裝所有必要的套件。

```
[ec2-user ~]$ sudo dnf list installed httpd mariadb-server php-mysqlnd
```

如果您的輸出未列出所需之任何套件，請使用 sudo yum install ***package*** 命令來安裝。

3. 刪除 `phpinfo.php` 檔案。雖然這可能是有用的資訊，但基於安全因素，您不應將其廣播至網際網路。

```
[ec2-user ~]$ rm /var/www/html/phpinfo.php
```

現在您應擁有功能齊全的 LAMP Web 伺服器。如果您將內容新增至 /var/www/html 的文件根目錄，則應該可以在執行個體的公有 DNS 地址檢視該內容。

步驟 3：保護資料庫伺服器

MariaDB 伺服器的預設安裝有幾項非常適合測試和開發的功能，但針對生產伺服器應將其停用或移除。mysql_secure_installation 命令將引導您完成設定根目錄密碼並從安裝中移除不安全功能的程序。即使您不打算使用 MariaDB 伺服器，我們也建議您執行此程序。

保護 MariaDB 伺服器

1. 啟動 MariaDB 伺服器。

   ```
   [ec2-user ~]$ sudo systemctl start mariadb
   ```

2. 執行 mysql_secure_installation。

   ```
   [ec2-user ~]$ sudo mysql_secure_installation
   ```

   a. 請在系統提示時，輸入根帳戶的密碼。

      i. 輸入目前的根密碼。根據預設，根帳戶並未設定密碼。按 Enter。

      ii. 輸入 **Y** 來設定密碼，然後輸入兩次安全密碼。如需建立安全密碼的詳細資訊，請參閱 https://identitysafe.norton.com/password-generator/。請確保此密碼存放於安全處。

      為 MariaDB 設定根密碼僅是確保資料庫安全的最基本措施。在建置或安裝資料庫驅動的應用程式時，通常會為該應用程式建立資料庫服務使用者，並避免使用根帳戶進行資料庫管理以外的任何作業。

   b. 輸入 **Y** 來移除匿名使用者帳戶。

   c. 輸入 **Y** 來停用遠端根登入。

   d. 輸入 **Y** 來移除測試資料庫。

   e. 輸入 **Y** 來載入使用者權限資料表並儲存您的變更。

3. (選用) 如果您不打算立即使用 MariaDB 伺服器，請將其停止。再次需要時，您可以將其重啟。

```
[ec2-user ~]$ sudo systemctl stop mariadb
```

4. (選用) 如果您希望 MariaDB 伺服器在每次系統開機時啟動，請輸入下列命令。

```
[ec2-user ~]$ sudo systemctl enable mariadb
```

步驟 4：(選用) 安裝 phpMyAdmin

phpMyAdmin 是一個 Web 型的資料庫管理工具，可用於檢視和編輯 EC2 執行個體上的 MySQL 資料庫。請遵循下列步驟在您的 Amazon Linux 執行個體上安裝並設定 phpMyAdmin。

> **Important**
>
> 我們不建議使用 phpMyAdmin 來存取 LAMP 伺服器，除非您在 Apache 中啟用了 SSL/TLS；否則，您的資料庫管理員密碼和其他資料將透過網際網路不安全傳輸。如需開發人員的安全性建議，請參閱保護您的 phpMyAdmin 安裝。如需有關在 EC2 執行個體上保護 Web 伺服器的一般資訊，請參閱教學課程：在 AL2023 上設定 SSL/TLS。

安裝 phpMyAdmin

1. 安裝所需的依存項目。

```
[ec2-user ~]$ sudo dnf install php-mbstring php-xml -y
```

2. 重新啟動 Apache。

```
[ec2-user ~]$ sudo systemctl restart httpd
```

3. 重新啟動 php-fpm。

```
[ec2-user ~]$ sudo systemctl restart php-fpm
```

4. 在 /var/www/html 中導覽至 Apache 文件根。

```
[ec2-user ~]$ cd /var/www/html
```

5. 從 https://www.phpmyadmin.net/downloads 選取最新 phpMyAdmin 版本的來源套件。若要直接將檔案下載到您的執行個體，請複製連結並將其貼入 wget 命令，如下所示：

```
[ec2-user html]$ wget https://www.phpmyadmin.net/downloads/phpMyAdmin-latest-all-languages.tar.gz
```

6. 以下列命令建立 phpMyAdmin 資料夾，並將套件擷取至該資料夾中。

```
[ec2-user html]$ mkdir phpMyAdmin && tar -xvzf phpMyAdmin-latest-all-languages.tar.gz -C phpMyAdmin --strip-components 1
```

7. 刪除 *phpMyAdmin-latest-all-languages.tar.gz* tarball。

```
[ec2-user html]$ rm phpMyAdmin-latest-all-languages.tar.gz
```

8. (選用) 如果 MySQL 伺服器未執行，請現在將其啟動。

```
[ec2-user ~]$ sudo systemctl start mariadb
```

9. 在 Web 瀏覽器中，輸入 phpMyAdmin 安裝的 URL。此 URL 為您執行個體的公有 DNS 地址 (或公有 IP 地址)，其後跟隨斜線和您安裝目錄的檔案名稱。例如：

```
http://my.public.dns.amazonaws.com/phpMyAdmin
```

   您現在應該會看見 phpMyAdmin 登入頁面：

10. 使用您之前建立的 `root` 使用者名稱和 MySQL 根密碼登入至您的 phpMyAdmin 安裝。

    必須先設定您的安裝，才能將其投入使用。我們建議您從手動建立組態檔案開始，如下所示：

    a. 若要從最小的組態檔開始，請使用您最愛的文字編輯器建立新檔案，然後將 `config.sample.inc.php` 的內容複製到其中。
    b. 將該檔案儲存為包含 `config.inc.php` 的 phpMyAdmin 目錄中的 `index.php`。
    c. 如需任何其他設定，請參閱 phpMyAdmin 安裝說明的 使用安裝指令碼一節中有關檔案建立後的說明。

    如需 phpMyAdmin 的資訊，請參閱 phpMyAdmin User Guide。

疑難排解

本節提供解決設定新 LAMP 伺服器時可能遇到之常見問題的建議。

我無法使用 Web 瀏覽器連線至我的伺服器

請執行下列檢查以查看您的 Apache Web 伺服器是否正在執行且可存取。

- Web 伺服器是否正在執行？

  要確認 httpd 已啟用，您可以執行以下命令：

  ```
  [ec2-user ~]$ sudo systemctl is-enabled httpd
  ```

  如果 httpd 程序未執行，請重複 準備 LAMP 伺服器所述的步驟。
- 防火牆是否設定正確？

  確認執行個體的安全性群組包含允許連接埠 80 上的 HTTP 流量的規則。如需詳細資訊，請參閱新增規則至安全群組。

我無法使用 HTTPS 連線至我的伺服器

請執行下列檢查以查看您的 Apache Web 伺服器是否設為支援 HTTPS。

- Web 伺服器是否正確設定？

  安裝 Apache 後，伺服器即針對 HTTP 流量進行設定。若要支援 HTTPS，請在伺服器上啟用 TLS 並安裝 SSL 憑證。如需相關資訊，請參閱教學課程：在 AL2023 上設定 SSL/TLS。
- 防火牆是否設定正確？

  確認執行個體的安全性群組包含允許連接埠 443 上的 HTTPS 流量的規則。如需詳細資訊，請參閱授權 Linux 執行個體的傳入流量。

相關主題

如需將檔案傳輸到執行個體或在 Web 伺服器上安裝 WordPress 部落格的詳細資訊，請參閱下列文件：

- 《Amazon EC2 使用者指南》中的使用 WinSCP 將檔案傳輸到您的 Linux 執行個體。

- 《Amazon EC2 使用者指南》中的使用 SCP 用戶端將檔案傳輸到 Linux 執行個體。
- 教學課程：託管 AL2023 上的 WordPress 部落格

如需本教學所使用的命令和軟體之詳細資訊，請參閱下列網頁：

- Apache Web 伺服器：http://httpd.apache.org/
- MariaDB 資料庫伺服器：https://mariadb.org/
- PHP 程式設計語言：http://php.net/

如需為您的 Web 伺服器註冊網域名稱或將現有網域名稱轉移至此主機的詳細資訊，請參閱Amazon Route 53 開發人員指南中的建立和遷移網域和子網域至 Amazon Route 53。

教學課程：在 AL2023 上設定 SSL/TLS

Secure Sockets Layer/Transport Layer Security (SSL/TLS) 會在 Web 伺服器與 Web 用戶端之間建立加密通路，保護傳輸中的資料以防遭到竊聽。本教學課程說明如何在具有 AL2023 和 Apache Web 伺服器的 EC2 執行個體上手動新增 SSL/TLS 支援。本教學假設您未使用負載平衡器。如果您正在使用 Elastic Load Balancing，您可以選擇使用來自 AWS Certificate Manager 的憑證設定負載平衡器上的 SSL 卸載。

基於歷史因素，Web 加密通常僅簡單以 SSL 指稱。雖然 Web 瀏覽器仍然支援 SSL，但其後來的通訊協定 TLS 較不易受攻擊。AL2023 預設會停用所有 SSL 版本的伺服器端支援。安全標準機構認為 TLS 1.0 不安全。TLS 1.0 和 TLS 1.1 已在 2021 年 3 月正式棄用。本教學課程包含完全基於啟用 TLS 1.2 的指導 TLS 1.3 已於 2018 年完成，且只要基礎 TLS 程式庫 （本教學課程中的 OpenSSL) 受到支援並啟用，即可在 AL2 中使用。用戶端必須在 2023 年 6 月 28 日之前支援 TLS 1.2 或更新版本。如需更新之加密標準的詳細資訊，請參閱 RFC 7568 和 RFC 8446。

本教學課程會將現代 Web 加密簡稱為 TLS。

> **Important**
>
> 這些程序旨在與 AL2023 搭配使用。如果您嘗試設定的 EC2 執行個體執行不同的發行版或執行舊版 Amazon Linux 的執行個體，則本教學中的一些程序可能不適用。若為 Ubuntu，請參閱以下 Ubuntu 社群文件：Open SSL on Ubuntu。對於 Red Hat Enterprise Linux，請參閱以下：設定 Apache HTTP Web 伺服器。若是其他發行版，請參閱其特定文件。

> **Note**
>
> 或者，您可以針對 AWS Nitro enclaves 使用 AWS Certificate Manager (ACM)，這是一個 enclave 應用程式，可讓您將公有和私有 SSL/TLS 憑證與在具有 AWS Nitro Enclaves 的 Amazon EC2 執行個體上執行的 Web 應用程式和伺服器搭配使用。Nitro Enclaves 是 Amazon EC2 功能，可建立隔離運算環境，保護並安全處理高度敏感資料，例如 SSL/TLS 憑證和私有金鑰。
> ACM for Nitro Enclaves 可搭配在您 Amazon EC2 Linux 執行個體上執行的 nginx，建立私有金鑰、分配憑證和私有金鑰，以及管理憑證續約。
> 若要使用 ACM for Nitro Enclaves，您必須使用啟用飛地的 Linux 執行個體。
> 如需詳細資訊，請參閱 AWS Nitro Enclaves 使用者指南中的什麼是 Nitro Enclaves？ AWS Certificate Manager 和 for AWS Nitro Enclaves。

目錄

先決條件

開始本教學之前，請先完成下列步驟：

- 啟動 EBS 支援的 AL2023 執行個體。如需詳細資訊，請參閱Amazon EC2 上的 AL2023。
- 設定安全群組允許執行個體接受下列 TCP 連接埠上的連線：
  - SSH (連接埠 22)
  - HTTP (連接埠 80)
  - HTTPS (連接埠 443)

  如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的授權 Linux 執行個體的傳入流量。
- 安裝 Apache Web 伺服器。如需逐步說明，請參閱 教學課程：在 AL2023 上安裝 LAMP Web 伺服器。只需要 httpd 套件和其相依性，因此您可以忽略包含 PHP 和 MariaDB 的說明。

- 若要識別和驗證網站，TLS 公有金鑰基礎設施 (PKI) 依賴網域名稱系統 (DNS)。若要使用 EC2 執行個體來託管公有網站，您需要註冊 Web 伺服器的網域名稱，或將現有網域名稱傳輸至 Amazon EC2 主機。現有多個第三方網域註冊和 DNS 託管服務可用，或者您可以使用 Amazon Route 53。

步驟 1：在伺服器上啟用 TLS

此程序會引導您完成使用自我簽署數位憑證在 AL2023 上設定 TLS 的程序。

> **Note**
>
> 自簽憑證可用於測試環境，而非生產環境。如果您在網際網路公開自簽憑證，來您網站光顧的訪客將會收到安全警告。

在伺服器上啟用 TLS

1. 連線至執行個體，並確認 Apache 正在執行。如需詳細資訊，請參閱連線至 AL2023 執行個體。

   ```
   [ec2-user ~]$ sudo systemctl is-enabled httpd
   ```

   如果傳回的值不是 "enabled"，請啟動 Apache，並設定為每次系統開機時都啟動。

   ```
   [ec2-user ~]$ sudo systemctl start httpd && sudo systemctl enable httpd
   ```

2. 為確保所有軟體套件皆為最新版本，請對您的執行個體執行快速軟體更新。本程序可能需費時幾分鐘，但確定您擁有最新的安全更新和錯誤修正至關重要。

   > **Note**
   >
   > -y 選項不要求確認就會安裝更新。如果您要先檢查更新再安裝，則可以略過此選項。

   ```
   [ec2-user ~]$ sudo dnf install openssl mod_ssl
   ```

3. 輸入以下命令後，您將看到一條提示，從中可以輸入有關網站的資訊。

```
[ec2-user ~]$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /
etc/pki/tls/private/apache-selfsigned.key -out /etc/pki/tls/certs/apache-
selfsigned.crt
```

這會產生兩個檔案：

- /etc/pki/tls/private/apache-selfsigned.key（私有金鑰）
- /etc/pki/tls/certs/apache-selfsigned.crt（自我簽署憑證）

憑證檔案名稱符合 中 SSLCertificateFile指令指派的預設值/etc/httpd/conf.d/ssl.conf。

您的執行個體現在有下列檔案，供您用來設定安全伺服器和建立用於測試的憑證：

- /etc/httpd/conf.d/ssl.conf

  mod_ssl 的組態檔案。檔案中包含的指示詞會告知 Apache 可在何處找到加密金鑰和憑證、要允許的 TLS 通訊協定版本，以及要接受的加密密碼。這將是您的本機憑證檔案：

- /etc/pki/tls/certs/apache-selfsigned.crt
- /etc/pki/tls/private/apache-selfsigned.key

.crt 檔案包含自我簽署的憑證，而.key檔案包含私有金鑰。Apache 規定憑證和金鑰必須是 PEM 格式，此格式由 "BEGIN" 和 "END" 行所框住的 Base64 編碼 ASCII 字元構成，如下方的憑證縮寫範例所示。

```
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQD2KKx/8Zk94m1q
3gQMZF9ZN66Ls19+3tHAgQ5Fpo9KJDhzLjOOCI8u1PTcGmAah5kEitCEc0wzmNeo
BCl0wYR6G0rGaKtK9Dn7CuIjvubtUysVyQoMVPQ97ldeakHWeRMiEJFXg6kZZ0vr
GvwnKoMh3DlK44D9dX7IDua2PlYx5+eroA+1Lqf32ZSaAO0bBIMIYTHigwbHMZoT
...
56tE7THvH7vOEf4/iUOsIrEzaMaJ0mqkmY1A70qQGQKBgBF3H1qNRNHuyMcPODFs
27hDzPDinrquSEvoZIggkDMlh2irTiipJ/GhkvTpoQlv0fK/VXw8vSgeaBuhwJvS
LXU9HvYq0U6O4FgD3nAyB9hI0BE13r1HjUvbjT7moH+RhnNz6eqqdscCS09VtRAo
4QQvAqOa8UheYeoXLdWcHaLP
-----END PRIVATE KEY-----

-----BEGIN CERTIFICATE-----
MIIEazCCA1OgAwIBAgICWxQwDQYJKoZIhvcNAQELBQAwgbExCzAJBgNVBAYTAi0t
```

```
MRIwEAYDVQQIDAlTb21lU3RhdGUxETAPBgNVBAcMCFNvbWVDaXR5MRkwFwYDVQQK
DBBTb21lT3JnYW5pemF0aW9uMR8wHQYDVQQLDBZTb21lT3JnYW5pemF0aW9uYWxV
bml0MRkwFwYDVQQDDBBpcC0xNzItMzEtMjAtMjM2MSQwIgYJKoZIhvcNAQkBFhVy
...
z5rRUE/XzxRLBZOoWZpNWTXJkQ3uFYH6s/sBwtHpKKZMzOvDedREjNKAvk4ws6F0
CuIjvubtUysVyQoMVPQ97ldeakHWeRMiEJFXg6kZZ0vrGvwnKoMh3DlK44D9dlU3
WanXWehT6FiSZvB4sTEXXJN2jdw8g+sHGnZ8zCOsclknYhHrCVD2vnBlZJKSZvak
3ZazhBxtQSukFMOnWPP2a0DMMFGYUHOd0BQE8sBJxg==
-----END CERTIFICATE-----
```

檔案名稱和副檔名僅為使用上的方便，不會影響功能。例如，只要 `cert.crt` 檔案中的相關指示詞使用相同的名稱，您就可以呼叫憑證 `cert.pem`、`ssl.conf` 或其他檔案名稱。

> **Note**
>
> 當您將預設 TLS 檔案取代為自己的自訂檔案時，請確認自訂檔案為 PEM 格式。

4. 重新啟動 Apache。

```
[ec2-user ~]$ sudo systemctl restart httpd
```

> **Note**
>
> 請確定可以在 EC2 執行個體上存取 TCP 連接埠 443，如先前所述。

5. Apache Web 伺服器現在應該支援透過連接埠 443 的 HTTPS (安全 HTTP)。測試方式是將 EC2 執行個體的 IP 地址或完整網域名稱加上字首 **`https://`** 後，一起輸入到瀏覽器 URL 列。

   因為您要使用自簽的不受信任主機憑證連線至網站，所以瀏覽器可能會顯示一系列的安全警告。請覆寫警告，並繼續前往網站。

   如果預設 Apache 測試頁面開啟，即表示您已於伺服器順利設定 TLS。現在所有在瀏覽器與伺服器之間傳遞的資料皆會加密。

   > **Note**
   >
   > 為了避免網站訪客碰到警告畫面，您必須取得 CA 簽署的受信任憑證，其不僅會加密也可將您公開驗證為網站擁有者。

步驟 2：取得 CA 簽署的憑證

您可以使用下列程序取得 CA 簽署的憑證：

- 從私有金鑰產生憑證簽署請求 (CSR)
- 將 CSR 提交至憑證授權機構 (CA)
- 取得簽署的主機憑證
- 設定 Apache 來使用憑證

在密碼編譯方面，自簽的 TLS X.509 主機憑證與 CA 簽署的憑證完全相同。兩者的差異在於往來的形式，無關數學性質。CA 允諾會至少先驗證網域的所有權，再將憑證發給申請人。每個 Web 瀏覽器皆含有受瀏覽器廠商信任能執行這項操作的 CA 名單。X.509 憑證主要包含對應至私有伺服器金鑰的公有金鑰，以及以密碼編譯方式繫結至公有金鑰的 CA 簽章。瀏覽器透過 HTTPS 連接至 Web 伺服器時，伺服器會呈現憑證給瀏覽器，讓瀏覽器檢查其信任的 CA 名單。如果簽署者位於名單上，或可透過包含其他受信任簽署者的「信任鏈」存取，瀏覽器會與伺服器協議一快速加密資料通路，並載入頁面。

憑證通常因包含驗證請求的勞力而需耗費成本，因此請貨比三家。一些 CA 免費提供基本層級憑證。這些 CA 當中最值得注意的是 Let's Encrypt 專案，此專案也支援自動化憑證建立和續約程序。如需使用 Let's Encrypt 憑證的詳細資訊，請參閱取得 Certbot。

如果您打算提供商業級服務，AWS Certificate Manager 會是不錯的選擇。

主機憑證的基礎就是金鑰。自 2019 年起，政府和產業團體建議採用最小金鑰 (模數) 大小為 2048 位元的 RSA 金鑰，用以保護文件至 2030 年。OpenSSL 在 AL2023 中產生的預設模數大小為 2048 位元，適用於 CA 簽署的憑證。對於需要自訂金鑰的人員，例如具有較大模數或使用不同加密演算法的金鑰，以下程序提供選用的步驟。

> **Important**
>
> 除非您擁有已註冊和託管的 DNS 網域，否則有關取得 CA 簽署的主機憑證的這些指示將不適用。

取得 CA 簽署的憑證

1. 連線至執行個體，並導覽至 /etc/pki/tls/private/。這是存放伺服器的 TLS 私有金鑰的目錄。如果您偏好使用現有主機金鑰來產生 CSR，請跳到步驟 3。如需連接至執行個體的詳細資訊，請參閱 連線至 AL2023 執行個體

2. (選用) 產生新私有金鑰。以下是金鑰組態的一些範例。任何產生的金鑰皆可用於您的 Web 伺服器，但金鑰所實作的安全程度和類型會不同。

   - 範例 1：建立預設 RSA 主機金鑰。產生的檔案 **custom.key** 是 2048 位元 RSA 私有金鑰。

     ```
     [ec2-user ~]$ sudo openssl genrsa -out custom.key
     ```

   - 範例 2：建立具有較大模數的較嚴格 RSA 金鑰。產生的檔案 **custom.key** 是 4096 位元 RSA 私有金鑰。

     ```
     [ec2-user ~]$ sudo openssl genrsa -out custom.key 4096
     ```

   - 範例 3：建立具有密碼保護的 4096 位元加密 RSA 金鑰。產生的檔案 **custom.key** 是以 AES-128 密碼加密的 4096 位元 RSA 私有金鑰。

     > **Important**
     >
     > 加密金鑰可提供更好的安全，但因為加密的金鑰需要密碼，所以無法自動啟動與其相依的服務。每次使用此金鑰時，您都必須透過 SSH 連線提供密碼 (在前述範例中為 "abcde12345")。

     ```
     [ec2-user ~]$ sudo openssl genrsa -aes128 -passout pass:abcde12345 -out
      custom.key 4096
     ```

   - 範例 4：使用非 RSA 密碼來加密金鑰。RSA 密碼編譯因為其公有金鑰的大小 (取決於兩個大質數的乘積)，可能相當慢。不過，為 TLS 建立使用非 RSA 密碼的金鑰是有可能的。傳送對等安全層級時，以橢圓曲線數學原理為基礎的金鑰會較小，且運算速度較快。

     ```
     [ec2-user ~]$ sudo openssl ecparam -name prime256v1 -out custom.key -genkey
     ```

     結果是使用 prime256v1 (OpenSSL 支援的一種「具名曲線」) 的 256 位元橢圓曲線私有金鑰。根據 NIST，其密碼編譯強度略大於 2048 位元 RSA 金鑰。

     > **Note**
     >
     > 有別於 RSA 金鑰，並非所有 CA 都能為橢圓曲線型金鑰提供相同層級的支援。

請確定新的私有金鑰具有高限制的所有權和許可 (擁有者=root、群組=root、僅限擁有者的讀寫權)。命令如下範例所示。

```
[ec2-user ~]$ sudo chown root:root custom.key
[ec2-user ~]$ sudo chmod 600 custom.key
[ec2-user ~]$ ls -al custom.key
```

上述命令會產生下列結果。

```
-rw------- root root custom.key
```

在您建立和設定滿意的金鑰之後，即可建立 CSR。

3. 使用偏好的金鑰建立 CSR。以下範例使用 **custom.key**。

```
[ec2-user ~]$ sudo openssl req -new -key custom.key -out csr.pem
```

OpenSSL 會開啟對話方塊，並提示您輸入下表中顯示的資訊。對於基本的已驗證網域之主機憑證，Common Name (通用名稱) 以外的所有欄位皆為選用欄位。

名稱	描述	範例
Country Name (國家/地區名稱)	兩個字母的 ISO 縮寫，用來代表您的國家/地區。	US (=美國)
State or Province Name (州或省名稱)	您組織位在的州名或省名。此名稱不得使用縮寫。	華盛頓州
Locality Name (地區名稱)	您組織的位置 (例如城市)。	西雅圖

名稱	描述	範例
Organization Name (組織名稱)	您組織的完整法定名稱。請不要使用您組織名稱的縮寫。	範例公司
Organizational Unit Name (組織單位名稱)	額外組織資訊 (如果有的話)。	範例部門
Common Name (通用名稱)	此值必須完全符合您預期使用者會在瀏覽器輸入的 web 地址。這通常表示字首為主機名稱或別名的網域名稱，格式為 **`www.example.com`** 。在使用自簽憑證且沒有 DNS 解析的測試中，通用名稱可能只包含主機名稱。CA 也提供費用較高的憑證，其可接受萬用字元名稱 (例如 **`*.example.com`**)。	www.example.com
電子郵件地址	伺服器管理員的電子郵件地址。	someone@example.com

最後，OpenSSL 會提示您輸入選用的挑戰密碼。此密碼只會套用至 CSR 以及您與 CA 之間的交易，因此請遵循 CA 對於這個和另一個選用欄位 (選用公司名稱) 的建議。CSR 挑戰密碼不會影響伺服器操作。

產生的檔案 **`csr.pem`** 會包含您的公有金鑰、您公有金鑰的數位簽章，以及您輸入的中繼資料。

4. 將 CSR 提交給 CA。這通常包含在文字編輯器開啟 CSR 檔案，以及將內容複製至 Web 表單。此時，系統可能會要求您提供要放在憑證上的一或多個主體別名 (SAN)。如果 **`www.example.com`** 是通用名稱，則 **`example.com`** 會是不錯的 SAN，反之亦然。輸入其中任一名稱的網站訪客會看到無錯誤連線。如果您的 CA Web 表單允許這項操作，請在 SAN 清單中包含通用名稱。部分 CA 會自動予以包含。

   在核准您的請求之後，您會收到 CA 所簽署的新主機憑證。系統也可能會指示您下載「中繼憑證」檔案，其中包含完成 CA 信任鏈所需的其他憑證。

> **Note**
>
> 您的 CA 可能會傳送多種格式的檔案給您，以供不同用途所需。在本教學中，您只應該使用 PEM 格式的憑證檔案，而憑證檔案通常 (但不一定) 會標上 `.pem` 或 `.crt` 副檔名。如果您不確定要使用的檔案，請使用文字編輯器開啟檔案，並尋找包含一或多個以下列這一行開頭之區塊的檔案。
>
> ```
> - - - - -BEGIN CERTIFICATE - - - - -
> ```
>
> 此檔案的結尾也應該是下列這一行。
>
> ```
> - - - -END CERTIFICATE - - - - -
> ```
>
> 您也可以在命令列測試檔案，如下所示。
>
> ```
> [ec2-user certs]$ openssl x509 -in certificate.crt -text
> ```
>
> 確認檔案中出現這幾行。請不要使用結尾為 `.p7b`、`.p7c` 或類似副檔名的檔案。

5. 將新的 CA 簽署憑證和任何中繼憑證放在 `/etc/pki/tls/certs` 目錄中。

> **Note**
>
> 有數種方式可以將新的憑證上傳至 EC2 執行個體，但最直接且有益的方式是在本機電腦和執行個體上開啟文字編輯器 (例如，vi、nano 或記事本)，然後在其間複製和貼上檔案內容。對 EC2 執行個體執行這些操作時，您需要有 root [sudo] 許可。因此，您可以立即看到是否有任何許可或路徑問題。不過，請注意不要在複製內容時新增其他行，或以任何方式變更它們。

從 `/etc/pki/tls/certs`目錄中，檢查檔案擁有權、群組和許可設定是否符合高限制的 AL2023 預設值 （擁有者=根、群組=根、僅供擁有者讀取/寫入）。以下範例顯示要使用的命令。

```
[ec2-user certs]$ sudo chown root:root custom.crt
[ec2-user certs]$ sudo chmod 600 custom.crt
[ec2-user certs]$ ls -al custom.crt
```

這些命令應該會產生下列結果。

```
-rw------- root root custom.crt
```

中繼憑證檔案的許可較不嚴格 (擁有者=root、群組=root、擁有者可以寫入、群組可以讀取、世界可以讀取)。以下範例顯示要使用的命令。

```
[ec2-user certs]$ sudo chown root:root intermediate.crt
[ec2-user certs]$ sudo chmod 644 intermediate.crt
[ec2-user certs]$ ls -al intermediate.crt
```

這些命令應該會產生下列結果。

```
-rw-r--r-- root root intermediate.crt
```

6. 將您用來建立 CSR 的私有金鑰放在 /etc/pki/tls/private/ 目錄中。

> **Note**
>
> 有數種方式可以將自訂金鑰上傳至 EC2 執行個體，但最直接且有益的方式是在本機電腦和執行個體上開啟文字編輯器 (例如，vi、nano 或記事本)，然後在其間複製和貼上檔案內容。對 EC2 執行個體執行這些操作時，您需要有 root [sudo] 許可。因此，您可以立即看到是否有任何許可或路徑問題。不過，請注意不要在複製內容時新增其他行，或以任何方式變更它們。

從 /etc/pki/tls/private目錄中，使用下列命令來驗證檔案擁有權、群組和許可設定是否符合高限制的 AL2023 預設值 （擁有者=根、群組=根、僅供擁有者讀取/寫入）。

```
[ec2-user private]$ sudo chown root:root custom.key
[ec2-user private]$ sudo chmod 600 custom.key
[ec2-user private]$ ls -al custom.key
```

這些命令應該會產生下列結果。

```
-rw------- root root custom.key
```

7. 編輯 /etc/httpd/conf.d/ssl.conf 以反映新的憑證和金鑰檔案。

> **Note**
>
> 請確定SSLEngine on已在 VirtualHost 區塊中設定 。

a. 在 Apache 的 SSLCertificateFile 指示詞中，提供 CA 簽署主機憑證的路徑和檔案名稱：

```
SSLCertificateFile /etc/pki/tls/certs/custom.crt
```

b. 如果您收到中繼憑證檔案 (在此範例中為 intermediate.crt)，請使用 Apache 的 SSLCACertificateFile 指示詞提供其路徑和檔案名稱：

```
SSLCACertificateFile /etc/pki/tls/certs/intermediate.crt
```

> **Note**
>
> 部分 CA 會將主機憑證和中繼憑證結合至單一檔案，讓 SSLCACertificateFile 指示詞變得不必要。請參閱 CA 所提供的說明。

c. 在 Apache 的 custom.key 指示詞中，提供私有金鑰 (在此範例中為 SSLCertificateKeyFile) 的路徑和檔案名稱：

```
SSLCertificateKeyFile /etc/pki/tls/private/custom.key
```

8. 儲存 /etc/httpd/conf.d/ssl.conf，並重新啟動 Apache。

```
[ec2-user ~]$ sudo systemctl restart httpd
```

9. 在瀏覽器 URL 列輸入網域名稱 (字首為 https://)，以測試伺服器。瀏覽器應該會透過 HTTPS 載入測試頁面，而不會產生錯誤。

步驟 3：測試和強化安全組態

在您的 TLS 運作並向大眾公開之後，您應測試其實際安全程度。這項操作能夠利用線上服務輕鬆完成，例如 Qualys SSL Labs 可免費為您的安全設定執行透徹的分析。根據結果，您可以決定透過控

制所接受的通訊協定、偏好的密碼，以及排除的項目，來強化預設安全組態。如需詳細資訊，請參閱 how Qualys formulates its scores。

> **Important**
>
> 實際測試對於伺服器安全而言十分重要。微小的組態錯誤可能會導致嚴重安全漏洞和資料遺失。由於建議的安全實務為因應研究與浮現的威脅而持續地變動，所以定期安全稽核是良好伺服器管理的必要項目。

在 Qualys SSL Labs 網站上，輸入伺服器的完整網域名稱，格式為 **www.example.com**。約兩分鐘之後，您會收到網站的評等 (從 A 到 F)，以及發現之項目的詳細分析。下表摘要說明網域的報告，其設定與 AL2023 上的預設 Apache 組態相同，且具有預設 Certbot 憑證。

整體評分	B
憑證	100%
通訊協定支援	95%
金鑰交換	70%
密碼強度	90%

雖然概觀顯示組態大致上很正確，但詳細報告指出幾個潛在問題，這裡依嚴重程度列出：

✗ 支援某些較舊的瀏覽器使用 RC4 加密。加密是加密演算法的數學核心。用來加密 TLS 資料串流的快速加密 RC4 已知有數個嚴重缺點。除非您有絕佳理由來支援舊版瀏覽器，否則應該停用此功能。

✗ 支援舊版 TLS。此組態支援 TLS 1.0 (已廢除) 和 TLS 1.1 (即將廢除)。自 2018 年起，只建議使用 TLS 1.2。

✗ 不完全支援前向保密。前向保密是使用衍生自私有金鑰之暫時 (短暫) 工作階段金鑰來加密的演算法的一項功能。這表示攻擊者實際上無法解密 HTTPS 資料，即使他們擁有 Web 伺服器的長期私有金鑰也是一樣。

更正和打造前瞻性的 TLS 組態

1. 在文字編輯器中開啟組態檔案 `/etc/httpd/conf.d/ssl.conf`，在下一行的開頭輸入 "#"，以變更為註解。

```
#SSLProtocol all -SSLv3
```

2. 新增下列指示詞：

```
#SSLProtocol all -SSLv3
SSLProtocol -SSLv2 -SSLv3 -TLSv1 -TLSv1.1 +TLSv1.2
```

此指示詞明確停用 SSL 版本 2 和 3，以及 TLS 版本 1.0 和 1.1。對於完全只使用 TLS 1.2 的用戶端，伺服器現在拒絕接受其加密連線。指示詞中的冗長言詞更清楚向讀者表達伺服器的設定用途。

> **Note**
>
> 以此種方式停用 TLS 1.0 和 1.1 版，可封鎖小部分的過期 Web 瀏覽器存取您的網站。

修改允許的加密清單

1. 在組態檔案 `/etc/httpd/conf.d/ssl.conf` 中，找出含有 **`SSLCipherSuite`** 指示詞的區段，在現有一行的開頭輸入 "#"，以變更為註解。

```
#SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5
```

2. 指定明確的加密套件，並指定加密順序將前向保密列為優先，避免不安全的加密。這裡使用的 `SSLCipherSuite` 指示詞是根據 Mozilla SSL Configuration Generator 的輸出，此工具可針對您伺服器上執行的特定軟體來量身打造 TLS 組態。首先，根據下列命令的輸出來確定您的 Apache 和 OpenSSL 版本。

```
[ec2-user ~]$ yum list installed | grep httpd

[ec2-user ~]$ yum list installed | grep openssl
```

例如，如果傳回的資訊是 Apache 2.4.34 和 OpenSSL 1.0.2，我們會在產生器中輸入此資訊。如果您選擇 "modern" 相容性模型，則會建立 `SSLCipherSuite` 指示詞來強制實施安全性，但仍適用於大多數瀏覽器。如果您的軟體不支援新式組態，您可以更新軟體或改為選擇 "intermediate" 組態。

```
SSLCipherSuite ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-
ECDSA-CHACHA20-POLY1305:
```

```
ECDHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:
ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-
RSA-AES128-SHA256
```

挑選的加密在名稱中包含 ECDHE (Elliptic Curve Diffie-Hellman Ephemeral 的縮寫)。ephemeral 這個字表示前向保密。附帶效果是這些加密不支援 RC4。

建議您使用明確密碼清單，而不是依賴預設值或是內容不可見的簡潔指示詞。

將產生的指示詞複製到 `/etc/httpd/conf.d/ssl.conf` 中。

> **Note**
>
> 指示詞複製到 `/etc/httpd/conf.d/ssl.conf` 時必須是單行，而且加密名稱之間只有冒號 (沒有空格)，這裡顯示成多行是為了方便閱讀。

3. 最後，移除下列這一行開頭的 "#"，以取消註解。

```
#SSLHonorCipherOrder on
```

此指示詞會強制伺服器優先選擇排名較高的加密，包括 (在此案例中) 支援前向保密的加密。開啟此指示詞時，伺服器會先嘗試建立嚴密的安全連線，再備援至具有較低安全的允許密碼。

完成這兩道程序後，請儲存 `/etc/httpd/conf.d/ssl.conf` 的變更，並重新啟動 Apache。

如果您在 Qualys SSL Labs 上重新測試網域，應該會發現 RC4 漏洞和其他警告已消失，而摘要如下所示。

整體評分	A
憑證	100%
通訊協定支援	100%
金鑰交換	90%
密碼強度	90%

每個 OpenSSL 更新都會產生新密碼，並移除舊密碼的支援。將您的 EC2 AL2023 執行個體保持在up-to-date、留意來自 OpenSSL 的安全公告，並留意技術新聞中新安全性漏洞的報告。

疑難排解

- 除非我輸入密碼，否則我的 Apache Web 伺服器不會啟動。

  如果您已安裝一個加密、密碼受保護的私有伺服器金鑰，這會是預期行為。

  您可以移除金鑰的加密和密碼需求。假設您在預設目錄中有稱為 `custom.key` 的私有加密 RSA 金鑰，且其上的密碼為 **abcde12345**，請於 EC2 執行個體上執行下列命令，以產生此金鑰的未加密版本。

  ```
  [ec2-user ~]$ cd /etc/pki/tls/private/
  [ec2-user private]$ sudo cp custom.key custom.key.bak
  [ec2-user private]$ sudo openssl rsa -in custom.key -passin pass:abcde12345 -out
   custom.key.nocrypt
  [ec2-user private]$ sudo mv custom.key.nocrypt custom.key
  [ec2-user private]$ sudo chown root:root custom.key
  [ec2-user private]$ sudo chmod 600 custom.key
  [ec2-user private]$ sudo systemctl restart httpd
  ```

  Apache 現在應該會啟動，而且系統不會提示您輸入密碼。

- 我在執行 sudo dnf install -y mod_ssl 時收到錯誤。

  在您安裝 SSL 的必要套件時，可能會看到與下列類似的錯誤。

  ```
  Error: httpd24-tools conflicts with httpd-tools-2.2.34-1.16.amzn1.x86_64
  Error: httpd24 conflicts with httpd-2.2.34-1.16.amzn1.x86_64
  ```

  這通常表示您的 EC2 執行個體未執行 AL2023。本教學課程僅支援從官方 AL2023 AMI 新建立的執行個體。

教學課程：託管 AL2023 上的 WordPress 部落格

下列程序可協助您在 AL2023 執行個體上安裝、設定和保護 WordPress 部落格。本教學課程正確提供 Amazon EC2 的使用簡介，您可在該執行個體上完全掌控託管 WordPress 部落格的 web 伺服器，其並非一般傳統型的託管服務。

您需負責為伺服器更新軟體套件及維護安全性修補程式。對於不需要與 Web 伺服器組態直接互動的更自動化 WordPress 安裝， CloudFormation 此服務提供的 WordPress 範本也可以讓您快速開始使用。如需詳細資訊，請參閱 AWS CloudFormation 使用者指南中的入門。若您需要具備解偶資料庫的高可用性解決方案，請參閱 AWS Elastic Beanstalk 開發人員指南中的部署高可用性 WordPress 網站。

> **Important**
>
> 這些程序旨在與 AL2023 搭配使用。如需其他分發的詳細資訊，請參閱其特定文件。本教學課程中的許多步驟不適用於 Ubuntu 執行個體。如需在 Ubuntu 執行個體上安裝 WordPress 的協助，請參閱 Ubuntu 文件中的 WordPress。您也可以使用 CodeDeploy，在 Amazon Linux、macOS 或 Unix 系統上完成此任務。

主題

先決條件

我們強烈建議您將彈性 IP 地址 (EIP) 與您用來託管 WordPress 部落格的執行個體建立關聯。如此可避免您執行個體的公有 DNS 地址變更及損害安裝。如果您擁有網域名稱，而且您想將該網域名稱用於部落格，您可更新網域名稱的 DNS 記錄，使其指向您的 EIP 地址 (如需這類的協助，請聯絡您的網域名稱註冊商)。您可免費將一個 EIP 地址與執行中的執行個體建立關聯。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的彈性 IP 位址。教學課程：在 AL2023 上安裝 LAMP Web 伺服器教學課程亦提供設定安全群組以允許 HTTP 和 HTTPS 流量的步驟，還有多個步驟可確保您的 web 伺服器已設定正確的檔案許可權限。如需將規則新增至安全群組的詳細資訊，請參閱將規則新增至安全群組。

如果您沒有網域名稱可用於部落格，您可向 Route 53 註冊網域名稱，並將執行個體的 EIP 地址與網域名稱建立關聯。如需詳細資訊，請參閱 Amazon Route 53 開發人員指南中的使用 Amazon Route 53 註冊網域名稱。

安裝 WordPress

連接至您的執行個體，然後下載 WordPress 安裝套件。如需連線至執行個體的詳細資訊，請參閱連線至 AL2023 執行個體。

1. 使用下列命令，下載並安裝這些套件。

```
dnf install wget php-mysqlnd httpd php-fpm php-mysqli mariadb105-server php-json
 php php-devel -y
```

2. 您可能會注意到輸出中顯示帶有類似措辭的警告 (版本可能會隨時間而有所變化)：

```
WARNING:
  A newer release of "Amazon Linux" is available.

  Available Versions:

dnf upgrade --releasever=2023.0.20230202

    Release notes:
     https://aws.amazon.com

  Version 2023.0.20230204:
    Run the following command to update to 2023.0.20230204:

      dnf upgrade --releasever=2023.0.20230204 ... etc
```

作為最佳實務，我們建議儘可能保持作業系統的最新狀態，但您可能想要逐一查看每個版本，以確保您的環境中沒有衝突。若步驟 1 中記下的先前套件安裝失敗，您可能需要更新至所列出較新版本的其中一個，然後再試一次。

3. 請用 wget 命令下載最新的 WordPress 安裝套件。以下命令將一律下載最新版本。

```
[ec2-user ~]$ wget https://wordpress.org/latest.tar.gz
```

4. 解壓縮並解除封存安裝套件。安裝資料夾將解壓縮到名為 wordpress 的資料夾。

```
[ec2-user ~]$ tar -xzf latest.tar.gz
```

建立 WordPress 安裝的資料庫使用者和資料庫

您的 WordPress 安裝需要將資訊存放在資料庫，例如部落格文章和使用者評論。此程序協助您建立部落格的資料庫，以及有權在該資料庫中讀取和儲存資訊的使用者。

1. 啟動資料庫和 Web 伺服器。

```
[ec2-user ~]$ sudo systemctl start mariadb httpd
```

2. 以 root 使用者的身分登入資料庫伺服器。出現提示時，輸入您資料庫的 root 密碼；此密碼可能不同於您的 root 系統密碼，假如您尚未設定資料庫伺服器的密碼，此密碼可能為空白。

   如果您尚未建立資料庫伺服器的保護機制，請務必這麼做。如需詳細資訊，請參閱 步驟 3：保護資料庫伺服器(AL2023)。

```
[ec2-user ~]$ mysql -u root -p
```

3. 為您的 MySQL 資料庫建立使用者和密碼。您的 WordPress 安裝使用這些值與 MySQL 資料庫通訊。請輸入下列命令，並換成唯一的使用者名稱與密碼。

```
CREATE USER 'wordpress-user'@'localhost' IDENTIFIED BY 'your_strong_password';
```

   務必為使用者建立高強度密碼。請勿在密碼中使用單引號字元 (')，因為這會使上述命令中斷。請勿重複使用現有密碼，並將此密碼存放在安全的地方。

4. 建立資料庫。為資料庫提供一個描述性有意義的名稱，例如 wordpress-db。

> **Note**
>
> 以下命令中資料庫名稱前後的標點符號稱為反引號。標準鍵盤上的反引號 (`) 鍵通常位在 Tab 鍵上方。反引號不一定為必要，但反引號可讓您在資料庫名稱中使用其他的非法字元，例如連字號。

```
CREATE DATABASE `wordpress-db`;
```

5. 將資料庫完整權限授予先前建立的 WordPress 使用者。

```
GRANT ALL PRIVILEGES ON `wordpress-db`.* TO "wordpress-user"@"localhost";
```

6. 排清資料庫權限，以套用所有變更。

```
FLUSH PRIVILEGES;
```

7. 離開 mysql 用戶端。

```
exit
```

建立及編輯 wp-config.php 檔案

WordPress 安裝資料夾包含組態檔案範例，其名稱為 wp-config-sample.php。在此程序中您將複製該檔案，並依照您的特定組態編輯檔案。

1. 將 wp-config-sample.php 檔案複製到稱為 wp-config.php 的檔案。此動作將建立一個新的組態檔案，並保留原本的範例檔案做為備份。

```
[ec2-user ~]$ cp wordpress/wp-config-sample.php wordpress/wp-config.php
```

2. 用您喜愛的文字編輯器 (例如 wp-config.php 或 nano) 編輯 vim 檔案，並輸入您的安裝值。如果您沒有喜愛的文字編輯器，nano 很適合入門者。

```
[ec2-user ~]$ nano wordpress/wp-config.php
```

   a. 找出定義 DB_NAME 的行，並將 database_name_here 變更為您在Step 4的建立 WordPress 安裝的資料庫使用者和資料庫中建立的資料庫名稱。

```
define('DB_NAME', 'wordpress-db');
```

   b. 找出定義 DB_USER 的行，並將 username_here 變更為您在Step 3的建立 WordPress 安裝的資料庫使用者和資料庫中建立的資料庫使用者。

```
define('DB_USER', 'wordpress-user');
```

   c. 找出定義 DB_PASSWORD 的行，並將 password_here 變更為您在Step 3的建立 WordPress 安裝的資料庫使用者和資料庫中建立的高強度密碼。

```
define('DB_PASSWORD', 'your_strong_password');
```

d. 找出稱為 Authentication Unique Keys and Salts 的區段。這些 KEY 和 SALT 的值會將一層加密提供給 WordPress 使用者存放於本機機器的瀏覽器 Cookie。基本上，於此新增長的隨機值將使您的網站更安全。請前往 https://api.wordpress.org/secret-key/1.1/salt/ 隨機產生一組金鑰值，您可將其複製並貼入到 wp-config.php 檔案內。若要將文字貼入到 PuTTY 終端，請將游標移至想要貼入文字的位置，然後在 PuTTY 終端內按一下滑鼠右鍵。

   如需安全金鑰的詳細資訊，請參閱 https://wordpress.org/support/article/editing-wp-config-php/#security-keys。

   > **Note**
   >
   > 下列值僅供範例使用；請勿使用這些值進行安裝。

```
define('AUTH_KEY',         ' #U$$+[RXN8:b^-L 0(WU_+ c+WFkI~c]o]-bHw+)/
Aj[wTwSiZ<Qb[mghEXcRh-');
define('SECURE_AUTH_KEY',  'Zsz._P=l/|y.Lq)XjlkwS1y5NJ76E6EJ.AV0pCKZZB,*~*r ?
6OP$eJT@;+(ndLg');
define('LOGGED_IN_KEY',    'ju}qwre3V*+8f_zOWf?{LlGsQ]Ye@2Jh^,8x>)Y |;(^[Iw]Pi
+LG#A4R?7N`YB3');
define('NONCE_KEY',        'P(g62HeZxEes|LnI^i=H,[XwK9I&[2s|:?0N}VJM%?;v2v]v+;
+^9eXUahg@::Cj');
define('AUTH_SALT',        'C$DpB4Hj[JK:?{ql`sRVa:{:7yShy(9A@5wg+`JJVb1fk%_-
Bx*M4(qc[Qg%JT!h');
define('SECURE_AUTH_SALT', 'd!uRu#}+q#{f$Z?Z9uFPG.${+S{n~1M&%@~gL>U>NV<zpD-@2-
Es7Q1O-bp28EKv');
define('LOGGED_IN_SALT',   ';j{00P*owZf)kVD+FVLn-~ >.|Y%Ug4#I^*LVd9QeZ^&XmK|
e(76miC+&W&+^0P/');
define('NONCE_SALT',       '-97r*V/cgxLmp?Zy4zUU4r99QQ_rGs2LTd%P;|
_e1tS)8_B/,.6[=UK<J_y9?JWG');
```

e. 儲存檔案並結束您的文字編輯器。

將 WordPress 檔案安裝至 Apache 文件根底下

- 現在您已解壓縮安裝資料夾、建立 MySQL 資料庫和使用者，並自訂 WordPress 組態檔案，接著您可將安裝檔案複製到 web 伺服器文件根，並執行安裝指令碼以完成安裝。這些檔案的位置取決於您想要在 web 伺服器的實際根 (例如，*my.public.dns.amazonaws.com*) 或根底下之子目錄或資料夾 (例如，*my.public.dns.amazonaws.com/blog*) 下使用 WordPress 部落格。

- 如果您想讓 WordPress 在文件根下執行，請複製 wordpress 安裝目錄的內容 (非目錄本身)，如下所示：

```
[ec2-user ~]$ cp -r wordpress/* /var/www/html/
```

- 如果您想讓 WordPress 在文件根下的其他目錄下執行，請先建立目錄，然後將檔案複製到目錄中。在此範例中，WordPress 將在 blog 目錄中執行：

```
[ec2-user ~]$ mkdir /var/www/html/blog
[ec2-user ~]$ cp -r wordpress/* /var/www/html/blog/
```

> **Important**
>
> 基於安全起見，如果您不是要立即移至下一個步驟，現在請先停止 Apache web 伺服器 (httpd)。將安裝移到 Apache 文件根底下後，WordPress 安裝指令碼未受保護，攻擊者可能在 Apache web 伺服器執行時存取您的部落格。若要停止 Apache Web 伺服器，輸入命令 sudo service httpd stop。如果您要移至下一個步驟，則不需要停止 Apache web 伺服器。

允許 WordPress 使用永久連結

WordPress 永久連結需要使用 Apache .htaccess 檔案才能正常運作，但此檔案在 Amazon Linux 上並非預設啟用。請使用此程序允許 Apache 文件根下的所有覆寫。

1. 使用您喜愛的文字編輯器 (例如 httpd.conf 或 nano) 開啟 vim 檔案。如果您沒有喜愛的文字編輯器，nano 很適合入門者。

```
[ec2-user ~]$ sudo vim /etc/httpd/conf/httpd.conf
```

2. 找出開頭為 <Directory "/var/www/html"> 的區段。

```
<Directory "/var/www/html">
    #
    # Possible values for the Options directive are "None", "All",
    # or any combination of:
    #   Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
    #
    # Note that "MultiViews" must be named *explicitly* --- "Options All"
    # doesn't give it to you.
```

```
    #
    # The Options directive is both complicated and important.  Please see
    # http://httpd.apache.org/docs/2.4/mod/core.html#options
    # for more information.
    #
    Options Indexes FollowSymLinks

    #
    # AllowOverride controls what directives may be placed in .htaccess files.
    # It can be "All", "None", or any combination of the keywords:
    #   Options FileInfo AuthConfig Limit
    #
    AllowOverride None

    #
    # Controls who can get stuff from this server.
    #
    Require all granted
</Directory>
```

3. 變更上述區段中的 `AllowOverride None` 行，以讀取 `AllowOverride All`。

> **Note**
>
> 此檔案中有多個 `AllowOverride` 行；請確定您變更的是 `<Directory "/var/www/html">` 區段中的行。

```
AllowOverride All
```

4. 儲存檔案並結束您的文字編輯器。

在 AL2023 上安裝 PHP 圖形繪製程式庫

適用於 PHP 的 GD 程式庫可讓您修改影像。如果您需要裁剪部落格的標題映像，請安裝此程式庫。您安裝的 phpMyAdmin 版本可能需要此程式庫的特定最低版本 (例如 8.1 版)。

使用以下命令在 AL2023 上安裝 PHP 圖形繪製程式庫。例如，如果您從來源安裝 php8.1 作為安裝 LAMP 堆疊的一部分，則此命令會安裝 PHP 圖形繪製庫的 8.1 版。

```
[ec2-user ~]$ sudo dnf install php-gd
```

若要確認已安裝的版本，請使用下列命令：

```
[ec2-user ~]$ sudo dnf list installed | grep php-gd
```

下列為範例輸出：

```
php-gd.x86_64                  8.1.30-1.amzn2           @amazonlinux
```

在 Amazon Linux AMI 安裝 PHP 圖形繪製庫

適用於 PHP 的 GD 程式庫可讓您修改影像。如果您需要裁剪部落格的標題映像，請安裝此程式庫。您安裝的 phpMyAdmin 版本可能需要此程式庫的特定最低版本 (例如 8.1 版)。

若要確認哪些版本可用，請使用下列命令：

```
[ec2-user ~]$ dnf list | grep php
```

以下是 PHP 圖形繪製庫 (8.1 版) 輸出的範例行：

```
php8.1.aarch64                                                 8.1.7-1.amzn2023.0.1
                       @amazonlinux
php8.1-cli.aarch64                                             8.1.7-1.amzn2023.0.1
                       @amazonlinux
php8.1-common.aarch64                                          8.1.7-1.amzn2023.0.1
                       @amazonlinux
php8.1-devel.aarch64                                           8.1.7-1.amzn2023.0.1
                       @amazonlinux
php8.1-fpm.aarch64                                             8.1.7-1.amzn2023.0.1
                       @amazonlinux
php8.1-gd.aarch64                                              8.1.7-1.amzn2023.0.1
                       @amazonlinux
```

使用下列命令，在 Amazon Linux AMI 上安裝特定版本的 PHP 圖形繪製庫 (例如，php8.1 版)：

```
[ec2-user ~]$ sudo dnf install -y php8.1-gd
```

修正 Apache web 伺服器的檔案權限

WordPress 中可用的部分功能需要 Apache 文件根的寫入存取 (例如透過管理畫面上傳媒體)。如果您尚未這樣做，請先套用下列的群組成員資格與許可 (詳細資訊詳述於 LAMP web 伺服器教學課程)。

1. 將 /var/www 及其內容的檔案所有權授予 apache 使用者。

```
[ec2-user ~]$ sudo chown -R apache /var/www
```

2. 將 /var/www 及其內容的群組所有權授予 apache 群組。

```
[ec2-user ~]$ sudo chgrp -R apache /var/www
```

3. 變更 /var/www 及其子目錄的目錄許可，以新增群組寫入許可並設定日後子目錄的群組 ID。

```
[ec2-user ~]$ sudo chmod 2775 /var/www
[ec2-user ~]$ find /var/www -type d -exec sudo chmod 2775 {} \;
```

4. 請以遞迴方式變更 /var/www 及其子目錄的檔案許可。

```
[ec2-user ~]$ find /var/www -type f -exec sudo chmod 0644 {} \;
```

> **Note**
>
> 如果您也想要使用 WordPress 做為 FTP 伺服器，此處您需要更寬鬆的群組設定。請檢視推薦的 WordPress 中的步驟和安全設定以完成此操作。

