A Zululandi Egyetem internetes tárkorlátrendszere

Webhasználatot felügyelő tárkorlátrendszer megvalósítása Squid címátirányítással.

A Zululandi Egyetem körülbelül 6000 hallgatót számláló, „történelmileg hátrányos helyzetű” egyetem Dél-Afrika észak-keleti partjainál. A hátrányos örökség még a faji elkülönítés megszüntetése után is érezteti hatását, tanulóink főként feketék és hátrányos anyagi helyzetűek. Következésképpen költségvetésünk meglehetősen szűkre szabott, így mindennapi kihívás, hogy ebből a kevésből minél többet hozzunk ki. Ilyen helyzetben nyilvánvalóan a Linux a megfelelő választás az internetes szolgáltatások megvalósítására – mi gyakorlatilag mindenre ezt használjuk: levél-, www-kiszolgálók, DNS, DHCP, HTTP-proxik, tűzfalak és telefonos bejelentkezés. A vezetőségnek nagyon tetszik a Linux ingyenessége, ám még ennél is nyomósabb okunk van arra, hogy Linuxot használjunk – az internetes szolgáltatások kivitelezésére a Linux a legmegfelelőbb rendszer.

Az internet-hozzáférés költségei

2000 elején a tanárok és tanulóink számára a legnagyobb anyagi kihívást az internet-hozzáférés biztosítása jelentette. Az internetes adatforgalom költségei sokkal magasabbak Dél-Afrikában, mint az Egyesült Államokban. A 128 Kb-es hozzáférésünk mintegy 5000 amerikai dollárba került havonta. Ezt a sávszélességet már napközben teljesen leterhelte az a 400 oktató, akinek internet-hozzáférése volt, ezen felül a hozzáférést még a forgalmas hallgatói laborokban lévő 350 munkaállomás számára is meg kellett teremtenünk.

Nagyobb sávszélességre volt szükségünk, de a költségvetés nem tette lehetővé, hogy a jelenlegi sávszélességet megkétszerezzük vagy megháromszorozzuk. Az internethasználatot figyelő és korlátozó rendszer hiányában a vezetőség nem szívesen egyezett bele az internet-hozzáférésre fordított összeg növelésébe. Vonakodtak attól, hogy nagy összegeket fizessenek ki havonta, amikor azt sem tudták, hogy kik és milyen céllal használják a Hálózatot.

Tárkorlát és hitelezés

A gondot az okozta, hogy – költségvetési keretünkön belül maradva – hogyan tudnánk elfogadható minőségű www-szolgáltatást biztosítani. Mivel az internet-hozzáférést nem szabályoztuk, az adatátviteli csatornák túlterhelése volt az egyetlen faktor, ami az igényeknek határt szabott, és ez olyan alacsony minőségű szolgáltatáshoz vezet, amit az internetfelhasználók kemény magja éppen csak elvisel.

Be kellett vezetnünk valamilyen internethasználati „költséget” annak érdekében, hogy szabályozhassuk az igényeket. A rendszer kézzelfoghatóvá tenné felhasználóinknak, hogy mibe is kerül az egyetemnek az internet-hozzáférés biztosítása. A hallgatók számára a költség előre fizetendő. Mivel a tandíjadósságok komoly nehézséget jelentenek a dél-afrikai hátrányos helyzetű egyetemek esetében, nem akartuk a helyzetet még internetes számlákkal is rontani. Ezért egy az internethasználatot szabályozó tárkorlátrendszer kidolgozása vált szükségessé. A sávszélességkorlátok azóta fellazultak, mivel tavaly új hálózati megoldást vezettek be a felsőoktatási intézményekben. Sávszélességünk 768 Kb-es lett (384 Kb-es CIR a nemzetközi forgalomban), mialatt a havi költségek nagyjából ugyanazon a szinten maradtak. Azonban továbbra is meg voltunk győződve arról, hogy a tárkorlátrendszerre szükségünk van, ha a felhasználóknak megfelelő minőségű szolgáltatást akarunk nyújtani.

Squid és címátirányítás

Az NLANR népszerű Squid HTTP-proxy programját használjuk www-szolgáltatásainkhoz. Egy tűzfal gondoskodik arról, hogy az oktatók és a diákok csak a proxyn keresztül férhessenek hozzá a Webhez, és a proxykat úgy állítottuk be, hogy a www-használat csak felhasználói név és jelszó megadása után lehetséges. Így a Squid által létrehozott ***access.log*** állomány valamennyi felhasználó www-használatának bejegyzéseit tartalmazni fogja. Az ***access.log*** állományban lévő adatok alapján egy adott felhasználó www-használatának nyomon követése egyszerű feladat.

