ip数据包转发和iptables

Linux内核数据包转发功能和 iptables 的关系

内核数据包转发（路由）功能是内核将从A网卡接收到的目的地址不是自身地址的ip数据包通过B网卡发送出去的功能（即路由器的功能）。

使用以下命令即可开启内核对`ipv4`数据包的路由功能

```
# 如果有sysctl命令
sysctl net.ipv4.ip_forward=1
# 如果没有sysctl命令
echo 1 > /proc/sys/net/ipv4/ip_forward
```

内核将根据 路由表 将接收到的非自身ip的数据包按照路由表中的规则进行转发，转发行为本身**不会**修改ip数据包的目标地址、源地址或者端口号（如果使用了TCP或者UDP）等数据。使用route命令可以查看和修改内核路由表。

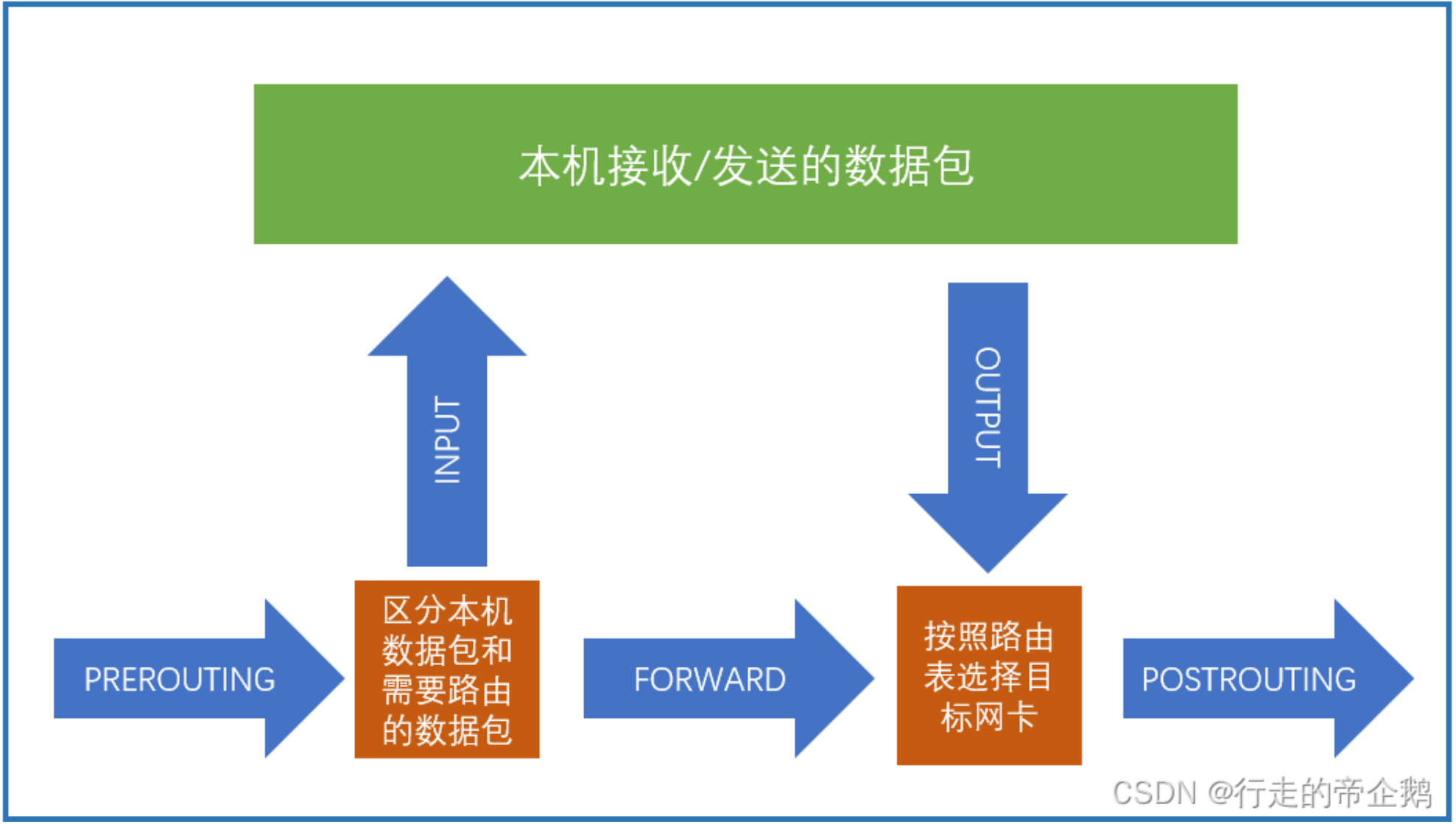
```
january@u18:/sys/kernel$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         _gateway        0.0.0.0         UG    100    0        0 enp0s3
link-local      0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
192.168.3.0     0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
```

