Szakdolgozat

Czipták Krisztián

Debrecen

2008

Debreceni Egyetem

Informatika Kar

Aktuális hálózati problémák megoldásainak vizsgálata

Quality of Service - QoS

Témavezető:

Dr. Almási Béla

Egyetemi docens

Készítette:

Czipták Krisztián

Mérnök Informatikus

Debrecen

2008

Tartalomjegyzék:

1. Bevezetés:

A diplomamunkám célja, hogy bemutassa az informatikai hálózatok fejlődésével, és a felhasználók számának növekedésével járó aktuális hálózati problémákat és ezekre a problémákra lehetséges megoldások közül részletesen ismertetni a minőségi hálózati szolgáltatás főbb megvalósításait, protokolljait, szabványait.

Az internet az amerikai egyetemi és kormányzati kutatói közösségben alakult ki a 70-es és 80-as években, annak igényeit elégítve ki, ezek rövid aszinkron üzenetek voltak (E-mail), ezek átvitele egyszerű volt, azóta új alkalmazások jelentek meg: pl.: www, VoIP stb., amelyek új követelményeket is jelentettek az adatátviteli kapacitásban, a szolgáltatás minőségében és az adatvédelemben (hitelesség, titkosítás, ...). A www sikere, legalább is részben annak volt köszönhető, hogy elősegítette nemcsak egyszerű szöveg, hanem űrlapok és viszonylag egyszerű grafika átvitelét is olyan módon, ami független volt mind a szerver, mind a kliens gép architektúrájától.

Azóta számos alkalmazást fejlesztettek ki, ami nagyon is valós grafikát igényel. Ezekhez az új alkalmazásokhoz megfelelő sávszélesség kell, hogy ezeket hatékonyan futtathassuk. Az Internet megnövekedett fizikai méretei és az új alkalmazások sávszélesség igényei is új megoldásokat kívánnak.

A felhasználók részéről: akik megbízható, jó minőségű hálózatot szeretnének az alkalmazásaikhoz, ill. az új alkalmazások (videó konferencia, VoIP, tudományos eredmények megjelenítése, orvosi diagnosztika, stb.) különleges minőségi követelményeket támasztanak az adatátvitellel szemben.

Az internet alapvetően csak azt garantálja, hogy az adatcsomagokat legjobb tudása szerint (best effort) továbbítja a célállomásra. Ez nem megfelelő az olyan szolgáltatásoknál, ahol garantált maximális késleltetési idő, vagy időben egyenletes adattovábbításra van szükség (hang és mozgóképátvitel). A prioritásos adatforgalom, a többszintű szolgáltatásminőség, a torlódásmentesség, a hálózatvezérlés legjobban az operációs rendszer QoS módszerével oldható meg.

A szóba jövő legfontosabb minőségi mutatók a teljes vég-vég késleltetés, valamint ennek ingadozása (jitter) ill. ennek felső határa, továbbá az alkalmazás számára rendelkezésére álló sávszélesség és a csomagvesztés (adatvesztés).

Az QoS mindig globális (nem lehet beszélni egy adott kapcsolat minőségéről) és statisztikus, azaz csak átlagosan teljesülnek a minőségi mutatók. Ezen felül kiemelt minőséget csak a kommunikáció bizonyos részének a különleges, általában prioritásos kezelésével lehet nyújtani.

A hálózati multimédiás alkalmazások térnyerése a felhasználóknál magával vonja, hogy a hálózatok tervezésénél, üzemeletetésénél komoly erőfeszítéseket kell tenni annak érdekében, hogy garantált szolgáltatásminőséget érjünk el. A hálózatok teljesítőképességeinek, átbocsátóképességének vizsgálatával és ezeket a hálózatokat használó alkalmazások kielégítő szolgálatminőség garanciáját szolgáltatja a QoS, amit az aktív elemek (router, switch) biztosítanak.

A munkám feltételez bizonyos jártasságot a hálózatok témakörben.

A QOS szolgáltatást az IEEE 802.1p szabvány támogatja.

Napjainkban a több megabites sávszélesség rendelkezésre állása miatt a QoS elenyésző szerepet tölt már be, a felhasználóknak adnak pl. egy 20Mbit/20Mbit-es kapcsolatot, ahol már kevésbé van szükség a QoS-re, mert ez így egy garantált, jó minőségű kapcsolat.

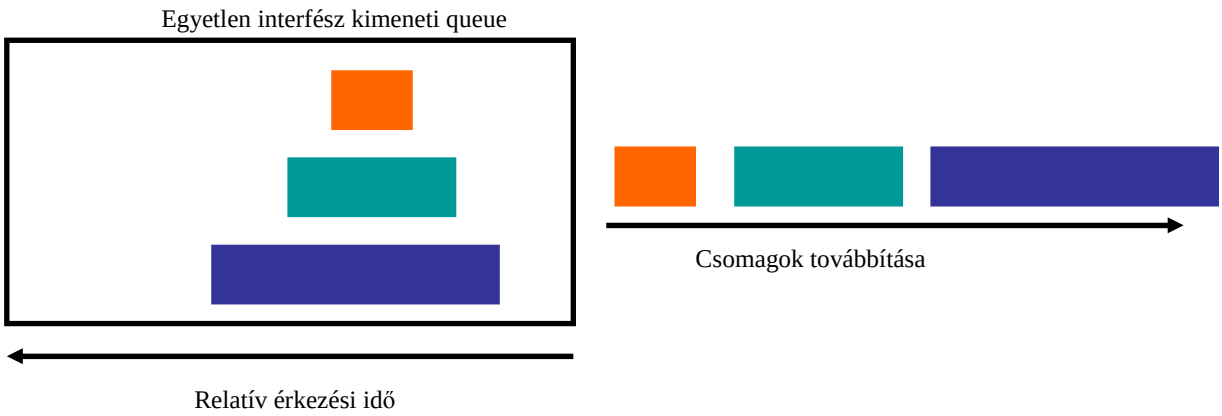
A szakdolgozatom a rétegelt architektúra hálózati rétegbeli szolgáltatás minőséget vizsgálja.

A dolgozatot a multimédiás alkalmazások QoS igényeinek tárgyalásával kezdem.

2. Quality of Service (QoS) alapjai, fogalmai, működése

2.1 A kezdet: A Best-effort

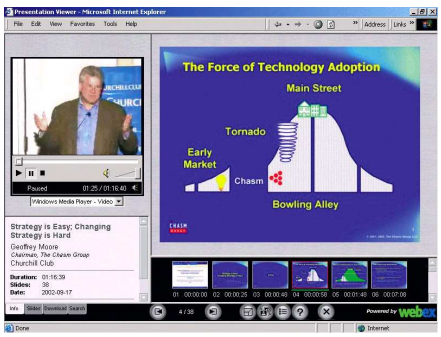
Az alkalmazás a hálózat megkérdezése vagy értesítése nélkül akkor és olyan mennyiségben küld adatot, amikor és amennyi számára szükséges [17]. A hálózat megbízhatósági, késleltetési és teljesítmény garancia nélkül, általában FIFO sorrendben kézbesíti a PDU-kat.

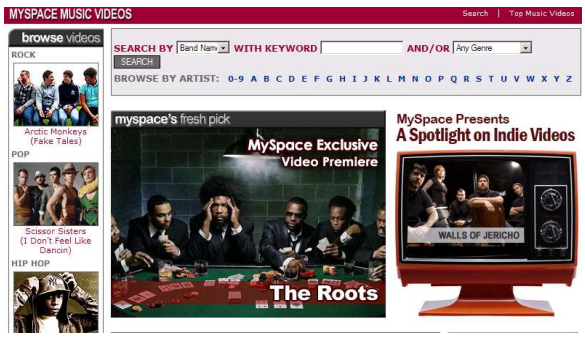


2.1 ábra

2.2 A multimédiás alkalmazások igényei

Egy forrásból egy adott célcsomópont felé tartó csomagok áramát folyamnak (flow) nevezzük. Egy összeköttetés alapú hálózatban az ugyanazon folyamhoz tartozó csomagok ugyanazt az útvonalat járják be, összeköttetés nélküli hálózatokban haladhatnak különböző utakon is. Az egyes folyamok igényeit alapvetően négy paraméterrel írhatjuk le: megbízhatóság, késleltetés, dzsitter és sávszélesség. Ezek együttesen határozzák meg a folyam által igényelt szolgáltatásminőséget [17].





A 2.2-es ábra a gyakori alkalmazásokat és azok követelményeit sorolja fel [15]:

Alkalmazás	**Megbízhatóság**	**Késleltetés**	**Dzsitter**	**Sávszélesség**
E-mail	Erős	Gyenge	Gyenge	Gyenge
Fájl átvitel	Erős	Gyenge	Gyenge	Közepes
Webes hozzáférés	Erős	Közepes	Gyenge	Közepes
Távoli használat	Erős	Közepes	Közepes	Gyenge
Hálózati zene	Gyenge	Gyenge	Erős	Közepes
Hálózati videó	Gyenge	Gyenge	Erős	Erős
Telefónia	Gyenge	Erős	Erős	Gyenge
Videókonferencia	Gyenge	Erős	Erős	Erős

2.2-es ábra.

Az első négy alkalmazás magas követelményeket támaszt a megbízhatósággal szemben. Ezeknél egyetlen bitet sem szabad hibásan átvinni. Ezt rendszerint úgy érik el, hogy ellenőrző összeggel látnak el minden csomagot, amit a cél csomópontnál ellenőriznek. Ha egy csomag megsérült az átvitel során, a nyugtában a hiányt küldi el, és ismétléssel elküldik újra. Az utolsó négy alkalmazásnál nem számítanak a kisebb bithibák.

A fájlátviteli alkalmazások, beleértve az e-mailt és a videózást is, nem érzékenyek a késleltetésre. Ha minden csomag egyformán néhány másodperc késleltetést szenved, az nem okoz problémát. A weben való szörfözés, távoli bejelentkezés és hasonló interaktív alkalmazások már érzékenyebbek a késleltetésre. A valós idejű alkalmazások (Telefónia, Videókonferencia) már kimondottan szigorú követelményeket támasztanak a késleltetéssel szemben. A hang és videó állományok egy kiszolgálóról történő lejátszása nem igényel alacsony késleltetést.

Az első három alkalmazás nem érzékeny arra, ha csomagok szabálytalan időközönként érkeznek be. A távoli bejelentkezés egy kicsit már kényesebb, mivel a karakterek apró ütemenként jelenhetnek meg a képernyőn, ha a kapcsolat nagyobb dzsitterrel terhelt. A videó és különösen a hang pedig rendkívül érzékeny a dzsitterre. Az nem okoz gondot, ha egy felhasználó egy filmet néz a hálózaton keresztül, és a keretek pontosan 1000 másodperc késleltetést szenvednek. De ha az átvitel ideje véletlenszerűen ingadozik 1 és 2 másodperc között, máris tragikus eredményt kapunk. A hangátvitelnél még néhány milliszekundomos dzsitter is tisztán hallható. Az alkalmazások végül a sávszélesség-igényeiket tekintve is

eltérnek egymástól, az e-mail és a távoli bejelentkezés nem sok sávszélességet használnak, de a videó bármely formája nagyon sokat (2.3 és 2.4-es ábra).

A videótovábbítás minőségének vélemény-érték (OS) metrikája [17]:

Dinamika	*Darabosság*	*Átlapoltság*	*Színhűség*	*Értelmezhetőség*	*Vélemény érték (OS)*
Merev	**Nagyon**	**Igen**	**Nem**	**Nem**	**1**
Akadozó	**Nagyon**	**Igen**	**Nem**	**Nem**	**2**
Akadozó	**Közepes**	**Igen**	**Nem**	**Kevésbé**	**3**
Akadozó	**Közepes**	**Igen**	**Nem**	**Közepes**	**4**
Akadozó	**Közepes**	**Igen**	**Nem**	**Közepes**	**5**
Mozgó	**Kevésbé**	**Nem**	**Nem**	**Közepes**	**6**
Mozgó	**Nem**	**Nem**	**Nem**	**Közepes**	**7**
Mozgó	**Nem**	**Nem**	**Nem**	**Elfogadható**	**8**
Mozgó	**Nem**	**Nem**	**Nem**	**Jó**	**9**
Mozgó	**Nem**	**Nem**	**Igen**	**Nagyon jó**	**10**

2.3-as ábra

Videótovábbítás minőségi jellemzői [17]:

2.4-es ábra

Az ATM-hálózatok QoS igényeik alapján 4 nagyobb csoportra sorolják a folyamokat, amelyek a következők:

1. Állandó bitsebesség (Telefónia)
2. Valós idejű, változó bitsebesség (Tömörített videókonferencia)
3. Nem valós idejű, változó bitsebesség (Filmet nézni az interneten)
4. Rendelkezésre álló bitsebesség (Fájlátvitel)

Ezek a kategóriák más hálózatokban, más célokra is hasznosak lehetnek. Az állandó bitsebességgel egy olyan vezetéket szimulálhatunk, mely változatlan sávszélességet és változatlan késleltetést biztosít. Változó bitsebesség például, akkor fordulhat elő, ha egy mozgóképet tömörítünk, és némely keret jobban tömöríthető, mint a többi, így egy nagyon részletgazdag keret elküldése sok bitet igényelhet, egy fehér fal képe viszont rendkívül jól tömöríthető. A rendelkezésre álló bitsebesség az olyan alkalmazásoknak felel meg, amelyek nem érzékenyek a késleltetésre vagy a dzsitterre, mint például az e-mail.

A multimédiás alkalmazások igényeinek a kielégítésére a következő részben bemutatott technológiákat használják.