5. 重新啟動 Apache web 伺服器，以套用新的群組與許可。

```
[ec2-user ~]$ sudo systemctl restart httpd
```

使用 AL2023 執行 WordPress 安裝指令碼

現在已就緒可安裝 WordPress。使用的指令取決於作業系統。此程序中的命令適用於 AL2023。使用下列程序搭配 AL2023 AMI。

1. 使用 systemctl 命令，確定每次系統開機時 httpd 和資料庫服務都會啟動。

```
[ec2-user ~]$ sudo systemctl enable httpd && sudo systemctl enable mariadb
```

2. 確認資料庫伺服器正在執行。

```
[ec2-user ~]$ sudo systemctl status mariadb
```

如果資料庫服務未執行，請啟動服務。

```
[ec2-user ~]$ sudo systemctl start mariadb
```

3. 確認 Apache web 伺服器 (httpd) 正在執行。

```
[ec2-user ~]$ sudo systemctl status httpd
```

如果 httpd 服務未執行，請啟動服務。

```
[ec2-user ~]$ sudo systemctl start httpd
```

4. 在網頁瀏覽器中，輸入您 WordPress 部落格的 URL (可能是執行個體的公有 DNS 地址，或是該地址後面加上 blog 資料夾)。接著應該會出現 WordPress 安裝指令碼。提供 WordPress 安裝所需的資訊。選擇安裝 WordPress 來完成安裝。如需詳細資訊，請參閱 WordPress 網站上的 Step 5: Run the Install Script (步驟 5：執行安裝指令碼)。

使用 AL2023 AMI 執行 WordPress 安裝指令碼

1. 使用 chkconfig 命令，確定每次系統開機時 httpd 和資料庫服務都會啟動。

```
[ec2-user ~]$ sudo chkconfig httpd on && sudo chkconfig mariadb on
```

2. 確認資料庫伺服器正在執行。

```
[ec2-user ~]$ sudo service mariadb status
```

如果資料庫服務未執行，請啟動服務。

```
[ec2-user ~]$ sudo service mariadb start
```

3. 確認 Apache web 伺服器 (httpd) 正在執行。

```
[ec2-user ~]$ sudo service httpd status
```

如果 httpd 服務未執行，請啟動服務。

```
[ec2-user ~]$ sudo service httpd start
```

4. 在網頁瀏覽器中，輸入您 WordPress 部落格的 URL (可能是執行個體的公有 DNS 地址，或是該地址後面加上 `blog` 資料夾)。接著應該會出現 WordPress 安裝指令碼。提供 WordPress 安裝所需的資訊。選擇安裝 WordPress 來完成安裝。如需詳細資訊，請參閱 WordPress 網站上的 Step 5: Run the Install Script (步驟 5：執行安裝指令碼)。

後續步驟

在完成 WordPress 部落格的測試之後，您可以考慮更新其組態。

使用自訂的網域名稱

如果您有網域名稱與 EC2 執行個體的 EIP 地址相關聯，您可設定部落格使用該名稱，而不是使用 EC2 公有 DNS 地址。如需詳細資訊，請參閱 WordPress 網站上的 Changing The Site URL (變更網站 URL)。

設定部落格

您可設定部落格使用不同的主題和外掛程式，為讀者提供更為個人化的使用體驗。但安裝程序有時會出錯，使您失去整個部落格。因此我們強烈建議您為執行個體建立備份的 Amazon Machine Image (AMI)，然後再嘗試安裝任何主題或外掛程式，如此安裝期間發生任何錯誤時便能還原部落格。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的建立您自己的 AMI。

增加容量

若您的 WordPress 部落格變得十分熱門而需要更多運算能力或儲存空間，請參考下列步驟：

- 擴展執行個體的儲存空間。如需詳細資訊，請參閱 Amazon EBS 彈性磁碟區。
- 將 MySQL 資料庫移往 Amazon RDS，善加運用該服務可輕鬆擴展的能力。

改善網際網路流量的網路效能

如果您希望您的部落格能夠提高來自世界各地使用者的流量，請考慮 AWS Global Accelerator。Global Accelerator 可改善使用者用戶端裝置與 AWS上執行的 WordPress 應用程式之間的網際網路流量效能，協助您實現較低的延遲。Global Accelerator 使用AWS 全球網路，將流量導向最接近用戶端的 AWS 區域中運作狀態良好的應用程式端點。

進一步了解 WordPress

下列連結包含 WordPress 的詳細資訊。

- 如需有關 WordPress 的資訊，請參閱 Codex 中的 WordPress Codex 說明文件。
- 如需有關對安裝進行故障診斷的詳細資訊，請前往常見安裝問題。
- 如需讓 WordPress 部落格更安全的資訊，請前往 Hardening WordPress。
- 如需有關將 WordPress 部落格保持在up-to-date的資訊，請前往更新 WordPress。

說明! 我的公有 DNS 名稱曾經變更且現在部落格無法使用

您的 WordPress 安裝會自動設定為使用 EC2 執行個體的公有 DNS 地址。如果您停止並重新啟動執行個體，公有 DNS 地址便會變更 (除非與彈性 IP 地址建立關聯)，而且部落格將無法再使用，因為其參考資源的地址已不存在 (或已指派至其他的 EC2 執行個體)。如需更詳細的問題說明和多個可能的解決方案，請參閱 https://wordpress.org/support/article/changing-the-site-url/。

如果您的 WordPress 安裝發生此狀況，您也許可用下列程序使用 WordPress 的 wp-cli 命令列界面復原部落格。

使用 wp-cli 變更 WordPress 網站 URL

1. 使用 SSH 連結至您的 EC2 執行個體。
2. 記下執行個體的舊網站 URL 和新網站 URL。舊網站 URL 可能是安裝 WordPress 時 EC2 執行個體的公有 DNS 名稱。新網站 URL 為 EC2 執行個體目前的公有 DNS 名稱。如果您不確定舊網站的 URL，請用 curl 使用下列命令來尋找 URL。

   ```
   [ec2-user ~]$ curl localhost | grep wp-content
   ```

   輸出應該會顯示舊公有 DNS 名稱的參考，其看起來類似 (舊網站的 URL 為紅色)：

   ```
   <script type='text/javascript' src='http://ec2-52-8-139-223.us-
   west-1.compute.amazonaws.com/wp-content/themes/twentyfifteen/js/functions.js?
   ver=20150330'></script>
   ```

3. 使用下列命令下載 wp-cli。

   ```
   [ec2-user ~]$ curl -O https://raw.githubusercontent.com/wp-cli/builds/gh-pages/
   phar/wp-cli.phar
   ```

4. 使用下列命令搜尋並取代 WordPress 安裝中的舊網站 URL。取代 EC2 執行個體的舊網站和新網站 URL，以及 WordPress 安裝的路徑 (通常為 /var/www/html 或 /var/www/html/blog)。

```
[ec2-user ~]$ php wp-cli.phar search-replace 'old_site_url' 'new_site_url' --path=/
path/to/wordpress/installation --skip-columns=guid
```

5. 在 web 瀏覽器中輸入 WordPress 部落格的新網站 URL，再次確認網站是否正常運作。如果不是，請參閱變更網站 URL 和常見安裝問題以取得詳細資訊。

教學課程：在 AL2023 上將 Redis 6 轉換為 Valkey

下列文件說明從 Redis 6 到 AL2023 上 Valkey 轉換的關鍵層面。

Redis 6 的支援時間表

Redis 6 於 2027 年 1 月 31 日達到其生命週期結束 (EOL)。在此日期之後，Redis 6 將不再收到來自 Redis 專案的更新或安全修補程式。我們強烈建議使用者在 2027 年 1 月之前遷移到 Valkey，以確保持續支援和安全性更新。

如需 Redis 版本支援時間表的詳細資訊，請參閱 Redis End-Of-Life排程文件。

Valkey 簡介

Valkey 是 Redis 7 的開放原始碼分支，由 Linux Foundation 維護。它與 Redis 開放原始碼軟體 (OSS) 2.x 版到 7.2.x 版完全相容。Valkey 會維護熟悉的 Redis API 和功能，同時提供數種增強功能：

- 透過多執行緒增強效能。
- 改善記憶體效率，尤其是在叢集模式中。
- 雙通道複寫可提供更佳的資料一致性。

遷移計畫和時間表

強烈建議使用者在 Redis 6 達到其生命週期結束 (EOL) 時，於 2027 年 1 月 31 日之前從 Redis 6 遷移至 Valkey。此遷移需要手動介入，而且不是自動的。

Amazon Linux 建議此遷移，以確保 Redis 相依應用程式的持續功能、支援和安全性更新。

遷移選項和步驟

我們根據您的部署需求和操作需求，向 Valkey 提議三個遷移路徑。

選項 1：新的執行個體安裝

對於新部署或不需要資料遷移時：

1. 安裝 Valkey：

```
[ec2-user ~]$ sudo dnf install valkey
```

2. 啟動 Valkey：

```
[ec2-user ~]$ sudo systemctl start valkey
```

3. （選用）在開機時啟用 Valkey：

```
[ec2-user ~]$ sudo systemctl enable valkey
```

4. 驗證安裝：

```
[ec2-user ~]$ valkey-cli info server
[ec2-user ~]$ valkey-cli ping
```

選項 2：就地取代

對於不需要資料持久性的現有執行個體：

1. 停止 Redis 6：

```
[ec2-user ~]$ sudo systemctl stop redis6
```

2. 安裝 Valkey：

```
[ec2-user ~]$ sudo dnf install valkey
```

3. （選用）在 Valkey 中使用 Redis 6 組態：

```
[ec2-user ~]$ sudo cp /etc/redis6/redis6.conf /etc/valkey/valkey.conf
[ec2-user ~]$ sudo cp /etc/valkey/valkey.conf /etc/valkey/valkey.conf.backup
[ec2-user ~]$ sudo chown valkey:root /etc/valkey/valkey.conf
[ec2-user ~]$ sudo sed -i 's|^dir\s.*|dir /var/lib/valkey|g' /etc/valkey/
valkey.conf
```

4. （選用）在 Valkey 中使用 Redis 6 sentinel 組態檔案：

```
[ec2-user ~]$ sudo cp /etc/redis6/sentinel.conf /etc/valkey/sentinel.conf
[ec2-user ~]$ sudo chown valkey:root /etc/valkey/sentinel.conf
```

5. 啟動 Valkey：

```
[ec2-user ~]$ sudo systemctl start valkey
```

6. （選用）在開機時啟用 Valkey：

```
[ec2-user ~]$ sudo systemctl enable valkey
```

7. 驗證 Valkey 安裝：

```
[ec2-user ~]$ valkey-cli info server
[ec2-user ~]$ valkey-cli ping
```

8. 移除 Redis 6：

```
[ec2-user ~]$ sudo dnf remove redis6
```

選項 3：資料遷移

此選項可讓您同時執行 Redis 6 和 Valkey。

1. 安裝 Valkey 而不移除 Redis 6：

```
[ec2-user ~]$ sudo dnf install valkey
```

2. （選用）在 Valkey 中使用 Redis 6 組態：

```
[ec2-user ~]$ sudo cp /etc/redis6/redis6.conf /etc/valkey/valkey.conf
[ec2-user ~]$ sudo cp /etc/valkey/valkey.conf /etc/valkey/valkey.conf.backup
[ec2-user ~]$ sudo chown valkey:root /etc/valkey/valkey.conf
[ec2-user ~]$ sudo sed -i 's|^dir\s.*|dir /var/lib/valkey|g' /etc/valkey/
valkey.conf
```

3. （選用）在 Valkey 中使用 Redis 6 sentinel 組態檔案：

```
[ec2-user ~]$ sudo cp /etc/redis6/sentinel.conf /etc/valkey/sentinel.conf
```

```
[ec2-user ~]$ sudo chown valkey:root /etc/valkey/sentinel.conf
```

4. 修改 Valkey 組態：

   編輯 'port' 指令`/etc/valkey/valkey.conf`並將其設定為不同的值（例如 6380），以避免與 Redis 6 衝突。

5. 啟動 Valkey：

```
[ec2-user ~]$ sudo systemctl start valkey
```

6. （選用）在開機時啟用 Valkey：

```
[ec2-user ~]$ sudo systemctl enable valkey
```

7. 驗證 Valkey 安裝：

```
[ec2-user ~]$ valkey-cli -p port info server
[ec2-user ~]$ valkey-cli -p port ping
```

> **Note**
>
> 將###取代為設定的連接埠號碼。

8. 遷移資料：

   您現在可以使用複寫或手動資料傳輸方法，將資料從 Redis 6 遷移到 Valkey。

9. 更新應用程式組態：

   逐漸更新您的應用程式以使用 Valkey 連接埠。

10. 移除 Redis 6：

    遷移所有資料和應用程式後，您可以停止並移除 Redis 6。

```
[ec2-user ~]$ sudo systemctl stop redis6
[ec2-user ~]$ sudo dnf remove redis6
```

> **Note**
>
> 強烈建議在生產系統中實作變更之前，先驗證測試環境中的遷移程序。

相關主題

如需 Valkey 的詳細資訊：

- Valkey：https：//https://valkey.io/
- Valkey 遷移：https：//https://valkey.io/topics/migration/

教學課程：在 AL2023 上安裝 GNOME 桌面環境

GNOME 桌面環境自 2023.7 版或更新版本起，可作為 AL2023 的選用圖形使用者介面。

下列程序可協助您在 AL2023 執行個體上安裝 GNOME 桌面環境。您可以使用此圖形界面，使用熟悉的桌面環境來與 Linux 系統互動，而不只是命令列界面。

目錄

先決條件

- 桌面環境至少需要 2.4 GB 的記憶體。因此，建議使用 `t2.medium`或更好的執行個體，以確保足夠的效能。記憶體不足的執行個體類型範例包括 `t2.nano`、 `t2.micro`和 `t2.small`。此限制也適用於此大小的 `t3`和 `t4` 執行個體，以及不符合記憶體需求的任何其他執行個體類型。
- 本教學假設您已使用執行 2023.7 版或更新版本的 AL2023 啟動執行個體。如需詳細資訊，請參閱 Amazon EC2 上的 AL2023和 更新 AL2023頁面。

安裝

- 安裝 GNOME 桌面環境和相關套件。

```
[ec2-user ~]$ sudo dnf groupinstall "Desktop" -y
```

> **Note**
>
> 若要存取圖形桌面環境，您需要安裝和設定其他軟體，例如 Amazon DCV 或 VNC。這些工具可讓您透過網路連線至圖形使用者界面並與之互動。

相關主題

如需圖形桌面環境的詳細資訊，請參閱下列文件：

- 《Amazon DCV 管理員指南》中的什麼是 Amazon DCV？
- 教學課程：在 AL2023 上設定 TigerVNC 伺服器

教學課程：在 AL2023 上設定 TigerVNC 伺服器

下列程序可協助您在 AL2023 執行個體上設定 VNC 伺服器。VNC 可讓您透過安全的網路連線，遠端存取圖形桌面環境並與之互動。

目錄

先決條件

- 本教學假設您已在 AL2023 執行個體上安裝 GNOME 桌面環境。如需詳細資訊，請參閱 教學課程：在 AL2023 上安裝 GNOME 桌面環境頁面。

- 本教學課程使用 SSH 連接埠轉送來存取 VNC 伺服器。如需設定金鑰對的詳細資訊，請參閱《Amazon EC2 使用者指南》中的使用 SSH 連線至 Linux 執行個體。
- 下列程序不會引導您完成安裝 VNC 用戶端的程序。您的本機電腦上必須安裝 VNC 用戶端，才能連線至桌面環境並與之互動。

步驟 1：安裝

1. 連線到您的執行個體。如需詳細資訊，請參閱連線至 AL2023 執行個體。
2. 安裝 AL2023 的 TigerVNC 伺服器套件。

   -y 選項會安裝套件，而不要求確認。如果您想要在安裝之前檢查套件，可以省略此選項。

   ```
   [ec2-user ~]$ sudo dnf install -y tigervnc-server
   ```

步驟 2：組態

1. 確定使用者已設定 VNC 密碼。

   ```
   [ec2-user ~]$ vncpasswd
   ```

2. 將顯示編號指派給使用者。

   ```
   [ec2-user ~]$ sudo vi /etc/tigervnc/vncserver.users
   ```

   新增下列組態：

   ```
   :1=ec2-user
   ```

   > **Note**
   >
   > 您可以指派任何顯示號碼給使用者。基於此範例 `:1`，我們使用顯示器。

3. 編輯 VNC 伺服器組態檔案。

   ```
   [ec2-user ~]$ sudo vi /etc/tigervnc/vncserver-config-defaults
   ```

新增下列組態：

```
session=gnome
securitytypes=vncauth,tlsvnc
geometry=1920x1080
localhost
alwaysshared
```

> **Note**
>
> 您可以使用 geometry 參數變更顯示器的解析度。基於此範例，我們使用 1920x1080 。

4. 啟動 VNC 伺服器。每次重新啟動執行個體時，都需要重複此程序。如果您想要自動化啟動此服務的程序，請參閱以下選用章節。

```
[ec2-user ~]$ sudo systemctl start vncserver@:1
```

> **Important**
>
> 啟動vncserver服務時， 之後的部分 @必須符合 /etc/tigervnc/vncserver.users 檔案中為使用者設定的顯示號碼。

執行此步驟後，您可以從本機電腦建立 SSH 通道，並使用 VNC 用戶端進行連線。

步驟 3：使用 VNC 用戶端連線

VNC 伺服器會公開用於用戶端連線的 TCP 通訊端。雖然您可以直接透過安全群組公開 VNC 連接埠，但本教學課程透過加密本機電腦與 EC2 執行個體之間的連線，示範如何使用 SSH 通道做為更安全的方法。透過通道連線後，您將使用您在上一個步驟中設定的密碼來驗證 VNC 伺服器。如需安全群組的詳細資訊，請參閱《Amazon EC2 使用者指南》中的變更 Amazon EC2 執行個體的安全群組。Amazon EC2

1. 從本機電腦建立 SSH 通道。

```
$ ssh -i <keypair> -L 5901:localhost:5901 ec2-user@<address>
```

> **Note**
>
> <keypair> 將 取代為 SSH 金鑰的路徑，並將 <address>取代為執行個體的公有 IP 或 DNS 名稱。連接埠會根據用來啟動 的顯示號碼而變更 vncserver。例如，顯示器:1使用連接埠 5901，顯示器:2使用連接埠 5902等。

2. 使用您的 VNC 用戶端以先前設定的 VNC 127.0.0.1:5901 密碼連線至 localhost:5901或 。

> **Important**
>
> 使用 VNC 時保持 SSH 通道開啟。如果 SSH 通道未開啟，您將無法使用 VNC 用戶端來檢視桌面環境並與之互動。

（選用）在開機時啟動服務

如果您計劃定期使用 VNC，建議您將 VNC 伺服器設定為在執行個體開機時自動啟動。這樣就不需要在每次重新啟動執行個體時手動啟動 VNC 伺服器。此組態可確保您的圖形桌面環境在執行個體完成其啟動程序後立即準備就緒並可存取。

- 將服務設定為在開機時啟動。

```
[ec2-user ~]$ sudo systemctl enable vncserver@:1
```

> **Important**
>
> 啟用vncserver服務時， 之後的部分 @必須符合 /etc/tigervnc/vncserver.users 檔案中為使用者設定的顯示號碼。此外，您可以在 之後傳遞 --now 引數enable，以立即啟動服務。

執行此步驟後，您不再需要 vncserver在每次重新啟動執行個體時啟動 。

（選用）停用閒置鎖定畫面

- 將閒置延遲設定為零，以便在使用者長時間處於非作用中狀態時停用鎖定畫面。

```
[ec2-user ~]$ gsettings set org.gnome.desktop.session idle-delay 0
```

相關主題

如需圖形桌面環境的詳細資訊，請參閱下列文件：

- 教學課程：在 AL2023 上安裝 GNOME 桌面環境
- 《Amazon DCV 管理員指南》中的什麼是 Amazon DCV？

在 AL2023 核心上使用多世代 LRU (MGLRU)

多世代 LRU 是 Linux 核心中的現代頁面回收演算法，旨在改善記憶體壓力下的記憶體管理效能。它會取代傳統 LRU（最近最少使用）機制，用於判斷系統記憶體不足時要回收的記憶體頁面。

傳統 LRU 機制使用雙清單模型（作用中和非作用中）來追蹤頁面使用情況，這在具有大型工作集的現代工作負載中可能會變得效率低下。MGLRU 將其取代為多個「世代」頁面，允許核心根據更精細的老化資訊做出更明智的決策。

MGLRU 的優點包括：

- 更好的回收決策：更準確地識別冷（未使用的）頁面。
- 降低延遲並改善輸送量：特別是對於具有大型地址空間或許多並行程序的工作負載。
- 改善快取保留：最近使用的頁面不太可能過早移出。
- 可擴展且鎖定效率的設計：在具有許多 CPUs機器上表現更好。

組態和調校

在 AL2023 核心上`CONFIG_LRU_GEN`啟用核心組態。這會在 MGLRU 中編譯，但預設不會啟用它。

可以使用 `/sys/kernel/mm/lru_gen/enabled` 檔案啟用和調校 MGLRU。值為位元遮罩。建議啟用所有元件，除非其中一些元件有不良的副作用。

位元	元件
0	多世代 LRU 的主開關。
1	當 MMU 設定時（例如在 x86 上），以大型批次清除分葉頁面資料表項目中存取的位元。此行為理論上會加重鎖定爭用 (mmap_lock)。如果停用，對於連續映射熱頁面的工作負載，多世代 LRU 將遭受輕微的效能降低，其存取的位元可以由較少的較大批次清除。
2	當 MMU 設定時（例如在 x86 上），也會清除非分葉頁面資料表項目中存取的位元。此行為未在 Intel 和 AMD 以外的 x86 變體上驗證。如果停用，多世代 LRU 將遭受可忽略的效能降低。
[yYnN]	啟用/停用上述所有元件。

如何啟用 MGLRU 的範例：

```
[ec2-user ~]$ echo y >/sys/kernel/mm/lru_gen/enabled
```

這會啟用所有元件：

```
[ec2-user ~]$ cat /sys/kernel/mm/lru_gen/enabled
0x0007
```

在 Amazon EC2 之外使用 Amazon Linux 2023

Amazon Linux 2023 容器映像可以在相容的容器執行期環境中執行。如需如何在容器內使用 Amazon Linux 2023 的詳細資訊，請參閱 容器中的 AL2023。

Amazon Linux 2023 (AL2023) 也可以作為虛擬化客體執行，而不是直接在 Amazon EC2 上執行。目前有 KVM(qcow2)、VMware (OVA) 和 Hyper-V (vhdx) 映像可用。

> **Note**
>
> Amazon Linux 2023 映像的組態與 Amazon Linux 2 不同。
> 如果您是從將 Amazon Linux 2 作為內部部署的虛擬機器執行的環境轉換，則需要調整您的組態以與 AL2023 相容。

下載 Amazon Linux 2023 映像以搭配 KVM、VMware 和 Hyper-V 使用

用於 KVM、VMware 和 Hyper-V 的 Amazon Linux 2023 磁碟映像可從 cdn.amazonlinux.com ：//。

支援的 Amazon Linux 2023 組態，可用於非 Amazon EC2 虛擬化環境

本節涵蓋在 KVM、VMware 或 和 Hyper-V 等非 Amazon EC2 虛擬化環境中執行 Amazon Linux 2023 的需求。

此基礎 AL2023 系統需求 適用於所有非 Amazon EC2 虛擬化環境。下列主題詳細說明每個 Hypervisor 環境的支援裝置型號清單。

KVM、VMware 和 Hyper-V 提供許多組態選項，而且需要小心，才能根據您的安全性、效能和可靠性需求進行設定。如需詳細資訊，請查看 Hypervisor 提供的說明文件。

主題

- 在 KVM 上執行 AL2023 的需求
- 在 上執行 AL2023 的需求 VMware

- 在 Hyper-V 上執行 Amazon Linux 2023 的需求

在 KVM 上執行 AL2023 的需求

本節說明在 KVM 上執行 AL2023 的需求。AL2023 的 KVM 映像可同時用於 aarch64 和 x86-64 架構。這些要求是 KVM 映像AL2023 系統需求的基礎。

主題

- 在 KVM 上執行 AL2023 的 KVM 主機需求
- KVM 上 AL2023 的裝置支援
- KVM 上 AL2023 的開機模式 (UEFI 和 BIOS) 支援
- 在 KVM 上執行 AL2023 的限制

在 KVM 上執行 AL2023 的 KVM 主機需求

KVM 映像目前在執行 Ubuntu 22.04.3 LTS 的主機上符合資格，qemu其版本為 6.2+dfsg-2ubuntu6.15，由此 Ubuntu 版本提供，使用適用於 的q35機器類型x86-64和適用於 的virt機器類型aarch64。

KVM 上 AL2023 的裝置支援

經測試可與 AL2023 KVM 映像 (**aarch64** 和 **x86-64** 皆可) 搭配使用的 **qemu** 裝置型號：

- virtio-blk (virtio 區塊型儲存裝置)
- virtio-scsi (有磁碟裝置的 virtio SCSI 控制器)
- virtio-net (virtio 網路裝置)
- ahci (與虛擬光碟機搭配使用)
- usb-storage (透過 xhci)

在 AL2023 KVM 映像資格中啟用但未大量執行的其他**qemu**裝置模型包括：

- VGA (qemu VGA)，只在 x86-64 上
- virtio-rng (虛擬隨機數產生器)
- 舊式 AT 鍵盤和 PS/2 滑鼠裝置

- 舊版序列裝置

KVM 上 AL2023 的開機模式 (UEFI 和 BIOS) 支援

x86-64 映像會同時使用舊版 BIOS 和 UEFI 開機模式進行測試。aarch64 映像會以 UEFI 開機模式進行測試。

> **Note**
>
> 根據預設，使用UEFI開機模式時，有些虛擬機器管理員會使用啟用安全開機的 Microsoft 安全開機金鑰來佈建 VM。此組態將無法啟動 AL2023。
> 由於 AL2023 開機載入器不是由 Microsoft 簽署，因此 VM 必須佈建為沒有 UEFI 金鑰，或使用 Secure Boot 的 AL2023 金鑰。

> **Important**
>
> 尚未驗證KVM映像的安全開機支援。

在 KVM 上執行 AL2023 的限制

在 KVM 上執行 AL2023 有一些已知的限制。

> **Note**
>
> 實作一些列出的不支援功能的程式碼可能存在於 AL2023 中，並正常運作。存在不支援的功能清單，因此您可以做出明智的決策，以決定今天要倚賴哪些工作，以及 Amazon Linux 團隊在未來更新中符合哪些工作資格。

在 KVM 上執行 AL2023 的已知限制

- KVM 客體代理程式目前未封裝或支援。
- 不支援熱插拔 CPU、記憶體或任何其他裝置類型。
- 不支援 VM 休眠。
- 不支援 VM 遷移。

- 不支援任何裝置 (例如透過 PCI 直通或 USB 直通) 的傳遞。

在 上執行 AL2023 的需求 VMware

本節說明在 上執行 AL2023 的需求VMware。AL2023 VMware的影像僅適用於 x86-64架構。VMware的影像aarch64不可用或不支援。這些要求是VMware映像AL2023 系統需求基礎以外的要求。

主題

VMware 在 上執行 AL2023 的主機需求 VMware

AL2023 VMware OVA 映像目前符合下列資格：

- VMware 使用 Intel(R) Xeon(R) Platinum 8124M 處理器在主機上執行的工作站 17.5.0
- VMware 使用 Intel(R) Xeon(R) Platinum 8275CL 處理器的 vSphere 8.0 8275CL

AL2023 VMware OVA 映像會指定 13 的機器硬體版本。

VMware 機器硬體版本 13 受下列支援：

- ESXi 6.5 或更新版本
- VMware 工作站 14 或更新版本

上的 AL2023 裝置支援 VMware

下列VMware裝置模型已通過測試，可與 AL2023 VMware OVA 映像搭配使用 (**x86-64**僅限)：

- `vmw_pvscsi` (VMware 全虛擬化SCSI控制器)
- `vmxnet3` (VMware 全虛擬化網路裝置)
- `ata_piix` (僅限與虛擬光碟機搭配的舊式 IDE)

在 AL2023 VMware映像資格中啟用但並未大量執行的其他VMware裝置模型：

- `vmw_vmci` 和相關`vsock`界面 (VMware訪客客服人員的虛擬通訊端傳輸）
- `vmw_balloon` 記憶氣球裝置
- VMware SVGA 控制器
- 舊式 AT 鍵盤和 PS/2 滑鼠裝置

VMware 訪客代理程式套件 (`open-vm-tools`) 是可用的，預設會安裝在 AL2023 OVA VMware 影像中。

上的 AL2023 開機模式 (UEFI 和 BIOS) 支援 VMware

截至 2023.3.20231211 版本，AL2023 VMware OVA 映像已在舊版BIOS和UEFI開機模式中驗證。OVA 預設組態仍為舊版，BIOS但可由使用者變更。

> **Important**
>
> 安全開機支援需要 UEFI，尚未針對在 上執行的 AL2023 進行驗證VMware。

在 上執行 AL2023 的限制 VMware

在 上執行 AL2023 有一些已知的限制VMware。

> **Note**
>
> 實作某些列出的非支援功能的程式碼可能存在於 AL2023 中，而且運作正常。存在不支援的功能清單，因此客戶可以根據目前的工作需求做出明智的決策，以及 Amazon Linux 團隊會在未來更新中認證哪些工作符合資格。

在 上執行 AL2023 的已知限制 VMware

- UEFI 安全開機目前未在 上使用 AL2023 驗證VMware。
- 不支援熱插拔 CPU、記憶體或任何其他裝置類型。
- 不支援 VM 休眠。
- 不支援 VM 遷移。
- 不支援任何裝置 (例如透過 PCI 直通或 USB 直通) 的傳遞。

在 Hyper-V 上執行 Amazon Linux 2023 的需求

本節涵蓋在 Hyper-V 上執行 Amazon Linux 2023 的需求。 AL2023 的 Hyper-V 映像僅適用於 `x86-64`架構。目前無法使用或支援 的 Hyper-V `aarch64` 映像。

本節涵蓋 Hyper-V AL2023 系統需求 映像基礎上的其他需求。

主題

在 Hyper-V 上執行 Amazon Linux 2023 的 Hyper-V 主機需求

Amazon Linux 2023 on Hyper-V 的主要資格發生在在 EC2 執行個體上執行的 Windows Server 2022。 `c5.metal`

Hyper-V 上的 Amazon Linux 2023 裝置支援

Amazon Linux 2023 經過第 1 代和第 2 代 Hyper-V 虛擬機器測試，具有下列一組虛擬化硬體：

- 第 1 代 （ 舊版 BIOS 開機 ） VM
- 第 2 代 (UEFI 開機 - 無安全開機 ） VM
- 下列裝置模型經過測試，可與 AL2023 Hyper-V 映像搭配使用：
  - 第 2 代 VMs 上根磁碟和模擬 CD-ROM 磁碟機`hv_storvsc`的 Hyper-V 虛擬儲存
  - `ata_piix` 第 1 代 VMs 虛擬 CD-ROM 磁碟機的模擬 PIIX IDE
  - Hyper-V 虛擬乙太網路 `hv_netvsc`
- 下列裝置模型已啟用，但經過輕度測試：
  - 第 1 代 VMs 上的舊版 VGA 文字模式
  - 第 2 代 VMs `simpledrmfb`上的 UEFI 韌體架構緩衝
  - Hyper-V 氣球 `hv_balloon`
  - Hyper-V HID/滑鼠 `hid_hyperv`
- 下列裝置模式目前未在 AL2023 中啟用：
  - Hyper-V PCI 傳遞
  - Hyper-V DRM 圖形

> **Important**
>
> 對於第 2 代虛擬機器，不支援安全開機，必須在啟動虛擬機器之前停用，才能成功啟動 Amazon Linux 2023。Hyper-V 目前僅支援在 Amazon Linux 開機載入器由 Amazon 私有金鑰簽署時，使用 Microsoft 自有金鑰簽署的軟體元件進行安全開機。Hyper-V 目前不支援匯入第三方金鑰。

在 Hyper-V 上執行 Amazon Linux 2023 的限制

以下是在 Hyper-V 上執行 Amazon Linux 2023 的一些已知限制：

> **Note**
>
> 實作某些列出的非支援功能的程式碼可能存在於 AL2023 中，而且運作正常。存在不支援的功能清單，因此客戶可以根據目前的工作需求做出明智的決策，以及 Amazon Linux 團隊會在未來更新中認證哪些工作符合資格。

在 Hyper-V 上執行 AL2023 的已知限制

- 目前不支援 UEFI 安全開機模式，也不適用於 Hyper-V 上的 AL2023
- 不支援熱插拔 CPU、記憶體或任何其他裝置類型。
- 不支援虛擬機器 (VM) 休眠。
- 不支援虛擬機器 (VM) 遷移。
- 不支援任何裝置 (例如透過 PCI 直通或 USB 直通) 的傳遞。

Amazon Linux 2023 在 Amazon EC2 外部使用時的設定和 `cloud-init` 組態

本節說明如何在未直接在 Amazon EC2 上執行時設定 Amazon Linux 2023 虛擬機器，例如在 KVM、VMware 或 Hyper-V 上執行時。

預設情況下，Amazon Linux 2023 虛擬機器映像不會隨任何使用者密碼或 SSH 金鑰佈建，而是在第一個探索到的網路界面上透過 DHCP 取得網路組態。這表示在預設情況下，如果沒有其他設定，就無法連線到產生的虛擬機器。

因此，需要將某種形式的組態提供給虛擬機器。針對 Amazon Linux 執行此操作的標準機制是透過 `cloud-init` 資料來源進行。

Amazon Linux 2023 已通過以下資料來源的資格：

NoCloud

這是透過虛擬光碟機設定內部部署映像的傳統方法，其中包含具有 `cloud-init` 組態檔案的種子 ISO9660 映像。

VMware

此外，Amazon Linux 2023 還支援透過 VMware `guestinfo.userdata` 和 `guestinfo.metadata` 的 VMware 特定資料來源設定在 vSphere 上執行的 VMware 映像。

> **Note**
>
> 資料來源的組態可能與 Amazon Linux 2 不同。更具體來說，Amazon Linux 2023 會將 `systemd-networkd` 作為組態，且需要使用 `cloud-init` 網路組態說明文件所述的「Networking Config Version 2」。

如需 Amazon Linux 2023 中 `cloud-init` 封裝版本 `cloud-init` 組態機制的完整文件，請參閱上游 `cloud-init` 文件。

在 KVM 和 VMware 上適用於 Amazon Linux 2023 的 NoCloud (**`seed.iso`**) **`cloud-init`** 組態

本節說明如何建立和使用`seed.iso`映像來設定在 KVM或 上執行的 Amazon Linux 2023VMware。由於 KVM和 VMware環境沒有 Amazon EC2 執行個體中繼資料服務 (IMDS)，因此需要另一種方法來設定 Amazon Linux 2023，而提供`seed.iso`映像是這些方法之一。

`seed.iso` 開機映像包括虛擬機器開機和設定所需的初始組態資訊，例如網路組態、主機名稱和使用者資料。

> **Note**
>
> `seed.iso` 開機映像僅包括 VM 開機所需的組態資訊。不包括 Amazon Linux 2023 作業系統檔案。

若要產生 seed.iso 映像，您至少需要兩個組態檔案，有時需要三個：

meta-data

此檔案通常會包含虛擬機器的主機名稱。

user-data

此檔案通常會設定使用者帳戶，並指定其密碼、ssh 金鑰對和/或存取機制。預設情況下，Amazon Linux 2023 KVM 映像會建立 ec2-user 使用者帳戶。您可以使用 user-data 組態檔案來設定此預設使用者帳戶的密碼和/或 SSH 金鑰。

network-config (選用)

此檔案通常會為虛擬機器提供網路組態，該設定將覆寫預設的組態。預設組態是在第一個可用的網路介面上使用 DHCP。

建立 **seed.iso** 磁碟映像

1. 在 Linux 或 macOS 電腦上，建立名為 seedconfig 的新資料夾並導覽到該資料夾。

   > **Note**
   >
   > Windows 或其他作業系統也可以用來完成這些步驟，但您必須找到與 mkisofs 等效的工具，才能完成建立 seed.iso 映像。

2. 建立 meta-data 組態檔案。

   a. 建立名為 meta-data 的新檔案。

   b. 使用您偏好的編輯器開啟 meta-data 檔案、新增下列命令，然後將 *vm-hostname* 替換成虛擬機器的主機名稱：

   ```
   #cloud-config
   local-hostname: vm-hostname
   ```

   c. 儲存並關閉 meta-data 組態檔案。

3. 建立 user-data 組態檔案。

   a. 建立名為 user-data 的新檔案。

   b. 使用您偏好的編輯器開啟 user-data 檔案，然後新增下列項目，以及視需要進行取代：

```
#cloud-config
#vim:syntax=yaml
users:
# A user by the name 'ec2-user' is created in the image by default.
  - default
  - name: ec2-user
ssh_authorized_keys:
  - ssh-rsa ssh-key
# In the above line, replace ssh key with the content of your ssh public key.
```

c. 您可以選擇將更多使用者帳戶新增至user-data組態檔案。

您也可以指定其他使用者帳戶、其存取機制、密碼和金鑰對。如需支援指令的詳細資訊，請參閱上游 cloud-init 文件。

d. 儲存並關閉 user-data 組態檔案。

4. (選擇性) 建立 network-config 組態檔

a. 建立名為 network-config 的新檔案。

b. 使用您偏好的編輯器開啟 network-config 檔案、新增下列命令，然後將各種 IP 地址替換成適合您設定的 IP 地址。

```
#cloud-config
version: 2
ethernets:
  enp1s0:
    addresses:
      - 192.168.122.161/24
    gateway4: 192.168.122.1
    nameservers:
      addresses: 192.168.122.1
```

> **Note**
>
> cloud-init 網路組態提供與介面 MAC 位址相符的機制，而不是指定可根據 VM 組態變更的介面名稱。如需此 (及其他) cloud-init 網路設定功能的詳細資訊，請參閱上游 cloud-init Network Config Version 2 說明文件。

c. 儲存並關閉 network-config 組態檔案。

5. 使用在先前步驟中建立的 meta-data、user-data 和選用的 network-config 組態檔，以建立 seed.iso 磁碟映像。

   根據您建立 seed.iso 磁碟映像的作業系統，執行下列其中一個動作。

   - 在 Linux 系統上，使用 **mkisofs** 或 **genisoimage** 等的工具來建立完成的 seed.iso 檔案。導覽至 seedconfig 資料夾並執行下列命令：

     ```
     $ mkisofs -output seed.iso -volid cidata -joliet -rock user-data meta-data
     ```

   - 如果您使用 network-config，請將其包含在 **mkisofs** 的調用中：

     ```
     $ mkisofs -output seed.iso -volid cidata -joliet -rock user-data meta-data
      network-config
     ```

   - 在 macOS 系統上，您可以使用 **hdiutil** 等的工具來產生完成的 seed.iso 檔案。由於 **hdiutil** 採用路徑名稱而非檔案清單，因此無論是否建立 network-config 組態檔案，都可以使用相同的調用。

     ```
     $ hdiutil makehybrid -o seed.iso -hfs -joliet -iso -default-volume-name cidata
      seedconfig/
     ```

6. 產生的seed.iso檔案現在可以使用虛擬 CD-ROM 磁碟機連接至新的 Amazon Linux 2023 虛擬機器cloud-init，讓 在第一次開機時尋找 並將組態套用至系統。

VMware 上的 AL2023 guestinfo **cloud-init**組態 VMware

VMware 環境沒有 Amazon EC2 執行個體中繼資料服務 (IMDS)，因此需要替代方法來設定 AL2023。本節說明如何使用 VMware vSphere 中可用的seed.iso虛擬 CD-ROM 磁碟機的替代組態機制。

此組態方法使用 VMwareextraconfig機制將組態資料提供給 cloud-init。對於下列每個金鑰，必須提供對應的 ***keyname*.encoding** 屬性。

以下金鑰可以提供給 VMwareextraconfig機制。

guestinfo.metadata

JSON 或包含 cloud-init 中繼資料的 YAML

guestinfo.userdata

包含 cloud-config 格式 cloud-init 使用者資料的 YAML 文件。

guestinfo.vendordata (選用)

YAML 包含cloud-init廠商資料

對應的編碼屬性 (guestinfo.metadata.encoding、guestinfo.userdata.encoding 和 guestinfo.vendordata.encoding) 可以包含：

base64

屬性的內容已 base64 編碼。

gzip+base64

base64 編碼後，會以 gzip 壓縮屬性的內容。

> **Note**
>
> seed.iso 方法支援單獨的 （選用） network-config組態檔案。 提供聯網組態的方式 VMwareguestinfo有所不同。以下章節提供其他資訊。

如果需要明確的網路組態，應該以兩個 YAML 或 JSON 屬性的形式嵌入 metadata：

network

包含 JSON 或 YAML 格式的編碼網路組態。

network.encoding

包含上述網路組態資料的編碼。cloud-init 支援的編碼與 guestinfo 資料的編碼相同：base64 和 gzip+base64。

Example使用 VMware vSphere CLI govc 工具透過 傳遞組態 guestinfo

1. 準備 meta-data、 user-data和選用network-config組態檔案，如中所述在 KVM 和 VMware 上適用於 Amazon Linux 2023 的 NoCloud (seed.iso) cloud-init 組態。
2. 將組態檔案轉換為 VMware 可用的格式guestinfo。

```
# 'meta-data', `user-data` and `network-config` are the configuration
# files in the same format that would be used by a NoCloud (seed.iso)
# data source, read-them and convert them to VMware guestinfo
```

```
#
# The VM_NAME variable is assumed to be set to the name of the VM
# It is assumed that the necessary govc environment (credentials etc...) are
 already set

metadata=$(cat "meta-data")
userdata=$(cat "user-data")
if [ -e "network-config" ] ; then
    # We need to embed the network config inside the meta-data
    netconf=$(base64 -w0 "network-config")
    metadata=$(printf "%s\nnetwork: %s\nnetwork.encoding: base64" "$metadata"
 "$netconf")
fi
metadata=$(base64 -w0 <<< "$metadata")
govc vm.change -vm "$VM_NAME" \
    -e guestinfo.metadata="$metadata" \
    -e guestinfo.metadata.encoding="base64"
userdata=$(base64 -w0 <<< "$userdata")
govc vm.change -vm "$VM_NAME" \
    -e guestinfo.userdata="$userdata" \
    -e guestinfo.userdata.encoding="base64"
```

比較安裝在 Amazon Linux 2023 標準 AMI 上的套件與 AL2023 KVM Image

AL2023 標準 AMI 上出現RPMs 與 AL2023 KVM 映像上出現RPMs 的比較。

套件	AMI	KVM
acl	2.3.1	2.3.1
acpid	2.0.32	
alternatives	1.15	1.15
amazon-chrony-config	4.3	
amazon-ec2-net-utils	2.5.1	
amazon-linux-onprem		1.2

套件	AMI	KVM
amazon-linux-repo-cdn		2023.6.20241031
amazon-linux-repo-s3	2023.6.20241031	
amazon-linux-sb-keys	2023.1	2023.1
amazon-onprem-network		1.2
amazon-rpm-config	228	228
amazon-ssm-agent	3.3.987.0	3.3.987.0
amd-ucode-firmware	20210208 (noarch)	20210208 (noarch)
at	3.1.23	3.1.23
attr	2.5.1	2.5.1
audit	3.0.6	3.0.6
audit-libs	3.0.6	3.0.6
aws-cfn-bootstrap	2.0	
awscli-2	2.15.30	2.15.30
basesystem	11	11
bash	5.2.15	5.2.15
bash-completion	2.11	2.11
bc	1.07.1	1.07.1
bind-libs	9.18.28	9.18.28
bind-license	9.18.28	9.18.28

套件	AMI	KVM
bind-utils	9.18.28	9.18.28
binutils	2.39	2.39
boost-filesystem	1.75.0	1.75.0
boost-system	1.75.0	1.75.0
boost-thread	1.75.0	1.75.0
bzip2	1.0.8	1.0.8
bzip2-libs	1.0.8	1.0.8
ca-certificates	2023.2.68	2023.2.68
c-ares	1.19.1	
checkpolicy	3.4	3.4
chkconfig	1.15	1.15
chrony	4.3	4.3
cloud-init	22.2.2	22.2.2
cloud-init-cfg-ec2	22.2.2	
cloud-init-cfg-onprem		22.2.2
cloud-utils-growpart	0.31	0.31
coreutils	8.32	8.32
coreutils-common	8.32	8.32
cpio	2.13	2.13
cracklib	2.9.6	2.9.6

套件	AMI	KVM
cracklib-dicts	2.9.6	2.9.6
crontabs	1.11	1.11
crypto-policies	20220428	20220428
crypto-policies-scripts	20220428	20220428
cryptsetup	2.6.1	2.6.1
cryptsetup-libs	2.6.1	2.6.1
curl-minimal	8.5.0	8.5.0
cyrus-sasl-lib	2.1.27	2.1.27
cyrus-sasl-plain	2.1.27	2.1.27
dbus	1.12.28	1.12.28
dbus-broker	32	32
dbus-common	1.12.28	1.12.28
dbus-libs	1.12.28	1.12.28
device-mapper	1.02.185	1.02.185
device-mapper-libs	1.02.185	1.02.185
diffutils	3.8	3.8
dnf	4.14.0	4.14.0
dnf-data	4.14.0	4.14.0
dnf-plugin-release-notification	1.2	1.2

套件	AMI	KVM
dnf-plugins-core	4.3.0	4.3.0
dnf-plugin-support-info	1.2	1.2
dnf-utils	4.3.0	4.3.0
dosfstools	4.2	4.2
dracut	055	055
dracut-config-ec2	3.0	
dracut-config-generic	055	055
dwz	0.14	0.14
dyninst	10.2.1	10.2.1
e2fsprogs	1.46.5	1.46.5
e2fsprogs-libs	1.46.5	1.46.5
ec2-hibinit-agent	1.0.8	
ec2-instance-connect	1.1	
ec2-instance-connect-selinux	1.1	
ec2-utils	2.2.0	
ed	1.14.2	1.14.2
efi-filesystem	5	5
efi-srpm-macros	5	5

套件	AMI	KVM
efivar	38	38
efivar-libs	38	38
elfutils-debuginfod-client	0.188	0.188
elfutils-default-yama-scope	0.188	0.188
elfutils-libelf	0.188	0.188
elfutils-libs	0.188	0.188
ethtool	5.15	5.15
expat	2.5.0	2.5.0
file	5.39	5.39
file-libs	5.39	5.39
filesystem	3.14	3.14
findutils	4.8.0	4.8.0
fonts-srpm-macros	2.0.5	2.0.5
fstrm	0.6.1	0.6.1
fuse-libs	2.9.9	2.9.9
gawk	5.1.0	5.1.0
gdbm-libs	1.19	1.19
gdisk	1.0.8	1.0.8
gettext	0.21	0.21

套件	AMI	KVM
gettext-libs	0.21	0.21
ghc-srpm-macros	1.5.0	1.5.0
glib2	2.74.7	2.74.7
glibc	2.34	2.34
glibc-all-langpacks	2.34	2.34
glibc-common	2.34	2.34
glibc-gconv-extra	2.34	2.34
glibc-locale-source	2.34	2.34
gmp	6.2.1	6.2.1
gnupg2-minimal	2.3.7	2.3.7
gnutls	3.8.0	3.8.0
go-srpm-macros	3.2.0	3.2.0
gpgme	1.15.1	1.15.1
gpm-libs	1.20.7	1.20.7
grep	3.8	3.8
groff-base	1.22.4	1.22.4
grub2-common	2.06	2.06
grub2-efi-aa64-ec2	2.06 (aarch64)	2.06 (aarch64)
grub2-efi-x64-ec2	2.06 (x86_64)	2.06 (x86_64)
grub2-pc		2.06 (x86_64)

套件	AMI	KVM
grub2-pc-modules	2.06	2.06 (noarch)
grub2-tools	2.06	2.06
grub2-tools-minimal	2.06	2.06
grubby	8.40	8.40
gssproxy	0.8.4	0.8.4
gzip	1.12	1.12
hostname	3.23	3.23
hunspell	1.7.0	1.7.0
hunspell-en	0.20140811.1	0.20140811.1
hunspell-en-GB	0.20140811.1	0.20140811.1
hunspell-en-US	0.20140811.1	0.20140811.1
hunspell-filesystem	1.7.0	1.7.0
hwdata	0.384	0.384
info	6.7	6.7
inih	49	49
initscripts	10.09	10.09
iproute	6.10.0	6.10.0
iputils	20210202	20210202
irqbalance	1.9.0	1.9.0
jansson	2.14	2.14

套件	AMI	KVM
jemalloc	5.2.1	5.2.1
jitterentropy	3.4.1	3.4.1
jq	1.7.1	
json-c	0.14	0.14
kbd	2.4.0	2.4.0
kbd-misc	2.4.0	2.4.0
kernel	6.1.112	6.1.112
kernel-libbpf	6.1.112	6.1.112
kernel-livepatch-repo-cdn		2023.6.20241031
kernel-livepatch-repo-s3	2023.6.20241031	
kernel-modules-extra		6.1.112
kernel-modules-extra-common		6.1.112
kernel-srpm-macros	1.0	1.0
kernel-tools	6.1.112	6.1.112
keyutils	1.6.3	1.6.3
keyutils-libs	1.6.3	1.6.3
kmod	29	29
kmod-libs	29	29

套件	AMI	KVM
kpatch-runtime	0.9.7	0.9.7
krb5-libs	1.21.3	1.21.3
less	608	608
libacl	2.3.1	2.3.1
libaio	0.3.111	0.3.111
libarchive	3.7.4	3.7.4
libargon2	20171227	20171227
libassuan	2.5.5	2.5.5
libattr	2.5.1	2.5.1
libbasicobjects	0.1.1	0.1.1
libblkid	2.37.4	2.37.4
libcap	2.48	2.48
libcap-ng	0.8.2	0.8.2
libcbor	0.7.0	0.7.0
libcollection	0.7.0	0.7.0
libcom_err	1.46.5	1.46.5
libcomps	0.1.20	0.1.20
libconfig	1.7.2	1.7.2
libcurl-minimal	8.5.0	8.5.0
libdb	5.3.28	5.3.28

套件	AMI	KVM
libdhash	0.5.0	
libdnf	0.69.0	0.69.0
libeconf	0.4.0	0.4.0
libedit	3.1	3.1
libev	4.33	4.33
libevent	2.1.12	2.1.12
libfdisk	2.37.4	2.37.4
libffi	3.4.4	3.4.4
libfido2	1.10.0	1.10.0
libgcc	11.4.1	11.4.1
libgcrypt	1.10.2	1.10.2
libgomp	11.4.1	11.4.1
libgpg-error	1.42	1.42
libibverbs	48.0	48.0
libidn2	2.3.2	2.3.2
libini_config	1.3.1	1.3.1
libkcapi	1.4.0	1.4.0
libkcapi-hmaccalc	1.4.0	1.4.0
libldb	2.6.2	
libmaxminddb	1.5.2	1.5.2

套件	AMI	KVM
libmetalink	0.1.3	0.1.3
libmnl	1.0.4	1.0.4
libmodulemd	2.13.0	2.13.0
libmount	2.37.4	2.37.4
libnfsidmap	2.5.4	2.5.4
libnghttp2	1.59.0	1.59.0
libnl3	3.5.0	3.5.0
libpath_utils	0.2.1	0.2.1
libpcap	1.10.1	1.10.1
libpipeline	1.5.3	1.5.3
libpkgconf	1.8.0	1.8.0
libpsl	0.21.1	0.21.1
libpwquality	1.4.4	1.4.4
libref_array	0.1.5	0.1.5
librepo	1.14.5	1.14.5
libreport-filesystem	2.15.2	2.15.2
libseccomp	2.5.3	2.5.3
libselinux	3.4	3.4
libselinux-utils	3.4	3.4
libsemanage	3.4	3.4

套件	AMI	KVM
libsepol	3.4	3.4
libsigsegv	2.13	2.13
libsmartcols	2.37.4	2.37.4
libsolv	0.7.22	0.7.22
libss	1.46.5	1.46.5
libsss_certmap	2.9.4	
libsss_idmap	2.9.4	2.9.4
libsss_nss_idmap	2.9.4	2.9.4
libsss_sudo	2.9.4	
libstdc++	11.4.1	11.4.1
libstoragemgmt	1.9.4	1.9.4
libtalloc	2.3.4	
libtasn1	4.19.0	4.19.0
libtdb	1.4.7	
libtevent	0.13.0	
libtextstyle	0.21	0.21
libtirpc	1.3.3	1.3.3
libunistring	0.9.10	0.9.10
libuser	0.63	0.63
libutempter	1.2.1	1.2.1

套件	AMI	KVM
libuuid	2.37.4	2.37.4
libuv	1.47.0	1.47.0
libverto	0.3.2	0.3.2
libverto-libev	0.3.2	0.3.2
libxcrypt	4.4.33	4.4.33
libxml2	2.10.4	2.10.4
libyaml	0.2.5	0.2.5
libzstd	1.5.5	1.5.5
linux-firmware-whence	20210208 (noarch)	20210208 (noarch)
lm_sensors-libs	3.6.0	3.6.0
lmdb-libs	0.9.29	0.9.29
logrotate	3.20.1	3.20.1
lsof	4.94.0	4.94.0
lua-libs	5.4.4	5.4.4
lua-srpm-macros	1	1
lz4-libs	1.9.4	1.9.4
man-db	2.9.3	2.9.3
man-pages	5.10	5.10
microcode_ctl	2.1 (x86_64)	2.1 (x86_64)
mpfr	4.1.0	4.1.0

套件	AMI	KVM
nano	5.8	5.8
ncurses	6.2	6.2
ncurses-base	6.2	6.2
ncurses-libs	6.2	6.2
nettle	3.8	3.8
net-tools	2.0	2.0
newt	0.52.21	0.52.21
nfs-utils	2.5.4	2.5.4
npth	1.6	1.6
nspr	4.35.0	4.35.0
nss	3.90.0	3.90.0
nss-softokn	3.90.0	3.90.0
nss-softokn-freebl	3.90.0	3.90.0
nss-sysinit	3.90.0	3.90.0
nss-util	3.90.0	3.90.0
ntsysv	1.15	1.15
numactl-libs	2.0.14	2.0.14
ocaml-srpm-macros	6	6
oniguruma	6.9.7.1	
openblas-srpm-macros	2	2

