TCP ANALÍZIS DIFFSERV KÖRNYEZETBEN

TCP ANALYSIS IN DIFFSERV ENVIRONMENT

Lengyel Miklós, mlengyel@inf.unideb.hu
Sztrik János, jsztrik@inf.unideb.hu
Debreceni Egyetem, Informatikai Rendszerek és Hálózatok Tanszék

1. Bevezetés

A Differentiated Services (Diffserv vagy DS) architektúra [1] egy modellt definiál az Internetbeli szolgáltatások megkülönböztetésének skálázható implementálására. A szolgáltatás alatt itt a hálózat egy adott irányába történő csomag továbbítás lényeges karakterisztikáit értjük. Ilyen karakterisztika az adatátvitel, a késleltetés, illetve a csomagvesztés. A Diffserv architektúra egy sokkal rugalmasabb és skálázhatóbb modellt nyújt mint az eddigi megkülönböztetett szolgáltatások modellei (Integrated Services vagy RSVP). Azonban ezek az alternatív modellek használhatók a Diffserv architektúra kiterjesztésére illetve az architektúrával történő együttműködésre. A DS architektúra a skálázhatóságot úgy éri el, hogy a komplex klasszifikáló funkciókat csak a hálózat határán lévő csomópontokban implementálja és aztán a hálózat belsejében a megfelelően megjelölt csomagokra alkalmazza a hozzátartozó PHB-t (per-hop behaviour) [2,3]. A csomagok megjelölése az IPv4 vagy IPv6 fejlécben lévő DS mező értékének beállítását jelenti. Ez az architektúra csak a csomag folyam egyik irányába nyújt szolgáltatás differenciálódást vagyis aszimmetrikus.

Jelen cikkben egy egyszerű súlyzó Diffserv hálózati topológiát vizsgálunk meg szimuláció segítségével; hatékonysági összehasonlítást végzünk (késleltetést, sorhosszt valamint adatátvitelt figyelembe véve) három hagyományos csomag ütemező algoritmus esetén. Ezen ütemezők a: Priority (PRI), Weighted Round Robin (WRR) és Weighted Interleaved Round Robin (WIRR). Random Early Detection-t (RED) használunk aktív sor menedzselő algoritmusként, a szállítási protokoll a Transmission Control Protocol (TCP). Összevetjük az eredményeket korábbi cikkünk [4] eredményeivel is.

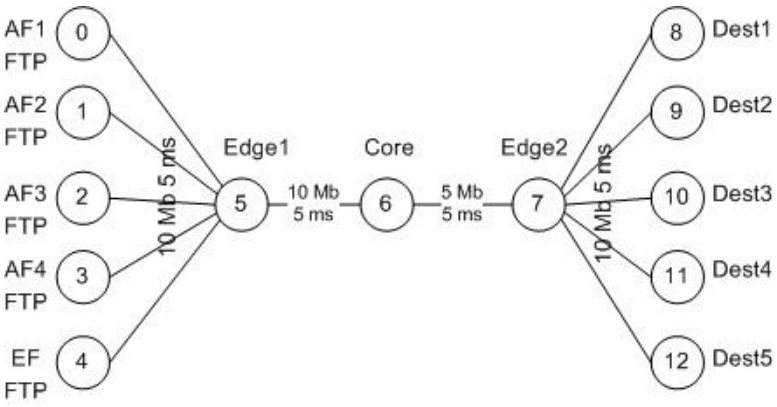
A cikk újdonsága ezen összehasonlítás, ugyanis a szerzők legjobb tudomása szerint a korábbi cikkekben csak egyedi ütemezőket vizsgáltak.

A második fejezet mutatja be a szimulációs eredményeket.

2. Szimulációs eredmények

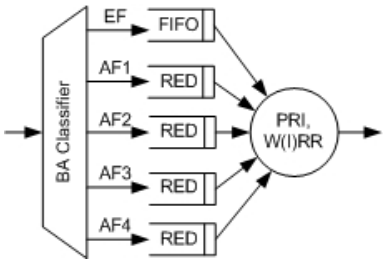
Szimulációnk során a Network Simulator (NS, verzió 2.25a) programcsomagot [5] használtuk, mely a Kaliforniai Egyetem került kifejlesztésre. Az NS egy eseményvezérelt hálózati szimulátor, mely C++-ban készült és OTcl (Object Tool Command Language) nyelvet használ parancs és konfigurációs interfészként. Az 1. ábrán látható egyszerű súlyzó

hálózati topológiát vizsgáltuk meg. Minden link 5 ms-os késleltetéssel rendelkezik. A vizsgálatot a Core csomópontban végeztük el, itt található a szűk keresztmetszetű adatátviteli vonal. A Core csomópont kimeneti interfészének a szerkezetét a 2. ábra mutatja.



