



浪潮信息云峦服务器操作系统 KeyarchOS

yum 源配置手册

浪潮电子信息产业股份有限公司

2023 年 12 月

目录

1 概述.....	3
1.1 文档简介.....	3
1.2 适用对象.....	3
1.3 适用范围.....	3
2 KeyarchOS 官方 yum 源使用	4
3 阿里云镜像站使用方法.....	8
4 本地 yum 源搭建方法.....	10

1 概述

1.1 文档简介

本文档用于指导用户使用浪潮信息云峦服务器操作系统 KeyarchOS 的软件仓库。

1.2 适用对象

本文档旨在帮助用户使用 KeyarchOS 操作系统的软件仓库。

本文档主要面向以下人员：

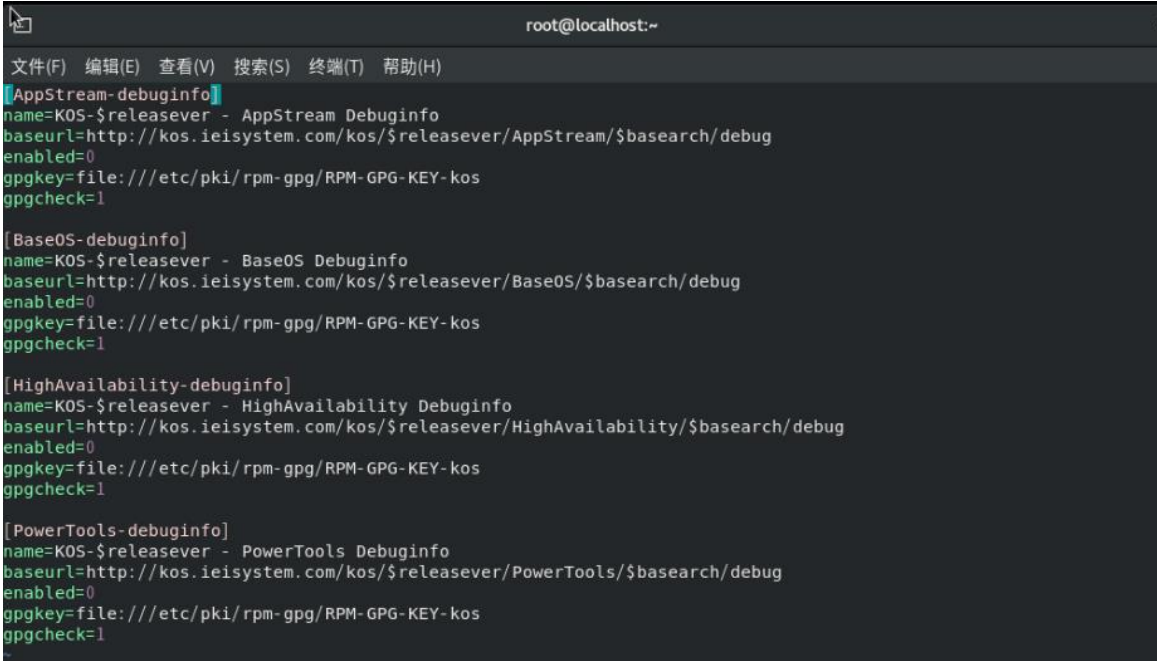
- ◆ 系统管理员
- ◆ 维护工程师
- ◆ 普通用户

1.3 适用范围

本文档为服务器 KeyarchOS 操作系统 yum 源配置手册，更多相关资源，请访问浪潮信息官网或联系浪潮信息技术人员。

2 KeyarchOS 官方 yum 源使用

KeyarchOS 默认已配置官方镜像源，可以直接使用官方镜像源。源的配置文件位于 `/etc/yum.repos.d`，其中 `KOS-AppStream.repo`、`KOS-BaseOS.repo`、`KOS-HighAvailability.repo`、`KOS-PowerTools.repo` 源默认开启，`KOS-Debuginfo.repo`、`KOS-Source.repo` 源默认关闭；可以通过更改 repo 文件中的 `enabled` 的值来开启和关闭 repo(值 0 为关闭，1 为启用)。



```
root@localhost:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
[AppStream-debuginfo]  
name=KOS-$releasever - AppStream Debuginfo  
baseurl=http://kos.ieisystem.com/kos/$releasever/AppStream/$basearch/debug  
enabled=0  
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-kos  
gpgcheck=1  
  
[BaseOS-debuginfo]  
name=KOS-$releasever - BaseOS Debuginfo  
baseurl=http://kos.ieisystem.com/kos/$releasever/BaseOS/$basearch/debug  
enabled=0  
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-kos  
gpgcheck=1  
  
[HighAvailability-debuginfo]  
name=KOS-$releasever - HighAvailability Debuginfo  
baseurl=http://kos.ieisystem.com/kos/$releasever/HighAvailability/$basearch/debug  
enabled=0  
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-kos  
gpgcheck=1  
  
[PowerTools-debuginfo]  
name=KOS-$releasever - PowerTools Debuginfo  
baseurl=http://kos.ieisystem.com/kos/$releasever/PowerTools/$basearch/debug  
enabled=0  
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-kos  
gpgcheck=1
```

