aws

用户指南

Amazon Linux 2

Amazon Linux 2: 用户指南

Table of Contents

什么是亚马逊 Linux 2？

亚马逊 Linux 2 (AL2) 是亚马逊 Web Services (AWS) 推出的 Linux 操作系统。 AL2 旨在为在 Amazon EC2 上运行的应用程序提供稳定、安全和高性能的环境。它还包括可与之高效集成的软件包AWS，包括启动配置工具和许多常用的AWS库和工具。 AWS为所有正在运行的实例提供持续的安全和维护更新AL2。许多在 CentOS 上开发的应用程序以及类似的发行版都可以在上面运行。 AL2 AL2 不收取额外费用。

> **Note**
>
> AL2 不再是亚马逊 Linux 的当前版本。 AL2023 是的继任者。 AL2有关更多信息，请参阅《AL2023 用户指南》中的 “比较 AL2 和 023” 以及 AL2023 中的 Package 更改列表。AL2

> **Note**
>
> AL2 密切关注上游 Firefox Extended Support Release (ESR) 版本，并尽快更新到下一个 ESR。有关更多信息，请参阅 Firefox ESR 发布日历和 F irefox 发行说明。

Amazon Linux 可用性

AWS提供 AL2 023 AL2、和亚马逊 Linux 1（AL1前身为亚马逊 Linux AMI）。如果您要从其他 Linux 发行版迁移到亚马逊 Linux，我们建议您迁移到 AL2 023。

> **Note**
>
> 的标准支持 AL1 已于 2020 年 12 月 31 日结束。 AL1 维护支持阶段已于 2023 年 12 月 31 日结束。有关 AL1 EOL 和维护支持的更多信息，请参阅博客文章 Amazon Linux AMI end-of-life 上的更新。

有关 Amazon Linux 的更多信息，请参阅 AL2023 AL2、和。AL1

有关 Amazon Linux 容器映像，请参阅《Amazon Elastic Container Registry 用户指南》中的 Amazon Linux 容器映像。

中已弃用的功能 AL2

以下各节介绍中支持 AL2 和中不存在的功能 AL2023。这是诸如功能和软件包之类的功能，它们存在于中 AL2，但不存在于中 AL2023 ，也不会添加到中 AL2023。有关此功能支持多长时间，请参阅 AL2 文档 AL2。

compat-软件包

提供前缀为的所有软件包compat-都是为了 AL2 与尚未针对该软件包的现代版本重建的旧二进制文件进行二进制兼容。Amazon Linux 的每个新主要版本都不会沿用之前发布版本中的任何 compat- 程序包。

Amazon Linux 发行版（例如 AL2）中的所有compat-软件包都已停产，并且不会出现在后续版本中（例如 AL2023）。我们强烈建议针对更新版本的库重新构建软件。

已弃用的功能已在中停用 AL1，已在中删除 AL2

本节介绍中提供的功能 AL1以及中不再提供的功能 AL2。

> **Note**
>
> 作为维护支持阶段的一部分 AL1，某些软件包的 end-of-life (EOL) 日期早于 EOL。 AL1有关更多信息，请参阅AL1 程序包支持声明。

> **Note**
>
> 某些 AL1 功能在早期版本中已停用。有关信息，请参阅AL1 发行说明。

主题

- 32 位 x86 (i686) AMIs
- aws-apitools-*取而代之的是 AWS CLI
- systemd替换upstart在 AL2

32 位 x86 (i686) AMIs

作为 2014.09 版本的一部分， AL1亚马逊 Linux 宣布这将是最后一个产生 32 位的版本。 AMIs因此，从 2015.03 版本开始， AL1Amazon Linux 不再支持在 32 位模式下运行系统。 AL2 为 x86-64 主机上的 32 位二进制文件提供了有限的运行时支持，并且不提供开发包来支持构建新的 32 位二进制文件。AL2023 不再包含任何 32 位用户空间包。我们建议用户在迁移到 AL2 023 之前完成向 64 位代码的过渡。

如果您需要在 023 上运行 32 位二进制文件，则可以从 AL2 023 之上运行的 AL2 容器 AL2 内部使用 32 位用户空间。 AL2

`aws-apitools-*`取而代之的是 AWS CLI

在 2013 AWS CLI 年 9 月发布之前，提供AWS了一组命令行实用程序，这些实用程序是在中实现的 Java，允许用户调用 Amazon EC2 API。这些工具已于 2015 年停产，AWS CLI成为通过命令行与 Amazon 进行交互 EC2 APIs 的首选方式。这组命令行实用程序包括以下 `aws-apitools-*` 程序包。

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

对 `aws-apitools-*` 程序包的上游支持已于 2017 年 3 月结束。尽管缺乏上游支持，Amazon Linux 仍继续提供其中一些命令行实用程序（例如 `aws-apitools-ec2`），以向用户提供向后兼容性。该 AWS CLI工具比`aws-apitools-*`软件包更强大、更完整，因为它得到了积极维护，并且提供了一种使用所有软件包的方法AWSAPIs。

`aws-apitools-*` 程序包已于 2017 年 3 月弃用，将不会收到进一步更新。其中任何一个软件包的所有用户都应AWS CLI尽快迁移到。这些软件包在 AL2 023 中不存在。

AL1 还提供了`aws-apitools-iam`和`aws-apitools-rds`软件包，这些软件包已在中弃用 AL1，从 AL2 此以后不存在于 Amazon Linux 中。

systemd替换upstart在 AL2

AL2 是第一个使用初systemd始化系统的 Amazon Linux 版本，取而代upstart之 AL1的是。在从较新版本的 Amazon Linux 迁移 AL1 到更新版本的过程中，必须更改任何upstart特定的配置。它不可能systemd在上使用 AL1，因此systemd只能在迁移upstart到最新的 Amazon Linux 主要版本（例如 AL2 或 AL2 023）时才能完成。

已在中弃用 AL2 和删除的功能 AL2023

本节介绍中已提供 AL2但中不再提供的功能 AL2023。

主题

- 32 位 x86（i686）程序包
- aws-apitools-*替换为 AWS CLI
- awslogs已弃用，转而使用统一的 Amazon CloudWatch Logs 代理
- bzr 版本控制系统
- cgroup v1
- log4j 热补丁 (log4j-cve-2021-44228-hotpatch)
- lsb_release 和 system-lsb-core 软件包
- mcrypt
- OpenJDK 7 (java-1.7.0-openjdk)
- Python 2.7
- rsyslog-openssl取代 rsyslog-gnutls
- 网络信息服务（NIS）/yp
- Amazon VPC 中的多个域名 create-dhcp-options
- Sun RPC中的glibc
- audit 日志中的 OpenSSH 密钥指纹
- ld.gold 链接器
- ping6
- ftp 程序包

32 位 x86（i686）程序包

作为 2014.09 版本的一部分 AL1，我们宣布这将是最后一个产生 32 位的版本。 AMIs因此，从 2015.03 版本开始， AL1亚马逊 Linux 不再支持在 32 位模式下运行系统。 AL2 为 x86-64 主机上的 32 位二进制文件提供了有限的运行时支持，并且不提供开发包来支持构建新的 32 位二进制文件。 AL2023 不再包含任何 32 位用户空间软件包。我们建议客户完成向 64 位代码的过渡。

如果您需要在上运行 32 位二进制文件 AL2023，则可以在上面运行的 AL2 容器 AL2 内使用 32 位用户空间。 AL2023

`aws-apitools-*`替换为 AWS CLI

在 2013 AWS CLI 年 9 月发布之前，提供 AWS 了一组命令行实用程序，这些实用程序已在中实现 Java，允许客户调用 Amazon EC2 API。这些工具已于 2015 年被弃用， AWS CLI 成为通过命令行与 Amazon EC2 进行交互 APIs 的首选方式。这包括以下 `aws-apitools-*` 程序包。

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

对 `aws-apitools-*` 程序包的上游支持已于 2017 年 3 月结束。尽管缺乏上游支持，Amazon Linux 仍继续提供其中一些命令行实用程序（例如 `aws-apitools-ec2`），以便为客户提供向后兼容性。该 AWS CLI 工具比`aws-apitools-*`软件包更强大、更完整，因为它得到了积极维护，并且提供了一种使用所有软件包的方法 AWS APIs。

`aws-apitools-*` 程序包已于 2017 年 3 月弃用，将不会收到进一步更新。其中任何一个软件包的所有用户都应 AWS CLI 尽快迁移到。中不存在这些软件包 AL2023。

`awslogs`已弃用，转而使用统一的 Amazon CloudWatch Logs 代理

该`awslogs`软件包已在中弃用 AL2 ，不再存在于中。 AL2023它已被`amazon-cloudwatch-agent`软件包中提供的统一 CloudWatch 日志代理所取代。有关更多信息，请参阅 Amazon CloudWatch 日志用户指南。

bzr 版本控制系统

GNU Bazaar(bzr) 版本控制系统已停产 AL2，不再存在于 AL2023。

建议 bzr 用户将其存储库迁移到 git。

cgroup v1

AL2023 移至统一控制组层次结构 (cgroup v2)，而 AL2 使用 cgroup v1。由于 AL2 不支持 cgroup v2，因此需要在迁移到的过程中完成此迁移。 AL2023

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

> **Note**
>
> 该log4j-cve-2021-44228-hotpatch软件包已在中弃用， AL2 并在中删除。 AL2023

为了回应 CVE-2021-44228，亚马逊 Linux 发布了适用于 Ap ache Log4j 的 Hotpatch 的 RPM 打包版本，适用于和。 AL1 AL2在宣布向 Amazon Linux 添加热补丁时，我们指出：“安装热补丁并不能取代更新到可缓解 CVE-2021-44228 或 CVE-2021-45046 的 log4j 版本。”

热补丁是一种缓解措施，可以留出时间来修补 log4j。的第一个正式发布版本 AL2023 是 CVE-2021-44228 发布的 15 个月后，因此 AL2023 不附带该热补丁（无论是否启用）。

建议在 Amazon Linux 上运行自己的 log4j 版本的客户务必更新到未受 CVE-2021-44228 或 CVE-2021-45046 影响的版本。

lsb_release 和 system-lsb-core 软件包

过去，有些软件会调用该lsb_release命令（ AL2 由软件system-lsb-core包提供）来获取有关其运行的 Linux 发行版的信息。Linux Standards Base (LSB) 引入了此命令，Linux 发行版采用了此命令。Linux 发行版已经演变为使用更简单的标准，将这些信息保存在 /etc/os-release 和其他相关文件中。

os-release 标准来自 systemd。有关更多信息，请参阅 systemd os-release 文档。

AL2023 不随lsb_release命令一起提供，也不包括system-lsb-core软件包。软件应完成向 os-release 标准的过渡，以保持与 Amazon Linux 和其他主要 Linux 发行版的兼容性。

mcrypt

该mcrypt库和相关PHP扩展已在中弃用 AL2，不再存在于中。 AL2023

上游 PHP 在 PHP 7.1 中弃用了 mcrypt 扩展，该版本最初于 2016 年 12 月发布，最终版本于 2019 年 10 月发布。

上游mcrypt库最后一次发布是在 2007 年，并未在 2017 年完成新提交SourceForge 所需的cvs版本控制迁移，最近一次提交（之前只有 3 年）是从 2011 年开始的，删除了该项目有维护者的提法。

建议所有剩余mcrypt的用户将其代码移植到OpenSSL，因为mcrypt不会添加到 AL2023。

OpenJDK 7 (java-1.7.0-openjdk)

> **Note**
>
> AL2023 提供了多个版本的 Amazon Corretto 来支持基于的工作负载。JavaOpenJDK 7 软件包已 AL2在中弃用，并且不再存在于中。 AL2023目前可用的最古老 AL2023 的 JDK 由 Corretto 8 提供。

有关 Amazon Linux 上 Java 的更多信息，请参阅 Java在 AL2。

Python 2.7

> **Note**
>
> AL2023 移除了 Python 2.7，因此任何需要 Python 的操作系统组件都是为了与 Python 3 配合使用而编写的。要继续使用 Amazon Linux 提供并支持的 Python 版本，请将 Python 2 代码转换为 Python 3。

有关 Amazon Linux 上 Python 的更多信息，请参阅 Python在 AL2。

rsyslog-openssl取代 rsyslog-gnutls

该rsyslog-gnutls软件包已在中弃用 AL2，不再存在于中。 AL2023rsyslog-openssl 程序包应能直接替代 rsyslog-gnutls 程序包的所有用途。

网络信息服务（NIS）/yp

网络信息服务 (NIS)，最初称为黄页，或者YP已在中弃用 AL2，不再存在于中 AL2023。这包括以下程序包：ypbind、ypserv 和 yp-tools。与之集成的其他软件包NIS已在中删除了此功能 AL2023。

Amazon VPC 中的多个域名 create-dhcp-options

在 Amazon Linux 2 中，可以在 domain-name 参数中向 create-dhcp-options 传递多个域名，这将导致 /etc/resolv.conf 包含类似 search foo.example.com bar.example.com 的内容。Amazon VPC DHCP 服务器使用 DHCP 选项 15 发送提供的域名列表，该选项仅支持单个域名（参阅 RFC 2132 第 3.17 节）。由于 AL2023 systemd-networkd用于网络配置（如下所示）RFC，因此中 AL2 不存在此意外功能 AL2023

AWS CLI 和 Amazon VPC 文档对此说明如下：“某些 Linux 操作系统接受以空格分隔的多个域名。但是，Windows 以及其他 Linux 操作系统将该值视为单个域，因而会导致意外行为。如果您的 DHCP 选项集与其中实例所运行操作系统将该值视为单个域的 Amazon VPC 关联，请仅指定一个域名。”

在这些系统上 AL2023，例如使用DHCP选项 15（仅允许一个）指定两个域，并且由于域名中的空格字符无效，这将导致空格字符被编码为032，从而导致/etc/resolv.conf包含search foo.exmple.com032bar.example.com。

为支持多个域名，DHCP 服务器应使用 DHCP 选项 119（参阅 RFC 3397 第 2 节）。有关 Amazon VPC DHCP 服务器何时支持此功能的信息，请参阅《Amazon VPC 用户指南》。

Sun RPC中的glibc

in 的实现已Sun RPC在中弃glibc用， AL2 并在中删除。 AL2023如果需要Sun RPC功能，建议客户转而使用该libtirpc库（在 AL2 和中提供 AL2023）。采用 libtirpc 还能使应用程序支持 IPv6。

此变更反映了更广泛社区对上游 glibc 移除此功能的采纳，例如 Fedora 从 glibc 中移除 Sun RPC 接口以及 Gentoo 中的类似变更。

audit 日志中的 OpenSSH 密钥指纹

在生命周期的后期 AL2，在 OpenSSH 包中添加了一个补丁，用于发出用于身份验证的密钥指纹。中不存在此功能 AL2023。

ld.gold 链接器

ld.gold链接器在中可用 AL2，并且已在中删除。 AL2023构建明确引用 gold 链接器的软件的客户应迁移至常规 (ld.bfd) 链接器。

上游 GNU Binutils 的 2.44 版（2025 年 2 月发布）发布说明记录了 ld.gold 的移除：“与我们以往做法不同，在此版本中，binutils-2.44.tar 压缩包不包含 gold 链接器的源代码。这是因为 gold 链接器现已弃用，除非有志愿者站出来愿意继续开发和维护，否则最终将被移除。”

ping6

在中 AL2023，常规ping实用程序原生支持 IPv6，/bin/ping6不再需要分开。中 AL2023，/usr/sbin/ping6是指向/usr/bin/ping可执行文件的符号链接。

这一变化是在更广泛的社区采用提供此功能的较新iputils版本之后进行的，例如 Fedora 中的 Ping IPv6 更改。

ftp 程序包

从 AL2 023 开始 AL2 ，中的ftp软件包在亚马逊 Linux 中不再可用。此决定是我们对安全性、可维护性和现代软件开发实践持续承诺的一部分。作为迁移到 AL2 023 的一部分（或之前），我们建议将对旧版ftp软件包的任何使用迁移到其替代方案之一。

背景

传统的 ftp 程序包在上游已多年未得到积极维护。其源代码的最后一次重要更新发生在 2000 年代初期，且原始源代码存储库已不可用。尽管一些 Linux 发行版提供了安全漏洞补丁，但该代码库基本上仍处于无人维护状态。

建议的替代方案

AL2023 为 FTP 功能提供了几种现代、积极维护的替代方案：

lftp（在 023 AL2 和 AL2 023 中可用）

一个复杂的文件传输功能程序，支持 FTP、HTTP、SFTP 和其他协议。它比传统的 ftp 客户端提供更多功能，并且得到积极维护。

安装命令：dnf install lftp。

curl（在 023 AL2 和 AL2 023 中可用）

一款用于传输数据的多功能命令行工具 URLs，支持 FTP、FTPS、HTTP、HTTPS 和许多其他协议。

默认情况下，在 AL2 023 中可通过curl-minimal软件包获得。如需更广泛的协议支持，可选择使用 curl-full 升级至 dnf swap curl-minimal curl-full。

`wget`（在 023 AL2 和 AL2 023 中可用）

一个用于从网络下载文件的非交互式命令行实用程序，支持 HTTP、HTTPS 和 FTP 协议。

使用以下方式安装：dnf install wget（并非所有 AL2 023 映像都默认安装）

`sftp`（在 023 AL2 和 AL2 023 中可用）

一种通过 SSH 运行的安全文件传输功能协议，提供加密的文件传输。

作为 OpenSSH 程序包的组成部分默认提供。

迁移注意事项

如果您的应用程序或脚本依赖传统的 `ftp` 客户端，请考虑以下迁移方法：

1. 更新脚本以使用现代替代方案：修改您的脚本以使用 `lftp`、`curl`、`wget`、或 `sftp` 替代传统的 `ftp` 客户端。
2. 检查程序包依赖关系：某些应用程序可能在其程序包元数据中将 `ftp` 程序包列为依赖项，即使它们内部早已迁移使用现代协议。在这些情况下，尽管软件`ftp`包中缺少 AL2 023，但应用程序仍可能在 023 上正常运行。`/usr/bin/ftp`请检查应用程序的实际要求，而非仅依赖声明的依赖关系。
3. 更新应用程序依赖关系：对于您维护的仍声明依赖 `ftp` 程序包但实际并未使用的应用程序，请更新程序包元数据以移除此不必要的依赖项。

安全考虑因素

FTP 协议以明文形式传输数据，包括身份验证凭证。对于安全敏感型应用程序，我们强烈建议使用加密替代方案（例如 SFTP 或 HTTPS），这些方案均受推荐替代工具的支持。

为迁移到 AL2 023 做好准备

在继续使用的 AL2同时，你可以准备移至 AL2 023。

主题

查看 AL2 023 年的变更清单

AL2023 文档包含此后 AL2实施的更改的详细列表。此信息位于 “比较 AL2 和 AL2 023” 部分。在 AL2023 的 Package 变更部分中还提供了软件包变更的完整列表。

AL2023 不包括`amazon-linux-extras`。相反，它提供了命名空间的软件包，其中提供了多个版本。由于许多软件包是在 AL2 023 中更新的，因此 AL2 023 中的基本版本可能会晚于您从中获取的版本。`amazon-linux-extras`

> **Note**
>
> 我们建议您不要跑步`amazon-linux-extras`，因为它已停产。

查看文档中的这些部分后，您可以确定 AL2 023 中是否存在可能需要您调整环境以适应迁移的更改。例如，你可能需要最终将 Python 2.7 脚本迁移到 Python 3。

从`cron`作业迁移到`systemd`计时器

默认情况下，`cron`未在 AL2 023 中安装。您可以将`cron`任务迁移到`systemd`计时器，为迁移到 AL2 023 做准备。 AL2 `systemd`具有许多优点，例如可以更精确地控制计时器的运行时间和改进的日志记录。

AL2 局限性

以下主题涵盖了各种限制 AL2，以及这些限制是否已在新版本的 Amazon Linux 中得到解决。

主题

yum无法验证使用 GPG 子密钥制作的 GPG 签名

中的rpm AL2 软件包管理器版本来自rpm添加了对验证使用 GPG 子密钥制作的包签名的支持。如果您要创建与之兼容的软件包 AL2，则需要确保使用与其中一部分兼容的 GPG 签名密钥 rpm AL2

为了确保现有用户的向后兼容性，in 版本仅 AL2 接收安全向后移植。rpm

AL2023 rpm 中的版本支持验证使用 GPG 子密钥制作的软件包签名。

比较 AL1 和 AL2

以下主题描述了 AL1 和之间的主要区别 AL2。它们还包含有关使用寿命和支持以及套餐变更的信息。

主题

AL1 支持和 EOL

AL1 现在已经结束了。 AL1 自 2020 年 12 月 31 日起终止了标准支持，并且在 2023 年 12 月 31 日之前一直处于维护支持阶段。

我们建议升级到最新的亚马逊 Linux 版本。

支持 AWS Graviton 处理器

AL2 引入了对 Graviton 处理器的支持。 AL2023 针对 Graviton 处理器进行了进一步优化。

systemd 取代 **upstart** 作为 **init** 系统

在中 AL2，`systemd`替换`upstart`为`init`系统。

Python 2.6 和 2.7 被 Python 3 所取代

尽管在 2018.03 版本中将 Python 2.6 AL1 标记为 EOL，但这些软件包仍在存储库中需要安装。 AL2 Python 2.7 是最早支持的 Python 版本。

AL2023 完成了向 Python 3 的过渡，存储库中不包含 Python 2.x 版本。

比较 AL1 和上安装的软件包 AL2 AMIs

程序包	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-插件		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

程序包	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

程序包	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
短划线	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

程序包	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	

程序包	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
文件	5.37	5.11
file-libs	5.37	5.11
filesystem	2.4.30	3.2
findutils	4.4.2	4.5.11

程序包	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

程序包	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	

程序包	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	

程序包	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	

程序包	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

程序包	AL1 AMI	AL2 AMI
libcom_err	1.43.5	1.42.9
libconfig		1.4.9
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	2.1.1	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

程序包	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	flink-client	
libnl3		3.2.28
libnl3-cli		3.2.28
libpath_utils		0.2.1

程序包	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

程序包	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	

程序包	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

程序包	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	

程序包	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

程序包	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

程序包	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

程序包	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_匹配主机名		3.5.0.1
python-cffi		1.6.0
python-chardet		2.2.1
python-configobj		4.7.2
python-daemon		1.6

程序包	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

程序包	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	

程序包	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	

程序包	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	

程序包	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
配额	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

程序包	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

程序包	AL1 AMI	AL2 AMI
selinux-policy-targeted		3.13.1
sendmail	8.14.4	
setserial	2.17	2.17
设置	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

程序包	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
时间	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

程序包	AL1 AMI	AL2 AMI
ustr	1.0.4	1.0.4
util-linux	2.23.2	2.30.2
vim-common	9.0.1712	9.0.2081
vim-data	9.0.1712	9.0.2081
vim-enhanced	9.0.1712	9.0.2081
vim-filesystem	9.0.1712	9.0.2081
vim-minimal	9.0.1712	9.0.2081
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	9.0.2081
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

程序包	AL1 AMI	AL2 AMI
yum-langpacks		0.4.2
yum-metadata-parser	flink-client	flink-client
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

比较安装在容器上的软件包 AL1 和 AL2 基础容器镜像

程序包	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

程序包	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	

程序包	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

程序包	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

程序包	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

程序包	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

程序包	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
设置	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		9.0.2081
vim-minimal		9.0.2081
xz-libs	5.2.2	5.2.2
yum	3.4.3	3.4.3
yum-metadata-parser	flink-client	flink-client
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

亚马逊 EC2 上的 AL2

> **Note**
>
> AL2 不再是亚马逊 Linux 的当前版本。AL2023 是 AL2 的继任者。AL2023 有关更多信息，请参阅 AL2023 用户指南中的比较 AL2 和 AL2023 以及 AL2023 中的 Package 变更列表。AL2023 AL2023 AL2023

主题

使用 AL2 AMI 启动亚马逊 EC2 实例

您可以使用 AL2 AMI 启动亚马逊 EC2 实例。有关更多信息，请参阅步骤 1：启动实例。

使用 Systems Manager 查找最新的 AL2 AMI

Amazon EC2 为由维护的公有 AMI 提供 AWS Systems Manager 公共参数 AWS ，您可以在启动实例时使用这些参数。例如，该 EC2-provided 参数在所有区域`/aws/service/ami-amazon-linux-latest/amzn2-ami-kernel-default-hvm-x86_64-gp2`都可用，并且始终指向给定区域中最新版本的 AL2 AMI。