套件	AMI	KVM
openldap	2.4.57	2.4.57
openssh	8.7p1	8.7p1
openssh-clients	8.7p1	8.7p1
openssh-server	8.7p1	8.7p1
openssl	3.0.8	3.0.8
openssl-libs	3.0.8	3.0.8
openssl-pkcs11	0.4.12	0.4.12
os-prober	1.77	1.77
p11-kit	0.24.1	0.24.1
p11-kit-trust	0.24.1	0.24.1
package-notes-srpm-macros	0.4	0.4
pam	1.5.1	1.5.1
parted	3.4	3.4
passwd	0.80	0.80
pciutils	3.7.0	3.7.0
pciutils-libs	3.7.0	3.7.0
pcre2	10.40	10.40
pcre2-syntax	10.40	10.40
perl-Carp	1.50	1.50
perl-Class-Struct	0.66	0.66

套件	AMI	KVM
perl-constant	1.33	1.33
perl-DynaLoader	1.47	1.47
perl-Encode	3.15	3.15
perl-Errno	1.30	1.30
perl-Exporter	5.74	5.74
perl-Fcntl	1.13	1.13
perl-File-Basename	2.85	2.85
perl-File-Path	2.18	2.18
perl-File-stat	1.09	1.09
perl-File-Temp	0.231.100	0.231.100
perl-Getopt-Long	2.52	2.52
perl-Getopt-Std	1.12	1.12
perl-HTTP-Tiny	0.078	0.078
perl-if	0.60.800	0.60.800
perl-interpreter	5.32.1	5.32.1
perl-IO	1.43	1.43
perl-IPC-Open3	1.21	1.21
perl-libs	5.32.1	5.32.1
perl-MIME-Base64	3.16	3.16
perl-mro	1.23	1.23

套件	AMI	KVM
perl-overload	1.31	1.31
perl-overloading	0.02	0.02
perl-parent	0.238	0.238
perl-PathTools	3.78	3.78
perl-Pod-Escapes	1.07	1.07
perl-podlators	4.14	4.14
perl-Pod-Perldoc	3.28.01	3.28.01
perl-Pod-Simple	3.42	3.42
perl-Pod-Usage	2.01	2.01
perl-POSIX	1.94	1.94
perl-Scalar-List-Utils	1.56	1.56
perl-SelectSaver	1.02	1.02
perl-Socket	2.032	2.032
perl-srpm-macros	1	1
perl-Storable	3.21	3.21
perl-subs	1.03	1.03
perl-Symbol	1.08	1.08
perl-Term-ANSIColor	5.01	5.01
perl-Term-Cap	1.17	1.17
perl-Text-ParseWords	3.30	3.30

套件	AMI	KVM
perl-Text-Tabs+Wrap	2021.0726	2021.0726
perl-Time-Local	1.300	1.300
perl-vars	1.05	1.05
pkgconf	1.8.0	1.8.0
pkgconf-m4	1.8.0	1.8.0
pkgconf-pkg-config	1.8.0	1.8.0
policycoreutils	3.4	3.4
policycoreutils-python-utils	3.4	
popt	1.18	1.18
procps-ng	3.3.17	3.3.17
protobuf-c	1.4.1	1.4.1
psacct	6.6.4	6.6.4
psmisc	23.4	23.4
publicsuffix-list-dafsa	20240212	20240212
python3	3.9.16	3.9.16
python3-attrs	20.3.0	20.3.0
python3-audit	3.0.6	3.0.6
python3-awscrt	2019 年 19 月 0 日	2019 年 19 月 0 日
python3-babel	2.9.1	2.9.1

套件	AMI	KVM
python3-cffi	1.14.5	1.14.5
python3-chardet	4.0.0	4.0.0
python3-colorama	0.4.4	0.4.4
python3-configobj	5.0.6	5.0.6
python3-cryptography	36.0.1	36.0.1
python3-daemon	2.3.0	
python3-dateutil	2.8.1	2.8.1
python3-dbus	1.2.18	1.2.18
python3-distro	1.5.0	1.5.0
python3-dnf	4.14.0	4.14.0
python3-dnf-plugins-core	4.3.0	4.3.0
python3-docutils	0.16	0.16
python3-gpg	1.15.1	1.15.1
python3-hawkey	0.69.0	0.69.0
python3-idna	2.10	2.10
python3-jinja2	2.11.3	2.11.3
python3-jmespath	0.10.0	0.10.0
python3-jsonpatch	1.21	1.21
python3-jsonpointer	2.0	2.0
python3-jsonschema	3.2.0	3.2.0

套件	AMI	KVM
python3-libcomps	0.1.20	0.1.20
python3-libdnf	0.69.0	0.69.0
python3-libs	3.9.16	3.9.16
python3-libselinux	3.4	3.4
python3-libsemanage	3.4	3.4
python3-libstorage mgmt	1.9.4	1.9.4
python3-lockfile	0.12.2	
python3-markupsafe	1.1.1	1.1.1
python3-netifaces	0.10.6	0.10.6
python3-oauthlib	3.0.2	3.0.2
python3-pip-wheel	21.3.1	21.3.1
python3-ply	3.11	3.11
python3-policycore utils	3.4	3.4
python3-prettytable	0.7.2	0.7.2
python3-prompt-too lkit	3.0.24	3.0.24
python3-pycparser	2.20	2.20
python3-pyrsistent	0.17.3	0.17.3
python3-pyserial	3.4	3.4

套件	AMI	KVM
python3-pysocks	1.7.1	1.7.1
python3-pytz	2022.7.1	2022.7.1
python3-pyyaml	5.4.1	5.4.1
python3-requests	2.25.1	2.25.1
python3-rpm	4.16.1.3	4.16.1.3
python3-ruamel-yaml	0.16.6	0.16.6
python3-ruamel-yaml-clib	0.1.2	0.1.2
python3-setools	4.4.1	4.4.1
python3-setuptools	59.6.0	59.6.0
python3-setuptools-wheel	59.6.0	59.6.0
python3-six	1.15.0	1.15.0
python3-systemd	235	235
python3-urllib3	1.25.10	1.25.10
python3-wcwidth	0.2.5	0.2.5
python-chevron	0.13.1	
python-srpm-macros	3.9	3.9
quota	4.06	4.06
quota-nls	4.06	4.06
readline	8.1	8.1

套件	AMI	KVM
rng-tools	6.14	6.14
rootfiles	8.1	8.1
rpcbind	1.2.6	1.2.6
rpm	4.16.1.3	4.16.1.3
rpm-build-libs	4.16.1.3	4.16.1.3
rpm-libs	4.16.1.3	4.16.1.3
rpm-plugin-selinux	4.16.1.3	4.16.1.3
rpm-plugin-systemd-inhibit	4.16.1.3	4.16.1.3
rpm-sign-libs	4.16.1.3	4.16.1.3
rsync	3.2.6	3.2.6
rust-srpm-macros	21	21
sbsigntools	0.9.4	0.9.4
screen	4.8.0	4.8.0
sed	4.8	4.8
selinux-policy	38.1.45	38.1.45
selinux-policy-targeted	38.1.45	38.1.45
setup	2.13.7	2.13.7
shadow-utils	4.9	4.9
slang	2.3.2	2.3.2

套件	AMI	KVM
sqlite-libs	3.40.0	3.40.0
sssd-client	2.9.4	2.9.4
sssd-common	2.9.4	
sssd-kcm	2.9.4	
sssd-nfs-idmap	2.9.4	
strace	6.8	6.8
sudo	1.9.15	1.9.15
sysctl-defaults	1.0	1.0
sysstat	12.5.6	12.5.6
systemd	252.23	252.23
systemd-libs	252.23	252.23
systemd-networkd	252.23	252.23
systemd-pam	252.23	252.23
systemd-resolved	252.23	252.23
systemd-udev	252.23	252.23
system-release	2023.6.20241031	2023.6.20241031
systemtap-runtime	4.8	4.8
tar	1.34	1.34
tbb	2020.3	2020.3
tcpdump	4.99.1	4.99.1

套件	AMI	KVM
tcsh	6.24.07	6.24.07
time	1.9	1.9
traceroute	2.1.3	2.1.3
tzdata	2024a	2024a
unzip	6.0	6.0
update-motd	2.2	2.2
userspace-rcu	0.12.1	0.12.1
util-linux	2.37.4	2.37.4
util-linux-core	2.37.4	2.37.4
vim-common	9.0.2153	9.0.2153
vim-data	9.0.2153	9.0.2153
vim-enhanced	9.0.2153	9.0.2153
vim-filesystem	9.0.2153	9.0.2153
vim-minimal	9.0.2153	9.0.2153
wget	1.21.3	1.21.3
which	2.21	2.21
words	3.0	3.0
xfsdump	3.1.11	3.1.11
xfsprogs	5.18.0	5.18.0
xxd	9.0.2153	9.0.2153

套件	AMI	KVM
xxhash-libs	0.8.0	0.8.0
xz	5.2.5	5.2.5
xz-libs	5.2.5	5.2.5
yum	4.14.0	4.14.0
zip	3.0	3.0
zlib	1.2.11	1.2.11
zram-generator	1.1.2	
zram-generator-defaults	1.1.2	
zstd	1.5.5	1.5.5

比較安裝在 Amazon Linux 2023 標準 AMI 上的套件與 AL2023 VMware OVA 映像

AL2023 標準 AMI 上出現RPMs 與 AL2023 VMware OVA 映像上出現RPMs 的比較。

套件	AMI	VMware OVA
acl	2.3.1	2.3.1
acpid	2.0.32	
alternatives	1.15	1.15
amazon-chrony-config	4.3	
amazon-ec2-net-utils	2.5.1	
amazon-linux-onprem		1.2

套件	AMI	VMware OVA
amazon-linux-repo-cdn		2023.6.20241031
amazon-linux-repo-s3	2023.6.20241031	
amazon-linux-sb-keys	2023.1	2023.1
amazon-onprem-network		1.2
amazon-rpm-config	228	228
amazon-ssm-agent	3.3.987.0	3.3.987.0
amd-ucode-firmware	20210208	20210208
at	3.1.23	3.1.23
attr	2.5.1	2.5.1
audit	3.0.6	3.0.6
audit-libs	3.0.6	3.0.6
aws-cfn-bootstrap	2.0	
awscli-2	2.15.30	2.15.30
basesystem	11	11
bash	5.2.15	5.2.15
bash-completion	2.11	2.11
bc	1.07.1	1.07.1
bind-libs	9.18.28	9.18.28
bind-license	9.18.28	9.18.28

套件	AMI	VMware OVA
bind-utils	9.18.28	9.18.28
binutils	2.39	2.39
boost-filesystem	1.75.0	1.75.0
boost-system	1.75.0	1.75.0
boost-thread	1.75.0	1.75.0
bzip2	1.0.8	1.0.8
bzip2-libs	1.0.8	1.0.8
ca-certificates	2023.2.68	2023.2.68
c-ares	1.19.1	
checkpolicy	3.4	3.4
chkconfig	1.15	1.15
chrony	4.3	4.3
cloud-init	22.2.2	22.2.2
cloud-init-cfg-ec2	22.2.2	
cloud-init-cfg-onprem		22.2.2
cloud-utils-growpart	0.31	0.31
coreutils	8.32	8.32
coreutils-common	8.32	8.32
cpio	2.13	2.13
cracklib	2.9.6	2.9.6

套件	AMI	VMware OVA
cracklib-dicts	2.9.6	2.9.6
crontabs	1.11	1.11
crypto-policies	20220428	20220428
crypto-policies-scripts	20220428	20220428
cryptsetup	2.6.1	2.6.1
cryptsetup-libs	2.6.1	2.6.1
curl-minimal	8.5.0	8.5.0
cyrus-sasl-lib	2.1.27	2.1.27
cyrus-sasl-plain	2.1.27	2.1.27
dbus	1.12.28	1.12.28
dbus-broker	32	32
dbus-common	1.12.28	1.12.28
dbus-libs	1.12.28	1.12.28
device-mapper	1.02.185	1.02.185
device-mapper-libs	1.02.185	1.02.185
diffutils	3.8	3.8
dnf	4.14.0	4.14.0
dnf-data	4.14.0	4.14.0
dnf-plugin-release-notification	1.2	1.2

套件	AMI	VMware OVA
dnf-plugins-core	4.3.0	4.3.0
dnf-plugin-support-info	1.2	1.2
dnf-utils	4.3.0	4.3.0
dosfstools	4.2	4.2
dracut	055	055
dracut-config-ec2	3.0	
dracut-config-generic	055	055
dwz	0.14	0.14
dyninst	10.2.1	10.2.1
e2fsprogs	1.46.5	1.46.5
e2fsprogs-libs	1.46.5	1.46.5
ec2-hibinit-agent	1.0.8	
ec2-instance-connect	1.1	
ec2-instance-connect-selinux	1.1	
ec2-utils	2.2.0	
ed	1.14.2	1.14.2
efi-filesystem	5	5
efi-srpm-macros	5	5

套件	AMI	VMware OVA
efivar	38	38
efivar-libs	38	38
elfutils-debuginfod-client	0.188	0.188
elfutils-default-yama-scope	0.188	0.188
elfutils-libelf	0.188	0.188
elfutils-libs	0.188	0.188
ethtool	5.15	5.15
expat	2.5.0	2.5.0
file	5.39	5.39
file-libs	5.39	5.39
filesystem	3.14	3.14
findutils	4.8.0	4.8.0
fonts-srpm-macros	2.0.5	2.0.5
fstrm	0.6.1	0.6.1
fuse3		3.10.4
fuse3-libs		3.10.4
fuse-common		3.10.4
fuse-libs	2.9.9	2.9.9
gawk	5.1.0	5.1.0

套件	AMI	VMware OVA
gdbm-libs	1.19	1.19
gdisk	1.0.8	1.0.8
gettext	0.21	0.21
gettext-libs	0.21	0.21
ghc-srpm-macros	1.5.0	1.5.0
glib2	2.74.7	2.74.7
glibc	2.34	2.34
glibc-all-langpacks	2.34	2.34
glibc-common	2.34	2.34
glibc-gconv-extra	2.34	2.34
glibc-locale-source	2.34	2.34
gmp	6.2.1	6.2.1
gnupg2-minimal	2.3.7	2.3.7
gnutls	3.8.0	3.8.0
go-srpm-macros	3.2.0	3.2.0
gpgme	1.15.1	1.15.1
gpm-libs	1.20.7	1.20.7
grep	3.8	3.8
groff-base	1.22.4	1.22.4
grub2-common	2.06	2.06

套件	AMI	VMware OVA
grub2-efi-x64-ec2	2.06	2.06
grub2-pc		2.06
grub2-pc-modules	2.06	2.06
grub2-tools	2.06	2.06
grub2-tools-minimal	2.06	2.06
grubby	8.40	8.40
gssproxy	0.8.4	0.8.4
gzip	1.12	1.12
hostname	3.23	3.23
hunspell	1.7.0	1.7.0
hunspell-en	0.20140811.1	0.20140811.1
hunspell-en-GB	0.20140811.1	0.20140811.1
hunspell-en-US	0.20140811.1	0.20140811.1
hunspell-filesystem	1.7.0	1.7.0
hwdata	0.384	0.384
info	6.7	6.7
inih	49	49
initscripts	10.09	10.09
iproute	6.10.0	6.10.0
iputils	20210202	20210202

套件	AMI	VMware OVA
irqbalance	1.9.0	1.9.0
jansson	2.14	2.14
jemalloc	5.2.1	5.2.1
jitterentropy	3.4.1	3.4.1
jq	1.7.1	
json-c	0.14	0.14
kbd	2.4.0	2.4.0
kbd-misc	2.4.0	2.4.0
kernel	6.1.112	6.1.112
kernel-libbpf	6.1.112	6.1.112
kernel-livepatch-repo-cdn		2023.6.20241031
kernel-livepatch-repo-s3	2023.6.20241031	
kernel-modules-extra		6.1.112
kernel-modules-extra-common		6.1.112
kernel-srpm-macros	1.0	1.0
kernel-tools	6.1.112	6.1.112
keyutils	1.6.3	1.6.3
keyutils-libs	1.6.3	1.6.3

套件	AMI	VMware OVA
kmod	29	29
kmod-libs	29	29
kpatch-runtime	0.9.7	0.9.7
krb5-libs	1.21.3	1.21.3
less	608	608
libacl	2.3.1	2.3.1
libaio	0.3.111	0.3.111
libarchive	3.7.4	3.7.4
libargon2	20171227	20171227
libassuan	2.5.5	2.5.5
libattr	2.5.1	2.5.1
libbasicobjects	0.1.1	0.1.1
libblkid	2.37.4	2.37.4
libcap	2.48	2.48
libcap-ng	0.8.2	0.8.2
libcbor	0.7.0	0.7.0
libcollection	0.7.0	0.7.0
libcom_err	1.46.5	1.46.5
libcomps	0.1.20	0.1.20
libconfig	1.7.2	1.7.2

套件	AMI	VMware OVA
libcurl-minimal	8.5.0	8.5.0
libdb	5.3.28	5.3.28
libdhash	0.5.0	
libdnf	0.69.0	0.69.0
libeconf	0.4.0	0.4.0
libedit	3.1	3.1
libev	4.33	4.33
libevent	2.1.12	2.1.12
libfdisk	2.37.4	2.37.4
libffi	3.4.4	3.4.4
libfido2	1.10.0	1.10.0
libgcc	11.4.1	11.4.1
libgcrypt	1.10.2	1.10.2
libgomp	11.4.1	11.4.1
libgpg-error	1.42	1.42
libibverbs	48.0	48.0
libidn2	2.3.2	2.3.2
libini_config	1.3.1	1.3.1
libkcapi	1.4.0	1.4.0
libkcapi-hmaccalc	1.4.0	1.4.0

套件	AMI	VMware OVA
libldb	2.6.2	
libmaxminddb	1.5.2	1.5.2
libmetalink	0.1.3	0.1.3
libmnl	1.0.4	1.0.4
libmodulemd	2.13.0	2.13.0
libmount	2.37.4	2.37.4
libmspack		0.10.1
libnfsidmap	2.5.4	2.5.4
libnghttp2	1.59.0	1.59.0
libnl3	3.5.0	3.5.0
libpath_utils	0.2.1	0.2.1
libpcap	1.10.1	1.10.1
libpipeline	1.5.3	1.5.3
libpkgconf	1.8.0	1.8.0
libpsl	0.21.1	0.21.1
libpwquality	1.4.4	1.4.4
libref_array	0.1.5	0.1.5
librepo	1.14.5	1.14.5
libreport-filesystem	2.15.2	2.15.2
libseccomp	2.5.3	2.5.3

套件	AMI	VMware OVA
libselinux	3.4	3.4
libselinux-utils	3.4	3.4
libsemanage	3.4	3.4
libsepol	3.4	3.4
libsigsegv	2.13	2.13
libsmartcols	2.37.4	2.37.4
libsolv	0.7.22	0.7.22
libss	1.46.5	1.46.5
libsss_certmap	2.9.4	
libsss_idmap	2.9.4	2.9.4
libsss_nss_idmap	2.9.4	2.9.4
libsss_sudo	2.9.4	
libstdc++	11.4.1	11.4.1
libstoragemgmt	1.9.4	1.9.4
libtalloc	2.3.4	
libtasn1	4.19.0	4.19.0
libtdb	1.4.7	
libtevent	0.13.0	
libtextstyle	0.21	0.21
libtirpc	1.3.3	1.3.3

套件	AMI	VMware OVA
libtool-ltdl		2.4.7
libunistring	0.9.10	0.9.10
libuser	0.63	0.63
libutempter	1.2.1	1.2.1
libuuid	2.37.4	2.37.4
libuv	1.47.0	1.47.0
libverto	0.3.2	0.3.2
libverto-libev	0.3.2	0.3.2
libxcrypt	4.4.33	4.4.33
libxml2	2.10.4	2.10.4
libxslt		1.1.34
libyaml	0.2.5	0.2.5
libzstd	1.5.5	1.5.5
linux-firmware-whence	20210208	20210208
lm_sensors-libs	3.6.0	3.6.0
lmdb-libs	0.9.29	0.9.29
logrotate	3.20.1	3.20.1
lsof	4.94.0	4.94.0
lua-libs	5.4.4	5.4.4
lua-srpm-macros	1	1

套件	AMI	VMware OVA
lz4-libs	1.9.4	1.9.4
man-db	2.9.3	2.9.3
man-pages	5.10	5.10
microcode_ctl	2.1	2.1
mpfr	4.1.0	4.1.0
nano	5.8	5.8
ncurses	6.2	6.2
ncurses-base	6.2	6.2
ncurses-libs	6.2	6.2
nettle	3.8	3.8
net-tools	2.0	2.0
newt	0.52.21	0.52.21
nfs-utils	2.5.4	2.5.4
npth	1.6	1.6
nspr	4.35.0	4.35.0
nss	3.90.0	3.90.0
nss-softokn	3.90.0	3.90.0
nss-softokn-freebl	3.90.0	3.90.0
nss-sysinit	3.90.0	3.90.0
nss-util	3.90.0	3.90.0

套件	AMI	VMware OVA
ntsysv	1.15	1.15
numactl-libs	2.0.14	2.0.14
ocaml-srpm-macros	6	6
oniguruma	6.9.7.1	
openblas-srpm-macros	2	2
openldap	2.4.57	2.4.57
openssh	8.7p1	8.7p1
openssh-clients	8.7p1	8.7p1
openssh-server	8.7p1	8.7p1
openssl	3.0.8	3.0.8
openssl-libs	3.0.8	3.0.8
openssl-pkcs11	0.4.12	0.4.12
open-vm-tools		12.3.0
os-prober	1.77	1.77
p11-kit	0.24.1	0.24.1
p11-kit-trust	0.24.1	0.24.1
package-notes-srpm-macros	0.4	0.4
pam	1.5.1	1.5.1
parted	3.4	3.4
passwd	0.80	0.80

套件	AMI	VMware OVA
pciutils	3.7.0	3.7.0
pciutils-libs	3.7.0	3.7.0
pcre2	10.40	10.40
pcre2-syntax	10.40	10.40
perl-Carp	1.50	1.50
perl-Class-Struct	0.66	0.66
perl-constant	1.33	1.33
perl-DynaLoader	1.47	1.47
perl-Encode	3.15	3.15
perl-Errno	1.30	1.30
perl-Exporter	5.74	5.74
perl-Fcntl	1.13	1.13
perl-File-Basename	2.85	2.85
perl-File-Path	2.18	2.18
perl-File-stat	1.09	1.09
perl-File-Temp	0.231.100	0.231.100
perl-Getopt-Long	2.52	2.52
perl-Getopt-Std	1.12	1.12
perl-HTTP-Tiny	0.078	0.078
perl-if	0.60.800	0.60.800

套件	AMI	VMware OVA
perl-interpreter	5.32.1	5.32.1
perl-IO	1.43	1.43
perl-IPC-Open3	1.21	1.21
perl-libs	5.32.1	5.32.1
perl-MIME-Base64	3.16	3.16
perl-mro	1.23	1.23
perl-overload	1.31	1.31
perl-overloading	0.02	0.02
perl-parent	0.238	0.238
perl-PathTools	3.78	3.78
perl-Pod-Escapes	1.07	1.07
perl-podlators	4.14	4.14
perl-Pod-Perldoc	3.28.01	3.28.01
perl-Pod-Simple	3.42	3.42
perl-Pod-Usage	2.01	2.01
perl-POSIX	1.94	1.94
perl-Scalar-List-Utils	1.56	1.56
perl-SelectSaver	1.02	1.02
perl-Socket	2.032	2.032
perl-srpm-macros	1	1

套件	AMI	VMware OVA
perl-Storable	3.21	3.21
perl-subs	1.03	1.03
perl-Symbol	1.08	1.08
perl-Term-ANSIColor	5.01	5.01
perl-Term-Cap	1.17	1.17
perl-Text-ParseWords	3.30	3.30
perl-Text-Tabs+Wrap	2021.0726	2021.0726
perl-Time-Local	1.300	1.300
perl-vars	1.05	1.05
pkgconf	1.8.0	1.8.0
pkgconf-m4	1.8.0	1.8.0
pkgconf-pkg-config	1.8.0	1.8.0
policycoreutils	3.4	3.4
policycoreutils-python-utils	3.4	
popt	1.18	1.18
procps-ng	3.3.17	3.3.17
protobuf-c	1.4.1	1.4.1
psacct	6.6.4	6.6.4
psmisc	23.4	23.4

套件	AMI	VMware OVA
publicsuffix-list-dafsa	20240212	20240212
python3	3.9.16	3.9.16
python3-attrs	20.3.0	20.3.0
python3-audit	3.0.6	3.0.6
python3-awscrt	2019 年 19 月 0 日	2019 年 19 月 0 日
python3-babel	2.9.1	2.9.1
python3-cffi	1.14.5	1.14.5
python3-chardet	4.0.0	4.0.0
python3-colorama	0.4.4	0.4.4
python3-configobj	5.0.6	5.0.6
python3-cryptography	36.0.1	36.0.1
python3-daemon	2.3.0	
python3-dateutil	2.8.1	2.8.1
python3-dbus	1.2.18	1.2.18
python3-distro	1.5.0	1.5.0
python3-dnf	4.14.0	4.14.0
python3-dnf-plugins-core	4.3.0	4.3.0
python3-docutils	0.16	0.16
python3-gpg	1.15.1	1.15.1

套件	AMI	VMware OVA
python3-hawkey	0.69.0	0.69.0
python3-idna	2.10	2.10
python3-jinja2	2.11.3	2.11.3
python3-jmespath	0.10.0	0.10.0
python3-jsonpatch	1.21	1.21
python3-jsonpointer	2.0	2.0
python3-jsonschema	3.2.0	3.2.0
python3-libcomps	0.1.20	0.1.20
python3-libdnf	0.69.0	0.69.0
python3-libs	3.9.16	3.9.16
python3-libselinux	3.4	3.4
python3-libsemanage	3.4	3.4
python3-libstorage mgmt	1.9.4	1.9.4
python3-lockfile	0.12.2	
python3-markupsafe	1.1.1	1.1.1
python3-netifaces	0.10.6	0.10.6
python3-oauthlib	3.0.2	3.0.2
python3-pip-wheel	21.3.1	21.3.1
python3-ply	3.11	3.11

套件	AMI	VMware OVA
python3-policycoreutils	3.4	3.4
python3-prettytable	0.7.2	0.7.2
python3-prompt-toolkit	3.0.24	3.0.24
python3-pycparser	2.20	2.20
python3-pyrsistent	0.17.3	0.17.3
python3-pyserial	3.4	3.4
python3-pysocks	1.7.1	1.7.1
python3-pytz	2022.7.1	2022.7.1
python3-pyyaml	5.4.1	5.4.1
python3-requests	2.25.1	2.25.1
python3-rpm	4.16.1.3	4.16.1.3
python3-ruamel-yaml	0.16.6	0.16.6
python3-ruamel-yaml-clib	0.1.2	0.1.2
python3-setools	4.4.1	4.4.1
python3-setuptools	59.6.0	59.6.0
python3-setuptools-wheel	59.6.0	59.6.0
python3-six	1.15.0	1.15.0
python3-systemd	235	235

套件	AMI	VMware OVA
python3-urllib3	1.25.10	1.25.10
python3-wcwidth	0.2.5	0.2.5
python-chevron	0.13.1	
python-srpm-macros	3.9	3.9
quota	4.06	4.06
quota-nls	4.06	4.06
readline	8.1	8.1
rng-tools	6.14	6.14
rootfiles	8.1	8.1
rpcbind	1.2.6	1.2.6
rpm	4.16.1.3	4.16.1.3
rpm-build-libs	4.16.1.3	4.16.1.3
rpm-libs	4.16.1.3	4.16.1.3
rpm-plugin-selinux	4.16.1.3	4.16.1.3
rpm-plugin-systemd-inhibit	4.16.1.3	4.16.1.3
rpm-sign-libs	4.16.1.3	4.16.1.3
rsync	3.2.6	3.2.6
rust-srpm-macros	21	21
sbsigntools	0.9.4	0.9.4
screen	4.8.0	4.8.0

套件	AMI	VMware OVA
sed	4.8	4.8
selinux-policy	38.1.45	38.1.45
selinux-policy-targeted	38.1.45	38.1.45
setup	2.13.7	2.13.7
shadow-utils	4.9	4.9
slang	2.3.2	2.3.2
sqlite-libs	3.40.0	3.40.0
sssd-client	2.9.4	2.9.4
sssd-common	2.9.4	
sssd-kcm	2.9.4	
sssd-nfs-idmap	2.9.4	
strace	6.8	6.8
sudo	1.9.15	1.9.15
sysctl-defaults	1.0	1.0
sysstat	12.5.6	12.5.6
systemd	252.23	252.23
systemd-libs	252.23	252.23
systemd-networkd	252.23	252.23
systemd-pam	252.23	252.23
systemd-resolved	252.23	252.23

套件	AMI	VMware OVA
systemd-udev	252.23	252.23
system-release	2023.6.20241031	2023.6.20241031
systemtap-runtime	4.8	4.8
tar	1.34	1.34
tbb	2020.3	2020.3
tcpdump	4.99.1	4.99.1
tcsh	6.24.07	6.24.07
time	1.9	1.9
traceroute	2.1.3	2.1.3
tzdata	2024a	2024a
unzip	6.0	6.0
update-motd	2.2	2.2
userspace-rcu	0.12.1	0.12.1
util-linux	2.37.4	2.37.4
util-linux-core	2.37.4	2.37.4
vim-common	9.0.2153	9.0.2153
vim-data	9.0.2153	9.0.2153
vim-enhanced	9.0.2153	9.0.2153
vim-filesystem	9.0.2153	9.0.2153
vim-minimal	9.0.2153	9.0.2153

套件	AMI	VMware OVA
wget	1.21.3	1.21.3
which	2.21	2.21
words	3.0	3.0
xfsdump	3.1.11	3.1.11
xfsprogs	5.18.0	5.18.0
xmlsec1		1.2.33
xmlsec1-openssl		1.2.33
xxd	9.0.2153	9.0.2153
xxhash-libs	0.8.0	0.8.0
xz	5.2.5	5.2.5
xz-libs	5.2.5	5.2.5
yum	4.14.0	4.14.0
zip	3.0	3.0
zlib	1.2.11	1.2.11
zram-generator	1.1.2	
zram-generator-defaults	1.1.2	
zstd	1.5.5	1.5.5

比較安裝在 Amazon Linux 2023 標準 AMI 上的套件與 AL2023 Hyper-V 映像

AL2023 標準 AMI 上出現RPMs 與 AL2023 Hyper-V 映像上出現RPMs 的比較。

套件	AMI	Hyper-V VHDX
acl	2.3.1	2.3.1
acpid	2.0.32	
alternatives	1.15	1.15
amazon-chrony-config	4.3	
amazon-ec2-net-utils	2.4.1	
amazon-linux-onprem		1.2
amazon-linux-repo-cdn		2023.4.20240319
amazon-linux-repo-s3	2023.4.20240319	
amazon-linux-sb-keys	2023.1	2023.1
amazon-onprem-network		1.2
amazon-rpm-config	228	228
amazon-ssm-agent	3.2.2303.0	3.2.2303.0
at	3.1.23	3.1.23
attr	2.5.1	2.5.1
audit	3.0.6	3.0.6
audit-libs	3.0.6	3.0.6

套件	AMI	Hyper-V VHDX
aws-cfn-bootstrap	2.0	
awscli-2	2.14.5	2.14.5
basesystem	11	11
bash	5.2.15	5.2.15
bash-completion	2.11	2.11
bc	1.07.1	1.07.1
bind-libs	9.16.48	9.16.48
bind-license	9.16.48	9.16.48
bind-utils	9.16.48	9.16.48
binutils	2.39	2.39
boost-filesystem	1.75.0	1.75.0
boost-system	1.75.0	1.75.0
boost-thread	1.75.0	1.75.0
bzip2	1.0.8	1.0.8
bzip2-libs	1.0.8	1.0.8
ca-certificates	2023.2.64	2023.2.64
c-ares	1.19.0	
checkpolicy	3.4	3.4
chkconfig	1.15	1.15
chrony	4.3	4.3

套件	AMI	Hyper-V VHDX
cloud-init	22.2.2	22.2.2
cloud-init-cfg-ec2	22.2.2	
cloud-init-cfg-onprem		22.2.2
cloud-utils-growpart	0.31	0.31
coreutils	8.32	8.32
coreutils-common	8.32	8.32
cpio	2.13	2.13
cracklib	2.9.6	2.9.6
cracklib-dicts	2.9.6	2.9.6
crontabs	1.11	1.11
crypto-policies	20220428	20220428
crypto-policies-scripts	20220428	20220428
cryptsetup	2.6.1	2.6.1
cryptsetup-libs	2.6.1	2.6.1
curl-minimal	8.5.0	8.5.0
cyrus-sasl-lib	2.1.27	2.1.27
cyrus-sasl-plain	2.1.27	2.1.27
dbus	1.12.28	1.12.28
dbus-broker	32	32

套件	AMI	Hyper-V VHDX
dbus-common	1.12.28	1.12.28
dbus-libs	1.12.28	1.12.28
device-mapper	1.02.185	1.02.185
device-mapper-libs	1.02.185	1.02.185
diffutils	3.8	3.8
dnf	4.14.0	4.14.0
dnf-data	4.14.0	4.14.0
dnf-plugin-release-notification	1.2	1.2
dnf-plugins-core	4.3.0	4.3.0
dnf-plugin-support-info	1.2	1.2
dnf-utils	4.3.0	4.3.0
dosfstools	4.2	4.2
dracut	055	055
dracut-config-ec2	3.0	
dracut-config-generic	055	055
dwz	0.14	0.14
dyninst	10.2.1	10.2.1
e2fsprogs	1.46.5	1.46.5

套件	AMI	Hyper-V VHDX
e2fsprogs-libs	1.46.5	1.46.5
ec2-hibinit-agent	1.0.8	
ec2-instance-connect	1.1	
ec2-instance-connect-selinux	1.1	
ec2-utils	2.2.0	
ed	1.14.2	1.14.2
efi-filesystem	5	5
efi-srpm-macros	5	5
efivar	38	38
efivar-libs	38	38
elfutils-debuginfod-client	0.188	0.188
elfutils-default-yama-scope	0.188	0.188
elfutils-libelf	0.188	0.188
elfutils-libs	0.188	0.188
ethtool	5.15	5.15
expat	2.5.0	2.5.0
file	5.39	5.39
file-libs	5.39	5.39

套件	AMI	Hyper-V VHDX
filesystem	3.14	3.14
findutils	4.8.0	4.8.0
fonts-srpm-macros	2.0.5	2.0.5
fstrm	0.6.1	0.6.1
fuse-libs	2.9.9	2.9.9
gawk	5.1.0	5.1.0
gdbm-libs	1.19	1.19
gdisk	1.0.8	1.0.8
gettext	0.21	0.21
gettext-libs	0.21	0.21
ghc-srpm-macros	1.5.0	1.5.0
glib2	2.74.7	2.74.7
glibc	2.34	2.34
glibc-all-langpacks	2.34	2.34
glibc-common	2.34	2.34
glibc-gconv-extra	2.34	2.34
glibc-locale-source	2.34	2.34
gmp	6.2.1	6.2.1
gnupg2-minimal	2.3.7	2.3.7
gnutls	3.8.0	3.8.0

套件	AMI	Hyper-V VHDX
go-srpm-macros	3.2.0	3.2.0
gpgme	1.15.1	1.15.1
gpm-libs	1.20.7	1.20.7
grep	3.8	3.8
groff-base	1.22.4	1.22.4
grub2-common	2.06	2.06
grub2-efi-x64-ec2	2.06	2.06
grub2-pc		2.06
grub2-pc-modules	2.06	2.06
grub2-tools	2.06	2.06
grub2-tools-minimal	2.06	2.06
grubby	8.40	8.40
gssproxy	0.8.4	0.8.4
gzip	1.12	1.12
hostname	3.23	3.23
hunspell	1.7.0	1.7.0
hunspell-en	0.20140811.1	0.20140811.1
hunspell-en-GB	0.20140811.1	0.20140811.1
hunspell-en-US	0.20140811.1	0.20140811.1
hunspell-filesystem	1.7.0	1.7.0

套件	AMI	Hyper-V VHDX
hwdata	0.353	0.353
hyperv-daemons		0
hyperv-daemons-license		0
hypervfcopyd		0
hypervkvpd		0
hyperv-tools		0
hypervvssd		0
info	6.7	6.7
inih	49	49
initscripts	10.09	10.09
iproute	5.10.0	5.10.0
iputils	20210202	20210202
irqbalance	1.9.0	1.9.0
jansson	2.14	2.14
jitterentropy	3.4.1	3.4.1
jq	1.7.1	1.7.1
json-c	0.14	0.14
kbd	2.4.0	2.4.0
kbd-misc	2.4.0	2.4.0
kernel	6.1.79	6.1.79

套件	AMI	Hyper-V VHDX
kernel-livepatch-repo-cdn		2023.4.20240319
kernel-livepatch-repo-s3	2023.4.20240319	
kernel-modules-extra		6.1.79
kernel-modules-extra-common		6.1.79
kernel-srpm-macros	1.0	1.0
kernel-tools	6.1.79	6.1.79
keyutils	1.6.3	1.6.3
keyutils-libs	1.6.3	1.6.3
kmod	29	29
kmod-libs	29	29
kpatch-runtime	0.9.7	0.9.7
krb5-libs	1.21	1.21
less	608	608
libacl	2.3.1	2.3.1
libaio	0.3.111	0.3.111
libarchive	3.5.3	3.5.3
libargon2	20171227	20171227
libassuan	2.5.5	2.5.5

套件	AMI	Hyper-V VHDX
libattr	2.5.1	2.5.1
libbasicobjects	0.1.1	0.1.1
libblkid	2.37.4	2.37.4
libcap	2.48	2.48
libcap-ng	0.8.2	0.8.2
libcbor	0.7.0	0.7.0
libcollection	0.7.0	0.7.0
libcom_err	1.46.5	1.46.5
libcomps	0.1.20	0.1.20
libconfig	1.7.2	1.7.2
libcurl-minimal	8.5.0	8.5.0
libdb	5.3.28	5.3.28
libdhash	0.5.0	
libdnf	0.69.0	0.69.0
libeconf	0.4.0	0.4.0
libedit	3.1	3.1
libev	4.33	4.33
libevent	2.1.12	2.1.12
libfdisk	2.37.4	2.37.4
libffi	3.4.4	3.4.4

套件	AMI	Hyper-V VHDX
libfido2	1.10.0	1.10.0
libgcc	11.4.1	11.4.1
libgcrypt	1.10.2	1.10.2
libgomp	11.4.1	11.4.1
libgpg-error	1.42	1.42
libibverbs	48.0	48.0
libidn2	2.3.2	2.3.2
libini_config	1.3.1	1.3.1
libkcapi	1.4.0	1.4.0
libkcapi-hmaccalc	1.4.0	1.4.0
libldb	2.6.2	
libmaxminddb	1.5.2	1.5.2
libmetalink	0.1.3	0.1.3
libmnl	1.0.4	1.0.4
libmodulemd	2.13.0	2.13.0
libmount	2.37.4	2.37.4
libnfsidmap	2.5.4	2.5.4
libnghttp2	1.57.0	1.57.0
libnl3	3.5.0	3.5.0
libpath_utils	0.2.1	0.2.1

套件	AMI	Hyper-V VHDX
libpcap	1.10.1	1.10.1
libpipeline	1.5.3	1.5.3
libpkgconf	1.8.0	1.8.0
libpsl	0.21.1	0.21.1
libpwquality	1.4.4	1.4.4
libref_array	0.1.5	0.1.5
librepo	1.14.5	1.14.5
libreport-filesystem	2.15.2	2.15.2
libseccomp	2.5.3	2.5.3
libselinux	3.4	3.4
libselinux-utils	3.4	3.4
libsemanage	3.4	3.4
libsepol	3.4	3.4
libsigsegv	2.13	2.13
libsmartcols	2.37.4	2.37.4
libsolv	0.7.22	0.7.22
libss	1.46.5	1.46.5
libsss_certmap	2.9.4	
libsss_idmap	2.9.4	2.9.4
libsss_nss_idmap	2.9.4	2.9.4

套件	AMI	Hyper-V VHDX
libsss_sudo	2.9.4	
libstdc++	11.4.1	11.4.1
libstoragemgmt	1.9.4	1.9.4
libtalloc	2.3.4	
libtasn1	4.19.0	4.19.0
libtdb	1.4.7	
libtevent	0.13.0	
libtextstyle	0.21	0.21
libtirpc	1.3.3	1.3.3
libunistring	0.9.10	0.9.10
libuser	0.63	0.63
libutempter	1.2.1	1.2.1
libuuid	2.37.4	2.37.4
libuv	1.47.0	1.47.0
libverto	0.3.2	0.3.2
libverto-libev	0.3.2	0.3.2
libxcrypt	4.4.33	4.4.33
libxml2	2.10.4	2.10.4
libyaml	0.2.5	0.2.5
libzstd	1.5.5	1.5.5

套件	AMI	Hyper-V VHDX
lm_sensors-libs	3.6.0	3.6.0
lmdb-libs	0.9.29	0.9.29
logrotate	3.20.1	3.20.1
lsof	4.94.0	4.94.0
lua-libs	5.4.4	5.4.4
lua-srpm-macros	1	1
lz4-libs	1.9.4	1.9.4
man-db	2.9.3	2.9.3
man-pages	5.10	5.10
microcode_ctl	2.1	2.1
mpfr	4.1.0	4.1.0
nano	5.8	5.8
ncurses	6.2	6.2
ncurses-base	6.2	6.2
ncurses-libs	6.2	6.2
nettle	3.8	3.8
net-tools	2.0	2.0
newt	0.52.21	0.52.21
nfs-utils	2.5.4	2.5.4
npth	1.6	1.6

套件	AMI	Hyper-V VHDX
nspr	4.35.0	4.35.0
nss	3.90.0	3.90.0
nss-softokn	3.90.0	3.90.0
nss-softokn-freebl	3.90.0	3.90.0
nss-sysinit	3.90.0	3.90.0
nss-util	3.90.0	3.90.0
ntsysv	1.15	1.15
numactl-libs	2.0.14	2.0.14
ocaml-srpm-macros	6	6
oniguruma	6.9.7.1	6.9.7.1
openblas-srpm-macros	2	2
openldap	2.4.57	2.4.57
openssh	8.7p1	8.7p1
openssh-clients	8.7p1	8.7p1
openssh-server	8.7p1	8.7p1
openssl	3.0.8	3.0.8
openssl-libs	3.0.8	3.0.8
openssl-pkcs11	0.4.12	0.4.12
os-prober	1.77	1.77
p11-kit	0.24.1	0.24.1

套件	AMI	Hyper-V VHDX
p11-kit-trust	0.24.1	0.24.1
package-notes-srpm-macros	0.4	0.4
pam	1.5.1	1.5.1
parted	3.4	3.4
passwd	0.80	0.80
pciutils	3.7.0	3.7.0
pciutils-libs	3.7.0	3.7.0
pcre2	10.40	10.40
pcre2-syntax	10.40	10.40
perl-Carp	1.50	1.50
perl-Class-Struct	0.66	0.66
perl-constant	1.33	1.33
perl-DynaLoader	1.47	1.47
perl-Encode	3.15	3.15
perl-Errno	1.30	1.30
perl-Exporter	5.74	5.74
perl-Fcntl	1.13	1.13
perl-File-Basename	2.85	2.85
perl-File-Path	2.18	2.18
perl-File-stat	1.09	1.09

套件	AMI	Hyper-V VHDX
perl-File-Temp	0.231.100	0.231.100
perl-Getopt-Long	2.52	2.52
perl-Getopt-Std	1.12	1.12
perl-HTTP-Tiny	0.078	0.078
perl-if	0.60.800	0.60.800
perl-interpreter	5.32.1	5.32.1
perl-IO	1.43	1.43
perl-IPC-Open3	1.21	1.21
perl-libs	5.32.1	5.32.1
perl-MIME-Base64	3.16	3.16
perl-mro	1.23	1.23
perl-overload	1.31	1.31
perl-overloading	0.02	0.02
perl-parent	0.238	0.238
perl-PathTools	3.78	3.78
perl-Pod-Escapes	1.07	1.07
perl-podlators	4.14	4.14
perl-Pod-Perldoc	3.28.01	3.28.01
perl-Pod-Simple	3.42	3.42
perl-Pod-Usage	2.01	2.01

套件	AMI	Hyper-V VHDX
perl-POSIX	1.94	1.94
perl-Scalar-List-Utils	1.56	1.56
perl-SelectSaver	1.02	1.02
perl-Socket	2.032	2.032
perl-srpm-macros	1	1
perl-Storable	3.21	3.21
perl-subs	1.03	1.03
perl-Symbol	1.08	1.08
perl-Term-ANSIColor	5.01	5.01
perl-Term-Cap	1.17	1.17
perl-Text-ParseWords	3.30	3.30
perl-Text-Tabs+Wrap	2021.0726	2021.0726
perl-Time-Local	1.300	1.300
perl-vars	1.05	1.05
pkgconf	1.8.0	1.8.0
pkgconf-m4	1.8.0	1.8.0
pkgconf-pkg-config	1.8.0	1.8.0
policycoreutils	3.4	3.4
policycoreutils-python-utils	3.4	

套件	AMI	Hyper-V VHDX
popt	1.18	1.18
procps-ng	3.3.17	3.3.17
protobuf-c	1.4.1	1.4.1
psacct	6.6.4	6.6.4
psmisc	23.4	23.4
publicsuffix-list-dafsa	20240212	20240212
python3	3.9.16	3.9.16
python3-attrs	20.3.0	20.3.0
python3-audit	3.0.6	3.0.6
python3-awscrt	2019 年 19 月 0 日	2019 年 19 月 0 日
python3-babel	2.9.1	2.9.1
python3-cffi	1.14.5	1.14.5
python3-chardet	4.0.0	4.0.0
python3-colorama	0.4.4	0.4.4
python3-configobj	5.0.6	5.0.6
python3-cryptography	36.0.1	36.0.1
python3-daemon	2.3.0	
python3-dateutil	2.8.1	2.8.1
python3-dbus	1.2.18	1.2.18
python3-distro	1.5.0	1.5.0

套件	AMI	Hyper-V VHDX
python3-dnf	4.14.0	4.14.0
python3-dnf-plugins-core	4.3.0	4.3.0
python3-docutils	0.16	0.16
python3-gpg	1.15.1	1.15.1
python3-hawkey	0.69.0	0.69.0
python3-idna	2.10	2.10
python3-jinja2	2.11.3	2.11.3
python3-jmespath	0.10.0	0.10.0
python3-jsonpatch	1.21	1.21
python3-jsonpointer	2.0	2.0
python3-jsonschema	3.2.0	3.2.0
python3-libcomps	0.1.20	0.1.20
python3-libdnf	0.69.0	0.69.0
python3-libs	3.9.16	3.9.16
python3-libselinux	3.4	3.4
python3-libsemanage	3.4	3.4
python3-libstorage mgmt	1.9.4	1.9.4
python3-lockfile	0.12.2	
python3-markupsafe	1.1.1	1.1.1

套件	AMI	Hyper-V VHDX
python3-netifaces	0.10.6	0.10.6
python3-oauthlib	3.0.2	3.0.2
python3-pip-wheel	21.3.1	21.3.1
python3-ply	3.11	3.11
python3-policycoreutils	3.4	3.4
python3-prettytable	0.7.2	0.7.2
python3-prompt-toolkit	3.0.24	3.0.24
python3-pycparser	2.20	2.20
python3-pyrsistent	0.17.3	0.17.3
python3-pyserial	3.4	3.4
python3-pysocks	1.7.1	1.7.1
python3-pytz	2022.7.1	2022.7.1
python3-pyyaml	5.4.1	5.4.1
python3-requests	2.25.1	2.25.1
python3-rpm	4.16.1.3	4.16.1.3
python3-ruamel-yaml	0.16.6	0.16.6
python3-ruamel-yaml-clib	0.1.2	0.1.2
python3-setools	4.4.1	4.4.1

套件	AMI	Hyper-V VHDX
python3-setuptools	59.6.0	59.6.0
python3-setuptools-wheel	59.6.0	59.6.0
python3-six	1.15.0	1.15.0
python3-systemd	235	235
python3-urllib3	1.25.10	1.25.10
python3-wcwidth	0.2.5	0.2.5
python-chevron	0.13.1	
python-srpm-macros	3.9	3.9
quota	4.06	4.06
quota-nls	4.06	4.06
readline	8.1	8.1
rng-tools	6.14	6.14
rootfiles	8.1	8.1
rpcbind	1.2.6	1.2.6
rpm	4.16.1.3	4.16.1.3
rpm-build-libs	4.16.1.3	4.16.1.3
rpm-libs	4.16.1.3	4.16.1.3
rpm-plugin-selinux	4.16.1.3	4.16.1.3
rpm-plugin-systemd-inhibit	4.16.1.3	4.16.1.3

套件	AMI	Hyper-V VHDX
rpm-sign-libs	4.16.1.3	4.16.1.3
rsync	3.2.6	3.2.6
rust-srpm-macros	21	21
sbsigntools	0.9.4	0.9.4
screen	4.8.0	4.8.0
sed	4.8	4.8
selinux-policy	37.22	37.22
selinux-policy-targeted	37.22	37.22
setup	2.13.7	2.13.7
shadow-utils	4.9	4.9
slang	2.3.2	2.3.2
sqlite-libs	3.40.0	3.40.0
sssd-client	2.9.4	2.9.4
sssd-common	2.9.4	
sssd-kcm	2.9.4	
sssd-nfs-idmap	2.9.4	
strace	5.16	5.16
sudo	1.9.14	1.9.14
sysctl-defaults	1.0	1.0
sysstat	12.5.6	12.5.6

套件	AMI	Hyper-V VHDX
systemd	252.16	252.16
systemd-libs	252.16	252.16
systemd-networkd	252.16	252.16
systemd-pam	252.16	252.16
systemd-resolved	252.16	252.16
systemd-udev	252.16	252.16
system-release	2023.4.20240319	2023.4.20240319
systemtap-runtime	4.8	4.8
tar	1.34	1.34
tbb	2020.3	2020.3
tcpdump	4.99.1	4.99.1
tcsh	6.24.07	6.24.07
time	1.9	1.9
traceroute	2.1.3	2.1.3
tzdata	2024a	2024a
unzip	6.0	6.0
update-motd	2.2	2.2
userspace-rcu	0.12.1	0.12.1
util-linux	2.37.4	2.37.4
util-linux-core	2.37.4	2.37.4

套件	AMI	Hyper-V VHDX
vim-common	9.0.2153	9.0.2153
vim-data	9.0.2153	9.0.2153
vim-enhanced	9.0.2153	9.0.2153
vim-filesystem	9.0.2153	9.0.2153
vim-minimal	9.0.2153	9.0.2153
wget	1.21.3	1.21.3
which	2.21	2.21
words	3.0	3.0
xfsdump	3.1.11	3.1.11
xfsprogs	5.18.0	5.18.0
xxd	9.0.2153	9.0.2153
xxhash-libs	0.8.0	0.8.0
xz	5.2.5	5.2.5
xz-libs	5.2.5	5.2.5
yum	4.14.0	4.14.0
zip	3.0	3.0
zlib	1.2.11	1.2.11
zram-generator	1.1.2	
zram-generator-defaults	1.1.2	
zstd	1.5.5	1.5.5

識別 Amazon Linux 執行個體和版本

能夠判斷 Linux 發行版本，以及作業系統映像或執行個體的發行版本非常重要。Amazon Linux 提供機制來識別與其他 Linux 發行版本不同的 Amazon Linux，以及識別映像的 Amazon Linux 發行版本。

本節將涵蓋可使用的不同方法、其限制，並介紹其使用的一些範例。

主題

使用 `os-release`標準

Amazon Linux 符合識別 Linux 發行版本`os-release`的標準。此檔案提供有關作業系統識別和版本資訊的機器可讀取資訊。

> **Note**
>
> 標準 會指定`/etc/os-release`先嘗試剖析，接著是 `/usr/lib/os-release`。應注意遵循有關檔案名稱和路徑的標準。

主題

金鑰識別差異

`os-release` 位於 `/etc/os-release`，如果不存在，則位於 `/usr/lib/os-release`。如需完整資訊，請參閱 `os-release`標準。

判斷執行個體是否執行 Amazon Linux 最可靠的方法是檢查 中的 `ID` 欄位`os-release`。

判斷版本差異最可靠的方法是檢查 中的 VERSION_ID 欄位os-release：

- Amazon Linux AMI：VERSION_ID包含以日期為基礎的版本 （ 例如 2018.03)
- AL2： VERSION_ID="2"
- AL2023： VERSION_ID="2023"

> **Note**
>
> 請記住， VERSION_ID 是機器可讀取的欄位，適用於程式設計用途，而 PRETTY_NAME 旨在向使用者顯示。如需欄位類型的詳細資訊the section called “欄位類型”，請參閱 。

欄位類型：機器可讀與人類可讀

/etc/os-release 檔案 (/usr/lib/os-release如果/etc/os-release不存在） 包含兩種類型的欄位：用於程式設計用途的機器可讀欄位，以及用於向使用者呈現的人類可讀欄位。

機器可讀取的欄位

這些欄位使用標準化格式，旨在供指令碼、套件管理員和其他自動化工具處理。它們只包含小寫字母、數字和有限的標點符號 （ 句點、底線和連字號 ）。

- ID – 作業系統識別符。Amazon Linux amzn在所有版本中使用 ，將其與 Debian (debian)、Ubuntu (ubuntu) 或 Fedora (fedora) 等其他發行版本區區區分開來
- VERSION_ID – 用於程式設計用途的作業系統版本 （ 例如 2023)
- ID_LIKE – 以空格分隔的相關分佈清單 （ 例如 fedora)
- VERSION_CODENAME – 指令碼的版本程式碼名稱 （ 例如 karoo)
- VARIANT_ID – 程式設計決策的變體識別符
- BUILD_ID – 建置系統映像的識別符
- IMAGE_ID – 容器化環境的影像識別符
- PLATFORM_ID – 平台識別符 （ 例如 platform:al2023)

人類可讀取的欄位

這些欄位適用於向使用者顯示，可能包含空格、混合大小寫和描述性文字。在使用者介面中呈現作業系統資訊時，應使用它們。

- NAME – 顯示的作業系統名稱 （例如 Amazon Linux)
- PRETTY_NAME – 顯示版本的完整作業系統名稱 （例如 Amazon Linux 2023.8.20250721)
- VERSION – 適用於使用者簡報的版本資訊
- VARIANT – 用於顯示的變體或版本名稱 （例如 Server Edition)