A Squid képes külső címátirányító programokkal együttműködni. A címátirányító beállítása után minden címkérelem, amit a proxy megkap, az átirányító program alapértelmezett bemenetére kerül. A program kétféle választ adhat: vagy megkéri a proxyt, hogy töltse le a kért URL mögötti erőforrást, vagy a kérést egy másik címre irányítja át. Ez a működésmód jellemző az olyan tartalomszűrő programokra is, mint a SquidGuard, ebben az esetben a címátirányító minden egyes címkérelmet összeveti egy olyan adatbázissal, amely a kiszűrendő címmintákat tartalmazza. Minden olyan kérelem, amely egy nem engedélyezett URL-re irányul, átirányítódik egy oldalra, ami közli a felhasználóval, hogy a kért oldalhoz a hozzáférés nem engedélyezett.

Rájöttünk, hogy a címátirányítót arra is felhasználhatjuk, hogy a felhasználói tárkorlátrendszert hatályba léptessük. A címátirányító ellenőrzi a felhasználó tárkorlátját, és amennyiben az engedélyezett keretet túllépte, egy olyan oldalra irányítja őt át, ami figyelmezteti erre. Bár elméletben egyszerűen hangzik, mindezeknek műveleteknek nagyon gyorsan kell lezajlaniuk, különben a proxykiszolgáló teljesítménye nagymértékben csökkenhet. A mi rendszerünkben a címkiszolgáló általában két ezredmásodpercen belül válaszol, amennyiben a kért URL engedélyezett.

A QUORUM kiszolgáló

A QUORUM (QUOta-based Resource Usage Manager, azaz tárkorlátalapú erőforráshasználat-mérő) egy kiszolgáló, egyszerű üzenet-API kétfajta üzenettípus számára: feljegyzés (egy adott felhasználó, illetve munkamenet számlázása egy adott tevékenységért) és lekérdezés (megkérdezzük a kiszolgálót, hogy egy adott felhasználó vagy munkamenet nem lépte-e túl a keretét).

A QUORUM kiszolgáló kezeli a proxy által lehívott összes cím feljegyzését, figyeli a felhasználók hitelkeretének alakulását, valamint követi a felhasználói munkafolyamatokat, amelyek egy meghatározott időtartamú folyamatos www-használatnak felelnek meg. A kiszolgáló a felhasználói és

