aws

使用者指南

Amazon Linux 2

Amazon Linux 2: 使用者指南

Table of Contents

什麼是 Amazon Linux 2？

Amazon Linux 2 (AL2) 是來自 Amazon Web Services () 的 Linux 作業系統AWS。AL2 旨在為在 Amazon EC2 上執行的應用程式提供穩定、安全且高效能的環境。它還包含可實現與 有效整合的套件AWS，包括啟動組態工具和許多熱門AWS程式庫和工具。 為所有執行 AL2 的執行個體AWS提供持續的安全和維護更新。在 CentOS 上開發的許多應用程式，以及類似的分佈，在 AL2 上執行。AL2 是免費提供的。

> **Note**
>
> AL2 不再是 Amazon Linux 的目前版本。AL2023 是 AL2 的後續版本。如需詳細資訊，請參閱《AL2023 使用者指南》中的比較 AL2 和 AL2023 中的套件變更AL2023清單。

> **Note**
>
> AL2 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。

Amazon Linux 可用性

AWS提供 AL2023, AL2和 Amazon Linux 1 (AL1，先前為 Amazon Linux AMI)。如果要從其他的 Linux 分佈遷移至 Amazon Linux，建議遷移至 AL2023。

> **Note**
>
> AL1 的標準支援已於 2020 年 12 月 31 日結束。AL1 維護支援階段已於 2023 年 12 月 31 日結束。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。

如需 Amazon Linux 的詳細資訊，請參閱 AL2023、AL2 和 AL1。

如需 Amazon Linux 容器映像，請參閱《Amazon Elastic Container Registry 使用者指南》的 Amazon Linux 容器映像。

AL2 中的已棄用功能

下列各節說明 AL2 中支援的功能，但 AL2023 中不存在。這是功能，例如 AL2 中存在但 AL2023 中沒有的功能和套件，而且不會新增至 AL2023。如需 AL2 支援此功能的時間長度，請參閱 AL2 文件。

`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-`套件都會終止，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 中停用的已棄用功能，在 AL2 中移除

本節說明 AL1 中可用的功能，而 AL2 中不再提供此功能。

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

主題

- 32 位元 x86 (i686) AMIs
- aws-apitools-* 取代為AWS CLI
- systemd 在 AL2 upstart中取代

32 位元 x86 (i686) AMIs

在 2014.09 版的 AL1 中，Amazon Linux 宣布將是產生 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 主機上提

供 32 位元二進位檔的有限執行時間支援，並且不提供開發套件來建立新的 32 位元二進位檔。AL2023 不再包含任何 32 位元使用者空間套件。我們建議使用者先完成轉換為 64 位元程式碼，再遷移至 AL2023。

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提供了一組在 中實作的命令列公用程式Java，允許使用者進行 Amazon EC2 API 呼叫。這些工具已於 2015 年終止，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

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 仍繼續提供一些這些命令列公用程式，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，因為它正在積極維護，並提供使用 AWSAPIs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 也提供在 AL1 中已棄用且從 AL2 之後不在 Amazon Linux 中的 aws-apitools-iam和 aws-apitools-rds套件。

systemd 在 AL2 **upstart**中取代

AL2 是第一個使用 systemd init 系統的 Amazon Linux 版本，upstart取代 AL1。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 Amazon Linux 較新主要版本的一部分，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

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 為 x86-64 主機上的 32 位元二進位檔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。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，因為它正在積極維護，並提供使用 AWS APIs的方法。

aws-apitools-* 套件已於 2017 年 3 月棄用，不會再收到進一步更新。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 為了支援統一的 Amazon CloudWatch Logs 代理程式而棄用

awslogs 套件已在 AL2 中棄用，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到可緩解 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 （啟用或停用）。

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。`mcrypt`PHP

上游`mcrypt`程式庫上次在 2007 年發行了版本，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，移除了對具有維護器之專案的提及。

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 提供多種版本的 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 Java 在 AL2 中。

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 Python 在 AL2 中。

`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下拉式選單。

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`做為遵循的網路組態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 Sun RPC中的 實作`glibc`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 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

多年來，舊版ftp套件尚未在上游主動維護。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漏洞攜帶修補程式，但程式碼庫在很大程度上保持不變。

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)

複雜的檔案傳輸程式，支援 FTP、HTTP、SFTP 和其他通訊協定。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 傳輸資料的多功能命令列工具，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，支援 HTTP、HTTPS 和 FTP 通訊協定。

使用 安裝： dnf install wget（並非所有 AL2023 映像都預設安裝）

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。

預設為 OpenSSH 套件的一部分。

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。

準備遷移至 AL2023

您可以準備移至 AL2023，同時繼續使用 AL2。

主題

檢閱 AL2023 中的變更清單

AL2023 文件包含自 AL2 以來已實作變更的詳細清單。此資訊位於比較 AL2 和 AL2023 區段中。在 AL2023 的套件變更區段中，也有完整的軟體套件變更清單。

AL2023 不包含 `amazon-linux-extras`。反之，它會提供提供多個版本的命名空間套件。由於在 AL2023 中更新了許多套件，AL2023 中的基本版本可能會晚於您從 取得的版本`amazon-linux-extras`。

> **Note**
>
> 我們建議您不要執行 `amazon-linux-extras`，因為它是 EOL。

檢閱文件中的這些區段後，您可以判斷 AL2023 中是否有變更，可能需要您調整環境以進行遷移。例如，您可能需要最終將 Python 2.7 指令碼遷移至 Python 3。

從`cron`任務遷移至`systemd`計時器

根據預設， `cron` 不會安裝在 AL2023 中。您可以將`cron`任務遷移至 AL2 中的`systemd`計時器，以準備遷移至 AL2023。 `systemd` 有許多優點，例如更精確地控制計時器的執行時間和改善記錄。

AL2 限制

下列主題涵蓋 AL2 的各種限制，以及它們是否已在較新版本的 Amazon Linux 中解決。

主題

yum 無法驗證使用 GPG 子金鑰建立的 GPG 簽章

在rpm新增支援驗證使用 GPG 子金鑰建立的rpm套件簽章之前，AL2 中的套件管理員版本來自 。如果您要建立與 AL2 相容的套件，則需要確保使用與rpm屬於 AL2 的 相容的 GPG 簽署金鑰

為了確保現有使用者的回溯相容性，AL2 rpm中的 版本只會收到安全後端連接埠。

AL2023 rpm中的 版本包含支援驗證使用 GPG 子金鑰建立的套件簽章。

比較 AL1 和 AL2

下列主題說明 AL1 和 AL2 之間的主要差異。它們也包含生命週期和支援，以及套件變更的相關資訊。

主題

AL1 支援和 EOL

AL1 現在是 EOL。AL1 自 2020 年 12 月 31 日起結束標準支援，並處於維護支援階段，直到 2023 年 12 月 31 日為止。

我們建議您升級至最新的 Amazon Linux 版本。

支援 AWSGraviton 處理器

AL2 推出對 Graviton 處理器的支援。AL2023 已針對 Graviton 處理器進一步最佳化。

systemd 取代 **upstart** 並成為 **init** 系統

在 AL2 中，`systemd`取代 `upstart`做為`init`系統。

Python 2.6 和 2.7 已取代為 Python 3

雖然 AL1 將 Python 2.6 標記為具有 2018.03 版本的 EOL，但套件仍在要安裝的儲存庫中。AL2 隨附 Python 2.7 作為最早支援的 Python 版本。

AL2023 完成轉換為 Python 3，而且儲存庫中不包含 Python 2.x 版本。

比較安裝在 AL1 和 AL2 AMIs 上的套件

套件	AL1 AMI	AL2 AMI
GeoIP		1.5.0
PyYAML		3.10
acl	2.2.49	2.2.51
acpid	2.0.19	2.0.19
alsa-lib	1.0.22	
amazon-linux-extras		2.0.3
amazon-linux-extras-yum-plugin		2.0.3
amazon-ssm-agent	3.2.1705.0	3.2.1705.0
at	3.1.10	3.1.13
attr	2.4.46	2.4.46
audit	2.6.5	2.8.1
audit-libs	2.6.5	2.8.1
authconfig	6.2.8	6.2.8
aws-amitools-ec2	1.5.13	
aws-cfn-bootstrap	1.4	2.0
aws-cli	1.18.107	
awscli		1.18.147
basesystem	10.0	10.0
bash	4.2.46	4.2.46
bash-completion		2.1

套件	AL1 AMI	AL2 AMI
bc	1.06.95	1.06.95
bind-export-libs		9.11.4
bind-libs	9.8.2	9.11.4
bind-libs-lite		9.11.4
bind-license		9.11.4
bind-utils	9.8.2	9.11.4
binutils	2.27	2.29.1
blktrace		1.0.5
boost-date-time		1.53.0
boost-system		1.53.0
boost-thread		1.53.0
bridge-utils		1.5
bzip2	1.0.6	1.0.6
bzip2-libs	1.0.6	1.0.6
ca-certificates	2023.2.62	2023.2.62
checkpolicy	2.1.10	
chkconfig	1.3.49.3	1.7.4
chrony		4.2
cloud-disk-utils	0.27	
cloud-init	0.7.6	19.3

套件	AL1 AMI	AL2 AMI
cloud-utils-growpart		0.31
copy-jdk-configs	3.3	
coreutils	8.22	8.22
cpio	2.10	2.12
cracklib	2.8.16	2.9.0
cracklib-dicts	2.8.16	2.9.0
cronie	1.4.4	1.4.11
cronie-anacron	1.4.4	1.4.11
crontabs	1.10	1.11
cryptsetup	1.6.7	1.7.4
cryptsetup-libs	1.6.7	1.7.4
curl	7.61.1	8.3.0
cyrus-sasl	2.1.23	
cyrus-sasl-lib	2.1.23	2.1.26
cyrus-sasl-plain	2.1.23	2.1.26
dash	0.5.5.1	
db4	4.7.25	
db4-utils	4.7.25	
dbus	1.6.12	1.10.24
dbus-libs	1.6.12	1.10.24

套件	AL1 AMI	AL2 AMI
dejavu-fonts-common	2.33	
dejavu-sans-fonts	2.33	
dejavu-serif-fonts	2.33	
device-mapper	1.02.135	1.02.170
device-mapper-event	1.02.135	1.02.170
device-mapper-event-libs	1.02.135	1.02.170
device-mapper-libs	1.02.135	1.02.170
device-mapper-persistent-data	0.6.3	0.7.3
dhclient	4.1.1	4.2.5
dhcp-common	4.1.1	4.2.5
dhcp-libs		4.2.5
diffutils	3.3	3.3
dmidecode		3.2
dmraid	1.0.0.rc16	1.0.0.rc16
dmraid-events	1.0.0.rc16	1.0.0.rc16
dosfstools		3.0.20
dracut	004	033
dracut-config-ec2		2.0
dracut-config-generic		033
dracut-modules-growroot	0.20	

套件	AL1 AMI	AL2 AMI
dump	0.4	
dyninst		9.3.1
e2fsprogs	1.43.5	1.42.9
e2fsprogs-libs	1.43.5	1.42.9
ec2-hibinit-agent	1.0.0	1.0.2
ec2-instance-connect		1.1
ec2-instance-connect-selinux		1.1
ec2-net-utils	0.7	1.7.3
ec2-utils	0.7	1.2
ed	1.1	1.9
elfutils-default-yama-scope		0.176
elfutils-libelf	0.168	0.176
elfutils-libs		0.176
epel-release	6	
ethtool	3.15	4.8
expat	2.1.0	2.1.0
file	5.37	5.11
file-libs	5.37	5.11
filesystem	2.4.30	3.2
findutils	4.4.2	4.5.11

套件	AL1 AMI	AL2 AMI
fipscheck	1.3.1	1.4.1
fipscheck-lib	1.3.1	1.4.1
fontconfig	2.8.0	
fontpackages-filesystem	1.41	
freetype	2.3.11	2.8
fuse-libs	2.9.4	2.9.2
gawk	3.1.7	4.0.2
gdbm	1.8.0	1.13
gdisk	0.8.10	0.8.10
generic-logos	17.0.0	18.0.0
get_reference_source	1.2	
gettext		0.19.8.1
gettext-libs		0.19.8.1
giflib	4.1.6	
glib2	2.36.3	2.56.1
glibc	2.17	2.26
glibc-all-langpacks		2.26
glibc-common	2.17	2.26
glibc-locale-source		2.26
glibc-minimal-langpack		2.26

套件	AL1 AMI	AL2 AMI
gmp	6.0.0	6.0.0
gnupg2	2.0.28	2.0.22
gpgme	1.4.3	1.3.2
gpm-libs	1.20.6	1.20.7
grep	2.20	2.20
groff	1.22.2	
groff-base	1.22.2	1.22.2
grub	0.97	
grub2		2.06
grub2-common		2.06
grub2-efi-x64-ec2		2.06
grub2-pc		2.06
grub2-pc-modules		2.06
grub2-tools		2.06
grub2-tools-minimal		2.06
grubby	7.0.15	8.28
gssproxy		0.7.0
gzip	1.5	1.5
hardlink		1.3
hesiod	3.1.0	

套件	AL1 AMI	AL2 AMI
hibagent	1.0.0	1.1.0
hmaccalc	0.9.12	
hostname		3.13
hunspell		1.3.2
hunspell-en		0.20121024
hunspell-en-GB		0.20121024
hunspell-en-US		0.20121024
hwdata	0.233	0.252
info	5.1	5.1
initscripts	9.03.58	9.49.47
iproute	4.4.0	5.10.0
iptables	1.4.21	1.8.4
iptables-libs		1.8.4
iputils	20121221	20180629
irqbalance	1.5.0	1.7.0
jansson		2.10
java-1.7.0-openjdk	1.7.0.321	
javapackages-tools	0.9.1	
jbigkit-libs		2.0
jpackage-utils	1.7.5	

套件	AL1 AMI	AL2 AMI
json-c		0.11
kbd	1.15	1.15.5
kbd-legacy		1.15.5
kbd-misc	1.15	1.15.5
kernel	4.14.326	5.10.199
kernel-tools	4.14.326	5.10.199
keyutils	1.5.8	1.5.8
keyutils-libs	1.5.8	1.5.8
kmod	14	25
kmod-libs	14	25
kpartx	0.4.9	0.4.9
kpatch-runtime		0.9.4
krb5-libs	1.15.1	1.15.1
langtable		0.0.31
langtable-data		0.0.31
langtable-python		0.0.31
lcms2	2.6	
less	436	458
libICE	1.0.6	
libSM	1.2.1	

套件	AL1 AMI	AL2 AMI
libX11	1.6.0	
libX11-common	1.6.0	
libXau	1.0.6	
libXcomposite	0.4.3	
libXext	1.3.2	
libXfont	1.4.5	
libXi	1.7.2	
libXrender	0.9.8	
libXtst	1.2.2	
libacl	2.2.49	2.2.51
libaio	0.3.109	0.3.109
libassuan	2.0.3	2.1.0
libattr	2.4.46	2.4.46
libbasicobjects		0.1.1
libblkid	2.23.2	2.30.2
libcap	2.16	2.54
libcap-ng	0.7.5	0.7.5
libcap54	2.54	
libcgroup	0.40.rc1	
libcollection		0.7.0

套件	AL1 AMI	AL2 AMI
libcom_err	1.43.5	1.42.9
libconfig		149
libcroco		0.6.12
libcrypt		2.26
libcurl	7.61.1	8.3.0
libdaemon		0.14
libdb		5.3.21
libdb-utils		5.3.21
libdrm		2.4.97
libdwarf		20130207
libedit	2.11	3.0
libestr		0.1.9
libevent	2.0.21	2.0.21
libfastjson		0.99.4
libfdisk		2.30.2
libffi	3.0.13	3.0.13
libfontenc	1.0.5	
libgcc		7.3.1
libgcc72	7.2.1	
libgcrypt	1.5.3	1.5.3

套件	AL1 AMI	AL2 AMI
libgomp		7.3.1
libgpg-error	1.11	1.12
libgssglue	0.1	
libicu	50.2	50.2
libidn	1.18	1.28
libidn2	2.3.0	2.3.0
libini_config		1.3.1
libjpeg-turbo	1.2.90	2.0.90
libmetalink		0.1.3
libmnl	1.0.3	1.0.3
libmount	2.23.2	2.30.2
libnetfilter_conntrack	1.0.4	1.0.6
libnfnetlink	1.0.1	1.0.1
libnfsidmap	0.25	0.25
libnghttp2	1.33.0	1.41.0
libnih	1.0.1	
libnl	1.1.4	
libnl3		3.2.28
libnl3-cli		3.2.28
libpath_utils		0.2.1

套件	AL1 AMI	AL2 AMI
libpcap		1.5.3
libpciaccess		0.14
libpipeline	1.2.3	1.2.3
libpng	1.2.49	1.5.13
libpsl	0.6.2	
libpwquality	1.2.3	1.2.3
libref_array		0.1.5
libseccomp		2.4.1
libselinux	2.1.10	2.5
libselinux-utils	2.1.10	2.5
libsemanage	2.1.6	2.5
libsepol	2.1.7	2.5
libsmartcols	2.23.2	2.30.2
libss	1.43.5	1.42.9
libssh2	1.4.2	1.4.3
libsss_idmap		1.16.5
libsss_nss_idmap		1.16.5
libstdc++		7.3.1
libstdc++72	7.2.1	
libstoragemgmt		1.6.1

套件	AL1 AMI	AL2 AMI
libstoragemgmt-python		1.6.1
libstoragemgmt-python-clibs		1.6.1
libsysfs	2.1.0	2.1.0
libtasn1	2.3	4.10
libteam		1.27
libtiff		4.0.3
libtirpc	0.2.4	0.2.4
libudev	173	
libunistring	0.9.3	0.9.3
libuser	0.60	0.60
libutempter	1.1.5	1.1.6
libuuid	2.23.2	2.30.2
libverto	0.2.5	0.2.5
libverto-libevent		0.2.5
libwebp		0.3.0
libxcb	1.11	
libxml2	2.9.1	2.9.1
libxml2-python		2.9.1
libxml2-python27	2.9.1	
libxslt	1.1.28	

套件	AL1 AMI	AL2 AMI
libyaml	0.1.6	0.1.4
lm_sensors-libs		3.4.0
log4j-cve-2021-44228-hotpatch	1.3	
logrotate	3.7.8	3.8.6
lsof	4.82	4.87
lua	5.1.4	5.1.4
lvm2	2.02.166	2.02.187
lvm2-libs	2.02.166	2.02.187
lz4		1.7.5
mailcap	2.1.31	
make	3.82	3.82
man-db	2.6.3	2.6.3
man-pages	4.10	3.53
man-pages-overrides		7.5.2
mariadb-libs		5.5.68
mdadm	3.2.6	4.0
microcode_ctl	2.1	2.1
mingetty	1.08	
mlocate		0.26
mtr		0.92

套件	AL1 AMI	AL2 AMI
nano	2.5.3	2.9.8
nc	1.84	
ncurses	5.7	6.0
ncurses-base	5.7	6.0
ncurses-libs	5.7	6.0
net-tools	1.60	2.0
nettle		2.7.1
newt	0.52.11	0.52.15
newt-python		0.52.15
newt-python27	0.52.11	
nfs-utils	1.3.0	1.3.0
nspr	4.25.0	4.35.0
nss	3.53.1	3.90.0
nss-pem	1.0.3	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	3.90.0
nss-util	3.53.1	3.90.0
ntp	4.2.8p15	

套件	AL1 AMI	AL2 AMI
ntpdate	4.2.8p15	
ntsysv	1.3.49.3	1.7.4
numactl	2.0.7	
numactl-libs		2.0.9
openldap	2.4.40	2.4.44
openssh	7.4p1	7.4p1
openssh-clients	7.4p1	7.4p1
openssh-server	7.4p1	7.4p1
OpenSSL	1.0.2k	1.0.2k
openssl-libs		1.0.2k
os-prober		1.58
p11-kit	0.18.5	0.23.22
p11-kit-trust	0.18.5	0.23.22
pam	1.1.8	1.1.8
pam_ccreds	10	
pam_krb5	2.3.11	
pam_passwdqc	1.0.5	
parted	2.1	3.1
passwd	0.79	0.79
pciutils	3.1.10	3.5.1

套件	AL1 AMI	AL2 AMI
pciutils-libs	3.1.10	3.5.1
pcre	8.21	8.32
pcre2		10.23
perl	5.16.3	5.16.3
perl-Carp	1.26	1.26
perl-Digest	1.17	
perl-Digest-HMAC	1.03	
perl-Digest-MD5	2.52	
perl-Digest-SHA	5.85	
perl-Encode	2.51	2.51
perl-Exporter	5.68	5.68
perl-File-Path	2.09	2.09
perl-File-Temp	0.23.01	0.23.01
perl-Filter	1.49	1.49
perl-Getopt-Long	2.40	2.40
perl-HTTP-Tiny	0.033	0.033
perl-PathTools	3.40	3.40
perl-Pod-Escapes	1.04	1.04
perl-Pod-Perldoc	3.20	3.20
perl-Pod-Simple	3.28	3.28

套件	AL1 AMI	AL2 AMI
perl-Pod-Usage	1.63	1.63
perl-Scalar-List-Utils	1.27	1.27
perl-Socket	2.010	2.010
perl-Storable	2.45	2.45
perl-Text-ParseWords	3.29	3.29
perl-Time-HiRes	1.9725	1.9725
perl-Time-Local	1.2300	1.2300
perl-constant	1.27	1.27
perl-libs	5.16.3	5.16.3
perl-macros	5.16.3	5.16.3
perl-parent	0.225	0.225
perl-podlators	2.5.1	2.5.1
perl-threads	1.87	1.87
perl-threads-shared	1.43	1.43
pinentry	0.7.6	0.8.1
pkgconfig	0.27.1	0.27.1
plymouth		0.8.9
plymouth-core-libs		0.8.9
plymouth-scripts		0.8.9
pm-utils	1.4.1	1.4.1