1. ábra

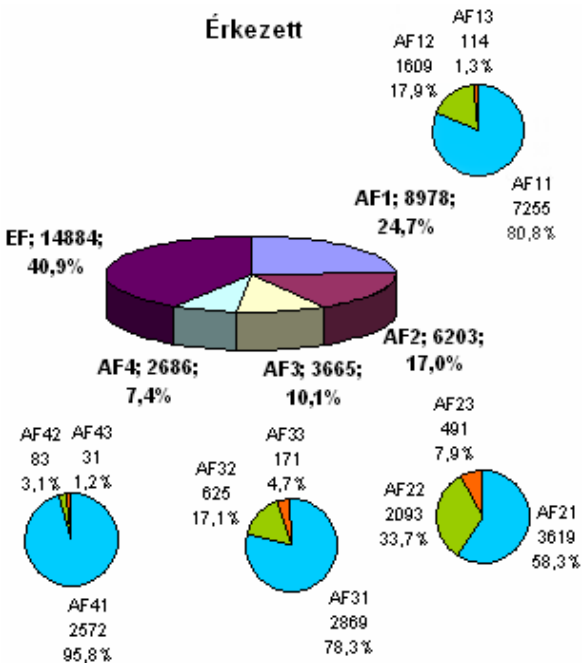
Szimulált hálózati topológia



2. ábra

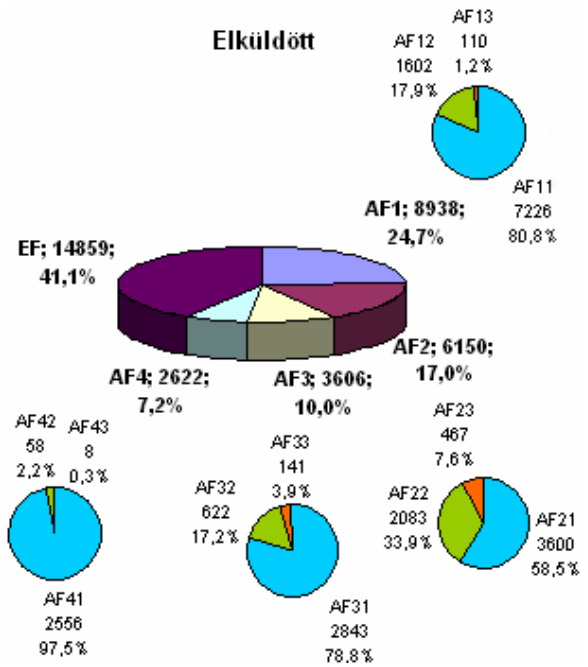
Kimeneti interfész

A szimulációkat 60 másodpercig futattuk. Minden forgalomgenerálónk FTP, melyek alatt TCP Sack implementációt állítottunk be. A 0-3 csomópontok AF1-AF4 forgalmat generálnak, míg a 4-es csomópont EF forgalmat. Az i.-ik csomópont az i+8.-ik csomópontnak küld adatot, i = 0,1,2,3,4. Egy AF osztály egy csomópontban egy olyan fizikai RED sorral van implementálva melynek három virtuális sora van, míg az EF osztály egy FIFO sorral kerül implementálásra.