2.3 A jó szolgáltatásminőséget biztosító megoldások

A dolgozat folytatásaként a lehetséges gyakorlati QoS technikákat mutatom be. Hogyan tudunk jó QoS-t implementálni? Önmagában egyetlen technika sem nyújt optimális módon hatékony, megbízható szolgáltatásminőséget. Mégis számos eljárást fejlesztettek már ki, melyek gyakorlati megoldásai sokszor több módszert is ötvöznek. A következőkben tekintsük át a rendszertervezők által a szolgáltatásminőség elérésére használt módszereket [1]:

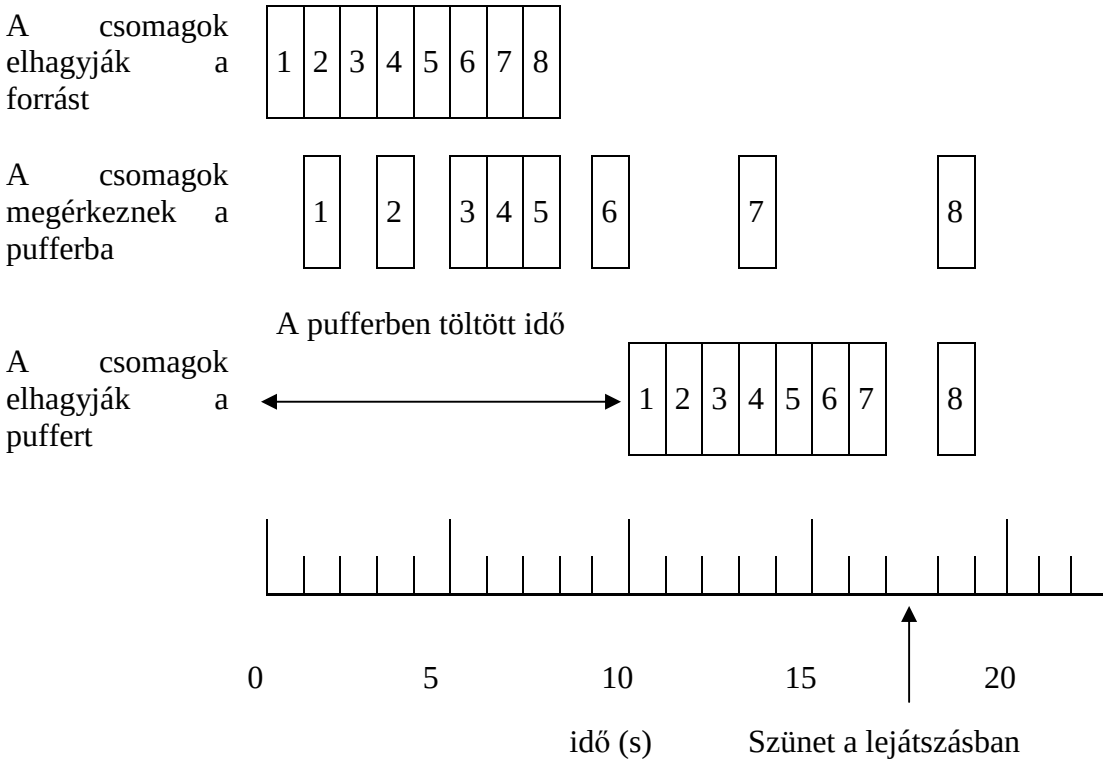
2.3.1 Túlméretezés

Egyszerű megoldásnak tűnik az, hogy annyi router kapacitást, puffer területet és sávszélességet biztosítunk, amennyivel a csomagok könnyedén átvihetők a hálózaton. Ennek a megoldásnak a hátránya,hogy drága. Ahogy azonban az idő halad és a tervezőknek több fogalmuk lesz arról, hogy valójában mennyi is az elég, ez a megoldás akár praktikussá is válhat. Bizonyos mértékig a telefonhálózat is túlméretezett. Ritkán fordul elő, hogy felvesszük a telefont és nem kapunk rögtön tárcsahangot (vonalat). Egyszerűen annyi a rendelkezésre álló kapacitás, hogy az igényeket mindig ki kell elégíteni.

2.3.2 Pufferelés

Az adatfolyamok kézbesítés előtt pufferelhetők a vételi oldalon. Ez nincs hatással a megbízhatóságra vagy a sávszélességre. A késleltetést növeli, de elsimítja a dzsittert. A hálózati hang- és mozgóképátvitel esetén a dzsitter okozza a legnagyobb problémát, ez a módszer tehát sokat segíthet.

A 2.5-ös ábra egy folyamot mutat, ahol a csomagok nagy dzsitterrel érkeznek meg. Az első csomagot a t = 0 s időpontban küldte el a kiszolgáló, és a t = 1 s időpontban érkezik meg a klienshez. A második csomag nagyobb késleltetést szenved, és 2 másodperc alatt érkezik meg. Beérkezésük után a csomagokat a kliens számítógépe puffereli. A lejátszás t = 10 s-nál kezdődik. Addigra az első 6 csomag bekerült a pufferbe, így ezeket egyenletes időközönként elővéve zökkenésmentes lesz a lejátszás. Sajnos a 8. csomag olyan sokat késett, hogy még, mindig nem áll rendelkezésre, amikor rákerülne a sor, így a lejátszásnak meg kell állnia, amíg beérkezik [1].



2.5-es ábra.

Ez kellemetlen szünetet okoz a zenében vagy a filmben. Ezt a problémát enyhítheti, ha még tovább késleltetjük a lejátszás kezdetét, bár ekkor nagyobb pufferre lesz szükségünk. A hang vagy videofolyamokat is tartalmazó kereskedelmi weboldalak ezért mind olyan lejátszókat használnak, amelynek körülbelül 10 s-os részeket tárolnak lejátszás előtt (2.6-os ábra) [15].

2.6 ábra

2.3.3 Forgalomformálás (Traffic shaping)

A fenti példában a kiszolgáló egyenlő időközönként adta le a csomagokat, de ez az adás lehet szabálytalan ütemű is, ami torlódást idézhet elő a hálózatban. Az általunk használt pufferelés bizonyos esetekben egyáltalán nem is alkalmazható. Ugyanakkor biztosan jobb szolgáltatásminőséget érhetnénk el, ha valaki rá tudná venni a kiszolgálókat (és hosztokat általában), hogy az erőforrásigényük jól meghatározott formában legyen, mivel a forgalom minden irányban egyenletes.

A forgalomformálás az adás átlagos sebességének és ütemszerűségének szabályozásáról szól. Amikor egy kapcsolat felépül, a kliens és a kiszolgáló megegyeznek egy, az adott áramkörre vonatkozó forgalommintázatban. Ezt szolgáltatásszintű megállapodásnak (Service level agreement) is hívjuk. A forgalomformálás csökkenti a torlódást, ezáltal segít a kiszolgálónak is, fájlátvitelnél ezek nem nagyon fontosak, de valós idejű átvitelnél fontos, például hang és videókapcsolatoknál, melyeknek szigorú szolgálatminőségi követelményeik vannak.

A forgalomformálásnál az ügyfél valójában azt mondja a kiszolgálónak: „Ilyen az átviteli mintám. Képes vagy kezelni?” Ha a kiszolgáló elfogadja, felmerül a kérdés, hogy meg tudja-e állapítani a kliensről valóban tudja-e fogadni, és mit tud tenni, ha kiderül, hogy nem. A forgalom haladásának, folyamának figyelését forgalmi rendfenntartásnak (traffic policing) nevezzük. Vonalkapcsolt alhálózatokban egyszerűbb egyeztetni, majd később fenntartani egy forgalomformát, mint csomagkapcsolt alhálózatokban. Ettől függetlenül a csomagkapcsolt alhálózatoknál is alkalmazhatjuk ugyanezeket az elgondolásokat a szállítási szintű kapcsolatoknál.

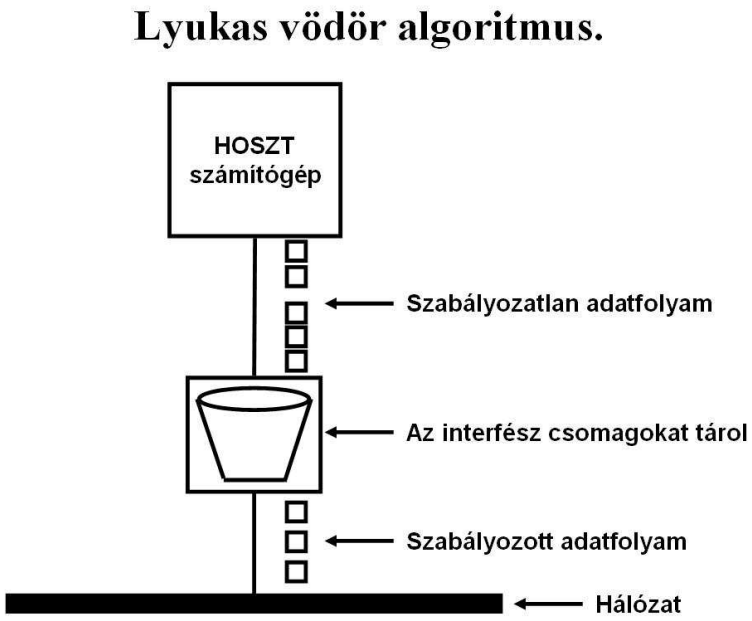
2.3.4 A lyukas vödör algoritmus

Képzeljünk el egy vödröt, az alján egy kis lyukkal, ahogy a 2.7-es ábra bal része mutatja. Mindegy, hogy a víz milyen sebességgel érkezik a vödörbe, a kimenő folyam konstans p sebességű, amikor van víz a vödörben, és nulla, amikor a vödör üres. Ha a vödör megtelt, a további érkező víz kicsordul és elveszlik.

Ugyanez az ötlet alkalmazható csomagokra is, ahogy a 2.7-es ábra jobb része mutatja [11]. A felfogás szerint minden kliens egy lyukas vödröt, vagyis egy véges belső sort tartalmazó

interfészen keresztül kapcsolódik a hálózathoz. Ha egy csomag akkor érkezik a sorba, amikor tele van, akkor a csomagot eldobják. Más szavakkal, mikor egy vagy több folyamat a hoszton belül akkor próbál csomagot küldeni, amikor már a sor betelt, az új csomagot minden felhajtás nélkül eldobják. Ez az elrendezés beépíthető a hardver interfészbe vagy szimulálható a hoszt operációs rendszere által. Ezt lyukas vödör algoritmusnak (leaky bucket) hívják. A gyakorlatban ez nem más, mint egy egykiszolgálós sorban állási rendszer állandó kiszolgálási idővel.

A hoszt óraütésenként egy csomagot tehet a hálózatra. Ezt megint csak az interfészkártya vagy az operációs rendszer kényszerítheti ki. Ez a mechanizmus a hoszton belüli felhasználói folyamatoktól eredő egyenetlen csomagfolyamot egyenletes csomagfolyamként adja a hálózatra, kisimítva a lökéseket és nagyban csökkentve a torlódás esélyét.



2.7-as ábra.

Amikor a csomagok mind azonos méretűek (pl. ATM cellák), akkor az algoritmus a leírt formájában használhatjuk. Viszont amikor változó méretű csomagokat használunk, gyakran jobb óraütésenként egy rögzített számú bájtot megengedni, mint egy csomagot. Így a szabály 1024 bájt/óraütés, akkor egy 1024 bájtos csomagot, két 512 bájtos csomagot, négy 256 bájtost stb. Ha a fennmaradó bájtszám túl kicsi, akkor megvárja a következő óraütést.

Az eredeti lyukas vödör algoritmust könnyű megvalósítani. A lyukas vödör egy véges sorból áll. Amikor egy csomag érkezik, ha van hely a sorban, hozzá illesztik, egyébként eldobják. Minden óraütéskor egy csomagot veszünk át, hacsak nem üres a sor.

A bájtszámolós lyukas vödröt majdnem ugyanígy valósítjuk meg. Minden óraütéskor a számlálót n-re állítjuk. Ha a sorban első csomagnak kevesebb bájtja van, mint a számláló jelenlegi értéke, átvisszük, és a számlálót a bájtok számával csökkentjük.

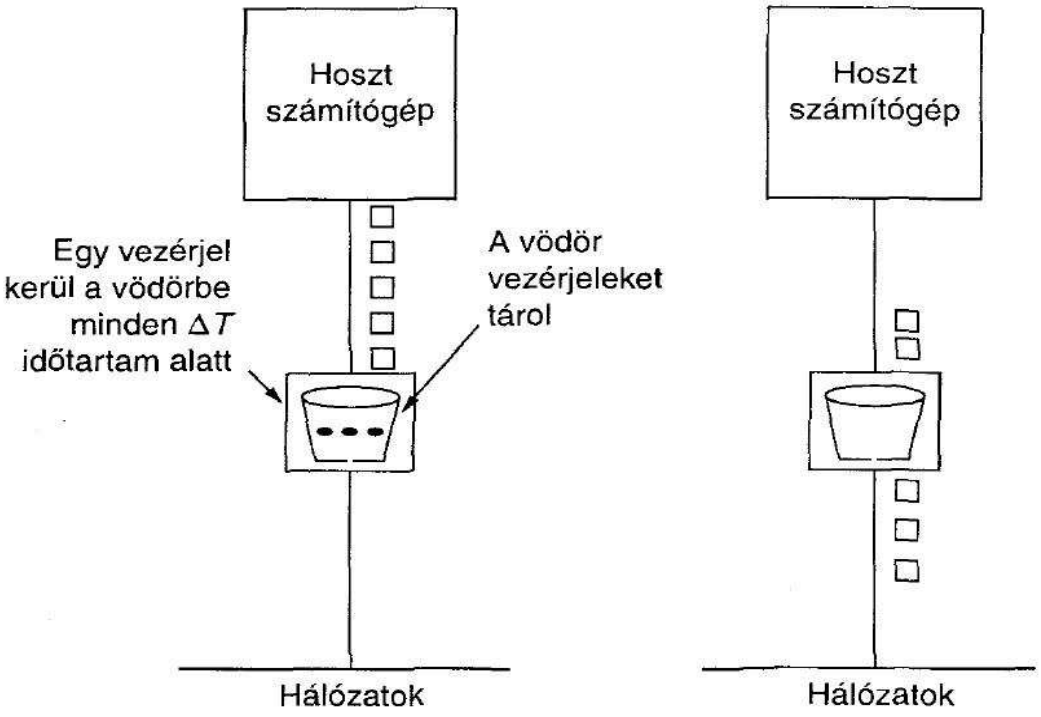
További csomagokat is küldhetünk, amíg a számláló értéke elég nagy. Amikor a számláló a sorban következő csomag hossza alá csökken, az átvitel megáll a következő óraütésig, amikor is felülírjuk és elveszítjük a fennmaradó bájtszámot.

2.3.5 A vezérjeles vödör algoritmus

A lyukas vödör algoritmus merev kimeneti mintát kényszerít az átlagos sebességre, a forgalom lökéseitől függetlenül. Sok alkalmazásnak jobb, ha a kimenetet engedjük gyorsulni valamelyest, amikor nagy ütések érkeznek, így egy rugalmasabb algoritmusra van szükség, lehetőség szerint olyanra, amely soha nem veszt adatot. Egy ilyen algoritmus a vezérjeles vödör (token bucket) algoritmus. Ebben az algoritmusban a lyukas vödör vezérjeleket tartalmaz, amelyeket egy óra állít elő egy vezérjel / ΔT sebességgel. A 2.8-as ábra bal részén egy vödröt láthatunk, amelyben három vezérjel van, és öt csomag vár továbbításra. Hogy egy csomag továbbítható legyen, el kell fognia és megsemmisítenie egy vezérjelet. A 2.8-as ábra bal részén láthatjuk, hogy három csomag keresztüljutott az ötből, de a másik kettő megrekedt, két további vezérjel előállítására várva [11].

