Задръствания и управление на потоците в мрежата на 2, 3 и 4 ниво

Задръстване и ресурси

Контролът за задръстване (Congestion control) в мрежите и заделянето на ресурси (resource allocation) са двете страни на една и съща монета.

Прецизното заделяне на ресурси е трудна задача, защото въпросните ресурси (буфери, пропускателна способност на линиите) са разпръснати по цялата мрежа.

Задръстване и ресурси (2)

Можете да изпращате в мрежата, колкото си искате пакети, да ги възстановявате, но това ще наруши нормалната работа.

Затова се налага да прилагате механизми за контрол на задръстванията, т.е. да има някаква предсказуемост на процесите.

Какво включва Congestion Control

Контролът за задръстване и *заделянето на ресурси* включва както крайните устройства (хостове), така и междинните - мрежовите (рутери и др.).

При мрежовите устройства могат да се прилагат различни *политики* за подреждане в опашките, при запълване на буферите и предаване по линиите, т.е. кои пакети да се предават, кои да се изхвърлят.

При хостовете контролът за задръстване се отнася до това колко бързо се разрешава на източниците да изпращат пакети за да не се случи задръстване, а ако се случи – лесно да бъде преодоляно.

Терминология

Заделяне на ресурси е процесът, чрез който мрежовите елементи се опитват да посрещнат изискванията на различните приложения по отношение на скорост на линии, буферно пространство, мощност на процесора в рутери и суичове.

Контрол за задръстване са усилията, полагани от мрежовите възли да предпазят или да реагират в случаи на свръхнатоварване.

Това може да се постигне най-просто чрез забрана на някои хостове да предават.

За да има справедливост, повечето механизми включват в себе си и заделяне на ресурси.

Flow control vs. Congestion control

Управлението на потока предпазва един “бърз” изпращач да препълни “бавен” получател.

Контролът на задръстванията предпазва множество от изпращачи (цялата мрежа) да изпращат много данни в мрежата при недостиг на ресурси.

Двете концепции споделят някои общи механизми.

Класическо “тясно място”

