第 5 章

Linux 系统配置

操作系统安装完成以后，为了提高工作效率及满足某些应用的安装和配置，需要对 Linux 系统进行配置。本章以 CentOS 系统进行举例，系统安装完成以后，可以安装一些基础的网络工具，方便后续的网络配置检测和使用，其他类型的 Linux 系统的原理和安装与此类似，命令如下：

```
yum install -y lsof net-tools wget procps psmisc iputils telnet curl tcpdump traceroute
```

5.1 Linux 网卡配置

网络配置对操作系统来讲是一个非常重要的配置，不管是下载软件还是访问服务都需要合理地配置好网络，一个系统可以根据网卡的情况配置多个 IP，每个 IP 配置不同的网段访问不同的服务。在云计算环境中，通常会有很多个网络，需要配置后进行访问，也需要进行网络隔离，所以掌握网络的基本概念和配置是非常有必要的。

1. 静态 IP 地址

进入 Linux 系统后，可以通过多种方式查看当前机器的网卡信息及 IP 地址，命令如下：

```
ip addr
ifconfig
```

查看不同网段 IP 的路由情况，命令如下：

```
route
#或者
route -n
```

进入网卡配置目录，命令如下：

```
/etc/sysconfig/network-scripts/
```

在网卡配置里，分为静态 IP 配置和动态 IP 配置。为什么要用静态 IP？静态 IP 最大的好处是，当系统重启或者断电关机后，如果业务系统恢复后，则还是原来的 IP 配置，不用动态地修改。这一点在云计算环境中很重要，云计算环境中对 IP 地址的配置基本上是固定的，前期规划好，不能轻易地修改，否则其中一个节点宕机后再恢复时如果 IP 变了，则会引起很多关联服务无法访问。动态 IP 配置应用得也很广泛，常用的有无线路由器，终端接入后，自动分配 IP 地址进行上网。在进行 IP 配置之前，有几个概念需要了解一下。

1）DNS

DNS 是域名系统（Domain Name System）的缩写，它是由解析器和域名服务器组成的。域名服务器是指保存着该网络中所有主机的域名和对应 IP 地址并具有将域名转换为 IP 地址功能的服务器，其中域名必须对应一个 IP 地址，而 IP 地址不一定有域名。域名系统采用类似目录树的等级结构。域名服务器为客户机/服务器模式中的服务器方，它主要有两种形式：主服务器和转发服务器。将域名映射为 IP 地址的过程称为“域名解析”。在 Internet 中域名与 IP 地址之间是一对一（或者多对一）的，域名虽然便于人们记忆，但机器之间只能互相认识 IP 地址，它们之间的转换工作称为域名解析，域名解析需要由专门的域名解析服务器来完成，DNS 就是进行域名解析的服务器。DNS 命名用于 Internet 等 TCP/IP 网络中，通过用户友好的名称查找计算机和服务。当用户在应用程序中输入 DNS 名称时，DNS 服务器可以将此名称解析为与之相关的其他信息，如 IP 地址。在上网时输入的网址是通过域名解析系统解析后找到了相对应的 IP 地址才能上网。其实，域名的最终指向是 IP。

2）网关

顾名思义，网关（Gateway）就是一个网络连接到另一个网络的“关口”。按照不同的分类标准，网关也有很多种。TCP/IP 里的网关是最常用的，在这里所讲的“网关”均指 TCP/IP 下的网关。那么网关到底是什么呢？网关实际上是一个网络通向其他网络的 IP 地址。例如有网络 A 和网络 B，网络 A 的 IP 地址范围为 192.168.1.1～192. 168.1.254，子网掩码为 255.255.255.0；网络 B 的 IP 地址范围为 192.168.2.1～192.168.2.254，子网掩码为 255.255.255.0。在没有路由器的情况下，两个网络之间是不能进行 TCP/IP 通信的，即使是两个网络连接在同一台交换机（或集线器）上，TCP/IP 也会根据子网掩码（255.255.255.0）判定两个网络中的主机处在不同的网络里，而要实现这两个网络之间的通信，则必须通过网关。如果网络 A 中的主机发现数据包的目的主机不在本地网络中，就把数据包转发给它自己的网关，再由网关转发给网络 B 的网关，网络 B 的网关再转发给网络 B 的某个主机。网络 B 向网络 A 转发数据包的过程也是如此。

所以说，只有设置好网关的 IP 地址，TCP/IP 才能实现不同网络之间的相互通信。那么这个 IP 地址是哪台机器的 IP 地址呢？网关的 IP 地址是具有路由功能的设备的 IP 地址，具有路由功能的设备有路由器、启用了路由协议的服务器（实际上相当于一台路由器）、代理服务器（也相当于一台路由器）。

什么是默认网关？如果搞清了什么是网关，默认网关也就好理解了。就好像一个房间

可以有多扇门一样，一台主机可以有多个网关。默认网关的意思是一台主机如果找不到可用的网关，就把数据包发给默认指定的网关，由这个网关来处理数据包。现在主机使用的网关，一般指的是默认网关。

如何设置默认网关？一台计算机的默认网关是不可以随便指定的，必须正确地指定，否则一台计算机就会将数据包发给不是网关的计算机，从而无法与其他网络的计算机通信。默认网关的设定有手动设置和自动设置两种方式。手动设置适用于计算机数量比较少、TCP/IP 参数基本不变的情况，例如只有几台到十几台计算机。因为这种方法需要在联入网络的每台计算机上设置“默认网关”，非常费时，一旦因为迁移等原因导致必须修改默认网关 IP 地址的情况，就会给网管带来很大的麻烦，所以不推荐使用。需要特别注意的是，默认网关必须是计算机自己所在的网段中的 IP 地址，而不能填写其他网段中的 IP 地址。自动设置就是利用 DHCP 服务器来自动给网络中的计算机分配 IP 地址、子网掩码和默认网关。这样做的好处是一旦网络的默认网关发生了变化，只要更改了 DHCP 服务器中默认网关的设置，那么网络中所有的计算机均获得了新的默认网关的 IP 地址。这种方法适用于网络规模较大，TCP/IP 参数有可能变动的网络。另外一种自动获得网关的办法是通过安装代理服务器软件(例如 MS Proxy)的客户端程序来自动获得，其原理和方法与 DHCP 有相似之处。

3）修改 IP

进入网卡配置目录后，可通过 ls 命令查看相关文件，使用编辑命令修改网卡配置，命令如下：

```
vim ifcfg-eth0
```

其中，ifcfg-eth0 根据实际生成的网卡名字选择，修改以下内容：

```
BOOTPROTO=static                    #将 dhcp 改为 static
ONBOOT=yes                          #开机启用本配置
IPADDR=192.168.7.106                #静态 IP
GATEWAY=192.168.7.1                 #默认网关
NETMASK=255.255.255.0               #子网掩码
DNS1=192.168.7.1                    #DNS 配置
```

修改后通过 cat 命令查看，命令如下：

```
cat ifcfg-eth0
```

输出的内容类似如下内容：

```
HWADDR=00:15:5D:07:F1:02
TYPE=Ethernet
BOOTPROTO=static                    #将 dhcp 改为 static
```

```
DEFROUTE = yes
PEERDNS = yes
PEERROUTES = yes
IPV4_FAILURE_FATAL = no
IPV6INIT = yes
IPV6_AUTOCONF = yes
IPV6_DEFROUTE = yes
IPV6_PEERDNS = yes
IPV6_PEERROUTES = yes
IPV6_FAILURE_FATAL = no
NAME = eth0
UUID = bb3a302d - dc46 - 461a - 881e - d46cafd0eb71
ONBOOT = yes                              #开机启用本配置
IPADDR = 192.168.7.106                    #静态 IP
GATEWAY = 192.168.7.1                     #默认网关
NETMASK = 255.255.255.0                   #子网掩码
DNS1 = 192.168.7.1                        #DNS 配置
```

最后，重启网络服务以便生效，命令如下：

```
systemctl restart network
```

添加 DNS，修改配置文件/etc/resolv.conf 中的内容，编辑该文件，命令如下：

```
vim /etc/resolv.conf
```

根据自己的需要添加或修改，内容与如下内容类似：

```
#访问公网 DNS
nameserver 8.8.8.8
nameserver 114.114..114.114
#访问内网 DNS
nameserver 10.85.152.99
```

