Anexo H
IPtables

Create a DiffServ Decision Point with Linux

What you need for this project:

- A Linux server capable of running the Netfilter firewall
- LAN

A Linux server that acts as a firewall or edge router can provide DiffServ policy enforcement just as a Cisco router can. It can also perform some packet-manipulations, say, to tag the priority of an 802.1p-tagged packet as it hits a locally connected Ethernet segment. Red Hat Linux 9 and Red Hat Enterprise Linux have DiffServ and 802.1p support precompiled. Other Linux kernels of Version 2.4 and above can have support for DiffServ and 802.1p compiled, as long as the kernel in question is compiled with these options enabled:

- Kernel/user netlink socket CONFIG_NETLINK
- Packet filtering CONFIG_NETFILTER
- QoS and/or fair queuing CONFIG_NET_SCHED

All options organized under that last QoS option must be enabled. If not using Red Hat 9 or Enterprise, you may also need to obtain, compile, and install the iproute2 package. This package, and the kernel modifications introduced for QoS, together form the Linux Traffic Control system, a software library that you can control using a program called tc. Issuing the tc command is the quickest way to find out whether iproute2 is installed. If you get a "command not found," then you don't have iproute2.

Since it was designed to handle the QoS issues related to all kinds of network communications, not just VoIP, the Linux Traffic Control framework can be an overwhelming topic. While this exercise will show you how to mark RTP packets for a DiffServ domain, there are more practical (albeit more expensive) options for DiffServ than LTC, such as commercial routers.

Before you go any further, you should know that Linux Traffic Control supports a vast array of buffering, prioritization, tagging, and other traffic-shaping tactics. In fact, using LTC and Linux's kernel-based firewall, iptables, about a dozen different QoS standards can be applied. Some use a technique called weighted or fair queuing, which may hold and forward packets based on size or by transmission rate control; technique sometimes called leaky bucket. LTC calls each technique for queuing or rate control a discipline. Other techniques use prioritization like 802.1p. Not all of LTC's capabilities are especially good for VoIP.
That said, configuring QoS enforcement points is more straightforward with brand-name routers because they offer well-documented, easy-to-administer QoS commands and integrate well with COPS. On Linux, a majority of QoS configurations are done using iptables, the command for altering the Linux kernel’s built-in packet-filtering firewall, known as Netfilter.

Using iptables to support DiffServ can create a policy decision point on the edge of the network. Core routers must also support DiffServ if the measure is to be successful from end to end.

Configure an iptables edge router for DiffServ

iptables can be configured to match packets based on origin/destination, protocol, or port numbers and then perform edge-style DiffServ classification before forwarding them. In this example, iptables matches all incoming or outgoing UDP traffic on a standard RTP destination port (5004) and assigns a DSCP class of EF using the iptables DSCP target:

```
iptables -A PREROUTING -p udp -d 0.0.0.0/0.0.0.0 --dport 5004 -j DSCP\  --set-dscp-class EF
```

The -A PREROUTING option tells Netfilter to apply this rule before the kernel routes the packet to its next hop. -p udp tells Netfilter to apply this rule only to UDP datagrams, ignoring TCP packets. The dport 5004 option tells Netfilter that only packets destined for port 5004 (that is, RTP packets) should have this rule applied. Next, -j DSCP describes a target chain in which to modify the packet. Last, the --set-dscp-class EF option, which works only when the DSCP target chain is specified, changes the DSCP class of the matched packets to Expedited Forwarding.

If your Linux firewall is a point of connectivity for more than one network, it’s possible to define DSCP classes on the basis of the destination network. If certain networks are associated with a different service level, you can tag the traffic as such. In this example, the 10.2.0.0 network gets Expedited Forwarding, while the 10.3.0.0 network gets Assured Forwarding:

```
iptables -A PREROUTING -p udp -d 10.2.0.0/255.255.0.0 --dport 5004 -j DSCP\  --set-dscp-class EF
iptables -A PREROUTING -p udp -d 10.3.0.0/255.255.0.0 --dport 5004 -j DSCP\  --set-dscp-class AF
```
Likewise, if protocols other than RTP are to make use of DiffServ, you can have your decision point tag them appropriately. This example tags IAX and RTP traffic as EF, while tagging all other UDP traffic as AF:

```
iptables -A PREROUTING -p udp -d 0.0.0.0/0.0.0.0 --dport 5004,5036,4569\
  -j DSCP --set-dscp-class EF
iptables -A PREROUTING -p udp -j DSCP --set-dscp-class AF
```

The DSCP target of iptables will work only if the kernel has been compiled with the CONFIG_IP_NF_TARGET_DSCP option enabled, as described earlier.

iptables provides DiffServ awareness for only IP Version 4.

Once a packet is marked with the appropriate DSCP class, it's up to the core routers to treat that mark with respect. If the DSCP class is EF, like the previous iptables example, then core routers need to expedite the traffic immediately.