3. ábra

Érkezett csomagok eloszlása



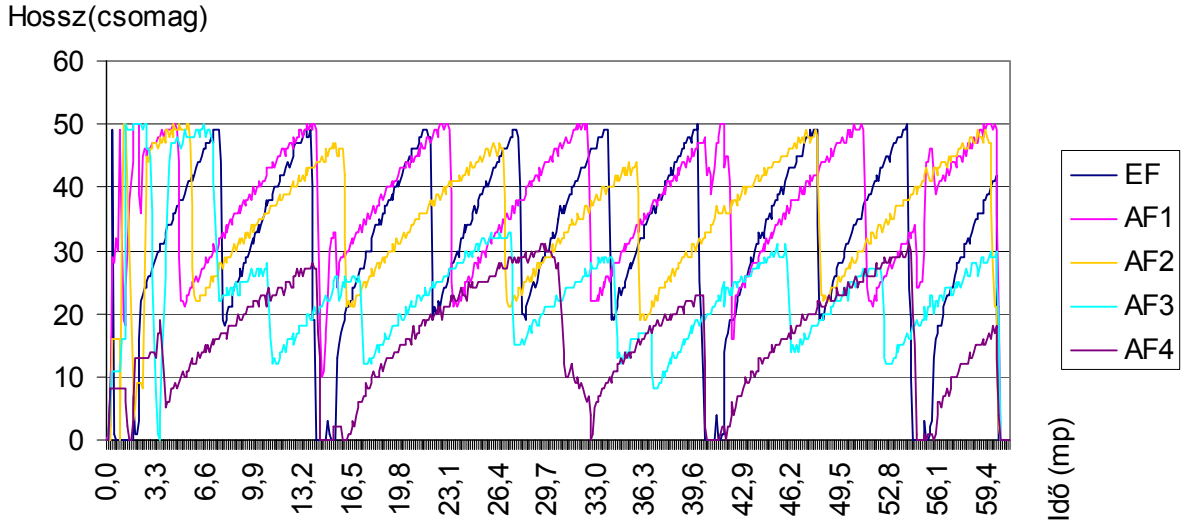
4. ábra

Elküldött csomagok eloszlása

A 3. és 4. ábrán látható a Core csomópontokba megérkező illetve távozó csomagok százalékos eloszlása és azok darabszáma. Ezen eloszlások az ütemező (PRI, W(I)RR) paramétereitől (QueueRate, QueueWeights), az osztályozó (TSW3CMPolicer - Time Sliding Window with 3 Color Marking Policer) CIR (Committed Information Rate) és PIR (Peak Information Rate) paramétereitől [5], valamint a 4 darab RED sor MinTh (Mininum Threshold), MaxTh (Maximum Threshold) és MaxP (Maximum Probability) paramétereitől [6] függenek. Látható, hogy a beérkező és a távozó eloszlások között nincs nagy különbség, melynek oka az, hogy a TCP protokoll visszafogja a küldő sebességét, ha torlódást, ütközést észlel a hálózaton és emiatt az érkező csomagok száma nem sokkal haladja meg a hálózati kapacitást. Előző cikkünkben [4] jelentős eltérés jelentkezett a szóban forgó két eloszlás között, melynek oka volt, hogy UDP protokollt és CBR (Constant Bit Rate) forrásokat alkalmaztunk, melynek következtében folyamatosan generálódott a forgalom a hálózaton, akkor is ha torlódás volt.

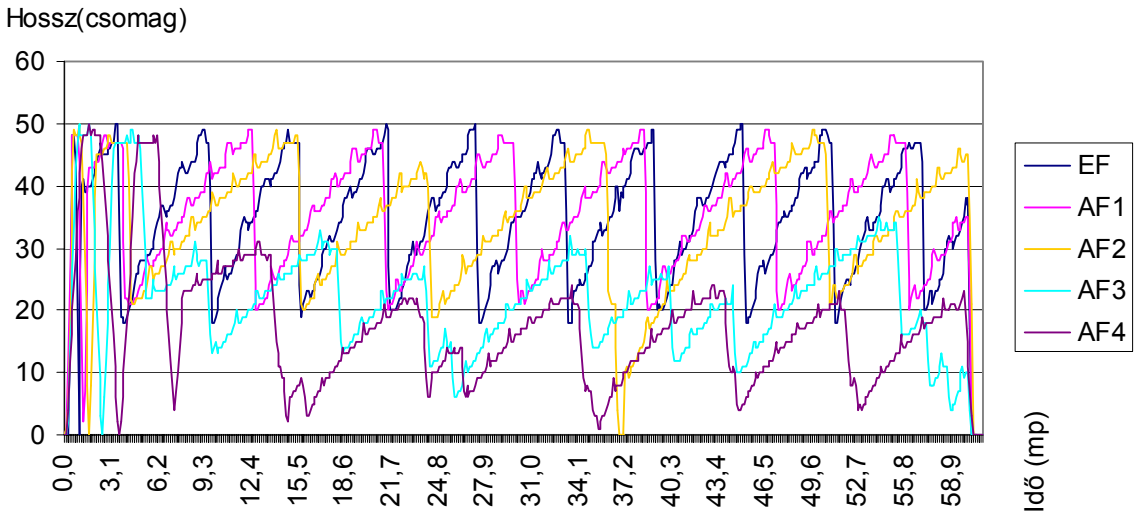
A szimulációkat úgy állítottuk be, hogy a fent említett eloszlások mindhárom ütemező esetén ugyanazok legyenek. Tehát a három szimuláció egymástól csak a használt ütemezőben (PRI, WRR, WIRR) és annak paraméterezésében különbözik. A kísérleteink során 1000 byte-os csomagokat használtunk, a csomópontokban lévő sorok hossza pedig 50 csomag.

