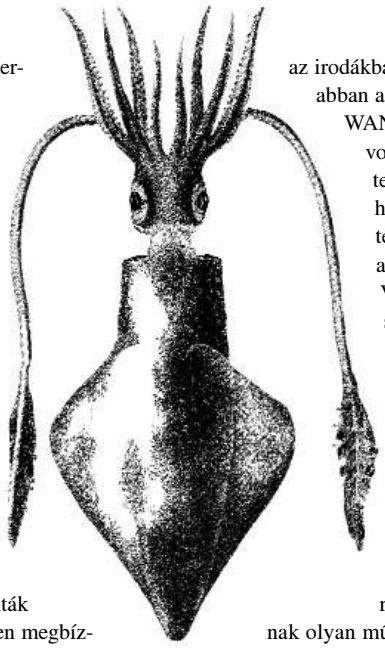
© Kiskapu Kft. Minden jog fenntartva

Squid proxykiszolgáló telepítése Linux alá

Egy példán keresztül bemutatjuk e „sokcsápos puhatestű” proxykiszolgáló telepítésének, beállításának és karbantartásának rejtelmeit.



A SAS intézet irodahálózata felhasználóinak internetelérését számos proxykiszolgáló biztosította világszerte, valamint a SAS európai központjában, a németországi Heidelbergben. Ezek a kiszolgálók a Squid (a szó jelentése tintahal) proxykiszolgáló programot futtatták, amely a GNU felhasználási szerződés hatálya alá tartozik. Dióhéjban, a Squid lehetővé teszi a gyorstárazást, illetve az adattovábbítást az FTP, a HTTP vagy a GOPHER protokollok alatt elérhető adatelemek számára. A webböngészők ezután a helyi Squid gyorstárkiszolgálót használhatják HTTP-proxynak, ezáltal mérsékelve az adatelérés idejét és a szükséges sávszélességet. A Squid az adatokat a memóriában vagy a merevlemezen tárolja.
A Squid kiszolgálókat többszintűen is lehet szervezni, hogy a központi kiszolgálók nagy adattárakat tegyenek elérhetővé az alacsonyabb szinten található kiszolgálók számára.
A SAS EMEA környezetében már régebben is használták a Squidet, és igen jól szerepelt. A program különlegesen megbízható és zökkenőmentes internetelérést nyújt az ügyfelek számára.
Az eredeti proxykiszolgálók 10.22-es HP-UX rendszert futtató HP munkaállomásokon a Squid 2.1 változatát használták. A rendszer sok különféle gépen futott, de többnyire 64 MB memóriával és 4 GB merevlemezzel rendelkező HP9000/720 munkaállomásokat használtak. Ennek az összeállításnak azonban elég nehézkes a támogatása.
A gépek sorra elérték azt a kort, amikor egymás után jöttek elő a hibáik, ezenkívül a növekvő internethasználat és az irodák bővülése miatti terhelésnövekedéshez már ezek a beállítások sem voltak megfelelőek. A legfőbb gond a lemezkezelés volt: a megnövekedett használat mellett a meglévő 100 MB-s naplóterületek és a 2 GB-os, tulajdonképpeni gyorstárkönyvtárak egyaránt szűkösnek bizonyultak.
Elkezdtünk valamilyen megoldás után kutatni, hogy fenntarthassuk a szolgáltatást. Mivel magával a Squid programmal elégedettek voltunk, és a beállításának rejtelmeiben is eléggé elmélyedtünk már, a Squid használata mellett döntöttünk, és a gépi környezet lecserélésének lehetőségeit kezdtük el vizsgálni. Mivel a Squid egy nyílt rendszer, és Linux alatt kedvező a támogatottsága, jó ötletnek tűnt, hogy a feladatot átlagos SAS EMEA Intel számítógépeken futó Linux-alapú alkalmazásokkal oldjuk meg.
Ehhez egy DELL asztali PC-t választottunk, 256 MB memóriával 500 MHz Intel Pentium processzorral, és belső, 20 GB IDE merevlemezzel. Mivel a DELL erősen kötődik a RedHathez, logikus választásnak ez a rendszer tűnt. Ráadásul a SAS nemrégiben a RedHat partnereként adott ki termékeket.