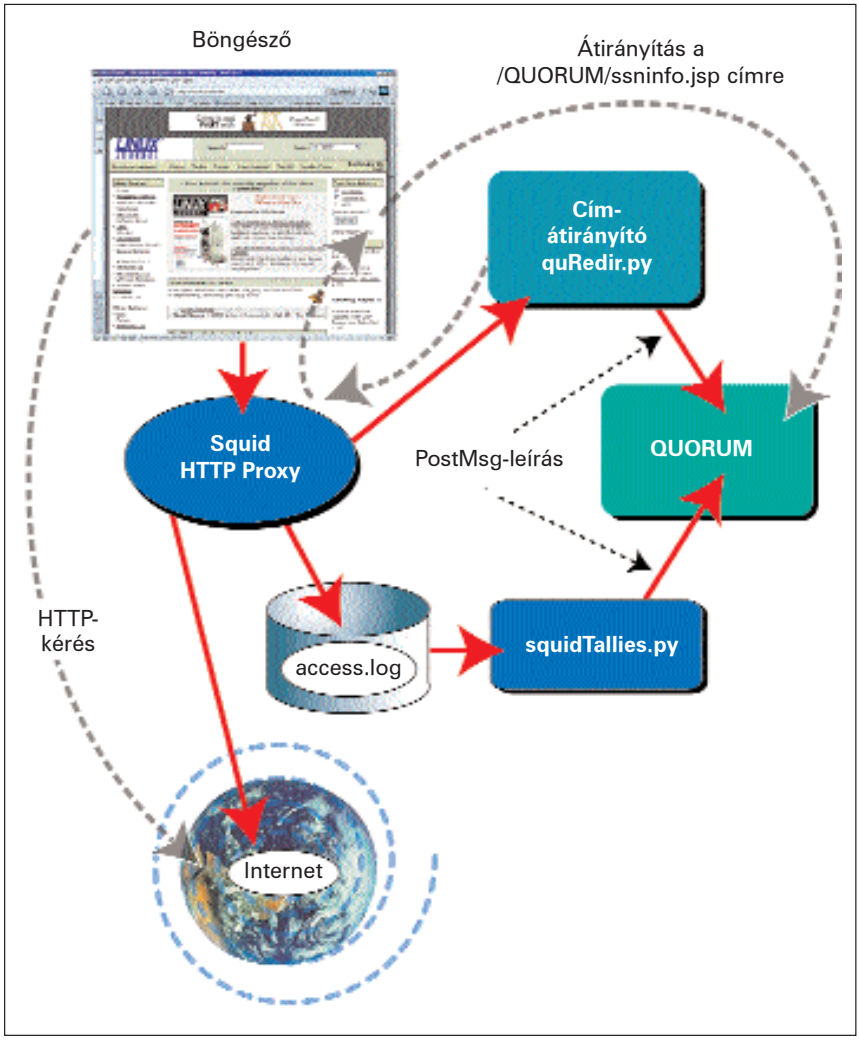
munkamenet-feljegyzéseket, valamint a hitelkeret-követéseket egy gyorsítótárban tartja. Adott időközönként a tár a megváltozott adatai kiíródnak egy adatbázisba. A kiszolgáló egy webalapú felületen jeleníti meg a felhasználható keretet, a felhasználói munkafolyamatokat, valamint az egyéb felügyeleti és hibakereséshez szükséges adatokat.
A kiszolgáló egy korai változatú C-ben íródott, és különböző CGI-programokon keresztül biztosította a webalapú felületet. Felhagytunk azonban ezzel a megközelítéssel, amikor találkoztunk a Java servletekkel. A jelenlegi QUORUM kiszolgáló Java servletek és JSP-k gyűjteménye, ami JDBC-n keresztül valósítja meg az adatbázissal való kapcsolódást.
A kiszolgálót a Caucho-féle Resin Servlet/JSP konténer alatt futtatjuk. Adatbázis-kezelőnek a MySQL-t választottuk, amelyhez a Resin beépített JDBC-osztályokat és adatbázis-kapcsolatcsoport létrehozást biztosít.
A Java servletek természetes választásnak tűnnek, amikor olyan alkalmazást akarunk készíteni, mint a QUORUM. A kiszolgáló számos, a munkafolyamatok számára szükséges állandó megosztott adatszerkezetet, munkafolyamatonkénti feljegyzéseket, valamint felhasználói feljegyzéseket és hitelkeret-feljegyzéseket tartalmaz. Mivel a servletek egyszerű Java-osztályok, amelyek egy www-kiszolgálót megvalósító konténer részei, ezeket az állandó objektumokat egyszerű a kiszolgáló részeként létrehozni, csakúgy, mint a háttérben futó szálakat, amelyek a tétlen munkafolyamatokat szűrik ki, és a megváltozott adatokat adott időközönként kiírják az adatbázisba.

Hogyan működik?

Amikor a felhasználó címet kér le, a böngésző elküldi az URL-re vonatkozó kérelmet a Squid proxinak. A Squid ezt a kérelmet továbbküldi az URL-átirányítónak. A címátirányító a `querySsn` üzenettel megkérdezi a QUORUM kiszolgálót, hogy a szóban forgó felhasználóhoz tartozik-e pillanatnyilag QUORUM-folyamat, és hogy a felhasználó rendelkezik-e még elegendő hitelkerettel. Ha minden rendben van, a címátirányító egy üres sort küld a Squidnek, amellyel megkéri, hogy hívja le az adott címet.
Ha a felhasználónak nincsen munkafolyamata vagy elfogyott a hitelkerete, a böngésző a QUORUM kiszolgálóban lévő ***ssnInfo.jsp***-re lesz átirányítva, ami megjeleníti a felhasználó adatait, a még meglévő hitelkeretet és megkéri, hogy a ***beginSsn.jsp***-re mutató hivatkozásra kattintva indítson el egy QUORUM-munkafolyamatot. Ha a felhasználónak még van hitelkerete, a JSP egy QUORUM-munkafolyamatot nyit meg a számára.
Valójában az történik, hogy a címátirányító átirányítási üzenetet küld egy servletnek, amely egy újabb átirányítási üzenetet küld az ***ssnInfo.jsp***-re. A címátirányító a felhasználó nevét és IP-címét a kérés jellemzőjében adja át a servletnek:

```
/QUORUM/servlet/Redirect?ssn_id=uname@ipaddr
```