要使用最新的 AL2023 AMI AWS Systems Manager，请参阅 AL2023 入门。AL2023 AL2023

在以下路径中提供 Amazon EC2 AMI 公有参数：

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

您可以通过运行以下 AWS CLI 命令来查看当前 AWS 区域中所有 Amazon Linux AMI 的列表。

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

使用公有参数启动实例

以下示例使用 EC2-provided 公共参数使用最新的 AL2 AMI 启动`m5.xlarge`实例。

要在命令中指定参数，请使用以下语法：`resolve:ssm:`*`public-parameter`*，其中 `resolve:ssm` 是标准前缀，`public-parameter` 是公有参数的路径和名称。

在本示例中，不包括 `--count` 和 `--security-group` 参数。对于 `--count`，默认为 1。如有默认 VPC 和默认安全组，则将使用它们。

```
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
```

有关更多信息，请参阅AWS Systems Manager 用户指南中的使用公共参数。

了解亚马逊 Linux 2 AMI 的名称

亚马逊 Linux 2 AMI 名称使用以下命名方案：

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

- 最小 AMI 附带一组最小化的预装软件包，以减小图像大小。
- 内核版本决定了相应的 AMI 上预安装的内核版本：
  - `kernel-5.10`选择 Linux 内核版本 5.10。这是 AL2 的推荐内核版本。
  - `kernel-default`为 AL2 选择推荐的默认内核。它是内核 5.10 的别名。

- `kernel-4.14`选择 Linux 内核版本 4.14。这只是为了兼容较早的 AMI 版本而提供的。请勿使用此版本启动新实例。预计此 AMI 将不再受支持。
- 存在一组特殊的 AMI 名称，但不引用特定的内核。这些 AMI 是内核 4.14 的别名。 提供这些 AMI 只是为了与较早的 AMI 版本兼容。请勿使用此 AMI 名称启动新实例。预计这些 AMI 的内核会更新。

- x86_64/aarch64决定要在哪个 CPU 平台上运行 AMI。对于基于英特尔和 AMD 的 EC2 实例，请选择 x86_64。为 EC2 Graviton 实例选择 aarch64。
- ebs/gp2确定用于为相应 AMI 提供服务的 EBS 卷类型。请参阅 EBS 卷类型以供参考。请务必选择 gp2。

连接亚马逊 EC2 实例

您可以通过多种方式连接您的 Amazon Linux 实例，包括 SSH 和 EC2 In AWS Systems Manager Session Manager stance Connect。有关更多信息，请参阅《Amazon EC2 用户指南》中的连接到 Linux 实例。

SSH 用户和 sudo

默认情况下，亚马逊 Linux 不允许`root`使用远程安全外壳 (SSH)。此外，禁用密码身份验证以防止暴力攻击。要在 Amazon Linux 实例上启用 SSH 登录，您必须在实例启动时为其提供密钥对。您还必须设置用于启动实例的安全组以允许 SSH 访问。默认情况下，唯一可以使用 SSH 远程登录的账户是`ec2-user`。此账户也有sudo权限。如果您启用远程`root`登录，请注意它不如依赖密钥对和辅助用户安全。

AL2 AMI 启动模式

AL2 AMI 没有设置启动模式参数。从 AL2 AMI 启动的实例遵循实例类型的默认启动模式值。有关更多信息，请参阅 Amazon EC2 用户指南中的启动模式。

程序包存储库

此信息适用于 AL2。有关 AL2023 的信息，请参阅亚马逊 Linux 2023 用户指南中的在 AL2023 中管理软件包和操作系统更新。AL2023 AL2023

AL2 和 AL1 旨在与每个 Amazon EC AWS 2 区域托管的在线软件包存储库一起使用。这些存储库在所有区域中提供，可使用 yum 更新工具进行访问。通过在每个区域托管存储库，我们可以快速部署更新，不会产生任何数据传输费。

> **Important**
>
> AL1 的最新版本已于 2023 年 12 月 31 日停产，自 2024 年 1 月 1 日起将不会收到任何安全更新或错误修复。有关更多信息，请参阅 Amazon Linux AMI 生命周期终止。

如果您不需要为实例保留数据或自定义设置，则可以使用当前的 AL2 AMI 启动新实例。如果您确实需要保留实例的数据或自定义设置，则可以通过 Amazon Linux 软件包存储库维护这些实例。这些存储库包含所有更新后的程序包。您可以选择将这些更新应用到正在运行的实例中。即使发布了新版本，AMI 和更新包的早期版本仍可继续使用。

> **Note**
>
> 要在 Amazon EC2 实例上更新和安装无需访问互联网的软件包，请参阅如何在运行 AL1、AL2 或 AL2023 的 Amazon EC2 实例上更新 yum 或在没有互联网访问的情况下安装软件包？AL2023

要安装程序包，请使用以下命令：

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

如果您发现 Amazon Linux 不包含您需要的应用程序，您可以直接在 Amazon Linux 实例上安装该应用程序。Amazon Linux 使用 RPM 和yum软件包管理，这可能是安装新应用程序的最直接方法。您应该首先查看我们的中央 Amazon Linux 存储库，确定其中是否有您需要应用程序，因为许多应用程序在那里都可以找到。然后，您可以将这些应用程序添加到您的 Amazon Linux 实例中。

要将应用程序上传到正在运行的 Amazon Linux 实例，请使用 scp 或 sftp，然后通过登录实例来配置应用程序。您还可以使用内置 cloud-init 程序包中的 PACKAGE_SETUP 操作，在实例启动时，上传应用程序。有关更多信息，请参阅在 AL2 上使用云初始化。

安全更新

安全更新是使用软件包存储库提供的。安全更新和更新后的 AMI 安全警报均在 Amazon Linux 安全中心发布。有关 AWS 安全策略的更多信息，或要报告安全问题，请访问 AWS 云安全。

AL1 和 AL2 配置为在启动时下载并安装关键或重要的安全更新。此配置中不包括内核更新。

在 AL2023 中，与 AL1 和 AL2 相比，此配置发生了变化。AL2023 有关 AL2023 安全更新的更多信息，请参阅亚马逊 Linux 2023 用户指南中的安全更新和功能。AL2023

我们建议您在启动后针对您的用例进行必要的更新。例如，您可能希望在启动时应用所有更新（而不仅仅是安全更新），或者评估每个更新并仅应用适用于您的系统的更新。这将使用以下 cloud-init 设置来进行控制：repo_upgrade。下方 cloud-init 配置片段显示了如何修改传递到实例初始化用户数据文本中的设置：

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

repo_upgrade 的可能值如下所示：

critical

应用未完成的关键安全更新。

important

应用未完成的关键和重要安全更新。

medium

应用未完成的关键、重要和中等安全更新。

low

应用所有未完成的安全更新，包括具有低严重性的安全更新。

security

应用 Amazon 标记为安全更新的明显关键或重要更新。

bugfix

应用 Amazon 标记为缺陷修正的更新。缺陷修正是一组较大的更新，其中包括安全更新和针对各种其他小漏洞的修正更新。

all

应用全部适用更新 (不论类别)。

none

实例启动时不应用任何更新。

> **备注**
>
> 亚马逊 Linux 不会将任何更新标记为`bugfix`。要应用来自亚马逊 Linux 的与安全无关的更新，请使用`repo_upgrade: all`。

`repo_upgrade` 的默认设置是安全的。也就是说，如果您未在用户数据中指定其他值，在默认情况下，Amazon Linux 会在启动时执行针对所有已安装程序包的安全升级。Amazon Linux 还会使用 `/etc/motd` 文件列出登录时可用更新的数量，通知您已安装程序包的任何更新。要安装这些更新，您需要在实例上运行 sudo yum upgrade。

存储库配置

对于 AL1 和 AL2，AMI 是创建 AMI 时可用包的快照，但安全更新除外。任何不在原始 AMI 上但在运行时安装的软件包都将是可用的最新版本。要获取可用于 AL2 的最新软件包，请运行yum update -y。

> **故障排除技巧**
>
> 如果您在 nano 实例类型上运行 yum update 时遇到 `cannot allocate memory` 错误（例如 `t3.nano`），则可能需要分配交换空间才能启用更新。

对于 AL2023，与 AL1 和 AL2 相比，存储库配置发生了变化。AL2023 有关 AL2023 存储库的更多信息，请参阅管理程序包和操作系统更新。

AL2023 之前的版本已配置为持续提供更新，以确保 Amazon Linux 次要版本的滚动更新，这也称为滚动发布。作为最佳实践，我们建议您将您的 AMI 更新为最新的可用的 AMI，而不是启动旧的 AMI 并应用更新。

In-place 不支持在主要 Amazon Linux 版本之间进行升级，例如从 AL1 升级到 AL2 或从 AL2 升级到 AL2023。AL2023 有关更多信息，请参阅 Amazon Linux 可用性。

在 AL2 上使用云初始化

cloud-init 程序包是由 Canonical 构建的开源应用程序，用于在云计算环境（例如 Amazon EC2）中引导 Linux 映像。Amazon Linux 包含 cloud-init 的自定义版本。这允许您指定实例在启动时应执行的操作。启动实例时，您可以通过用户数据字段将需要的操作传递到 cloud-init。这意味着，您可以将通用

AMI 用于许多使用案例，并在启动时进行动态配置。Amazon Linux 还使用 cloud-init 来执行 ec2-user 账户的初始配置。

有关更多信息，请参阅 cloud-init 文档。

Amazon Linux 使用在 `/etc/cloud/cloud.cfg.d` 和 `/etc/cloud/cloud.cfg` 中发现的 cloud-init 操作。您可以在 `/etc/cloud/cloud.cfg.d` 中创建自己的 cloud-init 操作文件。此目录中的所有文件均由 cloud-init 读取。它们是按词典顺序进行读取的，并且文件随后将覆盖之前文件中的值。

cloud-init 程序包将在启动时对实例执行这些 (以及其他) 常见配置任务：

- 设置默认区域设置。
- 设置主机名。
- 解析并处理用户数据。
- 生成主机私有 SSH 密钥。
- 将用户的公有 SSH 密钥添加到 `.ssh/authorized_keys`，以便于登录和管理。
- 准备存储库以进行程序包管理。
- 处理用户数据中定义的软件包操作。
- 运行在用户数据中找到的用户脚本。
- 装载实例存储卷 (如果适用)。
  - 默认情况下，`ephemeral0` 实例存储卷装载在 `/media/ephemeral0` (如果它存在且包含有效的文件系统；否则将不会安装)。
  - 默认情况下，将装载与实例关联的所有交换卷 (仅适用于 `m1.small` 和 `c1.medium` 实例类型)。
  - 您可以使用以下 cloud-init 指令覆盖默认实例存储卷安装：

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

    有关对安装的更多控制，请参阅 cloud-init 文档中的安装。
  - 在实例启动时，不会格式化支持 TRIM 的实例存储卷，因此，您必须先对这些卷进行分区和格式化，然后才能装载它们。有关更多信息，请参阅实例存储卷TRIM支持。您可以在启动时使用 `disk_setup` 模块对您的实例存储卷进行分区和格式化。有关更多信息，请参阅 cloud-init 文档中的磁盘设置。

支持的用户数据格式

cloud-init 软件包支持多种格式的用户数据处理：

- Gzip
  - 如果用户数据采用 gzip 压缩，则 cloud-init 会解压缩数据并对其进行适当的处理。
- MIME 多部分内容型
  - 使用 MIME 多部分内容型文件，您可以指定多种数据类型。例如，您可以同时指定用户数据脚本和云配置类型。如果多部分内容型文件的格式是受支持的格式，则 cloud-init 可以处理它的各部分内容。
- Base64 解码
  - 如果用户数据采用 base64 编码，则 cloud-init 将决定它能否将解码后的数据理解为支持的类型之一。如果它能理解解码后的数据，则会解码数据，并进行适当处理。如果不能，它将完整地返回 base64 数据。
- 用户数据脚本
  - 开头为 `#!` 或 `Content-Type: text/x-shellscript`。
  - 该脚本由 `/etc/init.d/cloud-init-user-scripts` 在首轮启动过程中运行。此操作会在启动过程的后期发生 (即执行初始配置操作后)。
- 包含文件
  - 开头为 `#include` 或 `Content-Type: text/x-include-url`。
  - 此内容是一个包含文件。该文件包含一个 URL 列表，每行一个 URL。系统会读取每个 URL，其内容会通过此相同规则集验证。从 URL 读取的内容可以是 gzip 压缩的 MIME-multi-part，也可以是纯文本。
- 云配置数据
  - 开头为 `#cloud-config` 或 `Content-Type: text/cloud-config`。
  - 此内容是云配置数据。
- Upstart 作业（AL2 不支持）
  - 开头为 `#upstart-job` 或 `Content-Type: text/upstart-job`。
  - 这些内容存储在中的文件中`/etc/init`，新贵会像处理其他暴发户一样消耗这些内容。
- Cloud boothook
  - 开头为 `#cloud-boothook` 或 `Content-Type: text/cloud-boothook`。
  - 此内容为 boothook 数据。它存储在 `/var/lib/cloud` 下的一个文件中并会立即运行。

- 这是最早可用的 hook。尚无仅供运行一次的机制。boothook 必须自行解决此问题。它的环境变量 `INSTANCE_ID` 中包含实例 ID。可使用此变量来提供一组一个实例可用一次的 boothook 数据。

配置 AL2 实例

成功启动并登录 AL2 实例后，您可以对其进行更改。可以通过许多不同方式配置实例以满足特定应用程序的需求。下面是一些可帮助您入门的常见任务。

目录

常见配置方案

Amazon Linux 的基本发行版包含基本服务器操作所需的软件包和实用工具。但是，各种软件存储库还提供许多软件包，还有更多软件包可供您从源代码进行构建。有关从这些位置安装和构建软件的更多信息，请参阅管理 AL2 实例上的软件。

Amazon Linux 实例预配置有 `ec2-user`，但是，您可能需要添加没有超级用户权限的其他用户。有关添加和删除用户的更多信息，请参阅 Amazon EC2 用户指南中的管理您的Linux实例上的用户。

如果您自己有注册了域名的网络，则可以更改实例的主机名，将它自身标识为该域名的一部分。您还可以在不更改主机名设置的情况下更改系统提示，以显示更有意义的名称。有关更多信息，请参阅更改 AL2 实例的主机名。您可以将实例配置成使用动态 DNS 服务提供商。有关更多信息，请参阅在您的 AL2 实例上设置动态 DNS。

当您在 Amazon EC2 中启动实例时，可以选择将用户数据传递到可用于执行常见配置任务甚至在实例启动后运行脚本的实例。您可以将两类用户数据传递到 Amazon EC2：cloud-init 指令和 Shell 脚本。有关更多信息，请参阅 Amazon EC2 用户指南中的启动时在Linux实例上运行命令。

管理 AL2 实例上的软件

Amazon Linux 的基本发行版包含基本服务器操作所需的软件包和实用工具。

此信息适用于 AL2。有关 AL2023 的信息，请参阅亚马逊 Linux 2023 用户指南中的在 AL2023 中管理软件包和操作系统更新。AL2023 AL2023

使软件保持最新非常重要。Linux 发行版中的许多程序包会经常更新，以修复错误、添加功能，以及防止安全漏洞。有关更多信息，请参阅 更新 AL2 实例上的实例软件。

默认情况下，AL2 实例在启用以下存储库的情况下启动：

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

虽然这些存储库中有许多由更新过的软件包 AWS，但另一个存储库中可能包含您要安装的软件包。有关更多信息，请参阅 在 AL2 实例上添加存储库。有关在启用的存储库中查找和安装程序包的帮助，请参阅 在 AL2 实例上查找并安装软件包。

并非所有软件均可在存储库中存储的软件包中获得；有些软件必须在实例上从其源代码进行编译。有关更多信息，请参阅 准备在 AL2 实例上编译软件。

AL2 实例使用 yum 包管理器管理其软件。yum 程序包管理器可安装、删除和更新软件，以及管理每个包的所有依赖关系。

内容

更新 AL2 实例上的实例软件

使软件保持最新非常重要。Linux 发行版中的程序包会经常更新，以修复错误、添加功能，以及防止安全漏洞。当您首次启动并连接到 Amazon Linux 实例时，您可能会看到出于安全目的要求您更新软件包的消息。本节介绍如何更新整个系统或仅更新单个程序包。

此信息适用于 AL2。有关 AL2023 的信息，请参阅亚马逊 Linux 2023 用户指南中的在 AL2023 中管理软件包和操作系统更新。AL2023 AL2023

有关 AL2 变更和更新的信息，请参阅 AL2 发行说明。

有关对 AL2023 的更改和更新的信息，请参阅 AL2023 版本注释。

> **Important**
>
> 如果您在 IPv6-only 子网中启动了使用 Amazon Linux 2 AMI 的 EC2 实例，则必须连接到该实例并运行`sudo amazon-linux-https disable`。这样，您的 AL2 实例才可以使用 http 补丁服务通过 IPv6 连接到 S3 中的 yum 存储库。

更新 AL2 实例上的所有软件包

1. （可选）在 Shell 窗口中启动 screen 会话。有时您可能会遇到网络中断，这样会断开到实例的 SSH 连接。如果在较长的软件更新期间发生这种情况，实例处于混乱、但可恢复的状态。即使连接中断，通过 screen 会话也可继续运行更新，您稍后可重新连接到此会话，不会有问题。

   a. 执行 screen 命令以开始会话。

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

   b. 如果会话中断，请再次登录实例并列出可用屏幕。

   ```
   [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 命令和前一命令的进程 ID 重新连接到屏幕。

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

   d. 使用 screen 完成操作后，使用 exit 命令关闭会话。

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

2. 运行 yum update 命令。您可以选择添加 `--security` 标记，这样仅应用安全更新。

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

3. 查看所列的程序包，输入 **y** 并按 Enter 接受更新。更新系统上的所有程序包可能需要几分钟。yum 输出显示更新运行状态。
4. （可选）重启您的实例，以确保您使用的是更新中的最新软件包和库；只有在重启后才会加载内核更新。更新任何 `glibc` 库后也应进行重启。对于用来控制服务的程序包的更新，重新启动服务可能就足以使更新生效，但系统重启可确保所有之前的程序包和库更新都是完整的。

更新 AL2 实例上的单个软件包

使用此过程可更新单个程序包 (及其依赖关系)，而非整个系统。

1. 使用要更新的程序包的名称运行 yum update 命令。

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

2. 查看所列的程序包信息，输入 **y** 并按 Enter 接受更新。如果存在必须解析的程序包依赖关系，有时会列出多个数据包。yum 输出显示更新运行状态。
3. （可选）重启您的实例，以确保您使用的是更新中的最新软件包和库；只有在重启后才会加载内核更新。更新任何 `glibc` 库后也应进行重启。对于用来控制服务的程序包的更新，重新启动服务可能就足以使更新生效，但系统重启可确保所有之前的程序包和库更新都是完整的。

在 AL2 实例上添加存储库

此信息适用于 AL2。有关 AL2023 的信息，请参阅亚马逊 Linux 2023 用户指南中的通过版本控制存储库在 AL2023 上进行确定性升级。AL2023 AL2023

默认情况下，AL2 实例在启用以下存储库的情况下启动：

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

尽管在 Amazon Web Services 更新的这些存储库中有许多程序包，但是您需要安装的程序包可能在其他存储库中。

要使用 yum 从不同存储库安装程序包，您需要将存储库信息添加到 `/etc/yum.conf` 文件中，或者添加到 *`repository`*`.repo` 目录中它自己的 `/etc/yum.repos.d` 文件中。您可以手动执行此操作，但大多数 yum 存储库在其存储库 URL 提供各自的 *`repository`*`.repo` 文件。

确定已安装的 yum 存储库

使用以下命令列出已安装的 yum 存储库：

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

输出结果会列出已安装的存储库，并报告每个存储库的状态。启用的存储库会显示其中包含的程序包数量。

将 yum 存储库添加到 /etc/yum.repos.d

1. 查找 .repo 文件的位置。这随要添加的存储库而异。在本示例中，.repo 文件位于 https://www.*example*.com/*repository*.repo。
2. 使用 yum-config-manager 命令添加存储库。

```
[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
```

安装存储库后，必须按照以下过程启用存储库。

在中启用 yum 存储库 /etc/yum.repos.d

使用带 yum-config-manager 标志的 `--enable` *repository* 命令。以下命令从 Fedora 项目启用 Extra Packages for Enterprise Linux (EPEL) 存储库。默认情况下，此存储库显示在 Amazon Linux AMI 实例上的 /etc/yum.repos.d 中，但未启用。

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

有关更多信息以及要下载此软件包的最新版本，请参阅https://fedoraproject.org/wiki/EPEL。

在 AL2 实例上查找并安装软件包

您可以使用软件包管理工具来查找和安装软件包。在 Amazon Linux 2 中，默认的软件包管理工具是 YUM。在 AL2023 中，默认的软件包管理工具是 DNF。有关更多信息，请参阅《亚马逊 Linux 2023 用户指南》中的 Pack age 管理工具。

在 AL2 实例上查找软件包

您可以使用 yum search 命令搜索在您配置的存储库中可用的程序包的描述。如果不知道要安装的程序包的确切名称，这尤其有帮助。只需将关键字搜索附加到该命令；对于多字词搜索，请使用引号括起搜索查询。

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

下面是示例输出。

```
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
```

引号中的多个字词搜索查询仅返回符合确切查询的结果。如果您没有看到需要的程序包，请将搜索简化为一个关键字，然后扫描结果。您还可以尝试使用关键字同义词来扩大搜索范围。

有关 AL2 软件包的更多信息，请参阅以下内容：

- AL2 额外内容库
- 程序包存储库

在 AL2 实例上安装软件包

在 AL2 中，yum 软件包管理工具会在所有启用的存储库中搜索不同的软件包，并处理软件安装过程中的任何依赖关系。有关在 AL2023 中安装软件包的信息，请参阅亚马逊 Linux 2023 用户指南中的管理软件包和操作系统更新。AL2023

从存储库安装软件包

使用yum install ***package***命令，*package*替换为要安装的软件的名称。例如，若要安装 links 基于文本的 Web 浏览器，请输入以下命令。

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

安装您已下载的 RPM 软件包文件

您还可使用 yum install 安装您已经从互联网下载的 RPM 程序包文件。为此，将 RPM 文件的路径名称而不是存储库程序包名称附加到安装命令。

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

列出已安装软件包

要查看实例上已安装的软件包的列表，请使用以下命令。

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

准备在 AL2 实例上编译软件

Open-source Internet 上有尚未预编译的软件，可供从软件包存储库中下载。您可能最终会发现需要您亲自从源代码编译的软件包。为了使您的系统能够在 AL2 和 Amazon Linux 中编译软件，您需要安装多种开发工具，例如makegcc、和autoconf。

因为软件编译不是每个 Amazon EC2 实例都需要的任务，所以在默认情况下不安装这些工具，不过，称为“开发工具”的程序包组中包含这些工具，而这个程序包组可通过 yum groupinstall 命令方便地添加到实例。

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

软件源代码包通常可以作为压缩存档文件（称为 tarball http://sourceforge.net/）下载（从https://github.com/和等网站）。这些 tarball 的文件扩展名通常为 `.tar.gz`。您可以使用 tar 命令解压缩这些存档。

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

将源代码包解压并解档后，应在源代码目录中查找 README 或 INSTALL 文件，这些文件包含有关编译和安装源代码的进一步说明。

检索 Amazon Linux 程序包的源代码

Amazon Web Services 提供所维护的程序包的源代码。您可以使用 yumdownloader --source 命令下载已安装的任何程序包的源代码。

运行yumdownloader --source *package*命令下载的源代码*package*。例如，若要下载 htop 程序包的源代码，请输入以下命令。