Először vizsgáljuk meg, hogyan változik a Core csomópontban a sor hossz a három különböző ütemező esetén. Az alábbi három ábra ezt kívánja szemléltetni. A sor hossz változása mind az öt osztály (EF, AF1-AF4) esetében a RED működésének megfelelően történik. Ez mindhárom ütemező esetén igaz. A PRI ütemező esetén a sor hossz átlagos eltérésének és a sor hossz átlagának aránya az EF és AF4 osztályok esetén nagyobb, míg az AF1, AF2 és AF3 osztályok esetén kisebb mint a WRR ütemező esetén. Ebben a tekintetben a WIRR ütemező a PRI-hez képest ugyanúgy viselkedik mint a WRR, csak az eltérések fokai mások.



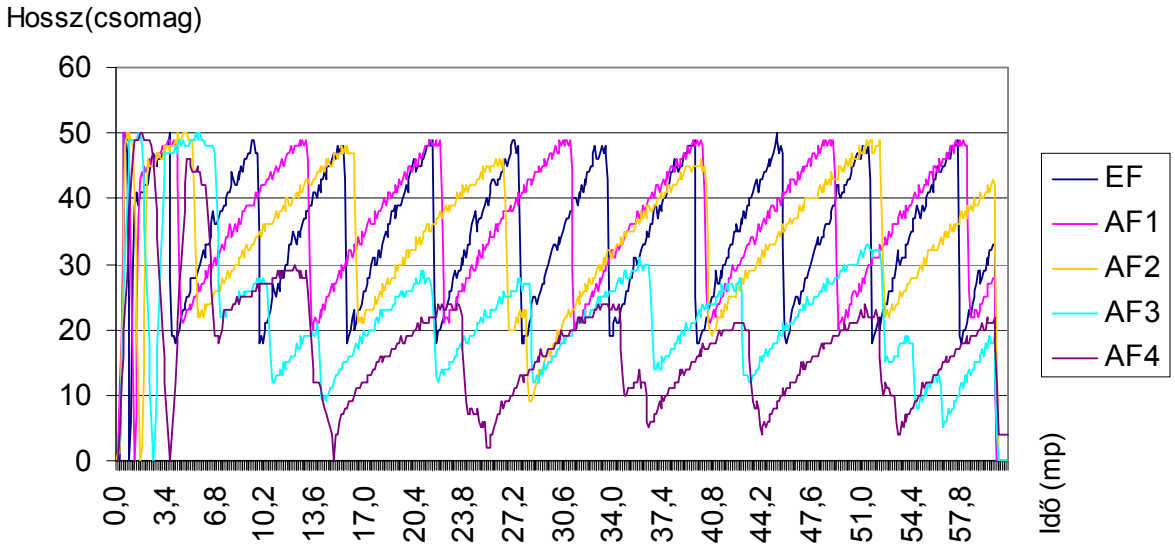
5. ábra

