Ebtables 的使用

一、ebtables 简介

ebtables 和 iptables 类似，都是 Linux 系统下网络数据包过滤的配置工具。既然称之为配置工具，就是说过滤功能是由内核底层提供支持的，这两个工具只是负责制定过滤的 rules.

ebtables 即是以太网桥防火墙，以太网桥工作在数据链路层，ebtables 来过滤数据链路层数据包。2.6 内核内置了 ebtables，要使用它必须先安装 ebtables 的用户空间工具（ebtables-v2.0.8-2），安装完成后就可以使用 ebtables 来过滤网桥的数据包。

二、ebtables 配置

ebtables 的配置分为表、链和规则三级。

1.表

表是内置且固定的，共有三种:filter,nat,broute，用-t 选项指定。最常用的就是 filter 了，所以不设-t 时默认就是这个表。nat 用于地址转换，broute 用于以太网桥。

2.链

链有内置和自定义两种。不同的表内置的链不同，这个从数据包的流程图中就可以看出来。所谓自定义的链也是挂接在对应的内置链内的，使用-j 让其跳转到新的链中。

3.规则

每个链中有一系列规则，每个规则定义了一些过滤选项。每个数据包都会匹配这些项，一但匹配成功就会执行对应的动作。

所谓动作，就是过滤的行为了。有四种，ACCEPT，DROP，RETURN 和 CONTINUE。常用的就是 ACCEPT 和 DROP，

Ebtables 使用规则如下：

ebtables[-ttable]-[ADI]chainrule-specification[match-extensions][watcher-extensions]

-ttable:一般为 FORWARD 链。

- ADI：A 添加到现有链的末尾；D 删除规则链（必须指明规则链号）；I 插入新的规则链（必须指明规则链号）。

-P:规则表的默认规则的设置。不同的表有不同的规则。

-F:对所有的规则表的规则链清空。

-L:指明规则表。可加参数，--Lc,--Ln

-p:指明使用的协议类型，ipv4,arp 等可选（使用时必选）详情见 /etc/ethertypes

--ip-proto:IP 包的类型，1 为 ICMP 包，6 为 TCP 包，17 为 UDP 包，在 /etc/protocols 下有详细说明

--ip-src:IP 包的源地址

--ip-dst:IP 包的目的地址

--ip-sport:IP 包的源端口

--ip-dport:IP 包的目的端口

-i:指明从那片网卡进入

-o:指明从那片网卡出去

3：实例

这个实例来自 ebtables 的官网 ,ebtables 实例。这个实例主要是结合 ebtables 和 TC 来实现对用户进行流量控制。

在实际的应用中，这种场景较常见。

基于 MAC 地址来对不同的网络宽带使用者进行流量控制。

以下代码供参考

Bridgeconfiguration

----------ifcfg-br0----------

DEVICE=br0

ONBOOT=no

BOOTPROTO=static

IPADDR=192.168.111.11

NETMASK=255.255.255.0

----------bridge_up.sh----------

#!/bin/bash

ifdowneth0

ifdowneth1//关闭网络中的网卡 eth0 和 eth1.

ifconfigeth00.0.0.0up

ifconfigeth10.0.0.0up//开启 eth0,eth1,但是没有给有效的 IP 地址。即没有 IP 地址。

brctladdbrbr0//使用 brctl 命令创建网桥 br0

brctladdifbr0eth0

brctladdifbr0eth1//将网络接口添加进网桥 br0

ifconfigbr0up

----------bridge_down.sh----------

#!/bin/bash

ifdowneth0

ifdowneth1

ifconfigbr0down

brctldelbrbr0

Therateshapingpart

We'reusingTCtodothedeed.Thisismyfirstattemptatthis,soImaybedoingsomethingswrong,especiallywiththeTCcommands-BUTITWORKS-soIfigure,I'llfixitlater.Youcanuseebtables-L--Lctoseeyourcustomer'susage.Idumpthisouthourly,addingthe-Zoptiontozerothecountersout,thenhaveaperlscriptparsethatoutputanddumpitintoaMySQLtablewhereIcanmakebetteruseofit.