```
[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
```

源 RPM 的位置位于您运行命令的目录中。

您的 Amazon EC2 AL2 实例的处理器状态控制

C-states 控制内核空闲时可以进入的睡眠级别。 C-states 从 C0（内核完全清醒并正在执行指令的最浅状态）开始编号，然后转到 C6（内核断电的最深空闲状态）。

P-states 控制内核所需的性能（以 CPU 频率计）。 P-states 从 P0（允许内核尽可能使用英特尔睿频加速技术提高频率的最高性能设置）开始编号，它们从 P1（要求最大基准频率的）到 P15（尽可能低的频率）。 P-state

您可能需要更改 C-state 或 P-state 设置以提高处理器性能一致性、减少延迟或针对特定工作负载调整您的实例。默认值 C-state 和 P-state 设置可提供最高性能，这对于大多数工作负载来说是最佳的。但

是，如果您的应用程序将受益于以更高的单核或双核频率为代价降低延迟，或者在较低频率下获得稳定的性能，而不是突发的 Turbo Boost 频率，则可以考虑尝试这些实例可用的 C-state 或 P-state 设置。

有关允许操作系统控制处理器 C-states 的 Amazon EC2 实例类型的信息 P-states，请参阅 Amazon EC2 用户指南中的 Amazon EC2 实例的处理器状态控制。

以下部分介绍了不同的处理器状态配置以及如何监控配置效果。这些程序是为亚马逊 Linux 编写的，适用于亚马逊 Linux；但是，它们也可能适用于其他 Linux 内核版本为 3.9 或更高版本的 Linux 发行版。

> **Note**
>
> 本页上的示例使用以下内容：
>
> - 显示处理器频率和 C-state 信息的turbostat实用程序。默认情况下，turbostat 实用程序在 Amazon Linux 上提供。
> - 模拟工作负载的 stress 命令。若要安装 stress，首先通过运行 sudo amazon-linux-extras install epel 启用 EPEL 存储库，然后运行 sudo yum install -y stress。
>
> 如果输出未显示 C-state 信息，请在命令 (sudo turbostat --debug stress ***<options>***) 中包含该--debug选项。

内容

具有最大睿频加速频率的最高性能

这是 Amazon Linux AMI 的默认处理器状态控制配置，推荐大多数工作负载使用。此配置可提供最高性能，且变化更少。允许非活动核心进入深层睡眠状态可提供单核或双核进程所需的热空间，以达到最大睿频加速潜能。

以下示例显示了具有两个有效执行工作且达到其最大处理器睿频加速频率的核心的 `c4.8xlarge` 实例。

```
[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
```

在此示例中，vCPU 21 和 vCPU 28 均以其最大睿频加速频率运行，因为其他内核已进入 C6 睡眠状态以节省性能，并为正在工作的内核提供性能和热空间。vCPU 3 和 vCPU 10（分别与 vCPU 21 和 vCPU 28 共享一个处理器内核）均处于等待指令的 C1 状态。

在以下示例中，所有 18 个核心均在有效执行工作，因此没有达到最大睿频加速频率的空间，但这些核心都在以 3.2 GHz 的“所有核心睿频加速”速度运行。

```
[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
...
```

通过深度限制实现高性能和低延迟 C-states

C-states 控制核心处于非活动状态时可能进入的睡眠水平。您可能需要控制 C-states 以调整系统的延迟与性能。将核心置于睡眠状态需要时间，尽管睡眠中的核心可为其他核心提供更多空间以加速至更高频率，但该睡眠中的核心也需要时间来重新唤醒并执行工作。例如，如果某个负责处理网络数据包中断的核心处于睡眠状态，那么在处理此类中断时可能会出现延迟。您可以将系统配置为不使用更深的空间 C-states，这样可以减少处理器的反应延迟，但这反过来也会减少其他内核可用于睿频加速的余量。

禁用深层睡眠状态的常见情形是 Redis 数据库应用程序，该应用程序将数据库存储在系统内存中，以实现最快速的查询响应。

限制 AL2 的深度睡眠状态

1. 使用所选编辑器打开 `/etc/default/grub` 文件。

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

2. 编辑该`GRUB_CMDLINE_LINUX_DEFAULT`行并添加`intel_idle.max_cstate=1`和`processor.max_cstate=1`选项，`C1`将其设置 C-state 为空闲内核的最深度。

   ```
   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
   ```

   该`intel_idle.max_cstate=1`选项配置 Intel-based 实例的 C-state 限制，该`processor.max_cstate=1`选项配置实例的 C-state 限制。 AMD-based 您可以放心将这两个选项添加到配置中。这可实现通过单个配置在英特尔和 AMD 处理器上设置所需的行为。

3. 保存文件并退出您的编辑器。
4. 运行以下命令重新构建启动配置。

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

5. 重启实例以启用新的内核选项。

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

限制 Amazon Linux AMI 上的深层睡眠状态

1. 使用所选编辑器打开 /boot/grub/grub.conf 文件。

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

2. 编辑第一个条目的kernel行，然后添加intel_idle.max_cstate=1和processor.max_cstate=1选项以设置C1 C-state 为空闲内核的最深度。

```
# 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
```

该intel_idle.max_cstate=1选项配置 Intel-based 实例的 C-state 限制，该processor.max_cstate=1选项配置实例的 C-state 限制。 AMD-based 您可以放心将这两个选项添加到配置中。这可实现通过单个配置在英特尔和 AMD 处理器上设置所需的行为。

3. 保存文件并退出您的编辑器。
4. 重启实例以启用新的内核选项。

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

以下示例显示的 c4.8xlarge 实例具有两个以“所有核心睿频加速”核心频率有效执行工作的核心。

```
[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
...
```

在此示例中，vCPU 19 和 28 的内核以 3.2 GHz 的频率运行，而其他内核则在等待指令C1 C-state中。尽管工作内核尚未达到其最大睿频加速频率，但非活动内核响应新请求的速度要比更深的内核快得多C6 C-state。

变化最少的基准性能

您可以使用减少处理器频率的变化。 P-states P-states 控制内核所需的性能（以 CPU 频率计）。大多数工作负载在 P0 状态下性能更好，该状态要求采用睿频加速频率。但是，您可能需要调校系统以获得稳定性能而非突发式性能，而突发式性能可能会在启用睿频加速频率后出现。

Intel 高级矢量扩展 (AVX 或 AVX2) 工作负载能够以较低的频率较好地运行，而 AVX 指令也可以使用更多性能。通过禁用睿频加速来以较低的频率运行处理器，可以降低所使用的性能并保持更稳定的速度。有关优化实例配置和 AVX 工作负载的更多信息，请参阅 英特尔公司网站。

CPU 空闲驱动程序控制 P-state。较新一代的 CPU 需要更新与内核级别对应的 CPU 空闲驱动程序，如下所示：

- Linux 内核版本 6.1 及更高版本 — 支持英特尔 Granite Rapids（例如 R8i）
- Linux 内核版本 5.10 及更高版本 — 支持 AMD Milan（例如 m6a）
- Linux 内核版本 5.6 及更高版本 — 支持英特尔 Icelake（例如 M6i）

要检测正在运行的系统的内核是否识别 CPU，请运行以下命令。

```
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
```

如果此命令的输出指示不支持，则建议您升级内核。

本节介绍如何限制深度睡眠状态和禁用 Turbo Boost（通过请求 P1 P-state），以便为这些类型的工作负载提供低延迟和最低的处理器速度变化。

在 AL2 上限制深度睡眠状态并禁用 Turbo Boost

1. 使用所选编辑器打开 `/etc/default/grub` 文件。

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

2. 编辑该`GRUB_CMDLINE_LINUX_DEFAULT`行并添加`intel_idle.max_cstate=1`和`processor.max_cstate=1`选项，`C1`将其设置 C-state 为空闲内核的最深度。

   ```
   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
   ```

   该`intel_idle.max_cstate=1`选项配置 Intel-based 实例的 C-state 限制，该`processor.max_cstate=1`选项配置实例的 C-state 限制。 AMD-based 您可以放心将这两个选项添加到配置中。这可实现通过单个配置在英特尔和 AMD 处理器上设置所需的行为。

3. 保存文件并退出您的编辑器。
4. 运行以下命令重新构建启动配置。

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

5. 重启实例以启用新的内核选项。

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

6. 当您需要P1 P-state 提供的低处理器速度变化时，请运行以下命令以禁用 Turbo Boost。

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

7. 在工作负载完成后，您可以使用以下命令重新启用睿频加速。

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

限制 Amazon Linux AMI 上的深层睡眠状态并禁用睿频加速

1. 使用所选编辑器打开 /boot/grub/grub.conf 文件。

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

2. 编辑第一个条目的kernel行，然后添加intel_idle.max_cstate=1和processor.max_cstate=1选项以设置C1 C-state 为空闲内核的最深度。

```
# 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
```

   该intel_idle.max_cstate=1选项配置 Intel-based 实例的 C-state 限制，该processor.max_cstate=1选项配置实例的 C-state 限制。 AMD-based 您可以放心将这两个选项添加到配置中。这可实现通过单个配置在英特尔和 AMD 处理器上设置所需的行为。

3. 保存文件并退出您的编辑器。
4. 重启实例以启用新的内核选项。

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

5. 当您需要P1 P-state 提供的低处理器速度变化时，请运行以下命令以禁用 Turbo Boost。

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

6. 在工作负载完成后，您可以使用以下命令重新启用睿频加速。

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

以下示例显示的 c4.8xlarge 实例具有两个以基准核心频率有效执行工作的 vCPU，这两个 vCPU 均没有启用睿频加速。

```
[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 的内核正在以 2.9 GHz 的基准处理器速度积极执行工作，所有非活动内核也都以基准速度运行，随时可以接受C1 C-state指令。

I/O AL2 的调度器

I/O 调度器是 Linux 操作系统的一部分，它对请求进行排序和合并，并确定 I/O 请求的处理顺序。

I/O 调度器对诸如磁性硬盘驱动器之类的设备特别有益，因为在这些设备中，寻道时间可能很昂贵，而且最好是合并同地请求。 I/O 调度程序对固态设备和虚拟化环境的影响较小。这是因为对于固态设备而言，顺序访问与随机访问没有区别，而对于虚拟化环境，则由主机提供自定义的调度层。

本主题讨论亚马逊 Linux I/O 计划程序。有关其他 Linux 发行版使用的 I/O调度程序的更多信息，请参阅其各自的文档。

主题

支持的调度器

亚马逊 Linux 支持以下 I/O 调度程序：

- deadline— Deadl in I/O e 调度器对 I/O 请求进行排序，并按最高效的顺序处理请求。它保证了每个 I/O请求的开始时间。它还会将待处理时间过长的 I/O 请求置于更高的优先级。
- cfq— 完全公平队列 (CFQ) I/O 调度程序尝试在进程之间公平分配 I/O资源。它将 I/O 请求排序并插入到每个进程的队列中。
- noop— 无操作 (noop) I/O 调度程序将所有 I/O 请求插入到 FIFO 队列中，然后将它们合并为一个请求。此调度器不会排序任何请求。

默认调度器

无操作 (noop) 是 Amazon Linux 的默认 I/O 调度程序。使用此调度器的原因如下：

- 许多实例类型都使用虚拟化设备，其中底层主机会为实例执行调度。
- 固态设备用于许多实例类型，在这些实例类型中， I/O 调度程序的优势影响较小。
- 它是侵入性最小的 I/O 调度程序，可以根据需要对其进行自定义。

更改调度器

更改 I/O 调度器可以提高或降低性能，具体取决于调度程序是在给定时间内完成更多还是更少的 I/O 请求。这在很大程度上取决于您的工作负载、正在使用的实例类型的生成以及正在访问的设备的类型。如果您更改正在使用的 I/O调度程序，我们建议您使用诸如 iotop 之类的工具来衡量 I/O 性能并确定更改是否对您的用例有利。

您可以使用以下命令查看设备的 I/O 调度程序，该命令nvme0n1用作示例。将以下命令中的 nvme0n1 替换为您实例上的 /sys/block 中列出的设备。

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

要为设备设置 I/O 调度程序，请使用以下命令。

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

例如，要将*xvda*设备的 I/O 调度程序设置为cfq，noop请使用以下命令。

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

更改 AL2 实例的主机名

当您将实例启动到私有 VPC 中时，Amazon EC2 会分配一个来宾操作系统主机名。Amazon EC2 分配的主机名类型取决于您的子网设置。有关 EC2 主机名的更多信息，请参阅 Amazon EC2 用户指南中的 Amazon EC2 实例主机名类型。

配置为使用 IPv4 地址 IP-based 命名的 EC2 实例的典型 Amazon EC2 私有 DNS 名称如下所示：ip-12-34-56-78.us-west-2.compute.internal，其中名称由内部域、服务（在本例中为compute）、区域和某种形式的私有 IPv4 地址组成。当您登录实例时，Shell 提示符处显示此主机名的一部分 (例如，ip-12-34-56-78)。每次停止和重新启动 Amazon EC2 实例时 (除非您使用的是弹性 IP 地址)，公有 IPv4 地址都会改变，而且公有 DNS 名称、系统主机名和 Shell 提示符也会改变。

> **Important**
>
> 此信息适用于 Amazon Linux。有关其他发布版本的信息，请参阅特定于该版本的文档。

更改系统主机名

如果为实例的 IP 地址注册了公用 DNS 名称 (如 webserver.mydomain.com)，则可以设置系统主机名，以便实例将自己标识为该域的一部分。这还会更改 shell 提示符，使其显示此名称的第一部分，而不是由 AWS（例如ip-12-34-56-78）提供的主机名。如果没有注册公用 DNS 名，还是可以更改主机名，但过程略有差异。

为使主机名持续更新，您必须确认 preserve_hostname cloud-init 设置已设为 true。您可以运行以下命令来编辑或添加此设置：

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

如果未列出 preserve_hostname 设置，请在文件末尾添加以下文本行：

```
preserve_hostname: true
```

将系统主机名更改为公用 DNS 名称

如果已注册了公用 DNS 名称，请执行此过程。

1. • 对于 AL2：使用hostnamectl命令设置您的主机名以反映完全限定的域名（例如**webserver.mydomain.com**）。

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

   • 对于 Amazon Linux AMI：在您的实例上，使用常用的文本编辑器打开 /etc/sysconfig/network 配置文件，更改 HOSTNAME 条目以反映完全限定域名（例如 **webserver.mydomain.com**）。

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

2. 重启实例以接受新主机名。

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

   或者，您也可以使用 Amazon EC2 控制台重启（在实例页面上，选择实例，然后依次选择实例状态、重启实例）。

3. 登录实例，验证主机名是否已更新。提示符应显示新主机名（直到第一个“.”），hostname 命令应显示完全限定域名。

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

在无公用 DNS 名称的情况下更改系统主机名

1. • 对于 AL2：使用hostnamectl命令设置您的主机名以反映所需的系统主机名（例如**webserver**）。

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

- 对于 Amazon Linux AMI：在您的实例上，使用常用的文本编辑器打开 /etc/sysconfig/network 配置文件，更改 HOSTNAME 条目以反映所需的系统主机名（例如 **webserver**）。

```
HOSTNAME=webserver.localdomain
```

2. 在您常用的文本编辑器中打开 /etc/hosts 文件，更改以 **127.0.0.1** 开始的条目，以匹配以下示例，替换为您自己的主机名。

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

3. 重启实例以接受新主机名。

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

或者，您也可以使用 Amazon EC2 控制台重启（在实例页面上，选择实例，然后依次选择实例状态、重启实例）。

4. 登录实例，验证主机名是否已更新。提示符应显示新主机名（直到第一个“.”），hostname 命令应显示完全限定域名。

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

您还可以实施更多的编程解决方案，例如指定用户数据以配置您的实例。如果您的实例是自动扩缩组的一部分，您可以使用生命周期挂钩定义用户数据。有关更多信息，请参阅《AWS CloudFormation 用户指南》中的启动时在 Linux 实例上运行命令和用于实例启动的生命周期挂钩。

在不影响主机名的情况下更改 Shell 提示符

如果您不想修改实例的主机名，但希望显示比提供的私有名称（例如**webserver**）更有用的系统名称 AWS（例如ip-12-34-56-78），则可以编辑 shell 提示符配置文件以显示您的系统昵称而不是主机名。

将 Shell 提示符更改为主机别名

1. 在 /etc/profile.d 中创建一个文件，用于将称为 NICKNAME 的环境变量设置为要在 Shell 提示符中显示的值。例如，若要将系统别名设置为 **webserver**，请运行以下命令。

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

2. 在您常/etc/bashrc用的文本编辑器/etc/bash.bashrc（例如或Debian/Ubuntu）中打开 (Red Hatnano) vim 或 () 文件。您需要在编辑器命令中使用 sudo，因为 /etc/bashrc 和 /etc/bash.bashrc 归 root 所有。
3. 编辑文件，将 Shell 提示符变量 (PS1) 更改为显示别名而不是主机名。在 /etc/bashrc 或 /etc/bash.bashrc 中查找以下设置 Shell 提示符的行（为了上下文需要，下面多显示了几行；查找以 ["$PS1" 开头的行）：

```
# 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
```

将该行中的 \h (hostname 的符号) 更改为 NICKNAME 变量的值。

```
# 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. (可选) 要将 Shell 窗口上的标题设置为新别名，请完成以下步骤。
   a. 创建一个名为的文件 /etc/sysconfig/bash-prompt-xterm。

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

   b. 使用以下命令使该文件可执行。

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

   c. 在您常用的文本编辑器（如 nano 或 vim）中打开 /etc/sysconfig/bash-prompt-xterm 文件。您需要在编辑器命令中使用 sudo，因为 /etc/sysconfig/bash-prompt-xterm 归 root 所有。
   d. 将以下行添加到该文件。

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

5. 注销，再重新登录，以接受新别名值。

在其他 Linux 发行版上更改主机名

此页面上的过程仅适用于 Amazon Linux。有关其他 Linux 发行版的更多信息，请参阅其特定文档和下列文章：

- 如何为运行 RHEL 7 或 Centos 7 的私有 Amazon EC2 实例分配静态主机名？

在您的 AL2 实例上设置动态 DNS

当您启动 EC2 实例时，系统会为其分配公有 IP 地址和公有域名系统（DNS）名称，您可以通过互联网访问这些地址和名称。因为 Amazon Web Services 域中有非常多主机，所以这些公用名称必须足够长才能使每个名称保持唯一。典型的 Amazon EC2 公有 DNS 名称如下所示：`ec2-12-34-56-78.us-west-2.compute.amazonaws.com`，其中名称由 Amazon Web Services 域、服务（在本例中为`compute`）AWS 区域、和一种形式的公有 IP 地址组成。

动态 DNS 服务在其域区域中提供自定义主机名，这些主机名便于记忆，也与主机的使用案例更为相关。其中一些服务也是免费的。您可以对 Amazon EC2 使用动态 DNS 提供商，可以将实例配置为每次实例启动时都更新与公用 DNS 名称关联的 IP 地址。有许多不同的提供商可以选择，本指南不介绍有关如何选择提供商以及如何向它们注册名称的具体详细信息。

对 Amazon EC2 使用动态 DNS

1. 向动态 DNS 服务提供商注册并利用其服务注册公用 DNS 名称。此过程使用 noip 提供的免费服务。 com/free举个例子。
2. 配置动态 DNS 更新客户端。有了动态 DNS 服务提供商并且使用其服务注册了公用 DNS 名称后，将 DNS 名称指向实例的 IP 地址。很多提供商（包括 noip.com）允许您从您在其网站上的账户页手动执行此操作，不过很多也支持软件更新客户端。如果更新客户端正在 EC2 实例上运行，则每次 IP 地址更改（如关机和重启后）都会更新动态 DNS 记录。在本例中，将安装 noip2 客户端，该客户端利用 noip.com 提供的服务。

   a. 启用适用于企业 Linux 的额外软件包 (EPEL) 存储库以获得对`noip2`客户端的访问权限。

> **Note**
>
> AL2 实例默认安装了 EPEL 存储库的 GPG 密钥和存储库信息。有关更多信息以及要下载此软件包的最新版本，请参阅https://fedoraproject.org/wiki/EPEL。

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

b. 安装 noip 软件包。

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

c. 创建配置文件。在提示时输入登录名和密码信息，并回答后续问题以配置客户端。

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

3. 启用 noip 服务。

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

4. 启动 noip 服务。

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

该命令启动客户端，读取先前创建的配置文件 (`/etc/no-ip2.conf`)，并且更新您选择的公用 DNS 名称的 IP 地址。

5. 验证更新客户端是否已为动态 DNS 名称设置了正确的 IP 地址。等待几分钟使 DNS 记录进行更新，然后尝试使用您在此过程中配置的公有 DNS 名称通过 SSH 连接到实例。

使用适用于 AL2 的 ec2-net-utils 配置您的网络接口

亚马逊 Linux 2 AMI 可能包含安装的其他脚本 AWS，即 ec2-net-utils。这些脚本可以选择性地自动配置您的网络接口。这些脚本仅适用于 AL2。

> **Note**
>
> 对于 Amazon Linux 2023，该`amazon-ec2-net-utils`软件包会在目录中生成特定于接口的`/run/systemd/network`配置。有关更多信息，请参阅 Amazon Linux 2023 用户指南中的联网服务。

如果软件包尚未安装，请使用以下命令在 AL2 上安装该软件包；如果已安装并且有其他更新可用，则使用以下命令对其进行更新：

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

以下组件属于 ec2-net-utils 的一部分；

udev 规则 (`/etc/udev/rules.d`)

在网络接口附加、分离或重新附加正在运行的实例时识别它们，并确保 hotplug 脚本运行 (`53-ec2-network-interfaces.rules`)。将 MAC 地址映射到设备名称 (生成 `75-persistent-net-generator.rules` 的 `70-persistent-net.rules`)。

hotplug 脚本

生成一个适用于 DHCP 的接口配置文件 (`/etc/sysconfig/network-scripts/ifcfg-ethN`)。并生成一个路由配置文件 (`/etc/sysconfig/network-scripts/route-ethN`)。

DHCP 脚本

每当网络接口收到一个新的 DHCP 租约时，此脚本会查询弹性 IP 地址的实例元数据。对于每个弹性 IP 地址，它会为路由策略数据库添加一个规则，确保来自该地址的出站流量使用正确的网络接口。它还会将每个私有 IP 地址作为辅助地址添加至网络接口。

ec2ifup ethN (`/usr/sbin/`)

扩展标准 ifup 的功能。在此脚本重写配置文件 `ifcfg-ethN` 和 `route-ethN` 之后，它将运行 ifup。

ec2ifdown ethN (`/usr/sbin/`)

扩展标准 ifdown 的功能。当此脚本从路由策略数据库中删除网络接口的任何规则后，它将运行 ifdown。

ec2ifscan (`/usr/sbin/`)

检查尚未配置的网络接口并对它们进行配置。

此脚本在初始版本的 ec2-net-utils 中不可用。

要列出任何由 ec2-net-utils 生成的配置文件，请使用以下命令：

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

要禁用自动化，您可以将 EC2SYNC=no 添加至相应的 ifcfg-ethN 文件。例如，您可以使用以下命令为 eth1 接口禁用自动化：

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

要彻底禁用自动化，可以使用以下命令删除该包：

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

用户提供的内核

如果您的 Amazon EC2 实例上需要自定义内核，您可以从接近于您想要的内核的 AMI 开始，在您的实例上编译自定义内核，并更新引导加载程序以指向新内核。该过程根据您的AMI所使用的虚拟化类型而异。有关更多信息，请参阅 Amazon EC2 用户指南中的 Linux AMI 虚拟化类型。

内容

HVM AMIs (GRUB)

HVM 实例卷就像是物理磁盘。引导过程类似于带分区磁盘和引导加载程序的裸机操作系统，使它能够在当前支持的所有 Linux 发行版中工作。最常见的引导加载程序是 GRUB 或 GRUB2。

默认情况下，GRUB 不会将其输出发送到实例控制台，因为它会造成额外启动延迟。有关更多信息，请参阅《Amazon EC2 用户指南》中的实例控制台输出。如果要安装自定义内核，则应考虑启用 GRUB 输出。

无需指定后备内核，但是我们建议您在测试新内核时准备好后备内核。如果新内核发生故障时，GRUB 可以退回到另一个内核。如果有后备内核，实例即使没有找到新内核也能进行引导。

旧版 GRUB for Amazon Linux 使用 `/boot/grub/menu.lst`。GRUB2 适用于 AL2 用途`/etc/default/grub`。有关更新引导加载程序中的默认内核的更多信息，请参阅 Linux 发行版的文档。

半虚拟化 AMI () PV-GRUB

使用半虚拟化 (PV) 虚拟化的 AMI 使用在启动PV-GRUB过程中调用的系统。 PV-GRUB 是一个半虚拟化引导加载程序，它运行 GNU GRUB 0.97 的补丁版本。启动实例时， PV-GRUB 启动启动过程，然后链式加载映像`menu.lst`文件指定的内核。

PV-GRUB 理解标准`grub.conf`或`menu.lst`命令，这使其能够与当前支持的所有Linux发行版一起使用。较旧发行版（如 Ubuntu 10.04 LTS、Oracle Enterprise Linux 或 CentOS 5.x）需要特殊的“ec2”或“xen”内核软件包，而较新发行版在默认内核软件包中包含所需驱动程序。

大多数现代半虚拟化 AM PV-GRUB I 默认使用 AKI（包括 Amazon EC2 Launch Wizard 快速入门菜单中所有可用的半虚拟化 Linux AMI），因此，只要您要使用的内核与您的发行版兼容，您无需采取其他步骤即可在实例上使用不同的内核。在实例上运行自定义内核的最佳方式是从接近于您想要的内核的 AMI 开始，然后在实例上编译自定义内核并修改 `menu.lst` 文件以使用该内核进行引导。

您可以验证 AMI 的内核映像是否为 A PV-GRUB KI。运行以下 describe-images 命令（替换为您的内核映像 ID），并检查 `Name` 字段是否以 `pv-grub` 开头：

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

内容

- 的局限性 PV-GRUB
- 为半虚拟化 AMIs 配置 GRUB
- 亚马逊 PV-GRUB 内核映像 ID
- 更新 PV-GRUB

的局限性 PV-GRUB

PV-GRUB 有以下限制：

- 不能使用 64 位版本的 PV-GRUB 来启动 32 位内核，反之亦然。
- 使用 A PV-GRUB KI 时，您无法指定亚马逊虚拟硬盘映像 (ARI)。
- AWS 已测试并验证它 PV-GRUB 适用于以下文件系统格式：EXT2、EXT3、EXT4、JFS、XFS 和 ReiserFS。其他文件系统格式可能不适用。

- PV-GRUB 可以启动使用 gzip、bzip2、lzo 和 xz 压缩格式压缩的内核。
- 集群 AMI 不支持也不需要 PV-GRUB，因为它们使用的是完整的硬件虚拟化 (HVM)。当半虚拟化实例 PV-GRUB 用于启动时，HVM 实例卷被视为实际磁盘，启动过程类似于带有分区磁盘和引导加载程序的裸机操作系统的启动过程。
- PV-GRUB 1.03 及更早版本不支持 GPT 分区；它们仅支持 MBR 分区。
- 如果您计划通过 Amazon Elastic Block Store (Amazon EBS) 卷使用逻辑卷管理 (LVM)，则需要在 LVM 外有一个独立的引导分区。然后，您可以通过 LVM 创建逻辑卷。

为半虚拟化 AMIs 配置 GRUB

要启动 PV-GRUB，映像中必须存在 GRUB `menu.lst` 文件；该文件最常见的位置是`/boot/grub/menu.lst`。

以下是使用 A PV-GRUB KI 启动 AMI 的`menu.lst`配置文件示例。在此示例中，有两个内核条目可供选择：Amazon Linux 2018.03（此 AMI 的原始内核）和 Vanilla Linux 4.16.4（Vanilla Linux 内核的较新版本）。https://www.kernel.org/Vanilla 条目是从此 AMI 的原始条目复制的，`kernel` 和 `initrd` 路径已更新为新位置。`default 0` 参数将引导加载程序指向其发现的第一个条目 (在此例中为 Vanilla 条目)，`fallback 1` 参数在引导第一个条目的过程中发生问题时，将引导加载程序指向下一个条目。