Szerkezet

A SAS EMEA eredeti szerkezete három központi „szülő” Squid-gyorstárat használt, közvetlen internetelléréssel, és „gyermek” Squid kiszolgálókat az országos irodákban. Néhány kisebb ország elérése közvetlenül a központi főhadiszállás kiszolgálóin keresztül történt, és úgy éreztük, az olcsóbb gépek lehetővé teszik majd, hogy ezekbe az irodákba is proxykiszolgálót telepítsünk. Továbbá, abban a néhány országban, ahol a SAS egymással WAN hálózaton keresztül összekapcsolt irodákkal volt jelen, az olcsóbb gépek lehetőséget teremtettek arra, hogy ide is proxykiszolgálókat helyezzünk el. Ezek a fejlesztések csökkentették a webforgalom válaszidejét, és mérsékelték a WAN hálózataink túlterheltségét.
Végül akadt néhány kifogásunk az eredeti szerkezet rugalmasságát és elérhetőségét illetően, és úgy éreztük, hogy az átalakított kiszolgáló és ügyfélrendszer beállításai növelik a belső felhasználóink számára nyújtott szolgáltatásaink színvonalát.
Az új szerkezet tulajdonképpen alapjaiban nem változott meg. Továbbra is három központi kiszolgálónk volt, csakhogy most Linux fut rajtuk. Több gyermekproxyt telepítettünk, és néhány irodában háromszintű rendszert építettünk ki. Például néhány országnak olyan műholdas irodái voltak, amelyek mindössze egyetlen, a főhadiszálláshoz kapcsolódó WAN hálózaton keresztül csatlakoztak a SAS belső hálózatához. Ebben az esetben proxykat telepítettünk a műholdas irodákba, amelyek lehetőleg a központi iroda helyett az országos központhoz kapcsolódtak.
A rendszerünk újítása volt még a Trend Interscan Virus Wall termék alkalmazása a HTTP-vírusok kiszűrésére. Három víruskereső rendszert telepítettünk szintén RedHat alá. Ezek a rendszerek a jelenlegi Squid szülő gyorstárak mögött helyezkedtek el, és egy vírusszűrő réteget képeztek a Squid gyorstár és a külső Internet között. Mivel a víruskeresők alapjában véve egyszerű folyamvizsgáló rendszerek, a legmagasabb szintű Squid kiszolgálókat állítottuk be úgy, hogy válogathassanak közöttük.

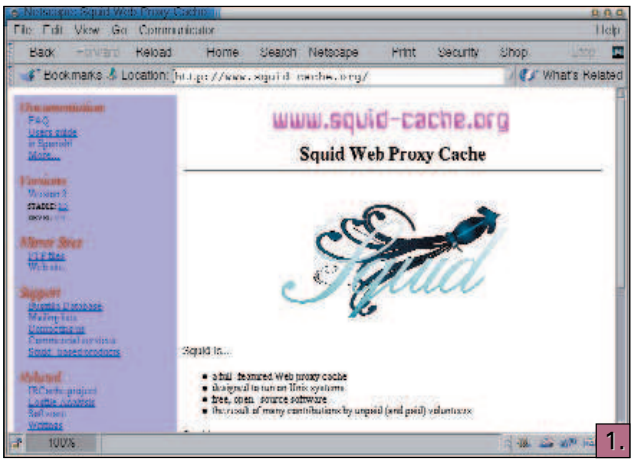
Telepítés

Az eredeti HP-UX kiszolgálók telepítése úgy történt, hogy egy ismert rendszer lemezének tartalmát sokszorosítottuk. Ez több szempontból sem volt kielégítő megoldás, nem beszélve arról, hogy igencsak nehéz volt gondoskodni a lenyomat karbantartásáról, ha valamelyik programot foltozni kellett, esetleg frissíteni szerettük volna a Squidet stb.
A célunk az volt, hogy parancsfájlokon alapuló önműködő telepítőrendszert hozzunk létre, amit a helyi iroda személyzete könnyedén és gyorsan végre tud hajtani. Az elgondolást végül sikerült kielégítően megvalósítani, ráadásul a rendszernek további előnyei is származtak a biztonsági helyreállítás és beállításkezelés terén.
A gépek felépítéséhez egy KickStart összeállítást készítettünk.
A KickStart a RedHat egyik önműködő telepítés-elősegítő eszköze. Alapesetben megadhatjuk a telepítőnek, miképpen ossza fel a merevlemezt, mely RPM-csomagokat telepítse, és végrehajthatunk néhány helyi rendszerbeállító héjparancsot is.
Mi a KickStart-beállítónkat egy hajlékonylemezre írtuk a szokásos RedHat indítóeszközökkel együtt, és úgy állítottuk be, hogy a CD-ről telepítsen.