A sor hossz változása PRI ütemező esetén



6. ábra

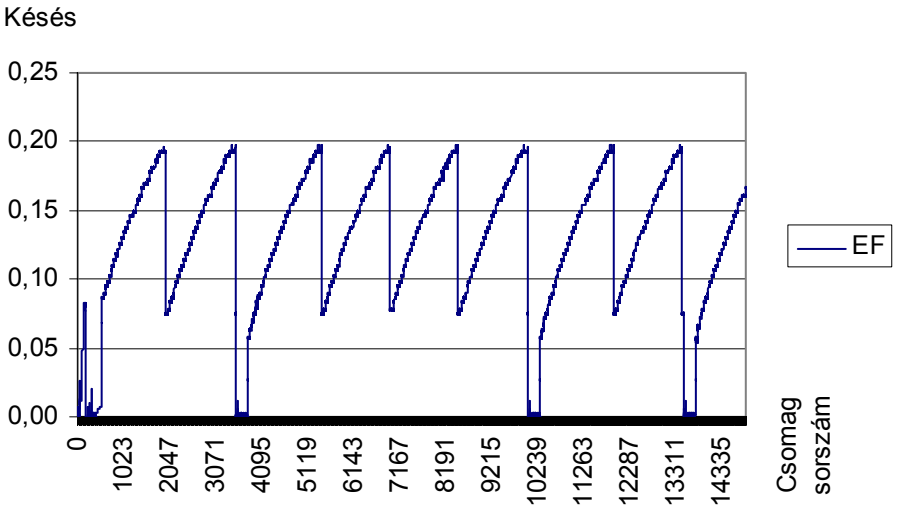
A sor hossz változása WRR ütemező esetén



7. ábra

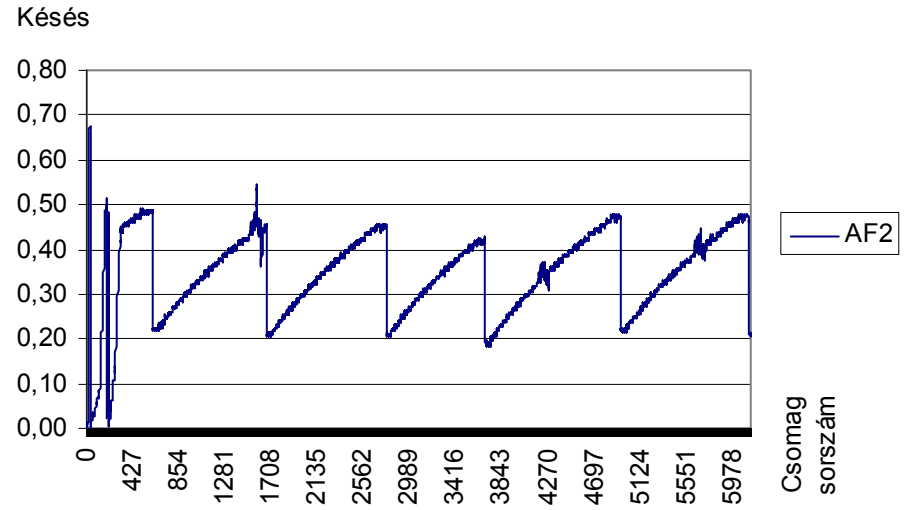
A sor hossz változása WIRR ütemező esetén

A következő két ábrán a csomagok késésének változását látjuk. A késés ugyanúgy változik ahogy a sorhossz változik, amely megfelel a jól ismert Little-formulának ($Q = \lambda * W$). Ez azt jelenti pontosan, hogy a sor hossz és a késleltetés aránya egy konstans érték. A cikk korlátozott oldalszáma miatt csak a PRI ütemező EF és AF2 osztályainak késleltetését mutatjuk be, de a fenti feltétel mind az öt osztály csomagjaira igaz mindhárom ütemező esetében.



