

Linux Bant Geniřliđi Yönetimi

Serkan Kenar

<serkan@labristeknoloji.com>

İçerik

- Trafik Kontrolü
- Linux Araçları
 - iproute2, tc
- Kuyruklama Düzenleri
 - HTB, TBF, SFQ
- Gelişmiş Kullanım Örnekleri
- Hazır Paketler

Trafik Kontrolü

- Ağ trafiğinin kontrol edilebilmesi amacıyla
 - önceliklerin belirlenmesi,
 - gecikmelerin azaltılması,
 - performansın iyileştirilmesi,
 - bant genişliğinin sınırlandırılması veya garanti edilmesi işlemlerine **trafik kontrolü** denir.
- **QoS**: Servis Kalitesi, farklı bağlantı türlerine farklı seviyelerde kalite tanımlamalarının yapılabilmesidir.

QoS Uygulamaları

- **FTP:** gecikmeler önemsiz, ama hızlı olmalı.
- **Sesli görüşmeler (VoIP):** gecikme, görüşme kalitesini düşürür.
- **SSH ve Telnet:** Küçük paketler ama etkileşim gerektirdiğinde hızlı tepki alınabilmelidir.
- Paket kaybına dirençsiz protokollere öncelik verilmeli.

Neden gerekli?

- Trafik önceliklerin belirlenmesi gerektiğinde
- Kullanım, mevcut bant genişliği kapasitesinin üzerindeyse
- Kullanıcılar kaynakları kötüye kullanıyorsa
- Kurumsal hizmetlerin kalitesinin sağlanması isteniyorsa
- Gecikmeye duyarlı hizmetlerin kalitesi yükseltmek isteniyorsa

Servis Kalitesi Kavramları

- **Gecikme:** Paketlerin gidiş-gelişlerde geçen süre
- **Jitter:** Gecikme süresinin zaman içinde değişiklik göstermesi
- **Paket Kayıpları:** Ağ üzerinde birikmeler nedeniyle oluşabilecek paket kayıpları

Kuyruklar

- Trafik kontrolünün temeli ađ geitleri zerindeki kuyruklardır.
- Geliş hızı, iletim hızını getiđinde, paketler kuyrukta bekletilir.
- Kuyruk dolduđunda paketler dşrlr.

Akış

- İki makine arasındaki ilgili trafiğe *akış* denir.
- TCP ve UDP bağlantıları birer akış oluşturur.
- Birden çok akış tek noktada bant genişliğini paylaşırlar.

Trafik Kontrol İşlemleri

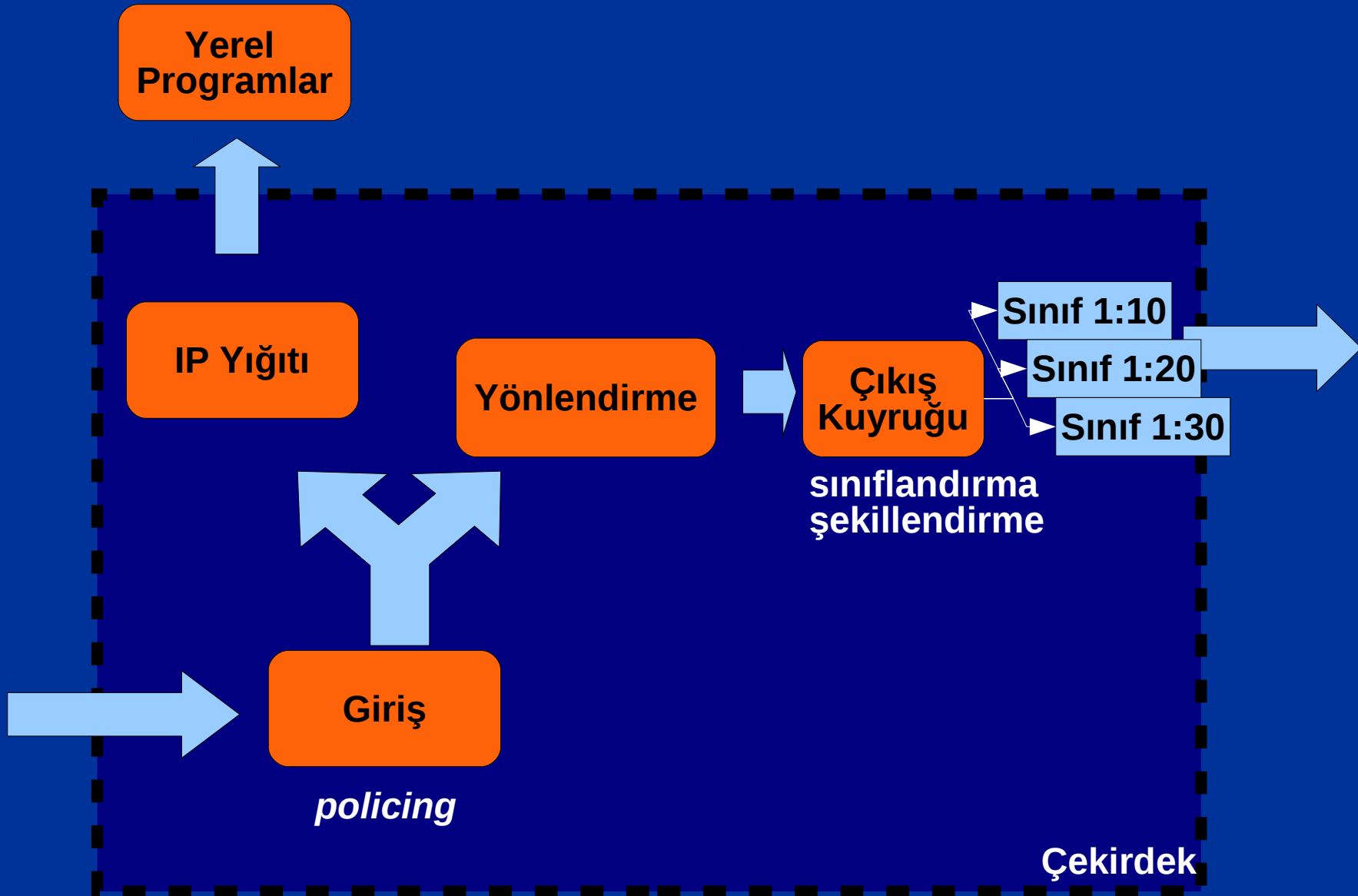
- **Şekillendirme:** Bağlantı hızını belirlemek için paketlerin bekletilmesi
- **Zamanlama:** Paketlerin giriş/çıkış sırasının değiştirilmesi
- **Sınıflandırma:** Paketlerin farklı kuyruklara sınıflandırılması
- **Policing:** Bağlantının kısıtlanması amacıyla sınırı aşan trafiğin düşürülmesi