A servlet az `ssn_id` változót a felhasználó böngészőjéhez tartozó munkafolyamatba menti. Ez a többi JSP számára lehetővé teszi, hogy egy HTTP-munkamenetváltozót használjanak a felhasználóval és a munkamenettel kapcsolatos adatok lekérésére.
A webhasználat könyvelése a Squid ***access.log*** állományában lévő adaton alapul. Miután a Squid lekért egy URL-t, hozzáfűz egy sort az ***access.log*** állományhoz, ami tartalmazza a lekért



A webhasználat tárkorlátainak megvalósítása

címet, az erőforrás méretét bájtban, a lekérdező felhasználói azonosítóját, és annak a munkaállomásnak az IP-címét, amelytől a kérés származik. A ***squidTallies.py*** nevű Python-parancsállomány beolvassa az ***access.log*** utolsó sorát, és létrehoz egy `tallySsnItem` nevű kérésüzenetet, amit a QUORUM kiszolgáló kap meg. A QUORUM feldolgozza ezeket az üzeneteket, és feljegyzi a felhasználói adatokat és munkafolyamatokat, valamint az összes tevékeny felhasználó hitelkeretét.
A hitelkeret, a használat könyvelése és a felhasználói munkafolyamatok feljegyzései egy MySQL-adatbázisba íródnak.
A QUORUM kiszolgáló egyik háttérben futó szála megadott időközönként átnézi a gyorsítótárban lévő munkamenet-bejegyzéseket, a felhasználó- és hitelkeretadatokat, és minden

megváltozott adatot kiír az adatbázisba. Ez lehetővé teszi, hogy a kiszolgáló mindig viszonylag friss adatokat tároljon az adatbázisban, anélkül, hogy túlterhelné az adatbázist azzal, hogy minden egyes címlekéréskor többszörös frissítést hajt végre.

A PostMsg servlet

A QUORUM kérés üzenetei egyszerű szöveges üzenetek, amelyeket egy újsor karakter zár. Például a

```
ref000001 querySsn ssn_id=soren@1.2.3.4
➥ccode=11000
```

üzenet egy kérésüzenet, ami megkérdezi a kiszolgálót, hogy az `1.2.3.4`-es IP-címen lévő soren nevű felhasználóhoz pillanatnyilag tartozik-e QUORUM-munkafolyamat és a 11000 hitelkódhoz tartozik-e felhasználható hitelkeret. Valamennyi kérésüzenet első mezője egy felhasználói referencia, amelyet a válaszüzenetben visszaküldünk az ügyfélnek, így téve lehetővé, hogy a többszálú ügyfelek összekapcsolják a válasz- és a kérőüzeneteket.

Mivel egy J2EE-alkalmazásról van szó, egy olyan üzenettovábbítással van dolgunk, mint például a HTTP-n keresztüli SOAP. Fontolgattuk, hogy valamennyi kérést független HTTP GET üzenetként kódolnánk a servlet számára, és aztán ezekből jönnének létre a különböző QUORUM üzenet API-k. Ez a megoldás jobb illeszkedést eredményezett volna a servlet keretrendszerhez, de a próbák azt mutatták, hogy a HTTP overhead az alkalmazásunkhoz túl nagy. Csúcsidőben a hallgatói laborokból 20–30 címkérés érkezik másodpercenként. Mivel mindegyik címkérés két QUORUM-kérésüzenetet eredményez (egy `querySsn` és egy `tallySsnItem` üzenetet), a kiszolgálónak másodpercenként 60 kérést kell kezelnie. Miután egyetlen QUORUM kiszolgálót használunk a tanári és diák proxyk kezelésére, legalább 100–150 kérelmet kell kezelnünk másodpercenként. Ezt a teljesítményt nem érhettük volna el, ha a QUORUM-kérésüzenetek megvalósítására független HTTP-üzeneteket használtunk volna, még akkor sem, ha oda-vissza üzenetekkel dolgoztunk volna egyetlen állandó HTTP-kapcsolaton keresztül. Ehelyett egyetlen servlet, a ***PostMsg*** kezeli le az ügyfelektől érkező összes QUORUM-kérésüzenetet. Egy ügyfélalkalmazás létrehoz egy HTTP POST-kérést, majd a POST-kérés TCP-kapcsolatát használva QUORUM-kérésüzeneteket küld, mintha egy állomány vagy más adat feltöltését végezné el a kérésben. A ***PostMsg*** servlet soronként egy kérőüzenetet olvas be a `ServletInputStream`-ből, a `readLine()` tagfüggvény segítségével. Feldolgozza a kérésüzeneteket, és válaszüzeneteket küld vissza az ügyfélnek a `ServletOutputStream`-en keresztül. A válasz elküldése után meghívja a `ServletHTTPResponse` objektum `flushBuffer()` tagfüggvényét, ami biztosítja, hogy a válasz tényleg el lesz küldve az ügyfélnek. Ezzel a módszerrel egyetlen QUORUM kiszolgáló több ezer kérést képes kezelni másodpercenként, egyetlen ügyfélkapcsolaton keresztül.

