aws

User Guide

Amazon Linux 2

Amazon Linux 2: User Guide

Table of Contents

What is Amazon Linux 2? 1
Amazon Linux availability 1
Deprecated functionality 2
compat- packages 2
Deprecated functionality discontinued in AL1, removed in AL2 2
32-bit x86 (i686) AMIs 3
aws-apitools-* replaced by AWS CLI 3
systemd replaces upstart in AL2 4
Functionality deprecated in AL2 and removed in AL2023 4
32-bit x86 (i686) Packages 4
aws-apitools-* replaced by AWS CLI 5
bzr revision control system 5
cgroup v1 6
log4j hotpatch (log4j-cve-2021-44228-hotpatch) 6
lsb_release and the system-lsb-core package 6
mcrypt 7
OpenJDK 7 (java-1.7.0-openjdk) 7
Python 2.7 7
rsyslog-openssl replaces rsyslog-gnutls 8
Preparing your migration to AL2023 9
Review the list of changes in AL2023 9
Migrate to systemd timers from cron jobs 9
Comparing AL1 and AL2 10
AL1 support and EOL 10
Support for AWS Graviton processors 10
systemd replaces upstart as init system 10
Python 2.6 and 2.7 were replaced with Python 3 10
AL1 and AL2 AMI comparison 11
AL1 and AL2 container comparison 40
Using AL2 on AWS 48
AL2 on Amazon EC2 48
Launch Amazon EC2 instance with AL2 AMI 48
Connecting to an Amazon EC2 instance 49
AL2 AMI boot mode 49

Package repository ... 49
Using cloud-init on AL2 ... 52
Enable FIPS Mode on AL2 ... 55
AL2 release notifications ... 56
AL2 outside Amazon EC2 ... 60
Run AL2 on premises ... 60
Step 1: Prepare the `seed.iso` boot image ... 60
Step 2: Download the AL2 VM image ... 63
Step 3: Boot and connect to your new VM ... 63
Identifying AL2 ... 66
Identify Amazon Linux images ... 66
AL2 ... 66
Amazon Linux AMI ... 67
AWS integration in AL2 ... 68
AWS command line tools ... 68
Programming languages and runtimes ... 69
C/C++ and Fortran ... 69
Go in AL2 ... 70
Java ... 70
Perl ... 71
Perl modules ... 71
PHP ... 71
Migrating from earlier PHP 8.x versions ... 71
Migrating from PHP 7.x versions ... 72
Python in AL2 ... 72
Rust in AL2 ... 72
AL2 kernel ... 74
AL2 supported kernels ... 74
Kernel Live Patching ... 75
Supported configurations and prerequisites ... 76
Work with Kernel Live Patching ... 78
Limitations ... 83
Frequently asked questions ... 84
AL2 Extras ... 85
List of Amazon Linux 2 Extras ... 86
AL2 Source Packages ... 91

What is Amazon Linux 2?

Amazon Linux 2 (AL2) is a Linux operating system from Amazon Web Services (AWS). AL2 is designed to provide a stable, secure, and high-performing environment for applications running on Amazon EC2. It also includes packages that enable efficient integration with AWS, including launch configuration tools and many popular AWS libraries and tools. AWS provides ongoing security and maintenance updates for all instances running AL2. Many applications developed on CentOS, and similar distributions, run on AL2. AL2 is provided at no additional charge.

> **Note**
>
> AL2 is no longer the current version of Amazon Linux. AL2023 is the successor to AL2. For more information, see Comparing AL2 and AL2023 and the list of Package changes in AL2023 in the AL2023 User Guide.

Amazon Linux availability

AWS provides AL2023, AL2, and Amazon Linux 1 (AL1, formerly Amazon Linux AMI). If you are migrating from another Linux distribution to Amazon Linux, we recommend that you migrate to AL2023.

> **Note**
>
> Standard support for AL1 ended on December 31, 2020. The AL1 maintenance support phase ended December 31, 2023. For more information about AL1 EOL and maintenance support, see the blog post Update on Amazon Linux AMI end-of-life.

For more information about Amazon Linux, see AL2023, AL2, and AL1.

For Amazon Linux container images, see Amazon Linux container image in the *Amazon Elastic Container Registry User Guide*.

Deprecated functionality in AL2

The following sections describe functionality supported in AL2 and not present in AL2023. This is functionality such as features and packages that are present in AL2, but not in AL2023 and will not be added to AL2023. See the AL2 documentation for how long this functionality is supported in AL2.

`compat-` packages

Any packages in AL2 with the prefix of `compat-` are provided for binary compatibility with older binaries that have not yet been rebuilt for modern versions of the package. Each new major version of Amazon Linux will not carry forward any `compat-` package from prior releases.

All `compat-` packages in a release of Amazon Linux (such as AL2) are discontinued, and not present in the subsequent version (such as AL2023). We strongly recommend that software is rebuilt against updated versions of the libraries.

Deprecated functionality discontinued in AL1, removed in AL2

This section describes functionality that is available in AL1, and is no longer available in AL2.

> **Note**
>
> As part of the maintenance support phase of AL1, some packages had an end-of-life (EOL) date earlier than the EOL of AL1. For more information, see AL1 Package support statements.

> **Note**
>
> Some AL1 functionality was discontinued in earlier releases. For information, see the AL1 Release Notes.

Topics

- 32-bit x86 (i686) AMIs

- aws-apitools-* replaced by AWS CLI
- systemd replaces upstart in AL2

32-bit x86 (i686) AMIs

As part of the 2014.09 release of AL1, Amazon Linux announced that it would be the last release to produce 32-bit AMIs. Therefore, starting from the 2015.03 release of AL1, Amazon Linux no longer supports running the system in 32-bit mode. AL2 offers limited runtime support for 32-bit binaries on x86-64 hosts and does not provide development packages to enable the building of new 32-bit binaries. AL2023 no longer includes any 32-bit user space packages. We recommend that users complete their transition to 64-bit code before migrating to AL2023.

If you need to run 32-bit binaries on AL2023, it is possible to use the 32-bit userspace from AL2 inside an AL2 container running on top of AL2023.

`aws-apitools-*` replaced by AWS CLI

Before the release of the AWS CLI in September 2013, AWS made a set of command line utilities available, implemented in Java, which allowed users to make Amazon EC2 API calls. These tools were discontinued in 2015, with the AWS CLI becoming the preferred way to interact with Amazon EC2 APIs from the command line. The set of command line utilities includes the following `aws-apitools-*` packages.

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

Upstream support for the `aws-apitools-*` packages ended in March of 2017. Despite the lack of upstream support, Amazon Linux continued to ship some of these command line utilities, such as `aws-apitools-ec2`, to provide backward compatibility for users. The AWS CLI is a more robust and complete tool than the `aws-apitools-*` packages as it is actively maintained and provides a means of using all AWS APIs.

The `aws-apitools-*` packages were deprecated in March 2017 and will not be receiving further updates. All users of any of these packages should migrate to the AWS CLI as soon as possible. These packages are not present in AL2023.

AL1 also provided the `aws-apitools-iam` and `aws-apitools-rds` packages, which were deprecated in AL1, and are not present in Amazon Linux from AL2 onward.

`systemd` replaces `upstart` in AL2

AL2 was the first Amazon Linux release to use the `systemd` init system, replacing `upstart` in AL1. Any `upstart` specific configuration must be changed as part of the migration from AL1 to a newer version of Amazon Linux. It is not possible to use `systemd` on AL1, so moving from `upstart` to `systemd` can only be done as part of moving to a more recent major version of Amazon Linux such as AL2 or AL2023.

Functionality deprecated in AL2 and removed in AL2023

This section describes functionality that is available in AL2, and no longer available in AL2023.

Topics

- 32-bit x86 (i686) Packages
- aws-apitools-* replaced by AWS CLI
- bzr revision control system
- cgroup v1
- log4j hotpatch (log4j-cve-2021-44228-hotpatch)
- lsb_release and the system-lsb-core package
- mcrypt
- OpenJDK 7 (java-1.7.0-openjdk)
- Python 2.7
- rsyslog-openssl replaces rsyslog-gnutls

32-bit x86 (i686) Packages

As part of the 2014.09 release of AL1, we announced that it would be the last release to produce 32-bit AMIs. Therefore, starting from the 2015.03 release of AL1, Amazon Linux no longer supports

running the system in 32-bit mode. AL2 provides limited runtime support for 32-bit binaries on x86-64 hosts and doesn't provide development packages to enable the building of new 32-bit binaries. AL2023 no longer includes any 32-bit userspace packages. We recommend customers complete their transition to 64-bit code.