----------rateshape----------

#!/bin/bash

#

#AllRatesareinKbits,soinordertogetsBytesdivideby8

#e.g.25Kbps==3.125KB/s

#

TC=/sbin/tc

EBTABLES=/sbin/ebtables#Locationofebtables

cd/usr/local/bridge

tc_start(){

$TCqdiscadddeveth0roothandle1:0cbqbandwidth100Mbitavpkt1000mpu64

$TCqdiscadddeveth1roothandle1:0cbqbandwidth100Mbitavpkt1000mpu64

在 TC 中，使用"major:minor"这样的句柄来标识队列和类别，其中 major 和 minor 都是数字。

对于队列来说，minor 总是为 0，即"major:0"这样的形式，也可以简写为"major:比如，队列 1:0 可以简写为 1:。需要注意的是，major 在一个网卡的所有队列中必须是惟一的。对于类别来说，其 major 必须和它的父类别或父队列的 major 相同，而 minor 在一个队列内部则必须是惟一的(因为类别肯定是包含在某个队列中的)。举个例子，如果队列 2:包含两个类别，则这两个类别的句柄必须是 2:x 这样的形式，并且它们的 x 不能相同，比如 2:1 和 2:2。

这里，命令中的"add 表示要添加，"deveth0 表示要操作的网卡为 eth0。

"root 表示为网卡 eth0 添加的是一个根队列。"handle1:表示队列的句柄为 1:。"cbq 表示要添加的队列为 cbq 队列。

#CustomerA

#TwoMACs:00:0D:BD:A4:E1:C8and00:20:78:B0:25:7D

#256kbpsdownloadspeed

${TC}classadddeveth0parent1:0classid1:1cbqrate256KBitallot1514prio1avpkt1000bounded

${TC}filteradddeveth0parent1:0protocoliphandle1fwflowid1:1

${EBTABLES}-AFORWARD-d00:0D:BD:A4:E1:C8-jmark--set-mark1--mark-targetACCEPT

${EBTABLES}-AFORWARD-d00:20:78:B0:25:7D-jmark--set-mark1--mark-targetACCEPT

#128kbpsuploadspeed

${TC}classadddeveth1parent1:0classid1:1cbqrate128KBitallot1514prio1avpkt1000bounded

${TC}filteradddeveth1parent1:0protocoliphandle1fwflowid1:1

${EBTABLES}-AFORWARD-s00:0D:BD:A4:E1:C8-jmark--set-mark1--mark-targetACCEPT

${EBTABLES}-AFORWARD-s00:20:78:B0:25:7D-jmark--set-mark1--mark-targetACCEPT

#CustomerB

#MACAddress:00:0D:BD:A4:D6:54

#800kbpsdownloadspeed

${TC}classadddeveth0parent1:0classid1:2cbqrate800KBitallot1514prio1avpkt1000bounded

${TC}filteradddeveth0parent1:0protocoliphandle2fwflowid1:2

${EBTABLES}-AFORWARD-d00:0D:BD:A4:D6:54-jmark--set-mark2--mark-targetACCEPT

```
#64kbpsuploadspeed

${TC}classadddeveth1parent1:0classid1:2cbqrate64KBitallot1514prio1avpkt1000bounded

${TC}filteradddeveth1parent1:0protocoliphandle2fwflowid1:2

${EBTABLES}-AFORWARD-s00:0D:BD:A4:D6:54-jmark--set-mark2--mark-targetACCEPT

#CustomerC

#MACAddress:00:0A:5E:22:D1:A3

#donotrateshape!

${EBTABLES}-AFORWARD-s00:0A:5E:22:D1:A3-jACCEPT

${EBTABLES}-AFORWARD-d00:0A:5E:22:D1:A3-jACCEPT

#Blockanythingwedidn'tspecifyabove.

${EBTABLES}-AFORWARD-jDROP--log

#<myconfighasover500customersandover1100MACaddresses>

#Justkeepincrementingtheclassid,handle,flowid,andmarkvaluesforeachcustomer's

#individualspeedqueues.

}

tc_stop(){

./save_and_reset_counters
```

```
${EBTABLES}-F//将删除 ebtables 中定义的规则

$TCqdiscdeldeveth0root

$TCqdiscdeldeveth1root//删除在 eth0 和 eth1 上面定义的队列

}

tc_restart(){

tc_stop

sleep1

tc_start

}

tc_show(){

//分别查看 eth0 和 eth1 上面定义的队列，类及过滤器。

echo""

echo"eth0"

$TCqdiscshowdeveth0

$TCclassshowdeveth0

$TCfiltershowdeveth0

echo""

echo"eth1"
```

```
$TCqdiscshowdeveth1

$TCclassshowdeveth1

$TCfiltershowdeveth1

}//下面是 shell 脚本的 case 语句。

case"$1"in

start)

echo-n"Startingbandwidthshaping:"

tc_start

echo"done"

;;

stop)

echo-n"Stoppingbandwidthshaping:"

tc_stop

echo"done"

;;

restart)

echo-n"Restartingbandwidthshaping:"

tc_restart
```

```
echo"done"

;;

show)

tc_show

;;

*)

echo"Usage:rateshape{start|stop|restart|show}"

;;

esac
```