İçerik

- Trafik Kontrolü
- Linux Araçları
 - iproute2, tc
- Kuyruklama Düzenleri
 - HTB, TBF, SFQ
- Gelişmiş Kullanım Örnekleri
- Hazır Paketler

Linux Araçları

- Trafik Kontrolü, çoğu dağıtım çekirdeğinde hazır olarak gelir.
- “tc” komutuyla yönetilebilir.
- iproute2 projesiyle ağ yapılandırması ve bant genişliği yönetimi yapılabilir.
- linux-net.osdl.org/index.php/lproute2



TC Komutu

- **Kullanım:**
- tc SEÇENEKLER NESNE { İŞLEM | help }
 - tc **qdisc** { add | del | replace | change | get } dev <DEV> ...
 - tc **class** { add | del | change | get } dev <DEV> ...
 - tc **filter** { add | del | change | get } dev <DEV> ...

QDisc

- Kuyruklama düzenleri paketlerin nasıl işleneceğini belirleyen algoritmalarıdır.
- Bazı qdisc'ler paketleri geldikleri sırayla gönderirken, bazıları hiyerarşik bir yapı kurulmasına olanak sağlayacak kadar karmaşıklaşabilir.

Sınıflar

- Sınıflı kuyruklama düzenlerinde yer alan kuyruklardır.
- Farklı sınıflar barındırabilir veya bir kuyruk düzeni içerip ayrılan trafiğin yönetilmesine olanak sağlar.

Filtreler

- Trafiğin çeşitli eşleşme kurallarıyla seçilerek bir sınıfa yönlendirilmesini sağlar. (Sınıflandırma işlemi)
- *Policing* kalıpları da barındırarak hızı belirlenebilir.
- Paket önce kök qdisc'e ait filtrelerden geçer, sonra alt sınıfların filtrelerinden de geçebilir.

İçerik

- Trafik Kontrolü
- Linux Araçları
 - iproute2, tc
- **Kuyruklama Düzenleri**
 - **HTB, TBF, SFQ**
- Gelişmiş Kullanım Örnekleri
- Hazır Paketler

Kuyruklama Düzenleri (*qdisc*)