If you need to run 32-bit binaries on AL2023, it is possible to use the 32-bit userspace from AL2 inside an AL2 container running on top of AL2023.

`aws-apitools-*` replaced by AWS CLI

Prior to release of the AWS CLI in September 2013, AWS made a set of command line utilities available, implemented in Java, which allowed customers to make Amazon EC2 API calls. These tools were deprecated in 2015, with the AWS CLI becoming the preferred way to interact with Amazon EC2 APIs from the command line. This includes the following `aws-apitools-*` packages.

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

Upstream support for the `aws-apitools-*` packages ended in March of 2017. Despite the lack of upstream support, Amazon Linux continued to ship some of these command line utilities (such as `aws-apitools-ec2`) in order to provide backwards compatibility for customers. The AWS CLI is a more robust and complete tool than the `aws-apitools-*` packages as it is actively maintained and provides a means of using all AWS APIs.

The `aws-apitools-*` packages were deprecated in March 2017 and will not be receiving further updates. All users of any of these packages should migrate to the AWS CLI as soon as possible. These packages are not present in AL2023.

`bzr` revision control system

The GNU Bazaar (`bzr`) revision control system is discontinued in AL2 and no longer present in AL2023.

Users of `bzr` are advised to migrate their repositories to `git`.

cgroup v1

AL2023 moves to Unified Control Group hierarchy (cgroup v2), whereas AL2 uses cgroup v1. As AL2 doesn't support cgroup v2, this migration needs to be completed as part of moving to AL2023.

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

> **Note**
>
> The `log4j-cve-2021-44228-hotpatch` package is deprecated in AL2 and removed in AL2023.

In response to CVE-2021-44228, Amazon Linux released an RPM packaged version of the Hotpatch for Apache Log4j for AL1 and AL2. In the announcement of the addition of the hotpatch to Amazon Linux , we noted that "Installing the hotpatch is not a replacement for updating to a log4j version that mitigates CVE-2021-44228 or CVE-2021-45046.".

The hotpatch was a mitigation to allow time to patch `log4j`. The first general availability release of AL2023 was 15 months after CVE-2021-44228, so AL2023 doesn't ship with the hotpatch (enabled or not).

Customers running their own `log4j` versions on Amazon Linux are advised to ensure they have updated to versions not affected by CVE-2021-44228 or CVE-2021-45046.

`lsb_release` and the `system-lsb-core` package

Historically, some software invoked the `lsb_release` command (provided in AL2 by the `system-lsb-core` package) to get information about the Linux distribution that it was being run on. The Linux Standards Base (LSB) introduced this command and Linux distributions adopted it. Linux distributions have evolved to use the simpler standard of holding this information in `/etc/os-release` and other related files.

The `os-release` standard comes out of `systemd`. For more information, see systemd os-release documentation.

AL2023 doesn't ship with the `lsb_release` command, and doesn't include the `system-lsb-core` package. Software should complete the transition to the `os-release` standard to maintain compatibility with Amazon Linux and other major Linux distributions.

`mcrypt`

The `mcrypt` library and associated PHP extension was deprecated in AL2, and is no longer present in AL2023.

Upstream PHP deprecated the `mcrypt` extension in PHP 7.1 which was first released in December 2016 and had its final release in October 2019.

The upstream `mcrypt` library last made a release in 2007, and has not made the migration from `cvs` revision control that SourceForge required for new commits in 2017, with the most recent commit (and only for 3 years prior) being from 2011 removing the mention of the project having a maintainer.

Any remaining users of `mcrypt` are advised to port their code to OpenSSL, as `mcrypt` will not be added to AL2023.

OpenJDK 7 (`java-1.7.0-openjdk`)

> **Note**
>
> AL2023 provides several versions of Amazon Corretto to support Java based workloads. The OpenJDK 7 packages are deprecated in AL2, and no longer present in AL2023. The oldest JDK available in AL2023 is provided by Corretto 8.

For more information about Java on Amazon Linux, see Java in AL2.

Python 2.7

> **Note**
>
> AL2023 removed Python 2.7, so any OS components requiring Python are written to work with Python 3. To continue to use a version of Python provided by and supported by Amazon Linux, convert Python 2 code to Python 3.

For more information about Python on Amazon Linux, see Python in AL2.

`rsyslog-openssl` replaces `rsyslog-gnutls`

The `rsyslog-gnutls` package is deprecated in AL2, and no longer present in AL2023. The `rsyslog-openssl` package should be a drop-in replacement for any usage of the `rsyslog-gnutls` package.

Preparing your migration to AL2023

You can prepare your move to AL2023 while you continue to use AL2.

Topics

- Review the list of changes in AL2023
- Migrate to systemd timers from cron jobs

Review the list of changes in AL2023

The AL2023 documentation contains a detailed list of changes that have been implemented since AL2. This information is located in the Comparing AL2 and AL2023 section. There is also a comprehensive list of software package changes located in the Package changes in AL2023 section.

AL2023 doesn't include `amazon-linux-extras`. Instead, it provides namespaced packages where multiple versions are provided. Because many packages are updated in AL2023, the base versions in AL2023 might be later than the versions that you are getting from `amazon-linux-extras`.

> **Note**
>
> We recommend that you don't run `amazon-linux-extras`, because it is EOL.

After you review these sections in the documentation, you can determine if there are changes in AL2023 that might require you to adapt your environment for the migration. For example, you might need to finally migrate a Python 2.7 script to Python 3.

Migrate to `systemd` timers from `cron` jobs

By default, `cron` is not installed in AL2023. You can migrate your `cron` jobs to `systemd` timers in AL2 in preparation for migrating to AL2023. `systemd` has many advantages, such as more precise control over when timers are run and improved logging.

Comparing AL1 and AL2

The following topics describe key differences between AL1 and AL2. They also contain information about lifespan and support, and package changes.

Topics

- AL1 support and EOL
- Support for AWS Graviton processors
- systemd replaces upstart as init system
- Python 2.6 and 2.7 were replaced with Python 3
- Comparing packages installed on AL1 and AL2 AMIs
- Comparing packages installed on AL1 and AL2 base container images

AL1 support and EOL

AL1 is now EOL. AL1 ended standard support as of December 31, 2020, and was in a maintenance support phase until December 31, 2023.

AL2 is supported until June 30, 2025. We recommend migrating to AL2023, as it is supported until 2028.

Support for AWS Graviton processors

AL2 introduced support for Graviton processors. AL2023 is further optimized for Graviton processors.

`systemd` replaces `upstart` as `init` system

In AL2, `systemd` replaced `upstart` as the `init` system.

Python 2.6 and 2.7 were replaced with Python 3

Although AL1 marked Python 2.6 as EOL with the 2018.03 release, the packages were still in the repositories to install. AL2 shipped with Python 2.7 as the earliest supported Python version.

AL2023 completes the transition to Python 3, and no Python 2.x versions are included in the repositories.

Comparing packages installed on AL1 and AL2 AMIs

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

Package	AL1 AMI	AL2 AMI
awscli		1.18.147
basesystem	10.0	10.0
bash	4.2.46	4.2.46
bash-completion		2.1
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	

Package	AL1 AMI	AL2 AMI
chkconfig	1.3.49.3	1.7.4
chrony		4.2
cloud-disk-utils	0.27	
cloud-init	0.7.6	19.3
cloud-utils-growpart		0.31
copy-jdk-configs	3.3	
coreutils	8.22	8.22
cpio	2.10	2.12
cracklib	2.8.16	2.9.0
cracklib-dicts	2.8.16	2.9.0
cronie	1.4.4	1.4.11
cronie-anacron	1.4.4	1.4.11
crontabs	1.10	1.11
cryptsetup	1.6.7	1.7.4
cryptsetup-libs	1.6.7	1.7.4
curl	7.61.1	8.3.0
cyrus-sasl	2.1.23	
cyrus-sasl-lib	2.1.23	2.1.26
cyrus-sasl-plain	2.1.23	2.1.26
dash	0.5.5.1	

Package	AL1 AMI	AL2 AMI
db4	4.7.25	
db4-utils	4.7.25	
dbus	1.6.12	1.10.24
dbus-libs	1.6.12	1.10.24
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

Package	AL1 AMI	AL2 AMI
dracut	004	033
dracut-config-ec2		2.0
dracut-config-generic		033
dracut-modules-growroot	0.20	
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

Package	AL1 AMI	AL2 AMI
file	5.37	5.11
file-libs	5.37	5.11
filesystem	2.4.30	3.2
findutils	4.4.2	4.5.11
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