这里分享一个笔者经历的有趣的小案例，在云计算环境集群搭建中，有很多台物理服务器，遇到了一个节点之间网络不通的情况，大家都以为是交换机 VLAN、防火墙、虚拟路由器、网关等的问题，进行抓包测试，花费了一两小时后才发现是网线接口松动了，网线没有被接好，重新插拔接好网线，系统恢复正常。这里要说明的是开发经验可以帮助快速定位，节省大量的时间和精力，如果以前遇到过类似的问题，可能就不会浪费一两小时的时间和精力了。

查看网卡对应的网线是否插拔好及是否连接正常，可以用 ethtool 命令进行查看，示例命令如下：

```
ethtool eth0
```

输出的内容类似如下的内容：

```
Settings for eth0:
        Supported ports: [ TP ]
        Supported link modes: 10baseT/Half 10baseT/Full
                              100baseT/Half 100baseT/Full
                              1000baseT/Half 1000baseT/Full
        Supports auto-negotiation: Yes
        Advertised link modes: 100baseT/Half 100baseT/Full
                               1000baseT/Half 1000baseT/Full
        Advertised auto-negotiation: Yes
        Speed: 1000Mb/s
        Duplex: Full
        Port: Twisted Pair
        PHYAD: 0
        Transceiver: external
        Auto-negotiation: off
        Link detected: yes
```

通过 Link detected：yes 可以判定网卡的网线连接正常。

2．多网卡配置

现在很多公司组建了规模不小的内部网络，在内部网络提供了很多服务，例如 OA 系统、财务系统、Git 代码仓库、Wiki 知识管理系统、Jira 项目管理系统等，将这些服务放在内网有很多优势，一是可以节省资源成本；二是可以减少一定的安全风险。这些服务可能会被划分为不同的网段分配 IP，如果同时想访问不同的网段的服务，则可以通过不同网卡配置不同网段的 IP 地址进行访问。配置步骤跟前面讲的步骤类似，只需根据实际的网段情况进行配置。

对于单网卡多 IP 的配置，在/etc/sysconfig/network-scripts/下，例如存在 ifcfg-enp0s3 文件，则说明 ifcfg-enp0s3 是原来网址的配置文件，只要将 ifcfg-enp0s3 复制到 ifcfg-enp0s3:0 中，修改里面的 IP 地址，就会在 enp0s3 对应的网卡下添加新 IP，以此类推，如果还要添加 IP，则可新添加配置文件 ifcfg-enp0s3:1。注意配置文件里面有个 UUID，这个是对网卡的唯一标识，如果是对不同的网卡添加 IP，则为了简便而复制已存在的配置文件时一定要注意应将里面的 hwaddr 字段修改为对应网卡的 MAC 地址，以及 UUID。查看 UUID 的命令为 nmcli con show，查看 MAC 地址的方法为 ip addr 或者 nmcli device show [interface]。有时查看某个网卡所对应的 UUID 时可能会提示有两个，出现这种情况的原因是此网卡所对应的配置文件里的 UUID 和系统分配的 UUID 不一致，将配置文件里的 UUID 改成系统分配的 UUID 后重启计算机，则此情况将会消失。

3．常用网络配置操作

关于使用 ip 命令的一些操作，命令格式如下：

```
ip [选项] 操作对象{link|addr|route...}
ip link show                              #显示网络接口信息
```

```
ip link set eth0 upi                                    #开启网卡
ip link set eth0 down                                   #关闭网卡
ip link set eth0 promisc on                             #开启网卡的混合模式
ip link set eth0 promisc offi                           #关闭网卡的混合模式
ip link set eth0 txqueuelen 1200                        #设置网卡队列长度
ip link set eth0 mtu 1400                               #设置网卡最大传输单元
ip addr show                                            #显示网卡 IP 信息
ip addr add 192.168.0.1/24 dev eth0                     #将 eth0 网卡的 IP 地址设置为 192.168.0.1
ip addr del 192.168.0.1/24 dev eth0                     #删除 eth0 网卡的 IP 地址
ip route list                                           #查看路由信息
ip route add 192.168.4.0/24 via 192.168.0.254 dev eth0      #将 192.168.4.0 网段的网关设置
#为 192.168.0.254,数据通过 eth0 接口传输
ip route add default via 192.168.0.254 dev eth0         #将默认网关设置为 192.168.0.254
ip route del 192.168.4.0/24                             #删除 192.168.4.0 网段的网关
ip route del default                                    #删除默认路由
```

查看经过的 IP 地址,命令如下:

```
traceroute -n baidu.com
```

找出 IP 对应的主机名称,命令如下:

```
nslookup 192.168.2.1
```

telnet 用来监测对应端口的服务状况,类似的命令还有 bye、netstat、netcat(nc),命令如下:

```
telnet 192.168.2.55 3306
```

使用 tcpdump 命令进行数据包捕获,类似的命令还有 ethereal,捕捉 eth0 网卡的数据包,命令如下:

```
tcpdump -i eth0 -nn
```

只捕捉 21 号端口的数据包,命令如下:

```
tcpdump -i eth0 -nn port 21
```

ifconfig、ifup、ifdown 这 3 个命令都用于启动网络接口,不过,ifup 与 ifdown 仅可对/etc/sysconfig/network-scripts 内的 ifcfg-ethx(x 为数字)进行启动或关闭操作,并不能直接修改网络参数,除非手动调整 ifcfg-ethx 文件才行。至于 ifconfig 则可以直接手动给予某个接口 IP 或调整其网络参数。

ifconfig 主要用于手动启动、观察与修改网络接口的相关参数，可以修改的参数很多，包括 IP 参数及 MTU 等都可以修改，它的语法如下：

```
ifconfig {interface} {up|down}      <== 观察与启动接口
ifconfig interface {options}        <== 设置与修改接口
```

一般来讲，直接输入 ifconfig 命令就会列出目前已经被启动的网卡，不论这个网卡是否设置了 IP 都会被显示出来，而如果是输入 ifconfig eth0，则会显示出这个接口的相关数据，而不管该接口是否启动，所以如果想要知道某个网卡的 Hardware Address，则可直接输入“ifconfig 网络接口名称”查看。在上述代码中出现的各项数据的意义如下。

eth0：网卡的代号，也有 lo 这个 loopback。

HWaddr：网卡的硬件地址，习惯性地称为 MAC。

inet addr：IPv4 的 IP 地址，后续的 Bcase、Mask 分别代表的是 Broadcast 与 Netmask。

inet6 addr：IPv6 版本的 IP，很少使用，可以根据业务的实际需要配置。

RX：那一行代表的是网络由启动到目前为止的数据包接收情况，packets 代表数据包数、errors 代表数据包发生错误的数量、dropped 代表数据包由于有问题而遭丢弃的数量等。

TX：与 RX 相反，为网络由启动到目前为止的传送情况。

collisions：代表数据包碰撞的情况，如果发生了很多次，则表示网络状况不太好。

txqueuelen：代表用来传输数据的缓冲区的储存长度。

RX Bytes、TX Bytes：总传送、接收的字节总量。

Interrupt、Memory：网卡硬件的数据，IRQ 岔断与内存地址。

设置网络接口，同时设置 MTU 的数值，命令如下：

```
ifconfig eth0 192.168.100.100 netmask 255.255.255.128  mtu 8000
```

在该网络接口上再仿真一个网络接口，即在一个网卡上设置多个 IP，命令如下：

```
ifconfig eth0:0 192.168.50.50
```

关掉 eth0:0 这个接口。如果启动 eth1，并且不设置任何网络参数，则命令如下：

```
ifconfig eth0:0 down
```

将 eth0 设置成混杂模式以嗅探(sniffing)数据包，命令如下：

```
ifconfig eth0 promisc
```

以 dhcp 模式启用 eth0，命令如下：

```
dhclient eth0
```

如果发现了以 dhcp 模式启动的网卡，并且此网卡没有被分配到 IP 地址，则可以手动执行上面的命令获取 IP 来进一步排查问题，这个问题在云计算环境开发的 Ubuntu 系统镜像中出现过，可以结合 tcpdump 抓包进一步排查出现问题的原因。