Két apró gond azonban van ezzel a módszerrel. Az első, hogy a servlet-konténer minden olyan HTTP-kapcsolatot lezár, ami egy megadott időtartamon, általában 30 másodpercen keresztül nem működik, vagyis az ügyféllel megszakítja a kapcsolatot, ha az bizonyos idő elteltével nem küld újabb kéréseket. Az ügyfélnek ilyenkor újra kell kapcsolódnia, és el kell küldenie egy újabb HTTP POST-üzenetet, mert csak ezután lesz képes QUORUM-kérésüzenetek küldésére. Írtunk egy Python-osztályt, ami egy Unix-csővezetéket hoz létre a kérések elküldésére és a válaszok olvasására. Kérésre az osztály létrehozza a HTTP-kapcsolatot, és elküldi a HTTP POST-kérőüzenetet. A kérésre való kapcsolódás előnye, hogy az üzenetküldések folyamata nagyméretűvé válik – a nem működő kapcsolatok nem maradnak fenn sokáig, és nem lép fel az a hiba, hogy az ügyfelek beragadnak, mivel azt hiszik, hogy a kapcsolat még él, holott a kiszolgáló már lezárta őket.

A másik felmerülő gond, hogy a HTTP/1.1 megköveteli, hogy a ***Content-length*** fejlécet a POST-üzenetek mind a kérő-, mind a válaszüzenetekben tartalmazzák. Míg a Tomcat ezt nem vette a figyelembe, a Resin igen, és minden olyan kapcsolatot lezárt, ahol az ügyfél vagy a kiszolgáló által küldött bájtok túllépték a ***Content-length*** értékét. Ezt a csapdát úgy kerültük meg, hogy a ***Content-length*** fejlécben igen nagy értéket állítottunk be, és megbizonyosodtunk arról, hogy az ügyfél még jóval azelőtt lezárja a kapcsolatot, mielőtt a küldött bájtok a megadott értéket túllépnék.

A címátirányító teljesítménye a proxyk szempontjából kényes, hiszen minden címkérelemnek várakoznia kell, amíg a címátirányító válaszol. A QUORUM címátirányítója a legutóbbi munkafolyamatok `querySsn` válaszait egy gyorsítótárban tárolja. Ennek eredményeként az átirányító – ha egy munkafolyamat már létrejött és van hozzá elég hitelkeret – a címlekérdezések alkalmával nagyon gyorsan fog működni, hiszen válaszait a gyorsítótárban lévő legfrissebb üzenetek alapján hozza létre. Ebben az esetben a választ általában két ezredmásodpercen belül vagy még ennél is rövidebb idő alatt elküldi. Mégha a válasz benne is van a gyorsítótárban, az átirányító mindig küld egy `querySsn` kérést a QUORUM kiszolgálónak. A válaszüzenet frissíti a gyorsítóban lévő válaszokat, így a tár mindig friss adatokat tárol a felhasználói folyamatok állapotáról és a hitelkeretről.

Ha a szükséges válasz nincs a gyorsítótárban, vagy ha a válasz kedvezőtlen (a felhasználónak elfogyott a hitelkerete), az átirányító egy `querySsn` kérést küld el, és egészen addig nem küld választ a proxynak, amíg a QUORUM kiszolgálótól nem érkezett válasz. Ebben az esetben az átirányító mintegy 20–50 ezredmásodperc múlva küldi el a választ.