Package	AL1 AMI	AL2 AMI
glibc-all-langpacks		2.26
glibc-common	2.17	2.26
glibc-locale-source		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
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

Package	AL1 AMI	AL2 AMI
gssproxy		0.7.0
gzip	1.5	1.5
hardlink		1.3
hesiod	3.1.0	
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

Package	AL1 AMI	AL2 AMI
java-1.7.0-openjdk	1.7.0.321	
javapackages-tools	0.9.1	
jbigkit-libs		2.0
jpackage-utils	1.7.5	
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

Package	AL1 AMI	AL2 AMI
lcms2	2.6	
less	436	458
libICE	1.0.6	
libSM	1.2.1	
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

Package	AL1 AMI	AL2 AMI
libcap-ng	0.7.5	0.7.5
libcap54	2.54	
libcgroup	0.40.rc1	
libcollection		0.7.0
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	2.11	3.0
libestr		0.1.9
libevent	2.0.21	2.0.21
libfastjson		0.99.4
libfdisk		2.30.2
libffi	3.0.13	3.0.13

Package	AL1 AMI	AL2 AMI
libfontenc	1.0.5	
libgcc		7.3.1
libgcc72	7.2.1	
libgcrypt	1.5.3	1.5.3
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	

Package	AL1 AMI	AL2 AMI
libnl	1.1.4	
libnl3		3.2.28
libnl3-cli		3.2.28
libpath_utils		0.2.1
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

Package	AL1 AMI	AL2 AMI
libsss_nss_idmap		1.16.5
libstdc++		7.3.1
libstdc++72	7.2.1	
libstoragemgmt		1.6.1
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	

Package	AL1 AMI	AL2 AMI
libxml2	2.9.1	2.9.1
libxml2-python		2.9.1
libxml2-python27	2.9.1	
libxslt	1.1.28	
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

Package	AL1 AMI	AL2 AMI
microcode_ctl	2.1	2.1
mingetty	1.08	
mlocate		0.26
mtr		0.92
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

Package	AL1 AMI	AL2 AMI
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	
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	

Package	AL1 AMI	AL2 AMI
pam_passwdqc	1.0.5	
parted	2.1	3.1
passwd	0.79	0.79
pciutils	3.1.10	3.5.1
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

Package	AL1 AMI	AL2 AMI
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
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

Package	AL1 AMI	AL2 AMI
plymouth		0.8.9
plymouth-core-libs		0.8.9
plymouth-scripts		0.8.9
pm-utils	1.4.1	1.4.1
policycoreutils	2.1.12	2.5
popt	1.13	1.13
postfix		2.10.1
procmail	3.22	
procps	3.2.8	
procps-ng		3.3.10
psacct	6.3.2	6.6.1
psmisc	22.20	22.20
pth	2.0.7	2.0.7
pygpgme		0.3
pyliblzma		0.5.3
pystache		0.5.3
python		2.7.18
python-babel		0.9.6
python-backports		1.0
python-backports-ssl_match_hostname		3.5.0.1

Package	AL1 AMI	AL2 AMI
python-cffi		1.6.0
python-chardet		2.2.1
python-configobj		4.7.2
python-daemon		1.6
python-devel		2.7.18
python-docutils		0.12
python-enum34		1.0.4
python-idna		2.4
python-iniparse		0.4
python-ipaddress		1.0.16
python-jinja2		2.7.2
python-jsonpatch		1.2
python-jsonpointer		1.9
python-jwcrypto		0.4.2
python-kitchen		1.1.1
python-libs		2.7.18
python-lockfile		0.9.1
python-markupsafe		0.11
python-pillow		2.0.0
python-ply		3.4

Package	AL1 AMI	AL2 AMI
python-pycparser		2.14
python-pycurl		7.19.0
python-repoze-lru		0.4
python-requests		2.6.0
python-simplejson		3.2.0
python-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

Package	AL1 AMI	AL2 AMI
python2-six		1.11.0
python27	2.7.18	
python27-PyYAML	3.10	
python27-babel	0.9.4	
python27-backports	1.0	
python27-backports-ssl_match_hostname	3.4.0.2	
python27-boto	2.48.0	
python27-botocore	1.17.31	
python27-chardet	2.0.1	
python27-colorama	0.4.1	
python27-configobj	4.7.2	
python27-crypto	2.6.1	
python27-daemon	1.5.2	
python27-dateutil	2.1	
python27-devel	2.7.18	
python27-docutils	0.11	
python27-ecdsa	0.11	
python27-futures	3.0.3	
python27-imaging	1.1.6	
python27-iniparse	0.3.1	

Package	AL1 AMI	AL2 AMI
python27-jinja2	2.7.2	
python27-jmespath	0.9.2	
python27-jsonpatch	1.2	
python27-jsonpointer	1.0	
python27-kitchen	1.1.1	
python27-libs	2.7.18	
python27-lockfile	0.8	
python27-markupsafe	0.11	
python27-paramiko	1.15.1	
python27-pip	9.0.3	
python27-ply	3.4	
python27-pyasn1	0.1.7	
python27-pycurl	7.19.0	
python27-pygpgme	0.3	
python27-pyliblzma	0.5.3	
python27-pystache	0.5.3	
python27-pyxattr	0.5.0	
python27-requests	1.2.3	
python27-rsa	3.4.1	
python27-setuptools	36.2.7	

Package	AL1 AMI	AL2 AMI
python27-simplejson	3.6.5	
python27-six	1.8.0	
python27-urlgrabber	3.10	
python27-urllib3	1.24.3	
python27-virtualenv	15.1.0	
python3		3.7.16
python3-daemon		2.2.3
python3-docutils		0.14
python3-libs		3.7.16
python3-lockfile		0.11.0
python3-pip		20.2.2
python3-pystache		0.5.4
python3-setuptools		49.1.3
python3-simplejson		3.2.0
pyxattr		0.5.1
qrencode-libs		3.4.1
quota	4.00	4.01
quota-nls	4.00	4.01
rdate		1.4
readline	6.2	6.2

Package	AL1 AMI	AL2 AMI
rmt	0.4	
rng-tools	5	6.8
rootfiles	8.1	8.1
rpcbind	0.2.0	0.2.0
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	

Package	AL1 AMI	AL2 AMI
scl-utils		20130529
screen	4.0.3	4.1.0
sed	4.2.1	4.2.2
selinux-policy		3.13.1
selinux-policy-targeted		3.13.1
sendmail	8.14.4	
setserial	2.17	2.17
setup	2.8.14	2.8.71
setuptool		1.19.11
sgpio	1.2.0.10	1.2.0.10
shadow-utils	4.1.4.2	4.1.5.1
shared-mime-info	1.1	1.8
slang	2.2.1	2.2.4
sqlite	3.7.17	3.7.17
sssd-client		1.16.5
strace		4.26
sudo	1.8.23	1.8.23
sysctl-defaults	1.0	1.0
sysfsutils	2.1.0	
sysstat		10.1.5

Package	AL1 AMI	AL2 AMI
system-release	2018.03	2
systemd		219
systemd-libs		219
systemd-sysv		219
systemtap-runtime		4.5
sysvinit	2.87	
sysvinit-tools		2.88
tar	1.26	1.26
tcp_wrappers	7.6	7.6
tcp_wrappers-libs	7.6	7.6
tcpdump		4.9.2
tcsh		6.18.01
teamd		1.27
time	1.7	1.7
tmpwatch	2.9.16	
traceroute	2.0.14	2.0.22
ttmkfdir	3.0.9	
tzdata	2023c	2023c
tzdata-java	2023c	
udev	173	

Package	AL1 AMI	AL2 AMI
unzip	6.0	6.0
update-motd	1.0.1	1.1.2
upstart	0.6.5	
usermode		1.111
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

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

Comparing packages installed on AL1 and AL2 base container images

Package	AL1 Container	AL2 Container
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

Package	AL1 Container	AL2 Container
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
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

Package	AL1 Container	AL2 Container
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	
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

Package	AL1 Container	AL2 Container
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
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	

Package	AL1 Container	AL2 Container
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
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

Package	AL1 Container	AL2 Container
pkgconfig	0.27.1	
popt	1.13	1.13
pth	2.0.7	2.0.7
pygpgme		0.3
pyliblzma		0.5.3
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	

Package	AL1 Container	AL2 Container
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
rpm-libs	4.11.3	4.11.3
rpm-python27	4.11.3	
sed	4.2.1	4.2.2
setup	2.8.14	2.8.71
shared-mime-info	1.1	1.8
sqlite	3.7.17	3.7.17
sysctl-defaults	1.0	
system-release	2018.03	2
tar	1.26	
tzdata	2023c	2023c
vim-data		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	1.1.4	1.1.4

Package	AL1 Container	AL2 Container
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

Using AL2 on AWS

You can set up AL2 for use with other AWS services. For example, you can choose an Amazon Linux image when you launch an Amazon EC2 instance.