要启动某个网络接口，但又不让它具有 IP 参数时，则可直接执行 ifconfig eth0 up 命令。这个操作经常在无线网卡中进行，因为需要启动无线网卡去检测 AP 是否存在。实时地手动修改一些网络接口参数，可以利用 ifconfig 实现，如果要直接配置文件，即在/etc/sysconfig/network-scripts 里面的 ifcfg-ethx 等文件中设置参数后启动，就要通过 ifup 或 ifdown 实现了，用法如下：

```
ifup  {interface}
ifdown {interface}
```

例如启动 eth0 网卡，命令如下：

```
ifup eth0
```

禁用一个 eth0 网络设备，命令如下：

```
ifdown eth0
```

ifup 与 ifdown 比较简单，这两个程序其实是脚本而已，它会直接到/etc/ sysconfig/network-scripts 目录下搜索对应的配置文件，例如 ifup eth0 命令，它会找出 ifcfg-eth0 文件的内容，然后加以配置运行。

5.2 Linux 环境配置

系统安装完成后，一般情况下需要进行一些必要的配置，方便应用的部署和机器管理，尤其对于大规模机器进行管理和配置，统一的规则和配置非常有必要，例如大公司里面对系统的 hostname 都需要按照一定的规则进行配置和管理。下面从几个常用的方面进行说明，这些配置在云计算环境搭建开始前都需要进行配置，若想节省精力则需要提前规划好，配置好这些选项后再开始业务的部署。

1. hostname 配置

在 Linux 系统中标示一个主机的名字，通常设置为永久的，命令如下：

```
hostnamectl set-hostname --static server-01
```

如果有多台 Linux 系统在同一个网络里组建集群或者同一类型的服务，则可以配置 hosts 文件通过主机名访问，查看文件命令如下：

```
cat /etc/hosts
```

输出的类似的内容如下：

```
172.20.xx.31 server-31
172.20.xx.32 server-32
172.20.xx.33 server-33
```

2. ssh 配置

不管是物理服务器 Linux 还是虚拟机创建的服务器 Linux，配置 ssh 远程链接都是一个不错的选择，可以实现在主机上进行命令粘贴、文件上传/下载、远程操作等，编辑 ssh 配置文件，命令如下：

```
vim /etc/ssh/sshd_config
```

需要修改的地方如下：

```
#允许 root 认证登录
PermitRootLogin yes

#允许密钥认证
#RSAAuthentication(rsa 认证)是只支持第 1 代 ssh 通信协议所使用的配置项,在
#CentOS 7.4 中被废除了
#而且前面提到过 CentOS 7 开始预设使用第 2 代通信协议,在 CentOS 7.4 中没有找到指定协议版
#本的配置行
RSAAuthentication yes

#第 2 代 ssh 通信协议的密钥验证选项是
PubkeyAuthentication yes

#默认公钥存放的位置
AuthorizedKeysFile .ssh/authorized_keys

#可使用密码进行 ssh 登录
PasswordAuthentication yes
```

3. firewall 配置

有些情况下 firewall 处于关闭状态，这是因为由 iptables 进行端口服务拦截访问。在进行云计算环境搭建时，也不需要用 firewall，而选用 iptables 进行端口拦截访问及端口服务重定向。如果没有使用 iptables，而是使用了 firewall 进行防火墙拦截，则可以通过以下操作进行端口拦截和访问。

查看 firewall 的服务状态，命令如下：

```
systemctl status firewalld
```

可以对 firewall 进行开启、重启、关闭操作，命令如下：

```
#开启
service firewalld start
#重启
service firewalld restart
#关闭
service firewalld stop
```

查看 firewall 的状态，命令如下：

```
firewall-cmd --state
```

查看防火墙规则，命令如下：

```
firewall-cmd --list-all
```

查询、开放、关闭端口，命令如下：

```
#查询端口是否开放
firewall-cmd --query-port=8080/tcp
#开放 80 端口
firewall-cmd --permanent --add-port=80/tcp
#移除端口
firewall-cmd --permanent --remove-port=8080/tcp

#重启防火墙(修改配置后要重启防火墙)
firewall-cmd --reload
```

以上参数的解释和含义如下。

firwall-cmd：由 Linux 提供的操作 firewall 的一个工具。

--permanent：表示设置为持久。

--add-port：标识添加的端口。

4. iptables 配置

因为云计算环境大量用到 iptables，这里进行一下详细的介绍。iptables 其实是 Linux 下的一个开源的信息过滤程序，包括地址转换和信息重定向等功能，它是由四表五链组成的，信息过滤功能十分强大，而所谓的硬件防火墙其实是由一个 Linux 核心加页面操作程序实现的，可以用于添加、编辑和移除规则。

如果主机上有一块网卡，当用户请求到达时，首先会到达硬件设备，能够在硬件上接收数据并且能够做后续处理的只有内核，而内核中用于网络属性管理的是 TCP/IP 协议栈，其实就是 TCP/IP 模块。一个报文由网卡到达本主机的 TCP/IP 协议栈后就要判断它的目标 IP 了，所以在 TCP/IP 协议栈有一个路由表，此路由表是由本机的路由模块实现的，如果它发现目标 IP 就是本机的 IP 地址，就会继续检查它的目标端口，如果目标端口被本机上的某个用户进程注册使用了，这个进程就监听在这个套接字上，所以检查到的这个端口的确是主机的某处进程在监听，这个报文就会通过这个套接字由用户空间转发给对应的进程，所以这个报文就到达本机的内部了。

如果路由检查机制发现目标 IP 不是本机的 IP 地址，就要检查路由是否允许进行网络间的转发，如果允许向外转发，那它就不会进入用户空间，在内核中直接交给另一块网卡（假如主机上有两块网卡）或另一个 IP 转发出去，相当于在内核中走一圈又出去了。

什么叫内部转发？如果主机上有两块网卡，或者说一块网卡有两个地址，一个地址可以接收请求，另一个地址可以把请求发出去，这就是网络间的转发机制，然而常说的防火墙其实是由 iptables 和 netfilter 两部分组成的，iptables 负责在 netfilter 上写规则，而 netfilter 相当于事先设好的卡哨，5 个卡哨组合起来叫作 netfilter，而 iptables 只负责在卡哨上填充规则，规则由真正检查者制定，卡哨可以将通路都挡掉，而在内核中这些卡哨被称为钩子函数，所以规则才是真正起防护作用的机制，而 netfilter 只是让这些规则得以生效。对 Linux 来讲，这些卡哨都有各自的名字，每个钩子函数都有自己的名字，根据这些报文的流向定义名称，而这些名称在 iptables 上被称为链，iptables 上的链有 5 条，这就是常说的 5 个钩子函数。

（1）INPUT：从本机进来的。

（2）OUTPUT：从本机出去的。

（3）FORWARD：从本机转发的，本机内部出去了，无论如何也不会经过 FORWARD。

（4）POSTROUTING：路由之后。

（5）PREROUTING：路由之前。

数据包控制方式包括以下 6 种行为。

（1）ACCEPT：允许数据包通过。

（2）DROP：直接丢弃数据包，不给出任何回应信息。

（3）REJECT：拒绝数据包通过，必要时会给数据发送端一个响应信息。

（4）LOG：在/var/log/messages 文件中记录日志信息，然后将数据包传递给下一条规则。

（5）QUEUE：防火墙将数据包移交到用户空间。

（6）RETURN：防火墙停止执行当前链中的后续 Rules，并返回调用链（the calling chain）。

iptables 其实不是真正的防火墙，可以把它理解成一个客户端代理，用户通过 iptables 这个代理，将用户的安全设定执行到对应的“安全框架”中，这个“安全框架”才是真正的防火