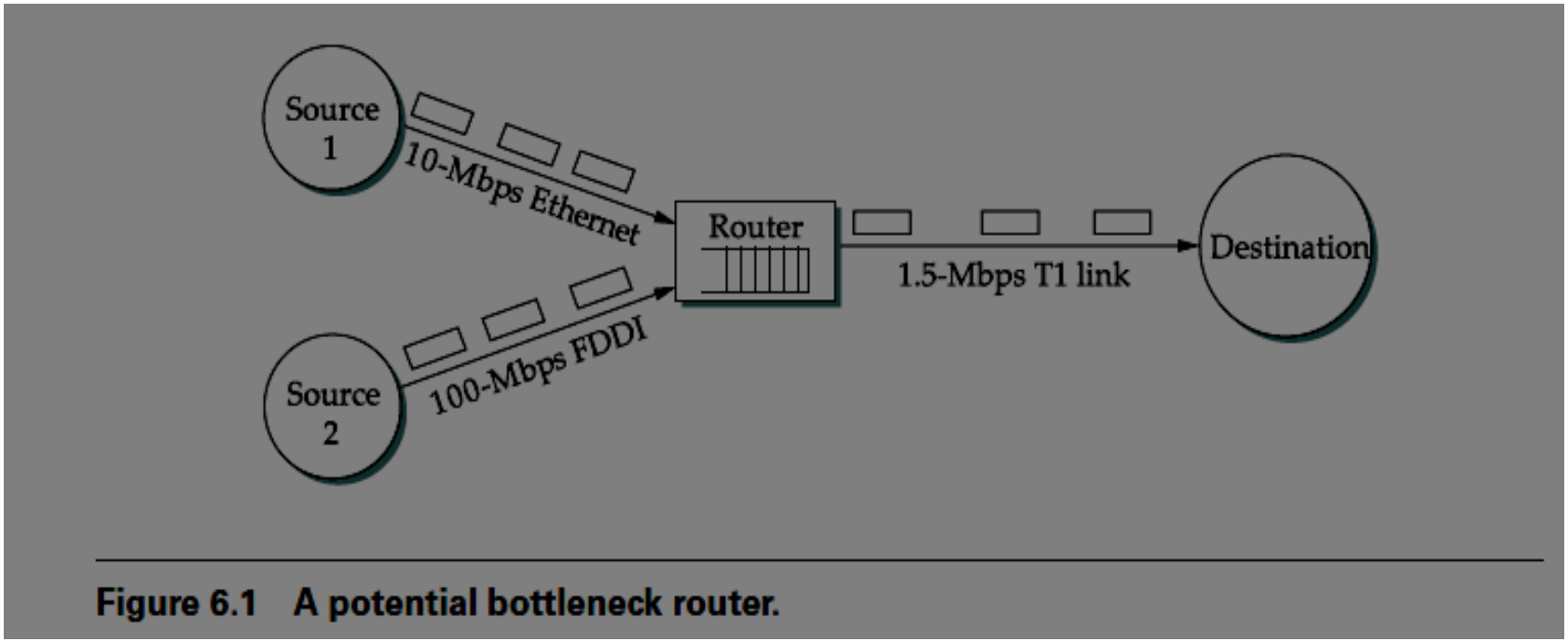


Figure 6.1 A potential bottleneck router.

Две висикоскоростни линии “захранват” нискоскоростна. Рутерът може да бъде задръстен и става “тясното място” (bottleneck router).

Resource Allocation схеми:

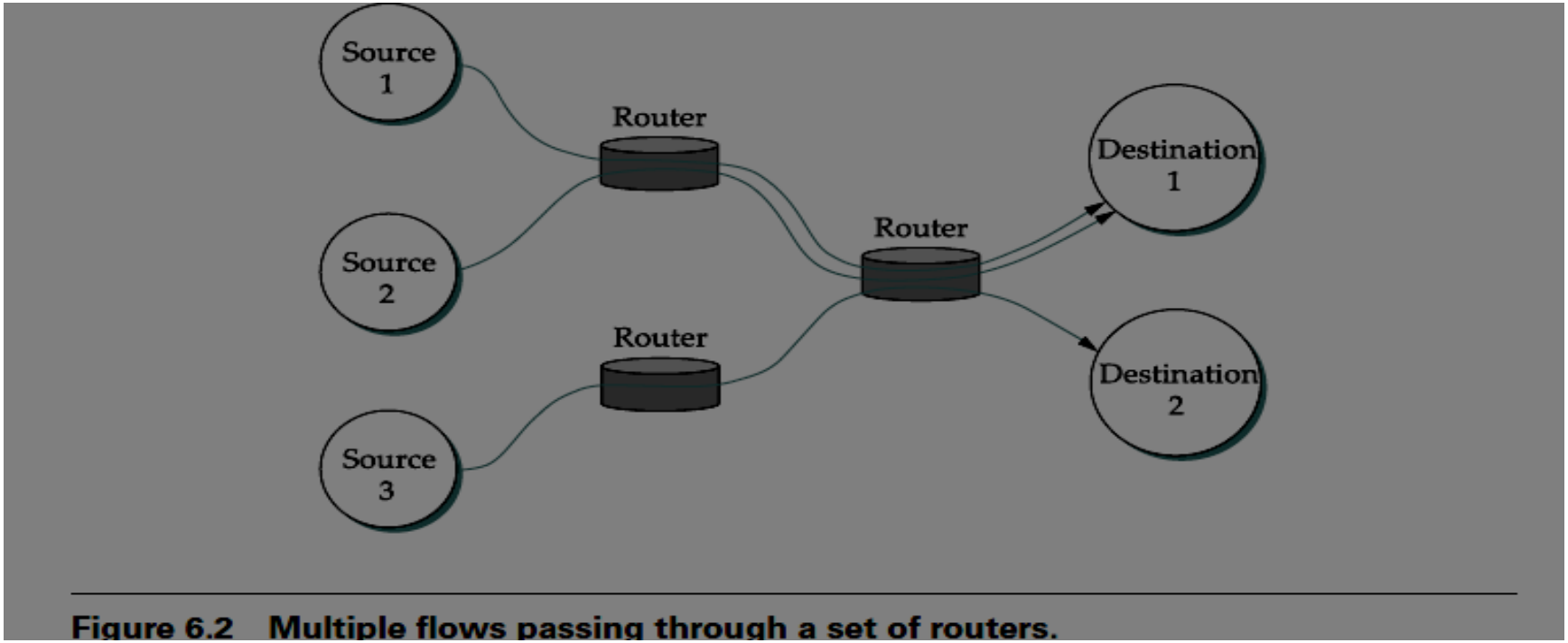


Figure 6.2 Multiple flows passing through a set of routers.