Tehát a proxykiszolgálóhoz olyan PC-t terveztünk, ami hasonlít az általunk elvárt összeállításhoz, majd leszállítottuk a CD-t és a hajlékonylemezt a távoli irodának, ahol a telepítést elvégezhetik.
A telepítési folyamat majdnem teljesen önműködő, mindössze három adatot kell megadni: a gépnevet, az IP-adatokat és a billentyűzet típusát, ugyanis néhány irodánkban az adott nyelvnek megfelelő billentyűzetet használnak. A KickStartban minden más beállítás előre rögzített. Például a telepítés nyelve mindig angol, a kiválasztott csomagok mindig ugyanazok, és a merevlemez is mindig ugyanúgy lesz felosztva. A teljes telepítés a lemez behelyezésétől kezdve a frissen települt operációs rendszer elindulásáig még tíz percet sem vesz igénybe. Ez sokkal gyorsabb, mint ahogy kézzel megoldhatnánk, és jóval kevesebb időt vesz igénybe, mint egy HP-UX telepítése. Nyilvánvaló, hogy ennek van néhány mellékhatása a biztonsági mentések és helyreállítás terén is.

Karbantartás

Rendszerünk futtatása során a szokásos gondokkal kerülünk szembe: a beállítási fájlokat folyamatosan frissíteni, a programokat újabb változatokra cserélni, a naplófájlokat forgatni, a folyamatokat pedig felügyelni kell. A múltban ezeket a feladatokat héjprogramok és cron-feladatok átláthatatlan kavalkádja oldotta meg, sőt, még azt is elnéztük, ha a rendszerek különböző felépítésűek voltak. Első körben ezeket a feladatokat az ***rdist*** csomaggal oldottuk meg, de ez sem volt kielégítő, így tovább kerestünk.
A legmegfelelőbbnek a cfengine tűnt. Ez a program lehetővé teszi számunkra, hogy egy központi helyen létrehozzuk az általános eszközmeghatározásokat és összeállításokat, ahonnan aztán szétoszthatjuk azokat az összes kiszolgálóra. A ***cfengine*** alkalmazása messzemenően sikeresnek bizonyult, annak ellenére, hogy a beállítások gondos megtervezését és a csomaghoz tartozó leírások alapos áttanulmányozását igényelte.
A szétosztani kívánt állományok közt néhány teljesen szokványos is akadt, ezeket egyszerűen átküldhettük a cfengine-nel, úgy ahogy voltak. Mások azonban, bár valamilyen általános sémán alapultak, minden rendszeren egyedi módosításokat igényeltek. Jó példa erre a Squid fő beállításfájlja. Az ilyen esetekben cfengine-nel áttöltöttük a mintaállományt és egy kis parancsfájlt, ami tudta, miképpen kell elvégezni a módosításokat. Kihasználva a cfengine képességét, hogy áttöltés után parancsokat képes végrehajtani, lefuttattuk ezt a parancsfájlt, majd jeleztük a Squidnek, hogy újra töltse be a beállításfájljait. Így egy központilag irányított és összehangolt beállítórendszert tudtunk létrehozni.
A cfengine-rendszer az általunk karbantartott helyi NIS-térképre épült. Ez a NIS-térkép egyszerűen összegyűjtötte a gépneveket, és különböző tulajdonságokat rendelt hozzájuk. Például a SQUID-CHILD kulcsszóval a második szinten elhelyezkedő gépeket jelöltük meg, melyek egy SQUID-MASTER-hez kell kapcsolódjanak. Ezt a NIS-térképet dolgoztuk fel a cfengine-hez szükséges osztályok előállításához, így végül a beállításadatokat központilag tároltuk, és nem az egyes kiszolgálókon.
Több gondot okozott viszont a telepített programok karbantartása. Nagyjából RedHat 6.2-alapú rendszert futattunk, de mialatt a proxykat telepítettük, a RedHat kiadott néhány frissítést, amire szükségünk volt. Főként a biztonsági javításoknak van számunkra jelentős szerepe. Ezenkívül van néhány általunk készített RPM-csomag is, ráadásul újabb Squid-változatot használunk, ami tartalmaz pár olyan lehetőséget, amit a RedHat még nem támogat. Az általunk használt útválasztó démon is különleges, néhány olyan útválasztó protokollt is ismer, mely az alapcsomagban nem található meg. Végül, van néhány olyan RPM-csomagunk is, ami soha nem is volt a RedHatben.
Magától értetődő lépés volt egy FTP-kiszolgáló létrehozása. A Network Appliance Fileren futtattuk a beépített FTP-szolgáltatást, mely a RedHat-változatokat is tartalmazta. Van egy tükrözőnk, ami mindig letölti a legfrissebb javításokat a RedHat tükörhelyről. Az FTP-kiszolgálón egy külön fában tároljuk a saját RPM-csomagjainkat.
Jó időbe tellett, mire sikerült összehozni a csomagfrissítő rendszert, de még mindig nem voltunk teljesen elégedettek. A legjobb eszköz, amit találtunk, az ***autorpm*** volt. Ezt az eszközt be tudtuk úgy állítani, hogy önműködően vizsgálja az FTP-kiszolgálónkat, és folyamatosan telepítsen minden RedHat-frissítést, viszont hagyja figyelmen kívül mindazokat az RPM-eket, melyeket nem kértünk első telepítéskor. Emellett beállítottuk, hogy az összes saját RPM-csomagunkat is folyamatosan frissítse.