A vezérjeles vödör algoritmus másfajta forgalomalakítást biztosít, mint a lyukas vödör algoritmusa. A lyukas vödör algoritmus nem engedi a tétlen hosztoknak, hogy az engedélyeket félretegyék, és később nagy ütéseket küldjenek. A vezérjeles vödör algoritmus megengedi ezt a félretevést, a vödör maximális méretéig, n-ig. Ez a tulajdonság azt jelenti, hogy akár n csomagból álló lökések is küldhetők egyszerre, amely megenged némi lökésszerűséget a kimeneten és gyorsabb választ ad a bemenet hirtelen lökéseire.

További különbség a két algoritmus között az, hogy a vezérjeles vödör eldobja a vezérjeleket, ha a vödör megtelik, de sosem dob el csomagokat. A lyukas vödör algoritmus ellenben a csomagokat is eldobja, ha megtelik a vödör [20].



2.8-as ábra.

Itt is elképzelhető egy kis módosítás, ha úgy vesszük, hogy a tokenek nem egy csomag, hanem k bájt elküldésére jogosítanak fel. Ilyenkor csak akkor lehet küldeni csomagot, ha van annyi tokenünk, amennyi lefedi a csomag hosszát bájtokban. A maradék tokeneket meg lehet őrizni későbbi felhasználásra.

Mind lyukas vödör, mind vezérjeles vödör algoritmusokat felhasználhatjuk a routerek közti forgalom egyenletesebbé tételére, valamint a hosztok kimenő forgalmának szabályozására is, amint azt a példában láthattuk. Van azonban egy jelentős különbség is a két felhasználás között. Egy hosztot szabályozó vezérjeles vödör nyugodtan leállíthatja a hoszt forgalmazását, ha a szabályok ezt indokolják. Ha viszont a routernek mondjuk, hogy álljon le a küldéssel, miközben a bemenetet elárasztja a bejövő forgalom, az eredmény adatvesztés lehet.

A vezérjeles vödör alapvető algoritmusának megvalósítása csak egy változó, amely a vezérjeleket számlálja. A számlálót eggyel növeljük ΔT időközönként, és eggyel csökkentjük, amikor csomagot küldünk. Amikor a számláló eléri a nullát, nem küldhetünk csomagot. A bájtszámlálásos változatban a számlálót k bájttal növeljük minden ΔT-ben, és mindegyik elküldött csomag hosszát kivonjuk belőle.

Egy lehetséges probléma a vezérjeles vödör algoritmusával, hogy ismét megenged nagy ütemeket, még ha a maximális ütem időtartama szabályozható is. Gyakran kívánatos csökkenteni a csúcssebességet, de a lyukas vödör alacsony értékéhez való visszatérés nélkül.

Sok fejtörést okozhat azonban, ha az összes ilyen sémát fenn akarjuk tartani. A hálózatnak alapjában véve követnie kell az algoritmust, és meg kell győződnie arról, hogy az engedélyezettnél több csomag vagy bájt nem kerül-e elküldésre. Ezek az eszközök ugyanakkor lehetőséget nyújtanak arra, hogy hálózati forgalmat kezelhetőbb formára hozzuk, ezzel is hozzájárulva a szolgáltatásminőségi követelmények kielégítéséhez.

2.3.6 Erőforrás-lefoglalás

A szolgáltatásminőség garantálásához jó kezdet, ha szabályozni tudjuk a felkínált forgalom alakját. Persze, ennek az információnak a hatékony felhasználása implicit módon azt is jelenti, hogy megköveteljük, hogy minden csomag ugyanazt az útvonalat kövesse. Nehéz lenne bármit is garantálni úgy, hogy a csomagokat véletlenszerűen szórjuk szét a különböző routerek között. Következésképp, a forrás és a cél csomópont között valamilyen, a virtuális áramkörhöz hasonló kapcsolatot kell létrehozni, és a folyamhoz tartozó összes csomagnak ezt az útvonalat kell követnie.

Amint megvan a folyam kijelölt útja, lehetővé válik, hogy az út mentén erőforrásokat foglaljunk le azért, hogy biztosan rendelkezésre álljanak a szükséges kapacitások. Háromféle erőforrást lehet lefoglalni:

1. Sávszélességet
2. Pufferterületet
3. Processzoridőt

A sávszélesség kérdése a legkézenfekvőbb. Ha egy folyamnak 1MBájt/s sávszélességre van szüksége, és a kimeneti vonal kapacitása 2MBájt/s, akkor azon a vonalon nyílván nem fogunk tudni három ilyen folyamot átvezetni. A sávszélesség lefoglalása tehát annyit jelent, hogy nem foglaljuk túl a kimenti vonalakat.

A második erőforrás, amiből gyakran hiányt szenvedünk, a pufferterület. Amikor egy csomag megérkezik, a hardware általában a hálózati illesztőkártyán helyezi el azt. A router szoftverének innen át kell másolnia egy, a memóriában lévő pufferterületre, és az adott puffert adáshoz sorba állítani a kiválasztott kimeneti vonalon. Ha nincs szabad puffer, a csomagot el kell dobni, mivel egyszerűen nincs hová tenni. A jobb szolgálatminőség érdekében néhány puffert fenntartanak meghatározott folyam számára, hogy az adott folyamnak ne kelljen a pufferért versengenie a többiekkel. Ily módon egy bizonyos határig mindig lesz szabad puffer, amikor a folyamnak szüksége van rá.

Végül a processzoridő is szűkös erőforrásnak számít. A routerek minden csomag feldolgozásához bizonyos processzoridőre van szüksége, így egy másodperc alatt csak korlátozott számú csomag feldolgozására van lehetőség. Ahhoz, hogy minden csomagot kellő időben feldolgozhassunk, biztosítanunk kell, hogy a processzor ne legyen túlterhelve.

Első pillantásra úgy tűnhet, hogy ha mondjuk 1µs ideig tart egy csomagot feldolgozni, akkor a router másodpercenként 1 millió csomagot tud kezelni. Ez a számítás azonban nem felel meg a valóságnak, mert a terhelés statisztikai ingadozásai miatt mindig lesznek tétlen periódusok. Ha a processzornak minden egyes ciklusra szüksége van a munkája elvégzéséhez, akkor egy esetleges tétlenség okozta pár ciklusnyi kiesés miatt is olyan hátrányba kerül, amit már soha nem fog tudni behozni.

Sőt, még a kevéssel az elméleti kapacitás alatt maradó terhelés esetén is sorok alakulhatnak ki, és késleltetés léphet fel. Tekintsünk egy olyan helyzetet, ahol a csomagok véletlenszerű időközönként, átlagosan λ csomag/s sebességgel érkeznek be. Az általuk igényelt processzoridő szintén véletlen eloszlású, az átlagos feldolgozási kapacitás μ csomag/s. Ha feltesszük, hogy mind a beérkezés, mind a kiszolgálás Poisson eloszlású, akkor a tömegkiszolgálás eszközeivel megmutatható, hogy a csomagok által elszenvedett T késleltetés átlagos értéke [19]:

$T = 1/\mu \times 1 / (1 - \lambda/\mu) = 1/\mu \times 1 / (1 - p)$

Ahol $p = \lambda/\mu$ a processzor kihasználtsága. Az első tényező $1/\mu$ azt mutatja meg, hogy mennyi lenne a kiszolgálási idő, versengés nélkül. A második tényező a többi folyammal való versengés okozta lassulást jelképezi. Például, ha λ = 950000 csomag/s és μ = 1000000 csomag/s, akkor p = 0,95 és az egyes csomagok által elszenvedett átlagos késleltetés 20 µs lesz 1 µs helyett. Ebben benne van a sorban állás és a kiszolgálás ideje is, amint az alacsony terhelés esetén látszik ($\lambda/\mu \approx 0$). Ha a folyam útvonalán mondjuk 30 router van, akkor egyedül a sorban állásból fakadó késleltetés is már 600 µs lesz.

2.3.7 Belépés-engedélyezés

Pillanatnyilag ott tartunk, hogy a valamely forrásból beérkező forgalom jól formált, és potenciálisan egyetlen útvonalat követ, melynek mentén előre lefoglaltuk az erőforrásokat. Amikor egy router egy ilyen folyammal találkozik, a kapacitását és a más folyamokkal szemben már vállalt kötelezettségeit figyelembe véve el kell döntenie, hogy elfogadja vagy visszautasítja-e az új folyamot.

Az elfogadásról vagy elutasításról szóló döntés nem pusztán a folyam kéréseinek (sávszélesség, pufferek, processzoridő) és a router ezen három dimenzió mértékében mért felesleges kapacitásainak összehasonlításából áll. Ennél azért bonyolultabb kérdésről van szó. Kezdjük azzal, hogy néhány alkalmazás tisztában van ugyan a saját sávszélesség igényével, a pufferekről és a processzoridőről azonban már keveset tudnak. Így mindenképp más utat kell találnunk a folyamok leírására. Továbbá, egyes alkalmazások sokkal jobban tudnak tolerálni egy esetleges lekésett határidőt, mint mások. Végül, egyes alkalmazások hajlandók lehetnek a folyami paraméterekről alkudozni, mások ellenben nem. Például egy általában 30 keret/s sebességgel működő filmlejátszó hajlandó lehet 25 keret/s sebességre visszaállni, ha nincs elég sávszélesség az előbbi sebességhez. Hasonlóképp változtatható a keretenkénti képpontok száma, a hang sávszélessége és egyéb jellemzők.

Mivel az egyeztetésben sok fél vesz részt, a tárgyalás alapját képező paramétereket tekintve pontosan le kell tudnunk írni a folyamokat. Az ilyen paraméterek halmazát folyam meghatározásnak (flow specification) hívjuk. Általában a feladó (pl. egy videókiszolgáló) állítja elő a folyam meghatározást, amikor is ajánlatot tesz az általa használni kívánt

paraméterekre. A meghatározás végighalad a leendő útvonalon, ahol minden router megvizsgálja azt, és igénye szerint módosítja az egyes paramétereket. A módosítások nem növelhetik a folyamot, csak csökkenthetik (pl. alacsonyabb bitsebességet megadhatnak, magasabbat nem). Az útvonal végére érve a paramétereket rögzíteni lehet.

A folyam meghatározás lehetséges tartalmára mutat példát a 2.9-es ábra, mely az RFC 2210-et és az RFC 2211-et veszi alapul. Itt öt paraméterünk van, melyek közül az első a vezérjeles vödör sebessége, ami a vödörbe kerülő bájtok másodpercenkénti számát adja meg. Ez a feladó által fenntartható legnagyobb adási sebesség, hosszú idő átlagában nézve.

Paraméter	**Egység**
Vezérjeles vödör sebessége	bájt/s
Vezérjeles vödör mérete	bájt
Adatsebesség csúcsértéke	bájt/s
Minimális csomagméret	bájt
Maximális csomagméret	bájt

2.9-es ábra.

A második paraméter a vödör bájtokban vett mérete. Ha például a vezérjeles vödör sebessége 1MBájt/s és a vezérjeles vödör mérete 500 KBájt, a vödröt fél másodpercen át lehet folyamatosan tölteni, amíg megtelik, ha közben nincs semmilyen más adás. Minden ezután küldött vezérjel elvész.

A harmadik paraméter, az adatsebesség csúcsértéke a maximális megengedett átviteli sebesség. A feladó sosem lépheti túl ezt az értéket, még rövid időre sem.

Az utolsó két paraméter a minimális és maximális csomagméretet határozza meg, beleértve a szállítási és a hálózati réteg fejrészeit is (pl. TCP és IP). A minimális méret azért fontos, mert a feldolgozás minden csomagnál időt vesz igénybe, függetlenül attól, hogy milyen rövid a csomag, Lehet egy router képes másodpercenként 10000 darab 1 KBájt-os csomagot kezelni, de korántsem biztos, hogy megbirkózik a másodpercenként 100000 darab 50 bájtos csomaggal, hiába felel ez meg alacsonyabb bitsebességnek. A maximális csomagméret a hálózat belső korlátjai miatt fontos, ezeket ugyanis nem lehet túllépni. Ha például az útvonal

egy része egy Etherneten keresztül vezet, akkor a maximális csomagméret nem lehet több mint 1500 bájt, függetlenül attól, hogy a hálózat fennmaradó része mennyit képes kezelni.

Érdekes kérdés, hogy a router hogyan alakíthatja át a folyam meghatározásokat konkrét erőforrás-foglalások halmazává. Ez a leképezés a tényleges megvalósítástól függ, és nincs szabványosítva. Tegyük fel, hogy egy router 100000 csomagot dolgoz fel másodpercenként. Ha felajánlanak neki egy 1MB/s-os folyamot, ahol a minimális és a maximális csomagméret is 512 bájt, a router kiszámíthatja, hogy az adott forrásból 2048 csomagot kaphat másodpercenként. Ebben az esetben a processzoridejének 2%-át kell fenntartania az adott folyam számára, sőt lehetőleg valamivel többet, hogy elkerülhetők legyenek a hosszú sorban állás okozta késleltetések. Ha a router stratégiája az, hogy sosem osztja ki a processzoridő több mint 50%-át, ami kétszeres késleltetést jelent, és már 49%-ban foglalt, akkor ezt a folyamatot vissza kell utasítania. Hasonló számítások szükségesek más erőforrások esetében is.

Minél szorosabb a folyam meghatározás, annál több hasznát veszik a routerek. Ha a meghatározás megállapítja, hogy 5MBájt/s vezérjeles vödör sebességre van szüksége, de a csomagméret 50-től 1500 bájtig terjedhet, akkor a csomagsebesség is bárhol lehet 3500 és 105000 csomag/s között. A router lehet, hogy pánikba esik az utóbbi láttán és visszautasítja a folyamot, pedig 1000 bájtos minimális csomagmérettel az 5MBájt/s-os folyamot még elfogadhatta volna [1].

