

# GNU/Linux 软件包安装



# 软件包管理 - YUM

YUM:

全称 :Yellow dog Updater Modified

功能：一个基於 **RPM** 包管理，能够从指定的服务器自动下载 RPM 包并且安装，可以自动处理依赖性关系，并且一次安装所有依赖的软体包。

特点：yum 提供了查找、安装、删除某一个、一组甚至全部软件包的命令，

# 软件包管理 - YUM

YUM:

特点：

1. yum 提供了查找、安装、删除某一个、一组甚至全部软件包的命令。
2. 所有软件包全部都在互联网中，能够实现非常快速的更新、升级
3. 采取的 **C/S** 架构模型

# 软件包管理 - YUM

YUM:

特点：

4. 可以同时配置多个资源库 (Repository)
5. 简洁的配置文件 (/etc/yum.conf)
6. 自动解决增加或删除 rpm 包时遇到的倚赖性问题

# 软件包管理 - YUM

YUM:

特点：

7. 使用方便

8. 保持与 RPM 数据库的一致性



# 软件包管理 -YUM

命令 :yum

功能 : 安装 / 查询 / 查找 / 删除软件包

语法格式 :yum [options] [command] [package ...]



# 软件包管理 -YUM

## 1. 安装软件包

`yum install` 全部安装

`yum install package1` 安装指定的安装包

`yum groups install group1` 安装指定程序组

`yum groups mark install group1` 标记指定的程序组，在安装时将自动安装丢失的程序包及关联文件

# 软件包管理 -YUM

## 1. 安装软件包

`yum install package1 --nogpgcheck` 忽略 gpg 检测





# 软件包管理 - YUM

## 2. 更新和升级

`yum update` 全部更新

`yum update package1` 更新指定程序包

`yum check-update` 检查可更新的程序

`yum upgrade package1` 升级指定程序包

`yum groups update group1` 升级指定程序组

# 软件包管理 - YUM

## 2. 更新和升级

`yum check-update` ← 检测可用更新信息

`yum update` <- 升级所有包括 kernel, 系统设置

`yum upgrade` <- 升级所有包括旧有的软件



# 软件包管理 -YUM

## 3. 查找和显示

`yum info package1` 显示指定安装包信息

`yum list` 显示所有已经安装和可以安装的程序包

`yum list package1` 显示指定程序包安装情况

`yum deplist packages` 显示软件包依赖关系

# 软件包管理 -YUM

## 3. 查找和显示

`yum groups info group1` 显示 指定程序组信息

`yum search string` 根据关键字 `string` 查找安装包

`yum provides / 路径 / 文件名` 查看文件属于哪个软件包

# 软件包管理 -YUM

## 3. 查找和显示

yum groups list 查看 group 列表

Yum groups info [ 软件组名 ] 查看组软件包列表信息



# 软件包管理 -YUM

## 4. 删除

`yum remove package1` 删除指定程序包

`yum groups remove group1` 删除程序组 group1

`yum clean all` 清除所有下载的 rpm 头文件及软件包

# 软件包管理 - YUM

## 6. 查看历史动作及恢复

`yum history` 查看 yum 历史 (与 `tail /var/log/yum.log` 一致)

`yum history undo N` (撤销历史操作, 恢复原有软件状态)



# 软件包管理 -YUM

## 5. 运算符 (yum groups 使用)

+: 除默认 groups 程序外, 安装 / 升级指定软件包

-: 指定的软件包不会安装 / 升级

=: 只安装 / 升级指定的软件包

no marker: 仅安装指定的组包, 无额外内容





# 软件包管理 -YUM

## 软件池管理

1. 列示本地所有软件池信息

```
#yum repolist all
```

2. 开启指定的软件池

```
#yum-config-manager --enable 软件池名
```

3. 关闭指定的软件池

```
#yum-config-manager --dsiable 软件池名
```

# 软件包管理 - YUM

## 软件池管理

### 4. 添加现有的 yum 源 (WEB/FTP)

```
#yum-config-manager --add-repo="yum url"
```

如

```
#yum-config-manager --add-repo=' "  
http://dl.snow.edu/pub/7/x86_64"'
```

### 5. 确认 rpm-gpg 秘钥路径

```
#ls /etc/pki/rpm-gpg
```



# 软件包管理 -YUM

客户端建立自己的 YUM 源

环境：以本地 ISO 作为 YUM 源

1) 挂载 ISO

```
#mount /dev/cdrom /mnt/cdrom -o loop
```

2) 建立 YUM 源

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

```
#vi my.repo
```



# 软件包管理 - YUM

3)my.repo 内容如下  
定义 YUM 源的名字  
[my-source]