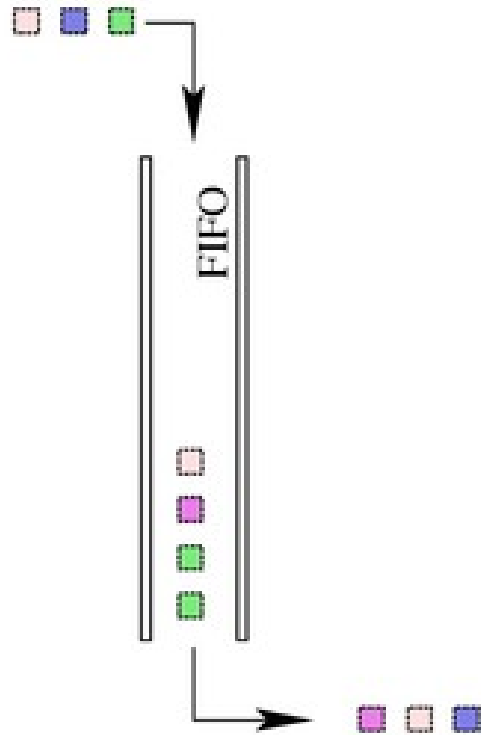
- Tüm çıkış arabirimleri bir *qdisc*'e sahiptir.
- Sadece çıkan trafik düzenlenebilir.
- Sınıfsız *qdisc*'ler ile trafik sınıflandırılmadan yönetilebilir.
- Sınıflı *qdisc*'ler ile farklı kuyruklarda farklı kuyruklama disiplinleri kullanılabilir.

Kuyruklama Düzenleri

- Sınıfsız Kuyruklama Düzenleri:
 - FIFO
 - PFIFO_Fast
 - SFQ
 - ESFQ
 - TBF
- Sınıflı Kuyruklama Düzenleri
 - HTB
 - CBQ

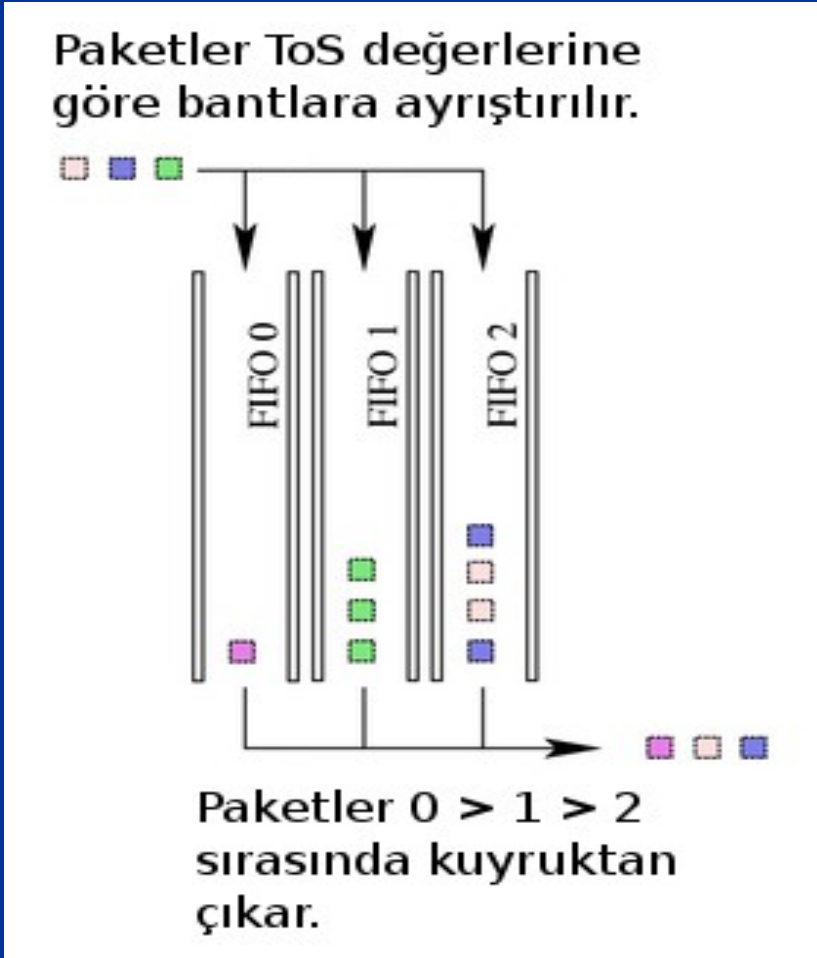
FIFO

First-in First-out (FIFO)



- İlk Gelen İlk Çıkar
- En hızlı, en az CPU harcayan yöntem
- Sınıflarda başka bir qdisc tanımlanıncaya kadar öntanımlı olarak FIFO kullanılır.