2.3.8 Arányos útvonalválasztás

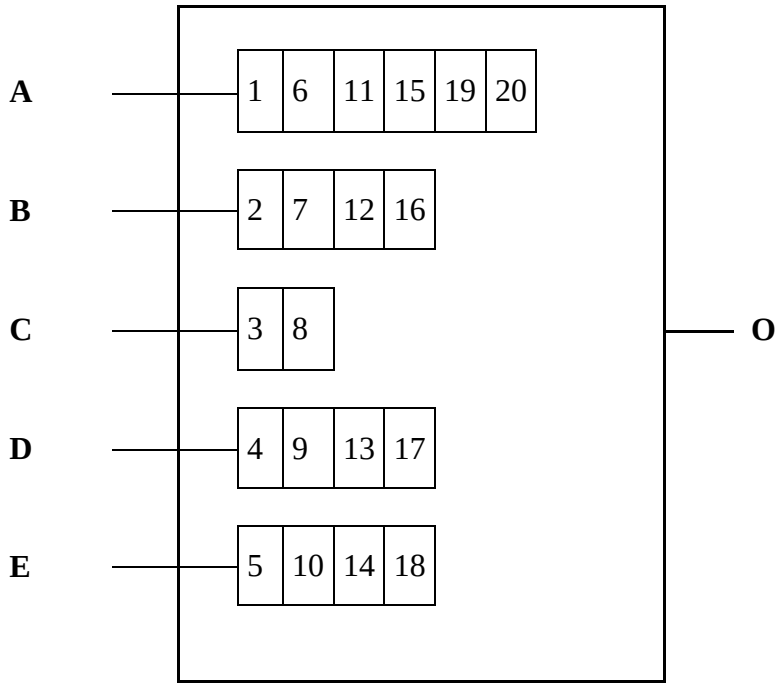
A legtöbb forgalomirányító algoritmus minden cél csomóponthoz igyekszik megtalálni a legjobb utat, és aztán minden forgalmat az adott címzetthez vezető legjobb úton lebonyolítani. A jobb szolgáltatásminőség érdekében azonban egy másik megközelítést is javasoltak: eszerint az egyazon címzetthez tartó forgalmakat is érdemes több útvonalra szétosztani. Mivel a routereknek általában nincs teljes rálátásuk a hálózat egészének forgalmára, az egyetlen járható út a forgalom több útvonal közti megosztására az, ha a helyben rendelkezésre álló információt használjuk fel. Egyszerű megoldásként a forgalmat felosztjuk egyenlő részekre, vagy megoszthatjuk esetleg a kimeneti vonalak kapacitásának arányában. Emellett más, kifinomultabb algoritmusok is léteznek, melyekről szakkönyvek írnak részletesen.

2.3.9 Csomagütemezés

Ha egy router több folyamot kezel, fennáll a veszélye annak, hogy egy folyam túl nagy részét sajátítja ki a kapacitásoknak, és ebből kifolyólag kiéheztet másokat. Ha a beérkezésük sorrendjében dolgozzuk fel a csomagokat, akkor egy kellően agresszív feladó megszerezheti a csomagjai által érintett routerek kapacitásának nagy részét, ezzel rontva másoknak jutó szolgáltatásminőséget. Az ilyen kísérletek meghiúsítására számos csomagütemező algoritmus született.

Az első ilyen elgondolások között volt az egyenlő esélyű sorbaállás (fair queuing) algoritmusa. Az algoritmus lényege, hogy a routerek külön várakozó sorokat tartanak fenn minden kimeneti vonalon, minden folyam számára egyet. Ha egy vonal tétlen lesz, a router körforgásos (round robin) alapon végignézi a sorokat, és kivesz egy csomagot a következő sorból. Ily módon, ha n kommunikációs viszony verseng egy adott kimeneti vonalért, akkor mindegyik egy csomagot küldhet el minden n csomagból. Több csomag küldése nem javítja ezt az arányt. Bár kiindulási alapnak jó, az algoritmusnak van egy problémája: nagyobb sávszélességet ad a nagy csomagokat használó alkalmazásoknak, mint a kis csomagokat használóknak. Továbbfejlesztették, ahol a körforgást úgy valósítják meg, hogy bájtonként körforgást szimulál csomagonkénti körforgás helyett. Ez úgy működik, hogy újra és újra végignézi a sorokat bájtról bájtra, amíg meg nem találja azt az óraütést, amikor minden csomag a végére ér. Ezután a csomagokat a véget érésük sorrendjébe rendezi, és ebben a sorrendben küldi el. Az algoritmust a 2.10-es ábra mutatja. Az ábra bal részén 2-6 bájt hosszú csomagokat láthatunk. Az első virtuális óraütéskor az A vonalon lévő csomag első bájtját küldjük el. Ezután a B vonal csomagjának első bájtját és így tovább. Az első véget érő csomag a C, nyolc óraütés után. Az ábra jobb részén megadjuk a sorrendet. Új érkezők hiányában a csomagokat a felsorolt sorrendben küldjük el C-től A-ig, FIFO elven [1].

Ennek az algoritmusnak egy a problémája, hogy minden applikációnak azonos prioritást biztosít. Sok esetben kívánatos a videóapplikációnak több sávszélességet adni, mint a hagyományos fájlapplikációnak, tehát akár két vagy több bájtot is adhatunk nekik óraütésenként. Ezt a módosított algoritmust súlyozott egyenlő esélyű sorba állásnak (weighted fair queueing) nevezik, és széles körben használják.



Csomag	Befejeződési idő
C	8
B	16
D	17
E	18
A	20

2.10-es ábra.

A súly megegyezhet az egy gépből kijövő folyamok számával, így minden folyamat egyenlő sávszélességet kap. Az algoritmus egy hatékony megvalósítását tárgyalja. A csomagok routeren vagy kapcsolón keresztül való tényleges továbbítását egyre inkább hardveresen végzik. Ezek az eljárások felborítják a rétegelt architechtúrát, mert az applikációs réteg módosítja a keret fejrész szerkezetét.

A lehetséges technológiák után az Integrált (IntServ) és Differenciált (DiffServ) QoS szolgáltatásait és megvalósításait mutatom be.

3. Integrált QoS szolgáltatások

3.1 Bevezetés

1995 és 1997 között az IETF sok energiát fektetett egy élő közvetítésű multimédia architektúra kidolgozásába. A munka eredménye két tucat RFC lett, az RFC 2205-től kezdve az RFC 2210-ig. E művek a folyam alapú algoritmusok, avagy integrált szolgáltatások nevet viselik. Az elgondolások egyaránt irányulnak az egyes küldéses és többes küldéses alkalmazásokra. Az előbbire példa egy szóló felhasználó, aki egy videóklippet néz egy híroldalról. Az utóbbira digitális televízió állomások egy csoportja, melyek egy IP csomagokból álló folyammal közvetítik műsorukat a különböző helyeken tartózkodó nézőknek. A továbbiakban az egyes küldés RSVP-vel foglalkozok.

3.2 RSVP – erőforrás foglalási protokoll

Az integrált szolgáltatások architektúrájának fő IETF-protokollja az RSVP (Resource reSerVation Protocol). Leírását az RFC 2205 és más RFC-k tartalmazzák. Ezt a protokollt a foglalásokhoz használják, az adatok elküldését más protokollok végzik. Az RSVP lehetővé teszi, hogy több adó adjon vevők több csoportjának, továbbá hogy az egyes vevők szabadon választhassanak a csatornák között. A protokoll ezen felül optimizálja a sávszélesség-felhasználását, miközben kiküszöböli a torlódásokat is [22].

Nemcsak egyféle módszer létezhet a hálózat erőforrásainak elosztására, de szükségszerű, hogy egy hálózat minden adattovábbító eleme képes legyen az erőforrásainak egy részét QoS csomagok számára fenntartani, és azonosítani a QoS szolgálatok számára fennmaradó erőforrásokat. Az IntServ modellben például minden QoS adatfolyam számára egyenként foglalunk le erőforrásokat, ami a hálózati erőforrások hatékony elosztását teszi lehetővé, kielégítve azok egyedi igényeit. Ennek azonban az a hátránya, hogy a hálózat erősen terhelt részeiben igen sok állapotinformáció tárolására van szükség. A kapcsolat felépítéséhez előzetes kommunikáció szükséges, ezt a feladatot egy magasabb rétegbeli mechanizmus vezérli. Ennek legfontosabb feladata az erőforrás-lefoglalási kérések közvetítése, másképp fogalmazva az, hogy egy garantált minőségű átviteli utat biztosítson egy új QoS igény számára. Kapcsolat-felépítéskor az alkalmazások közlik, hogy a felépítendő adatfolyamnak

milyen várható tulajdonságai vannak (sávszélesség, késleltetés stb.), ezeket a paramétereket a hálózat megkísérli biztosítani az új folyam számára.

Az RSVP két elemi műveletben szimplex csatornákat épít fel. Elsőként a küldő kibocsát egy „lefoglalás-létrehozó” (PATH) üzenetet, megadva az új folyam szükséges tulajdonságait, és a cél IP címét. A PATH üzenet számára kijelölt útvonal minősége az adatátvitel szempontjából kulcsfontosságú, hiszen ezen az útvonalon fog a folyam haladni. Amennyiben a fogadó fél elfogadja a PATH üzenetet, az erőforrás-lefoglalás a kijelölt útvonalon visszafelé megtörténik (RESV üzenet).

Az átviteli vonal routerein az RSVP csomagokat figyelő elemek döntenek a QoS kapcsolat-felépítés sikeréről, kezelik az erőforrás-lefoglaláshoz rendelt alacsony szintű funkciókat, például: pufferek, alacsony szintű csomagszűrők, ütemezési sorok stb.. Az IntServ modell fontos következménye, hogy az adatátviteli útvonalak folyamonként külön-külön jelölendők ki, ami a forgalomszabályozás szempontjából azt jelenti, hogy a hálózati terhelést az elérhető legnagyobb pontossággal kezelhetjük. Emiatt hasznos lehet egy speciális router protokoll, ami képes a QoS folyamok számára egyenként útvonalat választani, figyelembe véve a folyam tulajdonságait és a hálózat aktuális állapotát is.

A fenti követelmények megvalósítására példa lehet a QoSPF protokoll. Ez a protokoll a routerek összeköttetés-állapotainak és a hálózat más szabad erőforrásainak ismeretében kiszámolja az egyes célpontokig vezető legnagyobb szabad sávszélességet biztosító legrövidebb útvonalakat, és ezek alapján frissíti a saját elkülönített belső útvonal-választási táblázatát. Ugyanakkor a Best effort forgalmak számára továbbra is a hagyományos útvonal-választási módszer használatos. Így mikor az RSVP egy erőforrás-lefoglalást dolgoz fel, megkérdezi a QoS útvonalkezelő modult, hogy szolgáltasson információt a legjobb útvonalról, ha egyáltalán létezik a folyam paraméterei alapján ilyen.

Számos kutatási eredmény számol be a hálózati kihasználtság növekedéséről, és a szolgáltatásminőség javulásáról a QoSPF protokoll alkalmazása esetén. A felhasználók által érzékelt javulás mértéke akkor igazán jelentős, ha adott forrás és cél között a hálózatban több alternatív útvonal is rendelkezésre áll. Ebben az esetben a hálózat üzemeltetője a hálózati nyereség növekedését tapasztalja, ugyanakkor a QoS útvonalválasztás bonyolultsága nem

elhanyagolható, számos korábbi eredmény azt mutatja, hogy az ebből származó számítási kapacitás- és hálózati forgalomnövekedés még nagy hálózatok esetében is tolerálható.

Az RSVP és a QoSPF gyakran használja az alacsony szintű forgalomvezérlési rendszert, amely a megkülönböztetett szolgáltatást igénylő csomagok garantált kiszolgálásáért felelős. Az alacsony szintű forgalomvezérlési rendszer végzi a csomagütemezők és a csomagszűrők beállítását, illetve kezeli a csomagtovábbítási táblázatokat. Ezt a rendszert QoS csomagütemező rendszernek hívjuk és feltételezzük, hogy az operációs rendszer valósítja meg. Létezik egy különálló forgalomszabályozó elem, aminek bevezetésével egy olyan kommunikációs felület alakítható ki, amely segítségével a QoS csomagtovábbító rendszer platform-függetlenül elérhető mind az RSVP, mind pedig a QoSPF számára.

A QoS biztosítása az alkalmazási rétegtől az adatkapcsolati rétegig mindenhol változtatásokat követel meg, az utóbbin kiforrott ütemező algoritmusok használata válhat szükségessé. A hálózati réteget alkalmassá kell tenni a QoS folyamok csomagjainak felismerésére. Magasabb szinteken jelzési protokollok használata, és valós idejű szállítási rétegbeli protokollok használata szükséges.

Eredetileg az IP-t úgy alkották meg, hogy bármilyen adatkapcsolati réteggel együtt tudjon működni. QoS biztosításához viszont az IntServ erőforrás-lefoglalási rendszere speciális csomagtovábbító mechanizmust igényel, így az RSVP is pontosan definiálja a saját hozzáférési felületét (API), amin keresztül a kernel funkciókat el szeretné érni. Ennek a neve: LLDAL (Link-layer-dependent Adaptation Layer)

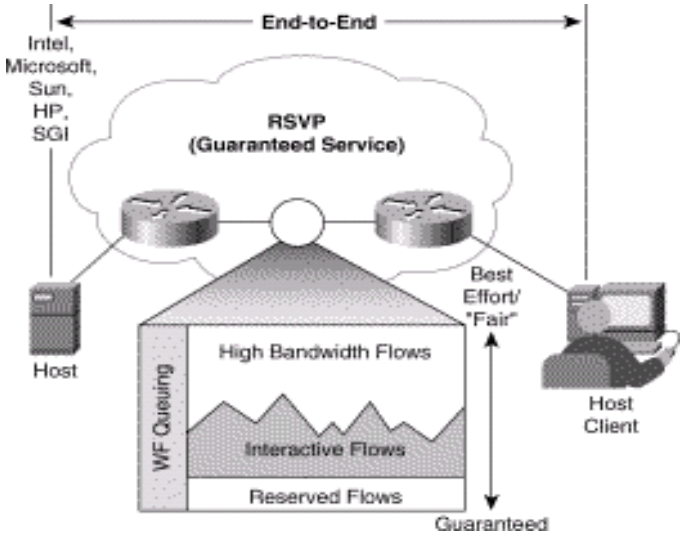
Az RSVP erőforrás-lefoglaláskor értesíti a QoS csomagtovábbító rendszert a szükséges pufferek méretéről, a létrehozandó csomagosztályozó szűrők tulajdonságairól, és a folyam csomagjainak továbbítási irányáról. A QoS útvonalválasztó modult értesíteni kell a lokális erőforrások mennyiségéről, hogy a hálózatban a szabad erőforrások elérhetőségére vonatkozó információ naprakész maradjon.

Példa: IETF RSVP: (3.1-es ábra) [17]

A router-ek ezáltal két szolgálatot képesek nyújtani:

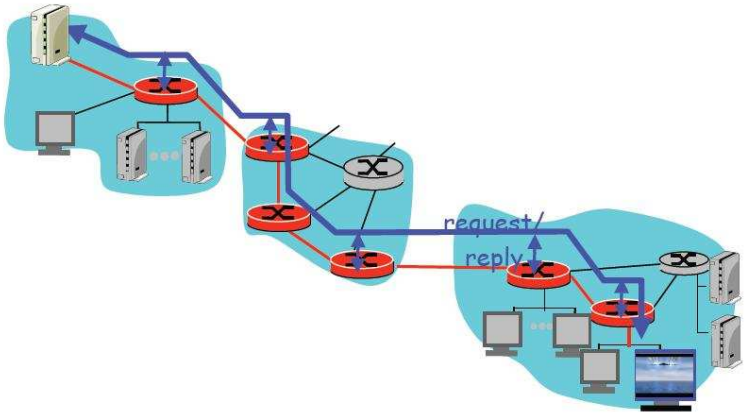
- *Garantált átviteli ráta*: sávszélesség foglalás (max. 32 Mbps két végpont közötti hangkapcsolat számára). Pl. Weighted Fair Queueing (WFQ)

- *Ellenőrzött terhelés*: alacsony késleltetés és magas ráta igénylés. Pl. Weighted Random Early Detection (WRED)



3.1-es ábra.