Topics

- AL2 on Amazon EC2

AL2 on Amazon EC2

> **Note**
>
> AL2 is no longer the current version of Amazon Linux. AL2023 is the successor to AL2. For more information, see Comparing AL2 and AL2023 and the list of Package changes in AL2023 in the AL2023 User Guide.

Topics

- Launch Amazon EC2 instance with AL2 AMI
- Connecting to an Amazon EC2 instance
- AL2 AMI boot mode
- Package repository
- Using cloud-init on AL2
- Enable FIPS Mode on AL2
- AL2 release notifications

Launch Amazon EC2 instance with AL2 AMI

You can launch an Amazon EC2 instance with the AL2 AMI. For more information, see Step 1: Launch an instance.

Connecting to an Amazon EC2 instance

There are several ways to connect to your Amazon Linux instance, including SSH, AWS Systems Manager Session Manager, and EC2 Instance Connect. For more information, see Connect to your Linux instance in the *Amazon EC2 User Guide for Linux Instances*.

SSH users and sudo

Amazon Linux does not allow remote `root` secure shell (SSH) by default. Also, password authentication is disabled to prevent brute force attacks. To enable SSH logins to an Amazon Linux instance, you must provide your key pair to the instance at launch. You must also set the security group used to launch your instance to allow SSH access. By default, the only account that can log in remotely using SSH is `ec2-user`. This account also has **sudo** privileges. If you enable remote `root` login, be aware that it is less secure than relying on key pairs and a secondary user.

AL2 AMI boot mode

AL2 AMIs don't have a boot mode parameter set. Instances launched from AL2 AMIs follow the default boot mode value of the instance type. For more information, see Boot modes in the *Amazon EC2 User Guide for Linux Instances*.

Package repository

This information applies to AL2 and AL1. For information about AL2023, see Managing packages and operating system updates in the *AL2023 User Guide*.

AL2 and AL1 are designed to be used with online package repositories hosted in each Amazon EC2 AWS Region. The repositories are available in all Regions and are accessed using **yum** update tools. Hosting repositories in each Region enables us to deploy updates quickly and without any data transfer charges.

> **Important**
>
> The last version of AL1 reached EOL on December 31, 2023 and will not receive any security updates or bug fixes starting January 1, 2024. For more information, see Amazon Linux AMI end-of-life.

If you don't need to preserve data or customizations for your instances, you can launch new instances using the current AL2 AMI. If you do need to preserve data or customizations for your

instances, you can maintain those instances through the Amazon Linux package repositories. These repositories contain all the updated packages. You can choose to apply these updates to your running instances. Earlier versions of the AMI and update packages continue to be available for use, even as new versions are released.

> **Note**
>
> To update and install packages without internet access on an Amazon EC2 instance, see How can I update yum or install packages without internet access on my Amazon EC2 instances running AL1, AL2, or AL2023?

To install packages, use the following command:

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

If you find that Amazon Linux doesn't contain an application that you need, you can install the application directly on your Amazon Linux instance. Amazon Linux uses RPMs and yum for package management, and that is likely the most direct way to install new applications. You should check to see if an application is available in our central Amazon Linux repository first, because many applications are available there. From there, you can add these applications to your Amazon Linux instance.

To upload your applications onto a running Amazon Linux instance, use **scp** or **sftp** and then configure the application by logging in to your instance. Your applications can also be uploaded during the instance launch by using the **PACKAGE_SETUP** action from the built-in cloud-init package. For more information, see Using cloud-init on AL2.

Security updates

Security updates are provided using the package repositories. Both security updates and updated AMI security alerts are published in the Amazon Linux Security Center. For more information about AWS security policies or to report a security problem, see AWS Cloud Security.

AL1 and AL2 are configured to download and install critical or important security updates at launch time. Kernel updates are not included in this configuration.

In AL2023, this configuration has changed compared to AL1 and AL2. For more information about security updates for AL2023, see Security updates and features in the *AL2023 User Guide*.

We recommend that you make the necessary updates for your use case after launch. For example, you might want to apply all updates (not just security updates) at launch, or evaluate each update and apply only the ones applicable to your system. This is controlled using the following cloud-init setting: `repo_upgrade`. The following snippet of cloud-init configuration shows how you can change the settings in the user data text you pass to your instance initialization:

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

The possible values for `repo_upgrade` are as follows:

`critical`

Apply outstanding critical security updates.

`important`

Apply outstanding critical and important security updates.

`medium`

Apply outstanding critical, important, and medium security updates.

`low`

Apply all outstanding security updates, including low-severity security updates.

`security`

Apply outstanding critical or important updates that Amazon marks as security updates.

`bugfix`

Apply updates that Amazon marks as bug fixes. Bug fixes are a larger set of updates, which include security updates and fixes for various other minor bugs.

`all`

Apply all applicable available updates, regardless of their classification.

`none`

Don't apply any updates to the instance on start up.

The default setting for `repo_upgrade` is security. That is, if you don't specify a different value in your user data, by default, Amazon Linux performs the security upgrades at launch for any packages installed at that time. Amazon Linux also notifies you of any updates to the installed packages by listing the number of available updates upon login using the `/etc/motd` file. To install these updates, you need to run **sudo yum upgrade** on the instance.

Repository configuration

For AL1 and AL2, AMIs are a snapshot of the packages available at the time the AMI was created, with the exception of security updates. Any packages not on the original AMI, but installed at runtime, will be the latest version available. To get the latest packages available for AL2, run **yum update -y**.

For AL2023, the repository configuration has changed compared to AL1 and AL2. For more information about the AL2023 repository, see Managing packages and operating system updates.

Versions up to AL2023 were configured to deliver a continuous flow of updates to roll from one minor version of Amazon Linux to the next version, also called *rolling releases*. As a best practice, we recommend you update your AMI to the latest available AMI rather than launching old AMIs and applying updates.

In-place upgrades are not supported between major Amazon Linux versions, such as from AL1 to AL2 or from AL2 to AL2023. For more information, see Amazon Linux availability.

Using cloud-init on AL2

The cloud-init package is an open-source application built by Canonical that is used to bootstrap Linux images in a cloud computing environment, such as Amazon EC2. Amazon Linux contains a customized version of cloud-init. This allows you to specify actions that should happen to your instance at boot time. You can pass desired actions to cloud-init through the user data fields when launching an instance. This means you can use common AMIs for many use cases and configure them dynamically at startup. Amazon Linux also uses cloud-init to perform initial configuration of the ec2-user account.

For more information, see the cloud-init documentation.

Amazon Linux uses the cloud-init actions found in `/etc/cloud/cloud.cfg.d` and `/etc/cloud/cloud.cfg`. You can create your own cloud-init action files in `/etc/cloud/cloud.cfg.d`. All files in this directory are read by cloud-init. They are read in lexical order, and later files overwrite values in earlier files.

The cloud-init package performs these (and other) common configuration tasks for instances at boot:

- Set the default locale.
- Set the hostname.
- Parse and handle user data.
- Generate host private SSH keys.
- Add a user's public SSH keys to `.ssh/authorized_keys` for easy login and administration.
- Prepare the repositories for package management.
- Handle package actions defined in user data.
- Run user scripts found in user data.
- Mount instance store volumes, if applicable.
  - By default, the `ephemeral0` instance store volume is mounted at `/media/ephemeral0` if it is present and contains a valid file system; otherwise, it is not mounted.
  - By default, any swap volumes associated with the instance are mounted (only for `m1.small` and `c1.medium` instance types).
  - You can override the default instance store volume mount with the following cloud-init directive:

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

    For more control over mounts, see Mounts in the cloud-init documentation.
  - Instance store volumes that support TRIM are not formatted when an instance launches, so you must partition and format them before you can mount them. For more information, see Instance store volume TRIM support. You can use the `disk_setup` module to partition and format your instance store volumes at boot. For more information, see Disk Setup in the cloud-init documentation.