PFIFO_Fast

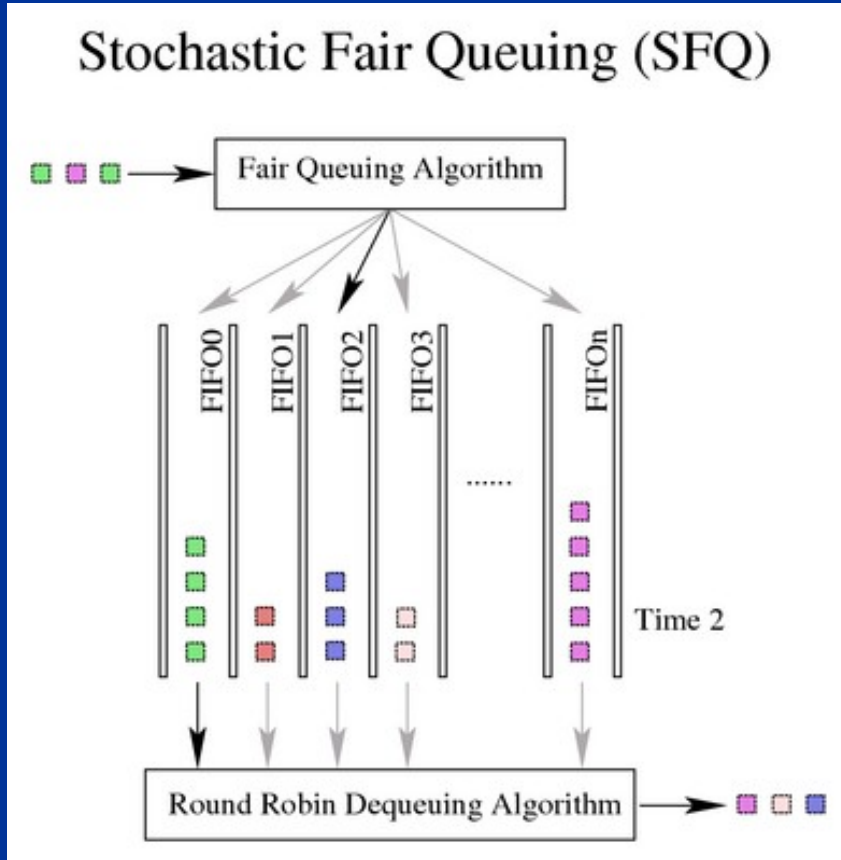


- Tüm arabirimlerde öntanımlı qdisc olarak kullanılır.
- 3 farklı önceliğe sahip bantlar üzerinde paketler dağıtılır.

Servis Tipleri

- Asgari gecikme (Interaktif) **0. Bant**
- Normal Servis veya Azami Güvenilir (En iyi çaba) **1. Bant**
- Asgari Maddi Harcama veya Azami Hız **2. Bant**