```
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
```

您无需在`menu.lst`文件中指定备用内核，但我们建议您在测试新内核时使用备用内核。 PV-GRUB如果新内核出现故障，可以回退到另一个内核。如果有后备内核，实例即使没有找到新内核也能进行引导。

PV-GRUB 使用它找到的`menu.lst`第一个位置来检查以下位置：

- (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

请注意，PV-GRUB 1.03 及更早版本仅检查此列表中前两个位置中的一个。

亚马逊 PV-GRUB 内核映像 ID

PV-GRUB AKI 在所有 Amazon EC2 区域都可用，亚太地区（大阪）除外。同时存在适用于 32 位和 64 位架构类型的 AKI。大多数现代 AMI 默认使用 A PV-GRUB KI。

我们建议您始终使用最新版本的 PV-GRUB AKI，因为并非所有版本的 PV-GRUB AKI 都与所有实例类型兼容。使用以下 d escribe-images 命令获取当前区域 PV-GRUB 的 AKI 列表：

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

PV-GRUB 是该ap-southeast-2地区唯一可用的 AKI。您应验证要复制到该区域的任何 AMI 是否正在使用 PV-GRUB 该区域中可用的版本。

以下是每个区域的当前 AKI ID。使用 hd0 AKI 注册新 AMI。

> **Note**
>
> 在之前提供 hd00 AKI 的区域，我们将继续提供 hd00 AKI，以实现向后兼容性。

ap-northeast-1，亚太区域（东京）

映像 ID	映像名称
aki-f975a998	pv-grub-hd0_1.05-i386.gz

映像 ID	映像名称
aki-7077ab11	pv-grub-hd0_1.05-x86_64.gz

ap-southeast-1、亚太区域（新加坡）

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

ap-southeast-2、亚太区域（悉尼）

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

eu-central-1、欧洲（法兰克福）

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

eu-west-1、欧洲（爱尔兰）

映像 ID	映像名称
aki-1c9fd86f	pv-grub-hd0_1.05-i386.gz
aki-dc9ed9af	pv-grub-hd0_1.05-x86_64.gz

sa-east-1、南美洲（圣保罗）

映像 ID	映像名称
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	映像名称
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 (US-West)

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

us-west-1、美国西部（加利福尼亚北部）

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

us-west-2、美国西部（俄勒冈）

映像 ID	映像名称
aki-7a69931a	pv-grub-hd0_1.05-i386.gz

映像 ID	映像名称
aki-70cb0e10	pv-grub-hd0_1.05-x86_64.gz

更新 PV-GRUB

我们建议您始终使用最新版本的 PV-GRUB AKI，因为并非所有版本的 PV-GRUB AKI 都与所有实例类型兼容。此外，旧版本的 PV-GRUB 并非在所有地区都可用，因此，如果您将使用旧版本的 AMI 复制到不支持该版本的区域，则在更新内核映像之前，您将无法启动从该 AMI 启动的实例。使用以下过程检查您的实例版本 PV-GRUB 并在必要时对其进行更新。

要检查您的 PV-GRUB 版本

1. 查找您的实例的内核 ID。

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

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

   此实例的内核 ID 是 aki-70cb0e10。

2. 查看该内核 ID 的版本信息。

```
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"
        }
    ]
}
```

这个内核镜像是 PV-GRUB 1.05。如果您的 PV-GRUB 版本不是最新版本（如所示亚马逊 PV-GRUB 内核映像 ID），则应使用以下步骤对其进行更新。

更新您的 PV-GRUB 版本

如果您的实例使用的是的旧版本 PV-GRUB，则应将其更新到最新版本。

1. 从中识别适用于您所在地区和处理器架构的最新 A PV-GRUB KI 亚马逊 PV-GRUB 内核映像 ID。
2. 停止您的实例。您的实例必须停止才能修改所使用的内核映像。

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

3. 修改用于您的实例的内核映像。

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

4. 重新启动您的实例。

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

AL2 AMI 发布通知

要收到新 Amazon Linux AMI 发布的通知，可使用 Amazon SNS 订阅。

有关订阅 AL2023 通知的信息，请参阅亚马逊 Linux 2023 用户指南中的接收有关新更新的通知。AL2023

> **Note**
>
> 对 AL1 的标准支持已于 2020 年 12 月 31 日结束。AL1 维护支持阶段已于 2023 年 12 月 31 日结束。有关 AL1 EOL 和维护支持的更多信息，请参阅博客文章 Amazon Linux AMI 生命周期终止更新。

订阅 Amazon Linux 通知

1. 从 https://console.aws.amazon.com/sns/v3/home 打开 Amazon SNS 控制台。

2. 如果需要，可在导航栏中将区域更改为美国东部（弗吉尼亚北部）。必须选择所订阅的 SNS 通知在创建时所在的区域。
3. 在导航窗格中，依次选择 Subscriptions 和 Create subscription。
4. 对于 Create subscription 对话框，执行以下操作：

   a. [AL2] 对于主题 ARN，请复制并粘贴以下亚马逊资源名称 (ARN)：。**arn:aws:sns:us-east-1:137112412989:amazon-linux-2-ami-updates**

   b. [Amazon Linux] 对于主题 ARN，复制并粘贴以下 Amazon Resource Name (ARN)：**arn:aws:sns:us-east-1:137112412989:amazon-linux-ami-updates**。

   c. 对于协议，选择电子邮件。

   d. 对于端点，输入可用来接收通知的电子邮件地址。

   e. 选择创建订阅。
5. 您会收到一封主题为 “AWS 通知-订阅确认” 的确认电子邮件。打开电子邮件，然后选择 Confirm subscription 以完成订阅。

每当发布新的 AMI 时，我们都会向相应主题的订阅者发送通知。若不想再接收这些通知，请使用以下过程取消订阅。

取消订阅 Amazon Linux 通知

1. 从 https://console.aws.amazon.com/sns/v3/home 打开 Amazon SNS 控制台。
2. 如果需要，可在导航栏中将区域更改为美国东部（弗吉尼亚北部）。必须使用创建 SNS 通知的区域。
3. 在导航窗格中，选择订阅，选择订阅，然后选择操作和删除订阅。
4. 当系统提示进行确认时，选择 Delete（删除）。

Amazon Linux AMI SNS 消息格式

SNS 消息的架构如下所示。

```
{
    "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"
    ]
}
```

配置 AL2 MATE 桌面连接

MATE 桌面环境 已在 AMI 中预装和预配置，描述如下：

".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)."

该环境提供了一个直观的图形用户界面，便于管理 AL2 实例，只需使用最少的命令行即可。该界面使用图形表示，例如图标、窗口、工具栏、文件夹、墙纸和桌面小部件。 Built-in，可以使用 GUI-based 工具来执行常见任务。例如，有一些工具可用于添加和删除软件、应用更新、组织文件、启动程序和监视系统运行状况。

> **Important**
>
> xrdp 是 AMI 中捆绑的远程桌面软件。默认情况下，xrdp 使用自签名 TLS 证书来加密远程桌面会话。xrdp维护者 AWS 也不建议在生产环境中使用自签名证书。而是从相应的证书颁发机构 (CA) 获取证书并将其安装在您的实例上。有关 TSL 配置的更多信息，请参阅 xrdp Wiki 上的 TLS 安全层。

> **Note**
>
> 如果您更喜欢使用虚拟网络计算 (VNC) 服务而不是 xrdp，请参阅如何在运行 AL AWS 2 知识中心的 Amazon EC2 实例上安装 GUI 一文。

先决条件

要运行本主题中显示的命令，必须安装 AWS Command Line Interface (AWS CLI) 或 AWS Tools for Windows PowerShell，然后配置您的 AWS 配置文件。

选项

1. 安装 AWS CLI — 有关更多信息，请参阅《AWS Command Line Interface 用户指南》中的安装 AWS CLI和配置基础知识。
2. 安装适用于 Windows 的工具 PowerShell -有关更多信息，请参阅《AWS Tools for PowerShell 用户指南》中的安装 AWS Tools for Windows PowerShell和共享凭据。

> **Tip**
>
> 作为完全安装的替代方案 AWS CLI，您可以使用AWS CloudShell直接从启动的基于浏览器、经过预先验证的 shell。 AWS 管理控制台请选中支持 AWS 区域，确保它在您所在的地区可用。

配置 RDP 连接

按照以下步骤设置从本地计算机到运行 MATE 桌面环境的 AL2 实例的远程桌面协议 (RDP) 连接。

1. 要获取 AMI 名称中包含 MATE 的 AL2 的 AMI 的 ID，您可以使用本地命令行工具中的 desc ribe-images 命令。如果您尚未安装命令行工具，则可以直接从 AWS CloudShell 会话中执行以下查询。有关如何从启动 shell 会话的信息 CloudShell，请参阅入门 AWS CloudShell。在 Amazon EC2 控制台中，您可以通过启动实例，然后在 MATE-included AMI 搜索栏MATE中输入来找到 AMI。预装了 MATE 的 AL2 快速入门将出现在搜索结果中。

```
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"
    ]
```

```
]
```

选择适合您使用的 AMI。

2. 使用您在上一步中找到的 AMI 启动 EC2 实例。将安全组配置为允许到端口 3389 的入站 TCP 流量。有关配置安全组的更多信息，请参阅 VPC 的安全组。此配置使您能够使用 RDP 客户端连接到实例。
3. 使用 SSH 连接到实例。
4. 更新实例上的软件和内核。

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

在更新完成后，重启实例以确保实例使用的是来自更新的最新程序包和库；重启发生前不会加载内核更新。

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

5. 重新连接到实例并在 Linux 实例上运行以下命令以设置 ec2-user 的密码。

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

6. 安装证书和密钥。

如果您已经拥有证书和密钥，将它们复制到 /etc/xrdp/ 目录，如下所示：

- 证书 — /etc/xrdp/cert.pem
- 密钥 — /etc/xrdp/key.pem

如果您没有证书和密钥，则使用以下命令在 /etc/xrdp 目录中生成。

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

> **Note**
>
> 此命令会生成有效期达 365 天的证书。

7. 在要连接到实例的计算机上打开 RDP 客户端（例如，运行 Microsoft Windows 的计算机上的远程桌面连接）。输入 ec2-user 作为用户名，然后输入您在上一步中设置的密码。

在您的亚马逊 EC2 实例上禁用 **xrdp**

您可以通过在 Linux 实例上运行以下命令之一随时禁用 xrdp。以下命令不会影响您通过 X11 服务器使用 MATE 的能力。

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

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

在您的亚马逊 EC2 实例上启用 **xrdp**

要重新启用xrdp以便您可以连接到运行 MATE 桌面环境的 AL2 实例，请在 Linux 实例上运行以下命令之一。

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

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

AL2 教程

以下教程向您展示如何使用运行 AL2 的 Amazon EC2 实例执行常见任务。有关视频教程，请参阅 AWS 教学视频和实验。

有关 AL2023 的说明，请参阅亚马逊 Linux 2023 用户指南中的教程。AL2023

教程

- 教程：在 AL2 上安装 LAMP 服务器
- 教程：在 AL SSL/TLS 2 上配置
- 教程：在 AL2 上发布 WordPress 博客

教程：在 AL2 上安装 LAMP 服务器

以下过程可帮助你在 AL2 实例（有时称为 LAMP 网络服务器或 LAMP 堆栈）上安装支持 PHP 和 M ar iaDB（社区开发的 MySQL 分支）的 Apache 网络服务器。您可以使用此服务器来托管静态网站或部署能对数据库中的信息执行读写操作的动态 PHP 应用程序。

> **Important**
>
> 如果您尝试在其他发行版（例如 Ubuntu 或红帽企业 Linux）上设置 LAMP Web 服务器，则本教程不适合。对于 AL2023，请参阅在 AL2023 上安装 LAMP 服务器。AL2023 AL2023 对于 Ubuntu，请参阅以下 Ubuntu 社区文档：SQLPHP。ApacheMy有关其他发布版本，请参阅特定于该版本的文档。

选项：使用 Automation 完成本教程

要使用 AWS Systems Manager 自动化而不是以下任务来完成本教程，请运行AWS Docs-InstallALAMPServer-AL2自动化文档。

任务

步骤 1：准备 LAMP 服务器

先决条件

- 本教程假设您已经使用 AL2 启动了一个新实例，其公有 DNS 名称可以从互联网访问。有关更多信息，请参阅《Amazon EC2 用户指南》中的启动实例。您还必须配置安全组，以便允许 SSH (端口 22)、HTTP (端口 80) 和 HTTPS (端口 443) 连接。有关这些先决条件的更多信息，请参阅 Amazon EC2 用户指南中的安全组规则。
- 以下过程安装当前php8.2在 AL2 上可用的最新 PHP 版本。如果您计划使用本教程中所述的 PHP 应用程序之外的 PHP 应用程序，则应检查其与 php8.2 的兼容性。

准备 LAMP 服务器

1. 连接到您的 实例。
2. 为确保您的所有软件包都处于最新状态，请对您的实例执行快速软件更新。此过程可能需要几分钟的时间，但必须确保您拥有最新的安全更新和缺陷修复。

   -y 选项安装更新时不提示确认。如果您希望在安装前检查更新，则可以忽略该选项。

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

3. 安装 mariadb10.5 Amazon Linux Extras 存储库，以获取 MariaDB 程序包的最新版本。

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

   如果您收到指示 sudo: amazon-linux-extras: command not found 的错误，则表示您的实例未与 Amazon Linux 2 AMI 一起启动（也许您可以改用 Amazon Linux AMI）。您可以使用以下命令查看 Amazon Linux 的版本。

   ```
   cat /etc/system-release
   ```

4. 安装 php8.2 Amazon Linux Extras 存储库以获取 AL2 PHP 软件包的最新版本。

   ```
   [ec2-user ~]$ sudo amazon-linux-extras install php8.2
   ```

5. 现在您的实例处于最新状态，您可以安装 Apache Web 服务器、MariaDB 和 PHP 软件包。使用 yum 安装命令可同时安装多个软件包和所有相关依赖项

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

   您可以使用以下命令查看这些程序包的当前版本：

   ```
   yum info package_name
   ```

6. 启动 Apache Web 服务器。

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

7. 使用 systemctl 命令配置 Apache Web 服务器，使其在每次系统启动时启动。

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

您可以通过运行以下命令验证 httpd 是否已启用：

```
[ec2-user ~]$ sudo systemctl is-enabled httpd
```

8. 如果您尚未这样做，请添加安全规则以允许与您的实例的入站 HTTP (端口 80) 连接。默认情况下，在初始化期间为您的实例设置了启动向导***N***安全组。此组包含一条允许 SSH 连接的规则。

   a. 打开位于 https://console.aws.amazon.com/ec2/ 的 Amazon EC2 控制台。

   b. 选择 Instances 并选择您的实例。

   c. 在安全选项卡上，查看入站规则。您应看到以下规则：

   ```
   Port range   Protocol     Source
   22           tcp          0.0.0.0/0
   ```

   > **Warning**
   >
   > 使用 `0.0.0.0/0` 可允许所有 IPv4 地址使用 SSH 访问您的实例。这在测试环境中可以接受一小段时间，但是在生产环境中并不安全。在生产环境中，您仅授权特定 IP 地址或地址范围访问您的实例。

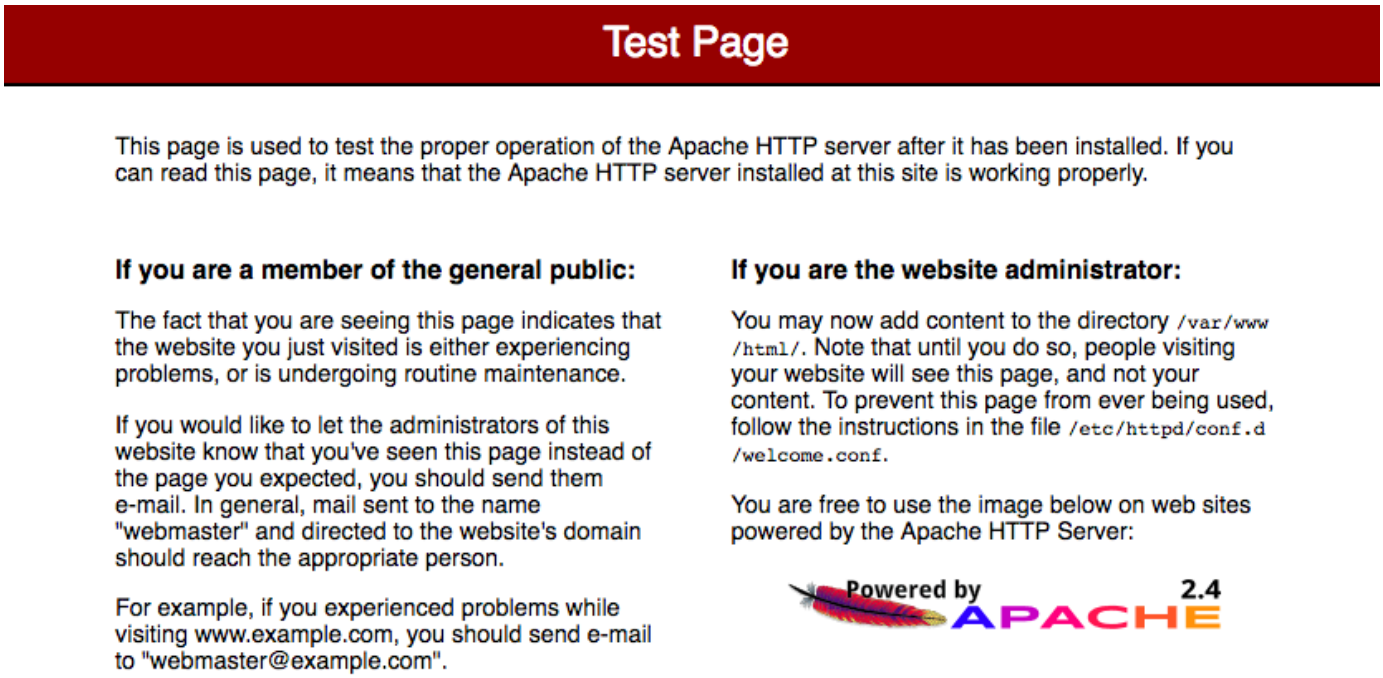
   d. 选择安全组的链接。使用向安全组添加规则中的步骤，添加具有以下值的新入站安全规则：

   - 类型：HTTP
   - 协议：TCP
   - Port Range：80
   - Source：Custom

9. 测试您的 Web 服务器。在 Web 浏览器中，键入您的实例的公有 DNS 地址 (或公有 IP 地址)。如果 `/var/www/html` 中没有内容，您应该会看到 Apache 测试页面。您可以使用 Amazon EC2 控制台获取实例的公有 DNS（检查公有 DNS 列；如果此列处于隐藏状态，请选择Show/Hide 列（齿轮形图标）并选择公有 DNS）。

   验证实例的安全组是否包含允许端口 80 上的 HTTP 流量的规则。更多信息，请参阅向安全组添加规则。

> **Important**
>
> 如果您使用的不是 Amazon Linux，则还可能需要在实例上配置防火墙才能允许这些连接。有关如何配置防火墙的更多信息，请参阅适用于特定分配的文档。



Apache httpd 提供的文件保存在名为 Apache 文档根目录的目录中。Amazon Linux Apache 文档根目录为 `/var/www/html`，默认情况下归根用户所有。

要允许 `ec2-user` 账户操作此目录中的文件，必须修改其所有权和权限。有多种方式可以完成此任务。在本教程中，可将 `ec2-user` 添加到 `apache` 组，将 `/var/www` 目录的所有权授予 `apache` 组，并为该组指定写入权限。

设置文件权限

1. 将您的用户 (这里指 `ec2-user`) 添加到 `apache`。

```
[ec2-user ~]$ sudo usermod -a -G apache ec2-user
```

2. 先退出再重新登录以选取新组，然后验证您的成员资格。

a. 退出（使用 exit 命令或关闭终端窗口）：

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

b. 要验证您是否为 apache 组的成员，请重新连接到实例，然后运行以下命令：

```
[ec2-user ~]$ groups
ec2-user adm wheel apache systemd-journal
```

3. 将 /var/www 及其内容的组所有权更改到 apache 组。

```
[ec2-user ~]$ sudo chown -R ec2-user:apache /var/www
```

4. 要添加组写入权限以及设置未来子目录上的组 ID，请更改 /var/www 及其子目录的目录权限。

```
[ec2-user ~]$ sudo chmod 2775 /var/www && find /var/www -type d -exec sudo chmod
 2775 {} \;
```

5. 要添加组写入权限，请递归地更改 /var/www 及其子目录的文件权限：

```
[ec2-user ~]$ find /var/www -type f -exec sudo chmod 0664 {} \;
```

这样，ec2-user (和 apache 组的任何未来成员) 可以添加、删除和编辑 Apache 文档根目录中的文件，允许您添加内容，如静态网站或 PHP 应用程序。

保护您的 Web 服务器 (可选)

运行 HTTP 协议的 Web 服务器不为其发送或接收的数据提供传输安全。当您使用 Web 浏览器连接 HTTP 服务器时，对于您访问的 URL、您接收的网页内容以及您提交的任何 HTML 表的内容 (包括密码)，窃取者可在网络路径上的任何位置看到。保护 Web 服务器的最佳做法是安装对 HTTPS（HTTP 安全）的支持，它通过 SSL/TLS加密来保护您的数据。

有关在服务器上启用 HTTPS 的信息，请参阅 教程：在 AL SSL/TLS 2 上配置。

步骤 2：测试 LAMP 服务器

如果服务器已安装并运行，且文件权限设置正确，则 ec2-user 账户应该能够在 /var/www/html 目录 (可从 Internet 访问) 中创建 PHP 文件。

测试您的 LAMP 服务器

1. 在 Apache 文档根目录中创建一个 PHP 文件。

```
[ec2-user ~]$ echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php
```

尝试运行该命令时，如果出现“Permission denied (权限被拒绝)”错误，请尝试先注销，再重新登录，以获取您在 设置文件权限 中配置的适当组权限。

2. 在 Web 浏览器中，键入您刚刚创建的文件的 URL。此 URL 是实例的公用 DNS 地址，后接正斜杠和文件名。例如：

```
http://my.public.dns.amazonaws.com/phpinfo.php
```

您应该会看到 PHP 信息页面：

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

如果您未看到此页面，请验证上一步中是否已正确创建 /var/www/html/phpinfo.php 文件。您还可以使用以下命令验证已经安装了所有必需的程序包。