Supported user data formats

The cloud-init package supports user data handling of a variety of formats:

- Gzip
  - If user data is gzip compressed, cloud-init decompresses the data and handles it appropriately.
- MIME multipart
  - Using a MIME multipart file, you can specify more than one type of data. For example, you could specify both a user data script and a cloud config type. Each part of the multipart file can be handled by cloud-init if it is one of the supported formats.
- Base64 decoding
  - If user data is base64-encoded, cloud-init determines if it can understand the decoded data as one of the supported types. If it understands the decoded data, it decodes the data and handles it appropriately. If not, it returns the base64 data intact.
- User data script
  - Begins with `#!` or `Content-Type: text/x-shellscript`.
  - The script is run by `/etc/init.d/cloud-init-user-scripts` during the first boot cycle. This occurs late in the boot process (after the initial configuration actions are performed).
- Include file
  - Begins with `#include` or `Content-Type: text/x-include-url`.
  - This content is an include file. The file contains a list of URLs, one per line. Each of the URLs is read, and their content passed through this same set of rules. The content read from the URL can be gzip compressed, MIME-multi-part, or plaintext.
- Cloud config data
  - Begins with `#cloud-config` or `Content-Type: text/cloud-config`.
  - This content is cloud config data.
- Upstart job (not supported on AL2)
  - Begins with `#upstart-job` or `Content-Type: text/upstart-job`.
  - This content is stored in a file in `/etc/init`, and upstart consumes the content as it does with other upstart jobs.
- Cloud boothook
  - Begins with `#cloud-boothook` or `Content-Type: text/cloud-boothook`.
  - This content is boothook data. It is stored in a file under `/var/lib/cloud` and then runs immediately.
  - This is the earliest *hook* available. There is no mechanism provided for running it only one time. The boothook must take care of this itself. It is provided with the instance ID in the

environment variable `INSTANCE_ID`. Use this variable to provide a once-per-instance set of boothook data.

Enable FIPS Mode on AL2

This section explains how to enable Federal Information Processing Standards (FIPS) on AL2. For more information about FIPS, see:

- Federal Information Processing Standard (FIPS)
- Compliance FAQs: Federal Information Processing Standards

Prerequisites

- An existing AL2 Amazon EC2 instance with access to the internet to download required packages. For more information about launching an AL2 Amazon EC2 instance, see AL2 on Amazon EC2.
- You must connect to your Amazon EC2 instance using SSH or AWS Systems Manager. .

> **Important**
>
> ED25519 SSH user keys aren't supported in FIPS mode. If you launched your Amazon EC2 instance using an ED25519 SSH key pair, you must generate new keys using another algorithm (such as RSA) or you may lose access to your instance after enabling FIPS mode. For more information see Create key pairs in the *Amazon EC2 User Guide for Linux Instances*.

Enable FIPS Mode

1. Connect to your AL2 instance using SSH or AWS Systems Manager.
2. Ensure the system is up to date. For more information, see Package repository.
3. Install and enable the `dracut-fips` module by running the following commmands.

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

4. Enable FIPS mode on the Linux kernel command-line using the following command.

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

5. Reboot your AL2 instance.

```
sudo reboot
```

6. To verify that FIPS mode is enabled, reconnect to your instance and run the following command.

```
sysctl crypto.fips_enabled
```

You should see the following output:

```
crypto.fips_enabled = 1
```

You can also verify that OpenSSH is in FIPS mode by running the following command:

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

You should see the following output:

```
FIPS mode initialized
```

AL2 release notifications

To be notified when new Amazon Linux AMIs are released, you can subscribe using Amazon SNS.

For information about subscribing to notifications for AL2023, see Receiving notifications on new updates in the *AL2023 User Guide*.

> **Note**
>
> Standard support for AL1 ended on December 31, 2020. The AL1 maintenance support phase ended December 31, 2023. For more information about the AL1 EOL and maintenance support, see the blog post Update on Amazon Linux AMI end-of-life.

To subscribe to Amazon Linux notifications

1. Open the Amazon SNS console at https://console.aws.amazon.com/sns/v3/home.
2. In the navigation bar, change the Region to **US East (N. Virginia)**, if necessary. You must select the Region in which the SNS notification that you are subscribing to was created.
3. In the navigation pane, choose **Subscriptions**, **Create subscription**.
4. For the **Create subscription** dialog box, do the following:
   a. [AL2] For **Topic ARN**, copy and paste the following Amazon Resource Name (ARN): **`arn:aws:sns:us-east-1:137112412989:amazon-linux-2-ami-updates`**.
   b. [Amazon Linux] For **Topic ARN**, copy and paste the following Amazon Resource Name (ARN): **`arn:aws:sns:us-east-1:137112412989:amazon-linux-ami-updates`**.
   c. For **Protocol**, choose **Email**.
   d. For **Endpoint**, enter an email address that you can use to receive the notifications.
   e. Choose **Create subscription**.
5. You receive a confirmation email with the subject line "AWS Notification - Subscription Confirmation". Open the email and choose **Confirm subscription** to complete your subscription.

Whenever AMIs are released, we send notifications to the subscribers of the corresponding topic. To stop receiving these notifications, use the following procedure to unsubscribe.

To unsubscribe from Amazon Linux notifications

1. Open the Amazon SNS console at https://console.aws.amazon.com/sns/v3/home.
2. In the navigation bar, change the Region to **US East (N. Virginia)**, if necessary. You must use the Region in which the SNS notification was created.
3. In the navigation pane, choose **Subscriptions**, select the subscription, and choose **Actions**, **Delete subscriptions**.
4. When prompted for confirmation, choose **Delete**.

Amazon Linux AMI SNS message format

The schema for the SNS message is as follows.

```
{
```

```
    "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"
    ]
}
```

Using Amazon Linux 2 outside of Amazon EC2

The AL2 container images can be run in compatible container runtime environments.

AL2 can also be run as a virtualized guest outside of directly being run on Amazon EC2.

> **Note**
>
> The configuration of AL2 images differs from AL2023.
> When migrating to AL2023, ensure that you review Using Amazon Linux 2023 outside of Amazon EC2 and adapt your configuration to be compatible with AL2023.

Run AL2 as a virtual machine on premises

Use the AL2 virtual machine (VM) images for on-premises development and testing. We offer a different AL2 VM image for each of the supported virtualization platforms. You can view the list of supported platforms on the Amazon Linux 2 virtual machine images page.

To use the AL2 virtual machine images with one of the supported virtualization platforms, do the following:

- Step 1: Prepare the seed.iso boot image
- Step 2: Download the AL2 VM image
- Step 3: Boot and connect to your new VM

Step 1: Prepare the `seed.iso` boot image

The `seed.iso` boot image includes the initial configuration information that is needed to boot your new VM, such as the network configuration, host name, and user data.

> **Note**
>
> The `seed.iso` boot image includes only the configuration information required to boot the VM. It does not include the AL2 operating system files.

To generate the `seed.iso` boot image, you need two configuration files:

- `meta-data` – This file includes the hostname and static network settings for the VM.
- `user-data` – This file configures user accounts, and specifies their passwords, key pairs, and access mechanisms. By default, the AL2 VM image creates an `ec2-user` user account. You use the `user-data` configuration file to set the password for the default user account.

To create the `seed.iso` boot disc

1. Create a new folder named `seedconfig` and navigate into it.
2. Create the `meta-data` configuration file.

   a. Create a new file named `meta-data`.

   b. Open the `meta-data` file using your preferred editor and add the following.

   ```
   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
   ```

   Replace *vm_hostname* with a VM host name of your choice, and configure the network settings as required.

   c. Save and close the `meta-data` configuration file.

   For an example `meta-data` configuration file that specifies a VM hostname (`amazonlinux.onprem`), configures the default network interface (`eth0`), and specifies static IP addresses for the necessary network devices, see the sample Seed.iso file.
3. Create the `user-data` configuration file.

   a. Create a new file named `user-data`.

   b. Open the `user-data` file using your preferred editor and add the following.

```
#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:'.
```

Replace *plain_text_password* with a password of your choice for the default `ec2-user` user account.