Потокът може да е host-to-host (един и същ source/destination IP) или process-to-process (един и същ source/destination port). Това може да се нарече канал.

Потокът е видим за рутера, а каналът е end-to-end.

Router-Centric vs. Host-centric.

Router-Centric – всеки рутер отговаря дали да препрати пакета или да го изхвърли и да информира хостовете колко пакета им е разрешено да изпращат.

Host-centric – крайните устройства (хостове) наблюдават колко пакета са минали успешно през мрежата и сами си настройват “поведението”.

Reservation-Based vs. Feedback-Based

Reservation-based – **хостът пита** (проверява) **мрежата** за ресурси. Подобно е на телефона – дава “свободно” или “заето”.

Feedback-based – хостът изпраща данни без “предварителна резервация” и след това си нагласява скоростта в зависимост от обратната връзка:

- експлицитно - “please slow down” или

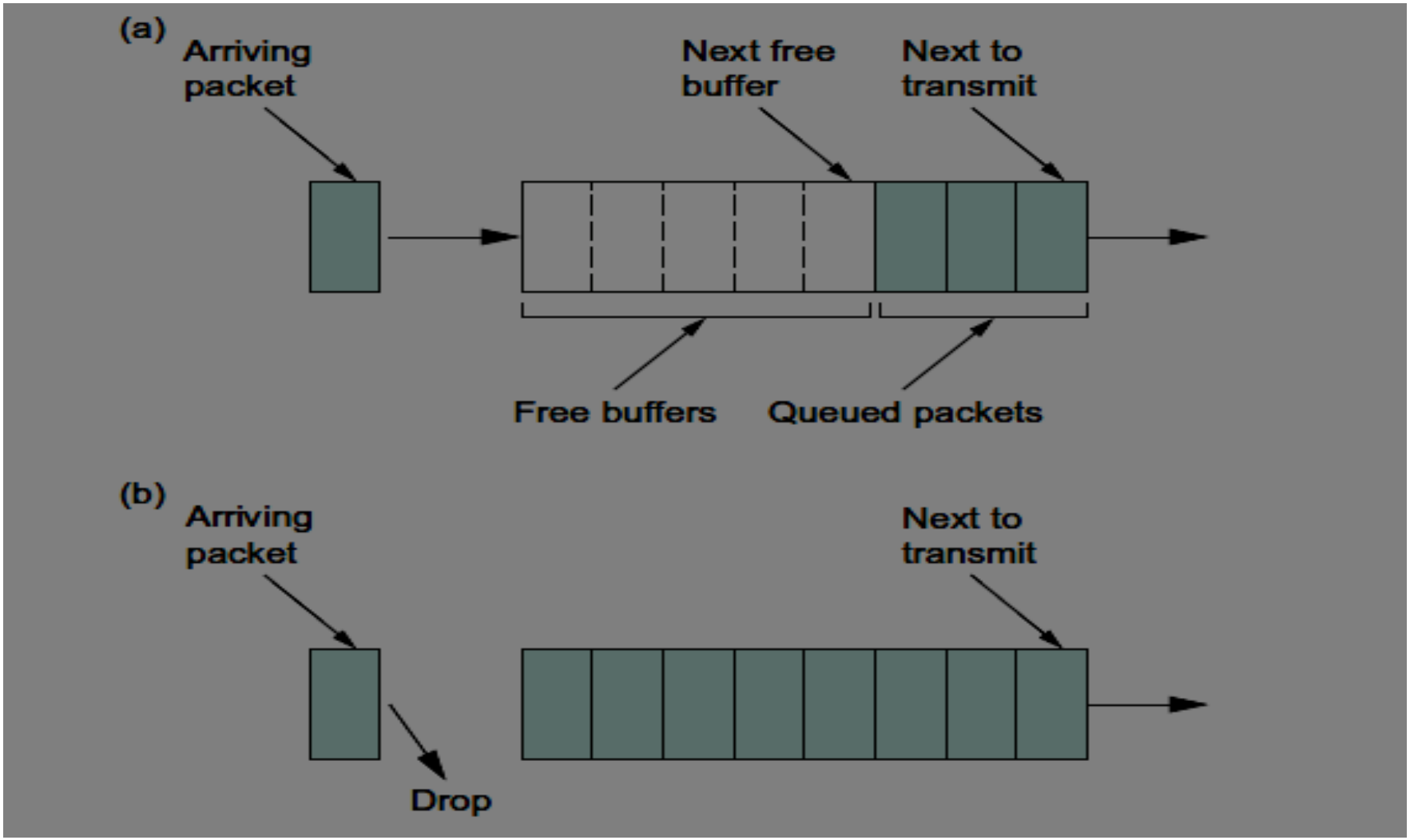
- имплицитно - в зависимост от положението на мрежата (напр. загуба на пакети).