Intserv modell [15]:



3.2-es ábra.

3.3 CISCO IOS 12.1 fontosabb RSVP QoS parancsai [18]

Parancs: ip* rsvp* bandwidth [interface-kbps*] [single-flow-kbps*]

Jelentése: Aktivizálja az RSVP-t az interfészen

Parancs: ip rsvp sender session-ip-address sender-ip-address [tcp | udp | ip-protocol] session-dport sender-sport previous-hop-ip-address previous-hop-interface bandwidth burst-size

Jelentése: A küldőket beírja a RSVP adatbázisba

Parancs: ip rsvp reservation session-ip-address sender-ip-address [tcp | udp | ip-protocol] session-dport sender-sport next-hop-ip-address next-hop-interface {ff | se | wf} {rate | load} bandwidth burst-size

Jelentése: A vevőkészülékeket beírja a RSVP adatbázisba

Parancs: ip rsvp precedence {conform precedence-value | exceed precedence-value}

Jelentése: Beállítja a precedencia értékeket

Parancs: ip rsvp tos {conform tos-value | exceed tos-value}

Jelentése: Beállítja a ToS értékeit (Type of Service)

Parancs: priority-list list-number protocol protocol-name {high | medium | normal | low} queue-keyword keyword-value

Jelentése: Megállapítja az alapvető sorbaállási prioritásokat protokoll típusától függően

4. Differenciált QoS szolgáltatások

4.1 Bevezetés

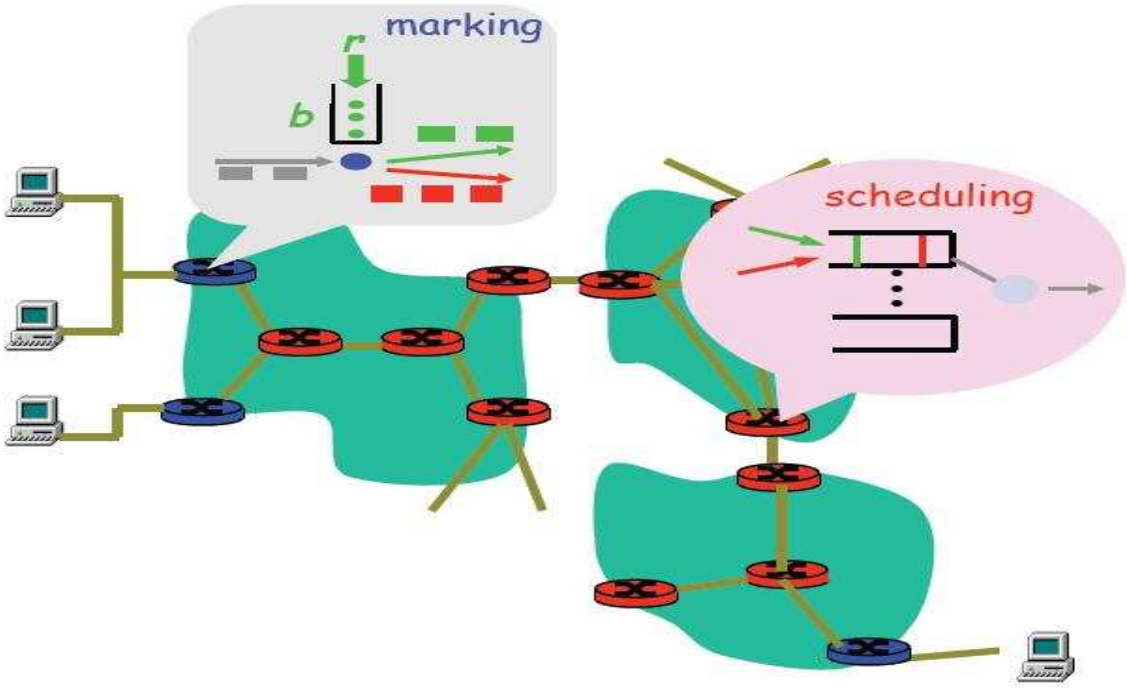
A folyam alapú algoritmusoknak megvan az a képességük, hogy jó szolgálatminőséget tudnak biztosítani egy vagy több folyam számára, mivel bármilyen igényelt erőforrást le tudnak foglalni az út mentén. Megvannak azonban a maguk hátrányai is. Minden egyes folyam kialakítása előkészületeket igényel, ez pedig nehezen kézben tartható, ha több ezer vagy millió folyamról van szó. Ráadásul a folyamok állapotát a routerekben tárolják, ami sebezhetővé teszi őket a routerek összeomlása esetén. Végül jelentős változtatásokat követelnek meg a routerek szoftverében is, a folyamok felépítése pedig bonyolult üzenetváltásokkal jár együtt a routerek között. Ebből fakadóan az RSVP-nek és a hasonló protokolloknak alig született még megvalósítása.

Ezen okok miatt az IETF is egy egyszerűbb megközelítést gondolt ki a szolgáltatásminőség megvalósítására, egy olyat, amely minden routerben javarészt helyileg implementálható, az előkészületek és az egész útvonal bevonása nélkül. Ezt a megoldást osztály alapú (class-based) szolgálatminőségnek nevezik (szemben az eddig látott folyam alapúval). Az IETF egy architektúrát is szabványosított a számára, differenciált szolgáltatások (differentiated services, DS) néven, melyet az RFC 2474, RFC 2475 és számos más RFC ír le. A következőkben itt is bemutatásra kerül [9].

Differenciált szolgáltatásokat (DS) az egy adminisztratív körzet (pl. egy internet szolgáltató vagy telefontársaság) alá tartozó routerek egy csoportja kínál. Az adminisztráció definiálja a szolgáltatási osztályok egy halmazát a nekik megfelelő továbbítási szabályokkal együtt. Ha egy ügyfél feliratkozik egy DS-re, akkor a körzetbe belépő csomagjai egy szolgáltatás típusa (Type of Service) mezőt is hordozhatnak, így bizonyos osztályok számára jobb szolgáltatások biztosíthatók (pl. prémiumszolgáltatás), mint mások számára. Egy adott osztályhoz tartozó forgalomtól megkövetelhető, hogy egy meghatározott mintának feleljen meg, ez lehet például egy adott sebességgel csöpögő lyukas vödör. Egy jó üzleti érzékkel megáldott szolgáltató külön díjat számolhat fel minden leszállított prémium csomag után, vagy engedélyezhet legfeljebb havi TV darab prémium csomagot egy rögzített havi különdíj ellenében. Vegyük észre, hogy ez a séma (szemben az integrált szolgáltatásokkal), nem igényel előzetes

kapcsolat-felépítést, sem erőforrás-lefoglalást, sem időigényes végpontok közti tárgyalásokat külön-külön minden egyes folyamhoz. Emiatt a DS viszonylag egyszerűen megvalósítható (4.1-es ábra).

Diffserv modell [15]:



4.1-es ábra

Osztály alapú szolgáltatásokkal más területeken is találkozhatunk. A csomagküldő szolgálatok például gyakran kínálnak éjszakai, kétnapos vagy háromnapos kézbesítést. A légitársaságok első osztályú, üzleti osztályú és turista osztályú szolgáltatásokat nyújtanak. Gyakran a távolsági vonatokon is többféle szolgáltatási osztály van, sőt még a párizsi metrónak is két osztálya van. A csomagok esetében az osztályok többek közt a késleltetés, a dzsitter és a torlódás esetén történő eldobás valószínűsége szerint különbözhetnek (ez a terjedelmesebb Ethernet keretekre valószínűleg nem érvényes).

Hogy jobban meg tudjuk különböztetni a folyam alapú és az osztály alapú szolgáltatásminőséget, vegyük szemügyre az Internet telefónia példáját. Folyam alapú sémát használva minden hívás megkapja a maga erőforrásait és garanciáit. Osztály alapú sémával az összes hívás együtt kapja meg az adott osztály számára lefoglalt erőforrásokat. Ezeket az

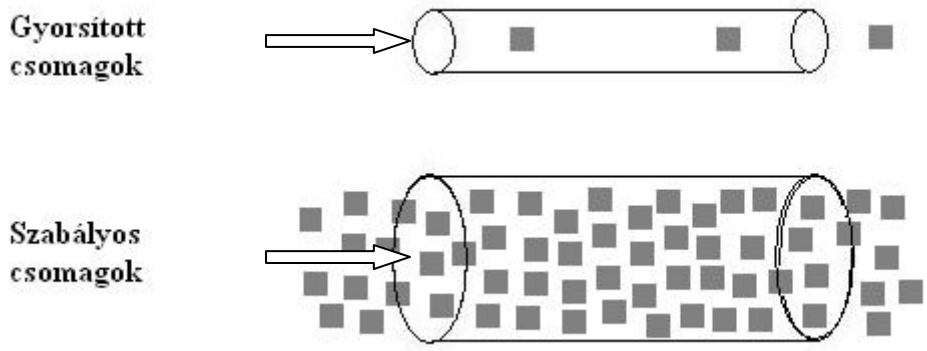
erőforrásokat nem vehetik el a fájlátviteli osztályhoz vagy más osztályhoz tartozó csomagok, de az egyes telefonhívásoknak sem lesznek fenntartva külön-külön erőforrások.

4.2 Gyorsított továbbítás

A szolgáltatási osztályok megválasztása a hálózat-üzemeltetők dolga, de mivel a csomagokat gyakran különböző szolgáltatókhoz tartozó alhálózatok között továbbítják, az IETF már a hálózat független szolgáltatási osztályok meghatározásán dolgozik. A legegyszerűbb ilyen osztály a gyorsított továbbítás (expedited forwarding), kezdjük most ezzel. Az osztályt az RFC 3246 írja le [9].

A gyorsított továbbítás mögött megbúvó ötlet nagyon egyszerű. Kétfajta szolgáltatási osztály áll rendelkezésre: szabályos és gyorsított. A forgalom túlnyomó része várhatóan szabályos lesz, de a csomagok egy kis hányadát gyorsítjuk. Az ilyen gyorsított csomagoknak elvileg úgy kell keresztülhaladniuk az alhálózaton, mintha más csomag nem is lenne ott jelen. Ezt a „kétcsöves" rendszert ábrázolja jelképesen a 4.2-es ábra. Vegyük észre, hogy fizikailag továbbra is csak egy vonalunk van. Az ábra logikai csővezetékei csak a sávszélesség fenntartásának egy módját jelzik, és nem egy második fizikai vonalat.

Ezt a stratégiát például úgy valósíthatjuk meg, hogy a routereket úgy programozzuk be, hogy két kimenő várakozási sort tartsanak fenn minden egyes kimeneti vonalhoz, egyet a szabályos, egyet a gyorsított csomagok számára. Amikor egy csomag megérkezik, a megfelelő sorba kerül. A csomagidőzítéshez célszerű valamilyen, a súlyozott egyenlő esélyű sorba állításhoz hasonló algoritmust használni. Például, ha a forgalom 10%-a gyorsított, és 90%-a szabályos, akkor a sávszélesség 20%-át megtarthatjuk a gyorsított forgalomnak, a maradékot pedig a szabályosnak. Ezzel kétszer annyi sáv- szélességet biztosítunk a gyorsított forgalom számára, mint amennyire szüksége van, így alacsony késleltetést is biztosíthatunk. Ezt a kiosztást úgy valósíthatjuk meg, ha egy gyorsított csomag küldése jut négy szabályos csomag elküldésére (feltéve, hogy mindkét osztálynál hasonló a csomagméretek eloszlása). Ily módon a gyorsított csomagok remélhetőleg egy terhelés nélküli alhálózatot látnak, még akkor is, ha valójában nagy a terhelés.

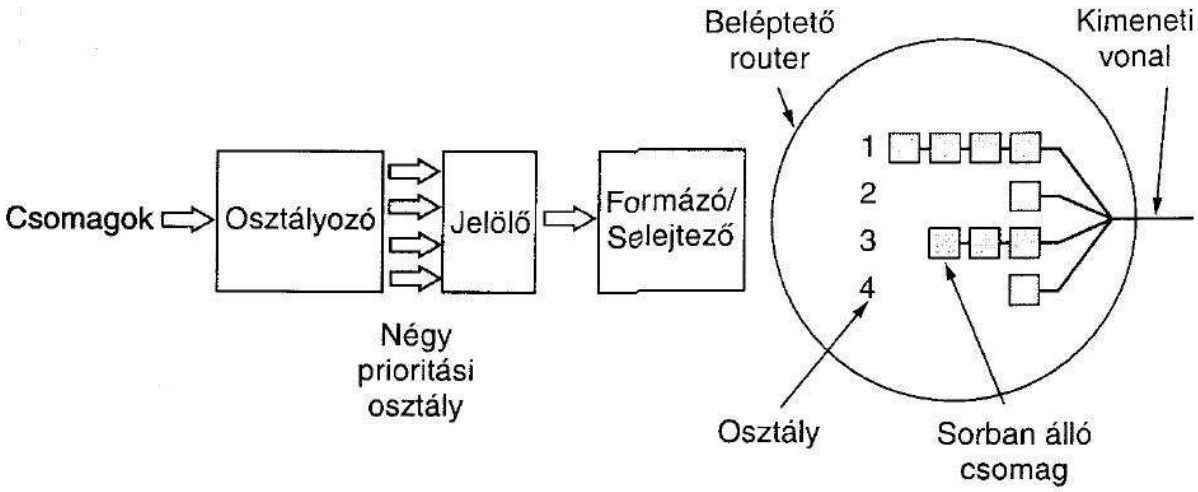


4.2-es ábra.

4.3 Biztosított továbbítás

A szolgáltatási osztály az előzőnél némileg kidolgozottabb kezelését nyújtja a biztosított továbbítás (assured forwarding) sémája, melyet az RFC 2597 ír le. A séma négy prioritási osztályt ad meg, itt minden osztályhoz saját erőforrások tartoznak. Ezen felül a torlódásba került csomagok eldobására is definiálnak három különböző valószínűséget (kis, közepes és nagy). Mindent összevetve, a két tényező együtt 12 szolgáltatási osztályt határoz meg [1].