c. (Optional) By default, cloud-init applies network settings each time the VM boots. Add the following to prevent cloud-init from applying network settings at each boot, and to retain the network settings applied during the first boot.

```
# 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. Save and close the `user-data` configuration file.

You can also create additional user accounts and specify their access mechanisms, passwords, and key pairs. For more information about the supported directives, see Module reference. For an example `user-data` file that creates three additional users and specifies a custom password for the default `ec2-user` user account, see the sample Seed.iso file.

4. Create the `seed.iso` boot image using the `meta-data` and `user-data` configuration files.

For Linux, use a tool such as **genisoimage**. Navigate into the `seedconfig` folder, and run the following command.

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

For macOS, use a tool such as **hdiutil**. Navigate one level up from the `seedconfig` folder, and run the following command.

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

Step 2: Download the AL2 VM image

We offer a different AL2 VM image for each of the supported virtualization platforms. You can view the list of supported platforms and download the correct VM image for your chosen platform from the Amazon Linux 2 virtual machine images page.

Step 3: Boot and connect to your new VM

To boot and connect to your new VM, you must have the `seed.iso` boot image (created in Step 1) and an AL2 VM image (downloaded in Step 2). The steps vary depending on your chosen VM platform.

VMware vSphere

The VM image for VMware is made available in the OVF format.

To boot the VM using VMware vSphere

1. Create a new datastore for the `seed.iso` file, or add it to an existing datastore.
2. Deploy the OVF template, but do not start the VM yet.
3. In the **Navigator** panel, right-click the new virtual machine and choose **Edit Settings**.
4. On the **Virtual Hardware** tab, for **New device**, choose **CD/DVD Drive**, and then choose **Add**.
5. For **New CD/DVD Drive**, choose **Datastore ISO File**. Select the datastore to which you added the `seed.iso` file, browse to and select the `seed.iso` file, and then choose **OK**.
6. For **New CD/DVD Drive**, select **Connect**, and then choose **OK**.

After you have associated the datastore with the VM, you should be able to boot it.

KVM

To boot the VM using KVM

1. Open the **Create new VM** wizard.
2. For Step 1, choose **Import existing disk image**.
3. For Step 2, browse to and select the VM image. For **OS type** and **Version**, choose **Linux** and **Red Hat Enterprise Linux 7.0** respectively.
4. For Step 3, specify the amount of RAM and the number of CPUs to use.
5. For Step 4, enter a name for the new VM and select **Customize configuration before install**, and choose **Finish**.
6. In the Configuration window for the VM, choose **Add Hardware**.
7. In the **Add New Virtual Hardware** window, choose **Storage**.
8. In the Storage configuration, choose **Select or create custom storage**. For **Device type**, choose **CDROM device**. Choose **Manage**, **Browse Local**, and then navigate to and select the `seed.iso` file. Choose **Finish**.
9. Choose **Begin Installation**.

Oracle VirtualBox

To boot the VM using Oracle VirtualBox

1. Open Oracle VirtualBox and choose **New**.
2. For **Name**, enter a descriptive name for the virtual machine, and for **Type** and **Version**, select **Linux** and **Red Hat (64-bit)** respectively. Choose **Continue**.
3. For **Memory size**, specify the amount of memory to allocate to the virtual machine, and then choose **Continue**.
4. For **Hard disk**, choose **Use an existing virtual hard disk file**, browse to and open the VM image, and then choose **Create**.
5. Before you start the VM, you must load the `seed.iso` file in the virtual machine's virtual optical drive:
   a. Select the new VM, choose **Settings**, and then choose **Storage**.
   b. In the **Storage Devices** list, under **Controller: IDE**, choose the *Empty* optical drive.

c. In the **Attributes** section for the optical drive, choose the browse button, select **Choose Virtual Optical Disk File**, and then select the `seed.iso` file. Choose **OK** to apply the changes and close the Settings.

After you have added the `seed.iso` file to the virtual optical drive, you should be able to start the VM.

Microsoft Hyper-V

The VM image for Microsoft Hyper-V is compressed into a zip file. You must extract the contents of the zip file.

To boot the VM using Microsoft Hyper-V

1. Open the **New Virtual Machine Wizard**.
2. When prompted to select a generation, select **Generation 1**.
3. When prompted to configure the network adapter, for **Connection** choose **External**.
4. When prompted to connect a virtual hard disk, choose **Use an existing virtual hard disk**, choose **Browse**, and then navigate to and select the VM image. Choose **Finish** to create the VM.
5. Right-click the new VM and choose **Settings**. In the **Settings** window, under **IDE Controller 1**, choose **DVD Drive**.
6. For the DVD drive, choose **Image file** and then browse to and select the `seed.iso` file.
7. Apply the changes and start the VM.

After the VM has booted, log in using one of the user accounts that is defined in the `user-data` configuration file. After you have logged in for the first time, you can then disconnect the `seed.iso` boot image from the VM.

Identifying AL2

The following information describes how to identify an AL2 instance from another Amazon Linux version or other Linux distribution.

Identify Amazon Linux images

Each image contains a unique `/etc/image-id` file that identifies it. This file contains the following information about the image:

- `image_name`, `image_version`, `image_arch` – Values from the build recipe that Amazon used to construct the image.
- `image_stamp` – A unique, random hex value generated during image creation.
- `image_date` – The UTC time of image creation, in *YYYYMMDDhhmmss* format.
- `recipe_name`, `recipe_id` – The name and ID of the build recipe Amazon used to construct the image.

Amazon Linux contains an `/etc/system-release` file that specifies the current release that is installed. This file is updated using **yum** and is part of the `system-release` RPM Package Manager (RPM).

Amazon Linux also contains a machine-readable version of `/etc/system-release` that follows the Common Platform Enumeration (CPE) specification; see `/etc/system-release-cpe`.

AL2

The following is an example of `/etc/image-id` for the current version of AL2.

```
[ec2-user ~]$ cat /etc/image-id
      image_name="amzn2-ami-hvm"
image_version="2"
image_arch="x86_64"
image_file="amzn2-ami-hvm-2.0.20180810-x86_64.xfs.gpt"
image_stamp="8008-2abd"
image_date="20180811020321"
recipe_name="amzn2 ami"
recipe_id="c652686a-2415-9819-65fb-4dee-9792-289d-1e2846bd"
```

The following is an example of `/etc/system-release` for the current version of AL2.

```
[ec2-user ~]$ cat /etc/system-release
      AL2
```

The following is an example of `/etc/os-release` for 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/"
```

Amazon Linux AMI

The following is an example of `/etc/image-id` for the current Amazon Linux AMI.

```
[ec2-user ~]$ cat /etc/image-id
      image_name="amzn-ami-hvm"
image_version="2018.03"
image_arch="x86_64"
image_file="amzn-ami-hvm-2018.03.0.20180811-x86_64.ext4.gpt"
image_stamp="cc81-f2f3"
image_date="20180811012746"
recipe_name="amzn ami"
recipe_id="5b283820-dc60-a7ea-d436-39fa-439f-02ea-5c802dbd"
```

The following is an example of `/etc/system-release` for the current Amazon Linux AMI.

```
[ec2-user ~]$ cat /etc/system-release
      Amazon Linux AMI release 2018.03
```

AWS integration in AL2

AWS command line tools

The AWS Command Line Interface (AWS CLI) is an open source tool that provides a consistent interface to interact with AWS services using commands in your command-line shell. For more information, see What is the AWS Command Line Interface? in the *AWS Command Line Interface User Guide*.

AL2 and AL1 have version 1 of the AWS CLI preinstalled. The current release of Amazon Linux, AL2023, has version 2 of the AWS CLI preinstalled. For more information about using the AWS CLI on AL2023, see Get started with AL2023 in the *AL2023 User Guide*.

Getting started with programming runtimes

AL2 provides different versions of certain language runtimes. We work with upstream projects, such as PHP, that support multiple versions at the same time. To find information about how to install and manage these name-versioned packages, use the `yum` command to search and install these packages. For more information, see Package repository.

The following topics describe how each language runtime functions in AL2.

Topics

- C, C++, and Fortran in AL2
- Go in AL2
- Java in AL2
- Perl in AL2
- PHP in AL2
- Python in AL2
- Rust in AL2

C, C++, and Fortran in AL2

AL2 includes both the GNU Compiler Collection (GCC) and the Clang frontend for LLVM.

The major version of GCC will remain constant throughout the lifetime of AL2. Bug and security fixes might be backported to the major version of GCC that ships in AL2.

By default, AL2 includes version 7.3 of GCC which builds almost all packages. The `gcc10` package makes GCC 10 available to a limited extent, but we don't recommend using GCC 10 to build packages.

The default compiler flags that build AL2 RPMs include some optimization and hardening flags. We recommend that you include some optimization and hardening flags if you are building your own code with GCC.

The default compiler and optimization flags in AL2023 improve upon what is present in AL2.

Go in AL2

You might want to build your own code written in Go on Amazon Linux using a toolchain provided with AL2.

The Go toolchain will be updated throughout the life of AL2. This might be in response to any CVE in the toolchain we ship, or as a prerequisite of addressing a CVE in another package.

Go is a relatively fast moving programming language. There might be a situation where existing applications written in Go have to adapt to new versions of the Go toolchain. For more information about Go, see Go 1 and the Future of Go Programs.

Although AL2 will incorporate new versions of the Go toolchain during its life, this will not be in lockstep with the upstream Go releases. Therefore, using the Go toolchain provided in AL2 might not be suitable if you want to build Go code using cutting-edge features of the Go language and standard library.