对 YUM 源的注释  
name=abc

YUM 源位置 ( 可支持 FTP/HTTP 等 )  
baseurl=file:///mnt/cdrom



# 软件包管理 -YUM

3)my.repo 内容如下

是否启用此 YUM 源 .0 为不启用,1 为启用

enabled=1

gpg 密钥检测功能是否支持 .0 为否,1 为允许

gpgcheck=0

指定 gpg 密钥所在路径及名称

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-test

# 软件包管理 -YUM

## 4) 测试

```
#yum search xxx
```

```
#yum install xxx -y
```



# 软件包管理 - YUM

当有多个 yum 源时可以通过配置优先级来确定哪个源最优先考虑 (CentOS)

```
#yum install yum-plugin-priorities -y
```

在 yum 的源配置文件中加入  
priority=N [N=1-99, 数字越低优先级越高]

priorities 的插件安装在 /etc/yum/pluginconf.d  
中, 配置文件为 priorities.conf

# 软件包管理 -YUM

如设置了优先级，需要注意当优先级 1( 最高优先级 ) 存在软件且其他优先级也存在此软件，则仅以优先级 1 为准。如低优先级此软件有新版本，而高优先级没有新版本 ( 如使用 ISO 作为 repo) 则软件不会更新。





# 软件包管理 -RPM

RPM:

RedHat Package Manage(RedHat 包管理)

RPM 为二进制软件包，用户可以直接对软件进行安装。

# 软件包管理 -RPM

RPM 的组成一般由四个部分组成

1. 软件名
2. 软件版本号
3. 架构 [i686,x86\_64,noarch]
4. 后缀名 [rpm( 二进制包 )/src.rpm( 源代码包 )]

# 软件包管理 -RPM

## RPM 特点

1. 二进制文件不用对软件进行编译然后在安装
2. 对 src.rpm 包可以再次进行调整并生成二进制软件
3. 对软件的控制、查询、卸载非常方便
4. 软件包关联处理不智能



# 软件包管理 -RPM

命令 :rpm

功能：安装后缀为 RPM 软件

语法格式 :rpm <选项> <软件包名>



# 软件包管理 -RPM

选项：

-i: 安装软件包

-v: 安装时显示安装信息

-h: 以 “#” 作为进度条显示安装进度

-e: 删除软件包



# 软件包管理 -RPM

选项：

-U: 升级软件包，如果升级的软件包不存在，则转为安装

-F: 升级软件包，如果升级的软件包不存在，则放弃升级

--nodeps: 忽略软件包的关联包

--force: 强制安装，忽略错误 / 提示等信息

# 软件包管理 -RPM

选项：

-q: 查询，查询选项有许多副参数

1) -qa: 查询本地系统当前所有已安装的软件包

2) -qf: 查询某个文件 / 目录由哪个软件包安装所产生的

3) -ql: 查询某个软件包的安装路径

# 软件包管理 -RPM

选项：

-q: 查询，查询选项有许多副参数

4) -qi: 查询指定软件包的详细信息

5) --whatrequires: 查询指定软件包的关联包

6) --whatprovides: 查询某个模块属于哪个软件包




# 软件包管理 -RPM

选项：

-q: 查询，查询选项有许多副参数

7) -qR: 查询指定软件需要哪些模块才能正常工作

8)-qd: 仅查询帮助文档 

9)-qc: 查找配置文档



# 软件包管理 -RPM

选项：

-q: 查询，查询选项有许多副参数

10)-q --scripts: 查询安装 / 删除软件包的脚本

11)-q --changelog: 显示软件包更改的信息

-qlp: 显示指定软件包的信息💬

--root: 指定软件的安装目录



# 软件包管理 -RPM

选项：

-V: 验证已安装的软件选项使用时

如果使用 -V 时没有出现任何信息则代表其软件安装的所有文件均没有别修改或其他动作。如出现相关信息则代表某个方面被修改过

其信息代码表示



# 软件包管理 -RPM

选项：

-V 代表表示

5:MD5 checksum

S: 文件大小

L :Symbolic link ( 连接符号 )

T: 文件的修改时间

D: 设备

U: 文件属主

G: 文件属组

M: 文件的权限及类型



# 软件包管理 -RPM

示例：



1. 安装 abc.rpm 软件

```
#rpm -ivh abc.rpm
```

2. 查看当前系统已经安装的软件

```
#rpm -qa
```

3. 查看 /etc/passwd 文件属于哪个软件

```
#rpm -qf /etc/passwd
```



# 软件包管理 -RPM

示例：

4. 查看 /lib64/ld-2.17.so 属于哪个软件

```
#rpm -q --whatprovides /lib64/ld-2.17.so
```

或

```
#rpm -qf /lib64/ld-2.17.so
```

5. 查看 xyz.rpm 软件包信息

```
#rpm -qi xyz
```

6. 对 sudo 软件包进行校验

```
#rpm -V sudo
```



# 软件包管理 -RPM

示例：

7. 删除 abc 软件包

```
#rpm -e abc
```

8. 对 apache2.0 软件进行降级安装 apache1.0

```
#rpm -ivh apache1.0.rpm --force
```

9. RPM 另类安装

```
#rpm -ivh http://www.xyz.com/abc.rpm
```