1.

Egy kis gondunk azért akadt, mivel az autorpm nem képes önműködően kezelni az előforduló körkörös függőségeket. Igaz, úgy tűnik, ez inkább az adott RPM-ek hibája, nem pedig az autorpm hiányossága.

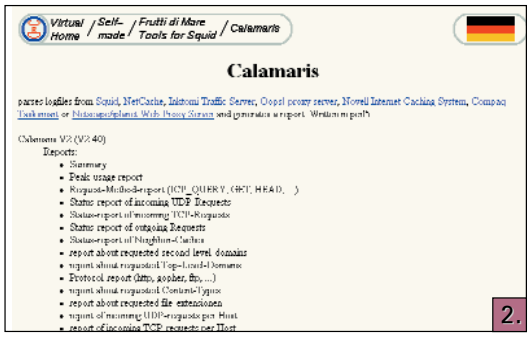
Vissza a telepítéshez

Ez a gond a csomagokkal a telepítésnél is visszaköszönt. Nagyon örültünk neki, hogy a telepítési időt tíz perc alá sikerült szorítanunk, és mindössze néhány percet vett igénybe a régi rdist fájlok átalakítása végrehajtása is. Most azonban néhány gépen több mint tíz percet vett igénybe az új csomagok telepítése, és emberi közbeavatkozás is szükségessé vált a beállítások befejezéséhez. Ez különösen a Squid esetében volt nehéz számunkra, ahol a saját beállítófájlunknak egy újabb Squid-változatra volt szüksége.
Magánál a telepítésnél is felmerült ugyanez a nehézség. Nem volt arra lehetőség, hogy a közbeavatkozásunk nélkül fejeződjön be a helyi telepítés, amennyiben frissítettük valamelyik programot. Ez nagyon fontos kérdés volt számunkra. Előfordult néhányszor, hogy valamilyen hibát tapasztaltak a távoli irodában és mi nem voltunk elérhetők. Ezekben az esetekben távolról szerettük volna elérni az irodát, és egyszerűen újratelepíteni a proxykiszolgálójukat, esetleg egy új gépkiépítésen. Ez viszont csak akkor működik, ha nem kell beavatkoznunk a folyamatba.
Végül visszatértünk a kezdetekhez, és újra megfontoltuk az egyik alapfeltételezésünket – ugye azt mondtuk, hogy egyszerű RedHat CD-ket szállítottunk, majd KickStarttal, cfengine-nel, és autorpmmel hangoltuk be a rendszert. Most úgy határoztunk, inkább elkészítjük a saját Linux-változatunkat, mely RedHat alapokon nyugszik, de tartalmazza az általunk készített új vagy felfrissített RPM-csomagokat is, és néhány beállításfájlt. Az alapötlet az volt, hogy a kezdeti telepítés hozzon létre egy működő proxykiszolgálót, és később az önműködő időzített héjprogramok elvégzik a szükséges javításokat. A saját változatunk meglehetősen sikeresnek bizonyult és egyik napról a másikra sikerült felépítenünk új RedHat kiadásból. Vettük az alap-RedHatet és számos RPM-et törültünk a fából. Természetesen ez nem volt különösebben tudományos alaposságú munka, nem töröl-