其他資訊欄位

這些欄位提供有關作業系統的其他中繼資料：

- HOME_URL – 專案首頁 URL
- DOCUMENTATION_URL – 文件 URL
- SUPPORT_URL – 支援資訊 URL
- BUG_REPORT_URL – 錯誤報告 URL
- VENDOR_NAME – 供應商名稱
- VENDOR_URL – 供應商 URL
- SUPPORT_END – End-of-support，格式為 YYYY-MM-DD
- CPE_NAME – 通用平台列舉識別符
- ANSI_COLOR – 終端機顯示的 ANSI 顏色代碼

編寫需要以程式設計方式識別 Amazon Linux 的指令碼或應用程式時，請使用機器可讀取的欄位，例如 ID和 VERSION_ID。向使用者顯示作業系統資訊時，請使用人類可讀取的欄位，例如 PRETTY_NAME。

/etc/os-release 範例

/etc/os-release 檔案內容因 Amazon Linux 版本而異：

AL2023

```
[ec2-user ~]$ cat /etc/os-release
```

```
NAME="Amazon Linux"
VERSION="2023"
ID="amzn"
ID_LIKE="fedora"
```

```
VERSION_ID="2023"
PLATFORM_ID="platform:al2023"
PRETTY_NAME="Amazon Linux 2023.8.20250721"
ANSI_COLOR="0;33"
CPE_NAME="cpe:2.3:o:amazon:amazon_linux:2023"
HOME_URL="https://aws.amazon.com/linux/amazon-linux-2023/"
DOCUMENTATION_URL="https://docs.aws.amazon.com/linux/"
SUPPORT_URL="https://aws.amazon.com/premiumsupport/"
BUG_REPORT_URL="https://github.com/amazonlinux/amazon-linux-2023"
VENDOR_NAME="AWS"
VENDOR_URL="https://aws.amazon.com/"
SUPPORT_END="2029-06-30"
```

AL2

```
[ec2-user ~]$ cat /etc/os-release
```

```
NAME="Amazon Linux"
VERSION="2"
ID="amzn"
ID_LIKE="centos rhel fedora"
VERSION_ID="2"
PRETTY_NAME="Amazon Linux 2"
ANSI_COLOR="0;33"
CPE_NAME="cpe:2.3:o:amazon:amazon_linux:2"
HOME_URL="https://amazonlinux.com/"
SUPPORT_END="2026-06-30"
```

Amazon Linux AMI

```
[ec2-user ~]$ cat /etc/os-release
```

```
NAME="Amazon Linux AMI"
VERSION="2018.03"
ID="amzn"
ID_LIKE="rhel fedora"
VERSION_ID="2018.03"
PRETTY_NAME="Amazon Linux AMI 2018.03"
ANSI_COLOR="0;33"
CPE_NAME="cpe:/o:amazon:linux:2018.03:ga"
HOME_URL="http://aws.amazon.com/amazon-linux-ami/"
```

與其他分佈的比較

若要了解 Amazon Linux 如何符合更廣泛的 Linux 生態系統，請將其/etc/os-release格式與其他主要發行版本進行比較：

Fedora

```
[ec2-user ~]$ cat /etc/os-release
```

```
NAME="Fedora Linux"
VERSION="42 (Container Image)"
RELEASE_TYPE=stable
ID=fedora
VERSION_ID=42
VERSION_CODENAME=""
PLATFORM_ID="platform:f42"
PRETTY_NAME="Fedora Linux 42 (Container Image)"
ANSI_COLOR="0;38;2;60;110;180"
LOGO=fedora-logo-icon
CPE_NAME="cpe:/o:fedoraproject:fedora:42"
DEFAULT_HOSTNAME="fedora"
HOME_URL="https://fedoraproject.org/"
DOCUMENTATION_URL="https://docs.fedoraproject.org/en-US/fedora/f42/system-
administrators-guide/"
SUPPORT_URL="https://ask.fedoraproject.org/"
BUG_REPORT_URL="https://bugzilla.redhat.com/"
REDHAT_BUGZILLA_PRODUCT="Fedora"
REDHAT_BUGZILLA_PRODUCT_VERSION=42
REDHAT_SUPPORT_PRODUCT="Fedora"
REDHAT_SUPPORT_PRODUCT_VERSION=42
SUPPORT_END=2026-05-13
VARIANT="Container Image"
VARIANT_ID=container
```

Debian

```
[ec2-user ~]$ cat /etc/os-release
```

```
PRETTY_NAME="Debian GNU/Linux 12 (bookworm)"
NAME="Debian GNU/Linux"
VERSION_ID="12"
```

```
VERSION="12 (bookworm)"
VERSION_CODENAME=bookworm
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
```

Ubuntu

```
[ec2-user ~]$ cat /etc/os-release
```

```
PRETTY_NAME="Ubuntu 24.04.2 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04.2 LTS (Noble Numbat)"
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=noble
LOGO=ubuntu-logo
```

請注意機器可讀取欄位如何跨分佈提供一致的識別：

- ID – 唯一識別作業系統：amzn適用於 Amazon Linux、fedora適用於 Fedora、debian適用於 Debian、ubuntu適用於 Ubuntu
- ID_LIKE – 顯示分佈關係：Amazon Linux 使用 fedora(AL2023) 或 centos rhel fedora(AL2)，而 Ubuntu 顯示 debian 表示其 Debian 傳統
- VERSION_ID – 提供機器可剖析的版本資訊：2023適用於 AL2023、42適用於 Fedora、12適用於 Debian、24.04適用於 Ubuntu

相反地，人類可讀欄位旨在向使用者顯示：

- NAME – 易於使用的作業系統名稱：Amazon Linux、Fedora Linux、Debian GNU/Linux、Ubuntu

- PRETTY_NAME – 完整顯示名稱，版本為：Amazon Linux 2023.8.20250721、Fedora Linux 42 (Container Image)、Debian GNU/Linux 12 (bookworm)、 Ubuntu 24.04.2 LTS
- VERSION – 人類可讀取的版本，具有程式碼名稱或發行類型等其他內容

撰寫跨平台指令碼時，請一律使用機器可讀欄位 (ID、VERSION_ID、ID_LIKE) 進行邏輯和決策，並僅使用人工可讀欄位 (PRETTY_NAME、NAME) 向使用者顯示資訊。

Amazon Linux 特定

有些檔案是 Amazon Linux 特有的，可用於識別 Amazon Linux 及其版本。新程式碼應使用 /etc/os-release標準，以便與跨分佈相容。不鼓勵使用任何 Amazon Linux 特定檔案。

主題

- /etc/system-release 檔案
- 影像識別檔案
- Amazon Linux 特定檔案的範例

/etc/system-release 檔案

Amazon Linux 包含 /etc/system-release 檔案，指定安裝的目前版本。此檔案使用套件管理員更新，Amazon Linux 是system-release套件的一部分。雖然 Fedora 等其他一些分佈也有此檔案，但它不存在於 Ubuntu 等以 Debian 為基礎的分佈中。

> **Note**
>
> /etc/system-release 檔案包含人類可讀取的字串，不應以程式設計方式用來識別作業系統或版本。請改用 /etc/os-release（或/usr/lib/os-release如果 /etc/os-release 不存在）中的機器可讀取欄位。

Amazon Linux 也包含機器可讀取的 版本/etc/system-release，其遵循 /etc/system-release-cpe 檔案中的通用平台列舉 (CPE) 規格。

影像識別檔案

每個 Amazon Linux 映像都包含一個唯一的/etc/image-id檔案，提供 Amazon Linux 團隊所產生之原始映像的其他資訊。此檔案專屬於 Amazon Linux，在其他 Linux 發行版本中找不到，例如 Debian、Ubuntu 或 Fedora。該檔案包含關於映像的下列資訊：

- image_name、image_version、 image_arch – 用於建構映像之建置配方的值。
- image_stamp – 在建立映像時產生的唯一隨機十六進位值。
- image_date – 建立映像時的 UTC 時間，格式為 YYYYMMDDhhmmss。
- recipe_name， recipe_id – 用於建構映像的建置配方的名稱和 ID。

Amazon Linux 特定檔案的範例

下列各節提供每個 Amazon Linux 主要版本之 Amazon Linux 特定識別檔案的範例。

> **Note**
>
> 在任何實際程式碼中，如果/etc/os-release檔案不存在，則/usr/lib/os-release應該使用 。

AL2023

下列範例顯示 AL2023 的識別檔案。

/etc/image-id 適用於 AL2023 的 範例：

```
[ec2-user ~]$ cat /etc/image-id
```

```
image_name="al2023-container"
image_version="2023"
image_arch="x86_64"
image_file="al2023-container-2023.8.20250721.2-x86_64"
image_stamp="822b-1a9e"
image_date="20250719211531"
recipe_name="al2023 container"
recipe_id="89b25f7b-be82-2215-a8eb-6e63-0830-94ea-658d41c4"
```

/etc/system-release 適用於 AL2023 的 範例：

```
[ec2-user ~]$ cat /etc/system-release
```

```
Amazon Linux release 2023.8.20250721 (Amazon Linux)
```

AL2

下列範例顯示 AL2 的識別檔案。

`/etc/image-id` 適用於 AL2 的 範例：

```
[ec2-user ~]$ cat /etc/image-id
```

```
image_name="amzn2-container-raw"
image_version="2"
image_arch="x86_64"
image_file="amzn2-container-raw-2.0.20250721.2-x86_64"
image_stamp="4126-16ad"
image_date="20250721225801"
recipe_name="amzn2 container"
recipe_id="948422df-a4e6-5fc8-ba89-ef2e-0e1f-e1bb-16f84087"
```

`/etc/system-release` 適用於 AL2 的 範例：

```
[ec2-user ~]$ cat /etc/system-release
```

```
Amazon Linux release 2 (Karoo)
```

Amazon Linux AMI

下列範例顯示 Amazon Linux AMI 的識別檔案。

適用於 Amazon Linux AMI `/etc/image-id`的 範例：

```
[ec2-user ~]$ cat /etc/image-id
```

```
image_name="amzn-container-minimal"
image_version="2018.03"
image_arch="x86_64"
image_file="amzn-container-minimal-2018.03.0.20231218.0-x86_64"
image_stamp="407d-5ef3"
```

```
image_date="20231218203210"
recipe_name="amzn container"
recipe_id="b1e7635e-14e3-dd57-b1ab-7351-edd0-d9e0-ca6852ea"
```

適用於 Amazon Linux AMI /etc/system-release的 範例：

```
[ec2-user ~]$ cat /etc/system-release
```

```
Amazon Linux AMI release 2018.03
```

作業系統偵測的範例程式碼

下列範例示範如何使用 /etc/os-release（或 /usr/lib/os-release /etc/os-release不存在）檔案，以程式設計方式偵測作業系統和版本。這些範例示範如何區分 Amazon Linux 和其他發行版本，以及如何使用 ID_LIKE 欄位來判斷發行版本系列。

以下指令碼以數種不同的程式設計語言實作，每個實作都會產生相同的輸出。

Shell

```
#!/bin/bash

# Function to get a specific field from os-release file
get_os_release_field() {
    local field="$1"
    local os_release_file

    # Find the os-release file
    if [ -f /etc/os-release ]; then
        os_release_file='/etc/os-release'
    elif [ -f /usr/lib/os-release ]; then
        os_release_file='/usr/lib/os-release'
    else
        echo "Error: os-release file not found" >&2
        return 1
    fi

    # Source the file in a subshell and return the requested field.
    #
    # A subshell means that variables from os-release are only available
    # within the subshell, and the main script environment remains clean.
```

```
    (
        . "$os_release_file"
        eval "echo \"\$$field\""
    )
}

is_amazon_linux() {
    [ "$(get_os_release_field ID)" = "amzn" ]
}

is_fedora() {
    [ "$(get_os_release_field ID)" = "fedora" ]
}

is_ubuntu() {
    [ "$(get_os_release_field ID)" = "ubuntu" ]
}

is_debian() {
    [ "$(get_os_release_field ID)" = "debian" ]
}

# Function to check if this is like Fedora (includes Amazon Linux, CentOS, RHEL,
 etc.)
is_like_fedora() {
    local id="$(get_os_release_field ID)"
    local id_like="$(get_os_release_field ID_LIKE)"
    [ "$id" = "fedora" ] || [[ "$id_like" == *"fedora"* ]]
}

# Function to check if this is like Debian (includes Ubuntu and derivatives)
is_like_debian() {
    local id="$(get_os_release_field ID)"
    local id_like="$(get_os_release_field ID_LIKE)"
    [ "$id" = "debian" ] || [[ "$id_like" == *"debian"* ]]
}

# Get the main fields we'll use multiple times
ID="$(get_os_release_field ID)"
VERSION_ID="$(get_os_release_field VERSION_ID)"
PRETTY_NAME="$(get_os_release_field PRETTY_NAME)"
ID_LIKE="$(get_os_release_field ID_LIKE)"

echo "Operating System Detection Results:"
```

```
echo "================================="
echo "Is Amazon Linux: $(is_amazon_linux && echo YES || echo NO)"
echo "Is Fedora: $(is_fedora && echo YES || echo NO)"
echo "Is Ubuntu: $(is_ubuntu && echo YES || echo NO)"
echo "Is Debian: $(is_debian && echo YES || echo NO)"
echo "Is like Fedora: $(is_like_fedora && echo YES || echo NO)"
echo "Is like Debian: $(is_like_debian && echo YES || echo NO)"
echo
echo "Detailed OS Information:"
echo "======================="
echo "ID: $ID"
echo "VERSION_ID: $VERSION_ID"
echo "PRETTY_NAME: $PRETTY_NAME"
[ -n "$ID_LIKE" ] && echo "ID_LIKE: $ID_LIKE"

# Amazon Linux specific information
if is_amazon_linux; then
    echo ""
    echo "Amazon Linux Version Details:"
    echo "============================"
    case "$VERSION_ID" in
        2018.03)
            echo "Amazon Linux AMI (version 1)"
            ;;
        2)
            echo "Amazon Linux 2"
            ;;
        2023)
            echo "Amazon Linux 2023"
            ;;
        *)
            echo "Unknown Amazon Linux version: $VERSION_ID"
            ;;
    esac

    # Check for Amazon Linux specific files
    [ -f /etc/image-id ] && echo "Amazon Linux image-id file present"
fi
```

Python 3.7-3.9

```
#!/usr/bin/env python3
```

```
import os
import sys

def parse_os_release():
    """Parse the os-release file and return a dictionary of key-value pairs."""
    os_release_data = {}

    # Try /etc/os-release first, then /usr/lib/os-release
    for path in ['/etc/os-release', '/usr/lib/os-release']:
        if os.path.exists(path):
            try:
                with open(path, 'r') as f:
                    for line in f:
                        line = line.strip()
                        if line and not line.startswith('#') and '=' in line:
                            key, value = line.split('=', 1)
                            # Remove quotes if present
                            value = value.strip('"\'')
                            os_release_data[key] = value
                return os_release_data
            except IOError:
                continue

    print("Error: os-release file not found")
    sys.exit(1)

def is_amazon_linux(os_data):
    """Check if this is Amazon Linux."""
    return os_data.get('ID') == 'amzn'

def is_fedora(os_data):
    """Check if this is Fedora."""
    return os_data.get('ID') == 'fedora'

def is_ubuntu(os_data):
    """Check if this is Ubuntu."""
    return os_data.get('ID') == 'ubuntu'

def is_debian(os_data):
    """Check if this is Debian."""
    return os_data.get('ID') == 'debian'

def is_like_fedora(os_data):
    """Check if this is like Fedora (includes Amazon Linux, CentOS, RHEL, etc.)."""
```

```
    if os_data.get('ID') == 'fedora':
        return True
    id_like = os_data.get('ID_LIKE', '')
    return 'fedora' in id_like

def is_like_debian(os_data):
    """Check if this is like Debian (includes Ubuntu and derivatives)."""
    if os_data.get('ID') == 'debian':
        return True
    id_like = os_data.get('ID_LIKE', '')
    return 'debian' in id_like

def main():
    # Parse os-release file
    os_data = parse_os_release()

    # Display results
    print("Operating System Detection Results:")
    print("==================================")
    print(f"Is Amazon Linux: {'YES' if is_amazon_linux(os_data) else 'NO'}")
    print(f"Is Fedora: {'YES' if is_fedora(os_data) else 'NO'}")
    print(f"Is Ubuntu: {'YES' if is_ubuntu(os_data) else 'NO'}")
    print(f"Is Debian: {'YES' if is_debian(os_data) else 'NO'}")
    print(f"Is like Fedora: {'YES' if is_like_fedora(os_data) else 'NO'}")
    print(f"Is like Debian: {'YES' if is_like_debian(os_data) else 'NO'}")

    # Additional information
    print()
    print("Detailed OS Information:")
    print("======================")
    print(f"ID: {os_data.get('ID', '')}")
    print(f"VERSION_ID: {os_data.get('VERSION_ID', '')}")
    print(f"PRETTY_NAME: {os_data.get('PRETTY_NAME', '')}")
    if os_data.get('ID_LIKE'):
        print(f"ID_LIKE: {os_data.get('ID_LIKE')}")

    # Amazon Linux specific information
    if is_amazon_linux(os_data):
        print()
        print("Amazon Linux Version Details:")
        print("============================")
        version_id = os_data.get('VERSION_ID', '')
        if version_id == '2018.03':
            print("Amazon Linux AMI (version 1)")
```

```
        elif version_id == '2':
            print("Amazon Linux 2")
        elif version_id == '2023':
            print("Amazon Linux 2023")
        else:
            print(f"Unknown Amazon Linux version: {version_id}")

        # Check for Amazon Linux specific files
        if os.path.exists('/etc/image-id'):
            print("Amazon Linux image-id file present")

if __name__ == '__main__':
    main()
```

Python 3.10+

```
#!/usr/bin/env python3

import os
import sys
import platform

def is_amazon_linux(os_data):
    """Check if this is Amazon Linux."""
    return os_data.get('ID') == 'amzn'

def is_fedora(os_data):
    """Check if this is Fedora."""
    return os_data.get('ID') == 'fedora'

def is_ubuntu(os_data):
    """Check if this is Ubuntu."""
    return os_data.get('ID') == 'ubuntu'

def is_debian(os_data):
    """Check if this is Debian."""
    return os_data.get('ID') == 'debian'

def is_like_fedora(os_data):
    """Check if this is like Fedora (includes Amazon Linux, CentOS, RHEL, etc.)."""
    if os_data.get('ID') == 'fedora':
        return True
    id_like = os_data.get('ID_LIKE', '')
```

```
    return 'fedora' in id_like

def is_like_debian(os_data):
    """Check if this is like Debian (includes Ubuntu and derivatives)."""
    if os_data.get('ID') == 'debian':
        return True
    id_like = os_data.get('ID_LIKE', '')
    return 'debian' in id_like

def main():
    # Parse os-release file using the standard library function (Python 3.10+)
    try:
        os_data = platform.freedesktop_os_release()
    except OSError:
        print("Error: os-release file not found")
        sys.exit(1)

    # Display results
    print("Operating System Detection Results:")
    print("==================================")
    print(f"Is Amazon Linux: {'YES' if is_amazon_linux(os_data) else 'NO'}")
    print(f"Is Fedora: {'YES' if is_fedora(os_data) else 'NO'}")
    print(f"Is Ubuntu: {'YES' if is_ubuntu(os_data) else 'NO'}")
    print(f"Is Debian: {'YES' if is_debian(os_data) else 'NO'}")
    print(f"Is like Fedora: {'YES' if is_like_fedora(os_data) else 'NO'}")
    print(f"Is like Debian: {'YES' if is_like_debian(os_data) else 'NO'}")

    # Additional information
    print()
    print("Detailed OS Information:")
    print("========================")
    print(f"ID: {os_data.get('ID', '')}")
    print(f"VERSION_ID: {os_data.get('VERSION_ID', '')}")
    print(f"PRETTY_NAME: {os_data.get('PRETTY_NAME', '')}")
    if os_data.get('ID_LIKE'):
        print(f"ID_LIKE: {os_data.get('ID_LIKE')}")

    # Amazon Linux specific information
    if is_amazon_linux(os_data):
        print()
        print("Amazon Linux Version Details:")
        print("=============================")
        version_id = os_data.get('VERSION_ID', '')
        if version_id == '2018.03':
```

```
            print("Amazon Linux AMI (version 1)")
        elif version_id == '2':
            print("Amazon Linux 2")
        elif version_id == '2023':
            print("Amazon Linux 2023")
        else:
            print(f"Unknown Amazon Linux version: {version_id}")

        # Check for Amazon Linux specific files
        if os.path.exists('/etc/image-id'):
            print("Amazon Linux image-id file present")

if __name__ == '__main__':
    main()
```

Perl

```
#!/usr/bin/env perl

use strict;
use warnings;

# Function to parse the os-release file and return a hash of key-value pairs
sub parse_os_release {
    my %os_release_data;

    # Try /etc/os-release first, then /usr/lib/os-release
    my @paths = ('/etc/os-release', '/usr/lib/os-release');

    for my $path (@paths) {
        if (-f $path) {
            if (open(my $fh, '<', $path)) {
                while (my $line = <$fh>) {
                    chomp $line;
                    next if $line =~ /^\s*$/ || $line =~ /^\s*#/;

                    if ($line =~ /^([^=]+)=(.*)$/) {
                        my ($key, $value) = ($1, $2);
                        # Remove quotes if present
                        $value =~ s/^["']|["']$//g;
                        $os_release_data{$key} = $value;
                    }
                }
```

```
                close($fh);
                return %os_release_data;
            }
        }
    }

    die "Error: os-release file not found\n";
}

# Function to check if this is Amazon Linux
sub is_amazon_linux {
    my %os_data = @_;
    return ($os_data{ID} // '') eq 'amzn';
}

# Function to check if this is Fedora
sub is_fedora {
    my %os_data = @_;
    return ($os_data{ID} // '') eq 'fedora';
}

# Function to check if this is Ubuntu
sub is_ubuntu {
    my %os_data = @_;
    return ($os_data{ID} // '') eq 'ubuntu';
}

# Function to check if this is Debian
sub is_debian {
    my %os_data = @_;
    return ($os_data{ID} // '') eq 'debian';
}

# Function to check if this is like Fedora (includes Amazon Linux, CentOS, RHEL,
 etc.)
sub is_like_fedora {
    my %os_data = @_;
    return 1 if ($os_data{ID} // '') eq 'fedora';
    my $id_like = $os_data{ID_LIKE} // '';
    return $id_like =~ /fedora/;
}

# Function to check if this is like Debian (includes Ubuntu and derivatives)
sub is_like_debian {
```

```
    my %os_data = @_;
    return 1 if ($os_data{ID} // '') eq 'debian';
    my $id_like = $os_data{ID_LIKE} // '';
    return $id_like =~ /debian/;
}

# Main execution
my %os_data = parse_os_release();

# Display results
print "Operating System Detection Results:\n";
print "==================================\n";
print "Is Amazon Linux: " . (is_amazon_linux(%os_data) ? "YES" : "NO") . "\n";
print "Is Fedora: " . (is_fedora(%os_data) ? "YES" : "NO") . "\n";
print "Is Ubuntu: " . (is_ubuntu(%os_data) ? "YES" : "NO") . "\n";
print "Is Debian: " . (is_debian(%os_data) ? "YES" : "NO") . "\n";
print "Is like Fedora: " . (is_like_fedora(%os_data) ? "YES" : "NO") . "\n";
print "Is like Debian: " . (is_like_debian(%os_data) ? "YES" : "NO") . "\n";
print "\n";

# Additional information
print "Detailed OS Information:\n";
print "========================\n";
print "ID: " . ($os_data{ID} // '') . "\n";
print "VERSION_ID: " . ($os_data{VERSION_ID} // '') . "\n";
print "PRETTY_NAME: " . ($os_data{PRETTY_NAME} // '') . "\n";
print "ID_LIKE: " . ($os_data{ID_LIKE} // '') . "\n" if $os_data{ID_LIKE};

# Amazon Linux specific information
if (is_amazon_linux(%os_data)) {
    print "\n";
    print "Amazon Linux Version Details:\n";
    print "=============================\n";
    my $version_id = $os_data{VERSION_ID} // '';

    if ($version_id eq '2018.03') {
        print "Amazon Linux AMI (version 1)\n";
    } elsif ($version_id eq '2') {
        print "Amazon Linux 2\n";
    } elsif ($version_id eq '2023') {
        print "Amazon Linux 2023\n";
    } else {
        print "Unknown Amazon Linux version: $version_id\n";
    }
```

```
    # Check for Amazon Linux specific files
    if (-f '/etc/image-id') {
        print "Amazon Linux image-id file present\n";
    }
}
```

當在不同系統上執行 時，指令碼會產生下列輸出：

AL2023

```
Operating System Detection Results:
===================================
Is Amazon Linux: YES
Is Fedora: NO
Is Ubuntu: NO
Is Debian: NO
Is like Fedora: YES
Is like Debian: NO

Detailed OS Information:
========================
ID: amzn
VERSION_ID: 2023
PRETTY_NAME: Amazon Linux 2023.8.20250721
ID_LIKE: fedora

Amazon Linux Version Details:
=============================
Amazon Linux 2023
Amazon Linux image-id file present
```

AL2

```
Operating System Detection Results:
===================================
Is Amazon Linux: YES
Is Fedora: NO
Is Ubuntu: NO
Is Debian: NO
Is like Fedora: YES
Is like Debian: NO
```

```
Detailed OS Information:
=======================
ID: amzn
VERSION_ID: 2
PRETTY_NAME: Amazon Linux 2
ID_LIKE: centos rhel fedora

Amazon Linux Version Details:
============================
Amazon Linux 2
Amazon Linux image-id file present
```

Amazon Linux AMI

```
Operating System Detection Results:
==================================
Is Amazon Linux: YES
Is Fedora: NO
Is Ubuntu: NO
Is Debian: NO
Is like Fedora: YES
Is like Debian: NO

Detailed OS Information:
=======================
ID: amzn
VERSION_ID: 2018.03
PRETTY_NAME: Amazon Linux AMI 2018.03
ID_LIKE: rhel fedora

Amazon Linux Version Details:
============================
Amazon Linux AMI (version 1)
Amazon Linux image-id file present
```

Ubuntu

```
Operating System Detection Results:
==================================
Is Amazon Linux: NO
Is Fedora: NO
Is Ubuntu: YES
```

```
Is Debian: NO
Is like Fedora: NO
Is like Debian: YES

Detailed OS Information:
========================
ID: ubuntu
VERSION_ID: 24.04
PRETTY_NAME: Ubuntu 24.04.2 LTS
ID_LIKE: debian
```

Debian

```
Operating System Detection Results:
===================================
Is Amazon Linux: NO
Is Fedora: NO
Is Ubuntu: NO
Is Debian: YES
Is like Fedora: NO
Is like Debian: YES

Detailed OS Information:
========================
ID: debian
VERSION_ID: 12
PRETTY_NAME: Debian GNU/Linux 12 (bookworm)
```

Fedora

```
Operating System Detection Results:
===================================
Is Amazon Linux: NO
Is Fedora: YES
Is Ubuntu: NO
Is Debian: NO
Is like Fedora: YES
Is like Debian: NO

Detailed OS Information:
========================
ID: fedora
VERSION_ID: 42
```

```
PRETTY_NAME: Fedora Linux 42 (Container Image)
```

檔案系統配置

本節涵蓋 AL2023 系統的檔案系統配置，包括可能特定於執行個體或 AL2023 型容器的詳細資訊。如需詳細資訊，請參閱 `file-hierarchy(7)`man頁面。

主題

/ （根目錄）

根據預設，AL2023 映像會設定為可寫入的 `/`，允許特殊權限使用者建立新的檔案和目錄。

您可以將 `systemd` 服務設定為使用不同的路徑或映像來顯示為/該服務，並在任何路徑上放置存取限制。

> **Note**
>
> 服務的最佳實務`systemd`是設定為限制服務可存取的內容。這可能包括使用 `ReadOnlyPaths=/`指令，讓該服務/唯讀。
> 如需使用 `systemd` 限制服務對系統存取的詳細資訊，請參閱 `systemd.exec(5)`man頁面。

/boot (Kernel、 initramfs等）

根據預設，可開機 AL2023 映像設定為/`boot`在root檔案系統上。`/boot` 路徑僅與可開機映像相關，因此在 AL2023 容器映像中未使用。

此目錄包含 AL2023 開機所需的檔案，例如 Linux 核心和 initramfs。此目錄的內容只能使用作業系統隨附的工具操作。

/boot/efi (EFI 系統分割區)

根據預設，可開機 AL2023 映像的設定會將EFI System分割區掛載在 /boot/efi。此檔案系統由作業系統管理，並包含啟動系統的關鍵程式碼和組態。

此路徑與容器映像無關。

/etc （系統組態）

AL2023 上的 /etc目錄包含系統特定的組態。根據預設，AL2023 映像隨附/etc於 root 檔案系統，並由具有特殊權限的使用者撰寫。

> **Note**
>
> 應用程式 （包括 systemd) 通常會保留預設組態/usr （系統資源），其可透過在 中放置組態來覆寫/etc （系統組態）。
> 對於這些應用程式，在 中變更檔案/usr （系統資源）而不是在 中覆寫預設組態，/etc可能會導致套件更新時覆寫變更。

/home （使用者主目錄）

一般使用者在 下有其主目錄/home，但軟體應一律尋找每個使用者$HOME的環境變數，而不是依賴模式，例如 /home/$USER。

根據預設，AL2023 映像/home在root檔案系統上具有 ，但軟體不應依賴此項目。將作業系統設定為/home獨立的檔案系統是完全有效的，該系統稍後會在開機期間掛載，或只在使用者向系統進行身分驗證之後才安裝。

root 使用者主目錄不在 中/home，而是/root (root 使用者主目錄）在無法掛載/home檔案系統時使用。

> **Note**
>
> 對於systemd不需要寫入存取權/home的服務，最佳實務是使用 ProtectHome=read-only指令來設定 。使用此選項時， /home、 /root和 /run/user 會針對服務設為唯讀。

也適用於不需要任何存取 的服務/home，以使用 ProtectHome=tmpfs指令進行設定，這會在沙盒中執行服務/root，其中 /home、 和 /run/user是空的唯讀tmpfs檔案系統。如需使用 systemd 限制服務對系統存取的詳細資訊，請參閱 systemd.exec(5)man頁面。

/root (root 使用者主目錄）

root 使用者的主目錄是 /root目錄，刻意與 分隔，/home （使用者主目錄）以便在無法使用/home （使用者主目錄）的檔案系統上的事件中存在。

設定 systemd服務的最佳實務/root與 相同/home （使用者主目錄）。

/srv （伺服器承載）

/srv 目錄由 系統管理員管理，Amazon Linux 2023 不會限制此目錄的組織方式。

您可以將/srv目錄設定為位於不同的檔案系統上，因此它只能在稍後開機時使用。

/tmp （小型暫存檔案）

Note

Amazon Linux 2023 與 Amazon Linux 2 不同，因為現在預設為 root 檔案系統上的/tmptmpfs路徑。

Note

在容器中執行時，通常是您的容器執行期組態，指示 /tmp是否為 tmpfs或磁碟上的路徑，以及是否有執行中的清除程序。

/tmp 目錄適用於小型、大小限制的暫存檔案。根據預設，AL2023 會將其設定為大小限制為 RAM 的 50% 且上限為一百萬個 tmpfs的檔案系統inodes。

應用程式應該偏好$TMPDIR環境變數中的路徑，而不是 /tmp。然後，使用者可以設定$TMPDIR環境變數來覆寫應用程式應該用於 的路徑 /tmp

對於較大的暫存檔案，/var/tmp應該改用 。

> **Warning**
>
> 由於 /tmp是共用的，因此請務必使用安全方法來建立暫存檔案。如需詳細資訊，請參閱使用 /tmp和 /var/tmp 安全上的上游systemd文件。

> **Note**
>
> 最佳實務是將systemd服務設定為在沙盒中執行服務的PrivateTmp=指令設定為 yes或 disconnected ，其中 /tmp和 /var/tmp 不會與主機或其他 服務共用。
> 如需詳細資訊，包括如何設定兩個服務以共用相同的私有暫時目錄，請參閱 systemd.exec(5)man頁面。

的內容/tmp通常會在開機時清除，未使用的檔案也會定期清除。根據預設，清除程序會在開機後不久執行，然後每天執行。如需如何設定暫存檔案清除的資訊，請參閱 tmpfiles.d(5)和 systemd-tmpfiles(8) man man 頁面。

/tmp 和 /var/tmp 路徑密切相關，並存在用於不同目的。

/run（執行時間資料）

系統套件會使用 /run目錄來存放少量執行時間資料（例如通訊端檔案）。它是tmpfs檔案系統，只能由特殊權限程式寫入。

系統元件可以在寫入檔案系統之前/var/log或在/var/log檔案系統可用之前，使用該/run/log目錄來存放登入。

/run/user/ 路徑包含每個使用者的執行時間目錄。根據預設，這些檔案tmpfs系統會在使用者登入systemd時由 掛載，並在使用者不再登入時清除。根據 XDG Base Directory 規格，不應直接參考這些路徑，而是透過$XDG_RUNTIME_DIR環境變數。

/usr（系統資源）

/usr 階層適用於廠商提供的作業系統資源。除了/usr/local階層之外，除了作業系統套件管理員/usr之外， 下的任何內容都不應修改。

軟體應用程式必須假設 /usr可以是唯讀的。/usr 階層不得用於揮發性資料。除了 之外/usr/local，/usr階層不得用於在套件安裝/移除之外新增或變更的任何資料，如作業系統套件管理員所完成。作業系統套件管理員可能會假設所有/usr階層（除了 /usr/local) 都是相同的掛載點。

安裝在作業系統套件管理員外部的軟體不應將資料存放在 中/usr，因為這可能會阻礙作業系統套件管理員日後的任何調用。/usr/local 階層是例外狀況，並保留給作業系統套件管理員以外的軟體。

/usr/bin（可執行檔）

應該出現在標準搜尋 中的可執行檔$PATH，對於從 shell 叫用很有用。對從 shell 叫用沒有幫助的協助程式和可執行檔，而是存在於 /usr/lib或 中/usr/libexec。

/usr/include (C/C++ 標頭）

/usr/include 目錄包含 C 和 C++ 標頭檔案，通常包含在尾碼為 -devel 的套件中。

/usr/lib 和 /usr/lib64（共用程式庫）

在 Amazon Linux 2023 上，/usr/lib64路徑用於 64 位元共用程式庫，以及與架構相關的套件資料。由於 AL2023 未隨附任何 32 位元使用者空間支援，因此只有 64 位元的共用程式庫可用。

/usr/lib 路徑適用於與所有架構相容的作業系統套件靜態資料。這可能包括通常不會從 shell 叫用的可執行檔，這也可以在 中找到/usr/libexec。共用程式庫位於 中，/usr/lib64而不是 /usr/lib。

/usr/local（系統管理員安裝的軟體）

在 Amazon Linux 2023 上，此/usr/local路徑可供系統管理員在作業系統中安裝軟體、非作業系統擁有的軟體，而且不會被作業系統觸碰。預設/usr/local階層會鏡像/階層。

/usr/share（共用資源）

文件、字型和時區資料等共用資源都存在於 中/usr/share。各種規格通常可準確決定資料存放在此目錄中的位置和格式。

/usr/share/doc（文件）

套件隨附的文件將存放在 中/usr/share/doc。

/var（持久性變數系統資料）

/var/cache（快取）

與 相反/var/lib，在 中清除資料/var/cache不會導致資料遺失，因為應用程式需要能夠從其他來源重建其/var/cache資料。

/var/lib（持久性系統資料）

/var/lib 目錄用於持久性系統資料。各種系統元件會將該元件的私有資料放在此處。與 相反/var/cache，在 中清除資料/var/lib會導致資料遺失。

例如，PostgreSQL 資料庫伺服器預設會將資料庫資料存放在 中/var/lib/pgsql。此資料的配置和檔案格式是 PostgreSQL 的私有格式，而且是持久性資料，就像清除一樣，使用者會遇到資料遺失。

/var/log（持久性日誌）

此目錄用於存放持久性日誌。建議軟體使用 syslog(3)或 sd_journal_print(3) API 呼叫，而不是直接將日誌檔案存放在 下/var/log。

> **Note**
>
> 在 AL2023 中systemd 日誌取代 rsyslog，這是與預設 Amazon Linux 2 組態的顯著差異。

如需使用 讀取日誌的詳細資訊journalctl，請參閱 journalctl 手動頁面。

許多應用程式使用自己的機制來撰寫，有時會輪換 中找到的日誌檔案/var/log。請參閱這些應用程式的文件，了解如何設定其日誌檔案。

/var/spool（郵件和印表機佇列）

此目錄用於持久性資料，例如郵件或印表機佇列。

/var/tmp（較大的暫存檔案）

對於小型、大小限制的暫存檔案， /tmp 應該改為使用。

雖然 /tmp 預設為磁碟tmpfs區，但 /var/tmp 預設為根檔案系統的路徑，因此是較大且更持久的暫存檔案。根據預設，會定期執行清除任務，以移除最近未存取的檔案。

如需如何設定暫存檔案清除的資訊，請參閱 tmpfiles.d(5)和 systemd-tmpfiles(8) man man 頁面。

如同 /tmp，應用程式應該偏好$TMPDIR環境變數中指定的路徑，而不是 /var/tmp。然後，使用者可以設定$TMPDIR環境變數來覆寫應用程式應該用於 的路徑/var/tmp。

> **Warning**
>
> 由於 /var/tmp 是共用的 （ 如同 /tmp，請務必使用安全方法來建立暫存檔案。如需詳細資訊，請參閱使用 /tmp和 /var/tmp 安全上的上游systemd文件。

> **Note**
>
> 最佳實務是將systemd服務設定為在沙盒中執行服務的PrivateTmp=指令設定為 yes或 disconnected ，其中 /tmp和 /var/tmp 不會與主機或其他 服務共用。
> 如需詳細資訊，包括如何設定兩個 服務以共用相同的私有暫時目錄，請參閱 systemd.exec(5)man頁面。

/tmp 和 /var/tmp 路徑密切相關，並存在用於不同目的。

更新 AL2023

請務必隨時更新 AL2023 版本，以便從安全性更新和新功能中獲益。使用 AL2023 時，您可以透過 透過 AL2023 上的版本控制儲存庫進行確定性升級 在整個環境中確保套件版本與更新之間的一致性。

> **Warning**
>
> 執行`dnf --releasever=latest update`不是最佳實務，而且可能會導致作業系統更新先在生產環境中進行測試。
> 使用特定的 AL2023 發行版本`latest`，而不是使用 。這可確保您在生產執行個體之間部署與先前測試相同的變更。例如，`dnf --releasever=2023.10.20260216 update` 一律會更新至 2023.10.20260216 版本。
> 如需詳細資訊，請參閱《AL2023 使用者指南》中的更新 AL2023 一節。

主題

安全部署更新的最佳實務

Amazon Linux 2023 (AL2023) 具有多種功能，旨在協助安全地部署作業系統更新，並能夠了解更新之間的變更，並視需要輕鬆還原至較舊的版本。本節探討 AWS 自使用 Amazon Linux 十多年的內部和外部經驗中學到的經驗。

> **Warning**
>
> 執行`dnf --releasever=latest update`不是最佳實務，而且可能會導致作業系統更新先在生產環境中進行測試。

使用特定的 AL2023 發行版本`latest`，而不是使用 。這可確保您在生產執行個體之間部署與先前測試相同的變更。例如， `dnf --releasever=2023.10.20260216 update` 一律會更新至 2023.10.20260216 版本。
如需詳細資訊，請參閱《AL2023 使用者指南》中的更新 AL2023 一節。

如果不規劃作業系統更新的部署安全性，應用程式/服務與作業系統更新之間非預期的負面互動的影響可能會明顯更大，甚至包括總中斷。如同任何軟體問題，越早偵測到問題，它對最終使用者的影響就越小。

重要的是不要陷入相信兩個基本上不真實的陷阱：

1. 作業系統廠商絕不會在更新作業系統時發生錯誤。
2. 您倚賴之作業系統的特定行為或界面符合作業系統廠商會認為倚賴之行為和界面。

   即作業系統廠商和您同意更新發生問題。

請勿依賴良好的意圖，將系統置於適當位置，以確保部署安全包含作業系統的任何更新。

不建議透過部署到生產環境來測試新的作業系統更新。最佳實務是將作業系統視為部署的另一個部分，並考慮將您認為適合任何其他變更的相同部署安全機制套用至生產環境。

最佳實務是在部署到生產系統之前測試任何和所有作業系統更新。部署時，建議使用結合良好監控的暫存推展。分階段推展可確保如果發生問題，即使不是立即發生，影響也會限制在機群的子集，並且可以停止進一步部署更新，同時可以進行進一步的調查和緩解。

對作業系統進行更新的任何負面影響的緩解通常是首要任務，然後解決問題，無論問題在哪裡。在引進作業系統更新與負面影響相關的情況下，還原至先前已知良好版本的作業系統是功能強大的工具。

Amazon Linux 2023 推出 透過版本控制的儲存庫進行確定性升級，這是功能強大的新功能，可確保作業系統版本（或個別套件）的任何變更都是可重複的。因此，如果在從某個作業系統版本移至下一個版本時遇到問題，則可以輕鬆地使用機制來粘附到已知工作中的作業系統版本，同時了解如何解決問題。

使用 AL2023 時，每當我們發行新的套件更新時，都有要鎖定的新版本，以及鎖定到該版本的新 AMIs。AL2023 版本備註涵蓋每個版本中的變更，並AL2023 的 Amazon Linux 安全建議涵蓋套件更新中解決的安全性問題。

例如，如果您受到 2023.6.20241028 版本中存在的問題影響，您可以立即還原為使用先前版本的 AMIs 和容器映像 2023.6.20241010。在此情況下，在後續 2023.6.20241031 版本中修正了套件中的錯誤，

但任何受影響的透過版本控制的儲存庫進行確定性升級人都可以立即採取簡單的動作來緩解：只要使用先前的映像即可。

透過版本控制的儲存庫進行確定性升級 也保證任何進行中的作業系統更新部署，無論是就地部署，還是啟動新的 AMIs 或容器映像，都不會受到後續發行的作業系統更新的影響。

在我們的第一個範例中，機群 A 是一個大型機群，在 2023.5.20241001 版本推出時，透過將更新從 2023.6.20241010 部署到 2023.6.20241028 版本進行一半。 透過版本控制的儲存庫進行確定性升級表示機群 A 的部署會繼續，而不會對其套用的更新進行任何變更。

波浪型或階段型部署策略的用途，例如先部署到 1% 的機群，然後 5%、10%、20%、40%，直到達到 100%，才能夠以有限的方式測試變更，然後再推出。這種類型的部署策略通常被視為部署任何生產變更的最佳實務。

透過以波動為基礎的部署策略和機群 2023.6.20241010 的更新處於一次部署到許多主機的階段，由於使用 ，2023.6.20241028 發佈的事實不會影響進行中的部署透過版本控制的儲存庫進行確定性升級。

如果機群 B 執行的是較舊的版本，例如 2023.5.20240708，並已開始將更新部署至 2023.6.20241028，且機群 B 受到該版本的問題影響，則會在部署初期注意到此問題。此時，可以決定是否暫停任何推展，直到該問題的修正可用，或者如果同時開始部署相同的版本機群 A 正在執行，則為 2023.6.20241010，以便機群 B 在 2023.5.20240708 和 2023.6.20241010 之間取得所有更新。

請務必注意，未及時進行作業系統更新可能會導致問題。新的更新可能包含可能與您的環境相關的錯誤和安全性修正。如需詳細資訊，請參閱Amazon Linux 2023 的安全與合規及在 AL2023 中管理套件和作業系統更新。

請務必將部署系統設定為能夠輕鬆進行新的作業系統更新、在部署至生產環境之前進行測試，以及使用波型部署等機制，將任何負面影響降至最低。為了減輕作業系統更新的任何負面影響，請務必了解如何讓您的部署系統指向先前已知良好的作業系統版本，一旦問題解決，就不會再鎖定至較舊的已知良好的版本，而是改用新的已知良好的版本。

準備次要更新

準備較小的作業系統更新，例如新的 AL2023 點版本，其用途僅限於不費吹灰之力。請務必閱讀 AL2023 版本備註，了解任何即將進行的變更。

即將結束的套件支援期間可能涉及移至較新版本的語言執行時間 （例如使用 AL2023 中的 PHP)。最佳實務是在支援期間結束之前，輕鬆移至新的語言執行時間版本，事先為此做好準備。

對於 等套件pcre 第 1 版，也有機會事先規劃並將任何程式碼遷移至其替換，在這種情況下，其為第 2 pcre版。最佳實務是盡快這樣做，以允許任何挫折的時間。

如果沒有直接取代，例如使用 Berkeley 資料庫 (libdb)，您可能需要根據您的使用案例做出選擇。

準備重大更新

更新至作業系統的新主要版本幾乎會普遍被視為需要規劃、努力適應變更或棄用的功能，以及在部署之前進行測試。能夠更遞增地準備下一個主要版本的 Amazon Linux 2023 並不罕見，例如在繼續移至下一個主要版本之前，處理任何已棄用或已移除的功能。

例如，從 AL2 移至 AL2023 時，讀取 AL2 中的功能已棄用，並在 AL2023 中移除區段可能會導致數個安全且小的步驟，這可能會在仍然使用 AL2 準備 AL2023 時發生。例如，任何Python 2.7 已替換為 Python 3用量（作業系統使用之外，例如yum套件管理員中的）都可以遷移至 Python 3，以準備使用 AL2023 中的 Python。如果使用 PHP，則 AL2（透過 PHP 8.2 AL2 Extra) 和 AL2023 都隨附 PHP 8.2，因此 PHP 版本遷移和作業系統遷移不必同時發生。

使用 AL2023 時，也可以立即準備下一個主要版本的 Amazon Linux 2023，同時使用 AL2023。在 AL2023 中已棄用 本節涵蓋在 AL2023 中已棄用且由於移除而淘汰的功能和套件。

例如，遷移任何剩餘的System V init (sysvinit)使用，例如init指令碼到其systemd對等項目，將讓您為未來做好準備，並允許您使用完整的systemd功能來監控服務、如何及是否重新啟動服務、它需要哪些其他服務，以及是否應套用任何資源或許可限制。

對於 32 位元支援等功能，棄用可以跨越作業系統的多個主要版本。對於 32 位元，Amazon Linux 1 (AL1) 已棄用 32 位元 x86 (i686) AMIs、Amazon Linux 2 已棄用 32 位元 x86 (i686) 套件和 Amazon Linux 2023 已棄用 32 位元 x86 (i686) 執行時間支援。從 轉換IMDSv1也跨越作業系統的多個主要版本。對於這些類型的變更，我們了解某些客戶需要更長的時間來適應這些變更，因此在 Amazon Linux 2023 中不再提供此功能之前，會有大量的關聯。

已棄用的功能清單會在作業系統的生命週期內更新，建議您隨時掌握其變更。

接收新更新的通知

每當新的 AL2023 AMI 發行時，您都可以收到通知。通知將透過 Amazon SNS 並使用下列主題發布。

```
arn:aws:sns:us-east-1:137112412989:amazon-linux-2023-ami-updates
```

當發布新的 AL2023 AMI 時，會在此處貼出訊息。AMI 的版本將包含在訊息中。

這些訊息可以使用幾種不同的方法來接收。建議您使用下列方法。

1. 開啟 Amazon SNS 主控台。

2. 在導覽列中，視需要將 AWS 區域 變更為美國東部 （維吉尼亞北部）。您必須選取已訂閱且建立 SNS 通知的區域。
3. 在導覽面板中依序選擇 Subscriptions (訂閱) 與 Create subscription (建立訂閱)。
4. 針對 Create subscription (建立訂閱) 對話方塊，執行下列作業：
   a. 針對主題 ARN，複製並貼上下列 Amazon Resource Name (ARN)：**`arn:aws:sns:us-east-1:137112412989:amazon-linux-2023-ami-updates`**。
   b. 對於 通訊協定 ，選擇 電子郵件 。
   c. 針對 Endpoint (端點)，請輸入可用於接收通知的電子郵件地址。
   d. 選擇 Create subscription (建立訂閱)。
5. 您會收到確認電子郵件，主旨行為「AWS 通知 - 訂閱確認」。開啟電子郵件並選擇 Confirm subscription (確認訂閱) 完成訂閱。

透過 AL2023 上的版本控制儲存庫進行確定性升級

> **Note**
>
> 根據預設，AL2023 執行個體不會在啟動時自動接收其他關鍵和重要的安全性更新。執行個體最初包含 AL2023 和所選 AMI 中可用的更新。

控制從主要和次要版本收到的更新

使用 AL2023 時，您可以在整個環境中確保套件版本與更新之間的一致性。您也可以確保相同 Amazon Machine Image (AMI) 的多個執行個體的一致性。透過版本化儲存庫功能進行確定性升級 (預設為開啟) 時，您可以根據所需排程套用更新。

每當我們發布新的套件更新時，都會有一個新版本要鎖定，以及鎖定到該版本的新 AMI。

AL2023 會鎖定特定版本的儲存庫。主要或次要版本都支援此功能。透過 SSM 參數公開的 AL2023 AMI 永遠是最新版本。其具有最新的套件和更新，包括關鍵和重要的安全性更新。

如果您從現有 AMI 啟動執行個體，則不會自動套用更新。在佈建時安裝的任何其他套件都會對應至現有 AMI 的儲存庫版本。

使用此功能時，您可以確保整個環境中的套件版本和更新之間的一致性。如果您要從同一 AMI 啟動多個執行個體，則尤其如此。您可以根據所需排程來套用更新。您也可以在啟動時套用一組特定的更新，因為這些更新也可以鎖定至特定的儲存庫版本。

主要和次要版本升級之間的差異

AL2023 的主要版本包含大規模更新，且可能會新增、刪除或更新套件。為確保相容性，請在您在該版本上測試應用程式後，再將執行個體升級至新的主要版本。

AL2023 的次要版本包含功能和安全性更新，但不包括套件變更。這樣可以確保 Linux 功能和系統程式庫 API 在新版本上保持可用。不需要在更新前測試您的應用程式。

了解何時提供更新

若要套用更新，您需要知道有可用的更新，然後了解如何部署更新。

對於在發行新的 AL2023 AMIs 時建置衍生AMIs，EC2 Image Builder 可以自動建置、修補和測試 AMIs。若要觸發您自己的 AMI 建置管道，或使用基本 AMIs，您可以 接收新更新的通知。

對於就地修補，您可以使用AWS Systems Manager 修補程式管理員等工具來協調跨機群套用更新。

對於以 AMIsAL2023 為基礎的其他公有 AMI，這些 AMIs 的供應商可能有自己的發行排程和通知方法。使用衍生AMIs 或容器映像時，請檢查發佈者到更新發佈時的 文件。

每個版本中的變更都會記錄在 AL2023 版本備註中。安全性更新發佈於 Amazon Linux 安全中心 (ALAS)。

控制 AL2023 儲存庫提供的套件更新

當我們發佈新版本的 AL2023 儲存庫時，所有先前的版本仍然可用。預設情況下，用於管理儲存庫版本的外掛程式會鎖定到用來建立 AMI 的相同版本。如果您想要控制套件更新，請遵循下列步驟。

1. 執行下列命令來探索可用的儲存庫版本。

```
$ sudo dnf check-release-update
```

2. 執行以下命令來選取版本。

```
$ sudo dnf upgrade --releasever=version
```

此命令會使用 dnf 並從目前 Amazon Linux 發行版本，開始更新到命令列指定的發行版本。套件更新的清單會由顯示 dnf。在處理更新之前，您必須確認更新。更新完成後，新發行版本會成為 dnf 用於所有未來活動的預設發行版本。

如需詳細資訊，請參閱在 AL2023 中管理套件和作業系統更新。

透過執行個體替換進行確定性更新

Amazon Linux 2023 透過 AL2023 上的版本控制儲存庫進行確定性升級的功能讓執行個體替換成為決定性且安全地推出 AL2023 更新版本的一種簡單方法。確定性更新表示，隨著新版本逐步推出，如果發現任何問題，在確定問題的原因時，還原到先前的 AMI 很簡單。

使用執行個體取代而非就地修補，表示更新更確定且可預測，因為啟動新容量可能是經過測試且具有清晰 A 和 B 狀態的程式碼路徑。部署開始之前，可在 CI/CD 系統中對每個 前後狀態進行良好測試。

執行就地修補時，套用更新前後有許多中間狀態，這較難測試所有狀態組合。

使用執行個體替換搭配確定性更新的作業系統更新策略，非常適合藍/綠、波浪和階段式部署模型。

透過版本控制的儲存庫使用確定性升級

主題

使用確定性升級的系統

> **Note**
>
> 套件管理員的預設行為已從 AL2 變更。

確定性升級是一種強而有力的方式，可確保在廣泛部署之前完整測試生產環境的所有變更。每個新的 AL2023 AMI 都會鎖定至特定版本的 AL2023。這可提供啟動特定 AMI 時已安裝哪些作業系統套件版本的確定性行為。就地更新可以是特定的發行版本，以確保整個機群的確定性行為。當您移至新的 AMIs 或就地更新版本時，您可以在 CI/CD 管道中測試每個更新版本，在部署到生產環境之前發現任何潛在問題。

您可以使用AWS Systems Manager 修補程式管理員等工具，協調跨機群套用更新。對於在發行新的 AL2023 AMIs 時建置衍生的 AMI，EC2 Image Builder 可以自動建置、修補和測試 AMIs，或者您可以接收新更新的通知知道何時有新的基礎 AMIs，或觸發您自己的 AMI 建置管道。 AMIs

如需限制更新特定諮詢之更新的資訊，請參閱 就地套用安全性更新

對於就地修補，您可以使用dnf套件管理員。當您執行 dnf upgrade 命令時，系統會檢查 releasever 變數指定的儲存庫中是否有升級。有效 releasever 為##版本或日期戳記版本，例如 *2023.10.20260216*。

您可以使用下列其中一種方法，來變更 releasever 的值。這些方法以降序系統優先順序列出。這表示方法 1 會覆寫方法 2 和方法 3，而方法 2 會覆寫方法 3。

1. 命令行旗標中的值 --releasever=*latest* (如有使用)。
2. 在覆寫變數檔案中指定的值 /etc/dnf/vars/releasever (如已設定)。
3. 目前安裝的 system-release 套件版本。

在下列範例中，版本為 *2023.0.20230210*：

```
$ rpm -q system-release
system-release-2023.0.20230210-0.amzn2023.noarch
```