套件	AL1 AMI	AL2 AMI
policycoreutils	2.1.12	2.5
popt	1.13	1.13
postfix		2.10.1
procmail	3.22	
procps	3.2.8	
procps-ng		3.3.10
psacct	6.3.2	6.6.1
psmisc	22.20	22.20
pth	2.0.7	2.0.7
pygpgme		0.3
pyliblzma		0.5.3
pystache		0.5.3
python		2.7.18
python-babel		0.9.6
python-backports		1.0
python-backports-ssl_match_hostname		3.5.0.1
python-cffi		1.6.0
python-chardet		2.2.1
python-configobj		4.7.2
python-daemon		1.6

套件	AL1 AMI	AL2 AMI
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
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

套件	AL1 AMI	AL2 AMI
python-simplejson		3.2.0
python-urlgrabber		3.10
python-urllib3		1.25.9
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
python27	2.7.18	
python27-PyYAML	3.10	
python27-babel	0.9.4	

套件	AL1 AMI	AL2 AMI
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	
python27-jsonpatch	1.2	
python27-jsonpointer	1.0	

套件	AL1 AMI	AL2 AMI
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-requests	1.2.3	
python27-rsa	3.4.1	
python27-setuptools	36.2.7	
python27-simplejson	3.6.5	
python27-six	1.8.0	
python27-urlgrabber	3.10	
python27-urllib3	1.24.3	

套件	AL1 AMI	AL2 AMI
python27-virtualenv	15.1.0	
python3		3.7.16
python3-daemon		2.2.3
python3-docutils		0.14
python3-libs		3.7.16
python3-lockfile		0.11.0
python3-pip		20.2.2
python3-pystache		0.5.4
python3-setuptools		49.1.3
python3-simplejson		3.2.0
pyxattr		0.5.1
qrencode-libs		3.4.1
quota	4.00	4.01
quota-nls	4.00	4.01
rdate		1.4
readline	6.2	6.2
rmt	0.4	
rng-tools	5	6.8
rootfiles	8.1	8.1
rpcbind	0.2.0	0.2.0

套件	AL1 AMI	AL2 AMI
rpm	4.11.3	4.11.3
rpm-build-libs	4.11.3	4.11.3
rpm-libs	4.11.3	4.11.3
rpm-plugin-systemd-inhibit		4.11.3
rpm-python27	4.11.3	
rsync	3.0.6	3.1.2
rsyslog	5.8.10	8.24.0
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	
scl-utils		20130529
screen	4.0.3	4.1.0
sed	4.2.1	4.2.2
selinux-policy		3.13.1

套件	AL1 AMI	AL2 AMI
selinux-policy-targeted		3.13.1
sendmail	8.14.4	
setserial	2.17	2.17
setup	2.8.14	2.8.71
setuptool		1.19.11
sgpio	1.2.0.10	1.2.0.10
shadow-utils	4.1.4.2	4.1.5.1
shared-mime-info	1.1	1.8
slang	2.2.1	2.2.4
sqlite	3.7.17	3.7.17
sssd-client		1.16.5
strace		4.26
sudo	1.8.23	1.8.23
sysctl-defaults	1.0	1.0
sysfsutils	2.1.0	
sysstat		10.1.5
system-release	2018.03	2
systemd		219
systemd-libs		219
systemd-sysv		219

套件	AL1 AMI	AL2 AMI
systemtap-runtime		4.5
sysvinit	2.87	
sysvinit-tools		2.88
tar	1.26	1.26
tcp_wrappers	7.6	7.6
tcp_wrappers-libs	7.6	7.6
tcpdump		4.9.2
tcsh		6.18.01
teamd		1.27
time	1.7	1.7
tmpwatch	2.9.16	
traceroute	2.0.14	2.0.22
ttmkfdir	3.0.9	
tzdata	2023c	2023c
tzdata-java	2023c	
udev	173	
unzip	6.0	6.0
update-motd	1.0.1	1.1.2
upstart	0.6.5	
usermode		1.111

套件	AL1 AMI	AL2 AMI
ustr	1.0.4	1.0.4
util-linux	2.23.2	2.30.2
vim-common	9.0.1712	2081 年 9 月 0 日
vim-data	9.0.1712	2081 年 9 月 0 日
vim-enhanced	9.0.1712	2081 年 9 月 0 日
vim-filesystem	9.0.1712	2081 年 9 月 0 日
vim-minimal	9.0.1712	2081 年 9 月 0 日
virt-what		1.18
wget	1.18	1.14
which	2.19	2.20
words	3.0	3.0
xfsdump		3.1.8
xfsprogs		5.0.0
xorg-x11-font-utils	7.2	
xorg-x11-fonts-Type1	7.2	
xxd	9.0.1712	2081 年 9 月 0 日
xz	5.2.2	5.2.2
xz-libs	5.2.2	5.2.2
yajl		2.0.4
yum	3.4.3	3.4.3

套件	AL1 AMI	AL2 AMI
yum-langpacks		0.4.2
yum-metadata-parser	1.1.4	1.1.4
yum-plugin-priorities	1.1.31	1.1.31
yum-plugin-upgrade-helper	1.1.31	
yum-utils	1.1.31	1.1.31
zip	3.0	3.0
zlib	1.2.8	1.2.7

比較安裝在 AL1 和 AL2 基礎容器映像上的套件

套件	AL1 容器	AL2 容器
amazon-linux-extras		2.0.3
basesystem	10.0	10.0
bash	4.2.46	4.2.46
bzip2-libs	1.0.6	1.0.6
ca-certificates	2023.2.62	2023.2.62
chkconfig	1.3.49.3	1.7.4
coreutils	8.22	8.22
cpio		2.12
curl	7.61.1	8.3.0
cyrus-sasl-lib	2.1.23	2.1.26

套件	AL1 容器	AL2 容器
db4	4.7.25	
db4-utils	4.7.25	
diffutils		3.3
elfutils-libelf	0.168	0.176
expat	2.1.0	2.1.0
file-libs	5.37	5.11
filesystem	2.4.30	3.2
findutils		4.5.11
gawk	3.1.7	4.0.2
gdbm	1.8.0	1.13
glib2	2.36.3	2.56.1
glibc	2.17	2.26
glibc-common	2.17	2.26
glibc-langpack-en		2.26
glibc-minimal-langpack		2.26
gmp	6.0.0	6.0.0
gnupg2	2.0.28	2.0.22
gpgme	1.4.3	1.3.2
grep	2.20	2.20
gzip	1.5	

套件	AL1 容器	AL2 容器
info	5.1	5.1
keyutils-libs	1.5.8	1.5.8
krb5-libs	1.15.1	1.15.1
libacl	2.2.49	2.2.51
libassuan	2.0.3	2.1.0
libattr	2.4.46	2.4.46
libblkid		2.30.2
libcap	2.16	2.54
libcom_err	1.43.5	1.42.9
libcrypt		2.26
libcurl	7.61.1	8.3.0
libdb		5.3.21
libdb-utils		5.3.21
libffi	3.0.13	3.0.13
libgcc		7.3.1
libgcc72	7.2.1	
libgcrypt	1.5.3	1.5.3
libgpg-error	1.11	1.12
libicu	50.2	
libidn2	2.3.0	2.3.0

套件	AL1 容器	AL2 容器
libmetalink		0.1.3
libmount		2.30.2
libnghttp2	1.33.0	1.41.0
libpsl	0.6.2	
libselinux	2.1.10	2.5
libsepol	2.1.7	2.5
libssh2	1.4.2	1.4.3
libstdc++		7.3.1
libstdc++72	7.2.1	
libtasn1	2.3	4.10
libunistring	0.9.3	0.9.3
libuuid		2.30.2
libverto	0.2.5	0.2.5
libxml2	2.9.1	2.9.1
libxml2-python27	2.9.1	
lua	5.1.4	5.1.4
make	3.82	
ncurses	5.7	6.0
ncurses-base	5.7	6.0
ncurses-libs	5.7	6.0

套件	AL1 容器	AL2 容器
nspr	4.25.0	4.35.0
nss	3.53.1	3.90.0
nss-pem	1.0.3	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	3.90.0
nss-util	3.53.1	3.90.0
openldap	2.4.40	2.4.44
OpenSSL	1.0.2k	
openssl-libs		1.0.2k
p11-kit	0.18.5	0.23.22
p11-kit-trust	0.18.5	0.23.22
pcre	8.21	8.32
pinentry	0.7.6	0.8.1
pkgconfig	0.27.1	
popt	1.13	1.13
pth	2.0.7	2.0.7
pygpgme		0.3
pyliblzma		0.5.3

套件	AL1 容器	AL2 容器
python		2.7.18
python-iniparse		0.4
python-libs		2.7.18
python-pycurl		7.19.0
python-urlgrabber		3.10
python2-rpm		4.11.3
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	
python27-pygpgme	0.3	
python27-pyliblzma	0.5.3	
python27-pyxattr	0.5.0	
python27-urlgrabber	3.10	
pyxattr		0.5.1
readline	6.2	6.2
rpm	4.11.3	4.11.3
rpm-build-libs	4.11.3	4.11.3

套件	AL1 容器	AL2 容器
rpm-libs	4.11.3	4.11.3
rpm-python27	4.11.3	
sed	4.2.1	4.2.2
setup	2.8.14	2.8.71
shared-mime-info	1.1	1.8
sqlite	3.7.17	3.7.17
sysctl-defaults	1.0	
system-release	2018.03	2
tar	1.26	
tzdata	2023c	2023c
vim-data		2081 年 9 月 0 日
vim-minimal		2081 年 9 月 0 日
xz-libs	5.2.2	5.2.2
yum	3.4.3	3.4.3
yum-metadata-parser	1.1.4	1.1.4
yum-plugin-ovl	1.1.31	1.1.31
yum-plugin-priorities	1.1.31	1.1.31
yum-utils	1.1.31	
zlib	1.2.8	1.2.7

Amazon EC2 上的 AL2 Amazon EC2

> **Note**
>
> AL2 不再是 Amazon Linux 的目前版本。AL2023 是 AL2 的後續版本。如需詳細資訊，請參閱《AL2023 使用者指南》中的比較 AL2 和 AL2023 中的套件變更AL2023清單。

主題

使用 AL2 AMI 啟動 Amazon EC2 執行個體 AL2

您可以使用 AL2 AMI 啟動 Amazon EC2 執行個體。 AL2 如需詳細資訊，請參閱步驟 1：啟動執行個體。

使用 Systems Manager 尋找最新的 AL2 AMI

Amazon EC2 為 維護的公有 AMIs 提供 AWS Systems Manager 公有參數 AWS ，您可以在啟動執行個體時使用這些參數。例如，EC2-provided參數`/aws/service/ami-amazon-linux-latest/amzn2-ami-kernel-default-hvm-x86_64-gp2`適用於所有區域，並一律指向指定區域中 AL2 AMI 的最新版本。

若要使用 尋找最新的 AL2023 AMI AWS Systems Manager，請參閱開始使用 AL2023。

該 Amazon EC2 AMI 公有參數可從以下路徑取得：

```
/aws/service/ami-amazon-linux-latest
```

您可以執行下列 AWS CLI 命令，檢視目前 AWS 區域中所有 Amazon Linux AMIs的清單。

```
aws ssm get-parameters-by-path --path /aws/service/ami-amazon-linux-latest --query
 "Parameters[].Name"
```

使用公有參數啟動執行個體

下列範例使用 EC2-provided公有參數，使用最新的 AL2 AMI 啟動m5.xlarge執行個體。

若要在命令中指定參數，請使用下列語法：resolve:ssm:*public-parameter*，其中 resolve:ssm 是標準前綴，而 public-parameter 是公有參數的路徑和名稱。

在此範例中，--count 和 --security-group 參數不包含在內。對於 --count，預設值為 1。如果您有預設 VPC 和預設安全群組，則會使用這些預設項目。

```
aws ec2 run-instances
    --image-id resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-kernel-
default-hvm-x86_64-gp2
    --instance-type m5.xlarge
    --key-name MyKeyPair
```

如需詳細資訊，請參閱AWS Systems Manager 《 使用者指南》中的使用公有參數。

了解 Amazon Linux 2 AMI 名稱

Amazon Linux 2 AMI 名稱使用以下命名機制：

```
amzn2-ami-[minimal-][kernel-{5.10,default,4.14}]-hvm-{x86_64,aarch64}-
{ebs,gp2}
```

- 最小 AMIs隨附一組最小化的預先安裝套件，以減少映像大小。
- kernel-VERSION 會決定個別 AMI 上預先安裝的核心版本：
  - kernel-5.10 選取 Linux 核心 5.10 版。這是 AL2 的建議核心版本。
  - kernel-default 會為 AL2 選取建議的預設核心。這是 kernel-5.10 的別名。

  - `kernel-4.14` 選取 Linux 核心 4.14 版。這僅適用於與舊版 AMI 的相容性。請勿將此版本用於新的執行個體啟動。預期此 AMI 變成不受支援。
  - 存在一組特殊的 AMI 名稱，而不參考特定的核心。這些 AMIs 是 kernel-4.14 的別名。這些 AMIs 僅為相容於較舊的 AMI 版本而提供。請勿將此 AMI 名稱用於新的執行個體啟動。預期要更新這些 AMIs的核心。
- x86_64/aarch64 決定要在其中執行 AMI 的 CPU 平台。針對 Intel 和 AMD 型 EC2 執行個體選取 x86_64。選取 EC2 Graviton 執行個體的 aarch64。
- ebs/gp2 會決定用來提供個別 AMI 的 EBS 磁碟區類型。如需參考，請參閱 EBS 磁碟區類型。一律選取 gp2。

連線至 Amazon EC2 執行個體

有數種方式可以連線至 Amazon Linux 執行個體，包括 SSH AWS Systems Manager Session Manager和 EC2 Instance Connect。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的連線至 Linux 執行個體。

SSH 使用者和 sudo

根據預設，Amazon Linux 不允許遠端`root`安全殼層 (SSH)。此外，也會停用密碼身分驗證，以防止暴力破解攻擊。若要允許 SSH 登入 Amazon Linux 執行個體，您必須在啟動時提供執行個體的金鑰對。您也必須設定用於啟動執行個體的安全群組，以允許 SSH 存取。根據預設，唯一可以使用 SSH 遠端登入的帳戶是 `ec2-user`。此帳戶也具有 sudo 權限。如果您啟用遠端`root`登入，請注意它比依賴金鑰對和次要使用者更不安全。

AL2 AMI 開機模式

AL2 AMIs未設定開機模式參數。從 AL2 AMIs執行個體遵循執行個體類型的預設開機模式值。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的開機模式。

套件儲存庫

此資訊適用於 AL2。如需有關 AL2023 的資訊，請參閱《Amazon Linux 2023 使用者指南》中的管理 AL2023 中的套件和作業系統更新。

AL2 和 AL1 旨在與每個 Amazon EC2 AWS Region 中託管的線上套件儲存庫搭配使用。儲存庫可供所有區域使用，且可用 yum 更新工具存取。各區域的託管儲存庫可讓我們快速部署更新，不會產生任何數據傳輸費。

> **Important**
>
> AL1 的最後一個版本已於 2023 年 12 月 31 日達到 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。如需詳細資訊，請參閱 Amazon Linux AMI 生命週期結束

如果您不需要保留執行個體的資料或自訂，您可以使用目前的 AL2 AMI 啟動新的執行個體。如果您需要保留執行個體的資料或自訂，您可以透過 Amazon Linux 套件儲存庫來維護這些執行個體。這些儲存庫將包含所有已更新的套件。您可選擇將這些更新套用至執行中的執行個體。即使發行新版本，舊版的 AMI 和更新套件仍可繼續使用。

> **Note**
>
> 若要在 Amazon EC2 執行個體上更新和安裝沒有網際網路存取權的套件，請參閱如何在執行 AL1, AL22 或 AL2023 的 Amazon EC2 執行個體上更新 yum 或安裝沒有網際網路存取權的套件？

若要安裝套件，請使用下列命令：

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

如果您發現 Amazon Linux 未包含所需的應用程式，可以直接將應用程式安裝到 Amazon Linux 執行個體上。Amazon Linux 使用 RPMs和 yum進行套件管理，這可能是安裝新應用程式最直接的方式。您應先檢查我們的集中 Amazon Linux 儲存庫上是否有應用程式可用，因為該儲存庫提供許多應用程式。您可以從該處將這些應用程式新增至 Amazon Linux 執行個體。

若要將應用程式上傳到執行中的 Amazon Linux 執行個體，請用 scp 或 sftp，然後登入您的執行個體並設定應用程式。也可從內建的 cloud-init 套件使用 PACKAGE_SETUP 動作，在執行個體啟動期間上傳您的應用程式。如需詳細資訊，請參閱在 AL2 上使用 cloud-init。

安全性更新

使用 套件儲存庫提供安全性更新。安全更新和更新的 AMI 安全提醒都會發佈在 Amazon Linux 安全中心。如需 AWS 安全政策的詳細資訊，或需回報安全問題，請參閱 AWS Cloud Security。

AL1 和 AL2 設定為在啟動時下載並安裝關鍵或重要的安全性更新。此組態中不包含核心更新。

在 AL2023 中，相較於 AL1 和 AL2，此組態已變更。如需 AL2023 安全性更新的詳細資訊，請參閱《Amazon Linux 2023 使用者指南》中的安全性更新和功能。

我們建議您在啟動後進行適用於您使用案例的必要更新。例如，您可能想要在啟動時套用所有更新（不只是安全性更新），或評估每個更新，並僅套用適用於您系統的更新。使用下列 cloud-init 設定進行控制：repo_upgrade。以下 cloud-init 組態的程式碼片段顯示如何變更傳送到執行個體初始化之使用者資料文字中的設定：

```
#cloud-config
repo_upgrade: security
```

repo_upgrade 可能的值如下：

critical

套用未完成的重大安全性更新。

important

套用未完成的重要和重要的安全性更新。

medium

套用未完成的重大、重要和中等安全性更新。

low

套用所有未完成的安全性更新，包括低嚴重性安全性更新。

security

套用 Amazon 標示為安全性更新且尚未完成的關鍵或重要更新。

bugfix

套用 Amazon 標示為錯誤修正的更新。錯誤修正為更大的更新集合，其中包含安全更新和各種其他次要錯誤的修正。

all

套用所有適用的可用更新，無論其分類為何。

none

不要在啟動時套用任何更新至執行個體。

> **注意**
>
> Amazon Linux 不會將任何更新標記為 `bugfix`。若要從 Amazon Linux 套用非安全性相關更新，請使用 `repo_upgrade: all`。

`repo_upgrade` 的預設設定為安全。也就是說，如果您未在使用者資料中指定其他值，Amazon Linux 將在啟動時對當時已安裝的任何套件執行安全更新。Amazon Linux 也會通知您已安裝套件的任何更新，方法是在登入時使用 `/etc/motd` 檔案列出可用更新數。若要安裝這些更新，您需要在執行個體上執行 sudo yum upgrade。

儲存庫組態

對於 AL1 和 AL2，AMIs 是建立 AMI 時可用的套件快照，但安全性更新除外。任何不在原始 AMI 上，但在執行時間安裝的套件，都將是可用的最新版本。若要取得適用於 AL2 的最新套件，請執行 yum update -y。

> **疑難排解秘訣**
>
> 如果您在奈米執行個體類型`cannot allocate memory`yum update上發生錯誤，例如 `t3.nano`，您可能需要配置交換空間以啟用更新。

對於 AL2023，相較於 AL1 和 AL2，儲存庫組態已變更。如需有關 AL2023 儲存庫的詳細資訊，請參閱管理套件和作業系統更新。

AL2023 之前的版本設定為提供源源不斷的更新，從 Amazon Linux 的次要版本滾動到下一個版本，也稱為滾動發行版。最佳實務是，建議您將 AMI 更新為最新的可用 AMI，而不是啟動舊 AMIs並套用更新。

主要 Amazon Linux 版本之間不支援就地升級，例如從 AL1 升級至 AL2，或從 AL2 升級至 AL2023。如需詳細資訊，請參閱Amazon Linux 可用性。

在 AL2 上使用 cloud-init

cloud-init 套件為 Canonical 所建立的開放原始碼應用程式，用於在 Amazon EC2 等雲端運算環境下引導 Linux 映像。Amazon Linux 包含自訂版本的 cloud-init。這可讓您指定執行個體在開機時應該發生的

動作。您可在執行個體啟動時透過使用者資料欄位將所需動作傳送至 cloud-init。這表示您可在許多使用案例下用常用 AMI，並在啟動時動態設定它們。Amazon Linux 也使用 cloud-init 執行 ec2-user 帳戶的初始組態。

如需詳細資訊，請參閱 cloud-init 文件。

Amazon Linux 使用在 `/etc/cloud/cloud.cfg.d` 和 `/etc/cloud/cloud.cfg` 中找到的 cloud-init 動作。您可在 `/etc/cloud/cloud.cfg.d` 中建立自己的 cloud-init 動作檔案。此目錄中的所有檔案將由 cloud-init 讀取。這些檔案將按辭典的順序讀取，且新檔案將覆寫舊檔案中的值。

cloud-init 套件將在開機時對執行個體執行這些 (和之後的) 一般組態任務：

- 設定預設的地區設定.
- 設定主機名稱.
- 剖析及處理使用者資料.
- 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.
- 執行使用者資料中找到的使用者指令碼。
- 掛載執行個體存放磁碟區 (如適用)。
  - 根據預設，`ephemeral0` 執行個體存放磁碟區將掛載在 `/media/ephemeral0` (若存在且包含有效的檔案系統的話)；否則將不掛載。
  - 根據預設，將掛載任何與執行個體關聯的置換磁碟區 (僅適用於 `m1.small` 和 `c1.medium` 執行個體類型)。
  - 您可用下列的 cloud-init 指令覆寫預設的執行個體存放磁碟區掛載：