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

tünk minden RPM-et, csak azokat, amelyek a legtöbb helyet foglalták el. Ezután a fába illesztettük a saját csomagjainkat, átalakítottunk pár vezérlőszerkezetet, és már írtuk is a CD-t.
Mivel az új kiadványunk tartalmazta a cfengine-t és az autorpmet, beállíthattuk a KickStart utótelepítést úgy, hogy futtassa le ezeket a folyamatokat a telepítést követő első újraindítás alkalmával. Ezáltal az új gép gyorsan naprakésszé válhatott az új beállításoknak megfelelően. Csakhogy mikor átálltunk a RedHat 6.2-re, egy KickStarttal kapcsolatos érdekes dologba ütköztünk. Az új változat CD-ről telepítése közben nem feltétlenül kérdezi meg az IP-címet és más hálózati beállításokat a felhasználótól, amennyiben azok nincsenek jelen a KickStart beállítófájljában. A kissé megváltoztatott leírás figyelmes átolvasása után megtudhattuk, hogy ez a szabadság növelésért történt, nekünk viszont komoly fejfájást okozott. Végül átírtunk egy részt az Anaconda telepítőben, így visszaállítottuk az eredeti viselkedést.



Biztonsági mentés és helyreállítás

Több különféle mentési és helyreállítási módszert is számításba vettünk, végül úgy döntöttünk, hogy a legegyszerűbbet választjuk, vagyis nem készítünk mentést. Mikor megnéztük a proxykat, észrevettük, hogy csak a naplóadatok és gyorstárszerkezetek különböznek az egyes gépeken. A naplóadatok időszakonként egy központi helyre másolódtak további vizsgálatok végzéséhez, a gyorstár pedig, bár értékes, időnként nyugodtan kitörölhető és újra létrehozható.
Az általunk alkalmazott linuxos proxyknak mostanáig nem volt semmilyen nagyobb szolgáltatáskihagyása vagy alkatrészhibája, de ha ilyesmi történik, a rendszer újratelepítését javasoljuk. Alkatrészhiba esetén bízunk benne, hogy minden irodában akad egy tartalék PC, amin a telepítést meg lehet ismételni.
Végeredményképpen nemcsak hogy nem kellett mentéseket végeznünk, de rugalmas alkatrészkészletről vagy tükröző módszerekről sem kellett gondoskodnunk. Tulajdonképpen a különleges alkatrészek használata csökkentené a rugalmasságunkat, hiszen nem biztos, hogy lenne tartalék alkatrész a távoli irodában.
Úgy számítjuk, hogy leállás esetén a távoli iroda megközelítőleg húsz perc alatt helyre tudja állítani a szolgáltatást, feltéve, hogy van a közelben egy felhasználható PC. Megpróbálkoztunk a böngészők beállításával is, hogy az egész folyamatot átlátszóvá tegyük az asztali PC-t működtető felhasználók számára. Ez egy különösen jól kihasználható megoldást adott a kezünkbe.