8. ábra

Az EF csomagok késleltetése a PRI ütemező esetén

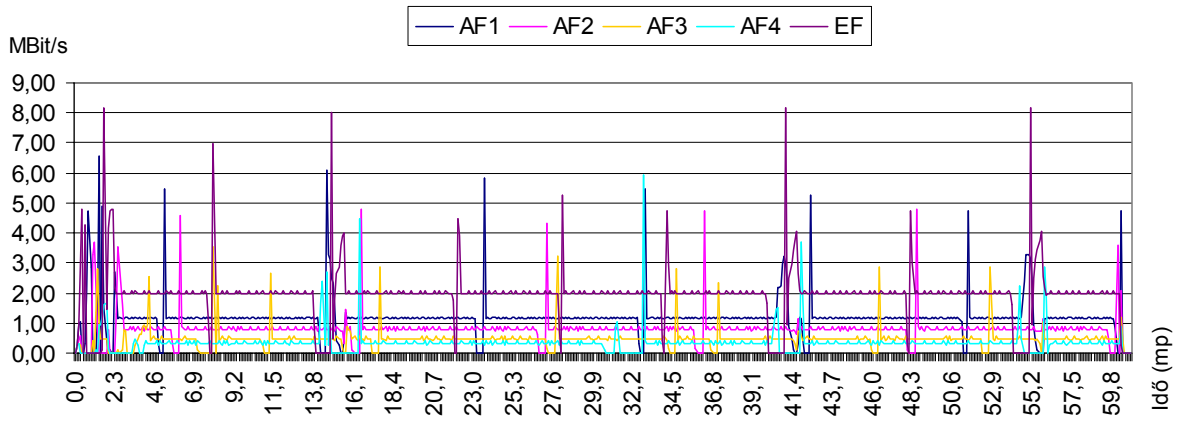


9. ábra

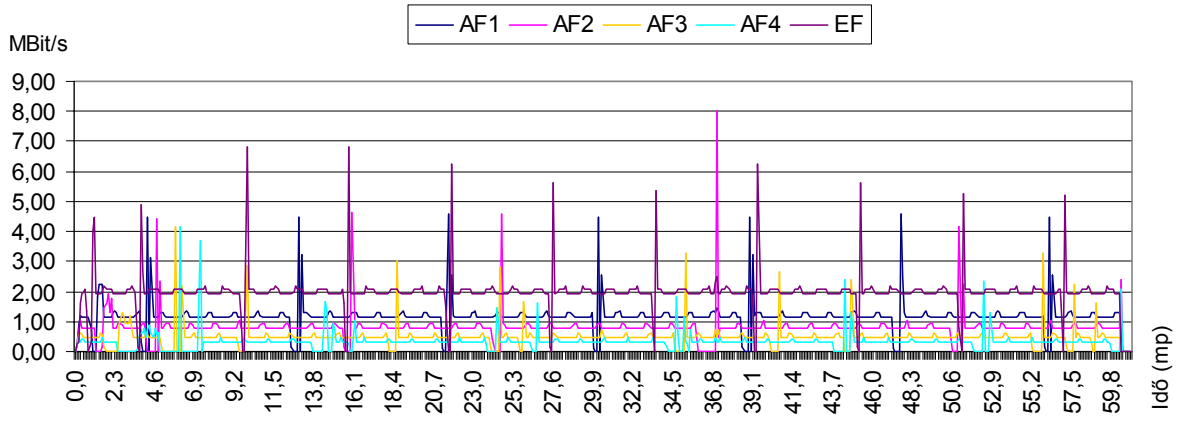
Az AF2 csomagok késleltetése a PRI ütemező esetén

Végül vizsgáljuk meg az adatátvitel változását. Az osztályonkénti átlagos realizált adatátvitel mindhárom ütemező esetén hasonló. Az adatátvitel átlagos eltérése sem mutat szignifikáns különbséget az ütemezők között. A terjedelmi megszorítások miatt most csak a PRI és WRR ütemezők adatátviteli grafikonját prezentáljuk.

A 10. ábra az adatátvitel változását mutatja be a PRI ütemező esetén, míg a 11. ábra a WRR ütemező esetén prezentálja ezt az adatsort.



10. ábra



11. ábra

Irodalomjegyzék

[1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, "An Architecture for Differentiated Services", *IETF RFC 2475*, 1998 december.

[2] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, "Assured Forwarding PHB Group", *IETF RFC 2597*, 1999 június.

[3] B. Davie, A. Charny, J. Bennett, K. Benson, J. Boudec, W. Courtney, S. Davari, "An Expedited Forwarding PHB", *IETF RFC 3246,* 2002 március.

[4] M. Lengyel, J. Sztrik, "Performance comparison of traditional schedulers in Diffserv architecture using NS", *16th European Simulation Symposium: Simulation in Industry, Budapest, Magyarország,* 2004.

[5] K. Fall, K. Varadhan, "The ns Manual", *http://www.isi.edu/nsnam/ns/ns-documentation.html*, 2002 április.

[6] B. Braden, D. Clark, B. Davie, S. Floyd, V. Jacobson, J. Wroclawski, L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet", *IETF RFC 2309*, 1998 április.