    ```
    #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 的執行個體存放磁碟區，因此您必須先分割及格式化磁碟區，然後再掛載。如需詳細資訊，請參閱執行個體存放區磁碟區TRIM支援。您可在開機時用

disk_setup 模組分割及格式化執行個體存放磁碟區。如需詳細資訊，請參閱 cloud-init 文件中的磁碟設定。

支援的使用者資料格式

cloud-init 套件支援各種格式的使用者資料處理：

- Gzip
  - 如果使用者資料經過 gzip 壓縮，Cloud-init 會解壓縮資料並適當處理。
- MIME 分段
  - 您可用 MIME 分段檔案指定一種以上的資料。例如，您可以同時指定使用者資料指令碼和雲端組態類型。分段檔案中的每個部分均可為 cloud-init 處理，前提是需為支援的其中一種格式。
- Base64 解碼
  - 如果使用者資料是 base64 編碼，Cloud-init 會判斷是否可以將解碼的資料理解為其中一個支援的類型。如果它能了解解碼的資料，它將解碼資料並用正確方式處理資料。否則，將傳回原封不動的 base64 資料。
- 使用者資料指令碼
  - 開頭為 #! 或 Content-Type: text/x-shellscript。
  - 指令碼會在第一次開機週期期間由 /etc/init.d/cloud-init-user-scripts 執行。發生於開機流程晚期 (執行初始組態動作之後)。
- 包含檔案
  - 開頭為 #include 或 Content-Type: text/x-include-url。
  - 此內容為包含檔案。檔案包含 URL 清單，每行一個。將讀取每個 URL，且其內容將通過相同的規則集合。從 URL 讀取的內容可能為 gzip 壓縮、MIME 分段或純文字。
- 雲端組態資料
  - 開頭為 #cloud-config 或 Content-Type: text/cloud-config。
  - 此內容是雲端組態資料。
- 啟動任務 (AL2 不支援)
  - 開頭為 #upstart-job 或 Content-Type: text/upstart-job。
  - 此內容存放在 的檔案中/etc/init，而且 upstart 會像使用其他 upstart 任務一樣使用內容。
- 雲端 boothook
  - 開頭為 #cloud-boothook 或 Content-Type: text/cloud-boothook。
  - 此內容為 boothook 資料。其存放在 /var/lib/cloud 下的檔案內，且將立即執行。

- 這是最早可用的關聯。未提供只允許其執行一次的機制。boothook 必須自我處理。其將提供含環境變數 `INSTANCE_ID` 的執行個體 ID。請用此變數提供 boothook 資料每執行個體一次的集合。

設定 AL2 執行個體

成功啟動並登入 AL2 執行個體後，您可以對其進行變更。您可以透過許多方式設定執行個體以符合特定應用程式的需求。以下是可協助您開始使用的常見任務。

內容

常見組態案例

Amazon Linux 的基礎分佈包含基本伺服器操作需要的軟體套件和公用程式。但是，您可以從不同軟體儲存庫中取得更多軟體套件。您甚至可以從來源碼建置更多套件。如需從這些位置安裝和建置軟體的詳細資訊，請參閱管理 AL2 執行個體上的軟體。

Amazon Linux 執行個體附帶預先設定的 `ec2-user`，但建議您新增其他不具有超級使用者權限的使用者。如需新增和移除使用者的詳細資訊，請參閱《Amazon EC2 使用者指南》中的管理Linux執行個體上的使用者。

若您有自己的網路，且其具備已註冊的網域名稱，您可以變更執行個體的主機名稱，將其識別為該網域的一部分。您也可以變更系統提示，顯示更有意義的名稱，而無須變更主機名稱設定。如需詳細資訊，請參閱變更 AL2 執行個體的主機名稱。您可以將執行個體設定為使用動態 DNS 服務提供者。如需詳細資訊，請參閱在 AL2 執行個體上設定動態 DNS。

當您在 Amazon EC2 上啟動執行個體時，您可以選擇將使用者資料傳遞給執行個體，用來執行常見的組態任務，甚至在執行個體啟動之後執行指令碼。您可以將兩種類型的使用者資料傳遞給 Amazon EC2：cloud-init 指示詞和 shell 指令碼。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的在執行個體啟動Linux時執行命令。

管理 AL2 執行個體上的軟體

Amazon Linux 的基礎分佈包含基本伺服器操作需要的軟體套件和公用程式。

此資訊適用於 AL2。如需有關 AL2023 的資訊，請參閱《Amazon Linux 2023 使用者指南》中的管理 AL2023 中的套件和作業系統更新。

將軟體維持在最新狀態是非常重要的。Linux 分佈中的許多套件都會頻繁更新以修正錯誤、新增功能和提供保護，以免遭利用安全漏洞。如需詳細資訊，請參閱更新 AL2 執行個體上的執行個體軟體。

根據預設，AL2 執行個體會在啟用下列儲存庫的情況下啟動：

- `amzn2-core`
- `amzn2extra-docker`

雖然這些儲存庫中有許多套件可供 更新 AWS，但您可能想要安裝另一個儲存庫中包含的套件。如需詳細資訊，請參閱在 AL2 執行個體上新增儲存庫。如需在已啟用的儲存庫中尋找並安裝套件的協助，請參閱 在 AL2 執行個體上尋找並安裝軟體套件。

並非所有軟體都可從存放在儲存庫中的軟體套件取得。有些軟體必須在執行個體上從來源碼編譯取得。如需詳細資訊，請參閱準備在 AL2 執行個體上編譯軟體。

AL2 執行個體使用 yum 套件管理員管理其軟體。yum 套件管理員可安裝、移除和更新軟體，也能管理每個套件的所有依存性。

目錄

更新 AL2 執行個體上的執行個體軟體

將軟體維持在最新狀態是非常重要的。Linux 分佈中的套件都會頻繁更新以修正錯誤、新增功能和提供保護，以免遭利用安全漏洞。當您第一次啟動並連線至 Amazon Linux 執行個體時，您可能會看到一則訊息，提示您更新軟體套件以維護安全。本節說明如何更新整個系統或單一套件。

此資訊適用於 AL2。如需有關 AL2023 的資訊，請參閱《Amazon Linux 2023 使用者指南》中的管理 AL2023 中的套件和作業系統更新。

如需 AL2 變更和更新的相關資訊，請參閱 AL2 版本備註。

如需有關 AL2023 變更與更新的資訊，請參閱 AL2023 版本備註。

> **Important**
>
> 如果您將使用 Amazon Linux 2 AMI 的 EC2 執行個體啟動到僅限 IPv6 的子網中，則必須連線到該執行個體並執行 `sudo amazon-linux-https disable`。這可以讓您的 AL2 執行個體使用 http 修補程式服務來透過 IPv6 連接到 S3 中的 yum 儲存庫。

若要更新 AL2 執行個體上的所有套件

1. (選用) 在您的 shell 視窗中啟動 screen 工作階段。有時候您可能會遭遇網路中斷，其會中斷您執行個體的 SSH 連線。當在長時間的軟體更新過程中發生這種情況時，可能會使執行個體進入可復原但混淆的狀態。screen 工作階段可讓您即使中斷連線，也能繼續執行更新，並可在稍後重新連線到工作階段而不會發生任何問題。

   a. 執行 screen 命令以開始工作階段。

   ```
   [ec2-user ~]$ screen
   ```

   b. 若您的工作階段中斷連線，請重新登入您的執行個體並列出可用的螢幕。

   ```
   [ec2-user ~]$ screen -ls
   There is a screen on:
    17793.pts-0.ip-12-34-56-78 (Detached)
   1 Socket in /var/run/screen/S-ec2-user.
   ```

   c. 使用 screen -r 命令和在先前命令中取得的程序 ID 重新連線到螢幕。

   ```
   [ec2-user ~]$ screen -r 17793
   ```

   d. 當您結束使用 screen 命令，請使用 exit 命令來關閉工作階段。

   ```
   [ec2-user ~]$ exit
   [screen is terminating]
   ```

2. 執行 yum update 命令。您可以選擇性新增 `--security` 標記，僅套用安全更新。

   ```
   [ec2-user ~]$ sudo yum update
   ```

3. 檢閱列出的套件，輸入 **y**，然後按下 Enter 鍵接受更新。更新系統上的所有套件可能需要幾分鐘的時間。yum 輸出會在執行過程中顯示更新的狀態。
4. （選用）重新啟動執行個體，以確保您使用的是更新中的最新套件和程式庫；在重新啟動之前，不會載入核心更新。任何對 glibc 程式庫進行的更新都需要重新啟動。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，但系統重新開機可確保所有先前的套件和程式庫更新都已完成。

更新 AL2 執行個體上的單一套件

使用此程序更新單一套件 (及其依存性)，而非整個系統。

1. 使用要更新的套件名稱執行 yum update 命令。

```
[ec2-user ~]$ sudo yum update openssl
```

2. 檢閱列出的套件資訊，輸入 **y**，然後按下 Enter 鍵接受更新。有時候若有需要解析的套件依存性，便會列出一個以上的套件。yum 輸出會在執行過程中顯示更新的狀態。
3. （選用）重新啟動執行個體，以確保您使用的是更新中的最新套件和程式庫；在重新啟動之前，不會載入核心更新。任何對 glibc 程式庫進行的更新都需要重新啟動。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，但系統重新開機可確保所有先前的套件和程式庫更新都已完成。

在 AL2 執行個體上新增儲存庫

此資訊適用於 AL2。如需 AL2023 的相關資訊，請參閱《Amazon Linux 2023 使用者指南》中的透過 AL2023 上的版本控制儲存庫進行確定性升級。

根據預設，AL2 執行個體會在啟用下列儲存庫的情況下啟動：

- amzn2-core
- amzn2extra-docker

雖然這些儲存庫中有許多可用的套件都會由 Amazon Web Services 更新，但是您希望安裝的套件可能會包含在其他儲存庫中。

若要使用 yum 命令從不同的儲存庫安裝套件，您需要為 /etc/yum.conf 檔案或其自身位於 *repository*.repo 目錄中的 /etc/yum.repos.d 檔案新增儲存庫的資訊。您可以手動執行此作業，但大多數的 yum 儲存庫會在其儲存庫 URL 中提供自身的 *repository*.repo 檔案。

判斷已安裝哪些 yum 儲存庫

請使用以下命令列出已安裝的 yum 儲存庫：

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

結果輸出會列出已安裝的儲存庫，並報告每個儲存庫的狀態。已啟用的儲存庫會顯示其包含的套件數目。

為 /etc/yum.repos.d 新增 yum 儲存庫

1. 尋找 .repo 檔案的位置。這會根據您將新增的儲存庫而有所不同。在此範例中，.repo 檔案位於 https://www.*example*.com/*repository*.repo。
2. 使用 yum-config-manager 命令新增儲存庫。

```
[ec2-user ~]$ sudo yum-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。

在 /etc/yum.repos.d 中啟用 yum 儲存庫

使用 yum-config-manager 命令搭配 --enable *repository* 旗標。以下命令會啟用來自 Fedora 專案的 Extra Packages for Enterprise Linux (EPEL) 儲存庫。根據預設，此儲存庫位在 Amazon Linux AMI 執行個體上的 /etc/yum.repos.d 中，但並未啟用。

```
[ec2-user ~]$ sudo yum-config-manager --enable epel
```

如需詳細資訊，以及下載此套件的最新版本，請參閱 https：//https://fedoraproject.org/wiki/EPEL。

在 AL2 執行個體上尋找並安裝軟體套件

您可以使用套件管理工具來尋找並安裝軟體套件。在 Amazon Linux 2 中，預設軟體套件管理工具為 YUM。在 AL2023 中，預設軟體套件管理工具為 DNF。如需詳細資訊，請參閱《Amazon Linux 2023 使用者指南》中的套件管理工具。

尋找 AL2 執行個體上的軟體套件

您可以使用 yum search 命令搜尋在您設定的儲存庫中可用套件的描述。這在您不知道希望安裝之套件的精確名稱時非常有用。您只需要將關鍵字搜尋附加到命令即可；針對多個關鍵字搜尋，請使用引號包圍搜尋查詢。

```
[ec2-user ~]$ yum search "find"
```

以下為範例輸出。

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
============================== N/S matched: find ===============================
findutils.x86_64 : The GNU versions of find utilities (find and xargs)
gedit-plugin-findinfiles.x86_64 : gedit findinfiles plugin
ocaml-findlib-devel.x86_64 : Development files for ocaml-findlib
perl-File-Find-Rule.noarch : Perl module implementing an alternative interface to
 File::Find
robotfindskitten.x86_64 : A game/zen simulation. You are robot. Your job is to find
 kitten.
mlocate.x86_64 : An utility for finding files by name
ocaml-findlib.x86_64 : Objective CAML package manager and build helper
perl-Devel-Cycle.noarch : Find memory cycles in objects
perl-Devel-EnforceEncapsulation.noarch : Find access violations to blessed objects
perl-File-Find-Rule-Perl.noarch : Common rules for searching for Perl things
perl-File-HomeDir.noarch : Find your home and other directories on any platform
perl-IPC-Cmd.noarch : Finding and running system commands made easy
perl-Perl-MinimumVersion.noarch : Find a minimum required version of perl for Perl code
texlive-xesearch.noarch : A string finder for XeTeX
valgrind.x86_64 : Tool for finding memory management bugs in programs
valgrind.i686 : Tool for finding memory management bugs in programs
```

引號中的多個關鍵字搜尋查詢只會傳回與查詢完全相符的結果。若您沒有看到預期的套件，請將您的搜尋簡化至一個關鍵字，然後掃描結果。您也可以嘗試關鍵字同義詞來擴大您的搜尋。

如需 AL2 套件的詳細資訊，請參閱下列內容：

- AL2 Extras 程式庫
- 套件儲存庫

在 AL2 執行個體上安裝軟體套件

在 AL2 中，yum 套件管理工具會搜尋所有已啟用的儲存庫，尋找不同的軟體套件，並處理軟體安裝程序中的任何相依性。如需在 AL2023 中安裝軟體套件的詳細資訊，請參閱《Amazon Linux 2023 使用者指南》中的管理套件和作業系統更新。

若要從儲存庫中安裝套件

請使用 yum install ***package*** 命令，並使用要安裝之軟體的名稱來取代##。例如，若要安裝 links 文字型 Web 瀏覽器，請輸入以下命令。

```
[ec2-user ~]$ sudo yum install links
```

安裝您已下載的 RPM 套件檔案

您也可以使用 yum install 安裝已從網際網路下載的 RPM 套件檔案。若要執行此操作，請將 RPM 檔案的路徑名稱附加到安裝命令 (而非儲存庫套件名稱)。

```
[ec2-user ~]$ sudo yum install my-package.rpm
```

列出已安裝的套件

若要檢視執行個體上已安裝的套件清單，請使用下列命令。

```
[ec2-user ~]$ yum list installed
```

準備在 AL2 執行個體上編譯軟體

網際網路上可用的開放原始碼軟體並未預先編譯並不能從套件儲存庫中下載。您最終可能會面臨需要自行從來源碼編譯的軟體套件。若要讓您的系統能夠在 AL2 和 Amazon Linux 中編譯軟體，您需要安裝多種開發工具，例如 make、 gcc和 autoconf。

因為軟體編譯並非每個 Amazon EC2 執行個體需要的任務，根據預設不會安裝這些工具，但仍可以在稱為 "Development Tools" 的套件群組中取得。您可以使用 yum groupinstall 命令輕鬆為執行個體新增該群組。

```
[ec2-user ~]$ sudo yum groupinstall "Development Tools"
```

軟體來源碼套件通常可做為壓縮封存檔案 (稱為 tarball) 進行下載 (例如 https://github.com/ 和 http://sourceforge.net/ 等網站)。這些 tarball 通常會具有 `.tar.gz` 檔案副檔名。您可以使用 tar 命令解壓縮這些封存檔。

```
[ec2-user ~]$ tar -xzf software.tar.gz
```

在您解壓縮及解除封存來源碼套件後，建議您在來源碼目錄中尋找 `README` 或 `INSTALL` 檔案。這些檔案可針對編譯和安裝來源碼提供您進一步的說明。

擷取 Amazon Linux 套件的原始程式碼

Amazon Web Services 提供維護套件的來源碼。您可以使用 yumdownloader --source 命令下載任何已安裝套件的來源碼。

執行 yumdownloader --source ***package*** 命令來下載 *package* 的來源碼。例如，若要下載 `htop` 套件的來源碼，請輸入以下命令。

```
[ec2-user ~]$ yumdownloader --source htop

Loaded plugins: priorities, update-motd, upgrade-helper
Enabling amzn-updates-source repository
Enabling amzn-main-source repository
amzn-main-source
                        | 1.9 kB  00:00:00
amzn-updates-source
                        | 1.9 kB  00:00:00
(1/2): amzn-updates-source/latest/primary_db
                        |  52 kB  00:00:00
(2/2): amzn-main-source/latest/primary_db
                        | 734 kB  00:00:00
htop-1.0.1-2.3.amzn1.src.rpm
```

來源 RPM 的位置位於您執行命令的目錄中。

Amazon EC2 AL2 執行個體的處理器狀態控制

C-state 可控制核心在閒置狀態要進入的休眠等級。C-state 從 C0 (最淺閒置狀態，此時核心處於喚醒狀態並執行指令) 開始編號，最高可到 C6 (最深閒置狀態，此時核心會關閉)。

P-states 則可控制所需的核心效能 (CPU 頻率)。P-state 從 P0 (最高效能設定，此時核心可視需要使用 Intel Turbo Boost 技術來提高頻率) 開始編號，之後則從 P1 (請求最大基準頻率的 P-state) 到 P15 (最低適用頻率)。

建議您變更 C-state 或 P-state 設定來提高處理器效能穩定性、減少延遲或針對特定工作負載微調執行個體。預設 C-state 和 P-state 設定提供最大效能，適合多數工作負載使用。然而，若您的應用程式更適合犧牲較高的單核心或雙核心頻率以降低延遲，或在較低頻率更能展現穩定效能 (而不適合使用突增 Turbo Boost 頻率)，請考慮嘗試調整這些執行個體的 C-state 或 P-state 設定。

如需可讓作業系統控制處理器 C-state 和 P-state 的 Amazon EC2 執行個體類型相關資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 執行個體的處理器狀態控制。 Amazon EC2

下列各節會說明處理器狀態的不同組態，以及如何監控組態的效果。這些程序是針對 Amazon Linux 撰寫，並適用於 Amazon Linux；不過，它們也可能適用於 Linux 核心版本為 3.9 或更新版本的其他 Linux 發行版本。

> **Note**
>
> 此頁面上的範例使用以下項目：
>
> - turbostat 公用程式，用於顯示處理器頻率和 C-state 資訊。根據預設，turbostat 執行個體在 Amazon Linux 上可用。
> - stress 命令，用於模擬工作負載。若要安裝 stress，首先透過執行 sudo amazon-linux-extras install epel 啟用 EPEL 儲存庫，然後執行 sudo yum install -y stress。
>
> 若輸出未顯示 C-state 資訊，請將 --debug 選項併入命令 (sudo turbostat --debug stress ***<options>***) 中。

內容

具備最高 Turbo Boost 頻率的最佳效能

此為 Amazon Linux AMI 的預設處理器狀態控制組態，建議多數工作負載使用。此組態提供最佳效能，而且變異較低。允許未使用的核心進入更深的休眠狀態，可提供單核心或雙核心處理器達到最大 Turbo Boost 潛力所需的散熱餘裕。

下列範例以 `c4.8xlarge` 執行個體為例，其中具備兩個執行工作並達到最大處理器 Turbo Boost 頻率的運作中核心。

```
[ec2-user ~]$ sudo turbostat stress -c 2 -t 10
stress: info: [30680] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
stress: info: [30680] successful run completed in 10s
pk cor CPU    %c0  GHz  TSC SMI    %c1    %c3    %c6    %c7   %pc2   %pc3   %pc6   %pc7
  Pkg_W RAM_W PKG_% RAM_%
            5.54 3.44 2.90   0   9.18   0.00  85.28   0.00   0.00   0.00   0.00   0.00
  94.04 32.70 54.18  0.00
 0   0   0   0.12 3.26 2.90   0   3.61   0.00  96.27   0.00   0.00   0.00   0.00   0.00
  48.12 18.88 26.02  0.00
 0   0  18   0.12 3.26 2.90   0   3.61
 0   1   1   0.12 3.26 2.90   0   4.11   0.00  95.77   0.00
 0   1  19   0.13 3.27 2.90   0   4.11
 0   2   2   0.13 3.28 2.90   0   4.45   0.00  95.42   0.00
 0   2  20   0.11 3.27 2.90   0   4.47
 0   3   3   0.05 3.42 2.90   0  99.91   0.00   0.05   0.00
 0   3  21  97.84 3.45 2.90   0   2.11
...
 1   1  10   0.06 3.33 2.90   0  99.88   0.01   0.06   0.00
 1   1  28  97.61 3.44 2.90   0   2.32
...
10.002556 sec
```

在此範例中，vCPU 21 和 28 均以最大 Turbo Boost 頻率運作，因為其他核心已進入 C6 休眠狀態以節省功耗，並讓工作中核心取得能力與散熱餘裕。vCPU 3 和 10 (個別與 vCPU 21 和 28 共享處理器核心) 處於等待指令的 C1 狀態。