Ügyféloldali megfontolások

A régi proxybeállítás abban bízott, hogy az ügyfél közvetlen módon hivatkozik a névvel azonosított proxykiszolgálóra. Azokon a helyeken, ahol egynél több proxykiszolgáló futott, a proxykiszolgálóként használt gépnév egy ***lbnamed*** által kezelt tartományban volt. A mi változatunkat kissé módosítottuk, de még így sem felelt meg maradéktalanul az igényeinknek. Az lbnamed az rpc.rstatd-t használja, hogy a listában található gépek terheltségadatait meghatározza, majd pedig a súly függvényében különféle gépneveket ad vissza, ezáltal biztosítva egy korlátozott képességű terheléselosztást. A gyakorlatban azonban többnyire mégsem osztja el hatékonyan a terhelést, annak ellenére, hogy tartalmaz egy olyan hasznos tulajdonságot, miszerint a nem üzemelő gépeket törli a listából. Sajnos, ennek a hasznos szolgáltatásnak sokat levon az értékéből az a tény, hogy a gépek súlyozásánál csak a terhelést (load) veszi figyelembe (az alapváltozatban). Ha a Squid kiszolgáló leáll, a gép terheltsége általában csökken, ezáltal a meghibásodott gép a lista tetejére kerül. C-ben létezik ugyan egy módszer arra, hogy a terhelésen kívül néhány más értéket is lekérdezzünk, de az ilyen irányú próbálkozásaink nem bizonyultak túl gyümölcsözőnek. Amikor telepítettük, az általános benyomásunk az volt, hogy valamivel azért sikeresebb, mint amilyen az egyszerű DNS-címforgatás lett volna.
A kipróbálást a három új Linux-kiszolgálónkon hajtottuk végre, és az egyik vizsgált tényező az volt, hogy a válaszidőnek csökkennie illene, ha egy olyan adatra küldünk http-kérést, ami nagy valószínűséggel megtalálható az adott proxy gyorstárában. A lekérdezéseket kezelő proxy-gyorstárak értelmes kiválasztásának ötlete igen könnyen megvalósítható, a legtöbb böngésző által támogatott Proxy Auto Configuration (PAC) fájl segítségével. Ez a lehetőség azon alapul, hogy létezik valahol egy olyan proxykiszolgáló, ami képes PAC-fájlokat visszaküldeni.
A mi feladatunk teljesen egyszerűvé vált, mivel valaki már készített egy példakódot a PAC-fájlokhoz, ami kiegyensúlyozta a terhelést néhány proxykiszolgáló közt. A Sharpnál készült Super Proxy Script nevű munkát vettük alapul, amely URL-kategorizálást használt. Ez egy nagyon egyszerű, de eredeti ötlet, ami elemzi az elérni kívánt URL-t, majd visszaadja a használandó proxy nevét. Ennek segítségével egyenletesen osztja el a terhelést a kiszolgálók közt, de azonos URL-re vonatkozó lekérdezések mindig ugyanazt a proxyt használják. Kihasználtuk a PAC-fájlok azon tulajdonságát is, hogy proxykiszolgálók listáját is vissza tudják adni. Ennek eredményeképpen, ha a listában elsőként szereplő kiszolgáló nem válaszol, a böngésző önműködően a következőt fogja használni, a felhasználó számára észrevétlenül.
A központ gépei esetében két vagy három proxyt tartalmazó listákat adtunk vissza a telepen belüli helyzetüktől függően. Azokon a helyeken, ahol csak egyetlen proxy volt elérhető, nem használtunk URL-elemzést, és csak két proxynevet adtunk vissza, a helyi kiszolgáló nevét, valamint hiba esetére, a központi kiszolgálóét. A központi kiszolgáló használata hiba esetén lehetővé tette számunkra a legnagyobb rugalmasságot. Ha a helyi proxy le is állna, a hozzá tartozó böngészők észrevennék a hibát, és a központi kiszolgálót vennék igénybe. Az ügyfelek 30 percenként ellenőrzik (MSIE és Netscape esetén), hogy helyreállt-e már a kapcsolat az eredeti kiszolgálóval, és ha így van, akkor visszatérnek a használatához.
Mivel néhány ügyfelünk Microsoft Explorer 5.0-t használ, a proxy.pac fájlt wpad.dat névre neveztük át. Így lehetővé vált, hogy a „beállítatlan” IE5-ügyfelek egy http://wpad.helyi.tartomány/wpad.dat formátumú URL-re történő WPAD rendszerű keresés használatával önműködően megtalálják a wpad.dat fájlt. A WPAD használata nem volt feltétlenül szükséges, de hasznosnak találtuk. A naplófájljaink elemzése rámutatott arra, hogy segítségnyújtó irodánkat mentesíthetjük jó néhány vándorló felhasználó hívásától, akik máskülönben segítséget kértek volna a kézi proxybeállítás elvégzéséhez, amikor másik irodába mentek át.
Jelenleg Apache webkiszolgálót használunk a PAC-fájlok visszaküldéséhez, és az Apache futhat a helyi proxykiszolgálón, vagy bármely más helyileg elérhető webkiszolgálón. Lehetséges, hogy egy lecsupaszított webkiszolgáló használata – ilyeneket Perlbe is átültettek – biztonságosabb lenne, és kevesebb fejfájást okozhatna. Ezt a megközelítést még nem vizsgáltuk meg.
A WPAD kiszolgálóink jelenleg több DNS-bejegyzéssel is rendelkeznek, így ha valamelyik WPAD kiszolgáló elromlana, még mindig rugalmasak maradhatunk. Azokon a helyeken, ahol csak egyetlen WPAD kiszolgálót alkalmazunk, ott az új ügyféloldali folyamatra