```
[ec2-user ~]$ sudo yum list installed httpd mariadb-server php-mysqlnd
```

如果输出中未列出任何必需的程序包，请使用 sudo yum install ***package*** 命令安装它们。另请验证在 php7.2 命令的输出中启用了 lamp-mariadb10.2-php7.2 和 amazon-linux-extras Extras。

3. 删除 phpinfo.php 文件。尽管此信息可能很有用，但出于安全考虑，不应将其传播到 Internet。

```
[ec2-user ~]$ rm /var/www/html/phpinfo.php
```

现在，您应该有了一个功能完善的 LAMP Web 服务器。如果您将内容添加到 Apache 文档根目录 (位于 /var/www/html)，您应该能够在您的实例的公有 DNS 地址中看到该内容。

步骤 3：确保数据库服务器的安全

MariaDB 服务器的默认安装提供有多种功能，这些功能对于测试和开发都很有帮助，但对于产品服务器，应禁用或删除这些功能。mysql_secure_installation 命令可引导您设置根密码并删除安装中的不安全功能。即使您不打算使用 MariaDB 服务器，我们也建议执行此步骤。

保护 MariaDB 服务器

1. 启动 MariaDB 服务器。

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

2. 运行 mysql_secure_installation。

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

   a. 在提示时，键入根账户的密码。

      i. 键入当前根密码。默认情况下，根账户没有设置密码。按 Enter。

      ii. 键入 **Y** 设置密码，然后键入两次安全密码。有关创建安全密码的更多信息，请参阅https://identitysafe.norton.com/password-generator/。确保将此密码存储在安全位置。

      设置 MariaDB 根密码仅是保护数据库的最基本措施。在您构建或安装数据库驱动的应用程序时，您通常可以为该应用程序创建数据库服务用户，并避免使用根账户执行除数据库管理以外的操作。

   b. 键入 **Y** 删除匿名用户账户。

   c. 键入 **Y** 禁用远程根登录。

   d. 键入 **Y** 删除测试数据库。

   e. 键入 **Y** 重新加载权限表并保存您的更改。

3. (可选) 如果您不打算立即使用 MariaDB 服务器，请停止它。您可以在需要时再次重新启动。

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

4. (可选) 如果您希望每次启动时 MariaDB 服务器都启动，请键入以下命令。

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

第 4 步：（可选）安装 php MyAdmin

php MyAdmin 是一个基于 Web 的数据库管理工具，可用于查看和编辑 EC2 实例上的 MySQL 数据库。按照下述步骤操作，在您的 Amazon Linux 实例上安装和配置 phpMyAdmin。

> **Important**
>
> 除非您已在 Apache SSL/TLS 中启用，否则我们不建议使用phpMyAdmin来访问 LAMP 服务器；否则，您的数据库管理员密码和其他数据将不安全地通过 Internet 传输。有关开发人员提出的安全建议，请参阅保护您的 php MyAdmin 安装。有关在 EC2 实例上保护 Web 服务器的一般信息，请参阅 教程：在 AL SSL/TLS 2 上配置。

要安装 php MyAdmin

1. 安装所需的依赖项。

```
[ec2-user ~]$ sudo yum install php-mbstring php-xml -y
```

2. 重启 Apache。

```
[ec2-user ~]$ sudo systemctl restart httpd
```

3. 重启 php-fpm。

```
[ec2-user ~]$ sudo systemctl restart php-fpm
```

4. 导航到位于 /var/www/html 的 Apache 文档根。

```
[ec2-user ~]$ cd /var/www/html
```

5. 从中选择最新 php MyAdmin 版本的源包https://www.phpmyadmin.net/downloads。要将文件直接下载到您的实例，请复制链接并将其粘贴到 wget 命令，如本示例中所述：

```
[ec2-user html]$ wget https://www.phpmyadmin.net/downloads/phpMyAdmin-latest-all-languages.tar.gz
```

6. 使用以下命令创建 phpMyAdmin 文件夹并将程序包提取到其中。

```
[ec2-user html]$ mkdir phpMyAdmin && tar -xvzf phpMyAdmin-latest-all-languages.tar.gz -C phpMyAdmin --strip-components 1
```

7. 删除*phpMyAdmin-latest-all-languages.tar.gz*压缩包。

```
[ec2-user html]$ rm phpMyAdmin-latest-all-languages.tar.gz
```

8. (可选) 如果 MySQL 服务器未运行，请立即启动它。

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

9. 在 Web 浏览器中，键入 php MyAdmin 安装的 URL。此 URL 是实例的公有 DNS 地址 (或公有 IP 地址)，后接正斜杠和您安装目录的名称。例如：

```
http://my.public.dns.amazonaws.com/phpMyAdmin
```

你应该会看到 php MyAdmin 登录页面：

10. 使用你之前创建的`root`用户名和 MySQL 根密码登录你的 php MyAdmin 安装程序。

    您的安装仍需进行配置，然后才能投入使用。我们建议您首先手动创建配置文件，如下所示：

    a. 要从最小的配置文件开始，请使用您常用的文本编辑器创建一个新文件，然后将`config.sample.inc.php` 的内容复制到该文件中。

    b. 将文件另存为`config.inc.php`包含的 php MyAdmin 目录中`index.php`。

    c. 有关任何其他设置，请参阅 php 安装说明的 “使用MyAdmin 安装脚本” 部分中的文件创建后说明。

    有关使用 php 的信息MyAdmin，请参阅 php MyAdmin 用户指南。

故障排除

本部分提供了解决在设置新 LAMP 服务器时可能遇到的常见问题的建议。

我无法使用 Web 浏览器连接到我的服务器

执行以下检查以查看您的 Apache Web 服务器是否正在运行且可以访问。

- Web 服务器正在运行吗？

  您可以通过运行以下命令验证 httpd 是否已启用：

  ```
  [ec2-user ~]$ sudo systemctl is-enabled httpd
  ```

  如果 httpd 进程未运行，请重复准备 LAMP 服务器中描述的步骤。
- 防火墙是否配置正确？

  验证实例的安全组是否包含允许端口 80 上的 HTTP 流量的规则。更多信息，请参阅向安全组添加规则。

我无法使用 HTTPS 连接到我的服务器

执行以下检查以查看 Apache Web 服务器是否配置为支持 HTTPS。

- Web 服务器配置是否正确？

  安装 Apache 后，服务器将针对 HTTP 流量进行配置。要支持 HTTPS，请在服务器上启用 TLS 并安装 SSL 证书。有关信息，请参阅 教程：在 AL SSL/TLS 2 上配置。
- 防火墙是否配置正确？

  验证实例的安全组是否包含允许端口 443 上的 HTTPS 流量的规则。有关更多信息，请参阅向安全组添加规则。

相关主题

有关将文件传输到您的实例或在 Web 服务器上安装 WordPress 博客的更多信息，请参阅以下文档：

- 使用将文件传输到您的 Linux 实例WinSCP。
- 使用SCP客户端将文件传输到 Linux 实例。

- 教程：在 AL2 上发布 WordPress 博客

有关本教程中使用的命令和软件的更多信息，请参阅以下网页：

- Apache 网络服务器：http://httpd.apache.org/
- MariaDB 数据库服务器：https://mariadb.org/
- PHP 编程语言：http://php.net/
- chmod命令：https://en.wikipedia.org/wiki/Chmod
- chown命令：https://en.wikipedia.org/wiki/Chown

有关注册 Web 服务器域名或将现有域名转移到此主机的更多信息，请参阅 Amazon Route 53 开发人员指南中的创建域和子域并将其迁移到 Amazon Route 53。

教程：在 AL SSL/TLS 2 上配置

Secure Sockets L Layer/Transport ayer Security (SSL/TLS) 在 Web 服务器和 Web 客户端之间创建了一个加密通道，以保护传输中的数据免遭窃听。本教程介绍如何在带有 AL2 和 Apache Web 服务器的 EC2 实例 SSL/TLS 上手动添加支持。本教程假定您未使用负载均衡器。如果您正在使用 Elastic Load Balancing，则可以选择使用来自 AWS Certificate Manager 的证书在负载均衡器上配置 SSL 卸载。

由于历史原因，Web 加密通常简称为 SSL。虽然 Web 浏览器仍支持 SSL，但使用其下一代协议 TLS 更不易受攻击。默认情况下，AL2 禁用服务器端对所有 SSL 版本的支持。安全标准机构认为 TLS 1.0 不安全。TLS 1.0 和 TLS 1.1 已于 2021 年 3 月正式弃用。本教程仅包含有关启用 TLS 1.2 的指导。TLS 1.3 于 2018 年定稿，只要底层 TLS 库（本教程中为 OpenSSL）受支持并启用，即可在 AL2 中使用。客户端必须在 2023 年 6 月 28 日之前支持 TLS 1.2 或更高版本。有关更新的加密标准的更多信息，请参阅 RFC 7568 和 RFC 8446。

在本教程中，将现代 Web 加密简称为 TLS。

> **Important**
>
> 这些程序适用于 AL2。我们还假定您从新的 Amazon EC2 实例开始。如果您正在尝试设置运行其他分配的 EC2 实例，或者设置运行旧版本 AL2 的实例，则本教程中的某些步骤可能无法运行。对于 Ubuntu，请参阅以下社区文档：Open SSL on Ubuntu。有关 Red Hat Enterprise Linux 的信息，请参阅以下：设置 Apache HTTP Web 服务器。有关其他发布版本，请参阅特定于该版本的文档。

> **Note**
>
> 或者，您可以将 AWS Certificate Manager (ACM) 用于 AWS Nitro 安全区，这是一种安全区应用程序，允许您在带有 Nitro Enclaves 的 Amazon EC2 实例上运行的 Web 应用程序和服务器上使用公有和私有 SSL/TLS 证书。 AWS Nitro Enclaves 是一项 Amazon EC2 功能，它允许创建隔离的计算环境，以保护和安全地处理高度敏感的数据，例如 SSL/TLS 证书和私钥。适用于 Nitro Enclaves 的 ACM 与运行在 Amazon EC2 Linux 实例上的 nginx 结合使用，以创建私有密钥、分发证书和私有密钥以及管理证书续订。
> 要使用适用于 Nitro Enclaves 的 ACM，必须使用启用了 Enclave 的 Linux 实例。
> 有关更多信息，请参阅什么是 AWS 硝基飞地？ 以及AWS Certificate Manager 《Nitro Enclaves 用户指南》中的 AWS Nitro Enclaves。

内容

前提条件

在开始本教程之前，请完成以下步骤：

- 启动由亚马逊 EBS 支持的 AL2 实例。有关更多信息，请参阅《Amazon EC2 用户指南》中的启动实例。
- 配置安全组以允许您的实例接受以下 TCP 端口上的连接：
  - SSH (端口 22)
  - HTTP (端口 80)
  - HTTPS (端口 443)

  有关更多信息，请参阅《Amazon EC2 用户指南》中的安全组规则。
- 安装 Apache Web 服务器。有关分步说明，请参阅教程：在 AL2 上安装 LAMP Web 服务器。仅需要 httpd 包及其依赖项，因此可以忽略涉及 PHP 和 MariaDB 的说明。

- 要识别和验证网站，TLS 公有密钥基础设施 (PKI) 依赖于域名系统 (DNS)。要使用 EC2 实例托管公共网站，您需要为 Web 服务器注册一个域名，或者将现有域名转让给您的 Amazon EC2 主机。可通过很多第三方域注册和 DNS 托管服务来执行此操作，也可以使用 Amazon Route 53 执行此操作。

步骤 1：在服务器上启用 TLS

选项：使用 Automation 完成本教程

要使用 AWS Systems Manager 自动化而不是以下任务来完成本教程，请运行自动化文档。

此过程将引导您完成使用自签名数字证书在 AL2 上设置 TLS 的过程。

> **Note**
>
> 自签名证书对于测试是可接受的，但对于生产不是。如果您将自签名证书公开到 Internet，您网站的访客将会看到安全警告。

在服务器上启用 TLS

1. 连接到您的实例并确认 Apache 正在运行。

   ```
   [ec2-user ~]$ sudo systemctl is-enabled httpd
   ```

   如果返回的值不是“启用”，则启动 Apache 并将它设置为每次随系统一起启动。

   ```
   [ec2-user ~]$ sudo systemctl start httpd && sudo systemctl enable httpd
   ```

2. 为确保您的所有软件包都处于最新状态，请对您的实例执行快速软件更新。此过程可能需要几分钟的时间，但必须确保您拥有最新的安全更新和缺陷修复。

   > **Note**
   >
   > -y 选项安装更新时不提示确认。如果您希望在安装前检查更新，则可以忽略该选项。

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

3. 现在，您的实例是最新的，请安装 Apache 模块 mod_ssl 以添加 TLS 支持。

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

您的实例现在具有以下文件，可使用这些文件配置安全服务器并创建证书以进行测试：

- /etc/httpd/conf.d/ssl.conf

  mod_ssl 的配置文件。它包含一些指令 以指示 Apache 在何处查找以下信息：加密密钥和证书、要允许的 TLS 协议版本以及要接受的加密密码。

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

  用于为服务器主机生成自签名 X.509 证书和私钥的脚本。要测试是否正确设置 Apache 以使用 TLS，该证书是非常有用的。由于不提供身份证明，因此，不应在生产环境中使用该证书。如果在生产环境中使用该证书，则将在 Web 浏览器中触发警告。

4. 运行脚本以生成自签名虚拟证书和密钥以进行测试。

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

这会在 /etc/pki/tls/certs/ 目录中生成一个新文件 localhost.crt。指定的文件名与 /etc/httpd/conf.d/ssl.conf 中的 SSLCertificateFile 指令指定的默认值匹配。

该文件包含自签名证书以及证书的私有密钥。Apache 要求证书和密钥采用 PEM 格式，该格式由由 “BEGIN” 和 “END” 行组成的 Base64-encoded ASCII 字符组成，如以下缩写示例所示。

```
-----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-----
```

文件名和扩展名只是为了提供便利，对功能没有影响。例如，只要 `ssl.conf` 文件中的相关指令使用相同的名称，您就可以将证书命名为 `cert.crt`、`cert.pem` 或任何其他文件名。

> **Note**
>
> 在使用您自己的自定义文件替换默认 TLS 文件时，请确保它们采用 PEM 格式。

5. 使用您常用的文本编辑器（如 vim 或 nano）以根用户身份打开 `/etc/httpd/conf.d/ssl.conf` 文件并注释掉以下行，因为自签名虚拟证书也包含密钥。如果在完成下一步之前没有注释掉该行，Apache 服务将无法启动。

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

6. 重启 Apache。

```
[ec2-user ~]$ sudo systemctl restart httpd
```

> **Note**
>
> 确保 TCP 端口 443 在您的 EC2 实例上是可访问的，如之前所述。

7. 现在，您的 Apache Web 服务器应通过端口 443 支持 HTTPS (安全 HTTP)。通过将您的 EC2 实例的 IP 地址或完全限定域名与前缀 **`https://`** 一起输入浏览器 URL 栏中来对其进行测试。

   由于您正在使用自签名的不可信主机证书连接到站点，因此您的浏览器可能会显示一系列安全警告。忽视这些警告并继续连接站点。

   如果默认 Apache 测试页面打开，这意味着您已成功在服务器上配置 TLS。在浏览器和服务器之间传输的所有数据现在都已加密。

> **Note**
>
> 为防止网站访问者遇到警告屏幕，您必须获得可信的 CA-signed 证书，该证书不仅可以加密，还可以公开验证您作为网站所有者的身份。

步骤 2：获取 CA-signed 证书

您可以使用以下过程来获取 CA-signed 证书：

- 从私有密钥生成证书签名请求 (CSR)
- 将 CSR 提交给证书颁发机构 (CA)
- 获取签名的主机证书
- 配置 Apache 以使用证书

自签名 TLS X.509 主机证书在密码学上与证书相同。 CA-signed二者之间的区别在于社交层面，而非数学层面。CA 承诺，在向申请者颁发证书之前，至少验证域的所有权。每个 Web 浏览器均包含一个 CA 的列表，浏览器供应商信任这些 CA 来执行此操作。 X.509证书主要由与您的私有服务器密钥对应的公钥和以加密方式绑定到公钥的 CA 签名组成。当浏览器通过 HTTPS 连接到 Web 服务器时，服务器将提供证书以便浏览器检查其可信 CA 的列表。如果签署人位于列表上，或可通过由其他可信签署人组成的一系列信任访问，则浏览器将与服务器协商一个快速加密数据通道并加载页面。

由于验证请求需要投入人力，证书通常会产生费用，因此应货比三家。一些 CA 免费提供基础级别证书。其中最值得注意的 CA 是 Let's Encrypt 项目，该项目还支持证书创建和续订过程的自动化。有关使用 Let's Encrypt 证书的更多信息，请参阅获取 Certbot。

如果您打算提供商业级服务，AWS Certificate Manager 是一个不错的选择。

主机证书的基础是密钥。从 2019 年开始，政府和行业群体建议 RSA 密钥使用 2048 位的最小密钥（模数）大小，旨在将文档一直保护到 2030 年。OpenSSL 在 AL2 中生成的默认模数大小为 2048 位，适合在证书中使用。 CA-signed在以下过程中，为需要自定义密钥的人员提供了一个可选步骤，例如，具有较大模数或使用不同加密算法的步骤。

> **Important**
>
> 除非您拥有注册和托管的 DNS 域，否则这些获取 CA-signed 主机证书的说明不起作用。

获取 CA-signed 证书

1. 连接到您的实例并导航到//etc/pkitls/private/。这是存储 TLS 的服务器私有密钥的目录。如果您希望使用现有的主机密钥生成 CSR，请跳到步骤 3。
2. (可选) 生成新的私有密钥。下面是一些密钥配置示例。任何生成的密钥都可用于您的 Web 服务器，但它们实施安全的程度和类型有所不同。

   - 示例 1：创建默认 RSA 主机密钥。生成的文件 **custom.key** 是一个 2048 位 RSA 私有密钥。

     ```
     [ec2-user ~]$ sudo openssl genrsa -out custom.key
     ```

   - 示例 2：创建具有更大模数的更严格的 RSA 密钥。生成的文件 **custom.key** 是一个 4096 位 RSA 私有密钥。

     ```
     [ec2-user ~]$ sudo openssl genrsa -out custom.key 4096
     ```

   - 示例 3：创建具有密码保护的 4096 位加密的 RSA 密钥。生成的文件是使用密码加密的 4096 位 RSA 私钥。**custom.key** AES-128

     > **Important**
     >
     > 对密钥进行加密可增强安全性，但由于加密的密钥需要密码，因此依赖于加密密钥的服务无法自动启动。每当您使用此密钥时，都必须通过 SSH 连接提供密码（在上一示例中为“abcde12345”）。

     ```
     [ec2-user ~]$ sudo openssl genrsa -aes128 -passout pass:abcde12345 -out
      custom.key 4096
     ```

   - 示例 4：使用非 RSA 密码创建密钥。RSA 加密可能相对较慢，因为其公有密钥的大小基于两个大素数的乘积。不过，可以为 TLS 创建使用非 RSA 密码的密钥。在交付同等级别的安全性时，基于椭圆曲线的数学运算的密钥更小，计算起来更快。

     ```
     [ec2-user ~]$ sudo openssl ecparam -name prime256v1 -out custom.key -genkey
     ```

     结果为一个使用 prime256v1（OpenSSL 支持的“命名曲线”）的 256 位椭圆曲线私有密钥。根据 NIST，其加密强度略高于 2048 位 RSA 密钥。

> **Note**
>
> 并非所有 CA 对基于椭圆曲线的密钥的支持级别与对 RSA 密钥的支持级别相同。

确保新的私钥具有高度限制的所有权和权限（owner=root，group=root，仅适用于所有者）。read/write 命令将如以下示例所示。

```
[ec2-user ~]$ sudo chown root:root custom.key
[ec2-user ~]$ sudo chmod 600 custom.key
[ec2-user ~]$ ls -al custom.key
```

上述命令生成以下结果。

```
-rw------- root root custom.key
```

在创建并配置满意的密钥后，可以创建 CSR。

3. 使用您首选的密钥创建 CSR。下面的示例使用了 **custom.key**。

```
[ec2-user ~]$ sudo openssl req -new -key custom.key -out csr.pem
```

OpenSSL 将打开一个对话框，并提示您输入下表中显示的信息。对于基本的经域验证的主机证书来说，除 Common Name 以外的所有字段都是可选字段。

Name	说明	示例
国家/地区名称	代表国家/地区的两个字母 ISO 缩写。	US (=美国)
州或省名称	组织所在州或省的名称。此名称不可使用缩写。	Washington
所在地名称	您的组织所在的位置，例如城市。	Seattle
组织名称	组织的法定全称。请勿缩写组织名称。	Example Corporation

Name	说明	示例
组织部门名称	额外的组织信息 (如果有)。	示例部门
公用名	此值必须与您希望用户输入浏览器中的 Web 地址完全匹配。通常，这表示以主机名称为前缀的域名或采用 **www.example.com** 格式的别名。在使用自签名证书且无 DNS 解析的测试中，公用名可能只包含主机名。CA 还提供费用更高的证书，这些证书接受通配符名称（例如 ***.example.com** ）。	www.example.com
电子邮件地址	服务器管理员的电子邮件地址。	someone@example.com

最后，OpenSSL 将提示您输入可选的质询密码。此密码仅适用于 CSR 和您与 CA 之间的事务，因此请遵循 CA 提供的有关此密码以及其他可选字段、可选公司名的建议。CSR 质询密码不会影响服务器操作。

生成的文件 **csr.pem** 包含您的公有密钥、您的公有密钥的数字签名以及您输入的元数据。

4. 将 CSR 提交给 CA。这通常包括在文本编辑器中打开 CSR 文件并将内容复制到 Web 表格中。此时，您可能需要提供一个或多个主题备用名称 (SAN) 以放置到证书上。如果 **www.example.com** 是公用名，则 **example.com** 将是一个很好的 SAN，反之亦然。您网站的访客如果输入这两个名称的任何一个，便可看到一个没有错误的连接。如果您的 CA Web 表格允许该连接，请在 SAN 列表中包含公用名。一些 CA 会自动包含公用名。

   在您的请求获得批准后，您将收到一个由 CA 签署的新主机证书。此外，系统可能会指示您下载中间证书文件，该文件包含完成 CA 的信任链所需的其他证书。

   > **Note**
   >
   > 您的 CA 可能会针对各种用途发送多种格式的文件。在本教程中，您应只使用 PEM 格式的证书文件，此格式通常会（但不总是）标有 `.pem` 或 `.crt` 文件扩展名。如果您不确定要使用哪个文件，请使用文本编辑器打开这些文件，并查找一个包含一个或多个以下面的行开始的块的文件。

```
- - - - -BEGIN CERTIFICATE - - - - -
```

该文件还应以下面的行结束。

```
- - - -END CERTIFICATE - - - - -
```

您还可以在命令行上测试文件，如下所示。

```
[ec2-user certs]$ openssl x509 -in certificate.crt -text
```

验证这些行是否显示在文件中。请勿使用结尾为 .p7b、.p7c 或类似文件扩展名的文件。

5. 将新 CA-signed 证书和所有中间证书放在/etc/pki/tls/certs目录中。

> **Note**
>
> 可通过多种方法将新证书上传到 EC2 实例，但最直接、最有益的方法是在本地计算机和 EC2 实例上打开一个文本编辑器（例如，vi、nano 或记事本），然后在这两者之间复制并粘贴文件内容。在 EC2 实例上执行这些操作时，您需要根 [sudo] 权限。这样，一旦有任何权限或路径问题，您可以立即看到。但请小心操作，不要在复制内容时添加任何多余的行或以任何方式更改内容。

在/etc/pki/tls/certs目录内部，检查文件所有权、组和权限设置是否与严格限制的 AL2 默认值相匹配（owner=root、group=root，仅适用于所有者）。 read/write 以下示例显示了要使用的命令。

```
[ec2-user certs]$ sudo chown root:root custom.crt
[ec2-user certs]$ sudo chmod 600 custom.crt
[ec2-user certs]$ ls -al custom.crt
```

这些命令应生成以下结果。

```
-rw------- root root custom.crt
```

中间证书文件的权限并不严格 (所有者=根、组=根、所有者可以写入、组可以读取、任何人均可读取)。以下示例显示了要使用的命令。

```
[ec2-user certs]$ sudo chown root:root intermediate.crt
[ec2-user certs]$ sudo chmod 644 intermediate.crt
[ec2-user certs]$ ls -al intermediate.crt
```

这些命令应生成以下结果。