图 2- 1 镜像源仓库设置

使用 KeyarchOS 官方镜像源可以快速地下载需要的软件包，如需要安装某个软件包，可以执行“`yum -y install 包名`”命令来进行安装：

```
[root@localhost ~]# yum install rpm-build
上次元数据过期检查: 3:25:27 前, 执行于 2023年12月25日 星期一 18时30分02秒。
依赖关系解决。
```

软件包	架构	版本	仓库	大小
安装:				
rpm-build	x86_64	4.14.3-24.0.1.kos5	AppStream	173 k
安装依赖关系:				
annobin	x86_64	10.67-3.0.1.kos5	AppStream	954 k
ctags	x86_64	5.8-23.0.1.kos5	AppStream	144 k
dwz	x86_64	0.12-10.0.1.kos5	AppStream	108 k
efi-srpm-macros	noarch	3-3.0.3.kos5	AppStream	21 k
gc	x86_64	7.6.4-3.0.2.kos5	AppStream	108 k
gdb-headless	x86_64	9.2-7.0.4.kos5	AppStream	4.0 M
ghc-srpm-macros	noarch	1.4-2-7.kos5	AppStream	8.2 k
go-srpm-macros	noarch	2-17.0.2.kos5	AppStream	12 k
guile	x86_64	5:2.0.14-7.0.2.kos5	AppStream	3.4 M
libatomic_ops	x86_64	7.6.2-3.0.2.kos5	AppStream	26 k
libtpt	x86_64	1.6-1-8.0.1.kos5	AppStream	48 k
ocaml-srpm-macros	noarch	5-4.0.1.kos5	AppStream	8.3 k
openblas-srpm-macros	noarch	2-2.0.1.kos5	AppStream	6.9 k
perl-srpm-macros	noarch	1-25.kos5	AppStream	9.6 k
python-rpm-macros	noarch	3-43.kos5	AppStream	14 k
python-srpm-macros	noarch	3-43.kos5	AppStream	14 k
python3-rpm-macros	noarch	3-43.kos5	AppStream	14 k
qt5-srpm-macros	noarch	5.15.3-1.0.1.kos5	AppStream	9.6 k
rust-srpm-macros	noarch	5-2.0.1.kos5	AppStream	8.2 k
source-highlight	x86_64	3.1.8-17.0.2.kos5	AppStream	466 k
system-rpm-config	noarch	129-1.0.2.kos5	PowerTools	88 k
zstd	x86_64	1.4.4-1.0.1.kos5	AppStream	380 k

```

事务概要
-----
安装 23 软件包

```

图 2- 2 软件安装

如需卸载某个软件包，可以执行“yum -y remove 包名”命令来进行卸载：

```
[root@localhost ~]# yum -y remove vim
依赖关系解决。
```

软件包	架构	版本	仓库	大小
移除:				
vim-enhanced	x86_64	2:8.0.1763-19.0.1.kos5.4	@AppStream	2.9 M
清除未被使用的依赖关系:				
gpm-libs	x86_64	1.20.7-17.0.1.kos5	@anaconda	28 k
vim-common	x86_64	2:8.0.1763-19.0.1.kos5.4	@AppStream	27 M
vim-filesystem	noarch	2:8.0.1763-19.0.1.kos5.4	@anaconda	40

```

事务概要
-----
移除 4 软件包

将会释放空间: 30 M
运行事务检查
事务检查成功。
运行事务测试
事务测试成功。
运行事务
准备中:
删除: vim-enhanced-2:8.0.1763-19.0.1.kos5.4.x86_64 1/1
删除: vim-common-2:8.0.1763-19.0.1.kos5.4.x86_64 1/4
删除: vim-filesystem-2:8.0.1763-19.0.1.kos5.4.noarch 2/4
删除: gpm-libs-1.20.7-17.0.1.kos5.x86_64 3/4
运行脚本: gpm-libs-1.20.7-17.0.1.kos5.x86_64 4/4
验证: gpm-libs-1.20.7-17.0.1.kos5.x86_64 4/4
验证: vim-common-2:8.0.1763-19.0.1.kos5.4.x86_64 1/4
验证: vim-enhanced-2:8.0.1763-19.0.1.kos5.4.x86_64 2/4
验证: vim-filesystem-2:8.0.1763-19.0.1.kos5.4.noarch 3/4
已移除:
gpm-libs-1.20.7-17.0.1.kos5.x86_64 vim-common-2:8.0.1763-19.0.1.kos5.4.x86_64
vim-enhanced-2:8.0.1763-19.0.1.kos5.4.x86_64 vim-filesystem-2:8.0.1763-19.0.1.kos5.4.noarch