bízzuk, hogy az előzőleg gyorstárazott beállításokat használja fel. Van viszont egy kis hiba ebben a megközelítésben, abban az esetben, ha a távoli helynek több proxykiszolgálója is van: az URL-elemzés ugyan gondosan kiválasztja a megfelelő helyi proxykiszolgálót, az viszont egyszerűen körbeküldi a lekérdezést a három központi rendszeren. Kihasználhatnánk a Squid néhány képességét a gyorstár-összesítők (digest) egymás közti cseréjére, s így a helyi proxy arra a központi kiszolgálóra küldené az adatot, ahol a legnagyobb valószínűséggel érne el találatot, de a gyakorlati tapasztalatok azt mutatják, hogy a WAN hálózat sávszélesség-felhasználásának növekedése miatt ez nem igazán hatékony megoldás. Ehelyett hagyjuk végrehajtani a körbekérdezést a központi kiszolgálókon, és a központi kiszolgálók közötti gyorstár-összesítők cserével érjük el a találatot, amennyiben lehetséges.

Fejlesztési lehetőségek

Említettük már, hogy a proxykiszolgálóinkon lehetőleg egy viszonylag korszerű RedHat-változatot szerettünk volna fenntartani. Bár a kisebb beállítás-változtatásokat a cfengine-re és az autorpm-re bíztuk, nem hisszük, hogy a teljes OS hálózaton keresztül történő frissítése megvalósítható lenne.

Ehelyett inkább új telepítések szétküldését terveztük a távoli irodák számára, ahol azokat igény szerint feltelepíthetik. Úgy számoltuk, évente megközelítőleg háromszor-négyszer fordul elő ilyen eset. Mivel ilyen tiszta és irányított telepítési eljárásunk van, viszonylag kevés hátránnyal jár a gyakori frissítés. Mivel a távoli iroda megtarthatja az előző változat anyagát, az előző változatra való visszatérés sem okoz túl nagy nehézséget.

Nagyon fontosnak tartottuk ezeket a frissítéseket, mivel igen sok hibát tapasztaltunk a korábbi HP-UX proxyk működése során, amelyek a programfrissítések és javítások hiányából eredtek. Például láttunk HP-UX gondokat, pedig ezekre lettek volna foltok, és a Squid és pár eszköz régebbi változatát futtattuk, mivel csak régebbi Perl-változatunk volt.

Az új változatra átállás akár rendes munkanapon is történhet a távoli irodában, az ehhez szükséges húsz percre az ügyfelek egyszerűen a központi hibaelhárító kiszolgálókat fogják használni; olyan irodák esetén, ahol több kiszolgáló is van, a saját helyi hibaelhárító kiszolgálóikat használhatják.

Rendszerfigyelés

Elemzés szempontjából a leghasznosabb adat a gyorstár tevékenységéről készített kimutatás. Ezt az adatot az MRTG (Multi Router Traffic Grapher) és a Squidbe épített SNMP-ügyfél használatával érjük el. Kicsit kényelmetlen ugyan beállítani, de hasznos ábrákat tudunk vele készíteni a proxyk teljesítményéről. Sok számadatot gyűjtünk be a proxykról, ami a belső weboldalról elérhető. Van olyan kódunk is, amely áttekintőlapokat készít az összes kiszolgálóról. A kiszolgálóink NIS-térképén végigmenve ez az eljárás gyors átnézeti képet is szolgáltat néhány számadatról. Minket különösen a gyorstártalálatok és tévesztések folyamata érdekel, illetve az összesített kérelmek növekedése.

Pillanatfelvételek előállítására és adatelemzésére a Calamaris eszközt (lásd a Kapcsolódó címeket) is használjuk.