墙，这个框架的名字叫 netfilter。netfilter 才是防火墙真正的安全框架（Framework），netfilter 位于内核空间。iptables 其实是一个命令行工具，位于用户空间，用这个工具操作真正的框架，所以说，虽然使用 service iptables start 启动 iptables 服务，但是其实准确地讲，iptables 并没有一个守护进程，所以并不能算是真正意义上的服务，而应该算是内核提供的功能。

iptables 的结构：iptables→Tables→Chains→Rules。简单地讲，Tables 由 Chains 组成，而 Chains 又由 Rules 组成。Tables 包括 Filter、NAT、Mangle、Raw 共 4 种表。Chains 包括 Input、Output、Forward、Prerouting、Postrouting 共 5 种链。Rules 用于指定所检查包的特征和目标。如果包不匹配，则将送往该链中下一条规则检查；如果匹配，则下一条规则由目标值确定。该目标值可以是用户定义的链名，或是某个专用值，如 ACCEPT[通过]、DROP[删除]、QUEUE[排队]或者 RETURN[返回]。

filter 表主要用于对数据包进行过滤，根据具体的规则决定是否放行该数据包，如 DROP、ACCEPT、REJECT、LOG。filter 表对应的内核模块为 iptable_filter，包含以下 3 个规则链。

（1）INPUT 链：INPUT 针对那些目的地是本地的包。

（2）FORWARD 链：FORWARD 用于过滤所有不是本地产生的并且目的地不是本地（本机只是负责转发）的包。

（3）OUTPUT 链：OUTPUT 用来过滤所有本地生成的包。

nat 表主要用于修改数据包的 IP 地址、端口号等信息（网络地址转换，如 SNAT、DNAT、MASQUERADE、REDIRECT）。属于一个流的包（因为包的大小限制导致数据可能会被分成多个数据包）只会经过这个表一次。如果第 1 个包被允许做 NAT 或 Masqueraded，则余下的包都会自动地被执行相同的操作，也就是说，余下的包不会再通过这个表。表对应的内核模块为 iptable_nat，包含以下 3 个链。

（1）PREROUTING 链：在包刚刚到达防火墙时改变它的目的地址。

（2）OUTPUT 链：改变本地产生的包的目的地址。

（3）POSTROUTING 链：在包就要离开防火墙之前改变其源地址。

mangle 表主要用于修改数据包的 ToS（Type of Service，服务类型）、TTL（Time To Live，生存周期）及为数据包设置 Mark 标记，以实现 QoS（Quality of Service，服务质量）调整及策略路由等应用，由于需要相应的路由设备支持，因此应用并不广泛。其包含了 5 个规则链，即 PREROUTING、POSTROUTING、INPUT、OUTPUT、FORWARD。

raw 表是自 1.2.9 以后版本的 iptables 新增的表，主要用于决定数据包是否被状态跟踪机制处理。在匹配数据包时，raw 表的规则要优先于其他表，包含两条规则链，即 OUTPUT、PREROUTING。

iptables 中数据包和被跟踪连接的 4 种不同状态如下。

（1）NEW：该包想要开始一个连接（重新连接或将连接重定向）。

（2）RELATED：该包属于某个已经建立的连接所建立的新连接。例如 FTP 的数据传

输连接就是控制连接所关联出来的连接。-icmp-type 0（ping 应答）就是-icmp-type 8（ping 请求）所关联出来的。

（3）ESTABLISHED：只要发送并接到应答，一个数据连接从 NEW 变为 ESTABLISHED，而且该状态会继续匹配这个连接的后续数据包。

（4）INVALID：数据包不能被识别属于哪个连接或没有任何状态，例如内存溢出，收到不知属于哪个连接的 ICMP 错误信息，一般应该丢弃这种状态的任何数据。

防火墙处理数据包的方式（规则）有以下几种。

（1）ACCEPT：允许数据包通过。

（2）DROP：直接丢弃数据包，不给任何回应信息。

（3）REJECT：拒绝数据包通过，必要时会给数据发送端一个响应的信息。

（4）SNAT：源地址转换。在进入路由层面的 route 后，出本地的网络栈前，改写源地址，目标地址不变，并在本机建立 NAT 表项，当数据返回时，根据 NAT 表将目的地址数据改写为数据发送出去时的源地址，并发送给主机。解决内网用户用同一个公网地址上网的问题。

（5）MASQUERADE：是 SNAT 的一种特殊形式，适用于像 ADSL 这种临时会变的 IP 上。

（6）DNAT：目标地址转换。和 SNAT 相反，IP 包经过 route 前，DNAT 会重新修改目标地址，源地址不变，在本机建立 NAT 表项，当数据返回时，根据 NAT 表将源地址修改为数据发送过来时的目标地址，并发给远程主机。可以隐藏后端服务器的真实地址。

（7）REDIRECT：是 DNAT 的一种特殊形式，将网络包转发到本地 host 上（不管 IP 头部指定的目标地址是什么），方便在本机进行端口转发。

（8）LOG：在/var/log/messages 文件中记录日志信息，然后将数据包传递给下一条规则。

除去最后一个 LOG，前 3 条规则匹配数据包后，该数据包不会再往下继续匹配了，所以编写的规则顺序极其关键。

iptables 编写规则，命令格式如下。

（1）[-t 表名]：该规则所操作的哪个表可以使用 filter、nat 等，如果没有指定，则默认为 filter。

（1）-A：将一条规则新增到该规则链列表的最后一行。

（2）-I：插入一条规则，原本该位置上的规则会往后按顺序移动，如果没有指定编号，则为 1。

（3）-D：从规则链中删除一条规则，要么输入完整的规则，要么指定规则编号加以删除。

（4）-R：替换某条规则，规则替换不会改变顺序，而且必须指定编号。

（5）-P：设置某条规则链的默认动作。

（6）-nL：-L、-n，查看当前运行的防火墙规则列表。

（7）chain 名：指定规则表的哪个链，如 INPUT、OUPUT、FORWARD、PREROUTING 等。

(8)[规则编号]：插入、删除、替换规则时用-line-numbers 显示号码。

(9)[-i|o 网卡名称]：i 用于指定数据包从哪块网卡进入，o 用于指定数据包从哪块网卡输出。

(10)[-p 协议类型]：可以指定规则应用的协议，包含 TCP、UDP 和 ICMP 等。

(11)[-s 源 IP 地址]：源主机的 IP 地址或子网地址。

(12)[--sport 源端口号]：数据包的 IP 的源端口号。

(13)--line-number：查看规则列表时，同时显示规则在链中的顺序号。

(14)[-d 目标 IP 地址]：目标主机的 IP 地址或子网地址。

(15)[--dport 目标端口号]：数据包的 IP 的目标端口号。

(16)-m：extend matches，这个选项用于提供更多的匹配参数，代码如下：

```
-m state - state ESTABLISHED,RELATED
-m tcp - dport 22
-m multiport - dports 80,8080
-m icmp - icmp-type 8
```

(17)<-j 动作>：处理数据包的动作，包括 ACCEPT、DROP、REJECT 等。

iptables 的基本操作有以下几种。

启动 iptables，命令如下：

```
service iptables start
```

关闭 iptables，命令如下：

```
service iptables stop
```

重启 iptables，命令如下：

```
service iptables restart
```

查看 iptables 的状态，命令如下：

```
service iptables status
```

保存 iptables 的配置，命令如下：

```
service iptables save
```

查看 iptables 的服务配置文件，命令如下：

```
cat /etc/sysconfig/iptables-config
```

查看 iptables 的规则保存文件，命令如下：

```
cat /etc/sysconfig/iptables
```

打开 iptables 转发，命令如下：

```
echo "1"> /proc/sys/net/ipv4/ip_forward
```

iptables 应用非常广泛，涉及很多个与网络相关的领域，这里列举几个开发中常用的一些小例子。

【例 5-1】 后端服务器迁移，前端或移动端固定的 IP，故需兼容两个 IP。Shell 脚本命令如下：

```
#!/bin/bash

OLD_IP=A1.A2.A3.A4
NEW_IP=B1.B2.B3.B4
OLD_PORT=12345
NEW_PORT=12345
#允许 IP 进行转发
echo 1 > /proc/sys/net/ipv4/ip_forward

#将旧的 IP 端口的流量重定向到新的 IP 端口
iptables -t nat -A PREROUTING -p tcp --dport $OLD_PORT -j DNAT --to-destination $NEW
_IP:$NEW_PORT
iptables -t nat -A POSTROUTING -p tcp -d $NEW_IP --dport $NEW_PORT -j SNAT --to-
source $OLD_IP


#最后修改相应的 IP (MASQUERADE)
iptables -t nat -A POSTROUTING -j MASQUERADE


service iptables stop

service iptables save

service iptables restart
```