During the lifetime of AL2, earlier package versions are not removed from the repositories. If an earlier Go toolchain is required, you can choose to forgo bug and security fixes of newer Go toolchains and install an earlier version from the repositories using the same mechanisms available for any RPM.

If you want to build your own Go code on AL2 you can use the Go toolchain included in AL2 with the knowledge that this toolchain might move forward through the lifetime of AL2.

Java in AL2

AL2 provides several versions of Amazon Corretto to support Java based workloads, as well as some OpenJDK versions. We recommend that you migrate to Amazon Corretto in preparation for migrating to AL2023.

Corretto is a build of the Open Java Development Kit (OpenJDK) with long-term support from Amazon. Corretto is certified using the Java Technical Compatibility Kit (TCK) to ensure it meets the Java SE standard and is available on Linux, Windows, and macOS.

An Amazon Corretto package is available for each of Corretto 1.8.0, Corretto 11, and Corretto 17.

Each Corretto version in AL2 is supported for the same period of time as the Corretto version is, or until the end of life of AL2, whichever is sooner. For more information, see the Amazon Corretto FAQs.

Perl in AL2

AL2 provides version 5.16 of the Perl programming language.

Perl modules in AL2

Various Perl modules are packaged as RPMs in AL2. Although there are many Perl modules available as RPMs, Amazon Linux does not try to package every possible Perl module. Modules packaged as RPMs might be relied upon by other operating system RPM packages, so Amazon Linux will prioritize ensuring they are security patched over pure feature updates.

AL2 also includes CPAN so that Perl developers can use the idiomatic package manager for Perl modules.

PHP in AL2

AL2 currently provides two fully supported versions of the PHP programming language as part of AL2 Extras. Each PHP version is supported for the same time frame as upstream PHP as listed under deprecated date in List of Amazon Linux 2 Extras.

To assist migration to AL2023, both PHP 8.1 and 8.2 are available on AL2 and AL2023.

> **Note**
>
> AL2 includes PHP 7.1, 7.2, 7.3, and 7.4 in `amazon-linux-extras`. All of these Extras are EOL and are not guaranteed to get any additional security updates.
> To find out when each version of PHP is deprecated in AL2, see the List of Amazon Linux 2 Extras.

Migrating from earlier PHP 8.x versions

The upstream PHP community put together comprehensive migration documentation for moving to PHP 8.2 from PHP 8.1. Documentation also exists for migrating from PHP 8.0 to 8.1.

AL2 includes PHP 8.0, 8.1, and 8.2 in `amazon-linux-extras` that enables an efficient upgrade path to AL2023. To find out when each version of PHP is deprecated in AL2, see the List of Amazon Linux 2 Extras.

Migrating from PHP 7.x versions

The upstream PHP community put together comprehensive migration documentation for moving to PHP 8.0 from PHP 7.4. Combined with the documentation referenced in the previous section on migrating to PHP 8.1, and PHP 8.2, you have all of the steps needed to migrate your PHP based application to modern PHP.

The PHP project maintains a list and schedule of supported versions, along with a list of unsupported branches.

> **Note**
>
> When AL2023 was released, all 7.x and 5.x versions of PHP were not supported by the PHP community, and were not included as options in AL2023.

Python in AL2

AL2 provides support and security patches for Python 2.7 until June 2025, as part of our long-term support commitment for AL2 core packages. This support extends beyond the upstream Python community declaration of Python 2.7 EOL of January 2020.

> **Note**
>
> AL2023 completely removed Python 2.7. Any components requiring Python are now written to work with Python 3.

AL2 uses the `yum` package manager that has a hard dependency on Python 2.7. In AL2023, the `dnf` package manager has migrated to Python 3, and no longer requires Python 2.7. AL2023 has completely moved to Python 3. We recommend that you complete your migration to Python 3.

Rust in AL2

You might want to build your own code written in Rust on AL2 using a toolchain provided with AL2.

The Rust toolchain will be updated throughout the life of AL2. This might be in response to a CVE in the toolchain we ship, or as prerequisite for a CVE update in another package.

Rust is a relatively fast moving language, with new releases at approximately a six-week cadence. The new releases might add new language or standard library features. Although AL2 will incorporate new versions of the Rust toolchain during its life, this will not be in lockstep with the upstream Rust releases. Therefore, using the Rust toolchain provided in AL2 might not be suitable if you want to build Rust code using cutting-edge features of the Rust language.

During the lifetime of AL2, previous package versions are not removed from the repositories. If a previous Rust toolchain is required, you can choose to forgo bug and security fixes of newer Rust toolchains and install a previous version from the repositories using the same processes available for any RPM.

To build your own Rust code on AL2, use the Rust toolchain included in AL2 with the knowledge that this toolchain might move forward throughout the lifetime of AL2.

AL2 kernel

AL2 originally shipped with a 4.14 kernel, with version 5.10 as the current default. If you are still using a 4.14 kernel, you are encouraged to migrate to the 5.10 kernel.

Kernel live patching is supported on AL2.

Topics

- AL2 supported kernels
- Kernel Live Patching on AL2

AL2 supported kernels

Supported kernel versions

Currently, AL2 AMIs are available with kernel versions 4.14 and 5.10, with version 5.10 as the default. We recommend that you use an AL2 AMI with kernel 5.10.

AL2023 AMIs are available with kernel version 6.1. For more information, see AL2023 Kernel changes from AL2 in the *AL2023 User Guide*.

Support Timeframe

The 5.10 kernel available on AL2 will be supported until the AL2 AMI reaches the end of standard support.

Live patching support

AL2 kernel version	Kernel live patching supported
4.14	Yes
5.10	Yes
5.15	No

Kernel Live Patching on AL2

Kernel Live Patching for AL2 allows you to apply security vulnerability and critical bug patches to a running Linux kernel, without reboots or disruptions to running applications. This allows you to benefit from improved service and application availability, while keeping your infrastructure secure and up to date.

For information about Kernel Live Patching for AL2023, see Kernel Live Patching on AL2023 in the *AL2023 User Guide*.

AWS releases two types of kernel live patches for AL2:

- **Security updates** – Include updates for Linux common vulnerabilities and exposures (CVE). These updates are typically rated as *important* or *critical* using the Amazon Linux Security Advisory ratings. They generally map to a Common Vulnerability Scoring System (CVSS) score of 7 and higher. In some cases, AWS might provide updates before a CVE is assigned. In these cases, the patches might appear as bug fixes.
- **Bug fixes** – Include fixes for critical bugs and stability issues that are not associated with CVEs.

AWS provides kernel live patches for an AL2 kernel version for up to 3 months after its release. After the 3-month period, you must update to a later kernel version to continue to receive kernel live patches.

AL2 kernel live patches are made available as signed RPM packages in the existing AL2 repositories. The patches can be installed on individual instances using existing **yum** workflows, or they can be installed on a group of managed instances using AWS Systems Manager.

Kernel Live Patching on AL2 is provided at no additional cost.

Topics

- Supported configurations and prerequisites
- Work with Kernel Live Patching
- Limitations
- Frequently asked questions

Supported configurations and prerequisites

Kernel Live Patching is supported on Amazon EC2 instances and on-premises virtual machines running AL2.

To use Kernel Live Patching on AL2, you must use:

- Kernel version `4.14` or `5.10` on the `x86_64` architecture
- Kernel version `5.10` on the `ARM64` architecture

Policy Requirements

To download packages from Amazon Linux repositories, Amazon EC2 needs access to service-owned Amazon S3 buckets. If you are using a Amazon Virtual Private Cloud (VPC) endpoint for Amazon S3 in your environment, you need to ensure that your VPC endpoint policy allows access to those public buckets.

The table describes each of the Amazon S3 buckets that EC2 might need to access for Kernel Live Patching.

S3 bucket ARN	Description
arn:aws:s3:::packages.*region*.amazonaws.com/*	Amazon S3 bucket containing Amazon Linux AMI packages
arn:aws:s3:::repo.*region*.amazonaws.com/*	Amazon S3 bucket containing Amazon Linux AMI repositories
arn:aws:s3:::amazonlinux.*region*.amazonaws.com/*	Amazon S3 bucket containing AL2 repositories
arn:aws:s3:::amazonlinux-2-repos-*region*/*	Amazon S3 bucket containing AL2 repositories

The following policy illustrates how to restrict access to identities and resources that belong to your organization and provide access to the Amazon S3 buckets required for Kernel Live Patching. Replace *`region`*, *`principal-org-id`* and *`resource-org-id`* with your organization's values.

```
{
  "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/*"
      ]
    }
  ]
}
```

Work with Kernel Live Patching

You can enable and use Kernel Live Patching on individual instances using the command line on the instance itself, or you can enable and use Kernel Live Patching on a group of managed instances using AWS Systems Manager.