SFQ ve ESFQ

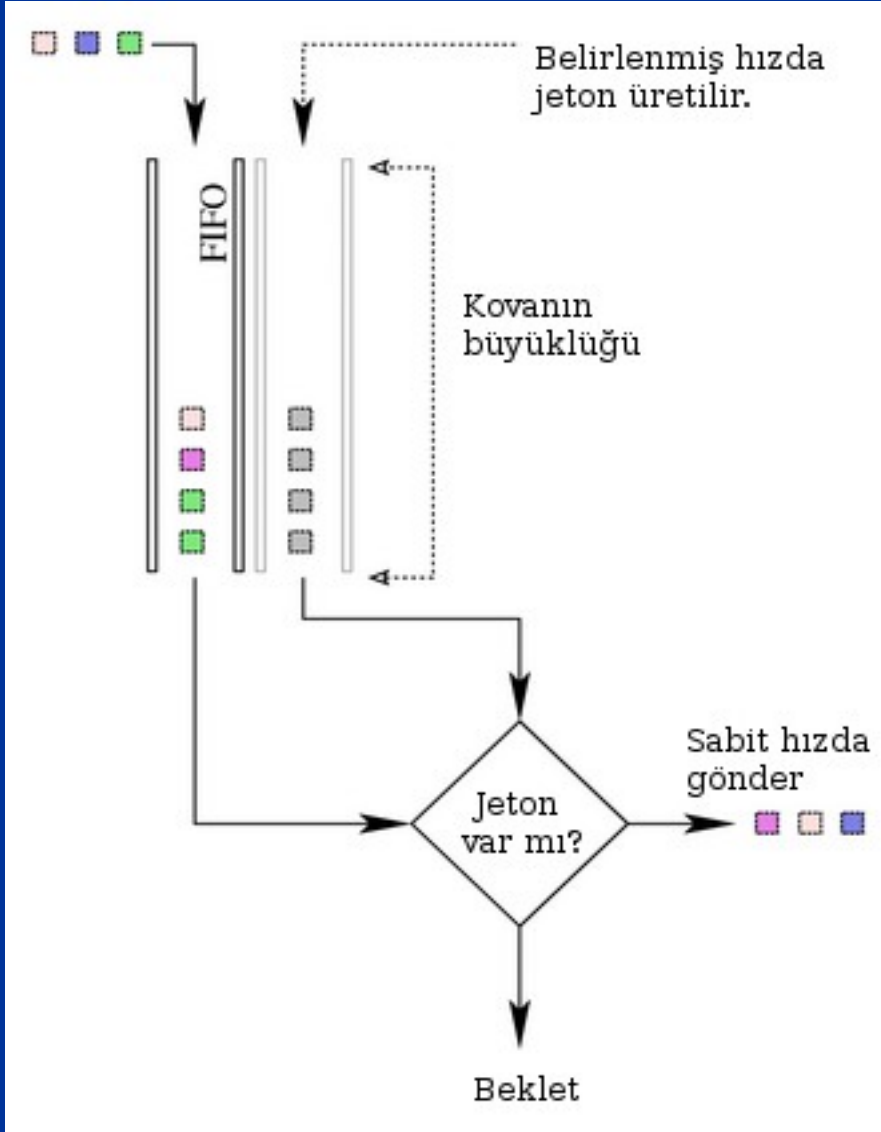


- Farklı bağlantı akışlarına eşit davranılmaya çalışılır.
- Tek bir bağlantının, bağlı bulunduğu sınıfı tamamen tüketmesi engellenir.

SFQ Tanımlama

- `tc qdisc add dev eth0 root sfq \`
`perturb 10`
- 10 saniyede bir bağlantılar tekrar dağıtılmaya başlanır.

TBF



- Bir arayüzden geçen trafiğin hızını kısıtlamak için idealdir.
- Hızı belirlenmiş seviyeye paketleri bekleterek düşürür.

TBF Tanımlama

- `tc qdisc add dev eth0 root tbf \
rate 0.5mbit \
burst 5kb \
latency 70ms`

CBQ - HTB

- **CBQ:** Karmaşık, doğru şekilde ayarlaması zor ve eski
- **HTB:** Basitçe ayarlanabilir, varsayılan davranışları daha mantıklıdır.
Muhtemelen ilk seferde doğru şekilde ayarlayabilirsiniz!
- CBQ, algoritması nedeniyle çoğunlukla doğru ölçüm yapamaz ve başarımı düşüktür.