A Squidet úgy is beállíthatjuk, hogy a címátirányítót több példányban indítsa el, ezáltal téve lehetővé az átirányítási kérelmek párhuzamos feldolgozását. Mivel a QUORUM címátirányító több szálon fut, a Squid számos címkérelmet küldhet el egyszerre. A Squid valójában egy kisméretű beillesztett program példányait szólítja meg, amelyek egy Unix alapú hálózati kapcsolaton keresztül a címátirányítási kérelmeket a címátirányító folya-

matnak küldik tovább. Így a címátirányító valamennyi kéréshez ugyanazokat a tárolt `querySsn` válaszokat használhatja.

Munkába állítás és felügyelet

A felhasználói fiókok egyszerű kezelése nagy fontosságú a rendszer sikeres működtetéséhez. A QUORUM-ban ezt úgy valósítottuk meg, hogy a Web eléréséhez az összes diáknak felhasználói tárkorláttal kell rendelkeznie. A QUORUM kiszolgáló az oktatók általi használatot is feljegyzi, de rájuk egyelőre még nem terjesztettük ki a kvótarendszert (az oktatói használat feljegyzése nagyon értékesnek bizonyul a hallgatók által ellopott oktatói fiókok használatának nyomon követésére).
Az adott tanórán történő internethasználat esetén az összes hallgató hitelkeretét módosítjuk egy adott mértékben, a hallgatói bejegyzések adatbázisa alapján. Ezt minden félév elején megtesszük azoknak az óráknak az esetében, ahol az oktatók internethasználati kérelemmel állnak elő.
A múltbéli tapasztalatok alapján mindig vannak olyan diákok, akik késve jegyeztetik be magukat, vagy valamilyen okból nem kerültek be a bejegyzettek adatbázisába, vagy egyszerűen csak nagyobb felhasználói tárkorlátra van szükségük. Ezért néhány tanszék számára további felhasználói fiókokat hoztunk létre. Az arra feljogosított oktatók szükség szerint további hiteleket irányíthatnak át ezekről a tanszéki fiókokról az egyes hallgatók fiókjaira. A mi tanszékünk így egy súlyos felügyeleti tehertől szabadul meg, a többi tanszék számára viszont belátást enged a felhasználói fiókok kezelésének terheibe.
Azoknak a hallgatóknak a részére, akik a kiosztott tárkorláton felül további felhasználói hitelt kérnek, egy előre fizetett számlázási rendszert vezetünk be. Ez a rendszer nem véletlenül hasonlít a diákjaink körében nagyon népszerű kártyás mobiltelefonok számlázási rendszeréhez. A diákok az egyetemi könyvesboltban vásárolhatják meg a kártyát, ami egy titkos hozzáférési kódot tartalmaz. A számla bekapcsolható, ha a tanuló a kódot beírja egy webűrlapba, és elküldi. A beírt szám MD5 hashben tárolódik, ami összehasonlításra kerül egy adatbázistáblában tárolt hash-értékekkel, és ha a hash megegyezik valamelyik tárolt értékkel, a hallgató felhasználói fiókjára terhelődik a kártyán lévő hitelkeret. Kezdeti tárgyalásokat folytattunk az egyetemünk területén vásárlási pontokat felállító cégekkel, hogy kiderítsük, árulhatjuk-e az internetes tárkorlátkártyáinkat olyan rendszerben, mint amilyet manapság a pénztáraknál a mobiltelefonok kártyáinak árulásánál használnak.
A kártyás rendszer fontos eleme annak a célkitűzésünknek, hogy felhasználóinknak rugalmas internetszolgáltatást nyújthassunk, pénzzel támogathassuk az oktatási célú internethasználatot, és fedezhessük az egyéb használati költségeket. Jelen pillanatban ez a rendszer még csak próbaüzemben működik. Tanulóink részéről erős nyomás érezhető a rendszer minél előbbi bevezetésére, mi azonban meglehetősen óvatosak vagyunk éles bevetésével kapcsolatban, hiszen ez egy olyan szolgáltatás lesz, amelyért a hallgatók ténylegesen fizetni fognak. Fontosnak tartjuk egy jól megalapozott, világosan átlátható rendszer kidolgozását, hogy a hallgatók pontosan tudják majd, mire is adnak ki pénzt.