Van egy másolatunk a HP OpenView hálózatkezelő termékekből is. Jelenleg csak a gépek állapotának figyelésére használjuk, de tervezzük a távoli gépeken futó Squid-programok felügyeletét is, hogy figyelmeztessen bennünket, ha bármelyik meghibásodna.

Elkezdtük cronból ötpercenként futtatott cfengine-t használni arra, hogy ellenőrizzük a különféle folyamatok állapotát, és szükség esetén megkíséreljük az önműködő újraindítást.

Továbbá a helyi irodákhoz internethasználati jelentéseket továbbítunk. Jelenleg a naplófájlokat a SAS adathalmazba gyűjtjük, és a SAS jelentéskészítő eszközt használjuk fel a jelentésekhez, van ezenkívül pár olyan termékünk is, mint amilyen a SAS IT Service Vision vagy a SAS WebHound, amelyek hasonló forgalomelemzésre képesek. Ezek nagyon hatékony eszközök, és az adatok sokkal teljesebb vizsgálatát teszik lehetővé. Elégedettek vagyunk a linuxos megoldásunk megbízhatóságával és teljesítményével.

Semmi kétségem afelől, hogy az alkatrészárak csökkentésével lehetőség nyílik arra, hogy olyan helyekre is telepítsünk proxykiszolgálókat, ahol eddig nem volt gazdaságos nagyobb rugalmasságot nyújtani a többi hely felé. Mivel az új összeállításunk rugalmasabb, és mindent egybevetve jobban beállított, mint az előző volt, sokkal kevesebb gondunk van most, mint a régi proxyrendszerekkel volt.

Az eddigi leggonoszabb hiba, amit a működés során tapasztaltunk, a fájlrendszerhiba volt. Az egyik távoli proxy áramkimaradás miatt leállt, és nem tudott elindulni fájlrendszerhiba miatt. Ideiglenes megoldásként átalakítottuk az indítókódot, hogy legyen elnézőbb az ilyen rendszerleállásokkal szemben, hosszú távú megoldás viszont inkább valamilyen jegyzőkönyvező fájlrendszer használata lehetne, amilyenek mostanság kezdenek hozzáférhetővé válni.

Azt vettük észre, hogy egyre több helyi kapcsolat épült ISP-szolgáltatók felé, ezért majd csiszolnunk kell az összeállításunkon, hogy támogassa a HTTP-alapú víruskeresést irodaszinten és az internet-proxy közvetlen kapcsolatán egyaránt. Kiegészítjük a cfconfig és autorpm eszközöket néhány új feladattal, hogy telepítsék és állítsák be az új részegységeket.

Mivel most már van egy receptünk a saját Linux-kiadásaink elkészítéséhez, valószínűleg elkezdünk foglalkozni egy általánosabb, belső felépítés gondolatával. Ez valamilyen szinten biztosan megtörténik, most, hogy a SAS-termékek már Linuxon is elérhetők. Azt reméljük, hogy a belső használatra szánt szabványos kiadások készítése bizonyos beállításbeli szabadságot ad majd. Valószínűleg átgondoljuk azt is, milyen más szolgáltatásokat lehetne haszonnal futtatni a Linuxon. Például átrakhatnánk néhány DNS/BIND szolgáltatást Linux alá.

Ian Spare
jelenleg tanácsadóként dolgozik a németországi SAS Intézetnél. Idejét leginkább hódeszkázással vagy sílécen szereti tölteni, de amikor éppen nem a havon tartózkodik, tanácsokat ad, vagy Unix- és Linux-rendszereket felügyel szerte Európában, a Közel-Keleten és Afrikában. Nincsenek gyermekei, viszont van három macskája.

Kapcsolódó címek

cfengine
➲ http://www.gnu.org/software/cfengine

Squid
➲ http://www.squid-cache.org *(1.kép)*

autorpm
➲ http://www.kaybee.org/~kirk/html/linux.html

lbnamed
➲ http://www.stanford.edu/~riepel/lbnamed

Proxy Auto Configuration
➲ http://home.netscape.com/eng/mozilla/2.0/relnotes/demo/proxy-live.html

Sharp „Super Proxy" Script
➲ http://naragw.sharp.co.jp/sps/index.html

Calamaris Squid Reporting Tool
➲ http://cord.de/tools/squid/calamaris *(2. kép)*

© Kiskapu Kft. Minden jog fenntartva