在下列範例中，所有 18 個核心均運作中執行工作，因此沒有達到最大 Turbo Boost 的餘裕，但它們均以「All Core Turbo Boost」運作，速度達到 3.2 GHz。

```
[ec2-user ~]$ sudo turbostat stress -c 36 -t 10
stress: info: [30685] dispatching hogs: 36 cpu, 0 io, 0 vm, 0 hdd
stress: info: [30685] successful run completed in 10s
pk cor CPU    %c0  GHz  TSC SMI    %c1    %c3    %c6    %c7   %pc2   %pc3   %pc6   %pc7
  Pkg_W RAM_W PKG_% RAM_%
           99.27 3.20 2.90   0   0.26   0.00   0.47   0.00   0.00   0.00   0.00   0.00
 228.59 31.33 199.26  0.00
 0   0   0  99.08 3.20 2.90   0   0.27   0.01   0.64   0.00   0.00   0.00   0.00   0.00
 114.69 18.55 99.32  0.00
 0   0  18  98.74 3.20 2.90   0   0.62
 0   1   1  99.14 3.20 2.90   0   0.09   0.00   0.76   0.00
 0   1  19  98.75 3.20 2.90   0   0.49
 0   2   2  99.07 3.20 2.90   0   0.10   0.02   0.81   0.00
```

```
  0    2  20  98.73 3.20 2.90    0    0.44
  0    3   3  99.02 3.20 2.90    0    0.24    0.00    0.74    0.00
  0    3  21  99.13 3.20 2.90    0    0.13
  0    4   4  99.26 3.20 2.90    0    0.09    0.00    0.65    0.00
  0    4  22  98.68 3.20 2.90    0    0.67
  0    5   5  99.19 3.20 2.90    0    0.08    0.00    0.73    0.00
  0    5  23  98.58 3.20 2.90    0    0.69
  0    6   6  99.01 3.20 2.90    0    0.11    0.00    0.89    0.00
  0    6  24  98.72 3.20 2.90    0    0.39
...
```

限制深層的 C-state 達到高效能與低延遲

C-state 可控制核心在未使用時要進入的休眠等級。建議您控制 C-state 來微調系統的延遲與效能。核心進入休眠狀態需要時間，而且雖然休眠核心讓其他核心更有餘裕加速至更高頻率，但是欲喚醒該休眠核心並開始執行工作也需要時間。例如，若負責處理網路封包中斷的核心處於休眠狀態，則處理此類中斷的服務就可能受到延遲。您可將系統設定為不使用深層 C-state，藉此減少處理器反應延遲，但同時也會壓縮其他核心進行 Turbo Boost 的餘裕。

停用深層休眠狀態常見於 Redis 資料庫應用程式，該應用程式將資料庫存放於系統記憶體中，以實現最快速的查詢回應時間。

在 AL2 上限制更深層的睡眠狀態

1. 使用您選取的編輯器開啟 `/etc/default/grub` 檔案。

   ```
   [ec2-user ~]$ sudo vim /etc/default/grub
   ```

2. 編輯 `GRUB_CMDLINE_LINUX_DEFAULT` 行，並新增 `intel_idle.max_cstate=1` 和 `processor.max_cstate=1` 選項將 C1 設定為閒置核心的最深 C-state。

   ```
   GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 console=ttyS0,115200n8 net.ifnames=0
    biosdevname=0 nvme_core.io_timeout=4294967295 intel_idle.max_cstate=1
    processor.max_cstate=1"
   GRUB_TIMEOUT=0
   ```

   `intel_idle.max_cstate=1` 選項為 Intel 型執行個體設定 C-state 限制，而 `processor.max_cstate=1` 選項則為 AMD 型執行個體設定 C-state 限制。將這兩個選項加入您的組態中是安全的做法。這允許單一組態在 Intel 和 AMD 上設定所需的行為。

3. 儲存檔案並結束您的編輯器。

4. 執行下列命令來重建開機組態。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

5. 重新啟動執行個體以啟用新的核心選項。

```
[ec2-user ~]$ sudo reboot
```

限制 Amazon Linux AMI 上的深層休眠狀態

1. 使用您選取的編輯器開啟 /boot/grub/grub.conf 檔案。

```
[ec2-user ~]$ sudo vim /boot/grub/grub.conf
```

2. 編輯第一個項目的 kernel 行，並新增 intel_idle.max_cstate=1 和 processor.max_cstate=1 選項將 C1 設定為閒置核心的最深 C-state。

```
# created by imagebuilder
default=0
timeout=1
hiddenmenu

title Amazon Linux 2014.09 (3.14.26-24.46.amzn1.x86_64)
root (hd0,0)
kernel /boot/vmlinuz-3.14.26-24.46.amzn1.x86_64 root=LABEL=/ console=ttyS0
 intel_idle.max_cstate=1  processor.max_cstate=1
initrd /boot/initramfs-3.14.26-24.46.amzn1.x86_64.img
```

   intel_idle.max_cstate=1 選項為 Intel 型執行個體設定 C-state 限制，而 processor.max_cstate=1 選項則為 AMD 型執行個體設定 C-state 限制。將這兩個選項加入您的組態中是安全的做法。這允許單一組態在 Intel 和 AMD 上設定所需的行為。

3. 儲存檔案並結束您的編輯器。
4. 重新啟動執行個體以啟用新的核心選項。

```
[ec2-user ~]$ sudo reboot
```

下列範例以 c4.8xlarge 執行個體為例，其中具備兩個執行工作並以「All Core Turbo Boost」核心頻率運作的運作中核心。

```
[ec2-user ~]$ sudo turbostat stress -c 2 -t 10
stress: info: [5322] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
stress: info: [5322] successful run completed in 10s
pk cor CPU    %c0  GHz  TSC SMI    %c1    %c3    %c6    %c7   %pc2   %pc3   %pc6   %pc7
  Pkg_W RAM_W PKG_% RAM_%
            5.56 3.20 2.90   0  94.44   0.00   0.00   0.00   0.00   0.00   0.00   0.00
 131.90 31.11 199.47  0.00
 0   0   0   0.03 2.08 2.90   0  99.97   0.00   0.00   0.00   0.00   0.00   0.00   0.00
  67.23 17.11 99.76  0.00
 0   0  18   0.01 1.93 2.90   0  99.99
 0   1   1   0.02 1.96 2.90   0  99.98   0.00   0.00   0.00
 0   1  19  99.70 3.20 2.90   0   0.30
...
 1   1  10   0.02 1.97 2.90   0  99.98   0.00   0.00   0.00
 1   1  28  99.67 3.20 2.90   0   0.33
 1   2  11   0.04 2.63 2.90   0  99.96   0.00   0.00   0.00
 1   2  29   0.02 2.11 2.90   0  99.98
...
```

在此範例中，vCPU 19 和 28 的核心以 3.2 GHz 運作，其他核心則處於等待指令的 C1 C-state。雖然運作中核心並未達到最大 Turbo Boost 頻率，但是未使用的核心將比處於深層 C6 C-state 更能夠快速回應新的請求。

以最低變異取得基準效能

您可使用 P-state 來減少處理器頻率的變異。P-state 則可控制所需的核心效能 (CPU 頻率)。多數工作負載以 P0 運作的效能較佳，此時會要求使用 Turbo Boost。然而，若要取得穩定效能，而非 Turbo Boost 啟用才出現的突增效能，建議您微調系統。

Intel Advanced Vector Extensions (AVX 或 AVX2) 工作負載能夠以較低頻率展現優異效能，而 AVX 指令也得以使用更多電力。停用 Turbo Boost 讓處理器以較低頻率運作，能夠減少功耗並保持速度穩定。如需有關最佳化 AVX 的執行個體組態與工作負載的詳細資訊，請參閱 Intel 網站。

CPU 閒置驅動程式控制 P-state。較新世代 CPU 需要較新的 CPU 閒置驅動程式 (對應於核心層級)，如下所示：

- Linux 核心 6.1 版和更新版本 – 支援 Intel Granite Rapids （例如 R8i)
- Linux 核心 5.10 版及更新版本 – 支援 AMD Milan （例如 M6a)
- Linux 核心 5.6 版和更新版本 – 支援 Intel Icelake （例如 M6i)

若要偵測執行中的系統核心是否能辨識 CPU，請執行以下命令。

```
if [ -d /sys/devices/system/cpu/cpu0/cpuidle ]; then echo "C-state control enabled";
 else echo "Kernel cpuidle driver does not recognize this CPU generation"; fi
```

如果此命令的輸出表示缺乏支援，我們建議您將核心升級。

本節將說明如何限制深層休眠狀態並停用 Turbo Boost (透過請求 P1 P-state)，為這些類型的工作負載提供低延遲及最小的處理器速度變異。

限制更深層的睡眠狀態，並在 AL2 上停用 Turbo Boost

1. 使用您選取的編輯器開啟 `/etc/default/grub` 檔案。

   ```
   [ec2-user ~]$ sudo vim /etc/default/grub
   ```

2. 編輯 `GRUB_CMDLINE_LINUX_DEFAULT` 行，並新增 `intel_idle.max_cstate=1` 和 `processor.max_cstate=1` 選項將 C1 設定為閒置核心的最深 C-state。

   ```
   GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 console=ttyS0,115200n8 net.ifnames=0
    biosdevname=0 nvme_core.io_timeout=4294967295 intel_idle.max_cstate=1
    processor.max_cstate=1"
   GRUB_TIMEOUT=0
   ```

   `intel_idle.max_cstate=1` 選項為 Intel 型執行個體設定 C-state 限制，而 `processor.max_cstate=1` 選項則為 AMD 型執行個體設定 C-state 限制。將這兩個選項加入您的組態中是安全的做法。這允許單一組態在 Intel 和 AMD 上設定所需的行為。

3. 儲存檔案並結束您的編輯器。
4. 執行下列命令來重建開機組態。

   ```
   [ec2-user ~]$ grub2-mkconfig -o /boot/grub2/grub.cfg
   ```

5. 重新啟動執行個體以啟用新的核心選項。

   ```
   [ec2-user ~]$ sudo reboot
   ```

6. 當您需要 P1 P-state 提供的低處理器速度變異，請執行下列命令以停用 Turbo Boost。

   ```
   [ec2-user ~]$ sudo sh -c "echo 1 > /sys/devices/system/cpu/intel_pstate/no_turbo"
   ```

7. 工作負載完成時，您可使用下列命令重新啟用 Turbo Boost。

```
[ec2-user ~]$ sudo sh -c "echo 0 > /sys/devices/system/cpu/intel_pstate/no_turbo"
```

限制 Amazon Linux AMI 上的深層休眠狀態並停用 Turbo Boost

1. 使用您選取的編輯器開啟 /boot/grub/grub.conf 檔案。

```
[ec2-user ~]$ sudo vim /boot/grub/grub.conf
```

2. 編輯第一個項目的 kernel 行，並新增 intel_idle.max_cstate=1 和 processor.max_cstate=1 選項將 C1 設定為閒置核心的最深 C-state。

```
# created by imagebuilder
default=0
timeout=1
hiddenmenu

title Amazon Linux 2014.09 (3.14.26-24.46.amzn1.x86_64)
root (hd0,0)
kernel /boot/vmlinuz-3.14.26-24.46.amzn1.x86_64 root=LABEL=/ console=ttyS0
 intel_idle.max_cstate=1 processor.max_cstate=1
initrd /boot/initramfs-3.14.26-24.46.amzn1.x86_64.img
```

   intel_idle.max_cstate=1 選項為 Intel 型執行個體設定 C-state 限制，而 processor.max_cstate=1 選項則為 AMD 型執行個體設定 C-state 限制。將這兩個選項加入您的組態中是安全的做法。這允許單一組態在 Intel 和 AMD 上設定所需的行為。

3. 儲存檔案並結束您的編輯器。
4. 重新啟動執行個體以啟用新的核心選項。

```
[ec2-user ~]$ sudo reboot
```

5. 當您需要 P1 P-state 提供的低處理器速度變異，請執行下列命令以停用 Turbo Boost。

```
[ec2-user ~]$ sudo sh -c "echo 1 > /sys/devices/system/cpu/intel_pstate/no_turbo"
```

6. 工作負載完成時，您可使用下列命令重新啟用 Turbo Boost。

```
[ec2-user ~]$ sudo sh -c "echo 0 > /sys/devices/system/cpu/intel_pstate/no_turbo"
```

下列範例以 c4.8xlarge 執行個體為例，其中具備兩個執行工作並以基準核心頻率 (無 Turbo Boost) 運作的 vCPU。

```
[ec2-user ~]$ sudo turbostat stress -c 2 -t 10
stress: info: [5389] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
stress: info: [5389] successful run completed in 10s
pk cor CPU    %c0  GHz  TSC SMI    %c1    %c3    %c6    %c7   %pc2   %pc3   %pc6   %pc7
  Pkg_W RAM_W PKG_% RAM_%
            5.59 2.90 2.90   0  94.41   0.00   0.00   0.00   0.00   0.00   0.00   0.00
 128.48 33.54 200.00  0.00
 0   0   0   0.04 2.90 2.90   0  99.96   0.00   0.00   0.00   0.00   0.00   0.00   0.00
  65.33 19.02 100.00  0.00
 0   0  18   0.04 2.90 2.90   0  99.96
 0   1   1   0.05 2.90 2.90   0  99.95   0.00   0.00   0.00
 0   1  19   0.04 2.90 2.90   0  99.96
 0   2   2   0.04 2.90 2.90   0  99.96   0.00   0.00   0.00
 0   2  20   0.04 2.90 2.90   0  99.96
 0   3   3   0.05 2.90 2.90   0  99.95   0.00   0.00   0.00
 0   3  21  99.95 2.90 2.90   0   0.05
...
 1   1  28  99.92 2.90 2.90   0   0.08
 1   2  11   0.06 2.90 2.90   0  99.94   0.00   0.00   0.00
 1   2  29   0.05 2.90 2.90   0  99.95
```

vCPU 21 和 28 的核心均以基準處理器速度 2.9 GHz 運作中執行工作，而且所有未使用的核心也在 C1 C-state 下同時以基準速度運作並準備好接受指令。

AL2 的 I/O 排程器

I/O 排程器是 Linux 作業系統的一部分，可排序和合併 I/O 請求，並決定它們的處理順序。

I/O 排程器是特別有益於磁性硬碟機這類的設備，其中搜尋時間可能很昂貴，而且最適合於合併主機代管請求。I/O 排程器對固態設備和虛擬化環境的影響較少。這是因為對於固態設備，循序和隨機存取沒有差異，而對於虛擬化環境，主機會提供自己的排程層。

本主題討論 Amazon Linux I/O 排程器。如需其他 Linux 發行版所使用之 I/O 排程器的詳細資訊，請參閱其各自的文件。

主題

支援的排程器

Amazon Linux 支援以下 I/O 排程器：

- deadline —「截止日期」I/O 排程器會排序 I/O 請求，並以最有效率的順序處理它們。它保證每個 I/O 請求的開始時間。它還給與已擱置太久的 I/O 請求更高的優先順序。
- cfq —「完全公平佇列」(CFQ) I/O 排程器會嘗試在程序之間公平地分配 I/O 資源。它會排序 I/O 請求並將其插入至根據程序的佇列中。
- noop —「無操作」(noop) I/O 排程器會將所有 I/O 請求插入至 FIFO 佇列，然後將它們合併成單一請求。此排程器不會執行任何請求排序。

預設排程器

無操作 (noop) 是 Amazon Linux 的預設 I/O 排程器。使用此排程器的原因如下：

- 許多執行個體類型使用虛擬化設備，其中基礎主機為執行個體執行排程。
- 固態設備用於許多執行個體類型中，其中 I/O 排程器的優勢較沒有效果。
- 它是最少侵入性的 I/O 排程器，可視需要進行自訂。

變更排程器

變更 I/O 排程器可以根據排程器導致更多還是更少的 I/O 請求在指定時間內完成，來增加或降低效能。這主要取決於您的工作負載、正在使用的執行個體類型世代，以及所存取的設備類型。若您變更所使用的 I/O 排程器，我們建議您使用工具 (例如 iotop)，來測量 I/O 效能並判斷變更是否對您的使用案例有益。

您可以使用下列命令來檢視設備的 I/O 排程器，此命令會使用 nvme0n1 做為範例。將以下命令中的 nvme0n1 替換為執行個體上 /sys/block 中所列出的裝置。

```
$  cat /sys/block/nvme0n1/queue/scheduler
```

若要設定設備的 I/O 排程器，請使用下列命令。

```
$  echo cfq|deadline|noop > /sys/block/nvme0n1/queue/scheduler
```

例如，若要將 *xvda* 裝置的輸入/輸出排程器從 noop 設定為 cfq，請使用下列命令。

```
$  echo cfq > /sys/block/xvda/queue/scheduler
```

變更 AL2 執行個體的主機名稱

當您在私有 VPC 中啟動執行個體時，Amazon EC2 會指派客體作業系統主機名稱。Amazon EC2 指派的主機名稱類型取決於您的子網路設定。如需 EC2 主機名稱的詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Amazon EC2 執行個體主機名稱類型。 Amazon EC2

設定為搭配使用 IP 型命名與 IPv4 地址之 EC2 執行個體的典型 Amazon EC2 私有 DNS 名稱，看起來會如同這樣：ip-12-34-56-78.us-west-2.compute.internal，其中名稱由內部網域、服務 (在此案例中為 compute)、區域，以及私有 IPv4 地址形式所組成。當您登入執行個體時，shell 提示中會顯示此主機名稱的一部分 (例如，ip-12-34-56-78)。每一次當您停止並重新啟動 Amazon EC2 執行個體時 (除非您使用彈性 IP 地址)，公有 IPv4 地址便會變更，您的公有 DNS 名稱、系統主機名稱和 shell 提示也都會變更。

> **Important**
>
> 此資訊適用於 Amazon Linux。如需其他分發的詳細資訊，請參閱其特定文件。

變更系統主機名稱

若您已為您執行個體的 IP 地址註冊公有 DNS 名稱 (例如 webserver.mydomain.com)，您可以設定系統主機名稱，使您的執行個體將其自身識別為該網域的一部分。這也會變更 shell 提示，使其顯示此名稱的第一個部分，而不是由 提供的主機名稱 AWS （例如，ip-12-34-56-78)。若您沒有已註冊的公有 DNS 名稱，您仍然可以變更主機名稱，只是程序會有些不同。

為了讓您的主機名稱更新持續，必須確認 preserve_hostname 雲端初始化設定已設定為 true。您可以執行下列命令來編輯或新增此設定：

```
sudo vi /etc/cloud/cloud.cfg
```

如果未列出 preserve_hostname 設定，請將下列文字行新增至檔案結尾：

```
preserve_hostname: true
```

將系統主機名稱變更為公有 DNS 名稱

若您有已註冊的公有 DNS 名稱，請遵循此程序。

1. • 對於 AL2：使用 hostnamectl命令來設定主機名稱，以反映完整網域名稱 （ 例如 **webserver.mydomain.com**)。

   ```
   [ec2-user ~]$ sudo hostnamectl set-hostname webserver.mydomain.com
   ```

   • 若為 Amazon Linux AMI：在您的執行個體上，在您喜愛的文字編輯器中開啟 /etc/sysconfig/network 組態檔案，然後變更 HOSTNAME 項目，使其反映完整的網域名稱 (例如 **webserver.mydomain.com**)。

   ```
   HOSTNAME=webserver.mydomain.com
   ```

2. 重新開機執行個體來套用新的主機名稱。

   ```
   [ec2-user ~]$ sudo reboot
   ```

   或者，您可以使用 Amazon EC2 主控台 (在 Instances (執行個體) 頁面上，選取執行個體，並選取 Instance state (執行個體狀態)、Reboot instance (重新啟動執行個體))，以重新開機。

3. 登入您的執行個體，確認主機名稱已更新。您的提示現在應該會顯示新的主機名稱 (最多顯示到第一個 ".")，而 hostname 命令應該會顯示完整的網域名稱。

   ```
   [ec2-user@webserver ~]$ hostname
   webserver.mydomain.com
   ```

不使用公有 DNS 名稱變更系統主機名稱

1. • 對於 AL2：使用 hostnamectl命令來設定主機名稱，以反映所需的系統主機名稱 （ 例如 **webserver**)。

   ```
   [ec2-user ~]$ sudo hostnamectl set-hostname webserver.localdomain
   ```

- 如為 Amazon Linux AMI：在您的執行個體上，使用您偏好的文字編輯器開啟 `/etc/sysconfig/network` 組態檔案，然後變更 `HOSTNAME` 項目，使其反映所需系統主機名稱 (例如 **webserver**)。

```
HOSTNAME=webserver.localdomain
```

2. 在您喜愛的文字編輯器中開啟 `/etc/hosts` 檔案，然後變更開頭為 **127.0.0.1** 的項目，使其反映以下的範例，並使用您自己的主機名稱取代。

```
127.0.0.1 webserver.localdomain webserver localhost4 localhost4.localdomain4
```

3. 重新開機執行個體來套用新的主機名稱。

```
[ec2-user ~]$ sudo reboot
```

或者，您可以使用 Amazon EC2 主控台 (在 Instances (執行個體) 頁面上，選取執行個體，並選取 Instance state (執行個體狀態)、Reboot instance (重新啟動執行個體))，以重新開機。

4. 登入您的執行個體，確認主機名稱已更新。您的提示現在應該會顯示新的主機名稱 (最多顯示到第一個 ".")，而 hostname 命令應該會顯示完整的網域名稱。

```
[ec2-user@webserver ~]$ hostname
webserver.localdomain
```

您也可以實作更多程式設計解決方案，例如指定使用者資料來設定執行個體。如果您的執行個體是 Auto Scaling 群組的一部分，您可以使用 lifecycle hook 來定義使用者資料。如需詳細資訊，請參閱《AWS CloudFormation 使用者指南》中的啟動時在 Linux 執行個體上執行命令和執行個體啟動時的 lifecycle hook。

變更 Shell 提示而不影響主機名稱

如果您不想修改執行個體的主機名稱，但想要顯示比 提供的私有名稱更有用的系統名稱 （ 例如 **webserver**) AWS （ 例如 `ip-12-34-56-78`)，您可以編輯 shell 提示組態檔案以顯示您的系統別名，而不是主機名稱。