```

图 2- 3 软件卸载

如需使用 module，可以执行“yum module list”来查看 module 是否启用，其中 d 为默认，e 为已启用，x 为已禁用，i 为已安装。

```

root@localhost:~# yum module list
OS-5.8 - AppStream
Name           Stream           Profiles Summary
python38       1.4 [d]          389 Directory Server (base)
python36       1.10 [d]         common [ Java build tool
python36       1.10 [d]         d]
podman         an8 [d][e]       common [ Most recent (rolling) versi
podman         an8 [d][e]       d]
podman         an8 [d][e]       ons of podman, buildah, sko
podman         an8 [d][e]       peo, runc, common, runc, co
podman         an8 [d][e]       mmon, CRIU, Udraca, etc as w
podman         an8 [d][e]       ell as dependencies such as
podman         an8 [d][e]       container-selinux built an
podman         an8 [d][e]       d tested together, and upda
podman         an8 [d][e]       ted as frequently as every
podman         an8 [d][e]       12 weeks
podman         2.0              common [ Stable versions of podman 1
podman         2.0              d]
podman         2.0              .6, buildah 1.11, skopeo 0.
podman         2.0              1, runc, common, etc as wel
podman         2.0              l as dependencies such as c
podman         2.0              ontainer-selinux built and
podman         2.0              tested together, and suppor
podman         2.0              ted as documented on the Ap
podman         2.0              plication Stream lifecycle
podman         2.0              page.
podman         3.0              common [ Stable versions of podman 3
podman         3.0              d]
podman         3.0              .0, buildah 1.19, skopeo 1.
podman         3.0              2, runc, common, etc as wel
podman         3.0              l as dependencies such as c
podman         3.0              ontainer-selinux built and
podman         3.0              tested together, and suppor
podman         3.0              ted as documented on the Ap
podman         3.0              plication Stream lifecycle
podman         3.0              page.
podman         4.0              common [ Most recent (rolling) versi

```

图 2- 4 module 模块列表

如需启用某个 module(例如需要启用 python38),可以执行“yum module enable python38”命令来进行启用 :

```

[root@localhost ~]# yum module enable python38
上次元数据过期检查: 2:08:51 前, 执行于 2023年06月07日 星期三 17时50分43秒。
依赖关系解决。
-----
软件包           架构           版本           仓库           大小
-----
启用模块流:
python38         x86_64         3.8
-----
事务概要
-----
确定吗? [y/N]:

```

图 2- 启用 module 模块

如需禁用某个 module (例如需要禁用 python36), 可以执行“yum module disable python36”命令来进行禁用 :

```
[root@localhost ~]# yum module disable python36
上次元数据过期检查: 2:10:10 前, 执行于 2023年06月07日 星期三 17时50分43秒。
依赖关系解决。
=====
软件包                架构                版本                仓库                大小
=====
禁用模块:
python36

事务概要
=====
确定吗? [y/N]: █
```

图 2- 禁用 module 模块

如需安装某个 module(例如需要安装 python27),可以执行“yum module install python27”命令来进行安装 :

```
[root@localhost ~]# yum module install python27
上次元数据过期检查: 2:06:44 前, 执行于 2023年06月07日 星期三 17时50分43秒。
依赖关系解决。
=====
软件包                架构                版本                仓库                大小
=====
安装组/模块包:
python2                x86_64                2.7.18-10.0.1.module_kos5+123+49ba50ab    AppStream            109 k
python2-libs            x86_64                2.7.18-10.0.1.module_kos5+123+49ba50ab    AppStream            6.0 M
python2-pip              noarch                9.0.3-19.module_kos5+123+49ba50ab         AppStream            1.6 M
python2-setuptools       noarch                39.0.1-13.module_kos5+123+49ba50ab         AppStream            641 k
安装依赖关系:
python2-pip-wheel        noarch                9.0.3-19.module_kos5+123+49ba50ab         AppStream            891 k
python2-setuptools-wheel noarch                39.0.1-13.module_kos5+123+49ba50ab         AppStream            286 k
安装模块配置档案:
python27/common

事务概要
=====
安装 6 软件包

总下载: 9.5 M
安装大小: 37 M
确定吗? [y/N]: █
```