```
-rw-r--r-- root root intermediate.crt
```

6. 将用于创建 CSR 的私有密钥放在 /etc/pki/tls/private/ 目录中。

> **Note**
>
> 可通过多种方法将自定义密钥上传到 EC2 实例，但最直接、最有益的方法是在本地计算机和 EC2 实例上打开一个文本编辑器（例如，vi、nano 或记事本），然后在这两者之间复制并粘贴文件内容。在 EC2 实例上执行这些操作时，您需要根 [sudo] 权限。这样，一旦有任何权限或路径问题，您可以立即看到。但请小心操作，不要在复制内容时添加任何多余的行或以任何方式更改内容。

在/etc/pki/tls/private目录内部，使用以下命令验证文件所有权、组和权限设置是否与严格限制的 AL2 默认值相匹配（owner=root，group=root，仅适用于所有者）。 read/write

```
[ec2-user private]$ sudo chown root:root custom.key
[ec2-user private]$ sudo chmod 600 custom.key
[ec2-user private]$ ls -al custom.key
```

这些命令应生成以下结果。

```
-rw------- root root custom.key
```

7. 编辑 /etc/httpd/conf.d/ssl.conf 以反映您的新证书和密钥文件。

   a. 在 Apache 的SSLCertificateFile指令中提供 CA-signed 主机证书的路径和文件名：

   ```
   SSLCertificateFile /etc/pki/tls/certs/custom.crt
   ```

   b. 如果您收到一个中间证书文件（此示例中为 intermediate.crt），请使用 Apache 的 SSLCACertificateFile 指令提供其路径和文件名：

```
SSLCACertificateFile /etc/pki/tls/certs/intermediate.crt
```

> **Note**
>
> 一些 CA 将主机证书和中间证书合并到单个文件中，从而不再需要使用 SSLCACertificateFile 指令。请查询您的 CA 提供的说明。

c. 在 Apache 的 SSLCertificateKeyFile 指令中提供私有密钥的路径和文件名（在该示例中为 custom.key）：

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

8. 保存 /etc/httpd/conf.d/ssl.conf 并重启 Apache。

```
[ec2-user ~]$ sudo systemctl restart httpd
```

9. 通过在浏览器 URL 栏中输入带有 https:// 前缀的域名来测试您的服务器。您的浏览器应通过 HTTPS 加载测试页面而不会产生错误。

步骤 3：测试和强化安全配置

在 TLS 可操作且公开发布后，应测试其实际安全性。使用在线服务（例如 Qualys SSL Labs，该服务可对您的安全设置执行免费的全面分析）可轻松执行此操作。根据结果，您可以决定通过控制接受的协议、首选的密码和排除的密码来强化默认安全配置。有关更多信息，请参阅 Qualys 如何用公式表示其分数。

> **Important**
>
> Real-world 测试对服务器的安全至关重要。少量配置错误可能导致严重的安全漏洞和数据丢失。由于建议的安全实践会不断变化以响应调查和新兴威胁，因此定期安全审核对于良好的服务器管理来说是必不可少的。

在 Qualys SSL Labs 站点上，使用 **www.example.com** 格式输入服务器的完全限定域名。约两分钟后，您将收到您站点的评级 (从 A 到 F) 和结果的详细信息。下表汇总了设置与 AL2 上默认 Apache 配置相同且具有默认 Certbot 证书的域的报告。

总评	B
证书	100%
协议支持	95%
密钥交换	70%
密码强度	90%

虽然概述信息显示配置基本正确，但详细报告标记了几个潜在的问题（在此处按严重性顺序列出）：

✗ 支持某些旧浏览器使用 RC4 密码。密码是加密算法的数学核心。RC4 是一种用于加密 TLS 数据流的快速密码，已知这种密码存在一些严重缺点。除非您有充分理由支持旧版浏览器，否则，应禁用该密码。

✗ 支持旧 TLS 版本。该配置支持 TLS 1.0（已弃用）和 TLS 1.1（即将弃用）。从 2018 年开始，仅建议使用 TLS 1.2。

✗ 不完全支持向前保密性。向前保密性是一种算法功能，它使用从私有密钥派生的临时会话密钥进行加密。这意味着，在实践中，攻击者无法解密 HTTPS 数据，即使他们拥有 Web 服务器的长期私有密钥。

纠正 TLS 配置并供将来使用

1. 在文本编辑器中打开 `/etc/httpd/conf.d/ssl.conf` 配置文件，并在以下行的开头输入“#”以注释掉该行。

   ```
   #SSLProtocol all -SSLv3
   ```

2. 添加以下指令：

   ```
   #SSLProtocol all -SSLv3
   SSLProtocol -SSLv2 -SSLv3 -TLSv1 -TLSv1.1 +TLSv1.2
   ```

   该指令显式禁用 SSL 版本 2 和 3 以及 TLS 版本 1.0 和 1.1。现在，服务器拒绝接受与使用 TLS 1.2 以外的任何协议的客户端之间的加密连接。指令中的冗长文字更清楚地向人类读者阐述为服务器配置的用途。

> **Note**
>
> 以此方式禁用 TLS 1.0 和 1.1 版可阻止一小部分过时的 Web 浏览器访问您的网站。

修改允许的密码列表

1. 在 `/etc/httpd/conf.d/ssl.conf` 配置文件中，找到包含 **`SSLCipherSuite`** 指令的部分，并通过在现有行的开头输入“#”来注释掉该行。

```
#SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5
```

2. 指定显式的密码套件，并指定密码顺序以优先使用向前保密性并避免不安全的密码。此处使用的 `SSLCipherSuite` 指令基于 Mozilla SSL 配置生成器的输出，该生成器根据服务器上运行的特定软件定制 TLS 配置。首先，通过使用以下命令的输出确定 Apache 和 OpenSSL 版本。

```
[ec2-user ~]$ yum list installed | grep httpd

[ec2-user ~]$ yum list installed | grep openssl
```

例如，如果返回的信息是 Apache 2.4.34 和 OpenSSL 1.0.2，我们将其输入到生成器中。如果您选择“现代”兼容性模型，这将创建一条 `SSLCipherSuite` 指令，虽然该指令积极实施安全性，但仍适用于大多数浏览器。如果您的软件不支持现代配置，则可以更新软件或改为选择“中间”配置。

```
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
```

所选密码的名称中有 ECD HE，这是 Elliptic Curve Ephemeral 的缩写。 Diffie-Hellman术语 ephemeral 表示向前保密性。副作用是，这些密码不支持 RC4。

建议您使用密码的明确列表，而不是依赖于内容不可见的默认值或简短指令。

将生成的指令复制到 `/etc/httpd/conf.d/ssl.conf` 中。

> **Note**
>
> 此处为方便阅读将指令显示为几行，但在复制到 `/etc/httpd/conf.d/ssl.conf` 时，该指令必须位于一行中，并且密码名称之间只有一个冒号（无空格）。

3. 最后，通过删除以下行开头的“#”来取消对该行的注释。

```
#SSLHonorCipherOrder on
```

该指令强制服务器优先使用排名较高的密码，包括（在该示例中）支持向前保密性的密码。启用此指令后，服务器会在回滚到允许的安全性较低的密码之前尝试建立高度安全的连接。

在完成这两个过程后，将更改保存到 `/etc/httpd/conf.d/ssl.conf` 并重新启动 Apache。

如果在 Qualys SSL Labs 上再次测试域，将会看到已修复 RC4 漏洞和其他警告，并且摘要如下所示。

总评	A
证书	100%
协议支持	100%
密钥交换	90%
密码强度	90%

在每次更新 OpenSSL 时，将引入新的密码并删除对旧密码的支持。保持您的 EC2 AL2 实例处于最新状态，留意来自 Open SS L 的安全公告，并警惕技术媒体上有关新安全漏洞的报告。

故障排除

- 除非我输入密码，否则我的 Apache Web 服务器不会启动

  如果您安装了受密码保护的加密的私有服务器密钥，这是预期行为。

  您可以从密钥中删除加密和密码要求。假设在默认目录中具有一个称为 `custom.key` 的加密的私有 RSA 密钥，并且此密钥上的密码是 **abcde12345**，则对 EC2 实例运行以下命令可生成此密钥的未加密版本。

```
[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 现在启动时应该不会提示您提供密码。

- 我在运行 sudo yum install -y mod_ssl 时收到了错误。

  在为 SSL 安装所需的程序包时，您可能会看到与以下内容类似的错误。

  ```
  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
  ```

  这通常意味着您的 EC2 实例未运行 AL2。本教程仅支持从官方 AL2 AMI 新创建的实例。

教程：在 AL2 上发布 WordPress 博客

以下过程将帮助您在 AL2 实例上安装、配置和保护 WordPress 博客。本教程很好地介绍了如何使用 Amazon EC2，因为您可以完全控制托管 WordPress 博客的 Web 服务器，这在传统托管服务中并不常见。

您负责更新软件包并为您的服务器维护安全补丁。对于不需要与 Web 服务器配置直接交互的自动化程度更高的 WordPress 安装，该 CloudFormation 服务提供了一个可以帮助您快速入门的 WordPress模板。有关更多信息，请参阅AWS CloudFormation 用户指南中的入门。如果您需要具有分离数据库的高可用性解决方案，请参阅开发人员指南中的部署高可用性 WordPress 网站。AWS Elastic Beanstalk

> **Important**
>
> 这些程序适用于 AL2。有关其他发布版本的更多信息，请参阅其具体文档。本教程中的很多步骤对 Ubuntu 实例并不适用。有关在 Ubuntu 实例 WordPress 上安装的帮助，请参阅 Ubuntu 文档WordPress中的。你也可以使用CodeDeploy在亚马逊 Linux、macOS 或 Unix 系统上完成此任务。

主题

先决条件

本教程假设您已按照中的所有步骤启动了一个 AL2 实例，该服务器支持了 PHP 和数据库（MySQL 或 MariaDB），该服务器支持运行 PHP 和数据库（MySQL 或 MariaDB）。教程：在 AL2 上安装 LAMP 服务器本教程还介绍了配置安全组以允许 HTTP 和 HTTPS 流量的步骤，以及用于确保为 Web 服务器正确设置文件权限的几个步骤。有关向安全组添加规则的信息，请参阅向安全组添加规则。

我们强烈建议您将弹性 IP 地址 (EIP) 关联到用于托管 WordPress 博客的实例。这将防止您的实例的公有 DNS 地址更改和中断您的安装。如果您有一个域名且打算将其用于您的博客，则可更新该域名的 DNS 记录，使其指向您的 EIP 地址 (如需帮助，请联系您的域名注册商)。您可以免费将一个 EIP 地址与正在运行的实例相关联。有关更多信息，请参阅《Amazon EC2 用户指南》中的弹性 IP 地址。

如果您的博客还没有域名，则可使用 Route 53 注册一个域名并将您的实例的 EIP 地址与您的域名相关联。有关更多信息，请参阅 Amazon Route 53 开发人员指南中的使用 Amazon Route 53 注册域名。

安装 WordPress

选项：使用 Automation 完成本教程

要使用 AWS Systems Manager 自动化而不是以下任务来完成本教程，请运行自动化文档。

连接到您的实例，然后下载 WordPress 安装包。

下载并解压缩 WordPress 安装包

1. 使用wget命令下载最新的 WordPress 安装包。以下命令始终会下载最新版本。

```
[ec2-user ~]$ wget https://wordpress.org/latest.tar.gz
```

2. 解压并解档安装包。安装文件夹解压到名为 `wordpress` 的文件夹。

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

为您的 WordPress 安装创建数据库用户和数据库

您的 WordPress 安装需要将博客文章和用户评论等信息存储在数据库中。此过程帮助您创建自己的博客数据库，并创建一个有权读取该数据库的信息并将信息保存到该数据库的用户。

1. 启动数据库服务器。

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

2. 以 root 用户身份登录数据库服务器。在系统提示时输入您的数据库 root 密码，它可能与您的 root 系统密码不同；如果您尚未给您的数据库服务器加密，它甚至可能是空的。

   如果您尚未给您的数据库服务器加密，则必须执行这项操作。有关更多信息，请参阅 保护 MariaDB 服务器 (AL2)。

   ```
   [ec2-user ~]$ mysql -u root -p
   ```

3. 为您的 MySQL 数据库创建用户和密码。您的 WordPress安装使用这些值与您的 MySQL 数据库进行通信。

   确保为您的用户创建强密码。请勿在您的密码中使用单引号字符 (')，因为这将中断前面的命令。请勿重复使用现有密码，并确保将密码保存在安全的位置。

   输入以下命令，以替换唯一的用户名和密码。

   ```
   CREATE USER 'wordpress-user'@'localhost' IDENTIFIED BY 'your_strong_password';
   ```

4. 创建数据库。为数据库提供一个有意义的描述性名称，例如 wordpress-db。

   > **Note**
   >
   > 以下命令中数据库名称两边的标点符号称为反引号。在标准键盘上，反引号 (`) 键通常位于 Tab 键的上方。并不总是需要反引号，但是它们允许您在数据库名称中使用其他的非法字符，例如连字符。

   ```
   CREATE DATABASE `wordpress-db`;
   ```

5. 向之前创建的 WordPress 用户授予数据库的完全权限。
   ```

```
GRANT ALL PRIVILEGES ON `wordpress-db`.* TO "wordpress-user"@"localhost";
```

6. 刷新数据库权限以接受您的所有更改。

```
FLUSH PRIVILEGES;
```

7. 退出 mysql 客户端。

```
exit
```

创建和编辑 wp-config.php 文件

WordPress 安装文件夹包含一个名为的示例配置文件wp-config-sample.php。在本步骤中，您将复制此文件并进行编辑以适合您的具体配置。

1. 将 wp-config-sample.php 文件复制为一个名为 wp-config.php 的文件。这样做会创建新的配置文件并将原先的示例配置文件原样保留作为备份。

```
[ec2-user ~]$ cp wordpress/wp-config-sample.php wordpress/wp-config.php
```

2. 用您喜欢的文本编辑器（例如 nano 或 vim）编辑 wp-config.php 文件并输入适用于您的安装的值。如果没有常用的文本编辑器，nano 比较适合初学者使用。

```
[ec2-user ~]$ nano wordpress/wp-config.php
```

   a. 查找定义 DB_NAME 的行并将 database_name_here 更改为您在 Step 4 的 为您的 WordPress 安装创建数据库用户和数据库 中创建的数据库名称。

```
define('DB_NAME', 'wordpress-db');
```

   b. 查找定义 DB_USER 的行并将 username_here 更改为您在 Step 3 的 为您的 WordPress 安装创建数据库用户和数据库 中创建的数据库用户。

```
define('DB_USER', 'wordpress-user');
```

   c. 查找定义 DB_PASSWORD 的行并将 password_here 更改为您在 Step 3 的 为您的 WordPress 安装创建数据库用户和数据库 中创建的强密码。

```
define('DB_PASSWORD', 'your_strong_password');
```

d. 查找名为 Authentication Unique Keys and Salts 的一节。这些KEY和SALT值为 WordPress 用户存储在本地计算机上的浏览器 Cookie 提供了一层加密。总而言之，添加长的随机值将使您的站点更安全。访问https://api.wordpress.org/secret-key/1.1/salt/以随机生成一组密钥值，您可以将其复制并粘贴到wp-config.php文件中。要粘贴文本到 PuTTY 终端，请将光标放在您要粘贴文本的地方，并在 PuTTY 终端内部右键单击鼠标。

有关安全密钥的更多信息，请访问 https://wordpress.org/support/article/editing-wp-config-php/#security-ke ys。

> **Note**
>
> 以下值仅用作示例；请勿使用以下值进行安装。

```
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. 保存文件并退出文本编辑器。

将 WordPress 文件安装在 Apache 文档根目录下

- 现在，您已经解压缩了安装文件夹，创建了 MySQL 数据库和用户，并自定义了 WordPress 配置文件，接下来就可以将安装文件复制到 Web 服务器文档根目录了，这样就可以运行完成安装

的安装脚本了。这些文件的位置取决于您是希望 WordPress 博客在 Web 服务器的实际根目录（例如*my.public.dns.amazonaws.com*）中可用，还是在根目录下的子目录或文件夹（例如*my.public.dns.amazonaws.com/blog*）中可用。

- 如果 WordPress 要在文档根目录下运行，请按如下方式复制 wordpress 安装目录的内容（但不是目录本身）：

```
[ec2-user ~]$ cp -r wordpress/* /var/www/html/
```

- 如果 WordPress 要在文档根目录下的备用目录中运行，请先创建该目录，然后将文件复制到该目录。在此示例中， WordPress将从以下目录运行blog：

```
[ec2-user ~]$ mkdir /var/www/html/blog
[ec2-user ~]$ cp -r wordpress/* /var/www/html/blog/
```

> **Important**
>
> 出于安全原因，如果您不打算立即进入到下一个过程，请立即停止 Apache Web 服务器 (httpd)。将安装移至 Apache 文档根目录下后， WordPress 安装脚本不受保护，如果 Apache Web 服务器正在运行，攻击者可能会访问您的博客。要终止 Apache Web 服务器，请输入命令 sudo systemctl stop httpd。如果您即将继续到下一个步骤，则不需要终止 Apache Web 服务器。

允许 WordPress 使用永久链接

WordPress 永久链接需要使用 Apache .htaccess 文件才能正常工作，但是 Amazon Linux 上默认不启用此功能。使用此过程可允许 Apache 文档根目录中的所有覆盖。

1. 使用您常用的文本编辑器（如 vim 或 nano）打开 httpd.conf 文件。如果没有常用的文本编辑器，nano 比较适合初学者使用。

```
[ec2-user ~]$ sudo vim /etc/httpd/conf/httpd.conf
```

2. 找到以 <Directory "/var/www/html"> 开头的部分。

```
<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. 在以上部分中将 AllowOverride None 行改为读取 AllowOverride *All*。

> **Note**
>
> 此文件中有多个 AllowOverride 行；请确保更改 <Directory "/var/www/html"> 部分中的行。

```
AllowOverride All
```

4. 保存文件并退出文本编辑器。

在 AL2 上安装 PHP 图形绘图库

PHP 的 GD 库允许您修改图像。如果您需要裁剪博客的标题图像，请安装此库。您安装MyAdmin 的 php 版本可能需要此库的特定最低版本（例如，版本 7.2）。

使用以下命令在 AL2 上安装 PHP 图形绘图库。例如，如果您在安装 LAMP 堆栈的过程中从 amazon-linux-extras 安装了 php7.2，则此命令将安装 7.2 版的 PHP 图形绘图库。

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

要验证安装的版本，请使用以下命令：

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

下面是示例输出：

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

修复 Apache Web 服务器的文件权限

中的某些可用功能 WordPress 需要对 Apache 文档根目录具有写入权限（例如通过 “管理” 屏幕上传媒体）。如果您尚未这样做，请应用以下群组成员资格和权限（详见中教程：在 AL2 上安装 LAMP 服务器）。

1. 将 /var/www 及其内容的文件所有权授予 apache 用户。

   ```
   [ec2-user ~]$ sudo chown -R apache /var/www
   ```

2. 将 /var/www 及其内容的组所有权授予 apache 组。

   ```
   [ec2-user ~]$ sudo chgrp -R apache /var/www
   ```

3. 更改 /var/www 及其子目录的目录权限，以添加组写入权限及设置未来子目录上的组 ID。

   ```
   [ec2-user ~]$ sudo chmod 2775 /var/www
   [ec2-user ~]$ find /var/www -type d -exec sudo chmod 2775 {} \;
   ```

4. 递归地更改 /var/www 及其子目录的文件权限。

   ```
   [ec2-user ~]$ find /var/www -type f -exec sudo chmod 0644 {} \;
   ```

> **Note**
>
> 如果您还打算用 WordPress 作 FTP 服务器，则需要在此处进行更宽松的群组设置。请查看中的建议步骤和安全设置 WordPress以完成此操作。

5. 重启 Apache Web 服务器，让新组和权限生效。

   - ```
     [ec2-user ~]$ sudo systemctl restart httpd
     ```

使用 AL2 运行 WordPress 安装脚本

您已准备好安装 WordPress。您使用的命令取决于操作系统。此过程中的命令适用于 AL2。

1. 使用 systemctl 命令确保 `httpd` 和数据库服务在每次系统启动时启动。

   ```
   [ec2-user ~]$ sudo systemctl enable httpd && sudo systemctl enable mariadb
   ```

2. 验证数据库服务器是否正在运行。

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

   如果数据库服务未运行，请启动。

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

3. 验证您的 Apache Web 服务器 (`httpd`) 正在运行。

   ```
   [ec2-user ~]$ sudo systemctl status httpd
   ```

   如果 `httpd` 服务未运行，请启动。

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

4. 在 Web 浏览器中，键入 WordPress 博客的 URL（要么是您的实例的公有 DNS 地址，要么是该地址后跟`blog`文件夹）。您应该会看到 WordPress 安装脚本。提供 WordPress 安装所需的信息。选择安装 WordPress以完成安装。有关更多信息，请参阅 WordPress 网站上的步骤 5：运行安装脚本。
   ```

后续步骤

测试完 WordPress 博客后，可以考虑更新其配置。

使用自定义域名

如果您有一个与您的 EC2 实例的 EIP 地址关联的域名，则可将您的博客配置为使用该域名而不是 EC2 公有 DNS 地址。有关更多信息，请参阅更改网站上的 WordPress 网站 URL。

配置您的博客

您可以将您的博客配置为使用不同的主题和插件，从而向您的读者提供更具个性化的体验。但是，有时安装过程可能事与愿违，从而导致您丢失您的整个博客。强烈建议您在尝试安装任何主题或插件之前，为您的实例创建一个备份 Amazon Machine Image (AMI)，以便在安装过程中出现任何问题时，您还可以还原您的博客。有关更多信息，请参阅创建自己的 AMI。

增加容量

如果您的 WordPress 博客越来越受欢迎，并且您需要更多的计算能力或存储空间，请考虑以下步骤：

- 对实例扩展存储空间。有关更多信息，请参阅《Amazon EBS 用户指南》中的 Amazon EBS 弹性卷。
- 将您的 MySQL 数据库移动到 Amazon RDS 以利用服务的轻松扩展功能。

提高互联网流量的网络性能

如果希望您的博客吸引世界各地用户的流量，请考虑 AWS Global Accelerator。Global Accelerator 通过改善用户的客户端设备和运行的 WordPress 应用程序之间的互联网流量性能，帮助您降低延迟 AWS。Global Accelerator 使用AWS 全球网络将流量引导到离客户端最近的 AWS 区域中运行良好的应用程序终端节点。

了解更多关于 WordPress

有关信息 WordPress，请参阅 WordPress Codex 帮助文档，网址为。http://codex.wordpress.org/

有关安装疑难解答的更多信息，请参阅常见安装问题。

有关提高 WordPress 博客安全性的信息，请参阅强化 WordPress。

有关保持 WordPress 博客最新状态的信息，请参阅更新 WordPress。

帮助！我的公有 DNS 名称发生更改导致我的博客瘫痪

您的 WordPress 安装是使用您的 EC2 实例的公有 DNS 地址自动配置的。如果您停止并重启实例，公有 DNS 地址将发生更改 (除非它与弹性 IP 地址相关联)，并且您的博客将不会再运行，因为您的博客引用了不再存在的地址 (或已分配给另一个 EC2 实例的地址) 上的资源。更改站点 URL 中概述了更详细的问题描述和几种可能的解决方案。

如果 WordPress 安装时发生了这种情况，则可以通过以下步骤恢复博客，该过程使用wp-cli命令行界面 WordPress。

要更改您的 WordPress 网站网址，请使用 wp-cli

1. 使用 SSH 连接到您的 EC2 实例。
2. 请记下您的实例的旧站点 URL 和新站点 URL。安装时，旧站点 URL 很可能是您的 EC2 实例的公有 DNS 名称 WordPress。新站点 URL 是您的 EC2 实例的当前公有 DNS 名称。如果您不确定旧站点 URL 是什么，则可通过以下命令使用 curl 来查找它。

   ```
   [ec2-user ~]$ curl localhost | grep wp-content
   ```

   您应该会在输出中看到对您的旧公有 DNS 名称的引用，如下所示 (旧站点 URL 用红色表示)：

   ```
   <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. 使用以下命令下载 wp-cli。

   ```
   [ec2-user ~]$ curl -O https://raw.githubusercontent.com/wp-cli/builds/gh-pages/
   phar/wp-cli.phar
   ```
4. 使用以下命令搜索并替换 WordPress 安装中的旧站点 URL。用新旧站点 URL 替换您的 EC2 实例和 WordPress 安装路径 (通常为 o /var/www/html r/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 浏览器中，输入 WordPress 博客的新网站 URL，以验证该网站是否恢复正常运行。如果不是，请参阅更改站点 URL 和常见安装问题了解更多信息。