查询 NAT 规则表，命令如下：

```
#PRETOURING 链下 NAT 表，默认查询 FILTER 表
iptables -t nat -L PREROUTING
```

规则链删除，命令如下：

```
#查询 NAT 规则表
iptables -t nat -L -n --line-numbers
#NAT 规则链删除
#1 表示行号
iptables -t nat -D PREROUTING 1
iptables -t nat -D POSTROUTING 1
```

【例 5-2】 将请求进来的 443 端口映射到 3306 端口。

使用 iptables 命令进行端口重定向，命令如下：

```
iptables -t nat -A PREROUTING -s 10.28.80.11 -p tcp --dport 443 -i eth0 -j REDIRECT --
to 3306

iptables -t nat -nvL
```

使用 tcpdump 命令对指定 IP 的 443 端口进行抓包，命令如下：

```
tcpdump -vv host 10.28.80.11 and tcp port 443
```

常用的 iptables 操作可应用于很多种场景，例如，对现有 iptables 规则进行修改调整。

(1) 删除 iptables 的现有规则，命令如下：

```
iptables -F
```

(2) 查看 iptables 的规则，命令如下：

```
iptables -L(iptables -L -v -n)
```

(3) 将一条规则增加到最后，命令如下：

```
iptables -A INPUT -i eth0 -p tcp --dport 80 -m state --state
NEW,ESTABLISHED -j ACCEPT
```

(4) 将一条规则添加到指定位置，命令如下：

```
iptables -I INPUT 2 -i eth0 -p tcp --dport 80 -m state --state
NEW,ESTABLISHED -j ACCEPT
```

(5) 删除一条规则，命令如下：

```
iptabels -D INPUT 2
```

(6) 修改一条规则,命令如下:

```
iptables -R INPUT 3 -i eth0 -p tcp --dport 80 -m state --state
NEW,ESTABLISHED -j ACCEPT
```

(7) 设置默认策略,命令如下:

```
iptables -P INPUT DROP
```

(8) 允许远程主机进行 SSH 连接,命令如下:

```
iptables -A INPUT -i eth0 -p tcp --dport 22 -m state --state
NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT
```

(9) 允许本地主机进行 SSH 连接,命令如下:

```
iptables -A OUTPUT -o eth0 -p tcp --dport 22 -m state --state
NEW,ESTABLISHED -j ACCEPT
iptables -A INTPUT -i eth0 -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT
```

(10) 允许 HTTP 请求,命令如下:

```
iptables -A INPUT -i eth0 -p tcp --dport 80 -m state --state
NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT
```

(11) 限制 ping 192.168.146.3 主机的数据包数,平均 2 个每秒,最多不能超过 3 个每秒,命令如下:

```
iptables -A INPUT -i eth0 -d 192.168.146.3 -p icmp --icmp-type 8 -m limit --limit 2/
second --limit-burst 3 -j ACCEPT
```

(12) 限制 SSH 连接速率(默认策略是 DROP),命令如下:

```
iptables -I INPUT 1 -p tcp --dport 22 -d 192.168.146.3 -m state --state ESTABLISHED -j ACCEPT
iptables -I INPUT 2 -p tcp --dport 22 -d 192.168.146.3 -m limit --limit 2/minute --
limit-burst 2 -m state --state NEW -j ACCEPT
```

有时候需要清除所有规则,重新按需要配置 iptables,下面对常用的一些操作进行举例。

(1) 删除现有规则,命令如下:

```
iptables -F
```

(2) 配置默认链策略,命令如下:

```
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
```

(3) 允许远程主机进行 SSH 连接,命令如下:

```
iptables -A INPUT -i eth0 -p tcp - dport 22 -m state - state
NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp - sport 22 -m state - state ESTABLISHED -j ACCEPT
```

(4) 允许本地主机进行 SSH 连接,命令如下:

```
iptables -A OUTPUT -o eth0 -p tcp - dport 22 -m state - state
NEW,ESTABLISHED -j ACCEPT
iptables -A INPUT -i eth0 -p tcp - sport 22 -m state - state ESTABLISHED -j ACCEPT
```

(5) 允许 HTTP 请求,命令如下:

```
iptables -A INPUT -i eth0 -p tcp - dport 80 -m state - state
NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp - sport 80 -m state - state ESTABLISHED -j ACCEPT
```

使用 iptables 抵抗常见攻击,这是非常有用的操作方式。

(1) 防止 syn 攻击。

方法一,限制 syn 的请求速度(这种方式需要调节一个合理的速度,否则会影响正常用户的请求),命令如下:

```
iptables -N syn-flood

iptables -A INPUT -p tcp --syn -j syn-flood

iptables -A syn-flood -m limit --limit 1/s --limit-burst 4 -j RETURN

iptables -A syn-flood -j DROP
```

方法二,限制单个 IP 的最大 syn 连接数,命令如下:

```
iptables - A INPUT - i eth0 - p tcp --syn -m connlimit --connlimit-above 15 -j DROP
```

(2) 防止 DOS 攻击。

利用 recent 模块抵御 DOS 攻击,命令如下:

```
iptables -I INPUT -p tcp -dport 22 -m connlimit --connlimit-above 3 -j DROP
```

单个 IP 最多连接 3 个会话,命令如下:

```
iptables -I INPUT -p tcp --dport 22 -m state --state NEW -m recent --set --name SSH
```

只要是新的连接请求,就把它加入 SSH 列表中,命令如下:

```
iptables -I INPUT -p tcp --dport 22 -m state NEW -m recent --update --
seconds 300 --hitcount 3 --name SSH -j DROP
```

如果 5min 内尝试次数达到 3 次,则拒绝提供 SSH 列表中的这个 IP 服务。被限制 5min 后即可恢复访问。

(3) 防止单个 IP 访问量过大,命令如下:

```
iptables -I INPUT -p tcp --dport 80 -m connlimit --connlimit-above 30 -j DROP
```

(4) 木马反弹,命令如下:

```
iptables - A OUTPUT - m state --state NEW - j DROP
```

(5) 防止 ping 攻击,命令如下:

```
iptables -A INPUT -p icmp --icmp-type echo-request -m limit --limit 1/m -j ACCEPT
```

(6) 批量转发配置,如将 10.10.10.1 的 1000-10000 端口的流量映射到 20.20.20.1 的 1000-10000 端口上,命令如下:

```
iptables - t nat - A PREROUTING - p tcp - m tcp -- dport 1000:10000 - j DNAT -- to -
destination 20.20.20.1:1000-10000
iptables - t nat - A PREROUTING - p udp - m udp -- dport 1000:10000 - j DNAT -- to -
destination 20.20.20.1:1000-10000
iptables -t nat -A POSTROUTING -d 20.20.20.1 -p tcp -m tcp --dport 1000:10000 -j SNAT
--to-source 10.10.10.1
iptables -t nat -A POSTROUTING -d 20.20.20.1 -p udp -m udp --dport 1000:10000 -j SNAT
--to-source 10.10.10.1

#保存配置
iptables-save
#查看配置
iptables -t nat -L
```

为了防止在 FORWARD 上被丢弃，添加规则允许通过，命令如下：

```
iptables -I FORWARD -d 220.181.111.188 -p tcp --dport 80 -j ACCEPT
iptables -I FORWARD -s 220.181.111.188 -p tcp --sport 80 -j ACCEPT
```

配置静态路由，使流量通过局域网传输，命令如下：

```
server1:ip route add 20.20.20.1/32 via 192.168.1.2
server2:ip route add 10.10.10.1/32 via 192.168.1.1
```

5.3 Linux 镜像换源

1. 系统镜像源配置

因为在线安装需要在服务器上下载所需软件和依赖关系文件，所以下载的速度很影响使用体验。Linux 默认的源安装和更新速度很慢，所以安装好系统后一般会选择换源。一般情况下，CentOS 7 的更新源文件放置在 /etc/yum.repos.d 目录下，这个目录下有多个以“.repo”为后缀的更新源文件，在更新软件时，最常用的是其中的 centos-Base.repo 文件，为了保证在国内网络环境下能够较快地完成更新，操作系统安装完成以后，可以对系统的 yum 源进行更换，以此提升软件安装的速度。常用的有清华源、阿里源等，下面以 CentOS 7 为例列出几个国内常用的源。