Window-Based vs. Rate-Based

Прозоречният механизъм ни е известен от TCP и каналния слой – поддържа управлението на потока.

Да се контролира поведението на изпращача чрез скоростта: колко bits per second може да поеме получател или мрежата. Има отношение към качеството на услугата (QoS).

Resource Allocation. Опашките: FIFO



FIFO

FIFO опашки (queuing) с политика first-come-first-served (FCFS).

Механизмът tail drop (изхвърляне на пакет - случай (b) на предишния слайд) широко се използва от рутерите в Интернет.

Вариант на FIFO са опашките с приоритети. Всеки IP пакет да има приоритет, например указан в полето Type of Service (TOS). Получават се множество FIFO опашки с различни приоритети.

Fair Queuing

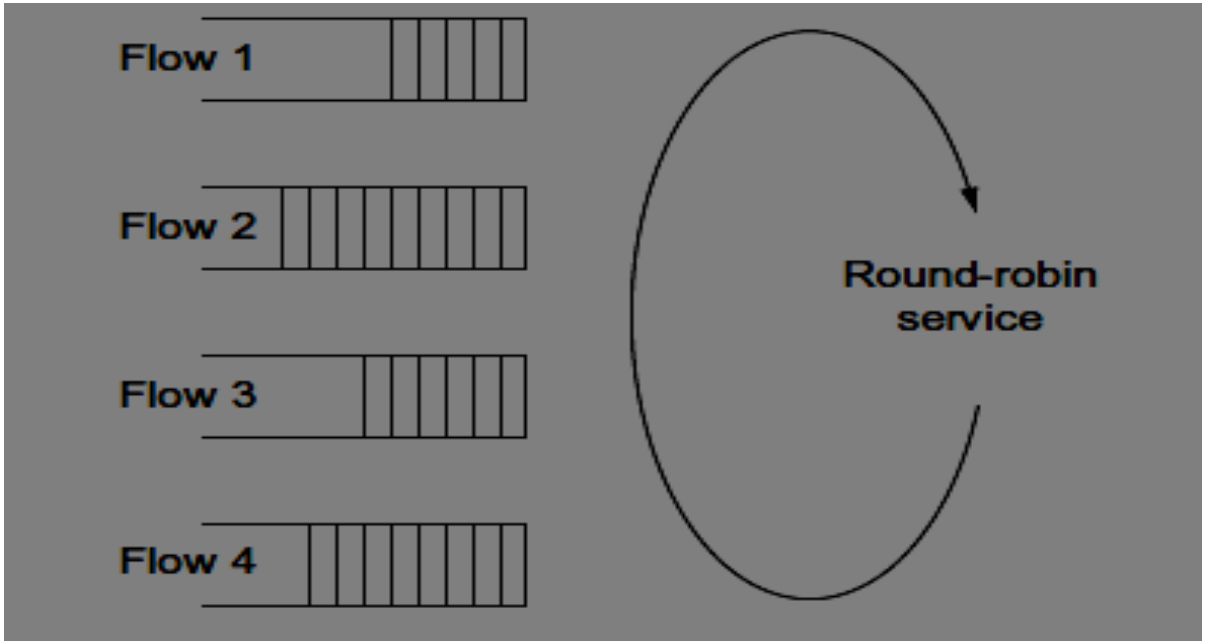
В Интернет не всички приложения са TCP-базирани, т.е. не ползват механизмите за управление на потока.