The following sections explain how to enable and use Kernel Live Patching on individual instances using the command line.

For more information about enabling and using Kernel Live Patching on a group of managed instances, see Use Kernel Live Patching on AL2 instances in the *AWS Systems Manager User Guide*.

Topics

- Enable Kernel Live Patching
- View the available kernel live patches
- Apply kernel live patches
- View the applied kernel live patches
- Disable Kernel Live Patching

Enable Kernel Live Patching

Kernel Live Patching is disabled by default on AL2. To use live patching, you must install the **yum** plugin for Kernel Live Patching and enable the live patching functionality.

Prerequisites

Kernel Live Patching requires `binutils`. If you do not have `binutils` installed, install it using the following command:

```
$ sudo yum install binutils
```

To enable Kernel Live Patching

1. Kernel live patches are available for the following AL2 kernel versions:

   - Kernel version `4.14` or `5.10` on the `x86_64` architecture
   - Kernel version `5.10` on the `ARM64` architecture

To check your kernel version, run the following command.

```
$ sudo yum list kernel
```

2. If you already have a supported kernel version, skip this step. If you do not have a supported kernel version, run the following commands to update the kernel to the latest version and to reboot the instance.

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

```
$ sudo reboot
```

3. Install the **yum** plugin for Kernel Live Patching.

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

4. Enable the **yum** plugin for Kernel Live Patching.

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

This command also installs the latest version of the kernel live patch RPM from the configured repositories.

5. To confirm that the **yum** plugin for kernel live patching has installed successfully, run the following command.

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

When you enable Kernel Live Patching, an empty kernel live patch RPM is automatically applied. If Kernel Live Patching was successfully enabled, this command returns a list that includes the initial empty kernel live patch RPM. The following is example output.

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

6. Install the **kpatch** package.

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

7. Update the **kpatch** service if it was previously installed.

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

8. Start the **kpatch** service. This service loads all of the kernel live patches upon initialization or at boot.

```
$ sudo systemctl enable kpatch.service
```

9. Enable the Kernel Live Patching topic in the AL2 Extras Library. This topic contains the kernel live patches.

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

View the available kernel live patches

Amazon Linux security alerts are published to the Amazon Linux Security Center. For more information about the AL2 security alerts, which include alerts for kernel live patches, see the Amazon Linux Security Center. Kernel live patches are prefixed with `ALASLIVEPATCH`. The Amazon Linux Security Center might not list kernel live patches that address bugs.

You can also discover the available kernel live patches for advisories and CVEs using the command line.

To list all available kernel live patches for advisories

Use the following command.

```
$ yum updateinfo list
```

The following shows example output.

```
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
```

To list all available kernel live patches for CVEs

Use the following command.

```
$ yum updateinfo list cves
```

The following shows example output.

```
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
```

Apply kernel live patches

You apply kernel live patches using the **yum** package manager in the same way that you would apply regular updates. The **yum** plugin for Kernel Live Patching manages the kernel live patches that are to be applied and eliminates the need to reboot.

> **Tip**
>
> We recommend that you update your kernel regularly using Kernel Live Patching to ensure that it remains secure and up to date.

You can choose to apply a specific kernel live patch, or to apply any available kernel live patches along with your regular security updates.

To apply a specific kernel live patch

1. Get the kernel live patch version using one of the commands described in View the available kernel live patches.
2. Apply the kernel live patch for your AL2 kernel.

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

   For example, the following command applies a kernel live patch for AL2 kernel version `5.10.102-99.473`.

```
$ sudo yum install kernel-livepatch-5.10.102-99.473-1.0-4.amzn2.x86_64
```

To apply any available kernel live patches along with your regular security updates

Use the following command.

```
$ sudo yum update --security
```

Omit the `--security` option to include bug fixes.

Important

- The kernel version is not updated after applying kernel live patches. The version is only updated to the new version after the instance is rebooted.
- An AL2 kernel receives kernel live patches for a period of three months. After the three month period has lapsed, no new kernel live patches are released for that kernel version. To continue to receive kernel live patches after the three-month period, you must reboot the instance to move to the new kernel version, which will then continue receiving kernel live patches for the next three months. To check the support window for your kernel version, run `yum kernel-livepatch supported`.

View the applied kernel live patches

To view the applied kernel live patches

Use the following command.

```
$ kpatch list
```

The command returns a list of the loaded and installed security update kernel live patches. The following is example output.

```
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

A single kernel live patch can include and install multiple live patches.

Disable Kernel Live Patching

If you no longer need to use Kernel Live Patching, you can disable it at any time.

To disable Kernel Live Patching

1. Remove the RPM packages for the applied kernel live patches.

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

2. Uninstall the **yum** plugin for Kernel Live Patching.

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

3. Reboot the instance.

   ```
   $ sudo reboot
   ```

Limitations

Kernel Live Patching has the following limitations:

- While applying a kernel live patch, you can't perform hibernation, use advanced debugging tools (such as SystemTap, kprobes, and eBPF-based tools), or access ftrace output files used by the Kernel Live Patching infrastructure.

Frequently asked questions

For frequently asked questions about Kernel Live Patching for AL2, see the Amazon Linux 2 Kernel Live Patching FAQ.

AL2 Extras

With AL2, you can use the Extras Library to install application and software updates on your instances. These software updates are known as *topics*. You can install a specific version of a topic or omit the version information to use the most recent version. Extras help alleviate having to compromise between the stability of an operating system and the freshness of available software.

The contents of Extras topics are exempt from the Amazon Linux policy on long-term support and binary compatibility. Extras topics provide access to a curated list of packages. The versions of the packages might be updated frequently or might not be supported for the same amount of time as AL2.

> **Note**
>
> Individual Extras topics might be deprecated before AL2 reaches EOL.

To list the available topics, use the following command.

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

To enable a topic and install the latest version of its package to ensure freshness, use the following command.

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

To enable topics and install specific versions of their packages to ensure stability, use the following command.

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

To remove a package installed from a topic, use the following command.

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

Note

This command does not remove packages that were installed as dependencies of the Extra.

To disable a topic and make the packages inaccessible to the yum package manager, use the following command.

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

Important

This command is intended for advanced users. Improper usage of this command could cause package compatibility conflicts.

List of Amazon Linux 2 Extras

Extra name	Deprecated date
BCC	
GraphicsMagick1.3	
R3.4	
R4	
ansible2	2023-09-30
aws-nitro-enclaves-cli	
awscli1	
collectd	
collectd-python3	
corretto8	2025-06-30

Extra name	Deprecated date
dnsmasq	
dnsmasq2.85	
docker	2025-06-30
ecs	
emacs	2018-11-14
epel	
firecracker	2022-11-08
firefox	
gimp	2018-11-14
golang1.11	2023-08-01
golang1.19	2023-09-30
golang1.9	2018-12-14
haproxy2	
httpd_modules	
java-openjdk11	2024-09-30
kernel-5.10	
kernel-5.15	
kernel-5.4	
kernel-ng	2022-08-08
lamp-mariadb10.2-php7.2	2020-11-30

Extra name	Deprecated date
libreoffice	
livepatch	
lustre	
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
php7.1	2020-01-15
php7.2	2020-11-30
php7.3	2021-12-06
php7.4	2022-11-03
php8.0	2023-11-26
php8.1	2024-11-25

Extra name	Deprecated date
php8.2	2025-06-30
postgresql10	2023-09-30
postgresql11	2023-11-09
postgresql12	2024-11-14
postgresql13	2025-06-30
postgresql14	2025-06-30
postgresql9.6	2022-08-09
python3	2018-08-22
python3.8	2024-10-14
redis4.0	2021-05-25
redis6	
ruby2.4	2020-08-27
ruby2.6	2023-03-31
ruby3.0	2024-03-31
rust1	
selinux-ng	
squid4	2023-09-30
testing	
tomcat8.5	2024-03-31
tomcat9	

Extra name	Deprecated date
unbound1.13	
unbound1.17	
vim	2018-11-14

AL2 Source Packages

You can view the source of packages you have installed on your instance for reference purposes by using tools provided in Amazon Linux. Source packages are available for all of the packages included in Amazon Linux and the online package repository. Determine the package name for the source package you want to install and use the **yumdownloader --source** command to view source within your running instance. For example:

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

The source RPM can be unpacked and, for reference, you can view the source tree using standard RPM tools. After you finish debugging, the package is available for use.