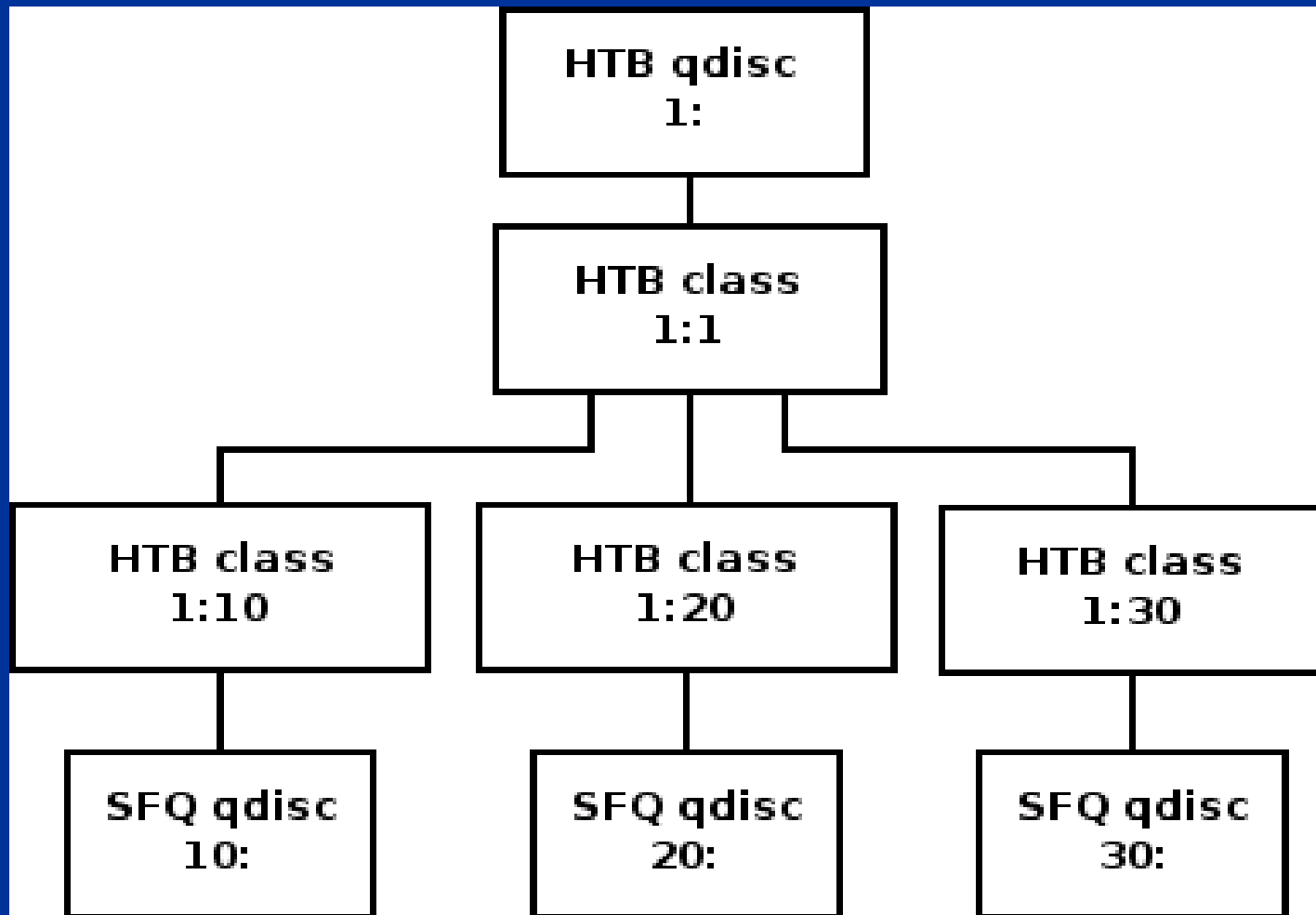
HTB

- Hierarchy Token Bucket
- CBQ'nun yerine kolay anlaşılır bir alternatif olarak geliştirilmiştir.
- Hattınızın çıkış trafiğini sınıflara ayırarak kontrol etmenize olanak sağlar.
- TBF algoritmasına benzer bir algoritma kullanır.

Sınıflar

- Sınıflar sadece sınıflı qdisc'lerde bulunur.
- Bir sınıf başka sınıflar içerebilir veya tek bir qdisc ile sonlanır.
- Her sınıfa filtreler bağlanır. Filtreler ile ilgili sınıfa aktarılacak olan trafik belirlenir.
- Uç sınıflar öntanımlı FIFO ile sonlanır.

Sınıflar (2)



Sınıflandırma

- HTB sınıflandırması sadece kök qdisc üzerindeki filtrelerle yapılır.
- Paketler sınıflandırılmaya kökten başlar. Aktarıldıkları sınıfta
 - a) filtreler çalıştırılır ve başka sınıfa aktarılır,
 - b) sınıf uç noktaysa, paket o sınıfa eklenir. Sınıfa bağlı filtrelerle işlem baştan devam eder.

HTB QDisc

- Kök kuyruklama düzeni parametleri:
 - **handle** <major:>, Bu kuyruklama düzeninin ismi
 - **parent** <major:minor>, Bu kuyruklama düzeninin üst-sınıfı
 - **default** <minor>, sınıflandırılmamış trafiğin gönderileceği çocuk sınıf

```
tc qdisc add dev eth0 root \  
    handle 1: htb default 10
```

HTB Sınıfları (1)

- **parent major:minor**; sınıfın üst-sınıfı
- **classid major:minor**; sınıf ismi
- **prio öncelik**; sınıfın önceliği (düşük sayı daha yüksek öncelik belirtir.)
- **rate hız**; bu sınıfa ve alt sınıflarına garanti edilen hız
- **ceil tavan**; üst sınıfın boş bant genişliğinden kullanılabilen tavan sınır. (Öntanımlı olarak hıza eşittir.)

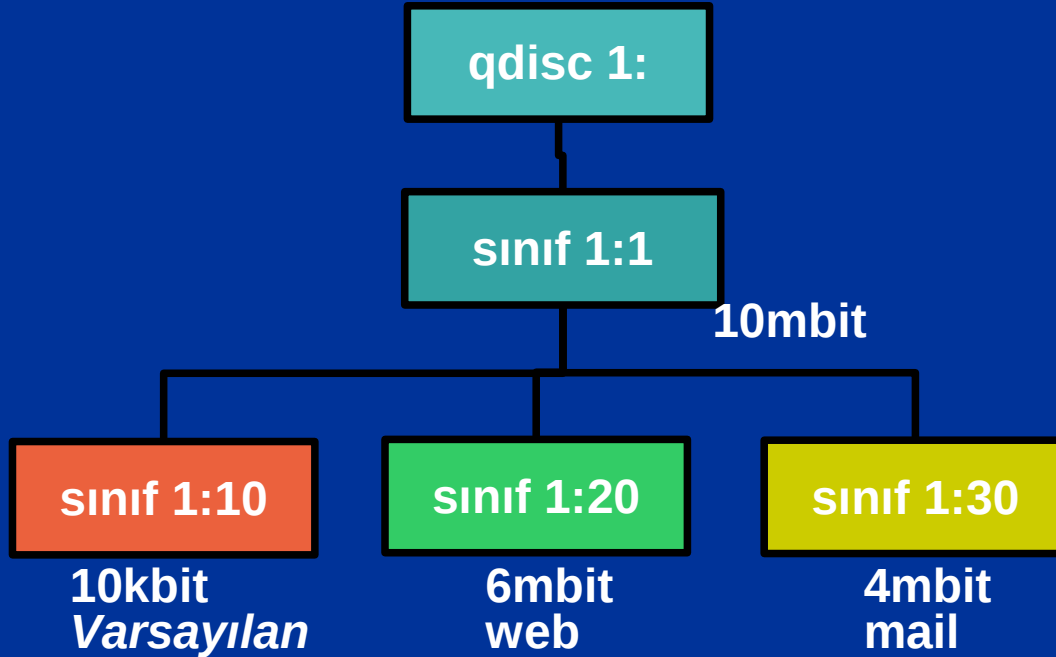
HTB Sınıfları (2)