阿里源的网址如下：

```
http://mirrors.aliyun.com/repo/centos-7.repo
```

清华源的网址如下：

```
https://mirror.tuna.tsinghua.edu.cn/help/centos/
```

腾讯源的网址如下：

```
http://mirrors.cloud.tencent.com/repo/centos7_base.repo
```

网易源的网址如下：

```
http://mirrors.163.com/.help/centos7-Base-163.repo
```

以上这些源的网址都可以在浏览器打开查看，它们其实是一个文本，只是文本内容所包含的不同 URL 网址分别指向不同的服务器，以更换阿里源为例，更换的命令如下：

```
mv /etc/yum.repos.d/centos-Base.repo /etc/yum.repos.d/centos-Base.repo.backup

wget -O /etc/yum.repos.d/centos-Base.repo\
http://mirrors.aliyun.com/repo/centos-7.repo
```

推荐添加 EPEL 源，因为很多软件包和工具在 Base 源里面没有，但添加 EPEL 源后就可以安装了，添加命令如下：

```
wget -O /etc/yum.repos.d/epel.repo\
http://mirrors.aliyun.com/repo/epel-7.repo
```

更换或者完成添加源操作后，清理缓存并生成新的缓存，命令如下：

```
yum clean all
yum makecache
```

对于不同的操作系统类型，其更换方式可能有些小区别，但是原理是一样的，对于同一种操作系统的不同版本也有多种实现方式。

CentOS 5 的命令如下：

```
wget -O /etc/yum.repos.d/centos-Base.repo\
http://mirrors.aliyun.com/repo/centos-5.repo
#或者
curl -o /etc/yum.repos.d/centos-Base.repo\
http://mirrors.aliyun.com/repo/centos-5.repo
```

CentOS 6 的命令如下：

```
wget -O /etc/yum.repos.d/centos-Base.repo\
http://mirrors.aliyun.com/repo/centos-6.repo
#或者
curl -o /etc/yum.repos.d/centos-Base.repo\
http://mirrors.aliyun.com/repo/centos-6.repo
```

CentOS 7 的命令如下：

```
wget -O /etc/yum.repos.d/centos-Base.repo\
http://mirrors.aliyun.com/repo/centos-7.repo
#或者
curl -o /etc/yum.repos.d/centos-Base.repo\
http://mirrors.aliyun.com/repo/centos-7.repo
```

CentOS 8 的命令如下：

```
wget -O /etc/yum.repos.d/centos-Base.repo\
http://mirrors.cloud.tencent.com/repo/centos8_base.repo
#或者
curl -o /etc/yum.repos.d/centos-Base.repo\
http://mirrors.cloud.tencent.com/repo/centos8_base.repo
```

2. 软件镜像源配置

在 Linux 系统中，安装或者搭建应用通常用到的是 pip 源，因为使用 C 或者 C++语言编写的应用编译打包是基于本地的 C 库执行 make && make install 命令来完成的；基于 Go 或者 Java 语言的应用或者软件在开发完成后都会编译出二进制 bin 文件或者 jar 包文件，不需要到 Linux 系统服务器上编译，但很多基于 Python 开发的开源项目和应用需要到 Linux 系统服务器上编译后执行安装，这应该也是绝大多数 Linux 系统自带 Python 安装的一个原因。在云计算环境搭建过程中，绝大多数项目是基于 Python 开发的，因此更换一个速度更快的 pip 源对提升工作效率非常有帮助。

与系统镜像源类似，也有很多 pip 源可以选择，常用的有清华源、阿里源等，下面列出几个国内常用的源。

阿里云的网址如下：

```
https://mirrors.aliyun.com/pypi/simple/
```

豆瓣(douban)的网址如下：

```
https://pypi.douban.com/simple/
```

清华大学的网址如下：

```
https://pypi.tuna.tsinghua.edu.cn/simple/
```

中国科学技术大学的网址如下：

```
https://pypi.mirrors.ustc.edu.cn/simple/
```

163pip 源地址的网址如下：

```
https://mirrors.163.com/pypi/
```

当需要使用某个 pip 源临时安装一个包时，可以在使用 pip 时加参数-i 进行安装，命令如下：

```
pip install SomePackage -i https://pypi.tuna.tsinghua.edu.cn/simple
```

这样就会从清华大学的镜像去安装 SomePackage 库。

如果想永久地配置一个 pip 源进行使用，则可以使用命令进行配置，命令如下：

```
vim ~/.pip/pip.conf
```

以配置豆瓣源为例(其他源只需替换对应的域名)，内容如下：

```
[global]
timeout = 6000
index-url = https://pypi.douban.com/simple/
[install]
trusted-host = pypi.douban.com
```

以后在其他地方使用 pip 进行包下载时，默认都会使用豆瓣源下载，比官方的源要快很多，也不容易因出现超时而失败的情况，极大地提高了工作效率。

5.4 Linux 系统与应用更新

1. 内核更新

新版本的 Linux 内核可能会支持很多新的特性和功能，尤其是某些新一点的应用软件，对 Linux 的内核有依赖性，所以有时为了安装应用软件也需要更新 Linux 内核。另外，云环境的搭建需要 Linux 内核 3.2 以上，如果不想重装系统，则可以采用内核升级的方式解决，一般情况下，CentOS 7.5 及以上的版本可以满足需求。

如果需要对 Linux 内核进行升级，这里以 CentOS 系统举例，则执行以下步骤即可完成。

(1) 查看当前内核版本，命令如下：

```
uname -r
#或者
uname -a
#或者
cat /etc/redhat-release
```

(2) 升级内核，通过命令进行安装。

更新 yum 源仓库，命令如下：

```
yum -y update
```

启用 ELRepo 仓库，ELRepo 仓库是基于社区的企业级 Linux 仓库，提供对 Red Hat

Enterprise(RHEL)和其他基于 RHEL 的 Linux 发行版(CentOS、Scientific、Fedora 等)的支持。ELRepo 聚焦于和硬件相关的软件包,包括文件系统驱动、显卡驱动、网络驱动、声卡驱动和摄像头驱动等。

导入 ELRepo 仓库的公共密钥,命令如下:

```
rpm --import https://www.elrepo.org/RPM-GPG-KEY-elrepo.org
```

安装 ELRepo 仓库的 yum 源,命令如下:

```
rpm -Uvh http://www.elrepo.org/elrepo-release-7.0-3.el7.elrepo.noarch.rpm
```

(3) 查看可用的系统内核包,命令如下:

```
yum --disablerepo="*" --enablerepo="elrepo-Kernel" list available
```

执行命令后会出现类似如下的内容,当然随着时间的推移及版本的更新迭代,看到的信息不可能和下面完全一致,这里可以看到 4.4 和 4.18 两个版本。

```
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * elrepo-Kernel: mirrors.tuna.tsinghua.edu.cn
elrepo-Kernel
| 2.9 kB 00:00:00
elrepo-Kernel/primary_db
| 1.8 MB 00:00:03
Available Packages
Kernel-lt.x86_64
4.4.155-1.el7.elrepo
elrepo-Kernel
Kernel-lt-devel.x86_64
4.4.155-1.el7.elrepo
elrepo-Kernel
Kernel-lt-doc.noarch
4.4.155-1.el7.elrepo
elrepo-Kernel
Kernel-lt-headers.x86_64
4.4.155-1.el7.elrepo
elrepo-Kernel
Kernel-lt-tools.x86_64
4.4.155-1.el7.elrepo
elrepo-Kernel
Kernel-lt-tools-libs.x86_64
4.4.155-1.el7.elrepo
elrepo-Kernel
```

```
Kernel - lt - tools - libs - devel.x86_64
4.4.155 - 1.el7.elrepo
elrepo - Kernel
Kernel - ml.x86_64
4.18.7 - 1.el7.elrepo
elrepo - Kernel
Kernel - ml - devel.x86_64
4.18.7 - 1.el7.elrepo
elrepo - Kernel
Kernel - ml - doc.noarch
4.18.7 - 1.el7.elrepo
elrepo - Kernel
Kernel - ml - headers.x86_64
4.18.7 - 1.el7.elrepo
elrepo - Kernel
Kernel - ml - tools.x86_64
4.18.7 - 1.el7.elrepo
elrepo - Kernel
Kernel - ml - tools - libs.x86_64
4.18.7 - 1.el7.elrepo
elrepo - Kernel
Kernel - ml - tools - libs - devel.x86_64
4.18.7 - 1.el7.elrepo
elrepo - Kernel
perf.x86_64
4.18.7 - 1.el7.elrepo
elrepo - Kernel
python - perf.x86_64
4.18.7 - 1.el7.elrepo
elrepo -- Kernel
```