图 2- 7 安装 module 模块

如需卸载某个 module (例如需要卸载 python27), 可以执行“yum module remove python27”命令来进行卸载 :

```

[root@localhost ~]# yum module remove python27
上次元数据过期检查: 2:07:44 前, 执行于 2023年06月07日 星期三 17时50分43秒。
依赖关系解决。
=====
软件包                                架构      版本                                仓库      大小
=====
移除:
python2                                x86_64    2.7.18-10.0.1.module_kos5+123+49ba50ab  @AppStream  80 k
python2-libs                            x86_64    2.7.18-10.0.1.module_kos5+123+49ba50ab  @AppStream  25 M
python2-pip                              noarch    9.0.3-19.module_kos5+123+49ba50ab       @AppStream  7.5 M
python2-setuptools                       noarch    39.0.1-13.module_kos5+123+49ba50ab       @AppStream  3.0 M
清除未被使用的依赖关系:
python2-pip-wheel                       noarch    9.0.3-19.module_kos5+123+49ba50ab       @AppStream  906 k
python2-setuptools-wheel                 noarch    39.0.1-13.module_kos5+123+49ba50ab       @AppStream  337 k
禁用模块配置档案:
python27/common

事务概要
-----
移除 6 软件包

将会释放空间: 37 M
确定吗? [y/N]: █

```

图 2- 8 卸载 module 模块

3 国内其他镜像站使用方法

为应对 KeyarchOS 官方镜像站因为网络或升级问题暂时无法使用的情况，KeyarchOS 在阿里云镜像站和上海交通大学镜像站有 yum 软件仓库的镜像源，地址为：

阿里云镜像站：<https://mirrors.aliyun.com/keyarchos/>

上海交通大学镜像站：<https://mirror.sjtu.edu.cn/keyarchos/>

一、配置方法为：

1、备份原配置文件

```
# cd /etc/yum.repos.d
```

```
# mkdir bak
```



```
# cp *.repo bak
```

2、地址替换

阿里镜像站：

```
# sudo sed -i -e  
"s|http://kos.ieisystem.com/kos/|https://mirrors.aliyun.com/keyarchos/|g"  
/etc/yum.repos.d/KOS*.repo
```

交大镜像站：

```
# sudo sed -i -e  
"s|http://kos.ieisystem.com/kos/|https://mirror.sjtu.edu.cn/keyarchos/|g"  
/etc/yum.repos.d/KOS*.repo
```

3、运行 yum makecache 生成缓存

```
# yum clean all
```

```
# yum makecache
```

至此，可以使用阿里云镜像站或者上海交通大学镜像站 KeyarchOS 软件源。

4 本地 yum 源搭建方法

如果存在不能访问网络的情况，用户可以搭建本地 yum 源，具体步骤如下：

1、下载 iso 至 linux 本地

从 KOS 官网下载 everything ISO 至本地，假设是/root 目录，x86 的环境下
载 x86 的 ISO，arm 的环境请下载 arm 的 ISO

2、创建挂载目录

```
假设是/mnt/iso 目录：# mkdir -p /mnt/iso
```

3、挂载镜像(xxx 请用具体镜像名称替代)

```
# mount /root/xxx.iso /mnt/iso
```

4、配置本地 yum 源配置文件

```
# cd /etc/yum.repos.d/
```

```
# mkdir bak
```

```
# mv *.repo bak
```

```
# vim local.repo
```

将下边内容复制到 local.repo 中：

```
[local-appstream]
```

```
name=local-appstream
```

```
baseurl=file:///mnt/iso/AppStream
```

```
enabled=1
```

```
gpgcheck=0
```

```
[local-baseos]
```

```
name=local-baseos
```

```
baseurl=file:///mnt/iso/BaseOS
```

```
enabled=1
```

```
gpgcheck=0
```

现在，可以使用本地 yum 源了