iptables是基于内核提供的数据包过滤机制netfilter实现的数据包控制软件，可以实现数据包过滤以及数据包修改功能，Linux系统中ip数据包的转发（路由）是不需要iptables支持的。iptables主要包括三个功能，第一是数据包过滤，包括对本地接收的数据、本地发送的数据包、本地转发（路由）的数据包进行过滤；第二是实现NAT，即对数据包的源地址、目的地址、端口号等进行修改后转发的功能；第三是实现数据包修改，包括修改ip数据包的TTL等字段。

iptables基本架构

链（chain）

iptables将数据包在系统内部传递的流程分为了五个部分，分别对应五个链（chain），如下图所示。



每个链定义了一些数据包过滤或者修改的规则，这些规则将在数据包经过对应链（chain）的时候执行。

表（table)

为了区分不同的数据包处理功能，iptables定义了不同的表（table），包括filter、nat、mangle以及raw。

- filter 表：用来对数据包进行过滤，具体的规则要求决定如何处理一个数据包，其规则可以应用到三个链：input、forward、output;

- nat 表：network address translation 网络地址转换，主要用来修改数据包的 IP 地址、端口号信息，其规则可以应用到四个链prerouting、input、output以及postrouting;
- mangle 表：主要用来修改数据包的服务类型，生存周期，为数据包设置标记，实现流量整形、策略路由表内包括五个链：prerouting、postrouting、input、output、forward;
- raw表：主要用来决定是否对数据包进行状态跟踪，其规则可以应用到两个链：prerouting和output

iptables基本操作

查看规则

```
iptables -t filter -L INPUT
```

涉及到的选项如下：

```
--list    -L [chain [rulenum]]
          List the rules in a chain or all chains
--table -t table    table to manipulate (default: `filter')
```

添加规则

```
iptables -t filter -A INPUT <rule>
```

涉及到的选项如下：

```
--append  -A chain      Append to chain
--insert  -I chain [rulenum]
    Insert in chain as rulenum (default 1=first)
```

相比`append`，`insert`可以跳转规则插入的位置

删除规则

```
iptables -t filter -D INPUT <rule_index>
```

涉及到的选项如下：

```
--delete  -D chain      Delete matching rule from chain
--delete  -D chain rulenum
--flush   -F [chain]        Delete all rules in  chain or all chains
```

iptables实现数据包处理功能

数据包过滤

```
iptables -I INPUT -p icmp -j DROP
```

涉及的选项如下：

```
 --jump -j target
                target for rule (may load target extension)
 -p, --protocol protocol
                 The  protocol of the rule or of the packet to check.  The speci-
                 fied protocol can be one of tcp, udp, udplite, icmp, icmpv6,esp,
                 ah,  sctp,  mh  or  the  special  keyword  "all", or it can be a
                 numeric value, representing one of these protocols or a  differ-
                 ent  one.   A protocol name from /etc/protocols is also allowed.
                 A "!" argument before the protocol inverts the test.  The number
                 zero  is  equivalent to all. "all" will match with all protocols
                 and is taken as default when this option is omitted.  Note that,
                 in ip6tables, IPv6 extension headers except esp are not allowed.
                 esp and ipv6-nonext can be used with Kernel  version  2.6.11  or
                 later.   The  number zero is equivalent to all, which means that
                 you cannot test the protocol field for the value 0 directly.  To
                 match  on a HBH header, even if it were the last, you cannot use
                 -p 0, but always need -m hbh.

```

路由端口转发/地址转换

```
iptables -t nat -A PREROUTING -i eth0 -p udp --dport 8000 -j DNAT --to 192.168.4.1:8001
```

`DNAT`表示`Destination NAT`即目的地址转换，对应`PREROUTING`使用；`SNAT`表示`Source NAT`即源地址转换，对应`POSTROUTING`使用。

本地端口转发

将**外网访问**本地的4444端口的流量转发到本地的22端口：

```
iptables -t nat -A PREROUTING -p tcp --dport 4444 -j REDIRECT --to-ports 22
```

将**本地访问**本地的4444端口的流量转发到本地的22端口：

```
iptables -t nat -A OUTPUT -p tcp --dport 4444 -j REDIRECT --to-ports 22
```

保存规则

使用`iptables-save`可以将当前规则保存到配置文件中，然后使用`iptables-restore`即可恢复配置。

```
# 保存规则到文件
iptables-save > iptables.conf
# 从文件中读取规则
iptables-restore iptables.conf
```

参考

> netfilter/iptables project homepage - The netfilter.org project
> iptables(8) - Linux man page (die.net)