在新安裝的系統中，覆寫變數不存在。沒有可用的升級，因為系統已鎖定至 system-release 的已安裝版本。

```
$ cat /etc/dnf/vars/releasever
cat: /etc/dnf/vars/releasever: No such file or directory
```

```
$ sudo dnf upgrade
Last metadata expiration check: 0:00:02 ago on Wed 15 Feb 2023 06:14:12 PM UTC.
Dependencies resolved.
Nothing to do.
Complete!
```

您可以使用 releasever 旗標來提供您想要的版本，以取得特定版本的套件。

```
$ rpm -q system-release
system-release-2023.0.20230222-0.amzn2023.noarch
```

```
$ sudo dnf upgrade --releasever=2023.0.20230329
Amazon Linux 2023 repository                    26 MB/s |  12 MB     00:00
Dependencies resolved.
================================================================================
 Package                 Arch    Version                      Repository   Size
```

```
================================================================================
Installing:
 kernel                   aarch64 6.1.21-1.45.amzn2023        amazonlinux   26 M
Upgrading:
 amazon-linux-repo-s3     noarch  2023.0.20230329-0.amzn2023  amazonlinux   18 k
 ca-certificates          noarch  2023.2.60-1.0.amzn2023.0.1  amazonlinux  828 k
 cloud-init               noarch  22.2.2-1.amzn2023.1.7       amazonlinux  1.1 M

         ... [ list edited for clarity ]

 system-release           noarch  2023.0.20230329-0.amzn2023  amazonlinux   29 k

         ... [ list edited for clarity ]

 vim-data                 noarch  2:9.0.1403-1.amzn2023.0.1   amazonlinux   25 k
 vim-minimal              aarch64 2:9.0.1403-1.amzn2023.0.1   amazonlinux  753 k

Transaction Summary
================================================================================
Install    1 Package
Upgrade   42 Packages

Total download size: 56 M
```

由於 `--releasever` 選項會同時覆寫 `system-release` 和 `/etc/dnf/vars/releasever`，因此此升級的結果如下：

1. 升級會取代所有在舊版和新版本之間變更的已安裝套件。
2. 升級會將系統鎖定至儲存庫的新版本 `system-release`。

透過一律指定要更新的內容 `releasever`（即 AL2023 版本），您在機群中有一組決定性變更。您已啟動 *A* 版，更新至 *B*，然後更新至 *C*。

確定性升級系統的選擇性更新

> **Note**
>
> 建議您安裝新版本中的所有更新，而不是選取特定的更新。只有將部分更新套用至作業系統，才應該是進行整個更新的標準實務的例外狀況。

您可能想要安裝最新版本的特定套件，並將系統鎖定為原始發行版本。

您可以使用 dnf check-update 來找出要升級的套件。

```
$ sudo dnf check-update --releasever=latest --security
Amazon Linux 2023 repository                    13 MB/s |  10 MB     00:00
Last metadata expiration check: 0:00:02 ago on Wed 15 Feb 2023 02:52:21 AM UTC.

bind-libs.aarch64                 32:9.16.27-1.amzn2023.0.1         amazonlinux
bind-license.noarch               32:9.16.27-1.amzn2023.0.1         amazonlinux
bind-utils.aarch64                32:9.16.27-1.amzn2023.0.1         amazonlinux
cryptsetup.aarch64                2.4.3-2.amzn2023.0.1              amazonlinux
cryptsetup-libs.aarch64           2.4.3-2.amzn2023.0.1              amazonlinux
curl-minimal.aarch64              7.85.0-1.amzn2023.0.1             amazonlinux
glibc.aarch64                     2.34-40.amzn2023.0.2              amazonlinux
glibc-all-langpacks.aarch64       2.34-40.amzn2023.0.2              amazonlinux
glibc-common.aarch64              2.34-40.amzn2023.0.2              amazonlinux
glibc-locale-source.aarch64       2.34-40.amzn2023.0.2              amazonlinux
gmp.aarch64                       1:6.2.1-2.amzn2023.0.1            amazonlinux
gnupg2-minimal.aarch64            2.3.7-1.amzn2023.0.2              amazonlinux
gzip.aarch64                      1.10-5.amzn2023.0.1               amazonlinux
kernel.aarch64                    6.1.12-17.42.amzn2023             amazonlinux
kernel-tools.aarch64              6.1.12-17.42.amzn2023             amazonlinux
libarchive.aarch64                3.5.3-2.amzn2023.0.1              amazonlinux
libcurl-minimal.aarch64           7.85.0-1.amzn2023.0.1             amazonlinux
libsepol.aarch64                  3.4-3.amzn2023.0.2                amazonlinux
libsolv.aarch64                   0.7.22-1.amzn2023.0.1             amazonlinux
libxml2.aarch64                   2.9.14-1.amzn2023.0.1             amazonlinux
logrotate.aarch64                 3.20.1-2.amzn2023.0.2             amazonlinux
lua-libs.aarch64                  5.4.4-3.amzn2023.0.1              amazonlinux
lz4-libs.aarch64                  1.9.4-1.amzn2023.0.1              amazonlinux
openssl.aarch64                   1:3.0.5-1.amzn2023.0.3            amazonlinux
openssl-libs.aarch64              1:3.0.5-1.amzn2023.0.3            amazonlinux
pcre2.aarch64                     10.40-1.amzn2023.0.1              amazonlinux
pcre2-syntax.noarch               10.40-1.amzn2023.0.1              amazonlinux
rsync.aarch64                     3.2.6-1.amzn2023.0.2              amazonlinux
vim-common.aarch64                2:9.0.475-1.amzn2023.0.1          amazonlinux
vim-data.noarch                   2:9.0.475-1.amzn2023.0.1          amazonlinux
vim-enhanced.aarch64              2:9.0.475-1.amzn2023.0.1          amazonlinux
vim-filesystem.noarch             2:9.0.475-1.amzn2023.0.1          amazonlinux
vim-minimal.aarch64               2:9.0.475-1.amzn2023.0.1          amazonlinux
xz.aarch64                        5.2.5-9.amzn2023.0.1              amazonlinux
xz-libs.aarch64                   5.2.5-9.amzn2023.0.1              amazonlinux
```

```
zlib.aarch64                          1.2.11-32.amzn2023.0.3           amazonlinux
```

安裝您要升級的套件。使用 sudo dnf upgrade --releasever=latest 和套件名稱來確保 system-release 套件維持不變。

```
$ sudo dnf upgrade --releasever=latest openssl openssl-libs
Last metadata expiration check: 0:01:28 ago on Wed 15 Feb 2023 02:52:21 AM UTC.
Dependencies resolved.
================================================================================
 Package          Arch        Version                       Repository       Size
================================================================================
Upgrading:
 openssl          aarch64     1:3.0.5-1.amzn2023.0.3        amazonlinux     1.1 M
 openssl-libs     aarch64     1:3.0.5-1.amzn2023.0.3        amazonlinux     2.1 M

Transaction Summary
================================================================================
Upgrade  2 Packages

Total download size: 3.2 M
```

> **Note**
>
> 使用 sudo dnf upgrade --releasever=latest 更新所有套件，包括 system-release。然後，除非您設定持續取代，否則版本會保持鎖定為新的 system-release。

使用具決定性升級的持續覆寫

> **Note**
>
> 透過確定性更新，您可以將作業系統變更整合到您的 CI/CD 管道。停用確定性更新會移除在部署之前進行測試的能力。

透過將變數值設定為##，您就能使用持續覆寫來解鎖系統，而不是加入 --releasever=latest。透過一律使用 latest，這會將 AL2023 的行為還原至 AL2 更新模型，其中對套件管理員的任何呼叫一律會查看最新版本，而且不會鎖定至任何特定版本的作業系統。

> **Warning**
>
> 透過使用持續覆寫確定性更新來解鎖套件管理員，您會承擔風險，在您的應用程式與生產環境中的作業系統更新之間發現任何可能的不相容。
> 雖然不相容的情況很罕見，但隨著作業系統更新，您正在將新的程式碼變更整合到您的環境中，整合測試可以防止部署對生產環境造成負面影響的程式碼變更。

```
$ echo latest | sudo tee /etc/dnf/vars/releasever
latest
```

```
$ sudo dnf upgrade
Last metadata expiration check: 0:03:36 ago on Wed 15 Feb 2023 02:52:21 AM UTC.
Dependencies resolved.
================================================================================
 Package                 Arch    Version                       Repository   Size
================================================================================
Installing:
 kernel                  aarch64 6.1.73-45.135.amzn2023        amazonlinux  24 M
Upgrading:
 acl                     aarch64 2.3.1-2.amzn2023.0.1          amazonlinux  72 k
 alternatives            aarch64 1.15-2.amzn2023.0.1           amazonlinux  36 k
 amazon-ec2-net-utils    noarch  2.3.0-1.amzn2023.0.1          amazonlinux  16 k
 at                      aarch64 3.1.23-6.amzn2023.0.1         amazonlinux  60 k
 attr                    aarch64 2.5.1-3.amzn2023.0.1          amazonlinux  59 k
 audit                   aarch64 3.0.6-1.amzn2023.0.1          amazonlinux 249 k
 audit-libs              aarch64 3.0.6-1.amzn2023.0.1          amazonlinux 116 k
 aws-c-auth-libs         aarch64 0.6.5-6.amzn2023.0.2          amazonlinux  79 k
 aws-c-cal-libs          aarch64 0.5.12-7.amzn2023.0.2         amazonlinux  34 k
 aws-c-common-libs       aarch64 0.6.14-6.amzn2023.0.2         amazonlinux 119 k
 aws-c-compression-libs  aarch64 0.2.14-5.amzn2023.0.2         amazonlinux  22 k
 aws-c-event-stream-libs aarch64 0.2.7-5.amzn2023.0.2          amazonlinux  47 k
 aws-c-http-libs         aarch64 0.6.8-6.amzn2023.0.2          amazonlinux 147 k
 aws-c-io-libs           aarch64 0.10.12-5.amzn2023.0.6        amazonlinux 109 k
 aws-c-mqtt-libs         aarch64 0.7.8-7.amzn2023.0.2          amazonlinux  61 k
 aws-c-s3-libs           aarch64 0.1.27-5.amzn2023.0.3         amazonlinux  54 k
 aws-c-sdkutils-libs     aarch64 0.1.1-5.amzn2023.0.2          amazonlinux  26 k
 aws-checksums-libs      aarch64 0.1.12-5.amzn2023.0.2         amazonlinux  50 k
 awscli-2                noarch  2.7.8-1.amzn2023.0.4          amazonlinux 7.3 M
 basesystem              noarch  11-11.amzn2023.0.1            amazonlinux 7.8 k
 bash                    aarch64 5.1.8-2.amzn2023.0.1          amazonlinux 1.6 M
```

```
 bash-completion          noarch  1:2.11-2.amzn2023.0.1        amazonlinux 292 k
 bc                       aarch64 1.07.1-14.amzn2023.0.1       amazonlinux 120 k
 bind-libs                aarch64 32:9.16.27-1.amzn2023.0.1    amazonlinux 1.2 M
 bind-license             noarch  32:9.16.27-1.amzn2023.0.1    amazonlinux  14 k
 bind-utils               aarch64 32:9.16.27-1.amzn2023.0.1    amazonlinux 206 k
 binutils                 aarch64 2.38-20.amzn2023.0.3         amazonlinux 4.6 M
 boost-filesystem         aarch64 1.75.0-4.amzn2023.0.1        amazonlinux  55 k
 boost-system             aarch64 1.75.0-4.amzn2023.0.1        amazonlinux  14 k
 boost-thread             aarch64 1.75.0-4.amzn2023.0.1        amazonlinux  54 k
 bzip2                    aarch64 1.0.8-6.amzn2023.0.1         amazonlinux  53 k
 bzip2-libs               aarch64 1.0.8-6.amzn2023.0.1         amazonlinux  44 k
 c-ares                   aarch64 1.17.2-1.amzn2023.0.1        amazonlinux 107 k
 ca-certificates          noarch  2021.2.50-1.0.amzn2023.0.3   amazonlinux 343 k
 checkpolicy              aarch64 3.4-3.amzn2023.0.1           amazonlinux 345 k
 chkconfig                aarch64 1.15-2.amzn2023.0.1          amazonlinux 162 k
 chrony                   aarch64 4.2-7.amzn2023.0.4           amazonlinux 314 k
 cloud-init               noarch  22.2.2-1.amzn2023.1.7        amazonlinux 1.1 M
 cloud-utils-growpart     aarch64 0.31-8.amzn2023.0.2          amazonlinux  31 k
 coreutils                aarch64 8.32-30.amzn2023.0.2         amazonlinux 1.1 M
 coreutils-common         aarch64 8.32-30.amzn2023.0.2         amazonlinux 2.0 M
 cpio                     aarch64 2.13-10.amzn2023.0.1         amazonlinux 269 k
 cracklib                 aarch64 2.9.6-27.amzn2023.0.1        amazonlinux  83 k
 cracklib-dicts           aarch64 2.9.6-27.amzn2023.0.1        amazonlinux 3.6 M
 crontabs                 noarch  1.11-24.20190603git.amzn2023.0.1
                                                               amazonlinux  19 k
 crypto-policies          noarch  20230128-1.gitdfb10ea.amzn2023.0.1
                                                               amazonlinux  61 k
 crypto-policies-scripts noarch  20230128-1.gitdfb10ea.amzn2023.0.1
                                                               amazonlinux  81 k
...
Installing dependencies:
 amazon-linux-repo-cdn    noarch  2023.0.20230210-0.amzn2023   amazonlinux  16 k
 xxhash-libs              aarch64 0.8.0-3.amzn2023.0.1         amazonlinux  32 k
Installing weak dependencies:
 amazon-chrony-config     noarch  4.2-7.amzn2023.0.4           amazonlinux  14 k
 gawk-all-langpacks       aarch64 5.1.0-3.amzn2023.0.1         amazonlinux 207 k

Transaction Summary
================================================================================
Install    5 Packages
Upgrade  413 Packages

Total download size: 199 M
```

> **Note**
>
> 如果您使用覆寫變數 `/etc/dnf/vars/releasever`，請使用以下命令，以透過刪除覆寫值來還原預設鎖定行為。
>
> ```
> $ sudo rm /etc/dnf/vars/releasever
> ```

使用持久性覆寫來使用 `latest` 而非規格版本，與 AL2 的預設行為類似。有些服務會根據 AL2 建置 AMIs，以停用此行為，並鎖定特定套件版本，如您在 AL2023 上預設取得的版本。

建議您將執行個體取代為從新 AMI 啟動的執行個體，而不是停用決定性更新。如果執行個體替換不是選項，我們建議您使用AWS Systems Manager 修補程式管理員等工具，協調跨機群套用更新。EC2 Image Builder 也可以自動建置、修補和測試自 AL2023 基礎映像衍生的 AMIs。您也可以接收新更新的通知用來觸發自己的 AMI 建置管道。

`latest` 在生產前環境中使用 ，然後使用 部署到生產環境，`latest`無法提供作業系統更新與應用程式之間任何問題的保護。 新的 AL2023 版本可以在任何時間點進行，因此所有 的使用都在生產裝載風險`latest`中。

在 AL2023 中管理套件和作業系統更新

與舊版 Amazon Linux 不同，AL2023 AMIs 會鎖定到特定版本的 Amazon Linux 儲存庫。若要同時將安全性和錯誤修正套用至 AL2023 執行個體，請將DNF組態更新為最新的可用版本。或者，啟動較新的 AL2023 執行個體。

本節說明如何在執行中的執行個體管理 DNF 套件和儲存庫。另外說明如何從使用者資料指令碼設定 DNF，以在啟動時啟用最新的 Amazon Linux 儲存庫。如需更多詳細資訊，請參閱 DNF 命令參考。

建議套用新 AL2023 版本中可用的所有更新。僅選擇安全性更新，或僅選擇特定更新應該是例外狀況，而不是規則。如需與特定執行個體安全建議相關的清單，請參閱 列出適用的建議。如需僅安裝與特定諮詢相關的更新的資訊，請參閱就地套用安全性更新。

> **Important**
>
> 如果您想要報告漏洞或對 AWS 雲端服務或開放原始碼專案有安全疑慮，請使用漏洞報告頁面聯絡 AWS 安全部門

主題

檢查可用的套件更新

您可以使用 `dnf check-update` 命令來檢查系統是否有任何更新。對於 AL2023，建議您將此 `--releasever=`*`version-number`* 選項新增至命令。

當您新增此選項時，DNF 也會檢查儲存庫較新版本的更新。例如，執行 `dnf check-update` 命令後，請使用最新傳回的版本作為 *`version-number`* 的值.

如果執行個體已更新為使用最新版本的儲存庫，則輸出會包含要更新的所有套件清單。

> **Note**
>
> 如果您沒有為 `dnf check-update` 命令指定有選用標誌的發行版本，則僅檢查目前設定的儲存庫版本。這表示不會檢查較新版本儲存庫中的套件。

Updates in a specific version

在此範例中，如果我們啟動了 2023.0.20230628 版本的容器，我們將查看 2023.1.20230315 版本中有哪些可用的更新。

> **Note**
>
> 此範例使用 2023.0.20230315 和 2023.1.20230628 版本，這些不是 AL2023 的最新版本。請參閱 AL2023 版本備註以取得最新版本，其中包含最新的安全性更新。

在此範例中，我們將從 2023.0.20230315 版本的容器映像開始。

首先，我們從容器登錄檔擷取此容器映像。結尾.0的 表示特定版本的映像版本；此映像版本通常為零。

```
$ docker pull public.ecr.aws/amazonlinux/amazonlinux:2023.0.20230315.0
   2023.0.20230315.0: Pulling from amazonlinux/amazonlinux
b76f3b09316a: Pull complete
Digest: sha256:94e7183b0739140dbd5b639fb7600f0a2299cec5df8780c26d9cb409da5315a9
Status: Downloaded newer image for public.ecr.aws/amazonlinux/
amazonlinux:2023.0.20230315.0
public.ecr.aws/amazonlinux/amazonlinux:2023.0.20230315.0
```

我們現在可以在容器內產生 shell，並從中檢查更新。

```
$ docker run -it public.ecr.aws/amazonlinux/amazonlinux:2023.0.20230315.0
   bash-5.2#
```

dnf check-update 命令現在用於檢查 2023.1.20230628 版中可用的更新。

> **Note**
>
> 套用套件更新是一項特殊權限操作。雖然在容器中執行 時通常不需要提升權限，但如果在 Amazon EC2 執行個體等非容器化環境中執行 ，則可以在不提升權限的情況下檢查更新。

```
$ dnf check-update --releasever=2023.1.20230628
Amazon Linux 2023 repository                    60 MB/s |  15 MB     00:00
Last metadata expiration check: 0:00:02 ago on Mon Jul 22 17:25:34 2024.

amazon-linux-repo-cdn.noarch        2023.1.20230628-0.amzn2023        amazonlinux
ca-certificates.noarch              2023.2.60-1.0.amzn2023.0.2        amazonlinux
curl-minimal.x86_64                 8.0.1-1.amzn2023                  amazonlinux
glib2.x86_64                        2.74.7-688.amzn2023.0.1           amazonlinux
glibc.x86_64                        2.34-52.amzn2023.0.3              amazonlinux
glibc-common.x86_64                 2.34-52.amzn2023.0.3              amazonlinux
glibc-minimal-langpack.x86_64       2.34-52.amzn2023.0.3              amazonlinux
gnupg2-minimal.x86_64               2.3.7-1.amzn2023.0.4              amazonlinux
keyutils-libs.x86_64                1.6.3-1.amzn2023                  amazonlinux
libcap.x86_64                       2.48-2.amzn2023.0.3               amazonlinux
libcurl-minimal.x86_64              8.0.1-1.amzn2023                  amazonlinux
```

```
libgcc.x86_64                   11.3.1-4.amzn2023.0.3           amazonlinux
libgomp.x86_64                  11.3.1-4.amzn2023.0.3           amazonlinux
libstdc++.x86_64                11.3.1-4.amzn2023.0.3           amazonlinux
libxml2.x86_64                  2.10.4-1.amzn2023.0.1           amazonlinux
ncurses-base.noarch             6.2-4.20200222.amzn2023.0.4     amazonlinux
ncurses-libs.x86_64             6.2-4.20200222.amzn2023.0.4     amazonlinux
openssl-libs.x86_64             1:3.0.8-1.amzn2023.0.3          amazonlinux
python3-rpm.x86_64              4.16.1.3-12.amzn2023.0.6        amazonlinux
rpm.x86_64                      4.16.1.3-12.amzn2023.0.6        amazonlinux
rpm-build-libs.x86_64           4.16.1.3-12.amzn2023.0.6        amazonlinux
rpm-libs.x86_64                 4.16.1.3-12.amzn2023.0.6        amazonlinux
rpm-sign-libs.x86_64            4.16.1.3-12.amzn2023.0.6        amazonlinux
system-release.noarch           2023.1.20230628-0.amzn2023      amazonlinux
tzdata.noarch                   2023c-1.amzn2023.0.1            amazonlinux
bash-5.2#
```

`system-release` 套件的版本顯示`dnf upgrade`命令將更新的版本，這是命令中請求的 2023.1.20230628 版本。 `dnf check-update --releasever=`*`2023.1.20230628`*

Updates in the latest version

在此範例中，如果我們啟動了 2023.4. `latest`版本的容器，我們將查看 AL2023 版本中有哪些可用的更新。 AL2023 20240319 撰寫時，`latest`版本為 2023.5.20240708，因此此範例中列出的更新將截至該版本。

> **Note**
>
> 此範例使用 2023.4.20240319 和 2023.5.20240708 版本，後者是撰寫時的最新版本。如需最新版本的詳細資訊，請參閱 AL2023 版本備註。

在此範例中，我們將從 2023.4.20240319 版的容器映像開始。

首先，我們從容器登錄檔擷取此容器映像。結尾`.1`的 表示特定版本的映像版本。雖然映像版本通常為零，但此範例會使用映像版本為 的版本。

```
$ docker pull public.ecr.aws/amazonlinux/amazonlinux:2023.4.20240319.1
   2023.4.20240319.1: Pulling from amazonlinux/amazonlinux
6de065fda9a2: Pull complete
Digest: sha256:b4838c4cc9211d966b6ea158dacc9eda7433a16ba94436508c2d9f01f7658b4e
Status: Downloaded newer image for public.ecr.aws/amazonlinux/
amazonlinux:2023.4.20240319.1
```

```
public.ecr.aws/amazonlinux/amazonlinux:2023.4.20240319.1
```

我們現在可以在容器內產生 shell，並從中檢查更新。

```
$ docker run -it public.ecr.aws/amazonlinux/amazonlinux:2023.4.20240319.1
   bash-5.2#
```

`dnf check-update` 命令現在用於檢查`latest`發行版本中可用的更新，在寫入時為 2023.5.20240708。

> **Note**
>
> 套用套件更新是一項特殊權限操作。雖然在容器中執行 時通常不需要提升權限，但如果在 Amazon EC2 執行個體等非容器化環境中執行 ，則可以在不提升權限的情況下檢查更新。

```
$ dnf --releasever=latest check-update
Amazon Linux 2023 repository                    78 MB/s |  25 MB     00:00
Last metadata expiration check: 0:00:04 ago on Mon Jul 22 17:39:13 2024.

amazon-linux-repo-cdn.noarch        2023.5.20240708-1.amzn2023        amazonlinux
curl-minimal.x86_64                 8.5.0-1.amzn2023.0.4              amazonlinux
dnf.noarch                          4.14.0-1.amzn2023.0.5             amazonlinux
dnf-data.noarch                     4.14.0-1.amzn2023.0.5             amazonlinux
expat.x86_64                        2.5.0-1.amzn2023.0.4              amazonlinux
glibc.x86_64                        2.34-52.amzn2023.0.10             amazonlinux
glibc-common.x86_64                 2.34-52.amzn2023.0.10             amazonlinux
glibc-minimal-langpack.x86_64       2.34-52.amzn2023.0.10             amazonlinux
krb5-libs.x86_64                    1.21-3.amzn2023.0.4               amazonlinux
libblkid.x86_64                     2.37.4-1.amzn2023.0.4             amazonlinux
libcurl-minimal.x86_64              8.5.0-1.amzn2023.0.4              amazonlinux
libmount.x86_64                     2.37.4-1.amzn2023.0.4             amazonlinux
libnghttp2.x86_64                   1.59.0-3.amzn2023.0.1             amazonlinux
libsmartcols.x86_64                 2.37.4-1.amzn2023.0.4             amazonlinux
libuuid.x86_64                      2.37.4-1.amzn2023.0.4             amazonlinux
openssl-libs.x86_64                 1:3.0.8-1.amzn2023.0.12           amazonlinux
python3.x86_64                      3.9.16-1.amzn2023.0.8             amazonlinux
python3-dnf.noarch                  4.14.0-1.amzn2023.0.5             amazonlinux
python3-libs.x86_64                 3.9.16-1.amzn2023.0.8             amazonlinux
system-release.noarch               2023.5.20240708-1.amzn2023        amazonlinux
yum.noarch                          4.14.0-1.amzn2023.0.5             amazonlinux
```

```
bash-5.2#
```

`system-release` 套件的版本會顯示`dnf upgrade`命令將更新的版本。

對於這個命令，如果有更新的套件可用，傳回碼是 100。如果沒有更新的套件可用，傳回碼是 0。此外，輸出也會列出所有要更新的套件。

使用 DNF 和儲存庫版本套用安全更新

新套件更新和安全更新僅適用於新的儲存庫版本。對於從舊版 AL2023 AMI 啟動的執行個體，您必須先更新儲存庫版本，然後才能安裝安全更新。`dnf check-release-update` 命令包含範例更新命令，可將系統上安裝的所有套件更新為較新儲存庫中的版本。

> **Note**
>
> 如果您沒有為 `dnf check-update` 命令指定有選用標誌的發行版本，則僅檢查目前設定的儲存庫版本。這表示不會套用任何更新版本之儲存庫中已安裝套件的任何更新。

本節涵蓋套用所有可用更新的建議升級路徑，而不是選擇個別更新或僅標記為安全性更新的更新。透過套用所有更新，現有執行個體會移至與啟動更新 AMI 相同的套件集。此一致性可減少機群中套件版本的變化。如需套用特定更新的詳細資訊，請參閱 就地套用安全性更新。

Applying updates in a specific version

在此範例中，如果我們啟動了 2023.0.20230628 版本的容器，我們將套用 2023.1.20230315 版本中可用的更新。

> **Note**
>
> 此範例使用 2023.0.20230315 和 2023.1.20230628 版本，這些不是 AL2023 的最新版本。請參閱 AL2023 版本備註以取得最新版本，其中包含最新的安全性更新。

在此範例中，我們將從 2023.0.20230315 版本的容器映像開始。

首先，我們從容器登錄檔擷取此容器映像。結尾`.0`的 表示特定版本的映像版本；此映像版本通常為零。

```
$ docker pull public.ecr.aws/amazonlinux/amazonlinux:2023.0.20230315.0
```

```
   2023.0.20230315.0: Pulling from amazonlinux/amazonlinux
b76f3b09316a: Pull complete
Digest: sha256:94e7183b0739140dbd5b639fb7600f0a2299cec5df8780c26d9cb409da5315a9
Status: Downloaded newer image for public.ecr.aws/amazonlinux/
amazonlinux:2023.0.20230315.0
public.ecr.aws/amazonlinux/amazonlinux:2023.0.20230315.0
```

我們現在可以在容器內產生殼層，從中套用更新。

```
$ docker run -it public.ecr.aws/amazonlinux/amazonlinux:2023.0.20230315.0
   bash-5.2#
```

dnf upgrade 命令現在用於套用 2023.1.20230628 版中存在的所有更新。

> **Note**
>
> 套用套件更新是一項特殊權限操作。雖然在容器中執行 時通常不需要提升權限，但如果在 Amazon EC2 執行個體等非容器化環境中執行 ，則需要以root使用者身分執行 dnf upgrade命令。這可以使用 sudo或 su命令來完成。

```
$ dnf upgrade --releasever=2023.1.20230628
Amazon Linux 2023 repository                    38 MB/s |  15 MB     00:00
Last metadata expiration check: 0:00:02 ago on Mon Jul 22 17:49:08 2024.
Dependencies resolved.
================================================================================
 Package                 Arch    Version                       Repository    Size
================================================================================
Upgrading:
 amazon-linux-repo-cdn   noarch  2023.1.20230628-0.amzn2023    amazonlinux   18 k
 ca-certificates         noarch  2023.2.60-1.0.amzn2023.0.2    amazonlinux  829 k
 curl-minimal            x86_64  8.0.1-1.amzn2023              amazonlinux  150 k
 glib2                   x86_64  2.74.7-688.amzn2023.0.1       amazonlinux  2.7 M
 glibc                   x86_64  2.34-52.amzn2023.0.3          amazonlinux  1.9 M
 glibc-common            x86_64  2.34-52.amzn2023.0.3          amazonlinux  307 k
 glibc-minimal-langpack  x86_64  2.34-52.amzn2023.0.3          amazonlinux   35 k
 gnupg2-minimal          x86_64  2.3.7-1.amzn2023.0.4          amazonlinux  421 k
 keyutils-libs           x86_64  1.6.3-1.amzn2023              amazonlinux   33 k
 libcap                  x86_64  2.48-2.amzn2023.0.3           amazonlinux   67 k
 libcurl-minimal         x86_64  8.0.1-1.amzn2023              amazonlinux  249 k
 libgcc                  x86_64  11.3.1-4.amzn2023.0.3         amazonlinux  105 k
 libgomp                 x86_64  11.3.1-4.amzn2023.0.3         amazonlinux  280 k
```

```
 libstdc++                x86_64  11.3.1-4.amzn2023.0.3         amazonlinux  744 k
 libxml2                  x86_64  2.10.4-1.amzn2023.0.1         amazonlinux  706 k
 ncurses-base             noarch  6.2-4.20200222.amzn2023.0.4   amazonlinux   60 k
 ncurses-libs             x86_64  6.2-4.20200222.amzn2023.0.4   amazonlinux  328 k
 openssl-libs             x86_64  1:3.0.8-1.amzn2023.0.3        amazonlinux  2.2 M
 python3-rpm              x86_64  4.16.1.3-12.amzn2023.0.6      amazonlinux   88 k
 rpm                      x86_64  4.16.1.3-12.amzn2023.0.6      amazonlinux  486 k
 rpm-build-libs           x86_64  4.16.1.3-12.amzn2023.0.6      amazonlinux   90 k
 rpm-libs                 x86_64  4.16.1.3-12.amzn2023.0.6      amazonlinux  309 k
 rpm-sign-libs            x86_64  4.16.1.3-12.amzn2023.0.6      amazonlinux   21 k
 system-release           noarch  2023.1.20230628-0.amzn2023    amazonlinux   29 k
 tzdata                   noarch  2023c-1.amzn2023.0.1          amazonlinux  433 k

Transaction Summary
================================================================================
Upgrade  25 Packages

Total download size: 12 M
Is this ok [y/N]:
```

system-release 套件的版本顯示dnf upgrade命令將更新的版本，這是命令中請求的 2023.1.20230628 版本。 dnf upgrade --releasever=*2023.1.20230628*

根據預設，dnf 會要求您確認是否要套用更新。您可以使用 -y旗標略過此提示dnf。在此範例中，dnf upgrade -y --releasever=*2023.1.20230628*命令不會在套用更新之前要求確認。這在指令碼或其他自動化環境中非常有用。

一旦確認您要套用更新，就會dnf套用更新。

```
Is this ok [y/N]:y
   Downloading Packages:
(1/25): libcap-2.48-2.amzn2023.0.3.x86_64.rpm    1.5 MB/s |  67 kB     00:00
(2/25): python3-rpm-4.16.1.3-12.amzn2023.0.6.x86 2.1 MB/s |  88 kB     00:00
(3/25): libcurl-minimal-8.0.1-1.amzn2023.x86_64. 2.6 MB/s | 249 kB     00:00
(4/25): glib2-2.74.7-688.amzn2023.0.1.x86_64.rpm  26 MB/s | 2.7 MB     00:00
(5/25): glibc-minimal-langpack-2.34-52.amzn2023. 1.3 MB/s |  35 kB     00:00
(6/25): rpm-build-libs-4.16.1.3-12.amzn2023.0.6. 2.8 MB/s |  90 kB     00:00
(7/25): rpm-libs-4.16.1.3-12.amzn2023.0.6.x86_64 6.6 MB/s | 309 kB     00:00
(8/25): libgcc-11.3.1-4.amzn2023.0.3.x86_64.rpm  3.9 MB/s | 105 kB     00:00
(9/25): glibc-common-2.34-52.amzn2023.0.3.x86_64  11 MB/s | 307 kB     00:00
(10/25): glibc-2.34-52.amzn2023.0.3.x86_64.rpm    31 MB/s | 1.9 MB     00:00
(11/25): rpm-sign-libs-4.16.1.3-12.amzn2023.0.6. 877 kB/s |  21 kB     00:00
(12/25): gnupg2-minimal-2.3.7-1.amzn2023.0.4.x86  15 MB/s | 421 kB     00:00
```

```
(13/25): openssl-libs-3.0.8-1.amzn2023.0.3.x86_6  35 MB/s | 2.2 MB     00:00
(14/25): libxml2-2.10.4-1.amzn2023.0.1.x86_64.rp  14 MB/s | 706 kB     00:00
(15/25): curl-minimal-8.0.1-1.amzn2023.x86_64.rp 4.2 MB/s | 150 kB     00:00
(16/25): rpm-4.16.1.3-12.amzn2023.0.6.x86_64.rpm  11 MB/s | 486 kB     00:00
(17/25): libgomp-11.3.1-4.amzn2023.0.3.x86_64.rp 7.0 MB/s | 280 kB     00:00
(18/25): libstdc++-11.3.1-4.amzn2023.0.3.x86_64.  14 MB/s | 744 kB     00:00
(19/25): keyutils-libs-1.6.3-1.amzn2023.x86_64.r 1.6 MB/s |  33 kB     00:00
(20/25): ncurses-libs-6.2-4.20200222.amzn2023.0.  10 MB/s | 328 kB     00:00
(21/25): tzdata-2023c-1.amzn2023.0.1.noarch.rpm   11 MB/s | 433 kB     00:00
(22/25): amazon-linux-repo-cdn-2023.1.20230628-0 781 kB/s |  18 kB     00:00
(23/25): ca-certificates-2023.2.60-1.0.amzn2023.  16 MB/s | 829 kB     00:00
(24/25): system-release-2023.1.20230628-0.amzn20 1.5 MB/s |  29 kB     00:00
(25/25): ncurses-base-6.2-4.20200222.amzn2023.0. 3.1 MB/s |  60 kB     00:00
--------------------------------------------------------------------------------
Total                                            28 MB/s |  12 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Upgrading        : libgcc-11.3.1-4.amzn2023.0.3.x86_64                   1/50
  Running scriptlet: libgcc-11.3.1-4.amzn2023.0.3.x86_64                   1/50
  Upgrading        : system-release-2023.1.20230628-0.amzn2023.noarch       2/50
  Upgrading        : amazon-linux-repo-cdn-2023.1.20230628-0.amzn2023.no    3/50
  Upgrading        : ncurses-base-6.2-4.20200222.amzn2023.0.4.noarch        4/50
  Upgrading        : tzdata-2023c-1.amzn2023.0.1.noarch                     5/50
  Upgrading        : glibc-common-2.34-52.amzn2023.0.3.x86_64               6/50
  Running scriptlet: glibc-2.34-52.amzn2023.0.3.x86_64                      7/50
  Upgrading        : glibc-2.34-52.amzn2023.0.3.x86_64                      7/50
  Running scriptlet: glibc-2.34-52.amzn2023.0.3.x86_64                      7/50
  Upgrading        : glibc-minimal-langpack-2.34-52.amzn2023.0.3.x86_64     8/50
  Upgrading        : libcap-2.48-2.amzn2023.0.3.x86_64                      9/50
  Upgrading        : gnupg2-minimal-2.3.7-1.amzn2023.0.4.x86_64            10/50
  Upgrading        : libgomp-11.3.1-4.amzn2023.0.3.x86_64                  11/50
  Running scriptlet: ca-certificates-2023.2.60-1.0.amzn2023.0.2.noarch     12/50
  Upgrading        : ca-certificates-2023.2.60-1.0.amzn2023.0.2.noarch     12/50
  Running scriptlet: ca-certificates-2023.2.60-1.0.amzn2023.0.2.noarch     12/50
  Upgrading        : openssl-libs-1:3.0.8-1.amzn2023.0.3.x86_64            13/50
  Upgrading        : libcurl-minimal-8.0.1-1.amzn2023.x86_64               14/50
  Upgrading        : curl-minimal-8.0.1-1.amzn2023.x86_64                  15/50
  Upgrading        : rpm-libs-4.16.1.3-12.amzn2023.0.6.x86_64              16/50
  Upgrading        : rpm-4.16.1.3-12.amzn2023.0.6.x86_64                   17/50
  Upgrading        : rpm-build-libs-4.16.1.3-12.amzn2023.0.6.x86_64        18/50
```

```
  Upgrading        : rpm-sign-libs-4.16.1.3-12.amzn2023.0.6.x86_64          19/50
  Upgrading        : python3-rpm-4.16.1.3-12.amzn2023.0.6.x86_64            20/50
  Upgrading        : glib2-2.74.7-688.amzn2023.0.1.x86_64                   21/50
  Upgrading        : libxml2-2.10.4-1.amzn2023.0.1.x86_64                   22/50
  Upgrading        : libstdc++-11.3.1-4.amzn2023.0.3.x86_64                 23/50
  Upgrading        : keyutils-libs-1.6.3-1.amzn2023.x86_64                  24/50
  Upgrading        : ncurses-libs-6.2-4.20200222.amzn2023.0.4.x86_64        25/50
  Cleanup          : glib2-2.73.2-680.amzn2023.0.3.x86_64                   26/50
  Cleanup          : libstdc++-11.3.1-4.amzn2023.0.2.x86_64                 27/50
  Cleanup          : libxml2-2.10.3-2.amzn2023.0.1.x86_64                   28/50
  Cleanup          : python3-rpm-4.16.1.3-12.amzn2023.0.5.x86_64            29/50
  Cleanup          : rpm-build-libs-4.16.1.3-12.amzn2023.0.5.x86_64         30/50
  Cleanup          : rpm-sign-libs-4.16.1.3-12.amzn2023.0.5.x86_64          31/50
  Cleanup          : rpm-libs-4.16.1.3-12.amzn2023.0.5.x86_64               32/50
  Cleanup          : libcap-2.48-2.amzn2023.0.2.x86_64                      33/50
  Cleanup          : gnupg2-minimal-2.3.7-1.amzn2023.0.3.x86_64             34/50
  Cleanup          : ncurses-libs-6.2-4.20200222.amzn2023.0.3.x86_64        35/50
  Cleanup          : libgomp-11.3.1-4.amzn2023.0.2.x86_64                   36/50
  Cleanup          : rpm-4.16.1.3-12.amzn2023.0.5.x86_64                    37/50
  Cleanup          : curl-minimal-7.88.1-1.amzn2023.0.1.x86_64              38/50
  Cleanup          : libcurl-minimal-7.88.1-1.amzn2023.0.1.x86_64           39/50
  Cleanup          : openssl-libs-1:3.0.8-1.amzn2023.0.1.x86_64             40/50
  Cleanup          : keyutils-libs-1.6.1-2.amzn2023.0.2.x86_64              41/50
  Cleanup          : amazon-linux-repo-cdn-2023.0.20230315-1.amzn2023.no    42/50
  Cleanup          : system-release-2023.0.20230315-1.amzn2023.noarch       43/50
  Cleanup          : ca-certificates-2023.2.60-1.0.amzn2023.0.1.noarch      44/50
  Cleanup          : ncurses-base-6.2-4.20200222.amzn2023.0.3.noarch        45/50
  Cleanup          : glibc-minimal-langpack-2.34-52.amzn2023.0.2.x86_64     46/50
  Cleanup          : glibc-2.34-52.amzn2023.0.2.x86_64                      47/50
  Cleanup          : glibc-common-2.34-52.amzn2023.0.2.x86_64               48/50
  Cleanup          : tzdata-2022g-1.amzn2023.0.1.noarch                     49/50
  Cleanup          : libgcc-11.3.1-4.amzn2023.0.2.x86_64                    50/50
Running scriptlet: libgcc-11.3.1-4.amzn2023.0.2.x86_64                    50/50
Running scriptlet: ca-certificates-2023.2.60-1.0.amzn2023.0.2.noarch       50/50
Running scriptlet: rpm-4.16.1.3-12.amzn2023.0.6.x86_64                     50/50
Running scriptlet: libgcc-11.3.1-4.amzn2023.0.2.x86_64                    50/50
  Verifying        : libcurl-minimal-8.0.1-1.amzn2023.x86_64                 1/50
  Verifying        : libcurl-minimal-7.88.1-1.amzn2023.0.1.x86_64            2/50
  Verifying        : libcap-2.48-2.amzn2023.0.3.x86_64                       3/50
  Verifying        : libcap-2.48-2.amzn2023.0.2.x86_64                       4/50
  Verifying        : glib2-2.74.7-688.amzn2023.0.1.x86_64                    5/50
  Verifying        : glib2-2.73.2-680.amzn2023.0.3.x86_64                    6/50
  Verifying        : python3-rpm-4.16.1.3-12.amzn2023.0.6.x86_64             7/50
  Verifying        : python3-rpm-4.16.1.3-12.amzn2023.0.5.x86_64             8/50
```

```
  Verifying        : glibc-minimal-langpack-2.34-52.amzn2023.0.3.x86_64     9/50
  Verifying        : glibc-minimal-langpack-2.34-52.amzn2023.0.2.x86_64    10/50
  Verifying        : rpm-libs-4.16.1.3-12.amzn2023.0.6.x86_64              11/50
  Verifying        : rpm-libs-4.16.1.3-12.amzn2023.0.5.x86_64              12/50
  Verifying        : rpm-build-libs-4.16.1.3-12.amzn2023.0.6.x86_64        13/50
  Verifying        : rpm-build-libs-4.16.1.3-12.amzn2023.0.5.x86_64        14/50
  Verifying        : glibc-2.34-52.amzn2023.0.3.x86_64                     15/50
  Verifying        : glibc-2.34-52.amzn2023.0.2.x86_64                     16/50
  Verifying        : libgcc-11.3.1-4.amzn2023.0.3.x86_64                   17/50
  Verifying        : libgcc-11.3.1-4.amzn2023.0.2.x86_64                   18/50
  Verifying        : glibc-common-2.34-52.amzn2023.0.3.x86_64              19/50
  Verifying        : glibc-common-2.34-52.amzn2023.0.2.x86_64              20/50
  Verifying        : rpm-sign-libs-4.16.1.3-12.amzn2023.0.6.x86_64         21/50
  Verifying        : rpm-sign-libs-4.16.1.3-12.amzn2023.0.5.x86_64         22/50
  Verifying        : openssl-libs-1:3.0.8-1.amzn2023.0.3.x86_64            23/50
  Verifying        : openssl-libs-1:3.0.8-1.amzn2023.0.1.x86_64            24/50
  Verifying        : gnupg2-minimal-2.3.7-1.amzn2023.0.4.x86_64            25/50
  Verifying        : gnupg2-minimal-2.3.7-1.amzn2023.0.3.x86_64            26/50
  Verifying        : libxml2-2.10.4-1.amzn2023.0.1.x86_64                  27/50
  Verifying        : libxml2-2.10.3-2.amzn2023.0.1.x86_64                  28/50
  Verifying        : curl-minimal-8.0.1-1.amzn2023.x86_64                  29/50
  Verifying        : curl-minimal-7.88.1-1.amzn2023.0.1.x86_64             30/50
  Verifying        : rpm-4.16.1.3-12.amzn2023.0.6.x86_64                   31/50
  Verifying        : rpm-4.16.1.3-12.amzn2023.0.5.x86_64                   32/50
  Verifying        : libstdc++-11.3.1-4.amzn2023.0.3.x86_64                33/50
  Verifying        : libstdc++-11.3.1-4.amzn2023.0.2.x86_64                34/50
  Verifying        : libgomp-11.3.1-4.amzn2023.0.3.x86_64                  35/50
  Verifying        : libgomp-11.3.1-4.amzn2023.0.2.x86_64                  36/50
  Verifying        : keyutils-libs-1.6.3-1.amzn2023.x86_64                 37/50
  Verifying        : keyutils-libs-1.6.1-2.amzn2023.0.2.x86_64             38/50
  Verifying        : ncurses-libs-6.2-4.20200222.amzn2023.0.4.x86_64       39/50
  Verifying        : ncurses-libs-6.2-4.20200222.amzn2023.0.3.x86_64       40/50
  Verifying        : ca-certificates-2023.2.60-1.0.amzn2023.0.2.noarch     41/50
  Verifying        : ca-certificates-2023.2.60-1.0.amzn2023.0.1.noarch     42/50
  Verifying        : tzdata-2023c-1.amzn2023.0.1.noarch                    43/50
  Verifying        : tzdata-2022g-1.amzn2023.0.1.noarch                    44/50
  Verifying        : amazon-linux-repo-cdn-2023.1.20230628-0.amzn2023.no   45/50
  Verifying        : amazon-linux-repo-cdn-2023.0.20230315-1.amzn2023.no   46/50
  Verifying        : system-release-2023.1.20230628-0.amzn2023.noarch       47/50
  Verifying        : system-release-2023.0.20230315-1.amzn2023.noarch       48/50
  Verifying        : ncurses-base-6.2-4.20200222.amzn2023.0.4.noarch       49/50
  Verifying        : ncurses-base-6.2-4.20200222.amzn2023.0.3.noarch       50/50

Upgraded:
```

```
  amazon-linux-repo-cdn-2023.1.20230628-0.amzn2023.noarch
  ca-certificates-2023.2.60-1.0.amzn2023.0.2.noarch
  curl-minimal-8.0.1-1.amzn2023.x86_64
  glib2-2.74.7-688.amzn2023.0.1.x86_64
  glibc-2.34-52.amzn2023.0.3.x86_64
  glibc-common-2.34-52.amzn2023.0.3.x86_64
  glibc-minimal-langpack-2.34-52.amzn2023.0.3.x86_64
  gnupg2-minimal-2.3.7-1.amzn2023.0.4.x86_64
  keyutils-libs-1.6.3-1.amzn2023.x86_64
  libcap-2.48-2.amzn2023.0.3.x86_64
  libcurl-minimal-8.0.1-1.amzn2023.x86_64
  libgcc-11.3.1-4.amzn2023.0.3.x86_64
  libgomp-11.3.1-4.amzn2023.0.3.x86_64
  libstdc++-11.3.1-4.amzn2023.0.3.x86_64
  libxml2-2.10.4-1.amzn2023.0.1.x86_64
  ncurses-base-6.2-4.20200222.amzn2023.0.4.noarch
  ncurses-libs-6.2-4.20200222.amzn2023.0.4.x86_64
  openssl-libs-1:3.0.8-1.amzn2023.0.3.x86_64
  python3-rpm-4.16.1.3-12.amzn2023.0.6.x86_64
  rpm-4.16.1.3-12.amzn2023.0.6.x86_64
  rpm-build-libs-4.16.1.3-12.amzn2023.0.6.x86_64
  rpm-libs-4.16.1.3-12.amzn2023.0.6.x86_64
  rpm-sign-libs-4.16.1.3-12.amzn2023.0.6.x86_64
  system-release-2023.1.20230628-0.amzn2023.noarch
  tzdata-2023c-1.amzn2023.0.1.noarch

  Complete!
bash-5.2#
```

Updates in the latest version

在此範例中，如果我們啟動了 2023.4. `latest`版本的容器，我們將套用 AL2023 版本的可用更新。AL2023 20240319 撰寫時，`latest`版本為 2023.5.20240708，因此此範例中列出的更新將截至該版本。

> **Note**
>
> 此範例使用 2023.4.20240319 和 2023.5.20240708 版本，後者是撰寫時的最新版本。如需最新版本的詳細資訊，請參閱 AL2023 版本備註。

在此範例中，我們將從 2023.4.20240319 版本的容器映像開始。

首先，我們從容器登錄檔擷取此容器映像。結尾.1的 表示特定版本的映像版本。雖然映像版本通常為零，但此範例會使用映像版本為 的版本。

```
$ docker pull public.ecr.aws/amazonlinux/amazonlinux:2023.4.20240319.1
   2023.4.20240319.1: Pulling from amazonlinux/amazonlinux
6de065fda9a2: Pull complete
Digest: sha256:b4838c4cc9211d966b6ea158dacc9eda7433a16ba94436508c2d9f01f7658b4e
Status: Downloaded newer image for public.ecr.aws/amazonlinux/
amazonlinux:2023.4.20240319.1
public.ecr.aws/amazonlinux/amazonlinux:2023.4.20240319.1
```

我們現在可以在容器內產生殼層，從中套用更新。

```
$ docker run -it public.ecr.aws/amazonlinux/amazonlinux:2023.4.20240319.1
   bash-5.2#
```

dnf upgrade 命令現在用於套用latest發行版本中可用的更新，在寫入時為 2023.5.20240708。

> **Note**
>
> 套用套件更新是一項特殊權限操作。雖然在容器中執行 時通常不需要提升權限，但如果在 Amazon EC2 執行個體等非容器化環境中執行 ，則需要以root使用者身分執行 dnf upgrade命令。這可以使用 sudo或 su命令來完成。