- **burst sığrama**; azami hızda gönderilecek bayt miktarı
- **cburst sığrama**; sonsuz hızda gönderilecek bayt miktarı
- `tc class add dev eth0 parent 1: classid 1:1 htb rate 100kbit burst 1k`

Örnek Yapılandırma

- `tc qdisc add dev eth0 root handle 1: htb default 10`
- `tc class add dev eth0 parent 1: classid 1:1 htb rate 10mbit burst 10k`
- `tc class add dev eth0 parent 1:1 classid 1:10 htb rate 10kbit ceil 10mbit burst 10k`
- `tc class add dev eth0 parent 1:1 classid 1:20 htb rate 6mbit burst 15k`
- `tc class add dev eth0 parent 1:1 classid 1:30 htb rate 4mbit ceil 10mbit burst 15k`

Örnek Yapılandırma (2)



- Sınıflara öntanımlı olarak FIFO qdisc'i eklenir.
- Henüz filtreleri tanımlamadık. Tüm trafik 1:10'dan geçecek.

Uç Sınıflar

- HTB ile birlikte öntanımlı FIFO kuyruklama düzenini kullanmak iyi performans vermeyebilir.
- SFQ daha iyi bir tercih olacaktır:

```
tc qdisc add dev eth0 parent \
  1:10 handle 10: sfq perturb
  10
```

Filtreler

- Örnek yapılandırmamızı tamamlayalım:
- `tc filter add dev eth0 protocol ip \ parent
1:0 prio 1 \
u32 match ip sport 80 0xffff \
flowid 1:20`
- `tc filter add dev eth0 protocol ip \ parent
1:0 prio 1 \
u32 match ip sport 25 0xffff \
flowid 1:30`

Filtreler (2)

- Trafiği sınıflandırmak için filtreler kullanılır.
- HTB kullanırken tüm filtreler kök sınıfa eklenmelidir.
- Kullanım:
 - `tc filter add dev <INT> parent <ID:> protocol ip prio <ÖNCELİK> <eşleşme metodu>`
- Eşleşme metodu: u32 ve fwmark

Eşleşme Metodu: *u32*

- Adrese göre filtreleme:
 - `tc filter add dev eth0 parent 1:0 protocol ip prio 1 u32 \`
`match ip dst 1.2.3.4/32` flowid 1:10
- Porta göre filtreleme:
 - **`match ip dport 22 0xffff`** flowid 1:20
- Protokole göre filtreleme:
 - **`match ip protocol 50 0xff`** flowid 1:30

Eşleşme Metodu: *fwmark*

- U32 ile eşleştirme kısıtlayıcı olabilir. Netfilter/Iptables ile paketler işaretlenebilir ve fwmark metoduyla eşleştirilebilir:

```
# iptables -A PREROUTING -t mangle -o  
eth0 -p tcp --dport 22 -j MARK  
--set-mark 23  
  
# tc filter add dev eth0 protocol ip  
parent 1:0 prio 1 handle 23 fw  
flowid 1:20
```

HTB - Baęlantı Paylaşımı

- Sık istenen bir durum, sınıfların kullanmadığı bant genişliğinin paylaşılabilmesidir.
- Örnekte sınıfları köke doğrudan eklemeden, farklı bir sınıfa ekledik.
- HTB bu durumda üst-sınıfın kullanılmayan bant genişliğini oranları nispetinde diğer sınıflara paylaşır.

Bağlantı Paylaşımı

- **A sınıfı:** 100kbit - 100kbit
 - **a sınıfı:** 30kbit - 100kbit
 - **b sınıfı:** 20kbit - 100kbit
 - **c sınıfı:** 50kbit - 100kbit

- c sınıfının kullanılmayan bant genişliği a ve b sınıfların oranları nispetinde paylaşılır: (3:2)

En iyi Kullanım Uygulamaları

- Etkileşimli trafiği hızlandırmak
- ACK paketlerini önceliklendirmek
 - Yüklü gönderim sırasında, indirmelerin etkilenmemesi için
- Gelen trafiği kısıtlamak (!)
 - Vardığı anda, zaten bant genişliği harcanmıştır. Düşürülürse tekrar gönderilmesi gerekecek.