將 shell 提示變更為主機別名

1. 在 `/etc/profile.d` 中建立一個檔案，並將稱為 `NICKNAME` 的環境變數設為您希望在 shell 提示中顯示的值。例如，若要將系統別名設為 **webserver**，請執行以下命令。

```
[ec2-user ~]$ sudo sh -c 'echo "export NICKNAME=webserver" > /etc/profile.d/
prompt.sh'
```

2. 在您慣用的文字編輯器中 (例如 /etc/bashrc 或 /etc/bash.bashrc) 開啟 vim (Red Hat) 或 nano (Debian/Ubuntu) 檔案。因為 sudo 和 /etc/bashrc 的擁有者是 /etc/bash.bashrc，所以您需要搭配編輯器命令使用 root。
3. 編輯檔案，將 shell 提示變數 (PS1) 變更為顯示您的別名，而非主機名稱。尋找以下在 /etc/bashrc 或 /etc/bash.bashrc 中設定 shell 提示的文字 (以下也會顯示上下幾行文字以供了解上下內容；請尋找開頭為 ["$PS1" 的文字)：

   ```
   # Turn on checkwinsize
   shopt -s checkwinsize
   [ "$PS1" = "\\s-\\v\\$ " ] && PS1="[\u@\h \W]\\$ "
   # You might want to have e.g. tty in prompt (e.g. more virtual machines)
   # and console windows
   ```

   將該行中的 \h (hostname 的符號) 變更為 NICKNAME 變數的值。

   ```
   # Turn on checkwinsize
   shopt -s checkwinsize
   [ "$PS1" = "\\s-\\v\\$ " ] && PS1="[\u@$NICKNAME \W]\\$ "
   # You might want to have e.g. tty in prompt (e.g. more virtual machines)
   # and console windows
   ```

4. (選用) 若要將 shell 視窗的標題設為新的別名，請完成以下步驟。

   a. 建立名為 /etc/sysconfig/bash-prompt-xterm 的檔案。

      ```
      [ec2-user ~]$ sudo touch /etc/sysconfig/bash-prompt-xterm
      ```

   b. 使用以下命令將檔案轉換成可執行檔。

      ```
      [ec2-user ~]$ sudo chmod +x /etc/sysconfig/bash-prompt-xterm
      ```

   c. 在您喜愛的文字編輯器 (例如 /etc/sysconfig/bash-prompt-xterm 或 vim) 中開啟 nano 檔案。您需要使用 sudo 搭配編輯器命令，因為 /etc/sysconfig/bash-prompt-xterm 的擁有者是 root。

   d. 為檔案新增下行。

```
echo -ne "\033]0;${USER}@${NICKNAME}:${PWD/#$HOME/~}\007"
```

5. 登出然後重新登入，以取得新的別名值。

在其他 Linux 發行版本上變更主機名稱

本頁面的程序僅適用於 Amazon Linux。如需其他 Linux 分佈的詳細資訊，請參閱其特定文件和以下文章：

- 如何將靜態主機名稱指派給執行 RHEL 7 或 Centos 7 的私有 Amazon EC2 執行個體？

在 AL2 執行個體上設定動態 DNS

當您啟動 EC2 執行個體時，它會獲得指派一個公有 IP 地址和公有網域名稱系統 (DNS) 名稱，讓您可以用來在網際網路上存取它。因為 Amazon Web Services 網域中有許多主機，為保持其唯一性，這些公有名稱可能會變得相當長。典型的 Amazon EC2 公有 DNS 名稱看起來像這樣：`ec2-12-34-56-78.us-west-2.compute.amazonaws.com`，其中名稱由 Amazon Web Services 網域、服務 （ 在此案例中為 `compute`) AWS 區域、 和公有 IP 地址的形式組成。

動態 DNS 服務可在其網域區域內提供自訂 DNS 主機名稱，不但易記，也能和您的主機使用案例更為相關。其中有些服務也是免費的。您可以搭配 Amazon EC2 使用動態 DNS 提供者，設定執行個體在每次執行個體啟動時更新與公有 DNS 名稱相關聯的 IP 地址。您有許多可以選擇的不同提供者，至於選擇提供者和使用提供者來註冊名稱的特定詳細資訊，則在此指南的範圍之外。

搭配 Amazon EC2 使用動態 DNS

1. 註冊動態 DNS 服務提供者，並使用其服務註冊一個公有 DNS 名稱。這項程序使用來自 noip.com/free 的免費服務做為範例。
2. 設定動態 DNS 更新用戶端。在您具備動態 DNS 服務提供者和使用其服務註冊的公有 DNS 名稱後，請將 DNS 名稱指向您執行個體的 IP 地址。許多提供者 (包括 noip.com) 允許您在其網站上的帳戶頁面手動執行此作業，但許多提供者也支援軟體更新用戶端。若更新用戶端正在您的 EC2 執行個體上執行，則您的動態 DNS 記錄便會在每一次 IP 地址變更時更新 (發生在關機並重新啟動之後)。在此範例中，您安裝 noip2 用戶端，該用戶端可和 noip.com 提供的服務搭配使用。

   a. 啟用適用於 Enterprise Linux (EPEL) 的額外套件儲存庫，以取得`noip2`用戶端的存取權。

> **Note**
>
> AL2 執行個體預設會安裝 EPEL 儲存庫的 GPG 金鑰和儲存庫資訊。如需詳細資訊，以及下載此套件的最新版本，請參閱 https：//https://fedoraproject.org/wiki/EPEL。

```
[ec2-user ~]$ sudo amazon-linux-extras install epel -y
```

b. 安裝 noip 套裝服務。

```
[ec2-user ~]$ sudo yum install -y noip
```

c. 建立組態檔案。在收到提示時，輸入登入及密碼資訊，並回答後續的問題以設定用戶端。

```
[ec2-user ~]$ sudo noip2 -C
```

3. 啟用 noip 服務。

```
[ec2-user ~]$ sudo systemctl enable noip.service
```

4. 啟動 noip 服務。

```
[ec2-user ~]$ sudo systemctl start noip.service
```

此命令會啟動用戶端，接著用戶端便會讀取您先前建立的組態檔案 (`/etc/no-ip2.conf`)，更新您選擇之公有 DNS 名稱的 IP 地址。

5. 確認更新用戶端已為您的動態 DNS 名稱設定正確的 IP 地址。DNS 記錄更新需要數分鐘的時間，接著您便可以嘗試搭配您在此程序中設定的公有 DNS 名稱，使用 SSH 連線到您的執行個體。

使用 ec2-net-utils for AL2 設定您的網路介面

Amazon Linux 2 AMIs 可能包含由 安裝的其他指令碼 AWS，稱為 ec2-net-utils。這些指令碼會選擇性自動化您網路介面的組態。這些指令碼僅適用於 AL2。

> **Note**
>
> 對於 Amazon Linux 2023， `amazon-ec2-net-utils`套件會在 `/run/systemd/network`目錄中產生介面特定的組態。如需詳細資訊，請參閱 Amazon Linux 2023 使用者指南中的聯網服務。

如果套件尚未安裝，請使用下列命令在 AL2 上安裝套件，或者如果套件已安裝且有其他更新可用，請使用它進行更新：

```
$ yum install ec2-net-utils
```

下列元件屬於 ec2-net-utils：

udev 規則 (`/etc/udev/rules.d`)

當網路介面連接、分離或重新連接到執行中的執行個體時識別它們，確保 hotplug 指令碼執行 (`53-ec2-network-interfaces.rules`)。將 MAC 地址映射到裝置名稱 (`75-persistent-net-generator.rules`，這會產生 `70-persistent-net.rules`)。

hotplug 指令碼

產生適合搭配 DHCP 使用的介面組態檔 (`/etc/sysconfig/network-scripts/ifcfg-ethN`)。也產生路由組態檔案 (`/etc/sysconfig/network-scripts/route-ethN`)。

DHCP 指令碼

每當網路介面收到新的 DHCP 租賃，此指令碼就會查詢彈性 IP 地址的執行個體中繼資料。它會為每一個彈性 IP 地址在路由政策資料庫中新增規則，以確保該地址使用正確的網路介面外送流量。它還會將每個私有 IP 地址新增至網路介面做為輔助地址。

ec2ifup ethN (`/usr/sbin/`)

擴充標準 ifup 的功能。在此指令碼重新撰寫組態檔 `ifcfg-ethN` 和 `route-ethN` 之後，它會執行 ifup。

ec2ifdown ethN (`/usr/sbin/`)

擴充標準 ifdown 的功能。在此指令碼從路由政策資料庫中移除網路介面的所有規則後，它會執行 ifdown。

ec2ifscan (`/usr/sbin/`)

檢查是否有尚未設定的網路介面並加以設定。

ec2-net-utils 初始版本不提供此指令碼。

使用下列命令列出 ec2-net-utils 產生的所有組態檔：

```
$ ls -l /etc/sysconfig/network-scripts/*-eth?
```

若要停用自動化，您可將 EC2SYNC=no 新增至對應的 ifcfg-ethN 檔案。例如，使用以下命令停用 eth1 執行個體的自動化：

```
$ sed -i -e 's/^EC2SYNC=yes/EC2SYNC=no/' /etc/sysconfig/network-scripts/ifcfg-eth1
```

若要完全停用自動化，您可使用以下命令移除套件：

```
$ yum remove ec2-net-utils
```

使用者提供的核心

如果您的 Amazon EC2 執行個體需要使用自訂核心，您可從近似所需的 AMI 開始，在您的執行個體上編譯自訂核心，然後更新開機載入程式以指向新核心。此程序會隨您 AMI 使用的虛擬化類型而異。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Linux AMI 虛擬化類型。

目錄

HVM AMIs (GRUB)

HVM 執行個體磁碟區視為真正的實體磁碟。此開機載入程式與有已分割磁碟和開機載入程式的裸機作業系統類似，可與目前支援的所有 Linux 發行版本一起工作。最常見的開機載入程式是 GRUB 或 GRUB2。

根據預設，GRUB 不會將其輸出傳送至執行個體主控台，因為它會造成額外的開機延遲。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的執行個體主控台輸出。如果您要安裝自訂核心，請考慮啟用 GRUB 輸出。

您不需要指定備用核心，但我們建議您在測試新核心時備妥備用核心。當新的核心失敗時，GRUB 可備援其他核心。擁有備用核心，即使找不到新核心，執行個體也能開機。

舊版 GRUB for Amazon Linux 會使用 `/boot/grub/menu.lst`。適用於 AL2 的 GRUB2 使用 `/etc/default/grub`。 AL2 如需更新開機載入程式中預設核心的詳細資訊，請參閱 Linux 發行版的說明文件。

全虛擬化 AMIs (PV-GRUB)

使用全虛擬化 (PV) 虛擬化的 AMIs 會在開機程序期間使用稱為 PV-GRUB 的系統。PV-GRUB 是全虛擬化的開機載入器，執行修補過的 GNU GRUB 0.97 版本。當您啟動執行個體時，PV-GRUB 會啟動開機程序，然後鏈會載入您影像之 `menu.lst` 檔案指定的核心。

PV-GRUB 了解標準的 `grub.conf` 或 `menu.lst` 命令，能與目前支援的所有 Linux 發行版本一起工作。Ubuntu 10.04 LTS、Oracle Enterprise Linux 或 CentOS 5.x 等舊版的發行版本需要特殊的「ec2」或「xen」核心套件，而較新的發行版本則會在預設核心套件中包含所需之驅動程式。

大部分的現代全虛擬化 AMI 預設使用 PV-GRUB AKI (包括 Amazon EC2 Launch Wizard Quick Start 選單中所有可用的全虛擬化 Linux AMI)，所以您不需要採取額外步驟即可在您的執行個體上使用不同的核心，假設您要使用的核心與您的發行版本相容。在您的執行個體上執行自訂核心的最佳方式，是從近似您所需的 AMI 開始，然後在您的執行個體上編譯自訂核心並修改 `menu.lst` 檔案，以該核心開機。

您可以驗證 AMI 的核心映像是否為 PV-GRUB AKI。執行以下 describe-images 命令 (替換您的內核映像 ID) 並檢查該 `Name` 欄位是否以 `pv-grub` 開頭：

```
aws ec2 describe-images --filters Name=image-id,Values=aki-880531cd
```

內容

PV-GRUB 的限制

PV-GRUB 具有下列限制：

- 您不能使用 64 位元版的 PV-GRUB 啟動 32 位元的核心，反之亦然。
- 使用 PV-GRUB AKI 時，您不能指定 Amazon ramdisk image (ARI)。

- AWS 已測試並確認 PV-GRUB 可搭配下列檔案系統格式使用：EXT2, EXT3, EXT4、JFS、XFS 和 ReiserFS。其他檔案系統格式可能不行。
- PV-GRUB 可以啟動使用 gzip、bzip2、lzo 和 xz 壓縮格式壓縮的核心。
- 叢集 AMI 不支援或需要 PV-GRUB，因為它們使用全硬體虛擬化 (HVM)。雖然全虛擬化執行個體使用 PV-GRUB 開機，但 HVM 執行個體磁碟區視為真正的磁碟，而開機程序類似有已分割磁碟和開機載入器之裸機作業系統的開機程序。
- PV-GRUB 1.03 版和舊版不支援 GPT 分割；它們只支援 MBR 分割。
- 您若打算對 Amazon Elastic Block Store (Amazon EBS) 磁碟區使用邏輯磁碟區管理工具 (LVM)，您需要在 LVM 之外另有開機分割。然後才能利用 LVM 建立邏輯磁碟區。

設定虛擬虛擬的 GRUB AMIs

若要開機 PV-GRUB，GRUB `menu.lst` 檔案必須存在於影像中，此檔案最常見的位置為 `/boot/grub/menu.lst`。

下列是使用 PV-GRUB AKI 啟動 AMI 的 `menu.lst` 組態檔範例。在本例中，有兩個核心項目可供選擇：Amazon Linux 2018.03 (此 AMI 的原始核心) 和 Vanilla Linux 4.16.4 (新版的 Vanilla Linux 核心，來自 https://www.kernel.org/)。Vanilla 項目是由此 AMI 的原始項目複製而來，而 `kernel` 和 `initrd` 路徑已更新至新位置。`default 0` 參數會將開機載入器指向它發現的第一個項目 (本例中為 Vanilla 項目)，如果第一個項目開機發生問題，`fallback 1` 參數會將開機載入器指向下一個項目。

```
default 0
fallback 1
timeout 0
hiddenmenu

title Vanilla Linux 4.16.4
root (hd0)
kernel /boot/vmlinuz-4.16.4 root=LABEL=/ console=hvc0
initrd /boot/initrd.img-4.16.4

title Amazon Linux 2018.03 (4.14.26-46.32.amzn1.x86_64)
root (hd0)
kernel /boot/vmlinuz-4.14.26-46.32.amzn1.x86_64 root=LABEL=/ console=hvc0
initrd /boot/initramfs-4.14.26-46.32.amzn1.x86_64.img
```

您不需要在 `menu.lst` 檔案中指定備用核心，但我們建議您在測試新核心時備妥備用核心。當新的核心失敗時，PV-GRUB 可備援其他核心。擁有備用核心，即使找不到新核心，執行個體也能開機。

PV-GRUB 會使用它找到的第一個核心來檢查下列 `menu.lst` 的位置：

- `(hd0)/boot/grub`
- `(hd0,0)/boot/grub`
- `(hd0,0)/grub`
- `(hd0,1)/boot/grub`
- `(hd0,1)/grub`
- `(hd0,2)/boot/grub`
- `(hd0,2)/grub`
- `(hd0,3)/boot/grub`
- `(hd0,3)/grub`

請注意，PV-GRUB 1.03 及舊版只檢查本清單前兩個位置的其中之一。

Amazon PV-GRUB 核心影像 ID

所有 Amazon EC2 區域皆提供 PV-GRUB AKI，亞太區域 (大阪) 除外。32 位元和 64 位元架構類型都有 AKI。大部分的現代 AMI 都預設使用 PV-GRUB AKI。

建議您一律使用最新版的 PV-GRUB AKI，因為不是所有的 PV-GRUB AKI 版本都與所有執行個體類型相容。使用下列 describe-images 命令取得目前區域的 PV-GRUB AKI 清單：

```
aws ec2 describe-images --owners amazon --filters Name=name,Values=pv-grub-*.gz
```

請注意，PV-GRUB 是 `ap-southeast-2` 區域唯一提供的 AKI。您應該確認您想要複製到此區域的任何 AMI 是否使用此區域提供的 PV-GRUB 版本。

下列各區域目前的 AKI ID：使用 hd0 AKI 註冊新的 AMI。

> **Note**
>
> 我們在過去提供 hd00 AKI 的區域仍繼續提供此 AKI 以保證回溯相容性。

ap-northeast-1，亞太區域 (東京)

映像 ID	映像名稱
aki-f975a998	pv-grub-hd0_1.05-i386.gz
aki-7077ab11	pv-grub-hd0_1.05-x86_64.gz

ap-southeast-1，亞太區域 (新加坡) 區域

映像 ID	映像名稱
aki-17a40074	pv-grub-hd0_1.05-i386.gz
aki-73a50110	pv-grub-hd0_1.05-x86_64.gz

ap-southeast-2，亞太區域 (雪梨)

映像 ID	映像名稱
aki-ba5665d9	pv-grub-hd0_1.05-i386.gz
aki-66506305	pv-grub-hd0_1.05-x86_64.gz

eu-central-1，歐洲 (法蘭克福)

映像 ID	映像名稱
aki-1419e57b	pv-grub-hd0_1.05-i386.gz
aki-931fe3fc	pv-grub-hd0_1.05-x86_64.gz

eu-west-1，歐洲 (愛爾蘭)

映像 ID	映像名稱
aki-1c9fd86f	pv-grub-hd0_1.05-i386.gz

映像 ID	映像名稱
aki-dc9ed9af	pv-grub-hd0_1.05-x86_64.gz

sa-east-1，南美洲 (聖保羅)

映像 ID	映像名稱
aki-7cd34110	pv-grub-hd0_1.05-i386.gz
aki-912fbcfd	pv-grub-hd0_1.05-x86_64.gz

us-east-1，US East (N. Virginia)

映像 ID	映像名稱
aki-04206613	pv-grub-hd0_1.05-i386.gz
aki-5c21674b	pv-grub-hd0_1.05-x86_64.gz

us-gov-west-1， AWS GovCloud （美國西部）

映像 ID	映像名稱
aki-5ee9573f	pv-grub-hd0_1.05-i386.gz
aki-9ee55bff	pv-grub-hd0_1.05-x86_64.gz

us-west-1，美國西部 (加州北部)

映像 ID	映像名稱
aki-43cf8123	pv-grub-hd0_1.05-i386.gz
aki-59cc8239	pv-grub-hd0_1.05-x86_64.gz

us-west-2，美國西部 (奧勒岡)

映像 ID	映像名稱
aki-7a69931a	pv-grub-hd0_1.05-i386.gz
aki-70cb0e10	pv-grub-hd0_1.05-x86_64.gz

更新 PV-GRUB

建議您一律使用最新版的 PV-GRUB AKI，因為不是所有的 PV-GRUB AKI 版本都與所有執行個體類型相容。而且，並非所有區域都提供較舊的 PV-GRUB 版本；所以，如果您將使用較舊版本的 AMI 複製到不支援該版本的區域，在您更新核心映像前，都將無法開機從該 AMI 啟動的執行個體。使用下列程序檢查您執行個體的 PV-GRUB 版本，並在有需要時更新它。

檢查您的 PV-GRUB 版本

1. 尋找您執行個體的核心 ID。

```
aws ec2 describe-instance-attribute --instance-id instance_id --attribute kernel --
region region

{
    "InstanceId": "instance_id",
    "KernelId": "aki-70cb0e10"
}
```

   此執行個體的核心 ID 是 aki-70cb0e10。

2. 檢視該核心 ID 的版本資訊。

```
aws ec2 describe-images --image-ids aki-70cb0e10 --region region

{
    "Images": [
        {
            "VirtualizationType": "paravirtual",
            "Name": "pv-grub-hd0_1.05-x86_64.gz",
            ...
            "Description": "PV-GRUB release 1.05, 64-bit"
        }
```

```
    ]
}
```

此核心影像是 PV-GRUB 1.05。如果您的 PV-GRUB 版本不是最新版 (如 Amazon PV-GRUB 核心影像 ID 中所示)，您應該使用下列程序更新它。

更新您的 PV-GRUB 版本

如果您的執行個體使用較舊的 PV-GRUB 版本，您應該將它更新至最新版本。

1. 從 Amazon PV-GRUB 核心影像 ID 中找出適用於您區域和處理器架構的最新 PV-GRUB AKI。
2. 停止您的執行個體。您的執行個體必須停止才能修改所用的核心影像。

```
aws ec2 stop-instances --instance-ids instance_id --region region
```

3. 修改用於您執行個體的核心影像。

```
aws ec2 modify-instance-attribute --instance-id instance_id --kernel kernel_id --
region region
```

4. 重新啟動您的執行個體。

```
aws ec2 start-instances --instance-ids instance_id --region region
```

AL2 AMI 版本通知

若要在有新的 Amazon Linux AMI 發佈時收到通知，您可以使用 Amazon SNS 訂閱。

如需訂閱 AL2023 通知的相關資訊，請參閱《Amazon Linux 2023 使用者指南》中的接收新更新的通知。

> **Note**
>
> AL1 的標準支援已於 2020 年 12 月 31 日結束。AL1 維護支援階段已於 2023 年 12 月 31 日結束。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 上的更新 end-of-life。

訂閱 Amazon Linux 通知

1. 在 https://console.aws.amazon.com/sns/v3/home 開啟 Amazon SNS 主控台。
2. 如有必要，請在導覽列中將「區域」變更為美國東部 (維吉尼亞北部)。您必須選取區域，您要訂閱的 SNS 通知已在該區域中建立完成。
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. 【AL2】 對於主題 ARN，複製並貼上下列 Amazon Resource Name (ARN)：**arn:aws:sns:us-east-1:137112412989:amazon-linux-2-ami-updates**。
   b. [Amazon Linux] 對於 Topic ARN (主題 ARN)，請複製下列 Amazon Resource Name (ARN)：**arn:aws:sns:us-east-1:137112412989:amazon-linux-ami-updates**。
   c. 對於 通訊協定 ，選擇 電子郵件 。
   d. 針對 Endpoint (端點)，請輸入可用於接收通知的電子郵件地址。
   e. 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。

只要有 AMI 發行，我們就會向對應主題的訂閱者傳送通知。若要停止接收這些通知，請使用下列程序取消訂閱。

取消訂閱 Amazon Linux 通知

1. 在 https://console.aws.amazon.com/sns/v3/home 開啟 Amazon SNS 主控台。
2. 如有必要，請在導覽列中將「區域」變更為美國東部 (維吉尼亞北部)。您必須使用區域，SNS 通知已在該區域中建立完成。
3. 在導覽窗格中選取 Subscriptions (訂閱)，選取訂閱然後選取 Actions (動作)，Delete subscriptions (刪除訂閱)。
4. 出現確認提示時，請選擇 Delete (刪除)。

Amazon Linux AMI SNS 訊息格式

SNS 訊息的結構描述如下所示。

```
{
```

```
    "description": "Validates output from AMI Release SNS message",
    "type": "object",
    "properties": {
        "v1": {
            "type": "object",
            "properties": {
                "ReleaseVersion": {
                    "description": "Major release (ex. 2018.03)",
                    "type": "string"
                },
                "ImageVersion": {
                    "description": "Full release (ex. 2018.03.0.20180412)",
                    "type": "string"
                },
                "ReleaseNotes": {
                    "description": "Human-readable string with extra information",
                    "type": "string"
                },
                "Regions": {
                    "type": "object",
                    "description": "Each key will be a region name (ex. us-east-1)",
                    "additionalProperties": {
                        "type": "array",
                        "items": {
                            "type": "object",
                            "properties": {
                                "Name": {
                                    "description": "AMI Name (ex. amzn-ami-
hvm-2018.03.0.20180412-x86_64-gp2)",
                                    "type": "string"
                                },
                                "ImageId": {
                                    "description": "AMI Name (ex.ami-467ca739)",
                                    "type": "string"
                                }
                            },
                            "required": [
                                "Name",
                                "ImageId"
                            ]
                        }
                    }
                }
            },
```

```
            "required": [
                "ReleaseVersion",
                "ImageVersion",
                "ReleaseNotes",
                "Regions"
            ]
        }
    },
    "required": [
        "v1"
    ]
}
```

設定 AL2 MATE 桌面連線

MATE 桌面環境已在 AMI 中預先安裝並預先設定，並具有下列說明：

".NET Core *x.x*, Mono *x.xx*, PowerShell *x.x*, and MATE DE pre-installed to run your .NET applications on Amazon Linux 2 with Long Term Support (LTS)."