Az 4.3-as ábra a csomagok biztosított továbbításának egy lehetséges módját mutatja. Első lépésben minden csomagot a négy prioritási osztály egyikébe sorolnak. Ezt megteheti a feladó hoszt (ahogy az ábra is mutatja) vagy az első (ún. beléptető) router. Az osztályozást azért előnyösebb a feladó hoszt oldalán elvégezni, mert itt több információ áll rendelkezésre arról, hogy mely csomagok tartoznak mely folyamokhoz. A második lépésben a csomagokat az osztályuknak megfelelően megjelöljük. Ehhez egy fejrész mezőre van szükség, szerencsére rendelkezésünkre áll egy 8 bites szolgálat típusa mező az IP-fejrészben, amint azt hamarosan látni fogjuk. Az RFC 2597 rögzíti, hogy ezek közül 6 bit használható a szolgáltatási osztály megadására, így a kódolásban marad hely a múltbeli és a jövőbeli szolgáltatási osztályok számára is [14][1].



4.3-as ábra.

A harmadik lépésben a csomagokat egy formázó/selejtező szűrőnek adjuk át, ami néhány csomagot késleltethet vagy eldobhat annak érdekében, hogy a négy folyamot elfogadható alakra hozza, például lyukas vagy vezérjeles vödör segítségével. Ha túl sok a csomag, itt el is lehet dobni néhányat, az eldobási kategória szerint. Még finomabb sémák is elképzelhetők, melyek méréseket vagy visszacsatolást is alkalmaznak. Ebben a példában az említett három lépést a feladó hoszt hajtja végre, és a kimeneti folyam már így kerül bele a beléptető routerbe. Érdemes megjegyezni, hogy ezeket a lépéseket egy speciális hálózati szoftver, vagy akár az operációs rendszer is elvégezheti, hogy ne kelljen a már létező alkalmazásokat megváltoztatni [14].

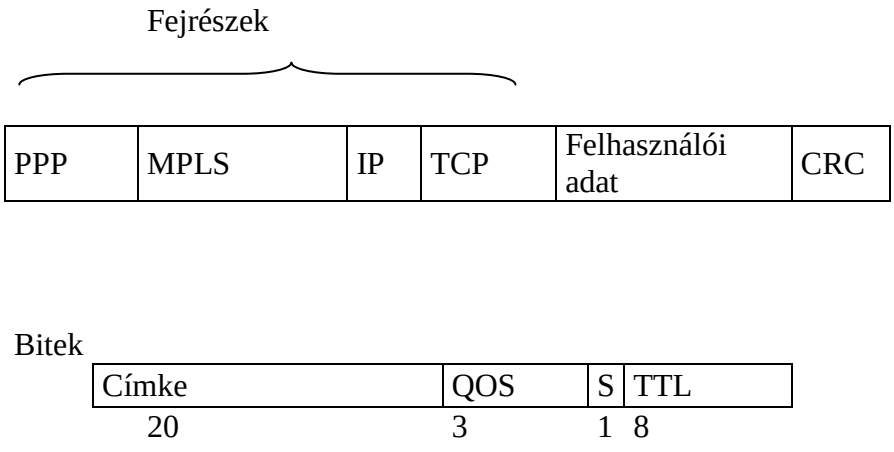
4.4 Címkekapcsolás és MPLS

Amikor az IETF kidolgozta az integrált és differenciált szolgáltatásokat, már számos router gyártó is dolgozott a továbbítási módszerek javításán. Ez a munka arra irányult, hogy egy címkét rakjanak minden csomag elé, és a forgalomirányítást a célcím helyett e címke alapján végezzék el. Azzal, hogy a címke egy belső táblázat egy sorát címzi meg, a megfelelő kimeneti vonal megtalálása pusztán egy táblázatban való keresés kérdése lesz. Ennek a módszernek a felhasználásával nagyon gyorsan el lehet végezni a forgalomirányítást, és le lehet foglalni a szükséges erőforrásokat az út mentén.

A címkézés ettől persze veszélyesen közel kerül a virtuális áramkörökhöz. Az X.25, az ATM, a keretátjátszás és más, virtuális áramkörökkel rendelkező hálózatok szintén egy címkét (azaz egy virtuális áramkör azonosítót) tesznek minden csomagba, majd kikeresik azt egy táblázatban, és a táblázat bejegyzése alapján végzik a forgalomirányítást. Annak ellenére, hogy az internetes közösségben sokan erős ellenérzéseket táplálnak az összeköttetés alapú hálózatok iránt, az ötlet mégis újra meg újra felbukkan, ezúttal a gyorsabb forgalomirányítás és a jobb szolgálatminőség érdekében. Ugyanakkor az Internet és az összeköttetés alapú hálózatok alapvetően különböző módon építik fel az útvonalaikat, tehát ez a módszer semmiképp sem azonos a hagyományos vonalkapcsolással [1].

Ez az „új" kapcsolási ötlet számos különböző (szabadalmaztatott) név alatt fut, mint például a címkekapcsolás (label switching) és a jelölőkapcsolás (tag switching). Az IETF végül a többprotokollos címkekapcsolás (MultiProtocol Label Switching, MPLS) néven szabványosította az elgondolást. Az alábbiakban mi is MPLS-nek fogjuk nevezni. A protokollt többek között az RFC 3031 írja le [9].

Kis kitérőként jegyezzük meg, hogy egyesek különbséget tesznek a forgalomirányítás és a kapcsolás között. A forgalomirányítás során a célcímet keressük ki egy táblázatból, hogy megtudjuk, hová kell küldeni a csomagot. A kapcsolás ellenben a csomagban található címkét használja indexként a továbbítási táblázathoz. Ezek a definíciók persze még messze nem univerzálisak.



4.4-es ábra.

Az első probléma az, hogy hová tegyük a címkét? Mivel az IP-csomagokat nem virtuális áramkörökhöz tervezték, az IP-fejrészben nincs mező a virtuális áramkör azonosítók számára. Emiatt új MPLS-fejrészt kell rakni az IP-fejrész elé. A 4.4-es ábra két router közötti vonalon használatos PPP adatkapcsolati protokoll keretszerkezetét mutatja, beleértve a PPP-, MPLS-, IP- és TCP-fejrészeket is. Ebben az értelemben az MPLS a 2. és a 3. réteg közötti 2,5. réteg.

Az általános MPLS-fejrésznek 6 mezője van, melyek közül a legfontosabb a Címke (Label) mező, amely az indexet tartalmazza. A QoS mező a szolgálatminőségi osztályt jelzi. Az S mező a hierarchikus hálózatokban használatos, és több, egymásra halmozott címkére utal. Ha ez eléri a 0-t, a csomagot eldobják. Ez akadályozza meg, hogy egy csomag a végtelenségig keringjen egy forgalomirányítási zavar esetén. Mivel az MPLS-fejrészek nem tartoznak sem a hálózati réteg csomagjához, sem az adatkapcsolati réteg keretéhez, az MPLS nagyban független mindkét rétegtől. Ez többek közt azt is jelenti, hogy olyan MPLS-kapcsolók is készíthetők, melyek mind IP-csomagokat, mind ATM-cellákat is képesek továbbítani, attól függően, hogy éppen mi érkezik. Innen származik a „többprotokollos" szó az elnevezésben.

Amikor egy MPLS-címkével kiegészített csomag vagy cella megérkezik egy MPLS-t ismerő routerhez, a címkét indexként használják egy táblázathoz annak meghatározására, hogy melyik kimeneti vonalat kell használni, és hogy milyen új címkét kell használni. Az effajta címkecserélgetést a virtuális áramkör alhálózatokban is használják, mert a címkéknek csak helyi jelentőségük van, és két különböző router össze nem tartozó csomagokat is továbbíthat ugyanazzal a címkével egy másik routerhez ugyanazon a kimeneti vonalon. Ahhoz, hogy a címkék a másik oldalon megkülönböztethetők legyenek, azokat minden ugrásnál újra le kell képezni.

Az MPLS a tömörítés mértékében eltér a virtuális áramköröktől. Az egyes folyamok minden további nélkül rendelkezhetnek saját címkekészlettel az alhálózatban. Sokkal gyakoribb azonban az, hogy a routerek több, egy adott routernél vagy LAN-nál végződő folyamot összefognak, és egy közös címkét használnak hozzájuk. Az egy címkéhez csoportosított folyamokat egyazon továbbítási ekvivalenciaosztályhoz (Forwarding Equivalence Class, FEC) tartozónak nevezzük. Ez az osztály nemcsak a csomagok címzettjét, hanem a szolgáltatási osztályukat is lefedi (a differenciált szolgáltatások értelmében), mert a továbbítás szempontjából az osztály minden csomagját egyformán kezelik.

A hagyományos virtuális áramkörös forgalomirányításnál nem lehet több, különböző címzetthez tartó útvonalat egyazon virtuális áramkör azonosítóval ellátni, mert ekkor a végső célállomásnál nem lehetne őket egymástól megkülönböztetni. MPLS használata esetén a csomagok továbbra is tartalmazzák a célcímet a címkén kívül, így a címkézett útvonal végén a címkefejrészt el lehet távolítani, és a továbbítás a megszokott módon folytatódhat a hálózati rétegbeli célcím felhasználásával.

Az MPLS és a hagyományos virtuális áramkörök között az egyik legnagyobb különbség a továbbítási táblázat felépítésének módjában van. Ha a felhasználó egy összeköttetést szeretne kiépíteni egy hagyományos virtuális áramkör hálózatban, akkor egy kapcsolatfelépítő csomagot indít útjára a hálózatban, hogy létrehozza az útvonalat és a megfelelő bejegyzéseket a továbbítási táblázatokban. Az MPLS nem így működik, mivel itt nincs minden egyes kapcsolat számára külön felépítési fázis (ez túl sok meglévő internet szoftvert érintene).

Ehelyett kétféleképpen lehet bejegyzéseket létrehozni a továbbítási táblázatban. Az adatvezérelt (data-driven) megközelítésben a csomag megérkezésekor az első érintett router felveszi a kapcsolatot a csomag útja mentén következő routerrel, és megkéri, hogy hozzon létre egy címkét a folyamnak. Ezt a módszert aztán rekurzív módon tovább alkalmazzák. Ezzel gyakorlatilag igény szerint hozzuk létre a virtuális áramköröket.

Az effajta terjesztést végző protokollok gondosan ügyelnek arra, hogy elkerüljék a hurkok kialakulását. Ehhez általában a színes szálak (colored threads) módszerét használják. A FEC visszafelé való terjedését ahhoz lehet hasonlítani, mintha egy egyedi színű szálat húznánk végig visszafelé a hálózaton. Ha egy router olyan színt lát, amilyen színű szállal már rendelkezik, akkor tudja, hogy hurok alakult ki, és valamilyen javító intézkedést tesz. Az adatvezérelt megközelítést elsősorban olyan hálózatokban használják, melyekben a szállítást az ATM végzi (mint a telefonrendszerek többségében) [1][2].

A másik, nem ATM alapú rendszerekben használatos megoldás a vezérlés alapú (control-driven) módszer. Ennek többféle változata is létezik, ezek közül az egyik működése a következő. Amikor egy router elindul, akkor megnézi, hogy mely útvonalak számára lesz ő a végállomás (azaz, hogy mely hosztok vannak a hozzá tartozó LAN-on). Ezeknek aztán létrehoz egy vagy több FEC-t, mindegyikhez hozzárendel egy címkét, majd átadja a címkéket a szomszédainak. Ezek aztán beírják a címkéket a továbbítási táblázataikba, és új címkéket

küldenek a szomszédaiknak, míg végül minden router megtanulja az útvonalat. A megfelelő szolgálatminőség érdekében erőforrások is lefoglalhatok az út kiépítése során.

Az MPLS egyszerre több szinten is működhet. A legmagasabb szinten minden szolgáltatót egyfajta metaroutemek tekinthetünk, amelynél létezik egy, a forrás és a cél csomópont között a metarouterekeken át vezető út. Ez az út használhatja az MPLS-t. Ugyanakkor az egyes szolgáltatók hálózatán belül is használhatunk MPLS-t, így egy alacsonyabb szintű címkézéshez jutunk, így egy csomag voltaképpen egy egész veremnyi címkét hordozhat magával. Az 4.4-es ábrán bemutatott S bit lehetővé teszi, hogy a router eltávolítson egy címkét, ha tudja, hogy maradnak még további címkék utána. Az S bitet l-re állítják a legalsó címkében, és 0-ra minden egyéb címkében. A gyakorlatban ezt a lehetőséget többnyire a virtuális magánhálózatok és a rekurzív alagutak megvalósítására használják. Bár az MPLS mögött rejlő alapötlet eléggé kézenfekvő, a részletek már rendkívül bonyolultak, számtalan változattal és optimalizációs lehetőséggel [15][16].

A 3 bites QoS mező értékkel adhatunk prioritást a kereteinknek, amik a következők lehetnek:

QoS 3 bit decimális értéke	Átvitel típusa
0	Legjobb továbbítás (Best effort)
1	Háttérbeni továbbítás (Background)
2	Kiváló továbbítás (Excellent effort)
3	Kritikus alkalmazások (Critical applications)
4	Videó továbbítás (< 100 ms késleltetés,ingadozás)
5	Hang továbbítás (< 10 ms késleltetés,ingadozás)
6	Internetwork control
7	Network control

4.5-ös ábra.

Az IntServ és DiffServ mechanizmusok után, a következő rész az egyik legnépszerűbb operációs rendszer, a Windows QoS beállításait tárgyalja.

5. A Windows szolgáltatásminősége

A Windows garantált szolgáltatásminősége (QoS) olyan módszerek összessége, melyek egy bizonyos típusú forgalomnak vagy hálózati működést végző programnak prioritást adnak ahelyett, hogy az elérhető legjobb szintű (best effort) továbbításra hagyatkoznának. A QoS mechanizmusok a Microsoft Windows 2000 és a Windows XP részeit képezik. A továbbiakban a Windows XP operációs rendszerben elérhető QoS-t ismertetem, de hivatkozok a Windows 2000-ben bevezetett szolgáltatásokra is [4].

5.1 Windows QoS alapjai

Internet kapcsolat megosztása esetében, ha egy hálózat lassú, például telefonos kapcsolaton keresztül kapcsolódik egy másik hálózathoz, előfordulhat, hogy a lassú kapcsolaton áthaladó forgalom késleltetése megnő. A késleltetés megjelenésének oka a két végpont által ismert sebesség és a lassú összekapcsolás sebessége közötti eltérés. A lassú összeköttetés szűk keresztmetszetet jelent a hálózati útvonalon. Ez csak a TCP protokoll összeköttetéses kommunikációjára érvényes.