И е възможно да задръстят рутерите.

Тук на помощ идват “справедливите” опашки (Fair queuing – FQ): поддържа отделна опашка за всеки поток през рутера.

Рутерът ги обслужва в цикличен ред (round-robin).

Round-robin



Когато даден поток изпраща много бързо, опашката се запълва и при определен праг пакетите започват да се изхвърлят.

Така съответният източник не може да запълни мрежата за сметка на други потоци.

Weighted **F**air **Q**ueuing

Вариант на FQ е WFQ – на всеки поток (опашка) се присвоява тежест (weight) – колко бита да се предават, когато рутерът обслужва опашката.

Простата FQ дава тежест 1 на всеки поток, логически да предава само по 1 bit, когато дойде реда.

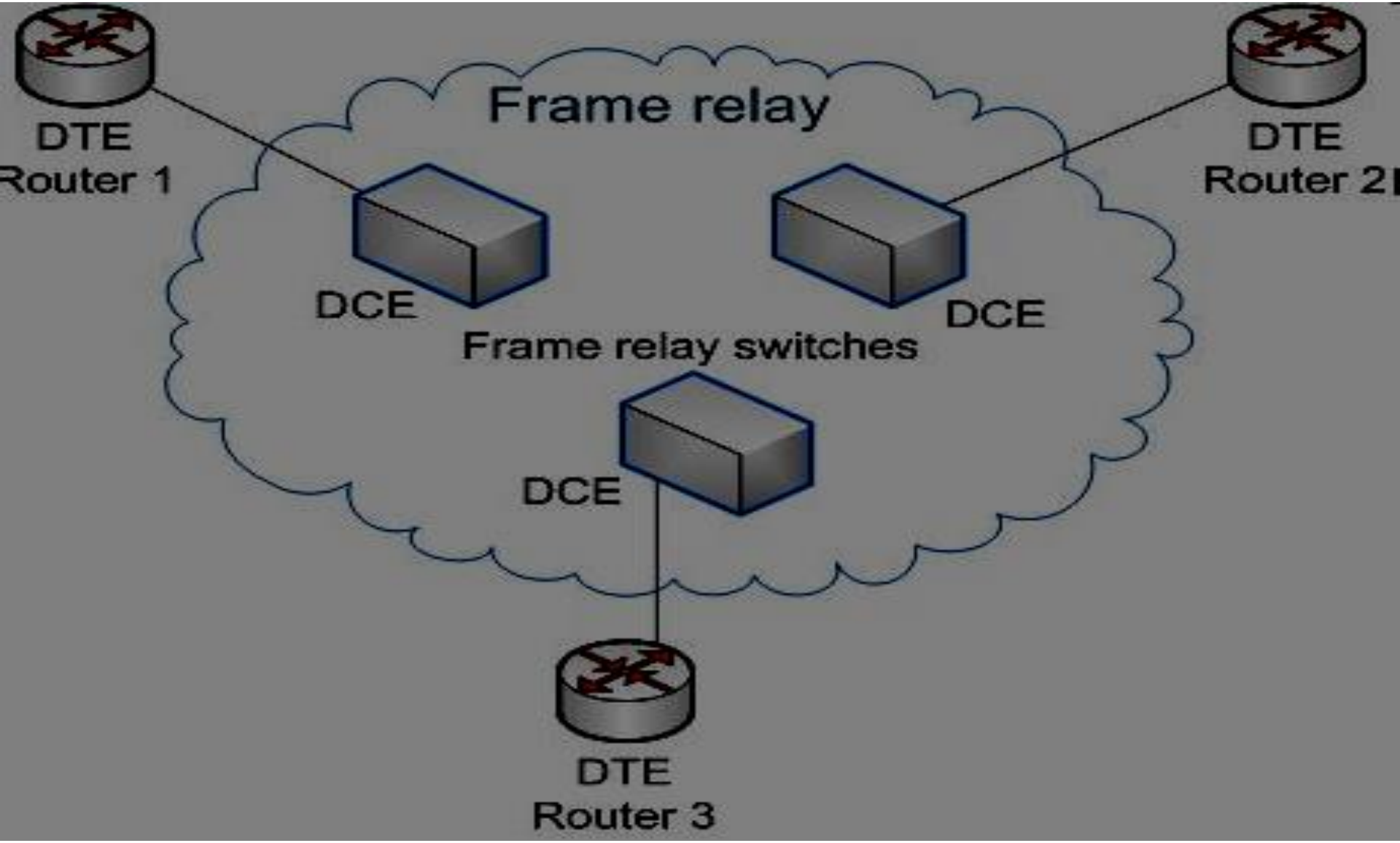
WFQ, ако тежестта е 2, ще предава 2 бита и т.н.