環境提供直覺式圖形使用者介面，以最少使用命令列的方式管理 AL2 執行個體。該介面使用圖形表示，例如圖示、視窗、工具列、資料夾、桌布和桌面小工具。內建的 GUI 式工具可用來執行一般工作。例如，有一些工具可用來新增和移除軟體、套用更新、組織檔案、啟動程式，以及監視系統運作狀態。

Important

xrdp 是 AMI 中搭售的遠程桌面軟體。根據預設，xrdp 會使用自我簽署的 TLS 憑證來加密遠端桌面工作階段。xrdp 維護者 AWS 和 都不建議在生產環境中使用自我簽署憑證。而是從適當的憑證授權機構 (CA) 取得憑證，並將其安裝在您的執行個體上。如需有關 TLS 組態的詳細資訊，請參閱 xrdp wiki 上的 TLS 安全層次。

Note

如果您偏好使用虛擬網路運算 (VNC) 服務而非 xrdp，請參閱如何在執行 AL2Knowledge Center 的 Amazon EC2 執行個體上安裝 GUI AWS 文章。

先決條件

若要執行本主題中顯示的命令，您必須安裝 AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell，並設定您的 AWS 設定檔。

選項

1. 安裝 AWS CLI – 如需詳細資訊，請參閱AWS Command Line Interface 《 使用者指南》中的安裝 AWS CLI和組態基本概念。
2. 安裝 Tools for Windows PowerShell – 如需詳細資訊，請參閱 AWS Tools for PowerShell 使用者指南中的安裝 AWS Tools for Windows PowerShell 和共用憑證。

> **Tip**
>
> 除了完整安裝 之外 AWS CLI，您也可以AWS CloudShell針對直接從 啟動的瀏覽器型預先驗證 Shell 使用 AWS 管理主控台。檢查支援的 AWS 區域，以確保它在您工作的區域可用。

設定 RDP 連線

請依照下列步驟，設定從本機電腦到執行 MATE 桌面環境之 AL2 執行個體的遠端桌面通訊協定 (RDP) 連線。

1. 若要取得 AMI 名稱中包含 MATE 之 AL2 的 AMI ID，您可以從本機命令列工具使用 describe-images 命令。如果您尚未安裝命令列工具，您可以直接從 AWS CloudShell 工作階段執行下列查詢。如需如何從 CloudShell 啟動 Shell 工作階段的詳細資訊，請參閱 AWS CloudShell 入門。從 Amazon EC2 主控台，您可以透過啟動執行個體，然後在 AMI 搜尋列中輸入 MATE，來尋找包含 MATE 的 AMI。預先安裝 MATE 的 AL2 Quick Start 會出現在搜尋結果中。

```
aws ec2 describe-images --filters "Name=name,Values=amzn2*MATE*" --query
 "Images[*].[ImageId,Name,Description]"
[
    [
        "ami-0123example0abc12",
        "amzn2-x86_64-MATEDE_DOTNET-2020.12.04",
        ".NET Core 5.0, Mono 6.12, PowerShell 7.1, and MATE DE pre-installed to run
 your .NET applications on Amazon Linux 2 with Long Term Support (LTS)."
    ],
    [
```

```
        "ami-0456example0def34",
        "amzn2-x86_64-MATEDE_DOTNET-2020.04.14",
        "Amazon Linux 2 with .Net Core, PowerShell, Mono, and MATE Desktop
 Environment"
    ]
]
```

選擇適合您使用的 AMI。

2. 使用您在上一個步驟中找到的 AMI 來啟動 EC2 執行個體。設定安全群組以允許將 TCP 流量傳入到連接埠 3389。如需安全群組的詳細資訊，請參閱 VPC 的安全群組。此組態可讓您使用 RDP 用戶端來連線至執行個體。
3. 使用 SSH 連線至執行個體。
4. 更新執行個體上的軟體和核心。

```
[ec2-user ~]$ sudo yum update
```

更新完成後，將執行個體重新啟動，確認您使用的是更新中最新的套件和程式庫；在重新啟動之前，不會載入核心更新。

```
[ec2-user ~]$ sudo reboot
```

5. 重新連接至執行個體，並在 Linux 執行個體上執行下列命令，以設定 ec2-user 的密碼。

```
[ec2-user ~]$ sudo passwd ec2-user
```

6. 安裝憑證和金鑰。

如果您已經擁有憑證和金鑰，請將它們複製到 /etc/xrdp/ 目錄，如下所示：

- 憑證 – /etc/xrdp/cert.pem
- 金鑰 — /etc/xrdp/key.pem

如果您沒有憑證和金鑰，請使用下列命令在 /etc/xrdp 目錄中產生。

```
$ sudo openssl req -x509 -sha384 -newkey rsa:3072 -nodes -keyout /etc/xrdp/key.pem
 -out /etc/xrdp/cert.pem -days 365
```

> **Note**
>
> 此命令會產生有效時間為 365 天的憑證。

7. 在您要連線到執行個體的電腦上開啟 RDP 用戶端 (例如，執行 Microsoft Windows 的電腦上的遠端桌面連線)。輸入 `ec2-user` 做為使用者名稱，然後輸入您在上一個步驟中設定的密碼。

在 Amazon EC2 執行個體上停用 **`xrdp`**

您可以在 Linux 執行個體上執行下列其中一個命令，以隨時停用 `xrdp`。下列命令不會影響您透過 X11 伺服器使用 MATE 的能力。

```
[ec2-user ~]$ sudo systemctl disable xrdp
```

```
[ec2-user ~]$ sudo systemctl stop xrdp
```

在 Amazon EC2 執行個體上啟用 **`xrdp`**

若要重新啟用，`xrdp`以便您可以連線至執行 MATE 桌面環境的 AL2 執行個體，請在 Linux 執行個體上執行下列其中一個命令。

```
[ec2-user ~]$ sudo systemctl enable xrdp
```

```
[ec2-user ~]$ sudo systemctl start xrdp
```

AL2 教學課程

下列教學課程說明如何使用執行 AL2 的 Amazon EC2 執行個體執行常見任務。如需影片教學課程，請參閱AWS 教學影片和實驗室。

如需 AL2023 說明，請參閱《Amazon Linux 2023 使用者指南》中的教學課程。

教學

- 教學課程：在 AL2 上安裝 LAMP 伺服器
- 教學課程：在 AL2 上設定 SSL/TLS
- 教學課程：在 AL2 上託管 WordPress 部落格

教學課程：在 AL2 上安裝 LAMP 伺服器

下列程序可協助您在 AL2 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**
>
> 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。如需 AL2023，請參閱在 AL2023 上安裝 LAMP 伺服器。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。

選項：使用自動化完成此教學課程

若要使用 AWS Systems Manager 自動化而非下列任務完成本教學課程，請執行 AWS Docs-InstallALAMPServer-AL2 Automation 文件。

任務

步驟 1：準備 LAMP 伺服器

先決條件

- 本教學課程假設您已使用 AL2 啟動新的執行個體，其公有 DNS 名稱可從網際網路存取。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的啟動執行個體。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》中的安全群組規則。
- 下列程序會安裝 AL2 上可用的最新 PHP 版本，目前為 php8.2。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 php8.2 的相容性。

準備 LAMP 伺服器

1. 連線到您的執行個體。
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 yum update -y
   ```
3. 安裝 mariadb10.5 Amazon Linux Extras 儲存庫，以取得 MariaDB 套件的最新版本。

   ```
   [ec2-user ~]$ sudo amazon-linux-extras install mariadb10.5
   ```

   如果出現錯誤，顯示 sudo: amazon-linux-extras: command not found，則代表執行個體不是以 Amazon Linux 2 AMI 啟動 (您可能是使用 Amazon Linux AMI)。您可以使用下列命令來檢視您的 Amazon Linux 版本。

   ```
   cat /etc/system-release
   ```
4. 安裝 php8.2 Amazon Linux Extras 儲存庫以取得最新版本的 AL2 PHP套件。

   ```
   [ec2-user ~]$ sudo amazon-linux-extras install php8.2
   ```
5. 現在您的執行個體是最新版本，可安裝 Apache Web 伺服器、PHP 和 MariaDB 軟體套件。使用 yum install 命令可以同時安裝多個軟體套件和所有相關的依存項。

   ```
   [ec2-user ~]$ sudo yum install -y httpd
   ```

   您可以使用下列命令來檢視這些套件的目前版本：

   ```
   yum info package_name
   ```
6. 啟動 Apache Web 伺服器。

   ```
   [ec2-user ~]$ sudo systemctl start httpd
   ```
7. 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
```

8. 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* 安全群組在初始化期間已針對您的執行個體設定完畢。該群組包含允許 SSH 連線的規則。

   a. 在 https://console.aws.amazon.com/ec2/ 開啟 Amazon EC2 主控台。

   b. 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。

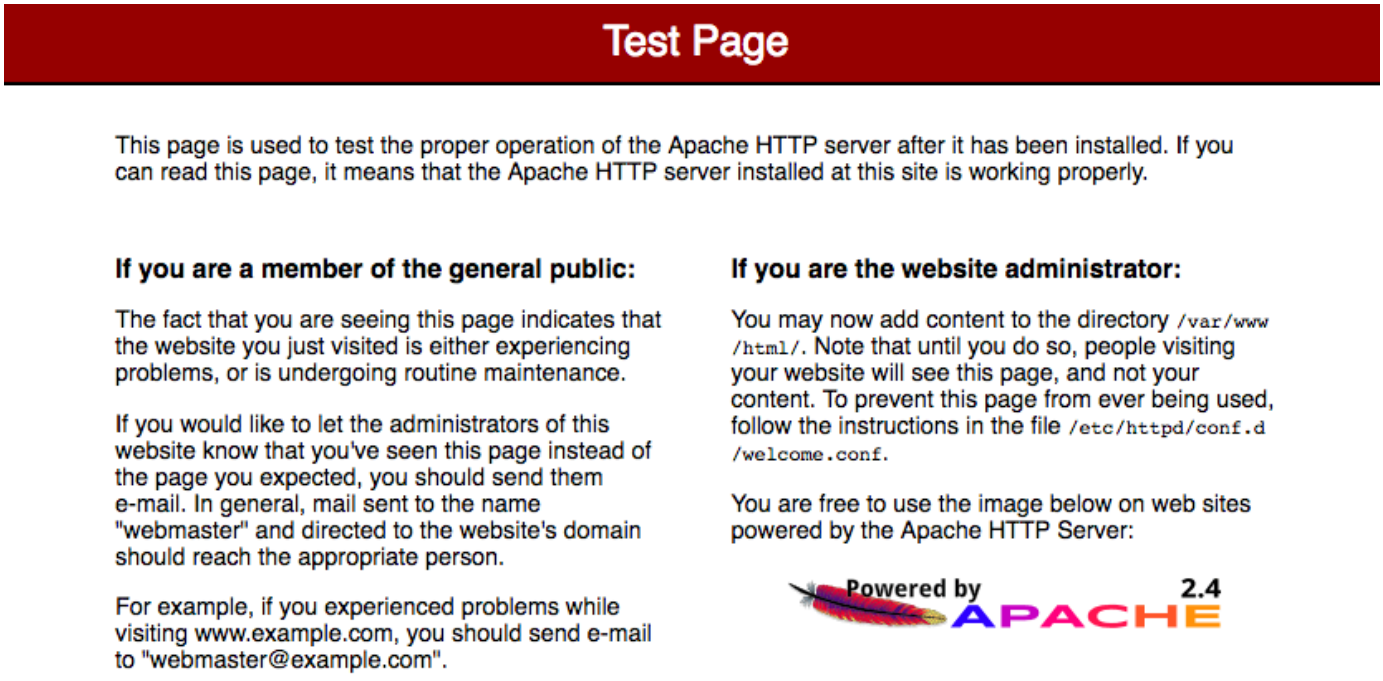
   d. 選擇安全群組的連結。使用將規則新增至安全群組中的程序，新增具有下列值的新傳入安全規則：

   - 類型：HTTP
   - Protocol (通訊協定)：TCP
   - Port Range (連接埠範圍)：80
   - Source (來源)：自訂

9. 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，您應該會看到 Apache 測試頁。您可以使用 Amazon EC2 主控台取得執行個體的公有 DNS (查看 Public DNS (公有 DNS) 欄；如果此欄為隱藏，請選擇 Show/Hide Columns (顯示/隱藏欄) (齒輪狀圖示) 並選擇 Public DNS (公有 DNS))。

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 AL2 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 7.2.0

System	Linux ip-172-31-22-15.us-west-2.compute.internal 4.9.62-10.57.amzn2.x86_64 #1 SMP Wed Dec 6 00:07:49 UTC 2017 x86_64
Build Date	Dec 13 2017 03:34:37
Server API	Apache 2.0 Handler
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/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-exif.ini, /etc/php.d/20-fileinfo.ini, /etc/php.d/20-ftp.ini, /etc/php.d/20-gettext.ini, /etc/php.d/20-iconv.ini, /etc/php.d/20-json.ini, /etc/php.d/20-mysqlnd.ini, /etc/php.d/20-pdo.ini, /etc/php.d/20-phar.ini, /etc/php.d/20-sockets.ini, /etc/php.d/20-sqlite3.ini, /etc/php.d/20-tokenizer.ini, /etc/php.d/30-mysqli.ini, /etc/php.d/30-pdo_mysql.ini, /etc/php.d/30-pdo_sqlite.ini
PHP API	20170718
PHP Extension	20170718
Zend Extension	320170718
Zend Extension Build	API320170718,NTS
PHP Extension Build	API20170718,NTS

   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 yum 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。另請確認在 amazon-linux-extras 命令的輸出中已啟用 php7.2 和 lamp-mariadb10.2-php7.2 extras。

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 AL2 上設定 SSL/TLS。

安裝 phpMyAdmin

1. 安裝所需的依存項目。

```
[ec2-user ~]$ sudo yum 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 AL2 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，請參閱將規則新增至安全群組。

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：

- 使用 將檔案傳輸到您的 Linux 執行個體WinSCP。
- 使用 SCP用戶端將檔案傳輸至 Linux 執行個體。

- 教學課程：在 AL2 上託管 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/
- chmod 命令：https://en.wikipedia.org/wiki/Chmod
- chown 命令：https://en.wikipedia.org/wiki/Chown

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 AL2 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 AL2 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。AL2 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 AL2 搭配使用。我們也假設您使用的是全新 Amazon EC2 執行個體。如果您嘗試設定執行不同分佈的 EC2 執行個體，或執行舊版 AL2 的執行個體，本教學中的某些程序可能無法運作。若為 Ubuntu，請參閱以下社群文件：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 Amazon EBS 支援的 AL2 執行個體。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的啟動執行個體。
- 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》中的安全群組規則。
- 安裝 Apache Web 伺服器。如需step-by-step說明，請參閱教學課程：在 AL2 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

選項：使用自動化完成此教學課程

若要使用 AWS Systems Manager 自動化而非下列任務來完成本教學課程，請執行自動化文件。

此程序會引導您完成使用自我簽署數位憑證在 AL2 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。

   ```
   [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，則可以省略此選項。

   ```
   [ec2-user ~]$ sudo yum update -y
   ```

3. 現在您的執行個體為最新，請安裝 Apache 模組 `mod_ssl` 來新增 TLS 支援。

```
[ec2-user ~]$ sudo yum install -y mod_ssl
```

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。

- `/etc/pki/tls/certs/make-dummy-cert`

  用來為伺服器主機產生自我簽署 X.509 憑證和私有金鑰的指令碼。對於測試 Apache 是否已妥善設定以使用 TLS，本憑證可派上用場。因為不提供身分證明，請勿用在生產環境中。如果用在生產環境中，則會在 Web 瀏覽器中觸發警告。

4. 執行下列指令碼，以產生用於測試的自我簽署虛擬憑證和金鑰。

```
[ec2-user ~]$ cd /etc/pki/tls/certs
sudo ./make-dummy-cert localhost.crt
```

這會在 `localhost.crt` 目錄中產生新檔案 `/etc/pki/tls/certs/`。指定的檔案名稱符合 SSLCertificateFile 中的 `/etc/httpd/conf.d/ssl.conf` 指示詞所指派的預設值。

此檔案包含自我簽署憑證和憑證的私有金鑰。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。

5. 使用偏好的文字編輯器 (例如 vim 或 nano)，以根使用者身分開啟 `/etc/httpd/conf.d/ssl.conf` 檔案並加入下列註解，因為自我簽署的虛擬憑證也包含此金鑰。如果您沒有加入此列註解就完成下列步驟，Apache 服務會無法開始。

```
SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
```

6. 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。

7. 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 AL2 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。
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

名稱	描述	範例
Locality Name (地區名稱)	您組織的位置 (例如城市)。	西雅圖
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 AL2 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 AL2 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。

   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 AL2 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。讓您的 EC2 AL2 執行個體保持up-to-date、留意來自 OpenSSL 的安全公告，並留意技術媒體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 yum 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 AL2。本教學課程僅支援從官方 AL2 AMI 新建立的執行個體。

教學課程：在 AL2 上託管 WordPress 部落格

下列程序可協助您在 AL2 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 AL2 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

本教學課程假設您已依照 中的所有步驟，以 PHP 和資料庫 (MySQL 或 MariaDB) 支援的功能性 Web 伺服器啟動 AL2 執行個體教學課程：在 AL2 上安裝 LAMP 伺服器。本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。

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。

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

選項：使用自動化完成此教學課程

若要使用 AWS Systems Manager 自動化而非下列任務來完成本教學課程，請執行自動化文件。

連接至您的執行個體，然後下載 WordPress 安裝套件。

下載並解壓縮 WordPress 安裝套件

1. 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
```

2. 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. 啟動資料庫伺服器。

   -
```
[ec2-user ~]$ sudo systemctl start mariadb
```

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 保護 MariaDB 伺服器(AL2)。

```
[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。

   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。

   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';
```

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 systemctl stop httpd。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 AL2 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 7.2 版)。

使用下列命令在 AL2 上安裝 PHP 圖形繪製程式庫。例如，如果您從 amazon-linux-extras 安裝 php7.2 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 7.2 版。