Etkileşimli Trafiği Hızlandırmak

- SSH gibi etkileşim gerektiren trafiğin hızlı akması istenir.
- Bu tür trafiği öncelikli ve garantili bir sınıfa aktarmalıyız.
- `tc filter add dev eth0 protocol ip parent 1:0 prio 10 u32 \`
`match ip tos 0x10 0xff flowid 1:10`

ACK Paketlerini Önceliklendirmek

- Her gelen paketi doğrulamak için ACK gönderilir.
 - Eğer gönderim yüklüyse, ağ geçidi üzerinde ACK paketleri gecikebilir veya düşürülebilir.
- ```
tc filter add dev eth0 parent 1: protocol ip prio 10 u32 \
match ip protocol 6 0xff \
match u8 0x05 0x0f at 0 \
match u16 0x0000 0xffc0 at 2 \
match u8 0x10 0xff at 33 \
flowid 1:10
```

 (Öncelikli sınıf)

# Gelen Trafiği Durdurmak

- Çoğunlukla gelen trafiğin kısıtlanması tercih edilmez.
- Gelen trafiği belli bir hıza kısıtlamak için:
  - `tc qdisc add dev eth0 handle ffff: ingress`
  - `tc filter add dev eth0 parent ffff: protocol ip prio 10 \`  
`u32 match ip src 0.0.0.0/0 police \`  
`rate 256kbit burst 10k drop flowid :1`

# Kurallar, Tavsiyeler

- Trafik şekillendirme yapacak olan ağ geçidi, hattın darboğaz noktası olmalıdır.
- Sadece gönderilen trafik şekillendirilebilir. Gelen trafik sadece düşürülebilir.
- Sınıflarda öntanımlı FIFO kullanılır. HTB ile birlikte SFQ kullanılması tavsiye edilir.

# Kurallar, Tavsiyeler (2)

- Bant genişliğinin bilinmediği durumlarda HTB kullanılabilir.
- Trafiği sınıflandırmak için Netfilter/Iptables ve fwmark ile eşleştirmeyi tercih edin.
- Her bir IP adresine belirli bir bant genişliği ayırmak için, aynı sayıda sınıf tanımlanmalıdır.

# Kurallar, Tavsiyeler (3)

- LAN'daki kullanıcıların indirme hızlarını düşürmek için, ağ geçidi üzerinde iç bacaktaki arayüzden makineye giden trafikte şekillendirme yapılmalıdır.

# İçerik

- Trafik Kontrolü
- Linux Araçları
  - iproute2, tc
- Kuyruklama Düzenleri
  - HTB, TBF, SFQ
- **Gelişmiş Kullanım Örnekleri**
- Hazır Paketler

# Gelişmiş Kullanım Örnekleri

- Iptables ile gelişmiş eşleşmeler yapılabilir.
- Bu sayede kullanıcılara göre (UNIX kullanıcı ID'si), çalışan uygulamalara göre filtreleme yapılabilir.
- P2P trafiğini kısıtlamak.
- SYN Flood saldırılarını engellemek için trafik kontrolü kullanılabilir.

# Programların Hızını Kısıtlamak

- Yazılımların gönderim veya indirim için hız kısıtlaması özelliği olmadığı durumlarda kullanılabilir.
- **iptables -t mangle -I POSTROUTING --m pid --pid 1182 -j MARK --set-mark 1**
  - Pid:1182 sürecin paketlerini işaretledik
  - Bu paketler fwmark ile kısıtlı bir sınıfa yönlendirilebilir.

# P2P Trafikini Engellemek

- **ipp2p** eşleşmesi kullanılarak, P2P trafiği kısıtlı bir sınıfa aktarılabilir.
- **iptables -t mangle -A PREROUTING \**  
**-p tcp -m ipp2p --ipp2p \**  
**-j MARK --set-mark 1**

# SYN Flood Saldırılarını Engellemek

- Gelen trafiği durdurmak için giriş (ingress) trafiğine kısıtlama getirilmelidir.
  - `tc qdisc add dev eth0 handle ffff: ingress`
  - `tc filter add dev eth0 parent ffff: protocol ip prio 50 handle 1 fw \`  
`police rate 1kbit burst 40 mtu 9k`  
`drop flowid :1`

# İçerik

- Trafik Kontrolü
- Linux Araçları
  - iproute2, tc
- Kuyruklama Düzenleri
  - HTB, TBF, SFQ
- Gelişmiş Kullanım Örnekleri
- **Hazır Paketler**

# Hazır Paketler

- ADSL bağlantılar için standart bir betik:  
[lartc.org/wondershaper/](http://lartc.org/wondershaper/)
- Kolay HTB kullanımı için hazır betik:  
[sourceforge.net/projects/htbinit/](http://sourceforge.net/projects/htbinit/)

# Sonuç

- Bant genişliği yönetimi, büyük ağlarda ve kısıtlı kaynaklarda kaçınılmazdır.
- Linux bunun için gerekli tüm araçları barındıran güçlü bir çözüm sunar.

# Kaynaklar

- [www.lartc.org](http://www.lartc.org)
- Traffic Control HOWTO
- *lartc* Mail Listesi

# Teşekkürler

Sorular ?