(4) 安装最新版本内核,命令如下:

```
yum --enablerepo = elrepo - Kernel install Kernel - ml
```

--enablerepo 选项用于开启 CentOS 系统上的指定仓库。默认开启的是 ELRepo 仓库,这里用 elrepo-Kernel 替换。

(5) 设置 grub2,内核安装好后,需要设置为默认启动选项,这样重启后才会生效。

查看系统上的所有可用内核,命令如下:

```
sudo awk - F\'' $ 1 == "menuentry " {print i++" : " $ 2}' /etc/grub2.cfg
```

出现的内容类似如下：

```
0 : CentOS Linux (4.18.7 - 1.el7.elrepo.x86_64) 7 (Core)
1 : CentOS Linux (3.10.0 - 862.11.6.el7.x86_64) 7 (Core)
2 : CentOS Linux (3.10.0 - 514.el7.x86_64) 7 (Core)
3 : CentOS Linux (0 - rescue - 063ec330caa04d4baae54c6902c62e54) 7 (Core)
```

将新的内核设置为 grub2 的默认版本，服务器上存在 4 个内核，假如要使用 4.18 版本，则可以通过 grub2-set-default 0 命令或编辑/etc/default/grub 文件进行设置。

方法一，通过 grub2-set-default 0 命令设置，其中 0 是上面查询出来的可用内核，命令如下：

```
grub2 - set - default 0
```

方法二，通过编辑/etc/default/grub 文件，设置 GRUB_DEFAULT＝0，以上面查询显示的编号为 0 的内核作为默认内核，命令如下：

```
vim /etc/default/grub
```

编辑内容类似如下：

```
GRUB_TIMEOUT = 5
GRUB_DISTRIBUTOR = " $ (sed 's, release . * $ ,,g' /etc/system - release)"
GRUB_DEFAULT = 0
GRUB_DISABLE_SUBMENU = true
GRUB_TERMINAL_OUTPUT = "console"
GRUB_CMDLINE_LINUX = "crashKernel = auto rd.lvm.lv = cl/root rhgb quiet"
GRUB_DISABLE_RECOVERY = "true"
```

生成 grub 配置文件并重启，命令如下：

```
grub2 - mkconfig  - o /boot/grub2/grub.cfg
```

执行命令后会看到类似如下的提示：

```
Generating grub configuration file ...
Found Linux image: /boot/vmlinuz - 4.18.7 - 1.el7.elrepo.x86_64
Found initrd image: /boot/initramfs - 4.18.7 - 1.el7.elrepo.x86_64.img
Found Linux image: /boot/vmlinuz - 3.10.0 - 862.11.6.el7.x86_64
Found initrd image: /boot/initramfs - 3.10.0 - 862.11.6.el7.x86_64.img
Found Linux image: /boot/vmlinuz - 3.10.0 - 514.el7.x86_64
Found initrd image: /boot/initramfs - 3.10.0 - 514.el7.x86_64.img
Found Linux image: /boot/vmlinuz - 0 - rescue - 063ec330caa04d4baae54c6902c62e54
```

```
Found initrd image: /boot/initramfs-0-rescue-063ec330caa04d4baae54c6902c62e54.img
done
```

重启系统生效，命令如下：

```
reboot
```

(6) 验证内核是否安装成功，以及当前内核的版本信息。

查看内核，命令如下：

```
uname -r
```

看到的内容类似如下：

```
4.18.7-1.el7.elrepo.x86_64
```

(7) 删除旧内核，节省磁盘资源(可选)。

查看系统中全部的内核，命令如下：

```
rpm -qa | grep Kernel
```

看到的内容类似如下：

```
Kernel-3.10.0-514.el7.x86_64
Kernel-ml-4.18.7-1.el7.elrepo.x86_64
Kernel-tools-libs-3.10.0-862.11.6.el7.x86_64
Kernel-tools-3.10.0-862.11.6.el7.x86_64
Kernel-3.10.0-862.11.6.el7.x86_64
```

方法一，使用 yum remove 命令删除旧内核的 RPM 包，命令如下：

```
yum remove Kernel-3.10.0-514.el7.x86_64 \
Kernel-tools-libs-3.10.0-862.11.6.el7.x86_64 \
Kernel-tools-3.10.0-862.11.6.el7.x86_64 \
Kernel-3.10.0-862.11.6.el7.x86_64
```

方法二，使用 yum-utils 工具，需要注意的是，如果安装的内核不多于 3 个，则 yum-utils 工具不会删除任何一个。只有在安装的内核大于 3 个时，才会自动删除旧内核。

安装 yum-utils 工具，命令如下：

```
yum install yum-utils
```

删除旧版本，命令如下：

```
package-cleanup --oldKernels
```

2. 应用更新

Linux 系统中应用软件的更新可以分为指定软件更新和全部更新，可以根据需要选择不同的更新。更新软件时，系统会列出相关的依赖，如果存在冲突的情况，则需要选择保留还是都更新。大多数情况下不会有冲突，yum 软件仓库会自动管理好相关的依赖情况，如果安装的软件依赖于一些额外的库，则这些库也会被自动安装，这也是 yum 软件仓库一个比较优秀的地方。

列出所有已安装的软件包，命令如下：

```
yum list installed
```

列出所有已安装但不在 Yum Repository 内的软件包，命令如下：

```
yum list extras
```

列出所指定软件包，命令如下：

```
yum list <package_name>
```

使用 yum 获取软件包信息，命令如下：

```
yum info <package_name>
```

列出所有已安装的软件包信息，命令如下：

```
yum info installed
```

列出所有可更新的软件包信息，命令如下：

```
yum info updates
```

列出所有已安装但不在 Yum Repository 内的软件包信息，命令如下：

```
yum info extras
```

列出所有可更新的软件清单，命令如下：

```
yum check-update
```

列出所有可更新的软件包，命令如下：

```
yum list updates
```

安装所有更新软件，命令如下：

```
yum update
```

仅安装指定的软件，命令如下：

```
yum install < package_name >
```

仅更新指定的软件，命令如下：

```
yum update < package_name >
```

列出所有可安装的软件清单，命令如下：

```
yum list
```

5.5 Linux 软件安装与卸载

Linux 常见的安装命令有 tar、zip、gz、rpm、deb、bin 等，主要可以分为三类，第一类是打包或压缩文件的 tar、zip、gz 等，一般解压后即完成安装或者解压后运行 sh 文件完成安装；第二类是对应的有管理工具的 deb、rpm 等，通常这类安装文件可以通过第三方的命令行或 UI 来简单地安装，例如用 Ubuntu 中的 apt 来安装 deb，用 Red Hat 中的 yum 来安装 rpm；第三类是. bin 类，其实就是把 sh 和 zip 打包为 bin，或把 sh 和 rpm 打包为 bin 等，当在命令行运行 bin 安装文件时，其实就是用 bin 里面的 sh 来解压 bin 中的 zip 或安装 rpm 的过程。