```
[ec2-user ~]$ sudo yum install php-gd
```

若要確認已安裝的版本，請使用下列命令：

```
[ec2-user ~]$ sudo yum list installed php-gd
```

下列為範例輸出：

```
php-gd.x86_64                  7.2.30-1.amzn2           @amzn2extra-php7.2
```

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)。如果您尚未這麼做，請套用下列群組成員資格和許可（如 中所述教學課程：在 AL2 上安裝 LAMP 伺服器)。

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 AL2 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 AL2。

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)。

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 使用者指南」中的 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

如需安裝 WordPress 的詳細資訊，請參閱 http://codex.wordpress.org/ 上的 WordPress Codex help documentation。

如需對安裝進行故障診斷的詳細資訊，請參閱常見的安裝問題。

如需讓 WordPress 部落格更安全的資訊，請參閱強化 WordPress。

如需有關將 WordPress 部落格保持在up-to-date的資訊，請參閱更新 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)。有關問題和幾個可能解決方案的更詳細描述，請參閱變更網站 URL。

如果您的 WordPress 安裝發生這種情況，您可以使用下列程序來復原您的部落格，該程序使用 WordPress 的wp-cli命令列界面。

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。

在 Amazon EC2 外部使用 Amazon Linux 2

AL2 容器映像可以在相容的容器執行時間環境中執行。

AL2 也可以在直接在 Amazon EC2 上執行之外以虛擬化訪客身分執行。

> **Note**
>
> AL2 映像的組態與 AL2023 不同。
> 遷移至 AL2023 時，請務必檢閱在 Amazon EC2 外部使用 Amazon Linux 2023，並將您的組態調整為與 AL2023 相容。

在內部部署中以虛擬機器執行 AL2

使用 AL2 虛擬機器 (VM) 映像進行內部部署開發和測試。我們為每個支援的虛擬化平台提供不同的 AL2 VM 映像。您可以在 Amazon Linux 2 虛擬機器映像頁面中檢視支援的平台清單。

若要搭配其中一個支援的虛擬化平台使用 AL2 虛擬機器映像，請執行下列動作：

- 步驟 1：準備 seed.iso 開機映像
- 步驟 2：下載 AL2 VM 映像
- 步驟 3：開機並連接至您的新 VM

步驟 1：準備 `seed.iso` 開機映像

`seed.iso` 開機映像包括 VM 開機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。它不包含 AL2 作業系統檔案。

若要產生 `seed.iso` 開機映像，您需要兩個組態檔案：

- `meta-data` – 此檔案包括 VM 的主機名稱和靜態網路設定。
- `user-data` – 此檔案會設定使用者帳戶，並指定其密碼、金鑰對和存取機制。根據預設，AL2 VM 映像會建立`ec2-user`使用者帳戶。您會使用 `user-data` 組態檔案來設定預設使用者帳戶的密碼。

建立 seed.iso 開機光碟

1. 建立一個名為 seedconfig 的新資料夾並導覽到該資料夾。
2. 建立 meta-data 組態檔案。

   a. 建立名為 meta-data 的新檔案。

   b. 使用您偏好的文字編輯器開啟 meta-data 檔案，然後新增下列命令。

   ```
   local-hostname: vm_hostname
   # eth0 is the default network interface enabled in the image. You can configure
    static network settings with an entry like the following.
   network-interfaces: |
     auto eth0
     iface eth0 inet static
     address 192.168.1.10
     network 192.168.1.0
     netmask 255.255.255.0
     broadcast 192.168.1.255
     gateway 192.168.1.254
   ```

   以您選擇的 VM 主機名稱取代 *vm_hostname*，然後視需要設定網路設定。

   c. 儲存並關閉 meta-data 組態檔案。

   例如，指定 VM 主機名稱 (meta-data) 的 amazonlinux.onprem 組態檔案內容，會設定預設網絡介面 (eth0)，並指定必要網路設備的靜態 IP 地址，請參閱範例 Seed.iso 檔案。

3. 建立 user-data 組態檔案。

   a. 建立名為 user-data 的新檔案。

   b. 使用您偏好的文字編輯器開啟 user-data 檔案，然後新增下列命令。

   ```
   #cloud-config
   #vim:syntax=yaml
   users:
   # A user by the name `ec2-user` is created in the image by default.
     - default
   chpasswd:
     list: |
       ec2-user:plain_text_password
   # In the above line, do not add any spaces after 'ec2-user:'.
   ```

以您為預設的 ec2-user 使用者帳戶所選擇的密碼取代 *plain_text_password*。

c. (選用) 根據預設，cloud-init 會在每次 VM 開機時套用網路設定。新增下列內容，以避免 cloud-init 在每次開機時套用網路設定，並保留第一次開機時套用的網路設定。

```
# NOTE: Cloud-init applies network settings on every boot by default. To retain
 network settings
# from first boot, add the following 'write_files' section:
write_files:
  - path: /etc/cloud/cloud.cfg.d/80_disable_network_after_firstboot.cfg
    content: |
      # Disable network configuration after first boot
      network:
        config: disabled
```

d. 儲存並關閉 user-data 組態檔案。

您也可以建立其他使用者帳戶並指定其存取機制、密碼和金鑰對。如需關於受支援指示詞的詳細資訊，請參閱模組參考。如需建立三個額外使用者，並為預設的 user-data 使用者帳戶指定自訂密碼的範例 ec2-user 檔案，請參閱範例 Seed.iso 檔案。

4. 使用 seed.iso 和 meta-data 組態檔案建立 user-data 開機映像。

如為 Linux，您可使用像是 genisoimage 等工具。導覽至 seedconfig 資料夾並執行下列命令。

```
$ genisoimage -output seed.iso -volid cidata -joliet -rock user-data meta-data
```

如為 macOS，您可使用像是 hdiutil 等工具。導覽至 seedconfig 資料夾的上一層並執行下列命令。

```
$ hdiutil makehybrid -o seed.iso -hfs -joliet -iso -default-volume-name cidata
 seedconfig/
```

步驟 2：下載 AL2 VM 映像

我們為每個支援的虛擬化平台提供不同的 AL2 VM 映像。您可以檢視支援的平台清單，並從 Amazon Linux 2 虛擬機器映像頁面中下載所選平台的正確 VM 映像。

步驟 3：開機並連接至您的新 VM

若要開機並連線至新的 VM，您必須擁有`seed.iso`開機映像（在步驟 1 中建立）和 AL2 VM 映像（在步驟 2 中下載）。根據您所選擇的 VM 平台不同，步驟可能有所差異。

VMware vSphere

VMware 的虛擬機器映像會以 OVF 格式提供。

使用 VMware vSphere 將虛擬機器開機

1. 為 `seed.iso` 檔案建立新的資料存放區，或將其新增至現有資料存放區。
2. 部署 OVF 範本，但尚不要啟動虛擬機器。
3. 在 Navigator (導覽器) 面板中，於新虛擬機器上按一下滑鼠右鍵，然後選擇 Edit Settings (編輯設定)。
4. 在 Virtual Hardware (虛擬硬體) 索引標籤上，針對 New device (新裝置)，選擇 CD/DVD Drive (CD/DVD 光碟機)，然後選擇 Add (新增)。
5. 針對 New CD/DVD Drive (新增 CD/DVD 光碟機)，選擇 Datastore ISO File (資料存放區 ISO 檔案)。選取您新增 `seed.iso` 檔案的資料存放區，瀏覽至並選取 `seed.iso` 檔案，然後選取 OK (確定)。
6. 針對新增 CD/DVD 光碟機，選取連線，然後選取確定。

將資料存放區與虛擬機器建立關聯後，您應該可以將其開機。

KVM

使用 KVM 將虛擬機器開機

1. 開啟 Create new VM (建立新虛擬機器) 精靈。
2. 針對步驟 1，選擇 Import existing disk image (匯入現有磁碟映像)。
3. 針對步驟 2，瀏覽至並選取虛擬機器映像。若為 OS type (作業系統類型) 和 Version (版本)，請分別選擇 Linux 和 Red Hat Enterprise Linux 7.0。
4. 針對步驟 3，指定要使用的 RAM 數量和 CPU 數量。
5. 針對步驟 4，輸入新虛擬機器的名稱，然後選取Customize configuration before install (安裝前自訂組態)，然後選取 Finish (完成)。
6. 在虛擬機器的 Configuration (組態) 視窗中，選擇 Add Hardware (新增硬體)。

7. 在 Add New Virtual Hardware (新增虛擬硬體) 視窗中，選擇 Storage (儲存體)。
8. 在 Storage (儲存區) 組態中，選取 Select or create custom storage (選取或建立自訂儲存區)。針對 Device type (裝置類型)，選擇 CDROM device (CDROM 裝置)。選取 Manage (管理)、Browse Local (瀏覽本機)，然後導覽至並選取 `seed.iso` 檔案。選擇 Finish (完成)。
9. 選擇 Begin Installation (開始安裝)。

Oracle VirtualBox

使用 Oracle VirtualBox 將虛擬機器開機

1. 開啟 Oracle VirtualBox 並選擇 New (新建)。
2. 若為 Name (名稱)，請輸入虛擬機器的描述性名稱，並在 Type (類型) 和 Version (版本) 中分別選取 Linux 和 Red Hat (64-bit) (Red Hat (64 位元))。選擇 Continue (繼續)。
3. 在 Memory size (記憶體大小) 中，指定要配置給虛擬機器的記憶體數量，然後選擇 Continue (繼續)。
4. 在 Hard disk (硬碟) 中，選擇 Use an existing virtual hard disk file (使用現有的虛擬硬碟檔案)，瀏覽至虛擬機器映像並開啟，然後選擇 Create (建立)。
5. 在啟動虛擬機器之前，您必須將 `seed.iso` 檔案載入虛擬機器的虛擬光碟機中：
   a. 選取新的虛擬機器，選取 Settings (設定)，然後選取 Storage (儲存區)。
   b. 在 Storage Devices (儲存裝置) 清單中，於 Controller: IDE (控制器: IDE) 下，選擇 Empty (空白) 的光碟機。
   c. 在光碟機的 Attributes (屬性) 區段中，選取 瀏覽按鈕，選取 Choose Virtual Optical Disk File (選取虛擬光碟檔案)，然後選取 `seed.iso` 檔案。選擇 OK (確定) 以套用變更並關閉「Settings (設定)」。

將 `seed.iso` 檔案新增至虛擬光碟機之後，您就可以啟動虛擬機器。

Microsoft Hyper-V

Microsoft Hyper-V 的虛擬機器映像會壓縮成一個 Zip 檔案。您必須擷取 zip 檔案的內容。

使用 Microsoft Hyper-V 將虛擬機器開機

1. 開啟新的 Virtual Machine Wizard (虛擬機器精靈)。
2. 提示您選取世代時，請選取第 1 代。
3. 提示您設定網路介面卡時，針對 Connection (連線) 選擇 External (外部)。

4. 提示您連線虛擬硬碟時，請選取 Use an existing virtual hard disk (使用現有的虛擬硬碟)，選取 Browse (瀏覽)，然後瀏覽至並選取虛擬機器映像。選擇 Finish (完成) 來建立 VM。
5. 在新虛擬機器上按一下滑鼠右鍵，然後選擇 Settings (設定)。在 Settings (設定) 視窗中，於 IDE Controller 1 (IDE 控制器 1) 下，選擇 DVD Drive (DVD 光碟機)。
6. 針對 DVD 光碟機，選取 Image file (映像檔)，然後瀏覽並選取 `seed.iso` 檔案。
7. 套用變更並啟動虛擬機器。

在 VM 開機之後，使用您在 `user-data` 組態檔案中定義的其中一個使用者帳戶登入。第一次登入之後，您可以從 VM 中斷 `seed.iso` 開機映像的連接。

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 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 特定
- 作業系統偵測的範例程式碼

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 會指定先`/etc/os-release`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
- 欄位類型：機器可讀與人類可讀
- /etc/os-release 範例
- 與其他分佈的比較

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

有些檔案專屬於 Amazon Linux，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**
>
> 在任何真實程式碼中，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，而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";
    }
}
```

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)
```

AWSAL2 中的 整合

AWS命令列工具

AWS Command Line Interface(AWS CLI) 是一種開放原始碼工具，提供一致的界面，可讓您AWS 服務在命令列 Shell 中使用命令與 互動。如需詳細資訊，請參閱AWS Command Line Interface《 使用者指南》中的什麼是 AWS Command Line Interface？。

AL2 和 AL1 具有AWS CLI預先安裝的 第 1 版。Amazon Linux 的目前版本 AL2023 已AWS CLI預先安裝 第 2 版。如需在 AL2023 AWS CLI上使用 的詳細資訊，請參閱《Amazon Linux AL20232023 入門。

開始使用程式設計執行期

AL2 提供特定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，例如 PHP。若要尋找如何安裝和管理這些名稱版本套件的資訊，請使用 yum命令來搜尋和安裝這些套件。如需詳細資訊，請參閱套件儲存庫。

下列主題說明 AL2 中每個語言執行期的運作方式。

主題

- CAL2 Fortran中的 C++、 和
- Go in AL2
- Java 在 AL2 中
- Perl 在 AL2 中
- PHP 在 AL2 中
- Python 在 AL2 中
- AL2 中的 Rust

CAL2 Fortran中的 C++、 和

AL2 同時包含 GNU 編譯器集合 (GCC) 和 的Clang前端LLVM。

的主要版本在 AL2 的生命週期內GCC將保持不變。錯誤和安全性修正可能會回溯到 AL2 中GCC隨附的 主要版本。

根據預設，AL2 包含 7.3 版GCC，其中建置了幾乎所有套件。gcc10 套件在有限的範圍內提供 GCC 10 個，但不建議使用 GCC 10 來建置套件。

建置 AL2 RPMs 的預設編譯器旗標包含一些最佳化和強化旗標。如果您使用 建置自己的程式碼，我們建議您包含一些最佳化和強化旗標GCC。

AL2023 中的預設編譯器和最佳化旗標會改善 AL2 中存在的項目。

Go in AL2

您可能想要使用 AL2 隨附的工具鏈，在 Amazon Linux Go上建置自己的程式碼。

Go 工具鏈將在 AL2 的整個生命週期中更新。這可能是為了回應我們交付的工具鏈中的任何 CVE，或作為在另一個套件中解決 CVE 的先決條件。

Go 是相對快速移動的程式設計語言。在某些情況下，寫入 的現有應用程式Go可能需要適應新版本的Go工具鏈。如需 的詳細資訊Go，請參閱 Go 1 和Go程式的未來。

雖然 AL2 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 AL2 中提供Go的工具鏈。

在 AL2 的生命週期內，舊版套件不會從儲存庫中移除。如果需要較早Go的工具鏈，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較早的版本。

如果您想要在 AL2 上建置自己的Go程式碼，您可以使用 AL2 中包含Go的工具鏈，並知道此工具鏈可能會在 AL2 的生命週期內向前移動。

Java 在 AL2 中

AL2 提供數個 Amazon Corretto 版本，以支援Java以 為基礎的工作負載，以及一些OpenJDK版本。我們建議您遷移至 Amazon Corretto，以準備遷移至 AL2023。

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 套件適用於 Corretto 1.8.0、Corretto 11 和 Corretto 17 的每個套件。

AL2 中的每個 Corretto 版本都支援與 Corretto 版本相同的期間，或直到 AL2 生命週期結束，以先到者為準。如需詳細資訊，請參閱 Amazon Corretto FAQs。

Perl 在 AL2 中

AL2 提供 5.16 版的Perl程式設計語言。

Perl AL2 中的模組

AL2 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 不會嘗試封裝每個可能的Perl模組。其他作業系統 RPMs可能會依賴封裝為 RPM 的模組，因此 Amazon Linux 會優先確保它們的安全修補，而不是單純的功能更新。

AL2 也包含 ，CPAN讓Perl開發人員可以使用模組的慣用套件管理員Perl。

PHP 在 AL2 中

AL2 目前提供兩種完全支援的PHP程式設計語言版本，作為 的一部分AL2 Extras 程式庫。每個PHP版本都支援與上游相同的時間範圍PHP，如 中已棄用日期所列Amazon Linux 2 Extras 的清單。

如需有關如何使用 AL2 Extras 在您的執行個體上安裝應用程式和軟體更新的資訊，請參閱 AL2 Extras 程式庫。

為了協助遷移至 AL2023，8.1 PHP 和 8.2 皆可在 AL2 和 AL2023 上使用。

> **Note**
>
> AL2 在 中包含 PHP 7.1、7.2、7.3 和 7`amazon-linux-extras`.4。所有這些額外項目都是 EOL，不保證會取得任何其他安全性更新。
> 若要了解 AL2 中每個版本的 何時PHP棄用，請參閱 Amazon Linux 2 Extras 的清單。

從舊版 PHP 8.x 遷移

上游PHP社群整合了完整的遷移文件，以便從 PHP 8.1 移至 PHP 8.2。也存在從 PHP 8.0 遷移到 8.1 的文件。

AL2 包含 PHP 8.0、8.1 和 8.2`amazon-linux-extras`，可實現 AL2023 的有效升級路徑。若要了解 AL2 中每個版本的 何時PHP棄用，請參閱 Amazon Linux 2 Extras 的清單。

從 PHP 7.x 版本遷移

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 8.1 PHP 和 PHP 8.2 參考的文件，您有將 PHP 型應用程式遷移至現代 所需的所有步驟。 PHP

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。

> **Note**
>
> 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。

Python 在 AL2 中

做為 AL2 核心套件長期支援承諾的一部分，AL2 提供 Python 2.7 至 20AL26 年 6 月的支援和安全性修補程式。此支援延伸超過 2020 年 Python 1 月 2.7 EOL 的上游Python社群宣告。

> **Note**
>
> AL2023 完全移除 Python 2.7。任何需要 的元件現在Python都會寫入以使用 Python 3。

AL2 使用對 2.7 Python 具有硬相依性的yum套件管理員。在 AL2023 中，dnf套件管理員已遷移至 Python 3，且不再需要 Python 2.7。AL2023 已完全移至 Python 3。我們建議您完成遷移至 Python 3。

AL2 中的 Rust

您可能想要使用 AL2 隨附的工具鏈，在 AL2 Rust上建置自己的程式碼。

Rust 工具鏈將在 AL2 的整個生命週期中更新。這可能是為了回應我們運送的工具鏈中的 CVE，或作為另一個套件中 CVE 更新的先決條件。

Rust 是相對快速的移動語言，新版本大約每六週發行一次。新版本可能會新增新的語言或標準程式庫功能。雖然 AL2 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 AL2 中提供Rust的工具鏈。

在 AL2 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Rust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 可用的相同程序從儲存庫安裝先前的版本。

若要在 AL2 上建置您自己的Rust程式碼，請使用 AL2 中包含Rust的工具鏈，並了解此工具鏈可能會在 AL2 的整個生命週期中向前移動。

AL2 核心

AL2 最初隨附於 4.14 核心，目前預設為 5.10 版。如果您仍在使用 4.14 核心，建議您遷移至 5.10 核心。

AL2 支援核心即時修補。

主題

AL2 支援的核心

支援的核心版本

目前，核心版本 4.14 和 5.10 提供 AL2 AMIs，預設版本 5.10。我們建議您使用 AL2 AMI 搭配核心 5.10。

AL2023 AMI 適用於核心 6.1 版。如需詳細資訊，請參閱《Amazon Linux AL2023 使用者指南》中的來自 AL2 的 AL2023 核心變更。

支援時間範圍

將支援 AL2 上可用的 5.10 核心，直到 AL2 AMI 達到標準支援結束為止。

即時修補支援

AL2 核心版本	支援核心即時修補
4.14	是
5.10	是
5.15	否

AL2 上的核心即時修補

> **Important**
>
> Amazon Linux 將於 2025-10-31 結束 AL2 核心 4.14 的即時修補。我們鼓勵客戶使用核心 5.10 做為 AL2 的預設核心 （ 請參閱 AL2 支援的核心)，或使用核心 6.1 和 6.12 移至 AL2023。Amazon Linux 將為 AL2 核心 5.10 提供即時修補程式，直到 AL2 在 2026-06-30 的生命週期結束為止。

適用於 AL2 的 Kernel Live Patching 可讓您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。這可讓您受益於改善的服務和應用程式可用性，同時套用這些修正，直到系統可以重新啟動為止。

如需 AL2023 的 Kernel Live Patching 相關資訊，請參閱《Amazon Linux 2023 使用者指南》中的 AL2023 上的 Kernel Live Patching。

AWS 為 AL2 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 – 包含修正與 CVE 無關的重大錯誤和穩定性問題。

AWS 提供 AL2 核心版本的核心即時修補程式，最長可達發行後 3 個月。在 3 個月期間之後，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。

AL2 核心即時修補程式可在現有 AL2 儲存庫中以簽章 RPM 套件的形式提供。這些修補程式可以使用現有的 yum 工作流程安裝在個別執行個體上，也可以使用 AWS Systems Manager 安裝在受管執行個體群組上。

在 AL2 上提供 Kernel Live Patching，無需額外費用。

主題

- 限制
- 常見問答集

支援的組態和先決條件

執行 AL22 的 Amazon EC2 執行個體和內部部署虛擬機器支援 Kernel Live Patching。

若要在 AL2 上使用核心即時修補，您必須使用：

- x86_64 架構上的核心版本 4.14 或 5.10
- ARM64 架構上的核心版本 5.10

政策要求

若要從 Amazon Linux 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適用於 Amazon S3 的 Amazon 虛擬私有雲端 (VPC) 端點，則需要確保您的 VPC 端點政策允許存取這些公有儲存貯體。

下表說明對於 Kernel Live Patching，EC2 可能需要存取的每個 Amazon S3 儲存貯體。

S3 儲存貯體 ARN	Description
arn:aws:s3:::packages.*region*.amazonaws.com/*	含有 Amazon Linux AMI 套件的 Amazon S3 儲存貯體
arn:aws:s3:::repo.*region*.amazonaws.com/*	含有 Amazon Linux AMI 儲存庫的 Amazon S3 儲存貯體
arn:aws:s3:::amazonlinux.*region*.amazonaws.com/*	包含 AL2 儲存庫的 Amazon S3 儲存貯體
arn:aws:s3:::amazonlinux-2-repos-*region*/*	包含 AL2 儲存庫的 Amazon S3 儲存貯體

下方政策會說明如何限制存取屬於您組織的 Amazon S3 儲存貯體的身分和資源，並提供 Kernel Live Patching 所需的 Amazon S3 儲存貯體的存取權限。使用您組織的值取代 *region*、*principal-org-id* 以及 *resource-org-id*。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowRequestsByOrgsIdentitiesToOrgsResources",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "principal-org-id",
          "aws:ResourceOrgID": "resource-org-id"
        }
      }
    },
    {
      "Sid": "AllowAccessToAmazonLinuxAMIRepositories",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::packages.region.amazonaws.com/*",
        "arn:aws:s3:::repo.region.amazonaws.com/*",
        "arn:aws:s3:::amazonlinux.region.amazonaws.com/*",
        "arn:aws:s3:::amazonlinux-2-repos-region/*"
      ]
    }
  ]
}
```