在亚马逊 EC2 之外使用亚马逊 Linux 2

AL2 容器镜像可以在兼容的容器运行时环境中运行。

除了直接在 Amazon EC2 上运行之外，AL2 也可以作为虚拟客户机运行。

> **Note**
>
> AL2 图像的配置不同于 AL2023。
> 迁移到时 AL2023，请务必查看在亚马逊 EC2 之外使用亚马逊 Linux 2023，并调整您的配置以使其与之兼容 AL2023。

在本地 AL2 作为虚拟机运行

使用 AL2 虚拟机 (VM) 映像进行本地开发和测试。我们为每个支持的 AL2 虚拟化平台提供不同的虚拟机映像。您可以在 Amazon Linux 2 虚拟机映像页面查看支持的平台列表。

要将 AL2 虚拟机映像与支持的虚拟化平台之一配合使用，请执行以下操作：

- 步骤 1：准备 seed.iso 启动映像
- 步骤 2：下载 AL2 虚拟机镜像
- 步骤 3：启动并连接到新 VM

步骤 1：准备 `seed.iso` 启动映像

`seed.iso` 启动映像包含启动新虚拟机所需的初始配置信息，如网络配置、主机名和用户数据。

> **Note**
>
> `seed.iso` 启动映像仅包括启动 VM 所需的配置信息。它不包括 AL2 操作系统文件。

要生成 `seed.iso` 启动映像，需要两个配置文件：

- `meta-data` – 此文件包括 VM 的主机名和静态网络设置。
- `user-data` - 此文件配置用户账户，并指定其密码、密钥对以及访问机制。默认情况下，AL2 虚拟机映像会创建一个`ec2-user`用户帐户。使用 `user-data` 配置文件设置默认用户账户的密码。

创建 seed.iso 启动盘

1. 创建一个名为 seedconfig 的新文件夹并导航到该文件夹。
2. 创建 meta-data 配置文件。

   a. 创建名为 meta-data 的新文件。

   b. 使用首选编辑器打开 meta-data 文件，并添加以下内容。

```
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
```

   *vm_hostname*替换为您选择的虚拟机主机名，并根据需要配置网络设置。

   c. 保存并关闭 meta-data 配置文件。

   有关示例 meta-data 配置文件（用于指定 VM 主机名 (amazonlinux.onprem)、配置默认网络接口 (eth0) 并为必要的网络设备指定静态 IP 地址），请参阅示例 Seed.iso 文件。

3. 创建 user-data 配置文件。

   a. 创建名为 user-data 的新文件。

   b. 使用首选编辑器打开 user-data 文件，并添加以下内容。

```
#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:'.
```

*plain_text_password*使用您为默认ec2-user用户帐户选择的密码替换。

c. （可选）默认情况下，VM 每次启动时，cloud-init 都会应用网络设置。添加以下内容，以防止 cloud-init 在每次启动时都应用网络设置，并保留首次启动期间应用的网络设置。

```
# 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. 保存并关闭 user-data 配置文件。

还可以创建其他用户账户并指定其访问机制、密码和密钥对。有关受支持指令的更多信息，请参阅模块参考。有关创建三个其他用户并为默认 user-data 用户账户指定自定义密码的示例 ec2-user 文件，请参阅示例 Seed.iso 文件。

4. 使用 seed.iso 和 meta-data 配置文件创建 user-data 启动映像。

对于 Linux，请使用类似 genisoimage 的工具。导航到 seedconfig 文件夹，并运行以下命令。

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

对于 macOS，请使用类似 hdiutil 的工具。从 seedconfig 文件夹往上导航一级，运行以下命令。

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

步骤 2：下载 AL2 虚拟机镜像

我们为每个支持的 AL2 虚拟化平台提供不同的虚拟机映像。您可以查看支持的平台列表，并在 Amazon Linux 2 虚拟机映像页面中为所选平台下载对应的 VM 映像。

步骤 3：启动并连接到新 VM

要启动并连接到您的新虚拟机，您必须拥有`seed.iso`启动映像（在步骤 1 中创建）和 AL2 虚拟机映像（在步骤 2 中下载）。具体步骤因您选择的 VM 平台而异。

VMware vSphere

的虚拟机 VMware 映像以 OVF 格式提供。

使用 VMware vSphere 启动虚拟机

1. 为 `seed.iso` 文件创建新的数据存储，或将其添加到现有的数据存储中。
2. 部署 OVF 模板，但先不启动虚拟机。
3. 在 Navigator (导航器) 面板中，右键单击新虚拟机，然后选择 Edit Settings (编辑设置)。
4. 在 Virtual Hardware (虚拟硬件) 选项卡上，为 New device (新设备) 选择 CD/DVD Drive (CD/DVD 驱动器)，然后选择 Add (添加)。
5. 对于 “新建 CD/DVD 驱动器”，选择 “数据存储 ISO 文件”。选择将 `seed.iso` 文件添加到的数据存储，浏览到并选择 `seed.iso` 文件，然后选择 OK (确定)。
6. 对于 “新建 CD/DVD 驱动器”，选择 Connect，然后选择 “确定”。

在将数据存储与虚拟机关联后，您应该能够引导该虚拟机。

KVM

使用 KVM 引导虚拟机

1. 打开 Create new VM (创建新的虚拟机) 向导。
2. 对于步骤 1，选择 Import existing disk image (导入现有的磁盘映像)。
3. 对于步骤 2，浏览到并选择虚拟机映像。对于 OS type（操作系统类型）和 Version（版本），分别选择 Linux 和 Red Hat Enterprise Linux 7.0。
4. 在步骤 3 中，指定 CPUs要使用的内存量和数量。
5. 对于步骤 4，输入新虚拟机的名称，然后选择 Customize configuration before install (安装前自定义配置)，然后选择 Finish (完成)。
6. 在虚拟机的 Configuration (配置) 窗口中，选择 Add Hardware (添加硬件)。
7. 在 Add New Virtual Hardware (添加新的虚拟硬件) 窗口中，选择 Storage (存储)。
8. 在 Storage configuration (存储配置) 中，选择 Select or create custom storage (选择或创建自定义存储)。对于 Device type (设备类型)，选择 CDROM device (CDROM 设备)。选择

Manage (管理)，选择 Browse Local (浏览本地)，然后导航到并选择 `seed.iso` 文件。选择 Finish。

9. 选择 Begin Installation (开始安装)。

Oracle VirtualBox

使用 Oracle 启动虚拟机 VirtualBox

1. 打开 Oracle VirtualBox 并选择新建。
2. 对于 Name（名称），输入虚拟机的描述性名称，对于 Type（类型）和 Version（版本），分别选择 Linux 和 Red Hat (64-bit)（Red Hat（64 位））。选择 Continue (继续)。
3. 对于内存大小，请指定要分配给虚拟机的内存容量，然后选择继续。
4. 对于硬盘，选择使用现有虚拟硬盘文件，浏览并打开虚拟机映像，然后选择创建。
5. 在启动虚拟机之前，您必须在虚拟机的虚拟光驱中加载 `seed.iso` 文件：

   a. 选择新的虚拟机，选择 Settings (设置)，然后选择 Storage (存储)。
   b. 在存储设备列表中，在控制器: IDE 下选择空的光驱。
   c. 在光驱的 Attributes (属性) 部分中，选择浏览按钮，选择 Choose Virtual Optical Disk File (选择虚拟光盘文件)，然后选择 `seed.iso` 文件。选择确定，以应用更改并关闭“设置”。

在将 `seed.iso` 文件添加到虚拟光驱后，您应该能够启动该虚拟机。

Microsoft Hyper-V

Microsoft Hyper-V 的虚拟机映像压缩为一个 zip 文件。您必须提取该 zip 文件的内容。

使用 Microsoft Hyper-V 引导虚拟机

1. 打开新建虚拟机向导。
2. 在提示您选择代时，选择 Generation 1 (第一代)。
3. 在提示您配置网络适配器时，为 Connection (连接) 选择 External (外部)。
4. 在提示您连接虚拟硬盘时，选择 Use an existing virtual hard disk (使用现有的虚拟硬盘)，选择 Browse (浏览)，然后导航到并选择虚拟机映像。选择 Finish (完成) 以创建虚拟机。
5. 右键单击新虚拟机，然后选择 Settings (设置)。在 Settings (设置) 窗口中，在 IDE Controller 1 (IDE 控制器 1) 下面选择 DVD Drive (DVD 驱动器)。
6. 对于 DVD 驱动器，选择 Image file (映像文件)，然后浏览到并选择 `seed.iso` 文件。

7. 应用更改并启动虚拟机。

在 VM 启动后，使用在 user-data 配置文件中定义的用户账户之一登录。在您第一次登录之后，就可以断开 seed.iso 启动映像与 VM 的连接。

识别 Amazon Linux 实例和版本

能够确定操作系统映像或实例属于哪个 Linux 发行版及其版本可能非常重要。Amazon Linux 提供了相应的机制来将其与其他 Linux 发行版区分开，并识别映像所属的 Amazon Linux 发布版本。

这部分将介绍可用的不同方法、它们的局限性，并通过一些使用示例进行说明。

主题

使用 `os-release` 标准

Amazon Linux 遵循用于识别 Linux 发行版的 `os-release` 标准。该文件提供关于操作系统标识和版本信息的机器可读信息。

> **Note**
>
> 该标准规定首先尝试解析 `/etc/os-release`，其次是 `/usr/lib/os-release`。应注意遵循有关文件名和路径的标准。

主题

关键识别差异

`os-release` 位于 `/etc/os-release`，如果该位置不存在，则位于 `/usr/lib/os-release`。完整信息请查阅 `os-release` 标准。

确定实例是否运行 Amazon Linux 最可靠的方法是检查 `os-release` 中的 `ID` 字段。

区分不同版本最可靠的方法是检查 os-release 中的 VERSION_ID 字段：

- Amazon Linux AMI：VERSION_ID 包含基于日期的版本（例如 2018.03）
- AL2: VERSION_ID="2"
- AL2023: VERSION_ID="2023"

> **Note**
>
> 请记住，VERSION_ID 是一个供编程使用的机器可读字段，而 PRETTY_NAME 是为向用户显示而设计的。有关字段类型的更多信息，请参阅 the section called “字段类型”。

字段类型：机器可读与人类可读

/etc/os-release 文件（或者如果 /etc/os-release 不存在，则为 /usr/lib/os-release 文件）包含两种类型的字段：供编程使用的机器可读字段，以及供向用户呈现信息用的人类可读字段。

机器可读字段

这些字段使用标准化格式，旨在供脚本、程序包管理器和其他自动化工具处理。它们仅包含小写字母、数字和有限的标点符号（句点、下划线和连字符）。

- ID：操作系统标识符。Amazon Linux 在所有版本中使用 amzn，以此区别于其他发行版，如 Debian (debian)、Ubuntu (ubuntu) 或 Fedora (fedora)
- VERSION_ID：供编程使用的操作系统版本（例如 2023）
- ID_LIKE：相关发行版的空间分隔列表（例如 fedora）
- VERSION_CODENAME：供脚本使用的发布代号（例如 karoo）
- VARIANT_ID：用于编程决策的变体标识符
- BUILD_ID：系统映像的构建标识符
- IMAGE_ID：容器化环境的映像标识符
- PLATFORM_ID：平台标识符（例如 platform:al2023）

人类可读字段

这些字段旨在向用户显示，可能包含空格、混合大小写和描述性文本。在用户界面中呈现操作系统信息时应使用它们。

- NAME：用于显示的操作系统名称（例如 Amazon Linux）
- PRETTY_NAME：用于显示的包含版本的完整操作系统名称（例如 Amazon Linux 2023.8.20250721）
- VERSION：适合向用户呈现的版本信息
- VARIANT：用于显示的变体或版本名称（例如 Server Edition）

其他信息字段

这些字段提供有关操作系统的额外元数据：

- HOME_URL：项目主页 URL
- DOCUMENTATION_URL：文档 URL
- SUPPORT_URL：支持信息 URL
- BUG_REPORT_URL：错误报告 URL
- VENDOR_NAME：供应商名称
- VENDOR_URL：供应商 URL
- SUPPORT_END— YYYY-MM-DD 格式中的 End-of-support日期
- CPE_NAME：通用平台枚举标识符
- ANSI_COLOR：用于终端显示的 ANSI 颜色代码

当编写需要以编程方式识别 Amazon Linux 的脚本或应用程序时，请使用机器可读字段，如 ID 和 VERSION_ID。当向用户显示操作系统信息时，请使用人类可读字段，如 PRETTY_NAME。

/etc/os-release 示例

/etc/os-release 文件内容在 Amazon Linux 各版本间有所不同：

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/"
```

与其他发行版的比较

要了解 Amazon Linux 在更广泛的 Linux 生态系统中的位置，可将其 `/etc/os-release` 格式与其他主要发行版进行比较：

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
```

请注意机器可读字段如何提供跨发行版的一致标识：

- ID：唯一标识操作系统：Amazon Linux 为 amzn，Fedora 为 fedora，Debian 为 debian，Ubuntu 为 ubuntu
- ID_LIKE— 显示发行关系：亚马逊 Linux 使用 fedora (AL2023) 或 centos rhel fedora (AL2)，而 Ubuntu 则显示了其 Debi debian an 传统
- VERSION_ID— 提供机器可解析的版本信息：2023 AL2023、Fedora、Debian、42 Ubuntu 12 24.04

相比之下，人类可读字段专为向用户显示而设计：

- NAME：用户友好的操作系统名称：Amazon Linux、Fedora Linux、Debian GNU/Linux、Ubuntu

- PRETTY_NAME：包含版本的完整显示名称：Amazon Linux 2023.8.20250721、Fedora Linux 42 (Container Image)、Debian GNU/Linux 12 (bookworm)、Ubuntu 24.04.2 LTS
- VERSION：具有附加上下文（如代号或发布类型）的人类可读版本

编写跨平台脚本时，应始终使用机器可读字段（ID、VERSION_ID、ID_LIKE）进行逻辑判断和决策，并仅使用人类可读字段（PRETTY_NAME、NAME）向用户显示信息。

Amazon Linux 特有文件

有一些特定于 Amazon Linux 的文件可用于识别 Amazon Linux 及其版本。新代码应使用 /etc/os-release 标准以实现跨发行版兼容。不鼓励使用任何 Amazon Linux 特有文件。

主题

- /etc/system-release 文件
- 映像标识文件
- Amazon Linux 特有文件示例

/etc/system-release 文件

Amazon Linux 包含 /etc/system-release 文件，用于指定当前已安装的版本。此文件通过程序包管理器更新，在 Amazon Linux 中是 system-release 程序包的一部分。虽然 Fedora 等其他发行版也有此文件，但基于 Debian 的发行版（如 Ubuntu）中不存在。

> **Note**
>
> /etc/system-release 文件包含一个人类可读的字符串，不应以编程方式用于识别操作系统或发布版本。请改用 /etc/os-release（或者如果 /etc/os-release 不存在，则使用 /usr/lib/os-release）中的机器可读字段。

Amazon Linux 还在 /etc/system-release-cpe 文件中包含遵循通用平台枚举（CPE）规范的 /etc/system-release 的机器可读版本。

映像标识文件

每个 Amazon Linux 映像都包含一个唯一的 `/etc/image-id` 文件，该文件提供有关 Amazon Linux 团队生成的原始映像的附加信息。此文件特定于 Amazon Linux，在其他 Linux 发行版（如 Debian、Ubuntu 或 Fedora）中找不到。此文件包含有关映像的以下信息：

- `image_name`、`image_version`、`image_arch`：来自用于构建该映像的构建配方中的值。
- `image_stamp` - 映像创建期间随机生成的一个唯一的十六进制值。
- `image_date`— 图像创建的 UTC 时间，以YYYYMMDDhhmmss格式表示。
- `recipe_name`、`recipe_id`：用于构建该映像的构建配方的名称和 ID。

Amazon Linux 特有文件示例

以下部分提供每个主要 Amazon Linux 版本的 Amazon Linux 特有标识文件示例。

> **Note**
>
> 在任何实际代码中，如果 `/etc/os-release` 文件不存在，则应使用 `/usr/lib/os-release`。

AL2023

以下示例显示了 AL2 023 的标识文件。

AL2023 `/etc/image-id` 的示例：

```
[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"
```

AL2023 `/etc/system-release` 的示例：

```
[ec2-user ~]$ cat /etc/system-release
```

```
Amazon Linux release 2023.8.20250721 (Amazon Linux)
```

AL2

以下示例显示了的标识文件 AL2。

的/etc/image-id示例 AL2：

```
[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"
```

的/etc/system-release示例 AL2：

```
[ec2-user ~]$ cat /etc/system-release
```

```
Amazon Linux release 2 (Karoo)
```

Amazon Linux AMI

以下示例显示 Amazon Linux AMI 的标识文件。

Amazon Linux AMI 的 /etc/image-id 示例：

```
[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"
```

Amazon Linux AMI 的 /etc/system-release 示例：

```
[ec2-user ~]$ cat /etc/system-release
```

```
Amazon Linux AMI release 2018.03
```

操作系统检测示例代码

以下示例演示如何使用 /etc/os-release 文件（或者如果 /etc/os-release 不存在，则使用 /usr/lib/os-release 文件）以编程方式检测操作系统和版本。这些示例展示如何区分 Amazon Linux 与其他发行版，以及如何使用 ID_LIKE 字段来确定发行版系列。

下面的脚本以几种不同的编程语言实现，每种实现都会产生相同的输出。

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";
    }
}
```

在不同系统上运行时，该脚本将产生以下输出：

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)
```

AWS整合在 AL2

AWS命令行工具

AWS Command Line Interface(AWS CLI) 是一个开源工具，它提供了一个一致的接口，便于AWS 服务使用命令行外壳中的命令进行交互。有关更多信息，请参阅什么是AWS Command Line Interface？在《AWS Command Line Interface用户指南》中。

AL2 并AWS CLI预装 AL1 了版本 1。当前版本的亚马逊 Linux AL2 023 AWS CLI 预装了第 2 版。有关AWS CLI在 AL2 023 上使用，请参阅《亚马逊 Linux 2023 用户 AL2指南》中的 023 入门。

编程运行时入门

AL2 提供了某些语言运行时的不同版本。我们使用同时支持多个版本的上游项目，例如 PHP。要查找有关如何安装和管理这些按名称版本控制的软件包的信息，请使用yum命令搜索并安装这些软件包。有关更多信息，请参阅 程序包存储库。

以下主题描述了每种语言的运行时在中是如何运行的 AL2。

主题

C、C++，然后Fortran在 AL2

AL2 包括 GNU 编译器集合 (GCC) 和的前Clang端。LLVM

的主要版本GCC将在的整个生命周期中保持不变 AL2。错误和安全修复可能会向后移植到该版本的主要GCC版本中。 AL2

默认情况下，包 AL2 含 7.3 版，GCC该版本几乎可以生成所有软件包。该gcc10软件包在有限的范围内提供 GCC 10 个，但我们不建议使用 GCC 10 来构建软件包。

构建的默认编译器标志 AL2 RPMs 包括一些优化和强化标志。如果您要使用构建自己的代码，我们建议您添加一些优化和强化标志。GCC

AL2023 中的默认编译器和优化标志比中 AL2存在的内容有所改进。

进去吧 AL2

您可能需要使用随 AL2附的工具链在 Amaz Goon Linux 上编写自己的代码。

Go工具链将在整个生命周期中进行更新。 AL2这可能是对我们提供的工具链中的任何 CVE 的响应，或者作为在另一个包中解决 CVE 的先决条件。

Go是一种发展速度相对较快的编程语言。可能会出现用 Go 编写的现有应用程序必须适应新版本 Go 工具链的情况。有关 Go 的更多信息，请参阅 Go 1 与 Go 程序的未来。

尽管 AL2 将在Go工具链的生命周期中加入新版本，但这不会与上游Go版本保持一致。因此，如果您想使用Go语言和标准库的尖端功能来构建Go代码，则 AL2 可能不适合使用中提供的Go工具链。

在的生命周期内 AL2，较早的软件包版本不会从存储库中删除。如果需要较早的Go工具链，则可以选择放弃新Go工具链的错误和安全修复，使用适用于任何 RPM 的相同机制从存储库中安装较早的版本。

如果你想在上面构建自己的Go代码， AL2 你可以使用中包含的Go工具链，因为你知道这个工具链可能会在 AL2 整个生命周期中向前发展。 AL2

Java在 AL2

AL2 提供了多个版本的 Amazon Cor retto 来Java支持基于的工作负载，还有一些版本。OpenJDK我们建议您迁移到 Amazon Corretto，为迁移到 023 做准备。 AL2

Corretto 是一个由 Amazon 提供长期支持的 Open Java Development Kit（OpenJDK）构建版本。Corretto 已使用 Java 技术兼容性套件 (TCK) 进行认证，以确保其符合 SE 标准Java，并且已在 Linux、和上线。Windows macOS

Corretto 1.8.0、Corretto 11 和 Corretto 17 分别提供亚马逊 Corretto 套餐。

AL2 中的每个 Corretto 版本的支持时间都与 Corretto 版本相同，或者直到其生命周期结束，以较早者为准。 AL2有关更多信息，请参阅 Amazon Corretto。 FAQs

Perl在 AL2

AL2 提供了 5.16 版的Perl编程语言。

Perl中的模块 AL2

各种Perl模块的封装 RPMs 方式如下 AL2。尽管有许多可用的Perl模块 RPMs，但 Amazon Linux 不会尝试打包所有可能的Perl模块。模块打包为其他操作系统 RPM 包 RPMs 可能依赖的模块，因此 Amazon Linux 将优先确保它们经过安全修补，而不是纯粹的功能更新。

AL2 还包括，CPAN以便Perl开发人员可以将惯用包管理器用于Perl模块。

PHP在 AL2

AL2 目前提供了两个完全支持的PHP编程语言版本作为其中的一部分AL2 额外内容库。每个PHP版本的支持时间范围与上游PHP版本相同，如中的 亚马逊 Linux 2 额外内容清单 “已弃用日期” 下所列。

有关如何使用 E AL2 xtras 在您的实例上安装应用程序和软件更新的信息，请参阅AL2 额外内容库。

为了帮助迁移到 AL2 023，PHP8.1 和 8.2 均在 AL2 和 AL2 023 上可用。

> **Note**
>
> AL2 包括 PHP 7.1、7.2、7.3 和 7.4 英寸`amazon-linux-extras`。所有这些额外功能均已停产，不保证会获得任何其他安全更新。
> 要了解中何时弃用了PHP的每个版本 AL2，请参阅。亚马逊 Linux 2 额外内容清单

从较早的 PHP 8.x 版本迁移

上游PHP社区整理了从 PHP 8. PHP 1 迁移到 8.2 的全面迁移文档。还有关于从 PHP 8.0 迁移到 8.1 的文档。

AL2 中包括 PHP 8.0、8.1 和 8.2`amazon-linux-extras`，可实现到 AL2 023 的有效升级路径。要了解中何时弃用了PHP的每个版本 AL2，请参阅。亚马逊 Linux 2 额外内容清单

从 PHP 7.x 版本迁移

上游 PHP 社区整理了从 PHP 7.4 迁移至 PHP 8.0 的全面文档。结合前一节中关于迁移到 PHP 8.1 和 PHP 8.2 的文档，您已经掌握了将PHP基于应用程序迁移到现代应用程序所需的所有步骤PHP。

该PHP项目维护一份受支持版本的列表和时间表，以及一份不支持的分支的列表。

> **Note**
>
> AL2023 发布时，PHP社区不支持所有 7.x 和 5.x 版本，PHP也未作为选项包含在 023 中。AL2

Python在 AL2

AL2 在 2026 年 6 月之前提供 Python 2.7 版的支持和安全补丁，这是我们对 AL2 核心包的长期支持承诺的一部分。这种支持不仅限于2020年1月上游Python社群发布的 Python 2.7 EOL声明。

> **Note**
>
> AL2023 已完全移除 Python 2.7。现在，所有需要Python的组件都可以与 Python 3 配合使用。

AL2 使用与 Python 2.7 有硬依赖关系的yum软件包管理器。在 AL2 023 中，dnf软件包管理器已迁移到 Python 3，不再需要 Python 2.7。 AL2023 已完全移至 Python 3。我们建议您完成向 Python 3 的迁移。

Rust in AL2

您可能需要 AL2 使用随 AL2附的Rust工具链来构建自己的代码。

Rust工具链将在的整个生命周期中进行更新。 AL2这可能是对我们提供的工具链中的 CVE 的响应，也可能是另一个软件包中 CVE 更新的先决条件。

Rust是一种变化相对较快的语言，新版本的发布周期约为六周。新版本可能会添加新的语言或标准库功能。尽管 AL2 将在Rust工具链的生命周期中加入新版本，但这不会与上游Rust版本保持一致。因此，如果您想使用Rust语言的尖端功能构建Rust代码，则 AL2 可能不适合使用中提供的Rust工具链。

在的生命周期内 AL2，不会从存储库中删除以前的软件包版本。如果需要以前的Rust工具链，则可以选择放弃新 Rust工具链的错误和安全修复，使用适用于任何 RPM 的相同流程从存储库中安装先前版本。

要在其上构建自己的Rust代码 AL2，请使用中包含的Rust工具链， AL2 并知道该工具链可能会在整个生命周期中向前发展。 AL2

AL2 内核

AL2 最初附带 4.14 内核，当前的默认版本为 5.10。如果您仍在使用 4.14 内核，建议您迁移到 5.10 内核。

AL2 支持内核实时补丁。

主题

支持 AL2 的内核

支持的内核版本

目前，AL2 AMI 适用于内核版本 4.14 和 5.10，默认版本为 5.10。我们建议您使用内核为 5.10 的 AL2 AMI。

AL2023 AMI 提供内核版本 6.1。有关更多信息，请参阅《亚马逊 Linux 2023 用户指南》中的 AL2023 内核与 AL2 相比的变化。AL2023

支持时间范围

在 AL2 AMI 的标准支持结束之前，AL2 上可用的 5.10 内核将一直受支持。

实时修补支持

AL2 内核版本	支持内核实时补丁
4.14	是
5.10	是
5.15	否

AL2 上的内核实时补丁

> **Important**
>
> 亚马逊 Linux 将于 2025-10-31 结束 AL2 内核 4.14 的上线补丁。鼓励客户使用内核 5.10 作为 AL2 的默认内核（参见 AL2 支持的内核），或者使用内核 6.1 和 6.12 迁移到 AL2023。AL2023
> 亚马逊 Linux 将为 AL2 内核 5.10 提供实时补丁，直到 2026-06-30 AL2 的生命周期结束。

AL2 的 Kernel Live Patching 允许您将特定的安全漏洞和严重错误补丁应用于正在运行的 Linux 内核，而无需重启或中断正在运行的应用程序。这使您可以从更高的服务和应用程序可用性中受益，同时应用这些修复程序直到系统可以重新启动。

有关 AL2023 内核实时补丁的信息，请参阅《亚马逊 Linux 2023 用户指南》中的 AL2023 内核实时补丁。AL2023 AL2023

AWS 为 AL2 发布了两种类型的内核实时补丁：

- 安全更新 - 包括 Linux 常见漏洞和风险（CVE）的更新。通常使用 Amazon Linux 安全通告评级将这些更新评为重要 或关键。它们通常对应于通用漏洞评分系统 (CVSS) 的 7 分或更高。在某些情况下， AWS 可能会在分配 CVE 之前提供更新。在这些情况下，补丁可能会显示为错误修复。
- 错误修复 - 包括与 CVE 无关的关键错误和稳定性问题的修复。

AWS 在 AL2 内核版本发布后，为其提供最长 3 个月的内核实时补丁。在 3 个月期限之后，您必须更新到更高版本的内核才能继续接收内核实时修补程序。

AL2 内核实时补丁作为已签名的 RPM 包在现有 AL2 存储库中提供。可以使用现有 yum 工作流程将补丁安装在单个实例上，也可以使用 Sy AWS stems Manager 将其安装在一组托管实例上。

AL2 上提供的内核实时补丁无需支付额外费用。

主题

支持的配置和先决条件

运行 AL2 的 Amazon EC2 实例和本地虚拟机支持内核实时补丁。

要在 AL2 上使用内核实时补丁，必须使用：

- x86_64 架构的内核版本 4.14 或 5.10
- ARM64 架构的内核版本 5.10

策略要求

要从亚马逊 Linux 存储库下载软件包，亚马逊 EC2 需要访问服务拥有的 Amazon S3 存储桶。如果您在环境中使用 Amazon S3 的 Amazon Virtual Private Cloud（VPC）端点，则需要确保您的 VPC 端点策略允许访问这些公有存储桶。

此表介绍了 EC2 访问“内核实时修补”可能需要的每种 Amazon S3 存储桶。

S3 存储桶 ARN	说明
arn: aws: s3:: packages。 *region*.amazonaws.com/*	包含 Amazon Linux AMI 程序包的 Amazon S3 存储桶
arn: aws: s3::: repo。 *region*.amazonaws.com/*	包含 Amazon Linux AMI 存储库的 Amazon S3 存储桶
arn: aws: s3::: 亚马逊 linux。*region*.amazonaws.com/*	包含 AL2 存储库的亚马逊 S3 存储桶
arn: aws: s3::: amazonlinux-2-repos-/* *region*	包含 AL2 存储库的亚马逊 S3 存储桶

以下策略说明了如何限制对属于您组织的身份和资源的访问权，以及如何提供对“内核实时修补”所需的 Amazon S3 存储桶的访问权。将*regionprincipal-org-id*、*resource-org-id*替换为贵组织的价值观。

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/*"
      ]
    }
  ]
}
```