Tervek a jövőre nézvést

A QUORUM-ot már kezdetektől fogva olyan alkalmazásnak szántuk, ami túlmutat a webhasználat könyvelésén és a tárkorlátok kezelésén. Bármely szolgáltatás, ami egy naplóállományba írja a használattal kapcsolatos adatokat, feldolgozható egy olyan kis program által, ami egy feljegyzési kérelmet küld a QUORUM kiszolgálónak. A közeljövőben bevezetjük a levélküldés könyvelését, így a nagyméretű csatolmányokat tartalmazó levelek küldőjének hitelkeretét megterhelhetjük. A levelezéssel való visszaélés (egy nagyszerű AVI-film elküldése a haveroknak) visszatérő gond nálunk. A jövőben felhasználóink szembesülnek azzal, hogy ha nagyméretű állományokat küldenek levélben, elképzelhető, hogy ezért a tettükért fizetniük kell majd.
A használatot a QUORUM-ban költségkódokban tároljuk. Jelenleg a www-használat nemzetközi és belföldi forgalomra van lebontva, amelyek költsége különbözik, valamint a Squid gyorsítótárából teljesíthető kérelmekre, amelyeknek egyáltalán nincs költségük. A költségek felbontása hierarchikus: az internethasználat www-, illetve levélhozzáférésre bontható le, és ezek is további csoportokra oszthatók. A következőkben azt tervezzük, hogy költségkódokat vezetünk be a hálózati nyomtatásért, a telefonos bejelentkezésért és a tűzfalon keresztüli közvetlen TCP/IP-forgalomért. A költségkódok hierarchikus rendszere lehetővé teszi, hogy az összes szolgáltatáshoz egyetlen tárkorlátot használjunk, vagy egy adott tárkorlát hitelkerete a költségkódszerkezet csupán egy meghatározott részére vonatkozzon. Például elhatározhatjuk, hogy egy adott tanfolyam hallgatóinak tárkorlátja csak az internetelérésre vonatkozzon, továbbá az előre kifizetett kártyákat hálózati nyomtatásra is lehessen használni.
Internetes sávszélességünket megpróbáljuk úgy hatékonyan kihasználni, hogy felhasználóinkat bölcs használatra bíztatjuk. Használatfigyelő rendszerünk azoknak kedvez, akik a nagyméretű állományokat dél-afrikai tükörkiszolgálókról töltik le, szemben azokkal, akik a tengerentúlról teszik ugyanezt – a belföldi sávszélesség kevesebbe kerül, mint a nemzetközi. Ehhez hasonlóan kedvezményeket kínálunk az esti órákban, és arra bíztatjuk felhasználóinkat, hogy a nagy állományokat ebben a napszakban töltsék le, hiszen ilyenkor kevésbé terheltek a vonalak.
Érdekes ötletnek ígérkezik egy üzenő-, illetve faliújságrendszer beépítése a QUORUM-ba, amolyan helyi hálózatos azonnali üzenetküldést megvalósítandó rendszeré.
Azt már most is látjuk, hogy melyik felhasználó netezik az adott pillanatban. A QUORUM-ot úgy is kibővíthetjük, hogy üzenetet küldhessünk egy felhasználónak vagy a felhasználók egy csoportjának. Amikor valamelyik címzett címkérelmet küld el, a QUORUM átirányítja őt egy olyan oldalra, ami a faliújságot jeleníti meg.

Összegzés

Rugalmas könyvelő- és tárkorlátkezelő rendszert fejlesztettünk ki, ami együttműködik meglévő Linux és OSS alapú internetes szolgáltatásainkkal. Ez a rendszer fontos szerepet játszik abban, hogy egyetemünkön jól kidolgozott internetes szolgáltatást kínáljunk, hogy a segítségével ellenőrizhessük az internethasználatot, és hasznos adatokkal lát el minket a szolgáltatásokkal kapcsolatban felmerülő igényekről is.

Linux Journal 2002. november, 103. szám

Soren Aalto
(soren@pan.uzulu.ac.za) a zululandi egyetem Hálózati szolgáltatások osztályán dolgozik, ahol munkaidejében rendszerfelügyelettel, webalkalmazások fejlesztésével és mások Linuxra való áttérítésével foglalkozik. Felesége előadó az egyetemen. Két gyermekük, három macskájuk, egy kutyájuk és változó számú trópusi haluk van.