1. yum 软件安装与卸载

rpm 安装与卸载，这种软件包就像 Windows 的 exe 可执行文件一样，各种文件已经编译好，并打了包，哪个文件该放到哪个文件夹都指定好了，安装非常方便，在图形界面里只需双击就能自动安装，但是有一点不好，就是包的依赖关系需要手动解决。

rpm 安装软件包的步骤如下：

(1) 找到相应的软件包，例如 soft. version. rpm，下载到本机某个目录。

(2) 打开一个终端，su 成 root 用户。

(3) 通过 cd 命令进入 soft. version. rpm 所在的目录。

(4) 输入 rpm-ivh soft. version. rpm 命令进行安装。

rpm 更新软件包：

输入 rpm-Uvh soft. version. rpm 命令进行更新。

rpm 卸载软件包的步骤如下：

(1) 查找欲卸载的软件包 rpm-qa | grep xxxx。

(2) 例如找到软件 mysql-4. 1. 22-2. el4_8. 4，执行命令 rpm-e mysql-4. 1. 22-2. el4_8. 4。

如果需要查询软件的安装目录，则可以使用命令 rpm-ql mysql-4. 1. 22-2. el4_8. 4。

以. bin 结尾的安装包，这种安装包的安装方式类似于 rpm 包的安装，也比较简单。

bin 安装软件包的步骤如下：

(1) 打开一个 Shell 终端。

(2) 用 cd 命令进入源代码压缩包所在的目录。

(3) 给文件加上可执行属性 chmod ＋x . bin (中间是字母 x，小写)。

(4) 执行命令. /. bin 或者直接执行 sh ******. bin 进行安装(* 代表压缩包名称)。

bin 卸载：

把安装时选择的安装目录删除，例如 rm-rf /usr/local/mysql/。

tar. gz(bz 或 bz2 等)结尾的源代码包里面存放的都是源程序，没有编译过，需要编译后才能安装。

源代码安装的步骤如下：

(1) 打开一个 Shell 终端。

(2) 用 cd 命令进入源代码压缩包所在的目录。

(3) 根据压缩包类型解压缩文件(* 代表压缩包名称)，命令如下：

```
tar - zxvf ****. tar. gz
tar - jxvf ****. tar. bz(或 bz2)
```

(4) 用 cd 命令进入解压缩后的目录。

(5) 输入编译文件命令. /configure(有的压缩包已经编译过，这一步可以省略)。

(6) 执行命令 make。

(7) 最后执行安装文件命令 make install。

安装过程中也可以通过. /configure--help 来查看配置软件的功能；大多软件提供了. /configure 配置软件的功能；少数的也没有。如果没有提供的就不用. /configure，可直接使用 make && make install 命令进行安装；. /configure 比较重要的一个参数是--prefix，用--prefix 参数可以指定软件安装目录。

源代码卸载的步骤如下：

(1) 打开一个 Shell 终端。

(2) 用 cd 命令进入编译后的软件目录，即安装时的目录。

(3) 执行反安装命令 make uninstall。

yum 是 rpm 的管理工具，用于管理一个软件库，可以很好地解决依赖关系。一般情况

下推荐使用 yum 进行软件的安装和卸载。如果想安装一个软件，只知道它和某方面有关，但又不能确切地知道它的名字，这时可以使用 yum 的查询功能，用 yum search keyword 命令搜索。例如要安装一个 Instant Messenger，但又不知到底有哪些，这时不妨用 yum search messenger 命令进行搜索，yum 会搜索所有可用 rpm 的描述，列出所有描述中和 messeger 有关的 rpm 包，如可以得到 gaim、kopete 等相关软件包的信息，并从中选择需要的软件包进行安装。

有时还会碰到安装了一个包，但又不知道其用途，可以用 yum info packagename 命令获取信息。

安装指定版本的 Redis 版本，先查询获取可用的版本，命令如下：

```
yum --enablerepo=remi list redis --showduplicates | sort -r
```

再执行命令进行安装，命令如下：

```
yum --enablerepo=remi install redis-6.0.6 -y
```

清除 yum 缓存：

yum 会把下载的软件包和 header 存储在 cache 中，而不会自动删除。如果觉得它们占用了磁盘空间，则可以使用 yum clean 命令进行清除，更精确的方法是用 yum clean headers 清除 header，用 yum clean packages 清除下载的 rpm 包，用 yum clean all 清除所有。

清除缓存目录(/var/cache/yum)下的软件包，命令如下：

```
yum clean packages
```

清除缓存目录(/var/cache/yum)下的 headers，命令如下：

```
yum clean headers
```

清除缓存目录(/var/cache/yum)下旧的 headers，命令如下：

```
yum clean oldheaders
```

清除缓存目录(/var/cache/yum)下的软件包及旧的 headers，命令如下：

```
yum clean, yum clean all
#跟下面命令的效果相同
yum clean packages && yum clean oldheaders
```

下面对 yum 命令工具的使用进行举例说明，注意“#”号在 Shell 代码中代表注释信息说明，在命令执行时不起作用，示例如下：

```
yum update                          #升级系统
yum install packagename             #安装指定软件包
yum update packagename              #升级指定软件包
yum remove packagename              #卸载指定软件
yum erase packagename               #删除指定软件
yum grouplist                       #查看系统中已经安装的和可用的软件组,可用的可以安装
yum grooupinstall packagename       #安装上一个命令显示的可用的软件组中的一个
yum grooupupdate packagename        #更新指定软件组中的软件包
yum grooupremove packagename        #卸载指定软件组中的软件包
yum deplist packagename             #查询指定软件包的依赖关系
yum list yum\ *                     #列出所有以 yum 开头的软件包
yum localinstall packagename        #从硬盘安装 rpm 包并使用 yum 解决依赖
```

在系统中安装中文字体,命令如下:

```
yum groupinstall "fonts"
```

安装完成后查看是否成功地安装了中文语言包,命令如下:

```
locale -a |grep "zh_CN"
```

可能显示的内容如下:

```
zh_CN
zh_CN.gb18030
zh_CN.GB2312
zh_CN.gbk
zh_CN.utf8
```

以上内容说明安装成功。

2. apt 软件安装与卸载

Linux 系统的发行版本有很多个,每个系统命令略有不同,大多数情况下只是前缀不同,后面的命令都是一样的。apt/apt-get 命令适用于 deb 包管理式的 Linux 操作系统(Debian、Ubuntu 等),主要用于自动从互联网软件仓库中搜索、下载、安装、升级、卸载软件或操作系统,类似 yum 的使用,不过命令之前经常要求加上 sudo,否则命令无法执行,apt/apt-get 的主要使用方法如下:

```
apt-get install package             #安装软件包
apt-get install Package=Version     #安装指定包的指定版本
apt-get reinstall package           #重新安装包
apt-get upgrade                     #更新已安装的包
apt-cache rdepends package          #查看该包被哪些包依赖
```

```
apt-cache depends package                    #了解使用依赖
apt-get clean && apt-get autoclean           #清理无用的包
apt-cache show package                       #获取包的相关信息,如说明、大小、版本等
apt-get remove package                       #删除包
apt-get purge package                        #删除包,包括删除配置文件等
apt-get autoremove                           #自动删除不需要的包
apt-get full-upgrade                         #在升级软件包时自动处理依赖关系
apt-get search                               #搜索应用程序
apt-get show                                 #显示软件包信息
apt-get build-dep Package                    #安装源代码包所需要的编译环境
apt-get -f install                           #修复依赖关系
apt-get source Package                       #下载软件包的源代码
```

对于不在 Ubuntu 软件源中的软件,可以使用 deb 软件包进行安装。这就像 Windows 中的 exe 安装文件。例如,下载了百度云的 deb 软件包后,可以使用下面的命令安装百度云软件:

```
dpkg -i baidunetdisk_3.4.1_amd64.deb
```

关于 dpkg 管理软件的一些常用的方法总结如下。

查看所有已安装的软件,命令如下:

```
dpkg -l
```

依靠 grep 命令可查看某个软件是否已安装,命令如下:

```
dpkg -l | grep software-name
```

安装 deb 软件包,命令如下:

```
dpkg -i xxx.deb
```

删除软件包,命令如下:

```
dpkg -r xxx.deb
```

连同配置文件一起删除,命令如下:

```
dpkg -r --purge xxx.deb
```

查看软件包信息,命令如下:

```
dpkg -info xxx.deb
```

查看文件复制详情，命令如下：

```
dpkg -L xxx.deb
```

连同软件包的配置文件一起删除，命令如下：

```
dpkg -r --purge xxx.deb
```

查看软件包信息，命令如下：

```
dpkg -info xxx.deb
```

重新配置软件包，命令如下：

```
dpkg-reconfigure xxx.deb
```