Ha a fogadó kliens viszonylag gyors, például 100 megabit/másodperces Ethernet hálózaton helyezkedik el az Internet kapcsolat megosztása szolgáltatást futtató Windows XP rendszerű számítógép mögött, és a szerver, amellyel a kliens kommunikál, egy távoli elérés mögött található egy gyors hálózaton, fellép az eltérés. Ebben az esetben a fogadó magas értékre állítja be fogadási ablakát, mivel saját kapcsolatának sebességét veszi alapul. A küldő kezdetben alacsony sebességgel forgalmaz, ha azonban a csomagok nem vesznek el, végül a teljes ablakméretnek megfelelő mennyiségű csomagot küld el.

A jelenség más, ugyanazon hálózaton áthaladó TCP kapcsolatok teljesítményére is hatással lehet. A csomagok egy potenciálisan nagyméretű sorban várakoznak a lassú hálózaton történő továbbításra. Csomagvesztés esetén az adatok újraküldésére van szükség, ez tovább növeli a kapcsolat forgalmát.

A megoldást az jelenti, ha a hálózat szélén található, az internet kapcsolat megosztását működtető számítógép automatikusan a lassú összeköttetésnek megfelelő kisebb értékre állítja a fogadási ablakot. A beállítás felülbírálja a fogadó választását. A beállítás nem befolyásolja

károsan a forgalmat, hiszen az ablakméret értéke olyan lesz, mintha a fogadó közvetlenül a lassú összeköttetéshez kapcsolódna. Az ablak beállítását az internet kapcsolat megosztását működtető számítógépen futó QoS Csomagütemező végzi.

Még pár éve sokan kapcsolódtak lassú, például 56 kilobit/másodperces vonalakon az internethez. Számos felhasználó még az ilyen kapcsolatok sebességi korlátai mellett is több hálózatra kapcsolódó programot futtat egyszerre. A felhasználók például egyszerre végezhetnek letöltést, levelezést, csevegést vagy akár hang- illetve videólejátszást is. A legtöbb ilyen program a TCP protokollt veszi igénybe az átvitelre, és mindegyik saját összeköttetést épít fel, akár többet is [4][5].

A hálózatot elsőként igénybe vevő program kizárólagos használatot élvez, amíg a kapcsolat el nem éri az állandósult állapotot. Az állandósult állapot egy teljes TCP ablaknyi adat átvitelét jelenti. Amikor a következő program megkezdi az adatok átvitelét, annak kapcsolatát egy lassú kezdési algoritmus fogja befolyásolni, amely korlátozza a nem visszaigazolt adatok mennyiségét az átvitelben. Az előző program által forgalmazott adatmennyiség miatt a második program számára csak sokkal később lehetséges az állandósult állapot elérése, így egy hasonló adatmennyiség átvitele sokkal lassabb lesz.

Lassú kapcsolat használata esetén a Windows XP egy veszteséges körbeforgó (Deficit Round Robin, DRR) méltányossági sémát valósít meg. A séma már a Windows 2000 operációs rendszerben is elérhető volt. A Windows XP alapértelmezésben a lassú kapcsolatok észlelése esetén kapcsolja be a sémát. A séma számos adatfolyamot foglal le és ezekhez új alkalmazási adatfolyamokat rendel. A folyamok automatikusan körbeforgó módon kerülnek kiszolgálásra. Ez a konfiguráció jobb válaszidőket és teljesítményt biztosít a hálózati kommunikációban és nem igényel kézi beállítást.

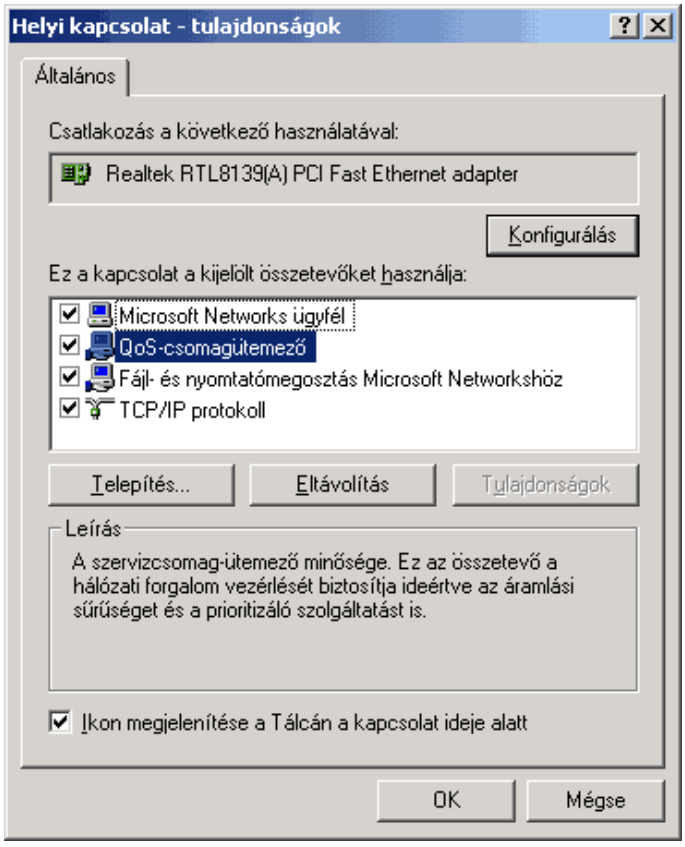
A Windows XP operációs rendszert futtató végpont számítógépek QoS használatának módszere, ahogyan a Windows 2000 esetében is, a programok a QoS API-k révén használhatják ki a QoS előnyeit. A hálózati sávszélesség száz százaléka megosztható a programok között, hacsak valamely program nem kér elsőbbségi sávszélességet. Ha a „lefoglalt” sávszélességet igénylő program nem küld adatot, a sávszélesség elérhető a többi program számára. A programok alapértelmezésben a számítógép minden egyes csatolója

esetében a kapcsolódó vonal sávszélességének összesen 20 százalékát foglalhatják le. Ha a sávszélességet lefoglaló program nem küld annyi adatot, hogy teljes mértékben kihasználja azt, a lefoglalt sávszélesség fel nem használt része az állomás többi adatfolyama számára is elérhető. Különböző műszaki cikkekben és hírcsoportokban jelentek meg olyan állítások, melyek szerint a Windows XP az elérhető sávszélesség 20 százalékát mindig a QoS számára foglalja le. Ezen állítások nem állják meg a helyüket [6].

5.2 QoS telepítése Windows 2000 Serverre és kliensre

Windows 2000 Server tartományvezérlő konfigurálása:

A kiszolgáló oldalt a Vezérlőpult> Programok telepítése/törlése > Windows összetevők hozzáadása vagy eltávolítása > Összetevők > Hálózati szolgáltatások > Részletek > QoS-belépésvezérlés szolgáltatás hozzáadásával lehet telepíteni.



3.1-es ábra.

Telepítés Windows 2000 Professional-re:

Az ügyfél oldal telepítését azon a hálózati kapcsolaton kell elvégezni, amelyiken használni akarjuk. Például a helyi hálózati kapcsolatnál a következő lépésekkel végezhető el a telepítés (3.1-es ábra):

Vezérlőpult > Hálózati és telefonos kapcsolatok > Helyi kapcsolat (vagy más használandó kapcsolat) > Tulajdonságok > Telepítés > Szolgáltatás > Hozzáadás > QoS-csomagütemező > OK. Ekkor a kapcsolat összetevői közé bekerül az ügyféloldali programcsomag.
Ha a Windows 2000 Server-t nem QoS kiszolgáló oldali szoftverrel akarjuk ellátni, hanem mint egyszerű munkaállomást kívánjuk üzemeltetni (QoS szempontból), akkor ott is így kell eljárnunk.

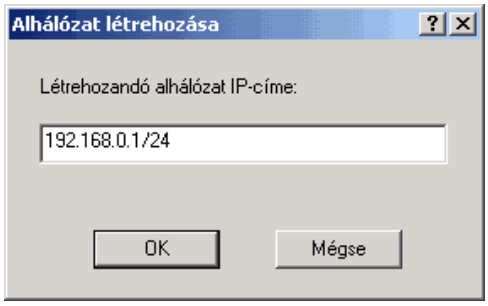
A kiszolgáló oldal konfigurálása:

Kattintsunk a tartományvezérlő Felügyeleti eszközök > QoS-belépésvezérlés menüpontjára a vezérlőkonzol megnyitásához.

A belépésvezérlést ún. házirendeken keresztül lehet konfigurálni. Két házirendcsoport létezik: vállalati a felhasználói szintű beállítások, és alhálózati a hálózati beállítások szintjén történő vezérléshez.
Nézzük először az alhálózati beállításokat:

Első feladatunk a működési területet leíró alhálózat hozzáadása: ehhez kattintsunk a konzol "Alhálózati beállítások" mappájára, utána a Művelet > Alhálózat hozzáadása menüpontra. Megjelenik egy ablak a "Létrehozandó alhálózat IP-Címe" feliratú mezővel, ahová a hálózat IP címét és hálózati azonosítójának hosszát kell beírni egy "/" jellel elválasztva. Ha a kezelendő alhálózat a 192.168.0.1-254 IP cím tartományból vesz fel címeket, és egy "C" osztályú alhálózatról van szó, ahol az alhálózati maszk: 255.255.255.0, akkor a hálózat címe: 192.168.0.0, a hálózati azonosító hossza: 24 bit (255.255.255.0 kettes számrendszerben: 11111111 11111111 11111111 00000000 így a hálózati információ (1-es bitek) hossza 24 (3*8)). Ezért ezt az alhálózatot a dialógus ablakban így kell megadni (3.2-es ábra): 192.168.0.0/24.

Hozzáadás után azonnal megjelenik a tulajdonságok ablaka, ahol elvégezhetjük a szükséges beállításokat.



3.2-es ábra

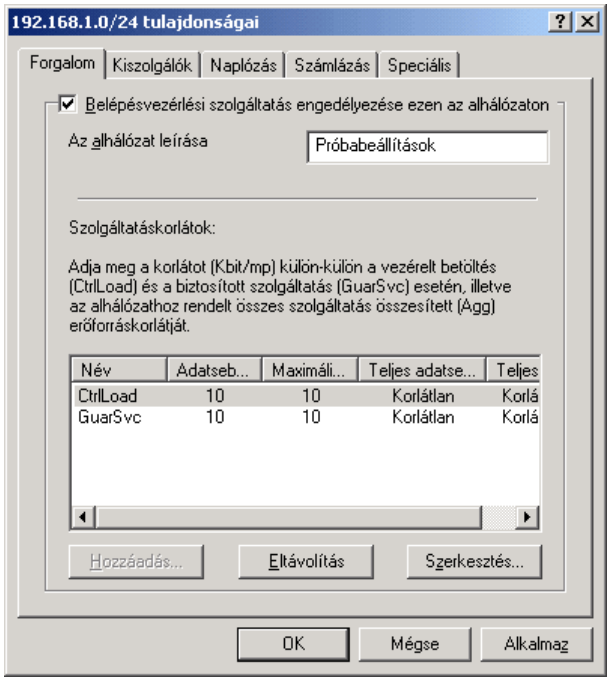
„Forgalom” fül:

A QoS szolgáltatás engedélyezését és tiltását a "Belépés vezérlési szolgáltatás engedélyezése ezen a hálózaton" jelölőnégyzet be- illetve kikapcsolásával lehet elérni. "Az alhálózat leírása" mezőben egy tetszőleges szöveget helyezhetünk el megjegyzésszerűen, a szolgáltatás működését nem befolyásolja. A "Szolgáltatáskorlátok" listába vehetünk fel értékeket a sávszélesség szabályozásához a "Hozzáadás" gombbal. Ekkor megjelenik egy újabb ablak, ahol szolgáltatástípusonként adhatók meg az értékek. Az ablak egyetlen lenyíló mezőjében lehet kiválasztani az adott típust, amely a következő értékeket veheti fel (3.3-as ábra): Szavatolt szolgáltatás (GuarSvc): A rendszer garantálja az átvitel minden pontján a megadott minimális sávszélességet.

Vezérelt betöltés (CtrlLoad): Nagy hálózati forgalom esetén adattorlódásnál csomagok csak kismértékben késhetnek és csak kevés számú csomag eldobása engedélyezett.

Összesítés (Agg): A kettő kombinációja.

A maximált sebességeket kilobit/másodpercben a megfelelő rádiógomb aktiválásával lehet kiválasztani, amelyet megadhatunk kommunikációs folyamatonként. Az összesített adatsebességek a folyamonkénti adatsebességek összegét jelentik. Megadhatjuk külön a CtrlLoad és GuarSvc értékeit is, de a kettőt együtt is az Agg hozzáadásával, miután ennek megfelelően kitöltöttük az értékét a lista teljessé vált, nem adhatunk hozzá új elemet.



3.3-as ábra.

„Kiszolgálók” fül:

Egy alhálózaton belül több kiszolgáló is futtathatja a QoS belépésvezérlést, amelyeket a "Kiszolgálók" oldalon található listába kell felvenni a "Hozzáadás" gombra kattintva. Ilyenkor megjelenik egy dialógusablak, ahol kéri tőlünk a rendszer a tartománynevet és meg kell adnunk egy jelszót. Ehhez tudnunk kell, hogy a belépésvezérlés telepítéskor létrejön egy "AcsService" felhasználói fiók a QoS bejelentkezések használatához. Ez azonban rejthet magában veszélyeket is, ha illetéktelen felhasználók ezzel a fiókkal kísérelnek meg bejelentkezni a rendszerbe. A szolgáltatás több kiszolgálón való futtatása folyamatos működést biztosít, arra az esetre is, ha ezek közül valamelyik üzemképtelenné válna.

„Naplózás" fül:

Kétféle naplózást használhatunk, az egyik a rendszer Eseménynapló szolgáltatásába rögzíti az adatokat ("Rendszeresemények naplója") a másik pedig az RSVP protokoll üzeneteit tárolja. Ez utóbbi alapértelmezésben nincs engedélyezve, bekapcsolásához jelöljük be az "RSVP üzenetek naplózása engedélyezése" jelölőnégyzetet. A "Naplófájl helye" mezőben található elérési úton az "RSVPTRACExx" nevű fájlokat kell keresnünk, ahol xx a fájl sorszáma. A

naplófájl "Maximális fájlméret"-ének megadásakor vegyük figyelembe, hogy 1 MB-os fájlban kb. 500 RSVP üzenet tárolódik. Az Eseménynapló szolgáltatásába történő rögzítést kikapcsolni nem lehet, de a rögzítendő mennyiséget szabályozhatjuk a "Naplózási szint" lenyíló menüben 0-tól 3-ig terjedő sorszámozással, ahol a 0 jelenti a minimális, 3 pedig a maximális mennyiségű naplózási információt.