根據預設， dnf 會要求您確認是否要套用更新。在此範例中，我們將使用 -y旗標來略過此提示dnf。

```
$ dnf -y --releasever=latest update
Amazon Linux 2023 repository                    75 MB/s |  25 MB     00:00
Last metadata expiration check: 0:00:04 ago on Mon Jul 22 18:00:10 2024.
Dependencies resolved.
================================================================================
 Package                 Arch    Version                      Repository    Size
================================================================================
Upgrading:
 amazon-linux-repo-cdn   noarch  2023.5.20240708-1.amzn2023   amazonlinux   17 k
 curl-minimal            x86_64  8.5.0-1.amzn2023.0.4         amazonlinux  160 k
 dnf                     noarch  4.14.0-1.amzn2023.0.5        amazonlinux  460 k
 dnf-data                noarch  4.14.0-1.amzn2023.0.5        amazonlinux   34 k
```

```
 expat                   x86_64  2.5.0-1.amzn2023.0.4         amazonlinux  117 k
 glibc                   x86_64  2.34-52.amzn2023.0.10        amazonlinux  1.9 M
 glibc-common            x86_64  2.34-52.amzn2023.0.10        amazonlinux  295 k
 glibc-minimal-langpack  x86_64  2.34-52.amzn2023.0.10        amazonlinux   23 k
 krb5-libs               x86_64  1.21-3.amzn2023.0.4          amazonlinux  758 k
 libblkid                x86_64  2.37.4-1.amzn2023.0.4        amazonlinux  105 k
 libcurl-minimal         x86_64  8.5.0-1.amzn2023.0.4         amazonlinux  275 k
 libmount                x86_64  2.37.4-1.amzn2023.0.4        amazonlinux  132 k
 libnghttp2              x86_64  1.59.0-3.amzn2023.0.1        amazonlinux   79 k
 libsmartcols            x86_64  2.37.4-1.amzn2023.0.4        amazonlinux   62 k
 libuuid                 x86_64  2.37.4-1.amzn2023.0.4        amazonlinux   26 k
 openssl-libs            x86_64  1:3.0.8-1.amzn2023.0.12      amazonlinux  2.2 M
 python3                 x86_64  3.9.16-1.amzn2023.0.8        amazonlinux   27 k
 python3-dnf             noarch  4.14.0-1.amzn2023.0.5        amazonlinux  409 k
 python3-libs            x86_64  3.9.16-1.amzn2023.0.8        amazonlinux  7.3 M
 system-release          noarch  2023.5.20240708-1.amzn2023   amazonlinux   28 k
 yum                     noarch  4.14.0-1.amzn2023.0.5        amazonlinux   32 k

 Transaction Summary
================================================================================
Upgrade  21 Packages

Total download size: 14 M
Downloading Packages:
(1/21): amazon-linux-repo-cdn-2023.5.20240708-1. 345 kB/s |  17 kB     00:00
(2/21): dnf-4.14.0-1.amzn2023.0.5.noarch.rpm     6.8 MB/s | 460 kB     00:00
(3/21): dnf-data-4.14.0-1.amzn2023.0.5.noarch.rp 1.6 MB/s |  34 kB     00:00
(4/21): expat-2.5.0-1.amzn2023.0.4.x86_64.rpm    4.6 MB/s | 117 kB     00:00
(5/21): glibc-2.34-52.amzn2023.0.10.x86_64.rpm    38 MB/s | 1.9 MB     00:00
(6/21): glibc-common-2.34-52.amzn2023.0.10.x86_6 8.8 MB/s | 295 kB     00:00
(7/21): glibc-minimal-langpack-2.34-52.amzn2023. 1.7 MB/s |  23 kB     00:00
(8/21): curl-minimal-8.5.0-1.amzn2023.0.4.x86_64 998 kB/s | 160 kB     00:00
(9/21): libblkid-2.37.4-1.amzn2023.0.4.x86_64.rp 4.1 MB/s | 105 kB     00:00
(10/21): krb5-libs-1.21-3.amzn2023.0.4.x86_64.rp  16 MB/s | 758 kB     00:00
(11/21): libmount-2.37.4-1.amzn2023.0.4.x86_64.r 7.9 MB/s | 132 kB     00:00
(12/21): libnghttp2-1.59.0-3.amzn2023.0.1.x86_64 5.6 MB/s |  79 kB     00:00
(13/21): libsmartcols-2.37.4-1.amzn2023.0.4.x86_ 4.4 MB/s |  62 kB     00:00
(14/21): libcurl-minimal-8.5.0-1.amzn2023.0.4.x8 7.1 MB/s | 275 kB     00:00
(15/21): libuuid-2.37.4-1.amzn2023.0.4.x86_64.rp 1.1 MB/s |  26 kB     00:00
(16/21): python3-3.9.16-1.amzn2023.0.8.x86_64.rp 1.5 MB/s |  27 kB     00:00
(17/21): python3-dnf-4.14.0-1.amzn2023.0.5.noarc  19 MB/s | 409 kB     00:00
(18/21): system-release-2023.5.20240708-1.amzn20 1.9 MB/s |  28 kB     00:00
(19/21): yum-4.14.0-1.amzn2023.0.5.noarch.rpm    1.6 MB/s |  32 kB     00:00
(20/21): openssl-libs-3.0.8-1.amzn2023.0.12.x86_  26 MB/s | 2.2 MB     00:00
```

```
(21/21): python3-libs-3.9.16-1.amzn2023.0.8.x86_  59 MB/s | 7.3 MB     00:00
--------------------------------------------------------------------------------
Total                                             34 MB/s |  14 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Upgrading        : glibc-common-2.34-52.amzn2023.0.10.x86_64              1/42
  Upgrading        : glibc-minimal-langpack-2.34-52.amzn2023.0.10.x86_64    2/42
  Running scriptlet: glibc-2.34-52.amzn2023.0.10.x86_64                     3/42
  Upgrading        : glibc-2.34-52.amzn2023.0.10.x86_64                     3/42
  Running scriptlet: glibc-2.34-52.amzn2023.0.10.x86_64                     3/42
  Upgrading        : libuuid-2.37.4-1.amzn2023.0.4.x86_64                   4/42
  Upgrading        : openssl-libs-1:3.0.8-1.amzn2023.0.12.x86_64            5/42
  Upgrading        : krb5-libs-1.21-3.amzn2023.0.4.x86_64                   6/42
  Upgrading        : libblkid-2.37.4-1.amzn2023.0.4.x86_64                  7/42
  Running scriptlet: libblkid-2.37.4-1.amzn2023.0.4.x86_64                  7/42
  Upgrading        : expat-2.5.0-1.amzn2023.0.4.x86_64                      8/42
  Upgrading        : python3-3.9.16-1.amzn2023.0.8.x86_64                   9/42
  Upgrading        : python3-libs-3.9.16-1.amzn2023.0.8.x86_64             10/42
  Upgrading        : libnghttp2-1.59.0-3.amzn2023.0.1.x86_64               11/42
  Upgrading        : libcurl-minimal-8.5.0-1.amzn2023.0.4.x86_64           12/42
  Upgrading        : system-release-2023.5.20240708-1.amzn2023.noarch      13/42
  Upgrading        : amazon-linux-repo-cdn-2023.5.20240708-1.amzn2023.no   14/42
  Upgrading        : dnf-data-4.14.0-1.amzn2023.0.5.noarch                 15/42
  Upgrading        : python3-dnf-4.14.0-1.amzn2023.0.5.noarch              16/42
  Upgrading        : dnf-4.14.0-1.amzn2023.0.5.noarch                      17/42
  Running scriptlet: dnf-4.14.0-1.amzn2023.0.5.noarch                      17/42
  Upgrading        : yum-4.14.0-1.amzn2023.0.5.noarch                      18/42
  Upgrading        : curl-minimal-8.5.0-1.amzn2023.0.4.x86_64              19/42
  Upgrading        : libmount-2.37.4-1.amzn2023.0.4.x86_64                 20/42
  Upgrading        : libsmartcols-2.37.4-1.amzn2023.0.4.x86_64             21/42
  Cleanup          : yum-4.14.0-1.amzn2023.0.4.noarch                      22/42
  Running scriptlet: dnf-4.14.0-1.amzn2023.0.4.noarch                      23/42
  Cleanup          : dnf-4.14.0-1.amzn2023.0.4.noarch                      23/42
  Running scriptlet: dnf-4.14.0-1.amzn2023.0.4.noarch                      23/42
  Cleanup          : python3-dnf-4.14.0-1.amzn2023.0.4.noarch              24/42
  Cleanup          : amazon-linux-repo-cdn-2023.4.20240319-1.amzn2023.no   25/42
  Cleanup          : libmount-2.37.4-1.amzn2023.0.3.x86_64                 26/42
  Cleanup          : curl-minimal-8.5.0-1.amzn2023.0.2.x86_64              27/42
  Cleanup          : libcurl-minimal-8.5.0-1.amzn2023.0.2.x86_64           28/42
  Cleanup          : krb5-libs-1.21-3.amzn2023.0.3.x86_64                  29/42
```

```
  Cleanup          : libblkid-2.37.4-1.amzn2023.0.3.x86_64                  30/42
  Cleanup          : libnghttp2-1.57.0-1.amzn2023.0.1.x86_64                31/42
  Cleanup          : libsmartcols-2.37.4-1.amzn2023.0.3.x86_64              32/42
  Cleanup          : system-release-2023.4.20240319-1.amzn2023.noarch       33/42
  Cleanup          : dnf-data-4.14.0-1.amzn2023.0.4.noarch                  34/42
  Cleanup          : python3-3.9.16-1.amzn2023.0.6.x86_64                   35/42
  Cleanup          : python3-libs-3.9.16-1.amzn2023.0.6.x86_64              36/42
  Cleanup          : openssl-libs-1:3.0.8-1.amzn2023.0.11.x86_64            37/42
  Cleanup          : libuuid-2.37.4-1.amzn2023.0.3.x86_64                   38/42
  Cleanup          : expat-2.5.0-1.amzn2023.0.3.x86_64                      39/42
  Cleanup          : glibc-2.34-52.amzn2023.0.8.x86_64                      40/42
  Cleanup          : glibc-minimal-langpack-2.34-52.amzn2023.0.8.x86_64     41/42
  Cleanup          : glibc-common-2.34-52.amzn2023.0.8.x86_64               42/42
  Running scriptlet: glibc-common-2.34-52.amzn2023.0.8.x86_64               42/42
  Verifying        : amazon-linux-repo-cdn-2023.5.20240708-1.amzn2023.no     1/42
  Verifying        : amazon-linux-repo-cdn-2023.4.20240319-1.amzn2023.no     2/42
  Verifying        : curl-minimal-8.5.0-1.amzn2023.0.4.x86_64                3/42
  Verifying        : curl-minimal-8.5.0-1.amzn2023.0.2.x86_64                4/42
  Verifying        : dnf-4.14.0-1.amzn2023.0.5.noarch                        5/42
  Verifying        : dnf-4.14.0-1.amzn2023.0.4.noarch                        6/42
  Verifying        : dnf-data-4.14.0-1.amzn2023.0.5.noarch                   7/42
  Verifying        : dnf-data-4.14.0-1.amzn2023.0.4.noarch                   8/42
  Verifying        : expat-2.5.0-1.amzn2023.0.4.x86_64                       9/42
  Verifying        : expat-2.5.0-1.amzn2023.0.3.x86_64                      10/42
  Verifying        : glibc-2.34-52.amzn2023.0.10.x86_64                     11/42
  Verifying        : glibc-2.34-52.amzn2023.0.8.x86_64                      12/42
  Verifying        : glibc-common-2.34-52.amzn2023.0.10.x86_64              13/42
  Verifying        : glibc-common-2.34-52.amzn2023.0.8.x86_64               14/42
  Verifying        : glibc-minimal-langpack-2.34-52.amzn2023.0.10.x86_64    15/42
  Verifying        : glibc-minimal-langpack-2.34-52.amzn2023.0.8.x86_64     16/42
  Verifying        : krb5-libs-1.21-3.amzn2023.0.4.x86_64                   17/42
  Verifying        : krb5-libs-1.21-3.amzn2023.0.3.x86_64                   18/42
  Verifying        : libblkid-2.37.4-1.amzn2023.0.4.x86_64                  19/42
  Verifying        : libblkid-2.37.4-1.amzn2023.0.3.x86_64                  20/42
  Verifying        : libcurl-minimal-8.5.0-1.amzn2023.0.4.x86_64            21/42
  Verifying        : libcurl-minimal-8.5.0-1.amzn2023.0.2.x86_64            22/42
  Verifying        : libmount-2.37.4-1.amzn2023.0.4.x86_64                  23/42
  Verifying        : libmount-2.37.4-1.amzn2023.0.3.x86_64                  24/42
  Verifying        : libnghttp2-1.59.0-3.amzn2023.0.1.x86_64                25/42
  Verifying        : libnghttp2-1.57.0-1.amzn2023.0.1.x86_64                26/42
  Verifying        : libsmartcols-2.37.4-1.amzn2023.0.4.x86_64              27/42
  Verifying        : libsmartcols-2.37.4-1.amzn2023.0.3.x86_64              28/42
  Verifying        : libuuid-2.37.4-1.amzn2023.0.4.x86_64                   29/42
  Verifying        : libuuid-2.37.4-1.amzn2023.0.3.x86_64                   30/42
```

```
  Verifying        : openssl-libs-1:3.0.8-1.amzn2023.0.12.x86_64           31/42
  Verifying        : openssl-libs-1:3.0.8-1.amzn2023.0.11.x86_64           32/42
  Verifying        : python3-3.9.16-1.amzn2023.0.8.x86_64                  33/42
  Verifying        : python3-3.9.16-1.amzn2023.0.6.x86_64                  34/42
  Verifying        : python3-dnf-4.14.0-1.amzn2023.0.5.noarch              35/42
  Verifying        : python3-dnf-4.14.0-1.amzn2023.0.4.noarch              36/42
  Verifying        : python3-libs-3.9.16-1.amzn2023.0.8.x86_64             37/42
  Verifying        : python3-libs-3.9.16-1.amzn2023.0.6.x86_64             38/42
  Verifying        : system-release-2023.5.20240708-1.amzn2023.noarch      39/42
  Verifying        : system-release-2023.4.20240319-1.amzn2023.noarch      40/42
  Verifying        : yum-4.14.0-1.amzn2023.0.5.noarch                      41/42
  Verifying        : yum-4.14.0-1.amzn2023.0.4.noarch                      42/42

Upgraded:
  amazon-linux-repo-cdn-2023.5.20240708-1.amzn2023.noarch
  curl-minimal-8.5.0-1.amzn2023.0.4.x86_64
  dnf-4.14.0-1.amzn2023.0.5.noarch
  dnf-data-4.14.0-1.amzn2023.0.5.noarch
  expat-2.5.0-1.amzn2023.0.4.x86_64
  glibc-2.34-52.amzn2023.0.10.x86_64
  glibc-common-2.34-52.amzn2023.0.10.x86_64
  glibc-minimal-langpack-2.34-52.amzn2023.0.10.x86_64
  krb5-libs-1.21-3.amzn2023.0.4.x86_64
  libblkid-2.37.4-1.amzn2023.0.4.x86_64
  libcurl-minimal-8.5.0-1.amzn2023.0.4.x86_64
  libmount-2.37.4-1.amzn2023.0.4.x86_64
  libnghttp2-1.59.0-3.amzn2023.0.1.x86_64
  libsmartcols-2.37.4-1.amzn2023.0.4.x86_64
  libuuid-2.37.4-1.amzn2023.0.4.x86_64
  openssl-libs-1:3.0.8-1.amzn2023.0.12.x86_64
  python3-3.9.16-1.amzn2023.0.8.x86_64
  python3-dnf-4.14.0-1.amzn2023.0.5.noarch
  python3-libs-3.9.16-1.amzn2023.0.8.x86_64
  system-release-2023.5.20240708-1.amzn2023.noarch
  yum-4.14.0-1.amzn2023.0.5.noarch

Complete!
bash-5.2#
```

若要探索 AL2023 更新，請執行下列一或多個動作：

- 執行 dnf check-update 命令。這會檢查您鎖定的 Amazon Linux 版本中是否有任何未套用的更新。如果您只更新system-release套件，這可能會顯示更新，移動執行個體鎖定的儲存庫版本，但不套用其中可用的任何更新。
- 訂閱 Amazon Linux 儲存庫更新 SNS 主題 (arn:aws:sns:us-east-1:137112412989:amazon-linux-2023-ami-updates)。如需詳細資訊，請參閱《Amazon Simple Notification Service 開發人員指南》中的訂閱 Amazon SNS 主題。
- 定期參閱 AL2023 版本備註。
- 透過 探索新版本使用 檢查較新的儲存庫版本 dnf check-release-update。

> **Important**
>
> 包含安全性更新的 AL2023 新版本會經常發行。請務必隨時掌握相關的安全修補程式。

在（安全性）更新後自動重新啟動服務

Amazon Linux 現在隨附智慧型重新啟動套件。每當使用系統套件管理員安裝或刪除套件時， 會在系統更新時Smart-restart重新啟動系統服務。只要執行 dnf (update|upgrade|downgrade)，就會發生這種情況。

Smart-restart 使用來自 的 needs-restarting套件dnf-utils和自訂拒絕清單機制，來判斷需要重新啟動哪些服務，以及是否建議重新啟動系統。如果建議重新啟動系統，則會產生重新啟動提示標記檔案 (/run/smart-restart/reboot-hint-marker)。

安裝 **smart-restart**

執行下列DNF命令（如同使用任何其他套件）。

```
$ sudo dnf install smart-restart
```

安裝之後，後續交易會觸發smart-restart邏輯。

拒絕清單

Smart-restart 可指示 封鎖特定服務重新啟動。封鎖的服務不會導致決定是否需要重新啟動。若要封鎖其他服務，請在 -denylist中新增尾碼為 的檔案/etc/smart-restart-conf.d/，如下列範例所示。

```
$ cat /etc/smart-restart-conf.d/custom-denylist
# Some comments
myservice.service
```

> **Note**
>
> 在決定是否需要重新啟動時，會讀取和評估所有*-denylist檔案。

自訂掛鉤

除了拒絕清單之外， smart-restart還提供在嘗試重新啟動服務之前和之後執行自訂指令碼的機制。自訂指令碼可用來手動執行準備步驟，或通知其他元件剩餘或已完成的重新啟動。

執行 中/etc/smart-restart-conf.d/具有尾碼 -pre-restart或 的所有指令碼-post-restart。如果順序很重要，請在所有指令碼前面加上數字，以確保執行順序，如下列範例所示。

```
$ ls /etc/smart-restart-conf.d/*-pre-restart
001-my-script-pre-restart
002-some-other-script-pre-restart
```

何時需要重新啟動才能套用安全性更新？

在某些情況下，Amazon Linux 需要重新開機才能套用更新：

- Linux 核心套件的更新需要重新開機，才能啟用具有最新安全性更新的新核心。核心即時修補可能允許您在有限的時間內延遲安全更新。如需詳細資訊，請參閱 AL2023 上的核心即時修補。
- 在 EC2 Metal 執行個體上，Amazon Linux 提供微碼更新 （透過 Intel CPUsmicrocode_ctl套件和 AMD CPU amd-ucode-firmware套件）。 CPUs.) 這些微碼更新只會在後續執行個體重新啟動時啟用。對於虛擬化 EC2 執行個體，基礎 AWS Nitro 系統會為您處理微碼更新。
- 有些執行中的系統化服務只有在完全系統重新啟動後才能正常運作。smart-restart 機制會保留重新啟動提示，通知您這類情況。請參閱 在 （安全性） 更新後自動重新啟動服務。

啟動已啟用最新儲存庫版本的執行個體

您可以將 DNF 命令新增至使用者資料指令碼，以控制啟動 Amazon Linux AMI 時要安裝哪些 RPM 套件。在下列範例中，使用者資料指令碼是用來確認以使用者資料指令碼啟動的任何執行個體都已安裝相同的套件更新。

```
#!/bin/bash
dnf upgrade --releasever=2023.0.20230210
# Additional setup and install commands below
dnf install httpd php7.4 mysql80
```

您必須以超級使用者 (根使用者) 的身分執行此指令碼。若要進行這項動作，請執行以下命令。

```
$ sudo sh -c "bash nameofscript.sh"
```

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的使用者資料和 shell 指令碼。

> **Note**
>
> 不要使用使用者資料指令碼，而是啟動最新的 Amazon Linux AMI 或以 Amazon Linux AMI 為基礎的自訂 AMI。最新的 Amazon Linux AMI 已安裝所有必要的更新，並且設定為指向特定的儲存庫版本。

取得套件支援資訊

AL2023 包含許多不同的開放原始碼軟體專案。這些專案全都是獨立於 Amazon Linux 管理，並且有不同的發行和終止支援排程。為了向您提供有關這些不同套件的 Amazon Linux 特定資訊，DNF`supportinfo` 外掛程式會提供有關套件的中繼資料。在下列範例中，**`dnf supportinfo`** 命令會傳回 `glibc` 套件的中繼資料。

```
$ sudo dnf supportinfo --pkg glibc
Last metadata expiration check: 0:07:56 ago on Wed Mar  1 23:21:49 2023.
Name                 : glibc
Version              : 2.34-52.amzn2023.0.2
State                : installed
Support Status       : supported
Support Periods      : from 2023-03-15      : supported
                     : from 2028-03-15      : unsupported
Support Statement    : Amazon Linux 2023 End Of Life
Link                 : https://aws.amazon.com/amazon-linux-ami/faqs/
Other Info           : This is the support statement for AL2023. The
     ...: end of life of Amazon Linux 2023 would be March 2028.
     ...: From this point, the Amazon Linux 2023 packages (listed
  ...: below) will no longer, receive any updates from AWS.
```

套件支援資訊也可在 AL2023 版本備註的支援陳述式區段中取得。

使用 檢查較新的儲存庫版本 dnf check-release-update

在 AL2023 執行個體中，您可以使用 DNF 公用程式來管理儲存庫和套用更新的 RPM 套件。這些套件可在 Amazon Linux 儲存庫中取得。您可以使用 DNF 命令 dnf check-release-update 來檢查 DNF 儲存庫的新版本。

> **Note**
>
> 根據預設，AL2023 容器映像不包含 dnf check-release-update命令。
>
> ```
> $ dnf check-release-update
> No such command: check-release-update. Please use /usr/bin/dnf --help
> It could be a DNF plugin command, try: "dnf install 'dnf-command(check-release-
> update)'"
> ```
>
> dnf install 'dnf-command(check-release-update)' 執行 時， dnf會安裝提供 check-release-update命令的 套件，也就是 dnf-plugin-release-notification套件。在下列範例中， 引-q數會提供給 dnf ，使其具有安靜的輸出。
>
> ```
> $ dnf -y -q install 'dnf-command(check-release-update)'
> Installed:
>   dnf-plugin-release-notification-1.2-1.amzn2023.0.2.noarch
> ```

在 Amazon EC2 執行個體等非容器化環境中，預設會包含 check-release-update命令。

```
$ sudo dnf check-release-update
WARNING:
  A newer release of "Amazon Linux" is available.

  Available Versions:

  Version 2023.0.20230210:
    Run the following command to update to 2023.0.20230210:

      dnf upgrade --releasever=2023.0.20230210

    Release notes:
```

```
    https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes.html
```

這將傳回所有可用 DNF 儲存庫新版本的完整清單。如果沒有傳回任何內容，這代表 DNF 目前設定為使用最新的可用版本。目前安裝的 system-release 套件版本設定 releasever DNF 變數。若要檢查目前儲存庫版本，請執行以下命令。

```
$ rpm -q system-release --qf "%{VERSION}\n"
```

當您執行 DNF 套件交易 (例如安裝、更新或移除命令) 時，會出現警告訊息，通知您是否有任何新的儲存庫版本。例如，如果您在從舊版 AL2023 啟動的執行個體上安裝 httpd 套件，則會傳回下列輸出。

```
$ sudo dnf install httpd -y
Last metadata expiration check: 0:16:52 ago on Wed Mar  1 23:21:49 2023.
Dependencies resolved.
================================================================================
 Package            Arch   Version                Repository   Size
================================================================================
Installing:
 httpd              x86_64 2.4.54-3.amzn2023.0.4  amazonlinux  46 k
Installing dependencies:
 apr                x86_64 1.7.2-2.amzn2023.0.2   amazonlinux 129 k
 apr-util           x86_64 1.6.3-1.amzn2023.0.1   amazonlinux  98 k
 generic-logos-httpd
                    noarch 18.0.0-12.amzn2023.0.3 amazonlinux  19 k
 httpd-core         x86_64 2.4.54-3.amzn2023.0.4  amazonlinux 1.3 M
 httpd-filesystem   noarch 2.4.54-3.amzn2023.0.4  amazonlinux  13 k
 httpd-tools        x86_64 2.4.54-3.amzn2023.0.4  amazonlinux  80 k
 libbrotli          x86_64 1.0.9-4.amzn2023.0.2   amazonlinux 315 k
 mailcap            noarch 2.1.49-3.amzn2023.0.3  amazonlinux  33 k
Installing weak dependencies:
 apr-util-openssl   x86_64 1.6.3-1.amzn2023.0.1   amazonlinux  17 k
 mod_http2          x86_64 1.15.24-1.amzn2023.0.3 amazonlinux 152 k
 mod_lua            x86_64 2.4.54-3.amzn2023.0.4  amazonlinux  60 k

Transaction Summary
================================================================================
Install  12 Packages

Total download size: 2.3 M
Installed size: 6.8 M
Downloading Packages:
(1/12): apr-util-openssl-1.6.3-1.am 212 kB/s |  17 kB     00:00
```

```
(2/12): apr-1.7.2-2.amzn2023.0.2.x8 1.1 MB/s | 129 kB     00:00
(3/12): httpd-core-2.4.54-3.amzn202 8.9 MB/s | 1.3 MB     00:00
(4/12): mod_http2-1.15.24-1.amzn202 1.9 MB/s | 152 kB     00:00
(5/12): apr-util-1.6.3-1.amzn2023.0 1.7 MB/s |  98 kB     00:00
(6/12): mod_lua-2.4.54-3.amzn2023.0 1.4 MB/s |  60 kB     00:00
(7/12): httpd-2.4.54-3.amzn2023.0.4 1.5 MB/s |  46 kB     00:00
(8/12): libbrotli-1.0.9-4.amzn2023. 4.4 MB/s | 315 kB     00:00
(9/12): mailcap-2.1.49-3.amzn2023.0 753 kB/s |  33 kB     00:00
(10/12): httpd-tools-2.4.54-3.amzn2 978 kB/s |  80 kB     00:00
(11/12): httpd-filesystem-2.4.54-3. 210 kB/s |  13 kB     00:00
(12/12): generic-logos-httpd-18.0.0 439 kB/s |  19 kB     00:00
--------------------------------------------------------------------
Total                               6.6 MB/s | 2.3 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                             1/1
  Installing       : apr-1.7.2-2.amzn2023.0.2.x86_64             1/12
  Installing       : apr-util-openssl-1.6.3-1.amzn2023.0.1.      2/12
  Installing       : apr-util-1.6.3-1.amzn2023.0.1.x86_64        3/12
  Installing       : mailcap-2.1.49-3.amzn2023.0.3.noarch        4/12
  Installing       : httpd-tools-2.4.54-3.amzn2023.0.4.x86_      5/12
  Installing       : generic-logos-httpd-18.0.0-12.amzn2023      6/12
  Running scriptlet: httpd-filesystem-2.4.54-3.amzn2023.0.4      7/12
  Installing       : httpd-filesystem-2.4.54-3.amzn2023.0.4      7/12
  Installing       : httpd-core-2.4.54-3.amzn2023.0.4.x86_6      8/12
  Installing       : mod_http2-1.15.24-1.amzn2023.0.3.x86_6      9/12
  Installing       : libbrotli-1.0.9-4.amzn2023.0.2.x86_64      10/12
  Installing       : mod_lua-2.4.54-3.amzn2023.0.4.x86_64       11/12
  Installing       : httpd-2.4.54-3.amzn2023.0.4.x86_64         12/12
  Running scriptlet: httpd-2.4.54-3.amzn2023.0.4.x86_64         12/12
  Verifying        : apr-1.7.2-2.amzn2023.0.2.x86_64             1/12
  Verifying        : apr-util-openssl-1.6.3-1.amzn2023.0.1.      2/12
  Verifying        : httpd-core-2.4.54-3.amzn2023.0.4.x86_6      3/12
  Verifying        : mod_http2-1.15.24-1.amzn2023.0.3.x86_6      4/12
  Verifying        : apr-util-1.6.3-1.amzn2023.0.1.x86_64        5/12
  Verifying        : mod_lua-2.4.54-3.amzn2023.0.4.x86_64        6/12
  Verifying        : libbrotli-1.0.9-4.amzn2023.0.2.x86_64       7/12
  Verifying        : httpd-2.4.54-3.amzn2023.0.4.x86_64          8/12
  Verifying        : httpd-tools-2.4.54-3.amzn2023.0.4.x86_      9/12
  Verifying        : mailcap-2.1.49-3.amzn2023.0.3.noarch       10/12
  Verifying        : httpd-filesystem-2.4.54-3.amzn2023.0.4     11/12
```

```
  Verifying        : generic-logos-httpd-18.0.0-12.amzn2023   12/12

Installed:
  apr-1.7.2-2.amzn2023.0.2.x86_64
  apr-util-1.6.3-1.amzn2023.0.1.x86_64
  apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64
  generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
  httpd-2.4.54-3.amzn2023.0.4.x86_64
  httpd-core-2.4.54-3.amzn2023.0.4.x86_64
  httpd-filesystem-2.4.54-3.amzn2023.0.4.noarch
  httpd-tools-2.4.54-3.amzn2023.0.4.x86_64
  libbrotli-1.0.9-4.amzn2023.0.2.x86_64
  mailcap-2.1.49-3.amzn2023.0.3.noarch
  mod_http2-1.15.24-1.amzn2023.0.3.x86_64
  mod_lua-2.4.54-3.amzn2023.0.4.x86_64

Complete!
```

新增、啟用或停用新儲存庫

> **Warning**
>
> 僅新增旨在與 AL2023 搭配使用的儲存庫。
> 雖然專為其他 分佈設計的儲存庫可能目前有效，但無法保證他們會繼續使用 AL2023 中的任何套件更新，或並非設計用於 AL2023 的儲存庫。

若要從與預設 Amazon Linux 儲存庫不同的儲存庫安裝套件，您需要設定DNF套件管理系統，以了解儲存庫的位置

若要dnf告知套件儲存庫，請將儲存庫資訊新增至 `/etc/yum.repos.d/`目錄中該儲存庫的組態檔案。許多第三方儲存庫提供組態檔案內容或包含組態檔案的可安裝套件。

> **Note**
>
> 雖然儲存庫可以直接在 `/etc/dnf/dnf.conf` 檔案中設定，但不建議這麼做。建議在 中以自己的檔案設定每個儲存庫`/etc/yum.repos.d/`。

若要了解目前啟用的儲存庫，可以執行以下命令：

```
$ dnf repolist all --verbose
Loaded plugins: builddep, changelog, config-manager, copr, debug, debuginfo-install,
 download, generate_completion_cache, groups-manager, needs-restarting, playground,
 release-notification, repoclosure, repodiff, repograph, repomanage, reposync,
 supportinfo
DNF version: 4.12.0
cachedir: /var/cache/dnf
Last metadata expiration check: 0:00:02 ago on Wed Mar  1 23:40:15 2023.
Repo-id            : amazonlinux
Repo-name          : Amazon Linux 2023 repository
Repo-status        : enabled
Repo-revision      : 1677203368
Repo-updated       : Fri Feb 24 01:49:28 2023
Repo-pkgs          : 12632
Repo-available-pkgs: 12632
Repo-size          : 12 G
Repo-mirrors       : https://al2023-repos-us-west-2-de612dc2.s3.dualstack.us-
west-2.amazonaws.com/core/mirrors/2023.0.20230222/x86_64/mirror.list
Repo-baseurl       : https://al2023-repos-us-west-2-de612dc2.s3.dualstack.us-
west-2.amazonaws.com/core/guids/
cf9296325a6c46ff40c775a8e2d632c4c3fd9d9164014ce3304715d61b90ca8e/x86_64/
                   : (0 more)
Repo-expire        : 172800 second(s) (last: Wed Mar  1 23:40:15
                   : 2023)
Repo-filename      : /etc/yum.repos.d/amazonlinux.repo

Repo-id            : amazonlinux-debuginfo
Repo-name          : Amazon Linux 2023 repository - Debug
Repo-status        : disabled
Repo-mirrors       : https://al2023-repos-us-west-2-de612dc2.s3.dualstack.us-
west-2.amazonaws.com/core/mirrors/2023.0.20230222/debuginfo/x86_64/mirror.list
Repo-expire        : 21600 second(s) (last: unknown)
Repo-filename      : /etc/yum.repos.d/amazonlinux.repo

Repo-id            : amazonlinux-source
Repo-name          : Amazon Linux 2023 repository - Source packages
Repo-status        : disabled
Repo-mirrors       : https://al2023-repos-us-west-2-de612dc2.s3.dualstack.us-
west-2.amazonaws.com/core/mirrors/2023.0.20230222/SRPMS/mirror.list
Repo-expire        : 21600 second(s) (last: unknown)
Repo-filename      : /etc/yum.repos.d/amazonlinux.repo

Repo-id            : kernel-livepatch
```

```
Repo-name           : Amazon Linux 2023 Kernel Livepatch repository
Repo-status         : disabled
Repo-mirrors        : https://al2023-repos-us-west-2-de612dc2.s3.dualstack.us-
west-2.amazonaws.com/kernel-livepatch/mirrors/al2023/x86_64/mirror.list
Repo-expire         : 172800 second(s) (last: unknown)
Repo-filename       : /etc/yum.repos.d/kernel-livepatch.repo

Repo-id             : kernel-livepatch-source
Repo-name           : Amazon Linux 2023 Kernel Livepatch repository -
                    : Source packages
Repo-status         : disabled
Repo-mirrors        : https://al2023-repos-us-west-2-de612dc2.s3.dualstack.us-
west-2.amazonaws.com/kernel-livepatch/mirrors/al2023/SRPMS/mirror.list
Repo-expire         : 21600 second(s) (last: unknown)
Repo-filename       : /etc/yum.repos.d/kernel-livepatch.repo
Total packages: 12632
```

> **Note**
>
> 如果您未新增 `--verbose` 選項旗標，輸出只會包含 `Repo-id`、`Repo-name` 和 `Repo-status` 資訊。

若要將 **yum** 儲存庫新增至 **/etc/yum.repos.d** 目錄：

1. 尋找 `.repo` 檔案的位置。在此範例中，`.repo` 檔案位於 *https://www.example.com/repository.repo*。
2. 使用 `dnf config-manager` 命令新增儲存庫。

```
$ sudo dnf config-manager --add-repo https://www.example.com/repository.repo
Loaded plugins: priorities, update-motd, upgrade-helper
adding repo from: https://www.example.com/repository.repo
grabbing file https://www.example.com/repository.repo to /etc/
yum.repos.d/repository.repo
repository.repo                                          | 4.0 kB     00:00
repo saved to /etc/yum.repos.d/repository.repo
```

在您安裝儲存庫後，您必須啟用它，如下一個程序中所說明。

若要在 /etc/yum.repos.d 中啟用 yum 儲存庫，請使用有 --enable 旗標和###名稱的 dnf config-manager 命令。

```
$ sudo dnf config-manager --enable repository
```

> **Note**
>
> 若要停用儲存庫，請使用相同的命令語法，但在命令中以 --disable 取代 --enable。

使用 cloud-init 新增儲存庫

除了使用上一個方法新增儲存庫外，您還可以使用 cloud-init 架構來新增儲存庫。

若要新增套件儲存庫，建議您使用下列範本。請考慮在本機儲存此檔案。

```
#cloud-config
yum_repos:
  repository.repo:
    baseurl: https://www.example.com/
    enabled: true
    gpgcheck: true
    gpgkey: file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EXAMPLE
    name: Example Repository
```

> **Note**
>
> 使用 cloud-init 的其中一個優點是您可以在設定檔案中加入 packages: 部分。在本節中，您可以加入要安裝的套件名稱。您可以從預設儲存庫或在 cloud-config 檔案中新增的新儲存庫安裝套件。
> 如需 YAML 檔案結構的詳細資訊，請參閱 *cloud-init*文件中的新增 YUM 儲存庫。

設定 YAML 格式檔案後，您可以在 cloud-init 的 AWS CLI架構中執行。務必加入 --userdata 選項和 .yml 檔案的名稱，以呼叫所需的操作。

```
$ aws ec2 run-instances \
  --image-id \
    resolve:ssm:/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-default-x86_64 \
```

```
  --instance-type m5.xlarge \
  --region us-east-1 \
  --key-name aws-key-us-east-1 \
  --security-group-ids sg-004a7650 \
  --user-data file://cloud-config.yml
```

AL2023 上的核心即時修補

您可以使用適用於 AL2023 的 Kernel Live Patching，將特定安全性漏洞和關鍵錯誤修補程式套用至執行中的 Linux 核心，而不會重新啟動或中斷執行中的應用程式。此外，核心即時修補有助於改善應用程式的可用性，同時套用這些修正，直到系統可以重新啟動為止。

AWS 為 AL2023 發行兩種類型的核心即時修補程式：

- 安全性更新 – 包含 Linux 常見漏洞和入侵程式 (CVE) 的更新。這些更新通常會使用 Amazon Linux 安全建議分級評定為 important (重要) 或 critical (嚴重)。它們通常對應到通用漏洞評分系統 (CVSS) 分數的 7 及以上。在某些情況下， AWS 可能會在指派 CVE 之前提供更新。在這些情況下，修補程式可能會顯示為錯誤修正。
- 錯誤修正 – 包含與 CVE 無關的重大錯誤和穩定性問題修正。

AWS 提供 AL2023 核心版本的核心即時修補程式，最長可達發行後 3 個月。經過此期間後，您必須更新至更新的核心版本，才能繼續接收核心即時修補程式。

AL2023 核心即時修補程式會在現有 AL2023 軟體庫中以已簽署的 RPM 套件形式提供。這些修補程式可以透過現有 DNF 套件管理員工作流程安裝至個別執行個體。或者，它們可以使用 AWS Systems Manager 安裝在一組受管執行個體上。

AL2023 的 Kernel Live Patching 以無需額外費用的方式提供。

主題

限制

套用核心即時修補程式時，您無法執行休眠、使用進階偵錯工具 (例如 SystemTap、kprobes 和 eBPF 為基礎的工具)，或存取 Kernel Live Patching 基礎架構所使用的 ftrace 輸出檔案。

> **Note**
>
> 由於技術限制，即時修補無法解決某些問題。因此，這些修正不會在核心即時修補程式套件中運送，只會在原生核心套件更新中運送。您可以安裝原生核心套件，並照常更新和重新啟動系統以啟用修補程式。

支援的組態和先決條件

Amazon EC2 執行個體和執行 AL2023 的內部部署虛擬機器支援 Kernel Live Patching。

若要在 AL2023 上使用 Kernel Live Patching，您必須使用下列：

- 64 位元 `x86_64` 或 `ARM64` 架構
- 核心版本`6.1`或 `6.12`

政策要求

若要從 AL2023 儲存庫下載套件，Amazon EC2 需要存取服務擁有的 Amazon S3 儲存貯體。如果您在環境中使用 Amazon S3 的 Amazon Virtual Private Cloud (VPC) 端點，請確定您的 VPC 端點政策允許存取這些公有儲存貯體。 Amazon S3 下表說明 Amazon EC2 存取 Kernel Live Patching 可能需要的 Amazon S3 儲存貯體。 Amazon EC2

S3 儲存貯體 ARN	描述
arn：aws：s3：：al2023-repos-*region*-de612dc2/*	包含 AL2023 儲存庫的 Amazon S3 儲存貯體 AL2023

使用 Kernel Live Patching

您可以透過執行個體本身的命令列，在個別執行個體啟用和使用 Kernel Live Patching。或者，您也可以使用 AWS 系統管理員，在受管執行個體群組啟用和使用 Kernel Live Patching。

下列各節說明如何透過命令列在個別執行個體上啟用並使用 Kernel Live Patching。

如需有關在受管執行個體群組上啟用並使用 Kernel Live Patching 的詳細資訊，請參閱 AWS Systems Manager 使用者指南中的在 AL2023 執行個體上使用 Kernel Live Patching。

主題

啟用 Kernel Live Patching

Kernel Live Patching 在 AL2023 上預設為停用狀態。若要使用即時修補，您必須安裝 Kernel Live Patching 的 DNF 外掛程式，並啟用即時修補功能。

啟用 Kernel Live Patching

1. 核心即時修補程式適用於搭載核心版本 6.1 的 AL2023。若要檢查您的核心版本，請執行下列命令。

   ```
   $ sudo dnf list kernel
   ```

2. 安裝 Kernel Live Patching 的 DNF 外掛程式。

   ```
   $ sudo dnf install -y kpatch-dnf
   ```

3. 啟用 Kernel Live Patching 的 DNF 外掛程式。

   ```
   $ sudo dnf kernel-livepatch -y auto
   ```

   此命令也會從設定的軟體庫安裝最新版本的核心即時修補程式 RPM。

4. 若要確認核心即時修補的 DNF 外掛程式是否成功安裝，請執行下列命令。

   當您啟用 Kernel Live Patching 時，系統會自動套用空白的核心即時修補程式 RPM。如果成功啟用核心即時修補，此命令會傳回包含初始空白核心即時修補 RPM （以及另一個 RPM 設定包含即時修補的 DNF 儲存庫） 的清單。

   ```
   $ sudo rpm -qa | grep kernel-livepatch
   kernel-livepatch-repo-s3-2023.7.20250428-0.amzn2023.noarch
   kernel-livepatch-6.1.134-150.224-1.0-0.amzn2023.x86_64
   ```

5. 安裝 kpatch 套件。

```
$ sudo dnf install -y kpatch-runtime
```

6. 如果之前已安裝 kpatch 服務，請更新此服務。

```
$ sudo dnf upgrade kpatch-runtime
```

7. 啟動 kpatch 服務。此服務會在初始化或開機時載入所有核心即時修補程式。

```
$ sudo systemctl enable kpatch.service && sudo systemctl start kpatch.service
```

檢視可用的核心即時修補程式

Amazon Linux 安全性警示會發佈至資訊 Amazon Linux 安全中心。如需有關 AL2023 安全性警示 (包括核心即時修補程式警示) 的詳細資訊，請參閱 Amazon Linux 安全中心。核心即時修補程式的字首為 ALASLIVEPATCH。Amazon Linux 安全中心可能不會列出解決錯誤的核心即時修補程式。

您也可以使用命令列來探索建議和 CVE 的可用核心即時修補程式。

列出所有可用的核心即時修補程式以供建議使用

使用下列 命令。

```
$ sudo dnf updateinfo list
Last metadata expiration check: 1:06:23 ago on Mon 13 Feb 2023 09:28:19 PM UTC.
ALAS2LIVEPATCH-2021-123   important/Sec. kernel-
livepatch-6.1.12-17.42-1.0-4.amzn2023.x86_64
ALAS2LIVEPATCH-2022-124   important/Sec. kernel-
livepatch-6.1.12-17.42-1.0-3.amzn2023.x86_64
```

列出 CVE 的所有可用核心即時修補程式

使用下列 命令。

```
$ sudo dnf updateinfo list cves
Last metadata expiration check: 1:07:26 ago on Mon 13 Feb 2023 09:28:19 PM UTC.
CVE-2022-0123    important/Sec. kernel-livepatch-6.1.12-17.42-1.0-4.amzn2023.x86_64
CVE-2022-3210    important/Sec. kernel-livepatch-6.1.12-17.42-1.0-3.amzn2023.x86_64
```

套用核心即時修補程式

您可以使用 DNF 套件管理員來套用核心即時修補程式，方法與套用定期更新相同。適用於 Kernel Live Patching 的 DNF 外掛程式會管理可供套用的核心即時修補程式。

> **Tip**
>
> 我們建議您定期使用 Kernel Live Patching 更新核心，以確保它收到特定的重要和關鍵安全修正，直到系統可以重新啟動為止。也請檢查是否已為無法部署為即時修補程式的原生核心套件提供其他修正，並針對這些情況更新和重新啟動核心更新。

您可以選擇套用特定的核心即時修補程式，或套用任何可用的核心即時修補程式，以及定期的安全性更新。

套用特定核心即時修補程式

1. 使用 檢視可用的核心即時修補程式 中描述的其中一個命令取得核心即時修補程式版本。
2. 為您的 AL2023 核心套用核心即時修補程式。

   ```
   $ sudo dnf install kernel-livepatch-kernel_version-package_version.amzn2023.x86_64
   ```

   例如，下列命令會針對 AL2023 核心版本 `6.1.12-17.42` 套用核心即時修補程式。

   ```
   $ sudo dnf install kernel-livepatch-6.1.12-17.42-1.0-4.amzn2023.x86_64
   ```

套用任何可用的核心即時修補程式，以及您的定期安全性更新

使用下列命令。

```
$ sudo dnf upgrade --security
```

省略包含錯誤修正的 `--security` 選項。

> **Important**
>
> - 套用核心即時修補程式後，核心版本不會更新。只有在執行個體重新啟動後，版本才會更新為新版本。

- AL2023 核心會接收核心即時修補程式，為期三個月。經過此期間後，該核心版本不會發行任何新的核心即時修補程式。
- 若要在 3 個月後繼續接收核心即時修補程式，您必須重新啟動執行個體，才能轉換到新的核心版本。執行個體會在您更新核心後的 3 個月內繼續收到核心即時修補程式。
- 若要檢查核心版本的支援視窗，請執行下列命令：

```
$ sudo dnf kernel-livepatch support
The current version of the Linux kernel you are running will no longer receive
 live patches after 2025-07-22.
```

檢視套用的核心即時修補程式

檢視套用的核心即時修補程式

使用下列命令。

```
$ sudo kpatch list
Loaded patch modules:
livepatch_CVE_2022_36946 [enabled]

Installed patch modules:
livepatch_CVE_2022_36946 (6.1.57-29.131.amzn2023.x86_64)
livepatch_CVE_2022_36946 (6.1.57-30.131.amzn2023.x86_64)
```

此命令會傳回已載入及已安裝的安全性更新核心即時修補程式清單。下列為範例輸出。

> **Note**
>
> 單一核心即時修補程式可包含並安裝多個即時修補程式。

停用核心即時修補

如果您不再需要使用 Kernel Live Patching，您可以隨時停用它。

- 停用使用 livepatches：
  1. 停用外掛程式：

```
$ sudo dnf kernel-livepatch manual
```

2. 停用 kpatch 服務：

```
$ sudo systemctl disable --now kpatch.service
```

- 完全移除 livepatch 工具：

1. 移除外掛程式：

```
$ sudo dnf remove kpatch-dnf
```

2. 移除 kpatch-runtime：

```
$ sudo dnf remove kpatch-runtime
```

3. 移除任何已安裝的 livepatches：

```
$ sudo dnf remove kernel-livepatch\*
```

在 AL2023 上更新 Linux 核心

主題

- AL2023 上的 Linux 核心版本
- 將 AL2023 更新至較新的核心版本
- AL2023 核心 - 常見問答集

AL2023 上的 Linux 核心版本

AL2023 定期包含以 Linux 核心長期支援 (LTS) 版本為基礎的新核心版本。

AL2023 最初於 2023 年 3 月發行，採用核心 6.1。

2025 年 4 月，AL2023 新增對 Linux 核心 6.12 的支援。此核心新增了新功能，包括 EEVDF 排程、FUSE 傳遞 I/O 支援、新的 Futex API，以及 eBPF 的改善。核心 6.12 也允許使用者空間程式在執行階段使用使用者空間陰影堆疊和記憶體密封來保護自己。

2026 年 3 月，AL2023 新增對 Linux 核心 6.18 的支援。更新的核心 6.18 可進一步改善處理器支援、虛擬化、安全性和效能。值得注意的功能包括改善跨架構的 IOMMU 功能，以及用於管理 CPU 漏洞緩解的攻擊向量控制。效能增強是透過更快速的 FSCRYPT 操作、記憶體管理改進，以及將 Sheaves 作為新的選擇加入、每個 CPU 陣列型快取層進行加密最佳化。

將 AL2023 更新至較新的核心版本

從 2026 年 6 月開始，AL2023 將每年更新預設核心。`al2023-ami-kernel-default` 一組 AMIs將更新為最新的 LTS 核心，因此新啟動的執行個體會自動產生新的核心版本，這是隨時掌握最新安全性修正和效能改善的最簡單方法。

如果您想要選擇特定的核心版本，您可以選取預先安裝所需核心的 AMI 或升級現有的 AL2023 EC2 執行個體，以核心 6.12 或 6.18 執行 AL2023。

使用特定核心版本執行 AL2023 AMI

您可以選擇使用透過 AWS 主控台或查詢 SSM 以取得特定參數預先安裝的特定核心來執行 AL2023 AMI。要查詢的 SSM 金鑰以 開頭，`/aws/service/ami-amazon-linux-latest/`後面接著其中一個

對於核心 6.12

- `al2023-ami-kernel-6.12-arm64` 適用於 arm64 架構
- `al2023-ami-minimal-kernel-6.12-arm64` 適用於 arm64 架構 (最小 AMI)
- `al2023-ami-kernel-6.12-x86_64` 適用於 x86_64 架構
- `al2023-ami-minimal-kernel-6.12-x86_64` 適用於 x86_64 架構 (最小 AMI)

對於核心 6.18

- `al2023-ami-kernel-6.18-arm64` 適用於 arm64 架構
- `al2023-ami-minimal-kernel-6.18-arm64` 適用於 arm64 架構 (最小 AMI)
- `al2023-ami-kernel-6.18-x86_64` 適用於 x86_64 架構
- `al2023-ami-minimal-kernel-6.18-x86_64` 適用於 x86_64 架構 (最小 AMI)

如需選取 AL2023 AMIs的詳細資訊使用 SSM 參數和 啟動 AL2023 AWS CLI，請參閱 。

將 AL2023 執行個體更新為較新的核心

您可以使用下列步驟，將執行中的 AL2023 執行個體就地升級至核心 6.12 或 6.18：

1. 偵測目前的核心並設定目標版本：

```
# Automatically detect current kernel version BEFORE upgrade
$ CURRENT_KERNEL=$(uname -r)
$ SOURCE_VERSION=""

$ if [[ $CURRENT_KERNEL == *"6.12"* ]]; then
    SOURCE_VERSION="6.12"
else
    SOURCE_VERSION=""
fi

# Save the source version to a persistent location for use after reboot
$ echo "${SOURCE_VERSION}" | sudo tee /var/lib/source_kernel_version > /dev/null
```

```
# Set your target version (change this to your desired kernel: 6.12 or 6.18)
$ TARGET_VERSION="6.12"
```

```
$ echo "Current kernel: ${SOURCE_VERSION:-6.1}"
$ echo "Upgrading to kernel ${TARGET_VERSION}"
```

2. 安裝目標核心套件：

```
$ sudo dnf install -y kernel${TARGET_VERSION}
```

3. 取得目標核心套件的最新版本：

```
$ version=$(rpm -q --qf '%{version}-%{release}.%{arch}\n' kernel${TARGET_VERSION} |
 sort -V | tail -1)
```

4. 將新核心設為預設核心：

```
$ sudo grubby --set-default "/boot/vmlinuz-$version"
```

5. 重新啟動您的系統：

```
$ sudo reboot
```

6. 解除安裝先前的核心：

```
# Read the source kernel version from the saved file
$ SOURCE_VERSION=$(sudo cat /var/lib/source_kernel_version)

# Uninstall the source kernel
$ sudo dnf remove -y kernel${SOURCE_VERSION}
```

7. 將額外的核心套件取代為其目標核心對等項目：

```
# Set your target version (change this to your desired kernel: 6.12 or 6.18)
$ TARGET_VERSION="6.12"
```

```
$ declare -A pkgs
$ pkgs=(
[bpftool${SOURCE_VERSION}]=bpftool${TARGET_VERSION}
[kernel${SOURCE_VERSION}-debuginfo]=kernel${TARGET_VERSION}-debuginfo
[kernel${SOURCE_VERSION}-debuginfo-common]=kernel${TARGET_VERSION}-debuginfo-common
[kernel${SOURCE_VERSION}-headers]=kernel${TARGET_VERSION}-headers
[kernel${SOURCE_VERSION}-libbpf]=kernel${TARGET_VERSION}-libbpf
[kernel${SOURCE_VERSION}-libbpf-devel]=kernel${TARGET_VERSION}-libbpf-devel
[kernel${SOURCE_VERSION}-libbpf-static]=kernel${TARGET_VERSION}-libbpf-static
[kernel${SOURCE_VERSION}-modules-extra-common]=kernel${TARGET_VERSION}-modules-
extra-common
[kernel${SOURCE_VERSION}-tools]=kernel${TARGET_VERSION}-tools
[kernel${SOURCE_VERSION}-tools-devel]=kernel${TARGET_VERSION}-tools-devel
[perf${SOURCE_VERSION}]=perf${TARGET_VERSION}
[python3-perf${SOURCE_VERSION}]=python3-perf${TARGET_VERSION}
)
$ for pkg in "${!pkgs[@]}"; do
  rpm -q $pkg && sudo dnf -y swap $pkg "${pkgs["$pkg"]}" ;
done
```

8. （選用）解除安裝先前核心版本的 kernel-devel：

```
$ rpm -q kernel${SOURCE_VERSION}-devel && sudo dnf remove -y kernel
${SOURCE_VERSION}-devel
```

降級至較舊的核心版本

如果您在任何時候需要降級至較早的核心版本，請使用下列步驟：

1. 偵測目前的核心並設定目標版本：

```
# Automatically detect current kernel version BEFORE downgrade
$ CURRENT_KERNEL=$(uname -r)
$ SOURCE_VERSION=""

$ if [[ $CURRENT_KERNEL == *"6.12"* ]]; then
    SOURCE_VERSION="6.12"
elif [[ $CURRENT_KERNEL == *"6.18"* ]]; then
    SOURCE_VERSION="6.18"
fi

# Save the source version to a persistent location for use after reboot
$ echo "${SOURCE_VERSION}" | sudo tee /var/lib/source_kernel_version > /dev/null
```

```
# Set your target version (change this to your desired kernel)
# Use "" for kernel 6.1, "6.12" for kernel 6.12
$ TARGET_VERSION=""
```

```
$ echo "Downgrading from kernel ${SOURCE_VERSION:-6.1} to kernel
 ${TARGET_VERSION:-6.1}"
```

2. 將額外的核心套件取代為其目標核心對等項目：

```
$ declare -A pkgs
$ pkgs=(
[bpftool${TARGET_VERSION}]=bpftool${SOURCE_VERSION}
[kernel${TARGET_VERSION}-debuginfo]=kernel${SOURCE_VERSION}-debuginfo
[kernel${TARGET_VERSION}-debuginfo-common]=kernel${SOURCE_VERSION}-debuginfo-common
[kernel${TARGET_VERSION}-headers]=kernel${SOURCE_VERSION}-headers
[kernel${TARGET_VERSION}-libbpf]=kernel${SOURCE_VERSION}-libbpf
[kernel${TARGET_VERSION}-libbpf-devel]=kernel${SOURCE_VERSION}-libbpf-devel
[kernel${TARGET_VERSION}-libbpf-static]=kernel${SOURCE_VERSION}-libbpf-static
[kernel${TARGET_VERSION}-modules-extra-common]=kernel${SOURCE_VERSION}-modules-
extra-common
[kernel${TARGET_VERSION}-tools]=kernel${SOURCE_VERSION}-tools
[kernel${TARGET_VERSION}-tools-devel]=kernel${SOURCE_VERSION}-tools-devel
[perf${TARGET_VERSION}]=perf${SOURCE_VERSION}
```

```
[python3-perf${TARGET_VERSION}]=python3-perf${SOURCE_VERSION}
)
$ for pkg in "${!pkgs[@]}"; do
  rpm -q "${pkgs["$pkg"]}" && sudo dnf -y swap "${pkgs["$pkg"]}" $pkg ;
done
```

3. 安裝目標核心套件：

```
$ sudo dnf install -y kernel${TARGET_VERSION}
```

4. 取得目標核心套件的最新版本：

```
$ version=$(rpm -q --qf '%{version}-%{release}.%{arch}\n' kernel${TARGET_VERSION} |
 sort -V | tail -1)
```

5. 將目標核心設為預設核心：

```
$ sudo grubby --set-default "/boot/vmlinuz-$version"
```

6. 重新啟動您的系統：

```
$ sudo reboot
```

7. 解除安裝來源核心：

```
# Read the source kernel version from the saved file
$ SOURCE_VERSION=$(sudo cat /var/lib/source_kernel_version)

# Uninstall the source kernel
$ sudo dnf remove -y kernel${SOURCE_VERSION}
```

AL2023 核心 - 常見問答集

1. 核心更新後是否需要重新啟動？

每個對執行中核心的變更都需要重新啟動。

2. 如何讓多個執行個體的核心up-to-date？

Amazon Linux 不提供管理執行個體機群的設施。我們建議您使用 AWS Systems Manager 等工具來修補大型機群。

3. 如何檢查目前執行的核心版本？

在 AL2023 執行個體上執行此命令：

```
$ uname -r
```

4. AL2023 建議我使用哪個核心？

建議升級至最新的 AL2023 核心 6.18，而所有其他 AL2023 核心仍受支援。建議客戶在升級之前測試工作負載。

5. 我現有的應用程式是否適用於任何 AL2023 核心？

AL2023 支援較新的核心 (6.12 或 6.18)，方式與核心 6.1 相同。應用程式將正常運作，並且正在幕後進行改善。無論如何，客戶都應該先測試其特定工作負載，再切換到較新的核心。

6. 如何為核心 6.12 或 6.18 安裝核心標頭、開發套件和額外模組？

請執行：

```
$ version=$(uname -r | grep -oP '^\d+\.\d+')
$ sudo dnf install -y kernel${version}-modules-extra-$(uname -r) kernel${version}-
headers-$(uname -r) kernel${version}-devel-$(uname -r)
```

7. 支援核心 6.12 和 6.18 多久？

支援 Kernel 6.12 和 6.18，直到 Amazon Linux 2023 的計劃生命週期結束，即 2029-06-30。

AL2023 上的程式設計執行時間入門

AL2023 提供某些語言執行時間的不同版本。我們使用同時支援多個版本的上游專案。了解有關如何使用 dnf 命令來搜尋和安裝這些套件，以安裝和管理這些名稱版本化套件的資訊。

下列主題概述 AL2023 中每個語言生態系統的存在方式。

主題

AL2023 中的 C、C++ 和 Fortran

AL2023 包含 GNU 編譯器集合 (GCC) 和 LLVM（低階虛擬機器） Clang的前端。

GCC 的主要版本將在整個 AL2023 的壽命週期保持不變。次要版本導入錯誤修正，並可能包含在 AL2023 版本中。其他錯誤、效能和安全修正可能會反向移植至 AL2023 隨附的 GCC 主要版本。

AL2023 包含 第 11 版GCC作為具有 C (gcc)、C++ (g++) 和 Fortran (gfortran) 前端的預設編譯器。此外，AL2023 提供第 14 GCC版做為選用的替代編譯器，可與預設版本一起安裝。

AL2023 不會啟用 Ada(gnat)、 Go(gcc-go)、 Objective-C 或 Objective-C++ 前端。

預設的編譯器會標記 AL2023 RPM 是使用包含最佳化和強化旗標建置。若要使用 GCC 建置您自己的程式碼，我們建議您包含最佳化和強化旗標。

> **Note**
>
> 調用 `gcc --version` 時，會顯示 `gcc (GCC) 11.3.1 20221121 (Red Hat 11.3.1-4)` 等的版本字串。`Red Hat` 指的是 Amazon Linux GCC 套件所基於的 GCC 供應商分支。根據 顯示的錯誤報告 URL`gcc --help`，所有錯誤報告和支援請求都應導向 Amazon Linux。
> 如需此廠商分支中某些長期變更的詳細資訊，例如`__GNUC_RH_RELEASE__`巨集，請參閱 Fedora 套件來源。