使用内核实时修补

您可以使用实例本身的命令行在单个实例上启用和使用内核实时补丁，也可以使用 Syst AWS ems Manager 在一组托管实例上启用和使用内核实时补丁。

以下部分介绍如何在单独的实例上使用命令行来启用和使用内核实时修补。

有关在一组托管实例上启用和使用内核实时补丁的更多信息，请参阅用户指南中的在 AL2 实例上使用内核实时补丁。AWS Systems Manager

主题

启用内核实时修补

AL2 上默认禁用内核实时补丁。要使用实时修补，必须为内核实时修补安装 yum 插件，并启用实时修补功能。

先决条件

内核实时修补需要 `binutils`。如果尚未安装 `binutils`，请使用以下命令进行安装：

```
$ sudo yum install binutils
```

启用内核实时修补

1. 内核实时补丁适用于以下 AL2 内核版本：

   - x86_64 架构的内核版本 4.14 或 5.10
   - ARM64 架构的内核版本 5.10

   要检查内核版本，请运行以下命令。

   ```
   $ sudo yum list kernel
   ```

2. 如果您已经拥有受支持的内核版本，请跳过此步骤。如果您没有受支持的内核版本，请运行以下命令将内核更新到最新版本并重启实例。

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

```
$ sudo reboot
```

3. 为内核实时修补安装 yum 插件。

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

4. 为内核实时修补启用 yum 插件。

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

此命令还会从配置的存储库安装最新版本的内核实时修补程序 RPM。

5. 要确认用于内核实时修补的 yum 插件已成功安装，请运行以下命令。

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

启用内核实时修补时，将自动应用空的内核实时修补程序 RPM。如果成功启用了内核实时修补，则此命令将返回一个列表，其中包括初始的空内核实时补丁 RPM。下面是示例输出。

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

6. 安装 kpatch 软件包。

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

7. 如果之前安装过 kpatch 服务，请更新它。

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

8. 启动 kpatch 服务。此服务在初始化或启动时会加载所有内核实时补丁。

```
$ sudo systemctl enable kpatch.service && sudo systemctl start kpatch.service
```

9. 在 AL2 Extras 库中启用 “内核实时补丁” 主题。本主题包含内核实时修补程序。

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

查看可用的内核实时补丁

Amazon Linux 安全警报会发布到 Amazon Linux 安全中心。有关 AL2 安全警报（包括内核实时补丁警报）的更多信息，请参阅 Amazon Linux 安全中心。内核实时补丁的前缀为 ALASLIVEPATCH。Amazon Linux 安全中心可能不会列出解决错误的内核实时补丁。

您还可以使用命令行搜索通告和 CVE 的可用内核实时补丁。

列出所有可用的内核实时补丁以获取通告

使用以下命令。

```
$ yum updateinfo list
```

下面显示了示例输出。

```
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 的所有可用内核实时修补程序

使用以下命令。

```
$ yum updateinfo list cves
```

下面显示了示例输出。

```
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
```

应用内核实时补丁

您可以使用 yum 程序包管理器应用内核实时修补程序，方式与应用定期更新相同。用于 Kernel Live Patching 的 yum 插件管理可供应用的内核实时补丁。

> **Tip**
>
> 我们建议您定期使用内核实时修补更新内核，以确保在系统可以重启之前接收特定的重要和关键安全修复。还请检查是否有其他修复已可用于本机内核程序包，这些修复无法作为实时补丁部署，并在这些情况下更新并重启到内核更新。

您可以选择应用特定的内核实时补丁，或者应用任何可用的内核实时补丁以及定期安全更新。

应用特定内核实时补丁

1. 使用 查看可用的内核实时补丁 中描述的命令之一获取内核实时补丁版本。
2. 为您的 AL2 内核应用内核实时补丁。

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

   例如，以下命令为 AL2 内核版本5.10.102-99.473应用内核实时补丁。

   ```
   $ sudo yum install kernel-livepatch-5.10.102-99.473-1.0-4.amzn2.x86_64
   ```

应用任何可用的内核实时补丁以及定期安全更新

使用以下命令。

```
$ sudo yum update --security
```

省略 `--security` 选项将包含错误修复。

> **Important**
>
> - 应用内核实时修补程序后，内核版本不会更新。仅当实例重启后，版本才会更新到新版本。
> - AL2 内核会收到为期三个月的内核实时补丁。三个月的时间段过去后，不会为该内核版本发布新的内核实时修补程序。要在三个月后继续接收内核实时修补程序，您必须重启实例以移动到新的内核版本，然后该版本将在接下来的三个月内继续接收内核实时修补程序。要检查内核版本的支持窗口，请运行 `yum kernel-livepatch supported`。

查看应用的内核实时修补程序

查看应用的内核实时补丁

使用以下命令。

```
$ kpatch list
```

该命令返回已加载和安装的安全更新内核实时补丁的列表。下面是示例输出。

```
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**
>
> 单个内核实时补丁可以包含和安装多个实时补丁。

禁用内核实时修补

如果您不再需要使用内核实时修补，可以随时禁用它。

禁用内核实时修补功能

1. 删除应用的内核实时修补程序的 RPM 软件包。

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

2. 卸载内核实时修补功能的 yum 插件。

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

3. 重启实例。

```
$ sudo reboot
```

限制

内核实时修补具有以下限制：

- 在应用内核实时补丁时，您无法执行休眠、使用高级调试工具（例如 SystemTap kprobes 和 e BPF-based 工具），也无法访问内核实时补丁基础架构使用的 ftrace 输出文件。
-

> **Note**
>
> 由于技术限制，有些问题无法通过实时修补解决。因此，这些修复将不会在内核实时修补程序包中提供，而仅在本机内核程序包更新中提供。您可以像往常一样安装本机内核软件包更新并重新启动系统以激活补丁。

常见问题

有关适用于 AL2 的内核实时补丁的常见问题，请参阅 A mazon Linux 2 内核实时补丁常见问题解答。

AL2 额外内容库

> **Warning**
>
> E epel xtra 启用第三方EPEL7存储库。自 2024 年 6 月 30 日起，不再维护第三方 EPEL7 存储库。
> 此第三方存储库未来将不再更新。这意味着 EPEL 存储库中的程序包将不会有安全修复。
> 有关某些EPEL软件包的选项，请参阅 Amazon Linux 2023 用户指南的EPEL部分。

借 AL2助，您可以使用 Extras 库在您的实例上安装应用程序和软件更新。这些软件更新称为主题。您可以安装主题的某特定版本或忽略要使用最新版本的版本信息。额外功能有助于减轻在操作系统的稳定性和可用软件的新鲜度之间进行折衷的麻烦。

Extras 主题的内容不受关于长期支持和二进制兼容性的 Amazon Linux 政策的约束。额外主题提供对精选软件包列表的访问权限。软件包的版本可能会经常更新，或者可能在与之相同的时间内不受支持 AL2。

> **Note**
>
> 个别额外内容主题可能会在到期 OOL 之前 AL2 被弃用。

要列出可用主题，请使用以下命令。

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

要启用主题并安装其软件包的最新版本以确保新鲜度，请使用以下命令。

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

要启用主题并安装其软件包的特定版本以确保稳定性，请使用以下命令。

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

要从主题中删除安装的软件包，请使用以下命令。

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

> **Note**
>
> 此命令不会删除作为 Extra 依赖项安装的软件包。

要禁用某个主题并使 yum 包管理器无法访问这些软件包，请使用以下命令。

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

> **Important**
>
> 该命令适用于高级用户。如果未正确使用该命令，可能会导致程序包兼容性冲突。

亚马逊 Linux 2 额外内容清单

额外名字	已弃用日期
BCC	
GraphicsMagick1.3	
R3.4	
R4	
ansible2	2023-09-30
aws-nitro-enclaves-cli	
awscli1	
collectd	
collectd-python3	

额外名字	已弃用日期
corretto8	
dnsmasq	
dnsmasq2.85	2025-05-01
Docker	
ecs	
emacs	2018-11-14
epel	2024-06-30
鞭炮	2022-11-08
火狐浏览器	
gimp	2018-11-14
golang1.11	2023-08-01
golang1.19	2023-09-30
golang1.9	2018-12-14
haproxy2	
httpd_modul	
java-openjdk11	2024-09-30
kernel-5.10	
kernel-5.15	
kernel-5.4	
kernel-ng	2022-08-08

额外名字	已弃用日期
lamp-mariadb10.2-php7.2	2020-11-30
libreof	
直播补丁	
光泽	
lustre2.10	
lynis	
mariadb10.5	2025-06-24
mate-desktop1.x	
memcached1.5	
mock	
mock2	
mono	
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

额外名字	已弃用日期
php8.0	2023-11-26
php8.1	2025-12-31
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
测试	

额外名字	已弃用日期
tomcat8.5	2024-03-31
tomcat9	
unbound1.13	2025-05-01
unbound1.17	
vim	2018-11-14

AL2 保留的用户和群组

AL2 在提供映像和安装某些软件包期间，预先分配特定的用户和组。为了防止发生冲突，此处列出了用户、组 UIDs 及其关联 GIDs 的用户、群组和群组。

主题

亚马逊 Linux 2 预留用户名单

按 UID 列出

用户名	UID
root	0
bin	1
daemon	2
adm	3
lp	4
同步	5
shutdown	6
停止	7
邮件	8
uucp	10
operator	11
游戏	12

用户名	UID
ftp	14
o个人资料	16
pkiuser	17
squid	23
named	25
postgres	26
mysql	27
nscd	28
nscd	28
rpcuser	29
rpc	32
amandaBackup	33
ntp	38
邮递员	41
gdm	42
mailnull	47
Apache	48
smmsp	51
Tomcat	53
ldap	55

用户名	UID
tss	59
nslcd	65
pegasus	66
avahi	70
tcpdump	72
sshd	74
radvd	75
赛勒斯	76
arpwatch	77
fax	78
dbus	81
postfix	89
quagga	92
radiusd	95
radiusd	95
hsqldb	96
dovecot	97
身份	98
nobody	99
qemu	107

用户名	UID
usbmuxd	113
stap-server	155
avahi-autoipd	170
pulse	171
rtkit	172
dhcpd	177
sanlock	179
haproxy	188
hacluster	189
systemd-journal-gateway	191
systemd-network	192
systemd-resolve	193
uuidd	357
唐	358
stapdev	359
stapsys	360
stapusr	361
systemd-journal-upload	362
systemd-journal-remote	363
精神错乱的	364

用户名	UID
pesign	365
pcpqa	366
pcp	367
memcached	368
ipsilon	369
ipaapi	370
kdcproxy	371
ods	372
sssd	373
gluster	374
fedfs	375
dovenull	376
coroqnetd	377
clevis	378
clamscan	379
clamilt	380
clamupdate	381
colord	382
geoclue	383
aws-kinesis-agent-user	384

用户名	UID
cwagent	385
unbound	386
polkitd	387
saslauth	388
dirsrv	389
chrony	996
ec2-instance-connect	997
rngd	998
libstoragemgmt	999
ec2-user	1000
nfsnobody	65534

按名称列出

用户名	UID
adm	3
amandaBackup	33
Apache	48
arpwatch	77
avahi	70
avahi-autoipd	170

用户名	UID
aws-kinesis-agent-user	384
bin	1
chrony	996
clamilt	380
clamscan	379
clamupdate	381
clevis	378
colord	382
coroqnetd	377
cwagent	385
赛勒斯	76
daemon	2
dbus	81
dhcpd	177
dirsrv	389
dovecot	97
dovenull	376
ec2-instance-connect	997
ec2-user	1000
fax	78

用户名	UID
fedfs	375
ftp	14
游戏	12
gdm	42
geoclue	383
gluster	374
hacluster	189
停止	7
haproxy	188
hsqldb	96
身份	98
ipaapi	370
ipsilon	369
kdcproxy	371
ldap	55
libstoragemgmt	999
lp	4
邮件	8
邮递员	41
mailnull	47

用户名	UID
memcached	368
mysql	27
named	25
nfsnobody	65534
nobody	99
nscd	28
nscd	28
nslcd	65
ntp	38
ods	372
operator	11
o个人资料	16
pcp	367
pcpqa	366
pegasus	66
pesign	365
pkiuser	17
polkitd	387
postfix	89
postgres	26

用户名	UID
pulse	171
qemu	107
quagga	92
radiusd	95
radiusd	95
radvd	75
rngd	998
root	0
rpc	32
rpcuser	29
rtkit	172
精神错乱的	364
sanlock	179
saslauth	388
shutdown	6
smmsp	51
squid	23
sshd	74
sssd	373
stap-server	155

用户名	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
唐	358
tcpdump	72
Tomcat	53
tss	59
unbound	386
usbmuxd	113
uucp	10
uuidd	357

亚马逊 Linux 2 预留群组清单

由 GID 列出

组名	GID
root	0
bin	1
daemon	2
sys	3
adm	4
tty	5
disk	6
disk	6
lp	7
mem	8
kmem	9
wheel	10
cdrom	11
邮件	12
uucp	14
man	15
o个人资料	16
pkiuser	17
dialout	18
floppy	19

组名	GID
游戏	20
slocate	21
utmp	22
squid	23
named	25
postgres	26
mysql	27
nscd	28
nscd	28
rpcuser	29
rpc	32
磁带	33
磁带	33
utempter	35
kvm	36
ntp	38
视频	39
浸	40
邮递员	41
gdm	42

组名	GID
mailnull	47
Apache	48
ftp	50
smmsp	51
Tomcat	53
锁定	54
ldap	55
tss	59
audio	63
pegasus	65
avahi	70
tcpdump	72
sshd	74
radvd	75
saslauth	76
saslauth	76
arpwatch	77
fax	78
dbus	81
screen	84

组名	GID
quaggavt	85
wbpriv	88
wbpriv	88
postfix	89
postdrop	90
quagga	92
radiusd	95
radiusd	95
hsqldb	96
dovecot	97
身份	98
nobody	99
用户	100
qemu	107
usbmuxd	113
stap-server	155
stapusr	156
stapusr	156
stapsys	157
stapdev	158

组名	GID
avahi-autoipd	170
pulse	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
唐	354
systemd-journal-upload	355
sfcb	356
systemd-journal-remote	356
精神错乱的	357

组名	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
gluster	370
fedfs	371
dovenull	372
Docker	373
coroqnetd	374
clevis	375
clamscan	376
clamilt	377

组名	GID
virusgroup	378
virusgroup	378
virusgroup	378
clamupdate	379
colord	380
geoclue	381
printadmin	382
aws-kinesis-agent-user	383
cwagent	384
pulse-rt	385
pulse-access	386
unbound	387
polkitd	388
dirsrv	389
cgred	993
chrony	994
ec2-instance-connect	995
rngd	996
libstoragemgmt	997
ssh_keys	998

组名	GID
input	999
ec2-user	1000
nfsnobody	65534

按名称列出

组名	GID
adm	4
Apache	48
arpwatch	77
audio	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

组名	GID
clevis	375
colord	380
coroqnetd	374
cwagent	384
daemon	2
dbus	81
dhcpd	177
dialout	18
浸	40
dirsrv	389
disk	6
disk	6
Docker	373
dovecot	97
dovenull	372
ec2-instance-connect	995
ec2-user	1000
fax	78
fedfs	371
floppy	19

组名	GID
ftp	50
游戏	20
gdm	42
geoclue	381
gluster	370
haclient	189
haproxy	188
hsqldb	96
身份	98
input	999
ipaapi	365
ipsilon	363
kdcproxy	366
kmem	9
kvm	36
ldap	55
libstoragemgmt	997
libvirt	369
锁定	54
lp	7

组名	GID
邮件	12
邮递员	41
mailnull	47
man	15
mem	8
memcached	361
mysql	27
named	25
nfsnobody	65534
nobody	99
nscd	28
nscd	28
ntp	38
ods	367
o个人资料	16
pcp	360
pcpqa	359
pegasus	65
pesign	358
pkcs11	364

组名	GID
pkiuser	17
polkitd	388
postdrop	90
postfix	89
postgres	26
printadmin	382
pulse	171
pulse-access	386
pulse-rt	385
qemu	107
quagga	92
quaggavt	85
radiusd	95
radiusd	95
radvd	75
rngd	996
root	0
rpc	32
rpcuser	29
rtkit	172

组名	GID
精神错乱的	357
sanlock	179
saslauth	76
saslauth	76
screen	84
sfcb	356
slocate	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

组名	GID
systemd-journal-gateway	191
systemd-journal-remote	356
systemd-journal-upload	355
systemd-network	192
systemd-resolve	193
唐	354
磁带	33
磁带	33
tcpdump	72
Tomcat	53
tss	59
tty	5
unbound	387
usbmon	351
usbmuxd	113
用户	100
utempter	35
utmp	22
uucp	14
uuidd	353

组名	GID
视频	39
virtlogin	362
virusgroup	378
virusgroup	378
virusgroup	378
wbpriv	88
wbpriv	88
wheel	10
wireshark	352

AL2 源软件包

您可以使用 Amazon Linux 中提供的工具，查看您已在实例上安装的软件包的源，获取参考信息。您可以查看 Amazon Linux 和在线软件包存储库中包含的全部软件包的源软件包。确定要安装的源软件包的软件包名称，然后使用yumdownloader --source命令查看正在运行的实例中的源代码。例如：

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

可以解压源 RPM，为了便于参考，您可以使用标准 RPM 工具查看源代码树。完成调试之后，该软件包可供使用。

中的安全性与合规性 AL2

云安全AWS是重中之重。作为AWS客户，您可以受益于专为满足大多数安全敏感型组织的要求而构建的数据中心和网络架构。

安全是双方共同承担AWS的责任。责任共担模式将此描述为云的安全性和云中的安全性：

- 云安全 — AWS 负责保护在AWS云中运行AWS服务的基础架构。 AWS还为您提供可以安全使用的服务。作为AWS合规计划合规计划合规计划合的一部分，第三方审计师定期测试和验证我们安全的有效性。要了解适用于 AL2 023 的合规计划，请参阅按合规计划划分的范围内的AWS服务按合规计划。
- 云中的安全性：您的责任由您使用的AWS服务决定。您还需要对其它因素负责，包括您的数据的敏感性、您公司的要求以及适用的法律法规。

启用 FIPS 模式 AL2

本节介绍如何在上 AL2启用《联邦信息处理标准》(FIPS)。有关 FIPS 的更多信息，请参阅：

- 美国联邦信息处理标准 (FIPS)
- 合规性 FAQs：联邦信息处理标准

先决条件

- 可以访问互联网下载所需软件包的现有 AL2 Amazon EC2 实例。有关启动 AL2 Amazon EC2 实例的更多信息，请参阅亚马逊 EC2 上的 AL2。
- 您必须使用 SSH 或连接到您的 Amazon EC2 实例AWS Systems Manager。

> **Important**
>
> ED25519 FIPS 模式下不支持 SSH 用户密钥。如果您使用 ED25519 SSH 密钥对启动 Amazon EC2 实例，则必须使用其他算法（例如 RSA）生成新密钥，否则在启用 FIPS 模式后您可能会失去对实例的访问权限。有关更多信息，请参阅 Amazon EC2 用户指南中的创建密钥对。

启用 FIPS 模式

1. 使用 SSH 或连接到您的 AL2 实例AWS Systems Manager。
2. 确保系统是最新版本。有关更多信息，请参阅 程序包存储库。
3. 运行以下命令安装并启用该`dracut-fips`模块。

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

4. 使用以下命令在 Linux 内核命令行上启用 FIPS 模式。这将为常见问题解答中列出的模块在系统范围内启用 FIPS 模式 AL2

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

5. 重启您的 AL2 实例。

```
sudo reboot
```

6. 要验证是否已启用 FIPS 模式，请重新连接到实例并运行以下命令。

```
sysctl crypto.fips_enabled
```

您应看到以下输出：

```
crypto.fips_enabled = 1
```

您还可以通过运行以下命令来验证 OpenSSH 是否处于 FIPS 模式：

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

您应看到以下输出：

```
FIPS mode initialized
```

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。