„Számlázás" fül:

Szintén egy naplózási szolgáltatás, ami hibakeresési céllal rögzíti a QoS forgalmi adatait. Beállításaira ugyanaz vonatkozik, mint az RSVP üzenetek naplózására. A naplófájlban található rekordok felépítése a következő: dátum időpont:GMT azonosító, fogadó IP címe:portszáma, [protokoll azonosító]; eseménytípus; tartomány\felhasználó; az utolsó ugrás IP címe:portszáma; üzenet állapota; üzenet adatok

„Speciális" fül:

Itt állíthatók be a DSBM (Designated Subnet Bandwidth Management ~ Megjelölt alhálózati sávszélesség kezelő) tulajdonságai, úgymint:

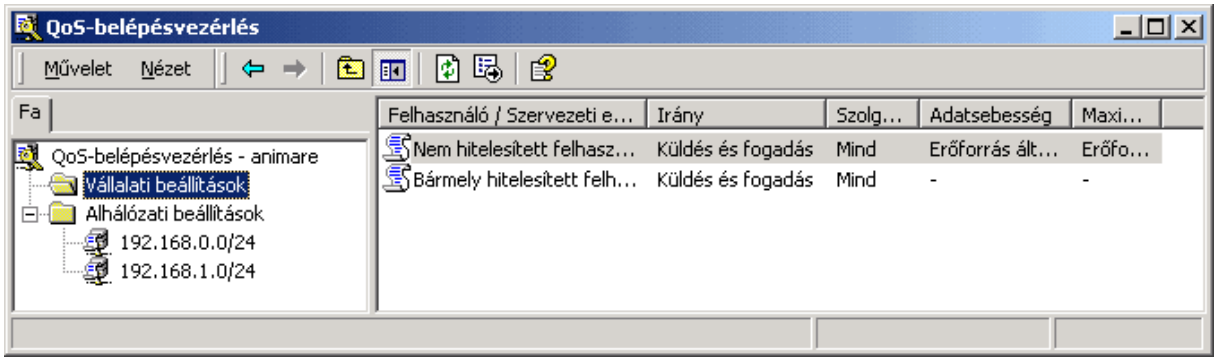
"Választási prioritás": Több QoS kiszolgáló esetén ki lehet választani egyet a sávszélesség kiosztás elvégzésére úgy, hogy a többinél alacsonyabb prioritás értéket adunk neki. Ha ez a kiszolgáló nem tudja teljesíteni a kérést, akkor a sorrendben következő prioritású kapja a feladatot. Alapértelmezés szerint a hálózat összes kiszolgálója 4-es prioritás értéket kap, ez automatikus választást jelent.

"Életben tartási időtartam": Egy sávszélesség foglalás ennyi ideig használható megújítás nélkül.
"Csendes időtartam": Az utolsó üzenetküldési után eltelt idő maximális hossza.
"Helyi házirend-gyorsítótár időkorlátja": Ennyi percenként történik meg a házirend változások ellenőrzése és az új változtatások életbe léptetése.

"Foglalás előtti adatsebesség": Kilobit/s formában azt az adatsebességet jelenti, amellyel akkor zajlik a kommunikáció, amikor még nem jött létre a QoS házirendnek megfelelő adattovábbítás elérése.

Vállalati beállítások:

Ezen szekció segítségével felhasználónként vagy szervezeti egységekként lehet további QoS beállításokat megadni az egész hálózatra vonatkozóan. Ezek egy alapbeállítást képeznek az alhálózati házirendek számára. Két alapértelmezett házirendet telepítés után már tartalmaznak, az egyik a hitelesített a másik pedig a nem hitelesített felhasználók adatforgalmi korlátozásait írja le. Ez utóbbi csoport tagjai vannak szűkebb keretek közé szorítva, ők 64 Kilobit/s sávszélességet kapnak egyszerre csak egy engedélyezett adatfolyam mellett. Míg a másik csoportnál ez 500 Kilobit/s és két darab adatfolyam lehet. Kattintsunk rá valamelyikre a kettő közül, utána pedig a Művelet > Tulajdonság parancsára, hogy megjeleníthessük a lehetséges értékeket.



3.4-es ábra.

"Általános" fül:

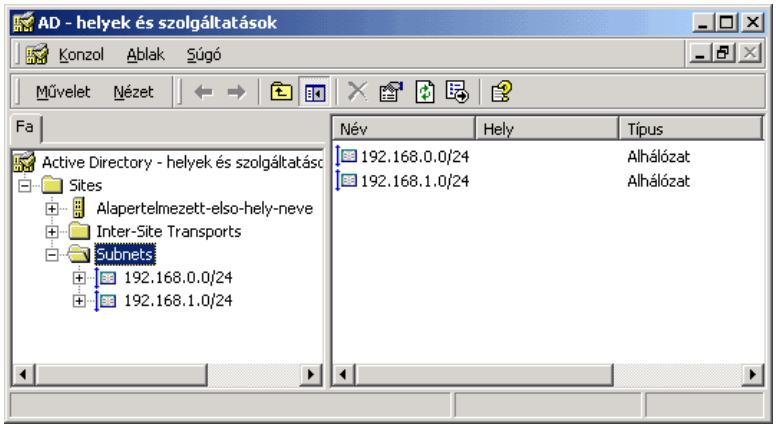
"Áramlás irány": A házirend csak az itt bejelölt irányú forgalom esetén lép életbe, ez lehet küldés, fogadás és mindkettő. Második pont a "Szolgáltatás szintje", amelyről az alhálózati beállítások "Forgalom" oldalának tárgyalásakor szóltunk. Az "Azonosító" szekcióban kell kijelölni azt a felhasználót vagy szervezeti egységet, amelyre a házirend vonatkozik. Nem hitelesített felhasználók közé sorolhatók azok, akik hozzáférhetnek bizonyos publikus erőforrásokhoz, de önálló felhasználói fiókkal nem rendelkeznek. Ugyanakkor a "Bármely hitelesített felhasználó" minden Active Directory felhasználói fiókra vonatkozik. A "Felhasználó" és "Szervezeti egység" kitöltését a mező neve melletti "Tallózás" gombbal végezhetjük el, eredményül egy szabványos LDAP szintaxist kapunk, azonban nem kötelező ezt a formátumot használni, beírhatjuk csak egyszerűen a felhasználó bejelentkező nevét is.

"Áramlási határok" fül:

Az adatsebességi szabályozások eltérnek az alhálózati beállításoknál szereplőktől. A folyamonkénti adatsebességen és maximális adatsebességen kívül megjelent egy időbeni korlátozás, ahol percben megadható a kapcsolat szintjének garantálása.

"Összesített korlátok" fül:

Ezen az oldalon tudjuk beállítani a kiszolgáló és kliens között maximálisan kialakítható adatfolyamokat az "Adatfolyamok száma" beállításcsoportban. Ahol a korlátlan = tetszőleges számú adatfolyam, de egy bizonyos szám felett már nem garantálható a minimális sávszélesség, úgyhogy ezzel a beállítással vigyázni kell. Alapértelmezett = hitelesített felhasználóknál: 2 db, nem hitelesítetteknél: 1 db adatfolyam. Egyéni = mi adhatjuk meg az adatfolyamok számát. Az adatsebességi és maximális adatsebességi beállítások itt az adatfolyamok összegére vonatkoznak.



3.5-ös ábra

Új házirend hozzáadása:

Kattintsunk a konzol "Vállalati beállítások" mappájára és a Művelet > Házirend hozzáadása menüpontra. Megjelenik egy az előzőekkel megegyező tulajdonságablak, ahol választhatunk, hogy a hitelesített, nem hitelesített, tetszőleges felhasználó vagy szervezeti egységre legyen érvényes a házirend. Azonos felhasználói fiókra (fiókokra) vagy szervezeti egységre vonatkozóan nem hozhatunk létre több házirendet, mert ilyenkor ütközés lép fel, amit a

rendszer sem összegzéssel, sem öröklődéssel nem old fel. Ilyenkor figyelmeztetés után döntenünk kell, hogy az azonosak közül melyiket tesszük érvényessé.

Házirendek törlése:

A "Vállalati beállítások" törlését az adott házirendre kattintva a Művelet > Törlés parancsával lehet elvégezni. Az "Alhálózati beállítások" mappába felvett házirendnél azonban nem ilyen egyszerű a helyzet. Két lehetőségünk van: az egyik, hogy eltávolíthatjuk az alhálózatban szereplő házirendobjektumokat. Ehhez kattintsunk az adott alhálózatra és a Művelet > QoS ACS házirendek törlése menüparancsra. Ilyenkor az alhálózat üresen, de megmarad a konzolban és itt nem is lehet eltávolítani.

Második lehetőségünk, hogy töröljük a teljes alhálózatot a benne szereplő összes házirenddel együtt a Felügyeleti eszközök > Active Directory - helyek és szolgáltatások konzol megnyitásával. Tallózunk el a Sites > Subnets mappáig, nyissuk ki, ekkor a konzol jobb oldalán megjelennek az Active Directory által kezelt alhálózatok (ez megegyezik a QoS konzolban felvettekkel), kattintsunk rá a megfelelőre és a Művelet > Törlés parancsára, ennek hatására a QoS konzolból is kikerül (ne felejtsük el frissíteni a nézetet) [4][5][6].

7. Összefoglalás

Dolgozatomban a rendelkezésre álló magyar szakirodalom és publikáció, és némi külföldi szakirodalommal sikerült átfogó képet adnom a minőségi szolgáltatásról, egészen az alapmechanizmusoktól kezdve. Ismertettem a multimédiás alkalmazások QoS igényeit, jellemzőit és ezeket az igényeket kielégítő technológiákat, majd az integrált és differenciált szolgáltatásait a minőségi szolgáltatásnak, végül egy konkrét példán egy lehetséges QoS beállítást a Windows operációs rendszeren.

Munkám során a témával kapcsolatos interneten található információkra és a hivatalos protokollok RFC dokumentumaira támaszkodtam.

Dolgozatom összefoglalásaként megemlíteném, hogy nem tértem ki részletesen a CISCO IOS összes QoS parancsára, ezeket a parancsokat és részletes leírását a parancsoknak egy majd 50 oldalas dokumentum tartalmazza.

Szakdolgozatom összefoglalásaként megemlíteném, hogy napjainkban a QoS kisebb szerepet tölt már be a hálózatoknál, mert a felhasználóknak több megabites sávszélességet biztosítanak, amit az esetek döntő többségében egyáltalán nem használnak ki teljes mértékben. Ennek ellenére a QoS a mai napig alkalmazott technika, mert a segítségével minimális sávszélesség biztosítható a különböző multimédiás programok számára, amivel biztosítható a minimális élvezhetőség a felhasználói oldalon.

A bemutatott elgondolások még nem teljesen kiforrottak és sokat változhatnak még, jelenleg is fejlesztenek szolgáltatás minőséggel és szabályozással foglalkozó protokollokat.

A dolgozatom zárásaként úgy vélem sikerült egyszerű, átlátható képet adnom egy aktuális hálózati probléma megoldására a minőségi szolgáltatásra, és remélem ezzel nagyban hozzájárultam a témával foglalkozó lelkes érdeklődők kérdéseinek kielégítésére.

8. Irodalomjegyzék:

[1] Andrew S. Tanenbaum - Számítógép-hálózatok, Panem Kiadó, 2003.

[2] Telbisz Ferenc - Internet QoS: lehetőség vagy délibáb?

https://nws.niif.hu/ncd2001/docs/eloadas/16/index.htm

[3] Telbisz Ferenc – Merre Tart az Internet?

http://www.iif.hu/rendezvenyek/networkshop/98/eloadas/html/b/ftelbisz/ftelbisz.htm

[4] Microsoft: A Windows XP garantált szolgáltatásminősége

http://support.microsoft.com/kb/316666/hu

[5] Software online – QoS bemutatása 1

http://www.softwareonline.hu/Article/View.aspx?id=2668

[6] Software online – QoS bemutatása 1

http://www.softwareonline.hu/Article/View.aspx?id=2677

[7] Hoc fórum – QoS felülírása

http://www.hoc.hu/forum/index.php?showtopic=9323

[8] Jim Kurose, Keith Ross - Computer networking

http://www.cs.huji.ac.il/course/2005/com1/Presentations/Lessons/qos.pdf

[9] RFC dokumentumok

http://www.rfc-editor.org

www.ietf.org/rfc.html

[10] McMahon, Richard A. – PC hálózatok a gyakorlatban, Panem Kiadó, 2004.

[11] Megyeri Péter – Torlódásvédelem alapjai

htttp://users.prx.hu/kacsa/Suli/Inf%20rendszerek%202/SzGhalE3.ppt

[12] RSVP alapú webkiszolgálás

http://www.hit.bme.hu/~mihaly/atmc/webqosfe.htm

[13] Integrated Services Internet

http://www.szabilinux.hu/trendek/trendek75.html

[14] Lengyel Miklós, Sztrik János - Diffserv emuláció és szimuláció

http://www.agr.unideb.hu/if2008/kiadvany/papers/F32.pdf

[15] Garantált szolgáltatásszint

http://rs1.szif.hu/~heckenas/okt/QoS.pdf

[16] QoS, a Best Effort-on túl

http://www.hit.bme.hu/~szabo/slides/dif_int_serv.pdf

[17] Gál Zoltán - Adatátviteli hálózatok QoS jellemzése

http://www.inf.unideb.hu/kutatas/gybin/gybin09/Gal_Zoltan.ppt

[18] Cisco QoS parancsok

http://www.cisco.com/en/US/docs/ios/12_1/qos/configuration/guide/qcdrsvp.html

http://www.cisco.com/en/US/docs/ios/12_1/qos/configuration/guide/qcdpq.html

http://www.cisco.com/en/US/docs/ios/12_1/qos/configuration/guide/qcdgts.html

http://www.cisco.com/en/US/docs/ios/12_2/qos/command/reference/qrfbook.pdf

[19] Sztrik János – Kiszolgálási problémák matematikai modellezése

http://irh.inf.unideb.hu/user/jsztrik/education/14/opkut.pdf

[20] Garzó András – Torlódásvédelem

http://www.linuxvilag.hu/content/files/cikk/53/cikk_53_73_76.pdf

[21] IEEE Standard for Local and metropolitan area networks

http://standards.ieee.org/getieee802/download/802.1Q-2005.pdf

[22] Szenes Tamás – RSVP erőforrás-lefoglalási protokoll

http://www.omikk.bme.hu:8080/cikkadat/bitstream/123456789/484/1/2002_6bol3.pdf

9. Köszönetnyilvánítás

Szeretnék köszönetet mondani témavezetőmnek, Dr. Almási Bélának a megfelelő szakmai segítségnyújtásért, tanácsaiért, melyek nagyban hozzájárultak a dolgozatom elkészítésében és a hozzám való türelméért. Köszönettel tartozom mindazoknak, akik ötleteikkel és tanácsaikkal segítették munkámat. Az olvasóknak sikeres problémamegoldást kívánok a minőségi szolgáltatáshoz.