如需核心工具鏈的詳細資訊，請參閱 核心工具鏈套件 glibc、gcc、binutils。

如需 AL2023 及其與其他 Linux 發行版本之關係的詳細資訊，請參閱 與 Fedora 的關係。

如需 AL2023 中相較於 AL2 的編譯器三元組變更的詳細資訊，請參閱 編譯器三元組。

主題

- GCC 14
- 語言標準版本比較

GCC 14

AL2023 提供 GCC 14 作為選用編譯器，可與預設 11 GCC 一起安裝。14 GCC 包含最新的語言功能和最佳化，因此適合需要較新的 C、C++ 或 Fortran 標準支援的專案。

若要安裝 GCC 14，請使用下列命令：

```
sudo dnf install gcc14 gcc14-c++ gcc14-gfortran
```

14 GCC 個編譯器會安裝版本特定的命令名稱，以避免與預設的 GCC 11 衝突：

- `gcc14-gcc` - C 編譯器
- `gcc14-g++` - C++ 編譯器
- `gcc14-gfortran` - Fortran 編譯器

使用範例：

```
gcc14-gcc -o myprogram myprogram.c
```

```
gcc14-g++ -o mycppprogram mycppprogram.cpp
gcc14-gfortran -o myfortranprogram myfortranprogram.f90
```

您可以執行下列動作來驗證已安裝的版本：

```
gcc14-gcc --version
```

這會顯示類似下列內容的版本資訊：`gcc14-gcc (GCC) 14.2.1 20250110 (Red Hat 14.2.1-7)`

> **Note**
>
> 11 GCC 和 GCC 14 可以同時安裝在相同的系統上。預設 `gcc`、 `g++`和 `gfortran`命令將繼續使用 GCC 11，而 GCC 14 是透過版本特定的命令存取。

語言標準版本比較

下表比較不同 Amazon Linux 版本和GCC編譯器版本的預設語言標準版本：

Amazon Linux 版本	C 標準 （預設）	C++ 標準 （預設）	Fortran Standard
AL2 搭配 GCC 7 （預設）	C11 (201112L)	C++14 (201402L)	Fortran 2008
AL2 搭配 GCC 10 （選用）	C17/C18 (201710L)	C++14 (201402L)	Fortran 2008
AL2023 搭配 GCC 11 （預設）	C17/C18 (201710L)	C++17 (201703L)	Fortran 2008
AL2023 搭配 GCC 14 （選用）	C17/C18 (201710L)	C++17 (201703L)	Fortran 2008

依GCC版本的關鍵改進：

- GCC 10 與 GCC 7：將預設 C 標準從 C11 升級至 C17/C18，新增對 C++20 功能的支援，並改善最佳化功能。

- GCC 11 與 GCC 10：將預設 C++ 標準從 C++14 升級至 C++17、增強 C++20 支援，並新增實驗性 C++23 功能。
- GCC 14 與 GCC 11：新增完整的 C23 標準支援、增強的 C++23 功能、改善的最佳化，以及更好的標準合規。

支援的語言標準：

- C 標準：所有版本都支援 C90, C99, C11 和 C17/C18。10+ GCC 支援 C2x（草稿 C23)，而 GCC 14 提供完整的 C23 支援。
- C++ 標準：所有版本都支援 C++98、C++03、C++11、C++14、C++17 和 C++20。11+ GCC 提供實驗性 C++23 支援，14 GCC 提供增強的 C++23 功能。
- Fortran 標準：所有版本主要支援 Fortran 2008，視GCC版本而定，Fortran 2018 功能具有不同層級。

> **Note**
>
> 雖然預設標準在 GCC 11 和 14 之間保持一致，但 GCC 14 在使用 `-std=`旗標明確請求時，可提供大幅改善的語言功能支援、更佳的最佳化、增強的診斷，以及更完整的新標準實作。

AL2023 中的 Go

您可能想要在 Amazon Linux Go上建置自己的程式碼，並可能想要使用 AL2023 隨附的工具鏈。與 AL2, AL2023會在作業系統的整個生命週期中更新Go工具鏈。這可能是為了回應隨附工具鏈中的任何 CVE，或是包含在季度版本中。

Go 是相對快速移動的語言。在某些情況下，寫入 的現有應用程式Go可能需要適應Go工具鏈的新版本。如需 的詳細資訊Go，請參閱 Go 1 和Go程式的未來。

雖然 AL2023 會在其生命週期內納入新版本的Go工具鏈，但這不會與上游Go版本鎖定。因此，如果您想要使用Go語言和標準程式庫的尖端功能建置Go程式碼，則可能不適合使用 AL2023 中提供Go的工具鏈。

在 AL2023 的生命週期內，先前的套件版本不會從儲存庫中移除。如果需要先前的Go工具鏈，您可以選擇放棄較新Go工具鏈的錯誤和安全性修正，並使用任何 RPM 可用的相同機制從儲存庫安裝先前的版本。

如果您想要在 AL2023 上建置自己的Go程式碼，您可以使用 AL2023 中包含Go的工具鏈，並知道此工具鏈可能會在 AL2023 的生命週期內向前移動。

AL2023 Lambda 函數寫入 Go

做為原生程式碼的Go編譯，Lambda 會將 Go視為自訂執行時間。您可以使用 `provided.al2023` 執行期在 AL2023 上將Go函數部署至 Lambda。

如需詳細資訊，請參閱《 AWS Lambda 開發人員指南》中的使用 建置 Lambda 函數Go。

AL2023 中的 Java

AL2023 提供數個版本的 Amazon Corretto，以支援Java以 為基礎的工作負載。AL2023 中包含的所有 Java 型套件都是使用 建置Amazon Corretto 17。

Corretto 是 Open Java 開發套件 (OpenJDK) 的建置，具有 的長期支援Amazon。Corretto 已使用 Java 技術相容性套件 (TCK) 進行認證，以確保其符合 Java SE 標準，並可在 Linux、 Windows和 上使用macOS。

Amazon Corretto 套件可用於每個版本的 Corretto 1.8.0、Corretto 11 和 Corretto 17。

AL2023 各個 Corretto 版本享有與 Corretto 版本相同的支援期間，或是到 AL2023 的壽命結束 (以較早者為準)。如需詳細資訊，請參閱 Amazon Linux 套件支援陳述式和 Amazon Corretto FAQs。

AL2023 中的 Node.js

Node.js AL2023 中的 是以版本 20、22 和 24 表示。Amazon Linux 遵循上游支援排程，而且任何 Node.js 版本的支援狀態都可以在套件支援狀態頁面上檢查。所有支援的Node.js版本都是命名空間，並且可以同時安裝在相同的系統上。Namespacing 可確保每個Node.js安裝在檔案系統中都是唯一的。這可透過根據執行時間版本重新命名金鑰目錄和檔案來實現。實際可執行檔名稱看起來會像 node-{MAJOR_VERSION} 或 npm-{MAJOR_VERSION}。不過，一次只能有一個Node.js版本處於作用中狀態。此作用中版本提供預設目錄和檔案名稱，例如節點、npm 或 /usr/lib/node_modules，將它們指向目前作用中的執行時間。

這是使用替代工具的功能來實現的。請務必記住，預設可執行檔名稱是虛擬的，當指向不同的已安裝Node.js版本時， 可能會隨時變更。此彈性可讓軟體使用 shebang 中的節點，在調用時選取所需的版本。不過，當需要特定版本的 Node.js 時，可以透過呼叫命名空間可執行檔 （ 例如 node-20 或 node-22) 來實現該版本的持久性，該執行期一律使用指定的版本。此外，npm 工具的命名空間可執行檔，例如 npm-20 或 npm-22，一律與對應的 Node.js 版本相關聯，無論目前作用中的執行時間為何。

Node.js 以數個以 "nodejs{MAJOR_VERSION}" 開頭的命名空間套件來分佈。這些套件提供節點、相容版本的 npm 工具、文件、程式庫等。例如， 和 nodejs22-npm套件分別提供 Node.js 22 的節點nodejs22和 npm。

替代工具提供單一命令以在Node.js版本之間切換。根據預設，替代方案會設定為處於自動模式，這會使用優先順序來判斷目前作用中的Node.js版本。不過，您可以隨時啟用任何已安裝的版本。目前，所有支援的 版本Node.js具有相同的優先順序，這表示將自動啟用第一個安裝的版本。

使用替代方案的一些實用範例

1. 檢查針對 設定哪些替代方案

```
alternatives --list
```

2. 檢查節點目前的組態

```
alternatives --display node
```

3. 以互動方式變更Node.js版本

```
alternatives --config node
```

4. 切換到手動模式並選取特定版本

```
alternatives --set node /usr/bin/node-{MAJOR_VERSION}
```

5. 切換回自動版本選擇模式

```
alternatives --auto node
```

AL2023 中的 Perl

AL2023 提供 5.32 版的Perl程式設計語言。

雖然在過去幾十年間Perl， 提供 5 Perl 個版本中高度的語言相容性，但 Amazon Linux 預期不會在 AL2023 版本期間從 Perl 5.32 開始移動。 AL2023 Amazon Linux 會根據我們的套件支援陳述式，在 AL2023 的生命週期Perl內繼續執行安全修補程式。

AL2023 中的 Perl 模組

AL2023 中將各種Perl模組封裝為 RPMs。雖然有許多Perl模組可用為 RPMs，但 Amazon Linux 的目標不是封裝每個可能的Perl模組。其他作業系統 RPMs模組，因此 Amazon Linux 會將這些安全修補程式優先於純功能更新。

AL2023 也包含 ，CPAN以便Perl開發人員可以針對Perl模組使用自有套件管理員。

AL2023 中的 PHP

AL2023 目前提供PHP程式設計語言 8.1、8.2、8.3、8.4 和 8.5 版。每個版本都支援與上游 相同的期間PHP。如需詳細資訊，請參閱套件支援陳述式。

從舊 PHP 版本遷移

上游PHP社群整合了完整的遷移文件以供移動：

- 從 PHP 8.4.x 到 PHP 8.5.x
- 從 PHP 8.3.x 到 PHP 8.4.x
- 從 PHP 8.2.x 到 PHP 8.3.x
- 從 PHP 8.1.x 到 PHP 8.2.x
- 從 PHP 8.0.x 到 PHP 8.1.x

AL2 包含 PHP 8.0、8.1 和 8.2，`amazon-linux-extras`可讓您輕鬆升級路徑至 AL2023。

從 PHP 7.x 版本遷移

> **Note**
>
> PHP 專案會維護支援版本的清單和排程，以及不支援分支的清單。
> AL2023 發行時，PHP社群PHP不支援 的所有 7.x 和 5.x 版本，且未包含在 AL2023 中做為選項。

上游PHP社群整合了完整的遷移文件，以便從 PHP 7.4 移至 PHP 8.0。結合上一節中關於遷移至 PHP 8.1 和 8.2 PHP 參考的文件，您可以將 PHP 型應用程式遷移至現代 。 PHP

> **Note**
>
> AL2 在 中包含 PHP 7.1、7.2、7.3 和 7amazon-linux-extras.4。請務必注意，所有額外項目皆為服務終止狀態，且不保證獲得任何進一步的安全更新。

AL2023 中的 PHP 模組

AL2023 包含許多包含在 PHP Core 中的PHP模組。AL2023 的目標不是在PHP延伸社群程式庫 (PECL) 中包含所有套件。

AL2023 中的 Python

AL2023 已移除 Python 2.7，且任何需要 的元件現在Python都會寫入以使用 Python 3。

AL2023 提供 Python 3 `/usr/bin/python3` 以保持與客戶程式碼的相容性，以及 AL2023 隨附的 Python 程式碼，這在 AL2023 Python 的生命週期內將保持為 3.9。

`/usr/bin/python3` 指向 的 Python 版本會被視為系統 Python，而對於 AL2023，這是 Python 3.9。

較新版本的 Python，例如 Python 3.11，可在 AL2023 中做為套件提供，並在上游版本的生命週期內受到支援。如需有關支援 Python 3.11 的時間長度的資訊，請參閱 Python 3.11。

AL2023 可以同時安裝多個版本的 Python。雖然 `/usr/bin/python3` 一律是 Python 3.9，但每個版本的 Python 都會命名空間，並可透過其版本編號找到。例如，如果已安裝 `python3.11`，則 `/usr/bin/python3.11` 將與 `/usr/bin/python3.9` 同時存在，且 `/usr/bin/python3` symlink 至 `/usr/bin/python3.9`。

> **Note**
>
> 請勿變更符號連結指向的 ，因為這可能會破壞 `/usr/bin/python3` AL2023 的核心功能。

AL2023 中的 Python 模組

AL2023 中將各種Python模組封裝為 RPMs。一般而言，Python 模組的 RPM 只會針對 Python 的系統版本建置。

AL2023 中的 Ruby

Ruby AL2023 中的 是以 3.2 和 3.4 版表示。Amazon Linux 遵循上游支援排程，且任何Ruby版本的支援狀態都可以在 Ruby 網站上檢查。所有支援的Ruby版本都是命名空間，並且可以同時安裝在相同的系統上。Namespacing 可確保每個Ruby安裝在檔案系統中都是唯一的。這可透過根據執行時間版本重新命名金鑰目錄和檔案來實現。實際可執行檔名稱看起來像 ruby{MAJOR.MINOR}（例如 `ruby3.2`或 `ruby3.4`)。 Ruby 3.4 也提供 MRI (Matz 的Ruby解譯器） 命名空間二進位 `ruby3.4-mri`，其參考的標準 C 型參考實作Ruby。不過，一次只能有一個Ruby版本處於作用中狀態。此作用中版本提供預設目錄和檔案名稱，例如 ruby、gem 或 bundle，將它們指向目前作用中的執行時間。

這是使用替代工具的功能來實現的。請務必記住，預設可執行檔名稱是虛擬的，當指向不同的已安裝Ruby版本時， 可能會隨時變更。此彈性可讓在 shebang 中使用 ruby 的軟體在調用時選取所需的版本。不過，當需要特定版本的 Ruby 時，可以透過呼叫命名空間可執行檔（例如， `ruby3.2`或 `ruby3.4`) 來實現該版本的持久性，這將始終使用指定的執行時間版本。此外，Gem 和 Bundler 工具的命名空間可執行檔，例如 `ruby3.2-gem`、`ruby3.2-bundler`、 `ruby3.4-gem`或 `ruby3.4-bundler`，一律與對應的Ruby版本相關聯，無論目前作用中的執行時間為何。

Ruby 以數個以 "`ruby{MAJOR.MINOR}`" 開頭的命名空間套件來分佈。這些套件提供 Gem 套件和套件工具、文件、程式庫等的 Ruby 相容版本。例如，核心 Ruby 3.2 執行期是由 `ruby3.2`套件提供，它會提取 `ruby3.2-rubygems`（提供 Gem) 和 `ruby3.2-rubygem-bundler`（提供 bundle 和 bundler) 做為相依性。

安裝Ruby版本後，搭配工具的項目可能會在替代組態中顯示為 null。這可以透過執行 進行驗證`alternatives --display ruby`。如果項目顯示為 null，則必須使用 手動註冊`alternatives --install`。例如，若要註冊 3.4 Ruby 的所有配套工具：

```
sudo alternatives --install /usr/bin/gem gem /usr/bin/ruby3.4-gem 34
sudo alternatives --install /usr/bin/bundle bundle /usr/bin/ruby3.4-bundle 34
sudo alternatives --install /usr/bin/bundler bundler /usr/bin/ruby3.4-bundler 34
sudo alternatives --install /usr/bin/erb erb /usr/bin/ruby3.4-erb 34
sudo alternatives --install /usr/bin/racc racc /usr/bin/ruby3.4-racc 34
sudo alternatives --install /usr/bin/rdoc rdoc /usr/bin/ruby3.4-rdoc 34
sudo alternatives --install /usr/bin/ri ri /usr/bin/ruby3.4-ri 34
```

優先順序值（例如，3.4 為 Ruby 34，3.2 為 Ruby 32) 應與主要 Ruby 替代項目中使用的優先順序相符。註冊後，配套工具將與 Ruby 替代方案一起自動管理。

替代工具提供單一命令以在Ruby版本之間切換。根據預設，替代方案會設定為處於自動模式，這會使用優先順序來判斷目前作用中的Ruby版本。不過，您可以隨時啟用任何已安裝的版本。目前，所有支援的 版本Ruby具有相同的優先順序，這表示將自動啟用第一個安裝的版本。

使用替代方案的一些實用範例

1. 檢查針對 設定哪些替代方案

```
alternatives --list
```

2. 檢查 ruby 目前的組態

```
alternatives --display ruby
```

3. 以互動方式變更Ruby版本

```
alternatives --config ruby
```

4. 切換到手動模式並選取特定版本

```
alternatives --set ruby /usr/bin/ruby{MAJOR.MINOR}
```

5. 切換回自動版本選擇模式

```
alternatives --auto ruby
```

AL2023 中的 Rust

您可能想要建置在 Amazon Linux Rust上寫入的程式碼，並可能想要使用 AL2023 隨附的工具鏈。

與 AL2, AL2023會在作業系統的整個生命週期中更新Rust工具鏈。這可能是為了回應隨附工具鏈中的任何 CVE，或是包含在季度版本中。

Rust 是一種相對快速的語言，新版本約每六週推出一次。這些版本可能會新增語言或標準資料庫功能。雖然 AL2023 會在其生命週期內納入新版本的Rust工具鏈，但這不會與上游Rust版本鎖定。因此，如果您想要使用Rust語言的尖端功能建置Rust程式碼，則可能不適合使用 AL2023 中提供Rust的工具鏈。

在 AL2023 的壽命週期內，舊的套件版本不會從儲存庫中移除。如果需要較舊Rust的工具鏈，您可以選擇放棄較新Rust工具鏈的錯誤和安全性修正，並使用任何 RPM 可用的相同機制從儲存庫安裝較舊的版本。

如果您想要在 AL2023 上建置自己的Rust程式碼，您可以使用 AL2023 中包含Rust的工具鏈，並知道此工具鏈可能會在 AL2023 的生命週期內向前移動。

AL2023 Lambda 函數寫入 Rust

由於 Rust會編譯至原生程式碼，因此 Lambda 會將 Rust視為自訂執行時間。您可以使用 `provided.al2023` 執行期在 AL2023 上將Rust函數部署至 Lambda。

如需詳細資訊，請參閱《 AWS Lambda 開發人員指南》中的使用 建置 Lambda 函數Rust。

AL2023 中的 TypeScript

> **Note**
>
> 本文件提供有關 TypeScript 及其 Node.js 型執行環境的基本資訊。它還涵蓋典型的開發工作流程，並說明 TypeScript 如何封裝在 AL2023 中，以提供一致且可重現的開發環境。

TypeScript (TS) 是一種以 JavaScript (JS) 為基礎的程式設計語言，可提供所有 JS 的功能，也可以使用類型系統進行擴展。在典型案例中，以 TS 撰寫的程式會先翻譯成 JS 程式碼，然後以 的任何其他一般 JS 程式執行Node.js。在 TS 的特定術語中，此轉譯程序稱為「編譯」，並由稱為 tsc 的「編譯器」執行。 tsc 編譯器本身是以 JS 撰寫，因此執行時也需要 JS 執行時間環境，例如 Node.js。與一些其他 JS 執行期環境不同， Node.js 目前只有實驗性和輕量型 TS 支援。完整的 TS 支援，包括類型檢查，仍然需要使用第三方套件，例如 typescript。取得Node.js執行時間環境 tsc (TS 編譯器） 的預期方法是安裝 typescript 節點模組。這可以使用其中一個套件管理員來完成，通常是 npm。有兩種方式可以使用 npm 安裝 TS 編譯器：全域和專案中。官方建議的方法是根據專案安裝 TS 編譯器，以確保專案的長期一致性和可重複性。不過，全域安裝 TS 編譯器可能仍然很有用，因為它為整個主機及其 JS 執行時間提供相同的版本，因此對於本機未安裝 TS 編譯器的專案。這是 Amazon Linux 上可用的 RPM 套件，例如 `nodejs20-typescript`或 `nodejs22-typescript`、在系統層級全域安裝 TS 編譯器，以及針對每個支援的Node.js版本個別安裝的方式。

tsc 不直接依賴任何Node.js版本。編譯器預期執行時間功能有特定層級，透過目標和 lib 等選項在特殊檔案中定義 (tsconfig.json)。這些選項的值代表 ECMAScript (ES) 標準的版本，JS 執行期環境可能支援（也可能不支援）。不同版本的 Node.js支援不同版本的 ES 標準。的 版本越新Node.js，支援的 ES 標準版本就越高且更完整。如果專案根目錄中不存在 tsconfig.json，則會使用預設的組態選項集。

具有不同 版本的相容性資料表Node.js和各種 ES 標準版本的支援功能可在 node.green 取得。tsc 有超過 100 個不同的選項，可在 tsconfig.json 中定義。當在另一個檔案中定義一些組態選項，然後包含在主要檔案中時，也支援組態鏈結。此方法允許安裝與特定 版本相容的 Base TS ConfigNode.js，然後使用專案特定的選項進行擴展。幸運的是， 的基礎 TS Config Node.js可作為節點模組使用，可以使用 npm 安裝在專案資料夾中。以下是組態的 18、20 和 22 Node.js版原始碼。

Node.js 基於 的執行時間設計有一定的弱點：它在主機上僅支援一個版本的執行時間，並且需要專案層級所有相依性的重現性和一致性。這導致以下常見的 TypeScript 使用方法：TS 編譯器、目前Node.js版本的 TS 基礎組態，以及所有軟體相依性都安裝在專案內的本機。雖然全域安裝的節點模組預期只是 CLI 工具，例如 npm、tsc，這也是 CLI 工具，但很少全域安裝。值得注意的是，tsc 的全域（全系統）和本機（專案內）安裝可以共存，沒有問題，也可以是獨立使用的不同版本。請注意，本機安裝的 tsc 應使用與 npm 一起安裝的 npx 工具執行。因此，即使使用系統 TS 編譯器，使用者仍有機會選擇執行時間元件的版本，例如 Node.js（透過透過替代方案切換作用中版本）、TS 編譯器（透過在本機安裝，或透過替代方案切換作用中版本），以及針對特定需求進行設定。

Amazon Linux 在每個Node.js版本基本上封裝 TS 編譯器的方式與其他全域安裝的節點模組相同，例如 npm。套件和二進位檔是命名空間，並包含 的主要版本Node.js做為其名稱的一部分。編譯器的預設可執行檔名稱 tsc 會在執行時間由替代工具管理，並指向目前作用中版本的 Node.js，其已停用且將由 執行。此選擇不會因目前的Node.js執行時間版本而減少。您可以讓節點可執行檔指向 Node.js 20，並將 tsc 設定為由 22 Node.js 解譯。您也可以獨立使用 TS 編譯器的命名空間名稱，例如 tsc-{MAJOR_VERSION}，以設定預設 tsc 名稱。

管理 TS 編譯器作用中版本的一些實用命令

1. 檢查針對 設定哪些替代方案

```
alternatives --list
```

2. 檢查 tsc 目前的組態

```
alternatives --display tsc
```

3. 以互動方式變更tsc版本

```
alternatives --config tsc
```

4. 切換到手動模式並選取特定版本

```
alternatives --set tsc /usr/bin/tsc-{MAJOR_VERSION}
```

5. 切換回自動版本選擇模式

```
alternatives --auto tsc
```

在同一系統上安裝和使用多個版本的 Node 和 TS 編譯器的範例：

```
# Check the AL2023 release
$ cat /etc/amazon-linux-release
Amazon Linux release 2023.9.20250929 (Amazon Linux)

# Install a TypeScript compiler for Node.js 20 and 22
# Node.js 20 and 22 will be installed automatically
$ sudo dnf install -qy nodejs20-typescript nodejs22-typescript

# Check what was installed
$ rpm -q nodejs20 nodejs20-typescript nodejs22 nodejs22-typescript
nodejs20-20.19.5-1.amzn2023.0.1.x86_64
nodejs20-typescript-5.9.2-1.amzn2023.0.1.noarch
nodejs22-22.19.0-1.amzn2023.0.1.x86_64
nodejs22-typescript-5.9.2-1.amzn2023.0.1.noarch

# Check the active version of Node.js - it is version 20
$ alternatives --display node
node - status is auto.
 link currently points to /usr/bin/node-20
/usr/bin/node-20 - priority 100
 slave npmrc: /usr/lib/nodejs20/lib/node_modules/npm/npmrc
 slave npm: /usr/bin/npm-20
 slave npx: /usr/bin/npx-20
 slave node_modules: /usr/lib/nodejs20/lib/node_modules
/usr/bin/node-22 - priority 100
 slave npmrc: /usr/lib/nodejs22/lib/node_modules/npm/npmrc
 slave npm: /usr/bin/npm-22
 slave npx: /usr/bin/npx-22
 slave node_modules: /usr/lib/nodejs22/lib/node_modules
Current 'best' version is /usr/bin/node-20.

# Check the active JS runtime version for TypeScript
# Currently, the tsc compiler will be executed by Node.js 22
$ alternatives --display tsc
tsc - status is auto.
```

```
 link currently points to /usr/bin/tsc-22
/usr/bin/tsc-22 - priority 100
 slave tsserver: /usr/bin/tsserver-22
/usr/bin/tsc-20 - priority 100
 slave tsserver: /usr/bin/tsserver-20
Current 'best' version is /usr/bin/tsc-22.

# Check versions printed by executables
$ node -v
v20.19.5

$ tsc -v
Version 5.9.2

# while the node is 20, tsc is executed by node 22 anyway
$ head -1 /usr/bin/tsc
#!/usr/bin/node-22

# However, instead of default executable names, e.g. node or tsc,
# we can use namespaced names to target any installed version
$ node-20 -v
v20.19.5

$ node-22 -v
v22.19.0

$ tsc-20 -v
Version 5.9.2

$ tsc-22 -v
Version 5.9.2

$ head -1 /usr/bin/tsc-20
#!/usr/bin/node-20

$ head -1 /usr/bin/tsc-22
#!/usr/bin/node-22
```

AL2023 預留使用者和群組

AL2023 會在佈建映像和安裝特定套件期間預先配置特定使用者和群組。使用者、群組及其相關聯的 UIDs和 GIDs會列在此處，以防止衝突。

主題

AL2023 預留使用者清單

使用者名稱	UID
根	0
bin	1
協助程式	2
廣告	3
lp	4
同步	5
shutdown	6
停止	7
郵件	8
operator	11
遊戲	12
ftp	14
squid	23

使用者名稱	UID
名為	25
postgres	26
mysql	27
nscd	28
nscd	28
rpcuser	29
rpc	32
mailnull	47
apache	48
smmsp	51
tomcat	53
ldap	55
tss	59
nslcd	65
avahi	70
tcpdump	72
sshd	74
radvd	75
dbus	81
postfix	89

使用者名稱	UID
鴿魚	97
stapusr	156
stapsys	157
stapdev	158
avahi-autoipd	170
脈衝	171
rtkit	172
sanlock	179
systemd-network	192
systemd-resolve	193
烏伊德	961
stap-server	962
systemd-journal-remote	963
redis6	970
pesign	971
smtpq	972
smtpd	973
nginx	974
munge	975
memcached	976

使用者名稱	UID
括號	977
haproxy	978
flatpak	979
debuginfod	980
dovenull	981
dnsmasq	982
取消繫結	983
蛤蜊	984
clamilt	985
更新	986
著色	987
ods	988
aws-kinesis-agent-user	989
美術士	990
cwagent	991
polkitd	992
ec2-instance-connect	993
chrony	994
systemd-timesync	995
systemd-coredump	996

使用者名稱	UID
libstoragemgmt	997
systemd-oom	999
ec2-user	1000
沒有人	65534

依名稱列出

使用者名稱	UID
廣告	3
apache	48
avahi	70
avahi-autoipd	170
aws-kinesis-agent-user	989
bin	1
chrony	994
clamilt	985
蛤蜊	984
更新	986
著色	987
cwagent	991
協助程式	2

使用者名稱	UID
dbus	81
debuginfod	980
dnsmasq	982
鴿魚	97
dovenull	981
ec2-instance-connect	993
ec2-user	1000
flatpak	979
ftp	14
遊戲	12
停止	7
haproxy	978
ldap	55
libstoragemgmt	997
lp	4
郵件	8
mailnull	47
memcached	976
munge	975
mysql	27

使用者名稱	UID
名為	25
nginx	974
沒有人	65534
nscd	28
nscd	28
nslcd	65
ods	988
operator	11
pesign	971
polkitd	992
postfix	89
postgres	26
脈衝	171
radvd	75
redis6	970
根	0
rpc	32
rpcuser	29
rtkit	172
sanlock	179

使用者名稱	UID
美術士	990
shutdown	6
smmsp	51
smtpd	973
smtpq	972
括號	977
squid	23
sshd	74
stap-server	962
stapdev	158
stapsys	157
stapusr	156
同步	5
systemd-coredump	996
systemd-journal-remote	963
systemd-network	192
systemd-oom	999
systemd-resolve	193
systemd-timesync	995
tcpdump	72

使用者名稱	UID
tomcat	53
tss	59
取消繫結	983
烏伊德	961

AL2023 預留群組清單

Group name (群組名稱)	GID
根	0
bin	1
協助程式	2
sys	3
廣告	4
tty	5
磁碟	6
磁碟	6
lp	7
mem	8
kmem	9
滾輪	10
cdrom	11

Group name (群組名稱)	GID
郵件	12
郵件	12
man	15
撥出	18
鬆軟	19
遊戲	20
配置	21
utmp	22
squid	23
名為	25
postgres	26
mysql	27
nscd	28
nscd	28
rpcuser	29
rpc	32
磁帶	33
utempter	35
kvm	36
視訊	39

Group name (群組名稱)	GID
mailnull	47
apache	48
ftp	50
smmsp	51
tomcat	53
鎖定	54
ldap	55
tss	59
音訊	63
avahi	70
tcpdump	72
sshd	74
radvd	75
巫士	76
dbus	81
screen	84
wbpriv	88
postfix	89
postdrop	90
鴿魚	97

Group name (群組名稱)	GID
使用者	100
input	104
轉譯	105
sgx	106
模擬	135
stapusr	156
stapusr	156
stapsys	157
stapsys	157
stapdev	158
stapdev	158
avahi-autoipd	170
脈衝	171
rtkit	172
sanlock	179
systemd-journal	190
systemd-network	192
systemd-resolve	193
usbmon	959
wireshark	960

Group name (群組名稱)	GID
烏伊德	961
stap-server	962
systemd-journal-remote	963
使用者共用	964
redis6	965
pesign	966
smtpq	967
smtpd	968
nginx	969
munge	970
memcached	971
括號	972
追蹤	973
haproxy	974
flatpak	975
debuginfod	976
dovenull	977
dnsmasq	978
取消繫結	979
蛤蜊	980

Group name (群組名稱)	GID
clamilt	981
病毒群組	982
病毒群組	982
病毒群組	982
更新	983
printadmin	984
著色	985
ods	986
docker	987
aws-kinesis-agent-user	988
cwagent	989
pulse-rt	990
脈衝存取	991
ec2-instance-connect	993
chrony	994
systemd-timesync	995
systemd-coredump	996
libstoragemgmt	997
ssh_keys	998
systemd-oom	999

Group name (群組名稱)	GID
ec2-user	1000
ne	1001
polkitd	9920
沒有人	65534

依名稱列出

Group name (群組名稱)	GID
廣告	4
apache	48
音訊	63
avahi	70
avahi-autoipd	170
aws-kinesis-agent-user	988
bin	1
cdrom	11
chrony	994
clamilt	981
蛤蜊	980
更新	983
著色	985

Group name (群組名稱)	GID
cwagent	989
協助程式	2
dbus	81
debuginfod	976
撥出	18
磁碟	6
磁碟	6
dnsmasq	978
docker	987
鴿魚	97
dovenull	977
ec2-instance-connect	993
ec2-user	1000
flatpak	975
鬆軟	19
ftp	50
遊戲	20
haproxy	974
input	104
kmem	9

Group name (群組名稱)	GID
kvm	36
ldap	55
libstoragemgmt	997
鎖定	54
lp	7
郵件	12
郵件	12
mailnull	47
man	15
mem	8
memcached	971
模擬	135
munge	970
mysql	27
名為	25
ne	1001
nginx	969
沒有人	65534
nscd	28
nscd	28

Group name (群組名稱)	GID
ods	986
pesign	966
polkitd	9920
postdrop	90
postfix	89
postgres	26
printadmin	984
脈衝	171
脈衝存取	991
pulse-rt	990
radvd	75
redis6	965
轉譯	105
根	0
rpc	32
rpcuser	29
rtkit	172
sanlock	179
巫士	76
screen	84

Group name (群組名稱)	GID
sgx	106
配置	21
smmsp	51
smtpd	968
smtpq	967
括號	972
squid	23
ssh_keys	998
sshd	74
stap-server	962
stapdev	158
stapdev	158
stapsys	157
stapsys	157
stapusr	156
stapusr	156
sys	3
systemd-coredump	996
systemd-journal	190
systemd-journal-remote	963

Group name (群組名稱)	GID
systemd-network	192
systemd-oom	999
systemd-resolve	193
systemd-timesync	995
磁帶	33
tcpdump	72
tomcat	53
追蹤	973
tss	59
tty	5
取消繫結	979
usbmon	959
使用者	100
使用者共用	964
utempter	35
utmp	22
烏伊德	961
視訊	39
病毒群組	982
病毒群組	982

Group name (群組名稱)	GID
病毒群組	982
wbpriv	88
滾輪	10
wireshark	960

AL2023 中可用的轉碼器清單

AL2023 透過其標準儲存庫提供多媒體轉碼器的選擇。此頁面提供編解碼器及其典型使用案例的概觀。

> **Important**
>
> 使用和分發 Amazon Linux 中包含的轉碼器可能需要您從第三方取得授權權利，包括特定第三方音訊和視訊格式的擁有者或授權方。您全權負責取得這些授權，並支付任何必要的權利金或費用。

轉碼器	Description
flac	免費的開放原始碼無失真音訊轉碼器，可壓縮音訊而不會遺失任何資料或品質，通常用於高品質音訊儲存
fdk-aac-free	AAC（進階音訊編解碼器）標準的開放原始碼實作，可為串流或檔案儲存等 MP3 替代方案提供高品質的音訊壓縮
webrtc-audio-processing	WebRTC (Web 即時通訊）中使用的音訊處理程式庫，提供雜訊抑制、回音消除和獲得控制等功能
opus	高度多樣化且高效的音訊轉碼器，專為即時串流而設計，可提供低延遲，並支援各種音訊應用程式，包括 VoIP 和音樂串流
libsndfile	一種程式庫，用於讀取和寫入各種格式的音訊檔案（例如 WAV、AIFF 和 FLAC)，常用於音訊處理和操作工具
openh264	H.264 視訊轉碼器的開放原始碼實作。它為串流、會議和多媒體應用程式中使用的 H.264 視訊提供編碼和解碼支援。

轉碼器	Description
svt-av1	AV1 視訊轉碼器的開放原始碼、高效能實作。它為 AV1 影片提供可擴展的編碼和解碼，並針對現代 CPUs和平行處理進行最佳化。
dav1d	AV1 影片解碼器的開放原始碼實作，著重於速度、效率和可攜性。它為各種平台和應用程式提供高效能 AV1 解碼。
libde265	H.265/HEVC 視訊解碼器的開放原始碼實作。它提供 HEVC 視訊串流的高效和可攜式解碼，可用於多媒體應用程式和架構。
libheif	用於讀取和寫入 HEIF 和 AVIF 影像檔案的開放原始碼程式庫。它使用 HEVC 和 AV1 等現代壓縮格式，提供有效的影像和影像序列編碼、解碼和轉換。
mpg123	MPEG 音訊串流的高效能解碼器，包括 MP3 檔案。它為播放、串流和音訊處理應用程式提供快速且準確的音訊解碼。
x265	x265 的主要目標是成為任何地方可用的最佳 H.265/HEVC 編碼器，在各種硬體平台上提供最高壓縮效率和最高效能。此套件包含命令列編碼器。

Amazon Linux 2023 的安全與合規

> **Important**
>
> 如果您想要報告漏洞或對 AWS 雲端服務或開放原始碼專案有安全疑慮，請使用漏洞報告頁面聯絡 AWS 安全部門

的雲端安全 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，該架構旨在滿足最安全敏感組織的需求。

安全是 AWS 與您之間的共同責任。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 Cloud 中執行 AWS 服務的基礎設施。 AWS 也為您提供可安全使用的服務。在AWS 合規計畫中，第三方稽核人員會定期測試和驗證我們安全的有效性若要了解適用於 AL2023 的合規計劃，請參閱AWS 合規計劃的 服務範圍。
- 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。

主題

AL2023 的 Amazon Linux 安全建議

儘管我們努力確保 Amazon Linux 的安全，但有時會出現需要修正的安全問題。當修正可用時，會發出諮詢。我們發佈建議的主要位置是 Amazon Linux 安全中心 (ALAS)。如需詳細資訊，請參閱 Amazon Linux 安全中心。

> **Important**
>
> 如果您想要報告漏洞或對 AWS 雲端服務或開放原始碼專案有安全疑慮，請使用漏洞報告頁面聯絡 AWS 安全部門

Amazon Linux 團隊會在數個位置發佈問題的相關資訊和影響 AL2023 的相關更新。安全工具通常會從這些主要來源擷取資訊並將結果呈現給您。因此，您可能不會直接與 Amazon Linux 發佈的主要來源互動，而是與您偏好工具提供的界面互動，例如 Amazon Inspector。

Amazon Linux 安全中心公告

Amazon Linux 公告會針對不符合建議的項目提供。本節包含有關 ALAS 本身的公告，以及不適合諮詢的資訊。如需詳細資訊，請參閱 Amazon Linux 安全中心 (ALAS) 公告。

例如，適用於 Apache Log4j 的 2021-001 - Amazon Linux Hotpatch 公告會納入公告，而非諮詢。在此公告中，Amazon Linux 新增了套件，協助客戶緩解軟體中不屬於 Amazon Linux 的安全問題。

Amazon Linux 安全中心 CVE Explorer 也在 ALAS 公告中宣布。如需詳細資訊，請參閱 CVEs 的新網站。

Amazon Linux 安全中心常見問答集

如需有關 ALAS 和 Amazon Linux 如何評估 CVEs 的一些常見問題解答，請參閱 Amazon Linux 安全中心 (ALAS) 常見問答集 (FAQs)。

ALAS Advisories

Amazon Linux 諮詢包含與 Amazon Linux 使用者相關的重要資訊，通常是有關安全性更新的資訊。Amazon Linux 安全中心是在 Web 上顯示 Advisories 的位置。諮詢資訊也是 RPM 套件儲存庫中繼資料的一部分。

建議和 RPM 儲存庫

Amazon Linux 2023 套件儲存庫可能包含描述零個或多個更新的中繼資料。`dnf updateinfo` 命令是以包含此資訊的儲存庫中繼資料檔案名稱 命名`updateinfo.xml`。雖然命令名為 `updateinfo`，且中繼資料檔案參考 `update`，但這些都參考屬於諮詢一部分的套件更新。

Amazon Linux Advisories 會在 Amazon Linux 安全中心網站上發佈，以及`dnf`套件管理員參考的 RPM 儲存庫中繼資料中存在的資訊。網站和儲存庫中繼資料最終一致，並且網站和儲存庫中繼資料中的資訊可能不一致。當新的 AL2023 版本正在發行時，通常會發生這種情況。在最新的 AL2023 版本之後，諮詢已更新。

雖然解決此問題的套件更新通常會同時發出新的諮詢，但情況並非總是如此。您可以針對已在發行的套件中解決的新問題建立諮詢。現有的諮詢也可能會使用現有更新所解決的新 CVEs 進行更新。

Amazon Linux 2023 透過 AL2023 上的版本控制儲存庫進行確定性升級的功能表示特定 AL2023 版本的 RPM 儲存庫包含該版本 RPM 儲存庫中繼資料的快照。這包括描述安全性更新的中繼資料。特定 AL2023 版本的 RPM 儲存庫在發行後不會更新。查看舊版 AL2023 RPM 儲存庫時，不會顯示新的或更新的安全性建議。如需如何使用`dnf`套件管理員查看`latest`儲存庫版本或特定 AL2023 版本，請參閱列出適用的建議一節。

諮詢 IDs

每個諮詢都由 參考`id`。目前，Amazon Linux 安全中心網站會將諮詢列為 ALAS-2024-581，而`dnf`套件管理員會將該建議列為 ID 為 ALAS2023-2024-581。如果參考特定諮詢，則需要使用就地套用安全性更新套件管理員 ID。

對於 Amazon Linux，每個作業系統的主要版本都有自己的諮詢 IDs 命名空間。不應對 Amazon Linux Advisory IDs的格式做出任何假設。過去，Amazon Linux 諮詢 IDs遵循的模式`NAMESPACE-YEAR-NUMBER`。`NAMESPACE` 未定義 的完整可能值範圍，但包含 `ALAS`、`ALASCORRETTO8`、`ALAS2023`、`ALASPYTHON3.8`、 `ALAS2`和 `ALASUNBOUND-1.17`。`YEAR` 是建立諮詢的年份，`NUMBER`也是命名空間內的唯一整數。

雖然諮詢 IDs通常是循序的，並依照更新發佈的順序，但有許多原因導致無法做到這一點，因此不應假設這一點。

將諮詢 ID 視為不透明字串，對於每個 Amazon Linux 主要版本都是唯一的。

在 Amazon Linux 2 中，每個 Extra 都位於單獨的 RPM 儲存庫中，而 Advisory 中繼資料僅包含在與其相關的儲存庫中。一個儲存庫的諮詢不適用於另一個儲存庫。在 Amazon Linux Security Center 網站上，目前每個主要 Amazon Linux 版本都有一個 Advisories 清單，而且不會區分為每個儲存庫清單。

由於 AL2023 不使用 Extras 機制來封裝套件的替代版本，因此目前只有兩個 RPM 儲存庫，每個儲存庫都有 Advisories、儲存`core`庫和`livepatch`儲存庫。`livepatch` 儲存庫適用於 AL2023 上的核心即時修補。

諮詢發佈日期和諮詢更新日期

Amazon Linux Advisories 的諮詢發佈日期指出第一次在 RPM 儲存庫中公開提供安全性更新的時間。修正可透過 RPM 儲存庫進行安裝後，建議會立即發佈到 Amazon Linux 安全中心網站。

建議更新日期指出在先前發佈後，何時將新資訊新增至建議。

AL2023 版本編號（例如 2023.6.20241031) 與該版本一起發佈的諮詢發佈日期之間不應有任何假設。

諮詢類型

RPM 儲存庫中繼資料支援不同類型的 Advisories。雖然 Amazon Linux 幾乎只有幾乎通用發行的 Advisories 是安全性更新，但不應擔任此角色。可能會發出錯誤修正、增強功能和新套件等事件的 Advisories，並將 Advisory 標記為包含該類型的更新。

諮詢嚴重性

每個諮詢都有自己的嚴重性，因為每個問題都會分別評估。單一諮詢中可能會處理多個 CVEs，而且每個 CVE 的評估可能不同，但諮詢本身具有一個嚴重性。可能會有多個 Advisories 參照單一套件更新，因此特定套件更新可能會有多個嚴重性（每個 Advisor 一個）。

為了降低嚴重性，Amazon Linux 已使用「重要」、「重要」、「中等」和「低」來表示諮詢的嚴重性。Amazon Linux Advisories 也可能沒有嚴重性，但這種情況非常罕見。

Amazon Linux 是使用「中等」一詞的 RPM 型 Linux 發行版本之一，而有些其他 RPM 型 Linux 發行版本則使用「中等」一詞。Amazon Linux 套件管理員會將這兩個術語視為同等術語，而第三方套件儲存庫可能會使用中一詞。

隨著時間的推移，Amazon Linux Advisories 可以變更嚴重性，因為有更多人了解諮詢中解決的相關問題。

諮詢的嚴重性通常會追蹤諮詢所參考CVEs 的最高 Amazon Linux 評估 CVSS 分數。在某些情況下，這可能不是這種情況。其中一個範例是發生未指派 CVE 的已解決問題。

如需 Amazon Linux 如何使用諮詢嚴重性評分的詳細資訊，請參閱 ALAS 常見問答集。

建議和套件

單一套件可以有許多公告，而且並非所有套件都會發佈公告公告。您可以在多個 Advisories 中參考特定套件版本，每個套件都有自己的嚴重性和 CVEs。

同一個套件更新的多個 Advisories 可以在一個新的 AL2023 版本中同時發出，或快速連續發出。

如同其他 Linux 發行版本，可以有一到多個不同的二進位套件從相同的來源套件建置。例如，ALAS-2024-698 是 Amazon Linux 安全中心網站 AL2023 區段上列為套用至`mariadb105`套件的諮詢。這是來源套件名稱，諮詢本身是指來源套件旁的二進位套件。在此情況下，系統會從一個`mariadb105`來源套件建置十幾個以上的二進位套件。雖然存在與來源套件同名的二進位套件非常常見，但這不是通用的。

雖然 Amazon Linux Advisories 通常會列出從更新後的來源套件建置的所有二進位套件，但不應假設。套件管理員和 RPM 儲存庫中繼資料格式允許 Advisories 列出更新二進位套件的子集。

特定諮詢也可能僅適用於特定 CPU 架構。可能有些套件並非針對所有架構建置，或問題不會影響所有架構。如果套件可在所有架構上使用，但問題僅適用於架構，則 Amazon Linux 通常不會發出僅參考受影響架構的諮詢，但不應假設。

由於套件相依性的性質，諮詢通常會參考一個套件，但安裝該更新將需要其他套件更新，包括未列在諮詢中的套件。`dnf` 套件管理員將負責安裝所需的相依性。

建議和 CVEs

諮詢可以解決零個或多個 CVEs，並且可能有多個參考相同 CVE 的諮詢。

諮詢可能參考零 CVEs的範例是 CVE 尚未（或曾經）指派給問題。

當（例如）CVE 適用於多個套件時，多個 Advisories 可能參考相同 CVE 的範例。例如，CVE-2024-21208 適用於 Corretto 8、11、17 和 21。每個 Corretto 版本都是 AL2023 中的個別套件，且每個套件都有一個諮詢：適用於 Corretto 8 的 ALAS-2024-754、適用於 Corretto 11 的 ALAS-2024-753、適用於 Corretto 17 的 ALAS-2024-752，以及適用於 Corretto 21 的 ALAS-2024-752。雖然這些 Corretto 版本都有相同的 CVEs 清單，但不應假設。

特定 CVE 可以針對不同的套件進行不同的評估。例如，如果在具有重要嚴重性的諮詢中參考特定 CVE，則可能會發出另一個諮詢，參考具有不同嚴重性的相同 CVE。

RPM 儲存庫中繼資料允許每個諮詢的參考清單。雖然 Amazon Linux 通常只參考 CVEs，但中繼資料格式允許其他參考類型。

RPM 套件儲存庫中繼資料只會參考具有可用修正的 CVEs。Amazon Linux 安全中心網站的探索區段包含 Amazon Linux 已評估CVEs 相關資訊。此評估可能會導致各種 Amazon Linux 版本和套件的 CVSS 基本分數、嚴重性和狀態。特定 Amazon Linux 版本或套件的 CVE 狀態可能未受影響、擱置中修正或未規劃修正。在發出諮詢之前，CVEs 的狀態和評估可能會多次變更。這包括重新評估 CVE 對 Amazon Linux 的適用性。

諮詢參考的 CVEs 清單可能會在該諮詢的初始發佈之後變更。

諮詢文字

諮詢也會包含說明問題的文字，或是建立諮詢的原因。此文字通常是未修改的 CVE 文字。此文字可能是指上游版本編號，其中提供與 Amazon Linux 已套用修正的套件版本不同的修正。Amazon Linux 通常會從較新的上游版本恢復連接埠修正。如果諮詢文字提及的上游版本與 Amazon Linux 版本中隨附的版本不同，則諮詢中的 Amazon Linux 套件版本對於 Amazon Linux 而言將準確。

RPM 儲存庫中繼資料中的諮詢文字可能是預留位置文字，只要參考 Amazon Linux 安全中心網站以取得詳細資訊即可。

核心即時修補程式建議

即時修補程式的建議在 中是唯一的，其參考的套件 (Linux 核心） 與諮詢所針對的套件 （ 例如 `kernel-livepatch-6.1.15-28.43`) 不同。

核心即時修補程式的諮詢將參考特定即時修補程式套件可針對套用即時修補程式套件的特定核心版本解決的問題 （ 例如 CVEs)。

每個即時修補程式適用於特定的核心版本。為了套用 CVE 的即時修補程式，需要安裝核心版本的正確即時修補程式套件，並套用即時修補程式。

例如，CVE-2023-6111 可以即時修補 AL2023 核心版本 `6.1.56-82.125`、 `6.1.59-84.139`和 `6.1.61-85.141`。也發行了具有此 CVE 修正程式的新核心版本，並具有單獨的諮詢。為了在 AL20232023 上解決 CVE-2023-6111，核心版本等於或晚於 ALAS2023-2023-461 指定的需要執行，或者具有此 CVE 即時修補程式的其中一個核心版本需要執行並套用適用的即時修補程式。

當已有可用的即時修補程式的特定核心版本有新的即時修補程式時，就會發行新版本的`kernel-livepatch-KERNEL_VERSION`套件。例如，ALASLIVEPATCH-2023-003諮詢是使用 `kernel-livepatch-6.1.15-28.43-1.0-1.amzn2023`套件發行的，其中包含涵蓋三個 CVEs `6.1.15-28.43` 的核心即時修補程式。稍後，使用 `kernel-livepatch-6.1.15-28.43-1.0-2.amzn2023` 套件發出ALASLIVEPATCH-2023-009諮詢；針對

包含其他三個 CVEs 6.1.15-28.43 即時修補程式的核心，更新先前的即時修補程式套件。其他核心版本也有其他即時修補程式 Advisories 問題，其中包含這些特定核心版本的即時修補程式套件。

如需核心即時修補的詳細資訊，請參閱 AL2023 上的核心即時修補。

對於任何開發安全建議相關工具的人，也建議您查看 適用於 Advisories 和 的 XML 結構描述 updateinfo.xml一節以取得更多資訊。

適用於 Advisories 和 的 XML 結構描述 **updateinfo.xml**

updateinfo.xml 檔案是套件儲存庫格式的一部分。這是dnf套件管理員剖析的中繼資料，以實作 列出適用的建議和 等功能就地套用安全性更新。

我們建議使用dnf套件管理員的 API，而不是編寫自訂程式碼來剖析儲存庫中繼資料格式。AL2023 dnf中的 版本可以剖析 AL2023 和 AL2 儲存庫格式，因此 API 可用來檢查任一作業系統版本的諮詢資訊。

RPM 軟體管理專案會在 GitHub 的 rpm 中繼資料儲存庫中記錄 RPM 中繼資料格式。

對於開發工具以直接剖析updateinfo.xml中繼資料的人員，強烈建議注意 rpm 中繼資料文件。文件涵蓋了在萬用字元中看到的內容，其中包括許多您可能合理解釋為中繼資料格式規則的例外狀況。

在 GitHub 的 raw-historical-rpm-repository-examples 儲存庫中，還有一組不斷成長的真實世界updateinfo.xml檔案範例。 raw-historical-rpm-repository-examples

如果文件中有任何不清楚的地方，您可以在 GitHub 專案上開啟問題，以便我們可以回答問題並適當地更新文件。作為開放原始碼專案，也歡迎提取請求更新文件。

列出適用的建議

dnf 套件管理員可以存取中繼資料，描述哪些套件版本中修正了哪些 Advisories。因此，它可以列出適用於執行個體或容器映像的建議。

> **Note**
>
> 等工具AWS Systems Manager可以使用此功能來顯示哪些更新與機群相關，而不只是單一執行個體。

列出更新時，您可以指示 dnf 查看特定 AL2023 版本的中繼資料，或最新版本的中繼資料。

> **Note**
>
> 一旦 AL2023 版本發行，就不會改變。因此，Amazon Linux Security Center 上的新建議或更新建議只會新增至 AL2023 新版本的中繼資料

我們現在將介紹適用於某些 AL2023 容器映像的建議範例。這些命令全都適用於非容器化環境，例如 EC2 執行個體。

Listing advisories in a specific version

在此範例中，我們將查看 2023.1.20230628 版中的哪些建議與 2023.0.20230315 版的容器映像相關。

> **Note**
>
> 此範例使用 2023.0.20230315 和 2023.1.20230628 版本，而且這些不是 AL2023 的最新版本。請參閱 AL2023 版本備註以取得最新版本，其中包含最新的安全性更新。

在此範例中，我們將從 2023.0.20230315 版本的容器映像開始。

首先，我們從容器登錄檔擷取此容器映像。結尾.0的 表示特定版本的映像版本；此映像版本通常為零。

```
$ docker pull public.ecr.aws/amazonlinux/amazonlinux:2023.0.20230315.0
   2023.0.20230315.0: Pulling from amazonlinux/amazonlinux
b76f3b09316a: Pull complete
Digest: sha256:94e7183b0739140dbd5b639fb7600f0a2299cec5df8780c26d9cb409da5315a9
Status: Downloaded newer image for public.ecr.aws/amazonlinux/
amazonlinux:2023.0.20230315.0
public.ecr.aws/amazonlinux/amazonlinux:2023.0.20230315.0
```

我們現在可以在容器內產生 shell，從中我們會要求 dnf列出與容器中安裝的套件相關的建議。

```
$ docker run -it public.ecr.aws/amazonlinux/amazonlinux:2023.0.20230315.0
   bash-5.2#
```

dnf updateinfo 命令現在用於顯示 2023.1.20230628 版本中與我們已安裝套件相關的建議摘要。

```
$ dnf updateinfo --releasever=2023.1.20230628
Amazon Linux 2023 repository                    42 MB/s |  15 MB     00:00
Last metadata expiration check: 0:00:02 ago on Mon Jul 22 20:24:24 2024.
Updates Information Summary: available
    8 Security notice(s)
        1 Important Security notice(s)
        5 Medium Security notice(s)
        2 Low Security notice(s)
```