使用 Kernel Live Patching

您可以使用執行個體本身的命令列在個別執行個體上啟用和使用 Kernel Live Patching，也可以使用 AWS Systems Manager 在受管執行個體群組上啟用和使用 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。

如需在一組受管執行個體上啟用和使用 Kernel Live Patching 的詳細資訊，請參閱AWS Systems Manager 《 使用者指南》中的在 AL2 執行個體上使用 Kernel Live Patching。

主題

啟用 Kernel Live Patching

AL2 上的核心即時修補預設為停用。若要使用即時修補，您必須安裝 Kernel Live Patching 的 yum 外掛程式，並啟用即時修補功能。

先決條件

Kernel Live Patching 需要 `binutils`。如果您未安裝 `binutils`，請使用下列命令進行安裝：

```
$ sudo yum install binutils
```

啟用 Kernel Live Patching

1. 核心即時修補程式適用於下列 AL2 核心版本：

   - `x86_64` 架構上的核心版本 `4.14` 或 `5.10`
   - `ARM64` 架構上的核心版本 `5.10`

   若要檢查您的核心版本，請執行下列命令。

   ```
   $ sudo yum list kernel
   ```

2. 如果您已經有支援的核心版本，請略過此步驟。如果您沒有支援的核心版本，請執行下列命令，將核心更新為最新版本，並重新啟動執行個體。

```
$ sudo yum install -y kernel
```

```
$ sudo reboot
```

3. 安裝 Kernel Live Patching 的 yum 外掛程式。

```
$ sudo yum install -y yum-plugin-kernel-livepatch
```

4. 啟用 Kernel Live Patching 的 yum 外掛程式。

```
$ sudo yum kernel-livepatch enable -y
```

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。

5. 若要確認核心即時修補的 yum 外掛程式是否成功安裝，請執行下列命令。

```
$ rpm -qa | grep kernel-livepatch
```

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 Kernel Live Patching 已順利啟用，此命令會傳回包含初始空白核心即時修補程式 RPM 的清單。以下為範例輸出。

```
yum-plugin-kernel-livepatch-1.0-0.11.amzn2.noarch
kernel-livepatch-5.10.102-99.473-1.0-0.amzn2.x86_64
```

6. 安裝 kpatch 套件。

```
$ sudo yum install -y kpatch-runtime
```

7. 如果之前已安裝 kpatch 服務，請更新此服務。

```
$ sudo yum update kpatch-runtime
```

8. 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
```

9. 在 AL2 Extras 程式庫中啟用核心即時修補主題。本主題包含核心即時修補程式。

```
$ sudo amazon-linux-extras enable livepatch
```

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 AL2 安全提醒的詳細資訊，包括核心即時修補程式的提醒，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。

```
$ yum updateinfo list
```

下面顯示了範例輸出。

```
Loaded plugins: extras_suggestions, kernel-livepatch, langpacks, priorities, update-
motd
ALAS2LIVEPATCH-2020-002 important/Sec. kernel-
livepatch-5.10.102-99.473-1.0-3.amzn2.x86_64
ALAS2LIVEPATCH-2020-005 medium/Sec. kernel-livepatch-5.10.102-99.473-1.0-4.amzn2.x86_64
updateinfo list done
```

列出 CVE 的所有可用核心即時修補程式

使用下列命令。

```
$ yum updateinfo list cves
```

下面顯示了範例輸出。

```
Loaded plugins: extras_suggestions, kernel-livepatch, langpacks, priorities, update-
motdamzn2-core/2/x86_64 | 2.4 kB 00:00:00
CVE-2019-15918 important/Sec. kernel-livepatch-5.10.102-99.473-1.0-3.amzn2.x86_64
```

```
CVE-2019-20096 important/Sec. kernel-livepatch-5.10.102-99.473-1.0-3.amzn2.x86_64
CVE-2020-8648 medium/Sec. kernel-livepatch-5.10.102-99.473-1.0-4.amzn2.x86_64
updateinfo list done
```

套用核心即時修補程式

您可以使用 yum 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 yum 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 AL2 核心套用核心即時修補程式。

   ```
   $ sudo yum install kernel-livepatch-kernel_version.x86_64
   ```

   例如，下列命令會套用 AL2 核心版本 的核心即時修補程式`5.10.102-99.473`。

   ```
   $ sudo yum install kernel-livepatch-5.10.102-99.473-1.0-4.amzn2.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 yum update --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。
- AL2 核心會在三個月內收到核心即時修補程式。三個月期間過後，該核心版本不會發行任何新的核心即時修補程式。若要在三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移至新的核心版本，然後才能在接下來的三個月內繼續接收核心即時修補程式。若要檢查核心版本的支援視窗，請執行 `yum kernel-livepatch supported`。

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。

```
$ kpatch list
```

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。

```
Loaded patch modules:
livepatch_cifs_lease_buffer_len [enabled]
livepatch_CVE_2019_20096 [enabled]
livepatch_CVE_2020_8648 [enabled]

Installed patch modules:
livepatch_cifs_lease_buffer_len (5.10.102-99.473.amzn2.x86_64)
livepatch_CVE_2019_20096 (5.10.102-99.473.amzn2.x86_64)
livepatch_CVE_2020_8648 (5.10.102-99.473.amzn2.x86_64)
```

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 Kernel Live Patching

1. 移除套用的核心即時修補程式 RPM 套件。

```
$ sudo yum kernel-livepatch disable
```

2. 解除安裝 Kernel Live Patching 的 yum 外掛程式。

```
$ sudo yum remove yum-plugin-kernel-livepatch
```

3. 重新啟動執行個體。

```
$ sudo reboot
```

限制

Kernel Live Patching 有以下限制：

- 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更新並重新啟動系統，以照常啟用修補程式。

常見問答集

如需 AL2 核心即時修補的常見問題，請參閱 Amazon Linux 2 核心即時修補常見問答集。

AL2 Extras 程式庫

> **Warning**
>
> epel Extra 會啟用第三方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。如需某些EPEL套件的選項，請參閱 EPEL Amazon Linux 2023 使用者指南中的一節。

透過 AL2，您可以使用 Extras Library 在執行個體上安裝應用程式和軟體更新。這些軟體更新即為主題。您可安裝特定版本的主題，或略過版本資訊，以使用最新版本。額外功能有助於減輕作業系統穩定性與可用軟體新鮮度之間的影響。

Extras 主題的內容不受 Amazon Linux 政策的長期支援和二進位相容性限制。Extras 主題可讓您存取精心策劃的套件清單。套件的版本可能會經常更新，也可能在與 AL2 相同的時間內不受支援。

> **Note**
>
> 在 AL2 達到 EOL 之前，個別額外項目主題可能會遭到取代。

若要列出可用的主題，請使用下列命令。

```
[ec2-user ~]$ amazon-linux-extras list
```

若要啟用主題並安裝其套件的最新版本，以確保新鮮度，請使用下列命令。

```
[ec2-user ~]$ sudo amazon-linux-extras install topic
```

若要啟用主題並安裝其套件的特定版本以確保穩定性，請使用下列命令。

```
[ec2-user ~]$ sudo amazon-linux-extras install topic=version topic=version
```

若要從主題移除安裝的套件，請使用下列命令。

```
[ec2-user ~]$ sudo yum remove $(yum list installed | grep amzn2extra-topic | awk
 '{ print $1 }')
```

> **Note**
>
> 此命令不會移除已安裝為 Extra 相依性的套件。

若要停用主題並讓 yum 套件管理員無法存取套件，請使用下列命令。

```
[ec2-user ~]$ sudo amazon-linux-extras disable topic
```

> **Important**
>
> 此命令適用於進階使用者。不當使用此命令可能會導致套件相容性衝突。

Amazon Linux 2 Extras 的清單

額外的名稱	已棄用日期
BCC	
GraphicsMagick1.3	
R3.4	
R4	
ansible2	2023-09-30
aws-nitro-enclaves-cli	
awscli1	
collectd	
collectd-python3	
corretto8	
dnsmasq	

額外的名稱	已棄用日期
dnsmasq2.85	2025-05-01
docker	
ecs	
emacs	2018-11-14
Epel	2024-06-30
消防員	2022-11-08
Firefox	
gimp	2018-11-14
golang1.11	2023-08-01
golang1.19	2023-09-30
golang1.9	2018-12-14
haproxy2	
httpd_modules	
java-openjdk11	2024-09-30
kernel-5.10	
kernel-5.15	
kernel-5.4	
kernel-ng	2022-08-08
lamp-mariadb10.2-php7.2	2020-11-30
libreoffice	

額外的名稱	已棄用日期
livepatch	
lustre	
lustre2.10	
lynis	
mariadb10.5	2025-06-24
mate-desktop1.x	
memcached1.5	
模擬	
模擬2	
單一	
nano	2018-11-14
nginx1	
nginx1.12	2019-09-20
nginx1.22.1	
php7.1	2020-01-15
php7.2	2020-11-30
php7.3	2021-12-06
php7.4	2022-11-03
php8.0	2023-11-26
php8.1	2025-12-31

額外的名稱	已棄用日期
php8.2	
postgresql10	2023-09-30
postgresql11	2023-11-09
postgresql12	2024-11-14
postgresql13	2025-11-13
postgresql14	
postgresql9.6	2022-08-09
python3	2018-08-22
python3.8	2024-10-14
redis4.0	2021-05-25
redis6	2026-01-31
ruby2.4	2020-08-27
ruby2.6	2023-03-31
ruby3.0	2024-03-31
rust1	2025-05-01
selinux-ng	
squid4	2023-09-30
測試	
tomcat8.5	2024-03-31
tomcat9	

額外的名稱	已棄用日期
unbound1.13	2025-05-01
unbound1.17	
vim	2018-11-14

AL2 預留使用者和群組

在佈建映像和安裝特定套件期間，AL2 會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

Amazon Linux 2 預留使用者的清單

依 UID 列出

使用者名稱	UID
根	0
bin	1
常駐程式	2
廣告	3
lp	4
同步	5
shutdown	6
停止	7
郵件	8
uucp	10
operator	11
遊戲	12

使用者名稱	UID
ftp	14
oprofile	16
pkiuser	17
squid	23
名為	25
postgres	26
mysql	27
nscd	28
nscd	28
rpcuser	29
rpc	32
amandabackup	33
ntp	38
mailman	4.1
gdm	42
mailnull	47
apache	48
smmsp	51
tomcat	53
ldap	55

使用者名稱	UID
tss	59
nslcd	65
pegasus	66
avahi	70
tcpdump	72
sshd	74
radvd	75
cyrus	76
arpwatch	77
fax	78
dbus	81
postfix	89
quagga	92
radiusd	95
radiusd	95
hsqldb	96
dovecot	97
身分	98
沒有人	99
qemu	107

使用者名稱	UID
usbmuxd	113
stap-server	155
avahi-autoipd	170
脈衝	171
rtkit	172
dhcpd	177
sanlock	179
haproxy	188
hacluster	189
systemd-journal-gateway	191
systemd-network	192
systemd-resolve	193
uuidd	357
tang	358
stapdev	359
stapsys	360
stapusr	361
systemd-journal-upload	362
systemd-journal-remote	363
已清除	364

使用者名稱	UID
pesign	365
pcpqa	366
pcp	367
memcached	368
ipsilon	369
ipaapi	370
kdcproxy	371
ods	372
sssd	373
叢集	374
饋送	375
dovenull	376
coroqnetd	377
clevis	378
clamscan	379
clamilt	380
clamupdate	381
著色	382
geoclue	383
aws-kinesis-agent-user	384

使用者名稱	UID
cwagent	385
取消繫結	386
polkitd	387
saslauth	388
dirsrv	389
chrony	996
ec2-instance-connect	997
rngd	998
libstoragemgmt	999
ec2-user	1000
nfsnobody	65534

依名稱列出

使用者名稱	UID
廣告	3
amandabackup	33
apache	48
arpwatch	77
avahi	70
avahi-autoipd	170

使用者名稱	UID
aws-kinesis-agent-user	384
bin	1
chrony	996
clamilt	380
clamscan	379
clamupdate	381
clevis	378
著色	382
coroqnetd	377
cwagent	385
cyrus	76
常駐程式	2
dbus	81
dhcpd	177
dirsrv	389
dovecot	97
dovenull	376
ec2-instance-connect	997
ec2-user	1000
fax	78

使用者名稱	UID
饋送	375
ftp	14
遊戲	12
gdm	42
geoclue	383
叢集	374
hacluster	189
停止	7
haproxy	188
hsqldb	96
身分	98
ipaapi	370
ipsilon	369
kdcproxy	371
ldap	55
libstoragemgmt	999
lp	4
郵件	8
mailman	4.1
mailnull	47

使用者名稱	UID
memcached	368
mysql	27
名為	25
nfsnobody	65534
沒有人	99
nscd	28
nscd	28
nslcd	65
ntp	38
ods	372
operator	11
oprofile	16
pcp	367
pcpqa	366
pegasus	66
pesign	365
pkiuser	17
polkitd	387
postfix	89
postgres	26

使用者名稱	UID
脈衝	171
qemu	107
quagga	92
radiusd	95
radiusd	95
radvd	75
rngd	998
根	0
rpc	32
rpcuser	29
rtkit	172
已清除	364
sanlock	179
saslauth	388
shutdown	6
smmsp	51
squid	23
sshd	74
sssd	373
stap-server	155

使用者名稱	UID
stapdev	359
stapsys	360
stapusr	361
同步	5
systemd-journal-gateway	191
systemd-journal-remote	363
systemd-journal-upload	362
systemd-network	192
systemd-resolve	193
tang	358
tcpdump	72
tomcat	53
tss	59
取消繫結	386
usbmuxd	113
uucp	10
uuidd	357

Amazon Linux 2 預留群組的清單

依 GID 列出

Group name (群組名稱)	GID
根	0
bin	1
常駐程式	2
sys	3
廣告	4
tty	5
磁碟	6
磁碟	6
lp	7
mem	8
kmem	9
車輪	10
cdrom	11
郵件	12
uucp	14
man	15
oprofile	16
pkiuser	17
撥出	18
鬆軟	19

Group name (群組名稱)	GID
遊戲	20
配置	21
utmp	22
squid	23
已命名為	25
postgres	26
mysql	27
nscd	28
nscd	28
rpcuser	29
rpc	32
磁帶	33
磁帶	33
utempter	35
kvm	36
ntp	38
影片	39
浸入	40
mailman	4.1
gdm	42

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
pegasus	65
avahi	70
tcpdump	72
sshd	74
radvd	75
saslauth	76
saslauth	76
arpwatch	77
fax	78
dbus	81
screen	84

Group name (群組名稱)	GID
quaggavt	85
wbpriv	88
wbpriv	88
postfix	89
postdrop	90
quagga	92
radiusd	95
radiusd	95
hsqldb	96
dovecot	97
身分	98
沒有人	99
使用者	100
qemu	107
usbmuxd	113
stap-server	155
stapusr	156
stapusr	156
stapsys	157
stapdev	158

Group name (群組名稱)	GID
avahi-autoipd	170
脈衝	171
rtkit	172
dhcpd	177
sanlock	179
haproxy	188
haclient	189
systemd-journal	190
systemd-journal	190
systemd-journal-gateway	191
systemd-network	192
systemd-resolve	193
usbmon	351
wireshark	352
uuidd	353
tang	354
systemd-journal-upload	355
sfcb	356
systemd-journal-remote	356
已清除	357

Group name (群組名稱)	GID
pesign	358
pcpqa	359
pcp	360
memcached	361
virtlogin	362
ipsilon	363
pkcs11	364
ipaapi	365
kdcproxy	366
ods	367
sssd	368
libvirt	369
叢集	370
fedfs	371
dovenull	372
docker	373
coroqnetd	374
clevis	375
clamscan	376
clamilt	377

Group name (群組名稱)	GID
病毒群組	378
病毒群組	378
病毒群組	378
clamupdate	379
著色	380
geoclue	381
printadmin	382
aws-kinesis-agent-user	383
cwagent	384
pulse-rt	385
脈衝存取	386
取消繫結	387
polkitd	388
dirsrv	389
cgred	993
chrony	994
ec2-instance-connect	995
rngd	996
libstoragemgmt	997
ssh_keys	998

Group name (群組名稱)	GID
input	999
ec2-user	1000
nfsnobody	65534

依名稱列出

Group name (群組名稱)	GID
廣告	4
apache	48
arpwatch	77
音訊	63
avahi	70
avahi-autoipd	170
aws-kinesis-agent-user	383
bin	1
cdrom	11
cgred	993
chrony	994
clamilt	377
clamscan	376
clamupdate	379

Group name (群組名稱)	GID
clevis	375
著色	380
coroqnetd	374
cwagent	384
常駐程式	2
dbus	81
dhcpd	177
撥出	18
浸入	40
dirsrv	389
磁碟	6
磁碟	6
docker	373
dovecot	97
dovenull	372
ec2-instance-connect	995
ec2-user	1000
fax	78
fedfs	371
鬆軟	19

Group name (群組名稱)	GID
ftp	50
遊戲	20
gdm	42
geoclue	381
叢集	370
haclient	189
haproxy	188
hsqldb	96
身分	98
input	999
ipaapi	365
ipsilon	363
kdcproxy	366
kmem	9
kvm	36
ldap	55
libstoragemgmt	997
libvirt	369
鎖定	54
lp	7

Group name (群組名稱)	GID
郵件	12
mailman	4.1
mailnull	47
man	15
mem	8
memcached	361
mysql	27
已命名為	25
nfsnobody	65534
沒有人	99
nscd	28
nscd	28
ntp	38
ods	367
oprofile	16
pcp	360
pcpqa	359
pegasus	65
pesign	358
pkcs11	364

Group name (群組名稱)	GID
pkiuser	17
polkitd	388
postdrop	90
postfix	89
postgres	26
printadmin	382
脈衝	171
脈衝存取	386
pulse-rt	385
qemu	107
quagga	92
quaggavt	85
radiusd	95
radiusd	95
radvd	75
rngd	996
根	0
rpc	32
rpcuser	29
rtkit	172

Group name (群組名稱)	GID
已清除	357
sanlock	179
saslauth	76
saslauth	76
screen	84
sfcb	356
配置	21
smmsp	51
squid	23
ssh_keys	998
sshd	74
sssd	368
stap-server	155
stapdev	158
stapsys	157
stapusr	156
stapusr	156
sys	3
systemd-journal	190
systemd-journal	190

Group name (群組名稱)	GID
systemd-journal-gateway	191
systemd-journal-remote	356
systemd-journal-upload	355
systemd-network	192
systemd-resolve	193
tang	354
磁帶	33
磁帶	33
tcpdump	72
tomcat	53
tss	59
tty	5
取消繫結	387
usbmon	351
usbmuxd	113
使用者	100
utempter	35
utmp	22
uucp	14
uuidd	353

Group name (群組名稱)	GID
影片	39
virtlogin	362
病毒群組	378
病毒群組	378
病毒群組	378
wbpriv	88
wbpriv	88
車輪	10
wireshark	352

AL2 來源套件

您可用 Amazon Linux 提供的工具檢視執行個體上已安裝套件的來源，以做為參考之用。來源套件適用於 Amazon Linux 和線上套件儲存庫內包含的所有套件。決定您要安裝之來源套件的套件名稱，並使用 yumdownloader --source命令在執行中的執行個體中檢視來源。例如：

```
[ec2-user ~]$ yumdownloader --source bash
```

來源 RPM 可以解壓縮，並且您可以使用標準 RPM 工具檢視來源樹狀目錄，以供參考。完成偵錯後，套件便可供使用。

AL2 中的安全與合規

的雲端安全性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，這些架構專為滿足最安全敏感組織的需求而建置。

安全性是 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 AWSCloud 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的安全有效性。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。

在 AL2 上啟用 FIPS 模式

本節說明如何在 AL2 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

先決條件

- 可存取網際網路以下載必要套件的現有 AL2 Amazon EC2 執行個體。如需啟動 AL2 Amazon EC2 執行個體的詳細資訊，請參閱 Amazon EC2 上的 AL2 Amazon EC2。
- 您必須使用 SSH 或 連線到 Amazon EC2 AWS Systems Manager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 連線到 AL2 執行個體AWS Systems Manager。

2. 請確保系統為最新版本。如需詳細資訊，請參閱套件儲存庫。
3. 執行下列命令來安裝和啟用dracut-fips模組。

```
sudo yum -y install dracut-fips
sudo dracut -f
```

4. 使用下列命令在 Linux 核心命令列上啟用 FIPS 模式。這將為 AL2 常見問答集中列出的模組啟用整個系統的 FIPS 模式

```
sudo /sbin/grubby --update-kernel=ALL --args="fips=1"
```

5. 重新啟動您的 AL2 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：

```
sysctl crypto.fips_enabled
```

您應該會看到下列輸出：

```
crypto.fips_enabled = 1
```

您也可以執行下列命令來驗證 OpenSSH 是否處於 FIPS 模式：

```
ssh localhost 2>&1 | grep FIPS
```

您應該會看到下列輸出：

```
FIPS mode initialized
```

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。