WFQ може да се приложи на “класове” от трафик

Например, полето TOS в IP хедъра идентифицира класовете и служи за алокиране на опашка и тежест за всеки клас.

Това е част от архитектурата Differentiated Services (DiffServ).

Frame Relay


Frame relay
DTE
Router 1
DTE
Router 2
DCE
DCE
Frame relay switches
DCE
DTE
Router 3

Frame Relay. Congestion Control на 2 слой.

Frame Relay комутаторите създават виртуални канали за свързване на отдалечени LANs или отделни машини. Нарича се Non Broadcast Multiple Access (NBMA), защото за потребителите е „облак“ - MA, но под него са комутаторите свързани „точка-точка“ в някаква топология - NB.

Евтина алтернатива на наетите линии (leased lines). Беше популярна преди появата на MAN.

В заглавието на кадъра (frame) има и следните битове, решаващи проблема със задръстванията (congestion control):

FECN=Forward Explicit Congestion Notification bit

BECN=Backward Explicit Congestion Notification bit

DE=Discard Eligibility bit

Frame Relay. Congestion Control.

Frame Relay мрежата е с опростен протокол, липсва flow-control връзка по канал. Затова производителността на мрежата като цяло е податлива на „скокове“ в натоварването от някои услуги - bursts и се нуждаят от механизми за congestion control – управление на задръстванията.

Congestion control включва следните елементи:

Admission Control. Мрежата решава дали да приеме нова заявка за връзка в зависимост от описания в заявката трафик (traffic descriptor), сравнявайки го с остатъчния капацитет. Traffic descriptor се състои от три компонента:

Frame Relay. Congestion Control.

- Committed Information Rate (CIR). Средната скорост (в bit/s), при която мрежата гарантира трансфера на информационни единици по време на интервал T:

  T = Bc/CIR.

- Committed Burst Size (Bc). Максималният брой предадени информационни единици през интервала T.
- Excess Burst Size (BE). Максималният брой непотвърдени информационни единици (в bits), които мрежата ще се опита да пренесе по време на интервала.

Frame Relay. Congestion Control.

След като мрежата е установила връзката, граничният възел трябва да следи трафика, така че да не се надвишават зададените стойности. Frame relay налага скоростта за крайните потребители и също така изхвърля информация, когато се надвиши заявената скорост.

Политиката на избягване на задръстванията включва непосредствено уведомяване. Така че да се постигне определено качество на услугата Quality of Service (QOS).

Frame Relay. Congestion Control.

За това служат специалните битове за контрол на задръстванията: FECN и BECN.

FECN - Forward Explicit Congestion Notification. Установява се в 1, когато се наблюдава задръстване в посоката на предаване на кадъра (frame), т.е. информира **дестинацията,** че има задръстване.

BECN - Backwards Explicit Congestion Notification. Установява се в 1, когато се наблюдава задръстване в посоката, обратна на предаване на кадъра (frame), т.е. информира **източника,** че има задръстване.

TCP Congestion Control

TCP Congestion Control (RFC 5681) е въведен в края на 1980-те от Van Jacobson.

Идеята на TCP CC е всеки източник да определя колко е свободния капацитет на мрежата, колко пакети може да изпрати.

Получаването на ACK е сигнал, че един пакет е напуснал мрежата и може да се въведе нов без опасност от задръстване.

Определянето на свободния капацитет не е лека задача, понеже има много потоци в мрежата.