若要取得建議清單，可將 `--list`選項提供給 `dnf updateinfo`。

```
$ dnf updateinfo --releasever=2023.1.20230628 --list
Last metadata expiration check: 0:01:22 ago on Mon Jul 22 20:24:24 2024.
ALAS2023-2023-193 Medium/Sec.    curl-minimal-8.0.1-1.amzn2023.x86_64
ALAS2023-2023-225 Medium/Sec.    glib2-2.74.7-688.amzn2023.0.1.x86_64
ALAS2023-2023-195 Low/Sec.       libcap-2.48-2.amzn2023.0.3.x86_64
ALAS2023-2023-193 Medium/Sec.    libcurl-minimal-8.0.1-1.amzn2023.x86_64
ALAS2023-2023-145 Low/Sec.       libgcc-11.3.1-4.amzn2023.0.3.x86_64
ALAS2023-2023-145 Low/Sec.       libgomp-11.3.1-4.amzn2023.0.3.x86_64
ALAS2023-2023-145 Low/Sec.       libstdc++-11.3.1-4.amzn2023.0.3.x86_64
ALAS2023-2023-163 Medium/Sec.    libxml2-2.10.4-1.amzn2023.0.1.x86_64
ALAS2023-2023-220 Important/Sec. ncurses-base-6.2-4.20200222.amzn2023.0.4.noarch
ALAS2023-2023-220 Important/Sec. ncurses-libs-6.2-4.20200222.amzn2023.0.4.x86_64
ALAS2023-2023-181 Medium/Sec.    openssl-libs-1:3.0.8-1.amzn2023.0.2.x86_64
ALAS2023-2023-222 Medium/Sec.    openssl-libs-1:3.0.8-1.amzn2023.0.3.x86_64
```

Listing advisories in the latest version

在這個範例中，如果我們啟動了 2023.4. `latest`版本的容器，我們將查看 AL2023 版本中有哪些可用的更新。 AL2023 20240319 撰寫時，`latest`版本為 2023.5.20240708，因此此範例中列出的更新將截至該版本。

> **Note**
>
> 此範例使用 2023.4.20240319 和 2023.5.20240708 版本，後者是撰寫時的最新版本。如需最新版本的詳細資訊，請參閱 AL2023 版本備註。

在此範例中，我們將從 2023.4.20240319 版本的容器映像開始。

首先，我們從容器登錄檔擷取此容器映像。結尾.1的 表示特定版本的映像版本。雖然映像版本通常為零，但此範例使用映像版本為 的版本。

```
$ docker pull public.ecr.aws/amazonlinux/amazonlinux:2023.4.20240319.1
   2023.4.20240319.1: Pulling from amazonlinux/amazonlinux
6de065fda9a2: Pull complete
Digest: sha256:b4838c4cc9211d966b6ea158dacc9eda7433a16ba94436508c2d9f01f7658b4e
Status: Downloaded newer image for public.ecr.aws/amazonlinux/
amazonlinux:2023.4.20240319.1
public.ecr.aws/amazonlinux/amazonlinux:2023.4.20240319.1
```

我們現在可以在容器內產生殼層，並從中檢查更新。

```
$ docker run -it public.ecr.aws/amazonlinux/amazonlinux:2023.4.20240319.1
   bash-5.2#
```

dnf updateinfo 命令現在用於顯示最新版本中與我們已安裝套件相關的建議摘要。撰寫時，2023.1.20230628 是最新版本。

```
$ dnf --releasever=latest updateinfo
Amazon Linux 2023 repository                    76 MB/s |  25 MB     00:00
Last metadata expiration check: 0:00:04 ago on Mon Jul 22 20:59:54 2024.
Updates Information Summary: available
    9 Security notice(s)
        4 Important Security notice(s)
        4 Medium Security notice(s)
        1 Low Security notice(s)
```

若要取得建議清單，可將 --list選項提供給 dnf updateinfo。

```
$ dnf updateinfo --releasever=latest --list
Last metadata expiration check: 0:00:58 ago on Mon Jul 22 20:59:54 2024.
ALAS2023-2024-581 Low/Sec.       curl-minimal-8.5.0-1.amzn2023.0.3.x86_64
ALAS2023-2024-596 Medium/Sec.    curl-minimal-8.5.0-1.amzn2023.0.4.x86_64
ALAS2023-2024-576 Important/Sec. expat-2.5.0-1.amzn2023.0.4.x86_64
ALAS2023-2024-589 Important/Sec. glibc-2.34-52.amzn2023.0.10.x86_64
ALAS2023-2024-589 Important/Sec. glibc-common-2.34-52.amzn2023.0.10.x86_64
ALAS2023-2024-589 Important/Sec. glibc-minimal-langpack-2.34-52.amzn2023.0.10.x86_64
ALAS2023-2024-586 Medium/Sec.    krb5-libs-1.21-3.amzn2023.0.4.x86_64
ALAS2023-2024-581 Low/Sec.       libcurl-minimal-8.5.0-1.amzn2023.0.3.x86_64
ALAS2023-2024-596 Medium/Sec.    libcurl-minimal-8.5.0-1.amzn2023.0.4.x86_64
```

```
ALAS2023-2024-592 Important/Sec. libnghttp2-1.59.0-3.amzn2023.0.1.x86_64
ALAS2023-2024-640 Medium/Sec.    openssl-libs-1:3.0.8-1.amzn2023.0.12.x86_64
ALAS2023-2024-605 Medium/Sec.    python3-3.9.16-1.amzn2023.0.7.x86_64
ALAS2023-2024-616 Important/Sec. python3-3.9.16-1.amzn2023.0.8.x86_64
ALAS2023-2024-605 Medium/Sec.    python3-libs-3.9.16-1.amzn2023.0.7.x86_64
ALAS2023-2024-616 Important/Sec. python3-libs-3.9.16-1.amzn2023.0.8.x86_64
```

就地套用安全性更新

如需套用更新的概觀，請參閱 使用 DNF 和儲存庫版本套用安全更新。的 `--security`選項`dnf upgrade`會將套件更新限制為只有具有 諮詢的套件更新。本節的其餘部分將介紹如何僅安裝特定安全性更新。

> **Note**
>
> 建議套用新 AL2023 版本中提供的所有更新。僅選擇安全性更新，或僅選擇特定更新應該是例外狀況，而不是規則。

套用諮詢中提到的更新

輸出第一欄的諮詢識別符`dnf upgradeinfo`可用來套用諮詢中提及之套件的更新。您可以指示dnf套件管理員將諮詢中的套件更新為最新的可用套件，或更新至諮詢中提及的版本。如果已安裝更新，則更新命令為無操作。

若要僅將受影響套件的更新套用到諮詢中提到的版本，請使用 `dnf upgrade-minimal`命令，同時使用 `--advisory`選項來指定諮詢。下列範例在 AL2023 版本 2023`dnf upgrade-minimal`.0. 容器中執行。 20230315

```
$ dnf upgrade-minimal -y --releasever=2023.1.20230628 --advisory ALAS2023-2023-193
Amazon Linux 2023 repository                    46 MB/s |  15 MB     00:00
Last metadata expiration check: 0:00:03 ago on Mon Jul 22 20:36:13 2024.
Dependencies resolved.
================================================================================
 Package              Arch        Version                Repository        Size
================================================================================
Upgrading:
 curl-minimal         x86_64      8.0.1-1.amzn2023       amazonlinux      150 k
 libcurl-minimal      x86_64      8.0.1-1.amzn2023       amazonlinux      249 k
```

```
Transaction Summary
================================================================================
Upgrade  2 Packages

Total download size: 399 k
Downloading Packages:
(1/2): curl-minimal-8.0.1-1.amzn2023.x86_64.rpm 2.7 MB/s | 150 kB     00:00
(2/2): libcurl-minimal-8.0.1-1.amzn2023.x86_64. 3.8 MB/s | 249 kB     00:00
--------------------------------------------------------------------------------
Total                                           2.5 MB/s | 399 kB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Upgrading        : libcurl-minimal-8.0.1-1.amzn2023.x86_64                1/4
  Upgrading        : curl-minimal-8.0.1-1.amzn2023.x86_64                   2/4
  Cleanup          : curl-minimal-7.88.1-1.amzn2023.0.1.x86_64              3/4
  Cleanup          : libcurl-minimal-7.88.1-1.amzn2023.0.1.x86_64           4/4
  Running scriptlet: libcurl-minimal-7.88.1-1.amzn2023.0.1.x86_64           4/4
  Verifying        : libcurl-minimal-8.0.1-1.amzn2023.x86_64                1/4
  Verifying        : libcurl-minimal-7.88.1-1.amzn2023.0.1.x86_64           2/4
  Verifying        : curl-minimal-8.0.1-1.amzn2023.x86_64                   3/4
  Verifying        : curl-minimal-7.88.1-1.amzn2023.0.1.x86_64              4/4

Upgraded:
  curl-minimal-8.0.1-1.amzn2023.x86_64  libcurl-minimal-8.0.1-1.amzn2023.x86_64

Complete!
```

即使`--releasever=latest`當請求是 dnf執行解決諮詢所需的最低更新時，相同的套件版本也會更新。

搭配 `--advisory`選項使用一般`dnf upgrade`命令，會將諮詢中提到的相關套件更新為可用的最新版本，可能比諮詢中提到的版本更新。

> **Note**
>
> 除非`system-release`套件已更新，否則`dnf`鎖定為 的 AL2023 儲存庫版本不會變更。

> **Warning**
>
> 從不同 AL2023 版本安裝更新時，如果未變更dnf鎖定的儲存庫版本，則必須注意任何後續的變動dnf操作。例如，安裝或更新套件時，由於較新版本中的套件相依性可能已變更，因此您保留的較舊版本可能無法滿足這些新的相依性。

下列範例是在 AL2023 版本 2023.0.20230315 容器中執行，此容器中參照 AL2023 的最新版本，其寫入時間為 2023.5.20240708。請注意，curl更新至 的兩個版本都比update-minimal更新至的版本更新，但此更新版本帶來新的相依性。

```
$ dnf upgrade -y --releasever=latest --advisory ALAS2023-2023-193
Amazon Linux 2023 repository                    80 MB/s |  25 MB     00:00
Last metadata expiration check: 0:00:04 ago on Mon Jul 22 20:48:38 2024.
Dependencies resolved.
================================================================================
 Package                  Arch     Version                 Repository     Size
================================================================================
Upgrading:
 curl-minimal             x86_64   8.5.0-1.amzn2023.0.4    amazonlinux   160 k
 libcurl-minimal          x86_64   8.5.0-1.amzn2023.0.4    amazonlinux   275 k
 libnghttp2               x86_64   1.59.0-3.amzn2023.0.1   amazonlinux    79 k
Installing dependencies:
 libpsl                   x86_64   0.21.1-3.amzn2023.0.2   amazonlinux    61 k
 publicsuffix-list-dafsa  noarch   20240212-61.amzn2023    amazonlinux    59 k

Transaction Summary
================================================================================
Install  2 Packages
Upgrade  3 Packages

Total download size: 634 k
Downloading Packages:
(1/5): publicsuffix-list-dafsa-20240212-61.amzn 1.1 MB/s |  59 kB     00:00
(2/5): curl-minimal-8.5.0-1.amzn2023.0.4.x86_64 2.6 MB/s | 160 kB     00:00
(3/5): libpsl-0.21.1-3.amzn2023.0.2.x86_64.rpm  949 kB/s |  61 kB     00:00
(4/5): libnghttp2-1.59.0-3.amzn2023.0.1.x86_64. 3.7 MB/s |  79 kB     00:00
(5/5): libcurl-minimal-8.5.0-1.amzn2023.0.4.x86 6.7 MB/s | 275 kB     00:00
--------------------------------------------------------------------------------
Total                                           3.5 MB/s | 634 kB     00:00
Running transaction check
Transaction check succeeded.
```

```
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Upgrading        : libnghttp2-1.59.0-3.amzn2023.0.1.x86_64                1/8
  Installing       : publicsuffix-list-dafsa-20240212-61.amzn2023.noarch    2/8
  Installing       : libpsl-0.21.1-3.amzn2023.0.2.x86_64                    3/8
  Upgrading        : libcurl-minimal-8.5.0-1.amzn2023.0.4.x86_64            4/8
  Upgrading        : curl-minimal-8.5.0-1.amzn2023.0.4.x86_64               5/8
  Cleanup          : curl-minimal-7.88.1-1.amzn2023.0.1.x86_64              6/8
  Cleanup          : libcurl-minimal-7.88.1-1.amzn2023.0.1.x86_64           7/8
  Cleanup          : libnghttp2-1.51.0-1.amzn2023.x86_64                    8/8
  Running scriptlet: libnghttp2-1.51.0-1.amzn2023.x86_64                    8/8
  Verifying        : libpsl-0.21.1-3.amzn2023.0.2.x86_64                    1/8
  Verifying        : publicsuffix-list-dafsa-20240212-61.amzn2023.noarch    2/8
  Verifying        : curl-minimal-8.5.0-1.amzn2023.0.4.x86_64               3/8
  Verifying        : curl-minimal-7.88.1-1.amzn2023.0.1.x86_64              4/8
  Verifying        : libcurl-minimal-8.5.0-1.amzn2023.0.4.x86_64            5/8
  Verifying        : libcurl-minimal-7.88.1-1.amzn2023.0.1.x86_64           6/8
  Verifying        : libnghttp2-1.59.0-3.amzn2023.0.1.x86_64                7/8
  Verifying        : libnghttp2-1.51.0-1.amzn2023.x86_64                    8/8

Upgraded:
  curl-minimal-8.5.0-1.amzn2023.0.4.x86_64
  libcurl-minimal-8.5.0-1.amzn2023.0.4.x86_64
  libnghttp2-1.59.0-3.amzn2023.0.1.x86_64
Installed:
  libpsl-0.21.1-3.amzn2023.0.2.x86_64
  publicsuffix-list-dafsa-20240212-61.amzn2023.noarch

Complete!
```

設定 AL2023 的 SELinux 模式

根據預設，安全性增強型 Linux (SELinux) 是 `enabled`，並設定為 AL2023 的 `permissive` 模式。在寬容模式中，會記錄權限遭拒的情形，但不會強制執行。SELinux 是核心功能和公用程式的集合，為核心的主要子系統提供了強大、靈活且強制的存取控制 (MAC) 架構。

SELinux 提供增強的機制，可以根據機密性和完整性需求強制執行資訊分離。這種資訊分離可減少竄改和略過應用程式安全機制的威脅。另外還限制惡意或有缺陷的應用程式可能造成的損壞。

SELinux 包含一組樣本安全政策組態檔案，旨在滿足日常安全目標。

如需 SELinux 特性和功能的詳細資訊，請參閱 SELinux Notebook和政策語言。

主題

AL2023 的預設 SELinux 狀態和模式

對於 AL2023，SELinux 預設是 `enabled`，並設定為 `permissive` 模式。在 `permissive` 模式中，會記錄權限遭拒的情形，但不會強制執行。

`getenforce` 或 **`sestatus`** 命令會告知您目前的 SELinux 狀態、政策和模式。

將預設狀態設為 `enabled` 和 `permissive` 時，**`getenforce`** 命令會傳回 `permissive`。

此**`sestatus`**命令會傳回 SELinux 狀態和目前的 SELinux 政策，如下列範例所示：

```
$ sestatus
SELinux status:                 enabled
  SELinuxfs mount:                /sys/fs/selinux
  SELinux root directory:         /etc/selinux
  Loaded policy name:             targeted
  Current mode:                   permissive
  Mode from config file:          permissive
  Policy MLS status:              enabled
  Policy deny_unknown status:     allowed
  Memory protection checking:     actual (secure)
  Max kernel policy version:      33
```

當您在 `permissive` 模式下執行 SELinux 時，使用者可能會不正確地標記檔案。當您以 `disabled` 狀態執行 SELinux 時，檔案不會被標記。當您變更為 `enforcing` 模式時，不正確或未標記的檔案都可能導致問題。

SELinux 會自動重新標記檔案，以避免此問題。SELinux 會在您將狀態變更為 `enabled` 時，透過自動重新標籤來防止標籤問題。

變更為 **enforcing** 模式

當您SELinux在 enforcing 模式下執行時，SELinux公用程式是enforcing設定的政策。 會根據政策的規則允許或拒絕存取，以SELinux控管特定應用程式的功能。

若要尋找目前的SELinux模式，請執行 getenforce命令。

```
getenforce
Permissive
```

編輯設定檔案以啟用 **enforcing** 模式

若要將 模式變更為 enforcing，請使用下列步驟。

1. 編輯 /etc/selinux/config 檔案以變更為 enforcing 模式。SELINUX 設定應如下所示。

   ```
   SELINUX=enforcing
   ```

2. 重新啟動系統以完成對 enforcing 模式的變更。

   ```
   $ sudo reboot
   ```

在下一次開機時， 會SELinux重新標記系統中的所有檔案和目錄。 SELinux也會新增當 SELinux為 時所建立之檔案和目錄SELinux的內容disabled。

變更為 enforcing 模式後， SELinux可能會因為不正確或缺少SELinux政策規則而拒絕某些動作。您可以使用下列命令檢視SELinux拒絕的動作。

```
$ sudo ausearch -m AVC,USER_AVC,SELINUX_ERR,USER_SELINUX_ERR -ts recent
```

使用 cloud-init 來啟用 **enforcing** 模式

或者，當您啟動執行個體時，請將下列 cloud-config 作為使用者資料傳遞以啟用 enforcing 模式。

```
#cloud-config
selinux:
  mode: enforcing
```

預設情況下，此設定會導致執行個體重新啟動。為了提高穩定性，建議您重新啟動執行個體 但是，您可以依照偏好輸入下列 cloud-config，以跳過重新開機。

```
#cloud-config
selinux:
  mode: enforcing
  selinux_no_reboot: 1
```

停用 SELinux for AL2023 的選項

當您停用 時SELinux，不會載入或強制執行SELinux政策，也不會記錄存取向量快取 (AVC) 訊息。您會失去執行 的所有優點SELinux。

建議您使用 permissive 模式SELinux，而不是停用 。在 permissive 模式下執行的成本比SELinux完全停用的成本高出一點。從 permissive 模式轉換到 enforcing 模式需要的組態調整，遠比停用後轉換回 enforcing 模式少得多SELinux。您可以標記檔案，然後系統可以追蹤和記錄作用中原則可能已拒絕的動作。

SELinux 變更為 **permissive** 模式

當您SELinux在 permissive 模式下執行時，不會強制執行SELinux政策。在 permissive模式下，SELinux會記錄 AVC 訊息，但不拒絕操作。您可以使用這些 AVC 訊息進行故障診斷、偵錯和SELinux政策改進。

若要SELinux變更為允許模式，請使用下列步驟。

1. 編輯 /etc/selinux/config 檔案以變更為 permissive 模式。SELINUX 值應如下所示。

   ```
   SELINUX=permissive
   ```

2. 重新啟動系統以完成對 permissive 模式的變更。

   ```
   sudo reboot
   ```

停用 SELinux

當您停用 時SELinux，不會載入或強制執行SELinux政策，也不會記錄 AVC 訊息。您會失去執行 的所有優點SELinux。

若要停用 SELinux，請使用下列步驟。

1. 確認套件grubby已安裝。

```
rpm -q grubby
grubby-version
```

2. 設定開機載入器，以將 selinux=0 新增到核心命令行。

```
sudo grubby --update-kernel ALL --args selinux=0
```

3. 重新啟動系統。

```
sudo reboot
```

4. 執行 getenforce 命令以確認 SELinux為 Disabled。

```
$ getenforce
Disabled
```

如需 的詳細資訊SELinux，請參閱SELinux筆記本和SELinux組態。

在 AL2023 上啟用 FIPS 模式

本節說明如何在 AL2023 上啟用聯邦資訊處理標準 (FIPS)。如需 FIPS 的詳細資訊，請參閱：

- 美國聯邦資訊處理標準 (FIPS)
- 法規遵循常見問題集：美國聯邦資訊處理標準

> **Note**
>
> 本節說明如何在 AL2023 中啟用 FIPS 模式，但不涵蓋 AL2023 加密模組的認證狀態。

先決條件

- 現有的 AL2023 (AL2023.2 或更新版本) Amazon EC2 執行個體可存取網際網路，以下載所需套件。如需啟動 AL2023 Amazon EC2 執行個體的詳細資訊，請參閱 使用 Amazon EC2 主控台啟動 AL2023。

- 您必須使用 SSH 或 AWS Systems Manager連線至 Amazon EC2 執行個體。如需詳細資訊，請參閱連線至 AL2023 執行個體。

> **Important**
>
> FIPS 模式不支援 ED25519 SSH 使用者金鑰。如果使用 ED25519 SSH 金鑰對啟動 Amazon EC2 執行個體，您必須使用其他演算法 (例如 RSA) 產生新金鑰，否則可能無法在啟用 FIPS 模式後存取執行個體。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的建立金鑰對。

啟用 FIPS 模式

1. 使用 SSH 或 AWS Systems Manager連線至 AL2023 執行個體。
2. 請確保系統為最新版本。如需詳細資訊，請參閱在 AL2023 中管理套件和作業系統更新。
3. 請確保 `crypto-policies` 已安裝公用程式且為最新版本。

   ```
   sudo dnf -y install crypto-policies crypto-policies-scripts
   ```

4. 執行下列命令來啟用 FIPS 模式。這將為 AL2023 常見問答集中列出的模組啟用全系統的 FIPS 模式

   ```
   sudo fips-mode-setup --enable
   ```

5. 使用下列命令重新啟動執行個體。

   ```
   sudo reboot
   ```

6. 若要驗證 FIPS 模式是否已啟用，請重新連線到執行個體，然後執行下列命令：

   ```
   sudo fips-mode-setup --check
   ```

   下列範例的輸出結果顯示 FIPS 模式已啟用：

   ```
   FIPS mode is enabled.
   ```

在 AL2023 容器中啟用 FIPS 模式

本節說明如何在 AL2023 容器中啟用聯邦資訊處理標準 (FIPS)。如需 FIPS 的詳細資訊，請參閱：

- 美國聯邦資訊處理標準 (FIPS)
- 法規遵循常見問題集：美國聯邦資訊處理標準

> **Note**
>
> 本節說明如何在 AL2023 容器中啟用 FIPS 模式。它不涵蓋 AL2023 密碼編譯模組的認證狀態。

先決條件

- 現有的 AL2023 (AL2023.2 或更新版本) Amazon EC2 執行個體可存取網際網路，以下載所需套件。如需啟動 AL2023 Amazon EC2 執行個體的詳細資訊，請參閱 使用 Amazon EC2 主控台啟動 AL2023。
- 您必須使用 SSH 或 AWS Systems Manager連線至 Amazon EC2 執行個體。如需詳細資訊，請參閱連線至 AL2023 執行個體。

> **Important**
>
> `fips-mode-setup` 命令無法從容器內正確運作。請閱讀下列步驟，在 AL2023 容器中正確設定 FIPS 模式。

在 AL2023 容器中啟用 FIPS 模式

1. 必須先在 AL2023 容器主機上啟用 FIPS 模式。依照 的指示在 AL2023 上啟用 FIPS 模式，在主機上啟用 FIPS 模式。
2. 使用 SSH 或 連線至 AL2023 容器主機執行個體 AWS Systems Manager。
3. 如果 AL2023 主機處於 FIPS 模式，且可從容器內存取`/proc/sys/crypto/fips_enabled`，則 AL2023 容器中會自動啟用 FIPS 模式。如果 的內容`/proc/sys/crypto/fips_enabled`為 `0`，則 FIPS 不會啟用，且 值1表示 FIPS 模式已啟用。

   您可以在 AL2023 主機和容器上執行下列命令，以確認 FIPS 已啟用：

```
cat /proc/sys/crypto/fips_enabled
```

4. 接著，在容器內啟用 FIPS 加密政策。有幾種方法可以完成此操作，如以下選項所述。使用最適合您環境的選項。

   a. 使用 update-crypto-policies命令在容器內手動啟用 FIPS 加密政策：

   ```
   # Run these commands inside the container
   dnf install -y crypto-policies-scripts
   update-crypto-policies --set FIPS
   ```

   b. 在 AL2023 容器內建立bind掛載（類似於在其他分發中的podman運作方式）：

   ```
   # Run these commands inside the container
   mount --bind /usr/share/crypto-policies/back-ends/FIPS /etc/crypto-policies/back-
   ends
   echo "FIPS" > /usr/share/crypto-policies/default-fips-config
   mount --bind /usr/share/crypto-policies/default-fips-config /etc/crypto-policies/
   config
   ```

   c. 您也可以建立繫結掛載，讓 AL2023 容器符合 AL2023 主機的加密政策。以下僅做為範例提供。如果容器和主機之間的加密政策和套件版本存在不相容的差異，則此組態可能會導致問題：

   ```
   sudo docker pull amazonlinux:2023
   sudo docker run --mount type=bind,readonly,src=/etc/crypto-policies,dst=/etc/
   crypto-policies -it amazonlinux:2023
   ```

5. 執行上述步驟後，您可以使用下列命令再次確認容器中已啟用 FIPS：

```
$ cat /etc/crypto-policies/config
FIPS

$ cat /proc/sys/crypto/fips_enabled
1
```

在 AL2023 上交換 OpenSSL FIPS 供應商

本節說明如何在 AL2023 上切換 latest和 certified OpenSSL FIPS 供應商。

如需 FIPS 的詳細資訊，請參閱：

- 美國聯邦資訊處理標準 (FIPS)
- 法規遵循常見問題集：美國聯邦資訊處理標準
- 密碼編譯模組選擇和使用的 FedRAMP 政策

> **Important**
>
> 在 AL2023.7 和更高版本上，預設 OpenSSL FIPS 提供者是 `openssl-fips-provider-latest`套件，它會接收定期錯誤修正和安全性更新。
> 以下指示僅適用於想要鎖定`openssl-fips-provider-certified`套件的客戶。此版本的 FIPS 供應商將符合 NIST 憑證上的檢查總和，而且可能沒有最新的更新。
> 如需 FIPS 認證模組和套件版本的詳細資訊，請參閱 AL2023 常見問答集。

先決條件

- 可存取網際網路以下載必要套件的現有 AL2023 (AL2023.7 或更高版本） Amazon EC2 執行個體。如需啟動 AL2023 Amazon EC2 執行個體的詳細資訊，請參閱 使用 Amazon EC2 主控台啟動 AL2023。
- 您必須使用 SSH 或 AWS Systems Manager連線至 Amazon EC2 執行個體。如需詳細資訊，請參閱連線至 AL2023 執行個體。
- 若要在 AL2023 上啟用 FIPS 模式，請遵循 中的指示在 AL2023 上啟用 FIPS 模式。

在 **`openssl-fips-provider-latest`**和 之間切換 **`openssl-fips-provider-certified`**

1. 使用 `dnf`切換 OpenSSL FIPS 供應商：

   ```
   sudo dnf -y swap openssl-fips-provider-latest openssl-fips-provider-certified
   ```

2. 檢查您是否使用已認證的 OpenSSL FIPS 供應商。在 FIPS 模式下使用 AL2023 時，執行下列命令：

   ```
   openssl list -providers
   ```

   您應該會看到下列輸出：

```
Providers:
  base
    name: OpenSSL Base Provider
    version: 3.2.2
    status: active
  default
    name: OpenSSL Default Provider
    version: 3.2.2
    status: active
  fips
    name: Amazon Linux 2023 - OpenSSL FIPS Provider
    version: 3.0.8-d694bfa693b76001
    status: active
```

AL2023 核心強化

AL2023 中的 6.1 Linux 核心已設定並建置數個強化選項和功能。

核心強化選項 (獨立於架構)

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_ACPI_CUSTOM_METHOD	n	n	N/A	N/A
CONFIG_BINFMT_MISC	m	m	m	m
CONFIG_BUG	y	y	y	y
CONFIG_BUG_ON_DATA_CORRUPTION	y	y	y	y

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_CFI_CLANG	N/A	N/A	N/A	N/A
CONFIG_CFI_PERMISSIVE	N/A	N/A	N/A	N/A
CONFIG_COMPAT	y	y	y	y
CONFIG_COMPAT_BRK	n	n	n	n
CONFIG_COMPAT_VDSO	N/A	n	N/A	n
CONFIG_DEBUG_CREDENTIALS	n	n	N/A	N/A
CONFIG_DEBUG_LIST	y	y	y	y
CONFIG_DEBUG_NOTIFIERS	n	n	n	n
CONFIG_DEBUG_SG	n	n	n	n
CONFIG_DEBUG_VIRTUAL	n	n	n	n
CONFIG_DEBUG_WX	n	n	n	n

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_DEFAULT_MMAP_MIN_ADDR	65536	65536	65536	65536
CONFIG_DEVKMEM	N/A	N/A	N/A	N/A
CONFIG_DEVMEM	n	n	n	n
CONFIG_EFI_DISABLE_PCI_DMA	n	n	n	n
CONFIG_FORTIFY_SOURCE	y	y	y	y
CONFIG_HARDENED_USERCOPY	y	y	y	y
CONFIG_HARDENED_USERCOPY_FALLBACK	N/A	N/A	N/A	N/A
CONFIG_HARDENED_USERCOPY_PAGESPAN	N/A	N/A	N/A	N/A
CONFIG_HIBERNATION	y	y	y	y

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_HW_RANDOM_TPM	N/A	N/A	N/A	N/A
CONFIG_INET_DIAG	m	m	m	m
CONFIG_INIT_ON_ALLOC_DEFAULT_ON	n	n	n	n
CONFIG_INIT_ON_FREE_DEFAULT_ON	n	n	n	n
CONFIG_INIT_STACK_ALL_ZERO	N/A	N/A	N/A	N/A
CONFIG_IOMMU_DEFAULT_DMA_STRICT	n	n	n	n
CONFIG_IOMMU_SUPPORT	y	y	y	y
CONFIG_IO_STRICT_DEVMEM	N/A	N/A	N/A	N/A

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_KEXEC	y	y	y	y
CONFIG_KFENCE	n	n	n	n
CONFIG_LDISC_AUTOLOAD	n	n	n	n
CONFIG_LEGACY_PTYS	n	n	n	n
CONFIG_LOCK_DOWN_KERNEL_FORCE_CONFIDENTIALITY	n	n	n	n
CONFIG_MODULES	y	y	y	y
CONFIG_MODULE_SIG	y	y	y	y
CONFIG_MODULE_SIG_ALL	y	y	y	y
CONFIG_MODULE_SIG_FORCE	n	n	n	n
CONFIG_MODULE_SIG_HASH	sha512	sha512	sha512	sha512

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_MODULE_SIG_KEY	certs/signing_key.pem	certs/signing_key.pem	certs/signing_key.pem	certs/signing_key.pem
CONFIG_MODULE_SIG_SHA512	y	y	y	y
CONFIG_PAGE_POISONING	n	n	n	n
CONFIG_PAGE_POISONING_NO_SANITY	N/A	N/A	N/A	N/A
CONFIG_PAGE_POISONING_ZERO	N/A	N/A	N/A	N/A
CONFIG_PANIC_ON_OOPS	y	y	y	y
CONFIG_PANIC_TIMEOUT	0	0	0	0
CONFIG_PROC_KCORE	y	y	y	y

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_RANDOMIZE_KSTACK_OFFSET_DEFAULT	n	n	n	n
CONFIG_RANDOM_TRUST_BOOTLOADER	y	y	N/A	N/A
CONFIG_RANDOM_TRUST_CPU	y	y	N/A	N/A
CONFIG_REFCOUNT_FULL	N/A	N/A	N/A	N/A
CONFIG_SCHED_CORE	N/A	y	N/A	y
CONFIG_SCHED_STACK_END_CHECK	y	y	y	y
CONFIG_SECCOMP	y	y	y	y
CONFIG_SECCOMP_FILTER	y	y	y	y
CONFIG_SECURITY	y	y	y	y

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_SECURITY_DMESG_RESTRICT	y	y	y	y
CONFIG_SECURITY_LANDLOCK	y	y	y	y
CONFIG_SECURITY_LOCKDOWN_LSM	y	y	y	y
CONFIG_SECURITY_LOCKDOWN_LSM_EARLY	y	y	y	y
CONFIG_SECURITY_SELINUX_BOOTPARAM	y	y	y	y
CONFIG_SECURITY_SELINUX_DEVELOP	y	y	y	y
CONFIG_SECURITY_SELINUX_DISABLE	n	n	N/A	N/A

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_SECURITY_WRITABLE_HOOKS	N/A	N/A	N/A	N/A
CONFIG_SECURITY_YAMA	y	y	y	y
CONFIG_SHUFFLE_PAGE_ALLOCATOR	y	y	y	y
CONFIG_SLAB_FREELIST_HARDENED	y	y	y	y
CONFIG_SLAB_FREELIST_RANDOM	y	y	y	y
CONFIG_SLUB_DEBUG	y	y	y	y
CONFIG_STACKPROTECTOR	y	y	y	y
CONFIG_STACKPROTECTOR_STRONG	y	y	y	y

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_STATIC_USERMODEHELPER	n	n	n	n
CONFIG_STRICT_DEVMEM	n	n	n	n
CONFIG_STRICT_KERNEL_RWX	y	y	y	y
CONFIG_STRICT_MODULE_RWX	y	y	y	y
CONFIG_SYN_COOKIES	y	y	y	y
CONFIG_VMAP_STACK	y	y	y	y
CONFIG_WERROR	n	n	n	n
CONFIG_ZERO_CALL_USED_REGS	n	n	n	n

允許在執行時間插入/取代 ACPI 方法 (CONFIG_ACPI_CUSTOM_METHOD)

Amazon Linux 會停用此選項，因為它允許根使用者寫入任意核心記憶體。

此選項是核心自我保護專案建議設定之一。

其他二進位格式 (**binfmt_misc**)

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。在 AL2023 中，此功能為選擇性，並建構作為核心模組。

BUG() 支援

此選項是核心自我保護專案建議設定之一。

BUG() 如果核心在檢查核心記憶體結構的有效性時發生資料損毀

Linux 核心的某些部分將檢查資料結構的內部一致性，並可在偵測到資料損毀時 BUG()。

此選項是核心自我保護專案建議設定之一。

COMPAT_BRK

停用此選項 (這是 Amazon Linux 設定核心的方式) 時，`randomize_va_space sysctl` 設定預設為 2，這也會以 `mmap` 基礎、堆疊和 VDSO 頁面隨機化為基礎啟用堆積隨機化。

此選項存在於核心中，以提供與 1996 及更舊版本的久遠 `libc.so.5` 二進位檔案的相容性。

此選項是核心自我保護專案建議設定之一。

COMPAT_VDSO

此組態選項與 `x86-64` 相關，但與 `aarch64` 無關。將此項設定為 n 時，Amazon Linux 核心不會讓 32 位元虛擬動態共用物件 (VDSO) 顯示在可預測的位址。已知可透過將此選項設為 n 來破壞的最新 `glibc` 自 2004 年開始設為 `glibc` 2.3.3。

此選項是核心自我保護專案建議設定之一。

CONFIG_DEBUG 門控強化

由 CONFIG_DEBUG 控制的 Linux 核心組態選項通常是設計用來在核心建置中進行問題除錯，因此效能等項目不是優先事項。AL2023 會啟用CONFIG_DEBUG_LIST強化選項。

設定 IOMMU 前，在 EFI 虛設常式中停用 PCI 裝置的 DMA

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。

在核心和使用者空間之間複製記憶體的強化

當核心需要將記憶體複製到使用者空間或從使用者空間複製記憶體時，此選項會啟用一些檢查，以防止某些類別的堆積溢位問題。

核心 4.16 到 5.15 中存在 CONFIG_HARDENED_USERCOPY_FALLBACK 選項，可協助核心開發人員透過 WARN( ) 找出任何遺失的 allowlist 項目。由於 AL2023 隨附 6.1 核心，此選項不再與 AL2023 相關。

核心中存在CONFIG_HARDENED_USERCOPY_PAGESPAN的選項主要做為開發人員的偵錯選項，不再適用於 AL2023 中的 6.1 核心。

此選項是核心自我保護專案建議設定之一。

休眠支援

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。必須啟用此選項，才能支援隨需執行個體休眠的能力，並支援休眠中斷 Spot 執行個體的功能

產生隨機數字

AL2023 核心已設定為確保 EC2 內有足夠的熵可供使用。

CONFIG_INET_DIAG

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。在 AL2023 中，此功能為選擇性，並建構作為核心模組。

在配置和解除配置時將所有核心頁面和 slab 分配器記憶體設為零

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。這些選項在 AL2023 中會停用，因為預設啟用此功能可能會對效能造成影響。CONFIG_INIT_ON_ALLOC_DEFAULT_ON 行為可透過將 init_on_alloc=1 加入核心命令行來啟用，並可透過加入 init_on_free=1 來啟用 CONFIG_INIT_ON_FREE_DEFAULT_ON 行為。

將所有堆疊變數初始化為零 (CONFIG_INIT_STACK_ALL_ZERO)

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。此選項需要 GCC 12 或更新版本，而 AL2023 則隨附 GCC 11。

核心模組簽署

AL2023 會簽署並驗證核心模組的簽章。為了保持與建置第三方模組之使用者的相容性，CONFIG_MODULE_SIG_FORCE 選項未啟用，因為這會要求模組具有有效的簽名。對於想要確保所有核心模組都已簽署的使用者， 鎖定 Linux 安全模組 (LSM) 可設為強制執行此操作。

kexec

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。此選項已啟用，以便使用 kdump 功能。

IOMMU 支援

AL2023 啟用 IOMMU 支援。預設情況下，不會啟用 CONFIG_IOMMU_DEFAULT_DMA_STRICT 選項，但可以透過將 iommu.passthrough=0 iommu.strict=1 加入核心命令行來設定此功能。

kfence

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。

舊版 pty 支援

AL2023 使用現代PTY界面 (devpts)。

此選項是核心自我保護專案建議設定之一。

鎖定 Linux 安全模組 (LSM)

AL2023 會建置 lockdown LSM，這會在使用安全開機時自動鎖定核心。

CONFIG_LOCK_DOWN_KERNEL_FORCE_CONFIDENTIALITY 選項未啟用。雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。未使用安全開機時，可以啟用鎖定 LSM 並視需要進行設定。

頁面中毒

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。與 類似 在配置和解除配置時將所有核心頁面和 slab 分配器記憶體設為零，這會在 AL2023 核心中停用，因為可能會影響效能。

堆疊保護器

AL2023 核心是使用 `-fstack-protector-strong`選項GCC啟用的堆疊保護器功能所建置。

此選項是核心自我保護專案建議設定之一。

seccomp BPF API

seccomp 強化功能會由 `systemd` 和容器執行期等的軟體使用，以加強使用者空間應用程式。

此選項是核心自我保護專案建議設定之一。

`panic()` 逾時

AL2023 核心已將此值設定為 `0`，這表示核心在驚慌之後不會重新啟動。雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。這可透過 `sysctl`、`/proc/sys/kernel/panic` 和在核心命令行上進行設定。

安全模型

AL2023 預設會以允許模式啟用 SELinux。如需詳細資訊，請參閱設定 AL2023 的 SELinux 模式。

鎖定 Linux 安全模組 (LSM) 和 `yama` 模組也會啟用。

`/proc/kcore`

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。

syscall 項目的核心堆疊位移隨機化

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。這可透過在核心命令行上設定 `randomize_kstack_offset=on` 來啟用。

參考計數檢查 (**`CONFIG_REFCOUNT_FULL`**)

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。由於此選項可能會對效能造成影響，因此無法正確啟用。

SMT 核心的排程器感知 (**`CONFIG_SCHED_CORE`**)

AL2023 核心是使用 建置`CONFIG_SCHED_CORE`，可讓使用者空間應用程式使用 `prctl(PR_SCHED_CORE)`。此選項是核心自我保護專案建議設定之一。

檢查呼叫 **schedule()** (**CONFIG_SCHED_STACK_END_CHECK**) 時的堆疊損毀

AL2023 核心的建置CONFIG_SCHED_STACK_END_CHECK已啟用。此選項是核心自我保護專案建議設定之一。

記憶體分配器強化

AL2023 核心可使用 CONFIG_SHUFFLE_PAGE_ALLOCATOR、CONFIG_SLAB_FREELIST_HARDENED和 CONFIG_SLAB_FREELIST_RANDOM選項來強化核心記憶體分配器。此選項是核心自我保護專案建議設定之一。

SLUB 除錯支援

AL2023 核心會啟用 CONFIG_SLUB_DEBUG，因為此選項會為可在核心命令列上啟用的分配器啟用選用偵錯功能。此選項是核心自我保護專案建議設定之一。

CONFIG_STATIC_USERMODEHELPER

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。這是因為 CONFIG_STATIC_USERMODEHELPER 需要發行版的特殊支援，且目前不存在 Amazon Linux 中。

唯讀核心文字和 rodata (**CONFIG_STRICT_KERNEL_RWX** 和 **CONFIG_STRICT_MODULE_RWX**)

AL2023 核心設定為將核心和核心模組文字和rodata記憶體標記為唯讀，並將非文字記憶體標記為無法執行。此選項是核心自我保護專案建議設定之一。

TCP syncookie 支援 (**CONFIG_SYN_COOKIES**)

AL2023 核心建置時支援 TCP syncookies。此選項是核心自我保護專案建議設定之一。

有保護頁面的虛擬映射堆疊 (**CONFIG_VMAP_STACK**)

AL2023 核心是使用 建置CONFIG_VMAP_STACK，啟用具有保護頁面的虛擬映射核心堆疊。此選項是核心自我保護專案建議設定之一。

使用編譯器警告作為錯誤建置 (**CONFIG_WERROR**)

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。

在函數退出登錄歸零 (CONFIG_ZERO_CALL_USED_REGS)

雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。

使用者空間分配的最小地址

此強化選項有助於減輕核心 NULL 指標錯誤的影響。此選項是核心自我保護專案建議設定之一。

clang 特定強化選項

AL2023 核心是使用 GCC 而非 建置clang，因此無法啟用CONFIG_CFI_CLANG強化選項，這也使 CONFIG_CFI_PERMISSIVE 不適用。雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。

x86-64 特定核心強化選項

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_AMD_IOMMU	N/A	y	N/A	y
CONFIG_AMD_IOMMU_V2	N/A	y	N/A	N/A
CONFIG_IA32_EMULATION	N/A	y	N/A	y
CONFIG_INTEL_IOMMU	N/A	y	N/A	y
CONFIG_INTEL_IOMMU_DEFAULT_ON	N/A	n	N/A	n

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_INTEL_IOMMU_SVM	N/A	n	N/A	n
CONFIG_LEGACY_VSYSCALL_NONE	N/A	n	N/A	n
CONFIG_MODIFY_LDT_SYSCALL	N/A	n	N/A	n
CONFIG_PAGE_TABLE_ISOLATION	N/A	y	N/A	N/A
CONFIG_RANDOMIZE_MEMORY	N/A	y	N/A	y
CONFIG_X86_64	N/A	y	N/A	y
CONFIG_X86_MSR	N/A	y	N/A	y
CONFIG_X86_VSYSCALL_EMULATION	N/A	y	N/A	y
CONFIG_X86_X32	N/A	N/A	N/A	N/A
CONFIG_X86_X32_ABI	N/A	n	N/A	n

x86-64 支援

基本 x86-64 支援包括實體位址延伸 (PAE) 和無執行 (NX) 位元支援。此選項是核心自我保護專案建議設定之一。

AMD 和 Intel IOMMU 支援

AL2023 核心建置支援 AMD 和 Intel IOMMUs。此選項是核心自我保護專案建議設定之一。

未設定 CONFIG_INTEL_IOMMU_DEFAULT_ON 選項，但可透過將 intel_iommu=on 傳遞至核心命令行來啟用。雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。

目前未在 AL2023 中啟用 CONFIG_INTEL_IOMMU_SVM選項。雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。

支援 32 位元使用者空間

> **Important**
>
> 對 32 位元 x86 使用者空間的支援已棄用，在未來的 Amazon Linux 主要版本中，可能會移除對執行 32 位元使用者空間二進位檔案的支援。

> **Note**
>
> 雖然 AL2023 不再包含任何 32 位元套件，但核心仍會支援執行 32 位元使用者空間。如需詳細資訊，請參閱32 位元 x86 (i686) 套件。

為了支援執行 32 位元使用者空間應用程式，AL2023 不會啟用 CONFIG_X86_VSYSCALL_EMULATION選項，並啟用 CONFIG_IA32_EMULATION、CONFIG_COMPAT和 CONFIG_X86_VSYSCALL_EMULATION選項。雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。

未啟用 64 位元處理器的 x32 原生 32 位元 ABI (CONFIG_X86_X32 和 CONFIG_X86_X32_ABI)。此選項是核心自我保護專案建議設定之一。

x86 模型特定暫存器 (MSR) 支援

CONFIG_X86_MSR 選項已啟用以支援 turbostat。雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。

modify_ldt syscall

AL2023 不允許使用者程式使用 modify_ldt syscall 修改 x86 Local Descriptor Table (LDT)。執行 16 位元或分段程式碼需要此呼叫，缺少此呼叫可能會破壞 dosemu 等的軟體、在 WINE 下執行某些程式，以及非常舊的執行緒庫。此選項是核心自我保護專案建議設定之一。

在使用者模式下移除核心對應

AL2023 會設定核心，讓大部分的核心地址不會映射到使用者空間。此選項是核心自我保護專案建議設定之一。

隨機化核心記憶體區段

AL2023 會設定核心，以隨機化核心記憶體區段的基本虛擬地址。此選項是核心自我保護專案建議設定之一。

aarch64 特定核心強化選項

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_ARM64_BTI	y	N/A	y	N/A
CONFIG_ARM64_BTI_KERNEL	N/A	N/A	N/A	N/A
CONFIG_ARM64_PTR_AUTH	y	N/A	y	N/A
CONFIG_ARM64_PTR_AUTH_KERNEL	y	N/A	y	N/A

CONFIG 選項	AL2023/6.1/aarch64	AL2023/6.1/x86_64	AL2023/6.12/aarch64	AL2023/6.12/x86_64
CONFIG_ARM64_SW_TTBR0_PAN	y	N/A	y	N/A
CONFIG_UNMAP_KERNEL_AT_EL0	y	N/A	y	N/A

分支目標識別

AL2023 核心支援分支目標識別 (CONFIG_ARM64_BTI)。此選項是核心自我保護專案建議設定之一。

CONFIG_ARM64_BTI_KERNEL 選項在 AL2023 中未啟用，因為它是使用 GCC 建置，並且由於 gcc 錯誤，上游核心目前已停用對使用此選項建置核心的支援。雖然此選項是核心自我保護專案 (KSPP) 建議設定之一，但 AL2023 並未將此組態選項設為 KSPP 建議的選項。

指標驗證 (**CONFIG_ARM64_PTR_AUTH**)

AL2023 核心建置時支援指標身分驗證延伸模組 (ARMv8.3 延伸模組的一部分），可用於協助緩解傳回導向程式設計 (ROP) 技術。Graviton 3 已推出對 Graviton 進行指標驗證所需的硬體支援。

CONFIG_ARM64_PTR_AUTH 選項已啟用，並為使用者空間提供指標驗證的支援。由於 CONFIG_ARM64_PTR_AUTH_KERNEL選項也已啟用，AL2023 核心能夠自行使用傳回地址保護。

此選項是核心自我保護專案建議設定之一。

模擬特權存取永不使用 **TTBR0_EL1** 切換

此選項可防止核心直接存取使用者空間記憶體，只會由使用者存取常式將 TTBR0_EL1 暫時設定為有效值。

此選項是核心自我保護專案建議設定之一。

在使用者空間中執行時取消對應核心

AL2023 核心設定為在使用者空間 () 中執行時取消核心映射CONFIG_UNMAP_KERNEL_AT_EL0。此選項是核心自我保護專案建議設定之一。

AL2023 上的 UEFI 安全開機

AL2023 支援從 2023.1 版開始的 UEFI 安全開機。您必須將 AL2023 與同時支援 UEFI 和 UEFI 安全開機的 Amazon EC2 執行個體搭配使用。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的在 UEFI 開機模式下啟動 Amazon EC2 執行個體的要求。 Amazon EC2

啟用 UEFI 安全開機的 AL2023 執行個體僅接受由 簽署的核心層級程式碼，包括 Linux 核心和模組，Amazon因此您可以確保執行個體僅執行由 簽署的核心層級程式碼 AWS。

如需 Amazon EC2 執行個體和 UEFI 安全開機的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 執行個體的 UEFI 安全開機。 Amazon EC2

先決條件

- 您必須使用 AL2023 版本 2023.1 或更新版本的 AMI。
- 執行個體類型必須支援 UEFI 安全開機。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的在 UEFI 開機模式下啟動 Amazon EC2 執行個體的要求。 Amazon EC2

在 AL2023 上啟用 UEFI 安全開機

標準 AL2023 AMI 包含開機載入器和由我方金鑰簽章的核心。您可以透過註冊現有執行個體或透過預先啟用 UEFI 安全開機建立 AMI，方法是從快照註冊映像來啟用 UEFI 安全開機。標準 AL2023 AMI 預設不會啟用 UEFI 安全開機。

AL2023 AMI 的開機模式已設定為 `uefi-preferred`，如果執行個體類型支援 UEFI，則可確保透過這些 AMI 啟動的執行個體將使用 UEFI 韌體。如果執行個體類型不支援 UEFI，則會在舊版 BIOS 上啟動執行個體。執行個體在舊版 BIOS 模式中啟動時，不會強制執行 UEFI 安全開機。

如需 Amazon EC2 執行個體上 AMI 開機模式的詳細資訊，請參閱《Amazon EC2 使用者指南》中的使用 Amazon EC2 開機模式的執行個體啟動行為。 Amazon EC2

主題

- 註冊現有執行個體
- 從快照註冊映像
- 撤銷更新
- UEFI 安全開機在 AL2023 上的運作方式
- 註冊自有金鑰

註冊現有執行個體

若要註冊現有的執行個體，請在特定 UEFI 韌體變數中填入一組金鑰，讓韌體能夠驗證開機載入器，以便開機載入器在下次開機時驗證核心。

1. Amazon Linux 提供簡化註冊過程的工具。執行下列命令以佈建執行個體，並使用必要的金鑰和憑證集合來佈建執行個體。

```
sudo amazon-linux-sb enroll
```

2. 執行下列命令重新開機 執行個體。執行個體重新開機後，將會啟用 UEFI 安全開機。

```
sudo reboot
```

> **Note**
>
> Amazon Linux AMI 目前不支援 Nitro 信任平台模組 (NitroTPM)。如果除了 UEFI 安全開機之外，您還需要 NitroTPM，請使用下一節中的資訊。

從快照註冊映像

使用 Amazon EC2 `register-image` API 並從 Amazon EBS 根磁碟區的快照註冊 AMI 時，您可以使用包含 UEFI 變數存放區狀態的二進位 Blob 佈建 AMI。透過提供 AL2023 `UefiData`，即可啟用 UEFI 安全開機，而且不需要遵循上一節中的步驟。

如需建立和使用二進位 Blob 的詳細資訊，請參閱《Amazon EC2 使用者指南》中的建立包含預先填入變數存放區的二進位 Blob。

AL2023 提供可直接在 Amazon EC2 執行個體上使用且預先建置的二進位 Blob。二進位 Blob 位於執行中執行個體的 `/usr/share/amazon-linux-sb-keys/uefi.vars`。此 Blob 是由 `amazon-linux-sb-keys` RPM 套件提供，此套件從 2023.1 版開始預設安裝在 AL2023 AMI 上。

> **Note**
>
> 若要確保您使用的是最新版本的金鑰和撤銷，請使用您用來建立 AMI 的相同 AL2023 版本中的 Blob。

註冊映像時，建議使用設定為 uefi 的 RegisterImageAPI BootMode 參數。這可讓您透過將 TpmSupport 參數設定為 v2.0 來啟用 NitroTPM。此外，將 BootMode 設定為 uefi 可確保啟用 UEFI 安全開機，且在切換至不支援 UEFI 的執行個體類型時，無法意外停用。

如需 NitroTPM 的詳細資訊，請參閱《Amazon EC2 使用者指南》中的適用於 Amazon EC2 執行個體的 NitroTPM。 Amazon EC2

撤銷更新

Amazon Linux 可能需要發行新版本的開機載入器 grub2，或使用更新金鑰簽署的 Linux 核心。在這種情況下，可能需要撤銷舊金鑰，以防止有人藉由舊版開機載入器的可利用錯誤繞過 UEFI 安全開機驗證程序。

grub2 或 kernel 套件的套件更新一律會自動將撤銷清單更新至執行中執行個體的 UEFI 變數存放區。這表示啟用 UEFI 安全開機並安裝套件的安全更新後，您將無法再執行舊版的套件。

UEFI 安全開機在 AL2023 上的運作方式

與其他 Linux 發行版不同，Amazon Linux 不提供稱為 Shim 的額外元件來作為第一階段的開機載入器。Shim 通常使用 Microsoft 金鑰簽署。例如，在具有 Shim 的 Linux 發行版上，Shim 會載入 grub2 開機載入器並 Shim 自己的程式碼來驗證 Linux 核心。此外，Shim 會在位於 UEFI 變數存放區的機器擁有者金鑰 (MOK) 資料庫中維護自有金鑰和撤銷組合，並透過 mokutil 工具控制。

Amazon Linux 不提供 Shim。由於 AMI 擁有者控制 UEFI 變數，因此不需要此中繼步驟，而且會對啟動和開機時間產生不利影響。此外，我們選擇預設不包括對任何供應商金鑰的信任，以減少不需要的二進位檔案可以執行的機會。與往常一樣，如果客戶選擇這樣做，則可以包含二進位檔案。

使用 Amazon Linux 時，UEFI 可以直接載入並驗證 grub2 開機載入器。grub2 開機載入器已修改為在載入 Linux 核心後使用 UEFI 對其進行驗證。因此，系統會使用儲存在一般 UEFI db 變數 (授權金鑰資料庫) 中的相同憑證來驗證 Linux 核心，並針對與開機載入器和其他 UEFI 二進位檔案相同的 dbx 變數 (撤銷資料庫) 進行測試。由於我們提供自有 PK 和 KEK 金鑰，用於控制對 db 資料庫和 dbx 資料庫的存取，因此我們可以根據需要發行已簽章的更新和撤銷，而無需中介 (如 Shim)。

如需 UEFI 安全開機的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 UEFI 安全開機如何與 Amazon EC2 執行個體搭配使用。 Amazon EC2

註冊自有金鑰

如上一節所述，Amazon Linux 不需要在 Amazon EC2 上使用 shim 以進行 UEFI 安全開機。當您正在閱讀其他 Linux 發行版的文件時，您可能會發現使用 mokutil 管理機器擁有者金鑰 (MOK) 資料庫的

說明文件，且這些資料庫不在 AL2023 中。shim 和 MOK 環境解決在 UEFI 韌體中註冊金鑰的特定限制，這些限制不適用於 Amazon EC2 實作 UEFI 安全開機的方式。使用 Amazon EC2 時，有一些機制可以輕鬆地直接操作 UEFI 變數存放區中的金鑰。

如果您想要註冊自己的金鑰，您可以透過在現有執行個體內操作變數存放區（請參閱從執行個體內將金鑰新增至變數存放區）或建構預先填入的二進位 Blob（請參閱建立包含預先填入變數存放區的二進位 Blob) 來執行此操作。

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。