TCP Congestion Window

Congestion Window (**cwnd**) ограничава количеството данни, които TCP може да изпрати. TCP не трябва да изпраща данни с:

`SeqNo > (AckNoMax + min(cwnd + rwnd)).`

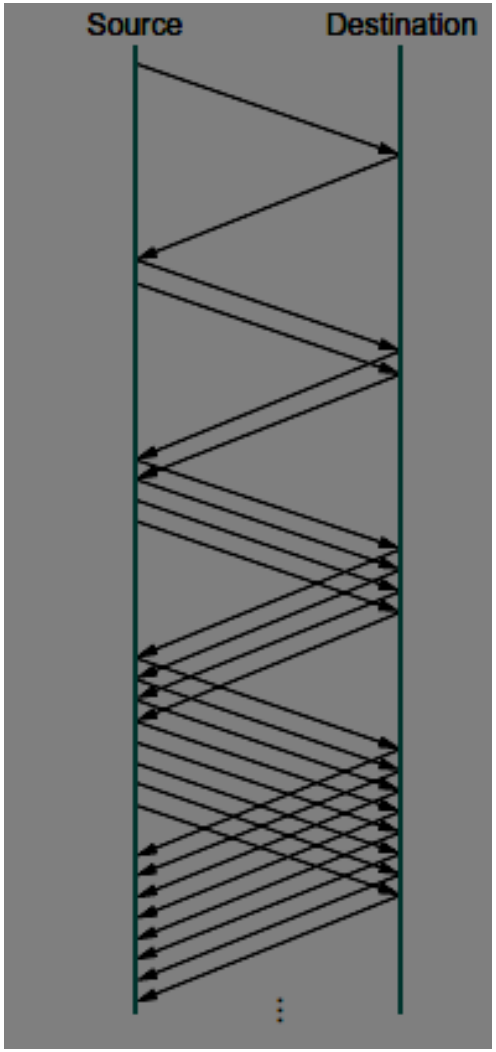
rwnd – Най-скоро рекламирания receiver window

Алгоритми за контрол на задръстването

RFC 5681 дефинира следните алгоритми: бавен старт, избягване на задръстването, бързо повторно предаване и бързо възстановяване.

Алгоритъм на бавния старт (slow start). Започвайки да предава при неизвестни условия, TCP бавно пробва мрежата, за да определи какъв е капацитета и да избегне задръстване с големи обеми от данни.

Бавен старт



Бавен старт. Начален прозорец.

IW, началната стойност на **cwnd**, приема една от следните стойности:

Ако SMSS (Sender MSS) > 2190 bytes:

IW = 2 * SMSS bytes

Ако (SMSS > 1095 bytes) и (SMSS <= 2190 bytes):

IW = 3 * SMSS bytes

Ако SMSS <= 1095 bytes:

IW = 4 * SMSS bytes

Прагът на бавния старт

Началната стойност на ssthresh (slow start threshold) се слага произволно висока (напр., възможно най-големият рекламиран прозорец), но при задръстване се редуцира.

При висока стойност на ssthresh условията в мрежата, а не някой хост ще определят скоростта.

След бавния старт

В RFC 5681 се препоръчва cwnd да се увеличава с min (N, SMSS),

N е броят на непотвърдените по-рано байтове, които са потвърдени от настоящия ACK.

Бавният старт приключва, когато cwnd надвиши прагово ниво ssthresh или при настъпване на задръстване.

Избягване на задръстването

При този алгоритъм cwnd се инкрементира с по един цял сегмент на всеки период round-trip time (RTT).

И продължава, докато се забележи задръстване.

Fast Retransmit/Fast Recovery

На следващия слайд е илюстрирано как дублирани ACKs водят до бързо повторно предаване.

Дестинацията получава сегменти 1 и 2, но сегмент 3 се губи и дестинацията ще изпрати дублиран ACK за сегмент 2, когато пристигне 4, после след 5 и т.н.

Изпращачът вижда третия дублиран ACK за пакет 2 – в отговор на сегмент 6, и повторно предава сегмент 3.

Когато в дестинацията пристигне повторно предадено копие на сегмент 3, приемникът изпраща към източника сумарен ACK за всичко получено до момента, включително сегмент 6.

Fast Retransmit/Fast Recovery